May 13-17, 2019
San Francisco, CA, USA



Association for Computing Machinery

Advancing Computing as a Science & Profession



THE WEB CONFERENCE

The Web Conference 2019

Proceedings of

The World Wide Web Conference WWW 2019

Editors:

Ling Liu (Georgia Tech, USA)
Ryen White (Microsoft Research, USA)
Amin Mantrach (Criteo AI Lab, France)
Fabrizio Silvestri (Facebook, UK)
Julian McAuley (University of California at San Diego, USA)
Ricardo Baeza-Yates (NTENT & Northeastern University, USA)
Leila Zia (Wikimedia Foundation, USA)



THE WEB CONFERENCE

IW3C2

Copyright © 2019 by IW3C2 (International World Wide Web Conference Committee). The papers are published under the Creative Commons Attribution 4.0 International (CC BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution. In case of republication, reuse, etc., the following attribution should be used: "Published in WWW2019 Proceedings © 2019 International World Wide Web Conference Committee, published under Creative Commons CC By 4.0 License."

ISBN: 978-1-4503-6674-8

Printed in the USA

The Web Conference 2019
Conference Chairs' Welcome

It is our great pleasure to welcome you to *The Web Conference 2019*. The Web Conference is the premier venue focused on understanding the current state and the evolution of the Web through the lens of computer science, computational social science, economics, policy, and many other disciplines. The 2019 edition of the conference is a reflection point as we celebrate the 30th anniversary of the Web.

The technical program of this year's edition of the conference consists of three keynotes (Jeff Dean, Lawrence Lessig and Claire Wardle), one panel to celebrate the 30 years of the Web, 12 tracks for refereed research papers (including full papers and short papers), and a demonstrations track that are published in the main volume of the proceedings. The companion volume of the proceedings includes 23 workshops, most of the 22 tutorials, and the Ph.D. Consortium. We also have several other co-located events and special tracks: W4A, BIG, The Web of Health, The Open Web, Building an Ethical Web, and W3C.

We are thankful to all the people that helped us organizing the conference and that are listed in the next section. Without them, this would have never happened. Special thanks to all our sponsors, as they make the quality difference: Microsoft (Diamond); Amazon, Bloomberg and Google (Gold); Criteo AI Lab, CISCO, NTENT, Spotify, Yahoo Research and Wikimedia Foundation (Silver); Baidu, DiDi, eBay, Facebook, LinkedIn, Megagon Labs, Mix, Mozilla, Netflix Research, Northeastern University, Pinterest, Quora, Visa Research and Walmart Labs (Bronze); as well as Airbnb, Letgo, Moore Foundation and Webcastor.

Finally, this conference would have not been possible if together with Ashish Goel and Ben Steichen we did not start Web4Good, a California based non-profit company that undersigned the event with IW3C2.

So we hope that you enjoy this week as well as San Francisco!

Ricardo Baeza-Yates
General Co-Chair
NTENT & Northeastern University at SV
USA

Leila Zia
General Co-Chair
Wikimedia Foundation
USA



The Web Conference 2019
Research Chairs' Preface

The research program of this year's edition consists of 12 tracks for refereed research papers (including full papers and short papers), and a demonstrations track. As in previous years, updates were made to ensure that track topics correctly reflect the key research areas of interest to the Web community.

The research tracks attracted submissions from 5042 distinct authors from 53 countries spanning six continents. Submissions were rigorously reviewed by a program committee (PC) comprising 27 track chairs, 69 senior PC members (on three of the tracks), 1112 PC members, and 594 external reviewers. A total of 5280 reviews were provided, with each paper receiving at least three reviews, and many papers receiving four or more reviews.

During the discussion phase, PC members, senior PC members, and track chairs discussed the reviews in detail. Meta-reviews were provided for most submissions, especially for borderline papers and high-scoring rejects, summarizing the reviews and discussion. For consistency across tracks, final decisions on papers were made at the face-to-face PC meeting between track chairs and the PC chairs. We accepted 225 (18%) of the 1247 full papers and 72 (20%) of the 361 short papers submitted to the research tracks. We also accepted a small number of full papers as short papers. New this year, all full and short papers were presented as a poster; some full papers also had an oral presentation.

Additionally, 27 submissions were selected for the demonstrations track after being reviewed by 59 PC members, with each of them reviewing an average of 3.95 papers. The majority of the demos got reviewed by 4 reviewers with a minimum of 3 reviewers per submission.

Ling Liu
Program Co-Chair
Georgia Tech University
USA

Ryen W. White
Program Co-Chair
Microsoft Research
USA

Amin Mantrach
Demonstrations Co-Chair
Criteo AI Lab
USA

Fabrizio Silvestri
Demonstrations Co-Chair
Facebook
UK

Julian McAuley
Proceedings Chair
University of California
at San Diego, USA



THE WEB CONFERENCE

Table of Contents

The Web Conference Organization xxix

Research Track Organization xxx

Senior Program Committee xxxi

Keynote Talk

- **Deep Learning for Solving Important Problems** 1
  Jeff Dean *(Google Mountain View, USA)*
- **The Law of the Horse at 20: Phases of the Net** 2
  Lawrence Lessig *(Harvard Law School, USA)*
- **Enlisting the Public to Build a Healthier Web Information Commons** 3
  Claire Wardle *(TED Fellow CIVIC, USA)*

Full Paper

- **Addressing Trust Bias for Unbiased Learning-to-Rank** 4
  Aman Agarwal *(Cornell, USA)*, Xuanhui Wang, Cheng Li, Michael Bendersky, Marc Najork *(Google, USA)*
- **Learning Edge Properties in Graphs from Path Aggregations** 15
  Rakshit Agrawal *(University of California, USA)*, Luca de Alfaro *(UC Santa Cruz, USA)*
- **Evaluating User Actions as a Proxy for Email Significance** 26
  Tarfah Alrashed *(Massachusetts Institute of Technology, USA)*, Chia-Jung Lee *(Microsoft, USA)*, Peter Bailey *(Microsoft, Australia)*, Christopher Lin, Milad Shokouhi, Susan Dumais *(Microsoft, USA)*
- **DDGK: Learning Graph Representations for Deep Divergence Graph Kernels** 37
  Rami Al-Rfou, Bryan Perozzi, Dustin Zelle *(Google AI, USA)*
- **Stereotypical Bias Removal for Hate Speech Detection Task using Knowledge-based Generalizations** 49
  Pinkesh Badjatiya, Manish Gupta, Vasudeva Varma *(IIIT Hyderabad, India)*
- **Personalized Bundle List Recommendation** 60
  Jinze Bai *(Peking University, China)*, Chang Zhou *(Alibaba Group, China)*, Junshuai Song, Xiaoru Qu *(Peking University, China)*, Weiting An, Zhao Li *(Alibaba Group, China)*, Jun Gao *(Peking University, China)*
- **Improving Medical Code Prediction from Clinical Text via Incorporating Online Knowledge Sources** 72
  Tian Bai, Slobodan Vucetic *(Temple University, USA)*
- **No Place to Hide: Catching Fraudulent Entities in Tensors** 83
  Ban Yikun *(Peking University, China)*, Liu Xin *(Tsinghua University, China)*, Huang Ling *(Fintec.ai, China)*, Duan Yitao *(Netease Youdao Inc., China)*, Liu Xue *(McGill University, Canada)*, Xu Wei *(Tsinghua University, China)*
- **Link Prediction in Networks with Core-Fringe Data** 94
  Austin Benson, Jon Kleinberg *(Cornell University, USA)*
- **Spiders like Onions: on the Network of Tor Hidden Services** 105
  Massimo Bernaschi, Alessandro Celestini, Stefano Guarino, Flavio Lombardi, Enrico Mastrostefano *(Istituto per le Applicazioni del Calcolo "Mauro Picone" (IAC - CNR), Italy)*
- **Be Concise and Precise: Synthesizing Open-Domain Entity Descriptions from Facts** 116
  Rajarshi Bhowmik, Gerard de Melo *(Rutgers University, USA)*
- **Navigating the Maze of Wikidata Query Logs** 127
  Angela Bonifati *(Lyon 1 University, France)*, Wim Martens, Thomas Timm *(University of Bayreuth, Germany)*
- **What happened? The Spread of Fake News Publisher Content During the 2016 U.S. Presidential Election** 139
  Ceren Budak *(University of Michigan School of Information, USA)*

- **Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences** 151
  Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, Tat-Seng Chua *(National University of Singapore, Singapore)*
- **Enriching News Articles with Related Search Queries** 162
  David Carmel *(Amazon Research, Israel)*, Yaroslav Fyodorov *(Yahoo Research, Israel)*, Saar Kuzi *(University of Illinois at Urbana-Champaign, USA)*, Avihai Mejer, Fiana Raiber *(Yahoo Research, Israel)*, Elad Rainshmidt *(Oath, Israel)*
- **Revisiting Opinion Dynamics with Varying Susceptibility to Persuasion via Non-Convex Local Search** 173
  T-H. Hubert Chan, Zhibin Liang *(The University of Hong Kong, Hong Kong)*, Mauro Sozio *(Télécom ParisTech, France)*
- **Trajectories of Blocked Community Members: Redemption, Recidivism and Departure** 184
  Jonathan Chang, Cristian Danescu-Niculescu-Mizil *(Cornell University, USA)*
- **Selling a Single Item with Negative Externalities** 196
  Matheus Xavier Ferreira, S. Matthew Weinberg, Danny Yuxing Huang, Nick Feamster, Tithi Chattopadhyay *(Princeton University, USA)*
- **Revisiting Mobile Advertising Threats with MAdLife** 207
  Gong Chen *(Georgia Institute of Technology, USA)*, Wei Meng *(The Chinese University of Hong Kong, Hong Kong)*, John Copeland *(Georgia Institute of Technology, USA)*
- **Modeling Relational Drug-Target-Disease Interactions via Tensor Factorization with Multiple Web Sources** 218
  Huiyuan Chen, Jing Li *(Case Western Reserve University, USA)*
- **SamWalker: Social Recommendation with Informative Sampling Strategy** 228
  Jiawei Chen, Can Wang *(Zhejiang University, China)*, Sheng Zhou *(ZheJiangUniversity, China)*, Qihao Shi, Yan Feng, Chun Chen *(Zhejiang University, China)*
- **How Serendipity Improves User Satisfaction with Recommendations? A Large-Scale User Evaluation** 240
  Li Chen *(Hong Kong Baptist University, Hong Kong)*, Yonghua Yang *(Alibaba Group, China)*, Ningxia Wang *(Hong Kong Baptist University, Hong Kong)*, Keping Yang *(Alibaba Group, China)*, Quan Yuan *(inspirAI Co. Ltd., China)*
- **Emoji-Powered Representation Learning for Cross-Lingual Sentiment Classification** 251
  Zhenpeng Chen *(Peking University, China)*, Sheng Shen *(University of California, USA)*, Ziniu Hu *(University of California, USA)*, Xuan Lu *(Peking University, China)*, Qiaozhu Mei *(University of Michigan, USA)*, Xuanzhe Liu *(Peking University, China)*
- **Decoupled Smoothing on Graphs** 263
  Alex Chin, Yatong Chen, Kristen M. Altenburger, Johan Ugander *(Stanford University, USA)*
- **Cross-Network Embedding for Multi-Network Alignment** 273
  Xiaokai Chu, Xinxin Fan, Di Yao, Zhihua Zhu, Jianhui Huang, Jingping Bi *(Institute of computing, Chinese Academy of Sciences, China)*
- **Improving Treatment Effect Estimators Through Experiment Splitting** 285
  Dominic Coey, Tom Cunningham *(Facebook, USA)*
- **A Semi-Supervised Active-learning Truth Estimator for Social Networks** 296
  Hang Cui *(University of Illinois, USA)*, Tarek Abdelzaher *(University of Illinois at Urbana-Champaign, USA)*, Lance Kaplan *(Army Research Laboratories, USA)*
- **Dressing as a Whole: Outfit Compatibility Learning Based on Node-wise Graph Neural Networks** 307
  Zeyu Cui *(Institute of Automation, Chinese Academy of Sciences, China)*, Zekun Li *(Institute of Information Engineering, Chinese Academy of Sciences, China)*, Shu Wu *(Institute of Automation, Chinese Academy of Sciences, China)*, Xiao-Yu Zhang *(Institute of Information Engineering, Chinese Academy of Sciences, China)*, Liang Wang *(Institute of Automation, Chinese Academy of Sciences, China)*
- **Are All Successful Communities Alike? Characterizing and Predicting the Success of Online Communities** 318
  Tiago Cunha, David Jurgens *(University of Michigan, USA)*, Chenhao Tan *(University of Colorado Boulder, USA)*, Daniel Romero *(University of Michigan, USA)*

- **Adversarial Training Methods for Network Embedding**.....329
  Quanyu Dai, Xiao Shen *(The Hong Kong Polytechnic University, China)*, Liang Zhang *(JD.com, China)*, Qiang Li *(Y-Lab, Kwai, China)*, Dan Wang *(The Hong Kong Polytechnic University, China)*
- **Local Matching Networks for Engineering Diagram Search**.....340
  Zhuyun Dai *(Carnegie Mellon University, USA)*, Zhen Fan *(Tsinghua University, China)*, Hafeezul Rahman, Jamie Callan *(Carnegie Mellon Univeristy, USA)*
- **Evaluating Anti-Fingerprinting Privacy Enhancing Technologies**.....351
  Amit Datta *(Snap, Inc., USA)*, Jianan Lu *(University of California, USA)*, Michael Carl Tschantz *(International Computer Science Institute, USA)*
- **Heterographic Pun Recognition via Pronunciation and Spelling Understanding Gated Attention Network**.....363
  Yufeng Diao, Hongfei Lin, Liang Yang, Xiaochao Fan, Di Wu, Dongyu Zhang, Kan Xu *(Dalian University of Technology, China)*
- **Evaluating Login Challenges as aDefense Against Account Takeover**.....372
  Periwinkle Doerfler *(NYU, USA)*, Kurt Thomas, Maija Marincenko, Juri Ranieri, Yu Jiang, Angelika Moscicki *(Google, USA)*, Damon McCoy *(NYU, USA)*
- **On Attribution of Recurrent Neural Network Predictions via Additive Decomposition**.....383
  Mengnan Du, Ninghao Liu, Fan Yang, Shuiwang Ji, Xia Hu *(Texas A&M University, USA)*
- **Is a Single Embedding Enough? Learning Node Representations that Capture Multiple Social Contexts**.....394
  Alessandro Epasto, Bryan Perozzi *(Google, USA)*
- **On-Device Algorithms for Public-Private Data with Absolute Privacy**.....405
  Alessandro Epasto, Hossein Esfandiari, Vahab Mirrokni *(Google, USA)*
- **Graph Neural Networks for Social Recommendation**.....417
  Wenqi Fan *(City University of Hong Kong, Hong Kong)*, Yao Ma *(Michigan State University, USA)*, Qing Li *(City University of Hong Kong, Hong Kong)*, Yuan He, Eric Zhao *(JD.com, China)*, Jiliang Tang *(Michigan State University, USA)*, Dawei Yin *(JD.com, China)*
- **Knowledge-Enhanced Ensemble Learning for Word Embeddings**.....427
  Lanting Fang *(Southeast University, China)*, Yong Luo, Kaiyu Feng, Kaiqi Zhao *(Nanyang Technological University, Singapore)*, Aiqun Hu *(Southeast University, China)*
- **Joint Entity Linking with Deep Reinforcement Learning**.....438
  Zheng Fang, Yanan Cao *(Institute of Information Engineering, CAS, China)*, Qian Li *(University of Technology Sydeney, China)*, Dongjie Zhang, Zhenyu Zhang, Yanbing Liu *(Institute of Information Engineering, CAS, China)*
- **Measurement and Early Detection of Third-Party Application Abuse on Twitter**.....448
  Shehroze Farooqi *(University of Iowa, USA)*, Zubair Shafiq *(The University of Iowa, USA)*
- **DPLink: User Identity Linkage via Deep Neural Network From Heterogeneous Mobility Data**.....459
  Jie Feng, Mingyang Zhang, Huandong Wang, Zeyu Yang *(Tsinghua University, China)*, Chao Zhang *(University of Illinois at Urbana-Champaign, USA)*, Yong Li, Depeng Jin *(Tsinghua University, China)*
- **MARINE: Multi-relational Network Embeddings with Relational Proximity and Node Attributes**.....470
  Noriaki Kawamae *(The University of Tokyo, Japan)*
- **Gaps in Information Access in Social Networks**.....480
  Benjamin Fish *(Microsoft, Canada)*, Ashkan Bashardoust *(University of Utah, USA)*, Danah Boyd *(Data & Society, USA)*, Sorelle Friedler *(Haverford College, USA)*, Carlos Scheidegger *(University of Arizona, USA)*, Suresh Venkatasubramanian *(University of Utah, USA)*
- **Cross-domain Recommendation Without Sharing User-relevant Data**.....491
  Chen Gao, Xiangning Chen *(Tsinghua University, China)*, Fuli Feng *(National University of Singapore, Singapore)*, Kai Zhao *(Tsinghua University, China)*, Xiangnan He *(National University of Singapore, Singapore)*, Yong Li, Depeng Jin *(Tsinghua University, China)*
- **Towards Self-Adaptive Metric Learning On the Fly**.....503
  Yang Gao, Yi-Fan Li, Swarup Chandra, Latifur Khan, Bhavani Thuraisingham *(The University of Texas at Dallas, USA)*

- **Knowledge-aware Assessment of Severity of Suicide Risk for Early Intervention**.....514
Manas Gaur, Amanuel Alambo, Joy Prakash Sain, Ugur Kursuncu, Krishnaprasad Thirunarayan *(Kno.e.sis Center, USA)*, Ramakanth Kavuluru *(University of Kentucky, USA)*, Amit Sheth *(Kno.e.sis Center, USA)*, Randy Welton *(Department of Psychiatry, Wright State University, USA)*, Jyotishman Pathak *(Division of Health Informatics, Weill Cornell Medical College, USA)*
- **SpeedReader: Reader Mode Made Fast and Private**.....526
Mohammad Ghasemisharif *(University of Illinois at Chicago, USA)*, Peter Snyder *(Brave Software, USA)*, Andrius Aucinas *(Brave Software, United Kingdom)*, Benjamin Livshits *(Brave Software, Imperial College London, United Kingdom)*
- **Externalities and Fairness**.....538
Masoud Seddighin *(Sharif University of Technology, Iran)*, Hamed Saleh *(University of Maryland, USA)*, Mohammad Ghodsi *(Sharif University of technology, IPM School of Computer Science, Iran)*
- **"It's almost like they're trying to hide it": How User-Provided Image Descriptions Have Failed to Make Twitter Accessible**.....549
Cole Gleason, Patrick Carrington *(Carnegie Mellon University, USA)*, Cameron Cassidy *(Texas A&M University, USA)*, Meredith Ringel Morris *(Microsoft, USA)*, Kris M. Kitani, Jeffrey P. Bigham *(Carnegie Mellon University, USA)*
- **Characterizing Speed and Scale of Cryptocurrency Discussion Spread on Reddit**.....560
Maria Glenski *(University of Notre Dame, USA)*, Emily Saldanha, Svitlana Volkova *(Pacific Nothwest National Laboratory, USA)*
- **Goal-setting And Achievement In Activity Tracking Apps: A Case Study Of MyFitnessPal**.....571
Mitchell Gordon, Tim Althoff, Jure Leskovec *(Stanford University, USA)*
- **Link Prediction on N-ary Relational Data**.....583
Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, Xueqi Cheng *(Institute of Computing Technology, Chinese Academy of Sciences, China)*
- **PrivIdEx: Privacy Preserving and Secure Exchange of Digital Identity Assets.**.....594
Hasini Gunasinghe *(Purdue University, USA)*, Ashish Kundu *(IBM, USA)*, Elisa Bertino *(Purdue University, USA)*, Hugo Krawczyk *(IBM, USA)*, Suresh Chari *(Unknown, USA)*, Kapil Singh *(IBM, USA)*, Dong Su *(Unknown, USA)*
- **RED: Redundancy-Driven Data Extraction from Result Pages**.....605
Jinsong Guo *(UBS Evidence Lab, United Kingdom)*, Valter Crescenzi *(the Department of Engineering, Universita Roma Tre, Italy)*, Tim Furche *(1) the Department of Computer Science, University of Oxford; 2) Wrapidity & Meltwater, United Kingdom)*, Giovanni Grasso *(1) the Department of Mathematics and Computer Science, University of Calabria; 2) Wrapidity & Meltwater, Italy)*, Georg Gottlob *(1) the Department of Computer Science, University of Oxford; 2) the Faculty of Informatics, TU Wien, United Kingdom)*
- **Securing the Deep Fraud Detector in Large-Scale E-Commerce Platform via Adversarial Machine Learning Approach**.....616
Qingyu Guo *(Nanyang Technological University, Singapore)*, Zhao Li *(Alibaba Group, China)*, Bo An *(Nanyang Technological University, Singapore)*, Pengrui Hui, Jiaming Huang, Long Zhang *(Alibaba Group, China)*, Mengchen Zhao *(Nanyang Technological University, Singapore)*
- **Predictive Crawling for Commercial Web Content**.....627
Shuguang Han *(Google, USA)*, Bernhard Brodowsky, Przemek Gajda, Sergey Novikov *(Google, Switzerland)*, Mike Bendersky, Marc Najork, Robin Dua, Alexandrin Popescul *(Google, USA)*
- **Generating Titles for Web Tables**.....638
Braden Hancock *(Stanford University, USA)*, Hongrae Lee, Cong Yu *(Google, USA)*
- **Understanding the Effects of the Neighbourhood Built Environment on Public Health with Open Data**.....648
Apinan Hasthanasombat, Cecilia Mascolo *(University of Cambridge, United Kingdom)*
- **Distributed Tensor Decomposition for Large Scale Health Analytics**.....659
Huan He *(Emory University, USA)*, Jette Henderson *(The University of Texas at Austin, USA)*, Joyce C Ho *(Emory University, USA)*
- **Debiasing Vandalism Detection Models at Wikidata**.....670
Stefan Heindorf, Yan Scholten *(Paderborn University, Germany)*, Gregor Engels *(University of Paderborn, Germany)*, Martin Potthast *(Leipzig University, Germany)*

- **Message Distortion in Information Cascades** 681
  Manoel Horta Ribeiro *(Universidade Federal de Minas Gerais, Brazil)*, Kristina Gligoric, Robert West *(Ecole Polytechnique F??d??rale de Lausanne, Switzerland)*
- **Auditing the Partisanship of Google Search Snippets** 693
  Desheng Hu, Shan Jiang, Ronald E. Robertson, Christo Wilson *(Northeastern University, USA)*
- **To Return or to Explore: Modelling Human Mobility and Dynamics in Cyberspace** 705
  Tianran Hu *(University of Rochester, USA)*, Yinglong Xia *(Futurewei Technologies, Ltd, USA)*, Jiebo Luo *(University of Rochester, USA)*
- **MiST: A Multiview and Multimodal Spatial-Temporal Learning Framework for Citywide Abnormal Event Forecasting** 717
  Chao Huang, Chuxu Zhang *(University of Notre Dame, USA)*, Jiashu Zhao *(JD.com, China)*, Xian Wu, Nitesh Chawla *(University of Notre Dame, USA)*, Dawei Yin *(JD.com, China)*
- **Who Watches the Watchmen: Exploring Complaints on the Web** 729
  Damilola Ibosiola, Ignacio Castro *(Queen Mary University of London, United Kingdom)*, Gianluca Stringhini *(Boston University, USA)*, Steve Uhlig, Gareth Tyson *(Queen Mary University of London, United Kingdom)*
- **Nonlinear Diffusion for Community Detection and Semi-Supervised Learning** 739
  Rania Ibrahim, David Gleich *(Purdue University, USA)*
- **Bayesian Exploration with Heterogeneous Agents** 751
  Nicole Immorlica *(Microsoft, USA)*, Jieming Mao *(University of Pennsylvania, USA)*, Aleksandrs Slivkins *(Microsoft, USA)*, Zhiwei Steven Wu *(University of Minnesota, USA)*
- **Diversity and Exploration in Social Learning** 762
  Nicole Immorlica *(Microsoft, USA)*, Jieming Mao *(University of Pennsylvania, USA)*, Christos Tzamos *(University of Wisconsin-Madison, USA)*
- **Alleviating Users' Pain of Waiting: Effective Task Grouping for Online-to-Offline Food Delivery Services** 773
  Shenggong Ji *(Southwest Jiaotong University, China)*, Yu Zheng *(JD Finance, China)*, Zhaoyuan Wang, Tianrui Li *(Southwest Jiaotong University, China)*
- **CommunityGAN: Community Detection with Generative Adversarial Nets** 784
  Yuting Jia, Qinqin Zhang, Weinan Zhang, Xinbing Wang *(Shanghai Jiao Tong Univeristy, China)*
- **Semantic Text Matching for Long-Form Documents** 795
  Jyun-Yu Jiang *(University of California, USA)*, Mingyang Zhang, Cheng Li, Michael Bendersky, Nadav Golbandi, Marc Najork *(Google, USA)*
- **RealGraph: A Graph Engine Leveraging the Power-Law Distribution of Real-World Graphs** 807
  Yong-Yeon Jo, Myung-Hwan Jang, Sang-Wook Kim *(Hanyang University, Republic of Korea)*, Sunju Park *(Yonsei University, Republic of Korea)*
- **(Mis)Information Dissemination in WhatsApp: Gathering, Analyzing and Countermeasures** ... 818
  Gustavo Resende, Philipe Melo, Hugo Sousa *(Universidade Federal de Minas Gerais, Brazil)*, Johnnatan Messias *(Max Planck Institute for Software Systems, Germany)*, Marisa Vasconcelos *(IBM, Brazil)*, Jussara Almeida, Fabrício Benevenuto *(Universidade Federal de Minas Gerais, Brazil)*
- **The Block Point Process Model for Continuous-time Event-based Dynamic Networks** 829
  Ruthwik Junuthula, Maysam Haghdan *(The University of Toledo, USA)*, Kevin S. Xu *(University of Toledo, USA)*, Vijay Devabhaktuni *(The University of Toledo, USA)*
- **OUTGUARD: Detecting In-Browser Covert Cryptocurrency Mining in the Wild** 840
  Amin Kharraz, Zane Ma, Paul Murley *(University of Illinois at Urbana-Champaign, USA)*, Charles Lever *(Georgia Institute of Technology, USA)*, Joshua Mason, Andrew Miller, Nikita Borisov *(University of Illinois at Urbana-Champaign, USA)*, Manos Antonakakis *(Georgia Institute of Technology, USA)*, Michael Bailey *(University of Illinois at Urbana-Champaign, USA)*
- **From Small-scale to Large-scale Text Classification** 853
  Kang-Min Kim, Yeachan Kim, Jungho Lee, Ji-Min Lee, Sangkeun Lee *(Korea University, Republic of Korea)*
- **Dual Neural Personalized Ranking** 863
  Seunghyeon Kim, Jongwuk Lee *(Sungkyunkwan University, Republic of Korea)*, Hyunjung Shim *(Yonsei University, Republic of Korea)*

- **Studying Preferences and Concerns about Information Disclosure in Email Notifications**.....874
  Yongsung Kim *(Northwestern University, USA)*, Adam Fourney, Ece Kamar *(Microsoft, USA)*
- **RiSER: Learning Better Representations for Richly Structured Emails**.....886
  Furkan Kocayusufoglu *(UC Santa Barbara, USA)*, Ying Sheng, Nguyen Vo, James Wendt, Qi Zhao, Sandeep Tata, Marc Najork *(Google, USA)*
- **Reputation Deflation Through Dynamic Expertise Assessment in Online Labor Markets**.....896
  Marios Kokkodis *(Carroll School of Management, Boston College, USA)*
- **Learning from On-Line User Feedback in Neural Question Answering on the Web**.....906
  Bernhard Kratzwald, Stefan Feuerriegel *(ETH Zurich, Switzerland)*
- **Blockchain Mining Games with Pay Forward**.....917
  Philip Lazos, Elias Koutsoupias, Foluso Ogunlana, Paolo Serafino *(University of Oxford, United Kingdom)*
- **ContraVis: Contrastive and Visual Topic Modeling for Comparing Document Collections**.....928
  Tuan Le *(Oakland University, USA)*, Leman Akoglu *(Carnegie Mellon University, USA)*
- **Classifying Extremely Short Texts by Exploiting Semantic Centroids in Word Mover's Distance Space**.....939
  Changchun Li, Jihong Ouyang, Ximing Li *(Jilin University, China)*
- **Large Scale Semantic Indexing with Deep Level-wise Extreme Multi-label Learning**.....950
  Dingcheng Li, Jingyuan Zhang, Ping Li *(Baidu Inc, USA)*
- **Current Flow Group Closeness Centrality for Complex Networks**.....961
  Huan Li *(Fudan University, China)*, Richard Peng *(Georgia Institute of Technology, USA)*, Liren Shan, Yuhao Yi, Zhongzhi Zhang *(Fudan University, China)*
- **Semi-Supervised Graph Classification: A Hierarchical Graph Perspective**.....972
  Jia Li *(The Chinese University of Hong Kong, Hong Kong)*, Yu Rong *(Tencent AI Lab, China)*, Hong Cheng, Helen Meng *(The Chinese University of Hong Kong, Hong Kong)*, Wenbing Huang, Junzhou Huang *(Tencent AI Lab, China)*
- **Efficient Ridesharing Order Dispatching with Mean Field Multi-Agent Reinforcement Learning**.....983
  Minne Li *(University College London, United Kingdom)*, Zhiwei Qin *(Didi Research America, USA)*, Yan Jiao *(DiDi Chuxing, USA)*, Yaodong Yang, Jun Wang *(University College London, United Kingdom)*, Chenxi Wang, Guobin Wu, Jieping Ye *(Didi Chuxing, China)*
- **Exploiting Ratings, Reviews and Relationships for Item Recommendations in Topic Based Social Networks**.....995
  Pengfei Li *(Zhejiang University, China)*, Hua Lu *(Aalborg University, Denmark)*, Gang Zheng *(Zhejiang University, China)*, Qian Zheng *(Nanyang Technological University, Singapore)*, Long Yang, Gang Pan *(Zhejiang University, China)*
- **Persona-Aware Tips Generation**.....1006
  Piji Li *(Tencent AI Lab, China)*, Zihao Wang *(The Chinese University of Hong Kong, China)*, Lidong Bing *(Alibaba, China)*, Wai Lam *(The Chinese University of Hong Kong, China)*
- **Learning Travel Time Distributions with Deep Generative Model**.....1017
  Xiucheng Li, Gao Cong, Aixin Sun *(Nanyang Technological University, Singapore)*, Yun Cheng *(ETH, Switzerland)*
- **Truth Inference at Scale: A Bayesian Model for Adjudicating Highly Redundant Crowd Annotations**.....1028
  Yuan Li, Benjamin I. P. Rubinstein, Trevor Cohn *(The University of Melbourne, Australia)*
- **Unsupervised Semantic Generative Adversarial Networks for Expert Retrieval**.....1039
  Shangsong Liang *(University College London, United Kingdom)*
- **Estimating the Total Volume of Queries to Google**.....1051
  Fabrizio Lillo *(University of Bologna, Italy)*, Salvatore Ruggieri *(University of Pisa, Italy)*
- **Forecasting U.S. Domestic Migration Using Internet Search Queries**.....1061
  Allen Yilun Lin *(Northwestern University, USA)*, Justin Cranshaw, Scott Counts *(Microsoft, USA)*
- **Learning Dual Retrieval Module for Semi-supervised Relation Extraction**.....1073
  Hongtao Lin *(University of Southern California, USA)*, Jun Yan *(Tsinghua University, China)*, Meng Qu *(University of Illinois at Urbana-Champaign, USA)*, Xiang Ren *(University of Southern California, USA)*

- **Distributed Algorithms for Fully Personalized PageRank on Large Graphs** 1084
  Wenqing Lin *(Tencent, China)*
- **Improving Outfit Recommendation with Co-supervision of Fashion Generation** 1095
  Yujie Lin *(Shandong University, China)*, Pengjie Ren *(University of Amsterdam, Netherlands)*, Zhumin Chen *(Shandong University, China)*, Zhaochun Ren *(Data Science Lab, JD.com, China)*, Jun Ma *(Shandong University, China)*, Maarten de Rijke *(University of Amsterdam, Netherlands)*
- **Learning to Generate Questions by LearningWhat not to Generate** 1106
  Bang Liu, Mingjun Zhao, Di Niu *(University of Alberta, Canada)*, Kunfeng Lai *(The Hong Kong Polytechnic University, China)*, Yancheng He, Haojie Wei, Yu Xu *(Tencent, China)*
- **Feature Generation by Convolutional Neural Network for Click-Through Rate Prediction** 1119
  Bin Liu, Ruiming Tang *(Huawei Noah's Ark Lab, China)*, Yingzhi Chen *(Jinan University, China)*, Jinkai Yu, Huifeng Guo, Yuzhou Zhang *(Huawei Noah's Ark Lab, China)*
- **Efficient (α, β)-core Computation: an Index-based Approach** 1130
  Boge Liu *(The University of New South Wales, Australia)*, Long Yuan *(Nanjing University of Science and Technology, China)*, Xuemin Lin *(The University of New South Wales, Australia)*, Lu Qin *(Centre for Artificial Intelligence, University of Technology, Australia)*, Wenjie Zhang *(The University of New South Wales, Australia)*, Jingren Zhou *(Alibaba Group, China)*
- **Neural Variational Correlated Topic Modeling** 1142
  Luyang Liu, Heyan Huang, Yang Gao *(Beijing Insititute of Technology, China)*, Yongfeng Zhang *(Rutgers University, USA)*, Xiaochi Wei *(Baidu Inc, China)*
- **A Hybrid BitFunnel and Partitioned Elias-Fano Inverted Index** 1153
  Xinyu Liu, Zhaohua Zhang *(Nankai University, China)*, Rebecca Stones *(Nankai University, Australia)*, Yusen Li, Gang Wang, Xiaoguang Liu *(Nankai University, China)*
- **How Do Your Neighbors Disclose Your Information: Social-Aware Time Series Imputation** 1164
  Zongtao Liu, Yang Yang, Wei Huang *(Zhejiang University, China)*, Zhongyi Tang, Ning Li *(State Grid Zhejiang Jinyun Power Supply Co. Ltd., China)*, Fei Wu *(Zhejiang University, China)*
- **Autonomous Learning for Face Recognition in the Wild via Ambient Wireless Cues** 1175
  Chris Xiaoxuan Lu *(University of Oxford, United Kingdom)*, Xuan Kan *(Tongji University, China)*, Bowen Du *(The University of Warwick, United Kingdom)*, Changhao Chen *(University of Oxford, United Kingdom)*, Hongkai Wen *(The University of Warwick, United Kingdom)*, Andrew Markham, Niki Trigoni *(University of Oxford, United Kingdom)*, John Stankovic *(University of Virginia, USA)*
- **Quality Effects on User Preferences and Behaviorsin Mobile News Streaming** 1187
  Hongyu Lu, Min Zhang, Weizhi Ma, Yunqiu Shao, Yiqun Liu, Shaoping Ma *(Tsinghua University, China)*
- **What We Vote for? Answer Selection from User Expertise View in Community Question Answering** 1198
  Shanshan Lyu, Wentao Ouyang *(Institute of Computing Technology, Chinese Academy of Sciences, China)*, Yongqing Wang *(Institute of Automation, Chinese Academy of Sciences, China)*, Huawei Shen *(Chinese Academy of Sciences, China)*, Xueqi Cheng *(Institute of Computing Technology, CAS, P. R. China, China)*
- **Jointly Learning Explainable Rules for Recommendation with Knowledge Graph** 1210
  Weizhi Ma, Min Zhang *(Tsinghua University, China)*, Yue Cao, Woojeong Jin *(University of Southern California, USA)*, Chenyang Wang, Yiqun Liu, Shaoping Ma *(Tsinghua University, China)*, Xiang Ren *(University of Southern California, USA)*
- **Exploring Perceived Emotional Intelligence of Personality-Driven Virtual Agents in Handling User Challenges** 1222
  Xiaojuan Ma, Emily Yang, Pascale Fung *(The Hong Kong University of Science and Technology, Hong Kong)*
- **Moving Deep Learning into Web Browser: How Far Can We Go?** 1234
  Yun Ma, Dongwei Xiang, Shuyu Zheng, Deyu Tian *(Key Laboratory of High Confidence Software Technologies, Ministry of Education, China)*, Xuanzhe Liu *(Peking University, China)*
- **Exploring User Behavior in Email Re-Finding Tasks** 1245
  Joel Mackenzie *(RMIT University, Australia)*, Kshitiz Gupta, Fang Qiao, Ahmed Hassan Awadallah, Milad Shokouhi *(Microsoft, USA)*
- **Jointly Leveraging Intent and Interaction Signals to Predict User Satisfaction with Slate Recommendations** 1256
  Rishabh Mehrotra, Mounia Lalmas *(Spotify Research, United Kingdom)*, Doug Kenney, Thomas Lim-Meng, Golli Hashemian *(Spotify, USA)*

- **SaGe: Web Preemption for Public SPARQL Query Services** 1268
  Thomas Minier *(LS2N, University of Nantes, France)*, Hala Skaf-Molli *(Nantes University, France)*, Pascal Molli *(University of Nantes - LS2N, France)*
- **Hack for Hire: Exploring the Emerging Market for Account Hijacking** 1279
  Ariana Mirian, Joe DeBlasio, Stefan Savage, Geoffrey M. Voelker *(University of California San Diego, USA)*, Kurt Thomas *(Google, USA)*
- **Anomaly Detection in the Dynamics of Web and Social Networks Using Associative Memory** 1290
  Volodymyr Miz, Benjamin Ricaud, Kirell Benzi, Pierre Vandergheynst *(Ecole Polytechnique Fédérale de Lausanne, Switzerland)*
- **Towards Predicting a Realisation of an Information Need based on Brain Signals** 1300
  Yashar Moshfeghi *(University of Strathclyde, United Kingdom)*, Frank Pollick *(University of Glasgow, United Kingdom)*
- **GhostLink: Latent Network Inference for Influence-aware Recommendation** 1310
  Subhabrata Mukherjee *(Amazon, USA)*, Stephan Guennemann *(Technical University of Munich, Germany)*
- **Estimating Walk-Based Similarities Using Random Walk** 1321
  Shogo Murai *(The University of Tokyo, Japan)*, Yuichi Yoshida *(National Institute of Informatics, Japan)*
- **Sensitivity Analysis of Centralities on Unweighted Networks** 1332
  Shogo Murai *(The University of Tokyo, Japan)*, Yuichi Yoshida *(National Institute of Informatics, Japan)*
- **Modeling Heart Rate and Activity Data for Personalized Fitness Recommendation** 1343
  Jianmo Ni, Larry Muhlstein, Julian McAuley *(University of California, USA)*
- **Generating Product Descriptions from User Reviews** 1354
  Slava Novgorodov *(eBay Research, Israel)*, Guy Elad *(Technion, Israel Institute of Technology, Israel)*, Ido Guy *(eBay Research, Israel)*, Kira Radinsky *(Technion, Israel Institute of Technology, Israel)*
- **Google Dataset Search: Building a search engine for datasets in an open Web ecosystem** 1365
  Dan Brickley, Matthew Burgess, Natasha Noy *(Google, USA)*
- **Dealing with Interdependencies and Uncertainty in Multi-Channel Advertising Campaigns Optimization** 1376
  Alessandro Nuara, Nicola Sosio, Francesco Trovò, Maria Chiara Zaccardi, Nicola Gatti, Marcello Restelli *(Politecnico di Milano, Italy)*
- **Reconciliation *k*-median: Clustering with Non-polarized Representatives** 1387
  Bruno Ordozgoiti *(Universidad Politécnica de Madrid, Spain)*, Aristides Gionis *(Aalto University, Finland)*
- **ActiveLink: Deep Active Learning for Link Prediction in Knowledge Graphs** 1398
  Natalia Ostapuk, Jie Yang *(eXascale Infolab, University of Fribourg, Switzerland)*, Philippe Cudre-Mauroux *(U. of Fribourg, Switzerland)*
- **Choosing to Grow a Graph: Modeling Network Formation as Discrete Choice** 1409
  Jan Overgoor *(Stanford University, USA)*, Austin Benson *(Cornell University, USA)*, Johan Ugander *(Stanford University, USA)*
- **Policy Gradients for Contextual Recommendations** 1421
  Feiyang Pan *(Institute of Computing Technology, Chinese Academy of Sciences, China)*, Qingpeng Cai, Pingzhong Tang *(Tsinghua University, China)*, Fuzhen Zhuang *(Institute of Computing Technology, Chinese Academy of Sciences, China)*, Qing He *(Institute of Computing Technology, CAS, China)*
- **Cookie Synchronization: Everything You Always Wanted to Know But Were Afraid to Ask** 1432
  Panagiotis Papadopoulos *(Brave Software, Greece)*, Nicolas Kourtellis *(Telefonica Research, Spain)*, Evangelos Markatos *(ICS-FORTH, Greece)*
- **Adversarial Sampling and Training for Semi-Supervised Information Retrieval** 1443
  Dae Hoon Park *(Huawei Research America, USA)*, Yi Chang *(Jilin University, China)*
- **Nameles: An intelligent system for Real-Time Filtering of Invalid Ad Traffic** 1454
  Antonio Pastor *(Universidad Carlos III de Madrid, Spain)*, Matti Pärssinen *(Aalto University, Spain)*, Patricia Callejo, Pelayo Vallina *(IMDEA Networks Institute / Universidad Carlos III de Madrid, Spain)*, Rubén Cuevas *(Telematics Engineering Departement, Universidad Carlos III de Madrid), Spain)*, Ángel Cuevas *(Universidad Carlos III de Madrid, Spain)*, Mikko Kotila *(Botlab, USA)*, Arturo Azcorra *(University Carlos III and IMDEA Networks, Spain)*

- **Learning How to Correct a Knowledge Base from the Edit History** 1465
  Thomas Pellissier Tanon *(Télécom ParisTech, France)*, Camille Bourgaux *(DI ENS, CNRS, ENS, PSL University, France)*, Fabian Suchanek *(Télécom ParisTech, France)*
- **Bootstrapping Domain-Specific Content Discovery on the Web** 1476
  Kien Pham, Aecio Santos, Juliana Freire *(New York University, USA)*
- **Privacy-Preserving Crowd-Sourcing of Web Searches with Private Data Donor** 1487
  Vincent Primault, Vasileios Lampos, Ingemar Cox, Emiliano De Cristofaro *(University College London, United Kingdom)*
- **Community Detection through Likelihood Optimization: In Search of a Sound Model** 1498
  Liudmila Prokhorenkova *(Moscow Institute of Physics and Technology & Yandex, Russian Fed.)*, Alexey Tikhonov *(Yandex, Germany)*
- **NetSMF: Large-Scale Network Embedding as Sparse Matrix Factorization** 1509
  Jiezhong Qiu *(Tsinghua University, China)*, Yuxiao Dong *(Microsoft, USA)*, Hao Ma *(Facebook AML, USA)*, Jian Li *(Tsinghua University, China)*, Chi Wang, Kuansan Wang *(Microsoft, USA)*, Jie Tang *(Tsinghua University, China)*
- **Crowd-Mapping Urban Objects from Street-Level Imagery** 1521
  Sihang Qiu, Achilleas Psyllidis, Alessandro Bozzon, Geert-Jan Houben *(Delft University of Technology, Netherlands)*
- **Web Experience in Mobile Networks: Lessons from Two Million Page Visits** 1532
  Mohammad Rajiullah *(Karlstad University, Sweden)*, Andra Lutu *(Telefonica Research, Spain)*, Ali Safari Khatouni *(Politecnico di Torino, Italy)*, Mah-Rukh Fida *(The University of Edinburgh, United Kingdom)*, Marco Mellia *(Politecnico di Torino, Italy)*, Anna Brunstrom *(Karlstad University, Sweden)*, Ozgu Alay *(Simula Research Laboratory, Norway)*, Stefan Alfredsson *(Karlstad University, Sweden)*, Vincenzo Mancuso *(IMDEA Networks Institute, Spain)*
- **A Dynamic Embedding Model of the Media Landscape** 1544
  Jérémie Rappaz, Dylan Bourgeois, Karl Aberer *(Ecole Polytechnique Fédérale de Lausanne, Switzerland)*
- **Keyphrase Extraction from Disaster-related Tweets** 1555
  Jishnu Ray Chowdhury *(Kansas State University, USA)*, Cornelia Caragea *(University of Illinois at Chicago, USA)*, Doina Caragea *(Kansas State University, USA)*
- **Citation Needed: A Taxonomy and Algorithmic Assessment of Wikipedia's Verifiability** 1567
  Miriam Redi *(Wikimedia Foundation, United Kingdom)*, Besnik Fetahu *(L3S Research Center, Germany)*, Jonathan Morgan, Dario Taraborelli *(Wikimedia Foundation, USA)*
- **SWAT: Seamless Web Authentication Technology** 1579
  Florentin Rochet *(UCLouvain Crypto Group, Belgium)*, Kyriakos Efthymiadis *(VUB, Belgium)*, François Koeune *(UCLouvain Crypto Group, Belgium)*, Olivier Pereira *(Uclouvain Crypto Group, Belgium)*
- **Before and After GDPR: The Changes in Third Party Presence at Public and Private European Websites** 1590
  Jannick Sørensen, Sokol Kosta *(Aalborg University, Denmark)*
- **Multiple Treatment Effect Estimation using Deep Generative Model with Task Embedding** 1601
  Shiv Kumar Saini, Sunny Dhamnani *(Adobe Research, India)*, Aakash *(Indian Institute of Technology Madras, India)*, Akil Arif Ibrahim *(Indian Institute of Technology Roorkee, India)*, Prithviraj Chavan *(Adobe, India)*
- **A Human-in-the-loop Attribute Design Framework for Classification** 1612
  Md Abdus Salam, Mary E. Koone *(UTA, USA)*, Saravanan Thirumuruganathan *(QCRI, Qatar)*, Gautam Das *(UTA, USA)*, Senjuti Basu Roy *(NJIT, USA)*
- **How Representative Is a SPARQL Benchmark? An Analysis of RDF Triplestore Benchmarks** 1623
  Muhammad Saleem *(AKSW, University Of Leizpig, Germany)*, Gábor Szárnyas *(Budapest University of Technology and Economics, Hungary)*, Felix Conrads *(AKSW, University Leipzig, Germany)*, Syed Ahmad Chan Bukhari *(Yale University, School of Medicine, USA)*, Qaiser Mehmood *(NUIG, Ireland)*, Axel-Cyrille Ngonga Ngomo *(Paderborn University, Germany)*
- **Self- and Cross-Excitation in Stack Exchange Question & Answer Communities** 1634
  Tiago Santos *(Graz University of Technology, Austria)*, Simon Walk *(Detego GmbH, Austria)*, Roman Kern *(Know-Center GmbH, Austria)*, Markus Strohmaier *(GESIS - Leibniz-Institute for the Social Sciences & RWTH Aachen University, Germany)*, Denis Helic *(Graz University of Technology, Austria)*

- **Automatic Boolean Query Refinement for Systematic Review Literature Search** 1646
  Harrisen Scells, Guido Zuccon *(The University of Queensland, Australia)*, Bevan Koopman *(CSIRO, Australia)*
- **Exploiting Diversity in Android TLS Implementations for Mobile App Traffic Classification** 1657
  Satadal Sengupta, Niloy Ganguly *(IIT Kharagpur, India)*, Pradipta De *(Georgia Southern University, USA)*, Sandip Chakraborty *(IIT Kharagpur, India)*
- **BaG: Behavior-aware Group Detection in Crowded Urban Spaces using WiFi Probes** 1669
  Jiaxing Shen, Jiannong Cao *(The Hong Kong Polytechnic University, Hong Kong)*, Xuefeng Liu *(Huazhong University of Science and Technology, China)*
- **SWeG: Lossless and Lossy Summarization of Web-Scale Graphs** 1679
  Kijung Shin *(Carnegie Mellon University, USA)*, Amol Ghoting, Myunghwan Kim, Hema Raghavan *(LinkedIn Corporation, USA)*
- **Generative Graph Models based on Laplacian Spectra** 1691
  Alana Shine, David Kempe *(University of Southern California, USA)*
- **VACCINE: Using Contextual Integrity ForData Leakage Detection** 1702
  Yan Shvartzshnaider, Zvonimir Pavlinovic, Ananth Balashankar, Thomas Wies, Lakshminarayanan Subramanian *(New York University, USA)*, Helen Nissenbaum *(Cornell Tech, USA)*, Prateek Mittal *(Princeton University, USA)*
- **Shapley Meets Uniform: An Axiomatic Framework for Attribution in Online Advertising** ... 1713
  Omar Besbes *(Columbia University, USA)*, Antoine Desir *(INSEAD, France)*, Vineet Goyal, Garud Iyengar, Raghav Singal *(Columbia University, USA)*
- **Urban Vibes and Rural Charms: Analysis of Geographic Diversity in Mobile Service Usage at National Scale** 1724
  Rajkarn Singh *(The University of Edinburgh, United Kingdom)*, Marco Fiore *(CNR IEIIT, Italy)*, Mahesh Marina *(The University of Edinburgh, United Kingdom)*, Alberto Tarable, Alessandro Nordio *(CNR IEIIT, Italy)*
- **Anything to Hide? Studying Minified and Obfuscated Code in the Web** 1735
  Philippe Skolka, Cristian-Alexandru Staicu, Michael Pradel *(TU Darmstadt, Germany)*
- **SciLens: Evaluating the Quality of Scientific News Articles Using Social Media and Scientific Literature Indicators** 1747
  Panayiotis Smeros *(École Polytechnique Fédérale de Lausanne, Switzerland)*, Carlos Castillo *(Universitat Pompeu Fabra (UPF), Spain)*, Karl Aberer *(École Polytechnique Fédérale de Lausanne, Switzerland)*
- **Using Variability as a Guiding Principle to Reduce Latency in Web Applications via OS Profiling** 1759
  Amoghavarsha Suresh, Anshul Gandhi *(Stony Brook University, USA)*
- **Aggregating E-commerce Search Results from Heterogeneous Sources via Hierarchical Reinforcement Learning** 1771
  Ryuichi Takanobu *(Tsinghua University, Japan)*, Tao Zhuang *(Alibaba Group, China)*, Minlie Huang *(Tsinghua University, China)*, Jun Feng *(State Grid, China)*, Haihong Tang, Bo Zheng *(Alibaba Group, China)*
- **Towards Neural Mixture Recommender for Long Range Dependent User Sequences** 1782
  Jiaxi Tang *(Simon Fraser University, Canada)*, Francois Belletti, Sagar Jain, Minmin Chen, Alex Beutel, Can Xu, Ed H. Chi *(Google, USA)*
- **Leveraging Peer Communication to Enhance Crowdsourcing** 1794
  Wei Tang *(Washington University in St. Louis, USA)*, Ming Yin *(Purdue University, USA)*, Chien-Ju Ho *(Washington University in St. Louis, USA)*
- **Joint Modeling of Dense and Incomplete Trajectories for Citywide Traffic Volume Inference** 1806
  Xianfeng Tang *(The Pennsylvania State University, USA)*, Boqing Gong *(Tencent, USA)*, Yanwei Yu *(Yantai University, China)*, Huaxiu Yao *(The Pennsylvania State University, USA)*, Yandong Li *(University of Central Florida, USA)*, Haiyong Xie *(University of Science and Technology of China, China)*, Xiaoyu Wang *(Intellifusion Inc., China)*
- **Listening between the Lines: Learning Personal Attributes from Conversations** 1818
  Anna Tigunova, Andrew Yates, Paramita Mirza, Gerhard Weikum *(Max Planck Institute for Informatics, Germany)*

- **Dynamic Deep Multi-modal Fusion for Image Privacy Prediction** 1829
  Ashwini Tonge *(Kansas State University, USA)*, Cornelia Caragea *(University of Illinois at Chicago, USA)*
- **Signed Distance-based Deep Memory Recommender** 1841
  Thanh Tran, Xinyue Liu, Kyumin Lee, Xiangnan Kong *(Worcester Polytechnic Institute, USA)*
- **Rating Worker Skills and Task Strains in Collaborative Crowd Computing: A Competitive Perspective** 1853
  George Trimponias *(Huawei Noah's Ark Lab, Hong Kong)*, Xiaojuan Ma, Qiang Yang *(The Hong Kong University of Science and Technology, Hong Kong)*
- **Multimodal Review Generation for Recommender Systems** 1864
  Quoc-Tuan Truong, Hady Lauw *(Singapore Management University, Singapore)*
- **Revisiting Wedge Sampling for Triangle Counting** 1875
  Ata Turk *(Boston University, USA)*, Duru Turkoglu *(Loyola University, USA)*
- **RAQ: Relationship-Aware Graph Querying in Large Networks** 1886
  Jithin Vachery *(Indian Institute of Technology Madras, India)*, Akhil Arora *(EPFL)*, Sayan Ranu *(Indian Institute of Technology Delhi, India)*, Arnab Bhattacharya *(Indian Institute of Technology, Kanpur, India)*
- **BOLT-K: Bootstrapping Ontology Learning via Transfer of Knowledge** 1897
  Nikhita Vedula, Pranav Maneriker, Srinivasan Parthasarathy *(The Ohio State University, USA)*
- **Learning Resolution Parameters for Graph Clustering** 1909
  Nate Veldt, David Gleich *(Purdue University, USA)*, Anthony Wirth *(The University of Melbourne, Australia)*
- **Auditing Offline Data Brokers via Facebook's Advertising Platform** 1920
  Giridhari Venkatadri, Piotr Sapiezynski *(Northeastern University, USA)*, Elissa M. Redmiles *(University of Maryland, USA)*, Alan Mislove *(Northeastern University, USA)*, Oana Goga *(LIG, Grenoble, France)*, Michelle Mazurek *(University of Maryland, USA)*, Krishna P. Gummadi *(MPI-SWS, Germany)*
- **"Data Strikes": Evaluating the Effectiveness of a New Form of Collective Action Against Technology Companies** 1931
  Nicholas Vincent, Brent Hecht *(Northwestern University, USA)*, Shilad Sen *(Macalester College, USA)*
- **Learning Semantic Models of Data Sources Using Probabilistic Graphical Models** 1944
  Binh Vu, Craig Knoblock, Jay Pujara *(University of Southern California, USA)*
- **Generalists and Specialists: Using Community Embeddings to Quantify Activity Diversity in Online Platforms** 1954
  Isaac Waller, Ashton Anderson *(University of Toronto, Canada)*
- **A Family of Fuzzy Orthogonal Projection Models for Monolingual and Cross-lingual Hypernymy Prediction** 1965
  Chengyu Wang, Yan Fan, Xiaofeng He, Aoying Zhou *(East China Normal University, China)*
- **Modeling Item-Specific Temporal Dynamics of Repeat Consumption for Recommender Systems** 1977
  Chenyang Wang, Min Zhang, Weizhi Ma, Yiqun Liu, Shaoping Ma *(Tsinghua University, China)*
- **Understanding the Evolution of Mobile App Ecosystems: A Longitudinal Measurement Study of Google Play** 1988
  Haoyu Wang *(Beijing University of Posts and Telecommunications, China)*, Hao Li *(OrangeAPK, China)*, Yao Guo *(Peking University, China)*
- **Multi-Task Feature Learning for Knowledge Graph Enhanced Recommendation** 2000
  Hongwei Wang *(Shanghai Jiao Tong University, China)*, Fuzheng Zhang *(MeituanDianping, China)*, Miao Zhao, Wenjie Li *(The Hong Kong Polytechnic University, Hong Kong)*, Xing Xie *(Microsoft, China)*, Minyi Guo *(Shanghai Jiao Tong University, China)*
- **Towards Efficient Sharing: A Usage Balancing Mechanism for Bike Sharing Systems** 2011
  Shuai Wang *(Southeast University, China)*, Tian He *(University of Minnesota, USA)*, Desheng Zhang *(Rutgers University, USA)*, Yunhuai Liu *(Peking University, China)*, Sang H. Son *(Daegu Gyeongbuk Institute of Science and Technology, North Korea)*
- **Heterogeneous Graph Attention Network** 2022
  Xiao Wang, Houye Ji, Chuan Shi, Bai Wang *(Beijing University of Posts and Telecommunications, China)*, Yanfang Ye *(West Virginia University, USA)*, Peng Cui *(Tsinghua University, China)*, Philip S Yu *(University of Illinois at Chicago, USA)*

- **Aspect-level Sentiment Analysis using AS-Capsules** 2033
  Yequan Wang *(Tsinghua University, China)*, Aixin Sun *(Nanyang Technological University, Singapore)*, Minlie Huang, Xiaoyan Zhu *(Tsinghua University, China)*
- **Quality-Sensitive Training! Social Advertisement Generation by Leveraging User Click Behavior** 2045
  Yongzhen Wang *(Dalian Maritime University, China)*, Heng Huang, Yuliang Yan *(Alibaba Group, China)*, Xiaozhong Liu *(Indiana University Bloomington, USA)*
- **Demographic Inference and Representative Population Estimates from Multilingual Social Media Data** 2056
  Zijian Wang *(Stanford University, USA)*, Scott Hale *(University of Oxford, United Kingdom)*, David Ifeoluwa Adelani *(Saarland, Germany)*, Przemyslaw Grabowicz *(Max Planck Institute for Software Systems, Germany)*, Timo Hartman, Fabian Flöck *(GESIS, Germany)*, David Jurgens *(University of Michigan, USA)*
- **Iterative Discriminant Tensor Factorization for Behavior Comparison in Massive Open Online Courses** 2068
  Xidao Wen, Yu-Ru Lin *(University of Pittsburgh, USA)*, Xi Liu *(Texas A&M University, USA)*, Peter Brusilovsky, Jordan Barría Pineda *(University of Pittsburgh, USA)*
- **Dynamic Ensemble of Contextual Bandits to Satisfy Users' Changing Interests** 2080
  Qingyun Wu, Huazheng Wang *(University of Virginia, USA)*, Yanen Li *(Snap Inc., USA)*, Hongning Wang *(University of Virginia, USA)*
- **Dual Graph Attention Networks for Deep Latent Representation of Multifaceted Social Effects in Recommender Systems** 2091
  Qitian Wu, Hengrui Zhang, Xiaofeng Gao *(Shanghai Jiao Tong University, China)*, Peng He *(Tencent, China)*, Paul Weng *(Shanghai Jiao Tong University, China)*, Han Gao *(Tencent, China)*, Guihai Chen *(Shanghai Jiao Tong University, China)*
- **Grid-based Evaluation Metrics for Web Image Search** 2103
  Xiaohui Xie, Jiaxin Mao, Yiqun Liu *(Tsinghua University, China)*, Maarten de Rijke *(University of Amsterdam, Netherlands)*, Yunqiu Shao *(Tsinghua University, China)*, Zixin Ye *(Beihang University, China)*, Min Zhang, Shaoping Ma *(Tsinghua Unviersity, China)*
- **Sarcasm Detection with Self-matching Networks and Low-rank Bilinear Pooling** 2115
  Tao Xiong, Peiran Zhang, Hongbo Zhu *(Alibaba Group, China)*, Yihui Yang *(AfterPay, China)*
- **A First Look at Deep Learning Apps on Smartphones** 2125
  Mengwei Xu, Jiawei Liu, Yuanqiang Liu *(Peking University, China)*, Felix Xiaozhu Lin *(Purdue ECE, USA)*, Yunxin Liu *(Microsoft, China)*, Xuanzhe Liu *(Peking University, China)*
- **Constrained Local Graph Clustering by Colored Random Walk** 2137
  Yaowei Yan, Yuchen Bian, Dongsheng Luo, Dongwon Lee, Xiang Zhang *(The Pennsylvania State University, USA)*
- **Revisiting User Mobility and Social Relationships in LBSNs: A Hypergraph Embedding Approach** 2147
  Dingqi Yang *(eXascale Infolab, University of Fribourg,, Switzerland)*, Bingqing Qu *(University of Fribourg, Switzerland)*, Jie Yang *(eXascale Infolab, University of Fribourg, Switzerland)*, Philippe Cudre-Mauroux *(U. of Fribourg, Switzerland)*
- **Scalpel-CD: Leveraging Crowdsourcing and Deep Probabilistic Modeling for Debugging Noisy Training Data** 2158
  Jie Yang *(U. of Fribourg, Switzerland)*, Alisa Smirnova *(University of Fribourg, Switzerland)*, Dingqi Yang *(U. of Fribourg, Switzerland)*, Gianluca Demartini *(The University of Queensland, Australia)*, Yuan Lu *(ING Bank, Netherlands)*, Philippe Cudre-Mauroux *(U. of Fribourg, Switzerland)*
- **How Intention Informed Recommendations Modulate Choices: A Field Study of Spoken Word Content** 2169
  Longqi Yang, Michael Sobolev *(Cornell University, USA)*, Yu Wang *(Himalaya Media, USA)*, Jenny Chen *(City University of New York, USA)*, Drew Dunne, Christina Tsangouri, Nicola Dell, Mor Naaman, Deborah Estrin *(Cornell University, USA)*
- **Learning from Multiple Cities: A Meta-Learning Approach for Spatial-Temporal Prediction** 2181
  Huaxiu Yao *(The Pennsylvania State University, USA)*, Yiding Liu *(Nanyang Technological University, Singapore)*, Ying Wei *(Tencent, China)*, Xianfeng Tang, Zhenhui Li *(The Pennsylvania State University, USA)*

- **STFNets: Learning Sensing Signals from the Time-Frequency Perspective with Short-Time Fourier Neural Networks** 2192
  Shuochao Yao *(University of Illinois at Urbana-Champaign, USA)*, Ailing Piao *(University of Washington, USA)*, Wenjun Jiang *(State University of New York at Buffalo, USA)*, Yiran Zhao *(University of Illinois at Urabna-Champaign, USA)*, Huajie Shao, Shengzhong Liu, Dongxin Liu, Jinyang Li, Tianshi Wang *(University of Illinois at Urbana-Champaign, USA)*, Shaohan Hu *(IBM, USA)*, Lu Su *(State University of New York at Buffalo, USA)*, Jiawei Han, Tarek Abdelzaher *(University of Illinois at Urbana-Champaign, USA)*
- **CoaCor: Code Annotation for Code Retrieval with Reinforcement Learning** 2203
  Ziyu Yao *(The Ohio State University, USA)*, Jayavardhan Reddy Peddamail *(The Ohio State University, USA)*, Huan Sun *(The Ohio State University, USA)*
- **Snapshot-based Loading Acceleration of Web Apps with Nondeterministic JavaScript Execution** 2215
  Jihwan Yeo, Changhyun Shin, Soo-Mook Moon *(Seoul National University, Republic of Korea)*
- **Doppelgängers on the Dark Web: A Large-scale Assessment on Phishing Hidden Web Services** 2225
  Changhoon Yoon *(S2W LAB Inc. and Korea Advanced Institute of Science and Technology, Republic of Korea)*, Kwanwoo Kim, Yongdae Kim, Seungwon Shin, Sooel Son *(Korea Advanced Institute of Science and Technology, Republic of Korea)*
- **Hierarchical Temporal Convolutional Networks for Dynamic Recommender Systems** 2236
  Jiaxuan You *(Stanford University, USA)*, Yichen Wang *(Georgia Institute of Technology, USA)*, Aditya Pal, Pong Eksombatchai, Chuck Rosenburg *(Pinterest, USA)*, Jure Leskovec *(Stanford University, USA)*
- **Spectrum-enhanced Pairwise Learning to Rank** 2247
  Wenhui Yu, Zheng Qin *(Tsinghua University, China)*
- **Beyond Shortest Paths: Route Recommendations for Ride-sharing** 2258
  Chak Fai Yuen, Abhishek Pratap Singh, Sagar Goyal, Sayan Ranu, Amitabha Bagchi *(Indian Institute of Technology Delhi, India)*
- **Detecting Low Self-Esteem in Youths from Web Search Data** 2270
  Anis Zaman, Rupam Acharyya, Henry Kautz, Vincent Silenzio *(University of Rochester, USA)*
- **The Matthew Effect in Computation Contests: High Difficulty May Lead to 51% Dominance** 2281
  Yulong Zeng *(Nebulas, China)*, Song Zuo *(Google, China)*
- **Judging a Book by Its Cover: The Effect of Facial Perception on Centrality in Social Networks** 2290
  Dongyu Zhang, Teng Guo, Hanxiao Pan, Jie Hou, Zhitao Feng *(School of Software, Dalian University of Technology, China)*, Liang Yang, Hongfei Lin *(School of Computer Science and Technology, Dalian University of Technology, China)*, Feng Xia *(School of Software, Dalian University of Technology, China)*
- **Pruning based Distance Sketches with Provable Guarantees on Random Graphs** 2301
  Hongyang Zhang *(Stanford University, USA)*, Huacheng Yu *(Harvard University, USA)*, Ashish Goel *(Stanford University, USA)*
- **Large-Scale Talent Flow Forecast with Dynamic Latent Factor Model** 2312
  Le Zhang *(University of Science and Technology of China, China)*, Hengshu Zhu *(Baidu Talent Intelligence Center, China)*, Tong Xu *(University of Science and Technology of China, China)*, Chen Zhu *(Baidu Talent Intelligence Center, China)*, Chuan Qin *(University of Science and Technology of China, China)*, Hui Xiong *(Baidu Talent Intelligence Center, China)*, Enhong Chen *(University of Science and Technology of China, China)*
- **From Stances' Imbalance to Their HierarchicalRepresentation and Detection** 2323
  Qiang Zhang, Shangsong Liang, Aldo Lipani *(University College London, United Kingdom)*, Zhaochun Ren *(Jingdong, China)*, Emine Yilmaz *(University College London, United Kingdom)*
- **Reply-Aided Detection of Misinformation via Bayesian Deep Learning** 2333
  Qiang Zhang, Aldo Lipani, Shangsong Liang, Emine Yilmaz *(University College London, United Kingdom)*
- **Multilevel Network Alignment** 2344
  Si Zhang, Hanghang Tong, Ross Maciejewski *(Arizona State University, USA)*, Tina Eliassi-Rad *(Northeastern University, USA)*

- **Automatic Generation of Pattern-controlled Product Description in E-commerce** 2355
  Tao Zhang, Jin Zhang, Chengfu Huo, Weijun Ren *(Alibaba Group, China)*
- **Iteratively Learning Embeddings and Rules for Knowledge Graph Reasoning** 2366
  Wen Zhang *(Zhejiang University, China)*, Bibek Paudel *(University of Zurich, Switzerland)*, Liang Wang *(Zhejiang University, China)*, Jiaoyan Chen *(University of Oxford, United Kingdom)*, Hai Zhu, Wei Zhang *(Alibaba Group, China)*, Abraham Bernstein *(University of Zurich, Switzerland)*, Huajun Chen *(Zhejiang University, China)*
- **Language in Our Time: An Empirical Analysis of Hashtags** 2378
  Yang Zhang *(CISPA, Germany)*
- **Neural IR Meets Graph Embedding: A Ranking Model for Product Search** 2390
  Yuan Zhang, Dong Wang, Yan Zhang *(Peking University, China)*
- **Neural Multimodal Belief Tracker with Adaptive Attention for Dialogue Systems** 2401
  Zheng Zhang *(Tsinghua University, China)*, Lizi Liao *(National University of Singapore, Singapore)*, Minlie Huang, Xiaoyan Zhu *(Tsinghua University, China)*, Tat-Seng Chua *(National University of Singapore, Singapore)*
- **Auto-EM: End-to-end Fuzzy Entity-Matching using Pre-trained Deep Models and Transfer Learning** 2413
  Chen Zhao *(University of Maryland, USA)*, Yeye He *(Microsoft, USA)*
- **Review Response Generation in E-Commerce Platforms with External Product Information** 2425
  Lujun Zhao *(Fudan University, China)*, Kaisong Song, Changlong Sun *(Alibaba Group, China)*, Qi Zhang, Xuanjing Huang *(Fudan University, China)*, Xiaozhong Liu *(Alibaba Group, China)*
- **CBHE: Corner-based Building Height Estimation for Complex Street Scene Images** 2436
  Yunxiang Zhao, Jianzhong Qi, Rui Zhang *(The University of Melbourne, Australia)*
- **Domain-Constrained Advertising Keyword Generation** 2448
  Hao Zhou, Minlie Huang *(Tsinghua University, China)*, Yishun Mao, Changlei Zhu, Peng Shu *(Sogou Inc., China)*, Xiaoyan Zhu *(Tsinghua University, China)*
- **Predicting ConceptNet Path Quality Using Crowdsourced Assessments of Naturalness** 2460
  Yilun Zhou *(MIT, USA)*, Steven Schockaert *(Cardiff University, United Kingdom)*, Julie Shah *(MIT, USA)*
- **A Hierarchical Attention Retrieval Model for Healthcare Question Answering** 2472
  Ming Zhu, Aman Ahuja *(Virginia Tech, USA)*, Wei Wei *(Google, USA)*, Chandan K. Reddy *(Virginia Tech, USA)*
- **ShadowBlock: A Lightweight and Stealthy Adblocking Browser** 2483
  Shitong Zhu *(University of California, USA)*, Umar Iqbal *(University of Iowa, USA)*, Zhongjie Wang, Zhiyun Qian *(University of California, USA)*, Zubair Shafiq *(University of Iowa, USA)*, Weiteng Chen *(University of California, USA)*
- **GraphVite: A High-Performance CPU-GPU Hybrid System for Node Embedding** 2494
  Zhaocheng Zhu *(Montr??al Institute for Learning Algorithms, Canada)*, Shizhen Xu *(Tsinghua University, China)*, Jian Tang *(HEC Montr??al / Montr??al Institute for Learning Algorithms, Canada)*, Meng Qu *(Montr??al Institute for Learning Algorithms, Canada)*
- **Transfer Learning for Unsupervised Influenza-like Illness Models from Online Search Data** 2505
  Bin Zou, Vasileios Lampos, Ingemar Cox *(University College London, United Kingdom)*
- **Tortoise or Hare? Quantifying the Effects of Performance on Mobile App Retention** 2517
  Agustin Zuniga, Huber Flores, Eemil Lagerspetz *(University of Helsinki, Finland)*, Petteri Nurmi *(Lancaster University & University of Helsinki, United Kingdom)*, Sasu Tarkoma *(University of Helsinki, Finland)*, Pan Hui *(University of Helsinki & Hong Kong University of Science and Technology, Finland)*, Jukka Manner *(Aalto University, Finland)*

Short Paper

- **BotCamp: Bot-driven Interactions in Social Campaigns** 2529
  Noor Abu-El-Rub, Abdullah Mueen *(University of New Mexico, USA)*
- **City-Wide Signal Strength Maps: Prediction with Random Forests** 2536
  Emmanouil Alimpertis, Athina Markopoulou, Carter Butts *(University of California, USA)*, Konstantinos Psounis *(University of Souther California, USA)*

- **Is Yelp Actually Cleaning Up the Restaurant Industry? A Re-Analysis on the Relative Usefulness of Consumer Reviews** 2543
  Kristen M. Altenburger *(Stanford University, USA)*, Daniel E. Ho *(Stanford Law School, USA)*
- **Bi-LSTM-CRF Sequence Labeling for Keyphrase Extraction from Scholarly Documents** 2551
  Rabah Alzaidy *(The Pennsylvania State University, USA)*, Cornelia Caragea *(University of Illinois at Chicago, USA)*, C. Lee Giles *(The Pennsylvania State University, USA)*
- **Longitudinal Adversarial Attack on Electronic Health Records Data** 2558
  Sungtae An *(Georgia Institute of Technology, USA)*, Cao Xiao *(IBM, USA)*, Walter F. Stewart *(HINT Consultants, USA)*, Jimeng Sun *(Georgia Institute of Technology, USA)*
- **A Graph is Worth a Thousand Words: Telling Event Stories using Timeline Summarization Graphs** 2565
  Jeffery Ansah, Lin Liu, Wei Kang, Selasie Kwashie, Jixue Li, Jiuyong Li *(University of South Australia, Australia)*
- **Firsthand Opiates Abuse on Social Media: Monitoring Geospatial Patterns of Interest Through a Digital Cohort** 2572
  Duilio Balsamo, Paolo Bajardi, Andr?? Panisson *(ISI Foundation, Italy)*
- **Learn2Clean: Optimizing the Sequence of Tasks for Web Data Preparation** 2580
  Laure Berti-Equille *(Institute of Research for Development (IRD), France)*
- **Global Vectors for Node Representations** 2587
  Robin Brochier *(Université de Lyon, Lyon 2, ERIC EA3083, Digital Scientific Research Technology)*, Adrien Guille, Julien Velcin *(Université de Lyon, Lyon 2, ERIC EA3083)*
- **The Music Streaming Sessions Dataset** 2594
  Brian Brost, Rishabh Mehrotra *(Spotify Research, United Kingdom)*, Tristan Jehan *(Spotify Research, USA)*
- **Rethinking the Detection of Child Sexual Abuse Imagery on the Internet** 2601
  Elie Bursztein, Einat Clarke *(Google, USA)*, Michelle DeLaune, David M. Elifff, Nick Hsu, Lindsey Olson, John Shehan *(NCMEC, USA)*, Madhukar Thakur, Kurt Thomas, Travis Bright *(Google, USA)*
- **The Illusion of Change: Correcting for Biases in Change Inference for Sparse, Societal-Scale Data** 2608
  Gabriel Cadamuro, Ramya Korlakai Vinayak *(Paul G.Allen School of Computer Science and Engineering, University of Washington, USA)*, Joshua Blumenstock *(School of Information, University of California, USA)*, Sham Kakade *(Paul G.Allen School of Computer Science and Engineering, University of Washington, USA)*, Jacob Shapiro *(Woodrow Wilson School of Public & International Affairs, Princeton University, USA)*
- **Rating Augmentation with Generative Adversarial Networks towards Accurate Collaborative Filtering** 2616
  Dong-Kyu Chae, Jin-Soo Kang, Sang-Wook Kim *(Hanyang University, Republic of Korea)*, Jaeho Choi *(Naver Corporation, Republic of Korea)*
- **On the Impact of Choice Architectures on Inequality in Online Donation Platforms** 2623
  Abhijnan Chakraborty *(Max Planck Institute for Software Systems, Germany)*, Nuno Mota *(Ecole Polytechnique Fédérale de Lausanne, Switzerland)*, Asia J. Biega *(Max Planck Institute for Informatics, Germany)*, Krishna P. Gummadi *(MPI-SWS, Germany)*, Hoda Heidari *(ETH Zurich, Switzerland)*
- **Multi-Domain Gated CNN for Review Helpfulness Prediction** 2630
  Cen Chen *(Ant Financial, Singapore)*, Minghui Qiu *(Singapore Management University, Singapore)*, Yinfei Yang *(Redfin Inc., USA)*, Jun Zhou *(Ant Financial, China)*, Jun Huang *(alibaba cloud, China)*, Xiaolong Li *(Ant Financial Group, China)*, Forrest Sheng Bao *(Iowa State University, USA)*
- **Collaborative Similarity Embedding for Recommender Systems** 2637
  Chih-Ming Chen *(National Chengchi University, Academia Sinica, Taiwan Roc)*, Chuan-Ju Wang *(Academia Sinica, Taiwan Roc)*, Ming-Feng Tsai *(National Chengchi University, Taiwan Roc)*, Yi-Hsuan Yang *(Academia Sinica, Taiwan Roc)*
- **Sampled in Pairs and Driven by Text: A New Graph Embedding Framework** 2644
  Liheng Chen, Yanru Qu, Zhenghui Wang, Lin Qiu, Weinan Zhang *(Apex Data & Knowledge Management Lab, Shanghai Jiao Tong University, China)*, Ken Chen, Shaodian Zhang *(Synyi LLC., China)*, Yong Yu *(Apex Data & Knowledge Management Lab, Shanghai Jiao Tong University, China)*
- **BoFGAN: Towards A New Structure of Backward-or-Forward Generative Adversarial Nets** 2652
  M.K.Sophie Chen, Xinyi Lin *(Peking University, China)*, Chen Wei *(Turing Robot, China)*, Rui Yan *(Peking University, China)*

- **Outage Prediction and Diagnosis for Cloud Service Systems** 2659
  Yujun Chen *(Beihang University, Microsoft Research, China)*, Xian Yang, Qingwei Lin *(Microsoft, China)*, Hongyu Zhang *(University of Newcastle, Australia)*, Feng Gao, Zhangwei Xu, Yingnong Dang *(Microsoft, USA)*, Dongmei Zhang, Hang Dong, Yong Xu, Hao Li, Yu Kang *(Microsoft, China)*
- **What Makes a Good Team? A Large-scale Study on the Effect of Team Composition in Honor of Kings** 2666
  Ziqiang Cheng, Yang Yang *(Zhejiang University, China)*, Chenhao Tan *(University of Colorado Boulder, USA)*, Denny Cheng, Alex Cheng *(Tencent Corporation, China)*, Yueting Zhuang *(Zhejiang University, China)*
- **TiFi: Taxonomy Induction for Fictional Domains** 2673
  Cuong Xuan Chu, Simon Razniewski, Gerhard Weikum *(Max Planck Institute for Informatics, Germany)*
- **Deriving User- and Content-specific Rewards for Contextual Bandits** 2680
  Paolo Dragone *(University of Trento, Italy)*, Rishabh Mehrotra, Mounia Lalmas *(Spotify, United Kingdom)*
- **Pcard: Personalized Restaurants Recommendation from Card Payment Transaction Records** 2687
  Min Du *(University of California, USA)*, Robert Christensen, Wei Zhang *(Visa Research, USA)*, Feifei Li *(University of Utah, USA)*
- **Improving Multiclass Classification in Crowdsourcing by Using Hierarchical Schemes** 2694
  Xiaoni Duan, Keishi Tajima *(Kyoto University, Japan)*
- **Modeling the Factors of User Success in Online Debate** 2701
  Esin Durmus, Claire Cardie *(Cornell University, USA)*
- **HopRank: How Semantic Structure Influences Teleportation in PageRank (A Case Study on BioPortal)** 2708
  Lisette Espín-Noboa *(GESIS & Uni. Koblenz-Landau)*, Florian Lemmerich *(RWTH Aachen University, Germany)*, Simon Walk *(Detego GmbH, Austria)*, Markus Strohmaier *(RWTH Aachen University, Germany)*, Mark Musen *(Stanford University, USA)*
- **Product-Aware Helpfulness Prediction of Online Reviews** 2715
  Miao Fan *(BAIDU RESEARCH, China)*, Chao Feng, Lin Guo *(Baidu Research, China)*, Mingming Sun *(BAIDU RESEARCH, China)*, Ping Li *(BAIDU RESEARCH, USA)*
- **The World Wants Mangoes and Kangaroos: A Study of New Emoji Requests Based on Thirty Million Tweets** 2722
  Yunhe Feng *(University of Tennessee, USA)*, Wenjun Zhou *(University of Tennessee Knoxville, USA)*, Zheng Lu *(University of Tennessee, USA)*, Zhibo Wang *(Wuhan University, China)*, Qing Cao *(University of Tennessee, USA)*
- **Online Learning for Measuring Incentive Compatibility in Ad Auctions** 2729
  Zhe Feng *(Harvard University, USA)*, Okke Schrijvers, Eric Sodomka *(Facebook Research, USA)*
- **TableNet: An Approach for Determining Fine-grained Relations for Wikipedia Tables** 2736
  Besnik Fetahu, Avishek Anand *(L3S Research Center, Germany)*, Maria Koutraki *(FIZ Karlsruhe, AIFB KIT, Germany)*
- **Learning Graph Pooling and Hybrid Convolutional Operations for Text Representations** 2743
  Hongyang Gao *(Texas A&M University, USA)*, Yongjun Chen *(Washington State University, USA)*, Shuiwang Ji *(Texas A&M University, USA)*
- **Predicting Human Mobility via Variational Attention** 2750
  Qiang Gao, Fan Zhou *(University of Science and Technology of China, China)*, Goce Trajcevski *(Iowa State University, USA)*, Kunpeng Zhang *(University of Maryland, USA)*, Ting Zhong, Fengli Zhang *(University of Science and Technology of China, China)*
- **Maximizing Marginal Utility per Dollar for Economic Recommendation** 2757
  Yingqiang Ge, Shuyuan Xu, Shuchang Liu, Shijie Geng, Zuohui Fu, Yongfeng Zhang *(Rutgers University, USA)*
- **The few-get-richer: a surprising consequence of popularity-based rankings** 2764
  Fabrizio Germano, Vicenç Gómez, Gaël Le Mens *(Universitat Pompeu Fabra, Spain)*
- **Context-Sensitive Malicious Spelling Error Correction** 2771
  Hongyu Gong, Yuchen Li, Suma Bhat, Pramod Viswanath *(University of Illinois at Urbana-Champaign, USA)*
- **With a Little Help from My Friends (and Their Friends): Influence Neighborhoods for Social Recommendations** 2778
  Avni Gulati, Magdalini Eirinaki *(San Jose State University, USA)*

- **Personalized Online Spell Correction for Personal Search** 2785
  Jai Gupta, Zhen Qin, Michael Bendersky, Donald Metzler *(Google, USA)*
- **Inferring Search Queries from Web Documents via a Graph-Augmented Sequence to Attention Network** 2792
  Fred.X Han, Di Niu *(University of Alberta, Canada)*, Kunfeng Lai, Weidong Guo, Yancheng He, Yu Xu *(Tencent, China)*
- **Learning Novelty-Aware Ranking of Answers to Complex Questions** 2799
  Shahar Harel *(Technion – Israel Institute of Technology, Israel)*, Sefi Albo *(Technion, Israel)*, Eugene Agichtein *(Emory University, USA)*, Kira Radinsky *(Technion – Israel Institute of Technology, Israel)*
- **Spatio-Temporal Capsule-based Reinforcement Learning for Mobility-on-Demand Network Coordination** 2806
  Suining He, Kang G. Shin *(University of Michigan, USA)*
- **An Investigation of Cyber Autonomy on Government Websites** 2814
  Hsu-Chun Hsiao *(National Taiwan University, Taiwan Roc)*, Tiffany Hyun-Jin Kim *(HRL Laboratories, USA)*, Yu-Ming Ku, Chun-Ming Chang, Hung-Fang Chen, Yu-Jen Chen *(National Taiwan University, Taiwan Roc)*, Chun-Wen Wang *(Cornell University, USA)*, Wei Jeng *(National Taiwan University, Taiwan Roc)*
- **Transfer Meets Hybrid: A Synthetic Approach for Cross-Domain Collaborative Filtering with Text** 2822
  Guangneng Hu, Yu Zhang, Qiang Yang *(The Hong Kong University of Science and Technology, China)*
- **Unbiased LambdaMART: An Unbiased Pairwise Learning-to-Rank Algorithm** 2830
  Ziniu Hu *(University of California, USA)*, Yang Wang, Qu Peng, Hang Li *(Bytedance AI Lab, China)*
- **Domain-aware Neural Model for Sequence Labeling using Joint Learning** 2837
  Huang Heng, Liu Xiaozhong *(Alibaba group, China)*, Yuliang Yan *(Alibaba, China)*
- **A Multimodal Text Matching Model for Obfuscated Language Identification in Adversarial Communication** 2844
  Longtao Huang, Ting Ma, Junyu Lin *(Institute of Information Engineering, Chinese Academy of Sciences, China)*, Jizhong Han *(Institute of Information Engineering, China)*, Songlin Hu *(Institute of Information Engineering, Chinese Academy of Sciences, China)*
- **The Chain of Implicit Trust: An Analysis of the Web Third-party Resources Loading** 2851
  Muhammad Ikram *(University of Michigan and Macquarie University, Australia)*, Rahat Masood *(UNSW and Data61, Australia)*, Gareth Tyson *(Queen Mary University of London and Alan Turing Institute, United Kingdom)*, Mohamed Ali Kaafar *(Data61 and Macquarie University, Australia)*, Noha Loizon *(Data61, France)*, Roya Ensafi *(University of Michigan, USA)*
- **Efficient Interaction-based Neural Ranking with Locality Sensitive Hashing** 2858
  Shiyu Ji, Jinjin Shao, Tao Yang *(University of California at Santa Barbara, USA)*
- **Triple Trustworthiness Measurement for Knowledge Graph** 2865
  Shengbin Jia, Yang Xiang, Xiaojun Chen *(Tongji University, China)*, Kun Wang *(East China Normal University, China)*, Shijia *(Tongji University, China)*
- **A Tree-Structured Neural Network Model for Household Energy Breakdown** 2872
  Yiling Jia *(University of Virginia, USA)*, Nipun Batra *(IIT Gandhinagar, India)*, Hongning Wang, Kamin Whitehouse *(University of Virginia, USA)*
- **Improving Neural Response Diversity with Frequency-Aware Cross-Entropy Loss** 2879
  Shaojie Jiang, Pengjie Ren, Christof Monz, Maarten de Rijke *(University of Amsterdam, Netherlands)*
- **A Novel Generative Topic Embedding Model by Introducing Network Communities** 2886
  Di Jin, Jiantao Huang, Pengfei Jiao *(College of Intelligence and Computing, Tianjin University, China)*, Liang Yang *(Hebei University of Technology, China)*, Dongxiao He *(College of Intelligence and Computing, Tianjin University, China)*, Fran??oise Soulie-Fogelman *(College of Intelligence and Computing, Tianjin University, France)*, Yuxiao Huang *(George Washington University, China)*
- **A Scalable Hybrid Research Paper Recommender System for Microsoft Academic** 2893
  Anshul Kanakia, Zhihong Shen, Darrin Eide, Kuansan Wang *(Microsoft, USA)*
- **Topic Structure-Aware Neural Language Model: Unified language model that maintains word and topic ordering by their embedded representations** 2900
  Noriaki Kawamae *(The University of Tokyo, Japan)*

- **Fairness in Algorithmic Decision Making: An Excursion Through the Lens of Causality** 2907
  Aria Khademi *(The Pennsylvania State University, USA)*, Sanghack Lee *(Purdue University, USA)*, David Foley, Vasant Honavar *(The Pennsylvania State University, USA)*
- **MVAE: Multimodal Variational Autoencoder for Fake News Detection** 2915
  Dhruv Khattar, Jaipal Singh Goud, Manish Gupta, Vasudeva Varma *(International Institute of Information Technology Hyderabad, India)*
- **Mobile App Risk Ranking via Exclusive Sparse Coding** 2922
  Deguang Kong *(Yahoo Research, USA)*, Lei Cen *(Purdue University, USA)*
- **A Neural Bag-of-Words Modelling Framework for Link Prediction in Knowledge Bases with Sparse Connectivity** 2929
  Fanshuang Kong, Richong Zhang *(Beihang University, China)*, Hongyu Guo *(National Research Council Canada, Canada)*, Samuel Mensah, Zhiyuan Hu *(Beihang University, China)*, Yongyi Mao *(University of Ottawa, Canada)*
- **FARE: Diagnostics for Fair Ranking using Pairwise Error Metrics** 2936
  Caitlin Kuhlman, Maryann Vanvalkenburg, Elke Rundensteiner *(Worcester Polytechnic Institute, USA)*
- **Adversarial Adaptation of Scene Graph Models for Understanding Civic Issues** 2943
  Shanu Kumar *(IIT Kanpur, India)*, Shubham Atreja, Anjali Singh, Mohit Jain *(IBM, India)*
- **Redesigning Bitcoin's fee market** 2950
  Ron Lavi *(Technion -- Israel Institute of Technology, Israel)*, Or Sattath *(Hebrew University, Israel)*, Aviv Zohar *(The Hebrew University of Jerusalem, Israel)*
- **Measuring Political Personalization of Google News Search** 2957
  Huyen Le *(Department of Computer Science, The University of Iowa, USA)*, Raven Maragh *(Department of Communication Studies, The University of Iowa, USA)*, Brian Ekdale *(School of Journalism and Mass Communication, The University of Iowa, USA)*, Timothy Havens, Andrew High *(Department of Communication Studies, The University of Iowa, USA)*, Zubair Shafiq *(Department of Computer Science, The University of Iowa, USA)*
- **TiSSA: A Time Slice Self-Attention Approach for Modeling Sequential User Behaviors** 2964
  Chenyi Lei *(Alibaba Group, Hangzhou, China)*, Shouling Ji *(Zhejiang University Hangzhou, China)*, Zhao Li *(Alibaba Group Hangzhou, China)*
- **Search Mindsets** 2971
  Ang Li *(University of Pittsburgh, USA)*, Jennifer Thom, Praveen Chandar, Christine Hosey, Brian St. Thomas, Jean Garcia-Gathright *(Spotify, USA)*
- **Click Feedback-Aware Query Recommendation Using Adversarial Examples** 2978
  Ruirui Li *(University of California, USA)*, Liangda Li *(Yahoo Research, USA)*, Xian Wu *(University of Notre Dame, USA)*, Yunhong Zhou *(Yahoo Research, USA)*, Wei Wang *(University of California, USA)*
- **Learning Fast Matching Models from Weak Annotations** 2985
  Xue Li, Zhipeng Luo, Hao Sun, Jianjin Zhang, Weihao Han, Xianqi Chu, Liangjie Zhang, Qi Zhang *(Microsoft China, China)*
- **Multistream Classification for Cyber Threat Data with Heterogeneous Feature Space** 2992
  Yi-Fan Li, Yang Gao, Gbadebo Ayoade, Khan Latifur, Hemeng Tao, Bhavani Thuraisingham *(The University of Texas at Dallas, USA)*
- **Predicting pregnancy using large-scale datafrom a women's health tracking mobile application** 2999
  Bo Liu, Shuyang Shi, Yongshang Wu *(Stanford, USA)*, Daniel Thomas *(Clue by BioWink GmbH, Germany)*, Laura Symul, Emma Pierson, Jure Leskovec *(Stanford, USA)*
- **Fuzzy Multi-task Learning for Hate Speech Type Identification** 3006
  Han Liu, Pete Burnap, Wafa Alorainy *(Cardiff University, United Kingdom)*, Matthew Williams *(School of Social Science, Cardiff University, United Kingdom)*
- **Neural Chinese Word Segmentation with Lexicon and Unlabeled Data via Posterior Regularization** 3013
  Junxin Liu *(Tsinghua University, China)*, Fangzhao Wu *(Microsoft, China)*, Chuhan Wu, Yongfeng Huang *(Tsinghua University, China)*, Xing Xie *(Microsoft, China)*
- **User-Video Co-Attention Network for Personalized Micro-video Recommendation** 3020
  Shang Liu, Zhenzhong Chen *(Wuhan University, China)*, Hongyi Liu *(Amazon, USA)*, Xinghai Hu *(Facebook, USA)*

- **Recommender Systems with Heterogeneous Side Information**.....3027
  Tianqiao Liu *(TAL AI Lab, China)*, Zhiwei Wang, Jiliang Tang *(Michigan State University, USA)*, Songfan Yang, Zitao Liu *(TAL AI Lab, China)*
- **Globally-Optimized Realtime Supply-Demand Matching in On-Demand Ridesharing**.....3034
  Yifang Liu *(Smule Inc., USA)*, Shaosu Liu *(Affiliation N/A, USA)*, Will Skinner *(Engine ML Inc., USA)*, Chongyuan Xiang *(Opendoor Labs Inc., USA)*
- **MCVAE: Margin-based Conditional Variational Autoencoder for Relation Classification and Pattern Generation**.....3041
  Fenglong Ma *(University at Buffalo, USA)*, Yaliang Li *(Tencent Medical AI Lab, USA)*, Chenwei Zhang *(University of Illinois at Chicago, USA)*, Jing Gao *(University at Buffalo, USA)*, Nan Du, Wei Fan *(Tencent Medical AI Lab, USA)*
- **Detect Rumors on Twitter by Promoting Information Campaigns with Generative Adversarial Learning**.....3049
  Jing Ma *(The Chinese University of Hong Kong, Hong Kong)*, Wei Gao *(Victoria University of Wellington, New Zealand)*, Kam-Fai Wong *(The Chinese University of Hong Kong, Hong Kong)*
- **Parametric Models for Intransitivity in Pairwise Rankings**.....3056
  Rahul Makhijani, Johan Ugander *(Stanford University, USA)*
- **A Large-scale Study on the Risks of the HTML5 WebAPI for Mobile Sensor-based Attacks**.....3063
  Francesco Marcantoni *(University of Illinois at Chicago, USA)*, Michalis Diamantaris, Sotiris Ioannidis *(FORTH, Greece)*, Jason Polakis *(University of Illinois at Chicago, USA)*
- **Pythia : a Framework for the Automated Analysis of Web Hosting Environments**.....3072
  Srdjan Matic *(University College London, United Kingdom)*, Gareth Tyson *(Queen Mary University of London, United Kingdom)*, Gianluca Stringhini *(Boston University, USA)*
- **Event Detection using Hierarchical Multi-Aspect Attention**.....3079
  Sneha Mehta, M. Raihanul Islam *(Virginia Polytechnic Institute, USA)*, Huzefa Rangwala *(George Mason University, USA)*, Naren Ramakrishnan *(Virginia Polytechnic Institute, USA)*
- **Signals Matter: Understanding Popularity and Impact of Users on Stack Overflow**.....3086
  Arpit Merchant, Daksh Shah, Gurpreet Singh Bhatia *(Indraprastha Institute of Information Technology, Delhi, India)*, Anurag Ghosh *(IIIT Hyderabad, India)*, Ponnurangam Kumaraguru *(Indraprastha Institute of Information Technology, Delhi, India)*
- **Fine-grained Type Inference in Knowledge Graphs via Probabilistic and Tensor Factorization Methods**.....3093
  A B M Moniruzzaman, Richi Nayak, Maolin Tang, Thirunavukarasu Balasubramaniam *(Queensland University of Technology, Australia)*
- **Augmenting Knowledge Tracing by Considering Forgetting Behavior**.....3101
  Koki Nagatani, Qian Zhang, Masahiro Sato *(Fuji Xerox Co., Ltd., Japan)*, Yan-Ying Chen, Francine Chen *(FX Palo Alto Laboratory, USA)*, Tomoko Ohkuma *(Fuji Xerox Co., Ltd., Japan)*
- **Think Outside the Dataset: Finding Fraudulent Reviews using Cross-Dataset Analysis**.....3108
  Shirin Nilizadeh *(University of Texas at Arlington, USA)*, Hojjat Aghakhani, Eric Gustafson, Christopher Kruegel, Giovanni Vigna *(University of California, USA)*
- **Entity Personalized Talent Search Models with Tree Interaction Features**.....3116
  Cagri Ozcaglar, Sahin Geyik, Brian Schmitz, Prakhar Sharma, Alex Shelkovnykov, Yiming Ma, Erik Buchanan *(LinkedIn, USA)*
- **Value-aware Recommendation based on Reinforcement Profit Maximization**.....3123
  Changhua Pei *(Alibaba Group, China)*, Xinru Yang *(CMU, USA)*, Qing Cui, Xiao Lin, Fei Sun, Peng Jiang, Wenwu Ou *(Alibaba Group, China)*, Yongfeng Zhang *(Rutgers University, USA)*
- **Semi-Supervised Entity Alignment via Knowledge Graph Embedding with Awareness of Degree Difference**.....3130
  Shichao Pei, Lu Yu, Robert Hoehndorf, Xiangliang Zhang *(King Abdullah University of Science and Technology, Saudi Arabia)*
- **Event-Driven Analysis of Crowd Dynamics in the *Black Lives Matter* Online Social Movement**.....3137
  Hao Peng, Ceren Budak, Daniel Romero *(University of Michigan, USA)*

- **CnGAN: Generative Adversarial Networks for Cross-network user preference generation for non-overlapped users** 3144
  Dilruk Perera, Roger Zimmermann *(National University of Singapore, Singapore)*
- **Learning Clusters through Information Diffusion** 3151
  Liudmila Prokhorenkova *(Moscow Institute of Physics and Technology and Yandex, Russian Fed.)*, Alexey Tikhonov *(Yandex, Germany)*, Nelly Litvak *(University of Twente, Netherlands)*
- **Constructing Test Collections using Multi-armed Bandits and Active Learning** 3158
  Md Mustafizur Rahman *(The University of Texas at Austin, USA)*, Mucahid Kutlu *(TOBB University of Economics and Technology, Turkey)*, Matthew Lease *(The University of Texas at Austin, USA)*
- **A Multi-modal Neural Embeddings Approach for Detecting Mobile Counterfeit Apps** 3165
  Jathushan Rajasegaran, Naveen Karunanayake *(CSIRO, Australia)*, Ashanie Gunathillake, Suranga Seneviratne *(The University of Sydney, Australia)*, Guillaume Jourjon *(CSIRO, Australia)*
- **Context-Aware Sequential Recommendations withStacked Recurrent Neural Networks** 3172
  Lakshmanan Rakkappan, Vaibhav Rajan *(National University of Singapore, Singapore)*
- **Improved Cross-Lingual Question Retrieval for Community Question Answering** 3179
  Andreas Rücklé, Krishnkant Swarnkar, Iryna Gurevych *(TU Darmstadt, Germany)*
- **TurkScanner: Predicting the Hourly Wage of Microtasks** 3187
  Susumu Saito *(Waseda University, Japan)*, Chun-Wei Chiang, Saiph Savage *(West Virginia University, USA)*, Teppei Nakano, Tetsunori Kobayashi *(Waseda University, Japan)*, Jeffrey Bigham *(Carnegie Mellon University, USA)*
- **A Large-scale Study of Wikipedia Users' Quality of Experience** 3194
  Flavia Salutari, Diego Da Hora *(Telecom ParisTech)*, Gilles Dubuc *(Wikimedia Foundation)*, Dario Rossi *(Huawei Technologies, Co. Ltd.)*
- **Genre Differences of Song Lyrics and Artist Wikis: An Analysis of Popularity, Length, Repetitiveness, and Readability** 3201
  Markus Schedl *(Johannes Kepler University, Austria)*
- **Growing Attributed Networks through Local Processes** 3208
  Harshay Shah *(University of Illinois at Urbana-Champaign, USA)*, Suhansanu Kumar *(University of Illinois, Urbana Champaign, USA)*, Hari Sundaram *(University of Illinois at Urbana-Champaign, USA)*
- **ϵ-Diagnosis: Unsupervised and Real-time Diagnosis of Small- window Long-tail Latency in Large-scale Microservice Platforms** 3215
  Huasong Shan, Yuan Chen *(JD.com Silicon Valley R&D Center, USA)*, Haifeng Liu, Yunpeng Zhang, Xiao Xiao, Xiaofeng He *(JD.com Inc., China)*, Min Li, Wei Ding *(JD.com Silicon Valley R&D Center, USA)*
- **Adaptive matrix completion for the users and the items in tail** 3223
  Mohit Sharma *(WalmartLabs, USA)*, George Karypis *(University of Minnesota, USA)*
- **What is in Your Password? Analyzing Memorable and Secure Passwords using a Tensor Decomposition** 3230
  Youjin Shin *(Stonybrook University, Republic of Korea)*, Simon Woo *(SUNY Korea, USA)*
- **Understanding Reader Backtracking Behavior in Online News Articles** 3237
  Uzi Smadja *(Technion – Israel Institute of Technology, Israel)*, Max Grusky, Yoav Artzi, Mor Naaman *(Cornell Tech, USA)*
- **Unnecessarily Identifiable: Quantifying the fingerprintability of browser extensions due to bloat** 3244
  Oleksii Starov, Pierre Laperdrix *(Stony Brook University, USA)*, Alexandros Kapravelos *(North Carolina State University, USA)*, Nick Nikiforakis *(Stony Brook University, USA)*
- **Embarrassingly Shallow Autoencoders for Sparse Data** 3251
  Harald Steck *(Netflix, USA)*
- **Rock, Rap, or Reggaeton?: Assessing Mexican Immigrants' Cultural Assimilation Using Facebook Data** 3258
  Ian Stewart *(Georgia Institute of Technology, USA)*, René D. Flores *(University of Chicago, USA)*, Timothy Riffe *(Max Planck Institute for Demographic Research, Germany)*, Ingmar Weber *(Qatar Computing Research Institute, Qatar)*, Emilio Zagheni *(Max Planck Institute for Demographic Research, Germany)*
- **Learning Intent to Book Metrics for Airbnb Search** 3265
  Bradley Turnbull *(Airbnb, USA)*

- **Detection and Analysis of Self-Disclosure in Online News Commentaries** 3272
  Prasanna Umar, Anna Squicciarini, Sarah Rajtmajer *(The Pennsylvania State University, USA)*
- **ViTOR: Learning to Rank Webpages Based on Visual Features** 3279
  Bram van den Akker, Ilya Markov, Maarten de Rijke *(University of Amsterdam, Netherlands)*
- **Evaluating Neural Text Simplification in the Medical Domain** 3286
  Laurens van den Bercken *(Delft University of Technology and myTomorrows, Netherlands)*, Robert-Jan Sips *(myTomorrows, Netherlands)*, Christoph Lofi *(Delft University of Technology, Netherlands)*
- **Semantic Hilbert Space for Text Representation Learning** 3293
  Benyou Wang, Qiuchi Li, Massimo Melucci *(University of Padua, Italy)*, Dawei Song *(Beijing Institute of Technology & The Open University, China)*
- **Learning Task-Specific City Region Partition** 3300
  Hongjian Wang, Porter Jenkins, Hua Wei, Fei Wu, Zhenhui Li *(The Pennsylvania State University, USA)*
- **Knowledge Graph Convolutional Networks for Recommender Systems** 3307
  Hongwei Wang *(Shanghai Jiao Tong University, China)*, Miao Zhao *(The Hong Kong Polytechnic University, Hong Kong)*, Xing Xie *(Microsoft, China)*, Wenjie Li *(The Hong Kong Polytechnic University, Hong Kong)*, Minyi Guo *(Shanghai Jiao Tong University, China)*
- **Tag2Vec: Learning Tag Representations in Tag Networks** 3314
  Junshan Wang, Zhicong Lu *(Peking University, China)*, Guojie Song *(Room 2307 in Science Building Two, Peking University, China)*, Yue Fan, Lun Du *(Peking University, China)*
- **The Silent Majority Speaks: Inferring Silent Users' Opinions in Online Social Networks** 3321
  Lei Wang, Jianwei Niu, Xuefeng Liu, Kaili Mao *(Beihang University, China)*
- **A Novel Unsupervised Approach for Precise Temporal Slot Filling from Incomplete and Noisy Temporal Contexts** 3328
  Xueying Wang, Haiqiao Zhang, Qi Li, Yiyu Shi, Meng Jiang *(University of Notre Dame, China)*
- **Learning Binary Hash Codes for Fast Anchor Link Retrieval across Networks** 3335
  Yongqing Wang *(Institute of Computing Technology, Chinese Academy of Sciences, China)*, Huawei Shen *(Chinese Academy of Sciences, China)*, Jinhua Gao *(ICT, China)*, Xueqi Cheng *(Institute of Computing Technology, Chinese Academy of Sciences, China)*
- **Neural Chinese Named Entity Recognition via CNN-LSTM-CRF and Joint Training with Word Segmentation** 3342
  Fangzhao Wu *(Microsoft, China)*, Junxin Liu, Chuhan Wu, Yongfeng Huang *(Tsinghua University, China)*, Xing Xie *(Microsoft, China)*
- **Semi-supervised Multi-view Individual and Sharable Feature Learning for Webpage Classification** 3349
  Fei Wu *(Nanjing University of Posts and Telecommunications, China)*, Xiao-Yuan Jing *(Wuhan University, Nanjing University of Posts and Telecommunications, China)*, Jun Zhou, Yimu Ji *(Nanjing University of Posts and Telecommunications, China)*, Chao Lan *(University of Wyoming, USA)*, Qinghua Huang, Ruchuan Wang *(Nanjing University of Posts and Telecommunications, China)*
- **On Convexity and Bounds of Fairness-aware Classification** 3356
  Yongkai Wu, Lu Zhang, Xintao Wu *(University of Arkansas, USA)*
- **Understanding Urban Dynamics via State-sharing Hidden Markov Model** 3363
  Tong Xia, Yue Yu, Fengli Xu, Yong Li *(Tsinghua University, China)*
- **Efficient Path Prediction for Semi-Supervised and Weakly Supervised Hierarchical Text Classification** 3370
  Huiru Xiao, Xin Liu, Yangqiu Song *(The Hong Kong University of Science and Technology, Hong Kong)*
- **Hierarchical Neural Variational Model for Personalized Sequential Recommendation** 3377
  Xiao Teng, Shangsong Liang, Zaiqiao Meng *(Sun Yat-sen University, China)*
- **Focusing Attention Network for Answer Ranking** 3384
  Yufei Xie, Shuchun Liu, Tangren Yao, Yao Peng *(Qiniu AtLab, China)*, Zhao Lu *(East China Normal University, China)*

- **DLocRL: A Deep Learning Pipeline for Fine-Grained Location Recognition and Linking in Tweets** 3391
  Canwen Xu *(School of Computer Science, Wuhan University, China)*, Jing Li *(Nanyang Technological University, Singapore)*, Xiangyang Luo *(State Key Lab of Mathematical Engineering and Advanced Computing, China)*, Jiaxin Pei *(School of Computer Science, Wuhan University, China)*, Chenliang Li, Donghong Ji *(School of Cyber Science and Engineering, Wuhan University, China)*
- **Recurrent Convolutional Neural Network for Sequential Recommendation** 3398
  Chengfeng Xu, Pengpeng Zhao *(Soochow University, China)*, Yanchi Liu *(Rutgers University, USA)*, Jiajie Xu *(Soochow University, China)*, Victor Sheng *(University of Central Arkansas, USA)*, Zhiming Cui *(Suzhou University of Science and Technology, China)*, Xiaofang Zhou *(The University of Queensland, Australia)*, Hui Xiong *(the State University of New Jersey, USA)*
- **No More than What I Post: Preventing Linkage Attacks on Check-in Services** 3405
  Fengli Xu, Zhen Tu, Hongjia Huang, Shuhao Chang, Yong Li *(Tsinghua University, China)*
- **Open-world Learning and Application to Product Classification** 3413
  Hu Xu, Bing Liu, Lei Shu, Philip S. Yu *(University of Illinois at Chicago, USA)*
- **Place Deduplication with Embeddings** 3420
  Carl Yang *(University of Illinois at Urbana-Champaign, USA)*, Huy Hoang Do, Tomas Mikolov *(Facebook Inc., USA)*, Jiawei Han *(University of Illinois at Urbana-Champaign, USA)*
- **Cyberbullying Ends Here: Towards Robust Detection of Cyberbullying in Social Media** 3427
  Mengfan Yao, Charalampos Chelmis, Daphney-Stavroula Zois *(University at Albany State University of New York, USA)*
- **Enhancing Fashion Recommendation with Visual Compatibility Relationship** 3434
  Ruiping Yin, Kan Li *(Beijing Institute of Technology, China)*, Jie Lu, Guangquan Zhang *(University of Technology, Australia)*
- **Discovering Product Defects and Solutions from Online User Generated Contents** 3441
  Xuan Zhang *(Virginia Tech, USA)*, Zhilei Qiao *(The University of Alabama at Birmingham, USA)*, Aman Ahuja *(Virginia Tech, USA)*, Weiguo Fan *(University of Iowa, USA)*, Edward Fox, Chandan K. Reddy *(Virginia Tech, USA)*
- ***Your Style Your Identity*: Leveraging Writing and Photography Styles for Drug Trafficker Identification in Darknet Markets over Attributed Heterogeneous Information Network** 3448
  Yanfang Ye *(West Virginia University, USA)*, Yiming Zhang, Yujie Fan, Shifu Hou, Xin Li *(West Virginia University, USA)*
- **Abstractive Meeting Summarization via Hierarchical Adaptive Segmental Network Learning** 3455
  Zhou Zhao, Haojie Pan *(Zhejiang University, China)*, Changjie Fan *(NetEase Fuxi AI Lab, China)*, Deng Cai *(Zhejiang University, China)*, Min Yang *(Shenzhen Institutes of Advanced Technology, China)*
- **Adversarial Point-of-Interest Recommendation** 3462
  Fan Zhou, Ruiyang Yin *(University of Science and Technology of China, China)*, Kunpeng Zhang *(University of Maryland, USA)*, Goce Trajcevski *(Iowa State University, USA)*, Ting Zhong, Jin Wu *(University of Science and Technology of China, China)*
- **Context-aware Variational Trajectory Encoding and Human Mobility Inference** 3469
  Fan Zhou, Xiaoli Yue *(University of Science and Technology of China, China)*, Goce Trajcevski *(Iowa State University, USA)*, Ting Zhong *(University of Science and Technology of China, China)*, Kunpeng Zhang *(University of Maryland, USA)*
- **Variational Session-based Recommendation Using Normalizing Flows** 3476
  Fan Zhou, Zijing Wen *(University of Science and Technology of China, China)*, Kunpeng Zhang *(University of Maryland, USA)*, Goce Trajcevski *(Iowa State University, USA)*, Ting Zhong *(University of Science and Technology of China, China)*
- **Improving Top-K Recommendation via JointCollaborative Autoencoders** 3483
  Ziwei Zhu, Jianling Wang, James Caverlee *(Texas A&M University, USA)*

Demos

- **Enlisting the Public to Build a Healthier Web Information Commons** 3490
  Update title and chair/organizer names here *(& their affiliations)*

- **RecBoard: A Web-based Platform for Recommendation System Research and Development** 3493
  Kriti Singh, Mohit Chawla, Longqi Yang, Deborah Estrin *(Cornell University, USA)*
- **CrowdPT: Summarizing Crowd Opinions as Professional Analyst** 3498
  Chung-Chi Chen, Hen-Hsen Huang, Chia-Wen Tsai, Hsin-Hsi Chen *(National Taiwan University, Taiwan)*
- **TIM: A Tool for Gaining Insights into Psychotherapy** 3503
  Ronan Cummins, Michael Ewbank, Alan Martin, Valentin Tablan, Ana Catarino, Andrew Blackwell *(Ieso Digital Health, United Kingdom)*
- **QAnswer: A Question Answering prototype bridging the gap between a considerable part of the LOD cloud and end-users** 3507
  Dennis Diefenbach *(University of Lyon, France)*, Pedro Migliatti *(Universidade Federal de São Carlos, Brazil)*, Omar Qawasmeh *(University of Lyon, France)*, Vincent Lully *(Sorbonne University, France)*, Kamal Singh, Pierre Maret *(University of Lyon, France)*
- **Voyageur: An Experiential Travel Search Engine** 3511
  Sara Evensen *(Megagon Labs, USA)*, Aaron Feng *(Megagon Labs, Japan)*, Alon Halevy, Jinfeng Li, Vivian Li, Yuliang Li, Huining Liu, George Mihaila, John Morales, Natalie Nuno, Ekaterina Pavlovic, Wang-Chiew Tan, Xiaolan Wang *(Megagon Labs, USA)*
- **Tracy: Tracing Facts over Knowledge Graphs and Text** 3516
  Mohamed H. Gad-Elrab *(Max Planck Institute for Informatics, Germany)*, Daria Stepanova *(Bosch Center for Artificial Intelligence, Germany)*, Jacopo Urbani *(Vrije Universiteit Amsterdam, Netherlands)*, Gerhard Weikum *(Max Planck Institute for Informatics, Germany)*
- **LearnerExp: Exploring and Explaining the Time Management of Online Learning Activity** 3521
  Huan He, Qinghua Zheng, Bo Dong *(Xi'an Jiaotong University, China)*
- **Gradual Machine Learning for Entity Resolution** 3526
  Boyi Hou, Qun Chen, Xin Liu, Ping Zhong, Yanyan Wang, Zhaoqiang Chen, Zhanhuai Li *(Northwestern Polytechnical University, China)*
- **Dixit: Interactive Visual Storytelling via Term Manipulation** 3531
  Chao-Chun Hsu, Yu-Hua Chen *(Academia Sinica, Taiwan)*, Zi-Yuan Chen *(IIS, Sinica, Taiwan)*, Hsin Yu Lin *(Institute of Information Science, Academia Sinica, Taiwan)*, Ting-Hao Huang *(The Pennsylvania State University, USA)*, Lun-Wei Ku *(Institute of Information Science, Academia Sinica, Taiwan)*
- **Exam Keeper: Detecting Questions with Easy-to-Find Answers** 3536
  Ting-Lun Hsu, Shih-Chieh Dai, Lun-Wei Ku *(Institute of Information Science, Academia Sinica, Taiwan)*
- **ExQuisiTe: Explaining Quantities in Text** 3541
  Yusra Ibrahim, Gerhard Weikum *(Max Planck Institute for Informatics, Germany)*
- **Can You Give Me a Reason?: Argument-inducing Online Forum by Argument Mining** 3545
  Makiko Ida, Gaku Morio, Kosui Iwasa, Takaki Yasui, Tomoyuki Tatsumi, Katsuhide Fujita *(Tokyo University of Agriculture and Technology, Japan)*
- **UNVEIL: Capture and Visualise WiFi Data Leakages** 3550
  Shubham Jain *(Imperial College London, United Kingdom)*, Eden Bensaid *(Massachusetts Institute of Technology, USA)*, Yves-Alexandre de Montjoye *(Imperial College London, United Kingdom)*
- **SmartDBO: Smart Docker Benchmarking Orchestrator for Web-application** 3555
  Devki Nandan Jha, Michael Nee, Zhenyu Wen *(Newcastle University, United Kingdom)*, Albert Zomaya *(The University of Sydney, Australia)*, Rajiv Ranjan *(Newcastle University, United Kingdom)*
- **R2SIGTP: a Novel Real-Time Recommendation System with Integration of Geography and Temporal Preference for Next Point-of-Interest** 3560
  Xu Jiao, Yingyuan Xiao, Wenguang Zheng *(Tianjin University of Technology, China)*
- **SGX-PySpark: Secure Distributed Data Analytics** 3564
  Do Le Quoc, Franz Gregor *(TU Dresden, Germany)*, Jatinder Singh *(University of Cambridge, United Kingdom)*, Christof Fetzer *(TU Dresden, Germany)*
- **TweetSenti: Target-dependent Tweet Sentiment Analysis** 3569
  Quanzhi Li, Qiong Zhang, Luo Si *(Alibaba Group, USA)*
- **Querying Data Lakes using Spark and Presto** 3574
  Mohamed Nadjib Mami, Damien Graux, Simon Scerri *(Fraunhofer, Germany)*, S??ren Auer *(TIB Leibniz Information Center Science and Technology, Germany)*

- **Automated Fact Checking in the News Room** 3579
  Sebastiao Miranda, David Nogueira, Afonso Mendes *(Priberam, Portugal)*, Andreas Vlachos *(Department of Computer Science and Technology, University of Cambridge, United Kingdom)*, Andrew Secker *(BBC News Labs, United Kingdom)*, Rebecca Garrett *(BBC Monitoring, United Kingdom)*, Jeff Mitchel *(Psychology Department, University of Bristol, United Kingdom)*, Zita Marinho *(Priberam, Portugal)*
- **InfraNodus: Generating Insight Using Text Network Analysis** 3584
  Dmitry Paranyushkin *(Nodus Labs, France)*
- **Bridging Screen Readers and Voice Assistants for Enhanced Eyes-Free Web Search** 3590
  Alexandra Vtyurina *(University of Waterloo, Canada)*, Adam Fourney, Meredith Ringel Morris *(Microsoft, USA)*, Leah Findlater *(University of Washington, USA)*, Ryen W. White *(Microsoft, USA)*
- **Graph-based Interactive Data Federation System for Heterogeneous Data Retrieval and Analytics** 3595
  Xuan-Son Vu, Addi Ait-Mlouk, Erik Elmroth, Lili Jiang *(Department of Computing Science, Umeå University, Sweden)*
- **XFake: Explainable Fake News Detector with Visualizations** 3600
  Fan Yang, Shiva Pentyala, Sina Mohseni, Mengnan Du *(Texas A&M University, USA)*, Hao Yuan *(Washington State University, USA)*, Rhema Linder *(Texas A&M University, USA)*, Eric Ragan *(University of Florida, USA)*, Shuiwang Ji, Xia Hu *(Texas A&M University, USA)*
- **GraviTIE: Exploratory Analysis of Large-Scale Heterogeneous Image Collections** 3605
  Sean Yang, Luke Rodriguez, Jevin West, Bill Howe *(University of Washington, USA)*
- **TaxVis: a Visual System for Detecting Tax Evasion Group** 3610
  Yu Hongchao, Huan He, Qinghua Zheng, Bo Dong *(Xi'an Jiaotong University, China)*
- **Tablepedia: Automating PDF Table Reading in an Experimental Evidence Exploration and Analytic System** 3615
  Wenhao Yu *(University of Notre Dame, USA)*, Zongze Li *(Sichuan University, China)*, Qingkai Zeng, Meng Jiang *(University of Notre Dame, USA)*
- **CityFlow: A Multi-Agent Reinforcement Learning Environment for Large Scale City Traffic Scenario** 3620
  Huichu Zhang, Siyuan Feng, Chang Liu, Yaoyao Ding, Yichen Zhu, Zihan Zhou, Weinan Zhang, Yong Yu, Haiming Jin *(Shanghai Jiao Tong University, China)*, Zhenhui Li *(The Pennsylvania State University, USA)*

Author Index 3625

Conference Organization

Conference Chairs: Ricardo Baeza-Yates (NTENT & Northeastern University, USA)
Leila Zia (Wikimedia Foundation, USA)

Program Chairs: Ling Liu (Georgia Tech, USA)
Ryen White (Microsoft Research, USA)

Local Organizing Chairs: Chris Brooks (University of San Francisco, USA)
Regan Robertson (Executivevents, USA)
Bebo White (SLAC National Accelerator Laboratory, USA)

Treasurer: Fabian Suchanek (Télécom ParisTech University, France)

Tutorial Chair: Ashish Goel (Stanford University, USA)

Workshop Chairs: Sihem Amer-Yahia (CNRS & Université Grenoble Alpes, France)
Mohammad Mahdian (Google, USA)

Demo Chairs: Amin Mantrach (Criteo, USA)
Fabrizio Silvestri (Facebook, UK)

PhD Symposium Chairs: Geert-Jan Houben (TU Delft, The Netherlands)
Kristina Lerman (Information Sciences Institute, USA)

BIG Chairs: Vanja Josifovski (Airbnb, USA)
Liane Lewin-Eytan (Amazon, Israel)

Web of Health Chairs: Evgeniy Gabrilovich (Google, USA)
Nigam H. Shah (Stanford University, USA)

Future of the Open Web Chairs: Stephen LaPorte (Wikimedia Foundation, USA)
Robert West (EPFL, Switzerland)

Designing an Ethical Web Chairs: Sharad Goel (Stanford University, USA)
Deirdre Mulligan (UC Berkeley, USA)

Sponsorship Chairs: Nuria Oliver (Vodafone Research and DataPop Alliance, Spain)
Jie Tang (Tsinghua University, China)
Kuansan Wang (Microsoft, USA)

Student Relations Chair: Ben Steichen (Cal Poly Pomona, USA)

Student Scholarship Chairs: Lora Aroyo (VU Amsterdam, The Netherlands)
Diego Sáez-Trumper (Wikimedia Foundation, Spain)

Communication Chairs: Dawn Anderson (Move It Marketing, UK)
Brent Hecht (Northwestern University, USA)
Xin Fu (Facebook, USA)

Proceedings Chair: Julian McAuley (UC San Diego, USA)

Poster Chairs: Ashton Anderson (University of Toronto, Canada)
Joe DeBlasio (Google, USA)
Reza Zafarani (Syracuse University, USA)

Track Chairs: Tarek Abdelzaher (University of Illinois Urbana-Champaign, USA)
Karl Aberer (École Polytechnique Fédérale de Lausanne, Switzerland)
Omar Alonso (Microsoft, USA)
Elisa Bertino (Purdue University, USA)
Paolo Boldi (University of Milan, Italy)
Carlos Castillo (Pompeu Fabra University, Spain)
Munmun De Choudhury (Georgia Institute of Technology, USA)
Ingemar Cox (University College London, UK)
Nick Craswell (Microsoft, USA)
Debora Donato (Mix, USA)
Aristides Gionis (Aalto University, Finland)
Arun Iyengar (IBM, USA)
Emre Kiciman (Microsoft, USA)
Ashish Kundu (IBM Research, USA)
Edith Law (University of Waterloo, Canada)
Hang Li (Bytedance AI Lab, Hong Kong)
Tie-Yan Liu (Microsoft, China)
Bing Liu (University of Illinois Chicago, USA)
Yoelle Marek (Amazon Research, Israel)
Peter Mika (Independent Consultant, Spain)
Alice Oh (Korea Advanced Institute of Science and Technology, South Korea)
Michael Rabinovich (Case Western Reserve University, USA)
Filip Radlinski (Google, UK)
Pavel Serdyukov (Yandex, Russia)
Sergei Vassilvitskii (Google, USA)
Nalini Venkatasubramanian (University of California Irvine, USA)
Cong Yu (Google, USA)



THE WEB CONFERENCE

Senior Program Committee

Eugene Agichtein (Emory University, USA)
Luca Maria Aiello (Nokia Bell Labs, UK)
Edoardo M. Airoldi (Harvard University, USA)
Omar Alonso (Microsoft, USA)
Andras A. Benczur (Hungarian Academy of Sciences, Hungary)
Shlomo Berkovsky (Commonwealth Scientific and Industrial Research Organisation, Australia)
Francesco Bonchi (ISI Foundation, Italy)
Peter Brusilovsky (University of Pittsburgh, USA)
Ceren Budak (University of Michigan, USA)
Jamie Callan (Carnegie Mellon University, USA)
Pablo Castells (Autonomous University of Madrid, Spain)
Yi Chang (Huawei Research America, USA)
Sanjay Chawla (Qatar Computing Research Institute, Qatar)
Ian Davidson (University of California, Davis, USA)
Brian Davison (Lehigh University, USA)
Nemanja Djuric (Uber, USA)
Alexey Drutsa (Yandex, Russia)
Nadia Fawaz (LinkedIn, USA)
Pierre Fraigniaud (CNRS / University Paris Diderot, France)
Norbert Fuhr (University of Duisburg-Essen, Germany)
David Gleich (Purdue University, USA)
Jennifer Golbeck (University of Maryland, USA)
Mihajlo Grbovic (Airbnb, USA)
Krishna Gummadi (Max Planck Institute for Software Systems, Germany)
Jiafeng Guo (Chinese Academy of Sciences, China)
Ido Guy (eBay, Israel)
Jiawei Han (University of Illinois Urbana-Champaign, USA)
Jake Hofman (Microsoft, USA)
Dietmar Jannach (Alpen-Adria-Universität Klagenfurt, Austria)
Jussi Karlgren (Gavagai and KTH Royal Institute of Technology, Sweden)
George Karypis (University of Minnesota, USA)
Oren Kurland (Technion, Israel)
Mounia Lalmas (Spotify, UK)
Silvio Lattanzi (Google, Switzerland)
Kristina Lerman (University of Southern California, USA)
Liane Lewin-Eytan (Amazon Research, Israel)
Yiqun Liu (Tsinghua University, China)
Qiaozhu Mei (University of Michigan, USA)
Vanessa Murdock (Amazon Research, USA)
Marc Najork (Google, USA)
Wolfgang Nejdl (L3S Research Center and University of Hannover, Germany)
Nuria Oliver (Vodafone Research, Spain)
Alessandro Panconesi (Sapienza University of Rome, Italy)
Dan Pelleg (Yahoo, Israel)
Kira Radinsky (eBay, Israel)
Vladan Radosavljevic (OLX, USA)
Yves Raimond (Netflix, UK)
Steffen Rendle (Google, USA)
Maarten de Rijke (University of Amsterdam, Netherlands)
Alessandra Sala (Bell Labs Ireland, Ireland)
C. Seshadhri (University of California Santa Cruz, USA)
Chirag Shah (Rutgers University, USA)
Milad Shokouhi (Microsoft, UK)
Fabrizio Silvestri (Facebook, UK)
Ian Soboroff (National Institute of Standards and Technology, USA)
Yizhou Sun (University of California Los Angeles, USA)
Evimaria Terzi (Boston University, USA)
Hanghang Tong (Arizona State University, USA)
Panayiotis Tsaparas (University of Ioannina, Greece)
Charalampos Tsourakakis (Harvard University, USA)
Sebastiano Vigna (University of Milan, Italy)
Ji-Rong Wen (Renmin University of China, China)
Jun Xu (Renmin University of China, China)
Elad Yom-Tov (Microsoft, Israel)
Ben Zhao (University of Chicago, USA)

Program Committee

E Philip Abdelmalik (World Health Organization, Switzerland)
Rediet Abebe (Cornell University, USA)
Sami Abu-El-Haija (Google, USA)
Panagiotis Adamopoulos (Emory University, USA)
Eytan Adar (University of Michigan, USA)
Nitish Aggarwal (IBM, USA)
Amr Ahmed (Google, USA)
Nesreen Ahmed (Intel, USA)
Jae-Wook Ahn (University of Pittsburgh, USA)
Qingyao Ai (University of Massachusetts Amherst, USA)
Luca Maria Aiello (Nokia Bell Labs, UK)
Deepak Ajwani (Nokia Bell Labs, Ireland)
Harith Alani (Open University, UK)
Panos Alexopoulos (Textkernel, Netherlands)
Mohammad Allahbakhsh (University of New South Wales, Australia)
Mark Allman (International Computer Science Institute, USA)
Jussara Almeida (Federal University of Minas Gerais, Brazil)
Virgilio Almeida (Federal University of Minas Gerais, Brazil)
Magnus Almgren (Chalmers University of Technology, Sweden)
Kristen Altenburger (Stanford University, USA)
Ismail Sengor Altingovde (Middle East Technical University, Turkey)
Jisun An (Qatar Computing Research Institute and Hamad Bin Khalifa University, Qatar)
Aris Anagnostopoulos (Sapienza University of Rome, Italy)
Avishek Anand (L3S Research Center, Germany)
Ashton Anderson (University of Toronto, Canada)
Renzo Angles (University of Talca, Chile)
Kemafor Anyanwu (North Carolina State University, USA)
Pablo Aragón (Pompeu Fabra University, Spain)
Carlos Buil Aranda (Federico Santa María Technical University, Chile)
Jaime Arguello (University of North Carolina Chapel Hill, USA)
Martin Arlitt (University of Calgary, Canada)
Lora Aroyo (Google, USA)
Ashwin Ashok (Georgia State University, USA)
Cigdem Aslay (Aalto University, Finland)
N. Asokan (Aalto University, Finland)
Manuel Atencia (University of Grenoble Alpes and INRIA, France)
Giuseppe Ateniese (Stevens Institute of Technology, USA)
Medha Atre (University of Oxford, UK)
Sören Auer (TIB Leibniz Information Center Science and Technology and University of Hannover, Germany)
Nathalie Aussenac-Gilles (CNRS / IRIT, France)
Ahmed Hassan Awadallah (Microsoft Research, USA)
Matthew Backus (Columbia University, USA)
Abraham Bagherjeiran (A9 / Amazon, USA)

Xiao Bai (Yahoo Research, USA)
Jinyeong Bak (Korea Advanced Institute of Science and Technology, South Korea)
Saeideh Bakhshi (Georgia Institute of Technology, USA)
Eric Balkanski (Harvard University, USA)
Krisztian Balog (University of Stavanger, Norway)
Zhifeng Bao (RMIT University, Australia)
Pablo Barberá (London School of Economics, UK)
Payam Barnaghi (University of Surrey, UK)
Daniel Barowy (Williams College, USA)
Valerio Basile (University of Turin, Italy)
Justin Basilico (Netflix, USA)
Kinjal Basu (LinkedIn, USA)
Christian Bauckhage (Fraunhofer, Germany)
Luca Becchetti (Sapienza University of Rome, Italy)
Srikanta Bedathur (Indian Institute of Technology Delhi, India)
Zohra Bellahsene (LIRMM, France)
Paolo Bellavista (University of Bologna, Italy)
Alejandro Bellogin (Autonomous University of Madrid, Spain)
Boualem Benatallah (University of New South Wales, Australia)
Michael Bendersky (Google, USA)

Fabricio Benevenuto (Federal University of Minas Gerais, Brazil)
Austin Benson (Cornell University, USA)
Klaus Berberich (Max Planck Institute for Informatics, Germany)
Michele Berlingerio (IBM, Ireland)
Maria Bermudez-Edo (University of Granada, UK)
George Berry (Facebook, USA)
Azer Bestavros (Boston University, USA)
Alex Beutel (Google, USA)
Anastasia Bezzubtseva (Yandex, Russia)
Tijl De Bie (Ghent University, Belgium)
Maria Bielikova (Slovak University of Technology, Slovakia)
Chris Biemann (University of Hamburg, Germany)
Battista Biggio (University of Cagliari, Italy)
Lidong Bing (Tencent, China)
Christian Bizer (University of Mannheim, Germany)
Jeremy Blackburn (University of Alabama Birmingham, USA)
Liad Blumrosen (Hebrew University, Israel)
Fernando Bobillo (University of Zaragoza, Spain)
Uldis Bojars (University of Latvia, Latvia)
Kalina Bontcheva (University of Sheffield, UK)
Ludovico Boratto (Eurecat, Spain)
Cecile Bothorel (Telecom Bretagne, France)
Alessandro Bozzon (Delft University of Technology, Netherlands)
Marco Brambilla (Polytechnic University of Milan, Italy)
Simina Branzei (Purdue University, USA)
Adrian M.P. Brasoveanu (MODUL Technology, Austria)
John Breslin (National University of Ireland Galway, Ireland)
David Buckeridge (McGill University, Canada)
Roman Budylin (Yandex, Russia)
Cody Buntain (University of Maryland, USA)
Keith Burghardt (University of Southern California, USA)
Elena Cabrio (University of Côte d'Azur and INRIA, France)
Jean-Paul Calbimonte (University of Applied Sciences and Arts Western Switzerland HES-SO, Switzerland)
Chico Camargo (University of Oxford, UK)
B. Barla Cambazoglu (RMIT University, Australia)
Jan Camenisch (Dfinity, Switzerland)
Ran Canetti (Boston University, USA; Tel Aviv University, Israel)
Iván Cantador (Autonomous University of Madrid, Spain)
Jiannong Cao (Hong Kong Polytechnic University, Hong Kong)
Cinzia Cappiello (Polytechnic University of Milan, Italy)
Justin Cappos (New York University, USA)
Bogdan Carbunar (Motorola Labs, USA)
Valeria Cardellini (University of Roma Tor Vergata, Italy)
Barbara Carminati (University of Insubria, Italy)
Leslie Carr (University of Southampton, UK)
David Carral (Technical University of Dresden, Germany)
Aniello Castiglione (University of Salerno, Italy)
Dario Catalano (University of Catania, Italy)
Lorenzo Cavallaro (University of London, UK)
James Caverlee (Texas A&M University, USA)
Davide Ceolin (Vrije Universiteit Amsterdam, Netherlands)
Sofia Ceppi (University of Edinburgh, UK)
Suleyman Cetintas (Yahoo Research, USA)
Meeyoung Cha (Korea Advanced Institute of Science and Technology, South Korea)
Augustin Chaintreau (Columbia University, USA)
Kaushik Chakrabarti (Microsoft, USA)
Tanmoy Chakraborty (Indraprastha Institute of Information Technology Delhi, India)
T-H. Hubert Chan (University of Hong Kong, Hong Kong)
Stevie Chancellor (Georgia Institute of Technology, USA)
Ashok Chandrashekar (Netflix, USA)
Kevin Chang (University of Illinois Urbana-Champaign, USA)
Thierry Charnois (LIPN CNRS University of PARIS 13, France)
Keke Chen (Wright State University, USA)
Lei Chen (Hong Kong University of Science and Technology, Hong Kong)
Li Chen (Hong Kong Baptist University, Hong Kong)
Pin-Yu Chen (IBM, USA)

Songqing Chen (George Mason University, USA)
Wei Chen (Microsoft, China)
Xiaojun Chen (Shenzhen University, China)
Zhiyuan Chen (Google, USA)
Zhumin Chen (Shandong University, China)
Gong Cheng (Nanjing University, China)
Ed H. Chi (Google, USA)
Christian Chiarcos (Goethe University, Germany)
Flavio Chierichetti (Sapienza University of Rome, Italy)
Jaegul Choo (Korea University, South Korea)
Richard Chow (Intel, USA)
Sherman S. M. Chow (Chinese University of Hong Kong, Hong Kong)
Omar Chowdhury (University of Iowa, USA)
Konstantina Christakopoulou (University of Minnesota, USA)
Aleksandr Chuklin (Google, Switzerland)
Rumi Chunara (New York University, USA)
Charles Clarke (Facebook, Canada)
Francisco Claude (eBay, USA)
Michael Cochez (Fraunhofer, Germany)
Daniel Cohen (University of Massachusetts Amherst, USA)
Michele Colajanni (University of Modena, Italy)
Michael Cole (LexisNexis, USA)
Pieter Colpaert (Ghent University, Belgium)
Simona Colucci (Polytechnic University of Bari, Italy)
Mauro Conti (University of Padua, Italy)
Olivier Corby (INRIA, France)
Oscar Corcho (Polytechnic University of Madrid, Spain)
Luca Costabello (Accenture Labs, Ireland)
Fabio Cozman (University of São Paulo, Brazil)
Henriette Cramer (Spotify, USA)
Lorrie Cranor (Carnegie Mellon University, USA)
Paolo Cremonesi (Polytechnic University of Milan, Italy)
Pierluigi Crescenzi (University of Florence, Italy)
Bruno Crispo (University of Trento, Italy)
Bruce Croft (University of Massachusetts Amherst, USA)
Mark Crovella (Boston University, USA)
Isabel Cruz (University of Illinois Chicago, USA)
Philippe Cudre-Mauroux (University of Fribourg, Switzerland)
Peng Cui (Tsinghua University, China)
Shane Culpepper (RMIT University, Australia)
Olivier Curé (University of Paris-Est, France)
Enrico Daga (Open University, UK)
Andrew Dai (Google, USA)
Zhuyun Dai (Carnegie Mellon University, USA)
Elizabeth M. Daly (IBM, Ireland)
Claudia d'Amato (University of Bari, Italy)
Florian Daniel (Polytechnic University of Milan, Italy)
Maximilien Danisch (Laboratoire d'Informatique de Paris 6, France)
Mathieu D'Aquin (National University of Ireland Galway, Ireland)
Anirban Dasgupta (Indian Institute of Technology Gandhinagar, India)
Lucas Davi (University of Duisburg-Essen, Germany)
Jérôme David (INRIA, France)
Thierry Declerck (DFKI, Germany)
Jaime Delgado (Polytechnic University of Catalonia, Spain)
Daniele Dell'Aglio (University of Zurich, Switzerland)
Gianluca Demartini (University of Queensland, Australia)
Elena Demidova (L3S Research Center, Germany)
Ronald Denaux (Expert System, Spain)
Hongbo Deng (Google, USA)
Sebastian Deri (Cornell University, USA)
Antoine Desir (INSEAD, France)
Nikhil Devanur (Microsoft, USA)
Dennis Diefenbach (University Jean Monet, France)
Jana Diesner (University of Illinois Urbana-Champaign, USA)
Stefan Dietze (GESIS - Leibniz Institute for the Social Sciences, Germany)
Djellel Difallah (New York University, USA)
Pavel Dmitriev (Outreach, USA)
Juan Manuel Dodero (University of Cadiz, Spain)
Carlotta Domeniconi (George Mason University, USA)
Yuxiao Dong (Microsoft, USA)
Zhicheng Dou (Renmin University of China, China)
Adam Doupé (Arizona State University, USA)
Mauro Dragoni (Fondazione Bruno Kessler, Italy)
Eduard Dragut (Temple University, USA)
Nan Du (Google, USA)
Nick Duffield (Texas A&M University, USA)
Casey Dugan (IBM Research, USA)

Adam Dunn (Macquarie University, Australia)
Schahram Dustdar (Vienna University of Technology, Austria)
Koji Eguchi (Kobe University, Japan)
Maud Ehrmann (École Polytechnique Fédérale de Lausanne, Switzerland)
Carsten Eickhoff (Brown University, USA)
Michael Ekstrand (Boise State University, USA)
Mehdi Elahi (Free University of Bozen-Bolzano, Italy)
Edith Elkind (University of Oxford, UK)
William Enck (North Carolina State University, USA)
Alina Ene (Boston University, USA)
Alessandro Epasto (Google, USA)
Vadim Ermolayev (Zaporizhzhia National University, Ukraine)
Hossein Esfandiari (Harvard University, USA)
Jérôme Euzenat (INRIA and University of Grenoble Alpes, France)
Alex Fabrikant (Google, USA)
Shobeir Fakhraei (University of Southern California, USA)
Hui Fang (University of Delaware, USA)
Nicola Fanizzi (University of Bari, Italy)
Valentina Fedorova (Yandex, Russia)
Alexander Felfernig (Graz University of Technology, Austria)
Anna Fensel (University of Innsbruck, Austria)
Miriam Fernandez (Knowledge Media Institute, UK)
Javier D. Fernández (Vienna University of Economics and Business, Austria)
Emilio Ferrara (University of Southern California, USA)
Elena Ferrari (University of Insubria, Italy)
Sebastien Ferre (University of Rennes 1, France)
Nicola Ferro (University of Padova, Italy)
Besnik Fetahu (L3S Research Center, Germany)
Aris Filos-Ratsikas (École Polytechnique Fédérale de Lausanne, Switzerland)
Tim Finin (University of Maryland Baltimore County, USA)
Valeria Fionda (University of Calabria, Italy)
Fabian Flöck (GESIS - Leibniz Institute for the Social Sciences, Germany)
Aurélien Francillon (Eurecom, France)
Flavius Frasincar (Erasmus University Rotterdam, Netherlands)
Irini Fundulaki (FORTH, Greece)
Adam Funk (University of Sheffield, UK)
Ujwal Gadiraju (L3S Research Center, Germany)
Sabrina Gaito (University of Milan, Italy)
Krzysztof Gajos (Harvard University, USA)
Esther Galbrun (University of Eastern Finland, Finland)
Aram Galstyan (University of Southern California, USA)
Anshul Gandhi (Stony Brook University, USA)
Niloy Ganguly (Indian Institute of Technology Kharagpur, India)
Raghu Ganti (IBM, USA)
Sam Ganzfried (Ganzfried Research, USA)
Hongchang Gao (University of Texas Arlington, USA)
Lixin Gao (University of Massachusetts Amherst, USA)
David Garcia (Medical University of Vienna and Complexity Science Hub, Austria)
Simson Garfinkel (US Census Bureau, USA)
Daniel Garijo (Information Sciences Institute, USA)
Kiran Garimella (Aalto University, Finland)
Wolfgang Gatterbauer (Northeastern University, USA)
Assefaw Gebremedhin (Washington State University, USA)
Marco De Gemmis (University of Bari, Italy)
Anna Lisa Gentile (IBM, USA)
Nikolaos Georgantas (INRIA, France)
Werner Geyer (IBM Research, USA)
Ran Gilad-Bachrach (Microsoft, Israel)
C. Lee Giles (Pennsylvania State University, USA)
Benjamin Girault (University of Southern California, USA)
Vasilis Gkatzelis (Drexel University, USA)
Nadav Golbandi (Google, USA)
Jennifer Golbeck (University of Maryland, USA)
Dan Goldstein (Microsoft, USA)
Dieter Gollmann (Hamburg University of Technology, Germany)
Negin Golrezaei (Massachusetts Institute of Technology, USA)
Jose Manuel Gomez-Perez (Expert System, Spain)
Rafael S Gonçalves (Stanford University, USA)
Graciela Gonzalez (University of Pennsylvania, USA)
Sandra Gonzalez-Bailon (University of Pennsylvania, USA)

Roberto González-Ibáñez (University of Santiago, Chile)
Mitchell Gordon (Stanford University, USA)
Przemyslaw Grabowicz (Max Planck Institute for Software Systems, Germany)
Jorge Gracia (University of Zaragoza, Spain)
Luis Gravano (Columbia University and Google, USA)
Gregory Grefenstette (IHMC and Biggerpan, France)
Marko Grobelnik (Jozef Stefan Institute, Slovenia)
Roberto Grossi (University of Pisa, Italy)
Paul Groth (University of Amsterdam, Netherlands)
Quanquan Gu (University of California Los Angeles, USA)
Christophe Guéret (Accenture Labs, Ireland)
Francesco Gullo (UniCredit, Italy)
Kalpa Gunaratna (Samsung Electronics, USA)
Stephan Günnemann (Technical University of Munich, Germany)
Gleb Gusev (Yandex, Russia)
Claudio Gutierrez (University of Chile, Chile)
Gahgene Gweon (Seoul National University, South Korea)
Peter Haase (metaphacts, Germany)
Hamed Haddadi (Imperial College London, UK)
Dilek Hakkani-Tur (Google, USA)
Scott Hale (University of Oxford, UK)
Armin Haller (Australian National University, Australia)
Harry Halpin (World Wide Web Consortium, UK)
William Hamilton (Stanford University, USA)
Shuguang Han Han (Google, USA)
Weili Han (Fudan University, China)
Wook-Shin Han (Pohang University of Science and Technology, South Korea)
Allan Hanbury (Vienna University of Technology, Austria)
Sanda Harabagiu (University of Texas Dallas, USA)
Elad Haramaty (Amazon, Israel)
Andreas Harth (University of Erlangen-Nuremberg, Germany)
Olaf Hartig (Linköping University, Sweden)
Mohammad Hasan (Indiana University and Purdue University, USA)
Ben He (Chinese Academy of Sciences, China)
Jingrui He (Arizona State University, USA)
Jiyin He (Signal, UK)
Xiangnan He (National University of Singapore, Singapore)
Yuan He (Tsinghua University, China)
Pascal Van Hentenryck (Georgia Institute of Technology, USA)
Chien-Ju Ho (Washington University in St. Louis, USA)
Joyce Ho (Emory University, USA)
Aidan Hogan (University of Chile, Chile)
Liangjie Hong (Etsy, USA)
Lichan Hong (Google, USA)
Frank Hopfgartner (University of Sheffield, UK)
Endadul Hoque (Florida International University, USA)
Katja Hose (Aalborg University, Denmark)
Andreas Hotho (University of Wuerzburg, Germany)
Geert-Jan Houben (Delft University of Technology, Netherlands)
Hsun-Ping Hsieh (National Cheng Kung University, Taiwan)
Cheng-Hsin Hsu (National Tsing Hua University, Taiwan)
Jane Hsu (National Taiwan University, Taiwan)
Hongxin Hu (Clemson University, USA)
Shaohan Hu (IBM, USA)
Wei Hu (Nanjing University, China)
Xia Ben Hu (Texas A&M University, USA)
Yuheng Hu (University of Illinois Chicago, USA)
Minlie Huang (Tsinghua University, China)
Polly Huang (Nanyang Technological University, Singapore)
Ting-Hao Huang (Carnegie Mellon University, USA)
Xin Huang (Hong Kong Baptist University, Hong Kong)
Xinyi Huang (Fujian Normal University, China)
Jina Huh (Michigan State University, USA)
Ioana Hulpus (University of Mannheim, Germany)
Patrick Hummel (Google, USA)
Neil Hurley (University College Dublin, Ireland)
Seung-Won Hwang (Yonsei University, South Korea)
Adriana Iamnitchi (University of South Florida, USA)
Yazmin A. Ibanez-Garcia (Vienna University of Technology, Austria)
Oana Inel (Vrije Universiteit Amsterdam, Netherlands)
Amir Ingber (Amazon Research, Israel)
Sotiris Ioannidis (FORTH, Greece)

Panos Ipeirotis (New York University, USA)
Yoshiharu Ishikawa (Nagoya University, Japan)
Valerie Issarny (INRIA, France)
Krishnamurthy Iyer (Cornell University, USA)
Abbie Jacobs (University of California Berkeley, USA)
Alejandro Jaimes (Dataminr, USA)
Shweta Jain (Indian Institute of Science Bangalore, India)
Hyeju Jang (University of British Columbia, Canada)
Krzysztof Janowicz (University of California Santa Barbara, USA)
Robert Jäschke (Humboldt University of Berlin, Germany)
Di Jiang (Hong Kong University of Science and Technology, Hong Kong)
Hongbo Jiang (Huazhong University of Science and Technology, China)
Meng Jiang (University of Notre Dame, USA)
Wei Jiang (University of Missouri-Columbia, USA)
Xin Jiang (Huawei Noah's Ark Lab, Hong Kong)
Ernesto Jimenez-Ruiz (Alan Turing Institute, UK)
Gareth Jones (Dublin City University, Ireland)
Clement Jonquet (University of Montpellier, France)
Joemon Jose (University of Glasgow, UK)
Wagner Meira Jr. (Federal University of Minas Gerais, Brazil)
Michael Jugovac (Technical University of Dortmund, Germany)
Christine Julien (University of Texas Austin, USA)
David Jurgens (University of Michigan, USA)
Panos Kalnis (King Abdullah University of Science and Technology, Saudi Arabia)
Vana Kalogeraki (Athens University of Economics and Business, Greece)
Andreas Kaltenbrunner (NTENT, Spain)
Benedikt Kämpgen (Empolis Information Management, Germany)
Jaap Kamps (University of Amsterdam, Netherlands)
Bhargav Kanagal (Google, USA)
U Kang (Seoul National University, South Korea)
Evangelos Kanoulas (University of Amsterdam, Netherlands)
Murat Kantarcioglu (University of Texas at Dallas, USA)
Thomas Karagiannis (Microsoft, UK)
Alexandros Karatzoglou (Telefonica Research, Spain)
Maryam Karimzadehgan (Google, USA)
Panagiotis Karras (Aarhus University, Denmark)
Ian Kash (University of Illinois Chicago, USA)
Aniket Kate (Purdue University, USA)
Makoto P. Kato (Kyoto University, Japan)
Tomi Kauppinen (Aalto University, Finland)
Takahiro Kawamura (Japan Science and Technology Agency, Japan)
Mark Keane (University College Dublin, Ireland)
Brian Keegan (University of Colorado Boulder, USA)
Mayank Kejriwal (Information Sciences Institute, USA)
Jeffrey Kephart (IBM, USA)
Florian Kerschbaum (University of Waterloo, Canada)
Carsten Keßler (Aalborg University, Denmark)
Arijit Khan (Nanyang Technological University, Singapore)
Megha Khosla (L3S Research Center, Leibniz University, Germany)
Joon Hee Kim (Amazon, USA)
Jooyeon Kim (Korea Advanced Institute of Science and Technology, South Korea)
Younghoon Kim (Hanyang University, South Korea)
Masahiro Kimura (Ryukoku University, Japan)
Sabrina Kirrane (Vienna University of Economics and Business, Austria)
Sergey Kirshner (Facebook, USA)
Julia Kiseleva (Microsoft Research AI, USA)
Martin Klein (Los Alamos National Laboratory, USA)
Daniel Kluver (University of Minnesota, USA)
Bart Knijnenburg (Clemson University, USA)
Matthias Knorr (New University of Lisbon, Portugal)
Marina Kogan (University of New Mexico, USA)
Olessia Koltsova (National Research University Higher School of Economics, Russia)
Deguang Kong (Yahoo Research, USA)
Weize Kong (Google, USA)
Xiangnan Kong (Worcester Polytechnic Institute, USA)
David Konopnicki (IBM, Israel)
Roberto Konow (eBay, USA)
Stasinos Konstantopoulos (NCSR Demokritos, Greece)
Roman Kontchakov (Birkbeck University of London, UK)
Jacek Kopecky (University of Portsmouth, UK)
Vassilis Kostakos (University of Melbourne, Australia)

Patty Kostkova (University College London, UK)
Egor V. Kostylev (University of Oxford, UK)
Spyros Kotoulas (Facebook, UK)
Alexander Kotov (Wayne State University, USA)
Manolis Koubarakis (National and Kapodistrian University Athens, Greece)
Ioannis Koutis (New Jersey Institute of Technology, USA)
Danai Koutra (University of Michigan, USA)
Zornitsa Kozareva (Google, USA)
Elad Kravi (Amazon Research, Israel)
Sara Krehbiel (University of Richmond, USA)
Udo Kruschwitz (University of Essex, UK)
Alan Kuhnle (University of Florida, USA)
Anagha Kulkarni (San Francisco State University, USA)
Anand Kulkarni (University of California Berkeley, USA)
Srijan Kumar (Stanford University, USA)
Ugur Kursuncu (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Aleksandar Kusmanovic (Northwestern University, USA)
Haewoon Kwak (Qatar Computing Research Institute, Qatar)
Maria Kyropoulou (University of Essex, UK)
Susana Ladra (University of A Coruña, Spain)
Sebastien Lahaie (Microsoft, USA)
Wai Lam (Chinese University of Hong Kong, Hong Kong)
Steffen Lamparter (Siemens, Germany)
Vasileios Lampos (University College London, UK)
Yanyan Lan (Chinese Academy of Sciences, China)
Nic Lane (University College London, UK)
Birger Larsen (Aalborg University, Denmark)
Kate Larson (University of Waterloo, Canada)
Matthieu Latapy (Laboratoire d'Informatique de Paris 6, France)
Hady Lauw (Singapore Management University, Singapore)
Agnieszka Lawrynowicz (Poznan University of Technology, Poland)
Gregor Leban (Jozef Stefan Institute, Slovenia)
Adam J. Lee (University of Pittsburgh, USA)
Chia-Jung Lee (Microsoft, USA)
Dongwon Lee (Pennsylvania State University, USA)
Kisung Lee (Louisiana State University, USA)
Kyumin Lee (Worcester Polytechnic Institute, USA)
Matthijs van Leeuwen (Leiden University, Netherlands)
Stefano Leonardi (Sapienza University of Rome, Italy)
Ben Leong (Educational Testing Service, USA)
Cheng Li (Google, USA)
Fangtao Li (Google, USA)
Juanzi Li (Tsinghua University, China)
Jundong Li (Arizona State University, USA)
Liangyue Li (Arizona State University, USA)
Ninghui Li (Purdue University, USA)
Shuai Li (University of Cambridge, UK)
Zhenhui Jessie Li (Pennsylvania State University, USA)
Zhenkai Liang (National University of Singapore, Singapore)
David Liben-Nowell (Carleton College, USA)
Alexander Libov (Amazon, Israel)
Jefrey Lijffijt (Ghent University, Belgium)
Ee-Peng Lim (Singapore Management University, Singapore)
Chenghua Lin (University of Aberdeen, UK)
Shou-De Lin (National Taiwan University, Taiwan)
Yu-Ru Lin (University of Pittsburgh, USA)
Christina Lioma (University of Copenhagen, Denmark)
Jingjing Liu (University of South Carolina, USA)
Kang Liu (Chinese Academy of Sciences, China)
Mingrong Liu (Sogou, China)
Weiwei Liu (University of Technology Sydney, Australia)
Xuanzhe Liu (Peking University, China)
Jorge Lobo (ICREA and Pompeu Fabra University, Spain)
Margaret Loper (Georgia Institute of Technology, USA)
Nuno Lopes (TopQuadrant, Ireland)
Vanessa Lopez (IBM, Ireland)
Pasquale Lops (University of Bari, Italy)
Alexander Löser (Beuth University of Applied Sciences Berlin, Germany)
Jiannan Lu (Microsoft, USA)
Claudio Lucchese (Ca' Foscari University of Venice, Italy)

Markus Luczak-Roesch (Victoria University of Wellington, New Zealand)
Bernd Ludwig (University Regensburg, Germany)
Thomas Lukasiewicz (University of Oxford, UK)
Vincent Lully (Paris-Sorbonne University, France)
Cheng Luo (Tsinghua University, China)
Luis Fernandez Luque (Qatar Computing Research Institute, Qatar)
Yuanhua Lv (Microsoft, USA)
Thodoris Lykouris (Cornell University, USA)
Ioanna Lytra (University of Bonn, Germany)
Hao Ma (Facebook, USA)
Qiang Ma (Google, USA)
Weidong Ma (Microsoft, China)
Andrew Macfarlane (City University London, UK)
Bruce Maggs (Duke University, USA)
Mohammad Mahdian (Google, USA)
Jalal Mahmud (IBM, USA)
Frederick Maier (University of Georgia, USA)
Azarakhsh Malekian (University of Toronto, Canada)
Momin Malik (Carnegie Mellon University, USA)
Fragkiskos Malliaros (University of Paris-Saclay, France)
Eric Malmi (Google, Switzerland)
Luigi Mancini (Sapienza University of Rome, Italy)
Lydia Manikonda (Arizona State University, USA)
Andrea Mannocci (Open University, UK)
Andrew Mao (Microsoft, USA)
Jiaxin Mao (Tsinghua University, China)
Andrea Marino (University of Pisa, Italy)
Evangelos Markatos (FORTH, Greece)
Ilya Markov (University of Amsterdam, Netherlands)
Trevor Martin (University of Bristol, UK)
Miguel A. Martinez-Prieto (University of Valladolid, Spain)
Cecilia Mascolo (University of Cambridge, UK)
Philip Massey (Drexel University, USA)
Andrea Maurino (University of Milano-Bicocca, Italy)
Wolfgang May (University of Göttingen, Germany)
Randolph McAfee (Microsoft, USA)
Julian McAuley (University of California San Diego, USA)
Julie McCann (Imperial College London, UK)
John P. McCrae (National University of Ireland Galway, Ireland)
Andres Munoz Medina (Google, USA)
Lionel Médini (University of Lyon, France)
Rishabh Mehrotra (Spotify, USA)
Edgar Meij (Bloomberg, UK)
Yelena Mejova (Qatar Computing Research Institute, Qatar)
Massimo Melucci (University of Padova, Italy)
Daniele Midi (Google, USA)
Alessandra Mileo (Dublin City University, Ireland)
Vahab Mirrokni (Google, USA)
Archan Misra (Singapore Management University, Singapore)
Satyajayant Misra (New Mexico State University, USA)
Lewis Mitchell (University of Adelaide, Australia)
Bhaskar Mitra (Microsoft, Canada)
Stefano Mizzaro (University of Udine, Italy)
Marie-Francine Moens (Katholieke Universiteit Leuven, Belgium)
Alistair Moffat (University of Melbourne, Australia)
Boughanem Mohand (IRIT / University Paul Sabatier Toulouse, France)
Pascal Molli (University of Nantes, France)
Misael Mongiovì (Institute of Cognitive Sciences and Technologies, Italy)
Sue Moon (Korea Advanced Institute of Science and Technology, South Korea)
Gianmarco De Francisci Morales (ISI Foundation, Italy)
Sebastian Moreno (Adolfo Ibáñez University, Chile)
Charles Morisset (Newcastle University, UK)
Fred Morstatter (University of Southern California, USA)
Alessandro Moschitti (Qatar Computing Research Institute, Qatar)
Yashar Moshfeghi (University of Strathclyde, UK)
Josiane Mothe (IRIT, France)
Davide Mottin (Aarhus University, Denmark)
Luca Mottola (Polytechnic University of Milan, Italy; SICS Swedish ICT, Sweden)
Isabelle Moulinier (Capital One, USA)
Henning Müller (University of Applied Sciences and Arts Western Switzerland HES-SO, Switzerland)
Kamesh Munagala (Duke University, USA)
Kevin Munger (New York University, USA)
Mariano Rodríguez Muro (Google, USA)

Raghava Mutharaju (Indraprastha Institute of Information Technology Delhi, India)
Erich Nahum (IBM, USA)
Franco Maria Nardini (Institute of Information Science and Technologies, Italy)
Wolfgang Nejdl (L3S Research Center and University of Hannover, Germany)
Laura Nelson (Northeastern University, USA)
Axel-Cyrille Ngonga Ngomo (Paderborn University, Germany)
Quoc Viet Hung Nguyen (Griffith University, Australia)
Jian-Yun Nie (University of Montreal, Canada)
Liqiang Nie (Shandong University, China)
Thomas Niebler (University of Wuerzburg, Germany)
Andriy Nikolov (metaphacts, Germany)
Elaine Nsoesie (University of Washington, USA)
Alexandros Ntoulas (LinkedIn, USA)
Neil O'Hare (Yahoo Research, USA)
Alexandra Olteanu (Microsoft, USA)
Constantin Orasan (University of Wolverhampton, UK)
Sigal Oren (Ben-Gurion University, Israel)
Peter Organisciak (University of Denver, USA)
Francesco Osborne (Open University, UK)
Eliyahu Osherovich (Amazon, Israel)
Jörg Ott (Technical University of Munich, Germany)
Federica Paci (University of Southampton, UK)
Martin Pal (Google, USA)
Balaji Palanisamy (University of Pittsburgh, USA)
George Pallis (University of Cyprus, Cyprus)
Raul Palma (Poznan Supercomputing and Networking Center, Poland)
Matteo Palmonari (University of Milano-Bicocca, Italy)
Robert Palovics (MTA SZTAKI, Hungary)
Shirui Pan (University of Technology Sydney, Australia)
Sinno Jialin Pan (Nanyang Technological University, Singapore)
Weike Pan (Shenzhen University, China)
Daniela Paolotti (ISI Foundation, Italy)
Evangelos Papalexakis (University of California Riverside, USA)
Stelios Paparizos (Google, USA)
Javier Parapar (University of A Coruña, Spain)
Paolo Parigi (Stanford University, USA)
Praveen Paritosh (Google, USA)
Sungjoon Park (Korea Advanced Institute of Science and Technology, South Korea)
Nikos Parotsidis (University of Rome Tor Vergata, Italy)
Denis Parra (Pontifical Catholic University of Chile, Chile)
Rahul Parundekar (Toyota Info-Technology Center, USA)
Andrea Passarella (Institute of Informatics and Telematics, Italy)
Bibek Paudel (University of Zurich, Switzerland)
Michael Paul (University of Colorado Boulder, USA)
Umashanthi Pavalanathan (Georgia Institute of Technology, USA)
Leto Peel (University Catholique of Louvain, Belgium)
Konstantinos Pelechrinis (University of Pittsburgh, USA)
Marco Pellegrini (Institute of Informatics and Telematics, Italy)
Tassilo Pellegrini (University of Applied Sciences St. Pölten, Austria)
David Pennock (Microsoft, USA)
Roberto Perdisci (University of Georgia, USA)
Sujan Perera (IBM Watson, USA)
Jorge Pérez (University of Chile, Chile)
Silvio Peroni (University of Bologna, Italy)
Bryan Perozzi (Google, USA)
Nicola Perra (University of Greenwich, UK)
Juergen Pfeffer (Technical University of Munich, Germany)
Danh Le Phuoc (Technical University of Berlin, Germany)
Reinhard Pichler (Vienna University of Technology, Austria)
Guillaume Pierre (IRISA / University of Rennes 1, France)
Emma Pierson (Stanford University, USA)
Emmanuel Pietriga (INRIA, France)
Roberto Di Pietro (Hamad Bin Khalifa University, Qatar)
Georgios Piliouras (Singapore University of Technology and Design, Singapore)
Ali Pinar (Sandia National Laboratories, USA)
H. Sofia Pinto (Instituto Superior Tecnico, Portugal)

Giuseppe Pirrò (Institute for High Performance Computing and Networking, Italy)
Evaggelia Pitoura (University of Ioannina, Greece)
Benjamin Piwowarski (CNRS / University Pierre et Marie Curie, France)
Luiz Pizzato (Commonwealth Bank of Australia, Australia)
Vassilis Plachouras (Facebook, UK)
Dimitris Plexousakis (FORTH, Greece)
Barbara Poblete (University of Chile, Chile)
Jean Pouget-Abadie (Harvard University, USA)
B. Aditya Prakash (Virginia Polytechnic Institute and State University, USA)
Livia Predoiu (University of Oxford, UK)
Julia Proskurnia (École Polytechnique Fédérale de Lausanne, Switzerland)
Cédric Pruski (Luxembourg Institute of Science and Technology, Luxembourg)
Calton Pu (Georgia Institute of Technology, USA)
Geppino Pucci (University of Padova, Italy)
Hemant Purohit (George Mason University, USA)
Qi Qi (Hong Kong University of Science and Technology, Hong Kong)
Tieyun Qian (Wuhan University, China)
Tao Qin (Microsoft, China)
Yuzhong Qu (Nanjing University, China)
Daniele Quercia (Yahoo Labs, Spain)
Alexander Quinn (Purdue University, USA)
Fiana Raiber (Yahoo Research, Israel)
Karthik Raman (Google, USA)
Maya Ramanath (Indian Institute of Technology Delhi, India)
Lida Rashidi (University of Melbourne, Australia)
Edie Rasmussen (University of British Columbia, Canada)
Hadas Raviv (Technion, Israel)
Indrakshi Ray (Colorado State University, USA)
Soumya Ray (Case Western Reserve University, USA)
Simon Razniewski (Max Planck Institute for Informatics, Germany)
Miriam Redi (Wikimedia Foundation, UK)
Georg Rehm (DFKI, Germany)
Manuel Reinert (SAP, Germany)
Reza Rejaie (University of Oregon, USA)
Xiang Ren (University of Southern California, USA)
Martin Rezk (Rakuten, Japan)
Bruno Ribeiro (Purdue University, USA)
Matteo Riondato (Two Sigma Investments, USA)
Petar Ristoski (iBM Research, USA)
Alan Ritter (Ohio State University, USA)
Carlos R. Rivero (Rochester Institute of Technology, USA)
Marian-Andrei Rizoiu (NICTA Research Laboratory, Australia)
Giuseppe Rizzo (Istituto Superiore Mario Boella, Italy)
Kirk Roberts (University of Texas Dallas, USA)
Manuel Gomez Rodriguez (Max Planck Institute for Software Systems, Germany)
Kay Roemer (Graz University of Technology, Austria)
Haggai Roitman (IBM Research Haifa, Israel)
Daniel Romero (University of Michigan, USA)
Marco Rospocher (University of Verona, Italy)
Christopher Rossbach (University of Texas Austin and VMware Research Group, USA)
Gustavo Rossi (National University of La Plata, Argentina)
Ryan Rossi (Adobe Research, USA)
Rahmtin Rotabi (Google and Cornell University, USA)
Camille Roth (CNRS, France)
Marie-Christine Rousset (University of Grenoble Alpes, France)
Polina Rozenshtein (Aalto University, Finland)
Natali Ruchansky (University of Southern California, USA)
Edna Ruckhaus (Polytechnic University of Madrid, Spain)
Giancarlo Ruffo (University of Turin, Italy)
Giovanni Russello (University of Auckland, New Zealand)
Andrey Rzhetsky (University of Chicago, USA)
Andrei Sabelfeld (Chalmers University of Technology, Sweden)
Owen Sacco (University of Malta, Malta)
Harald Sack (Leibniz Institute for Information Infrastructure and Karlsruhe Institute of Technology, Germany)
Ahmad-Reza Sadeghi (Technical University of Darmstadt, Germany)
Reyhaneh Safavi-Naini (University of Calgary, Canada)
Shaghayegh Sahebi (University at Albany - SUNY, USA)

Satya Sahoo (Case Western Reserve University, USA)
Alan Said (University of Skövde, Sweden)
Yasushi Sakurai (Kumamoto University, Japan)
Angelo Antonio Salatino (Open University, UK)
Muhammad Saleem (University of Leizpig, Germany)
Brendan Saltaformaggio (Georgia Institute of Technology, USA)
Iraj Saniee (Nokia, Bell Laboratories, USA)
Scott Sanner (University of Toronto, Canada)
Massimo Santini (University of Milan, Italy)
Rodrygo Santos (Federal University of Minas Gerais, Brazil)
A. Erdem Sarıyüce (University at Buffalo - SUNY, USA)
Tamas Sarlos (Google, USA)
Felix Sasaki (Lambdawerk, Germany)
Bahar Sateli (Concordia University, Canada)
Kai-Uwe Sattler (Technical University of Ilmenau, Germany)
Mahadev Satyanarayanan (Carnegie Mellon University, USA)
Vadim Savenkov (Vienna University of Economics and Business, Austria)
Marco Luca Sbodio (IBM, Ireland)
Samuel Scarpino (University of Vermont, USA)
Johann Schaible (GESIS - Leibniz Institute for the Social Sciences, Germany)
Markus Schedl (Johannes Kepler University, Austria)
Ansgar Scherp (Kiel University and ZBW - Leibniz Information Center for Economics, Germany)
Rossano Schifanella (University of Turin, Italy)
Stefan Schlobach (Vrije Universiteit Amsterdam, Netherlands)
Douglas Schmidt (Vanderbilt University, USA)
Lars Schmidt-Thieme (University of Hildesheim, Germany)
Grant Schoenebeck (University of Michigan, USA)
Uri Schonfeld (Yahoo, USA)
Okke Schrijvers (Stanford University, USA)
Lutz Schröder (Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany)
Anne Schuth (de Persgroep, Netherlands)
Daniel Schwabe (Pontifical Catholic University of Rio de Janeiro, Brazil)
Chris Schwiegelshohn (Technical University of Dortmund, Germany)
Kent Seamons (Brigham Young University, USA)
Satoshi Sekine (New York University, USA)
Erik Selberg (Amazon, USA)
Juan F. Sequeda (Capsenta Labs, USA)
Sandra Servia-Rodríguez (University of Cambridge, UK)
Huasong Shan (JD.com, USA)
Amit Sharma (Microsoft, USA)
Aneesh Sharma (Google, USA)
Saeedeh Shekarpour (University of Dayton, USA)
Huawei Shen (Chinese Academy of Sciences, China)
Xiaobo Shen (Nanjing University, China)
Prashant Shenoy (University of Massachusetts, USA)
Amit Sheth (Wright State University, USA)
Weisong Shi (Wayne State University, USA)
Kyuseok Shim (Seoul National University, South Korea)
Kijung Shin (Carnegie Mellon University, USA)
Alessio Signorini (Evidation Health, USA)
Arlei Silva (University of California Santa Barbara, USA)
Francesco Silvestri (University of Padova, Italy)
Gerardo Simari (National University of South and CONICET, Argentina)
Elena Simperl (University of Southampton, UK)
Ambuj Singh (University of California Santa Barbara, USA)
Munindar Singh (North Carolina State University, USA)
Moumita Sinha (Adobe Research, India)
Balasubramanian Sivan (Google, USA)
Hala Skaf-Molli (University of Nantes, France)
Steven Skiena (Stony Brook University, USA)
Kevin Small (Amazon, USA)
Luca Soldaini (Amazon, USA)
Dezhao Song (Thomson Reuters, USA)
Junehwa Song (Korea Advanced Institute of Science and Technology, South Korea)
Akshay Soni (Yahoo, USA)
Sucheta Soundarajan (Syracuse University, USA)
Mauro Sozio (Télécom ParisTech, France)
Marc Spaniol (University of Caen Normandy, France)
Anna Squicciarini (Pennsylvania State University, USA)
Mudhakar Srivatsa (IBM, USA)

Steffen Staab (University Koblenz-Landau, Germany; University of Southampton, UK)
John Stankovic (University of Virginia, USA)
Milan Stankovic (Sépage and STIH / Paris-Sorbonne University, France)
Yannis Stavrakas (Institute for the Management of Information Systems, Greece)
Harald Steck (Netflix, USA)
Maarten Van Steen (University of Twente, Netherlands)
Deian Stefan (University of California San Diego, USA)
Armando Stellato (University of Rome Tor Vergata, Italy)
Marc Stoecklin (IBM, USA)
Audun Stolpe (Norwegian Defence Research Establishment, Norway)
Umberto Straccia (Institute of Information Science and Technologies, Italy)
Gianluca Stringhini (Boston University, USA)
Markus Strohmaier (RWTH Aachen University and GESIS, Germany)
Fabian M. Suchanek (Télécom ParisTech, France)
Torsten Suel (New York University, USA)
Paul Suganthan (Google, USA)
Kazunari Sugiyama (National University of Singapore, Singapore)
Salmin Sultana (Intel, USA)
Aixin Sun (Nanyang Technological University, Singapore)
Hari Sundaram (University of Illinois Urbana-Champaign, USA)
Neel Sundaresan (Microsoft, USA)
York Sure-Vetter (Karlsruhe Institute of Technology, Germany)
Neeraj Suri (Technical University of Darmstadt, Germany)
Marcin Sydow (Polish Japanese Institute of Information Technology and Polish Academy of Sciences, Poland)
Panagiotis Symeonidis (Free University of Bozen-Bolzano, Italy)
Vasilis Syrgkanis (Microsoft, USA)
Pedro Szekely (University of Southern California, USA)
Andrea Tagarelli (University of Calabria, Italy)
Hassan Takabi (University of North Texas, USA)
Hideaki Takeda (National Institute of Informatics, Japan)
Ben Tan (Tencent, China)
Chenhao Tan (University of Colorado Boulder, USA)
Kian-Lee Tan (National University of Singapore, Singapore)
Hao Tang (Massachusetts Institute of Technology, USA)
Jian Tang (HEC Montreal and MILA, Canada)
Jie Tang (Tsinghua University, China)
Jiliang Tang (Michigan State University, USA)
Richard Tang (Syracuse University, USA)
Sasu Tarkoma (University of Helsinki, Finland)
Nikolaj Tatti (University of Helsinki, Finland)
Wallapak Tavanapong (Iowa State University, USA)
Andrea Tettamanzi (University of Nice Sophia Antipolis, France)
Kia Teymourian (Boston University, USA)
Dhavalkumar Thakker (University of Bradford, UK)
Krishnaprasad Thirunarayan (Wright State University, USA)
Ilaria Tiddi (Vrije University, Netherlands)
Thanassis Tiropanis (University of Southampton, UK)
Srikanta Tirthapura (Iowa State University, USA)
Mitul Tiwari (LinkedIn, USA)
Marko Tkalcic (Free University of Bozen-Bolzano, Italy)
Konstantin Todorov (University of Montpellier, France)
Gabriele Tolomei (University of Padua, Italy)
Nicola Tonellotto (Institute of Information Science and Technologies, Italy)
Farouk Toumani (Blaise Pascal University, France)
Duc Thanh Tran (Institute AIFB, Germany)
Christoph Trattner (University of Bergen, Norway)
Cassia Trojahn (University of Toulouse-Jean Jaurès and IRIT, France)
Raphaël Troncy (Eurecom, France)
Andrew Trotman (University of Otago, New Zealand)
Pinelopi Troullinou (Open University, UK)
Konstantinos Tserpes (Harokopio University of Athens, Greece)
Tania Tudorache (Stanford University, USA)
Johan Ugander (Stanford University, USA)
Antti Ukkonen (University of Helsinki, Finland)

Joerg Unbehauen (Leipzig University, Germany)
Hossein Vahabi (Pandora Media, USA)
Alejandro A. Vaisman (Buenos Aires Institute of Technology, Argentina)
Emanuele Della Valle (Polytechnic University of Milan, Italy)
Onur Varol (Northeastern University, USA)
Luke Veldt (Purdue University, USA)
Suresh Venkatasubramanian (University of Utah, USA)
Karin Verspoor (University of Melbourne, Australia)
Maria Esther Vidal (Simón Bolívar University, Venezuela)
Serena Villata (CNRS / I3S, France)
Vishwa Vinay (Adobe Research, India)
Svitlana Volkova (Pacific Nothwest National Laboratory, USA)
Yana Volkovich (Xandr, USA)
Aleksandr Vorobev (Yandex, Russia)
Domagoj Vrgoc (Pontifical Catholic University of Chile, Chile)
Slobodan Vucetic (Temple University, USA)
V.G.Vinod Vydiswaran (University of Michigan, USA)
Claudia Wagner (GESIS - Leibniz Institute for the Social Sciences, Germany)
Simon Walk (Graz University of Technology, Austria)
Xiaojun Wan (Peking University, China)
Bin Wang (Chinese Academy of Sciences, China)
Gang Wang (Virginia Polytechnic Institute and State University, USA)
Haofen Wang (Shanghai Leyan Technologies, China)
Honggang Wang (University of Massachusetts, USA)
Hongning Wang (University of Virginia, USA)
Kewen Wang (Griffith University, Australia)
Quan Wang (Chinese Academy of Sciences, China)
Suhang Wang (Pennsylvania State University, USA)
Xin Wang (Tianjin University, China)
Xuanhui Wang (Google, USA)
Yue Wang (University of Michigan, USA)
Zhichun Wang (Beijing Normal University, China)
Mirjam Wattenhofer (Google, Switzerland)
Ingmar Weber (Qatar Computing Research Institute, Qatar)
Matthew Weber (University of Minnesota, USA)
Jinpeng Wei (University of North Carolina Charlotte, USA)
Xiaokai Wei (University of Illinois Chicago, USA)
Katrin Weller (GESIS - Leibniz Institute for the Social Sciences, Germany)
Chris Welty (Google, USA)
Christopher Welty (Google, USA)
Tim Weninger (University of Notre Dame, USA)
Robert West (École Polytechnique Fédérale de Lausanne, Switzerland)
Steven Euijong Whang (Korea Advanced Institute of Science and Technology, South Korea)
Joyce Jiyoung Whang (Sungkyunkwan University, South Korea)
Erik Wilde (CA Technologies, Switzerland)
Christopher Wilkens (Facebook, USA)
Martijn Willemsen (Eindhoven University of Technology, Netherlands)
Carey Williamson (University of Calgary, Canada)
Craig Wills (Worcester Polytechnic Institute, USA)
Lars Wolf (Technical University of Braunschweig, Germany)
Ran Wolff (Yahoo Research, Oath, Israel)
Bernard Wong (University of Waterloo, Canada)
Timothy Wood (George Washington University, USA)
James Wright (University of Alberta, USA)
Jiewen Wu (Agency for Science, Technology and Research and Institute for InfoComm Research, Singapore)
Shaomei Wu (Facebook, USA)
Wei Wu (Microsoft, China)
Xintao Wu (University of Arkansas, USA)
Eric Wustrow (University of Colorado Boulder, USA)
Lirong Xia (Rensselaer Polytechnic Institute, USA)
Lexing Xie (Australian National University, Australia)
Xing Xie (Microsoft Research Asia, China)
Chenyan Xiong (Carnegie Mellon University and Microsoft, USA)
Hui Xiong (Rutgers University, USA)
Hu Xu (University of Illinois Chicago, USA)
Jian Xu (Alibaba Group, USA)
Kevin S. Xu (University of Toledo, USA)
Makoto Yamada (RIKEN AIP, Japan)
Xiaoran Yan (Indiana University Bloomington, USA)
Liu Yang (University of Massachusetts Amherst, USA)
Tao Yang (University of California Santa Barbara, USA)

Danfeng Yao (Virginia Polytechnic Institute and State University, USA)
Attila A Yavuz (Oregon State University, USA)
I-Ling Yen (University of Texas Dallas, USA)
Xun Yi (RMIT University, Australia)
Dawei Yin (JD.com, USA)
Ming Yin (Purdue University, USA)
Rex Ying (Stanford University, USA)
Fan Yixing (Chinese Academy of Sciences, China)
Hwanjo Yu (Pohang University of Science and Technology, South Korea)
Ting Yu (Qatar Computing Research Institute, Qatar)
Wei Yu (Towson University, USA)
Fouad Zablith (American University of Beirut, Lebanon)
Hamed Zamani (University of Massachusetts Amherst, USA)
Ondřej Zamazal (University of Economics Prague, Czechia)
Stefano Zanero (Polytechnic University of Milan, Italy)
Nicola Zannone (Eindhoven University of Technology, Netherlands)
Benjamin Zapilko (GESIS - Leibniz Institute for the Social Sciences, Germany)
Amrapali Zaveri (Maastricht University, Netherlands)
Dell Zhang (Birkbeck University of London, UK)
Fuzheng Zhang (Meituan, China)
Jenny Zhang (RMIT University, Australia)
Jiawei Zhang (Florida State University, USA)
Mingyang Zhang (Google, USA)
Lei Zhang (LinkedIn, USA)
Liang Zhang (LinkedIn, USA)
Min Zhang (Tsinghua University, China)
Peng Zhang (Tianjin University, China)
Qi Zhang (Fudan University, China)
Rui Zhang (University of Melbourne, Australia)
Si Zhang (Arizona State University, USA)
Xiangliang Zhang (King Abdullah University of Science and Technology, Saudi Arabia)
Yongfeng Zhang (Rutgers University, USA)
Ziqi Zhang (University of Sheffield, UK)
Le Zhao (Houzz, USA)
Elena Zheleva (University of Illinois Chicago, USA)
Vincent W. Zheng (Advanced Digital Sciences Center, Singapore)
Yong Zheng (Illinois Institute of Technology, USA)
Ke Zhou (University of Nottingham, UK)
Zhanxing Zhu (Peking University, China)
Roger Zimmermann (National University of Singapore, Singapore)
Marinka Zitnik (Stanford University, USA)
Chengqing Zong (Chinese Academy of Sciences, China)
Bin Zou (University College London, UK)
Arkaitz Zubiaga (University of Warwick, UK)
Guido Zuccon (University of Queensland, Australia)
Catherine Faron Zucker (University of Nice Sophia Antipolis, France)

Research Track Additional Reviewers

Hossein Karkeh Abadi (King's College London, UK)
Andres Abeliuk (University of Southern California, USA)
Seun Adekunle (Maastricht University, Netherlands)
David Adelani (Max Planck Institute for Software Systems, Germany)
Bijaya Adhikari (Virginia Polytechnic Institute and State University, USA)
Rachit Agarwal (INRIA, France)
Rehan Ahmad (University of Toledo, USA)
Sina Ahmadi (Dublin City University, Ireland)
Nirav Ajmeri (North Carolina State University, USA)
Cuneyt Gurcan Akcora (University of Texas Dallas, USA)
Betty van Aken (Beuth University of Applied Sciences Berlin, Germany)
Abdulatif Alabdulatif (RMIT University, Australia)
Ahmed Ali-Eldin (University of Massachusetts, USA)
Nazanin Alipourfard (University of Southern California, USA)
Pegah Alizadeh (Pôle Universitaire Léonard de Vinci, France)
Abdullah Aljebreen (Temple University, USA)

Hussein Al-Olimat (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Kareem Amin (Google, USA)
Amina Annane (École Nationale Supérieure d'Informatique, Algeria)
Spiros Antonatos (IBM Research, Ireland)
Makan Arastuie (University of Toledo, USA)
Frederico Araujo (IBM, USA)
Sebastian Arnold (Beuth University of Applied Sciences Berlin, Germany)
Akhil Arora (École Polytechnique Fédérale de Lausanne, Switzerland)
Ramnik Arora (Facebook, USA)
Zahra Ashktorab (IBM, USA)
Omid Askarisichani (University of California Santa Barbara, USA)
Aref Asvadishirehjini (University of Texas Dallas, USA)
Buse Gul Atli (Aalto University, Finland)
Sepideh Avizheh (University of Calgary, Canada)
Mihai Avram (University of Illinois Urbana-Champaign, USA)
Tobias Backes (GESIS - Leibniz Institute for the Social Sciences, Germany)
Adam Badawy (University of Southern California, USA)
Paritosh Bahirat (Clemson University, USA)
Ramah Al Balawi (University of Illinois Chicago, USA)
Antonio Balderas (University of Cadiz, Spain)
Nathalie Baracaldo (University of Pittsburgh, USA)
Gianluca Bardaro (Polytechnic University of Milan, Italy)
Farokh Bastani (University of Texas Dallas, USA)
Favyen Bastani (Massachusetts Institute of Technology, USA)
Osbert Bastani (Stanford University, USA)
Emanuele Bastianelli (Open University, UK)
Víctor Gutiérrez Basulto (University of Cardiff, UK)
Matthias Baumgartner (University of Zurich, Switzerland)
Niloofar Bayat (Columbia University, USA)
Soheil Behnezhad (University of Maryland, USA)
Rouzbeh Behnia (Oregon State University, USA)
Vibha C Belavadi (University of Texas Dallas, USA)
Francois Belletti (Google, USA)
Chemseddine Berbague (University of Annaba, Algeria)
Niels van Berkel (University of Melbourne, Australia)
Daniel Bernau (SAP, Germany)
Bruhadeshwar Bezawada (Mahindra École Centrale, India)
Shreyansh Bhatt (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Suchanda Bhattacharyya (RWTH Aachen University, Germany)
Federico Bianchi (University of Milano-Bicocca, Italy)
Umanga Bista (Australian National University, Australia)
Russa Biswas (FIZ Karlsruhe, Germany)
Aleksandar Bojchevski (Technical University of Munich, Germany)
Katarina Boland (GESIS - Leibniz Institute for the Social Sciences, Germany)
Geoffray Bonnin (University of Lorraine, France)
Ilaria Bordino (UniCredit, Italy)
Adam Bowers (Missouri University of Science and Technology, USA)
Barry Bradlyn (University of Illinois Urbana-Champaign, USA)
Olympia Brikis (Siemens, Germany)
Marc Bron (Schibsted, Netherlands)
Andrew Brooks (Outreach, USA)
Yi Bu (Indiana University, USA)
Qiong Bu (University of Southampton, UK)
Karan Budhraja (University of Maryland Baltimore County, USA)
Josemar Caetano (Federal University of Minas Gerais, Brazil)
Antonio Caliò (University of Calabria, Italy)
Javier Cánovas (Open University of Catalonia, Spain)
Da Cao (Hunan University, China)
Arthur Capozzi (University of Torino, Italy)
Samuel Carton (University of Michigan, USA)
Arnaud Castelltord (University of Montpellier, France)
Remy Cazabet (Claude Bernard University Lyon 1, France)
Stefano Cecconello (University of Padova, Italy)
Remzi Çelebi (Maastricht University, Netherlands)
Patricio Cerda (Pontifical Catholic University of Chile, Chile)

Abhijnan Chakraborty (Max Planck Institute for Software Systems, Germany)
Pierre-Antoine Champin (LIRIS, France)
Erick Chandra (National Taiwan University, Taiwan)
Olivier Chapelle (Google, USA)
Victor Charpenay (Siemens, Germany)
Sze Yiu Chau (Purdue University, USA)
Sneha Chaudhari (LinkedIn, USA)
Chen Chen (Arizona State University, USA)
Cheng Chen (University of Illinois Chicago, USA)
Hanqing Chen (IBM, USA)
Jiaoyan Chen (University of Oxford, UK)
Xi Chen (Columbia University, USA)
Yuanzhao Chen (King Abdullah University of Science and Technology, Saudi Arabia)
Zhou Chen (Hong Kong University of Science and Technology, Hong Kong)
Agnese Chiatti (Open University, UK)
Hana Chockler (King's College London, UK)
Nachshon Cohen (Amazon, Israel)
Pietro Colombo (University of Insubria, Italy)
Florin Constantin (Harvard University, USA)
Alessio Conte (University of Pisa, Italy)
Francisco Javier Marmolejo Cossío (University of Oxford, UK)
Shaobo Cui (Tsinghua University, China)
Shujie Cui (Imperial College London, UK)
Tiago Cunha (University of Michigan, USA)
Debajyoti Das (Purdue University, USA)
Ashlynn Daughton (University of Colorado Boulder, USA)
Neisarg Dave (Pennsylvania State University, USA)
Abir De (Max Planck Institute for Software Systems, Germany)
Ambra Demontis (University of Cagliari, Italy)
Tyler Derr (Michigan State University, USA)
Nirmit Desai (IBM, USA)
Ernesto Diaz-Aviles (Libre AI, Ireland)
Kaize Ding (Arizona State University, USA)
Thanh-Nam Doan (University at Albany - SUNY, USA)
Shahar Dobzinski (Weizmann Institute of Science, Israel)
Rafael Rego Drumond (University of Hildesheim, Germany)
Edlira Dushku (Sapienza University of Rome, Italy)
Monireh Ebrahimi (Wright State University, USA)
Vasilis Efthymiou (IBM, USA)
Mohamed Ben Ellefi (LIS-LAB / Aix-Marseille University, France)
Gonenc Ercan (Hacettepe University, Turkey)
Özge Sevgili Ergüven (University of Hamburg, Germany)
Benjamin Eriksson (Chalmers University of Technology, Sweden)
Lisette Espín-Noboa (GESIS - Leibniz Institute for the Social Sciences, Germany)
Jonas Falkner (University of Hildesheim, Germany)
Wenqi Fan (City University of Hong Kong, Hong Kong)
Jie-Yu Fang (National Cheng Kung University, Taiwan)
Adriano Fazzone (Sapienza University of Rome, Italy)
Geli Fei (Google, USA)
Michael Fell (University of Nice Sophia Antipolis, France)
Fuli Feng (National University of Singapore, Singapore)
Lampros Flokas (Columbia University, USA)
Huber Flores (University of Helsinki, Finland)
Jorge Garcia Flores (LIPN CNRS University of PARIS 13, France)
Giorgos Flouris (FORTH, Greece)
Rafael Frongillo (University of Colorado Boulder, USA)
Kata Gábor (INALCO, France)
Edoardo Galimberti (ISI Foundation, Italy)
Michael Gamon (Microsoft Research AI, USA)
Ankit Gangwal (University of Padova, Italy)
Yifan Gao (Chinese University of Hong Kong, Hong Kong)
Zheng Gao (Indiana University, USA)
Zhenxiang Gao (University of Southern California, USA)
Camilo Garrido (University of Chile, Chile)
Peter Gašpar (Slovak University of Technology, Slovakia)
Fabio De Gaspari (Sapienza University of Rome, Italy)
Paolo Gasti (New York Institute of Technology, USA)
Manas Gaur (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Laura Genga (Marche Polytechnic University, Italy)

Michele Gentili (Sapienza University of Rome, Italy)
Björn Gernert (Technical University of Braunschweig, Germany)
Ganesh Ghalme (Indian Institute of Science Bangalore, India)
Majid Ghasemi-Gol (University of Southern California, USA)
Raji Ghawi (Technical University Munich, Germany)
Mohammad Amin Ghiasi (University of Maryland, USA)
Pouya Ghiasnezhadomran (Griffith University, Australia)
Jaume Gibert (Eurecat, Spain)
Sujatha Das Gollapalli (Agency for Science, Technology and Research, Singapore)
Sreenivas Gollapudi (Google, USA)
Ken Goss (University of Missouri, USA)
Sayan Goswami (Louisiana State University, USA)
Palash Goyal (University of Southern California, USA)
Andreea Grigoriu (Maastricht University, Netherlands)
Stephan Grimm (Siemens, Germany)
Yu Gu (Nanjing University, China)
Yiming Gu (Uber, USA)
Gabriele Gualandi (Sapienza University of Rome, Italy)
Sujit Gujar (International Institute of Information Technology Hyderabad, India)
Hasini Gunasinghe (Purdue University, USA)
Wentian Guo (National University of Singapore, Singapore)
Hui Guo (North Carolina State University, USA)
Vinayak Gupta (Indian Institute of Technology Delhi, India)
Dhruv Gupta (Max Planck Institute for Informatics, Germany)
Tural Gurbanov (Free University of Bozen-Bolzano, Italy)
Lucas de Haas (de Persgroep, Netherlands)
Haraldur Hallgrimsson (University of California Santa Barbara, USA)
Lei Han (University of Queensland, Australia)
Peng Han (King Abdullah University of Science and Technology, Saudi Arabia)
Qing Han (University of California Irvine, USA)
Christian Hansen (University of Copenhagen, Denmark)
Yannis Haralambous (IMT Atlantique, France)
Robert Hartung (Technical University of Braunschweig, Germany)
Rakebul Hasan (Siemens, Germany)
Dafang He (Pennsylvania State University, USA)
Lihong He (Temple University, USA)
Yangyang He (Clemson University, USA)
Lars Heling (Karlsruhe Institute of Technology, Germany)
Anthony Hernandez (University of South Florida, USA)
Daniel Hernandez (University of Chile, Chile)
Nestor Hernandez (Florida International University, USA)
Jose Miguel Herrera (University of Chile, Chile)
Danula Hettiachchi (University of Melbourne, Australia)
Briland Hitaj (Stevens Institute of Technology, USA)
Dorjan Hitaj (Sapienza University of Rome, Italy)
Tuan-Anh Hoang (L3S Research Center, Germany)
Maartje ter Hoeve (University of Amsterdam, Netherlands)
Kijae Hong (Pohang University of Science and Technology, South Korea)
Sameera Horawalavithana (University of South Florida, USA)
Yasanka Sameera Horawalavithana (University of South Florida, USA)
Homa Hosseinmardi (University of Southern California, USA)
Keke Huang (Nanyang Technological University, Singapore)
Gaoping Huang (Purdue University, USA)
Di Huang (University of Southern California, USA)
Christoph Hube (L3S Research Center, Germany)
Julio Hurtado (Pontifical Catholic University of Chile, Chile)
Syed Hussain (Purdue University, USA)
Georgiana Ifrim (Dublin City University, Ireland)
Raphael Isemann (Chalmers University of Technology, Sweden)
Morshed Islam (University of Calgary, Canada)
Rolf Jagerman (University of Amsterdam, Netherlands)

Fuad Jamour (King Abdullah University of Science and Technology, Saudi Arabia)
Jeroen Jansze (de Persgroep, Netherlands)
Jianhui Ji (Alibaba Group, China)
Shiyu Ji (University of California Santa Barbara, USA)
Mitchell Joblin (Siemens, Germany)
Isaac Johnson (Northwestern University, USA)
Alex Jones (University of California Santa Barbara, USA)
Woohwan Jung (Seoul National University, South Korea)
Ruthwik Junuthula (University of Toledo, USA)
Mika Juuti (Aalto University, Finland)
Tobias Käfer (Karlsruhe Institute of Technology, Germany)
Dimitris Kalimeris (Harvard University, USA)
Bo Kang (Ghent University, Belgium)
Amir Kantor (IBM, Israel)
Hsien-Te Kao (University of Southern California, USA)
Giannis Karamanolakis (Columbia University, USA)
Danae Pla Karidi (Institute for the Management of Information Systems and National Technical University of Athens, Greece)
Leila Karimi (University of Pittsburgh, USA)
Ljubica Kärkkäinen (Technical University of Munich, Germany)
Tobias Kaufmann (University of Oslo, Norway)
Xiangyu Ke (Nanyang Technological University, Singapore)
Andrei Kelarev (University of Newcastle, UK)
Prashant Khare (Open University, UK)
Byungju Kim (Pohang University of Science and Technology, South Korea)
Jeonghwan Kim (Pohang University of Science and Technology, South Korea)
Kunho Kim (Pennsylvania State University, USA)
Konstantin Kobs (University of Wuerzburg, Germany)
Furkan Kocayusufoglu (University of California Santa Barbara, USA)
Sarah Kohail (University of Hamburg, Germany)
Michal Kompan (Slovak University of Technology, Slovakia)
Quyu Kong (Australian National University, Australia)
Tobias Koopmann (University of Wuerzburg, Germany)
Aljaz Kosmerlj (Jozef Stefan Institute, Slovenia)
Gangeshwar Krishnamurthy (Agency for Science, Technology and Research, Singapore)
Adit Krishnan (University of Illinois Urbana-Champaign, USA)
Rachel Krohn (University of Notre Dame, USA)
Cedric Kulbach (Forschungszentrum Informatik, Germany)
Masahito Kumano (Ryukoku University, Japan)
Suhansanu Kumar (University of Illinois Urbana-Champaign, USA)
Muni Venkateswarlu Kumaramangalam (University of Calgary, Canada)
Suyong Kwon (Seoul National University, South Korea)
Shibamouli Lahiri (University of Michigan, USA)
Mirko Lai (University of Turin, Italy)
Sarasi Lalithsena (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Robin Lamarche-Perrin (Laboratoire d'Informatique de Paris 6, France)
Riccardo Lancellotti (University of Modena and Reggio Emilia, Italy)
Alfonso Landin (University of A Coruña, Spain)
Juho Lauri (Nokia Bell Labs, Ireland)
Konstantina Lazaridou (University of Potsdam, Germany)
Arnon Lazerson (Amazon, Israel)
Philip Lazos (Oxford University, UK)
Tam Le (RIKEN AIP, Japan)
Christopher Leckie (University of Melbourne, Australia)
Dongha Lee (Pohang University of Science and Technology, South Korea)
Grace Lee (Nanyang Technological University, Singapore)
Seung-Min Lee (Pohang University of Science and Technology, South Korea)
Wonseok Lee (Pohang University of Science and Technology, South Korea)
Alyssa Lees (Google, USA)
Oliver Lehmberg (University of Mannheim, Germany)

Jérôme Lenßen (RWTH Aachen University, Germany)
Chang Li (University of Amsterdam, Netherlands)
Chao Li (University of Pittsburgh, USA)
Dan Li (University of Amsterdam, Netherlands)
Jane Li (Victoria University of Wellington, New Zealand)
Jing Li (Nanyang Technological University, Singapore)
Junjie Li (Chinese Academy of Sciences, China)
Haoran Li (Chinese Academy of Sciences, China)
Shidi Li (Australian National University, Australia)
Shuxin Li (Nanjing University, China)
Xin Li (Chinese University of Hong Kong, Hong Kong)
Jianxun Lian (University of Science and Technology of China, China)
Hongyu Lin (Chinese Academy of Sciences, China)
Vinicius Monteiro de Lira (Institute of Information Science and Technologies, Italy)
Pasquale Lisena (Eurecom, France)
Qingxia Liu (Nanjing University, China)
Renhao Liu (University of South Florida, USA)
Xi Liu (Texas A&M University, USA)
Xiaoning Liu (RMIT University, Australia)
Xuan Liu (Michigan State University, USA)
Yanchi Liu (Rutgers University, USA)
Yong Liu (Outreach, USA)
Zhe Liu (IBM, USA)
Zheng Liu (Hong Kong University of Science and Technology, Hong Kong)
Babak Loni (Delft University of Technology, Netherlands)
Jun Lu (RIKEN AIP, Japan)
Ana Lucic (University of Amsterdam, Netherlands)
Malte Ludewig (Technical University of Dortmund, Germany)
Chu Luo (University of Melbourne, Australia)
Yao Ma (Michigan State University, USA)
Simona Maggio (Rakuten Institute of Technology, France)
Gabriel Magno (Federal University of Minas Gerais, Brazil)
Ahmad Mahmoody (Microsoft, USA)
Laura Mahrenbach (Technical University of Munich, Germany)
Gengchen Mai (University of California Santa Barbara, USA)
Roland Maio (Columbia University, USA)
Ali Makhdoumi (Massachusetts Institute of Technology, USA)
Ali Makhdoumikakhaki (Massachusetts Institute of Technology, USA)
Ankur Mali (Pennsylvania State University, USA)
Domenico Mandaglio (University of Calabria, Italy)
Samuel Marchal (Aalto University, Finland)
Mirco Marchetti (University of Modena and Reggio Emilia, Italy)
Gabriele Marini (University of Melbourne, Australia)
Manuel Jesus Cobo Martin (University of Cadiz, Spain)
Akira Matsui (University of Southern California, USA)
Andrea Mauri (Delft University of Technology, Netherlands)
Panagiotis Mavridis (Delft University of Technology, Netherlands)
Sourav Medya (University of California Santa Barbara, USA)
Shagufta Mehnaz (Purdue University, USA)
Ninareh Mehrabi (University of Southern California, USA)
Cristina Menghini (Sapienza University of Rome, Italy)
Vijay Menon (University of Waterloo, Canada)
Martino Mensio (Open University, UK)
Sepideh Mesbah (Delft University of Technology, Netherlands)
Pablo Messina (Pontifical Catholic University of Chile, Chile)
Andrea Michienzi (University of Pisa, Italy)
Alice Millour (Paris-Sorbonne University, France)
Mohsen Minaei (Purdue University, USA)
Seyed Ali Mirheidari (University of Trento, Italy)
Vitaly Mirkis (Huawei, Israel)
Mehrnoosh Mirtaheri (University of Southern California, USA)
Shaunak Mishra (University of California Los Angeles, USA)
Shubhanshu Mishra (University of Illinois Urbana-Champaign, USA)
Amita Misra (IBM, USA)

Ellen Mitsopoulou (Athens University of Economics and Business, Greece)
Serghei Mogoreanu (Siemens, Germany)
Negar Mokhberian (University of Southern California, USA)
Diego Monti (Polytechnic University of Turin, Italy)
Gabriela Montoya (Aalborg University, Denmark)
Ali Mousavi (Google, USA)
Nasim Mousavi (Emory University, USA)
Frederick Mubang (University of South Florida, USA)
Cristina Ioana Muntean (Institute of Information Science and Technologies, Italy)
Goran Muric (University of Southern California, USA)
Cataldo Musto (University of Bari, Italy)
Mohamed Nabeel (Hamad Bin Khalifa University, Qatar)
Vinod P Nair (University of Padova, Italy)
Moses Namara (Clemson University, USA)
Kanika Narang (University of Illinois Urbana-Champaign, USA)
Nicolò Navarin (University of Padova, Italy)
Zahra Nazari (Spotify, USA)
Arun Reddy Nelakurthi (Arizona State University, USA)
Sebastian Neumaier (Vienna University of Economics and Business, Austria)
Ka-Lok Ng (Chinese University of Hong Kong, Hong Kong)
Yan Bin Ng (Agency for Science, Technology and Research, Singapore)
Hoa Ngo (Commonwealth Scientific and Industrial Research Organisation, Australia)
Anna Nguyen (Karlsruhe Institute of Technology, Germany)
Thanh Nguyen (Iowa State University, USA)
Elnaz Nouri (Microsoft Research AI, USA)
Inna Novalija (Jozef Stefan Institute, Slovenia)
Tom Oberhauser (Beuth University of Applied Sciences Berlin, Germany)
Keiko Ono (Ryukoku University, Japan)
Eirik Opland (Norwegian Defence Research Establishment, Norway)
Yusuf Ozkaya (Georgia Institute of Technology, USA)
Ozgur Ozmen (Oregon State University, USA)
Swati Padhee (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Ankur Padia (University of Maryland Baltimore County, USA)
Renato Paes (Cornell University, USA)
Manuel Palomo-Duarte (University of Cadiz, Spain)
Panagiotis Papadakos (FORTH, Greece)
Maria Papadopouli (FORTH, Greece)
David Pardoe (LinkedIn, USA)
Chanyoung Park (Pohang University of Science and Technology, South Korea)
Yeon Su Park (Pohang University of Science and Technology, South Korea)
Raj Patel (George Mason University, USA)
Russell Paulet (RMIT University, Australia)
Hao Peng (University of Michigan, USA)
Jinhua Peng (Baidu, China)
Zhen Peng (Xi'an Jiaotong University, China)
Diego Perna (University of Calabria, Italy)
Domenico Perna (University of Calabria, Italy)
Pedro Pernias (University of Alicante, Spain)
Cindarella Petz (Technical University of Munich, Germany)
Patrick Philipp (Karlsruhe Institute of Technology, Germany)
Richard Platania (Louisiana State University, USA)
Edward Platt (University of Michigan, USA)
Chara Podimata (Harvard University, USA)
Marco Polignano (University of Bari, Italy)
Utkarsh Porwal (eBay, USA)
Jedrzej Potoniec (Poznan University of Technology, Poland)
Romila Pradhan (Purdue University, USA)
Sven Pullwitt (Technical University of Braunschweig, Germany)
Dan Puzzuoli (University of Ottawa and Carleton University, Canada)
Ehsan Qasemi (University of Southern California, USA)
Sharon Qian (Harvard University, USA)
Sihang Qiu (Delft University of Technology, Netherlands)
Georges Quénot (LIG-CNRS, France)
Lara Quijano (Charles III University of Madrid, Spain)
Dimitrios Rafailidis (Maastricht University, Netherlands)

Stephen Ragain (Stanford University, USA)
Manish Raghavan (Cornell University, USA)
Yves Raimond (Netflix, USA)
Tharindu Ranasinghe (University of Wolverhampton, UK)
Ashwini Rao (Carnegie Mellon University, USA)
Fang-Yu Rao (Purdue University, USA)
Ahmed Rashed (ISMLL / University of Hildesheim, Germany)
Baharak Rastegari (University of Southampton, UK)
Kan Ren (Shanghai Jiao Tong University, China)
Matt Revelle (George Mason University, USA)
Felipe del Rio (Pontifical Catholic University of Chile, Chile)
Or Rivlin (IBM, Israel)
Michael Rodler (University of Duisburg-Essen, Germany)
Mercedes Rodriguez-Garcia (University of Cadiz, Spain)
Shaurya Rohatgi (Pennsylvania State University, USA)
Wang Ruijie (Xi'an Jiaotong University, China)
Anisa Rula (University of Milano-Bicocca, Italy)
Natalia Rümmele (Siemens, Germany)
Arkaprava Saha (Nanyang Technological University, Singapore)
Koustuv Saha (Georgia Institute of Technology, USA)
Merve Sahin (Eurecom, France)
Fariborz Salehi (California Institute of Technology, USA)
Amir Salimi (Qualcomm Research, USA)
Iskander Sanchez-Rola (University of Deusto, Spain)
Seyed-Vahid Sanei-Mehri (Iowa State University, USA)
Alberto de Santos (Vodaphone, Spain)
Janina Sarol (University of Illinois Urbana-Champaign, USA)
Zhanna Sarsenbayeva (University of Melbourne, Australia)
Sanja Scepanovic (Nokia Bell Labs, Finland)
Sarah Scheffler (Boston University, USA)
Niko Schenk (Goethe University, Germany)
Claudio Schifanella (University of Turin, Italy)
Lukas Schmelzeisen (University Koblenz-Landau, Germany)
Andrew Schneider (Temple University, USA)
Jon Schneider (Princeton University, USA)
Rudolf Schneider (Beuth University of Applied Sciences Berlin, Germany)
Mirco Schönfeld (Technical University of Munich, Germany)
Simon Schweighofer (Complexity Science Hub Vienna, Austria)
Saeed Seddighin (University of Maryland, USA)
Ella Segev (Ben-Gurion University, Israel)
In Seo (Pohang University of Science and Technology, South Korea)
C. Seshadhri (University of California Santa Cruz, USA)
Vinay Setty (University of Stavanger, Norway)
Mo Sha (National University of Singapore, Singapore)
Shayan Shams (Louisiana State University, USA)
Jinjin Shao (University of California Santa Barbara, USA)
Hossam Sharara (University of Maryland, USA)
Setareh Sharifian (University of Calgary, Canada)
Sagar Sharma (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Minjeong Shin (Australian National University, Australia)
Hossein Shirazi (Colorado State University, USA)
Michal Shmueli-Scheuer (IBM, Israel)
Kai Shu (Arizona State University, USA)
Jose Luis Sierra (Complutense University of Madrid, Spain)
Satyaki Sikdar (University of Notre Dame, USA)
Jorge Silvestre (University of Valladolid, Spain)
Robert Sim (Microsoft Research AI, USA)
Jakob Grue Simonsen (University of Copenhagen, Denmark)
Jaspreet Singh (L3S Research Center, Germany)
Kuldeep Singh (Fraunhofer, Germany)
Valentina Sintsova (École Polytechnique Fédérale de Lausanne, Switzerland)
Alexander Sjösten (Chalmers University of Technology, Sweden)
J. David Smith (University of Florida, USA)
Michael Smith (University of California San Diego, USA)
Byeong Hoon So (Pohang University of Science and Technology, South Korea)

Gwangho Song (Seoul National University, South Korea)
Xingyi Song (University of Sheffield, UK)
Xuemeng Song (Shandong University, China)
Sandeep Soni (Georgia Institute of Technology, USA)
Blerina Spahiu (University of Milano-Bicocca, Italy)
Adam Spiro (IBM, Israel)
Riccardo Spolaor (Oxford University, UK)
Kostas Stefanidis (FORTH, Greece)
Michael Steininger (University of Wuerzburg, Germany)
Ana Stoica (Columbia University, USA)
Win Suen (Hong Kong University of Science and Technology, Hong Kong)
Marie-Theres Suer (Bosch and Technical University of Braunschweig, Germany)
Jun Sun (University of Koblenz-Landau, Germany)
Lichao Sun (University of Illinois Chicago, USA)
Yawei Sun (Nanjing University, China)
Yiwei Sun (Pennsylvania State University, USA)
Swathi Sunder (Siemens, Germany)
Chul Sung (IBM, USA)
Abhijit Suprem (Georgia Institute of Technology, USA)
Sebastian Surminski (University of Duisburg-Essen, Germany)
Marcella Tambuscio (University of Turin, Italy)
Biaoshuai Tao (University of Michigan, USA)
Nazgol Tavabi (Sharif University of Technology, Iran)
Ravi Tejwani (Massachusetts Institute of Technology, USA)
Nicolas Tempelmeier (L3S Research Center, Germany)
Mingfei Teng (Rutgers University, USA)
Thiviyan Thanapalasingam (Open University, UK)
Hoang Thang (Oregon State University, USA)
Son Thanh (Agency for Science, Technology and Research, Singapore)
Steffen Thoma (Karlsruhe Institute of Technology, Germany)
Pam Thomas (University of Notre Dame, USA)
Tabea Tietz (FIZ Karlsruhe, Germany)
Dimitrios Tomaras (Athens University of Economics and Business, Greece)
Sara Tonelli (Fondazione Bruno Kessler, Italy)
Leonardo Tonetto (Technical University of Munich, Germany)
Alasdair Tran (Australian National University, Australia)
Roberto Trani (Institute of Information Science and Technologies, Italy)
Salvatore Trani (Institute of Information Science and Technologies, Italy)
Anton Tsitsulin (Hasso-Plattner Institute, Germany)
Rima Türker (FIZ Karlsruhe, Germany)
Juergen Umbrich (Vienna University of Economics and Business, Austria)
Nitish Uplavikar (University of Missouri, USA)
Alessandra Urbinati (ISI Foundation, Italy)
Przemek Uznanski (University of Wrocław, Poland)
Rohit Vaish (Rensselaer Polytechnic Institute, USA)
Camilo Valenzuela (Federico Santa María Technical University, Chile)
Praveen Venkateswaran (University of California Irvine, USA)
Gaurav Verma (Adobe, India)
Luca Versari (University of Pisa, Italy)
Salvatore Vilella (ISI Foundation, Italy)
Bhanukiran Vinzamuri (IBM, USA)
Quaizar Vohra (Passage AI, USA)
Pavlos Vougiouklis (University of Southampton, UK)
Johannes Wachs (Central European University, Hungary)
Hongwei Wang (Stanford University, USA)
Qinglong Wang (McGill University, Canada)
Shaonan Wang (Chinese Academy of Sciences, China)
Tianhao Wang (Purdue University, USA)
Wenwei Wang (Hong Kong University of Science and Technology, Hong Kong)
Xiang Wang (National University of Singapore, Singapore)
Yanhao Wang (National University of Singapore, Singapore)
Yifan Wang (Chinese Academy of Sciences, China)
Zhe Wang (Griffith University, Australia)
Zhiwei Wang (Michigan State University, USA)
John Watrous (University of Waterloo, Canada)
Benjamin Weggenmann (SAP, Germany)
Junqiu Wei (Huawei, Hong Kong)
Daniel Weidele (University of Konstanz, Germany)

Ruwan Wickramarachchi (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Gregor Wiedemann (University of Hamburg, Germany)
Senuri Wijenayake (University of Melbourne, Australia)
Thilini Wijesiriwardene (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Daricia Wilkinson (Clemson University, USA)
Benjamin Winter (Beuth University of Applied Sciences Berlin, Germany)
René Witte (Concordia University, Canada)
Fangzhao Wu (Microsoft, China)
Liang Wu (Arizona State University, USA)
Siqi Wu (Australian National University, Australia)
Tianxing Wu (Nanyang Technological University, Singapore)
Xueyang Wu (Hong Kong University of Science and Technology, Hong Kong)
Zhe Wu (eBay, USA)
Zhiwei Steven Wu (University of Minnesota, USA)
Liu Xi (Texas A&M University, USA)
Yingce Xia (Microsoft, China)
Chuan Xiao (Nagoya University, Japan)
Fei Xiao (Alibaba Group, China)
Zhu Xiao (Hunan University, China)
Kun Xie (Hunan University, China)
Jinlai Xu (University of Pittsburgh, USA)
Min Xu (Fudan University, China)
Erman Yafay (Middle East Technical University, Turkey)
Hadi Yami (University of Maryland, USA)
Peng Yan (3M, USA)
Shen Yan (University of Southern California, USA)
Carl Yang (University of Illinois Urbana-Champaign, USA)
Lin Yang (Hong Kong University of Science and Technology, Hong Kong)
Xiao Yang (Pennsylvania State University, USA)
Xu Yang (RMIT University, Australia)
Yang Yang (Rutgers University, USA)
Yatao Yang (Alibaba Group, China)
Jin-Ge Yao (Peking University, China)
Wei Ye (University of California Santa Barbara, USA)
Hongyuan You (University of California, Santa Barbara, USA)
Fang-Yi Yu (University of Michigan, USA)
Shuhan Yuan (University of Arkansas, USA)
Zixuan Yuan (Rutgers University, USA)
Omnia Zayed (Dublin City University, Ireland)
Denghui Zhang (Rutgers University, USA)
Dongxiang Zhang (University of Electronic Science and Technology, China)
Junjie Zhang (Wright State University, USA)
Ke Zhang (University of Pittsburgh, USA)
Wuwei Zhang (Purdue University, USA)
Xingzhou Zhang (Chinese Academy of Sciences, China)
Yi Zhang (Hong Kong University of Science and Technology, Hong Kong)
Yu Zhang (Iowa State University, USA)
Keran Zhao (University of Illinois Chicago, USA)
Li Zhao (Microsoft, USA)
Shubin Zhao (Google, USA)
Xiangyu Zhao (Michigan State University, USA)
Xiaoting Zhao (Etsy, USA)
Yongjun Zhao (Chinese University of Hong Kong, Hong Kong)
Zhibing Zhao (Rensselaer Polytechnic Institute, USA)
Qinghai Zhou (Arizona State University, USA)
Yan Zhou (University of Texas Dallas, USA)
Yao Zhou (Arizona State University, USA)
Zhen Zhu (Stanford University, USA)
Matteo Zignani (University of Milan, Italy)

Demonstrations Program Committee

Y Luca Maria Aiello (Nokia Bell Labs, UK)
Hussein Al-Olimat (Wright State University, USA)
Muhammad Intizar Ali (National University of Ireland Galway, Ireland)
Ghislain Auguste Atemezing (Mondeca, France)
Xiao Bai (Yahoo Research, USA)
Payam Barnaghi (University of Surrey, UK)
Paolo Bellavista (University of Bologna, Italy)
Maria Bermudez-Edo (University of Granada, UK)
Karima Boudaoud (I3S-CNRS / University of Nice Sophia Antipolis, France)
Andrei Ciortea (University of St. Gallen, Switzerland)
Luca Costabello (Accenture Labs, Ireland)
Anwitaman Datta (Nanyang Technological University, Singapore)
Soumya Kanti Datta (EURECOM, France)
Yasmin Fathy (University of Surrey, UK)
Raúl García-Castro (Polytechnic University of Madrid, Spain)
Manas Gaur (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Paul Grace (University of Southampton, UK)
Christophe Gravier (Jean Monnet University, France)
Bin Guo (Telecom SudParis, France)
Amelie Gyrard (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Manfred Hauswirth (Technical University of Berlin and Fraunhofer FOKUS, Germany)
Erisa Karafili (Imperial College London, UK)
Mehdi Kaytoue (Infologic, France)
David Laniado (Eurecat - Technology Centre of Catalonia, Spain)
Gyu Myoung Lee (Liverpool John Moores University, UK)
Chunbin Lin (Amazon, USA)
Fangming Liu (Huazhong University of Science and Technology, China)
Maria Maleshkova (University of Bonn, Germany)
Albert Meroño-Peñuela (Vrije Universiteit Amsterdam, Netherlands)
Franco Maria Nardini (Institute of Information Science and Technologies, Italy)
David Nichols (University of Waikato, New Zealand)
Salvatore Orlando (Ca' Foscari University of Venice, Italy)
Josiane Xavier Parreira (Siemens, Austria)
Pankesh Patel (Fraunhofer CESE, USA)
Filipa Peleja (New University of Lisbon, Portugal)
Raffaele Perego (ISTI-CNR, Italy)
Adriano Pereira (Federal University of Minas Gerais, Brazil)
Danh Le Phuoc (Technical University of Berlin, Germany)
Gauthier Picard (Laboratoire Hubert Curien, Institut Henri Fayol, and MINES Saint-Etienne, France)
Thomas Ricatte (Criteo AI Lab, France)
Michele Ruta (Polytechnic University of Bari, Italy)
Saeedeh Shekarpour (University of Dayton, USA)
Bart Thomee (Google, USA)
Nicola Tonellotto (Institute of Information Science and Technologies, Italy)
Flavian Vasile (Criteo AI Lab, France)
Maria Esther Vidal (Simón Bolívar University, Venezuela)
Ziqi Wang (Facebook, UK)
Bebo White (SLAC National Accelerator Laboratory, USA)
Sanjaya Wijeratne (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Zhengming Xing (Criteo AI Lab, USA)
Pranjul Yadav (Criteo AI Lab, USA)
Achille Zappa (National University of Ireland Galway, Ireland)
Antoine Zimmermann (École des Mines de Saint-Étienne, France)

Demonstrations Additional Reviewers

Flavien Balbo (École des Mines de Saint-Étienne, France)
Cristian Consonni (University of Trento, France)
Walid Iguider (University of Cagliari, Italy)
Mirko Marras (University of Cagliari, Italy)
Swati Padhee (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Joy Prakash Sain (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Julien Tissier (Jean Monnet University, France)

Deep Learning for Solving Important Problems

Jeff Dean
Google
Mountain View, CA, USA, jeff@google.com

ABSTRACT

In this keynote we describe progress in work that our research teams have been doing over the past years, including advances in difficult problems in artificial intelligence, on building large-scale computer systems for machine learning research, and, in collaboration with many teams at Google, on applying our research and systems to dozens of Google products. Our group has open-sourced the TensorFlow system [2], a widely popular system designed to easily express machine learning ideas, and to quickly train, evaluate and deploy machine learning systems. We then highlight some of our research accomplishments, and relate them to the National Academy of Engineering's Grand Engineering Challenges for the 21st Century.

This is joint work with many people at Google.

CCS Concepts/ACM Classifiers

• Computing methodologies → Machine learning.

• Information systems → Search engine architectures and scalability.

Author Keywords

Applied research, large-scale machine learning, TensorFlow.

ACM Reference format:

Jeff Dean. 2019. Deep Learning for Solving Important Problems. In *Proceedings of WWW '19: The Web Conference (WWW '19), May 13, 2019, San Francisco, USA*. ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3308558.3313785

1 BIOGRAPHY

Jeff Dean [1] joined Google in 1999 and is currently a Google Senior Fellow and SVP of Google AI, Research, and Health. He and his collaborators are working on systems for speech recognition, computer vision, language understanding, and various other machine learning tasks. He has co-designed/implemented many generations of Google's crawling, indexing, and query serving systems, and co-designed/implemented major pieces of Google's initial advertising and AdSense for Content systems. He is also a co-designer and co-implementor of Google's distributed computing infrastructure, including the MapReduce, BigTable and Spanner systems, protocol buffers, the open-source TensorFlow system for machine learning, and a variety of internal and external libraries and developer tools.



Jeff received a Ph.D. in Computer Science from the University of Washington in 1996, working with Craig Chambers on whole-program optimization techniques for object-oriented languages. He received a B.S. in computer science & economics from the University of Minnesota in 1990. He is a member of the National Academy of Engineering, and of the American Academy of Arts and Sciences, a Fellow of the Association for Computing Machinery (ACM), a Fellow of the American Association for the Advancement of Sciences (AAAS), and a winner of the ACM Prize in Computing.

REFERENCES

[1] Jeff Dean: https://research.google.com/people/jeff
[2] TensorFlow: https://www.tensorflow.org/

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13, 2019, San Francisco, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313785

The Law of the Horse at 20: Phases of the Net

Lawrence Lessig
Harvard Law School
Cambridge, MA, USA, lessig@this.is

ABSTRACT

The Internet moves in phases, and we are entering the third in 20 years. In this keynote, using a framework drawn from the Law of the Horse [1], I describe the phase we are entering — the surveillance phase — and the threat it presents to society generally, and democracy in particular. Along the way, I offer an understanding of the Net circa 1999, and the phase that followed it, circa 2009. At each stage, our inability to govern has been a significant liability. In the phase we are entering, it will be devastating.

CCS Concepts/ACM Classifiers

• Social and professional topics → Computing / technology policy

• Social and professional topics → Surveillance

Author Keywords

Democracy, Internet law, cyberlaw, government technology policies, digital surveillance, Web technology policies.

ACM Reference format:

Lawrence Lessig. 2019. The Law of the Horse at 20: Phases of the Net. In *Proceedings of WWW '19: The Web Conference (WWW '19), May 13, 2019, San Francisco, USA.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3308558.3313786

1 BIOGRAPHY

Lawrence Lessig is the Roy L. Furman Professor of Law and Leadership at Harvard Law School. Prior to returning to Harvard, he taught at Stanford Law School, where he founded the Center for Internet and Society, and at the University of Chicago. Lessig is a founding board member of Creative Commons. He serves on the Scientific Board of AXA Research Fund. A member of the American Academy of Arts and Sciences and the American Philosophical Association, he has received numerous awards, including a Webby, the Free Software Foundation's Freedom Award, Scientific American 50 Award, and Fastcase 50 Award. Cited by The New Yorker as "the most important thinker on intellectual property in the Internet era," Lessig has focused much of his career on law and technology, especially as it affects copyright. His current work addresses "institutional corruption"—relationships which, while legal, weaken public trust in an institution—especially as that affects democracy.



His books include: Republic, Lost v2 (2015), Republic, Lost (2012), Remix: Making Art and Commerce Thrive in the Hybrid Economy (2009), Code v2 (2006), The Future of Ideas (2001), and Code and Other Laws of Cyberspace (2000). Lessig holds a BA in economics and a BS in management from the University of Pennsylvania, an MA in philosophy from Cambridge University, and a JD from Yale.

REFERENCES

[1] Lawrence Lessig. The Law of the Horse: What Cyberlaw Might Teach, Harvard Law Review 113, 501-549, 1999. URL: https://cyber.harvard.edu/works/lessig/finalhls.pdf

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13, 2019, San Francisco, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313786

Enlisting the Public to Build a Healthier Web Information Commons

Claire Wardle
TED Fellow
CIVIC, New York, NY, USA, claire@ted.com

ABSTRACT

Over the past three years, platforms, governments and a plethora of nonprofit initiatives have prioritized fighting online misinformation through a variety of different means. Yet the current framework is too fragmented to deliver global results. The big tech platforms have data, but no public accountability. Governments (mostly) have democratic legitimacy, but little information on what is actually going on in the platforms they're itching to regulate. And nonprofit initiatives too often lack the scale to affect change at the level needed. What if we came up with a dramatically new deliberative process that involves a global community of concerned citizens ready to share information and participate in consultations to improve collective decision-making? What if a more accountable, diverse and verifiable Web were still possible?

CCS Concepts/ACM Classifiers

• Social and professional topics → Computing / technology policy

• Social and professional topics → User characteristics

Author Keywords

Democracy, government technology policies, Web technology policies, user generated content.

ACM Reference format:

Claire Wardle. 2019. Enlisting the Public to Build a Healthier Web Information Commons. In *Proceedings of WWW '19: The Web Conference (WWW '19), May 13, 2019, San Francisco, USA*. ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3308558.3313787

1 BIOGRAPHY

Claire Wardle is a TED fellow working on a new participatory project aimed at building an information commons where people can reliably find factual information that serves their needs, hold accountable those who create or disseminate information for mass consumption and help build an information commons that is truly representative of the global community. Claire also leads the strategic direction and research for First Draft and is the co-founder of one of First Draft's founding partners, Eyewitness Media Hub. Before First Draft, she was the Research Director at the Tow Center for Digital Journalism at Columbia University Graduate School of Journalism. She was also head of social media for the United Nations Refugee Agency and Director of News Services for Storyful.



She is one of the world's experts on user-generated content (UGC), and has led substantial research projects—with an emphasis on qualitative research methodologies—investigating how UGC is handled by news organizations. She was a member of the World Economic Forum's Global Agenda Council on the Future of Information and Entertainment. She holds a PhD in Communications from the University of Pennsylvania.

More information about the speaker can be found at [1].

REFERENCES

[1] Claire Wardle. Personal website: https://clairewardle.com/

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13, 2019, San Francisco, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313787

Addressing Trust Bias for Unbiased Learning-to-Rank

Aman Agarwal
Cornell University
Ithaca, NY
aa2398@cornell.edu

Xuanhui Wang
Google Inc.
Mountain View, CA
xuanhui@google.com

Cheng Li
Google Inc.
Mountain View, CA
chgli@google.com

Mike Bendersky
Google Inc.
Mountain View, CA
bemike@google.com

Marc Najork
Google Inc.
Mountain View, CA
najork@google.com

ABSTRACT

Existing unbiased learning-to-rank models use counterfactual inference, notably Inverse Propensity Scoring (IPS), to learn a ranking function from biased click data. They handle the click incompleteness bias, but usually assume that the clicks are noise-free, i.e., a clicked document is always assumed to be relevant. In this paper, we relax this unrealistic assumption and study click noise explicitly in the unbiased learning-to-rank setting. Specifically, we model the noise as the position-dependent trust bias and propose a noise-aware Position-Based Model, named TrustPBM, to better capture user click behavior. We propose an Expectation-Maximization algorithm to estimate both examination and trust bias from click data in TrustPBM. Furthermore, we show that it is difficult to use a pure IPS method to incorporate click noise and thus propose a novel method that combines a Bayes rule application with IPS for unbiased learning-to-rank. We evaluate our proposed methods on three personal search data sets and demonstrate that our proposed model can significantly outperform the existing unbiased learning-to-rank methods.

KEYWORDS

Unbiased learning-to-rank; trust bias; inverse propensity scoring; click noise

ACM Reference Format:
Aman Agarwal, Xuanhui Wang, Cheng Li, Mike Bendersky, and Marc Najork. 2019. Addressing Trust Bias for Unbiased Learning-to-Rank. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313707

1 INTRODUCTION

Implicit user feedback (such as clicks, dwelling time etc.) is routinely logged in many applications such as web search and personal email search. Using this data for learning-to-rank (LTR) is attractive not only because it is abundantly available, but also because it is the only viable large-scale labeled data for many applications like personal search [32, 33]. Moreover, unlike explicit relevance judgments obtained from experts or crowd-sourcing which can become obsolete quickly, implicit feedback reflects the time-varying preferences of the actual user population and is easy to maintain.

Despite these clear benefits, LTR from implicit feedback, in particular clicks on search results, is challenging due to inherent biases in user behavior. For example, due to position bias, a result displayed at a higher position is more likely to receive clicks than a result at a lower position. Thus directly using clicks as relevance judgments can lead to biased results because (1) clicks are incomplete as relevant documents may be missed non-uniformly and (2) clicks are noisy since a click does not necessarily imply the document is relevant.

Recently, unbiased LTR has been actively studied as a promising approach to learn from biased click logs. It is based on counterfactual inference [21] and has been shown to outperform the traditional click modeling approaches [3]. The commonly used technique in unbiased LTR is Inverse Propensity Scoring (IPS). Different from the traditional IPS approaches used in counterfactual inference where the propensity is known, a key question in unbiased LTR is how to estimate the propensity – the probability that the relevance of a result is observed. Examination-based click models, and the Position-Based Model (PBM) in particular, have been leveraged for this. A nice property of PBM is that the examination bias in it is the propensity needed. Different techniques have been proposed to estimate the examination bias using result randomization [21, 32], random pairs harvesting [2], a regression-based Expectation-Maximization (EM) technique [33], and a dual learning algorithm [3].

Existing PBM methods mainly focus on addressing the click incompleteness bias, by assuming that clicks are noise-free. Such an assumption is unrealistic. How to address click noise for unbiased LTR is still an open question. Joachims et al. [21] showed the robustness of the IPS-based approach when *position-independent constant* noise is present in click data. However, as the user study in [19] showed qualitatively, there is a *trust bias* after examining results, capturing the intuition that users may overestimate (or underestimate) the relevance of a result given its high (or low) position, due to their trust in the effectiveness of the search application to rank relevant documents higher. Thus a non-relevant result may be clicked and a relevant result may not be clicked, and the probabilities are not constant but depend on the position of the result.

This paper is published under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 International (CC-BY-NC-ND 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY-NC-ND 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313707

Trust bias can be thought of as another form of position bias, but to the best of our knowledge, it has not been quantitatively formulated in the existing literature. In this paper, we introduce a noise-aware PBM, named TrustPBM, to model trust bias in user clicks. The standard PBM model assumes that an examined and relevant document is always clicked. We extend it to capture the following uncertainty: after examination, a relevant document can be missed, and meanwhile a non-relevant document can be clicked. TrustPBM is a click model that gives a rigorous mathematical formulation of the trust bias. However, its bias parameters cannot be estimated based on intervention like the result randomization used for PBM. Even if we can estimate bias parameters in TrustPBM, it is still unclear how to use them for unbiased LTR.

In this paper, we first show that TrustPBM can be effectively estimated using an EM algorithm, inspired by the previous work [33]. Since TrustPBM is no longer a factorization model, a purely IPS method for unbiased LTR is not feasible. We thus derive a novel and mathematically principled Bayes-IPS correction that uses a Bayesian approach for trust bias and IPS for examination bias. Conceptually, the Bayes-IPS method addresses both click incompleteness bias (using IPS) and click noise (using Bayes rule) for unbiased LTR.

We empirically evaluate our method on click logs from search services for Gmail, Google Drive, and Gmail Expert users. Our TrustPBM estimates obtained via the EM algorithm reveal novel insights into the nature of relevance noise present in real settings. Most interestingly, we observe that most of the noise comes from highly ranked irrelevant documents being judged relevant, while relevant documents tend to be judged accurately even at lower positions. Moreover, our results demonstrate that the TrustPBM fits the click data better than the plain PBM. Furthermore, ranking models trained with the Bayes-IPS correction have superior live experiment metrics than those trained with only the IPS correction based on PBM, estimated either via EM or result randomization.

In summary, the contributions of this paper are as follows:

- We propose TrustPBM, a novel click model that captures both position-dependent click incompleteness and click noise.
- We extend the EM algorithm designed for PBM to TrustPBM and show that it can effectively estimate the model parameters for TrustPBM.
- We present a novel Bayes-IPS correction method for unbiased LTR based on the parameters in TrustPBM estimated by the EM algorithm.
- We validate our approach on large-scale click logs. The experimental results show interesting patterns for user behavior and our proposed Bayes-IPS method achieves superior results compared to baselines.

The rest of the paper is organized as follows. In Section 2, we review previous related work. We describe our TrustPBM model in Section 3 and the EM algorithm for its parameter estimation in Section 4. Our Bayes-IPS approach for unbiased LTR is presented in Section 5. We report our evaluation on click logs in Section 6. Finally, we conclude this paper and discuss future work in Section 7.

2 RELATED WORK

Leveraging user interaction data such as clicks has been shown to be quite promising in improving search quality. A simple approach is to use click and non-click information as relevance judgment of results. This can be used for evaluation or to train a new ranking function in the LTR setting. However, such an approach can lead to misleading evaluation results or sub-optimal ranking functions due to various types of bias in click data, e.g., position bias [19], presentation bias [35], and trust bias [19, 25]. In the past, a large amount of research has been devoted to extracting more accurate signals from click data. For example, some heuristic methods have been proposed to address the position bias by utilizing the pairwise preferences between clicked and skipped documents (e.g., SkipAbove) [18–20]. Though these methods have been found to provide more accurate relevance assessments, their data is fundamentally biased. For example, a ranking function trained based on the SkipAbove heuristic tends to reverse the presented order due to its sampling bias [18].

Recently, unbiased LTR has been actively studied to learn a ranking function from click data based on the counterfactual inference framework [21]. With the assumption that feedback is noise-free, it provides an IPS-based correction method which is a provably unbiased and consistent approach to LTR even with biased feedback data. The IPS method requires knowledge of the propensity of obtaining a particular feedback signal. Click modeling, especially PBM, has been used for estimating propensity. Under this model, Wang et al. [32] proposed a result randomization method that randomly shuffles the top n results and uses the average CTR at each position as the propensity. Joachims et al. [21] proposed a pivot-based randomization where position 1 and k are randomly shuffled. The relative propensity between these two positions p_k/p_1 can be derived and this is sufficient for LTR. Wang et al. [33] proposed a RandPair approach where adjacent positions are randomly flipped to obtain p_{k+1}/p_k and a chain rule is used to estimate the ratio p_k/p_1. Wang et al. [32] extended the PBM model to let the propensity depend on query or user contexts, in addition to positions. All of these approaches require interventions.

The more recent development in unbiased LTR is on how to estimate propensity without interventions. Wang et al. [33] proposed a regression-based EM algorithm to fit the PBM model to the regular click logs without interventions. Similarly, Ai et al. [3] presented a framework to jointly learn an unbiased ranker and an unbiased propensity model from regular click data. Agarwal et al. [2] proposed a method for harvesting naturally randomized pairs from logs of multiple rankers in operational systems. All of these methods are based on the standard PBM model and do not address position-dependent click noise.

With the propensity estimated, the IPS-based approach is also used for search quality evaluations, in addition to learning a ranking function. For example, Carterette and Chandar [6] extend the counterfactual inference framework [21] by considering the case of evaluating new rankers that can retrieve previously unseen documents. Chandar and Carterette [7] studied the click-through bias in the cascade model for evaluation.

IPS was developed in causal inference [28] and is a widely accepted technique for handling sampling bias. It has been employed

in unbiased evaluation and learning [1, 11, 22, 23, 29, 30]. The common assumption in most of these studies is that the propensities are under the system's control and are thus known. In the unbiased LTR setting, however, propensities arise from user behavior and thus need to be estimated. While the IPS can be used to address the sample bias or click incompleteness, it is not straightforward for click noise.

Besides PBM, there are many other click models for extracting relevance signals from click data. Click models are parameterized generative models [9] and their parameters are typically estimated via generative maximum likelihood under specific modeling assumptions about user behavior. There are two classic click models: the PBM model [27] and the Cascade model [10]. Based on these two models, more advanced models have been developed. For example, the user browsing model (UBM) [12] extends PBM to condition the examination on a previously clicked position, in addition to the position of the current result. The dynamic Bayesian network model (DBN) [8] and the click chain model (CCM) [15] extend the Cascade model to handle multiple clicks. Both models assume a sequential user behavior over the result list. They focus on modeling the probability that document gets examined depending on the previous documents and differ in their modeling formulation. DBN and CCM are different from PBM and UBM in that the examination of a later document depends on the relevance of previous documents. However, most of these models assume clicks are noise-free. The focus of this paper is on click noise, specifically trust bias. We propose the TrustPBM model by extending the standard PBM model.

Though, in theory, the estimated examination probability in any click model can be used in unbiased LTR, the PBM model is more robust given that the examination does not depend on the relevance of previous documents since estimating relevance is as hard as learning a ranking function and thus leads to high variance of the estimated examination probability. Our TrustPBM model enjoys the same advantage where the trust bias depends on positions only.

The alternate approach of directly using position as a feature is generally not effective for learning a ranking function. This is because the position of a result is highly predictive of its click rate, however, the position feature cannot be used during inference time when a collection of documents must be scored to determine their positions in the presented ranking. Using the position feature in training and then unplugging it during inference leads to worse results [33].

3 TRUST ENHANCED POSITION-BASED MODEL

3.1 Position-Based Model

The Position-Based Model (PBM) is a simple generative model. It has been extensively used in unbiased LTR due to its simplicity. Most existing click models have the *examination hypothesis*. Suppose we have N positions, and for a query q, document d is displayed at position $k \in [1, N]$. A click can be generated only after the user examines the result. In the PBM model, the user clicks the document if and only if they examine the document and the document is relevant. The examination only depends on the position k, but not on q or d. Such a model factors out q and d, and makes the estimation of the probability of examination simpler and more robust, compared to other click models.

Formally, let C be a binary random variable signifying whether the user clicks document d, E whether the user examines the document, R for the true relevance and $\tilde{R}$ to represent the perceived or judged relevance. The PBM makes the following assumptions

$$C = 1 \Leftrightarrow E = 1, \tilde{R} = 1$$
$$R \Leftrightarrow \tilde{R}$$
$$\Pr(E = 1|q, d, k) = \Pr(E = 1|k)$$
$$\Pr(R|q, d, k) = \Pr(R|q, d)$$

Thus, the probability of examining a document only depends on the position k and the perceived relevance is the same as the true relevance. Based on these assumptions, the PBM has the following probability for a click.

$$\begin{aligned}\Pr(C = 1|q, d, k) &= \Pr(E = 1, \tilde{R} = 1|q, d, k) \\ &= \Pr(E = 1, R = 1|q, d, k) \\ &= \Pr(E = 1|k) \Pr(R = 1|q, d) \\ &= \theta_k \gamma_{q,d}\end{aligned}$$

where we use θ_k and $\gamma_{q,d}$ as short-hands for the probabilities.

3.2 Trust Bias

The PBM model and its extensions [32] have a noise-free assumption that perceived relevance is the same as the true relevance. This is not the case in general. How to model the relation between perceived relevance and true relevance is relatively less studied, even in the click models. DBN [8] is one of the few models that incorporates this relation. In DBN, a set of document-dependent parameters α_d are introduced to model the relation between perceived relevance and true relevance:

$$\Pr(R = 1|q, d) = \alpha_d \Pr(\tilde{R} = 1|q, d)$$

where a parameter α_d depend on the document d. The intuition is that the probability of true relevance is a discounted version of perceived relevance and that the discount is document-dependent. As a result, the model has a large number of additional parameters that equals to the number of unique documents in the data.

In this paper, we model the relation between perceived relevance and true relevance based on trust bias introduced in [19]. In this user study, eye tracking was used to monitor examination. The study showed that, after examination, users tend to trust the results presented higher on the result page more, demonstrated by higher click ratio despite the fact that the documents are both examined and their expected relevance is the same. This suggests a position-dependent trust bias in addition to examination bias, due to which higher ranked non-relevant documents may be clicked and lower ranked relevant documents not clicked. However, this has not been mathematically formulated in the past. In this paper, we model the trust bias as follows,

$$\Pr(\tilde{R} = 1|R = 1, E = 1, k) = \epsilon_k^+$$
$$\Pr(\tilde{R} = 1|R = 0, E = 1, k) = \epsilon_k^-$$

Since $C = 1 \Leftrightarrow E = 1, \tilde{R} = 1$, we have $\Pr(C = 1|R = 1, E = 1, k) = \epsilon_k^+$ and $\Pr(C = 1|R = 0, E = 1, k) = \epsilon_k^-$. Intuitively, after

examination, a relevant document at position k can be missed with probability $1-\epsilon_k^+$, but a non-relevant document can be clicked mistakenly with probability ϵ_k^-. Notice that the noise is position-dependent. A similar formulation has been studied in [21] where the noise is constant across positions.

3.3 TrustPBM

We enhance the PBM model by explicitly modeling the trust bias introduced above. We have

$$\begin{aligned}&\Pr(\tilde{R}=1|E=1,q,d,k)\\ =&\Pr(\tilde{R}=1|R=1,E=1,k)\Pr(R=1|q,d)\\ +&\Pr(\tilde{R}=1|R=0,E=1,k)\Pr(R=0|q,d)\\ =&\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d})\end{aligned}$$

Thus, in TrustPBM, the probability of a click is given as

$$\begin{aligned}\Pr(C=1|q,d,k)&=\Pr(E=1|k)\Pr(\tilde{R}=1|E=1,q,d,k)\\ &=\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))\end{aligned}$$

PBM can be viewed as a special case of TrustPBM, where $\epsilon_k^+=1$ and $\epsilon_k^-=0$ to reflect the noise-free assumption. Both PBM and TrustPBM model examination bias in the same way. On the other hand, TrustPBM is different from DBN. The noise depends on position only, not on the document. Compared to PBM, we introduce $2N$ more parameters into the model where N is usually small (e.g., 10), so the model complexity of TrustPBM is much smaller than DBN.

The TrustPBM parameters can be estimated using logged click data. Let $\mathcal{L}$ be the logged data consisting of tuples (q,d,k,c) for each received user query q, each document d displayed at any position k and a binary indicator c for whether the document was clicked or not (i.e. the realized value of the random variable C). Then, the log-likelihood of generating $\mathcal{L}$ (over the randomness in user click behavior) is,

$$\begin{aligned}\log\Pr(\mathcal{L})=&\sum_{(q,d,k,c)\in\mathcal{L}}c\log\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))\\ &+(1-c)\log(1-\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))).\end{aligned}$$

In order to find the parameters that maximize the above log-likelihood objective, we use Expectation-Maximization, extending the Regression-based EM method for the plain PBM in [33].

4 ESTIMATION VIA EXPECTATION-MAXIMIZATION

As we will show in Section 5, we only need the parameters θ_k, ϵ_k^+, and ϵ_k^- for unbiased LTR. However, unlike the standard PBM model whose parameters can be estimated based on result randomization, TrustPBM does not allow estimation of these parameters using a randomized data set. In this paper, we extend the previously proposed regression-based EM algorithm [33] to estimate the parameters.

In the standard EM procedure, parameters are estimated by alternating Expectation and Maximization steps until convergence. Suppose $\{\theta_k^{(t)},\epsilon_k^{+(t)},\epsilon_k^{-(t)}:k\in[N]\}$ and $\{\gamma_{q,d}^{(t)}:q,d\in\mathcal{L}\}$ are the parameter estimates at iteration t (initialized appropriately at the first iteration). We will get a new set of parameter estimates after iteration $t+1$.

4.1 Expectation Step

In the Expectation step at iteration $t+1$, the posterior distribution of the hidden variables of examination E and true relevance R given the observed click data $\mathcal{L}$ is expressed in terms of the parameter estimates at iteration t, as shown in Figure 1 (we omit superscript (t) for readability). All the formulas follow directly from Bayes rules. For instance, in the third from last equation, we have

$$\begin{aligned}&\Pr(E=0,R=1|q,d,k,c=0)\\ =&\frac{\Pr(C=0|E=0,R=1,q,d,k)\Pr(E=0,R=1|q,d,k)}{\Pr(C=0|q,d,k)}\\ =&\frac{\Pr(E=0,R=1|q,d,k)}{\Pr(C=0|q,d,k)}\\ =&\frac{\Pr(E=0|k)\Pr(R=1|q,d)}{\Pr(C=0|q,d,k)}\end{aligned}$$

since $\Pr(C=0|E=0,R=1,q,d,k)=1$, and examination and relevance are independent events. Also, a click means that the document must have been examined and thus $\Pr(E=0,R=1|q,d,k,c=1)=0$ in the second equation.

4.2 Maximization Step

In the Maximization step at iteration $t+1$, the parameters are updated to their maximum likelihood values given the click data and the posterior probabilities from the Expectation step as shown in Figure 2. Once again, we drop the superscript $(t+1)$ for clarity in these formulas. Different from PBM model, we have additional ϵ_k^+ and ϵ_k^- parameters updated in this step.

Regression-based EM. For the M-step to work with query-document pairs (q,d), a requirement is that (q,d) should repeat and also be shown in different positions. Since the query-document pairs (q,d) vary arbitrarily, using free parameters $\gamma_{q,d}$ makes the objective very sparse and difficult to learn. Moreover, exposing individual (q,d) identifiers can also pose a privacy or security risk. Therefore, as in [33], we propose to estimate the $\gamma_{q,d}$ parameters in each Maximization step via regression, i.e. fit a function (such as a Gradient Boosted Decision Tree) from features of (q,d) to the derived target value for $P(R=1|c,q,d,k)$ at iteration $t+1$.

More specifically, we assume that there is a feature vector $\mathbf{x}_{q,d}$ representing them and use a function to compute the relevance $\gamma_{q,d}=f(\mathbf{x}_{q,d})$. The Maximization step is then to find a regression function $f(\mathbf{x})$ to maximize the likelihood given the estimation from the Expectation step. Specifically, for each $(q,d,k,c)\in\mathcal{L}$, the expectation step gives a probability $P(R=1|q,d,k,c)$. Intuitively, we can regress the feature vector $\mathbf{x}_{q,d}$ to the probability $P(R=1|c,q,d,k)$. Similar to [33], we convert such a regression problem to a classification problem based on sampling: we sample a binary relevance label $r\in\{0,1\}$ according to $P(R=1|c,q,d,k)$. This conversion allows us to use widely available classification tools to solve our problem. After sampling, we have a training set $\{(\mathbf{x},r)\}$

$$Pr(E=1,R=1|q,d,k,c=1)=\frac{\epsilon_k^+\gamma_{q,d}}{\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d})},\qquad \Pr(E=0,R=1|q,d,k,c=1)=0$$

$$Pr(E=1,R=0|q,d,k,c=1)=\frac{\epsilon_k^-(1-\gamma_{q,d})}{\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d})},\qquad \Pr(E=0,R=0|q,d,k,c=1)=0$$

$$Pr(E=1,R=1|q,d,k,c=0)=\frac{\theta_k(1-\epsilon_k^+)\gamma_{q,d}}{1-\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))},\qquad \Pr(E=0,R=1|q,d,k,c=0)=\frac{(1-\theta_k)\gamma_{q,d}}{1-\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))}$$

$$\Pr(E=1,R=0|q,d,k,c=0)=\frac{\theta_k(1-\epsilon_k^-)(1-\gamma_{q,d})}{1-\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))},\qquad \Pr(E=0,R=0|q,d,k,c=0)=\frac{(1-\theta_k)(1-\gamma_{q,d})}{1-\theta_k(\epsilon_k^+\gamma_{q,d}+\epsilon_k^-(1-\gamma_{q,d}))}$$

Figure 1: Formulas for computing probabilities of hidden variables in the E-step of the EM algorithm for TrustPBM.

$$\theta_k=\frac{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}(c+(1-c)P(E=1|q,d,k,C=0))}{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}}$$

$$\gamma_{q,d}=\frac{\sum_{(q',d',k,c)\in\mathcal{L}}1\{q'=q,d'=d\}(c\cdot P(R=1|q,d,k,C=1)+(1-c)P(R=1|q,d,k,C=0))}{\sum_{(q',d',k,c)\in\mathcal{L}}1\{q'=q,d'=d\}}$$

$$\epsilon_k^+=\frac{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}\cdot c\cdot P(E=1,R=1|,q,d,k,C=1)}{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}(c\cdot P(E=1,R=1|q,d,k,C=1)+(1-c)P(E=1,R=1|q,d,k,C=0))}$$

$$\epsilon_k^-=\frac{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}\cdot c\cdot P(E=1,R=0|,q,d,k,C=1)}{\sum_{(q,d,k',c)\in\mathcal{L}}1\{k'=k\}(c\cdot P(E=1,R=0|q,d,k,C=1)+(1-c)P(E=1,R=0|q,d,k,C=0))}$$

Figure 2: Formulas for updating parameters in the M-step of the EM algorithm for TrustPBM, where 1 is the indicator function.

for $f(\mathbf{x})$. The objective function is the log likelihood:

$$\sum_{\{(\mathbf{x},r)\}} r\log(f(\mathbf{x}))+(1-r)\log(1-f(\mathbf{x})),$$

where we use the sigmoid in the objective function

$$f(\mathbf{x})=\frac{1}{1+e^{-F(\mathbf{x})}}$$

$F(x)$ is the log odd of function $f(x)$ and is learned by Gradient Boosted Decision Tree (GBDT) method [13] in this paper.

5 UNBIASED LTR VIA BAYES-IPS CORRECTION

Now we turn to the question of using the parameter estimates of the TrustPBM to learn a new ranking function from the click data in an unbiased way. Specifically, we need to adjust each click signal in the logged data such that it reflects the actual underlying relevance signal in expectation, unconfounded by the inherent examination and trust biases captured by TrustPBM.

In existing work [21, 33] which deals with the standard PBM, this is achieved via Inverse Propensity Scoring (IPS) which corrects for the examination bias factor. Since TrustPBM does not factorize into purely bias (from examination and trust) and relevance terms, a direct IPS correction is not feasible. In the following, we first review the existing unbiased LTR and then present our two-step approach to deal with noise: correct for examination bias via IPS, and then de-noise for trust bias via Bayes law.

5.1 LTR

We begin with a brief overview of the LTR setting. In the standard LTR, a set of labeled data for query and documents are given $\mathcal{L}_U=\{(q,d,r)\}$ where r represents the relevance labels for the (q,d) pair. We also use $\Omega_q=\{d|(q,d)\in\mathcal{L}_U\}$ to represent all documents that are associated with query q. Collection $\mathcal{L}_U$ is *unbiased* and a uniform sample from the space of queries, documents, and relevance. In general, fully supervised LTR algorithms such as LambdaMART and RankingSVM are used to directly or indirectly optimize a performance metric which decomposes as a sum over the (q,d) samples in the training set, as such

$$M=\sum_{(q,d,r)\in\mathcal{L}_U} r\cdot f(q,d,\Omega_q)$$

where f is usually a rank-related function. For instance, Average Relevance Position (ARP) [21, 33, 34] and Discounted Cumulative Gain (DCG) like metrics [17, 33] are shown as below,

$$ARP=\sum_{(q,d,r)\in\mathcal{L}_U} r\,\mathrm{rank}(q,d,\Omega_q)$$

$$DCG=\sum_{(q,d,r)\in\mathcal{L}_U} r\,\frac{1}{\mathrm{rank}(q,d,\Omega_q)}.$$

Note that lower *ACP* and higher *DCG* indicate better ranking performance with more relevant documents at higher ranks. We use a simple DCG definition that uses a binary relevance and a linear function for position discount to be consistent with the rest of paper. But the methodology is applicable to any DCG definitions.

In pairwise learning-to-rank algorithms, scores s are learned for (q, d) pairs by expressing rank as a pairwise objective

$$\begin{aligned} ARP &= \sum_{(q,d,r)\in\mathcal{L}_U} r \,\mathrm{rank}(q, d, \Omega_q) \\ &= \sum_{(q,d,r)\in\mathcal{L}_U} r \,(\sum_{d'\in\Omega_q} \mathbb{I}_{s_{(q,d)}<s_{(q,d')}} + 1) \\ &= \sum_{(q,d,r)\in\mathcal{L}_U} \sum_{d'\in\Omega_q} r\, \mathbb{I}_{s_{(q,d)}<s_{(q,d')}} + const \end{aligned} \tag{1}$$

The loss above can then be upper-bounded by common classification loss functions (e.g., the hinge loss or the logistic loss [16]) as in the RankingSVM method. Moreover, rank-based metrics such as *ARP* and *DCG* can also be optimized in the LambdaLoss framework [34] where LambdaRank and LambdaMART [5] are special cases in the framework.

5.2 Unbiased LTR

The challenge in learning from implicit feedback such as click data is that only the click information c is observed, not the true relevance r for each (q, d) pair. The relation between r and c is captured by click models. The Inverse Propensity Scoring (IPS) technique is used to tackle this issue by de-biasing click data because the examination bias in PBM is equivalent to the propensity of observing relevance labels in the counterfactual framework [21].

Recall that we have logged data $\mathcal{L}$ consisting of tuples (q, d, k, c),

$$\mathbb{E}[C|q, d, k] = \Pr(C = 1|q, d, k) = \theta_k \Pr(\tilde{R} = 1|E = 1, q, d, k)$$

where the expectation is over the randomness in the user click behavior. As in previous work, we can employ IPS to correct for the examination bias in the following way,

$$\begin{aligned} M_{IPS} &= \sum_{(q,d,k,c)\in\mathcal{L}} \frac{c}{\theta_k} f(q, d, \Omega_q) \\ &= \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} f(q, d, \Omega_q) \end{aligned}$$

M_{IPS} is an unbiased estimation of metric M because

$$\begin{aligned} \mathbb{E}[M_{IPS}] &= \sum_{(q,d,k,c)\in\mathcal{L}} \frac{\mathbb{E}[C]}{\theta_k} f(q, d, \Omega_q) \\ &= \sum_{(q,d,k,c)\in\mathcal{L}} \Pr(\tilde{R} = 1|E = 1, q, d, k) f(q, d, \Omega_q) \end{aligned}$$

In the plain PBM which assumes that $\Pr(\tilde{R} = 1|E = 1, q, d, k) = \Pr(R = 1|q, d)$, i.e. judgment upon examination reflects actual relevance free from noise, interpreting r as $\Pr(R = 1|q, d)$ (the degree of true relevance), M_{IPS} equals M in expectation and thus can be used as a direct unbiased substitute in the learning-to-rank algorithm since it is simply a linear weight modification.

$$\begin{aligned} ARP_{IPS} &= \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} \,\mathrm{rank}(q, d, \Omega_q) \\ DCG_{IPS} &= \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} \frac{1}{\mathrm{rank}(q, d, \Omega_q)}. \end{aligned}$$

For instance, the upper-bounding technique shown in Eq 1 still applies with ARP_{IPS} and an IPS-based RankingSVM was derived in [21] and the LambdaMART algorithm can be used to optimize DCG_{IPS}.

5.3 Bayes-IPS Correction

Now, for the TrustPBM, we do not have the equivalence between perceived relevance $\tilde{R}$ and R. The standard IPS approach gives an unbiased estimation of $\tilde{R}$. Note that we only care about the clicked documents in M_{IPS}. To get the estimate of R, we only need to estimate it for $C = 1$, i.e., $\Pr(R = 1|C = 1, q, d, k)$. Such an estimation can involve many parameters due to the dependency on (q, d) pairs. To make it tractable, we thus approximate it by the average relevance at position k: $\Pr(R = 1|C = 1, k)$, i.e., $\Pr(R = 1|\tilde{R} = 1, E = 1, k)$. By Bayes law, we have

$$\begin{aligned} &\Pr(R = 1|\tilde{R} = 1, E = 1, k) \\ &= \frac{\Pr(\tilde{R} = 1|R = 1, E = 1, k) \Pr(R = 1|E = 1, k)}{\Pr(\tilde{R} = 1|E = 1, k)} \\ &= \frac{\epsilon_k^+ \Pr(R = 1|E = 1, k)}{\epsilon_k^+ \Pr(R = 1|E = 1, k) + \epsilon_k^- \Pr(R = 0|E = 1, k)} \\ &= \frac{\epsilon_k^+ \Pr(R = 1)}{\epsilon_k^+ \Pr(R = 1) + \epsilon_k^- \Pr(R = 0)} \\ &= \frac{\epsilon_k^+}{\epsilon_k^+ + \epsilon_k^-} \end{aligned}$$

The first step is based on Bayes law. The second is the definition of ϵ_k^+ and ϵ_k^-. The third step is because R is independent of E and k. The last step is based on a weak uninformative *prior* on actual relevance that $\Pr(R = 1) = \Pr(R = 0) = 0.5$. Putting it all together, we have

$$\begin{aligned} M_{Bayes-IPS} &= \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} \Pr(R = 1|C = 1, k) f(q, d, \Omega_q) \\ &= \sum_{(q,d,k,c=1)\in\mathcal{L}} \frac{1}{\theta_k} \frac{\epsilon_k^+}{\epsilon_k^+ + \epsilon_k^-} f(q, d, \Omega_q). \end{aligned}$$

Informally, the Bayes-IPS estimate corrects for the examination bias and the likelihood that the result was actually relevant given that it was judged so upon examination. As a sanity check, note that in the noise-free case of the plain PBM, i.e. when $\epsilon_k^+ = 1$ and $\epsilon_k^- = 0$, $M_{Bayes-IPS}$ reduces to M_{IPS}. Moreover, since the proposed correction is simply a change of weights, a learning-to-rank framework with IPS can be easily adapted to use Bayes-IPS weights to replace IPS weights.

6 EMPIRICAL EVALUATION

In this section, we evaluate our proposed TrustPBM click model and the Bayes-IPS correction for unbiased LTR. We first describe our experiment setup and then present our experimental results.

6.1 Experiment Setup

Our experiments are based on both offline evaluation and online experiments. For offline evaluation, we use the standard framework for supervised LTR evaluation [24] by splitting the data into training and test sets. For online experiments, we evaluate our proposed methods through online A/B testing. In this section, we describe

Table 1: Statistics of the three data sets used in offline evaluation.

	Email	File Storage	Email Expert
#Queries per user	> 3	> 3	< 4
#Docs in total	59.69M	50.77M	13.80M
#Docs per query	5.99	5.00	6.00

the data sets and metrics used in offline evaluation as well as the metrics used in online experiments.

6.1.1 Data Sets. For offline evaluation, we use search logs collected from three search services: Gmail for general users (denoted as **Email**), Google Drive for general users (denoted as **File Storage**), and Gmail for expert users (denoted as **Email Expert**). The Email and Email Expert services have the same type of contents, but the users in the Email Expert dataset are more active, as measured by their search activity. The File Storage service has a different type of content compared with the other two. In all services, each query can result in at most a single click of a document because an overlay showing results is displayed as a query is typed and it disappears as a result is clicked. The clicked document is then shown on the screen.

In our data sets, we discard all the queries that did not result in any clicks. No further pre-processing is done. In all these services, ranking features are routinely computed and logged. We collected a sample from the search logs of each service in a two-week period in May 2018. Data from the first week are used for training, and those from the second week are used for testing. Basic statistics for the data sets are shown in Table 1. Each data set contains millions of queries, and between 5 and 6 documents per query. The logs also contain existing ranking features for each query-document pair, which we use in our regression-based EM algorithms and unbiased LTR algorithms.

6.1.2 Offline evaluation metrics. Our offline evaluation consists of two parts. The first part is to compare the PBM and TrustPBM models based on their fitness to the data. For this part, we use the standard log-likelihood on how the models predict clicks in our test data.

$$\text{LogLikelihood} = \frac{1}{|\mathcal{L}|} \sum_{\mathcal{L}} c \log(p) + (1 - c) \log(1 - p) \tag{2}$$

where $p = \theta_k \gamma_{q,d}$ for PBM and $p = \theta_k(\epsilon_k^+ \gamma_{q,d} + \epsilon_k^-(1 - \gamma_{q,d}))$ for TrustPBM. Log-likelihood is commonly used to evaluate click models [9]. It is a negative number and the larger the better.

For the second part, we compared different methods on the effectiveness of learned ranking functions for unbiased LTR. Following [32, 33], we use the weighted version of Mean Reciprocal Rank (MRR) as the evaluation metric for offline experiment. It is an unbiased version of MRR and more suitable than standard MRR for offline evaluation given click bias. Specifically, given a test data set with n queries, the standard MRR is defined as:

$$\text{MRR} = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{\text{rank}_i} \tag{3}$$

where rank_i denotes the rank of the first clicked document of the i-th query in our data sets. Note that since each query can have at most one clicked document, we do not consider more complicated metrics like normalized discounted cumulative gain. We denote weighted MRR [33] by MRR_w, which is defined as:

$$\text{MRR}_w = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i \frac{1}{\text{rank}_i} \tag{4}$$

where w_i is the IPS correction weight estimated from our algorithm. Given different propensity estimation methods, the weights are used for both training and offline evaluation.

Table 2: Comparison of PBM and TrustPBM click models based on log-likelihood. The best number per row is in bold face and + means statistically significance based on the student t-test.

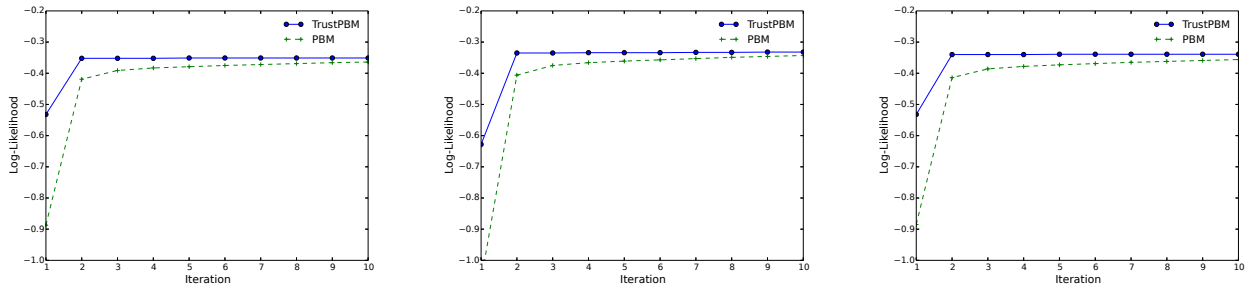
	PBM	TrustPBM	Improvement
Email	-0.364	**-0.351**	$3.57\%^+$
File Storage	-0.343	**-0.332**	$3.21\%^+$
Email Expert	-0.356	**-0.339**	$4.78\%^+$

6.1.3 Online experiment metrics. For online experiments, we run A/B testing using ranking functions learned based on different IPS weights. This is the ultimate test of effectiveness of ranking functions. A better ranking function can attract more clicks from users and push the user clicks to higher positions. We thus use the click-through rate (CTR) that is defined as the ratio of queries with clicks and the total number of queries. We also use the standard MRR in Eq 3 for online evaluation to take into account the effect of positions. For any queries without clicks, their MRR values are set to 0 for online experiments.

6.2 Experimental Results

In this section, we report both offline and online experimental results. Our comparisons are conducted for both click modeling and unbiased LTR.

6.2.1 Does the extended model fit the data better? Both PBM and TrustPBM are click models. Our first result is to compare them on how well they can fit our click data. We run our EM algorithms for both PBM and TrustPBM respectively and compare them in terms of log-likelihood in Eq 2. The results on our test data sets are shown in Table 2. From this table, we can see that the TrustPBM model has higher log-likelihood values than the PBM model on all the three data sets. We also report the relative improvement of the log-likelihood in this table. All these improvements are statistically significant based on the student t-test. This means that our proposed method of incorporating click noise in TrustPBM is reasonable since TrustPBM can indeed fit the data significantly better than PBM. In addition, we also study the convergence of the EM algorithms for PBM and TrustPBM in Figure 3, where we plot log-likelihood along with the EM iterations. It can be observed that TrustPBM not only converges to a higher log-likelihood, but also converges faster than PBM at early stages. This confirms that our proposed EM algorithm can estimate parameters for TrustPBM effectively. Overall, this result shows that TrustPBM is better that PBM.



(a) Email (b) File Storage (c) Email Expert

Figure 3: Convergence of EM based on log-likelihood.

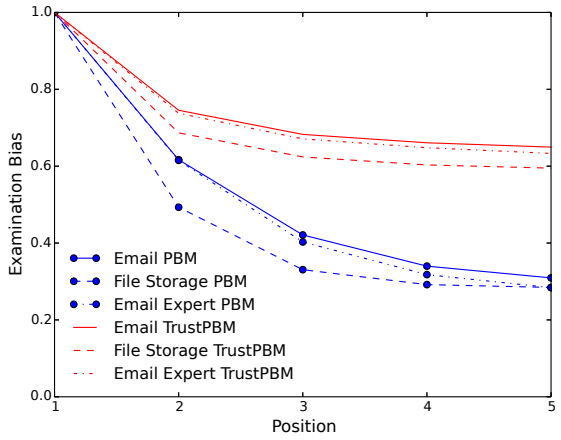


Figure 4: The normalized examination bias estimated from PBM and TrustPBM using EM.

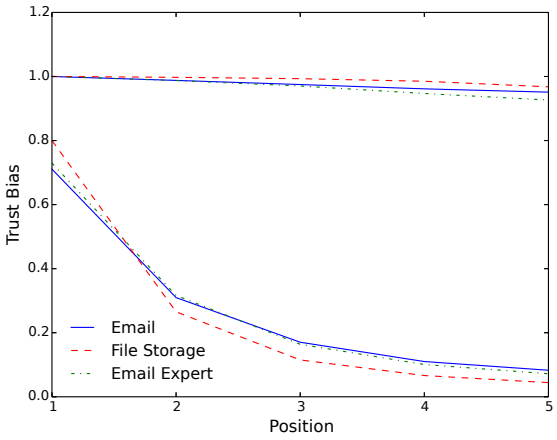


Figure 5: The normalized trust bias estimated for TrustPBM using EM. The upper lines are ϵ_k^+ and the lower lines are ϵ_k^-.

6.2.2 What are the biases estimated from data? We have examination bias for both PBM and TrustPBM. For TrustPBM, we also have the trust bias ϵ_k^+ and ϵ_k^-. We compare these estimated biases from data and show some interesting findings.

The examination biases for PBM and TrustPBM are plotted in Figure 4, where we normalize the bias values by the one at position 1. This figure shows that PBM and TrustPBM give quite different estimation of examination bias, especially for lower ranked documents. TrustPBM shows that there are higher probabilities that users examine lower ranked documents than PBM. This aligns better with the user study in [19] as the user study showed that users examine the surrounding documents before issuing clicks, especially the ones immediately below their clicked ones.

The trust bias estimated by TrustPBM is shown in Figure 5 where we normalize all of them by the ϵ_k^+ for $k = 1$. There are a few interesting observations. (1) All ϵ_k^+ are higher than ϵ_k^-. This means that the probability of clicks given relevant documents is higher than the probability given non-relevant documents after examination. Thus clicks represent reasonable relevance judgments even though they are noisy. (2) ϵ_k^+ are very close to 1 over all positions k, and the difference over positions is very small, showing that relevant documents are not missed often once examined. (3) In contrast, ϵ_k^- varies a lot over positions k. It is much higher at the higher positions (lower values of k) than lower positions, meaning that users are more inclined to click non-relevant documents shown at high positions. This is congruent with the trust bias user study [19], which found that users may place more trust in highly-ranked results. (4) Across different data sets, the trust bias is different. For example, the File Storage data set has a steeper curve for ϵ_k^-, meaning that there is higher trust bias for this service. Overall, this result shows that TrustPBM is better that PBM qualitatively.

6.2.3 Is the Bayes-IPS correction different? We now turn to the unbiased LTR evaluation. The inverse propensity weights estimated by different methods in the Email data set are shown in Figure 7. For result randomization, we use the FairPair approach in [33], that

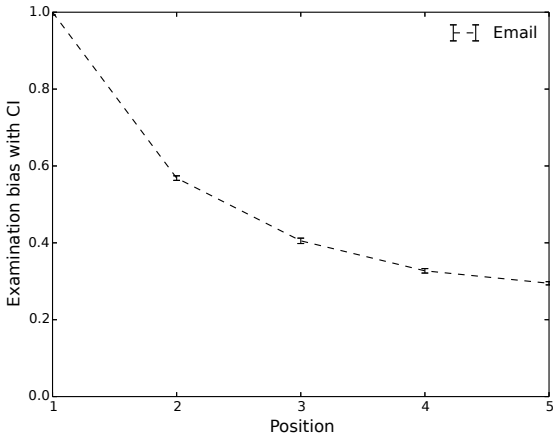


Figure 6: The examination bias and the confidence interval (CI) based on randomization data.

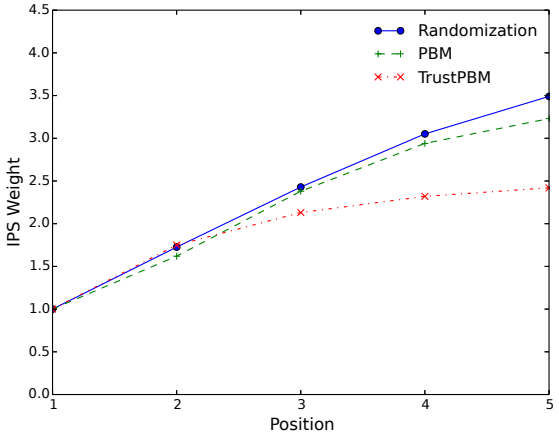


Figure 7: The IPS or Bayes-IPS weights based on Randomization (IPS), PBM (IPS) and TrustPBM (Bayes-IPS).

is, two adjacent documents are randomly swapped before presenting results to users. This method has been shown to be aligned well with other randomization methods for the PBM model. We use Randomization to denote the IPS weights estimated from this method. We use PBM to denote the IPS weights estimated by EM in PBM, and we TrustPBM to denote the Bayes-IPS weights estimated by EM in TrustPBM. Compared with TrustPBM, the EM-based PBM is much closer to Randomization. This result is expected since Randomization is also based on PBM. However, TrustPBM gives much smaller estimation of weights for lower positions. This is because TrustPBM uses a different model that disentangles examination and trust bias, leading to rather different estimation of inverse propensity weights.

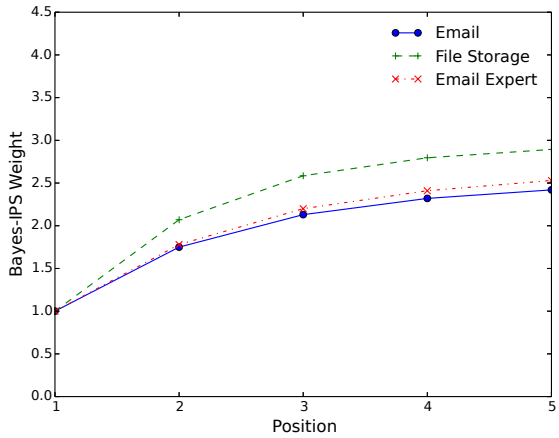


Figure 8: The Bayes-IPS weights estimated for different data sets.

Table 3: Cross offline evaluation to compare IPS weights for training and evaluation.

Training	Evaluation: Weighted MRR in Eq 4			
	NoIPS	Randomization	PBM	TrustPBM
NoIPS	**0.00%**	-0.50%	-0.36%	-0.20%
Randomization	-1.11%	**0.00%**	0.00%	-0.11%
PBM	-0.89%	-0.02%	**0.00%**	-0.06%
TrustPBM	-0.48%	-0.11%	-0.05%	**0.00%**

The lower IPS weights at lower positions in TrustPBM is more desired. As commonly known, IPS-based approaches usually suffer from the problem of the unbounded variance since the variance for a smaller propensity (corresponding to the lower positions in unbiased LTR) can lead to a large difference in IPS weights. Several techniques such as capping methods are developed to address this problem [4, 11, 14, 26, 31]. For Randomization, we compute the variance of our estimated propensity θ_k as follows. We split our randomized data set into 10 folds and use them to estimate the confidence interval (CI). The results are show in Figure 6. From this figure, we see that there is a rather negligible variance based on CI. This means that we have sufficient randomized data for IPS weights estimation. As we will show later that TrustPBM leads to better online experiment results, the result here indicates that PBM is a less accurate model than TrustPBM for extracting relevance from user click behaviors.

In addition, Figure 8 shows the Bayes-IPS weights estimated for different data sets. The weights being different for different data sets means that our method can indeed take click noise into account that varies across data sets.

6.2.4 Can the learning algorithms leverage different IPS weights? We have shown that different methods give different IPS weights. These IPS weights can be used for both evaluation and learning for unbiased LTR. In this sense, we do not have ground-truth IPS

Table 4: Online experiment results based on LambdaMART models learned using different weights. Relative improvement over Randomization is reported, together with the confidence interval (CI). The best number per column is bold and + means the improvement is significant based on student t-test.

	MRR	MRR CI	CTR	CTR CI
Randomization	0.00%	-	0.00%	-
PBM	0.25%	±0.39%	0.26%	±0.40%
TrustPBM	**0.33%**$^+$	±0.29%	**0.33%**$^+$	±0.28%

weights to compare different methods. We rely on the online experiments to do the comparison. Before doing so, we need to make sure that our learning algorithms can effectively leverage the IPS weights during training.

We use an in-house implementation of LambdaMART as our learning algorithm. Table 3 shows the cross comparison using different IPS weights for both training and evaluation. This comparison serves as a sanity check as to whether our training methods can take into account different settings of weights. All the values in Table 3 are relative numbers comparing to the diagonal lines per column. The diagonal values are expected to be the largest numbers per column since both training and evaluation use the same weights. The results in this table confirm that. For example, using the Bayes-IPS weights in TrustPBM, all models trained with different weights show negative relative evaluation numbers. Thus our learning algorithms are able to differentiate different weights to optimize their intended metrics.

6.2.5 Results of online experiments. We compare Randomization, PBM and TrustPBM using the live traffic of the Email service, conducting an online experiment for each method on a portion of the search traffic. These experiments were run for several weeks in August 2018 and our comparisons are based on approximately 100M user queries per experiment. For clarity, we normalize the online evaluation metrics using the Randomization method as the baseline.

We compute online metrics MRR and CTR; the results are shown in Table 4. In this table, we can see that TrustPBM improves over the Randomization method significantly, with MRR and CTR both increased by +0.33%, and these results are statistically significant. In contrast, PBM achieves neutral results compared with the Randomization baseline. Moreover, TrustPBM is more robust than PBM because it has a much smaller confidence interval (CI). This is probably due to the fact that having higher weights on lower positions can lead to higher variance when learning a ranking function for the PBM model. By addressing click noise, TrustPBM is more robust.

7 CONCLUSION

In this paper, we proposed to handle click noise for unbiased LTR. We have introduced an enhanced position bias model called TrustPBM which explicitly models position-dependent click noise, conceptualized as a result of the trust users have in the quality of the search ranking. The parameters of the TrustPBM are estimated from offline click logs via regression-based Expectation Maximization, and the novel Bayes-IPS correction provides a simple weighting scheme for unbiased, de-noised LTR using the estimated parameters. Experiments on three widely used commercial search services revealed new insights about the asymmetry in positive and negative trust bias. Furthermore, the Bayes-IPS correction yielded superior performance metrics over the IPS correction based on the vanilla PBM estimated via EM or randomization.

Future work can include alternate methods for TrustPBM estimation, such as via intervention harvesting [2] which requires click logs from multiple rankers but avoids relevance modeling, and alternate schemes for unbiased LTR that weaken the prior assumption in our Bayes-IPS correction.

REFERENCES

[1] Aman Agarwal, Soumya Basu, Tobias Schnabel, and Thorsten Joachims. 2017. Effective Evaluation using Logged Bandit Feedback from Multiple Loggers. In *Proc. of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 687–696.

[2] Aman Agarwal, Ivan Zaitsev, and Thorsten Joachims. 2018. Consistent Position Bias Estimation without Online Interventions for Learning-to-Rank. (2018). arXiv:cs.LG/1806.03555

[3] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W. Bruce Croft. 2018. Unbiased Learning to Rank with Unbiased Propensity Estimation. In *Proc. of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR)*. 385–394.

[4] Léon Bottou, Jonas Peters, Joaquin Qui nonero Candela, Denis X. Charles, D. Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. 2013. Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising. *Journal of Machine Learning Research* 14 (2013), 3207–3260.

[5] Chris J.C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview.* Technical Report MSR-TR-2010-82. Microsoft Research.

[6] Ben Carterette and Praveen Chandar. 2018. Offline Comparative Evaluation with Incremental, Minimally-Invasive Online Feedback. In *Proc. of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR)*. 705–714.

[7] Praveen Chandar and Ben Carterette. 2018. Estimating Clickthrough Bias in the Cascade Model. In *Proc. of the 27th ACM International Conference on Information and Knowledge Management (CIKM)*. 1587–1590.

[8] Olivier Chapelle and Ya Zhang. 2009. A dynamic Bayesian network click model for web search ranking. In *Proceedings of the 18th International Conference on World Wide Web (WWW)*. 1–10.

[9] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. Click Models for Web Search. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 7, 3 (2015), 1–115.

[10] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An Experimental Comparison of Click Position-bias Models. In *Proc. of the 1st International Conference on Web Search and Data Mining (WSDM)*. 87–94.

[11] Miroslav Dudík, John Langford, and Lihong Li. 2011. Doubly Robust Policy Evaluation and Learning. In *Proc. of the 28th International Conference on Machine Learning (ICML)*. 1097–1104.

[12] Georges E. Dupret and Benjamin Piwowarski. 2008. A User Browsing Model to Predict Search Engine Click Data from Past Observations. In *Proc. of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 331–338.

[13] Jerome H. Friedman. 2000. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics* 29 (2000), 1189–1232.

[14] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline A/B Testing for Recommender Systems. In *Proc. of the 11th ACM International Conference on Web Search and Data Mining (WSDM)*. 198–206.

[15] Fan Guo, Chao Liu, Anitha Kannan, Tom Minka, Michael Taylor, Yi-Min Wang, and Christos Faloutsos. 2009. Click Chain Model in Web Search. In *Proc. of the 18th International Conference on World Wide Web (WWW)*. 11–20.

[16] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning.* Springer.

[17] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[18] Thorsten Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. In *Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 133–142.

[19] Thorsten Joachims, Laura A. Granka, Bing Pan, Helene Hembrooke, and Geri

Gay. 2005. Accurately Interpreting Clickthrough Data as Implicit Feedback. In *Proc. of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 154–161.

[20] Thorsten Joachims, Laura A. Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. 2007. Evaluating the Accuracy of Implicit Feedback from Clicks and Query Reformulations in Web Search. *ACM Transactions on Information Systems (TOIS)* 25, 2 (April 2007).

[21] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Proc. of the 10th ACM International Conference on Web Search and Data Mining (WSDM)*. 781–789.

[22] Lihong Li, Shunbao Chen, Jim Kleban, and Ankur Gupta. 2014. Counterfactual Estimation and Optimization of Click Metrics for Search Engines. (2014). arXiv:cs.LG/1403.1891

[23] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased Offline Evaluation of Contextual-bandit-based News Article Recommendation Algorithms. In *Proc. of the 4th International Conference on Web Search and Web Data Mining (WSDM)*. 297–306.

[24] Tie-Yan Liu. 2009. Learning to Rank for Information Retrieval. *Foundations and Trends in Information Retrieval* 3, 3 (March 2009), 225–331.

[25] Maeve O'Brien and Mark T. Keane. 2007. Modeling User Behavior Using a Search-Engine. In *Proc. of the 12th International Conference on Intelligent User Interfaces (IUI)*. 357–360.

[26] M. J. D. Powell and J. Swann. 1966. Weighted Uniform Sampling — a Monte Carlo Technique for Reducing Variance. *IMA Journal of Applied Mathematics* 2, 3 (1966), 228–236.

[27] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting Clicks: Estimating the Click-through Rate for New Ads. In *Proc. of the 16th International Conference on World Wide Web (WWW)*. 521–530.

[28] Paul R. Rosenbaum and Donald B. Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70, 1 (1983), 41–55.

[29] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as Treatments: Debiasing Learning and Evaluation. In *Proceedings of the 33nd International Conference on Machine Learning (ICML)*. 1670–1679.

[30] Adith Swaminathan and Thorsten Joachims. 2015. Batch Learning from Logged Bandit Feedback through Counterfactual Risk Minimization. *Journal of Machine Learning Research (JMLR)* 16 (Sep 2015), 1731–1755.

[31] Adith Swaminathan and Thorsten Joachims. 2015. The Self-Normalized Estimator for Counterfactual Learning. In *Proc. of the 29th Conference on Neural Information Processing Systems (NIPS)*. 3231–3239.

[32] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *Proc. of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 115–124.

[33] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position Bias Estimation for Unbiased Learning to Rank in Personal Search. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining (WSDM)*. 610–618.

[34] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. 2018. The LambdaLoss Framework for Ranking Metric Optimization. In *Proc. of the 27th ACM International Conference on Information and Knowledge Management (CIKM)*. 1313–1322.

[35] Yisong Yue, Rajan Patel, and Hein Roehrig. 2010. Beyond Position Bias: Examining Result Attractiveness as a Source of Presentation Bias in Clickthrough Data. In *Proc. of the 19th International Conference on World Wide Web (WWW)*. 1011–1018.

Learning Edge Properties in Graphs from Path Aggregations

Rakshit Agrawal
University of California, Santa Cruz
Santa Cruz, California
ragrawa1@ucsc.edu

Luca de Alfaro
University of California, Santa Cruz
Santa Cruz, California
luca@ucsc.edu

ABSTRACT

Graph edges, along with their labels, can represent information of fundamental importance, such as links between web pages, friendship between users, the rating given by users to other users or items, and much more. We introduce LEAP, a trainable, general framework for predicting the presence and properties of edges on the basis of the local structure, topology, and labels of the graph. The LEAP framework is based on the exploration and machine-learning aggregation of the paths connecting nodes in a graph. We provide several methods for performing the aggregation phase by training path aggregators, and we demonstrate the flexibility and generality of the framework by applying it to the prediction of links and user ratings in social networks.

We validate the LEAP framework on two problems: link prediction, and user rating prediction. On eight large datasets, among which the arXiv collaboration network, the Yeast protein-protein interaction, and the US airlines routes network, we show that the link prediction performance of LEAP is at least as good as the current state of the art methods, such as SEAL and WLNM. Next, we consider the problem of predicting user ratings on other users: this problem is known as the edge-weight prediction problem in weighted signed networks (WSN). On Bitcoin networks, and Wikipedia RfA, we show that LEAP performs consistently better than the Fairness & Goodness based regression models, varying the amount of training edges between 10 to 90%. These examples demonstrate that LEAP, in spite of its generality, can match or best the performance of approaches that have been especially crafted to solve very specific edge prediction problems.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; • **Theory of computation** → *Graph algorithms analysis.*

KEYWORDS

Path Aggregation; Neural Networks; Edge Learning

ACM Reference Format:
Rakshit Agrawal and Luca de Alfaro. 2019. Learning Edge Properties in Graphs from Path Aggregations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313695

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313695

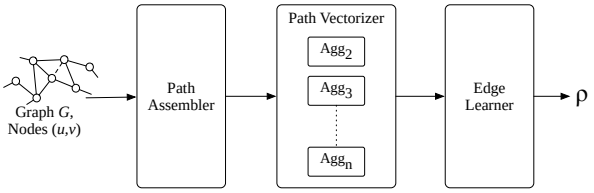


Figure 1: Summarized architecture of LEAP framework

1 INTRODUCTION

Graphs and networks provide the natural representation for many real-world systems and phenomena. In social networks, for instance, different users can be represented by nodes in a graph, and each friendship relation can be represented as an edge. Similarly, in physical systems such as railroad networks or communication networks, each terminal or cellular station is a node in the graph connected to several other terminals. In these graphs, each edge can itself contain significant amount of information. For instance, the presence of an edge in a social network gives a binary signal of presence or absence of a relationship, and a weighted edge on the same network can give a numerical measure of the relationship, hence increasing the degrees of available information. A signed network further contains "positive" and "negative" edge weights, indicating the intensity as well as the direction of a relationship.

The presence and properties of edges in graphs are influenced by several structural factors such as the local neighborhood of the edge, the topology of the graph, and properties and labels associated with surrounding edges in the graph, among others. Machine learning methods can be used to predict the existence of edges, or their properties. For instance, the problem of link prediction is a well explored research area where machine learning methods ranging from heuristics to deep neural networks have been experimented. Similarly, problems around predicting specific edge properties or weights in a graph can also be addressed using learning algorithms.

In this paper, we present a general deep learning framework for learning and predicting edge properties in graphs on the basis of the local neighborhood of the edges. The framework uses the concept of aggregating paths in the graph and is named LEAP (Learning Edges by Aggregation of Paths). A distinctive feature of LEAP is its generality: its ability to learn any kind of edge properties without special need for feature extraction.

In LEAP, for a given graph $G = (V, E)$ where V is the set of nodes and E is the set of edges in the graph, and any two given nodes $(u, v) \in V$, we aim at predicting properties associated with the edge $e_{u,v}$ between the two nodes. For instance, in link prediction, we aim at predicting the presence or absence of this edge, whereas in edge weight prediction, our objective is to predict the weight $w_{u,v}$ associated with the edge $e_{u,v}$. LEAP has a modular architecture consisting of three primary modules: path assembler, path vectorizer, and edge learner (Figure 1). Combined together, these three modules form an end-to-end trainable system. The path assembler gathers paths of different lengths between the node pair (u, v). The pact vectorizer uses an aggregator to summarize the information on the paths into a single vector representation. The edge learner uses the vector representation derived from the path vectorizer, and learns a specific objective for any given edge property.

LEAP's modular architecture makes it easy to implement and experiment. The path aggregators used in LEAP are deep learning modules that take as inputs the raw features of the paths, and produce a trainable aggregation, which is akin to an embedding of the edge set into a vector space. The aggregators thus perform — automatically by computing an embedding — the feature engineering that used to be performed manually, for each property of interest (edge prediction, edge weight prediction, and so on). LEAP, while being an end-to-end learning system that learns embeddings for the nodes and paths itself, can also use pre-trained node embeddings [8, 30], node features, and edge features whenever available.

We present four different kinds of aggregators for LEAP. The aggregators, AvgPool, DenseMax, SeqOfSeq, and EdgeConv, use different neural components and operate at different levels of complexity, focusing on properties like the ordered nature of nodes in a path, and the properties of edges in the paths. We use standard neural modules such as Long Short-Term memory (LSTM) [11] networks, Convolutional Neural Networks (CNN) [19], Pooling operations (Max and Average), and Feed-forward neural networks while constructing our aggregators.

We validate our framework on two specific graph problems: link prediction [21], and edge weight prediction in weighted signed networks (WSN) [18]. In link prediction, we evaluate LEAP on eight real world datasets and present comparisons on the Area under the ROC curve (AUC) score with the current state of the art models WLNM [41] and SEAL [42], and more baseline methods. In the WSN edge weight prediction task presented in Kumar et al. [18] , we evaluate LEAP on three user-user interaction datasets. Two of these datasets refer to Bitcoin trading networks where users provide a rating to other users based on trust. In the third dataset, we learn weights for the votes and sentiment scores assigned by users in Wikipedia to other users when one submits a request for adminship (RfA). We show that LEAP performs similar or better on both these problems against dedicated methods crafted for the specific problems.

The primary contributions of this work can be summarized as follows:

- We present and implement a novel deep learning framework, LEAP, for learning and predicting edge properties of graphs. The framework is general, and it requires no feature engineering, as it relies on deep learning to predict edge properties.
- We define several edge aggregators for LEAP, each suited to particular classes of prediction problems, and we illustrate how LEAP can take advantage of any graph embeddings that may be already available.
- We consider two standard graph prediction problems: link prediction (used, e.g., to predict the formation of connections in social networks) and edge weight prediction (used, e.g., to predict user ratings). We show that LEAP, in spite of its generality, closely matches or improves, on the performance of specialized systems that have been built for these tasks.

In the paper, we will first discuss some related methods for edge property prediction in graphs. We then discuss the motivation behind our LEAP framework and the usage of paths. This is followed by the system design and detailed discussion on aggregators. We then present results from an extensive evaluation over several datasets. We conclude the paper with some considerations on the extensibility and modular design of LEAP.

2 RELATED WORK

Graphs have generated intense research interest over the years in machine learning problems. Commonly studied problems include link prediction [21], node classification, and node ranking [31], among others.

With growing interest in deep learning for graphs, several algorithms for learning node representations have been suggested. These include embedding methods such as LINE [33], DeepWalk [30] and node2vec [8]. More neural network based methods for learning node representations include Graph Convolutional Networks [16], GraphSAGE [9], and Graph Attention Networks [35]. These methods are also often adapted for edge-based learning tasks such as link prediction.

Link prediction has been performed with methods ranging from heuristics to deep neural networks. Martinez *et al.* [23] have categorized the existing link prediction methods into the similarity based, probabilistic and statistical, algorithmic, and preprocessing categories. Similarity based methods operate on the intuition of similar nodes having an affinity towards each other. Locally focused methods like Common Neighbors [21] and Adamic-Adar [2] are interpretable simple methods used extensively for link prediction. More complex similarity-based methods include Katz index [14], PageRank [29], and SimRank [13], among others. Al Hasan *et al.* [10] have explored the use of standard machine learning classifiers for link prediction. Factorization method are also used by Menon and Elkan [24] for link prediction.

Weisfeiler-Lehman Neural Machine (WLNM) [41], and Subgraphs, Embeddings, and Attributes for Link prediction (SEAL) [42] present dedicated deep learning systems for link prediction and define the current state of the art in the space.

In weighted signed networks, for predicting edge weights, Kumar et al [18] defined the learning objective and adapted methods like Bias-Deserve [26], Signed Eigenvector Centrality [5], PageRank [29], and more trust based algorithms. Edge based methods are also used in applications such as SHINE [37] and Rev2 [17] where properties from graphs are used in determine dataset specific tasks.

For the tasks of link prediction and edge weight prediction in weighted signed networks, we present comparison of LEAP with many of these methods later in the paper.

3 MOTIVATION

Edges in real world networks are representative of latent properties within the graph as well as properties among the nodes. Intuitively, for any two nodes (u, v) in the graph, the properties of an edge $e_{u,v}$ between them should depend on the characteristics of the nodes themselves. However, the nodes themselves can be characterized by the edges involving these nodes, hence increasing the dependence to neighborhoods. Therefore, $e_{u,v}$ can be affected by several other nodes and edges in the graph and not just the node pair (u, v). In order to learn more about the edge, it is therefore important to explore the neighborhood of u and v.

For example, if the graph represents a professional or social network, the formation of an edge $e_{u,v}$ between u and v may depend on the properties of u and v, and also on the properties of common friends and friend-of-friends, that is, on the properties of paths emanating from u and v. Equally, in a trust network, the amount of trust of a user u on a user v, constituting a label for $e_{u,v}$, can depend not only on the properties of u and v themselves, but also on the other sets of users that trust, and are trusted by, u and v.

In order to learn more about an edge $e_{u,v}$, it is therefore important to explore the neighborhood of u and v, and assemble the nodes and edges from the graph that can impact the edge $e_{u,v}$ the most. In particular, our framework will consider the paths originating at u and ending at v. These paths can involve a large set of nodes and edges, each of which are related to the two nodes by being an intermediary in a path between them. These intermediate nodes and edges give us information on u and v. The framework we present will enable us to learn from this shared neighborhood of u and v when predicting the properties of $e_{u,v}$.

We note that we could also consider paths that originate at u or v, but do not connect u with v; these paths would provide a characterization of one of u and v only. It would be easy to extend our framework to consider these paths also. However, graph embeddings already enable us to summarize properties of individual node neighborhoods; for this reason, our framework considers mainly the *shared* neighborhood around the edge to be predicted.

4 THE LEAP FRAMEWORK

LEAP is an end-to-end deep learning framework for graph edge learning. The core concept driving LEAP is the ability to learn edge properties in a graph simply from the graph structure, without any need for feature engineering. Moreover, in the presence of explicit features, LEAP can use both the available features as well as self-learned representations from the structure of the graph, such as embeddings. The LEAP framework consists of three separate modules: path assembler, path vectorizer, and edge learner; an overview of the system is presented in Figure 2.

LEAP operates on a given graph $G = (V, E)$, where V is the set of vertices (nodes), and $E \subseteq V \times V$ is the set of edges. An edge can be directed or undirected, weighted or unweighted, and signed or unsigned. A path p^l of length l between the two nodes (u, v) consists of a sequence of nodes $u_0, u_1, \ldots, u_l$, with $u_0 = u$ and $u_l = v$. The set of paths $\mathcal{P}^l_{u,v} = \{p^l_1, p^l_2, \ldots, p^l_n\}$ consists of all the paths of length l between the nodes (u, v). For simplicity, we will often use the notation p^l and $\mathcal{P}^l$ when referring to a path $p^l_{u,v}$ and the set of paths $\mathcal{P}^l_{u,v}$ of length l between the vertices (u, v).

The end-to-end learning objective of LEAP is guided by the input of graph G, two specified nodes (u, v), and the target property ρ being predicted about (u, v). For instance, in case of link prediction, where the framework can be used to predict the presence or absence of an edge between the two nodes (u, v), the output prediction $\rho \in [0, 1]$ from the model represents the probability of an edge existing between (u, v). Similarly, in an edge weight prediction task, the model output ρ can be the predicted weight for edge $e_{u,v}$. We now define the separate modules of the framework used for these learning objectives.

4.1 Path Assembler

The first phase of LEAP is an exploration task which gathers data from the graph structure to be used by the subsequent learning modules. Given a graph G and a pair of nodes (u, v), we start by assembling paths of different lengths between the two nodes. As the learning objective is concerned with the pair (u, v), we do not include the possible 1-length path u, v among the paths considered. As a hyper-parameter provided to the system, we define a set $\mathbf{L} = \{l_1, l_2, \ldots, l_k\}$, as the set of path lengths to be used.

For any length $l \in \mathbf{L}$, we now need to collect a set of paths $\mathcal{P}^l$ of length l. For ease of computation, we can limit the size of each subset, as well as the size of $\mathbf{L}$ as required by the problem and the dataset: if there are too many paths between two nodes, a random subset of paths can be extracted for use by the framework. From the perspective of the framework, the processing is independent of these sizes. Once collected, the paths are then made available to the system in the form of k path-sets $\mathcal{P}^l$ with each set consisting of paths of same length l.

The objective of assembling the paths in length-specific sets is to allow our system to learn properties particular to path lengths. For example, when considering paths of length

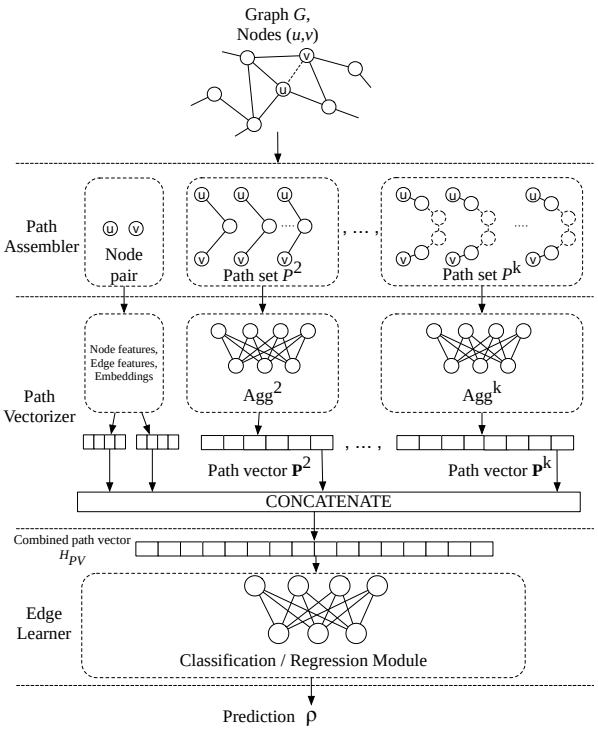


Figure 2: Overview of the LEAP framework

$l = 2$, each path between (u, v) differs only by one vertex. By processing such paths together, we increase the capability of the learning modules to focus on a specific variable property at a time, and therefore capture potential factors affecting the predicted output ρ.

The LEAP framework can be generalized to include the exploration of general paths from u, or to v, rather than only paths from u to v.This can be particularly useful when few (or no) paths connect u to v, while the nodes u and v, individually, belong to many paths.

4.2 Path Vectorizer

At the second phase of LEAP, the assembled paths are passed through a deep learning system. This phase is called the path vectorizer. The objective of this phase is to combine all the available information about the node pair (u, v) using the nodes and the paths, and obtain a vector representation that can be used with different edge learning objectives. This module is inspired by the deep learning architectures capable of learning from complex data structures with different dimensionality. For a graph G with N nodes, each node $x \in V$ is given an integer symbol $x_i \in [1, N]$. In order for LEAP to learn from the graph, these symbolic nodes need to be represented as vectors that can be further processed using neural networks. Different methods of vectorization can be used to represent each node. The simplest notation, which retains the symbolic nature of the nodes, can be to use a one-hot vector of size N. In this notation, node $x_i \in V$ is represented as a vector χ_i of size N where $\chi_i[k] = 1$ if $k = x_i$, and $\chi_i[k] = 0$ otherwise. While this representation allows one to identify each node differently, it does not provide the model with any additional information about the node that can be indicative of its relationship to other nodes or the graph structure. For this purpose, instead of the one-hot vectors, in this paper we use the concept of dense embeddings [8, 9, 25, 30]. We represent each node x_i as a dense vector $\chi_i \in \mathbb{R}^K$ of a fixed dimensionality K. This representation of a node, is obtained through a reference lookup Emb by the entire framework, where $\chi_i = \text{Emb}(x_i)$. Similar to the general use of dense embeddings in deep learning systems, the embeddings can either be trained with the entire system, or can be pre-trained using an embedding generation method first, and then later used with the framework. Additionally, if we have set of node features $\mathcal{N}$ representing engineered feature sets for each node, then they can be used in the framework by combining them with the embeddings. For a node x_i, therefore, the embedding representation used by the system will be $\chi_i \Leftarrow \chi_i | \mathcal{N}(x_i)$, where $(|)$ is a concatenation operation.

The nodes and the paths made available by the path assembler are then used into the path vectorizer using the embedding lookup Emb. Each path, represented as a sequence of nodes in their integer representation is passed through the embedding layer, in order to obtain sequences of vectors, with each sequence of length l representing a path of length $l - 1$. At this stage, we have the two concerned nodes u and v, set of path lengths $\mathbf{L}$, the set of paths $\mathcal{P}^l$ for each length l, and the embedding lookup Emb. In case where edge features $\mathcal{E}$ are available for the graph, LEAP can use these features within the learning module as well. We use Algorithm 1 for obtaining a vector representation H_{PV} from these inputs.

Aggregators. In Algorithm 1, the paths are first converted into their vector representation and then passed through the Aggregator. Aggregators are the primary learning units in LEAP, and are discussed in detail in Section 5. For each path length l, an associated Aggregator Agg_l is responsible for processing the paths and learning a vector representation from them. Initially, in the path vectorizer, path sets of different lengths are provided. Since we aim at learning length-wise significant information from these paths, we process the set for each length separately. An aggregator Agg_l is a deep learning module which takes the path set $\mathbf{P}^l$ as an input, and learns a vector representation $\mathbf{h}_l$. Due to this separate modular structure, LEAP can use several different types of aggregators. The aggregator can also use an edge feature set $\mathcal{E}$ when engineered features for the graph edges are also available. In the path vectorizer, the vectors $\mathbf{h}_l$ learned for each length $l \in \mathbf{L}$ are concatenated together along with the vector representations for the input node pair (u, v) to obtain a final vector representation H_{PV}.

4.3 Edge Learner

The last step of LEAP is to perform problem specific learning on the edge $e_{u,v}$ between the two given nodes (u, v). The

Algorithm 1 PathVectorizer

Input: Node pair (u, v), Path lengths $\mathbf{L}$,
Path sets $\mathcal{P}^l$ for $l \in \mathbf{L}$, Embedding lookup Emb,
Node features $\mathcal{N}$, Edge features $\mathcal{E}$
$\mathbf{u} \leftarrow \text{Emb}(u)|\mathcal{N}(u)$
$\mathbf{v} \leftarrow \text{Emb}(v)|\mathcal{N}(v)$
for l in $\mathbf{L}$ **do**
  $\mathbf{P}^l \leftarrow \text{Emb}(\mathcal{P}^l)|\mathcal{N}(\mathcal{P}^l)$
  $\mathbf{h}_l = \text{Aggregator}(\mathbf{P}^l, \mathcal{E})$
end for
$H_{PV} = \text{Concat}[(\mathbf{u}, \mathbf{v}, \mathbf{h}_l)|l \in \mathbf{L}]$
return H_{PV}

input to the edge learning module is the combined vector H_{PV}. This vector can now be used by any classification or regression method for respective supervised learning problem. For instance, in link prediction problem, where the objective is to detect the presence of an edge between the two nodes (u, v), this module can be used as a binary classifier to classify between *"link exists"* and *"link does not exist"* by predicting a probability $\rho \in [0, 1]$ of a link between (u, v). In general, for any edge based classification, whether binary or multi-class, the module can be used as a classifier with an input vector H_{PV}. The edge learning module can also be used for regression problems. For instance, in edge weight prediction, we can use the output ρ from the module as the predicted weight for the edge. In case of signed edge weight, the same regression module can be used by allowing it to produce both positive and negative values. In multi-class classification, the output ρ can be treated as a vector $\rho \in \mathbb{R}^M$ for M number of classes.

Since the edge learner by itself is simply a classification or a regression module, any corresponding learning algorithm can be used here. For maintaining LEAP as an end-to-end trainable deep learning system, we use feed-forward neural networks for the edge learner. These networks can vary in depth by increasing the number of layers, with the first layer receiving input vector H_{PV} and the final layer predicting the output ρ. Given the vector H_{PV}, and parameter N_{EL} as the number of layers, the process of the edge learner is presented in Algorithm 2

Algorithm 2 EdgeLearner

Input: Combined Path vector H_{PV}, Layer count N_{EL},
$\mathbf{h_{EL}} \leftarrow H_{PV}$
for $c = 1 \ldots N_{EL}$ **do**
  $\mathbf{h_{EL}} \leftarrow \sigma_c(\mathbf{W}_c \cdot \mathbf{h_{EL}} + \mathbf{b}_c)$
end for
$\rho = \sigma_p(\mathbf{W}_p \cdot \mathbf{h_{EL}} + \mathbf{b}_p)$
return ρ

For each layer $c = 1 \ldots N_{EL}$, the weight matrix of the neural network layer is represented by $\mathbf{W}_c$ and the bias of the layer is represented by $\mathbf{b}_c$. σ_c refers to the activation function used by the network layers such as $tanh$, $ReLU$, $sigmoid$ among others. For the final prediction output layer, the weight $\mathbf{W}_p$ and bias $\mathbf{b}_p$ are used. The activation function σ_f for this layer is decided based on the nature of the problem. In case of binary classification, often the $sigmoid \in [0, 1]$ function is used. For multi-class classification, it is common to use the $softmax$ function. In case of normalized edge weight prediction in signed networks, with $w_{u,v} \in [-1, 1]$, we use the $tanh \in [-1, 1]$ activation function to obtain a floating point value representing the predicted signed edge weight.

5 AGGREGATION MODELS

In the previous section, we explained the architecture of LEAP and discussed the requirement of aggregators for performing path vectorization. The concept of using aggregators in graphs is inspired by GraphSAGE [9] where they perform node classification by learning representations for the nodes in a graph. In our system, we have adapted the concept of aggregators for combining paths between two nodes (u, v) and obtaining representations for edges in the graph. Within the path vectorization module, an aggregator Agg_l for paths of length l gets the vectorized path set $\mathbf{P}^l$ as input with an objective of generating output vector $\mathbf{h}_l$.

Since the LEAP framework consists of neural network based layers, each aggregator is itself a deep learning model where the input is a tensor of rank 4 — (batch size, number of paths, path length, node embedding). Each path itself is a sequence of node vectors, and each path set is a set of several such sequences. Therefore, in order to derive single vector representation of the path set, we first need to aggregate all the nodes in the path using aggregator Agg_{node} and then aggregate these paths using the aggregator Agg_{path}. The training of these aggregators is performed with the overall LEAP system using gradient descent based methods. While several other variants are possible, we present four different kinds of aggregators used in this paper.

5.1 AvgPool Aggregator

Our first aggregator follows a simple architecture of combining different vectors together. We call the model for this aggregator AvgPool. This model relies only on the embeddings χ_i for each node x_i under consideration. The model does not have training parameters.

In AvgPool, Agg_{node} concatenates all the node vectors along the path into a single vector. Then, on the set of these derived path vectors, Agg_{path} performs a one-dimensional average pooling operation. The resulting vector $\mathbf{h}_l \in \mathbb{R}^{(l+1)K}$ is therefore a single vector obtained by averaging the paths between the two nodes (u, v) across the paths $k \in K$. The AvgPool aggregator can be summarized as:

$$\mathbf{h}_l = \text{AvgPool}([\overline{(\mathbf{p}_i^l)},\ \forall \mathbf{p}_i^l \in \mathbf{P}^l]) \tag{1}$$

where AvgPool is the one-dimensional average pooling operation, and $\overline{(\cdot)}$ is the vector concatenation operation which

combines multiple vectors by concatenating them together.

$$\overline{([\chi_1, \chi_2, \ldots, \chi_l])} = \chi_1|\chi_2|\ldots|\chi_{l+1} \quad (2)$$

where (|) is the concatenation of two vectors. $\mathbf{P}^l$ is the set of vectorized paths of length l in graph G between the two nodes (u, v). $\mathbf{p}_i^l$, indexed by i is an individual path of length l in set $\mathbf{P}^l$.

AvgPool relies on the embeddings for each node, and represents a path as a fixed size vector of all the nodes combined. Since the first and last node of these paths are the nodes u and v respectively, the only changing bits belong to the nodes within the paths. By performing a bitwise pooling operation over these nodes, we can derive mean vector representations for the changing nodes in the path set. Since the embeddings themselves are still trained by the complete framework, the gradients obtained for updating the node embeddings correspond to these average representations and their influence on the final output ρ.

5.2 DenseMax Aggregator

The DenseMax aggregator is a learning model that uses a dense (feed-forward) neural network layer for each path. Similar to AvgPool, in DenseMax, Agg_{node} obtains the representation for each path by concatenating the node vectors into a single long vector. In this model, at Agg_{path}, the path vector is first passed through a dense neural layer. The resulting activations are then passed through a max-pooling operation which helps derive a single vector representation for the paths of length l. Therefore, Agg_{path} in DenseMax consists of a dense neural network layer and a one-dimensional max pooling operation. The operations of the DenseMax aggregator can be summarized as:

$$\mathbf{h}_l = \textsc{MaxPool}([\sigma(\mathbf{W}_l \cdot \overline{(\mathbf{p}_i^l)} + \mathbf{b}_l), \forall \mathbf{p}_i^l \in \mathbf{P}^l]) \quad (3)$$

where $\mathbf{W}_l$ and $\mathbf{b}_l$ are the weight matrix and bias for the dense neural layer. σ is the activation function used by the dense layer. $\overline{(\cdot)}$ is the vector concatenation operation. MaxPool is the one-dimensional max pooling operation which selects bitwise maximum value from multiple vectors to derive a single final vector. The MaxPool operation is used on these representations in order to capture the most activated bits that can affect the final output ρ.

5.3 SeqOfSeq Aggregator

The sequence of nodes from u to v can hold information relevant to the final prediction. For instance, if the existence of an edge between u and v depends on the presence of a path between the two nodes with consecutively increasing edge weight, the sequential order of nodes contains information which holds significance to the final outcome of the model. Therefore, in SeqOfSeq aggregator, we treat the paths as ordered sequences of nodes. We can further consider the path set as a sequence of different paths, if the paths can be ordered using some characteristics. For example, if the edges are labeled by weights, then the total weight of a path p^l is the sum of edge weights for each edge in the path. If the paths from u to v are then sorted according to their total weights, we can process them in a specified order. To this end, the aggregators Agg_{node} and Agg_{path} would need to be sensitive to the order of the inputs.

In the SeqOfSeq aggregator, we first use an LSTM $\textsc{Lstm}_{inner}$ on each path. From the output activations of $\textsc{Lstm}_{inner}$, we extract a vector representation for the path by performing a max-pooling operation. The aggregator Agg_{node}, in this case, consists of both the $\textsc{Lstm}_{inner}$ and the max-pooling operation. We use a max pool here instead of using only the activation from last timestep of the $lstm_{inner}$: we believe that since our objective is to extract information from the path itself, a max-pooling operation can be more effective in summarizing the path than the final activation. After summarizing each path into a single vector, the sequence of path vectors is processed by a combination of another LSTM $\textsc{Lstm}_{outer}$, followed by a max-pooling operations, as the Agg_{path} aggregator. The SeqOfSeq aggregator can be summarized as:

$$\begin{aligned} H_{inner} &= [\textsc{MaxPool}(\textsc{Lstm}_{inner}(\mathbf{p}_i^l)), \forall \mathbf{p}_i^l \in \mathbf{P}^l] \\ \mathbf{h}_l &= \textsc{MaxPool}(\textsc{Lstm}_{outer}(H_{inner})) \end{aligned} \quad (4)$$

where MaxPool is the one-dimensional max pooling operation, H_{inner} is the intermediate sequence of derived path vectors, and $\textsc{Lstm}_{inner}$ and $\textsc{Lstm}_{outer}$ are the inner and outer LSTMs respectively, used for processing corresponding sequences.

Variants of SeqOfSeq can also be created to use order information only at the paths, or only at the nodes. With the use of sequence learning neural networks such as LSTMs, the SeqOfSeq aggregator is more powerful than the AvgPool or DenseMax aggregators, and it trains a much larger number of parameters.

5.4 EdgeConv Aggregator

Edges of a path can themselves contain significant information. In order to emphasize learning also from the the edges, we propose an aggregator called EdgeConv which focuses on edges while operating over paths. In order to build a learning widget that can operate on the edge, we use a one-dimensional Convolutional Neural Network (CNN) with a window size of 2 that takes as input two consecutive nodes forming an edge. Therefore, when a path is represented as a sequence of nodes, the convolution kernel focuses on all pairs of consecutive nodes along the edge. Given the convolutional results on all pairs of consecutive nodes, we apply a max-pooling operation to compute the overall path label. The aggregator Agg_{node} for EdgeConv, therefore, consists of a one-dimensional CNN and a max-pooling operation. Considering the set of derived paths as an ordered sequence, the Agg_{path} for this case also uses an LSTM and max-pooling operation. Therefore, all the path vectors for paths of length l derived using Agg_{node} are then processed using an LSTM, followed by another max-pooling operation to derive the final vector representation $\mathbf{h}_l$.

EdgeConv can be summarized as :

$$H_{inner} = [\textsc{MaxPool}(\textsc{Conv1D}(\mathbf{p}_i^l)), \forall \mathbf{p}_i^l \in \mathbf{P}^l]$$
$$\mathbf{h}_l = \textsc{MaxPool}(\textsc{Lstm}(H_{inner})) \quad (5)$$

where MaxPool is the one-dimensional max pooling operation, H_{inner} is the intermediate sequence of derived path vectors, and Lstm is the LSTM module used to learn from different path vectors.

Similar to SeqOfSeq, it is not necessary to treat the paths as an ordered sequence, and different variants of EdgeConv can consider the set of derived paths as an unordered set. The sequence of nodes, however, needs to be ordered in EdgeConv, as it operates consecutively over the edges.

5.5 Aggregator Extensions

The four aggregators presented above provide different levels of complexity and use different neural network modules. Together, they illustrate how the modular nature of LEAP allows us to use different neural network architectures as part of the full system for an end-to-end training. We believe that the use of aggregators is key to achieving an extensible and flexible framework. Specialized aggregators can be trained by focusing on significant properties under concern. The aggregators can also include the new deep learning concepts of attention [3, 22] and memory [40]. Similarly, graph specific neural models like Graph Convolutional Networks [16] and Graph Attention Networks [35] can be adapted as aggregators by representing the assembled path sets as subgraphs. While we train the aggregators along with the entire framework, they can be trained separately using any objective function. In case of transfer learning, a well trained model can be transfered into the LEAP framework and can be used simply as a function without training further. Similarly, a partially trained model can be used as an aggregator, and it can further be trained by the learning objective of the complete framework.

6 EVALUATION

The design of the LEAP system, and the use of aggregators, make it an easy to use end-to-end learning system for any kind of graph. For small graphs with fewer nodes, simpler aggregators with few parameters can be used. For very large datasets, we can construct complex aggregators targeted at several latent properties in the graph. In order to demonstrate the learning abilities of this system, we evaluate it on two commonly studied problems in graphs and social networks — link prediction, and edge weight prediction in weighted signed networks.

6.1 Link Prediction

Graphs and networks evolve over time by creation of newer links between the nodes. Given a graph G and a pair of nodes (u, v), the link prediction problem aims at predicting the probability of existence of an edge $e_{u,v}$ between the two nodes.

Table 1: Summary of the datasets used for evaluation

Type	Name	Nodes	Edges
Link Prediction	USAir	332	2,126
	NS	1,589	2,742
	PB	1,222	16,714
	Yeast	2,375	11,693
	C.ele	297	2,148
	E.coli	1,805	14,660
	arXiv	18,722	198,110
	FB	4,039	88,234
Weighted Signed Networks	Bitcoin-OTC	5,881	35,592
	Bitcoin-Alpha	3,783	24,186
	Wikipedia-RFA	9,654	104,554

6.1.1 Learning Objective. In order to learn this objective using LEAP framework, in the Edge Learner module, we can treat it as a binary classification problem. From the graph $G = (V, E)$, the set of edges E can be considered as the positive sample set. Similarly a set of node pairs $(x_1, x_2) \in V$ can be sampled from the graph where edge $e_{x_1,x_2} \notin E$ is the negative set for classification. A label $\tau = 1$, therefore can be associated with positive pairs and a label $\tau = 0$ is associated with the negative pairs.

6.1.2 Datasets. We evaluate the LEAP Link Prediction model on eight real world datasets. The choice of datasets is motivated by the link prediction results presented in two state of the art models — WLNM [41] (Weisfeiler-Lehman Neural Machine) and SEAL [42] (Subgraphs, Embeddings, and Attributes for Link prediction). The datasets used in this paper are listed in Table 1.

USAir [4] is a network graph for US airlines. Network Science (NS) [27] is the collaboration network for researchers in the subject of network science. Political Blogs (PB) [28] is a political blog network form the US. Yeast [36] is a PPI (protein-protein interaction) network for yeast. C.ele is the neural network of the worm Caenorhabditis elegans [38]. E.coli is the dataset of pairwise reaction network of metabolites in E.coli [43]. arXiv [20] is the collaboration network of research papers on arXiv under the Astro Physics category. Facebook (FB) [20] is the dataset of friend lists from the Facebook social network.

6.1.3 Experiment Setup. We performed an extensive set of experiments on the above mentioned datasets in order to evaluate and compare our framework against state of the art methods in link prediction. For each dataset, we sampled a variable number of data samples into the training set and evaluated the model on the remaining samples. The results presented in this paper adopt the partitioning used in the current state of the art method SEAL [42]. For smaller datasets with less than 2500 nodes, we use 90% of the graph edges and an equal number of negative samples for training, and present evaluation results on the remaining 10% edges and equal number of negative

Table 2: Area under the ROC curve (AUC) comparison of LEAP with baselines. Best LEAP results and best dataset results are highlighted. *n2v* refers to the use of node2vec embeddings with LEAP. *OOM* refers to Out-of-Memory.

	USAir	NS	PB	Yeast	C.ele	E.coli	arXiv	FB
Adamic-Adar	0.9507	0.9498	0.9250	0.8973	0.8659	0.9524	-	-
Katz	0.9273	0.9524	0.9306	0.9264	0.8606	0.9329	-	-
PageRank	0.9486	0.9529	0.9374	0.9314	0.9046	0.9548	-	-
node2vec	0.9122	0.9198	0.8621	0.9407	0.8387	0.9075	0.9618	0.9905
Spectral Clustering	0.7482	0.8829	0.8261	0.9346	0.5007	0.9514	0.8700	0.9859
WLK	0.9598	0.9864	OOM	0.9550	0.8965	OOM	-	-
WLNM	0.9571	**0.9886**	0.9363	0.9582	0.8603	0.9706	0.9919	0.9924
SEAL	**0.9729**	0.9761	0.9540	**0.9693**	0.9114	0.9704	0.9940	0.**9940**
LEAP-AvgPool	0.9259	0.9362	0.9555	0.9474	0.9011	0.9484	0.9918	0.9916
LEAP-DenseMax	0.9555	**0.9785**	0.9541	0.9573	0.9050	0.9662	0.9940	0.9914
LEAP-SeqOfSeq	0.9576	0.9635	0.9547	0.9540	0.9153	0.9626	**0.9941**	0.9907
LEAP-EdgeConv	**0.9639**	0.9621	**0.9577**	0.9554	0.9058	0.9614	**0.9941**	0.9908
LEAP-n2v-AvgPool	0.9086	0.9068	**0.9586**	0.9551	0.8909	0.9505	0.9919	0.9920
LEAP-n2v-DenseMax	0.9518	0.9636	0.9564	0.9652	0.9129	**0.9719**	0.9934	0.9914
LEAP-n2v-SeqOfSeq	0.9532	0.9618	0.9571	0.9610	0.9083	0.9662	0.9938	**0.9924**
LEAP-n2v-EdgeConv	0.9547	0.9622	0.9575	**0.9639**	**0.9185**	0.9678	**0.9941**	0.9921

pairs. For relatively larger datasets with more than at least 4000 nodes, we partition the training and evaluation datasets at 50%.

The LEAP[1] system and the aggregators were written in Python with Keras [7] deep learning framework, using the Tensorflow [1] backend. The hyperparameters for each aggregator were selected using multiple trials. We report the results with the best hyperparameters for each setting individually. In all the reported results, we used the path lengths in set $\mathbf{L} = \{3, 4\}$, and used upto 50 paths for each length selected randomly. All the methods were trained using a loss function of binary cross-entropy and the Adam [15] optimizer for gradient descent with a learning rate of 0.001. Each model was trained for upto 30 epochs with early stopping enabled.

6.1.4 Results. We present the results obtained as per the above mentioned experimental setup for LEAP in table 2. The results comprise of LEAP with the four aggregators — AvgPool, DenseMax, SeqOfSeq, and EdgeConv, discussed in the paper. Additionally, we also compare the ability of our model to learn node embeddings themselves against using pre-trained embeddings. For pre-trained embeddings, similar to SEAL, we use the node2vec [8] method first on the graph to derive node embeddings, and then use them with the LEAP framework without updating them further during the system training. We compare these results with three different kinds of methods used for link prediction. We first use the heuristics including Adamic-Adar [2], Katz index [14], and the PageRank [6] algorithm. For feature learning based models, we use spectral clustering [34], and the node2vec [8] algorithm which learned node embeddings and then performs a link prediction task on them. Finally, we compare our system with subgraph-based link prediction methods, defining the current state of the art. These methods include Weisfeiler-Lehman graph kernel (WLK) [32], Weisfeiler-Lehman Neural Machine (WLNM) [41], and Subgraphs, Embeddings, and Attributes for Link prediction (SEAL) [42]. Performance of all the models is compared using the Area under the ROC curve (AUC) metric. We use the settings and results by Zhang et al. [42] for all our baseline methods.

As can be seen in table 2, LEAP performs best or close-to-best on each dataset. Further, we show that an external method for learning node embeddings like node2vec can easily be used within the system. Similarly we can also incorporate known feature vectors for the nodes whenever available.

In comparison to deep learning methods and current state of the art WLNM and SEAL, LEAP achieves equivalent or better performance with the presented aggregators. However, due to its modular nature, the LEAP framework can be highly extended and adapted for different datasets with different latent properties. Further, SEAL requires two prominent steps of node labeling and embedding generation before the neural network can be trained on the graph. In making a learning framework easily deployable on multiple platforms, it is highly advantageous to have an end-to-end trainable system and LEAP provides this particular ability with sufficient modularity to tune the simplicity of the model as required.

[1]The code is available at https://github.com/rakshit-agrawal/LEAP

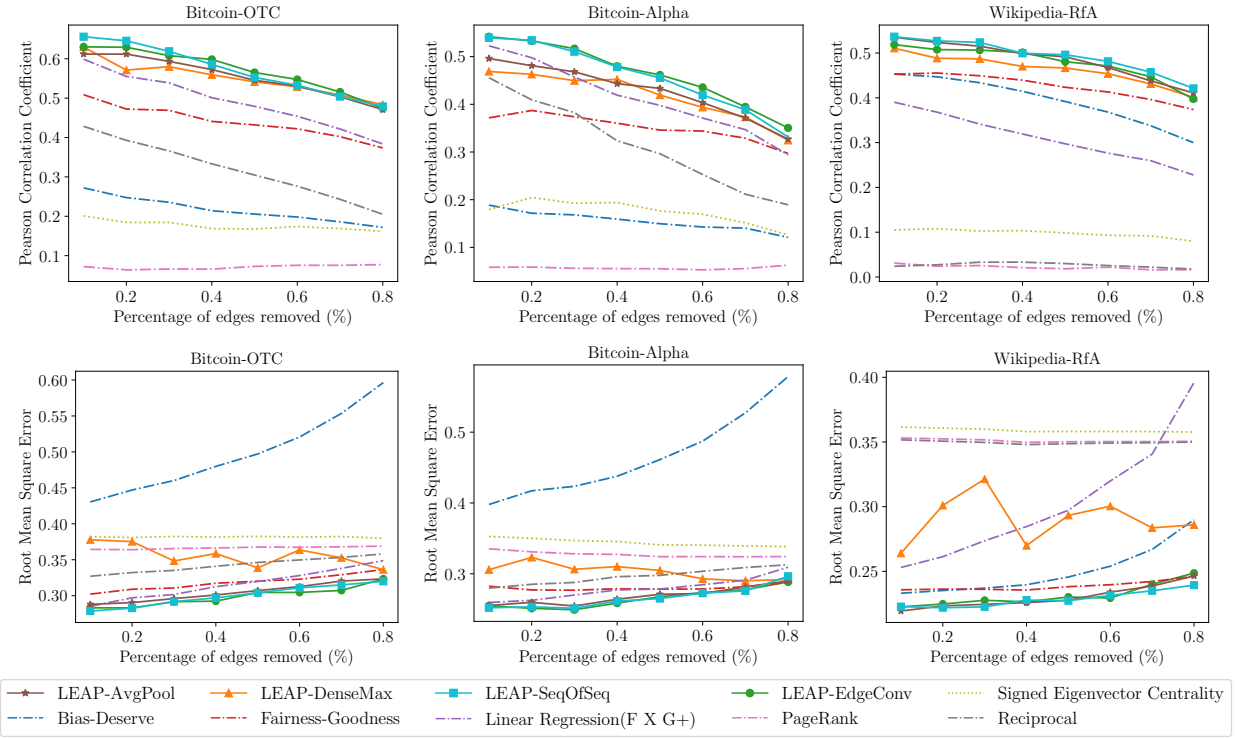


Figure 3: Plots for PCC(top) and RMSE(bottom) on the three datasets for Weighted Signed Networks. The x-axis refers to the percentage of edges removed while training the models. Over all the datasets, and along both the metrics, LEAP based methods show significantly better performance, with the complex aggregators SeqOfSeq and EdgeConv giving the best performance on both the metrics.

6.2 Weighted Signed Networks

In real world datasets representing relations of certain kind among the nodes, the edges can possess different meaningful properties that are of significance to the underlying network. For instance, in a user-user interaction system, each user can have a trust associated with another user. This trust further propagates through the network, affecting the trust between two different users in a certain way. A generalized representation of such networks is obtained through weighted signed networks (WSN). A WSN consists of a graph $G = (V, E)$ where an edge $e_{u,v} \in E$ between two nodes $(u, v) \in V$ has a weight $w_{u,v}$ associated to it. The weight $w_{u,v}$ can be a signed weight, specifying a positive or negative sentiment with a magnitude $|w_{u,v}|$ specifying the intensity of the relation. A WSN graph can either be directed or undirected.

6.2.1 Learning Objective. Kumar et al [18] presented the task of predicting edge weight in WSNs where given a graph G and a node pair (u, v), the objective is to predict the signed edge weight $w_{u,v}$ between the two nodes. For simplicity, the edge weights are normalized to a scale of $w_{u,v} \in [-1, 1]$. In the case of the LEAP framework, the problem of predicting edge weight can be treated as a regression problem for the Edge Learner module. From the graph $G = (V, E)$, we can use the entire set of edges E with the edge nodes $(x_1, x_2) \in V$ for edge $e_{x_1,x_2} \in E$ being the input nodes to the system. Each such node pair can be associated with a regression label $\tau = w_{x_1,x_2}$ using the weights associated with each edge.

6.2.2 Datasets. For evaluation of the LEAP WSN Edge Weight Predictor, we use three real world datasets of user-user interaction networks. These datasets used here are influenced by Kumar et al [18] and are available from SNAP database [20]. The three datasets are listed in Table 1.

Bitcoin-OTC [18] is the "who-trusts-whom" network of people trading Bitcoins on the platform "Bitcoin OTC". The directed signed edge weight between the users on this network refers to the rating given by a user to the other user on the network on a scale from -10 to 10. Bitcoin-Alpha [18] is a similar network for a different trading platform called "Bitcoin Alpha". Wikipedia-RFA [18, 39] is a voting network for Wikipedia request for adminship (RfA). The edges on the network refer to directed votes between users. To associate weights to the signed edges, Kumar et al [18] used the VADER sentiment engine [12] and used the difference between the positive and negative sentiment scores obtained for the vote explanation text in the Wikipedia-RFA dataset.

6.2.3 Experiment Setup. In this evaluation, we present and compare the results for regression on predicting edge weights with $\delta\%$ edges removed. We vary the value of δ between 10% to 80%, with a step size of 10%, specifying the range for partitioning the training and the evaluation datasets.

The system and hyperparameter settings for this task are same as the link prediction model. The only difference in this adaptation of LEAP for a regression task is the choice of loss function. We use the Mean Squared Error (MSE) as the loss function used to compute the gradients for training the Edge Weight Prediction model. The hyperparameter selection was performed using multiple trials, and the best settings were used to report the results.

6.2.4 Results. Similarly to link prediction, we evaluated our system for this objective with the four aggregators — AvgPool, DenseMax, SeqOfSeq, and EdgeConv, discussed in the paper. Neural network based models are not used directly for the task of edge weight prediction in WSN and therefore we compare our results to the heuristic and feature learning based methods previously applied on these datasets.

In the definition of edge weight prediction on WSNs by Kumar et al [18], they adapted several algorithms for this task, providing us with a set of baseline measures on these datasets. We first use a basic method of Reciprocal [18], where the edge weight $w_{u,v}$ is same as that of the reciprocal edge weight $w_{v,u}$ if there exists an edge $e{+}v, u \in E$, and 0 otherwise. We then use two graph algorithms PageRank [29] and Signed Eigenvector Centrality [5]. Each of these algorithms independently learn a score for each node in the graph. The edge weight predicted by these methods simply refers to the difference between the scores obtained for the nodes u and v. Finally, we compare our system with relation specific algorithms used for extracting interaction measures between the nodes. These include Bias-Deserve [26] and Fairness-Goodness [18]. These are iterative algorithms that associate two properties with each node, and learn them by performing sequential iterations on all the nodes in the graph, updating one property at a time. An additional method used in [18] is a Linear Regression model using the above mentioned values as features. This method is identified as Linear Regression (F × G +)

All these methods were evaluated using the experiment setup mentioned above and the results were measures across two standard metrics. We first measure the Root Mean Square Error (RMSE) on the predicted weights to highlight the closeness between predicted and true weights. We then measure the Pearson Correlation Coefficient (PCC) on the predicted weights in order to measure the relative trend in the prediction.

As can be seen in Figure 3, LEAP based methods outperform all the baseline methods on the three datasets, hence defining the new state of the art for edge weight prediction. Moreover, with decreasing number of known edges, the performance of LEAP degrades much slower compared to the other methods.

6.3 Discussion

The results presented above on two edge learning problems prove the ability of LEAP to learn edge properties directly from the structure of the graph. Beyond this, we believe that this framework presents a unique quality of extensibility. LEAP is a highly modular system that can be adapted for any learning objective on the egde properties. In the two tasks presented in the paper, by just making a small change in the edge learner module for each task, the system was able to match the state of the art results. In doing so, it did not require any heuristics or feature extraction methods, and could learn only through the aggregation of paths between the two concerned nodes. Further, LEAP can be trained end-to-end, making it much more convenient to code, use, and maintain, as well as more interpretable since the weights throughout the network are updated using a common objective. The adaptability of LEAP also makes it a potential platform for future research in learning edge properties, and for use of neural networks in graphs.

7 CONCLUSIONS

We presented a novel end-to-end deep learning framework called LEAP for learning edge properties in a graph in this paper. LEAP includes a modular structure consisting of a path assembler, path vectorizer, and an edge learner. For any graph $G = (V, E)$ and two given nodes $(u, v) \in V$, the system learns properties associated with the edge $e_{u,v}$ by aggregating paths between them from the graph. The aggregation is performed using deep learning modules which can be selected based on the dataset and the problem under concern. The system can perform different kinds of supervised learning tasks such as binary or multi-class classification, multi-label classification and regression among others. Being powered by neural modules, the complete framework of LEAP is a layered deep learning model that can be trained end-to-end using gradient descent based methods.

We demonstrate that LEAP can obtain state-of-the-art performance for different learning problems on several real world datasets. For two specific problem of link prediction, and edge weight prediction in weighted signed networks, LEAP shows great performance by matching or improving upon the current state of the art. We also show that the LEAP framework is easily extensible, and can also incorporate node embeddings, node features and edge features into the system. We believe that this system can act as a great platform for experimentation in edge learning, and can be adapted for several different problems. We also believe that the simple architecture of LEAP allows it to be an easily deployable neural model in production environments.

REFERENCES

[1] Martín Abadi and et al. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/ Software available from tensorflow.org.

[2] Eytan Adamic and Lada A. Adar. 2003. Friends and neighbors on the web. 3 (July 2003), 211–230.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*

abs/1409.0473 (2014). arXiv:1409.0473 http://arxiv.org/abs/1409.0473
[4] Vladimir Batagelj and Andrej Mrvar. 2006. Pajek datasets. http://vlado.fmf.uni-lj.si/pub/networks/data/.
[5] Phillip Bonacich. 2007. Some unique properties of eigenvector centrality. *Social Networks* 29 (2007), 555–564.
[6] Sergey Brin and Lawrence Page. 1998. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer Networks* 30 (1998), 107–117.
[7] François Chollet et al. 2015. Keras. https://keras.io.
[8] Aditya Grover and Jure Leskovec. 2016. Node2Vec: Scalable Feature Learning for Networks. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 855–864. https://doi.org/10.1145/2939672.2939754
[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 1024–1034. http://papers.nips.cc/paper/6703-inductive-representation-learning-on-large-graphs.pdf
[10] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. 2006. Link prediction using supervised learning. In *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*.
[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9 (1997), 1735–1780.
[12] Clayton J. Hutto and Eric Gilbert. 2014. VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text.. In *ICWSM*, Eytan Adar, Paul Resnick, Munmun De Choudhury, Bernie Hogan, and Alice H. Oh (Eds.). The AAAI Press. http://dblp.uni-trier.de/db/conf/icwsm/icwsm2014.html#HuttoG14
[13] Glen Jeh and Jennifer Widom. 2002. SimRank: A Measure of Structural-context Similarity. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02)*. ACM, New York, NY, USA, 538–543. https://doi.org/10.1145/775047.775126
[14] Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (01 Mar 1953), 39–43. https://doi.org/10.1007/BF02289026
[15] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).
[16] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.
[17] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and V.S. Subrahmanian. 2018. REV2: Fraudulent User Prediction in Rating Platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 333–341. https://doi.org/10.1145/3159652.3159729
[18] Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. 2016. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 221–230.
[19] Yann LeCun and Yoshua Bengio. 1998. Convolutional networks for images, speech, and time series.
[20] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.
[21] David Liben-Nowell and Jon Kleinberg. 2003. The Link Prediction Problem for Social Networks. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management (CIKM '03)*. ACM, New York, NY, USA, 556–559. https://doi.org/10.1145/956863.956972
[22] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. *CoRR* abs/1508.04025 (2015). arXiv:1508.04025 http://arxiv.org/abs/1508.04025
[23] Víctor Martínez, Fernando Berzal, and Juan-Carlos Cubero. 2016. A Survey of Link Prediction in Complex Networks. *ACM Comput. Surv.* 49, 4, Article 69 (Dec. 2016), 33 pages. https://doi.org/10.1145/3012704
[24] Aditya Krishna Menon and Charles Elkan. 2011. Link Prediction via Matrix Factorization. In *Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part II (ECML PKDD'11)*. Springer-Verlag, Berlin, Heidelberg, 437–452. http://dl.acm.org/citation.cfm?id=2034117.2034146
[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. (oct 2013). arXiv:1310.4546 http://arxiv.org/abs/1310.4546
[26] Abhinav Mishra and Arnab Bhattacharya. 2011. Finding the Bias and Prestige of Nodes in Networks Based on Trust Scores. In *Proceedings of the 20th International Conference on World Wide Web (WWW '11)*. ACM, New York, NY, USA, 567–576. https://doi.org/10.1145/1963405.1963485
[27] Micaleah Newman. 2006. Finding community structure in networks using the eigenvectors of matrices. *Physical review. E, Statistical, nonlinear, and soft matter physics* 74 3 Pt 2 (2006), 036104.
[28] M. E. J. Newman. 2006. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E* 74 (Sep 2006), 036104. Issue 3. https://doi.org/10.1103/PhysRevE.74.036104
[29] L. Page, S. Brin, R. Motwani, and T. Winograd. 1998. The PageRank citation ranking: Bringing order to the Web. In *Proceedings of the 7th International World Wide Web Conference*. Brisbane, Australia, 161–172. citeseer.nj.nec.com/page98pagerank.html
[30] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, New York, NY, USA, 701–710. https://doi.org/10.1145/2623330.2623732
[31] Moshen Shahriari and Mahdi Jalili. 2014. Ranking Nodes in Signed Social Networks. *Social Network Analysis and Mining* 4, 1 (30 Jan 2014), 172. https://doi.org/10.1007/s13278-014-0172-x
[32] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. 2011. Weisfeiler-Lehman Graph Kernels. *J. Mach. Learn. Res.* 12 (Nov. 2011), 2539–2561. http://dl.acm.org/citation.cfm?id=1953048.2078187
[33] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1067–1077. https://doi.org/10.1145/2736277.2741093
[34] Lei Tang and Huan Liu. 2011. Leveraging social media networks for classification. *Data Mining and Knowledge Discovery* 23, 3 (01 Nov 2011), 447–478. https://doi.org/10.1007/s10618-010-0210-x
[35] Petar VeliÄŒkoviÄĞ, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro LiÃš, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJXMpikCZ
[36] Christian Von Mering, Roland Krause, Berend Snel, Michael Cornell, Stephen G. Oliver, Stanley Fields, and Peer Bork. 2002. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature* 417, 6887 (23 5 2002), 399–403. https://doi.org/10.1038/nature750
[37] Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. 2018. SHINE: Signed Heterogeneous Information Network Embedding for Sentiment Link Prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 592–600. https://doi.org/10.1145/3159652.3159666
[38] Duncan J. Watts and Steven H. Strogatz. 1998. Collective dynamics of 'small-world' networks. *Nature* 393, 6684 (04 June 1998), 440–442. https://doi.org/10.1038/30918
[39] Robert West, Hristo S. Paskov, Jure Leskovec, and Christopher Potts. 2014. Exploiting Social Network Structure for Person-to-Person Sentiment Analysis. *TACL* 2 (2014), 297–310.
[40] Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory Networks. *CoRR* abs/1410.3916 (2014).
[41] Muhan Zhang and Yixin Chen. 2017. Weisfeiler-Lehman Neural Machine for Link Prediction. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*. ACM, New York, NY, USA, 575–583. https://doi.org/10.1145/3097983.3097996
[42] Muhan Zhang and Yixin Chen. 2018. Link Prediction Based on Graph Neural Networks. In *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). Curran Associates, Inc., 5165–5175. http://papers.nips.cc/paper/7763-link-prediction-based-on-graph-neural-networks.pdf
[43] Muhan Zhang, Zhicheng Cui, Shali Jiang, and Yixin Chen. 2018. Beyond Link Prediction: Predicting Hyperlinks in Adjacency Space. (2018). https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17136

Evaluating User Actions as a Proxy for Email Significance

Tarfah Alrashed
MIT, USA
tarfah@mit.edu

Chia-Jung Lee
Microsoft, USA
cjlee@microsoft.com

Peter Bailey
Microsoft, Australia
pbailey@microsoft.com

Christopher Lin
Microsoft, USA
christol@microsoft.com

Milad Shokouhi
Microsoft, USA
milads@microsoft.com

Susan Dumais
Microsoft, USA
sdumais@microsoft.com

ABSTRACT

Email remains a critical channel for communicating information in both personal and work accounts. The number of emails people receive every day can be overwhelming, which in turn creates challenges for efficient information management and consumption. Having a good estimate of the significance of emails forms the foundation for many downstream tasks (e.g. email prioritization); but determining significance at scale is expensive and challenging.

In this work, we hypothesize that the cumulative set of actions on any individual email can be considered as a proxy for the perceived significance of that email. We propose two approaches to summarize observed actions on emails, which we then evaluate against the perceived significance. The first approach is a fixed-form utility function parameterized on a set of weights, and we study the impact of different weight assignment strategies. In the second approach, we build machine learning models to capture users' significance directly based on the observed actions. For evaluation, we collect human judgments on email significance for both personal and work emails. Our analysis suggests that there is a positive correlation between actions and significance of emails and that actions performed on personal and work emails are different. We also find that the degree of correlation varies across people, which may reflect the individualized nature of email activity patterns or significance. Subsequently, we develop an example of real-time email significance prediction by using action summaries as implicit feedback at scale. Evaluation results suggest that the resulting significance predictions have positive agreement with human assessments, albeit not at statistically strong levels. We speculate that we may require personalized significance prediction to improve agreement levels.

CCS CONCEPTS

• **Information systems** → **Email**; *Clustering and classification*; *Task models*; • **Human-centered computing** → *User models*.

KEYWORDS

Email communication significance; log data; user activity modelling

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313624

ACM Reference Format:
Tarfah Alrashed, Chia-Jung Lee, Peter Bailey, Christopher Lin, Milad Shokouhi, and Susan Dumais. 2019. Evaluating User Actions as a Proxy for Email Significance. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, Article 4, 11 pages. https://doi.org/10.1145/3308558.3313624

1 INTRODUCTION

It is a truism that email is still one of the most important means of online communication. While individual email volumes vary greatly, we have observed that work email accounts receive more than 100 emails per day on average, while personal email accounts receive an order of magnitude less on average per day, based on a one-week sample of mailboxes from Microsoft Outlook.[1] To assist people with this amount of incoming email, previous work constructs predictive models to direct attention to emails of potentially higher importance or urgency (e.g. [1, 8, 10, 32]). These models tend to focus on predicting whether some set of strong actions such as Reply or Forward will take place. While such predictors have demonstrated value, strong actions such as Reply comprise only a small subset of the possible sets of actions people can take within their email applications [8]. This may create a gap for identifying important emails because strong actions are not the sole actions indicative of importance (e.g. reading an email several times can represent high importance).

In this work, we set out to understand what, if any, relationship exists between the actions carried out on email by people and their perceptions of the significance of those emails. By *significance*, we mean the quality of being worthy of attention that a person accords to an email. Emails that are important, or urgent, or both, are likely to be significant. Other factors may also contribute, such as who it is from or the topic of the email. Our initial analysis based on a large email log sample in Section 3 shows that the actions people perform differ for emails identified as significant versus insignificant. We draw inspiration from an arc of research in the web search community on user actions and inferred document relevance, proceeding from simple clicks [16], to clicks and dwell-time [12], to an entire set of user actions [2].

We conjecture that significance is a quality that exists on a continuous spectrum, and that people make explicit and implicit choices when dealing with their email that reflects this awareness. We also hypothesize that actions carry different semantics and contribute to significance to varying degrees; therefore, diverse actions should be considered when establishing their relationship with significance. Two approaches are investigated for summarizing observed actions

[1] These statistics align with those from a market analysis report by Radicati [22].

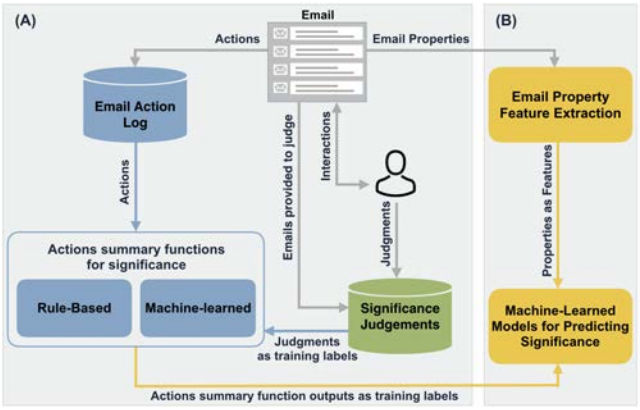


Figure 1: (A) Our framework of evaluating actions as a proxy for email significance, based on action logs and human annotation data. (B) Real-time email significance prediction model trained from actions summary function outputs (§ 7).

in a way that the summaries can then approximate email significance. The first approach presents the notion of weighted action utility (WAU), which is a rule-based fixed-form utility function that summarizes the set of observed actions using a linear combination parameterized on a set of action-specific weights. The second approach attempts to model significance directly using machine learning (ML) techniques.

To evaluate the approaches, we create a human intelligence task (HIT) survey to collect significance judgments from people, including both personal and work accounts. Our results suggest that WAU makes better summaries for determining email significance compared to a reply-only baseline, which is widely adopted as a standard notion for approximating email importance [32] in the literature. As expected, using ML techniques is more effective to make inference over actions and predict email significance. Our follow-up analysis demonstrates that significance prediction using actions varies across users, which may reflect the personalized nature of email activity patterns and significance.

Overall our evaluation results demonstrate that user actions, be it rule-based or machine-learned, can be used as a proxy for users' perceived significance over their own emails. This finding lays an important foundation for downstream tasks such as email prioritization. Building an effective email prioritization predictor with ML usually requires a large quantity of training data, especially given that past research (e.g. [7, 31]) has indicated that email processing is highly personal. Therefore having a means to create unsupervised or semi-supervised training labels at scale is more tractable and practically feasible than aiming for huge quantities of hand-annotated data or explicit user feedback. With the relationship between actions and significance established, we can train such a predictor by leveraging action summaries as a training label, which can be obtained at a low cost by mining action logs. Indeed, in the context of web search, similar ideas of leveraging users' implicit feedback have been studied extensively (e.g. [2, 12, 14, 16]). Specifically, users' actions such as clicks and dwell-time on documents in response to queries have been shown as effective implicit labels of document relevance, based on which standard learning-to-rank techniques are then applicable. To demonstrate this use case, we show an example of how to create a real-time email significance predictor on large-scale email action samples by using the outputs of an action summary function as labels and email properties as features. Our results suggest that the email significance predictions have positive agreement with human assessments, albeit not at statistically strong levels.

To summarize our main contributions, we first evaluate and show that user actions can be used as a proxy for email significance. We consider and conduct extensive experiments on both rule-based and machine-learned techniques to summarize user actions. The results suggest that both approaches outperform a standard reply-only baseline by showing a higher correlation with significance. Figure 1 (A) depicts a high level overview of the process. Furthermore, we identify the analogy between using more than just clicks as implicit feedback for relevance labels in web search and using action summaries as proxies for significance in email processing. In particular, we demonstrate an example of how to leverage an action summary as supervision to train a large-scale email significance predictor, as shown in Figure 1 (B). The predictor can be executed in real-time as it relies on a set of email property features that can be extracted upon email arrival.

2 RELATED WORK

As the volume of email grows and the demand for human attention increases correspondingly, challenges related to email management and retrieval increase [10]. Previous research investigates the different ways that people use and manage their emails in both work and personal settings [6, 11, 31]. As the stream of emails increases, managing email flow is another challenge that has been addressed in the literature [15, 28]. Previous work suggests that with the incoming email flow, people would choose to visit some email messages before others, and spend time selecting which messages to check first [3, 28, 29]. A study has shown that people scan their inbox a couple of times on average before selecting a message to read [4]. Venolia et al. [29] described five stages of email workflow: email flow, triage, task management, archive, and retrieve. Siu et al. [28] extended the work of [29] and their results suggested that people interleave flow, triage and task management, and that handling incoming email involves three steps: glance, scan, and defer. People will usually glance at their inbox several times until a number of new emails have arrived or a message they are expecting has arrived. They would then scan the headers of the new email messages to decide which of these emails need to be read or acted upon. Then they will begin acting on some of these emails messages and/or defer others to return to them later. Although Siu et al. [28] describe the overall steps people take to deal with and manage email flow, it does not address the features people use during the message selecting process and does not describe why people pay more attention to certain email messages over others. Through a user survey and log analysis, Sarrafzadeh et al. [25] investigated how people choose to defer their email for later processing, and whether the decision to defer can be predicted.

Previous work on email prioritization has tried to facilitate the message selection process by including a priority field in messages,

helping people pay attention to important emails easily [21]. Other work focused on allowing the sender to assign a priority to a message by adding a "price" to some messages and not others [19, 23]. However, such approaches were not widely adopted, and the priority field was usually ignored by the users. An analysis to understand why people pay attention to some emails over others is needed to better support the development of such systems. Research by Wainer et al. [30] tried to identify why people attend to some emails and not others based on inbox-level cues about message content. In their think-aloud study, they found that individuals make inferences about message content based on top-level cues and that inferred utility as well as curiosity seem to drive attention to a message. In a controlled laboratory experiment they conducted, in which they investigated the relationship between information gap, utility and demand, they found that curiosity drives attention to email under conditions of low demand, and independent of the marked importance of a message. The work by Wainer et al. is an interesting resource to understand what attracts people's attention to their emails when they arrive; however, more insights are needed to be able to predict such attention. Actions on the other hand, have been used as an indicator in several previous works. Machine learning algorithms to predict the likelihood that a message requires a response are described by Yang et al. [32]. That said, the work by Dabbish et al. [8] found that the need to respond is only one part of defining the importance of an email message, and that people responded to information requests or social messages, even though these messages were unimportant for work.

Other email prioritization work focuses on making a personalized prediction of the significance label of emails [33, 34]. Aberdeen et al. [1] used linear logistic regression models to rank mail in the Gmail mail service by how likely the user was to act on an email without explicit labeling from the user. In that model, they used four categories of features: social, content, thread and label features. Neustaedter et al. [21] described a prototype email client that aggregates social meta-data about email correspondents to support email triage. They define metrics for measuring the social importance of users based on elements like recipients' information and email activities, which can be used for predicting relative email ordering.

The work we are presenting here is similar to this line of research in that we are also interested in understanding email prioritization. However, in our work, we are focused just on whether actions that people take on their emails are related to how they perceive significance, rather than attempting to directly introduce a prioritized email system. Our work follows a similar approach to recent work on email search by Kim et al. [18], in which they mapped explicit *in situ* judgments of email search success and effort to implicit actions that people perform on email; instead, we are mapping our users' judgments to email significance. It is also similar to previous work on web search, in which Huang et al. [14] examined mouse cursor behavior on search engine results pages (clicks, cursor movements and hovers over different page regions), as a proxy for relevance; we are using email interactions, specifically actions performed on emails, as a proxy for email significance.

3 ACTIONS AND IMPLICIT SIGNIFICANCE

In an effort to help users be more productive with their email, popular email services have introduced a number of user experience controls to indicate the significance of individual emails. For example, Gmail's Priority Inbox and Outlook's Focused Inbox both attempt to automatically group emails into two categories of importance. These experiences can often provide interesting insights into users' perceptions of email significance, since they allow users to explicitly move emails from one group to another. In the case of Outlook, users can move emails between the Focused and Other tabs. When a user moves an email to the Focused tab, such explicit interaction can say something about the intention of the user – the current email is likely to be more significant. The reversed operation (i.e., moving an email to the Other tab) may implicitly imply that the user considers this email is less significant. We use these explicit interactions with emails to conduct a quantitative study to investigate if action distributions resemble or differ in the two classes of emails, which we refer to as *implicit significant* and *implicit insignificant*. This helps test if actions can be a good proxy for significance. While we understand emails grouped based on these explicit interactions may not necessarily correspond to a user's notion of significance, this is a close surrogate signal for us based on existing user experiences.

To compare the action distributions, we analyze two random, anonymized action log samples provided by Outlook, one for personal and the other for work email activities. In the personal email sample, we have approximately 500 million users and 17 billion messages, and for work emails, we have 170 million users and 8 billion messages. The email service can be accessed from a number of clients, including native apps for both desktop and mobile as well as browser-based interfaces. The logs did not provide access to any content of the email message, email headers, or email search queries. The logs did contain records of the actions performed against the emails, with corresponding timestamps and other metadata such as the client interface type.

Based on these two samples, we analyze the distribution of actions for these two classes, where we consider diverse actions including but not limited to Read, Reply, Forward, Open an Attachment, Click a Link (in the email), Delete, etc. Figure 2 compares the results. The plot suggests that action distributions for implicit significant and implicit insignificant emails are clearly different. This implies that people interact very differently with emails of different properties, as reflected by the corresponding actions. Particular actions, such as replying to a work email or opening an attachment in a personal email might be good indicators of the significance of this email. Conversely, actions such as deleting an email might indicate it to be of low importance. This observation, to some extent, shows supportive evidence for our hypothesis that using actions as significance proxy is sensible.

4 ACTIONS AS A PROXY FOR SIGNIFICANCE

Next we present approaches for summarizing email significance based on a rule-based function and on machine-learned predictors.

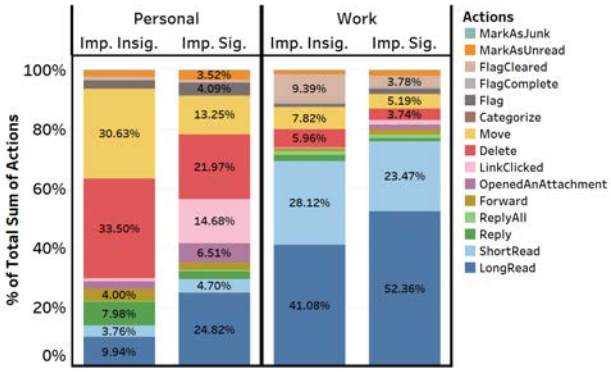


Figure 2: Distribution of actions by implicit insignificant (Imp. Insig.) and implicit significant (Imp. Sig.) emails from a sample of personal and work mailboxes.

4.1 Weighted Action Utility

To summarize the utility obtained from a person's attention on an email, given a set of actions performed on that email, we need some quantity that represents the all-up utility derived from those actions. By *utility* we mean that the attention was useful or beneficial (but not necessarily significant). A naive approach would be to count all actions, giving equal utility for any action. Prior research (e.g. [8]) indicates that different actions (e.g. a Reply to an email vs a Delete) have different degrees of utility. In addition, we can consider that certain compound actions (e.g. a Read followed by a Reply) may indicate that the Read action was more valuable than a Read action in isolation. Thus we also need to allow each type of action (or compound action) to be given a weight.

We propose Weighted Action Utility (WAU), a rule-based fixed-form utility function that summarizes the set of observed actions using a linear combination parameterized on a set of action-specific weights, which is defined as:

$$WAU(e) = \sum_{i=0}^{n} w_{A_i}.A_i \tag{1}$$

where e is an email, $\{A_1, A_2, ..., A_n\}$ is the set of actions performed on e, and w_{A_i} is the weight associated with an action A_i. There are many ways to determine the weights to be associated with each type of action; here we propose two weight configurations:

4.1.1 Authors' weights. Users can benefit from email in many ways, like by gaining information or organizing their thoughts, and this utility is manifested in the logs through their interactions in their email client (e.g. reading emails, writing emails, pinning emails). Thus, our definition of utility first simply assigns utility scores to a set of user actions and sums these scores over every action taken by the user in their email client. Note however, that some actions do not impart utility by themselves, but in combination with whatever item was acted upon. For example, reading an email that teaches the user something has high utility, whereas reading an email to determine that it is spam provides no utility. Thus, we extend our definition of utility by assigning scores to actions that indicate positive utility if they are preceded by a Read. In other words, whereas normally an action like Pin has zero utility, if it is preceded by a Read, then we assign it some positive utility. The way we assign weights to actions is by determining how important and valuable certain actions are compared to others. For example, we believe that Reply and Reply-all actions are important actions, so we assign them higher weights compared to Delete actions which we do not think should be assigned high value.

4.1.2 Crowd-sourced estimates. People might have very different perspectives from us in terms of how they accord utility to their actions. To obtain a broader and more representative estimate of these utility weights, we ask a pool of 40 people from a private crowd-worker service to assign weights to different types of actions commonly carried out on email. We then take the average of these weights and apply them within WAU. By considering these weights alongside the authors' weights, the objective is to see if different sets of weights can give different results for both or either personal and work emails. Our results and findings are discussed in Section 6.

4.2 Predicting Significance using ML

An alternative method that sidesteps WAU altogether is to learn a model from the actions to the significance annotations created by human directly. One challenge here is whether we can obtain sufficient annotation data to learn a model that does not overfit to the available annotations, given the deeply personal nature of email significance. We will describe the details of the annotation data in Section 5.

We compare a number of ML prediction techniques, including Averaged Perceptron [24], Gradient Boosted Trees [13], and logistic regression. To form a set of useful signals for the model training task, we focus on featurizing actions that have been performed on the emails, including:

- Action unigrams – occurrences of actions in isolation.
- Action bigrams – occurrences of pairs of actions observed for an email. While unigrams reflect the occurrence of each type of action, bigrams may capture specific action sequences that may indicate higher significance of emails (e.g. Flag followed by Reply).
- Time information features including: the total time spent on reading; time to reply; time to read; and time to forward.
- The total number of actions.

We note that we only use action-related features for approximating email significance in this section. As discussed in Section 1, having an action summary function derived purely from user actions can benefit the scalability of large production mail systems. Therefore, features such as the historical interactions between sender and receivers or features that require content analysis, although they could be highly useful, should be investigated separately.

The two methods, WAU and ML, are complementary and together provide a better understanding of how actions can indicate significance. Inferring significance using ML bypasses the need to design a summary function and can potentially result in better predictions when interactions between actions are hidden. However, it is important for production systems to have access to an unsupervised or semi-supervised rule-based "labeler" additionally to create training labels for downstream tasks, and this is where WAU can come into play. Ultimately, the performance of supervised ML predictors will be determined by the scale of data available for training, which for human-annotated emails is generally going to

be small-scale and expensive to obtain. Although WAU still poses a few parameters, the simplicity of the formula makes it easy for the weights to be decided elsewhere or heuristically, and also offers high interpretability. A final remark on WAU and ML methods is that they cannot be applied at the time when emails are delivered because at that time no actions are observed. It is similar to web search in that inferring document relevance for a user can only happen after their dwells and clicks. We demonstrate how to apply an action summary in an online prediction task in Section 7.

5 PEOPLE'S PERCEIVED SIGNIFICANCE

Although log data provides implicit characterization of attention utility, people's judgments of the significance of their emails are more valuable for deriving insight. Moreover, as we believe significance is better studied in a continuous spectrum, the fact that current major email clients support only a coarse dichotomy due to binary UI controls is sub-optimal. We developed a human intelligence task (HIT) survey to address these limitations.

The resulting data from HIT can be used as a reference for evaluating action summaries. In the case of an ML action summary, the judgments can also be used as labels in the training phase.

5.1 Email Significance HIT Survey

The email significance HIT survey is a user-email survey that lets people identify or label the absolute significance of individual emails. Each HIT chooses a random email from the most recent 200 emails delivered within the last two months to the user's mailbox, and shows them a rendering of the email's From, Subject, and Body content. It asks them to rate the email by its significance to them at the time they first read it. At the same time, it asks them to select one or more reasons for their decision. Up to 100 emails can be annotated by each person.

Although we believe significance to be a continuous-valued property, for ease of annotation we used a unipolar 5-point Likert scale design. The significance labels ranged from "Insignificant" to "Extremely significant". While the users were seeing their own email messages, we did not record any content of the email. We only collected their significance labels, reasons, and sufficient identifying metadata to let us match the labeled email with its action log record. In addition, we provided users of the HIT survey with the ability to skip any email they felt uncomfortable providing data on to us. While we understand that this might potentially introduce a selection bias towards particular emails, the number of emails that were rated by the users could reduce this bias if any.

Our task guidelines emphasized that rating significance was a personal decision by the participant. We provided suggestions for how to distinguish between different ratings as follows, though not all conditions might exist to select a rating. An "Extremely significant" email is one that communicates essential or important news, or an important and urgent task at work, that the person needed to give their immediate attention to. A "Very significant" email is one that the person would like to pay attention to within the next hour or that the person would return to on multiple occasions within a week after receiving it. A "Moderately significant" email is one the person would like to pay attention to within the next day from receiving it or that they return to at least once. A "Slightly significant" email is one the person would like to pay attention within a week of receiving that email, possibly read or skim the email once, but is unlikely to revisit it again. An "Insignificant" email is an email that the user usually ignored or did not even wish to read.

In the HIT survey we asked people to provide reasons they gave their ratings. The list of reasons we provided to the HIT survey participants includes: Email was from an important person; an Important topic; Part of a conversation I started; Requesting information from me; Information I needed to act on; Of interest but no immediate action; From an organization I often read; I did not have time to pay attention to it; About something I am uninterested in; I usually ignore this sender; and Other. We provided a free-form text field for Other, in case none of the suggested reasons were appropriate. People could select as many reasons as they liked.

5.2 Actions and Perceived Significance

We distributed the survey to both work and personal email participants. For personal emails, we had 118 participants or "judges" and 5774 hits or "judgments". The process of judge selection was managed in a way where bias was minimized, although inevitably they are required to an Outlook user to begin with. Figure 3 shows the distribution of actions (i.e., the percentage of the total number of actions for emails as was ranked "Insignificant" to "Extremely significant"). The most common actions performed on personal emails in all the significance classes are: Read (long and short) and Open an Attachment. A Short Read is one that is 2 seconds or less, to account for read actions that occur when a person is clicking quickly through a list of email, but not stopping to read the content in depth. Long Reads are all reads longer than Short Reads. Separating Short and Long Reads is only one way that attempts to address the complexity of the reading behavior, which is intrinsically affected by different contextual cues such as email length. Both Short and Long Reads are observed at similar rates across different levels of significance. Nevertheless, there is an increase for Open an Attachment actions with the increase of significance, and an increase of the Delete action with decrease of significance. One interesting thing we notice is that Reply and Reply-all have limited impact on how significant an email is. This outcome confirms the results of Dabbish et al. [8] from a study in which they found that the need to respond is only one part of defining the importance of an email message and that people respond to information requests or social messages, even though these messages were unimportant, compared to the work-related messages.

For work emails, we had 24 judges and 560 hits. Figure 4 shows that the distribution of actions for these emails is quite different than their distribution in personal emails shown in Figure 3, as well as the relationship between these actions and the significance of emails. Unlike personal emails, Reply and Reply-all are important actions for work emails. We see an increase of these actions with the increase of the significance of emails.

6 RESULTS AND DISCUSSION

For both WAU and ML predictors, we conduct predictions in two ways. The first casts significance prediction as a binary classification problem, where the outcome directly classifies emails into

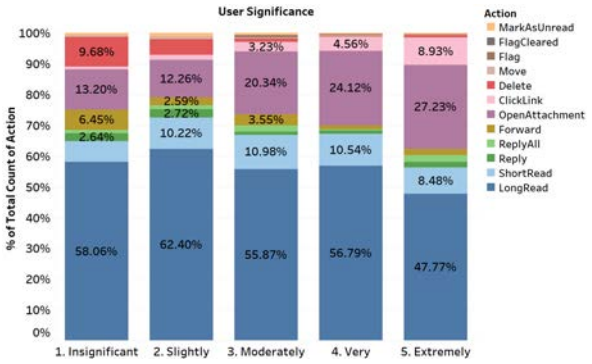


Figure 3: Distribution of actions for personal emails over users' significance

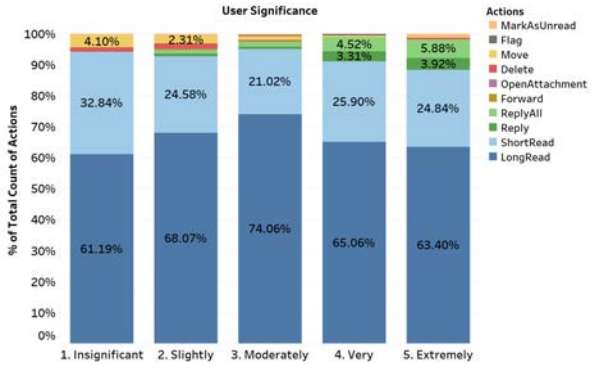


Figure 4: Distribution of actions for work emails over users' significance

positive and negative classes of significance. This task mimics what current solutions popular email services provide today. The second considers predicting the graded significance levels obtained from the HIT Survey data, based on which a partial ordering or even ranking over the emails can be formed.

6.1 Binary Classification

We organize our five significance labels in two different ways for binary classification:

- Binary Classification Low (BC-Low): positive label includes all significant labels, "Extremely", "Very", "Moderately", and "Slightly" significant. The negative class only includes the "Insignificant" label. This division addresses scenarios where missing out any slightly important email may result in a high penalty.
- Binary Classification High (BC-High): positive class includes "Extremely" and "Very" significant labels, and the negative class includes "Insignificant", "Slightly", and "Moderately" significant labels. This division addresses scenarios where only the top significant emails should draw users' attention.

For our baseline analysis, we classify our emails into emails that have at least one Reply or Reply-all as positive and emails that do not as negative. We choose this reply-only baseline because it

Table 1: Precision (P), recall (R), and AUC, for our baseline analysis using both BC-Low and BC-High

	Personal			Work		
BC Type	**P**	**R**	**AUC**	**P**	**R**	**AUC**
BC-Low	0.895	0.023	0.509	1.000	0.097	0.548
BC-High	0.416	0.027	0.507	0.625	0.271	0.619

is widely adopted as a standard notion for approximating email importance [32]. Table 1 shows the precision, recall and AUC for our baseline analysis for both work and personal emails. Overall the AUC for both email types is around 0.5, which is far from ideal. There is one exception for work emails using BC-High (0.619), suggesting that under certain contexts this reply-only predictor can work well, which is consistent with previous findings. The reason why AUC is higher in BC-High compared to BC-Low is that, looking at Figure 4, emails rated "Very" and "Extremely" significant have more Reply and Reply-all actions than the other emails. One drawback of this simple predictor, however, is that it cannot account for 30% of our judges who have no Reply actions at all on the emails that they judged, even though the emails were in fact assigned with different significance labels.

6.1.1 Using Rule-Based WAU. We compute the relationship between the WAU value and the user significance label. We consider the two types of WAU computed using the authors' and crowd-sourced weights. We normalize WAU values using a generalized logistic function. We choose different thresholds (0.1, 0.2, ..., 0.9) to divide the WAU value, where everything beyond the threshold is predicted as positive and everything below is negative. We did not do a train-test split to get an optimal cut because it was important for us to understand the impact of different thresholds.

Table 2 and Table 3 show the prediction results respectively for the tasks BC-Low and BC-High. In both tables, we are only showing 0.4, 0.5, and 0.6 thresholds, due to space limits, and to the fact that extreme thresholds tend to result in more skewed performance (e.g. high precision, low recall). In general, BC-Low appears to be an easier task compared to BC-High (higher precision and AUC values). Among all the thresholds we test, we are getting the highest AUC for both personal and work emails at threshold 0.6. Using BC-Low, the WAU values computed using author's weights can better predict users' significance than the ones computed using crowd-sourced weights in personal emails, and the crowd-sourced weights worked better for work emails. This finding suggests that the same set of weights might not be appropriate to be applied for both types of emails, since personal and work emails tend to have very different action distributions.

6.1.2 Using Machine-Learned Models. We form training and test sets split using 5-fold cross validation on personal and work emails respectively and use McNemar's Test for significance assessment between classifiers. Previous work [33, 34] suggested that personalization is a key factor for improving performance of generically learned models. To study the effects of inter-person variance, when forming the training and test sets, we split the data by either emails or by judges. In both cases, there are no overlapped instances between the splits, and we hypothesize splitting by judges is a more

Table 2: Precision (P), recall (R), and AUC, for authors (A) and crowd-sourced (CS) WAU weights(W), using BC-Low with 0.4, 0.5, and 0.6 thresholds (Th)

		Personal			Work		
W	**Th**	**P**	**R**	**AUC**	**P**	**R**	**AUC**
A	**0.4**	0.772	0.991	0.510	0.735	0.995	0.510
	0.5	0.780	0.914	0.530	0.733	0.904	0.502
	0.6	0.905	0.493	0.661	0.863	0.324	0.592
CS	**0.4**	0.621	1.000	0.500	0.731	1.000	0.500
	0.5	0.621	1.000	0.500	0.731	1.000	0.500
	0.6	0.851	0.282	0.601	0.793	0.723	0.610

Table 3: Precision (P), recall (R), and AUC, for authors (A) and crowd-sourced (CS) WAU weights(W), using BC-High with 0.4, 0.5, and 0.6 thresholds (Th)

		Personal			Work		
W	**Th**	**P**	**R**	**AUC**	**P**	**R**	**AUC**
A	**0.4**	0.123	1.000	0.508	0.165	1.000	0.510
	0.5	0.131	0.972	0.541	0.171	0.945	0.530
	0.6	0.212	0.729	0.677	0.240	0.402	0.580
CS	**0.4**	0.248	1.000	0.500	0.164	1.000	0.500
	0.5	0.248	1.000	0.500	0.164	1.000	0.500
	0.6	0.410	0.340	0.590	0.182	0.7391	0.543

challenging task due to different personal behavior. We experiment with both linear and tree learners, including Averaged Perceptron [24], Boosted Trees [13] and LightGBM [17] for binary classification using off-the-shelf ML libraries. In the following analysis, we present results with Boosted Trees only due to space limit.

Table 4 and Table 5 show the prediction results for the tasks BC-Low and BC-High. Compared to Table 2, the performance of boosted trees is in general better than that of fixed cut-off WAU in terms of AUC with $p\text{-}value < 0.01$, especially for work emails. This highlights that, while WAU describes a good summary of action distributions, the flexibility of incorporating different dimensions of action metadata, such as action sequences and reading time, can further help significance prediction. The same observation can be found in the task BC-High by comparing Tables 3 and 5.

As expected, cross validation with splits by judges tends to be a harder problem, where AUC decreases between 3% – 15% compared to splits by emails. This observation opens up an interesting research question on how we could use personalization strategies for associating actions and significance, which is beyond the scope of this paper and left for future work.

6.2 Multi-Grade Predictions

Next, we present our findings on prediction tasks where multi-grade significance labels are considered. We examine two prediction strategies. Firstly we form a ranking on emails using the continuous WAU, and compare it with the ranked list based on the five classes of significance collected from the HIT Survey. The second strategy casts the prediction problem as a multi-class classification

Table 4: Precision (P), recall (R), and AUC based on Boosted Trees. Average results are reported with 5-fold cross validation split by emails or by judges, for BC-Low

	Personal			Work		
Split By	**P**	**R**	**AUC**	**P**	**R**	**AUC**
Emails	0.622	0.994	0.643	0.793	0.911	0.698
Judges	0.611	0.983	0.638	0.696	0.900	0.666

Table 5: Precision (P), recall (R), and AUC based on Boosted Trees. Average results are reported with 5-fold cross validation split by emails or by judges, for BC-High

	Personal			Work		
Split By	**P**	**R**	**AUC**	**P**	**R**	**AUC**
Emails	0.603	0.157	0.643	0.517	0.212	0.690
Judges	0.468	0.157	0.626	0.300	0.203	0.600

task, where the outcome aims to distinguish emails into the five significance classes.

6.2.1 Using Rule-Based WAU. We first use the WAU value to order the set of annotated emails. Then we compute Spearman's Rho rank correlation coefficient (with correction for ties) between the ordering formed by WAU and the ordering formed by the five classes of users' significance. Both authors' and crowd-sourced weights are considered. Multi-graded prediction is challenging, thus, as we expected, we get low correlation. For both personal and work emails, we get a correlation that ranges from 0.250 to 0.273 using the two different weights. With binary classification results presented in Tables 2 and 3, we are getting better predictions and correlation with user significance, compared to using the 5-class labels. Nevertheless, computing the correlation per judge (discussed in detail in Section 6.3.2), we observe that some judges have high correlation and others have low correlation, supporting our view that perceptions of significance may be highly personal.

6.2.2 Using Machine-Learned Models. Similar to Section 6.1.2, we conduct 5-fold cross validation and split the data by emails or judges. For the learners, we consider again both linear (multi-class logistic regression) and tree (LightGBM) learners. Micro-accuracy and macro-accuracy are two standard metrics for evaluating multi-class classification effectiveness. Micro-accuracy is defined as the ratio of the number of correctly predicted instances to the total number of instances; this metric is more robust towards class size imbalance problems. Macro-accuracy first computes accuracy for each class and reports the average of per-class accuracies. For our task, we first examine micro-accuracy and macro-accuracy in each fold, take the average of the 5 folds, and report the results in Table 6. Spearman's Rho is additionally computed based on the rankings formed by labels and by predicted classes, so as to capture the correlation between the two. Table 6 shows that predicting graded significance tends to be simpler for personal than work emails. This could be because the action distributions appear more distinguishable across different significance classes in personal than in work data, as suggested by Figures 3 and 4. Compared to the results presented in Section 6.2.1, we see that predicting significance with ML models

Table 6: Micro-avg and macro-avg accuracy and Spearman's Rho results based on multi-class logistic regression.

	Personal			Work		
Split By	**Micro**	**Macro**	**Rho**	**Micro**	**Macro**	**Rho**
Emails	0.413	0.275	0.315	0.369	0.279	0.295
Judges	0.391	0.245	0.265	0.308	0.241	0.177

results in higher correlation with the ground truth rankings for data splits by emails, i.e., Rho=0.315 and Rho=0.295 for personal and work respectively. However, when eliminating personal interactions (i.e., split by judges), the correlation is either on par or worse than using WAU. This outcome may suggest that an evaluation metric which focuses on ranking, i.e. Rho, is more heavily impacted by per-person variance.

6.3 Discussion

While we have demonstrated the potential of summarizing actions to predict email significance, based on both WAU and ML, we observe some challenges which are discussed below.

6.3.1 Differentiating perceived significance. Differentiating perceived significance is challenging in general. As mentioned in Section 5.1, we asked our participants to not only judge and rank the significance of their emails, but to also give us feedback on the reasons for their judgments. While there is usually a clear distinction between "Insignificant" and "Extremely" significant, the distinction between "Slightly", "Moderately", and "Very" significant can be more subtle and highly personal. Asking for participants' feedback was one way to make that distinction clearer for us. Figure 5 shows the distribution of the feedback our participants gave in comparison to their email significance ranking for work emails (personal emails have similar distribution). In both types of emails, the reasons the participants gave for the emails they rated "Extremely" are different than the ones rated as "Insignificant". For "Insignificant" rated emails, we notice that the reasons mostly included: something they are not interested in, they usually ignore the sender, or did not have time to pay attention to it. However, for "Extremely" rated emails, the reasons were typically: an important topic or from an important person, or information they needed to act on. The reasons our participants gave for "Very" and "Extremely" rated emails were similar. This similarity could help explain why we obtain better correlation using the binary classification approach. To some extent, this may also explain why the correlation results presented in Section 6.2 based on correlation using the multi-graded labels are worse in comparison to the binary labels in Section 6.1.

Identifying the same level of perceived significance can be challenging even for the same judge on the same email. A subset of emails were judged repeatedly by the personal judges in the HIT survey. In total 998 unique emails were judged multiple times by 101 judges, with each email being displayed on average 4 times. Note that for all the results presented in Section 6, we remove the duplicates and only experiment with those with a single judgment.

To quantify the degree of inconsistency, we calculate entropy for each unique email based on per-significance distribution using

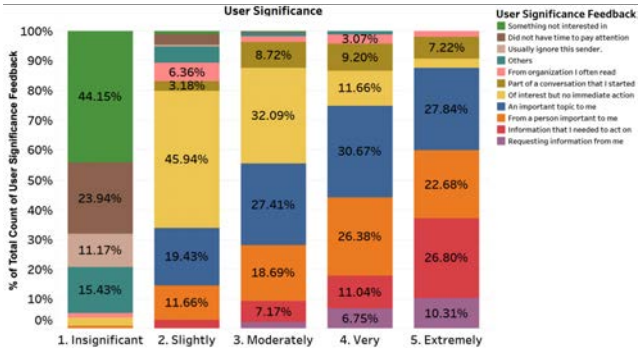


Figure 5: Explanations for the five email significance labels ranked by the HIT Survey participants

$H(x) = -\sum_{s=1}^{5} P(x_s) log_2 P(x_s)$, where s indicates email significance level[2]. If the outcome is certain (i.e. no inconsistency) we would expect entropy of 0. In fact, we see the average entropy is 0.35 for this data, and gets even higher, i.e., 0.41, when the number of repetitions is greater than or equal to 5. This suggests that email owners may have difficulties giving consistent judgments. Scholer et al. [27] also found high rates of intra-judge search relevance judging inconsistency of between 15% and 24% across a range of TREC test collections.

6.3.2 Email significance is highly personal. Another challenge we face is that the significance of emails varies based on the user. To better understand the correlation between our WAU significance values and the judged significance, we compute the correlation for each person. Figure 6 shows the correlation for each of the personal email users in our sample data in comparison to the number of hits. We observe that the number of emails the people are judging has no impact on the the correlation. Some people have a high correlation and others have low correlation. More than half of the people have positive correlation, but some of them have no or negative correlation. The large number of zero correlation scores is due to the tie-adjusted Rho computation.

6.3.3 Action Limitations. Looking at our data, both the log data and the HIT Survey labels, we observed two limitations. The same set of actions performed on emails (e.g. emails that have Long Read and Move) can be rated by the HIT Survey participants with different levels of email significance. That is a major challenge given that we depend on actions as features to predict email significance. These set of actions that have different levels of significance vary in terms of both the numbers of this happening and the different labels assigned to them by the judges. We computed the entropy to better understand the variance of each of the actions sets we have and the judges' assigned labels. We noticed that some of the emails that have common action sets (e.g. Long Read, Short Read) have been labeled "Insignificant" to "Extremely" significant by different users, thus have higher entropy (1.5-2). However, the emails that have a unique set of actions were judged using only one label.

[2] $0log0$ is defined as 0 in information theory.

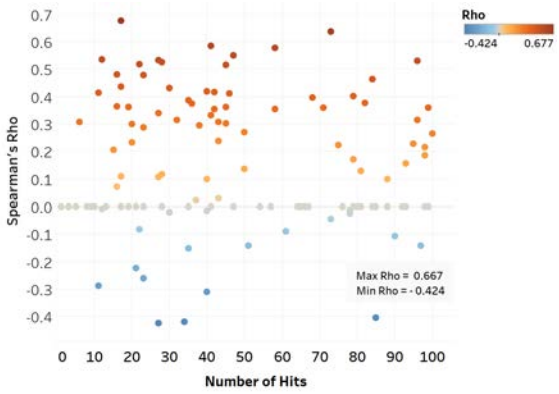


Figure 6: Spearman's Rho for each personal email user over the number of HIT Survey hits

7 REAL-TIME SIGNIFICANCE PREDICTION

Above, We studied how and to what extent user actions on emails can be harnessed as a proxy for email significance. We demonstrated that different action summary functions correlate with significance in varying degrees and overall the correlation is positive. To apply our learning to a practical scenario, we train a machine-learned model based on large scale data for real-time significance prediction, as shown in part (B) of Figure 1. Specifically a regression model is trained to fit the output from an action summary function, which is effectively used as the training label. Then we extract a set of non-action features that can be obtained at the time when an email arrives at a mailbox. In this way, the resulting predictive model can forecast email significance in real-time (i.e., email arrival time) because actions are needed only for offline training.

7.1 Prediction Model

Our goal is to have a predictor that is able to predict email significance in real-time. In an ideal world, a supervised predictor can be trained to reach oracle performance if human significance labels can be obtained at scale at a low cost. However, obtaining large amount of human assessments is impractical in practice; instead leveraging implicit feedback from users as a proxy is commonly a preferred, scalable way. In the search world, one of the most successful examples is to treat users' actions including clicks and dwell-time on documents in response to user queries as positive feedback, and thus the corresponding documents are deemed relevant for training [14]. For email significance prediction, we propose to use the output of action summary functions as implicit significance feedback. In this section, we choose to experiment with the rule-based summarizer WAU as the training label, as we have shown that WAU outperforms a reply-only baseline. Although a machine-learned summary function can be more correlated with significance, the amount of human annotation training data heavily determines the performance of the outcome. To focus on presenting the feasibility of building a real-time predictor, a rule-based summarizer is preferred as it provides high simplicity which in turn benefits production systems.

7.1.1 Model training. We reuse the large-scale log samples described in Section 3 to train two real-time predictors respectively for personal and work accounts. In particular, each of the log sample represents an email e_i. We first compute for each e_i its WAU value as the training label y_i based on the actions performed by actual users. Then we extract a number of non-action features $\boldsymbol{x_i}$, resulting in a dataset of $\{(y_i, \boldsymbol{x_i})\}$. We cast the prediction task as a regression problem with an objective that minimizes the $L2$ distance between the prediction $\hat{y_i}$ and y_i. We adopt LightGBM [17] for regression using an off-the-shelf ML library for training. Two held-out datasets (from different users and time periods) in the same format as the samples are added for validation and parameter selection. Parameters on the number of boosting iterations $\{50, 100, 150, 500, 1000\}$, learning rates, and minimum number of items in leaves $\{10, 20, 50\}$ are swept. Training is done on standard CPU machines without parallelization.

7.1.2 Hand crafted email property features. We adopt a total number of 35 non-action features that may reflect email significance. Not all features are available for both personal and work email; we simply impute 0 for missing values. The features are designed in a way that their values can be extracted during email delivery time, which in turn enables real-time significance prediction.

- Sender-related. Binary features that capture if the sender is of high impact, such as if a sender is in the address book or if a sender is more senior in the management chain.
- Recipient-related. Features that capture if the recipient receives an email as a user on the To, CC or BCC field.
- Historical interactions between sender and recipient. Features that reflect in the past how often the sender and recipient interact with each other (e.g., email read rate).
- Binary classifiers. Output from in-house classifiers that predict if the content of an email may belong to a category (i.e., a newsletter, a promotion or a purchase).

We note that any future user action performed on a current email is not considered in this feature list. That is, there is no leak or overlap between the features on email properties and the actual actions that may be performed on this current email at a future time. We should also emphasize that the focus of this section is not to develop a comprehensive set of email property features, but instead to show how online prediction is possible by leveraging output from action summary functions as supervision for ML.

7.2 Evaluation

We compare the real-time prediction $\hat{y_i}$ and the ground truth label $L(e_i)$ on both the personal and work human judgment sets. One way is to compute Spearman's Rho between the rankings formed by predictions and labels as is done in Section 6.2. On the personal set the Rho value is 0.265, and on the work set, it is 0.370. Compared to the results in Section 6.2, we see overall an increase in the correlation for both data sets. One reason for this improvement is that the real-time prediction models tend to create more continuity in the predictions than WAU and the action-features-only models. This continuity breaks ties for emails of similar user responses; the direction of separation usually aligns with the judgments.

In addition to Rho, we also report Krippendorf's α [20] for evaluation on pairwise data formed from the human judgment sets. In

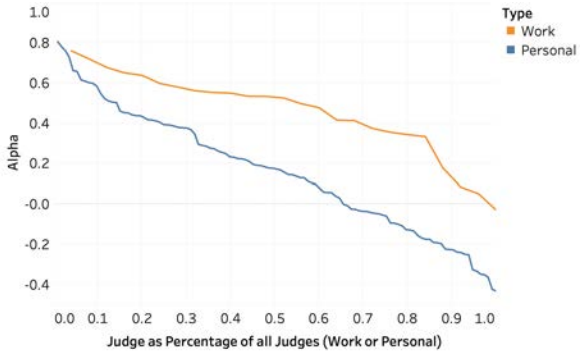


Figure 7: Krippendorf's α for each personal and work email user over their email pairs for significance labels and model prediction scores. The x-axis is the pool size normalized to lie between 0 and 1; agreement rates decline faster with personal judges.

the search literature, the amount of quality relevance assessments is key to training an effective ranker. To increase the amount of training data, a common approach is to permute pairs of labelled observations (within some group, such as a query) and incorporate pairwise comparisons in the objective function for optimization (e.g. [5]). In our work, although our goal is not to facilitate learning-to-rank training, we leverage pairwise data generated from the human sets to obtain more observations. Based on that, we can provide evaluation in more detail using Krippendorf's α.

Krippendorf's α is commonly used for assessing agreements between target and reference distributions for various data types (e.g., ordinal, interval and ratio), and deals with ties and missing values naturally. Given a pair of emails $(e_i, e_j), i \neq j$, our goal is to quantify how often the model predictions agree with ground truth ordering. To this end, α is used to assess the degree of consistency between the difference in model scores, $M(e_i) - M(e_j)$, and the difference in human significance labels, $L(e_i) - L(e_j)$. Consistency is higher when the difference in model scores resembles that in labels. For example, when a user regards e_i to be "Very significant" (4) and e_j to be "Insignificant" (1), a model that gives a large positive difference in predictive scores will result in high consistency. The α scores range from -1 (complete anti-agreement) to 1 (complete agreement), where the middle point 0 is effectively random agreement.

We calculate α as the agreement measure on both personal and work judgment data, for all possible pairs of emails belonging to each person. Values for α in aggregate are 0.131 and 0.339 respectively. Overall, we can see that the work data has better agreement in ordering than the personal data, which is consistent with the Rho results. In neither case do the scores show strong levels of agreement; indeed for the personal email corpus it is close to random levels of agreement. (As a comparison, inter-rater agreement for search relevance has also been found to vary considerably, ranging from lows of 0.15 in one study of 180 raters [26] to highs of 0.557–0.705 (depending on condition and scale) in another study involving 16 raters [9], both involving university students.) Search relevance is a challenging labeling task for judges and email significance is also hard for models to predict, based on these outcomes.

In addition to the overall α, we calculate α values on a per person basis. Figure 7 shows these sorted by decreasing α values, and normalized (as a percentage of the pool size) in the x-axis to allow easier comparison between personal and work judges. We see as previously that there is a wide range of agreement according to individuals. Approximately 20% of the work judges and 10% of the personal judges have α values above the 0.667 threshold, which is considered "possibly reliable" for agreement. In the personal case, around a third of the judges start to be anti-correlated. All these findings suggest that, for future work, studying user engagement in depth in personal emails will be important. Indeed, Cecchinato et al [7] found that in personal accounts, people usually looked for order confirmations, travel or money related emails etc. For these cases, it is hard to separate observed user engagement such as reading from other less significant emails such as promotions, given that people also read promotions.

8 CONCLUSIONS

Managing email flow is something that people carry out often. Some emails require more attention than others, and hence, different actions are performed on them accordingly. We investigated our hypothesis that the cumulative set of actions on an email can be considered as a proxy for people's perceived significance of the email using two approaches. The rule-based approach WAU was simple and made better summaries for significance compared to a reply-only baseline; meanwhile the machine-learned approach can more effectively summarize actions and predict email significance.

We found that there are differences in actions between significant and insignificant emails in aggregate. Our analysis also showed that the degree of correlation between actions and significance varies across personal and work accounts as well as across people. Nevertheless, email processing is a much more complex task than traditional web search, in which mapping actions to relevance in search has been shown to have a strong correlation. Thus we carried out more analysis and discussed a number of challenges in our framework, including the fact that the same sets of actions can be observed for emails from different classes of significance. Overall, our results suggest that the cumulative set of actions on any email can be treated as a partial proxy for the email's perceived significance, at least for some people or account types.

Although action summaries are not perfectly correlated with human significance judgments, using actions as a proxy still has potential to be used in predicting significance on emails in real-time, which is similar to modeling document relevance using clicks and dwell time in the context of web search. Specifically, we presented an example of using a rule-based action summary function, WAU, as supervision for training an email significance predictor based on large-scale email action samples, where the resulting predictor can be executed upon email arrival. Evaluation of this predictor showed reasonable correlation with human assessments, more so for work accounts.

Our work provides insights on how the different actions performed on emails can have an impact on the significance of these emails, and how to apply the findings to create a real-time predictor for email prioritization. Future work is required to understand how

the personalized nature of significance can be more accurately modeled, including by collecting substantially more annotation data, conducting deeper feature engineering and analysis, or performing *in situ* surveys.

ACKNOWLEDGEMENTS

We gratefully thank Kshitiz Gupta for his help in creating the HIT survey codebase and the anonymous personal and work judges for their judging work.

REFERENCES

[1] D. Aberdeen, O. Pacovsky, and A. Slater. 2010. The learning behind Gmail Priority Inbox. In *LCCC: NIPS 2010 Workshop on Learning on Cores, Clusters and Clouds*.
[2] E. Agichtein, E. Brill, and S. Dumais. 2006. Improving web search ranking by incorporating user behavior information. In *Proc. SIGIR*. 19–26.
[3] T. Alrashed, A H. Awadallah, and S. Dumais. 2018. The lifetime of email messages: a large-scale analysis of email revisitation. In *Proc. CHIIR*. 120–129.
[4] O. Bälter and C. L. Sidner. 2002. Bifrost Inbox Organizer: Giving users control over the inbox. In *Proceedings of the second Nordic conference on Human-Computer Interaction*. 111–118.
[5] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. 2005. Learning to rank using gradient descent. In *Proc. ICML*. 89–96.
[6] R. Capra, J. Khanova, and S. Ramdeen. 2013. Work and personal e-mail use by university employees: PIM practices across domain boundaries. *JASIST* 64, 5 (2013), 1029–1044.
[7] M. E. Cecchinato, A. Sellen, M. Shokouhi, and G. Smyth. 2016. Finding Email in a Multi-Account, Multi-Device World. In *Proc. CHI*. 1200–1210.
[8] L. A. Dabbish, R. E. Kraut, S. Fussell, and S. Kiesler. 2005. Understanding email use: predicting action on a message. In *Proc. CHI*. 691–700.
[9] T. T. Damessie, F. Scholer, K. Järvelin, and J. S. Culpepper. 2016. The effect of document order and topic difficulty on assessor agreement. In *Proc. ICTIR*. 73–76.
[10] D. Di Castro, Z. Karnin, L. Lewin-Eytan, and Y. Maarek. 2016. You've got mail, and here is what you could do with it!: Analyzing and predicting actions on email messages. In *Proc. WSDM*. 307–316.
[11] N. Ducheneaut and V. Bellotti. 2001. E-mail as habitat: an exploration of embedded personal information management. *Interactions* 8, 5 (2001), 30–38.
[12] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White. 2005. Evaluating implicit measures to improve web search. *ACM TOIS* 23, 2 (2005), 147–168.
[13] J. H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of Statistics* (2001), 1189–1232.
[14] J. Huang, R W. White, and S. Dumais. 2011. No clicks, no problem: using cursor movements to understand and improve search. In *Proc. CHI*. 1225–1234.
[15] K.M. Huessner. 2010. Tech stress: How many emails can you handle a day. Retrieved 14 August, 2018 from https://abcnews.go.com/Technology/tech-stress-emails-handle-day/story?id=11201183
[16] T. Joachims. 2002. Optimizing search engines using clickthrough data. In *Proc. KDD*. 133–142.
[17] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. In *Proc. NIPS*. 3146–3154.
[18] J Y. Kim, N. Craswell, S. Dumais, F. Radlinski, and F. Liu. 2017. Understanding and Modeling Success in Email Search. In *Proc. SIGIR*. 265–274.
[19] R. E. Kraut, S. Sunder, R. Telang, and J. Morris. 2005. Pricing electronic mail to solve the problem of spam. *Human-Computer Interaction* 20, 1 (2005), 195–223.
[20] K. Krippendorff. 2004. *Content analysis: An introduction to its methodology. Second Edition*. Sage.
[21] C. Neustaedter, A.J. Bernheim Brush, M. A. Smith, and D. Fisher. 2005. The Social Network and Relationship Finder: Social Sorting for Email Triage. In *CEAS*.
[22] S. Radicati. 2014. Email Statistics Report, 2014-2018. Retrieved 14 August, 2018 from http://www.radicati.com/wp/wp-content/uploads/2014/01/Email-Statistics-Report-2014-2018-Executive-Summary.pdf
[23] B. Reeves, S. Roy, B. Gorman, and T. Morley. 2008. A marketplace for attention: Responses to a synthetic currency used to signal information importance in e-mail. *First Monday* 13, 5 (2008).
[24] Frank Rosenblatt. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review* 65, 6 (1958), 386.
[25] B. Sarrafzadeh, A.H. Awadallah, C. Lin, C-J. Lee, and M Shokouhi. 2019. Characterizing and Predicting Email Deferral Behavior. In *Proc. WSDM*.
[26] P. Schaer. 2012. Better than their reputation? on the reliability of relevance assessments with students. In *International Conference of the Cross-Language Evaluation Forum for European Languages*. Springer, 124–135.
[27] F. Scholer, A. Turpin, and M. Sanderson. 2011. Quantifying test collection quality based on the consistency of relevance judgements. In *Proc. SIGIR*. 1063–1072.
[28] N. Siu, L. Iverson, and A. Tang. 2006. Going with the flow: email awareness and task management. In *Proc. CSCW*. 441–450.
[29] G. Venolia, L. Dabbish, J.J. Cadiz, and A. Gupta. 2001. *Supporting email workflow*. Technical Report MSR-TR-2001-88. Microsoft Research.
[30] J. Wainer, L. Dabbish, and R. Kraut. 2011. Should I open this email?: inbox-level cues, curiosity and attention to email. In *Proc. CHI*. 3439–3448.
[31] S. Whittaker and C. Sidner. 1996. Email overload: exploring personal information management of email. In *Proc. CHI*. 276–283.
[32] L. Yang, S. T. Dumais, P. N. Bennett, and A. H. Awadallah. 2017. Characterizing and predicting enterprise email reply behavior. In *Proc. SIGIR*. 235–244.
[33] S. Yoo, Y. Yang, and J. Carbonell. 2011. Modeling personalized email prioritization: classification-based and regression-based approaches. In *Proc. CIKM*. 729–738.
[34] S. Yoo, Y. Yang, F. Lin, and I.-C. Moon. 2009. Mining social networks for personalized email prioritization. In *Proc. KDD*. 967–976.

DDGK: Learning Graph Representations for Deep Divergence Graph Kernels

Rami Al-Rfou
Google AI
Mountain View, CA
rmyeid@google.com

Dustin Zelle
Google AI
New York, NY
dzelle@google.com

Bryan Perozzi
Google AI
New York, NY
bperozzi@acm.org

ABSTRACT

Can neural networks learn to compare graphs without feature engineering? In this paper, we show that it is possible to learn representations for graph similarity with neither domain knowledge nor supervision (i.e. feature engineering or labeled graphs). We propose Deep Divergence Graph Kernels, an unsupervised method for learning representations over graphs that encodes a relaxed notion of graph isomorphism. Our method consists of three parts. First, we learn an encoder for each anchor graph to capture its structure. Second, for each pair of graphs, we train a cross-graph attention network which uses the node representations of an anchor graph to reconstruct another graph. This approach, which we call *isomorphism attention*, captures how well the representations of one graph can encode another. We use the attention-augmented encoder's predictions to define a divergence score for each pair of graphs. Finally, we construct an embedding space for all graphs using these pair-wise divergence scores.

Unlike previous work, much of which relies on 1) supervision, 2) domain specific knowledge (e.g. a reliance on Weisfeiler-Lehman kernels), and 3) known node alignment, our unsupervised method jointly learns node representations, graph representations, and an attention-based alignment between graphs.

Our experimental results show that Deep Divergence Graph Kernels can learn an unsupervised alignment between graphs, and that the learned representations achieve competitive results when used as features on a number of challenging graph classification tasks. Furthermore, we illustrate how the learned attention allows insight into the the alignment of sub-structures across graphs.

KEYWORDS

Graph Kernels; Graph Neural Networks; Representation Learning; Similarity and Search

ACM Reference Format:
Rami Al-Rfou, Dustin Zelle, and Bryan Perozzi. 2019. DDGK: Learning Graph Representations for Deep Divergence Graph Kernels. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313668

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313668

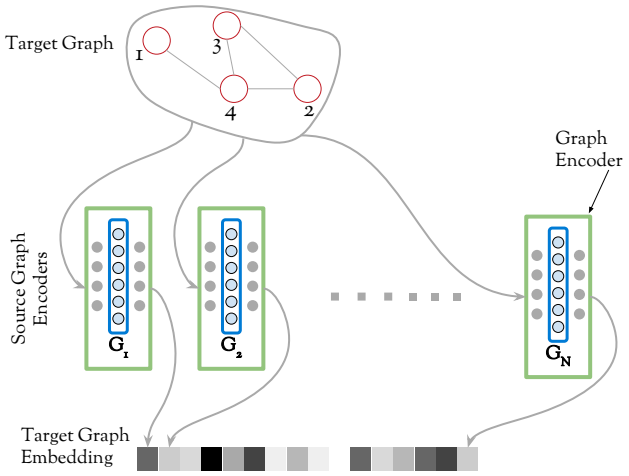


Figure 1: Our method of learning graph representations by measuring the divergence of a target graph across a population of source graph encoders. First, we train a graph encoder for each graph in our source graph population {G_1, G_2, ..., G_N}. Second, for each of these encoders we measure the divergence of the target graph from the associated source graph. Finally, these divergence scores are used to compose the vector representation of the target graph.

1 INTRODUCTION

Deep learning methods have achieved tremendous success in domains where the structure of the data is known a priori. For example domains like speech and language have intrinsic sequential structure to exploit, while computer vision applications have spatial structure (images) and perhaps temporal structure (videos). In all these cases, our intuition guides us to build models and learning algorithms based on the structure of the data. For example, translation invariant convolution networks might search for shapes regardless of their physical position in an image, or recurrent neural networks might share a common latent representation of a concept across distant time steps or diverse domains such as languages. In contrast, graph learning represents a more general class of problems because the structure of the data is free from any constraints. A neural network model must learn to solve both the desired task at hand (e.g. node classification) and to represent the structure of the problem itself – that of the graph's nodes, edges, attributes, and communities.

Despite the challenges, there has been a recent surge of interest in applying neural network models to such graph-structured data

[26, 30, 40, 50, 63]. While initial approaches like DeepWalk [40] focused on generic representations of graph primitives (e.g. a graph's nodes [40] or edges [1]), present approaches ignore learning general graph and node representations in favor of maximizing accuracy on a set of narrow classification tasks. These approaches, broadly referred to as Graph Neural Networks (GNNs), seek to leverage the structure between data items as a scaffolding to perform computation (e.g. message passing, gradient updates, etc). The parameters and the activations, use the structure during training, but are tuned primarily to classify the graph's nodes, edges, and/or attributes.

While much effort has focused on unsupervised learning of node representations [12, 24, 40, 48], edge representations [1], or latent community structure [13, 52, 62], relatively little work has focused on the unsupervised learning of representations for entire graphs – a problem of practical interest in domains such as information retrieval, biology, and natural language processing [5, 23]. In cases where GNNs have been applied to the task of learning similarity between graphs, the approaches considered generally come in two flavors: an end-to-end *supervised graph classification* or *graph representation learning*.

In supervised graph classification, the task is to solve an end-to-end whole-graph classification problem (i.e. the problem of assigning a label to the entire graph). These supervised approaches [33, 36, 45, 61] learn an intermediate representation of an entire graph as a precondition in order to solve the classification task. This learned representation can be used to compare similarity between graphs, but is heavily biased towards maximizing performance on the classification task of interest.

The second class of approaches focuses on the more general problem of learning graph representations [43]. While much exciting progress has been made in this area, the existing approaches suffer from one or more of the following limitations. First, many existing methods rely on feature engineering, such as the graph's clustering coefficient, its motif distribution, or its spectral decomposition, to represent graphs [6, 47, 56]. By limiting the features that they consider, these methods are limited to composing only known graph signals. Second, many of these approaches [36, 61] have sought to encode algorithmic heuristics from the graph isomorphism literature (especially the intuition encoded in the Weisfeiler Lehman algorithm [42]). Relying heavily on existing heuristics to solve a hard problem raises an important question: how well can a learning-only approach solve a classic algorithmic problem? Finally, other work in this area of graph similarity assumes that identical nodes in both graphs share the same id (i.e. the alignment is already given). While this can be useful for calculating a similarity score, we find the general problem more compelling.

In this work, we propose a method of learning graph representations driven by the similarity between a pair of graphs as measured by the divergence in their structures. We show the representations learned through our method, Deep Divergence Graph Kernels (DDGK), capture the attributes of graphs by using them as features for several classification problems. In addition, we show that our representations capture the local similarity of graph pairs and the global similarity across families of graphs.

DDGK has three key differentiators. First, it makes no assumptions about the structure of the matching problem. In order to solve the matching problem, we propose an attention mechanism: *isomorphism attention* to align the nodes across graph pairs. Second, DDGK does not rely on any existing heuristics for graph similarity. Instead, we learn the kernel method jointly with the node representation and alignment networks. This allows the model the freedom to learn representations that best preserve the graph, and does not impose artificial oversights. Finally, as an unsupervised method, the representations it learns emphasize structural similarity, and does not correlate with a downstream labeling tasks. This is especially useful for ranking tasks where labeling may not be available.
To summarize, our main contributions are:

- **Deep Divergence Graph Kernels**: A novel method for learning unsupervised representations of graphs and their nodes. Our kernel is learnable and does not depend on feature engineering or domain knowledge.
- **Isomorphism Attention**: A cross-graph attention mechanism to probabilisticly align representations of nodes between graph pairs. These attention networks allow for great interpretablity of graph structure and discoverablilty of similar substructures.
- **Experimental results**: We show that DDGK both encodes graph structure to distinguish families of graphs, and when used as features, the learned representations achieve competitive results on challenging graph classification problems like predicting the functional roles of proteins.

2 LEARNING GRAPH REPRESENTATIONS

In this section, we lay out the problem definition of representing graphs and the connection between our representations and the kernel framework.

2.1 Problem Definition

A graph is defined to be a tuple $G = (V, E)$, where V is the set of vertices and E is the set of edges, $E \subseteq (V \times V)$. A graph G can have an attribute vector Y for each of its nodes or edges. We denote the attributes of node v_i as y_i, and denote the attributes of an edge (v_i, v_j) as y_{ij}.

Given a family of graphs G_0, G_1, ..., G_N we aim to learn a continuous representation for each graph $\Psi(G) \in \mathbb{R}^N$ that encodes its attributes and its structure. For this representation to be useful, it has to be comparable to other graph representations. However, it is likely that our method of graph encoding will produce one of many equally good representations each time we run it. For example we can get two different, but equal, representations by permuting the dimensions of the first one. Those representations are not comparable given they exist in two different spaces.

To avoid this problem, we seek to develop an equivalence class across all possible encodings of a graph. Essentially, two encodings of a graph are equivalent if they lead to the same pair-wise similarity scores when used to compare the graph to all other graphs in the set. We note that this issue arises when working with embedding based representations across domains, and several equivalence methods have been proposed [3, 58].

2.2 Embedding Based Kernels

In this work, we study the development of graph kernels, which are functions to compute the pairwise similarity between graphs. Specifically, given two graphs G_1, G_2, a classic example of a kernel defined over graph pairs is the geometric random walk kernel [9] as shown in Eq. 1:

$$k_\times(G_1, G_2) = e^T (I - \lambda A_\times)^{-1} e, \tag{1}$$

where $A_\times$ is the adjacency matrix of the product graph of G_1 and G_2, and λ is a hyper-parameter which encodes the importance of each step in the random walk. We aim to learn an embedding based kernel function $k()$ as a similarity metric for graph pairs, defined as the following:

$$k(G_1, G_2) = ||\Psi(G_1) - \Psi(G_2)||^2 \tag{2}$$

For a dataset of N *source*[1] graphs $\mathcal{S}$ and M *target* graphs ($\mathcal{T}$), for any member of the target graph set we define the i^{th} dimension of the representation $\Psi(G \in \mathcal{T}) \in \mathbb{R}^N$ to be:

$$\Psi(G)_i = \sum_{v_j \in V_T} f_{g_i}(v_j), \tag{3}$$

where $g_i \in \mathcal{S}$ and $f_{g_i}()$ is a predictor of some structural property of the graph G but parameterized by the graph g_i. We note that the source and target graphs sets $(\mathcal{S}, \mathcal{T})$ could be disjoint, overlapping, or equal.

3 LEARNING TO ALIGN GRAPH REPRESENTATIONS

We propose to learn a graph representation by comparing it to a population of graphs. To compare the similarity of a pair of graphs (*source*, *target*), we rely on deep neural networks to measure the divergence between their structure and attributes. First, we learn the structure of the source graph by passing it through a graph encoder. Second, to measure how much the target graph diverges from the source graph, we use the source graph encoder to predict the structure of the target graph. If the pair is similar, we expect the source graph encoder to correctly predict the target graph's structure. In this section, we develop the three key components necessary to learn the similarity between a pair of graphs.

First, in Section 3.1, we discuss encoding graphs. The quality of the graph representation depends on the extent to which the encoder of each source graph is able to discover its structure.

Second, in Section 3.2, we propose a cross-graph attention mechanism to learn a soft alignment between graphs. This is necessary because a target graph may not share its vertex ids with any of the source graphs – indeed, they could even have differing number of nodes! Therefore, we need to learn an alignment between the nodes of the target graph and each source graph. This leads to an alignment that is not necessarily a one-to-one correspondence.

Third, in Section 3.3 we introduce additional constraints on the cross-graph attention learning. For example, let us assume that $v_i \in V_{G_1}$ is assigned to $u_j \in V_{G_2}$. While both v_i and u_j may be structurally similar, they may belong to different node classes as indicated by their attributes. These attributes may be of significant

[1]In this paper, we use source and anchor interchangeably when referring to the encoded graph.

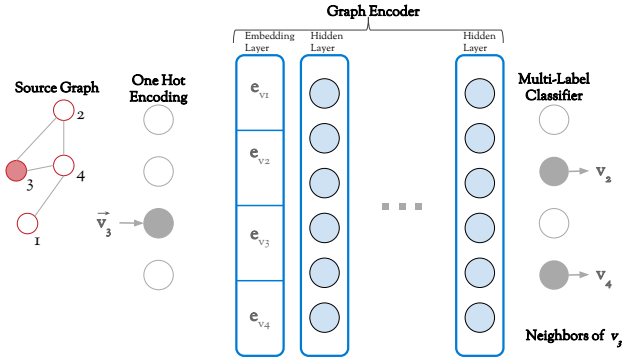


Figure 2: A Node-To-Edges Encoder. Here the input graph contains 4 vertices, and the encoder has to predict the neighbors of vertex v_3. First, v_3 is represented by a one-hot encoding $\vec{v}_3$. Second, $\vec{v}_3$ is multiplied by a linear embedding layer. Third, this embedding $\mathbf{e}_{v_3}$ is passed to a DNN which produces scores for each vertex in V. Finally, these scores are normalized using the *sigmoid* function to produce the final predictions, in this case, $\{v_2, v_4\}$.

importance to the nature of the graph. For instance, swapping one element for another in a graph representing a molecule could drastically change its chemical structure.

We will see how these pairwise alignments can produce divergence scores suitable for Graph Kernels in Section 4.

3.1 Graph Encoding

To learn the structure of a graph, we train an encoder capable of reconstructing such structure given partial or distorted information. In this paper, we choose a *Node-To-Edges* encoder (Figure 2) for its simplicity, but we note that additional choices are certainly possible (see Section 7 for more discussion).

Node-To-Edges Encoder. - In this setup, an encoder is given a single vertex and it is expected to predict its neighbors. This can be modeled as a multilabel classification task since the predictions are not mutually exclusive. Specifically, we are maximizing the following objective function $J(\theta)$,

$$J(\theta) = \sum_i \sum_{\substack{j \\ e_{ij} \in E}} \log \Pr(v_j \mid v_i, \theta). \tag{4}$$

Each vertex v_i in the graph is represented by one-hot encoding vector $\vec{v}_i$. Then to embed the vertex we multiply its encoding vector with a linear layer $\mathbf{E} \in \mathbb{R}^{|V| \times d}$ resulting in an embedded vertex $\mathbf{e}_{v_i} \in \mathbb{R}^d$, where $|V|$ is the number of vertices in the graph, and d is the size of the embedding space.

For graphs with a large number of nodes, we can replace this multiplication with a table lookup, extracting one row from the embedding matrix. This embedding vector represents the feature set given to the encoder tasked with predicting all adjacent vertices. Our encoder H, is implemented as a fully connected deep neural network (DNN) with an output layer of size $|V|$ and trained as a multilabel classifier.

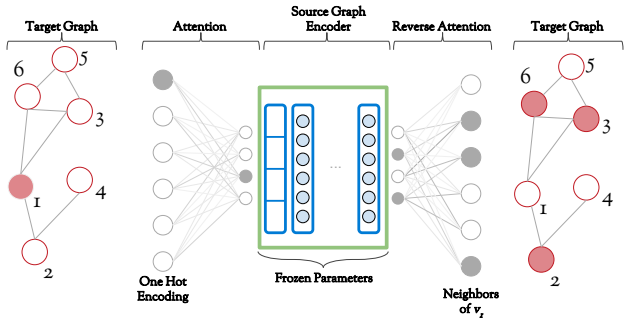


Figure 3: Attention layers mapping the target graph nodes onto the source graph. The augmented encoder has to predict the neighbors of node 1 in the target graph. First, node 1 is passed to the attention layer which assigns it mainly to node 3 of the source graph. Second, the source graph encoder learned earlier (in Figure 2) that the neighbors of node 3 are {2, 4}. Finally, the reverse attention network maps nodes {2, 4} of the source graph to nodes {2, 3, 6} of the target graph which are the neighbors of node 1.

3.2 Cross-Graph Attention

So far, we have developed a utility to encode individual graphs. However, we seek to develop a method which can compare pairs of graphs, which may differ in size (differing node sets) and structure (differing edge sets). For this to happen we need a method of learning an alignment between the graphs. Ideally this method will operate in the absence of a direct mapping between nodes.

In other areas, attention models have been proposed to align structured data. For example, attention models have been proposed to align pairs of images and text [55], pairs of sentences for translation [49], and pairs of speech and transcription [16]. Inspired by these efforts, we formalize the problem of aligning two graphs as that of attention. We propose an attention mechanism, *isomorphism attention*, that aligns the nodes of a target graph against those of a source graph.

3.2.1 Isomorphism Attention. Given two graphs S (*source graph*) and T (*target graph*), we propose a model that allows bi-directional mapping across the pair's nodes. This requires two separate attention networks. The first network allows nodes in the target graph to *attend* to the nodes in the source graph. The second network, allows neighborhood representations in the source graph to *attend* to neighborhoods in the target graph.

We denote the first attention network as ($\mathcal{M}_{T\to S}$), which assigns every node in the target graph ($u_i \in T$) a probability distribution over the nodes of the source graph ($v_j \in S$). This attention network will allow us to pass the nodes of the target graph as an input to the source graph encoder. We implement this attention network using a multiclass classifier,

$$\Pr(v_j \mid u_i) = \frac{e^{\mathcal{M}_{T\to S}(v_j, u_i)}}{\sum_{v_k \in V_S} e^{\mathcal{M}_{T\to S}(v_k, u_i)}}. \tag{5}$$

The second network is a *reverse attention* network ($\mathcal{M}_{S\to T}$) which aims to learn how to map a neighborhood's representation in the source graph to a neighborhood in the target graph. By adding both attention networks to the source graph encoder, we will be able to construct a target graph encoder that is able to predict the neighbors of each node – but utilizing the structure of the source graph. We implement the reverse attention as a multilabel classifier,

$$\Pr(u_j \mid \mathcal{N}(v_i)) = \frac{1}{1 + e^{-\mathcal{M}_{S\to T}(u_j, \mathcal{N}(v_i))}}. \tag{6}$$

Figure 3 shows the attention network ($\mathcal{M}_{T\to S}$) receiving a one-hot encoding vector representing a node (u_i) in the target graph and mapping it onto the most structurally similar node (v_j) from the source graph. The source graph encoder, then, predicts the neighbors of v_j, $\mathcal{N}(v_j)$. The *reverse attention* network ($\mathcal{M}_{S\to T}$), takes $\mathcal{N}(v_j)$ and maps them to the neighbors of u_i, $\mathcal{N}(u_i)$.

Both attention networks may be implemented as linear transformations $\mathbf{W}_A \in \mathbb{R}^{|V_Q|\times|V_P|}$. In the case that either $|V_P|$ or $|V_Q|$ are prohibitively large, the attention network parameters can be decreased by substituting a DNN with hidden layers of fixed size. This will reduce the attention network size from $\Theta(|V_P| \times |V_Q|)$ to $\Theta(|V_P| + |V_Q|)$.

3.3 Attributes Consistency

Labeled graphs are not defined only by their structures, but also by the attributes of their nodes and edges. The attention network assigns each node in the target graph a probability distribution over the nodes of the source graph. There might be several, equally good, nodes in the source graph with similar structural features. However, these nodes may differ in their attributes. To learn an alignment that preserves nodes and edges attributes, we add regularizing losses to the attention and reverse-attention networks.

More specifically, we refer to the nodes as v and u for the source and target graphs, respectively. We refer to the set of attributes as $\mathcal{Y}$ and the distribution of attributes over the graph nodes as ($Q_n = \Pr(y_i \mid u)$). Given that the attention network $\mathcal{M}_{T\to S}$ learns the distribution $\Pr(u_k \mid v_j)$, we can calculate a probability distribution over the attributes as inferred by the attention process as the following:

$$Q_n(y_i|u_j) = \sum_k \mathcal{M}_{T\to S}(y_i|v_k) \Pr(v_k \mid u_j). \tag{7}$$

We define, the attention regularizing loss over the nodes attributes to be the average cross entropy loss between the observed distribution of attributes and the inferred one (See Eq. 8).

$$L = \frac{1}{|V_T|} \sum_j^{|V_T|} \sum_i \Pr(y_i \mid u_j) \log(Q_n(y_i|u_j)), \tag{8}$$

where $|V_T|$ is the number of nodes in the target graph.

For preserving edge attributes over nodes, we define $Q_e(y_i \mid u) = \Pr(y_i \mid u)$ to be the normalized attributes count over all edges connected to the node u. For instance, if a node u has 5 edges with 2 of them colored red and the other three colored yellow, $Q_e(red \mid u) = 0.4$ By replacing Q_n with Q_e in Equations 7 and 8, we create a regularization loss for edge attributes.

We also introduce these regularization losses for *reverse attention* networks. Reverse attention networks maps a neighborhood in the source graph to a neighborhood in the target graph. The distribution of attributes over a node's neighborhood will be the frequency of

each attribute occurrence in the neighborhood normalized by the number of attributes appearing in the neighborhood. For edges, the node's neighborhood edges are the edges appearing at 2-hops distance from the node. Similarly, we can define the probability of the edges attributes by normalizing their frequencies over the total number of attributes of edges connected to the neighborhood.

4 DEEP DIVERGENCE GRAPH KERNELS

So far, we have proposed a method for learning representations of graphs, and an attention mechanism for aligning graphs based on a set of encoded graph representations. Here we discuss our proposed method for using the alignment to construct a graph kernel based on divergence scores. First, in Section 4.1, we show how we can utilize the divergence scores to construct a full graph representation. Divergence is driven by the target graph structure and attribute prediction error as calculated using a source graph encoder. Next we introduce DDGK, our method for learning graph representations for Deep Divergence Graph Kernels in Section 4.3. Then in Section 4.4 we discuss how we train these representations. Finally we discuss the scalability of this approach in Section 4.5.

4.1 Graph Divergence

In Section 3 we presented a method to align two graphs by using a source graph encoder, augmented with attention layers, to encode a target graph. Here, we propose to use the ability of the augmented encoder at predicting the structure of the target graph as a measure of those graphs similarity. To explain, let us assume the trivial case where both the source and target graphs are identical. First, we train the source graph encoder. Second, we augment it with attention networks and train it to predict the structure of the target graph. The attention networks will (ideally) learn the identity function. Therefore, the source graph encoder is able to encode the target graph as accurately as encoding itself. We would reasonably conclude that these graphs are similar.

We aim to learn a metric that measures the divergence score between a pair of graphs $\{S, T\}$. If two graphs are similar, we expect their divergence to be correspondingly low. We refer to the encoder trained on a graph S as H_S and the divergence score given to the target graph T to be

$$\mathcal{D}'(T \parallel S) = \sum_{v_i \in V_T} \sum_{\substack{j \\ e_{ji} \in E_T}} -\log \Pr(v_j \mid v_i, H_S) \tag{9}$$

Given that H_S is not a perfect predictor of the graph S structure, we can safely assume that $\mathcal{D}'(S \parallel S) \neq 0$. To rectify this problem we define

$$\mathcal{D}(S \parallel T) = \mathcal{D}'(S \parallel T) - \mathcal{D}'(S \parallel S), \tag{10}$$

which sets $\mathcal{D}(S \parallel S)$ to zero.

We note that this definition is not symmetric (as $\mathcal{D}(T \parallel S)$ might not necessarily equal to $\mathcal{D}(S \parallel T)$). If symmetry is required, we can define $\mathcal{D}(S, T) = \mathcal{D}(S \parallel T) + \mathcal{D}(T \parallel S)$.

4.2 Graph Embedding

Given a set of source graphs, we can establish a vector space where each dimension corresponds to one graph in the source set. Target graphs are represented as points in this vector space where the value of the i_{th} dimension for a given target graph T_j is $\mathcal{D}(T_j \parallel S_i)$.

More formally, for a set of N source graphs we can define our target graph representation to be:

$$\Psi(G_T) = [\mathcal{D}(T \parallel S_0), \mathcal{D}(T \parallel S_1), \ldots, \mathcal{D}(T \parallel S_N)] \tag{11}$$

To create a kernel out of our graph embeddings, we use the Euclidean distance measure as outlined in Eq 2. This distance measure will guarantee a positive definite kernel [25, 54].

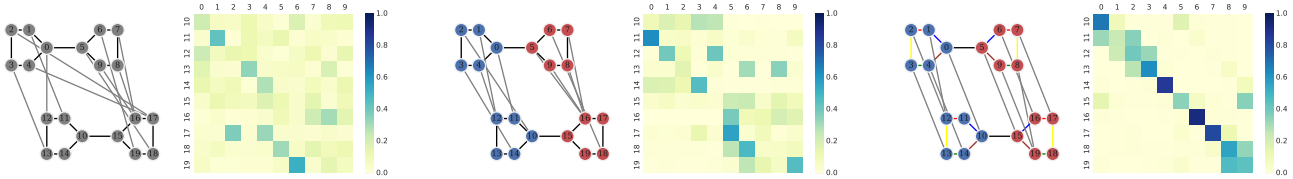
input : Set of N source graphs $\mathcal{S}$
Set of M target graphs $\mathcal{T}$
Learning rate α
Encoding epochs τ
Scoring epochs ρ
output : All graph representations $\Psi \in \mathbb{R}^{M \times N}$

1 *// learn graph encodings*
2 **foreach** $g_i \in \mathcal{S}$ **do**
3 $V, E \leftarrow g_i$
4 **for** *step* $\leftarrow 0$ **to** τ **do**
5 $J(\theta) = -\sum_s \sum_{\substack{t \\ e_{st} \in E}} \log \Pr(v_t \mid v_s, \theta)$
6 $\theta = \theta - \alpha * \frac{\partial J}{\partial \theta}$
7 **end**
8 encodings$[i] \leftarrow \theta$
9 **end**
10 **foreach** $g_i \in \mathcal{T}$ **do**
11 $V, E \leftarrow g_i$
12 **foreach** $\theta_j \in$ encodings **do**
13 *// learn cross-graph attention $\mathcal{M}_{T \to S}$ and $\mathcal{M}_{T \to S}$*
14 **for** *step* $\leftarrow 0$ **to** ρ **do**
15 $J(\mathcal{M}_{T \to S}, \mathcal{M}_{S \to T}) =$
$-\sum_s \sum_{\substack{t \\ e_{st} \in E}} \log \Pr(v_t | v_s, \theta_j, \mathcal{M}_{T \to S}, \mathcal{M}_{S \to T})$
16 $\mathcal{M}_{T \to S} = \mathcal{M}_{T \to S} - \alpha * \frac{\partial J}{\partial \mathcal{M}_{T \to S}}$
17 $\mathcal{M}_{S \to T} = \mathcal{M}_{S \to T} - \alpha * \frac{\partial J}{\partial \mathcal{M}_{S \to T}}$
18 **end**
19 *// calculate graph divergences*
20 $\Psi[i, j] \leftarrow J(\mathcal{M}_{T \to S}, \mathcal{M}_{S \to T})$
21 **end**
22 **end**
23 **return** Ψ

Algorithm 1: DDGK: An unsupervised algorithm for learning graph representations.

4.3 Algorithm : DDGK

We present pseudo-code for DDGK in Algorithm 1. The algorithm has two parts. First, a *Node-To-Edges* encoder is trained for all source graphs (Algorithm 1 line 5 and line 6). Second, cross-graph attentions are learned for all target-source graph pairs (Algorithm 1 line 15, line 16 and line 17). We implement DDGK using a deep neural network for its *Node-To-Edges* encoder and linear transformations for its isomorphism attention.



(a) No attributes. (b) Labeled nodes. (c) Labeled nodes and edges.

Figure 4: The effect of attributes preserving losses on the attention networks. Our method is given a pair of identical graphs, the upper graph represents the target and the other represents the source graph. Each graph consists of two rings of size 5 connected with one edge ((0, 5) and (10, 15) respectively). We visualize the strongest attention weights as cross-graph edges. On the right of each figure we visualize the rest of the attention weights as a heatmap. When the graph attends to itself without attribute preserving losses, there are several solutions that are equally good because of several symmetries available. Once we add the nodes attributes, we can see an immediate effect where the nodes from the same label class attend only to each other. This behavior further intensifies after also adding the edge attributes.

4.4 Training

We implement our models using TensorFlow [44], calculate our gradients using backpropagation, and update our parameters using Adam [29]. We train each source graph on its adjacency matrix for a constant number of iterations.

4.4.1 Target Graph Encoding. Here, the augmented encoder has to predict the neighboring vertices for each vertex in the target graph with the help of the attention and reverse-attention layers. To learn the augmented target graph encoder (which consists of the source graph encoder with the additional attention layers), we use the following procedure:

(1) First, freeze the parameters of the source graph encoder.
(2) Second, add two additional networks, one for attention and another for reverse attention mapping between the target graph nodes to the source graph nodes and vice versa.
(3) Third, add the regularizing losses to preserve the nodes or edges attributes if they are available.
(4) Fourth, train the augmented encoder on the input, which is the adjacency matrix of the target graph, and a node attribute and/or edge attribute matrix (if available).

Finally, once the training of the attention layers is done, we use the augmented encoder to compute the divergence between the graph pair as discussed in 4.1.

4.5 Scalability

We start by defining the following quantities: N the number of source graphs in the dataset, M the number of target graphs in the dataset, V the average number of nodes, τ the number of epochs to encode source graphs, ρ the number of epochs to encode target graphs, l the number of encoder hidden layers, m the number of attention hidden layers, and d the embedding and hidden layer size

Our method relies on pairwise similarity, therefore, we will have $M \times N$ computations that each involves scoring a target graph against one source graph. Training a source graph encoder requires τ steps that each involves $2\times V\times d+l\times d^2$ computations. In addition to running the source graph encoder, the target graph alignment learns the attention networks which represents $\rho\times(2\times d\times V+m\times d^2)$. If we define $T = max(\rho, \tau)$, $k = max(l, m)$, and $M = N$ then the total computation cost is $\Theta(N^2 \times T \times (V \times d + k \times d^2)$. Because V is likely much larger than d^2, we interpret the computational complexity as $O(TN^2V)$.

In Section 5.4, we explore the effect of sampling to hasten DDGK's runtime on large datasets. We show that not all $M \times N$ comparisons are necessary to achieve high performance: empirically, it seems that less than 20% of source graphs are required, significantly speeding our approach.

5 EXPERIMENTS

In this section, we demonstrate our method through a series of qualitative and quantitative experiments. First, we show how our attention based alignment works under different conditions. Then, we show how our representations are capable of capturing the structure of the space of graphs by applying hierarchical clustering on the kernel space. Finally, we show that the learned graph embeddings represent a sufficient feature set to predict the graph label on several challenging classification problems in real-world biological datasets.

5.1 Cross-Graph Attention

In this qualitative experiment, we seek to understand how two graphs are related to each other. Comparing different patterns between different graphs is an important application in domains such as biology.

Figure 4a shows two identical unlabeled barbell graphs. Each graph consists of two rings of size 5 connected with the edge (0, 5) and (10, 15). The upper graph represents the target graph while the lower one represents the source graph. The edges connecting the source and target graphs represent the strongest attention weights for each node in the target graph. The heatmap shows the full attention matrix for more thorough analysis. Aligning these identical graphs is an easy task for the naked eye. However, our method can find many possible symmetries to exploit while still achieving perfect predictions. For example, nodes in the left ring can attend to the right ring of the source graph and vice versa.

Figure 4b shows the previous setup with labeled graph nodes. This introduces a regularizing loss to preserve the node attributes. The attention heat map shows significant weights for the upper

left and lower rights quadrants. The right ring does not attend to nodes in the left ring anymore, and vice versa. Still we can see the method exploiting symmetries within the same ring.

Finally, by also adding edge labels, the alignment problem is constrained enough that the attention heatmap is concentrated along the diagonal (See Figure 4c). We can observe that the attention edges correspond in a one-to-one relationship between the target and source graphs. This synthetic experiment shows the effect of attribute preserving losses on learning the alignment between graphs.

5.2 Hierarchical Clustering

To understand the global structure of the graph embedding space, we explore it qualitatively using hierarchical clustering. First, we create a dataset which is a composition of 6 different families of graphs. Three graph families are mutated graphs and three families are subset of a larger set of realistic graphs. From each family we sample 5 graphs, creating a universe of 30 graphs. Then, we embed the graphs using our method constructing a graph embedding space. Finally, we cluster the embeddings according to their pairwise euclidean distances.

5.2.1 Mutated Graphs. For these datasets we start with a known graph and generate a sequence of mutations to produce a family of graphs. In particular, we consider the following graphs.

- C. Elegans [53]: represents the neural network of the C. Elegans worm.
- Karate Club [60]: social network of friendships between 34 members of a karate club.
- Word Network [35]: adjacency network of common adjectives and nouns in the novel David Copperfield by Charles Dickens.

In order to generate a family $G_1 \cdots G_k$ for each original graph G_0, we employ the following mutation procedure. At each of the k time steps, there is a $p = 0.5$ chance of performing an edge deletion or addition. For additions, we select the two nodes to connect from any unlinked nodes according to the preferential attachment model characterized by G_0 [17]. For deletions, we select an edge at random and remove it. We run this procedure for 4 times with $k = 50$ time steps, creating a family of 5 related graphs. The initial seed for any of these mutations is denoted by the suffix "-0".

5.2.2 Realistic Graphs. We randomly pick 5 graphs from three of the real-world families of graphs we consider (**D&D**, **PTC**, and **NCI1**). See Section 5.3.2 for more information about these graphs.

Figure 5 shows the result of clustering the pairwise distances between our graph embeddings. We are able to retrieve perfect clusters of {`c-elegans`, `words`} where there are clusters of size of 5 that consist only of graphs of the same type. For {**NCI1**, **D&D**}, we can cluster 4 graphs out of 5 before adding a graph which is out of the family.

5.3 Graph Classification

Our learned graph representations respect both attributes and structure. They can be used for graph classification tasks where the graph structure, node attributes, and/or edge attributes convey meaning or function. To demonstrate this, we use DDGK representations of several chemo- and bio- informatics datasets as features for classification tasks. We report our results against both unsupervised and supervised methods.

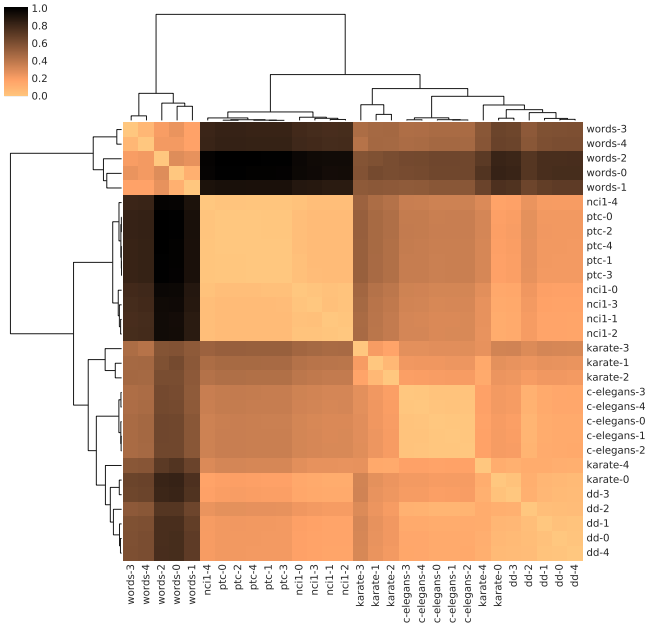


Figure 5: A hierarchical clustering of the graph kernel space for several different graph families. It shows 30 graphs that belong to 6 different families. The values of the matrix are the pairwise Euclidean distances between the graph embeddings.

Hyper-Parameter	Values
Node embedding	2, 4, 8, 16, 32
Encoder layers	1, 2, 3, 4
Learning rate	10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1
Encoding epochs	100, 300, 600
Scoring epochs	100, 300, 600
Node preserving loss coefficient	0, 0.25, 0.5, 1.0, 1.5, 2.0
Edge preserving loss coefficient	0, 0.25, 0.5, 1.0, 1.5, 2.0

Table 1: Values used during our grid search for DDGK graph representations learning hyper-parameters.

5.3.1 Hyper-parameters Search. To choose DDGK hyperparameters (See Table 1), we perform grid searches for each dataset. We create splits of each dataset to avoid over-fitting, they are: {`train`, `dev`, `test`}. We use the scikit-learn SVM [39] as our classifier, and we vary the kernel choices between {`linear`, `rbf`, `poly`, `sigmoid`} and the regularization coefficient C between 10 and 10^9. We choose the hyper-parameters of both DDGK and the classifier that maximize the accuracy on the `dev` dataset.

	# Graphs	Average Nodes	Average Edges	# Labels	# Node Labels	# Edge Labels
D&D	1178	284	716	2	89	–
NCI1	4110	30	32	2	37	–
PTC	344	14	15	2	18	4
MUTAG	188	18	20	2	7	4

Table 2: Statistics of the chemo- and bio-informatics datasets.

5.3.2 Datasets. Four benchmark graph classification datasets from chemo- and bio-informatics domains are used. The datasets include **D&D**, **NCI1**, **PTC** and **MUTAG**. All datasets include node labels. The **PTC** and **MUTAG** datasets also include edge labels. Table 2 shows network statistics for each dataset. The datasets:

- **D&D** [20]: contains 1178 proteins labeled as enzymes or non-enzymes.
- **NCI1** [51]: contains 4110 chemical compounds labeled as active or inactive against non-small cell lung cancer.
- **PTC** [46]: contains 344 chemical compounds labeled according to their carcinogenicity in male rats.
- **MUTAG** [19]: contains 188 mutagenic aromatic and heteroaromatic compounds labeled according to their mutagenic effect on a specific gram negative bacterium.

5.3.3 Results. The results of these experiments are presented in Table 3. We see that DDGK is quite competitive, with higher average performance on both the **D&D** and **MUTAG** datasets than any of the baselines. This is especially surprising given that the supervised methods have additional information available to them. We note that DDGK achieves its strong results without engineered features, or access to information from Weisfeiler-Lehman kernels. For **PTC**, we also see that DDGK attains competitive performance against all other methods, only being outperformed by 2 of the 9 baselines. Finally, on **NCI1**, we see that DDGK performs better than the method using the most similar kind of information (`node2vec`), but find that baselines using the WL kernel perform best on this dataset (indeed, the WL kernel itself takes the top two spots). We find this dependence quite interesting, and will seek to characterize it better in future work.

5.4 Dimension Sampling

So far, we have been setting the source graphs set to be equal to the target graphs set. This pairwise computation is quite expensive for large datasets. To reduce the computational complexity of our method, we study the effect of sub-sampling the dimensions of our graph embedding space on the quality of graph classification.

To do that, we construct a source graph set that is a subset of the original graph set. We learn divergence scores for all target graphs against this subset. We use the reduced embeddings as features to predict graph categories. Figure 6 shows that we are able to achieve stable and competitive results with less than 20% of the graphs being used as source graphs.

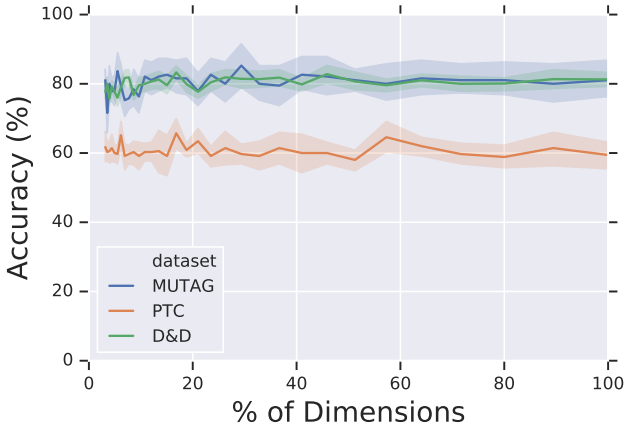


Figure 6: Effect of sub-sampling source graphs on graph classification tasks. Here we vary the number of source graphs available to each method, and observe that very few dimensions are needed to achieve our final classification performance (less than $< 20\%$ of the dimensions for the datasets considered).

6 RELATED WORK

The main differences between our proposed method and previous work can be summarized as follows:

(1) We are an unsupervised method, taking only a graph as input.
(2) We use no domain-specific information about what primitives are important in a graph, using only the edges.
(3) We use no algorithmic insights from the literature in graph isomorphism (e.g. the Weisfeiler-Lehman kernel).
(4) We assume nothing about the mapping of node ids between graphs, instead learning the alignment.

While many approaches exist that contain at least one of these differentiators, we are, to the best of our knowledge, the only proposed method that meets all four of these conditions. In this section we will briefly cover related work in graph similarity and other applications of neural networks to graph representation.

6.1 Unsupervised Graph Similarity

We divide our brief survey of the literature into three kinds of unsupervised methods for graph similarity. The first seeks to explicitly define a kernel over graph features, or use the intuition from such a kernel as part of the representation learning process. The second focuses on the representation of individual elements of the graph, learning primitives that maximize some kind of reconstruction of the graph. The third group of work constructs a similarity function between graphs by an explicit vector of statistical features constructed by the graph.

Traditional Graph Kernels: There has been considerable work done on unsupervised methods for graph kernel learning. Initial efforts in the area focused on theoretical views of the problem, defining graph similarity via the Graph Edit Distance [22] or the size of the Maximum Common Subgraph [11] between graphs.

Method		Unsupervised	No Weisfeiler-Lehman	No Feature Engineering	D&D	NCI1	PTC	MUTAG
PSCN	[36]				76.27 ± 2.64	76.34 ± 1.68	62.29 ± 5.68	88.95 ± 4.37
DGCNN	[61]				–	74.44 ± 0.40	58.59 ± 2.40	85.83 ± 1.60
SP Kernel	[8]	✓			79.00 ± 0.60	74.50 ± 0.30	58.90 ± 2.20	83.00 ± 1.40
WL Kernel	[32]	✓			79.00 ± 0.40	85.80 ± 0.20	61.30 ± 1.40	86.00 ± 1.70
WL-OA	[32]	✓			79.20 ± 0.40	**86.10** ± 0.20	63.60 ± 1.50	86.00 ± 1.70
DGK	[56]	✓			–	80.30 ± 0.40	60.10 ± 2.50	87.40 ± 2.70
graph2vec	[34]	✓			–	73.22 ± 1.90	60.17 ± 6.90	83.15 ± 9.20
S2S-N2N-PP	[43]	✓			–	83.72 ± 0.40	**64.54** ± 1.10	89.86 ± 1.10
node2vec	[24]	✓	✓		–	61.91 ± 0.30	55.60 ± 1.40	82.01 ± 1.00
DDGK	(this paper)	✓	✓	✓	**83.14** ± 2.72	68.10 ± 2.30	63.14 ± 6.57	**91.58** ± 6.74

Table 3: Average accuracy in ten-fold cross validation on our graph classification task. Methods are grouped by their level of supervision during the similarity metric learning, whether they use algorithm insights the Weisfeiler-Lehman algorithm, and whether they use feature engineering (e.g. graph motifs, random walks, etc.). Baseline results taken from [32, 36, 43] (missing results are missing from these works). We note that DDGK performs surprisingly competitively for an unsupervised method with no hard-coded insights.

Unfortunately these problems are both NP-Complete in the general case, require a known correspondence between the nodes of the two graphs of interest.

Many approaches are built around the graph similarity measure computed by the Weisfeiler-Lehman (WL) subtree graph kernel [32, 42]. At its core, the WL algorithm collapses the labels of a node's neighbors into a ordered sequence, and then hashes that sequence into a new label for the node. This process repeats iteratively to average information over the neighborhood together. Other functions that use different types of predefined features for graph similarity, such as shortest-paths kernels [8], and random walk kernels [28] have also been proposed, but their naive implementations suffer from high asymptotic complexity ($O(n^4)$ and $O(n^6)$, respectively). Faster implementations of these kernels have been proposed [10, 27]. Some unsupervised methods also focus on extending the algorithm intuition of these classic approaches to the problem. For instance [43] learns a representation for each position in a WL ordering jointly while learning a graph representation.

Unlike all of these approaches, our method deliberately avoids algorithmic insights. Our proposed isomorphism attention mechanism allows capturing higher-order structure between graphs (beyond immediate neighborhoods).

Node embedding methods: Since DeepWalk [40] proposed embedding the nodes via a sequence of random walks, the problem of node representation learning has received considerable attention [2, 7, 15, 21, 24, 41, 48]. In general, all of these methods utilize insights about similarity functions which are important to the graph. While these methods seek the best way to represent nodes, the representations are learned independently between graphs, which makes them generally unsuitable for graph similarity computations. For more information on this area, we recommend recent surveys in the area [14, 18]. Unlike these methods, our goal is to learn representations of graphs, not of nodes.

Graph statistics: Finally, another family of unsupervised graph similarity measures define a hand-engineered feature vector to compute graph similarity. The NetSmilie method [6] operates by constructing a fixed size feature value of graph statistics and uses this as a similarity embedding over graphs. Similarly, DetlaCon [31] defines the similarity over two graphs with known node-to-node mapping via the similarity in their propagation of belief, and [38] proposes a number of similarity measures over directed web graphs.

Unlike these methods, DDGK does not explicitly engineer its features for the problem. Instead, the similarity is learned function directly from the edges present in the adjacency matrix, with no assumptions about which features are important for the application task.

6.2 Supervised Graph Similarity

The first class of supervised methods uses some supervision to inform a similarity function constructed over different hand-engineered graph features.

A number of supervised approaches also utilize intuitions from the Weisfeiler-Lehman graph kernel. `Patchy-SAN` [37] proposes an approach for convolutional operations on graph structured data. The core of their method uses the ordering from the WL kernel to order the nodes of a rooted subgraph into a sequence, and then apply standard 1-dimensional convolutional filters. This approach is further generalized by [61], who use the WL ordering to sort a graph sample in a pooling layer. Another branch of work has focused on extending the Graph Convolutional Networks (GCNs) proposed by [30] to perform supervised classification of graphs. Proposed extensions include a pooling architecture that learns a soft clustering of the graph [59], or a two-tower model which frames graph similarity as link prediction between GCN representations [4]. Interestingly, it has been shown that many of these methods

are not necessarily more expressive than the original Weisfeiler-Lehman subtree kernel itself [33].

Unlike all of these approaches, our method learns representations of graphs without supervision — we use no labels about the class label of a graph, and no external information about which pairs of graphs are related. Our proposed isomorphism attention mechanism allows capturing higher-order structure between graphs (beyond immediate neighborhoods).

7 EXTENSIONS & FUTURE WORK

Here we briefly discuss a number of areas of future investigation for our method.

7.1 Graph Encoders

Given the choice of input and reconstructed output, several additional graph encoders are possible, in addition to the Nodes-To-Edges encoder which we used in this work. To mention a few options:

Edge-To-Nodes Encoder. - This encoder is trained to predict the source and destination vertices given a specific edge. Similar to the *Node-To-Edges* encoder, this could be expressed as a multilabel classification task with the following objective function,

$$J(\theta) = \sum_{e_{ij} \in E} \log \Pr(v_i \mid e_{ij}, \theta) + \log \Pr(v_j \mid e_{ij}, \theta) \tag{12}$$

Note that the number of edges in a graph could grow quadratically, therefore, iterating over the edges is more expensive than the nodes.

Neighborhood Encoder. - In this case, the encoder is trained to predict a set of vertices or edges that are beyond the immediate neighbors. Random walks could serve as a mechanism to calculate a neighborhood around a specific node or edge. Given a partial random walk, the encoder has to predict the vertices that could have been visited within a specific number of hops.

$$J(\theta) = \sum_{\substack{(v_1, v_2, \cdots, v_i) \\ \sim RandomWalk(G,E,V)}} \log \Pr\Big(v_j \mid (v_1, v_2, \cdots, v_i, \theta)\Big) \tag{13}$$

7.2 Attention Mechanism

We proposed a simple attention mechanism which uses node-to-node alignment. As we discussed in Section 4.5, we could replace the linear layer with a deep neural network to reduce the size of the model if scability is an issue. While node-to-node alignment enhances the interpretability of our models, subgraph alignment could lead to better and easier understanding of how two graphs are similar. Hierarchical attention models [57] could lead to higher levels of abstractions which could learn community structure and which communities are similar across a pair of graphs. Hierarchy has already been used within the context of learning better node embeddings, for example [59] showed that a better understanding of the graph substructure can lead to better graph classification. Therefore, we believe extending the work beyond node-to-node alignment will significantly improve our results.

7.3 Regularization

We proposed attribute based losses to regularize our isomorphism attention mechanism. The graph encoder capacity was adjusted according to the source graph size. However, the source graph encoder could still suffer from overfitting which would reduce its utility in recognizing similar target graphs. Therefore, further research is necessary to understand the relation between the encoder training characteristics and the quality of the generated divergence scores

7.4 Feature Engineering

In this work we have focused on developing an approach for representing graphs that operated without any feature engineering or algorithmic insights. While this willful ignorance has allowed us to design a new paradigm for graph similarity, we suspect that there are many fruitful combinations of this idea with other approaches for graph classification. For example, the graph embeddings we learn could be used as additional features for approaches based on learning supervised classifiers over graphs.

8 CONCLUSION

In this work, we have shown that neural networks can learn powerful representations of graphs without explicit feature engineering. Our proposed method, Deep Divergence Graph Kernels, learns an encoder for each graph to capture its structure, and uses a novel *isomorphism preserving attention mechanism* to align node representations across graphs without the use of supervision. We show that representing graphs by their divergence from different source graphs provides a powerful embedding space over families of graphs. Our proposed model is both flexible and amenable to extensions. We illustrate this by proposing extensions to handle many commonly occurring varieties of graphs, including graphs with attributed nodes, and graphs with attributed edges.

Our experimental analysis shows that despite being trained with only the graph's edges (and no feature engineering) the learned representations encode a variety of local and global information. When the representations produced by DDGK are used as features for graph classification methods, we find them to be competitive with challenging baselines which use at least one of graph labels, engineered features, or the Weisfeiler-Lehman framework. In addition to being powerful, DDGK models are incredibly informative. The learned isomorphism attention weights allow a level of insight into the alignment between a pair of graphs, which is not possible with other deep learning methods developed for graph similarity.

Unsupervised representation learning for graphs is an important problem, and we believe that the method of Deep Divergence Graph Kernels we have introduced here is an exciting step forward in this area. As future work, we will investigate 1) enhanced method for choosing informative source from the space of all graphs, 2) improving the architecture of our encoders and attention models, 3.) making it easier to reproduce research results in the area of graph similarity, and 4) making graph similarity models even easier to understand.

REFERENCES

[1] Sami Abu-El-Haija, Bryan Perozzi, and Rami Al-Rfou. 2017. Learning Edge Representations via Low-Rank Asymmetric Projections. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. ACM, New York, NY, USA, 1787–1796. https://doi.org/10.1145/3132847.3132959

[2] Sami Abu-El-Haija, Bryan Perozzi, Rami Al-Rfou, and Alexander A Alemi. 2018. Watch Your Step: Learning Node Embeddings via Graph Attention. In *Advances in Neural Information Processing Systems*. 9198–9208.

[3] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2289–2294. https://doi.org/10.18653/v1/D16-1250

[4] Yunsheng Bai, Hao Ding, Song Bian, Yizhou Sun, and Wei Wang. 2018. Graph Edit Distance Computation via Graph Neural Networks. *arXiv preprint arXiv:1808.05689* (2018).

[5] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261* (2018).

[6] Michele Berlingerio, Danai Koutra, Tina Eliassi-Rad, and Christos Faloutsos. 2012. Netsimile: A scalable approach to size-independent network similarity. *arXiv preprint arXiv:1209.2684* (2012).

[7] Aleksandar Bojchevski and Stephan Günnemann. 2017. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815* (2017).

[8] Karsten M Borgwardt and Hans-Peter Kriegel. 2005. Shortest-path kernels on graphs. In *Data Mining, Fifth IEEE International Conference on*. IEEE, 8–pp.

[9] Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. 2005. Protein function prediction via graph kernels. *Bioinformatics* 21, suppl_1 (2005), i47–i56.

[10] Karsten M Borgwardt, Nicol N Schraudolph, and SVN Vishwanathan. 2007. Fast computation of graph kernels. In *Advances in neural information processing systems*. 1449–1456.

[11] Horst Bunke, Pasquale Foggia, Corrado Guidobaldi, Carlo Sansone, and Mario Vento. 2002. A comparison of algorithms for maximum common subgraph on randomly connected graphs. In *SSPR*. Springer, 123–132.

[12] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep Neural Networks for Learning Graph Representations. In *Proceedings of the Association for the Advancement of Artificial Intelligence*.

[13] Sandro Cavallari, Vincent W Zheng, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. 2017. Learning community embedding with community detection and node embedding on graphs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 377–386.

[14] Haochen Chen, Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2018. A Tutorial on Network Embeddings. *arXiv preprint arXiv:1808.02590* (2018).

[15] Haochen Chen, Bryan Perozzi, Yifan Hu, and Steven Skiena. 2018. Harp: Hierarchical representation learning for networks. (2018).

[16] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-Based Models for Speech Recognition. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 577–585. http://papers.nips.cc/paper/5847-attention-based-models-for-speech-recognition.pdf

[17] Fan Chung, Fan RK Chung, Fan Chung Graham, Linyuan Lu, Kian Fan Chung, et al. 2006. A Generative Model - the Preferential Attachment Scheme. In *Complex graphs and networks*. Number 107. American Mathematical Soc., Chapter 3.

[18] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2018. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[19] Asim Kumar Debnath, Rosa L. Lopez de Compadre, Gargi Debnath, Alan J. Shusterman, and Corwin Hansch. 1991. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity. *Journal of Medicinal Chemistry* 34, 2 (1991), 786–797. https://doi.org/10.1021/jm00106a046 arXiv:https://doi.org/10.1021/jm00106a046

[20] Paul D Dobson and Andrew J Doig. 2003. Distinguishing enzyme structures from non-enzymes without alignments. *Journal of molecular biology* 330, 4 (2003), 771–783.

[21] Alessandro Epasto and Bryan Perozzi. 2019. Is a Single Embedding Enough? Learning Node Representations that Capture Multiple Social Contexts. *WWW* (2019).

[22] Xinbo Gao, Bing Xiao, Dacheng Tao, and Xuelong Li. 2010. A survey of graph edit distance. *Pattern Analysis and applications* 13, 1 (2010), 113–129.

[23] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 1263–1272.

[24] A. Grover and J. Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[25] Bernard Haasdonk and Claus Bahlmann. 2004. Learning with distance substitution kernels. In *Joint Pattern Recognition Symposium*. Springer, 220–227.

[26] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[27] U Kang, Hanghang Tong, and Jimeng Sun. 2012. Fast random walk graph kernel. In *Proceedings of the 2012 SIAM International Conference on Data Mining*. SIAM, 828–838.

[28] Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. 2003. Marginalized kernels between labeled graphs. In *Proceedings of the 20th international conference on machine learning (ICML-03)*. 321–328.

[29] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[30] T. Kipf and M. Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint arXiv:1609.02907*.

[31] Danai Koutra, Joshua T Vogelstein, and Christos Faloutsos. 2013. Deltacon: A principled massive-graph similarity function. In *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 162–170.

[32] Nils M Kriege, Pierre-Louis Giscard, and Richard Wilson. 2016. On valid optimal assignment kernels and applications to graph classification. In *Advances in Neural Information Processing Systems*. 1623–1631.

[33] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. 2018. Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks. *arXiv preprint arXiv:1810.02244* (2018).

[34] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. 2017. graph2vec: Learning Distributed Representations of Graphs. *CoRR* abs/1707.05005 (2017). arXiv:1707.05005 http://arxiv.org/abs/1707.05005

[35] M. E. J. Newman. 2006. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E* 74 (Sep 2006), 036104. Issue 3. https://doi.org/10.1103/PhysRevE.74.036104

[36] M. Niepert, M. Ahmed, and K. Kutzkov. 2016. Learning Convolutional Neural Networks for Graphs. In *International Conference on Machine Learning (ICML)*.

[37] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. 2016. Learning convolutional neural networks for graphs. In *International conference on machine learning*. 2014–2023.

[38] Panagiotis Papadimitriou, Ali Dasdan, and Hector Garcia-Molina. 2010. Web graph similarity for anomaly detection. *Journal of Internet Services and Applications* 1, 1 (2010), 19–30.

[39] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[40] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[41] Bryan Perozzi, Vivek Kulkarni, Haochen Chen, and Steven Skiena. 2017. Don't Walk, Skip!: Online Learning of Multi-scale Network Embeddings. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*. ACM, 258–265.

[42] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research* 12, Sep (2011), 2539–2561.

[43] Aynaz Taheri, Kevin Gimpel, and Tanya Berger-Wolf. 2018. Learning Graph Representations with Recurrent Neural Network Autoencoders. In *KDD'18 Deep Learning Day*.

[44] TensorflowTeam. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. http://tensorflow.org/ Software available from tensorflow.org.

[45] Antoine J-P Tixier, Giannis Nikolentzos, Polykarpos Meladianos, and Michalis Vazirgiannis. 2018. Graph Classification with 2D Convolutional Neural Networks. (2018).

[46] Hannu Toivonen, Ashwin Srinivasan, Ross D King, Stefan Kramer, and Christoph Helma. 2003. Statistical evaluation of the predictive toxicology challenge 2000–2001. *Bioinformatics* 19, 10 (2003), 1183–1193.

[47] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, Alex Bronstein, and Emmanuel Müller. 2018. NetLSD: Hearing the Shape of a Graph. *KDD* (2018).

[48] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2017. VERSE: Versatile Graph Embeddings from Similarity Measures. (2017).

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[50] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* 1, 2 (2017).

[51] Nikil Wale and George Karypis. 2006. Comparison of descriptor spaces for chemical compound retrieval and classification. In *Proceedings - Sixth International Conference on Data Mining, ICDM 2006*. 678–689. https://doi.org/10.1109/ICDM.2006.39

[52] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community Preserving Network Embedding.. In *AAAI*. 203–209.

[53] Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of âĂŸsmall-worldâĂŹnetworks. *nature* 393, 6684 (1998), 440.

[54] Lingfei Wu, Ian En-Hsu Yen, Fangli Xu, Pradeep Ravikuma, and Michael Witbrock. 2018. D2KE: From Distance to Kernel and Embedding. *arXiv preprint arXiv:1802.04956* (2018).

[55] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. 2048–2057.

[56] Pinar Yanardag and SVN Vishwanathan. 2015. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1365–1374.

[57] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[58] Zi Yin and Yuanyuan Shen. 2018. On the Dimensionality of Word Embedding. In *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). Curran Associates, Inc., 887–898. http://papers.nips.cc/paper/7368-on-the-dimensionality-of-word-embedding.pdf

[59] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L Hamilton, and Jure Leskovec. 2018. Hierarchical Graph Representation Learning with Differentiable Pooling. *arXiv preprint arXiv:1806.08804* (2018).

[60] Wayne W. Zachary. 1977. An Information Flow Model for Conflict and Fission in Small Groups. *Journal of Anthropological Research* 33, 4 (1977), 452–473.

[61] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. 2018. An end-to-end deep learning architecture for graph classification.

[62] Vincent W Zheng, Sandro Cavallari, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. 2016. From node embedding to community embedding. *arXiv preprint arXiv:1610.09950* (2016).

[63] Dingyuan Zhu, Peng Cui, Daixin Wang, and Wenwu Zhu. 2018. Deep Variational Network Embedding in Wasserstein Space. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2827–2836.

Stereotypical Bias Removal for Hate Speech Detection Task using Knowledge-based Generalizations

Pinkesh Badjatiya, Manish Gupta*, Vasudeva Varma
IIIT Hyderabad
Hyderabad, India
pinkesh.badjatiya@research.iiit.ac.in,{manish.gupta,vv}@iiit.ac.in

ABSTRACT

With the ever-increasing cases of hate spread on social media platforms, it is critical to design abuse detection mechanisms to proactively avoid and control such incidents. While there exist methods for hate speech detection, they stereotype words and hence suffer from inherently biased training. Bias removal has been traditionally studied for structured datasets, but we aim at bias mitigation from unstructured text data.

In this paper, we make two important contributions. First, we systematically design methods to quantify the bias for any model and propose algorithms for identifying the set of words which the model stereotypes. Second, we propose novel methods leveraging knowledge-based generalizations for bias-free learning.

Knowledge-based generalization provides an effective way to encode knowledge because the abstraction they provide not only generalizes content but also facilitates retraction of information from the hate speech detection classifier, thereby reducing the imbalance. We experiment with multiple knowledge generalization policies and analyze their effect on general performance and in mitigating bias. Our experiments with two real-world datasets, a Wikipedia Talk Pages dataset (WikiDetox) of size ~96k and a Twitter dataset of size ~24k, show that the use of knowledge-based generalizations results in better performance by forcing the classifier to learn from generalized content. Our methods utilize existing knowledge-bases and can easily be extended to other tasks.

CCS CONCEPTS

• **Networks** → **Online social networks**; • **Social and professional topics** → **Hate speech**; • **Computing methodologies** → **Supervised learning by classification**; **Neural networks**; *Lexical semantics.*

KEYWORDS

hate speech; stereotypical bias; knowledge-based generalization; natural language processing; bias detection; bias removal

ACM Reference Format:
Pinkesh Badjatiya, Manish Gupta*, Vasudeva Varma. 2019. Stereotypical Bias Removal for Hate Speech Detection Task using Knowledge-based Generalizations. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313504

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313504

Examples	Predicted Hate Label (Score)	True Label
those guys are nerds	Hateful (0.83)	Neutral
can you throw that garbage please	Hateful (0.74)	
People will die if they kill Obamacare	Hateful (0.78)	
oh shit, i did that mistake again	Hateful (0.91)	
that arab killed the plants	Hateful (0.87)	
I support gay marriage. I believe they have a right to be as miserable as the rest of us.	Hateful (0.77)	

Table 1: Examples of Incorrect Predictions from the Perspective API

1 INTRODUCTION

With the massive increase in user-generated content online, there has also been an increase in automated systems for a large variety of prediction tasks. While such systems report high accuracy on collected data used for testing them, many times such data suffers from multiple types of unintended biases. A popular unintended bias, stereotypical bias (SB), can be based on typical perspectives like skin tone, gender, race, demography, disability, Arab-Muslim background, etc. or can be complicated combinations of these as well as other confounding factors. Not being able to build unbiased prediction systems can lead to low-quality unfair results for victim communities. Since many of such prediction systems are critical in real-life decision making, this unfairness can propagate into government/organizational policy making and general societal perceptions about members of the victim communities [13].

In this paper, we focus on the problem of detecting and removing such bias for the hate speech detection task. With the massive increase in social interactions online, there has also been an increase in hateful activities. Hateful content refers to "posts" that contain abusive speech targeting individuals (cyber-bullying, a politician, a celebrity, a product) or particular groups (a country, LGBT, a religion, gender, an organization, etc.). Detecting such hateful speech is important for discouraging such wrongful activities. While there have been systems for hate speech detection [1, 9, 16, 21], they do not explicitly handle unintended bias.

*Author is also a Principal Applied Researcher at Microsoft. (gmanish@microsoft.com)

We address a very practical issue in short text classification: that the mere presence of a word may (sometimes incorrectly) determine the predicted label. We thus propose methods to identify if the model suffers from such an issue and thus provide techniques to reduce the undue influence of certain keywords.

Acknowledging the importance of hate speech detection, Google launched Perspective API [1] in 2017. Table 1 shows examples of sentences that get incorrectly labeled by the API (as on 15^{th} Aug 2018). We believe that for many of these examples, a significant reason for the errors is bias in training data. Besides this API, we show examples of stereotypical bias from our datasets in Table 2.

Traditionally bias in datasets has been studied from a perspective of detecting and handling imbalance, selection bias, capture bias, and negative set bias [14]. In this paper, we discuss the removal of stereotypical bias. While there have been efforts on identifying a broad range of unintended biases in the research and practices around social data [18], there is hardly any rigorous work on principled detection and removal of such bias. One widely used technique for bias removal is sampling, but it is challenging to obtain instances with desired labels, especially positive labels, that help in bias removal. These techniques are also prone to sneak in further bias due to sample addition. To address such drawbacks, we design *automated* techniques for bias detection, and *knowledge-based generalization* techniques for bias removal with a focus on hate speech detection.

Enormous amounts of user-generated content is a boon for learning robust prediction models. However, since stereotypical bias is highly prevalent in user-generated data, prediction models implicitly suffer from such bias [5]. To de-bias the training data, we propose a two-stage method: Detection and Replacement. In the first stage (Detection), we propose skewed prediction probability and class distribution imbalance based novel heuristics for bias sensitive word (BSW) detection. Further, in the second stage (Replacement), we present novel bias removal strategies leveraging knowledge-based generalizations. These include replacing BSWs with generalizations like their Part-of-Speech (POS) tags, Named Entity tags, WordNet [17] based linguistic equivalents, and word embedding based generalizations.

We experiment with two real-world datasets – WikiDetox and Twitter of sizes ~96K and ~24K respectively. The goal of the experiments is to explore the trade-off between hate speech detection accuracy and stereotypical bias mitigation for various combinations of detection and replacement strategies. Since the task is hardly studied, we evaluate using an existing metric Pinned AUC, and also propose a novel metric called PB (Pinned Bias). Our experiments show that the proposed methods result into proportional reduction in bias with an increase in the degree of generalization, without any significant loss in hate speech detection task performance.

In this paper, we make the following important contributions.

- We design a principled two-stage framework for stereotypical bias mitigation for hate speech detection task.
- We propose novel bias sensitive word (BSW) detection strategies, as well as multiple BSW replacement strategies and show improved performance on existing metrics.
- We propose a novel metric called PB (Pinned Bias) that is easier to compute and is effective.
- We perform experiments on two large real-world datasets to show the efficacy of the proposed framework.

The remainder of the paper is organized as follows. In Section 2, we discuss related work in the area of bias mitigation for both structured and unstructured data. In Section 3, we discuss a few definitions and formalize the problem definition. We present the two-stage framework in Section 4. We present experimental results and analysis in Section 5. Finally, we conclude with a brief summary in Section 6.

2 RELATED WORK

Traditionally bias in datasets has been studied from a perspective of detecting and handling imbalance, selection bias, capture bias, and negative set bias [14]. In this paper, we discuss the removal of stereotypical bias.

2.1 Handling Bias for Structured Data

These bias mitigation models require structured data with bias sensitive attributes ("protected attributes") explicitly known. While there have been efforts on identifying a broad range of unintended biases in the research and practices around social data [18], only recently some research work has been done about bias detection and mitigation in the fairness, accountability, and transparency in machine learning (FAT-ML) community. While most of the papers in the community relate to establishing the existence of bias [3, 20], some of these papers provide bias mitigation strategies by altering training data [10], trained models [12], or embeddings [2]. Kleinberg et al. [15] and Friedler et al. [11] both compare several different fairness metrics for structured data. Unlike this line of work, our work is focused on unstructured data. In particular, we deal with the bias for the hate speech detection task.

2.2 Handling Bias for Unstructured Data

Little prior work exists on fairness for text classification tasks. Blodgett and O'Connor [3], Hovy and Spruit [13] and Tatman [20] discuss the impact of using unfair natural language processing models for real-world tasks, but do not provide mitigation strategies. Bolukbasi et al. [4] demonstrate gender bias in word embeddings and provide a technique to "de-bias" them, allowing these more fair embeddings to be used for any text-based task. But they tackle only gender-based stereotypes while we perform a very generic handling of various stereotypes.

There is hardly any rigorous work on principled detection and removal of stereotypical bias from text data. Dixon et al.'s work [8] is most closely related to our efforts. They handle stereotypical bias using a dictionary of hand-curated biased words, which is not generalizable and scalable. Also, one popular method for bias correction is data augmentation. But it is challenging to obtain instances which help in bias removal, and which do not sneak in further bias. To address such drawbacks, we design *automated* techniques for bias detection, and *knowledge-based generalization* techniques for bias removal with a focus on hate speech detection.

[1] https://www.perspectiveapi.com/

Sample	Classification prob. (Prediction)	Inference
kat is a woman alice is a woman	0.63 (Hateful) 0.23 (Neutral)	Replacing the name from Kat to Alice should not have resulted in such a difference in prediction probability.
that's just fucking great, man. that's just sad man. shit, that's just sad man.	0.80 (Hateful) 0.24 (Neutral) 0.85 (Hateful)	Use of the word "fucking" somehow makes the sentence hateful. Similarly, use of the word "shit" abruptly increases the hate probability.
rt @ABC: @DEF @GHI @JKL i don't want to fit anything into their little boxes ...	0.80 (Hateful)	In the training dataset, the word "@ABC" is mentioned in 17 tweets of which 13 (76%) are "Hateful". Hence, the classifier associates @ABC with "Hateful"
my house is dirty i gotta go damn, my house is dirty. i gotta go and clean it	0.86 (Hateful) 0.71 (Hateful) 0.99 (Hateful)	The words "dirty" and "gotta" are present heavily in instances labeled as hateful in the training dataset. This resulted in the classifier learning to wrongfully infer the sentence as being hateful.

Table 2: Examples of Incorrect Predictions from our Datasets described in Section 5.1

3 PROBLEM FORMULATION

In this section, we present a few definitions and formulate a formal problem statement.

3.1 Preliminary Definitions

We start by presenting definitions of a few terms relevant to this study in the context of hate speech detection task.

Definition 3.1. Hate Speech: Hate speech is speech that attacks a person or group on the basis of attributes such as race, religion, ethnic origin, national origin, sex, disability, sexual orientation, or gender identity. Its not the same as using certain profane words in text. A sentence can use profane words and still might not be hate speech. Eg, *Oh shit! I forgot to call him*

Definition 3.2. Hate Speech Detection: Hate Speech Detection is a classification task where a classifier is first trained using labeled text data to classify test instances into one of the two classes: Hateful or Neutral.

Definition 3.3. Stereotypical Bias (SB): In social psychology, a stereotype is an over-generalized belief about a particular category of people. However, in the context of hate speech detection, we define SB as an over-generalized belief about a word being Hateful or Neutral.

For example, as shown in Table 2, "kat" and "@ABC" have a bias of being associated with "Hateful" class. Typical stereotype parameters represented as strings could also pertain to stereotypical bias in the context of hate speech detection, e.g. "girls", "blacks", etc.

Definition 3.4. Feature Units: These are the smallest units of the feature space which are used by different text processing methods.

SB in the context of hate speech detection can be studied in terms of various feature units like character n-grams, word n-grams, phrases, or sentences. In this work, although we focus on words as feature units, the work can be easily generalized to other levels of text.

Definition 3.5. Bias Sensitive Word (BSW): A word w is defined as a bias sensitive word for a classifier if the classifier is unreasonably biased with respect to w to a very high degree.

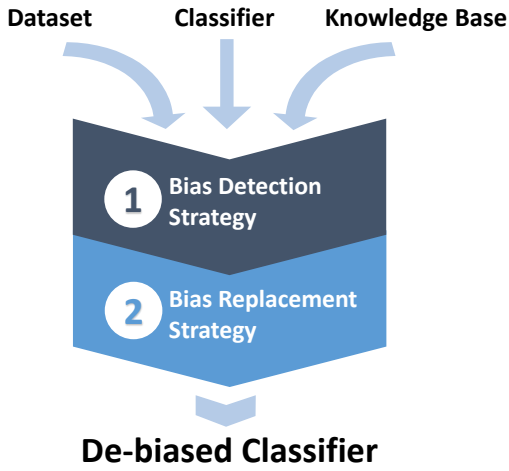


Figure 1: Conceptual Diagram for the proposed 2-Stage De-biasing Framework

For example, if a classifier labels the sentence with just a word "muslims" as "Hateful" with probability 0.81, then we say that the word "muslims" is a BSW for the classifier. Similarly, if a classifier predicts the sentence *'Is Alex acting weird because she is a woman ?'* with much higher probability than *'Is Alex acting weird because he is a man ?'* as Hateful, then the words "woman" and "she" are BSWs for that classifier.

3.2 Problem Formulation

Given: A hate speech detection classifier.
Find:

- A list of BSW words from its training data.
- De-biased dataset to mitigate the impact of BSWs.

We propose a two-stage framework to solve the problem. In the first stage (Detection), we propose biased-model prediction based novel heuristics for bias sensitive word (BSW) detection. Further, in the second stage (Replacement), we present novel bias removal strategies leveraging knowledge-based generalizations. Figure 1 shows the basic architecture diagram of the proposed framework.

4 TWO-STAGE DE-BIASING APPROACH

In this section, we first discuss the first stage of the proposed framework which deals with detection of BSWs. Next, we discuss choices related to bias correction strategies that we could employ. Finally, as part of the second stage, we propose multiple knowledge generalization based BSW replacement strategies.

4.1 Stage 1: Identifying BSWs

Many text-related tasks (e.g., gender profiling based on user-written text, named entity recognition) can extract strong signals from individual word occurrences within the text. However, higher order natural language processing tasks like hate speech detection, sentiment prediction, sarcasm detection cannot depend on strong signals from individual word occurrences; they need to extract signals from a combination of such words. Consider the sentence "Pakistan is a country with beautiful mountains". Here it is good for a named entity recognition classifier to use its knowledge about the significant association between the word "Pakistan" and a label "location". But it is not good for a hate speech detection classifier to use its knowledge about associating the word "Pakistan" with label "Hateful" which it might have unintendedly learned from the training data.

Note that although it is important to identify BSWs, it is important to ensure that obvious hateful words should not be removed while doing bias mitigation. Using a standard abusive word dictionary, one can easily ignore such words from being considered a BSW. To identify stereotypical bias and avoid the classifier from propagating the bias from training data to its parameters, we propose the following three strategies for identifying BSWs.

4.1.1 Manual List Creation. Dixon et al. [8] manually created a set of 50 terms after performing analysis on the term usage across the training dataset. The selected set contains terms of similar nature but with disproportionate usage in hateful comments. Examples: lesbian, gay, bisexual, transgender, trans, queer, lgbt, etc.

4.1.2 Skewed Occurrence Across Classes (SOAC). If a word is present a significant number of times in a particular class (say Hateful), the classifier is highly likely to learn a high probability of classifying a sentence containing that word into that particular class. Hence, we define the skewed occurrence measure to detect such BSWs.

For the hate speech detection task, consider the two classes: Hateful (+) vs Neutral (-). Let $tf(w)$ be the total term frequency of the word w in the training dataset. Let $df(w)$ be the total document frequency, $df^+(w)$ be the document frequency of the word for documents labeled as Hateful, and $df^-(w)$ be the document frequency of the word for documents labeled as Neutral. Then, we can use the procedure mentioned in Algorithm 1 to rank the BSWs.

Algorithm 1 BSW Detection based on 'Skewed Occurrence Across Classes' detection strategy in Training Dataset

1: S is a ordered list of words, tf is term-frequency of these words, df is document-frequency values of these words, ϕ is the cutoff tf threshold.
2: $S^* \leftarrow \{w | w \in S \text{ and } tf(w) > \phi \text{ and } df^+(w) > df^-(w)\}$
3: Sort S^* in non-ascending order by $(df(w), df^+(w)/df(w))$.

Filtering based on just the high frequency would not be enough as that would include stopwords as well, hence we add additional constraints to filter these unnecessary terms.

Essentially, Algorithm 1 ranks a word higher if (1) it occurs with a minimum ϕ frequency in the entire dataset (2) it occurs in many training dataset documents (3) it occurs in Hateful documents much more than in Neutral documents. The drawback of this method is that we need to have access to the exact training corpus.

4.1.3 Skewed Predicted Class Probability Distribution (SPCPD). Let $p(c|w)$ denote the classifier prediction probability of assigning a sentence containing only word w to class c. Let c_ϕ denote a "Neutral" or an "Others" class, or any such "catchall" class. Then, we define the SPCPD score as the maximum probability of w belonging to one of the classes excluding c_ϕ.

$$SPCPD(w) = \max_{\forall c \in C, c \neq c_\phi} p(c|w) \quad (1)$$

Note that a high value of $SPCPD(w)$ means that the classifier has stereotyped the word w to belong to the corresponding class. For example, if a classifier labels the sentence with just a word "muslims" as "Hateful" with probability 0.81, $SPCPD_{muslims}$=0.81. A word w is called bias sensitive word (BSW) if $SPCPD(w) \geq \tau$, where τ is a threshold, usually close to 0.5 (for binary classification).

In settings where Hateful class is further sub-divided into multiple subtypes, using the maximum probability of classification amongst the positive target classes allows us to identify words that force the decision of the model towards a target class. Instead, an ideal classifier should predict a uniform probability distribution on the target classes except for the neutral class. The advantage of this bias detection strategy is that it does not need access to the original training dataset and can be used even if we just have access to the classifier prediction API.

The *SOAC & SPCPD* bias sensitive words detection strategies return an ordered list of BSWs with scores. We can pick the top few BSWs from each of these strategies, and subject them to bias removal as detailed in the remainder of this section.

Both the *SOAC & SPCPD* bias sensitive words detection strategies can be used for n-grams as well. Considering that higher word-n-grams would require considerable computation power, we focus on unigrams only. Additionally, using prediction on single-word documents might seem trivial but later, in Section 5, we show that such a simple intuitive strategy can be quite effective in identifying BSWs.

4.2 Choices for Bias Correction Strategy

For removing bias from a biased training dataset, we can do the following:

(1) **Statistical Correction:** This includes techniques that attempt to uniformly distribute the samples of every kind in all the target classes, augmenting the train set with samples to balance the term usage across the classes. "Strategic Sampling" is a simple statistical correction technique which involves strategically creating batches for training the classifier with aim of normalizing the training set. Data Augmentation is another technique that encourages adding new examples instead of performing sampling with replacement.

Example: Hatespeech corpus usually contains instances where 'women' are getting abused in huge proportion as compared to 'men' getting abused. Statistical correction can be useful in balancing this disproportion.

(2) **Data Correction:** This technique focuses on converting the samples to a simpler form by reducing the amount of information available to the classifier. This strategy is based on the idea of Knowledge generalization, when the classifier should be able to decide if the text is hateful or not based on the generalized sentence, not considering the private attribute information, as these attributes should not be a part of decision making of whether the sentence is hateful or not. Some popular examples of private attributes include Gender, Caste, Religion, Age, Name etc.
Example: For a sentence, *That Indian guy is an idiot and he owns a Ferrari*, private attribute information should not be used while learning to classify as abuse. The new generalized sentence after removing private attributes should look like: *That <NATIONALITY> guy is an idiot and he owns a <CAR>*.

(3) **Model Correction or Post-processing:** Rather than correcting the input dataset or augmenting, one can either make changes to the model like modifying word embeddings or post-process the output appropriately to compensate for the bias.
Example: A simple technique would involve denying predictions when the sentence contains private attributes to reduce false positives.

In this paper, we direct our effort towards effective data correction based methods to counter bias in learning. This is mainly because of the following reasons: (1) for statistical correction, it is challenging to obtain instances which help in bias removal, and which do not sneak in further bias. In other words, the additional words in augmented instances can skew the distribution of existing words, inducing new bias. (2) for model correction, the strategy could be very specific to the type of classifier model.

We consider dataset augmentation or statistical correction as a baseline strategy. In order to prevent differential learning, we can add additional examples of neutral text in the training set containing the BSWs forcing the classifier to learn features which are less biased towards these BSWs. Dixon et al. [8] use this strategy to balance the term usage in the train dataset thereby preventing biased learning.

4.3 Stage 2: Replacement of BSWs

Next, we propose multiple knowledge generalization based BSW replacement strategies as follows.

4.3.1 Replacing with Part-of-speech (POS) tags. In order to reduce the bias of the BSWs, we can replace each of these words with their corresponding POS tags. This process masks some of the information available to the model during training inhibiting the classifier from learning bias towards the BSWs.

Example: Replace the word 'Muhammad' with POS tag 'NOUN' in the sentence - *Muhammad set the example for his followers, and his example shows him to be a cold-blooded murderer.* POS replacement substitutes specific information about the word 'Muhammad' from the text and only exposes partial information about the word, denoted by its POS tag 'NOUN', forcing it to use signals that do not give significant importance to the word 'Muhammad' to improve its predictions.

4.3.2 Replacing with Named-entity (NE) tags. Consider the sentence - *Mohan is a rock star of Hollywood.* The probability of the sentence being hateful or not does not depend on the named entities 'Mohan' and 'Hollywood'. However, classifiers are prone to learn parameters based on the presence of these words. Hence, in this strategy, we replace the named entities with their corresponding entity type tag. This technique generalizes all the named-entities thus preventing information about the entities from being exposed to the classifier.

We use the standard set of NE tags which includes PERSON, DATE, PRODUCT, ORGANIZATION etc.

4.3.3 K-Nearest Neighbour. Given a word embedding space (say GloVe [19]) and a BSW w, we find K nearest neighbors for w in the embedding space. Let the $N_K(w)$ indicate the set of these K nearest neighbors plus the word w. For every occurrence of w in the training dataset, we randomly replace this occurrence with a word selected from $N_K(w)$. The intuition behind the strategy is as follows. If we replace all instances of a word with another word then it may not decrease the bias. But if we replace some sentences with other variants of the word, then it should add some variance in the sentences, and hence reduce the bias.

This strategy should serve as an equivalent of adding additional examples in the text to reduce the class imbalance.

4.3.4 Knowledge generalization using lexical databases. Replacing surface mentions of entities with their named entity types holds back important information about the content of the sentence; it generalizes the content too much. A more general approach would involve replacing BSWs with the corresponding generalized counterparts. A major reason for learning bias is because of differential use of similar terms. Consider the terms, 'women' and 'man'. For the domain of hate speech, the term 'women' would be present more frequently as compared to 'man', even though they represent similar, if not same, groups.

We leverage WordNet [17] database to obtain semantically related words and the lexical relations in order to find a suitable candidate for replacement. In this strategy, words are replaced with their corresponding hypernym as obtained from the WordNet. A hypernym is a word with a broad meaning constituting a category into which words with more specific meanings fall; a superordinate. For example, *colour is a hypernym of red.* A BSW might have multiple hypernyms and therefore different paths leading to a common root in WordNet. We perform a Breadth-First-Search (BFS) over the sets of synonyms, known as synsets, from the child node as obtained from the WordNet for a BSW. A synset contains lemmas, which are the base form of a word. We then iterate over all the lemmas in the synset to obtain the replacement candidate. This allows us to identify the lowest parent in the hypernym-tree which is already present in the training dataset vocabulary.

We try out all the words in the synonym-set to select a suitable candidate for replacement. Replacing a word with a parent which is not present in the text won't be of any importance since it will not help reduce bias when learning the classifier. Hence, we ignore such

words. Not every word can be located in WordNet, especially the ones obtained from casual conversational forums like Twitter. To obtain better performance, we use spell correction before searching for the suitable synset in the WordNet.

4.3.5 Centroid Embedding. All the above techniques exploit the information in the text space either by using NLP tools or lexical databases. In order to leverage the complex relationships in the language in embedding space, we can use the existing word embeddings which sufficiently accurately capture the relationship between different words.

In the centroid embedding method, we replace a bias sensitive word with a dummy tag whose embedding we compute as follows. We find the POS tag for that occurrence of the word, and then find similar words from the word embeddings with similar POS usage. Further, we compute the centroid of the top k neighbors, including the original word to obtain the centroid embedding. We set k to 5 in our experiments.

The centroid represents the region of the embedding space to which the bias sensitive word belongs. So, rather than using the exact point embedding of the word, we smooth it out by using the average embedding of the neighborhood in which the point lies. This strategy is a continuous version of the WordNet-based strategy as the generalizations happen in the embedding space rather in text or word space. Additionally, computing the centroid of sufficient number of words helps in obtaining bias-free vector representations giving improved performance (as shown in Section 5).

5 EXPERIMENTS

In this section, we first discuss our datasets and metrics. Next, we present comparative results with respect to hate speech detection accuracy and bias mitigation using different BSW detection and replacement strategies.

5.1 Datasets

In order to measure the efficacy of the proposed techniques, we use the following datasets to evaluate the performance.

WikiDetox Dataset [22]: We use the dataset from the Wikipedia Talk Pages containing 95,692 samples for training, 32,128 for development and 31,866 for testing. Each sample in the dataset is an edit on the Wikipedia Talk Page which is labeled as 'Neutral' or 'Hateful'.

Twitter Dataset [7]: We use the dataset consisting of hateful tweets obtained from Twitter. It consists of 24,783 tweets labeled as hateful, offensive or neutral. We combine the hateful and offensive categories; and randomly split the obtained dataset into train, development, and test in the ratio 8:1:1 to obtain 19,881 train samples, 2,452 development samples, and 2,450 test samples.

"Wiki Madlibs" Eval Dataset [8]: For the Dataset augmentation based bias mitigation baseline strategy, we follow the approach proposed by Dixon et al. [8]. For every BSW, they manually create templates for both hateful instances and neutral instances. Further, the templates are instantiated using multiple dictionaries resulting into ~77k examples with 50% examples labeled as Hateful. For example, for a BSW word "lesbian", a template for a neutral sentence could be "Being lesbian is ⟨adjectivePositive⟩." where ⟨adjectivePositive⟩ could be a word from dictionary containing "great, wonderful, etc." A template for a Hateful sentence could be "Being lesbian is ⟨adjectiveNegative⟩." where ⟨adjectiveNegative⟩ could be a word from dictionary containing "disgusting, terrible, etc.".

5.2 Metrics

5.2.1 ROC-AUC Accuracy. For the hate speech detection task, while we mitigate the bias, it is still important to maintain a good task accuracy. Hence, we use the standard ROC-AUC metric to track the change in the task accuracy.

5.2.2 Pinned AUC Equality Difference. In order to empirically measure the performance of the de-biasing procedures, Dixon et al. [8] proposed the Pinned AUC Equality Difference ($pAUC$) metric to quantify the bias in learning for the dataset augmentation strategy only.

For a BSW w, let D_w indicate the subgroup of instances in the "Wiki Madlibs" eval dataset containing the term w. Let $\overline{D_w}$ be the instances which do not contain the term w such that $|D_w| = |\overline{D_w}|$. Further, the classifier is used to obtain predictions for the entire subgroup dataset $\{D_w \cup \overline{D_w}\}$. Let $AUC(w)$ denote the subgroup AUC obtained using these predictions. Further, let AUC denote the AUC obtained using predictions considering the entire "Wiki Madlibs" dataset $\{\sum_{\forall w}(D_w \cup \overline{D_w})\}$. Then, a good way to capture bias is to look at divergence between subgroup AUCs and the overall AUC. Dixon et al. [8] thus define the Pinned AUC Equality Difference ($pAUC$) metric as follows,

$$pAUC = \sum_{w \in BSW} |AUC - AUC(w)| \tag{2}$$

A lower $pAUC$ sum represents less variance between performance on individual term subgroups and therefore less bias. However, the $pAUC$ metric suffers from the following drawbacks:

- Computation of $pAUC$ requires creating a synthetic dataset with balanced examples for each bias sensitive word (BSW) w, making it challenging to compute and scale with the addition of new BSWs as it would require an addition of new examples with desired labels.
- The "Wiki Madlibs" Eval dataset is synthetically generated based on templates defined for BSWs. The pinning to the AUC obtained on such a dataset is quite ad hoc. An ideal metric should be independent of such relative pinning.

5.2.3 Pinned Bias (PB). To address the above-mentioned drawbacks with $pAUC$, we propose a family of metrics called Pinned Bias Metric (PB) that is better able to capture the stereotypical bias. Note that small values of PB are preferred.

Let T represent the set of BSW words. Let $p(\text{"}Hateful\text{"}|w)$ be the prediction probability for a sentence which contains only word w. We define PB as the absolute difference in the predictions 'pinned' to some predictive value for the set T. This captures the variation in Bias Probability of the classifier for the given set of words with respect to the pinned value. Keeping these intuitions in mind, we define the general PB metric as follows.

Detection Strategy	WikiDetox Dataset	Twitter Dataset
Skewed Occurrence Across Classes	get, youre, nerd, mom, yourself, kiss, basement, cant, donkey, urself, boi	lol, yo, my, ya, girl, wanna, gotta, dont, yall, yeah
Skewed Predicted Class Probability Distribution	pissed, gotta, sexist, righteous, kitten, kidding, snake, wash, jew, dude	feelin, clap, lovin, gimme, goodnight, callin, tryin, screamin, cryin, doin

Detection Strategy	WikiDetox Dataset
Manually curated set of words	lesbian, gay, bisexual, transgender, trans, queer, lgbt, lgbtq, homosexual, straight, heterosexual, male, female, nonbinary, african, african american, black, white, european, hispanic, latino, latina, latinx, mexican, canadian, american, asian, indian, middle eastern, chinese, japanese, christian, muslim, jewish, buddhist, catholic, protestant, sikh, taoist, old, older, young, younger, teenage, millenial, middle aged, elderly, blind, deaf, paralyzed

Table 3: BSWs obtained using the Bias Detection strategies on the WikiDetox and Twitter dataset. The words in the order from left to right are ranked based on the strategy specific ranking criteria.

Detection Strategy →	Skewed Occurrence Across Classes				Skewed Predicted Class Prob. Distribution			
Replacement Strategy ↓	ROC-AUC	PB_{mean}	PB_{sym}	PB_{asym}	ROC-AUC	PB_{mean}	PB_{sym}	PB_{asym}
Biased	0.955	0.0174	0.4370	0.4370	0.958	<u>0.0848</u>	0.2553	0.2553
NER tags	<u>0.961</u>	<u>0.0140</u>	0.4542	0.4542	0.957	0.1266	0.1869	0.1753
POS tags	0.960	0.1271	<u>0.1209</u>	<u>0.0901</u>	**0.961**	0.1235	<u>0.1478</u>	<u>0.1110</u>
K-Nearest Neighbour	0.957	0.1323	0.3281	0.3248	**0.961**	0.1311	0.2421	0.2370
WordNet - 0	0.960	0.2238	0.3081	0.2906	0.954	0.1105	0.2079	0.2016
WordNet - 1	0.961	0.2206	0.3167	0.2951	0.957	0.1234	0.2123	0.2016
WordNet - 2	0.953	0.1944	0.3016	0.2870	**0.961**	0.1590	0.2059	0.1782
WordNet - 3	**0.963**	0.1824	0.3420	0.3259	**0.961**	0.1255	0.1889	0.1787
WordNet - 4	0.957	0.0960	0.3851	0.3831	0.956	0.1308	0.2053	0.1928
WordNet - 5	0.956	0.0808	0.3909	0.3873	0.959	0.1089	0.1909	0.1844
Centroid Embedding	0.958	**0.0000**	**0.0204**	**0.0000**	<u>0.960</u>	**0.0709**	**0.0578**	**0.0439**

Table 4: Hate Speech Detection Accuracy and Bias Mitigation Results Across Multiple Detection and Replacement Strategies on Twitter Dataset. The best replacement strategy is highlighted in *bold* while the 2^{nd} best is *<u>underlined</u>* for each metric.

$$PB = \sum_{w \in T} \frac{|p(\text{“}Hateful\text{”}|w) - \phi|}{|T|} \tag{3}$$

, where ϕ is the pinned value which differs for different metrics in the PB family. Next, we define three specific members of the PB family as follows.

- PB_{mean}: For a similar set of terms, it is important to have consistency in the predictions. To capture this deviation in prediction from the general classifier consensus, we pin the metric with respect to the mean prediction. For this metric, ϕ in Eq. 3 is set as follows.

$$\phi = \sum_{w \in T} \frac{p(\text{“ Hateful ”}|w)}{|T|} \tag{4}$$

- PB_{sym}: PB_{mean} is not effective in situations where the terms are of varied nature or from diverse domains. Also pinning it relative to the general consensus of the classifier again can introduce errors.
  Hence, for this metric, ϕ in Eq. 3 is set as ϕ=0.5 for binary classification. In case of k classes, ϕ can be set to $\frac{1}{k}$.
- PB_{asym}: Intuitively, prediction probability of 0.70 being hateful as compared to probability of 0.30 being hateful for a BSW should not be considered equal as a degree of being hateful is much more sensitive as compared to a degree of being neutral.
  A high value of $p(\text{“}Hateful\text{”}|w)$ should be penalized much more than a value less than 0.50 (for binary classification; $\frac{1}{k}$ for k-class classification). Hence, for this metric, ϕ in Eq. 3 is set as follows.

$$\phi = \min(p(\text{“}Hateful\text{”}|w),\ 0.5) \tag{5}$$

The PB metrics, in general, represent the average deviation of the probability scores from the ideal pinning value. Eg, A score of $PB_{sym} = 0.2$ would mean that the probabilities have an average deviation of 0.2 from the ideal value of 0.5. The PB metrics can be modified, if needed, to incorporate the bias in n-grams.

5.3 Experimental Settings

We perform multiple experiments to test different BSW identification and bias mitigation strategies. We experiment with multiple classification methods like Logistic Regression, Multi-layer Perceptron (MLP), Convolutional Neural Network (CNN), an ensemble of classifiers. Since we obtained similar results irrespective of the classification method, we report results obtained using CNNs in this section.

We use identical Convolutional Neural Network (CNN) [8] implemented in Keras [6] for all the experiments. Fig. 2 shows the architecture diagram of the neural network. We use 100D GloVe [19]

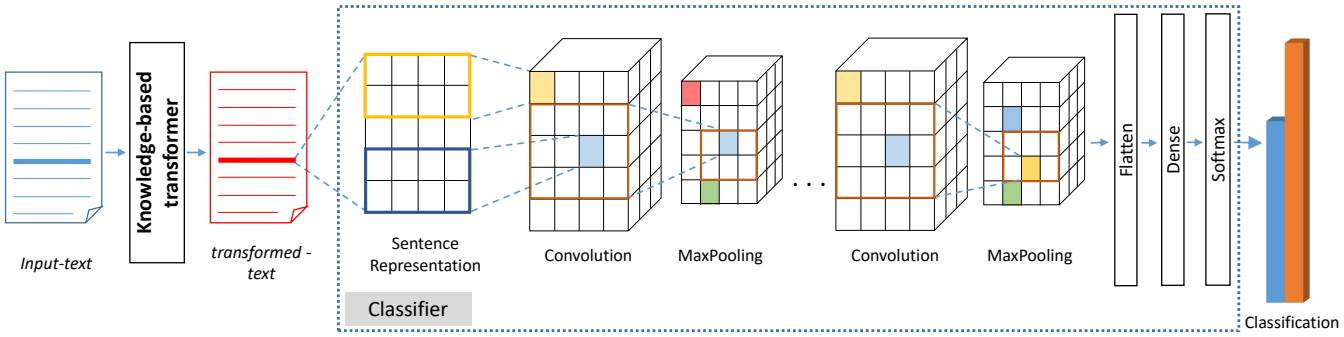


Figure 2: Model Architecture Diagram

embeddings to encode text and CNN architecture with 128 filters of size 5×5×5 for each convolution layer, dropout of 0.3, max-sequence length as 250, categorical-cross-entropy as the loss function and RMSPROP as optimizer with learning-rate of 0.00005. We use the best parameter setting as described in Dixon et al. [8] and train the classifiers for our task with batch size 128 and early stopping to a maximum of 20 epochs, keeping other hyperparameters constant throughout the experiments.

We use the spaCy [2] implementation of POS tagger and NER tool. More details about the experiments can be found in the source code [3].

5.4 Results

	ROC-AUC	$pAUC$
Biased	0.948	2.785
Wiki De-bias [8]	0.957	1.699
NER tags	0.945	2.893
POS tags	<u>0.958</u>	<u>1.053</u>
K-Nearest Neighbour	0.910	3.480
WordNet - 0	0.943	2.781
WordNet - 1	0.949	2.530
WordNet - 2	<u>0.958</u>	1.634
WordNet - 3	0.954	2.068
WordNet - 4	0.951	2.390
WordNet - 5	0.954	1.992
Centroid Embedding	**0.971**	**0.629**

Table 5: Bias Mitigation Results using pAUC metric and ROC-AUC across Multiple Replacement Strategies on Wiki Madlibs Dataset. The best replacement strategy is highlighted in *bold* while the 2^{nd} best is <u>*underlined*</u> for each metric.

5.4.1 BSWs identified by Various Strategies. Table 3 shows the list of BSWs obtained using different Bias Detection strategies. The words obtained from both the SOAC and SPCPD detection strategies are not obvious and differ from the manual set of words as the distribution of the terms in the training set can be arbitrary, motivating us to design detection strategies. In case of the Twitter Dataset, in Table 3, we do not show results for the 'Manual List Creation' detection strategy because we do not have a manual dataset of bias sensitive words for Twitter. The simple strategy of detecting biased words using Skewed Predicted Class Probability Distribution (SPCPD) is effective and works for arbitrary classifiers as well. The words obtained from the SPCPD detection strategy are not very obvious but the prediction on sentences that contain those words have very high discrepancy than expected prediction scores.

The BSWs obtained using the detection strategies are better than the ones obtained using the 'Manual List Creation' strategy because of the high PB_{asym} scores for the 'Biased' classifier for different detection strategies as tabulated in Table 6. Additionally, the PB_{asym} scores for BSWs from the manual list are very low in spite of a large number of words which suggests that the classifier is not biased towards these terms. For the Twitter dataset, the PB_{asym} score for the SOAC detection strategy is the lowest, but in WikiDetox where the training dataset size is large, the skewness in distribution reduces resulting in the PB_{asym} scores for the SPCPD detection strategy to be the lowest.

5.4.2 Effectiveness of various Bias Detection-Replacement Strategies. Tables 4, 5 and 6 show the hate speech detection accuracy and bias mitigation results across various bias detection and bias replacement strategies (averaged over 10 runs because of high variance) for the Twitter, Wiki Madlibs and WikiDetox datasets. 'Biased' replacement strategy indicates the biased classifier, i.e., the one where no bias has been removed. For Wiki De-bias bias removal strategy, we do not have results for non-manual detection strategies since they require template creation (like 'Wiki Madlibs') and dataset curation. We train the 'Wiki De-bias' [8] baseline using the additional data containing 3,465 non-toxic samples as part of the dataset augmentation method as proposed in the original paper. For Wiki Madlibs dataset (Table 5), we show results using only manual list creation but across multiple bias replacement strategies, using $pAUC$ metric only since it is meant for this combination alone. WordNet-i denotes the WordNet strategy where the starting synset level for finding hypernyms was set to i levels up compared to the level of the bias sensitive word. For Tables 4 and 6, we use the ROC-AUC scores for comparing the general model performance while the PB_{asym} metric for bias evaluation. We do not use $pAUC$ for these tables because they need manual generation of a dataset like Wiki

[2] https://spacy.io/
[3] http://bit.ly/stereotypical-bias-removal

Detection Strategy →	Manual List Creation				Skewed Occurrence Across Classes				Skewed Predicted Class Prob. Distribution			
Replacement Strategy ↓	ROC-AUC	PB_{mean}	PB_{sym}	PB_{asym}	ROC-AUC	PB_{mean}	PB_{sym}	PB_{asym}	ROC-AUC	PB_{mean}	PB_{sym}	PB_{asym}
Biased	0.957	0.0806	0.1223	0.0144	0.952	0.1424	0.1473	0.0666	0.956	0.1826	0.2101	0.0505
Wiki Debias [8]	**0.958**	0.0491	0.1433	0.0058	NA	NA	NA	NA	NA	NA	NA	NA
NE tags	0.956	0.0833	0.1269	0.0137	**0.959**	0.1741	0.2252	0.0421	0.955	0.1533	0.2118	0.0304
POS tags	0.953	0.0251	0.0540	0.0020	0.954	0.1614	0.2033	0.0481	**0.958**	0.0789	0.3656	**0.0000**
K-Nearest Neighbour	0.937	0.0554	0.1120	0.0023	0.936	**0.1154**	**0.1196**	0.0431	0.934	**0.0618**	**0.1284**	0.0035
Wordnet - 0	0.956	0.0765	0.1188	0.0121	0.946	0.1445	0.1620	0.0435	0.951	0.1835	0.2602	0.0283
Wordnet - 1	0.956	0.0756	0.1201	0.0133	0.954	0.1820	0.2132	0.0529	**0.958**	0.1191	0.3401	0.0002
Wordnet - 2	0.956	0.0735	0.1182	0.0110	0.952	0.1767	0.1986	0.0550	0.955	0.1453	0.3281	0.0037
Wordnet - 3	0.953	0.0708	0.1151	0.0098	0.941	0.1297	0.1622	**0.0326**	0.952	0.1699	0.3029	0.0121
Wordnet - 4	0.953	0.0733	0.1133	0.0097	0.946	0.1398	0.1570	0.0393	0.955	0.1751	0.3150	0.0123
Wordnet - 5	0.948	0.0653	0.1084	0.0057	**0.959**	0.1766	0.2336	0.0468	0.955	0.1561	0.3475	0.0060
Centroid Embedding	0.955	**0.0000**	**0.0499**	**0.0000**	0.953	0.1803	0.1918	0.0535	0.957	0.0774	0.1949	**0.0000**

Table 6: Hate Speech Detection Accuracy and Bias Mitigation Results Across Multiple Detection and Replacement Strategies on WikiDetox Dataset. The best replacement strategy is highlighted in *bold* while the 2^{nd} best is *underlined* for each metric.

Madlibs. Finally, note that the scores in the Table 4 and Table 6 are vertically grouped based on different detection strategies which involve varying set of BSWs making the results non-comparable across the detection strategies in general.

As seen in Table 5, for the Wiki Madlibs dataset, Centroid Embedding strategy performs the best beating the state-of-the-art technique (Wiki De-Bias) on both ROC-AUC as well as pAUC metrics. Additionally, POS tags and WordNet-based strategies also perform better than the Wiki De-bias strategy on the *pAUC* metric with just a minor reduction in $ROC - AUC$ accuracy. The POS tags strategy performs better because the replaced words mostly belong to the same POS tag resulting in high generalization.

For the Twitter dataset, Centroid Embedding replacement method works best across all the detection strategies and across different metrics. The overall general classification performance for all the strategies remain almost similar, though the WordNet-3 replacement strategy works best. WordNet based replacement strategy is not as effective as the PB_{asym} scores gradually increase with the increase in generalization, due to the absence of slang Twitter words in WordNet. The WordNet-based replacement strategies also show decreasing PB_{asym} scores as the strategy is unable to find a suitable generalized concept that is already present in the Twitter dataset because of its small vocabulary size.

The WordNet and Centroid Embedding replacement strategies work best for the WikiDetox dataset. For the WordNet strategy, the best results vary based on detection strategy, but it performs either the best or gives comparable results to the best strategy. It also has an added benefit of deciding on the level of generalization which can be tweaked as per the task. For "manual list creation" detection strategy, Centroid Embedding provides unexpectedly low *PB* scores because most of the words in the list are nouns. The non-zero PB_{mean} metric in case of manual-list-creation detection strategy along with POS tags replacement strategy is due to the fact that the words like 'gay' are marked as Adjective instead of Noun, which is expected.

Finally, note that bias mitigation usually does not lead to any significant change in the ROC-AUC values, if performed strategically, while generally, it reduces bias in the learning.

5.4.3 Details of WordNet-based bias removal strategy. Table 7 shows WordNet-based replacement examples for the manual set of words with different values of the start synset level.

Original Word	**Replaced Words**					
	Level 0	**Level 1**	**Level 2**	**Level 3**	**Level 4**	**Level 5**
sikh	disciple		follower	person	cause	entity
homosexual	homosexual	person	cause	entity		object
queer	faggot	homosexual		person	cause	entity
gay	homosexual	person	cause	entity		object
straight	homosexual	person	cause	collection	entity	object
muslim	person			cause	entity	
deaf	deaf	people	group	abstraction	entity	-
latino	inhabitant			person	cause	entity

Table 7: WordNet based bias correction examples for the 'Manual List Creation' detection strategy with different start Synset Levels

Fig. 3 shows details about WordNet based replacement. As we traverse from right-to-left along the edges, we obtain generalizations for each of the words. The figure highlights two sets of similar words where one set (denoted by ▭) is highly prone towards getting stereotypically biased, while all the highlighted terms (denoted by ⬭) represent "a group of people with specific attributes". Each node represents the term which can be used for BSW replacement. In the case of multiple generalizations, the selection was made randomly.

Wordnet-based strategy often leads to the best results across our datasets. However, the results could be improved further using an appropriate normalization scale that captures generalization to different levels. For example, *Hindu* requires three levels of generalization, while *Muslim* requires two (see Fig. 3). This requires a generalization scheme that considers the inconsistency in the sibling nodes and performs differential normalization.

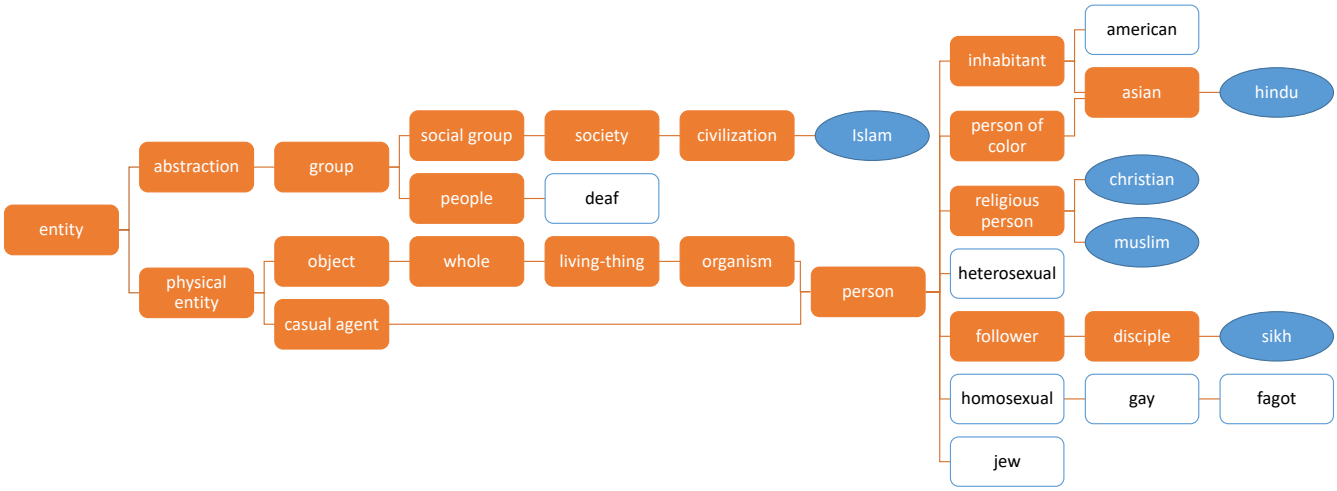


Figure 3: Horizontal representation of the WordNet Hypernym Hierarchy for BSWs.

5.4.4 Case Studies. In order to evaluate the effectiveness of the Bias Detection strategy in identifying BSWs, we perform an experiment on a publicly accessible classifier, the Perspective API [4]. We leverage the vocabulary of size 400k of GloVe vectors and obtain hate (toxicity) scores for each word in the vocabulary and rank them using the Skewed Predicted Class Probability Distribution Bias Detection strategy. This allows us to identify the set of bias sensitive words allowing us to create examples to test the quality of the obtained words. Some of these examples are tabulated in Table 1 where the API gives incorrect scores. The obtained examples show the effectiveness of the obtained BSWs.

Further, Table 8 shows some examples which were wrongly predicted as being "hateful" by the original 'Biased' classifier and the corresponding prediction scores after bias removal using the above strategies. These simple manually created examples clearly show the efficacy of our proposed method.

Examples	**Toxicity Predictions**	
	before bias removal	**after bias removal**
are you a **nerd** ?	0.928	0.294
he hit the **donkey**	0.289	0.021
he is a **gay**	0.786	0.493
One of John brothers was **homosexual** while the other is a **black transgender**	0.690	0.048
katrina is a **female african** who is **sikh**	0.782	0.535

Table 8: Examples for predictions on sentences before and after the classifier had undergone the bias correction stage using the Centroid Embedding strategy for the manually constructed sentences containing identified BSWs. Bias Sensitive Words are marked in bold. The ideal prediction probabilities in all the cases should be less than 0.5

[4]https://www.perspectiveapi.com/

5.4.5 Unigrams versus higher order ngrams. In this paper, we experimented with unigrams only. From our experiments, we observed that the SB bias in unigrams propagates to word n-grams and char-n-grams as well. E.g., SB towards a word 'women' propagates to the char n-grams 'wom, omen, wome' as the distribution of char n-grams is similar to that of the corresponding word and also similar to the word n-grams, *women are*, *womens suck*, etc. Thus, considering unigrams is reasonably sufficient. That said, the PB metrics rely on the probability scores and thus can be easily used for bias sensitive *n-grams* as well.

6 CONCLUSIONS

In this paper, we discussed the problem of bias mitigation for the hate speech detection task. We proposed a two-stage framework to address the problem. We proposed various heuristics to detect bias sensitive words. We also proposed multiple novel knowledge generalization based strategies for bias sensitive word replacement which can be extended to other short-text classification tasks as well. Using two real-world datasets, we showed the efficacy of the proposed methods. We also empirically show that the data-correction based removal techniques can reduce the bias without reducing the overall model performance.

We also demonstrate the effectiveness of these strategies by performing the experiment on Google Perspective API and perform a qualitative analysis to support our observations using multiple examples.

REFERENCES

[1] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 759–760.

[2] Alex Beutel, Jilin Chen, Zhe Zhao, and Ed H Chi. 2017. Data decisions and theoretical implications when adversarially learning fair representations. *arXiv preprint arXiv:1707.00075* (2017).

[3] Su Lin Blodgett and Brendan O'Connor. 2017. Racial Disparity in Natural Language Processing: A Case Study of Social Media African-American English. *arXiv preprint arXiv:1707.00061* (2017).

[4] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*. 4349–4357.

[5] Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science* 356, 6334 (2017), 183–186.

[6] François Chollet et al. 2015. Keras. https://keras.io.

[7] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. (2017).

[8] Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2017. Measuring and Mitigating Unintended Bias in Text Classification. (2017).

[9] Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015. Hate Speech Detection with Comment Embeddings. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15 Companion)*. ACM, New York, NY, USA, 29–30. https://doi.org/10.1145/2740908.2742760

[10] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 259–268.

[11] Sorelle A Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. 2016. On the (im) possibility of fairness. *arXiv preprint arXiv:1609.07236* (2016).

[12] Moritz Hardt, Eric Price, Nati Srebro, et al. 2016. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*. 3315–3323.

[13] Dirk Hovy and Shannon L Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 591–598.

[14] Aditya Khosla, Tinghui Zhou, Tomasz Malisiewicz, Alexei A Efros, and Antonio Torralba. 2012. Undoing the damage of dataset bias. In *European Conference on Computer Vision*. Springer, 158–171.

[15] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. 2016. Inherent trade-offs in the fair determination of risk scores. *arXiv preprint arXiv:1609.05807* (2016).

[16] Irene Kwok and Yuzhou Wang. 2013. Locate the Hate: Detecting Tweets against Blacks.. In *AAAI*.

[17] George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38, 11 (Nov. 1995), 39–41. https://doi.org/10.1145/219717.219748

[18] Alexandra Olteanu, Emre Kiciman, and Carlos Castillo. 2018. A Critical Review of Online Social Data: Biases, Methodological Pitfalls, and Ethical Boundaries. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 785–786. https://doi.org/10.1145/3159652.3162004

[19] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[20] Rachael Tatman. 2017. Gender and Dialect Bias in YouTube's Automatic Captions. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*. 53–59.

[21] Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*. 88–93.

[22] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1391–1399.

Personalized Bundle List Recommendation

Jinze Bai
Key Laboratory of High Confidence Software Technologies, EECS, Peking University
baijinze@pku.edu.cn

Chang Zhou*
Alibaba Group
ericzhou.zc@alibaba-inc.com

Junshuai Song
Key Laboratory of High Confidence Software Technologies, EECS, Peking University
songjs@pku.edu.cn

Xiaoru Qu
Key Laboratory of High Confidence Software Technologies, EECS, Peking University
quxiaoru@pku.edu.cn

Weiting An
Alibaba Group
weiting.awt@alibaba-inc.com

Zhao Li
Alibaba Group
lizhao.lz@alibaba-inc.com

Jun Gao
Key Laboratory of High Confidence Software Technologies, EECS, Peking University
gaojun@pku.edu.cn

ABSTRACT

Product bundling, offering a combination of items to customers, is one of the marketing strategies commonly used in online e-commerce and offline retailers. A high-quality bundle generalizes frequent items of interest, and diversity across bundles boosts the user-experience and eventually increases transaction volume. In this paper, we formalize the personalized bundle list recommendation as a structured prediction problem and propose a bundle generation network (BGN), which decomposes the problem into quality/diversity parts by the determinantal point processes (DPPs). BGN uses a typical encoder-decoder framework with a proposed feature-aware softmax to alleviate the inadequate representation of traditional softmax, and integrates the masked beam search and DPP selection to produce high-quality and diversified bundle list with an appropriate bundle size. We conduct extensive experiments on three public datasets and one industrial dataset, including two generated from co-purchase records and the other two extracted from real-world online bundle services. BGN significantly outperforms the state-of-the-art methods in terms of quality, diversity and response time over all datasets. In particular, BGN improves the precision of the best competitors by 16% on average while maintaining the highest diversity on four datasets, and yields a 3.85x improvement of response time over the best competitors in the bundle list recommendation problem.

KEYWORDS

Bundle Recommendation; Bundle Generation; Diversity

*Corresponding Author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313568

ACM Reference Format:
Jinze Bai, Chang Zhou, Junshuai Song, Xiaoru Qu, Weiting An, Zhao Li, Jun Gao. 2019. Personalized Bundle List Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313568

1 INTRODUCTION

A bundle is a collection of products or services for sale as a whole, and the bundle list recommendation is to recommend a personalized list of bundles. Traditional business domains, e.g., communication services, offline retailers and supermarkets, take bundling as a critical marketing strategy, and they usually make bundles by human insights or non-personalized mining methods [2]. Modern e-commerce websites and online service business, e.g., Amazon, Taobao, Steam and Netflix, also deploy new applications [22, 32], which recommend and sale a list of bundles rather than a list of items.

Customers, sellers, as well as the recommendation platform, could benefit from the personalized bundle recommendation service. For customers, high-quality bundles broaden user interests and indicate the complementary products directly in the shortest path after purchase. It is well known in classic marketing research that carefully-designed bundle brings mutual promotions.

For sellers, bundling increases per customer transaction. The average order size for walmart.com is 2.3 [36]. By attaching some highly correlated products, the seller can launch more products and increase gross merchandise volume (GMV). Besides, buying a bundle of products may cost less than buying each product separately for both customers and sellers. For example, it is a common practice to waive shipping fee if the items in one order exceed a certain amount in e-commerce websites.

For the recommendation system, bundling is a more efficient way to organize and present products. The recommendation list is structured by bundles where the products are highly correlated



Figure 1: Bundle List Recommendation Example

in contrast to the list of items, and multiple diversified bundles avoid monotonous choices for users. The recommendation system shows bundle as a whole and thus presents more products per unit area. It is crucial to reduce swipe-up operations for improving user experiences for the mobile application.

In Figure 1, we illustrate a bundle list example. The items in the bundle should be highly correlated. Besides, diversity across bundles becomes a critical metric as an interpretable, business-oriented goal, which is independent of click-through rate (CTR) and precision, trying to avoid tedious choices and quantify the user-experience in bundle recommendation system. The identical high-quality item tends to be generated in different bundles [22], which compose the confusing and monotonous bundle list, so there is a trade-off between precision/likelihood and diversity. Other meaningful problems, e.g., how to set the price discount for maximizing the profit, how to compose the picture of the bundle, are beyond the scope of this paper and can be the future work. Our paper focuses on how to generate high-quality and diversified bundles.

There are several challenges for the personalized bundle recommendation problem. First, the recommendation system should generate high-quality bundles automatically and flexibly without manual involvement. Frequent itemsets mining algorithms provide high-quality bundling but are not personalized. For modeling the distribution of bundle, we use the co-purchase data or pre-defined bundles to supervise the quality of bundles with the user context. However, the sequence generation methods suffer from inadequate representation for rich features in bundle recommendation scenario, because the weight matrix of traditional softmax only considers the item_id. Besides, in practice, we may need the larger size of bundles for the purpose of sellers' promotion, but seq2seq usually suffers from generating short sequences [14].

Second, diversity across bundles emerges as a critical metric for bundle recommendation. Different bundles with the same high-quality item may reach high ranking scores at the same time, so duplicated items tend to be seen in these generated bundle list [22]. Besides duplication, similar items due to the similarity of features may produce confusing and tedious bundle list and decrease user interest. A bundle recommendation system should consider the cross-bundle diversity considering both duplication and similarity of items. Here, the diversity is a explicit metric instead of some hidden relationships of similarity learned from data. The explicit similarity pre-defined by side information and prior knowledge provides the exact signal for the purpose of measurement and adjustment. We use the determinantal point processes (DPPs) as the objective function which considers the global negative correlations during inference to decompose the quality and diversity.

Last, compared to the traditional item list recommendation, the search space is the doubly exponential number of items for the bundle recommendation, since a bundle can contain arbitrary-length combination theoretically and a list contains multiple bundles [9]. Bundle sparsity should be carefully considered because the training data in the form of co-purchase contains few bundle ground truths w.r.t. the whole search space. Most of the bundles in search space are low-quality, so we need a method to generate high-quality candidate bundles efficiently for each user, instead of the ranking method applied in all the possible bundles.

In this paper, we aim to the problem of the personalized bundle list recommendation and propose a **Bundle Generation Network (BGN)**. We summarize the contributions of this paper as follows:

- It is the first attempt to address the bundle list recommendation problem by the sequence generation approach considering both the quality and diversity. We decompose the generation model based on determinantal point processes, and integrate the neural network with DPP selection to produce the high-quality and diversified bundle list.
- We propose a feature-aware softmax in bundle recommendation scenario, for making up the inadequate representation of traditional seq2seq for rich features, which improves the modeling of quality significantly. The loss of feature-aware softmax is a generalization of optimizing BPR loss at each step.
- We conduct extensive experiments on both the real-world dataset and public datasets. Our model improves the precision of the best baseline by 16% on average while maintaining the highest diversity on four datasets, and shows the fastest response time in the bundle list recommendation problem. Besides, BGN can control the bundle size with a tolerable reduction in precision.

2 RELATED WORKS

We present the existing bundle recommendation approaches and introduce the sequence generation model (Seq2seq) and Determinantal Point Processes (DPPs) as related works.

2.1 Bundle Recommendation

Product bundling is one of the marking strategies commonly used in both offline retailers and online e-Commerce websites. Frequent itemsets mining algorithms like Apriori [2] provide high-quality bundling but are not personalized. Parameswaran et al. [21] study some complex constraints like cross-item dependencies in personalized curriculum bundling. The complexity of top-K bundle recommendation is discussed comprehensively in [9]. All above only use frequency of items, which cannot utilize rich features and fail to generalize on similar items and users.

For the single bundle recommendation, Zhu et al. [36] propose a personalized method based on integer quadratic programming. This method considers the first and the second terms of cross-item dependencies. However, the estimation of the second term is heuristic, and this method ignores the higher order terms of cross-item dependencies, which gives a sub-optimal result. The bundle recommendation problem to groups rather than users is investigated by [23]. For the bundle list recommendation, Xie et al. [32] generate top-k bundles via sampling and preference elicitation. This method uses an additive utility function with a linear combination of the corresponding (aggregate) feature values, which fails to capture the complicated dependencies. In [22], Pathak and Gupta build an optimization criterion upon the latent factor model, Bayesian Personalized Ranking (BPR) [24], with the mean pair-wise Pearson correlation of the items. This method generates personalized bundles by a two-stage approach, combining the corresponding preference scores with a greedy annealing schedule, which might be caught in the local optimum due to randomness and instability of annealing.

2.2 Sequence Generation Approaches

In recent years, sequence generation methods achieve great successes in many tasks like Machine Translation [26] and Dialog Generation [29]. They adopt an encoder-decoder architecture to generate sequences, and use the beam search strategy [30] to produce the maximum possible sequences approximately.

In recommendation systems, some works based on RNN, CNN, attention extract the user embedding from the behaviors for each user [31, 33, 34]. The recurrent model and the memory network are also adopted for sequential recommendation [7, 18]. The difference between the sequential recommendation and the bundle recommendation is that, the former uses the encoder only, while the latter uses the encoder to model the user context and uses the decoder to model the bundle distribution. The decoder usually has separate trainable parameters without sharing with the encoder. Besides, The bundle recommendation produces multiple bundles, so we need to know when to end the generation and control diversity of bundles.

2.3 Determinantal Point Processes

Determinantal Point Processes (DPPs), first introduced to model fermion behavior, have gained popularity due to their elegant balancing of quality and diversity. DPPs have been studied for their theoretical properties [11, 16], and their machine learning applications including summarization [5, 17], object retrieval [1] and conversation generation [25]. Structured Determinantal Point Processes (SDPPs) [15] extend the DPP model to handle an exponentially-sized set of particles (structures) via a natural factorization combined with the second-order message passing, which leads to tractable algorithms for inference.

Recent works use DPPs for item list recommendation in [10, 35] to solve cross-item diversity, but to the best of our knowledge, there are no studies about cross-bundle diversity. In this paper, we use SDPP as the tool to decompose the quality/diversity term.

3 PERSONALIZED BUNDLE LIST RECOMMENDATION

Notation. Let $\mathcal{I}$ be the whole set of items that one could select from and $|N|$ be the number of all items. A bundle b is any combination out of $\mathcal{I}$ which consists of items, and T denotes the number of items in a bundle. The items are unordered in bundles. When regarding the bundle as a sequence, we usually give the order by sorting the items of the bundle in terms of their prices in a descendant order, because the cheaper items are more substitutable for the bundle and could endure more exposure bias in the sequence generation model. Let $\mathcal{B}$ be the set of all possible bundles, so $|\mathcal{B}|$ equals $\sum_{t=1}^{T}\binom{N}{t}$ with the fixed maximum bundle size T, where $\binom{N}{t} = \frac{N!}{t!(N-t)!}$ is the combination of t out of N.

$$\mathcal{I} = \{1, 2, \ldots, N\} \qquad |\mathcal{I}| = N$$

$$\mathcal{B} = \{b = \{i_1, i_2, \ldots, i_T\} | i \in \mathcal{I}\} \qquad |b| = T,\ |\mathcal{B}| = \sum_{t=1}^{T}\binom{N}{t}$$

$$\mathcal{Y} = \{y = \{b_1, b_2, \ldots, b_K\} | b \in \mathcal{B}\} \quad |y| = K,\ |\mathcal{Y}| = \binom{|\mathcal{B}|}{K}$$

Here K is the size of a recommended bundle list y. Still, we consider the order does not matter in the bundle list. Let $\mathcal{Y}$ be the set of all possible bundle list. The user context C_u is usually represented as the item sequence he/she has clicked/bought in the history.

Problem Formalization. Let $\tilde{\mathcal{P}}(\cdot,\cdot) : \mathcal{Y} \times \{C_u\} \to \mathbb{R}^+$ be a universal compatibility function measuring the score (compatibility) of the bundle list y and the user context C_u, and we refer $\tilde{\mathcal{P}}(y|C_u)$ to a unnormalized conditional probability of y given C_u. Then, the personalized bundle list recommendation problem can be formalized as a structured prediction problem [3], which tries to find the bundle list $\hat{y}$ satisfying:

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} \tilde{\mathcal{P}}(y = \{b_1, b_2, \ldots, b_K\} | C_u) \tag{1}$$

Let $\mathcal{P}(y|C_u)$ be the normalized conditional probability of $\tilde{\mathcal{P}}$ over y. $\mathcal{P}(y|C_u)$ is actually modeling a distribution over the *doubly exponential* number of N, so there are $\binom{\sum_{t=1}^{T}\binom{N}{t}}{K}$ possible structures of y given C_u naively, which poses an extreme computational challenge. In this paper, we consider the compatibility function $\tilde{\mathcal{P}}$ related to both the quality and diversity.

The bundle list recommendation is a generalization of the single bundle recommendation problem and single-item list recommendation problem. When each bundle is independent given the user context, we have $\mathcal{P}(y = \{b_1, b_2, \ldots, b_K\}|C_u)) = \prod_k \mathcal{P}(b_k|C_u)$. In this case, there are no correlations among the bundle list, and the

problem becomes the single bundle recommendation problem [36], sometimes considered as the personalized bundle ranking problem [22]. Furthermore, when the bundle size T equals one it is the single-item list recommendation problem.

4 BUNDLE GENERATION NETWORK

Here we propose a *Bundle Generation Network* (BGN) to solve the personalized bundle list recommendation problem considering both the quality and diversity.

In the following subsections, we introduce how to decompose the bundle list probability into the quality/diversity part by SDPP and factorize them respectively. Then, we illustrate the measurement of diversity and design an improved neural network with the feature-aware softmax to learn the quality part. Besides, we propose a masked beam search to control the bundle size and avoid duplicated items. Finally, we demonstrate how to integrate these parts to BGN to tackle the process of generation for the bundle recommendation.

4.1 SDPP-based Bundle Generation Model

We define a *structured determinantal point process* (SDPP) [15] on the probability space $(\mathcal{B}, \mathcal{Y}, \mathcal{P})$. When Y is a random variable of bundle recommendation list drawn according to the probability mass function (PMF) $\mathcal{P}$, the distribution is given by:

$$\mathcal{P}(\mathrm{Y} = y) \triangleq det(L_y)/Z \tag{2}$$

Here, L is a symmetric positive semi-definite kernel of SDPP, with the size of $|\mathcal{B}| * |\mathcal{B}|$. L_y denotes the restriction of L to the entries indexed by elements of y, and $Z = \sum_{y' \in \mathcal{Y}} det(L_{y'})$ is the normalization constant. There is an intuitive geometric interpretation of the kernel L, that is, $\det(L_y)$ is the square of the volume spanned by its associated feature vectors [16]. Notice that, each element of y refers to a bundle which has a certain structure, so we consider that the DPP is structured.

One of the benefits of defining a DPP probability model is that we can decompose quality and diversity in a very natural way. For any element L_{ab} of positive semi-definite matrix L we could always find a quality/diversity decomposition:

$$\begin{aligned} L_{ab} &= q(a)\phi(a)^T\phi(b)q(b) \\ &= q(a)S(a,b)q(b) \end{aligned} \tag{3}$$

where $a, b \in \mathcal{B}$, L_{ab} denotes the element indexed by bundle a and b, $q(b) \in \mathbb{R}$ denotes the quality term of bundle, and $\phi(b) \in \mathbb{R}^D$ denotes the normalized diversity feature with $||\phi(b)||_2 = 1$. Here we define a similarity function/metric $S(a,b) : \mathcal{B} \times \mathcal{B} \to \mathbb{R}$ which satisfies $S(a,b) = \phi(a)^T\phi(b)$ and keeps the corresponding similarity matrix S positive semi-definite. We reformalize Equation (3) in the matrix way as

$$L = QSQ \tag{4}$$

where Q denotes a diagonal matrix $\mathrm{diag}\{q(b)_{b \in \mathcal{B}}\}$. The decomposition balances the quality and the diversity. Then, the optimization criterion of the personalized bundle list recommendation is to find a bundle list y maximizing the $\mathcal{P}(y|C_u;\theta)$ given the user context C_u:

$$\arg\max_{y} \ \det(S_y) \cdot \prod_{b \in y} q^2(b|C_u;\theta) \tag{5}$$

One major challenge for solving the personalized bundle list recommendation is that, the optimization criterion of the bundle list has the complexity over doubly exponential number of N. According to the decomposition in Equation (5), we reduce the complexity to a quality term q over the measure space $\mathcal{B}$, and a diversity term S over the measure space $\mathcal{B} \times \mathcal{B}$. However, both of the terms still have exponential space which is intractable.

SDPP considers that any elements in measure space $\mathcal{B}$ can factorize over a set of factors F [15], where a factor $\alpha \in F$ is a small subset of the parts of a structure. Then, we continue to factorize the quality term and the diversity term respectively:

$$q(b) = \prod_{\alpha \in F} q_\alpha(b_\alpha) \qquad \phi(b) = \sum_{\alpha \in F} \phi_\alpha(b_\alpha) \tag{6}$$

The factorization defines how we interpret the model structure and keeps the model tractable. Then we shall specify the factorization for the diversity and quality respectively, and integrate them to produce high-quality and diversified bundle list.

4.2 The Measurement and Factorization of Diversity

The diversity we measure in our work is a interpretable metric/goal, trying to quantify the user-experience in bundle recommendation system, which is independent of the likelihood from data. This implicit relationship learned from data may not be the same 'diversity' as we target, and the correlations among items from data have been considered into the quality part. Whereas, the explicit similarities defined by side information and prior knowledge provide exact signals for the purpose of the measurement and adjustment. Besides, the recommendation system is usually lake of the diversity-labeled ground-truth list, which makes it hard to supervise the complete likelihood.

With Equation (3) and Equation (6) and , we factorize the bundle similarity function to the item similarity function:

$$\begin{aligned} S(a,b) &= \phi(a)^T\phi(b) \\ &= (\sum_{\alpha \in F} \phi_\alpha(a_\alpha)^T)(\sum_{\beta \in F} \phi_\beta(b_\beta)) \\ &= \sum_{\alpha,\beta} \phi_\alpha(a_\alpha)^T\phi_\beta(b_\beta) \\ &= \sum_{i \in a, j \in b} s_{a,b}(i,j) \end{aligned} \tag{7}$$

Here, $s_{a,b}(i,j)$ measures the similarity of items rather than bundles, which is independent of user context. The similarity funciton usually should be consistent with the metrics of diversity predefined according to domain knowledge. For example, in our experiments the $s_{a,b}(i,j)$ is given by Jaccard similarity of bundle:

$$s_{a,b}(i,j) = \frac{1}{|a \cup b|}\delta(i = j) \tag{8}$$

where $\delta(\cdot)$ denotes the indicator function. Note that the bundles do not contain duplicated items inside. This function can maintain the positive semi-definition of L [4] and prevent the identical items shown in the different bundles. The function keeps consistent local measurement with Equation (19), and SDPP optimizes the global negative correlations. Besides, the metric of diversity could also be

defined by calculating the similarity of attributes, which provides the fine-grained diversity of attributes.

4.3 Modeling the Quality Part with Feature-Aware Softmax

Here, we focus on modeling the quality part $q(b|C_u;\theta)$ by a typical *Encoder-Decoder* framework, where the encoder extracts the information from the user context and the decoder generates the high-quality candidate bundle list with a proposed feature-aware softmax.

For the encoder, the user context can be represented as the purchase history of items. We use a text-CNN [13] to extract the information, which has recently been shown of effectiveness in text classification and information extraction [33, 34]. We show the basic network architecture in Figure 2 and omit the discussion of the details because it is not one of our main contributions. The text-CNN is parallelizable and faster than bi-LSTM, and it could also be easily improved by Transformer [27] based on the self-attention architecture in the future work.

For the decoder, the quality of a bundle $q(b|C_u;\theta)$ is an unnormalized joint probability parameterized in terms of a generic θ, so we factorize it in the Bayesian formula way. Note that the Bayesian formula always holds and models the complete distribution of quality, the assumption here is sharing the parameters in a sequence way.

$$\begin{aligned} q(b|C_u;\theta) &\propto p(b=\{i_1,i_2,\ldots,i_T\}|C_u;\theta) \\ &= \prod_{t=1}^{T} p(i_t|i_1,...,i_{t-1},C_u;\theta) \end{aligned} \tag{9}$$

In Equation (9), we factorize the joint probabilities of any bundle by multiplying all conditional probabilities for each of its elements. By sharing the same weights of those conditional probabilities, we use a sequence generation model with the (stacked) LSTM cell with the well-known Luong Attention [19] to calculate the probability. Let x_t be the decoder's input and t be the item position in the bundle. We initialize h_0 with the encoder's output. Still, we omit the common details of LSTM and attention.

$$\begin{aligned} h_0 &= \text{text-CNN}(C_u) \\ x_t &= [\text{item_emb}(i_t), \text{cate_emb}(i_t), ...] \\ h_t &= \text{stacked-LSTM-attention}(h_{t-1}, x_{t-1}, C_u) \\ p(i_t|i_1,...,i_{t-1},C_u;\theta) &= \text{feature-aware-softmax}(h_t)_{i_t} \end{aligned} \tag{10}$$

The traditional softmax layer contains a weight matrix $W^{(h+1)*N} = [w_1, w_2, ..., w_N]$, where w_i could be considered as an embedding of item_id of i. Note that, we eliminate the bias term here just for writing convenience.

$$\text{softmax}(h_t, W)_j = \frac{exp(h_t^T w_j)}{\sum_{\hat{j}=1}^{N} exp(h_t^T w_{\hat{j}})} \tag{11}$$

However, the item candidates in recommendation system usually have more belonging features rather than just item_id. The traditional softmax could result in inadequate representation for weight matrix owing to the lack of features.

Inspired by pointer network [28], we propose **feature-aware (FA) softmax**, which modifies the softmax operator with feature-aware weight matrix instead of a stationary weight matrix, and makes the softmax sensitive to dynamic information. Feature-aware softmax uses the dynamic weight matrix built by $F(x_i)$, where F is a nonlinear transformation function for all kinds of features, so that the softmax could be sensitive to features besides item_id.

$$\begin{aligned} E^{(h+1)*N} &= [e_1, e_2, ..., e_N]^1, \qquad e_i = F(x_i) \\ \text{feature-aware-softmax}(h_t)_j &= \text{softmax}(h_t, E)_j \end{aligned} \tag{12}$$

We use the co-purchased data $\mathcal{D} = \{b, C_u\}_{1:|\mathcal{D}|}$ to guide the learning process. The loss could be defined as the mean cross entropy for the softmax layer of each step, which guarantees the same form as Equation (9) and the model is to maximize the likelihood of a single bundle.

$$\mathcal{L}_{\text{MLE}} = -\frac{1}{T}\sum_{t=1}^{T} \log p(i_t|i_1,...,i_{t-1},C_u;\theta) \tag{13}$$

However, the feature-aware softmax might be time-consuming during training owing to constructing the weight matrix E dynamically. So we apply feature-aware softmax with sampled softmax loss [12], denoted as the sampled feature-aware softmax loss, which accelerates the training process.

We show that the sampled feature-aware softmax is a generalization of optimizing the average BPR [24] at each step. The sampled feature-aware softmax samples a part of negative items at each step plus the ground truth of the next step, and then builds the corresponding weight matrix. We regard h_t as the user vector of BPR, e_{i_t} as the positive item vector, and e_{s_t} as the negative item vector, where s_t is the sampled negative item at step t. When setting the sample number to be 1, the loss of the sampled feature-aware softmax is given by:

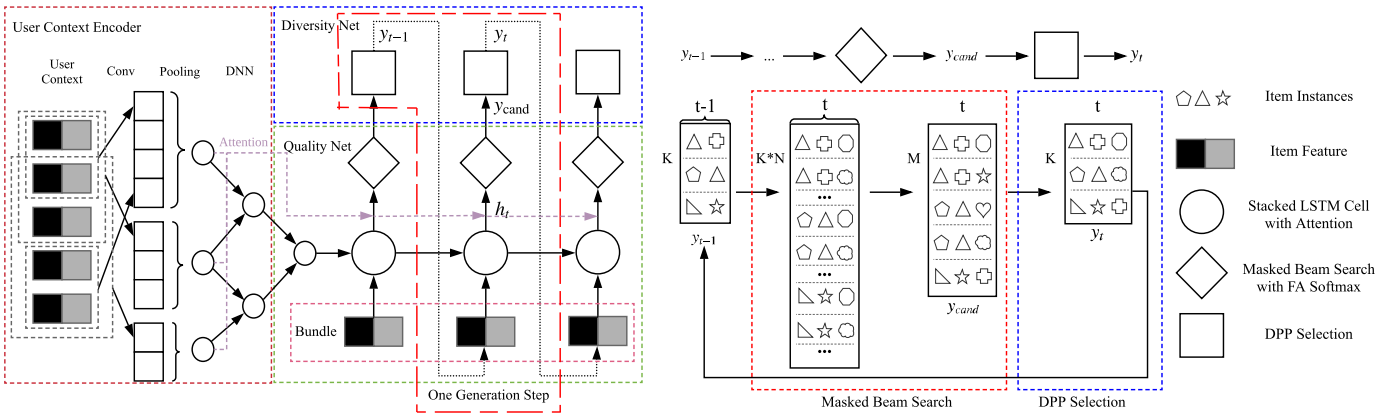
$$\begin{aligned} \mathcal{L}_{\text{sample-1}} &= -\frac{1}{T}\sum_{t=1}^{T} \log \frac{exp(h_t^T e_{i_t})}{exp(h_t^T e_{i_t}) + exp(h_t^T e_{s_t})} + \lambda_\theta ||\theta||^2 \\ &= -\frac{1}{T}\sum_{t=1}^{T} \log \sigma(h_t^T (e_{i_t} - e_{s_t})) + \lambda_\theta ||\theta||^2 \\ &= -\frac{1}{T}\sum_{t=1}^{T} \log p(\theta| \geq_{t,C_u}) = \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{\text{BPR}}(t) \end{aligned} \tag{14}$$

where $\sigma(\cdot)$ is the logistic sigmoid function, and $\geq_{t,C_u}$ is the pairwise preference at each step as [24]. So the feature-aware softmax with one negative item is equivalent to optimizing average BPR at each time in equation (14).

4.4 Overall Process of Bundle Generation

The goal of bundle list recommendation is to generate the high-quality and diversified bundles list y maximizing the $p(y|C_u)$ in Equation (5). We integrate and summarize the overall process of bundle generation in this subsection.

Figure 2(a) shows the overall inference process of BGN. We initialize h_0 with the output of text-CNN and then produce the bundle list by expanding the generated prefix of bundles at each step. Figure 2(b) shows how the bundle list y_t expands at each generation step. Each small shape icon denotes an item, each row



a. Overall Inference Process

b. One Generation Step

Figure 2: Inference Process of Bundle Generation Network.

of the rectangular block is a generated bundle prefix, and t indicates the aligned size of the generated bundle prefix. At each generation step, we use the beam search to produce $K * N$ candidate bundles. Beam search [30] is a heuristic search algorithm that explores the search space by expanding the most promising item in a limited set, which is widely used [26] in sequence generation approaches. Here, we use the beam search to prune the low-quality ones down to y_{cand} with width M, which is crucial for the efficiency. The submodular assumption [16] of DPPs is widely used during inference in practice including recommendation systems [6, 10], where one can find the solution in polynomial time. Following Equation (5), we choose the bundle b from y_{cand} one by one maximizing $P(y \cup \{b\})$, which is given by:

$$\begin{aligned} &\arg\max_{b \in y_{\text{cand}}} \log\det S_{y\cup\{b\}} + \sum_{\beta \in y \cup \{b\}} \log q^2(\beta|C_u) \\ \Leftrightarrow &\arg\max_{b \in y_{\text{cand}}} \log p(b|C_u) + \lambda \log\det S_{y\cup\{b\}} \end{aligned} \tag{15}$$

Here, λ is a hyper-parameter controlling the tradeoff between quality and diversity. When we increase λ, we pay more attention to the diversity among the generated bundles than the precision of the final list, and vice versa.

In addition, we need to avoid duplicated items in the bundle, and would like to generate bundles with the larger size in practice. Usually, We add a particular end token of an item to mark the end of bundle generation so that we can produce the various-length bundle. However, BGN uses the co-purchase data or pre-defined bundles to supervise the quality part. The distribution of groundtruth bundle size may not be consistent with expected distribution for target applications. In practice, one may need larger bundle size due to sellers' promotion. Besides, the traditional beam search often has a significant length bias and squeezes the distribution of bundle size to a much more skewed one that prefers shorter size, as we show in Figure 4(b) in our experiments. If we simply select the dataset with larger bundle size when training, it could cause more sparsity issue owing to the reduction of the training set.

For controlling the distribution of bundle size, as well as eliminating duplicated items, we improve the softmax with masks during beam search, which denotes the masked beam search. The masked beam search subtracts a mask vector $m_t \in \mathbb{R}^N$ to the logits for decreasing the probability in beam search during inference, which is given by:

$$\text{masked_softmax}(h_t, E, m_t)_j = \frac{exp(h_t^T e_j - m_{t,j})}{\sum_{\hat{j}=1}^{N} exp(h_t^T e_{\hat{j}} - m_{t,\hat{j}})} \tag{16}$$

where $m_{t,j}$ is the j-th element of m_t. Notice that, the mask vector reduces the certain unnormalized probabilities exponentially. In the bundle recommendation problem, we set m_t to be:

$$m_{t,j} = \begin{cases} \max(C - t, 0) & \text{j is the end token} \\ +\infty & \text{j has been generated before } t \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

where C is a positive hyper-parameter for shifting the distribution of the bundle size, which gives a threshold for end token. The reduction effect due to the shifting hyper-parameter C decays exponentially with time step and dribbles away after C steps. Besides, the mask vector can avoid the duplicated elements in the bundle. We can set $m_{t,j}$ to be infinity when we definitely don't want to generate the item j at time step t.

The overall algorithm is shown in Algorithm 1, where line 5-15 show the single generation step and line 10-12 show the DPP selection according to Equation (15). Each generation step consists of the stacked LSTM cell, the masked beam search with feature-aware softmax, and the DPP selection. We consider the hidden dimension of LSTM cell is constant here. The time complexity of generating candidate list y_{cand} using beam search is $O(TMN)$, and we usually choose M larger than K to ensure that the candidate set is sufficiently diversified. The time complexity of the DPP selection at each step is $O(MK^\omega)$ where the K^ω term comes from the determinant calculation and ω denotes 2.37. Note that we do not need $O(N^2)$ space to compute and store all item pair's similarity because we only compute S_y rather than entire S. The total time complexity of inference process for BGN is $O(TMN + TMK^\omega)$.

5 EXPERIMENT

We organize the experiment section as follows. First, we clarify the experiment setup, measurement metrics, and all competitors. Second, we illustrate the performance gain of feature-aware softmax

Algorithm 1 Bundle Generation Network during Inference

Input: User context C_u, Similarity function $S(a, b)$, Quality/diversity hyper-parameter λ, Maximum bundle size T, Beam search width M, Output list size K
Output: Diversified bundle list y with size K
$h_0 \leftarrow$ text-CNN(C_u)
$y_0 \leftarrow \varnothing$
Building the weight matrix E of the feature-aware softmax according to Equation (12)
for $t \in [1, ..., T]$ **do**
    $h_t \leftarrow$ Stacked LSTM Cell with Attention(h_{t-1}, y_{t-1}, C_u)
    $y_{\text{cand}} \leftarrow$ Masked Beam Search with FA Softmax(h_t, E, m_t)
    // y_{cand} contains M bundles with maximum size t
    $y_t \leftarrow \varnothing$
    // DPP Selection
    while $|y_t| \neq K$ **do**
        Choose b from y_{cand} according to Equation (15)
        $y_{\text{cand}} \leftarrow y_{\text{cand}} - \{b\}$
        $y_t \leftarrow y_t \cup \{b\}$
    end while
    //y_t has K bundles with maximum bundle size t
end for
return $y \leftarrow y_T$

and the performance impact of the masked beam search. Then, we show the balances the diversity and the precision. Finally, we compare with all competitors to see the advantages of our methods on metrics of precision, diversity, response time for generation and AUC for ranking.

5.1 Experiment Setup

Dataset. We use three public datasets and one industrial dataset: Electro and Clothe are two public datasets of Amazon in [20] where bundles are generated from co-purchase records synthetically. Steam is a public dataset where bundles are extracted from video game distribution network in [22]. Taobao is an industrial dataset where we collect the real sales transaction of bundles from the online bundle list recommendation services. We show the statistics for each dataset in Table 1.

Dataset	Electro	Clothe	Steam	Taobao
Items	63,001	23,033	10,978	104,328
Users	192,403	39,387	88,310	2,767
Categories	801	484	-	3,607
Records	1,689,188	278,677	87,565	54,344
Bundles	327,874	72,784	615	1,236
Average Bundle Size	1.50	2.47	5.73	4.06

Table 1: Statistics of each dataset.

Amazon Dataset[1]. We use two subsets (Electro and Clothe) of Amazon product data in [20], which have already been reduced to satisfy the 5-core property, such that each of the remaining users and items has five reviews. We regard the co-purchase items as a bundle, where items are sorted in terms of their prices in a descendant order. The behavior history of each user can be represented as $(b_1, b_2, ..., b_k, ..., b_K)$ where b_k is a bundle consisting of co-purchase items. We make the first k bundle behaviors of each user as the user context to predict the $k+1$ th bundle behavior in the training set and validation set, where $k = 1, 2, ..., K-2$, and we use the first $K-1$ bundle behaviors to predict the last one in the test set. The average bundle size is 1.50 for Electro and is 2.47 for Clothe, and we notice that many bundles consist of only one item in these two datasets. For alleviating the sparsity of bundle purchase records and keeping enough training records, we consider the bundles with one item is valid in training set. However, in the test set, we only use bundles consisting of more than one items to measure the performance, which isolates the measurement from the single-item list recommendation. The features that we use contain user id, item id, category id, and bundle id. The category is a feature of items.

Steam Dataset[2]. This dataset is collected in [22] from the Steam video game distribution platform. The item is a certain game in this dataset. We partition all purchase records into 70%/10%/20% training/validation/test splits. As in Amazon dataset and Taobao dataset, Some combinations of items in the test set may never be seen in the training set, which leads to the difficulty in generation. Note that there is no feature of the category in the Steam dataset.

Taobao Dataset. This dataset is collected from an online bundle list recommendation service of a commercial e-business, Taobao[3]. Different from the synthetic bundles consisting of co-purchase items in Amazon dataset, the bundles in the Taobao dataset are predefined manually by the sellers. Correspondingly, they have higher quality and longer average bundle size. We conduct the careful collection which guarantees that each user has at least three bundle behaviors. We use the last one of bundle behaviors in the test set and the others in the training set. The user context consists of items which users have purchased. The features that we use contain user id, item id, category id, and bundle id.

Measurements. For the bundle generation problem, the metrics we mainly consider are the top-k precision ***pre@k*** measuring the quality and the cross-bundle diversity ***div***. The *pre@k* is the average precision with the ground truth over each user and each position in the final bundle list. The *div* is defined as 1 − Jaccard Similarity over each pair of the final bundle list.

$$pre@k = \frac{1}{|U|} \sum_{u} \frac{1}{|K|} \sum_{b \in y_u} \frac{|b \cap gt_u|}{|b \cup gt_u|} \tag{18}$$

$$div = \frac{1}{|U|} \sum_{u} \frac{1}{|K|(|K|-1)} \sum_{a,b \in y_u} (1 - \frac{|a \cap b|}{|a \cup b|}) \tag{19}$$

where gt_u is the ground truth bundle that a user buys or clicks. According to the dataset setup above, there is only one bundle in gt_u for the Amazon and Taobao dataset, and there are multiple bundles in gt_u for the Steam dataset. Let y_u be the final recommendation bundle list with size K, U be all users in the test set. These two metrics can be applied to any case no matter whether the bundle is predefined. Note that, the repeated items belonging to different

[1] http://jmcauley.ucsd.edu/data/amazon

[2] http://jmcauley.ucsd.edu/data/steam.tar.gz

[3] https://www.taobao.com

bundles hit the ground truth and increase the *pre@k*, but they decrease *div*.

For studying whether our method can adapt to the bundle ranking problem, we use the **AUC** to evaluate the performance, which is given by:

$$AUC = \frac{1}{|U|} \sum_{u} \frac{1}{|gt_u|} \sum_{b^+ \in gt_u} \delta(\hat{p_{u,b^+}} > \hat{p_{u,b^-}}) \tag{20}$$

where $\hat{p_{u,b}}$ is the predicted probability that a user may act on a bundle and $\delta(\cdot)$ denotes an indicator function. Note that, b^- is sampled uniformly from all valid bundles in the dataset, which are all co-purchase records consisting of more than two items for the Amazon dataset and all pre-defined bundles for the Steam and Taobao dataset.

Competitors. We mainly consider six methods as below:

- **BGN**. This denotes our Bundle Generation Network method. For CNN encoder of user context, we use various filters with the window size of $\{1, 2, 4, 8, 12, 16, 32, 64\}$, and each filter contains 12 output channels, so the final dimension of output channel is 96. The other latent dimension is set to be 64. We use two layers' stacked LSTM cell with attention in the decoder. The batch size is set to be 16. We use l2-loss in the training process, and the weight is set to be 5e-5. Adam is adopted as the optimizer. The beam search width is set to be 50. The sample number of sampled feature-aware softmax is set to be 1024. We use Equation (8) as the similarity function. λ is the quality/diversity hyper-parameter ranging from 0.0 to 5.0 with step 0.25. C is the shifting hyper-parameter controlling the distribution of bundle size, which ranges from 0 to 20 with step 1.0.
- **Freq**. This is a well-known frequent itemsets mining algorithm introduced in [2], and we generate bundles according to frequent itemsets discovered in co-purchase or pre-defined bundles. Then we recommend the most frequent k bundles to users. Note that this is not a personalized method.
- **BRP**. *BRP* (Bundle Recommendation Problem) [36] is a personalized single bundle recommendation algorithm, which optimizes the expected reward based on Mixed-Integer Quadratic Programming (MIQP). We use matrix factorization as the probability model which has the best performance in the original paper, and use Gurobi[4], one of the fastest public mathematical programming solver, to optimize the MIQP problem.
- **BBPR**. *BBPR* (Bundle Bayesian Personalized Ranking) is proposed by [22]. It builds the optimization criterion upon the Bayesian Personalized Ranking(BPR) [24], with the mean pair-wise Pearson correlation of the items. Combined with traditional matrix factorization techniques, it utilizes both item and bundle information including bundle size and item compatibility for better performance. As for the bundle generation, it proposes a greedy algorithm following an annealing schedule to give personalized bundles, which avoids ranking for all possible bundles. In the Steam dataset, we maintain the setup of typer-parameters as the original paper, and in other datasets, we deploy a small range of fine-tuning and report the best results.
- **RankAll**. *RankAll* ranks all the existing bundles in the training set and returns the top-K results, and we consider it as a strong baseline to validate whether the ranking methods are good enough for the bundle generation task regardless of the consumption of time and space. Note that, *RankAll* refers to rank all the existing bundles in the training set, instead of ranking all possible bundles in $\mathcal{B}$, which is too large to enumerate even for modern machines. Specifically, *RankAll* adopts the same loss as *BBPR*, but utilizes two text-CNN networks to extract the user context and the bundle information respectively. The hyper-parameter setting of text-CNN is agree with *BGN*. Ranking all the existing bundles in the training set is time-consuming and usually is unacceptable in practice.

Environment. All experiments are conducted on a machine with one GPU (NVIDIA Tesla P100 16G) and 96 CPUs (Intel Xeon Platinum 2.50GHz). We use Tensorflow [5] as the main toolkit of neural network.

5.2 Feature-Aware Softmax Performance Gain

The feature-aware softmax combines more features besides the item_id in the weight matrix of softmax. We demonstrate the performance gain of all datasets in Table 2, where 'w/o FA' denotes 'without feature-aware' and 'w/ FA' denotes 'with feature-aware' for short.

Dataset		Electro	Clothe	Steam	Taobao
pre@10	BGN w/o FA	0.32%	0.13%	-	1.75%
	BGN w/ FA	0.60%	0.38%	9.65%	2.82%
pre@5	BGN wo/ FA	0.38%	0.12%	-	1.83%
	BGN w/ FA	0.67%	0.43%	14.96%	3.09%

Table 2: Feature-Aware (FA) softmax Performance Gain

In Table 2, we observe that the proposed feature-aware softmax can improve the precision significantly, which yields the improvements on *pre*@10 by 0.87x, 1.92x and 0.61x on Electro, Clothe and Taobao dataset respectively. The gain brought by feature-aware softmax indicates that, compared to the traditional softmax, the feature-aware softmax considers more features in the weight matrix and alleviates the sparsity issue of item_id effectively in bundle generation. The performance gain also verifies the importance of features for recommendation tasks as in [8]. Note that there is no feature of categories in the Steam dataset, so there is no difference with feature-aware softmax.

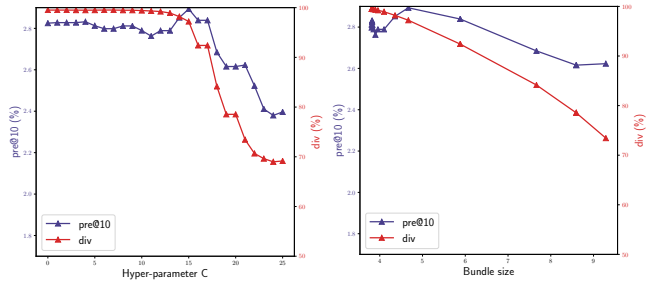
Note that all competitors except *Freq* consider the same complete features as BGN, but BGN with feature-aware softmax performs better because we factorize the bundle probability as a sequence of conditional probability, which is equivalent to optimizing generalized BPR loss directly at each step as shown in Equation (14).

[4] http://www.gurobi.com/

[5] https://tensorflow.org

5.3 Performance Impact due to controlling the Bundle Size

The distribution of the training data may not be consistent with the distribution of the bundle size for the target application. According to the masked beam search, we study the performance impact due to controlling the Bundle Size.



(a) Hyper-parameter C (b) Bundle Size

Figure 3: Performance Impact of the Masked Beam Search (Taobao)

The masked beam search controls the distribution of bundle size by changing a shifting typer-parameter C according to Equation (17), which reduces the unnormalized probabilities exponentially of generating the end token. We show in the Taobao dataset how the distribution of bundle size is altered in Figure 4 and the effect on the precision and diversity in Figure 3.

In Figure 4, we notice that the masked beam search does shift the distribution of bundle size. The distribution shifts significantly compared to the original beam search when setting C to be 14, and with the increase of C the average bundle size shifts as we expected. However, the shifting parameter C enforces the network to generate the most likely items, although the generation should have ended. As shown in Figure 3, C is set from 0 to 20 with step 1.0, which leads to the bundle size varying from 3.8 to 9.2, then we show the performance impact about the precision and diversity respectively for the C and bundle size. We notice that the precision still holds when we shift the bundle size from 3.8 to 6.0, but both the precision and diversity decrease if we continue shifting. Though our shifting strategy is simple, it is good enough to control the distribution in a small range in practice and it might be promoted in the future work through considering the long-term reward in advance by reinforcement learning.

5.4 Balance Between Precision and Diversity

According to Algorithm 1, we produce the diversified bundle list by combining the beam search and DPPs, which chooses the item maximizing the objective function of the quality and diversity in Equation (15). The hyper-parameter λ balances between the precision and the diversity. The metric of diversity is a separate metric from CTR and precision, trying to quantify the user-experience in a bundle recommendation system. When we increase λ, we pay more attention to the diversity among the generated bundles than the precision of the final list, and vice versa. We select K bundles from M candidates generated from the beam search with $M = 50$, $K = 10$ at each step, and λ is set from 0 to 5 with step 0.25 and C is set to be 0.

We show in Figure 5 that in all datasets, the precision decreases along with the growth of diversity as we expected. We also mark the other competitors' results as points in the figure, and we omit the competitors whose precision is too poor. Feature-aware softmax and beam search guarantee that BGN has a high precision base, meanwhile, DPP selection ensures to improve the diversity steadily with a tolerable reduction in precision. With adjustment of the λ, the curve formed by BGN is completely at the top right of all competitors. We achieve the highest precision and diversity in contrast with all competitors in all datasets.

5.5 Comparison in the Bundle Generation Problem

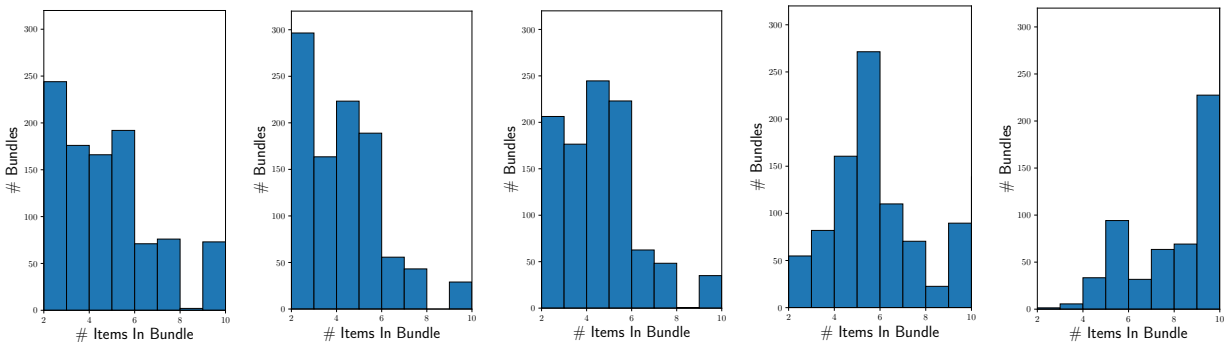
By comparing the proposed BGN against the other competitors including, *Freq*, *BRP*, *BBPR*, *RankAll*, *Seq2seq*, we report and analyze their performance in terms of the precision metric *pre@k*, the diversity metric *div*, and the average response time per user of these methods during inference, denoted by *avg-time* in Table 3. Note that the shifting hyper-parameter C of BGN is set to be 0 in this part becasue we have shown the performance results of shifting the bundle size in Figure 3.

Freq performs poorly in term of precision on all datasets, because it is a non-personalized method, but *Freq* performs quite diverse in Amazon dataset. We think that it is because the bundles of two Amazon datasets are generated synthetically by co-purchase records, which contain more diversified bundles naturally, whereas, the bundles of Steam and Taobao dataset is pre-defined by sellers.

BRP produces only one bundle, so there is no diversity reported. We repeat the results K times to produce the list for achieving the higher precision, which is well-defined because our metric of precision averages on each position. With regard to precision, *BRP* performs better than *Freq* owing to the personalization and consideration of the the first and second cross-item dependencies, but it ignores the higher order terms and gives a sub-optimal result. Besides, compared with other methods, solving the mixed-integer quadratic programming (MIQP) problem is time-consuming.

BBPR produces a personalized list of bundles instead of one. This method utilizes both item and bundle information including bundle size and the pair-wise Pearson correlation of the items, which performs better than *Freq* and *BRP* on most of the dataset. However, the annealing schedule leads to randomness in generating bundles and is easily caught in the local optimum due to instability of annealing.

RankAll performs better than *BBPR* and *BRP* on the metrics of precision and diversity on all datasets, because it uses CNN as a context extractor capturing more combinatorial generalization of features. But for the response time, *RankAll* is the most time-consuming approach on almost all datasets. Because the inference time of *RankAll* is approximately proportional to the number of bundles, whereas, other methods are approximately proportional to the number of items. Essentially, *RankAll* is not a generative model, but a ranking model which ranks the existing bundles to give the final recommendation list. It wastes most of the time scoring the low-quality bundles and it is not scalable in practice. Furthermore, it



(a) Groundtruth (b) Original Beam Search (c) C=14 (d) C=17 (e) C=20

Figure 4: Bundle Size Distribution Shifting by the Masked Beam Search (Taobao)

Dataset	Synthetic Bundle						Real bundle					
	Electro			Clothe			Steam			Taobao		
	pre@10	*div*	*avg-time*	*pre@10*	*div*	*avg-time*	*pre@10*	*div*	*avg-time*	*pre@10*	*div*	*avg-time*
Freq	0.260%	95.56%	-	0.096%	94.07%	-	5.831%	69.44%	-	0.192%	60.00%	-
BRP	0.324%	-	4.85s	0.149%	-	3.96s	6.750%	-	0.44s	0.332%	-	5.47s
BBPR	0.308%	95.87%	0.68s	0.122%	85.56%	0.34s	7.168%	87.32%	0.29s	0.357%	95.49%	1.27s
RankAll	0.443%	88.87%	55.97s	0.280%	89.23%	7.94s	9.497%	97.63%	0.11s	1.540%	98.19%	1.38s
BGN ($\lambda = 0$)	**0.607%**	87.22%	**0.05s**	**0.387%**	81.63%	**0.03s**	**9.655%**	98.01%	**0.04s**	**2.825%**	99.50%	**0.24s**
BGN ($\lambda = 5$)	0.469%	**96.68%**	0.09s	0.235%	**95.75%**	0.06s	9.617%	**99.08%**	0.06s	2.739%	**99.88%**	0.28s

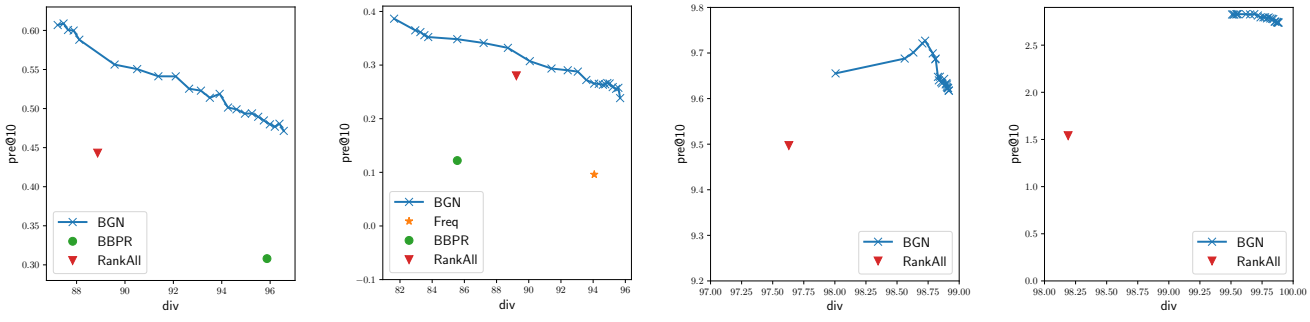
Table 3: *pre*@10, *div*, *avg-time* of Different Methods in the Bundle List Recommendation Problem.

could not produce new bundles beyond the existing ones in training data, which limits the precision.

BGN achieves the best precision, diversity and fastest response time at the same time on all datasets. The high precision benefits from the the feature-aware softmax. However, when setting λ to be 0, BGN has low diversity especially in the synthetic bundle because traditional seq2seq is easy to generate similar sentences. When we simply set the λ to be 5, BGN balances the quality and diversity, which improves the precision of the best baselines by 16% on average while maintaining the highest diversity on four datasets. Besides, BGN is a sequnce generation approach without ranking or recursive searching. The time complexity of BGN during inference is proportional to the number of items. So our model achieves fastest response time in bundle list recommendation problem. Specifically, BGN improves the response time by 6.5x, 4.6x, 0.8x, 3.5x on four datasets respectively over the best competitors when λ equals 5.

5.6 Comparison in the Bundle Ranking Problem

BGN is designed for the bundle list generation problem. However, we can adopt it in the bundle ranking problem, by feeding the candidate bundles to the inputs of the decoder and taking the negative loss as the ranking score, so that we compare with other competitors in the bundle ranking problem. Note that, the absolute values of the AUC in the bundle ranking problem are usually lower than the one in the item ranking problem because the sample space for bundles is larger and more sparse.



(a) Electro (b) Clothe (c) Steam (d) Taobao

Figure 5: Balance between Precision and Diversity.

AUC	Electro	Clothe	Steam	Taobao
BBPR	74.85%	66.20%	90.27%	60.24%
RankAll	75.51%	66.97%	**99.66%**	60.93%
BGN	**78.11%**	**69.99%**	99.36%	**61.27%**

Table 4: AUC Tests in the Bundle Ranking Problem.

In Table 4, we show the measurement of AUC for different methods, and BGN improves the best baseline by 2.04% on average on four datasets. Actually, in the bundle ranking setup, all competitors in Table 4 can be regarded as the encoder-decoder framework, where the encoder extracts the user context information and the decoder extracts the bundle information. The major differences are the architectures of specific neural network and the loss function. With the help of the feature-aware softmax, BGN is good at utilizing the feature information, because the weight matrix of softmax is built dynamically and its loss is a generalization of optimizing the average BPR at each step. The Steam dataset has no category feature, so BGN is worse than *RankAll* slightly on this dataset.

6 CONCLUSIONS

This paper studies a problem of personalized bundle list recommendation and proposes a bundle generation network (BGN). BGN decomposes the problem into the quality/diversity part to produce the high-quality and diversified bundle list. The proposed feature-aware softmax makes up the inadequate representation of traditional sequence generation for rich features in bundle recommendation scenario, which improves the modeling of quality significantly. We conduct extensive experiments on three public datasets and one industrial dataset. BGN achieves the best state-of-the-art results on the metrics of precision, diversity and response time on all datasets. Besides, BGN can control the bundle size with a tolerable reduction in precision.

ACKNOWLEDGMENTS

This work was partially supported by National Key Research and Development Program No. 2016YFB1000700, NSFC under Grant No. 61572040 and 61832001, and Alibaba-PKU joint Program.

REFERENCES

[1] Raja Hafiz Affandi, Emily Fox, Ryan Adams, and Ben Taskar. 2014. Learning the parameters of determinantal point process kernels. In *International Conference on Machine Learning*. 1224–1232.

[2] Rakesh Agrawal, Ramakrishnan Srikant, et al. 1994. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, Vol. 1215. 487–499.

[3] David Belanger and Andrew Mccallum. 2015. Structured prediction energy networks. (2015), 983–992.

[4] Mathieu Bouchard, Anne-Laure Jousselme, and Pierre-Emmanuel Doré. 2013. A proof for the positive definiteness of the Jaccard index matrix. *International Journal of Approximate Reasoning* 54, 5 (2013), 615–626.

[5] Wei-Lun Chao, Boqing Gong, Kristen Grauman, and Fei Sha. 2015. Large-Margin Determinantal Point Processes.. In *UAI*. 191–200.

[6] Laming Chen, Guoxin Zhang, and Hanning Zhou. 2018. Improving the Diversity of Top-N Recommendation via Determinantal Point Process. *Advances in Neural Information Processing Systems* (2018).

[7] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 108–116.

[8] Heng-Tze Cheng, Levent Koc, and Jeremiah Harmsen. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems (DLRS 2016)*. 7–10. https://doi.org/10.1145/2988450.2988454

[9] Ting Deng, Wenfei Fan, and Floris Geerts. 2013. On the complexity of package recommendation problems. *SIAM J. Comput.* 42, 5 (2013), 1940–1986.

[10] Mike Gartrell, Ulrich Paquet, and Noam Koenigstein. 2016. Bayesian low-rank determinantal point processes. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 349–356.

[11] Jennifer A Gillenwater, Alex Kulesza, Emily Fox, and Ben Taskar. 2014. Expectation-maximization for learning determinantal point processes. In *Advances in Neural Information Processing Systems*. 3149–3157.

[12] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2014. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007* (2014).

[13] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[14] Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872* (2017).

[15] Alex Kulesza and Ben Taskar. 2010. Structured determinantal point processes. In *Advances in neural information processing systems*. 1171–1179.

[16] Alex Kulesza, Ben Taskar, et al. 2012. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning* 5, 2–3 (2012), 123–286.

[17] Hui Lin and Jeff A Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. *arXiv preprint arXiv:1210.4871* (2012).

[18] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. 2016. Context-aware sequential recommendation. *arXiv preprint arXiv:1609.05787* (2016).

[19] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[20] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 43–52.

[21] Aditya Parameswaran, Petros Venetis, and Hector Garcia-Molina. 2011. Recommendation systems with complex constraints:A course recommendation perspective. *Acm Transactions on Information Systems* 29, 4 (2011), 1–33.

[22] Apurva Pathak, Kshitiz Gupta, and Julian Mcauley. 2017. Generating and Personalizing Bundle Recommendations on Steam. In *The International ACM SIGIR Conference*. 1073–1076.

[23] Shuyao Qi, Nikos Mamoulis, Evaggelia Pitoura, and Panayiotis Tsaparas. 2016. Recommending packages to groups. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 449–458.

[24] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[25] Yiping Song, Rui Yan, Yansong Feng, Yaoyuan Zhang, Dongyan Zhao, and Ming Zhang. 2018. Towards a Neural Conversation Model With Diversity Net Using Determinantal Point Processes.. In *AAAI*.

[26] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[28] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*. 2692–2700.

[29] Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869* (2015).

[30] Sam Wiseman and Alexander M Rush. 2016. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960* (2016).

[31] Sai Wu, Weichao Ren, Chengchao Yu, Gang Chen, Dongxiang Zhang, and Jingbo Zhu. 2016. Personal recommendation using deep recurrent neural networks in NetEase. In *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*. IEEE, 1218–1229.

[32] Min Xie, Laks VS Lakshmanan, and Peter T Wood. 2014. Generating top-k packages via preference elicitation. *Proceedings of the VLDB Endowment* 7, 14 (2014), 1941–1952.

[33] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 425–434.

[34] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2017. ATRank: An Attention-Based User Behavior Modeling Framework for Recommendation. *CoRR* abs/1711.06632 (2017). arXiv:1711.06632 http://arxiv.org/abs/1711.06632

[35] Tao Zhou, Zoltán Kuscsik, Jian-Guo Liu, Matúš Medo, Joseph Rushton Wakeling, and Yi-Cheng Zhang. 2010. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences* 107, 10 (2010), 4511–4515.

[36] Tao Zhu, Patrick Harrington, Junjun Li, and Lei Tang. 2014. Bundle recommendation in ecommerce. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 657–666.

Improving Medical Code Prediction from Clinical Text via Incorporating Online Knowledge Sources

Tian Bai
Temple University
Philadelphia, PA, USA
tue98264@temple.edu

Slobodan Vucetic
Temple University
Philadelphia, PA, USA
vucetic@temple.edu

ABSTRACT

Clinical notes contain detailed information about health status of patients for each of their encounters with a health system. Developing effective models to automatically assign medical codes to clinical notes has been a long-standing active research area. Despite a great recent progress in medical informatics fueled by deep learning, it is still a challenge to find the specific piece of evidence in a clinical note which justifies a particular medical code out of all possible codes. Considering the large amount of online disease knowledge sources, which contain detailed information about signs and symptoms of different diseases, their risk factors, and epidemiology, there is an opportunity to exploit such sources. In this paper we consider Wikipedia as an external knowledge source and propose **K**nowledge **S**ource **I**ntegration (KSI), a novel end-to-end code assignment framework, which can integrate external knowledge during training of any baseline deep learning model. The main idea of KSI is to calculate matching scores between a clinical note and disease related Wikipedia documents, and combine the scores with output of the baseline model. To evaluate KSI, we experimented with automatic assignment of ICD-9 diagnosis codes to the emergency department clinical notes from MIMIC-III data set, aided by Wikipedia documents corresponding to the ICD-9 codes. We evaluated several baseline models, ranging from logistic regression to recently proposed deep learning models known to achieve the state-of-the-art accuracy on clinical notes. The results show that KSI consistently improves the baseline models and that it is particularly successful in assignment of rare codes. In addition, by analyzing weights of KSI models, we can gain understanding about which words in Wikipedia documents provide useful information for predictions.

KEYWORDS

Multi-label classification; document similarity learning; attention mechanism; healthcare

ACM Reference Format:
Tian Bai and Slobodan Vucetic. 2019. Improving Medical Code Prediction from Clinical Text via Incorporating Online Knowledge Sources. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313485

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313485

1 INTRODUCTION

Clinical notes are free-text documents generated by healthcare professionals documenting clinical status of patients during hospital stay or outpatient care. Clinical notes contain various nuanced information such as a history of illness, lifestyle, symptoms, treatment, and medications. Clinical notes are used to facilitate communication between medical professionals treating the same patient over time, to provide a basis for billing, and to create a record trail in case of legal issues. Medical coding, which refers to translating a clinical note into a set of medical codes from a medical ontology, is necessary to obtain reimbursement for the provided healthcare services. The most popular medical coding ontology is the International Classification of Diseases (ICD, with the most recent versions ICD-9 and ICD-10), whose codes provide alphanumeric encoding of diagnoses and treatments as shown in Table 1. Interestingly, beyond their primary billing purpose, medical codes have been widely used in healthcare research such as predictive modeling [1, 3, 4, 8, 33, 34] and retrospective epidemiologic studies [24, 36, 40]. The reason for popularity of medical codes in research is that it abstracts away fine details of care from the free-text clinical notes and normalizes a patient encounter to a standardized computer-readable format.

Medical coding is typically performed by professional staff trained to understand clinical notes containing complex medical terminology with frequent misspellings and abbreviations and transcribe the notes into a limited set of appropriate medical codes from a large menu of options (e.g., ICD-10 contains more than 100,000 codes). Thus, medical coding is an expensive, time consuming, and inexact process. Due to these issues, there has been a significant interest in developing automatic code assignment models and it has been a long-standing research challenge [6, 10, 11, 13, 22, 25, 29–31, 35, 37, 39].

Recently, deep neural networks with attention mechanism have been shown to achieve the state-of-the-art performance on automatic code assignment task [6, 29, 35, 39]. The attention mechanism [2, 26] uses one or more layers of neurons to identify words indicative of a specific disease and assigns them large weights. The success of the attention mechanism could be attributed to the property of medical code assignment: the relevant information for a specific medical code is often located in a small portion of a clinical note.

Despite the success of deep neural networks with the attention mechanism, there are still outstanding challenges that remain to be addressed. In this paper we address an issue related to recognition of rare codes. To illustrate the issue, let us consider MIMIC-III [18], the largest publicly-available database containing 59,652 discharge notes from the Beth Israel Deaconess Medical Center's Intensive Units. There are 942 unique 3-digit ICD-9 codes occurring in the dataset in a highly unbalanced way, where the most common 10

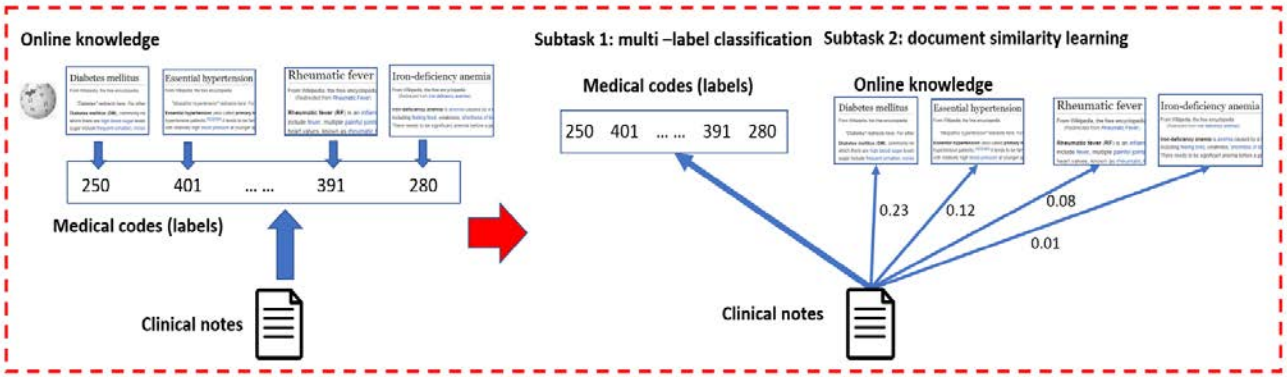


Figure 1: Our main goal is to predict medical codes from clinical notes by taking advantage of online knowledge sources. We decompose the task into two parallel subtasks: the first subtask is traditional multi-label classification and the second subtask is document similarity learning.

Table 1: Examples of 3-digit ICD-9 codes and their descriptions (249-259).

ICD-9 code	Description
249	Secondary diabetes mellitus
250	Diabetes mellitus
251	Other disorders of pancreatic internal secretion
252	Disorders of parathyroid gland
253	Disorders of the pituitary gland and its hypothalamic control
254	Diseases of thymus gland
255	Disorders of adrenal glands
256	Ovarian dysfunction
257	Testicular dysfunction
258	Polyglandular dysfunction and related disorders
259	Other endocrine disorders

Table 2: Wikipedia webpage for ICD-9 code "250" (partially shown).

ICD-9 code: 250 (Diabetes mellitus)
Diabetes mellitus Diabetes mellitus (DM), commonly referred to as diabetes, is a group of metabolic disorders in which there are high blood sugar levels over a prolonged period. Symptoms of high blood sugar include frequent urination, increased thirst, and increased hunger. If left untreated, diabetes can cause many complications. Acute complications can include diabetic ketoacidosis, hyperosmolar hyperglycemic state, or death. Serious long-term complications include cardiovascular disease, stroke, chronic kidney disease, foot ulcers, and damage to the eyes.

codes account for 26% of all code occurrences and the least common 437 codes account for only 1% of the occurrences. Thus, there is highly limited information available for training a neural network to recognize such rare codes. The rare codes problem is particularly severe considering the length of clinical notes: each MIMIC-III discharge note contains an average of 1,596 words. Even with the attention mechanisms, it would be very difficult to learn to identify relevant portions of text from only a few examples.

In this paper we consider exploiting online knowledge sources to improve accuracy of medical code prediction from clinical notes. There are rich disease-related online knowledge sources such as Mayo Clinic, Wikipedia, and Medscape, which contain detailed information such as signs, symptoms, and epidemiology useful for differentiating different diseases in the clinical notes. For example, as shown in Table 2, from a Wikipedia article introducing diabetes, we learn that the typical signs of diabetes include frequent urination, increased thirst, and increased hunger. If a clinical note also mentions in a passage that a patient is having these symptoms, then the passage could be used as an evidence for diabetes. In this paper we use Wikipedia, because it is an easily accessible online resource with loose license constraints that has already been widely studied in the research community. In order to take advantage of the existing state of the art neural networks for medical code prediction, instead of building a completely new model, we develop a framework called **K**nowledge **S**ource **I**ntegration (KSI), which is able to enhance the state of the art neural networks with information from the external knowledge sources. As illustrated in Figure 1, the proposed architecture consists of two components. The first component is directly predicting codes from a note, and it could be any of the state of the art neural networks. Our main contribution is the second component, which is a neural network whose objective is to learn similarity scores between a note and Wikipedia articles about various diseases. Each similarity score measures how closely a clinical note is associated with a corresponding medical code. Finally, the proposed architecture combines the similarity scores with the output of the first subtask and learns to optimize both components to minimize the loss between the predictions and the true labels.

Using Wikipedia as an external knowledge source for analyzing clinical notes poses a couple of challenges. First, there are significant differences between content of Wikipedia articles and clinical notes. Wikipedia articles are mostly carefully edited and the terminology used is optimized toward the general audience. Since Wikipedia can be edited by anyone with Internet access, there is a certain risk that the provided information is incorrect. On the contrary, clinical

notes are written by medical professionals in a hurry and contain a lot of non-standard abbreviations, misspellings and specialized medical terminology. Second, the coverage of Wikipedia and clinical notes are very different. As a knowledge base, Wikipedia elaborates different aspects of diseases including pathophysiology, prevention, history, and social impact, which are unlikely to appear in clinical notes. On the other hand, clinical notes contain patient conditions, including medications on admission and allergies, which are unlikely to occur in Wikipedia articles. In summary, these two types of documents have very different data distributions, which makes unsupervised document similarity measures such as Word Mover's Distance [21] unsuitable, as we need to carefully deal with noise and misleading information in the documents and focus on aspects important for medical code prediction.

In order to properly handle the above challenges, in this paper we develop a supervised document similarity learning model, which is designed to measure the similarity of two types of documents with very different data distributions. To accomplish this, we first take intersection of a clinical note and a Wikipedia document. The intersection operation discards disparate parts of the text and focuses on the common aspects of the two documents. Then, we project the intersection into a low dimensional vector space to reduce dimensionality. Since terms remaining in the intersection might have different importance for medical code prediction, next we apply an attention mechanism to weight the terms. Finally, the weighted intersection vector is transformed into a matching score, which measures the similarity between the two documents. All matching scores are combined with the output of the baseline neural network model and trained to predict medical codes.

In our experiments we selected a variety of baseline models, from simple logistic regression to a recently proposed attention based deep neural network, which achieves the state-of-the-art performance. We show that our KSI framework of incorporating external knowledge sources consistently improves the baseline models. Moreover, thorough analysis shows that KSI significantly improves accuracy of rare code prediction. Finally, by analyzing attention values and weights of the model, our framework reveals which words in Wikipedia documents are the most influential to the prediction, which is important for understanding the reasoning behind the prediction.

Our work makes the following contributions:

- We propose a novel end-to-end framework to integrate external knowledge sources with any end-to-end baseline neural network model for the task of medical code prediction from clinical notes.
- We empirically show that our framework improves a variety of baseline models, including the state of the art attention based neural networks, and that the improvement is particularly large on rare codes.
- We demonstrate that our framework allows interpretability via analyzing attention values and weights of the model, which sheds light on the connection between the external knowledge source and clinical text.

We organize the rest of the paper as follows: in Section 2 we discuss the related previous work. Section 3 presents our KSI framework. In Section 4 we explain the experiment design. The results are shown and discussed in Section 5.

2 RELATED WORK

2.1 Prediction of Medical Codes from Clinical Notes

Automatic assignment of medical codes is a long-standing research area dating back to 1990s [11]. [11] represents clinical notes and ICD-9 codes as two TF-IDF vectors and calculates cosine similarity in the joint vector space. Later, a variety of methods were applied to this area, ranging from rule-based methods [10, 13] to machine learning methods such as K-nearest-neighbor, relevance feedback, Bayesian independence classifiers [22] and support vector machines [25]. In [30, 31] the authors develop methods which exploit the hierarchical tree structure of ICD-9 ontology. More recent models combine neural networks with attention mechanism. In [6], the authors apply hierarchical attention networks [41], which calculate both word-level attention and sentence-level attention while predicting medical codes. In [35], the authors apply character-aware neural network to generate hidden representations of ICD-9 codes and diagnosis section in clinical notes and use the attention mechanism to match these two hidden representations. Their ICD-9 descriptions are quite short (typically, less than 10 words), while in our case a Wikipedia document contains thousands of words providing multiple views of a disease. In [37] the authors propose a modified GRU, where each dimension in the hidden vector corresponds to a specific label, and use their model on medical code prediction. In [29], the authors combine the attention mechanism with a convolutional neural network [19] and report state-of-the-art accuracy on a medical prediction task. In [39], the authors project both words and labels into the same low-dimensional vector space and calculate the compatibility/attention between them. Then, they combine the attention values with word embeddings to get the final document representation. None of the above approaches incorporate external knowledge sources on the task of medical code prediction.

In [32], the authors use memory network to infer clinical diagnosis from clinical notes. While they also use Wikipedia as knowledge source, their work is different from ours in several ways: first, their focus is on a different task, which uses a much smaller diagnosis label set directly derived from clinical notes, while we use accurate medical codes as labels. Second, their memory network is highly specialized and hard to generalize, while our framework can incorporate knowledge into any end-to-end neural network model developed in this field. Lastly, their model is hard to interpret, while our attention based model sheds light on how knowledge source and notes are connected.

2.2 Attention Mechanism

The attention mechanism was first proposed in machine translation [2, 26] to select the most relevant words in a source sentence while predicting the next word in the target sentence. Typically, a neural network is applied to the context vector, generated by either Recurrent Neural Network (RNN) or Convolutional Neural Network (CNN), and attention values are expected to capture the importance

of the context vectors. Final sentence or document representations are simply the sum of context vectors weighted by the attention values. The attention mechanism is widely used in natural language processing [2, 6, 29, 35, 38, 39, 41] and sequence modeling in healthcare domain [5, 9, 27, 33]. Our variable-level attention mechanism is the most similar to [9] because we compute attention on different vector elements. Our work is different in a way that we focus on learning attention of clinical words, instead of medical codes.

2.3 Document Similarity Learning

Document similarity computation transforms two documents into a score depicting how similar the two documents are. Documents are typically transformed into vectors, with each dimension corresponding to the statistic of a word (count, occurrence or TF-IDF [16]) in the vocabulary. Then, similarity between two documents is calculated as the cosine similarity between the two vectors. Other methods transform documents into low-dimensional vectors which capture semantics of documents, such as singular vector decomposition [12] or latent dirichlet allocation [7]. More recently, doc2vec [23] uses architecture similar to word2vec [28] in order to learn document emebddings. Word Mover's Distance [21] exploits high quality word embeddings and treats the document distance as an instance of Earth Mover's Distance. The above approaches are unsupervised. Supervised approaches such as supervised Word Mover's Distance [17] and neighborhood components analysis [14] consider label information while learning similarity between two vectors. However, there is no explicit way to handle the situation where each instance has multiple labels. Moreover, as mentioned before, Wikipedia and clinical notes have very different distributions, thus making direct measurement of the similarity unsuitable. On the contrary, our approach first calculates the overlap between two documents and then uses the attention mechanism to select the most informative parts for medical code prediction.

3 METHODOLOGY

In this section we describe the proposed KSI framework. We first introduce the problem setup and our goal. Then, we describe the details of the proposed approach. Finally, we show how to interpret the learned model via analyzing attention values and weights of neural network associated with words in clinical notes and Wikipedia documents.

3.1 Basic Problem Setup

Assume our dataset is a collection of clinical notes. Each note N consists of a sequence of words. N is accompanied with a set of medical codes M corresponding to the note. We denote the size of word vocabulary $W = \left\{w_1, w_2, ..., w_{|W|}\right\}$ as $|W|$ and the size of medical code vocabulary $C = \left\{c_1, c_2, ..., c_{|C|}\right\}$ as $|C|$. In addition, we have an external knowledge source K which contains a set of documents $K = \left\{k_1, k_2, ..., k_{|C|}\right\}$ in which each document k_i is a sequence of words. Each unique medical code c_i in the medical code vocabulary C has a corresponding document k_i in K. Given a clinical note N, our goal is to predict associated medical codes M when knowledge source K is available. This is naturally a multi-label text classification problem.

3.2 KSI Framework

As shown in Figure 1, when each label (*i.e.*, medical code) c_i has a document k_i in knowledge source K, we can calculate the likelihood of label c_i as the similarity between document k_i and note N. Therefore, we decompose the prediction task into two parallel subtasks. As shown in Figure 2, the first subtask is to directly predict codes from notes, which can be achieved by any end-to-end multi-label classification model in this field. We denote the baseline model as f, which is a mapping from notes N to the label space. The last layer of f is a vector $o_1 \in \mathbb{R}^{|C|}$

$$o_1 = f(N). \tag{1}$$

Since the task is multi-label classification, a sigmoid activation function is applied to o_1 and the i-th dimension in the resulting vector represents the probability that code c_i is the true label. Loss values are calculated between the output and the true labels. In KSI framework o_1 is combined with the output of subtask 2 to get the final output. We select a variety of baseline models and describe the details in the next section. Our main focus is the second subtask, *i.e.*, to learn similarity scores between Wikipedia documents and a note.

First, we represent note N as binary vector $q \in \{0,1\}^{|W|}$ and Wikipedia document k_i as binary vector $x_i \in \{0,1\}^{|W|}$. Binary vectors encode the presence of words. If word occurs in the document, then the corresponding element has value 1, otherwise it is 0. By encoding documents as binary vectors we lose the frequency information. However, as mentioned before, the frequency does not necessarily reflect the importance of words for code prediction since these two types of documents have very different data distributions, *e.g.*, in the Wikipedia document about "influenza", word "birds" has high frequency because there is one section in the document talking about "bird flu". As we formulate the similarity learning in a supervised framework, we expect the network to be able to assign large weights to important words for the task.

After we represent both N and k_i as binary vectors, we calculate their element-wise multiplication as

$$z_i = x_i \odot q, \tag{2}$$

where $z_i \in \{0,1\}^{|W|}$ represents the intersection of two documents. Value of words that appear in both documents equals 1, otherwise it is 0. By taking the intersection operation, we discard irrelevant parts in two documents and focus on the common content between them.

Then, we use an embedding matrix $W_e \in \mathbb{R}^{m \times |W|}$ to transform the high-dimensional sparse vector z_i into a low-dimensional vector space

$$e_i = W_e z_i, \tag{3}$$

where $e_i \in \mathbb{R}^m$ denotes the embedding of z_i and m is the dimensionality of the embedding space. e_i is expected to capture the semantic information about the intersection set. In e_i, each element has its own latent meaning and represents a hidden topic. Since not all topics are relevant to the prediction task, we use a variable-level attention mechanism to weight variables in the vector. We use weight matrix $W_a \in \mathbb{R}^{m \times m}$ to calculate the attention values as

$$\alpha_i = sigmoid(W_a e_i) \tag{4}$$

$$v_i = \alpha_i \odot e_i, \tag{5}$$

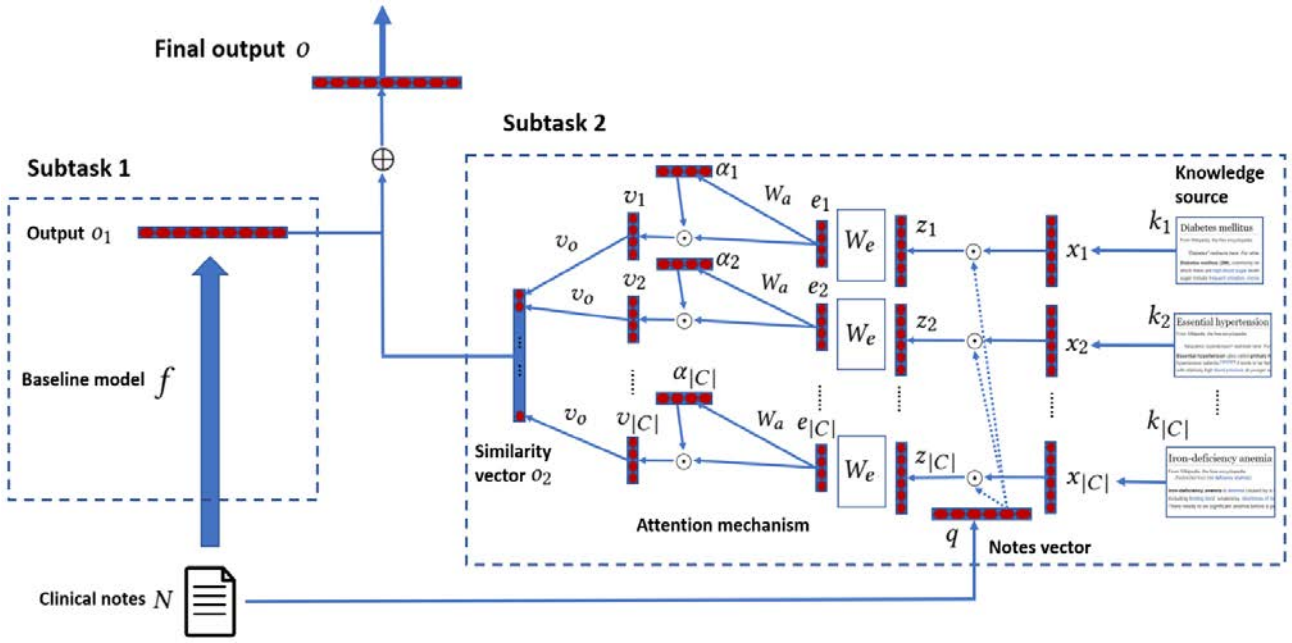


Figure 2: The illustration of our KSI framework. First, it represents document k_i and clinical note N as binary vectors x_i and q. Then, it performs element-wise multiplication of x_i and q to obtain the intersection vector z_i. Next, it projects z_i into a low-dimensional vector e_i. Next, the attention mechanism is used to transform e_i into a weighted vector v_i. Finally, it projects v_i into a similarity score and the concatenation of all scores is combined with the output of baseline model to get the final output.

where $\alpha_i \in \mathbb{R}^m$ is the attention weight vector and $sigmoid()$ is the sigmoid activation function. $v_i \in \mathbb{R}^m$ is the weighted embedding which is the element-wise multiplication of α_i and e_i.

Next, we use a vector $v_o \in \mathbb{R}^m$ to transform v_i into a similarity score s_i as

$$s_i = v_o^T v_i + b_o, \tag{6}$$

where s_i depicts how similar Wikipedia document k_i and note N are and b_o is a scalar bias term. We concatenate the similarity scores for all $k_i \in K$ to get the similarity vector $o_2 \in \mathbb{R}^{|C|}$,

$$o_2 = [s_1, s_2, ..., s_{|C|}]. \tag{7}$$

Finally, the similarity vector o_2 is combined with the output of the baseline model o_1 to get the final output vector $o \in \mathbb{R}^{|C|}$ as

$$o = o_1 + o_2 \tag{8}$$

$$\hat{y} = sigmoid(o), \tag{9}$$

where $\hat{y} \in \mathbb{R}^{|C|}$ is the predicted label vector. True labels are also represented as a binary vector $y \in \{0,1\}^{|C|}$ and the training procedure minimizes the binary cross-entropy loss between the predicted values and the true values,

$$L_{BCE}(N, y) = -(\sum_{i=1}^{|C|} y_i \log(\hat{y_i}) + (1 - y_i) \log(1 - \hat{y_i})), \tag{10}$$

where we sum the binary cross entropy errors over all dimensions of $\hat{y}$. Note that equation (10) is the loss for one note. In our implementation, we calculate the average loss for multiple notes. We use Adam optimizer [20] to minimize the above loss values. For deep learning based models (CNN and RNN based models), it typically takes a long time to converge. Therefore, in our training, we first deactivate the document similarity learning part of the model to let the baseline model converge. We activate the document similarity learning part and disable parameters updating in the baseline model when the performance of baseline model stops increasing on the validation set as the early stopping criteria.

3.3 Interpreting KSI framework

Interpretability is a model's ability to provide rationale behind its behavior. In KSI framework, the larger the s_i is, the more evidence Wikidedia knowledge source provides for predicting code c_i. To understand what evidence Wikipedia knowledge source provides, we need to calculate the contribution of each variable in x_i to the final similarity score. By combining equations (6) and (5), we get

$$s_i = v_o^T v_i + b_o = v_o^T (\alpha_i \odot e_i) + b_o. \tag{11}$$

Since α_i are weights of e_i, we keep them fixed and decompose e_i. As in equation (3), e_i is the sum of the columns of W_e weighted by each variable in z_i, and we can rewrite equation (11) as

$$\begin{aligned} s_i &= v_o^T (\alpha_i \odot (\sum_{j=1}^{|W|} z_{i,j} W_e[:, j])) + b_o \\ &= \sum_{j=1}^{|W|} v_o^T (\alpha_i \odot W_e[:, j]) z_{i,j} + b_o. \end{aligned} \tag{12}$$

Using equation (2), we can further decompose $z_{i,j}$ as the product of $x_{i,j}$ and q_j,

$$s_i = \sum_{j=1}^{|W|} v_o^T(\alpha_i \odot W_e[:,j]q_j)x_{i,j} + b_o. \quad (13)$$

Since $x_{i,j}$ is a binary value indicating the presence of word w_j in document k_i, $\lambda(x_{i,j}) = v_o^T(\alpha_i \odot W_e[:,j]q_j)$ is the coefficient for $x_{i,j}$ and measures how much word w_j in k_i contributes to final similarity s_i.

4 EXPERIMENTAL SETUP

4.1 DataSets

In our experiments we use clinical notes in MIMIC-III dataset and Wikipedia pages of ICD-9 diagnosis codes. Both datasets are publicly available. The source code of KSI is available at https://github.com/tiantiantu/KSI.

Clinical notes dataset: MIMIC-III Critical Care Database [18] is the largest publicly available electronic health records dataset which contains de-identified health records of 46,518 patients who stayed in the Beth Israel Deaconess Medical Center's Intensive Units from 2001 to 2012. We use all discharge summary notes and the accompanying ICD-9 diagnosis codes. The total number of discharge summary notes is 59,652. For each patient visit, we aggregated all discharge summary notes. The number of aggregated discharge summary notes is 52,722. On average, in each aggregated note there are 1,596 words. During preprocessing we lowercased all tokens and removed punctuations, stop words, words containing only digits, and words whose frequency is less than 10. The final word vocabulary contains 47,965 unique words. We extracted all listed ICD-9 diagnosis codes for each visit and grouped them by their first three digits. On average, each visit has 11 medical codes. The code vocabulary contains 942 codes. Of those codes, we selected a subset of 344 codes for which we found the corresponding Wikipedia document and used those codes in our experiments.

Wikipedia knowledge dataset: Wikipedia maintains a web page including all ICD-9 diagnosis codes and links to their web pages if available[1]. Out of the 389 Wikipedia pages available for the first three digits ICD-9 diagnosis codes, we found 344 of them in MIMIC-III medical code vocabulary. We extracted Wikipedia documents for these 344 medical codes and transformed them into pure text. Then we removed punctuations and lowercased all tokens. On average each processed document has 1,058 words. The size of the word vocabulary of Wikipedia documents is 60,968, out of which only 12,173 are also in the word vocabulary of MIMIC-III clinical notes, showing that the two types of documents have very different word distributions.

4.2 Baseline model selection

We selected 5 baseline models f for this study. In our experiments, we use them stand-alone and as a component of the KSI framework that incorporates Wikipedia knowledge in order to evaluate the improvement the KSI framework brings. We briefly introduce the selected baseline models below.

[1]https://en.wikipedia.org/wiki/List_of_ICD-9_codes

Logistic regression (LR). First, we represent note N as binary bag-of-words vector $n \in \{0,1\}^{|W|}$. Then, we use a weight matrix $W_{LR} \in \mathbb{R}^{|C|\times|W|}$ to map feature space to label space as

$$o_1 = W_{LR}n. \quad (14)$$

For all baseline models, when combined with the KSI framework we use equations (8) and (9) to get the predicted values. When used as a stand-alone model, we apply sigmoid function to its output,

$$\hat{y} = sigmoid(o_1). \quad (15)$$

Recurrent neural network (RNN). We first project each word in note N to a low-dimensional vector $emb \in \mathbb{R}^m$. Then we transform the sequence of words into a sequence of word embeddings and feed it to a recurrent neural network,

$$h_1, h_2, ..., h_{|N|} = RNN(emb_1, emb_2, ..., emb_{|N|}), \quad (16)$$

in which hidden vector $h_i \in \mathbb{R}^{d_o}$ captures the context information of emb_i. Then, max-pooling is used to compute the sequence vector $g \in \mathbb{R}^{d_o}$ and transform it into output o_1 as

$$g(i) = \max_k h_{k,i} \quad (17)$$

$$o_1 = W_{RNN} \cdot g + b_{RNN}, \quad (18)$$

where $W_{RNN} \in \mathbb{R}^{|C|\times d_o} \in$ and $b_{RNN} \in \mathbb{R}^{|C|}$. In our experiments we use the long short-term memory network [15] as the specific RNN selection.

RNN with attention (RNNatt). Instead of using max-pooling of hidden vectors as the sequence vector, we can apply label-specific attentions to them. Let $H = [h_1, h_2, ..., h_{|N|}]$. For each label $c_j \in C$ we use $u_j \in \mathbb{R}^{d_o}$ to calculate the attention as

$$\alpha_j = softmax(H^T u_j), \quad (19)$$

where $softmax = \frac{\exp(x)}{\sum_i \exp(x_i)}$ and $\exp(x)$ is the element-wise exponentiation of x. Then we calculate vector representation for label c_j as

$$p_j = \sum_{k=1}^{|N|} \alpha_{j,k} h_k, \quad (20)$$

and the final likelihood for label c_j in o_1 is

$$o_1(j) = \beta_j^T p_j + b_j, \quad (21)$$

in which $\beta_j \in \mathbb{R}^{d_o}$ and b_j is a scalar bias.

Convolutional neural network (CNN) [19]. As in the RNN case, we first project each word in note N to a low-dimensional vector $emb \in \mathbb{R}^m$ and transform the sequence of words into a sequence of word embeddings. Next, we use a convolutional filter $W_{CNN1} \in \mathbb{R}^{k\times m\times d_o}$ to aggregate context information, where k is the filter width and d_o is the size of filter output. Then for each word embedding emb_n, we use W_{CNN1} to capture its context as

$$h_n = ReLU(W_{CNN1} * emb_{n:n+k-1} + b_{CNN1}), \quad (22)$$

where $*$ is the convolution operator, $ReLU$ is the rectified linear unit and $b_{CNN1} \in \mathbb{R}^{d_o}$ is the bias. We pad both sides of the input to make $H = [h_1, h_2, ..., h_{|N|}] \in \mathbb{R}^{d_o\times|N|}$. Here, as in the RNN case, H is the matrix capturing context information of each word. In [19], max-pooling is used to compute the sequence vector as

$$g(i) = \max_k h_{k,i}. \quad (23)$$

Table 3: Label frequency distribution

Frequency range	Number of unique codes	Percentage of code occurrences
1-10	80	0.1%
11-50	73	0.6%
51-100	25	0.6%
101-500	82	6.7%
>500	84	92.0%

Finally, $g \in \mathbb{R}^{d_o}$ is mapped to the label space:

$$o_1 = W_{CNN2} \cdot g + b_{CNN2}, \tag{24}$$

where $W_{CNN2} \in \mathbb{R}^{|C| \times d_o}$ and $b_{CNN2} \in \mathbb{R}^{|C|}$.

Convolutional Attention (CAML) [29]: CAML combines the attention mechanism with CNN and achieves the state-of-the-art performance on medical code prediction. After calculating context matrix H in equation (22), instead of applying max-pooling in equation (23), CAML applies label-specific attentions following equations (19) (20) (21). For each label $c_j \in C$, we use $u_j \in \mathbb{R}^{d_o}$ to calculate the attention as

$$\alpha_j = softmax(H^T u_j). \tag{25}$$

Then we can calculate vector representation for label c_j as

$$p_j = \sum_{k=1}^{|N|} \alpha_{j,k} h_k, \tag{26}$$

and the final likelihood for label c_j in o_1 as

$$o_1(j) = \beta_j^T p_j + b_j, \tag{27}$$

where $\beta_j \in \mathbb{R}^{d_o}$ and b_j is a scalar bias.

4.3 Evaluation metric

In order to properly evaluate different models, we report six different metrics: average loss value, top-10 recall, macro-averaged AUC, micro-averaged AUC, macro-averaged F1 and micro-averaged F1. The average loss value is the average of binary cross-entropy defined in equation (10) over all test notes. Top-10 recall is defined as the number of correct medical codes ranked in the top $\max(10, |M|)$ in $\hat{y}$ divided by $|M|$, where $|M|$ is the number of medical codes for the note, and averaged over all test notes. This metric is motivated by the potential use case in computer-aided coding where the system recommends several codes for human experts to review. AUC (area under ROC curve) measures the probability that a classifier ranks a randomly chosen positive instance higher than a randomly chosen negative instance. F1 score is defined as the harmonic mean between the precision and the recall. Micro-averaged values are calculated by aggregating instances of all classes and calculating the average metric over all instances. Therefore, this metric will be dominated by the most frequent medical codes. Medical codes in our dataset (3-digit ICD-9 codes which have Wikipedia webpage) are highly imbalanced. As shown in Table 3, the most common 84 codes account for 92% of all code occurrences. On the other hand, the macro-averaged metric first calculates the value for each medical code separately and then takes the average over all the codes. Since a frequent class is weighted the same as a rare class, the macro-averaged metrics puts a higher emphasis on rare medical code prediction.

Implementation details. All models are implemented with Pytorch. We randomly divide the dataset into training, validation and test sets in a 0.7, 0.1, 0.2 ratio. Early stopping on the validation set is used to determine when to stop training. Training terminates if top-10 recall does not improve in the next 5 epochs and the model with the highest top-10 recall is used on the test set. Each batch contains 32 notes. We use Adam optimizer with the learning rate of 0.001 to minimize the loss function. The dimensionality of word embedding m and hidden vector of RNN and CNN d_o are set to 100 and 200, respectively.

5 EXPERIMENTAL RESULTS

5.1 Predictive results

We show the accuracies of the baseline models alone and with the KSI framework in Table 4. From the table we can see that regardless of the baseline model, KSI framework consistently improves the accuracy. Although different metrics measure different aspect of model performance, the KSI framework improves most of them. Overall, the micro-averaged score is much higher than the macro-averaged score, as expected because the macro-averaged score emphasizes performance on the rare medical codes on which the number of available training examples is too small for accurate learning. On the other hand, in the micro-averaged scores medical codes contribute proportionally to their frequency and the frequent medical codes such as diabetes and essential hypertension provide more examples for accurate learning.

Overall, deep learning models outperform the logistic regression model, and KSI framework improves these complicated models further, showing the effectiveness of exploiting external online knowledge sources. The attention mechanism based models (RNNAtt and CAML) outperform other models while CAML achieves the best overall performance compared to the other baseline models. The success of CAML and RNNatt could be attributed to their multi-label attention mechanism. For each label, both models use a label-specific weight matrix to generate label specific attentions over all words in the text. Therefore, they are able to pay attention to locations of text segments which refer to a specific disease, which is important for medical notes since evidence about different diseases can occur in different parts of clinical notes.

It is worth noting that KSI framework improves all macro F1 and macro AUC of all models by a large margin. For macro AUC: 4.8% for LR, 3.1% for RNN, 3.0% for RNNatt, 3.4% for CNN and 3.6% for CAML. As macro AUC is the unweighted average AUC over all labels we hypothesize that the KSI framework is able to improve AUC for labels which are not so frequent in the dataset. In particular, since the frequent medical codes have a larger number of examples for accurate learning, they stand to benefit less from the external sources. On the contrary, the rare codes lack examples and are benefiting greatly from the external sources.

To validate this, we divide medical codes into 5 groups based on their frequencies in the dataset: [1, 10], [11, 50], [51, 100], [101, 500] and [500, $+\infty$). Then, we calculate macro-averaged AUC of each medical code group for RNN, RNNatt, CNN, and CAML baselines

Table 4: The predictive performance of baseline models alone and with KSI framework.

Method	Macro AUC	Micro AUC	Macro F1	Micro F1	Test loss value	Top-10 recall
LR	0.716	0.956	0.167	0.554	**0.048**	0.724
KSI+LR	**0.764**	**0.961**	**0.196**	**0.557**	0.055	**0.738**
RNN	0.854	0.972	0.204	0.653	0.032	0.772
KSI+RNN	**0.885**	**0.978**	**0.244**	**0.662**	**0.030**	**0.798**
RNNatt	0.850	0.973	0.264	0.647	0.035	0.791
KSI+RNNatt	**0.880**	**0.976**	**0.300**	**0.659**	**0.033**	**0.802**
CNN	0.825	0.968	0.214	0.626	0.040	0.753
KSI+CNN	**0.859**	**0.973**	**0.237**	**0.637**	**0.039**	**0.775**
CAML	0.855	0.978	0.257	0.656	0.032	0.806
KSI+CAML	**0.891**	**0.980**	**0.285**	**0.659**	0.032	**0.814**

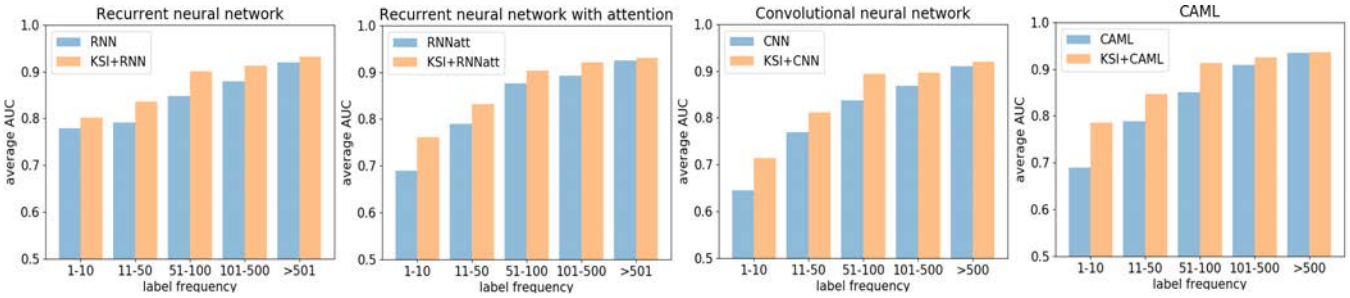


Figure 3: Macro-averaged AUC by label frequency group for RNN, RNNatt, CNN and CAML. x-axis denotes the label frequency group and y-axis denotes the macro-averaged AUC for each group.

and their counterparts under the KSI framework. The results are summarized in Figure 3.

As shown in the figure, different methods perform differently on each medical code group. Overall, the average AUC of medical code prediction increases with the code frequency. For the least frequent group (occurring less than 10 times), the average AUC of medical codes is less than 0.8. For the most common group of medical codes (occurring more than 500 times), all methods achieve an averaged AUC of more than 0.9. The figure also shows the trend in which the KSI improvement decreases with the frequency of medical codes in the dataset. For example, for CAML model, KSI framework brings 9.6% improvement of AUC on the least common [1-10] group. On the [11-50] and [51-100] groups, KSI brings 5.8% and 6.3% improvement, respectively. On the more common groups [101-500] and [500,+∞), the KSI framework only increases AUC by 1.5% and 0.1% respectively. The benefit of the KSI framework on rare medical codes can intuitively be explained in the following way. An intersection of a note and a Wikipedia document can be viewed as a feature selection; it significantly narrows down the feature space and makes it easier for the attention mechanism to find the evidence for the rare medical codes. To validate this, in the following paragraph we take a look at the contribution coefficient we derived in section 3.3.

5.2 Informative evidence extraction

The success of medical code prediction depends on the ability to extract evidence for inferring the specific codes. In equation (13), we can calculate coefficient $\lambda(x_{i,j})$ of each word w_j for document k_i. $\lambda(x_{i,j})$ depicts the contribution of each word to predicting code c_i. We calculate the coefficient for each word in the document and sort the words by their values. We study 10 words with the highest coefficients in order to gain insight into the performance of the KSI framework. In Table 5 we show the top 10 words according to the document similarity learning sub-network (subtask 2) and we also show the top 10 substrings extracted by the baseline sub-network (subtask 1). For the table, we used CAML as our baseline sub-network since it outperformed other baseline models. In equation (25), CAML calculates the attentions α_j for all words in a clinical note. These attentions are specific to medical code $c_j \in C$, the higher the attentions, the more the underlying words contribute to predicting c_j. Since in equation (22) CAML uses a convolutional layer to aggregate context information for a given word, the attentions are generated on top of the n-grams, where n equals to the filter size. Since in our experiments we set the filter size n to 3, the table is showing the top 10 trigrams with the highest attentions.

In Table 5 we show results for three representative diseases, both frequent and rare ones in order to better understand how KSI framework creates its two sub-networks.

As shown in the table, the first medical code is “250" (diabetes mellitus), which is one of the most frequent codes in our dataset. Since CAML sub-network alone is able to predict this disease with AUC as high as 0.97, the similarity sub-network cannot improve the performance any further. By checking the trigrams with the highest coefficients, we find that CAML sub-network finds highly meaningful substrings for inferring diabetes such as “failure diabetes mellitus" and “significant diabetes mellitus". By contrast, the similarity sub-network assigns the largest weight to word “name" rather than to some words that are more directly related to diabetes.

Table 5: Words/trigrams with highest contribution by document similarity learning (subtask 2) and baseline model alone (subtask 1) for medical codes 250, 138 and 501.

Code: 250 (Diabetes mellitus) Count: 13,804			
Subtask 2 Final AUC: 0.97		**Subtask 1 (CAML) Final AUC: 0.97**	
Word	Weight	Trigram	Weight
name	0.69	failure diabetes mellitus	0.29
mellitus	0.44	significant diabetes mellitus	0.19
coronary	0.19	diabetes mellitus treated	0.15
different	0.15	diabetes mellitus chronic	0.11
date	0.14	dictated bylast name	0.05
showed	0.13	hematocrit percent past	0.05
oral	0.08	fraction percent hypokinesis	0.04
number	0.07	md number1 dictated	0.02
diabetes	0.07	heart failure diabetes	0.01
white	0.07	percent hypokinesis right	0.01
Code: 138 (Polio, late effects) Count: 68			
Subtask 2 Final AUC: 0.98		**Subtask 1 (CAML) Final AUC: 0.74**	
Word	Weight	Trigram	Weight
polio	2.48	angioseal device deployed	0.15
puncture	0.33	anxiety per pt	0.10
proximal	0.22	postcath restarted oa	0.06
illness	0.18	qd gi cocktail	0.06
femoral	0.14	hypotension pt remained	0.06
constipation	0.12	hypotension postcath restarted	0.04
hospital	0.08	ptca comments french	0.03
surgery	0.08	stricture web although	0.03
continued	0.08	discharge date service	0.03
condition	0.08	protonix qd gi	0.02
Code: 501 (Asbestosis) Count: 115			
Subtask 2 Final AUC: 0.96		**Subtask 1 (CAML) Final AUC: 0.76**	
Word	Weight	Trigram	Weight
asbestosis	5.21	asbestos exposure hyperlipidemia	0.95
transferred	0.68	medical history cad	0.01
inhalation	0.53	discharge diagnosis cad	0.01
illness	0.45	history cad sp	0.004
condition	0.30	one day five	0.003
two	0.17	cad sp pci	0.002
breath	0.16	cad sp pci	0.002
family	0.10	disp50 tablets refills	0.002
shortness	0.08	date date birth	0.002
current	0.01	homecare discharge diagnosis	0.002

This can be explained as follows: since there are many notes in the training dataset to allow the CAML sub-network to achieve high accuracy, errors back-propagated into the similarity sub-network are small and the impact of code “250" on any of its weights is small. Next, we take a look at two less common diseases on which CAML sub-networks achieves smaller accuracy.

The second medical code shown in the table is “138" (polio, late effects). This code occurs only in 68 notes. AUC of the CAML sub-network is only 0.74. The similarity sub-network is very helpful and it results in increasing the AUC to 0.98. Among the top 10 trigrams of the CAML sub-network, we see that only “hypotension" is vaguely related to poliomyelitis. On the other hand, the similarity sub-network identifies highly important terms such as “polio", “puncture" and “constipation", which are all related to poliomyelitis problems. These words are mentioned both in Wikipedia document about “poliomyelitis" and in the clinical notes and are assigned large coefficients.

Table 6: Intersection of a clinical note and Wikipedia document for medical code “214" (Lipoma; count: 33) and final coefficient for each word.

After taking intersection (containing only 35 words):
known surgical present cm last first prior one medical family physical exam soft results blood disease possible identified multiple greater diagnosis condition internal years size tissue obese include removed wound common tumors minor lipomas
Final weights for words in intersection set:
obese: 1.42 lipomas: 0.70 tumors: 0.48 minor: 0.30 soft: 0.29 last: 0.24 wound: 0.22 tissue: 0.13 ... ...

Finally, we look at another rare code “501" (asbestosis). CAML sub-network recognizes important word “asbestos". The similarity sub-network identifies related words “asbestosis", “inhalation", “breath" and “shortness", which are all symptoms of asbestosis. The similarity sub-network improves the AUC for this disease from 0.76 to 0.96.

The strengths of the KSI framework in finding the important evidence can be explained by the way it works: the main obstacle to finding evidence in clinical notes for rare medical codes is that they do not have enough occurrences for models to select useful features from a large feature space. For example, an average note contains 1,596 words and CAML calculates an attention value for each word in the document. Thus, it is difficult for CAML to learn the attention values among more than a thousand words when medical codes are rare. On the other hand, external knowledge sources such as Wikipedia contain important words (signs, symptoms, typical treatments) for each disease, which can be used to narrow down the feature space. On average, each clinical note contains 407 unique words and each Wikipedia document contains 517 unique words. After taking their intersection, the resulting intersection set only contains an average of 64 words, which is much smaller than the original feature space. Then, the KSI framework learns weight for each word in the intersection set, which is much easier compared to learning weights from the original feature space. An example intersection set and final weights are shown in Table 6. Even for very rare labels such as “lipoma" the KSI framework is able to learn large weights for relevant words. We will also show the effectiveness of the intersection process in the next subsection.

5.3 Ablation study

To study contribution of each component in the KSI framework to learning the document similarity score, we compare our method to several variations of the framework. To focus on accuracy of different variations, in this subsection we fix the baseline model to be CAML and we study the effect of changing components of document similarity learning part on accuracy. We consider the following variations:

Table 7: The predictive performance of KSI and its variations.

Method	Macro AUC	Micro AUC	Macro F1	Micro F1	Test loss value	Top-10 recall
KSI-attention	0.871	0.978	0.255	0.654	0.032	0.807
KSI-intersection	0.864	0.979	0.263	**0.662**	0.032	0.809
KSI	**0.891**	**0.980**	**0.285**	0.659	0.032	**0.814**

KSI-attention. In order to study the effect of variable-level attention mechanism, we remove the attention vector in equation (5) and directly use e_i as input to the next layer.

KSI-intersection. In section 5.2 we analyzed the advantage of taking intersection of clinical note vector q and Wikipedia document vector k_i. In this variation we swap the order of element-wise multiplication and embedding. We first embed both q and k_i into low-dimensional vector space as

$$e_q = W_e q \tag{28}$$

$$e_{x_i} = W_e x_i, \tag{29}$$

where $e_q \in \mathbb{R}^m$ and $e_{x_i} \in \mathbb{R}^m$. Both e_q and e_{k_i} contain information about all words in the respective documents. Then we calculate element-wise multiplication of e_q and e_{x_i} as

$$e_i = e_q \odot e_{x_i}, \tag{30}$$

where e_i is used as in equations (4) and (5).

The results are shown in Table 7. From the table we can see that KSI outperforms other variations. Since the baseline model is fixed to CAML, which performs well on frequent labels, all accuracies are similar on micro-averaged metric. However, for macro-averaged metric, we can observe sizable differences. KSI significantly outperforms other methods on macro AUC and macro F1. KSI-intersection performs the worst on macro AUC, showing the importance of the intersection on inferring rare medical codes. KSI-attention also underperforms KSI by 2.0% and 3.0% on macro AUC and macro F1, respectively, showing the effectiveness of attention mechanism on selecting important features for prediction.

5.4 Qualitative insights from coefficients of KSI

Recent research about medical codes prediction [29, 35] observed improved performance by incorporating words in ICD-9 code descriptions into prediction model. These ICD-9 code descriptions are typically very short, with less than 10 words (e.g., "secondary diabetes mellitus with hyperosmolarity"). Nonetheless, it would be very promising to design a small knowledge base specialized for providing evidence for medical code prediction. To design such a knowledge base, it is important to understand which evidence is the most helpful for the task.

Besides serving as a framework that incorporates knowledge into other end-to-end models, the KSI could provide an insight into how Wikipedia helps clinical code prediction. As we showed by calculating coefficient λ, we are able to see which words in the intersection set are influencing the final prediction. We calculated the average coefficient of each word for a given Wikipedia document. Words with high average coefficients are influential and should be considered for future specialized knowledge base construction. We show words with the highest average coefficients for four Wikipedia documents in Table 8.

Table 8: Words with highest average coefficients for four disease Wikipedia documents.

Wiki documents	Words with highest coefficients
Episodic mood disorders	lithium bipolar affective dysregulation mdd manic anxiety psychiatrist hypomanic behavior
Lipoma	lipoma lipomatosis lipomas malignant tumor wound repeat tumors subcutaneous obese
Mild intellectual disabilities	epilepsy retardation service supervision abuse benzodiazepines psychiatry cerebral threatened
Kaposi's sarcoma	sarcoma aids tumor haart kaposi ks antiretroviral chemotherapy meical hiv viral

As shown in the table, the influential words include representative symptoms for a disease such as "dysregulation", "anxiety", "manic" and "retardation". The full name and abbreviation of diseases are other types of highly informative words such as "lipoma", "ks" and "mdd". Besides, risk factors such as "hiv" and "obese" are also important evidence for inferring medical codes. These results shed light on the connection between the knowledge source and clinical notes, which provides guideline for future specialized knowledge-based design.

6 CONCLUSION

Medical code prediction from clinical notes is an active research area. The success of the task depends on finding evidence to infer diagnosis. However, it is challenging to learn what is the corresponding evidence in the notes when medical codes are rare. In this paper we propose a framework called KSI which is able to incorporate an external online knowledge source into any neural network multi-label predictive model. The key component of the KSI framework is to take intersection of a clinical note and external knowledge document, then use an attention mechanism to select important features in the intersection set. Finally, it calculates a matching score between the note and each knowledge document. KSI framework consistently improves the baseline model, especially when predicting rare codes. KSI framework also provides an insight into the evidence from an external source that was useful for the prediction. While our work uses medical codes which have Wikipedia web pages as labels, there is still a substantial number of medical codes without Wikipedia documentation. Our results suggest that it would be worthwhile for the Wikipedia community to create pages for all major ICD-9 and ICD-10 codes to not only enhance the health knowledge of public, but also to bring benefits to predictive modeling of clinical notes.

REFERENCES

[1] Anand Avati, Kenneth Jung, Stephanie Harman, Lance Downing, Andrew Ng, and Nigam H Shah. 2017. Improving palliative care with deep learning. In *Bioinformatics and Biomedicine (BIBM), 2017 IEEE International Conference on*. IEEE, 311–316.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[3] Tian Bai, Ashis Kumar Chanda, Brian L Egleston, and Slobodan Vucetic. 2017. Joint learning of representations of medical concepts and words from ehr data. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 764–769.

[4] Tian Bai, Ashis Kumar Chanda, Brian L Egleston, and Slobodan Vucetic. 2018. EHR phenotyping via jointly embedding medical concepts and words into a unified vector space. *BMC medical informatics and decision making* 18, 4 (2018), 123.

[5] Tian Bai, Shanshan Zhang, Brian L Egleston, and Slobodan Vucetic. 2018. Interpretable Representation Learning for Healthcare via Capturing Disease Progression through Time. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 43–51.

[6] Tal Baumel, Jumana Nassour-Kassis, Raphael Cohen, Michael Elhadad, and Noemie Elhadad. 2017. Multi-label classification of patient notes a case study on ICD code assignment. *arXiv preprint arXiv:1709.09587* (2017).

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[8] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*. 301–318.

[9] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. 2016. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*. 3504–3512.

[10] Koby Crammer, Mark Dredze, Kuzman Ganchev, Partha Pratim Talukdar, and Steven Carroll. 2007. Automatic code assignment to medical text. In *Proceedings of the workshop on bionlp 2007: Biological, translational, and clinical language processing*. Association for Computational Linguistics, 129–136.

[11] Luciano RS de Lima, Alberto HF Laender, and Berthier A Ribeiro-Neto. 1998. A hierarchical approach to the automatic categorization of medical documents. In *Proceedings of the seventh international conference on Information and knowledge management*. ACM, 132–139.

[12] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science* 41, 6 (1990), 391–407.

[13] Richárd Farkas and György Szarvas. 2008. Automatic construction of rule-based ICD-9-CM coding systems. In *BMC bioinformatics*, Vol. 9. BioMed Central, S10.

[14] Jacob Goldberger, Geoffrey E Hinton, Sam T Roweis, and Ruslan R Salakhutdinov. 2005. Neighbourhood components analysis. In *Advances in neural information processing systems*. 513–520.

[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[16] Karla L Hoffman and Ted K Ralphs. 2013. Integer and combinatorial optimization. In *Encyclopedia of Operations Research and Management Science*. Springer, 771–783.

[17] Gao Huang, Chuan Guo, Matt J Kusner, Yu Sun, Fei Sha, and Kilian Q Weinberger. 2016. Supervised word mover's distance. In *Advances in Neural Information Processing Systems*. 4862–4870.

[18] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data* 3 (2016), 160035.

[19] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[21] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *International Conference on Machine Learning*. 957–966.

[22] Leah S Larkey and W Bruce Croft. 1995. *Automatic assignment of icd9 codes to discharge summaries*. Technical Report. Technical report, University of Massachusetts at Amherst, Amherst, MA.

[23] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*. 1188–1196.

[24] Nathan Levitan, A Dowlati, SC Remick, HI Tahsildar, LD Sivinski, R Beyth, and AA Rimm. 1999. Rates of initial and recurrent thromboembolic disease among patients with malignancy versus those without malignancy. *Risk analysis using Medicare claims data. Medicine (Baltimore)* 78, 5 (1999), 285–91.

[25] Lucian Vlad Lita, Shipeng Yu, Stefan Niculescu, and Jinbo Bi. 2008. Large scale diagnostic code classification for medical patient records. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

[26] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[27] Fenglong Ma, Radha Chitta, Jing Zhou, Quanzeng You, Tong Sun, and Jing Gao. 2017. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1903–1911.

[28] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[29] James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable Prediction of Medical Codes from Clinical Text. *arXiv preprint arXiv:1802.05695* (2018).

[30] Adler Perotte, Rimma Pivovarov, Karthik Natarajan, Nicole Weiskopf, Frank Wood, and Noémie Elhadad. 2013. Diagnosis code assignment: models and evaluation metrics. *Journal of the American Medical Informatics Association* 21, 2 (2013), 231–237.

[31] Adler J Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. 2011. Hierarchically supervised latent Dirichlet allocation. In *Advances in Neural Information Processing Systems*. 2609–2617.

[32] Aaditya Prakash, Siyuan Zhao, Sadid A Hasan, Vivek V Datla, Kathy Lee, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2017. Condensed Memory Networks for Clinical Diagnostic Inferencing.. In *AAAI*. 3274–3280.

[33] Alvin Rajkomar, Eyal Oren, Kai Chen, Andrew M Dai, Nissan Hajaj, Peter J Liu, Xiaobing Liu, Mimi Sun, Patrik Sundberg, Hector Yee, et al. 2018. Scalable and accurate deep learning for electronic health records. *arXiv preprint arXiv:1801.07860* (2018).

[34] Narges Razavian, Saul Blecker, Ann Marie Schmidt, Aaron Smith-McLallen, Somesh Nigam, and David Sontag. 2015. Population-level prediction of type 2 diabetes from claims data and analysis of risk factors. *Big Data* 3, 4 (2015), 277–287.

[35] Haoran Shi, Pengtao Xie, Zhiting Hu, Ming Zhang, and Eric P Xing. 2017. Towards Automated ICD Coding Using Deep Learning. *arXiv preprint arXiv:1711.04075* (2017).

[36] Donald H Taylor Jr, Truls Østbye, Kenneth M Langa, David Weir, and Brenda L Plassman. 2009. The accuracy of Medicare claims as an epidemiological tool: the case of dementia revisited. *Journal of Alzheimer's Disease* 17, 4 (2009), 807–815.

[37] Ankit Vani, Yacine Jernite, and David Sontag. 2017. Grounded Recurrent Neural Networks. *arXiv preprint arXiv:1705.08557* (2017).

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 6000–6010.

[39] Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. 2018. Joint Embedding of Words and Labels for Text Classification. *arXiv preprint arXiv:1805.04174* (2018).

[40] Wolfgang C Winkelmayer, Sebastian Schneeweiss, Helen Mogun, Amanda R Patrick, Jerry Avorn, and Daniel H Solomon. 2005. Identification of individuals with CKD from Medicare claims data: a validation study. *American Journal of Kidney Diseases* 46, 2 (2005), 225–232.

[41] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

No Place to Hide: Catching Fraudulent Entities in Tensors

Yikun Ban*
Peking University
banyikun@pku.edu.cn

Xin Liu
Tsinghua University
liuxin16@mails.tsinghua.edu.cn

Yitao Duan
Fintec.ai
duan@fintec.ai

Xue Liu
McGill University
xueliu@cs.mcgill.ca

Wei Xu
Tsinghua University
weixu@tsinghua.edu.cn

ABSTRACT

Many approaches focus on detecting dense blocks in the tensor of multimodal data to prevent fraudulent entities (e.g., accounts, links) from retweet boosting, hashtag hijacking, link advertising, etc. However, no existing method is effective to find the dense block if it only possesses high density on a subset of all dimensions in tensors. In this paper, we novelly identify dense-block detection with dense-subgraph mining, by modeling a tensor into a weighted graph without any density information lost. Based on the weighted graph, which we call information sharing graph (ISG), we propose an algorithm for finding multiple densest subgraphs, D-Spot, that is faster (up to 11x faster than the state-of-the-art algorithm) and can be computed in parallel. In an N-dimensional tensor, the entity group found by the ISG+D-Spot is at least 1/2 of the optimum with respect to density, compared with the 1/N guarantee ensured by competing methods. We use nine datasets to demonstrate that ISG+D-Spot becomes new state-of-the-art dense-block detection method in terms of accuracy specifically for fraud detection.

CCS CONCEPTS

• **Information systems → Wrappers (data mining)**.

KEYWORDS

Dense-block Detection; Graph Algorithms; Fraud Detection

ACM Reference Format:
Yikun Ban, Xin Liu, Yitao Duan, Xue Liu, and Wei Xu. 2019. No Place to Hide: Catching Fraudulent Entities in Tensors. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313403

1 INTRODUCTION

Fraud represents a serious threat to the integrity of social or review networks such as Twitter and Amazon, with people introducing fraudulent entities (e.g., fake accounts, reviews, etc.) to gain more publicity/profit over a brief period. For example, on a social network or media sharing website, people may wish to enhance their account's popularity by illegally buying more followers [27]; on e-commerce websites, fraudsters may register multiple accounts to benefit from "new user" promotions.

Consider the typical log data generated from a social review site (e.g., Amazon), which contains four-dimensional features: users, products, timestamps, rating scores. These data are often formulated as a tensor, in which each dimension denotes a separate feature and an entry (tuple) of the tensor represents a review action. Based on previous studies [12, 30], fraudulent entities form dense blocks (subtensors) within the main tensor, such as when a mass of fraudulent user accounts create an enormous number of fake reviews for a set of products over a short period. Dense-block detection has also been applied to network intrusion detection [20, 30], retweet boosting detection [12], bot activities detection [30], and genetics applications [20, 26].

Various dense-block detection methods have been developed. One approach uses tensor decomposition, such as CP decomposition and higher-order singular value decomposition [20]. However, as observed in [32], such methods are outperformed by search-based techniques [12, 30, 32] in terms of accuracy, speed, and flexibility regarding support for different density metrics. Furthermore, [30, 32] provide an approximation guarantee for finding the densest/optimal block in a tensor.

We have examined the limitations of search-based methods for dense-block detection. First, these methods are incapable of detecting *hidden-densest blocks*. We define a hidden-densest block as one that does not have a high-density signal on all dimensions of a tensor, but evidently has a high density on a subset of all dimensions. Moreover, existing methods neglect the data type and distribution of each dimension on the tensor. Assuming that two dense blocks A and B have the same density, however, A is the densest on a subset of critical features, such as IP address and device ID, whereas B is the densest on some trivial features such as age and gender. Can we simply believe that A is as suspicious as B? Unfortunately, the answer when using existing methods is 'yes.'

To address these limitations, we propose a dense-block detection framework and focus on entities that form dense blocks on tensors. The proposed framework is designed using a novel approach. Given a tensor, the formation of dense blocks is the result of value sharing (the behavior whereby two or more different entries share a distinct value (entity) in the tensor). Based on this key point, we propose a novel *Information Sharing Graph (ISG)* model, which accurately captures each instance of value sharing. The transformation from dense blocks in a tensor to dense subgraphs in ISG leads us to propose a fast, high-accuracy algorithm, D-Spot, for determining fraudulent entities with a provable guarantee regarding the densities of the detected subgraphs.

*Yikun did the project during his visit at Tsinghua University.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313403

In summary, the main contributions of this study are as follows:

1) **[Graph Model].** We propose the novel ISG model, which converts every value sharing in a tensor to the representation of weighted edges or nodes (entities). Furthermore, our graph model considers diverse data types and their corresponding distributions based on information theory to automatically prioritize multiple features.

2) **[Algorithm].** We propose the *D-Spot* algorithm, which is able to find multiple densest subgraphs in one run. And we theoretically prove that the multiple subgraphs found by D-Spot must contain some subgraphs that are at least 1/2 as dense as the optimum. In real-world graphs, D-Spot is up to 11× faster than the state-of-the-art competing algorithm.

3) **[Effectiveness].** In addition to dense blocks, ISG+D-Spot also effectively differentiates hidden-densest blocks from normal ones. In experiments using eight public real-world datasets, ISG+D-Spot detected fraudulent entities more accurately than conventional methods.

2 BACKGROUND

2.1 Economics of Fraudsters

As most fraudulent schemes are designed for financial gain, it is essential to understand the economics behind the fraud. Only when the benefits to a fraudster outweigh their costs will they perform a scam.

To maximize profits, fraudsters have to share/multiplex different resources (e.g., fake accounts, IP addresses, and device IDs) over multiple frauds. For example, [13] found that many users are associated with a particular group of followers on Twitter; [36] identified that many cases of phone number reuse; [4] observed that the IP addresses of many spam proxies and scam hosts fall into a few uniform ranges; and [38] revealed that fake accounts often conduct fraudulent activities over a short time period.

Thus, fraudulent activities often form dense blocks in a tensor (as described below) because of this resource sharing.

2.2 Related Work

Search-based dense-block detection in tensors. Previous studies [12, 20, 30] have shown the benefit of incorporating features such as timestamps and IP addresses, which are often formulated as a multi-dimensional tensor. Mining dense blocks with the aim of maximizing a density metric on tensors is a successful approach. CrossSpot [12] randomly chooses a seed block and then greedily adjusts it in each dimension until the local optimum is attained. This technique usually requires enormous seed blocks and does not provide any approximation guarantee for finding the global optimum. In contrast to adding feature values to seed blocks, M-Zoom [30] removes feature values from the initial tensor one by one using a similar greedy strategy, providing a $1/N$-approximation guarantee for finding the optimum (where N is the number of dimensions in the tensor). M-Biz [31] also starts from a seed block and then greedily adds or removes feature values until the block reaches a local optimum. Unlike M-Zoom, D-Cube [32] deletes a set of feature values on each step to reduce the number of iterations, and is implemented in a distributed disk-based manner. D-Cube provides the same approximation guarantee as M-Zoom.

Table 1: ISG+D-Spot vs. existing dense-block detection methods.

	MAF [20]	CrossSpot [12]	M-zoom [30]	M-biz [31]	D-cube[32]	ISG+D-Spot
Applicable to N-dimensional data?	√	√	√	√	√	√
Catch densest blocks?	√	√	√	√	√	√
Catch hidden-densest blocks?	×	×	×	×	×	√
%-Approximation Guarantee?	×	×	1/N	×	1/N	1/2

Tensor decomposition methods. Tensor decomposition [17] is often applied to detect dense blocks within tensors [20]. Scalable algorithms, such as those described in [23, 33, 37], have been developed for tensor decomposition. However, as observed in [12, 32], these methods are limited regarding the detection of dense blocks, and usually detect blocks with significantly lower densities, provide less flexibility with regard to the choice of density metric, and do not provide any approximation guarantee.

Dense-subgraph detection. A graph can be represented by a two-dimensional tensor, where an edge corresponds to a non-zero entry in the tensor. The mining of dense subgraphs has been extensively studied [18]. Detecting the densest subgraph is often formulated as finding the subgraph with the maximum average degree, and may use exact algorithms [10, 16] or approximate algorithms [6, 16]. Fraudar [11] is an extended approximate algorithm that can be applied to fraud detection in social or review graphs. CoreScope [29] tends to find dense subgraphs in which all nodes have a degree of at least k. Implicitly, singular value decomposition (SVD) also focuses on dense regions in matrixes. EigenSpoke [24] reads scatter plots of pairs of singular vectors to find patterns and chip communities, [7] extracts dense subgraphs using a spectral cluster framework, and [14, 27] use the top eigenvectors from SVD to identify abnormal users.

Other anomaly/fraud detection methods The use of belief propagation [3, 22] and HITS-like ideas [8, 9, 13] is intended to catch rare behavior patterns in graphs. Belief propagation has been used to assign labels to the nodes in a network representation of a Markov random field [3]. When adequate labeled data are available, classifiers can be constructed based on multi-kernel learning [2], support vector machines [35], and k-nearest neighbor [34] approaches.

3 DEFINITIONS AND MOTIVATION

In this section, we introduce the notations and definitions used throughout the paper, analyze the limitations of existing approaches, and describe our key motivations.

3.1 Notation and Formulations

Table 2 lists the notations used in this paper. We use $[N] = \{1, ..., N\}$ for brevity. Let $\mathbf{R}(A_1, ..., A_N, X) = \{t_0, ..., t_{|X|}\}$ be a relation with N-dimensional features, denoted by $\{A_1, ..., A_N\}$, and a dimensional entry identifiers, denoted by X. For each entry (tuple) $t \in \mathbf{R}$, $t = (a_1, ..., a_N, x)$, where $\forall n \in [N]$, we use $t[A_n]$ to denote the value of A_n in t, $t[A_n] = a_n$ and $t[X]$ to denote the identification of t, $t[X] = x$, $x \in X$. We define the mass of $\mathbf{R}$ as $|\mathbf{R}|$, which is the total number of such entries, $|\mathbf{R}| = |X|$. For each $n \in [N]$, we use $\mathbf{R}_n$ to

Table 2: Symbols and Definitions

Symbol	Interpretation
N	number of dimensions in a tensor
$[N]$	set {1,..., N}
$\mathbf{R}(A_1, \ldots, A_N, X)$	relation representing a tensor
A_n	n-th dimensional values of $\mathbf{R}$
$\mathbf{R}_n$	set of distinct values of A_n of $\mathbf{R}$
$t = (a_1, \ldots, a_N, x)$	an entry (tuple) of $\mathbf{R}$
$\mathcal{B}(\mathcal{A}_1, \ldots, \mathcal{A}_N, \mathcal{X})$	a block in $\mathbf{R}$
$\mathcal{B}_n$	set of distinct values of $\mathcal{A}_n$ of $\mathcal{B}$
U	target dimension in $\mathbf{R}$
$\mathbf{V} = \{u_1, \ldots, u\}$	set of distinct values of U
$\mathbf{G} = (\mathbf{V}, \mathbf{E})$	Information Sharing Graph
$\mathcal{S}_{i,j}$	$\mathcal{S}$-score between u_i and u_j
$\mathcal{S}_i$	$\mathcal{S}$-score of u_i
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	subgraph in $\mathbf{G}$
$\mathcal{F}$	density metric

denote the set of distinct values of A_n. Thus, $\mathbf{R}$ naturally represents an N-dimensional tensor of size $|\mathbf{R}_1| \times \ldots \times |\mathbf{R}_N|$.

A block $\mathcal{B}$ in $\mathbf{R}$ is defined as $\mathcal{B}(\mathcal{A}_1, \ldots, \mathcal{A}_N, \mathcal{X}) = \{t \in \mathbf{R} : t[X] \in \mathcal{X}\}$ and $\mathcal{X} \subseteq X$. Additionally, the mass $|\mathcal{B}|$ is the number of entries of $\mathcal{B}$ and $\mathcal{B}_n$ is the set of distinct values of $\mathcal{A}_n$. Let $\mathcal{B}(a, A_n) = \{t \in \mathbf{R} : t[A_n] = a\}$ represent all entries that take the value a on A_n. The mass $|\mathcal{B}(a, A_n)|$ is the number of such entries. A simple example is given as follows.

Example 1 (Amazon review logs). *Assume a relation* $\mathbf{R}(\underline{user}, \underline{product}, \underline{timestamp}, X)$*, where* $\forall t \in \mathbf{R}$*,* $t = (a_1, a_2, a_3, x)$ *indicates a review action where user* a_1 *reviews product* a_2 *at timestamp* a_3*, and the identification of the action is* x*. Because* a_1 *may review* a_2 *at* a_3 *(we assume that* a_3 *represents a period) multiple times,* X *helps us distinguish each such action. The mass of* $\mathbf{R}$*, denoted by* $|\mathbf{R}|$*, is the number of all review actions in the dataset. The number of distinct users in* $\mathbf{R}$ *is* $|\mathbf{R}_1|$*. A block* $\mathcal{B}(a_1, user)$ *is the set of all rating entries operated by user* a_1*, and the number of overall entries of* $\mathcal{B}(a_1, user)$ *is* $|\mathcal{B}(a_1, user)|$*.*

First, we present a density metric that is known to be useful for fraud detection [30, 32]:

Definition 1. (Arithmetic Average Mass ρ). Given a block $\mathcal{B}(\mathcal{A}_1, \ldots, \mathcal{A}_N, \mathcal{X})$, the arithmetic average mass of $\mathcal{B}$ on dimensions $\mathcal{N}$ is

$$\rho(\mathcal{B}, \mathcal{N}) = \frac{|\mathcal{B}|}{\frac{1}{|\mathcal{N}|}\sum_{n \in \mathcal{N}} |\mathcal{B}_n|},$$

where $\mathcal{N}$ is a subset of $[N]$ and obviously $\rho \in [1.0, +\infty)$.

If block $\mathcal{B}$ *is dense in* $\mathbf{R}$*, then* $\rho(\mathcal{B}, [N]) > 1.0$.

Other density metrics listed in [32] are also effective for fraud detection. It is broadly true that all density measures are functions of the cardinalities of the dimensions and masses of $\mathcal{B}$ and $\mathbf{R}$. In $\mathbf{R}$, previous studies [12, 30–32] have focused on detecting the top-k densest blocks in terms of a density metric. In the remainder of this paper, we use the density metric ρ to illustrate our key points.

3.2 Shortcomings of Existing Approaches and Motivation

In practice, the blocks formed by fraudulent entities in $\mathbf{R}$ may be described by *hidden-densest blocks*. To illustrate hidden-densest blocks, we present the following definitions and examples.

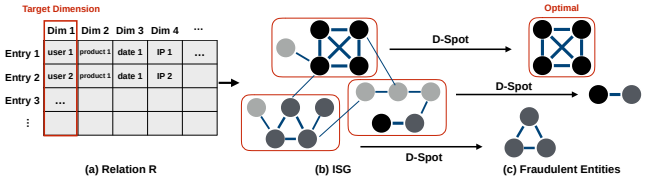


Figure 1: Workflow of ISG+D-Spot.

Definition 2. In $\mathbf{R}(A_1, \ldots, A_N, X)$, we say that $\mathcal{B}(\mathcal{A}_1, \ldots, \mathcal{A}_N, \mathcal{X})$ is the densest on a dimension A_n if $\rho(\mathcal{B}, \{n\})$ is the maximal value of all possible $\rho(\hat{\mathcal{B}}, \{n\})$, where $\hat{\mathcal{B}}$ is any possible block in $\mathbf{R}$.

Definition 3. (Hidden-Densest Block). In $\mathbf{R}(A_1, \ldots, A_N, X)$, $\mathcal{B}(\mathcal{A}_1, \ldots, \mathcal{A}_N, \mathcal{X})$ is the hidden-densest block if $\mathcal{B}$ is the densest on a small subset of $\{A_1, \ldots, A_N\}$.

Example 2 (Registration logs). *In a registration dataset with 19 features, fake accounts only exhibit conspicuous resource sharing with respect to the IP address feature.*

Example 3 (TCP dumps). *The DARPA dataset [1] has 43 features, but the block formed by malicious connections is only the densest on two features.*

Thus, catching hidden-densest blocks has a significant utility in the real world. Unfortunately, the problem is intractable using existing approaches [12, 30–32].

First, assuming that the hidden-densest block $\mathcal{B}(\mathcal{A}_1, \ldots, \mathcal{A}_N, \mathcal{X})$ is only the densest on dimension A_N, we have that

$$\rho(\mathcal{B}, [N-1]) = \frac{|\mathcal{B}|}{\frac{1}{N-1}\sum_{n \in [N-1]} |\mathcal{B}_n|} \approx \rho(\mathcal{B}, [N]) = \frac{|\mathcal{B}|}{\frac{1}{N}\sum_{n \in [N]} |\mathcal{B}_n|}$$

when N is sufficiently large. Assuming $\rho(\mathcal{B}, [N-1])$ is very low, then the methods in [12, 30–32], which try to find the block $\mathcal{B}$ that maximizes $\rho(\mathcal{B}, [N])$, have a limited ability to detect the hidden-densest block.

Second, consider a block $\mathcal{B}$ formed by fraudulent entities, in which $\mathcal{B}$ is only the densest on $\{A_2, A_3, A_5\}$, and thus $\rho(\mathcal{B}, \{2, 3, 5\})$ is maximal. However, the techniques of [12, 30–32] cannot find $\{A_2, A_3, A_5\}$ when the feature combinations are exploded.

Furthermore, in $\mathbf{R}(A_1, \ldots, A_N, X)$, consider two blocks $\mathcal{B}_1$ and $\mathcal{B}_2$, where $\mathcal{B}_1$ is the densest on A_i, $\mathcal{B}_2$ is the densest on A_j, and $\rho(\mathcal{B}_1, [N]) = \rho(\mathcal{B}_2, [N])$. Does this indicate that $\mathcal{B}_1$ and $\mathcal{B}_2$ are equally suspicious? No, absolutely not, because A_i could be the IP address feature and A_j could be a trivial feature such as the user's age, location, or gender.

[Value Sharing]. Based on the considerations above, we design our approach from a different angle. The key reason behind the formation of dense blocks is value sharing. Given $t_1 \in \mathbf{R}$, a dimension A_n, and $t_1[A_n] = a$, we can identify value sharing when $\exists t_2 \in \mathbf{R}$, $t_2 \neq t_1$, and $t_2[A_n] = a$.

Obviously, if a block $\mathcal{B}$ is dense, $\rho(\mathcal{B}, [N]) > 1.0$, then value sharing must be occurring, i.e., **value sharing results in dense blocks**.

Therefore, detecting dense blocks is equivalent to catching value sharing signals. We propose ISG based on information theory and design the D-Spot algorithm to leverage graph features, allowing us to catch fraudulent entities within dense blocks and overcome the limitations mentioned above.

4 ISG BUILDING

In this section, we present the Information Sharing Graph (ISG), which is constructed on the relation $\mathbf{R}$.

4.1 Problem Formulation

Catching fraudulent entities is equivalent to detecting a subset of distinct values in a certain dimension. Let U denote the target dimension in which a subset of distinct values form the fraudulent entities we wish to detect. In $\mathbf{R} = (A_1, ..., A_N, X)$, **we choose a dimension and set it as U, and denote the remaining $(N-1)$ dimensions as K dimensions**, $\underline{k \in [K]}$, for brevity. We build the ISG of U, i.e., the weighted-undirected graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, in which $\mathbf{V} = \{u_1, ..., u_n\}$ is the set of distinct values of U.

In Example 1, $\mathbf{R}(\underline{user}, \underline{product}, \underline{timestamp}, X)$, we set $U = \underline{user}$ if we wish to detect fraudulent user accounts. In Example 2, we set $U = \underline{account}$ if we would like to identify fake accounts. In Example 3, we set $U = \underline{connection}$ to catch malicious connections.

To specifically describe the process of value sharing, we present the two following definitions:

Definition 4. (Pairwise Value Sharing). Given $u_i, u_j \in \mathbf{V}$ and $a \in A_k$, we say that u_i and u_j share value a on A_k if $\exists t_1, t_2 \in \mathbf{R}$ such that $t_1[U] = u_i, t_2[U] = u_j$ and $t_1[A_k] = t_2[A_k] = a$.

Pairwise value sharing occurs when a distinct value is shared by multiple individual entities. Given a value sharing process in which a is shared by $\mathcal{V} \subset \mathbf{V}$, we denote this as $\frac{|\mathcal{V}|(|\mathcal{V}|-1)}{2}$ pairwise value sharing.

Definition 5. (Self-Value Sharing). Given $t_1 \in \mathbf{R}$, where $t_1[U] = u_i, u_i \in \mathbf{V}$, and $t_1[A_k] = a$, we say that u_i shares value a on A_k if $\exists t_2 \in \mathbf{R}$ and $t_2 \neq t_1$ such that $t_2[U] = u_i$ and $t_2[A_k] = a$.

Another type of value sharing occurs when the distinct value a is shared n times by an entity u_i, which can be represented by n instances of self-value sharing.

In ISG $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, for some edge $(u_i, u_j) \in \mathbf{E}$, $\mathcal{S}_{i,j}$ represents the information between u_i and u_j derived from the other K dimensions, and for some node $u_i \in \mathbf{V}$, $\mathcal{S}_i$ denotes the information of u_i calculated from the other K dimensions. From the definitions and notations defined in the previous section, Problem 1 gives a formal definition of how to build the ISG of a tensor.

Problem 1 (Building a pairwise information graph). **(1) Input:** *a relation* $\mathbf{R}$*, the target dimension* U*,* **(2) Output:** *the information sharing graph* $\mathbf{G} = (\mathbf{V}, \mathbf{E})$.

4.2 Building an ISG

Given a dimension A_k, the target dimension U, any $u_i \in \mathbf{V}$, and an entry $t_1 \in \mathbf{R}$ for which $t_1[U] = u_i$, then for each $a \in A_k$, we assume that the probability of $t_1[A_k] = a$ is $p^k(a)$.

Edge Construction. Based on information theory [28], the self-information of the event that u_i and u_j share a is:

$$\mathcal{I}^k_{i,j}(a) = \log(\frac{1}{p^k(a)})^2. \tag{1}$$

To compute the pairwise value sharing between u_i and u_j across all K dimensions, we propose the metric $\mathcal{S}$-score as the edge weight of ISG:

$$\mathcal{S}_{i,j} = \sum_{k=1}^{K} \sum_{a \in H_k(u_i, u_j)} \mathcal{I}^k_{i,j}(a), \tag{2}$$

where $H_k(u_i, u_j)$ is the set of all values shared by u_i and u_j on A_k. Note that $\mathcal{S}_{i,j} = 0.0$ if $\bigcup_{k=1}^{K} H_k(u_i, u_j) = \emptyset$.

Intuitively, if u_i and u_j *do not* have any shared values, which is to be expected in normal circumstances, we have zero information. Otherwise, we obtain some information. Thus, the higher the value of $\mathcal{S}_{i,j}$ is, the more similar u_i is to u_j. In practice, the $\mathcal{S}$-score has a large variance. For example, fraud user pairs sharing an IP subnet and device ID will have a high $\mathcal{S}$-score, whereas normal users are unlikely to share these values with anyone, and will thus have an $\mathcal{S}$-score close to zero. Additionally, the information we obtain for u_i and u_j sharing the value a is related to the overall probability of that value. For example, it would be much less surprising if they both follow Donald Trump on Twitter than if they both follow a relatively unknown user.

Node Setting. For a node $u_i \in \mathbf{V}$, let $\mathcal{B}(a, A_k, u_i, U)$ be the set $\{t \in \mathbf{R} : (t[A_k] = a) \wedge (t[U] = u_i)\}$. When $|\mathcal{B}(a, A_k, u_i, U)| \geq 2$. The information of forming $\mathcal{B}(a, A_k, u_i, U)$ is:

$$\mathcal{I}^k_i(a) = \log(\frac{1}{p^k(a)})^{|\mathcal{B}(a, A_k, u_i, U)|}. \tag{3}$$

We now define $\mathcal{S}_i$ to compute the self-value sharing for u_i across all K dimensions:

$$\mathcal{S}_i = \sum_{k=1}^{K} \sum_{\mathcal{B} \in H_k(u_i)} \mathcal{I}^k_i(a), \tag{4}$$

where $H_k(u_i)$ is the set $\{\mathcal{B}(a, A_k, u_i, U), \forall a \in \mathbf{R}_k\}$ and $\mathbf{R}_k$ is the set of distinct values of A_k. Note that $\mathcal{S}_i = 0.0$ if $\bigcup_{k=1}^{K} H_k(u_i) = \emptyset$.

In effect, self-value sharing occurs in certain fraud cases. For instance, a fraudulent user may create several fake reviews for a product/restaurant on Amazon/Yelp [25] over a few days. In terms of network attacks [19], a malicious TCP connection tends to attack a server multiple times.

Determining [$p^k(a)$]. We can extend the $\mathcal{S}$-score to accommodate different data types and distributions.

It is difficult to determine $p^k(a)$, as we do not always know the distribution of A_k. In this case, for dimensions that are attribute features, we assume a uniform distribution and simply set

$$p^k(a) = 1/|\mathbf{R}_k|. \tag{5}$$

This approximation works well for many fraud-related properties such as IP subnets and device IDs, which usually follow a Poisson distribution [12].

However, the uniform assumption works poorly for low-entropy distributions, such as the long-tail distribution, which is common in dimensions such as items purchased or users followed. Low entropy implies that many users behave similarly anyway, independent of frauds. Intuitively for such distributions, there is no surprise in following a celebrity (head of the distribution), but considerable information if they both follow someone at the tail. For example, 20% of users correspond to more than 80% of the "follows" in online social networks. The dense subgraphs between celebrities and their fans are very unlikely to be fraudulent. If feature A_k has a long-tail distribution, its entropy is very low. For example, the entropy of the uniform distribution over 50 values is 3.91, but the entropy of a

long-tail distribution with 90% of probabilities centered around one value is only 0.71. Therefore, we set $p^k(a)$ based on the empirical distribution as

$$p^k(a) = |\mathcal{B}(a, A_k)|/|\mathbf{R}|, \tag{6}$$

when the values in A_k have low entropy. We also provide an interface so that users can define their own $p^k(a)$ function.

Optimization of ISG Construction. In theory, a graph with $|\mathbf{V}|$ nodes has $O(|\mathbf{V}|^2)$ edges. Naively, therefore, it takes $O(|\mathbf{V}|^2)$ time for graph initialization and traversal.

To reduce the complexity of building the ISG, we use the *key-value* approach. The *key* corresponds to a value a on A_k and the *value* represents the block $\mathcal{B}(a, A_k)$. Let $\mathcal{V} \subseteq \mathbf{V}$ denote the entities that occur in $\mathcal{B}(a, A_k)$. As each pair $(u_i, u_j) \in \mathcal{V}$ shares a, we increase the value of $\mathcal{S}_{i,j}$ by $\mathcal{I}^k_{i,j}(a)$. Additionally, for each $u_i \in \mathcal{V}$, there exists some $\mathcal{B}(a, A_k, u_i, U) \subseteq \mathcal{B}(a, A_k)$. Thus, we increase the value of $\mathcal{S}_i$ by $\mathcal{I}^k_i(a)$ if $|\mathcal{B}(a, A_k, u_i, U)| \geq 2$.

To build the ISG, we compute all *key-value* pairs across K dimensions by traversing $\mathbf{R}$ in parallel. Thus, it takes $O(K|\mathbf{R}| + |\mathbf{E}|)$ time to build the graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$. Note that we only retain positive $\mathcal{S}_{i,j}$ and $\mathcal{S}_i$. In practice, $\mathbf{G}$ is usually sparse, which is discussed in Section 6.1.

4.3 Key Observations on ISG

Given a relation $\mathbf{R} = (A_1, ..., A_N, X)$ in which we set $U = A_N$, we construct the ISG of U, $\mathbf{G} = (\mathbf{V}, \mathbf{E})$. Assuming there is a **fraudulent block** $\mathcal{B}$ in $\mathbf{R}$, $\mathcal{B} = (\mathcal{A}_1, ..., \mathcal{A}_N, \mathcal{X})$ is transformed into a subgraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ in $\mathbf{G}$, where $\mathcal{V}$ is the set of distinct values of $\mathcal{A}_N$ and an edge $\mathcal{S}_{i,j} \in \mathcal{E}$ denotes the information between u_i and u_j calculated from the other K dimensions. Then, $\mathcal{V}$ is the fraud group comprised of fraudulent entities that we wish to detect.

We summarize three critical observations of $\mathcal{G}$ that directly lead to the algorithms presented in Section 5.2. Given $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we define the **edge density** of $\mathcal{G}$ as

$$\rho_{edge}(\mathcal{G}) = \frac{|\mathcal{E}|}{|\mathcal{V}|(|\mathcal{V}| - 1)}$$

1) The value of $\mathcal{S}_{i,j}$ or $\mathcal{S}_i$ is unusually high. Value sharing may happen frequently, but sharing across certain features, even certain values, is more suspicious than others. Intuitively, it might be suspicious if two users share an IP address or follow the same random "nobody" on Twitter. However, it is not so suspicious if they have a common gender, city, or follow the same celebrity. In other words, certain value sharing is likely to be fraudulent because the probability of sharing across a particular dimension, or at a certain value, is quite low. Thus, the information value is high, which is accurately captured by $\mathcal{S}_{i,j}$ and $\mathcal{S}_i$.

2) $|\mathcal{V}|$ is usually large. Fraudsters perform the same actions many times to achieve economies of scale. Thus, we expect to find multiple pairwise complicities among fraudulent accounts. A number of studies have found that large cluster sizes are a crucial indicator of fraud [5, 38]. Intuitively, while it is natural for a few family members to share an IP address, it is highly suspicious when dozens of users share one.

3) The closer $\rho_{edge}(\mathcal{G})$ is to 1.0, the more suspicious $\mathcal{G}$ is. Fraudsters usually operate a number of accounts for the same job, and thus it is likely that users manipulated by the same fraudster will share the same set of values. Thus, the $\mathcal{G}$ formed by the fraud group will be well-connected.

Appearance of legitimate entities on ISG. In $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, given some u_i that we assume to be **legitimate**, let $h(u_i)$ denote the set of its neighbor nodes. We have two findings. (1) For u_i, $\mathcal{S}_i + \sum_{u_j \in \mathbf{V}} \mathcal{S}_{i,j} \to 0$ because u_i is unlikely to share values with others. Even if exists, the shared values should have a high probability (see observation 1) and therefore small $\mathcal{S}_{i,j}$. (2) The subgraph $\mathcal{G}$ induced by $h(u_i)$ is typically not well-connected, as resource sharing is uncommon in the real world. If $\mathcal{G}$ is well-connected, $|h(u_i)|$ is quite small compared with the fraud group size (see observation 2).

In summary, the techniques described in [12, 20, 30–32] work directly on the tensor, indicating that they consider value sharing on each dimension, and even certain values, as equivalent. In contrast, ISG assigns each instance of value sharing a theoretical weight based on the edges and nodes of the ISG, which is more effective for identifying the (hidden-) densest blocks (comparison in Sec.6.2).

5 SPOTTING FRAUD

Based on the observations in Section 4.3, we now describe our method for finding objective subgraphs in $\mathbf{G}$. This section is divided into two parts: first, we define a density metric $\mathcal{F}_\mathcal{G}$, and then we illustrate the proposed D-Spot algorithm.

5.1 Density Metric and Problem Definition

To find the objective $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we define a density metric $\mathcal{F}_\mathcal{G}$ as [6, 11]:

$$\mathcal{F}_\mathcal{G} = \frac{\sum_{(u_i,u_j)\in\mathcal{E}} \mathcal{S}_{i,j} + \sum_{u_i \in \mathcal{V}} \mathcal{S}_i}{|\mathcal{V}|}. \tag{7}$$

The form of $\mathcal{F}_\mathcal{G}$ satisfies the three key observations of $\mathcal{G}$ in Section 4.3.

(1) Keeping $|V|$ fixed, we have that $\sum_{(u_i,u_j)\in\mathcal{E}} \mathcal{S}_{i,j} + \sum_{u_i \in \mathcal{V}} \mathcal{S}_i \uparrow \Rightarrow \mathcal{F}_\mathcal{G} \uparrow$.
(2) Keeping $\mathcal{S}_{i,j}$, $\mathcal{S}_i$, and $\rho_{edge}(\mathcal{G})$ fixed, we have that $|V| \uparrow \Rightarrow \mathcal{F}_\mathcal{G} \uparrow$.
(3) Keeping $\mathcal{S}_{i,j}$, $\mathcal{S}_i$, and $|\mathcal{V}|$ fixed, we have that $\rho_{edge}(\mathcal{G}) \uparrow \Rightarrow \mathcal{F}_\mathcal{G} \uparrow$.

Thus, our subgraph-detection problem can be defined as follows:
Problem 2 (Detecting dense subgraphs). **(1) Input:** *the information sharing graph* $\mathbf{G} = (\mathbf{V}, \mathbf{E})$. **(2) Find:** *multiple subgraphs of* $\mathbf{G}$ *that maximize* $\mathcal{F}$.

5.2 D-Spot (Algorithm 1-3)

In real-world datasets, there are usually numerous fraud groups forming multiple dense subgraphs. Based on the considerations described above, we propose D-Spot (Algorithms 1–3). Compared with other well-known algorithms for finding the densest subgraph [6, 11], D-Spot has two differences:

(1) D-Spot can detect multiple densest subgraphs simultaneously. D-Spot first partitions the graph, and then detects a single densest subgraph in each partition.
(2) D-Spot is faster. First, instead of removing nodes one by one, D-Spot removes a set of nodes at once, reducing the number of iterations. Second, it detects the single densest subgraph

in partition $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, rather than in graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $|\mathcal{V}| << |\mathbf{V}|$ and $|\mathcal{E}| << |\mathbf{E}|$.

D-Spot consists of two main steps: (1) Given $\mathbf{G}$, divide $\mathbf{G}$ into multiple partitions (Algorithm 1); (2) In each $\mathcal{G}$, find a single dense subgraph (Algorithms 2 and 3).

Algorithm 1: graph partitioning. Let $\hat{\mathcal{G}}$ be a dense subgraph formed by a fraud group that we are about to detect. In $\mathbf{G}$, there are usually multiple $\hat{\mathcal{G}}$s, where each $\hat{\mathcal{G}}$ should be independent or connected with others by small values of $\mathcal{S}_{i,j}$. Thus we let $\mathcal{G}$s be the connected components of $\mathbf{G}$ as partitions (line 6). For each $\mathcal{G} \in \mathcal{G}s$, we run Algorithms 2 and 3 (lines 7–9) to find $\hat{\mathcal{G}}$. Finally, Algorithm 1 returns multiple dense subgraphs $\hat{\mathcal{G}}$s (line 10). Note that there is a guarantee that $\hat{\mathcal{G}}$s must contain the $\hat{\mathcal{G}}$ that is at least 1/2 of the optimum of $\mathbf{G}$ in terms of $\mathcal{F}$ (proof in Sec.6.3).

Information pruning (recommended). As mentioned before, Fraud entities usually have surprising similarities that are quantified by $\mathcal{S}_{i,j}$. We want to delete edges with regular weights and thus we provide a threshold for removing edges:

$$\theta = \frac{\sum_{(u_i,u_j)\in\mathbf{E}} \mathcal{S}_{i,j}}{|\mathbf{V}|(|\mathbf{V}|-1)} \tag{8}$$

It is easy to see that θ (conservative) is the average information of all possible pairs (u_i, u_j). Thus, we iterate through all edges in $\mathbf{G}$, and remove those for which $\mathcal{S}_{i,j} < \theta$ (lines 3–5). In all experiments of this paper, we used θ and get the expected conclusion that the performance of D-Spot using θ is hardly different from no pruning but using θ is able to significantly decrease the running cost of D-Spot.

Algorithm 1 *find multiple dense subgraphs* in $\mathbf{G}$

Require: $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, θ (Eq.8), $w()$ (Eq. 9)
Ensure: $\hat{\mathcal{G}}$s
1: $\hat{\mathcal{G}}$s $\leftarrow \emptyset$
2: **if** needed **then**
3: **for** each $\mathcal{S}_{i,j} \in \mathbf{E}$ **do**
4: **if** $\mathcal{S}_{i,j} < \theta$ **then**
5: remove $\mathcal{S}_{i,j}$
6: $\mathcal{G}$s $\leftarrow$ connected components of $\mathbf{G}$
7: **for** each $\mathcal{G} \in \mathcal{G}$s **do**
8: $\hat{\mathcal{G}} \leftarrow$ *find a dense subgraph* $(\mathcal{G}, w())$
9: $\hat{\mathcal{G}}$s $\leftarrow \hat{\mathcal{G}}$s $\cup \{\hat{\mathcal{G}}\}$
10: **return** $\hat{\mathcal{G}}$s

Algorithms 2 and 3: find a dense subgraph. Initially, let $\mathcal{V}_c$ be a copy of $\mathcal{V}$. In each iteration (lines 5–14), we delete *a set of nodes* (R, line 6) from $\mathcal{V}_c$ until $\mathcal{V}_c$ is empty. Of all the $\mathcal{V}_c$ constructed during the execution of the algorithm, that maximizing $\mathcal{F}$ ($\hat{R}$, line 15) is returned as the output of the algorithm.

Given a subgraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for some $u_i \in \mathcal{V}$, we define $w(u_i, \mathcal{G})$ as

$$w(u_i, \mathcal{G}) = \sum_{(u_j\in\mathcal{V})\wedge((u_i,u_j)\in\mathcal{E})} \mathcal{S}_{i,j} + \mathcal{S}_i. \tag{9}$$

Lines 1–4 initialize the parameters used in the algorithm. *Dict*2 records the w value of each node. *Dict*1 records the order in which the nodes are deleted (line 10), which allows us to determine the value of $\hat{R}$ that maximizes $\mathcal{F}$. Line 6 determines which R are deleted in each iteration. R is confirmed by $\{u \in \mathcal{V} : w(u, \mathcal{G}) \le \overline{w}\}$ (line 6), where the average $\overline{w}$ is given by:

$$\begin{aligned}\overline{w} &= \frac{\sum_{u\in\mathcal{V}} w(u,\mathcal{G})}{|\mathcal{V}|} \\ &= \frac{2\sum_{(u_i,u_j)\in\mathcal{E}} \mathcal{S}_{i,j} + \sum_{u_i\in\mathcal{V}} \mathcal{S}_i}{|\mathcal{V}|} \le 2\mathcal{F}_{\mathcal{G}},\end{aligned} \tag{10}$$

because each edge $\mathcal{S}_{i,j}$ is counted twice in $\sum_{u\in\mathcal{V}} w(u, \mathcal{G})$. In lines 7–14, the nodes in R are removed from $\mathcal{V}_c$ in each iteration (In contrast, [11] recomputes all nodes and finds those with the minimal w after deleting a node). As removing a subset of R may result in a higher value of $\mathcal{F}$, D-Spot records each change of $\mathcal{F}$, as if the nodes were removed one by one (lines 8–14). Algorithm 3 describes how the edges are updated after a node is removed, requiring a total of $|\mathcal{E}|$ updates. Finally, Algorithm 2 returns the subgraph $\hat{\mathcal{G}}$ induced by $\hat{R}$, the set of nodes achieving $\mathcal{F}^{max}$, according to *top* and *Dict*1 (line 15).

Summary. As R contains at least one node, the worst-case time complexity of Algorithm 2 is $O(|\mathcal{V}|^2 + |\mathcal{E}|)$. In practice, the worst case is too pessimistic. In line 6, R usually contains plenty of nodes, significantly reducing the number of scans of $\mathcal{V}_c$ (see Section 7.2).

Algorithm 2 *find a dense subgraph*

Require: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, $w()$(Eq. 9)
Ensure: $\hat{\mathcal{G}}$
1: $\mathcal{V}_c \leftarrow copy(\mathcal{V})$
2: $\mathcal{S}_{sum} \leftarrow \sum_{(u_i,u_j)\in\mathcal{E}} \mathcal{S}_{i,j} + \sum_{u_i\in\mathcal{V}} \mathcal{S}_i$
3: $\forall u \in \mathcal{V}, Dict1[u] \leftarrow 0, Dict2[u] = w(u, \mathcal{G})$
4: $index \leftarrow 0,\ \mathcal{F}^{max} \leftarrow \frac{\mathcal{S}_{sum}}{|\mathcal{V}_c|},\ top \leftarrow 0$
5: **while** $\mathcal{V}_c \neq \emptyset$ **do**
6: $R \leftarrow \{u \in \mathcal{V}_c : Dict2[u] \le \frac{2\sum_{(u_i,u_j)\in\mathcal{E}} \mathcal{S}_{i,j} + \sum_{u_i\in\mathcal{V}_c} \mathcal{S}_i}{|\mathcal{V}_c|}\}$ (Eq. 10)
7: sort R in increasing order of $Dict2[u]$
8: **for** each $u \in R$ **do**
9: $\mathcal{V}_c \leftarrow \mathcal{V}_c - u,\ \mathcal{S}_{sum} \leftarrow \mathcal{S}_{sum} - Dict2[u]$
10: $index \leftarrow index + 1, Dict1[u] \leftarrow index$
11: $\mathcal{F} = \frac{\mathcal{S}_{sum}}{|\mathcal{V}_c|}$
12: **if** $\mathcal{F} > \mathcal{F}^{max}$ **then**
13: $\mathcal{F}^{max} \leftarrow \mathcal{F},\ top \leftarrow index$
14: $Dict2 \leftarrow$ *update edges* $(u, \mathcal{V}_c, Dict2, \mathcal{G})$
15: $\hat{R} \leftarrow \{u \in \mathcal{V} : Dict1[u] > top\}$
16: **return** $\hat{\mathcal{G}}$ (the subgraph induced by $\hat{R}$)

Algorithm 3 *update edges*

Require: $u_i, \mathcal{V}_c, Dict2, \mathcal{G} = (\mathcal{V}, \mathcal{E})$
Ensure: *Dict*2
1: **for** each $u_j \in \mathcal{V}_c$ **do**
2: **if** $(u_i, u_j) \in \mathcal{E}$ **then**
3: $Dict2[u_j] \leftarrow Dict2[u_j] - \mathcal{S}_{i,j}$
4: remove (u_i, u_j) from $\mathcal{E}$
5: **return** *Dict*2

6 ANALYSIS

6.1 Complexity.

In the graph initialization stage, it takes $O(K|\mathbf{R}| + |\mathbf{E}|)$ time to build $\mathbf{G}$ based on the optimization in Section 4.2. In D-Spot, the cost of partitioning $\mathbf{G}$ is $O(|\mathbf{E}|)$, and detecting a dense block in a partition $\mathcal{G}$ requires $O(|\mathcal{E}|+|\mathcal{V}|^2)$ operations, where $|\mathcal{E}| << |\mathbf{E}|$, $|\mathcal{V}| << |\mathbf{V}|$. Thus, the complexity of ISG+D-Spot is linear with respect to $|\mathbf{E}|$.

In the worst case, admittedly, $|\mathbf{E}| = |\mathbf{V}|^2$ when there is some dimension A_k in which $|\mathbf{R}_k| = 1$. However, that is too pessimistic. In the target fraud attacks, fraud groups typically exhibit strong value sharing while legit entities should not. Hence, we expect $\mathbf{G}$ to be sparse because the u_i only have positive edges with a small subset of $\mathbf{V}$. We constructed a version of $\mathbf{G}$ using several real-world datasets (see Fig. 3), and the edge densities were all less than 0.06.

6.2 Effectiveness of ISG+D-Spot

Theorem 6. *(Spotting the Hidden-Densest Block). Given a dense block $\mathcal{B}(\mathcal{A}_1, ..., \mathcal{A}_N, \mathcal{X})$ in which the target dimension $U = A_N$ and $\mathcal{V}$ denotes the set of distinct values of $\mathcal{A}_N$, a shared value a exists in $\mathcal{B}$ such that, $\forall u \in \mathcal{V}$, $\exists t \in \mathcal{B}$ satisfying $(t[U] = u) \wedge (t[A_k] = a)$. Then, $\mathcal{B}$ must form a dense subgraph $\mathcal{G}$ in $\mathbf{G}$.*

Proof. Using the optimization algorithm in Section 4.2, we build $\mathbf{G}$ by scanning all values in $\mathbf{R}$ once. Hence, the block $\mathcal{B}(a, A_k)$ must be found. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the subgraph induced by $\mathcal{V}$ in $\mathbf{G}$. Then, $\forall(u_i, u_j) \in \mathcal{E}$, the edge $\mathcal{S}_{i,j} \geq \mathcal{I}^k_{i,j}(a)$. Hence, $\rho_{edge}(\mathcal{G}) = 1.0$ and $\mathcal{F}_{\mathcal{G}} = \frac{\sum_{\forall(u_i,u_j)\in\mathcal{E}} \mathcal{S}_{i,j}+\sum_{u_i\in\mathcal{V}} \mathcal{S}_i}{|\mathcal{V}|} \geq \frac{|\mathcal{V}|(|\mathcal{V}|-1)\mathcal{I}^k_{i,j}(a)}{|\mathcal{V}|} = (|\mathcal{V}| - 1)\mathcal{I}^k_{i,j}(a)$. □

Observation. (Effectiveness of ISG+D-Spot) Consider a hidden-densest block $\mathcal{B}(\mathcal{A}_1, ..., \mathcal{A}_{N-1}, \mathcal{A}_N, \mathcal{X})$ of size $|\mathcal{X}| \times ... \times |\mathcal{X}| \times 1$ and $|\mathcal{B}| = |\mathcal{X}|$, i.e., $\mathcal{B}$ is the densest on A_N by sharing the value a. Then, assuming the target dimension $U = A_1$ and the fraudulent entities $\mathcal{V}$ are distinct values of $\mathcal{A}_1$, ISG+D-Spot captures $\mathcal{V}$ more accurately than other algorithms based on tensors (denoted as Tensor+Other Algorithms).

Proof. Let us consider a non-dense block $\hat{\mathcal{B}}(\hat{\mathcal{A}}_1, ..., \hat{\mathcal{A}}_N, \mathcal{X})$ of size $|\mathcal{X}| \times ... \times |\mathcal{X}|$, $|\hat{\mathcal{B}}| = |\mathcal{X}|$, and let $\hat{\mathcal{V}}$ denote the distinct values of $\hat{\mathcal{A}}_1$. Denoting legitimate entities as $\hat{\mathcal{V}}$ and fraudulent entities as $\mathcal{V}$, we now discuss the difference between ISG+D-Spot and Tensor+Other Algorithms.

[*Working on the tensor*]. On $\mathbf{R}$, $\hat{\mathcal{B}}$ is not dense and thus $\rho(\hat{\mathcal{B}}, [N]) = 1$. For $\mathcal{B}$, because $\{|\mathcal{B}_1|, ..., |\mathcal{B}_{N-1}|, |\mathcal{B}_N|\} = \{|\mathcal{X}|, ..., |\mathcal{X}|, 1\}$, we have $\rho(\mathcal{B}, [N]) = \frac{|\mathcal{B}|}{\frac{1}{N}\sum_{n\in[N]}|\mathcal{B}_n|} \approx 1$ for sufficiently large N.

[*Working on ISG*]. On $\mathbf{G}$, let $\hat{\mathcal{G}}$ denote the subgraph induced by $\hat{\mathcal{V}}$ and $\mathcal{G}$ denote the subgraph formed by $\mathcal{V}$. We know that $\hat{\mathcal{G}}$, $\mathcal{F}_{\hat{\mathcal{G}}} = 0$, because $\hat{\mathcal{B}}$ does not have any shared values. For $\mathcal{G}$, $\mathcal{F}_{\mathcal{G}} = (\mathcal{V} - 1)\mathcal{I}^k_{i,j}(a)$ according to Theorem 6.

[*Other Algorithms*]. M-Zoom[30] and D-Cube [32] are known to find blocks that are at least $1/N$ of the optimum in term of ρ on $\mathbf{R}$ ($\frac{1}{N}$-Approximation guarantee).

[*D-Spot*]. In Section 6.3, we will show that the subgraph detected by D-Spot is at least 1/2 of the optimum in term of $\mathcal{F}_{\mathcal{G}}$ on ISG ($\frac{1}{2}$-Approximation guarantee).

In summary, Tensor+Other Algorithms vs. ISG+D-Spot corresponds to:

$$\left(\rho(\hat{\mathcal{B}}, [N]) = 1 \mid \rho(\mathcal{B}, [N]) \approx 1 \ + \ (\frac{1}{N}\text{-Approximation})\right)$$
$$vs. \left(\mathcal{F}_{\hat{\mathcal{G}}} = 0 \mid \mathcal{F}_{\mathcal{G}} = (\mathcal{V} - 1)\mathcal{I}^k_{i,j}(a) \ + \ (\frac{1}{2}\text{-Approximation})\right)$$

Therefore, ISG+D-Spot catches fraudulent entities within hidden-densest blocks more accurately than Tensor+Other Algorithms. □

From the observation, ISG+D-Spot can effectively detect hidden-densest blocks. Similarly, when $\mathcal{B}$ becomes denser, the $\mathcal{G}$ formed by $\mathcal{B}$ will also be much denser, and thus ISG+D-Spot will be more accurate in detecting the densest block.

6.3 Accuracy Guarantee of D-Spot

For brevity, we use $[\mathcal{V}]$ to denote a subgraph induced by the set of nodes $\mathcal{V}$.

Theorem 7. *(Algorithm 1 Guarantee). Given $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, let $\mathcal{G}s = \{\mathcal{G}_1, ..., \mathcal{G}_n\}$ denote the connected components of $\mathbf{G}$. Let $\mathcal{F}_i^{opt}$ denote the optimal $\mathcal{F}$ on $\mathcal{G}_i$, i.e., $\nexists \mathcal{G}' \subseteq \mathcal{G}_i$ satisfying $\mathcal{F}_{\mathcal{G}'} > \mathcal{F}_i^{opt}$. Then, if $\mathcal{F}_n^{opt}$ is the maximal value of $\{\mathcal{F}_1^{opt}, ..., \mathcal{F}_n^{opt}\}$, $\mathcal{F}_n^{opt}$ must be the optimum in terms of $\mathcal{F}$ on $\mathbf{G}$.*

Proof. Given any two sets of nodes $\mathcal{V}_1$ and $\mathcal{V}_2$ and assuming there are no edges connecting $\mathcal{V}_1$ and $\mathcal{V}_2$, we assume that $\mathcal{F}_{[\mathcal{V}_1]} > \mathcal{F}_{[\mathcal{V}_2]} \Rightarrow \frac{c_1}{|\mathcal{V}_1|} > \frac{c_2}{|\mathcal{V}_2|} \Rightarrow c_1|\mathcal{V}_2| > c_2|\mathcal{V}_1|$. Then,

$$\mathcal{F}_{[\mathcal{V}_1]} - \mathcal{F}_{[\mathcal{V}_1\cup\mathcal{V}_2]} \Rightarrow \frac{c_1}{|\mathcal{V}_1|} - \frac{c_1 + c_2}{|\mathcal{V}_1| + |\mathcal{V}_2|}$$
$$\Rightarrow \frac{c_1|\mathcal{V}_2| - c_2|\mathcal{V}_1|}{|\mathcal{V}_1|(|\mathcal{V}_1| + |\mathcal{V}_2|)} > \frac{c_2|\mathcal{V}_1| - c_2|\mathcal{V}_1|}{|\mathcal{V}_1|(|\mathcal{V}_1| + |\mathcal{V}_2|)} = 0.$$

Thus, for any $\mathcal{V}_1$ and $\mathcal{V}_2$ that are not connected by any edges, it follows that $\mathcal{F}_{[\mathcal{V}_1\cup\mathcal{V}_2]} \leq \max(\mathcal{F}_{[\mathcal{V}_1]}, \mathcal{F}_{[\mathcal{V}_2]})$ (Conclusion 1).

In $\mathcal{G}_n = (\mathcal{V}_n, \mathcal{E}_n)$, we use $\hat{\mathcal{V}}$ to denote the set of nodes satisfying $\hat{\mathcal{V}} \subseteq \mathcal{V}_n$ and $\mathcal{F}_{[\hat{\mathcal{V}}]} = \mathcal{F}_n^{opt}$. Let $\mathcal{V}'$ be a set of nodes satisfying $\mathcal{V}' \subset \mathbf{V}$ and $\mathcal{V}' \cap \hat{\mathcal{V}} = \emptyset$. Now, let us consider two conditions.

First, if $\mathcal{V}' \subset \mathcal{V}_n$, then $\mathcal{F}_{[\mathcal{V}']} \leq \mathcal{F}_{[\hat{\mathcal{V}}]}$ and $\mathcal{F}_{[\mathcal{V}'\cup\hat{\mathcal{V}}]} \leq \mathcal{F}_{[\hat{\mathcal{V}}]}$ because $\mathcal{F}_{[\hat{\mathcal{V}}]}$ is the optimum on $\mathcal{G}_n$.

Second, if $\mathcal{V}' \cap \mathcal{V}_n = \emptyset$, then $\mathcal{F}_{[\mathcal{V}']} \leq \mathcal{F}_{[\hat{\mathcal{V}}]}$ and $\mathcal{F}_{[\mathcal{V}'\cup\hat{\mathcal{V}}]} \leq \mathcal{F}_{[\hat{\mathcal{V}}]}$ by Conclusion 1 and because $\mathcal{F}_{[\hat{\mathcal{V}}]}$ is the maximum of $\{\mathcal{F}_1^{opt}, ..., \mathcal{F}_n^{opt}\}$.

If $\mathcal{V}' \cap \mathcal{V}_n \neq \emptyset$, then $\mathcal{V}'$ can be divided into two parts conforming with the two conditions stated above.

Therefore, $\nexists \mathcal{V}' \subset \mathbf{V}$ satisfies $\mathcal{F}_{[\mathcal{V}']} > \mathcal{F}_{[\hat{\mathcal{V}}]}$ or $\mathcal{F}_{[\mathcal{V}'\cup\hat{\mathcal{V}}]} > \mathcal{F}_{[\hat{\mathcal{V}}]}$. We can conclude that $\mathcal{F}_n^{opt} = \mathcal{F}_{[\hat{\mathcal{V}}]}$ must be the optimum in terms of $\mathcal{F}$ on $\mathbf{G}$. □

Theorem 8. *(Algorithm 2 Guarantee). Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, let Q^* be a subset of nodes maximizing $\mathcal{F}_{[Q^*]}$ in $\mathcal{G}$. Let $[Q]$ be the subgraph returned by Algorithm 2 with $\mathcal{F}_{[Q]}$. Then, $\mathcal{F}_{[Q]} \geq \frac{1}{2}\mathcal{F}_{[Q^*]}$.*

Proof. Consider the optimal set Q^*. We know that, $\forall u \in Q^*$, $w(u, [Q^*]) \geq \mathcal{F}_{[Q^*]}$, because if we remove a node u for which

$w(u, [Q^*]) < \mathcal{F}_{[Q^*]}$,

$$\mathcal{F}' = \frac{|Q^*|\mathcal{F}_{[Q^*]} - w(u, Q^*)}{|Q^*| - 1} > \frac{|Q^*|\mathcal{F}_{Q^*} - \mathcal{F}_{Q^*}}{|Q^*| - 1} = \mathcal{F}_{[Q^*]},$$

which contradicts the definition of Q^*.

Denote the first node that Algorithm 2 removes from Q^* as u_i, $u_i \in R$, and denote the node set before Algorithm 2 starts removing R as Q'. Because $Q^* \subseteq Q'$, we have $w(u_i, [Q^*]) \leq w(u_i, [Q'])$. According to Algorithm 2 (line 6), $w(u_i, [Q']) \leq 2\mathcal{F}_{[Q']}$ (Eq. 10). Additionally, Algorithm 2 returns the best solution when deleting nodes one by one, and so $\mathcal{F}_{[Q]} \geq \mathcal{F}_{[Q']}$. We conclude that

$$\mathcal{F}_{[Q]} \geq \mathcal{F}_{[Q']} \geq \frac{w(u_i, [Q'])}{2} \geq \frac{w(u_i, [Q^*])}{2} \geq \frac{\mathcal{F}_{Q^*}}{2}.$$

□

In summary, let $\{\mathcal{G}_1, ..., \mathcal{G}_n\}$ be the subgraphs returned by D-Spot, and $\{\mathcal{F}_{\mathcal{G}_1}, ..., \mathcal{F}_{\mathcal{G}_n}\}$ be the corresponding scores. Then, based on Theorems 7 and 8, $\mathcal{F}^{max} = max(\mathcal{F}_{\mathcal{G}_1}, ..., \mathcal{F}_{\mathcal{G}_n})$ is at least 1/2 of the optimum in terms of $\mathcal{F}$ on $\mathbf{G}$ (1/2-Approximation guarantee).

7 EVALUATION

A series of evaluation experiments were conducted under the following conditions:

Implementation. We implemented ISG+D-Spot in Python, and conducted all experiments on a server with two 2.20 GHz Intel(R) CPUs and 64 GB memory.

Baselines. We selected several state-of-the-art dense-block detection methods (M-Zoom [30], M-Biz [31], and D-Cube [32]) as the baselines (using open source code). To obtain optimal performance, we run three different density metrics from [32] for each baseline: ρ **(ari)**, Geometric Average Mass **(geo)**, and Suspiciousness **(sus)**.

Suspiciousness Score Setting. For the baselines, we considered a detected block $\mathcal{B}(\mathcal{A}_1, ..., \mathcal{A}_N, \mathcal{X})$ and let $\theta = \rho(\mathcal{B}, N)$. For any unique value a within $\mathcal{B}$, we then set the suspiciousness score of a to θ. If a occurred in multiple detected dense blocks, we chose the one with the maximal value of θ. Regarding ISG+D-Spot, given a detected subgraph $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}})$, for a unique value $a \in \hat{\mathcal{V}}$, we set the suspiciousness score of a to $w(a, \hat{\mathcal{G}})$ (Eq. 9). Finally, we evaluated the ranking of the suspiciousness scores of unique values using the standard area under the ROC curve (AUC) metric.

7.1 Datasets

Table 3 summarizes the datasets used in the experiments.

Synthetic is a series of datasets we synthesized using the same method as in [12]. First, we generated random seven-dimensional relations $\mathbf{R}(A_1, ..., A_7, X)$, in which $|\mathbf{R}| = 10000$ and the size of $\mathbf{R}$ is $1000 \times 500 \times ... \times 500$. In $\mathbf{R}$, we assume that A_1 corresponds to users and the other six dimensions are features. To specifically check the detection performance of the hidden-densest block using each method, we injected a dense block $\mathcal{B}(\mathcal{A}_1, ..., \mathcal{A}_7, \mathcal{X})$ into $\mathbf{R}$ five separate times, with each injection assigned a different configuration to generate five datasets. For $\mathcal{B}$, $|\mathcal{B}_1| = 50$ and $|\mathcal{B}| = 500$. We introduce the parameter λ, which denotes the number of dimensions on which $\mathcal{B}$ is the densest. For example, when $\lambda = 1$, the size of $\mathcal{B}$ is $50\times\mathbf{12}\times25... \times 25$; when $\lambda = 5$, the size of $\mathcal{B}$ is $50\times\mathbf{12}\times...\times\mathbf{12}\times25$. Obviously, $\rho(\mathcal{B}, 7) > \rho(\mathbf{R}, 7)$ and $\mathcal{B}$ is the hidden-densest block when λ is small. Finally, we labeled the users within $\mathcal{B}$ as "fraud".

Amazon [15]. AmaOffice, AmaBaby, and AmaTools are three collections of reviews about office products, baby-related products, and tool products, respectively, on Amazon. They can be modeled using the relation $\mathbf{R}(user, product, timestamp, X)$. For each entry $t \in \mathbf{R}$, $t = (u, p, t, x)$ indicates a review x that user u reviewed product p at time t. According to the specific cases of fraud discovered by previous studies [12, 38], fraudulent groups usually exhibit suspicious synchronized behavior in social networks. For instance, a large group of users may surprisingly review the same group of products over a short period. Thus, we use a similar method as in [12, 30, 32, 38]. We use a dense block $\mathcal{B}$ to represent the synchronized behavior, where $\mathcal{B}(user, product, timestamp, \mathcal{X})$ has a size of $200 \times 30 \times 1$. In total, we injected four such blocks $\mathcal{B}$ with a mass randomly selected from [1000, 2000]. The users in the injected blocks were labeled as "malicious."

Yelp [25]. The YelpChi, YelpNYU, and YelpZip datasets [21, 25] contain restaurant reviews submitted to Yelp. They can be represented by the relation $\mathbf{R}(user, restaurant, date, X)$, where each entry $t = (u, r, d, x)$ denotes a review x by user u of restaurant r on date d. Note that all three datasets include labels indicating whether or not each review is fake. The detection of malicious reviews or users is studied in [25] using text information. In these datasets, we focus on detecting fraudulent restaurants that purchase fake reviews using the three-dimensional features. Intuitively, the more fake reviews a restaurant has, the more suspicious it is. As some legitimate users have the potential of reviewing fraudulent restaurants, we label a restaurant as "fraudulent" if it has received more than 40 fake reviews.

DARPA [19] was collected by the Cyber Systems and Technology Group in 1998 regarding network attacks in TCP dumps. The data has the form $\mathbf{R}(sourceIP, targetIP, timestamp, X)$. Each entry $t = (IP_1, IP_2, t)$ represents a connection made from IP_1 to IP_2 at time t. The dataset includes labels indicating whether or not each connection is malicious. In practice, the punishment for malicious connections is to block the corresponding IP address. Thus, we compared the detection performance of suspicious IP addresses. We labeled an IP address as suspicious if it was involved in a malicious connection.

AirForce [1] was used for the KDD Cup 1999, and has also been considered in [30, 32]. This dataset includes a wide variety of simulated intrusions in a military network environment. However, it does not contain any specific IP addresses. According to the cardinality of each dimension, we chose the top-2 features and built the relation $\mathbf{R}(src\ bytes, dst\ bytes, connections, X)$, where src bytes denotes the number of data bytes sent from source to destination and dst bytes denotes the number of data bytes sent from destination to source. The target dimension U was set to be *connections*. Note that this dataset includes labels indicating whether or not each connection is malicious.

7.2 Speed and Accuracy of D-Spot

First, we measured the speed and accuracy with which D-Spot detected dense subgraphs in real-world graphs. We compared the

Table 3: Multi-dimensional datasets used in our experiments

	Synthetic	TCP Dumps		Review Data					
		DARPA	AirForce	YelpChi	YelpNYU	YelpZip	AmaOffice	AmaBaby	AmaTools
Entries (Mass)	10K	4.6M	30K	67K	359K	1.14M	53K	160K	134K
Dimensions	7	3	3	3	3	3	3	3	3

performance of D-Spot with that of another dense-subgraph detection algorithm, Fraudar [11], the extension of [6], which maximizes the density metric by greedily deleting nodes **one by one**. We used the three Amazon datasets and applied D-Spot and Fraudar to the same bipartite graph built on the first two dimensions, *users* and *products*, where each edge in the graph represents an entry. We measured the wall-clock time (average over three runs) required to detect the top-4 subgraphs. Figure 2 illustrates the runtime and performance of the two algorithms.

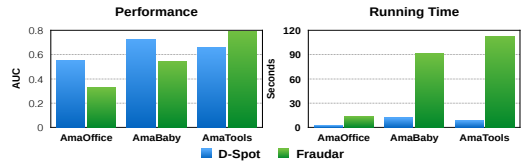


Figure 2: Comparison of D-Spot and Fraudar using the Amazon datasets.

D-Spot provides the best trade-off between speed and accuracy. Specifically, D-Spot is up to 11× faster than Fraudar. This supports our claim in Section 5.2 that the worst-case time complexity of D-Spot ($O(|\mathcal{V}|^2 + |\mathcal{E}|)$) is too pessimistic.

7.3 Effectiveness of ISG+D-Spot

This section illustrates the effectiveness of ISG+D-Spot for detecting fraudulent entities on multi-dimensional tensors. ISG+D-Spot exhibits extraordinary performance compared with the baseline methods (Fraudar is not in the baselines as it only works on the bipartite graph).

Synthetic. Table 4 presents the detection performance of each method for the hidden-densest block. We assume that the injected block $\mathcal{B}$ is the hidden-densest block when $\lambda \leq 3$. In detail, ISG+D-Spot achieves extraordinary performance even when $\lambda = 1$, because each instance of value sharing in $\mathcal{B}$ is accurately captured by ISG and D-Spot, providing a higher accuracy guarantee than the baselines (Theorems 2 and 3). When $\lambda > 3$, the performance of each method improves because the density of $\mathcal{B}$ increases as λ increases.

Amazon. Table 5 presents the results for catching suspicious users by detecting the top-4 dense blocks on the Amazon datasets. ISG+D-Spot detects synchronized behavior accurately. The typical attack scenario involves a mass of fraudulent users creating massive numbers of fake reviews for a comparatively small group of products over a short period. This behavior is represented by the injected blocks. ISG+D-Spot exhibits robust and near-perfect performance. However, the other baselines produce worse performance on the AmaOffice and AmaBaby datasets, even with the multiple supported metrics.

Table 4: Performance (AUC) on the Synthetic datasets

	$\lambda = 1$	$\lambda = 2$	$\lambda = 3$	$\lambda = 4$	$\lambda = 5$
M-Zoom (ari)	0.5005	0.5005	0.5000	0.5489	0.6567
M-Zoom (geo)	0.5005	0.5005	0.5000	0.6789	0.7543
M-Zoom (sus)	0.7404	0.7715	0.8238	0.9685	0.9767
M-Biz (ari)	0.5005	0.5005	0.5005	0.5005	0.6834
M-Biz (geo)	0.5005	0.5005	0.5005	0.6235	0.7230
M-Biz (sus)	0.6916	0.7638	0.8067	0.9844	**0.9948**
D-Cube (ari)	0.5005	0.5005	0.5005	0.5670	0.6432
D-Cube (geo)	0.5005	0.5005	0.5340	0.6876	0.6542
D-Cube (sus)	**0.8279**	**0.8712**	**0.9148**	**0.9909**	0.9725
ISG+D-Spot	**0.9843**	**0.9957**	**0.9949**	**1.0000**	**1.0000**

Table 5: Performance (AUC) on the Amazon datasets

	AmaOffice	AmaBaby	AmaTools
M-Zoom (ari)	0.6795	0.5894	0.8689
M-Zoom (geo)	0.8049	0.8049	1.0000
M-Zoom (sus)	0.7553	0.6944	0.6503
M-Biz (ari)	0.6328	0.5461	0.8384
M-Biz (geo)	0.8049	**0.8339**	1.0000
M-Biz (sus)	0.7478	0.6944	0.6503
D-Cube (ari)	0.7127	0.5956	0.6250
D-Cube (geo)	**0.8115**	0.7561	1.0000
D-Cube (sus)	0.7412	0.6190	0.5907
ISG+D-Spot	**0.8358**	**0.9995**	**1.0000**

Yelp. Table 6 reports the (highest) accuracy with which collusive restaurants were detected by each method. In summary, using ISG+D-Spot results in the highest accuracy across all three datasets, because D-Spot applies a higher theoretical bound to the ISG.

Table 6: Performance (AUC) on the Yelp datasets

	YelpChi	YelpNYU	YelpZip
M-Zoom (ari)	0.9174	0.6669	0.8859
M-Zoom (geo)	0.9752	0.8826	0.9274
M-Zoom (sus)	0.9831	**0.9451**	0.9426
M-Biz (ari)	0.9174	0.6669	0.8863
M-Biz (geo)	0.9757	0.8826	0.9271
M-Biz (sus)	**0.9831**	0.9345	**0.9403**
D-Cube (ari)	0.5000	0.5000	0.9033
D-Cube (geo)	0.9793	0.9223	0.9376
D-Cube (sus)	0.9810	0.9007	0.9365
ISG+D-Spot	**0.9875**	**0.9546**	**0.9529**

DARPA. Table 7 lists the accuracy of each method for detecting the source IP and the target IP. ISG+D-Spot assigns each IP address

a specific suspiciousness score. We chose a detected IP with the highest score and found that the IP participated in more than 1M malicious connections. The top ten suspicious IPs were all involved in more than 10k malicious connections. Thus, using ISG+D-Spot would enable us to crack down on these malicious IP addresses in the real world.

Table 7: Performance (AUC) on the DARPA dataset

Dataset	DARPA	
	U = source IP	U = target IP
M-Zoom (ari)	0.5649	0.5584
M-Zoom (geo)	0.7086	**0.5714**
M-Zoom (sus)	0.6989	0.3878
M-Biz (ari)	0.5649	0.5584
M-Biz (geo)	**0.7502**	0.5679
M-Biz (sus)	0.6989	0.3878
D-Cube (ari)	0.3728	0.5323
D-Cube (geo)	0.4083	0.3926
D-Cube (sus)	0.4002	0.3720
ISG+D-Spot	**0.7561**	**0.8181**

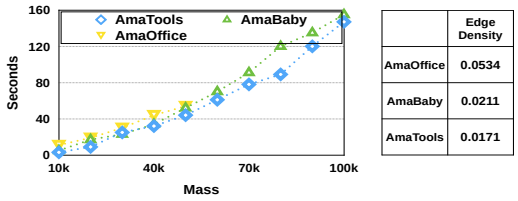
AirForce. As this dataset does not contain IP addresses, we set the target dimension U = *connections*. We randomly sample 30k connections from the dataset [1] three times. Table 8 lists the accuracy of each method on samples 1–3. Malicious connections form dense blocks on the two-dimensional features. The results demonstrate that ISG+D-Spot effectively detected the densest blocks.

Table 8: Performance (AUC) on the AirForce dataset

	Sample 1	Sample 2	Sample 3
M-Zoom (ari)	0.8696	0.8675	0.8644
M-Zoom (geo)	0.9693	0.9693	0.9738
M-Zoom (sus)	0.9684	0.9683	0.9726
M-Biz (ari)	0.9038	0.8848	0.8885
M-Biz (geo)	0.9694	0.9693	0.9741
M-Biz (sus)	0.9684	0.9683	0.9726
D-Cube (ari)	**0.9824**	**0.9823**	0.9851
D-Cube (geo)	0.9695	0.9691	**0.9862**
D-Cube (sus)	0.9697	0.9692	0.9737
ISG+D-Spot	**0.9835**	**0.9824**	**0.9877**

7.4 Scalability

As mentioned in Sec.6.1, the ISG **G**s built using real-world tensors are typically sparse, as value sharing should only conspicuously appear in fraudulent entities. We implemented **G** on the three Amazon datasets (details in Figure 3). The edge densities of **G** are quite low (lower than 0.06) across all datasets, which indicates that the worst-case time complexity discussed in Section 6.1 rarely occurs. Figure 3 reports the runtime of ISG+D-Spot on the three Amazon datasets, where the number of edges was varied by subsampling entries in the dataset. In practice, $|\mathbf{E}|$ increases near-linearly with the mass of the dataset. Additionally, because the time complexity of ISG+D-Spot is linear with respect to $|\mathbf{E}|$, ISG+D-Spot exhibits near-linear scaling with the mass of the dataset. Figure 3 demonstrates our conclusion.



	Edge Density
AmaOffice	0.0534
AmaBaby	0.0211
AmaTools	0.0171

Figure 3: ISG+D-Spot runs in near-linear time with respect to the mass of the dataset.

7.5 Feature Prioritization

This section is to demonstrate that ISG+D-Spot is more robust to resist noisy features than existing approaches. ISG automatically weighs each feature and continuously accumulates value sharing by one scan of the tensor, and D-Spot amounts to finds entities with the maximum of value sharing. We conducted the following experiment to demonstrate our conclusion.

Registration is a dataset derived from an e-commerce company, in which each record contains two crucial features, IP subnet and phone prefix, and three noisy features, IP city, phone city, and timestamp. The dataset also includes labels showing whether or not the account is a "zombie" account. Thus, it can be formulated as $\mathbf{R}(accounts, IP, phone, IP\ city, phone\ city, timestamp, X)$. To compare the detection performance of malicious accounts, we applied each method on various **R** by successively appending 1–5 features to $\mathbf{R}(accounts, X)$.

Table 9: Performance (AUC) on the Registration dataset. 'C' represents 'crucial feature' and 'N' represents 'noisy feature'

	1C	2C	2C+1N	2C+2N	2C+3N
M-Zoom (ari)	0.5000	0.5000	0.5031	0.5000	0.5430
M-Zoom (geo)	0.7676	0.8880	**0.8827**	**0.8744**	**0.8439**
M-Zoom (sus)	0.7466	0.8328	0.4009	0.4878	0.4874
M-Biz (ari)	0.5000	0.5000	0.5000	0.5000	0.5004
M-Biz (geo)	**0.7677**	0.8842	0.8827	0.8744	0.8439
M-Biz (sus)	0.7466	0.8328	0.4009	0.4878	0.4874
D-Cube (ari)	0.7073	0.8189	0.8213	0.8295	0.7987
D-Cube (geo)	0.7073	**0.9201**	0.8586	0.8312	0.7324
D-Cube (sus)	0.7522	0.8956	0.7877	0.7642	0.7080
ISG+D-Spot	**0.7699**	**0.9946**	**0.9935**	**0.9917**	**0.9859**

Table 9 gives the variation of each method with regard to the added noisy features (3– 5 dimensions). As each account only possesses one entry, **R** is quite sparse. We found that existing methods usually miss small-scale instances of value sharing because their density is close to the legitimate range on **R**. For example, a 51-member group sharing a single IP subnet was missed by the baseline methods. However, ISG amplifies each instance of value sharing through its information-theoretic and graph features, allowing D-Spot to accurately capture fraudulent entities.

8 CONCLUSIONS AND FUTURE WORK

In this paper, we novelly identified dense-block detection with dense-subgraph mining, by modeling a tensor in ISG. Additionally, we propose a multiple dense-subgraphs detection algorithm that is faster and can be computed in parallel. In future, ISG + D-Spot will be implemented on Apache Spark [39] to support very large tensors.

REFERENCES

[1] 1999. Kdd cup 1999 data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html..

[2] Melih Abdulhayoglu, Melih Abdulhayoglu, Melih Abdulhayoglu, and Melih Abdulhayoglu. 2017. HinDroid: An Intelligent Android Malware Detection System Based on Structured Heterogeneous Information Network. In *ACM SIGKDD*. 1507–1515.

[3] Leman Akoglu, Rishi Chandy, and Christos Faloutsos. 2013. Opinion Fraud Detection in Online Reviews by Network Effects.. In *ICWSM*. The AAAI Press.

[4] David S Anderson, Chris Fleizach, Stefan Savage, and Geoffrey M Voelker. 2007. Spamscatter: characterizing internet scam hosting infrastructure. In *Usenix Security Symposium on Usenix Security Symposium*. 1132–1141.

[5] Qiang Cao, Xiaowei Yang, Jieqi Yu, and Christopher Palow. 2014. Uncovering Large Groups of Active Malicious Accounts in Online Social Networks. In *ACM CCS*. 477–488.

[6] Moses Charikar. 2000. Greedy approximation algorithms for finding dense components in a graph. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*. Springer, 84–95.

[7] Jie Chen and Yousef Saad. 2012. Dense Subgraph Extraction with Application to Community Detection. *IEEE TKDE* 24, 7 (2012), 1216–1230.

[8] Hector Garcia-Molina and Jan Pedersen. 2004. Combating web spam with trustrank. In *VLDB*. 576–587.

[9] Saptarshi Ghosh, Bimal Viswanath, Farshad Kooti, Naveen Kumar Sharma, Gautam Korlam, Fabricio Benevenuto, Niloy Ganguly, and Krishna Phani Gummadi. 2012. Understanding and combating link farming in the Twitter social network. In *WWW*.

[10] A. V Goldberg. 1984. *Finding a Maximum Density Subgraph.* Technical Report.

[11] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. FRAUDAR: Bounding Graph Fraud in the Face of Camouflage. In *ACM SIGKDD*. 895–904.

[12] Meng Jiang, Alex Beutel, Peng Cui, Bryan Hooi, Shiqiang Yang, and Christos Faloutsos. 2016. Spotting Suspicious Behaviors in Multimodal Data: A General Metric and Algorithms. *IEEE TKDE* 28, 8 (2016), 2187–2200.

[13] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. CatchSync: catching synchronized behavior in large directed graphs. In *ACM SIGKDD*. 941–950.

[14] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2016. Inferring lockstep behavior from connectivity pattern in large graphs. *Knowledge & Information Systems* 48, 2 (2016), 399–428.

[15] McAuley Julian. [n. d.]. Amazon product data. http://jmcauley.ucsd.edu/data/amazon/.

[16] Samir Khuller and Barna Saha. 2009. On Finding Dense Subgraphs. In *Automata, Languages and Programming, International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings*. 597–608.

[17] Tamara G. Kolda and Brett W. Bader. 2009. Tensor Decompositions and Applications. *Siam Review* 51, 3 (2009), 455–500.

[18] Victor E. Lee, Ruan Ning, Ruoming Jin, and Charu Aggarwal. 2010. *A Survey of Algorithms for Dense Subgraph Discovery.* 303–336 pages.

[19] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, and M. A. Zissman. 2000. Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation. In *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, Vol. 2. 12–26 vol.2. https://doi.org/10.1109/DISCEX.2000.821506

[20] Koji Maruhashi, Fan Guo, and Christos Faloutsos. 2011. MultiAspectForensics: Pattern Mining on Large-Scale Heterogeneous Networks with Tensor Analysis. In *ASONAM*. 203–210.

[21] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie Glance. 2013. What yelp fake review filter might be doing?. In *Proceedings of the 7th International Conference on Weblogs and Social Media, ICWSM 2013*. AAAI press, 409–418.

[22] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. 2007. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *WWW*. 201–210.

[23] Evangelos E. Papalexakis, Christos Faloutsos, and Nicholas D. Sidiropoulos. 2012. ParCube: sparse parallelizable tensor decompositions. In *PKDD*. 521–536.

[24] B. Aditya Prakash, Ashwin Sridharan, Mukund Seshadri, Sridhar Machiraju, and Christos Faloutsos. 2010. EigenSpokes: Surprising Patterns and Scalable Community Chipping in Large Graphs. In *PAKDD*.

[25] Shebuti Rayana and Leman Akoglu. 2015. Collective Opinion Spam Detection: Bridging Review Networks and metadata. In *ACM SIGKDD*.

[26] Barna Saha, Allison Hoch, Samir Khuller, Louiqa Raschid, and Xiao Ning Zhang. 2010. *Dense Subgraphs with Restrictions and Applications to Gene Annotation Graphs.* Springer Berlin Heidelberg. 456–472 pages.

[27] Neil Shah, Alex Beutel, Brian Gallagher, and Christos Faloutsos. 2014. Spotting Suspicious Link Behavior with fBox: An Adversarial Perspective. In *IEEE ICDM*. 959–964.

[28] C. E Shannon. 1948. A mathematical theory of communication. *Bell Labs Technical Journal* 27, 4 (1948), 379–423.

[29] K. Shin, T. Eliassi-Rad, and C. Faloutsos. 2017. CoreScope: Graph Mining Using k-Core Analysis – Patterns, Anomalies and Algorithms. In *ICDM*, Vol. 00. 469–478.

[30] Kijung Shin, Bryan Hooi, and Christos Faloutsos. 2016. M-Zoom: Fast Dense-Block Detection in Tensors with Quality Guarantees. In *ECML PKDD*. 264–280.

[31] Kijung Shin, Bryan Hooi, and Christo Faloutsos. 2018. Fast, Accurate, and Flexible Algorithms for Dense Subtensor Mining. *ACM Transactions on Knowledge Discovery from Data* 12, 3 (2018), 1–30.

[32] Kijung Shin, Bryan Hooi, Jisu Kim, and Christos Faloutsos. 2017. D-Cube: Dense-Block Detection in Terabyte-Scale Tensors. In *WSDM*. 681–689.

[33] Kijung Shin and U. Kang. 2015. Distributed Methods for High-Dimensional and Large-Scale Tensor Factorization. In *IEEE ICDM*. 989–994.

[34] Ming Yang Su. 2011. Real-time anomaly detection systems for Denial-of-Service attacks by weighted k-nearest-neighbor classifiers. *Expert Systems with Applications* 38, 4 (2011), 3492–3498.

[35] Hua Tang and Zhuolin Cao. 2009. Machine Learning-based Intrusion Detection Algorithms. *Journal of Computational Information Systems* (2009), 1825–1831.

[36] Kurt Thomas, Dmytro Iatskiv, Elie Bursztein, Tadek Pietraszek, Chris Grier, and Damon McCoy. 2014. Dialing back abuse on phone verified accounts. In *Proceedings of the 2014 ACM SIGSAC*. ACM, 465–476.

[37] Yining Wang, Hsiao Yu Tung, Alexander Smola, and Animashree Anandkumar. 2015. Fast and Guaranteed Tensor Decomposition via Sketching. *NIPS*.

[38] Wanhong Xu, Wanhong Xu, Christopher Palow, Christopher Palow, and Christos Faloutsos. 2013. CopyCatch: stopping group attacks by spotting lockstep behavior in social networks. In *WWW*. 119–130.

[39] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy Mccauley, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2012. Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In *Usenix NSDI*. 2–2.

Link Prediction in Networks with Core-Fringe Data

Austin R. Benson
Cornell University
arb@cs.cornell.edu

Jon Kleinberg
Cornell University
kleinber@cs.cornell.edu

ABSTRACT

Data collection often involves the partial measurement of a larger system. A common example arises in collecting network data: we often obtain network datasets by recording all of the interactions among a small set of core nodes, so that we end up with a measurement of the network consisting of these core nodes along with a potentially much larger set of fringe nodes that have links to the core. Given the ubiquity of this process for assembling network data, it is crucial to understand the role of such a "core-fringe" structure.

Here we study how the inclusion of fringe nodes affects the standard task of network link prediction. One might initially think the inclusion of any additional data is useful, and hence that it should be beneficial to include all fringe nodes that are available. However, we find that this is not true; in fact, there is substantial variability in the value of the fringe nodes for prediction. Once an algorithm is selected, in some datasets, including any additional data from the fringe can actually hurt prediction performance; in other datasets, including some amount of fringe information is useful before prediction performance saturates or even declines; and in further cases, including the entire fringe leads to the best performance. While such variety might seem surprising, we show that these behaviors are exhibited by simple random graph models.

ACM Reference Format:
Austin R. Benson and Jon Kleinberg. 2019. Link Prediction in Networks with Core-Fringe Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313626

1 INTRODUCTION

In a wide range of data analysis problems, the underlying data typically comes from partial measurement of a larger system. This is a ubiquitous issue in the study of networks, where the network we are analyzing is almost always embedded in some larger surrounding network [17, 20, 21]. Such considerations apply to systems at all scales. For example, when studying the communication network of an organization, we can potentially gain additional information if we know the structure of employee interactions with people outside the organization as well [32]. A similar issue applies to large-scale systems. If we are analyzing the links within a large online social network, or the call traffic data from a large telecommunications provider, we could benefit from knowing the interactions that members of these systems have with individuals who are not part of the platform, or who do not receive service from the provider.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313626

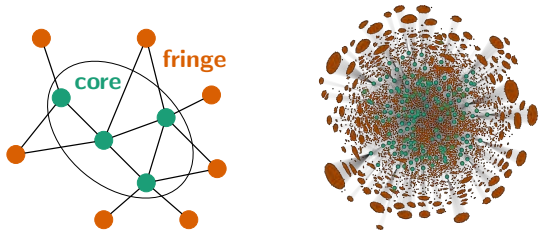


Figure 1: Core-fringe structure. (Left) Illustrative network with labeled core-fringe structure (core nodes in green and fringe nodes in orange). We observe all of the links involving core nodes (in green). Each edge is between two core nodes or between one core and one fringe node. (Right) Core-fringe structure in the Enron email network, which results from data collection—the core nodes in green correspond to the roughly 150 accounts whose emails were released as part of a federal investigation [19]. Here, the number of fringe nodes is orders of magnitude larger than the number of core nodes.

Network data can therefore be viewed as having a *core-fringe* structure (following the terminology of our previous work [6]): we collect data by measuring all of the interactions involving a *core* set of nodes, and in the process we also learn about the core's interaction with a typically larger set of additional nodes—the *fringe*—that does not directly belong to the measured system. Figure 1 illustrates the basic structure schematically and also in a typical real-life scenario: If we collect a dataset by measuring all email communication to and from the executives of a company (the core), then the data will also include links to people outside this set with whom members of the core exchanged email (the fringe).[1] We thus have two kinds of links: links between two members of the core, and links between a member of the core and a member of the fringe. Links between fringe members are not visible, even though we are aware of both fringe nodes through their interactions with the core.

Despite the fundamental role of core-fringe structure in network data and a long history of awareness of this issue in the social sciences [21], there has been little systematic attention paid to its implications in basic network inference tasks. If we are trying to predict network structure on a measured set of core nodes, what is the best way to make use of the fringe? Is it even clear that incorporating the fringe nodes will help? To study these questions, it is important to have a concrete task on the network where notions of performance as a function of available data are precise.

The present work: Core-fringe link prediction. In this paper, we study the role of core-fringe structure through one of the standard network inference problems: *link prediction* [24, 26]. Link

[1] This distinction between core and fringe is fundamentally driven by measurement of the available data; we have measured all interactions involving members of the core, and this brings the fringe indirectly into the data. As such, it is distinct from work on the *core-periphery structure* of networks, which typically refers to settings in which the core and periphery both fully belong to the measured network, and the distinction is in the level of centrality or status that the core has relative to the periphery [7, 16, 31, 37].

prediction is a problem in which the goal is to predict the presence of *unseen* links in a network. Links may be unseen for a variety of reasons, depending on the application—we may have observed the network up to a certain point in time and want to forecast new links, or we may have collected a subset of the links and want to know which additional ones are present.

Abstractly, we will think of the link prediction problem as operating on a graph $G = (V, E)$ whose edges are divided into a set of observed edges and a set of unseen edges. From the network structure on the observed edges, we would like to predict the presence of the unseen edges as accurately as possible. A large range of heuristics have been proposed for this problem, many of them based on the empirical principle that nodes with neighbors in common are generally more likely to be connected by a link [24, 26, 28].

The issue of core-fringe structure shows up starkly in the link prediction problem. Suppose the graph G has nodes that are divided into a core set C and a fringe set F, and our goal is to predict unseen links between pairs of nodes in the core. One option would be to perform this task using only the portion of G induced on the core nodes. But we could also perform the task using larger amounts of G by taking the union of the core nodes with any subset of the fringe, or with all of the fringe. The key question is how much of the fringe we should include if our goal is to maximize performance on the core; existing work provides little guidance about this question.

How much do fringe nodes help in link prediction? We explore this question in a broad collection of network datasets derived from email, telecommunication, and online social networks. For concreteness, our most basic formulation draws on common-neighbor heuristics to answer the following version of the link prediction question: given two pairs of nodes drawn from the core, $\{u, v\}$ and $\{w, z\}$, which pair is more likely to be connected by a link? (In our evaluation framework, we will focus on cases in which exactly one of these pairs is truly connected by a link, thus yielding a setting with a clear correct answer.) To answer this question, we could use information about the common neighbors that $\{u, v\}$ and $\{w, z\}$ have only in the core, or also in any subset of the fringe. How much fringe information should we use, if we want to maximize our probability of getting the correct answer?

It would be natural to suspect that using all available data, i.e., including all of the fringe nodes, would maximize performance. What we find, however, is a wide range of behaviors. In some of our domains—particularly the social-networking data—link prediction performance increases monotonically in the amount of fringe data used, though with diminishing returns as we incorporate the entire fringe. In the other domains, however, we find a number of instances where using an intermediate level of fringe, i.e., a proper subset of the fringe nodes, yields a performance that dominates the option of including all of the fringe or none of it. And there are also cases where prediction is best when we ignore the fringe entirely. Given that proper subsets of the fringe can yield better performance than either extreme, we also consider the process of selecting a subset of the fringe; in particular, we study different natural *orderings* of the fringe nodes and then select a subset by searching over prefixes of these orderings.

To try understanding this diversity of results, we turn to basic random graph models, adapting them to capture the problem of link prediction in the presence of core-fringe structure. We find that simple models are rich enough to display the same diversity of behaviors in performance, where the optimal amount of fringe might be all, some, enough, or none. More specifically, we analyze the signal-to-noise ratio for our basic link prediction primitive in two heavily-studied network models: stochastic block models (SBMs), in which random edges are added with different probabilities between a set of planted clusters [1, 2, 10, 27]; and small-world lattice models, in which nodes are embedded in a lattice, and links between nodes are added with probability decaying as a power of the distance [18, 35]. We prove that there are instances of the SBM with certain linking probabilities in which the signal-to-noise ratio is optimized by including all the fringe, enough of the fringe, or none of the fringe. For small-world lattice models, we find in the most basic formulation that the signal-to-noise ratio is optimized by including an intermediate amount of fringe: essentially, if the core is a bounded geometric region in the lattice, then the optimal strategy for link prediction is to include the fringe in a larger region that extends out from the core; but if we grow this region too far then performance will decline.

The analysis of these models provides us with some qualitative higher-level insight into the role of fringe nodes in link prediction. In particular, the analysis can be roughly summarized as follows: the fringe nodes that are most well-connected to the core are providing valuable predictive signal without significantly increasing the noise; but as we continue including fringe nodes that are less and less well-connected to the core, the signal decreases much faster than the noise, and eventually the further fringe nodes are primarily adding noise in a way that hurts prediction performance.

More broadly, the results here indicate that the question of how to handle core-fringe structure in network prediction problems is a rich subject for investigation, and an important one given how common this structure is in network data collection. An implication of both our empirical and theoretical results is that it can be important for problems such as link prediction to measure performance with varying amounts of additional data, and to accurately evaluate the extent to which this additional data is primarily adding signal or noise to the underlying decision problem.

Finally, software and data associated with this paper are available at https://github.com/arbenson/cflp.

2 EMPIRICAL NETWORK ANALYSIS

We first empirically study how including fringe nodes can affect link prediction on a number of datasets. While we might guess that any additional data we can gather would be useful for prediction, we see that this is not the case. In different datasets, incorporating all, none, some, or enough fringe data leads to the best performance. We then show in Section 3 that this variability is also exhibited in the behavior of simple random graph models.

Evaluating link prediction with core-fringe structure. There are several ways to evaluate link prediction performance [24, 26]. We set up the prediction task in a natural way that is also amenable to theoretical analysis in Section 3. We assume that we have a graph $G = (V, E)$ with a known set of core nodes $C \subseteq V$ and fringe nodes $F = V - C$. The edge set is partitioned into E^{train} and E^{test}, where E^{test} is a subset of the edges that connect two nodes in the core C. The form of E^{test} depends on the dataset, which we describe in

the following sections. In general, our core-fringe link prediction evaluation is based on how well we can predict elements of E^{test} given the graph $G^{\text{train}} = (V, E^{\text{train}})$.

Our atomic prediction task considers two pairs of nodes $\{u, v\}$ and $\{w, z\}$ such that (i) all four nodes are in the core (i.e., $u, v, w, z \in C$); (ii) neither pair is an edge in E^{train}; (iii) the edge (u, v) is a positive sample, meaning that $(u, v) \in E^{\text{test}}$; and (iv) the edge (w, z) is a negative sample, meaning that $(w, z) \notin E^{\text{test}}$. We use an algorithm that takes as input G^{train} and outputs a score function $s(x, y)$ for any pair of nodes $x, y \in C$; the algorithm then predicts that the pair of nodes with the higher score is more likely to be in the test set. Thus, the algorithm makes a correct prediction if $s(u, v) > s(w, z)$. We sample many such 4-tuples of nodes uniformly at random and measure the fraction of correct predictions.

We evaluate two score functions that are common heuristics for link prediction [24]. The first is the *number of common neighbors*:

$$s(x, y) = |N(x) \cap N(y)|, \tag{1}$$

where $N(z)$ is the set of neighbors of node z in the graph. The second is the *Jaccard similarity* of the neighbor sets:

$$s(x, y) = \frac{|N(x) \cap N(y)|}{|N(x) \cup N(y)|}. \tag{2}$$

We choose these score functions for a few reasons. First, they are flexible enough to be feasibly deployed even if only minimal information about the fringe is available; more generally, their robustness has motivated their use as heuristics in practice [14, 15] and throughout the line of research on link prediction [24]. Second, they are amenable to analysis: we can explain some of our results by analyzing the common neighbors heuristic on random graph models, and they are rich enough to expose a complex landscape of behavior. This is sufficient for the present study, but it would be interesting to examine more sophisticated link prediction algorithms in our core-fringe framework as well.

We parameterize the training data by how much fringe information is included. To do this, we construct a nested sequence of sets of vertices, each of which induces a set of training edges. Specifically, the initial set of vertices is the core, and we continue to add fringe nodes to construct a nested sequence of vertices:

$$C = V_0 \subseteq V_1 \subseteq \cdots \subseteq V_D = V. \tag{3}$$

The nested sequence of vertex sets then induces a nested sequence of edges that are the training data for the link prediction algorithm; for a value of d between 0 and D, we write

$$E_d^{\text{train}} = \{(u, v) \in E \mid u, v \in V_d\} \cap E^{\text{train}}. \tag{4}$$

From Eqs. (3) and (4), $E_d^{\text{train}} \subseteq E_{d+1}^{\text{train}}$, and $E_D^{\text{train}} = E^{\text{train}}$. The parameterization of the vertex sets will depend on the dataset, and we examine multiple sequences $\{V_d\}$ to study how different interpretations of the fringe give varying outcomes. Our main point of study is link prediction performance *as a function of* d.

2.1 Email networks

Our first set of experiments analyzes email networks. The core nodes in these datasets are members of some organization, and the fringe nodes are those outside of the organization that communicate with those in the core. We use four email networks; in each, the nodes are email addresses, and the time that an edge formed is given by the timestamp of the first email between two nodes. For simplicity, we consider all graphs to be undirected, even though there is natural directionality in the links. Thus, each dataset is a simple, undirected graph, where each edge has a timestamp and each node is labeled as core or fringe. The four datasets are (i) *email-Enron*: the network in Figure 1, where the core nodes correspond to accounts whose emails were released as part of a federal investigation [6, 19]; (ii) *email-Avocado*: the email network of a now-defunct company, where the core nodes are employees (we removed accounts associated with non-people, such as conference rooms).[2] (iii) *email-Eu*: a network that consists of emails involving members of a European research institution, where the core nodes are the institution's members [23, 36]; and (iv) *email-W3C*: a network from W3C email threads, where core nodes are those addresses with a `w3.org` domain [6, 9]. Table 1 provides basic summary statistics.

Table 1: Summary statistics of email datasets.

Dataset	# core nodes	# fringe nodes	# core-core edges	# core-fringe edges
email-Enron	148	18,444	1,344	41,883
email-Avocado	256	27,988	7,416	50,048
email-Eu	1,218	200,372	16,064	303,793
email-W3C	1,995	18,086	1,777	30,097
email-Enron-1	37	7,511	86	11,862
email-Enron-2	37	6,440	81	10,648
email-Enron-3	37	6,379	80	10,390
email-Enron-4	37	6,587	95	10,987

An entire email network dataset is a graph $G = (V, E)$, where $C \subseteq V$ is a set of core nodes, and each edge $e \in E$ is associated with a timestamp t_e. Here, our test set is derived from the temporal information. Let t^* be the 80th percentile of timestamps on edges between core nodes. Our test set of edges is the final 20% of edges between core nodes that appear in the dataset:

$$E^{\text{test}} = \{(u, v) \in E \mid u, v \in C \text{ and } t_{(u,v)} \geq t^*\}. \tag{5}$$

The training set is then given by edges appearing before t^*, i.e., the edges appearing in the first 80% of time spanned by the data:

$$E^{\text{train}} = \{(u, v) \in E \mid t_{(u,v)} < t^*\}. \tag{6}$$

Fringe ordering. Next, we form the nested sequence of training set edges (Eq. (4)) by sequentially increasing the amount of fringe information (Eq. (3)). Recall that E_d^{train} is simply the set of edges in E^{train} in which both end points are in the vertex set V_d. To build $\{V_d\}$, we start with $V_0 = C$, the core set, and then add the fringe nodes one by one in order of decreasing degree in the graph $G^{\text{train}} = (V, E^{\text{train}})$. By definition, fringe nodes cannot link between themselves, so this ordering is equivalent to adding fringe nodes in order of the number of core nodes to which they connect. For purposes of comparison, we also evaluate this ordering relative to a uniformly random ordering of the fringe nodes. To summarize, given an ordering σ of the fringe nodes F, we create a nested sequence of node sets $V_d = C \cup \{\sigma_1, \ldots, \sigma_d\}$ in two ways: (i) *Most connected*: σ is the ordering of nodes in the fringe F by decreasing

[2] https://catalog.ldc.upenn.edu/LDC2015T03

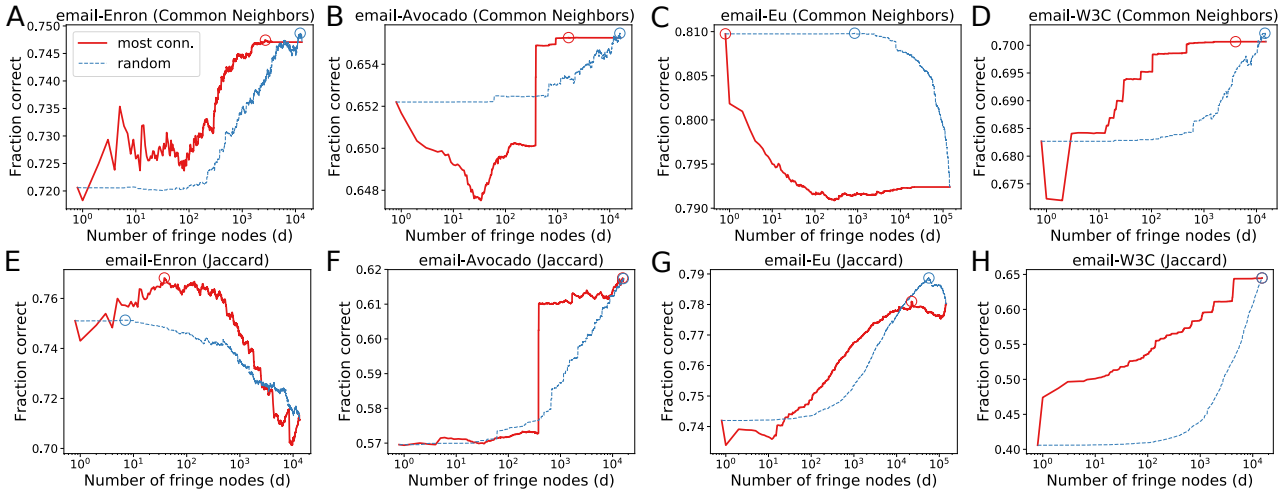


Figure 2: Link prediction performance of the Common Neighbors (top) and Jaccard similarity (bottom) score functions on four email networks as a function of d, the number of fringe nodes included. Two orderings of fringe nodes are considered: one by the most connections to the core (red) and one random (blue). A circle marks the best performance. There is a striking variety in how the fringe affects performance. In some cases, we should ignore the fringe entirely (C); in others, performance increases with the size of the fringe (F, H); and in still others, some intermediate amount of fringe is optimal (E, G).

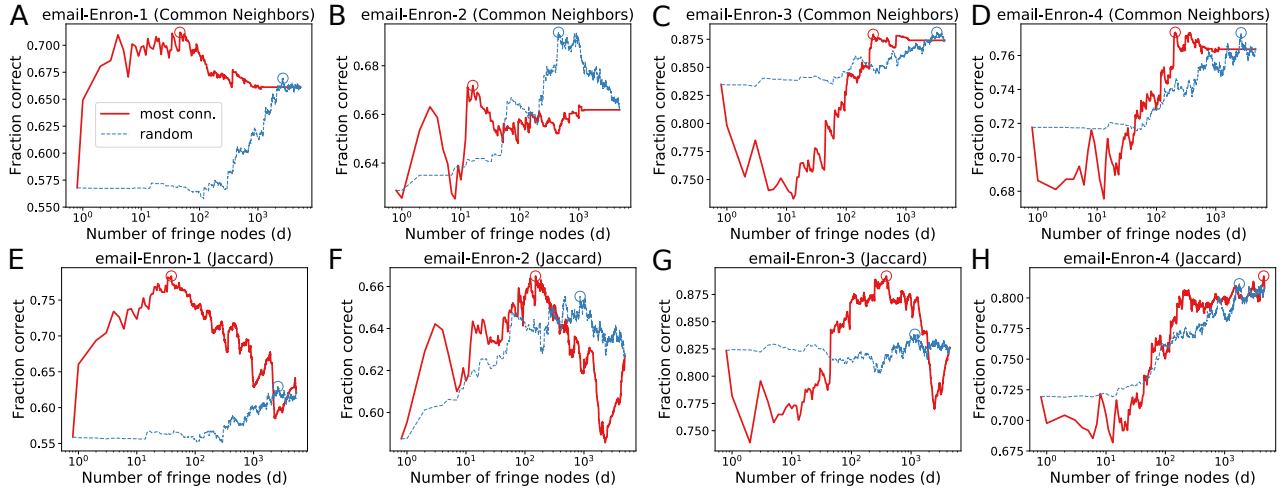


Figure 3: Link prediction performance experiments analogous to those in Figure 2 but on four subsets of email-Enron. In most cases, including some interior amount of fringe nodes—between 10 and a few hundred—yields the optimal performance.

degree in the graph induced by E^{train} (Eq. (6)); and (ii) *Random*: σ is a random ordering of the nodes in F.

Link prediction. We use the *most connected* and *random* ordering to predict links in the test set of edges, as described at the beginning of Section 2. Recall that we needed a set of candidate comparisons between two potential edges (one of which does appear in the test). We guess that the pair of nodes with the larger number of common neighbors or larger Jaccard similarity score will be the set that appears in the test set. We sample $10 \cdot |E^{\text{test}}|$ pairs from E^{test} (allowing for repeats) and combine each of them with two nodes selected uniformly at random that never form an edge. Prediction performance is measured in terms of the fraction of correct guesses as a function of d, the number of fringe nodes included. This entire procedure is repeated 10 times (with 10 different sets of random samples) and the mean accuracy is reported in Figure 2.

The results exhibit a wide variety of behavior. In some cases, performance tends to increase monotonically with the number of fringe nodes (Figures 2F and 2H). In one case, we achieve optimal performance by ignoring the fringe entirely (Figure 2C). In yet another case, some interior amount of fringe is optimal before prediction performance degrades (Figure 2E). In several cases, we see a saturation effect, where performance flattens as we increase more fringe nodes (e.g., Figures 2A and 2D). This is partly a consequence of how we ordered the fringe—nodes included towards the end of the sequence are less connected and thus have relatively less impact on the score functions. In these cases, one practical consequence is

Table 2: Summary statistics of telecommunications datasets. Core nodes are participants in the Reality Mining study.

Dataset	# core nodes	# fringe nodes	# core-core edges	# core-fringe edges
call-Reality	91	8,927	127	10,512
text-Reality	91	1,087	32	1,920

that we could ignore large amounts of the data and get roughly the same performance, which would save computation time. In another case, the first few hundred most connected fringe nodes leads to worse performance, but eventually having enough fringe improves performance (Figure 2B). Finally, there are also cases where the optimal performance over d is better for a random ordering of the fringe than for the most connected ordering (Figures 2D and 2G).

We repeated the same set of experiments on subgraphs of email-Enron by partitioning the core set of nodes C into four groups to induce four different datasets. In each dataset, the other members of the core are removed from the graph entirely (the bottom part of Table 1 lists basic summary statistics). The results in Figure 3 provide further evidence that it is a priori unclear how much fringe information one should include to achieve the best performance. In nearly all cases, the optimal performance when including fringe nodes in order of connectedness is somewhere between 10 and a few hundred nodes, out of a total of several thousand. And we again see that including initial fringe information by connectedness has worse performance than ignoring the fringe entirely, until eventually incorporating enough fringe information provides enough signal to improve prediction performance (Figures 3C and 3D).

2.2 Telecommunications networks

Next, we study telecommunications datasets from cell phone usage amongst individuals participating in the Reality Mining project [12]. This project recorded cell phone activity of students and faculty in the MIT Media Laboratory, including calls and SMS texts between phone numbers. We consider the participants (more, specifically, their phone numbers) as the core nodes in our network. Edges are phone calls or SMS texts between two people, some of which are fringe nodes corresponding to people who were not recruited for the experiment. We process the data in the same way as for email networks—directionality was removed and the edges are accompanied by the earliest timestamp of communication between the two nodes. We study two datasets: (i) *call-Reality*: the network of phone calls [6, 12]; and (ii) *text-Reality*: the network of SMS texts [6, 12]. Table 2 provides some basic summary statistics.

Fringe ordering. The structure of these networks is the same as the email networks—the dataset is a recording of the interactions of a small set of core nodes with a larger set of fringe nodes. We use the same two orderings—most connected to core and random—as we did for the email datasets. Thus, the nested sequence of node sets $\{V_d\}$ is again constructed by adding one node at a time.

Link prediction. Figure 4 shows the link prediction performance on the telecommunications datasets. With the Common Neighbors score function, we again find that the optimal amount of fringe is a small fraction of the entire dataset—around 100 of nearly 9,000 nodes in call-Reality (Figure 4A) and around 10 of over 1,000 nodes in text-Reality (Figure 4B). The performance of the Jaccard similarity

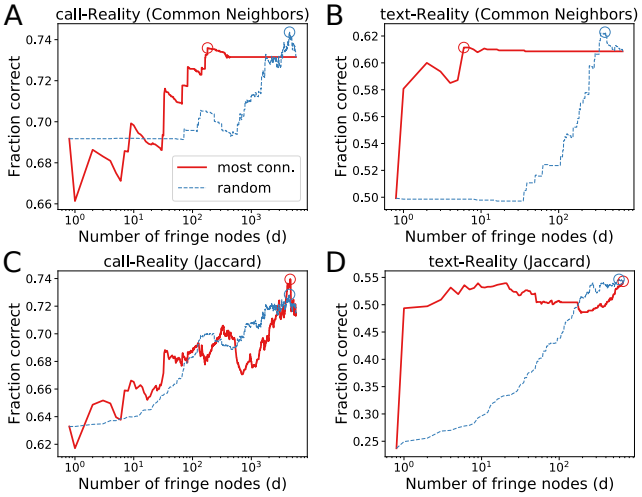


Figure 4: Link prediction performance in the telecommunications datasets as a function of d, the number of fringe nodes used for prediction. Circles mark the largest value. With the Common Neighbors score function, a small number of fringe nodes is optimal for these datasets.

also has an interior optimum for the call-Reality dataset, although the optimum size here is larger—around half of the nodes.

Prediction performance with the fringe nodes ordered by connectedness to the core is in general quite erratic for the call-Reality dataset. This is additional evidence that the fringe nodes can be a noisy source of information. For instance, just including the first fringe node results in a noticeable drop in prediction performance for both the Common Neighbors and Jaccard score functions.

2.3 Online social networks

We now turn to links in an online social network of bloggers from the LiveJournal community [25]. Edges are derived from users listing friends in their profile (here, we consider all links as undirected). Users also list their geographic location, and for the purposes of this study, we have restricted the dataset to users reporting locations in the United States and Puerto Rico. For each user, we have both their territory of residence (one of the 50 U.S. states, Washington D.C., or Puerto Rico; henceforth simply referred to as "state") as well as their county of residence, when applicable.

Table 3: Summary statistics of LiveJournal networks. The sets of core nodes are users in particular states or counties.

Dataset	# core nodes	# fringe nodes	# core-core edges	# core-fringe edges
Wisconsin	16,842	58,965	48,078	87,723
Texas	65,617	155,357	256,174	312,746
New York	82,275	208,516	281,981	477,845
California	152,171	244,605	712,803	722,835
Marathon	223	1,032	390	1,363
Eau Claire	295	1,392	252	1,635
Dane	2,281	15,580	5,192	21,156
Milwaukee	3,743	19,934	10,750	30,955

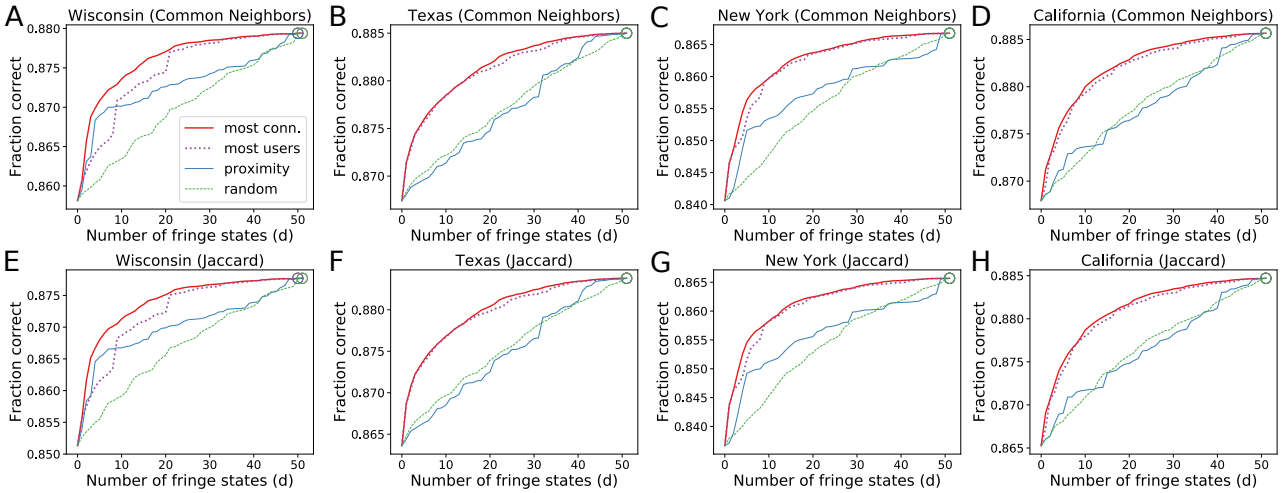


Figure 5: Link prediction performance of the Common Neighbors (top) and Jaccard similarity (bottom) score functions on four LiveJournal datasets where the core consists of users in a state. Performance is measured as a function of d, the number of fringe states included for prediction (for four different orderings of the fringe). The most connected ordering performs the best, monotonically increases with d, and saturates when $d \geq 20$.

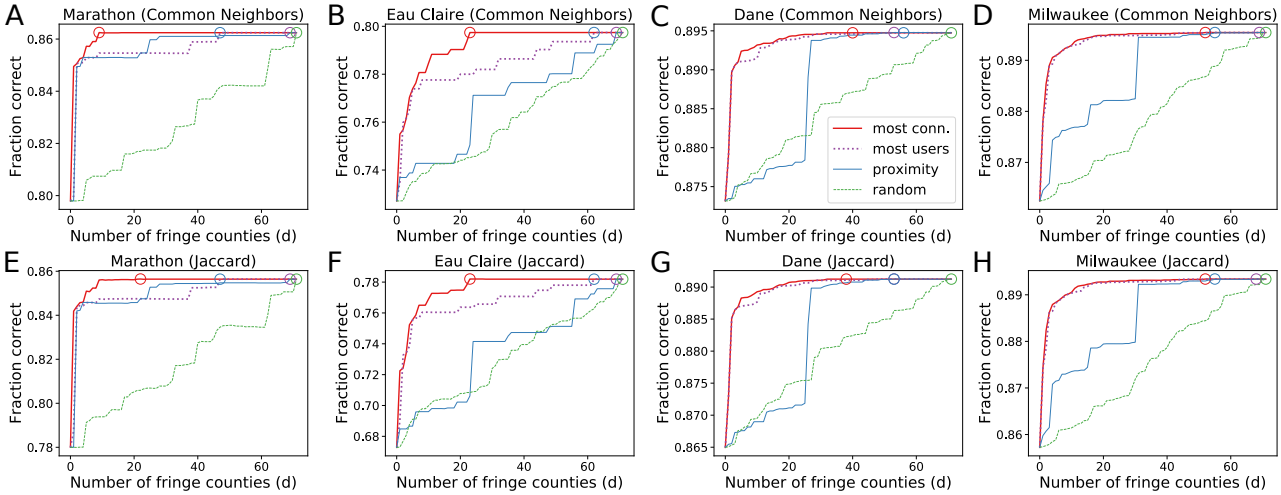


Figure 6: Link prediction analogous to Figure 5 but where the core is a Wisconsin county and the fringe is the rest of the state. Performance is measured as a function of d, the number of fringe counties included for prediction. The most connected ordering performs the best, and performance with this ordering quickly saturates.

We construct core-fringe networks in two ways. First, we form a core from all user residing in a given state S. The core-fringe network then consists of all friendships where at least one node is in state S. We construct four such networks, using the states Wisconsin, Texas, California, and New York. Second, we form a core from users residing in a county and construct a core-fringe network in the same way, but we only consider friendships amongst users in the state containing the county. We construct four such networks, using Marathon, Eau Claire, Dane, and Milwaukee counties (all in Wisconsin). Table 3 lists summary statistics of the datasets.

Unlike the email and telecommunications networks, we do not have timestamps on the edges. We instead construct E^{test} from a random sample of 20% of the edges between core nodes, i.e., from $\{(u, v) \in E \mid u, v \in C\}$, and set $E^{\text{train}} = E - E^{\text{test}}$. Predicting on such held out test sets is used for predicting missing links [8, 13]; here, we use it for link prediction, as is standard practice [26].

Fringe ordering. We again incorporate fringe nodes from a nested sequence of node sets $\{V_d\}$, where $V_d \subseteq V_{d+1}$ and $V_0 = C$, the set of core nodes. The nested sequence of training sets is then $E_d^{\text{train}} = \{(u, v) \in E^{\text{train}} \mid u, v \in V_d\}$. With email and telecommunications, we considered fringe nodes one by one to form the sequence $\{V_d\}$. For LiveJournal, each successive node set instead corresponds to adding all nodes in a state or a county. For the cores constructed from users in a state (Wisconsin, Texas, New York, or California), we form orderings σ of the remaining states in four ways: (i) *Most connected*: σ is the ordering of states by decreasing

number of links to the core state; (ii) *Most users*: σ is the ordering by decreasing number of users; (iii) *Proximity*: σ is the ordering of states by closest proximity to the core state (measured by great-circle distance between geographic centers); and (iv) *Random*: σ is a random ordering of the states.

Let U_S be the users in state S. The sequence of vertex sets is all users in the core and first d states in the ordering σ. Formally, $V_d = C \cup (\cup_{t=1}^{d} U_{\sigma_t})$. For networks whose core are users in a Wisconsin county, we use the same orderings, except we order counties instead of states and the fringe is only counties in Wisconsin.

Link prediction. We measure the mean prediction performance over 10 random trials as a function of the number d of fringe states or counties included in the training data. When states form the core, prediction performance is largely consistent (Figure 5). For both score functions, ordering by number of connections tends to perform the best, with a rapid performance increase from approximately the first 10 states and then saturation with a slow monotonic increase. The prediction by states with the most users performs nearly the same for the three largest states (Texas, New York, and California; Fiugres 5B to 5D). In Wisconsin and New York, ordering by proximity shows a steep rise in performance for the first few states but then levels off (Figures 5A, 5C, 5E and 5G). On the other hand, in California and Texas, ordering by proximity performs roughly as well as a random ordering.

The networks where the cores are users from counties in Wisconsin have similar characteristics to the networks where the cores are users from particular states (Figure 6). The ordering by county with the most connections performs the best. Prediction performance quickly saturates in the two larger counties (Figures 6C and 6D), and the proximity ordering can be good in some cases (Figure 6A).

Summary. Usually, collecting additional data is thought to improve performance in machine learning. Here we have seen that this is not the case in some networks with core-fringe structure. In fact, including additional fringe information can affect link prediction performance in a number of ways. In some cases, it is indeed true that additional fringe always helps, which was largely the case with LiveJournal (Figure 5). In one email network, including any fringe data hurt performance (Figure 2C). And yet in other cases, some intermediate amount of fringe data gave the best performance (Figures 4A and 4B; Figure 3). We also observed saturation in link prediction performance as we increased the fringe size (Figure 6) and that sometimes we need enough fringe before prediction becomes better than incorporating no fringe at all (Figures 2B and 3C). While this landscape is complex, we show in the next section how these behaviors can emerge in simple random graph models.

3 RANDOM GRAPH MODEL ANALYSIS

We now turn to the question of *why* link prediction in core-fringe networks exhibits such a wide variation in performance. To gain insight into this question, we analyze the link prediction problem on basic random graph models that have been adapted to contain core-fringe structure.

Recall how our link prediction problem is evaluated: we are given two pairs $\{u, v\}$ and $\{w, z\}$; our algorithm predicts which of the two candidate edges is the one that appears in the data through a score function; and the values of the score function (and hence the predictions) can change based on the inclusion of fringe nodes. In a random graph model, we can think about using the same score functions for the candidate edges (u, v) and (w, z), but now the score functions and the existence of edges are random variables. As we will show, the signal-to-noise ratio of the difference in score functions is a key statistic to optimize in order to make the most accurate predictions, and this can vary in different ways when including fringe nodes.

The signal-to-noise ratio (SNR). Suppose our data is generated by a random graph model and that the indicator random variables $X, Y \in \{0, 1\}$ correspond to the existence of two candidate edges $\{u, v\}$ and $\{w, z\}$, respectively, where nodes u, v, w, and z are distinct and chosen uniformly at random amongst a set of core nodes. Without loss of generality, we assume that $\Pr[X] > \Pr[Y]$ so that (u, v) is more likely to appear.

We would like our algorithm to predict that the edge $\{u, v\}$ is the one that exists, since this is the more likely edge (by assumption). However, our algorithm does not observe X and Y directly; instead, it sees proxy measurements $(\hat{X}, \hat{Y})$, which are themselves random variables. These proxy measurements correspond to the score function used by the algorithm; in this section, we focus on the number of common neighbors score. Our algorithm will (correctly) predict that edge $\{u, v\}$ is more likely if and only if $\hat{X} > \hat{Y}$.

Furthermore, the proxy measurements are parameterized by the amount of fringe information we have. Following our previous notation, we call these random variables $\hat{X}_d$ and $\hat{Y}_d$. These variables represent the same measurements as $\hat{X}$ and $\hat{Y}$ (such as the number of common neighbors), just on a set of graphs parameterized by the amount of fringe information d.

Our goal is to optimize the amount of fringe to get the most accurate predictions. Formally, if we let $Z_d \triangleq \hat{X}_d - \hat{Y}_d$, this means:

$$\underset{d}{\text{maximize}} \quad \Pr[\hat{X}_d > \hat{Y}_d] \iff \underset{d}{\text{maximize}} \quad \Pr[Z_d > 0].$$

We assume that we have access to $\mathbb{E}[Z_d]$ and $\mathbb{V}[Z_d]$ for all values of d. Our approach will be to maximize the signal-to-noise ratio (SNR) statistic of Z_d, i.e.,

$$\underset{d}{\text{maximize}} \quad \frac{\mathbb{E}[Z_d]}{\sqrt{\mathbb{V}[Z_d]}} \triangleq \text{SNR}[Z_d].$$

We motivate this approach as follows. Absent any additional information beyond the expectation and variance, we must use some concentration inequality. Under the reasonable assumption that $\mathbb{E}[Z_d] > 0$ (which we later show holds in our models), the proper inequality is Cantelli's: $\Pr[Z_d \geq 0] \geq 1 - \frac{\mathbb{V}[Z_d]}{\mathbb{V}[Z_d]+\mathbb{E}[Z_d]^2} = \frac{\text{SNR}[Z_d]^2}{1+\text{SNR}[Z_d]^2}$. Thus, the lower bound on correctly choosing X is monotonically increasing in the SNR of our proxy measurement. Using this probabilistic framework, we can now see how some of the empirical behaviors in Section 2 might arise.

3.1 Stochastic block models

In the stochastic block model, the nodes are partitioned into K *blocks*, and for parameters $P_{i,j}$ (with $1 \leq i, j \leq K$), each node in block i is connected to each node in block j independently with probability $P_{i,j}$. (Since our graphs are undirected, $P_{i,j} = P_{j,i}$.) In our core-fringe model here, $K = 4$, the core corresponds to blocks 1 and 2, and the fringe corresponds to blocks 3 and 4. This model turns out to be flexible enough to demonstrate a wide range of

behaviors observed in Section 2. We use the following notation for block probabilities:

$$P = \begin{bmatrix} p & q & r & s \\ q & p & s & r \\ r & s & 0 & 0 \\ s & r & 0 & 0 \end{bmatrix}. \tag{7}$$

Our assumptions on the probabilities are that $q < p$ and $s \leq r$. We also assume that the first two blocks each contain n_c nodes and that the last two blocks each contain n_f nodes.

We further assume we are given samples of four distinct nodes u, v, w, and z chosen uniformly at random from the core blocks such that u, v, and w are in block 1 and z is in block 2. Following our notation above, let X be the random variable that (u, v) is an edge and Y be the random variable that (w, z) is an edge ($\Pr[X] > \Pr[Y]$ since $p > q$). Our proxy measurements $\hat{X}$ and $\hat{Y}$ are the number of common neighbors of candidate edges $\{u, v\}$ and $\{w, z\}$. Our algorithm will correctly predict that (u, v) is more likely if $\hat{X} > \hat{Y}$.

Our proxy measurements are parameterized by the amount of fringe information that they incorporate. Here, we arbitrarily order the nodes in the two equi-sized fringe blocks and say that the random variable $\hat{X}_d$ is the number of common neighbors between nodes u and v when including the first d nodes in both fringe blocks. Similarly, the random variable $\hat{Y}_d$ is the number of common neighbors of nodes w and z. By independence of the edge probabilities, some straightforward calculations show that

$$\mathbb{E}[\hat{X}_d] = 2(n_c - 1)p^2 + dr^2 + ds^2,\ \mathbb{E}[\hat{Y}_d] = 2(n_c - 1)pq + 2drs$$
$$\mathbb{V}[\hat{X}_d] = 2(n_c - 1)p^2(1 - p^2) + dr^2(1 - r^2) + ds^2(1 - s^2)$$
$$\mathbb{V}[\hat{Y}_d] = 2(n_c - 1)pq(1 - pq) + 2drs(1 - rs).$$

With no fringe information, it is immediate that the SNR is positive, i.e., $\mathrm{SNR}[Z_0] > 0$: $\mathbb{E}[\hat{X}_0 - \hat{Y}_0] = 2(n_c - 1)p[p - q] > 0$ as $p > q$. If the two fringe blocks connect to the two core blocks with equal probability, then the SNR with no fringe is optimal.

Lemma 3.1 (No-fringe optimality). *If $r = s$ in the core-fringe SBM, then $\mathrm{SNR}[Z_d]$ decreases monotonically in d.*

Proof. When $r = s$, by independence of $\hat{X}_d$ and $\hat{Y}_d$,

$$\mathbb{E}[Z_0] = \mathbb{E}[\hat{X}_d - \hat{Y}_d] = 2(n_c - 1)p^2 + dr^2 + ds^2 - 2(n_c - 1)pq - 2drs$$
$$= 2(n_c - 1)p^2 - 2(n_c - 1)pq = \mathbb{E}[\hat{X}_0 - \hat{Y}_0],$$

and $\mathbb{V}[Z_d] = \mathbb{V}[\hat{X}_d] + \mathbb{V}[\hat{Y}_d] > \mathbb{V}[\hat{X}_0] + \mathbb{V}[\hat{Y}_0] = \mathbb{V}[Z_0]$. □

This result is intuitive. If the two fringe blocks connect to the core nodes with equal probability, then the node pairs (u, v) and (w, z) receive additional extra common neighbors according to exactly the same distributions. Thus, these fringe nodes provide noise but no signal. We confirm this result numerically with parameter settings $p = 0.5$, $q = 0.3$, $s = r = 0.2$, and $n_c = 10$ (Figure 7A). Indeed, the SNR monotonically decreases as we include more fringe.

In Section 2, we saw cases where including any additional fringe information always helped. The following lemma shows that we can set parameters in our core-fringe SBM such that including any additional fringe information always increases the SNR.

Lemma 3.2 (All-fringe optimality). *Let $r > 0$, $s = 0$ and $4(n_c - 1)(p^2 - p^4) > 1$ in the core-fringe SBM. Then $\mathrm{SNR}[Z_d]$ monotonically increases in d and $\lim_{d\to\infty} \mathrm{SNR}[Z_d] = \infty$.*

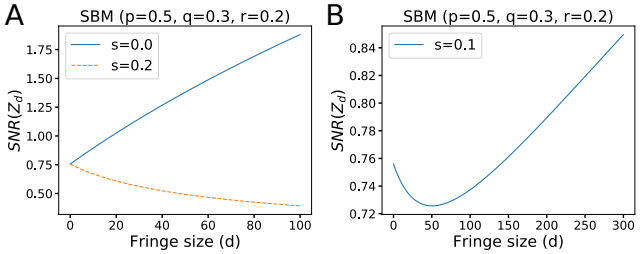


Figure 7: SNR for the difference in the common neighbors in our stochastic block model of core-fringe structure with $n_c = 10$. (A) When the fringe blocks have equal probability of connecting to the two core blocks ($r = s$), the SNR decreases monotonically with fringe size by Lemma 3.1. When fringe blocks only connect to one of the core blocks ($s = 0.0$), the SNR increases monotonically with fringe size Lemma 3.2. (B) For an intermediate parameter setting, including the fringe hurts the SNR until enough fringe is included, at which point the SNR increases monotonically (Lemma 3.3).

Proof. We have that

$$\mathrm{SNR}[Z_d] = \frac{2(n_c-1)p(p-q)+dr^2}{\sqrt{2(n_c-1)(p^2(1-p^2)+pq(1-pq))+dr^2(1-r^2)}}.$$

We can treat this function as continuous in d. Then

$$\lim_{d\to\infty} \mathrm{SNR}[Z_d] = \lim_{d\to\infty} \frac{dr^2}{\sqrt{dr^2(1-r^2)}} = \infty.$$

Similarly, we can compute the derivative with respect to d:

$$\frac{\partial}{\partial d}\mathrm{SNR}[Z_d] = \frac{r^2\sqrt{\mathbb{V}[Z_d]} - \frac{1}{2}r^2(1-r^2)/\sqrt{\mathbb{V}[Z_d]}}{\mathbb{V}[Z_d]}$$

The derivative is positive provided that $r^2\mathbb{V}[Z_d] > \frac{1}{2}r^2(1 - r^2)$, which is true if $\mathbb{V}[Z_0] > \frac{1}{2}$ since $\mathbb{V}[Z_0]$ is monotonically increasing in d. We have that

$$\mathbb{V}[Z_0] = 2(n_c - 1)(p^2(1 - p^2) + pq(1 - pq)) > 2(n_c - 1)(p^2 - p^4).$$

Thus, the result holds provided $4(n_c - 1)(p^2 - p^4) > 1$. □

The result is again intuitive. By setting $s = 0$, the pair of nodes in different blocks (w and z) get no additional common neighbors from the fringe, whereas the pair of nodes in the same block (u and v) get additional common neighbors. This should only help our prediction performance, which is why the SNR monotonically increases. We confirm this result numerically (Figure 7A), where we use the same parameters as the experiment described above. We can also check that the conditions of Lemma 3.2 are met: $n_c = 10$ and $p = 0.5$, so $4(n_c - 1)(p^2 - p^4) = 6.75 > 1$.

The SBM can also exhibit additional behaviors that we observed in Section 2. For example, with the email-Enron-4 dataset, prediction performance initially decreased as we included more fringe and then began to increase (Figures 3D and 3H). The following lemma says that the core-fringe SBM can capture this behavior.

Lemma 3.3 (Enough-fringe optimality). *Let p, q, and r be given. Then there exists a value of s in the core-fringe SBM such that $\mathrm{SNR}[Z_d]$ initially decreases and then increases without bound.*

Proof. To simplify notation, consider the following constants: $\alpha = \mathbb{E}[Z_0] = 2(n_c - 1)(p^2 - pq)$, $\beta = (r - s)^2$, $\gamma = \mathbb{V}[Z_0] = 2(n_c - 1)[p^2(1-p^2)+pq(1-pq)]$, and $\delta = r^2(1-r^2)+s^2(1-s^2)+2rs(1-rs)$.

With this notation, $\text{SNR}[Z_d] = (\alpha + \beta d)/\sqrt{\gamma + \delta d}$. Treating this as a continuous function in d, the derivative is: $\frac{\partial}{\partial d}\text{SNR}[Z_d] = (-\alpha\delta + 2\beta\gamma + \beta\delta d)/(2(\gamma + \delta d)^{3/2})$. For any s, we can choose a sufficiently large D such that the derivative is positive when $d > D$, meaning that $\text{SNR}[Z_d]$ is increasing. Furthermore, $\text{SNR}[Z_d]$ grows as $O(\sqrt{d})$. It is easy to check that the derivative also has at most one root, $d_0 = \alpha/\beta - 2\gamma/\delta$. We claim that d_0 can be made as large as desired. By setting s sufficiently close to r, β approaches 0, while δ is bounded away from 0. The remaining terms are positive constants. Finally, when $d = 0$, the value of the derivative is $(-\alpha\delta + 2\beta\gamma)/(2\gamma^{3/2})$. Again, we can make s sufficiently close to r so that β approaches 0 and the derivative is negative at $d = 0$. Therefore, there exists an s such that its derivative has one root $d_0 \geq 1$, $\text{SNR}[Z_d]$ decreases for small enough d and eventually increases without bound. □

By setting $n_c = 10$, $p = 0.5$, $q = 0.3$, $r = 0.2$, and $s = 0.1$, we see the behavior described by Lemma 3.3—the SNR initially decreases with additional fringe but then increases monotonically (Figure 7B).

By extending the SBM to include a third fringe block, we can also have a case where an intermediate amount of core is optimal. We argue informally as follows. We begin with a setup as in Lemma 3.2, where $s = 0$. Including all of the fringe available in these blocks is optimal. We then add a third fringe block that connects with equal probability to the two core blocks. By the arguments in Lemma 3.1, this only hurts the SNR. Thus, it is optimal to include two of the three fringe blocks, which is an intermediate amount of fringe.

3.2 Small-world lattice models

In the one-dimensional small-world lattice model [18], there is a node for each integer in $\mathbb{Z}$ and a parameter $\alpha \geq 0$. The probability that edge (i, j) exists is

$$\Pr[(i,j) \in E] = \tfrac{1}{|j-i|^\alpha}. \tag{8}$$

We start with a core of size $2c + 1$, centered around 0: $V_0 = C = \{-c, \ldots, c\}$. We then sample two nodes v and w such that

$$u = -c < v < w < c = z \text{ and } 2 \leq v - u < z - w. \tag{9}$$

In our language at the beginning of Section 3, X is still the random variable that edge (u, v) exists and Y is the random variable that edge (w, z) exists. By our assumptions and Eq. (8), we know that $\Pr[X] > \Pr[Y]$. However, we will again assume that we are only given access to the number of common neighbors through the proxy random variables $\hat{X}$ and $\hat{Y}$.

Our parameterization of the proxy measurements are a distance d that we examine beyond the core. Specifically, the nested sequence of vertex sets that incorporate fringe information is given by $V_d = \{-(c + d), \ldots, c + d\}$, and our proxy measurements are

$$\hat{X}_d = |\{s \in V_d \mid (u, s) \text{ and } (v, s) \text{ are edges}\}|$$
$$\hat{Y}_d = |\{s \in V_d \mid (w, s) \text{ and } (z, s) \text{ are edges}\}|.$$

We will analyze the random variable $Z_d = \hat{X}_d - \hat{Y}_d$. We correctly predict that (u, v) is more likely than (w, z) to exist if $Z_d > 0$. As argued above, our goal is to find a d that maximizes $\text{SNR}[Z_d]$.

We focus our analysis on the case of $\alpha = 1$ in Eq. (8). Let A_s be the indicator random variable that node s is a common neighbor of nodes u and v, for $s \in \mathbb{Z}\backslash\{u, v\}$, and let B_s be the indicator random variable that node s is a common neighbor of w and z, for $s \in \mathbb{Z}\backslash\{w, z\}$. Since $u = -c$ and $z = c$, our proxy measurements are

$$\hat{X}_d = \textstyle\sum_{s=-(c+d)}^{-(c+1)} A_s + \sum_{s=-c+1}^{v-1} A_s + \sum_{s=v+1}^{c+d} A_s \tag{10}$$

$$\hat{Y}_d = \textstyle\sum_{s=-(c+d)}^{w-1} B_s + \sum_{s=w+1}^{c-1} B_s + \sum_{s=c+1}^{c+d} B_s \tag{11}$$

Define the independent indicator random variables $I_{s,r}$ and $J_{s,r}$ where $\Pr[I_{s,r} = 1] = 1/(s(s + r))$ and $\Pr[J_{s,r} = 1] = 1/(s(r - s))$. Now we can re-write the expressions for $\hat{X}_d$ and $\hat{Y}_d$ as follows:

$$\hat{X}_d = \textstyle\sum_{s=1}^{d} I_{s,v-u} + \sum_{s=1}^{c+d-v} I_{s,v-u} + \sum_{s=1}^{v-u-1} J_{s,v-u} \tag{12}$$

$$\hat{Y}_d = \textstyle\sum_{s=1}^{d} I_{s,z-w} + \sum_{s=1}^{w-c-d} I_{s,z-w} + \sum_{s=1}^{z-w-1} J_{s,z-w}. \tag{13}$$

The expectations are given by

$$\mathbb{E}[\hat{X}_d] = \textstyle\sum_{s=1}^{d} \frac{1}{s(s+v-u)} + \sum_{s=1}^{c+d-v} \frac{1}{s(s+v-u)} + \sum_{s=1}^{v-u-1} \frac{1}{s(v-u-s)}$$
$$\mathbb{E}[\hat{Y}_d] = \textstyle\sum_{s=1}^{d} \frac{1}{s(s+z-w)} + \sum_{s=1}^{w-c-d} \frac{1}{s(s+z-w)} + \sum_{s=1}^{z-w-1} \frac{1}{s(z-w-s)}.$$

With these expressions, we can now analyze how Z_d behaves as we vary d. The following lemma establishes that Z_d converges to a positive value. Later, we use this to show that the SNR also converges to a positive value.

Lemma 3.4. $\lim_{d\to\infty} \mathbb{E}[Z_d] = Z^* > 0$.

Proof. Let $a = v - u$. Then $\lim_{d\to\infty} \mathbb{E}[\hat{X}_d] = 2\sum_{s=1}^{\infty} \frac{1}{s(s+a)} + \sum_{s=1}^{a-1} \frac{1}{s(a-s)} = 2(\psi(a + 1) + \psi(a) + 2\gamma)/a = 2(2\psi(a) + 1/a + 2\gamma)/a$, where $\psi(\cdot)$ is the digamma function. Similarly, if $b = z - w$, then

$$\lim_{d\to\infty} \mathbb{E}[\hat{Y}_d] = 2(2\psi(b) + 1/b + 2\gamma)/b.$$

Thus, $Z^* = \lim_{d\to\infty} \mathbb{E}[\hat{X}_d] - \mathbb{E}[\hat{Y}_d]$ exists, and $Z^* > 0$ if and only if

$$b(\psi(a) + 1/(2a) + \gamma) - a(\psi(b) + 1/(2b) + \gamma) > 0 \tag{14}$$

Recall that by Eq. (9), $2 \leq a < b$. Numerically, Eq. (14) holds for $(a, b) = (2, 3)$. Since the left-hand-side monotonically increases in b, this inequality holds for $a = 2$.

Now assume $b > a \geq 3$. The Puiseux series expansion of ψ at ∞ gives $\psi(x) + 1/(2x) \in \log(x) \pm \frac{1}{12(x^2-1)}$. Thus, it is sufficient to show that $b(\log(a) - 1/96) + (b - a)\gamma > a(\log(b) + 1/180)$, or that $0.99b\log(a) + \gamma > 1.01a\log(b)$, which holds for $b > a \geq 3$. □

The next theorem shows that the signal-to-noise ratio converges to a positive value. Thus, by measuring enough fringe, our proxy measurements are at least providing the correct direction of information. However, the SNR converges, so at some point, our information saturates.

Theorem 3.5 (SNR saturation). $\lim_{d\to\infty} \text{SNR}[Z_d] = S^* > 0$.

Proof. By Lemma 3.4, $\mathbb{E}[Z_d]$ converges to a positive value. Thus, it is sufficient to show that $\mathbb{V}[Z_d]$ converges. Following Eq. (12),

$$\mathbb{V}[\textstyle\sum_{s=1}^{\infty} I_{s,v-u}] = \sum_{s=1}^{\infty} \mathbb{V}[I_{s,v-u}] = \sum_{s=1}^{\infty} \frac{1}{s(s+v-u)} - \frac{1}{(s(s+v-u))^2}$$

by independence, and converges. □

The random variable $W_{d+1,J} \triangleq Z_{d+J} - Z_d = \sum_{k=d+1}^{k=d+1+J} I_{k,v-u} - I_{k,z-w}$ will be useful for our subsequent analysis. This is the additional measurement available to us if we measured at distance $d + J$ instead of d. The next lemma says that, as we increase d, the expectation of $W_{d,J}$ goes to zero faster than its variance.

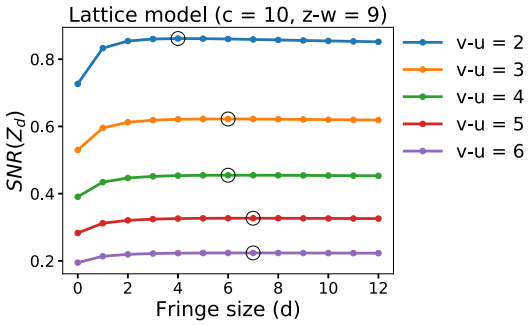


Figure 8: Saturation and interior optima of the SNR the one-dimensional small-world lattice model. Nodes v and w are sampled from $\{u = -c, \ldots, c = z\}$ with $2 \leq v - u < z - w$. The random variable Z_d is the difference in the number of common neighbors of $\{u, v\}$ and $\{w, z\}$ on the node set $\{-(c + d), \ldots, c + d\}$. Since $\text{SNR}[Z_1] > \text{SNR}[Z_0]$, an intermediate amount of fringe produces the optimal SNR (Corollary 3.8); these optima are circled in black. The SNR converges by Theorem 3.5, and we indeed see the consequent saturation.

Lemma 3.6. *For any J, $\lim_{d\to\infty} \mathbb{E}[W_{d,J}]/\mathbb{V}[W_{d,J}] = 0$.*

Proof. Let $a = v - u$ and $b = z - w$. By independence,

$$\mathbb{E}[W_{d,J}] = \sum_{k=d}^{k=d+J} \frac{1}{k(k+a)} - \frac{1}{k(k+b)} = \sum_{k=d}^{k=d+J} \frac{b-a}{k(k+a)(k+b)} = O(\sum_{k=d}^{k=d+J} 1/k^3).$$

For large enough d, $\mathbb{V}[W_{d,J}] = \sum_{k=d}^{k=d+J} \frac{1}{k(k+a)} + \frac{1}{k(k+b)} - \frac{1}{(k(k+b))^2} - \frac{1}{(k(k+a))^2}$, which is $O(\sum_{k=d}^{k=d+J} 1/k^2)$. □

The next theorem now shows that in the one-dimensional lattice model, the signal-to-noise ratio eventually begins to decrease. This means that at some point, the noise overwhelms the signal. Thus, it will never be best to gather as much fringe as possible.

Theorem 3.7. *There exists a D for which $\text{SNR}[Z_D] > \text{SNR}[Z_{D+j}]$ for any $j > 0$.*

Proof. We have that $\text{SNR}[Z_d] > \text{SNR}[Z_{d+j}]$ if and only if

$$\frac{\mathbb{E}[Z_d]}{\sqrt{\mathbb{V}[Z_d]}} > \frac{\mathbb{E}[Z_d + W_{d+1,j}]}{\sqrt{\mathbb{V}[Z_d + W_{d+1,j}]}} \iff \frac{\mathbb{E}[Z_d]^2}{\mathbb{V}[Z_d]} > \frac{2\mathbb{E}[Z_d]\mathbb{E}[W_{d+1,j}] + \mathbb{E}[W_{d+1,j}]^2}{\mathbb{V}[W_{d+1,j}]}.$$

The second inequality above comes from squaring both sides of the first inequality; both terms in the first inequality are positive for large enough d by Lemma 3.4, so we can keep the direction of the inequality. By Theorem 3.5, the left-hand-side of the inequality converges to a positive constant. We claim that the right-hand-side converges to 0.

By Lemma 3.4, $\mathbb{E}[Z_d]$ converges, so it must be bounded by a positive constant constant. Furthermore, since $Z_{d+j} = Z_d + W_{d+1,j}$, we have that $\mathbb{E}[W_{d+1,j}] \to 0$. Combining these results, $2\mathbb{E}[Z_d]\mathbb{E}[W_{d+1,j}] + \mathbb{E}[W_{d+1,j}]^2 = O(\mathbb{E}[W_{d+1,j}])$, and we have that $\mathbb{E}[W_{d+1,j}]/\mathbb{V}[W_{d+1,j}] \to 0$ by Lemma 3.6. □

A consequence of this theorem is that if the SNR initially increases, then an intermediate amount of fringe is optimal. The reason is that the SNR initially increases but at some point begins to decrease monotonically (Theorem 3.7) before converging to a positive value (Theorem 3.5). We formalize this as follows.

Corollary 3.8 (Intermediate-fringe optimality). *If $\text{SNR}[Z_0] < \text{SNR}[Z_1]$, then $d^* = \arg\max_d \text{SNR}[Z_d]$ satisfies $0 < d^* < \infty$.*

Numerically, $\text{SNR}[Z_0] < \text{SNR}[Z_1]$ in several cases (Figure 8). In this experiment, we fix $c = 10$ and $w = 1$ (so $z - w = 9$) and vary the amount of fringe from 0 to 12 nodes on either end of the core. We also vary v so that $v - u \in \{2, 3, 4, 5, 6\}$. We observe two phenomena consistent with our theory. First, by Corollary 3.8, an intermediate amount of fringe information should be optimal; indeed, this is the case. Second, by Theorem 3.5, the SNR converges, indicating that saturation should kick in at some finite fringe size. This is true in our experiments, where saturation occurs after around $d = 8$.

4 DISCUSSION

Link prediction is a cornerstone problem in network science [24, 26], and the models for prediction include those that are mechanistic [5], statistical [8], or implicitly captured by a principled heuristic [3, 4]. The major difference in our work is that we explicitly study the consequences of a common dataset collection process that results in core-fringe structure. Most related to our analysis of random graph models are theoretical justifications of principled heuristics such as the number of common neighbors in latent space models [34] and in general stochastic block models [33].

The core-fringe structure that we study can be interpreted as an extreme case of core-periphery structure in complex networks [7, 16, 31, 37]. In more classical social and economic network analysis, core-periphery structure is a consequence of differential status [11, 22]. In this paper, the structure emerges from data collection mechanisms, which raises new research questions of the kind that we have addressed. However, our results hint that periphery nodes could also be noisy sources of information and possibly warrant omission in standard link prediction. Our fringe measurements can also be viewed as adding noisy training data, which is related to training data augmentation methods [29, 30].

Conventional machine learning wisdom says that more data generally helps make better predictions. We showed that this is far from true in the common problem of network link prediction, where additional data comes from observing how some core set of nodes interacts with the rest of the world, inducing core-fringe structure. Our empirical results show that the inclusion of additional fringe information leads to substantial variability in prediction performance with common link prediction heuristics. We observed cases where fringe information is (i) always harmful, (ii) always beneficial, (iii) beneficial only *up to* a certain amount of collection, and (iv) beneficial only with *enough* collection.

At first glance, this variability seems difficult to characterize. However, we showed that these behaviors arise in some simple graph models—namely, the stochastic block model and the one-dimensional small-world lattice model—by interpreting the benefit of the fringe information as changing the signal-to-noise ratio in our prediction problem. Our datasets are certainly more complex than these models, but our analysis suggests that variability in prediction performance when incorporating fringe data is much more plausible than one might initially suspect. Even when fringe

data is available in network analysis, we must be careful how we incorporate this data into the prediction models we build.

Acknowledgements

We thank David Liben-Nowell for access to the LiveJournal data and Jure Leskovec for access to the email-Eu data. This research was supported by NSF Awards DMS-1830274, CCF-1740822, and SES-1741441; a Simons Investigator Award; and ARO Award W911NF-19-1-0057.

REFERENCES

[1] Emmanuel Abbe. 2018. Community Detection and Stochastic Block Models: Recent Developments. *Journal of Machine Learning Research* 18, 177 (2018), 1–86. http://jmlr.org/papers/v18/16-480.html

[2] Emmanuel Abbe, Afonso S. Bandeira, and Georgina Hall. 2016. Exact Recovery in the Stochastic Block Model. *IEEE Transactions on Information Theory* 62, 1 (2016), 471–487. https://doi.org/10.1109/tit.2015.2490670

[3] Lada A Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social networks* 25, 3 (2003), 211–230.

[4] Lars Backstrom and Jure Leskovec. 2011. Supervised Random Walks: Predicting and Recommending Links in Social Networks. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*. ACM, 635–644. https://doi.org/10.1145/1935826.1935914

[5] A.L Barabási, H Jeong, Z Néda, E Ravasz, A Schubert, and T Vicsek. 2002. Evolution of the social network of scientific collaborations. *Physica A: Statistical Mechanics and its Applications* 311, 3-4 (2002), 590–614. https://doi.org/10.1016/s0378-4371(02)00736-7

[6] Austin R. Benson and Jon Kleinberg. 2018. Found Graph Data and Planted Vertex Covers. In *Advances in Neural Information Processing Systems*.

[7] Stephen P Borgatti and Martin G Everett. 2000. Models of core/periphery structures. *Social Networks* 21, 4 (2000), 375–395. https://doi.org/10.1016/s0378-8733(99)00019-2

[8] Aaron Clauset, Cristopher Moore, and M. E. J. Newman. 2008. Hierarchical structure and the prediction of missing links in networks. *Nature* 453, 7191 (2008).

[9] Nick Craswell, Arjen P de Vries, and Ian Soboroff. 2005. Overview of the TREC 2005 Enterprise Track. In *TREC*, Vol. 5. 199–205.

[10] Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. 2011. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E* 84, 6 (2011). https://doi.org/10.1103/physreve.84.066106

[11] Patrick Doreian. 1985. Structural equivalence in a psychology journal network. *Journal of the American Society for Information Science* 36, 6 (1985), 411–417. https://doi.org/10.1002/asi.4630360611

[12] Nathan Eagle and Alex (Sandy) Pentland. 2005. Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing* 10, 4 (2005), 255–268. https://doi.org/10.1007/s00779-005-0046-3

[13] Amir Ghasemian, Homa Hosseinmardi, and Aaron Clauset. 2018. Evaluating overfit and underfit in models of network community structure. *arXiv:1802.10582* (2018).

[14] Ashish Goel, Aneesh Sharma, Dong Wang, and Zhijun Yin. 2013. Discovering similar users on Twitter. In *11th Workshop on Mining and Learning with Graphs*.

[15] Pankaj Gupta, Ashish Goel, Jimmy Lin, Aneesh Sharma, Dong Wang, and Reza Zadeh. 2013. WTF: the who to follow service at Twitter. In *Proceedings of the 22nd international conference on World Wide Web*. ACM Press. https://doi.org/10.1145/2488388.2488433

[16] Petter Holme. 2005. Core-periphery organization of complex networks. *Physical Review E* 72, 4 (2005). https://doi.org/10.1103/physreve.72.046111

[17] Myunghwan Kim and Jure Leskovec. 2011. The Network Completion Problem: Inferring Missing Nodes and Edges in Networks. In *Proceedings of the SIAM Conference on Data Mining*. Society for Industrial and Applied Mathematics, 47–58. https://doi.org/10.1137/1.9781611972818.5

[18] Jon Kleinberg. 2006. Complex Networks and Decentralized Search Algorithms. In *Proceedings of the International Congress of Mathematicians*.

[19] Bryan Klimt and Yiming Yang. 2004. The Enron Corpus: A New Dataset for Email Classification Research. In *Machine Learning: ECML 2004*. Springer Berlin Heidelberg, 217–226. https://doi.org/10.1007/978-3-540-30115-8_22

[20] Gueorgi Kossinets. 2006. Effects of missing data in social networks. *Social Networks* 28, 3 (2006), 247–268. https://doi.org/10.1016/j.socnet.2005.07.002

[21] Edward O Laumann, Peter V Marsden, and David Prensky. 1989. The boundary specification problem in network analysis. *Research methods in social network analysis* 61 (1989), 87.

[22] Edward O. Laumann and Franz U. Pappi. 1976. *Networks of collective action: A perspective on community influence systems (Quantitative studies in social relations)*. Academic Press.

[23] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data* 1, 1 (2007), 2–es. https://doi.org/10.1145/1217299.1217301

[24] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology* 58, 7 (2007), 1019–1031. https://doi.org/10.1002/asi.20591

[25] D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins. 2005. Geographic routing in social networks. *Proceedings of the National Academy of Sciences* 102, 33 (aug 2005), 11623–11628. https://doi.org/10.1073/pnas.0503018102

[26] Linyuan Lü and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications* 390, 6 (2011), 1150–1170. https://doi.org/10.1016/j.physa.2010.11.027

[27] Elchanan Mossel, Joe Neeman, and Allan Sly. 2014. Belief propagation, robust reconstruction and optimal recovery of block models. In *Conference on Learning Theory*. 356–370.

[28] Anatole Rapoport. 1953. Spread of information through a population with socio-structural bias I: Assumption of transitivity. *Bulletin of Mathematical Biophysics* 15, 4 (Dec. 1953), 523–533.

[29] Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment* 11, 3 (2017), 269–282. https://doi.org/10.14778/3157794.3157797

[30] Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. In *Advances in Neural Information Processing Systems*. 3567–3575.

[31] Puck Rombach, Mason A. Porter, James H. Fowler, and Peter J. Mucha. 2017. Core-Periphery Structure in Networks (Revisited). *SIAM Rev.* 59, 3 (2017), 619–646. https://doi.org/10.1137/17m1130046

[32] Daniel M. Romero, Brian Uzzi, and Jon M. Kleinberg. 2016. Social Networks Under Stress. In *Proc. International World Wide Web Conference*. 9–20. https://doi.org/10.1145/2872427.2883063

[33] Purnamrita Sarkar, Deepayan Chakrabarti, et al. 2015. The consistency of common neighbors for link prediction in stochastic blockmodels. In *Advances in Neural Information Processing Systems*. 3016–3024.

[34] Purnamrita Sarkar, Deepayan Chakrabarti, and Andrew W. Moore. 2011. Theoretical Justification of Popular Link Prediction Heuristics. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*. AAAI Press, 2722–2727. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-453

[35] Duncan J. Watts and Steven H. Strogatz. 1998. Collective dynamics of 'small-world' networks. *Nature* 393 (1998), 440–442.

[36] Hao Yin, Austin R. Benson, Jure Leskovec, and David F. Gleich. 2017. Local Higher-Order Graph Clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 555–564. https://doi.org/10.1145/3097983.3098069

[37] Xiao Zhang, Travis Martin, and M. E. J. Newman. 2015. Identification of core-periphery structure in networks. *Physical Review E* 91, 3 (2015). https://doi.org/10.1103/physreve.91.032803

Spiders like Onions: on the Network of Tor Hidden Services

Massimo Bernaschi
Institute for Applied Computing,
National Research Council of Italy
Rome, Italy
massimo.bernaschi@cnr.it

Alessandro Celestini
Institute for Applied Computing,
National Research Council of Italy
Rome, Italy
a.celestini@iac.cnr.it

Stefano Guarino
Institute for Applied Computing,
National Research Council of Italy
Rome, Italy
s.guarino@iac.cnr.it

Flavio Lombardi
Institute for Applied Computing,
National Research Council of Italy
Rome, Italy
flavio.lombardi@cnr.it

Enrico Mastrostefano
Institute for Applied Computing,
National Research Council of Italy
Rome, Italy
e.mastrostefano@iac.cnr.it

ABSTRACT

Tor hidden services allow offering and accessing various Internet resources while guaranteeing a high degree of provider and user anonymity. So far, most research work on the Tor network aimed at discovering protocol vulnerabilities to de-anonymize users and services. Other work aimed at estimating the number of available hidden services and classifying them. Something that still remains largely unknown is the structure of the graph defined by the network of Tor services. In this paper, we describe the topology of the Tor graph (aggregated at the hidden service level) measuring both global and local properties by means of well-known metrics. We consider three different snapshots obtained by extensively crawling Tor three times over a 5 months time frame. We separately study these three graphs and their shared "stable" core. In doing so, other than assessing the renowned volatility of Tor hidden services, we make it possible to distinguish time dependent and structural aspects of the Tor graph. Our findings show that, among other things, the graph of Tor hidden services presents some of the characteristics of social and surface web graphs, along with a few unique peculiarities, such as a very high percentage of nodes having no outbound links.

CCS CONCEPTS

• **Information systems** → **Web mining**; **Deep web**; *Information retrieval*; • **Mathematics of computing** → *Graph theory*.

KEYWORDS

Tor; Web Graph; Dark Web; Complex Networks

ACM Reference Format:
Massimo Bernaschi, Alessandro Celestini, Stefano Guarino, Flavio Lombardi, and Enrico Mastrostefano. 2019. Spiders like Onions: on the Network of Tor Hidden Services. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313687

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313687

1 INTRODUCTION

Research efforts on The Onion Router network (Tor) have recently flourished, focusing on evaluating its security [8], understanding its evolution [24], and discussing its thematic organization [38]. Nevertheless, the limited number of entry points to Tor on the surface Web makes it difficult to fully uncover many of Tor's characteristics. In particular, despite some insights provided over the last years [6, 15, 23, 36], few information are available regards the topology and the volatility of the Tor network. In this study we exploit three sets of crawling data collected over three different time frames to present a thorough analysis of the structural properties of the graph of Tor Hidden Services (HS) that makes it possible to tell apart persistent from variable characteristics.

Similarly to the surface WWW [12], the structure of the Tor HS Web graph can be seen as an indicator of intrinsic characteristics of the Tor network and of latent patterns of interactions among Tor users. Since the work of Watts & Strogats [40] and Barabasi & Albert [5] it has been widely recognized that the in-depth study of the properties of the underlying graph is crucial for determining behavioral and structural aspects of a complex system, and for understanding and possibly explaining the emergence of specific features in real world networks. Our efforts for identifying the distinguishing traits of the topology of the Tor Web can therefore be of great help to shed light on the usage patterns, the dynamics and the vulnerabilities of the Tor network.

The paper also addresses the evolution of the Tor Web graph showing the actual changes that take place in the quality and quantity of available services and in the persistence of their interconnections over time. In particular, we provide a rich set of results and discussions on deltas over time that allow for detailed reasoning on Tor Web connection/topological trends. To the best of our knowledge there are no similar studies on the Tor Web. Our present study, albeit limited as regards its timeframe, therefore provides useful information and hints to foster further research in the area.

The rest of the paper is organized as follows: Section 2 reviews background information and related work; Section 3 describes the methodology used for collecting data, extracting the graphs and studying their properties; Section 4 analyzes the obtained graphs and details on results; Section 5 discusses in depth our findings, in comparison with well-known graph models; finally, Section 6 draws conclusions and suggests directions for future work.

2 BACKGROUND AND RELATED WORK

Past research work on Tor has mainly been devoted to assess its vulnerabilities. However, as a positive side effect, novel data and insights on Tor services and network have been obtained.

Biryukov *et. al.* [8] in 2013 exploited a Tor vulnerability to collect all hidden service descriptors in approximately 2 days using a modest amount of resources. They found out that, while the contents of Tor hidden services is rather varied, the most popular hidden services were related to botnets. It is worth noticing that their approach cannot be reproduced, because they exploited a Tor bug that was fixed in recent versions of the software.

In a previous work [6], we leveraged automated Tor network exploration to the purpose of relating semantic content similarity with Tor topology at the page, host, and service level. The present work largely extends on both data collection and analysis over [6].

Also Ghosh *et. al.* [22] developed an automated tool to explore the Tor network and analyze the contents of onion sites. Their classification framework maps onion site content to a set of categories, and clusters services to categorize onion content. The main difference with respect to our work is that they focus on page content/semantics, and do not consider network topology. Owen *et. al.* [35], by operating 40 relays over a 6 month time frame, reported over hidden services persistence, contents, and popularity. Their aim was classifying services based on their content.

Similarly to our present work, Christin *et. al.* [15] collected crawling data on Tor hidden services over an 8 month lifespan. They evaluated the evolution/persistence of such services over time, and performed a study on the content and topology of the explored network. The main difference with our present work is that the Tor graph we explore is much larger, not being limited to a single marketplace. In addition, we present here a more in depth evaluation of the graph topology. De Domenico *et. al.* [17], used the data collected in [4] to study the topology of the Tor network . They gave a characterization of the topology of the Darknet and proposed a generative model for the Tor network to study its resilience. Their viewpoint is quite different from our own here, as they consider the network at the autonomous system (AS) level.

Very recently Griffith *et. al.* [23] performed a topological analysis of the Tor hidden services graph. They crawled the Tor network using the commercial service *scrapinghub.com*, through the *tor2web* proxy onion link. Interestingly, they reported that more than the 87% of Darkweb sites never link to another site. The main difference with our present work lies in both the extent of the explored network (we collected a much more extensive dataset than that accessible through *tor2web*) and the depth of the analysis of the network itself (we evaluate a larger set of network characteristics).

Differently from the Literature on the Tor network, the topology of the WWW has been the subject of a large number of studies in the past. In this paper we also aim at comparing Tor network characteristics with those of the surface Web, briefly surveyed below.

Among random graph models suitable for the surface Web [28],[10] Kleinberg in particular [26] introduced algorithms for improved Web search and automatic community discovery, thus providing one of the first publicly known portraits of the Web graph. Kleinberg also stressed that traditional random graph models, such as the well-known Erdős-Rényi model [19] do not exhibit many properties of the Web graph. Among other studies on the WWW, Broder *et. al.* [12] discovered the heavy-tailed distribution of node degrees, claiming a power-law distribution, and the presence of large hubs along with a peculiar structure of the graph they called *bow-tie*. Bearing in mind that the adopted crawling technique affects the structure of the results, subsequent studies have somewhat reached different results, [1, 18, 37]. In a recent work Meusel *et. al.* [30] analyzed the structure of the WWW at different levels: pages, hosts and pay-level domain. The last aggregation level can be seen as the Tor service level and we will discuss similarities and differences of their results compared to our findings on the Tor HS graph.

3 METHODOLOGY

In this Section we describe the methodology used for collecting data, building the graphs and studying their properties.

3.1 Data Collection

We aim at characterizing the portion of the Tor Web that can be accessed by using a custom web scraping procedure. Specifically, we assembled a large root set by merging onion urls advertised on well-known Tor wikis and link directories (*e.g.*, "The Hidden Wiki" [1]), or obtained from standard (*e.g.*, Google) and Tor-specific (*e.g.*, Ahmia) search engines. Then, in the 5-month time frame between January 2017 and May 2017, we launched our customized crawler three times and let each execution run for about six weeks. Thus, we obtained three different "snapshots" of the Tor Web, denoted SNP1, SNP2, and SNP3, respectively.

The numbers of our datasets, reported in Table 1, are comparable to – and, as a matter of fact, greater than – similar studies in the Literature [7, 23, 38]. Yet, if we refer to the statistics provided by the Tor Project for the corresponding time window [2], our crawls only reached 25% to 35% of the total number of daily published hidden services. It is not clear to which extent those estimates are inflated by the existence of Tor-specific messaging services in which each user is identified by a unique onion domain [23] and by hidden services that do not host websites. In any case, reaching all active onion urls is not arguably possible with ordinary resources [3] and, to the best of our knowledge, the present study is the widest exploration of the Tor Web performed so far.

To access the Tor network and to collect data from hidden services we evaluated different crawlers. In particular, we evaluated the following alternatives: Apache Nutch [4] [25], Heritrix [5] [31] and BUbiNG [9]. By considering criteria such as performance, configurability and extensibility, we found BUbiNG to be the most appropriate choice for our goals. BUbiNG is a high-performance, scalable, distributed, open-source crawler, written in Java, and developed by the Laboratory for Web Algorithmics (LAW) part of the Computer Science Department of the University of Milan. To allow

[1] wikitjerrta4qgz4.onion
[2] https://metrics.torproject.org/hidserv-dir-onions-seen.html?start=2017-01-01&end=2017-05-01
[3] Tor's working principles make it possible to run a hidden service whose existence is only known to the relays where the introductory points of that service are published [34].
[4] http://nutch.apache.org
[5] https://webarchive.jira.com/wiki/display/Heritrix

Table 1: Outcomes of the three crawling processes.

Crawl	End Date	# records per response type				
		2xx	3xx	4xx	5xx	Total
SNP1	22/02/17	1821842	277813	197128	141205	2437989
SNP2	10/04/17	2339718	471519	262403	324552	3398192
SNP3	22/05/17	765876	393018	105406	67115	1331415

A status code 3xx is related to Web redirection (https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html). Status codes 4xx and 5xx are error codes.

BUbiNG to operate in the Tor network (instead of the surface Web), we used a HTTP Proxy configured with the SOCKS Proxy provided by Tor. After testing some alternatives we chose privoxy [6]. In particular, we decided not to use polipo [7] that is often used in combination with Tor, because it is no longer maintained and seemed unable to correctly manage the format of some HTTP responses. During the crawling phase we observed that some hidden services check the user-agent of the requester and, if it does not match the last version of the Tor Web browser, they reply with an error. This behavior had to be taken into account when collecting data, to allow the crawler to reach the largest possible portion of hidden services. Another issue that raised during the crawling is the load of the Tor client, *i.e.*, the software used to access Tor. We noticed that under stress (*i.e.*, when too many requests are performed in parallel), the Tor client, quite often, does not respond correctly, *i.e.*, it may mistakenly report that a hidden service is not available, even if the service is actually up and running. The maximum load depends on the specifications of the machine where the software runs, and we assessed it for our configuration during the experimental phase.

3.2 Graph Extraction

For each one of the three snapshots we extracted the associated directed Tor Service Graph (SG) aggregating pages at the service level. To extract the graph we used the Graph Builder module of the toolkit presented in [14]. In the SG, each node represents the set of pages belonging to a hidden service, *i.e.*, a Tor domain identified by a sequence of 16 characters (base32 encoded).[8] In the SG an edge connecting one hidden service to another represents the existence of at least one page of the first service that contains a hypertextual link to any page of the second service. Since we only analyzed onion links, all surface web services and all edges from/to the surface web have been ignored and have not been included in the graphs. We believe including surface web nodes/links would have introduced a bias in the Tor network analysis, due to the large difference in scale between the two underlying graphs. Even just introducing surface border nodes would have affected the analysis and it would not have added relevant information.

[6] https://www.privoxy.org
[7] https://www.irif.fr/ jch/software/polipo/
[8] In October/November 2017 a new generation of hidden services was introduced and supported by the Tor browser. They are identified by character sequences of length 56 instead of the usual 16 (https://blog.torproject.org/tors-fall-harvest-next-generation-onion-services). This change was introduced after our data collection period ended in May 2017.

To further clarify how we built the graphs, let us consider the following example, depicted in Figure 1. We found the hidden service:

`duskgytldkxiuqc6.onion`

to host three pages:

`duskgytldkxiuqc6.onion/comsense.html`;
`duskgytldkxiuqc6.onion/fedpapers/`;
`duskgytldkxiuqc6.onion/fedpapers/federa02.htm`.

To represent these resources we use a single node in the SG.

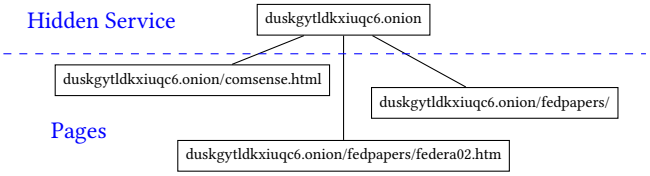


Figure 1: An example of graph construction

Besides the SGs of the three snapshots acquired with our crawls, we considered a fourth graph representing Tor's "stable core". It corresponds to the communal subgraph of SNP1, SNP2 and SNP3 induced by the edges that appear in all the three graphs.

3.3 Graph Analysis

As a first step towards the understanding of Tor dynamics, we compare macroscopic features of the four graphs to assess the persistence of Tor hidden services and of their connections. Next, we characterize the four graphs on both a global and on a local scale. Specifically, for each graph:

- We compute a set of global metrics, including measures of centralization, transitivity and efficiency.
- We extract the in- and out-degree distribution and assess whether these distributions follow a power law.
- We count the number of strongly connected components and characterize the giant strongly connected component (LSCC in the following).
- We consider several centrality measures, draw their distribution and match them with one another.
- We provide and analyze a bow-tie decomposition of the graphs under study.

Based on all gained pieces of information, we infer the general structure of the four graphs, spotting differences and highlighting common aspects that may be assumed to define the topology of Tor hidden services. Additionally, we identify a small set of hidden services that seem to play an especially important role in the graph and, through direct examination, we aimed at explaining why. All symbols and metrics used in the paper are summarized in Table 2.

4 RESULTS

In this Section we report and briefly comment the results of our analysis.

4.1 Services Persistence

As showed by other studies [7, 8, 35], there is a huge variability in the persistence of Tor hidden services. This must be carefully taken

Table 2: Notations and definitions used throughout the present paper.

Symbol	Definition
$G = (V, E)$	Graph with vertex set V and edge set E
N	Number of nodes: $N = \|V\|$
M	Number of edges: $M = \|E\|$
D	Density: $\frac{M}{N(N-1)}$
δ_{in}	Minimum in-degree
Δ_{in}	Maximum in-degree
δ_{out}	Minimum out-degree
Δ_{out}	Maximum out-degree
$\langle \deg \rangle$	Average in/out-degree
ρ	Assortativity: see (26) in [33]
C_{in}	In-degree centralization: $\frac{N*\Delta_{in}-\sum_{v\in V} \deg_{in}(v)}{(N-1)^2}$
C_{out}	Out-degree centralization: $\frac{N*\Delta_{out}-\sum_{v\in V} \deg_{out}(v)}{(N-1)^2}$
C	Global clustering coefficient: $\frac{\#\text{ closed triplets}}{\#\text{ all triplets}}$
T_1	Global transitivity of type 1: $\frac{\#(u,v,w)\text{: } u\to v\wedge v\to w\wedge w\to u}{\#(u,v,w)\text{: } u\to v\wedge v\to w}$
T_2	Global transitivity of type 2: $\frac{\#(u,v,w)\text{: } u\to v\wedge u\to w\wedge (v\to w\vee w\to v)}{\#(u,v,w)\text{: } u\to v\wedge u\to w}$
σ_{vu}	Number of shortest paths from v to u
$\sigma_{vu}(t)$	Number of shortest paths from v to u including t
$dist(v,u)$	Shortest path length from v to u
d	Diameter: $\max_{v\in V} \max_{u\in V} dist(v,u)$
r_{in}	Radius with in-paths: $\min_{v\in V} \max_{u\in V} dist(u,v)$
r_{out}	Radius with out-paths: $\min_{v\in V} \max_{u\in V} dist(v,u)$
$\langle dist \rangle$	Average shortest path length
E_{in}	Global efficiency with in-paths: $\frac{1}{N(N-1)} \sum_{u<v\in V} \frac{1}{d(v,u)}$
E_{out}	Global efficiency with out-paths: $\frac{1}{N(N-1)} \sum_{u<v\in V} \frac{1}{d(u,v)}$
B	Betweenness centrality: $B(v) = \sum_{s\neq v\neq t\in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$
PR	PageRank: see [21]

into consideration in any attempt of characterizing the topology of the Tor Web graph, because by scraping the Tor network we only obtain a snapshot of the hidden services that were active at the time the crawler issued a connection request. More generally, even for the surface Web there is evidence that the crawling process may affect the structure of the extracted graph, leading to incomplete or wrong conclusions [29]. We therefore repeated the entire data collection phase three times over five months in an effort to reduce variance and guarantee the consistency of our results. As a side benefit, we are able to further assess the renowned volatility of Tor hidden services, other than possibly telling apart time dependent from structural features of the Tor web graph.

As a starting point, Figure 2 shows how the total body of accessed hidden services is distributed across the three snapshots. SNP2 clearly emerges as the largest dataset, but this is not surprising if we compare our results with the statistics provided by the Tor Project [9], that show a spike of published hidden services in the second half of March 2017. We note that 10685 hidden services were found by all three crawling runs, suggesting that these services were durably present over the considered five month time frame. This core set represents the 83.3% of services reached during SNP1, the 42.2% of services reached during SNP2 and the 61.2% of services reached during SNP3. We also see that all pairwise intersections are not empty, meaning that during our data harvesting process (at least) 1612 hidden services disappeared, 3066 new hidden services appeared, and most notably, 76 hidden services reappeared after having gone inactive at some point in time. There is also the possibility that some of the hidden services that disappeared in a snapshot were actually active but not reachable by our crawler, for instance due to all paths to those services being temporarily unavailable.

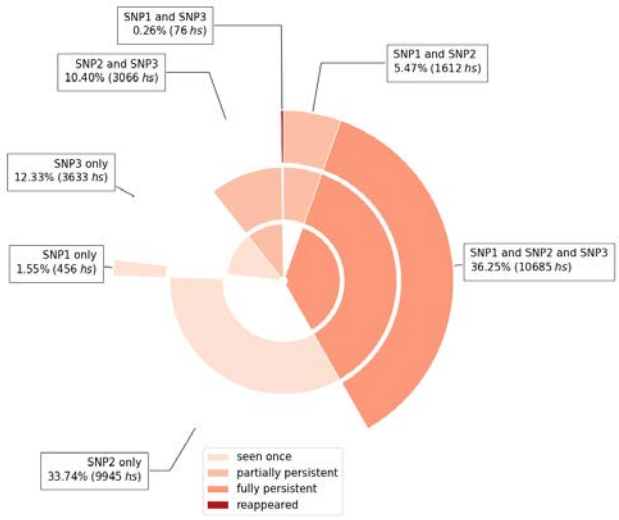


Figure 2: Services persistance over time: inner disc is SNP1, middle disc is SNP2, outer disc is SNP3.

The question now arises of whether hidden services found in two or more snapshots induce the same subgraph in the corresponding snapshots. The answer, summarized in Table 3 by looking at edge density, is no. As a consequence, if we aim at identifying the stable core of our dataset we should not just look at durable hidden services, but we must refer to durable edges. The total amount of durable edges turns out to be 28914, but somewhat surprisingly, one of these edges is isolated from the rest. The common subgraph induced by the set of stable edges consists of two weakly connected components: (i) a giant one, denoted CORE graph in the following, composed of 7669 vertices and 28913 edges, and (ii) a tiny one composed of a single edge connecting the hidden service violet77pvqdmsiy.onion to the hidden service typefacew3ijwkgg.onion.

[9] https://metrics.torproject.org/hidserv-dir-onions-seen.html?start=2017-01-01&end=2017-05-01

Table 3: Density of the subgraphs induced by different nodesets intersections in different graphs.

Nodeset	Density in SNP1	Density in SNP2	Density in SNP3
SNP1 ∩ SNP2	0.000474	0.000461	nd
SNP1 ∩ SNP3	0.000588	nd	0.000507
SNP2 ∩ SNP3	nd	0.000516	0.000435
SNP1 ∩ SNP2 ∩ SNP3	0.000595	0.000584	0.000511

4.2 Global Metrics

The global properties of our four graphs (the three snapshots plus the CORE graph) are summarized in Table 4 and in Table 5. As already mentioned, there is a significant variance in the sizes N and M of the three snapshots, which is however consistent with publicly available aggregated statistics. A common aspect is the lack of a very large hub gathering most of the connections, opposed to the presence of a single vertex that links to, respectively, 44%, 51% and 61% of the whole network. By manual inspection, we found that these onion urls are Tor link directories, not surprisingly. We also checked that the hidden services with greater in- and/or out-degree are generally persistent over the three snapshots, although their rank in the top-degree chart may change. Yet, stability is not a common property of *all* high-degree hidden services, in fact: (i) SNP3 is the only graph including a vertex with in-degree 1464 (which explains the larger C_{in} of that graph), (ii) in the CORE graph the ratios Δ_{in}/N and Δ_{out}/N are comparably smaller with respect to the snapshots, and (iii) in general all parameters strictly related to the presence of hidden services with large in- and/or out-degree (D, Δ_{in}, Δ_{out}, $\langle \deg \rangle$, ρ, C_{in} and C_{out}) appear to be variable over time. The vertex with in-degree 1464 is the hidden service *dhosting4okcs22v.onion* that is a hosting service named `Daniel's Hosting`. Tor users can get a hosting account on the server of the hidden service. The website specifies few rules regarding the contents that can be hosted for the purpose of avoiding illegal or offensive material. The same hidden service contains several sections including a link directory, but each section is registered with a different onion address, *i.e.*, a different hidden service.

Transitivity and clustering coefficients are also variable across time, but in this case the corresponding values for the CORE graph are close to the average of the three snapshots, suggesting that the variance is due to statistical fluctuations in the composition of a network with many volatile nodes. In general the overall frequency of triangles C is substantially in line with the average degree $\langle \deg \rangle$ (as we will better discuss in Section 5), with cycles (T_1) being more frequent than other types of triangles (T_2). The diameter d is stable and logarithmic in N for all graphs, and the ratio of r and d suggests a certain level of symmetry in the graph. The average shortest path length $\langle dist \rangle$ is instead consistently smaller than $\log(N)$, an important factor in determining to what extent the graph resembles a random graph (again, more details will be given in Section 5). Finally, the global efficiencies E_{in} and E_{out}, which should be comparable to $1/\langle dist \rangle$ in uniformly connected networks, are instead diluted by the fact that many pairs of nodes are disconnected (*i.e.*, have no paths connecting each other).

Table 4: Global metrics computed for the services graph of each snapshot

metrics	Graph SNP1	Graph SNP2	Graph SNP3	Graph CORE
$\|V\|$	12829	25308	17460	7669
$\|E\|$	72556	113014	103402	28913
D	0.00044	0.00018	0.00034	0.00049
δ_{in}	1	1	1	0
Δ_{in}	204	262	1464	55
δ_{out}	0	0	0	0
Δ_{out}	5603	12852	10664	2670
$\langle \deg \rangle$	5.65562	4.46554	5.92222	3.77011
ρ	-0.319	-0.32655	-0.16206	-0.37393
C_{in}	0.01546	0.01018	0.08352	0.00668
C_{out}	0.43637	0.50769	0.6105	0.34775
C	0.00943	0.00407	0.00492	0.00873
T_1	0.00617	0.00779	0.00253	0.00535
T_2	0.0039	0.0016	0.00197	0.00356
d	10	12	10	10
r_{in}	5	7	5	0
r_{out}	0	0	0	0
$\langle dist \rangle$	3.79316	4.96028	3.66455	3.98291
E_{in}	0.00549	0.00386	0.02095	0.00371
E_{out}	0.00531	0.00366	0.01965	0.00318

Table 5: Snapshot Data Details

snp	#scc	LSCC size	out-degree 0	in-degree 1
SNP1	12305	466	90.77%	17.478%
SNP2	24433	820	94.74%	43.15%
SNP3	15029	2371	83.32%	24.43%
CORE	7477	169	95.5%	25.09%

4.3 Degree and Centralities

To deepen our understanding of the structure of the Tor Web graph, we now analyze the distribution of a few metrics that quantify the importance of single vertices in the topology of the network and that can be used to gain an insight into the dynamics and the information flow in the graph. Many measures have been introduced in the last 50 years to understand who occupies critical, or *central*, positions in a network [11]. We chose to focus on the in- and out-degree, the PageRank and the betweenness centrality, which are among the most commonly used to describe real-world graphs because they respond to *semantically* different notions of "vertex centrality". The definition of PageRank and betweenness centrality is reported in Table 2.

Figures 3a and 3b show the distributions of, respectively, in-degree and out-degree for all four graphs on a log-log scale. Based on Figure 3a, the in-degree distribution seems to follow a power law decay at least for degrees lying in some intermediate range between ~ 10 and ~ 60. This is not surprising: power law degree distributions are typical in social and web networks, as we will better discuss

in Section 5. To confirm this intuition, we fitted a power law to the distribution using the statistical methods developed in [16]. In particular, we relied on the implementation provided by the POWERLAW python package [3]. POWERLAW autonomously finds a lower-bound $k_{\min}$ for degrees to be fitted, and tries to fit the whole tail unless a context-driven upper-bound $k_{\max}$ is explicitly provided by the user. We also used *Fibonacci binning* [39], as done in a previous work about the surface WWW [29], to show how the distribution looks like if a logarithmic binning is used to smoothen the tail. Two aspects of the obtained fit must be underlined: (i) for SNP2 and SNP3 the α exponent is greater than the threshold 3 that is known to control the variance of the distribution, whereas for SNP1 α is close to 2.9 and for the CORE graph it is 2.7; (ii) the $k_{\min}$ returned by POWERLAW is, respectively, 11, 16, 17 and 5, but the vertices with in-degree greater than or equal to this $k_{\min}$ are only 1624 ($\sim$ 12.67%) in SNP1, 1155 ($\sim$ 4.56%) in SNP2 and 964 ($\sim$ 5.52%) in SNP3, whereas they are 1779 ($\sim$ 23.20%) in the CORE graph, which also has a shorter tail. Summing up, the in-degree provides a further element in support of the intuition that the structure of the CORE graph differs significantly from the snapshots.

Figure 3b makes apparent that the out-degree distribution does not follow a power law. Yet, there are at least two remarkable aspects in this distribution. On the one hand, vertices of out-degree 0 weight 83% up to 95%, according to the graph (this number is reported only in the legend since $k = 0$ is cut-off by the log scale). In other words, a vast majority of Tor's hidden services do not link to any other hidden service. On the other hand, the distribution has a long tail, meaning that the large value of Δ_{out} observed in Section 4.2 is not an isolated case, but rather an evidence of a general trend. To better understand how easily the whole graph can be explored from just a few starting points, in Figure 4 we plot the cumulative percentage of the network that is at distance one from the top out-degree vertices. We see that the top-3 and top-6 out-degree hidden services suffice to reach more than 90% of the graph in just one click in SNP3 and SNP2 respectively. In SNP1 we need the top-20 out-degree hidden services to reach the same percentage of the graph, whereas with the top-6 out-degree services we reach 80% of the nodes. This phenomenon is less evident for the CORE graph, albeit 10 hidden services still contain direct links to more than 80% of the network.

Figures 3c and 3d show the distributions of, respectively, the PageRank and the betweenness centrality for all four graphs on a log-log scale. We opted for a log-log scale in order to make these distributions directy comparable with the in- and out-degree, other than with one another. This is especially important for the PageRank because it has been shown that in many real-world networks (*e.g.*, in scale-free networks) the PageRank distribution "mimics" the in-degree distribution, following a power law with very similar exponent [29]. According to the data points plotted in Figure 3c, this may not seem to be the case for the Tor Web: although it has a heavy tail the decay looks much faster than a power law. However, since PageRank is a continuous metrics, a power law decay can only be appreciated graphically when using a suitable binning. We therefore proceeded exactly as for the in-degree by using POWERLAW to fit the distribution and by applying Fibonacci binning to have a more reliable visual perspective. Unfortunately, the apparently good fit plotted in Figure 3c only regards a minimal portion of the graph: it is only valid for 723 vertices ($\sim$ 5.64%) of SNP1, 772 vertices ($\sim$ 3.05%) of SNP2, 298 vertices ($\sim$ 1.71%) of SNP3 and just 113 vertices ($\sim$ 1.47%) of the CORE graph. For what concerns the betweenness centrality, in all four graphs the long tail and the fast decay are accompanied by more than 90% of the vertices having $B(V) = 0$. This is not surprising since, by definition, all vertices having out-degree 0 must have betweennes 0. Figure 3d partially resembles Figure 3b, suggesting that due to the huge percentage of sinks and to the greater imbalance of the out-degree with respect to the in-degree, in the Tor Web the out-degree impacts on the betweenness of a hidden service more that its in-degree. To confirm or deny this intuition, and more generally to assess the level of correlation between different centrality measures, in Figure 5 we plot the pairwise comparison of (normalized) in-degree, out-degree, PageRank and betweenness centrality.

4.4 Bow-Tie Structure

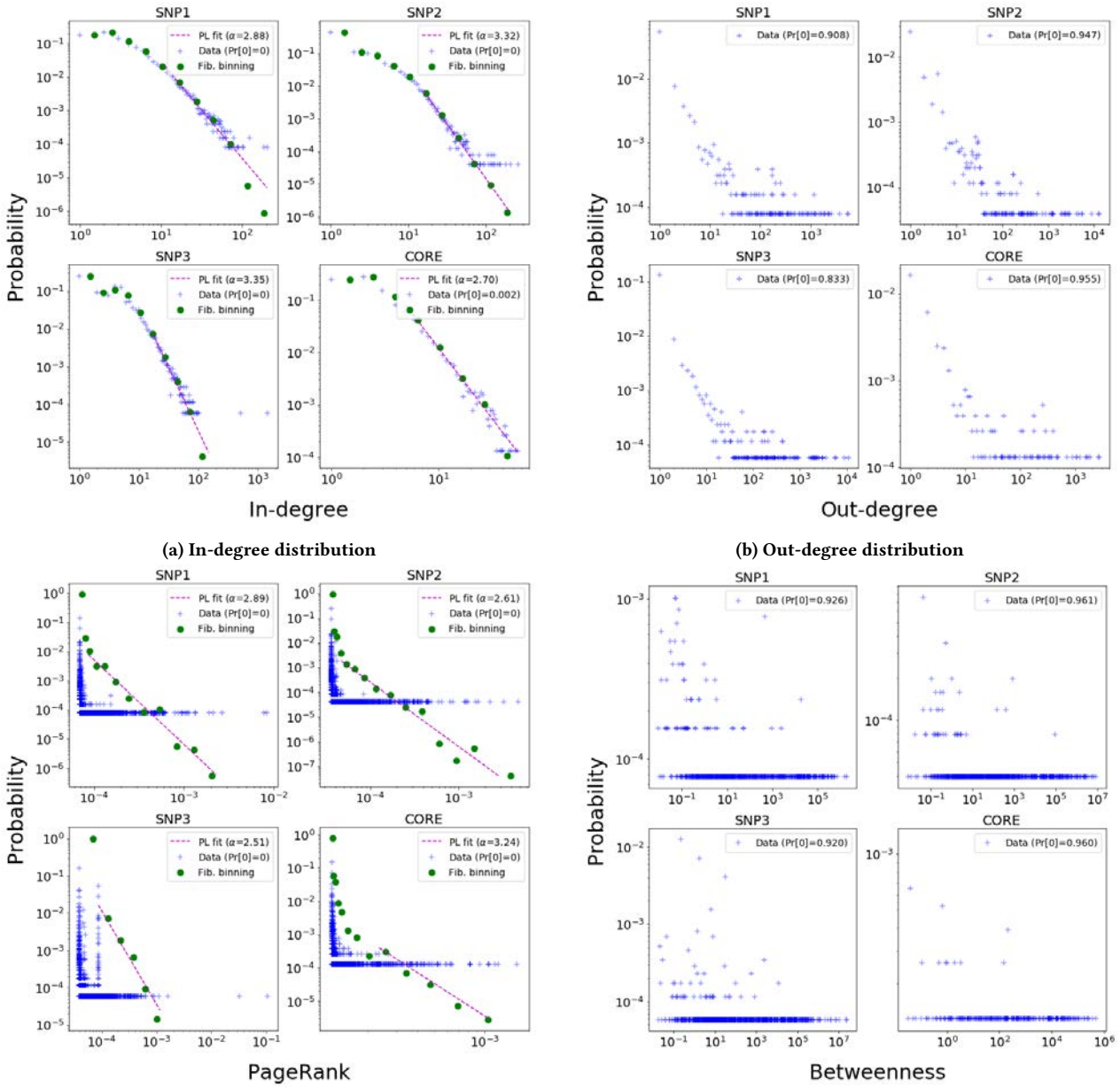
As commonly done to describe Web graphs [12], in Table 6 we provide a bow-tie decomposition of our graphs compared with previous results from the literature. Our findings broadly confirm what emerged in [23], *i.e.*, that the difference between the Tor Web and the WWW is huge and well synthetized by two facts: the LSSC is very small and it lies "on top" of everything else. However, with respect to [23] we implemented a more thorough data collection that brings to the light three novel features of the Tor Web. First, there exists a nonempty set of active hidden services that are completely disconnected from the rest of the graph. Second, the share of the LSCC in the total size of the graph is significantly variable over time, to the point that in SNP3 it is $\sim 4\times$ larger than in the other two snapshots. Finally, the structure of the CORE graph has a few peculiarities: the IN component is non-empty, but instead it is composed of a tiny set of 9 hidden services; the LSCC is even smaller than in the snapshots; the DISCONNECTED component is significanly larger, meaning that in general hyperlinks are more volatile than hidden services.

4.5 Top Hidden Services by Centralities

Considering the pairwise comparison of in-degree, out-degree, PageRank and betweenness of each hidden service shown in Figure 5, we observed that the most interesting services are usually link directories. In SNP1 we find *fhostingesps6bly,*, the service with the top in-degree and PageRank values, that contains an URL redirection[10] to the Hidden Wiki whose current onion address is *zqktlwi4i34kbat3*. The Hidden Wiki is a Tor link directory, probably the most famous. In SNP1 we find also *underdj5ziov3ic7* that is the service with the top betweenness and out-degree values. This service contains a Tor link directory named UnderDir - The Undernet Directory. The last service in SNP1 is *blockchainbdgpzk* that is the service with the second PageRank and in-degree values. This service contains an URL redirection to a surface website of a company named Blockchain Luxembourg S.A.R.L. that offers services related to digital currencies.

In SNP2 we find *tt3j2x4k5ycaa5zt* that is the service with the top PageRank and second in-degree values; it contains a personal

[10]The HTTP Status Code used for an URL redirection/URL forwarding is a 3XX status code



(a) In-degree distribution

(b) Out-degree distribution

(c) PageRank distribution

(d) Betweenness distribution

Figure 3: Probability distribution of centrality measures.

website named Daniel's Home. In the website a section is dedicated to a collection of Tor links. In SNP2 we find also *zlal32teyptf4tvi* that is the top betweenness and second out-degree service. This service is a Tor link directory named Fresh Onions. The last service in SNP2 is *vj5wxqmjaes2bae5* that has the top out-degree value, it contains a Tor link directory.

In SNP3 we find *dhosting4okcs22v* that is the service with top PageRank, betweenness and in-degree values; it contains a hosting service named Daniel's Hosting. This is a section of the website we found in SNP2 named Daniel's Home. It uses a different onion address, but it is actually the same website. The last service in SNP2 is *zlal32teyptf4tvi* that is the service with the top out-degree value containing the Tor link directory named Fresh Onions, that we found in SNP2.

Finally in CORE we find *blockchainbdgpzk* and *underdj5ziov3ic7* that are respectively the services with the top PageRank, second in-degree service and the top betweenness, out-degree values. We already discussed these two onion addresses for SNP1. In CORE we find also *grams7enufi7jmdl* that is the service with the second in-degree value. This service contains a Tor search engine focused on

Table 6: Bow-Tie structure

Component	WWW from [29]		Tor from [23]		SNP1		SNP2		SNP3		CORE	
	# nodes	%	# nodes	%	# nodes	%	# nodes	%	# nodes	%	# nodes	%
LSCC	22.3M	51.94%	297	4.14%	466	3.63%	820	3.24%	2371	13.58%	169	2.2%
IN	3.3M	7.65%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	9	0.12%
OUT	13.3M	30.98%	6881	95.86%	12312	95.94%	24468	96.68%	15057	86.24%	7353	95.88%
TUBES	17K	0.04%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
TENDRILS	514k	1.2%	0	0.0%	0	0.0%	0	0.0%	0	0.0%	0	0.0%
DISCONNECTED	3.5M	8.2%	0	0.0%	55	0.43%	20	0.08%	32	0.18%	138	1.8%

LSCC is the largest strongly connected component.
IN is the set of nodes $v \in V \setminus$ LSCC such that there is a path from v to LSCC.
OUT is the set of nodes $v \in V \setminus$ LSCC such that there is a path from LSCC to v.
TUBES is the set of nodes $v \in V \setminus$ (LSCC $\cup$ IN $\cup$ OUT) such that there is a path from IN to v as well as a path from v to OUT.
TENDRILS is the set of nodes $v \in V \setminus$ (LSCC $\cup$ IN $\cup$ OUT) such that there is either a path from IN to v or a path from v to OUT, but not both.
DISCONNECTED is the set of all other nodes $v \in V \setminus$ (LSCC $\cup$ IN $\cup$ OUT $\cup$ TUBES $\cup$ TENDRILS).

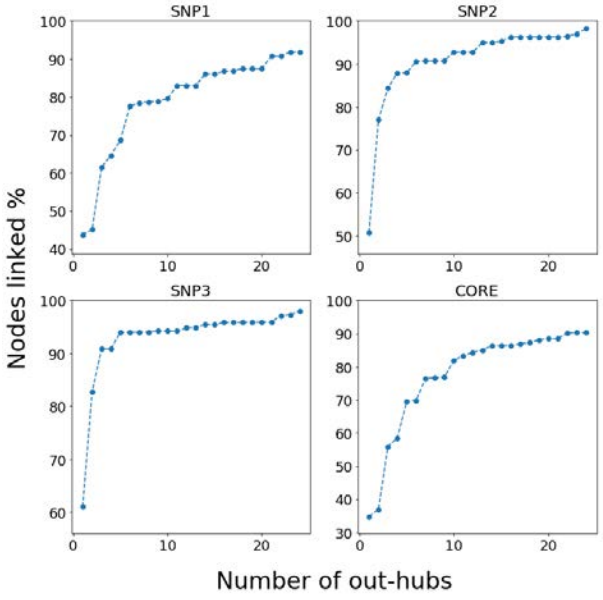


Figure 4: Cumulative percentage of the graph linked by the top out-degree vertices.

Tor marketplaces. The last service in CORE is *zqktlwi4fecvo6ri* that is the service with the second betweenness value and it contains an URL redirection to the Hidden Wiki.

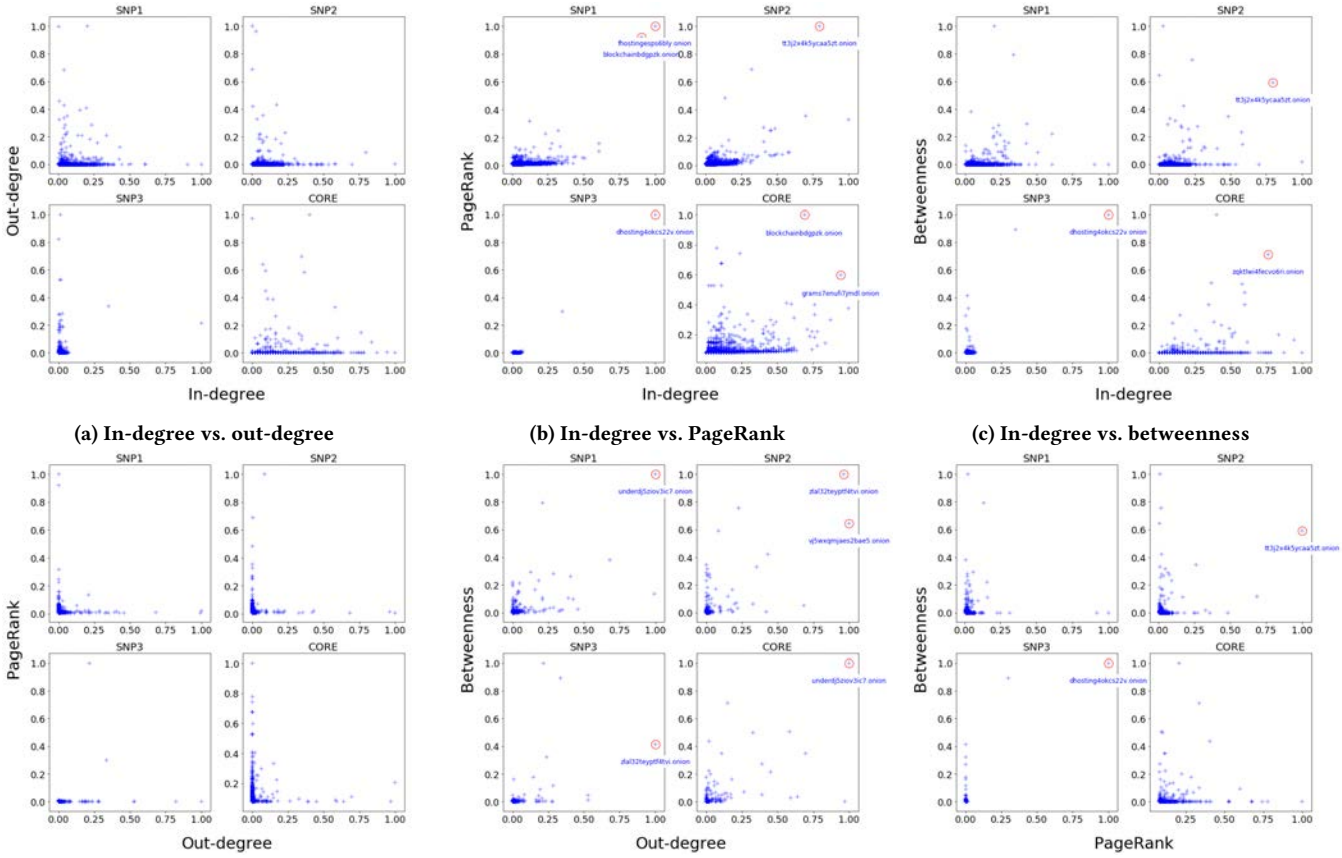
5 DISCUSSION

In this Section we read and discuss our findings in the light of the body of work on real world complex networks. Aiming at assessing to which extent the Tor Web graph fits the three most-known generative models for random graphs – Erdos-Renyi (ER) [20], Watts-Strogatz (WS) [40], Barabasi-Albert (BA) [5] – we focus on three properties that are especially informative: the average shortest path length (or average distance), the clustering coefficient (or transitivity) and the shape of the degree distribution. Notably, these three characteristics are also known to be discriminatory in many practical settings, such as: for predicting the growth dynamics of a network [2], for controlling the spreading of viruses/rumors [32], or for determining the robustness against random node failures [13].

For what concerns the average distance, the question is whether the Tor graph looks like a *small world* or even an *ultra-small world* network. In small world networks the average distance satisfies $\langle dist \rangle \propto \ln(N)$, whereas in ultra-small networks $\langle dist \rangle \propto \ln\ln(N)$. Although the latter are asymptotic estimates, our numbers suggest that Tor belongs, at least, to the class of small world networks: for all four graphs considered (the three snapshots and the CORE) $\langle dist \rangle$ satisfies $3.5 < \langle dist \rangle < 5$, whereas $2 < \ln\ln(N) < 2.5$ and $8.5 < \ln(N) < 10.5$.

The global clustering coefficient measures the frequency of closed triangles in the graph, thus representing an indicator of the existence of some level of correlation in the adjacency patterns of neighboring vertices. If edges occur independently and uniformly at random, the clustering coefficient C satisfies $C \propto \frac{\langle \deg \rangle}{N}$, where $\langle \deg \rangle$ is the average degree of the network. Again, although the latter is only an asymptotic estimate, our findings speak in favor of the existence of a positive correlation for Tor edges. For directed graphs, other than the global (undirected) clustering coefficient it is possible to consider a few types of directed transitivities: we chose two such metrics, denoted T_1 and T_2 and defined in Table 2. Similarly, the average degree can be computed both ignoring or considering edge directions, and the value reported in Table 4 for $\langle \deg \rangle$ is the average number of in-bound or (equivalently) out-bound edges, thus C should be compared with $\frac{2\langle \deg \rangle}{N}$ in our case. In all four Tor graphs, C is one order of magnitude greater than $\frac{2\langle \deg \rangle}{N}$, and both T_1 and T_2 are one order of magnitude greater than $\frac{\langle \deg \rangle}{N}$.

Finally, when looking at the degree distribution of a network, the first aspect to consider is whether it has a heavy tail, which is symptomatic of the tendency of nodes to connect to "authoritative" hubs. By looking at Figures 3a and 3b it is clear that this is the case for the Tor Web. However, while the in-degree distribution seems to follow a power law for all four graphs, albeit with different



(a) In-degree vs. out-degree **(b) In-degree vs. PageRank** **(c) In-degree vs. betweenness**

(d) Out-degree vs. PageRank **(e) Out-degree vs. betweenness** **(f) PageRank vs. betweenness**

Figure 5: Pairwise comparison between centrality measures.

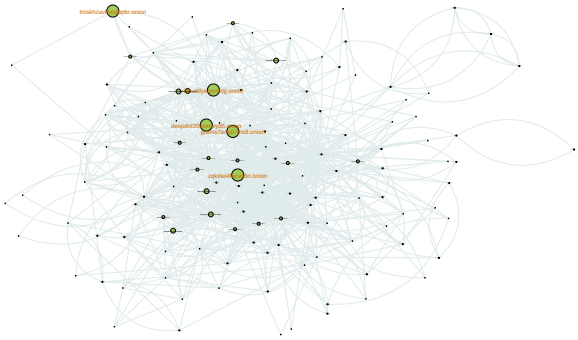
exponents, the out-degree shows an even slower decay and an even longer tail.

Our findings make it clear that the Tor Web is not an ER graph. In the ER random graph model with N vertices and with expected degree $\langle \deg \rangle$, the expected average distance satisfies $\langle dist \rangle \propto \ln(N)$, but the expected global clustering coefficient is $C \propto \frac{\langle \deg \rangle}{N}$ and the degree distribution has no heavy tail. The WS model predicts that $\langle dist \rangle \propto \ln(N)$ and that $C \gg \frac{\langle \deg \rangle}{N}$, thus suiting our findings quite well. Yet, the values of C measured in our four graphs are not as large as in other real-world small world networks [5, 32, 40]. Additionally, the WS model alone does not predict the observed power law distribution of the in-degree. The power law degree distribution is the defining property of the BA model. However, the BA model theoretically predicts the existence of two regimes according to whether the power law exponent α satisfies $2 < \alpha < 3$ or $\alpha > 3$. In the former case, the variance of the distribution diverges and the network is ultra-small. In the latter, the variance of the distribution is finite and the network is small. The existence of this threshold is not visible in our findings: SNP1 and CORE have $2 < \alpha < 3$, whereas SNP2 and SNP3 have $\alpha > 3$, but all four graphs have very similar average distance and degree distribution. Although real networks cannot have an infinite variance, if N is large enough, node degrees should span several orders of magnitude if and only if $2 < \alpha < 3$, but this seems not be the case for the out-degree in Tor.

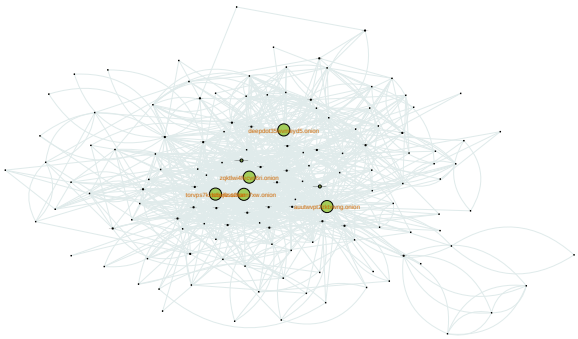
Summing up, Tor shows a blend of different features and a deeper analysis is needed to understand whether any existing generative model matches its characteristics. Among Tor's unique features, we observe that the overall structure is clearly dominated by the presence of a small number of *out-hubs*, which connect both central and peripheral nodes. It is also worth noting that the sizes of the LSCCs are relatively small compared to those reported by other studies on the surface WWW or social networks [5, 27, 32]. It is interesting that the nodes in the LSCC core are persistent in the three snapshots, suggesting that some subnet inside Tor could have specific properties and even a different structure with respect to the whole network.

Many questions that are still open require further analysis. Tor is built with anonymity in mind thus many hidden services are supposedly not interested in having visibility, yet some of its topological features, *e.g.* the radius and diameter, suggest the existence of a peculiar mechanism that leads to the growth of the network. On the practical side, one may be especially interested in understanding how the existence of large *out-hubs* impacts on the topological

properties of the network, and whether topologically similar nodes host analogous contents. To provide a few insights in this regards, we removed from the CORE set the 10 services with maximum out-degree and analyzed the LSCC of the obtained graph, depicted in Figure 6. The most remarkable finding is that, albeit the average shortest path remains almost unchanged, the obtained network has a clustering coefficient more than 20 times larger than that of a random graph of equal size. We also found that *out-hubs* are almost all *hidden directories* and we have some evidence that other centrality measures can be related to the content of the hidden services. In particular, as reported in Table 7, 4 out of the top 5 PageRank services are related to marketplaces, whereas high betweenness seems not equally characteristic of a specific category of services.



(a) Node size proportional to their PageRank.



(b) Node size proportional to their betweenness centrality.

Figure 6: The LSCC of the CORE graph.

6 CONCLUSIONS

This paper studied three sets of crawling data collected over three different time frames as well as their common "stable" core. It provided a deep characterization of the topology of the Tor services graph, identifying structural and temporal features and further assessing the persistence of hidden services and hyperlinks.

While previous work only focused on the volatility of Tor hidden services, thanks to a graph-oriented perspective we were also able to assess the persistence of Tor hyperlinks. This led to the key finding that edges are more volatile than nodes in the Tor Web graph, as proved by the fact that hidden services shared by two or more snapshots do not induce the same subgraph in these snapshots. Additionally, we observed that the LSCC of the CORE graph is persistent in all three snapshots. Compared with the generally high volatility of the Tor network, this is the first evidence that Tor may be comprised of different layers, each with a precise role in the connectivity patterns of the network, and possibly with different inter- and intra-layer structures.

onion	PAGERANK	Topic
grams7enufi7jmdl	0.053	Market SE
deepdot35wvmeyd5	0.048	Market
lchudifyeqm4ldjj	0.046	Market
zqktlwi4fecvo6ri	0.045	Hidden Wiki
tmskhzavkycdupbr	0.041	Market
onion	**BETWEENNESS**	**Topic**
auutwvpt2zktxwng	6331	Directory
zqktlwi4fecvo6ri	6105	Hidden Wiki
deepdot35wvmeyd5	3036	Market
torpress2sarn7xw	2766	News/Blogs
torvps7kzis5ujfz	2396	News/Blogs

Table 7: Best 5 nodes with respect to PageRank and betweenness centrality.

We computed several topological metrics on the Tor snapshots and compared them to well-known network models (ER, WS, BA). None of those models appears to be suitable to accurately represent Tor. A small number of *out-hubs* connecting both central and peripheral nodes dominates the structure of the graph, whereas LSCCs are (relatively/due proportions made) smaller than those of the surface Web or social networks. By removing the *out-hubs* with higher degree, the clustering coefficient of the network grows but the average shortest path remains almost constant: this again suggests that additional insights into Tor's dynamics could be obtained by removing and/or isolating specific subnets. Centrality metrics also indicate that there could be some interesting relation among node role/position and content: nodes with higher degree are almost always link directories, whereas we found that four out of five nodes in the top PageRank are related to marketplaces. While these results do not suffice to draw a clear picture, they surely indicate that further research must be carried out [11].

We believe the results presented here will foster a larger discussion on the topic, and will be a useful reference for evaluation and comparison against other real-world graphs.

REFERENCES

[1] Dimitris Achlioptas, Aaron Clauset, David Kempe, and Cristopher Moore. 2009. On the Bias of Traceroute Sampling: Or, Power-law Degree Distributions in Regular Graphs. *J. ACM* 56, 4, Article 21 (July 2009), 28 pages. https://doi.org/10.1145/1538902.1538905

[2] Réka Albert, Hawoong Jeong, and Albert-László Barabási. 1999. Internet: Diameter of the world-wide web. *nature* 401, 6749 (1999), 130.

[3] Jeff Alstott, Ed Bullmore, and Dietmar Plenz. 2014. Powerlaw: a Python package for analysis of heavy-tailed distributions. *PloS one* 9, 1 (2014), e85777.

[11] Further information is available here http://www.cranic.it/tor.html, here 757qx3mfy4opg73o.onion, and here dym3ubprp66kbj33o22q6mehjwvixknrblndb65oi5sk2v2fpx4enrad.onion

[4] Robert Annessi and Martin Schmiedecker. 2016. NavigaTor: Finding Faster Paths to Anonymity. In *IEEE European Symposium on Security and Privacy (Euro S&P)*. IEEE.

[5] Albert-László Barabási and Réka Albert. 1999. Emergence of Scaling in Random Networks. *Science* 286, 5439 (1999), 509–512. https://doi.org/10.1126/science.286.5439.509

[6] Massimo Bernaschi, Alessandro Celestini, Stefano Guarino, and Flavio Lombardi. 2017. Exploring and Analyzing the Tor Hidden Services Graph. *ACM Trans. Web* 11, 4, Article 24 (jul 2017), 24:1–24:26 pages. https://doi.org/10.1145/3008662

[7] Alex Biryukov, Ivan Pustogarov, Fabrice Thill, and Ralf-Philipp Weinmann. 2014. Content and Popularity Analysis of Tor Hidden Services. In *Distributed Computing Systems Workshops (ICDCSW), 2014 IEEE 34th International Conference on*. 188–193. https://doi.org/10.1109/ICDCSW.2014.20

[8] Alex Biryukov, Ivan Pustogarov, and Ralf-Philipp Weinmann. 2013. Trawling for Tor Hidden Services: Detection, Measurement, Deanonymization. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy (SP '13)*. IEEE Computer Society, Washington, DC, USA, 80–94. https://doi.org/10.1109/SP.2013.15

[9] Paolo Boldi, Andrea Marino, Massimo Santini, and Sebastiano Vigna. 2014. BUbiNG: Massive crawling for the masses. In *Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion*. 227–228.

[10] Anthony Bonato. 2005. A Survey of Models of the Web Graph. In *Combinatorial and Algorithmic Aspects of Networking*, Alejandro Lopez-Ortiz and Angelem Hamel (Eds.). Lecture Notes in Computer Science, Vol. 3405. Springer Berlin Heidelberg, 159–172. https://doi.org/10.1007/11527954_16

[11] Stephen P. Borgatti and Martin G. Everett. 2006. A Graph-theoretic perspective on centrality. *Social Networks* 28, 4 (2006), 466 – 484. https://doi.org/10.1016/j.socnet.2005.11.005

[12] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. 2000. Graph structure in the Web. *Computer Networks* 33, 1-6 (2000), 309 – 320. https://doi.org/10.1016/S1389-1286(00)00083-9

[13] Duncan S. Callaway, M. E. J. Newman, Steven H. Strogatz, and Duncan J. Watts. 2000. Network Robustness and Fragility: Percolation on Random Graphs. *Phys. Rev. Lett.* 85 (Dec 2000), 5468–5471. Issue 25. https://doi.org/10.1103/PhysRevLett.85.5468

[14] Alessandro Celestini and Stefano Guarino. 2017. Design, Implementation and Test of a Flexible Tor-oriented Web Mining Toolkit. In *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics (WIMS '17)*. ACM, New York, NY, USA, Article 19, 10 pages. https://doi.org/10.1145/3102254.3102266

[15] Nicolas Christin. 2013. Traveling the Silk Road: A Measurement Analysis of a Large Anonymous Online Marketplace. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13)*. ACM, New York, NY, USA, 213–224. https://doi.org/10.1145/2488388.2488408

[16] Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. 2009. Power-law distributions in empirical data. *SIAM review* 51, 4 (2009), 661–703.

[17] Manlio De Domenico and Alex Arenas. 2017. Modeling structure and resilience of the dark network. *Phys. Rev. E* 95 (Feb 2017), 022313. Issue 2. https://doi.org/10.1103/PhysRevE.95.022313

[18] Debora Donato, Stefano Leonardi, Stefano Millozzi, and Panayiotis Tsaparas. 2008. Mining the inner structure of the Web graph. *Journal of Physics A: Mathematical and Theoretical* 41, 22 (2008), 224017. http://stacks.iop.org/1751-8121/41/i=22/a=224017

[19] Paul Erdős and Alfréd Rényi. 1959. On random graphs. *Publicationes Mathematicae Debrecen* 6 (1959), 290–297.

[20] P. ErdÅŠs and A RÃľnyi. 1960. On the Evolution of Random Graphs. In *PUBLICATION OF THE MATHEMATICAL INSTITUTE OF THE HUNGARIAN ACADEMY OF SCIENCES*. 17–61.

[21] Massimo Franceschet. 2011. PageRank: Standing on the Shoulders of Giants. *Commun. ACM* 54, 6 (June 2011), 92–101. https://doi.org/10.1145/1953122.1953146

[22] Shalini Ghosh, Ariyam Das, Phil Porras, Vinod Yegneswaran, and Ashish Gehani. 2017. Automated Categorization of Onion Sites for Analyzing the Darkweb Ecosystem. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*. ACM, New York, NY, USA, 1793–1802. https://doi.org/10.1145/3097983.3098193

[23] Virgil Griffith, Yang Xu, and Carlo Ratti. 2017. Graph Theoretic Properties of the Darkweb. *arXiv preprint arXiv:1704.07525* (2017).

[24] Rob Jansen, Kevin Bauer, Nicholas Hopper, and Roger Dingledine. 2012. Methodically Modeling the Tor Network. In *Proceedings of the 5th USENIX Conference on Cyber Security Experimentation and Test (CSET'12)*. USENIX Association, Berkeley, CA, USA, 8–8. http://dl.acm.org/citation.cfm?id=2372336.2372347

[25] Rohit Khare, Doug Cutting, Kragen Sitaker, and Adam Rifkin. 2004. Nutch: A flexible and scalable open-source web search engine. *Oregon State University* 1 (2004), 32–32.

[26] JonM. Kleinberg, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and AndrewS. Tomkins. 1999. The Web as a Graph: Measurements, Models, and Methods. In *Computing and Combinatorics*, Takano Asano, Hideki Imai, D.T. Lee, Shin-ichi Nakano, and Takeshi Tokuyama (Eds.). Lecture Notes in Computer Science, Vol. 1627. Springer Berlin Heidelberg, 1–17. https://doi.org/10.1007/3-540-48686-0_1

[27] Ravi Kumar, Jasmine Novak, and Andrew Tomkins. 2010. *Structure and Evolution of Online Social Networks*. Springer New York, New York, NY, 337–357. https://doi.org/10.1007/978-1-4419-6515-8_13

[28] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D Sivakumar, Andrew Tomkins, and Eli Upfal. 2000. Stochastic models for the Web graph. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*. 57–65. https://doi.org/10.1109/SFCS.2000.892065

[29] Oliver Lehmberg, Robert Meusel, and Christian Bizer. 2014. Graph Structure in the Web: Aggregated by Pay-level Domain. In *Proceedings of the 2014 ACM Conference on Web Science (WebSci '14)*. ACM, New York, NY, USA, 119–128. https://doi.org/10.1145/2615569.2615674

[30] Robert Meusel, Sebastiano Vigna, Oliver Lehmberg, and Christian Bizer. 2015. The Graph Structure in the Web âĂŞ Analyzed on Different Aggregation Levels. *The Journal of Web Science* 1, 1 (2015), 33–47. https://doi.org/10.1561/106.00000003

[31] Gordon Mohr, Michael Stack, Igor Ranitovic, Dan Avery, and Michele Kimpton. 2004. An Introduction to Heritrix An open source archival quality web crawler. In *In IWAWâĂŹ04, 4th International Web Archiving Workshop*. Citeseer.

[32] Mark Newman, Albert-Laszlo Barabasi, and Duncan J. Watts. 2006. *The Structure and Dynamics of Networks: (Princeton Studies in Complexity)*. Princeton University Press, Princeton, NJ, USA.

[33] M. E. J. Newman. 2003. Mixing patterns in networks. *Phys. Rev. E* 67, 2 (Feb 2003), 026126. https://doi.org/10.1103/PhysRevE.67.026126

[34] Gareth Owen and Nick Savage. 2015. The Tor dark net. (2015).

[35] Gareth Owen and Nick Savage. 2016. Empirical analysis of Tor hidden services. *IET Information Security* 10, 3 (2016), 113–118.

[36] Iskander Sanchez-Rola, Davide Balzarotti, and Igor Santos. 2017. The Onions Have Eyes: A Comprehensive Structure and Privacy Analysis of Tor Hidden Services. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1251–1260. https://doi.org/10.1145/3038912.3052657

[37] M. Ángeles Serrano, Ana Maguitman, Marián Boguñá, Santo Fortunato, and Alessandro Vespignani. 2007. Decoding the Structure of the WWW: A Comparative Analysis of Web Crawls. *ACM Trans. Web* 1, 2, Article 10 (Aug. 2007). https://doi.org/10.1145/1255438.1255442

[38] Martijn Spitters, Stefan Verbruggen, and Mark van Staalduinen. 2014. Towards a Comprehensive Insight into the Thematic Organization of the Tor Hidden Services. In *Intelligence and Security Informatics Conference (JISIC), 2014 IEEE Joint*. 220–223. https://doi.org/10.1109/JISIC.2014.40

[39] Sebastiano Vigna. 2013. Fibonacci binning. *arXiv preprint arXiv:1312.3749* (2013).

[40] Duncan J. Watts. 1999. *Small Worlds: The Dynamics of Networks Between Order and Randomness*. Princeton University Press, Princeton, NJ, USA.

Be Concise and Precise: Synthesizing Open-Domain Entity Descriptions from Facts

Rajarshi Bhowmik
Rutgers University - New Brunswick
Piscataway, New Jersey
rajarshi.bhowmik@rutgers.edu

Gerard de Melo
Rutgers University - New Brunswick
Piscataway, New Jersey
gerard.demelo@cs.rutgers.edu

ABSTRACT

Despite being vast repositories of factual information, cross-domain knowledge graphs, such as Wikidata and the Google Knowledge Graph, only sparsely provide short synoptic descriptions for entities. Such descriptions that briefly identify the most discernible features of an entity provide readers with a near-instantaneous understanding of what kind of entity they are being presented. They can also aid in tasks such as named entity disambiguation, ontological type determination, and answering entity queries. Given the rapidly increasing numbers of entities in knowledge graphs, a fully automated synthesis of succinct textual descriptions from underlying factual information is essential. To this end, we propose a novel fact-to-sequence encoder–decoder model with a suitable copy mechanism to generate concise and precise textual descriptions of entities. In an in-depth evaluation, we demonstrate that our method significantly outperforms state-of-the-art alternatives.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; *Natural language generation.*

KEYWORDS

synoptic description generation; knowledge graphs; open-domain factual knowledge

ACM Reference Format:
Rajarshi Bhowmik and Gerard de Melo. 2019. Be Concise and Precise: Synthesizing Open-Domain Entity Descriptions from Facts. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313656

1 INTRODUCTION

Motivation. A substantial percentage of online search requests involve named entities such as people, locations, businesses, etc. Search engines and digital personal assistants (e.g., Google Assistant, Alexa, Cortana, Siri) alike now extensively draw on *knowledge graphs* as vast databases of entities and their properties.

Most large cross-domain factual knowledge graphs, such as Wikidata [25], the Google Knowledge Graph, YAGO [11], and MENTA [6], include short textual descriptions of entities. These can be provided in response to entity queries (such as *Where is Fogo Island?*), e.g., as an informative direct answer or to enrich a knowledge panel given by a Web search engine such as Google, as shown in Figure 1 for Fogo Island. Such textual descriptions effectively provide for a near-instantaneous human understanding of an entity. They can also be helpful in a number of linguistic tasks, including named entity disambiguation, and can serve as fine-grained ontological types in question answering and reasoning-driven applications.

Despite their eminent importance for information retrieval and other applications, these descriptions are only sparsely available, typically for the more well-known entities, leaving large numbers of entities with no such descriptions. For example, in Wikidata, which is used by Apple's Siri, as many as 7.2 million entities do not have

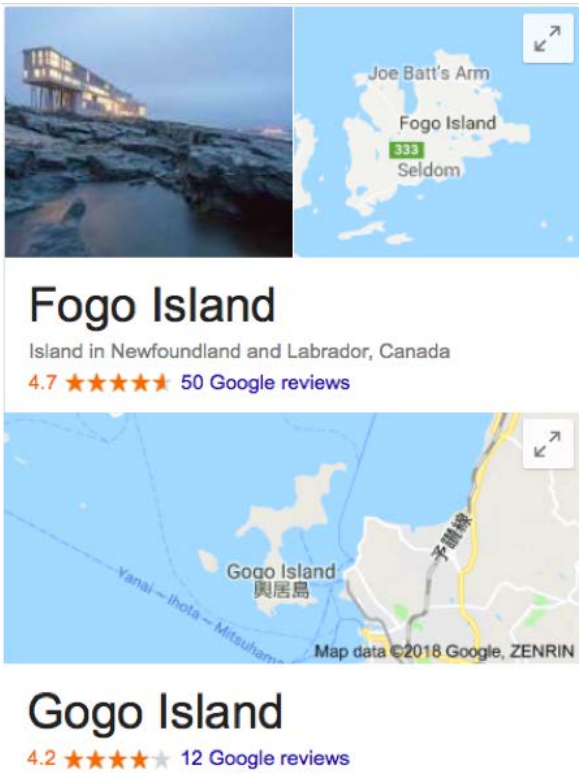


Figure 1: An example of a missing description in Google's Knowledge Graph. Similar to *Fogo Island*, a synoptic description for *Gogo Island* could be *Island in Ehime, Japan.*

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313656

any description in any language as of December 2018.[1] Moreover, new entities are added frequently to the existing knowledge graphs. For example, the number of entities in Wikidata almost doubled in the last couple of years.

Given the large number of entities lacking textual descriptions, a promising solution is to fully automatically generate synoptic textual descriptions from the underlying factual data. Ideally, this automatic generation process should be able to synthesize very concise descriptions, retaining only a very small number of particularly discernible pieces of information about an entity. For example, in Wikidata and Google's Knowledge Graph, *occupation* and *nationality* are among the preferred attributes invoked in describing a person (e.g., *Magnus Carlsen* as a *Norwegian chess Grandmaster*). At the same time, however, an *open-domain* solution is needed, in order to cope with the various different kinds of entities users may search for in a knowledge graph, and the various kinds of names that might appear in such descriptions. For instance, for Gogo Island in Figure 1, an appropriate description would perhaps refer to *Ehime, Japan* as the prefecture in which the island is located. Moreover, apart from people and locations, users could likewise also search for paintings, films, or even asteroids. Additionally, the generated descriptions should also be precise, coherent, and non-redundant.

Overview and Contributions. In this paper, we present a novel fact-to-sequence encoder–decoder model that solves this challenging task. Our deep neural model attempts to generate a textual description similar to those created in image and video captioning tasks [17], but instead starts from structured data. Our model is equipped with an explicit copy mechanism to copy fact-specific tokens directly to the output description. Since many names and words in the desired output descriptions are also present in the underlying facts, it is an intuitive design choice to selectively copy words from the facts to the description. The copy mechanism is important, as it is difficult to predict rare or unseen words based on mere statistical co-occurrences of words. In comparison to the state-of-the-art, our model is more than 14x more parameter-efficient and gains 5–8 absolute percentage points across a number of metrics in an empirical evaluation.

This approach is substantially different from previous work on this topic. In contrast to some prior works that focus on generating Wikipedia-style summary descriptions from factual data [1, 14], or textual verbalization of RDF triples [5, 8], our work is focused on synthesizing quickly graspable synoptic textual descriptions from factual knowledge. In most of the cases, these are single-phrase descriptions as compared to multi-sentence Wikipedia-style summaries. Note that we restrict our work only to English description generation, as English is the most prevalent language in which descriptions are available in knowledge graphs. Typically, the first few sentences of Wikipedia articles contain a detailed description of the entity, and one can attempt to generate brief textual descriptions by learning to reproduce the first few sentences based on the remaining article text. However, our goal is to make our method open-domain, i.e., applicable to any type of entity with factual information that may not have a corresponding Wikipedia-like article available. Indeed, Wikidata currently has more than 51 million items, whereas the English Wikipedia has only 5.6 million articles.

[1] https://tools.wmflabs.org/wikidata-todo/stats.php

Our specific contributions in this paper are as follows:

- We introduce a fact-to-sequence encoder–decoder architecture that precisely copies fact-specific words using a two-step pointing and copying mechanism.
- Our model is 14x as parameter-efficient than the state-of-the-art models and 14–20x as parameter-efficient compared to competitive baselines.
- Through extensive evaluation, we show that our method outperforms other baselines with significant improvement, especially with regard to descriptions requiring open-domain name mentions.

2 MODEL

In order to enable concise and precise open-domain entity descriptions, we propose a generative model consisting of a fact-to-sequence encoder–decoder architecture, aided by a pointer network based copy mechanism. Our model comprises a positional fact encoder, a GRU-based sequence decoder, and a copy function to directly transfer factual words to the generated descriptions. Before delving into the details of these key components, however, we first introduce some essential terminologies that we shall use in the following.

2.1 Preliminaries

Property. Each subject entity $s \in \mathcal{E}$ has a number of properties $P \in \mathcal{P}$ associated with it. Each property name is described as a sequence of tokens $P = (w_1^P, w_2^P, ..., w_{|P|}^P)$. Note that these properties are predefined in a schema-based knowledge graph.

Property Value. Each property P of entity s has a corresponding value O_P, which is also a sequence of tokens $O_P = (w_1^o, w_2^o, ..., w_{|O_P|}^o)$. The property values could be another entity $e \in \mathcal{E}$, a date–time value, a string, or a numeric value, etc.

Fact. Each of the property–value pairs is deemed a *fact*. Each subject entity $s \in \mathcal{E}$ has a number of different facts $\mathcal{F} = \{f_1, f_2, \dots f_N\}$ that characterize it.

Description. A short textual description of entity s is a sequence of tokens $\mathcal{D} = (w_1, w_2, \dots, w_n)$. Each word in the description D can be a *factual word*, or a *vocabulary word*.

Factual Words. The factual words for each fact $f \in \mathcal{F}$, denoted as $\mathcal{V}_f$, are the words appearing in the property value. Stop words (e.g., *for, of, in* etc.) are excluded from the factual words vocabulary. Note that the sets of factual words are instance-specific, i.e., $\bigcup_f \mathcal{V}_f$ is different for different entities $s \in \mathcal{E}$.

Vocabulary Words. The vocabulary words, denoted as $\mathcal{V}$, could be a list of frequent words in the English dictionary. For our experiments, we rely on the 1,000 most frequent words appearing in our corpus as vocabulary words. Typically, the words that appear in the property names constitute this vocabulary. This is aligned with the expectation for a schema-based knowledge graph with fixed number of properties. Note that $\mathcal{V}_f \cap \mathcal{V} \neq \emptyset$, which indicates that there could be words in the property value that are also vocabulary words.

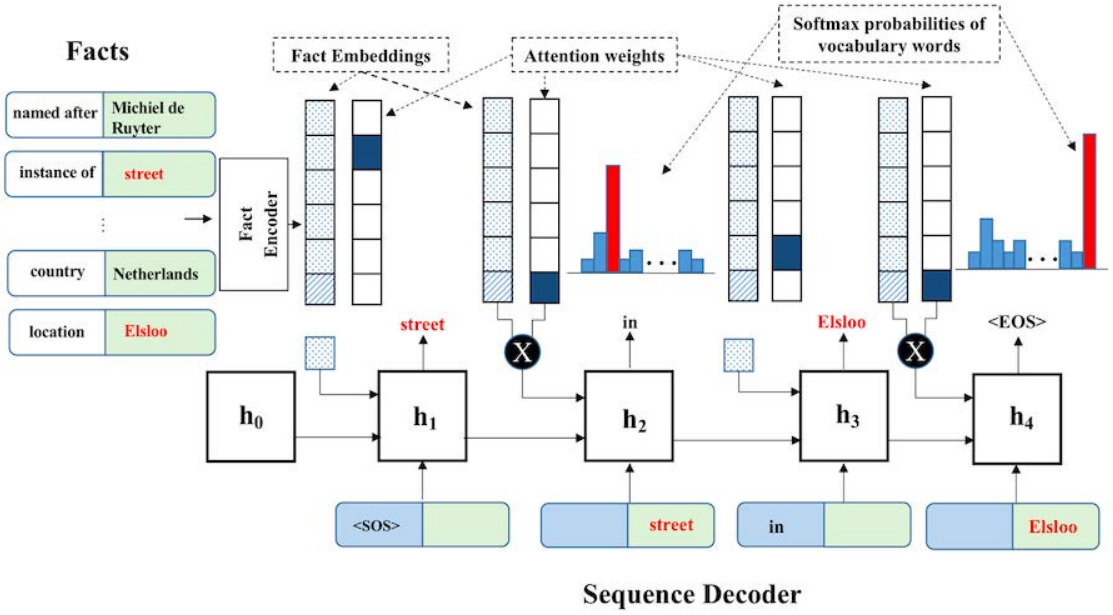


Figure 2: Model architecture. For Wikidata item Q19345316 (*Michiel de Ruyterstraat*), factual words such as *street* and *Elsloo* are directly copied from the underlying facts *(Instance of, street)* and *(location, Elsloo)*, respectively, while the general vocabulary words *in* and *<EOS>* are selected by a softmax classifier.

2.2 Fact Encoder

The fact encoder transforms a set of input facts $\mathcal{F} = \{f_1, f_2, \ldots f_N\}$ into *fact embeddings* – distributed vector representations of facts. For simplicity and efficiency, we use a positional encoding scheme proposed by Sukhbaatar et al. [22] that was also adopted in other architectures [3, 26].

Using the positional encoding scheme, each fact f_i is represented as $\mathbf{f}_i = \sum_j \mathbf{l}_j \odot \mathbf{w}_j^i$, where $\odot$ denotes element-wise multiplication, $\mathbf{l}_j$ is a column vector with the structure $l_{kj} = (1 - \frac{j}{J}) - (k/d)(1 - 2\frac{j}{J})$, with J being the number of words in the fact, and $\mathbf{w}_j^i \in \mathrm{R}^d$ is the word embedding of the j-th word in fact f_i. If the word w_i^j is not a vocabulary word, we replace it with the special token *<UNK>*. This arrangement deals with the rare or unseen words that appear in the factual phrase f_i, which is a concatenation of words in the property name and property value.

We append the fact embeddings with a *mean fact* – a special fact that is responsible for generating vocabulary words. It is derived as the element-wise mean of the fact embeddings, i.e., $\mathbf{f}_{N+1} = \frac{1}{N} \sum \mathbf{f}_i$. We train the model to attend to the mean fact if it has to generate a vocabulary word in the output sequence.

2.3 Output Sequence Decoder

At every time step t, the sequence decoder selects a fact. This selection governs the generation of the next word either from the corresponding factual words, or from the vocabulary words.

2.3.1 Fact Selection. Given the set of fact embeddings $\{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_{N+1}\}$ as input, the decoder selects a fact at each time step t using an attention mechanism. The attention scores are calculated as a likelihood distribution using the hidden state of the decoder in the previous time step and the fact embeddings. Formally,

$$\mathbf{e}_i = \mathbf{W}_2 \tanh(\mathbf{W}_1[\mathbf{f}_i; \mathbf{h}_{t-1}]) \;\; \forall i \in \{1, N+1\}, \tag{1}$$

$$P(f = f_i \mid \mathbf{f}_i, \mathbf{h}_{t-1}) = \frac{\exp(\mathbf{e}_i)}{\sum_{i' \in \{1, N+1\}} \exp(\mathbf{e}_{i'})}, \tag{2}$$

where $\mathbf{e}_i$ denotes the attention energy of the i-th fact, $[;]$ denotes the concatenation operation, and $\mathbf{W}_1 \in \mathrm{R}^{m \times 2d}$, $\mathbf{W}_2 \in \mathrm{R}^m$ are affine transformations of a 2-layer feed-forward network.

We select the fact with maximum attention score at time step t, denoted as f_t and its corresponding fact embedding $\mathbf{f}_t$ as

$$f_t = \underset{i \in \{1, \ldots, N+1\}}{\arg\max} \; P(f = f_i \mid \mathbf{f}_i, \mathbf{h}_{t-1}) \tag{3}$$

$$\mathbf{f}_t = \mathbf{f}_{f_t} \tag{4}$$

2.3.2 GRU Decoder. We rely on Gated Recurrent Units (GRU), which is a Recurrent Neural Network (RNN) variant, as our preferred decoder. At each time step t, the decoder input is a concatenation of three vectors: the embedding $\mathbf{f}_t$ of a selected fact in the current time step, the embedding $\mathbf{w}_{t-1}$ of the vocabulary word at the previous time step, and a one-hot vector $\mathbf{v}_{t-1}$ corresponding to the position of the copied factual word in the previous time step. Note that since the generated word in the previous time step can either be a vocabulary word or a factual word, either $\mathbf{w}_{t-1}$ or $\mathbf{v}_{t-1}$ is set to a zero vector.

The input representation $\mathbf{x}_t = [\mathbf{f}_t; \mathbf{w}_{t-1}; \mathbf{v}_{t-1}]$ is fed to the GRU decoder to update the states.

$$\mathbf{h}_t = \mathrm{GRU}(\mathbf{x}_t, \mathbf{h}_{t-1}) \tag{5}$$

2.3.3 Generating Vocabulary Words. If the attention mechanism assigns the maximum score to the mean fact f_{N+1}, then the decoder generates a vocabulary word $w_t \in \mathcal{V}$. To generate the vocabulary word, we use the attention-weighted context $\mathbf{c}_t = \sum_i \alpha_i \mathbf{f}_i$ and the GRU output state $\mathbf{h}_t$. A concatenation of these two vectors are fed to a 2-layer feed-forward network with a non-linear ReLU activation applied to the hidden layer. Formally,

$$\mathbf{o}_t = \mathbf{W}_a \text{ReLU}(\mathbf{W}_b[\mathbf{c}_t; \mathbf{h}_t]), \tag{6}$$

where $\mathbf{W}_a$ and $\mathbf{W}_b$ are affine transformations and $\text{ReLU}(\mathbf{x}) = \max(\mathbf{0}, \mathbf{x})$. Finally, a probability distribution over the vocabulary words is obtained by a softmax function and the word with the maximum probability is emitted by the decoder. Formally,

$$P(w \mid \mathbf{c}_t, \mathbf{h}_t) = \text{Softmax}(\mathbf{o}_t) \tag{7}$$

$$w_t = \arg\max_{w \in \mathcal{V}} P(w \mid \mathbf{c}_t, \mathbf{h}_t) \tag{8}$$

2.3.4 Copying Factual Words. The decoder copies factual words directly to the output when the fact selection process selects one of the N facts. To facilitate the copy mechanism, the decoder must select the position index of the factual word within the factual phrase. The position index is predicted by a 2-layer feed-forward network that takes a concatenation of the selected fact embedding $\mathbf{f}_t$ and the output state of the GRU $\mathbf{h}_t$ as input. Then, the position index of the word to copy is determined as follows.

$$\mathbf{r}_t = \mathbf{W}_c \text{ReLU}(\mathbf{W}_d[\mathbf{f}_t; \mathbf{h}_t]) \tag{9}$$

$$P(n \mid \mathbf{f}_t, \mathbf{h}_t) = \text{Softmax}(\mathbf{r}_t) \tag{10}$$

$$n_t = \arg\max_{n \in \{1, \ldots, |\mathcal{V}_{f_t}|\}} P(n \mid \mathbf{f}_t, \mathbf{h}_t) \tag{11}$$

$$w_t = (\mathcal{V}_{f_t})_{n_t} \tag{12}$$

Here, n_t is the position index of the factual word to copy and $\mathcal{V}_{f_t}$ is the sequence of factual words corresponding to fact f_t and $(\mathcal{V}_{f_t})_i$ denotes the i-th item in that sequence.

2.4 Training

Our method requires selective copying of factual words to generate a description. In order to obviate the need for ground truth alignments of output description words with facts for training, we introduce an automated method of annotating each token in the description so as to align it to a matching fact. Specifically, we rely on a greedy string matching algorithm as detailed in Algorithm 1 for this purpose. If a token is not aligned with any of the facts, it is annotated as a vocabulary word. However, if the token is neither present in the vocabulary word set nor in the factual word set, it is assigned the special token <*UNK*> to denote that it is neither a factual word nor a vocabulary word. The symbol NaF in Algorithm 1 indicates that such words are not aligned to any fact. Since in our implementation, we consider the mean fact as a source of vocabulary words, we align these words to the mean fact.

Note that the greedy string matching algorithm used for fact alignment is a heuristic process that can be noisy. If a token in the output description appears in more than one fact, the string matching algorithm greedily aligns the token to the first fact it encounters, even if it is not the most relevant one. However, our manual inspection suggests that in most of the cases the alignment is relevant, justifying our choice of such a greedy approach.

Algorithm 1: Algorithm for fact alignment of description

Data: Input: Set of facts $\mathcal{F}$, Description $\mathcal{D} = (w_1, w_2, \ldots, w_n)$
Result: Ordered sequence of fact aligned description $\tilde{\mathcal{D}}$

```
for t ∈ {1, ..., n} do
    factual_flag ← False
    for f ∈ F do
        if w_t ∈ V_f and factual_flag = False then
            d̃_t ← (w_t, f)
            factual_flag ← True
        end
    end
    if factual_flag = False then
        if w_t ∈ V then
            d̃_t ← (w_t, NaF)
        else
            d̃_t ← (UNK, NaF)
        end
    end
end
D̃ = (d̃_1, d̃_2, ..., d̃_n)
```

Given the input facts $\mathcal{F}$ and the fact-aligned description $\tilde{\mathcal{D}}$, the model maximizes the log-likelihood of the observed words in the description with respect to the model parameters θ,

$$\theta^* = \arg\max_{\theta} \log P(\tilde{\mathcal{D}} \mid \mathcal{F}), \tag{13}$$

which can be further decomposed as

$$\log P(\tilde{\mathcal{D}} \mid \mathcal{F}) = \sum_{t=1}^{|\tilde{\mathcal{D}}|} \log P(w_t \mid w_{1:t-1}, \mathcal{F}). \tag{14}$$

Since the log-likelihood of the word w_t also depends on the underlying fact selection, we can further decompose $P(w_t)$ as

$$P(w_t) = P(w_t \mid f_t, w_{1:t-1})\, P(f_t \mid w_{1:t-1}) \tag{15}$$

Therefore, we train our model end-to-end by optimizing the following objective function:

$$\mathcal{L}(\theta) = -\sum_{t=1}^{|\mathcal{D}|} \log P(w_t \mid f_t, w_{1:t-1}) - \sum_{t=1}^{|\mathcal{D}|} \log P(f_t \mid w_{1:t-1}). \tag{16}$$

Note that the alignment $\tilde{\mathcal{D}}$ of the description $\mathcal{D}$ to the facts $\mathcal{F}$ provides a ground truth fact f_t as each time step t during training.

3 EVALUATION

3.1 Baselines

We compare the performance of our model against a number of competitive baselines, including recent state-of-the-art deep neural methods.

3.1.1 Dynamic Memory-based Generative Model. We use the model proposed by Bhowmik and de Melo [3] as one of our baseline methods, because that model claims to solve essentially the same problem using a dynamic memory-based generative model, achieving state-of-the-art results. The dynamic memory component memorizes

Table 1: Experimental results for the WikiFacts10K-Imbalanced benchmark dataset

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR	CIDEr
Dynamic Memory-based Generative Model	61.1	53.5	48.5	46.1	64.1	35.3	3.295
Fact2Seq w. Attention	62.8	56.3	53.0	52.7	63.0	35.0	3.321
NKLM	34.7	27.7	29.1	29.0	44.1	20.1	1.949
Our Model	**61.9**	**57.3**	**55.6**	**54.3**	**64.9**	**36.3**	**3.420**
– without Positional Encoder	58.9	51.9	46.3	42.5	63.4	33.7	3.126
– without Mean Fact	58.0	52.2	49.5	48.6	64.3	34.8	3.139
– Copy Only	26.8	21.2	16.5	11.8	45.3	20.6	1.766

Table 2: Experimental results on the WikiFacts10K-OpenDomain benchmark dataset

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR	CIDEr
Dynamic Memory-based Generative Model	57.8	50.7	43.6	39.7	67.6	34.9	3.556
Fact2Seq w. Attention	62.7	56.3	50.0	46.2	67.7	35.8	3.629
NKLM	47.9	42.1	37.5	32.3	57.4	25.9	2.958
Our Model	**68.2**	**61.8**	**56.6**	**51.9**	**70.0**	**37.3**	**4.084**
– without Positional Encoder	66.2	58.4	51.8	45.9	67.9	34.7	3.717
– without Mean Fact	62.4	58.0	55.1	51.8	67.7	36.0	3.856
– Copy Only	37.5	26.4	18.4	10.9	55.2	24.8	2.136

how much information from a particular fact is used by the previous memory state and how much information of a particular fact is invoked in the current context of the output sequence. However, through our experiments and analysis, we show that this approach is unable to obtain precise open-domain descriptions with arbitrary tokens. We use the original implementation[2].

3.1.2 Fact-to-Sequence with Attention. The comparison with this baseline method can be deemed as an ablation study, in which the decoder has no access to the copy mechanism. This baseline resembles the standard sequence-to-sequence with attention mechanism proposed by Bahdanau et al. [2]. However, unlike in their work, the input to this model consists of positionally encoded discrete facts.

Each word in the output sequence is predicted by providing the attention-weighted fact embeddings and the previous hidden state as an input to the GRU decoder. Then, the output of the decoder is concatenated with the attention-weighted fact embeddings and passed through a 2-layer feed-forward network with a ReLU activation. Finally, the output of the MLP is fed to a softmax classifier that outputs a probability distribution for the combined vocabulary of factual and non-factual words $\mathcal{V} \cup \{\bigcup_{s \in \mathcal{E}} \mathcal{V}_f^s\}$. The word with the maximum probability is emitted by the decoder.

Note that Bhowmik and de Melo [3] also implements a "Fact2Seq w. Attention" baseline. However, one key difference is that the baseline considered here has a skip-connection from the attention-weighted fact embeddings, concatenating them with the output of the GRU decoder that is then passed through a 2-layer feed-forward network. The experimental results suggest that our variant with this skip-connection leads to substantially more accurate descriptions.

[2]https://github.com/kingsaint/Open-vocabulary-entity-type-description

3.1.3 Neural Knowledge Language Model (NKLM). We compare against Ahn et al.'s Neural Knowledge Language Model [1], which is able to generate Wikipedia-style multi-sentence summary paragraphs for movie actors. Although the descriptions in our case are much shorter, we adopted their model as a representative baseline for methods yielding multi-sentence summaries, as the tasks are similar in nature. NKLM also adopts a copy mechanism but the decision about whether to copy is made by a binary gating variable that is provided as an additional label by augmenting the dataset during training. By predicting whether the word to generate has an underlying fact or not, the model can generate such words by copying from the selected fact. On the contrary, our model decides to copy whenever a fact other than the mean fact is selected. We implemented and trained their model with the benchmark datasets. Their model also requires an alignment of descriptions to the underlying facts. Additionally, NKLM relies on pre-trained TransE [4] embeddings of objects and relations to obtain the fact embeddings that are provided as inputs to the model.

3.2 Dataset

For our experiments, we rely on two benchmark datasets, each of which consists of 10K entities with at least 5 facts and an English description. The first of these datasets, denoted as *WikiData10K-Imbalanced*, is the one used in Bhowmik & de Melo (2018) [3]. However, since the entities in that dataset were randomly sampled from the Wikidata RDF dump [7], the ontological types of the sampled entities have a long-tail distribution, while an overwhelming 69.17% of entities are instances of *human*. This highly skewed distribution makes the dataset biased towards a particular domain. To decrease the percentage of such entities in the dataset, we created a new dataset, *WikiData10K-OpenDomain*, in which the instances of

WikiFacts10k-OpenDomain		WikiFacts10k-Imbalanced	
Instance of	Pct.	Instance of	Pct.
human	11.45%	human	69.17%
painting	8.73%	painting	2.02%
commune of france	6.41%	commune of france	1.43%
film	6.27%	scientific article	1.36%
scientific article	5.56%	film	1.34%
encyclopedic article	3.30%	encyclopedic article	0.89%
asteroid	2.89%	asteroid	0.62%
taxon	2.58%	taxon	0.57%
album	2.19%	road	0.52%
road	2.02%	album	0.46%

Table 3: Frequency distribution of the top-10 domains in the two datasets.

human are downsampled to 11.45% to accommodate more instances of other ontological types as evinced by the frequency distribution in Table 3. These datasets, both available online[3], are split into training, dev., and test sets in a 80:10:10 ratio.

3.3 Metrics

Following previous work, we use the automatic evaluation metrics BLEU (1 - 4) [19], ROUGE-L [15], METEOR [13], and CIDEr [23] for a quantitative evaluation of the generated descriptions with respect to the ground truth descriptions provided in the benchmark data. These metrics are widely used in the literature for the evaluation of machine translation, text summarization, and image captioning. BLEU is a precision-focused metric, ROUGE-L is a recall-based metric, METEOR uses both precision and recall with more weight on recall than precision, and CIDEr considers TF-IDF-weighted n-gram similarities. Following standard practice, as in the official implementation[4], the raw scores for CIDEr are multiplied by a factor of 10 for better readability, yielding values in $[0, 10]$. For the same reason, following standard practice, the scores for other metrics are multiplied by 100, yielding scores in the range of $[0, 100]$. Note that typically these metrics rely on more than one human-written ground truth output per instance for evaluation. The lack of any alternative descriptions in the benchmark datasets implies that the generated descriptions are each evaluated on a single ground truth description. Additionally, these metrics take a very conservative approach in that they look for overlapping words, word alignment, longest common subsequence etc.

3.4 Experimental Setup

Our model and all other baselines are trained for a maximum 25 epochs. We report the results of the best performing models on the dev set. For NKLM and the dynamic memory-based generative model, we use the default hyper-parameter settings used by the authors. For NKLM, following the authors, we obtain the fact embeddings by concatenating the embeddings of object entity and relation that are obtained by using the TransE embedding model. For our model and the fact-to-sequence with attention baseline, we fix the embedding dimensions of facts and words to 100. The hidden layers of the GRU and the 2-layer feed-forward networks are 100-dimensional. We use Adam as our optimizer with a fixed learning rate of 0.001. We fix the maximum number of facts to 60 including the mean fact. For instances with less that 60 facts, we resort to a suitable masking to obtain the attention scores of the relevant facts. The maximum number of factual words for each fact is limited to 60.

We use the 1,000 most frequent words in the dataset as our default vocabulary. One can also consider using a much larger vocabulary of frequent English words. However, for our experiments we found it unnecessary. Still, because of this restricted vocabulary, our model may occasionally sparsely generate <UNK> tokens, which we remove from the generated description.

The datasets and the PyTorch implementations of all our experiments are available online.[5]

3.5 Results

Tables 1 and 2 provide the evaluation results of our model and the baselines on the WikiFacts10k-Imbalanced and WikiFacts10k-OpenDomain datasets, respectively. Our model outperforms the state-of-the-art dynamic memory-based generative model by more than 8 BLEU-4 points on the imbalanced dataset. We also observe 1 to 7 point gains on the scores across all other metrics. Similar trends are observed for the more challenging WikiFacts10k-OpenDomain dataset, in which our model improves upon the dynamic memory-based generative model by 12.2 points in terms of the BLEU-4 metric. For this dataset, we also observe 5 to 8 point gains across all other metrics. A substantial gain in the BLEU scores indicates that our model can generate longer sequences more accurately than the state-of-the-art model.

Furthermore, we observe that our attention-based fact-to-sequence model also outperforms the dynamic memory-based generative model in most of the metrics across both the datasets. This observation shows that one can do without a dynamic memory module for memorizing fact specific information. It turns out that a GRU-based architecture with an intelligently chosen encoding of the input and an extra skip-connection in the decoder has sufficient memory to retain fact-specific information required to generate words in the output.

The NKLM model by Ahn et al. obtains sub-par results on both datasets, particularly the WikiFacts10k-Imbalanced data, although the original paper focused on descriptions of humans. This shows that their method for generating a short text is unsuitable for our demanding task of generating very concise entity descriptions. Additionally, the pre-trained TransE embeddings required to obtain fact embeddings are not available for previously unseen entities in the test set, thus severely restricting the model's ability to cope with unknown entities.

3.6 Ablation Study

In addition to comparing our method against a number of competitive baselines, we perform a series of ablation studies to evaluate the effect of different components within our model. The results of these experiments are included in Tables 1 and 2.

[3] https://github.com/kingsaint/Wikidata-Descriptions
[4] https://github.com/vrama91/cider
[5] https://github.com/kingsaint/Wikidata-Descriptions

3.6.1 Without Positional Encoder. To understand the effect of the positional encoder, we replace the positional encoder with the average pooling of the constituent word embeddings of the facts to obtain the fact embeddings. The differences between the BLEU scores of our model and this ablated version indicates that the position of the words in the factual phrases indeed plays an important role in encoding a fact that provides a useful signal to the decoder.

3.6.2 Without Mean Fact. To understand the effect of mean fact in our model, we replace the mean fact with a fixed embedding vector of dimension d. The elements of this vector are sampled from a uniform distribution $\mathcal{U}(-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}})$ and do not change during training. The result of this experiment suggests that the role of the mean fact in our model is not of just that of a *sentinel* responsible for generating general vocabulary words. Rather, it encodes some useful semantic information that the model can exploit.

3.6.3 Copy Only. This experiment is an ablation study that we perform to understand how precisely a model can generate descriptions if it only copies factual words. This baseline is a restricted version of our model, in which the model never generates any general vocabulary word. The much inferior performance of this model shows that merely copying factual words to the output does not yield precise descriptions. This further demonstrates the importance of a model that dynamically decides whether to copy words from the input facts or instead emit general vocabulary words.

3.7 Parameter Efficiency

Model	#Parameters
Dynamic memory-based generative model	14,197,741
Fact2Seq w. Attention	14,159,561
Neural Knowledge Language Model (NKLM)	20,569,361
Our Model	979,986

Table 4: Comparison of the number of learnable parameters.

Table 4 shows the number of learnable parameters for our model as well as the baselines. Our model is 14x more parameter-efficient than the competitive baselines. As there are fewer parameters to learn, this drastically improves the average training time of the model as compared to the other baselines. The number of parameters depends on the vocabulary size of the output softmax layer and the input to the word embeddings. Reducing the size of the softmax vocabulary to the most frequent words and making the model copy fact-specific words directly from the input contributes to the parameter efficiency and faster training of the model.

3.8 Importance of Fact Alignment of Descriptions

The alignment of facts to the description teaches the model to choose the right fact for generating a factual word. As shown in Fig. 3, for each word of the generated description, our model precisely selects the relevant underlying fact. The NKLM also shows a similar effect. However, the dynamic memory-based generative model does not always pick the most relevant fact, although it might occasionally generate the right word due to the statistical co-occurrence of the words in the training set.

3.9 Significance of the Copy Mechanism

The copy mechanism enables our model to copy fact-specific rare or previously unseen words directly to the output sequence, generating factually correct descriptions. To demonstrate the positive effect of this copy mechanism, we select instances from a subset of 8 ontological types that tend to require explicit copying of factual words. This subset accounts for 19.6% of the test set. Table 6 shows some examples from this subset. We also perform an automatic evaluation on this subset and provide the results in Table 5. Both the models with copy mechanism significantly outperform the baselines lacking any explicit copy mechanism. These results demonstrate that a suitable copy mechanism generates far more precise open-domain descriptions.

4 RELATED WORK

In the following, we review previous research and describe how it differs from the task and approach we consider in this paper.

4.1 Text Generation from Structured Data

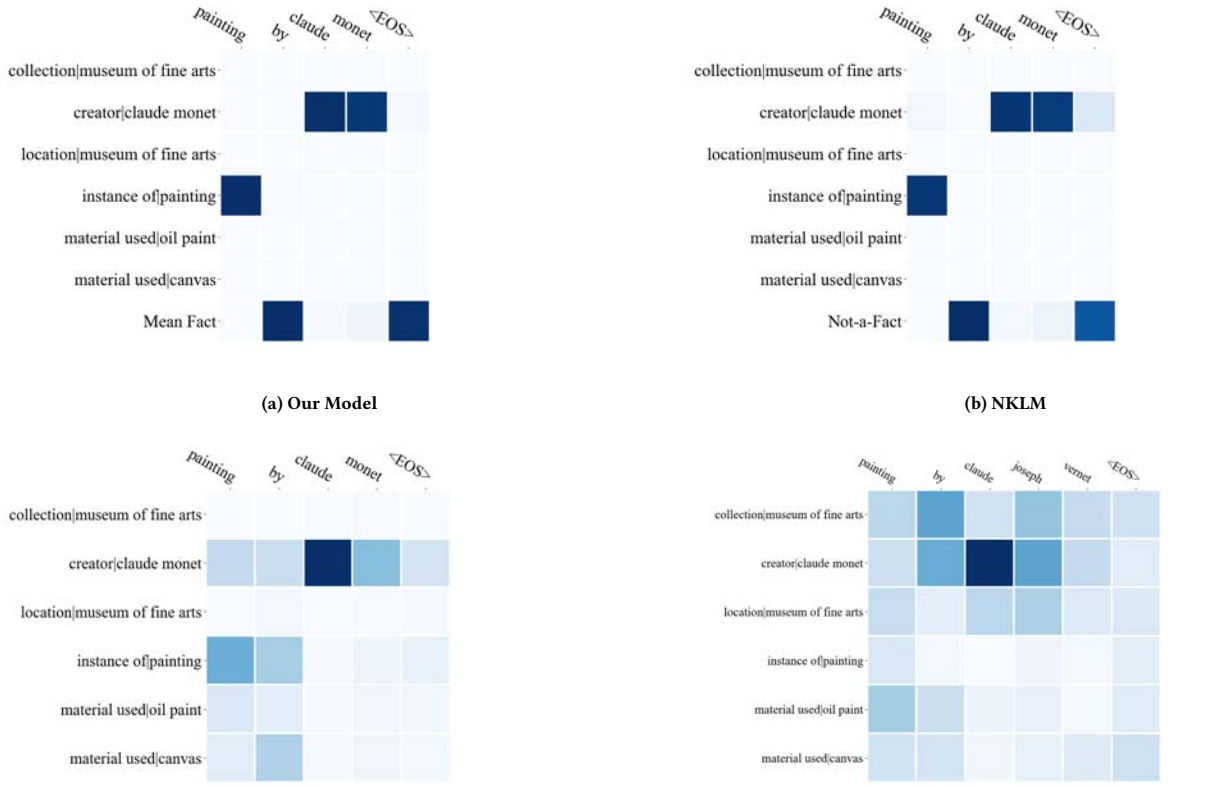
Lebret et al. [14] take Wikipedia infobox data as input and train a neural language model that, conditioned on occurrences of words in the input table, generates biographical sentences as output. In a similar vein, Ahn et al. [1] infused factual knowledge into an RNN-based language model to generate Wikipedia-style summary paragraphs of film actors. Similar to ours, their approach also uses a copy mechanism to copy specific words from the input facts to the description. However, these approaches are not directly compatible with our problem setting, which focuses on generating synoptic, rather than detailed multi-sentence descriptions. Additionally, in contrast to our setting, which requires dynamically considering a wide variety of domains and entity types, the previous studies consider just human biographies as a single domain. Our experiments show that our method substantially outperforms the approach by Ahn et al. on our task.

The WebNLG Challenge [8] is another task aiming at generating text from RDF triples. However, it differs quite substantially from the task we study in this paper, as it demands a textual verbalization of every single triple. Our task, in contrast, requires synthesizing a short synoptic description by precisely selecting the most relevant and distinctive facts from the set of all available facts about the entity.

Bhowmik and de Melo (2018) [3] introduced a dynamic memory-based neural architecture that generates type descriptions for Wikidata items. Although the study has similar aims as the present one, the approach proposed there mainly exploits the statistical co-occurrence of words in the target descriptions of the training data. In our experiments, we show that the dynamic memory method is unable to be precise in an open-domain setting, which may require invoking named entities such as *Ehime* for the example in Figure 1.

4.2 Referring Expression Generation

Referring Expression Generation (REG) is a subtask of Natural Language Generation (NLG) that focuses on the creation of noun



(a) Our Model **(b) NKLM**

(c) Fact2Seq **(d) Dynamic Memory-based Generative Model**

Figure 3: A visualization of attention weights for selecting relevant facts.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR	CIDEr
Dynamic Memory-based Generative Model	39.5	31.9	22.3	14.8	55.9	23.6	2.319
Fact2Seq w. Attention	52.8	44.5	33.5	24.0	56.9	24.1	2.337
Neural Knowledge Language Model	74.3	68.3	**63.1**	**57.2**	73.9	40.1	5.348
Our Model	**75.4**	**69.1**	62.8	55.5	**76.8**	**41.4**	**5.387**
- Copy Only	42.7	26.3	15.4	0.0	57.9	27.1	2.089

Table 5: Experimental results for the subset of ontological types that require explicit copying of factual words.

Instance Of	Ground Truth Description	Generated Description	
Album	album by hypocrisy	Dynamic Memory	album by michelangelo
		Fact2Seq	album by song
		NKLM	album by hypocrisy hypocrisy
		Our Model	album by hypocrisy
	album by canadian country music group family brown	Dynamic Memory	czech hong kong by rapper laurana
		Fact2Seq	sparta album
		NKLM	album by family brown
		Our Model	album by family brown
Book	science fiction novel by richard k morgan	Dynamic Memory	science fiction novel by
		Fact2Seq	episode of a book by bernard dyer
		NKLM	fiction by richard k
		Our Model	novel by richard k
Painting	painting by hendrick cornelisz van vliet	Dynamic Memory	painting by cornelis de vos
		Fact2Seq	painting by abraham van (ii) switzerland
		NKLM	painting by hendrick cornelisz
		Our Model	painting by hendrick cornelisz van vliet
	painting by eustache le sueur	Dynamic Memory	painting by thomas hovenden
		Fact2Seq	painting by california
		NKLM	painting by eustache le
		Our Model	painting by le sueur sueur
Road	highway in new york	Dynamic Memory	highway in new york
		Fact2Seq	highway in new york
		NKLM	<UNK> highways york
		Our Model	highway in new york
	road in england	Dynamic Memory	area in the london borough of croydon
		Fact2Seq	of in london
		NKLM	road road in the church of england
		Our Model	road in the area london
Sculpture	sculpture by antoine coysevox	Dynamic Memory	sculpture by frederick william pomeroy
		Fact2Seq	sculpture by unknown singer
		NKLM	sculpture by antoine coysevox
		Our Model	artwork by antoine coysevox
	sculpture by donatello	Dynamic Memory	by henry and final by the carducci
		Fact2Seq	sculpture by statue kreis comedy
		NKLM	sculpture by donatello donatello nilo
		Our Model	sculpture by donatello
Single/ Song	1967 gilbert bécaud song	Dynamic Memory	2014 by 2012 of the czech 2014 by czech
		Fact2Seq	song
		NKLM	song by gilbert bécaud
		Our Model	song by gilbert bécaud
	single	Dynamic Memory	song by dutch by 2014 municipality
		Fact2Seq	1980 song by northern song
		NKLM	single by michael jackson
		Our Model	song by michael jackson
Street	street in boelenslaan	Dynamic Memory	street in richard
		Fact2Seq	street in collection
		NKLM	street in boelenslaan
		Our Model	street in achtkarspelen
	street in echt	Dynamic Memory	street in singer
		Fact2Seq	street in one
		NKLM	street in echt
		Our Model	street in echt

Table 6: Examples from the subset of ontological types that benefits from copy mechanism.

Item	Instance of	Generated Description
Q11584386	Human	japanese tarento
Q2198428	Human	netherlands businessperson
Q3260917	Human	french military personnel
Q1494733	Painting	painting by august macke
Q16054316	Painting	painting by liselotte schramm-heckmann
Q15880468	Painting	painting by emile wauters
Q10288648	Book	book by izomar camargo guilherme
Q10270545	Book	novel by antonin kratochvil
Q10272202	Book	novel by jose louzeiro
Q1001786	Street	street in budapest
Q10552208	Street	street in orebro
Q10570752	Street	street in malmo municipality

Table 7: Examples of generated descriptions for the Wikidata entities with missing descriptions.

phrases that identify specific entities. The task comprises two steps. The *content selection* subtask determines which set of properties distinguish the target entity, and the *linguistic realization* part defines how these properties are translated into natural language. There is a long history of research on generating referring expressions. In one of the recent approaches, Kutlak et al. [12] convert property–value pairs to text using a hand-crafted mapping scheme. However, their method requires specific templates for each domain. Applying template-based methods to open-domain knowledge bases is extremely challenging, as this would require too many different templates for different types of entities. Recently, Ferreira et al. [5] proposed an end-to-end neural approach to REG called NeuralREG. They used a delexicalized WebNLG corpus for the training and evaluation of their model. NeuralREG generates a delexicalized template of the referring expression.

4.3 Neural Text Summarization

Generating descriptions for entities is related to the task of text summarization in that the salience of information needs to be assessed [27]. Similar to abstractive summarization, our task requires the generation of words not seen in the inputs. At the same time, in a similar vein to extractive summarization, our task also requires a selection of words from input facts for copying to the output sequence. The surge of sequence-to-sequence language modeling via LSTMs naturally extends to the task of abstractive summarization by training a model to accept a longer sequence as input and learning to generate a shorter compressed sequence as a summary. To this end, Rush et al. [20] employed this idea to generate a short headline from the first sentence of a text. Recently, Liu et al. [16] presented a model that generates an entire Wikipedia article via a neural decoder component that performs abstractive summarization of multiple source documents. Our work differs from such previous work in that we do not consider a text sequence as input. Rather, our inputs are a series of property–value pairs deemed as facts.

4.4 Pointer Networks and Copy Mechanisms

In order to learn how to solve combinatorial optimization problems that involve output dictionaries of varying sizes, such as the traveling salesman problem, Vinyals et al. [24] proposed a deep neural architecture known as *Pointer Networks*. Such networks rely on an attention mechanism to repeatedly select elements from the input as output. Subsequent works [9, 10, 18] incorporated this idea into hybrid text-based sequence-to-sequence models that occasionally select words from the input and otherwise generate words from a regular output dictionary. This addresses how to cope with rare and unknown words in the desired output sequence, which constitutes one of the key challenges in deploying neural text generation in practice. Since it is not feasible to directly learn from existing examples how to generate all possible vocabulary words that might be needed in the output, oftentimes, it is easier to learn to directly select suitable words from the given input. See et al. [21] investigated the use of such architectures for the task of abstractive text summarization, so as to better cope with long input texts. As mentioned, Ahn et al. [1] rely on a similar copy mechanism to transform facts into summaries.

While our model adopts similar principles, there is a significant difference in our copy mechanism. In the above-mentioned existing works, the pointing and copying is a one-step process, in which a token within a context window is chosen based on simple attention weights. In our model, the pointing and copying is a two-step process. The model first needs to identify a pertinent fact that is salient enough to merit consideration. Then, within the chosen fact, it selects a suitable word for copying. The context window also varies depending on the number of facts and the number of words within a fact. In our experiments, we show that our approach greatly outperforms the model with copy mechanism proposed by Ahn et al.

5 CONCLUSION

Synoptic descriptions of entities provide end-users a near-instantaneous understanding of the identity of an entity. Generating such descriptions requires the selection of the most discernible facts and composing these facts into a coherent sequence. In this paper, we introduced a fact-to-sequence encoder–decoder model with a custom copy mechanism. Through an extensive evaluation, we have shown that our method consistently outperforms several competitive baselines on two benchmark datasets. Additional analysis also shows that our model can precisely choose facts and copy factual words effectively. The success of this novel architecture for generating concise descriptions suggests that it could be adapted for additional applications that may benefit from a repository of structured facts, e.g. dialogue systems and knowledge graph-driven question answering.

ACKNOWLEDGMENTS

Gerard de Melo's research is in part supported by the Defense Advanced Research Projects Agency (DARPA) and the Army Research Office (ARO) under Contract No. W911NF-17-C-0098. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA and the ARO.

REFERENCES

[1] Sungjin Ahn, Heeyoul Choi, Tanel Pärnamaa, and Yoshua Bengio. 2016. A Neural Knowledge Language Model. *CoRR* abs/1608.00318 (2016). arXiv:1608.00318 http://arxiv.org/abs/1608.00318

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014). arXiv:1409.0473 http://arxiv.org/abs/1409.0473

[3] Rajarshi Bhowmik and Gerard Melo. 2018. Generating Fine-Grained Open Vocabulary Entity Type Descriptions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 877–888. http://aclweb.org/anthology/P18-1081

[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2787–2795. http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data.pdf

[5] Thiago Castro Ferreira, Diego Moussallem, Ákos Kádár, Sander Wubben, and Emiel Krahmer. 2018. NeuralREG: An end-to-end approach to referring expression generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1959–1969. http://aclweb.org/anthology/P18-1182

[6] Gerard de Melo and Gerhard Weikum. 2010. MENTA: Inducing Multilingual Taxonomies from Wikipedia. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management (CIKM 2010)*, Jimmy Huang, Nick Koudas, Gareth Jones, Xindong Wu, Kevyn Collins-Thompson, and Aijun An (Eds.). ACM, New York, NY, USA, 1099–1108.

[7] Fredo Erxleben, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandečić. 2014. Introducing Wikidata to the Linked Data Web. In *Proceedings of the 13th International Semantic Web Conference (ISWC'14) (LNCS)*, Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig A. Knoblock, Denny Vrandečić, Paul T. Groth, Natasha F. Noy, Krzysztof Janowicz, and Carole A. Goble (Eds.), Vol. 8796. Springer, 50–65.

[8] Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG Challenge: Generating Text from RDF Data. In *Proceedings of the 10th International Conference on Natural Language Generation*. Association for Computational Linguistics, Santiago de Compostela, Spain, 124–133. http://www.aclweb.org/anthology/W17-3518

[9] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, 1631–1640. http://www.aclweb.org/anthology/P16-1154

[10] Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the Unknown Words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 140–149. https://doi.org/10.18653/v1/P16-1014

[11] Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, Edwin Lewis-Kelham, Gerard de Melo, and Gerhard Weikum. 2011. YAGO2: Exploring and Querying World Knowledge in Time, Space, Context, and Many Languages. In *Proceedings of the 20th International World Wide Web Conference (WWW 2011)*, Sadagopan Srinivasan, Krithi Ramamritham, Arun Kumar, M. P. Ravindra, Elisa Bertino, and Ravi Kumar (Eds.). ACM, New York, NY, USA, 229–232. https://doi.org/10.1145/1963192.1963296

[12] Roman Kutlak, Kees van Deemter, and Christopher Stuart Mellish. 2013. Generation of Referring Expressions in Large Domains.

[13] Alon Lavie and Abhaya Agarwal. 2007. Meteor: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation (StatMT '07)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 228–231. http://dl.acm.org/citation.cfm?id=1626355.1626389

[14] Rémi Lebret, David Grangier, and Michael Auli. 2016. Generating Text from Structured Data with Application to the Biography Domain. *CoRR* abs/1603.07771 (2016). arXiv:1603.07771 http://arxiv.org/abs/1603.07771

[15] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*. 10. http://research.microsoft.com/~cyl/download/papers/WAS2004.pdf

[16] Peter Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating Wikipedia by Summarizing Long Sequences. *CoRR* abs/1801.10198 (2018). http://arxiv.org/abs/1801.10198

[17] Xiang Long, Chuang Gan, and Gerard de Melo. 2018. Video Captioning with Multi-Faceted Attention. *Transactions of the Association for Computational Linguistics (TACL)* 6 (2018), 173–184. https://transacl.org/ojs/index.php/tacl/article/view/1289

[18] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer Sentinel Mixture Models. *CoRR* abs/1609.07843 (2016). arXiv:1609.07843 http://arxiv.org/abs/1609.07843

[19] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. http://aclweb.org/anthology/P02-1040

[20] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685* (2015).

[21] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. 1073–1083. https://doi.org/10.18653/v1/P17-1099

[22] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-To-End Memory Networks. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 2440–2448. http://papers.nips.cc/paper/5846-end-to-end-memory-networks.pdf

[23] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2014. CIDEr: Consensus-based Image Description Evaluation. *CoRR* abs/1411.5726 (2014). arXiv:1411.5726 http://arxiv.org/abs/1411.5726

[24] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer Networks. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 2692–2700. http://papers.nips.cc/paper/5866-pointer-networks.pdf

[25] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Commun. ACM* 57, 10 (Sept. 2014), 78–85. https://doi.org/10.1145/2629489

[26] Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic Memory Networks for Visual and Textual Question Answering. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. JMLR.org, 2397–2406. http://dl.acm.org/citation.cfm?id=3045390.3045643

[27] Qian Yang, Yong Cheng, Sen Wang, and Gerard de Melo. 2017. HiText: Text Reading with Dynamic Salience Marking. In *Proceedings of WWW 2017 (Digital Learning Track)*. ACM.

Navigating the Maze of Wikidata Query Logs

Angela Bonifati
Lyon 1 University
angela.bonifati@univ-lyon1.fr

Wim Martens
University of Bayreuth
wim.martens@uni-bayreuth.de

Thomas Timm
University of Bayreuth
thomas.timm@uni-bayreuth.de

ABSTRACT

This paper provides an in-depth and diversified analysis of the Wikidata query logs, recently made publicly available. Although the usage of Wikidata queries has been the object of recent studies, our analysis of the query traffic reveals interesting and unforeseen findings concerning the usage, types of recursion, and the shape classification of complex recursive queries. Wikidata specific features combined with recursion let us identify a significant subset of the entire corpus that can be used by the community for further assessment. We considered and analyzed the queries across many different dimensions, such as the robotic and organic queries, the presence/absence of constants along with the correctly executed and timed out queries. A further investigation that we pursue in this paper is to find, given a query, a number of queries structurally similar to the given query. We provide a thorough characterization of the queries in terms of their expressive power, their topological structure and shape, along with a deeper understanding of the usage of recursion in these logs. We make the code for the analysis available as open source.

CCS CONCEPTS

• **Information systems** → **Query log analysis**; *Query languages for non-relational engines*; *World Wide Web*; *Information retrieval query processing*; • **Theory of computation** → *Database query languages (principles).*

KEYWORDS

Query Log Analysis; Query Shapes; Query Similarity Search; SPARQL endpoint; Knowledge Graph

ACM Reference Format:
Angela Bonifati, Wim Martens, and Thomas Timm. 2019. Navigating the Maze of Wikidata Query Logs. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313472

1 INTRODUCTION

Wikidata [31] is a free collaborative knowledge base that has been characterized by a gigantic growth in terms of number of edits, number of users and developers, and amount of automated software since its inception in 2012 by the Wikimedia Foundation. The interplay between user and bot activities on Wikidata is for instance an interesting subject to study in order to make sense of the quality of the newly added items produced by the massive numbers of edits in the knowledge base [27]. Contrarily to Wikidata data dumps, which are readily available and allow a flurry of analyses, the activity of both humans and bots on the Wikidata SPARQL endpoints can only be investigated since recently, thanks to the release of large anonymized query logs.[1] These query logs represent a rich swath of information about the robotic and organic query traffic on Wikidata and deserve our attention for further investigation, in particular to understand the structure of complex queries. A preliminary analysis of the Wikidata query logs bootstrapped with a recent paper by Malyshev et al. [20], who first introduced the Wikidata SPARQL service WDQS[2] and pinpointed its technical characteristics and current usage. They also provided a classification of the Wikidata queries into robotic and organic requests that we readily adopt in the present paper and that we recapitulate in Section 9. They made several observations on which we build in this paper, namely that robotic query traffic dominates organic query traffic in terms of volume and query load, and that robotic queries are issued by a single source whereas organic queries are typically multi-source. They also identified a massive presence of recursive queries in these logs, whose prominent fragment consists of queries only containing joins and property paths,[3] also known as conjunctive 2-way regular path queries (C2RPQs) in the literature. These massive logs of recursive queries are the first encountered so far, as opposed to negligible percentages of these queries in previous large-scale logs, including DBpedia queries [9].

C2RPQs are the basic building blocks of graph query languages in the literature of RDF and graph databases. They allow to express navigational patterns on the graph instances by leveraging regular expressions, also known as Property Paths in the SPARQL 1.1 specification [15]. In Wikidata, they are particularly important since they emulate ontological reasoning in SPARQL and also express complex label-constrained reachability queries.

We focus our analysis on these recursive queries in the Wikidata logs and further extend the class of C2RPQs by incorporating the use of Service, Values, Bind, Filter, and Optional. Indeed, Service, Values, and Bind occurred only rarely in other massive logs [9] but are very prominent here. This fragment, which we call C2RPQ+, constitutes more than 85% of the valid queries in the logs.[4] The consideration of this fragment leads us to redesign many tests conducted in [9], among which the triple count and shape analysis, which were based on the less expressive *conjunctive queries*.

Although the distinction between robotic and organic query traffic has been introduced in [20], it has never been used in the

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313472

[1] Accompanying the publication by Malyshev et al. [20], the Knowledge-Based Systems Group at TU Dresden released anonymised logs of several hundred million SPARQL queries from the Wikidata SPARQL endpoint [6].
[2] Whose user interface can be found at https://query.wikidata.org/, while the raw SPARQL endpoint is at https://query.wikidata.org/sparql
[3] After removal of subqueries that correspond to Wikidata's labeling service. They also allow the use of Values, which indeed can be done to some extent, see Section 6.
[4] In fact, the 85% only refers to a subfragment that is *suitable for graph shape analysis.*

complex analysis that we carry out in this paper regarding the types of property paths, the computation of triples when property paths are present and the shape analysis of the C2RPQ+ fragment. We also provide a view from rather different angles of these logs by considering them with or without duplicates and by separating the analysis of successfully executed and timeout queries, the latter being analyzed for the first time in our study. We also developed a query similarity search tool capable of identifying from an initial query the set of structurally similar queries by using tree-edit distance. This tool allows to further inspect the logs by having a specific query in mind and in a sense permits to reproduce and reapply the previous complex analysis to the obtained sets for future studies.

We focus in this study on the following research questions: What is the distribution of query sizes? Which qualifiers are popular in queries? How are property paths used and what is their structure? How prominent are conjunctive queries (and variants thereof)? What is the shape and (hyper)treewidth of queries? Given a query, can we find in a subset of the logs the queries that are structurally similar to it? Furthermore, we are also interested in meta-questions, such as: Are there differences between robotic and organic queries? Does a comparison with an earlier study [9] allow us to identify some trends?

As a general remark, our study is significantly more extensive than what we can present in this paper. Our code for the analysis is publicly available [11].

2 DATA SETS

Our corpus consists of all queries in the Wikidata query logs that were recently made publicly available [6]. Precisely, the queries considered in this paper have been downloaded on October 12th, 2018. These logs are anonymized and represent queries that were submitted to the Wikidata SPARQL endpoint from June 12th until September 3rd in 2017. The same queries have been considered in the work of Malyshev et al. [20], as discussed in Section 1. We have partitioned these log files in four disjoint sets: queries for which the HTTP request was successful[5], further partitioned in organic (OrganicOK) and robotic queries (RoboticOK); and timeout queries, further partitioned in organic (OrganicTO) and robotic queries (RoboticTO).

Each query in the downloadable log files has an annotation that indicates if it was classified as a bot or user query by Malyshev et al. [20]. We used the same classification.[6] Sometimes we use OK, (resp., TO) to refer to OrganicOK ∪ RoboticOK (resp., OrganicTO ∪ RoboticTO) for brevity. The TO queries have not been considered in the work of Malyshev et al. [20].

Table 1 describes, for each of the log types, its number of queries (Total #Q), number of valid queries, i.e., queries that parse using Apache Jena 3.7.0 (Valid #Q), and the number of valid queries after removal of duplicates (Unique #Q). For duplicate removal, we considered two queries to be the same if they are the same string after whitespace removal.

[5]HTTP code 200.

[6]Our number of queries in Organic is slightly higher than the number of queries reported on the download page of the query logs. We believe that the Dresden file may be incomplete, since we found organic queries in the "all queries" log files that do not show up in the organic subset in [6].

Table 1: The query logs in our corpus

Source	*Total #Q*	*Valid #Q*	*Unique #Q*
RoboticOK	207,505,296	207,464,954	34,523,883
OrganicOK	661,769	651,385	251,994
RoboticTO	33,616	33,465	3,168
OrganicTO	14,528	14,087	8,729
Robotic	207,538,912	207,498,419	34,527,051
Organic	676,297	665,472	260,723
OK	208,167,065	208,116,339	34,775,877
TO	48,144	47,552	11,897
Total	208,215,209	208,163,891	34,787,774

Table 2: Distribution of Select, Ask, Construct, and Describe

		Valid #Q	*Valid %*	*Unique #Q*	*Unique %*
Select	Robotic	206,006,783	99.28%	34,261,882	99.23%
	Organic	664,323	99.83%	260,114	99.77%
Ask	Robotic	1,127,396	0.54%	73,019	0.21%
	Organic	306	0.05%	132	0.05%
Construct	Robotic	188,088	0.09%	24,587	0.07%
	Organic	516	0.08%	217	0.08%
Describe	Robotic	176,152	0.08%	167,563	0.49%
	Organic	327	0.05%	260	0.10%

Throughout the entire paper, we will use the following notation to discuss data sets. Whenever we report a number or a percentage in the format X (Y), the number X refers to the Valid and the number Y refers to the Unique sets of queries. This notation allows the reader to stay informed throughout the paper about the queries that the endpoint actually receives (Valid) and about those without duplicates in this set (Unique).

Already from Table 1 we can make a number of interesting observations. One simple observation is that the robotic logs contain many more duplicates than the organic logs. Indeed, whereas Organic contains 39.18% unique queries, Robotic only contains 16.64% unique queries. A second observation is that, even though queries do not timeout very often, organic queries time out 100 times more often than robotic queries. The fraction of OrganicTO queries to Organic queries is 2.12% (3.35%), whereas the fraction of RoboticTO to Robotic queries is 0.02% (0.01%). In the set of *unique* timeout queries, a whopping 73.37% are organic.

Data Sets for Analysis. We noticed that almost all Describe queries do not have a body. For this reason, and because Describe queries do not have a well-defined semantics, we solely focus from now on in the paper on the Select, Ask, and Construct queries. This subset of the corpus has 207,987,412 valid queries, 34,619,951 of which are unique. The robotic subset contains 207,322,267 (34,359,488) and the organic subset contains 665,145 (260,463) queries. In all cases, these sets encompass more than 99.5% of the total sets in Table 1, see Table 2.

3 BASIC DEFINITIONS

We assume familiarity with SPARQL, but recall the very basics of the language. The presentation of the following definitions is strongly based on Picalausa and Vansummeren [26] and Bonifati et al. [9]. A *SPARQL query Q* can be seen as a tuple of the form

(*query-type*, *pattern P*, *solution-modifier*).

Here, *query-type* is one of Select, Ask, Construct, and Describe. The *pattern* is the central component of the query, which we will discuss in more detail next, and the *solution-modifier* is for performing aggregation, grouping, sorting, duplicate removal, and returning only a specific window (e.g., the first ten) of the solutions returned by the pattern.

Patterns. By $\mathcal{I}$, $\mathcal{B}$, and $\mathcal{L}$ we denote the sets of *IRIs*, *blank nodes*, and *literals* from SPARQL, respectively. Let $\mathcal{V} = \{?x, ?y, ?z, ?x_1, \ldots\}$ be an infinite set of variables, disjoint from $\mathcal{I}$, $\mathcal{B}$, and $\mathcal{L}$. As in SPARQL, we always prefix variables by a question mark. A *triple pattern* is an element of $(\mathcal{I} \cup \mathcal{B} \cup \mathcal{V}) \times (\mathcal{I} \cup \mathcal{V}) \times (\mathcal{I} \cup \mathcal{B} \cup \mathcal{L} \cup \mathcal{V})$. A *property path* is a regular expression over the alphabet $\mathcal{I}$. A *property path pattern* is an element of $(\mathcal{I} \cup \mathcal{B} \cup \mathcal{V}) \times pp \times (\mathcal{I} \cup \mathcal{B} \cup \mathcal{L} \cup \mathcal{V})$, where pp is a property path. A *SPARQL pattern* P is an expression generated from the following grammar:

$$P ::= \quad t \mid pp \mid Q \mid P_1 \text{ And } P_2 \mid P \text{ Filter } R \mid P_1 \text{ Union } P_2 \mid P_1 \text{ Optional } P_2 \mid \text{Bind } X \text{ AS } v \mid \text{Service } sPR \mid \text{Values } tup\ T$$

Here, t is a triple pattern, pp is a property path pattern, Q is again a SPARQL query, and R is a so-called *SPARQL filter constraint*. SPARQL filter constraints R are built-in conditions which can have a SPARQL query predicate is equalled *SPARQL filter constraint*. SPARQL filter constraints R are built-in conditions which can have unary predicates, (in)equalities between variables, and Boolean combinations thereof. Bind associates a unary expression to a single variable v. Service calls a remote service with name n and sends it a pattern P. Finally, Values binds a tuple tup to values in a given table T. We note that property paths (pp) and subqueries (Q) in the above grammar are new features since SPARQL 1.1. We refer to the SPARQL 1.1 recommendation [16] and the literature [26] for the precise syntax of filter constraints and the semantics of SPARQL queries. We write $vars(P)$ to denote the set of variables occurring in P.

We illustrate by example how parts of our definition correspond to real SPARQL queries. The following query comes from Wikidata ("Locations of archaeological sites", from [32]).

```
SELECT ?label ?coord ?subj
WHERE
{ ?subj wdt:P31/wdt:P279* wd:Q839954 .
  ?subj wdt:P625 ?coord .
  ?subj rdfs:label ?label FILTER(lang(?label)="en") }
```

The query uses the property path `wdt:P31/wdt:P279*`, the literal `wd:Q839954`, and the triple pattern `?subj wdt:P625 ?coord`. It also uses a filter constraint. In SPARQL, the And operator is denoted by a dot (and is sometimes implicit in alternative, even more succinct syntax).

4 QUERY SIZES AND OTHER COUNTING MEASURES

We report in Figure 1 the distribution of length of the queries in terms of the number of their triples. Contrarily to other triple count distributions reported in previous studies, we explicitly focus here on triple counts that include property paths. The reason is that a property path pattern of the form `?x wdt:P31/wdt:P279* ?y` is a shorthand notation for *two* triples and we feel that it should be considered as such. We therefore count property path patterns as follows. Let pp be a property path encoded as a regular expression. We say that the *size* of pp is the number of alphabet symbols in the regular expression. For example, the property path wdt:P279/wdt:P279 has size two. The query illustrated in Section 3 contains two triples of size one and one triple of size two.[7]

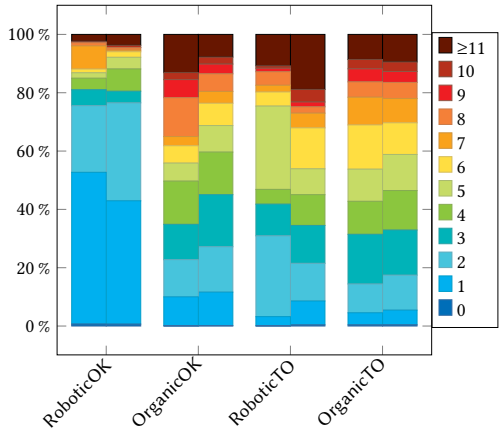


Figure 1: Percentages of queries with corresponding sizes (number of triples plus sizes of property paths) for each dataset – Valid (left) versus Unique (right). The sizes are reported on the right-hand side and range from 0 to ≥ 11.

We analyzed here the triples and property paths inside the Select, Ask, and Construct clauses and explicitly exclude the queries with a Describe clause, as discussed in Section 2. The results for the four data sets can be found in Figure 1. Here, each bar is split in two, where the left side represents the Valid and the right side the Unique version of each data set.

The conclusions of this specific analysis are similar to those made in previous work [7, 20], assessing that user queries are more diverse than robotic ones. Nevertheless, the triple count distributions shown in Figure 1 shows the breakdown in terms of the other dimensions considered in our analysis, encompassing the valid and unique queries and the OK and timeout queries. The timeout queries for instance show a relatively higher complexity in terms of number of triples than the OK ones, and this observation also applies to RoboticTO queries, that are thus quite different from RoboticOK queries. This seems to suggest that timeout queries are queries that failed because of a higher number of triples, which could be interesting to consider in graph query evaluation and optimization studies. The information about the average number of triples confirmed this, since Valid queries have on average 2.58 (2.65) triples, whereas timeout queries have 5.65 (5.94) triples. As a side remark, the highest number of triples that we observed is in the RoboticOK Valid logs and is equal to 67, which was found in 68 queries (in 34 queries in the RoboticOK Valid logs, respectively). The largest size of a property path triple was 19.

Number of Constants and Variables. Counting the number of triples is only one possible measure of the complexity of a query. We enrich this analysis by considering further characteristics of the triples, i.e., whether the triples contain variables or constants. This information is useful for the shape analysis that we conduct in Section 6, in which we show actual differentiations in the obtained shapes by removing or including the constants. We only count the

[7] Notice that the query size computed as described above even with Kleene-star ('*') and transitive closure operators ('+') does not depend on the length of the actual paths in the graph instance when evaluating the query.

prepare for download
save with right click
requires sunburst.css

Table 3: Number of constants and variables (Unique logs)

# Const	# Organic	# Robotic	# Vars	# Organic
0–9	239,003	33,485,122	0–9	255,383
10–19	21,307	480,936	10–19	3,999
20–32	153	393,430	20–27	1,081

the triples, i.e., whether the triples contain variabl
This information is useful for the shape analysis tha
Section 6, in which we show actual differentiations
shapes by removing or including the constants. We
number of distinct variables and constants in the
Table 3 reports the numbers of constants and variat
in the Unique OrganicOK and RoboticOK logs.

Precisely, we counted the number of different var
constants, respectively) of each query in the logs a
total number of queries that have this number. For
aggregate these numbers into intervals in Table 3.
that for numbers of constants and variables greater than 11, the
queries with these numbers of constants are more abundant than
queries with these numbers of variables.

Usage of Wikidata Properties. We then analyzed the organicOK Unique query logs in order to get a feel of the usage of different Wikidata properties (P856, P31,…) appearing in the queries. The total number of properties found in these logs are 881,490. These are divided into two major namespaces at a top level: `www.wikidata.org` (805,196), and `www.w3.org` (70,829). We report in Figure 2 a sunburst diagram showing the segmentation of the Wikidata properties inside the largest top-level namespace qualifier `www.wikidata.org`. In order to avoid clutter and for ease of presentation in the paper, we have solely annotated the sunburst with the properties whose occurrences are above a given threshold (6,000). More views about the sunburst including the other top-level properties and with complete information about all the number of occurrences, is available via an interactive version of the diagram [29]. Hovering over the various segments of the rings provides the information omitted here in order to avoid clutter in Figure 2.

Subqueries and Projection. Roughly 1% of the queries in the Unique Logs use subqueries. This number goes down to 0.37% for the corresponding subqueries in the Valid logs.

We also ran a test for mining the number of queries that use projection. Projection is a cause of complexity increase of query evaluation for CQ queries, that goes from NP-complete is projection is present to PTIME if projection is absent [5, 18]. Similarly to [9] we use the test for projection in Section 18.2.1 in the SPARQL 1.1 recommendation [15]. Out of the valid queries from Table 2, we found 25,569,947 (12.28%) queries that use projection.[8] Out of the valid Organic queries as reported in Table 2, the amount of queries with projection even rises to 28.85%. These percentages become 18.01% and 28.05% for the unique valid and Organic queries, respectively. In particular, they are much higher than the 13.12% Select/Ask queries found within the Unique query logs of the DBpedia corpus in [9].

[8]This is a lower bound, since our test is sound but incomplete.

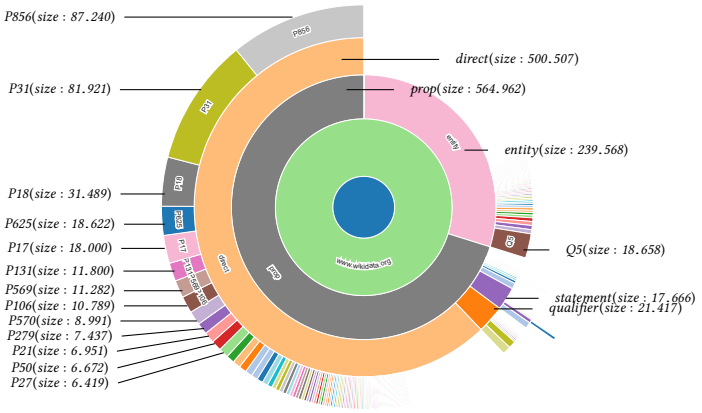


Figure 2: Sunburst distribution of the property qualifiers in Wikidata queries (Unique OrganicOK query logs.). An interactive version is available online at [29].

5 PROPERTY PATH ANALYSIS

Overall, 49,971,258 (13,480,433) queries in our logs use property paths, which amounts to a total of 24.03% (38.94%) of the entire logs. In these queries, we found 165,343 (82,764) property paths in organic queries and 55,168,101 (14,106,489) in robotic ones. (Notice that the same query can contain multiple property paths.) This massive corpus of property paths is the first encountered so far, and significantly (57x) greater than the one found in a large corpus in [9] featuring only 247,404 property paths.

Both organic and robotic property path corpuses are interesting for an analytical study and deserve a deeper inspection in order to classify the occurring path expressions into distinct types. Indeed, when we looked at the *structure* of these path expressions, we found 234 different types of organic expressions, compared to only 64 types of robotic expressions. Such a thorough classification revealed the different characteristics of the organic property paths with respect to the robotic ones, as the former exhibit more variety and heterogeneity than the latter despite their lower occurrences. Here, the *type* of a property path is obtained as follows. We replace each variable or IRI by letters from the alphabet in increasing order. (If a variable or IRI is repeated in the property path, we replace it by the same alphabet letter.) For example, `wdt:P31*/wdt:P279*` is of the type a^*b^* and `wdt:P31/wdt:P31*/wdt:P279*` is of the type aa^*b^*.

Robotic Property Paths. Table 4 contains a summary of the most common types of property paths in robotic queries. The columns with "V" represent results for the *Valid* queries, and the columns with "U" for the *Unique* queries. For succinctness, we aggregated different types together. For example, we aggregated each type with its reverse type. For instance, the row for ab^* also contains the expressions of the form a^*b. Furthermore, we treated $\hat{}a$ ("follow an a-edge in reverse direction") the same as a single label.[9] Finally, we also grouped disjunctions together, denoted by capital letters. In Table 4, a capital letter A denotes a subexpression that matches

[9]The operator $\hat{}$ is used in 0.80% (1.10%) of robotic and 2.03% (3.18%) of organic queries.

Table 4: Structure of property paths for all robotic queries

Expression Type	*AbsoluteV*	*RelativeV*	*AbsoluteU*	*RelativeU*
a^*	27,850,487	50.48%	1,392,865	9.87%
ab^*, a^+	9,417,166	17.07%	2,816,134	19.96%
ab^*c^*	823,153	1.49%	67,502	0.48%
A^*	328,895	0.60%	51,860	0.37%
ab^*c	122,286	0.22%	1,680	0.01%
a^*b^*	62,784	0.11%	608	
abc^*	27,287	0.05%	4,083	0.03%
$a?b^*$	15,893	0.03%	11,999	0.09%
A^+	4,674	0.01%	2,043	0.01%
Ab^*	1,562		674	
Other transitive	1,643		161	
$a_1 \cdots a_k$	13,382,005	24.26%	9,368,442	66.41%
A	3,043,725	5.52%	381,434	2.70%
$A?$	31,150	0.06%	296	
$a_1a_2? \cdots a_k?$	25,872	0.05%	5,940	0.04%
$\hat{}a$	21,202	0.04%	471	
$abc?$	7,620	0.01%	8	
Other non-transitive	697		289	
Total	55,168,101	100%	14,106,489	100%

a *disjunction of at least two symbols*. Empirically, an A either denotes an expression of the form $!a$, $(a|!a)$, or a disjunction of the form $(a_1|\cdots|a_k)$ with $k > 1$. We divided Table 4 into *transitive expressions* (top) and *non-transitive expressions* (bottom). Transitive expressions are those that match arbitrarily long paths (i.e., they use the operators * or $^+$). The empty cells represent values that round down to 0.00%.

Interestingly, we see significant differences between the numbers of expressions in the valid and in the unique sets. Whereas the type a^* accounts for 50.48% of the expressions in the valid data set, this drops to 9.87% in the unique set. On the other hand, concatenations of symbols (type $a_1 \cdots a_k$) represent 24.26% in the valid queries, but over 66% in the unique queries. To further understand this phenomenon, we focused on the valid RoboticOK and OrganicOK and computed the most popular 20 queries containing a property path with Kleene-star. The top most popular robotic query with property paths (having 281,096 occurrences) belong to the second most occurring type in Table 4 since it contains a single path expression of the kind `wdt:P31/wdt:P279`*. The query is in fact a *conjunctive regular path query*. [10]

Organic Property Paths. Table 5 contains results on the Organic datasets. Since we had 234 different types of expressions, we needed to aggregate more aggressively to make the results presentable. We grouped the types into 41 different categories, from which we omitted some in the table due to space restrictions. The main difference with Table 4 is that we also allow a capital letter to denote a single symbol. So, A can denote expressions equivalent to a, $(a_1|\cdots|a_k)$, $!a$, or $(!a|a)$. The other difference is that we grouped the types $(ab)^*$ and $a(bc)^*$ together in the type $\sim(ab)^*$. These expressions stand out from the rest, since they are the *only* type of transitive expressions we found that put length constraints on arbitrarily long paths. Indeed, all other transitive expressions allow paths of arbitrary length, once the length exceeds a certain value. This is not the case for $(ab)^*$, since it only allows paths of even length.

[10] By looking at the top most popular organic query (with 1,778 occurrences), we noticed that it belongs to the first most occurring type in Table 5 and exhibits the same single path expression. As a side remark, we can notice that the latter query is fairly more complex and uses Union, Filter, Bind, and Service clauses.

Table 5: Structure of property paths for all organic queries

Expression Type	*AbsoluteV*	*RelativeV*	*AbsoluteU*	*RelativeU*
AB^*	57,913	35.03%	28,034	33.87%
A^*	41,777	25.27%	22,071	26.67%
ABC^*	6,497	3.93%	3,044	3.68%
a^*b^*	3,330	2.01%	849	1.03%
ab^*c^*	2,704	1.64%	1,172	1.42%
$a^*B_1?b_2? \cdots b_k?$	1,789	1.08%	422	0.51%
$ab\|c^*d$	1,514	0.92%	534	0.65%
$a^*\|b^*$	347	0.21%	253	0.31%
$abCD^*$	283	0.17%	219	0.26%
ab^*c	113	0.07%	90	0.11%
$a^*\|B$	102	0.06%	76	0.09%
$\sim(ab)^*$	101	0.06%	82	0.10%
ab^*c^*d	86	0.05%	72	0.09%
$ab^*\|c$	70	0.04%	59	0.07%
$a^*b?c$	56	0.03%	27	0.03%
$a^*b^*c^*$	32	0.02%	28	0.03%
$ab^*\|b^+a^*$	16	0.01%	12	0.01%
ab^+c	13	0.01%	12	0.01%
$ab^*\|cd^*$	13	0.01%	12	0.01%
a^*bc^*	11	0.01%	11	0.01%
Other transitive	22	0.01%	20	0.02%
$A_1a_2 \cdots a_k$	31,032	18.77%	15,754	19.03%
A	13,248	8.01%	7,592	9.17%
$a_1? \cdots a_k?$	1,938	1.17%	1,470	1.78%
$A?$	1,178	0.71%	302	0.36%
$ab_1? \cdots b_k?$	1,117	0.68%	529	0.64%
$ab\|c$	27	0.02%	5	0.01%
Other non-transitive	14	0.01%	13	0.02%
total	165,343	100.00%	82,764	100.00%

Here, the percentages in the unique sets are quite similar to those in the valid sets. The Organic queries generally contain more challenging property paths to evaluate. On average, organic property paths are also larger than robotic ones. They contain 2.07 (2.01) literals on average, whereas robotic property paths only contain 1.49 (1.89) literals on average. There were even expression types which occurred more often in the TO (timeout) logs than in the OK logs, such as $\sim(ab)^*$. This is interesting, because such expressions are known to be complex (NP-complete) to evaluate under simple path semantics [3].

Additional Insights on Wikidata Property Paths. We conclude the section with a discussion of the differences between Tables 4 and 5 and the results of the property path analysis by Bonifati et al. [9]. The remarkable difference between the present study and the former (which was done on a corpus mainly consisting of DBPedia queries) is that here, a much larger fraction of the queries use property paths. This is probably due to the peculiar characteristics of the Wikidata data. Property paths are often used in queries performing class navigation in Wikidata and emulating ontological reasoning. Wikidata has relatively long paths in the data that are labeled with the same label (and that are popular to query, e.g., InstanceOf paths), whereas DBpedia has comparably shorter paths and is more flat.

The remarkable similarity, however, is in the *structure* of the property paths. Martens and Trautner [21] defined the class of *simple transitive expressions*, which are syntactically very restricted, but covered over 99% of the property paths in the corpus of Bonifati et al. [9]. In our corpus, 1.61% (0.48%) of the robotic and 3.83% (2.72%) of the organic property paths are not simple transitive expressions. The most significant reason why property paths fall out of this

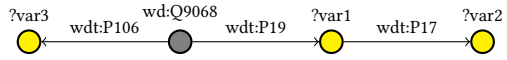


Figure 3: Visualization of a query as a graph

fragment is the use of a^*b^* as a subexpression, whereas simple transitive expressions only use one subexpression with Kleene star.

Furthermore, all property paths except 198 (98) are in C_{tract}, which is a broader class introduced by Bagan et al. [3] and which precisely characterizes the set of regular expressions for which the data complexity under simple path semantics is tractable if P ≠ NP.

6 STRUCTURAL ANALYSIS AND RECURSIVE PROPERTIES

Consider the following query, which searches the occupations (P106) of Voltaire (Q9068), and returns the place (P19) and country (P17) of birth.

```
SELECT * WHERE { wd:Q9068 wdt:P19 ?var1 . ?var1 wdt:P17 ?var2
               . wd:Q9068 wdt:P106 ?var3 }
```

This query can be visualized as a graph, see Figure 3. Answering the query essentially boils down to matching this graph in the data and returning answers for each such match.

This query is an example of a *conjunctive query*, i.e., a query that only uses the And operator. For conjunctive queries, there is an extensive body of research that correlates the shape of the graph with the complexity of their evaluation problem (see, e.g., [10, 13, 17, 33]). In particular, *cycles* in the graph play a major role. In general, conjunctive query evaluation is well known to be NP-complete. On an intuitive level, the reason is that, if a conjunctive query has the shape of a k-clique (which is heavily cyclic), deciding if it returns a non-empty result is equivalent to deciding if the data has a k-clique, which is NP-complete. On the other hand, using Yannakakis' algorithm, we can evaluate *acyclic* conjunctive queries in polynomial time [33].

It is known that even a single cycle can have a significant impact on the run-time of queries [9, 16]. Therefore, for queries that behave similar to conjunctive queries, the shape of their graph is important for understanding their complexity. Furthermore, it gives useful insights on the structure of real queries.

In the following, we will define the *graph shape* of several kinds of queries. We focus exclusively on queries that behave similar to conjunctive queries and for which the cyclicity of the graph shape correlates with the complexity of the evaluation. We start by considering conjunctive queries and add other operators one by one.

Bonifati et al. [9] performed a graph shape analysis on the queries in their corpus, but the corpus does not provide much insight for Wikidata, because it only included 308 Wikidata queries. Furthermore, they only considered shapes of conjunctive queries, extended with (safe use of) Filter and Optional.[11] Since only 25.89% (42.48%) of our robotic and 17.89% (21.15%) of our organic queries are eligible for such an analysis, we extended the analysis to incorporate (the safe use of) four extra features: property paths, Bind, Values, and Service. By doing so, we were able to make the analysis applicable to *2 to 3 times more queries in every fragment*, that is, 85.22% (88.39%) of the robotic and 51.50% (65.22%) of the organic queries. Notice that the queries thereby obtained are not merely subsets of the queries that use a given set of keywords. This obviously implies that the shape test is more sophisticated than the keyword test. We explain how we deal with each keyword separately in Sections 6.2–6.4.

[11] By *safe use* we refer to the use of these operators so that the queries still behave similar to conjunctive queries.

6.1 Graph Patterns and Canonical Graphs

In the following we want to define the *graphs* of different types of queries. These graphs will be undirected, that is, $G = (V, E)$ where V is its (finite) set of nodes and E is its set of edges, where an edge e is a set of one or two nodes, i.e., $e \subseteq V$ and $|e| = 1$ or $|e| = 2$.

We need to determine which queries can be adequately represented as graphs. Since graphs are node pairs and SPARQL queries have triple patterns, we need to be careful with the use of variables in the predicate position. In our corpus, 91.68% (96.43%) of the queries only use triple patterns (s, p, o) where p is an IRI, so these queries can be adequately represented as graphs. We also allow $p \in$ Vars if p is not used elsewhere in the query (in this case, p serves as a wildcard, possibly binding to a value that is returned to the output.). We call such patterns *graph patterns*.

The *canonical graph* of a graph pattern P is the graph $G_P = (V_P, E_P)$ with $E_P = \{\{x, y\} \mid (x, \ell, y)$ is a triple pattern in P and $\ell \in \mathcal{I} \cup \mathcal{V}\}$ and $V_P = \{x \mid (x, \ell, y) \in E_P$ or $(y, \ell, x) \in E_P\}$.

6.2 Conjunctive Queries

Conjunctive queries are the basic building blocks of our shape analysis. In the context of SPARQL, we define them as follows.

Definition 6.1. A *conjunctive query (CQ)* is a SPARQL pattern that only uses the triple patterns and the operator And.

In our corpus, 20.88% (33.69%) of the queries are CQs, which is quite low compared to the study of Bonifati et al. [9]. Part of the reason is that Wikidata queries extensively use Service, most commonly for Wikidata's labeling service [7]. In our logs, 8.38% (12.94%) of the queries use Service in some way.

The good news is that 99.97% (99.997%) of the CQs are graph patterns. We call such CQs *eligible* for graph shape analysis. We note that it is also possible to investigate the shape (or cyclicity) of non-eligible CQs, but we need to consider their *hypergraphs*, see Section 7.

The importance of the shape of conjunctive queries becomes clear in the following result, linking the *treewidth (tw)* of the query's graph to the complexity of query evaluation. The precise definition of treewidth is not important for the paper but, intuitively, treewidth measures *how close the graph is to a tree*. For instance, a tree has treewidth 1 and a k-clique (which is very cyclic) has treewidth k. Queries with tw = 1 are also called *acyclic*.

Theorem 6.2 (cfr. [10, 13, 17]). *Let G be a graph and Q an eligible conjunctive query for which the canonical graph has treewith k. Then it can be tested in time $|G|^{O(k)}|Q|^{O(k)}$ if Q returns a non-empty result on G.*

6.3 Adding Filter and Optional

Bonifati et al. [9] investigated the structure of graph patterns that only use the operators And, Optional, and Filter. In our corpus, 27.72% (44.24%) of the queries are in this fragment. Our extension of CQs with Filter and Optional is very similar to the one of Bonifati et al., that is, we focus on *well-designed* patterns (cf. [25]) with *interface width 1* (cf. [4]). Our main difference is that we now allow binary filter constraints to be edges in the graph.

6.4 Adding Recursion, Bind, Service, and Values

Although the previous tests already classify 53,804,198 (14,649,616) queries to be eligible for graph shape analysis, this is still only 25.87% (42.32%) of our queries. We now discuss how property paths (recursion), Bind, Service, and Values can be incorporated to increase the number of suitable queries to 177,022,071 (30,540,864), or 85.11% (88.22%) of the logs.

Property paths are unproblematic. Conjunctive queries extended with property paths closely correspond to the well known *conjunctive two-way regular path queries (C2RPQs)*, which form a basis of navigational query languages for graphs. Indeed, property paths are very closely related to regular path queries and, due to the ˆ-operator, they can navigate edges in both forward and backward direction, which makes them *two-way*. Although there are some semantical differences between two-way regular path queries and property paths [1, 15, 19], these differences are not crucial for the present analysis.

Definition 6.3. A *conjunctive two-way regular path query (C2RPQ)* is a SPARQL pattern that only uses triple patterns, the operator And, and property paths.

Every C2RPQ that is a graph pattern is suitable. The *graph of a C2RPQ* P is obtained from G_P by adding the edges $\{\{x, y\} \mid (x, pp, y)$ is a property path pattern in $P\}$ to E_P (and adding nodes to V_P if necessary).

Adding Filter and Optional to C2RPQs is analogous to adding them to CQs. The resulting classes of queries are referred to as C2RPQ_F and C2RPQ_OF. As such, we obtain that 46.03% (76.44%) of our queries are C2RPQ_OF queries that are suitable for graph shape analysis.

Concerning Bind, we follow a similar approach to Filter. We call a Bind-condition k-ary if it involves k variables. Unary and binary Bind-conditions are considered to be suitable for graph analysis and are materialized as edges in the graph. Higher-arity Bind-conditions are considered in Section 7.

Values-constructs are essentially used to test if a tuple of variables is in a given set. For instance, the subquery

```
VALUES (?book ?title)
{ (:book1 "Robin Hood")
  (:book2 "Little Red Riding Hood") }
```

emposes a binary constraint and is satisfied when the pair of variables (?book, ?title) can be bound to one of the pairs in the body. In the query corpus, it is used almost exclusively for unary conditions, that is, to test if the value of a single variable is in a given set of constants. We therefore also treat Values similar to Filter and Bind. That is, we call a Values block k-ary if there is a k-ary tuple following the Values keyword. Unary and binary Values subqueries are suitable for graph shape analysis and we extend the graph with an edge for each binary Values constraint. Again, higher arity Values conditions are considered in Section 7.

Service is used extensively in Wikidata queries, most commonly for Wikidata's labeling service. For this reason, Bielefeldt et al. [7] ignore the labeling service entirely in their co-occurrence analysis of SPARQL features. Similarly to before, we say that a Service subquery S is k-ary if it contains k variables. All unary or binary Service subqueries are suitable for graph analysis. When we consider the graph of patterns with Service, we add edges of the form $\{x, y\}$ for all binary Service subqueries, in which x and y are the variables. Higher arity Values conditions are considered in Section 7.

By C2RPQ+ we denote the entire fragment that uses And, Optional, Filter, property paths, Bind, Service, and Values and that is suitable for graph analysis. In total, this amounts to 176,679,495 (30,371,003) robotic and 342,576 (169,861) organic queries, which make up 85.22% (88.39%) and 51.50% (65.22%) of the robotic and organic queries, respectively.

6.5 The Set of Shape Analysis Experiments

We have analyzed the graphs of 8 fragments of queries, namely CQ, C2RPQ, CQ_F, C2RPQ_F, CQ_OF, C2RPQ_OF, $\text{CQ}_{\text{OF+}}$, and C2RPQ+. All fragments were analyzed across three dimensions: *robotic* versus *organic*, *Valid* versus *Unique*, and *with constants* versus *without constants*. In the analyses without constants, we removed all nodes in the graphs that originated from IRIs or literals.[12] Furthermore, for all fragments, we analyzed the time-out (TO) queries separately from the others (OK). This results in 64 separate runs of the shape analysis.

We classified the graphs of queries into the following set of shapes, inspired by [9]. A *chain (of length k)* is graph that is isomorphic to the undirected graph with edges $\{x_0, x_1\}, \{x_1, x_2\}, \ldots, \{x_{k-1}, x_k\}$. As an edge case, we allow chains of length zero, i.e., a single node. A *chain set* is a graph in which every connected component is a chain.

A *tree* is a nonempty undirected graph such that, for every pair of nodes x and y, there exists exactly one undirected path from x to y. A *forest* is a nonempty graph in which every connected component is a tree.

A *star* is a tree for which there exists at most one node with more than two neighbors. Hence, every chain is a star.

We also recall the definition of *flowers* from Bonifati et al. A *petal* is a graph consisting of a source node s, target node t, and a set of at least two node-disjoint paths from s to t. (For instance, a cycle is a petal that uses two paths.) A *flower* is a graph consisting of a node x with three types of attachments: chains (the *stamens*), trees that are not chains (the *stems*), and *petals*. Finally, a *bouquet* is a graph in which every connected component is a flower.

[12] We do not have the space to present all these analyses, but we plan to release a full version on ArXiv as soon as the paper is accepted.

Table 6: Cumulative shape analysis of nonempty graph patterns in C2RPQ+ across the valid logs

C2RPQ+ with constants / Valid				
Shape	*#Queries (Organic)*	*Relative % (Organic)*	*#Queries (Robotic)*	*Relative % (Robotic)*
node (without edges)	0	0.00%	0	0.00%
chain (length ≤ 1)	107,436	31.48%	125,277,683	71.49%
chain	207,292	60.74%	158,150,895	90.25%
star	290,665	85.17%	171,767,410	98.02%
tree	329,701	96.61%	171,885,247	98.09%
flower	335,271	98.24%	172,070,177	98.19%
chain set	209,540	61.40%	158,723,217	90.57%
forest	333,725	97.79%	172,461,994	98.41%
bouquet	339,440	99.46%	172,646,921	98.52%
tw ≤ 2	341,221	99.98%	175,240,211	100.00%
tw ≤ 3	341,268	100.00%	175,240,228	100.00%
tw ≤ 4	341,274	100.00%	175,240,237	100.00%
total	341,274	100.00%	175,240,237	100.00%

Table 7: Cumulative shape analysis of nonempty graph patterns in C2RPQ+ across the valid logs

C2RPQ+ without constants / Valid			
#Queries (Organic)	*Relative % (Organic)*	*#Queries (Robotic)*	*Relative % (Robotic)*
85,601	25.13%	63,048,127	36.58%
200,597	58.89%	155,126,595	89.99%
259,074	76.06%	163,055,757	94.59%
312,128	91.63%	170,205,746	98.74%
321,914	94.50%	170,243,847	98.76%
323,830	95.07%	170,413,306	98.86%
273,992	80.44%	165,014,582	95.73%
337,730	99.15%	172,203,761	99.90%
339,661	99.71%	172,373,220	100.00%
340,632	100.00%	172,375,717	100.00%
340,632	100.00%	172,375,717	100.00%
340,632	100.00%	172,375,717	100.00%
340,632	100.00%	172,375,717	100.00%

6.6 Shape Classification for C2RPQ+

Due to space constraints, we cannot present our complete shape analysis, but we will show and discuss the results on the valid queries in the largest fragment, the C2RPQ+ queries. Furthermore, we give some insights about how the results change for the other fragments. The results for the valid C2RPQ+ queries are in Table 6. We note that some of the queries were empty (0.8% with constants and 2.44% after removing constants); we did not include them.

In Table 6, we see several trends that we also observed in the analysis for the other fragments. First of all, in the shapes that include constants, *stars are quite common*. In the C2RPQ+ queries, 85.17% (87.13%) of the organic and 98.02% (99.30%) of the robotic queries are stars. The number of acyclic queries is even larger: consistently over 99% when constants are absent. As opposed to valid queries, in the timeout logs, the number of cyclic queries significantly increases. For organic CQs, for instance, the number of cyclic queries goes up to about 10%. [13] This number decreases somehow for more complex query fragments, but is still about 7.5% for the unique C2RPQ+ queries (both organic and robotic) and around 3%–4% if constants are removed. Together with the observation from Section 4 that valid queries contain 2.58 triples on average, whereas valid timeout queries have 5.65 triples on average, this suggests that cyclicity and query size play an important role in efficient query evaluation.

The logs strongly confirm a hypothesis that is often stated in theoretical research: the cyclic queries in practical applications are only *mildly cyclic*, i.e., tw $\leq k$ for small values of k. This means that database queries typically do not have large k-cliques encoded in their shape, but remain tree-like. Indeed, the largest treewidth we found in the entire logs was four, for which we found 15 (5) queries.

[13] We omit the exact numbers due to space constraints.

7 HYPERGRAPH ANALYSIS

Hypergraphs generalize graphs in the sense that they allow more than two nodes per edge. As such, the queries that were not suitable for graph shape analysis in Section 6 because they either went beyond *graph patterns* or used Filter, Service, Bind, or Values constraints with arity three or more can be considered here. This amounts to a total of 1,915,550 (1,229,035) $\mathrm{CQ}_{\mathrm{OF+}}$ queries that were not yet analysed in Section 6.

We keep the restriction on well-designed Optional constructs with interface width 1, since for these queries, there still is a correlation between the cyclicity of the hypergraph and complexity of query evaluation [4]. We do not consider queries with property paths in the hypergraph analysis.

A *hypergraph* is a pair $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is its finite set of nodes and $\mathcal{E} \subseteq 2^{\mathcal{V}}$ is a set of *hyperedges*. The *canonical hypergraph of a SPARQL pattern P* is defined as $\mathcal{E} = \{X \mid$ there is a triple pattern t in P such that X is the set of blank nodes and variables appearing $t\}$ and $\mathcal{V}$ is the union of the nodes in the edges in $\mathcal{E}$.

Using the tool `detkdecomp` [12], we analyzed the hypergraphs of all $\mathrm{CQ}_{\mathrm{OF+}}$ queries for which the Optional constructs are well-designed and have interface width one. Overall, we found 590,005 (273,947) remaining queries with hypertreewidth two. All others new queries had hypertreewidth one.

8 QUERY SIMILARITY SEARCH

Massive query logs are valuable sources of information as long as they are made usable for exploration and analysis. In order to improve the usability of the logs, we designed a query similarity search facility that will be released with our code and whose principles and effectiveness are discussed here. Let us first observe that string similarity search is not suitable for Wikidata logs since they have been modified by an anonymization process prior to their release. Hence, one cannot find a query by simply looking at its exact original string or a sufficiently approximate version of it, by applying string edit distance (SED), for instance. So, how can we enable query similarity search in Wikidata query logs?

Given the differentiations of these logs in terms of the structural features of the underlying queries as explained in Section 6, we opted for using the query structure as a yardstick to compare queries in terms of their similarity. This choice solves several problems when comparing queries: comments, prefixes, and variable names do not affect the structure of a query and can be excluded from the computation of query similarities. Furthermore, the anonymization of the Wikidata logs renames all variable names to names with ascending numbers (?var1, ?var2, etc., see Figure 3). Therefore, if a new variable in the query is placed before other variables, it will shift the names of all subsequent variables in the anonymized version, even though they were unchanged in the original query. This could potentially lead the similarity engine to believe that a large modification took place even though it did not. The anonymisation also removes all prefixes and inlines them, hence the impact of changing a single IRI could become large depending on the length of the IRI in case of adoption of a SED measure.

To enable a structural search on Wikidata queries, we translate the abstract syntax tree (AST) of a query into a tree structure as follows. Recall the definition of a *SPARQL query Q* in Section 3

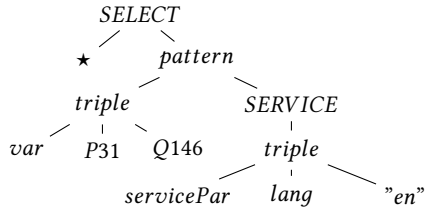


Figure 4: Abstract Syntax Tree of an example query

as a triple (*query-type*, *pattern P*, *solution-modifier*). The *query-type* and the *solution-modifier* of the query respectively become the root and a leaf node in the tree (with a value for the latter), whereas the *pattern P* is a subtree instantiated as a triple t with three leaves ($\langle s, p, o \rangle$), or a property path pattern, or a subtree rooted in And, Filter, Union, Optional, Graph whose respective patterns are also subtrees, or, recursively, the subtree of a query Q.

Figure 4 exemplifies the translation of the following Wikidata query from the logs into the AST.

```
SELECT ?item ?itemLabel WHERE { ?item wdt:P31 wd:Q146.
  SERVICE wikibase:label
   { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en"}}
```

Notice that in Figure 4, we ignore the actual instantiation of the variable ?item by naming it *var* and we also ignore the variables that are used in the target list of the Select clause for the same reason. We also removed the namespace prefixes. We now measure query similarity by using a tree edit distance (TED) measure. Each node in the above example is given a weight contributing to the TED. In particular, in the above AST for the example query, all nodes have equal weight (= 1 to not introduce bias). Adjustments of the weights are of course possible targeting a specific user requirement. The above example AST tree has thus a total weight of 12, this being the sum of the weight of its nodes. Within the logs, we found a query almost identical to the above example that only changed the object wd:Q146 to wd:Q11538. This renaming a single node in a triple would result in a similarity of 11/12 = 0.92.

Renaming two nodes in a triple would result in a similarity of 0.83. Adding a single triple anywhere, for example, outside the Service clause, would result in a total weight of 16. Such a change amounting to a total weight of 4 would result in a similarity of 12/16 = 0.75. Conversely, this can be seen as as a removal by starting with a tree exhibiting the additional triple and deleting it.

As for the implementation of the TED algorithm, we opted for an available implementation of the APTED algorithm [23, 24], because the implementation was easy to embed into our software, offered the option to use different weights for nodes, and yielded good performance results in our tests.

To calculate the similarity between an initial query Q and a new query Q_2, we take the size of the largest AST and divide it with the calculated TED. We invert this value by subtracting it from 1.0 in order to measure similarity instead of dissimilarity. When collecting matches in a collection of queries (within the same log file), we used a minimum threshold of 0.75. A threshold of 0.5 would mean that two queries are equally similar and dissimilar, so the

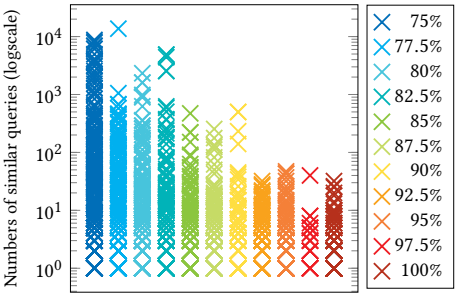


Figure 5: Number of queries that exhibit a similarity degree between 75% and 100%.

step to 0.75 is right in the middle between this and being classified as identical (which corresponds to similarity equal to 1). Based on manually inspecting samples, the threshold of 0.75 already seemed to be rather lenient, and our method based on TED yielded more predictable results for us than another method we experimented with based on SED. For example, a typical Wikidata entity has a string length of about 36. Even if all triples are changing, this can result in almost a 0.9 string similarity for the entire query, because the prefixes of entities are always the same.

In order to measure the structural similarity of Wikidata queries, we focused on the OrganicOK query logs by considering their Unique version only.[14] Since we needed seed queries for which we want to find similar queries, we considered the well known set of online Wikidata examples from [32]. We extracted[15] 356 out of the available 412 queries, which were the ones that could be parsed by Jena. It turns out that matches for most online example queries are found in the Wikidata logs analysed in our paper. Only 28 queries could not be matched, while the rest had at least one match above the threshold of 0.75.

Figure 5 shows in logscale the number of similar queries that we found in the OrganicOK Unique query logs in the order of ascending similarity. We can observe some outliers in several buckets in the scatterplot, showing that these queries have higher number of similar counterparts with respect to their pairs in the same bucket. The entire log could be scanned for the similarity search quite efficiently in less than 20 mins (without any optimization.)

9 RELATED WORK

Several empirical studies of SPARQL query logs have been conducted in the literature investigating statistical features, such as such as occurrences of triple patterns and types of queries [3, 15, 23, 29] along with recent studies on more complex features of the queries, such as structural features and complexity of the queries [10, 27]. However, the analysis of Wikidata queries has been limited in the past due to their unavailability. Bonifati et al. [10] only focused on the online 308 Wikidata queries in their large corpus and, more recently, Malyshev et al. [23] and Bielefeldt et al. [8] were the first to analyze a massive collection of Wikidata queries. Malyshev et al. first introduced the Wikidata SPARQL service WDQS and analyzed basic characteristics of Wikidata queries related to their usage in this service spanning from SPARQL feature prevalence and correlation to annotations and language distributions. They also isolated the robotic and organic queries, on whose classification we rely in this paper (modulo a small correction highlighted in Section 2). This classification considered as organic the queries issued by a browser and robotic the remaining queries. They further corrected the first number by identifying high-volume traffic from a single source (more than 2,000 entries), in which case the queries would be considered as machine-generated. They also identified the C2RPQs fragment, i.e. the largest fragment encountered so far of conjunctive 2-way regular path queries on which we focus in this paper for an in-depth analysis. To the best of our knowledge, the latter is the largest fragment of recursive graph queries available to our community. In this paper, we address the first large-scale structural analysis of this gigantic query collection and articulate it as follows. We provide a structural classification of real-world property paths, on the first large set of property paths relevant for Wikidata, and 57x larger than the set considered mainly for DBpedia in Bonifati et al. [9]. We also investigate the shape of C2RPQs, which could not be possible with the classification for only conjunctive queries (CQs) in Bonifati et al. [9]. The occurrences of property paths in the latter corpus is negligible with respect to the size of the C2RPQ fragment considered in our work (smaller by two orders of magnitude).

10 CONCLUSIONS AND LESSONS LEARNED

We have presented an in-depth analysis of the recently released Wikidata query logs and highlighted the presence of their most prominent query fragment, i.e. C2RPQs. This fragment corresponds to highly complex recursive queries with joins and property paths. Apart from simple counting measures on this fragment, we have focused on tailoring property path analysis and shape analysis to these queries, whereas previous work merely looked at conjunctive queries (CQs) in DBpedia logs [9]. Even though we agree with Bielefeldt et al. [7] on the difficulty of obtaining stable observations from these query logs due to massive presence of robotic traffic, we discovered several similarities, like low hypertreewidth and structure of property paths seem to be relatively consistent between the present study and previous work.

But what about the distinguishing features of these logs compared to previous logs and the findings that were possible on this newly discovered fragment, i.e. C2RPQs? Entirely new observations were made across the distinction in terms of Valid and Unique logs, further segmented into RoboticOK and OrganicOK, the addition of timeout logs never analysed before, which led us to add interesting dimensions to the analysis.

We now revisit the research questions posed at the end of Section 1 and we succinctly report the main findings of our analysis for those questions. About the distributions of sizes, we found out that the RoboticOK queries are less skewed in terms of sizes than the OrganicOK queries and this also applies to OrganicTO and RoboticTO queries, that are also inherently more complex in terms of sizes than the RoboticOK plus OrganicOK queries.

[14] The same analysis can be applied to the other logs but is probably less interesting for the present discussion.

[15] On August 23, 2018. Jena version same as in Section 2.

Next, property paths occur in these logs 57x more than in previously analyzed logs, which made us focus on the most representative query fragment C2RPQs. We noticed that a bigger variety of property path classes again occurs with Organic queries compared to Robotic queries. Still, we see that almost all property paths match paths of *any possible length* once a certain minimum length is exceeded. (Typical such expressions are ab^* or ab^*c.) Expressions that do not satisfy this property, like $(ab)^*$, appear as well, but are less prominent. In fact, the most occurring types of property paths found were similar to those found in an earlier study [9], which may allow us to identify subclasses of expressions that are important in practice. Such a discovery might spur interesting query processing and query optimization questions around C2RPQs, which were not addressed for the much simpler fragments of CQs. For instance, landmarking indexes have been introduced for one of the prominent classes of Robotic queries (A^*) in Valstar et al. [30], but also the other prominent classes need attention when designing indexes for C2RPQs.

In our analysis, we also addressed the question on the prominence of CQs and C2RPQs in these logs compared to other logs, thus bringing to the surface the most occurring recursive fragment of C2RPQs enriched with And, Optional, Filter, Bind, Service and Values (C2RPQ+). We ran a shape classification with and without constants for several fragments ranging between the class of CQs and C2RPQ+ by considering/excluding constants. Here, we see that *star shapes* and *tree-like queries* are very common. The shape analysis with or without constants also led us to identify shifts in the shape classes due to removal of constants that are worth looking at. For instance, if we remove constants from the graph shape of queries in the logs, many (25% organic and 36% robotic) disintegrate to a single node. Constants have been disregarded in the study of C2RPQs, where indexing techniques have mainly considered the labeled paths as key index terms. The combination of indexing techniques looking at constants and labeled paths could thus be a direction to pursue in future studies on indexing structures and index maintenance for these queries [8]. The timeout queries are also interesting because they are on average larger in size and more cyclic than the valid queries.

Concerning treewidth and hypertreewidth, the logs strongly confirm a hypothesis that is often stated in theoretical research: the cyclic queries in practical applications are only *mildly cyclic*. This means that database queries typically do not have large k-cliques encoded in their shape, but remain tree-like. This observation is in line with the data from a previous study [9].

Next, a novel Wikidata-specific query similarity search allows to efficiently navigate the query logs starting from an initial query that the user has at her disposal. This search improves the usability of the Wikidata query logs for both recursive and non-recursive queries and is valuable for identifying subsets of similar queries on which further assessment is possible.

After pre-processing, i.e., computing the Valid and Unique data sets, our entire analysis (except query similarity search, whose performance has been separately measured) takes 12 hours on a 24 core machine with a 2.6 GHz CPU and 128 GB RAM.

We believe that isolating complex query fragments and studying suitable sophisticated metrics is valuable for the community and may lead to further studies and assessment of these logs. We believe that this work can serve as a basis for researchers to find further interesting fragments of queries to study; we report what we see in the logs. However, one should always keep in mind that we are looking at *specific query logs*. It cannot be concluded from this study that a given fragment, operator, or type of query is *not* interesting to study.

REFERENCES

[1] Marcelo Arenas, Sebastián Conca, and Jorge Pérez. 2012. Counting beyond a Yottabyte, or how SPARQL 1.1 property paths will prevent adoption of the standard. In *World Wide Web Conference (WWW)*. 629–638.

[2] Mario Arias, Javier D. Fernández, Miguel A. Martínez-Prieto, and Pablo de la Fuente. 2011. An Empirical Study of Real-World SPARQL Queries. *CoRR* abs/1103.5043 (2011).

[3] Guillaume Bagan, Angela Bonifati, and Benoît Groz. 2013. A trichotomy for regular simple path queries on graphs. In *Principles of Database Systems (PODS)*. 261–272.

[4] Pablo Barceló, Markus Kröll, Reinhard Pichler, and Sebastian Skritek. 2018. Efficient Evaluation and Static Analysis for Well-Designed Pattern Trees with Projection. *ACM Trans. Database Syst.* 43, 2 (2018), 8:1–8:44.

[5] Pablo Barceló, Reinhard Pichler, and Sebastian Skritek. 2015. Efficient Evaluation and Approximation of Well-designed Pattern Trees. In *Principles of Database Systems (PODS)*. 131–144. https://doi.org/10.1145/2745754.2745767

[6] Adrian Bielefeldt, Julius Gonsior, Larry Gonzalez, Markus Krötzsch, and Stanislav Malyshev. 2018. Wikidata SPARQL Logs. https://iccl.inf.tu-dresden.de/web/Wikidata_SPARQL_Logs/en.

[7] Adrian Bielefeldt, Julius Gonsior, and Markus Krötzsch. 2018. Practical Linked Data Access via SPARQL: The Case of Wikidata. In *Linked Data on the Web (LDOW)*. http://events.linkeddata.org/ldow2018/papers/LDOW2018_paper_4.pdf.

[8] Angela Bonifati, George H. L. Fletcher, Hannes Voigts, and Nikolay Yakovets. 2018. *Querying Graphs*. Morgan & Claypool Publishers.

[9] Angela Bonifati, Wim Martens, and Thomas Timm. 2017. An Analytical Study of Large SPARQL Query Logs. *PVLDB* 11, 2 (2017), 149–161.

[10] Chandra Chekuri and Anand Rajaraman. 1997. Conjunctive Query Containment Revisited. In *International Conference on Database Theory (ICDT)*. 56–70.

[11] DARQL. 2019. https://github.com/PoDMR/darql/.

[12] Detkdecomp. 2018. https://github.com/daajoe/detkdecomp. Visited on September 10th, 2018.

[13] Georg Gottlob, Nicola Leone, and Francesco Scarcello. 2001. The complexity of acyclic conjunctive queries. *J. ACM* 48, 3 (2001), 431–498.

[14] Xingwang Han, Zhiyong Feng, Xiaowang Zhang, Xin Wang, Guozheng Rao, and Shuo Jiang. 2016. On the statistical analysis of practical SPARQL queries. In *International Workshop on Web and Databases (WebDB)*. 2.

[15] Stever Harris and Andy Seaborne. 2013. *SPARQL 1.1 query language*. Technical Report. World Wide Web Consortium (W3C). https://www.w3.org/TR/2013/REC-sparql11-query-20130321.

[16] Oren Kalinsky, Yoav Etsion, and Benny Kimelfeld. 2017. Flexible Caching in Trie Joins. In *International Conference on Extending Database Technology (EDBT)*. 282–293.

[17] Phokion G. Kolaitis and Moshe Y. Vardi. 1998. Conjunctive-Query Containment and Constraint Satisfaction. In *Symposium on Principles of Database Systems (PODS)*. 205–213.

[18] Andrés Letelier, Jorge Pérez, Reinhard Pichler, and Sebastian Skritek. 2013. Static analysis and optimization of semantic web queries. *ACM Trans. Database Syst.* 38, 4 (2013), 25:1–25:45.

[19] Katja Losemann and Wim Martens. 2013. The complexity of regular expressions and property paths in SPARQL. *ACM Trans. Database Syst.* 38, 4 (2013), 24:1–24:39.

[20] Stanislav Malyshev, Markus Krötzsch, Larry González, Julius Gonsior, and Adrian Bielefeldt. 2018. Getting the Most out of Wikidata: Semantic Technology Usage in Wikipedia's Knowledge Graph. In *International Semantic Web Conference (ISWC)*. 376–394.

[21] Wim Martens and Tina Trautner. 2018. Evaluation and Enumeration Problems for Regular Path Queries. In *International Conference on Database Theory (ICDT)*. 19:1–19:21.

[22] K. Möller, M. Hausenblas, R. Cyganiak, S. Handschuh, and G. Grimnes. 2010. Learning from linked open data usage: Patterns & metrics. In *Web Science Conference (WSC)*.

[23] Mateusz Pawlik and Nikolaus Augsten. 2015. Efficient Computation of the Tree Edit Distance. *ACM Trans. Database Syst.* 40, 1, Article 3 (March 2015), 40 pages.

[24] Mateusz Pawlik and Nikolaus Augsten. 2015. Tree edit distance: Robust and memory-efficient. *Information Systems* 56 (08 2015). https://doi.org/10.1016/j.is.2015.08.004

[25] Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. 2009. Semantics and complexity of SPARQL. *ACM Trans. Database Syst.* 34, 3 (2009), 16:1–16:45. https://doi.org/10.1145/1567274.1567278
[26] François Picalausa and Stijn Vansummeren. 2011. What are real SPARQL queries like?. In *International Workshop on Semantic Web Information Management (SWIM)*. 1–7.
[27] Alessandro Piscopo. 2018. Wikidata: A New Paradigm of Human-Bot Collaboration? *arXiv preprint arXiv:1810.00931* (2018).
[28] Muhammad Saleem, Intizar Ali, Aidan Hogan, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo. 2015. LSQ: The Linked SPARQL Queries Dataset. In *International Semantic Web Conference (ISWC)*. 261–269.
[29] Sunburst diagram. 2019. Sunburst diagram. https://podmr.github.io/darql/property-sunburst/.
[30] Lucien D. J. Valstar, George H. L. Fletcher, and Yuichi Yoshida. 2017. Landmark Indexing for Evaluation of Label-Constrained Reachability Queries. In *International Conference on Management of Data (SIGMOD)*. 345–358.
[31] Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.
[32] Wikidata. 2018. https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service/queries/examples.
[33] Mihalis Yannakakis. 1981. Algorithms for Acyclic Database Schemes. In *International Conference on Very Large Data Bases (VLDB)*. 82–94.

What happened? The Spread of Fake News Publisher Content During the 2016 U.S. Presidential Election

Ceren Budak
University of Michigan
Ann Arbor, Michigan
cbudak@umich.edu

ABSTRACT

The spread of content produced by fake news publishers was one of the most discussed characteristics of the 2016 U.S. Presidential Election. Yet, little is known about the prevalence and focus of such content, how its prevalence changed over time, and how this prevalence related to important election dynamics. In this paper, we address these questions using tweets that mention the two presidential candidates sampled at the daily level, the news content mentioned in such tweets, and open-ended responses from nationally representative telephone interviews. The results of our analysis highlight various important lessons for news consumers and journalists. We find that (i.) traditional news producers outperformed fake news producers in aggregate, (ii.) the prevalence of content produced by fake news publishers increased over the course of the campaign–particularly among tweets that mentioned Clinton, and (iii.) changes in such prevalence were closely following changes in net Clinton favorability. Turning to content, we (iv.) identify similarities and differences in agenda setting by fake and traditional news media and show that (v.) information individuals most commonly reported to having read, seen or heard about the candidates was more closely aligned with content produced by fake news outlets than traditional news outlets, in particular for information Republican voters retained about Clinton. We also model fake-ness of retained information as a function of demographics characteristics. Implications for platform owners, news consumers, and journalists are discussed.

CCS CONCEPTS

• **Social and professional topics** → **Political speech**; • **Information systems** → *Social networking sites.*

KEYWORDS

news media; fake news; topic modeling; multi-level regression

ACM Reference Format:
Ceren Budak. 2019. What happened? The Spread of Fake News Publisher Content During the 2016 U.S. Presidential Election. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313721

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313721

1 INTRODUCTION

News production and consumption have always played an important role in our democracy–with noted consequences for political participation [23], voting [21], and charitable giving [13]. This role has been repeatedly reshaped due to the changes in the media technology–from the introduction of cheap newsprint [41] to the growth of online news production and consumption [24]. Each change has led to concerns about the quality of news and its consequences for our democracy. Most recently, the concerns have centered around the shift towards news production and consumption through social media platforms [3, 24]–with a specific focus on the increasing influence of fake news. While the spread of fake news content on social media platforms is not new, the 2016 U.S. Presidential election has highlighted its extent, raised fears that this new paradigm of news consumption is misinforming voters and corroding our democracy [22, 53, 64]. These concerns have garnered significant attention in both media and policy circles, with some journalists even going as far as claiming that results of the 2016 election were a consequence of the spread of fake news [22, 53].

There are a number of factors that likely contributed to the spread of fake news publisher content, such as increasing use of social media platforms that lack a centralized control mechanism for news consumption [29], an overall decreasing trust in mainstream media [25, 62, 69], profitability of fake news production [34], and recognition of otherwise fringe ideas and news by high profile politicians [6, 47]. Understanding to what extent the aforementioned reasons contributed to the spread of fake news content will certainly take a long time. Here, instead of the "why", we focus on the "what"; and examine to what degree individuals shared news from fake news producers as opposed to traditional news outlets during the 2016 U.S. presidential election. Using a rich dataset that includes daily random sample of tweets mentioning the two presidential candidates, the content of external urls shared in those tweets, and nationally representative interviews conducted by Gallup at the daily level, we study the overall prevalence and focus of fake news producers, how the prevalence changed over time, and examine how the production and consumption of such news relate to election campaign dynamics.

By analyzing the overall prevalence of content produced by fake and traditional news outlets and shared on Twitter, we show that:

(1) An overwhelming majority of news content shared on Twitter was produced by *traditional* news outlets.
(2) The prevalence of fake news increased over time.
(3) The increase in prevalence was more drastic in tweets about Hillary Clinton and followed significant campaign moments.
(4) The changes in fake news prevalence followed changes in net Clinton favorability, highlighting a strong correlation

between candidate favorability and fake news production and consumption.

Next, by analyzing the content of the articles, we provide the following contributions:

(1) We identify the similarities and differences between content produced by fake and traditional news outlets using semi-supervised machine learning methods.
(2) We determine that the retained information about Clinton (what likely voters have reported to having *read, seen or heard* about her) was more aligned with fake news agenda compared to traditional news agenda.
(3) We perform statistical modeling of fake-ness of retained information and show that Republican, younger and male interviewees had a higher tendency to report content more closely aligned with fake news publishers. Race and location had no significant effect.

2 RELATED WORK

Social Media and the News. As more news consumption moves to social media platforms [29]—with social media news consumption even providing predictive signals of consumption through all channels [17]—there has been an increasing interest in studying the trustworthiness of content shared through this channel [57, 73].

Recent increase in the volume of misinformation, conspiracy theories, and disinformation and the likely role social media played in this uptick have sparked growing interest in related research [14, 33, 46, 52, 56, 60, 68]. While earlier studies focus on limiting the spread of misinformation as a theoretical optimization problem [14], more recent studies propose data mining and machine learning techniques to identify and limit spread online [56, 60]. Some studies characterize fake news, misinformation, conspiracy theories, and disinformation online [33, 46, 52, 68, 71], while others focus on the types of agents–such as bots—that disseminate such information [1, 7, 20]. Most relatedly, [71] examines rumors that spread on Twitter between 2006 and 2017 and finds that false news spread faster and deeper than true news. Unlike these studies, we focus specifically on the spread of news during 2016 U.S. presidential election.

News Dynamics During 2016 Election. 2016 election has generated increased interest in the study of news dynamics with a particular focus on fake news (e.g. [3, 7, 10, 31, 35, 42, 44, 47, 58]).

Most relatedly, two studies concurrently published with our work investigate the spread of fake news through Twitter [10, 31]. [31] studies the exposure of registered voters on Twitter to fake news content and shows high concentration—only 1% of individuals accounted for 80% of fake news source exposures. They also study engagement with fake news sources as a function of demographics characteristics. [10] studies networks of information flow on Twitter leading up to the 2016 U.S. elections and show that top influencers spreading center and left leaning news influences Clinton supporters. In contrast, the activity of Trump supporters influences the behavior of top fake spreaders.

Other relevant studies include [47] which studies manipulation and disinformation online, provides a survey of various important case studies, and characterizes potential outcomes of such spread. While the main focus is broader than the 2016 election, the authors emphasize the relationship between the mis- and disinformation ecosystem and election campaign dynamics. [3] studies the effect of fake news during the 2016 election. The authors show that social media was an important but not dominant source of election news, and that fake news stories about Hillary Clinton received more attention. They further estimate the fake news exposure of an average American. [7], aiming to determine the role bots played during the 2016 election, examines tweets shared during the campaign season and finds that about one-fifth of the entire conversation was produced by bots. [64]—focusing on the most commonly shared urls on Facebook—shows that (i.) hyper-partisan news articles attracted more attention than mainstream news articles on Facebook during the 2016 election, (ii.) hyper-partisan and fake news sites had a strong ideological bias against Hillary Clinton. Also relatedly, [42] studies the peer-production of content related to the two presidential candidates on Wikipedia; [58] provides a new method to estimate media bias; and [35] presents a new dataset for examining misinformation on social media.

Our study complements these investigations by bringing together data sources from social media, news media, and Gallup interviews. Unlike related work, we study the focus and the changes in dynamics of fake news prevalence and relate these to important election dynamics. Furthermore, we present the first study that analyzes the relationship between the favorability of the candidates and the production and consumption of news—with a particular focus on content produced by fake news.

3 DEFINING FAKE AND TRADITIONAL NEWS

Identifying the credibility and quality of news content is a challenging task. A recent publication by leading scholars studying misinformation advocates for classification at the outlet level [45]. The authors advocate for "... focusing on the original sources—the publishers—rather than individual stories, because we view the defining element of fake news to be the intent and processes of the publisher. A focus on publishers also allows us to avoid the morass of trying to evaluate the accuracy of every single news story". This paper follows this guideline and defines fake and traditional news at the producer (web domain) level.

Fake News Publishers. In this study, we rely on the fake news classification provided by [4]. The set of fake news producers published by [4] are gathered from 5 distinct lists: (i) a research project by Grinberg et al. [30] (490 publishers); (ii) PolitiFact's article titled "PolitiFact's guide to fake news websites and what they peddle" [27] (325 publishers); (iii) the list provided in the three articles published by BuzzFeed on fake news [63, 65, 66] (223 sites); (iv) a research project by Guess et al. [32] (92 publishers); and (v) FactCheck's article titled "Websites that post fake and satirical stories" [59]. [27, 59, 63, 65, 66] are published by organizations that apply journalistic standards in evaluating the validity of news articles. [30, 32] use a combination of these resources as well as lists published by [12, 75] with some further verification from Snopes.com.

There are a number of of fake news producer lists available online (e.g. [27, 30, 32, 59, 63, 65, 66] as described briefly above). We rely on the classification provided by [4] for two main reasons. First, this list provides the widest coverage. Second, the authors provide the most detailed robustness checks of their list compared to other

efforts (e.g. [75]). A key concern here is the potential selection bias in the list of fake news sites gathered. The authors perform two robustness checks. First, they focus on sites that are identified as fake news sites by at least two or three lists instead of one. Second, they consider lists of sites assembled from any four out of the five original lists. They show that the general trends of fake news coverage is consistent across these definitions.

Traditional news. We use Alexa [2] to identify traditional news domains. Alexa provides commercial web traffic analytics using a large panel of Internet users and their web-browsing patterns. This site also provides a rich ontology of the web [1]. Using the Alexa ontology, we identify the set of websites listed under the news category. There are in total 5,497 domains listed under this category including national, regional as well as international news sources. We remove the set of web domains listed by [4] and classify the rest of the sites as traditional news sources. Note that the news categorization provided by Alexa has also been used in past work to identify news outlets (e.g. [3, 58]).

We note that future work is needed to measure and improve the accuracy and coverage of the aforementioned lists of traditional and fake news outlets. Furthermore, such lists need to be maintained in a dynamic fashion to account for changes in the news media space.

4 DATA

Social and News Media. The social media dataset used in this study is described in detail in [54]. Data are collected through Sysomos MAP[2]. This service allows gathering of tweets sampled uniformly at random on a daily level. The dataset covers the time period between May 23, 2014 and January 1, 2017. For each day within that time period, we use (i.) 5,000 tweets that were uniformly randomly sampled from all tweets that included the keyword *Trump*, and (ii.) 5,000 tweets that are similarly sampled from all tweets that mention *Clinton* [54]. The resulting dataset includes approximately 4.8 million tweets about Donald Trump and, similarly, approximately 4.8 million tweets about Hillary Clinton. Given that our goal is to analyze news content, we then remove tweets that do not include an external url. The dataset after filtering tweets without urls includes: (i.) 3,072,251 shares of 1,529,251 unique shortened urls by 600,915 unique Twitter users, and (ii.) 4,689,087 shares of 2,025,203 unique shortened urls by 602,613 unique Twitter users for Trump and Clinton.

Given the set of tweets that include a shortened url (of the form: http://t.co/xxxxxxxxxx), we next use automated scripts to identify the url redirect and extract the content of the webpage. We remove urls internal to Twitter since such urls correspond to quoted retweets as opposed to external news content. The resulting dataset includes approximately 1.7 million unique urls. For each share (i.e. tweet including a valid url), we record the following data: i.) the shortened url, ii.) the original url, iii.) domain name (e.g. wsj.com), iv.) title of the document, v.) text body of the document, vi.) the date of the tweet, vii.) Twitter id of the user sharing the url, and (viii.) a binary categorization that indicates whether this tweet is in our Clinton or Trump sample. When we limit our data to domains that are included in the fake and traditional news domains lists (detailed below), there are approximately 270K unique articles that were shared in approximately 564K tweets about Trump (in tweets that mention Trump), and approximately 350K unique articles about Clinton that were shared in 1.1M Clinton tweets.

Gallup Data. In order to characterize the election dynamics as perceived by the U.S. population, we also use responses to two open-ended questions fielded as part of the Gallup Daily survey. Respondents were asked, "What have you recently read, seen, or heard about [CANDIDATE]?" for both Donald Trump and Hillary Clinton (asked in random order). To the best of their ability, telephone interviewers recorded individuals' responses verbatim. Gallup asked these questions to a random sample of 500 adults (ages 18 and higher, living in all 50 U.S. states and the District of Columbia, approximately 60% cellphone and 40% landline) each day between July 11 and November 7, 2016. The margin of sampling error at the weekly level is ±2 percentage points at the 95% confidence level. For each interview response, the dataset also includes the gender, age, location, and ideological leaning of the respondent. We refer the reader to [9] for additional details about the survey.

5 OVERALL PREVALENCE OF FAKE NEWS PRODUCER CONTENT

A number of investigative efforts suggest wide spread of fake and hyper-partisan news consumption during the 2016 election. For instance, a *Buzzfeed News* study [64] showed that *top* hyper-partisan news articles attracted more attention (measured in terms of total number of shares, reactions, and comments) than *top* mainstream news articles on Facebook by examining top Facebook stories of the election season. Here, we extend this knowledge in three important ways: First, we examine the aggregate popularity (measured in terms of number of shares) of all content shared by traditional and fake news publishers, as opposed to focusing on the top few most commonly shared articles. Second, given that our dataset is balanced over time due to random tweet sampling at the daily level, our results are not dominated by few articles that were in the public eye for a short amount of time. Third, in this study we use data from Twitter, which has important differences in dynamics of information sharing compared to Facebook[3]. As a result, we extend the understanding of the prevalence of fake news in 2016 Presidential election cycle to a new social media platform.

We find that traditional news articles were shared by 2.8 users on average while this number was 2.88 for articles published by fake news outlets–the small difference observed is not statistically significant ($p \approx 0.08$). Similarly, a traditional news article was shared on 4.7 days on average while this number is 5.5 for articles published by fake news producers ($p < 0.001$ for the difference in means). The distributions are highly skewed ($\approx$ 14 standard deviation for the number of shares and $\approx$ 33 for the number of days). Median of the distributions are the same (1 for both number of shares and number of days). These measures point to comparable popularity *per article* produced by fake and traditional news publishers[4]. Yet, because the

[1] Since our data collection, Alexa has stopped sharing the ontology.
[2] http://sysomos.com/products/overview/sysomos-map

[3] Since the writing of this study, two other studies ([10, 31]) also examined Twitter content, albeit through different data collection mechanisms. We find similar qualitative findings in fake news prevalence despite different data collection choices.
[4] Note. however, that our data only include articles with *at least one* share.

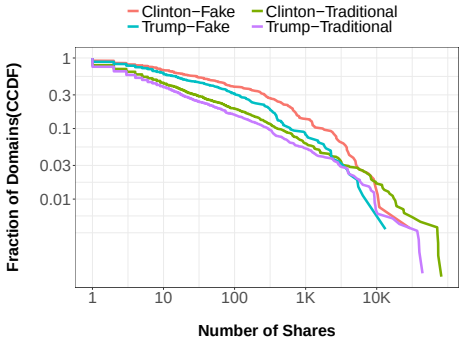


Figure 1: Popularity of Fake and Traditional News Domains for Clinton and Trump Related Tweets.

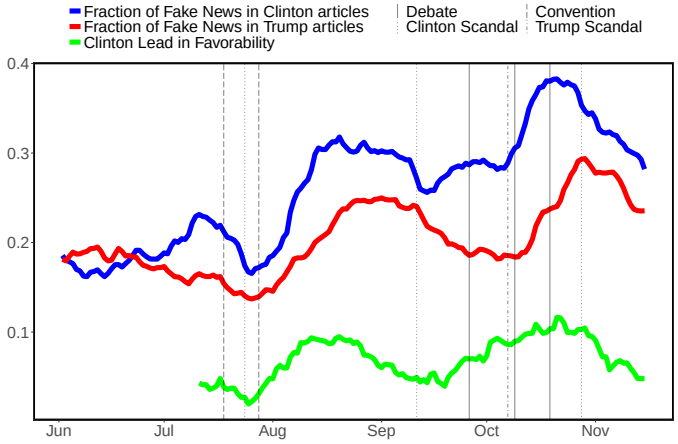


Figure 2: Fraction of article shares that are from fake news producers and Clinton net favorability over time. The error bars give standard errors.

volume of traditional news production was higher, collectively traditional news accounted for significantly more shares–the number of users sharing traditional news content outnumbered those that share content produced by fake news outlets roughly one to four.

We further investigate how this popularity is concentrated across different publishers and present the results of our findings in Figure 1. The x-axis denotes the popularity of a given domain (separately among tweets mentioning Trump and Clinton) and the y-axis denotes the fraction of domains of the corresponding type (e.g. a fake news domain in Clinton tweets is given by the red curve) with at least that much popularity. Here popularity is defined as the number of tweets that included a url from the corresponding domain. This figure shows that there are a larger fraction of fake news publishers that were able to garner *reasonable* attention compared to traditional news sites. For instance, 68% and 61% of fake news publishers had popularity strictly less than 100 for Trump and Clinton respectively. These numbers are 84% and 81% for traditional news publishers. In contrast, traditional news coverage dominance was mostly driven by a small number of big outlets that drove most of the traffic. For instance, there are no fake news publishers that single-handedly garnered more than 30K shares in our dataset while there are a number of traditional news sites (e.g. New York Times, Fox News). This highlights important differences in production and consumption of fake and traditional news publishers—popularity was more concentrated in the head for traditional news compared to fake news.

6 CHANGES IN FAKE NEWS PRODUCTION AND CONSUMPTION

Here we investigate the relationship between fake news production/consumption and the favorability of the two candidates. Our goal is to determine whether the prevalence of content produced by fake news outlets was correlated with campaign dynamics–following a pattern similar to the relative favorability of candidates and changes to this measure. The favorability of Hillary Clinton on a given day i, denoted by $Fav_{Clinton,i}$, is computed using the Gallup interview dataset as:

$$Fav_{Clinton,i} = \frac{\sum_{r \in R_i} I_{r,Clinton} * w_r}{\sum_{r \in R_i} w_r} \quad (1)$$

where R_i is the set of respondents on day i, $I_{r,Clinton}$ is an indicator that is equal to 1 if respondent r is favorable towards Hillary Clinton, and w_r is the weight used for post-stratification to correctly estimate sentiment across the United States.[5] A similar method is used to identify Donald Trump favorability ($Fav_{Trump,i}$) at the daily level. Next, Clinton lead in favorability on day i is computed as $Fav_{\Delta Clinton,i} = Fav_{Clinton,i} - Fav_{Trump,i}$.

We next compute the fake news ratio on a daily level. First, we define 4 sets of tweets for each day i: $T_{Clinton,i}$, $T_{Trump,i}$, $F_{Clinton,i}$, and $F_{Trump,i}$ that correspond to the set of tweets that include (i) link to a traditional publisher and the word "Clinton", (ii) link to a traditional publisher and "Trump", (iii) link to a fake news publisher and "Clinton", and (iv) link to a fake news publisher and "Trump". Next, fake news content fraction for Clinton and Trump tweets are measured as $f_{Clinton,i} = \frac{|F_{Clinton,i}|}{|F_{Clinton,i}|+|T_{Clinton,i}|}$ and $f_{Trump,i} = \frac{|F_{Trump,i}|}{|F_{Trump,i}|+|T_{Trump,i}|}$ respectively.

The prevalence of fake news content ($f_{Clinton,i}$ and $f_{Trump,i}$) and net Clinton favorability ($Fav_{\Delta Clinton,i}$) over time are provided in Figure 2. The blue curve in Figure 2 denotes the prominence of articles produced by fake news producers in tweets about Clinton. Since $f_{Clinton,i}$ can fluctuate from day-to-day, we smooth out the curve by using and plotting a moving average of 15 days centered at i. The red curve gives a similar measure for the fraction of fake news content in Trump tweets ($f_{Trump,i}$). The green curve denotes the net Clinton favorability over time ($Fav_{\Delta Clinton,i}$). This time series is smoothed out to give the 15-day running average for consistency.

Figure 2 provides a number of important insights. First, tweets about Clinton had a higher fraction of shares from fake news publishers compared to tweets about Trump. Past work shows that ideological bias is commonly exercised by criticizing the other side as opposed to championing for the preferred party [15]. Therefore,

[5]The computation of the weight are described here: https://www.gallup.com/185462/gallup-daily-work.aspx.

our finding suggests that fake news domains were commonly publishing negative content about Hillary Clinton. This finding is also in agreement with [64] that claims that fake news content commonly had an anti-Hillary Clinton bias. We further substantiated this conjecture by performing tone analysis of content published by fake and traditional news producers that are shared in Clinton tweets using Lexicoder [74].[6]

Figure 2 also shows an increasing popularity of fake news producers over time—more so for Clinton tweets but also for Trump tweets. While the average fraction of fake news content is about 20% before the conventions, it lies at $\approx 30\%$ after the nominations and reaches $\approx 40\%$ right before the election. For both time series, we observe two peaks. The first upward trending starts after the conventions and the second starts after the second presidential debate. Fake news content popularity is ultimately a function of both production and consumption. This trend can be due to an increase in production of fake news content for ideological reasons to drive public opinion or due to consumption that is a result of changes in public opinion, or a combination of the two.

Finally, and most intriguingly, net Clinton favorability (blue curve) and the prevalence of fake news content (red and green curves) are highly correlated (pearson correlation of 0.8). Intrigued by this correlation, we next perform cross-correlation analysis [70] to investigate whether changes in net Clinton favorability *lead* or *lag* changes in fake news content ratio. CCF or cross correlation function is defined as the set of correlations between the two time series s^1_{t+l} and s^2_t for lags $l = 0, \pm 1, \pm 2, \ldots$. A negative value for l represents a correlation between the s^1-series at a time *before* t and the s^2-series at time t. If, for example, the lag $l = -3$, then the cross correlation value would give the correlation between s^1_{t+3} and s^2_t.

The results of the cross-correlation analysis reveals that the highest correlation of 0.85 is observed for $l = 4$, indicating that fake news production/consumption changes *lag* changes in net Clinton favorability. We caution the reader that this analysis is only correlational. While the results suggest a strong relationship between production of fake news content and changes in candidate favorability, this should not be taken as evidence for the *effect* of fake news in the outcome of the 2016 election. If anything, our results suggest that fake news production and consumption responded to changes in the 2016 election campaigns as opposed to affecting them. We further caution that the causal mechanism is unknown. These observed patterns can be due to production (with outlets producing more damaging content for Hillary Clinton to combat improvements in her favorability numbers) or due to consumption, or rather a combination of the two. It is also entirely possible for both candidate favorability patterns and the fake news production and consumption to be responding to a third signal—with the fake news response being delayed longer. Future research is needed to identify the causal story—our analysis highlights the importance of carrying out such analysis.

[6]Lexicoder is a Java-based, multi-platform software for automated content analysis of text. Lexicoder provides two dictionaries—negative and positive words—designed to capture the sentiment of political texts. This method has been used by various studies to examine tone in news coverage [28, 67]. Using the dictionaries, we identify the set of positive and negative words in each article in our dataset and measure the net positivity as #positive - #negative words. Next, we measure the average net positivity of articles shared from fake and traditional news outlets separately. Our findings indeed show that articles published by fake news outlets and shared in tweets mentioning Clinton are on average more negative compared to articles published by traditional news outlets. The average tone is -1.3 and 2.2 respectively for fake and traditional news content.

Table 1: List of Topics, Most Weighted Keywords, and F1 Scores inferred through the methodology described in [11].

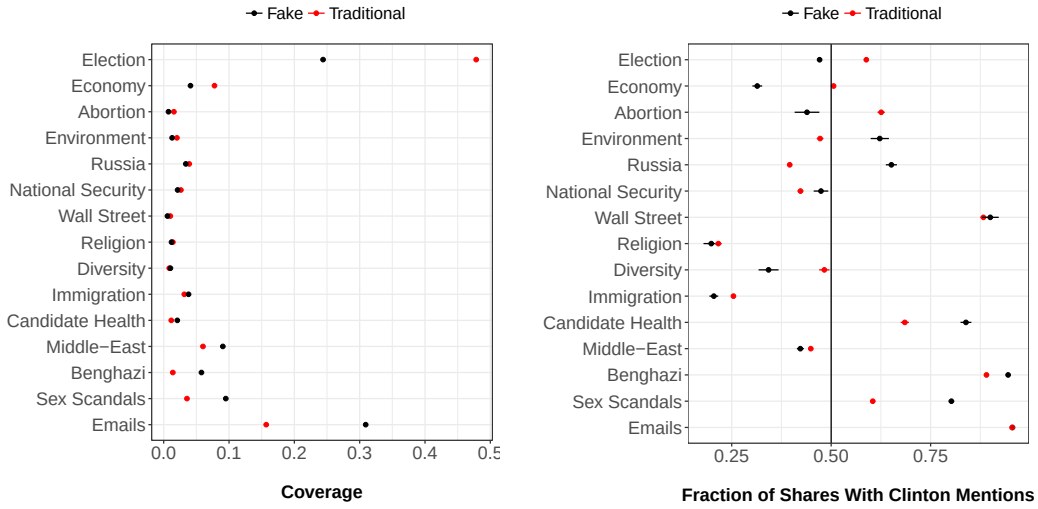
Topic	Important Words	F1
Abortion	woman abort life plan_parenthood issu punish femal	0.87
Benghazi	attack benghazi libya committe report secretari secur	0.75
Candidate Health	medic doctor releas report mental suffer letter pneumonia	0.75
Climate	climat coal environment industri land administr regul power	0.89
Wall Street	speech wall_street talk ask question issu transcript releas	0.82
Diversity	commun lgbt issu group equal woman discrimin anti marriag	0.78
Economy	trade job china deal compani manufactur econom	0.79
Election	sander berni primari voter percent poll voter cruz	0.77
Email	email depart investig inform server classifi comey secretari	0.84
Immigration	immigr border mexico wall illeg deport mexican build latino	0.85
Middle East	muslim islam israel isi terror terrorist attack unit syria obama	0.76
Religion	christian evangel church faith religi leader pastor religion pope	0.78
Russia	russia russian putin intellig hack offici govern vladimir_putin	0.76
National Security	iran china nuclear polici foreign deal unit world nato secur	0.78
Sex Scandals	woman accus alleg rape husband sexual claim sexual_assault	0.82

7 FAKE AND TRADITIONAL NEWS TOPICS

Here our goal is to determine the similarities and differences between fake and traditional news coverage leading up to the 2016 election through topic modeling.

The topics covered by traditional and fake news media and shared by Twitter users are inferred through the methodology proposed by [11]. This methodology involves four high level steps: 1) identifying candidate topics and topic-words through vanilla LDA [8] where the number of topics is chosen through topic coherence scores [51], 2) remapping topics and topic words to a smaller subset of coherent topics and associated seed words through expert manual labeling, 3) employing guided LDA [38] using these seed word lists, and 4) collapsing topics that share the same human readable category into a single topic and removing topics without coherence (using top words per topic). This approach produces 19 distinct topics. Each article is then assigned to a single topic according to the maximum probability of its topic distribution. These article-topic pairs are next evaluated manually through Amazon Mechanical Turk. The results reveal that 15 out of the 19 topics to have high accuracy ($F1 \geq 0.75$). Table 1 provides a summary of the 15 coherent topics identified by [11]. We refer the reader to [11] for more details about the methodology.

Having identified the topic of each article in our sample and filtered out articles that do not map to one of the 15 coherent topics, we next examine the similarities and differences between the agenda setting functions of fake and traditional news outlets. In Figure 3(a), we summarize the frequency of each topic separately for fake and traditional news outlets among the set of articles that map to one of the 15 topics. The topics are ordered as a function of the difference in coverage by fake and traditional news outlets. For instance, the top-most topic—*election*—accounts for approximately 48% of traditional news shares while it only accounts for 24% of fake news shares. The difference of 24% signifies to what degree this topic was more central to traditional news production and



(a) Topic Coverage

(b) Candidate Focus

Figure 3: Topical Analysis of Traditional and Fake News Media Shared on Twitter. The topics are ordered from most uniquely associated with traditional news (top) to most uniquely associated with fake news producers (bottom)

consumption compared to fake news production and consumption. In contrast, the bottom-most topic—*emails*—accounts for 31% of fake news shares while it accounts for 16% of traditional news shares. The difference of −15% indicates to what degree emails was a central theme of the fake news coverage and attention.

A number of important patterns emerge here. First, we see that election-related—or in other words horse-race—topic was most significantly overly represented in traditional news coverage. This finding is in line with established research that highlights the significance of the horse-race in news coverage [37] leading up to elections. Here, we show that fake news production diverges from traditional news in this context. Other topics more unique to traditional news outlets include policy focused topics such as the economy, women, and the environment. In comparison, topics that relate to Clinton scandals (emails, sex scandals—commonly about Bill Clinton—, the Benghazi scandal, candidate health issues) were most distinctly covered in articles published by fake news outlets and shared on Twitter. These findings highlight important differences in agenda setting and coverage by these two types of sources.

Despite these clear differences, important similarities emerge when one focuses on the coverage in absolute numbers. For instance, even though emails was a more important topic for fake news outlets compared to traditional news, this topic still made up a significant fraction (16%) of traditional news shares—making emails a more important topic for traditional news than the economy or other policy issues. We further emphasize that no Trump related scandal was common enough to be identified as a coherent topic. Overall, we find that traditional news coverage was extremely light on policy and the focus on a single scandal was disproportionately applied to Clinton. This finding is in line with related work [55] which suggests that traditional new outlets failed to inform the electorate properly leading up to the 2016 U.S. presidential election.

Next, we also investigate whether the coverage of topics were more particularly focused on Clinton or Trump and how that pattern differed across topics. To measure the Clinton focus of a given topic, we compute the fraction of times articles on that topic were shared in Clinton tweets (as opposed to Trump tweets)—separately for fake and traditional news outlets. The results are presented in Figure 3(b). The topics are ordered in the same order as Figure 3(a) for ease of viewing. The x-axis denotes the Clinton focus of the given topic. For instance, the first row shows that approximately 48% of shares of fake news sources on the election were mentioning Clinton, while approximately 54% of shares of traditional news sources on the same topic mentioned Clinton.

Topics with $x > 0.5$ for both fake and traditional news media are consistently Clinton scandals (emails, sex scandals—commonly about Bill Clinton—, Benghazi, candidate health problems, Clinton Wall Street speech scandal). Furthermore, when a topic can relate to scandals of either candidate (e.g. health problems and sex scandals[7]), the coverage of fake news producers focused significantly more on Clinton compared to Trump. The opposite is observed for policy coverage—fake news coverage had a stronger Trump emphasis when covering topics such as abortion and the economy. This suggests that the fake news outlets had a higher tendency to cover Trump policy positions, conditional on covering a given policy topic. These findings further support the claim that fake news production was consistently pro-Trump and anti-Clinton.

8 FAKE NEWS EXPOSURE

Did the information Gallup interview respondents retain about the two candidates better align with traditional or fake news coverage? To answer this question, we compute a measure of fake-ness—the

[7]Both Hillary Clinton and Trump had health issues leading up to the election. Similarly, sex scandals could be referring to Donald Trump or Bill Clinton.

degree to which the given word is unique to fake news coverage—for each unique word uttered by respondents when asked about what they have seen, heard, or read about Clinton (and separately for Trump). Next, we aggregate this information across all interviews to (i.) examine the fake-ness of popular phrases uttered in interviews by individuals with different party alignments, and (ii.) to build a mixed-effects model to characterize fake-ness of interview responses as a function of the attributes of the interviewee (e.g. party, gender, age).

Measuring word fake-ness. We adopt the log-odds-ratio technique with informative Dirichlet priors [48] to compute the fake-ness measure for each word for each candidate. This method has the advantage of de-emphasizing fluctuation of rare words by leveraging the prior information obtained from the background corpus, and has been shown to outperform other methods such as PMI and TF-IDF [40, 48, 50].

Using the log-odds-ratio with informed Dirichlet priors, the usage difference for word w among two corpora is computed as:

$$\delta_w^{(i-j)} = \log \frac{y_w^i + \alpha_w}{n^i + \alpha_0 - y_w^i - \alpha_w} - \log \frac{y_w^j + \alpha_w}{n^j + \alpha_0 - y_w^j - \alpha_w} \quad (2)$$

where n^i (resp. n^j) is the size of corpus i (resp. j), y_w^i (resp. y_w^j) is the word count of w in corpus i (resp. j), α_0 is the size of the background corpus, and α_w is the word count of w in the background corpus. The variance of this measure is:

$$\sigma^2(\delta_w^{(i-j)}) \approx \frac{1}{(y_w^i + \alpha_w)} + \frac{1}{(y_w^j + \alpha_w)} \quad (3)$$

and the Z-score is:

$$Z = \frac{\delta_w^{(i-j)}}{\sqrt{\sigma^2(\delta_w^{(i-j)})}} \quad (4)$$

We measure the fake-ness of a word w (through its Z-score) separately for Clinton and Trump. We do this since the usage of the same phrase when uttered for the two candidates has divergent implications.[8] Here we describe the process to compute fake-ness scores for each unique lemmatized word $w \in W_{int,Clinton}$ uttered when interviewees were asked what they heard, read or saw about Clinton: We first create three documents that correspond to corpus i, j, and background corpus described above. Corpus i (fake news corpus) includes all $w \in W_{int,Clinton}$ weighted by the number of times w is observed in the body of an article shared in a Clinton tweet in our dataset[9]. Similarly, corpus j (traditional news corpus) includes all $w \in W_{int,Clinton}$ weighted by the number of times w is observed in the body of an article produced by traditional news outlets and shared on Twitter with a Clinton mention. The background corpus is the concatenation of the two documents i and j. Next, we apply equations (2-4) to compute the fake-ness scores of each word when uttered for Clinton. We refer to this measure as $f(w_{Clinton,fake})$. The same method is applied next to compute the fake-ness of each word for Trump ($f(w_{Trump,fake})$).

This method assigns a value of 68.2 for the word *email* when used for Hillary Clinton—meaning that the word is more frequently observed in fake news articles and is therefore associated with positive *fake-ness*. The word *debat* ("debate" lemmatized), on the other hand is more frequently observed in traditional news articles and has a *fake-ness* measure of -48.5. Table 2 provides an ordered list of the top-20 fake words (words with the highest Z-score) and the list of the top-20 traditional news words (words with the lowest Z-score) for Trump and Clinton. This table highlights the effectiveness of this methodology. For instance, words relating to Clinton scandals (e.g. Huma, Wikileaks, corrupt) have high fake-ness measures (in line with [64]).

Table 2: Words that are most unique to fake vs. traditional articles for Trump and Clinton tweets.

News Outlet	Candidate	Most unique words
Fake	Trump	advertis, video, islam, continu, pleas, twitter, liber, obama, media, illeg, america, leav, establish, advanc, cruz, share, lie, support, christian, alien
Traditional	Trump	voter, campaign, republican, compani, casino, percent, photo, former, citi, iowa, advis, univers, hotel, tower, regist, month, busi, presidenti, chariti, build
Fake	Clinton	video, foundat, lawsuit, document, twitter, corrupt, huma, wikileak, lie, die, file, sex, court, death, report, email, content, forget, advertis, healthi
Traditional	Clinton	sander, percent, voter, campaign, democrat, photo, iowa, candid, republican, presidenti, primari, support, hampshir, senat, nomin, debat, poll, race, biden, gop

Fake-ness of popular interview phrases. Next, we compute the fake-ness of top-k words[10] uttered in Gallup surveys by a set of respondents R for each candidate c as follows:

$$f_{c,k,R}^G = \frac{\sum_{w \in G_{c,k,R}} f(w_{c,fake}) * n_{c,w}}{\sum_{w \in G_{c,k}} n_{c,w}} \quad (5)$$

where $G_{c,k}$ are the set of top-k words uttered by respondents R for candidate c in the Gallup interviews and $n_{c,w}$ is the number of times word w occurred in responses from R for candidate c. In other words, when computing the fake-ness of top-k words uttered by a group of respondents R, we weigh the estimate by how frequently each word is uttered by this group. For instance, email occurs 2.5 times more frequently than *debate* when considering all respondents. Therefore, "email" will account 2.5 times more heavily compared to "debate". Next, we inspect $f_{c,k,R}^G$ for both candidates separately for (i.) all respondents, (ii.) only Democrats, and (iii.) only Republicans, with k ranging between 1 and 100. As a result, we are able to compare and contrast the alignment between fake news coverage and information retained by different groups. The result of this analysis is summarized in Figure 4. The first panel in Figure 4 characterizes the fake-ness of responses from the three groups

[8] Consider for instance, the word "foundation". This word, when uttered for Clinton, is likely to refer to the Clinton foundation and the relevant scandals that were damaging for her [39]. The same phrase when uttered about Trump is likely to refer to the Trump foundation and the scandals that were damaging to him [26].

[9] If a word w is seen three times in a fake news article that is shared 10 times and twice in another article that is shared 20 times, w will be repeated $3 * 10 + 2 * 20 = 70$ times in corpus i.

[10] Stop words are removed for this analysis.

when asked about what they had read, seen or heard about Hillary Clinton recently. The solid, dotted, and dashed lines correspond to all respondents, Democrats, and Republicans respectively. The second panel presents the same information for the responses about Donald Trump.

A number of observations are worth noting from Figure 4. First, the top-100 words the interviewees uttered about Hillary Clinton consistently resemble fake news coverage more than traditional news coverage (with y-values greater than 0). Second, there is a clear divide between the Republican and Democrat interviewees. Popular Republican leaning interviewee response words are consistently in line with fake news coverage while the popular words used by Democrat leaning interviewees are more in line with traditional news coverage. It is important to note that our findings are purely correlational. While suggestive, one cannot conclude that the fake news producers and content shared by them *resulted in* the topics individuals remember. Indeed, the result we observe can very well be due to fake news domains focusing more on topics that were more memorable (discriminately for the Republican leaning voters). Indeed, *email* provides a good example here. While this word had a positive fake-ness, i.e. more central to fake news coverage, an interviewee could very well have heard about it from traditional news. While we cannot establish causality using this observational data without a more careful causal inference framework, the alignment between the fake news agenda and retained information extracted from Gallup interviews is worthy of further exploration.

The results presented for Donald Trump (second panel) are rather distinct from what we observed for Hillary Clinton. The most frequent words for all interviews were more in line with traditional news coverage compared to the first panel. The fake-ness becomes positive only beyond the most popular topics and compared to Hillary Clinton analysis, the magnitude is small. In addition, the difference between the Republican and Democrat leaning interviewees is much smaller—even hard to detect—and the directionality of the results is largely consistent between these two groups. What explains the relatively low degree to which survey responses align with fake news content about Donald Trump? One conjecture is as follows: Past research claims that fake and hyper-partisan sites were predominantly pro- Donald Trump and anti- Hillary Clinton [64]. Our findings suggest that they showed their ideological leaning not by writing positive articles about Donald Trump but by writing critical articles about Hillary Clinton. Such behavior could lead to a lack of alignment in this topic

Measuring fake-ness of interview responses as a function of demographic characteristics. Figure 4 demonstrates that party affiliation had a strong correlation with the fake-ness of the content interviewees remembered about each candidate. We investigate to what extend party affiliation and other demographic characteristics such as gender, age, race, and geographic area matters through statistical analysis.

To that end, we first measure the fake-ness of an interview response—or rather the fake-ness of the words uttered in it—as a function of the demographic characteristics of the interviewee

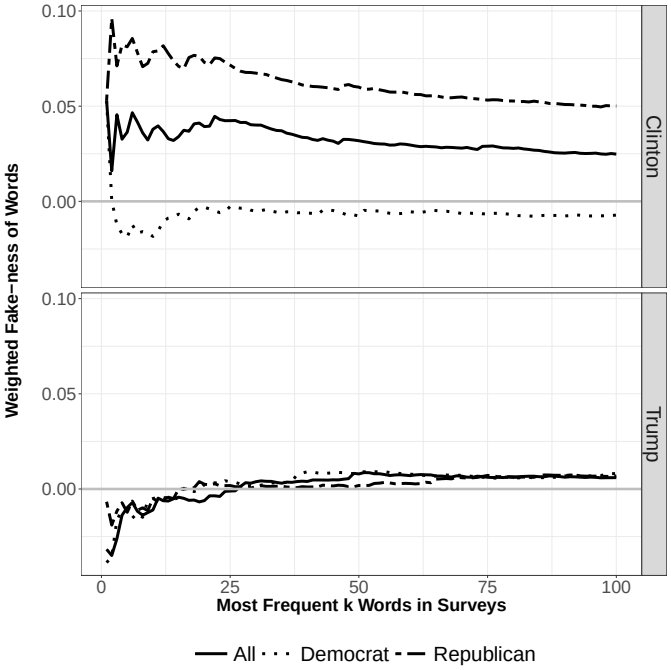


Figure 4: Fake-ness of the top-100 words uttered about Hillary Clinton and Donald Trump in interviews with respondents with party affiliations

and temporal patterns. In particular, we fit the following random-intercept multi-level regression model:

$$f_{i,j} = \beta_0 + \beta_1 party_i + \beta_2 age_i + gender_i + \beta_3 race_i + \beta_4 location_i + \beta_5 months_i + \alpha_i + \epsilon_{i,j} \quad (6)$$

$$\epsilon_{i,j} \sim N(0, \sigma_e^2) \quad (7)$$

$$\alpha_i \sim N(0, \sigma_u^2) \quad (8)$$

where $f_{i,j}$ is the fake-ness of the j−th word uttered by interviewee i, $party_i$ gives the party of interviewee i (Democrat, Independent, or Republican), age_i denotes the age (categorical scale, interview responses coded as 18-29, 30-49, 50-64, or 65+.), $gender_i$ denotes the gender of interviewee i, $race_i$ gives the race (Asian, Hispanic, Non-Hispanic White, and Non-Hispanic Black [11]), $location_i$ denotes the region where the interviewee resides (Southeast, New England, East Central/Great Lakes, Middle Atlantic, West Central, Rocky Mountain, Pacific or Southwest) and $months_i$ denotes the timing of the interview as the time between the interview and the election. This model allows grouping of all words uttered in the same response as opposed to treating them like independent measurements. We fit $f_{i,j}$ separately for response about Clinton and Trump to identify demographic and temporal characteristics that determine how aligned retained information for each candidate was with fake news coverage. Furthermore, since our fake-ness measurement of words that are more commonly observed in the news dataset is less noisy, we perform weighted generalized linear regression where each datapoint ($f_{i,j}$) is weighted by the frequency

[11] A fifth category of "Other" is filtered out.

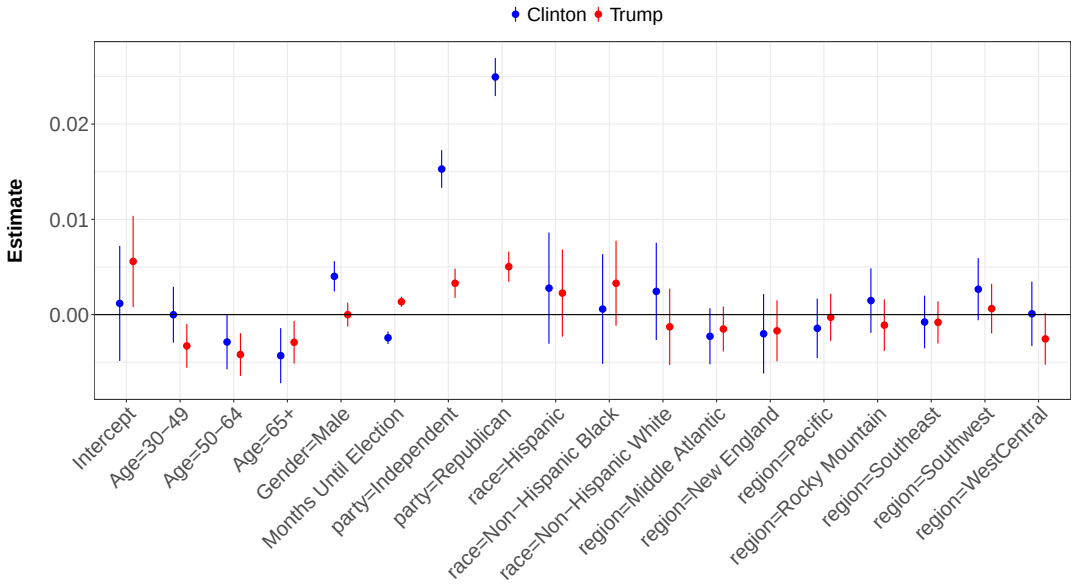


Figure 5: Coefficient estimates from linear mixed-effect models for fake-ness of recalled information about Clinton and Trump shared in interview responses as a function of interviewee characteristics. Base class for gender is female; race is Asian; region is East Central/Great Lakes. Data points are colored to according to the candidate the interviewee was listing retained information about. Each data point gives the coefficient estimate. Error bars mark standard errors of the estimates.

with which we observe the given word in our news sample (across fake and real news combined, in log-scale).

The estimated coefficients are provided in Figure 5. Statistically significant effect sizes are identified for age, gender, party, and timing. The effect directions are largely consistent across the Trump and Clinton responses. However, the effect sizes are significantly larger for Clinton responses. The exception to the consistency between the models is observed for timing patterns. The fake-ness of interview responses increases over time for retained information about Clinton while it decreases for Trump. This difference in temporal patterns can readily be explained by the *increasing* emphasis fake news producers placed on Clinton—observed through various analysis presented earlier in the paper (e.g. Figure 2). Focusing on effects with consistent direction and divergent effect sizes, we see that the Republicans (and to a lesser extent, Independents), younger and male interviewees had a higher tendency to remember information better aligned with fake news publishers. The effects were larger for Clinton. Geographical area and race are not statistically significant. The patterns observed for age are consistent with [3] that claims that fake news commonly spread on social media and older voters were less likely to be exposed to such content. Increased tendency of male respondents to share words with positive fake-ness measures are in line with recent research that showed significant gender effects in views of the two candidates. This highlights the importance of future research to better understand what characteristics are predictive of consuming and retaining information more aligned with fake news publishers.

9 LIMITATIONS

It is worthwhile to note a number of shortcomings of the analysis provided here. First, identifying fake, hyper-partisan, and otherwise fake news at the domain (as opposed to article) level, while supported by related work [45], has certain shortcomings. First, domains classified as fake at times might publish reliable news. Second, even mainstream news sites exhibit ideological bias and at times can disseminate misinformation. Future work that increases the accuracy of fake news classification—both at the article and domain levels—will be crucial for forming a richer understanding. This open problem is something that platforms are increasingly facing. The perfect solution is yet to be identified. This is evidenced in how platforms such as Facebook are alternating between publisher [19] and article [18] level solutions.

Second, content analysis performed in this paper is at the word level. Future studies that employ other content analysis methods and provide a rich qualitative analysis of open-ended Gallup responses can provide further insights about the alignment between fake news coverage and retained information about the candidates. Third, much of our analysis is based on Twitter. While Twitter is one of the popular social media sites used for sharing and consuming the news, it certainly is not the only one [29] and its population is not representative of the U.S. population. Furthermore, we can only examine articles shared at least once in our Twitter sample. Consequently, we cannot disentangle the effects of fake news production and consumption. Future work that examines *all* articles published by these producers can help us determine the effects of consumption and production separately.

Most notably, the analyses presented in this paper are purely correlational. Our findings unearth interesting patterns—such as the relationship between favorability of candidates and fake news consumption or the alignment between the information retained by voters and the content of fake news articles. However, such patterns are only suggestive. Finally, we do not consider the effects of fake news on readers' voting or other relevant attitudes. For instance, our study cannot help determine whether the spread of fake news determined the outcome of the 2016 U.S. Presidential election.

10 DISCUSSION AND IMPLICATIONS

Given the growing pace with which fake news spreads and the societal implications of such spread, it is crucial to examine fake news publisher content and understand its appeal. Motivated by this challenge, here we study the overall prevalence and focus of fake news content leading up to the 2016 U.S. Presidential election and examine how fake news production and consumption related to election campaign dynamics using a rich dataset that includes daily sample of random tweets mentioning the two presidential candidates, content of external urls shared in those tweets and nationally representative interviews conducted by Gallup at the daily level. Consequently, this paper extends our understanding fake news production and consumption and highlights various implications for platform owners and journalists.

First, despite initial fears raised by investigative journalism [64], we observe that fake news producer prevalence was limited compared to traditional news producers. Despite this positive finding, however, the prevalence of fake news producers was far from negligible. This highlights the importance of technology and design mechanisms that filter out unreliable content or provide means for the consumers to easily inspect the quality of a piece of information. Platform owners are already taking notice. For instance, Facebook has [19] announced in early 2018 that they plan to begin prioritizing "trustworthy" media outlets in users' feeds. Yet, this is a serious technological challenge that requires further research. Indeed, a few months later, Facebook announced that they started labeling content at the article (or video/photo) level [18].

Second, we show that the prevalence of fake news content quickly increased over the course of the campaign. This highlights the importance of tracking mechanisms that keep platform owners in the know early enough for the platform to respond to such challenges. Encouragingly, companies are beginning to reach out to academia to help them define important metrics to keep track of for combatting misinformation [36]. Our study highlights the importance of keeping track of prevalence of fake news content in a continuous manner.

Third, we present the first study to examine the relationship between the favorability of presidential candidates and the production and consumption of fake and traditional news. We show that the patterns in prevalence of fake news and favorability of the candidates are consistent. Indeed, we observe that increases in production and consumption of fake news *follows* increase in relative favorability of Clinton. This contribution highlights the importance of studying these two phenomena in conjunction. Yet, one must be cautious here. Our findings do not determine a *causal* relationship between these two trends.

Fourth, through semi-supervised machine learning techniques, we identify similarities and differences in agenda setting and coverage by fake and traditional news outlets. To the best of our knowledge, this is the first study to examines the similarities and differences between the agenda setting functions of fake and traditional news media leading up to the 2016 U.S. Presidential election. Some of our findings here suggest a failing by traditional news outlets, as evidenced by the similarity between the popular topics for fake and traditional news domains. Through the inspection of traditional news content, we observed that these outlets had a rather weak focus on policy and covered scandals of the two candidates unevenly. The behavior of traditional news outlets might be partially explained through the pressure of competing with fake news domains. Regardless, this brings about questions that journalists need to tackle to provide richer and more useful information to voters in the future. Our findings highlight the potential of tools that reveal to journalists similarities and differences between their coverage and the coverage of fake publishers.

Finally, we also showed that information voters retained about Clinton—especially by Republican voters—were more in line with fake news content than traditional news. Our study also highlights the relationship between demographic characteristics and fake-ness of the information people remembered about presidential candidates. The heterogeneity of vulnerability of different demographic groups will be important to keep in mind while building solutions to combat misinformation. This perhaps highlights the importance of diversity of social connections. Individuals with a diverse ego-network can be exposed to more diverse information and being exposed to diverse information can help one question the reliability of fake content. However, challenges still exist given the confirmation bias tendency. In fact, research is inconclusive. Diversity seems to play a positive role under some circumstances [16] while being negative in others [5]–highlighting the importance of context in which people are exposed to diverse ideas.

Where do we go from here? If fake news corrodes our democracy, what can and should be done about it? Social media platforms, while initially skeptical about their role [43], have taken steps to identify fake news content and warn their consumers [49, 61], mostly relying on independent organizations for fact checking. A similar step has been taken by search companies that provide news aggregation tools [72]. These are undoubtedly important steps towards a more robust news ecosystem. We believe further research on identifying and combating such problematic content is vital to the health of our democracy. We need to build the tools of strategy for traditional news organizations to stay relevant and compete with this growing industry and tools of literacy for news consumers to distinguish credible news content from unreliable ones.

ACKNOWLEDGMENTS

This research was partly supported by Michigan Institute for Data Science (MIDAS) at the University of Michigan and the National Science Foundation (Grant IIS-1815875). The author would like to thank Leticia Bode, Jonathan M. Ladd, Frank Newport, Josh Pasek, Lisa O. Singh, Stuart N. Soroka, and Michael W. Traugott for helpful comments and data.

REFERENCES

[1] Norah Abokhodair, Daisy Yoo, and David W. McDonald. 2015. Dissecting a Social Botnet: Growth, Content and Influence in Twitter. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. ACM, New York, NY, USA, 839–851. https://doi.org/10.1145/2675133.2675208

[2] Inc. Alexa Internet. 1996. Alexa–About Us. http://www.alexa.com/about.

[3] Hunt Allcott and Matthew Gentzkow. 2017. *Social media and fake news in the 2016 election.* Technical Report. National Bureau of Economic Research.

[4] Hunt Allcott, Matthew Gentzkow, and Chuan Yu. 2019. *Trends in the diffusion of misinformation on social media.* Technical Report. National Bureau of Economic Research.

[5] Christopher A Bail, Lisa Argyle, Taylor Brown, John Bumpuss, Haohan Chen, M.B. F Hunzaker, Jaemin Lee, Marcus Mann, Friedolin Merhout, and Alexander Volfovsky. 2018. Exposure to Opposing Views can Increase Political Polarization: Evidence from a Large-Scale Field Experiment on Social Media. https://doi.org/10.31235/osf.io/4ygux

[6] Michael Barbaro. 2016. Donald Trump Clung to 'Birther' Lie for Years, and Still Isn't Apologetic. https://www.nytimes.com/2016/09/17/us/politics/donald-trump-obama-birther.html.

[7] Alessandro Bessi and Emilio Ferrara. 2016. Social bots distort the 2016 U.S. Presidential election online discussion. *First Monday* 21, 11 (2016). http://firstmonday.org/ojs/index.php/fm/article/view/7090

[8] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[9] Leticia Bode, Ceren Budak., Jonathan M. Ladd, Frank Newport, Josh Pasek, Lisa O. Singh, Stuart N. Soroka, and Michael W. Traugott. Forthcoming. *Words That Matter: How the News and Social Media Shaped the 2016 Presidential Campaign.* Brookings Institution Press, Washington, D.C.

[10] Alexandre Bovet and Hernán A Makse. 2019. Influence of fake news in Twitter during the 2016 US presidential election. *Nature communications* 10, 1 (2019), 7.

[11] Lia Bozarth, Aparajita Saraf, and Ceren Budak. 2019. Higher Ground? How Groundtruth Labeling Impacts Our Understanding of the Spread of Fake News During the 2016 Election. https://www.ssrn.com/abstract=3340173. *SSRN* (2019).

[12] Ed Brayton. 2016. Please Stop Sharing Links to These Sites.

[13] Phil Brown and Jessica Minty. 2006. Media coverage & charitable giving after the 2004 tsunami. (2006).

[14] Ceren Budak, Divyakant Agrawal, and Amr El Abbadi. 2011. Limiting the Spread of Misinformation in Social Networks. In *Proceedings of the 20th International Conference on World Wide Web (WWW '11)*. ACM, New York, NY, USA, 665–674. https://doi.org/10.1145/1963405.1963499

[15] Ceren Budak, Sharad Goel, and Justin M Rao. 2016. Fair and balanced? quantifying media bias through crowdsourced content analysis. *Public Opinion Quarterly* 80, S1 (2016), 250–271.

[16] Ceren Budak and Duncan J Watts. 2015. Dissecting the spirit of Gezi: Influence vs. selection in the Occupy Gezi movement. *Sociological Science* 2 (2015), 370–397.

[17] Carlos Castillo, Mohammed El-Haddad, Jürgen Pfeffer, and Matt Stempeck. 2014. Characterizing the Life Cycle of Online News Stories Using Social Media Reactions. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '14)*. ACM, New York, NY, USA, 211–223. https://doi.org/10.1145/2531602.2531623

[18] Josh Constine. 2018. Facebook starts fact checking photos/videos, blocks millions of fake accounts per day. https://techcrunch.com/2018/03/29/facebook-fact-check-photos/.

[19] Rob Crilly. 2018. Facebook to start 'trust ratings' for media outlets as it fights back against fake news. http://www.telegraph.co.uk/news/2018/01/19/facebook-start-trust-\ratings-media-outlets-fights-back-against/.

[20] Clayton Allen Davis, Onur Varol, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2016. BotOrNot: A System to Evaluate Social Bots. In *Proceedings of the 25th International Conference Companion on World Wide Web (WWW '16 Companion)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 273–274. https://doi.org/10.1145/2872518.2889302

[21] Stefano DellaVigna and Ethan Kaplan. 2007. The Fox News effect: media bias and voting. *The Quarterly Journal of Economics* 122, 3 (2007), 1187–1234.

[22] Caitlin Dewey. 2016. Facebook Fake-News Writer:'I Think Donald Trump is in the White House Because of Me'.

[23] Susanna Dilliplane. 2011. All the News You Want to Hear: The Impact of Partisan News Exposure on Political Participation. *Public Opinion Quarterly* 75, 2 (2011), 287–316. https://doi.org/10.1093/poq/nfr006

[24] Seth Flaxman, Sharad Goel, and Justin M Rao. 2016. Filter bubbles, echo chambers, and online news consumption. *Public Opinion Quarterly* 80, S1 (2016), 298–320.

[25] Jack Fuller. 2010. *What is happening to news: The information explosion and the crisis in journalism.* University of Chicago Press, Chicago.

[26] Lauren Gambino. 2016. Trump Foundation: the allegations against Donald's charity, explained. https://www.theguardian.com/us-news/2016/oct/05/donald-trump-foundation-allegations-charity-new-york.

[27] Joshua Gillin. 2017. PolitiFact's Guide to Fake News Websites and What They Peddle. http://bit.ly/2o8kj3b.

[28] Sandra González-Bailón and Georgios Paltoglou. 2015. Signals of public opinion in online communication: A comparison of methods and data sources. *The ANNALS of the American Academy of Political and Social Science* 659, 1 (2015), 95–107.

[29] Jeffrey Gottfried and Elisa Shearer. 2016. News use across social media platforms 2016. http://www.journalism.org/2016/05/26/news-use-across-social-media-platforms-2016.

[30] N Grinberg, K Joseph, L Friedland, B Swire-Thompson, and D Lazer. 2018. *Fake news on Twitter during the 2016 US presidential election.* Technical Report. Working Paper. Available from the authors.

[31] Nir Grinberg, Kenneth Joseph, Lisa Friedland, Briony Swire-Thompson, and David Lazer. 2019. Fake news on Twitter during the 2016 US presidential election. *Science* 363, 6425 (2019), 374–378.

[32] Andrew Guess, Brendan Nyhan, and Jason Reifler. 2018. Selective Exposure to Misinformation: Evidence from the consumption of fake news during the 2016 US presidential campaign. , 14 pages.

[33] Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. Faking Sandy: Characterizing and Identifying Fake Images on Twitter During Hurricane Sandy. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13 Companion)*. ACM, New York, NY, USA, 729–736. https://doi.org/10.1145/2487788.2488033

[34] ANDREW HIGGINS, MIKE McINTIRE, and GABRIEL J.X. DANCE. 2016. Inside a Fake News Sausage Factory: ?This Is All About Income? https://www.nytimes.com/2016/11/25/world/europe/fake-news-donald-trump-hillary-clinton-georgia.html.

[35] Pik-Mai Hui, Chengcheng Shao, Alessandro Flammini, Filippo Menczer, and Giovanni Luca Ciampaglia. 2018. The Hoaxy Misinformation and Fact-Checking Diffusion Network. In *Proceedings of the Twelfth International Conference on Web and Social Media, ICWSM 2018, June 25-28, 2018.* AAAI Press, Stanford, California, USA, 528–530. https://aaai.org/ocs/index.php/ICWSM/ICWSM18/paper/view/17851

[36] Twitter Inc. 2018. Twitter Health Metrics Proposal Submission. https://blog.twitter.com/official/en_us/topics/company/2018/twitter-health-metrics-proposal-submission.html.

[37] Shanto Iyengar, Helmut Norpoth, and Kyu S Hahn. 2004. Consumer demand for election news: The horserace sells. *Journal of Politics* 66, 1 (2004), 157–175.

[38] Jagadeesh Jagarlamudi, Hal Daumé, III, and Raghavendra Udupa. 2012. Incorporating Lexical Priors into Topic Models. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL '12)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 204–213. http://dl.acm.org/citation.cfm?id=2380816.2380844

[39] Elise Jordan. 2016. Hillary Clinton's Reaction to Her Foundation Scandal Is Disastrous. http://time.com/4466911/hillary-clinton-email-excuses/.

[40] Dan Jurafsky, Victor Chahuneau, Bryan R Routledge, and Noah A Smith. 2014. Narrative framing of consumer sentiment in online restaurant reviews. *First Monday* 19, 4 (2014).

[41] Richard L Kaplan. 2002. *Politics and the American press: The rise of objectivity, 1865-1920.* Cambridge University Press, Cambridge.

[42] Brian C. Keegan. 2018. The Dynamics of Peer-Produced Political Information During the 2016 U.S. Presidential Campaign. In *Proceedings of the Twelfth International Conference on Web and Social Media, ICWSM 2018, June 25-28, 2018.* AAAI Press, Stanford, California, USA, 632–635. https://aaai.org/ocs/index.php/ICWSM/ICWSM18/paper/view/17800

[43] Kia Kokalitcheva. 2016. Mark Zuckerberg Says Fake News on Facebook Affecting the Election Is a 'Crazy Idea'. http://fortune.com/2016/11/11/facebook-election-fake-news-mark-zuckerberg/.

[44] Haewoon Kwak, Jisun An, Joni Salminen, Soon-Gyo Jung, and Bernard J. Jansen. 2018. What We Read, What We Search: Media Attention and Public Attention Among 193 Countries. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 893–902. https://doi.org/10.1145/3178876.3186137

[45] David M. J. Lazer, Matthew A. Baum, Yochai Benkler, Adam J. Berinsky, Kelly M. Greenhill, Filippo Menczer, Miriam J. Metzger, Brendan Nyhan, Gordon Pennycook, David Rothschild, Michael Schudson, Steven A. Sloman, Cass R. Sunstein, Emily A. Thorson, Duncan J. Watts, and Jonathan L. Zittrain. 2018. The science of fake news. *Science* 359, 6380 (2018), 1094–1096. https://doi.org/10.1126/science.aao2998 arXiv:http://science.sciencemag.org/content/359/6380/1094.full.pdf

[46] Jim Maddock, Kate Starbird, Haneen J. Al-Hassani, Daniel E. Sandoval, Mania Orand, and Robert M. Mason. 2015. Characterizing Online Rumoring Behavior Using Multi-Dimensional Signatures. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. ACM, New York, NY, USA, 228–241. https://doi.org/10.1145/2675133.2675280

[47] Alice Marwick and Rebecca Lewis. 2017. Media Manipulation and Disinformation Online.

[48] Burt L Monroe, Michael P Colaresi, and Kevin M Quinn. 2008. Fightin'words: Lexical feature selection and evaluation for identifying the content of political

conflict. *Political Analysis* 16, 4 (2008), 372–403.

[49] Adam Mosseri. 2016. News feed fyi: Addressing hoaxes and fake news. http://newsroom.fb.com/news/2016/12/news-feed-fyiaddressing-hoaxes-and-fake-news.

[50] Supun Nakandala, Giovanni Luca Ciampaglia, Norman Makoto Su, and Yong-Yeol Ahn. 2016. Gendered Conversation in a Social Game-Streaming Platform. *CoRR* abs/1611.06459 (2016). arXiv:1611.06459 http://arxiv.org/abs/1611.06459

[51] David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. 2010. Automatic Evaluation of Topic Coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 100–108. http://dl.acm.org/citation.cfm?id=1857999.1858011

[52] A. Conrad Nied, Leo Stewart, Emma Spiro, and Kate Starbird. 2017. Alternative Narratives of Crisis Events: Communities and Social Botnets Engaged on Social Media. In *Companion of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17 Companion)*. ACM, New York, NY, USA, 263–266. https://doi.org/10.1145/3022198.3026307

[53] Hannah Jane Parkinson. 2016. Click and elect: how fake news helped Donald Trump win a real election.

[54] Josh Pasek, Lisa O. Singh, Yifang Wei, Stuart N. Soroka, Jonathan M. Ladd, Michael W. Traugott, Ceren Budak, Leticia Bode, and Frank Newport. 2018. Attention to Campaign Events: Do Twitter and Self-Report Metrics Tell the Same Story? *Forthcoming in BIGSURV18: Big Data Meets Survey Science Edited Volume* (2018).

[55] Thomas E Patterson. 2016. *News Coverage of the 2016 General Election: How the Press Failed the Voters.* Technical Report. Harvard University, John F. Kennedy School of Government.

[56] Paul Resnick, Samuel Carton, Souneil Park, Yuncheng Shen, and Nicole Zeffer. 2014. Rumorlens: A system for analyzing the impact of rumors and corrections in social media.

[57] Paul Resnick, R. Kelly Garrett, Travis Kriplean, Sean A. Munson, and Natalie Jomini Stroud. 2013. Bursting Your (Filter) Bubble: Strategies for Promoting Diverse Exposure. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work Companion (CSCW '13)*. ACM, New York, NY, USA, 95–100. https://doi.org/10.1145/2441955.2441981

[58] Filipe Nunes Ribeiro, Lucas Henrique, Fabrício Benevenuto, Abhijnan Chakraborty, Juhi Kulshrestha, Mahmoudreza Babaei, and Krishna P. Gummadi. 2018. Media Bias Monitor: Quantifying Biases of Social Media News Outlets at Large-Scale. In *Proceedings of the Twelfth International Conference on Web and Social Media, ICWSM 2018, June 25-28, 2018*. AAAI Press, Stanford, California, USA, 290–299. https://aaai.org/ocs/index.php/ICWSM/ICWSM18/paper/view/17878

[59] S Schaedel. 2018. Websites that post fake and satirical stories. FactCheck. https://www.factcheck.org/2017/07/websites-post-fake-satirical-stories.

[60] Chengcheng Shao, Giovanni Luca Ciampaglia, Alessandro Flammini, and Filippo Menczer. 2016. Hoaxy: A Platform for Tracking Online Misinformation. In *Proceedings of the 25th International Conference Companion on World Wide Web (WWW '16 Companion)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 745–750. https://doi.org/10.1145/2872518.2890098

[61] Mallory Shelbourne. 2017. Facebook rolls out feature to combat fake news. http://thehill.com/policy/technology/322427-facebook-begins-to-introduce-new-\feature-to-combat-fake-news.

[62] Ignacio Siles and Pablo J Boczkowski. 2012. Making sense of the newspaper crisis: A critical assessment of existing research and an agenda for future work. *New Media & Society* 14, 8 (2012), 1375–1394.

[63] Craig Silverman. 2016. Here are 50 of the biggest fake news hits on Facebook from 2016. https://www.buzzfeed.com/craigsilverman/top-fake-news-of-2016.

[64] C Silverman. 2016. This analysis shows how fake election news stories outperformed real news on Facebook. https://www.buzzfeed.com/craigsilverman/viral-fake-election-news-outperformed-real-news-on-facebook.

[65] Craig Silverman, Jane Lytvynenko, and Scott Pham. 2017. These are 50 of the biggest fake news hits on Facebook in 2017. https://www.buzzfeednews.com/article/craigsilverman/these-are-50-of-the-biggest-fake-news-hits-on-facebook-in.

[66] Craig Silverman and Jeremy Singer-Vine. 2016. Most Americans who see fake news believe it, new survey says. https://www.buzzfeed.com/craigsilverman/fake-newssurvey.

[67] Stuart N Soroka. 2012. The gatekeeping function: Distributions of information in media and the real world. *The Journal of Politics* 74, 2 (2012), 514–528.

[68] Kate Starbird. 2017. Examining the Alternative Media Ecosystem Through the Production of Alternative Narratives of Mass Shooting Events on Twitter.. In *ICWSM*. AAAI Press, Montréal, Québec, Canada, 230–239.

[69] Art Swift. 2016. Americans' trust in mass media sinks to new low.

[70] William N Venables and Brian D Ripley. 2013. *Modern applied statistics with S-PLUS.* Springer Science & Business Media, Berlin, Germany.

[71] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science* 359, 6380 (2018), 1146–1151. https://doi.org/10.1126/science.aap9559 arXiv:http://science.sciencemag.org/content/359/6380/1146.full.pdf

[72] Nick Wingfield, Mike Isaac, and Katie Benner. 2016. Google and Facebook take aim at fake news sites. *The New York Times, November* 14 (2016), 2016.

[73] Donghee Yvette Wohn, Casey Fiesler, Libby Hemphill, Munmun De Choudhury, and J. Nathan Matias. 2017. How to Handle Online Risks?: Discussing Content Curation and Moderation in Social Media. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. ACM, New York, NY, USA, 1271–1276. https://doi.org/10.1145/3027063.3051141

[74] Lori Young and Stuart Soroka. 2012. Lexicoder sentiment dictionary.

[75] Melissa Zimdars. 2016. False, Misleading, Clickbait-y, and Satirical "News" Sources. https://docs.google.com/document/u/1/d/10eA5-mCZLSS4MQY5QGb5ewC3VAL6pLkT53V_81ZyitM/mobilebasic.

Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences

Yixin Cao
National University of Singapore
caoyixin2011@gmail.com

Xiang Wang
National University of Singapore
xiangwang1223@gmail.com

Xiangnan He*
University of Science and Technology of China
xiangnanhe@gmail.com

Zikun Hu
National University of Singapore
zikunhu1016@gmail.com

Tat-Seng Chua
National University of Singapore
dcscts@nus.edu.sg

ABSTRACT

Incorporating knowledge graph (KG) into recommender system is promising in improving the recommendation accuracy and explainability. However, existing methods largely assume that a KG is complete and simply transfer the "knowledge" in KG at the shallow level of entity raw data or embeddings. This may lead to suboptimal performance, since a practical KG can hardly be complete, and it is common that a KG has missing facts, relations, and entities. Thus, we argue that it is crucial to consider the incomplete nature of KG when incorporating it into recommender system.

In this paper, we jointly learn the model of recommendation and knowledge graph completion. Distinct from previous KG-based recommendation methods, we transfer the **relation** information in KG, so as to understand the reasons that a user likes an item. As an example, if a user has watched several movies *directed by* (relation) the *same person* (entity), we can infer that the director relation plays a critical role when the user makes the decision, thus help to understand the user's preference at a finer granularity.

Technically, we contribute a new translation-based recommendation model, which specially accounts for various preferences in translating a user to an item, and then jointly train it with a KG completion model by combining several transfer schemes. Extensive experiments on two benchmark datasets show that our method outperforms state-of-the-art KG-based recommendation methods. Further analysis verifies the positive effect of joint training on both tasks of recommendation and KG completion, and the advantage of our model in understanding user preference. We publish our project at https://github.com/TaoMiner/joint-kg-recommender.

CCS CONCEPTS

• **Information systems** → **Personalization**.

KEYWORDS

Item Recommendation;Knowledge Graph;Embedding;Joint Model;

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313705

ACM Reference Format:
Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313705

1 INTRODUCTION

Knowledge Graph (KG) is a heterogeneous structure that stores the world's knowledge in the form of machine-readable graphs, where nodes denote entities, and edges denote the relations between entities. Since proposed, KG has attracted much attention in many fields, ranging from recommendation [40], dialogue system [18, 21], to information extraction [3]. Focusing on recommendation, the structural knowledge has shown great potential in providing rich information about the items, offers a promising solution to improving the accuracy and explainability of recommender systems.

Nevertheless, existing KGs (e.g., DBPedia [20]) are far from complete, which limits the benefits of the transferred knowledge. As illustrated in Figure 1, the red dashed line between *Robert Zemeckis* and *Death Becomes Her* indicates a missing relation *isDirectorOf*. Assuming that the user has chosen movies *Back to The Future I & II* and *Forrest Gump*, by using the KG, we can attribute the reason of user's choices to the director *Robert Zemeckis*. In this case, although we have accurately captured the user's preference on movies, we may still fail to recommend *Death Becomes Her* (which is also of interest to the user), due to the missing relation in the KG (cf. the red dashed line). As such, we believe that it is critical to consider the incomplete nature of KG when using it for recommendation, and more interestingly, can the completion of KG benefit from the improved modeling of user-item interactions?

In this paper, we propose to unify the two tasks of recommendation and KG completion in a joint model for mutually enhancements. The basic idea is twofold: 1) utilizing the facts in KG as auxiliary data to augment the modeling of user-item interaction, and 2) completing the missing facts in KG based on the enhanced user-item modeling. For example, we are able to understand the user's preference on director via the related entities and relations; meanwhile we can predict that *Robert Zemeckis* is the director of *Death Becomes Her*, if there exist some users who like the movie also have a preference of other movies directed by *Robert Zemeckis*.

Although many prior efforts have leveraged KG in recommender systems [16, 29, 30, 44, 48, 50], there are few works that jointly

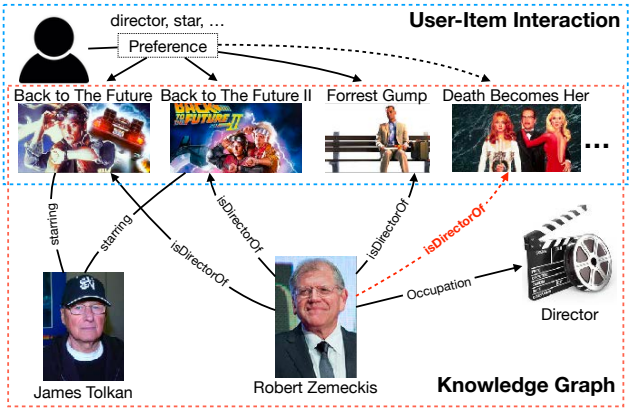


Figure 1: An illustrative example on the necessity of considering the missing relations in KG for recommendation.

model the two tasks of knowledge graph learning and recommendation. CoFM [31] is the most similar work that aligns the two latent vector spaces in each task together by regularizing or sharing entity and item embeddings if they refer to the same thing. However, it ignores the important role of entity relations in user-item modeling, and fails to offer interpretation ability.

In this work, we propose a **T**ranslation-based **U**ser **P**reference model (TUP) to integrate with KG learning seamlessly. The key idea is that there exists multiple (implicit) relations between users and items, which reveal the preferences (i.e., reasons) of users on consuming items. An example of the "preference" is the director information in Figure 1 that drives the user to watch the movies *Back to Future I & II* and *Forest Gump*. While we can pre-define the number of preferences and train TUP from user-item interaction data, the preferences are expressed as latent vectors that are opaque to deeper understanding. To endow the preferences with explicit semantics, we align them with the relations in KG, capturing the intuition that the type of item attributes plays a crucial role in user decision-making process. Technically speaking, we transfer the relation embeddings as well as entity embeddings learned from KG to TUP, simultaneously training the KG completion and recommendation tasks. We term the method as **K**nowledge-enhanced TUP (KTUP) that jointly learns the representations of users, items, entities, and relations. The main contributions are summarized:

- We propose a new translation-based model, which exploits the implicit preference representations to capture the relations between users and items.
- We emphasize the importance of jointly modeling item recommendation and KG completion to couple the preference representations with the knowledge-aware relations, thus empowering the model with explainability.
- We perform extensive experiments on two datasets for top-N recommendation and KG completion tasks, which verify the rationality of joint learning. Experimental results demonstrate the effectiveness and explainability of our model.

2 RELATED WORK

Our proposed methods involve two tasks: item recommendation and KG completion. In this section, we first introduce related work for each task, before discussing their relations.

2.1 Item Recommendation

In the early stage of item recommendation, researchers focus on recommending similar users or items to a target user using history interactions alone, such as collaborative filtering (CF) [35], factorization machines [33], matrix factorization techniques [19], BPRMF [34], and their extensions [46]. The key challenge here lies in extracting features of users and items to compute their similarity, namely **similarity-based methods**.

With the surge of neural network (NN) models, a host of methods extend **similarity-based methods with NN** and present a more effective mechanism in automatically extracting latent features of users and items for recommendation [7, 12–14]. However, they still suffer from the data sparsity issue and cold-start problem. **Content-based methods** deal with the issues by introducing various side information, such as the contextual reviews [9, 25], relational data [10, 36] and knowledge graphs [6]. Another advantage of additional contents is the improved explainable ability to understand why an item is being recommended out of others. This has been found to be important for the effectiveness, efficiency, persuasiveness, and user satisfaction of recommender systems [8, 39, 49].

Among the side information, knowledge graphs (e.g., DBPedia [20]) show great potentials on recommendations due to its well-defined structures and adequant resources. This type of methods mostly transfer structural knowledge of entities from KG to user-item interaction modeling based on the given mapping between entities and items. We roughly classify them into two groups: the methods augmenting the data of user-item pairs with KG triplets, and the methods combining item and entity embeddings learned from different sources. In the first group, Piao and Breslin [29] extracted lightweight features drived from KG (i.e. property-object, subject-property) for factorization machines. Zhang et al. [48] constructed a unified graph by adding a *buy* relation between users and items, then applied transE [2] to model relational data. On the other hand, the methods in the second group usually improve the quality of item embeddings using entity embeddings if they refer to the same thing [16, 44]. Piao and Breslin [30] summerized the recommendation results using different entity embeddings (i.e. node2vec, doc2vec and transE), and found that node2vec improves the most. CoFM [31] first takes into account the refinements of entity embeddings from user-item modeling as another transfering task. However, the above methods heavily rely on the alignments between items and entities. Zhou et al. [50] introduced entity concepts in KG to deal with the sparisity issue of the alignments, but still fail to consider the importance of entity relations in transfering knowledge from KG.

Another line of work is **translation-based recommendation**, inspired by KG representation learning. It assumes that the selection of items satisfies translational relations in the latent vector space, where the relations are either regarded as related to users in sequential recommendation [11], or modeled implicitly via Memory-based Attention [37]. We thus improve this type of method by considering the N-to-N issue[1] in modeling user preferences as translational relations, which shall be further enhanced by transfering knowledge of entities and their relations from KG.

[1]N-to-N issue here means that one user may like multiple items, and also, several users may like a single item, which will be detailed in Section 4.2.

2.2 KG Completion

External knowledge has been found to be effective in many tasks of natural language processing, such as the question answering [47], which accelerates the popularity of KGs. Although there are a host of methods for finding entities [4, 5] and their relations from texts [22], existing KGs are far from complete. Recent studies for KG completion show a great enthusiasm for learning low-dimensional represention of entities and relations while perserving structural knowledge of the graph. We roughly categorize such representation learning methods into two groups: translational distance models and semantic matching models.

TransE [2] first proposed the core idea of translational distance models that the relationship between two entities corresponds to a translation in their vector space. Although it is simple and effective, it is sometimes confusing because some relations can translate one entity to various entities, namely the 1-to-N problem. Similarly, there are other N-to-1 and N-to-N problems. To address these problems, a host of methods extended TransE by introducing additional hyperplanes [42], vector spaces [23], textual information [41] and relational path [22].

The second group measures the plausibility of facts by matching semantic representations of entities and relations through similarity-based scoring functions. RESCAL [27] represents each relation as a matrix to capture the compositional semantics between entities, and utilizes a bilinear function as similarity metrics. To simplify the learning of relation matrices, DistMult [43] restricts them to be diagonal, HolE [26] defines a circular correlation [32] to compress relation matrices as vectors, and ComplEx [38] introduces complex-value for asymmetric relations. Instead of modeling the compositional relation, another line of methods directly introduce NNs for matching. SME [1] learns relation specific layers for head entity and tail entity, respectively, and then feeds them into the final matching layer (e.g., dot production), while NAM [24] conducts the semantic matching with a deep architecture.

2.3 Relationship between Two Tasks

Items usually correspond to entities in many fields, such as books, movies and musics, making it possible to transfer knowledge between fields. These information involving in two tasks are complementary, revealing the connectivity among items or between users and items. In terms of models, the two tasks both aim to rank candidates given a query (i.e., an entity or a user) as well as their implict or explict relatedness. For example, KG completion aims to find correct movies (e.g., *Death Becomes Her*) for the person *Robert Zemeckis* given the explicit relation *isDirectorOf*, while item recommendation aims at recommending movies for a target user satisfying some implicit preference. Therefore, we are able to fill in the gap between item recommendation and KG completion via a joint model, to systematically investigate how the two tasks impact each other.

3 PRELIMINARY

Before introducing our proposed methods, let's first formally define the two tasks as well as TransH [42] as the component for KG completion in our model.

3.1 Tasks and Notations

Item Recommendation : Given a list of user-item interactions $\mathcal{Y} = \{(u, i)\}$, we use implicit feedback as the protocol so that each pair (u, i) implies the user $u \in \mathcal{U}$ consumes the item $i \in \mathcal{I}$. The goal is to recommend top-N items for a target user.

KG Completion : A **Knowledge Graph** $\mathcal{KG}$ is a directed graph composed of *subject-property-object* triple facts. Each triplet denotes that there is a relationship r from head entity e_h to tail entity e_t, formally defined by (e_h, e_t, r), where $e_h, e_t \in \mathcal{E}$ are entities and $r \in \mathcal{R}$ are relations. Due to the incompleteness nature of KGs, KG completion is to predict the missing entity e_h or e_t for a triplet (e_h, e_t, r), which can also be regarded as recommending top-N entities for a target (e_t, r) or (e_h, r).

TUP denotes the model for item recommendation. It takes a list of user-item pairs $\mathcal{Y}$ as input, and outputs a relevance score $g(u, i; p)$ indicating the likelihood that u likes i, given the preference $p \in \mathcal{P}$, where the number of the preference set $\mathcal{P}$ is predefined. For each user-item pair, we induce a preference, serving as a similar role with the relation for two entities. To deal with the N-to-N issue, we introduce preference hyperplanes, and assign each preference with two vectors: $\mathbf{w}_p$ for the projection to a hyperplane, $\mathbf{p}$ for the translation between users and items.

KTUP is a multi-task architecture. Given $\mathcal{KG}$, $\mathcal{Y}$, and a set of item-entity alignments $\mathcal{A} = \{(i, e)|i \in \mathcal{I}, e \in \mathcal{E}\}$, where each (i, e) means that i can be mapped to an entity e in the given KG. KTUP is able to output not only $g(u, i; p)$, but also a score $f(e_h, e_t, r)$ indicating how possible the fact is true, based on the jointly learned embeddings of users $\mathbf{u}$, items $\mathbf{i}$, preferences $\mathbf{p}, \mathbf{w}_p$, entities $\mathbf{e}$ and relations $\mathbf{r}, \mathbf{w}_r$.

Example 3.1. As shown in Figure 1, given a user, the interacted movies (e.g., *Back to The Future I & II* and *Forrest Gump*), and the related triplets, KTUP is able to (1) figure out the user preference of the *isDirectorOf* relation on movies, (2) recommend the movie *Death Becomes Her* based on the induced preference, and (3) predict the missing head or tail entity for the triplet (*Death Becomes Her-isDirectorOf-Robert Zemeckis*). The above three goals shall be achieved by considering not only the structural knowledge in KG but also the user-item interactions.

Next, we will briefly describe transH as the module of KG completion in our joint model.

3.2 TransH for KG Completion

Learning distributional representations of KG provides an effective and efficient way of manipulating entities while perserving their structural knowledge. TransE [2] is a widely used method due to its simplicity and remarkable effectiveness, inspiring many research fileds [45]. Its basic idea is to learn embeddings for entities and relations, satisfying $\mathbf{e}_h + \mathbf{r} \approx \mathbf{e}_t$ if there is a triplet (e_h, e_t, r) in KG. However, a single relation type may correspond to multiple head entities or tail entities, leading to serious 1-to-N, N-to-1 and N-to-N issues [42].

Therefore, TransH [42] learns different representations for an entity conditioned on different relations. It assumes that each relation owns a hyperplane, and the translation between head entity and

tail entity is valid only if they are projected to the same hyperplane. It defines an energy score function for a triplet as follows:

$$f(e_h, e_t, r) = \| \mathbf{e}_h^{\perp} + \mathbf{r} - \mathbf{e}_t^{\perp} \| \tag{1}$$

where a lower score of $f(e_h, e_t, r)$ indicates that the triplet is possbily true, otherwise no. $\mathbf{e}_h^{\perp}$ and $\mathbf{e}_t^{\perp}$ are projected entity vectors:

$$\mathbf{e}_h^{\perp} = \mathbf{e}_h - \mathbf{w}_r^{\mathrm{T}} \mathbf{e}_h \mathbf{w}_r \tag{2}$$

$$\mathbf{e}_t^{\perp} = \mathbf{e}_t - \mathbf{w}_r^{\mathrm{T}} \mathbf{e}_t \mathbf{w}_r \tag{3}$$

where $\mathbf{w}_r$ and $\mathbf{r}$ are two learned vectors of relation r, $\mathbf{w}_r$ denotes the projection vector of the corresponding hyperplane, and $\mathbf{r}$ is the translation vector. $\| \cdot \|$ denotes the L1-norm distance function used in the presented paper.

Finally, the training of TransH encourages the discrimination between valid triplets and incorrect ones using margin-based ranking loss:

$$\mathcal{L}_k = \sum_{(e_h, e_t, r) \in \mathcal{KG}} \sum_{(e'_h, e'_t, r') \in \mathcal{KG}^-} [f(e_h, e_t, r) + \gamma - f(e'_h, e'_t, r')]_+ \tag{4}$$

where $[\cdot]_+ \triangleq \max(0, \cdot)$, $\mathcal{KG}^-$ contains incorrect triplets constructed by replacing head entity or tail entity in a valid triplet randomly, and γ controls the margin between positive and negative triplets.

4 TUP FOR ITEM RECOMMENDATION

Inspired by the above translation assumption between two entities in KG, we propose TUP to explictly models user preferences and regards them as translational relationships between users and items. Given a set of user-item interactions $\mathcal{Y}$, it automatically induces a preference for a user-items pair, and learns the embeddings of preference $\mathbf{p}$, user $\mathbf{u}$ and item $\mathbf{i}$, satisfying $\mathbf{u} + \mathbf{p} \approx \mathbf{i}$.

Considering the implicity and variety of user preferences, we design two main components in TUP: Preference Induction and Hyperplane-based Translation.

4.1 Preference Induction

Given a user-item pair (u, i), this component is to induce a preference from a set of latent factors $\mathcal{P}$. These factors are shared by all users, and each $p \in \mathcal{P}$ denotes a different preference, which aims at capturing the commonality among users as global features complementary to user embeddings that focus on a single user locally. Similar with topic models, the number $P = |\mathcal{P}|$ is a hyperparameter, and we cannot nominate the exact meaning of each preference. With the help of KG, the number of preferences can be automatically set and each preference is assigned with explanations (Section 5).

We design two strategies for preference induction: a hard approach that selects one out of the P preferences, and a soft way that combines all preferences with attentions.

4.1.1 Hard Strategy. The intuition behind our hard strategy is that only a single preference takes effect when a user makes a decision over items. We use Straight-Through (ST) Gumbel SoftMax [17] for discretely sampling the preference given a user-item pair, which utilizes reparameterization trick for backpropagation, making it possible to calculate the continuous gradients of model parameters during the end-to-end training.

ST Gumbel SoftMax approximately samples one-hot vectors from a multi-classification distribution. Assuming that the probability of belonging to class p in a P-way classification distribution is defined as log softmax:

$$\phi(p) = \frac{\exp(\log(\pi_p))}{\sum_{j=1}^{P} \exp(\log(\pi_j))} \tag{5}$$

where π_p is the unnormalized output of a score function. Then, we sample a one-hot vector $\mathbf{z} = [z_1, \cdots, z_P] \in \mathbb{R}^P$ from the above distribution as follows:

$$z_p = \begin{cases} 1, & p = \arg\max_j(\log(\pi_j) + g_j) \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $g = -\log(-\log(u))$ is the Gumbel noise and u is generated by a certain noise distribution (e.g., $u \sim \mathcal{N}(0, 1)$). The noise term increases the stochastic of arg max function and makes the process become equivalent to drawing a sample accroding to a continuous probability distribution $\mathbf{y} = [y_1, \cdots, y_p, \cdots, y_P]$:

$$y_p = \frac{\exp((\log(\pi_p) + g_p)/\tau)}{\sum_{j=1}^{P} \exp((\log(\pi_j) + g_j)/\tau)} \tag{7}$$

This is called Gumbel-Softmax distribution, where τ is a temperature parameter. The related proofs can be found in the original papers.

Straight-Through (ST) gumbel-Softmax takes different paths in the forward and backward propagation, so as to maintain sparsity yet support stochastic gredient descent (SGD). In the forward pass, it discretizes the continuous probability distribution for a one-hot vector as mentioned above. And in the backward pass, it simply follows the continuous $\mathbf{y}$, thus the error signal is still able to be backpropagated.

In the hard strategy, we define the score function for π_p as the similarity between the user-item pair and preference:

$$\phi(u, i, p) = \text{Similarity}(\mathbf{u} + \mathbf{i}, \mathbf{p}) \tag{8}$$

We use dot product as similarity function.

4.1.2 Soft Strategy. Actually, a user might like an item according to various factors, which have no distinct boundary. Instead of selecting the most prominent preference, the soft strategy is to combine multiple preferences via the attention mechanism:

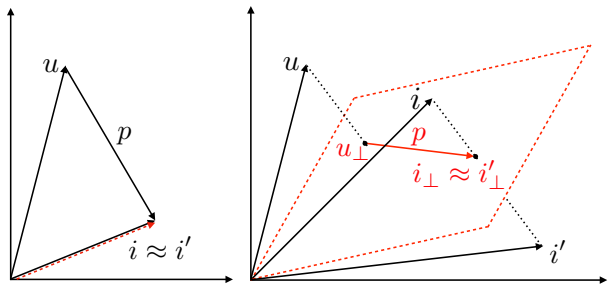
$$\mathbf{p} = \sum_{p' \in \mathcal{P}} \alpha_{p'} \mathbf{p}' \tag{9}$$

where $\alpha_{p'}$ is the attention weight of preference p', and defined as proportional to the similarity score:

$$\alpha_{p'} \propto \phi(u, i, p') \tag{10}$$

4.2 Hyperplane-based Translation

Inspired by TransH, we introduce the hyperplane to deal with the variety of preferences. That is, different users may share the same preference to different items (i.e., N-to-N issue), which is pretty common in practice. Obviously, it is confusing for the TransE-like



(a) TransE (b) TransH

Figure 2: Illustration of the two translation schemes for item recommendation

transition: the embeddings of items are close as long as a user who likes both of them are due to the some preference (Figure 2(a)), leading to an incorrect conclusion that a user consuming one shall consume the other, no matter what the user's preference is.

Such limitations are alleviated by introducing perference hyperplanes as illustrated in Figure 2(b): i and i' have different representations, and are similar only when they are projected to a specific hyperplane. We thus define the hyperplane-based translation function as follows:

$$g(u, i; p) = \| \mathbf{u}^{\perp} + \mathbf{p} - \mathbf{i}^{\perp} \| \tag{11}$$

where $\mathbf{u}^{\perp}$ and $\mathbf{i}^{\perp}$ are projected vectors of the user and the item, and are obtained through the induced preference p that plays a similar role as relations in TransH:

$$\mathbf{u}^{\perp} = \mathbf{u} - \mathbf{w}_p^{\mathrm{T}} \mathbf{u} \mathbf{w}_p \tag{12}$$

$$\mathbf{i}^{\perp} = \mathbf{i} - \mathbf{w}_p^{\mathrm{T}} \mathbf{i} \mathbf{w}_p \tag{13}$$

where $\mathbf{w}_p$ is the projection vector that is obtained along with the induction process of preferences p: either to pick up the corresponding one using the hard strategy, or through attentive addition of all projection vectors based on the induced attention weights in the soft strategy:

$$\mathbf{w}_p = \sum_{p' \in \mathcal{P}} \alpha_{p'} \mathbf{w}_{p'} \tag{14}$$

We encourage the translation distances of the interacted items to be smaller than random ones for each user through BPR Loss function:

$$\mathcal{L}_p = \sum_{(u,i) \in \mathcal{Y}} \sum_{(u,i') \in \mathcal{Y}'} -\log \sigma[g(u, i'; p') - g(u, i; p)] \tag{15}$$

where $\mathcal{Y}'$ contains negative interactions by randomly corrupting an interacted item to a non-interacted one for each user.

Tranditional methods (e.g., BPRMF [34]) recommend items for a user by computing a scalar score based on user and item embeddings, which indicates to which extent the user prefers to the item. Instead, we model preferences as vectors in order to (1) capture the commonality among users as global latent features, compared with user embeddings which only concerns with the local features of the user alone, and (2) reflect richer semantics for explainable ability.

5 JOINT LEARNING VIA KTUP FOR TWO TASKS

KTUP extends the translation-based recommendation model, TUP, by incorporating KG knowledge of entities as well as relations. Intuitively, the auxiliary knowledge supplements the connectivity among items as constraints to model user-item pairs. On the other hand, the understanding of users' preferences to items shall reveal their commonality related to some relation types and entities, which may be missing in the given KG.

5.1 KTUP

Figure 3 presents the overall framework of KTUP. On the left side is the inputs: user-item interactions, knowledge graph and the alignments between items and entities. At the top-right corner is TUP for item recommendation, while TransH for knowledge graph completion is at the bottom-right corner. KTUP jointly learns the two tasks by enhancing the embeddings of items and preferences with that of entities and relations. We define the knowledge enhanced TUP translation function as follows:

$$g(u, i; p) = \| \mathbf{u}^{\perp} + \hat{\mathbf{p}} - \hat{\mathbf{i}}^{\perp} \| \tag{16}$$

where $\hat{\mathbf{i}}^{\perp}$ is the projected vector for the enhanced item embedding $\hat{\mathbf{i}}$ by the corresponding entity embedding $\mathbf{e}$:

$$\hat{\mathbf{i}}^{\perp} = \hat{\mathbf{i}} - \hat{\mathbf{w}}_p^{\mathrm{T}} \hat{\mathbf{i}} \hat{\mathbf{w}}_p \tag{17}$$

$$\hat{\mathbf{i}} = \mathbf{i} + \mathbf{e}, \ (i, e) \in \mathcal{A} \tag{18}$$

And $\hat{\mathbf{p}}$ and $\hat{\mathbf{w}}_p$ are the translation vector and the projection vector enhanced by those of the corresponding relation embedding according a predefined one-to-one mapping $\mathcal{R} \to \mathcal{P}$. We obtain these two vectors as follows:

$$\hat{\mathbf{p}} = \mathbf{p} + \mathbf{r} \tag{19}$$

$$\hat{\mathbf{w}}_p = \mathbf{w}_p + \mathbf{w}_r \tag{20}$$

Thus, for entities and items, the enhanced item embeddings contain the relational knowledge among items that is complementary to user-item interactions, and improves item recommendation, since the entity embedding $\mathbf{e}$ perserves the structural knowledge in KG. Meanwhile, the entity embedding $\mathbf{e}$ shall be fine tuned by the additional connectivity through users and items during back-propagation. Note that we don't use the combined embeddings for both tasks, because it makes the embeddings of items the same as corresponding entities in two tasks, which actually degrades our model to share embeddings between items and entities.

For relations and preferences, the usage of relations not only offers explicit interpretation of explainability, but also further combines the two tasks more sufficiently at model level. On one hand, through the one-to-one mapping, the meaning of each preference is revealed by the relation label. For instance, the relation *isDirectorOf* reveals a preference to director, or *starring* for a preference to movie stars. On the other hand, many items have no aligned entities due

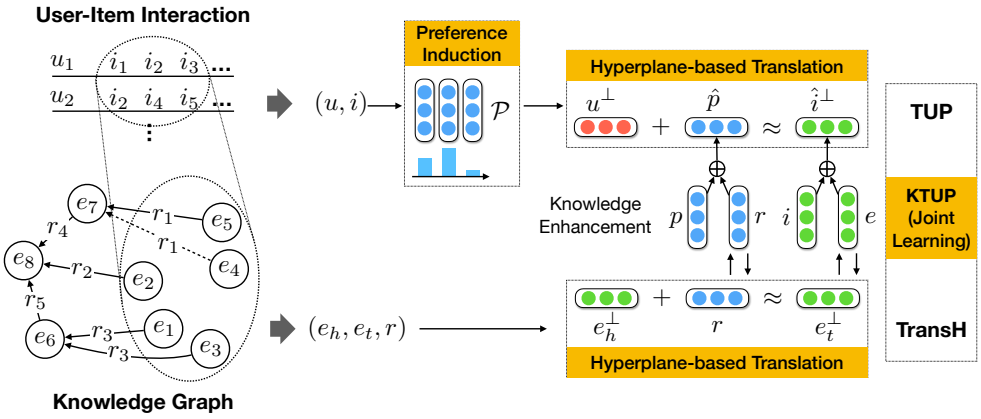


Figure 3: Framwork of KTUP. At the top is TUP for item recommendation including two components: preference induction and hyperplane-based translation. KTUP jointly learns TUP and TransH to enhance the item and preference modeling by transfering knowledge of entities as well as relations.

to the incompleteness of KG, which limits the mutual impacts to the alignments between entities and items in the models that only transfer knowledge of entities. Considering that each user-item pair has a preference and so does the relations between two entities, KTUP optimizes all users, items and entities more thoroughly.

5.2 Training

We train KTUP using the overall objective function as follows:

$$\mathcal{L} = \lambda \mathcal{L}_p + (1 - \lambda)\mathcal{L}_k \tag{21}$$

where λ is a hyperparameter to balance the two tasks.

5.3 Relationship to SOTA Models

In this section, we give a discussion on the relationship between KTUP and the other state-of-the-art KG-based recommendation methods to facilite a deep understanding between two tasks in Section 6. We choose three typical models that transfer knowledge of entities at data level (CFKG [48]), at embedding level (CKE [44]) and in both directions (CoFM [31]). We summerize the main differences and similarities from the following aspects:

Implicity of User Preference CKE and CoFM can be regarded as extentions of collaborative filtering. This type of methods consider the preferences from users to items implicitly and rely on their embeddings to compute a score (i.e., dot product) indicating in which degree the user likes the item. CFKG and KTUP model the preferences explictly and learn the vectoral representations instead of scalars to capture more comprehensive semantics.

Variety of User Preference CFKG defines the only *buy* preference between users and items, which obviously suffers from the serious N-to-N issue and fails to deal with it through the TransE-like scoring function. TKUP distinguishes different user perferences and introduces hyperplanes for each preference as well as each relation to learn the various representations of items and entities.

Transfered Knowledge from KG CKE and CoFM only focus on transfering knowledge of entities. CFKG also transfers relations in the way of data integration through a unifed graph. Except for entities and items, KTUP combines the embeddings of relations and preferences according to the predefined one-to-one mapping, which brings another byproduct of the explainable ability to recommendation mechanism.

6 EXPERIMENTS

In this section, we conduct quantitative experiments on separate tasks of item recommendation and KG completion. For each task, we use two datasets in different domains and give further evaluations on the influence of data sparsity as well as the N-to-N issue. We then investigate the mutual impacts between the two tasks during joint training. Finally, we highlight real examples for qualitative analysis to give an intuitive impression of explainability. We publish our project at https://github.com/TaoMiner/joint-kg-recommender.

6.1 Datasets

Following CoFM [31], we use two publicly available datasets in the movie and book domains: MovieLens-1m [28] and DBbook2014 [2]. Both datasets consist of users and their ratings on movies or books, which are then refined for LODRecSys [15, 28, 29] by mapping items into DBPedia entities if there is a mapping available. Following most item recommendation work that models implicit feedback [40], we treat existing ratings as positive interactions, and generate negative ones by randomly corrupting items.

To collect the related facts from DBPedia, we only consider those triplets that are directly related to the entities with mapped items, no matter which role (i.e. subject or object) the entity serves as. We then preprocess the two datasets by: filtering out low frequency users and items (i.e., lower than 10 in MovieLens and 5 in DBbook), filtering out infrequent entities (i.e., lower than 10 in both datasets), cutting off unrelated relations and merging similar relations manually.

Table 1 shows the statistics of MovieLens-1m and DBbook2014 datasets[3]. After preprocessing, there are 6,040 users and 3,230 items

[2] http://2014.eswc-conferences.org/important-dates/call-RecSys.html

[3] Because we cannot find the released triplets for the two datasets, we collect them as our KG as mentioned above, which leads to a little difference with those papers using the same datasets (e.g., CoFM [31]).

Table 1: Statistics of MovieLens-1m and DBbook2014

		MovieLens-1m	DBbook2014
User-Item Interactions	# Users	6,040	5,576
	# Items	3,240	2,680
	# Ratings	998,539	65,961
	# Avg. ratings	165	12
	Sparsity	94.9%	99.6%
KG	# Entity	14,708	13,882
	# Relation	20	13
	# Triple	434,189	334,511
Multi-Tasks	# Item-Entity Alignments	2,934	2,534
	Coverage	90.6%	94.6%

with 998,539 ratings in Movielens-1m, the average number of ratings per user is 165 and the sparisity rate is 94.9%. The data sparisity issue is more severe in DBbook2014. It consists of 5,576 users and 2,680 items with 65,961 ratings, where the average number of ratings per user is 12 and the sparisity rate reaches up to 99.6%. The triplets used in the two datasets are at the same scale, where the subgraph for MovieLens-1m composes of 434,189 triplets with 14,708 entities and 20 relations, while the subgraph of DBbook has 334,511 triplets with 13,882 entities and 13 relations. Note that the alignments between items and entities for transfering are fewer in MovieLens-1m than that in DBbook2014.

6.2 Baselines

For item recommendation, we compare our proposed models with the following state-of-the-art baselines involving typical similarity-based methods and KG-based methods.

- Typical similarity-based methods: we choose the widely used collaborative filtering models, **FM** [33] and **BPRMF** [34], because they are the foundations of other baselines and also achieve the state-of-the-art performance on many benchmark datasets.
- **CFKG** [48] that integrates the data of two sources and applies TransE on a unifed graph including users, items, entities and relations;
- **CKE** [44] that combines various item embeddings from different sources including TransR on KG;
- **CoFM** [31] that jointly trains FM and TransE by sharing parameters or regularization of aligned items and entities. We mark the two schemes as CoFM (share) and CoFM (reg), respectively.

For KG completion, we choose the typical methods **TransE** [2], **TransH** [42] and **TransR** [23] that are widely used in this field. Also, we evaluate the above KG-based methods even if they have not been done so in the original papers to investigate the impacts of different transfer schemes.

For fair comparison, we carefully reimplement them in our released codes because they did not report the results on the same datasets and we cannot find their released codes. Note that we remove the components of side information modeling like reviews and visual information, since they are not available in the datasets and are out of the scope of this paper.

6.3 Training Details

We construct training set, validation set and test set by randomly spliting the dataset with the ratio of $7:1:2$. For item recommendation, we split the items for each user and ensure at least one item exist in the test set.

For hyperparameters, we apply a grid search on BPRMF and TransE to find the best settings for each task, and use them for all of the other models since they share the basic learning ideas[4]. The learning rate is searched in $\{0.0005, 0.005, 0.001, 0.05, 0.01\}$, the coefficient of L_2 regularization is in $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 0\}$, and the optimization methods include Adaptive Moment Estimation (Adam), Adagrad and SGD. Finally, we set the learning rate as 0.005 and 0.001 for item recommendation and KG completion, respectively, L_2 coefficient is set to 10^{-5} and 0, and the optimization methods is set to Adagrad and Adam. Particularly, for the models involving two tasks, we have tried both sets of parameters, and pick up the latter set of parameters due to its superior performance.

Other hypermeters are empirically set as follows: the batch size is 256, the embedding size is 100 and we perform early stopping strategy on the validation sets.

We predefine the number of preferences in TUP as 20 and 13 for MovieLens-1m and DBbook2014, respectively, which are set according to the relations of the collected triplets. For the models involving two tasks (i.e., CFKG, CKE, CoFM and KTUP), we set the joint hyperparameter λ as 0.5 and 0.7 on the two datasets after searching in $\{0.7, 0.5, 0.3\}$, so as to balance their impacts, and use the pretrained embeddings of the basic model (i.e., BPRMF and TransE).

The main goal in this paper is to investigate the mutual impacts on each task during jointly training, rather than achieving best performance by tuning parameters. Thus, our proposed models as well as baseline methods are trained once for each dataset and evaluate for both tasks of item recommendation and KG completion.

6.4 Item Recommendation

In this section, we evaluate our models as well as the baseline methods on the task of item recommendation. Given a user, we take all items in test sets as candidates and rank them according to the scores computed based on the embeddings of users and items. Thus, the N items ranked at top are the recommended items.

6.4.1 Metrics. We use five evaluation metrics that have been widely used in previous work:

- Precision@N: It is the fraction of the items recommended that are relevant to the user. We compute the mean of all users as the final precision.
- Recall@N: It is the proportion of items relevant to the user that have been successfully recommended. We compute the mean of all users as the final recall.
- F1 score@N: It is the harmonic mean of precision at rank N and recall at rank N.

[4]We have tested other parameters for baselines and find performance drop.

Table 2: Overall performance on Item Recommendation

	MovieLens-1m (@10, %)					DBbook2014 (@10, %)				
	Precision	**Recall**	**F1**	**Hit**	**NDCG**	**Precision**	**Recall**	**F1**	**Hit**	**NDCG**
FM	29.28	11.92	13.81	81.06	59.48	3.44	21.55	5.75	30.15	20.10
BPRMF	30.81	12.95	14.84	83.18	61.02	3.56	22.46	5.96	31.26	21.01
CFKG	29.45	12.49	14.23	82.24	58.97	3.17	19.69	5.30	28.09	19.87
CKE	38.67	16.65	18.94	88.36	67.05	3.92	23.41	6.51	33.18	**27.78**
CoFM (share)	32.08	13.02	15.12	83.30	58.69	3.41	20.78	5.67	29.84	20.92
CoFM (reg)	31.74	12.74	14.87	82.67	58.66	3.32	20.54	5.54	28.96	20.53
TUP (hard)	37.29	17.07	18.98	**89.60**	67.40	3.40	21.11	5.67	29.56	20.19
TUP (soft)	37.00	16.79	18.76	89.47	67.02	3.62	22.81	6.06	31.42	21.54
KTUP (hard)	40.87	17.24	19.79	88.97	69.65	4.04	24.48	6.71	34.49	27.38
KTUP (soft)	**41.03**	**17.25**	**19.82**	89.03	**69.92**	**4.05**	**24.51**	**6.73**	**34.61**	27.62

- Hit ratio@N: It is 1 if any gold items are recommended within the top N items, otherwise 0. We compute the mean of all users as the final hit ratio score.
- nDCG@N: Normalized Discounted Cumulative Gain (nDCG) is a standard measure of ranking quality, considering the graded relevance among positive and negative items within the top N of the ranking list.

6.4.2 Overall Results. Table 2 shows the overall performance of our proposed models as well as the baseline methods, where *hard* and *soft* denote the two preference induction strategies in Section 4.1. We can observe that:

- Our proposed methods perform the best compared with the baseline methods on the two datasets. Particularly, TUP performs competitively to other KG-based models, while it doesn't require any additional information. This is because TUP automatically infers the knowledge of preferences from the user-item interactions, and performs much better especially when the amount of interaction data is sufficient, like MovieLens-1m. By incorporating KG, KTUP further presents more promising improvements on DBbook than MovieLens (i.e., 11.06% v.s. 4.43% gains in F1), which implies that the knowledge is more helpful for sparse data.
- The hard strategy performs better than the soft strategy only when it is used for TUP on MovieLens-1m, which implies that to induce a deterministic user perference requires sufficient data, and the soft strategy is more robust.
- CFKG and CoFM perform slightly better than the typical models (i.e., FM and BPRMF) on MovieLens-1m, but perform worse on the sparse dataset of DBbook2014. One possible reason is that they both transfer entities by forcing their embeddings to be similar with the aligned items, leading to the loss of knowledge that has been perserved in the embeddings, and the loss becomes more serious when there is insufficient training data.
- CKE achieves pretty good performance on the two datasets mainly because it combines the embeddings of items and entities that perserve the information from both sources, instead of aligning them with similar positions in the latent space.
- All models preform much better on MovieLens-1m than on DBbook2014 due to the relatively sufficient training data and an easier test (a higher value even using random initializations). Interestingly, the improvement by utilizing KG is larger on dense dataset of MovieLens than that on the sparse dataset of DBbook. This goes against our intuitions that the more sparse the dataset is, the more potentials it shall have in absorbing richer knowledge. Thus, we further split the test set according to different sparisity levels of training data, and investigate the impacts from KG on each subset in the next section.

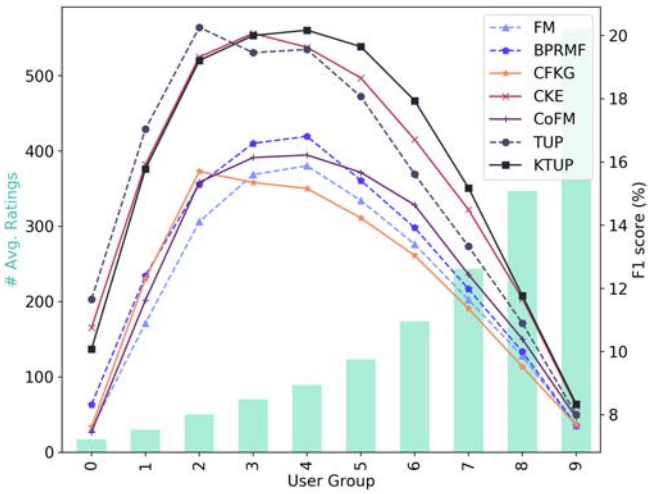


Figure 4: Influence of Different Sparsity on MovieLens-1m. The x-axis shows 10 user groups splited according to interaction number, the left y-axis corresponds to the bars indicating the number of interactions in each user group, and the right y-axis denotes F1-score of curves.

6.4.3 Influence of Training Data Sparsity. To investigate the influence of data sparsity on knowledge transfer, we split the test set of MovieLens-1m into 10 subsets according to the rating number of each user for training; meanwhile we also try to balance the number of users as well as ratings in each subset. The detailed results of F1 score are shown in Figure 4. Green bars represent the average

Table 3: Performance on MovieLens by Relation Category

Task	Prediction Head (Hits@10, %)				Prediction Tail (Hits@10, %)			
Relation Category	**1-to-1**	**1-to-N**	**N-to-1**	**N-to-N**	**1-to-1**	**1-to-N**	**N-to-1**	**N-to-N**
TransE	59.62	56.76	64.55	24.56	**65.38**	62.16	78.52	46.25
TransH	61.54	48.65	65.73	25.51	57.69	78.38	75.62	46.73
TransR	17.31	29.73	32.88	18.50	17.31	43.24	53.12	38.88
CFKG	59.62	51.35	63.31	20.30	57.69	70.27	78.56	41.22
CKE	19.23	21.62	24.16	14.81	7.69	24.32	37.83	34.82
CoFM (share)	65.38	59.46	66.13	24.42	61.54	72.97	**81.05**	45.99
CoFM (reg)	69.23	**70.27**	66.09	24.30	48.08	**86.49**	80.72	45.79
KTUP (hard)	67.31	59.46	66.42	25.67	57.69	81.08	79.22	47.24
KTUP (soft)	**75.00**	56.76	**67.16**	**26.09**	63.46	81.08	78.34	**47.65**

rating number per user ranging from 17 to 563[5]. We denote the models without KG knowledge as dashed lines, and other models as solid lines.

We can see that (1) KG-based methods (i.e., CKE and KTUP) outperform the other models the most when the average number of ratings per user ranges from 100 to 200. (2) The gap between the two types of models is getting closer as the amount of the training data decreases, and the improvements become similar with that on DBbook when their training data is at the similar sparisity level. (3) Meanwhile, the gap almost disappears when the average ratings are 563 (the left most bar), which implies that the impacts of KG become negligible if there if sufficient training data. Note that the performance of all models are getting worse when the average ratings are larger than 89. The possible reason is the user likes so many items that the preferences are too general to capture. (4) TUP outperforms KTUP when user preferences are relative simple to model (i.e., #rating<50), showing the effectiveness and necessity to fully utilize user-item interactions for preference modeling.

Table 4: Overall performance on KG Completion

	MovieLens-1m		DBbook2014	
	Hit@10 (%)	**Mean Rank**	**Hit@10 (%)**	**Mean Rank**
TransE	46.95	537	60.71	531
TransH	47.63	537	60.06	556
TransR	38.93	609	56.33	563
CFKG	41.56	523	58.83	547
CKE	34.37	585	54.66	593
CoFM (share)	46.62	515	57.01	529
CoFM (reg)	46.51	**506**	60.81	521
KTUP (hard)	48.39	525	60.53	501
KTUP (soft)	**48.90**	527	**60.75**	**499**

6.5 Knowledge Graph Completion

In this section, we evaluate on the task of KG completion. It is to predict the missing entity e_h or e_t for a given triplet (e_h, e_t, r). For each missing entity, we take all entities as candidates and rank them according to the scores computed based on entity and relation embeddings.

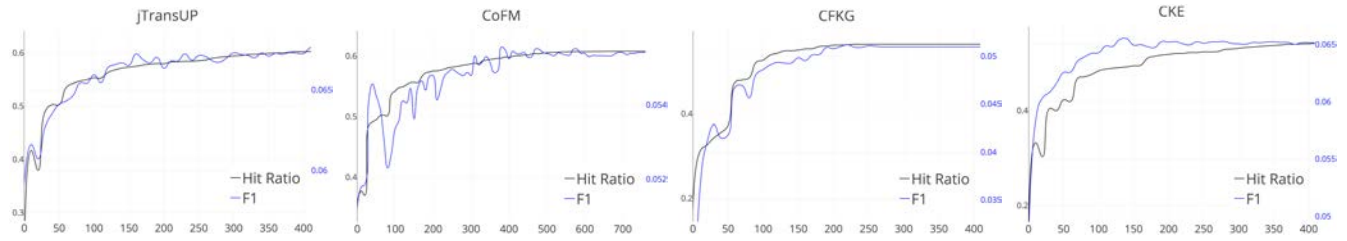
[5]Note that the sparisity of DBBook2014 can be concluded into the 0th user group in MovieLens-1m, in which we observe similar results.

6.5.1 Metrics. We use two evaluation metrics that have been widely used in previous work [42]:

- Hit ratio@N: It is 1 if the miss entity are ranked within the top N candidates, otherwise 0. We compute the mean of all triplets as the final hit ratio score.
- Mean Rank : It is the averaged rank of the missing entities, the smaller the better.

6.5.2 Overall Results. Table 4 shows the overall performance. We can see that KTUP almost outperforms all the other models on both datasets except the mean rank value on MovieLens-1m. We argure this metric is less important since it is easily reduced by an obstinate triple with a low rank [41]. Compared with TransH, it achieves a larger improvement on Hit Ratio for MovieLens-1m as compared to that for DBbook2014 (2.67% v.s. 1.15%), because Movielens-1m contains more connectivities between users and items that are helpful for modeling structural knowledge between entities. We also observe that CFKG, CKE and CoFM show a performance drop as compared to their basic KG components: TransE and TransR. One reason may be that these methods force the embeddings of aligned entities to satisfy the other task of item recommendation, while the aligned entities are only a small portion (i.e. 19.95% and 18.25% on the two datasets), which actually degrades the learning for KG completion. Another reason is that the N-to-N issues of user preferences have negative impacts on the representation learning of entities and relations, especially for the *buy* relation in CFKG. CKE takes this issue into account but TransR contains a lot of trainable parameters and does not work well on such a small training set.

6.5.3 Capability to Handle N-to-N Relations. Table 3 shows the separate evaluation results on each relation category. Following [2], we divide relations into four types: 1-to-1, 1-to-N, N-to-1 and N-to-N. We can see that (1) TransR and its related models (i.e., CKE) perform the worst which is consistent with the above overall performance. (2) KTUP achieves the best performance on N-to-N issues, and also performs competitively with TransE and CoFM on 1-to-1, 1-to-N and N-to-1 problems, which indicates the capability of our methods in hanling complex relations and improves both tasks. (3) CFKG presents a lower value on N-to-N relations than TransE, which implies that the unifed graph may have introduced more confusing relational semantics. (4) CoFM performs competitively in KG completion task while worse in item recommendation, becuase their knowledge transfering schemes lead to unstable joint training.



(a) KTUP ($\rho = 0.97$) (b) CoFM ($\rho = 0.81$) (c) CFKG ($\rho = 0.97$) (d) CKE ($\rho = 0.94$)

Figure 5: Correlation of Training Curves between Two Tasks on DBbook2014, which is denoted by the Pearson's correlation coefficient ρ. The x-axis is training epoch, the left y-axis corresponds to KG completion via hit ratio, and the right y-axis is for item recommendation through F1. (Note that we scale the values of both F1 and Hit Ratio to the same magnitude.)

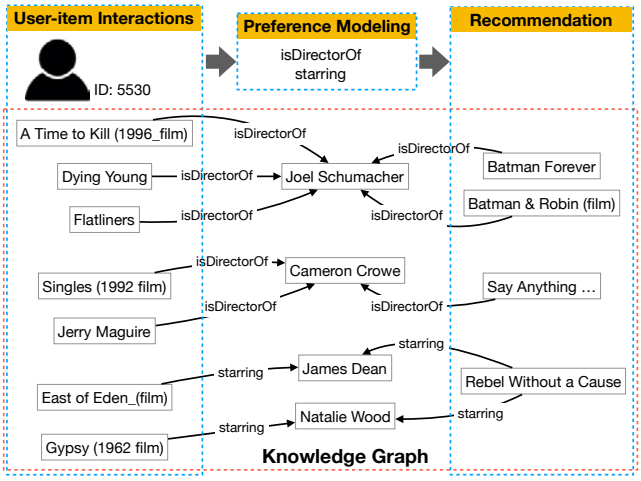


Figure 6: Real Example from MovieLens-1m

That is, it is difficult to control the positive impacts of knowledge transfer on which task, and different parameters for separately training CoFM on each task is required, which is also concluded in the original paper [31].

6.6 Mutual Benefits of Two Tasks

Although the evalutions on separate tasks have been conducted, it is still unclear how different transfer schemes take effect. We thus investigate the correlations between the training curves of two tasks. Intuitively, a strong correlation implies a more complete transfer learning, and a better utilization of the complementary information from each other. Because KG completion has no F1 measures, so we present its hit ratio corresponds to left y-axis, and item recommendation through F1 is presented on right y-axis.

As shown in Figure 5, we can see that KTUP and CFKG present the strongest correlations between their curves, that is, the increase and decrease of one curve shall be reflected on the other curve simultaneously. This implies that the transfer of relations plays an important role in training the two tasks together. However, CFKG does not perform well on both tasks (as shown in Table 2 and Table 4) mainly because of 2 reasons. First, it cannot deal with complex relations; second, it only increase the connectivity in the unified graph through the integration of relations and preferences, which are actually not transitional. Instead, KTUP combines the embeddings of relations and preferences that perserve two types of structural knowledge, and meanwhile introduces hyperplanes for the N-to-N issue. The curves of CoFM and CKE are obviously not correlated strongly due to the small portion of transfered entities. Concretely, CoFM forces the embeddings of aligned entities and items to be similar which may result in unstable training. CKE focuses on unidirectional enhancements by combining their embeddings, and thus performs well in item recommendation but worse in KG completion.

6.7 Case Study

In this section, we present an example of Movielens-1m to give an intuitive impression of our explainability. On the left is a user who has interacted with 7 movies. KTUP first induces user preferences to these movies, and finds that what the user is concerned is the relations of *isDirectorOf* and *starring* (the preferences having highest attention in Section 4.1). Thus, it searches the nearest items according to Equation 16 based on the induced preferences. We present the recommended four movies on the right side. In particular, *Batman Forever* and *Batman & Robin (film)* are recommended because the user shows preference with their director *Joel Schumacher*. Similarly, the preference to director also helps to induce the movie *Say Anything ...* directed by *Cameron Crowe*. Besides, the user also has preferences on *starring*, such as *James Dean* in *East of Eden (film)* and *Natalie Wood* in *Gypsy (1962 film)*; together, the system suggests another movie *Rebel Without a Cause*.

7 CONCLUSION

In this paper, we proposed a novel translation-based recommender model, TUP, and extended it to integrate KG completion seamlessly, namely KTUP. TUP is capable of modeling various implicit relations between users and items which reveal the preferences of users on consuming items. KTUP further enhances the model explainability via aligned relations and perferences, and improves the preformances of both tasks via joint learning.

In future, we are interested in inducing more complex user preferences over multi-hop entity relations and introducing KG reasoning (e.g., rule mining) techniques for unseen user preferences to deal with the cold start problem.

ACKNOWLEDGMENTS

This research is part of NExT research which is supported by the National Research Foundation, Prime Minister's Office, Singapore under its IRC@SG Funding Initiative.

REFERENCES

[1] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning* (2014).
[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*.
[3] Yixin Cao, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Neural Collective Entity Linking. In *COLING*.
[4] Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Chengjiang Li, Xu Chen, and Tiansi Dong. 2018. Joint Representation Learning of Cross-lingual Words and Entities via Attentive Distant Supervision. In *EMNLP*.
[5] Yixin Cao, Lifu Huang, Heng Ji, Xu Chen, and Juanzi Li. 2017. Bridge text and knowledge by learning multi-prototype entity mention embedding. In *ACL*.
[6] Rose Catherine and William Cohen. 2016. Personalized recommendations using knowledge graphs: A probabilistic logic programming approach. In *RecSys*.
[7] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *WSDM*.
[8] Xu Chen, Yongfeng Zhang, Hongteng Xu, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Visually Explainable Recommendation. *arXiv preprint arXiv:1801.10288* (2018).
[9] Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan Kankanhalli. 2018. Aspect-Aware Latent Factor Model: Rating Prediction with Ratings and Reviews. In *WWW*.
[10] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. 2019. Temporal Relational Ranking for Stock Prediction. *ACM Transactions on Information Systems* (2019).
[11] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based recommendation. In *RecSys*.
[12] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *SIGIR*.
[13] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial personalized ranking for recommendation. In *SIGIR*.
[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*.
[15] Benjamin Heitmann. 2012. An open framework for multi-source, cross-domain personalisation with semantic interest graphs. In *RecSys*.
[16] Jin Huang, Wayne Xin Zhao, Hong-Jian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *SIGIR*.
[17] Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. *ICLR* (2017).
[18] Xisen Jin, Wenqiang Lei, Zhaochun Ren, Hongshen Chen, Shangsong Liang, Yihong Zhao, and Dawei Yin. 2018. Explicit State Tracking with Semi-Supervisionfor Neural Dialogue Generation. In *CIKM*.
[19] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* (2009).
[20] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* (2015).
[21] Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying Task-oriented Dialogue Systems with Single Sequence-to-Sequence Architectures. In *ACL*.
[22] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling Relation Paths for Representation Learning of Knowledge Bases. In *EMNLP*.
[23] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion.. In *AAAI*.
[24] Quan Liu, Hui Jiang, Andrew Evdokimov, Zhen-Hua Ling, Xiaodan Zhu, Si Wei, and Yu Hu. 2016. Probabilistic reasoning via deep learning: Neural association models. *arXiv preprint arXiv:1603.07704* (2016).
[25] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*.
[26] Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, et al. 2016. Holographic Embeddings of Knowledge Graphs.. In *AAAI*.
[27] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2012. Factorizing yago: scalable machine learning for linked data. In *WWW*.
[28] Tommaso Di Noia, Vito Claudio Ostuni, Paolo Tomeo, and Eugenio Di Sciascio. 2016. Sprank: Semantic path-based ranking for top-n recommendations using linked open data. *ACM Transactions on Intelligent Systems and Technology (TIST)* (2016).
[29] Guangyuan Piao and John G Breslin. 2017. Factorization machines leveraging lightweight linked open data-enabled features for top-N recommendations. In *WISE*.
[30] Guangyuan Piao and John G. Breslin. 2018. A Study of the Similarities of Entity Embeddings Learned from Different Aspects of a Knowledge Base for Item Recommendations. In *ESWC*.
[31] Guangyuan Piao and John G. Breslin. 2018. Transfer Learning for Item Recommendations and Knowledge Graph Completion in Item Related Domains via a Co-Factorization Model. In *ESWC*.
[32] Tony A Plate. 1995. Holographic reduced representations. *IEEE Transactions on Neural networks* (1995).
[33] Steffen Rendle. 2010. Factorization machines. In *ICDM*.
[34] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*.
[35] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*.
[36] Amit Sharma and Dan Cosley. 2013. Do social explanations work?: studying and modeling the effects of social explanations in recommender systems. In *WWW*.
[37] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent relational metric learning via memory-based attention for collaborative ranking. In *WWW*.
[38] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*.
[39] Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. 2018. Tem: Tree-enhanced embedding model for explainable recommendation. In *WWW*.
[40] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable Reasoning over Knowledge Graphs for Recommendation. In *AAAI*.
[41] Zhigang Wang and Juan-Zi Li. 2016. Text-Enhanced Representation Learning for Knowledge Graph.. In *IJCAI*. 1293–1299.
[42] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes.. In *AAAI*.
[43] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.
[44] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. In *SIGKDD*.
[45] Hanwang Zhang, Zawlin Kyaw, Shih-Fu Chang, and Tat-Seng Chua. 2017. Visual translation embedding network for visual relation detection. In *CVPR*.
[46] Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat-Seng Chua. 2016. Discrete collaborative filtering. In *SIGIR*.
[47] Mengdi Zhang, Tao Huang, Yixin Cao, and Lei Hou. 2015. Target Detection and Knowledge Learning for Domain Restricted Question Answering. In *NLPCC*.
[48] Yongfeng Zhang, Qingyao Ai, Xu Chen, and Pengfei Wang. 2018. Learning over Knowledge-Base Embeddings for Recommendation. In *SIGIR*.
[49] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *SIGIR*.
[50] Zili Zhou, Shaowu Liu, Guandong Xu, Xing Xie, Jun Yin, Yidong Li, and Wu Zhang. 2018. Knowledge-Based Recommendation with Hierarchical Collaborative Embedding. In *PAKDD*.

Enriching News Articles with Related Search Queries

David Carmel*
Amazon Research
dacarmel@amazon.com

Yaroslav Fyodorov
Yahoo Research
yfyodorov@verizonmedia.com

Saar Kuzi†
University of Illinois
skuzi2@illinois.edu

Avihai Mejer
Yahoo Research
amejer@verizonmedia.com

Fiana Raiber
Yahoo Research
fiana@verizonmedia.com

Elad Rainshmidt
Yahoo Research
eladr@verizonmedia.com

ABSTRACT

Enriching the content of news articles with auxiliary resources is a technique often employed by online news services to keep articles up-to-date and thereby increase users' engagement. We address the task of enriching news articles with related search queries, which are extracted from a search engine's query log. Clicking on a recommended query invokes a search session that allows the user to further explore content related to the article. We present a three-phase retrieval framework for query recommendation that incorporates various article-dependent and article-independent relevance signals. Evaluation based on an offline experiment, performed using annotations by professional editors, and a large-scale online experiment, conducted with real users, demonstrates the merits of our approach. In addition, a comprehensive analysis of our online experiment reveals interesting characteristics of the type of queries users tend to click and the nature of their interaction with the resultant search engine results page.

ACM Reference Format:
David Carmel, Yaroslav Fyodorov, Saar Kuzi, Avihai Mejer, Fiana Raiber, and Elad Rainshmidt. 2019. Enriching News Articles with Related Search Queries. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313588

1 INTRODUCTION

Enriching news articles with auxiliary content that is extracted from rich and diverse sources is a popular trend used by Web news services to improve user engagement. Content enrichment solutions provide an opportunity to bring relevant knowledge, meaning, and added value to the content consumed. One such approach exposes the user to additional related news articles [30]. Other popular forms are based on user-generated content (UGC), where users are empowered to comment on an article, or to reply to others' comments, generating lively discussion threads [31]. Relevant tweets and blog posts can further enrich an article's content [20], providing complementary and sometimes controversial points of view.

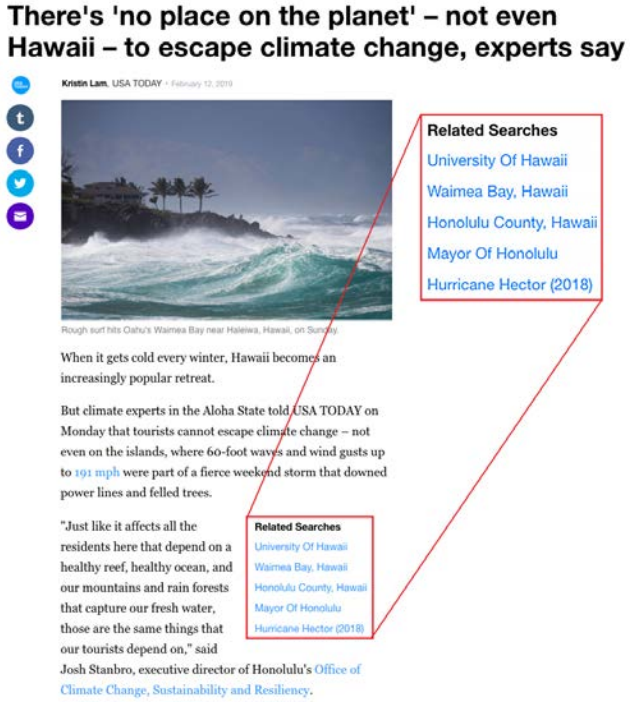


Figure 1: Example of recommended search queries.

These auxiliary sources provide a dynamic and continuously updated content stream, focused on the article's topic. This keeps the article relevant and fresh, thereby deepening users' engagement with the news service.

In this work we address the task of enriching news articles with related search queries that are extracted from a search engine's query log [5]. A search session is invoked when a user clicks on one of the recommended queries, and the subsequent search engine results page (SERP) exposes the user to supplementary content. An example is shown in Figure 1.

There are several requirements that we consider important for the recommended queries to satisfy. First and foremost, the queries should be related to the article's content. We consider the following two notions of query relatedness. The queries might focus on one of the topics covered in the article, allowing the user to further explore that specific topic. Such queries can potentially help users discover different angles of a story. Alternatively, recommended

*Part of this research was conducted while the author was working at Yahoo Research.
†Part of this research was conducted while the author was an intern at Yahoo Research.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313588

queries could expose new and interesting aspects related to the topic, which are not explicitly covered in the article [5]. Given that our task is to enrich news articles, only very fresh and emerging queries are considered relevant. Freshness, our second key requirement, is especially important when dealing with breaking news articles that discuss trending stories [11]. Privacy is another important requirement. Because the recommended queries are extracted from a query log, they must respect the privacy of the users who issued them, without exposing any personal or sensitive information [18]. In addition, queries should respect the article's tone and sentiment. For example, it would be inappropriate to recommend queries related to travel deals to Hawaii for an article dealing with a plane crash in Hawaii. Similarly, a query about buying sneakers for boys is an inappropriate recommendation for an article describing a police search for a missing boy. Like any other UGC, queries extracted from the log might be offensive or malformed [13, 34, 35]. We should avoid presenting such queries to users. Finally, although this is not our main focus, we can prioritize queries that can contribute to the overall monetization of the system while also improving user engagement.

We propose a three-phase retrieval approach to address the task. In the first phase, an initial query list is retrieved in response to the article using a limited set of features. In the second phase, a rich set of features are used to re-rank the list retrieved in the previous phase. The weights of the features are learned using a learning-to-rank approach [23]. We consider both article-dependent features, such as the textual and semantic similarity between the query and the article's title, as well as article-independent features, such as the query length and the number of distinct users who submitted the query. In the third phase, two types of filters are employed. One of the filters is used to improve the quality of the recommendations by removing semantically unrelated queries. The other filter is used to diversify the list of recommended queries by removing similar queries that express the same information need.

We conducted an offline evaluation using a dataset manually created by professional editors, as well as a large-scale online experiment with real users. The results of the two experiments demonstrate the merits of the proposed three-phase retrieval approach. A deeper analysis of our results reveals interesting insights regarding the characteristics of the queries users tend to click as well as the nature of the users' interaction with the resultant SERP . For example, we found that users are more likely to click on longer queries that are expressed as questions. In addition, we found that users who clicked on recommended queries are more likely to explore images and videos, and are less likely to click on organic results ("blue links").

2 RELATED WORK

News articles are often augmented with related social media content, including tweets and blogs [8], encyclopedic knowledge [25], and related search queries [5]. In what follows we cover some of the works most relevant to ours.

Mihalcea et al. [25] introduced the use of Wikipedia as a resource to automatically enrich a text with links to encyclopedic knowledge. Given an input text, important concepts are identified and automatically linked to the corresponding Wikipedia pages. Cheng et al. [9] showed that user queries are often triggered by previously consumed content. The query in such scenarios might be better served by a search engine that uses the originating page as contextual information. The authors used multiple features in a learning-to-rank approach for query recommendation. Some of the features are based on the textual similarity between the query and the page. Others are extracted from a page-query bipartite graph that was constructed using historical query logs. The main limitation of this approach is that it is more effective for popular pages that were visited many times by many different users, making it inherently inapplicable for fresh content such as news articles. Kong et al. [19] explored the reasons users initiate a search. The authors conducted an in-depth analysis using a large-scale query log, studying the relations between previously consumed content and the triggered queries. A feature-based approach is proposed to predict which queries a user is likely to issue after visiting a particular page. The features are computed based on the textual match between the query and the visited page, the user's search history, and the entities appearing on the page.

Bordino et al. [5] addressed a task similar to ours. An article was represented using a set of Wikipedia entities that were extracted from it. A random walk on an entity-query graph, which is based on a long-history query log, was used to identify additional related entities and queries. In a following work, Bordino et al. [6] studied the question of what makes a result serendipitous, i.e., encourages users to explore further. For this purpose, the content of an entity graph was further enriched with metadata about the sentiment, writing quality, and topical category of the text surrounding an entity. The authors investigated whether entities that encompass more emotion provide better results. Their findings show that such entities are not enough to draw the user's attention, suggesting a more complex relationship between topics and entities in serendipitous search. While the entity-query graph provides a very appealing framework for content enrichment with related queries, it does not suit our news-article enrichment task, since only very fresh and emerging queries related to the article are considered relevant in our case. Indeed, our experiments show that queries with decreasing popularity are less suitable for recommendation in our case. On the other hand, recommending serendipitous queries for content enrichment might be very appealing in our setting. We leave this direction for future work.

Another related direction is automatic question generation from some form of input [14, 29]. Such techniques focus primarily on generating questions that can be answered by the given text, while our focus is on queries that enhance and elaborate the article's content. We note, however, that such generation techniques, modified for content enhancement, might be a suitable complementary approach that deserves further investigation. Synthetic query generation is another complementary approach [22]. Queries in the work by Lee et al. were generated from text passages selected by the user rather than identified in a query log. No emphasis was put on freshness.

3 RETRIEVAL FRAMEWORK

Given an article r and a corpus of queries Q, our goal is to produce a highly precise ranking of queries that fulfill several requirements.

The queries should: be relevant to the article, address recent events, disclose nothing about the user who issued them, respect the tone of the article, be non-offensive, and be possibly monetizable. To this end, we take a three-phase retrieval approach. In the first phase, an initial query list Q_{init} is retrieved in response to the article using a limited set of features. In the second phase, a rich set of features is used for re-ranking the n most highly ranked queries in Q_{init}. Additional refining filters are applied in the third phase and the top-k ($k < n$) recommendations are presented to the user. We now present further details for the three retrieval phases. Details regarding the creation of the query corpus are provided in Section 4.1.1.

3.1 First Phase: Initial Ranking

We represent each query $q \in Q$ using the query's raw text (q_T), a concatenation of the search result titles previously retrieved in different searches in which the same query was issued (q_R), and the entities identified in the query's text (q_E). Similarly, an article r is represented using its title (r_T); body (r_B); summary (r_S), which contains a subset of the top-θ terms selected from the article's body according to their TF.IDF values; and the set of entities identified in the article's title and body (r_E). Additional details about the entity-linking tools used are provided in Section 4.

The initial query retrieval is based on the similarity between the different texts representing the article and query, as well as a set F of article-independent query-quality measures:

$$Score_{\text{Init}}(q, r) \stackrel{def}{=} \sum_{X \in \{T,R\}; Y \in \{T,S\}} \lambda_{XY} Sim(q_X, r_Y) + \lambda_E Sim(q_E, r_E) + \sum_{f \in F} \lambda_f f(q); \quad (1)$$

$Sim(\cdot,\cdot)$ is an inter-text similarity measure described below; λ_{XY}, λ_E and λ_f are free parameters; $f(q)$ is the score assigned to q by a quality measure f. Inspired by the state-of-the-art performance of the Sequential Dependence approach in the Markov Random Field model [24], to compute the similarity between texts x and y, we use

$$Sim(x, y) \stackrel{def}{=} \lambda_T Sim_{\text{T}}(x, y) + \lambda_U Sim_{\text{U}}(x, y), \quad (2)$$

where $Sim_{\text{T}}(x, y)$ and $Sim_{\text{U}}(x, y)$ are the similarity values, computed using BM25 [28], between terms and unordered term-pairs (within a 5-term window), respectively, in x and y; λ_T and λ_U are free parameters.

We use four measures $f \in F$ to quantify different article-independent properties of the query. Popular queries that were issued by many users might be more interesting. Hence, we count the number of distinct users who issued the query during the last three weeks (**DistinctUsers**). We also examine the interaction of users with the SERP retrieved in response to the query: rich interaction with the SERP is a potential indicator of how attractive and informative it is, and hence of the high query quality. We compute the average number of cards[1] [32] on the page (**Cards**), the average number of clicked URLs after the query was issued (**ClicksPerUser**), and the clickthrough rate (CTR) of ads presented on the SERP (**AdsCTR**). Additional measures are integrated via our learning-to-rank approach in the second retrieval phase, as detailed in the next section.

[1]Cards are results presented directly on the SERP to spare a user an additional click [32].

Table 1: Features used in the learning-to-rank algorithm.

	Feature	Description
Article Dependent	EntitySim	$Sim(q_E, r_E)$
	TitleSim	$Sim(q_X, r_T)$ for $X \in \{T, R\}$
	SummarySim	$Sim(q_X, r_S)$ for $X \in \{T, R\}$
	BodySim	$Sim(q_X, r_B)$ for $X \in \{T, R\}$
	Overlap	Jaccard similarity between q_T and r_T
	W2V	Semantic similarity using word2vec
	ESA	Semantic similarity over Wikipedia concepts
Article Independent	QueryLength	Number of query terms
	AvgIDF , MaxIDF	Average and maximum IDF of query terms
	DistinctUsers	Number of distinct users that submitted the query
	RPM	Revenue per mille (thousand query submissions)
	AdsCTR	Clickthrough rate of ads presented on the SERP retrieved for the query
	ClicksPerUser	Number of clicked URLs divided by the number of users that issued the query
	DistinctClicked	Distinct number of clicked URLs
	DecDistinctUsers	Decaying sum of the number of users that submitted the query during the last three weeks
	AvgUsers , StdvUsers	Average and standard deviation of the number of users that submitted the query during the last three weeks
	AvgUserHistory	Average number of users who issued the query in a specific time period during the last three weeks

3.2 Second Phase: Learning to Rank

We re-rank Q_{init}, the initial query list retrieved in the first phase, to improve its quality. The re-ranking model was trained using a learning-to-rank (**LTR**) approach [23], details of which are provided in Section 4.1.4. Each pair of an article and query is represented as a feature vector. As in Equation 1, we consider two types of features: article dependent and article independent. The features[2] are listed in Table 1.

3.2.1 *Article-Dependent Features.*

Textual Similarities. We use several textual similarity estimates between the different texts representing the article and query. One group of features, which we also used in the initial retrieval phase, includes the BM25 similarity between the entities identified in the article and query (**EntitySim**). We use two entity-linking tools to identify entities in texts (see Section 4 for details). Hence, considering all possible combinations of the entity sets extracted by the two tools, we get four features. Another group of features includes the similarities between the query and the article's title (**TitleSim**), body (**BodySim**), and summary (**SummarySim**), where the query is represented using either its raw text or by a concatenation of its result titles. While a single similarity estimate (BM25) was

[2]We applied Z-score normalization over feature values, which we found to be more effective than applying Min-Max normalization or not normalizing at all.

used to instantiate Equation 2 in the first retrieval phase, here we use three estimates: TF.IDF , BM25 [28], and LM [21] (language models with Jelinek-Mercer smoothing). We get 18 features: 3 similarity estimates (TF.IDF , BM25 , and LM) × 3 texts representing the article (r_T, r_S, and r_B) × 2 texts representing the query (q_T and q_R). In addition, we compute the Jaccard similarity between the query's raw text and the article's title (**Overlap**), where texts are represented by stemmed unigrams after stopword removal.

Semantic Similarities. We use two semantic similarity measures to address the potential vocabulary mismatch between the query and article, which cannot be captured by the textual similarity measures defined above. The first is based on Explicit Semantic Analysis (**ESA**) [12], in which the query's text (q_T) and the article's title (r_T) are embedded into a Wikipedia concept space. Specifically, each text is represented as a weight vector over all Wikipedia concepts (roughly 6M concepts), where the weight of a concept is the similarity value (computed using BM25) between the text and the concept's Wikipedia page. The second measure is based on a word2vec (**W2V**) representation [26]. We train a W2V model using Wikipedia, and represent each text using the centroid of its terms' W2V vectors. We use cosine similarity to compare the query and corresponding article representations for both ESA and W2V . While ESA captures semantic relatedness through a comparison over a concept space, W2V captures the same notion through a comparison of the context of the two texts. Using more complicated entity embedding techniques is an interesting future direction we intend to explore [4].

3.2.2 Article-Independent Features. As already noted, we use multiple article-independent measures to quantify different properties of the query. The first group of measures are computed based on the query's text: query length (**QueryLength**), and average (**AvgIDF**) and maximum (**MaxIDF**) IDF of the query terms. We assume that longer queries with higher IDF values are more interesting. The DistinctUsers measure, which was also used in the first retrieval phase, quantifies query popularity. In addition, high-revenue queries might be favored not only to increase the system's revenue, but also because they are likely to be popular. Hence, we also consider the revenue per thousand query submissions on resultant SERP s (**RPM**). To capture the informativeness and attractiveness of the resultant SERP , we compute several features based on the interaction of users with the SERP: the CTR on Ads (AdsCTR), the average number of clicks performed per user (ClicksPerUser), and the distinct number of clicked URLs (**DistinctClicked**). Finally, we use several features to quantify the trendiness of a query. Queries that were issued only a few times in the past but became popular lately might be more interesting than those whose popularity has been decreasing over time. Hence, we examine the submission history of the query per hour and compute the average (**AvgUsers**) and standard deviation (**StdvUsers**) of the number of distinct users who submitted the query during a one-hour sliding time window over the past three weeks. We also compute a weighted sum of the number of users who submitted the query, where the value in each hour is multiplied by a decaying factor (**DecDistinctUsers**). We experimented with several decaying factors and ended with the following three: Light (0.996), Moderate (0.993), and Heavy (0.986). These provided different decay patterns in user interests for a large portion of the queries. The higher the decaying factor, the lower the difference between weights assigned to different hours. For example, the weight associated with the number of distinct users who issued the query (a day, a week, two weeks) ago will be (0.908, 0.510, 0.260), (0.845, 0.307, 0.094), and (0.713, 0.094, 0.009), for the Light , Moderate , and Heavy decaying factors, respectively. In addition, we divide the three-week hourly submission history into five equal time periods and compute the average number of distinct users who submitted the query in each period (**AvgUserHistory**), resulting in five features.

3.3 Third Phase: Query Filtering

We apply two types of filters on the query ranking produced in the second phase. The first is used to further improve the quality of the recommended queries. The filter is based on the ESA and W2V representations of the query's text and article's title. We filter out queries for which the cosine similarity between their ESA or W2V representation and the corresponding representation of the article is below a given threshold, τ_{ESA} or τ_{W2V}, respectively.

The second filter is used to diversify the list of queries presented to the user by eliminating highly similar queries that express the same information need. To this end, we scan the query ranking from top to bottom. If the similarity between the next query in the ranking and all the preceding queries is below a given threshold τ_{JAC}, the query is considered novel and is added to the final result list; otherwise, it is ignored and the next query in the list is examined. The similarity between queries is measured using Jaccard, where each query is represented via the union of its unigrams and bigrams.

4 EVALUATION

4.1 Experimental Setting

4.1.1 Query Extraction and Indexing. To guarantee that all the articles will be associated with the latest queries, we update the queries dataset (and index) on a regular basis. This is especially important for breaking news articles, where relevant queries start to appear very quickly after the event occurs. New queries that hit the log of the Yahoo commercial Web search engine are identified every four hours and added to the dataset. To ensure that private information about the users who issued these queries will not be revealed, we only consider queries that were searched at least 10 times (by different users) in a day and at least twice during the last 4 hours. Additionally, we filter out offensive and adult queries using blacklists manually curated by in-house professional editors. Finally, we filter out non-Latin queries and queries longer than 10 terms.

For each instance of a query submission, we collect the query's raw text, the top-10 results ("blue links") presented on the SERP and clicks on these results, the displayed ads and the clicks on these ads, and the cards (direct answers) presented on the SERP . We then combine the queries based on their raw text (after lower-casing, but before stemming) over a history of three weeks. Hence, each entry in our query dataset contains a unique query with its top-100 aggregated search result titles scored by their frequency, the number of distinct users who submitted the query, the clickthrough rate for ads, revenue per 1000 submissions (RPM), and the submission history per hour during the last 3 weeks. The 3-week restriction

Table 2: Examples of article titles and queries labeled relevant ('+') and non-relevant ('−') by the editors.

Teachers in Philadelphia Plan a Black Lives Matter Week
+ Black Lives Matter Movement
− Lesson Plans for Teachers
Oregon Man Finds Year-Old Lottery Ticket and Wins $1 Million 8 Days Before it Expired
+ Oregon Lottery Winning Numbers
− Million Man March
3 Colors You Should Always Wear in a Job Interview
+ What to Say in an Interview
− Wear TV 3

ensures that only relatively fresh queries will be served. Our query dataset, on average, contains approximately 500K queries.

4.1.2 Offline Evaluation. Since our query dataset is constantly updated, we fixed a dataset of 411,646 queries collected on April 25, 2017, and randomly sampled a set of 3082 news articles published on this date on Yahoo News. By varying the free-parameter values in Equation 1, we collected a large set of queries to be judged by in-house professional editors. The results were pooled together and up to 30 queries were selected per article, based on their aggregated score across the different free-parameter variations. Binary relevance labels were assigned by the editors; each query was annotated by one editor. The editors were instructed to label queries that satisfied all the requirements discussed in Section 1 as relevant. Table 2 lists a few examples of queries labeled relevant and non-relevant by the editors. We randomly sampled 250 articles for testing. Two thirds of the remaining articles were used to train the learning-to-rank model and one third to set parameter values.

We are mostly focused on high precision. Hence, to estimate the effectiveness of our approach, we used the precision of the top-3 (p@3) and top-5 (p@5) results, the mean average precision of the top-5 results (MAP), and the mean reciprocal rank of the top-5 results (MRR). In addition to the above-mentioned metrics, we also study in Section 4.2.2 the effectiveness of our approach in terms of the overall expected system revenue. Statistically significant differences in performance were determined using the two-tailed paired t-test at a 95% confidence level.

4.1.3 Online Evaluation. In addition to the offline evaluation performed using the dataset described above, we also conducted an online experiment involving two parts. Our goal in the first part was to collect data for training a learning-to-rank model using explicit user feedback rather than editorial judgments. During a period of 40 days (December 22, 2017 until January 30, 2018) our recommended queries were presented to all Yahoo News users. The users were from the United States and used only desktop devices. The initial ranking (first retrieval phase) and the filters (third retrieval phase) were applied with the free-parameter values learned using our offline dataset. The learning-to-rank approach was not employed in this part of the experiment. After applying the two filters, the retrieval score assigned to queries in the first phase was linearly interpolated with the query's RPM value using a free parameter ρ; we set $\rho = 0.3$ following an analysis performed using the offline dataset, as shown in Section 4.2. This linear interpolation allowed us to control the balance between query relevance and system revenue.

The collected feedback was used to train two learning-to-rank models so as to overcome the bias incurred in users' clicking decisions.[3] (Additional analysis about the position bias is provided in Section 4.2.8.) The first, henceforth **Skip** , was learned using an approach similar to "Click > Skip Above" [17]. Specifically, the training data consisted of all the searches (approximately 35K) in which a "skip" occurred: a user clicked on a query in rank $i > 1$, and skipped (did not click on) the queries ranked above it. All the queries ranked above rank i, and the unclicked queries ranked below it, were considered non-relevant. The query in rank i and all the clicked queries ranked below it were considered relevant. The second model, **Balanced** , was trained using the data collected to train the Skip model as well as searches in which the highest ranked query was clicked. For these searches, all the clicked queries were considered relevant, while the non-clicked were deemed non-relevant. We note that the number of searches in which a user clicked on the highest ranked query was much higher than those in which a skip occurred. Hence, we randomly subsampled the data to get a balanced training set.

In the second part of the experiment, we studied the effectiveness of our approach via A/B testing [15]. We simultaneously launched five buckets each of which served 5% of Yahoo News traffic; users were randomly split between buckets. This experiment lasted 10 days. One of the buckets corresponded to the model used in the first part of the experiment (**Control**). Two additional buckets corresponded to the two models described above. Finally, as was the case in the first part of our experiment, we also linearly interpolated the retrieval score of a query with its RPM value ($\rho = 0.3$), which resulted in two additional buckets: **Skip+RPM** and **Balanced+RPM** .

Since our main focus is on improving user engagement, we use the clickthrough rate on the recommended queries (CTR) to estimate the effectiveness of our approach. In addition, since any click on one of the recommended queries initiates a standard search, we also report the total system revenue per thousand sessions that were initiated by such queries (Revenue) and the query reformulation rate (Reformulation) on the SERP . Low reformulation rate presumably implies that the user was satisfied with the results without having to refine the query, attesting to the effectiveness of the recommended query. We report the improvement in percentages with respect to the Control model.

4.1.4 Additional Implementation Details. We used Apache Lucene[4] for experiments. We applied porter stemming and removed stopwords using Lucene's English stopword list. We used an online variant of RankSVM with a linear kernel [16] to train the re-ranking model. The algorithm uses pairwise training examples and iteratively updates the model via the AROW update step [10]. The hyper-parameters were set to optimize p@5 over the validation set. To identify entities in a text, we applied two in-house entity-linking tools. Each entity was represented using a Wikipedia entity ID

[3] Two thirds of the data was used to train the learning-to-rank models and one third to set parameter values.
[4] http://www.lucene.apache.org

Table 3: Offline results. 'i' and 'l' mark statistically significant differences with Init and Init+LTR+ESA+W2V , respectively. The best result in a column is underlined.

	p@3	p@5	MAP	MRR
Init	44.3_{l}	44.6_{l}	58.0_{l}	62.3_{l}
Init+ESA	57.5^{i}_{l}	55.1^{i}	73.0^{i}_{l}	76.9^{i}_{l}
Init+W2V	55.9^{i}_{l}	52.4^{i}_{l}	71.2^{i}_{l}	73.8^{i}_{l}
Init+ESA+W2V	60.3^{i}	$\underline{57.0}^{i}$	76.2^{i}_{l}	78.3^{i}_{l}
Init+LTR	54.4^{i}_{l}	45.8_{l}	71.5^{i}_{l}	74.8^{i}_{l}
Init+LTR+ESA	61.9^{i}	52.4^{i}_{l}	77.4^{i}	80.3^{i}
Init+LTR+W2V	59.1^{i}_{l}	51.4^{i}_{l}	76.9^{i}_{l}	79.6^{i}_{l}
Init+LTR+ESA+W2V	$\underline{62.5}^{i}$	55.8^{i}	$\underline{78.6}^{i}$	$\underline{80.8}^{i}$

along with a score reflecting the confidence of the tool. To compute $Sim(q_E, r_E)$, the IDF of each entity was scaled using the confidence score assigned to it by the entity-linking tool [27].

We mapped the article-independent feature values into a five-dimensional binary feature vector by dividing the feature's range into equal-sized bins. The value in a bin equals one if the feature's value falls in that bin's interval. This binary transformation resulted in better performance compared to using the raw feature values. We applied a logarithmic transformation to the AdsCTR , ClicksPerUser and DistinctClicked features since the ranges of these features were rather large; binary transformation was not applied in this case.

Free-Parameter Values. As mentioned above, free-parameter values were set using a validation set in the online and offline experiments. The number of queries initially retrieved using Equation 1 per article was $n = 25$; $k = 5$ queries were presented to users. The number of terms used to create the article's summary was $\theta = 10$. In Equation 1, we set $\lambda_{TT} = \lambda_{TS} = 10$, $\lambda_{RT} = \lambda_{RS} = 3.5$, $\lambda_E = 2.5$ and $\lambda_f = 0.25$. In Equation 2, we set $\lambda_T = 0.8$ and $\lambda_U = 0.2$. The article's summary (r_S) and the entities identified in the query (q_E) and article (r_E) are unordered sets of tokens, hence when computing the similarity with these texts we set $\lambda_T = 1$ and $\lambda_U = 0$. The free parameters of BM25 ($k1$ and b) and the Jelinek-Mercer smoothing parameter were set to standard values: 1.2, 0.75 and 0.1, respectively. We used W2V 's Continuous Bag-of-Words (CBOW) model. The dimension of the vectors and the window size were set to 300 and 5, respectively; all other parameters were set to default values. In the ESA and W2V filters, we set $\tau_{ESA} = 0.15$ and $\tau_{W2V} = 0.6$. To filter out similar queries, we set $\tau_{JAC} = 0.66$.

4.2 Experimental Results

4.2.1 Offline Results. The results of our offline evaluation are presented in Table 3. We study the effectiveness of the query rankings produced in the first (Init), second (Init+LTR), and third (Init+LTR+ESA+W2V) retrieval phases. We also examine the contribution of the LTR approach and the semantic filters (ESA and W2V) when integrated in different stages of our framework. Note that the similar-queries filter, presented in Section 3.3, was applied to all query rankings. We can see that re-ranking the query list retrieved in the first phase using our LTR model yields statistically significant performance improvements over the initial ranking (Init+LTR vs. Init). Applying the semantic filters (either after the first retrieval phase or after the second phase) substantially improves the performance in both cases. The ESA filter is somewhat more effective than W2V ; however, integrating the two filters leads to further improvements compared to using each alone. Interestingly, Init+ESA+W2V outperforms Init+LTR , further attesting to the clear merits of applying the semantic filters. The best results are almost always attained for Init+LTR+ESA+W2V when considering all three retrieval phases; Init+LTR+ESA+W2V is outperformed only by Init+ESA+W2V for p@5 , but the difference is not statistically significant.

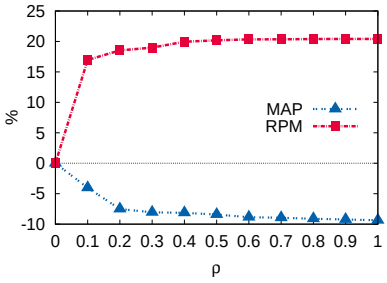


Figure 2: The effect of ρ on MAP and average RPM of the top-five results when interpolating the retrieval score of a query with its RPM value. We report the relative improvement with respect to using the original model Init+LTR+ESA+W2V ($\rho = 0$).

4.2.2 Relevance versus Revenue. After a user clicks on a recommended query, she is taken to a regular SERP where a variety of ads and sponsored links may appear next to organic search results. Such commercial sources of information can increase the overall revenue of the system. Similar to past work on sponsored search [36], we study the tradeoff between recommending queries that are intended to increase user engagement and those aimed at maximizing system revenue. To facilitate the promotion of high-revenue queries, we linearly interpolated the final retrieval score of a query after the third retrieval phase with its RPM value using a free parameter ρ. Figure 2 presents the effect of ρ on MAP and average RPM of the top-five results; the percentages indicate the improvement with respect to using only the original score, i.e., employing the Init+LTR+ESA+W2V model. As ρ increases and more weight is given to the query's RPM value, MAP decreases and the system's expected revenue (RPM) increases. The steepest drop in performance is observed for $\rho \leq 0.3$. For $\rho > 0.3$, MAP continues to drop at a much slower pace.

In Table 4 we show examples of queries recommended by our algorithm for $\rho = 0$, where no weight was given to the query's RPM value, and for $\rho = 0.3$, where a moderate weight was assigned.[5] We see that the queries recommended for $\rho = 0.3$ have a stronger commercial intent. These queries include keywords that users are more likely to use before making a purchase. Conversely, the queries recommended for $\rho = 0$ have a stronger news intent and tend to be more informative. For example, based on a random sample of 280 articles, we found that the terms "sale", "tickets" and "buy" were

[5]We note that in both example articles, the queries recommended for $\rho > 0.3$ were identical to those recommended for $\rho = 0.3$, but they were not necessarily ranked in the same order.

Table 4: Examples of article titles and the top-three recommended queries for different values of ρ. The higher the value of ρ, the more weight is given to the query's RPM value.

	iPhone 9 and X Plus? New iPhone Rumours, Release Dates and Everything We Know
$\rho = 0$	1. iPhone 11 Release Date 2. iPhone Store Near Me 3. iPhone Battery Drains Fast
$\rho = 0.3$	1. Can iPhones Get Viruses 2. iPhone 8 Plus Price 3. Used iPhones for Sale
	Sacred Riana's Weird 'America's Got Talent' Magic Act
$\rho = 0$	1. America Got Talent 2018 2. Courtney America Got Talent 3. America Got Talent Judges
$\rho = 0.3$	1. America Got Talent Tickets 2. NBC America Got Talent 3. America Got Talent 2018

more popular for $\rho = 0.3$. These terms appeared in 6.1%, 5.4% and 4.6% of the recommended queries, respectively, for $\rho = 0.3$ vs. 1.8%, 0% and 1.4% for $\rho = 0$. In contrast, the terms "news", "today" and "twitter" were more popular for $\rho = 0$: they appeared in 9.3%, 5.7% and 2.1% of the queries for $\rho = 0$ vs. 5%, 2.1% and 0% for $\rho = 0.3$.

4.2.3 Online Results. The results of the online evaluation are presented in Table 5. We inspect three metrics quantifying different aspects of our approach: CTR , the user's engagement; Revenue , the system's revenue; and Reformulation , a proxy for the user's dissatisfaction with the search results. Supporting our offline results, we observe substantial improvements in CTR when applying the LTR model in all four buckets; the highest improvement is attained for the Balanced model. Examining Revenue , we see a decrease in revenue for both the Skip and Skip+RPM models, even though the latter explicitly incorporates RPM into the retrieval process. Comparing Balanced with Balanced+RPM , we observe a trend similar to the one we saw in our offline experiment: the CTR is higher for the Balanced model, which uses RPM as a feature in the learning-to-rank model, whereas Revenue is higher for Balanced+RPM , which also explicitly interpolates the RPM value of a query with its retrieval score. The best performance is attained by the Balanced model, which had the highest CTR , second highest Revenue , and lowest Reformulation . Table 6 lists a few examples of queries recommended by this model.

Figure 3 illustrates the change in CTR during the 10-day experiment. As was the case thus far, we report the relative improvement in CTR with respect to the Control model. We first observe that among the 4 considered models, Balanced was the most stable throughout the experiment, especially during the last 6 days. We also see that Balanced is the only model for which the change in CTR was always positive. In contrast, for the other three models, the CTR decreased compared to the Control model at least once during the experiment. These findings, along with the results presented in Table 5, attest to the effectiveness of the Balanced model.

Table 5: Online results. The relative improvement (in percentages) over the Control model. The best result in a row is underlined.

	Skip	Balanced	Skip+RPM	Balanced+RPM
CTR	18.3	<u>22.6</u>	6.5	16.1
Revenue	−23.6	17.4	−23.5	<u>22.4</u>
Reformulation	14.4	<u>−13.5</u>	−7.0	0.6

Table 6: Examples of article titles and the top-three recommended queries by the Balanced model.

Hawaii Officials Plead for Visitors to Keep Travel Plans 1. Is it Safe to Travel to Hawaii Right Now 2. Hawaii Travel Advisory 3. Travel to Hawaii
Prince Harry and Meghan Markle ask U.S. Bishop to Deliver Wedding Address 1. How Old is Meghan Markle and Prince Harry 2. Prince Harry and Meghan Markle Wedding Date 3. How did Meghan Markle Meet Prince Harry
Sears is Closing 40 Stores in 24 States – Here's the List 1. Sears Store Closing List 2018 2. Sears Store Near Me 3. Sears Store Locations

Table 7: Features used in the Balanced model ranked in descending order of attributed importance.

1 − 5	6 − 10	11 − 15	16 − 19
RPM	AvgUsers	DistinctUsers	DistinctClicked
BodySim	QueryLength	EntitySim	MaxIDF
TitleSim	ClicksPerUser	W2V	AdsCTR
Overlap	DecDistinctUsers	AvgUserHistory	ESA
SummarySim	StdvUsers	AvgIDF	

4.2.4 Feature Analysis. We next analyze the contribution of the features and feature groups used in the learning-to-rank approach employed in the second retrieval phase. To this end, we arrange the features in descending order of their weights in learning-to-rank model and assign each feature with a score that is the reciprocal of its rank. For example, the scores of the top-three features are 1, $\frac{1}{2}$ and $\frac{1}{3}$, respectively. These feature scores are then averaged over groups of features as follows. For each of the article-independent features, excluding AdsCTR , ClicksPerUser , and DistinctClicked , we average the scores of their corresponding five binary features (refer back to Section 4.1.4). In addition, for DecDistinctUsers , the scores are also averaged over the three decaying factors and for AvgUserHistory over the five time periods. For the article-dependent features, TitleSim , SummarySim , and BodySim , we average the scores over the three similarity measures (BM25 , TF.IDF and LM)

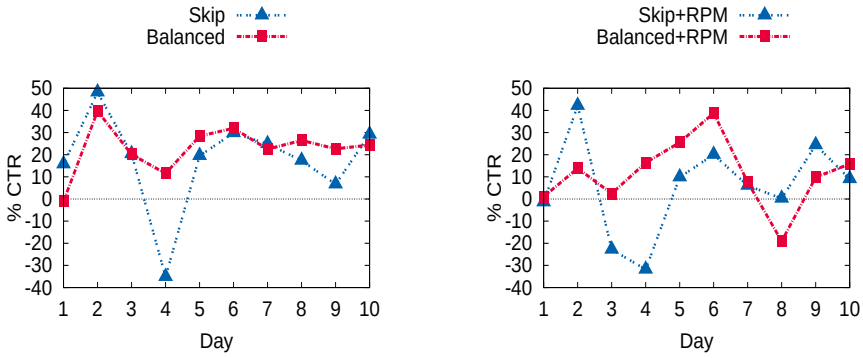


Figure 3: Relative improvement in CTR over time.

and the two texts representing the query (q_T and q_R). Finally, for EntitySim , we average the scores over the four features corresponding to the different combinations of the entity-linking tools.

Table 7 presents the features in descending order of importance for the Balanced model, which is the best performing model among the four we examined. We see that among the top-five features, four (TitleSim , SummarySim , BodySim and Overlap) are estimates of the textual similarity between the article and query, which attests to the importance of the textual similarity features in our task. The semantic similarity features (ESA and W2V), on the other hand, are ranked much lower. This may be because these features are also used to filter out queries in the third retrieval phase, resulting in some redundancy. RPM is the most important article-independent feature, which might explain the lift in Revenue we observed in Table 5 for the Balanced model. AdsCTR is among the least important features. Because this feature was also used in the first retrieval phase, it might be somewhat redundant in the second phase. The AvgUsers , DecDistinctUsers , and StdvUsers features, which quantify the trendiness of the query, are among the top-10, supporting our postulation about the importance of presenting users with very fresh and popular queries. Finally, inspecting the textual similarity features, we found that among the 3 similarity estimates, BM25 , LM , and TF.IDF , BM25 is the most important while LM is the least important. In addition, we found that features measuring the similarity with the query's text (q_T) are less important than those considering the concatenation of the result titles retrieved in response to the query (q_R).

4.2.5 Query Characteristics. To get a better sense of the type of queries users tend to click, we examined the characteristics of all the queries recommended by the five models tested in the second part of our online experiment. We distinguish between clicked and non-clicked queries, where the values extracted per each clicked (non-clicked) query are scaled by the number of times the query was presented and clicked (non-clicked). We performed a similar analysis for queries recommended by each of our five models separately. Here, we scaled the values attained per query by the number of times the query was presented to the users in the corresponding bucket. We consider the following five attributes: (i) Questions : percentage of queries formulated as questions[6]; (ii) Length1 : query length in words; (iii) Length2 : query length excluding stopwords of all queries except those that begin with a question word (iv) SW1 : percentage of stopwords in a stopword list that appear in the query [3]; and, (v) SW2 : percentage of terms in the query that are stopwords[7] [3].

Table 8: Characteristics of queries. Questions : percentage of queries starting with a question word. Length1 : query length in words. Length2 : query length excluding stopwords and queries starting with a question word. SW1 and SW2 : stopword-based measures [3].

	Questions	Length1	Length2	SW1	SW2
Clicked	5.37	4.39	4.00	1.08	6.55
Not Clicked	4.95	3.87	3.56	0.81	5.53
Control	3.47	3.35	3.12	0.60	4.18
Skip	5.93	4.10	3.77	0.89	5.84
Balanced	4.65	3.95	3.64	0.85	5.77
Skip+RPM	6.00	3.97	3.64	0.88	6.06
Balanced+RPM	4.65	3.93	3.62	0.83	5.73

Table 8 presents the results. Comparing clicked versus non-clicked queries, we see that users click on longer queries, which are more likely to contain stopwords and start with a question word. This implies that users prefer queries expressed in a natural language as opposed to those solely containing important keywords, e.g., "how old is prince harry" vs. "prince harry age". The queries recommended by the Control model, for which the lowest CTR was attained, are shorter than those recommended by the other models and are the least likely to contain stopwords or start with a question word. The queries recommended by Skip are longer than those recommended by Balanced , and are more likely to start with a question word. Nevertheless, Balanced outperformed Skip in terms of CTR , indicating that these attributes are important but insufficient in determining query relevance.

[6] We consider queries that start with one of the following question words: who, where, why, when, how, what, which, whose, whom, is, are, do, does, did, was, were, has, have, will and can.
[7] We use Lucene's English stopword list.

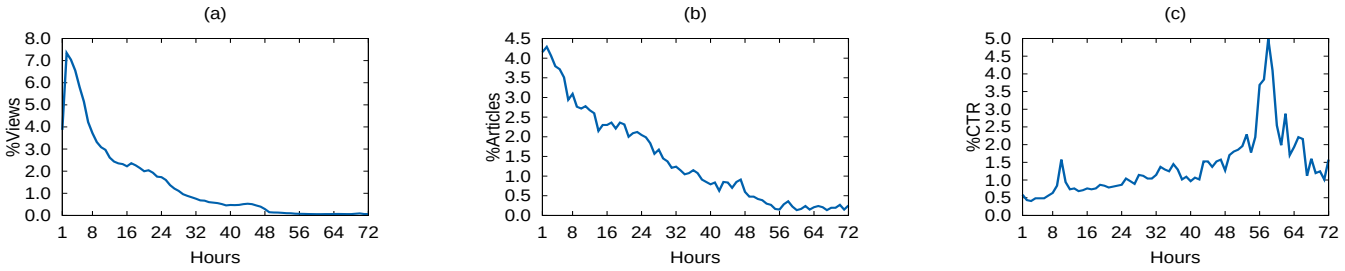


Figure 4: The change in (a) the percentage of article views, (b) the percentage of articles with at least five clicks on recommended queries, and (c) the CTR on the queries as a function of the number of hours that passed since an articles was first viewed.

Table 9: Distribution of clicks on organic results and the various verticals shown on the SERP .

	Balanced	SameQuery	AfterNewsURL	AfterNewsOther
Organic	10.82	36.72	38.60	54.84
News	32.28	22.96	19.45	5.11
Entities	5.63	3.79	7.00	2.96
Images	25.25	10.01	17.26	7.57
Videos	13.92	6.02	8.37	3.69
Ads	10.89	8.11	6.64	20.02
Other	1.21	12.39	2.68	5.81

4.2.6 Recommended versus Native Queries. In Section 4.2.3 we studied the interaction of users with the SERP presented after a recommended query was clicked in terms of Revenue and Reformulation . In what follows, we compare users' interactions with the SERP once they open it either by clicking on a recommended query or by issuing a native query themselves. We extracted all the searches initiated by queries that were recommended by the Balanced model. This was done during a non-consecutive two-day time period in which the model served all Yahoo News users. (These users were from the United States working on desktop devices.) We compared these searches with those invoked by native user queries. Specifically, we consider searches in which the user issued a query, not necessarily after reading a news article; this query was also clicked by users presented with queries recommended by the Balanced model (SameQuery). In addition, we consider searches initiated by a user query that was issued within the first five minutes after the user read a news article, where no other actions were performed in between. We distinguish between searches in which at least one (AfterNewsURL) or none (AfterNewsOther) of the query terms appeared in the originating article's URL; the former queries are more likely to be related to the news article. We excluded searches that were initiated by 40 different navigational queries, such as "facebook", "youtube", and "google". Since we recommend fresh queries that were added to our dataset recently (e.g., the day before), for the native searches we inspected not only the two days mentioned above, but also the searches that took place a day before, i.e., we inspected four days in total.

In Table 9 we present the click distribution on different result types (verticals) that are shown on the page [2], including Organic (organic "blue links" results), News (news articles), Entities (mostly presenting information about persons and locations [7]), Images , Videos and Ads . We can see that among the four types of searches we inspected, users arriving to a search page after clicking on a recommended query are more likely to examine additional news articles that are potentially related to the previously read article. These users are also more likely to click on images and videos. This might suggest that recommended queries elicit users to seek auxiliary visual information to further deepen their knowledge and satisfy their curiosity. Similar to Kong et al. [19], we see that users are more likely to seek information about various entities after reading a news article (see Balanced and AfterNewsURL vs. SameQuery and AfterNewsOther).

Although the SameQuery searches were invoked by queries that were also recommended by the Balanced model, they are quite different. For example, in the SameQuery searches, users are more inclined to click on organic results. This is also the case for the AfterNewsURL and AfterNewsOther searches, which are invoked following an interaction with a news article. We also see that the Ads vertical receives the highest percentage of clicks in the AfterNewsOther searches. Presumably queries leading to these searches are less likely to be related to the originating news article and are therefore more likely to have some commercial intent.

4.2.7 Article Freshness. As already noted, time plays a key role when dealing with news, especially breaking news, articles. In this work, we have taken a number of steps to promote fresh and emerging queries. First, the query index was updated on a regular basis. Second, only queries that were issued in the past three weeks were kept in the index. Third, several features that quantify the trendiness of queries were used in the learning-to-rank model (i.e., AvgUsers , StdvUsers , DecDistinctUsers and AvgUserHistory).

While the emphasis so far was on query freshness, in what follows we focus on article freshness. In Figure 4 we examine various aspects of the change in the users' behavior when interacting with news articles, and our recommended queries in particular, as a function of the time that has passed from the moment an article was first viewed. This analysis is based on logs collected during the second part of the online experiment. Here, we do not distinguish between the different buckets. At any given time, we only consider articles with at least five clicks on the recommended queries. All

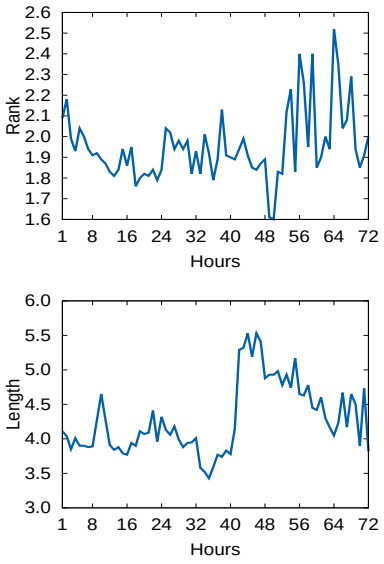


Figure 5: The change in the average click rank and average query length as a function of the number of hours that passed since an articles was first viewed.

the numbers are sum-normalized with respect to the total attained during the three-day time period

We start by studying the percentage of views an article received in an hour over the course of the three days. We observe a sharp decline in the number of views. For example, we found that on average an article reaches 50.1% of its views during the first ten hours and 81.2% during the first day. This finding demonstrates the relatively short lifespan of news articles and the fast pace of news consumption. We next examine the percentage of articles with at least five clicks on the recommended queries in an hour. Here again we observe a sharp decline, which testifies to the need for a fast algorithm that can produce recommendations immediately after the article is published. Inspecting the CTR (sum normalized over the three-day time period), we see that users start clicking on recommendations from the very first moment the article is viewed, i.e., already in the first hour. We observe a steady lift in CTR during the first two days, and a rather sharp increase during the third day. We postulate that the few articles that still get clicks during the third day discuss compelling and ongoing events, in which case users are more inclined to explore further. Indeed, we found that the CTR of articles with clicks in the first day only was 3.4 times lower than that of articles with clicks in all three days.

In Figure 5 we examine the properties of the clicked queries as a function of article freshness. We notice that the average rank of the clicked queries is rather stable in the first two days (average: 1.91; standard deviation: 0.09) and is slightly higher during the third day (average: 2.03; standard deviation: 0.24). A similar observation can be made about query length that increases during the third day. We attribute these changes to the somewhat different nature of the articles that are still being read during the third day and the users' desire for additional related content for such articles.

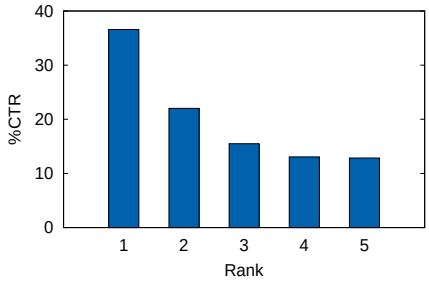


Figure 6: Sum-normalized CTR across the top-five ranks.

For example, we found that the average query length was 3.8 for articles that received clicks during the first day only, and 4.3 for those that received clicks during the entire three-day time period. Integrating the article's freshness into the query recommendation algorithm is an interesting future research direction.

4.2.8 Position Bias. To learn feature weights using user feedback in the second retrieval phase, we experimented with two different training sets so as to overcome the bias incurred in users' clicking decisions [17]. In what follows, we study the severity of the position (trust) bias wherein users tend to click on highly ranked items even if they are less relevant than those ranked lower. In Figure 6 we show the CTR across the five ranks in which queries were recommended during a two-week time period to all Yahoo News users. During this time the top-five queries recommended by the Balanced model were randomly shuffled across ranks. As expected, a rather sharp position bias can be observed: the higher the query is presented, the more likely it is to receive clicks. This finding, along with the differences in performance observed earlier between the Skip and Balanced models, attest to the importance of correctly interpreting user clicks. Testing additional approaches is left for future work [1, 33].

5 CONCLUSIONS AND FUTURE WORK

Content enrichment lets users access information they may not be aware of, such as related articles or relevant facts or events, thereby serving to increase engagement. We addressed the task of enriching news-articles by recommending related search queries that are extracted from a search engine's query log. The three-phase approach we proposed for generating the recommendations is based on a learning-to-rank model that uses various article-dependent and article-independent features. Our approach also integrates several highly effective semantic query filters. Empirical evaluation performed using an offline dataset created by professional editors and a large-scale online experiment with real users attested to the effectiveness of our approach. A deeper analysis of the results revealed interesting characteristics of the types of queries users tend to click and the unique behavior of users after clicking on such queries. For future work we intend to study the effectiveness of our approach when applied to other domains. For example, focusing on specific categories, such as sports, finance or scientific articles with a unique jargon (e.g., Jets, Dolphins and Giants are sport team names). We also plan to study the influence of the user interface on user engagement, considering the position of the recommendations

within the article, the number of recommendations per article and the possible differences between desktop and mobile devices.

Acknowledgements We thank the reviewers for their comments. We also thank Jonathan Braude, Adi Grossman, Ori Koral, Dan Pelleg, Ido Yablonka and Vadim Zak for technical assistance and valuable discussions.

REFERENCES

[1] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W. Bruce Croft. 2018. Unbiased Learning to Rank with Unbiased Propensity Estimation. In *Proceedings of SIGIR*. 385–394.

[2] Jaime Arguello, Fernando Diaz, Jamie Callan, and Jean-Francois Crespo. 2009. Sources of evidence for vertical selection. In *Proceedings of SIGIR*. 315–322.

[3] Michael Bendersky, W. Bruce Croft, and Yanlei Diao. 2011. Quality-biased ranking of web documents. In *Proceedings of WSDM*. 95–104.

[4] Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. 2011. Learning Structured Embeddings of Knowledge Bases. In *Proceedings of AAAI*. 301–306.

[5] Ilaria Bordino, Gianmarco De Francisci Morales, Ingmar Weber, and Francesco Bonchi. 2013. From Machu_Picchu to Rafting the Urubamba River: Anticipating Information Needs via the Entity-query Graph. In *Proceedings of WSDM*. 275–284.

[6] Ilaria Bordino, Yelena Mejova, and Mounia Lalmas. 2013. Penguins in Sweaters, or Serendipitous Entity Search on User-generated Content. In *Proceedings of CIKM*. 109–118.

[7] Horatiu Bota, Ke Zhou, and Joemon M Jose. 2016. Playing your cards right: The effect of entity cards on search behaviour and workload. In *Proceedings of CHIIR*. 131–140.

[8] Xuezhi Cao, Kailong Chen, Rui Long, Guoqing Zheng, and Yong Yu. 2012. News Comments Generation via Mining Microblogs. In *Proceedings of WWW Companion*. 471–472.

[9] Zhicong Cheng, Bin Gao, and Tie-Yan Liu. 2010. Actively Predicting Diverse Search Intent from User Browsing Behaviors. In *Proceedings of WWW*. 221–230.

[10] Koby Crammer, Alex Kulesza, and Mark Dredze. 2009. Adaptive Regularization of Weight Vectors. In *Proceedings of NIPS*. 414–422.

[11] Na Dai, Milad Shokouhi, and Brian D Davison. 2011. Learning to rank for freshness and relevance. In *Proceedings of SIGIR*. 95–104.

[12] Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of IJCAI*. 1606–1611.

[13] Parth Gupta and José Santos. 2017. Learning to Classify Inappropriate Query-Completions. In *Proceedings of ECIR*. 548–554.

[14] Michael Heilman and Noah A. Smith. 2010. Good Question! Statistical Ranking for Question Generation. In *Proceedings of NAACL HLT*. 609–617.

[15] Katja Hofmann, Lihong Li, and Filip Radlinski. 2016. Online Evaluation for Information Retrieval. *Foundations and Trends in Information Retrieval* 10, 1 (2016), 1–117.

[16] Thorsten Joachims. 2006. Training linear SVMs in linear time. In *Proceedings of SIGKDD*. 217–226.

[17] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of SIGIR*. 154–161.

[18] Rosie Jones, Ravi Kumar, Bo Pang, and Andrew Tomkins. 2007. I know what you did last summer: query logs and user privacy. In *Proceedings of CIKM*. 909–914.

[19] Weize Kong, Rui Li, Jie Luo, Aston Zhang, Yi Chang, and James Allan. 2015. Predicting Search Intent Based on Pre-Search Context. In *Proceedings of SIGIR*. 503–512.

[20] Ralf Krestel, Thomas Werkmeister, Timur Pratama Wiradarma, and Gjergji Kasneci. 2015. Tweet-Recommender: Finding Relevant Tweets for News Articles. In *Proceedings of WWW Companion*. 53–54.

[21] John D. Lafferty and Chengxiang Zhai. 2001. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of SIGIR*. 111–119.

[22] Chia-Jung Lee and W Bruce Croft. 2012. Generating queries from user-selected text. In *Proceedings of IIiX*. 100–109.

[23] Tie-Yan Liu. 2011. *Learning to Rank for Information Retrieval*. Springer. I–XVII, 1–285 pages.

[24] Donald Metzler and W. Bruce Croft. 2005. A Markov random field model for term dependencies. In *Proceedings of SIGIR*. 472–479.

[25] Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking Documents to Encyclopedic Knowledge. In *Proceedings of CIKM*. 233–242.

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of NIPS*. 3111–3119.

[27] Hadas Raviv, Oren Kurland, and David Carmel. 2016. Document Retrieval Using Entity-Based Language Models. In *Proceedings of SIGIR*. 65–74.

[28] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of TREC*.

[29] Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Cristian Moldovan. 2010. The First Question Generation Shared Task Evaluation Challenge. In *Proceedings of INLG*. 251–257.

[30] J. Ben Schafer, Joseph A. Konstan, and John Riedl. 2001. E-Commerce Recommendation Applications. *Data Min. Knowl. Discov.* 5, 1-2 (Jan. 2001), 115–153.

[31] Erez Shmueli, Amit Kagian, Yehuda Koren, and Ronny Lempel. 2012. Care to Comment?: Recommendations for Commenting on News Stories. In *Proceedings of WWW*. 429–438.

[32] Milad Shokouhi and Qi Guo. 2015. From Queries to Cards: Re-ranking Proactive Card Recommendations Based on Reactive Search History. In *Proceedings of SIGIR*. 695–704.

[33] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position Bias Estimation for Unbiased Learning to Rank in Personal Search. In *Proceedings of WSDM*. 610–618.

[34] Guang Xiang, Bin Fan, Ling Wang, Jason Hong, and Carolyn Rose. 2012. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *Proceedings of CIKM*. 1980–1984.

[35] Harish Yenala, Ashish Jhanwar, Manoj K Chinnakotla, and Jay Goyal. 2017. Deep learning for detecting inappropriate content in text. *International Journal of Data Science and Analytics* (2017), 1–14.

[36] Yunzhang Zhu, Gang Wang, Junli Yang, Dakan Wang, Jun Yan, Jian Hu, and Zheng Chen. 2009. Optimizing search engine revenue in sponsored search. In *Proceedings of SIGIR*. 588–595.

Revisiting Opinion Dynamics with Varying Susceptibility to Persuasion via Non-Convex Local Search

T-H. Hubert Chan*
The University of Hong Kong

Zhibin Liang
The University of Hong Kong

Mauro Sozio
LTCI, Télécom ParisTech University

ABSTRACT

We revisit the opinion susceptibility problem that was proposed by Abebe et al. [1], in which agents influence one another's opinions through an iterative process. Each agent has some fixed innate opinion. In each step, the opinion of an agent is updated to some convex combination between its innate opinion and the weighted average of its neighbors' opinions in the previous step. The resistance of an agent measures the importance it places on its innate opinion in the above convex combination. Under non-trivial conditions, this iterative process converges to some equilibrium opinion vector. For the unbudgeted variant of the problem, the goal is to select the resistance of each agent (from some given range) such that the sum of the equilibrium opinions is minimized.

Contrary to the claim in the aforementioned KDD 2018 paper, the objective function is in general non-convex. Hence, formulating the problem as a convex program might have potential correctness issues. We instead analyze the structure of the objective function, and show that any local optimum is also a global optimum, which is somehow surprising as the objective function might not be convex. Furthermore, we combine the iterative process and the local search paradigm to design very efficient algorithms that can solve the unbudgeted variant of the problem optimally on large-scale graphs containing millions of nodes.

CCS CONCEPTS

• **Human-centered computing** → **Social network analysis**; • **Mathematics of computing** → **Graph algorithms**.

KEYWORDS

opinion dynamics, susceptibility to persuasion, non-convex local search

ACM Reference Format:
T-H. Hubert Chan, Zhibin Liang, and Mauro Sozio. 2019. Revisiting Opinion Dynamics with Varying Susceptibility to Persuasion via Non-Convex Local Search. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313509

*This research was partially supported by a grant from the PROCORE France-Hong Kong Joint Research Scheme sponsored by the Research Grants Council of Hong Kong and the Consulate General of France in Hong Kong under the project F-HKU702/16. T-H. Hubert Chan was partially supported by the Hong Kong RGC under the grant 17200817.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313509

1 INTRODUCTION

We revisit the opinion susceptibility problem that was proposed by Abebe et al. [1], in which agents in a network influence one another's opinions through an iterative process. In this model, the opinion of an agent at each step is a function of its *innate* opinion and its neighbors' opinions in the previous step. The susceptibility of an agent i to persuasion is measured by a resistance parameter $\alpha_i \in (0, 1)$, where a larger value of α_i means that agent i places more importance on its innate opinion. Under very mild conditions (such as every $\alpha_i > 0$), the iterative process converges to a unique equilibrium opinion vector.

As remarked in [1], opinion interference in social networks have wide applications in product marketing and political campaigns. While most previous works [6, 11, 16] investigated models on changing the opinions directly, [1] is the first work that considered modifying the susceptibility of individual agents. Using a number in $[0, 1]$ to represent an agent's opinion, they considered the problem of modifying the agents' resistances such that the sum of the equilibrium opinions is minimized (or equivalently maximized by applying the transformation $x \mapsto 1 - x$ on each opinion).

Abebe et al. [1] pointed out several works in social psychology that have investigated people's susceptibility to persuasion, including studies on how peer pressure can make adolescents exhibit risky and antisocial behavior [2, 5, 7], and on the role of susceptibility in politics [19]. Factors such as self-esteem, locus of control, awareness of specific issues, as well as exposure to people in the same network can contribute to an individual's susceptibility to persuasion. As a result, by affecting these factors or introducing persuasive cues [15], one can make an individual more or less susceptible to persuasion, i.e., the resistance parameter can be lowered or increased.

Observe that the problem is trivial if the resistance of each agent can be picked from the closed interval $[0, 1]$. For minimizing the equilibrium opinions, it suffices to make the agent with the minimum innate opinion the most resistant (setting its resistance to 1) and everyone else totally compliant (setting its resistance to 0). The problem is non-trivial if the resistance α_i of each agent i can take value from some interval $[l_i, u_i]$, where $0 < l_i < u_i < 1$. Under this restriction, Abebe et al. [1] showed that it suffices to consider extreme points in the resistance vector space, i.e., for each agent i, one can pick $\alpha_i \in \{l_i, u_i\}$. Moreover, they attempt to model the problem using a convex program. However, contrary to their claim, we have discovered that the objective function is in general neither convex nor concave (see Figure 1 in Section 2). Hence, there could be potential correctness issues with their approach. Furthermore, a general convex program is quite expensive to solve, and indeed, the experiments in [1] were performed on networks with less than 1000 nodes, most likely because their approach is non-scalable to large networks.

This work resolves both the correctness and scalability issues in [1]. We give a thorough analysis of the problem structure to design a provably correct local search framework, and further exploit the properties of the iterative process to implement our algorithms that are efficient enough to run on networks with millions of nodes.

Our Contributions. We study the structure of the opinion susceptibility problem [1] in detail, and also make algorithmic contributions, which we describe as follows.

- *Analysis of problem structure.* Contrary to the claim in [1], we discover that the objective function is in general neither convex nor concave. However, we have also recovered (in Corollary 3.4) that for each agent, it is sufficient to set its resistance to either the lower or the upper bound, i.e., we can search for an optimal resistance vector from the set of extreme points of its feasible region.
  We analyze the mathematical structure of the problem in Section 3. Perhaps the most important technical insight in this paper is that we show (in Lemma 3.9) that if the current vector solution is not optimal, then there exists a coordinate that can be flipped such that the objective will be strictly improved. This shows that an optimal vector can be found by a simple local search algorithm.
- *Local search with irrevocable updates.* In general, local search could still take exponential time to find an optimal solution, for instance, the simplex algorithm for linear programming. For minimizing the sum of equilibrium opinions, we show (in Lemma 4.1) that starting from the upper bound resistance vector, then the local search algorithm will flip each coordinate at most once, which implies that an optimal vector can be found in polynomial time.
- *Efficient Local Search on Large-Scale Graphs.* Typically, in local search, the objective function needs to be evaluated at the current solution in each step. However, since the objective function involves matrix inverse, its evaluation will be too expensive when the dimension of the matrix is in the order of millions. Instead, we use the iterative process of the opinion dynamics model itself to approximate the equilibrium vector. We have developed several update strategies for local search. For *conservative* or *opportunistic* updates, one always makes sure that the error of the estimated equilibrium vector is small enough before any coordinate of the resistance vector is flipped. For *optimistic* update, one might flip a coordinate of the resistance vector even before the estimated equilibrium vector is accurate enough. However, this might introduce mistakes which need to be corrected later. Nevertheless, experiments show that mistakes are rarely made by the optimistic update strategy. In any case, for all three update strategies, an optimal vector will be returned when the local search terminates.
  Our approaches are scalable and can run on networks with millions of nodes. We report the experimental results in Section 5. In particular, using multiple number of threads, the optimistic update strategy can solve the problem optimally on networks with up to around 65 million nodes.

1.1 Related Works

The most related work is the KDD 2018 paper [1] by Abebe et al., who gave a history of the problem and other related works, to which the reader can refer. Here we just give a brief summary.

History of Model. As mentioned in [1], the opinion dynamics model is inspired by the works of DeGroot [4] and Friedkin and Johnsen [8]. DeGroot considered the special case where all resistance α_i's equal 0, and the opinion vector update rule reduces to $z^{(t)} := Pz^{(t-1)} \in \mathbb{R}^V$, for some row stochastic matrix P. This model is well understood through the theory of Markov chains, which typically uses the transpose of P in the update rule: $z^{(t)} := P^{\mathsf{T}}z^{(t-1)} \in \mathbb{R}^V$. The iterative process considered in this paper is exactly the model in [8], which also investigated the conditions under which the process converges to a unique equilibrium vector.

Budgeted Variant. We have so far discussed only the *unbudgeted* variant of the opinion optimization problem, where the resistance of every agent can be modified. The *budgeted* variant of the problem has also been considered in [1], where some initial resistance vector $\alpha \in (0,1)^V$ and some budget $k \in \mathbb{Z}$ are given, and the algorithm is allowed to change at most k coordinates of α. It is shown that the budgeted variant is NP-hard [1]. An obvious approach is to use some greedy paradigm to pick k coordinates to modify one by one, where the next coordinate is picked to have the largest improvement in the objective function. However, no theoretical guarantee was given for budgeted variant in [1]. Since our analysis of the unbudgeted variant is already quite involved, this paper will focus on the unbudgeted case. We will leave the analysis of the greedy algorithm for the budgeted variant as future work, which we briefly discuss in the conclusion section.

Non-Convex Optimization. In general, optimizing a non-convex function under non-convex constraints is NP-hard. However, in many cases, one can exploit the structure of the objective function or constraints to devise polynomial-time algorithms; see the survey by Jain and Kar [13] on non-convex optimization algorithms encountered in machine learning. Indeed, variants of the gradient descent have been investigated to escape saddle points by Jin et al. [14], who also gave examples of problems where all local optima are also global optima; some examples are tensor decomposition [9], dictionary learning [22], phase retrieval [21], matrix sensing [3, 20] and matrix completion [10]. However, all these problems involve some quadratic loss functions, whose structures are totally different from our objective functions which involve matrix inverse.

Hartman [12] considered the special case that the objective function is the difference of two convex functions. Strekalovsky devised a local search method to optimize such objective functions. Even though the objective functions in our problem are somewhere convex and somewhere concave (see Figure 1), it is not immediately clear if they can be expressed as differences of convex functions.

2 PRELIMINARIES

We revisit the opinion susceptibility problem that was proposed by Abebe et al. [1], which we rephrase using more general parameters. In an opinion dynamics model, there is a set V of agents, whose innate opinions are given by a vector $s \in [0,1]^V$. The interaction between agents are captured by a row stochastic matrix $P \in [0,1]^{V \times V}$, i.e., each entry of P is non-negative and every row sums to 1. In

the rest of this paper, we always call P the interaction matrix. The susceptibility of the agents is measured by the resistance vector $\alpha \in (0,1)^V$, where a higher α_i value means that agent i is more resistant.

As mentioned in [1], the opinion vector of agents evolves in discrete time according to the equation $z^{(t+1)} := As + (I - A)Pz^{(t)}$, where $A = \text{Diag}(\alpha)$ is the diagonal matrix with $A_{ii} = \alpha_i$ and I is the identity matrix. Equating $z^{(t)}$ with $z^{(t+1)}$, one can see that the equilibrium opinion vector is given by $z = [I - (I - A)P]^{-1}As$, which exists under non-trivial conditions such as every $\alpha_i > 0$. The objective is to choose a resistance vector α to minimize the sum of equilibrium opinions $\langle \mathbf{1}, z \rangle = \mathbf{1}^T z$. Observe that one can also consider maximizing the sum of equilibrium opinions; however, since the techniques are essentially the same, we will focus on the minimization variant of the problem.

Definition 2.1 (Opinion Susceptibility Problem). Given a set V of agents with innate opinions $s \in [0,1]^V$ and interaction matrix $P \in [0,1]^{V \times V}$, suppose for each $i \in V$, its resistance is restricted to some interval $\mathcal{I}_i := [l_i, u_i] \subseteq [0,1]$ where we assume that $0 < l_i < u_i < 1$.

The objective is to choose $\alpha \in \mathcal{I}_V := \times_{i \in V} \mathcal{I}_i \subseteq [0,1]^V$ such that the following objective function is minimized:

$$f(\alpha) := \mathbf{1}^T [I - (I - A)P]^{-1} As,$$

where $A = \text{Diag}(\alpha)$ is the diagonal matrix with $A_{ii} = \alpha_i$. Observe that the assumption $\alpha > 0$ ensures that the above inverse exists.

Unbudgeted vs Budgeted Variants. In Definition 2.1, we are allowed to modify the resistance of any agent, and this is known as the unbudgeted variant. However, in [1], the budgeted variant is also considered: given some initial resistance vector and a budget k, the resistance of at most k agents can be changed. In this paper, we focus on efficient algorithms that optimally solve the unbudgeted variant.

Technical Assumption. To simplify our proofs, we assume that the interaction matrix P corresponds to an irreducible random walk. Irreducibility is satisfied if P arises from a connected graph.

Non-convex Objective. Contrary to the claim in [1], the objective f in Definition 2.1 is in general not a convex function of α. In fact, the following example shows that it might be neither convex nor concave. Consider three vertices $V = \{1, 2, 3\}$, where the innate vector s and the interaction matrix P are given by: $s = \begin{bmatrix} 1 \\ 0.5 \\ 0 \end{bmatrix}$ and $P = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.5 & 0 & 0.5 \\ 0.5 & 0.5 & 0 \end{bmatrix}$.

Suppose we fix $\alpha_2 = \alpha_3 = 0.1$ and consider the objective as a function of α_1 as $g(\alpha_1) = \mathbf{1}^T [I - (I - A)P]^{-1} As$, where $A = \text{Diag}(\alpha_1, \alpha_2, \alpha_3)$. Then, the plot of g in Figure 1 (a) shows that it is not convex. Moreover, suppose this time we fix $\alpha_1 = \alpha_2 = 0.1$ and consider the objective as a function of α_3 as $h(\alpha_3) = \mathbf{1}^T [I - (I - A)P]^{-1} As$. Then, the plot of h in Figure 1 (b) shows that it is not concave.

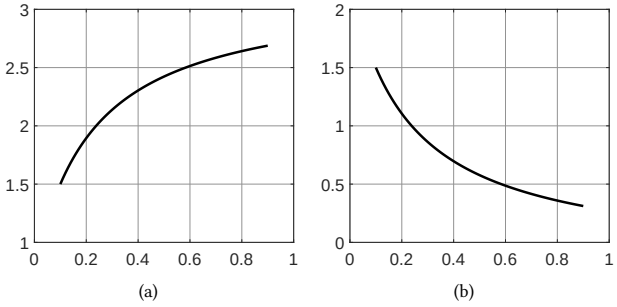


Figure 1: Cross-Sections of Objective Function

3 STRUCTURAL PROPERTIES OF OBJECTIVE FUNCTION

In this section, we investigate the properties of the objective function f in Definition 2.1; we assume that the interaction matrix P and the innate opinion vector s are fixed, and f is a function on the resistance vector α. Contrary to the claim in [1], the objective function is in general not convex. Fortunately, we can still exploit some properties of the function. As we shall see, even when the function is not convex, every local optimum (which will be defined formally) is a global optimum. This enables us to use variants of the local search method to solve the problem optimally.

3.1 Marginal Monotonicity

As in [1], we show that when one chooses the resistance α_i for each agent $i \in V$, it suffices to consider the extreme points $\{l_i, u_i\}$. Our approach explicitly analyzes the partial derivative $\frac{\partial f(\alpha)}{\partial \alpha_i}$ which plays a major role in the local search algorithm that we later develop.

Intuition: Guidance by Current Equilibrium Vector. Observe that given the innate opinion vector s and irreducible interaction matrix P, for some resistance vector $\alpha \in (0,1)^V$, the equilibrium opinion vector is given by $z(\alpha) = [I - (I - A)P]^{-1}As$, where $A = \text{Diag}(\alpha)$. For some $i \in V$, if the innate opinion s_i is larger than its equilibrium $z_i(\alpha)$, this suggests that by being more stubborn, agent i should be able to increase its equilibrium opinion. In other words, one would expect $\frac{\partial z_i(\alpha)}{\partial \alpha_i}$ and $s_i - z_i(\alpha)$ to have the same sign. However, what is surprising is that in Lemma 3.2, we shall see that even for any $j \in V$, $\frac{\partial z_j(\alpha)}{\partial \alpha_i}$ and $s_i - z_i(\alpha)$ have the same sign.

Notation. For any $\alpha \in \mathbb{R}^V$ and $K \subseteq V$, let $\alpha_{-K} \in \mathbb{R}^V$ denote the vector such that $\alpha_{-K}(i) = \alpha(i)$ if $i \notin K$, and $\alpha_{-K}(i) = 0$ if $i \in K$, i.e., the coordinates K of α are replaced with 0. Similarly, given $\alpha \in \mathbb{R}^V$, we denote $A_{-K} = \text{Diag}(\alpha_{-K})$.

In Definition 2.1, observe that the inverse $[I - (I - A)P]^{-1}$ is involved in the objective function $f(\alpha)$, where $A = \text{Diag}(\alpha)$. Since we wish to analyze the effect on $f(\alpha)$ of changing only a subset of coordinates in α, the next lemma will be used for simplifying matrix arithmetic involving the computation of inverses. Its proof is deferred to Section 6.

Lemma 3.1 (Inverse Arithmetic). *Given $K \subsetneq V$ and $\alpha \in (0,1)^V$, let $A := \text{Diag}(\alpha)$ and recall that P is the irreducible interaction matrix. Then, the inverse $M = [I - (I - A_{-K})P]^{-1}$ exists, and every entry*

of M is positive. Moreover, for each $k \in V$, define $a_k = 0$ if $k \in K$, otherwise $a_k = \alpha_k$. Then, we have:

(1) $(PM)_{kk} = \frac{M_{kk}-1}{1-a_k} > 0$;

(2) $(PM)_{kj} = \frac{M_{kj}}{1-a_k} > 0$, *for each $j \neq k$.*

Lemma 3.2 (Sign of Partial Derivative). *In the Opinion Susceptibility Problem in Definition 2.1, given the innate opinion vector s and irreducible interaction matrix P, recall that $z(\alpha) := [I-(I-A)P]^{-1}As$, where $A = \mathrm{Diag}(\alpha)$. Then, for any $\alpha \in (0,1)^V$ and any $i, k \in V$, the two values $\frac{\partial z_k(\alpha)}{\partial \alpha_i}$ and $s_i - z_i(\alpha)$ have exactly the same sign in $\{-, 0, +\}$.*

In particular, this implies that $\frac{\partial f(\alpha)}{\partial \alpha_i} = \sum_{k\in V} \frac{\partial z_k(\alpha)}{\partial \alpha_i}$ also has the same sign as $s_i - z_i(\alpha)$.

Proof. By the definition of the inverse of a matrix B, we have $BB^{-1} = I$. The partial derivative with respect to a variable t is: $\frac{\partial B}{\partial t}B^{-1} + B\frac{\partial B^{-1}}{\partial t} = 0$. Hence, we have $\frac{\partial B^{-1}}{\partial t} = -B^{-1}\frac{\partial B}{\partial t}B^{-1}$. Applying the above result with $B = I - (I - A)P$ and $t = \alpha_i$ and denoting $M = [I - (I - A)P]^{-1}$, we get $\frac{\partial M}{\partial \alpha_i} = -M\mathbf{e}_i\mathbf{e}_i^T PM$. Considering $z(\alpha) = MAs$, we have $\frac{\partial z(\alpha)}{\partial \alpha_i} = \frac{\partial M}{\partial \alpha_i}As + M\mathbf{e}_i\mathbf{e}_i^T s$. Replacing $\frac{\partial M}{\partial \alpha_i}$, we obtain for any $i, k \in V$:

$$\frac{\partial z_k(\alpha)}{\partial \alpha_i} = -\mathbf{e}_k^T M\mathbf{e}_i\mathbf{e}_i^T PMAs + \mathbf{e}_k^T M\mathbf{e}_i\mathbf{e}_i^T s = M_{ki}\cdot[s_i - \mathbf{e}_i^T Pz(\alpha)].$$

By Lemma 3.1 with $K = \emptyset$, we know that every entry of M is positive. Thus, the sign of $\frac{\partial z_k(\alpha)}{\partial \alpha_i}$ is the same as that of the scalar $s_i - \mathbf{e}_i^T Pz(\alpha)$.

Recalling $M = [I - (I - A)P]^{-1}$, we have $[I - (I - A)P]M = I \Rightarrow (I - A)PM = M - I \Rightarrow PM = (I - A)^{-1}(M - I)$ where $(I - A)^{-1}$ exists since $\alpha_j < 1$ for each $j \in V$.

Next, since $z(\alpha) = MAs$, we have:

$$Pz(\alpha) = PMAs = (I - A)^{-1}(M - I)As = (I - A)^{-1}[z(\alpha) - As].$$

Finally, replacing $Pz(\alpha)$, we have

$$\begin{aligned} s_i - \mathbf{e}_i^T Pz(\alpha) =& s_i - \mathbf{e}_i^T (I - A)^{-1}[z(\alpha) - As] \\ =& s_i - \frac{1}{1-\alpha_i}[z_i(\alpha) - \alpha_i s_i] \\ =& \frac{1}{1-\alpha_i}[s_i - z_i(\alpha)]. \end{aligned}$$

Since $1-\alpha_i > 0$, it follows that $\frac{\partial z_k(\alpha)}{\partial \alpha_i}$ and $s_i - z_i(\alpha)$ have exactly the same sign in $\{-, 0, +\}$, as required. □

The next lemma shows that the sign of the partial derivatives with respect to coordinate i is actually independent of the current value α_i. Its proof is deferred to Section 6.

Lemma 3.3 (Sign of Partial Derivative Independent of Coordinate Value). *Referring to Lemma 3.2. For any $\alpha \in (0,1)^V$ and any $i \in V$, denote $M = [I - (I - A_{-\{i\}})P]^{-1}$. Then, $\frac{\partial f(\alpha)}{\partial \alpha_i}$ has the same sign in $\{-, 0, +\}$ as $s_i - \sum_{j\neq i} M_{ij}\alpha_j s_j$, which is independent of α_i.*

Corollary 3.4 (Extreme Points are Sufficient). *In Definition 2.1, for any $i \in V$, fixing the resistance values of all other agents except i, the objective $f(\alpha)$ is a monotone function in α_i. This implies that to minimize f, it suffices to consider the extreme points $\alpha_i \in \{l_i, u_i\}$, for each $i \in V$.*

3.2 Local vs Global Optimum

As shown in Corollary 3.4, it suffices to choose the resistance vector α from the extreme points in Definition 2.1. Lemma 3.2 readily gives a method to decide, given a current choice of α, whether the objective f can be decreased by changing the resistance of some agent. In Lemma 3.9, we show that if α is not a global minimum, then such an agent must exist. As we shall see, this implies that a local search method can find a global minimum.

Given α and $\alpha' \in \mathbb{R}^V$, denote $\Delta(\alpha, \alpha') := \{i \in V : \alpha_i \neq \alpha'_i\}$ as the set of coordinates at which the vectors differ.

Definition 3.5 (Local Minimizer). Given an objective function $f : \mathcal{I}_V \to \mathbb{R}$, a vector $\alpha \in \mathcal{I}_V$ is a local minimizer of f, if for all $\alpha' \in \mathcal{I}_V$ such that $\Delta(\alpha, \alpha') = 1$, $f(\alpha) \leq f(\alpha')$.

Notation. When we wish to consider the effect of changing the resistance of only 2 agents $i \neq k \in V$, we write $f(\alpha) = f(\alpha_i, \alpha_k)$, assuming that $\alpha_{-\{i,k\}}$ is fixed.

Lemmas 3.6 and 3.7 give some technical results involving changing the resistance of two agents. Their proofs are deferred to Section 6.

Lemma 3.6. *For any $i, k \in V$ such that $i \neq k$, let $M = [I - (I - A_{-\{i\}})P]^{-1}$ and $R = [I - (I - A_{-\{i,k\}})P]^{-1}$. Then for any $j \in V$, we have*

(1) $M_{jk} = \frac{R_{jk}}{1+\alpha_k R_{kk} - \alpha_k}$,

(2) $M_{jh} = R_{jh} - \frac{\alpha_k R_{jk} R_{kh}}{1+\alpha_k R_{kk} - \alpha_k}$, *for $h \neq k$.*

In particular, the quantity in Lemma 3.3 can be rewritten as follows:

$$s_i - \sum_{j\neq i} M_{ij}\alpha_j s_j = s_i - \sum_{j\neq i,k} R_{ij}\alpha_j s_j - \frac{\alpha_k R_{ik}}{1 + \alpha_k R_{kk} - \alpha_k}(s_k - \sum_{j\neq i,k} R_{kj}\alpha_j s_j).$$

Lemma 3.7 (Diagonal Entry). *Suppose $\alpha \in (0,1)^V$, recall that $A_{-\{i,k\}} := \mathrm{Diag}(\alpha_{-\{i,k\}})$, and P corresponds to an irreducible interaction matrix. For any $i, k \in V$ such that $i \neq k$, let $R = [I - (I - A_{-\{i,k\}})P]^{-1}$, then $R_{ii} = \max_{j\in V} R_{ji}$. Moreover, $R_{ii} = R_{ki}$ if and only if $P_{kk} + P_{ki} = 1$.*

The following lemma gives the key insight for why local search works. Intuitively, it shows that there does not exist any discrete "saddle point". Even though its proof is technical, we still include it here because of its importance.

Lemma 3.8 (Switching Lemma). *Recall that f is defined in Definition 2.1 with an irreducible interaction matrix P, and assume $|V| \geq 3$. Suppose $\alpha, \beta \in (0,1)^V$ such that $\Delta(\alpha, \beta) = \{i, k\}$ for some $i \neq k$. Moreover, suppose further that*

$$\min\{f(\alpha_i, \alpha_k), f(\beta_i, \beta_k)\} < \min\{f(\alpha_i, \beta_k), f(\beta_i, \alpha_k)\}.$$

Then, we have

$$\max\{f(\alpha_i, \alpha_k), f(\beta_i, \beta_k)\} > \min\{f(\alpha_i, \beta_k), f(\beta_i, \alpha_k)\}.$$

Proof. We prove the lemma by contradiction. Suppose

$$\max\{f(\alpha_i, \alpha_k), f(\beta_i, \beta_k)\} \leq \min\{f(\alpha_i, \beta_k), f(\beta_i, \alpha_k)\}.$$

Without loss of generality, suppose further that $f(\alpha_i, \alpha_k) \geq f(\beta_i, \beta_k)$. Then, we have

$$f(\alpha_i, \alpha_k) \leq \begin{cases} f(\beta_i, \alpha_k) \\ f(\alpha_i, \beta_k) \end{cases} \quad \text{and } f(\beta_i, \beta_k) < \begin{cases} f(\beta_i, \alpha_k) \\ f(\alpha_i, \beta_k). \end{cases}$$

We remark that it is important to distinguish between the strict and non-strict inequality. We use the notation f_i' to denote the partial derivative with respect to coordinate i.

From $f(\alpha_i, \alpha_k) \leq f(\beta_i, \alpha_k)$ and the fact that f is marginally monotone (Lemma 3.3) and $f_i'(x, \alpha_k)$ has the same sign in $\{-, 0, +\}$ for $x \in (0, 1)$, we have

$$f_i'(_, \alpha_k) \cdot (\alpha_i - \beta_i) \leq 0. \tag{1}$$

On the other hand, from the strict inequality $f(\beta_i, \beta_k) < f(\beta_i, \alpha_k)$, we know the partial derivative $f_k'(\beta_i, y)$ must have the same non-zero sign in $\{-, +\}$, again from Lemma 3.3. Therefore, we have:

$$f_k'(\beta_i, _) \cdot (\alpha_k - \beta_k) > 0. \tag{2}$$

Similarly, $f(\alpha_i, \alpha_k) \leq f(\alpha_i, \beta_k)$ and $f(\beta_i, \beta_k) < f(\alpha_i, \beta_k)$ give the following:

$$f_k'(\alpha_i, _) \cdot (\alpha_k - \beta_k) \leq 0, \tag{3}$$

$$f_i'(_, \beta_k) \cdot (\alpha_i - \beta_i) > 0. \tag{4}$$

Next, using Lemma 3.3 and R as defined in Lemma 3.6, the above inequalities (1) to (4) become:

$$[s_i - \sum_{j \neq i,k} R_{ij}\alpha_j s_j - \frac{\alpha_k R_{ik}}{1 + \alpha_k R_{kk} - \alpha_k}(s_k - \sum_{j \neq i,k} R_{kj}\alpha_j s_j)](\alpha_i - \beta_i) \leq 0,$$

$$[s_k - \sum_{j \neq i,k} R_{kj}\beta_j s_j - \frac{\beta_i R_{ki}}{1 + \beta_i R_{ii} - \beta_i}(s_i - \sum_{j \neq i,k} R_{ij}\beta_j s_j)](\alpha_k - \beta_k) > 0,$$

$$[s_k - \sum_{j \neq i,k} R_{kj}\alpha_j s_j - \frac{\alpha_i R_{ki}}{1 + \alpha_i R_{ii} - \alpha_i}(s_i - \sum_{j \neq i,k} R_{ij}\alpha_j s_j)](\alpha_k - \beta_k) \leq 0,$$

$$[s_i - \sum_{j \neq i,k} R_{ij}\beta_j s_j - \frac{\beta_k R_{ik}}{1 + \beta_k R_{kk} - \beta_k}(s_k - \sum_{j \neq i,k} R_{kj}\beta_j s_j)](\alpha_i - \beta_i) > 0.$$

Recall that $\alpha_j = \beta_j$ for $j \neq i, k$. Hence, we denote:

$$c_i := s_i - \sum_{j \neq i,k} R_{ij}\alpha_j s_j = s_i - \sum_{j \neq i,k} R_{ij}\beta_j s_j,$$

$$c_k := s_k - \sum_{j \neq i,k} R_{kj}\alpha_j s_j = s_k - \sum_{j \neq i,k} R_{kj}\beta_j s_j,$$

$$g_i(x) := \frac{xR_{ki}}{1 + xR_{ii} - x} \text{ and } g_k(x) := \frac{xR_{ik}}{1 + xR_{kk} - x}.$$

Then, we have

$$[c_i - g_k(\alpha_k)c_k](\alpha_i - \beta_i) \leq 0, \tag{5}$$

$$[c_k - g_i(\beta_i)c_i](\alpha_k - \beta_k) > 0, \tag{6}$$

$$[c_k - g_i(\alpha_i)c_i](\alpha_k - \beta_k) \leq 0, \tag{7}$$

$$[c_i - g_k(\beta_k)c_k](\alpha_i - \beta_i) > 0. \tag{8}$$

Observe that $c_i \neq 0$, otherwise (6) and (7) contradict each other. Similarly, $c_k \neq 0$, otherwise (5) and (8) contradict each other. We next argue that $c_i c_k > 0$.

From (5) and (8) we have

$$[c_i - g_k(\alpha_k)c_k][c_i - g_k(\beta_k)c_k] \leq 0. \tag{9}$$

If $c_i c_k < 0$, then the above expression will be positive, because $g_k(\cdot) \geq 0$ (we shall see that later). Hence, we conclude that c_i and c_k have the same sign.

From (6) and (7) we have

$$[c_k - g_i(\beta_i)c_i][c_k - g_i(\alpha_i)c_i] \leq 0. \tag{10}$$

Rearranging (9) and (10), we have:

$$[\frac{c_i}{c_k} - g_k(\alpha_k)][\frac{c_i}{c_k} - g_k(\beta_k)] \leq 0,$$

and

$$[\frac{c_k}{c_i} - g_i(\beta_i)][\frac{c_k}{c_i} - g_i(\alpha_i)] \leq 0.$$

Note that every entry of R is positive by Lemma 3.1 and we can easily prove $g_i(\cdot)$ and $g_k(\cdot)$ are both strictly increasing functions in $[0, 1]$. Since $\alpha, \beta \in (0, 1)^V$, the above two inequalities imply that

$$0 = g_k(0) < g_k(\min\{\alpha_k, \beta_k\}) \leq \frac{c_i}{c_k},$$

$$\frac{c_i}{c_k} \leq g_k(\max\{\alpha_k, \beta_k\}) < g_k(1) = \frac{R_{ik}}{R_{kk}} \leq 1,$$

and

$$0 = g_i(0) < g_i(\min\{\alpha_i, \beta_i\}) \leq \frac{c_k}{c_i},$$

$$\frac{c_k}{c_i} \leq g_i(\max\{\alpha_i, \beta_i\}) < g_i(1) = \frac{R_{ki}}{R_{ii}} \leq 1,$$

where $\frac{R_{ik}}{R_{kk}} \leq 1$ and $\frac{R_{ki}}{R_{ii}} \leq 1$ are from Lemma 3.7.

Notice that we get $0 < \frac{c_i}{c_k} < 1$ and $0 < \frac{c_k}{c_i} < 1$, which is a contradiction. Hence, the proof is completed. □

Lemma 3.9 (Descending Coordinate). *Let f be the function as defined in Definition 2.1. Suppose $\alpha, \beta \in (0, 1)^V$ such that $f(\alpha) > f(\beta)$. Then, there exists some $i \in \Delta(\alpha, \beta)$ and $\gamma \in (0, 1)^V$ such that $\Delta(\alpha, \gamma) = \{i\}$, $\Delta(\gamma, \beta) = \Delta(\alpha, \beta) \setminus \{i\}$, and $f(\alpha) > f(\gamma)$.*

In other words, by switching one coordinate (corresponding to i) of α to that of β, the objective function f decreases strictly.

Proof. We prove the lemma by induction on $|\Delta(\alpha, \beta)|$. The base case $|\Delta(\alpha, \beta)| = 1$ is trivial.

We consider the inductive step with $|\Delta(\alpha, \beta)| = q$, for some $q \geq 2$. Given a list S of coordinates from $\Delta(\alpha, \beta)$, we use $\alpha_{[S]}$ to denote the resulting vector obtained from switching coordinates S of α to those of β.

For contradiction's sake, we assume that for all $j \in \Delta(\alpha, \beta)$, $f(\alpha_{[j]}) \geq f(\alpha)$; moreover, we pick $i \in \Delta(\alpha, \beta)$ such that $f(\alpha_{[i]})$ is minimized.

Observe that $f(\alpha_{[i]}) \geq f(\alpha) > f(\beta)$ and $|\Delta(\alpha_{[i]}, \beta)| = q - 1$. Therefore, by the induction hypothesis, there exists some $k \in \Delta(\alpha_{[i]}, \beta)$ such that $f(\alpha_{[i]}) > f(\alpha_{[i,k]})$.

Next, starting from α, we shall fix all coordinates in V except i and k, and we write the objective $f(x, y)$ as a function on only these two coordinates.

Observe that we have already assumed that

$$f(\alpha_i, \alpha_k) \leq \min\{f(\beta_i, \alpha_k), f(\alpha_i, \beta_k)\}. \tag{11}$$

Moreover, from above, we have $f(\alpha_{[i]}) > f(\alpha_{[i,k]})$, which translates to $f(\beta_i, \alpha_k) > f(\beta_i, \beta_k)$. Observe that we must have $f(\alpha_i, \beta_k) \leq f(\beta_i, \beta_k)$; otherwise, we have $f(\beta_i, \beta_k) < \min\{f(\beta_i, \alpha_k), f(\alpha_i, \beta_k)\}$, which, together with (11), will contradict Lemma 3.8.

Therefore, we have $f(\alpha_{[k]}) = f(\alpha_i, \beta_k) \le f(\beta_i, \beta_k) < f(\beta_i, \alpha_k) = f(\alpha_{[i]})$, which contradicts the choice of $i \in \Delta(\alpha, \beta)$ to minimize $f(\alpha_{[i]})$. This completes the inductive step and also the proof of the lemma. □

Corollary 3.10. *For the function f in Definition 2.1 every local minimizer is a global minimizer.*

Proof. Suppose that α is a local minimizer, but there is some β with $f(\alpha) > f(\beta)$. Then, Lemma 3.9 implies there is some γ with $|\Delta(\alpha, \gamma)| = 1$ such that $f(\alpha) > f(\gamma)$, contradicting that α is a local minimizer. □

4 EFFICIENT LOCAL SEARCH

In Section 3, we conclude in Corollary 3.4 that it suffices to consider the extreme points of the search space of resistance vectors. Moreover, Corollary 3.10 states that every local minimizer is a global minimizer. Since we know how to compute the sign of the partial derivative with respect to each coordinate using Lemma 3.2, we can design a simple local search algorithm to find a global minimizer.

However, it is possible that $O(2^n)$ extreme points are encountered before a global minimizer is reached. Fortunately, in this section, we will explore further properties of the objective function, and design a local search algorithm that encounters at most $O(n)$ extreme points before finding a global minimizer.

4.1 Irrevocable Updates

Local Search Strategy. We shall start with the upper bound resistance vector, i.e., for each $i \in V$, $\alpha_i = u_i$. This induces the corresponding equilibrium opinion vector $z(\alpha)$. According to Lemma 3.2, if there is some agent i such that $\alpha_i = u_i$ and $s_i - z_i(\alpha) > 0$, then we should flip α_i to the lower bound l_i. The following lemma shows that each α_i will be flipped at most once. Essentially, we show that we will never encounter the situation that there is some agent k such that $\alpha_k = l_k$ and $s_k - z_k(\alpha) < 0$, in which case we would have to switch α_k back to u_k.

Lemma 4.1 (Each Coordinate Flipped at Most Once). *Starting from the upper bound resistance vector, the above local search strategy flips each α_i at most once.*

Proof. We first show that for each agent $k \in V$, the quantity $s_k - z_k(\alpha)$ cannot decrease when α is modified according to the local search strategy. According to the strategy, α is modified because there is some agent i such that $\alpha_i = u_i$ and $s_i - z_i(\alpha) > 0$. By Lemma 3.2, $\frac{\partial z_k}{\partial \alpha_i} > 0$ for each $k \in V$. Hence, after α_i is switched from u_i to l_i, $z_k(\alpha)$ decreases, and the quantity $s_k - z_k(\alpha)$ increases.

Observe that if a coordinate α_k is ever flipped from u_k to l_k, this means that at that moment, we must have $s_k - z_k(\alpha) > 0$, which, as we have just shown, will stay positive after α is subsequently updated. □

4.2 Approximating the Equilibrium Vector

Observe that in our local search algorithm, we need to compute the equilibrium opinion vector $z(\alpha) = [I - (I - A)P)]^{-1}As$ for the current resistance vector α, where $A = \mathrm{Diag}(\alpha)$. However, computing matrix inverse is an expensive operation. Instead, we approximate $z(\alpha)$ using the recurrence $z^{(0)} \in [0,1]^V$ and $z^{(t+1)} := As + (I - A)Pz^{(t)}$. The following lemma gives an upper bound on the additive error for each coordinate.

Lemma 4.2 (Approximation Error). *Suppose for some $\epsilon > 0$, for all $i \in V$, $\alpha_i \ge \epsilon$. Then, for every $t \ge 0$, $\|z(\alpha) - z^{(t)}\|_\infty \le \frac{(1-\epsilon)^t}{\epsilon}$.*

Proof. Using the Neumann series $[I - (I - A)P)]^{-1} = \sum_{j=0}^{\infty}[(I - A)P]^j$, we have $z(\alpha) - z^{(t)} = \sum_{j=t}^{\infty}[(I - A)P]^j As - [(I - A)P]^t z^{(0)}$.

We next prove, by induction, that for any $x \in [0,1]^V$, $\|[(I - A)P]^j x\|_\infty \le (1 - \epsilon)^j$, for all $j \ge 0$. The base case $j = 0$ is trivial because every coordinate of x is between 0 and 1. For the inductive step, assume that for some $j \ge 0$, every coordinate of $y = [(I - A)P]^j x$ has magnitude at most $(1 - \epsilon)^i$. Since P is a row stochastic matrix, it follows that $\|Py\|_\infty \le (1 - \epsilon)^j$; finally, since $\alpha_i \ge \epsilon$ for all $i \in V$, we have $\|(I - A)Py\|_\infty \le (1 - \epsilon)^{j+1}$, completing the induction proof.

Finally, observing that both $\sum_{j=t}^{\infty}[(I - A)P]^j As$ and $[(I - A)P]^t z^{(0)}$ have non-negative coordinates, we have

$$\|z(\alpha) - z^{(t)}\|_\infty \le \max\{\|\sum_{j=t}^{\infty}[(I - A)P]^j As\|, \|[(I - A)P]^t z^{(0)}\|\}$$
$$\le \sum_{j=t}^{\infty}(1 - \epsilon)^j = \frac{(1 - \epsilon)^t}{\epsilon}$$

as required. □

4.3 Local Search Algorithm

Based on Lemmas 4.1 and 4.2, we give a local search framework in Algorithm 1. Observe that in line 1, we perturb the innate opinions s slightly to ensure that for each resistance vector α encountered, no coordinate of s and $z(\alpha)$ would coincide.

The **while** loop in line 4 combines local search to update α and estimation of the equilibrium vector $z(\alpha)$. Here are two general update strategies, which are both captured by the non-deterministic step in line 7:

- **Conservative Update.** The opinion vector z is iteratively updated in line 5 until all coordinates of z and s are sufficiently far apart. Then, for every coordinate α_i such that $\alpha_i = u_i$ and $z_i < s_i$, we flip α_i to the lower bound l_i.
  After we update α, we reset t to 0, and continue to iteratively update z. Whenever we update α and set t to 0, we say that a new *phase* begins; we use the convention that the initial phase is known as phase 0.
- **Opportunistic Update.** Instead of waiting for the approximation error of every coordinate of z to be small enough, we can update some coordinates α_i, if $\alpha_i = u_i$ and $z_i \le s_i - \mathrm{err}(t)$ is small enough. However, there is some tradeoff between waiting for the errors of all coordinates to be small enough and updating coordinates of α that are ready sooner. In Section 5, we will evaluate empirically different update strategies.

Optimistic Update. In both conservative and opportunistic updates, a coordinate α_i is flipped only when we know for sure that the current estimate z_i has small enough error with respect to the equilibrium $z_i(\alpha)$; hence, no mistake in flipping any α_i is ever made.

Algorithm 1: Local Search Framework

Input: Innate opinions $s \in [0,1]^V$; interaction matrix P; for each agent $i \in V$, upper u_i and lower l_i bounds for resistance.
Output: Optimal resistance vector $\alpha \in \times_{i \in V}\{l_i, u_i\}$.

1 *(Technical step.)* Randomly perturb each coordinate of s slightly.
2 Initially, for each agent i, set $\alpha_i \leftarrow u_i$ to its upper bound; denote $\epsilon_\alpha := \min_{i \in V} \alpha_i$.
3 Pick arbitrary $z \in [0,1]^V$, and set $t \leftarrow 0$; denote $\mathrm{err}(t) := \frac{(1-\epsilon_\alpha)^t}{\epsilon_\alpha}$.
4 **while** $\exists i \in V : |s_i - z_i| \le \mathrm{err}(t) \vee (z_i < s_i \wedge \alpha_i = u_i)$ **do**
5 $z \leftarrow As + (I - A)Pz$, where $A = \mathrm{Diag}(\alpha)$
6 $t \leftarrow t + 1$
7 *(Non-deterministic step.)* Pick arbitrary $L \subseteq V$ (that can be empty) such that for each $i \in L$, $z_i \le s_i - \mathrm{err}(t)$ and $\alpha_i = u_i$.
8 **if** $L \neq \emptyset$ **then**
9 **for** *each* $i \in L$ **do**
10 Set $\alpha_i \leftarrow l_i$ to its lower bound (and update ϵ_α).
11 $t \leftarrow 0$
12 **return** *Resistance vector* α.

Algorithm 2: Optimistic Local Search

Input: Innate opinions $s \in [0,1]^V$; interaction matrix P; for each agent $i \in V$, upper u_i and lower l_i bounds for resistance.
Output: Optimal resistance vector $\alpha \in \times_{i \in V}\{l_i, u_i\}$.

1 *(Technical step.)* Randomly perturb each coordinate of s slightly.
2 Initially, for each agent i, set $\alpha_i \leftarrow u_i$ to its upper bound; denote $\epsilon_\alpha := \min_{i \in V} \alpha_i$.
3 Pick $z = (1, 1, \ldots, 1)$, and set $t \leftarrow 0$; denote $\mathrm{err}(t) := \frac{(1-\epsilon_\alpha)^t}{\epsilon_\alpha}$.
4 **while** $\exists i \in V : |s_i - z_i| \le \mathrm{err}(t)$ **do**
5 $z \leftarrow As + (I - A)Pz$, where $A = \mathrm{Diag}(\alpha)$
6 $t \leftarrow t + 1$
7 *(Optimistic Candidates.)* Set $L \leftarrow \{i \in V : z_i \le s_i \wedge \alpha_i = u_i\}$.
8 *(Rare Mistakes.)* Set $J \leftarrow \{i \in V : z_i > s_i \wedge \alpha_i = l_i\}$.
9 **if** $L \cup J \neq \emptyset$ **then**
10 **for** *each* $i \in L$ **do**
11 Set $\alpha_i \leftarrow l_i$ to its lower bound (and update ϵ_α).
12 **for** *each* $i \in J$ **do**
13 Set $\alpha_i \leftarrow u_i$ to its upper bound (and update ϵ_α).
14 $t \leftarrow 0$
15 **return** *Resistance vector* α.

However, our insight is that as the algorithm proceeds, the general trend is for every z_i to decrease.

The first intuition is that if we start with some $z^{(0)}$ such that every coordinate of $z^{(0)}$ is at least its equilibrium coordinate of $z(\alpha)$, then $z^{(t)}$ should converge to $z(\alpha)$ from above. The second observation is that every time we flip some α_i, this will not increase any coordinate of the equilibrium vector $z(\alpha)$, thereby preserving the condition that the current estimate $z^{(t)} \ge z(\alpha)$. Hence, without worrying about the accuracy of the current estimate z, we will simply flip coordinate α_i to l_i when z_i drops below s_i. However, it is possible that we might need to flip α_i back to u_i, if z_i increases in the next iteration and becomes larger than s_i again. We shall see in Section 5 that this scenario is extremely rare. Specifically, in line 8 of Algorithm 2, the set J is (almost) always empty.

5 EXPERIMENTS

Experimental setup. Our experiments run on a server with 2.1 GHz Intel Xeon Gold 6152 CPU and 64GB of main memory. The server is limited to activate at most 24 threads by the administrator. The real network topologies we use in our experiment are shown in Table 1; we interpret each network as an undirected graph. The number n of nodes in the dataset networks ranges from about 1 million to 65 million; in each network, the number m of edges is around $2n$ to $30n$.

Input Generation. For each dataset, we utilize the network topology and generate the input parameters as follows. The innate opinion s_i of each agent i is independently generated uniformly at random from $[0,1]$. For each edge $\{i,j\}$ in the network, we independently pick w_{ij} uniformly at random from $[0,1]$; otherwise, we set $w_{ij} = 0$. For $(i,j) \in V \times V$, we normalize $P_{ij} := \frac{w_{ij}}{\sum_{k \in V} w_{ik}}$. From Lemma 4.2, one can see that approximating the equilibrium opinions is more difficult when the resistance is low. However, since we still want to demonstrate that the resistance for each agent can have varied lower and upper bounds, we set the lower bound l_i of each agent i independently such that with probability 0.99, l_i equals 0.001, and with probability 0.01, it is picked uniformly at random from $[0.001, 0.1]$. Similarly, each upper bound u_i is independently selected such that with probability 0.99, it equals to 0.999, and with probability 0.01, it is chosen uniformly at random from $[0.9, 0.999]$.

5.1 Update Strategies Comparison

We compare the following three update strategies described in Section 4: conservative, opportunistic and optimistic. For the three smaller networks (com-Youtube, com-LiveJournal, LiveJournal), we apply all three update strategies. For the largest network (com-Friendster), we only report the performance of the optimistic update strategy, as the other two update strategies are not efficient enough for such a large dataset.

Experimental Setup. For fair comparison among the update strategies, we always initialize $z = (1, 1, \ldots, 1)$. To compare their performances, we plot a curve for each strategy. The curves have a common x-axis, which corresponds to the number of times that the vector z has been updated so far, i.e., the number of times line 5 (in both Algorithms 1 and 2) has been executed. Since line 5 is the most time-consuming part of the algorithms, it will be a suitable common reference. We use the term *iteration* to refer to each time z is updated. For each update strategy, we explain what is plotted for the y-axis.

- **Conservative Strategy.** We run Algorithm 1 such that in line 7, L is non-empty only if for all $i \in V$, $|s_i - z_i| > \mathrm{err}(t)$, in which case, we pick L to be the collection of all i's such that $\alpha_i = u_i$ and $z_i \le s_i - \mathrm{err}(t)$.
  For the y-axis, we plot the ratio of agents i for which currently $\alpha_i = l_i$, or we know definitely that α_i should be switched to l_i, i.e., currently $\alpha_i = u_i$ and $z_i \le s_i - \mathrm{err}(t)$.
  In Algorithm 1, the iterations (referring to each time z is updated) are grouped into *phases*, where a non-empty L in an iteration marks the end of a phase. Observe that at the end of a phase, for all $i \in L$, α_i is set to l_i and t is reset to 0.

Table 1: Datasets Information

Name	Number n of Nodes	Number m of Edges	Description	Source
com-Youtube	1,134,890	2,987,624	Youtube online social network	[18]
com-LiveJournal	3,997,962	34,681,189	LiveJournal online social network	[18]
LiveJournal	10,690,276	112,307,385	LiveJournal user and group membership network	[17]
com-Friendster	65,608,366	1,806,067,135	Friendster online social network	[18]

Hence, in the next iteration, no coordinate α_i is certain to be switched. Hence, the curve has a step-like shape, where each plateau occurs after the end of each phase.

Observer that initially $\epsilon_\alpha = \min_i u_i \geq 0.9$. Hence, it takes very few number of iterations to satisfy $\forall i \in V : |s_i - z_i| > \text{err}(t)$; we call this the phase 0. At the end of the phase 0, we set some $\alpha_i = l_i$ and ϵ_α decreases significantly. Hence, subsequent phases have many more iterations.

Observe that we can stop the iterative process, when for all $i \in V$, $|s_i - z_i| > \text{err}(t)$, but there is no $i \in V$ such that $z_i < s_i$ and $\alpha_i = u_i$. This marks the end of the curve.

In each phase, we pick L of line 7 as the collection of all i's such that $z_i \leq s_i - \text{err}(t)$ only when $\forall i \in V : |s_i - z_i| > \text{err}(t)$ (otherwise, we pick $L = \emptyset$). Then, we set $\alpha_i = l_i$ for each $i \in L$ and $t = 0$. We call such a phase a *conservative phase*.

- **Opportunistic Strategy.** We run Algorithm 1 similarly as before. Phase 0 is the same as the conservative strategy; we call a phase *conservative*, if it follows the conservative update strategy.

  Starting from phase 1, we can perform it in an opportunistic manner as follows. Recall that at the beginning of a phase, t has just been reset to 0. At the t-th iteration of that phase, we use $L(t)$ to denote the collection of i's such that $z_i \leq s_i - \text{err}(t)$. For every 1000 iterations, we compute an estimate $\text{slope}(k) := \frac{|L(1000k)| - |L(1000(k-1))|}{1000}$ of the slope; we keep track k_m of the maximum slope computed so far. After some iteration, if the estimated slope drops below some factor (we use 0.1 in our experiments) of k_m, then we end this phase. Intuitively, each additional iteration flips only a small number of coordinates α_i, and hence, one would like to end this phase. We call such a phase *opportunistic*.

  Since typically the total number of phases is around 8, we run phase 1 to 6 opportunistically, after which we run the remaining phases conservatively to make sure that all coordinates α_i that need to be changed will be flipped.

  As in the conservative update strategy, for the y-axis, we plot the ratio of coordinates α_i that currently $\alpha_i = l_i$, or we know for sure should be switched to l_i, i.e. $\alpha_i = u_i$ and $z_i \leq s_i - \text{err}(t)$.
- **Optimistic Strategy.** We implement Algorithm 2, where in each iteration after z is updated, a coordinate α_i is (re)set to l_i if $z_i < s_i$, and (re)set to u_i if $z_i > s_i$. For the y-axis, we plot the ratio of coordinates that currently take their lower bounds. The curve ends when enough iterations are performed after some coordinate of α is last updated, in order to ensure that the estimate z is close enough to the equilibrium vector according to Lemma 4.2.

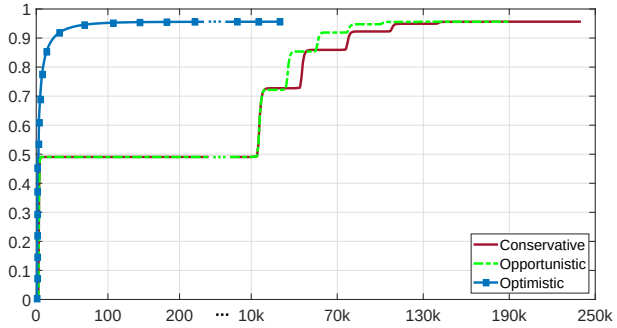


Figure 2: Update Strategies Comparison on com-Youtube.

Experiment Results. Each of Figures 2, 3 and 4 shows the plots for the three strategies in the corresponding network (com-Youtube, com-LiveJournal or LiveJournal). Figure 5 shows the plot of the optimistic strategy in the com-Friendster network, where the other two strategies are not efficient enough for such a large network. As expected, the opportunistic strategy is slightly better than the conservative strategy. From the positions of the plateaus, we can see that the initial opportunistic phases end sooner than their conservative counterparts. Hence, overall the opportunistic strategy performs slightly better than the conservative strategy; in increasing sizes of the three tested networks, the numbers of iterations taken by the opportunistic strategy are 79.2%, 77.9% and 71.5%, respectively, of those taken by the conservative strategy.

On the other hand, the optimistic strategy can achieve the optimal resistance vector with much fewer number of iterations than the other two strategies. In increasing sizes of the three smaller networks, the numbers of iterations taken by the optimistic strategy are only 12.8%, 13.4% and 12.4%, respectively, of those taken by the conservative strategy. Moreover, the optimistic strategy makes very few mistakes; in increasing sizes of the four networks, the numbers of times coordinates are flipped from lower bounds back to upper bounds are 1, 0, 13 and 168, which are negligible for networks with millions of nodes.

5.2 Running Time with Multiple Threads

We compare the actual running time using different number of threads for the optimistic strategy on only the three smaller networks, since the largest network takes too long using only one thread. Using all 24 available threads, running the optimistic strategy on the com-Friendster network already takes around 50 hours.

The three bar graphs in Figure 6 show the running time (measured in minutes) for running the optimistic strategy with different

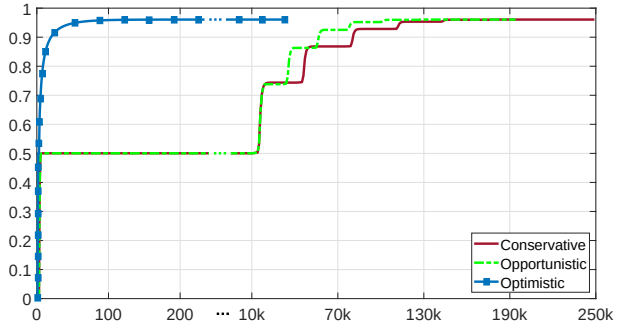


Figure 3: Update Strategies Comparison on com-LiveJournal.

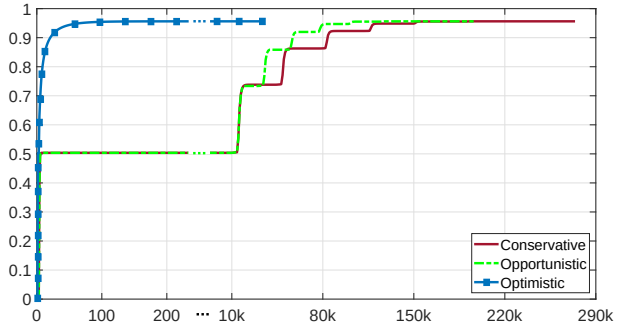


Figure 4: Update Strategies Comparison on LiveJournal.

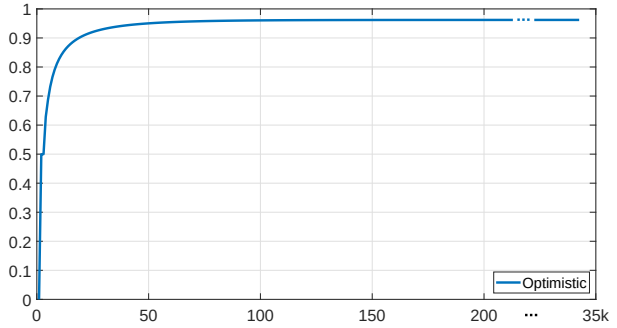


Figure 5: Optimistic Strategy on com-Friendster.

number of threads on the com-Youtube, com-LiveJournal and LiveJournal networks. Since updating z (line 5 of Algorithm 2) is the most time-consuming part of the algorithm, the fact that it is readily parallelizable supports the empirical results that using multiple threads can greatly reduce the running time, where the effect is more prominent for larger networks.

6 TECHNICAL PROOFS

Lemma 6.1. *(Lemma 3.1 restated) Given $K \subsetneq V$ and $\alpha \in (0,1)^V$, let $A := \mathrm{Diag}(\alpha)$ and recall that P is the irreducible interaction matrix. Then, the inverse $M = [I - (I - A_{-K})P]^{-1}$ exists, and every entry*

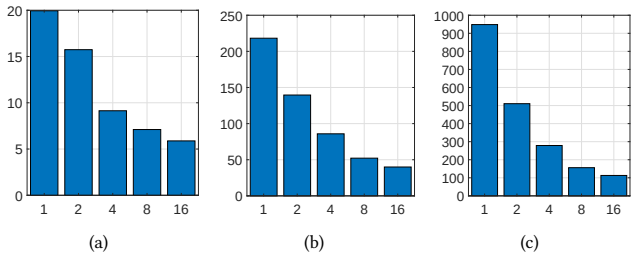


Figure 6: Running Time with Different Number of Threads on com-Youtube, com-LiveJournal and LiveJournal

of M is positive. Moreover, for each $k \in V$, define $a_k = 0$ if $k \in K$, otherwise $a_k = \alpha_k$. Then, we have:

(1) $(PM)_{kk} = \frac{M_{kk}-1}{1-a_k} > 0$;

(2) $(PM)_{kj} = \frac{M_{kj}}{1-a_k} > 0$, *for each $j \neq k$.*

Proof. Observe that P corresponds to an irreducible random walk. Hence, $(I - A_{-K})P$ represents a diluted random walk, where at the beginning of each step, the measure at nodes $i \notin K$ will suffer a factor of $1 - \alpha_i \in (0,1)$. The irreducibility of the random walk P means that every state is reachable from any state. Hence, starting from any measure vector, eventually the measure at every node will tend to 0. This means that $(I - A_{-K})P$ has eigenvalues with magnitude strictly less than 1. Therefore, we can consider the following Neumann series of a matrix:

$$M = [I - (I - A_{-K})P]^{-1} = I + \sum_{k=1}^{\infty} [(I - A_{-K})P]^k,$$

which implies that the inverse M exists, and every entry of M is positive; in particular, for every $k \in V$, $M_{kk} > 1$.

By the definition of M, we have $[I - (I - A_{-K})P]M = I$. We fix some $k \in V$. By considering the (k,k)-the entry, i.e., the dot product between the k-th row of $[I - (I - A_{-K})P]$ and the k-th column of M, we have

$$M_{kk} - \sum_{i \in V} (1 - a_k) P_{ki} M_{ik} = 1$$

where $a_k = 0$ if $k \in K$, otherwise $a_k = \alpha_k < 1$. Hence, we have $(PM)_{kk} = \sum_{i \in V} P_{ki} M_{ik} = \frac{M_{kk}-1}{1-a_k}$.

Similarly, for $j \neq k$, by considering the dot product between the k-th row of $[I - (I - A_{-K})P]$ and the j-th column of M, we have

$$M_{kj} - \sum_{i \in V} (1 - a_k) P_{ki} M_{ij} = 0.$$

Hence, we have for $j \neq k$,

$$(PM)_{kj} = \sum_{i \in V} P_{ki} M_{ij} = \frac{M_{kj}}{1 - a_k},$$

as required. □

Lemma 6.2. *(Lemma 3.3 restated) Referring to Lemma 3.2. For any $\alpha \in (0,1)^V$ and any $i \in V$, denote $M = [I - (I - A_{-\{i\}})P]^{-1}$. Then, $\frac{\partial f(\alpha)}{\partial \alpha_i}$ has the same sign in $\{-, 0, +\}$ as $s_i - \sum_{j \neq i} M_{ij} \alpha_j s_j$, which is independent of α_i.*

Proof. Using the Sherman-Morrison formula, we consider

$$\begin{aligned}X :=&[I-(I-A)P]^{-1}=[I-(I-A_{-\{i\}}-\alpha_i\mathbf{e}_i\mathbf{e}_i^T)P]^{-1}\\ =&[I-(I-A_{-\{i\}})P+\alpha_i\mathbf{e}_i\mathbf{e}_i^TP]^{-1}\\ =&M-\frac{\alpha_i}{1+\alpha_i\mathbf{e}_i^TPM\mathbf{e}_i}M\mathbf{e}_i\mathbf{e}_i^TPM.\end{aligned}$$

Observe that $\mathbf{e}_i^TPM\mathbf{e}_i=(PM)_{ii}$ and $(M\mathbf{e}_i\mathbf{e}_i^TPM)_{ij}=M_{ii}(PM)_{ij}$ for each $j\in V$. Then, by Lemma 3.1 with $K=\{i\}$, we have

$$\begin{aligned}X_{ii}=&M_{ii}-\frac{\alpha_iM_{ii}(PM)_{ii}}{1+\alpha_i(PM)_{ii}}=M_{ii}-\frac{\alpha_iM_{ii}(M_{ii}-1)}{1+\alpha_i(M_{ii}-1)}\\ =&\frac{(1-\alpha_i+\alpha_iM_{ii}-\alpha_iM_{ii}+\alpha_i)M_{ii}}{1-\alpha_i+\alpha_iM_{ii}}=\frac{M_{ii}}{1-\alpha_i+\alpha_iM_{ii}};\end{aligned}$$

and for $j\neq i$,

$$\begin{aligned}X_{ij}=&M_{ij}-\frac{\alpha_iM_{ii}(PM)_{ij}}{1+\alpha_i(PM)_{ii}}=M_{ij}-\frac{\alpha_iM_{ii}M_{ij}}{1+\alpha_i(M_{ii}-1)}\\ =&\frac{(1-\alpha_i+\alpha_iM_{ii}-\alpha_iM_{ii})M_{ij}}{1-\alpha_i+\alpha_iM_{ii}}=\frac{(1-\alpha_i)M_{ij}}{1-\alpha_i+\alpha_iM_{ii}}.\end{aligned}$$

By Lemma 3.2, we know $\frac{\partial f(\alpha)}{\partial\alpha_i}$ and $s_i-z_i(\alpha)$ have the same sign in $\{-,0,+\}$. Recall that $z(\alpha)=[I-(I-A)P]^{-1}As=XAs$. Applying the above results, we have

$$\begin{aligned}s_i-z_i(\alpha)=&s_i-\sum_{j\in V}X_{ij}\alpha_js_j\\ =&s_i-\frac{M_{ii}\alpha_is_i}{1-\alpha_i+\alpha_iM_{ii}}-\sum_{j\neq i}\frac{(1-\alpha_i)M_{ij}\alpha_js_j}{1-\alpha_i+\alpha_iM_{ii}}\\ =&\frac{s_i(1-\alpha_i+\alpha_iM_{ii})-M_{ii}\alpha_is_i-(1-\alpha_i)\sum_{j\neq i}M_{ij}\alpha_js_j}{1-\alpha_i+\alpha_iM_{ii}}\\ =&\frac{(1-\alpha_i)(s_i-\sum_{j\neq i}M_{ij}\alpha_js_j)}{1-\alpha_i+\alpha_iM_{ii}}.\end{aligned}$$

Since $\alpha_i\in(0,1)$, we conclude that $\frac{1-\alpha_i}{1-\alpha_i+\alpha_iM_{ii}}>0$. Thus $\frac{\partial f(\alpha)}{\partial\alpha_i}$, $s_i-z_i(\alpha)$ and $s_i-\sum_{j\neq i}M_{ij}\alpha_js_j$ have the same sign. □

Lemma 6.3. *(Lemma 3.6 restated) For any $i,k\in V$ such that $i\neq k$, let $M=[I-(I-A_{-\{i\}})P]^{-1}$ and $R=[I-(I-A_{-\{i,k\}})P]^{-1}$. Then for any $j\in V$, we have*

(1) $M_{jk}=\frac{R_{jk}}{1+\alpha_kR_{kk}-\alpha_k}$,

(2) $M_{jh}=R_{jh}-\frac{\alpha_kR_{jk}R_{kh}}{1+\alpha_kR_{kk}-\alpha_k}$, *for each $h\neq k$.*

In particular, the quantity in Lemma 3.3 can be rewritten as follows:

$$s_i-\sum_{j\neq i}M_{ij}\alpha_js_j=s_i-\sum_{j\neq i,k}R_{ij}\alpha_js_j-\frac{\alpha_kR_{ik}}{1+\alpha_kR_{kk}-\alpha_k}(s_k-\sum_{j\neq i,k}R_{kj}\alpha_js_j).$$

Proof. Using the Sherman-Morrison formula, we have

$$\begin{aligned}M&=[I-(I-A_{-\{i,k\}})P+\alpha_k\mathbf{e}_k\mathbf{e}_k^TP]^{-1}\\ &=R-\frac{\alpha_k}{1+\alpha_k\mathbf{e}_k^TPR\mathbf{e}_k}R\mathbf{e}_k\mathbf{e}_k^TPR\end{aligned}$$

We can compute that $\mathbf{e}_k^TPR\mathbf{e}_k=(PR)_{kk}$ and $(R\mathbf{e}_k\mathbf{e}_k^TPR)_{jh}=R_{jk}(PR)_{kh}$ for $j,h\in V$. Then we have

$$M_{jh}=R_{jh}-\frac{\alpha_kR_{jk}(PR)_{kh}}{1+\alpha_k(PR)_{kk}}.$$

By Lemma 3.1, we obtain

$$M_{jh}=R_{jh}-\frac{\alpha_kR_{jk}R_{kh}}{1+\alpha_kR_{kk}-\alpha_k}\text{ for }j,h\in V\text{ and }h\neq k,$$

and

$$M_{jk}=R_{jk}-\frac{\alpha_kR_{jk}(R_{kk}-1)}{1+\alpha_kR_{kk}-\alpha_k}=\frac{R_{jk}}{1+\alpha_kR_{kk}-\alpha_k}\text{ for }j\in V.$$

as required. □

Lemma 6.4. *(Lemma 3.7 restated) Suppose $\alpha\in(0,1)^V$, recall that $A_{-\{i,k\}}:=\mathrm{Diag}(\alpha_{-\{i,k\}})$, and P corresponds to an irreducible interaction matrix. For any $i,k\in V$ such that $i\neq k$, let $R=[I-(I-A_{-\{i,k\}})P]^{-1}$, then $R_{ii}=\max_{j\in V}R_{ji}$. Moreover, $R_{ii}=R_{ki}$ if and only if $P_{kk}+P_{ki}=1$.*

Proof. We have $[I-(I-A_{-\{i,k\}})P]R=I$. By considering the dot product between each row of $[I-(I-A_{-\{i,k\}})P]$ and column i of R, we have

$$R_{ii}-\sum_{h\in V}P_{ih}R_{hi}=1,R_{ki}-\sum_{h\in V}P_{kh}R_{hi}=0,$$

$$\text{and }R_{ji}-\sum_{h\in V}(1-\alpha_j)P_{jh}R_{hi}=0,\text{ for }j\neq i,k.$$

After rearranging, we have

$$R_{ii}=1+\sum_{h\in V}P_{ih}R_{hi},R_{ki}=\sum_{h\in V}P_{kh}R_{hi},$$

$$\text{and }R_{ji}=(1-\alpha_j)\sum_{h\in V}P_{jh}R_{hi},\text{ for }j\neq i,k.$$

Now it suffices to show that for $j\neq i,k$, the above R_{ji} cannot be the maximum among them, and $R_{ki}\leq R_{ii}$.

First, we show that R_{ji} cannot be the maximum. Since $\sum_{h\in V}P_{jh}=1$ and $\alpha_j\in(0,1)$, we have

$$R_{ji}=(1-\alpha_j)\sum_{h\in V}P_{jh}R_{hi}\leq(1-\alpha_j)\max_{h\in V}R_{hi}<\max_{h\in V}R_{hi}.$$

Thus, R_{ji} cannot be the maximum.

Next, we show that $R_{ki}\leq R_{ii}$ by contradiction. Suppose $R_{ki}>R_{ii}$, then R_{ki} is the unique maximum in the i-th column of R. Since $\sum_{h\in V}P_{kh}=1$ and $R_{ki}=\sum_{h\in V}P_{kh}R_{hi}$, it must be the case that $P_{kk}=1$. This means P corresponds to a random walk with absorbing state k, which contradicts that P is irreducible. Therefore, we have $R_{ki}\leq R_{ii}$, and hence $R_{ii}=\max_{h\in V}R_{hi}$.

Observe that we already know $R_{ji}<R_{ii}$ for $j\neq i,k$, and $R_{ki}=\sum_{h\in V}P_{kh}R_{hi}$. Hence, $R_{ki}=R_{ii}$ implies that $P_{kk}+P_{ki}=1$.

Conversely, $P_{kk}+P_{ki}=1$ implies that $R_{ki}=P_{kk}R_{ki}+P_{ki}R_{ii}$. As argued above, we must have $P_{kk}\neq1$, which implies $R_{ki}=R_{ii}$. □

7 CONCLUSION AND FUTURE WORK

We have given a solid theoretical analysis of the unbudgeted variant of the opinion susceptibility problem, and designed scalable local search algorithms that can solve the problem optimally on graphs with millions of nodes.

We believe that our techniques will lead to insights for the analysis of the budgeted variant of the problem. We leave the task of providing theoretical guarantees for greedy algorithms on the budgeted variant as future work.

REFERENCES

[1] Rediet Abebe, Jon M. Kleinberg, David C. Parkes, and Charalampos E. Tsourakakis. 2018. Opinion Dynamics with Varying Susceptibility to Persuasion. In *KDD*. ACM, 1089–1098.

[2] Joseph P. Allen, Maryfrances R. Porter, and F. Christy McFarland. 2006. Leaders and followers in adolescent close friendships: Susceptibility to peer influence as a predictor of risky behavior, friendship instability, and depression. *Development and psychopathology* 18, 1 (2006), 155–172.

[3] Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. 2016. Global Optimality of Local Search for Low Rank Matrix Recovery. In *NIPS*. 3873–3881.

[4] Morris H. DeGroot. 1974. Reaching a Consensus. *J. Amer. Statist. Assoc.* 69, 345 (1974), 118–121. http://www.jstor.org/stable/2285509

[5] T. E. Dielman, Deborah D. Kloska, Sharon L. Leech, John E. Schulenberg, and Jean T. Shope. 1992. Susceptibility to peer pressure as an explanatory variable for the differential effectiveness of an alcohol misuse prevention program in elementary schools. *Journal of School Health* 62, 6 (1992), 233–237.

[6] Pedro Domingos and Matt Richardson. 2001. Mining the Network Value of Customers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01)*. ACM, New York, NY, USA, 57–66. https://doi.org/10.1145/502512.502525

[7] William N. Evans, Wallace E. Oates, and Robert M. Schwab. 1992. Measuring peer group effects: A study of teenage behavior. *Journal of Political Economy* 100, 5 (1992), 966–991.

[8] Noah E. Friedkin and Eugene C. Johnsen. 1999. Social Influence Networks and Opinion Change. 16 (01 1999), 1–19.

[9] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. 2015. Escaping From Saddle Points - Online Stochastic Gradient for Tensor Decomposition. In *COLT (JMLR Workshop and Conference Proceedings)*, Vol. 40. JMLR.org, 797–842.

[10] Rong Ge, Jason D. Lee, and Tengyu Ma. 2016. Matrix Completion has No Spurious Local Minimum. In *NIPS*. 2973–2981.

[11] Aristides Gionis, Evimaria Terzi, and Panayiotis Tsaparas. 2013. Opinion Maximization in Social Networks. In *SDM*. SIAM, 387–395.

[12] Philip Hartman. 1959. On functions representable as a difference of convex functions. *Pacific J. Math.* 9, 3 (1959), 707–713. https://projecteuclid.org:443/euclid.pjm/1103039111

[13] Prateek Jain and Purushottam Kar. 2017. Non-convex Optimization for Machine Learning. *Foundations and Trends in Machine Learning* 10, 3-4 (2017), 142–336.

[14] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M. Kakade, and Michael I. Jordan. 2017. How to Escape Saddle Points Efficiently. In *ICML (Proceedings of Machine Learning Research)*, Vol. 70. PMLR, 1724–1732.

[15] Maurits Kaptein, Panos Markopoulos, Boris Ruyter, and Emile Aarts. 2009. Can You Be Persuaded? Individual Differences in Susceptibility to Persuasion. In *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part I (INTERACT '09)*. Springer-Verlag, Berlin, Heidelberg, 115–118. https://doi.org/10.1007/978-3-642-03655-2_13

[16] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the Spread of Influence Through a Social Network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*. ACM, New York, NY, USA, 137–146. https://doi.org/10.1145/956750.956769

[17] Jérôme Kunegis. 2017. KONECT: Konect Network Dataset. http://konect.uni-koblenz.de.

[18] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[19] Kevin M. Murphy and Andrei Shleifer. 2004. Persuasion in politics. *American Economic Review* 94, 2 (2004), 435–439.

[20] Dohyung Park, Anastasios Kyrillidis, Constantine Caramanis, and Sujay Sanghavi. 2017. Non-square matrix sensing without spurious local minima via the Burer-Monteiro approach. In *AISTATS (Proceedings of Machine Learning Research)*, Vol. 54. PMLR, 65–74.

[21] Ju Sun, Qing Qu, and John Wright. 2016. A geometric analysis of phase retrieval. In *ISIT*. IEEE, 2379–2383.

[22] Ju Sun, Qing Qu, and John Wright. 2017. Complete Dictionary Recovery Over the Sphere I: Overview and the Geometric Picture. *IEEE Trans. Information Theory* 63, 2 (2017), 853–884.

Trajectories of Blocked Community Members: Redemption, Recidivism and Departure

Jonathan P. Chang
Cornell University
jpc362@cornell.edu

Cristian Danescu-Niculescu-Mizil
Cornell University
cristian@cs.cornell.edu

ABSTRACT

Community norm violations can impair constructive communication and collaboration online. As a defense mechanism, community moderators often address such transgressions by temporarily blocking the perpetrator. Such actions, however, come with the cost of potentially alienating community members. Given this tradeoff, it is essential to understand to what extent, and in which situations, this common moderation practice is effective in reinforcing community rules.

In this work, we introduce a computational framework for studying the future behavior of blocked users on Wikipedia. After their block expires, they can take several distinct paths: they can reform and adhere to the rules, but they can also recidivate, or straight-out abandon the community. We reveal that these trajectories are tied to factors rooted both in the characteristics of the blocked individual and in whether they perceived the block to be fair and justified. Based on these insights, we formulate a series of prediction tasks aiming to determine which of these paths a user is likely to take after being blocked for their first offense, and demonstrate the feasibility of these new tasks. Overall, this work builds towards a more nuanced approach to moderation by highlighting the tradeoffs that are in play.

ACM Reference Format:
Jonathan P. Chang and Cristian Danescu-Niculescu-Mizil. 2019. Trajectories of Blocked Community Members: Redemption, Recidivism and Departure. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313638

1 INTRODUCTION

The health of online communities is often threatened by individuals that violate their norms and rules [24]. Communities have limited defense mechanisms against such offenses, ranging from the deletion of problematic comments [4], to temporarily blocking the perpetrators, to outright excluding them [22]. Such actions, however, come at the cost of potentially alienating valuable community members and raise issues of freedom of speech, fairness and bias [22, 25, 45]. Considering their potential negative impacts, it is important to understand to what extent, and in which situations, these moderation mechanisms are effective at enforcing community norms.

In this work we take a first step towards answering these questions by focusing on *temporary blocks* in the Wikipedia community of editors. Unlike other moderation actions, these have the explicitly stated purpose of "deterring any future possible repetitions of inappropriate conduct" by the perpetrators, while reintegrating them in the community (thus the temporary nature of the block). However, following their initial block, nearly half of first-time offenders either recidivate—18% of established members violate the rules again within 6 months of the block—or straight-out abandon the community—30% of established members depart within 6 months (a third of which depart during the block). This discrepancy between the stated goal and the empirical outcomes of temporary blocks motivates an investigation into the factors that determine the likely trajectory a community member will take after their block expires: redemption, recidivism or departure.

We ground our investigation in offline theories of deterrence and defiance [46], which point to at least three broad classes of factors determining future compliance: those pertaining to the characteristics of the perpetrator [48], those pertaining to the actual severity of the punishment [16, 27] and those pertaining to whether the perpetrator perceives their treatment as fair [40]. While relying on these theories to guide our selection of factors and potential confounds to consider in our analysis, we refrain from drawing any direct parallels between offline and online rule-enforcement strategies given their distinct nature and goals.

Individual characteristics. Behavior of community members and their adherence to community norms has been shown to vary according to their level of involvement with the community [6, 10, 18, 42]. We find that individuals that exhibit a high level of community involvement (e.g., have interacted with more users over a longer period of time) are less likely to abandon the community during a temporary

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.

WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313638

block. On the other hand, these individuals are also more likely to recidivate soon after the block. This suggests that users with high community involvement are more immune to the effects of blocks, both undesired (departure) and desired (reforming).

Block duration. Existing literature on offline rule compliance varies in their accounts of how the nature of the punishment determines the likelihood of recidivism [49]. The traditional (institutional) perspective is that the severity of the punishment is the main factor determining compliance [16, 27]. A contrasting (normative) perspective, however, argues that perhaps a more important factor is whether the perpetrator regards the punishment as being fair and justified [32, 48]. In the context of the Wikipedia community, we find that even though longer—and thus more severe—blocks come at the cost of higher departure rates, they do not appear to have any effect on recidivism.

Perceived fairness. To capture whether an individual perceives their initial block as fair, we leverage a mechanism specific to online moderation: block appeals. Blocked individuals can contest a block by opening a discussion with Wikipedia moderators during the block. In turn, moderators have the option to lift the block early—i.e., to *unblock* the individual—as a result of these discussions. This process provides us with a glimpse into a blocked individual's perception of fairness: we can analyze their language as well as the outcome of the appeal.

Using this methodology, we find that users who perceive the block to be unfair have an increased rate of recidivism, while those that acknowledge their wrongdoing and apologize have a decreased rate. Furthermore, even though the block duration itself does not appear to affect recidivism, the post-hoc reduction of this duration by a moderator drastically diminishes recidivism rates, perhaps by improving the user's perception of the moderation system as being fair and reasonable. Combined, these results suggest that the effectiveness of temporary blocks is not so much dependent on their actual severity, but rather on the blocked user's perception of how appropriate the block is.

To summarize, in this work we:

- investigate the effectiveness of temporary blocks by considering possible trajectories an individual can take after their block expires;
- reveal factors that are indicative of these future trajectories, reflecting both the engagement patterns of the blocked individual, as well their perception of the fairness of the block;
- analyze the relative importance of each of these factors by using them in a series of forecasting tasks aiming to determine which path a user is likely to take after their first block.

More broadly, we propose a computational framework for analyzing the outcomes of a common moderation practice in online communities. In doing so, we expose the underlying tradeoffs and provide the means to further explore factors that mediate them.

2 BLOCKS ON WIKIPEDIA

While Wikipedia is primarily known as an online encyclopedia, it also plays host to a vibrant community of editors who continually write new articles and improve existing ones. To support this community, Wikipedia has a feature known as *talk pages*: special pages on which editors can discuss a particular article or Wikipedia policy, or simply unwind with casual conversation. Every Wikipedia article has an associated talk page, on which editors can discuss proposed edits to the article. Similarly, every registered user has an associated *user talk page*, where other editors can engage directly with the respective user on issues pertaining to their general behavior. This is where the social functions of the community surface: editors discuss inappropriate behavior and norm violations, ask for support, offer advice or encourage each other. In this work we use the complete conversational history between English Wikipedia editors on both article and user talk pages. With over 90 million conversations between 4 million users on 24 million talk pages, this is one of the largest collections of public conversations [20].

Like many online communities, Wikipedia has a moderation system aimed at imposing community rules. What makes Wikipedia particularly suitable for our proposed investigation is that its rules apply uniformly to the entire community and moderation is entirely transparent to the public [13]. This is unlike other large communities, such as Reddit, where rules can vary drastically between sub-communities and where aspects of the moderation remain private [4].

Blocking mechanism. Wikipedia moderators (formally referred to as administrators, though we will continue to use the term "moderator" as we are focused on their role in maintaining community norms) are elected from among community members through public elections [1, 30]. Among other privileges, moderators have the ability to block users from making any edits or comments. They can either block a user indefinitely, in cases of accounts that are clearly used for bad-faith purposes, or temporarily if they see the user as a potentially valuable member of the community that has momentarily gone awry. In fact, the stated purpose of temporary blocks is "to deter any future possible repetitions of inappropriate conduct", thus keeping the door open for reintegration after the expirations of the blocks.[1] Our focus in this work is on temporary blocks (henceforth simply called blocks), and their effectiveness in reforming blocked users.

[1] https://en.wikipedia.org/wiki/Wikipedia:Blocking_policy

Total number of blocks	104,245
Number of blocked users	72,332
Number of blocking moderators	1,706
Users with block reason in disruption subset	21,043
Users with first block in disruption subset	18,909
Users remaining after minimum activity filters	6,026

Table 1: Statistics of the blocked users dataset.

When temporarily blocking an individual, moderators are required to notify the user by posting a message on their user talk page. This notification includes the reason for the block and its duration. The duration of the block is at the discretion of the moderator, and can vary greatly—the standard deviation of block durations is 194 days. However, this high variance is largely the result of a small set of extreme outliers, and in practice most blocks are very brief—the median duration is 1 day.

For the duration of the block, the targeted user may not contribute to Wikipedia articles or make any comments on talk pages, with the sole exception of their own user talk page, where they are allowed to discuss the block.

Block reason. When moderators block a user, they must cite a reason that reflects one of the community rules. Wikipedia groups these reasons under two broad categories: *protection* and *disruption*. The former concerns behaviors that may put Wikipedia editors or Wikipedia itself at risk, such as making legal threats, releasing personal information or violating copyright law. The latter concerns the health of the community—or, as Wikipedia phrases it, maintaining a "civil, collegial atmosphere". We identify a subset of four such disruption block types relating to breaches of community norms (rather than to legal concerns specific to Wikipedia): personal attacks and incivility, harassment, edit warring, and disruptive editing. These reflect norms that can be broken even by established and valuable members of the community, unlike spam or vandalism which most often come from illegitimate, disruption-only accounts.

Blocked users dataset. We extract information about the nature of the blocks from the *block log*, a centralized and public record of all blocks that have ever happened on Wikipedia containing over 136,000 actions (including both blocks and manual block modifications) at time of collection. We automatically extract from this log the block's duration,[2] its type, its date,[3] and the involved users (blocking moderator and blocked individual). We seek to focus our analysis on actual members of the community as opposed to disruption-only accounts, because such accounts are made in bad faith and hence have no reason to reform once caught [15]. To this end, we discard accounts whose first block was not one of the four disruption block types previously described, and impose a minimum activity before the first block: at least one month and eight comments. Finally, we join the resulting block meta-data with the conversational record. Details of the resulting dataset are summarized in Table 1. We make this processed dataset publicly available[4] to encourage the study of moderation in a community where all activity is by design fully transparent and public.[5]

3 TRAJECTORIES AFTER THE BLOCK

After their block expires, a user can take several distinct trajectories: they can reform and adhere to the rules, but they can also recidivate, or straight-out abandon the community. Here we formally define these (un)desired outcomes and reveal their relative prevalence.

Throughout, we focus on the very *first* block a user receives, thus discarding any potential lingering effects of previous blocks and putting all users in our dataset on equal ground at the time of investigation. We consider a maximum future horizon of 6 months, and thus are constrained to discard users whose initial block takes place less than 6 months before the end of our dataset (June 2018) since for these we cannot know what the future holds. This is a common solution for limited horizon in longitudinal studies [10].

Departure. Wikipedia and other collaborative communities have traditionally struggled to retain their members [17]. Thus, the risk of alienating users needs to be taken into consideration when evaluating the efficacy of moderation. This is particularly true for temporary blocks, which by their very nature aim at re-integrating the targeted user into the community rather than discouraging them to contribute again (for which a moderator would use a permanent block).

We consider a user to *depart* from the community at the time of their last comment on community talk pages; we use this instead of their last edit on an article as the indicator of quitting because we are specifically interested in their engagement with the community. We impose the same 6 month maximum future horizon as we did when considering recidivism. Out of all 6,026 blocked users selected for this study (Section 2), 30% depart in the 6 months following their first block, and 10% depart *during* the block itself.

To illustrate the connection between departure rates and the block itself, Figure 1 compares the probability of a user

[2]We merge consecutive blocks and account for post-hoc reduction in block duration.

[3]In our analysis we discard blocks happening in the first five years of the community, allowing for Wikipedia community norms to stabilize [13, 17].

[4]As part of ConvoKit (convokit.cornell.edu)

[5]All editors are expressly informed, during all edits, that "Work submitted to Wikipedia can be edited, used, and redistributed—by anyone", where "work" is defined as content on any Wikipedia page, talk pages included. See https://en.wikipedia.org/wiki/Wikipedia:Ownership_of_content.

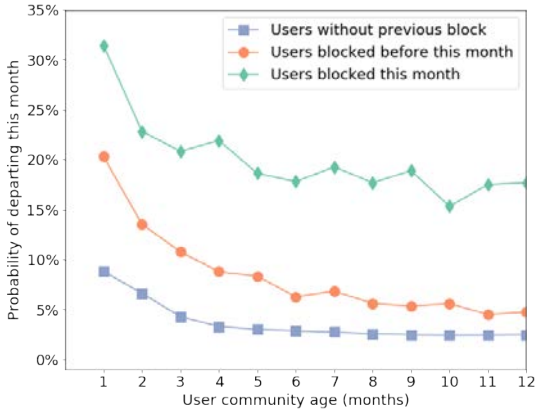


Figure 1: Probability of a user quitting at a given community age (in months), given that they were never blocked before (blue squares), blocked anytime prior to that month (orange circles) or blocked in that same month (green diamonds). The trends continue beyond the 12 months illustrated here.

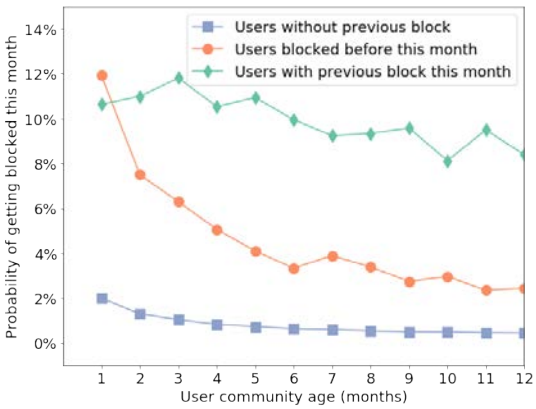


Figure 2: Probability of a user getting blocked at a given community age (in months), given that they were never blocked before (blue squares), previously blocked anytime prior to that month (orange circles) or previously blocked in that same month (green diamonds).

abandoning the community at different stages in their community life (i.e., time since their first activity), conditioning on whether (and when) they were blocked before. We first note that departure is not independent from blocking events (compare blue squares with orange circles): users are much more likely to depart if they were blocked before, especially in the first months of their life in the community. But even controlling for the fact that a user was previously blocked (compare orange circles with green diamonds), they are much more likely to depart during the month of their block. This strong recency effect shows that blocking and departure are events that are also temporally connected.

Recidivism. A previously-blocked user *recidivates* if they are blocked again for any other breach of community rules, and *reforms* if they never get blocked again. There is a wide variance in how long it takes recidivst users to re-offend. While some get blocked again within a week, others remain "clean" for years before re-offending. Only 47.7% of blocked users in our dataset have no (blocked) offenses after their first block. Among the users who do have a second known offense, the median time to reoffense is 32 days.

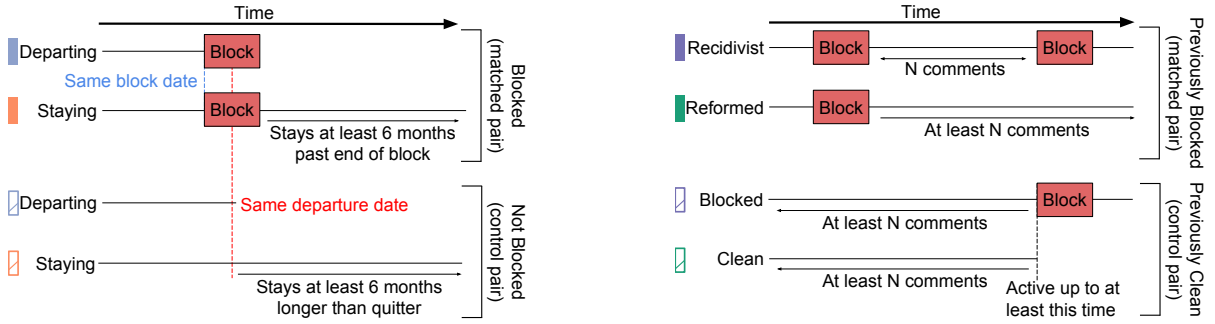
To study recidivism at different temporal scales, we consider two categories of users: *long-term recidivists* are those who re-offend within the next 6 months after their initial block, and *short-term recidivists* are the subset who re-offend within the first week after their first offense. Out of all blocked users selected for this study, 38.5% breach the norms again in the 6 months following their initial block; and 15.4% recidivate in just one week after the initial block.

To put recidivism rates into perspective and differentiate the phenomenon from general rule violations, Figure 2 compares the probability of a user getting (re)blocked at different stages in their community life, conditioned on whether (and when) they were already blocked or not. Perhaps unsurprisingly, individual block events are not independent: the probability of being blocked is much higher for individuals who were already blocked before than for those that were not, even after controlling for community age (compare blue squares with orange circles). We also observe a dramatic decrease in probability of recidivism during the first year of a user's community age. Importantly, this effect is specific to recidivism, rather than just reflecting a tendency to get blocked in general—the probability of getting blocked for the first time (blue squares) stays relatively flat over time. Finally, as in the case of departure, we also observe a recency effect (compare green diamonds with orange circles), showing that consecutive blocks are temporally related events.

Having established a relation between blocks and departure or recidivism, we now move towards a more thorough analysis of the factors that are in play. We will explore two broad classes of factors inspired by theories of deterrence and defiance [46]. The first class of factors concerns the characteristics of the perpetrator (Section 4): some types of individuals might simply be more likely to (re)offend or depart. The second class of factors concerns the actual context of the block: its actual severity and how the perpetrator perceives it (Section 5).

4 USER CHARACTERISTICS

We have already seen that community age is an important factor mediating the likelihood of departure and (re)offense: overall, newcomers are more likely than experienced users to follow one of these undesired paths after being temporarily blocked. Here we explore how these outcomes relate to



(a) Departure matching and control

(b) Recidivism matching and control

Figure 3: We matched comparable blocked users with opposite outcomes: (a) departs or not during the first block; (b) offends or not after the first block. For each matched pair, we also select a control pair of users (with the same opposite outcomes) who were not blocked.

other user characteristics, focusing on factors indicating a member's level and type of engagement in the community.

We consider two broad types of features: how active the user is and how broad their social connections are in the community. To foreshadow our findings, we find that individuals that are more active in the community are overall more immune to blocks: they are both less likely to depart during a block, but also less likely to reform. However, the way an individual spreads their activity is also informative: if their activity is directed at a broader set of users they are more likely to either depart or reform.

Activity level. We consider both the number of comments a user contributes to other users' talkpages (contributed activity), as well as the number of comments they receive from other users on their own talkpage (received activity) before the block. To disentangle a user's activity from their community age, we normalize these measures by the number of days between the user's first recorded activity in the community and the time of the block.

Activity spread. An active individual can either engage a broad audience, or instead concentrate on a narrow set of individuals. We capture the *received activity spread* of an individual by counting the number of unique users who wrote comments on their talk page before the block, then normalizing it by the total number of such comments. A higher value of this ratio indicates brief interactions with a relatively large number of peers, while conversely, a smaller ratio indicates extended interactions with a relatively small group of peers. Analogously, we define *contributed activity spread* as the ratio between the number of other users' talk pages to which an individual contributed comments and the number of such comments they contributed.

Departure. To make meaningful comparisons between users who depart *during* their first block and those who don't, hence eliminating trivial confounds, we use a matching technique inspired by causal inference methods [43]. As illustrated in Figure 3a (top), we pair each departing user with a user whose first block took place on the same date (within 1% tolerance) but who stayed active for at least 6 more months (our maximum future horizon). This procedure results in 1,156 paired departing and staying users.

As a secondary question, we also want to understand whether any differences we observe are specific to blocked individuals, or generally characterize community departure. To this end, for each matched pair of blocked departing-staying individuals, we construct a control pair of departing-staying individuals that were not blocked. This control pair consists of a non-blocked departing individual with the same departure date (within tolerance) as the blocked departing individual, and a staying individual who remained active at least 6 months past that date. This process is also illustrated in Figure 3a (bottom).

Figure 4 summarizes how our engagement factors relate to departure during the block (solid bars). Departed users receive and contribute less daily activity in their life before the block than non-departed users. The activity they do receive comes from a broader set of users, potentially signaling they are not integrated into a tight social group.[6] This is however not true for the spread of the activity they contribute, which is similar between departing and staying users.

We note that these relations between activity and departure are specific to blocked individuals (compare hashed bars in Figure 4). Departing users in our matched pairs do not have significantly different activity levels or activity spreads than their staying counterparts. This finding reinforces the

[6]These trends continue to hold, and remain significant, if we consider only the top half of active users in our comparison, thus separating them from the previously described trends in raw activity counts.

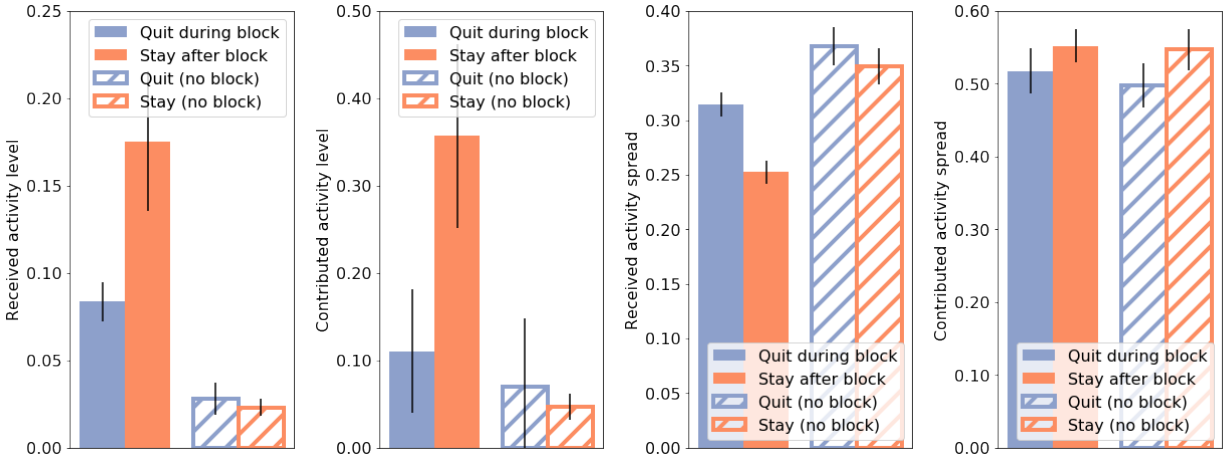


Figure 4: Difference between the community engagement patterns of departing and staying users. From left to right: receiving activity level, contributed activity level, receiving activity spread, contributed activity spread. Solid bars compare blocked departing and staying users; hashed bars compare non-blocked departing and staying users. Throughout errorbars indicate 95% binomial proportion confidence intervals.

notion that departure during a block is a distinct phenomenon from departure in general.

Recidivism. Having shown that activity level and activity spread are indicative of departure, we now explore whether they might also signal recidivism. As before, we use matching to eliminate trivial confounds. In this setting, the primary concern is that higher activity might trivially result in higher likelihood of recidivism merely because users with more activity tend to write more comments, which means they have more things they could be blocked for, probabilistically speaking. We therefore match each recidivist user by counting the number of actions (new comments, comment edits, or comment deletions) they performed between consecutive blocks, and choosing a reformed user who made at least that many actions in their lifetime after their first (and only) block; this is visualized in Figure 3b (top). This ensures that both users had at least as many opportunities to be blocked, so any differences that remain must result from the quality, not quantity, of their actions. This pairing results in 1,332 paired short-term recidivists and reformed users and 3,278 paired long-term recidivists and reformed users.

As in the departure analysis, we also find a set of control pairs to help us understand whether any differences we observe are specific to recidivism, or apply generally to norm violations. Each control pair contains a first-time offender and a user who is completely clean up to the time of the first-time offender's block. Selection is done on number of actions, just like the pairing of the experimental group. This procedure is visualized in Figure 3b (bottom).

Figure 5 summarizes how our engagement factors relate to recidivism and to blocking in general. For space considerations, we only present results for long-term recidivism; short term recidivism shows the same trends and statistical significance levels. Recidivist users receive and contribute more daily comments, but the received activity comes from a narrower set of users (solid bars). Taken together, these indicate that recidivist users' interaction patterns have more depth than breadth; they engage with a smaller set of users but each engagement is more prolonged. This may signal the presence of sustained back-and-forth with a restricted circle of editors. These engagement patterns echo general blocking patterns (hashed bars): unsurprisingly, the type of individuals that are likely to get blocked in general are also more likely to recidivate (although this was not the case for departure). In the following sections we will complement this general observation by exploring factors that relate to the context of the first block, and thus are specific to recidivism.

Predicting future trajectory. To estimate the relative predictive power of these user characteristics factors and compare them with natural baselines we formulate three new *future* prediction tasks: one for departure, one for long-term recidivism, and one for short-term recidivism. Each task is run on the respective paired sets of users previously described, resulting in a balanced prediction task.

For transparency, we train a simple SVM classifier with our activity features as input, employing both the normalized and unnormalized versions of the activity level features. In each prediction task, hyperparameters for the SVM are selected through iteration over different combinations on a held-out

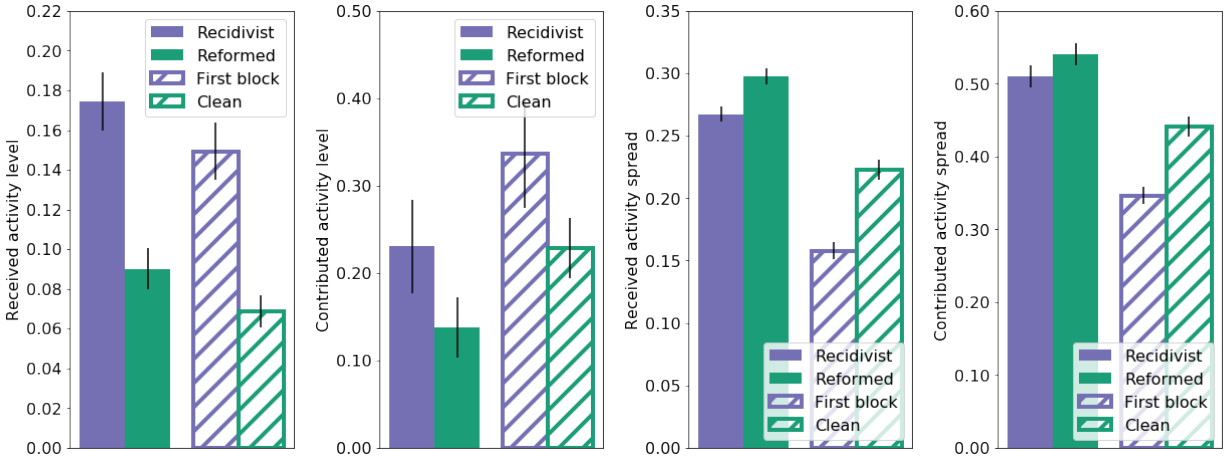


Figure 5: Difference between the community engagement patterns of recidivist and reformed users (solid bars). From left to right: receiving activity level, contributed activity level, receiving activity spread, contributed activity spread. For reference, hashed bars compare users that get blocked for the first time with those that never get blocked.

Feature set	Departure	Long-term recidivism	Short-term recidivism
Baseline: reason	59.0%	52.9%	51.9%
Baseline: duration	56.7%	50.0%	43.8%
Community age	58.6%	58.7%*	56.3%*
Engagement patterns	61.4%*	59.0%*	59.1%*
Engagement + Age	66.2%*	60.6%*	58.8%*

Table 2: Accuracies on three post-block trajectory prediction tasks (random baseline is 50%). *s indicate results that are significantly outperforming the best performing baseline for the respective task ($p < 0.05$).

development set, and accuracy is reported using leave-one-out cross-validation on the rest of the data.

We consider as features the user characteristics introduced earlier, calculated based only on information available *before* the block: community age and engagement patterns (activity level and spread). We compare their predictive power to two natural baselines that indicate the type and duration of the initial block. These account for the fact that certain kinds of offenses might be inherently more or less likely to lead to departure or recidivism.

Table 2 summarizes the results of the prediction experiments. We find that both community age and engagement patterns outperform baselines in all three prediction settings, showing they encode additional information about the future trajectory of the individual. Furthermore, they also combine to outperform each individual feature, showing that they capture different aspects of the phenomenon.

Overall, these results demonstrate that forecasting the future post-block trajectory of a user is at least a feasible and well-formed task. While our main goal in formulating these prediction tasks is to better understand the relative importance of different factors, we advise against using the resulting classifiers for any applications of algorithmically-assisted moderation. Given the sensitive nature of such applications, a public-facing model would need to be thoroughly tuned to ensure fair outcomes and exhaustively studied to understand if any biases in the data are being encoded [7, 12, 54]. Understanding the sources of such potential biases in the data, and devising mechanisms to eliminate them, constitutes an important avenue for future work.

5 BLOCK CONTEXT

Having established the relation between general user characteristics and post-block trajectories, we now seek to understand how the nature and context of the block itself mediates recidivism. Deterrence and defiance theories contrast two possible factors that relate to future compliance [46]. The traditional perspective is that the more severe the punishment is, the more likely it is to deter the perpetrator from misbehaving in the future [16, 27]. More recent work, however, suggests that a factor that is at least as important is whether the perpetrator perceives the punishment to be fair and justified [40, 52]: intuitively, individuals are unlikely to follow rules that they perceive to be imposed by an unfair and antagonistic society. We explore both these perspectives in the context of online recidivism, and bring supporting evidence for the perceived fairness perspective.

Block severity

The severity of temporary blocks corresponds naturally to their duration. For the purposes of this analysis, we only consider blocks that never had their duration changed. Since block durations are highly concentrated on the shorter end of the scale, with extreme outliers being sparsely spread in a long tail, we binarize durations as short (1 day or less) and long (more than 1 day), where the cutoff of 1 day was chosen because it is the median duration. Perhaps surprisingly, we find no statistically significant difference (chi-squared test) between these two classes in terms of recidivism rates, neither for long-term blocks, nor for short-term blocks.

While block duration does not appear to impact future compliance, it does correlate with departure rates: 30% of the users receiving long blocks quit within one month after the start of the block, compared with 24% for short blocks (significant at $p < 0.001$).[7] This result holds even when controlling for the age and activity of the blocked individual.

Perceived fairness

Considering that in our online setting the actual severity of a block does not appear to impact its effectiveness, we can investigate how the way a user *perceives* their block relates to their future compliance with community rules.[8]

The main challenge in addressing questions related to user perception is capturing such a subjective notion at a sufficiently large scale. Unlike offline studies that rely on interviews with punished individuals to assess their perception of the fairness of the punishment [40], our observational data does not provide direct access to such variables. We can, however, leverage a specific Wikipedia mechanism in order to estimate them for a subset of users: block appeals.

A block appeal is a formal avenue through which a blocked user can contest the block, admit their wrongdoing, or ask for clemency. In response to an appeal, any moderator—not necessarily the one responsible for the original block—may choose to reduce the duration of the block. This happens rather rarely, with only 15% of blocked users having their block duration reduced. As described on Wikipedia's official policy page on block appeals,[9] there are two reasons a moderator may choose to grant an appeal. The most common reason is clemency: if the blocked user takes responsibility for their offense, acknowledges that they were in the wrong, and convincingly argues that they will not offend again, the reviewing moderator may grant a shortened block. The other reason is if the moderator judges the original block to be in error, e.g., due to a case of mistaken identity. However, according to the cited Wikipedia policy, the latter scenario is "extremely rare", due in part to the public nature of the moderation in Wikipedia and to the strict procedure which blocking moderators need to undergo in order to block somebody, which involves identifying a specific reason.

Linguistic signals. Block appeals provide two separate opportunities to (admittedly imperfectly) estimate how users perceive the block. The first one is through the language the blocked individual uses in their appeals. They can signal whether they perceived the block as being justifiable and acknowledge the wrongdoing or, conversely, whether they disagree with the justification for the block and contest it.

To capture such signals we consider three types of linguistic cues (Table 3) that are intuitively related to the user's perception of the fairness of the block. Users that apologize signal that they acknowledge their wrongdoing; we expect such users to perceive the block as being fair and thus to be less likely to recidivate. Conversely, users that openly and directly challenge their block, either by aggressively questioning the administrators or by explicitly claiming unjust treatment, signal that they perceive the block as unfair and thus we would expect that they are more likely to recidivate.

To capture apologizing and direct questioning we use off the shelf tools[10] that perform regular expression matching on dependency trees [9]. To capture explicit claims of unfairness we experimented with hand-building a small lexicon by examining a sample of examples from the data:[11] "unjust", "illegitimate", "illegal", "unfair", "not fair", "accuse" (and different forms), "wrongly", "falsely", "injustice", "unfounded", "allege" (and different forms), and "unwarranted".

We extract these linguistic cues from the messages recidivist and reformed users direct at moderators during their block, capturing their use in both formal and informal appeals. Figure 6 (a-c) compares the relative ratios of recidivist and reformed users among those that used a given type of fairness cue or not. Compared to a baseline ratio of 45% recidivism and 55% reform that consistently holds in all three cases, apologizing corresponds to a reduction in the rate of recidivism (chi-squared test $p < 0.05$), while direct questioning and explicitly calling for unfairness correspond to a moderate increase in the rate of recidivism ($p < 0.05$ and $p = 0.053$, respectively).

Granted appeals. A second opportunity to estimate the user's perception of a block is by analyzing *granted* appeals. When an appeal is granted, the duration of the block is shortened. This positive response from the moderators arguably signals to the user that the system is fair and that their views

[7] Rather than analyzing departure during of the block as in the previous analysis, we are constrained here to consider a fixed time window in order to account for the time a user has to depart.

[8] While for brevity we only discuss long-term recidivism, the same effects and statistically significant levels hold for short-term recidivism.

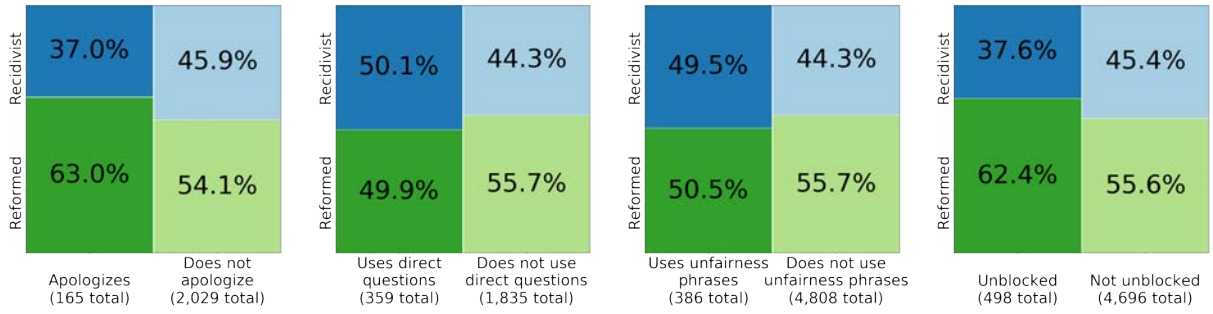
[9] https://en.wikipedia.org/wiki/Wikipedia:Appealing_a_block

[10] From ConvoKit: convokit.cornell.edu

[11] We also performed a Fightin' Words analysis [34] to identify salient cues not present in the sample data we examined.

Feature	Examples
Apologizing	It is true that the tag-war got out of hand. I **apologize** if it caused any disruption. I am deeply **sorry** for not understanding the whole situation, and ask for your forgiveness. Given the recent history on the article, **forgive me** if claims of lawyers are met with skepticism.
Direct questioning	**Why** was I blocked, when I had not the right, but rather the duty to remove BLP inaccuracies? **So what** policy, precisely have I violated? **How** is it this person won't leave me alone and I repeatedly ask to be left alone and I get blocked?
Unfairness lexicon	this block is **unjustified**. none of the changes were in violation of the 3rr rule. i have alerted another administrator about your blatent and **unwarranted** abuse of power. iv been **wrongly accused** of making up information

Table 3: Example blocked user messages exhibiting linguistic indicators of (un)fairness perception. Bolded words indicate matched patterns.



(a) Apologizing **(b) Direct questioning** **(c) Unfairness lexicon** **(d) Unblocks**

Figure 6: Normalized contingency tables showing the relative ratios of recidivist and reformed users conditioned on the presence or absence of fairness perception indicators. Table cell heights are scaled to the frequencies they represent in order to facilitate comparison between the two conditions (compare left to right in each figure).

are taken seriously. Paternoster et al. ([40], p. 169) shows that such perceptions about the enforcing authorities decrease the rate of recidivism offline, thus we hypothesize that this could hold true in online settings as well. Figure 6d shows that this principle is indeed echoed on Wikipedia; users who are unblocked are significantly more likely to reform ($p < 0.001$). This effect is even more striking when considering that, as discussed above, the duration of the block itself has no noticeable effect on recidivism and that granted appeals often shorten the duration of the block by less than one day (the median reduction is 22 hours).

Alternative explanations. One explanation that does not relate to perceived fairness could be that these appeal-based effects are simply differentiating between individuals who did not deserve to be blocked in the first place (and thus did not commit an offense and are unlikely to offend at all) and those that did commit an offense. While our observational approach precludes ruling out this alternative explanation completely, we argue that it is at least unlikely. First, as noted earlier, it is extremely rare for users to be blocked in error on Wikipedia. Second, some of the linguistic effects go in the opposite direction of what such an explanation would predict: for example, apologizing generally comes with an admittance of guilt, and under the alternative explanation we would not expect such guilty individuals to have a reduced rate of recidivism. In fact, we find that individuals who apologize are significantly ($p < 0.001$, by chi-squared test) more likely to get unblocked than ones who do not, a pattern that is consistent with what we would expect if unblocks tend to be granted because of clemency (acknowledgment of wrongdoing being one of the factors that moderators look for when deciding whether to grant an appeal).

In an attempt to further discard explanations based on user characteristics we also repeat these experiments after controlling for age and activity (i.e., pairing recidivist and reformed users based on these variables before conducting the analysis) and found similar trends, with exception of the unfairness lexicon (which had a moderate effect to start with and was likely sensitive to the sample size reduction).

We argue that these observations, while not of causal nature, at least motivate further investigations into the role of perceived fairness in online moderation.

6 FURTHER RELATED WORK

Antisocial behavior. Online moderation largely addresses the problem of antisocial behavior, which occurs in the form of harassment [50], cyberbullying [47], and general aggression [23]. Approaches to moderating such content include decentralized, community-driven methods [14], as well as top-down methods relying on designated community managers or moderators [33]. Prior research in this area ranges from understanding the actors involved in antisocial behavior [6, 28, 36, 42, 51] to analyzing its effects [39] to tools for identifying such behavior [38, 53], and even forecasting future instances [31, 55]. While inspired by this line of work, our present study extends it by focusing on what happens *after* a user is blocked for violating community rules.

Effects of moderation. Prior work has examined the effects of different kinds of moderation on online platforms. Community-driven moderation can affect overall user participation [8, 35] and comment quality [5] in conversations. Centralized moderation can also have effects on conversation, mainly through the moderator's role as an authority figure [44]. These existing studies of moderation effects have focused on the short term, largely operating at the level of individual conversations. By contrast, in this work we intend to study the long term, *user* level effects of moderation: what happens to a user in the days, weeks, and months after a moderator takes action against them?

Norms and engagement. A major factor governing engagement in online communities is a sense of belonging [37], which in many communities engenders the emergence of community-specific norms, such as specific patterns of language [3, 10, 19, 56]. Wikipedia is no exception: it relies on group dynamics to promote editing productivity [26] and quality [41], as well as participation in governance [1, 30].

Fairness in moderation. Moderation is fundamentally a difficult process, sparking discussions about design and best practices [2, 24], and moral questions regarding bias [45] and free speech [21]. One interesting avenue of research in this broad space deals with understanding the *perception* of fairness. Prior work has looked at this from the perspective of moderators, showing that moderators on the technology news forum Slashdot frequently judge each others' actions as unfair, but that this rarely leads to reversal of moderation decisions [29]. We instead explore the perspective of users, exploring ways to determine when a user regards a moderation decision as unfair.

7 DISCUSSION

We introduced a computational framework for characterizing the possible trajectory a user can take after a temporary block: departure, recidivism, or redemption. Taking inspiration from theories of offline punishment, our investigation focused on two types of factors that are tied to these outcomes: those related to a user's involvement in the community, and those related to their perception of the moderation action affecting them.[12] Overall, our work builds towards a more nuanced approach to moderation that more broadly accounts for the tradeoffs between possible outcomes and that considers how the affected individuals might perceive the moderators and their actions.

Our study has several limitations that naturally translate into opportunities for future work. First, while we have focused on the blocked users, it would be valuable to also understand the normative effect of blocks on other users that are witnessing them. Additionally, our study is limited to revealing correlational effects, and different methodologies (involving surveys or experimentation) would be necessary to study the underlying causal mechanism.

In terms of methodology, there are several ways in which the definitions of the post-block trajectories themselves could be refined. While we measure departure as a binary value, in reality community participation occurs on a variable scale and it is possible that blocked users do not depart, but drastically reduce their participation (or conversely, increase it). Similarly, defining recidivism in terms of a second block is an imperfect approach because it is possible that some users reoffend without getting blocked. This is especially true in the case of personal attacks, where each individual's threshold of acceptable behavior may vary. One possible way of addressing this would be to combine our approach with prior work in automated detection of toxic language [11, 42, 53].

We also note possible ways in which the factors we explore could be expanded upon. The activity features we introduced in Section 4 effectively serve as crude approximations of network effects. Future work could apply a more formal network analysis to uncover more powerful representations of activity. Similarly, our linguistic measures of fairness are currently limited. One immediate way of expanding them would be to build a more comprehensive lexicon of unfairness, perhaps using a semi-automated approach based on word co-occurrences with the currently existing lexicon items. A more sophisticated approach might involve engineering dependency-based features similar those used to extract apologies and direct questioning. Understanding whether a user feels they are treated unfairly constitutes an interesting research question on its own.

Acknowledgements. The authors thank Lucas Dixon, Liye Fu, Jon Kleinberg, Lillian Lee, Dario Taraborelli, Justine Zhang and Andrew Wang and the anonymous reviewers for their helpful comments. This work is supported in part by a Google Faculty Research Award, NSF CAREER award IIS-1750615 and NSF Grant SES-1741441.

[12] We refrain from drawing any explicit connections between online and offline punishment, acknowledging their different nature and goals.

REFERENCES

[1] Moira Burke and Robert Kraut. 2008. Mopping up: Modeling Wikipedia Promotion Decisions. In *Proceedings of CSCW*.

[2] Carlos Castillo, Francesco Fabbri, and Diego Saez-Trumper. 2017. Current Practices of Online Community Managers. (2017).

[3] Stevie Chancellor, Andrea Hu, and Munmun De Choudhury. 2018. Norms Matter: Contrasting Social Support Around Behavior Change in Online Weight Loss Communities. In *Proceedings of CHI*.

[4] Eshwar Chandrasekharan, Mattia Samory, Shagun Jhaver, Hunter Charvat, Amy Bruckman, Cliff Lampe, Jacob Eisenstein, and Eric Gilbert. 2018. The Internet's Hidden Rules: An Empirical Study of Reddit Norm Violations at Micro, Meso, and Macro Scales. In *Proceedings of CSCW*, Vol. 2.

[5] Justin Cheng, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2014. How Community Feedback Shapes User Behavior. In *Proceedings of ICWSM*.

[6] Justin Cheng, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2017. Antisocial Behavior in Online Discussion Communities. In *Proceedings of CSCW*.

[7] Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. 2017. Algorithmic Decision Making and the Cost of Fairness. In *Proceedings of KDD*.

[8] Tiago Oliveira Cunha, Ingmar Weber, Hamed Haddadi, and Gisele L. Pappa. 2016. The Effect of Social Feedback in a Reddit Weight Loss Community. In *Proceedings of DH*.

[9] Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A Computational Approach to Politeness with Application to Social Factors. In *Proceedings of ACL*.

[10] Cristian Danescu-Niculescu-Mizil, Robert West, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. No Country for Old Members: User Lifecycle and Linguistic Change in Online Communities. In *Proceedings of WWW*.

[11] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of ICWSM*.

[12] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In *Proceedings of KDD*.

[13] A. Forte and A. Bruckman. 2008. Scaling Consensus: Increasing Decentralization in Wikipedia Governance. In *Proceedings of HICSS*.

[14] R. Stuart Geiger. 2016. Bot-Based Collective Blocklists in Twitter: The Counterpublic Moderation of Harassment in a Networked Public Space. *Information, Communication & Society* 19, 6 (June 2016).

[15] R. Stuart Geiger and David Ribes. 2010. The Work of Sustaining Order in Wikipedia: The Banning of a Vandal. In *Proceedings of CSCW*.

[16] Harold G. Grasmick and George J. Bryjak. 1980. The Deterrent Effect of Perceived Severity of Punishment. *Social Forces* 59, 2 (Dec. 1980).

[17] Aaron Halfaker, R. Stuart Geiger, Jonathan T. Morgan, and John Riedl. 2013. The Rise and Decline of an Open Collaboration System. *American Behavioral Scientist* 57, 5 (May 2013).

[18] Aaron Halfaker, Aniket Kittur, and John Riedl. 2011. Don't Bite the Newbies: How Reverts Affect the Quantity and Quality of Wikipedia Work. In *Proceedings of WikiSym*.

[19] William L. Hamilton, Justine Zhang, Cristian Danescu-Niculescu-Mizil, Dan Jurafsky, and Jure Leskovec. 2017. Loyalty in Online Communities. In *Proceedings of ICWSM*.

[20] Yiqing Hua, Cristian Danescu-Niculescu-Mizil, Dario Taraborelli, Nithum Thain, Jeffery Sorensen, and Lucas Dixon. 2018. WikiConv: A Corpus of the Complete Conversational History of a Large Online Collaborative Community. In *Proceedings of EMNLP*.

[21] Shagun Jhaver, Larry Chan, and Amy Bruckman. 2018. The View from the Other Side: The Border between Controversial Speech and Harassment on Kotaku in Action. *First Monday* 23, 2 (Feb. 2018).

[22] Shagun Jhaver, Sucheta Ghoshal, Amy Bruckman, and Eric Gilbert. 2018. Online Harassment and Content Moderation: The Case of Blocklists. In *Proceedings of CSCW*.

[23] Joseph M. Kayany. 1998. Contexts of Uninhibited Online Behavior: Flaming in Social Newsgroups on Usenet. *Journal of the American Society for Information Science* 49, 12 (Jan. 1998).

[24] Sara Kiesler, Robert Kraut, Paul Resnick, and Aniket Kittur. 2012. Regulating Behavior in Online Communities. In *Building Successful Online Communities: Evidence-Based Social Design*. MIT Press.

[25] Gary King, Jennifer Pan, and Margaret E. Roberts. 2013. How Censorship in China Allows Government Criticism but Silences Collective Expression. *American Political Science Review* 107, 02 (May 2013).

[26] Aniket Kittur, Bryan Pendleton, and Robert E. Kraut. 2009. Herding the Cats: The Influence of Groups in Coordinating Peer Production. In *Proceedings of WikiSym*.

[27] Steven Klepper and Daniel Nagin. 1989. Tax Compliance and Perceptions of the Risks of Detection and Criminal Prosecution. *Law & Society Review* 23, 2 (1989).

[28] Srijan Kumar, William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2018. Community Interaction and Conflict on the Web.

[29] Cliff Lampe and Paul Resnick. 2004. Slash(Dot) and Burn: Distributed Moderation in a Large Online Conversation Space. In *Proceedings of CHI*.

[30] J Leskovec, D P Huttenlocher, and Kleinberg. 2010. Governance in Social Media: A Case Study of the Wikipedia Promotion Process. In *Proceedings of ICWSM*.

[31] Ping Liu, Joshua Guberman, Libby Hemphill, and Aron Culotta. 2018. Forecasting the Presence and Intensity of Hostility on Instagram Using Linguistic and Social Features. In *Proceedings of ICWSM*.

[32] Toni Makkai and John Braithwaite. 1994. Reintegrative Shaming and Compliance with Regulatory Standards. *Criminology* 32, 3 (Aug. 1994).

[33] Aiden R McGillicuddy, Jean-Gregoire Bernard, and Jocelyn Ann Cranefield. 2016. Controlling Bad Behavior in Online Communities: An Examination of Moderation Work. In *Proceedings of ICIS*.

[34] Burt L. Monroe, Michael P. Colaresi, and Kevin M. Quinn. 2008. Fightin' Words: Lexical Feature Selection and Evaluation for Identifying the Content of Political Conflict. *Political Analysis* 16, 04 (2008).

[35] Lev Muchnik, Sinan Aral, and Sean Taylor. 2013. Social Influence Bias: A Randomized Experiment. *Science* 341 (2013).

[36] Kevin Munger. 2017. Tweetment Effects on the Tweeted: Experimentally Reducing Racist Harassment. *Political Behavior* 39, 3 (Sept. 2017).

[37] Edward Newell, David Jurgens, Haji Mohammad Saleem, Hardik Vala, Jad Sassine, Caitrin Armstrong, and Derek Ruths. 2016. User Migration in Online Social Networks: A Case Study on Reddit During a Period of Community Unrest. In *Proceedings of ICWSM*.

[38] Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive Language Detection in Online User Content. In *Proceedings of WWW*.

[39] Alexandra Olteanu, Carlos Castillo, Jeremy Boy, and Kush Varshney. 2018. The Effect of Extremist Violence on Hateful Speech Online. In *Proceedings of ICWSM*.

[40] Raymond Paternoster, Robert Brame, Ronet Bachman, and Lawrence W. Sherman. 1997. Do Fair Procedures Matter? The Effect of Procedural Justice on Spouse Assault. *Law & Society Review* 31, 1 (1997).

[41] Umashanthi Pavalanathan, Xiaochuang Han, and Jacob Eisenstein. 2018. Mind Your POV: Convergence of Articles and Editors Towards Wikipedia's Neutrality Norm. In *Proceedings of CSCW*.

[42] Manoel Horta Ribeiro, Pedro H. Calais, Yuri A. Santos, Virgílio A. F. Almeida, and Wagner Meira Jr. 2018. Characterizing and Detecting Hateful Users on Twitter. In *Proceedings of ICWSM*.

[43] Paul R Rosenbaum. 2010. *Design of Observational Studies*. Springer, New York.

[44] Joseph Seering, Robert Kraut, and Laura Dabbish. 2017. Shaping Pro and Anti-Social Behavior on Twitch Through Moderation and Example-Setting. In *Proceedings of CSCW*.

[45] Qinlan Shen, Michael Miller Yoder, Yohan Jo, and Carolyn P. Rose. 2018. Perceptions of Censorship and Moderation Bias in Political Debate Forums. In *Proceedings of ICWSM*.

[46] Lawrence W. Sherman. 1993. Defiance, Deterrence, and Irrelevance: A Theory of the Criminal Sanction. *Journal of Research in Crime and Delinquency* 30, 4 (Nov. 1993).

[47] Vivek K. Singh, Marie L. Radford, Qianjia Huang, and Susan Furrer. 2017. "They Basically like Destroyed the School One Day": On Newer App Features and Cyberbullying in Schools. In *Proceedings of CSCW*.

[48] Michael Tonry. 2008. Learning from the Limitations of Deterrence Research. *Crime and Justice* 37, 1 (Jan. 2008).

[49] Tom R. Tyler. 1990. *Why People Obey the Law*. Yale University Press, New Haven.

[50] Jessica Vitak, Kalyani Chadha, Linda Steiner, and Zahra Ashktorab. 2017. Identifying Women's Experiences With and Strategies for Mitigating Negative Effects of Online Harassment. In *Proceedings of CSCW*.

[51] Svitlana Volkova and Eric Bell. 2017. Identifying Effective Signals to Predict Deleted and Suspended Accounts on Twitter across Languages. In *Proceedings of ICWSM*.

[52] Kirk R. Williams. 2005. Arrest and Intimate Partner Violence: Toward a More Complete Application of Deterrence Theory. *Aggression and Violent Behavior* 10, 6 (Sept. 2005).

[53] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex Machina: Personal Attacks Seen at Scale. In *Proceedings of WWW*.

[54] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. 2017. Fairness Beyond Disparate Treatment & Disparate Impact: Learning Classification Without Disparate Mistreatment. In *Proceedings of WWW*.

[55] Justine Zhang, Jonathan Chang, Cristian Danescu-Niculescu-Mizil, Lucas Dixon, Yiqing Hua, Dario Taraborelli, and Nithum Thain. 2018. Conversations Gone Awry: Detecting Early Signs of Conversational Failure. In *Proceedings of ACL*.

[56] Justine Zhang, William L. Hamilton, Cristian Danescu-Niculescu-Mizil, Dan Jurafsky, and Jure Leskovec. 2017. Community Identity and User Engagement in a Multi-Community Landscape. In *Proceedings of ICWSM*.

Selling a Single Item with Negative Externalities

To Regulate Production or Payments?

Tithi Chattopadhyay
Princeton University
Princeton, New Jersey
tithic@princeton.edu

Nick Feamster
Princeton University
Princeton, New Jersey
feamster@princeton.edu

Matheus V. X. Ferreira
Princeton University
Princeton, New Jersey
mvxf@cs.princeton.edu

Danny Yuxing Huang
Princeton University
Princeton, New Jersey
yuxingh@cs.princeton.edu

S. Matthew Weinberg
Princeton University
Princeton, New Jersey
smweinberg@princeton.edu

ABSTRACT

We consider the problem of regulating products with negative externalities to a third party that is neither the buyer nor the seller, but where both the buyer and seller can take steps to mitigate the externality. The motivating example to have in mind is the sale of Internet-of-Things (IoT) devices, many of which have historically been compromised for DDoS attacks that disrupted Internet-wide services such as Twitter [5, 26]. Neither the buyer (i.e., consumers) nor seller (i.e., IoT manufacturers) was known to suffer from the attack, but both have the power to expend effort to secure their devices. We consider a regulator who regulates payments (via fines if the device is compromised, or market prices directly), or the product directly via mandatory security requirements.

Both regulations come at a cost—implementing security requirements increases production costs, and the existence of fines decreases consumers' values—thereby reducing the seller's profits. The focus of this paper is to understand the *efficiency* of various regulatory policies. That is, policy A is more efficient than policy B if A more successfully minimizes negatives externalities, while both A and B reduce seller's profits equally.

We develop a simple model to capture the impact of regulatory policies on a buyer's behavior. In this model, we show that for *homogeneous* markets—where the buyer's ability to follow security practices is always high or always low—the optimal (externality-minimizing for a given profit constraint) regulatory policy need regulate *only* payments *or* production. In arbitrary markets, by contrast, we show that while the optimal policy may require regulating both aspects, there is always an approximately optimal policy which regulates just one.

CCS CONCEPTS

• **Theory of computation → Theory and algorithms for application domains**; • **Security and privacy →** *Human and societal aspects of security and privacy.*

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313692

KEYWORDS

Mechanism Design and Approximation; Auction Design; Negative Externalities; Tragedy of the Commons; Policy and Regulation.

ACM Reference Format:
Tithi Chattopadhyay, Nick Feamster, Matheus V. X. Ferreira, Danny Yuxing Huang, and S. Matthew Weinberg. 2019. Selling a Single Item with Negative Externalities: To Regulate Production or Payments?. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313692

1 INTRODUCTION

The Tragedy of the Commons is a well-documented phenomenon where agents act in their own personal interests, but their collective action brings detriments to the common good [13]. One motivating example that we will keep referencing in the paper is the sale of Internet-of-Things (IoT) devices, such as Internet-connected cameras, light bulbs, and refrigerators. Recent years have seen a proliferation of these "smart-home" devices, many of which are known to contain security vulnerabilities that have been exploited to launch high-profile attacks and disrupt Internet-wide services such as Twitter and Reddit [5, 26]. Both the owners and manufacturers of IoT devices have the ability to protect the common good (i.e., Internet-wide service for all users) from being attacked by securing their devices, but have little incentive to do so. For the manufacturers, implementing security features, such as using encryption or having no default passwords, introduces extra engineering cost [2]. Similarly, security practices, such as regularly updating the firmware or using complex and difficult-to-remember passwords, can be a costly endeavor for the consumers [9, 27]. The results of their actions cause a negative *externality*, where Internet service is disrupted for other users.

One way to reduce the negative externality is regulation. In the context of IoT sales, a regulator can, for instance, set minimum security standards for the manufacturers or impose fines on owners of hacked IoT devices that engage in attacks. Fines could come in a few forms: direct levies on the consumer, or indirect monetary incentives. For instance, ISPs could offer discounts to users whose networks have not displayed any signs of malicious activities. One might argue that such penalty-based policies could be too futuristic, but it is worth noting that similar practices are being adopted in other industries to mitigate negative externalities [3]. One example

is the levying of fines on users (such as cars and factories) that cause pollution [11]. While there are a lot of practicalities that have to be kept in mind and the decision of when to implement consumer/user fines depends on various factors, this is certainly one of the various policy alternatives that is worthwhile to study. Such regulations, however, can potentially increase the cost of production, discourage consumers from purchasing IoT devices, and reduce the manufacturer's profit. Our focus is to compare the *efficiency* of various regulatory policies: for two policies which equally hurt the seller's profits, which one better mitigates externalities? We will also be interested in understanding the *optimal* policy: the minimum security standards for the manufactures and fines on owners that together best mitigate externalities subject to a minimum seller's profit.

We first develop a model that consists of a buyer (e.g., consumers interested in purchasing IoT devices) and a single product for sale (e.g., IoT device). The product may come with some (costly to increase) level of security, c, and a consumer purchasing the device may choose to spend additional effort h to further secure the device. We consider a mechanism for regulating the market through incentives, for example, by requiring that the product being sold implement security features that cost the seller c dollars, imposing a fine of y dollars on the buyer if the product is later compromised and used in attacks, or both. Which intervention is more appropriate depends on how efficient buyers are in securing the product. The goal of the regulator is to minimize the negative externalities subject to a cap on the negative impact on the seller's profits—the idea being that any policy which too negatively impacts the seller could be unimplementable due to industry backlash.

Understanding the effects of such regulations on the behavior of a *single* consumer is relatively straightforward. For example: as fines go up, consumers adjust (upwards) the optimal level of effort to expend, lowering their total value for the item. Yet, reasoning about how an entire market of consumers will respond to changes, and how these responses impact seller profits becomes more complex.

Our contributions are as follows: (i) We model the sale of a single item with negative externalities, using the sale of IoT devices as the motivating example (Section 3). (ii) We show in Sections 5 and 6 that when the population of consumers is *homogeneous* (i.e., all consumers are comparably effective at translating effort into security) that optimal policies need only to regulate *either* the product (via minimum security standards) *or* the payments (via fines). (iii) We provide an example of non-homogeneous markets where the optimal policy regulates both product and payments, but prove that in *all* markets, it is always approximately optimal to regulate only one (Sections 7.1 and 7.2). The technical sections additionally contain numerous examples witnessing the subtleties in reasoning about these problems, and that any assumptions made in our theorem statements are necessary.

2 RELATED WORK

Auction Design with Externalities. There is ample prior work studying auction design with network externalities in the following sense: if the item for sale is a phone, then one consumer's value for the phone increases when another consumer purchases a phone as well (which is a positive externality, because they can talk to more people). Similarly, the item could be advertising space, in which case one consumer's value for advertising space could decrease as other consumers receive space (which is a negative externality, as now each unit of space is less likely to grab attention) [4, 6, 12, 15, 17, 23]. Our work differs in that it is a third party, who is neither selling nor purchasing an item, who suffers the externalities.

Improving the Commons. There is also a large body of work studying the regulation of common goods (e.g., clean air, security, spectrum access) in the form of taxes or licenses. For example, a government agency can regulate the emission of pollution by auctioning licenses (perhaps towards minimizing the total social cost—regulation cost plus negative externalities) [10, 19, 22, 24, 28, 29]. Our work differs in that our regulations are constrained to guarantee minimum profit to the seller, rather than focusing exclusively on the social good.

Approximation in Auction Design. Owing to the inherent complexity of optimal auctions for most settings of interest, it is now commonplace in the Economics and Computation community to design simple but approximately auctions. Our work too follows this paradigm. We refer the reader to previous work [16] for an overview of this literature.

Mitigating Security Problems. Computer security is a particular example of the Tragedy of the Commons, where a software or hardware provider sells an insecure product, and where consumers may purchase the product without considering or taking actions to reduce the security risks. In addition, users might be unable to distinguish insecure products from insecure ones [1]. One mitigation strategy is to have the vendor release updates with security features, although this could be a costly process, as August observes [2]. However, identifying the existence of security vulnerabilities in the first place may take time for the vendors; for instance, a common software vulnerability known as buffer overflow remained in more than 800 open-source products for a median period of two years before the vendors fixed the problems, according to a study by Li [21].

An alternative to relying on a vendor to implement security features or releasing updates is to incentivize the users to follow security practices. Redmiles has found that users who adopt security practices, like using two-factor authentication, have a lower overall utility for themselves than if they adopt no security practices at all, as security practices may introduce inconvenience [27]. Even if users were notified of security problems that they were presumably unaware of, it took as long as two weeks for fewer than 40% of the users to take remedial actions, according to a study [20]. To introduce incentives, vendors could, for instance, offer discounts to users who adopt security behaviors [2]; regulators, on the other hand, could incur fines to users whose software or devices were hacked [18], which is a part of our model in this paper.

3 MODEL

In this section, we introduce our model, which consists of a population of rational buyers and a single item for sale. After introducing each of the concepts one-by-one, we include a table (Table 1) at the end of this section to remind the reader of each of the components.

Buyer properties. Buyers in our model have two parameters: $(v, k) \in \mathbb{R}^2_+$. v denotes the buyer's value for the item (i.e., how much value does the buyer derive from the IoT device in isolation, independent of fines, etc.). k denotes the buyer's *effectiveness* in translating effort into improved security. That is, a buyer with high k can spend little effort and greatly reduce the risk of being hacked (e.g. because they are well-versed in security measures). A buyer with low k requires significant effort for minimal security gains. We will often use $t := (v, k)$ to denote a buyer's *type*.

Security. A buyer who chooses to purchase an item will spend some level of effort $h \geq 0$ securing it, which causes disutility h to the buyer. The seller may also include some default security level c. If the buyer has effectiveness k, we then denote the combined effort by $\textsc{effort}(k, c, h) := c + kh$. The idea is that buyers with higher effectiveness are more effective at securing the device for the same disutility. Note that buyers with effectiveness $k > 1$ are more effective than the producer, and buyers with effectiveness $k < 1$ are less effective. Highly effective buyers should not necessarily be interpreted as "more skilled" than producers, but some security measures (e.g., password management) are simply more effective for consumers than producers to implement.

We model the probability that a device is compromised as a function $g(\cdot)$ of $\textsc{effort}$, with $g(x) := e^{-x}$. This modeling decision is clearly stylized, and meant as an approximation to practice which captures the following two important features: (a) as effort x approaches ∞, $g(x) \to 0$ (that is, it is possible to shrink the probability of being compromised arbitrarily small with sufficient effort), and (b) $g''(x) \geq 0$. That is, the initial units of effort are more effective (i.e. $g'(x)$ is larger in absolute value) than latter ones (when $g'(x)$ is smaller in absolute value). The idea is that consumers/producers will take the highest "bang-for-buck" steps first (e.g., setting a password). Note that our results do not qualitively change if, for instance, $g(x) := \lambda_1 e^{\lambda_2 x}$ for some constants $\lambda_1 \in (0, 1], \lambda_2 > 0$, but since the model is stylized anyway we set $\lambda_1 = \lambda_2 = 1$ for simplicity of notation.

Regulatory Policy. The regulator selects a policy/strategy $s = (y, c, p) \in \in \mathbb{R}^3_+$. Here, c denotes the security standards the producer must include which is equivalent to the production cost. y denotes the fine the consumer pays should their device be compromised. p denotes the price of the item. Conceptually, one should think of the regulator inducing the producer to set security standard c and price p via particular regulatory policies (e.g., requiring a minimum security level c', or mandating purchase of insurance). Mathematically, we will not belabor exactly how the regulator arrives at (y, c, p). We will also be interested in "simple" policies, which regulate either y or c.

Definition 3.1 (Simple Policy). For a policy $s = (y, c, p)$, we say s is a *fine policy* if $c = 0$, a *cost policy* if $y = 0$ and a *simple policy* if s is either a *fine policy* or a *cost policy*.

Utilities. Recall that so far our buyer has value v and efficiency k, and chooses to put in effort h. The regulator mandates security c (which is equivalent to the production cost) on the item (which has price p) and imposes fine y for compromised items. The probability that an item is compromised is $g(\textsc{effort}(k, c, h)) = e^{-c-kh}$. The buyer's utility is therefore: $v - p - h - y \cdot e^{-c-kh}$. Observe that the buyer is in control of h (but not v, p, y, c, k). So the buyer will optimize over $h \geq 0$ to minimize $h + y \cdot e^{-c-kh}$. By taking the derivative with respect to h, we get a closed form for the choice of effort $h^*(t, s)$ (recalling that we denote the buyer's type $t = (v, k)$ and the regulator's strategy $s = (y, c, p)$):

$$h^*(t, s) = \max\left(0, \frac{\ln(yk) - c}{k}\right) \tag{1}$$

We can now see that the probability that the buyer's item is compromised, conditioned on expending the optimally chosen effort is:

$$\textsc{risk}(t, s) := \min\left\{e^{-c}, \frac{1}{yk}\right\}. \tag{2}$$

We will additionally refer to the buyer's (security) *loss* as the expected fines they suffer plus the effort they spend. That is:

$$\ell(t, s) := y \cdot \textsc{risk}(t, s) + h^*(t, s) := \begin{cases} \frac{\ln(yk)-c+1}{k}, & yk \geq e^c \\ ye^{-c}, & yk < e^c \end{cases} \tag{3}$$

It then follows that the buyer's utility (value minus price minus expected fines) is:

$$u(t, s) := v - p - \ell(t, s) = \begin{cases} v - p - 1/k - \frac{\ln(yk)-c}{k}, & yk \geq e^c \\ v - p - ye^{-c}, & yk < e^c \end{cases} \tag{4}$$

Population of Buyers. We model the population of buyers as a distribution D over types t. Additionally, we make the now-typical assumption in the multi-dimensional mechanism design literature (e.g. [8, 14, and follow-up work]) that the parameters v and k are drawn independently, so that $D := D_v \times D_k$.[1] The seller's profits are then:

$$\textsc{Prof}_D(s) := (p - c) \cdot \Pr_{t \leftarrow D}[u(t, s) \geq 0] \tag{5}$$

Externalities. Finally, we define the externalities caused and the regulator's objective function. Each device sold has some probability of being compromised, and the regulator wishes to minimize the total fraction of compromised devices.[2] That is, we measure the externalities caused as:[3]

$$\textsc{Ext}_D(s) := \frac{\mathbb{E}_{t \leftarrow D}[\textsc{risk}(t, s) \cdot \mathbb{I}(u(t, s) \geq 0)]}{\Pr_{t \leftarrow D}[u(t, s) \geq 0]} \tag{6}$$

Optimization. The regulator's objective is to propose an $s = (y, c, p)$ that minimizes $\textsc{Ext}_D(s)$. Observe that, if left unconstrained, the regulator can simply propose $c \to \infty$, resulting in 0 externalities. Such a policy is completely unrealistic, as it would cause costs to approach ∞ and destroy the industry. Similarly, taking $y \to \infty$ would cause consumers to have negative utility even to get the item for free (again destroying the industry). We therefore impose a minimum profit constraint for a policy to be considered feasible. Indeed, this forces the regulator to trade off profits for externalities

[1] This assumption is even more justified in our setting than usual, as it is hard to imagine correlation between the *value* a consumer derives from using a smart refridgerator and their *ability* to secure IoT devices.

[2] It would be equally natural for the regulator to aim to minimize the total mass of compromised devices. Most of our results do not rely on optimizing one objective versus the other, but we stick with one in order to unify the presentation.

[3] Below, $\mathbb{I}(\cdot)$ denotes the indicator function, which takes value $\mathbb{I}(X) = 1$ whenever event X occurs, and 0 otherwise.

as effectively as possible. Therefore, our regulator is given some profit constraint R, and aims to find:

$$\arg\min_{s,\textsc{Prof}_D(s)\geq R}\{\textsc{Ext}_D(s)\}$$

We will only consider cases where there is *some* feasible s (that is, we will only consider R such that there exists a p with $R \leq \textsc{Prof}_D(0,0,p)$. If no such p exists, then the profit constraint exceeds the optimal achievable profit without regulation, and the problem is unsolvable).

Recap of model. Table 1 recaps the parameters of our model for future reference. Note also that many parameters (e.g. $\ell(\cdot,\cdot)$ are formally defined as a function of $t = (v,k)$ and $s = (y,c,p)$, but only depend on (e.g.) k, y, c. As such, it will often be clearer to overload notation and write $\ell(k,y,c)$, rather than defining a new $t = (v,k)$ with a meaningless parameter. Sometimes, though, it will be clearer to use the defined notation for a type t that was just defined. In the interest of clarity, we will overload notation for these variables, but it will be clear from context what they refer to.

Table 1: Model Variables

Variable	Text Definition	Formal Definition
$t = (v,k)$	(value, effectiveness)	N/A
$D := D_v \times D_k$	buyer population	N/A
$s = (y,c,p)$	(fine, security, price)	N/A
$h^*(t,s)$	buyer optimal effort	$\max\{0, \frac{\ln(yk)-c}{k}\}$,(1)
$\textsc{risk}(t,s)$	compromise prob.	$\min\{e^{-c}, \frac{1}{yk}\}$, (2)
$\ell(t,s)$	buyer security loss	Equation (3)
$u(t,s)$	buyer utility	$v - p - \ell(t,s)$, (4)
$\textsc{Prof}_D(s)$	seller profits	Equation (5)
$\textsc{Ext}_D(s)$	frac. compromised	Equation (6)

Final Thoughts on Model. We propose a stylized model to capture the following salient aspects of this market: (a) neither buyer nor seller suffer externalities when the item is compromised, (b) the regulator can regulate both the product (via c) and payments (via y, p), (c) there is a population of buyers, each with different value v and effectiveness k at translating effort into security, and (d) the regulator must effectively trade off externalities with profits by minimizing negative externalities, subject to a minimum profit constraint R. The goal of this model is not to capture every potentially relevant parameter, but to isolate the salient features above.

3.1 An Intuitive Example

In this section, we provide one example to help give intuition for the interaction between the fines y, default security c, and seller's profits $\textsc{Prof}_D(s)$. In particular, Figure 1 plots the maximum achievable $\textsc{Prof}_D(s)$ over all s with a fixed c (the x-axis) and y (the color of the plot). In all three examples, D_v is the uniform distribution on $[0,20]$, and k is drawn from either the uniform distribution on $[0,1]$, $[0,3]$, or $[2,3]$, respectively. Note that $k \geq 1$ is the threshold when a buyer is more efficient than the seller in mitigating externalities, so these examples cover two *homogeneous* populations, where all consumers are more (respectively, less) efficient than the producer, and one *heterogeneous* population, where some consumers are more efficient, and others are not.

For each possible (partial) regulation (y,c), the profit-maximizing choice of p is essentially a classic single-item problem (e.g. [25]), as the buyer's "modified value" v' is simply $v - \ell(t,s) - c$, and the seller's profit for setting price p is just $p \cdot \Pr_{t\leftarrow D}[v' \geq p]$. Therefore, for each partial regulation (y,c), we can construct the modified distribution and simply maximize $p \cdot \Pr_{t\leftarrow D}[v' \geq p]$ as above.

Observe in Figure 1, when $y = 0$, the seller gets greater profits with lower c. This should be intuitive, as neither the buyer nor seller suffer when the device is compromised. When $y > 0$, and $D_k = U([0,1])$, the seller's profits can increase with c. This should also be intuitive: now that the buyer suffers when the device is compromised, they prefer to buy a secure device.

On the other hand, when the market contains only efficient buyers ($k > 1$ always), the buyer prefers to provide her own security; any increased cost will always decrease the buyer's utility. Indeed, observe that $\frac{\partial \ell(t,s)}{\partial c}$ is either 0 (if $yk < e^c$) or $1 - 1/k$ (otherwise). If $k > 1$, then this is always positive, so higher c results in (weakly) higher loss for the consumer, and lower utility.

3.2 Preliminary Observations

We conclude with two observations which allow an easy comparison between the profits of certain policies. Intuitively, Observation 1 claims that any policy which makes *every single* consumer in the population have lower loss generates greater profits for the seller. We will make use of Observation 1 repeatedly throughout the technical sections to modify existing policies into ones which improve profits (ideally while also improving externalities, although that is not covered by Observation 1).

Observation 1. *Let $s = (y,c,p)$, $s' = (y',c',p')$ be such that $p' - c' = p - c \geq 0$ and for all $k \in$ support(D_k), $\ell(k,s) + c \leq \ell(k,s') + c'$. Then for all D_v, $\textsc{Prof}_{D_v\times D_k}(s) \geq \textsc{Prof}_{D_v\times D_k}(s')$.*

Proof. Observe that for both s and s', the seller's profit per sale is identical (as $p' - c' = p - c$). So we just wish to show that the probability of sale for s' is larger than that for s. Indeed, observe that for all t:

$$\begin{aligned} u(t,s) &= v - \ell(k,s) - p \\ &= v - (\ell(k,s) + c) + c - p \\ &\geq v - (\ell(k,s') + c') + c' - p' \\ &= v - \ell(k,s') - p' = u(t,s'). \end{aligned}$$

Therefore, any consumer (v,k) who chooses to purchase the item under policy s' will also choose to purchase under policy s, and therefore the probability of sale is at least as large for s as s'. □

Observation 2 below claims that the profit of any policy s is larger in populations D where every consumer is more effective than in D'.

Observation 2. *Let D_k* stochastically dominate *D'_k.*[4] *Then for all policies $s = (y,c,p)$ with $p \geq c$, and all D_v, $\textsc{Prof}_{D_v\times D_k}(s) \geq \textsc{Prof}_{D_v\times D'_k}(s)$.*

[4]That is, it is possible to couple draws (k,k') from (D_k, D'_k) so that $k \geq k'$ with probability 1. Equivalently: for all x, $\Pr[k \geq x, k \leftarrow D_k] \geq \Pr[k' \geq x, k' \leftarrow D'_k]$.

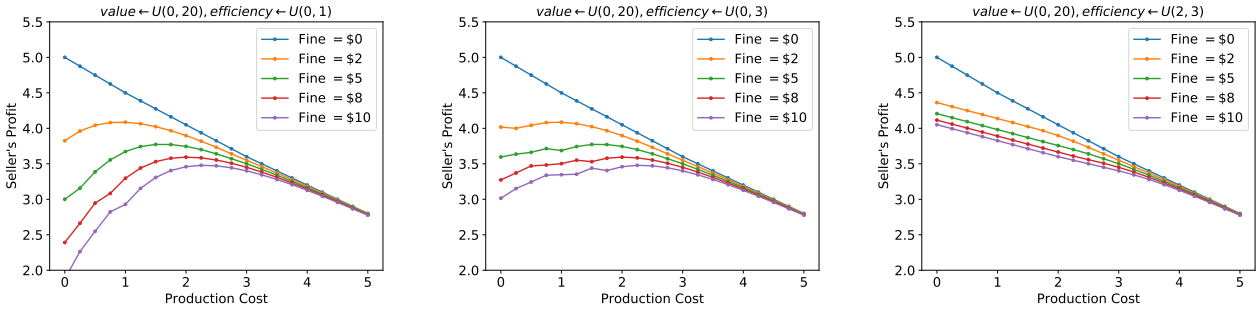


Figure 1: Seller's optimal profits under different distributions for efficiency, k. We plot the seller's profits on the vertical axis and the default security c on the horizontal axis. Each curve corresponds to different fines, y. Importantly, observe that when the fine is zero, the seller achieves greatest profits with lower default security. However, when the fine is non-zero, the seller may actually *increase* their profits with default security, but the benefits (to the seller) of default security decrease as the buyer population becomes more efficient.

Proof. As D_k stochastically dominates D'_k, it is possible to couple draws (t, t') from $(D_v \times D_k, D_v \times D'_k)$ such that $v = v'$ and $k \geq k'$. Observe simply that $u(t, s) \geq u(t', s)$ always. Therefore, $\Pr[u(t, s) \geq 0] \geq \Pr[u(t', s) \geq 0]$, and $\textsc{Prof}_{D_v \times D_k}(s) \geq \textsc{Prof}_{D \times D'_k}(s)$. □

Observe, however, that Observation 2, perhaps counterintuitively, does *not* hold if we replace profits with externalities. That is, for a fixed policy s, we might *increase* all consumers' effectiveness yet also *increase* the externalities caused. Intuitively, this might happen (for instance) in a fine policy which successfully only sells the item to extremely effective consumers who effectively secure their purchase. Ineffective consumers choose not to purchase the product to avoid fines. However, if these ineffective consumers are instead somewhat effective, they may now choose to purchase the item, thereby increasing externalities. Below is a concrete instantiation:

Example 3.2. Consider the population where D_v is a point-mass at e, and D_k takes on effectiveness 0 with probability $1/2$ and $x > 1$ with probability $1/2$. Consider the policy $s = (e, 0, e - 2.5)$. Then the $(e, 0)$ consumer chooses not to purchase: $\ell(e, 0, s) = e$, so their utility would be $e - e - (e - 2.5) < 0$. The (e, x) consumer chooses to purchase, as their loss is $\frac{2+\ln(x)}{x} < 2$ (as $x > 1$). So $\textsc{Ext}_{D_v \times D_k}(s) = \frac{1}{ex}$.

Consider now improving the effectiveness of the $k = 0$ consumers to $k = 1$ (so D'_k now takes on 1 with probability $1/2$ and x with probability $1/2$). The $(e, 1)$ consumer now chooses to purchase, as their loss is 2 (so their utility is $e - 2 - (e - 2.5) = 1/2$). So now $\textsc{Ext}_{D_v \times D'_k}(s) = (\frac{1}{e} + \frac{1}{ex})/2$. As $x > 1$, the externalities have gone up. If $x \geq 1$, the externalities may have gone up quite significantly.

In Example 3.2, of course "the right" thing to do is to also change the policy. Indeed, it is still the case that, for a *fixed consumer who purchases the item*, increasing effectiveness can only decrease externalities. But without fixing whether the consumer has purchased the item or not, the claim is false. Observation 3 captures what we can claim about risk, loss, etc. on a per-consumer basis. Proofs for the claims in Observation 3 all follow immediately from the definitions in Section 3.

Observation 3. *Let $k > k'$, then for all s:*

- $\textsc{risk}(k, s) \leq \textsc{risk}(k', s)$.
- $\ell(k, s) \leq \ell(k', s)$.
- $h^*(k, s) \geq h^*(k', s)$.
- $u(k, s) \geq u(k', s)$.

4 ROADMAP OF TECHNICAL SECTIONS

Now that we have the appropriate technical language, we provide a brief roadmap of the results to come.

- In Section 5, we provide a technical warmup to get the reader familiar with how to reason about our problem. The main result of this section is Theorem 5.1, which claims that the optimal policy when D_k is a point-mass is simple. The proof of this theorem helps illustrate one key aspect of our later arguments, and will also be used as a building block for later proofs.
- In Section 6, we prove our first main result (Theorem 6.1): as a function of R and D_v, there exists a cutoff T. If D_k is supported on $[0, T]$, then a cost policy is optimal. If D_k is supported on $[T, \infty)$, then a fine policy outperforms all *profits-maximizing* policies (we define this term in the relevant section — intuitively a policy is profits-maximizing if the price is the seller's best response to (y, c)). Section 6 also contains a surprising example witnessing that the additional profits-maximizing qualification is necessary.
- In Section 7, we consider general distributions. Unsurprisingly, simple policies are no longer optimal. Perhaps surprisingly, if one insists on exceeding the profits benchmark *exactly*, no simple policy can guarantee any bounded approximation to the optimal externalities (Corollary 7.5). However, we also show (Theorem 7.6) that it is possible to get a bicriterion approximation: if one is willing to approximately satisfy the profits constraint, it is possible to approximately minimize externalities with a simple policy. That is, for any s, D, there is a simple policy s' with $\textsc{Prof}_D(s') = \Omega(1) \cdot \textsc{Prof}_D(s)$ and $\textsc{Ext}_D(s') = O(1) \cdot \textsc{Ext}_D(s)$.

- We include complete proofs for most of our results on point-mass and homogeneous distributions, as these convey many of the key ideas. By Theorem 7.6, the proofs get quite technical so we provide a sketch of the main ideas. This and other omitted proofs can be found in [7].

5 WARM-UP: POINT-MASS EFFECTIVENESS

As a warm-up, we first study the case where D_k is a point mass (that is, all buyers in the population have the same effectiveness k). In this case, we show that a simple policy is optimal. The proof is fairly intuitive, with one catch. The intuitive part is that every consumer will put in the same effort, conditioned on buying the item. It therefore seems intuitive that if $k < 1$, it is better for all parties involved if any effort spent by the consumer is transferred to the producer instead (and this is true). It also seems intuitive that if $k > 1$, it is again better for all parties involved if any effort spent by the producer is "transferred" to the consumer instead (e.g. by raising fines so that the consumer chooses to spend the desired level of effort). This is not quite true: the catch is that the fine required to induce the desired buyer behavior may be too high to satisfy the profit constraint. But, the above argument does work for sufficiently large k. Importantly, there is some cutoff T such that for all $k \le T$, the optimal policy is a cost policy ($y = 0$), while for all $k \ge T$, the optimal policy is a fine policy ($c = 0$). Below, when we write $D_v \times \{k\}$, we mean the distribution which draws v from D_v and outputs (v, k).

Theorem 5.1. *For all D_v, R, and k, the externality-minimizing policy for $D_v \times \{k\}$ is a simple policy. Moreover, for all R, D_v, there is a cutoff T such that if $k \le T$, then the optimal policy is a cost policy. If $k \ge T$, then the optimal policy is a fine policy.*

Proof. Consider any policy $s = (y, c, p)$. Because all consumers have the same effectiveness k, s induces the same loss for all consumers. We first claim the following: □

Lemma 5.2. *Let $k \le 1$. Then for all D_v and any policy $s = (y, c, p)$, there is an alternative policy $s' = (0, c', p)$ with $\textsc{Prof}_{D_v \times \{k\}}(s') \ge \textsc{Prof}_{D_v \times \{k\}}(s)$ and $\textsc{Ext}_{D_v \times \{k\}}(s') \le \textsc{Ext}_{D_v \times \{k\}}(s)$.*

Proof. In policy s, all consumers have the same loss $\ell(k, s)$. This therefore is a good opportunity to try and make use of Observation 1. First, consider the possibility that $yk < e^c$. In this case, $h^*(k, s) = 0$, $\ell(k, s) = ye^{-c}$, and $\textsc{risk}(k, s) = e^{-c}$. This implies that $\textsc{Ext}_{D_v \times \{k\}}(s) = e^{-c}$. Consider instead the policy $s' = (0, c, p)$. Then $\ell(k, s') = 0$, but $\textsc{risk}(k, s) = e^{-c}$ and $\textsc{Ext}_{D_v \times \{k\}}(s) = e^{-c}$ like before. So the externalities are the same. An application of Observation 1 concludes that the profits have improved (indeed, (c, p) are the same in both policies, and the loss decreases as we switch from policy s to s').

Consider now the possibility that $yk \ge e^c$. In this case, $h^*(k, s) = \frac{\ln(yk) - c}{k}$, $\ell(k, s) = \frac{\ln(yk) - c + 1}{k}$, and $\textsc{risk}(k, s) = \frac{1}{yk}$. Consider instead the policy $s' = (0, \ln(yk), p - c + \ln(yk))$. In this new policy, $\ell(k, s') = 0$ and $\textsc{risk}(k, s') = \frac{1}{yk}$. So indeed, the new policy has the same externalities. We just need to ensure that we can apply Observation 1. To this end, observe that:

$$\begin{aligned}\ell(k, s) + c - (\ell(k, s') + c') &= \frac{\ln(yk) - c + 1}{k} + c - \ln(yk) \\ &= (1/k - 1) \cdot (\ln(yk) - c) + 1/k \\ &\ge 0.\end{aligned}$$

The last line follows because $k \le 1$ and $\ln(yk) \ge c$ (because $yk \ge e^c$). So the hypotheses of Observation 1 hold, and we can apply Observation 1 to conclude that the profits improve from s to s' as well. □

Lemma 5.2 covers the cases when $k \le 1$: there is always an optimal cost policy. We now move to the case when $k > 1$. There are two cases to consider: one where the optimal policy will be a cost policy, and one where the optimal policy will be a fine policy. The distinguishing feature between these cases will be for a given c, how big of a fine is necessary to incentivize the consumer to put in effort c/k, and what the consumer's loss looks like for this choice of y. Below, c^* is defined to be the maximum c such that there exists a p such that $\textsc{Prof}_{D_v \times \{0\}}(0, c, p) \ge R$. Observe that c^* is also equal to the maximum ℓ such that there exists a p such that $\textsc{Prof}_{(D_v - \ell) \times \{0\}}(0, 0, p) \ge R$ (here, $D_v - \ell$ denotes the distribution which samples v from D_v and then subtracts ℓ, taking a maximum with 0 if desired). That is, c^* is the maximum loss that can be uniformly applied to all consumers (drawn from D_v) while still resulting in a distribution for which profit $\ge R$ is achievable.

Lemma 5.3. *Let c^* denote the maximum c such that there exists a p such that $\textsc{Prof}_{D_v \times \{0\}}(0, c, p) \ge R$. Then a cost policy is optimal for $D_v \times \{k\}$ if $k \in [1, 1 + 1/c^*]$.*

Proof. First, observe that the lemma hypothesis implies that *any* feasible policy must have $\ell(k, s) + c \le c^*$ (if not, then an application of Observation 1 lets us contradict the lemma's hypothesis with a feasible $c' = \ell(k, s) + c > c^*$).

Consider now $k \in [1, 1 + 1/c^*]$, and start from some policy $s = (y, c, p)$. If this policy has $h^*(k, s) = 0$, then certainly we can just update $s' = (0, c, p)$ and get better profits with the same externalities (by Observation 1). If instead $h^*(k, s) > 0$, then $\ell(k, s) = \frac{\ln(yk) - c + 1}{k}$, and $\textsc{risk}(k, s) = \frac{1}{yk}$. Consider instead $s^* = (0, c^*, p^*)$, for whichever p^* witnesses $\textsc{Prof}_D(s^*) \ge R$ (we know that such a p^* exists by the lemma's hypothesis). So now we just need to compare externalities. Assume for contradiction that $\textsc{risk}(k, s^*) > \textsc{risk}(k, s)$. Then we get:

$$\begin{aligned}\textsc{risk}(k, s^*) > \textsc{risk}(k, s) &\Rightarrow e^{-c^*} > \frac{1}{yk} \\ &\Rightarrow c^* < \ln(yk) \\ &\Rightarrow \frac{\ln(yk) - c + 1}{k} > \frac{c^* - c + 1}{k} \\ &\Rightarrow \ell(k, s) + c > \frac{c^* - c + 1}{k} + c \\ &\Rightarrow \ell(k, s) + c > \frac{c^* + 1}{k} \\ &\Rightarrow \ell(k, s) + c > c^* \Rightarrow\Leftarrow .\end{aligned}$$

The last implication uses the fact that $k \le 1 + 1/c^*$. The line before this uses that $k \ge 1$. The contradiction arises because this would imply a scheme (s) with profit $\ge R$ with loss $> c^*$, contradicting

the definition of c^* by the reasoning in the first paragraph of this proof. □

Lemma 5.4. *Let c^* denote the maximum c such that there exists a p such that $\text{Prof}_{D_v \times \{0\}}(0, c, p) \geq R$. Then a fine policy is optimal for $D_v \times \{k\}$ if $k \geq 1 + 1/c^*$.*

Proof. Again start from some policy $s = (y, c, p)$, inducing some loss $\ell(k, s)$. First, maybe $h^*(k, s) > 0$. In this case, the risk is $\frac{1}{yk}$ and the loss plus cost is $\frac{\ln(yk)-c+1}{k} + c$. In particular, observe that the partial derivative of the loss plus cost with respect to c is $1-1/k > 0$. So the policy $s' = (y, 0, p - c)$ has $\text{risk}(k, s') = \text{risk}(k, s)$ but also $\ell(k, s') + c' < \ell(k, s) + c$. So Observation 1 claims that this policy gets at least as much profits (and the risk is the same).

If instead, $h^*(k, s) = 0$, then the risk is e^{-c} and the loss is $y \cdot e^{-c}$. In this case, consider instead y^* such that $\frac{\ln(y^*k)+1}{k} = c^*$ and using $s^* = (y^*, 0, p^*)$, for the p^* satisfying $\text{Prof}_D(s^*) \geq R$ (again, such a p^* must exist by definition of c^*, and the fact that $\ell(k, s^*) = c^*$, plus Observation 1). We just need to analyze the risk. Similar to the previous proof, assume for contradiction that $\text{risk}(k, s^*) > \text{risk}(k, s)$. Then:

$$\begin{aligned}
\text{risk}(k, s^*) > \text{risk}(k, s) &\Rightarrow e^{-c} < \frac{1}{y^*k} \\
&\rightarrow c > \ln(y^*k) \\
&\Rightarrow \frac{\ln(y^*k) + 1}{k} < \frac{c+1}{k} \\
&\Rightarrow c^* < \frac{c+1}{k} \\
&\Rightarrow c^* > c^* \cdot \frac{c+1}{1+c^*} \Rightarrow\Leftarrow .
\end{aligned}$$

The last inequality uses the fact that $k \geq 1 + 1/c^*$, and derives a contradiction as $c \leq c^*$ (if $c > c^*$, then certainly $\ell(k, s) + c > c^*$, contradicting the definition of c^*). □

All three cases together prove Theorem 5.1. The T prescribed in the theorem statement is exactly $1 + 1/c^*$, where c^* is the maximum c such that there exists a p for which $\text{Prof}_{D_v \times \{0\}}(0, c, p) \geq R$.

We conclude with one last proposition regarding the behavior of the threshold with respect to the profits constraints R. Proposition 5.5 below states that as R increases, the threshold beyond which a fine policy is optimal increases as well.

Proposition 5.5. *Let $T(D_v, R)$ denote the threshold such that both a fine policy and cost policy are optimal for $D_v \times \{T(D_v, R)\}$ subject to profits constraints R. Then $T(D_v, R)$ is monotone increasing in R.*

Proof. To see this, let $c^*(D_v, R)$ denote the maximum c such that there exists a p such that $\text{Prof}_{D_v \times \{0\}}(0, c, p) \geq R$. Then $c^*(D_v, R)$ is *decreasing* in R (as the profits constraint goes up, we can't afford as much security). So $1 + 1/c^*(D_v, R)$ is *increasing* in R. This means that the threshold $T(D_v, R)$ beyond which a fine policy is optimal for $D_v \times \{T\}$ is increasing as a function of the profits constraint R (because $T = 1 + 1/c^*(D_v, R)$). □

This concludes our treatment of the case where k is a point-mass. Theorem 5.1 should both be viewed as a warm-up to introduce some of our core techniques, and also as a building block towards our stronger theorems (in the following sections). The main technique we introduced is the ability to reduce risk and loss simultaneously to improve both profits and externalities. The idea was that if the buyer is less effective than the seller, everyone prefers that the seller put in effort ($y = 0, c > 0$). If the buyer is more effective than the seller, everyone prefers that the buyer put in effort. However, the regulator can not directly mandate that the buyer put in effort, and unfortunately the fines required to extract the desired buyer behavior may too negatively affect the profit. This is why the transition from cost to fine policies is $1 + 1/c^*$ instead of 1.

6 HOMOGENEOUS DISTRIBUTIONS

In this section, we show that for populations that are sufficiently homogeneous in effectiveness, the optimal policy remains simple. The second half of Theorem 6.1 requires a technical assumption. Specifically, we say that a policy (y, c, p) is *profits-maximizing* if, conditioned on y, c, p is set to maximize the seller's profits (that is, $\text{Prof}_D(y, c, p) \geq \text{Prof}_D(y, c, p')$ for all p').

Theorem 6.1. *For all D_v, R, there exists a cutoff T such that*

- *For all D_k supported on $[0, T]$, the externality-minimizing policy for $D_v \times D_k$ subject to profits R is a cost policy.*
- *For all D_k supported on $[T, \infty)$, the externality-minimizing policy for $D_v \times D_k$ subject to profits R is either a fine policy, or it is* not *profits-maximizing.*

The proof of Theorem 6.1 will follow from Lemmas 6.3 and 6.7, which handle the two claims in the theorem separately. Finally, we show in Section 6.3 that the profits-maximizing qualification in part two of Theorem 6.1 is necessary:

Proposition 6.2. *There exist distributions D_v, D_k, and profits constraint R such that:*

- *T is such that for all $k \geq T$, the externality-minimizing policy for $D_v \times \{k\}$ subject to profits constraints R is a fine policy.*
- *D_k is supported on $[T, \infty)$.*
- No *fine policy is externality-minimizing policy for $D_v \times D_k$ subject to profits constraints R.*
- *The externality-minimizing policy for $D_v \times D_k$ subject to profits constraints R is not simple, and not profits-maximizing (the latter is implied by the second bullet of Theorem 6.1).*

Proposition 6.2 is perhaps surprising: a fine policy is externality-minimizing for $D_v \times \{T\}$, and D_k stochastically dominates T, so the same fine policy has *even lower* externalities, and potentially greater profit for $D_v \times D_k$. Indeed, the optimal fine policy for $D_v \times D_k$ achieves lower externalities than that of $D_v \times \{T\}$. The catch is that an *even better* non-simple policy becomes viable, and achieves *still lower* externalities. Theorem 6.1 claims, however, that the optimal non-simple policy must not be profits-maximizing.

6.1 Extension Lemma for small k

The small k case follows roughly from the following intuition. For cost policies, neither the buyer's loss nor her risk depend on k. So whichever cost policy is optimal for $D_v \times \{T\}$ achieves the same profits and externalities as $D_v \times D_k$. Intuitively, going from $\{T\}$ to D_k supported on $[0, T]$ cannot possibly *increase* the profits of any scheme (formally: Observation 2), so the initial cost policy should remain optimal.

Lemma 6.3 (Extension of Cost Policy). *Let s be a cost policy that is optimal for $D_v \times \{T\}$ subject to profits R. Then for all D_k supported on $[0, T]$, s is optimal for $D_v \times D_k$ subject to profits R.*

Proof. First, we observe that $\text{Prof}_{D_v \times \{T\}}(s) = \text{Prof}_{D_v \times D_k}(s)$. This is simply because the loss of consumers is independent of k (as $y = 0$). Similarly, $\text{Ext}_{D_v \times \{T\}}(s) = \text{Ext}_{D_v \times D_k}(s)$. This is again because the risk of consumers is independent of k.

Now, assume for contradiction that there is some policy s' with profits $\text{Prof}_{D_v \times D_k}(s') \geq R$ and also $\text{Ext}_{D_v \times D_k}(s') < \text{Ext}_{D_v \times \{T\}}(s)$. Then we have the following inequality from Observation 2:

$$R \leq \text{Prof}_{D_v \times D_k}(s') \leq \text{Prof}_{D_v \times \{T\}}(s').$$

Therefore, as s is optimal for $D_v \times \{T\}$ subject to profits R, we must have:

$$\text{Ext}_{D_v \times \{T\}}(s') \geq \text{Ext}_{D_v \times \{T\}}(s).$$

This now lets us conclude the following chain of inequalities, where the first line is a corollary of Observation 3: the consumer in a population with D_k supported on $[0, T]$ whose device is least likely to be compromised is a consumer with $k = T$. The third line follows from the reasoning above (that s' achieves profits at least R on $D_v \times \{T\}$, and is therefore feasible). The final line follows because the externalities of a cost policy are independent of k.

$$\begin{aligned}
\text{Ext}_{D_v \times D_k}(s') &\geq \text{risk}(T, s') \\
&= \text{Ext}_{D_v \times \{T\}}(s') \\
&\geq \text{Ext}_{D_v \times \{T\}}(s) \\
&= \text{Ext}_{D_v \times D_k}(s).
\end{aligned}$$

□

Lemma 6.3 proves the first bullet of Theorem 6.1.

6.2 Extension Lemma for large k

In this section, we sketch the proof for the large k case of Theorem 6.1. Refer to Section 6 of [7] for omitted proofs. The proof will be a little more involved this time, since we can no longer claim that the externalities of a fine policy are independent of k (whereas this does hold for cost policies). The intuition for this case is the same though: if a fine policy is optimal for $D_v \times \{k\}$ for all $k \geq T$, and D_k is supported on $[T, \infty)$, fine policies should remain optimal for $D_v \times D_k$. Most of the proof does not make use of the technical assumption that the s we are competing with is a profits-maximizing policy: this assumption only arises at the very end.

The first step in our proof is the following concept, which captures the change in loss for a consumer (v, k) for regulation s versus s':

Definition 6.4 (Policy Comparison Function). For two policies s and s', we define the *policy comparison function* $g_{s,s'}(\cdot)$ so that $g_{s,s'}(k) = \ell(k, s) - \ell(k, s')$.

The policy comparison function takes as input an effectiveness k, and outputs the change in loss for a consumer under one policy versus another. Our first lemma argues that for certain pairs (s, s'), the policy comparison function is monotone in k. That is, consumers with more effectives have greater preference for one policy over another.

Lemma 6.5. *Let $s = (y, c, p)$ and $s' = (y', c', p')$ be such that $ye^{-c} \leq y'e^{-c'}$. Then $g_{s,s'}(\cdot)$ is monotone non-decreasing. Observe that the hypothesis holds if $y \leq y'$ and $c \geq c'$.*

We use Lemma 6.5 to claim the following corollary, which essentially states that if a policy change universally lowers loss *and* risk, then it is possible to adjust the price so that the profits go up and externalities go down.

Corollary 6.6. *Let $(y, c), (y', c')$ be such that (a) $ye^{-c} \leq y'e^{-c'}$ and (b) for all k in the support of D_k, $\ell(k, y, c) + c \geq \ell(k, y', c') + c'$ and $\text{risk}(k, y, c) \geq \text{risk}(k, y', c')$. Then for all p and all D_v, there exists a p' such that:*

$$\textit{Prof}_{D_v \times D_k}(y', c', p') \geq \textit{Prof}_{D_v \times D_k}(y, c, p),$$

$$\textit{Ext}_{D_v \times D_k}(y', c', p') \leq \textit{Ext}_{D_v \times D_k}(y, c, p).$$

Now we are ready to formally state the extension lemma for large k.

Lemma 6.7 (Extension of Fine Policy). *Let D_k be supported on $[T, \infty)$, where T is such that a fine policy is optimal for $D_v \times \{T\}$ subject to profits R. Then there is a fine policy s' with $\textit{Prof}_D(s') \geq R$ such that for all profits-maximizing s with $\textit{Prof}_D(s) \geq R$, $\textit{Ext}_D(s') \leq \textit{Ext}_D(s)$.*

Proof Sketch. Consider any proposed optimal policy $s = (y, c, p)$. We first consider the case where $\ell(T, s) + c \geq \ell(T, s^*)$ where s^* is the optimal fine policy on $D_v \times \{T\}$. If that is the case, then consider a fine policy $s' = (y', 0, p - c)$ where $\ell(T, s') = \ell(T, s) + c$. Observe we must have $y' \geq y$ since we can only obtain equality by increasing fines, then by Lemma 6.5, we have $\ell(k, s') \leq \ell(k, s) + c$ for all k in the support of D_k and the profit under s' can only be higher.

Now, we need to show that the risk is only lower for all $k \geq T$. First, as $\ell(T, s') \geq \ell(T, s^*)$ we conclude that $y' \geq y^*$. Next, we can argue (see [7] for the full proof) that we must have $\frac{1}{y^*T} \leq e^{-c}$. This allows us to conclude that $\frac{1}{y'T} \leq e^{-c}$, and we already have that $y' \geq y$ so $\frac{1}{y'k} \leq \frac{1}{yk}$. This gives us that $\text{risk}(k, s') \leq \text{risk}(k, s)$ for all $k \geq T$. Then Corollary 6.6 let's us claim that there exists a price p' for which both the profits and the externalities are better for $(y', 0, p')$ than s.

For the case where $\ell(T, s) + c < \ell(T, s^*)$, observe that $\ell(D_k, s) + c$ is strictly stochastic dominated by $\ell(D_k, s^*)$; therefore, there is a price $\hat{p}$ where the profit strictly higher than R. Lemma 6.9 bellow, allows us conclude that if s is optimal, then $\text{Prof}_{D_v \times D_k}(s) = R$; otherwise, we can compromise $\varepsilon > 0$ fraction of the profit to strictly improve externalities. This implies that if the policy s we start with is optimal, then it is not profits-maximizing. □

Definition 6.8 (Invariant Transformation). Given a policy $s = (y, c, p)$, define

$$\text{Inv}(s, \alpha) := \left(ye^{(p-c)(1-\alpha)}, \alpha c + (1 - \alpha)p, p\right)$$

where $\alpha \in [0, \frac{p}{p-c}]$.

Lemma 6.9 (Invariant Property). *Let $s' = \textsc{Inv}(s, \alpha)$, then for all $k \in \mathbb{R}^+$*

- $h^*(k, s') = h^*(k, s)$.
- $\ell(k, s') = \ell(k, s)$.
- $u(t, s) = u(t, s')$.

In addition,

$$\textsc{Prof}_D(s') = \alpha \textsc{Prof}_D(s)$$
$$\textsc{Ext}_D(s') = e^{-(1-\alpha)(p-c)} \textsc{Ext}_D(s, p)$$

Proof Sketch. By applying the definition of buyer's efficiency, we can see that for all k, $h^*(k, s') = h^*(k, s)$ which implies $\ell(k, s') = \ell(k, s)$, and $\textsc{risk}(k, s') = e^{-(1-\alpha)(p-c)} \textsc{risk}(k, s)$. □

This concludes the proof of bullet two of Theorem 6.1.

6.3 Example: The Profits-Maximizing Qualification is Necessary

In this section we provide the example promised in Proposition 6.2. Refer to Section 6 of [7] for omitted proofs. Consider the following distribution, and profits constraint $R := 0.5$:

$$D_v = \begin{cases} v_1 = 1 & \text{w. p. } \frac{1}{2} \\ v_2 = 16/15 & \text{w. p. } \frac{1}{2} \end{cases}$$

$$D_k = \begin{cases} k_1 = 3 & \text{w. p. } \frac{1}{2} \\ k_2 = x \to \infty & \text{w. p. } \frac{1}{2} \end{cases}$$

Above, x will be finite, but approaching ∞, and ε will be finite but approaching 0). The proposition will follow from the following sequence of claims. First, we will establish bullet one for $T := 3$.

Claim 4. *A fine policy is optimal for $D_v \times \{3\}$.*

Bullet two now immediately follows, as D_k is indeed supported on $[3, \infty)$. We now just need to find the optimal fine policy for $D_v \times D_k$, and establish a better policy that is not simple. We now search for the optimal fine policy. Such a policy might sell only to $(16/15, x)$, but then the profits is at most $4/5$, which is too little. Such a policy might sell only to $(16/15, x)$ and $(1, x)$. But since x is finite, such a policy certainly charges price < 1 (unless $y = 0$, in which case the policy sells to all four types), and sells with probability $\leq 1/2$, so the profits are also too small. Such a policy might sell to all four types, which we analyze below. Or it might sell to all types except $(1, 3)$, which we analyze after.

Claim 5. *The optimal fine policy s which sells to all four types has $\textsc{Ext}_{D_v \times D_k}(s) \geq \frac{1}{2\sqrt{e}}$.*

Claim 6. *The optimal fine policy s which sells to all types except $(1, 3)$ has $\textsc{Ext}_{D_v \times D_k}(s) \geq e^{-1/5}/3$.*

Corollary 6.10. *The optimal fine policy s has $\textsc{Ext}_{D_v \times D_k}(s) \geq e^{-1/5}/3$.*

Here's now some intuition for how we're going to design a better non-simple policy: given that we wish to sell to all types except $(1, 3)$, we can set y very close to 0 and have $\textsc{risk}(x, s) \approx 0$, because x is so large. The remaining question is then whether we wish to use y or c to make the risk of $(16/15, 3)$ as small as possible. Note that we must keep their loss under $2/5 < 1/2$ (as above). But for $k = 3$, a loss of $1/2$ is *exactly* the cutoff when it becomes more efficient to use a fine policy instead of a cost policy. So if we use c instead, we can get the risk lower for the same loss.

Claim 7. *Let ε be such that $\frac{\ln(x)+1}{x} \leq \varepsilon$. Then set $c = 1/3 - \varepsilon$, and $y = (2/5 - c)e^c$. Then $\textsc{Ext}_{D_v \times D_k}(y, c, 2/3 + c) = \frac{2}{3yx} + e^{-1/3+\varepsilon}/3$ and $\textsc{Prof}_{D_v \times D_k}(y, c, 2/3 + c) = 1/2$.*

Now, we just need to compare $e^{-1/5}/3$ and $e^{-1/3+\varepsilon}/3 + \frac{2}{3yx}$. Observe that as $x \to \infty$, $\varepsilon \to 0$ and $e^{-1/3+\varepsilon}/3$ approaches $e^{-1/3}/3$. So $\frac{2}{3yx} + e^{-1/3+\varepsilon}/3 \to 0 + e^{-1/3}/3 < e^{-1/5}/3$, and the externalities are indeed lower.

As a sanity check, we'll show that $((2/5-c)e^{1/3-\varepsilon}, 1/3-\varepsilon, 2/3+c)$ is not profits-maximizing (technically, Theorem 6.1 doesn't imply this, since we didn't prove that the scheme is optimal. But as this scheme is better than all fine policies, certainly the optimal policy is not simple, and therefore not profits-maximizing by Theorem 6.1. So the fourth bullet is already proven).

Claim 8. *$((2/5 - c)e^{1/3-\varepsilon}, 1/3 - \varepsilon, 2/3 + c)$ is not profits-maximizing.*

7 GENERAL DISTRIBUTIONS: AN APPROXIMATION

In this section, we consider general distributions. Clearly, one should not expect a simple policy to be optimal in general. Given that simple policies are optimal for homogeneous populations, one might reasonably expect that simple policies are approximately optimal for general distributions by simply ignoring half of the population and targeting the half that is responsible for most of the externalities. This idea works in one direction: if the "low k" region is responsible for most of the externalities in the optimum solution, then using a cost policy for the entire distribution is a good idea: the high k consumers may have significantly higher risk than previously, but this doesn't outweigh the original risk from the low k region.

This idea fails horribly, however, if the "high k" region is responsible for most of the externalities in the optimum solution. The problem is that while we can choose a policy to exclusively target this subpopulation, any low k (think: $k = 0$) consumers who choose to purchase anyway may have enormous risk in comparison to before (i.e. it could now be 1 when it was previously e^{-c} for large c). We first show that this intuition can indeed manifest in a concrete example by presenting a lower bound in Section 7.1. This rules out a *single-criterion* approximation that satisfies the profits constraint exactly, and approximates the externalities. In Section 7, we present a bicriterion approximation which approximately satisfies the profits constraint and also approximately minimizes externalities. This approximation is our most technical result. As such, we provide mainly proof sketches to overview the key steps.

7.1 Lower Bound on Heterogeneous Distributions

The key insight for our example is to make the profits constraint so binding that the only way to match it exactly is for the entire population to purchase the item. Part of the population will have $k = 0$, and part will have $k \to \infty$. With both c and y, it will be

feasible to get the $k \to \infty$ consumers to have risk essentially 0, while the $k = 0$ consumers will have reasonably small risk. But with either $c = 0$ or $y = 0$, one of these will be lost, which causes significant risk increase.

Example 7.1. Let D_v be a point mass at $v_0 = 2e^{x/2} \cdot (x + e^{-x})$. Let D_k be a distribution with two point masses, one at $k = 0$ with probability $e^{-x/2}$, one at $e^{xe^{x/2}}$ with probability $1 - e^{-x/2}$. Let $R := v_0 - e^{-x} - x$.

Lemma 7.2. *The policy $(1, x, R+x)$ achieves profit R in Example 7.1, and has externalities $\leq e^{-x/2} \cdot e^{-x} + 1 \cdot e^{-xe^{x/2}}$.*

Proof. The utility of $(v_0, 0)$ is exactly $v_0 - e^{-x} - R - x = 0$, so they will choose to purchase. $(v_0, e^{xe^{x/2}})$ has only larger utility, so they will purchase as well. Therefore, the profit is indeed R.

The externalities are computed simply as the probability of having consumer $(v_0, 0)$ times their risk (e^{-x}) plus (upper bound on the) probability of consumer $(v_0, e^{xe^{x/2}})$ times their risk $(e^{-xe^{x/2}})$. □

Lemma 7.3. *Any cost policy that achieves profit R has externalities at least e^{-x+1}*

Proof. The maximum security we can set and still have profit R is $x + e^{-x}$. If we set this, then the risk of all consumers (which is now independent of k) is $e^{-x+e^{-x}} \geq e^{-x+1}$. □

Lemma 7.4. *Any fine policy that achieves profit R has externalities at least $e^{-x/2}$.*

Proof. To achieve profit R, the policy must sell to the entire population. The consumer with $k = 0$ will not put in any effort, and therefore their risk will be one, and the externalities will be at least $e^{-x/2}$. □

Corollary 7.5. *For all x, there exists a distribution $D_v \times D_k$ and profits constraint R such that the optimal policy is not simple, and any simple policy that satisfies profits constraints R has externalities at least a factor of x larger than the optimum.*

Corollary 7.5 is the main result of this section. Clearly the distribution witnessing Corollary 7.5 is highly contrived and unrealistic. And clearly, the way to get around this is to allow for a slight relaxation in the profits constraint so that we don't have to sell to the entire market (indeed, even allowing to relax the constraint by a $(1 - e^{-x/2})$ fraction in this case would suffice). So the subsequent section shows that by relaxing the profits constraint, an approximation guarantee is possible.

7.2 A Bicriterion approximation

Given the lower bound in Section 7.1, we show that simple policies guarantee a bicriterion approximation. As is traditional with worst-case approximation guarantees, our constants are not particularly close to 1, but are still relatively small. This is not meant to imply that the seller should be happy with (e.g.) a 1/8-fraction of the original profits, but more qualitatively to conclude that simple policies can reap many of the benefits of optimal ones (see [16] for further discussion about the role of approximation in mechanism design). As referenced previously, the proof of Theorem 7.6 is quite technical, so we sketch the key steps. The complete proof can be found on [7].

Theorem 7.6. *For all distributions D, and all policies s, there exists a simple policy s' such that*

$$\textsc{Prof}_D(s') \geq \textsc{Prof}_D(s)/8,$$

$$\textsc{Ext}_D(s') \leq 40/3 \cdot \textsc{Ext}_D(s).$$

Proof Sketch. Given an arbitrary policy $s = (y, c, p)$, consider the conditional distribution of buyers that purchase under s. If with constant probability a buyer has efficiency $k \leq 1$, then we output the cost policy $s_1 := (0, c + \ell(\sigma, s), p + \ell(\sigma, s))$ where σ is chosen such that a buyer continues to purchase with constant probability. We can show that $c + \ell(\sigma, s)$ is sufficiently large such that $\textsc{risk}(D_k, s') \leq \textsc{risk}(D_k, s)$ with constant probability.

For the case where with constant probability a buyer has efficiency $k > 1$, we define a blowup of the fines such that with constant probability a buyer continues to purchase but with the hope that inefficient buyers stop to purchase. The blowup can fail in two conditions: (1) D_k is not heavy tail, (2) D_v is heavy tail. For (1), we cannot derive a significant blowup if D_k is concentrated close to 1. For (2), we cannot drive inefficient buyers out of the market if they have high value. Either condition allow us to construct cost policies that give good externality guarantees. □

8 SUMMARY

We propose a stylized model to study regulation of single item sales with negative externalities, from which neither the buyer nor seller suffer. We first show that a simple policy is optimal in homogenous markets: That is, for all D_v, R, there exists a cutoff T such that when the effectiveness of consumers ranges in $[0, T]$, the optimal policy regulates only the product (and does not impose fines). Similarly, if all consumers have effectiveness in $[T, \infty)$, a policy which regulates only payments (via fines, and does not impose default security features) outperforms all profits-maximizing policies. Importantly, T is not necessarily the cutoff at which the consumers are more effective than the producer (which would be $T = 1$), but actually depends on the value distribution D_v and profit constraint R.

We then show in general markets that while a simple policy may not be optimal, one is always approximately optimal. In particular, we show that while no simple scheme can guarantee any finite approximation while satisfying the profit constraint exactly, a bi-criterion approximation exist, which approximately satisfies the profit constraint and also approximately minimizes externalities. Going forward, we must better understand the effectiveness of consumers to decide which regulation strategy is more appropriate to approximately minimizes externalities.

While stylized, our model captures the key salient features of this problem. We chose to study the single seller/single item setting in order to isolate these features without bringing in additional complexities (and the numerous examples throughout our paper demonstrate that even the single seller/single item setting is quite rich). Now that our results develop this understanding, a good direction for future work is to consider competing sellers or multiple items.

REFERENCES

[1] George A Akerlof. 1978. The market for "lemons": Quality uncertainty and the market mechanism. In *Uncertainty in Economics*. Elsevier, 235–251.

[2] Terrence August, Duy Dao, and Kihoon Kim. 2016. Market Segmentation and Software Security: Pricing Patching Rights. *Workshop on the Economics of Information Security (WEIS)* (2016).

[3] William J. Baumol. 1972. On Taxation and the Control of Externalities. *The American Economic Review* 62, 3 (1972), 307–322. http://www.jstor.org/stable/1803378

[4] Sayan Bhattacharya, Janardhan Kulkarni, Kamesh Munagala, and Xiaoming Xu. 2011. On allocations with negative externalities. In *International Workshop on Internet and Network Economics*. Springer, 25–36.

[5] Brian Krebs. 2017. IoT Reality: Smart Devices, Dumb Defaults. *Krebs on Security Blog, https://krebsonsecurity.com/2016/02/iot-reality-smart-devices-dumb-defaults/* (2017).

[6] Ozan Candogan, Kostas Bimpikis, and Asuman Ozdaglar. 2012. Optimal pricing in networks with externalities. *Operations Research* 60, 4 (2012), 883–905.

[7] Tithi Chattopadhyay, Nick Feamster, Matheus V. X. Ferreira, Danny Yuxing Huang, and S. Matthew Weinberg. 2019. Selling a Single Item with Negative Externalities: To Regulate Production or Payments? *CoRR* (2019).

[8] Shuchi Chawla, Jason D. Hartline, and Robert D. Kleinberg. 2007. Algorithmic Pricing via Virtual Valuations. In *the 8th ACM Conference on Electronic Commerce (EC)*.

[9] Jay Pil Choi, Chaim Fershtman, and Neil Gandal. 2010. Network security: Vulnerabilities and disclosure policy. *The Journal of Industrial Economics* 58, 4 (2010), 868–894.

[10] Michal Feldman, David Kempe, Brendan Lucier, and Renato Paes Leme. 2013. Pricing public goods for private sale. In *Proceedings of the fourteenth ACM conference on Electronic commerce*. ACM, 417–434.

[11] Don Fullerton. 1997. Environmental Levies and Distortionary Taxation: Comment. *The American Economic Review* 87, 1 (1997), 245–251. http://www.jstor.org/stable/2950868

[12] Nima Haghpanah, Nicole Immorlica, Vahab Mirrokni, and Kamesh Munagala. 2013. Optimal auctions with positive network externalities. *ACM Transactions on Economics and Computation* 1, 2 (2013), 13.

[13] Garrett Hardin. 1968. The tragedy of the commons. *science* 162, 3859 (1968), 1243–1248.

[14] Sergiu Hart and Noam Nisan. 2017. Approximate revenue maximization with multiple items. *J. Economic Theory* 172 (2017), 313–347. https://doi.org/10.1016/j.jet.2017.09.001 Preliminary version in EC 2012.

[15] Jason Hartline, Vahab Mirrokni, and Mukund Sundararajan. 2008. Optimal marketing strategies over social networks. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 189–198.

[16] Jason D Hartline. 2013. Mechanism design and approximation. *Book draft. October* 122 (2013).

[17] Philippe Jehiel, Benny Moldovanu, and Ennio Stacchetti. 1996. How (not) to sell nuclear weapons. *The American Economic Review* (1996), 814–829.

[18] Howard Kunreuther and Geoffrey Heal. 2003. Interdependent security. *Journal of risk and uncertainty* 26, 2-3 (2003), 231–249.

[19] William Lehr and Jon Crowcroft. 2005. Managing shared access to a spectrum commons. In *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*. IEEE, 420–444.

[20] Frank Li, Zakir Durumeric, Jakub Czyz, Mohammad Karami, Michael Bailey, Damon McCoy, Stefan Savage, and Vern Paxson. 2016. You've Got Vulnerability: Exploring Effective Vulnerability Notifications.. In *USENIX Security Symposium*. 1033–1050.

[21] Frank Li and Vern Paxson. 2017. A large-scale empirical study of security patches. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2201–2215.

[22] David Martimort and Wilfried Sand-Zantman. 2016. A mechanism design approach to climate-change agreements. *Journal of the European Economic Association* 14, 3 (2016), 669–718.

[23] Vahab S Mirrokni, Sebastien Roch, and Mukund Sundararajan. 2012. On fixed-price marketing for goods with positive network externalities. In *International workshop on internet and network economics*. Springer, 532–538.

[24] Juan-Pablo Montero. 2008. A simple auction mechanism for the optimal allocation of the commons. *American Economic Review* 98, 1 (2008), 496–518.

[25] Roger B Myerson. 1981. Optimal auction design. *Mathematics of operations research* 6, 1 (1981), 58–73.

[26] Nicky Woolf. 2016. DDoS attack that disrupted internet was largest of its kind in history, experts say. *The Guardian, https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet* (2016).

[27] Elissa M. Redmiles, Michelle L. Mazurek, and John P. Dickerson. 2018. Dancing Pigs or Externalities? Measuring the Rationality of Security Decisions. *To appear in ACM Conference on Economics and Computation* (2018).

[28] Paul Seabright. 1993. Managing local commons: theoretical issues in incentive design. *Journal of economic perspectives* 7, 4 (1993), 113–134.

[29] Martin L Weitzman. 1974. Prices vs. quantities. *The review of economic studies* 41, 4 (1974), 477–491.

Revisiting Mobile Advertising Threats with MAdLife

Gong Chen
Georgia Institute of Technology
gchen@ece.gatech.edu

Wei Meng
Chinese University of Hong Kong
wei@cse.cuhk.edu.hk

John A. Copeland
Georgia Institute of Technology
jcopeland@ece.gatech.edu

ABSTRACT

Online advertising is one of the primary sources of funding for content, services, and applications on both web and mobile platforms. Mobile in-app advertisements are implemented on top of existing web technologies with the same ad-serving model (i.e., users - publishers - ad networks - advertisers). Even so, in-app advertising is different from traditional web advertising. For example, malicious mobile app developers can generate fraudulent ad clicks in an automated fashion, but malicious web publishers have to leverage bots to launch click fraud. In spite of using the same underlying web infrastructure, these ad threats behave differently on different platforms.

Existing works have separately studied click fraud and malvertising in mobile settings. However, it is not known if there is a strong relationship between these two dominant threats. In this paper, we develop an ad collection framework – MAdLife– on Android to capture all in-app ad traffic generated during each ad's entire lifespan. We revisit both threats in a fine-grained manner with MAdLife to determine the relationship between them. Furthermore, MAdLife also allows us to explore other threats related to landing pages.

We analyzed 5.7K Android apps crawled from Google Play store, and collected 83K ads and their landing pages with MAdLife. 58K ads land on a web page, which is similar to traditional web ads. We discovered 37 click-fraud apps, and revealed that 1.49% of the 58K ads are related to malvertising. We also found a strong correlation between fraudulent apps and malicious ads. Specifically, over 15.44% of malicious ads originate from the fraudulent apps. Conversely, 18.36% of the ads displayed in the fraudulent apps are malicious, compared to only 1.28% found in the rest apps. Due to fraudulent apps, users are much more (14x) likely to encounter malvertising ads. Additionally, we discovered 243 popular JavaScript snippets embedded by over 10% of the landing pages are malicious. Finally, we also present the first analysis on inappropriate mobile in-app ads.

CCS CONCEPTS

• **Information systems → Online advertising**; • **Security and privacy → Software and application security**.

KEYWORDS

Online Advertising; Mobile Apps; Ad Fraud; Malvertising; Measurement

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.4089729

ACM Reference Format:
Gong Chen, Wei Meng, and John A. Copeland. 2019. Revisiting Mobile Advertising Threats with MAdLife. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.4089729

1 INTRODUCTION

Online digital advertising is the primary way to monetize content, services, and applications (apps) on the Internet. Companies like Google and Facebook generated 85-95% of their revenues through running the largest digital advertising platforms in Q2 2018 [1, 21]. Traditionally, these ad platforms, or ad networks, display advertisements (ads) from advertisers on a publisher's website that is part of the network. This practice is known as web advertising, which has generated over $35B in revenue per year in the last five years according to recent IAB reports [28]. Meanwhile, web advertising has rapidly expanded to mobile platforms. Mobile advertising revenue experienced a substantial (7x) growth from $7.1B in 2013 to $49.9B in 2017.

Mobile in-app ads are usually implemented on top of existing web technologies by rendering an HTML ad within a mobile app's WebView. Although sharing many underlying technologies, mobile in-app advertising is different from traditional web advertising. A website includes a JavaScript snippet to display ads, whereas a mobile app embeds a custom SDK to load the JavaScript code and ads in a special WebView – AdView. To date, ad blocking solutions, such as DNS66[1], DISCONNECT[2], NoMoAds [43], [6], are not widely used by real users for in-app ads, but ad blockers are very popular for traditional web advertising. As a result, the abusive practices in web advertising, such as *click fraud* and *malvertising*, exhibit different characteristics on mobile platforms. In particular, a web browser allows an ad script to detect automatically generated fraudulent ad clicks. Therefore, malicious publishers have to leverage bots instead of real users to automate ad clicks, which can be easily detected by ad networks. On the contrary, mobile users need to interact with HTML ads through their app's user interface (UI). This enables a malicious app developer to automatically generate fraudulent ad clicks, which are difficult for the restricted JavaScript running in the AdView to detect. Therefore, it is easier to launch click fraud from a genuine user device with the help of UI automation on mobile platforms. Malvertising on both platforms behaves very similarly. By clicking on an ad, a user may be brought to a malicious page that serves drive-by-downloads or scams. In particular, drive-by downloads have been the primary form of malvertising for web advertising, according to the study in [31]. Scams are, however, the dominating malvertising form for mobile in-app ads [39].

The abusive practices in mobile advertising have been explored by prior works. Crussell et al. examined click fraud in each app's

[1] https://f-droid.org/en/packages/org.jak_linux.dns66/
[2] https://disconnect.me

first/main activity by running in an emulator without UI intervention [15]. However, their work did not reveal how the fraudulent apps initiate automatic ad clicks. Rastogi et al. [39] detected malvertising cases by collecting and scanning redirect chains using VirusTotal[3]. However, their work did not look into JavaScript code loaded from external sources, which can also be malicious. Overall, both frameworks are time consuming for app collection. Also, click fraud and malvertising may happen together, and these tools cannot be used to study both threats altogether.

To overcome the above limitations, we present MAdLife, a framework for studying mobile in-app advertising. It can record all necessary data generated during each ad's entire lifespan (i.e., request → load/render → click → redirect → land) on Android. Specifically, we modify the Android WebView to collect pre-click data. After each ad is completed, we automatically trigger a click through lightweight UI automation, and then redirect the post-click traffic to our data collection app. The two datasets allow us to conduct a more fine-grained study on click fraud and malvertising. We can also investigate other related threats in mobile in-app advertising. Usually, advertisers' landing pages embed external JavaScript code. Unlike the prior work on mobile malvertising, we reveal that the external JavaScript snippets can also be malicious even though the embedding landing pages are considered benign. Besides, following up with two trending news event related to political advertising on Facebook [13, 41], we recognize that the contents appear in advertising have been receiving more and more public concerns. Thus, we also take a first attempt to analyze inappropriate content for mobile in-app ads. Specifically, by employing content analysis techniques, we are able to detect and classify inappropriate ads into two categories: 1) policy non-compliant [20, 24, 25, 46], which are ads that do not comply with the policies of ad networks (e.g., including age-restricted content); and 2) controversial, which are ads that are generally believed inappropriate to some population groups (e.g., promoting online gambling).

We used MAdLife to collect 84K ads from 5.7K Android apps from January to February in 2018. First, we identified that 37 apps had committed click fraud, and 1.49% of ads were malicious according to the VirusTotal query results on their landing pages. Second, we discovered a strong statistical relation between click-fraud apps and malicious ads. On one hand, over 15.44% of infected ads originated from those fraudulent apps. On the other hand, 18.36% of the ads displayed by click-fraud apps are malicious, whereas in non click-fraud apps only 1.25% of ads are malicious. Therefore, users of the click-fraud apps are much more (14x) likely to be exposed to malicious ads than other users. Also, we found over 32 cryptojacking ads in our dataset. Moreover, we discovered 243 (0.21%) of the 115K external JavaScript snippets are unsafe. They are very popular and were used in over 10% of the landing pages in our dataset. Finally, we identified that around 8% of ad landing pages were inappropriate – 7.9% are non-compliant with ad policies, and 0.266% are controversial.

Our contributions are threefold:

1) We developed MAdLife– the first framework to monitor an ad' entire life span on the mobile platform. As a result, we are able to build a panoramic view for digital ads.

2) We explored the abusive practices involved in mobile in-app advertising, and demonstrated a strong statistical relation between click fraud and malvertising. We discovered that due to click fraud, mobile app users are much more likely to experience malvertising. Moreover, we also found several emerging threats (i.e, cryptojacking ads) in mobile in-app advertising.

3) We are the first to extend the research scope of analyzing landing pages from two aspects. First, we discovered that a few unsafe external JavaScript code may occur in a great number of landing pages. Second, we classified inappropriate ads into two categories (i.e., policy non-compliant, and controversial).

In summary, previous works treated click fraud and malvertising as two independent mobile ad threats, and thus studied them separately. In contrast, MAdLife allows us to study mobile ad threats comprehensively. Therefore, researchers can have a broader view of mobile ad threats and thus build a safer mobile ad ecosystem.

2 BACKGROUND

2.1 Android WebView

Android apps are usually required to include custom AdViews, and the underlying implementation of most AdViews is based on WebView. While transparent to app developers and mobile users, the nature of the WebView component has been changed a lot with the evolution of Android OS. Since Android 4.4, it has moved from using the WebKit rendering engine to sharing the same Chromium-based codebase with Chrome for Android. In Android 5.0, the component came out as a standalone APK, where mobile users can find its updates on Play Store. Starting from Android 6.0, only prebuilt WebView versions are shared within the Android Open Source Project (AOSP). Afterward in Android 7.0, the Chrome APK is used for provide and render WebView. Recently, the Safe Browsing feature has been available in WebView with Android 8.0.

Based on Chromium source tree, WebView depends on a C++-compiled shared library and a source set. Within the source set, Java code encapsulates C++ and provides an AwContents object. By adding logcat messages for function calls within AwContents, we can thus use adb to communicate with an Android emulator and view the device log in real-time.

As a View object that displays webpages, WebView cannot be used standalone without an Activity component. With UI Automation tools, developers can access, identify, and manipulate the UI elements. However, such tools cannot access the Document Object Model (DOM) within a WebView. Unlike ads in a browser where researchers can directly resolve ad URLs from their DOM trees, when an ad WebView receives a touch event, it triggers an AwContents object's `onTouchEvent()` first, and then consecutively tops down to the Blink rendering engine for resolving the embedded URL from its DOM tree.

2.2 Mobile Advertising

Serving in-app ads is a pervasive approach to app monetization for *publishers*/*developers*. In order to get continuous revenue, app developers register their apps, and app store links to one or several *ad networks*. In turn, the developers receive a unique identifier per ad space, all related *ad libraries*, which fetch ads with the resolved URLs at runtime, and common guidelines. For example, an ad's

[3]https://www.virustotal.com

standard Cascading Style Sheets (CSS) size, which is different from a smartphone's resolution size, could be recommended. While developers have rights to choose ad types, including video ads and display ads (i.e., small-size banner ads, and full-screen interstitial ads) shown on their apps and earn from displaying/clicking ads, *advertisers* sit on the other end to provide funds for spreading out the content they wish to publicize. Together with innocent users downloading and launching apps, all other players, including publishers/developers, ad networks/ad libraries, and advertisers, form a perfect ecosystem in the mobile settings [9].

However, such an ecosystem may temporarily be interrupted, when a user launches an app, the app requests an ad for a vacant space from the linked ad network, but the ad network has nothing in its inventory when the app is launched. As a result, the user may see a void ad slot. In the case that ads are not loaded within the WebView, we call it *incapable ad loading*. Otherwise, different *ad syndications* are created by ad networks in order to fill the gap in time. Therefore, when the subscribed ad network does not find an ad for the ad slot, the ad request will be forwarded to syndicated ad networks. Imaginably, communications between the app and multiple ad networks under the ad syndication lead to a longer *redirect chain*, before arriving at the final *landing page*. Also, an ad may not be properly displayed at a WebView, due to network overload or ad networks' backend algorithms. We call that *improper ad rendering*. As for *app install ads*, their final destination may end up with Play Store.

2.3 Ad Threats

Click fraud. In order to earn more, unscrupulous app developers use click fraud to programmatically trigger fake ad clicks.
Malvertising. Due to untrusted ad networks, or unwanted redirect chains, malvertising may result in either drive-by downloads, or scams and phishing (e.g., pornography).
Vulnerable External JavaScript Code. Usually, advertisers use JavaScript snippets to track user behaviors, or more. However, landing pages containing vulnerable external JavaScript code may be exposed to both security and privacy risks.
Ad Inappropriateness. Ad networks regulate advertised content, but untrusted advertisers can still smuggle inappropriate ads. It is infelicitous, especially for children. That's why Unity Ads [47] talk about COPPA [12] within their common guidelines.

2.4 Public APIs

VirusTotal aggregates over 60 scanners, and offers APIs to analyze files/URLs. A rationale behind using a cluster of antivirus scanners is to fight against mistaken detection by individual scanning tool, or so called *false positive*.
Google Cloud APIs[4] uses REST calls to automate different workflows. We used the Natural Language API to reveal the structure and meaning of text from different aspects (e.g., "entity analysis" and "content classification"). Also, we called the Vision API to find similar images and web entities as well as detect explicit content (i.e., adult, medical, spoof, and violence), respectively.

[4]https://cloud.google.com/apis/

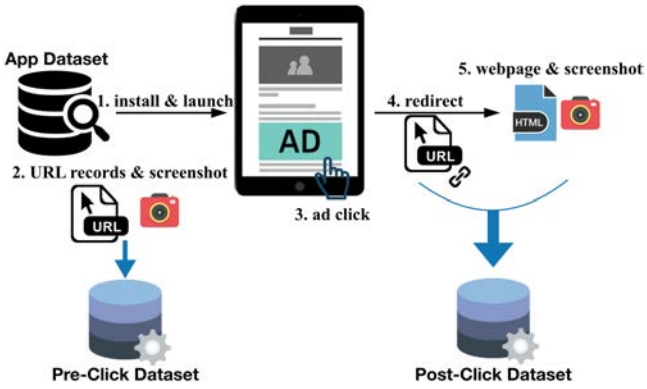


Figure 1: Workflow for data collection

3 METHODOLOGY

In this section, we present the design and implementation of MAdLife (Section 3.1), the steps we took to select our app dataset (Section 3.2), and the statistics about that dataset (Section 3.3).

3.1 MAdLife

MAdLife is a data collection framework on a mobile platform for in-app ads. It can record all necessary data generated within each ad's entire lifespan (i.e., request → load/render → click → redirect → land). Figure 1 demonstrates the workflow of MAdLife. First, MAdLife uses the AndroidViewClient[5] tool (version 13.6.0) to install and launch an app automatically. Afterwards, an ad is returned from a remote host. Second, MAdLife listens to the ad traffic at the same time, and then stores the *pre-click* data into a table in our database. Third, MAdLife clicks the ad. Fourth, the embedded ad URL is sent to our data collection app – Depot, and then redirected to the landing page. Fifth, MAdLife stores the *post-click* data into another table in our database. Finally, MAdLife stops all launched activities after the post-click data is collected. A timeout is set to stop the data collection when no ad is loaded or rendered.

We designed and implemented MAdLife on top of an Android emulator – Genymotion[6] 2.11. We did not select the AOSP emulators and those provided by Android Studio, because apps running in these emulator can receive only test ads for the AdMob SDK, which is the most widely seen ad SDK on Android. Instead, we equipped our testbed with Android 7.1 (API level 25) on Genymotion. After installing Play Store, we set the "Parental Controls" setting to the most restrictive level[7]. We took this step to ensure that we do not reach an age-restricted Play Store page via clicking the ads. We will show later in Section 4.3.3 that age-restricted ads cannot be avoided. Also, we enabled the ARM translator to run ARM apps on an x86 machine.

In order to collect the pre-click data, we modified the Android WebView (20 LOC in Java) to intercept a WebView's network traffic. This allowed us to determine if WebView is used to communicate

[5]https://github.com/dtmilano/AndroidViewClient
[6]https://www.genymotion.com
[7]"Apps & games": Everyone, "Movies": G, "TV": C, and restrictions on: "Books" and "Music".

Table 1: Our collected datasets

Pre-click Data		package name, ad size, ad screenshot, pre-click URLs, pre-click time
Post-click Data	Browser	screenshot of the landing page, HTML source of the landing page, URL redirect chain, post-click time
	Play Store	screenshot of the landing page, maturity rating

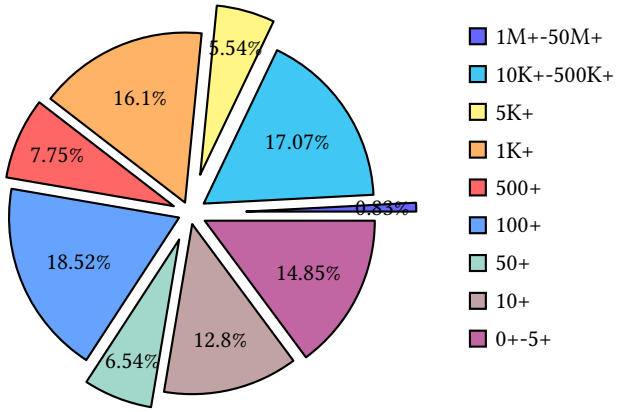


Figure 2: Distribution by app downloads

with a known ad host. After an ad is returned from a remote host, MAdLife saves the ad's screenshot and logs all pre-click URLs when several methods[8] related to ad rendering are called.

In order to collect the post-click data, we replaced each ad navigation URL with a custom URL scheme (i.e., MAdLife://) to direct the navigation to Depot, instead of the default mobile web browser. MAdLife generates an ad click to let the app switch to Depot or Play Store for post-click data collection. In case of switching to Depot, the app will take a screenshot of the landing page, save its HTML source, and records the redirects if there are any. However, it cannot directly identify which app initiates the ad click. To solve this problem, we link the pre-click data and post-click data based on their timestamps. In the case of switching to Play Store, MAdLife will take a screenshot of the Play Store page, and log the advertised app's maturity rating (e.g., Everyone, Teen, Mature 17+).

MAdLife also handles special cases where some ads require users to click on a control button or require more than one click to trigger a redirection. In particular, almost all full-screen interstitial ads require a click on a control button (e.g., "INSTALL", "Visit Site", "Learn More", and "Click Now"). Further, a few banner ads require to click twice, first to display a control button, and second to trigger the redirection. Therefore, MAdLife calls AndroidViewClient's `dump()` to discover the button with keywords, and then programmatically clicks such ads. The record for an ad click consists of two parts, as shown in Table 1.

[8]Methods include `loadUrl()`, `didFinishNavigation()`, `onTouchEvent()`, `didFinishLoad()`, `didStopLoading()`, `shouldIgnoreNavigation()`, `shouldInterceptRequest()`, and `loadDataWithBaseUrl()`.

3.2 App Selection

Initially, we crawled 143K free Play Store apps in December 2016 and March 2017. Due to the fact that not all of these apps have ads, it is not necessary to analyze the entire app dataset. Additionally, we were not able to analyze all of them due to our limited computing power. Thus, we decided to narrow down our analysis space by applying an app selection procedure. Also, it is not practical to analyze all ads displayed within one app, which requires complex UI automation. Given that the subsequent requests are similar to their first ad-related HTTP request for 94.7% of the top 3K ad-supported free apps [36], we focused on the apps that display ads in their first activities.

First, we excluded any apps whose first activities do not contain a WebView, with AndroidViewClient. We also removed apps with multiple WebViews in the first activities, because it cannot distinguish which WebView is clicked for data generation. As a result, 29.3K apps were kept.

Second, we further excluded apps that do not communicate with a known ad domain. In particular, we constructed a list of 1,183 known ad domains[9]. Only 11.8K apps were observed to communicate with one of the domains within 15 seconds. We set this 15-second threshold, since "most apps make the first ad request during launch time", as observed in [36]. However, we discovered that around 50% of the 11.8K apps did not display an ad in their first activity. It might be possible that those requests in the first activities were sent for tracking purposes, and the ads might be shown in other activities.

Last, we ruled out apps that do not exhibit any changes in their first activity after the ads are clicked, because users will normally be redirected to either a landing page or the Play Store app after an ad click. We finally retained 5.7K apps. Moreover, we tried the method in [19] to identify ads by using WebView sizes. However, we found that this method may not be robust because we identified more than 300 different WebView sizes in the 11.8K apps' first activities. For example, 320x50 is a standard CSS size for mobile ads, but we also observed ad sizes such as 320x49 and 320x51.

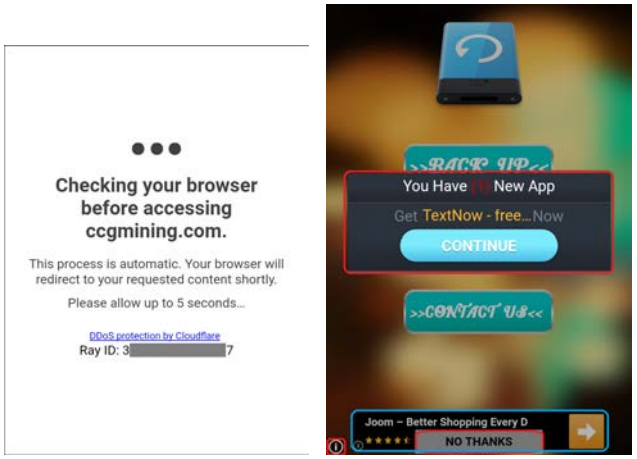
The selected 5.7K apps, developed by over 2.5K developers, exhibit high diversity, in terms of both app categories[10] and number of app downloads (Figure 2). According to [3, 4, 44], we believe that our app dataset is unbiased.

3.3 Ad Collection

We deployed MAdLife in two locations in the United States and one location in Canada between mid January and late February in 2018. According to [36], even though the subsequent requests are similar to their first ad request, ads returned by similar requests can be different. Therefore, in order to make sure our ad collection was unbiased, we crawled ads for each app in 10 runs at a time, of which each takes about 32-45 seconds to complete. It is worth pointing out that, MAdLife is more efficient, as it takes at least 2 minutes for other frameworks [15, 37, 39] to complete. At the end, we collected nearly 48GB of data, representing over 84K ads.

[9]https://adaway.org/hosts.txt, https://filters.adtidy.org/windows/filters/11.txt?id=11

[10]The apps are in 47 categories, including one in "Dating", four in "Casino", and over 300 in "Books & References", "Education", "Entertainment", "Lifestyle", "Music & Audio", "Personalization", and "Tools".



(a) Landing page redirect (b) Overlapped red and blue ads

Figure 3: Special cases encountered during our ad collection

During the ad collection phase, we encountered a great number of special cases (e.g., "incapable ad loading", and "improper ad rendering"). Additionally, we encountered two special situations. First, an ad may have more than one landing page due to *landing page redirect*, as shown in Figure 3(a). Such a redirection would occur after the initial landing page is loaded in Depot. As a result, MAdLife could collect two post-click records for that ad, but in such cases, we retain only the final landing page as the post-click record.

Second, the pre-click data we collected may sometimes come from different sources. Overlapped ads, a kind of ad fraud behavior in [32], may be the cause. For example, as shown in Figure 3(b), a sample app may satisfy all the criteria we define in Section 3.2. It contains a full-screen ad WebView (red) and another WebView (blue) which is covered by the top ad WebView. The pre-click data we collected from this app could contain ad traffic from the top ad WebView as well as background traffic from the blue WebView. We are not able to distinguish the exact WebView URLs from all the labeled network traffic in the pre-click data. It is a limitation of our framework. Therefore, we only used pre-click URLs for identifying the communications with one of the ad domains.

4 EVALUATION

In this section, we talk about how to preprocess our collected initial data, then analyze the security issues in mobile advertising, and finally study new threats related to advertisers.

4.1 Ad Dataset

Our pre-click data matched 133 rules of 97 ad networks in the list of ad-related domains. For example, Leadbolt uses four different domains (i.e., *leadbolt.net*, *leadboltads.net*, *leadboltmobile.net*, and *leadboltapps.net*). The matched ads in our pre-click data are in 103 distinct sizes. Meanwhile, we observed Google's ad domains (i.e., *admob.com*, and *doubleclick.net*) in over 90% of all ads. As for our post-click data, we got 58,876 unique redirect chains, which totaled 203,783 unique URLs. Excluding URLs from well-known domains (e.g., *doubleclick.net*, *google.com*, *facebook.com*, *amazon.com*, and *yahoo.com*), we still have 56,914 URLs left for our malvertising analysis (Section 4.2.2). The landing pages collected in Depot belong to nearly 3.4K distinct advertisers.

In order to conduct further analysis, we matched both pre- and post-click data. In [15], the authors used a fixed time window to group all data generated by each Android app. However, we could not apply a fixed time window, since the time of each ad collection varied. Therefore, we devised a heuristic strategy to match pre-click ads and their post-click landing pages. For the pre-click data of a particular ad, its post-click data should be generated within 45 seconds in our data collection process. Therefore, we gradually increased the window size from 32 to 45 seconds until a match is found. Finally, we got over 83K matched ads after linking the pre- and post-click data. In particular, 25,764 ads landed at Play Store, and 57,880 ads ended up with Depot.

4.2 Click Fraud & Malvertising

Here we revisited the two prominent ad threats. After revealing their own trends, we further explore their interconnection.

4.2.1 *Click Fraud.*

Identifying click fraud is more or less related to detecting fraudulent app developers and their apps. Ad networks provide their own WebView-based AdView. In our settings, if an app automatically clicks an ad right after the app starts, the foreground running view may have already been switched to Depot before MAdLife calls AndroidViewClient's `dump()` to log the pre-click data. MAdLife would take a screenshot of the landing page as usual. However, MAdLife may also accidentally record the landing page in both the pre- and post-click data of an ad. Therefore, we took both situations into account: 1) post-click data from the unmatched collection, where we found 356 apps from 575 cases; and 2) pre-click data from the matched dataset, whose screenshots have the same resolution size as Depot, and where we got 38 apps from 132 cases. Both situations totaled 372 unique apps. We run our test with these samples at a slower speed in order to properly trigger click fraud. Still, 81 apps fell into one of the above situations. We could run such tests recursively, until their click-fraud behaviors were confirmed. But after our manual inspection we found 37 click-fraud apps. These click-fraud apps experienced more than 660K total downloads, and two of them obtained over 100K downloads each.

According to Table 2, we saw that click fraud is still not a well-solved problem. Surprisingly, one of the seven fraudulent app developers owns 30 out of the 37 apps. After checking with VirusTotal, we obtained three benign apps within them. We further looked into their network traffic by running our framework again. Our test showed that all samples have only three origins, and two of the three sources are fed by the apps' local files, including those 30 apps.

4.2.2 *Malvertising.*

We used VirusTotal to analyze the 56,914 URLs to determine if the associated ads are malicious or not. Specifically, we got 1,127 positive URL cases, in which 248 were flagged by at least three scanners. We call these cases *malicious* URLs. These 248 URLs belonged to 65 unique domains, and also resulted in 199 redirect

Table 2: MAdFraud [15] vs. MAdLife (click fraud)

	MAdFraud	MAdLife
# Samples	130K from third-party markets + 35K malware	143K from Play Store
# Positive Cases	13 from third-party markets + 3 from Play Store + 6 malware	37 from Play Store

Table 3: Rastogi et al. [39] vs. MAdLife (malvertising)

	[39] (US & CN)	MAdLife (US & CA)
# Samples	600K from 5 markets	5.7K from Play Store
# Collected Links	1M (US)+415K (CN)	203.7K (US & CA)
# Malicious URLs	948 (US)+1,475 (CN)	248 (US & CA)
# Unique Domains	64 (US)+139 (CN)	65 (US & CA)

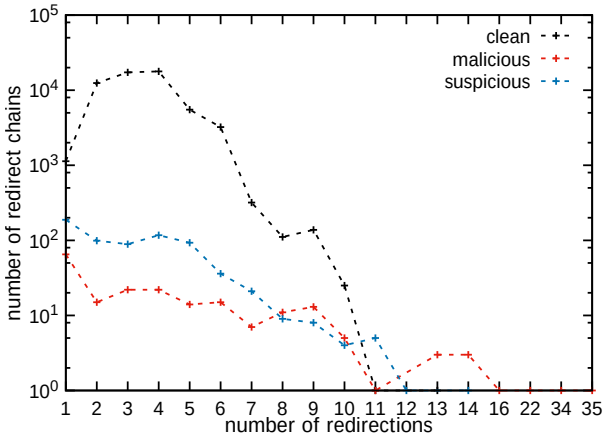


Figure 4: Statistics about different types of redirect chains

chains. The rest of those 1,127 URLs are flagged by one or two scanners. We call them *suspicious* URLs. They came from 147 unique domains, and resulted in the other 669 redirect chains[11]. Therefore, 1.49% of the 57,880 ads were from malicious or suspicious redirect chains. In Table 3, comparing with the findings from [39] in 2016, we found relatively more malvertising domains. However, we analyzed less apps in our experiment and got less links crawled from mobile ads. Therefore, we believe that more attackers are getting into this battlefield, and the situation related to malvertising is becoming worse.

Afterwards, we looked into the results from four aspects. First, we made an observation similar to the ones in [31, 39] that, longer redirect chains are more likely to be malicious. It is even more evident with our dataset. For the three kinds of redirect chains, we plot their relations between number and length in Figure 4. Accordingly, while only two cases with all benign URLs have more than 10 hops, the length of a malicious chain can even reach 35 redirections.

Second, only 32.16% of the malicious, and 64.87% of the suspicious redirect chains had non-blank landing pages. Moreover, two of the

[11] Among these 879 links flagged by less than three scanners, we excluded those URLs appeared in the previously mentioned 199 redirect chains.

Table 4: Malvertising traffic from the click-fraud apps

	Malicious	Suspicious
# Post-click	199	669
# Pre-click	187	580
# Unmatched Post-click	12	89
# Matched Pre-click	11	22
Percentage	11.56%	16.59%
Overall Percentage	15.44%	

locations where we collected ads used the Palo Alto Firewall[12], which blocked 26 ad clicks. 14 of these 26 problematic cases are malicious, and 11 have no redirections. During our ad collection process, no drive-by-download case was encountered.

Third, we discovered 92 from the 199 malicious, and 166 advertisers from the 669 suspicious redirect chains, respectively. There were 236 unique advertisers in total.

Last, malicious and suspicious redirect chains came from 134 and 263 apps, respectively. Also, 74 of the malicious , and 133 of the suspicious apps were detected by VirusTotal. We found that, 168 unique apps getting at least one positive from VirusTotal contributed to malvertising.

4.2.3 *Correlation.*

We found statistical correlations between click fraud and malvertising. According to Table 4, after giving equal opportunities to run target apps, we found that 134 malvertising cases (15.44%) came from the click-fraud apps, including 11.56% of the malicious and 16.59% of the suspicious redirect chains, respectively. Furthermore, as over 730 ads are generated by the 37 click-fraud apps in the dataset, these apps have a 18.36% chance to initiate malvertising. Also, since the other 734 malvertising cases came from the rest 57,150 ads, the non click-fraud apps merely had a 1.28% chance to initiate malvertising. As a result, a user of a click-fraud app, is 14.34x more likely to be shown a malicious ad compared to a user of a legitimate app.

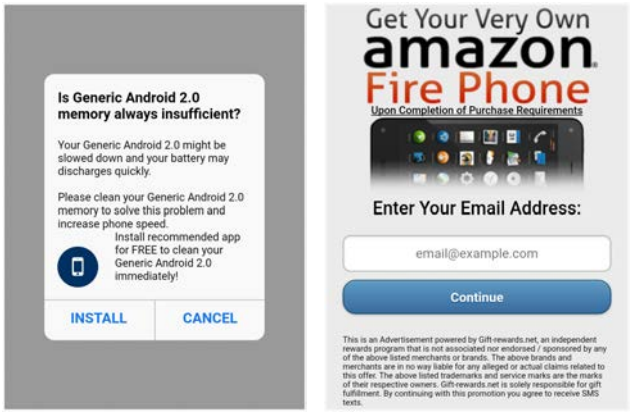
Why are the two ad-related security issues more likely correlated? A mobile user needs to interact with an HTML ad through the app's UI. It enables attackers to generate fraudulent ad clicks automatically, which is difficult to detect for the restricted JavaScript running in the app's WebView. Such attackers may usually select ad networks with poor SDK implementations and vulnerable scrutiny processes for malvertising. Therefore, ad networks should take on more responsibility to mitigate such risks, as they are responsible for strengthening their SDKs and scrutinizing malvertising instances.

4.2.4 *Scam Cases.*

Although scam cases have been well studied [39], we found new and interesting observations by looking into the HTML files of our scam cases. Figure 5(a) illustrates the first case, an ad for a fake anti-virus app. No matter whether users click "Install" or "Cancel", the same malicious link will be triggered.It reveals that, ad viewers are more likely to enter a malicious website with such a trick.

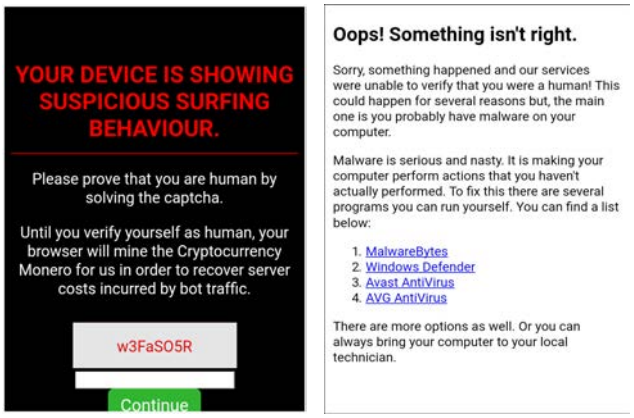
Figure 5(b) shows an unexpected prize and lottery scam, which is alleged to offer a free phone. In our dataset, we found two ads

[12] http://www.paloalto-firewalls.com



(a) Ad with deceptive download (b) Ad with fraudulent reward

Figure 5: Scam examples



(a) Drive-by cryptomining (b) Deceptive cryptojacking

Figure 6: Cryptojacking examples

with the same image. However, these screenshots were loaded from two different malicious domains (i.e., *primerewardz.com* and *premiumrewardsusa.com*), with no redirections. Users should refrain from surrendering their private information, as some scams ask for emails to get a fraudulent reward.

```
<script>
  ...
  var miner = new CoinHive.User("...", "tt", {throttle:0});
  miner.start(CoinHive.FORCE_EXCLUSIVE_TAB)
</script>
```

Listing 1: Drive-by cryptomining script found in Figure 6a

```
<script type="text/javascript" defer="" async="" src="https://load
      .jsecoin.com/load/.../0/0/"></script>
<iframe src="https://claimers.io/a-mining?address=..." style="
      visibility: hidden;"></iframe>
```

Listing 2: Two cryptomining scripts found in Figure 6b

4.2.5 *Cryptojacking Cases.*

Here we show two cryptojacking instances, in which attackers secretly mine cryptocurrency with victims' computing power. While we discovered over 25 obvious drive-by cryptomining cases, as seen in Figure 6(a), we also found 7 subtle examples like that in Figure 6(b).

The former case, with up to 9 redirections, was not from a single domain, but 13 different ad domains[13]. However, all these redirections finished at one of the two websites (i.e., *rcyclmnr.com*, and *rcyclmnrepv.com*). Code 1 shows that the COINHIVE[14] script was used in 6(a). We submitted the embedded JavaScript file[15] to VirusTotal, where 33 out of 58 scanners detected this script. We later found out that our discovery was confirmed by both Symantec [30] and Malwarebytes [40].

The latter case includes at most 2 redirections, originates from the same domain (i.e., *ezanga.com*), and ends up with *dropped-click.com*. A closer inspection revealed two interesting tactics used by these developers: 1) the website provided benign links for users to click; and 2) the website used three different cryptomining scripts. According to the HTML files, the other two cases were undergoing landing page redirects; therefore, we had no further information for those webpages. One example was with *coinhive*, two instances were with *jsecoin.com*, and another two cases were with *claimers.io*. See Code 2. To the best of our knowledge, we were the first to find these samples, while no scanner of VirusTotal flagged the *claimers.io* domain as a potential risk yet.

Furthermore, we looked into the issues from two aspects. First, we examined the 49 apps where these cryptojacking sample were found. Surprisingly, 21 of the sample apps were benign after scanning with VirusTotal. Thus, the malvertising cases found in these apps were solely due to untrusted ad networks.

Second, fraudulent app developers contributed to the issues as well. Specifically, the aforementioned developer, who owned 30 click-fraud apps, has raised our attention again due to the 6 apps among our revisited samples. Our first impression of the developer's apps is about click fraud: after launching any of these apps, users would sometimes be automatically redirected to a landing page showing arbitrary content (e.g., automotive sales, lottery rewards, and pornographic chat rooms). We found that a local file, named EXIT.HTML, was called in each of the 6 apps. Although the file is totally clean under VirusTotal, it calls the `doStartAppClick()` method at every app start, to automatically trigger a call to visit an ad URL[16]. Ironically, this URL also seemed benign under VirusTotal, but it could serve as an entry point for redirecting users to different sites, either dirty or clean. We loaded the URL in a web browser, and were redirected to a couple of automatic malware downloading pages[17]. At the time of this writing, these apps were still listed on

[13] Domains of the first link in a redirect chain include: *leadbolt.net*, *tc-clicks.com*, *apperol.com*, *shootmedia-hk.com*, *despiteracy.com*, *leadzuaf.com*, *spxtraff.com*, *smartoffer.site*, *bestperforming.site*, *mobcampaign.site*, *tracknet.site*, *topcampaign.site*, and *ads.gold*

[14] https://coinhive.com

[15] Its VirusTotal SHA-256 is 5d514880ad502302dd4bf0ef8da5d38356385d1c43689f6739f6771ed7a4ef73

[16] The URL, *http://pub.reacheffect.com/ra/878/1042/p/m/%7Bcampaign_id%7 D/CA*, was taken down in April, 2018

[17] Their VirusTotal SHA-256's are 0c6e40eb1c3b00de1c72f22dec5cffddc3df66672360f79d54d9922c018f4aa6 and 1654cf25365332198c84f7e1e17b237abf0447a97e18800cc349e91cc2eb9a71

Play Store, although on January 15th, 2018 the developer switched to other ad networks, and then finally disabled the auto click feature.

To sum up, the tricks of fraudulent app developers and of untrusted ad networks were continuously evolving. It is not limited to only deceiving users but also developing new techniques to hijack their computing power.

4.3 Other Threats

Besides the previous two well-known threats, we have also studied two new types of ad threats on a large scale: 1) malicious external JavaScript on landing pages; and 2) inappropriate ads.

4.3.1 *Data Preparation.*

In order to facilitate our analysis, we removed duplicates from both landing pages and screenshots. First, as the HTML files of the same website we crawled might contain different metadata, we were not able to use the traditional cryptographic hashing algorithm to classify websites. Therefore, we used the CETD algorithm [45] to extract text content and anchored links from webpages. We finally retained 12,036 distinct documents.

Second, the same website with a countdown timer may be photo-taken differently in two ad instances. Therefore, in order to retain useful content, we filtered out pictures with pure white or black background. The file size of such images is usually less than 15KB in size. Afterwards, we consecutively employed three image fingerprinting methods[18] (i.e., wavelet hashing, perception hashing, and difference hashing), which generate similar output hashes between two cognate input files. The process produced 6,337 unique images.

4.3.2 *Malicious External JavaScript Code.*

Advertisers may embed JavaScript code from their own host or a third-party host in their landing pages. Some external scripts could be malicious. However, VirusTotal scanners would report the results for the only submitted URL, not including its embedded external scripts. Especially, it is very hard for VirusTotal scanners to detect a known malicious JavaScript snippet after code obfuscation. As a result, this approach would miss malicious content presented to users on a landing page. Therefore, we further leveraged VirusTotal to scan all external scripts of each landing page.

After extracting 146K unique external script URLs and distilling 115K URLs starting with HTTP(S) from almost 58K HTML files, we found only 243 positively flagged URLs (0.21%) with VirusTotal. Surprisingly, these scripts were used in more than 5,880 landing pages (i.e., 10.16% of the 57,880 HTML file). Among these 5,880 landing pages, we discovered 53 malvertising cases and 5,827 benign landing pages. Lastly, after running the CETD algorithm, we got 1,384 unique documents (i.e., 11.5% of the 12,036 distinct documents).

External JavaScript code can be loaded either from the same domain or from third-party domains. For example, we witnessed that a "benign" landing page consisted of more than 41 positively flagged URLs for the advertiser's own JavaScript code. That is to say, illicit advertisers may use such tricks to avoid traditional scrutiny techniques. Third-party JavaScript code is usually dynamically loaded; therefore, it may be subject to change on the third-party side. In our dataset, we discovered that a few snippets from Google-registered domains (e.g., gstatic.com, and ytimg.com) were also positively flagged. Table 5 shows the statistics of positively flagged URLs.

Table 5: Top 10 domains with positively flagged URLs

# Occurrences	# Unique URLs	Domain
409	6	gstatic.com
287	4	ytimg.com
84	1	adroll.com
64	1	engagio.com
50	1	parastorage.com
44	1	bootstrapcdn.com
18	1	bkrtx.com
14	1	googleapis.com
13	1	roberthalfgcs.com
10	1	ucarecdn.com

4.3.3 *Ad Inappropriateness.*

The issue of inappropriate ads on the Internet has existed for a long time but has not yet been fully explored. Legislatures in the United States have created various acts (e.g., COPPA [12], CIPA [11]) to safeguard children online, including from ad views/clicks. In addition, well-known ad networks [20, 23–25, 46] also establish content policies for advertisers. But, self regulation is not enough. Here we took the first step to analyze and classify inappropriate ad content on a large scale.

We used the Google Cloud Natural Language API to get known entities and content categories. After identifying documents with the same known entities, we reduced duplicate files with a list of the same entities from 12,036 to 7,067. These documents contain all 27 level-1 categories[19]. We selected a few sensitive categories (i.e., Adult, and Sensitive Subjects) for our analysis. Afterwards, we used known entities to recursively select other sensitive categories.

Determining the inappropriateness of an ad could be very subjective. Therefore, we adhered to the following guidelines based on ad network policies and public opinion to identify inappropriate ads. First, AdWords does not allow "collecting personal information from children under 13 or targeting interest content to children under the age of 13" [25]. Thus, we listed all ads related to age-restricted content. Second, well-known ad networks [17, 18, 22] announced their plan of blocking cryptocurrency-related ads, one after another, as the content is often associated with deception and fraud [38]. Thereafter, we labeled all related ads. Third, due to Russian agents-used political ads and the Cambridge Analytica scandal, Facebook received criticism and modified its policy for political ads [5]. Therefore, we put all political ads into the "public concerns" category. Accordingly, we classified inappropriate ads into two severity levels:

Policy Compliance: adult (e.g., pornography), criminal record, cryptocurrency, drug (e.g., marijuana), prize (e.g., fake awards), security (e.g., fake antivirus), store (e.g., age-restricted product)

Public Concerns: age (e.g., tattoo), extramarital affair (e.g., Ashley Madison), gambling (e.g., casino), health (e.g., plastic surgery), political campaign (e.g., voting), privacy (e.g., phone number)

The result is depicted in Table 6. In summary, 345 out of 7,067 unique ad landing pages are in the "Policy Compliance" category,

[18] https://github.com/JohannesBuchner/imagehash

[19] https://cloud.google.com/natural-language/docs/categories

Table 6: Inappropriate ads and their categories

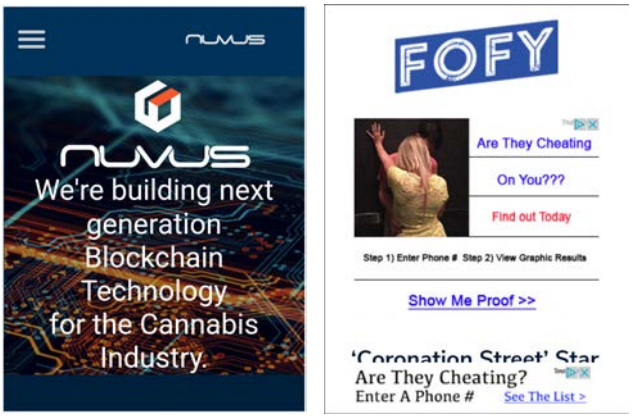
Ad Category			Unique Ads (#)	Full Dataset (%)
Policy Compliance	345 & 7.9%	adult	14	0.33%
		criminal record	7	1.44%
		cryptocurrency	146	63.49%
		drug	3	0.17%
		prize	46	2.43%
		security	6	0.33%
		store	123	31.81%
Public Concerns	24 & 0.266%	age	2	1.30%
		extramarital affair	3	0.26%
		gambling	5	20.78%
		health	3	5.19%
		political campaign	8	33.12%
		privacy	3	31.82%

and 24 are in the "Public Concerns" category; 4,577 (7.9%) out of 57,880 ad landing impressions are in the "Policy Compliance" category, and 154 (0.26%) are in the "Public Concerns" category.

We also leveraged the Google Vision API on our screenshots to detect offensive contents and find web entities with similar images. We found the API performed quite well in detecting pornography. The API was also good at finding similar images that contain very few characters. For example, a prize wheel image with barely any characters was flagged as potentially dangerous by the API, because similar images were found at some malicious websites. However, it was difficult to accurately detect any other kinds of offensive ad content using the Vision API in general. Thus, we used only the Natural Language API in our analysis.

4.3.4 Case Studies.
Unlike click fraud and malvertising which affect all users, inappropriate ads may merely influence a specific subset of population. Here we exemplify two cases to show the aggravating circumstances, as depicted in Figure 7. First, along with their price volatility, cryptocurrencies attract attention from people all over the world. As a result, the entire industry suddenly booms. During our ad collection, we collected mobile ads of 39 different cryptocurrencies and of 27 cryptocurrency-related reports from 15 online media. One of such ads, shown in Figure 7(a), publicized its cryptocurrency for the cannabis industry. As we learned from the Internet, marijuana is not legalized at the federal level in the United States and in Canada [48] at the time of our experiment. Such ads should thus be prohibited.

Second, it is even worse that *nested ads* can bypass all restrictions. Figure 7(b) illustrates the landing page of a blog website, where another ad is nested. Although ad networks can scrutinize matches between ads and their landing pages, their arms may not be able to reach at nested ads. To the best of our knowledge, we did not find any policies to directly regulate such a situation. As a result, inappropriate ad content may be broadcast within landing pages subscribed to well-known ad networks. Moreover, we considered it as an adware, because for more than 30 FOFY ads we collected, only ads were shown within the screenshots but their real content displayed underneath. Accordingly, the situation of inappropriate ads is getting worse.



(a) Controversial topics in an ad (b) Ads within an adware

Figure 7: Examples for ad inappropriateness

4.3.5 Takeaways.
Our studies revealed that embedded external scripts in landing pages should also be examined for two reasons: 1) illicit advertisers may use this approach to escape from ad networks' security lookups; and 2) third-party JavaScript links may change their backend code for their own interests. Therefore, rather than only checking inappropriate ad content, ad networks should also scrutinize the JavaScript files embedded by advertisers.

Last but not the least, after Google blocked cryptocurrency ads in June 2018 [18], we utilized MAdLife again to crawl ads in the next month. We did not find any cryptocurrency ad in this crawl. In reverse, after banning cryptocurrency ads for a few months, Facebook gave Coinbase a privileged position to allow its crypto-related ads again in July 2018 [29]. Ad policies thus may be more inconsistent than what we expected. More importantly, nested ads could be used to bypass new content-based regulations.

5 DISCUSSION

Ethics. Our ad collection experiment may inflate advertisers' budgets, like other studies [7, 10, 31, 36, 39, 50]. However, we tried to reduce the impact on the real ad ecosystem in our research. On one hand, MAdLife clicked less than 15 ads within each app and visited the landing page of each advertiser for less than 25 times on average. Therefore, each individual app developer's income and each individual advertiser's cost would not be significantly affected. On the other hand, clicking the ads is necessary for studying these ad threats. Although our research might have inflated the budgets of some advertisers, our findings can benefit the whole ad ecosystem in the long run from a broader perspective. We believe that a better ad ecosystem will be more cost-efficient for advertisers. Moreover, we designed our experiment in a way that MAdLife's artificial behaviors can be easily detected by ad networks (e.g., 10 successive clicks within one app and from the same IP addresses). A responsible ad network shall be able to detect them as invalid clicks, and thus invalidate any charges incurred to advertisers.

Limitations. First, we run the experiment on API 25 with Genymotion 2.11.0. Nowadays, Android 8.0 (API level 26) has enabled

safe browsing on WebView. If ad networks adapt the feature, the malvertising issues could be somewhat mitigated. Second, our pre-click data may contain URLs other than those we expect, since we cannot detect WebViews underneath a full-screen WebView and separate each WebView's logcat traffic. Third, to the best of our knowledge, the Google Cloud APIs are among the best APIs available on the market. We also have tested APIs provided from two other services, which did not outperform Google's APIs. However, we cannot measure the accuracy of Google's APIs as we are unable to manually build ground truth for thousands of images and webpages. Last, Google asks that, ads displayed in family apps are expected to be consistent with their maturity ratings [26]. In [10], authors talk about this issue. However, whether the parental control settings are on or off, children are able to access all installed apps on a device. Therefore, our studies only focus on inappropriate ads in general instead of with age-restricted apps.

6 RELATED WORK

Ad Collection. Due to the fact that ad content is dynamically generated, various methodologies are used to crawl ads. Collecting online ads is relatively easy when not in mobile settings. In online advertising, researchers can either run scripts with their add-ons [7, 31] or build a Selenium-based crawler [50], with Alexa's top-ranked website lists. Also, the researchers can either intercept HTTP traffic [31, 50] or only harvest ad-related visual elements [7]. Nevertheless, ads' life spans can be easily deduced without triggering ad clicks. Whereas, in mobile advertising, researchers are still able to analyze HTTP traffic [15, 36, 39], but it's more difficult to track ads' lifespans. For example, since no user interaction (except for click fraud) is involved in [15], their studies stop at ad loading/rendering. Besides, both [36] and [39] consider every touch event as a starting point.

Click Fraud. Ad fraud can be conducted with [2, 32] or without [15, 35] human intervention. Likewise, researchers reveal that such attacks in web browsers [2, 35] are technically more advanced than with handheld devices [15, 32]. While [35] depicts two clickbot families, [2] traces from an IP back to the conspiracy with DNS hijacking. In mobile advertising, the term can be subdivided into display fraud [32] and click fraud [15]. Other than these, researchers in [51] detected duplicate clicks in pay-per-click streams with two bloom filter related algorithms.

Malvertising. Malvertising lurks at the pre-click phase, but is exposed at the post-click phase. Researchers, who collect ads in online advertising, focus on either ad networks [50] or redirect chains [31]. Likewise, such research works in mobile advertising result in the exploration of either ad networks [27, 52] or redirect chains [39]. Because of ad syndication, we believe that the use of redirect chains would be more beneficial. With the help of studies like [33], authorities will be able to take down malvertising networks without a case-by-case investigation.

Dangerous JavaScript Detection. The studies of detecting dangerous JavaScript code are numerous. JSAND [14] conducts classification based on static and dynamic features, and instruments JavaScript runtime environments to detect the execution of malicious scripts. Prophiler [8] statically analyze features of the HTML page, of the embedded JavaScript code, and of the associated URL, and examines malicious content on massive webpages with obfuscated JavaScript. Similarly, ZOZZLE [16] leverages features associated with JavaScript context to detect unobfuscated exploits. As for code obfuscation, Sharif et al. [42] revealed that conditional code obfuscation could be used to hinder malware analysis.

Content Analysis. Nowadays, contextual advertising is used to target users. [49] describes a feature-based keyword extraction system for online contextual advertising. In order to analyze web content, [34] uses the Alchemy API to label all crawled webpages, and categorizes them into different topic categories. Likewise, we use Google Cloud's Natural Language API to analyze our dataset, besides using the CETD algorithm [45] to only encompass web content and anchored links. Similar to finding resolved ad URLs, content scraping in the mobile settings is not so easy as that in a browser. Therefore, both [32] and [37] capture third-party apps' page contents on Windows-based mobile platforms. While the former extracts page information with the UI extraction channel, the latter instruments app binaries to insert custom logging code. Unlike our work that evaluates ad inappropriateness in general, [10] studies whether ads are consistent with host apps' maturity ratings on a small scale that is below 4K ads. Also, their "topic classification" part is not well elaborated; therefore, we are thus unable to make any technical comparison.

7 CONCLUSION

While implemented on top of existing web technologies, mobile in-app ads face similar threats in different forms. In this paper, we implemented MAdLife– a data collection framework on Android to record all necessary data generated within each ad's entire lifespan. By using MAdLife, we explored the abusive practices, including click fraud, malvertising, and other threats on landing pages, with 84K ads. We discovered 37 click-fraud apps and over 860 malvertising redirect chains. More importantly, we revealed that a user of a click-fraud app is 14.34x more likely to be shown with a malicious ad comparing with a user of a legitimate app. Further, we revealed that 250 widely used JavaScript snippets are actually harmful to users. These snippets were used on over 10% of our collected landing pages. Lastly, we identified that 8% ads are inappropriate, either not complying with ad policies or showing controversial content. We suggest that ad networks should, and only they can, take more effective measures to protect mobile users and themselves from the above ad threats.

REFERENCES

[1] Alphabet. 2018. Google Second Quarter 2018 Results. https://abc.xyz/investor/pdf/2018Q2_alphabet_earnings_release.pdf. [Online; accessed 17-February-2019].

[2] Sumayah A Alrwais, Alexandre Gerber, Christopher W Dunn, Oliver Spatscheck, Minaxi Gupta, and Eric Osterweil. 2012. Dissecting ghost clicks: Ad fraud via misdirected human clicks. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*.

[3] Appbrain. 2018. Android app download statistics on google play. https://www.appbrain.com/stats/android-app-downloads. [Online; accessed 17-February-2019].

[4] Appbrain. 2018. Most popular google play categories. https://www.appbrain.com/stats/android-market-app-categories. [Online; accessed 17-February-2019].

[5] Daniel Arkin. 2018. Facebook announces major changes to political ad policies. https://www.nbcnews.com/tech/social-media/facebook-announces-majorchanges-political-ad-policies-n863416. [Online; accessed 17-February-2019].

[6] Michael Backes, Sven Bugiel, Philipp von Styp-Rekowsky, and Marvin Wißfeld. 2017. Seamless in-app ad blocking on stock android. In *2017 IEEE Security and Privacy Workshops (SPW)*. IEEE, 163–168.

[7] Paul Barford, Igor Canadi, Darja Krushevskaja, Qiang Ma, and S Muthukrishnan. 2011. Adscape: Harvesting and analyzing online display ads. In *Proceedings of the 21st International World Wide Web Conference (WWW)*. Seoul, Korea.

[8] Davide Canali, Marco Cova, Giovanni Vigna, and Christopher Kruegel. 2011. Prophiler: a fast filter for the large-scale detection of malicious web pages. In *Proceedings of the 20th International World Wide Web Conference (WWW)*. Hyderabad, India.

[9] Gong Chen, Jacob H Cox, A Selcuk Uluagac, and John A Copeland. 2016. In-depth survey of digital advertising technologies. *IEEE Communications Surveys & Tutorials* 18, 3 (2016), 2124–2148.

[10] Ying Chen, Sencun Zhu, Heng Xu, and Yilu Zhou. 2013. Children's exposure to mobile in-app advertising: An analysis of content appropriateness. In *2013 International Conference on Social Computing (SocialCom)*. IEEE, 196–203.

[11] Federal Communications Commission. 2018. Children's internet protection act. https://www.fcc.gov/consumers/guides/childrens-internet-protection-act. [Online; accessed 17-February-2019].

[12] Federal Trade Commission. 2018. Children's Online Privacy Protection Rule. https://www.ftc.gov/enforcement/rules/rulemaking-regulatory-reform-proceedings/childrens-online-privacy-protection-rule. [Online; accessed 17-February-2019].

[13] Josh Constine and Taylor Hatmaker. 2018. Facebook admits Cambridge Analytica hijacked data on up to 87M users. https://techcrunch.com/2018/04/04/cambridge-analytica-87-million. [Online; accessed 17-February-2019].

[14] Marco Cova, Christopher Kruegel, and Giovanni Vigna. 2010. Detection and analysis of drive-by-download attacks and malicious JavaScript code. In *Proceedings of the 19th International World Wide Web Conference (WWW)*. Raleigh, NC.

[15] Jonathan Crussell, Ryan Stevens, and Hao Chen. 2014. Madfraud: Investigating ad fraud in android applications. In *Proceedings of the 12th annual international conference on Mobile systems, applications, and services (MOBISYS)*. ACM, 123–134.

[16] Charlie Curtsinger, Benjamin Livshits, Benjamin G Zorn, and Christian Seifert. 2011. ZOZZLE: Fast and Precise In-Browser JavaScript Malware Detection.. In *Proceedings of the 20th USENIX Security Symposium (Security)*. San Francisco, CA.

[17] Megan Rose Dickey. 2018. Facebook is banning cryptocurrency and ico ads. https://techcrunch.com/2018/01/30/facebook-is-banning-cryptocurrency-and-ico-ads/. [Online; accessed 17-February-2019].

[18] Jillian D'Onfro. 2018. Google' will ban all cryptocurrency-related advertising. https://www.cnbc.com/2018/03/13/google-bans-crypto-ads.html. [Online; accessed 17-February-2019].

[19] Feng Dong, Haoyu Wang, Yuanchun Li, Yao Guo, Li Li, Shaodong Zhang, and Guoai Xu. 2017. Fraudroid: An accurate and scalable approach to automated mobile ad fraud detection. *arXiv preprint arXiv:1709.01213* (2017).

[20] Facebook. 2018. Advertising Policies. https://www.facebook.com/policies/ads/. [Online; accessed 17-February-2019].

[21] Facebook. 2018. Facebook Second Quarter 2018 Results. https://investor.fb.com/investor-news/press-release-details/2018/Facebook-Reports-Second-Quarter-2018-Results/default.aspx. [Online; accessed 17-February-2019].

[22] Jon Fingas. 2018. Twitter will ban most cryptocurrency ads. https://www.engadget.com/2018/03/26/twitter-bans-most-cryptocurrency-ads. [Online; accessed 17-February-2019].

[23] Google. 2018. Ad policies. https://support.google.com/youtube/topic/30084?hl=en&ref_topic=2972865. [Online; accessed 17-February-2019].

[24] Google. 2018. AdMob & AdSense Policies. https://support.google.com/admob/answer/6128543?hl=en. [Online; accessed 17-February-2019].

[25] Google. 2018. AdWords policies. https://support.google.com/adwordspolicy. [Online; accessed 17-February-2019].

[26] Google. 2018. Parent guide to google play. https://support.google.com/googleplay/answer/6209547?hl=en. [Online; accessed 17-February-2019].

[27] Michael C Grace, Wu Zhou, Xuxian Jiang, and Ahmad-Reza Sadeghi. 2012. Unsafe exposure analysis of mobile in-app advertisements. In *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*. ACM, 101–112.

[28] IAB. 2019. IAB Internet Advertising Revenue Report. https://www.iab.com/insights/iab-internet-advertising-revenue-report-conducted-by-pricewaterhousecoopers-pwc-2. [Online; accessed 17-February-2019].

[29] Jose Antonio Lanz. 2018. Facebook restores coinbased cryptocurrency ads. https://ethereumworldnews.com/facebook-restores-coinbase-cryptocurrency-ads/. [Online; accessed 17-February-2019].

[30] Hon Lau. 2017. Browser-based cryptocurrency mining makes unexpected return from the dead. https://www.symantec.com/blogs/threat-intelligence/browser-mining-cryptocurrency. [Online; accessed 17-February-2019].

[31] Zhou Li, Kehuan Zhang, Yinglian Xie, Fang Yu, and XiaoFeng Wang. 2012. Knowing your enemy: understanding and detecting malicious web advertising. In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS)*. Raleigh, NC.

[32] Bin Liu, Suman Nath, Ramesh Govindan, and Jie Liu. 2014. {DECAF}: Detecting and Characterizing Ad Fraud in Mobile Apps. In *Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. Seattle, WA.

[33] Hesham Mekky, Ruben Torres, Zhi-Li Zhang, Sabyasachi Saha, and Antonio Nucci. 2014. Detecting malicious http redirections using trees of user browsing activity. In *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 1159–1167.

[34] Wei Meng, Byoungyoung Lee, Xinyu Xing, and Wenke Lee. 2016. Trackmeornot: Enabling flexible control on web tracking. In *Proceedings of the 25th International World Wide Web Conference (WWW)*. Montreal, Canada.

[35] Brad Miller, Paul Pearce, Chris Grier, Christian Kreibich, and Vern Paxson. 2011. What's clicking what? techniques and innovations of today's clickbots. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*. Springer, 164–183.

[36] Suman Nath. 2015. Madscope: Characterizing mobile in-app targeted ads. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services (MOBISYS)*. ACM, 59–73.

[37] Suman Nath, Felix Xiaozhu Lin, Lenin Ravindranath, and Jitendra Padhye. 2013. SmartAds: bringing contextual ads to mobile apps. In *Proceeding of the 11th annual international conference on Mobile systems, applications, and services (MOBISYS)*. ACM, 111–124.

[38] Stephen O'Neal. 2018. Big tech are banning crypto and ico ads - is there a reason to panic? https://cointelegraph.com/news/big-tech-are-banning-crypto-and-ico-ads-is-there-a-reason-to-panic. [Online; accessed 17-February-2019].

[39] Vaibhav Rastogi, Rui Shao, Yan Chen, Xiang Pan, Shihong Zou, and Ryan Riley. 2016. Are these Ads Safe: Detecting Hidden Attacks through the Mobile App-Web Interfaces.. In *Proceedings of the 2016 Annual Network and Distributed System Security Symposium (NDSS)*. San Diego, CA.

[40] Jérôme Segura. 2018. Drive-by cryptomining campaign targets millions of android users. https://blog.malwarebytes.com/threat-analysis/2018/02/drive-by-cryptomining-campaign-attracts-millions-of-android-users/. [Online; accessed 17-February-2019].

[41] Scott Shane. 2017. Ads Bought by Russia on Facebook. https://www.nytimes.com/2017/11/01/us/politics/russia-2016-election-facebook.html. [Online; accessed 17-February-2019].

[42] Monirul I Sharif, Andrea Lanzi, Jonathon T Giffin, and Wenke Lee. 2008. Impeding Malware Analysis Using Conditional Code Obfuscation.. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS)*. San Diego, CA.

[43] Anastasia Shuba, Athina Markopoulou, and Zubair Shafiq. 2018. NoMoAds: Effective and Efficient Cross-App Mobile Ad-Blocking. *Proceedings on Privacy Enhancing Technologies* 2018, 4 (2018), 125–140.

[44] Statista. 2018. Most popular google play app categories as of 1st quarter 2018, by share of available apps. https://www.statista.com/statistics/279286/google-play-android-app-categories/. [Online; accessed 17-February-2019].

[45] Fei Sun, Dandan Song, and Lejian Liao. 2011. Dom based content extraction via text density. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 245–254.

[46] Twitter. 2018. Prohibited Content Policies. https://business.twitter.com/en/help/ads-policies/prohibited-content-policies.html. [Online; accessed 17-February-2019].

[47] Unity. 2018. Unity monetization service. https://unity3d.com/legal/monetization-services-terms-of-service. [Online; accessed 17-February-2019].

[48] Wikipedia. 2018. Legality of cannabis by country. https://en.wikipedia.org/wiki/Legality_of_cannabis_by_country. [Online; accessed 17-February-2019].

[49] Wen-tau Yih, Joshua Goodman, and Vitor R Carvalho. 2006. Finding advertising keywords on web pages. In *Proceedings of the 15th International World Wide Web Conference (WWW)*. Edinburgh, Scotland.

[50] Apostolis Zarras, Alexandros Kapravelos, Gianluca Stringhini, Thorsten Holz, Christopher Kruegel, and Giovanni Vigna. 2014. The dark alleys of madison avenue: Understanding malicious advertisements. In *Proceedings of the ACM Internet Measurement Conference (IMC)*. Vancouver, Canada.

[51] Linfeng Zhang and Yong Guan. 2008. Detecting click fraud in pay-per-click streams of online advertising networks. In *2008 The 28th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 77–84.

[52] Yajin Zhou, Zhi Wang, Wu Zhou, and Xuxian Jiang. 2012. Hey, you, get off of my market: detecting malicious apps in official and alternative android markets.. In *Proceedings of the 19th Annual Network and Distributed System Security Symposium (NDSS)*. San Diego, CA.

Modeling Relational Drug-Target-Disease Interactions via Tensor Factorization with Multiple Web Sources

Huiyuan Chen
Electrical Engineering and Computer Science
Case Western Reserve University
hxc501@case.edu

Jing Li
Electrical Engineering and Computer Science
Case Western Reserve University
jingli@cwru.edu

ABSTRACT

Modeling the behaviors of drug-target-disease interactions is crucial in the early stage of drug discovery and holds great promise for precision medicine and personalized treatments. The growing availability of new types of data on the internet brings great opportunity of learning a more comprehensive relationship among drugs, targets, and diseases. However, existing methods often consider drug-target interactions or drug-disease interactions separately, which ignores the dependencies among these three entities. Also, many of them cannot directly incorporate rich heterogeneous information from diverse sources.

In this work, we investigate the utility of tensor factorization to model the relationships of drug-target-disease, specifically leveraging different types of online data. Our motivation is two-fold. First, in human metabolic systems, many drugs interact with protein targets in cells to modulate target activities, which in turn alter biological pathways to promote healthy functions and to treat diseases. Instead of binary relationships of <drug, disease> or <drug, target>, a tighter triple relationships <drug, target, disease> should be exploited to better understand drug mechanism of actions (MoAs). Second, medical data could be collected from different sources (i.e., drug's chemical structure, target's sequence, or expression measurements). Therefore, effectively exploiting the complementarity among multiple sources is of great importance. Our method elegantly explores a <drug, target, disease> tensor together with complementarity among different data sources, thus improves prediction accuracy. We achieve this goal by formulating the problem into a coupled tensor-matrix factorization problem and directly optimize it on the nonlinear manifold. Experimental results on real-world datasets show that the proposed model outperforms several competitive methods. Our model opens up opportunities to use large Web data to predict drugs' MoAs in pharmacological studies.

CCS CONCEPTS

• **Computing methodologies** → **Factorization methods**; • **Applied computing** → **Health informatics**.

KEYWORDS

Tensor factorization; Multi-view learning; Manifold optimization; Grassmann manifold; Drug discovery; Disease analysis;

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313476

ACM Reference Format:
Huiyuan Chen and Jing Li. 2019. Modeling Relational Drug-Target-Disease Interactions via Tensor Factorization with Multiple Web Sources. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313476

1 INTRODUCTION

Targeted therapies and personalized treatments are the most promising approaches to treat complex human diseases such as cancer. Clear understanding of drugs' mechanism of actions (MoAs) is a critical step in drug discovery [21]. Traditional high-throughput screening methods are desirable to identify a new drug against a chosen target (most time a druggable protein) for a special disease. However, the process has been costly and lengthy. A conservative estimate is that it takes $ 2.87 billion and more than 10 years to bring a new drug into market [26]. On the other hand, statistical and machine learning models provide an alternative way to accelerate the process of understanding drugs' MoAs by mining bioinformatics, cheminformatics, and Web data sources [3, 4, 7, 16, 25, 32, 37]. Learning tasks have been proposed from different angles, such as exploring drug behaviors [15, 38, 46], assessing target activities [19, 34, 37], and understanding disease models [33, 44, 47]. Two of the most prominent formulations among these recent advancements are the drug-target [16] prediction and drug-disease prediction, also known as drug repositioning [25]. In these two tasks, researchers have attempted to collect a variety of omics data from scientific literature and online Web sources, and discover new interactions between drugs and targets/diseases through different statistical models [9, 24, 35, 46]. For example, a network-based inference method was proposed to infer new targets for known drugs by leveraging the drug-target bipartite network [13]. The method Decagon modeled drug polypharmacy side-effect via graph convolutional networks [46]. Another approach, PREDICT, integrated multiple drug-drug and disease-disease similarities to infer potential drug-disease interactions [20]. Approaches based on multiple kernel learning were developed to predict drug-target [30] or drug-disease [9] relationships, by integrating additional heterogeneous information sources.

However, despite the obvious connections, most existing work treat drug-disease and drug-target predictions as two independent tasks, which is viewed as a major shortcoming. Indeed, the therapeutic effect of drugs on a disease is through their abilities to bind and to modulate biological targets that involve the disease pathways, which in turn promote healthy function of the metabolic system and cure the disease [21]. Therefore, instead of a binary relationship such as <drug, disease> or <drug, target>, a strong

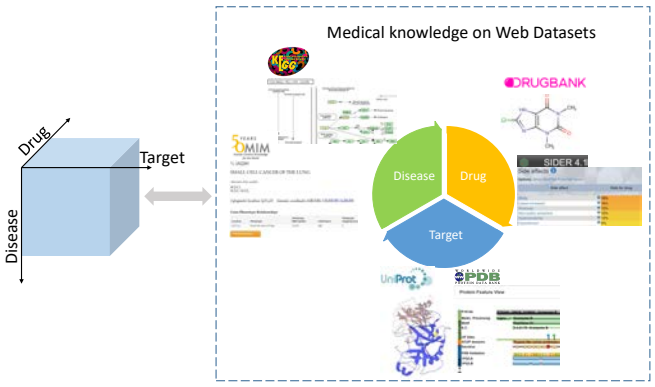


Figure 1: Illustration of DTD. Our model jointly explores the $drug \times target \times disease$ tensor along with rich existing medical knowledge on the Web.

triple relationship <drug, target, disease> should be considered to better model their interplay. In addition, the growing availability of new types of data on the web (Figure 1) brings new opportunities to learn a more comprehensive relationship among drugs, targets, and diseases [3, 4, 11, 30, 32, 45]. Incorporating such heterogeneous information can significantly improve our understanding of the underlying biological processes. For example, joint analysis of drugs' chemical structures, drugs' side-effects, and protein-protein networks can improve success rates of finding novel drug-target interactions [20, 30, 45].

In this work, we propose a tensor completion method, termed **DTD**, to model **D**rug-**T**arget-**D**isease interactions, with the help of existing data from the Web. DTD explicitly learns a third-order $drug \times target \times disease$ tensor using Tucker Decomposition [23]. In addition, to alleviate the data sparsity issue, DTD incorporates multiple auxiliary information sources of drugs, targets, and diseases. For example, in Figure 1, in addition to $drug \times target \times disease$ tensor, there are rich data on the Web describing drugs (e.g., chemical structures in DrugBank and side-effects in SIDER database), targets (e.g., protein sequence in Uniprot), and diseases (phenotype description in OMIM or KEGG). The tensor and multiple feature matrices are coupled in the "drug", "target", and "disease" mode, respectively. These feature matrices from Web sources are very useful to learn extra static information about each mode of the tensor. Fusing those datasets together can lead to better interpretations of the complex biological processes. To achieve this goal, DTD further explores the correlations among the latent matrices from the tensor and these multiple data sources via a coupled tensor-matrix factorization. This ensures that knowledge in the tensor aligns more closely with existing medical knowledge of each of the entities.

Another critical challenge is how to solve the tensor completion problem effectively. Because of the nonlinear and non-convex orthogonality constraints in the tensor Tucker model, the solutions of the popular HOSVD and HOOI Euclidean solvers are not unique, which makes it hard to couple with auxiliary information [23]. Motivated by the fast growing Riemannian optimization [1, 22, 43], we cast the coupled tensor-matrix factorization problem as a nonlinear program with the factor matrices constrained to the Grassmann manifold. Rather than a special non-uniqueness solution in Euclidean solvers, DTD obtains an equivalence class of matrices on Grassmann manifold, which leads to more meaningful subspace representations of factor matrices and can be well coupled with auxiliary information. Moreover, empirical studies have shown that nonlinear Riemannian solvers are significantly faster comparing to the Euclidean solvers [22, 43]. Our contributions are summarized as follows.

- We investigate the relationships of drug-target-disease by utilizing a tensor completion method. Integration of all these biological entities naturally leads to more meaningful interpretations of the results.
- We further explore the correlations among latent matrices from $drug \times target \times disease$ tensor and multiple data sources via a coupled tensor-matrix factorization, which ensures that the sparse tensor is supplemented using existing knowledge of the three entities.
- We directly optimize the tensor completion problem by adopting an optimization technique on Grassmann manifold, which obtains more reliable subspace representations of the tensor and can be well coupled with spectral embeddings of auxiliary information.
- We conduct extensive experiments on real-life datasets. Experimental results demonstrate DTD outperforms several state-of-the-art comparative models. It shows that DTD can effectively utilize rich information on the Web to ease the issue of sparsity.

The rest of the paper is organized as follows. Sec. 2 gives the background and task. Sec. 3 and Sec. 4 introduce DTD method in details. Sec. 5 evaluates the experimental performance. Sec. 6 provides a short review of related work. Sec. 7 concludes the paper.

2 BACKGROUND AND TASK DESCRIPTION

2.1 Tensor Algebra

In this study, we follow the notations introduced by Kolda and Bader [23]. We denote matrices by boldface uppercase letters (e.g., $\mathbf{A}$). *Tensors* are multidimensional arrays that extend the concept of matrices, and they are represented by boldface caligraphic letters (e.g., $\boldsymbol{\mathcal{X}}$). The *order* of a tensor is the number of its dimensions, also known as ways or modes. A *fiber* is a vector extracted from a tensor by fixing every index but one. A *slice* is a matrix extracted from a tensor by fixing all but two indices. Note that an N-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ reduces to a vector when $N = 1$, and a matrix when $N = 2$. The $(i_1, \ldots, i_N)$-th element of $\boldsymbol{\mathcal{X}}$ is denoted as $\boldsymbol{\mathcal{X}}_{i_1,\ldots,i_N}$. *Matricization*, also known as unfolding or flattening, is the process of reordering the elements of a tensor into a matrix. The mode-n matricization of an N-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is represented as $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times I_1 \ldots I_{n-1} \times I_{n+1} \ldots I_N}$ and is arranging the mode-n fibers of the tensor as columns of the long matrix. The *n-mode matrix product* of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ is denoted as $\boldsymbol{\mathcal{X}} \times_n \mathbf{U}$ with size of $I_1 \cdots \times I_{n-1} \times J \times I_{n+1} \cdots \times I_N$. We also give the definition of Tucker decomposition and coupled tensors for third-order tensors.

Definition 1. ***(Tucker Decomposition)***. *The Tucker decomposition is a form of higher-order PCA. It decomposes a tensor into a core tensor multiplied by a matrix along each mode. For a three-way tensor* $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$*, its Tucker decomposition is*

$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$$

where $\mathbf{A} \in \mathbb{R}^{I \times R_1}$, $\mathbf{B} \in \mathbb{R}^{J \times R_2}$ *and* $\mathbf{C} \in \mathbb{R}^{K \times R_3}$ *are the factor matrices (which are usually orthogonal) and can be regarded as the principal components in each mode. The tensor* $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ *is the core tensor and captures interactions between factor matrices.*

Definition 2. ***(Coupled Tensors)***. *If a tensor shares one or more modes with other matrices or other tensors, they are called coupled tensors or coupled tensor-matrices. For example, in a recommender system, a triple relationship user* $\times$ *movie* $\times$ *review tensor and a user* $\times$ *movie rating matrix are coupled because they share user and movie modes.*

2.2 Task Description

Our aim is to infer potential drug-target-disease interactions for rational drug repositioning. We formulate the task as a tensor completion problem. To be specific, the input can be organized as a three-way *drug* $\times$ *target* $\times$ *disease* tensor $\mathcal{X}$ of size $n_1 \times n_2 \times n_3$, in which n_1, n_2, and n_3 denote the number of drugs, targets, and diseases, respectively. An entry $\mathcal{X}_{ijk} = 1$ if drug i binds to target j and treats disease k. Otherwise, the entries are set to 0. In practice, we can only observe part of the tensor $\mathcal{X}$ and this partially observed tensor is denoted as $\mathcal{T}$. Our goal is to predict the potential concurrences of <drug, target, disease>, which can be achieved by completing tensor $\mathcal{X}$ given the incomplete tensor $\mathcal{T}$.

In real-world applications, tensor $\mathcal{T}$ is often sparse with a large number of unknown entries. Recovering tensor $\mathcal{X}$ is challenging when relying only on the observed tensor $\mathcal{T}$. Fortunately, for *drug* $\times$ *target* $\times$ *disease*, there exist many additional data sources on the Web to describe drugs, targets, and diseases. For example, for most drugs, their chemical structures and side-effects can be obtained from online databases, which represent different views of drugs. Datasets regarding targets and diseases are also available online. In most cases, one can further summarize such auxiliary data as drug-drug, target-target, and disease-disease similarity/kernel matrices, which can be incorporated in the learning process. Formally, let $\mathcal{S}_A = \{\mathbf{S}_A^{(1)}, \ldots, \mathbf{S}_A^{(n_a)}\}$ denote the similarity matrices constructed from n_a views of drugs. $\mathcal{S}_B = \{\mathbf{S}_B^{(1)}, \ldots, \mathbf{S}_B^{(n_b)}\}$ and $\mathcal{S}_C = \{\mathbf{S}_C^{(1)}, \ldots, \mathbf{S}_C^{(n_c)}\}$ are defined similarly from n_b views of targets and n_c views of diseases, respectively. All of them are assumed to be symmetric and non-negative. The details of their constructions will be presented in Sec. 5. The proposed DTD framework jointly explores the main tensor $\mathcal{X}$ together with multi-view auxiliary information $\mathcal{S}_A$, $\mathcal{S}_B$ and $\mathcal{S}_C$ to predict more meaningful interactions of drug-target-disease.

3 THE DTD MODEL

In this section, we propose an effective coupled tensor-matrix factorization formulation named DTD and show how different data sources can be included in a principled way. Notations used throughout the paper are summerized in Table 1.

Table 1: Main Notation

Symbol	Description
$\mathcal{X}$, $\mathcal{T}$	Full recovery tensor and observed tensor
$\mathcal{G}$	Core tensor in Tucker model
$\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$	Drug, target, disease factor matrix
$\mathbf{L}_A^{(i)}$, $\mathbf{A}^{(i)}$	Laplacian matrix and its spectral embedding from i-th view of drug auxiliary information
$\mathbf{L}_B^{(j)}$, $\mathbf{B}^{(j)}$	Laplacian matrix and its spectral embedding from j-th view of target auxiliary information
$\mathbf{L}_C^{(k)}$, $\mathbf{C}^{(k)}$	Laplacian matrix and its spectral embedding from k-th view of disease auxiliary information

3.1 Recover the Main Tensor

To complete the tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ described in Sec. 2.2, we adopt the Tucker Decomposition (Definition 1), which can be represented by the following optimization problem:

$$\min_{\mathcal{X}, \mathcal{G}, \mathbf{A}, \mathbf{B}, \mathbf{C}} \|\mathcal{X} - \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}\|_F^2 \quad s.t. \quad \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T}) \tag{1}$$

where $\|\cdot\|_F$ is the Frobenius norm. Recall that $\mathcal{G}$ is the core tensor; $\mathbf{A} \in \mathbb{R}^{n_1 \times r_1}$, $\mathbf{B} \in \mathbb{R}^{n_2 \times r_2}$, and $\mathbf{C} \in \mathbb{R}^{n_3 \times r_3}$ are the factor matrices with respect to the drug, target, and disease mode. Ω is the set that contains the indices of observed entities and $\mathcal{P}_\Omega(\cdot)$ keeps the entries in Ω and zeros out others [22, 41]. The equality constraint ensures that the corresponding elements of the recovering tensor $\mathcal{X}$ should match the observed elements in tensor $\mathcal{T}$.

In addition, the factor matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are orthogonal matrices, i.e., $\mathbf{A}^T\mathbf{A} = \mathbf{B}^T\mathbf{B} = \mathbf{C}^T\mathbf{C} = \mathbf{I}$ and $\mathbf{I}$ is the identity matrix. The orthogonal constraints ensure that matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are well defined on the so-called *Stiefel manifold* [1], which contains a set of $n \times m$ orthonormal matrices and is defined as:

$$St(m, n) = \{\mathbf{U} \in \mathbb{R}^{n \times m} | \mathbf{U}^T\mathbf{U} = \mathbf{I}_m\}$$

We will show the advantage of Stiefel manifold when coupling the tensor with auxiliary information.

3.2 Coupled with Auxiliary Information

To incorporate multi-view auxiliary information, we adopt the idea of spectral clustering, due to its flexibility and ease of implementation [28, 31]. We first give a brief introduction of spectral clustering. Suppose $\mathbf{S} \in \mathbb{R}^{N \times N}$ is the similarity/affinity matrix for N objects, where $\mathbf{S}_{ij}$ measures the similarity between object i and object j. One can then compute the normalized Laplacian matrix: $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$, in which $\mathbf{D}$ is the diagonal matrix with $\mathbf{D}_{ii} = \sum_{j=1}^{N} \mathbf{S}_{ij}$. The spectral clustering is to solve the following optimization:

$$\min_{\mathbf{U} \in \mathbb{R}^{N \times k}} \quad \langle \mathbf{U}\mathbf{U}^T, \mathbf{L} \rangle \quad s.t. \quad \mathbf{U}^T\mathbf{U} = \mathbf{I} \tag{2}$$

where $\langle \cdot, \cdot \rangle$ denotes the matrix inner product. $\mathbf{U}$ can be regarded as the low-dimensional spectral embedding of N objects. For clustering task, the k-means algorithm can be then applied to $\mathbf{U}$ to get the clustering indicators. Furthermore, the orthogonal constraint indicates that the spectral embedding $\mathbf{U}$ is also well defined on the Stiefel manifold. Therefore, it is reasonable to jointly consider the factor matrices in Eq. (1) together with the spectral embedding in Eq. (2) when performing the coupled tensor-matrix factorization in the sense that both of embeddings are on the Stiefel manifold.

For the drug mode of tensor $\mathcal{X}$, considering drug factor matrix $\mathbf{A} \in St(r_1, n_1)$ and its multi-view auxiliary information $\mathcal{S}_A = \{\mathbf{S}_A^{(1)}, \ldots, \mathbf{S}_A^{(n_a)}\}$, we extend the single-view spectral embedding to the follow multi-view co-training optimization function as [10, 28]:

$$\min_{\mathbf{A}, \mathbf{A}^{(i)} \in St(r_1, n_1)} \sum_{i=1}^{n_a} (\langle \mathbf{A}^{(i)} \mathbf{A}^{(i)^T}, \mathbf{L}_A^{(i)} \rangle + \|\mathbf{A}^{(i)} \mathbf{A}^{(i)^T} - \mathbf{A}\mathbf{A}^T\|_F^2) \quad (3)$$

where $\mathbf{L}_A^{(i)}$ is the normalized Laplacian matrix of $\mathbf{S}_A^{(i)}$ and $\mathbf{A}^{(i)}$ is the corresponding spectral embedding. The key idea behind Eq. (3) is that all the spectral embeddings $\mathbf{A}^{(i)}$ $(0 \le i \le n_a)$ from auxiliary information should be close to the drug factor matrix $\mathbf{A}$ since they all represent the same drugs. We achieve this by minimizing the disagreements $d(\mathbf{A}^{(i)}, \mathbf{A}) = \|\mathbf{A}^{(i)} \mathbf{A}^{(i)^T} - \mathbf{A}\mathbf{A}^T\|_F^2$. The reason for choosing $d(\mathbf{A}^{(i)}, \mathbf{A})$ is two-fold: (i) the reconstruction of similarity/kernel matrix $\mathbf{A}^{(i)} \mathbf{A}^{(i)^T}$ from spectral embedding is expected to be consistent with similarity $\mathbf{A}\mathbf{A}^T$ from tensor factor matrix, which is our assumption. (ii) we later show that the joint optimization is further defined on the Grassmann manifold, the quotient space of Stiefel manifold [1]. $d(\mathbf{A}^{(i)}, \mathbf{A})$ exactly measures the *geodesic distance* between two Grassmannian points $\mathbf{A}^{(i)}$ and $\mathbf{A}$ [14].

The same analysis can be applied to the target and disease factor matrices $\mathbf{B}$ and $\mathbf{C}$ with auxiliary information $\mathcal{S}_B$ and $\mathcal{S}_C$. We leave the details in the unified model in Eq. (4) later.

3.3 Overall Model

We propose the coupled tensor-matrix factorization model by combining the loss functions in Eq. (1) and Eq. (3). In addition to two functions, to better represent drugs, we further add l_1-norm on $\mathbf{A}\mathbf{A}^T$ [28]. We argue that this type of regularization does make sense. It has been shown that the new affinity/similarity matrix $\mathbf{A}^{(i)} \mathbf{A}^{(i)^T}$ in Eq. (3) implies the true membership of data clusters, and it is naturally *sparse* in an ideal case [28, 31]. Because the geodesic distance measurement $d(\mathbf{A}^{(i)}, \mathbf{A})$ is included in the overall objective function for minimization, the matrix $\mathbf{A}\mathbf{A}^T$ is also expected to be sparse. Similar arguments can be made for the target's and disease's factor matrices. Putting everything together, our method DTD tries to minimize the following onjective function:

$$\begin{aligned} \min f = {} & \|\mathcal{X} - \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}\|_F^2 + \rho \cdot (\|\mathbf{A}\mathbf{A}^T\|_1 + \|\mathbf{B}\mathbf{B}^T\|_1 + \|\mathbf{C}\mathbf{C}^T\|_1) \\ & + \alpha \cdot \sum_{i=1}^{n_a} (\langle \mathbf{A}^{(i)} \mathbf{A}^{(i)^T}, \mathbf{L}_A^{(i)} \rangle + \|\mathbf{A}^{(i)} \mathbf{A}^{(i)^T} - \mathbf{A}\mathbf{A}^T\|_F^2) \\ & + \beta \cdot \sum_{j=1}^{n_b} (\langle \mathbf{B}^{(j)} \mathbf{B}^{(j)^T}, \mathbf{L}_B^{(j)} \rangle + \|\mathbf{B}^{(j)} \mathbf{B}^{(j)^T} - \mathbf{B}\mathbf{B}^T\|_F^2) \\ & + \gamma \cdot \sum_{k=1}^{n_c} (\langle \mathbf{C}^{(k)} \mathbf{C}^{(k)^T}, \mathbf{L}_C^{(k)} \rangle + \|\mathbf{C}^{(k)} \mathbf{C}^{(k)^T} - \mathbf{C}\mathbf{C}^T\|_F^2) \\ s.t. \quad & \mathbf{A}, \mathbf{A}^{(i)} \in St(r_1, n_1);\ \mathbf{B}, \mathbf{B}^{(j)} \in St(r_2, n_2);\ \mathbf{C}, \mathbf{C}^{(k)} \in St(r_3, n_3) \\ & \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T});\ \mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times r_3} \end{aligned} \quad (4)$$

where parameter ρ controls the sparsity of factor matrices. And α, β and γ represent the impact of auxiliary information on each mode of tensor, i.e., how important such knowledge is to improve the performance. At first glance, the objective function f is complicated with nonlinear and non-convex orthogonality constraints. Next, we provide an effective algorithm to solve the problem.

4 OPTIMIZATION ALGORITHM

In this section, we develop an alternating minimization algorithm to optimize the objective function in Eq. (4). To be specific, the objective function f is successively minimized with respect to one variable while fixing others until convergence. To deal with orthogonality constraints, we directly optimize it on a Grassmann manifold, which is an emerging topic in nonlinear programming that leverages the smooth geometry of the search space with guaranteed convergence [1, 22, 39, 43].

Updating core tensor $\mathcal{G}$: The objective with respect to $\mathcal{G}$ is:

$$\min f(\mathcal{G}) = \|\mathcal{X} - \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}\|_F^2$$

The core tensor $\mathcal{G}$ is obtained as the closed form solution

$$\mathcal{G} = \mathcal{X} \times_1 \mathbf{A}^T \times_2 \mathbf{B}^T \times_3 \mathbf{C}^T \quad (5)$$

Updating factor matrices A, B and C: For matrix $\mathbf{A}$, the objective $f(\mathbf{A})$ can be regarded as an unconstrained manifold optimization problem on the Stiefel manifold:

$$\begin{aligned} \min_{\mathbf{A} \in St(r_1, n_1)} f(\mathbf{A}) = {} & \|\mathcal{X} - \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}\|_F^2 + \rho \|\mathbf{A}\mathbf{A}^T\|_1 \\ & + \alpha \sum_{i=1}^{n_a} \|\mathbf{A}^{(i)} \mathbf{A}^{(i)^T} - \mathbf{A}\mathbf{A}^T\|_F^2 \end{aligned}$$

The above optimization can be further converted into the following [23]:

$$\min_{\mathbf{A} \in St(r_1, n_1)} f(\mathbf{A}) = -\|\mathbf{A}^T \mathbf{W}\|_F^2 + \alpha \sum_{i=1}^{n_a} \|\mathbf{A}^{(i)} \mathbf{A}^{(i)^T} - \mathbf{A}\mathbf{A}^T\|_F^2 + \rho \|\mathbf{A}\mathbf{A}^T\|_1 \quad (6)$$

where $\mathbf{W} = \mathbf{X}_{(1)}(\mathbf{C} \otimes \mathbf{B})$, in which $\mathbf{X}_{(1)}$ is the mode-1 matricization of tensor $\mathcal{X}$ and $\otimes$ is the Kronecker product. However, simply optimizing problem (6) on Stiefel manifold may result in identifiability issue [1]. We next analyze problem (6) more deeply.

Consider the r_1-order group $\mathcal{S}(r_1) = \{\mathbf{Q} \in \mathbb{R}^{r_1 \times r_1} | \mathbf{Q}^T \mathbf{Q} = \mathbf{I}\}$ that contains all the $r_1 \times r_1$ orthogonal matrices. With $\mathcal{S}(r_1)$, we can define an equivalent relation $\sim$ on the Stiefel manifold $St(r_1, n_1)$ in the sense that $\mathbf{A} \sim \mathbf{A}'$ if there exists a $\mathbf{Q} \in \mathcal{S}(r_1)$ such that $\mathbf{A} = \mathbf{A}'\mathbf{Q}$. The quotient space of Stiefel manifold $St(r_1, n_1)$ under this equivalence relation is exactly the *Grassmann manifold* $Gr(r_1, n_1)$ [1], which consists of all the r_1-dimensional subspaces in n_1-dimensional Euclidean space $\mathbb{R}^{n_1}$ $(0 \le r_1 \le n_1)$. Moreover, it is interesting to observe that for any $\mathbf{Q} \in \mathcal{S}(r_1)$, we have the following invariance property:

$$f(\mathbf{A}) = f(\mathbf{A}\mathbf{Q})$$

Above equivalence indicates that the function $f(\mathbf{A})$ is independent from the choice of basis spanned by $\mathbf{A}$ and it is thus well defined on the Grassmann manifolds. Instead of optimizing $f(\mathbf{A})$ on Stiefel manifold, a better strategy is thus to regard the problem (6) as an unconstrained Grassmann manifold optimization problem:

$$\min_{\mathbf{A} \in Gr(r_1, n_1)} f(\mathbf{A}) = \underbrace{-\|\mathbf{A}^T \mathbf{W}\|_F^2 + \alpha \sum_{i=1}^{n_a} \|\mathbf{A}^{(i)} \mathbf{A}^{(i)^T} - \mathbf{A}\mathbf{A}^T\|_F^2}_{\mathcal{J}_1(\mathbf{A})} + \rho \|\mathbf{A}\mathbf{A}^T\|_1 \quad (7)$$

Problem (7) can then be efficiently solved by using standard gradient descent algoirthms on the Grassmann manifold such as

Riemannian conjugate gradient descent algorithm or trust region algorithm [1]. For Grassmann manifold, its Riemannian gradient is the projection of Euclidean gradient into relevant tangent space of the manifold. We next compute the Euclidean gradient of function $f(\mathbf{A})$.

For the first two terms in the problem (7), we have:

$$\nabla \mathcal{J}_1(\mathbf{A}) = -2\mathbf{W}\mathbf{W}^T\mathbf{A} + \alpha \sum_{i=1}^{n_a}(4\mathbf{A}\mathbf{A}^T\mathbf{A} - 4\mathbf{A}^{(i)}{\mathbf{A}^{(i)}}^T\mathbf{A}) \tag{8}$$

The third term $\|\mathbf{A}\mathbf{A}^T\|_1$ in objective function (7) is not differentiable when the elements of $\mathbf{A}\mathbf{A}^T$ are zeros, we consider the sub-differential. According to the chain rule:

$$\mathrm{vec}\left(\frac{\partial \|\mathbf{A}\mathbf{A}^T\|_1}{\partial(\mathbf{A})}\right)^T = \mathrm{vec}(\mathrm{sgn}(\mathbf{A}\mathbf{A}^T))^T \frac{\partial \mathbf{A}\mathbf{A}^T}{\partial \mathbf{A}} \tag{9}$$

where $\mathrm{sgn}(\cdot)$ denotes sign function and $\mathrm{vec}(\cdot)$ is vectorize operator that stacks all columns of a matrix into a long vector. Also, from the partial equation: $\partial(\mathbf{A}\mathbf{A}^T) = (\partial\mathbf{A})\mathbf{A}^T + \mathbf{A}\partial(\mathbf{A}^T)$, we can get:

$$\begin{aligned}\partial\,\mathrm{vec}(\mathbf{A}\mathbf{A}^T) =& (\mathbf{A}\otimes\mathbf{I}_{n_1})\partial\,\mathrm{vec}(\mathbf{A}) + (\mathbf{I}_{n_1}\otimes\mathbf{A})\partial\,\mathrm{vec}(\mathbf{A}^T)\\ =& \left((\mathbf{A}\otimes\mathbf{I}_{n_1}) + \mathbf{K}_{(n_1^2,n_1^2)}(\mathbf{A}\otimes\mathbf{I}_{n_1})\right)\partial\,\mathrm{vec}(\mathbf{A})\\ =& (\mathbf{I}_{n_1^2} + \mathbf{K}_{(n_1^2,n_1^2)})(\mathbf{A}\otimes\mathbf{I}_{n_1})\partial\,\mathrm{vec}(\mathbf{A})\end{aligned}$$

From above equation, the derivative part in Eq. (9) is

$$\frac{\partial \mathbf{A}\mathbf{A}^T}{\partial \mathbf{A}} = (\mathbf{I}_{n_1^2} + \mathbf{K}_{(n_1^2,n_1^2)})(\mathbf{A}\otimes\mathbf{I}_{n_1})$$

where $\mathbf{I}_{n_1^2}$ is the identity matrix with size $n_1^2 \times n_1^2$ and $\mathbf{K}_{(n_1^2,n_1^2)}$ is the commutation matrix. Although both matrix $\mathbf{I}_{n_1^2}$ and $\mathbf{K}_{(n_1^2,n_1^2)}$ are large, they are very sparse with a large number of zeros. In practice, they can be encoded in a sparse matrix form, which can greatly save the computational space.

Define the column vector $\mathbf{d}$ as

$$\mathbf{d} = (\frac{\partial \mathbf{A}\mathbf{A}^T}{\partial \mathbf{A}})^T \mathrm{vec}(\mathrm{sgn}(\mathbf{A}\mathbf{A}^T)) \tag{10}$$

Combining Eq. (8) and (10), the Euclidean gradient of the objective function $f(\mathbf{A})$ is

$$\nabla f(\mathbf{A}) = \nabla \mathcal{J}_1(\mathbf{A}) + \rho \cdot \mathrm{ivec}(\mathbf{d}) \tag{11}$$

where $\mathrm{ivec}(\cdot)$ is the inverse vectorize operator, i.e., $\mathrm{ivec}(\mathrm{vec}(\mathbf{X})) = \mathbf{X}$. With the Euclidean gradient $\nabla f(\mathbf{A})$, the Riemannian gradient with respect to $\mathbf{A} \in Gr(r_1, n_1)$ can be computed as

$$\mathrm{grad}\, f(\mathbf{A}) = (\mathbf{I} - \mathbf{A}\mathbf{A}^T)\cdot\nabla f(\mathbf{A}) \tag{12}$$

With the Riemannian gradient, we can use the popular nonlinear ManOpt solver to solve problem (7) effectively [6].

For updating factor matrices $\mathbf{B}$ and $\mathbf{C}$, their objective function $f(\mathbf{B})$ and $f(\mathbf{C})$ share the similar optimization structure as $f(\mathbf{A})$. Therefore, they can be solved in the same way with corresponding Riemannian gradients. The details are omitted here.

Updating spectral embedding $\mathbf{A}^{(i)}$,$\mathbf{B}^{(j)}$ and $\mathbf{C}^{(k)}$: For matrix $\mathbf{A}^{(i)}$, the objective $f(\mathbf{A}^{(i)})$ is

$$\min_{\mathbf{A}^{(i)}\in St(r_1,n_1)} f(\mathbf{A}^{(i)}) = \langle \mathbf{A}^{(i)}{\mathbf{A}^{(i)}}^T, \mathbf{L}_A^{(i)}\rangle + \|\mathbf{A}^{(i)}{\mathbf{A}^{(i)}}^T - \mathbf{A}\mathbf{A}^T\|_F^2 \tag{13}$$

The objective function $f(\mathbf{A}^{(i)})$ is also invariant to $\mathbf{Q} \in \mathcal{S}(r_1)$, i.e., $f(\mathbf{A}^{(i)}) = f(\mathbf{A}^{(i)}\mathbf{Q})$. Therefore, problem (13) can be also regarded as an unconstrained Grassmann manifold optimization problem. The Euclidean gradient of the objective function $f(\mathbf{A}^{(i)})$ is

$$\nabla f(\mathbf{A}^{(i)}) = 2\mathbf{L}_A^{(i)}\mathbf{A}^{(i)} + 4\mathbf{A}^{(i)}{\mathbf{A}^{(i)}}^T\mathbf{A}^{(i)} - 4\mathbf{A}\mathbf{A}^T\mathbf{A}^{(i)} \tag{14}$$

The Riemannian gradient grad $f(\mathbf{A}^{(i)})$ can be obtained in the same way as in Eq. (12). The Riemannian trust-regions algorithm is then used to compute the optimal solution since the objective function $f(\mathbf{A}^{(i)})$ in Eq. (13) is smooth. The initial $n_1 \times r_1$ spectral embedding $\mathbf{A}^{(i)}$ can be approximated by the r_1 eigenvectors of $\mathbf{L}_A^{(i)}$ corresponding to the first r_1 smallest eigenvalues. The objective function $f(\mathbf{B}^{(j)})$ and $f(\mathbf{C}^{(k)})$ can be solved in a similar fashion.

Updating tensor $\mathcal{X}$: The optimization problem with respect to $\mathcal{X}$ is formulated as follows:

$$\min_{\mathcal{X}} \|\mathcal{X} - \mathcal{G}\times_1\mathbf{A}\times_2\mathbf{B}\times_3\mathbf{C}\|_F^2 \quad s.t. \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T})$$

The optimal solution is given by:

$$\mathcal{X} = \mathcal{P}_\Omega(\mathcal{T}) + \mathcal{P}_{\Omega^c}(\mathcal{G}\times_1\mathbf{A}\times_2\mathbf{B}\times_3\mathbf{C}) \tag{15}$$

where Ω^c is the complement of Ω, i.e., the set of indexes of the unobserved elements.

According to above analysis, we summarize the algorithm for solving the tensor completion problem (4) in Algorithm 1.

Remark: Comparison with Euclidean solvers that usually deal with the orthogonality constraints by solving eigenvalue decomposition problem [23], our solver seeks to find concise subspace on the Grassmann manifold for all factor matrices. As pointed out in spectral clustering [31], when the leading eigenvalues are almost equal, the best spectral embedding is better determined by the subspace rather than a particular eigenvector as most of Euclidean solvers do. The same analysis can be analogously applied to solve Tucker decomposition in Euclidean space, i.e., using eigenvalue decomposition. The Riemannian solver is thus more interpretable in a straightforward way.

5 EXPERIMENTS

5.1 Datasets

We collect data from a variety of public sources. We first download the drug-disease associations from the Comparative Toxicogenomics Database [1] (CTD). The original dataset contains 1,048,547 pairs of drug-disease associations. Here, we only focus on those drugs with DrugBank [2] identifier and diseases with OMIM [3] identifier for conveniently integrating with auxiliary information on other public datasets. As discussed before, the drug's targets, usually proteins that drugs can bind, through which drugs interact with biological pathways and possibly change their behaviors and functions, are crucial in drug discovery. The drug's targets can be collected from the DrugBank database to obtain drug-target interactions. To get dense data, we only include those drugs that interact with at least two targets. We then merge the drug-disease and drug-target interactions into data schema $< drug, target, disease >$. The

[1] http://ctdbase.org/downloads/
[2] https://www.drugbank.ca/
[3] https://www.omim.org/

Algorithm 1: DTD

Input: $\mathcal{T}$, Ω, $\{\mathbf{S}_A^{(i)}\}_{i=1}^{n_a}$, $\{\mathbf{S}_B^{(j)}\}_{j=1}^{n_b}$, $\{\mathbf{S}_C^{(k)}\}_{k=1}^{n_c}$ and tol parameters α, β, γ ρ and rank (r_1, r_2, r_3).

Compute the all Laplacian matrix $\mathbf{L}_A^{(i)}$, $\mathbf{L}_B^{(j)}$ and $\mathbf{L}_C^{(k)}$
Initialize $\mathcal{X}$, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathcal{G}$ randomly, set $\mathcal{X}_\Omega = \mathcal{T}_\Omega$
Initialize $\mathbf{A}_0^{(i)}$ with r_1 eigenvectors of $\mathbf{L}_A^{(i)}$ corresponding to the first r_1 smallest eigenvalues. Same process to initialize $\mathbf{B}_0^{(j)}$ and $\mathbf{C}_0^{(k)}$
repeat
  Update $\mathcal{G}_{t+1}$ by Eq. (5)
  Compute Riemannian gradient grad $f(\mathbf{A}_t)$ by Eq. (12)
  Update $\mathbf{A}_{t+1}$ by using Conjugate-gradient solver
  Update $\mathbf{B}_{t+1}$ and $\mathbf{C}_{t+1}$ the same way as updating $\mathbf{A}_{t+1}$
  for $i \leftarrow 1$ *to* n_a **do**
    Update $\mathbf{A}_{t+1}^{(i)}$ by using trust-regions solver
  end
  for $j \leftarrow 1$ *to* n_b **do**
    Update $\mathbf{B}_{t+1}^{(j)}$ by using trust-regions solver
  end
  for $k \leftarrow 1$ *to* n_c **do**
    Update $\mathbf{C}_{t+1}^{(k)}$ by using trust-regions solver
  end
  Update $\mathcal{X}_{t+1}$ by Eq. (15)
until *Objective:* $\|f_{t+1} - f_t\|_F \leq tol$
return $\mathcal{X}$

Table 2: Dataset statistics

#drug	#target	#disease	#interaction	sparsity rate
450	708	1,267	188,479	0.047%

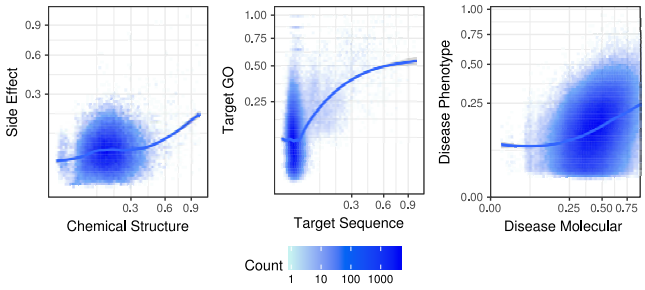


Figure 2: The pairwise scatter plots among different views of drugs, targets and diseases.

dataset contains 188,479 drug-target-disease interactions, involving 450 drugs, 708 targets and 1,267 diseases. Data statistics are shown in Table 2.

The data are then encoded into a $drug \times target \times disease$ tensor $\mathcal{X}$. An entry $\mathcal{X}_{ijk} = 1$ in the tensor indicates that drug i binds to target j and it can treat disease k; $\mathcal{X}_{ijk} = 0$ otherwise. The tensor is very sparse (with sparsity rate 0.047%). We further collect auxiliary information to align each mode of tensor with existing medical knowledge.

Auxiliary information: Following several previous studies [7, 9, 20, 30, 45], we collect auxiliary information from different datasets with respect to drugs, targets, and diseases. For drugs, we define two drug-drug similarities based on drugs' chemical structures and drugs' side effects [7]. For targets, two target-target similarities are considered based on target amino acid sequences and Gene Ontology (GO) terms, both of which can directly be obtain from Uniprot [4] database [20, 45]. For diseases, two disease-disease similarities are computed according to disease molecular profiles from biomedical literatures [8] and disease phenotypes on OMIM datasets [9].

Each view of data contains its own features and emphasizes different aspects. Different views may share some consistency and complementary properties. We begin our investigation by examining the strength of the associations among those different views of drugs, targets, and diseases. The scatter plots of the pairwise similarity scores are shown in Figure 2. We further assess the strength of their associations by calculating the correlation coefficient between two views of similarity scores, respectively. All correlations are positive with the correlation coefficient values of 0.148 (drug chemical structures vs. drug side effects), 0.372 (target sequences vs. target GO terms), and 0.374 (disease molecular profiles vs. disease phenotypes). Despite small values in some of the coefficients, all of them are significant because of the large sample sizes (e.g., 802, 011 data instances in the disease scatter plot). Therefore, it's reasonable to integrate the multi-view auxiliary information in the DTD framework and to improve the overall performance for modeling drug-target-disease interactions, as shown later.

5.2 Comparison Methods

As mentioned earlier, many existing methods can only handle the binary relationships of drug-target or drug-disease. We mainly compare with existing tensor completion approaches due to their abilities to represent the triple relationships of drug-target-disease. To show the effectiveness of the proposed DTD model, we compare with several existing models as described below.

- **Tucker decomposition** (TD): It only decomposes the tensor without any auxiliary information for each mode.
- **CMTF** [2]: CMTF constructs common latent factors shared by a tensor and single-view of auxiliary information by using coupled matrix-tensor CP-decomposition.
- **ConCMTF** [5]: A novel constrained tensor model with non-negativity, sparsity and orthogonality constraints. Similar to CMTF, it can incorporate single-view of auxiliary information but with Tucker tensor decomposition as base.
- **FaLRTC** [27]: FaLRTC can estimate the low rank structure by imposing the trace norm on its unfolding matrices and the algorithm is based upon alternating direction method of multipliers (ADMM).
- **TFAI** [29]: it recovers the tensor by incorporating within-mode auxiliary information and adopt alternating least squares algorithm to solve its problem.
- **Rubik** [41]: A novel knowledge-guided tensor factorization and completion framework to fit electronic health record data with non-negativity and sparsity constraints. It is also solved by ADMM algorithm.

[4] https://www.uniprot.org/

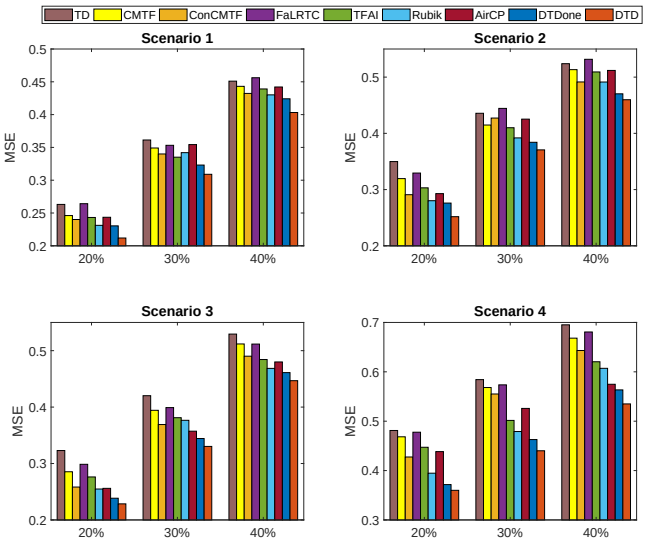


Figure 3: Comparison of recovery results over four scenarios with 20%, 30% and 40% test dataset.

- **AirCP** [18]: Another tensor model that integrates single-view auxiliary information using Laplacian regularization and it is optimized by ADMM algorithm.
- **DTDone**: A degraded version of our DTD model with single-view auxiliary information for each mode of tensor.

Note that the approaches TD and FaLRTC cannot integrate any auxiliary information when completing the tensor. The rest of comparison methods can only incorporate single-view auxiliary information for each mode. One has to either only include single-view for each mode, or a straightforward concatenation (e.g., average) of all similarity matrices into one. We tries both ways but the performance do not show much differences. The reason is that concatenation of all views may not be physically meaningful because each view has its owns specific statistical property. For these single-view models, we thus chose the drug chemical structure, target sequence, and disease phenotypes, all of which have long been considered valuable knowledge in drug discovery [20, 30, 45].

For all Tucker-based tensor models, we set the rank of core tensor $\mathcal{G}$, $r_i = 0.05n_i$ $(1 \leq i \leq 3)$; For CP-based tensor models, we set CP-rank $r = 0.05 \min(n_1, n_2, n_3)$. The regularization parameters of all comparison methods are tuned using the grid-based search algorithm for optimal performance. All methods use the same stopping threshold for a fair comparison, *i.e.*, the variation of two consecutive objective value is less than 10^{-4}. For the proposed DTD and DTDone model, we manually set the parameters $\alpha = \beta = \gamma = \rho = 0.01$. The impact of the regularization parameters on the performance of DTD will be discussed later.

5.3 Experimental Performance

To investigate the performance of different methods, we randomly selected a subset $\Omega' \in \Omega$ to be used as the unobserved elements (that is the test dataset), and evaluated the *mean squared errors* (MSE) between true values $\mathcal{T}_{\Omega'}$ and predicted values $\mathcal{X}_{\Omega'}$ [29]. We mainly consider the following random selection scenarios [18]: (i) Scenario 1 (random selection): we randomly select a fraction of elements across drug-target-disease tensor and assume that these are unobservable. (ii) Scenario 2 (random selection in drug mode): we randomly select a fraction of drugs, and assume that those drugs are completely unobservable across the whole tensor. (iii) Scenario 3 (random selection in target mode): similarly, we randomly select a fraction of targets. (iv) Scenario 4 (random selection in disease mode): A fraction of diseases are randomly selected. For each method, the experiment is repeated ten times independently and the average result is reported.

Figure 3 shows the performance of all the methods with different sampling ratios 20%, 30% and 40% (i.e., the fractions of test dataset). DTD consistently performs better than all the comparison methods within a wide range of sampling ratios. In addition, we have the following observations. First, the methods that integrate single-view or multi-view auxiliary information achieve better performance than TD and FaLRTC, indicating the importance of those auxiliary information. With the guidance of extra knowledge, these tensor models keep their performance even with a very sparse input tensor. Second, ConCMTF (Tucker decomposition) perform slightly better than CMTF (CP decomposition) in many situations. The main difference between them is that ConCMTF seeks to find as many non-overlapping structures as possible. Such non-overlapping latent structures are more concise and become specially favorable when jointly decomposing the tensor and matrices. The proposed DTD and DTDone also preserve such non-overlapping structures by imposing orthogonality constraint on factor matrices, which leads to better representations of data. Third, the proposed DTDone further achieves better performance than ConCMTF with an average improvement of 5.76%. As discussed before, the solutions of DTDone, both factor matrices and spectral embeddings, are exactly on the Grassmann manifold and can be well coupled together with each other to obtain better performance. And such advantages make DTDone generally outperform other single-view tensor completion methods such as TEAI, Rubik and AirCP. Finally, the proposed DTD method integrating all available auxiliary information achieves better performance than those only integrating single-view auxiliary information, which illustrates that DTD successfully makes use of all useful information sources to perform effective recovery for the drug-target-disease tensor. In summary, DTD can effectively predict potential interactions of drug, target and disease by leveraging multiple auxiliary data sources and has great potential to accelerate the drug development.

Top-k Prediction: In clinics, given a disease, it is critical to know which drugs can treat this disease as well as the targets involved in the disease pathway. These pairs of <drug, target> related to a special disease are very important in personalized treatments. We further evaluate the performance of top-k prediction for Scenario 4. Recall that we randomly select a subset of diseases and remove all of their interactions in the Scenario 4. For each disease, we can predict a top-k list of <drug, target> candidates.

We adopt Precision@k and Recall@k as our evaluation metrics for different methods. Both of them have been widely used to evaluate the quality of top-k predictions [18, 36]. In our experiments,

Table 3: Precision@k and Recall@k for different methods.

Methods	Prec@5	Rec@5	Prec@10	Rec@10
CMTF	0.287	0.347	0.264	0.283
ConCMTF	0.340	0.371	0.270	0.305
TEAI	0.334	0.359	0.254	0.284
Rubik	0.291	0.362	0.269	0.296
AirCP	0.289	0.352	0.261	0.298
DTDone	0.342	0.378	0.273	0.316
DTD	**0.353**	**0.394**	**0.292**	**0.331**

Table 4: Top 10 novel triple relationship of (drug→target→disease) by DTD model.

Carbamazepine→ Nuclear receptor subfamily 1 group I member 2 → Osteoporosis
Testosterone→ Estrogen receptor→Myocardial infarction
Nefazodone→D(2) dopamine receptor→Schizophrenia
Raloxifene→Estrogen receptor→Obesity
Fenofibrate→Matrix metalloproteinase→Psoriatic arthritis
Acetazolamide→Estrogen receptor→Amyotrophic lateral sclerosis
Raloxifene→Androgen receptor→Breast cancer
Amoxapine→Potassium voltage-gated channel subfamily H member 2 →Hepatocellular carcinoma
Nefazodone→Histamine H1 receptor→Schizophrenia
Promethazine→Muscarinic acetylcholine receptor M3→Obesity

we evaluate those tensor models with auxiliary information and test for k at 5 and 10 for both precision and recall.

Table 3 shows the top-k prediction performance of all the methods with 20% fractions of test dataset. Overall, the proposed DTD model performs the best among all methods in terms of both precision and recall with different values of k. The trend is very similar to the results based on the MSE metric. For example, DTD performs better than CMTF with an average improvement of 16.8% in precision and 15.3% in recall. We attribute the poor performance of the CMTF approach to the sparseness of tensor and its weak abilities of coupling tensor with auxiliary matrices (e.g., overlap structure). Moreover, DTD has a better performance than DTDone, indicating the superiority of a tensor model integrating rich multi-view auxiliary information. All these results illustrate that the proposed method can successfully predict top-k <drug, target> pairs by exploiting the compatible and complementary information from multi-view data sources.

Given these encouraging results, we use DTD model to predict novel triple relationships of drug-target-disease by leveraging all the observed data. The top-10 novel triple relationships are listed in Table 4. The results can then be evaluated by domain experts to see whether such interactions are clinically meaningful. Also, such top-k candidates can be further validated *in vitro* and *in vivo*. In summary, our proposed model efficiently predict potential interaction candidates with high accuracy, providing a systematic approach to narrow down the search space for further wet-lab investigations.

5.4 Parameter Studies

We further analyze the effect of key hyperparameters, the rank of core tensor (r_1, r_2, r_3) and the regularized parameters α, β, γ, and ρ.

5.4.1 The impact of tensor rank: In Tucker-based tensor decomposition, the hyperparameter (r_1, r_2, r_3) controls the rank of the core tensor $\mathcal{G}$, which also decides the number of latent features for each mode, i.e., **A**, **B** and **C**. In contrast to Tucker model, the CP decomposition only have one hyperparameter r, the CP-rank. In the experiments, we set the rank $r_i = \delta \cdot n_i$ $(1 \le i \le 3)$ for Tucker-based model and $r = \delta \cdot \min(n_1, n_2, n_3)$ for CP-based model. The δ is the ratio varying within $[0.05, 0.1, 0.15, 0.2, 0.25]$. We then evaluate all methods for scenario 4 with 20% as the test dataset. Figure 4 shows that our model benefits slightly from larger numbers of latent dimensions in terms of MSE and Prec@5 metric. But generally speaking, both DTD and DTDone are fairly robust with respect to δ.

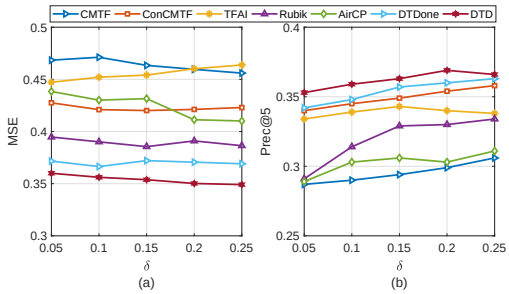


Figure 4: Effect of the tensor rank in scenario 4.

5.4.2 The impact of regularized parameters. Finally, we explore the impact of the regularization parameters on the quality of tensor completion. Recall that α, β and γ control the contributions of auxiliary information of drugs, targets, and disease, respectively. ρ controls the sparsity of factor matrices. In order to better understand the effect of these parameters, we vary their values in the range $[0.001, 0.01, 0.1, 1, 10]$, and then evaluate DTDone and DTD for scenario 4 with 20% as testing data. When studying one variable (e.g., α), we fix all the rest variables to be 0.01. As shown in Figure 5, DTDone and DTD are stable over a wide range of α, β and γ in terms of MSE. Specifically, a relatively low MSE can be achieved when those regularized parameters are around 0.01.

6 RELATED WORK

We review related research on computational predictions of drugs, targets, and diseases, as well as on tensor completion for high-dimensional structured data.

Modeling drug-target-disease. Treating human diseases caused by complex biological processes involves activities of many biological entities such as drugs and targets. Methods in computational pharmacology aim to find associations among those entities, and understand drug's MoAs [21]. Two excellent surveys [25], [16] provide a very detailed overview of different computational methods

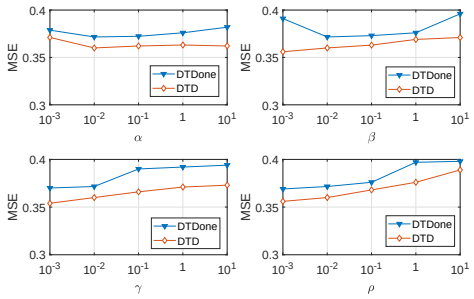


Figure 5: Effect of regularized parameters:α, β, γ and ρ .

for predicting drug-disease and drug-target relationships, respectively. Among existing approaches, one very popular method is network-based inference models, e.g., a bipartite network consisting of one layer for drugs and one layer for diseases (targets). Different machine learning algorithms, such as random-walk [12], matrix factorization [17], and support vector machine [42], have been applied to predict novel interactions of drug-disease (or target). In addition to network topology, existing medical knowledge, such as a variety of omics data, on the Web can be incorporated to better understand complex human metabolic systems [9, 30, 45, 46]. For example, a multiple kernel learning method has been developed to predict drug-target relationships by integrating multiple similarities of drugs and targets [30]. Decagon models drug polypharmacy side effects via graph convolutional networks with additional drug-drug and protein-protein interactions [46].

However, current studies generally considered drug-disease and drug-target predictions as two independent tasks and the relationships of drug-target-disease is typically ignored. There exist several studies incorporating target information in the task of drug-disease predictions [13, 40, 46]. However, their extensions are unsatisfactory because their goals are still on modeling binary relationships, not on the triple drug-target-disease patterns we aim to learn.

Tensor completion. Tensors are very powerful in real-world applications because of their ability to represent multi-aspects or high-dimensional data [23]. In many applications, we are often interested in analyzing tensors together with matrices from additional information, such as computational phenotyping [41], recommender system [18], and time-evolving topic discovery [5]. This coupled tensor-matrix factorization method has been developed over the years. For example, Acar et. al. proposed a gradient-based algorithm for coupled matrix-tensor factorization [2]. Narita et. al. incorporated valuable auxiliary information to further improve the quality of tensor recovery [29]. Wang et. al. proposed a knowledge guided tensor factorization model for health data analysis [41]. Ge et. al. proposed a spatiotemporal dynamics recovery framework, which could capture the latent relationships among locations, memes, and times by coupled factorization [18]. However, none of these methods are guaranteed to non-overlapping structures in both tensor and matrices and can only integrate single-view of additional information for *each mode* of tensor. Some models [5, 23, 29] try to impose orthogonality constraint on factor matrices. However, their solutions are usually not unique, which is hard to couple with auxiliary information. Recently, concrete tensor/matrix representations on the manifold is a fast growing research topic [1, 22, 43]. Kasai et. al. recently developed a novel Riemannian manifold preconditioning approach for tensor completion [43], but they did not take the auxiliary information into account. The proposed approach is innovative and properly addresses the challenges in coupled tensor-matrix factorization with multi-view auxiliary information.

7 CONCLUSION

Modeling drug-target-disease interactions is important for understanding drugs' MoAs in drug development. The problem has conventionally been addressed separately by considering associations of drug-target or drug-disease, and most existing methods cannot leverage intrinsic interactions among these three biological entities. Here we present a novel approach DTD, which explicitly explores a three-way drug-target-disease tensor via coupled tensor-matrix factorization. Completing such tensor is challenging because of its large sparsity. DTD elegantly integrates multiple Web data to align existing knowledge with the tensor. With the guidance of auxiliary information, DTD infers the concurrence of drug-target-disease more accurately than do baselines. Another distinguishing aspect of DTD is that it directly optimizes on the Grassmann manifold, which is more effective than Euclidean solvers. Experimental results on a real-world dataset demonstrate the effectiveness and efficiency of the proposed method.

The proposed DTD can easily incorporate additional domain knowledge and can be extended to detect high-order tensor (e.g., four-way *drug-target-gene-disease* tensor) with relatively little effort. Future work should aim to extend this model to simulate human metabolic systems by considering more biological entities (e.g., gene) as well as other domain knowledge (e.g., gene expression data). Deep understanding of those entities opens up opportunities to use rich Web data to assist follow-up analysis via formal pharmacological studies.

Acknowledgements

This work has been supported in part by NIH grants R03HG008632, NSF CCF1815139, and a Faculty Investment Fund from CWRU.

REFERENCES

[1] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. 2009. *Optimization algorithms on matrix manifolds.* Princeton University Press.
[2] Evrim Acar, Tamara G Kolda, and Daniel M Dunlavy. 2011. All-at-once optimization for coupled matrix and tensor factorizations. *arXiv preprint arXiv:1105.3422* (2011).
[3] Benjamin M Althouse, Samuel V Scarpino, Lauren Ancel Meyers, John W Ayers, Marisa Bargsten, Joan Baumbach, John S Brownstein, Lauren Castro, Hannah Clapham, Derek AT Cummings, et al. 2015. Enhancing disease surveillance with novel data streams: challenges and opportunities. *EPJ Data Science* 4, 1 (2015), 17.
[4] Matheus Araujo, Yelena Mejova, Ingmar Weber, and Fabricio Benevenuto. 2017. Using facebook ads audiences for global lifestyle disease surveillance: Promises and limitations. In *WebSci.*
[5] Sanaz Bahargam and E Papalexakis. 2018. Constrained Coupled Matrix-Tensor Factorization and its Application in Pattern and Topic Detection. In *ASONAM.*
[6] Nicolas Boumal, Bamdev Mishra, P-A Absil, and Rodolphe Sepulchre. 2014. Manopt, a Matlab toolbox for optimization on manifolds. *The Journal of Machine Learning Research* 15, 1 (2014), 1455–1459.
[7] Monica Campillos, Michael Kuhn, Anne-Claude Gavin, Lars Juhl Jensen, and Peer Bork. 2008. Drug target identification using side-effect similarity. *Science* 321, 5886 (2008), 263–266.

[8] Horacio Caniza, Alfonso E Romero, and Alberto Paccanaro. 2015. A network medicine approach to quantify distance between hereditary disease modules on the interactome. *Scientific reports* 5 (2015), 17658.

[9] Huiyuan Chen and Jing Li. 2017. A flexible and robust multi-source learning algorithm for drug repositioning. In *BCB*. ACM.

[10] Huiyuan Chen and Jing Li. 2017. Learning Multiple Similarities of Users and Items in Recommender Systems. In *ICDM*.

[11] Huiyuan Chen and Jing Li. 2018. DrugCom: Synergistic Discovery of Drug Combinations Using Tensor Decomposition. In *ICDM*.

[12] Xing Chen, Ming-Xi Liu, and Gui-Ying Yan. 2012. Drug–target interaction prediction by random walk on the heterogeneous network. *Molecular BioSystems* 8, 7 (2012), 1970–1978.

[13] Feixiong Cheng, Chuang Liu, Jing Jiang, Weiqiang Lu, Weihua Li, Guixia Liu, Weixing Zhou, Jin Huang, and Yun Tang. 2012. Prediction of drug-target interactions and drug repositioning via network-based inference. *PLoS computational biology* 8, 5 (2012), e1002503.

[14] Alan Edelman, Tomás A Arias, and Steven T Smith. 1998. The geometry of algorithms with orthogonality constraints. *SIAM journal on Matrix Analysis and Applications* 20, 2 (1998), 303–353.

[15] Tewodros Eguale, David L Buckeridge, Aman Verma, Nancy E Winslade, Andrea Benedetti, James A Hanley, and Robyn Tamblyn. 2016. Association of off-label drug use and adverse drug events in an adult population. *JAMA internal medicine* 176, 1 (2016), 55–63.

[16] Ali Ezzat, Min Wu, Xiao-Li Li, and Chee-Keong Kwoh. 2018. Computational prediction of drug–target interactions using chemogenomic approaches: an empirical survey. *Briefings in bioinformatics* (2018).

[17] Ali Ezzat, Peilin Zhao, Min Wu, Xiao-Li Li, and Chee-Keong Kwoh. 2017. Drug-target interaction prediction with graph regularized matrix factorization. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* 14, 3 (2017), 646–656.

[18] Hancheng Ge, James Caverlee, Nan Zhang, and Anna Squicciarini. 2016. Uncovering the spatio-temporal dynamics of memes in the presence of incomplete information. In *CIKM*.

[19] Graciela Gonzalez, Juan C Uribe, Luis Tari, Colleen Brophy, and Chitta Baral. 2007. Mining gene-disease relationships from biomedical literature: weighting protein–protein interactions and connectivity measures. In *Biocomputing 2007*. World Scientific, 28–39.

[20] Assaf Gottlieb, Gideon Y Stein, Eytan Ruppin, and Roded Sharan. 2011. PREDICT: a method for inferring novel drug indications with application to personalized medicine. *Molecular systems biology* 7, 1 (2011), 496.

[21] Alexander S Hauser, Misty M Attwood, Mathias Rask-Andersen, Helgi B Schiöth, and David E Gloriam. 2017. Trends in GPCR drug discovery: new agents, targets and indications. *Nature Reviews Drug Discovery* 16, 12 (2017), 829.

[22] Hiroyuki Kasai and Bamdev Mishra. 2016. Low-rank tensor completion: a Riemannian manifold preconditioning approach. In *ICML*.

[23] Tamara G Kolda and Brett W Bader. 2009. Tensor decompositions and applications. *SIAM review* 51, 3 (2009), 455–500.

[24] Stephanie N Lewis, Elaine Nsoesie, Charles Weeks, Dan Qiao, and Liqing Zhang. 2011. Prediction of disease and phenotype associations from genome-wide association studies. *PLoS One* 6, 11 (2011), e27175.

[25] Jiao Li, Si Zheng, Bin Chen, Atul J Butte, S Joshua Swamidass, and Zhiyong Lu. 2015. A survey of current trends in computational drug repositioning. *Briefings in bioinformatics* 17, 1 (2015), 2–12.

[26] Craig W Lindsley. 2014. New statistics on the cost of new drug development and the trouble with CNS drugs.

[27] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. 2013. Tensor completion for estimating missing values in visual data. *IEEE transactions on pattern analysis and machine intelligence* 35, 1 (2013), 208–220.

[28] Canyi Lu, Shuicheng Yan, and Zhouchen Lin. 2016. Convex sparse spectral clustering: Single-view to multi-view. *IEEE Transactions on Image Processing* 25, 6 (2016), 2833–2843.

[29] Atsuhiro Narita, Kohei Hayashi, Ryota Tomioka, and Hisashi Kashima. 2012. Tensor factorization using auxiliary information. *Data Mining and Knowledge Discovery* 25 (2012).

[30] André CA Nascimento, Ricardo BC Prudêncio, and Ivan G Costa. 2016. A multiple kernel learning algorithm for drug-target interaction prediction. *BMC bioinformatics* 17, 1 (2016), 46.

[31] Andrew Y Ng, Michael I Jordan, and Yair Weiss. 2002. On spectral clustering: Analysis and an algorithm. In *NIPS*.

[32] Michael J Paul and Mark Dredze. 2013. Drug extraction from the web: Summarizing drug experiences with multi-dimensional topic models. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 168–178.

[33] Nicola Perra, Duygu Balcan, Bruno Gonçalves, and Alessandro Vespignani. 2011. Towards a characterization of behavior-disease models. *PloS one* 6, 8 (2011), e23084.

[34] Predrag Radivojac, Wyatt T Clark, Tal Ronnen Oron, Alexandra M Schnoes, Tobias Wittkop, Artem Sokolov, Kiley Graim, Christopher Funk, Karin Verspoor, Asa Ben-Hur, et al. 2013. A large-scale evaluation of computational protein function prediction. *Nature methods* 10, 3 (2013), 221.

[35] Bisakha Ray, Elodie Ghedin, and Rumi Chunara. 2016. Network inference from multimodal data: a review of approaches from infectious disease transmission. *Journal of biomedical informatics* 64 (2016), 44–54.

[36] Steffen Rendle, Leandro Balby Marinho, Alexandros Nanopoulos, and Lars Schmidt-Thieme. 2009. Learning optimal ranking with tensor factorization for tag recommendation. In *KDD*.

[37] Rita Santos, Oleg Ursu, Anna Gaulton, A Patrícia Bento, Ramesh S Donadi, Cristian G Bologa, Anneli Karlsson, Bissan Al-Lazikani, Anne Hersey, Tudor I Oprea, et al. 2017. A comprehensive map of molecular drug targets. *Nature reviews Drug discovery* 16, 1 (2017), 19.

[38] Abeed Sarker and Graciela Gonzalez. 2015. Portable automatic text classification for adverse drug reaction detection via multi-corpus training. *Journal of biomedical informatics* 53 (2015), 196–207.

[39] Qiong Wang, Junbin Gao, and Hong Li. 2017. Grassmannian manifold optimization assisted sparse spectral clustering. In *CVPR*.

[40] Wenhui Wang, Sen Yang, Xiang Zhang, and Jing Li. 2014. Drug repositioning by integrating target information through a heterogeneous network model. *Bioinformatics* 30, 20 (2014), 2923–2930.

[41] Yichen Wang, Robert Chen, Joydeep Ghosh, Joshua C Denny, Abel Kho, You Chen, Bradley A Malin, and Jimeng Sun. 2015. Rubik: Knowledge guided tensor factorization and completion for health data analytics. In *KDD*.

[42] Yoshihiro Yamanishi, Michihiro Araki, Alex Gutteridge, Wataru Honda, and Minoru Kanehisa. 2008. Prediction of drug–target interaction networks from the integration of chemical and genomic spaces. *Bioinformatics* 24, 13 (2008), i232–i240.

[43] Hongyi Zhang, Sashank J Reddi, and Suvrit Sra. 2016. Riemannian SVRG: Fast stochastic optimization on Riemannian manifolds. In *NIPS*.

[44] Qian Zhang, Nicola Perra, Daniela Perrotta, Michele Tizzoni, Daniela Paolotti, and Alessandro Vespignani. 2017. Forecasting seasonal influenza fusing digital indicators and a mechanistic disease model. In *WWW*.

[45] Xiaodong Zheng, Hao Ding, Hiroshi Mamitsuka, and Shanfeng Zhu. 2013. Collaborative matrix factorization with multiple similarities for predicting drug-target interactions. In *KDD*.

[46] Marinka Zitnik, Monica Agrawal, and Jure Leskovec. 2018. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics* (2018).

[47] Bin Zou, Vasileios Lampos, and Ingemar Cox. 2018. Multi-Task Learning Improves Disease Models from Web Search. In *WWW*.

SamWalker: Social Recommendation with Informative Sampling Strategy

Jiawei Chen[1,2], Can Wang[1,2,†], Sheng Zhou[1], Qihao Shi[1,2], Yan Feng[1,2], Chun Chen[1]
[1]College of Computer Science, Zhejiang University, China
[2]Zhejiang University-LianlianPay Joint Research Center
{sleepyhunt,wcan,zhousheng_zju,shiqihao321,fengyan,chenc}@zju.edu.cn

ABSTRACT

Recommendation from implicit feedback is a highly challenging task due to the lack of reliable negative feedback data. Only positive feedback are observed and the unobserved feedback can be attributed to two reasons: *unknow* or *dislike*. Existing methods address this challenge by treating all the un-observed data as negative (dislike) but downweight the confidence of these data. However, this treatment causes two problems: (1) Confidence weights of the unobserved data are usually assigned manually, which lack flexible and may create empirical bias in evaluating user's preference. (2) To handle massive volume of the unobserved feedback data, most of the existing methods rely on stochastic inference and data sampling strategies. However, since users are only aware of a very small fraction of items in a large dataset, it is difficult for existing samplers to select *informative* training instances in which the user really dislikes the item rather than does not know it.

To address the above two problems, we propose a new recommendation method SamWalker that leverages social information to infer data confidence and guide the sampling process. By modeling data confidence with a social context-aware function, SamWalker can adaptively specify different weights to different data based on users' social contexts. Further, a personalized random-walk-based sampling strategy is developed to adaptively draw informative training instances, which can speed up gradient estimation and reduce sampling variance. Extensive experiments on three real-world datasets demonstrate the superiority of the proposed SamWalker method and its sampling strategy.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Social recommendation, Implicit feedback, Sampling

ACM Reference Format:
Jiawei Chen[1,2], Can Wang[1,2,†], Sheng Zhou[1], Qihao Shi[1,2], Yan Feng[1,2], Chun Chen[1]. 2019. SamWalker: Social Recommendation with Informative Sampling Strategy. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313582

[†]Corresponding author: wcan@zju.edu.cn.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313582

1 INTRODUCTION

With the exponential growth of information on electronic commerce websites such as Amazon and Taobao, recommender systems are drawing more attention from both academia and industry [6, 15, 16, 25, 27, 39, 56, 61]. Collaborative filtering (CF), as the prevalent recommendation model in these systems, infers user's preference and produces recommendations based on user's historical behaviors. Early work on CF mainly focused on explicit feedback, where the numerical ratings which directly reflect users' preference are provided.

However, explicit feedback may not be available in many applications. We usually only have access to implicit feedback derived from user actions, e.g. users' video viewing and product purchasing history. Although implicit feedback is abundant in practice, recommendation from implicit feedback is more difficult than from explicit feedback. The reason is that learning from implicit feedback lacks reliable negative feedback data, i.e. only the positive feedback are observed in the implicit feedback data. The un-observed user-item feedback data (e.g. a user has not bought an item yet) are a mixture of real negative feedback (a user does not like it) and missing values (a user just does not know it). While the positive feedback suggests that the user likes the item, the un-observed feedback does not necessarily mean the user dislikes the item. In most cases, users may just not know the items that they have not consumed.

Existing methods address this problem by treating all the unobserved data as negative (dislike) but downweight the confidence of these data. Most of these methods rely on the careful assignment of confidence weights to the data. Although it may be effective, choosing these weights usually involves heuristic alterations to the data and thus need expensive exhaustive grid search via cross-validation. Furthermore, it is unrealistic for researchers to manually set flexible and diverse weights for millions of data. In practical scenarios, different data may have different confidence on estimating users' preference. Some unobserved feedback data may be attributed to users' preference, while others are the result of users' limited awareness. Coarse-grained manual confidence weights will create empirical bias on estimating user's preference.

Another problem in recommendation from implicit feedback is learning efficiency. Due to the lack of reliable negative feedback data, all the un-observed data need to be considered to learn users' preference [4]. However, this degrades the learning efficiency

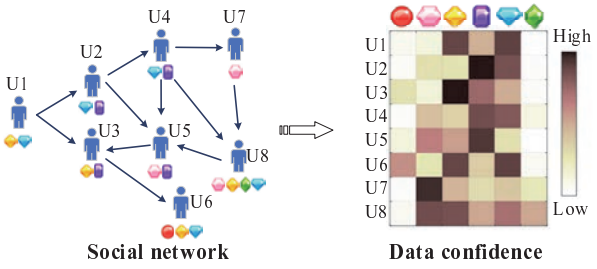


Figure 1: SamWalker estimates data confidence based on user's social relations. The left part of the figure illustrates a social network including implicit feedback expressed by users. The consumed items (i.e. the items with positive feedback) are shown below the users. The right part shows our inferred data confidence.

because of the large number of unobserved data. For efficient recommendation, most of the existing methods employ stochastic gradient descent solvers and corresponding data sampling strategies (e.g.uniform, item popularity-based). However, in real world, users typically are only aware of a relatively small fraction of the potential items [31]. In such cases, existing samplers usually select uninformative data with low confidence weight in which the user just does not know the item rather than dislikes it. This will affect the convergence and recommendation performance of the methods.

To deal with these problems, we propose a novel recommendation method SamWalker that leverages social information to simultaneously learn the personalized data confidence and draw informative training instances. With the development of online social websites, social relations have become a major information resource when users select items to consume [36]. Users usually get item information from social friends (neighbors) [10] and their *exposure* to items (i.e. whether a user knows the items) will inevitable be dominated by their *social contexts* (i.e. whether their direct or indirect social neighbors have consumed the items). Thus, users' social contexts can affect how users are exposed to the items and suggest the confidence of the data. It is consistent with our intuitions. Note that there exists two reasons for negative feedback: unknown or dislike. When an item is more popular among the user's social neighbors (e.g. the purple gem comparing with the green gem for user U1 in Figure 1), the user is more likely to know the item and his feedback is more attributed to his preference. Correspondingly, the data will be more reliable in deriving user's preference. To capture this insight, as illustrated in Figure 1, SamWalker simulates item information propagation along the social network and models individual confidence weights as a social context-aware function. By iteratively learning transformation function and user's preference based on EXMF framework (exposure-based matrix factorization [30]), SamWalker can adaptively specify different weights to different data based on user's social contexts.

Due to the large number of unobserved data, developing an efficient informative sampling strategy is challenging. It is apparently inefficient to estimate and rank the current learned confidence weights for every data to select informative data. Instead, we propose an efficient sampling strategy based on the random walk along the social network. Intuitively, if our more and closer social friends (neighbors) have consumed the item, we are more likely to know it due to the information sharing in the social network. Our feedback on this item can thus be attributed to our preference with more confidence and the corresponding user-item feedback data is more informative. Consequently, we conduct personalized random walk for each user to explore his local social contexts and pick out items consumed by these close direct or indirect social neighbors. theoretical analysis proves that the distribution of the proposed sampling strategy is proportional to the data confidence, which can reduce the sampling variance and speed up gradient estimation.

It is worthwhile to highlight the following contributions:

- We propose a new recommendation method SamWalker that adaptively learns the data confidence based on users' social context.
- We propose an efficient social-based sampling strategy to draw informative training instances, which can both reduce sampling variance and speed up gradient estimation.
- Our experimental evaluation on three well-known benchmark datasets demonstrates that SamWalker outperforms a range of state-of-the-art methods and analyzes the superiority of the proposed social-based sampling strategy.

The rest of this paper is organized as follows. We briefly review related works in section 2. We give the problem definition and background in section 3. The SamWalker model is introduced in section 4. The experimental results and discussions are presented in section 5. Finally, we conclude the paper and present some directions for future work in section 6.

2 RELATED WORK

In this section, we review the most related works from the following three perspectives.

Recommendation from implicit feedback data. Due to the lack of reliable negative feedback data, existing recommendation methods for implicit feedback data treat all the un-observed data as negative but downweight the confidence of these data. However, most of the existing methods manually assign coarse-grained confidence weights. For example, the classic weighted factorization matrix model (WMF) [22] and many neural-based collaborative methods (e.g. CDAE[49], NCF[17]) used a simple heuristic where all negative feedback data are equally downweighted vis-a-vis the positive feedback data. [18] and [54] also manually assign the confidence weights based on item popularity.

More recently, a new probabilistic model EXMF[30] was proposed to incorporate user's exposure to items into the CF methods. When inferring user's preference, EXMF can translate user's exposure as data confidence. However, this method suffers from the efficiency problem. We will analyze EXMF model in details in section 3.

Some other pair-wise methods treat negative feedback data in a different way. Bayesian personalized ranking (BPR) [38] focuses on the learning of relative preferences and aims at maximizing the AUC objective function. They believe that user's preference on his consumed items is above un-consumed items. Similarly, the weighted approximated ranking pairwise (WARP) loss proposed in [48] optimizes Precision@K.

Efficient Recommendation. For efficient recommendation, most of the existing methods rely on the stochastic optimization and corresponding data sampling strategy. The most popular sampling strategy is to draw un-observed feedback data uniformly, which has been widely adopted by WMF [22], BPR [38],CDAE [49], NCF [17], etc. Also, [54] and [19] further propose item popularity-based and item-user co-bias sampling strategy to reduce sampling variance. A detailed comparison of these sampling methods is presented in Table 4. However, as users are only exposed to very small fraction of the items, these samplers usually select uninformative data to estimate user's preference, which affects convergence and recommendation performance.

Some other sampling strategies are proposed to improve convergence and accuracy from different perspectives. [37, 55, 57] propose subtle dynamic sampling strategies to oversample the "difficult" negative examples in which the prediction is much different from the ground-truth. Although effective, sampling these "difficult" data for advanced preference model (e.g. BMF [19] or neural-based CF) still suffer from low efficiency [12]. Besides, the stochastic gradient estimator is biased and may amplify the natural noise in user-item feedback data [29]. [55] also proposes to draw positive data based on random walk along user-item bipartite graph. Our sampling strategy differs [55] in that we pay more attention to sample informative negative data. [9] proposes several sampling strategies to balance backward computation of the item-dependent neural network and the user-item interaction function. [12] proposes an effective sampling strategy by leveraging view data.

Also, [4, 18, 22] propose memoization strategies (e.g. ALS, eALS) to speed up inference. However, these strategies are just suitable for the model with K-separable property and L2 loss function (Guassian likelihood). In fact, cross-entropy loss function (Bernoulli likelihood) is more natural for the binary implicit feedback data and has been validated better performance than L2 [26].

Social recommendation. Social information has been utilized to improve recommendation performance in recent works. These methods mainly assume that connected users will share similar preference [2, 13, 14, 23, 41, 44, 52, 53, 60, 62]: Sorec [33], TrustMF [51], PSLF [40], jointly factorize rating matrix and trust (social) matrix by sharing a common latent user space; In [3, 7, 32, 42, 46, 50], users' feedback is considered as synthetic results of their preference and social influence; [24, 46] utilize a social regularization term to constrain user's latent preference close to his trusted friends; [47, 58] extend pair-wise BPR framework by further assuming that for all items with negative feedback, a user would prefer the items consumed by their friends over the rest.

Also, there are two recent works believe that comparing with users' preference, users' exposure is more influenced by their social friends (neighbors). Thus, [45] and [26] integrate social influence on user's exposure into the generative process of EXMF model. [45] extends EXMF with social regularization (SERec-Re) and social boosting (SERec-Bo). [26] integrates social knowledge influence and social consumption influence into EXMF model. However, these two methods need to infer $n \times m$ parameters of user's exposure, which will suffer from overfitting and become the efficiency bottleneck for practise application. SamWalker employs social-based variational posterior of user's exposure, which requires much fewer parameters and better captures heterogenous influence from both direct and indirect social neighbors. Further, SamWalker uses a social-based sampler to speed up inference and reduce sampling variance.

3 PRELIMINARIES

In this section, we first give the problem definition of implicit recommendation. Then, we introduce exposure-based matrix factorization (EXMF) [30] framework from variational perspective to provide usual insight about the relation between user's exposure and data confidence.

3.1 Problem definition

Suppose we have a recommender system with user set U (including n users) and item set I (including m items). The implicit feedback data is represented as $n \times m$ matrix X with entries x_{ij} denoting whether or not the user i has consumed the item j. Social information indicates the connections between users, represented as $n \times n$ matrix T. Also, $\mathcal{T}_i$ denotes the set of the connected social friends (direct neighbors) of user i. The task of a recommender system can be stated as follow: recommending items for each user that are most likely to be consumed by him.

3.2 Exposure-based matrix factorization (EXMF)

EXMF [30] directly incorporates user's exposure into collaborative filtering. Firstly EXMF generates the latent variable a_{ij}, which indicates whether user i has been exposed to item j. Then, EXMF models user's consumption x_{ij} based on a_{ij},

$$a_{ij} \sim Bernoulli(\eta_{ij}) \tag{1}$$

$$(x_{ij}|a_{ij} = 1) \sim Bernoulli(\sigma(u_i^\top v_j)) \tag{2}$$

$$(x_{ij}|a_{ij} = 0) \sim \delta_0 \approx Bernoulli(\varepsilon) \tag{3}$$

where δ_0 denotes $p(x_{ij} = 0|a_{ij} = 0) = 1$; η_{ij} is the prior probability of exposure. Here we relax function δ_0 as $Bernoulli(\varepsilon)$ to make model more robust, where ε is a small constant (e.g. ε=0.001). When $a_{ij} = 0$, we have $x_{ij} \approx 0$, since the user does not know the item and he can not consume it. When $a_{ij} = 1$, when the user has learned the item, he will decide whether or not to consume the item based on his preference. Thus, x_{ij} is modeled as the classic preference model[1] and factorized by the latent vectors u_i and v_j, which respectively characterize the latent preferences of user i and latent attributes of item j. To facilitate the description, here we collect the parameters of the preference model as $\theta = \{u_i, v_j\}_{i \in U, j \in I}$.

3.3 Analyses of EXMF from variational perspective

The marginal likelihood of EXMF is composed of a sum over the marginal likelihood of individual datapoint $\log p(X) = \sum_{ij} \log p(x_{ij})$,

[1]Here the preference model is slightly different from the original model presented in work [30] in that we employ Bernoulli likelihood instead of Gaussian likelihood on x_{ij}. In fact, Bernoulli likelihood is more natural for the binary variable [26].

which can be rewritten as:

$$\begin{aligned}\log p(x_{ij}) &= E_q[\log p(x_{ij}, a_{ij}) - \log q(a_{ij}|x_{ij})] \\ &\quad + E_q[\log p(a_{ij}|x_{ij}) - \log q(a_{ij}|x_{ij})] \\ &= L(\theta, q; x_{ij}) + KL(q(a_{ij}|x_{ij})||p(a_{ij}|x_{ij}))\end{aligned} \tag{4}$$

where $q(a_{ij}|x_{ij})$ is defined as an approximated variational posterior of a_{ij}. The second term is the KL divergence of the approximate variational posterior from the true posterior. Since the KL-divergence is non-negative, the first term $L(\theta, q; x_{ij})$ is the evidence lower bound on (ELBO) the margin likelihood. Thus, optimizing marginal likelihood can be translated as optimizing the lower bound $L(\theta, q; x_{ij})$ w.r.t. both the variational posterior and the preference parameters θ. Classic variational methods [20] usually employ conjugate variational distribution and individual variational parameters[2], i.e. $q(a_{ij}|x_{ij}) = Bernoulli(\gamma_{ij})$. For convenience we collect variational parameters γ_{ij} for every user-item pairs (i, j) as matrix Y. Then, the ELBO can be transferred into:

$$\begin{aligned}L(\theta, Y; X) &= \sum_{ij} E_q[\log p(x_{ij}, a_{ij}) - \log q(a_{ij}|x_{ij})] \\ &= \sum_{ij} \gamma_{ij} \ell(x_{ij}, \sigma(u_i^\top v_j)) + \sum_{ij} g(\gamma_{ij})\end{aligned} \tag{5}$$

The EBLO is composed of the two terms, the first term is a weighted cross-Entropy loss for the predicted preference, where $\ell(a, b) = alog(b) + (1 - a)log(1 - b)$. The second term is a loss function w.r.t γ_{ij}:

$$g(\gamma_{ij}) = (1 - \gamma_{ij})\ell(x_{ij}, \varepsilon) + \ell(\gamma_{ij}, \eta_{ij}) - \ell(\gamma_{ij}, \gamma_{ij}) \tag{6}$$

Exposure probability as the data confidence. From equation (5), a good property is observed that the variational parameters γ_{ij}, which characterize the probability that user i is exposed to item j, act as the confidence of the corresponding data to infer the preference parameters θ ($\theta = \{u_i, v_j\}_{i \in U, j \in I}$). This is clear by considering the following fact: when γ_{ij} becomes larger (or smaller), the inferred user and item factors θ make more (or less) contributions on the objective function. This finding is consistent with our intuitions. Only if the user has been exposed to the item, he can decide whether or not to consume the items based on his preference. Thus, the data with larger exposure is more reliable in deriving user's preference.

Weaknesses. However, although EXMF can adaptively derive the confidence of the data, we emphasize two latent weaknesses of the EXMF model: (1) EXMF is costly in accounting for all unobserved data ($O(n \times m)$) and is thus unrealistic for practical use. Although some sampling strategies can be used to speed up the algorithm, the gradient estimator exhibits high variance. Typically, in real world large datasets, each user will only be exposed to or be aware of a relatively small fraction of the potential items that they could interact with. That is, the γ_{ij} of most data are small and these data make limited contribution on updating parameter θ. Existing coarse-grained sampling strategy will usually draw uninformative data with small γ_{ij}, which affects the convergence and the recommended performance of the methods. (2) EXMF assumes user-independent posteriors of user's exposure. On the one hand, the number of variational parameters γ_{ij} grows quickly with the number of users and items ($n \times m$). This will suffer from serious overfitting and become the efficiency bottleneck for practise recommender systems. On the other hand, social network enable effective information sharing [1, 10]. Users' exposure to items will be influenced by their social contexts. Independent assumption of users' exposure is not practical in real world.

Thus, we are interested in, and propose a solution to two related problems:

(1) A social-based variational distribution of user's exposure that can both model the social influence between users and employ fewer variational parameters to speed up inference and alleviate overfitting.
(2) A sampling strategy that can draw informative training instances to speed up gradient estimation and reduce sampling variance.

4 SAMWALKER

For the purpose of solving the above problems, as illustrated in Figure 2, we propose a new recommendation method SamWalker that replaces individual variational parameter with a social context-aware function: $\gamma_{ij} = g_\varphi(X, T)$, where φ are replaced variational parameters. That is, we design an transformation function g_φ that maps the local social context of the user, i.e. whether his direct or indirect social friends (neighbors) have consumed the item, into the probability of user's exposure to the item. It is reasonable since users usually get item information from social network and their exposure to items depend on their local social contexts. An idea of modeling transformation function g_φ is to iteratively simulate the information spread via the social network. Similar to PageRank algorithm [34], we initially set the label of user's exposure according to his consumption. Then, all users spread their item information to their connected friends via the social network. The spread process is repeated until a global stable state is achieved. Concretely, in each step, users collect information from the connected social friends (neighbors) and reconstruct their exposure as follows:

$$\gamma_{ij}^{(t+1)} = (1 - c)x_{ij} + \sum_{k \in \mathcal{T}_i} c\varphi_{ik}\gamma_{kj}^{(t)} \tag{7}$$

The parameter c ($0 \le c \le 1$) specifies the relative contributions from the social influence and the initial label. φ_{ik} is defined as tie strength, which balances the heterogenous influence from different neighborhoods ($k \in \mathcal{T}_i$) and meets $\sum_{k \in \mathcal{T}_i} \varphi_{ik} = 1$. Overall, SamWalker replaces γ_{ij} with a social context-aware function g parameterized by φ, to which the equation (7) converges:

$$Y = g_\varphi(X, T) \equiv \lim_{t \to \infty} Y^{(t)} = (I - c\Phi)^{-1}(1 - c)X \tag{8}$$

where we collect variables $\gamma_{ij}^{(t)}$ for every user-item pairs (i, j) as a matrix $Y^{(t)}$. Also, we collect φ_{ik} as a matrix Φ, in which $\Phi_{ij} = \varphi_{ij}$ for connected user pairs and $\Phi_{ij} = 0$ for others. As we can see from equation (8), SamWalker replaces the posterior expectation of user's exposure with a weighted combination of the users' consumption in his social network. The weight matrix $(I - c\Phi)^{-1}$ is a graph or diffusion kernel [59], which has been widely adopted to measure nodes proximity in the network and depends on the tie strength parameters φ for every social ties. Overall, SamWalker can capture

[2]Note that the EM algorithm presented in [30] is a special case of the classic variational inference.

the heterogeneous social influence between users and reduce the number of variational parameters from $O(n \times m)$ to $O(|E|)$, where $|E|$ denotes the number of ties. By iteratively learning the transformation function and user's preference, SamWalker can adaptively specify different weights to different data based on users' social contexts.

4.1 Informative Sampler based on personalized random walk

As mentioned above, inferring user's preference from implicit feedback data is time-consuming due to the massive volume of the unobserved data. Thus, for efficient recommendation, SamWalker employs stochastic inference based on data sampling. That is, in each iteration, we update the preference parameters θ based on the estimated gradient from only a subset of the data. Note that different data may have different confidence weights. This makes the data sampling strategy important because it determines which data are used to update parameters and how often. Intuitively, the informative data with larger confidence γ_{ij} should be sampled with larger probability, since these terms make more contribution to the objective function. In fact, we have the following lemma:

Lemma 1. *To evaluate the unbiased gradient of L w.r.t θ, the sampling strategy with distribution $p_{ij} \propto \gamma_{ij}$ can reduce sampling variance.*

The proof are presented in appendix.

Also, the sampler with distribution $p_{ij} \propto \gamma_{ij}$ can speed up gradient estimation. As we can see from the following equation (9) (here ℓ_{ij} as shorthand for $\ell(x_{ij}, \sigma(u_i^T v_j))$), when the data sampling probability is proportional to the data confidence ($p_{ij} = \gamma_{ij}/G$), the confidence weights γ_{ij} can be absorbed into the sampling bias. Thus, in each iteration the gradient estimator does not need to calculate current learned confidence weights γ_{ij} to scale down the contribution from different data, which saves much time.

$$\frac{\partial L}{\partial \theta} = \sum_{i \in U, j \in I} \gamma_{ij} \frac{\partial \ell_{ij}}{\partial \theta} = \sum_{i \in U, j \in I} \frac{p_{ij}}{G} \frac{\partial \ell_{ij}}{\partial \theta} = E_p[\sum_{(a,b) \in p} \frac{1}{G} \frac{\partial \ell_{ab}}{\partial \theta}] \quad (9)$$

A naive implement of informative sampler with distribution $p_{ij} \propto \gamma_{ij}$ is to estimate and rank the current learned confidence weights γ_{ij} for every user-item pairs and then pick out the informative data based on γ_{ij}. It is apparently inefficient and can't satisfy practical requirement. To avoid having to estimate $n \times m$ confidence weights, we propose a subtle sampling strategy based on random walk along the network. Here we do some transformation of equation (8) to bring some insights:

$$\begin{aligned} Y &= (I - c\Phi)^{-1}(1-c)X \\ &= (1 + c\Phi + (c\Phi)^2 + (c\Phi)^3 \ldots)(1-c)X \end{aligned} \quad (10)$$

We can simulate the equation (10) and develop following sampling strategy:

For target user i, we perform the personalized random walk from user i to sample the informative feedback data of user i. At each step t of random walk, we are at a certain user u. We have two options: (1) With probability c, we do not continue the random walk. We stay at user u and randomly (uniformly) select a portion of (N_u/β) items that have been consumed by user u, where N_u denotes the number of items consumed by user u. Then we add the feedback data of user i on these selected items into sampled set S. (2) With probability $(1-c)$, we continue our random walk. We randomly select one of u's connected friends v ($v \in \mathcal{T}_u$) based on personalized tie strength (transition probability) φ_{uv} and walk to v for the next walk step.

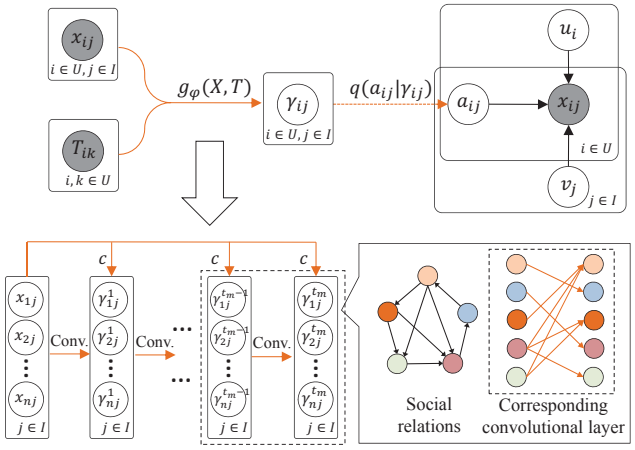


Figure 2: A schematic view of proposed model SamWalker (upper), the transformation function $g_\varphi(X, T)$ based on multi-layer convolutions (bottom-left), and the convolutional layer for the specific social network (bottom-right).

It is easy to check that the (i, v)-th element of matrix $(c\Phi)^t(1-c)$ is the probability of starting from source user i and terminating at user v in step t. Further, the (i, j)-th element of matrix $(c\Phi)^t(1-c)X$ represents the sampled probability of the user-item feedback data x_{ij} in step t. Sum over the probability in different step, we have the sampled probability of the data as follow:

$$P = \sum_{t=0}^{\infty} (c\Phi)^t (1-c)X/\beta \propto Y \quad (11)$$

which is proportional to the data confidence. Overall, our sampling strategy can efficiently draw informative training instances to speed up inference and reduce sampling variance.

In practice, we usually conduct α times random walk for each user to achieve more reliable mini-batch stochastic optimization. The paremeters α and β control the batch size. Note that there is a chance for a single random walk to continue forever. In fact, we pay more attention to user's local social context and thus terminate the random walk when we go very far from the source user ($t > t_m$). Concretely, when $t > t_m$, we walk to random user in the system (uniformly) and generated sampled data as option (1).

4.2 Inference of the personalized tie strength φ

Chaney et al. [7] and Wang et al. [46] claim that different social ties may have different influence strength. For example, we are more likely to get information from our close friends than from our acquaintances. Thus, the parameters φ of tie strength in transformation function tend to be diverse and is thus hard to be assigned manually. To deal with this problem, SamWalker fits transformation function and learns its parameters φ from the data.

Algorithm 1 Inference of SamWalker

1: Initialize parameters randomly;
2: **while** not converge **do**
3: Sample a set of data S based on the random walk as mentioned in subsection 4.1.
4: Update parameters θ of the preference model based on the estimated gradient from the sampled data (equation (9)).
5: randomly select a portion of N_{SI} items.
6: update personalized tie strength φ based on backward propagation along the neural network for the selected items.
7: **end while**

We achieve this by optimizing the lower bound of margin likelihood (equation 5) w.r.t tie strength φ_{ij} based on gradient methods. However, directly deriving gradient from transformation function g_φ (equation (8)) involves matrix inversion and suffers from low efficiency. Alteratively, as illustrated in Figure 2, SamWalker iteratively simulates information spread as equation (7), and constructs equivalent social-based convolutional neural network [28, 56] to infer the personalized tie strength. In fact, using convolutional neural network to model information spread in social network is natural: The nodes in t-th layer can represent users' exposure $\gamma_{ij}^{(t)}$ in t-th iteration. The forward process from layer t to $t+1$ models t-th information spread. That is, nodes/users collect the information from their social connected friends (neighbors) and generate new representation in $t+1$ layer:

$$\gamma_{ij}^{(t+1)} = \rho\Big(\sum_{k\in N_i} c\varphi_{ik}\gamma_{kj}^{(t)} + (1-c)x_{ij}\Big) \tag{12}$$

where $\rho(.)$ is an activation function. In SamWalker, we use identity activation function $\rho(x) = x$. In fact, other activation function can be employed for different prorogation pattern (e.g. sigmoid function for Linear Threshold [11]). Further, attentive mechanism has been employed to balance the social influence from different social friends (neighbors). Here we simply use independent attention and reparameterize $\varphi_{ik} = \frac{\exp(w_{ik})}{\sum_{k\in N_i}\exp(w_{ik})}$ using a Softmax transformation. In fact, more sophisticated attentive network can be employed to capture complex and context-aware tie strength [8, 21]. We will leave it in the future work.

With above neural network, backward prorogation can be conducted to infer tie strength φ, without requiring time-consuming matrix inversion. Further, mini-batch-based stochastic gradient methods can be employed to speed up the inference. That is, in each step, we randomly (uniformly) select a portion of N_{SI} items and update tie strength based on users' exposure on these selected items. Overall, the inference of our SamWalker is presented in Algorithm 1.

4.3 Discussion of the SamWalker

Global influence Vs. local influence. If a user has no connections with others, the method will face degeneration when estimating user's exposure or conducting random walk. To deal with this problem, we borrow the idea from Pagerank and consider a user may also get information from a random user. That is, when estimating user's exposure, users are assumed to link out to all other users but with very weak common tie strength. Thus, we revise equation (7) as follow:

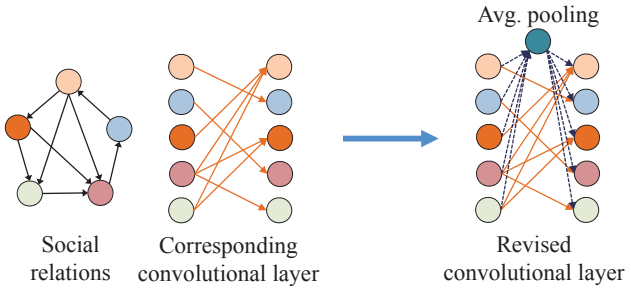


Figure 3: The revised convolutional layer for modeling both local influence and global influence.

$$\gamma_{ij}^{(t+1)} = (1-c)X_{ij} + c\Big(\sum_{k\in\mathcal{T}_i}\varphi_{ik}\gamma_{kj}^{(t)} + \frac{\varphi_{i0}}{n}\sum_{k\in U}\gamma_{kj}^{(t)}\Big) \tag{13}$$

where we introduce the influence from all other users to iteratively reconstruct user's exposure. Also, the parameters $\varphi_{i0}, \varphi_{ik}$ have been employed to balance the local social influence from connected users and the global social influence from all other users (i.e. $\varphi_{i0} + \sum_{k\in\mathcal{T}_i}\varphi_{ik} = 1$). From another perspective, the additional global term can be considered as the mean priori of users' exposure. As claimed by [36], users' selection behaviour is mainly driven by the local social influence among friends while the global popularity plays a supplementary role driving the behaviour only when there is little local information for the user to refer to. Thus, On the one hand, we introduce the global influence to pull user's exposure closer to the global average. On the other hand, users have their personalized local social contexts and thus are exposed to diverse item information. Correspondingly, local social influence term has been introduced to push users' exposure away from average and towards diversity. Overall, we combine these two effects to better capture user's exposure.

Correspondingly, to let our sampling strategy adapt to the revised data confidence, we add random transition into random walk. That is, in each step of random walk, we add another option (3): With probability $c\varphi_{i0}$, we walk to another user ($v \in U$) in the network by random (uniformly) and continue the random walk on v.

Also, as illustrated in Figure 3, with additional pooling units, global influence can be easily integrated into the social-based convolutional neural network to infer the tie strength $\varphi_{i0}, \varphi_{ik}$. Formally, we revise equation (12) as:

$$\bar{\gamma}_j^{(t)} = \frac{1}{n}\sum_{k\in U}\gamma_{kj}^{(t)}$$
$$\gamma_{ij}^{(t+1)} = \rho\Big(c\Big(\sum_{k\in N_i}\varphi_{ik}\gamma_{kj}^{(t)} + \varphi_{i0}\bar{\gamma}_j^{(t)}\Big) + (1-c)x_{ij}\Big) \tag{14}$$

Complexity Analysis. The computational time of the inference of SamWalker consists of three parts: (1) In sampling, we need to conduct α times random walk for each user to get sampled data set S. The time for sampling is $O(\alpha n t_m + |S|)$, where n denotes the number of users in the system, t_m denotes the max depth of random walk

Table 1: Statistics of three datasets.

Datasets	#Users	#Items	#Links	#User-item interactions
LastFM	1,892	4,489	25,434	52,668
Ciao	5,298	19,301	106,640	138,840
Epinions	21,290	34,075	414,549	333,916

and $|S|$ denotes the number of data in the set S. (2) When inferring the preference parameters θ, we just estimate loss and gradient for the sampled data based on equation (9). The time to update these parameters is $O(|S|d)$. (3) We infer the tie strength φ based on the gradient back propagation along the neural network for the selected N_{SI} items. The complexity for this part is $O(N_{SI}|E|)$. Hence, the overall computational complexity is $O(\alpha n t_m + |S|d + N_{SI}|E|)$. In practise, we usually let $|S|$ be five times as large as the number of observed data and let the number of selected items N_{SI} be 100. Thus, our algorithm is efficient on sparse implicit feedback data.

5 EXPERIMENTS AND ANALYSES

In this section, we conduct experiments to evaluate the running time and the recommendation quality of SamWalker. Our experiments are intended to address the following questions:

(Q1) Does SamWalker outperform state-of-the-art recommendation methods?
(Q2) How does the proposed social-based sampling strategy perform?
(Q3) Is it beneficial to model the personalized tie strength? Is it beneficial to model the both local social influence and global popularity influence?
(Q4) How does the parameter t_m (the max depth of random walk) affect the recommendation performance?

5.1 Experimental protocol

Datasets. Three datasets Epinions[3], Ciao[4], LastFM[5] are used in our experiments. These datasets contain users' feedback and social relations. Specifically, the datasets Epinions and Ciao contain users' ratings on movies, while the dataset LastFM contains users' clicks on music. The dataset statistics are presented in Table 1. Similar to [18, 50], we preprocess the datasets so that all items have at least three interactions and "binarize" user's feedback into implicit feedback. That is, as long as there exists some user-item interactions (ratings or clicks), the corresponding implicit feedback is assigned a value of 1. Also, we drop out items that have been consumed by too many (larger than 100) or too few (smaller than 3) users to moderate the popularity biases [5] in estimation. Grid search and 5-fold cross validation are used to find the best parameters. In our experiments, we set $\eta_{ij} = 0.1$, $\alpha = 100$, $\beta = 20$, $c = 0.9$, $t_m = 5$. All experiments are conducted on a server with 2 Intel E5-2620 CPUs and 256G RAM[6].

Compared methods. We compare SamWalker with following baseline methods. Table 2 concludes the characteristics of them.

[3]http://www.trustlet.org/epinions
[4]http://www.cse.msu.edu/~tangjili/trust
[5]https://grouplens.org/datasets/hetrec-2011/
[6]Source code will be available at https://github.com/jiawei-chen/Samwalker

Table 2: The characteristics of the compared methods.

Methods	Social?	Exposure-based?	Sample?	Complexity
WMF(ALS)	\	\	\	$O((n+m)d^3)$
BPR	\	\	√	$O((n+m+\|S\|)d)$
SBPR	√	\	√	$O((n+m+\|S\|)d)$
SPF	√	\	\	$O((n+m))d$
CDAE	\	\	√	$O((n+m+\|S\|)d)$
EXMF	\	√	\	$O(nmd)$
SERec-Bo	√	√	\	$O(nmd)$
SoEXBMF	√	√	\	$O(nmd^2)$
SamWalker	√	√	√	$O(\alpha n t_m + \|S\|d + N_{SI}\|E\|)$

- WMF(ALS) [22, 35]: The classic weighted matrix factorization model for implicit feedback data. The corresponding ALS-based [22] algorithm can reduce inference complexity.
- BPR [38]: The classic pair-wise method for recommendation, coupled with matrix factorization. For efficient recommendation, BPR employs uniform sampling strategy to draw the training instances.
- SBPR[58]: SBPR integrates social information into BPR by assuming that the items consumed by connected friends are ranked higher than those not.
- SPF [7]: A social recommendation model that incorporates social influence with users' latent preference based on poisson factorization.
- CDAE [49]: The advanced recommendation method based on Auto-Encoders, which is a generalization of WMF with more flexible components. However, efficient ALS-based inference algorithm are no longer suitable. Thus CDAE employs uniform sampling strategy to draw the training instances.
- EXMF [30]: A probabilistic model that directly incorporates user's exposure to items into traditional matrix factorization. EXMF does not utilize social information and chooses an item-dependent prior of user's exposure.
- SERec-Bo[45]: A probabilistic model that extends the EXMF model with social influence on user's exposure. Note that in [45] the authors reported that the performance of SERec-Bo is consistently better than their other model SERec-Re. Thus, here we choose SERec-Bo as a comparison.
- SoEXBMF [26]: A probabilistic model that further extends the EXMF model with both social knowledge influence and social consumption influence on user's exposure.

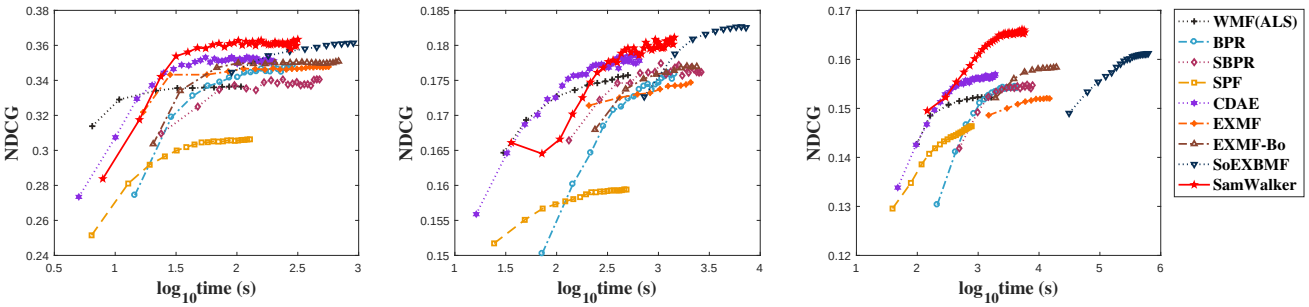
Evaluation Metrics. We adopt four well-known metrics Recall@K (Rec@K), Precision@K (Pre@K), Normalized Discounted Cumulative Gain (NDCG) and Mean Reciprocal Rank (MRR) to evaluate recommendation performance: Recall@K (Rec@K) quantifies the fraction of consumed items that are in the top-K ranking list sorted by their estimated rankings; Precision@K (Pre@K) measures the fraction of the top-K items that are indeed consumed by the user; NDCG and MRR evaluate ranking performance of the methods, which can be interpreted as the ease of finding all consumed items, as higher numbers indicate the consumed items are higher in the list. Refer to [47] for more details about these metrics.

5.2 Performance comparison (Q1)

Table 3 presents the performance of the compared methods in terms of three evaluation metrics. The boldface font denotes the winner in that column. Overall, with few exceptions, SamWalker outperforms

Table 3: The performance metrics of the compared methods. The boldface font denotes the winner in that column. The row 'Impv-e' indicates the relative performance gain of our SamWalker compared to the best results among these efficient baselines including WMF(ALS), BPR, CDAE, SPF, SBPR. The row 'Impv-a' indicates the relative performance gain of our SamWalker compared to the best results among all the baselines.

Methods		LastFM				Ciao				Epinions			
		Pre@5	Rec@5	NDCG	MRR	Pre@5	Rec@5	NDCG	MRR	Pre@5	Rec@5	NDCG	MRR
Efficient	WMF(ALS)	0.0928	0.0841	0.3364	0.3552	0.0172	0.0123	0.1757	0.0738	0.0095	0.0087	0.1522	0.0446
	BPR	0.1004	0.0888	0.3485	0.3704	0.0144	0.0125	0.1759	0.0649	0.0087	0.0096	0.1541	0.0406
	SBPR	0.0956	0.0851	0.3405	0.3556	0.0156	0.0124	0.1774	0.0678	0.0088	0.0089	0.1542	0.0411
	SPF	0.0714	0.0622	0.3132	0.2909	0.0081	0.0061	0.1603	0.0379	0.0052	0.0063	0.1464	0.0276
	CDAE	0.1008	0.0890	0.3509	0.3784	0.0144	0.0116	0.1778	0.0647	0.0092	0.0112	0.1570	0.0423
Inefficient	EXMF	0.0957	0.0859	0.3477	0.3665	0.0095	0.0105	0.1747	0.0485	0.0051	0.0082	0.1513	0.0275
	SERec-Bo	0.1018	0.0907	0.3509	0.3787	0.0118	0.0124	0.1770	0.0532	0.0073	0.0101	0.1577	0.0365
	SoEXBMF	0.1108	0.1014	0.3617	0.4140	0.0181	0.0152	**0.1827**	0.0775	0.0120	0.0130	0.1612	0.0537
Proposed	SamWalker	**0.1177**	**0.1072**	**0.3634**	**0.4250**	**0.0182**	**0.0167**	0.1811	**0.0775**	**0.0149**	**0.0184**	**0.1656**	**0.0628**
	Impv-e%	16.8%	20.4%	3.6%	12.3%	5.9%	34.0%	1.9%	5.0%	56.0%	64.2%	5.5%	40.9%
	Impv-a%	6.2%	5.7%	0.5%	2.7%	0.2%	10.4%	-0.8%	0.0%	23.9%	41.1%	2.7%	16.9%



(a) On dataset LastFM (b) On dataset Ciao (c) On dataset Epinions

Figure 4: NDCG for each method in different steps versus time

all compared methods on all datasets for all metrics. For the sake of clarity, the last two rows of Table 3 also show the relative improvements achieved by SamWalker over the efficient baselines (Impv-e) and over all baselines (Impv-a) respectively. The improvement of SamWalker over the efficient baselines is apparent. Especially on the large dataset Epinions, the improvement is 56.0%, 64.2%,5.5%,40.9% in terms of Pre@5, Rec@5, NDCG, MRR respectively. The improvement of SamWalker over these efficient baselines can be attributed to two aspects: (1) In the real world, users have personalized social contexts and thus are exposed to diverse information. Correspondingly, data have diverse confidence for estimating user's preference. That is, some un-observed feedback are more likely attributed to user's preference while others are the results of users' awareness. By adaptively fitting fine-grained data confidence weights based on users' social contexts, SamWalker achieves better performance than those baselines with manually coarse-grained confidence weights. (2) Instead of employing pre-defined coarse-grained sampling methods, SamWalker employs random walk-based sampler to adaptively draw informative data, which can reduce sampling variance and achieve better performance. We also conduct specific experiments for different samplers in subsection 5.3 to validate this point.

Also, even although SamWalker employs stochastic optimization to reduce time complexity, SamWalker still beat these exposure-based methods (EXMF, SERec-Bo, SoEXBMF). This result validate the effectiveness of employing social-based variational posterior of user's exposure, which models the heterogenous influence from both direct and indirect social friends (neighbors). Although SoEXBMF can also capture such influence, it costs much more running time and requires pre-defined community distribution. In fact, accurate community distribution is not easy to get, which affects the performance of SoEXBMF model.

Runtime vs. NDCG. Figure 4 depicts running time (X-axis) vs. NDCG (Y-axis) of the nine compared recommendation methods. As we can see, generally, SamWalker achieves best performance. The powerful competitor is CDAE, which models user's preference based on collaborative auto-encoder. Although CDAE has better NDCG than SamWalker at first, SamWalker overtakes CDAE soon and before CDAE gets convergence. Especially on large dataset Epinions, SamWalker outperforms CDAE almost in all time and achieves much better performance than CDAE when they get convergence. Also, we observe that these exposure-based methods (EXMF,SERec,SoEXBMF) achieve good performance but cost a lot of time. Although these methods can learn the personalized confidence of the data, they need infer $n \times m$ parameters of user's exposure, which will suffer from low efficiency and overfitting. By employing social-based posterior structure of exposure and random walk-based sampler, our method SamWalker can beat

Table 4: Average empirical variance of the estimated gradients using different sampling strategies. Here $X^{(1)}$, $X^{(0)}$ denote the number of ones or zeros in the matrix X. Similarly, $r_i^{(1)}$, $r_i^{(0)}$ denote the number of ones or zeros in the i-th row of matrix X and $c_j^{(1)}$, $c_j^{(0)}$ denote the number of ones or zeros in the j-th column of matrix X.

Sampling strategy	Distribution	Average Variance					
		Batchsize=5$\|X^{(1)}\|$			Batchsize=$\|X^{(1)}\|$		
		50 It.	100 It.	500 It.	50 It.	100 It.	500 It.
S-allunion	$p_{ij} = 1/(n \times m)$	2.3793	2.5640	2.6663	11.8566	13.2071	13.2475
S-balunion	$p_{ij} = 1/(2\|X^{(x_{ij})}\|)$	0.4154	0.4941	0.5368	2.0834	2.4732	2.6779
S-itempop	$p_{ij} = I[x_{ij} = 1]/(2\|X^{(1)}\|) + I[x_{ij} = 0]c_j^{(1)}/(2\sum_{1\le b\le m} c_b^{(1)})$	0.1720	0.1938	0.2190	0.8702	0.9890	1.1052
S-cobias	$p_{ij} = r_i^{(1-x_{ij})} c_j^{(1-x_{ij})}/(2\sum_{1\le a\le n}\sum_{1\le b\le m} I[x_{ab} = x_{ij}] r_a^{(1-x_{ij})} c_b^{(1-x_{ij})})$	0.1617	0.1874	0.2031	0.8074	0.9663	1.0647
Random Walk	$p_{ij} \propto \gamma_{ij}$	0.1358	0.1464	0.1510	0.6469	0.6977	0.7038

EXMF,SERec,SoEXBMF in both running time and recommendation performance.

5.3 Performance of the different samplers (Q2)

In this subsection, we compare our sampler with other sampling strategies including: (1) S-allunion, the uniform sampling strategy; (2) S-balunion [19, 35], which samples un-observed data (zeros) and observed data (ones) with equal probability to deal with unbalance data problem; (3) S-itempop [54], which samples data based on item popularity; (4) S-cobias [19], whose probability of sampling a un-observed data (zero) at location (i, j) is proportional to the number of ones in the i-th row (the number of user's consumption) and the number of zeros in the j-th colunm (item popularity). Also, an equivalent bias is introduced for the observed data. The detailed distributions of these sampling strategies are presented in Table 4.

Variance. We first empirically compare the variance of the estimated gradient of our objective w.r.t θ by using different sampling strategy. To do this, we train our model for 50,100 or 500 epochs on dataset LastFM. After training, we generate mini-batch with various sampling strategy and calculate un-biased estimated gradients of our objective w.r.t θ. We repeat this proceeding for 1000 times and calculate the variance of the estimated gradients for different sampling strategy. The final results presented in Table 4 are averaged over various preference parameters. Table 4 shows that our random walk-based sampling strategy achieves the lowest average variance among various samplers for all conditions. This result is coincide with our theory analysis in section 4.1. Our sampler, whose sampling probability is proportional to the data confidence, can indeed reduce the sampling variance. Also, we observe the following interesting phenomenon: With more training epoches, the variance will become larger, not smaller as usual. It may be explained as follow: SamWalker models the diverse data confidence and the personalized tie strength. At first, SamWalker keeps relative similar data confidence. With training proceeding, driven by the data, the confidence weights and the tie strength in SamWalker exhibit more and more heterogeneity. Thus, the variance of the estimated gradients will become larger.

Performance. Figure 5 presents the NDCG of SamWalker on dataset LastFM with different sampling strategies versus the number of iteration and running time. As we can see, our adaptive random walk-based sampler performs better than others in all convergence, speed and recommended performance. One reason is that our sampler has low variance, as presented in Table 4. Another reason is that in our sampler the confidence weights γ_{ij} are absorbed into the bias of the sampling distribution. Thus, the gradient estimator does not need to calculate γ_{ij} in each iteration to scale down the contribution from the different data, which saves much time.

5.4 Heterogenous social influence (Q3)

To show the effect of modeling heterogenous social influence, we compare SamWalker with its three special case: (1) SamWalker-ho, the special case of SamWalker with homogeneous tie strength. (2) SamWalker-g (item-pop): the special case which leave out heterogenous local social influence and models user's exposure just based on item popularity. (3) SamWalker-l: the special case which leaves out global influence. The performance of these special cases comparing with SamWalker is presented in Figure 6.

Personalized tie strength. As shown in Figure 6, SamWalker with personalized tie strength consistently achieves better performance than its special case SamWalker-ho with the homogeneous tie strength in all three datasets. These results support our intuition that different ties may have different influence strength. For example, the items consumed by our close friends that have high frequency of interactions, are more likely to come to our attention than those items consumed by our acquaintances.

Global influence Vs. local influence. Figure 6 also shows that SamWalker outperforms its special cases SamWalker-g and SamWalker-l. The result validates our point that user's exposure will be affected by both local social influence and global popularity influence. Especially, as claimed by some sociologic related works [36], user's social relations have become a major information resource when he select items to consume. Thus, we observe the apparent improvement of SamWalker over the special case SamWalker-g (item-pop) which leaves out local social influence.

5.5 Effect of parameter t_m (Q4)

It will be interesting to explore how parameter t_m affects the performance of SamWalker, where t_m indicates the max depth of the random walk. As we can see from Figure 7, as t_m become larger, with few exception, the performance will become better first. The

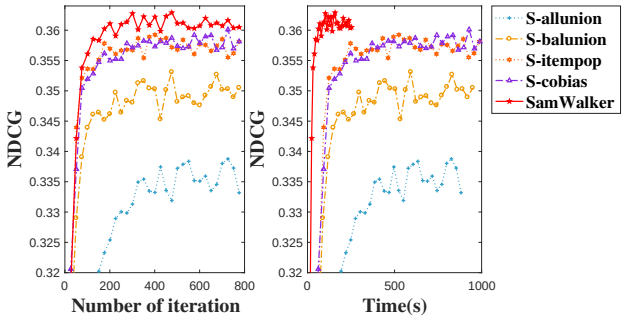
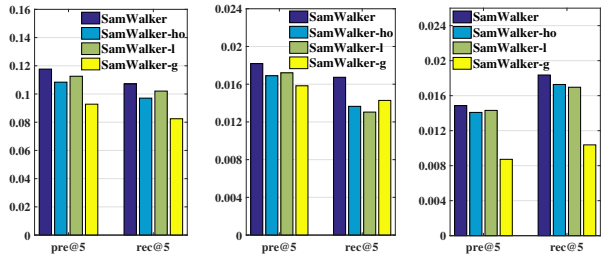
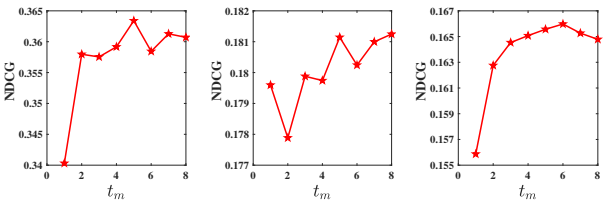


Figure 5: NDCG for different sampling strategy versus the number of iterations and running time



(a) On dataset LastFM (b) On dataset Ciao (c) On dataset Epinions

Figure 6: Performance comparison of SamWalker with its three special cases



(a) On dataset LastFM (b) On dataset Ciao (c) On dataset Epinions

Figure 7: Performance comparison with varying t_m

reason is that the information about items propagates along the social network. User's exposure will also be influenced by his high order neighbors. However, when t_m surpasses a threshold ($t_m > 5$), the performance becomes unaffected or even worse with further increase of t_m. In fact, based on the idea of "six-degrees of separation [43]", the random walk with length 5 can visit most of the nodes in the social network. Too deep random walk will not bring more usual information and sometimes bring some noises.

6 CONCLUSIONS

In this paper, we present a novel recommendation method SamWalker that leverages social information to simultaneously learn the personalized data confidence and draw informative training instances. On the one hand, SamWalker models the posterior of user's exposure with a social context-aware function, which can reduce the number of learned parameters and adaptively specify different confidence weights to different data. On the other hand, we propose a personalized random walker-based sampling strategy to draw informative training instances to speed up inference and reduce sampling variance. The personalized transition probability (tie strength) can be inferred efficiently based on our attentive neural network. The experimental results on three real-world datasets demonstrate the superiority of SamWalker over existing methods.

One interesting direction for future work is to explore dynamic exposure-based recommendation. In the real world, users' preference, exposure and relations may evolve over time. Further, we did not examine other attentive structure when modeling tie strength. It will be also interesting to develop more complex attentive model and explore context-aware social influence.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (Grant No: U1866602) and by a Discovery Grant from the National Science and Engineering Research Council of Canada. It is also partially supported by ByteDance.

A PROOF OF LEMMA 1

Given the confidence weights γ_{ij}, we have the following un-biased gradient estimator of the objection (equation 5) based on the specific sampler p_{ij}:

$$\nabla_\theta L = \sum_{i\in U, j\in I} \gamma_{ij} \nabla_\theta \ell_{ij} = E_p[\frac{nm}{|S|} \sum_{a,b\in p} \frac{\gamma_{ab} \nabla_\theta \ell_{ab}}{p_{ab}}] \quad (15)$$

where ℓ_{ij} is shorthand for $\ell(x_{ij}, \sigma(u_i^T v_j))$ and $|S|$ denotes mini-batch size. Then, we have following variance of the gradient estimator:

$$\begin{aligned} & Var_{(a,b)\sim p}[\frac{nm}{|S|} \sum_{a,b\in p} \frac{\gamma_{ab} \nabla_\theta \ell_{ab}}{p_{ab}}] \\ &= \frac{n^2 m^2}{|S|} Var_{(a,b)\sim p}[\frac{\gamma_{ab} \nabla_\theta \ell_{ab}}{p_{ab}}] \\ &= \frac{n^2 m^2}{|S|} (E_p[(\frac{\gamma_{ab} \nabla_\theta \ell_{ab}}{p_{ab}})^2] - {E_p}^2[\frac{\gamma_{ab} \nabla_\theta \ell_{ab}}{p_{ab}}]) \\ &= \frac{n^2 m^2}{|S|} E_p[(\frac{\gamma_{ab}^2}{p_{ab}^2} \nabla_\theta^2 \ell_{ab}] - (\sum_{i\in U, j\in I} \gamma_{ij} \nabla_\theta \ell_{ab})^2 \end{aligned} \quad (16)$$

It is untractable to find optimized p_{ij} based on equation (16) due to the unknown of the terms $\nabla_\theta^2 \ell_{ab}$. But we can derive the tight upper-bound of the variance based on Cauchy inequality as follow:

$$\begin{aligned} & Var_{(a,b)\sim p}[\frac{nm}{|S|} \sum_{a,b\in p} \frac{\gamma_{ab} \nabla_\theta \ell_{ab}}{p_{ab}}] \\ &= \frac{n^2 m^2}{|S|} E_p[(\frac{\gamma_{ab}^2}{p_{ab}^2} \nabla_\theta^2 \ell_{ab}] - (\sum_{i\in U, j\in I} \gamma_{ij} \nabla_\theta \ell_{ab})^2 \\ &\le \frac{n^2 m^2}{|S|} \sum_{ij} \frac{\gamma_{ij}^2}{p_{ij}} \times E_{p,\ell_{ab}}[\nabla_\theta^2 \ell_{ab}] - (\sum_{i\in U, j\in I} \gamma_{ij} \nabla_\theta \ell_{ab})^2 \end{aligned} \quad (17)$$

Also, we can find the minimum of $\sum_i \frac{\gamma_{ij}^2}{p_{ij}}$ as follow:

$$\sum_i \frac{\gamma_{ij}^2}{p_{ij}} = \sum_i \frac{\gamma_{ij}^2}{p_{ij}} \sum_i p_{ij} \geq (\sum_i \frac{\gamma_{ij}}{\sqrt{p_{ij}}} \sqrt{p_{ij}})^2 = (\sum_i \gamma_{ij})^2 \quad (18)$$

The equation holds if and only if $p_{ij} \propto \gamma_{ij}$. Now, the sampling strategy has the lowest upper-bound of the variance.

REFERENCES

[1] Luca Maria Aiello, Alain Barrat, Ciro Cattuto, Rossano Schifanella, and Giancarlo Ruffo. 2012. Link creation and information spreading over social and communication ties in an interest-based online social network. *EPJ Data Science* 1, 1 (2012), 12.

[2] Luca Maria Aiello, Alain Barrat, Rossano Schifanella, Ciro Cattuto, Benjamin Markines, and Filippo Menczer. 2012. Friendship prediction and homophily in social media. *ACM Transactions on the Web (TWEB)* 6, 2 (2012), 9.

[3] Y Bao, H Fang, and J Zhang. 2014. Leveraging decomposed trust in probabilistic matrix factorization for effective recommendation. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI)*. 350.

[4] Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. 2017. A generic coordinate descent framework for learning from implicit feedback. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1341–1350.

[5] Rocío Cañamares and Pablo Castells. 2018. Should i follow the crowd?: A probabilistic analysis of the effectiveness of popularity in recommender systems. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 415–424.

[6] Pablo Castells, Neil J Hurley, and Saul Vargas. 2015. Novelty and diversity in recommender systems. In *Recommender Systems Handbook*. Springer, 881–918.

[7] Allison JB Chaney, David M Blei, and Tina Eliassi-Rad. 2015. A probabilistic model for using social networks in personalized item recommendation. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 43–50.

[8] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 335–344.

[9] Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On sampling strategies for neural network-based collaborative filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 767–776.

[10] Wei Chen, Laks VS Lakshmanan, and Carlos Castillo. 2013. Information and influence propagation in social networks. *Synthesis Lectures on Data Management* 5, 4 (2013), 1–177.

[11] Wei Chen, Yifei Yuan, and Li Zhang. 2010. Scalable influence maximization in social networks under the linear threshold model. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 88–97.

[12] Jingtao Ding, Fuli Feng, Xiangnan He, Guanghui Yu, Yong Li, and Depeng Jin. 2018. An improved sampler for bayesian personalized ranking by leveraging view data. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 13–14.

[13] Jennifer Golbeck. 2006. Generating predictive movie recommendations from trust in social networks. In *International Conference on Trust Management*. Springer, 93–104.

[14] Jennifer Golbeck. 2009. Trust and nuanced profile similarity in online social networks. *ACM Transactions on the Web (TWEB)* 3, 4 (2009), 12.

[15] Ido Guy. 2015. Social recommender systems. In *Recommender Systems Handbook*. Springer, 511–543.

[16] Ido Guy, Alejandro Jaimes, Pau Agulló, Pat Moore, Palash Nandy, Chahab Nastar, and Henrik Schinzel. 2010. Will recommenders kill search?: recommender systems-an industry perspective. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 7–12.

[17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[18] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 549–558.

[19] José Miguel Hernández-Lobato, Neil Houlsby, and Zoubin Ghahramani. 2014. Stochastic inference for scalable probabilistic modeling of binary matrices. In *International Conference on Machine Learning*. 379–387.

[20] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. 2013. Stochastic variational inference. *The Journal of Machine Learning Research* 14, 1 (2013), 1303–1347.

[21] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1531–1540.

[22] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee, 263–272.

[23] Bert Huang, Angelika Kimmig, Lise Getoor, and Jennifer Golbeck. 2013. A flexible framework for probabilistic models of social trust. In *International conference on social computing, behavioral-cultural modeling, and prediction*. Springer, 265–273.

[24] Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 135–142.

[25] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. 2010. *Recommender systems: an introduction*. Cambridge University Press.

[26] Chen Jiawei, Feng Yan, Ester Martin, Zhou Sheng, Chen Chun, and Can Wang. 2018. Modeling Users' Exposure with Social Knowledge Influence and Consumption Influence for Recommendation. In *Proceedings of the 27th ACM International on Conference on Information and Knowledge Management*. ACM, 953–962.

[27] Mozhgan Karimi, Dietmar Jannach, and Michael Jugovac. 2018. News recommender systems–Survey and roads ahead. *Information Processing & Management* (2018).

[28] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[29] Dongsheng Li, Chao Chen, Qin Lv, Hansu Gu, Tun Lu, Li Shang, Ning Gu, and Stephen M Chu. 2018. AdaError: An Adaptive Learning Rate Method for Matrix Approximation-based Collaborative Filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 741–751.

[30] Dawen Liang, Laurent Charlin, James McInerney, and David M Blei. 2016. Modeling user exposure in recommendation. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 951–961.

[31] Moshe Lichman and Padhraic Smyth. 2018. Prediction of Sparse User-Item Consumption Rates with Zero-Inflated Poisson Regression. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 719–728.

[32] Hao Ma, Irwin King, and Michael R Lyu. 2009. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 203–210.

[33] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. 2008. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 931–940.

[34] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web*. Technical Report. Stanford InfoLab.

[35] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. *Proceedings - IEEE International Conference on Data Mining, ICDM* (2008), 502–511. https://doi.org/10.1109/ICDM.2008.16

[36] Xue Pan, Lei Hou, and Kecheng Liu. 2017. Social influence on selection behaviour: Distinguishing local-and global-driven preferential attachment. *PloS one* 12, 4 (2017), e0175761.

[37] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 273–282.

[38] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[39] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2015. Recommender systems: introduction and challenges. In *Recommender systems handbook*. Springer, 1–34.

[40] Yelong Shen and Ruoming Jin. 2012. Learning personal+ social latent factor model for social recommendation. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1303–1311.

[41] Qihao Shi, Can Wang, Jiawei Chen, Yan Feng, and Chun Chen. 2019. Location driven influence maximization: Online spread via offline deployment. *Knowledge-Based Systems* 166 (2019), 30–41.

[42] Jiliang Tang, Huiji Gao, and Huan Liu. 2012. mTrust: discerning multi-faceted trust in a connected world. In *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 93–102.

[43] Jeffrey Travers and Stanley Milgram. 1967. The small world problem. *Phychology Today* 1, 1 (1967), 61–67.

[44] Beidou Wang, Can Wang, Jiajun Bu, Chun Chen, Wei Vivian Zhang, Deng Cai, and Xiaofei He. 2013. Whom to mention: expand the diffusion of tweets by@ recommendation on micro-blogging systems. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1331–1340.

[45] Menghan Wang, Xiaolin Zheng, Yang Yang, and Kun Zhang. 2018. Collaborative Filtering With Social Exposure: A Modular Approach to Social Recommendation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.

[46] Xin Wang, Steven CH Hoi, Martin Ester, Jiajun Bu, and Chun Chen. 2017. Learning personalized preference of strong and weak ties for social recommendation. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1601–1610.

[47] Xin Wang, Wei Lu, Martin Ester, Can Wang, and Chun Chen. 2016. Social Recommendation with Strong and Weak Ties. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 5–14.

[48] Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, Vol. 11. 2764–2770.

[49] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 153–162.

[50] Lin Xiao, Zhang Min, Zhang Yongfeng, Liu Yiqun, and Ma Shaoping. 2017. Learning and transferring social and item visibilities for personalized recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 337–346.

[51] Bo Yang, Yu Lei, Dayou Liu, and Jiming Liu. 2013. Social collaborative filtering by trust. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*. AAAI Press, 2747–2753.

[52] Xiwang Yang, Harald Steck, and Yong Liu. 2012. Circle-based recommendation in online social networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1267–1275.

[53] Weilong Yao, Jing He, Guangyan Huang, and Yanchun Zhang. 2014. Modeling dual role preferences for trust-aware recommendation. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 975–978.

[54] Hsiang-Fu Yu, Mikhail Bilenko, and Chih-Jen Lin. 2017. Selection of negative samples for one-class matrix factorization. In *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 363–371.

[55] Lu Yu, Chuxu Zhang, Shichao Pei, Guolei Sun, and Xiangliang Zhang. 2018. Walkranker: A unified pairwise ranking model with multiple relations for item recommendation. AAAI.

[56] Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep learning based recommender system: A survey and new perspectives. *arXiv preprint arXiv:1707.07435* (2017).

[57] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 785–788.

[58] Tong Zhao, Julian McAuley, and Irwin King. 2014. Leveraging social connections to improve personalized ranking for collaborative filtering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 261–270.

[59] Denny Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *Advances in neural information processing systems*. 321–328.

[60] Sheng Zhou, Hongxia Yang, Xin Wang, Jiajun Bu, Martin Ester, Pinggang Yu, Jianwei Zhang, and Can Wang. 2018. PRRE: Personalized Relation Ranking Embedding for Attributed Networks. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. ACM, 823–832.

[61] Yu Zhu, Jinghao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2019. Addressing the Item Cold-start Problem by Attribute-driven Active Learning. *IEEE Transactions on Knowledge and Data Engineering* (2019).

[62] Cai-Nicolas Ziegler and Jennifer Golbeck. 2007. Investigating interactions of trust and interest similarity. *Decision support systems* 43, 2 (2007), 460–475.

How Serendipity Improves User Satisfaction with Recommendations? A Large-Scale User Evaluation

Li Chen
Hong Kong Baptist University
Hong Kong, China
lichen@comp.hkbu.edu.hk

Yonghua Yang
Alibaba Group
China
huazai.yyh@alibaba-inc.com

Ningxia Wang
Hong Kong Baptist University
Hong Kong, China
nxwang@comp.hkbu.edu.hk

Keping Yang
Alibaba Group
China
shaoyao@alibaba-inc.com

Quan Yuan
inspirAI Co. Ltd.
China
quanyuan007@gmail.com

ABSTRACT

Recommendation serendipity is being increasingly recognized as being equally important as the other beyond-accuracy objectives (such as novelty and diversity), in eliminating the "filter bubble" phenomenon of the traditional recommender systems. However, little work has empirically verified the effects of serendipity on increasing user satisfaction and behavioral intention. In this paper, we report the results of a large-scale user survey (involving over 3,000 users) conducted in an industrial mobile e-commerce setting. The study has identified the significant causal relationships from novelty, unexpectedness, relevance, and timeliness to serendipity, and from serendipity to user satisfaction and purchase intention. Moreover, our findings reveal that user curiosity plays a moderating role in strengthening the relationships from novelty to serendipity and from serendipity to satisfaction. Our third contribution lies in the comparison of several recommender algorithms, which demonstrates the significant improvements of the serendipity-oriented algorithm over the relevance- and novelty-oriented approaches in terms of user perceptions. We finally discuss the implications of this experiment, which include the feasibility of developing a more precise metric for measuring recommendation serendipity, and the potential benefit of a curiosity-based personalized serendipity strategy for recommender systems.

CCS CONCEPTS

• **Human-centered computing** → **Empirical studies in interaction design**; *User models*; *User studies*; • **Information systems** → *Recommender systems*; *Personalization.*

KEYWORDS

Recommender systems; serendipity; curiosity; user satisfaction; large-scale user evaluation

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313469

ACM Reference Format:
Li Chen, Yonghua Yang, Ningxia Wang, Keping Yang, and Quan Yuan. 2019. How Serendipity Improves User Satisfaction with Recommendations? A Large-Scale User Evaluation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313469

1 INTRODUCTION

Recommender systems have been popularly used in many Web applications to eliminate users' information overload, with recommended items/information being tailored to the users' individual interests. Traditionally, accuracy is the primary metric for judging a recommender system's effectiveness [15, 28, 40]. However, it may cause the "filter bubble" phenomenon in which users are trapped in a subspace of options that are too similar to their profile, and hence lose the opportunity to explore outside alternatives that may potentially match their preferences [26, 28, 34].

Therefore, some beyond-accuracy objectives, especially *diversity*, *novelty*, and *serendipity*, have been emphasized in the recent literature, because they are targeted to allow users to discover new and different items to broaden their horizons [12, 15, 26]. By definition, *diversity* refers to the difference between the current recommendation and the user's profile (e.g., the user's previously purchased items) or the system's prior recommendations [21][1]. *Novelty* stresses whether the item is unknown to the user. *Serendipity* emphasizes whether the user feels surprised when s/he sees a relevant recommendation, which implies that the item should not only be relevant to the user's interests, but also be unexpected (i.e., not intentionally looked for by the user). Therefore, different from diversity and novelty that may compromise accuracy to a certain degree, serendipity aims to preserve both accuracy and positive emotional response evoked by the recommendation [19, 20, 24].

However, because the "surprising" nature of serendipity is difficult to measure and simulate [6, 15, 29], the actual benefit of serendipity for recommender systems is still unclear. It is not conclusive whether the increased serendipity of recommendation would

[1] Another definition of diversity refers to the intra-list diversity within a set of recommended items [48], which is beyond the scope of this paper.

necessarily lead to users' higher satisfaction with the recommendation [19], and even their higher intention to purchase the recommended item if in an e-commerce domain.

In this study, we conducted a large-scale user survey (involving over 3,000 participants) to measure users' perceptions of recommendations, in relation to serendipity and its components (unexpectedness and relevance), novelty, diversity, user satisfaction, and purchase intention. The results are thus likely to help identify the causal relationships among these factors. For example, *are diversity and novelty significantly related to serendipity?* And *does serendipity significantly improve user satisfaction and purchase intention?* To design the questionnaire, we selected the popularly used questions from related literatures and translated them into Chinese (because our experimental system was plugged into a Chinese industrial mobile e-commerce app called *Mobile Taobao*). It is worth noting that the Chinese translation of the term "serendipity" is "惊喜" that contains two characters: '惊' (surprising) and '喜' (relevant), so it would be intuitive for Chinese users to easily interpret the meaning of "serendipity" when they answer the related questions.

In addition to acquiring users' perceptions of recommendations, we included a curiosity quiz to measure whether they have a strong desire for new knowledge or experiences in general. Indeed, curiosity has been widely regarded as an important antecedent of users' appetite for novelty in the field of psychology [2, 23, 39]. It can greatly affect the level of pleasure a user experiences when s/he explores new and surprising things. For instance, if an item under or over matches her/his curiosity, s/he may feel bored or overwhelmed, rather than pleasant [29, 47]. In this sense, the degree in which a recommendation is perceived as serendipitous might be more or less dependent on the user's curiosity. To verify this point, we integrated curiosity as a moderating factor into our path model to test its effects.

Our third contribution lies in the comparison of four recommender algorithms in relation to user perceptions, which include a non-personalized popularity-based approach, and three variants of the collaborative filtering (CF) based method that are respectively relevance-, novelty-, and serendipity-oriented. To the best of our knowledge, most of the existing algorithm evaluations for e-commerce products have relied on offline metrics [1, 12, 15, 30]. Given there is often a gap between offline measurements and user perceptions (especially regarding the beyond-accuracy objectives) [37], we believe it should be meaningful to conduct an online user study to evaluate whether users would be practically more satisfied with an algorithm's recommendation if it is perceived of higher serendipity. The results could also help improve the existing offline metrics (e.g., more precisely measuring serendipity).

In short, this study makes four major contributions. First, we conducted a large-scale online user survey in an industrial mobile e-commerce setting to collect user feedback on various beyond-accuracy assessments. Second, we validated a hypothesized path model that reveals the causal relationships from novelty, unexpectedness, relevance, and timeliness to serendipity, and from serendipity to user satisfaction and purchase intention. Third, we identified the moderating effects of user curiosity on the relationships from novelty to serendipity and from serendipity to satisfaction. Fourth, we experimentally compared four algorithms for recommending e-commerce products, and found a serendipity-oriented approach is significantly more effective in terms of enhancing user perceptions of the recommendation.

The remainder is organized as follows. We first introduce the related works on serendipity in recommender systems and those considering user curiosity to improve recommendations (Section 2). We then present our user survey, including the experimental procedure, data collection, questionnaire design, and hypothesized model (Section 3). The results are then analyzed through model validation, moderation test, and algorithm comparison (Section 4). We discuss the practical implications of our findings in Section 5, and draw the final conclusions in Section 6.

2 RELATED WORK

2.1 Serendipity in Recommender Systems

The original definition of serendipity is "[...] *making discoveries, by accidents and sagacity, of things which they were not in quest for* [...]" [3, 24, 42]. [12] is one of earlier papers to list serendipity as one of the beyond-accuracy objectives for recommender systems. Since then, serendipity has been used to measure how surprising and relevant a recommendation is [6, 20, 28]. In other words, this factor is comprised of two core components: *surprise* in the sense of unexpected (i.e., different from the user's expectations) and *relevance* (i.e., useful to the user).

However, the assessment of *surprise* is not straightforward because it involves the user's emotional response to a recommendation. In previous work, surprise (or called unexpectedness) was defined as the deviation from a primitive prediction method that produces expected recommendations [10, 30]. For example, [10] proposed a formulation that combines unexpectedness with item relevance, for which the set of unexpected recommendations is acquired by subtracting from those items generated by a primitive prediction model, and the relevance of a recommendation is approximated offline based on users' ratings. Considering this approach is sensitive to the choice of the primitive prediction model, [1] regarded items rated by the user and those similar to the rated ones to be the expected recommendations.

As stated in [15], evaluations of recommendation serendipity that do not involve user feedback can be unreliable, because it is not evident whether the recommended item would be perceived to be serendipitous by the user, even though some objective metric indicates it is. So far, few studies have empirically measured recommendation serendipity and its actual benefits from users' perspective. For instance, [46] conducted a small-scale user study (with 21 participants) to evaluate user perceptions of a serendipity-enhancing system that recommends music artists. The results showed most of the users overall preferred the serendipity-oriented system to the accuracy-oriented baseline especially in terms of serendipity and novelty, but gave lower ratings to serendipity-based recommendations. The authors concluded that the users are willing to compromise accuracy for serendipity, which nevertheless is contrary to the original definition of serendipity as comprising both relevance and unexpectedness. [33] designed a fusion-based recommender interface that allows users to experience extrinsic and intrinsic accidents for discovering serendipitous books. They recruited 9 users to evaluate the interface, which demonstrated its advantage over the standard Amazon interface as for serendipity.

In a more recent work, [6] combined the questionnaire and facial expression detection approaches in a preliminary user experiment (involving 40 subjects) performed on their proposed graph-based random walk algorithm. Two questions were asked to respectively assess a user's perceived relevance of a recommended movie and its unexpectedness. The user's facial expression was detected simultaneously to implicitly infer her/his degree of surprised when seeing the item. The experimental results showed their algorithm was perceived to be more relevant and serendipitous than a random recommendation method. They also found a moderate agreement between the questionnaire answers and facial expression detection results.

Given that there are different interpretations of serendipity in the related literature, [19] conducted a user survey (participated by 475 users) to acquire users' responses to eight different serendipity definitions in a movie recommender. The survey also aimed to verify whether it is meaningful to recommend serendipitous items. Specifically, by running a regression model on the users' answers to various serendipity assessments with the two dependent variables *preference broadening* and *user satisfaction*, the authors found that most definitions of serendipity helped broaden user preferences, but none of them affected user satisfaction.

The novelty of our work lies in revealing the relationships among the three beyond-accuracy factors, *novelty*, *diversity*, and *serendipity*, and their respective impacts on *user satisfaction* and *purchase intention*. From the algorithm's perspective, we compared a serendipity-oriented approach with three related methods, focusing on identifying their differences in terms of user perceptions. In addition, differing from the small-scale user studies in the related work, our study, to the best of our knowledge, is the first one performed on an industrial mobile e-commerce platform for a large-scale user experiment.

2.2 Recommendation based on User Curiosity

Curiosity, as a particular psychological trait that can affect a user's desire for exploration, has been found to likely influence the user's reaction to recommendations [29]. For example, it was shown that more curious people prefer recommendations containing some unexpected items, but less curious individuals prefer recommendations that are mostly similar to what they experienced before [29, 47].

Thus, some studies have attempted to incorporate curiosity into their recommendation methods. For example, [47] developed a curiosity-based music recommender, in which the recommendations can be personalized to the individual user's curiosity level. They mainly implemented a probabilistic curiosity model learnt for each user based on her/his access history. With this model, a curiousness score is computed for each item to indicate how curious the target user will be about it. Those items that are both relevant and suited to the user's curiosity level are then recommended. A major finding of their offline experiment is that the proposed method can improve the recommendation personalization and accuracy at the same time. In [29], a personalized recommendation architecture was presented in a tourism context, which aimed to adapt the degree of surprise of a recommended item to the user's curiosity. For this purpose, they proposed to predict a user's curiosity from her/his data shared on social network websites like Facebook, and then generate a recommendation list with the degree of unexpectedness tailored to the user's curiosity value. [24] also considered users' curiosity, with the objective of enhancing a TV program recommender. They adopted a psychological curiosity theory that defines two appraisals of curious emotion: *novelty check* and *coping potential check*. They then proposed to estimate the first appraisal based on an item's dissimilarity to those in the user profile and the second one based on the diversity of items within the user profile. A user study involving 165 participants showed that using curiosity to guide recommendations can be promising in terms of balancing serendipity and precision.

However, the above-described studies mostly inferred users' curiosity from their behavioral data and then incorporated the inferred curiosity into the process of generating recommendations. Little ground work has been done to verify the exact effect of curiosity on users' perceived recommendation serendipity. Therefore, we particularly investigated this effect in our user survey, which may be constructive for developing more personalized serendipity strategy based on user curiosity.

3 USER SURVEY

3.1 Setup and Measurement

We conducted a user survey on a popular mobile e-commerce platform in China (*Mobile Taobao*) from Dec. 21, 2017 to Jan. 11, 2018. The users were able to access the survey's link after they logged in the system. If a user volunteered to take part in, s/he first received a recommended product (that was generated by one of our tested algorithms, and was from one of various product domains such as "clothes," "toys," "home appliances," "foods," etc.) including its name, image, short description, and price. The user then completed a questionnaire that assessed her/his immediate feedback on this recommendation. S/he was also asked to answer several questions about her/his personal background (e.g., age, gender), and fill out a psychological curiosity quiz. As the incentive, all of the participants were placed in a lottery draw with customized presents as awards given to the winners.

3,039 users joined in our survey. We carefully checked their responses in order to filter out invalid answers. For example, if a user did not answer all of the questions, or gave the same rating to all the questions (some were asked in the reverse way), her/his response was deleted. We also removed redundant responses (by the same user) and only kept her/his first response. As a result, 2,401 users remained (1,651 females). An analysis of the users' historical behaviors over the past three months (from Oct. to Dec., 2017) showed that all of them had clicked at least one item before taking the survey (98.5% had more than 100 clicks).

In the following, we introduce the variables assessed in the user survey.

3.1.1 User Perceptions of Recommendation. Because the survey was completed on the user's mobile device, s/he would have been less patient in responding to a lengthy questionnaire consisting of many questions [41]. Therefore, to reduce the survey duration, we adopted a short version of ResQue (a widely used user-centric evaluation framework for recommender systems [36]), which, as

claimed by the authors, can provide a fast and reliable way to assess user perceptions of recommendations. Specifically, we used the suggested question for each of the four variables (see Table 1): *recommendation relevance* (Q1), *recommendation novelty* (Q2), *satisfaction* (Q8), and *purchase intention* (Q9). For recommendation diversity, because we emphasize the difference of the currently recommended item from those the user previously experienced, we included two specific questions: one is about the difference from the user's previously purchased product types (Q3: *pur_diversity*), and the other is about the difference from the system's prior recommendations (Q4: *rec_diversity*). In addition, we asked two questions particularly related to recommendation serendipity: *unexpectedness* (Q5) and *serendipity* (Q6). As mentioned before, there is a popularly used Chinese word "惊喜" that can well convey the meaning of serendipity to Chinese users, so we asked about the user's perceived serendipity directly, instead of choosing indirect statements designed to address the interpretation difficulty of the English word "serendipity" [19, 24, 37]. We also added a question about the recommendation's context compatibility [36], i.e., whether the item is recommended at the right time (Q7: *timeliness*).

Overall, nine questions were included in our survey to assess a user's perceptions of the recommendation (each responded on a 5-point Likert scale from "strongly disagree" to "strongly agree"). In Section 3.2, we present a hypothesized path model to depict their potential causal relationships.

3.1.2 Curiosity Instrument. According to the psychological theory [2, 23, 39], curiosity is an intrinsic human trait, triggered when there is a gap between the person's current knowledge level and the desired level. Because different people may have different desires for new knowledge, in recommender systems, some users may feel excited when they see an unexpected recommendation, while some may be reluctant to explore unfamiliar items [29, 47]. Thus, in our survey, we asked the participants to respond to a popular curiosity instrument, Curiosity and Exploration Inventory-II (CEI-II) [16] (an improved version of the original Curiosity and Exploration Inventory (CEI) [17]), because it was proven capable of assessing individual differences in the "general tendency to embrace novel or uncertain situations and to seek out new experiences." Specifically, it is a 10-item self-report scale embodying two factors: *Stretching* ("motivation to seek out knowledge and new experiences") and *Embracing* ("willingness to embrace the novel, uncertain, and unpredictable nature of everyday life"). Each item is rated on a 5-point Likert scale from "very slightly or not at all" to "extremely." This instrument has been shown to have acceptable internal reliability and stable validity across time. It is also short enough to be possibly completed within two minutes [5]. Another favorable consideration is that it was validated as having good psychometric properties in a Chinese context [45].

3.2 Hypothesized Path Model

Figure 1 shows our hypothesized path model covering all of the observed variables. We first linked the two major components, *unexpectedness* and *relevance*, to *serendipity*, because it has been claimed that a serendipitous recommendation should be not only unexpected (surprising), but also relevant to the user's preferences

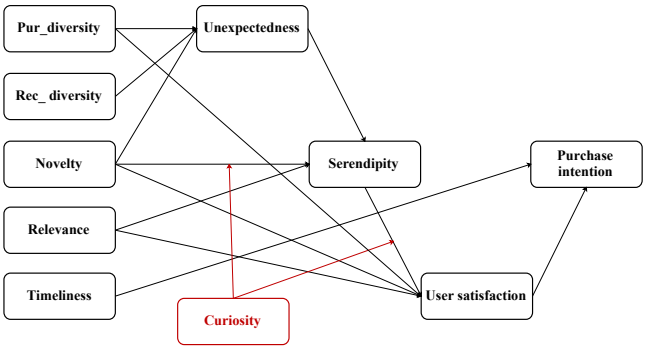


Figure 1: Hypothesized path model.

[15, 19, 24]. We were hence interested in verifying the two components' respective contributions to serendipity. In some related work, *novelty* has also been closely related to serendipity as one key component [12, 19, 28]. For example, [12] stated that a serendipitous item should be both novel (unknown to the user) and surprising. Therefore, we also linked novelty to serendipity, so as to identify to what degree a user may feel a recommendation is serendipitous when s/he perceives it as novel.

In addition, we postulated that *novelty* and *diversity* lead to *unexpectedness* [15]. That is, a novel item, or an item different from the user's previously purchased items (*pur_diversity*) or from the system's prior recommendations (*rec_diversity*), would make the user feel that it is unexpected. In this sense, unexpectedness may act as a mediator in passing some of the effect from novelty to serendipity, and also build a bridge between diversity and serendipity to reflect the diversity's indirect effect.

More importantly, we aimed to identify the factors that empirically influence *user satisfaction* with the recommendation. Although it has been recognized that accuracy alone does not always result in user satisfaction [12], few studies have verified whether and how the other beyond-accuracy factors, especially *serendipity*, have a positive effect in this regard. For instance, in a recent work [19], the effect of serendipity on satisfaction was not found to be significant. As for *diversity* and *novelty*, some user studies showed that diversity positively affects user satisfaction, whereas novelty has a negative effect [7]. However, in another work [36], novelty was demonstrated to have positively higher impact than diversity on satisfaction. We therefore expected that by collecting a larger amount of user feedback in a realistic recommending environment (e.g., mobile e-commerce in our experiment), we could empirically identify these factors' exact roles.

Finally, we postulated that the context compatibility and user satisfaction lead to higher *purchase intention* [36]. In particular, context compatibility refers to the timely compatibility (i.e., *timeliness*) of the current recommendation in our experiment. For example, although a user generally likes digital cameras, s/he may not want to buy a new one at present because s/he has already owned one. It was hence assumed that if one user is satisfied with a recommendation that also meets her/his current requirement, s/he would be more likely to purchase it.

Table 1: Assessment of user perceptions of the recommendation and descriptive statistics (of 2,348 responses after filtering out invalid records and outliers)

Subjective variable and assessment question	Mean	Std.	Median	Kolmogorov-Smirnov test	Skewness	Kurtosis
Relevance: *Q1. "The item recommended to me matches my interests."*	3.32	1.410	4.00	0.255^{***}	-0.419	-1.192
Novelty: *Q2. "The item recommended to me is novel."*	3.06	1.424	3.00	0.235^{***}	-0.146	-1.391
Pur_diversity: *Q3. "The item recommended to me is different from the types of products I bought before."*	3.39	1.215	4.00	0.221^{***}	-0.400	-0.813
Rec_diversity: *Q4. "The item recommended to me is similar to the system's prior recommendations."*(reversed)	2.93	1.302	3.00	0.206^{***}	0.214	-1.109
Unexpectedness: *Q5. "The item recommended to me is unexpected."*	3.16	1.437	3.00	0.207^{***}	-0.199	-1.337
Serendipity: *Q6. "The item recommended to me is a pleasant surprise."*	2.73	1.456	2.50	0.193^{***}	0.195	-1.400
Timeliness: *Q7. "The item recommended to me is very timely."*	3.00	1.484	3.00	0.207^{***}	-0.074	-1.450
User satisfaction: *Q8. "I am satisfied with this recommendation."*	3.21	1.140	3.00	0.210^{***}	-0.286	-0.466
Purchase intention: *Q9. "I would buy the item recommended, given the opportunity."*	2.83	1.456	3.00	0.191^{***}	0.003	-1.418
Curiosity: Curiosity and Exploration Inventory-II (CEI-II) with a 10-item self-report scale [16]	3.13	0.831	3.10	0.043^{***}	0.088	-0.402

Note: $^{***}p < 0.001$; all of the questions were accompanied by Chinese translations.

Furthermore, considering that *curiosity* is a personal trait, we incorporated it as a moderating factor into our path model. By definition, "moderation" means that this factor will moderate the effect of one variable on another variable [22]. For instance, highly curious users may be more inclined to perceive a novel item as serendipitous, because they tend to embrace uncertain situations (i.e., curiosity moderates the causal relationship *from novelty to serendipity*). Moreover, they may be more satisfied with a serendipitous item, compared to less curious users (i.e., curiosity moderates the relationship *from serendipity to satisfaction*).

3.3 Algorithms

Another objective of our user survey was to compare several recommender algorithms in terms of the users' perceived serendipity and the other subjective variables, which may help explain whether and why some particular algorithm would be preferred over others. Specifically, we implemented a popularity based approach as the baseline, and three variants of the collaborative filtering (CF) based method that are respectively tailored to highlight relevance, novelty, and serendipity of the recommendation.

(1) The baseline recommends a product with the most clicks to users (referred to as *HOT* henceforth), so it is purely popularity based without considering the target users's preferences.

(2) For the second method, we intended to strengthen a recommendation's relevance to the user's preferences. For this purpose, we revised the standard user-based CF by calculating a domain-specific similarity score $sim_d(u, v)$ between two users u and v (see Equation (1)), so that if they have often clicked the same item within the same domain (e.g., "clothes," "toys," "home appliances") and the clicking time is close, their overall similarity will be enhanced. This approach is shortened as *Rel-CF*.

$$sim_d(u,v) = \frac{\sum_{i \in I_{du} \cap I_{dv}} W_i^2 / (1 + \alpha \cdot |t_{ui} - t_{vi}|)}{\sqrt{\sum_{i \in I_{du}} W_i^2} \sqrt{\sum_{j \in I_{dv}} W_j^2}} \quad (1)$$

where I_{du} is the item set that the user u has clicked in domain d, and t_{ui} is the time stamp when u clicked i (so $|t_{ui} - t_{vi}|$ gives the time interval between u's and v's clicks on the same item i). W_i is the weight of an item i, computed via $W_i = \frac{1}{\log_2(3+q_i)}$, so it will be higher if the item has been less frequently clicked (q_i is the clicking frequency of i). The score for an unknown item for user u is predicted as:

$$score(u,i) = \sum_{d \in D} \sum_{\substack{v \in (U_i \\ \cap S_d(u))}} sim_d(u,v) * r(v,i) \quad (2)$$

where D is the set of all domains (36 domains in our experiment), $S_d(u)$ contains the top n users who are most similar to user u w.r.t. domain d according to $sim_d(u, v)$, U_i is the set of users who have clicked item i, and $r(v, i)$ is the rating score that v gave to i (that is the implicit clicking behavior in our case, i.e., 1 "clicked" and 0 "not clicked").

(3) The third method aims to be novelty oriented, by recommending an item from a category/domain outside of the user's profile (so unlikely known by the user). We adopted the item-based CF, but only calculated the similarity between two items if they belong to two different categories of a domain or two different domains (one domain can consist of multiple categories, e.g., the domain "clothes" includes the categories "T-shirt," "skirt," "sweater," etc.). This approach is shortened as *Nov-CF*.

$$sim(i,j) = \frac{\sum_{(c_i \neq c_j) \& (u \in U_i \cap U_j)} W_u^2 / (1 + \alpha \cdot |t_{ui} - t_{uj}|)}{\sqrt{\sum_{u \in U_i} W_u^2} \sqrt{\sum_{v \in U_j} W_v^2}} \quad (3)$$

where c_i is the category of i, and W_u is the weight of user u, computed via $W_u = \frac{1}{\log_2(3+q_u)}$, so if a user has clicked fewer items (i.e., q_u), her/his weight will be higher. The prediction score for an unknown item for user u is hence computed as:

$$score(u,i) = \sum_{j \in (S(i) \cap I_u)} sim(i,j) * r(u,j) \quad (4)$$

where $S(i)$ is the set of the top n items that are most similar to item i according to $sim(i, j)$, and I_u is the set of all items that have been clicked by user u.

(4) The fourth is more serendipity oriented (shortened as *Ser-CF*), because its recommendation's relevance to the target users's preferences is more strengthened on top of *Nov-CF*. Specifically, it considers the time sequence of two items clicked by a user as well as the other items clicked between them in the sequence:

$$sim_{seq}(i,j) = \sum_{(u \in U_i \cap U_j) \& (p_{uj} < p_{ui})} W_{uij} \cdot s_u(p_{ui}, p_{uj}) \quad (5)$$

where $s_u()$ gives the sum of similarities between any two adjacent items positioned from p_{uj} to p_{ui} (p_{uj} is the position of item j within the time sequence of items user u has clicked, and $p_{uj} < p_{ui}$ indicating that j was clicked before i). $W_{uij} = \frac{\sum 1/|I_u \cap I_v|}{|U_i \cap U_j| - 1}$ denoting the distance of u from other users who also clicked both items i and j (i.e., $v \in (U_i \cap U_j) \backslash \{u\}$), so that if the whole set of items s/he clicked (I_u) is largely different from other users', her/his weight will be higher.

$$s_u(p_{ui}, p_{uj}) = \sum_{\substack{(p_{uj} \le p_{um}, p_{un} \le p_{ui}) \\ \&(p_{un} = p_{um}+1)}} sim(m, n) \quad (6)$$

where $sim(m, n)$ is computed via Equation (3). The prediction score is thus computed as:

$$score(u, i) = \sum_{j \in (S(i) \cap I_u)} sim_{seq}(i, j) * r(u, j) \quad (7)$$

Each participant was randomly assigned one of the four algorithms' recommendations, which basically adheres to the between-subjects experimental design for simulating real-world experiences [11]. The numbers of users assigned to the four algorithms are respectively: 570 for HOT, 596 for Rel-CF, 589 for Nov-CF, and 593 for Ser-CF (out of 2,348 users after filtering out invalid records and outliers).

4 RESULTS ANALYSIS

We chose IBM SPSS Amos to run the path analysis [4], because it allows us to test the causal relations among the nine observed variables (see Table 1). We first checked whether our data meet its assumption of multivariate normality. The results show the multivariate kurtosis value is 7.749 (Mardia's coefficient), with a critical ratio (c.r.) of 5.798, which is greater than the desired level 1.96 [9]. Therefore, we deleted some obvious outliers with very large Malanobis d-squared distances (i.e., improbably far from the centroid). In consequence, there were 2,348 users' responses left (1,621 females), which were found to be normal by a Mardia's coefficient of 1.440 (c.r. = 1.904). We then conducted normality testing for each variable. The p values of the Kolmogorov-Smirnov test (suitable for sample size greater than 2,000 [25]) are all less than 0.001 (see Table 1), showing that the null hypothesis of normal distribution is rejected for each variable (including curiosity). Therefore, we chose non-parametric tests that do not assume normality for the following correlation and comparison analyses. But we still used Amos to test our path model, because the multivariate distribution of the observed variables is normal, and each variable's univariate skewnesses and kurtosis are in the acceptable ranges of -1.0 to 1.0 and -2.0 to 0 respectively (so not likely to inflate the Chi-squared statistics of path analysis according to [31]).

In addition, we checked whether there are high inter-correlations among those variables, because such multicollinearity phenomenon may cause disturbance and unreliability in the regression analysis [8]. For this purpose, we calculated the Spearman's correlation coefficients[2], which are all below 0.80 suggesting that there is no serious problem with multicollinearity (we later performed collinearity diagnostics for each outcome variable for further confirmation).

[2]Spearman's correlation is a non-parametric test used to measure the degree of association between two variables.

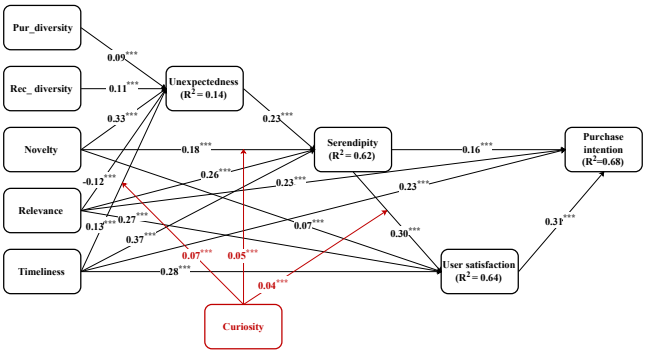


Figure 2: Path model fit. The value associated with each path is the standardized regression coefficient via bootstrap estimates, and the interaction term involving curiosity is standardized for moderation analysis (*$p < 0.001$, **$p < 0.01$, and *$p < 0.05$).**

4.1 Path Model Validation

We then tested our hypothesized path model (see Figure 1). To obtain unbiased estimates given univariate non-normalities, we performed bootstrapping (on 500 bootstrap samples at 95% biased-corrected confidence intervals) [4]. Figure 2 shows the model after the suggested modifications[3], which obtains superior goodness of fit: $\chi^2 = 8.608$ ($df = 7$, $p = 0.282$), $CFI = 1.000$, $AGFI = 0.995$, $RMSEA = 0.010$, all surpassing the suggested values of these model fit indices [13][4]. The measurement portion of the model is also good, with all of the R^2 estimates being larger than 0.10, and thus appropriate and informative to examine the significances of the paths associated with the outcome variables [36]. More notably, except that the R^2 estimate for *unexpectedness* is 0.14, the other R^2 values all exceed 0.60, indicating that the associated predictors can account for a large proportion of the variance in the corresponding outcome variable.

In more detail, we found there are significant causal relationships from *novelty*, *pur_diversity*, *rec_diversity*, *relevance*, and *timeliness* to *unexpectedness*, which implies that if a user perceives the recommended item as novel ($\beta = 0.327$ [5], $p = 0.002$), different from her/his previously purchased product types ($\beta = 0.095$, $p = 0.006$), different from the system's prior recommendations ($\beta = 0.108$, $p = 0.008$), not relevant to her/his preferences ($\beta = -0.119$, $p = 0.003$), or timely ($\beta = 0.134$, $p = 0.004$), s/he will be likely to perceive the item as unexpected. As a whole, all of these predictors account for 14% of the variance in unexpectedness.

Regarding *serendipity*, there are four significant predictors (in the descending order of influence): *timeliness* ($\beta = 0.369$, $p = 0.009$), *relevance* ($\beta = 0.264$, $p = 0.004$), *unexpectedness* ($\beta = 0.234$, $p = 0.004$), and *novelty* ($\beta = 0.181$, $p = 0.001$), which altogether

[3]We started with a full model and then went through model trimming by deleting one non-significant path at a time.

[4]χ^2, i.e., the chi-square value, should better be with probability $p \ge 0.05$ to indicate absolute fit; the cutoff values of the other fit indices CFI (Comparative Fit Index), AGFI (Adjusted Goodness of Fit), and RMSEA (Root Mean Square Error of Approximation) are respectively ≥ 0.90, ≥ 0.90, and ≤ 0.05 [13].

[5]This is the bootstrap estimate of the standardized regression weight (the same to the other β values).

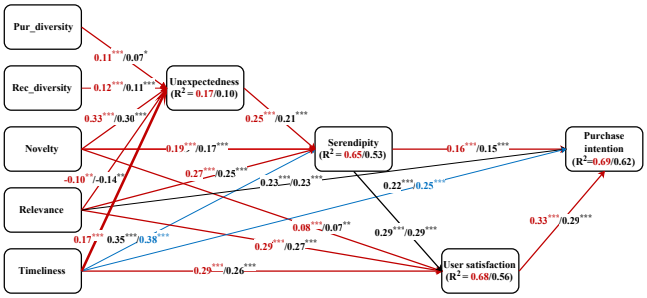


Figure 3: Multi-group path model analysis between *high* and *low* curiosity groups. Red highlights higher β coefficients and R^2 estimates in the *high* curiosity group, and blue highlights higher values in the *low* curiosity group.

account for 62% of the variance in serendipity, inferring that if a recommendation is timely, relevant, unexpected, and novel, the user will very likely perceive it as serendipitous. Because there are also direct paths from novelty, relevance, and timeliness to unexpectedness, it suggests that some of their effects on serendipity are partially mediated by unexpectedness.

Serendipity, allied with *timeliness*, *relevance*, and *novelty*, can be further used to predict *user satisfaction*, with standardized regression weights of $\beta = 0.305$ ($p = 0.006$), $\beta = 0.275$ ($p = 0.003$), $\beta = 0.273$ ($p = 0.003$), and $\beta = 0.075$ ($p = 0.003$) respectively. The R^2 value associated with user satisfaction is 0.65, indicating that 65% of its variance can be accounted for by the set of these four variables, among which the effects of serendipity, timeliness, and relevance are much higher than that of novelty. Here we also see the mediated paths passing through serendipity, suggesting that serendipity has a certain mediating effect on the causal relationships from timeliness, relevance, and novelty to user satisfaction.

The last outcome variable is *purchase intention*, which is significantly affected by *user satisfaction* ($\beta = 0.314$, $p = 0.004$), *timeliness* ($\beta = 0.232$, $p = 0.002$), *relevance* ($\beta = 0.228$, $p = 0.002$), and *serendipity* ($\beta = 0.161$, $p = 0.005$), with 68% of the variance explained by them. In combination with the results above, this suggests that timeliness and relevance can directly affect users' purchase intention, while serendipity and satisfaction mainly play mediating roles.

Collinearity diagnostics were then conducted for all of the four outcome variables (i.e., *unexpectedness*, *Serendipity*, *user satisfaction*, and *purchase intention*). Because the values of Tolerance and VIF ($Tolerance > 0.34$ and $VIF < 3$) all meet the desired standards[6], we can confirm there is no multicollinearity issue in our path analysis.

4.2 Moderating Effect of User Curiosity

Next, we conducted an in-depth investigation on the role of *curiosity* in affecting user perceptions of the recommendation, for which we first checked the internal reliability of our used curiosity instrument (10-item scale). Cronbach's alpha value is 0.876 and the item-total correlations are all between 0.381 and 0.701, which suggest that the 10 items all reliably measure the same variable [35]. We also performed principal components analysis (PCA), by which two factors were extracted: questions 1-3, 5 of higher loadings on one factor, and questions 4, 6-10 of higher loadings on the other factor (all exceeding the acceptable level of 0.5 [35]). However, because this 2-factor structure does not fully replicate the original Stretching/Embracing structure[7] [16], we used the 1-factor structure in our analysis (i.e., taking the 10 items together to represent *curiosity*).

We then performed two types of analysis. First, multi-group analysis enabled us to compare observations across two population groups (i.e., low curiosity vs. high curiosity). To divide all users into the two groups, we used the median split method [14], resulting in 1,132 users in the low curiosity group (< median 3.1 that is the cutoff point) and 1,216 in the high curiosity group (>= 3.1). Then, by imposing structural weights constraint on the path model, we found the Chi-square difference statistic is significant ($\chi^2 = 45.542$ with 17 *df*, $p < 0.001$), which implies significant inequalities of path coefficients across the two groups[8] (see Figure 3).

In particular, we detected three major differences between the two groups: 1. For the low curiosity group, there is no significant path relationship from timeliness to unexpectedness; 2. the regression weights of paths originating from *novelty*, *unexpectedness*, and *serendipity* are mostly higher in the high curiosity group; 3. the R^2 estimates for the outcome variables are all higher in the high curiosity group (for example, 0.65 for *serendipity* in the high curiosity group, vs. 0.53 in the low curiosity group), indicating that the associated predictor variables can be more accurate to estimate the corresponding outcome variable.

Second, we performed the moderation analysis, aiming to identify whether curiosity has actual moderating effects on the causal relationships in our path model[9]. The results show that it primarily acts as a moderator for three relationships (see Figure 2): *from timeliness to unexpectedness* (the standardized regression weight (s.r.w.) of the interaction term *curiosity×timeliness* is 0.073, $p = 0.006$), *from novelty to serendipity* (the s.r.w. of *curiosity×novelty* is 0.048, $p = 0.004$), and *from serendipity to satisfaction* (the s.r.w. of *curiosity×serendipity* is 0.041, $p = 0.005$). In other words, it suggests that more curious users are more likely to perceive a timely recommendation as unexpected, a novel recommendation as serendipitous, and be satisfied with the serendipitous item, while these relationships are not so strong for less curious people. The results hence validate the observations of the above multi-group analysis, implying the prominent role of curiosity in influencing users' perceived novelty, serendipity, and satisfaction with the recommendation.

[6] To be a problem, Tolerance has to approach 0 and VIF has to approach 10 [32].

[7] Similar phenomenon was found in [45] that was also tested in a Chinese context, so more studies might be needed to verify this 2-factor structure under different cultural backgrounds.

[8] We also conducted multi-group analyses with respect to age (young vs. old) and gender (female vs. male), but no significance was found ($\chi^2 = 17.059$ with 17 *df*, $p = 0.450$ w.r.t. age; and $\chi^2 = 16.943$ with 17 *df*, $p = 0.458$ w.r.t. gender).

[9] The path model still fits well after being incorporated with the curiosity and interaction terms ($\chi^2 = 25.466$ ($df = 18$, $p = 0.113$), $CFI = 0.999$, $AGFI = 0.992$, $RMSEA = 0.013$).

Table 2: Results of algorithm comparison by Kruskal-Wallis 1-way ANOVA test (the superscript indicates that the corresponding algorithm significantly outperforms the numbered one, with $p < 0.05$ adjusted by the Bonferroni correction)

Subjective variable	Mean rank				Sig.
	HOT[1]	Rel-CF[2]	Nov-CF[3]	Ser-CF[4]	
Relevance	1047.61	1168.76[1]	1130.39	**1346.05**[1,2,3]	0.000
Novelty	1079.22	1148.04	1182.39[1]	**1284.84**[1,2,3]	0.000
Pur_diversity	1201.83	1164.10	1182.21	1151.03	0.578
Rec_diversity	**1391.74**[2,3,4]	1203.96[3,4]	1092.20	1017.82	0.000
Unexpectedness	**1300.69**[2,3,4]	1178.18	1101.88	1121.64	0.000
Serendipity	1062.11	1142.97	1182.73[1]	**1306.05**[1,2,3]	0.000
Timeliness	1002.74	1110.57[1]	1182.43[1]	**1395.98**[1,2,3]	0.000
User satisfaction	1013.12	1154.61[1]	1184.08[1]	**1340.10**[1,2,3]	0.000
Purchase intention	1033.38	1142.73[1]	1162.42[1]	**1354.07**[1,2,3]	0.000

4.3 Algorithm Comparison

The third objective of this experiment was to compare the four recommender algorithms (see Section 3.3) in terms of user perceptions. We used the Kruskal-Wallis 1-way ANOVA test[10] to handle the non-normally distributed dependent variables. As above mentioned, there were 570, 596, 589, and 593 users respectively assigned to the HOT, Rel-CF, Nov-CF, and Ser-CF algorithms. Table 2 reports the results. It can be seen that there is very strong evidence of differences among the four algorithms, in terms of *novelty*, *relevance*, *rec_diversity*, *timeliness*, *unexpectedness*, *serendipity*, *satisfaction*, and *purchase intention*. Moreover, the mean rank of Ser-CF seems to perform the best compared to the other three, except unexpectedness and rec_diversity. It is interesting to see that the HOT algorithm was perceived the most unexpected and the most different from the system's prior recommendations, but it was the worst as for most of the other variables including user satisfaction and purchase intention. Because this method only returns the most popular item without taking into account the user's preferences, it implies that unexpectedness can reflect unpleasant surprise in this condition.

We further ran the Mann Whitney Wilcoxon test[11] for the pairwise comparisons. The results show strong evidence ($p < 0.05$ adjusted by the Bonferroni correction for multiple tests) of the differences between Ser-CF and the other three algorithms. Specifically, Ser-CF is significantly better than HOT, Rel-CF, and Nov-CF, in respect of *novelty*, *relevance*, *timeliness*, *serendipity*, *satisfaction*, and *purchase intention*. The comparison between Nov-CF and HOT shows that the former obtains significantly higher mean ranks regarding *novelty*, timeliness, serendipity, satisfaction, and purchase intention; while the comparison between Rel-CF and HOT reveals the former is particularly better at *relevance* and timeliness, which also results in increased satisfaction and purchase intention. However, there are no significant differences between Rel-CF and Nov-CF as for most of the variables. One more interesting observation is that unexpectedness and rec_diversity are significantly higher for HOT relative to the other three algorithms.

[10] A nonparametric test used to compare the mean ranks of more than two independent groups, when the dependent variable is ordinal or not normally distributed [27].
[11] It is also a nonparametric test for comparing two independent groups.

All of the results suggest that the Ser-CF algorithm is capable of accommodating both *novelty* and *relevance* to more effectively increase users' perceived recommendation *serendipity* and *satisfaction*, compared to Rel-CF that is more relevance focused, and Nov-CF that is more novelty focused. In addition, we found that people with a high curiosity level were more satisfied with Ser-CF, because its differences from Nov-CF and Rel-CF are significant in the high curiosity group regarding *user satisfaction* (Ser-CF vs. Nov-CF: 686.59 mean rank vs. 611.69, $p = 0.032$; Ser-CF vs. Rel-CF: 686.59 vs. 595.45, $p = 0.004$), but not significant in the low curiosity group (Ser-CF vs. Nov-CF: 630.68 vs. 580.93, $p = 0.365$; Ser-CF vs. Rel-CF: 630.68 vs. 565.39, $p = 0.082$). These results imply the effect of curiosity on users' evaluation of a particular recommender algorithm.

We also recorded all participants' post-survey behavior to see whether they revisited the recommended item after the survey[12]. A Chi-square test reveals a significant association between the algorithms and users' revisiting behavior ($\chi^2 = 188.027, p < 0.001$). The values of Phi and Cramer's V again indicate that the strength of this association is strong (both are 0.283, $p < 0.001$). More specifically, more Ser-CF users (over 17%) clicked the recommended item after the survey, whereas fewer users who used algorithms Nov-CF (6.1%), Rel-CF (1.0%), and HOT (0%) revisited their recommendations. However, because the revisiting behavior did not occur immediately after our experiment (e.g., one user revisited the item 6 hours after taking the survey), we cannot claim it was necessarily caused by the recommendation. Therefore, in the future, we will conduct more studies to measure the correlation between users' perceptions of a recommendation and their actual behavior (e.g., clicking and purchasing).

5 DISCUSSION

In this section, we summarize the major findings of our experiment and their practical implications, and discuss the limitations of our work.

5.1 Serendipity and User Satisfaction

One major finding is that serendipity significantly positively affects user satisfaction and purchase intention. Its contribution is largely higher than that of novelty, more direct relative to diversity's, and comparable to those of relevance and timeliness. As a result, more than 60% of the variance in the two outcome variables (satisfaction and purchase intention) can be explained by the associated predictors.

Moreover, as relevance, timeliness, and novelty can directly affect serendipity, it discloses serendipity's mediating role in transmitting part of their effects on user satisfaction. However, two diversity related variables (i.e., pur_diversity and rec_diversity) are not directly related to serendipity, but have a positive influence on unexpectedness. This suggests that if the recommendation is different from the user's previously purchased product types or the system's prior recommendations, s/he may first feel surprised, which will then combine with the other factors (i.e., relevance, timeliness, novelty) to induce serendipity (altogether accounting for over 60% of the variance in serendipity).

[12] We measured a user's revisiting behavior during the day s/he took part in the survey.

Compared with the related work that primarily takes unexpectedness, novelty, and relevance as principal components of serendipity [6, 20, 28], we have two new observations. First, ***timeliness*** can be more important than unexpectedness, novelty, and relevance, in terms of affecting serendipity. However, this factor is often neglected in the related literature. For example, in [36], this variable was eliminated due to low correlations with other variables. In our study, we find it has a significantly positive impact on multiple perception variables, such as serendipity, satisfaction, and purchase intention. One possible explanation is that our experiment was conducted in a mobile e-commerce environment, so whether the recommendation was given at the best possible time can be regarded as critically important by users.

Second, we find that ***unexpectedness*** and ***novelty*** take different roles in influencing users' perceived serendipity. Novelty acts more positively as it has a significantly direct relationship with serendipity and user satisfaction, whereas unexpectedness is only related to serendipity. In addition, unexpectedness may reflect unpleasant surprise in some conditions (e.g., when the recommendation is not relevant to the user's preferences). Another finding is that although unexpectedness is affected by several variables, only 14% of its variance is explained by them all, suggesting that there would be other unexplored variables greatly impacting it.

For novelty, our results are basically consistent with those of [36] that also indicated a positive effect of novelty on user satisfaction, but contradict the findings of [7] that showed its negative influence. As discussed in [15], this may be caused by the studied product domain (e-commerce products vs. movies in [7]), or the formulation of novelty-related question (positive tone such as "*The item recommended to me is novel*" vs. negative tone such as "*Which list has more movies you would not have thought to consider?*" in [7]). More studies are thus needed to verify these confounding effects.

Implication. Because the experiment shows there are four variables, namely *timeliness*, *relevance*, *unexpectedness*, and *novelty*, are significantly related to serendipity, we may improve the existing offline metrics of measuring serendipity [1, 10], in a more precise way. For instance, we could add the component *timeliness*, as well as more clearly distinguishing the roles of *unexpectedness* and *novelty* in the formulation. It would also be meaningful to introduce some indirectly related factors (such as *pur_diversity*), and other potential factors that may lead to unexpectedness.

5.2 The Role of User Curiosity

The second finding is about the exact role of user curiosity in moderating the effect of one perception variable on another. The results of moderation analysis show that curiosity can not only strengthen the positive effect of *novelty on serendipity*, but also that of *serendipity on satisfaction*. It hence implies that a more curious person will be more likely to perceive a novel recommendation as serendipitous, and be more satisfied with the serendipitous item.

Furthermore, we find that the path models are significantly different between the high and low curiosity groups of users through the multi-group analysis. In particular, *novelty*, *unexpectedness*, and *serendipity* behave more actively in positively affecting the other variables in the high curiosity group. The accuracy of predicting serendipity (and furthermore user satisfaction and purchase intention) can also be likely higher in the high curiosity group. Overall, these results verify our hypothesis about the effect of curiosity on users' appetite for serendipitous recommendations.

Implication. Recently, some studies have attempted to incorporate curiosity into recommendation generation [24, 29, 47], but their methods were mainly based on some assumption without empirical validation of the curiosity's actual effect. The results of our study could be constructive for the related work to further strengthen the role of curiosity in their recommender algorithms. For instance, given that curiosity significantly moderates the causal relationships from novelty to serendipity and from serendipity to satisfaction, a personalized recommendation strategy might be developed to dynamically adjust the serendipity degree according to the target user's curiosity value, instead of being simply maximized for everyone.

5.3 User Evaluation of Recommender Algorithms

This experiment also explains why a particular recommender approach is preferred to others. We compared four algorithms: HOT, Rel-CF, Nov-CF, and Ser-CF, which are respectively popularity, relevance, novelty, and serendipity oriented. The results show that Ser-CF is the best in terms of novelty, relevance, timeliness, serendipity, user satisfaction, and purchase intention; Nov-CF and Rel-CF lie in the middle; and HOT provides the most unexpected recommendations, but performs worst regarding most of the other variables. It thus infers that a serendipity-oriented algorithm might be more satisfying and stimulating users' intention to purchase its recommended items. The observations related to HOT suggest that unexpectedness can reflect users' unpleasant surprise if the recommendation is purely popularity based without taking into account the user's preferences.

Implication. The way we evaluated recommender algorithms might be referential to related researchers for assessing user perceptions of their algorithms. On the other hand, since the advantage of the serendipity-oriented algorithm (Ser-CF) is not very obvious among less curious users, it would be meaningful to have a personalized serendipity approach as we discussed above. Recently, a personality-based framework was proposed to adapt recommendations' diversity to individual users' intrinsic needs [44]. We could extend it to accommodate serendipity, so as to achieve optimal user satisfaction with the recommender system.

5.4 Limitations of Our Work

Our work has several limitations. First, the users who took part in our experiment mainly represent the population that has used mobile apps to buy products [43]. The results might have been different if it was conducted in another setting (e.g., traditional e-commerce) or even other domains (such as social media and tourism). Therefore, we are interested in performing a cross-platform validation in the future, so as to identify any platform-specific features. Second, a cross-cultural study would also be necessary, because our survey mainly involved Chinese users. It will be interesting to see whether similar findings would occur among users in other countries, or there would be any cultural differences. Third, we used the short

version of ResQue questionnaire [36] for the sake of mobile survey, which limited our statistical approach to path analysis that does not contain latent variables to account for measurement error [38]. It would therefore be better to verify the findings by using the complete version of ResQue [36], and the other user-centric evaluation frameworks [18] (given that different formulations of the survey questions may engender different user responses [15]). Fourth, we did not test the state-of-the-art algorithms, but emphasized on tailoring the classical CF methods to strengthen the recommendation's relevance, novelty, and serendipity respectively. For the future work, it will be useful to identify whether the results could be generalizable to other algorithms, especially those that are serendipity oriented [24, 33, 46].

6 CONCLUSIONS

Different from related work on recommendation serendipity that either focused on algorithm development or conducted small-scale user evaluations, we have been engaged in empirically revealing the causal relationships between serendipity and various user perceptions via a large scale online study performed on an industrial mobile e-commerce platform. The results show that there are four major components of serendipity, namely *timeliness*, *relevance*, *unexpectedness*, and *novelty*. Furthermore, *serendipity* behaves more effectively than the other two beyond-accuracy objectives (novelty and diversity) in affecting *user satisfaction* and *purchase intention*.

Additionally, this study verifies the moderating effects of *user curiosity* on the causal relationships from novelty to serendipity and from serendipity to satisfaction. This implies that more curious users will be more likely to perceive a novel item as serendipitous and be satisfied with a serendipitous recommendation. Last but not least, the user evaluation of four recommender algorithms indicates that a serendipity-oriented method (*Ser-CF*) performs the best. Inspired by those findings, we believe that a personalized serendipity approach based on curiosity would be potentially able to optimize user satisfaction with the recommender system.

ACKNOWLEDGMENTS

This work was partially supported by Hong Kong Research Grants Council (RGC) (project RGC/HKBU12200415).

REFERENCES

[1] Panagiotis Adamopoulos and Alexander Tuzhilin. 2014. On Unexpectedness in Recommender Systems: Or How to Better Expect the Unexpected. *ACM Trans. Intell. Syst. Technol.* 5, 4, Article 54 (Dec. 2014), 32 pages. https://doi.org/10.1145/2559952

[2] Daniel Ellis Berlyne. 1960. *Conflict, Arousal and Curiosity.* McGraw-Hill.

[3] Toine Bogers and Lennart Björneborn. 2013. Micro-serendipity: Meaningful Coincidences in Everyday Life Shared on Twitter. In *Proceedings of the iConference 2013.* 196–208.

[4] Barbara M. Byrne. 2016. *Structural Equation Modeling With AMOS: Basic Concepts, Applications, and Programming* (3rd ed.). Routledge.

[5] Curiosity and Exploration Inventory (CEI-II) [n. d.]. http://www.midss.org/content/curiosity-and-exploration-inventory-cei-ii.

[6] Marco de Gemmis, Pasquale Lops, Giovanni Semeraro, and Cataldo Musto. 2015. An Investigation on the Serendipity Problem in Recommender Systems. *Inf. Process. Manage.* 51, 5 (Sept. 2015), 695–717. https://doi.org/10.1016/j.ipm.2015.06.008

[7] Michael D. Ekstrand, F. Maxwell Harper, Martijn C. Willemsen, and Joseph A. Konstan. 2014. User Perception of Differences in Recommender Algorithms. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14).* ACM, New York, NY, USA, 161–168. https://doi.org/10.1145/2645710.2645737

[8] Donald E. Farrar and Robert R. Glauber. 1967. Multicollinearity in Regression Analysis: The Problem Revisited. *The Review of Economics and Statistics* 49, 1 (1967), 92–107.

[9] Shengyi Gao, Patricia L. Mokhtarian, and Robert A. Johnston. 2008. Nonnormality of Data in Structural Equation Models. *Transportation Research Record* 2082, 1 (2008), 116–124. https://doi.org/10.3141/2082-14

[10] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. 2010. Beyond Accuracy: Evaluating Recommender Systems by Coverage and Serendipity. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys '10).* ACM, New York, NY, USA, 257–260. https://doi.org/10.1145/1864708.1864761

[11] Asela Gunawardana and Guy Shani. 2015. Evaluating Recommendation Systems. In *Recommender Systems Handbook.* Springer US, 265–308. https://doi.org/10.1007/978-1-4899-7637-6_8

[12] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. 2004. Evaluating Collaborative Filtering Recommender Systems. *ACM Trans. Inf. Syst.* 22, 1 (Jan. 2004), 5–53. https://doi.org/10.1145/963770.963772

[13] Daire Hooper, Joseph Coughlan, and Michael Mullen. 2008. Structural Equation Modelling: Guidelines for Determining Model Fit. *Electronic Journal of Business Research Methods* 6, 1 (2008), 53–60.

[14] Dawn Iacobucci, Steven S. Posavac, Frank R. Kardes, Matthew J. Schneider, and Deidre L. Popovich. 2015. The Median Split: Robust, Refined, and Revived. *Journal of Consumer Psychology* 25, 4 (2015), 690–704. https://doi.org/10.1016/j.jcps.2015.06.014

[15] Marius Kaminskas and Derek Bridge. 2016. Diversity, Serendipity, Novelty, and Coverage: A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems. *ACM Trans. Interact. Intell. Syst.* 7, 1, Article 2 (Dec. 2016), 42 pages. https://doi.org/10.1145/2926720

[16] Todd B. Kashdan, Matthew W. Gallagher, Paul J. Silvia, Beate P. Winterstein, William E. Breen, Daniel Terhar, and Michael F. Steger. 2009. The Curiosity and Exploration Inventory-II: Development, Factor Structure, and Psychometrics. *Journal of Research in Personality* 43, 6 (2009), 987–998. https://doi.org/10.1016/j.jrp.2009.04.011

[17] Todd B. Kashdan, Paul Rose, and Frank D. Fincham. 2004. Curiosity and Exploration: Facilitating Positive Subjective Experiences and Personal Growth Opportunities. *Journal of Personality Assessment* 82, 3 (2004), 291–305. https://doi.org/10.1207/s15327752jpa8203_05

[18] Bart P. Knijnenburg, Martijn C. Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. 2012. Explaining the User Experience of Recommender Systems. *User Modeling and User-Adapted Interaction* 22, 4-5 (Oct. 2012), 441–504. https://doi.org/10.1007/s11257-011-9118-4

[19] Denis Kotkov, Joseph A. Konstan, Qian Zhao, and Jari Veijalainen. 2018. Investigating Serendipity in Recommender Systems Based on Real User Feedback. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing (SAC '18).* ACM, New York, NY, USA, 1341–1350. https://doi.org/10.1145/3167132.3167276

[20] Denis Kotkov, Shuaiqiang Wang, and Jari Veijalainen. 2016. A Survey of Serendipity in Recommender Systems. *Knowledge-Based Systems* 111 (Nov. 2016), 180–192. https://doi.org/10.1016/j.knosys.2016.08.014

[21] Neal Lathia, Stephen Hailes, Licia Capra, and Xavier Amatriain. 2010. Temporal Diversity in Recommender Systems. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '10).* ACM, New York, NY, USA, 210–217. https://doi.org/10.1145/1835449.1835486

[22] Todd D. Little, Noel A. Card, James A. Bovaird, Kristopher J. Preacher, and Christian S. Crandall. 2012. Structural Equation Modeling of Mediation and Moderation with Contextual Factors. In *Modeling Contextual Effects in Longitudinal Studies.* Taylor and Francis, 207–230. https://doi.org/10.4324/9780203936825

[23] George Loewenstein. 1994. The Psychology of Curiosity: A Review and Reinterpretation. *Psychological Bulletin* (1994), 75–98.

[24] Valentina Maccatrozzo, Manon Terstall, Lora Aroyo, and Guus Schreiber. 2017. SIRUP: Serendipity In Recommendations via User Perceptions. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces (IUI '17).* ACM, New York, NY, USA, 35–44. https://doi.org/10.1145/3025171.3025185

[25] Frank J. Massey. 1951. The Kolmogorov-Smirnov Test for Goodness of Fit. *J. Amer. Statist. Assoc.* 46, 253 (1951), 68–78. https://doi.org/10.1080/01621459.1951.10500769

[26] Christian Matt, Thomas Hess, Alexander Benlian, and Christian Weiß. 2014. Escaping from the Filter Bubble? The Effects of Novelty and Serendipity on Users' Evaluations of Online Recommendations. (2014). https://EconPapers.repec.org/RePEc:dar:wpaper:66193

[27] John H. McDonald. 2014. *Handbook of Biological Statistics, Third Edition.* Sparky House Publishing.

[28] Sean M. McNee, John Riedl, and Joseph A. Konstan. 2006. Being Accurate is Not Enough: How Accuracy Metrics Have Hurt Recommender Systems. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems (CHI EA '06).* ACM, New York, NY, USA, 1097–1101. https://doi.org/10.1145/1125451.1125659

[29] Alan Menk, Laura Sebastia, and Rebeca Ferreira. 2017. CURUMIM: A Serendipitous Recommender System for Tourism Based on Human Curiosity. In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI).* 788–795. https://doi.org/10.1109/ICTAI.2017.00124

[30] Tomoko Murakami, Koichiro Mori, and Ryohei Orihara. 2008. Metrics for Evaluating the Serendipity of Recommendation Lists. In *New Frontiers in Artificial Intelligence*. Springer Berlin Heidelberg, 40–46.

[31] Bengt Muthén and David Kaplan. 1985. A Comparison of Methodologies for the Factor Analysis of Non-Normal Likert Variables. *Brit. J. Math. Statist. Psych.* 38, 1 (1985), 171–189.

[32] Robert M. O'Brien. 2007. A Caution Regarding Rules of Thumb for Variance Inflation Factors. *Quality & Quantity* 41, 5 (01 Oct 2007), 673–690. https://doi.org/10.1007/s11135-006-9018-6

[33] Kenta Oku and Fumio Hattori. 2012. User Evaluation of Fusion-Based Approach for Serendipity-Oriented Recommender System. In *Proceedings of the Workshop on Recommendation Utility Evaluation: Beyond RMSE (RUE 2012)*. 39–44.

[34] Eli Pariser. 2011. *The Filter Bubble: What the Internet Is Hiding from You.* Penguin UK.

[35] Robert A. Peterson. 1994. A Meta-Analysis of Cronbach's Coefficient Alpha. *Journal of Consumer Research* 21, 2 (1994), 381–391.

[36] Pearl Pu, Li Chen, and Rong Hu. 2011. A User-centric Evaluation Framework for Recommender Systems. In *Proceedings of the 5th ACM Conference on Recommender Systems (RecSys '11)*. ACM, New York, NY, USA, 157–164. https://doi.org/10.1145/2043932.2043962

[37] Alan Said, Ben Fields, Brijnesh J. Jain, and Sahin Albayrak. 2013. User-centric Evaluation of a K-furthest Neighbor Collaborative Filtering Recommender Algorithm. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW '13)*. ACM, New York, NY, USA, 1399–1408. https://doi.org/10.1145/2441776.2441933

[38] Randall E. Schumacker and Richard G. Lomax. 2015. *A Beginner's Guide to Structural Equation Modeling* (4th ed.). Routledge.

[39] Paul J. Silvia. 2008. Interest–The Curious Emotion. *Current Directions in Psychological Science* 17, 1 (02 2008), 57–60.

[40] Kirsten Swearingen and Rashmi Sinha. 2001. Beyond Algorithms: An HCI Perspective on Recommender Systems. In *ACM SIGIR. Workshop on Recommender Systems*, Vol. 13. Citeseer, 1–11.

[41] Paula Vicente, Elizabeth Reis, and Maria Santos. 2009. Using Mobile Phones for Survey Research: A Comparison with Fixed Phones. *International Journal of Market Research* 51, 5 (2009), 613–633. https://doi.org/10.2501/S1470785309200852

[42] Horace Walpole. 1960. To Mann, Monday 18 January 1754. In *Horace Walpole's Correspondence*. Yale University Press, 407–411.

[43] Jen-Her Wu and Shu-Ching Wang. 2005. What Drives Mobile Commerce? *Information and Management* 42, 5 (July 2005), 719–729. https://doi.org/10.1016/j.im.2004.07.001

[44] Wen Wu, Li Chen, and Yu Zhao. 2018. Personalizing Recommendation Diversity Based on User Personality. *User Modeling and User-Adapted Interaction* 28, 3 (01 Aug 2018), 237–276. https://doi.org/10.1007/s11257-018-9205-x

[45] Shengquan Ye, Ting Kin Ng, Kin Hang Yim, and Jun Wang. 2015. Validation of the Curiosity and Exploration Inventory-II (CEI-II) Among Chinese University Students in Hong Kong. *Journal of Personality Assessment* 97, 4 (2015), 403–410. https://doi.org/10.1080/00223891.2015.1013546 PMID: 25774779.

[46] Yuan Cao Zhang, Diarmuid Ó Séaghdha, Daniele Quercia, and Tamas Jambor. 2012. Auralist: Introducing Serendipity into Music Recommendation. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM '12)*. ACM, New York, NY, USA, 13–22. https://doi.org/10.1145/2124295.2124300

[47] Pengfei Zhao and Dik Lun Lee. 2016. How Much Novelty is Relevant?: It Depends on Your Curiosity. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. ACM, New York, NY, USA, 315–324. https://doi.org/10.1145/2911451.2911488

[48] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. 2005. Improving Recommendation Lists Through Topic Diversification. In *Proceedings of the 14th International Conference on World Wide Web (WWW '05)*. ACM, New York, NY, USA, 22–32. https://doi.org/10.1145/1060745.1060754

Emoji-Powered Representation Learning for Cross-Lingual Sentiment Classification

Zhenpeng Chen
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
czp@pku.edu.cn

Sheng Shen
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
University of California, Berkeley, USA
sheng.s@berkeley.edu

Ziniu Hu
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
University of California, Los Angeles, USA
bull@cs.ucla.edu

Xuan Lu
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
luxuan@pku.edu.cn

Qiaozhu Mei
School of Information, University of Michigan, Ann Arbor, USA
qmei@umich.edu

Xuanzhe Liu*
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
xzl@pku.edu.cn

ABSTRACT

Sentiment classification typically relies on a large amount of labeled data. In practice, the availability of labels is highly imbalanced among different languages, e.g., more English texts are labeled than texts in any other languages, which creates a considerable inequality in the quality of related information services received by users speaking different languages. To tackle this problem, cross-lingual sentiment classification approaches aim to transfer knowledge learned from one language that has abundant labeled examples (i.e., the source language, usually English) to another language with fewer labels (i.e., the target language). The source and the target languages are usually bridged through off-the-shelf machine translation tools. Through such a channel, cross-language sentiment patterns can be successfully learned from English and transferred into the target languages. This approach, however, often fails to capture sentiment knowledge specific to the target language, and thus compromises the accuracy of the downstream classification task. In this paper, we employ emojis, which are widely available in many languages, as a new channel to learn both the cross-language and the language-specific sentiment patterns. We propose a novel representation learning method that uses emoji prediction as an instrument to learn respective sentiment-aware representations for each language. The learned representations are then integrated to facilitate cross-lingual sentiment classification. The proposed method demonstrates state-of-the-art performance on benchmark datasets, which is sustained even when sentiment labels are scarce.

CCS CONCEPTS

• **Information systems → Sentiment analysis**.

*Corresponding author: Xuanzhe Liu (xzl@pku.edu.cn).

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313600

KEYWORDS

Emoji; cross-lingual analysis; sentiment classification

ACM Reference Format:
Zhenpeng Chen, Sheng Shen, Ziniu Hu, Xuan Lu, Qiaozhu Mei, and Xuanzhe Liu. 2019. Emoji-Powered Representation Learning for Cross-Lingual Sentiment Classification. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313600

1 INTRODUCTION

Sentiment analysis has become a critical topic in various research communities, including natural language processing (NLP) [22, 26], Web mining [23, 38], information retrieval [14, 57], ubiquitous computing [30, 53], and human-computer interaction [24, 59]. Due to its effectiveness in understanding user attitudes, emotions, and even latent psychological statuses from text, sentiment analysis has been widely applied to all kinds of Web content such as blogs, Tweets, user reviews, and forum discussions, and it has been a critical component in many applications such as customer review tracking [27], sales prediction [39], product ranking [41], stock market prediction [54], opinion polling [45], recommender systems [55], personalized content delivery [29], and online advertising [52].

Similar to many other text mining tasks, existing work on sentiment analysis mainly deals with English texts [22, 26, 38, 57]. Although some efforts have also been made with other languages such as Japanese [51], sentiment analysis for non-English languages is far behind. This creates a considerable inequality in the quality of the aforementioned Web services received by non-English users, especially considering that 74.6% of Internet users are non-English speakers as of 2018 [9]. The cause of this inequality is quite simple: effective sentiment analysis tools are often built upon supervised learning techniques, and *there are way more labeled examples in English than in other languages.*

A straightforward solution is to transfer the knowledge learned from a label-rich language (i.e., the source language, usually English) to another language that has fewer labels (i.e., the target language), an approach known as *cross-lingual sentiment classification* [19].

In practice, the biggest challenge of cross-lingual sentiment classification is how to fill the linguistic gap between English and the target language. Many choose to bridge the gap through standard NLP techniques, and in particular, most recent studies have been using off-the-shelf machine translation tools to generate pseudo parallel corpora and then learn bilingual representations for the downstream sentiment classification task [50, 58, 62]. More specifically, many of these methods enforce the aligned bilingual texts to share a unified embedding space, and sentiment analysis of the target language is conducted in that space.

Although this approach looks sensible and easily executable, the performance of these machine translation-based methods often falls short. Indeed, a major obstacle of cross-lingual sentiment analysis is the so-called *language discrepancy* problem [19], which machine translation does not tackle well. More specifically, sentiment expressions often differ a lot across languages. Machine translation is able to retain the general expressions of sentiments that are shared across languages (e.g., "*angry*" or "怒っている" for negative sentiment), but it usually loses or even alters the sentiments in language-specific expressions [44]. As an example, in Japanese, the common expression "湯水のように使う" indicates a negative sentiment, describing the excessive usage or waste of a resource. However, its translation in English, "*use it like hot water*," not only loses the negative sentiment but also sounds odd.

The reason behind this pitfall is easy to explain: machine translation tools are usually trained on parallel corpora that are built in the first place to capture patterns shared across languages instead of patterns specific to individual languages. In other words, the problem is due to the failure to retain language-specific sentiment knowledge when unilaterally pursuing generalization across languages. A new bridge needs to be built beyond machine translation, which not only transfers "general sentiment knowedge" from the source language but also captures "private sentiment knowledge" of the target language. *That bridge can be built with emojis.*

In this paper, we tackle the problem of cross-lingual sentiment analysis by employing *emojis* as an instrument. Emojis are considered an emerging ubiquitous language used worldwide [16, 40]; in our approach they serve both as a proxy of sentiment labels and as a bridge between languages. Their functionality of expressing emotions [21, 34] motivates us to employ emojis as complementary labels for sentiments, while their ubiquity [16, 40] makes it feasible to learn emoji-sentiment representations for almost every active language. Coupled with machine translation, the cross-language patterns of emoji usage can complement the pseudo parallel corpora and narrow the language gap, and the language-specific patterns of emoji usage help address the language discrepancy problem.

We propose ELSA, a novel framework of *Emoji-powered representation learning for cross-Lingual Sentiment Analysis*. Through ELSA, language-specific representations are first derived based on modeling how emojis are used alongside words in each language. These per-language representations are then integrated and refined to predict the rich sentiment labels in the source language, through the help of machine translation. Different from the mandatorily aligned bilingual representations in existing studies, the joint representation learned through ELSA catches not only the general sentiment patterns across languages, but also the language-specific patterns. In this way, the new representation and the downstream tasks are no longer dominated by the source language.

We evaluate the performance of ELSA on a benchmark Amazon review dataset that has been used in various cross-lingual sentiment classification studies [50, 58, 62]. The benchmark dataset covers nine tasks combined from three target languages (i.e., Japanese, French, and German) and three domains (i.e., book, DVD, and music). Results indicate that ELSA outperforms existing approaches on all of these tasks in terms of classification accuracy. Experiments also show that the emoji-powered model works well even when the volume of unlabeled and labeled data are rather limited. To evaluate the generalizability of ELSA, we also apply the method to Tweets, which again demonstrates state-of-the-art performance. In summary, the major contributions of this paper are as follows:

- To the best of our knowledge, this is the first study that leverages emojis as an instrument in cross-lingual sentiment classification. We demonstrate that emojis provide not only surrogate sentiment labels but also an effective way to address language discrepancy.
- We propose a novel representation learning method to incorporating language-specific knowledge into cross-lingual sentiment classification, which uses an attention-based Long Short-Term Memory (LSTM) model to capture sentiments from emoji usage.
- We demonstrate the effectiveness and efficiency of ELSA for cross-lingual sentiment classification using multiple large-scale datasets. ELSA significantly improves the state-of-the-art results on the benchmark datasets.[1]
- The use of emojis as a bridge provides actionable insights into other Web mining applications that suffer from similar problem of inequality among languages.

The rest of this paper is organized as follows. Section 2 presents the related work. Section 3 formulates the problem and presents the proposed approach (ELSA) to cross-lingual representation learning. Section 4 evaluates ELSA and analyzes the effectiveness of emojis in the learning process. Section 5 discusses the scalability and generalizability of ELSA, followed by concluding remarks in Section 6.

2 RELATED WORK

We start with a summary of existing literature related to our study.

Emojis. Emojis, also known as ideograms or smileys, can be used as compact expressions of objects, topics, and emotions. Being encoded in Unicode, they have no language barriers and are diffused on the Internet rapidly [40]. The prevalence of emojis has attracted researchers from various research communities such as NLP, ubiquitous computing, human-computer interaction, multimedia, and Web mining [12, 16, 20, 21, 34, 40, 43]. Many efforts have been devoted to studying their usage across platforms [43], across genders [20], across languages [16], and across cultures [40]. The various non-verbal functions of emojis play an important role in their wide adoption. Emojis are used to replace content words, express situational and additional emotions, adjust tones, express intimacy, etc. [21, 34]. In particular, expressing sentiment is demonstrated to be the most popular intention for using emojis [34], so that emojis

[1]The benchmark datasets, scripts, and pre-trained models are available at https://github.com/sInceraSs/ELSA.

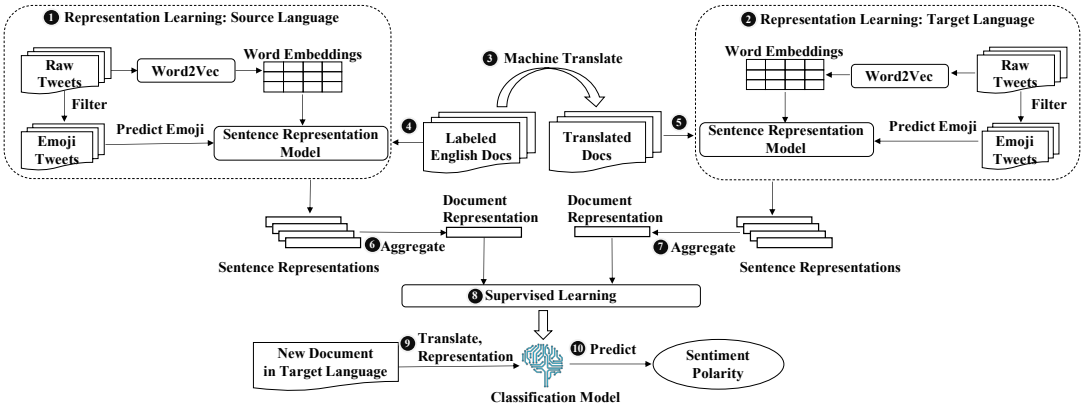


Figure 1: The workflow of ELSA.

can be used as effective proxies for sentiment polarities [26]. Considering the ubiquitous usage of emojis across languages and their functionality of expressing sentiments, we make the first effort to use emojis as an instrument to improve cross-lingual sentiment analysis.

Textual Sentiment Analysis. Sentiment analysis is a classical NLP task aiming to study the emotions, opinions, evaluations, appraisals, and attitudes of people from text data [37]. Many widely used tools, such as SentiStrength [56] and LIWC [49], simply aggregate the polarity of individual words to determine the overall sentiment score of a text. Better performance of sentiment classification is often obtained through supervised machine learning [46]. Recently, with the emergence of deep learning techniques, many researchers have attempted to use advanced neural network models for sentiment analysis [60]. Supervised machine learning methods, including deep learning models, usually require a large volume of labeled data for training. In reality, however, high-quality sentiment labels are often scarce due to the labor-consuming and error-prone human annotation process [26]. To address this limitation, researchers have used sentimental hashtags and emoticons as weak sentiment labels [22, 23]. These weak labels are usually language/community-specific. In addition, figuring out the sentiment polarities of certain hashtags or emoticons can be hard. In recent years, emoticons have been gradually replaced by increasingly popular emojis [48], and emojis have started to be explored as proxies of sentiment labels [26]. We follow the same intuition and utilize emojis as surrogate labels to learn per-language representations. Instead of attempting to directly map emojis to sentiment polarities, however, we integrate these language-specific representations and feed them through downstream tasks to predict real, high quality sentiment labels (in the source language).

Cross-Lingual Text Classification. There is a significant imbalance in the availability of labeled corpora among different languages: more in English, and much fewer in other languages. Cross-lingual learning is a common approach to tackling this problem in various text mining tasks such as Web page classification [36], topic categorization [61], and sentiment analysis [50, 58, 62]. Many researchers divide cross-lingual learning process into two stages: first encoding texts in the source and the target languages into continuous representations, and then utilizing these representations for the final classification task in the target language [50, 58, 62]. To bridge the linguistic gap between the source and the target languages, most studies introduce a translation oracle to project different languages' representations into a unified space at different (e.g., word or document) levels [18, 50, 58, 62]. The performance of these methods thus heavily depends on the quality of the machine translation tools and the pseudo parallel corpora they generate. Unfortunately, different from topical words, emotional language patterns like sentiment (or sarcasm, humor), which present strong language-specific characteristics, cannot be easily transferred in this way. We utilize the easily accessible emoji-texts to incorporate both cross-language and language-specific knowledge into the representations of the source and the target languages. The implicit sentiment knowledge encoded in the usage of diverse emojis solves both the label imbalance and the language discrepancy problems.

3 THE ELSA APPROACH

To better illustrate the workflow of ELSA, we first give a formulation of our problem. Cross-lingual sentiment classification aims to use the labeled data in a source language (i.e., English) to learn a model that can classify the sentiment of test data in a target language. In our setting, besides labeled English documents (L_S), we also have large-scale unlabeled data in English (U_S) and in the target language (U_T). Furthermore, there exist unlabeled data containing emojis, both in English (E_S) and in the target language (E_T). In practice, these unlabeled, emoji-rich data can be easily obtained from online social media such as Twitter. Our task is to build a model that can classify the sentiment polarity of document in the target language solely based on the labeled data in the source language (i.e., L_S) and the different kinds of unlabeled data (i.e., U_S, U_T, E_S and E_T). Finally, we use a held-out set of labeled documents in the target language (L_T), which can be small, to evaluate the model.

The workflow of ELSA is illustrated in Figure 1, with the following steps. In *step 1* and *step 2*, we build sentence representation models for both the source and the target languages. Specifically, for each language, we employ a large number of Tweets to learn word

embeddings (through Word2Vec [42]) in an unsupervised fashion. From these word embeddings, we learn higher-level sentence representation through predicting the emojis used in a sentence. This can be viewed as a distantly supervised learning process, where emojis serve as surrogate sentiment labels. In *step 3*, we translate each labeled English document into the target language, sentence by sentence, through *Google Translate*. Both the English sentences and their translations are fed into the representation models learned in steps 1 and 2 to obtain their per-language representations (*step 4* and *step 5*). Then in *step 6* and *step 7* we aggregate these sentence representations back to form two compact representations for each training document, one in English and the other in the target language. In *step 8*, we use the two representations as features to predict the real sentiment label of each document and obtain the final sentiment classifier. In the test phase, for a new document in the target language, we translate it into English and then follow the previous steps to obtain its representation (*step 9*), based on which we predict the sentiment label using the classifier (*step 10*).

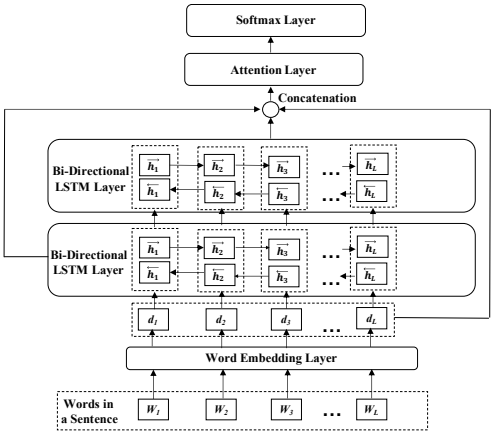


Figure 2: Network architecture for representation learning through emoji prediction.

3.1 Representation Learning

Representations of documents need to be learned before we train the sentiment classifier. Intuitively, one could simply use off-the-shelf word embedding techniques to create word representations and then average the word vectors to obtain document embeddings. Such embeddings, however, capture neither per-language nor cross-language sentiment patterns. Since emojis are widely used to express sentiments across languages, we learn sentiment-aware representations of documents using emoji prediction as an instrument. Specifically, in a distantly supervised way, we use emojis as surrogate sentiment labels and learn sentence embeddings by predicting which emojis are used in a sentence. This representation learning process is conducted separately in the source and the target languages to capture language-specific sentiment expressions.

The architecture of the representation learning model is illustrated in Figure 2. First, we pre-train low-level word embeddings using tens of millions of unlabeled Tweets (i.e., the word embedding layer). Then, we represent every single word as a unique vector and use stacked bi-directional LSTM layers and one attention layer to encode these word vectors into sentence representations. The attention layer takes the outputs of both the embedding layer and the two LSTM layers as input, through the skip-connection algorithm [31], which enables unimpeded information flow in the whole training process. Finally, the model parameters are learned by minimizing the output error of the softmax layer. The details of the architecture are elaborated below.

Word Embedding Layer. The word embeddings are pre-trained with the skip-gram algorithm [42] based on either U_S or U_T, which encode every single word into a continuous vector space. Words that commonly occur in a similar context are embedded closely in the vector space, which captures word semantic information. We leave the details of this standard Word2Vec process to the readers [42].

Bi-Directional LSTM Layer. As a special type of recurrent neural network (RNN), LSTM [33] is particularly suitable for modeling the sequential property of text data. At each step (e.g., word token), LSTM combines the current input and knowledge from the previous steps to update the states of the hidden layer. To tackle the gradient vanishing problem [32] of traditional RNNs, LSTM incorporates a gating mechanism to determine when and how the states of hidden layers can be updated. Each LSTM unit contains a memory cell and three gates (i.e., an input gate, a forget gate, and an output gate) [47]. The input and output gates control the input activations into the memory cell and the output flow of cell activations into the rest of the network, respectively. The memory cells in LSTM store the sequential states of the network, and each memory cell has a self-loop whose weight is controlled by the forget gate.

Let us denote each sentence in E_S or E_T as (x, e), where $x = [d_1, d_2, ..., d_L]$ as a sequence of *word* vectors representing the plain text (by removing emojis) and e as one emoji contained in the text. At step t, LSTM computes unit states of the network as follows:

$$
\begin{aligned}
i^{(t)} &= \sigma(U_i d_t + W_i h^{(t-1)} + b_i),\\
f^{(t)} &= \sigma(U_f d_t + W_f h^{(t-1)} + b_f),\\
o^{(t)} &= \sigma(U_o d_t + W_o h^{(t-1)} + b_o),\\
c^{(t)} &= f_t \odot c^{(t-1)} + i^{(t)} \odot tanh(U_c d_t + W_c h^{(t-1)} + b_c),\\
h^{(t)} &= o^{(t)} \odot tanh(c^{(t)}),
\end{aligned}
$$

where $i^{(t)}$, $f^{(t)}$, $o^{(t)}$, $c^{(t)}$, and $h^{(t)}$ denote the state of the input gate, forget gate, output gate, memory cell, and hidden layer at step t. W, U, b respectively denote the recurrent weights, input weights, and biases. $\odot$ is the element-wise product. We can extract the latent vector for each step t from LSTM. In order to capture the information from the context both preceding and following a word, we use the bi-directional LSTM. We concatenate the latent vectors from both directions to construct a bi-directional encoded vector h_i for every single word vector d_i, which is:

$$
\begin{aligned}
\overrightarrow{h_i} &= \overrightarrow{LSTM}(d_i), i \in [1, L],\\
\overleftarrow{h_i} &= \overleftarrow{LSTM}(d_i), i \in [L, 1],\\
h_i &= [\overrightarrow{h_i}, \overleftarrow{h_i}].
\end{aligned}
$$

Attention Layer. We employ a skip-connection that concatenates the outputs of the embedding layer and the two bi-directional LSTM

layers as the input of the attention layer. The i-th word of the input sentence can be represented as u_i:

$$u_i = [d_i, h_{i1}, h_{i2}],$$

where d_i, h_{i1}, and h_{i2} denote the encoded vectors of words extracted in the word embedding layer and the first and second bi-directional LSTMs, respectively. Since not all words contribute equally to predicting emojis or expressing sentiments, we employ the attention mechanism [13] to determine the importance of every single word. The attention score of the i-th word is calculated by

$$a_i = \frac{exp(W_a u_i)}{\sum_{j=1}^{L} exp(W_a u_j)},$$

where W_a is the weight matrix used by the attention layer. Then each sentence can be represented as the weighted sum of all words in it, using the attention scores as weights. That is,

$$v = \sum_{i=1}^{L} a_i u_i.$$

Softmax Layer. The sentence representation is then transferred into the softmax layer, which returns a probability vector Y. Each element of this vector indicates the probability that this sentence contains a specific emoji. The i-th element of the probability vector is calculated as:

$$y_i = \frac{exp(v^T w_i + b_i)}{\sum_{j=1}^{K} exp(v^T w_j + b_j)},$$

where w_i and b_i define the weight and bias of the i-th element. Finally, we learn the model parameters by minimizing the cross entropy between the output probability vectors and the one-hot vectors of the emoji contained in each sentence. After learning the parameters, we can extract the output of the attention layer to represent each input sentence. Through this emoji-prediction process, words with distinctive sentiments can be identified, and the plain text surrounding the same emojis will be represented similarly. Given the fact that the sentiment labels are limited, once the emoji-powered sentence representations are trained, they are locked in the downstream sentiment prediction task to avoid over-fitting.

3.2 Training the Sentiment Classifier

Based on the pre-trained, per-language sentence representations, we then learn document representations and conduct cross-lingual sentiment classification.

First, for each English document $D_s \in L_S$, we use the pre-trained English representation model to embed every single sentence in it. Second, we aggregate these sentence representations to derive a compact document representation. Because different parts of a document contribute differently to the overall sentiment, we once again adopt the attention mechanism here. Supposing the sentence vectors as v_i, we calculate the document vector r_s as:

$$r_s = \sum_{i=1}^{N} \beta_i v_i, \text{ where}$$
$$\beta_i = \frac{exp(W_b v_i)}{\sum_{j=1}^{N} exp(W_b v_j)},$$

Table 1: The sizes of the Tweets and emoji-Tweets.

Language	English	Japanese	French	German
Raw Tweets	39.4M	19.5M	29.2M	12.4M
Emoji-Tweets	6.6M	2.9M	4.4M	2.7M

where W_b is the weight matrix of the attention layer and β_i is the attention score of the i-th sentence in the document. Next, we use *Google Translate* to translate D_s into the target language (D_t). We then leverage the pre-trained target-language representation model to form representations for each translated document following the same process above. Supposing the text representations of D_s and D_t are r_s and r_t respectively, we concatenate them into a joint representation $r_c = [r_s, r_t]$, which contains sentiment knowledge from both English and the target language, ensuring that our model is not dominated by the labeled English documents. Finally, we input r_c into an additional softmax layer to predict the real sentiment label of D_s.

3.3 Sentiment Classification for Target Language

When we receive an unlabeled document in L_T, we first translate it into English. Based on the representation models trained above, the original document and its English translation can be represented as r_t and r_s. We represent this document as $[r_s, r_t]$ and input it into the classifier, which outputs a predicted sentiment polarity.

4 EVALUATION

In this section, we evaluate the effectiveness and efficiency of ELSA using standard benchmark datasets for cross-lingual sentiment classification as well as a large-scale corpus of Tweets.

4.1 The Dataset

The labeled data (L_S for training and L_T for testing) used in our work are from the Amazon review dataset [3] created by Prettenhofer and Stein [50]. This dataset is representative and used in a variety of cross-lingual sentiment classification work [50, 58, 62]. It covers four languages (i.e., English, Japanese, French, and German) and three domains (i.e., book, DVD, and music). For each combination of language and domain, the dataset contains 1,000 positive reviews and 1,000 negative reviews. We select English as the source language and the other three as the target languages. Therefore, we can evaluate our approach on nine tasks in total (i.e., combinations of the three domains and three target languages). For each task, we use the 2,000 labeled English reviews in the corresponding domain for training and the 2,000 labeled reviews in each target language for evaluation. The translations of the test reviews are already provided in this dataset, so we only need to translate the English reviews into target languages.

To achieve unlabeled data (U_S and U_T), we collect a sample of English, Japanese, French, and German Tweets between September 2016 and March 2018. All collected Tweets are used to train the word embeddings. As emojis are widely used on Twitter [48], we are able to extract emoji-labeled Tweets, which are used to learn emoji-powered sentence representations. For each language, we extract Tweets containing the top 64 emojis used in this language.

As many Tweets contain multiple emojis, for each Tweet, we create separate examples for each unique emoji used in it to make the emoji prediction a single-label classification task instead of more complicated multi-label classification.

We then conduct the following preprocessing procedures for the documents. We remove all Retweets, and Tweets that contain URLs, to ensure that words appear in their original contexts and that the meaning of the Tweets do not depend on external content. Then we tokenize all the texts (including reviews and Tweets) into words, convert them into lowercase, and shorten the words with redundant characters into their canonical forms (e.g., "cooooool" is converted to "cool"). As Japanese words are not separated by white spaces, we use a tokenization tool called *MeCab* [2] to segment Japanese documents. In addition, we use special tokens to replace mentions and numbers. The processed emoji-Tweets provide the E_S and E_T datasets, whose statistics are presented in Table 1.

4.2 Implementation Details

We learn the initial word embeddings using the skip-gram model with the window-size of 5 on the raw Tweets. The word vectors are then fine-tuned during the sentence representation learning phase. In the representation learning phase, to regularize our model, L2 regularization with parameter 10^{-6} is applied for embedding weights. Dropout is applied at the rate of 0.5 before the softmax layer. The hidden units of bi-directional LSTM layers are set as 1,024 (512 in each direction). We randomly split the emoji-Tweets into the training, validation, and test sets in the proportion of 7:2:1. Accordingly, we use early stopping [17] to tune hyperparameters based on the validation performance through 50 epochs, with mini-batch size of 250. We used the Adam algorithm [35] for optimization, with the two momentum parameters set to 0.9 and 0.999, respectively. The initial learning rate was set to 10^{-3}. In the phase of training the sentiment classifier, for exhaustive parameter tuning, we randomly select 90% of the labeled data as the training set and the remaining 10% as the validation set. The whole framework is implemented with TensorFlow [11].

4.3 Baselines and Accuracy Comparison

To evaluate the performance of ELSA, we employ three representative baseline methods for comparison:

MT-BOW uses the bag-of-words features to learn a linear classifier on the labeled English data [50]. It uses *Google Translate* to translate the test data into English and applies the pre-trained classifier to predict the sentiment polarity of the translated documents.

CL-RL is the word-aligned representation learning method proposed by Xiao and Guo [58]. It constructs a unified word representation that consists of both language-specific components and shared components, for the source and the target languages. To establish connections between the two languages, it leverages *Google Translate* to create a set of critical parallel word pairs, and then it forces each parallel word pair to share the same word representation. The document representation is computed by taking the average over all words in the document. Given the representation as features, it trains a linear SVM model using the labeled English data.

Table 2: The accuracy of ELSA (standard deviations in parentheses) and baseline methods on the nine benchmark tasks.

Language	Domain	MT-BOW	CL-RL	BiDRL	ELSA
Japanese	Book	0.702	0.711	0.732	**0.783 (0.003)**
	DVD	0.713	0.731	0.768	**0.791 (0.004)**
	Music	0.720	0.744	0.788	**0.808 (0.005)**
French	Book	0.808	0.783	0.844	**0.860 (0.002)**
	DVD	0.788	0.748	0.836	**0.857 (0.002)**
	Music	0.758	0.787	0.825	**0.860 (0.002)**
German	Book	0.797	0.799	0.841	**0.864 (0.001)**
	DVD	0.779	0.771	0.841	**0.861 (0.001)**
	Music	0.772	0.773	0.847	**0.878 (0.002)**

BiDRL is the document-aligned representation learning method proposed by Zhou *et al.* [62]. It uses *Google Translate* to create labeled parallel documents and forces the pseudo parallel documents to share the same embedding space. It also enforces constraints to make the document vectors associated with different sentiments fall into different positions in the embedding space. Furthermore, it forces documents with large textual differences but the same sentiment to have similar representations. After this representation learning process, it concatenates the vectors of one document in both languages and trains a logistic regression sentiment classifier.

As the benchmark datasets have quite balanced positive and negative reviews, we follow the aforementioned studies to use accuracy as an evaluation metric. All the baseline methods have been evaluated with exactly the same training and test data sets used in previous studies [62], so we make direct comparisons with their reported results. Unfortunately, we cannot obtain the individual predictions of these methods, so we are not able to report the statistical significance (such as McNemar's test [25]) of the difference between these baselines and ELSA. To alleviate this problem and get robust results, we run ELSA 10 times with different random initiations and summarize its average accuracy and standard deviation in Table 2, as well as the reported performance of the baselines.

As illustrated in Table 2, ELSA outperforms all three baseline methods on all nine tasks. Looking more closely, the performance of all methods in Japanese sentiment classification is worse than in French and German tasks. According to the language systems defined by ISO 639 [10], English, French, and German belong to the same language family (i.e., Indo-European), while Japanese belongs to the Japonic family. In other words, French and German are more in common with English, and it is expected to be easier to translate English texts into French and German and transfer the sentiment knowledge from English to them. Therefore, in fact, Japanese tasks are most difficult and none of the previous methods have been able to achieve an accuracy above 0.8. It is encouraging to find that ELSA achieves an accuracy of 0.808 on the Japanese music task and an accuracy close to 0.8 (0.791) on the Japanese DVD task. The 0.783 accuracy on the book task is also non-negligible as it improves on the best existing model by almost 7 percent. In addition, although the French and German tasks are a little easier than the Japanese ones, none of the existing approaches can achieve an accuracy over 0.85 on any of the six tasks. However, our approach can achieve a mean accuracy higher than 0.85 on all of the six tasks.

Next, we compare the results more thoroughly and further demonstrate the advantages of our approach. As is shown, the representation learning approaches (CL-RL, BiDRL, and ELSA) all

outperform the shallow method MT-BOW on most tasks. This is reasonable as representation learning approaches embed words into high-dimensional vectors in a continuous semantic space and thus overcome the feature sparsity issue of traditional bag-of-words approaches. Furthermore, we observe that the document-level representation approaches (BiDRL and ELSA) outperform the word-level CL-RL. This indicates that incorporating document-level information into representations is more effective than focusing on individual words. Finally, ELSA outperforms the BiDRL on all tasks. In order to narrow the linguistic gap, BiDRL leverages only pseudo parallel texts to learn the common sentiment patterns between languages. Besides the pseudo parallel texts, ELSA also learns from the emoji usage in both languages. On the one hand, as a ubiquitous emotional signal, emojis are adopted across languages to express common sentiment patterns, which can complement the pseudo parallel corpus. On the other hand, the language-specific patterns of emoji usage help incorporate the language-specific knowledge of sentiments into the representation learning, which can benefit the downstream sentiment classification in the target language. As a next step, we explore the role of emojis in the learning process with a more comprehensive investigation.

4.4 The Power of Emojis

To further evaluate the contribution of emojis in ELSA, we conduct subsequent experiments to investigate the effects of emojis from three perspectives, i.e., overall performance, effectiveness of representation learning, and text comprehension.

4.4.1 Overall Performance. To understand how emojis affect cross-lingual sentiment classification in general, a straightforward idea is to remove the emoji-prediction phase and compare simplified versions of ELSA:

N-ELSA removes the emoji-prediction phase of both languages and directly uses two attention layers to realize the transformation from word vectors to the final document representation. There is no emoji data used in this model.

T-ELSA removes the emoji-based representation learning on the English side. It uses the emoji-powered representations for the target language and translates labeled English documents into the target language to train a sentiment classifier for the target language. This model only leverages emoji usage in the target language.

S-ELSA removes the emoji-based representation learning in the target language. It uses the emoji-powered representations of English and trains a sentiment classifier based on labeled English documents. Documents in the target language are first translated into English and then classified. This model only leverages emoji usage in the source language (i.e., English).

Test accuracy of these models is illustrated in Table 3. We find that ELSA outperforms N-ELSA on all nine tasks. N-ELSA is only a little better than uniform guess (50%) since it learns the common patterns between languages only from pseudo parallel texts and does not incorporate sentiment information effectively. An alternative conjecture is that 2,000 reviews are insufficient to train such a complex model, which may have led to the problem of over-fitting.

To test between the two hypotheses, we mix up the labeled reviews in English and in the target language and randomly select 2,000 examples from the mixed set for training and use the remaining samples as a new test set. All other settings of the experiment are kept the same except for the new train/test split. Trained and tested in this way, the accuracy of N-ELSA becomes acceptable, with an average accuracy of 0.777 on all tasks. This indicates that over-fitting might not have been the major reason, while language discrepancy might be. Indeed, N-ELSA can still work well if we effectively incorporate cross-language sentiment information into the training process. More specifically, the original N-ELSA is dominated by English sentiment information learned from pseudo parallel texts and fails to generalize to the target language correctly. When we input the sentiment information (labeled documents) of both English and the target language into the model, performance improves. Unfortunately, in a cross-lingual sentiment classification setting, we can not acquire enough labels in the target language. Emojis help the model capture generalizable sentiment knowledge, even if there is no labeled example for training in the target language.

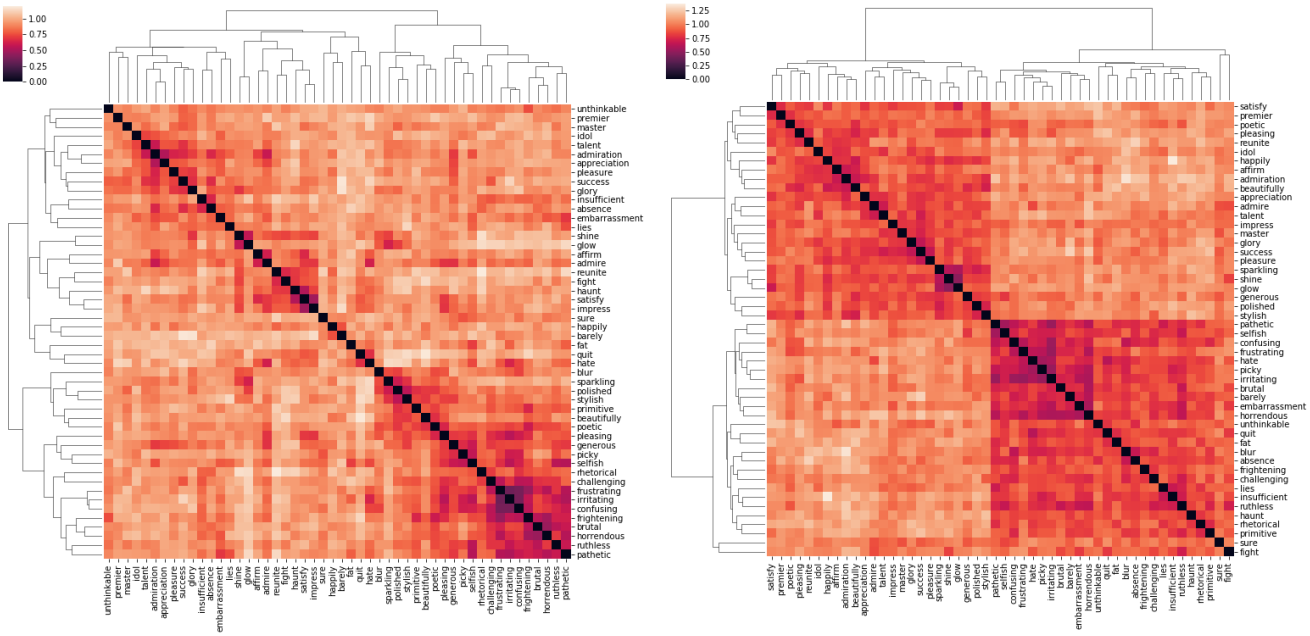
Table 3: Performance of ELSA and its simplified versions.

Language	Domain	N-ELSA	T-ELSA	S-ELSA	ELSA
Japanese	Book	0.527*	0.742*	0.753*	0.783
	DVD	0.507*	0.756*	0.766*	0.791
	Music	0.513*	0.792*	0.778*	0.808
French	Book	0.505*	0.821*	0.850*	0.860
	DVD	0.507*	0.816*	0.843*	0.857
	Music	0.503*	0.811*	0.848*	0.860
German	Book	0.513*	0.804*	0.848*	0.864
	DVD	0.521*	0.790*	0.849*	0.861
	Music	0.513*	0.818*	0.863*	0.878

* indicates the difference between ELSA and its simplified versions is statistically significant ($p < 0.05$) by McNemar's test.

In addition, ELSA also consistently achieves better accuracy compared to T-ELSA and S-ELSA on all tasks (McNemar's test [25] is performed and the differences are all statistically significant at the 5% level). The superiority of ELSA shows that only extracting sentiment information from one language is not enough for the cross-lingual sentiment task and that incorporating language-specific knowledge for both languages is critical to the model's performance. Indeed, S-ELSA fails to capture sentiment patterns in the target language; and T-ELSA falls short in extracting transferable sentiment patterns from English (indicating that emojis are still beneficial even if there are sentiment labels in a language).

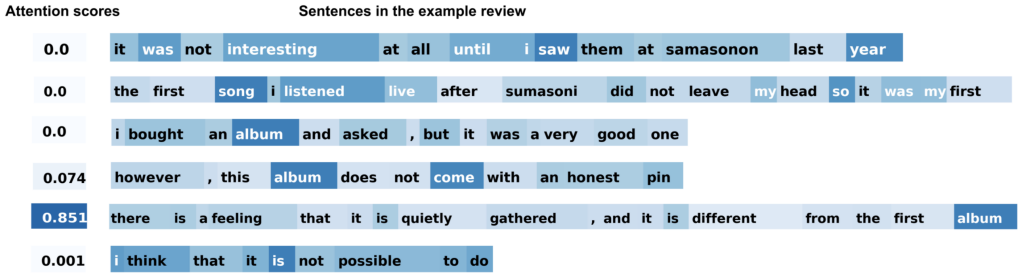
4.4.2 Effectiveness of Representation Learning. To better understand the sentiment information learned through the emoji usage, we then conduct an empirical experiment at the word representation level. Recall that after the word embedding phase, each individual word can be represented by a unique vector and that these word vectors are then fine-tuned in the emoji-prediction phase. Next, we would like to evaluate whether sentiment information is better captured by the new word representations under the effects of emojis. We sample 50 English words with distinct sentiments from the MPQA subjectivity lexicon [1] based on their frequencies in our corpus. These words are manually labeled in terms of positive or negative polarity from MPQA, and we regard these labels as the ground-truth for further evaluation.



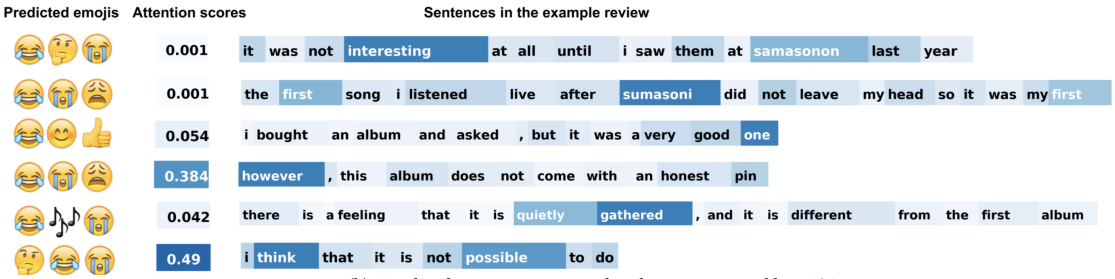
(a) Clusters of selected words with Word2Vec representations.

(b) Clusters of selected words with emoji-powered representations.

Figure 3: Comparison of word representations *with* and *without* emoji prediction.



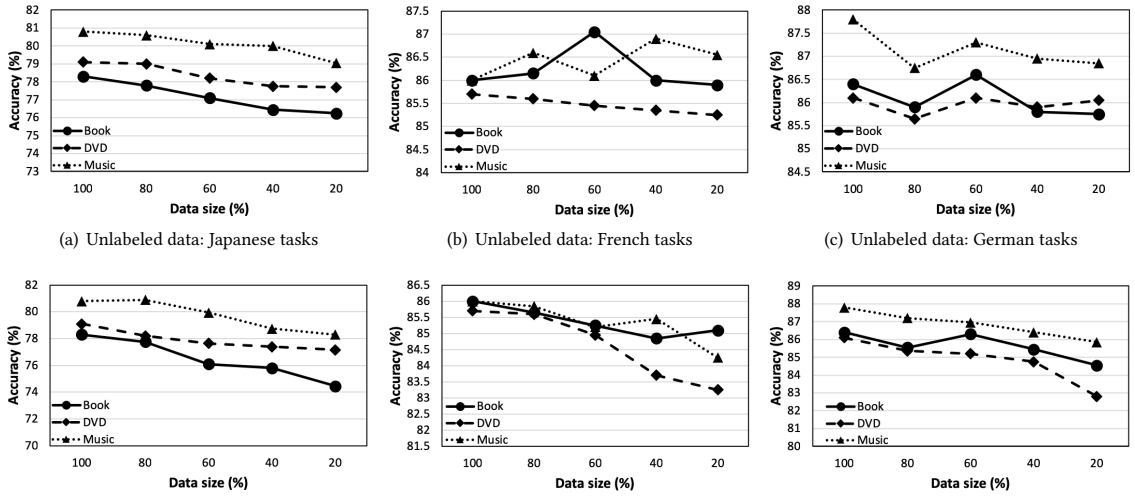
(a) Word and sentence attention distribution generated by N-ELSA.



(b) Word and sentence attention distribution generated by ELSA.

Figure 4: Case study: Effect of emojis on text comprehension.

We expect that an informative representation can embed words with same sentiment polarity closely in the vector space. To measure and illustrate the similarity, we calculate the similarity score between every two words using the cosine of the corresponding embedding vectors. Based on the cosine similarity, we perform a hierarchical clustering [15] and visualize the clustering results in Figure 3. The color scale of each cell indicates the similarity between the two words. The darker the cell, the more similar the representations of the two words.



(a) Unlabeled data: Japanese tasks (b) Unlabeled data: French tasks (c) Unlabeled data: German tasks

(d) Labeled English data: Japanese tasks (e) Labeled English data: French tasks (f) Labeled English data: German tasks

Figure 5: Accuracy of ELSA when size of unlabeled and labeled data changes.

In Figure 3(a), we use naive embeddings learned by Word2Vec (no emoji), and words with different sentiments cannot be clearly separated. Many words with different sentiments are embedded closely, for example, "generous" and "picky" in the bottom right section. This indicates that shallow word embeddings do not effectively capture the sentiment information.

In contrast, in Figure 3(b), we can easily observe two clusters after the fine-tuned emoji-prediction model. The top left corner cluster contains the positive words, while the bottom right corner contains the negative words. Only one positive word, "sure," is incorrectly clustered with negative words. By checking the contexts of this word in our corpus, we find it is usually co-used with both positive and negative words, making its polarity ambiguous. The correct clustering of nearly all the words indicates that emoji usage is an effective channel to capture sentiment knowledge, which is desirable for downstream sentiment classification.

4.4.3 Text Comprehension. We then explore how the emoji-powered representations benefit text comprehension. We select a representative case that is incorrectly classified by N-ELSA but correctly classified by ELSA. This case is selected from the Japanese test samples and we use the segment of its translated English version for illustration in Figure 4. Although the whole document expresses dissatisfaction with an album, it is not that easy to identify this intent directly from each single sentence due to the translation quality and the document's complex compositions. For example, if we consider only the third sentence without context, the author seems to express a positive attitude. However, in fact, the author expresses an obviously negative attitude in the fourth and sixth sentences.

In Figure 4, we present the attention distribution of words and sentences generated by N-ELSA and ELSA, which indicates how the two models comprehend this document, or the rationale behind their classification decisions. We use the color scale of the background to indicate the attention scores of words in each sentence. The darker the word, the more it is attended to. For each sentence, we list its attention score in this document. In Figure 4(b), we also list the top 3 emojis ELSA predicts for each sentence, which may indicate its sentiment polarity predicted by ELSA.

Let us first look at Figure 4(a), which demonstrates how N-ELSA processes the sentiment information. On the word level, N-ELSA tends to focus more on neutral words like "song" or "album" instead of sentimental words. On the sentence level, an extremely high attention is placed on the fifth sentence. However, the fifth sentence describes how the album is different from the first one and it does not express the obviously negative sentiment.

In contrast, after incorporating of emojis, ELSA is able to work with a proper logic (see Figure 4(b)). ELSA places its attention to the emotional adjectives, such as "interesting" and "not possible," and contrast conjunctions such as "however." Thus, it manages to identify the sentiment of each sentence as expected, which can be further explained by the predicted emojis on the left. Besides the most popular 😂 in our corpus, 😭 and 😩 predicted for the fourth and sixth sentence indicate the negative sentiment of the author, while 👍 and 😊 in the third sentence indicate positive sentiment. Then on the sentence level, ELSA places less attention to the positive third sentence, while centering upon the fourth and the sixth sentences. Through this comparison, we can see that emojis bring additional knowledge to the text comprehension and make the attention mechanism more effective.

5 DISCUSSION

So far, we have presented the performance of ELSA on benchmark datasets and demonstrated the power of emojis in our learning process. There are some issues that could potentially affect its effectiveness and efficiency, which call for further discussion.

5.1 Sensitivity on Data Volume

As we learn text representations from large amount of Tweets, we want to investigate whether ELSA works well with a smaller volume of data. First, we investigate the size of unlabeled data. The English representation model, once learned, can be reused by any other English-target language pair. We only need to scale down the Tweets and emoji-Tweets in the target language and observe the changes in performance on benchmarks. In details, we use 80%, 60%, 40%, and 20% of the collected Tweets to re-train the target-language representation model and keep the final supervised training fixed. We summarize the results in Figures 5(a), 5(b), and 5(c). For the Japanese tasks, when we scale down the unlabeled data, the performance gets slightly worse. Comparing the results using 20% and 100% of the Tweets, the accuracy differences in three domains are 0.021, 0.014, and 0.018, respectively. For French and German, the performance fluctuates less than 0.01. Most importantly, ELSA can outperform the existing methods on all nine tasks even with the 20% unlabeled data. This indicates that even though a target language is not as actively used on Twitter, our approach still works.

Furthermore, although there are more labeled examples in English than other languages, in general, labels are still scarce. Hence, if a model can rely on even fewer labeled English documents, it is very desirable. To test this, we scale down the labeled data by 80%, 60%, 40%, and 20%. As shown in Figures 5(d), 5(e), and 5(f), the performance of ELSA slightly declines with the decrease of labels, but even with 20% labels (i.e., 400 labeled English samples), ELSA outperforms the existing methods using all 2,000 labeled samples on almost all tasks. This shows that with the help of large-scale emoji-Tweets, the model is less dependent on sentiment labels.

5.2 Generalizability

Most previous cross-lingual sentiment studies [50, 58, 62] used the Amazon review dataset for evaluation. To compare with them, we also adopt this dataset in the main experiment of this paper. Sentiment classification in other domains such as social media is also important. Can ELSA still work well in a new domain? To evaluate the generalizability of our approach, we apply ELSA to a representative type of social media data – Tweets. As Tweets are short and informal, sentiment classification for them is considered to be a big challenge [28].

As cross-lingual studies on Tweets are very limited, we take only one recent cross-lingual method (MT-CNN) proposed by Deriu *et al.* [23] for comparison. It also relies on large-scale unlabeled Tweets and a translation tool. It first trains a sentiment classifier for English and then applies it to the translations of text documents in the target language. The training process for English Tweets contains three phases. First, it uses raw Tweets to create word embeddings just like our method. Second, it leverages ":)" and ":(" as weak labels and applies a multi-layer CNN model to adapt the word embeddings. Finally, it trains the model on labeled English Tweets. This work and our work both have coverage of French and German Tweets, so we use the two as the target languages for comparison.

As the sentiment-labeled Tweets used by [23] are released in forms of Twitter IDs and some of them are no longer available now, we cannot directly compare our model to the reported results in [23]. For fair comparison, we reproduce their method on the labeled Tweets that can still be collected. Based on the pre-trained representation models of MT-CNN [5] and ELSA, we use the same labeled English Tweets to train and validate the two classifiers and then test them on the same data (i.e., labeled French and German Tweets that can be collected). We list the sizes of the labeled English, French, and German Tweets we use in Table 4. From the distribution, a naive baseline using uniform guess would achieve an accuracy of 0.451 for French and 0.628 for German.

Table 4: The sizes of labeled Tweets collected.

Dataset	Language	Positive	Neutral	Negative
Training	English [8]	5,101	3,742	1,643
Validation	English [7]	1,038	987	365
Test	French [4]	987	1,389	718
	German [6]	1,057	4,441	1,573

Table 5: Classification accuracy on French and German Tweets.

Language	ELSA	MT-CNN	Uniform Guess
French	0.696	0.535*	0.451*
German	0.809	0.654*	0.628*

*indicates the difference between ELSA and the baseline methods is statistically significant ($p < 0.05$) by McNemar's test.

Results are summarized in Table 5. The two approaches both outperform uniform guess, and ELSA outperforms the MT-CNN by 0.161 on French task and 0.155 on German task. Although we use the same training, validation, and test set for both approaches, we are still concerned about whether the pre-trained representation models have introduced unfairness. Specifically, if we have used more unlabeled Tweets for representation learning than MT-CNN, our outstanding performance may simply attribute to the size of data. To answer this question, we refer to [23] about their data size. We find that they uses 300M raw Tweets and 60M Tweets containing ":)" and ":(" for representation learning. In contrast, we only used 81M raw Tweets and 13.7M emoji-Tweets in three languages combined. Considering that emoticons are significantly less used than emojis on Twitter [48], although they use about 4.4 times more weak-labeled Tweets, these Tweets had to be collected from much more than 4.4 times of raw Tweets than ours. It is clear ELSA outperforms MT-CNN and relies less on data size.

6 CONCLUSION

As a ubiquitous emotional signal, emojis are widely adopted across languages to express sentiments. We leverage this characteristic of emojis, both using them as surrogate sentiment labels and using emoji prediction an instrument to address the language discrepancy in cross-lingual sentiment classification. We have presented ELSA, a novel emoji-powered representation learning framework, to capture both general and language-specific sentiment knowledge in the source and the target languages for cross-lingual sentiment classification. The representations learned by ELSA capture not only sentiment knowledge that generalizes across languages, but also language-specific patterns. We evaluate ELSA with comprehensive experiments on various benchmark datasets, which outperforms the state-of-the-art cross-lingual sentiment classification methods

even when the size of labeled and unlabeled data decreases. The promising results indicate that emojis may be used as an a general instrument for text mining tasks that suffer from the scarcity of labeled examples, especially in situations where an inequality among different languages presents.

ACKNOWLEDGMENT

This work was in part supported by the National Key R&D Program of China under the grant number 2018YFB1004800 and the Beijing Municipal Science and Technology Project under the grant number Z171100005117002. Qiaozhu Mei's work was supported by the National Science Foundation under grant numbers 1633370, 1131500, and 1620319. The authors would like to thank the invaluable supports from Mr. Wei Ai at University of Michigan and Ms. Jiawei Liu at Peking University. Zhenpeng Chen and Sheng Shen made equal contributions to this work. The work of Sheng Shen and Ziniu Hu was carried out when they were undergraduate students at Peking University.

REFERENCES

[1] 2005. MPQA opinion corpus. https://mpqa.cs.pitt.edu/lexicons/subj_lexicon/. (2005). Retrieved on October 22, 2018.
[2] 2006. MeCab: Yet Another Part-of-Speech and Morphological Analyzer. http://taku910.github.io/mecab. (2006). Retrieved on October 22, 2018.
[3] 2010. Webis-CLS-10. https://www.uni-weimar.de/en/media/chairs/computer-science-department/webis/data/corpus-webis-cls-10/. (2010). Retrieved on October 22, 2018.
[4] 2015. DEFT 2015: Test Corpus. https://deft.limsi.fr/2015/corpus.fr.php?lang=en. (2015). Retrieved on April 28, 2018.
[5] 2017. Deep-mlsa. https://github.com/spinningbytes/deep-mlsa. (2017). Retrieved on October 22, 2018.
[6] 2017. SB-10k: German Sentiment Corpus. https://www.spinningbytes.com/resources/germansentiment/. (2017). Retrieved on April 28, 2018.
[7] 2017. Twitter-2015test-A. http://alt.qcri.org/semeval2017/task4/index.php?id=download-the-full-training-data-for-semeval-2017-task-4. (2017). Retrieved on April 28, 2018.
[8] 2017. Twitter-2015train-A, Twitter-2016train-A, Twitter-2016dev-A, and Twitter-2016devtest-A. http://alt.qcri.org/semeval2017/task4/index.php?id=download-the-full-training-data-for-semeval-2017-task-4. (2017). Retrieved on April 28, 2018.
[9] 2018. Internet world users by language. https://www.internetworldstats.com/stats7.htm. (2018). Retrieved on October 22, 2018.
[10] 2018. ISO 639. https://www.iso.org/iso-639-language-codes.html. (2018). Retrieved on October 22, 2018.
[11] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning. In *OSDI*, Vol. 16. 265–283.
[12] Wei Ai, Xuan Lu, Xuanzhe Liu, Ning Wang, Gang Huang, and Qiaozhu Mei. 2017. Untangling emoji popularity through semantic embeddings. In *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017*. 2–11.
[13] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2014*.
[14] Georgios Balikas, Simon Moura, and Massih-Reza Amini. 2017. Multitask learning for fine-grained Twitter sentiment analysis. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2017*. 1005–1008.
[15] Ziv Bar-Joseph, David K. Gifford, and Tommi S. Jaakkola. 2001. Fast optimal leaf ordering for hierarchical clustering. In *Proceedings of the Ninth International Conference on Intelligent Systems for Molecular Biology*. 22–29.
[16] Francesco Barbieri, Germán Kruszewski, Francesco Ronzano, and Horacio Saggion. 2016. How cosmopolitan are emojis?: Exploring emojis usage and meaning over different languages with distributional semantics. In *Proceedings of the 2016 ACM Conference on Multimedia Conference, MM 2016*. 531–535.
[17] Rich Caruana, Steve Lawrence, and C. Lee Giles. 2000. Overfitting in neural nets: backpropagation, conjugate gradient, and early stopping. In *Proceedings of advances in neural information processing systems 13, NIPS 2000*. 402–408.
[18] A. P. Sarath Chandar, Stanislas Lauly, Hugo Larochelle, Mitesh M. Khapra, Balaraman Ravindran, Vikas C. Raykar, and Amrita Saha. 2014. An autoencoder approach to learning bilingual word representations. In *Advances in Neural Information Processing Systems 27, NIPS 2014*. 1853–1861.
[19] Qiang Chen, Chenliang Li, and Wenjie Li. 2017. Modeling language discrepancy for cross-lingual sentiment analysis. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017*. 117–126.
[20] Zhenpeng Chen, Xuan Lu, Wei Ai, Huoran Li, Qiaozhu Mei, and Xuanzhe Liu. 2018. Through a gender lens: learning usage patterns of emojis from large-scale Android users. In *Proceedings of the 2018 World Wide Web Conference, WWW 2018*. 763–772.
[21] Henriette Cramer, Paloma de Juan, and Joel R. Tetreault. 2016. Sender-intended functions of emojis in US messaging. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI 2016*. 504–509.
[22] Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using Twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics, COLING 2010*. 241–249.
[23] Jan Deriu, Aurélien Lucchi, Valeria De Luca, Aliaksei Severyn, Simon Müller, Mark Cieliebak, Thomas Hofmann, and Martin Jaggi. 2017. Leveraging large amounts of weakly supervised data for multi-language sentiment classification. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017*. 1045–1052.
[24] Nicholas Diakopoulos and David A. Shamma. 2010. Characterizing debate performance via aggregated Twitter sentiment. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010*. 1195–1198.
[25] Thomas G Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation* 10, 7 (1998), 1895–1923.
[26] Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017*. 1615–1625.
[27] Michael Gamon. 2004. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of 20th International Conference on Computational Linguistics, COLING 2004*.
[28] Anastasia Giachanou and Fabio Crestani. 2016. Like it or not: a survey of Twitter sentiment analysis methods. *Comput. Surveys* 49, 2 (2016), 28:1–28:41.
[29] Ryosuke Harakawa, Daichi Takehara, Takahiro Ogawa, and Miki Haseyama. 2018. Sentiment-aware personalized Tweet recommendation through multimodal FFM. *Multimedia Tools Appl.* 77, 14 (2018), 18741–18759.
[30] Kiraz Candan Herdem. 2012. Reactions: Twitter based mobile application for awareness of friends' emotions. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing, UbiComp 2012*. 796–797.
[31] M. Hermans and B. Schrauwen. 2013. Training and analysing deep recurrent neural networks. *Proceedings of advances in Neural Information Processing Systems, NIPS 2013*, 190–198.
[32] Sepp Hochreiter. 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6, 2 (1998), 107–116.
[33] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.
[34] Tianran Hu, Han Guo, Hao Sun, Thuy-vy Thi Nguyen, and Jiebo Luo. 2017. Spice up your chat: the intentions and sentiment effects of using emojis. In *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017*. 102–111.
[35] Diederik P. Kingma and Jimmy Ba. 2014. Adam: a method for stochastic optimization. *CoRR* abs/1412.6980 (2014).
[36] Xiao Ling, Gui-Rong Xue, Wenyuan Dai, Yun Jiang, Qiang Yang, and Yong Yu. 2008. Can Chinese Web pages be classified with English data source?. In *Proceedings of the 17th International Conference on World Wide Web, WWW 2008*. 969–978.
[37] Bing Liu. 2012. *Sentiment analysis and opinion mining*. Morgan & Claypool Publishers.
[38] Qiao Liu, Haibin Zhang, Yifu Zeng, Ziqi Huang, and Zufeng Wu. 2018. Content attention model for aspect based sentiment analysis. In *Proceedings of the 2018 World Wide Web Conference, WWW 2018*. 1023–1032.
[39] Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. 2007. ARSA: a sentiment-aware model for predicting sales performance using blogs. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2007*. 607–614.
[40] Xuan Lu, Wei Ai, Xuanzhe Liu, Qian Li, Ning Wang, Gang Huang, and Qiaozhu Mei. 2016. Learning from the ubiquitous language: an empirical analysis of emoji usage of smartphone users. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2016*. 770–780.
[41] Mary McGlohon, Natalie S. Glance, and Zach Reiter. 2010. Star quality: aggregating reviews to rank products and merchants. In *Proceedings of the Fourth International Conference on Weblogs and Social Media, ICWSM 2010*.
[42] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Computer Science* (2013).

[43] Hannah Jean Miller, Jacob Thebault-Spieker, Shuo Chang, Isaac L. Johnson, Loren G. Terveen, and Brent J. Hecht. 2016. "Blissfully happy" or "ready to fight": varying interpretations of emoji. In *Proceedings of the Tenth International Conference on Web and Social Media, ICWSM 2016*. 259–268.

[44] Saif M. Mohammad, Mohammad Salameh, and Svetlana Kiritchenko. 2016. How translation alters sentiment. *J. Artif. Intell. Res.* 55 (2016), 95–130.

[45] Brendan O'Connor, Ramnath Balasubramanyan, Bryan R. Routledge, and Noah A. Smith. 2010. From Tweets to polls: linking text sentiment to public opinion time series. In *Proceedings of the Fourth International Conference on Weblogs and Social Media, ICWSM 2010*.

[46] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing, EMNLP 2002*. 79–86.

[47] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013*. 1310–1318.

[48] Umashanthi Pavalanathan and Jacob Eisenstein. 2015. Emoticons vs. emojis on Twitter: a causal inference approach. *CoRR* abs/1510.08480 (2015).

[49] J. W. Pennebaker, L. E. Francis, and R. J. Booth. 1999. Linguistic inquiry and word count: LIWC. *Lawrence Erlbaum Associates Mahwah Nj* (1999).

[50] Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL 2010*. 1118–1127.

[51] Michal Ptaszynski, Rafal Rzepka, Kenji Araki, and Yoshio Momouchi. 2012. Automatically annotating a five-billion-word corpus of Japanese blogs for affect and sentiment analysis. In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis, WASSA@ACL 2012*. 89–98.

[52] Guang Qiu, Xiaofei He, Feng Zhang, Yuan Shi, Jiajun Bu, and Chun Chen. 2010. DASA: dissatisfaction-oriented advertising based on sentiment analysis. *Expert Systems with Applications* 37, 9 (2010), 6182–6191.

[53] Koustuv Saha, Larry Chan, Kaya de Barbaro, Gregory D. Abowd, and Munmun De Choudhury. 2017. Inferring mood instability on social media by leveraging ecological momentary assessments. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, IMWUT* 1, 3 (2017), 95:1–95:27.

[54] Jianfeng Si, Arjun Mukherjee, Bing Liu, Qing Li, Huayi Li, and Xiaotie Deng. 2013. Exploiting topic based Twitter sentiment for stock prediction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013*. 24–29.

[55] Lihua Sun, Junpeng Guo, and Yanlin Zhu. 2019. Applying uncertainty theory into the restaurant recommender system based on sentiment analysis of online Chinese reviews. *World Wide Web* 22, 1 (2019), 83–100.

[56] Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. 2010. Sentiment in short strength detection informal text. *JASIST* 61, 12 (2010), 2544–2558.

[57] Fangzhao Wu, Jia Zhang, Zhigang Yuan, Sixing Wu, Yongfeng Huang, and Jun Yan. 2017. Sentence-level sentiment classification with weak supervision. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2017*. 973–976.

[58] Min Xiao and Yuhong Guo. 2013. Semi-supervised representation learning for cross-lingual text classification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013*. 1465–1475.

[59] Jiang Yang, Lada A. Adamic, Mark S. Ackerman, Zhen Wen, and Ching-Yung Lin. 2012. The way I talk to you: sentiment expression in an organizational context. In *Proceedings of the 2012 International Conference on Human Factors in Computing Systems, CHI 2012*.

[60] Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: a survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* 8, 4 (2018).

[61] Joey Tianyi Zhou, Sinno Jialin Pan, Ivor W. Tsang, and Shen-Shyang Ho. 2016. Transfer learning for cross-language text categorization through active correspondences construction. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI 2016*. 2400–2406.

[62] Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. 2016. Cross-lingual sentiment classification with bilingual document representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016*. 1403–1412.

Decoupled Smoothing on Graphs

Alex Chin
Stanford University
ajchin@stanford.edu

Yatong Chen
Stanford University
yatong@stanford.edu

Kristen M. Altenburger
Stanford University
kaltenb@stanford.edu

Johan Ugander
Stanford University
jugander@stanford.edu

ABSTRACT

Graph smoothing methods are an extremely popular family of approaches for semi-supervised learning. The choice of graph used to represent relationships in these learning problems is often a more important decision than the particular algorithm or loss function used, yet this choice is less well-studied in the literature. In this work, we demonstrate that for social networks, the basic friendship graph itself may often not be the appropriate graph for predicting node attributes using graph smoothing. More specifically, standard graph smoothing is designed to harness the social phenomenon of homophily whereby individuals are similar to "the company they keep." We present a *decoupled* approach to graph smoothing that decouples notions of "identity" and "preference," resulting in an alternative social phenomenon of monophily whereby individuals are similar to "the company they're kept in," as observed in recent empirical work. Our model results in a rigorous extension of the Gaussian Markov Random Field (GMRF) models that underlie graph smoothing, interpretable as smoothing on an appropriate auxiliary graph of weighted or unweighted two-hop relationships.

KEYWORDS

Semi-supervised learning; graph smoothing; attribute prediction

ACM Reference Format:
Alex Chin, Yatong Chen, Kristen M. Altenburger, and Johan Ugander. 2019. Decoupled Smoothing on Graphs. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313748

1 INTRODUCTION

Graph-based learning describes a broad class of problems in which response values are observed on a subset of the nodes of a graph, and the learning objective is to infer responses for the unlabeled nodes. Inference methods for graph-based learning nearly unanimously derive their success from an assumption that connected nodes are correlated in their responses, akin to the social phenomenon of homophily whereby "birds of a feather flock together" [22]. Many varieties of models derived from this assumption have been studied and successfully applied. The most popular methods include the work of Zhu, Ghahramani, and Lafferty (ZGL) on semi-supervised learning using Gaussian Markov Random Fields [33], Zhou et al.'s related method for random walk smoothing [31], and Xu, Dyer, and Owen's work connecting the former two methods to methods for kriging from spatial statistics [30]. See [32] for a survey of semi-supervised learning methods on graphs.

While presented as graph-based methods, the graphs that underlie the typical applications of these methods are often synthetic in nature. For example, they may be derived from high-dimensional text or image data. These typical applications begin with a semi-supervised learning problem studying high-dimensional data points $x_i \in \mathbb{R}^D$ associated with response values $y_i \in \mathbb{R}$ (such as images x_i associated with quality scores y_i) and then induce a graph between the data points by taking a k-nearest neighbor graph in the space $\mathbb{R}^D$ to obtain a sparse similarity graph. Despite the synthetic nature of these graphs, graph-based learning methods have been highly effective for solving machine learning problems.

In this work, we focus on graph-based learning problems on social networks, where the relationship between the graph and the response variables can be quite different than the basic similarity relationship between responses on synthetic graphs produced by standard induction methods such as k-nearest neighbors. We rely on the recent observation that there are contexts where the fundamental assumption of correlation that drives graph-based learning—that connected nodes are correlated in their responses—may not be true or necessary for inference to succeed [1, 25]. The work in [1], which focused on the social structure of gender, observed a fundamental difference between similarities with "the company you keep" and "the company you're kept in" in social networks. That work found that the two-hop similarities implied by the latter can exist in the complete absence of any one-hop similarities. In the present work, we develop these differing social assumptions into an alternative semi-supervised learning problem, where correlations on the social networks are based on a decoupling of separate "real" and "target" responses of individuals corresponding to separated notions of social identity and social preference. The key principle is to allow the outcomes of an individual to be possibly different from the outcomes with which the individual prefers to associate.

The consequences of this decoupling between real and target responses is that individuals connected in the graph are only correlated in their responses if the real and target responses of the individuals are correlated with each other. But even if they are entirely uncorrelated, a subtler correlation is nonetheless induced, as pointed out in [1], which can be particularly useful for inference: even if an individual is not correlated with their graph neighbors, the fact that they have preferences for a target response will still imply that their neighbors are correlated with each other. As a

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313748

result, individuals can be similar to their friends-of-friends without necessarily being similar to their friends.

In this work we show that the above decoupling can be modeled in a very natural semi-supervised learning problem for an appropriately constructed auxiliary similarity matrix encoding two-hop similarities. The two-hop similarity matrix that we consider is derived from previously unrelated literature on combining estimators [8, 11, 12, 19, 23, 26], since the "real" and "target" state of each node must be estimated from the dispersed evidence surrounding them in the graph. By making this connection, our work also introduces an iterative procedure for the combining estimators problem when data is being collected on a graph.

As a byproduct of our investigation, we also strengthen the known connections between random walk-based methods (such as ZGL) and Kriging-based methods, introduce a procedure that maps any Kriging problem to a ZGL problem. This connection between the methods enables us to take a "decoupled" view of either problem class, and also allows us to use computationally expedient iterative solvers for standard ZGL-type problems to solve Kriging problems, coupled or decoupled.

2 GRAPH SMOOTHING PRELIMINARIES

In this section we review two standard formulations of graph smoothing, the semi-supervised learning problem of [33], which we refer to here simply as smoothing, and the graph regularization approach involving noisy observations that we refer to as soft smoothing. We review the closed form solutions, and later give recurrences that converge on the closed form solutions for both problems.

We assume that we observe a social network of n people with connections represented by an undirected, unweighted graph $G = (V, E)$. We let A and D denote the adjacency matrix and degree matrix of the graph, respectively, where D has diagonal entries $d_i = \sum_j A_{ij}$. We denote the neighborhood of node i by $\mathcal{N}_i = \{j : A_{ij} > 0\}$. Each person i has a outcome value of interest $\theta_i \in \mathbb{R}$; we write $\theta \in \mathbb{R}^n$ for the vector of response values. We only observe the labels θ_i for a subset of the nodes; we denote the set of labeled nodes V_0 and the set of unlabeled nodes $V_1 = V \setminus V_0$. Let $\mathcal{N}_i^0 = \{j : A_{ij} > 0, j \in V_0\}$ be the set of neighbors of i for which the labels are observed, and $\mathcal{N}_i^1 = \{j : A_{ij} > 0, j \in V_1\}$ be the set of neighbors of i for which the labels are unobserved. Correspondingly define $d_i^0 = |\mathcal{N}_i^0|$ and $d_i^1 = |\mathcal{N}_i^1|$, the "labeled degree" and "unlabeled degree," respectively. The vectors $\theta_0 \in \mathbb{R}^{|V_0|}$ and $\theta_1 \in \mathbb{R}^{|V_1|}$ represent the restriction of the response vector θ to the observed and unobserved node sets. In general, we reserve the subscripts 0 and 1 for subsetting on labeled and unlabeled nodes, respectively.

2.1 Smoothing

The standard formulation of graph smoothing, proposed in [33], is to solve the optimization problem

$$\min_{\theta} \sum_{(i,j)\in E} A_{ij}(\theta_i - \theta_j)^2, \qquad \text{subject to } \theta|_{V_0} = \theta_0. \tag{1}$$

The loss function in Equation (1) is $\theta^\top L\theta$, where $L = D - A$ is the graph Laplacian.

If we define the transition matrix $P = D^{-1}A$ and identify blocks of P according to the labeled nodes V_0 and unlabeled nodes V_1, the closed-form solution to Equation (1) for the unlabeled nodes is then:

$$\hat{\theta}_1 = (I - P_{11})^{-1}P_{10}\theta_0, \text{ where } P = \begin{pmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{pmatrix}. \tag{2}$$

This solution has a Bayesian interpretation [20, 30]. Suppose we place a Gaussian Markov Random Field (GMRF) on the node set by placing a prior $\theta \sim N\left(0, \tau^2(D - \gamma A)^{-1}\right)$ on θ. This prior is the *conditional autoregressive* (CAR) model popular in the spatial statistics literature [5, 6], and has the property that θ_i conditional on the other values of θ follows the distribution

$$\theta_i|(\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_n) \sim N\left(\frac{\gamma}{d_i}\sum_{j\in\mathcal{N}_i}\theta_j, \frac{\tau^2}{d_i}\right). \tag{3}$$

Under this GMRF prior, the Bayes estimator of θ, conditional on having observed the labels θ_i, $i \in V_0$, is the solution to Equation (1), when $\gamma \to 1$. The parameter $\gamma < 1$ is a correlation parameter that is necessary for the distribution to be non-degenerate. In practice it is common to add a small ridge to the diagonal of the Laplacian when solving Equation (1) for numerical stability, which achieves a similar purpose.

2.2 Soft smoothing

A second approach to graph smoothing has its origins in graph-based regularization, as studied in [4]. In this problem, we no longer observe θ_0 directly but rather observe $y_0 = \theta_0 + \varepsilon$ where $\varepsilon \sim N(0, \zeta^2 I)$. We will refer to this problem as soft smoothing, to emphasize the close relationship to ordinary smoothing. Consider the estimator

$$\hat{\theta} = \operatorname*{argmin}_{\theta} \|y_0 - \theta_0\|_2^2 + \lambda\theta^\top(D - A)\theta. \tag{4}$$

Under the prior assumption $\theta \sim N(0, \tau^2(D - \gamma A)^{-1})$, which is equivalent to the conditional specification of the GMRF on the node set in Equation (3), the estimator $\hat{\theta}$ in Equation (4) is the Bayes estimator for this model with $\lambda = \zeta^2/\tau^2$ and $\gamma \to 1$. This estimator $\hat{\theta}$ has the following closed form solution when $\lambda > 0$:

$$\hat{\theta} = (J + \lambda L)^{-1}Jy^*, \text{ where } J = \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix}, \tag{5}$$

and y^* is a vector that agrees with the labeled points and takes on any value on the unlabeled points. If $\zeta^2 = 0$ (whereby $\lambda = 0$) then the optimization problem in Equation (4) reduces to the noiseless graph smoothing problem in Equation (1) [30]. Later, we will show that even when $\zeta^2 > 0$ there is a direct mapping between soft and "hard" smoothing problem instances.

3 DECOUPLED GRAPH SMOOTHING

Crucial to the performance of graph smoothing estimators like Equation (2) is a smoothness assumption positing that the response values vary smoothly across the topology of the graph. This assumption in often appropriate in synthetic graph applications like those constructed for image segmentation and changepoint detection problems, and smooth kernel edge weights are often assumed in theoretical studies of semi-supervised learning performance [13, 24].

However, this assumption can fail to hold in social networks. Indeed, [1] shows that even when homophily is minimal or nonexistent, additional variation known as *monophily* among friend-of-friend (two-hop) relations can be exploited to predict outcomes. Intuitively, if an unlabeled individual i is friends with person j who tends to befriend a high proportion of female individuals, then we might be more certain that the unlabeled individual i is female as well. In this case, what matters for classifying individual i is not the label of individual j, but rather individual j's preferences in her friends. In this work we propose *decoupling* the true parameter of interest θ_j from a *target parameter* ϕ_j that captures the true parameters of the neighbors of j. We now study a model that gives rise to such a decoupling.

Suppose we have a weight matrix W, and denote the row sums by $z_i = \sum_j W_{ij}$ and the column sums by $z'_j = \sum_i W_{ij}$. Our approach is to relate the real θ_i and target ϕ_i via W as follows:

$$\theta_i \approx \frac{1}{z_i} \sum_{j=1}^{n} W_{ij}\phi_j, \tag{6}$$

$$\phi_j \approx \frac{1}{z'_j} \sum_{i=1}^{n} W_{ij}\phi_i. \tag{7}$$

Note that W need not be symmetric. The choice of W is to be set by the researcher and may depend on the particular problem being solved. For example, if W is set to be the adjacency matrix A, then ϕ_i and θ_i become unweighted averages of peer values. Section 5 contains a discussion of alternative options for setting W, where we argue that more complex choices of W may indeed be desirable.

Expressions (6) and (7) are to be understood in an "average" and informal sense. Formally, we consider the Gaussian Markov random field model

$$\theta_i|\phi \sim N\left(\frac{\gamma}{z_i}\sum_{j=1}^{n} W_{ij}\phi_j, \frac{\tau^2}{z_i}\right), \quad \phi_j|\theta \sim N\left(\frac{\gamma}{z'_j}\sum_{i=1}^{n} W_{ij}\theta_i, \frac{\tau^2}{z'_j}\right), \tag{8}$$

where γ and τ^2 are constants. We now establish that this model is equivalent to marginally specifying the joint Gaussian distribution for θ and ϕ as follows. A proof of this equivalence is found in the appendix.

Theorem 3.1. *Let W be a weight matrix with row sums $z_i = \sum_j W_{ij}$ and column sums $z'_j = \sum_i W_{ij}$. Let $\tau^2 > 0$ and $\gamma \in (0, 1)$. Then the conditional specifications*

$$\theta_i|\phi \sim N\left(\frac{\gamma}{z_i}\sum_{j=1}^{n} W_{ij}\phi_j, \frac{\tau^2}{z_i}\right), \qquad \phi_j|\theta \sim N\left(\frac{\gamma}{z'_j}\sum_{i=1}^{n} W_{ij}\theta_i, \frac{\tau^2}{z'_j}\right)$$

define a valid, non-degenerate probability distribution over θ and ϕ with marginal distribution $\begin{pmatrix}\theta\\ \phi\end{pmatrix} \sim N(\mu, \Sigma)$, where $\mu = 0$ and

$$\Sigma = \tau^2 \begin{pmatrix} Z & -\gamma W \\ -\gamma W^\top & Z' \end{pmatrix}^{-1}, \tag{9}$$

where $Z = \mathrm{diag}(z_1, \ldots, z_n)$ and $Z' = \mathrm{diag}(z'_1, \ldots, z'_n)$.

Because our goal is to obtain predictions for the real attributes θ, we view the target attributes ϕ as nuisance parameters and marginalize them out. By studying the precision matrix $M = \Sigma^{-1}$ and applying the standard 2×2 block matrix inversion Schur complement

$$(M^{-1})_{11} = (M_{11} - M_{12}M_{22}^{-1}M_{21})^{-1},$$

we find the marginal prior for θ is then Gaussian with mean 0 and covariance matrix $\tau^2\left(Z - \gamma^2 W Z'^{-1} W^\top\right)^{-1}$. Therefore, minimizing the posterior log-likelihood conditional on observing values θ_i for $i \in V_0$ reduces to the optimization problem

$$\min_\theta \theta^\top L' \theta, \qquad \text{subject to } \theta|_{V_0} = \theta_0 \tag{10}$$

for the modified Laplacian

$$L' = (Z - \gamma^2 W Z'^{-1} W^\top). \tag{11}$$

We call this modified Laplacian the *decoupled Laplacian*, to emphasize the decoupling between the real responses θ and the target responses ϕ in the underlying model.

From this expression for the decoupled Laplacian we can view $\tilde{A} = WZ'^{-1}W^\top$ as a weighted adjacency matrix for an auxiliary graph that is essentially connecting nodes to their two-hop neighbors with appropriately weighted edges. With this modified auxiliary matrix, the solution to the decoupled smoothing objective is then

$$\hat{\theta}_1 = (I - P_{11})^{-1} P_{10}\theta_0, \tag{12}$$

as before in Equation (2), but now with $P = Z^{-1}(Z - \gamma^2 W Z'^{-1} W^\top)$. The closed form solution to the soft decoupled smoothing objective, as in Equation (5), becomes

$$\hat{\theta} = (J + \lambda(Z - \gamma^2 W Z'^{-1} W^\top))^{-1} J y^*. \tag{13}$$

We have intentionally kept the weight matrix W generic in these derivations. In Section 5, we use ideas from the literature on expert aggregation to motivate our choice of W.

3.1 Coupling $\theta = \phi$ reduces to the standard smoother

Here we further motivate the sense in which the above decoupled smoothing problem is strongly connected to the basic smoothing problem: by conditioning the distribution of the GMRF upon $\theta = \phi$, we recover the basic smoothing objective.

Under the transformation $\nu = \theta - \phi$, the log-likelihood of the vector (θ, ν) is

$$p(\theta, \nu) \propto \exp\Big\{ -\frac{1}{2\tau^2}\Big(\theta^\top(Z + Z' - 2\gamma W)\theta - 2\theta^\top(Z' - \gamma W)\nu + \nu^\top Z'\nu\Big)\Big\}.$$

If we let the weight matrix W be equal to the adjacency matrix A, so that the row and column sums reduce to the degree matrix $Z = Z' = D$, then the joint parameters (θ, ν) have the covariance matrix

$$\tau^2 \begin{pmatrix} 2(D - \gamma A) & -(D - \gamma A) \\ -(D - \gamma A) & D \end{pmatrix}^{-1}.$$

The upper left block, $2\tau^2(D - A)$, is the precision matrix of θ conditional on ν, which shows that θ follows the ordinary smoother distribution in Section 2.1 with τ^2 replaced by $2\tau^2$.

4 ITERATIVE PERSPECTIVE ON SMOOTHING

In this section we outline how the closed form solutions to the smoothing problems discussed in this work can be formulated as the solutions to the iterative application of recurrence relations. We first review the known iterative formulation of smoothing. We then contribute a reduction which shows that any *soft* smoothing problem can be written as a *hard* smoothing problem, and can thus be written iteratively as well. We formulate the recurrence relation that underlies the decoupled smoothing problem studied in this work. In the next section, Section 5, we will show how this recurrence can be interpreted in the language of expert opinion aggregation, giving us an intuition for how to choose the previously unspecified weight matrix W in the recurrence we derive here.

4.1 Iterative formulation of smoothing

The closed form solution to the smoothing objective in Equation (1) is known to arise from a repeated application of majority vote [7, 21] in the following sense: define the time 0 estimate $\hat{\theta}^0$ to agree with the true labels on V_0. Take the transition matrix $P = D^{-1}A$ and perform the updates

$$\hat{\theta}_1^t = P_{10}\theta_0 + P_{11}\hat{\theta}_1^{t-1}, \qquad \hat{\theta}_0^t = \theta_0, \tag{14}$$

where $P = \begin{pmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{pmatrix}$ has been partitioned into labeled and unlabeled blocks, as before. In other words, the time t estimate is the majority vote estimate using the time $t-1$ predictions, where after each step we replace the labeled predictions by their original, true labels. In the limit,

$$\hat{\theta}_1 = \lim_{t\to\infty} \hat{\theta}_1^t = (I - P_{11})^{-1}P_{10}\theta_0, \tag{15}$$

which is the solution to Equation (1) given in Equation (2).

4.2 Iterative formulation of soft smoothing

We now contribute an iterative algorithm for soft smoothing by showing that a soft smoothing solution is equivalent to a hard smoothing solution on an appropriate auxiliary graph. Consider an augmented graph $\tilde{G}$ formed by attaching a degree-one node to each labeled node on the original graph, with edge weight $\alpha > 0$. The vertex set of the resulting graph contains a copy of the labeled nodes, denoted $\tilde{V}_0$, in addition to the original labeled node set V_0 and unlabeled node set V_1.

This new graph has adjacency matrix

$$\tilde{A} = \begin{pmatrix} 0 & \alpha I & 0 \\ \alpha I & A_{00} & A_{01} \\ 0 & A_{10} & A_{11} \end{pmatrix},$$

where the first block corresponds to $\tilde{V}_0$, the second block corresponds to V_0, and the third block corresponds to V_1. If we compute the hard smoothing solution on this augmented graph treating $\tilde{V}_0$ as the labeled set and the entire original vertex set V as the unlabeled set, the corresponding random walk transition matrix is

$$\tilde{P} = \tilde{D}^{-1}\tilde{A} = \begin{pmatrix} 0 & I & 0 \\ \tilde{D}_0^{-1}A_{10} & \tilde{D}_0^{-1}A_{00} & \alpha\tilde{D}_0^{-1} \\ D_1^{-1}A_{10} & D_1^{-1}A_{11} & 0 \end{pmatrix},$$

where D_1 is the degree matrix of the original graph restricted to the unlabeled node set V_1, and $\tilde{D}_0 = D_0 + \alpha I$ is the degree matrix restricted to the labeled node set V_0, adjusted to account for the new auxiliary nodes.

The hard smoothing solution in Equation (15) is $\hat{\theta} = (I - P_{11})^{-1}P_{10}\theta_0$. Applying this solution to the above problem, we have

$$\begin{aligned} P_{11} &= (D + \alpha J)^{-1}A, \\ P_{10}\theta_0 &= \alpha(D_0 + \alpha I)^{-1}\theta_0 = \alpha(D + \alpha J)^{-1}Jy^*, \end{aligned}$$

which results in the estimator

$$\begin{aligned} \hat{\theta} &= (I - (D + \alpha J)^{-1}A)^{-1}\alpha(D + \alpha J)^{-1}Jy^* \\ &= (\alpha J + D - A)^{-1}\alpha Jy^* \\ &= (J + \alpha^{-1}L)^{-1}Jy^*. \end{aligned}$$

This estimator is precisely the soft smoothing closed-form solution in Equation (5) when $\alpha = \lambda^{-1}$. Notice the non-equivalence between this solution and the solution in the case of $\zeta^2 = 0$ (for which $\lambda = 0$).

For soft smoothing, the iterative update, Equation (14), then amounts to

$$\begin{aligned} \hat{\theta}^t &= \alpha(D + \alpha J)^{-1}Jy^* + (D + \alpha J)^{-1}A\hat{\theta}^{t-1} \\ &= (D + \alpha J)^{-1}(\alpha Jy^* + A\hat{\theta}^{t-1}), \end{aligned} \tag{16}$$

where $D + \alpha J$ is diagonal and therefore easily invertible, making for an expedient computational procedure for solving soft smoothing problems. This interpretation of soft smoothing connects to a related used of "shadow nodes" in the *adsorption* method of graph smoothing [3].

4.3 Iterative formulation of decoupled smoothing

Examining the decoupled Laplacian in Equation (11) alongside the iterative smoothing formulation provides an iterative algorithm for the decoupled smoother. We define an auxiliary weighted, directed graph with weighted adjacency matrix $\tilde{A} = WZ'^{-1}W^\top$, which has edge weight $\tilde{A}_{ij} = \sum_k \frac{W_{ik}W_{jk}}{z'_k}$. The out-degree of node i reduces to $\sum_j \tilde{A}_{ij} = z_i$, where z_i is the same row sum defined in Section 3. Hence the degree matrix of $\tilde{A}$ is Z, and the solution to the decoupled smoothing problem in Equation (10) results from performing the iterative one-hop majority vote updates, Equation (14), on the auxiliary, directed graph.

By employing the update equations in Equation (14) with the transition matrix $P = Z^{-1}WZ'^{-1}W^\top$, we can see that decoupled smoothing amounts to an iterative update of a weighted *two-hop* majority vote. The reduction from soft smoothing to ordinary smoothing in Section 4.2 also gives an iterative formulation of soft decoupled smoothing.

4.4 Improving majority vote with local regularization

The iterative perspective is not only useful for computational purposes but also gives insights into how to improve the basic iterated majority vote. Here we describe an improvement to the basic smoothing algorithm, inspired by the details of implementing the

iterative algorithm, which can be applied in either the standard, soft, or decoupled setting.

Limiting noise in early steps. Since iterative majority vote is recursively defined, it relies on defining an initial set of guesses for the unlabeled nodes; when $t = 1$, equation (14) requires a value for $\hat{\theta}_1^0$ which can be safely set to random initial labels without compromising the limiting result. Then, equation (14) can also be written element-wise as

$$\hat{\theta}_i^t = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \hat{\theta}_j^{t-1}. \tag{17}$$

for every unlabeled node $i \in V_1$. From here, one sees that the performance of the first few iterations can be quite unsatisfactory, because it depends strongly on the initial noise input $\hat{\theta}_1^0$.

An alternative strategy is to set the first iteration of the unlabeled nodes to be the average value of friends labelled at the previous time-step only. For $t = 1$ this technique gives:

$$\hat{\theta}_i^1 = \frac{1}{|\mathcal{N}_i^0|} \sum_{j \in \mathcal{N}_i^0} \theta_j, \tag{18}$$

for $i \in V_1$. Recall that $\mathcal{N}_i^0$ is the set of neighbors of i for which a label was observed. For nodes $i \in V_1$ with no labelled neighbors, we do not update $\hat{\theta}_i^1$. More generally, we can postpone making assignments $\hat{\theta}_i^t$ until a node has some neighbor with an estimate at time $t-1$. If using $\hat{\theta}^t$ to make predictions, for missing values we then guess randomly with the class proportions of the training sample.

This strategy, which sidesteps dependence on random initial $\hat{\theta}_1^0$, tends to lead to a slight bump in performance in early iterations; see Section 6 for example illustrations.

Local regularization towards true labels. We can further generalize this idea of upweighting the true labels when they should be trusted more than haphazard (random) guesses. Consider the convex combination update

$$\hat{\theta}_i^t = \lambda_i^t \frac{1}{d_i^0} \sum_{j \in N_i^0} \theta_j + (1 - \lambda_i^t) \frac{1}{d_i^1} \sum_{j \in N_i^1} \hat{\theta}_j^{t-1}, \tag{19}$$

where $\lambda_i^t \in [0, 1]$ are weight parameters that control the amount of trust to place in the guesses of previous iterations. This places weight λ_i^t on the true labels and weight $1 - \lambda_i^t$ on the predicted values for iteration $t-1$. Most generally λ_i^t may be indexed by both the unit i and the time step t, as it is reasonable to expect that this weight should be personalized to individuals (e.g., vary based on degree) and that estimates of later iterations should be trusted more (which would have λ_i^t decreasing in time t).

Decomposing the sum in equation (17) as

$$\hat{\theta}_i^t = \frac{1}{d_i} \left[\sum_{j \in \mathcal{N}_i^0} \theta_j + \sum_{j \in \mathcal{N}_i^1} \hat{\theta}_j^{t-1} \right],$$

we see that we recover equation (19) from equation (17) when $\lambda_i^t = d_i^0 / d_i$, which is constant in t.

The search space of weights λ_i^t is quite large and we leave a formal analysis of this space to future work, restricting ourselves here to providing intuition for choices of λ_i^t that appear to work well in our empirical experiments. The goal is to place more weight on labeled nodes in the early stages and less weight on labeled nodes at later iterations, which suggests λ_i^t decaying in t. Different choices of λ_i^t will lead to different limiting values $\lim_{t \to \infty} \hat{\theta}^t$, some of which appear to outperform the basic version of majority vote.

Consider parameterizing $\lambda_i^t = f_i(t)$ for a function $f_i(\cdot)$ that reduces the number of parameters. For example one may consider the choice $\lambda_i^t = (d_i^0 / d_i)^t$, which represents exponential decay in t. This extension can be written in matrix-vector form as follows. Let $D_1 = \text{diag}(d_i^1)$ and $D_0 = \text{diag}(d_i^0)$ be diagonal degree matrices restricted to the unlabeled and labeled node sets, respectively. Consider the transition matrices $T_{10} = (D_1^{-1} A)_{10}$ and $T_{11} = (D_0^{-1} A)_{11}$. Then we may write the update rule as

$$\hat{\theta}_1^t = \lambda_t T_{10} \theta_0 + (1 - \lambda_t) T_{11} \hat{\theta}_1^{t-1},$$

where $\lambda_t \in [0, 1]^U$ is a vector of penalty parameters for time t. Depending on the form of λ_t, this recursion may not reduce cleanly to a power series that can be written in closed form.

This approach is a form of regularization in the sense that it constrains the complexity of the fitting function θ and restricts how far it can deviate from the observed values θ_0.

The choice of weight $\lambda_i^t = (d_i^0 / d_i)^t$ is related to the sequence of weights commonly used on the popular personalized PageRank diffusion kernel for community detection [2], although it is used in a different way. While personalized PageRank uses the kernel to model decaying influence as a function of graph distance, here we use this kernel to define how much to regularize the solution towards labeled values as a function of time, thus representing the amount of trust placed in labeled nodes. The limiting solution, therefore, will not be the same as the personalized PageRank diffusion, but this conceptual link can be instructive for suggesting analogs of other diffusion kernels that may be used as well [17, 18]. While the discussion of this regularization has centered on standard (one-hop) graph smoothing, the same ideas can be applied to decoupled smoothing as well.

5 AGGREGATING EXPERT INFORMATION

In this section we suggest possible ways of selecting the weights W for the method described in Section 3. Recall that in the decoupled smoothing model, equation (8), we view the target attributes ϕ_j as capturing information from the true responses in the neighborhood $\mathcal{N}_j$, which are useful for estimating θ_i. As a thought experiment, supposing the target parameters ϕ are known. Then the conditional mean (when $\gamma = 1$) is

$$\mathbf{E}[\theta_i | \phi] = \frac{1}{z_i} \sum_{j=1}^{n} W_{ij} \phi_j. \tag{20}$$

Most simply, it may be reasonable in many cases to take $W = A$ to agree with the network structure so that $\hat{\theta}_i$ is just an unweighted average of the estimated target parameters of the neighbors unit i. More generally, however, the row $\{W_{ij} : j = 1, \ldots, n\}$ of weights should be chosen in a way that highlights the nodes j where ϕ_j is most useful for prediction θ_i. In many graph contexts there are highly-skewed degree distributions, and as a result we expect some friends of i to provide more accurate estimates than others of i's true state. Notice that W_{ij} may be asymmetric in general because

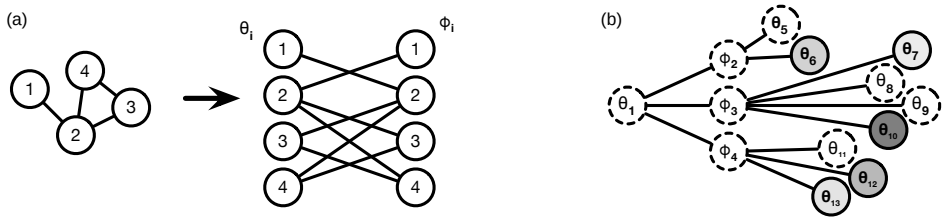


Figure 1: (a) An illustration of how an undirected graph can be decoupled into a bipartite graph with real responses θ_i and target responses ϕ_i. (b) An illustration of how two hop information can be used to estimate the expert opinions ϕ_j, which in turn inform the estimate of the unknown value θ_1.

the information provided by ϕ_i for estimating θ_j may not be the same as the information provided by ϕ_j for estimating θ_i, which is one of the key insights of the decoupled approach.

In this section we discuss how one may use an otherwise unrelated literature on expert opinion aggregation in the graph smoothing context. In this discussion we assume W has the same sparsity pattern as the adjacency matrix A, so that $W_{ij} \neq 0$ whenever $A_{ij} \neq 0$, although this need not generally be true; sometimes *not* being friends with someone can be highly informative as well.

The overall procedure of how a network is converted into a decoupled graph of real and target responses is illustrated schematically in Figure 1(a), and a schematic illustration of how the expert opinion aggregation pulls in information to a single node is illustrated in Figure 1(b).

5.1 Combining independent estimators

Consider that the information contributed by each expert (friend) j for estimating θ_i is in the form of the "observations" $\{\theta_k : k \in \mathcal{N}_j\}$, which are values located two steps away from unit i. One way to think about combining this information has been studied extensively in the statistics literature in the context of estimating a common location parameter from samples of varying precision [8, 11, 12, 14, 19, 23, 26].

Explicitly, suppose the variables in the set $\{\theta_k : k \in \mathcal{N}_j\}$ follow a distribution with mean θ_i and variance σ_j. That is, all observations contribute unbiased information for estimating θ_i, but they have varying precisions which are modulated by unit j (the "expert"). Then the weight matrix entry reduces to $W_{ij} = A_{ij}/\sigma_j^2$, with row sum $z_i = \sum_{\ell \in \mathcal{N}_i} \sigma_\ell^{-2}$ and column sum $z'_j = d_j/\sigma_j^2$. In this case, we now show that we obtain a concise recurrence recognizable as a particular weighting of 2-hop majority vote.

From Section 3, the auxiliary graph with this diagonal covariance specification has an adjacency matrix with entries

$$\tilde{A}_{ij} = \sum_k A_{ik}A_{jk}/(d_k\sigma_k^2), \tag{21}$$

with the smoothing update rule being $\hat{\theta}^t = Z^{-1}\tilde{A}\hat{\theta}^{t-1}$. For an unlabeled node i, if we employ the weights derived here we obtain the recurrence:

$$\begin{aligned}\hat{\theta}_i^t &= (Z^{-1}\tilde{A}\hat{\theta}^{t-1})_i = \frac{1}{z_i}\sum_{j=1}^n \tilde{A}_{ij}\hat{\theta}_j^{t-1} \\ &= \frac{1}{\sum_{\ell\in\mathcal{N}_i}\sigma_\ell^{-2}} \sum_{k\in\mathcal{N}_i}\frac{1}{d_k\sigma_k^2}\sum_{j\in\mathcal{N}_k}\hat{\theta}_j^{t-1}. \end{aligned}\tag{22}$$

By viewing the aggregation as performed on a graph, we can in fact turn this standard estimation procedure into an iterative procedure, linking to smoothing as shown in Section 4.

As a generic problem of aggregating estimators, if we observe $X_{jk} \sim N(\theta, \zeta_j^2)$, $k = 1, \ldots, d_j$, then the minimum variance, linear unbiased estimator (MVLUE) of θ when the ζ_j^2 are known is $\hat{\theta} = \sum_j w_j \bar{X}_j$ with weights given by $w_j = (d_j/\zeta_j^2)/\sum_k (d_k/\zeta_k^2)$. Our formulation of expert aggregation aligns with this view, where the expert variances are $\sigma^2 = \zeta_i^2/d_i$ and higher degree nodes therefore having appropriately more precise information.

In order to estimate σ_j^2 we can notice that it essentially represents the standard error for the expert estimate. Hence we can use the regular standard error estimate for the Gaussian sample mean, $\hat{\sigma}_j^2 = S_j^2/d_j$, where

$$S_j^2 = \frac{1}{d_j^0}\sum_{k\in\mathcal{N}_j^0}\left(\theta_k - \frac{1}{d_j^0}\sum_{\ell\in\mathcal{N}_j^0}\theta_\ell\right)^2$$

is the sample variance of the labeled nodes in the neighborhood of j (recall that $\mathcal{N}_j^0$ is the labeled neighborhood and $d_j^0 = |\mathcal{N}_j^0|$ is the labeled degree). We then use $\hat{\sigma}_j^2$ as a plugin estimate in the update rule in Equation (22), giving

$$\hat{\theta}_i^t = \frac{1}{\sum_{\ell\in\mathcal{N}_i}(S_\ell^2/d_\ell)^{-1}}\sum_{k\in\mathcal{N}_i}\frac{1}{S_k^2}\sum_{j\in\mathcal{N}_k}\hat{\theta}_j^{t-1}. \tag{23}$$

Alternatively, we can directly impose homogeneous standard errors, $\sigma_i^2 = \sigma^2/d_i$, in which case the normalization term reduces to $1/\sum_{\ell\in\mathcal{N}_i}\sigma_\ell^{-2} = 1/\sum_{\ell\in\mathcal{N}_i} d_\ell$, the number of nodes in the two-step neighborhood of i, and we obtain the update rule

$$\hat{\theta}_i^t = \frac{1}{\sum_{\ell\in\mathcal{N}_i} d_\ell}\sum_{k\in\mathcal{N}_i}\sum_{j\in\mathcal{N}_k}\hat{\theta}_j^{t-1}. \tag{24}$$

For exposition here we have let d_ℓ represent the total graph degree of unit ℓ, which disregards the number of labeled nodes. The shortcomings of this choice discussed in Section 4.4 still hold, and the improvements discussed in that section can be applied here.

We thus see how iterating a simple two-hop majority vote update can be motivated for graph smoothing, despite initial appearances as defining a "non-physical" process whereby information bypasses individuals. The simple recurrence in equation (24) emerges as the MVLUE under the assumption that expert friends contribute independent opinions, an assumption which appears to be reasonable for the graph-based learning problems we study.

5.2 Risk analysis and Bayesian interpretation

This idea of combining information from different expert sources is also related to methods developed in the risk analysis literature [10]. Here we discuss graph analogs of the model and analysis of Halperin [15], later derived independently by Winkler [29]. In this model, expert opinions ϕ_j are viewed as unbiased samples from the true unknown response θ_i. That is, the expert predictions have the form

$$\phi_{\mathcal{N}_i} = \theta_i \mathbf{1} + \varepsilon_{\mathcal{N}_i},$$

where $\phi_{\mathcal{N}_i} \in \mathbb{R}^{d_i}$ is the restriction of the target vector ϕ to the neighborhood $\mathcal{N}_i$, $\mathbf{1}$ is the ones vector of length d_i, and $\varepsilon_{\mathcal{N}_i} \sim N(0, \Sigma_{\mathcal{N}_i})$ is Gaussian noise. The covariance matrix $\Sigma_{\mathcal{N}_i} \in \mathbb{R}^{d_i \times d_i}$ captures the variance and dependencies among the expert opinions, and is a generalization of the discussion in Section 5.1 in that expert opinions may be correlated with each other. Under this model and a flat prior on θ_i, the posterior distribution of θ_i given $\phi_{\mathcal{N}_i}$ with known covariance $\Sigma_{\mathcal{N}_i}$ is given by

$$\theta_i | \phi_{\mathcal{N}_i} \sim N\left(\frac{\mathbf{1}^\top \Sigma_{\mathcal{N}_i}^{-1} \phi_{\mathcal{N}_i}}{\mathbf{1}^\top \Sigma_{\mathcal{N}_i}^{-1} \mathbf{1}}, \frac{1}{\mathbf{1}^\top \Sigma_{\mathcal{N}_i}^{-1} \mathbf{1}} \right).$$

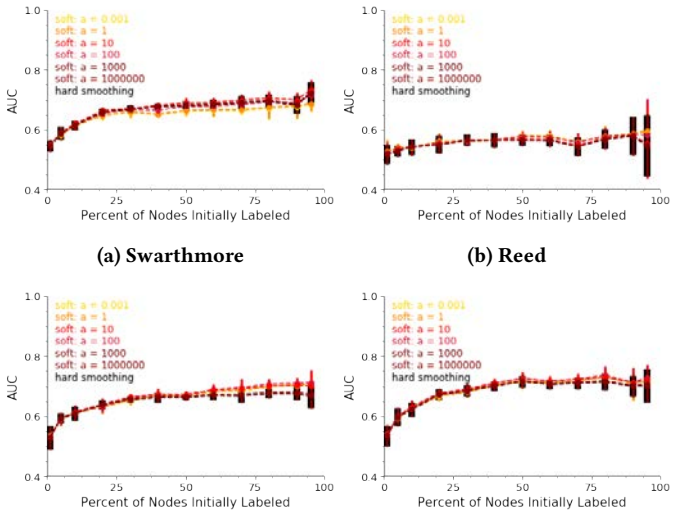
The Bayes estimator is then the posterior mean,

$$\hat{\theta} = (\mathbf{1}^\top \Sigma_{\mathcal{N}_i}^{-1} \phi_{\mathcal{N}_i}) / (\mathbf{1}^\top \Sigma_{\mathcal{N}_i}^{-1} \mathbf{1}).$$

From comparing to the conditional mean in Equation (20), the Bayes estimator suggests taking the weight W_{ij} to be element of the vector $(\mathbf{1}^\top \Sigma_{\mathcal{N}_i}^{-1})$ that corresponds to node j if i is connected to j, and $W_{ij} = 0$ otherwise. The normalization $z_i = \sum_{j \in \mathcal{N}_i} W_{ij} = \mathbf{1}^\top \Sigma_{\mathcal{N}_i}^{-1} \mathbf{1}$ then falls out naturally.

More general covariance structures. A principal outstanding question is determining which assumptions to place on each $\Sigma_{\mathcal{N}_i}$, which are unknown in practice. We notice that determining the $\Sigma_{\mathcal{N}_i}$ is equivalent to defining a single covariance matrix for the expert opinions $\Sigma \in \mathbb{R}^{n \times n}$ for the entire graph, where Σ_{jk} represents the covariance between the expert opinions provided by individuals j and k. The weight matrix results by taking $W = A\Sigma^{-1}$. The specification of the covariance matrix Σ is crucial for the form of the resulting decoupled smoothing estimator.

In practice on social networks, dependencies between expert opinions are bound to occur due to intersecting friendship neighborhoods. if the estimation target is i, and two neighbors j and k of i share many mutual friends other than i, then their estimates of θ_i will be dependent. The presence of such correlations indicates that aggregation and smoothing methods that are able to harness these arbitrary covariance matrices may yield a powerful tool for node classification in some application areas, and we leave a detailed study of such extensions to future work.



(a) Swarthmore (b) Reed

(c) Amherst (d) Haverford

Figure 2: Predicting gender based on soft smoothing. Horizontal axis displays initial proportion of unlabeled nodes and vertical axis displays AUC scores. Soft smoothing is essentially identical in performance to hard smoothing for the gender classification task.

6 EMPIRICAL ILLUSTRATIONS

We perform experiments on a sample of undergraduate college networks collected from a single-day snapshot of Facebook in September 2005 [27, 28]. We focus on the task of gender classification in these networks, restricting our analyses to the subset of nodes that self-reported their gender to the platform. We use the largest connected components from four medium-sized colleges, `Amherst`, `Reed`, `Haverford`, and `Swarthmore`. Amherst has 2032 nodes and 78733 edges, Reed has 962 nodes and 18812 edges, Haverford has 1350 nodes and 53904 edges, and Swarthmore has 1517 nodes and 53725 edges.

We varied the percentage of initially labelled nodes by selecting a labelled sample uniformly at random. We trained our models varying the percentage of initially labelled nodes in the network. For a given fixed percent of labelled individuals (training dataset), we measure classification performance (by AUC) on the remaining unlabelled nodes (testing dataset), using the same train/test splits across the different inference methods. For all plots in this section we attempt classification 10 times based on different independent labelled subsets of nodes using stratified sampling (stratified based on gender). We show the average AUC with error bars denoting the standard deviation across the 10 runs. We also provide results for synthetic networks generated from the stochastic block model [16] and the overdispersed stochastic block model [1].

6.1 Soft smoothing

We first run our soft smoothing algorithm on the four Facebook networks. As shown in Section 4, soft smoothing is equivalent to hard smoothing on a modified graph in which each originally labeled node has its label removed and is instead connected to a new degree-one node via an edge with weight $\alpha > 0$. Here we

choose a range of different α values to study its influence on the performance of soft smoothing method. The baseline algorithm is the hard smoothing approach of [33], which is the solution to Equation (15). Observe that α quantifies the impact of the new attached nodes on the original labeled nodes; therefore, the larger the value of α, the closer the soft smoothing result resembles the hard smoothing result.

In our experiments we observe that the outputs from soft smoothing are essentially identical to those from hard smoothing. As Figure 2 shows, since our goal is to classify gender, which is self-reported as a binary variable, adding a "noise" term to the labeled nodes has relatively little effect. Our hypothesis is that the soft smoothing method may achieve better performance when applied to real-valued outcomes.

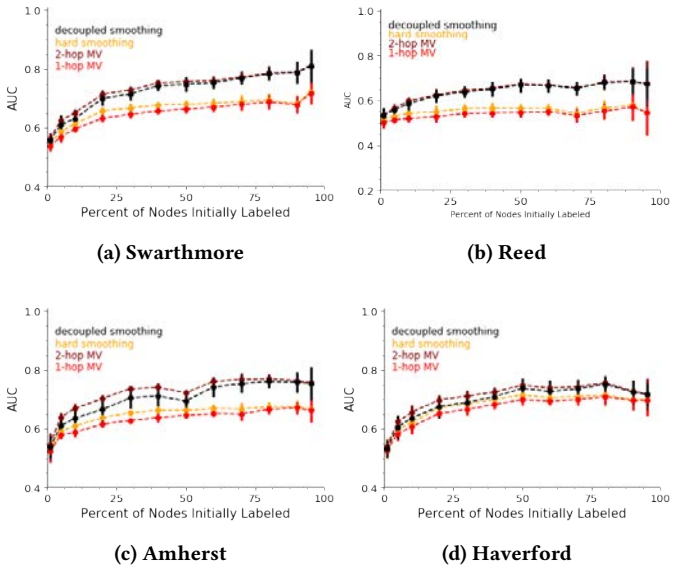
6.2 Decoupled smoothing

In Figure 3 we see our experiments with decoupled smoothing, which indicate that the two-hop majority vote update given by Equation (24) outperforms both the standard 1-hop majority vote estimator and the corresponding (ZGL) smoothing estimator in terms of AUC, regardless of the percentage of initially labeled nodes. Meanwhile we also observe that decoupled smoothing performs slightly worse than the much simpler 2-hop majority vote estimator in some situations (namely `Amherst` and `Haverford`). Recall from Section 4.4 that decoupled smoothing can be interpreted as iterated 2-hop majority vote, but with randomly initialized guesses. We suspect that the better performance of the plain 2-hop majority vote is due to the fact that local information is more pertinent for this particular task (gender prediction) than global information, and the smoothing algorithms are inappropriately synthesizing information from local and global sources. This hypothesis motivates the improvements discussed in Section 4.4, the results for which are discussed in the following section.

6.3 Regularized iterations

In Section 4.4 we considered a modified iterated majority vote algorithm that includes a local regularization $\lambda_i^t = (d_i^L/d_i)^t$ for each unlabeled node i. This modification was inspired by the empirical observation that 2-hop majority vote outperforms the limiting iterated smoother. As a secondary inspiration, using equation (18) as the first iteration's update rule instead of (17) greatly reduces the number of iterations needed for convergence. In this section, we present experimental results from applying these modifications for both hard smoothing and decoupled smoothing on a synthetic stochastic block model graph as well as the on the Facebook networks.

6.3.1 Improved iterative hard smoothing. We first display results on a graph sampled from a stochastic block model (SBM). The particular SBM we use has two blocks with 500 nodes in each block, representing 500 male and 500 female individuals. The expected average degree is 42, so as to achieve the same edge density as used in [1]. In Figure 4 we plot the AUC corresponding to the predictions given by equation (14) for each iteration, $t = 1, 2, 3, \ldots$. Interestingly, in Figure 4(a), we see that some early iterations (in this case, iteration 2) actually outperform the limiting ZGL/hard smoothing solution. This suggests that improvements can be made



(a) Swarthmore **(b) Reed**

(c) Amherst **(d) Haverford**

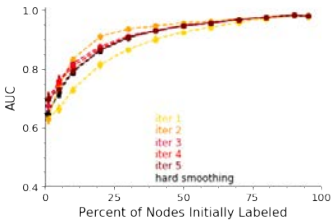
Figure 3: Predicting gender based on decoupled smoothing, compared with hard smoothing (ZGL) and 1-hop and 2-hop majority vote. The estimators based on two-step neighborhood information clearly outperform those based on one-step information, but 2-hop majority vote sometimes outperforms decoupled smoothing.

by teaching the classifier to hone in on these successful iterations and give them more weight, which is what the regularizer seeks to do. We next compare the original and regularized versions of the hard smoothing iterations. As shown in Figure 4(b), the regularization proposed in Section 4.4 improves the overall prediction accuracy for hard smoothing under the stochastic block model. The method is essentially converged after five iterations.

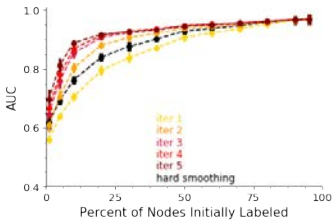
Results are more mixed on empirical networks; in Figure 4(c-d) we display performance for the `Amherst` Facebook network. We see that regularization does not significantly improve the performance over hard smoothing for gender prediction on this network.

6.3.2 Improved iterative decoupled smoothing. Our regularization ideas can also be applied to the decoupled smoother. We first test our modification in an overdispersed stochastic block model (oSBM), an extension of the stochastic blockmodel that contains an additional parameter to model *monophily*. It is thus designed to capture aspects of the network that are particularly well suited for 2-hop estimators. Again, we use two blocks with 500 nodes in each block representing 500 males and 500 females. The expected average degree is 42 and dispersion rate is 0.004, giving the same edge density and dispersion rate as in [1]. Here we compare the iterative method results for the original decoupled smoothing method against the regularized iterative decoupled smoothing method. As shown in Figure 5b, the regularization improves the overall prediction accuracy for decoupled smoothing under the overdispersed stochastic block model.

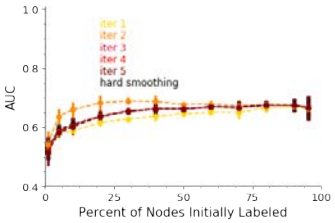
On the Facebook `Amherst` network we use the regularization $\lambda_i^t = (d_i^0/(rd_i))^{t-1}$, where r is the initial percent of labeled nodes.



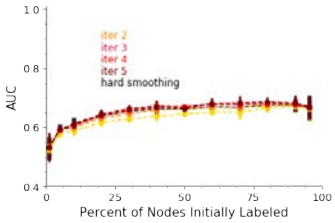
(a) SBM: hard smoothing



(b) SBM: regularized hard smoothing

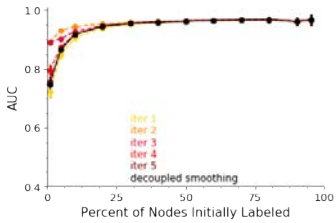


(c) Amherst: hard smoothing

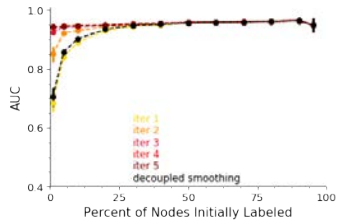


(d) Amherst: regularized hard smoothing

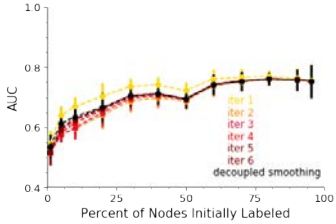
Figure 4: Hard smoothing iterations, with and without regularization, for classifying gender on an SBM graph and the `Amherst` dataset. Regularization mostly improves the predictive performance.



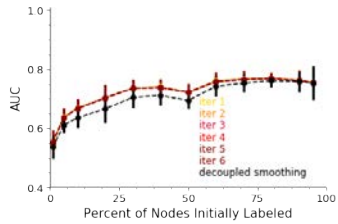
(a) oSBM: decoupled smoothing



(b) oSBM: regularized decoupled smoothing



(c) Amherst: decoupled smoothing



(d) Amherst: regularized decoupled smoothing

Figure 5: Decoupled smoothing iterations, with and without regularization, for classifying gender on an oSBM and the `Amherst` network. Regularization mostly improves the predictive performance.

This choice is motivated by the fact that the relative importance of local to global information should depend on the proportion of labeled nodes; if there is little local information available, then it makes sense to pull in information from farther away. We can see that with this particular regularization term, the smoother modestly improves the overall prediction accuracy for decoupled smoothing. It is encouraging that from pure intuition we can see better results; with a more careful optimization over the way λ_i^t is parameterized, it is possible that performance can be further improved.

7 DISCUSSION

In this work we investigate the use of graph-based smoothing for node attribute prediction on social networks, where a thoughtful understanding of social forces that underlie network formation can help inform the choice of the smoothing model. Our work is motivated by the investigation into empirical observations in [1], which highlights the distinction between "the company you keep" and "the company you're kept in." We develop a model for what we call *decoupled* graph smoothing that links this empirical observation to graph smoothing, semi-supervised learning, and diffusion algorithms popularly used for node classification tasks. We provide a Bayesian viewpoint of this model which is related to the literature on expert opinion aggregation.

As a part of our analysis, we contribute an iterative algorithm for soft smoothing, which allows us to solve soft smoothing problems efficiently on large datasets. We find that a close examination into the form of iteration is not only crucial for computational efficiency but also for efficacy of predictive performance, as the basic iterated majority vote algorithms make suboptimal choices in the initial iterations. We contribute a generalization that allows one to place greater weight on and regularize toward labeled values. This method displays improved performance on some simulated and real datasets. This generalization is flexible enough that the practitioner has a lot of control over the resulting algorithm. The best choices for regularization has yet to be fully explored and may well vary depending on the particular domain of application.

Graph-based semi-supervised learning is broadly applied to classification and prediction problems on derived graphs, typically from the nearest-neighbor graphs of point clouds in $\mathbb{R}^n$. In this work, we observe that classification and prediction problems involving explicit graph structure, such as predicting node attributes in social networks, can benefit from a careful consideration of how the prediction target quantity and the graph structure may be related. While our evaluation only touches gender, other attribute prediction tasks in social networks such as predicting abusive or spam accounts could also benefit from similar careful consideration. While our empirical improvements for gender are modest, our results on synthetic graphs suggest that decoupled graph smoothing has the potential to offer considerable improvements in other diverse network prediction tasks.

Code availability. Notebooks and code available online: https://github.com/YatongChen/decoupled_smoothing_on_graphs. All code was run using Python version 3.6.2.

Acknowledgements. This work was supported in part by NSF grant IIS-1657104 and an ARO Young Investigator Award. KMA was supported in part by a National Defense Science and Engineering Graduate Fellowship.

REFERENCES

[1] Kristen M Altenburger and Johan Ugander. 2018. Monophily in social networks introduces similarity among friends-of-friends. *Nature Human Behaviour* 2, 4 (2018), 284.

[2] Reid Andersen, Fan Chung, and Kevin Lang. 2006. Local graph partitioning using PageRank vectors. In *null*. IEEE, 475–486.

[3] Shumeet Baluja, Rohan Seth, D Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. 2008. Video suggestion and discovery for YouTube: taking random walks through the view graph. In *Proceedings of the 17th International Conference on World Wide Web*. ACM, 895–904.

[4] Mikhail Belkin, Irina Matveeva, and Partha Niyogi. 2004. Regularization and semi-supervised learning on large graphs. In *International Conference on Computational Learning Theory*. Springer, 624–638.

[5] Julian Besag. 1974. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)* (1974), 192–236.

[6] Julian Besag, Jeremy York, Annie Mollié, et al. 1991. Bayesian image restoration, with two applications in spatial statistics. *Annals of the Institute of Statistical Mathematics* 43, 1 (1991), 1–20.

[7] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. 2011. Node classification in social networks. In *Social Network Data Analytics*. Springer, 115–148.

[8] Leo Breiman. 1996. Stacked regressions. *Machine Learning* 24, 1 (1996), 49–64.

[9] D Brook. 1964. On the distinction between the conditional probability and the joint probability approaches in the specification of nearest-neighbour systems. *Biometrika* 51, 3/4 (1964), 481–483.

[10] Robert T Clemen and Robert L Winkler. 1999. Combining probability distributions from experts in risk analysis. *Risk Analysis* 19, 2 (1999), 187–203.

[11] William G Cochran. 1937. Problems arising in the analysis of a series of similar experiments. *Supplement to the Journal of the Royal Statistical Society* 4, 1 (1937), 102–118.

[12] William G Cochran. 1954. The combination of estimates from different experiments. *Biometrics* 10, 1 (1954), 101–129.

[13] Ahmed El Alaoui, Xiang Cheng, Aaditya Ramdas, Martin J Wainwright, and Michael I Jordan. 2016. Asymptotic behavior of\ell_p-based laplacian regularization in semi-supervised learning. In *Conference on Learning Theory*. 879–906.

[14] William R Fairweather. 1972. A method of obtaining an exact confidence interval for the common mean of several normal populations. *Applied Statistics* (1972), 229–233.

[15] Max Halperin. 1961. Almost linearly-optimum combination of unbiased estimates. *J. Amer. Statist. Assoc.* 56, 293 (1961), 36–43.

[16] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. 1983. Stochastic blockmodels: First steps. *Social networks* 5, 2 (1983), 109–137.

[17] Kyle Kloster and David F Gleich. 2014. Heat kernel based community detection. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1386–1395.

[18] Isabel M Kloumann, Johan Ugander, and Jon Kleinberg. 2017. Block models and personalized PageRank. *Proceedings of the National Academy of Sciences* 114, 1 (2017), 33–38.

[19] Frédéric Lavancier and Paul Rochet. 2016. A general procedure to combine estimators. *Computational Statistics & Data Analysis* 94 (2016), 175–192.

[20] Tianxi Li, Elizaveta Levina, and Ji Zhu. 2016. Prediction models for network-linked data. *arXiv preprint arXiv:1602.01192* (2016).

[21] Sofus A Macskassy and Foster Provost. 2007. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research* 8, May (2007), 935–983.

[22] Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology* 27, 1 (2001), 415–444.

[23] Ronny Meir. 1995. Bias, variance and the combination of least squares estimators. In *Advances in Neural Information Processing Systems*. 295–302.

[24] Boaz Nadler, Nathan Srebro, and Xueyuan Zhou. 2009. Semi-supervised learning with the graph Laplacian: The limit of infinite unlabelled data. (2009).

[25] Leto Peel. 2016. Graph-based semi-supervised learning for relational networks. *arXiv preprint arXiv:1612.05001* (2016).

[26] JNK Rao and Kathleen Subrahmaniam. 1971. Combining independent estimators and estimation in linear regression with unequal variances. *Biometrics* (1971), 971–990.

[27] Amanda L Traud, Eric D Kelsic, Peter J Mucha, and Mason A Porter. 2011. Comparing community structure to characteristics in online collegiate social networks. *SIAM review* 53, 3 (2011), 526–543.

[28] Amanda L Traud, Peter J Mucha, and Mason A Porter. 2012. Social structure of Facebook networks. *Physica A: Statistical Mechanics and its Applications* 391, 16 (2012), 4165–4180.

[29] Robert L Winkler. 1981. Combining probability distributions from dependent information sources. *Management Science* 27, 4 (1981), 479–488.

[30] Ya Xu, Justin S Dyer, and Art B Owen. 2010. Empirical stationary correlations for semi-supervised learning on graphs. *The Annals of Applied Statistics* (2010), 589–614.

[31] Denny Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *Advances in neural information processing systems*. 321–328.

[32] Xiaojin Zhu. 2006. Semi-supervised learning literature survey. *Computer Science, University of Wisconsin-Madison* 2, 3 (2006), 4.

[33] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*. 912–919.

APPENDIX

1 PROOF OF GMRF EQUIVALENCE

Theorem 3.1. *Let W be a weight matrix with row sums $z_i = \sum_j W_{ij}$ and column sums $z'_j = \sum_i W_{ij}$. Let $\tau^2 > 0$ and $\gamma \in (0, 1)$. Then the conditional specifications*

$$\theta_i|\phi \sim N\left(\frac{\gamma}{z_i}\sum_{j=1}^{n} W_{ij}\phi_j, \frac{\tau^2}{z_i}\right), \qquad \phi_j|\theta \sim N\left(\frac{\gamma}{z'_j}\sum_{i=1}^{n} W_{ij}\theta_i, \frac{\tau^2}{z'_j}\right)$$

define a valid, non-degenerate probability distribution over θ and ϕ with marginal distribution $\begin{pmatrix}\theta\\ \phi\end{pmatrix} \sim N(\mu, \Sigma)$, where $\mu = 0$ and

$$\Sigma = \tau^2 \begin{pmatrix} Z & -\gamma W \\ -\gamma W^\top & Z' \end{pmatrix}^{-1}, \tag{9}$$

where $Z = \mathrm{diag}(z_1, \ldots, z_n)$ and $Z' = \mathrm{diag}(z'_1, \ldots, z'_n)$.

Proof. We use Brook's lemma [9], which states that for any distribution such that $p(x) > 0$ for all x, then for any x and x',

$$\frac{p(x)}{p(x')} = \prod_{i=1}^{n} \frac{p(x_i|x_1, \ldots, x_{i-1}, x'_{i+1}, \ldots, x'_n)}{p(x'_i|x_1, \ldots, x_{i-1}, x'_{i+1}, \ldots, x'_n)}.$$

Applying Brook's lemma with $x = (\theta, \phi)$ and $x' = 0$, we see that

$$\begin{aligned} p(\theta, \phi) &\propto \prod_{i=1}^{n} \Bigg(\frac{p(\theta_i|\theta_1, \ldots, \theta_{i-1}, 0_{i+1}, \ldots, 0_{2n})}{p(0_i|\theta_1, \ldots, \theta_{i-1}, 0_{i+1}, \ldots, 0_{2n})} \times \\ &\qquad \frac{p(\phi_i|\theta_1, \ldots, \theta_n, \phi_1, \ldots, \phi_{i-1}, 0_{i+1}, \ldots, 0_{2n})}{p(0_i|\theta_1, \ldots, \theta_n, \phi_1, \ldots, \phi_{i-1}, 0_{i+1}, \ldots, 0_{2n})} \Bigg) \\ &= \prod_{i=1}^{n} \frac{p(\theta_i|\phi = 0)}{p(\theta_i = 0|\phi = 0)} \frac{p(\phi_i|\theta)}{p(\phi_i = 0|\theta)} \end{aligned}$$

where we have indexed the 0s for clarity. Substituting in the likelihood from the conditional models in Theorem 3.1,

$$\begin{aligned} p(\theta, \phi) &\propto \frac{\exp\left\{-\frac{z_i}{2\tau^2}\theta_i^2\right\}}{1} \frac{\exp\left\{-\frac{z'_i}{2\tau^2}\left(\phi_i - \frac{\gamma}{z'_i}\sum_j W_{ji}\theta_j\right)^2\right\}}{\exp\left\{-\frac{z'_i}{2\tau^2}\left(\frac{\gamma}{z'_i}\sum_j W_{ji}\theta_j\right)^2\right\}} \\ &= \exp\left\{-\frac{1}{2\tau^2}\sum_{i=1}^{n}\left(z_i\theta_i^2 + z'_i\phi_i^2 - 2\gamma\phi_i\sum_j W_{ji}\theta_j\right)\right\} \\ &= \exp\left\{-\frac{1}{2\tau^2}\left(\theta^\top Z\theta + \phi^\top Z'\phi - 2\gamma\theta^\top W\phi\right)\right\}, \end{aligned}$$

which is the likelihood corresponding to a mean zero Gaussian random vector with covariance given by Equation (9). □

Cross-Network Embedding for Multi-Network Alignment

Xiaokai Chu
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
University of Chinese Academy of Sciences, China
chuxiaokai@ict.ac.cn

Xinxin Fan
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
fanxinxin@ict.ac.cn

Di Yao
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
University of Chinese Academy of Sciences, China
yaodi@ict.ac.cn

Zhihua Zhu
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
University of Chinese Academy of Sciences, China
zhuzhihua@ict.ac.cn

Jianhui Huang
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
huangjianhui@ict.ac.cn

Jingping Bi
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
bjp@ict.ac.cn

ABSTRACT

Recently, data mining through analyzing the complex structure and diverse relationships on multi-network has attracted much attention in both academia and industry. One crucial prerequisite for this kind of multi-network mining is to map the nodes across different networks, i.e., so-called network alignment. In this paper, we propose a cross-network embedding method CrossMNA for multi-network alignment problem through investigating structural information only. Unlike previous methods focusing on pair-wise learning and holding the topology consistent assumption, our proposed CrossMNA considers the multi-network scenarios which involve at least two types of networks with diverse network structures. CrossMNA leverages the cross-network information to refine two types of node embedding vectors, i.e., *inter-vector* for network alignment and *intra-vector* for other downstream network analysis tasks. Finally, we verify the effectiveness and efficiency of our proposed method using several real-world datasets. The extensive experiments show that our CrossMNA can significantly outperform the existing baseline methods on multi-network alignment task, and also achieve better performance for link prediction task with less memory usage.

CCS CONCEPTS

• **Computing methodologies** → **Learning latent representations**; • **Information systems** → *Data mining*.

KEYWORDS

multi-network alignment, network embedding, node representation, network mining

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313499

ACM Reference Format:
Xiaokai Chu, Xinxin Fan, Di Yao, Zhihua Zhu, Jianhui Huang, and Jingping Bi. 2019. Cross-Network Embedding for Multi-Network Alignment. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313499

1 INTRODUCTION

The unprecedented growth of the diverse information has produced a large volume of networks, such as social networks, citation networks, and biological networks, etc. Nowadays, people usually participate in multiple networks, on the one hand, each network can depict the topological structure of all participants corresponding to some particular relationships; on the other hand, all the networks can be related by the same participants/nodes. In recent years, many approaches have been proposed to mine the potential information inherent in these related networks, such as cross-network recommendation [42], mutual community detection [39] and genetic diseases classification [33], etc. Although the related networks can share the same participants, they are mostly isolated in different networks without any known connections accordingly among them. Therefore, a crucial prerequisite for multi-network mining is to map the nodes/participants among these related networks, i.e., so-called network alignment. In the field of network alignment, the shared participants among the networks are defined as anchor nodes, as they act like anchors aligning the networks they participate in, and the relationships among anchor nodes across networks are called anchor links [12]. In many cases, a few anchor links can be known beforehand, for example, some users in Foursquare may leave their corresponding accounts of Twitter, but most of the correspondences are unknown. Therefore network alignment aims at inferring these unknown or potential anchor links among the networks.

In recent years, many literatures have been proposed to handle the network alignment problem [13, 16, 37, 40, 43, 44]. However, there still exist some issues that need further concern. First, most existing works only consider the two-network scenarios or perform

pair-wise learning in multi-network applications. However, in real-world a network is usually related to multiple networks. Thus the pair-wise learning methods can ignore much valuable complementary information across the multiple networks. Second, many previous works hold the assumption of **topology consistency**, which indicates that a node tends to have a consistent connectivity structure across different networks. Although the same node may show some similar features among the networks, the differences between the network semantic meanings can lead to quite diverse local structure of this node in each network, and this assumption can be easily violated in many applications [16, 23]. Thereby these methods can make misleading alignment in the multi-network scenarios. Third, most previous works heavily **rely on attributes**, e.g., the username, gender or other profile information. However, the attribute information is usually incomplete and unreliable [16] or unavailable [1], thus these attribute-based methods are not applicable in many realistic scenarios.

Keeping these problems in our mind, we, in this paper, study the network alignment problem in a multi-network scenario, wherein the number of networks is at least two with no attribute information available, in addition that the topology structure of each network can be diverse. Taking into account the effectiveness of preserving network structures and the efficiency of calculation in continuous vector space, we propose a novel **cross**-network embedding method for **m**ulti-**n**etwork **a**lignment, namely CrossMNA. CrossMNA can integrate cross-network information to refine much more powerful embedding vectors for alignment tasks, and can also be effectively applied to other downstream multi-network analysis tasks, such as link prediction. Compared to previous embedding-based network alignment methods [10, 15, 45] or multi-network embedding methods [19, 35, 38], CrossMNA can also dramatically decline the space overhead especially in large-scale multi-network scenarios.

In CrossMNA, an additional vector named *network vector* is proposed to extract the semantic meaning of the network, which can reflect the difference of global structure among the networks. What's more, two kinds of embedding vectors are refined for each node: (i) *inter-vector*, which reflects the common features of the anchor nodes in different networks and is shared among the known anchor nodes; and (ii) *intra-vector*, which preserves the specific structural feature for a node in its selected network and is generated through the combination of network vector and intra-vector. This combination strategy can save much more space overhead but not sacrifice the performance. We also propose a transformation matrix to align these vectors with different dimensions. We summarize our contributions as follows:

(i) We propose a novel cross-network embedding approach CrossMNA to deal with network alignment in a more general scenario, where the number of networks is at least two with no attribute information available, in addition that the structure of each network can be quite diverse.
(ii) CrossMNA declines the physical memory overhead for model storage to a large extent compared to other embedding based network alignment methods or multi-network embedding models, which enables our CrossMNA appropriate to tackle today's large-scale multi-network applications.
(iii) Extensive experiments show that our CrossMNA significantly outperforms existing network alignment methods, i.e., achieves 10% to 20% improvement with the Twitter dataset and 10% with the arXiv datasets. Furthermore, CrossMNA can achieve 5% improvement in link prediction mission with the Twitter dataset compared to existing multi-network embedding methods.

We organize the remains of our work as follows. Section 2 states the problems of multi-network alignment. We investigate several inherent vulnerabilities in previous works and propose the challenges in multi-network alignment. We formally describe our CrossMNA in Section 3, validate our approach by analyzing extensive experiments in Section 4, present related work in Section 5 and conclude our work in Section 6.

2 PROBLEM STATEMENT

As a crucial prerequisite for many cross-network applications, network alignment aims to establish the node correspondences across different networks. Therefore, in recent years, network alignment has become a hot spot in both academia and industry. However, there still exist some problems in currently existing works:

(i) **Topology Consistency.** Many previous works [2, 3, 10, 13, 14, 29, 41, 45] hold the assumption of topology consistency, namely the same node has a consistent connectivity structure across different networks. For example, NetAlign [3] utilizes max-product belief propagation based on the network topology. REGAL [10] proposes an embedding-based method based on the assumption that nodes with similar structural connectivity or degrees have a high probability to be aligned. Although one node may share some similar features in related networks, its local structural connections can be entirely different in each network due to the distinctiveness in network semantic meanings, such as the types of interaction between proteins in bioinformatics. Some works have proved that topology consistency can be easily violated in many multi-network scenarios. For example, in bioinformatics, different types of genetic interactions can construct diverse network structures [23]. Another similar example can also be found in online social network sites [16], such as Twitter and Facebook, where usersâĂŹ behavior may be divergent and platform dependent, making different social platforms show various connection relationships. Thereby, previous methods may lead to sub-optimal or even misleading alignments in these scenarios.

(ii) **Pair-Wise Learning.** Most previous works only consider the two-network scenarios or perform pair-wise learning in the multi-network scenario. However, if we jointly consider all related networks, we can obtain much more useful information to benefit the node matching. Figure 1 illustrates a toy example with three related networks. If we only consider each pair of networks, it is hard to infer the node v_4 in network G^3 is the counterpart of the node v_1 in network G^1 and G^2. However, if we consider the three networks together, this alignment can be easily found.

(iii) **Attributes Dependence.** Many previous works utilize the attribute information, like username or gender in social networks, to directly match nodes in different networks [26] or make use of

*Jingping Bi (bjp@ict.ac.cn) is the corresponding author

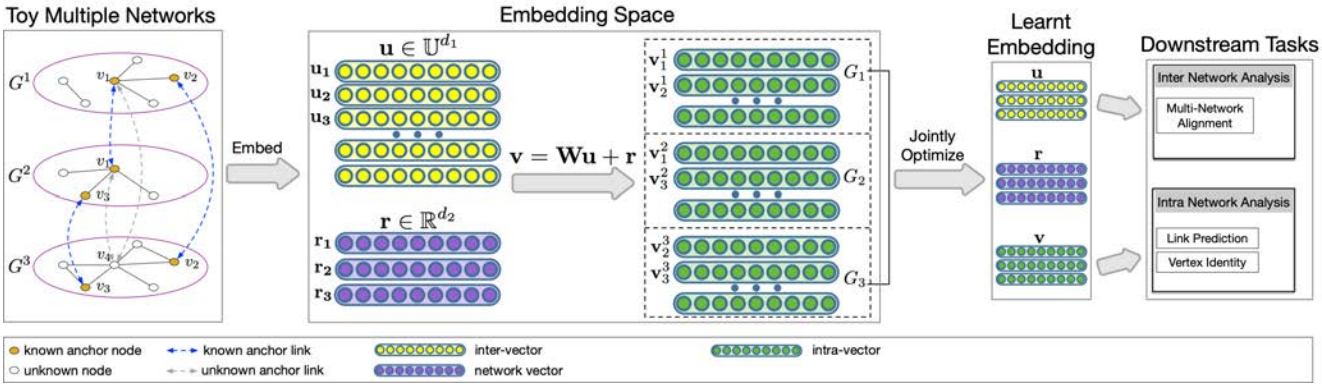


Figure 1: An illustration of a multi-network scenario. CrossMNA learns two types of embedding vectors, i.e., network vector for each network and inter-vector for each node, which is used for node matching. The intra-vector, which can be adopted for other downstream tasks, is combined by these two types of embedding vectors

the attributes to guide model learning structure information [44]. However, on the one hand, users may deliberately hide certain pieces of personal information or provide false data on their selected networks [16]. On the other hand, network data available for research is usually anonymized by the service providers for privacy concerns, where the attribute information is removed or replaced with meaningless unique identifiers [41]. Therefore, the methods based on attribute information are not general in many applications.

Considering the issues above, we intend to study the network alignment problem in a more general scenario wherein the number of networks is at least two, only topological structure available, even the networks may have different structures. In addition to the importance and novelty of this problem, we present the following challenges:

- **Semantics diversity**. The diversities in network semantics lead to the different interactional behaviors of the same node in each network, which can effect the accuracy of the node matching. Moreover, the more networks taken into consideration, the more complicated the problem is. Therefore, how to alleviate the impact of diverse network semantics on node matching is a big challenge.
- **Data imbalance**. The data imbalance has two aspects. First, the size of each network may vary considerably, including the numbers of nodes or edges in each network. Second, the number of anchor links between each pair of networks can be unequal. It is obvious that the previous pair-wise learning methods inevitably suffer from the data imbalance problem as they only consider each pair of networks. Thus how to make full use of the information across all the networks to deal with the data imbalance problem is another challenge.
- **Model Storage**. Network embedding is a practical approach to extract structural features of the node and has been applied in some network alignment methods [10, 15, 45]. However, in large-scale multi-network scenarios, it is essential to take into account the space overhead of the methods. Previous embedding-based methods need to generate the embedding vector for each node in each pair of networks in multi-network scenarios, and this takes too much storage space. Thereby, how to make overhead cheaper but not sacrifice the performance is what we should concern.

To address the challenges above, we propose a cross-network embedding based multi-network alignment method CrossMNA. It extracts an extra feature vector named *network vector* for each network, which reflects the difference of global structure among the networks. This means if the global structure of two networks is similar, their network vectors will be close in vector space. For each node in a network, we propose two types of embedding vectors: *inter-vector* and *intra-vector*. The former depicts the commonness of anchor nodes in different networks and is shared among the known anchor nodes. The latter reflects the specific structural feature of this node in its selected network but is generated through a combination of the network vector and inter-vector. By jointly training all the networks, CrossMNA can refine powerful inter-vector for network alignment tasks, and intra-vector for other downstream multi-network analysis tasks. What's more, the shared inter-vector and the combining intra-vector can significantly save space overhead without sacrificing the performance.

3 CROSSMNA: A MULTI-NETWORK ALIGNMENT APPROACH

3.1 Problem Formulation

In this work, we suppose the networks are unweighted and all the edges are directed, as an undirected edge can be divided into two directed edges. For the sake of easy understanding, we follow the definitions of network alignment as reported in [41, 42].

Definition 1 *(Multiple Aligned Networks).* We define a set of networks $\mathcal{G} = ((G^1, G^2, \cdots, G^N), (\mathcal{A}^{(1,2)}, \mathcal{A}^{(1,3)}, \cdots, \mathcal{A}^{(N-1,N)}))$ as multiple aligned networks, where $G^i, i \in \{1, 2, \cdots, N\}$ represents the i-th network in the set, N is the number of related networks, and $\mathcal{A}^{(i,j)}, i, j \in \{1, 2, \cdots, N\}$ denotes the set of anchor links between G^i and G^j. We define each network G^i as $\{V^i, E^i\}$, where V^i is the set of nodes in G^i and E^i is the set of links.

Definition 2 *(Anchor Link).* Given each two networks G^i and G^j, we define an anchor link between them as $(v_k^i, v_k^j) \in \mathcal{A}^{(i,j)}$, where

v_k^i and v_k^j is the anchor node in networks G^i and G^j respectively. The anchor links in multiple networks following the *transitivity law* defined in [41], where if $(v_k^i, v_k^j) \in \mathcal{A}^{(i,j)}$ and $(v_k^i, v_k^h) \in \mathcal{A}^{(i,h)}$, then $(v_k^h, v_k^j) \in \mathcal{A}^{(h,j)}$. For notational convenience, the nodes in different networks with the same subscript are the known anchor nodes.

Definition 3 *(Multi-Network Alignment Problem).* Given a set of networks and part of known anchor links among the networks, this problem is to discover the unknown or potential anchor links. It is noteworthy that the networks are partially aligned [42], which means not all the nodes have counterparts in other networks, and the anchor nodes follow the one-to-one matching constraint [41].

3.2 Cross-Network Embedding

As stated before, the different network semantic meanings lead to diverse interactional behaviors of the anchor nodes, which influences the global structure of the networks. For instance, in Figure 1 the anchor node v_1 shows different connection relationships in G^1 and G^2. However, as the same entity, the anchor node should also display some common features across the networks. For example, suppose that we have known that node v_4 in G^3 is the counterpart of node v_1, we can observe that the anchor node v_1 tends to interact with nodes v_2 or v_3 in each network. Therefore, we propose CrossMNA in multi-network scenarios under the assumption that an anchor node in a selected network can both show some similar structural features with its counterparts and distinctive connection relationships due to its network semantic meaning. It is noteworthy that if the related networks have the same semantic meaning, the anchor nodes will display consistent connection relationships among these networks, which is the topology consistent assumption as previous works hold.

It is obvious that the common features among the anchor nodes are what we need for network alignment. To this end, we propose an *inter-vector* $\mathbf{u}$ to preserve the common features among the anchor nodes. Through training, we hope the inter-vector of an unknown anchor node can be close to its counterparts in vector space. Nevertheless, this inter-vector is hard to be learnt directly, as there is no direct correlation between the unknown anchor nodes. For example, there is no direct relationship between the node v_1 in G^1 and the node v_4 in G^3. Therefore, we have to learn the inter-vector indirectly.

On the other side, it is straightforward to extract structural features of nodes in a network via network embedding methods, which named *intra-vector* $\mathbf{v}$ in CrossMNA. This type of vector contains both the commonness among counterparts and the specific local connections in its selected network due to the semantics, so it cannot be applied for node matching unless we can divestiture the impact of network semantic. Thus, we are motivated to present the following equation to build a correlation among intra-vector, inter-vector, and network semantics:

$$\mathbf{v}_i^k = \mathbf{u}_i + \mathbf{r}^k, \tag{1}$$

where $\mathbf{v}_i^k$ is the intra-vector of node v_i in network G^k which can be easily learnt, $\mathbf{u}_i$ is the inter-vector of node v_i and its known counterparts are shared this vector, $\mathbf{r}^k$ is the *network vector* which can extract the unique characteristics of G^k and reflects the global difference among the networks. Thus, we can refine the inter-vector of the anchor nodes indirectly by training the combining-based intra-vectors.

We take the toy multi-network in Figure 1 as example for further explanation. Considering the anchor node v_1, it should share some common features among the networks and this commonness is what the inter-vector should represent. Meanwhile, as G^1 and G^2 are two different networks, the local connections of v_1 can also be distinct because of the diverse network semantic meanings. Following Equation (1), we can give the intra-vector of node v_1 in G^1 and G^2 as:

$$\begin{aligned} \mathbf{v}_1^1 &= \mathbf{u}_1 + \mathbf{r}^1, \\ \mathbf{v}_1^2 &= \mathbf{u}_1 + \mathbf{r}^2. \end{aligned} \tag{2}$$

On the one hand, through jointly training the intra-vectors, the shared $\mathbf{u}_1$ can store the complementary information between the two networks: for example v_1 can be connected to v_2 and v_3 at the same time. On the other hand, $\mathbf{r}^1$ and $\mathbf{r}^2$ can reflect the global differences between the two networks owing to the common information transmitting in $\mathbf{u}_1$.

Then we consider the node v_4 in G^3. Its intra-vector can be writen as:

$$\mathbf{v}_4^3 = \mathbf{u}_4 + \mathbf{r}^3. \tag{3}$$

By learning the structural information in G^3, $\mathbf{v}_4^3$ can contain some common features similar to $\mathbf{v}_1^1$ or $\mathbf{v}_1^2$, for example v_4 also tends to interact with v_2 and v_3. Owing to to other known anchor links between G^3 and the other networks, the network vector $\mathbf{r}^3$ can preserve the specific semantic features of G^3. Therefore, by peeling off the impact of network semantic difference via $\mathbf{u}_4 = \mathbf{v}_4^3 - \mathbf{r}^3$, the inter-vector $\mathbf{u}_4$ can reflect much more similar features with $\mathbf{u}_1$, by which we can infer that v_4 is the counterpart of v_1.

The meanings of the inter-vector and intra-vector are different, they can be in different vector spaces. Therefore, we propose a transformation matrix $\mathbf{W}$ to align them with different dimensions and rewrite the Equation (1) as:

$$\mathbf{v}_i^k = \mathbf{W}\mathbf{u}_i + \mathbf{r}^k, \mathbf{u}_i \in \mathbb{U}^{d_1}, \mathbf{v}_i^k \in \mathbb{R}^{d_2}, \tag{4}$$

where $\mathbb{U}^{d_1}$ and $\mathbb{R}^{d_2}$ are two different vector spaces with dimension d_1 and d_2 respectively. $\mathbf{W}$, $\mathbf{u}_i$ and $\mathbf{r}^k$ are the parameters need to learned in CrossMNA.

From the above discussion, we can find that theoretically the more related networks we considered, the more accurate the node matching is. Therefore, CrossMNA needs to jointly train all the related work, and we thus propose the total objective function as:

$$\mathcal{J} = \sum_k \mathcal{J}^k, \tag{5}$$

where $\mathcal{J}^k$ is the objective function of each G^k, which tries to preserve the structural information of each node in G^k. Following [31], for each directed edge (v_i^k, v_j^k) in G^k, we define the conditional

probability of v_j^k generated by v_i^k as:

$$p(v_j^k|v_i^k) = \frac{exp(\mathbf{v}_i^k \cdot \mathbf{v}_j^k)}{\sum_{v_z^k \in V^k} exp(\mathbf{v}_i^k \cdot \mathbf{v}_z^k)} = \frac{exp((\mathbf{W}\mathbf{u}_i + \mathbf{r}^k) \cdot (\mathbf{W}\mathbf{u}_j + \mathbf{r}^k))}{\sum_{v_z^k \in V^k} exp((\mathbf{W}\mathbf{u}_i + \mathbf{r}^k) \cdot (\mathbf{W}\mathbf{u}_z + \mathbf{r}^k))}, \quad (6)$$

and the objective of each network G^k can be defined as:

$$\mathcal{J}^k = \sum_{(v_i^k, v_j^k) \in E^k} p(v_j^k|v_i^k). \quad (7)$$

By jointly optimizing, the known anchor nodes will play their roles as much as possible to transmit the structural information through the shared inter-vector $\mathbf{u}$, making this vector contain the commonness among the anchor nodes. At the same time, through Equation (6), the intra-vector of each anchor node can preserve the diverse structural features in different networks, which makes the network vector $\mathbf{r}^k$ extract the specific feature of its own global structure.

For each unknown anchor nodes v_i^k, although we cannot directly make connections to its counterparts, its combining-based intra-vector can preserve some similar features as its counterparts in other networks. By peeling off the impact of the network semantic meaning $\mathbf{r}^k$, the inter-vector $\mathbf{u}_i$ will display the characteristic features as its anchor nodes do, which can be effectively used for node matching.

To speed up the training process, following word2vec [21], we perform negative sampling to approximate the objective function (6) as:

$$E_{(v_i^k, v_j^k)} = -log\sigma(\mathbf{v}_i^k \cdot \mathbf{v}_j^k) - \sum_{v_z^k \in P(z)} log\sigma(-\mathbf{v}_i^k \cdot \mathbf{v}_z^k), \quad (8)$$

where $\sigma(x) = 1/(1 + exp(-x))$ is the sigmoid function and the distribution $P(v) \propto d_v^{3/4}$, where d_v is the degree of node v in the given network. Therefore, we rewrite the objective function of (5) as the total loss function of the whole multi-network:

$$\mathcal{L} = \sum_{k \in [1,N]} \sum_{(v_i^k, v_j^k) \in E^k} -log\sigma(\mathbf{v}_i^k \cdot \mathbf{v}_j^k) - \sum_z log\sigma(-\mathbf{v}_i^k \cdot \mathbf{v}_z^k). \quad (9)$$

We adopt Adam[11] to minimize the total loss function.

As we know, in real-world, it is ubiquitous to meet the data imbalance problem among multiple networks. Previous methods can easily suffer from the data imbalance problem as they under the pair-wise learning scheme. However, our CrossMNA can make up this problem, because CrossMNA jointly trains all the networks and transmits the complementary information among the known anchor nodes through the shared inter-vectors, which can alleviate the influence of the data imbalance problem. What's more, compared to existing embedding-based methods, CrossMNA is much more lightweight. Instead of generating embedding vector for each node in each pair of networks, CrossMNA shares the common intra-vector across the networks and uses a more flexible combining-based vector generation strategy as defined in Equation (4) to generate the intra-vector for each node in each network. No matter compared to the embedding-based network alignment methods or the existing multi-network embedding methods, CrossMNA can always significantly reduce the memory overhead, and we will compare the size of CrossMNA with other models in Section 4.7.

3.3 Multi-Network Mining

After training, we can get two types of embedding vectors for each node: the intra-vector reflects the structural information of a node in a selected network, which is suitable for the intra-network analysis tasks, while the inter-vector interprets the commonness of the anchor nodes in different networks, which can be effectively applied to multi-network alignment task.

A naive way to find the alignments for a node is to compute all pairs of similarities between the inter-vectors, which is time-consuming. However, in practice, we usually only need to find the soft alignments for each node by returning its top-α most likely nodes in other networks. So following [10], we use the k-d tree data structure to accelerate similarity search [4] in the matching process. We here use cosine similarity to define the distance of two nodes in vector space.

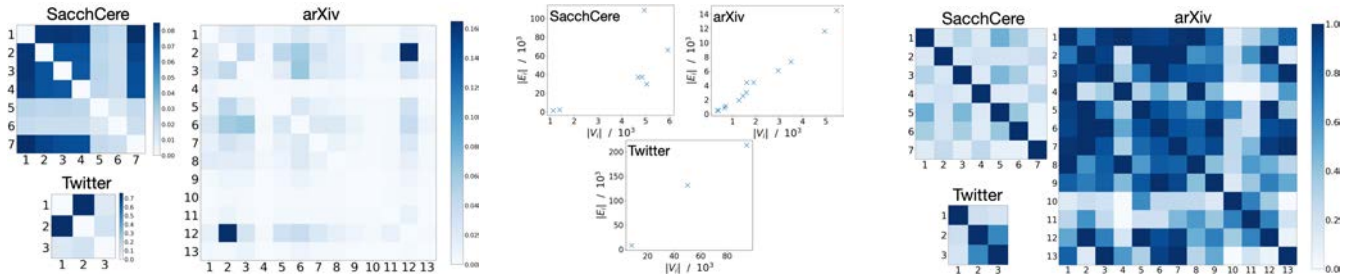
In some applications, we can get the complete groundtruth among multiple networks beforehand, which means all the aligned links are known. As our CrossMNA is a cross-network embedding approach, it can be easily exploited in these applications to make the data mining tasks on multiple networks more actionable. During training, the inter-vector takes the role of transmitting the structural information among the anchor nodes. Therefore, CrossMNA can preserve the complementary information across multiple networks. At the same time, because we extract the specific features of each network into its network vector, the intra-vector which is combined by the inter-vector and the specific network vector, can both keep the complementary information of its counterparts in other networks and the distinctive properties of its selected network. This can be effective in many downstream multi-network analysis tasks, such as link prediction and vertex identity.

4 EXPERIMENTAL EVALUATION

In this section, we perform a set of experiments to validate our proposed CrossMNA. We first compare our method with several state-of-the-art methods on the multi-network alignment task. Then, to verify the effectiveness of CrossMNA in extracting useful features of the nodes, we also compare our CrossMNA with several multi-network embedding methods on link prediction tasks.

4.1 Dataset Description

We employ three real-world multi-network datasets from different fields, e.g., social platform, bioinformatics, and academics. To reflect the data imbalance problem in each dataset, we present the distribution of anchor links between each pair of networks in Figure 2(a) and size of each network in Figure 2(b). We can find that the scale of each network vary greatly in all datasets and distribution of anchor links can be uneven, especially in Twitter and SacchCere. We also present the global structural similarities between each pair of networks, by comparing the common neighbors reachable in three steps of the anchor nodes. From Figure 2(c) we can find that the related networks have quite diversity structures, especially in Twitter and SacchCere. We detail the datasets as follows:



(a) Distribution of anchor links. (b) Size of networks. (c) Global similarity between networks.

Figure 2: Detailed information of the datasets. (a) Each grid represents $|A^{(i,j)}|/|A|$. (b) Each dot is a network, the abscissa/ordinate represents the number of nodes/edges. (c) Each grid reflects the global similarity between two networks.

arXiv[1] [6] consists of various networks in terms of different arXiv categories. There are 14,489 nodes, 59,026 coauthorship connections, and 13 networks. The number of anchor links among the networks is 23,626.

Twitter[1] [24] This dataset is a specific Twitter dataset which focused on People's Climate March in 2014. It makes use of 3 networks, corresponding to retweet, mentions, and replies. 102,439 nodes, 353,495 edges and 55,600 anchor links are included.

SacchCere[1] [7, 30] is a subset of BioGRID, which is a public database that archives and disseminates genetic and protein interaction data from humans and model organisms. It contains 6,570 nodes, 282,755 connections and 7 networks, each represents one type of genetic interaction. There are a total of 55,831 anchor links.

4.2 Baseline Methods

Multi-Network Alignment. To show the effectiveness of our method in addressing the multi-network alignment problem, we compare with four different baseline methods through only using network structure information.

NetAlign [3] proposes the message passing algorithms to match networks under unsupervised schemas.

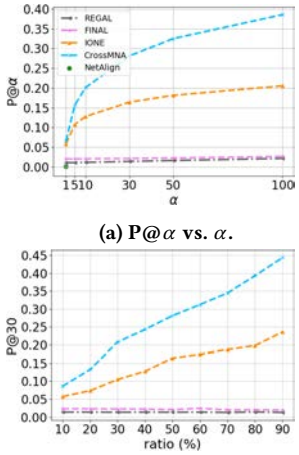
FINAL [43] proposes a family of algorithms to align attributed networks. It formulates network alignment from an optimization perspective referring to the alignment consistency principle.

REGAL [10] is an unsupervised multi-network alignment method, which extracts similarity-based representations among graphs and infers the soft alignments by comparing the learnt embedding vectors.

IONE [15] is a state-of-the-art network embedding based method under semi-supervised schemes, which solves both the network embedding problem and the user alignment problem simultaneously with a unified optimization framework.

As NetAlign and FINAL both require the prior alignment information, following previous work we construct a degree similarity matrix and take the top-$log|V|$ entries for each node.

Intra-Link Prediction. To verify the ability of CrossMNA on extracting effective features of the nodes among the networks. We also compare our proposed CrossMNA with five network embedding methods on link prediction task, which are divided into two



(a) P@α vs. α.

(b) P@30 vs. Training ratio.

Figure 3: Performance with Twitter.

categories in terms of single-network embedding or multi-network embedding:

DeepWalk [27] is a typical network embedding method that learns vertex representations based on single network structures. It performs a random walk on the network to obtain vertex sequences and conducts Skip-Gram model to train the sequences.

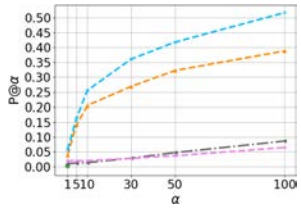
LINE [31] is another network embedding model which aims at learning node embeddings in a large-scale network. It minimizes a loss function to preserve both first-order and second-order proximity between nodes.

node2vec [8] defines a flexible notion of a nodeâĂŹs neighborhood and propose a biased random walk, which is a trade-off between Breadth First Search and Depth First Search.

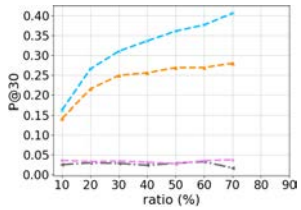
PMNE [17] is a multi-network embedding method which proposes two simply merged approaches and a Co-analysis method to obtain one overall embedding for each node. In our experiments, we compare with its final Co-analysis method. The Co-analysis PMNE performs a biased random-walk across each network and conducts Skip-Gram model to train the node sequences.

MELL [19] is a multi-network embedding method which simultaneously learns the embedding vector of each node and a layer

[1] https://comunelab.fbk.eu/data.php

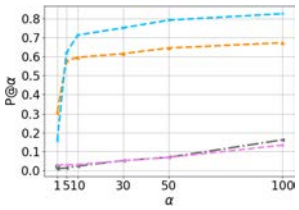


(a) P@α vs. α.

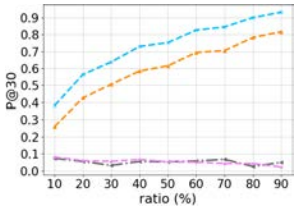


(b) P@30 vs. Training ratio.

Figure 4: Performance with SacchCere.



(a) P@α vs. α.



(b) P@30 vs. Training ratio.

Figure 5: Performance with arXiv.

embedding of each single network, using all of the network structures.

4.3 Experiment Configuration

The parameters of the compared methods are set as follows: For FINAL, we follow the default setting, i.e. $\{\alpha, t_{max}\} = \{0.3, 30\}$. For REGAL we set the maximum hop distance $K = 2$; For Deepwalk, we set walks per vertex as 20, window size as 5 and walk length as 80; For LINE, we employ both first-order and second-order proximity and obtain representations via concatenation, and we set the negative samples $k = 5$; For node2vec, we empirically set $p = 2$ and $q = 0.5$. For PMNE and MELL, we follow their default settings, i.e., $\{\alpha, p, q\} = \{0.5, 0.5, 0.5\}$, $\{k, \lambda, \beta, \gamma\} = \{4, 1, 1, 1\}$. For fair comparisons, we set the same node dimension $d = 200$ for all embedding-based methods in network alignment, and $d = 100$ for each network embedding model in link prediction. For our method CrossMNA, we thus set $d_1 = 200$, $d_2 = 100$ and the number of negative samples as 1 to speed up the training process. We use the inter-vector $\mathbf{u}$ for network alignment task and the intra-vector $\mathbf{v}$ for link prediction task.

For multi-network alignment task, we randomly remove part of the anchor links as the test set and the rest as the training set. We compare the precision of the soft alignment for each method. We first introduce the $Precision^{(i,j)}@\alpha$ as the evaluation metric of mapping nodes from G^i to G^j:

$$Precision^{(i,j)}@\alpha = \frac{|CorrectNodes^{(i,j)}@\alpha|}{|UnMappedAnchors^{(i,j)}|}, \tag{10}$$

where $|UnMappedAnchors^{(i,j)}|$ is the number of unknown anchor links between G^i and G^j and $|CorrectNodes^{(i,j)}@\alpha|$ is the number of the correct alignments from G^i to G^j in top-α choices. Therefore, the evaluation metric on multi-network alignment can be defined as:

$$Precision@\alpha = \frac{1}{N(N-1)} \sum_i \sum_{j \neq i} Precision^{(i,j)}@\alpha, \tag{11}$$

where N is the number of networks.

For intra-link prediction task, we randomly split all edges into two sets for training and testing respectively. We also randomly sample an unconnected node pair as a negative edge for each positive edge in the test set and use both the positive and negative edges for testing. We here adopt a standard evaluation metric ROC-AUC [9] in our experiments.

4.4 Performance Evaluation

Multi-Network Alignment. The experimental results are presented in Figure 3-5. We first keep the training ratios to 0.5 and give the results of different @α settings, which is the subfigure (a) for each dataset. Then we vary the training ratio from 0.1 to 0.9 with an interval of 0.1 and evaluate the *Precision*@30 of each method, which is the subfigure (b) for each dataset. We remove the results of training ratio 0.8 and 0.9 on SacchCere, because most networks have few anchor nodes in the test set, leading to the unstable performance of each method. We only give *P*@1 for NetAlign as it aims to find the hard matching for the node. From the results, we have the following observations:

(i) CrossMNA can outperform all the baselines in each dataset on almost all training ratios. We can gain 10% to 20% improvement on Twitter dataset than the state-of-the-art method IONE, and 5% to 12% improvement on SacchCere. In arXiv dataset, CrossMNA can also achieve about 10% improvement on different training ratios. Furtherly, even when the training ratio is very small, such as 10% and 20%, CrossMNA can still achieve the best performance and get 5% to 10% improvement. This result demonstrates that our method is still effective when the known anchor links are rare, which is a very common problem in many real-world applications.

(ii) The unsupervised methods NetAlign, REGAL, and FINAL show poor performance on all datasets, because the structures of each network do not follow the topology consistent assumption they hold, which leads to the misleading alignments in all datasets. The state-of-the-art embedding-based method IONE, however, can achieve good performance on each dataset. However, it only considers the pair-wise learning, thus it ignores many useful complementary information across different networks. What's more, it does not explicitly consider the problems of distinctive network structures, resulting in the mismatching of some anchor nodes with different local structures. Therefore, it is hard to achieve better performance when the networks have quite different structures.

(iii) Combined with Figure 2, we can find that most networks in arXiv dataset tend to have similar global structure, and the anchor nodes are more evenly distributed across networks. Therefore, IONE can achieve good performance on this dataset, e.g., IONE

Table 1: Results of alignment between each pair of networks in Twitter dataset. The training ratio is 50% and 1→2 denotes matching nodes from the 1st network to the 2nd network.

$P^{(i,j)}$@30	1→2	2→1	1→3	3→1	2→3	3→2
IONE	0.056	0.054	0.035	0.032	0.4601	0.4589
CrossMNA	**0.1327**	**0.1106**	**0.2817**	**0.1676**	**0.5863**	0.4210

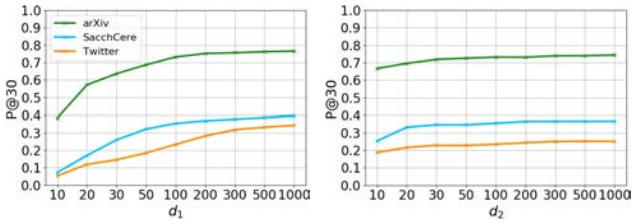


Figure 6: Parameter study on multi-network alignment.

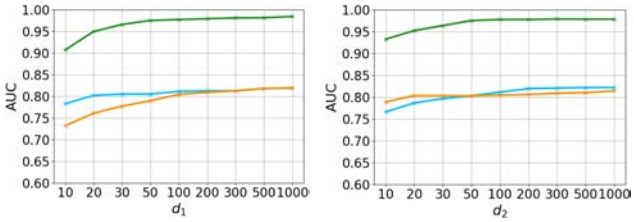


Figure 7: Parameter study on link prediction.

can even gain better performance on $P@1$ when the training ratio is 50% than CrossMNA. However, CrossMNA can take advantage of the jointly learning all the networks, so our CrossMNA can have a stable improvement of about 10% on different ratios. For SacchCere dataset, as the semantic meanings are different among the networks, the structure of each network is diverse, which prevents the baselines from achieving good results. However, as we jointly learn from all the networks, the abundant complementary information among the networks can improve the quality of learnt embeddings, alleviating the problem above. For Twitter dataset, the state-of-the-art method IONE cannot achieve satisfactory result due to two reasons: (i) the sizes of networks vary a lot and the distribution of anchor nodes is uneven in this dataset, e.g., the 1st network has nearly 100,000 nodes but 3nd network has only about 8,000 nodes and most anchor links are between the 1st and 2nd networks; (ii) the networks show diverse structures, e.g., the 1st network shows quite different topology structure compared to the other two networks. For better understanding, we present the performance of alignment between each pair of networks on Twitter in Table 1. We can observe that IONE can gain a comparable result with our CrossMNA in alignment between the 2nd and 3rd networks, which have the similar topology structure. However, for alignment between the 1st-2nd networks and the 1st-3rd networks, IONE cannot achieve satisfactory results because the size of the networks vary a lot and they show quite diversity global structures, which we can find in Figure 2(c). However, as our CrossMNA extracts an extra network vector to peel off the influences of network semantic on node's local connections, the inter-vector can preserve the common features among the anchor nodes as much as possible, which shows significant improvement on this dataset.

To summarize, the above observations illustrate our method is significantly effective and appropriate to address multi-network alignment problem.

Intra-Link Prediction. Besides applied on network alignment, the node embedding learnt by our CrossMNA can also be adopted in other downstream network analysis tasks, e.g., link prediction and vertex identity. We take the link prediction task as an example to evaluate the quality of the learnt features. We compare with five state-of-the-art methods, three for the single-layer network and two for multiple networks. For the models designed for the single-layer network, we train a separate embedding for each network and use that to predict the links of the corresponding networks. We evaluate the AUC values of different models with the training ratio 30%, 50%, and 80% and propose the results in Table 2.

Upon the experimental results, we find that for all the datasets, our CrossMNA can achieve comparable performance or even significantly outperform all the baselines. What calls for special attention is that CrossMNA can achieve a nearly 5% improvement on Twitter dataset on all the training ratios. This is because the inter-vector in CrossMNA plays a role to transmit the complementary information across the networks while the network vectors can store the distinctive properties of each network, which make the combining-based intra-vector integrate the cross-network information without sacrificing the distinctive properties in its selected network. To summarize, the above results indicate the effectiveness of features refined by our method in multi-network mining tasks.

4.5 Parameter Sensitivity

There are two parameters in our method: the dimension of inter-vector d_1 and intra-vector d_2. To explore the influence of the parameters on the model performance, we set $d_1 = d_2 = 100$ as default, pick up one parameter each time and vary that parameter while fixing another to check this parameter's impact on the tasks of network alignment and link prediction. For network alignment task, we keep the training ratio to 50% and use the $P@30$ as the metric. For link prediction task, we also keep the training ratio to 50%.

From Figures 6 and 7, we can observe that d_1 has a great influence on both tasks, and as d_1 grows the performance is significantly improved. While d_1 is larger than a threshold, the performance will be stable, and the threshold is related to the size of the network. When d_1 is larger than 200 on arXiv and SacchCere, the performance will shift towards a steady distribution. However, $d_1 = 300$ has a significant improvement than $d_1 = 200$ on Twitter dataset in alignment task, as the scale of this dataset is very large. We also find that d_2 has a relatively smaller effect on the performance if d_1 chooses a suitable number. Upon these observations, we are motivated to set a relative larger d_1, such as 200, 300 and a very small d_2, such as 30, 50 to save the memory in practice.

Table 2: AUC score of link prediction task.

Dataset	arXiv			SacchCere			Twitter		
	30%	50%	80%	30%	50%	80%	30%	50%	80%
DeepWalk	87.86	94.41	98.12	69.20	73.96	78.29	57.09	59.96	63.74
LINE	75.05	85.73	94.75	60.54	65.05	68.87	53.12	52.97	53.22
node2vec	88.06	94.57	97.11	71.15	76.33	80.16	56.84	61.22	65.93
PMNE	90.12	94.47	95.24	77.61	79.85	81.35	61.12	70.72	75.91
MELL	93.51	96.30	98.84	76.18	79.92	81.21	70.64	75.89	79.84
CrossMNA	**96.46**	**97.53**	**99.19**	76.88	**81.12**	**82.59**	**75.85**	**80.48**	**85.29**

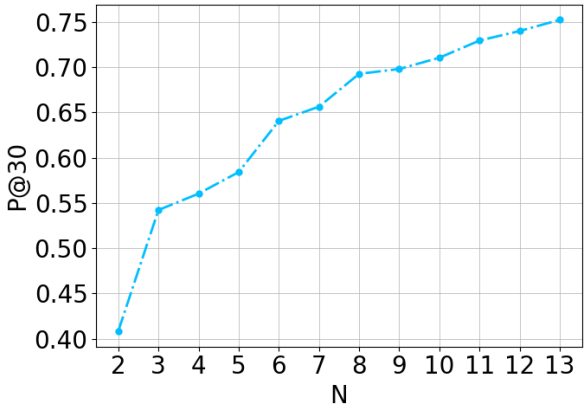


Figure 8: Precision *w.r.t.* N.

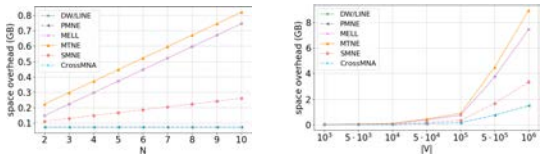


Figure 9: Memory use comparisons among multi-network embedding methods.

4.6 Case Study

In this section, we perform a set of experiments to explore whether CrossMNA can gain better performance with more related networks taken into account. To make the experimental results more convincing, we choose the arXiv dataset which has the most number of networks. This means the relationships among the networks are very complicated.

We vary the number of networks from 2 to 13 and set the training ratio to 50%. We perform this experiment 10 times, and at each time we shuffle the order of the networks which are taken into consideration. We calculate the average score and present the result of $P@30$ in Figure 8. We can observe that as the number of networks increases, the overall trend of precision also goes up. These results indicate that the complementary information among the networks benefits the result of alignment, and the more related networks considered, the better performance our CrossMNA can achieve.

Table 3: Space complexity analysis among embedding-based network alignment methods.

model	number of parameters	p=0	p=0.1	p=0.2
IONE	$d \cdot (N-1)N\|V\|$	3051.75 (MB)		
REGAL	$d \cdot N\|V\|$	762.93		
CrossMNA	$d_1 \cdot \|V\|\,(N - Np + p) + d_2 \cdot N + d_1 \cdot d_2$	763.01	701.21	640.94

Notes:

In practice, we usually set $d_2 \ll d_1 \approx d$.
p denotes the ratio of known anchor links.

4.7 Scalability

In this section, we explore the scalability of our CrossMNA. Concretely, we analyze the space complexity and time complexity of our method respectively.

4.7.1 Space Complexity Analysis. As we know, in real life the networks could be very huge. In large-scale multi-network analysis tasks, if we learn embeddings for all nodes in each network, the storage issue could be a big challenge. Therefore, we ought to analyze the space complexity of the embedding-based models. We first analyze the number of parameters in each model. Then we perform a set of experiments to show the actual space overhead with regard to the size of the network. For convenience, we assume that each network shares the same set of nodes. We here use a subgraph of Youtube [32] and adopt randomized permutation to generate multiple networks. We analyze the embedding-based network alignment methods and the multi-network embedding methods respectively.

From Table 3, we observe that IONE inevitably takes too much memory especially when many related networks exist. Given IONE belongs to pair-wise learning, it needs to generate embeddings for each pair of networks. Unlike IONE, REGAL saves much space as it can directly train multiple networks together owing to the randomly sampling landmarks strategy to approximate the similarities among nodes. For our CrossMNA, because of jointly training multiple networks and the combination strategy, it only generates the inter-vector for the node and the network vector for each network. It is noteworthy that the inter-vector is shared among the known anchor nodes, so the more the known anchor nodes, the less space our CrossMNA takes. For easy understanding, we give an example with five networks and 100,000 nodes and present the space overhead with different ratios of the known anchor links. We set $d = d_1 =$

Table 4: Space complexity analysis among multi-network embedding methods.

model	number of parameters
DeepWalk/LINE	$d \cdot N\|V\|$
PMNE	$d \cdot \|V\|$
MELL	$d \cdot N(\|V\| + 1)$
MTNE [35]	$d \cdot (N + 1)\|V\|$
SMNE [38]	$d \cdot \|V\| + s \cdot N\|V\| + d \cdot s \cdot N$
CrossMNA	$d_1 \cdot \|V\| + d_2 \cdot N + d_1 \cdot d_2$
Notes:	
s is the dimension of typed relation vector in SMNE and $s \ll d$.	

$200, d_2 = 50$. We can find that even in the extreme case where there is no known anchor link, our method can still take up nearly the same amount of space compared to REGAL. When there are a few known anchor links, our CrossMNA can take advantage of its combination strategy and show less space overhead.

Then we compare CrossMNA with existing multi-network embedding methods. From Table 4, we can find that PMNE has the least number of parameters among all the methods. Compared to other methods, the size of SMNE is relatively smaller as its hyper-parameter s can be set as a small number. However, our CrossMNA can take the smallest memory overhead in most cases, because we do not need to generate the embedding vector for each node in each network, which requires $dN|V|$ parameters. We use the strategy of combining the common inter-vector and the network vector via a transformation matrix to generate the vector for each node in each network. Therefore CrossMNA only generates $d_1|V| + d_2N + d_1d_2$ parameters, where $d_1 \approx d$ and d_2 can be a very small number. To show the difference more clearly, we extend the size of a multi-network in the number of networks and nodes respectively, and present the memory overhead of each model in Figure 9. We can observe that CrossMNA and PMNE are significantly lighter than other models, and as the size of the network increases, the space overhead of CrossMNA and PMNE grows very slowly. However, as PMNE merges the information from all networks into one type of embedding, it inevitably loses the distinctive information in each network as shown in our previous experiments. Thus our CrossMNA is dramatically space efficient for large-scale multi-network applications compared to existing methods.

4.7.2 Time Complexity Analysis. The runtime of CrossMNA is in two parts: learning cross-network embedding and matching nodes through vectors comparison. The total time complexity of learning embeddings is approximately $O(tN\,(d_1d_2|V| + d_2|E|))$, where t is the number of iterations, N denotes the number of networks, and $|V|, |E|$ denote the number of nodes and edges in each network respectively. The time complexity of finding soft alignment between each two networks is $O(|V|log|V|)$.

5 RELATED WORK

5.1 Network Alignment

Network alignment is a fundamental problem for cross-network mining and many pieces of literature have been proposed to handle this problem. Most previous works make full use of the attribute information of nodes, e.g., username, gender, etc. The username is the most commonly used feature in almost all these works. Perito et al. [26] employ binary classifiers to determine if the cross-platform user with the same name is the same person. Vosecky et al. [34] extracts distance-based profile features and build classifiers to match users in multiple networks, and similar works can also be found in [18, 25]. In some scenarios, we can get some special contents information of the nodes. Therefore, there also exist some methods to take account of these useful contents for more precision node matching. Riederer et al. [28] links users by considering their trajectory-based content features. Almishari et al. [22] aligns users by exploiting their writing style.

The connection relationship between the nodes is another common and vital feature. Thus many works have been proposed to consider both the content information and the network structure. COSNET [44] uses an energy-based model to link users by considering both local and global consistency. HYDRA [16] presents a multi-objective framework to model heterogeneous behaviors and structure consistency simultaneously.

However, as the attribute information of the nodes is usually missing, unreliable or even unavailable in real-life applications, some methods have been proposed to align nodes only based on the structural information. BigAlign [13] proposes to align two bipartite graphs with a fast alignment algorithm. UMA [41] jointly optimizes multiple anonymized social networks in unsupervised schemes, under the constraints of transitivity law and one-to-one property. IONE [15] learns the follower-ship/followee-ship of each user under the framework of network embeddings and utilizes the embedding vectors to match unknown anchor users.

People nowadays usually participate in multiple diversities of networks simultaneously, and jointly learning from these related networks can get much more useful complementary information for alignment. UMA [41] and REGAL [10] have been proposed to optimize multiple networks together only considering structural information. However, the assumption of topology consistency makes them fail to deal with networks with distinct structures. Considering the problems above, we propose CrossMNA, a light-weight cross-network embedding based network alignment method, to jointly learn structural information across diversities of networks.

5.2 Network Embedding

Network embedding has been emerged as an effective and efficient approach for learning low-dimension distributed representations for the nodes in networks and is researched intensively. Previous works can be roughly divided into two categories: single-network embedding and multi-network embedding.

Most of the previous works focus on the single-layer network. Inspired by the distributed representation learning of words in NLP [20], DeepWalk [27] performs random walk over networks to generate vertex sequences and conducts Skip-Gram to obtain node

embeddings. On top of DeepWalk, Node2vec [8] modifies the random walk strategy into a biased random walk to explore network structure more efficiently. Unlike the random walk based methods above, LINE [31] optimizes two objective functions to separately approximate first-order and second-order proximity in the large-scale networks. Another general approach for obtaining node embeddings is matrix factorization. GreRap [5] proposed a matrix factorization based methods to encode k-step representations, where each step reflects different local information. TADW [36] incorporates text contents into network embedding under the framework of matrix factorization. Although these methods have achieved satisfactory performance on many single-layer network mining tasks, they ignore the multiple networks scenario.

To make the data mining task on the multiple networks more actionable, recently several works have been proposed to transform the multiple networks into low-dimension vector space. PMNE [17] proposes two simple merge-based methods which only consider inter-layer edges or intra-layer edges, and one cross-layer method which performs a biased random-walk across each layer, to obtain one overall embedding for each node. MTNE [35] and SMNE [38] builds a bridge among different layers by sharing a common embedding across each layer of the multiple networks. MELL [19] proposes the method of simultaneously learning node embeddings and layer embeddings using all of the network structures.

However, most of these multi-network embedding models ignore the problem of space overhead, which makes them hard to be applied in large-scale multi-network applications. Unlike these methods, our CrossMNA occupies much less physical without sacrificing the performance, owing to the shared inter-vectors and the flexible combination strategy.

6 CONCLUSION

We have studied the problem of multi-network alignment and propose CrossMNA, a light-weight cross-network embedding based network alignment method for tackling today's large-scale multi-network applications. CrossMNA novelly defines two categories of embedding vectors for each node, i.e. inter-vector and intra-vector. The commonness of anchor nodes in different networks is represented by the inter-vector, while the specific structural feature is interpreted by the intra-vector. In addition, the coordination of inter-vector and intra-vector dramatically decline the space overhead while reserving the adequate performance. Extensive experiments showed analytically our CrossMNA can significantly outperform other currently existing baselines and multi-network embedding methods.

ACKNOWLEDGMENTS

This work has been supported by the National Natural Science Foundation of China under Grant No.: 61702470, 61472403, and Beijing Natural Science Foundation under Grant No.: 4182062.

REFERENCES

[1] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. 2007. Wherefore art thou r3579x?:anonymized social networks, hidden patterns, and structural steganography. In *International Conference on World Wide Web*. 181–190.

[2] Mohsen Bayati, Margot Gerritsen, David F. Gleich, Amin Saberi, and Ying Wang. 2009. Algorithms for Large, Sparse Network Alignment Problems. In *Proceedings of the 9th ICDM IEEE International Conference on Data Mining, ICDM '09, 2009*. 705–710.

[3] Mohsen Bayati, David F. Gleich, Amin Saberi, and Ying Wang. 2013. Message-Passing Algorithms for Sparse Network Alignment. *ACM Transactions on Knowledge Discovery from Data* 7, 1 (2013), 1–31.

[4] Nitin Bhatia and Vandana. 2010. Survey of Nearest Neighbor Techniques. *CoRR* abs/1007.0085 (2010).

[5] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning Graph Representations with Global Structural Information. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM '15, 2015*. 891–900.

[6] Manlio De Domenico, Andrea Lancichinetti, Alex Arenas, and Martin Rosvall. 2015. Identifying Modular Flows on Multilayer Networks Reveals Highly Overlapping Organization in Interconnected Systems. *Physical Review X* 5, 1 (2015), 011027.

[7] Manlio De Domenico, Vincenzo Nicosia, Alexandre Arenas, and Vito Latora. 2015. Structural Reducibility of Multilayer Networks. *Nature Communications* 6 (2015), 6864.

[8] Aditya Grover and Jure Leskovec. 2016. Node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, 2016*. 855–864.

[9] J. A. Hanley and B. J. Mcneil. 1982. The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. *Radiology* 143, 1 (1982), 29.

[10] Mark Heimann, Haoming Shen, Tara Safavi, and Danai Koutra. 2018. REGAL: Representation Learning-based Graph Alignment. In *Proceedings of the 27th CIKM ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*. 117–126.

[11] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[12] Xiangnan Kong, Jiawei Zhang, and Philip S. Yu. 2013. Inferring Anchor Links across Multiple Heterogeneous Social Networks. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, CIKM '13, 2013*. 179–188.

[13] Danai Koutra, Hanghang Tong, and David Lubensky. 2013. BIG-ALIGN: Fast Bipartite Graph Alignment. In *Proceedings of the 13th ICDM IEEE International Conference on Data Mining, ICDM '13, 2013*. 389–398.

[14] Chung-Shou Liao, Kanghao Lu, Michael Baym, Rohit Singh, and Bonnie Berger. 2009. IsoRankN: spectral methods for global alignment of multiple protein networks. *Bioinformatics* 25, 12 (2009), 253–8.

[15] Li Liu, William K. Cheung, Xin Li, and Lejian Liao. 2016. Aligning Users across Social Networks Using Network Embedding. In *Proceedings of the 25th IJCAI International Joint Conference on Artificial Intelligence, IJCAI '16, 2016*. 1774–1780.

[16] Siyuan Liu, Shuhui Wang, Feida Zhu, Jinbo Zhang, and Ramayya Krishnan. 2014. HYDRA: Large-scale Social Identity Linkage via Heterogeneous Behavior Modeling. In *Proceedings of the 41th International Conference on Management of Data, SIGMOD '14, 2014*. 51–62.

[17] Weiyi Liu, Pin-Yu Chen, Sailung Yeung, Toyotaro Suzumura, and Lingli Chen. 2017. Principled Multilayer Network Embedding. In *Proceedings of the 17th ICDM IEEE International Conference on Data Mining Workshops, ICDM Workshops '17, 2017*. 134–141.

[18] Anshu Malhotra, Luam C. Totti, Wagner Meira Jr., Ponnurangam Kumaraguru, and Virgílio A. F. Almeida. 2012. Studying User Footprints in Different Online Social Networks. In *Proceedings of the 3rd ASONAM International Conference on Advances in Social Networks Analysis and Mining, ASONAM '12, 2012*. 1065–1070.

[19] Ryuta Matsuno and Tsuyoshi Murata. 2018. MELL: Effective Embedding Method for Multiplex Networks. In *Proceedings of the 27th World Wide Web Conference on World Wide Web, WWW '18, 2018*. 1261–1268.

[20] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013).

[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of Advances in Neural Information Processing Systems, NIPS '13, 2013*. 3111–3119.

[22] Mishari Al Mishari and Gene Tsudik. 2012. Exploring Linkability of User Reviews. In *Proceedings of the 17th ESORICS European Symposium on Research in Computer Security, ESORICS '12, 2012*. 307–324.

[23] Jingchao Ni, Hanghang Tong, Wei Fan, and Xiang Zhang. 2014. Inside the Atoms: Ranking on a Network of Networks. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, 2014*. 1356–1365.

[24] Elisa Omodei, Manlio De De Domenico, and Alex Arenas. 2015. Characterizing Interactions in Online Social Networks during Exceptional Events. *Frontiers in Physics* 3 (2015), 59.

[25] Olga Peled, Michael Fire, Lior Rokach, and Yuval Elovici. 2013. Entity Matching in Online Social Networks. In *Proceedings of the 5th SocialCom International Conference on Social Computing, SocialCom '13, 2013*. 339–344.

[26] Daniele Perito, Claude Castelluccia, Mohamed Ali Kâafar, and Pere Manils. 2011. How Unique and Traceable Are Usernames?. In *Proceedings of the 11th PETS International Symposium on Privacy Enhancing Technologies, PETS '11, 2011*. 1–17.

[27] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, 2014*. 701–710.

[28] Christopher J. Riederer, Yunsung Kim, Augustin Chaintreau, Nitish Korula, and Silvio Lattanzi. 2016. Linking Users Across Domains with Location Data: Theory and Validation. In *Proceedings of the 25th WWW International Conference on World Wide Web, WWW '16, 2016*. 707–719.

[29] Rohit Singh, Jinbo Xu, and Bonnie Berger. 2008. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences of the United States of America* 105, 35 (2008), 12763–12768.

[30] Chris Stark, Bobby-Joe Breitkreutz, Teresa Reguly, Lorrie Boucher, Ashton Breitkreutz, and Mike Tyers. 2006. BioGRID: a General Repository for Interaction Datasets. *Nucleic Acids Research* 34, suppl_1 (2006), 535–539.

[31] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web, WWW '15, 2015*. 1067–1077.

[32] Lei Tang and Huan Liu. 2009. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '15, 2009*. 817–826.

[33] Marc A van Driel, Jorn Bruggeman, Gert Vriend, Han G Brunner, and Jack A M Leunissen. 2006. A text-mining analysis of the human phenome. *European Journal of Human Genetics* 14, 5 (2006), 535–542.

[34] Jan Vosecky, Dan Hong, and Vincent Y. Shen. 2010. User identification across multiple social networks. In *Proceedings of the 1st NDT International Conference on Networked Digital Technologies, NDT '10, 2010*. 360–365.

[35] Linchuan Xu, Xiaokai Wei, Jiannong Cao, and Philip S. Yu. 2017. Multi-task Network Embedding. In *Proceedings of the 4th IEEE International Conference on Data Science and Advanced Analytics, DSAA '17, 2017*. 571–580.

[36] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. 2015. Network Representation Learning with Rich Text Information. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence, IJCAI '15, 2015*. 2111–2117.

[37] Reza Zafarani and Huan Liu. 2013. Connecting Users Across Social Media Sites:a Behavioral-modeling Approach. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13, 2013*. 41–49.

[38] Hongming Zhang, Liwei Qiu, Lingling Yi, and Yangqiu Song. 2018. Scalable Multiplex Network Embedding. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence, IJCAI '18, 2018*. 3082–3088.

[39] Jiawei Zhang and Philip S. Yu. 2015. Community Detection for Emerging Networks. In *Proceedings of the 15th SIAM International Conference on Data Mining, SIAM '15, 2015*. 127–135.

[40] Jiawei Zhang and Philip S. Yu. 2015. Integrated Anchor and Social Link Predictions across Social Networks. In *Proceedings of the 24th IJCAI International Joint Conference on Artificial Intelligence, IJCAI '15, 2015*. 2125–2132.

[41] Jiawei Zhang and Philip S. Yu. 2015. Multiple Anonymized Social Networks Alignment. In *Proceedings of 15th ICDM IEEE International Conference on Data Mining, ICDM '15, 2015*. 599–608.

[42] Jiawei Zhang, Philip S. Yu, and Zhi-Hua Zhou. 2014. Meta-path based Multi-network Collective Link Prediction. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, 2014*. 1286–1295.

[43] Si Zhang and Hanghang Tong. 2016. FINAL: Fast Attributed Network Alignment. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, 2016*. 1345–1354.

[44] Yutao Zhang, Jie Tang, Zhilin Yang, Jian Pei, and Philip S. Yu. 2015. COSNET: Connecting Heterogeneous Social Networks with Local and Global Consistency. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '15, 2015*. 1485–1494.

[45] Xiaoping Zhou, Xun Liang, Xiaoyong Du, and Jichao Zhao. 2018. Structure Based User Identification across Social Networks. *IEEE Trans. Knowl. Data Eng.* 30, 6 (2018), 1178–1191.

Improving Treatment Effect Estimators Through Experiment Splitting

Dominic Coey
Facebook
Menlo Park, California
coey@fb.com

Tom Cunningham
Facebook
Menlo Park, California
tomcunningham@fb.com

ABSTRACT

We present a method for implementing shrinkage of treatment effect estimators, and hence improving their precision, via experiment splitting. Experiment splitting reduces shrinkage to a standard prediction problem. The method makes minimal distributional assumptions, and allows for the degree of shrinkage in one metric to depend on other metrics. Using a dataset of 226 Facebook News Feed A/B tests, we show that a lasso estimator based on repeated experiment splitting has a 44% lower mean squared predictive error than the conventional, unshrunk treatment effect estimator, a 18% lower mean squared predictive error than the James-Stein shrinkage estimator, and would lead to substantially improved launch decisions over both.

CCS CONCEPTS

• **General and reference** → **Experimentation**; • **Mathematics of computing** → *Nonparametric statistics*; *Multivariate statistics*; • **Computing methodologies** → *Supervised learning by regression.*

KEYWORDS

Causal Inference; A/B Tests; Empirical Bayes Shrinkage; Sample Splitting; Experiment Meta-Analysis.

ACM Reference Format:
Dominic Coey and Tom Cunningham. 2019. Improving Treatment Effect Estimators Through Experiment Splitting. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313452

1 INTRODUCTION

As the cost of experimentation decreases, researchers are increasingly able to collect data on multiple, related experiments, which test similar kinds of changes [6, 8, 15, 27, 30]. Technology companies may test multiple versions of the machine learning models determining what content is shown to users, where models may differ in their features, architecture, hyperparameter values, or the optimization algorithm used. This raises the prospect of being able to improve inferences for any given experiment, by using the information contained in the other, related experiments. In Bayesian terms, estimating a prior for treatment effect sizes based on previous experiments, and updating it given the data from the current experiment, may yield a better estimate of the true treatment effect than would be possible given the current experiment's data alone. This kind of *empirical Bayes* analysis has a rich history in statistics [11, 12, 19, 22, 31]. With the proliferation of A/B testing, it is increasingly relevant to the modern analysis of experimental data.

We propose a methodology for performing nonparametric, multivariate empirical Bayes shrinkage, which turns the shrinkage problem into a standard prediction problem. The basic idea is as follows: we split each experiment randomly into two subexperiments, each with their own test and control group. We regress the estimated treatment effect in the first subexperiment on the estimated treatment effect in the second subexperiment. Because the dependent variable is unbiased, this regression also estimates the conditional mean of the true treatment effect, given the data in the second subexperiment. In other words, it is an empirical Bayes estimator of the parameter of interest. Other variables likely to be predictive of the treatment effect can be added as covariates into this regression, including estimated treatment effects on auxiliary metrics in the second subexperiment. The prediction algorithm used may be a simple linear regression, but could also be more flexible and nonparametric, allowing for nonlinearities, interactions, and regularization of covariates.

In this methodology, the prior distribution of treatment effects is not explicitly estimated.[1] Instead, we directly estimate the conditional mean of the treatment effect given the observed data, using the experiment splits. Other experiments implicitly determine the prior, and hence the degree of shrinkage implied by this conditional mean. This raises the question of how to choose the fraction of data allocated to the first and the second subexperiments. We derive an expression for the mean squared error of the experiment splitting estimator with univariate least squares shrinkage, and characterize the limiting behavior of the optimal experiment splitting fraction as a function of the number of experiments and the variance of treatment effects and sampling error.

Using a dataset of 226 Facebook News Feed A/B tests, we evaluate the performance of experiment splitting estimators relative to the leading alternatives, including other empirical Bayes shrinkage estimators. We find that experiment splitting reduces mean squared prediction error by 44% compared to the conventional, unshrunk estimator, by 18% compared to James-Stein shrinkage, and by 13% compared to the "Tweedie" estimator. Launching all experiments with positive estimates results in a 40% higher estimated cumulative lift for the lasso than the unshrunk estimator. While experiment splitting generates some performance improvements from flexible,

[1] Other empirical Bayes methods which avoid estimating priors are surveyed in [18], including Robbins' formula [31], the local false-discovery rate [17], and Tweedie's formula [16].

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313452

nonparametric shrinkage and appropriate regularization of treatment effect predictors, we find the main gains are attributable to the multivariate shrinkage that experiment splitting straightforwardly enables. Estimated treatment effects in one metric may reveal information about the true treatment effect in another metric, and experiment splitting makes it easy to incorporate this covariance information.

We focus on shrinkage of treatment effect estimators across experiments, but the idea of using split sample regressions to generate better estimators applies more broadly. If the heterogeneity in treatment effects across user subgroups within an experiment is of interest, experiment splitting can be applied at the level of user subgroups within a single experiment. In non-experimental settings where multiple means from separate samples are being estimated, sample splitting can be used to determine the appropriate degree of shrinkage for each sample mean.

Relative to existing work on empirical Bayesian shrinkage, which we discuss in Section 2, our contribution is to formulate the shrinkage problem as a prediction problem to which standard machine learning algorithms can be applied. This leads to easy-to-implement algorithms for very general forms of shrinkage, which do not require parametric assumptions on the treatment effect or sampling error distributions, and which allow for incorporating information from a potentially high-dimensional vector of auxiliary metrics.

While using split sample regressions to shrink treatment effect estimates appears to be new to the literature, other applications of sample splitting for causal inference abound. Sample splitting has been proposed as a means for constructing valid confidence intervals for heterogeneous treatment effect estimates [3, 37], making the statistical analysis of random forests more tractable [4, 7], and reducing false positives from specification searches [2, 23]. A fast-growing body of work analyzes the properties of semiparametric "cross-fitting" estimators, in which high-dimensional nuisance parameters are estimated by machine learning algorithms on a subsample of the data, and their out-of-sample predictions used to estimate the causal parameters of interest [5, 13, 14, 29, 36, 38].

2 EMPIRICAL BAYESIAN SHRINKAGE

For each of N_E experiments, we have a test group and a control group, each made up of N_P people.[2] In experiment i, we observe test and control group participant outcomes, $(Y^t_{i,k})_{k=1}^{N_P}$ and $(Y^c_{i,k})_{k=1}^{N_P}$. Within each experiment, test and control outcomes are iid across people. The treatment effect in experiment i is defined as $\theta_i = E(Y^t_{i,k}) - E(Y^c_{i,k})$, and the θ_i are themselves iid across experiments. We denote by $\widehat{\theta}_i$ the difference of sample means treatment effect estimator: $\widehat{\theta}_i = \frac{1}{N_P}\sum_{i=k}^{N_P}(Y^t_{i,k} - Y^c_{i,k}) = \theta_i + \varepsilon_i$, where ε_i is sampling error, assumed independent of θ_i. We wish to compute the conditional mean of the treatment effects given the observed data. Of particular interest is the Bayes estimator $E(\theta_i|\widehat{\theta}_i)$, which minimizes mean squared error [31]. We briefly review existing approaches to this problem under various distributional assumptions on $(\theta_i, \varepsilon_i)$.

[2]The assumption that the test and control groups are of equal size is made only to simplify the exposition, and is inessential in what follows.

2.1 Normal Treatments, Normal Errors

Known variances. Consider the case where $\theta_i \sim N(0, \sigma^2_\theta)$, $Y^c_{i,k} \mid \theta_i \sim N(0, \sigma^2_Y)$ and $Y^t_{i,k} \mid \theta_i \sim N(\theta_i, \sigma^2_Y)$, for known variances σ^2_θ and σ^2_Y. The difference of sample means estimator has distribution $\widehat{\theta}_i \mid \theta_i \sim N(\theta_i, \sigma^2_\varepsilon)$, where $\sigma^2_\varepsilon = 2\sigma^2_Y/N_P$. By Bayes' rule the posterior distribution of the treatment effect given the data is $\theta_i \mid \widehat{\theta}_i \sim N(\alpha\widehat{\theta}_i, \alpha\sigma^2_\varepsilon)$, where $\alpha = \sigma^2_\theta/(\sigma^2_\theta + \sigma^2_\varepsilon)$. The Bayes estimator of θ_i is $E(\theta_i \mid \widehat{\theta}_i) = \alpha\widehat{\theta}_i$. This shrinks the difference of sample means estimator, $\widehat{\theta}_i$, towards the prior mean of the treatment effects, zero, by the coefficient α.[3] Thus with normal treatment effects and sampling error, optimal shrinkage is linear in the estimate $\widehat{\theta}_i$. The Bayes risk with squared loss for the difference of sample means $\widehat{\theta}_i$ is $\sum_{i=1}^{N_E} E(\widehat{\theta}_i - \theta_i)^2 = \sigma^2_\varepsilon N_E$, whereas for the Bayesian shrinkage estimator $\alpha\widehat{\theta}_i$, it is $\sum_{i=1}^{N_E} E(\alpha\widehat{\theta}_i - \theta_i)^2 = \alpha\sigma^2_\varepsilon N_E$ [17]. The gains from optimal shrinkage are large if the signal-to-noise ratio $\sigma^2_\theta/\sigma^2_\varepsilon$, and hence α, is small. When the sampling error variance σ^2_ε is very large, almost all of the variation in $\widehat{\theta}_i$ reflects sampling error, rather than variation in the underlying treatment effect θ_i, and the Bayes estimator offsets the effect of sampling error by aggressively shrinking $\widehat{\theta}_i$ towards the prior mean.

Unknown treatment effect variance. In practice σ^2_θ, and hence the desired shrinkage factor α, may be unknown. One may specify a prior distribution over σ^2_θ and conduct full Bayesian inference, computing the posterior of θ_i given the data and priors [34, 35]. An alternative is empirical Bayes, in which the shrinkage factor is estimated from the data. The celebrated James-Stein estimator is

$$\widehat{\theta}^{JS}_i = \left(1 - \frac{(N_E - 2)\sigma^2_\varepsilon}{\sum_{i=1}^{N_E}\widehat{\theta}^2_i}\right)\widehat{\theta}_i.$$

Although originally studied in a frequentist setting with fixed $\theta_i's$ [26, 32], it can be viewed as an empirical Bayes estimator, where $1 - (N_E - 2)\sigma^2_\varepsilon/\sum_{i=1}^{N_E}\widehat{\theta}^2_{i,2}$ is an unbiased estimator of the Bayes optimal shrinkage factor [17, 20, 33].

Unknown sampling error variance or prior mean. In experimental settings the sampling error variance σ^2_ε is typically unknown, but can be readily estimated. Letting $\widehat{\sigma}^2_Y$ denote the sample variance of all test and control outcomes across all people and experiments, we use the estimator $\widehat{\sigma}^2_\varepsilon = 2\widehat{\sigma}^2_Y/N_P$. When the prior mean is non-zero, the same shrinkage logic as above applies, with estimates being linearly shrunk towards the prior mean instead of towards zero. When this prior mean is unknown, it can be estimated as the sample mean across experiments of the observed $\widehat{\theta}_i$.

2.2 General Treatments, Normal Errors

The treatment effects θ_i need not be normally distributed, and are significantly non-normal in the data we study in Section 5. This implies that $E(\theta_i \mid \widehat{\theta}_i)$ need not be linear in $\widehat{\theta}_i$, unlike in the normal case. Let f denote the marginal density of $\widehat{\theta}_i$. Then *Tweedie's formula* ([10, 16]) for the posterior mean of θ_i is:

[3]The fact that the Bayes optimal estimator of θ_i involves shrinking $\widehat{\theta}_i$ towards zero is consistent with $\widehat{\theta}_i$ being unbiased for θ. The former observation is that $E(\theta_i \mid \widehat{\theta}_i) \leq \widehat{\theta}_i$; the latter that $E(\widehat{\theta}_i \mid \theta_i) = \theta_i$.

$$E(\theta_i \mid \widehat{\theta}_i) = \theta_i + \sigma_\varepsilon^2 \frac{d \log f(\widehat{\theta}_i)}{d\theta_i}. \quad (1)$$

Because the $\widehat{\theta}_i$ are observed, unlike the θ_i, we can directly estimate the density f from the data, and use this estimate in equation (1) to construct an estimator of $E(\theta_i \mid \widehat{\theta}_i)$. Even if the treatment effects are non-normal, normality may still be a reasonable assumption for the sampling error. If N_P is large and the assumptions of the central limit theorem hold, ε_i will be close to normal.

3 EXPERIMENT SPLITTING

Experiment splitting takes a different approach to estimating treatment effects. The main observation is that multiple independent estimates of the same treatment effect can be obtained from a single experiment, by simply partitioning the experiment into subexperiments. In brief, by randomly splitting each experiment into two subexperiments, we can estimate the conditional mean of the treatment effect estimate in the second subexperiment given the treatment effect estimate in the first. This conditional mean is what is required for optimally shrinking our estimates from the second subexperiment.

In more detail, we split both the test group and the control group into two subgroups, with a fraction γ of people selected uniformly at random for the first subexperiment (up to integer constraints), and the remainder assigned to the second subexperiment. Without loss of generality, let $(Y^t_{i,k})_{k=1}^{\lfloor \gamma N_P \rfloor}$ and $(Y^c_{i,k})_{k=1}^{\lfloor \gamma N_P \rfloor}$ be the test and control outcomes of the people in the first subexperiment, and similarly $(Y^t_{i,k})_{k=\lfloor \gamma N_P \rfloor+1}^{N_P}$ and $(Y^c_{i,k})_{k=\lfloor \gamma N_P \rfloor+1}^{N_P}$ for people in the second subexperiment. In each subexperiment, we construct the difference of means treatment effect estimator: $\widehat{\theta}_{i,1} = \frac{1}{\lfloor \gamma N_P \rfloor} \sum_{k=1}^{\lfloor \gamma N_P \rfloor} (Y^t_{i,k} - Y^c_{i,k})$ and $\widehat{\theta}_{i,2} = \frac{1}{N_P - \lfloor \gamma N_P \rfloor} \sum_{k=\lfloor \gamma N_P \rfloor+1}^{N_P} (Y^t_{i,k} - Y^c_{i,k})$. It follows that $E(\widehat{\theta}_{i,2} \mid \widehat{\theta}_{i,1}) = E(\theta_i \mid \widehat{\theta}_{i,1})$, as $\widehat{\theta}_{i,2}$ equals θ_i plus a mean zero error term independent of $\widehat{\theta}_{i,1}$. Moreover, $E(\widehat{\theta}_{i,2} \mid \widehat{\theta}_{i,1})$, and hence $E(\theta_i \mid \widehat{\theta}_{i,1})$, can be estimated by standard regression or prediction techniques. This conditional mean is the Bayes estimator of θ_i given the data in the first subexperiment. Thus by having a set of "hold-out" subexperiments, we can estimate the optimal level of shrinkage. This requires no assumptions on the marginal distributions of θ_i or ε_i, beyond the mild regularity conditions for estimating a conditional mean.

Experiment splitting yields an estimate of $E(\theta_i \mid \widehat{\theta}_{i,1})$, not $E(\theta_i \mid \widehat{\theta}_i)$. The conditioning is on the difference in means in only the first subexperiment, not the entire data. In principle not conditioning on all data might impair estimator performance, to the extent that more conventional estimators dominate experiment splitting. The results of Section 5 indicate to the contrary that experiment splitting estimators can substantially outperform unshrunk, James-Stein and Tweedie estimators, in real-world A/B tests.

This framework also naturally extends to accommodate covariates. Letting X denote a vector of variables available based on data from the first subexperiment, we can estimate $E(\widehat{\theta}_{i,2} \mid X_i)$. In contrast to conventional experiment meta-analysis, X_i includes an estimate of the outcome of interest formed from the data in experiment i, $\widehat{\theta}_{i,1}$. In addition, the covariate vector X_i may include treatment effects on other metrics in the same experiment, or experimental metadata, all of which may be useful for predicting θ_i. Thus the experiment splitting estimator's performance may be improved, if auxiliary metrics or experiment metadata are available that are predictive of the outcome of interest. The conditional expectation can be estimated by standard prediction algorithms like the lasso, gradient boosted decision trees, or neural nets. Thus experiment splitting allows us to treat nonparametric multivariate shrinkage as a simple prediction problem. Unlike classical approaches to the problem of multivariate shrinkage (e.g. [21]), it is unnecessary to estimate the full covariance matrix of all metrics' treatment effects.[4] The covariate vector may even be high-dimensional, as this can be accommodated by common supervised learning algorithms.

Generating confidence intervals around the estimate of $E(\theta_i \mid X_i)$ poses no special difficulty for the experiment splitting estimator. The only requirement is that valid confidence intervals can be calculated for whatever statistical procedure is applied for estimating the conditional mean.

3.1 Selecting the Experiment Splitting Fraction

Because of the relation between experiment splitting and cross-validation described in Section 3.4, we also call the first subexperiment (used to construct the covariates in the experiment splitting regression) the "training" subexperiment, and the second subexperiment (used to construct the outcomes) the "validation" subexperiment. The relative amount of data in training and validation subexperiments affects the properties of the experiment splitting estimator. If most of the data is assigned to the training subexperiment, then the sampling error in $\widehat{\theta}_{i,1}$ is relatively small; if most of the data is assigned to the validation subexperiment, then the sampling error in $\widehat{\theta}_{i,2}$ is relatively small. The optimal splitting fraction strikes a balance so that $\widehat{E}(\theta_i \mid \widehat{\theta}_{i,1})$, the estimator of the conditional mean $E(\theta_i \mid \widehat{\theta}_{i,1})$, is as close as possible on average to θ_i.

We present two strategies for estimating the out-of-sample error rate for selecting the optimal experiment splitting fraction. The first is cross-validation. This requires splitting the original experiment into *three*: two subexperiments corresponding to the training and validation data, and a third, "test" subexperiment, which is used as a out-of-sample, independent measure of the truth. The difference between a model's predictions—which are a function of the training and validation data—and the estimates of the true effect from the test data, can be used to estimate the model's performance. We perform the random assignment of data into training, validation and tests sets $S > 1$ times and average the results, to reduce the variance in the estimated performance metric due to the choice of splits. This entire procedure can be performed for various choices of γ (which determines the relative sizes of the training and validation subexperiments), and the γ yielding the lowest prediction error can be selected. Algorithm 1 summarizes this procedure, given $\gamma \in \{\gamma_1, \gamma_2 \ldots, \gamma_G\}$, and with half of the overall data allocated to the test subexperiment.

[4] In addition [21] assume that the metrics' sampling error is independent across metrics and homoskedastic. These assumptions—especially independence across metrics—are hard to justify in typical A/B tests.

Algorithm 1 Cross-Validation for Optimal Splitting

(1) For splitting fraction indices $g = 1$ to G:
  (a) For splitting simulations $s = 1$ to S:
    (i) For experiments $i = 1$ to N_E, randomly split experiment i into the training, validation and test subexperiments, each with a proportion $0.5\gamma_g, 0.5(1-\gamma_g)$ and 0.5 of the data, respectively.
    (ii) Apply the experiment splitting estimator to the training and validation subexperiments, and obtain predictions of the true effect in each experiment.
    (iii) Calculate the mean squared difference between these predictions and the estimated effects in the test subexperiment. Denote this err_{gs}.
  (b) Calculate the mean squared prediction error over splitting simulations s, $\text{MSPE}_g := \frac{1}{S}\sum_{s=1}^{S}\text{err}_{gs}$.
(2) Return γ_{g^*}, where $g^* \in \arg\min_g \text{MSPE}_g$.

The second strategy for selecting the experiment splitting fraction involves minimizing an unbiased risk estimate. This is analogous to covariance penalty approaches to estimating prediction error, including Mallows' C_p [28], the Akaike Information Criterion (AIC) [1], and Stein's unbiased risk estimate [34]. Unlike cross-validation, the unbiased risk estimate we develop applies only to multivariate OLS shrinkage, but does not require holding out a test subexperiment, is computationally simpler, and yields insight into how the optimal splitting fraction depends on sample sizes and variance parameters. Given a vector of experiment covariates X_i of length l, the treatment effect estimator for experiment i is $X_i'\widehat{\beta}$, where $\widehat{\beta} = \arg\min_\beta \sum_i(\widehat{\theta}_{i,2} - X_i'\widehat{\beta})^2$. The unshrunk subexperiment treatment effect estimates are written as $\widehat{\theta}_{i,j} = \theta_i + \varepsilon_{i,j}$, for $j = 1, 2$. We assume that the error $\varepsilon_{i,j}$ is independent across i and j, and that $\varepsilon_{i,2}$ is independent of X_i. Let $\sigma^2_{\varepsilon_j}$ denote the variance of $\varepsilon_{i,j}$. We also assume that the $N_E \times l$ regressor matrix $X = (X_1, X_2, \ldots, X_{N_E})'$ has rank l with probability 1. The following proposition forms the basis of our unbiased risk estimator. All proofs are in the appendix.

Proposition 1. *The mean squared error of the multivariate linear shrinkage estimators* $(X_i'\widehat{\beta})_{i=1}^{N_E}$ *is*

$$\sum_{i=1}^{N_E} E(\theta_i - X_i'\widehat{\beta})^2 = \left(\sum_{i=1}^{N_E} E(\widehat{\theta}_{i,2} - X_i'\widehat{\beta})^2\right) - (N_E - 2l)\sigma^2_{\varepsilon_2}.$$

Proposition 1 shows that to estimate risk, we simply adjust the sum of squared residuals by a term which depends on N_E, the number of experiments, and l, the number of regressors. As l increases, the in-sample model fit, measured by the sum of squared residuals, increasingly understates the true error. Let $\widehat{\sigma}^2_{\varepsilon_2}$ denote an unbiased estimator of $\sigma^2_{\varepsilon_2}$, which can be easily calculated from the individual-level outcomes in the second subexperiment. Then $\left(\sum_{i=1}^{N_E}(\widehat{\theta}_{i,2} - X_i'\widehat{\beta})^2\right) - (N_E - 2l)\widehat{\sigma}^2_{\varepsilon_2}$ is unbiased for the risk of the experiment splitting estimator. This risk estimate can be calculated for different experiment splitting fractions, in order to select the estimated risk-minimizing fraction.

3.2 Determinants of the Optimal Split

In the case of univariate OLS, we regress the estimated treatment effect in the validation subexperiment on a constant and the estimated treatment effect in the training subexperiment. Our shrinkage estimator for experiment i is $\widehat{\beta}_0 + \widehat{\beta}_1\widehat{\theta}_{i,1}$, where $(\widehat{\beta}_0, \widehat{\beta}_1) = \arg\min_{\beta_0,\beta_1}\sum_i(\widehat{\theta}_{i,2} - \beta_0 - \beta_1\widehat{\theta}_{i,1})^2$. With relatively more data in the training subexperiment, the variance of $\widehat{\theta}_{i,1}$ decreases. But little data in the validation subexperiment means that the $\widehat{\theta}_{i,2}$ are imprecisely estimated, increasing the variance of $(\widehat{\beta}_0, \widehat{\beta}_1)$. We show how the optimal splitting fraction that balances these two effects depends on the number of experiments, and the variance of treatment effects and sampling error. The following result gives an explicit expression for risk of univariate OLS shrinkage in terms of the sample size and variance parameters.

Corollary 1. *The mean squared error of the univariate linear shrinkage estimators* $(\widehat{\beta}_0 + \widehat{\beta}_1\widehat{\theta}_{i,1})_{i=1}^{N_E}$ *is*

$$\sum_{i=1}^{N_E} E(\theta_i - \widehat{\beta}_0 - \widehat{\beta}_1\widehat{\theta}_{i,1})^2 = \frac{(N_E - 2)\sigma^2_\theta\sigma^2_{\varepsilon_1}}{\sigma^2_\theta + \sigma^2_{\varepsilon_1}} + \sigma^2_{\varepsilon_2}.$$

The next result derives the limiting behavior of the optimal split. For simplicity we treat the sample sizes and the experiment splitting fraction as real-valued variables, so the necessary derivatives are defined. We require $0 < \underline{\gamma} \le \gamma \le \overline{\gamma} < 1$ for some $\underline{\gamma}$ and $\overline{\gamma}$ close to zero and one, so that no subexperiment has zero observations.

Proposition 2. *Assume* $Var(Y_i^t|\theta_i) = Var(Y_i^c|\theta_i) = \sigma^2_Y$ *for all experiments* i, *and* $N_E \ge 3$. *Define* $\sigma^2_\varepsilon = \frac{2\sigma^2_Y}{N_P}$, *and let* $\gamma^* \in [\underline{\gamma}, \overline{\gamma}]$ *minimize mean squared error from univariate linear shrinkage. Then* $\lim_{N_E\to\infty}\gamma^* = \overline{\gamma}$, $\lim_{\sigma^2_\theta\to 0}\gamma^* = \lim_{\sigma^2_\varepsilon\to\infty}\gamma^* = \underline{\gamma}$, *and* $\lim_{\sigma^2_\theta\to\infty}\gamma^* = \lim_{\sigma^2_\varepsilon\to 0}\gamma^* = \frac{\sqrt{N_E-2}}{1+\sqrt{N_E-2}}$.

Of particular note is the limiting behavior of γ^* as N_E grows. The optimal split puts increasingly more data in the training subexperiment, as any noise in the dependent variable of the splitting regression can be compensated for with a sufficiently large number of experiments N_E. Figure 1 shows the mean squared error from univariate linear shrinkage, for varying values of σ^2_θ and N_E, for σ^2_ε fixed at 1, and normal treatment effects and sampling errors. Consistent with Proposition 2, the optimal γ is increasing in σ^2_θ and N_E. The figure illustrates the asymmetry in risk caused by very small vs. very large values of γ. As γ approaches 0, the signal in $\widehat{\theta}_{i,1}$ vanishes, $\widehat{\beta}_1$ approaches 0, and experiment splitting approaches the bounded risk of the complete shrinkage case. As γ approaches 1, risk is unbounded: extreme imprecision in the dependent variable can cause extreme errors in the shrinkage estimator.

When $\sigma^2_\theta = 0.1$ and $N_E = 100$, there is no interior solution for γ^*, and risk is strictly increasing in γ. This implies that experiment splitting is futile if the signal-to-noise ratio is too small. For large enough sampling error variance, complete shrinkage—the limiting case as γ approaches 0—is always superior to experiment splitting. The suboptimality involved in performing experiment splitting nonetheless is small, however, even for moderate values of γ.

Figure 2 shows γ^*, the optimal fraction of data to allocate to the training subexperiment, as a function of the variance of treatment

Figure 1: Experiment Splitting Fractions and RMSE

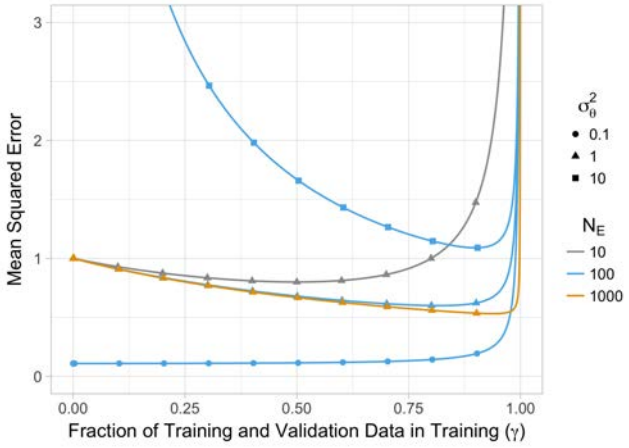
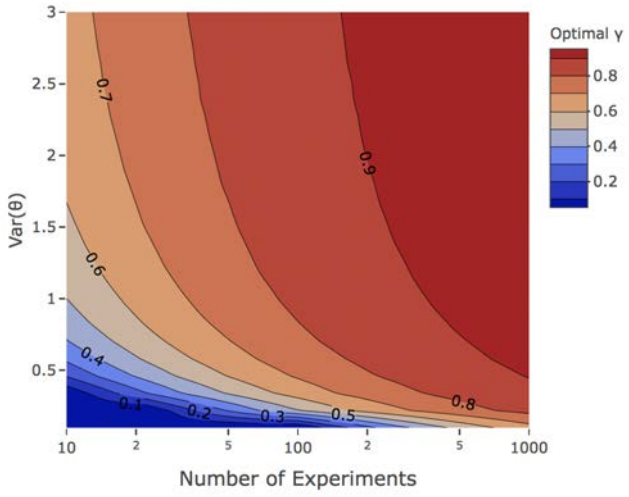


Figure 2: Optimal Experiment Splitting Fraction



effects, σ^2_θ, and the number of experiments, N_E. The sampling error σ^2_ε is again fixed at one. With N_E between 100 and 200 and a signal-to-noise ratio between around one and three, the training subexperiment should contain about 80% to 90% of the data.

3.3 Repeated Experiment Splitting

Our focus until now has been on generating predictions from datasets in which each experiment is split *once*.[5] An alternative is repeated splitting, where we randomly split each experiment into training and validation subexperiments $R > 1$ times. This is appealing, as averaging over repeated splits reduces the "excess" variance due to the random choice of user assignments to subexperiments. This averaging can be understood as a form of Rao-Blackwellization. The experiment splitting estimator is a function of the training and validation assignment, and the individual-level data. Conditioning on the individual-level data—a sufficient statistic—by averaging possible training and validation splits will improve the mean-squared error of the experiment splitting estimator. In the empirical results of Section 5, we verify that this is indeed the case.

[5]This is true even in the cross-validation procedure of Algorithm 1: the additional split there is used to evaluate the quality of the model predictions, not to generate the predictions themselves.

Algorithm 2 Repeated Experiment Splitting

(1) For experiment splitting repetitions $r = 1$ to R:
  (a) For $i = 1$ to N_E, randomly split experiment i into two subexperiments, with a fraction γ of data in the training subexperiment and $1 - \gamma$ in the validation subexperiment. Let $\widehat{\theta}^{(r)}_{i,1}$ and $\widehat{\theta}^{(r)}_{i,2}$ denote the unshrunk estimated treatment effects in those subexperiments.
  (b) Given the set of N_E pairs $\left(\widehat{\theta}^{(r)}_{i,1}, \widehat{\theta}^{(r)}_{i,2}\right)_{i=1}^{N_E}$, estimate the conditional mean $E\left(\widehat{\theta}^{(r)}_{i,2} \mid \widehat{\theta}^{(r)}_{i,1}\right)$. Denote this estimator by $\widehat{m}^{(r)}(\cdot)$.
  (c) For $i = 1$ to N_E, compute the shrinkage estimates $s^{(r)}_{i,1} := \widehat{m}^{(r)}\left(\widehat{\theta}^{(r)}_{i,1}\right)$.
(2) For $i = 1$ to N_E, return the repeated experiment splitting shrinkage estimator for experiment i, given by $\frac{1}{R}\sum_{r=1}^{R} s^{(r)}_{i,1}$.

Algorithm 2 describes the repeated experiment splitting algorithm. The case of $R = 1$ reduces to the original, single repetition experiment splitting estimator. The validity of this procedure for $R > 1$ follows immediately from the validity of the $R = 1$ case. The conditional mean $E\left(\widehat{\theta}^{(r)}_{i,2} \mid \widehat{\theta}^{(r)}_{i,1}\right)$ does not depend on the repetition r, since the data is identically distributed across repetitions. Denote this conditional mean function by $m(\cdot)$. For each repetition r, as long as our estimator $\widehat{m}^{(r)}(\cdot)$ is consistent for $m(\cdot)$, the average of these estimators over r will also be consistent. If the estimates $\widehat{m}^{(r)}(\cdot)$ were independent across r, setting the number of repetitions R to be sufficiently high would ensure arbitrarily precise estimates of m. In practice the $\widehat{m}^{(r)}(\cdot)$ are correlated as they are all functions of the same dataset, and this will not be the case.

For cross-validation of the splitting fraction, we nest this algorithm in the cross-validation meta-algorithm, Algorithm 1. The inner loop of Algorithm 1 calls for applying an experiment splitting estimator given a splitting fraction γ_g. In this setting, implementing that step itself requires splitting the data R times.

3.4 Connection with Cross-Validation

Experiment splitting may appear similar to the practice of cross-validating predictive models. The researcher splits experiments and estimates the optimal degree of shrinkage, based on predicting outcomes in the "held-out" subexperiments. Similarly, in cross-validation, the researcher splits data and estimates the optimal degree of regularization based on predicted outcomes in the held-out data. This is more than an analogy. Experiment splitting with univariate OLS shrinkage and a prior treatment effects mean of zero

is equivalent to cross-validation of a particular ridge regression. This connection holds for any fixed splitting fraction γ, and is unrelated to the use of cross-validation for selecting the splitting fraction as described in Section 3.1.

In experiment splitting with a prior treatment effect mean of zero, the univariate linear regression problem is $\min_\beta \sum_i (\widehat{\theta}_{i,2} - \beta\widehat{\theta}_{i,1})^2$. Define $\Delta_{i,k} = Y^t_{i,k} - Y^c_{i,k}$. That is, we arbitrarily pair up test and control participants into N_P pairs, and denote each pair's difference in outcomes by $\Delta_{i,k}$. Note that $E(\Delta_{i,k}) = \theta_i$ for all i. Regress $\Delta_{i,k}$ on N_E indicator variables, one corresponding to each experiment, on the sample of people in the first subexperiment:

$$\Delta_{i,k} = \sum_{j=1}^{N_E} \alpha_j \mathbb{I}(i = j) + \epsilon_{i,k},$$

for $i = 1, 2, \ldots, N_E$ and $k = 1, 2, \ldots \lfloor \gamma N_P \rfloor$. Let $\widehat{\alpha}_{OLS,j}$ denote the OLS estimates of the α_j. Each $\widehat{\alpha}_{OLS,j}$ will simply be the average of the dependent variable for observations corresponding to that experiment, and so we recover the difference in sample means estimator: $\widehat{\alpha}_{OLS,j} = \widehat{\theta}_{j,1}$ for all experiments j. Letting $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_{N_E})'$, the ridge regression problem for a given regularization parameter λ is

$$\min_\alpha \left\{ \left(\Delta_{i,k} - \sum_{j=1}^{N_E} \alpha_j \mathbb{I}(i = j) \right)^2 + \lambda \alpha' \alpha \right\}. \tag{2}$$

Denoting the design matrix for this problem by X and the dependent variable by y, the solution is $\widehat{\alpha}_{Ridge} = (X'X + \lambda I)^{-1} X'y$. Since X is orthogonal, this implies $\widehat{\alpha}_{Ridge,j} = \frac{\lfloor \gamma N_P \rfloor}{\lfloor \gamma N_P \rfloor + \lambda} \widehat{\theta}_{j,1}$ for all j. Thus ridge regression shrinks all the raw estimates to zero by the same relative amount, where the shrinkage factor depends on λ.

Choosing λ by cross-validation, i.e. minimizing out-of-sample squared error in predictions on the data from the second subexperiments, amounts to solving:

$$\min_\lambda \sum_{i=1}^{N_E} \sum_{k=\lfloor \gamma N_P \rfloor + 1}^{N_P} \left(\Delta_{i,k} - \frac{\lfloor \gamma N_P \rfloor}{\lfloor \gamma N_P \rfloor + \lambda} \widehat{\theta}_{i,1} \right)^2.$$

Because $\widehat{\theta}_{i,2} = \frac{1}{N_P - \lfloor \gamma N_P \rfloor} \sum_{k=\lfloor \gamma N_P \rfloor + 1}^{N_P} \Delta_{i,k}$, this is equivalent to

$$\min_\lambda \sum_{i=1}^{N_E} \left(\widehat{\theta}_{i,2} - \frac{\lfloor \gamma N_P \rfloor}{\lfloor \gamma N_P \rfloor + \lambda} \widehat{\theta}_{i,1} \right)^2,$$

which in turn is equivalent to the original experiment splitting problem, $\min_\beta \sum_{i=1}^{N_E} (\widehat{\theta}_{i,2} - \beta\widehat{\theta}_{i,1})^2$. It follows that the experiment splitting gives the same estimators as cross-validated ridge regression. Similarly, the repeated experiment splitting procedure described in Algorithm 2 is equivalent to a variant of k-fold cross-validation, where for each fold, each experiment is randomly split into training and validation datasets. This connection motivates the naming convention of calling the first subexperiment the training subexperiment and the second the validation subexperiment, as equation (2) is estimated on the first dataset, and its tuning parameter λ is chosen by cross-validation against the second dataset.

Section 2.1 notes that with normal sampling error and treatment effects, the optimal shrinkage is linear. The Bayesian interpretation of ridge regression makes the same assumptions of normality on the treatment effects and the sampling error, so it is natural that it also implies linear shrinkage. Other penalties on the α parameter in equation (2) correspond to non-normal priors on treatment effects, and hence to fitting different, nonlinear regression models in the experiment splitting problem. The L_1 lasso penalty, for example, is equivalent to solving the experiment splitting problem with the soft-thresholding operator $\min_\alpha \sum_{i=1}^{N_E} \left(\widehat{\theta}_{i,2} - \text{sign}(\widehat{\theta}_{i,1})(|\widehat{\theta}_{i,1}| - \alpha)^+ \right)^2$ (see [24] for discussion of the relation between the lasso and the soft-thresholding operator).

Relative to the experiment-level regression of $\widehat{\theta}_{i,2}$ on $\widehat{\theta}_{i,1}$, the individual-level regression in equation (2) requires specifying the form of regularization term (and implicitly, the prior distribution of treatment effects). Inappropriate choices of regularization could lead to suboptimal forms of shrinkage. The L_2 penalty in ridge regression, implying linear shrinkage, may be inappropriate when the distribution of treatment effects is highly non-normal. The experiment-level regression of $\widehat{\theta}_{i,2}$ on $\widehat{\theta}_{i,1}$ requires no such assumptions, and will asymptotically recover the optimal shrinkage with any consistent nonparametric regression technique.

4 EXTENSIONS

4.1 Shrinkage Across Subgroups

Our focus has been on experiment splitting regressions where each data point in the regression corresponds to a separate experiment. We may also wish to perform shrinkage across non-overlapping subgroups of people within an experiment, either instead of, or in addition to, shrinkage across experiments. The experiment splitting methodology extends directly to this setting: instead of i indexing experiments, let i index subgroups (or subgroup by experiment pairs), and the same methodology applies without modification. Subgroup metrics which may be correlated with the magnitude of treatment effects or their sampling error variance, including demographics and subgroup size, are natural covariates to include in the experiment splitting regression.

4.2 Predicting Using Pooled Estimates

A variation on experiment splitting is to estimate $E(\theta_i \mid \widehat{\theta}_{i,1})$ as usual, but to generate predictions of θ_i by evaluating this conditional mean estimate $\widehat{m}$ at the *pooled* estimate $\widehat{\theta}_i$, instead of the subexperiment estimate $\widehat{\theta}_{i,1}$. Intuitively, evaluating $\widehat{m}$ at an estimator with the same mean and a lower variance should improve performance. We find this indeed results in a non-trivial decrease in prediction error in our data, but only if repeated experiment splitting is not already being used. A heuristic explanation can be given in the case of univariate OLS experiment splitting. For simplicity of exposition we assume the prior mean of θ_i to be zero. Let $\mathcal{D}$ denote all experimental data, and $\mathcal{A}$ the random assignments of people to subexperiments. Conditional on the data and averaging over repeated splits, the prediction from repeated experiment splitting as described in Algorithm 2 is $E_{\mathcal{A}}(\widehat{\beta}_1 \widehat{\theta}_{i,1} \mid \mathcal{D}) + O(1/R)$. The average prediction from repeated experiment splitting, evaluating the estimated conditional mean at the pooled mean $\widehat{\theta}_i$, is $E_{\mathcal{A}}(\widehat{\beta}_1 \mid \mathcal{D})\widehat{\theta}_i$. Under standard regularity conditions, for any fixed experiment i, $E_{\mathcal{A}}(\widehat{\beta}_1 \widehat{\theta}_{i,1} \mid \mathcal{D}) = E_{\mathcal{A}}(\widehat{\beta}_1 \mid \mathcal{D}) E_{\mathcal{A}}(\widehat{\theta}_{i,1} \mid \mathcal{D}) + O(1/N_E)$, as the

influence any single experiment has on the estimated regression parameter $\widehat{\beta}_1$ is $O(1/N_E)$. But $E_{\mathcal{A}}(\widehat{\theta}_{i,1} \mid \mathcal{D}) = \widehat{\theta}_i$, so with repeated experiment splitting, the difference between predictions using pooled estimates and predictions using subexperiment estimates must be of order $O(1/R) + O(1/N_E)$.

It may appear that the reason for this lack of performance improvement is that the function $\widehat{m}$, designed as it is for optimally shrinking the subexperiment estimate $\widehat{\theta}_{i,1}$, results in "overshrinking" the more accurate, pooled estimate $\widehat{\theta}_i$. A feasible procedure for transforming an estimate of $E(\theta_i \mid \widehat{\theta}_{i,1})$ into an estimate of $E(\theta_i \mid \widehat{\theta}_i)$ would help avoid this issue. One could estimate the former function by experiment splitting, transform it to an estimate of the latter function, and evaluate the latter function at the pooled estimate, $\widehat{\theta}_i$. It appears that performing this transformation in the general, multi-metric, non-normal case would effectively require estimating the full joint distribution of treatment effects and sampling error.[6] This would void the benefits that shrinkage via experiment splitting has in simplicity and ease of implementation, so we do not pursue this extension here.

5 EMPIRICAL RESULTS

5.1 Data

We study a set of 226 Facebook News Feed A/B tests conducted in 2018.[7] Each A/B test caused a change in the order in which stories appear in News Feed, with the goal of showing users more relevant content first. We analyze a single day's worth of data from each test. The outcome metric we study is the percentage change in posts per person to News Feed in the test group relative to the control group. In addition, for each test we track the percentage change in 23 auxiliary metrics, including likes, comments and link clicks. These will be used as covariates in our regressions, allowing us to make more precise inferences about the percentage change in posts per person. People in each test are randomly assigned with equal probability to the test and control groups. The sample size in each test varies between 10.0 and 10.9 million people.[8]

Figure 3 shows the distributions across experiments of the percentage change in posts, and its standard error. We calculate standard errors using the delta method. The percentage change in posts is non-normal: the Shapiro-Wilk test rejects the null hypothesis of normality with a p-value below 0.01%. This remains true after removing the experiments with a greater than 2% change in absolute value. This suggests that linear shrinkage, as implied by normal treatment effects and normal sampling error (and hence normal observed outcomes), is likely suboptimal.

Figure 4 shows the scatterplot of the percentage change in posts, for the training and validation subexperiments. There are two sets of points. The first is for $\gamma = 0.2$ (a 20%-80% split of data into

Figure 3: Distributions of Estimates and Standard Errors

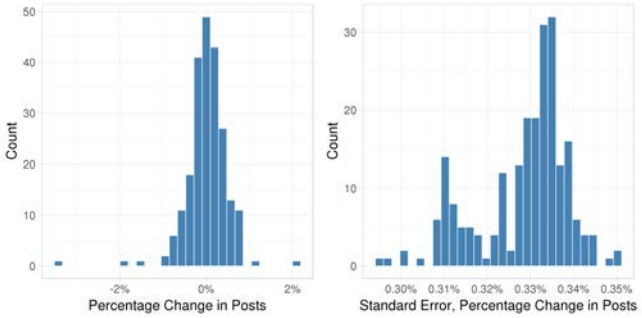
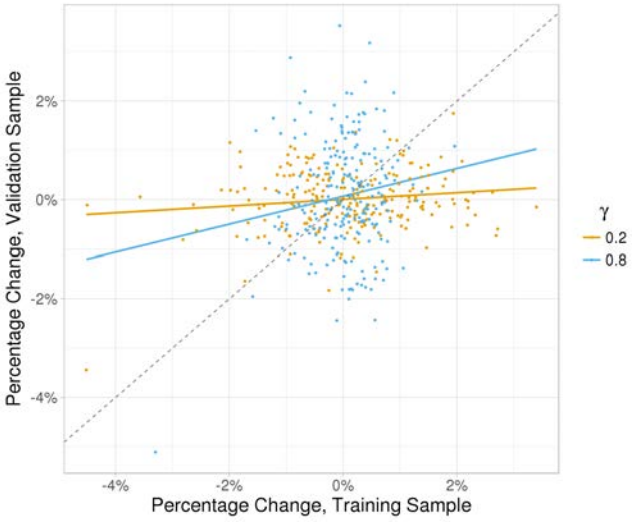


Figure 4: Percentage Change in Posts, $\gamma = 0.2, 0.8$



training and validation sets), and the second is for $\gamma = 0.8$. For each set of points, we plot the OLS line of best fit. The dashed gray line is the 45-degree line. For large N_E, the OLS slope coefficient is approximately $Cov(\widehat{\theta}_{i,2}, \widehat{\theta}_{i,1})/Var(\widehat{\theta}_{i,1}) = \sigma_\theta^2/(\sigma_\theta^2 + \sigma_{\varepsilon_1}^2)$, which is decreasing in $\sigma_{\varepsilon_1}^2$ and less than one.[9] Consistent with this, in Figure 4, increasing γ from 0.2 to 0.8 increases the slope of the line of best fit, but the slope remains less than than one.

5.2 Estimators

The estimators we consider fall into two classes. The first are the experiment splitting estimators, which fit predictive models for the percentage change in posts in the validation subexperiments, given the data from the training subexperiments. The second are the pooling estimators, which combine the data from the training

[6] The special case of a single metric with normal treatment effects and sampling error is straightforward. With a fraction γ of data in the training subexperiment, if $E(\theta_i \mid \widehat{\theta}_{i,1}) = \beta_1 \widehat{\theta}_{i,1}$, then it can be shown that $E(\theta_i \mid \widehat{\theta}_i) = \frac{\beta_1}{\lambda(1-\beta_1)+\beta_1} \widehat{\theta}_i$.

[7] This is a convenience sample, and is not intended to give a representative picture of the effects of News Feed tests generally.

[8] Estimates from smaller experiments should be shrunk more than estimates from larger experiments, all else equal. In practice the variation in sample sizes in this dataset is relatively small, and the gains from including sample size as an additional covariate in our predictive models were minimal.

[9] If $\gamma = 0.5$, the training and validation datasets have the same distributions, and it might be supposed that, by symmetry, the line of best fit through the scatterplot must have a slope of approximately one. This intuition is incorrect, as the formula for the slope coefficient shows.

and validation experiments. In either case we measure estimator performance by calculating the squared distance between estimator predictions and the percentage change in posts in the test subexperiments, on average across all experiments, as in Algorithm 1.

Experiment Splitting Estimators. The experiment splitting estimators all predict the percentage change in posts for the validation subexperiments. The estimators differ in the prediction algorithm used, and the covariates included. The covariates are either the percentage change in posts for the training subexperiments, or the percentage changes in all metrics in the training subexperiments. The predicted values from the estimated models provide the estimated treatment effect for each experiment. We study the following experiment splitting estimators.[10]

- *Univariate OLS*: Linear regression on a constant and the percentage change in posts in the training subexperiments.
- *Univariate Loess*: Locally quadratic regression, using the same regressors as univariate OLS.
- *Multivariate OLS*: Linear regression, using the percentage change in all metrics (posts, and all 23 auxiliary metrics) in the training subexperiments as predictors.
- *Lasso*: Lasso, using the same predictors as multivariate OLS.

In addition, we study the repeated experiment splitting versions of the above estimators, with R, the number of times the training and validation data is repeatedly split, equal to 10. Larger values of R result in only minimal improvements in estimator performance. The tuning parameters required for the loess and lasso models are selected by experiment-level cross-validation on the union of the training and validation data.

Pooling Estimators. The pooling estimators combine training and validation data. Unlike experiment splitting, which uses the validation observations to construct the outcomes and the training observations to construct the predictors, the pooling estimators make no distinction in which sample an observation is from.

- *Unshrunk*: The percentage change in posts in the combined training and validation test group, relative to the combined training and validation control group.
- *James-Stein*: The James-Stein type method of moments estimator described in [9], which allows for heteroskedastic sampling error.
- *Tweedie*: The nonparametric estimator based on equation (1), using Lindsey's method to estimate the density, as in [16].

Figure 5 plots the unshrunk treatment effect estimates against the unshrunk, James-Stein, lasso without repeated splits, and lasso with repeated splits estimators. The splitting estimators are calculated with $\gamma = 0.2$. By construction, the unshrunk points lie on the 45-degree line. The other estimators tend to shrink the unshrunk estimates towards the overall mean, but do so in different ways. James-Stein is an almost linear function of the unshrunk estimates, with a slope of less than one.[11] The lasso estimators are not smooth functions of the unshrunk estimator, reflecting the influence of auxiliary metrics. They sometimes shrink more and sometimes less

[10] Random forests and gradient boosted decision trees perform worse than multivariate OLS and lasso, and are omitted from the results.

[11] The James-Stein estimates are not exactly linear in the unshrunk estimates, because of the adjustment for heteroskedasticity, as in [9].

Figure 5: Unshrunk vs. Other Estimators, $\gamma = 0.2$

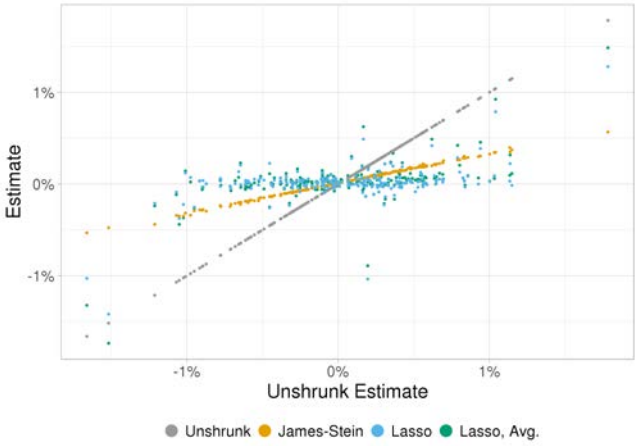
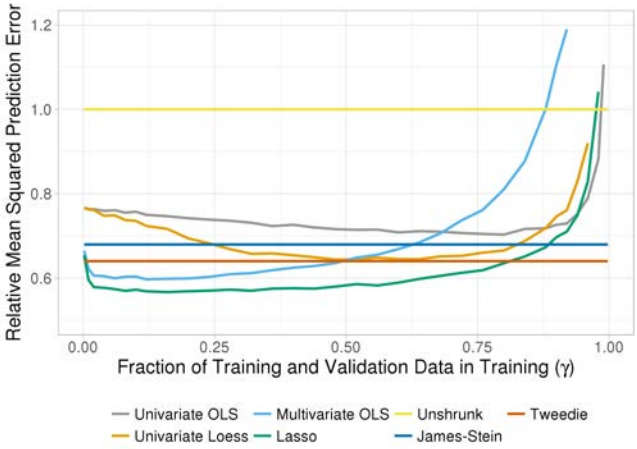


Figure 6: Pooled and Single Split Estimators, Error



than the James-Stein estimates, but on average both have larger variance than James-Stein.

Figure 6 shows the prediction errors for the basic experiment splitting estimators, and Figure 7 shows the prediction errors for the experiment splitting estimators with averaging over repeated splits. Both figures also show the three pooling estimators. The prediction errors are calculated as in Algorithm 1, with the number of splitting simulations S set to 100. The unshrunk estimator performs quite poorly. James-Stein shrinkage reduces mean squared prediction error (MSPE) by 32% relative to the unshrunk case. The non-normal distribution of effect sizes in Figure 3 suggests that Tweedie's estimator, which can adapt to a non-normal distribution of the truth, θ_i, should outperform James-Stein. This is indeed the case, with Tweedie having a 6% lower MSPE than James-Stein.

The unaveraged estimators, with the exception of univariate OLS, all perform better than James-Stein for an appropriately chosen

Figure 7: Pooled and Repeated Split Estimators, Error

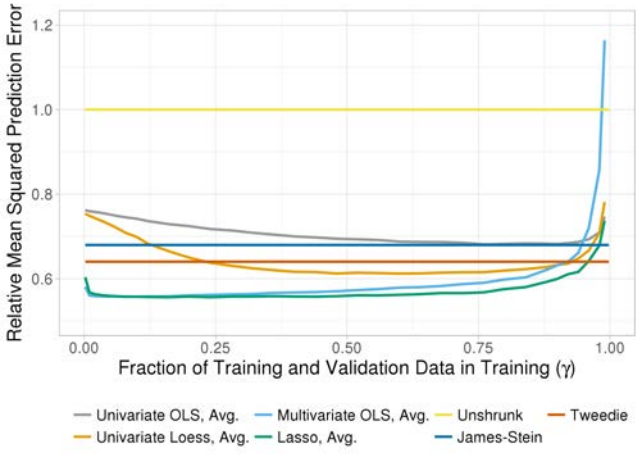


Table 1: Relative MSPE and Standard Errors, $\gamma = 0.24$

Estimator	Relative MSPE	Std. Error
Unshrunk	1.000	0.009
James-Stein	0.680	0.006
Tweedie	0.640	0.006
Univariate OLS	0.739	0.006
Univariate OLS, Avg.	0.717	0.006
Univariate Loess	0.682	0.006
Univariate Loess, Avg.	0.638	0.006
Multivariate OLS	0.603	0.006
Multivariate OLS, Avg.	0.562	0.005
Lasso	0.570	0.005
Lasso, Avg.	0.556	0.005

value of γ. Of particular note are the multivariate models, multivariate OLS and lasso. One might expect the comovements in auxiliary metrics to be informative: intuitively, if a experiment shows a large movement on the metric of interest, but not on auxiliary metrics which tend to covary with the metric of interest, we might infer that the large movement is likely a false positive due to sampling error. Indeed, the multivariate OLS and lasso estimators perform well. At their optimal splitting fractions they improve on the unshrunk estimator by 40% and 43%, and on Tweedie by 7% and 11%. This implies that our inferences about changes in posts can be much improved by taking into account how *other* metrics have changed, because of the correlation between the true treatment effects for posts and for other metrics. Thus experimental analyses which analyze one metric in isolation, without taking into account the comovements in other, related metrics may be quite suboptimal.

The MSPEs of the estimators which average over repeated splits are uniformly lower than their unaveraged counterparts. Repeated splitting reduces the minimum MSPE of multivariate OLS and lasso by 7% and 2%. The best performing estimator is the lasso averaged over repeated splits. It improves on the unshrunk estimator's MSPE by 44%, on James-Stein by 18%, and on Tweedie by 13%, and its minimum MSPE is reached at $\gamma = 0.24$. Table 1 shows all estimators' MSPE relative to the unshrunk estimator at $\gamma = 0.24$, along with standard errors summarizing statistical uncertainty caused by the finite number of splitting simulations, $S = 100$.

Table 2: Cumulative Treatment Effects, $\gamma = 0.24$

Estimator	Top 1%	Top 5%	Top 10%	All Positive
Unshrunk	0.81%	2.63%	4.36%	11.20%
Lasso, Avg.	2.34%	6.27%	6.83%	15.66%

Figure 6 also shows that choosing precisely the right γ for the multivariate repeated experiment splitting estimators is not necessary in order to obtain large performance improvements relative to the pooling estimators. This is especially true for the repeated lasso splitting estimator, as both averaging and the regularization of the lasso penalty have the effect of reducing the sensitivity of MSPE to the splitting fraction, γ. The repeated splitting lasso estimator has a roughly constant error for $\gamma \in [0.05, 0.75]$. By contrast the performance of the univariate OLS and loess estimators are considerably more sensitive with respect to γ, and their optimal γ is larger.

Some distinctive patterns in Figures 6 and 7 are the relatively flat MSPE curves for the lasso, and the observation that repeated splitting does not much decrease the lasso's minimal RMPSE. These features are consistent with a data generating process in which lifts in the most important covariates are precisely estimated, for any reasonable sample size. This implies that for γ above a small threshold, those covariate values will change little. If the error in the outcome variable is not too large (γ is not too close to one), the lasso gives high quality predictions, which will vary little both as a function of γ, and across repeated experiment splits.

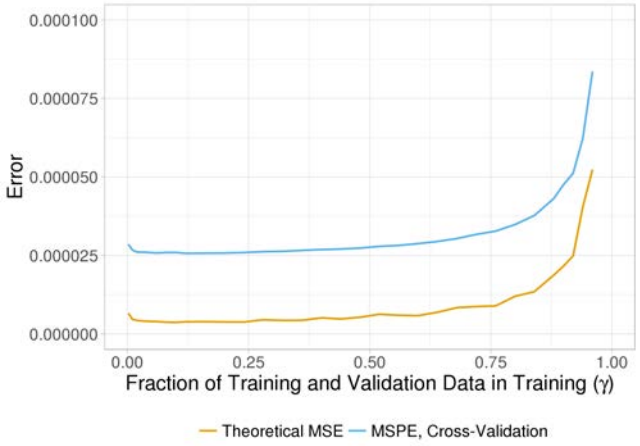
The purpose of A/B tests is typically to identify and launch the best performing treatments. Consequently MSPE may not be the most relevant evaluation criterion for practitioners. Instead, the key desideratum for an estimator may be whether the experiments it ranks highest do indeed have relatively large treatment effects. We can assess this by selecting the most promising experiments, as ranked by a given estimator, and evaluating their performance on the out-of-sample, test subexperiments. Table 2 shows that repeated lasso, the best performing estimator by MSPE, also performs well according to this criterion. The top 10% of experiments, as ranked by the unshrunk estimator, have a cumulative treatment effect of 4.36%.[12] The corresponding figure for the lasso is 57% higher, at 6.83%. The relative difference between the lasso and unshrunk estimators are even more pronounced in the far right tail, as the columns for the top 5% and top 1% of experiments show. Launching all estimates with a positive estimated treatment effect gives an estimated cumulative lift of 11.20% for the unshrunk estimator, while the figure for the lasso is 40% higher, at 15.66%.[13]

Proposition 1 allows us to estimate the mean squared error of the multivariate OLS experiment splitting estimator as a function of γ.

[12]The James-Stein estimator performs very similarly to the unshrunk estimator, as it results in almost the same ordering of experiments.

[13]The repeated lasso outperforms all other experiment splitting estimates according to all criteria in Table 2 except "Top 10%", where it is second to repeated multivariate OLS, which has a marginally higher cumulative treatment effect of 6.84%.

Figure 8: Multivariate OLS Error Estimates



This can be compared to the prediction error of multivariate OLS as computed by the cross-validation procedure of Algorithm 1. The latter is larger than the former for all γ, as it measures prediction error relative to the *estimated effects* in the test subexperiments, rather than error relative to the truth. However, Figure 8 indicates that estimating the optimal γ according to Proposition 1 performs comparably to cross-validation.

6 CONCLUSION

Inference on a particular metric and experiment can generally be improved by borrowing strength both across other experiments and other metrics. Experiment splitting provides a easily implementable methodology for realizing these performance gains, by transforming the problem of developing an appropriate shrinkage estimator into a simple prediction problem. In our sample of Facebook News Feed A/B tests, experiment splitting outperforms unshrunk estimators and conventional shrinkage estimators, especially after incorporating information from comovements in other metrics related to the outcome of interest. While our focus has been on average treatment effects in the context of experimentation, analogous "shrinkage via sample splitting" techniques are likely to be useful in causal inference more generally, with potential applications to estimating quantile treatment effects, local average treatment effects given instrumental variables, and average treatment effects under unconfoundedness.

ACKNOWLEDGMENTS

The authors are grateful to Peter Aronow, Susan Athey, Eytan Bakshy, Matt Goldman, Alex Peysakhovich, Jas Sekhon, Sean Taylor, Stefan Wager and Ark Zhang for useful comments and discussions.

APPENDIX

Proof of Proposition 1.

We have $\sum_{i=1}^{N_E} E(\theta_i - X_i'\widehat{\beta})^2 = N_E\sigma_{\varepsilon_2}^2 + \sum_{i=1}^{N_E} E(\widehat{\theta}_{i,2} - X_i'\widehat{\beta})^2 - 2\sum_{i=1}^{N_E} E\left(\varepsilon_{i,2}(\widehat{\theta}_{i,2} - X_i'\widehat{\beta})\right)$. Collect the $\widehat{\theta}_{i,2}$ and $\varepsilon_{i,2}$ observations across experiments i into the N_E-vectors $\widehat{\boldsymbol{\theta}}_2$ and $\boldsymbol{\varepsilon}_2$. Define the matrix $M = I - X(X'X)^{-1}X'$, with elements m_{ij}. Then

$$\begin{aligned} E\sum_{i=1}^{N_E} \varepsilon_{i,2}(\widehat{\theta}_{i,2} - X_i'\widehat{\beta}) &= E(\boldsymbol{\varepsilon}_2' M \widehat{\boldsymbol{\theta}}_2) \\ &= E\left\{E\left(\sum_i m_{ii}\varepsilon_{i,2}\widehat{\theta}_{i,2} \mid M\right)\right\} \\ &= E\left\{\sum_i m_{ii}\sigma_{\varepsilon_2}^2\right\} \\ &= \sigma_{\varepsilon_2}^2 E(\mathrm{tr}(M)) \\ &= (N_E - l)\sigma_{\varepsilon_2}^2. \end{aligned}$$

Hence $\sum_{i=1}^{N_E} E(\theta_i - X_i'\widehat{\beta})^2 = \left(\sum_{i=1}^{N_E} E(\widehat{\theta}_{i,2} - X_i'\widehat{\beta})^2\right) - (N_E - 2l)\sigma_{\varepsilon_2}^2$.

Proof of Corollary 1.

From standard results on the relation between the variance of regression residuals and the variance of regression errors (e.g. [25]),

$$\sum_{i=1}^{N_E} E(\widehat{\theta}_{i,2} - \widehat{\beta}_0 - \widehat{\beta}_1\widehat{\theta}_{i,1})^2 = (N_E - 2)E(\widehat{\theta}_{i,2} - \beta_0 - \beta_1\widehat{\theta}_{i,1})^2.$$

Further,

$$\begin{aligned} E(\widehat{\theta}_{i,2} - \beta_0 - \beta_1\widehat{\theta}_{i,1})^2 &= Var(\widehat{\theta}_{i,2}) - \frac{Cov^2(\widehat{\theta}_{i,2}, \widehat{\theta}_{i,1})}{Var(\widehat{\theta}_{i,1})} \\ &= \sigma_\theta^2 + \sigma_{\varepsilon_2}^2 - \frac{\sigma_\theta^4}{\sigma_\theta^2 + \sigma_{\varepsilon_1}^2}. \end{aligned} \tag{3}$$

The first equality follows from $\beta_1 = Cov(\widehat{\theta}_{i,2}, \widehat{\theta}_{i,1})Var(\widehat{\theta}_{i,1})^{-1}$, and the second from independence of sampling errors and treatment effects. Combining equation (3) and Proposition 1 gives the result.

Proof of Proposition 2.

With fractions γ and $1-\gamma$ of data in the training and validation subexperiments, we have $\sigma_{\varepsilon_1}^2 = \sigma_\varepsilon^2/\gamma$ and $\sigma_{\varepsilon_2}^2 = \sigma_\varepsilon^2/(1-\gamma)$. Define

$$h(\gamma, \sigma_\varepsilon^2, \sigma_\theta^2, N_E) = \frac{(N_E - 2)\sigma_\theta^2\sigma_\varepsilon^2/\gamma}{\sigma_\theta^2 + \sigma_\varepsilon^2/\gamma} + \frac{\sigma_\varepsilon^2}{(1-\gamma)}.$$

By Corollary 1, the problem of minimizing the mean squared error is $\min_{\gamma\in[\underline{\gamma},\overline{\gamma}]} h(\gamma, \sigma_\varepsilon^2, \sigma_\theta^2, N_E)$. Some calculations show

$$\frac{\partial h}{\partial \gamma} = \sigma_\varepsilon^2\left(\frac{1}{(1-\gamma)^2} - \frac{(N_E - 2)\sigma_\theta^4}{(\sigma_\varepsilon^2 + \gamma\sigma_\theta^2)^2}\right), \tag{4}$$

$$\frac{\partial^2 h}{\partial \gamma^2} = 2\sigma_\varepsilon^2\left(\frac{(N_E - 2)\sigma_\theta^6}{\left(\sigma_\varepsilon^2 + \gamma\sigma_\theta^2\right)^3} + \frac{1}{1-\gamma^3}\right). \tag{5}$$

From equation (5), $\frac{\partial^2 h}{\partial \gamma^2} > 0$. By convexity of h in γ, if $\left.\frac{\partial h}{\partial \gamma}\right|_{\gamma=\overline{\gamma}} < 0$ then $\gamma^* = \overline{\gamma}$, and if $\left.\frac{\partial h}{\partial \gamma}\right|_{\gamma=\underline{\gamma}} > 0$ then $\gamma^* = \underline{\gamma}$. Otherwise γ^* solves $\left.\frac{\partial h}{\partial \gamma}\right|_{\gamma=\gamma^*} = 0$. The limiting behavior of γ^* follows from these observations and equation (4). For large N_E, $\left.\frac{\partial h}{\partial \gamma}\right|_{\gamma=\overline{\gamma}} < 0$ and so $\lim_{N_E\to\infty} \gamma^* = \overline{\gamma}$. Arguments for limits wrt σ_θ^2 and σ_ε^2 are similar.

REFERENCES

[1] Hirotogu Akaike. 1998. Information theory and an extension of the maximum likelihood principle. In *Selected Papers of Hirotugu Akaike*. Springer, 199–213.

[2] Michael L Anderson and Jeremy Magruder. 2017. *Split-sample strategies for avoiding false discoveries.* Technical Report. National Bureau of Economic Research.

[3] Susan Athey and Guido Imbens. 2016. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences* 113, 27 (2016), 7353–7360.

[4] Susan Athey, Julie Tibshirani, and Stefan Wager. 2016. Generalized random forests. *arXiv preprint arXiv:1610.01271* (2016).

[5] Susan Athey and Stefan Wager. 2017. Efficient policy learning. *arXiv preprint arXiv:1702.02896* (2017).

[6] Eduardo M Azevedo, Alex Deng, Jose Luis Montiel Olea, Justin Rao, and E Glen Weyl. 2018. The A/B Testing Problem. In *Proceedings of the 2018 ACM Conference on Economics and Computation*. ACM, 461–462.

[7] Gerard Biau. 2012. Analysis of a random forests model. *Journal of Machine Learning Research* 13, Apr (2012), 1063–1095.

[8] Thomas Blake and Dominic Coey. 2014. Why marketplace experimentation is harder than it seems: The role of test-control interference. In *Proceedings of the fifteenth ACM conference on Economics and computation*. ACM, 567–582.

[9] Lawrence D Brown. 2008. In-season prediction of batting averages: A field test of empirical Bayes and Bayes methodologies. *The Annals of Applied Statistics* (2008), 113–152.

[10] Lawrence D Brown and Eitan Greenshtein. 2009. Nonparametric empirical Bayes and compound decision approaches to estimation of a high-dimensional vector of normal means. *The Annals of Statistics* (2009), 1685–1704.

[11] Bradley P Carlin and Thomas A Louis. 2010. *Bayes and empirical Bayes methods for data analysis*. Chapman and Hall/CRC.

[12] George Casella. 1985. An introduction to empirical Bayes data analysis. *The American Statistician* 39, 2 (1985), 83–87.

[13] Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. 2018. Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal* 21, 1 (2018), C1–C68.

[14] Victor Chernozhukov, Whitney Newey, and James Robins. 2018. Double/debiased machine learning using regularized Riesz representers. *arXiv preprint arXiv:1802.08667* (2018).

[15] Alex Deng. 2015. Objective bayesian two sample hypothesis testing for online controlled experiments. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 923–928.

[16] Bradley Efron. 2011. Tweedie's formula and selection bias. *J. Amer. Statist. Assoc.* 106, 496 (2011), 1602–1614.

[17] Bradley Efron. 2012. *Large-scale inference: empirical Bayes methods for estimation, testing, and prediction*. Cambridge University Press.

[18] Bradley Efron and Trevor Hastie. 2016. *Computer age statistical inference*. Cambridge University Press.

[19] Bradley Efron and Carl Morris. 1973. Stein's estimation rule and its competitors–an empirical Bayes approach. *J. Amer. Statist. Assoc.* 68, 341 (1973), 117–130.

[20] Bradley Efron and Carl Morris. 1975. Data analysis using Stein's estimator and its generalizations. *J. Amer. Statist. Assoc.* 70, 350 (1975), 311–319.

[21] Bradley Efron, Carl Morris, et al. 1976. Multivariate empirical Bayes and estimation of covariance matrices. *The Annals of Statistics* 4, 1 (1976), 22–32.

[22] Bradley Efron, Robert Tibshirani, John D Storey, and Virginia Tusher. 2001. Empirical Bayes analysis of a microarray experiment. *Journal of the American statistical association* 96, 456 (2001), 1151–1160.

[23] Marcel Fafchamps and Julien Labonne. 2017. Using Split Samples to Improve Inference on Causal Effects. *Political Analysis* 25, 4 (2017), 465–482.

[24] Jerome Friedman, Trevor Hastie, Holger Höfling, Robert Tibshirani, et al. 2007. Pathwise coordinate optimization. *The Annals of Applied Statistics* 1, 2 (2007), 302–332.

[25] F. Hayashi. 2011. *Econometrics*. Princeton University Press.

[26] William James and Charles Stein. 1961. Estimation with quadratic loss. In *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability*, Vol. 1. 361–379.

[27] Ron Kohavi, Alex Deng, Brian Frasca, Toby Walker, Ya Xu, and Nils Pohlmann. 2013. Online controlled experiments at large scale. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1168–1176.

[28] Colin L Mallows. 1973. Some comments on Cp. *Technometrics* 15, 4 (1973), 661–675.

[29] Whitney K Newey and James R Robins. 2018. Cross-fitting and fast remainder rates for semiparametric estimation. *arXiv preprint arXiv:1801.09138* (2018).

[30] Alexander Peysakhovich and Dean Eckles. 2018. Learning Causal Effects From Many Randomized Experiments Using Regularized Instrumental Variables. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 699–707.

[31] Herbert Robbins. 1956. An Empirical Bayes Approach to Statistics. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. The Regents of the University of California.

[32] Charles M Stein. 1956. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proc. Third Berkeley Symp. Math. Statist. Probab., 1956*, Vol. 1. Univ. California Press, 197–206.

[33] Charles M Stein. 1962. Confidence sets for the mean of a multivariate normal distribution. *Journal of the Royal Statistical Society. Series B (Methodological)* (1962), 265–296.

[34] Charles M Stein. 1981. Estimation of the mean of a multivariate normal distribution. *The annals of Statistics* (1981), 1135–1151.

[35] William E Strawderman. 1971. Proper Bayes minimax estimators of the multivariate normal mean. *The Annals of Mathematical Statistics* 42, 1 (1971), 385–388.

[36] Mark J Van der Laan and Sherri Rose. 2011. *Targeted learning: causal inference for observational and experimental data*. Springer Science & Business Media.

[37] Stefan Wager and Susan Athey. 2017. Estimation and inference of heterogeneous treatment effects using random forests. *J. Amer. Statist. Assoc.* just-accepted (2017).

[38] Stefan Wager, Wenfei Du, Jonathan Taylor, and Robert J Tibshirani. 2016. High-dimensional regression adjustments in randomized experiments. *Proceedings of the National Academy of Sciences* 113, 45 (2016), 12673–12678.

A Semi-Supervised Active-learning Truth Estimator for Social Networks

Hang Cui
Department of Computer Science, University of Illinois at Urbana-Champaign
hangcui2@illinois.edu

Tarek Abdelzaher
Department of Computer Science, University of Illinois at Urbana-Champaign
zaher@illinois.edu

Lance Kaplan
Army Research Laboratories
Adelphi, MD
lance.m.kaplan.civ@mail.mil

ABSTRACT

This paper introduces an active-learning-based truth estimator for social networks, such as Twitter, that enhances estimation accuracy significantly by requesting a well-selected (small) fraction of data to be labeled. Data assessment and truth discovery from arbitrary open online sources are a hard problem due to uncertainty regarding source reliability. Multiple truth finding systems were developed to solve this problem. Their accuracy is limited by the noisy nature of the data, where distortions, fabrications, omissions, and duplication are introduced. This paper presents a semi-supervised truth estimator for social networks, in which a portion of inputs are carefully selected to be reliably verified. The challenge is to find the subset of observations to verify that would maximally enhance the overall fact-finding accuracy. This work extends previous passive approaches to recursive truth estimation, as well as semi-supervised approaches where the estimator has no control over the choice of data to be labeled. Results show that by optimally selecting claims to be verified, we improve estimated accuracy by 12% over unsupervised baseline, and by 5% over previous semi-supervised approaches.

CCS CONCEPTS

• **Information systems → Collaborative and social computing systems and tools**.

KEYWORDS

Social Sensing, Truth Discovery, Semi Supervision, Active Learning , Maximum Likelihood Estimation

ACM Reference Format:
Hang Cui, Tarek Abdelzaher, and Lance Kaplan. 2019. A Semi-Supervised Active-learning Truth Estimator for Social Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313712

1 INTRODUCTION

This paper presents an active-learning based truth estimator for data collected on social media. We envision scenarios, such as disaster response, where a large number of unvetted local sources make claims about the state of events, possibly combining fact with rumor. A claim, in this paper, refers to the statement of a tweet, or a similar expression of fact (on social media) that may be correct or not. A major challenge in consuming such social media feeds lies in uncertainty regarding source reliability and thus difficulty assessing correctness of claims. Online sources who contribute the information are often unknown or have little prior reputation to consider. The problem becomes: is it possible to correctly filter true observations from the collective output of online sources of unknown reliability?

Our estimator computes, for each reported claim, the probability that the claim is correct. The estimator is novel in that it selects some well-chosen small subsets of claims for verification. We assume that another mechanism is available to verify the selected claims (i.e., give the estimator their ground-truth true/false label). Presumably, this external mechanism is costly and so has to be applied with care, which motivates the problem of selecting the best claims to verify from the perspective of maximally improving the overall accuracy of our estimator.

We extend a large body of work on fact-finding [5, 12, 14, 15, 19, 20] that estimates the reliability of sources iteratively with assessing the veracity of claims over the *entire* data set. We build on a state of the art estimator [3] that considers sources with significant error correlations. Those correlations arise from the *open* nature of online data. That is to say, sources are not independent. They can see each other's outputs and may occasionally copy others. Source non-independence creates opportunities for the spread of data contamination, as bad data (e.g., rumors) can be copied from one source to another. The non-independence of online open sources calls for estimators, such as the one used in this paper, that explicitly take source dependencies into account.

Finally, our estimator is unique in finding pivot observations whose true/false values, if verified, would maximally improve overall fact-finding accuracy. It selects a small set of pivot observations from the large volume of inputs for an external party (e.g., a human) to label. These labeled observations are then reinserted into the estimator to maximally improve its accuracy. The size of the pivot set can be either pre-defined or adjusted on the run. As shown in the evaluation, the above active selection of best observations to verify improves over semi-supervised approaches that do not have control over which claims are verified, or that use uncertainty in truth value alone as the basis for selection. To the authors' knowledge, ours is the first *semi-supervised* truth estimator for social media that *actively learns* the most judicious data claims to label (while accommodating non-independent sources).

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313712

We evaluate our truth estimator using both simulations and real data. We show that our approach significantly outperforms several baselines in a wide range of experimental conditions.

The rest of this paper is organized as follows. Section 2 describes the data model and problem statement. Section 3 presents the active-learning-based truth estimation algorithm. Section 4 evaluates the accuracy of estimation results using simulations. Section 5 presents empirical evaluation results. Observations and limitations are discussed in Section 6. Section 7 summarizes related work.

2 MODEL AND PROBLEM STATEMENT

Our active-learning truth estimator integrates social sensing data with human supervision to estimate the probability of correctness of individual claims. Unsupervised truth estimators of social network posts are constrained by uncertainty in reliability of arbitrary online sources. To improve accuracy over previous work, we allow for verifying a small faction ($< 5\%$) of social media posts. Our algorithm adopts an active learning structure that automatically determines the best set of posts to verify.

Let the set of all data sources be denoted by $\mathcal{S}$, where $|\mathcal{S}| = n$. The observations they make constitute the set of assertions, denoted by C, where $|C| = m$. The ith source and the jth assertion are referred to as $S_i \in \mathcal{S}$ and $C_j \in C$, where $i \in \{1, 2, \cdots, n\}$ and $j \in \{1, 2, \cdots, m\}$. The act of making an assertion by a specific source is called a *claim*. If source S_i makes an assertion C_j, we say logically that S_i claims $C_j = True$, denoted by claim $S_iC_j = 1$. The same assertion can be made by multiple sources at different times. The set of all claims made by all sources is represented by source claim matrix $\mathcal{SC}$, where each element is of the form $S_iC_j = value$, where $value = 1$ if source S_i makes claim $C_j = 1$ and $value = 0$ otherwise. Note that, the only natural language processing needed in this paper is to identity similar claims made by multiple sources and cluster them into one assertion. In this paper, we use simple cosine similarity between tweets as a distance metric. Previous work suggested that cosine similarity is sufficient for Twitter data[18]. Hence, clusters of tweets that are sufficiently close by this metric are considered to make the same assertion.

Our active-learning truth estimator selects a set of assertions to label by an external (accurate) entity in order to improve accuracy. Note that, we only verify a small ratio ($< 5\%$) of total assertions. Verification can incur human labor and cost, which should be kept as low as possible. Let us denote the set of labeled assertions as set $\bar{C}$, the size of labeled assertions as V, verification result as $\mathcal{V}$, in which $\mathcal{V}_j = 1$ and 0 means the verification result of assertion j is *True* and *False*, respectively. In this paper, we assume that human participants who verify $\bar{C}$ have full access to ground truth, such that all the labeled data are accurate. Our algorithm automatically estimates the best set of assertions to verify, which means the set $\bar{C}$ is not preset and is derived on the run.

Since we assume that sources are open (and hence their outputs can be seen by other sources), in general, a source may report something original or may copy an observation posted by others. This copying behavior is very common today on social media, causing fast spread of rumors and misinformation. If source S_a occasionally copies from source S_b, we say that S_a is *influenced* by S_b. Prior literature described algorithms for empirically detecting the existence of influence (manifesting in copying behavior) among sources [9]. Intuitively, these algorithms infer influence from unusual correlations among source outputs much in the same way cheating can be detected from correlated answers on exams. Since data are time-stamped, the direction of influence can be estimated, assuming that data published earlier influences data published later.

Table 1: List of Parameters

$\mathcal{S}$	Set of all sources
C	Set of all assertions
n	# sources
m	# claims
$\mathcal{SC}$	Source-claim matrix
$\bar{C}$	Set of labeled assertions
$\tilde{C}$	Set of unlabeled assertions
$\mathcal{V}$	Set of verification results
V	Number of labeled assertions
$\mathcal{Z}$	Estimated values of assertions
$\mathbf{A}$	Source original claim probability matrix
$\mathbf{P}$	Source dependent claim probability matrix
G	Influence network
$\mathbf{D}$	Dependency matrix
θ	Set of unknown parameters

We therefore assume the existence of an observed influence network, G, where nodes represent individual sources and directional edges denote the direction of influence between them. Note that, the existence of an edge in G between a parent node and a child node does not mean that the child always copies from the parent. It merely means that *some* claims of the child may be copied from the parent.

A claim S_iC_j is called original if no ancestors of S_i in the influence network, G, made the same claim with an earlier timestamp. Otherwise, we say the claim is dependent (meaning it might have been copied because an ancestor of node S_i in the influence graph made the same claim earlier). We use the indicator D_{ij} to denote claim dependencies. Specifically, $D_{ij} = 0$ indicates that claim S_iC_j is original. $D_{ij} = 1$ means the claim is dependent.

We use $\tau(C_j)$ to denote the actual truth value of assertion C_j. The goal is to identify the truth value of all observed assertions and select the best set of assertions to verify, given the source claim matrix $\mathcal{SC}$ and influence graph G.

3 EXPECTATION MAXIMIZATION FRAMEWORK OVERVIEW

3.1 Solution Overview

The estimator uses an expectation maximization framework that maximizes a pre-defined likelihood function. The algorithm jointly estimates source reliability and claim correctness, then uses estimates to select a set of assertions for human participants to verify. The labeled assertions are then combined with unlabeled data to compute improved estimates via EM framework.

Let $\mathcal{SC}$ denote the set of claims made in the interested interval. $\mathcal{S}$ and C are the corresponding sets of sources (who reported observations) and measured assertions (for which values were reported

in time-slot), respectively. Our goal is to estimate the truth value $\tau(C_j)$ for every assertion C_j, as well as to update the reliability of each source. We introduce two sets of parameters to characterize the behavior of sources. Specifically, for each source i:

- Let $\mathbf{A}^i$ denotes the *original claim probability* matrix, in which the entry $\mathbf{A}^i_{pq} = P(S_iC_j = p|\tau(C_j) = q, \mathbf{D}_{ij} = 0)$ denotes the probability that source S_i claims that assertion C_j has value p, when the ground truth value is q and the claim is original.
- Let $\mathbf{P}^i$ denotes the *dependent claim probability* matrix, in which the entry $\mathbf{P}^i_{pq} = P(S_iC_j = p|\tau(C_j) = q, \mathbf{D}_{ij} = 1)$ denotes the probability that source S_i claims that assertion C_j has value p, when the ground truth is q, and given that some ancestor of S_i in the influence graph, G, claimed $C_j = p$ earlier.

The estimated value of assertion C_j is denoted by the vector $\mathcal{Z}_j$. For a binary assertion, $\mathcal{Z}_j = \{z_{j,0}, z_{j,1}\}$, where $z_{j,q} = P(\tau(C_j) = q)$, is the probability that C_j has value q. The active-learning truth estimator proceeds in three stages:

- Estimating the true value of each assertion C_j and the reliability of each source S_i by EM framework: the estimator adapts an expectation maximization (EM) algorithm to jointly estimate reliability parameters of sources and truth values of variables from data. Given the claims made by sources, denoted by matrix $\mathcal{SC}$, and the source dependency matrix $\mathbf{D}$, the EM algorithm outputs the reliability for each source $S_i \in \mathcal{S}$ and the estimated truth value of each assertion $C_j \in C$. Note that, those estimates are unsupervised (No verification is involved in this step). They remain to be fused with a set of verified assertions, which are selected in next step.
- Selecting a set of assertions of size V to verify: Our truth estimator is blinded to ground truth (We do not have access to ground truth until we output the verification set). To estimate the optimal set of assertions to verify, we use the likelihood function as metric: the set which maximizes the likelihood function is selected. Note that, we do not have access to ground truth in this step, which means we have no information on verification results. Thus, for each assertion to verify, we need to estimate the effect of both *True* and *False*. Expected likelihood is then computed by averaging over all verification results. The probability of verification results is derived from truth estimates in step 1. The algorithm iterates through every possible sets of assertions to derive the optimal set. In later section, we also propose an heuristic approximation to dramatically reduce computation cost.
- Re-estimating truth value of assertions and reliability of sources with labeled data. After obtaining verification results from human participants, we combine labeled assertion with unlabeled ones to update our estimation via semi-supervised EM framework.

The above three steps are discussed in the following three subsections respectively.

3.2 Estimating Truth Value and Source Reliability via EM Framework[14]

Let us define the concatenation of source parameter matrices, $\mathbf{A}^i$ and $\mathbf{P}^i$, as the unknown parameter set $\boldsymbol{\theta}$ to be estimated. We adapt an expectation maximization algorithm to estimate these parameters based on matrix $\mathcal{SC}$ collected in the interested interval. The expectation maximization likelihood is formulated as log likelihood of observation matrix:

$$\theta_{MAX} = argmax_{\boldsymbol{\theta}}\{\log(\mathbb{P}(\mathcal{SC}|\boldsymbol{\theta}, \mathbf{D}))\} \tag{1}$$

The expected likelihood of observation matrix is formulated as multiplication of assertions:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\theta}}[\mathbb{P}(\mathcal{SC}|\boldsymbol{\theta}, \mathbf{D})] = \prod_{C_j \in C} \sum_{q=0}^{v} \mathbb{P}(SC_j, C_j = q|\boldsymbol{\theta}, \mathbf{D})$$

$$= \prod_{C_j \in C} \sum_{q=0}^{v} \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}, \mathbf{D})\mathbb{P}(C_j = q|\boldsymbol{\theta}, \mathbf{D}) \tag{2}$$

where v is the largest value for an assertion. For example, in the binary case, an assertion can take either the value 0 or 1. Hence, $v = 1$. SC_j is the jth column of SC, representing all the claims made on assertion j. $\mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}, \mathbf{D})$ represents the collective probability of all claims that said assertion $C_j = q$. It can be expressed as multiplication of all sources who made claims to assertion j:

$$\mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}, \mathbf{D}) = \prod_{S_i \in \mathcal{S}_j} \mathbb{P}(S_iC_j|C_j = q, \boldsymbol{\theta}, \mathbf{D}_{ij}) \tag{3}$$

where

$$\mathbb{P}(S_iC_j|C_j, \boldsymbol{\theta}, \mathbf{D}_{ij}) = \begin{cases} \mathbf{A}^i_{pq} & C_j = q, S_iC_j = p, \mathbf{D}_{ij} = 0 \\ \mathbf{P}^i_{pq} & C_j = q, S_iC_j = p, \mathbf{D}_{ij} = 1 \end{cases} \tag{4}$$

are the source reliability parameters.

Using E-step and M-step of standard EM-algorithm, we get the following iterative updates:

$$\mathcal{Z}_{j,p} = \frac{B_p}{\sum_{q=1}^{v} B_q}$$

$$\mathbf{A}^i_{pq} = \frac{\sum_{S_iC_j=p, \mathbf{D}_{ij}=0} z_{j,q}}{\sum_{S_iC_j, \mathbf{D}_{ij}=0} z_{j,q}}$$

$$\mathbf{P}^i_{pq} = \frac{\sum_{S_iC_j=p, \mathbf{D}_{ij}=1} z_{j,q}}{\sum_{S_iC_j, \mathbf{D}_{ij}=1} z_{j,q}} \tag{5}$$

where

$$B_p = \prod_{q=1}^{v} \prod_{S_iC_j=q} \mathbb{P}(S_iC_j|C_j = p) \tag{6}$$

The EM framework then iteratively updates the above formula until they converge.

3.3 Selecting Set of Assertions to Verify

Providing the estimated truth values via expectation maximization framework, we derive the set of verified assertions that maximize the expectation maximization likelihood. Let $\tilde{C}$ denote the set of labeled claims, $\bar{C}$ be the set of unlabeled claims, $\bar{C} = C - \tilde{C}$. $\mathcal{V}$ denotes the set of verification result, in which $\mathcal{V}_j \in \mathcal{V}$ is the verified

Algorithm 1 Semi-supervised Expectation Maximization

Input: Source-Claim matrix SC
Initialize $\boldsymbol{\theta}^{(0)}$, $\mathcal{Z}^{(0)}$, $\tilde{\boldsymbol{\theta}}^{(0)}$, $\mathcal{L}$ and $\tilde{C}$
while $\boldsymbol{\theta}^{(r)}$ does not converge **do**
  for C_j in C **do**
    Compute probability: $\mathbb{P}(S_iC_j|C_j, \boldsymbol{\theta}^{(r+1)}, \mathbf{D})$ according to (3)
  end for
  for S_i in S **do**
    Estimate parameters: $\mathbf{A}_{pq}^{i,(r+1)}$, $\mathbf{P}_{pq}^{i,(r+1)}$ and $\mathcal{Z}_j^{(r+1)}$ according to (5)
  end for
end while
Update EM likelihood $\mathcal{L}$ according to (2)
for $C' \subset C, |C'| = V$ **do**
  Compute expected likelihood estimation $\mathcal{L}_{C'}$ according to (12)
  if $\mathcal{L}_{C'} > \mathcal{L}$ **then**
    $\mathcal{L} = \mathcal{L}_{C'}$
    $\tilde{C} = C'$
  end if
end for
Label $\tilde{C}$ by human participants
while $\tilde{\boldsymbol{\theta}}^{(r)}$ does not converge **do**
  for C_j in C **do**
    Update parameters $\mathbf{A}_{pq}^{i,(r+1)}$, $\mathbf{P}_{pq}^{i,(r+1)}$ and $\mathcal{Z}_j^{(r+1)}$ according to (13)(14)
  end for
end while
Output: $\mathbf{A}_{pq}^{i}$, $\mathbf{P}_{pq}^{i,(r+1)}$ and $\mathcal{Z}_j^{k}$
END

result of assertion j. The expected complete data log likelihood conditioning on verification result $\mathcal{V}$ can be written as:

$$\begin{aligned} Q_{\mathcal{V}} &= \mathbb{E}_{\boldsymbol{\theta}_{\mathcal{V}}}[\log \mathbb{P}(SC|\boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V})] \\ &= \sum_{C_j \in \tilde{C}} \sum_{q=0}^{v} \mathbb{P}(C_j = q|\boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \log \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \\ &= \sum_{C_j \in \tilde{C}} \sum_{q=0}^{v} \mathbb{P}(C_j = q|\boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \log \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \\ &+ \sum_{C_j \in \bar{\tilde{C}}} \sum_{q=0}^{v} \mathbb{P}(C_j = q|\boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \log \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \end{aligned} \tag{7}$$

in which first term is the log likelihood of labeled assertions, the probability of assertion j of value q is

$$\mathbb{P}(C_j = q|\boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) = \begin{cases} 1 & q = \mathcal{V}_j \\ 0 & q \neq \mathcal{V}_j \end{cases} \tag{8}$$

The second term is the log likelihood of unlabeled assertions, it is derived according to EM algorithm described in section 3.2. The semi-supervised expectation maximization likelihood is then formulated as:

$$\begin{aligned} Q_{\mathcal{V}} = &\sum_{C_j \in \tilde{C}} \sum_{q=0}^{v} \mathbb{P}(C_j = q|\boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \log \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \\ &+ \sum_{C_j \in \bar{\tilde{C}}} \mathbb{P}(C_j = \mathcal{V}_j|\boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \log \mathbb{P}(SC_j|C_j = \mathcal{V}_j, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \end{aligned} \tag{9}$$

Our goal is to estimate the set of labeled assertions that achieve best semi-supervision results. To maximize the above likelihood over all possible combinations of labeled assertions, we adopt linear programming to estimate semi-supervised EM. The main observation is: Given the bipartite $\mathcal{SC}$ graph, as we verify an assertion (change from estimated truth value to ground truth), the reliability estimates of sources that make claims on the verified assertion are explicitly influenced: those sources gain higher reliability estimates since they correctly claim the verified assertion, while other sources slightly lose reliability estimates depending on their speak rates. As source reliability estimates change, other assertions are indirectly influenced. It can in turn affects more sources, however the changes are minimal. Thus in verification selection, we assume the estimated truth of non-verified assertions remain unchanged, which enables linear programming to simplify the above EM likelihood. The simplified log likelihood conditioning on $\mathcal{V}$ is:

$$\begin{aligned} Q_{\mathcal{V}} = &\sum_{C_j \in \tilde{C}} \sum_{q=0}^{v} \mathbb{P}(C_j = q) \log \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \\ &+ \sum_{C_j \in \bar{\tilde{C}}} \mathbb{P}(C_j = \mathcal{V}_j|\boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \log \mathbb{P}(SC_j|C_j = \mathcal{V}_j, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \\ = &\sum_{C_j \in \tilde{C}} \sum_{S_i \in \mathcal{SC}_j} \sum_{q=0}^{v} \mathbb{P}(C_j = q) \log \mathbb{P}(S_iC_j|C_j = q, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \\ &+ \sum_{C_j \in \bar{\tilde{C}}} \sum_{S_i \in \mathcal{S}_j} \mathbb{P}(C_j = \mathcal{V}_j|\boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \log \mathbb{P}(S_iC_j|C_j = \mathcal{V}_j, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) \end{aligned} \tag{10}$$

where $\mathbb{P}(C_j = q)$ is estimated in previous section 3.2, and

$$\mathbb{P}(S_iC_j|C_j, \boldsymbol{\theta}_{\mathcal{V}}, \mathbf{D}, \mathcal{V}) = \begin{cases} \mathbf{A}^i_{pq|\mathcal{V}} & C_j = q, S_iC_j = p, \mathbf{D}_{ij} = 0 \\ \mathbf{P}^i_{pq|\mathcal{V}} & C_j = q, S_iC_j = p, \mathbf{D}_{ij} = 1 \end{cases} \tag{11}$$

$\mathbf{P}^i_{pq|\mathcal{V}}$ describes dependent behaviours between two sources (follow or retweet) given verification result $\mathcal{V}$. $\mathbf{A}^i_{pq|\mathcal{V}}$ is the probability of making independent assertions providing the verification result is $\mathcal{V}$. Note that, the verification result $\mathcal{V}$ is unknown in this section. We hence need to compute the expected likelihood by averaging over all possible verification results. The probability of $\mathcal{V}$ is derived from truth estimation $\mathcal{Z}$ in previous section 3.2, such that $\mathbb{P}(\mathcal{V}) = \prod_{v_j \in \mathcal{V}} z_{j,v_j}$. The expected likelihood is then formulated as:

$$Q = \sum_{\mathcal{V}} \mathbb{P}(\mathcal{V}) Q_{\mathcal{V}} \tag{12}$$

The simplified EM likelihood can be maximized as a linear programming, with constraint $\sum_q \mathbf{A}^i_{pq|\mathcal{V}} = 1$, $\sum_q \mathbf{P}^i_{pq|\mathcal{V}} = 1$ and $|\bar{C}| = V$, where V is the number of assertions to be verified.

3.4 Re-estimating Truth Value and Source Reliability with Labeled Assertions

The optimal set of assertions is then verified by human participants. After obtaining the verification results, we re-compute truth estimates from semi-supervised EM likelihood formulated in (9). Using E-step and M-step described in section 3.2, we can derive iterative updates:

$$\mathcal{Z}_{j,p} = \begin{cases} \frac{B_p}{\sum_{q=1}^{v} B_q} & j \notin \bar{C} \\ 1 & j \in \bar{C}, q = v_j \\ 0 & j \in \bar{C}, q \neq v_j \end{cases}$$
$$\mathbf{A}^i_{pq} = \frac{\sum_{S_iC_j=p, \mathbf{D}_{ij}=0} z_{j,q}}{\sum_{S_iC_j, \mathbf{D}_{ij}=0} z_{j,q}}$$
$$\mathbf{P}^i_{pq} = \frac{\sum_{S_iC_j=p, \mathbf{D}_{ij}=1} z_{j,q}}{\sum_{S_iC_j, \mathbf{D}_{ij}=1} z_{j,q}} \tag{13}$$

where

$$B_p = \prod_{q=1}^{v} \prod_{S_iC_j=q} \mathbb{P}(S_iC_j|C_j = p) \tag{14}$$

Algorithm 2 Greedy Approximation

```
Input: Source-Claim matrix SC and Number of assertions to label V
Initialize C̄ = ∅, C̃ = C, L and d
Compute EM estimation according to (5)(6)
while |C̄| < V do
    for C_j ∈ C̃ do
        Compute expected likelihood L_j according to (16)
        if L_j > L then
            L = L_j
            Set j to be the verification index d
        end if
    end for
    Add C_j to C̄
end while
Re-Compute EM estimations according to (13)(14)
END
```

3.5 A Greedy Approach

The above verification algorithm takes V^d set of parameters to cover all d possible verification results of V assertions. Although each verification estimation takes minimal computation cost, the number of combinations can grow exponentially as the dataset. In this section, we describe a greedy approach to significantly reduce number of parameters on such large dataset. Start with $\bar{C} = \emptyset$, the approach adds one assertion to $\bar{C}$ via greedy approach (select the assertion that maximizes expected likelihood). We run the above iteration V times, getting one verified claim in each iteration. We may also redo expectation maximization to update source reliability and truth after certain number of verifications.

In this section, we analyze the greedy approach to demonstrate it is a good estimation. We study the effect of verifying one assertion. Denote the verified assertion as C_{ve}, estimated correctness before verification is z_{ve}. $\bar{C}$ is the set of labeled assertions (not including C_{ve}), $\tilde{C}$ is the set of unlabeled assertions. The likelihood before verifying C_{ve} is:

$$\begin{aligned} Q &= \sum_{C_j \in \bar{C}} \sum_{q=0}^{v} \mathbb{P}(C_j = q|\boldsymbol{\theta}_{ve}, \mathbf{D}) \log \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}_{ve}, \mathbf{D}) \\ &+ \sum_{C_j \in \tilde{C}} \sum_{q=0}^{v} \mathbb{P}(C_j = q) \log \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}_{ve}, \mathbf{D}) \\ &= \sum_{C_j \in \bar{C}} \sum_{q=0}^{v} \mathbb{P}(C_j = q|\boldsymbol{\theta}_{ve}, \mathbf{D}) \log \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}_{ve}, \mathbf{D}) \\ &+ \sum_{C_j \in \tilde{C} - C_{ve}} \sum_{q=0}^{v} \mathbb{P}(C_j = q) \log \mathbb{P}(SC_j|C_j = q, \boldsymbol{\theta}_{ve}, \mathbf{D}) \\ &+ \sum_{q=0}^{v} \mathbb{P}(C_{ve} = q) \log \mathbb{P}(SC_j|C_{ve} = q, \boldsymbol{\theta}_{ve}, \mathbf{D}) \end{aligned} \tag{15}$$

On verifying C_{ve}, with probability $\mathbb{P}(C_{ve} = q_{ve})$, the result is $C_{ve} = q_{ve}$. The expected likelihood after verifying C_{ve} is:

$$\begin{aligned} Q' &= \sum_{q_{ve}=0}^{v} \mathbb{P}(C_{ve} = q_{ve}) \times \\ &\{ \sum_{C_j \in \bar{C}} \sum_{q=0}^{v} \mathbb{P}(C_j = q|\boldsymbol{\theta}_{q_{ve}}, C_{ve}, \mathbf{D}) \log \mathbb{P}(SC_j|C_j = q, C_{ve}, \boldsymbol{\theta}_{q_{ve}}, \mathbf{D}) \\ &+ \sum_{C_j \in \tilde{C} - C_{ve}} \sum_{q=0}^{v} \mathbb{P}(C_j = q) \log \mathbb{P}(SC_j|C_j = q, C_{ve}, \boldsymbol{\theta}_{q_{ve}}, \mathbf{D}) \\ &+ \log \mathbb{P}(SC_j|C_{ve}, \boldsymbol{\theta}_{q_{ve}}, \mathbf{D}) \} \end{aligned} \tag{16}$$

Since we assume the estimated truth of non-verified assertions remain unchanged, thus for any assertion $j \in \tilde{C}, j \neq ve$, we have:

$$\mathbb{P}(C_j = q|\boldsymbol{\theta}_{C_{ve}, q_{ve}}, \mathbf{D}) = \mathbb{P}(C_j = q) \tag{17}$$

The effect of verifying an assertion is two-fold: Change on estimated correctness of verified assertion and change on estimated reliability scores of related sources.

Change on estimated correctness of verified assertion

First, we study the effect on estimated correctness of verified assertion C_{ve}. Estimates of C_{ve} changes from z_{ve} before verification, to 1 with probability z_{ve} and 0 with probability $1 - z_{ve}$. This part of likelihood is represented as the last item of (15) and (16). We can see the overall expected likelihood does not change, since the probability of verified results follows the correctness estimation of C_{ve}. Thus, multiple verified assertions are independent in terms of influence of estimated correctness.

Change on estimated reliability scores of related sources
Verifying an assertion changes source reliability scores, which implicitly affects estimated truth value of unlabeled assertions. Original claim probability matrix of sources are estimated as:

$$\mathbf{A}^i_{pq} = \frac{\sum_{S_iC_j=p} z_{j,q}}{\sum_{S_iC_j} z_{j,q}} \tag{18}$$

After verification, estimated reliability matrices become:

$$\mathbf{A}^i_{pq} = \begin{cases} \frac{\sum_{j\in C-C_{ve}, S_iC_j=p} z_{j,q}+1}{\sum_{j\in C-C_{ve}, S_iC_j} z_{j,q}+1} & \mathbb{P} = z_{ve} \\ \frac{\sum_{j\in C-C_{ve}, S_iC_j=p} z_{j,q}}{\sum_{j\in C-C_{ve}, S_iC_j} z_{j,q}} & \mathbb{P} = 1 - z_{ve} \end{cases}$$

Same applies to dependent claim probability matrix. Since verification affects both nominator and denominator in reliability estimates, it does change the expected likelihood. Multiple verifications interact dependently on source reliability scores. Providing we are only verifying a small set of assertions, we can make a further approximation that the changes on source reliability are positive (With high probability, a verification overall improves source reliability estimation). With this assumption, the problem becomes submodular and monotone, and thus greedy algorithm gives a $(1-1/e)$-approximation solution to our problem. However, in sparse networks, the above assumption does not hold and will cause slight performance drop.

4 EVALUATION USING SYNTHETIC DATA

We evaluate our algorithm using both real and synthetic data sets. This section describes evaluation using synthetic data. Evaluation using real-world data is presented in the next section.

Our use of synthetic data admittedly misses realistic qualities of target applications. The advantage, however, lies in exercising full control over ground truth, nature of assertions, and nature of sources, hence allowing evaluation under a broad set of conditions. By changing parameters of synthetic data generation, we are able to comment on general trends and on robustness of our results across a wide range of conditions. This broader picture complements our more localized (if accurate) understanding derived from the point cases represented by the specific collected data sets (in the next section).

We develop a synthetic data generator that creates n sources $\{S_1, S_2, \cdots, S_n\}$, of which some fraction are not reliable. Each source makes a subset of assertion selected from some larger set maintained by the generator. The generator works as follows:

- First, it generates m assertions $\{C_1, C_2, \cdots, C_m\}$ to choose from, and chooses source reliability and speak rates for each of the n sources. Reliability matrices and speak rates are generated as Gaussian distributions of preset parameters. A control parameter P_{true} determines the percentage of true assertions in the overall set. Hence, turning that parameter down would increase the fraction of "rumors" compared to true statements among the data reported.
- Second, the generator generates claims in accordance with the chosen source parameters and populates the set $\mathcal{SC}$ for the period of simulation. It loops over all sources. In each round, the speak rate of a source, P_{speak}, decides if the source will make a claim. If so, another probability, P_{depend}, decides whether or not this claim will be a repeat of a previous one (e.g., retweet). A repeat is generated in accordance with the dependency graph by repeating some claim of an ancestor in the graph. If P_{depend} decides that the claim is an original claim, then either a true or a false assertion is chosen according to reliability parameters of this source.
- The generated claims using the above method are given to our set of algorithms to analyze.

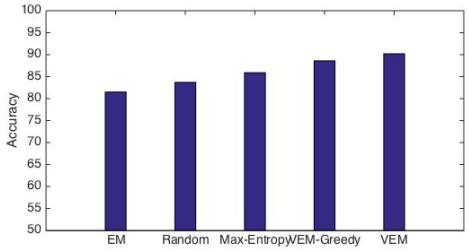


Figure 1: Simulation accuracy

In the simulation, reliable sources have expected 70% chance of making correct claims, while unreliable sources have expected 40%. We evaluate five different methods:

- EM: The pure *EM* uses expectation maximization but without any verification. It jointly estimates source reliability and assertion truth values from source claim matrix *SC* and dependency graph *G*. The original algorithm was published in [14].
- Random: It augments *EM* by randomly selecting the assertions to verify.
- Max-Entropy: The *Max-Entropy* algorithm picks the assertions that have the closest probability of true and false values (i.e., the most uncertainty in estimated value).
- VEM: *VEM* is the active-learning algorithm described in section 3.2 - 3.4.
- VEM-Greedy: It is the Greedy approximation of *VEM*, described in 3.5.

We conduct 50 independent experiments for each data point. For data generation, unless mentioned otherwise, we use number of sources $n = 50$, speak rate $P_{speak} = 0.2$, expected reliability of good sources = 0.7 and expected reliability of bad sources = 0.4. A total of $m = 250$ claims are generated in each simulation. We limit the assertions to verify to 5%. In general, we expect our algorithm to work better when sources are more predictable (e.g., good sources are nearly 100% accurate, and bad sources are always inaccurate). The above settings (of 70% and 40% reliability) were meant to increase entropy in order to stress test the algorithm.

4.1 Simulation Accuracy

We compare the simulation results of the five algorithms to ground truth. From Fig. 1, we see that our *VEM* algorithm outperforms all other algorithms. The *Random* selection of assertions to verify can occasionally correct a claim but is inferior to others. The *Max-Entropy* algorithm selects more uncertain assertions, but it is not the best choice, since it does not take into account the global

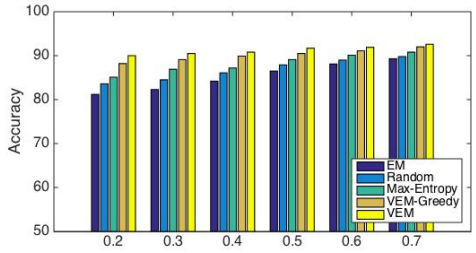


Figure 2: Vary Claim Prob.

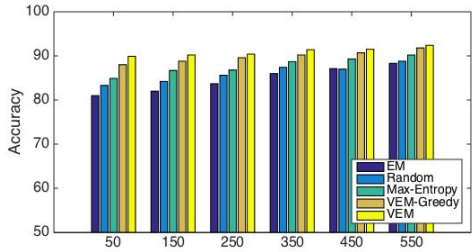


Figure 3: Vary # of Sources

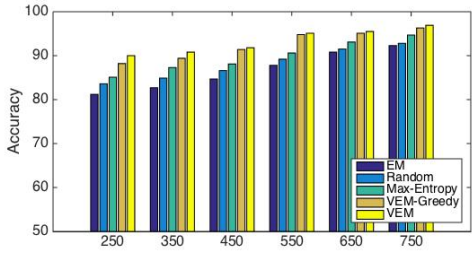


Figure 4: Vary # of Assertions

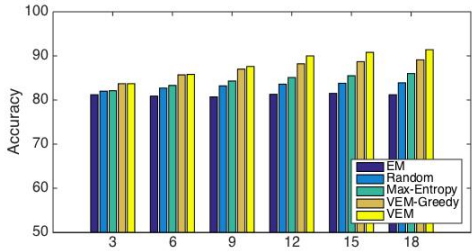


Figure 5: Vary # of verified assertions

interaction between assertions. Our algorithm achieves better accuracy than the above two methods. The reason is two-fold. First, *VEM* jointly considers an assertion's value uncertainty and its interactions with other assertions. For example, if only one unknown source made this assertion then knowing its true value has a more limited effect (on understanding source reliability) than if several unknown sources made this assertion (in which case the verification will yield information pertinent to several sources at once). Second, *VEM* selects the best set of assertions to verify by iteratively estimating verification outcome. *VEM-Greedy* provides a low cost approximation to our *VEM* algorithm to reduce computational complexity. It achieves comparable accuracy but is much lighter. Next, we study the effect of varying data set generation parameters. In each experiment, we vary one parameter while keeping others unchanged. The simulator generates a new dataset of each set of parameters. First, we vary the parameter P_{speak}, the probability of a source making claim, from 0.2 to 0.7 with increments of 0.1. This has the effect of modulating the density of the source claim matrix. In Fig. 2, our *VEM* achieves the best accuracy for every speak rate setting. *VEM-Greedy* has comparable performance to *VEM* and still beats the other three algorithms. Note that, although *VEM* has better accuracy than *VEM-Greedy*, it iterates through all possible combinations of assertions, which is computationally expensive in a large dataset. Another observation is that, at low speak rates, our algorithm significantly outperforms *EM*, *Random* and *Max-Entropy*. The reason is, in a sparse source claim matrix, *EM* produces more uncertain results: more truth estimations fall close to 0.5 than 0 or 1. *VEM* provides better improvements over baselines on such sparse graphs. In the second experiment, we change the number of sources n from 50 to 550 in increments of 100. In Fig. 3, our algorithm outperforms the others over the entire range. Adding sources has a similar effect on accuracy as increasing speak rate. In our default setting, with $P_{speak} = 0.2$, the graph is sparse, which impairs the performance of the *EM* algorithm. When we increase the number of sources, the number of edges connecting to each assertion is increased. This leads to an improvement in performance. Again, our algorithm outperforms others for every setting of parameters. In third experiment, we vary the number of assertions from 250 to 750 in increments of 100. The speak rate P_{speak} and the number of sources are unchanged. In Fig. 4, the baseline *EM*, *Random* and *Max-Entropy* have similar trends as in the previous two simulations. Our *VEM* and *VEM-Greedy* improve slightly more when increasing the number of assertions. The reason is: since we always verify 5% of the total number of assertions, having more assertions allows us to verify more as well, deriving better value due to their judicious selection of claims to verify. The immediate implication is that our algorithm fits well with larger data sets.

In the next simulation, we study the effect of varying number of verified assertions. We change the this number from from 3 to 18 with increment of 3, keeping the total at 250 (i.e., roughly 1.2% to 7.5%). In Fig. 5, our algorithm is shown to outperform the others in all 6 scenarios. There are two observations:

- We observe an increasing performance gap between *VEM* (*VEM-Greedy*) and other algorithms as we increase number of verified assertions. *Random* and *Max-Entropy* use a passive learning approach, which has no control over labeled data. Thus, they are susceptible to selecting less advantageous assertions. Our *VEM* algorithm determines the best assertions to verify offering a better advantage as the number of verified assertions increases.
- The improvements in total accuracy achieved slow down after 12 assertions. This demonstrates 5% is a good fraction for pivot assertions.

4.2 Compare Computation Time

Here we compare the computation overheads of different algorithms. We increase the size of the source claim matrix by increasing the number of sources and assertions as indicated on the horizontal axis. We then average the time needed to process the data sets over

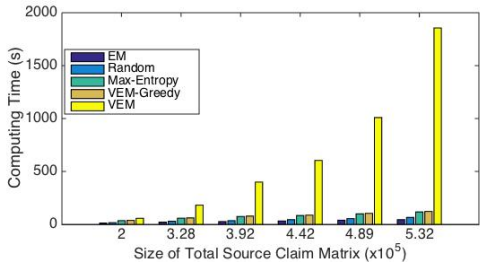


Figure 6: Computation Time Comparison

20 independent runs. The result is shown in Fig. 6. *Random*, *Max-Entropy* and *VEM-Greedy* have slight overheads over baseline *EM*. Our *VEM*, however, grows exponentially with the number of claims. The simulation indicates that *VEM* is good for smaller datasets but does not scale well to larger ones. However, *VEM-Greedy*, its greedy approximation, has a roughly similar overhead as *Random* and *Max-Entropy*, yet still achieves better results than other methods.

Results in this section confirm two main observations. First, the accuracy improvement attributed to the new algorithm holds true across a broad set of conditions on sources, assertions, percentage of verified assertions, and overall sparsity of the Source-Claim matrix. Overall, better advantages are seen in larger experiments. Second, the quality of the results is fairly robust to changes in synthetic data generation. Hence, when discussing specific data sets in the next section, we have reason to believe that the observed performance advantages are not coincidental to only those specific data sets and configurations. With that, we turn to results obtained when experimenting with real data.

5 EMPIRICAL EVALUATION

In this section, we evaluate our algorithm on four empirical data sets collected on Twitter in 2018 and 2015. The data were collected using Twitter API. Each data set was collected by specifying event keywords or a geographic location. The data collection tool would then collect tweets that contain the indicated keywords or originate from the indicated location. Four datasets were created, labeled: (1) Hurricane, (2) Facebook, (3) Kirkuk, (4) LA Marathon, representing the events of: (1) Hurricane Florence in 2018, (2) Facebook under federal investigation, (3) Battle with ISIS in Kirkuk city, and (4) A marathon in LA:

- Hurricane Florence: In September 2018, Hurricane Florence hit the US East Coast, including North Carolina, South Carolina and Maryland. The disaster caused 53 deaths and over 38 *billion* USD dollars in damage over its course.
- Facebook Federal Investigation: In 2018, Facebook faced a federal investigation into its sharing of user data with the political consulting firm Cambridge Analytica. Some 87 million facebook users' data were claimed to be improperly shared.
- Kirkuk: On March 10th 2015, Kurdish forces in Northern Iraq attacked the self proclaimed "Islamic State" (ISIS) outpost, west of the city of Kirkuk. Battles raged and a lot of commentary followed on social media about state of affairs on the ground.
- LA Marathon: This data set collects tweets about the Los Angeles Marathon on March 15th, 2015. The runners started from Dodger Stadium and ran a course to the coastline in Santa Monica, passing through downtown Los Angeles, Hollywood, the Sunset Strip, Santa Monica Blvd, and Ocean Avenue, among other landmarks. Much Twitter activity followed the event as it unfolded.

Table 2: Information Summary of Twitter Dataset

	#Assertions	#Sources	#Claims
Hurricane Florence	9052	7645	9378
Facebook	10238	8236	11395
Kirkuk	6172	4816	6188
LA Marathon	7072	5174	7148

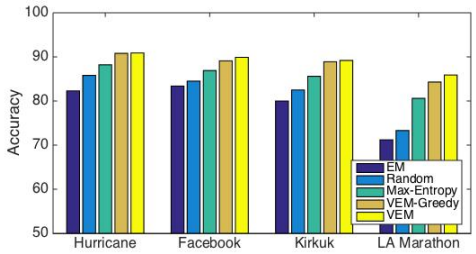


Figure 7: Empirical evaluation

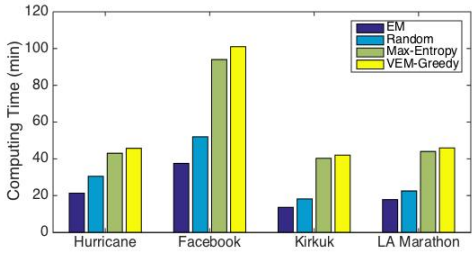


Figure 8: Computation Time

Our intent was to use data sets that differ significantly in time (e.g., today versus three years ago), and differ significantly in topic (e.g., social event, political event, or natural disaster). To evaluate our algorithm, the most popular 1000 assertions were selected (i.e., those supported by the highest number of claims from sources). Each algorithm then computes the probability of each of these 1000 assertions to be true. To avoid variations in the number of assertions returned, the highest-probability 200 assertions (ranked true) by each algorithm are computed. We then manually grade the correctness of those tweets by human graders. To prevent bias, the graders have no information on which algorithm generated which output. Each tweet is marked as "True", "False", or "Unconfirmed" by the grader. An "Uncomfirmed" tweet is either non-verifiable or is an opinion (a subjective assessment, such as 'Facebook is user-friendly'). We then compute accuracy as the percentage of assertions labeled "True" by the grader in the top 200 returned by each algorithm: $\#True/(\#True+\#False+\#Unconfirmed)$. Note that our accuracy analysis presents a pessimistic performance evaluation, since we count unconfirmed assertions as "False".

The results of this evaluation are shown in Fig. 7. Consistently with simulations, our algorithm outperforms all baselines. The

reason is: *VEM* iterates through every combination of assertions to estimate the semi-supervised EM likelihood, which is theoretically more accurate than other heuristic methods. *Random* selection and *Max-Entropy* fail to select the best assertions to verify.

Our algorithm improves accuracy by 8.6%, 6.5%, 9.2%, 12.7% respectively over baseline *EM* when only 5% of the total assertions are verified. Said differently, it reduces error roughly in half (from the neighborhood of 20% to the neighborhood of 10%). Thus, it does significantly improve truth finding quality in social sensing systems. In addition, the lighter version, *VEM-Greedy*, has comparable performance in all four datasets.

Lets take a deeper look at those algorithms. *EM* is the baseline approach. It allows us to understand the benefits of active learning. Prior work shows that *EM* outperforms traditional iterative algorithms (i.e., Sum, Average Log, Truth Finder, in social sensing system) [18]. The main sources of inaccuracy in the unsupervised *EM* are: (1) It requires relatively dense data to have better accuracy. (2) Our evaluation counts subjective opinions as False. Subjective assessments made by reliable sources get high credibility scores, which reduces accuracy in our evaluation.

Random reflects performance of semi-supervision if the set of labeled assertions was not controlled (i.e., when labeled data are arbitrarily chosen). It allows us to assess the performance improvement due to a good choice of assertions to verify. Similarly, the original *Max-Entropy* shows that a good choice depends on more than just the uncertainty in the value of the assertion. *Max-Entropy* improves accuracy over *Random* by verifying the most uncertain assertions. Our algorithm does better because it succeeds in finding assertions that are connected in the source claim graph in such a way that their verification potentially benefits other estimated values. Hence, by verifying a small set of assertions, we also improve the estimation of source reliability, which influences the estimation of other assertions. *VEM* uses linear programming to find the best assertions to verify. It does incur a high computational cost. Thus, the lighter version of *VEM*, *VEM-Greedy* is derived. Empirical results show that it still outperforms *Random* and *Max-Entropy*.

Finally, Fig. 8 gives a comparasion of computation time on real datasets. *Random* has similar execution time as baseline *EM* since the algorithm only does one round of EM estimation. *Max-Entropy* and *VEM-Greedy* does two rounds of EM because those two algorithms select assertions to verify based on EM estimates then re-estimate with labeled assertions. *VEM* takes 2276, 3365, 350 and 564 minutes to execute due to its high combinatorial computation cost. Note that, *VEM* does not need to iterate over every combinations in real datasets, since some subsets of assertions can be eliminated by heuristics.

The evaluation shows that across a diverse selection of Twitter data sets from different times and topics, our algorithm, *VEM*, improves accuracy, compared to the baselines in a manner largely consistent with simulation results. Moreover, *VEM-Greedy* offers nearly the same performance at a much lower overhead.

6 DISCUSSION AND FUTURE WORK

The work presented in this paper can be extended in several ways.

Errors and imperfections: The accuracy of our approach is contingent on the accuracy of several underlying components. For example, we group claims by cosine similarity, but semantically-aware clustering methods may yield better results. We assume an "oracle" that allows the selected claims to be reliably verified, but the oracle itself might be wrong. Not all assertions might be verifiable by an oracle. In some scenarios, only a subset can be verified. Our model relaxation that enables the greedy algorithm to perform well does not hold when networks are sparse. This deficiency partially explains the performance gap between the expensive form of inference and the greedy form. A more thorough analysis is needed in order to assess the sensitivity of our approach to these and other sources of error.

One-hot constraints: Our approach as discussed evaluates the truth values of individual assertions independently. In reality, there may be "one-hot" constraints on the value that is true. For example, different assertions might describe a suspect in a street shooting differently, but if the descriptions are conflicting, only one description is true. Our algorithm can be easily extended to accommodate such constraints in scenarios where it is possible to identify claims that are mutually conflicting. This might be easier to do for more structured data, such as crowdsensing data, where users report multiple-choice observations via an app.

Scalability: One important feature of crowdsourcing systems is scalability. Our algorithm runs under a batch setting, which means all data are considered at the same time in the estimation. In practice, this is not always a concern. Often one desires to verify information on specific claims or topics, which allows pre-filtering of larger datasets to the issue of concern. However, should one decide to run the algorithm on larger data, one possibility is to use an incremental form. Specifically, previous work presented online recursive truth estimators based on an EM framework that allow propagating information from one window to the next as priors [18] to be undated incrementally in the current window. This allows the algorithm to work on one slice of data at a time, making it more scalable.

Two changes must be made to support such an online extension. First, time should be slotted into intervals. The EM framework would only be applied to claims collected in the current time-slot. Posterior beliefs are then computed from EM estimates of current and previous time-slots, summarizing source reliability and claim correctness probabilities from previous history. Posterior beliefs then become prior beliefs in the next time-slot and are included in online EM estimation [18]. Second, since user queries can be made at any time-slot, an interesting challenge is whether the labeling overhead can be distributed more judiciously over time. For example, if no significant uncertainty exists, maybe spare some verification budget for the next window.

Collusion: As with many fact-finding and reputation systems, one challenge is malicious attacks. A group of collaborators or multiple replicas generated by a single user can attempt to fool the system to boost correctness of false assertions. This can be done by gradually gaining trust from providing true observations, then coordinating on delivering the same false message(s). While it is virtually impossible to detect a single such manipulation, repeated manipulation over time is detectable. Previous literature proposed mechanisms to detect such collaborations based on several observations on social networks. For example, collaborating groups tend to coincidentally make observations on the same set assertions repeatedly [10] and

tend to have more friend/follow links among themselves than external links to honest users [8]. Our assertion selection can combine those features to better assess source reliability in the presence of malicious attacks.

Verification cost: We implicitly assume that every verification attempt has the same cost, leading to a formulation where only the total fraction of verified assertions is bounded. Our algorithm can be further extended to the settings where verification of different assertions has a different cost. The problem becomes to select an optimal set of assertions while keeping cost below a preset value. Extending *VEM* algorithm is straight-forward, since it iterates through every possible set of assertions and estimates their verification effect. *VEM-Greedy*, however, has to consider verification cost in some appropriate manner since it selects assertions greedily one at a time. A simple solution is to select the assertion which has highest approximate improvement per unit cost.

Natural language processing: Our approach minimizes the use of natural language processing and does not interpret the content of tweets, which makes it easily applicable to any language, dialect, or topic with no prior language training. If tools for natural language processing were to be used, they would improve our truth estimator in several ways. For example, the same observation can be expressed in multiple ways or languages, which we shall fail to recognize as the same assertion. Some natural language processing could mitigate this problem.

Applicability and real world scenarios: We are currently working on integrating our algorithm into our Apollo social sensing toolkit (http://apollo2.cs.illinois.edu). Apollo extracts tweets hourly using the Twitter API and processes them to discover real-world events related to a given broad topic, such as disasters, world conflicts, local traffic, or other categories, supplied by the user as query terms. It further filters descriptions of these events, retaining only what it believes to be the most trustworthy claims (tweets). Using the approach described in this paper, the tool can be extended by selecting a very small subset of tweets and asking the user to assess their veracity. In return, our algorithm will improve the quality of assessment and filtering of the majority of claims. This capability can be applied in many contexts. For example, in post-disaster scenarios, rumors often proliferate about the extent of damage, availability of help, and instructions for survivors. Apollo can help mitigate the confusion. More recently, actors on social media were blamed for misinformation campaigns, for purposes ranging from intimidation to political manipulation. Apollo may offer a countermeasure that reduces information contamination. Finally, in the absence of disasters and conflicts, the tool can be used simply as an online (near) real-time news source, distilling myriads of evolving events, on a topic of user choice, into a much smaller amount of higher-quality newsworthy headlines. In the authors' own lab, Apollo is being used by graduate students to catch up on the most salient events in their home countries, especially those that do not receive much coverage in the local news. This is an anecdotal proof of value of the tool as a new personalizable medium to keep up with evolving real-time events worldwide.

7 RELATED WORK

This paper builds on advances in social sensing, which evolved from participatory sensing, initially introduced in [2]. Recent applications include BigRoad[6], RST[7], and Nutrilyzer[13].

Reliability and data quality are among the key problems of social sensing. To eliminate conflicts among multi-source data, the topic of truth-discovery has been studied extensively in machine learning/data mining [5, 19, 20] and social network literature [1, 15–17].

Early truth finders were based on the concepts of hubs and authority, in which the source reliability and claim trustworthiness updates each other in an iterative fashion. Yin et al. utilized interdependencies between source trustworthiness and fact confidence, introducing TruthFinder as an unsupervised fact finder. Later work extended TruthFinder by introducing heuristics into the analysis, including: small set of pre-labeled truth [20], incorporating prior knowledge [11], similarity between claim values [4] and extra parameters to source reliability [12]. Wang et al. [15] first formulated fact finding into a maximum likelihood estimation problem and jointly estimated source reliability and claim correctness using an expectation maximization approach. Subsequent work [14, 16] extended the original EM by taking into account dependencies.

The above work assumed a batch setting in which all data have to be available at once. However, the amount of data can grow exponentially over time, and the gap between staring and finishing of data collection can be large, which leads to high computation cost and slow response to queries. Yao et al. proposed a recursive estimator for streaming social media data[17]. The online recursive estimator adapted a batch EM framework by passing posterior belief across time windows. We are the first extension of the maximum-likelihood estimation approach for social media truth discovery to the case of active learning that investigates the selection of data to label for maximum improvement in results.

8 CONCLUSIONS

In this paper, we presented an active-learning based truth estimator that discovers truth values of observations made in social networks. Our algorithm achieves significantly higher accuracy compared with existing methods. We evaluated our approach in simulation as well as on real data collected from Twitter. Our algorithms significantly outperformed existing work. A light-weight version of the original approach was developed that offered the best compromise between quality of results and computation efforts. The paper is a first example of applying an active learning approach to improve the performane of truth discovery on social media.

ACKNOWLEDGMENT

Research reported in this paper was sponsored in part by the U.S. Army Research Laboratory under Cooperative Agreements W911NF-17-2-0196 and W911NF-09-2-0053, DARPA contract W911NF-17-C-0099, and NSF grants CNS 13-29886 and CNS 16-18627. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] Md Tanvir Al Amin, Charu Aggarwal, Shuochao Yao, Tarek Abdelzaher, and Lance Kaplan. 2017. *Unveiling polarization in social networks: A matrix factorization approach.* Technical Report. IEEE.

[2] Jeffrey A Burke, Deborah Estrin, Mark Hansen, Andrew Parker, Nithya Ramanathan, Sasank Reddy, and Mani B Srivastava. 2006. Participatory sensing. *Center for Embedded Network Sensing* (2006).

[3] Hang Cui, Tarek Abdelzaher, and Lance Kaplan. 2018. Recursive Truth Estimation of Time-Varying Sensing Data from Online Open Sources. In *International Conference on Distributed Computing in Sensor Systems (DCOSS).* New York, NY.

[4] Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava. 2009. Integrating conflicting data: the role of source dependence. *Proceedings of the VLDB Endowment* 2, 1 (2009), 550–561.

[5] Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava. 2009. Truth discovery and copying detection in a dynamic world. *Proceedings of the VLDB Endowment* 2, 1 (2009), 562–573.

[6] Luyang Liu, Hongyu Li, Jian Liu, Cagdas Karatas, Yan Wang, Marco Gruteser, Yingying Chen, and Richard P Martin. 2017. Bigroad: Scaling road data acquisition for dependable self-driving. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services.* ACM, 371–384.

[7] Chuishi Meng, Houping Xiao, Lu Su, and Yun Cheng. 2016. Tackling the Redundancy and Sparsity in Crowd Sensing Applications.. In *SenSys.* 150–163.

[8] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. 2007. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement.* ACM, 29–42.

[9] Praneeth Netrapalli and Sujay Sanghavi. 2012. Learning the Graph of Epidemic Cascades. *SIGMETRICS Perform. Eval. Rev.* 40, 1 (June 2012), 211–222. https://doi.org/10.1145/2318857.2254783

[10] Praneeth Netrapalli and Sujay Sanghavi. 2012. Learning the graph of epidemic cascades. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 40. ACM, 211–222.

[11] Jeff Pasternack and Dan Roth. 2010. Knowing what to believe (when you already know something). In *Proceedings of the 23rd International Conference on Computational Linguistics.* Association for Computational Linguistics, 877–885.

[12] Jeff Pasternack and Dan Roth. 2013. Latent credibility analysis. In *Proceedings of the 22nd international conference on World Wide Web.* ACM, 1009–1020.

[13] Tauhidur Rahman, Alexander Travis Adams, Perry Schein, Aadhar Jain, David Erickson, and Tanzeem Choudhury. 2016. Nutrilyzer: A Mobile System for Characterizing Liquid Food with Photoacoustic Effect.. In *SenSys.* 123–136.

[14] Dong Wang, Md Tanvir Amin, Shen Li, Tarek Abdelzaher, Lance Kaplan, Siyu Gu, Chenji Pan, Hengchang Liu, Charu C Aggarwal, Raghu Ganti, et al. 2014. Using humans as sensors: an estimation-theoretic perspective. In *Information Processing in Sensor Networks, IPSN-14 Proceedings of the 13th International Symposium on.* IEEE, 35–46.

[15] Dong Wang, Lance Kaplan, Hieu Le, and Tarek Abdelzaher. 2012. On truth discovery in social sensing: A maximum likelihood estimation approach. In *Information Processing in Sensor Networks (IPSN), 2012 ACM/IEEE 11th International Conference on.* IEEE, 233–244.

[16] Shiguang Wang, Dong Wang, Lu Su, Lance Kaplan, and Tarek F Abdelzaher. 2014. Towards cyber-physical systems in social spaces: The data reliability challenge. In *Real-Time Systems Symposium (RTSS), 2014 IEEE.* IEEE, 74–85.

[17] Shuochao Yao, Md Tanvir Amin, Lu Su, Shaohan Hu, Shen Li, Shiguang Wang, Yiran Zhao, Tarek Abdelzaher, Lance Kaplan, Charu Aggarwal, et al. 2016. Recursive ground truth estimator for social data streams. In *Information Processing in Sensor Networks (IPSN), 2016 15th ACM/IEEE International Conference on.* IEEE, 1–12.

[18] Shuochao Yao, Md Tanvir Amin, Lu Su, Shaohan Hu, Shen Li, Shiguang Wang, Yiran Zhao, Tarek Abdelzaher, Lance Kaplan, Charu Aggarwal, and Aylin Yener. 2016. Recursive Ground Truth Estimator for Social Data Streams. In *Proceedings of the 15th International Conference on Information Processing in Sensor Networks (IPSN '16).* IEEE Press, Piscataway, NJ, USA, Article 14, 12 pages. http://dl.acm.org/citation.cfm?id=2959355.2959369

[19] Xiaoxin Yin, Jiawei Han, and S Yu Philip. 2008. Truth discovery with multiple conflicting information providers on the web. *IEEE Transactions on Knowledge and Data Engineering* 20, 6 (2008), 796–808.

[20] Xiaoxin Yin and Wenzhao Tan. 2011. Semi-supervised truth discovery. In *Proceedings of the 20th international conference on World wide web.* ACM, 217–226.

Dressing as a Whole: Outfit Compatibility Learning Based on Node-wise Graph Neural Networks

Zeyu Cui
Institute of Automation, Chinese Academy of Sciences
University of Chinese Academy of Sciences
zeyu.cui@nlpr.ia.ac.cn

Zekun Li
Institute of Information Engineering, Chinese Academy of Sciences
University of Chinese Academy of Sciences
lizekunlee@gmail.com

Shu Wu
Institute of Automation, Chinese Academy of Sciences
University of Chinese Academy of Sciences
shu.wu@nlpr.ia.ac.cn

Xiaoyu Zhang
Institute of Information Engineering, Chinese Academy of Sciences
zhangxiaoyu@iie.ac.cn

Liang Wang
Institute of Automation, Chinese Academy of Sciences
University of Chinese Academy of Sciences
wangliang@nlpr.ia.ac.cn

ABSTRACT

With the rapid development of fashion market, the customers' demands of customers for fashion recommendation are rising. In this paper, we aim to investigate a practical problem of fashion recommendation by answering the question "which item should we select to match with the given fashion items and form a compatible outfit". The key to this problem is to estimate the outfit compatibility. Previous works which focus on the compatibility of two items or represent an outfit as a sequence fail to make full use of the complex relations among items in an outfit. To remedy this, we propose to represent an outfit as a graph. In particular, we construct a *Fashion Graph*, where each node represents a category and each edge represents interaction between two categories. Accordingly, each outfit can be represented as a subgraph by putting items into their corresponding category nodes. To infer the outfit compatibility from such a graph, we propose *Node-wise Graph Neural Networks* (NGNN) which can better model node interactions and learn better node representations. In NGNN, the node interaction on each edge is different, which is determined by parameters correlated to the two connected nodes. An attention mechanism is utilized to calculate the outfit compatibility score with learned node representations. NGNN can not only be used to model outfit compatibility from visual or textual modality but also from multiple modalities. We conduct experiments on two tasks: (1) Fill-in-the-blank: suggesting an item that matches with existing components of outfit; (2) Compatibility prediction: predicting the compatibility scores of given outfits. Experimental results demonstrate the great superiority of our proposed method over others.

The first two authors Zeyu Cui and Zekun Li contribute to this work equally and are listed as joint first authors. Shu Wu and Xiaoyu Zhang are both corresponding authors.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313444

CCS CONCEPTS

• **Applied computing → Online shopping**.

KEYWORDS

Compatibility learning, graph neural networks, multi-modal

ACM Reference Format:
Zeyu Cui, Zekun Li, Shu Wu, Xiaoyu Zhang, and Liang Wang. 2019. Dressing as a Whole: Outfit Compatibility Learning Based on Node-wise Graph Neural Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313444

1 INTRODUCTION

Clothing plays an increasingly significant role in human's social life, as a proper outfit can enhance one's beauty and display the personality. However, not everyone has a strong fashion sensitivity. "How to make a suitable and beautiful outfit?" has become a daily headache for many people. Solving this problem requires modeling human notions of the compatibility. Here we aim to propose an effective method to predict the compatibility of outfits and thus help people make proper outfits.

Previous works on compatibility learning mainly focused on the compatibility of two items. These works followed the idea to map the items into a style space and estimate the distance between style vectors of items. Veit *et.al* used SiameseCNNs [1, 38] to transform items from the image space to the style space. Mcauley *et.al* [23] proposed to use Low-rank Mahalanobis Transformation to map items into the style space. Lately, some works proposed to model the compatibility of items by mapping items into several style spaces [11, 31]. Recently, there appear a few works directly measure the compatibility of outfit. Li *et.al* [17] proposed to represent an outfit as a sequence and adopt Recurrent Neural Network (RNN) [3, 6, 12, 21, 24, 25, 33] to model the compatibility. Similarly, another work [9] further represented an outfit as a bidirectional sequence with a specific order (i.e., top to bottom and then to accessories), and used Bidirectional Long Short Term Memories (Bi-LSTMs) to predict the next item in an outfit.

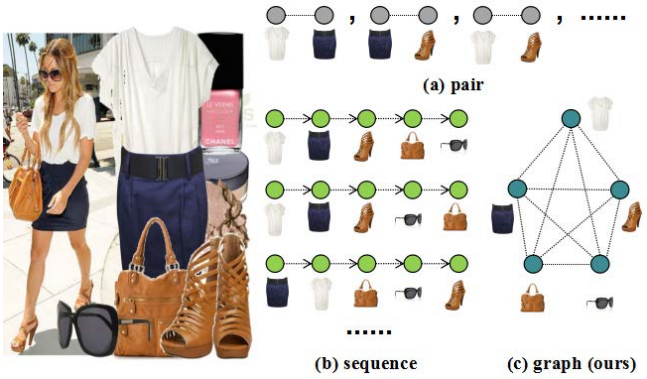


Figure 1: Different representations of an outfit. Left is the outfit. (a), (b) and (c) in the right are its pair, sequence and graph representations respectively. Circles represent the items and links represent the relations.

Solving the task of modeling outfit compatibility relies on an appropriate outfit representation. The previous works take advantage of two kinds of outfit representations: *pair representation* [11, 23, 31, 38] and *sequence representation* [9]. We show different outfit representations in Figure 1. From (a) we can see that representing the outfit as pairs separately cannot reflect the complex relations among multiple items. In term of *sequence representation* in (b), there doesn't exist a fixed order of items. More importantly, the relations among items in an outfit are not ordered since an item has relations with not only its prior or rear item in the sequence. To remedy this, we propose a new outfit representation: *graph representation*. As shown in (c), it can better reflect the dense and complex relations among multiple items in an outfit.

In particular, we construct a *Fashion Graph* based on Polyvore dataset, where each node represents a category and each edge represents the interaction between two nodes. If two categories have matching relation (i.e., they appear in the same outfit), there should be interactions between them. We have two directed edges between them in *Fashion Graph* since the interactions in two directions ought to be different. For example, one may select a pair of socks to match the shoes but not select shoes for socks in particular. A compatible outfit should have a key property: the categories of items in it are not overlapped (e.g., there won't be two pairs of shoes in an outfit). Thus, by putting each item into its corresponding node, an outfit can be represented as a subgraph of *Fashion Graph*. The task of modeling outfit compatibility can be accordingly converted to a graph inference problem.

A possible method is using Graph Neural Networks (GNN) to model the node interactions and infer compatibility from the graph. GNN is an approach to applying neural networks to graph-structured data, first proposed by Gori *et al.* [7]. It can be applied on most kinds of graphs, including directed, undirected and cyclic graphs. However, it has trouble in propagating information across a long range in a graph. To address this, Li *et al.* [18] proposed Gated Graph Neural Network (GGNN) by introducing a Gated Recurrent Units (GRU) for updating. Nevertheless, in GGNN, the nodes interact with others by communicating their state information on the edges in a fixed way, which leads to a limitation that it has difficulty in modeling flexible and complex node interactions.

To address this limitation, we propose a novel model NGNN to better model node interactions and better infer the compatibility from graph. In NGNN, the communication of state information on each edge is different. Specifically, when two nodes communicate state information on the edge, the state information will first be transformed by a transformation function determined by parameters correlated to the two nodes. In this way, NGNN can model edge-wise interactions with node-wise parameters. Since the parameters are correlated to nodes instead of numerous edges, it can greatly reduce the parameter space. During training process, NGNN only deals with the nodes in the subgraph each time, which reduces the time complexity as well. Moreover, when there are more data to train the model, we only need to update the parameters of these concerned nodes without changing the edges or other nodes, which is effective in real applications. With the learned node representations, we utilize an attention mechanism to calculate a graph-level output to serve as the outfit compatibility score.

The framework of our proposed method is shown in Figure 2. We first construct a *Fashion Graph* where an outfit can be represented as a subgraph. Then we use the proposed novel model NGNN to model node interactions and learn the node representations. Finally, we predict the compatibility score by calculating a graph-level output using an attention layer. This proposed method can model outfit compatibility from visual, textual or any single modality with a channel of input features. By using two channels of input to NGNN, it can also be used to jointly model the outfit compatibility from visual and textual modality.

We conduct experiments on two tasks: (1) Fill-in-the-blank: suggesting an item that matches with existing components of outfit; (2) Compatibility prediction: predicting the compatibility scores of given outfits. Experiment results on the Polyvore dataset demonstrate the great superiority of our method over others. The code and dataset of our work have been released[1].

Our main contributions can be summarized in threefold:

- To the best of our knowledge, this is the first attempt to represent an outfit as a graph, which can better capture the complex relations among multiple items in an outfit.
- We propose a novel model NGNN which can better model node interactions in the graph and learn better node representations. Taking advantage of NGNN, we can not only model outfit compatibility from visual or textual modality, but also from multiple modalities.
- Experimental results on a real-world dataset (Polyvore dataset) demonstrate the great superiority of our method over others.

2 RELATED WORK

2.1 Fashion Compatibility Modeling

There have been many studies focusing on modeling the compatibility of clothing recently, since it is the key to fashion recommendation. McAuley *et al.* assumed that there existed a latent style space where compatible items stayed close. They used Low-rank Mahalanobis Transformation (LMT) to map items into a latent space and

[1] https://github.com/CRIPAC-DIG/NGNN

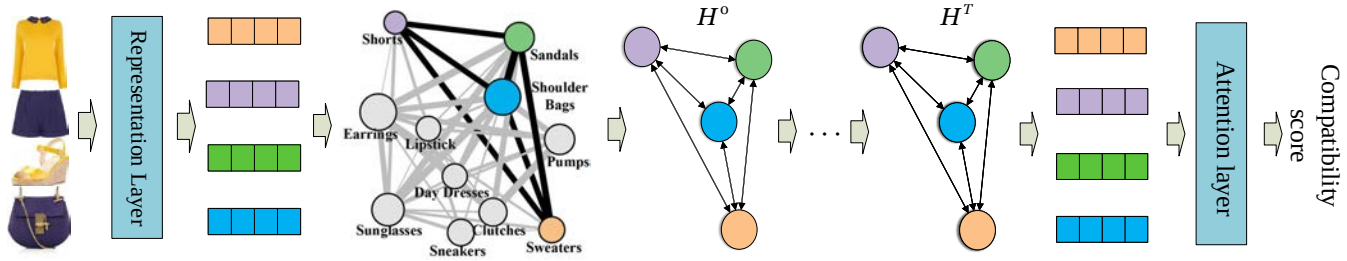


Figure 2: Overview of the proposed method. Based on the dataset, we first construct a *Fashion Graph*, where each node represents a category and each edge represents the interaction between two nodes. An outfit (e.g. sweater, short, sandals, shoulder bag) can be represented as a subgraph. We then design NGNN to model the node interactions and learn node representations. An attention layer is finally utilized to calculate the compatibility score.

measured the distance there [23]. Following that, Veit *et al.* proposed to learn the distance metric with an end-to-end SiameseCNNs [38]. Some other works mapped items into several latent spaces and jointly modeled the distances in these latent spaces, which can measure the compatibility of items in different aspects [2, 11, 31]. However, these works only focus on the compatibility of two items instead of a whole outfit.

Recently, some studies have been trying to model the compatibility of a whole outfit directly. Li *et al.* extracted multi-modal information from items and evaluated the outfit compatibility with an Recurrent Neural Network (RNN) [17]. Then they constructed an outfit recommendation by selecting the items which achieve the highest compatibility scores along with the given items. Han *et al.* represented an outfit as a sequence with a specific order. Then they used bidirectional LSTMs to predict the next item given a set of items and also the compatibility scores of outfits [9]. Modeling outfit compatibility relies on a proper representation of the outfit, which can reflect the complex relations among multiple items. In this work, we represent an outfit as a graph, which is proved to be more effective than pairs and sequence in the later experiments.

Categorical information plays a dominant role in the representation of an item. In the above conventional methods modeling items in a common visual latent space, the items of same categories tends to be close, which makes it harder to reflect the style information. For example, in the common visual latent space, the similarity between suit pants and leather shoes is much smaller than the similarity between suit pants and jeans. In Sherlock [10], the embedding matrices for transferring visual features to style features vary among different categories. Liu *et al.* proposed that items consist of two components: style and category [20]. In this work, we utilize the categorical information directly by putting items into its corresponding category nodes in the *Fashion Graph*.

2.2 Graph Neural Networks

There have been various ways to deal with graph-structured data. Some works convert graph-structured data into sequence-structured data to deal with. Inspired by word2vec [26], Perozzi *et al.* proposed an unsupervised DeepWalk algorithm to learn node embedding in graph based on random walks [27]. After that, Tang *et al.* proposed a network embedding algorithm LINE [36], followed by node2vec [8]. Some works apply neural networks on the graph-structured data directly. Duvenaud *et al.* designed a special hashing function so that Convolutional Neural Network (CNN) can be used on the graphs [5]. Kipf *et al.* proposed a semi-supervised method on graphs based on an efficient variant of CNN [14].

Gori *et al.* first proposed Graph Neural Networks (GNN) [7], another approach to applying neural networks directly to graph-structured data. GNN can be applied on most kinds of graphs, including directed, undirected and cyclic graphs. Feed-forward neural networks are applied to every node in the graph recurrently. The nodes interact with others by communicating their state information. Thus, the hidden state of nodes will update dynamically. Scarselli *et al.* utilized multi-layer perceptions (MLP) to update the hidden state of nodes [29]. However, there exists problem in propagating information across a long range in a graph. To remedy this, Li *et al.* proposed Gated Graph Neural Network (GGNN) by introducing a Gated Recurrent Units (GRU) for updating [18]. GGNN has been utilized for various tasks, such as image classification [22], situation recognition [16], recommendation [39] and script event prediction [19]. Nevertheless, it has difficulty in modeling complex node interactions since the communication of state information on the edges are in a fixed way. To address this limitation, we propose a novel model NGNN, which can better model node interactions so as to better infer outfit compatibility from the graph. In NGNN, the communication of state information on each edge is different. It can model edge-wise interactions with node-wise parameters.

3 DATASET AND FEATURES

In this section, we introduce the dataset we use and how we extract the visual and textual features of items, which are used as input of our proposed method.

3.1 Polyvore Dataset

Polyvore dataset released by [9] was collected from a popular fashion website Polyvore.com[2], where fashion stylists can share the outfits they created to the public. As shown in Figure 3, items in the outfits have rich information including clear images, category information, titles, numbers of likes, etc. We only use the images, titles and category information of items in this paper. The dataset has been utilized by several works on fashion analysis [13, 17, 32].

[2]http://www.polyvore.com/.



Category	Image	Title
Coats		Irregular lapel blue trench coat
Knee Length Skirts		Beautiful cotton lace skirt yellow
Sandals		Mcq alexander mcqueen mirrored leather sandals
Necklaces		Navy blue yellow statement bib necklace earrings set

Figure 3: Examples of outfit composition on Polyvore.com. Each item has a clear image, titles and category information.

There are totally 21889 outfits covering 380 categories in the Polyvore dataset, which was split into three non-overlapping sets, 17316 for training, 1497 for validation and 3076 for testing. Following [32], there are many categories appearing too few times in the dataset such as *Toys*, *Furniture* and *Cleaning*, which are not actually fashion categories. To ensure the training quality, we keep categories which appear more than 100 times in the dataset and 120 categories remain. To ensure the integrity of outfits, we filtered out the outfits consisting of less than 3 items. Finally, there remain 16983 outfits in the training set, 1497 in the validation set and 2697 in the test set, with totally 126054 items covering 120 categories. The maximum size of outfits is 8 and the average size is 6.2.

3.2 Features

We extract the visual features using the images of items and textual features using titles. We will model the outfit compatibility using these features of items as described in the following.

3.2.1 Visual Feature. We utilize the advanced deep convolutional neural network GoogleNet InceptionV3 model [34] to extract the visual features, which has been proved to be effective in image representation learning [4, 9, 15, 30]. In particular, we feed the images of items into the model and adopted the linear layer output as the visual feature. The visual feature of each item is a 2048-dimensional vector.

3.2.2 Textual Feature. Following [32], we first construct a vocabulary based on the words of the titles in the dataset. Due to the noise of these user-generated titles, we first filter out the words which appear in less than 5 items. Since there are many meaningless words such as 'a', 'an' and 'de', we then filter out words with less than 3 characters. Finally we obtain a vocabulary with 2757 words and therefore we represented the textual feature of each item as a 2757-dimensional Boolean vector.

4 PROPOSED METHOD

We first formulate the problem. Then we introduce how to construct the *Fashion Graph* and model outfit compatibility from the graph with NGNN. We also introduce how to jointly model outfit compatibility from multiple modalities with NGNN. Finally, we describe the training strategy and analyze our model.

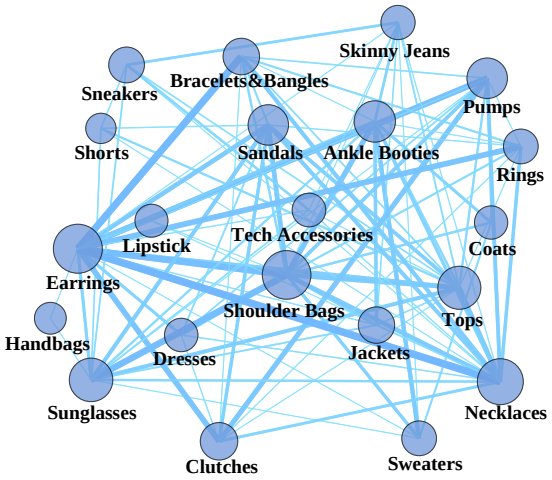


Figure 4: Illustration of the most popular categories in Polyvore dataset. Each circle denotes a category, and each link denotes the matching relation between two categories (i.e., they have appeared in the same outfit). The areas of circles and the widths of the links are proportional to the number of fashion items in the corresponding categories and the co-occurrence frequency between categories.

4.1 Problem Formulation

In this paper, we aim to model the compatibility of outfits. We have a set of outfits $\mathcal{S} = \{s_1, s_2, s_3, ...\}$ in the training set. The total item set is $\mathcal{V}$. Given an outfit s consisting of $|s|$ items (each item has an image and textual description), we aim to predict the compatibility score x_s of the outfit. The score can be used for various tasks of fashion recommendation.

We introduce some notations which will be used in the following. For each item $v_i \in \mathcal{V}$, its feature is f_i and the representation of f_i in the latent space is r_i. The category of v_i is c_i. The corresponding node of c_i in the *Fashion Graph* is n_i, while the state of node n_i in NGNN is h_i.

4.2 Graph Construction

In Figure 4, we illustrate the most popular categories in the Polyvore dataset. Each circle denotes a category, and each link denotes the matching relation between two categories (i.e., the two categories have appeared in the same outfit). The areas of circles and the widths of the links are proportional to the number of fashion items in the corresponding categories and the co-occurrence frequency between categories.

Similar to Figure 4, we construct a *Fashion Graph* $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ according to the training dataset. Each node in the graph $n_i \in \mathcal{N}$ represents a category $c_i \in C$, so that $|\mathcal{N}| = |C|$. The edge represents interaction between nodes. If two categories have matching relation in the training dataset, there are two directed edges in reverse directions between the two corresponding nodes in *Fashion Graph*, since the interactions in two directions ought to different.

For example, one may select a pair of socks to match the shoes but less likely to select a pair of shoes to match socks particularly.

According to the training dataset, we calculate the weights of directed edges as follow,

$$w(n_i, n_j) = \frac{Count_{c_i,c_j}/Count_{c_j}}{\sum_k Count_{c_i,c_k}/Count_{c_k}}, \tag{1}$$

where $w(n_i, n_j)$ denotes the weight of edge from n_i to n_j, $Count_{c_i,c_j}$ is the co-occurrence frequency of category c_i and c_j, and $Count_{c_j}$ is the occurrence frequency of category c_j.

The constructed *Fashion Graph* has 120 nodes and 12500 directed and weighted edges. By putting each item into its corresponding node, an outfit can be represented as a subgraph of *Fashion Graph*. Then we use NGNN to infer the compatibility from the subgraph as described in the next section.

4.3 Node-wise Graph Neural Network

Our proposed method NGNN consists of three steps. The first step is to learn the initial node state. The second step is to model node interactions and update node state. The third step is to calculate the compatibility score with attention mechanism.

4.3.1 Learning Initial Node State. In NGNN, each node n_i is associated with a hidden state vector h_i, which will be updated dynamically. The input of NGNN is features (either visual or textual features) of items, which can be used to initialize the state of their corresponding nodes. For each item $v_i \in s$, we first map its feature f_i to a latent style space with the size of d. Considering the difference among the categories, we have a linear mapping matrix W_h^i for each category c_i. Thus, we can obtain the representation of item v_i in the latent space as follows,

$$r_i = W_h^i f_i. \tag{2}$$

We then use the representation to initialize the hidden state h_i^0 of its category corresponding node n_i as,

$$h_i^0 = tanh(r_i). \tag{3}$$

Thus, the size of node state is also d.

4.3.2 Modeling Node Interactions. Each time the nodes communicate (i.e., propagate their state information to others and receive others' information), which is called a propagation step, the state of nodes will be updated.

In the traditional GGNN, at propagation step t, the nodes will receive sum of neighbors' state as,

$$a_i^t = \sum_{n_j \to n_i \in \mathcal{E}} \mathbf{A}[n_j, n_i] W_p h_j^{t-1} + b_p, \tag{4}$$

where W_p and b_p are weights and biases of a shared linear transformation on all the edges. $\mathbf{A}$ is the adjacency matrix:

$$\mathbf{A}[n_i, n_j] = \begin{cases} w(n_i, n_j), \text{ if } n_i \to n_j \in \mathcal{E}, \\ 0, \text{ others }. \end{cases} \tag{5}$$

The traditional GGNN use the same W_p and b_p to model interactions between different nodes. However, it fails to model the complex and flexible interactions between them. Since the node interaction on each edge ought to be different, we aim to model edge-wise interaction. There is a straight-forward way to achieve edge-wise node interactions, that is to let each edge has its own transformation weight and bias W_p and b_p. However, it will cost a great parameter space with numerous edges in a graph and cannot be applied on large scale graphs.

To remedy the problem above, we have an output matrix W_{out}^i and an input matrix W_{in}^i for each node n_i in NGNN. The transformation function of edge $n_i \to n_j$ from node n_i to node n_j could be written as

$$W_p^{n_i \to n_j} = W_{out}^i W_{in}^j. \tag{6}$$

Thus, the Equation 4 could be rewritten as,

$$a_i^t = \sum_{n_j \to n_i \in \mathcal{E}} \mathbf{A}[n_j, n_i] W_{out}^j W_{in}^i h_j^{t-1} + b_p. \tag{7}$$

After receiving state information a_i^t, the state of node n_i will be updated as following,

$$z_i^t = \sigma(W_z a_i^t + U_z h_i^{t-1} + b_z), \tag{8}$$

$$r_i^t = \sigma(W_r a_i^t + U_r h_i^{t-1} + b_r), \tag{9}$$

$$\tilde{h}_i^t = tanh(W_h a_i^t + U_h(r_i^t \odot h_i^{t-1}) + b_h), \tag{10}$$

$$h_i^t = \tilde{h}_i^t \odot z_i^t + h_i^{t-1} \odot (1 - z_i^t), \tag{11}$$

where, $W_z, W_r, W_h, b_z, b_r, b_h$ are weights and biases of the updating function, which are similar to Gated Recurrent Unit (GRU) [18]. z_i^t and r_i^t are update gate vector and reset gate vector, respectively.

After T propagation steps, we can obtain the final state of nodes, which are also the final node representations. Then we can generate a graph-level output to serve as the compatibility score x_s of outfit s. We adopt the attention mechanism described in next section to compute the graph-level output.

4.3.3 Compatibility Calculating with Attention Mechanism. We contend two viewpoints on how fashion item influence the outfit compatibility here. The first viewpoint is that items have different influence on an outfit. For example, an improper top and an improper necklace have different influence to the outfit influence. The second is that the same item plays different roles in different outfits. For example, a coat maybe suitable for an outfit and promote the compatibility, while it will decrease the compatibility together with summer clothes. Thus, we aim to design an attention mechanism to model the influence of items on the outfit compatibility. Since a node receives state information from other nodes, their node representations are aware of the global information (i.e., other nodes' state). Similar with [18], we utilize self-attention [35, 37] to calculate the graph-level output and also the outfit compatibility score as following,

$$x_s = \sum_{i=1}^{|s|} \sigma(\theta(h_i^t)) \cdot \alpha(\delta(h_i^t)). \tag{12}$$

$\theta(\cdot)$ and $\delta(\cdot)$ are two perception networks to output a real value. $\theta(\cdot)$ is used to model the weights of items (i.e., importance of items' influence on the outfit compatibility), and $\delta(\cdot)$ is used to model the compatibility score of items (i.e., how the item goes with the other items in the outfit). $\alpha(\cdot)$ and $\sigma(\cdot)$ are leaky relu and sigmoid activate functions $\alpha(x) = max(0.01x, x), \sigma(x) = 1/(1 + e^{-x})$.

4.4 NGNN with Multi-modal Input

As described above, with the input of visual features or textual features of items, NGNN can be used to model the outfit compatibility from visual or textual modality. We here introduce how to jointly model the outfit compatibility from both visual and textual modality with NGNN.

We use two channels of NGNN, the input of a channel is visual features, and the input of another channel is textual features. Specifically, for each outfit s, we input the visual features and textual features to the two channels of NGNN respectively and obtain a visual compatibility score x_s^{vis} and a textual compatibility score x_s^{txt}. We calculate the final outfit compatibility score x_s as follow,

$$x_s = \beta x_s^{vis} + (1-\beta) x_s^{txt} \tag{13}$$

where β is a non-negative trade-off parameter.

As described in Section 4.3.1, for each item v_i, its visual feature f_i^{vis} and textual feature f_i^{txt} are mapped into a shared latent space respectively. We denote their representations in latent space as r_i^{vis} and r_i^{txt}. Considering the coherent relation between the visual and textual information of items, we use a regularization to ensure the consistency between the visual and textual feature of the same item in the shared latent space as follow,

$$\mathcal{L}_{con} = \sum_{i=1}^{|s|} \left\| r_i^{vis} - r_i^{txt} \right\|^2 \tag{14}$$

4.5 Training Strategy

We denote outfits in the training dataset as positive outfits. For each outfit $s = \{v_1, v_2, ..., v_p, ...\} \in \mathcal{S}$, we randomly select an item v_p and replace it with a random item $v_q \in \mathcal{V}$ to form a negative outfit $s^- = \{v_1, v_2, ..., v_q, ...\}$. Thus, we can obtain a dataset $\mathcal{P}_{train}$ consisting of these pairs (s, s^-). Similar to Bayesian Personalized Ranking [28], we assume that the positive outfit has higher compatibility than the negative one. Therefore, we have the following objective function,

$$\mathcal{L}_{bpr} = \sum_{(s,s^-)\in\mathcal{P}_{train}} -\ln \sigma(x_s - x_{s^-}). \tag{15}$$

The objective function of NGNN is:

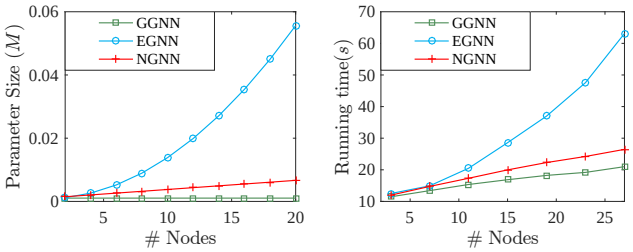
$$\mathcal{L}_1 = \mathcal{L}_{bpr} + \frac{\lambda}{2} \|\Theta\|^2, \tag{16}$$

where Θ refers to the set of parameters and λ is the hyper-parameter of L2 regularizer, which is designed to avoid overfitting.

When it comes to NGNN with multi-modal input, we have an additional loss function $\mathcal{L}_{con}$ as described in Section 4.4. Thus, we have the following objective function,

$$\mathcal{L}_2 = \mathcal{L}_{bpr} + \mathcal{L}_{con} + \frac{\lambda}{2} \|\Theta\|^2. \tag{17}$$

We only train NGNN on the subgraph of outfit each time. That is to say, only the nodes in the subgraph are trained and only their parameters are updated each time, which greatly reduces time complexity. When we have more data to train the model, we only need to update the parameters of concerned nodes without changing the parameters of edges or other nodes, which is effective in real applications.



(a) Parameter Size (b) Running Time

Figure 5: The parameter sizes and running time of GGNN, EGNN and NGNN on graphs with the number of nodes varying from 2 to 30. EGNN is a simple variant of GGNN, in which every edge has a different dtransformation matrix for propagating state information.

4.6 Model Analysis

In this section, we analyze the parameter space and time complexity of our proposed method NGNN.

4.6.1 Parameter Space. The parameter space of our proposed method consists of two parts: the parameters correlated to nodes and the perception networks in attention mechanism. For each node n_i, we have a linear mapping matrix W_h^i to map the input features into the latent space, an input matrix W_{in}^i and an output matrix W_{out}^i to propagate state information. Totally we have $3|\mathcal{N}|$ matrices, which are proportional to the number of nodes $|\mathcal{N}|$. In addition, we have two matrices of perception networks in the attention mechanism. Overall, we have $3|\mathcal{N}|$ matrices. Therefore, we say that the parameters of NGNN is node-wise.

4.6.2 Time Complexity. For each input outfit s, the time complexity of training process in NGNN consists of three parts. The first part is calculating the transformed state information nodes receive, which is $O(|s|)$. The second part is updating the state of the concerned nodes, which is also $O(|s|)$. The last part is updating the parameters of concerned nodes. Since the number of parameters correlated to the concerned nodes is $3|s|$ as described above, its complexity is also $O(|s|)$. According, the total complexity is $O(|s|)$.

We compare NGNN with GGNN [18] and Edge-wise GNN (EGNN). EGNN is a straight-forward way to achieve edge-wise interactions. In EGNN, each edge has an unique transformation matrix for propagating state information. We train the three models on a toy dataset consisting of outfits with the number of items varying from 2 to 30. In another word, the three models are trained on graphs with the number of nodes varying from 2 to 30. The parameter size and running time are shown in Figure 5. As can be seen, the parameter size and running time of EGNN increased quadratically while those of NGNN and GGNN increased linearly.

In summary, NGNN can better model edge-wise interactions and learn better node representations with appropriate parameter space and time complexity. When there are more training data, we can only update the parameters of concerned nodes offline and upload them online without changing the edges or other nodes, which is effective in real applications.

5 EXPERIMENTS

As the main contribution of this work is to model outfit compatibility, we aim to answer the following research questions via experiments.

- **RQ1** How does our proposed NGNN perform compared with other pair or sequence representation models?
- **RQ2** How do different components in NGNN effect the performance?
- **RQ3** How do different modalities influence the inference of compatibility?
- **RQ4** How do the hyper-parameters d and T effect the performance?

Next, we first describe the two tasks we conduct experiments on and the experimental settings. We then report the results by answering the above research questions in turn. Finally, we analyze the parameter space and time complexity of our proposed model.

5.1 Task Description

We conduct experiments on two tasks: fill-in-the-blank fashion recommendation and compatibility prediction on the polyvore dataset described in Section 3.1.

5.1.1 Fill-in-the-blank Fashion Recommendation. The task of fill-in-the-blank (FITB) in fashion recommendation is first introduced in [9]. In this task, Given a set of fashion items and a blank, we aim to find the most compatible item from the candidates set to fill in the blank as shown in Figure 6. This is a practical scenario in people's daily life. For example, a person wants to select clothes to match the pants, hat and shoes. Following [9], we create a fill-in-the-blank dataset based on the Polyvore test dataset. For each outfit in the test dataset, we randomly select an item from it and replace the item with a blank. By randomly select 3 negative items from $\mathcal{I}$ we can form a 4-length candidates set along with the positive one, assuming the negative items are less compatible than the positive one. In our work, we respectively put the four candidate items into the blank to form an outfit and calculate their compatibility scores with our model. The one which achieves the highest outfit compatibility score is chosen as the answer. The performance is evaluated by the accuracy of choosing the right answer from the four candidates (FITB accuracy), which is 25% by randomly selecting.

5.1.2 Compatibility Prediction. This task is to predict the compatibility score for any given outfit. To evaluate our model, we first construct a dataset $\mathcal{P}_{test}$ using the Polyvore test dataset. Specifically, for each positive outfit s in the Polyvore test dataset, we create a negative outfit s^- by randomly selecting $|s|$ items from $\mathcal{V}$. Thus, we can obtain a dataset $\mathcal{P}_{test}$ consisting of evaluation pairs (s, s^-). We then adopt the widely used metric AUC (Area Under the ROC curve) to evaluate the performance, which is defined as,

$$AUC = \frac{1}{|\mathcal{P}_{test}|} \sum_{(s,s^-)\in\mathcal{P}_{test}} \delta\left(x_s > x_{s^-}\right), \tag{18}$$

where $\delta(a)$ is an indicate function that returns one if the argument a is *true* and zero otherwise.

Table 1: The performance of models evaluated by Fill-in-the-blank accuracy and compatibility prediction AUC.

Method	Accuracy (FITB)	AUC (Compatibility)
Random	24.07%	0.5104
SiameseCNNs	48.09%	0.7087
LMT	50.91%	0.6782
Bi-LSTM	67.01%	0.8427
GGNN (visual)	72.78%	0.9516
GGNN (textual)	73.63%	0.9591
NGNN (visual)	77.01%	0.9600
NGNN (textual)	77.78%	0.9716
NGNN (multi-modal)	**78.13%**	**0.9722**

5.2 Experimental Settings

For optimization, we adopt the RMSProp optimizer. The optimal hyper-parameters (learning rate, batch size, trade-off parameter β, parameter for L2 regularization λ, hidden size d and the propagation steps T) are determined by the grid search strategy. We experimentally find that the model achieves optimal performance with the learning rate as 0.001, batch size as 16, β as 0.2, λ as 0.001, d as 12 and T as 3. We early stop the training process when the loss stabilizes, usually at 12 epoch. All the experiments were conducted over a sever equipped with 3 NVIDIA Titan X GPUs.

5.3 Model Comparison (RQ1)

To evaluate our proposed method, we compare it with the following state-of-the-art baselines. Note that SiameseCNNs and LMT focus on the compatibility between two items. In the task of outfit recommendation, we sum up the compatibility scores of all the pairs in the outfit as the score of the outfit.

Random: Randomly assign the compatibility scores of outfits.

SiameseCNNs: It's a pair-wise compatibility learning method [38], which minimizes the distance of image vectors between compatible items and maximums the distance between incompatible items. This method only focuses on the visual information.

LMT: Proposed by [23], it models compatibility between items in a style space via a single low-rank Mahalanobis embedding matrix. This method also focuses on visual modality.

Bi-LSTM: It's proposed by [9]. By viewing an outfit as a sequence, it uses the bidirectional LSTMs to predict the next item, taking multi-modal data as input, which can also predict the compatibility scores of outfits. This method focus on multi-modal information.

GGNN (visual / textual): We use GGNN to model the outfit compatibility with input of visual or textual features of items.

NGNN (visual / textual): We use our proposed NGNN to model the outfit compatibility with input of visual or textual features of items.

NGNN (multi-modal): We jointly models the outfit compatibility with NGNN from the visual and textual modalities.

We conduct comparison experiments on the tasks of fill-in-the-blank fashion recommendation and compatibility prediction. The results are shown in Table 1. Next we analyze the results of FITB accuracy and compatibility AUC respectively.



Figure 6: Comparison of our model and Bi-LSTM on several examples in fill-in-the-blank task. Green fonts represent the right chosen and red fonts represent wrong chosen.



Figure 7: Comparison of our model and Bi-LSTM on several example of task compatibility prediction. Green fonts represent the right evaluation and red fonts represent wrong.

5.3.1 FITB Accuracy. The comparison of FITB accuracy is shown in the middle column of Table 1. We can obtain the following observations from the table: (1) SiameseCNNs and LMT achieve worse performance compared with the other methods, which demonstrates that the pair representation ignores the integrity of outfit. (2) GGNN outperforms Bi-LSTM, suggesting that our proposed graph representation can better reflect the complex relations among items in an outfit than sequence representation. (3) NGNN achieves a better performance than GGNN, which proves that our proposed NGNN can better infer compatibility information from the graph by better modeling node interactions. (4) Using textual features achieves better performance than visual features for both GGNN and NGNN, which suggests that the textual features can better represent fashion items than visual features in that the titles accurately summarize the key features of items. (5) NGNN with Multi-modal input achieves best performance, proving that the visual and textual information are good supplements to each other. In addition, our proposed method can jointly model the outfit compatibility from multiple modalities. We randomly select several example outfits in the FITB dataset to test on our model and Bi-LSTM. The results are shown in Figure 6. From example 1, we can see that Bi-LSTM and our model can both correctly select the item of complementary category as the outfit lacks of a pair of shoes. In example 2, our model chooses the right item while Bi-LSTM choose the skirt wrongly, which conflicts with the pants. This may be because that the position of pants is far away from the blank so that it has little influence on it. In example 3, our model selects the most compatible bottom jeans with the whole set from the four bottoms while Bi-LSTM selects the white short skirt which may be most compatible with the white shirt, but not such compatible with the other items. The wrong chosen of Bi-LSTM may be because that the nearby white shirt has strong influence on the blank. From these example, we can draw a conclusion: the items have strong influence on nearby items in the sequence while have less influence on items faraway. This makes the sequence represented method relies on a proper sequence order. Compared with the sequence-represented model Bi-LSTM, our graph-represented model can better model the complex relations among items in an outfit.

5.3.2 Compatibility AUC. The performance comparison of models evaluated by compatibility AUC is shown in the third column of Table 1. Similar with the comparison of performance evaluated by FITB accuracy, NGNN achieves best performance, GGNN is second best, and then is the sequence represented method Bi-LSTM, the pair represented methods (SiameseCNNs, LMT) are worst. Using textual features can achieve better performance than visual features. NGNN with multi-modal data achieves the best performance. These observations support our analysis in Section 5.3.1. In addition, it can be seen that GGNN and NGNN all achieve great performance and the gap is small, which proves the superiority of graph representation on compatibility prediction. GGNN is already good enough for compatibility prediction.

We randomly select several example outfits and show the compatibility scores predicted of Bi-LSTM and our model in Figure 7. In example 1, our model gives the outfit a high score correctly while Bi-LSTM gives it a relatively low score, which may be because that the sequence representation method fails to capture the high compatibility between the first top and the fifth handbag. While Bi-LSTM and our model both give the outfit a high score correctly

in example 2. In example 3, the fifth shorts conflict with the first skirts. Our model gives it a low score correctly while Bi-LSTM gives it a relatively high score, which may be because Bi-LSTM fails to capture the conflict of the two bottoms across a long distance. Examples 2 and 3 suggest that Bi-LSTM relies on a proper order of sequence, which is flexible and hard to learn. The influence between two items is inversely proportional to their distance in the sequence. Our model can better reflect the relation among items compared with Bi-LSTM. In example 4, our model gives it a low score to the positive outfit. A plausible explanation is that there is little matching of coat, sweater and shorts in training dataset.

Table 2: The performance of our proposed method with different components.

Method	FITB Accuracy	Compatibility AUC
NGNN(-W-A)	75.17%	0.9541
NGNN(-W)	75.71%	0.9559
NGNN(-A)	76.12%	0.9578
NGNN	**77.01%**	**0.9600**

5.4 Component Comparison (RQ2)

To investigate the effect of different components on the performance of our proposed method, we compare our method with different components.

NGNN (-W): NGNN without weighted adjacency matrix. Specifically, we use 0-1 adjacency matrix instead of the weighted one as described in Section 4.2.

NGNN (-A): NGNN without attention mechanism. We only use a perception network to model the compatibility scores of nodes and sum them up them.

NGNN (-W-A): NGNN with neither the attention mechanism nor weighted adjacency matrix.

The performances are shown in Table 2. If not specifically stated, we use visual features as input of NGNN. We observe that NGNN outperforms all the ablative methods, which proves the necessity of the two components in our model. NGNN (-W) achieves worse performance than NGNN, suggesting that the weighted adjacency matrix we calculated is a supplement to the information of graph. NGNN outperforms NGNN (-A), which confirms the necessity of our attention mechanism.

To intuitively illustrate how the attention mechanism works, we show the attention weights of several example outfits in Figure 8. The depths of color are proportional to the attention weight of items. The higher attention weight item has, the deeper the color of bar below it is, the stronger influence the item has to the outfit compatibility. As can be seen in example 1, the shirt, short skirt and necklace have distinct and compatible styles and they have strong influence to the outfit compatibility. The black bag and black shoes have little influence to outfit compatibility while they have ordinary styles. Similarly, the sweater, shoes and hat in example 2 are all of distinct rustic style and they have strong influence to the outfit compatibility. The jeans and shoes in example 3 can match well with most styles while they have little influence to the outfit compatibility. The white outwear and jeans which can match with many styles in example 4 also have little influence to outfit compatibility. From these examples, we can find that more distinct style items have, more influential they are to the outfit compatibility. On the contrary, more ordinary items are (i.e., can match with many styles), less influential they are. This is reasonable in common sense.

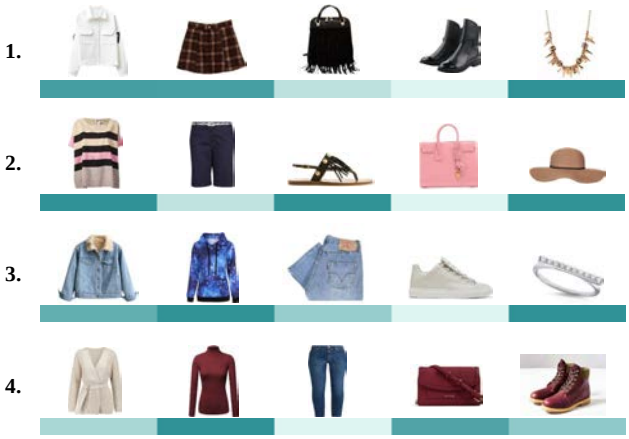


Figure 8: Illustration of attention mechanism. The depth of color are proportional to the attention weights of items.

5.5 Modality Comparison (RQ3)

To investigate the influence of different modalities on the performance, we compare the NGNNs (multi-modal) with different modality combinations. As described in Equation 13, β is the weight of visual compatibility score while that of textual compatibility score is $1-\beta$. Note that when $\beta = 0$ the model is actually NGNN (visual), when $\beta = 1$ it's NGNN (textual). In Figure 10, we show the performance of NGNN (multi-modal) with β varying from 0 to 1. It can be seen when $\beta = 0.2$, our model achieves the best performance, which suggests that the visual and textual information are good supplement to each other and both contribute s to the inference of compatibility. We can also see that using textual features can achieve better performance than visual features. A plausible explanation is that the textual information is more concise to capture the key features of items.

To intuitively see the difference of NGNN with visual, textual and multi-modal input, we show the comparison of them on several testing examples in Figure 9. In the example 1, Visual NGNN chose the false answer due to its similar color and material with the jacket, which the visual information can reflect. However the similarity between the textual information (titles) of the true answer and the dress, jacket directly reveals their high compatibility. In example 2, it's hard to choose the answer with textual input since the titles of the two answers are similar, while with visual input NGNN can distinct the high compatibility between the styles of the true answer and the outfit from visual information. In summary, the visual and textual information are great supplements to each other. By integrating them we can achieve better performance.

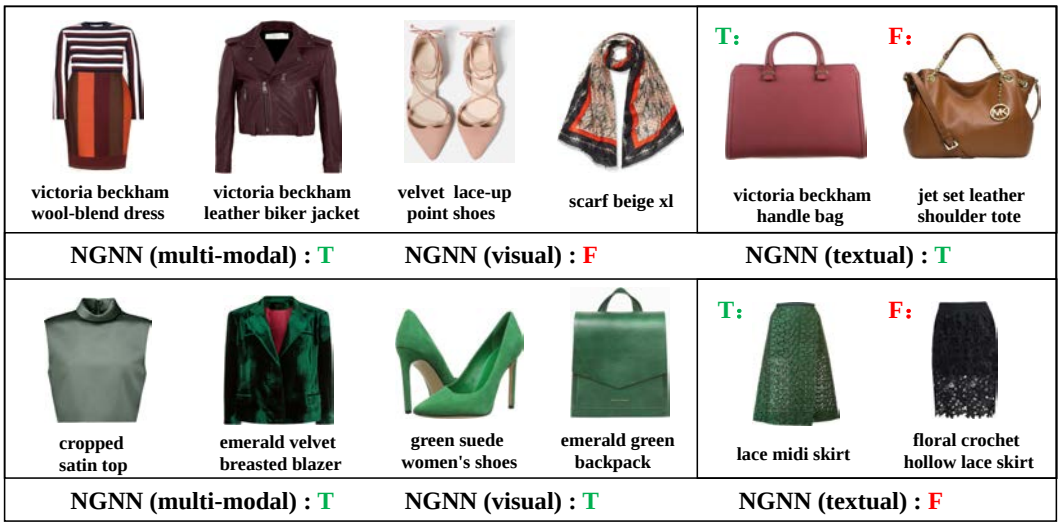


Figure 9: Comparison of NGNN(visual), NGNN(textual) and NGNN(multi-modal). For each outfit in the left, there is a true answer (T) and a false answer (F). Green fonts reprsent true chosen while red fonts represent false chosen.

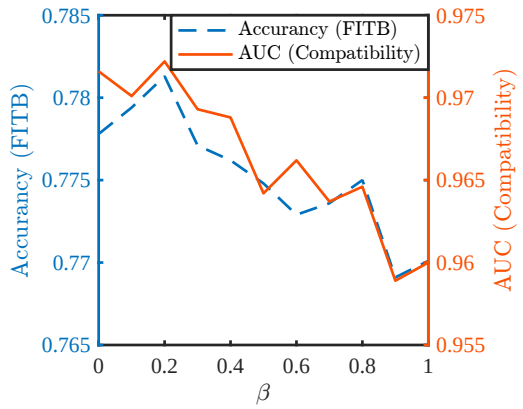
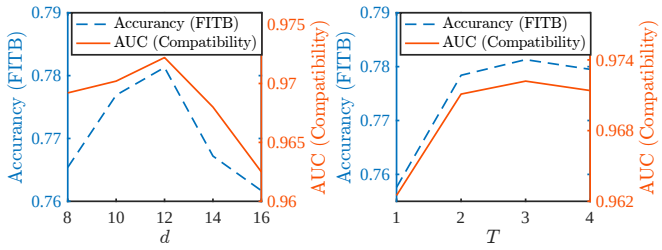


Figure 10: FITB accuracy and compatibility AUC of our model with different β.



(a) State dimensionality (b) Propagation times

Figure 11: AUC and FITB accuracy performance with different dimensionality d (left) and propagation times T (right).

5.6 Hyper-parameter Discussion (RQ4)

In this section, we examine how the dimensionality of hidden vectors d and the propagation time d affect the performance. Comparison experiments are conducted on NGNN (multi-modal). Figure 11 shows the performance of our model with varying d and d. From the figure in the left, we can observe that the FITB accuracy and compatibility AUC reach the highest point when d is around 12. When d is large than 12, the performance drops sharply. This indicates that there is no need to represent the node state with too much parameters. From the figure in the right, it can be seen that the performance saturates when T is 3 and the nodes in NGNN have interacted with others sufficiently, suggesting that it's important to model the multi-hop propagation so that the nodes are better aware of other nodes' information. Moreover, when T is high, the superfluous information propagation will confuse the nodes.

6 CONCLUSION

In this paper, we first propose to model the outfit compatibility by representing an outfit as a graph since the graph structure can better capture the complex relations among items in outfits. To infer the compatibility information from the graph, we further propose a novel model NGNN, which can better model flexible node interactions and learn better node representations. Our proposed method can model outfit compatibility not only from visual or textual modality respectively, but also from multiple modalities jointly. Experimental results on a real-world dataset (Polyvore dataset) prove the great superiority of our method over existing works. In the future, we aim to model personalized compatibility since people have different notions of fashion compatibility.

ACKNOWLEDGMENTS

This work is jointly supported by National Natural Science Foundation of China (61772528, 61871378) and National Key Research and Development Program (2016YFB1001000).

REFERENCES

[1] Sean Bell and Kavita Bala. 2015. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 98.

[2] Long Chen and Yuhang He. 2018. Dress Fashionably: Learn Fashion Collocation With Deep Mixed-Category Metric Learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. 2103–2110.

[3] Qiang Cui, Shu Wu, Qiang Liu, Wen Zhong, and Liang Wang. 2018. MV-RNN: A Multi-View Recurrent Neural Network for Sequential Recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[4] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*. 647–655.

[5] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*. 2224–2232.

[6] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 1999. Learning to forget: Continual prediction with LSTM. (1999).

[7] Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, Vol. 2. IEEE, 729–734.

[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[9] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S Davis. 2017. Learning fashion compatibility with bidirectional lstms. In *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 1078–1086.

[10] Ruining He, Chunbin Lin, Jianguo Wang, and Julian McAuley. 2016. Sherlock: sparse hierarchical embeddings for visually-aware one-class collaborative filtering. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. AAAI Press, 3740–3746.

[11] Ruining He, Charles Packer, and Julian McAuley. 2016. Learning compatibility across categories for heterogeneous item recommendation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 937–942.

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[13] Yang Hu, Xi Yi, and Larry S Davis. 2015. Collaborative fashion recommendation: A functional tensor factorization approach. In *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 129–138.

[14] Thomas N Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint arXiv:1609.02907* (2016).

[15] Hanbit Lee, Jinseok Seol, and Sang-goo Lee. 2017. Style2Vec: Representation Learning for Fashion Items from Style Sets. *arXiv preprint arXiv:1708.04014* (2017).

[16] Ruiyu Li, Makarand Tapaswi, Renjie Liao, Jiaya Jia, Raquel Urtasun, and Sanja Fidler. 2017. Situation recognition with graph neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*. 4173–4182.

[17] Yuncheng Li, Liangliang Cao, Jiang Zhu, and Jiebo Luo. 2017. Mining fashion outfit composition using an end-to-end deep learning approach on set data. *IEEE Transactions on Multimedia* 19, 8 (2017), 1946–1955.

[18] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2015. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493* (2015).

[19] Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Constructing Narrative Event Evolutionary Graph for Script Event Prediction. *arXiv preprint arXiv:1805.05081* (2018).

[20] Qiang Liu, Shu Wu, and Liang Wang. 2017. Deepstyle: Learning user preferences for visual recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 841–844.

[21] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: a recurrent model with spatial and temporal contexts. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI Press, 194–200.

[22] Kenneth Marino, Ruslan Salakhutdinov, and Abhinav Gupta. 2017. The More You Know: Using Knowledge Graphs for Image Classification. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 20–28.

[23] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 43–52.

[24] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

[25] Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 5528–5531.

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[27] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[29] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks* 20, 1 (2009), 61–80.

[30] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014. CNN features off-the-shelf: an astounding baseline for recognition. In *CVPR workshops*.

[31] Yong-Siang Shih, Kai-Yueh Chang, Hsuan-Tien Lin, and Min Sun. 2018. Compatibility family learning for item recommendation and generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*. 2403–2410.

[32] Xuemeng Song, Fuli Feng, Jinhuan Liu, Zekun Li, Liqiang Nie, and Jun Ma. 2017. Neurostylist: Neural compatibility modeling for clothing matching. In *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 753–761.

[33] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*.

[34] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2818–2826.

[35] Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2017. Deep Semantic Role Labeling with Self-Attention. *arXiv preprint arXiv:1712.01586* (2017).

[36] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 1067–1077.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[38] Andreas Veit, Balazs Kovacs, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie. 2015. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *Proceedings of the IEEE International Conference on Computer Vision*. 4642–4650.

[39] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Xing Xie, and Tieniu Tan. 2018. Session-based Recommendation with Graph Neural Networks. In *Thirty-Third AAAI Conference on Artificial Intelligence*.

Are All Successful Communities Alike? Characterizing and Predicting the Success of Online Communities

Tiago Cunha
University of Michigan
tiolivei@umich.edu

David Jurgens
University of Michigan
jurgens@umich.edu

Chenhao Tan
University of Colorado Boulder
chenhao.tan@colorado.edu

Daniel M. Romero
University of Michigan
drom@umich.edu

ABSTRACT

The proliferation of online communities has created exciting opportunities to study the mechanisms that explain group success. While a growing body of research investigates community success through a single measure — typically, the number of members — we argue that there are multiple ways of measuring success. Here, we present a systematic study to understand the relations between these success definitions and test how well they can be predicted based on community properties and behaviors from the earliest period of a community's lifetime. We identify four success measures that are desirable for most communities: (i) growth in the number of members; (ii) retention of members; (iii) long term survival of the community; and (iv) volume of activities within the community. Surprisingly, we find that our measures do not exhibit very high correlations, suggesting that they capture different types of success. Additionally, we find that different success measures are predicted by different attributes of online communities, suggesting that success can be achieved through different behaviors. Our work sheds light on the basic understanding on what success represents in online communities and what predicts it. Our results suggest that success is multi-faceted and cannot be measured nor predicted by a single measurement. This insight has practical implications for the creation of new online communities and the design of platforms that facilitate such communities.

CCS CONCEPTS

• **Human-centered computing** → *User studies.*

KEYWORDS

Online Communities; Success; Group Dynamics; Reddit

ACM Reference Format:
Tiago Cunha, David Jurgens, Chenhao Tan, and Daniel M. Romero. 2019. Are All Successful Communities Alike? Characterizing and Predicting the Success of Online Communities . In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, Article 4, 11 pages. https://doi.org/10.1145/3308558.3313689

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313689

1 INTRODUCTION

Understanding the complex ways in which communities are successful over time is essential for building and maintaining vibrant online communities [36, 39]. User-created online communities in social media platforms enable people with shared points of view and interest to connect with each other and are one of the main ways in which the Web has shaped how people interact and find each other. Although the low barrier for creating online communities has led to a large number of communities that are constantly involving and continually emerging, not all of them are successful. Fortunately, massive datasets from these online communities have allowed researchers to study the dynamics of the creation and lifecycles of online communities at a large scale [2, 8, 23, 34, 55, 59]. In particular, a problem of interest is to predict the eventual success of a new community from a set of community attributes that can be measured early in its lifetime. That is, *can we tell early on whether a community will be successful?*

An implicit yet important question is concerned with the definition of community success. In fact, most existing research takes a very narrow view of what defines success in online communities. Existing research has typically considered a single measure of community success and this measure is often determined by the number of users who eventually participate in the community [2, 8, 34, 55]. While this is a reasonable measure of success for many communities, it is not necessarily appropriate for all. Consider, for example, a community created with the goal of maintaining social relationships among the graduates from a small college on a given year. The members of such community would probably not define success by the size of their group, relative to other communities. A better measure of success in this case would be whether active members continue to stay active over time (i.e., retention). In contrast, a community created with the goal of raising awareness about an issue such as global warming would likely measure its success by how many people joined the cause.

We argue that, given the rich and complex nature of online communities, considering a single measure of success has limited our understanding on what makes communities successful. Our work aims to fill this gap by exploring a variety of success measures of newly created communities on Reddit. We consider four classes of success measures that should be generally desirable by most communities: (i) growth in the number of members; (ii) retention of members; (iii) long term survival of the community; and (iv) volume of activities within the community. We find that, while these measures are positively correlated, the correlations are not very

high, suggesting that each of the measures captures a fundamentally different aspect of a community's success. By focusing only on a community's growth in the number of members, we miss other dimensions that are also and sometimes even uniquely desirable.

Given the observed differences between our success measures, we then explore the features of newly created communities that are predictive of future success. Since we expect that different measures of success be predicted by different features, we consider a variety of features to capture different aspects of the community early in its lifetime. In our framework, we observe a community from the time of its creation until the time when it has attracted a fixed number of members. Then, using attributes of the community that can be measured up until that time, we predict the success of the community based on each of our success measures.

Specifically, we consider six different classes of features to predict success based on existing research and theory on online community success, group development, and organization theory: (i) the volume and speed of initial activities; (ii) the distribution of activities over users and time; (iii) the composition of its early members; (iv) linguistic style features of early content; (v) features of the communication network among early members; and (iv) the communities that early members belong to before they join the new community. We test the predictive performance of each class of features separately as well as all features combined. We find that no single class of features outperforms the combination of all regardless of the measure of success we use, suggesting that each class captures a meaningful dimension of the community's success. Furthermore, we find that the single most predictive class highly depends on the measure of success. For example, while linguistic style features are the best predictors of a community's survival, the distribution of activities over time best predicts retention.

While we find that different measures of success are predicted by different classes of features, we also identify community behaviors that predict success across different measures. A particularly robust example is the inequality in the distribution of posts and comments per user. We find that successful communities tend to distribute their activities in a highly skewed fashion, where a small number of users contribute a large fraction of the activity in the community. Our interpretation of this pattern is that for a community to be successful, it needs to have a small number of committed members that maintain the community active early in its lifetime until the community reaches a critical mass. These observations resonate with the importance of levels of activity and commitment of an online community's founders for its ultimate success [38].

Understanding and predicting the success of online communities early in their lifetime can provide hypotheses for experimental studies to identify causal relations and help community creators manage their communities in ways that are known to predict success. Our findings suggest that online communities can be successful in a variety of ways and are predicted by different early behaviors. Thus, depending on the goals of a community, its members should focus on different aspects of their behavior in order to maximize their chances of success. Our results can help us begin to understand success in online communities as a multi-faceted concept and identify the behaviors that drive each of its possible facets.

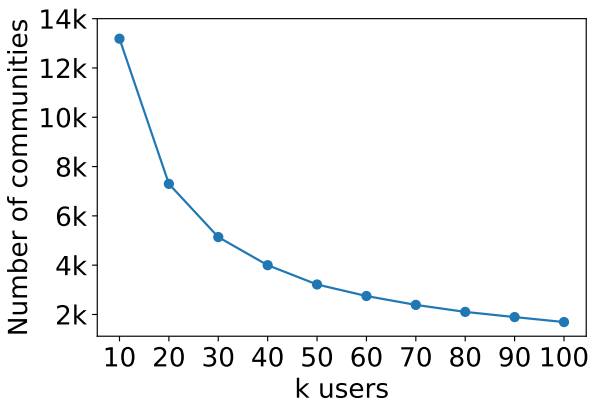


Figure 1: Number of communities that attract k users in at most 3 months, with k varying from 10 to 100.

2 REDDIT COMMUNITIES

We begin with a brief description of our testbed, Reddit. Reddit is a popular social news website and forum, which allows users to self-organize into user-created communities by areas of interest called subreddits. In Reddit, most communities are public and users can join as many public communities as they desire. Reddit also allows users to submit content, such as textual posts or URLs to other contexts, comment on posts, and upvote and downvote content. We consider a user a member of a community if they post or comment in that community.

We obtained all posts and comments from subreddits created in 2014 [4, 5]. Since our goal is to investigate the relationship between early community behavior and eventual success, we focus on communities that attracted at least k users within 3 months from the time when they were created so that we have sufficient data to measure the community's early behavior. We vary the value of k from 10 to 100 users. It follows that communities with less than 10 users within this time frame are not used in our analysis. Most subreddits exhibit very low activity after creation. Of all the communities created in 2014, only 5% attract 10 users within 3 months of their creation. Figure 1 shows the total number of communities created in 2014 that attracted k users in the first 3 months. As expected, the number of communities drops with k. However, even for relatively large values, such as $k = 100$, we will have more than 1,000 communities that pass the threshold.

3 CHARACTERIZING COMMUNITY SUCCESS

Although community success is an essential problem in understanding group dynamics [36, 39], success is an overloaded concept and has been defined in many different ways. For instance, a battery of studies have focused on growth [34, 55], which implies that communities that attract a lot of users are successful. These studies tend to focus on factors that attract new users such as social influence and tightly connect with research on diffusion [3, 8, 11, 48–50]. Another heavily studied metrics is retention, also known as churn prediction, which examines the likelihood that a user leaves the community after initial participation [18, 20, 33, 56]. A closely related metric to

retention is survival, which examines how long meaningful activity in a community is sustained [34, 59]. In practice, Web companies also use daily active users as a metric of success [37, 47].

An important open question is how these success metrics relate to each other and whether they capture the same underlying notion of "success", simply known as different names, or they capture inherently different types of success. In this section, we formally define multiple metrics of success and provide a large-scale characterization of these success metrics and their relations on Reddit communities. We believe that all these definitions can be appropriate yet *different* because success is not uni-dimensional and can differ due to the varying nature of communities. A successful online discussion group on political news likely present different desirable characteristics when compared to a group formed by editors of Wikipedia articles. For a political news group, an important goal is to sustain daily activity in order for the community to stay updated with daily news, while a group of Wikipedia editors may focus on producing high-quality articles, which requires retaining trained users for a long period of time rather than simply growing the number of users.

3.1 Defining Community Success

We consider four classes of success measures that should be generally desirable by most communities: (i) growth in the number of members; (ii) retention of members; (iii) long term survival of the community; and (iv) volume of activities within the community. In addition, submitting posts and commenting on posts or other comments are two different ways of engaging with a community and they may be valued differently across subreddits [1]. For example, KotakuInAction is dedicated to discussing controversy centered on issues of sexism and progressivism in video game culture and a single post can generate long discussions through thousands of comments. Other sudreddits are more focused on producing content through posts such as subreddits dedicated to breaking news. We thus further distinguish growth and activity in posts and comments.

To define the success measures, we consider a fixed period after k users joined the new community and use month as the unit for measuring community characteristics. We define T_i^{posts} and T_i^{com} as the number of posts and comments in the community in month i, respectively. We let U_i^{all}, U_i^{posts}, U_i^{com} be all active users, users who posted, and users who commented in the community in month i, respectively. Finally, we let U^k be the initial k users who participated in the community and $U_{i\to i+1}$ be the subset of users that posted or commented in the community in month i and posted or commented again in the community in month $i+1$.

We then formalize our success measures in the following:

- **Growth** refers to the number of new users that joined the community in the one year window after the community reached k users. We consider two types of growth, growth in number of new users that commented (commenters) in the following year, $G^{\text{com}} = |\bigcup_{i=1}^{12} U_i^{\text{com}}|$, and growth in the number of users that posted in the following year, $G^{\text{posters}} = |\bigcup_{i=1}^{12} U_i^{\text{posts}}|$.
- **User retention** refers to the average monthly retention rate in the first year after reaching k, $R = \frac{1}{12}\sum_{i=1}^{12} \frac{|U_{i\to i+1}|}{|U_i^{all}|}$.
- **Long term survival of a community** is measured by computing the fraction of activity in the final part of our dataset for each community. That is, instead of looking at the following year, we investigate if the community "died", or stopped being active after two years. We measure the percentage of activities in the last 3 months of 24 months time window after the community size reaches k, $\frac{\sum_{i=22}^{24}(T_i^{\text{posts}}+T_i^{\text{com}})}{\sum_{i=1}^{24}(T_i^{\text{posts}}+T_i^{\text{com}})}$. Our intuition is that if the volume of activity during these final months is very small relative to the overall activity, then the community is not surviving in the long term.
- **Volume of activities** is divided in two types — the average number of posts in the first year after reaching k users, $\frac{1}{12}\sum_{i=1}^{12} T_i^{\text{posts}}$, and the average number of comments in the first year after reaching k users, $\frac{1}{12}\sum_{i=1}^{12} T_i^{\text{com}}$.

3.2 Correlation between Success Metrics

The premise for this paper is that online communities can be successful in different ways. Thus, our first task is to test this premise by exploring the relationship between the different success measures we identified. A high correlation between all of our success measures would indicate that, in reality, there is a single type of success that achieves all the desirable qualities we described.

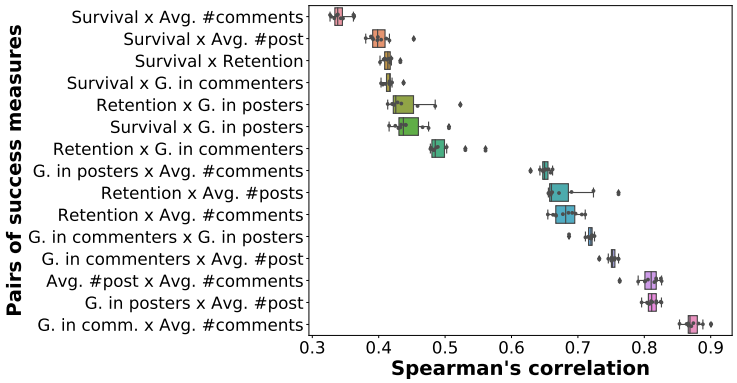
We begin by computing all pairwise Spearman's correlations among the set of success measures,[2] which are shown in figure 2a in ascending order. Since we can compute the correlation between success measures for each value of k, the figure shows a box plot for each pair of success measures for the distribution across k. We observe that, while all pairs of success measures have a positive correlation, many pairs have a relatively small correlations — especially considering that all our success measures depend on users posting content on these communities.

We also present in Figure 2b a heatmap with the average Spearman's correlation between the success measures. We find that survival has among the lowest correlation when paired with another success measure — with all such pairs exhibiting a correlation of at most 0.5. This suggests that growth, retention, and high volume of activities do not necessarily imply survival. Similarly, retention exhibits correlations of less than 0.7 with other success measures, except for average #comments that presented a correlation of 0.75. Correlations are highest among measures of growth and volume of activity, which is not surprising considering that a community with many members will necessarily have a high number of posts and comments since membership is established by activity. The opposite does not necessarily hold. A community could have a small number of very active members.

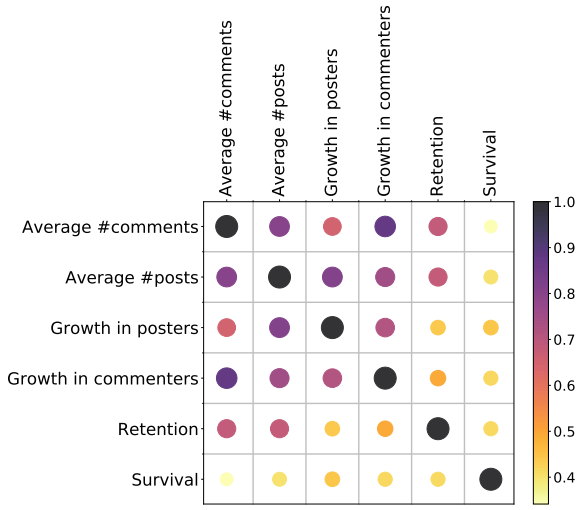
These results suggest that indeed, there is significant diversity in the ways that communities achieve success. While some communities can grow and attract a large number of members, others can remain small but succeed by having a set of committed users who always return to the community or by exhibiting long term survival. They also suggest that measuring success with a single

[1] In fact, some prior studies only focus on posting behavior for this reason [55, 56].

[2] We obtain very similar results when we apply Kendall rank correlation [35].



(a) . Each box plot represents the distribution of average Spearman's correlation coefficient between two success measures (the number of samples in a box is 10, corresponding to $k = 10, 20, \ldots, 100$). Although all the pairwise correlations between success measures are positive, there is great variance with regard to how correlated they are, ranging from 0.34 (between survival and average #comments) to 0.86 (between growth in commenters and average #comments).



(b) Pairwise Spearman correlation between the success measures. Measures related to volume of activities and growth present highest correlations. While survival and retention present the lowest correlations.

Figure 2: Spearman's correlation between success measures. We chose Spearman's correlation because it is non-parametric, we also experimented with Kendall's Tau correlation and the results were similar.

metric can overlook important ways in which communities achieve success.

A case study between growth and retention (Figure 3). To further understand the differences between success measures, we show the scatter plot of user retention and growth in commenters for all communities that reach $k = 100$. In this scatter plot, successful communities can assume various combinations of values for both dimensions. Visually, the growth in commenters can take a wide range of values for communities with zero user retention. A closer examination of extreme points in this scatter plot reveals interesting examples. Consider the community */r/millionairemakers*. In this community, users organize themselves in a "lottery", where a user is randomly selected each month and everyone is encouraged to send this user $1. The winner typically receives between $1,000 and $5,000. This community exhibits a large number of commenters because users enter the lottery by leaving a comment. The low user retention might be explained by the fact that winning the lottery is highly unlikely, which may discourage users from participating multiple times. On the other hand, the community StoryBattles — a subreddit where users can post a picture and the rest of community is asked to write a story about it in the comment section, has very high retention but low growth in commenters. This behavior may be explained by the niche nature of the community, which targets at people interested in story writing. Most Reddit users are probably not interested in story writing, but those who are, return and contribute to this community repeatedly over time.

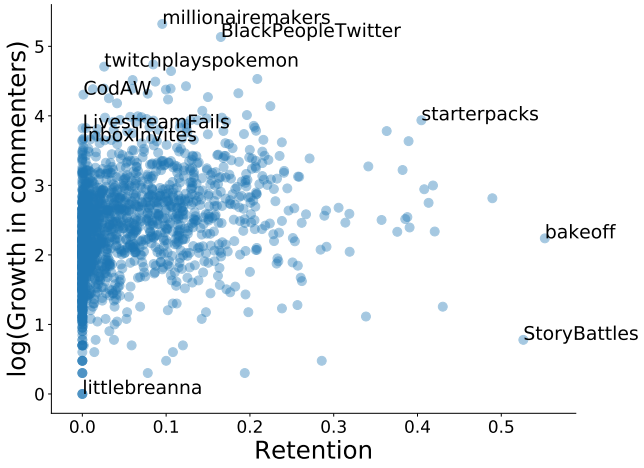


Figure 3: A case study to show the scatter plot between growth in commenters and retention for $k = 100$.

4 PREDICTIVE FEATURES OF SUCCESS

Given the relatively low correlation between success measures, it remains an open question what characteristics of communities predict future community success. Inspired by existing research and theory in online communities, group dynamics, and organization theory, we defined a set of features to be used later in our success prediction tasks (Section 5). We focus on features that generalize

across different types of communities. We divide these features into six categories which capture different dynamics of the communities. All of our predictive features are based on the actions taken by users between the time the community was created and the time its k^{th} user arrived.

Volume and speed of activities. The first set of features capture both the speed of and volume of activities early during the early stages of the formation of the community, which have been shown to predict future success [40]. It includes, the number of posters, the number of commenters, the date the community was created, the number of posts, the median number of replies received per posts, the median number of comments and posts per user, the number of days that the community took to attract k members. It also includes the speed at which comments and posts arrive — average number of days between posts and comments), which constitute strong baselines used in previous studies [10, 34].

Distribution of activities. Another important set of features we explore is related to how the content produced in the community is spread among users and over time (i.e. is a small number of users responsible for producing a large fraction of the activities in the community? Is the majority of the content concentrated in a certain point in time in a bursty fashion?). We capture this spread through the Gini coefficient [22], which is a measure of statistical dispersion commonly used to measure inequality. A Gini coefficient (Equation 1) of 0 indicates perfect equality where all members produced the same amount of content, while 1 indicates perfect inequality where a single individual generated all posts and comments. We compute the Gini coefficient of the number of posts and comments per users and the Gini coefficient of days between posts and comments. A high Gini coefficient in the number of posts or comments per user is indicative of a small number of community members who are committed to the community enough to produce a large fraction of its content. Such a small set of committed users could ensure the long term success of the community through committed leadership or suggest its demise due to the lack of participation by other community members.

$$Gini = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} |x_i - x_j|}{2n \sum_{i=1}^{n} |x_i|} \tag{1}$$

User composition. The third set of features accounts for the users' activities *prior to joining the community*. It includes the median and standard deviation of users' score ($upvotes - downvotes$) received in posts and comments, the median and standard deviation of the number of users' prior activities and the median number of days users have been active on Reddit. Except for the number of days on the site, we limit this set of features to one month prior to joining the new community. We also measure the fraction of new users who do not have any previous community membership, which includes both new users on Reddit and Reddit users who did not post or comment in the one month before posting or commenting in the new community. As older, more experienced, and more successful users (with respect of upvotes and downvotes) tend to have more experience participating in communities, our hypothesis is that communities that attract more experienced users can benefit as those users were exposed to other communities norms, thus they might produce higher quality content and consequently help the community to achieve its goals. Previous studies have shown that the experience and expertise levels of founders is a good predictor of the success of online communities and offline firms [21, 38].

Linguistic style. The next set of features capture how the language in the content created in the communities can help understand the desirable characteristics of a community. Previous studies have revealed important stylistic changes in user-written language as users develop a sense of belonging to a community [13, 19, 31, 56] and how the sentiment present in the community's feedback affects the likelihood that users will remain engaged [16]. Language can also provide an important signal of users satisfaction, which is a key factor for successful communities, since satisfied users are more likely to contribute to the community and stay engaged [41]. First, we compute features that capture the length of the content created in the new community: median post, titles, and comments length and the size of the vocabulary used in the community, i.e. the number of unique words, after pre-processing the content produced by the first k users, divided by the number of posts and comments. Next, motivated by studies of socialization and engagement in online communities [19, 31, 43], we measure the distribution of categories from the psychological lexicon LIWC [1, 45] [3], which measures the various emotional, cognitive, and structural components present in text. Lastly, inspired by research on corporate success that has found that personality traits are predictive of employees engagement, which boosts the organizational productivity [32, 53, 60], we also include the Big Five personality traits [4]: Extraversion, Conscientiousness, Neuroticism, Agreeableness and Openness to Experience [28]. These five domains have been shown to be good at predicting behavioral patterns, such as well-being and mental health, job performance and marital relations [26]. We compute all measures separately for posts and comments.

Social networks. We also include in our analysis a set of features that represent the structure of communication and information exchange among the users in a community. We construct a communication network among early members and extract a variety of features that describe its structure. Network theory has received considerable attention when studying community success. We represent the social interaction among users as an undirected graph $G(V, E)$, where V is the set of vertices and E is the set of edges between a pair of vertices. Each vertex $v \in V$ represent a user in the community, and an undirected edge $e(i, j) \in E$ exists between user v_i and user v_j if user v_i has replied to user v_j or v_j has replied to v_i in a thread. Our choice for undirected edges is based on the assumption that when a post or a comment receives a reply, both users involved are exposed to each other's content. A user i must first make a post and a user j must read it in order to reply. Once j replies, we assume that i sees the reply. Thus, we consider that i and j interacted with each other.

We then compute the following network measures: transitivity, average clustering coefficient, density, fraction of users in the largest component and fraction singletons (i.e. fraction of isolated nodes in the graph) [6]. Density, fraction of users in the largest component, and fraction of singletons measure the extent to which the network is well-connected rather than fragmented into many

[3] http://www.liwc.net

[4] To compute the Big five personality traits we used the IBM Watson Personality Insights API (https://www.ibm.com/watson/services/personality-insights/).

disconnected pieces. In a fragmented network, users are not engaging with each other enough to form a large structural community, which could lead to polarization and structural holes [7]. Transitivity and average clustering coefficient measure the extent to which users have a tendency for sharing connections (or forming triangles). Social networks with high clustering are known to facilitate trust and social capital [14], which is important for community building. Several studies have investigated the relationship between network structure and how the groups change over time [2, 8, 30, 34, 46, 51, 55]. They show that social features are good predictors of communities desirable properties, with special attention given to groups' growth, often growth is treated as a process similar to the diffusion of innovations, where joining a new group is analogous to adopting an innovation.

Finally, we also measure the fraction of posts and comments that received at least one reply as it was shown that community feedback has a positive effect over users likelihood to return to the community [12, 16, 17].

Parents communities. The last set of features aims to reveal the emergence of communities through the relationship of the communities that early members were participating before joining the new community. Following Tan's work [55], we employ features derived from the genealogy graph of parents communities. In the genealogy graph, we consider an edge between two parent communities if they share at least two early members. Then we compute density, transitivity, and number of parents communities. We also include the maximum, minimum and standard deviation of the size of parent communities measured by the log of the number of members one month prior the creation of the new community). Additionally, we include the Gini coefficient of the number of users from the parent communities. Lastly we measure the median distance between the language model of the focal community and its parents communities (i.e., the median language distance (cross-entropy) between the new community and the parents communities). We consider large language distance as a sign of diversity as Uzzi et al. [57] has shown that atypical combination is related to scientific impact. Note that communities in Tan [55] are only based on posts, while we use both posts and comments.

5 PREDICTION RESULTS

We now test how well our features described in Section 4 predict the different measures of success described in Section 3.1.

5.1 Experimental Setup

The prediction for each success measure is framed as a binary task to predict whether that measure for a community will exceed the median value of that measure for all communities considered. This prediction setup naturally leads to a balanced classification task for all success measures. It also allows us to ask how the predictability of communities desirable properties varies over the range from small to large number of members.

We train separate logistic regression classifiers with *L2* regularization for each feature category as well as a single model with the combination of all features. For each classification task, we randomly split the data for training (80%) and testing (20%). The feature values are standardized to make regression coefficients comparable across features from different scales. Models are then evaluated using AUC; a random baseline achieves an AUC score of 0.5. We grid search the best regularization hyperparameter using 10-fold cross-validation on the training set. We do not report statistical significance for regression coefficients due to their opaque interpretation when using the biased estimates [27].

Models are trained on the Reddit communities that were created in 2014 and reached k users within three months, where the number of users is measured by the total number of unique users who posted or commented on the community. To capture differences in behavioral information at different stages of a community's early life, we vary k in 10-user increments from 10 to 100. Note that fewer communities reach 100 users than 10 users, so the total number of communities that we consider decreases with k (see Figure 1). We compute the median value of each success measure for each k separately to determine the label of each community (positive if its success measure exceeds the median).

5.2 Can Success Be Predicted?

Our experiments show that success can indeed be predicted: the best median AUC score for each task is at least 0.72 when predicting with the combination of all features. In the best case, the median AUC score is 0.84 when predicting users retention. The prediction performance results for each model are shown in Figure 4.

5.2.1 Community Growth. Although growth in posters and commenters are intuitively related, models varied surprisingly in their accuracy at predicting each version of growth. In particular, user features and parent communities perform substantially worse in predicting the eventual growth in users who post (Figure 4b) than growth in users who comment (Figure 4a). This may be related to the fact that the majority of early members based on our definition are commenters. For growth in commenters, the model trained on all available features have a median performance of 0.75, yet models trained on a single set of features perform substantially worse. Among the single-feature category, the distribution of activities is the strongest single set of features with a median model performance of 0.63. The performance gap between using all features and using distribution of activities is statistically significant ($p < 0.001$), suggesting no single model captures all the aspects involved in community growth in commenters. For growth in posters, volume and speed of activities is the strongest single feature set, which echoes previous results that predict growth only considering information about posts [10, 34, 55]. However, models in previous works excluded other community features, which when combined with volume and speed, substantially enhances our ability to predict both types of growth.

5.2.2 Users Retention. Results for the task of predicting retention are shown in Figure 4c. Our first observation is that this task appears to be easier to predict than future growth. Indeed, the set of all features combined presents the best performance with a median AUC of 0.84, which is larger than any other success measure. When looking for single set of features, there is a clear best model, with distribution of activities features performing stronger than the rest of the models, displaying an AUC of 0.82. The worst performing

model is the users features, which suggests that the users' background is not important for predicting if a user will become "loyal" to the community and return to it.

5.2.3 Survival. Survival is the most difficult type of success measure to predict from our features. The prediction results are shown in Figure 4d. In this task, our model with all features achieves an AUC of 0.72. The single best model is content features with an AUC of 0.67, followed by the users composition features with an AUC of 0.63. This observation further confirms the importance of the content generated in the community to keep users committed to the community. It also suggests that early members' previous experience and characteristics are important to community survival, which resonates with Kraut and Fiore [38] that found that founders social capital is a good predictor of online communities survival.

5.2.4 Activity level. Next, we evaluate the predictive power of our feature sets to predict the future activity level. The results for average #comments are shown in Figure 4e and results for average #posts are shown in Figure 4f. The results for the two tasks are similar, with the model with all features performing slightly better for average #comments. This can be explained by the performance of the social features model, which presented a better result (0.67 vs 0.58, $p < 0.001$) for the task of predicting average #comments. The model with all features outperformed the single features models for both tasks, but this difference was only statistically significant for the task of predicting average #comments (0.80 vs 0.76, $p < 0.001$.). The worst results for single features models were presented by the model with user features, which suggests that the users' history before joining a new community does not play an important role in the frequency of users to post/comment in the future.

5.3 How Early is Success Predictable?

Each of the box plots in Figure 4 contains the resulting AUC for each value of $k = 10, \ldots, 100$. One may expect the performance of the model to increase with k, as a higher k represents data from more users to predict the eventual success of the community. Surprisingly, most of the box plots display a small range of variation — indicating that the performance of the model is in fact not very sensitive to the number of early users we base our predictions on. In particular, the AUC for $k = 10$ is not very different from that of $k = 100$. Table 1 shows the standard deviation of the AUCs for all features combined when applied to all success measures. The small deviation values confirm our intuition that indeed the features are predictive since the community's infancy. This result is important for communities maintainers and moderators as they can diagnose their communities in their initial stage and identify "at risk" communities.

5.4 Which Features Predict Success?

In this subsection we examine the most predictive features of success when we combine all our features into a single model. Recall that we have a separate model for each value of k. For each k, we rank the features of the resulting model by the magnitude of its coefficient. We then use the mean reciprocal rank [15] of the features over the individual ranks for $k = 10, \ldots 100$ to generate a single ranking of the features. Figure 5 shows the top 10 features and their mean coefficients.

Success measure	Std. AUC
Growth in commenters	0.030
Growth in posters	0.020
Retention	0.006
Survival	0.022
Average #comments	0.023
Average #posts	0.024

Table 1: Standard deviation of results for the model with all features combined for all success measures.

The most important set of features is the distribution of activities. Its features appeared among the most predictive for all measures of success. It shows that except for average #posts, higher Gini coefficient of posts and comments per users is positively associated with all success measures, which means that if the initial activities in the community are concentrated in a small number of users, it increases the overall likelihood that the community will be successful. In early stages, when communities do not have many members or much content, they must rely on committed users that are more likely to engage and produce high-quality content.

We also find that the time the community takes to reach k users is predictive of all success measures, except growth in posters. However, the direction of the effect of this feature is not consistent across success measures. While a small number of days to reach k users (i.e. fast initial membership growth) increases the likelihood that a community will grow and survive, it decreases the ability of the communities to retain users. These findings resonate with research on the founding of organizations, which shows that fast initial growth predicts longer organization survival [38]. However, our results show that this long term survival can come at the cost of lower retention. Additionally, we note that average #posts and #comments are also affected differently by this feature. Though a small number of days to attract k new members predicts an increased average number of comments, but is related to an decreased future average #posts. Thus, it appears that communities that grow slowly in membership during early stages exhibit higher levels of interactions among users through comments at the cost of lower production of original content through posts.

Regarding social features, increased average clustering and fraction of users in the largest connected component are negatively associated with growth, while transitivity is positively associated with user retention, which is in accordance with previous results in groups growth [2, 30, 34], where users in very clustered groups tend to persist in the community. However, this property also decreases overall growth and the likelihood to survive longer. In addition, when members are too tightly connected to the largest component, as they are in groups with high transitivity, groups might become too closed, restricting the possibility of gaining new information and attracting new members [34]. Lastly, increased density predicts increased level of activities, which suggests that social interaction predicts future activity level.

In term of linguistic features, we note an interesting negative association between the use of first-person plural pronouns "*we*" and a community's long term survival and growth. This finding

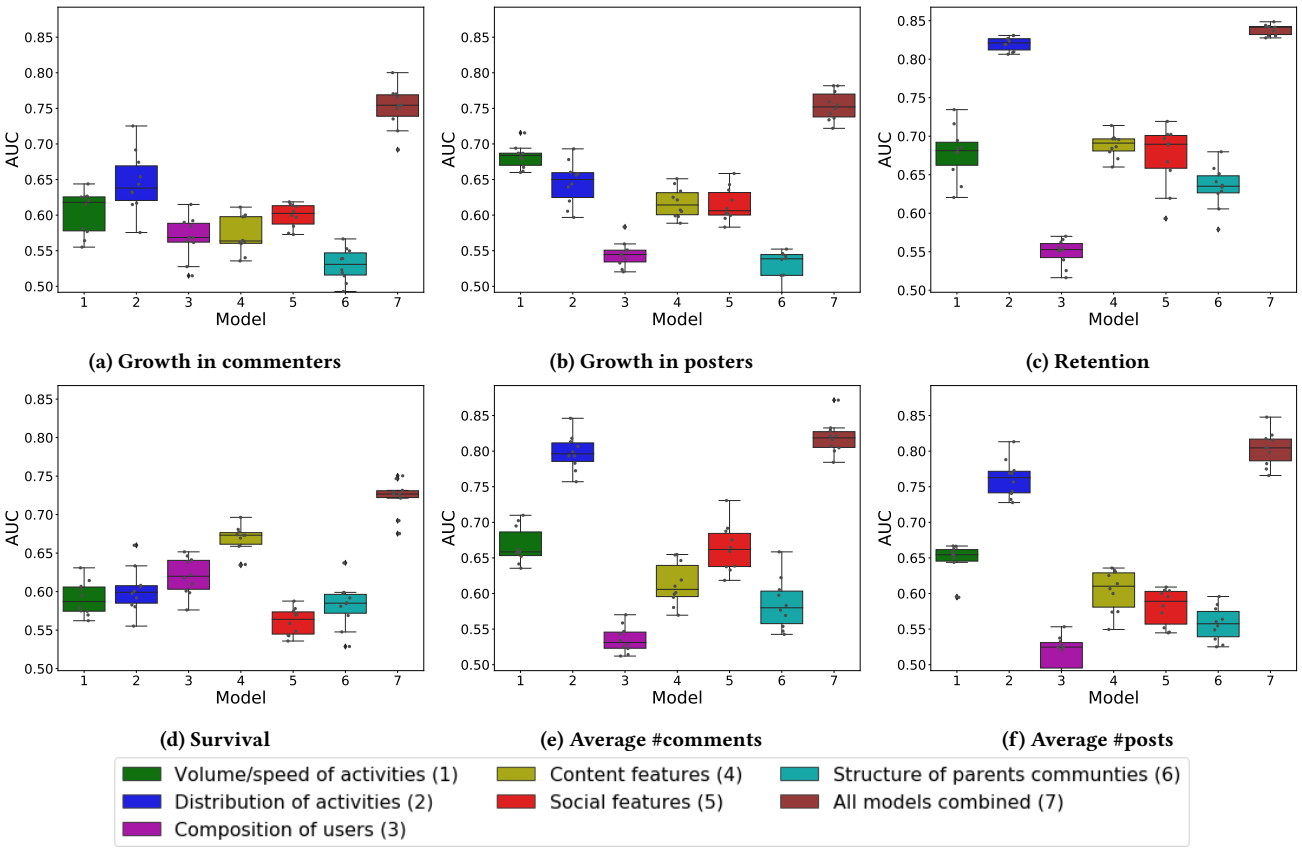


Figure 4: Results for the predictive tasks of success measures. Box plots graphically depict the results for k varying from 10 to 100 users. They show that our features sets are predictive of communities' success measures, the model of all features combined presents at least 0.72 AUC for all task.

indicates that users in large communities that survive longer are less likely to assume a collective identity, which echoes Palla et al. [44] that suggests a connection between fluid dynamics in membership and a group's long-term survival. In contrast, findings from sociolinguistics show that loyal members (in terms of user preference and commitment) of communities presented a higher use of first-person plural pronouns "*we*" and language that signals collective identity [13, 31, 52]. The diverging effect of "*we*" confirms the low correlation between retention and survival, and further demonstrates the multi-faceted nature of success.

6 DISCUSSIONS

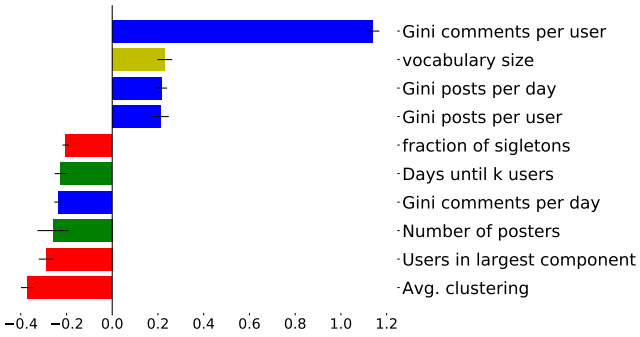
6.1 Limitations

Not controlling for effect of a community's topic. It is possible that communities related to certain topics have different likelihoods to be successful. Although we do not control for the topic of the communities or how its chosen topic fits within the larger ecosystem, we aim to identify a set of features that generalize across topics and types of communities such that our recommendations are general enough and can be applied to any type of new community.

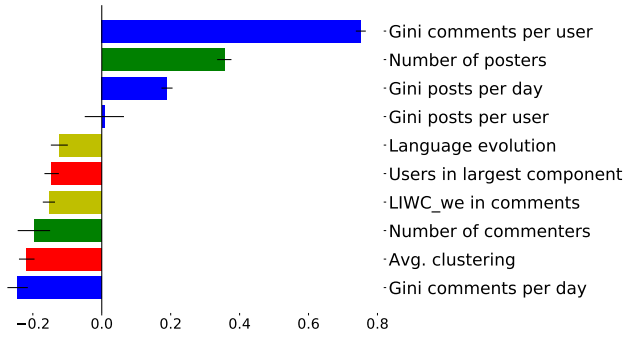
Experiments focus on subreddits created in 2014. Our entire study is based on 2014 Reddit data, which is late (8 years) into the Reddit ecosystem, when many communities have already been established. While analyzing success of communities in the earlier stage of Reddit could reveal interesting patterns, the analysis would be more challenging as Reddit was changing its platform and evolving rapidly during its early years. Additionally, the large number of potential missing (i.e. deleted users) data from Reddit's earliest years may also introduce biases to the analysis [25]. However, a replication of our analysis using a much larger time period or studying Reddit alternatives during their formative growth periods [42] could determine if the predictors of success change over time.

Causality. In our models, we have interpretable features, but we do not have a causal model to suggest that the presence of a feature *causes* the community to succeed. Applying experimental or other causal inference techniques such as propensity score matching would be a possible line of future research to establish a causal link between community behaviors and success.

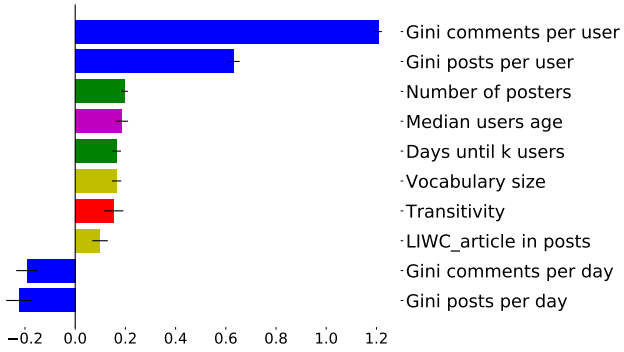
Our analysis only uses posts and comments. There are other aspects of Reddit communities that may be informative such as stated community norms, moderator behaviors such as banning users or removing posts. However, currently this type of data is



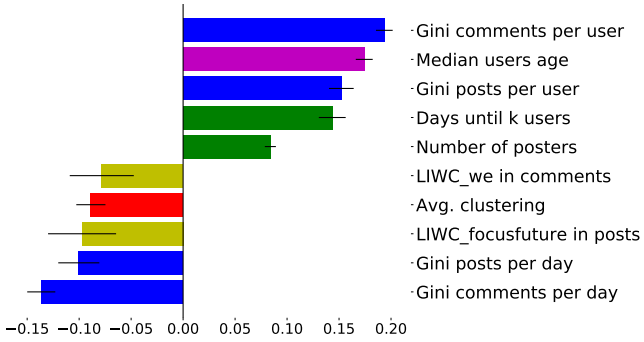
(a) Top 10 features for growth in commenters.



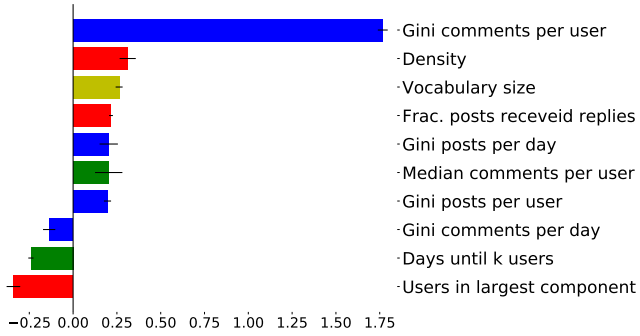
(b) Top 10 features for growth in posters.



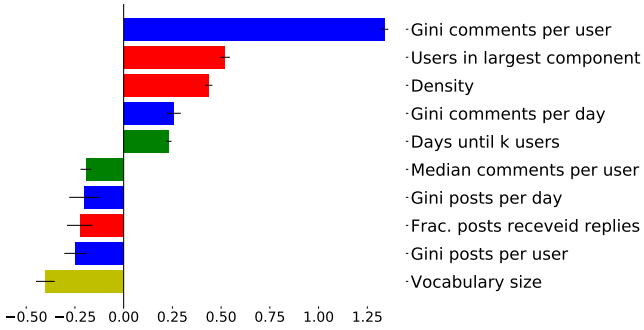
(c) Top 10 features for users retention.



(d) Top features for community survival.



(e) Top 10 features for average #comments.



(f) Top 10 features for average #posts.

Figure 5: Top 10 most predictive features for the model with all features combined

either not available or it's difficult to obtain a historical version, which makes their use impractical in our analysis [9].

6.2 Design implications

New online communities are formed every day. Given that very early behaviors of such communities is strongly predictive of multiple types of success, we discuss design implications for new community organizers to increase their likelihood of success. We reiterate the important caveat that our study is purely observational and does not establish causal relations between behaviors and success.

High Gini in posts and comments per user always predict success, regardless of success measure, which results from having a few users that perform most of the activities. This result suggests that having a small group of highly committed groups of participants early on is key to future success. Community organizers need not necessarily worry when not all participants engage frequently, but need to ensure that a sufficient number of members engage at producing content in order to establish a critical mass, which is responsible for helping the community to become self-sustaining and create further growth [29, 48].

Taking longer to attract k members is negatively associated to growth in commenters, survival and average #comments, but predicts retention and average #posts. We interpret these results as reflecting two kinds of community goals. For those communities

that want to attract many users, stay active longer and have its activities primarily concentrated on discussions, attracting users quickly is important. However, for retaining those users, what matters is getting the right type of users who are likely to remain active, which may take longer initially.

Transitivity—which is highly correlated to average clustering—is predictive of retention, suggesting that having a close-knit set of participants who bond well with each other is important for retention but not growth, which is consistent with existing literature on community growth [2, 30, 34]. Combined with our result for timing, we posit these results may point to the importance of a sense of belonging and identity within a community; users who join a rapidly growing community (low time-to-k) may find themselves lost in the mix and slow to accumulate social capital. It also suggests the importance of strong relationships, which take more time to build and are the combination of frequent engagement, deep interaction, and time spent together in the communities.

The distribution of arrival rates for comments and posts (measured through the Gini coefficient of gaps) point to two diverging recommendations. Bursty commenting followed by long droughts (high Gini in comment gaps) is negatively associated to all measures of success, except average #posts. Uniform arrival rates for comments leads to more success — though as the first recommendation suggests, these comments need not come from a large group of contributors. Here, we posit that because comments are a form of interaction and accumulation of social capital, regular arrival rates encourages users to come back frequently and follow up on existing discussions. This also might be explained by the *anticipation and uncertainty of reward mechanism* present in social media, where constant arrival of new content stimulates the production of the hormone Dopamine, a chemical produced in various parts of the brain and controls moods, motivation and sense of reward, which is linked to users' increased presence and activity in social media [54]. This result suggests that new community organizers should encourage new comments on regular intervals to promote opportunities for others to become involved in the discussion.

7 RELATED WORK

A large body of literature investigates how community-level characteristics influence the success of online communities. Here we discuss how the literature defines success and how the early users affect the likelihood of communities becoming successful.

Success as the number of members. The majority of works on community success adopt a very narrow view of how success manifest in communities. Most existing work tries to predict the volume of popularity of communities, such as the growth in the number of members the community. Often, the growth in membership is treated as process similar to the diffusion of innovations, where joining a new group is analogous to adopting an innovation [2, 8, 30, 34, 55]. Two works present a broader view of success, Kairam *et. al.* investigate how communities social networks predict growth and community survival, their definition of survival is based on the time the community stops to grow rather than stop to produce content, while Ellis et. al. [24] estimate the health of a community by measuring the retention rate of participants and the period of time the group stays active.

On the importance of early members. Another important line of work focuses on predicting the success of communities is the characteristics of its early members. Inspired by research on offline organizations that demonstrated the importance of the early members for organization success [21], they investigate whether user characteristics and behavior early in a community's history predict community success [55, 58, 59]. Early members are responsible for creating the initial content and norms of the community, acting as recruiters of new members. For example, Kraut and Fiore [39] found that human and social capital, such as users' age and experience with other communities' early decisions are predictive of success. Their findings suggest that the resources early members bring to the group are important, but they can also be detrimental if the group depends on them too much.

Our work contributes to prior research in two major aspects. First, we shed light over the idea that success is a complex concept and can be described in multiple dimensions that depend on the goal of a community. We thus investigate success across four desirable properties. Second, we present a set of features that represent the communities' behaviors and can generalize across the different types of communities. Finally, we show that success can be predicted and reveal which features are the most important for each success definition.

8 CONCLUSIONS

New groups are created frequently in online platforms. What does it take for a new group to succeed and how do we measure success? We answer these questions using a large-scale study on tens of thousands of groups on Reddit. Our work offers two main contributions. First, we quantify success using four measures and show, surprisingly, that success according to one measures is not necessary met with success with the other three. Instead, while the measures are positively correlated, groups succeed in their own way, in part due to the diversity of *why* a group forms. A small community may thrive through consistent posting, while never growing large, whereas another group may expand to a massive number of users, with only a few core people participating. Second, we show that the future success of a group can be predicted along each of the four measures. Drawing upon prior work and theories of group organization, we quantify six different group behaviors to test their predictiveness of success. Our results show that no single behavior drives a group to be successful in each dimension. For example, while retention and growth are well-predicted by the inequality of creating content, group longevity is best predicted by the linguistic style of the content. Further, predictive accuracy is enhanced when combining all behaviors, suggesting that each behavior plays a role in the eventual success (or failure) of a community.

Together, our results point to the complex roles groups serve online: each group is created with a different purpose and the success of that group is dependent in part on cultivating the types of behaviors needed. Our results also provide practical advice for individuals wanting to start their own new community online.

ACKNOWLEDGMENTS

This research was partly supported by the National Science Foundation under Grant No. IIS-1617820.

REFERENCES

[1] J. Arguello, B. Butler, E. Joyce, R. Kraut, K. Ling, C. Rosé, and X. Wang. 2006. Talk to Me: Foundations for Successful Individual-group Interactions in Online Communities. In *Proceedings of CHI*.

[2] Lars Backstrom, Dan Huttenlocher, Jon Kleinberg, and Xiangyang Lan. 2006. Group Formation in Large Social Networks: Membership, Growth, and Evolution. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[3] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. 2012. The Role of Social Networks in Information Diffusion. In *Proceedings of WWW*.

[4] Baumgartner, J. 2015. https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment/ [Online; accessed 02-November-2018].

[5] Baumgartner, J. 2018. Reddit repository. https://files.pushshift.io/reddit/ [Online; accessed 02-November-2018].

[6] Phillip Bonacich. 1987. Power and centrality: A family of measures. *American journal of sociology* (1987).

[7] Ronald S Burt. 2009. *Structural holes: The social structure of competition*. Harvard university press.

[8] Damon Centola. 2010. The spread of behavior in an online social network experiment. *science* (2010).

[9] E. Chandrasekharan, M. Samory, S. Jhaver, H. Charvat, A. Bruckman, C. Lampe, J. Eisenstein, and E. Gilbert. 2018. The Internet's Hidden Rules: An Empirical Study of Reddit Norm Violations at Micro, Meso, and Macro Scales. *Proceedings of the ACM on Human-Computer Interaction* CSCW (2018).

[10] Justin Cheng, Lada Adamic, P Alex Dow, Jon Michael Kleinberg, and Jure Leskovec. 2014. Can cascades be predicted?. In *Proceedings of the 23rd international conference on World wide web*.

[11] Justin Cheng, Lada A. Adamic, P. Alex Dow, Jon Kleinberg, and Jure Leskovec. 2014. Can cascades be predicted?. In *Proceedings of WWW*.

[12] Justin Cheng, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2014. How Community Feedback Shapes User Behavior.

[13] Cindy Chung and James W Pennebaker. 2007. The psychological functions of function words. *Social communication* (2007).

[14] James S Coleman. 1988. Social capital in the creation of human capital. *American journal of sociology* (1988).

[15] Nick Craswell. 2009. Mean reciprocal rank. In *Encyclopedia of Database Systems*.

[16] Tiago Cunha, Ingmar Weber, Hamed Haddadi, and Gisele Pappa. 2016. The Effect of Social Feedback in a Reddit Weight Loss Community. In *Proceedings of the 6th International Conference on Digital Health Conference*.

[17] Tiago Cunha, Ingmar Weber, and Gisele Pappa. 2017. A Warm Welcome Matters!: The Link Between Social Feedback and Weight Loss in /R/Loseit. In *Proceedings of the 26th International Conference on World Wide Web Companion*.

[18] Cristian Danescu-Niculescu-Mizil, Robert West, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. No Country for Old Members: User Lifecycle and Linguistic Change in Online Communities. In *Proceedings of WWW*.

[19] Cristian Danescu-Niculescu-Mizil, Robert West, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. No Country for Old Members: User Lifecycle and Linguistic Change in Online Communities. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13)*.

[20] K. Dasgupta, R. Singh, B. Viswanathan, D. Chakraborty, S. Mukherjea, A. Nanavati, and A. Joshi. 2008. Social Ties and Their Relevance to Churn in Mobile Telecom Networks. In *Proceedings of the 11th International Conference on Extending Database Technology: Advances in Database Technology*.

[21] Per Davidsson and Benson Honig. 2003. The role of social and human capital among nascent entrepreneurs. *Journal of business venturing* (2003).

[22] Robert Dorfman. 1979. A formula for the Gini coefficient. *The review of economics and statistics* (1979).

[23] Nicolas Ducheneaut, Nicholas Yee, Eric Nickell, and Robert J. Moore. 2007. The Life and Death of Online Gaming Communities. In *Proceedings of CHI*.

[24] Katherine Ellis, Moises Goldszmidt, Gert Lanckriet, Nina Mishra, and Omer Reingold. 2016. Equality and social mobility in Twitter discussion groups. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*.

[25] Matias JN. Gaffney D. 2018. Caveat emptor, computational social science: Large-scale missing data in a widely-published Reddit corpus. In *PLoS One*.

[26] Martin Gerlach, Beatrice Farb, William Revelle, and Luís A Nunes Amaral. 2018. A robust data-driven approach identifies four personality types across four large data sets. *Nature Human Behaviour* (2018).

[27] Jelle Goeman, Rosa Meijer, and Nimisha Chaturvedi. 2012. L1 and L2 penalized regression models. *cran. r-project. or* (2012).

[28] Lewis R Goldberg. 1990. An alternative" description of personality": the big-five factor structure. *Journal of personality and social psychology* (1990).

[29] Mark Granovetter. 1978. Threshold models of collective behavior. *American journal of sociology* (1978).

[30] Mark S Granovetter. 1977. The strength of weak ties. In *Social networks*.

[31] W. Hamilton, J. Zhang, C. Danescu-Niculescu-Mizil, D. Jurafsky, and J. Leskovec. 2017. Loyalty in online communities. In *Proceedings of the ICWSM*.

[32] Timothy A Judge, Chad A Higgins, Carl J Thoresen, and Murray R Barrick. 1999. The big five personality traits, general mental ability, and career success across the life span. *Personnel psychology* (1999).

[33] David Jurgens, James McCorriston, and Derek Ruths. 2017. An Analysis of Individuals' Behavior Change in Online Groups. In *International Conference on Social Informatics*.

[34] Sanjay Ram Kairam, Dan J. Wang, and Jure Leskovec. 2012. The Life and Death of Online Groups: Predicting Group Growth and Longevity. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*.

[35] Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika* (1938).

[36] Amy Jo Kim. 2000. *Community Building on the Web: Secret Strategies for Successful Online Communities* (1st ed.). Addison-Wesley Longman Publishing Co., Inc.

[37] Isabel Kloumann, Lada Adamic, Jon Kleinberg, and Shaomei Wu. 2015. The Lifecycles of Apps in a Social Ecosystem. In *Proceedings of WWW*.

[38] Robert E Kraut and Andrew T Fiore. 2014. The role of founders in building online groups. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*.

[39] Robert E Kraut and Paul Resnick. 2012. *Building successful online communities: Evidence-based social design*. Mit Press.

[40] Eric Kutcher, Olivia Nottebohm, and Kara Sprague. 2014. Grow fast or die slow. *McKinsey & Company (http://www. mckinsey. com/Insights/High_Tech_Telecoms_Internet/Grow_fast_or_die_slow)* (2014).

[41] Tara Matthews, Jalal U Mahmud, Jilin Chen, Michael Muller, Eben Haber, and Hernan Badenes. 2015. They said what?: Exploring the relationship between language use and member satisfaction in communities. In *Proceedings of the 18th ACM CSCW*.

[42] Edward Newell, David Jurgens, Haji Mohammad Saleem, Hardik Vala, Jad Sassine, Caitrin Armstrong, and Derek Ruths. 2016. User migration in online social networks: A case study on Reddit during a period of community unrest. In *Tenth International AAAI Conference on Web and Social Media*.

[43] Dong Nguyen and Carolyn P Rosé. 2011. Language use as a reflection of socialization in online communities. In *Proceedings of the Workshop on Languages in Social Media*.

[44] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. 2007. Quantitative social group dynamics on a large scale. *communities* (2007).

[45] James W Pennebaker, Ryan L Boyd, Kayla Jordan, and Kate Blackburn. 2015. *The development and psychometric properties of LIWC2015*. Technical Report.

[46] E Platt and DM Romero. 2018. Network Structure, Efficiency, and Performance in WikiProjects. In *Proceedings of International AAAI Conference on Web and Social Media (ICWSM)*.

[47] Emil Protalinski. 2014. Facebook passes 1.23 billion monthly active users, 945 million mobile users, and 757 million daily users. *The Next Web* (2014).

[48] Everett M Rogers. 2010. *Diffusion of innovations*. Simon and Schuster.

[49] Daniel M. Romero, Brendan Meeder, and Jon Kleinberg. 2011. Differences in the Mechanics of Information Diffusion Across Topics: Idioms, Political Hashtags, and Complex Contagion on Twitter. In *Proceedings of WWW*.

[50] Daniel M. Romero, Chenhao Tan, and Johan Ugander. 2013. On the interplay between social and topical structure. In *Proceedings of ICWSM*.

[51] Daniel M Romero, Brian Uzzi, and Jon Kleinberg. 2016. Social networks under stress. In *Proceedings of the 25th International Conference on World Wide Web*.

[52] J. Sherblom. 1990. Organization involvement expressed through pronoun use in computer mediated communication. *Communication Research Reports* (1990).

[53] Sunita Shukla, Parul Aggarwal, Bhavana Adhikari, and Vikas Singh. 2014. Relationship between employee engagement and big five personality factors: a study of online B2C e-commerce company. *JIMS8M: The Journal of Indian Management & Strategy* (2014).

[54] Molly Soat. 2015. Social Media Triggers a Dopamine High. *Marketing News* (2015).

[55] Chenhao Tan. 2018. Tracing Community Genealogy: How New Communities Emerge from the Old. In *Proceedings of ICWSM*.

[56] Chenhao Tan and Lillian Lee. 2015. All who wander: On the prevalence and characteristics of multi-community engagement. In *Proceedings of WWW*.

[57] Brian Uzzi, Satyam Mukherjee, Michael Stringer, and Ben Jones. 2013. Atypical combinations and scientific impact. *Science* (2013).

[58] Haiyi Zhu, Jilin Chen, Tara Matthews, Aditya Pal, Hernan Badenes, and Robert E. Kraut. 2014. Selecting an Effective Niche: An Ecological View of the Success of Online Communities. In *Proceedings of CHI*.

[59] Haiyi Zhu, Robert E. Kraut, and Aniket Kittur. 2014. The Impact of Membership Overlap on the Survival of Online Communities. In *Proceedings of CHI*.

[60] A Ziapour and N Kianipour. 2015. A study of the relationship between characteristic traits and Employee Engagement (A case study of nurses across Kermanshah, Iran in 2015). *Journal of medicine and life* (2015).

Adversarial Training Methods for Network Embedding

Quanyu Dai
The Hong Kong Polytechnic University
Kowloon, Hong Kong
dqyzm100@hotmail.com

Xiao Shen
The Hong Kong Polytechnic University
Kowloon, Hong Kong
shenxiaocam@163.com

Liang Zhang
JD.com
Beijing, China
zhangliang16@jd.com

Qiang Li
Y-tech, Kwai
Beijing, China
leetsiang.cloud@gmail.com

Dan Wang
The Hong Kong Polytechnic University
Kowloon, Hong Kong
csdwang@comp.polyu.edu.hk

ABSTRACT

Network Embedding is the task of learning continuous node representations for networks, which has been shown effective in a variety of tasks such as link prediction and node classification. Most of existing works aim to preserve different network structures and properties in low-dimensional embedding vectors, while neglecting the existence of noisy information in many real-world networks and the overfitting issue in the embedding learning process. Most recently, generative adversarial networks (GANs) based regularization methods are exploited to regularize embedding learning process, which can encourage a global smoothness of embedding vectors. These methods have very complicated architecture and suffer from the well-recognized non-convergence problem of GANs. In this paper, we aim to introduce a more succinct and effective local regularization method, namely adversarial training, to network embedding so as to achieve model robustness and better generalization performance. Firstly, the adversarial training method is applied by defining adversarial perturbations in the embedding space with an adaptive L_2 norm constraint that depends on the connectivity pattern of node pairs. Though effective as a regularizer, it suffers from the interpretability issue which may hinder its application in certain real-world scenarios. To improve this strategy, we further propose an interpretable adversarial training method by enforcing the reconstruction of the adversarial examples in the discrete graph domain. These two regularization methods can be applied to many existing embedding models, and we take DeepWalk as the base model for illustration in the paper. Empirical evaluations in both link prediction and node classification demonstrate the effectiveness of the proposed methods.

KEYWORDS

Network Embedding, Adversarial Training, Robustness

ACM Reference Format:
Quanyu Dai, Xiao Shen, Liang Zhang, Qiang Li, and Dan Wang. 2019. Adversarial Training Methods for Network Embedding. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313445

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313445

1 INTRODUCTION

Network embedding strategies, as an effective way for extracting features from graph structured data automatically, have gained increasing attention in both academia and industry in recent years. The learned node representations from embedding methods can be utilized to facilitate a wide range of downstream learning tasks, including some traditional network analysis tasks such as link prediction and node classification, and many important applications in industry such as product recommendation in e-commerce website and advertisement distribution in social networks. Therefore, under such great application interest, substantial efforts have been devoted to designing effective and scalable network embedding models.

Most of the existing works focus on preserving network structures and properties in low-dimensional embedding vectors [4, 35, 37]. Firstly, DeepWalk [35] defines random walk based neighborhood for capturing node dependencies, and node2vec [13] extends it with more flexibility in balancing local and global structural properties. LINE [35] preserves both first-order and second-order proximities through considering existing connection information. Further, GraRep [4] manages to learn different high-order proximities based on different k-step transition probability matrix. Aside from the above mentioned structure-preserving methods, several research works investigate the learning of property-aware network embeddings. For example, network transitivity, as the driving force of link formation, is considered in [26], and node popularity, as another important factor affecting link generation, is incorporated into RaRE [14] to learn social-rank aware and proximity-preserving embedding vectors. However, the existence of nosiy information in real-world networks and the overfitting issue in the embedding learning process are neglected in most of these methods, which leaves the necessity and potential improvement space for further exploration.

Most recently, adversarial learning regularization method is exploited for improving model robustness and generalization performance in network embedding [8, 41]. ANE [8] is the first try in this direction, which imposes a prior distribution on embedding vectors through adversarial learning. Then, the adversarially regularized autoencoder is adopted in NetRA [41] to overcome the mode-collapse problem in ANE method. These two methods both encourage the global smoothness of the embedding distribution based on generative adversarial networks (GANs) [11]. Thus, they have very complicated frameworks and suffer from the well-recognized hard training problems of GANs [3, 29].

In this paper, we aim to leverage the adversarial training (AdvT) method [12, 34] for network embedding to achieve model robustness and better generalization ability. AdvT is a local smoothness regularization method with more succinct architecture. Specifically, it forces the learned classifier to be robust to adversarial examples generated from clean ones with small crafted perturbation [34]. Such designed noise with respect to each input example is dynamically obtained through finding the direction to maximize model loss based on current model parameters, and can be approximately computed with fast gradient method [12]. It has been demonstrated to be extremely useful for some classification problems [12, 23].

However, how to adapt AdvT for graph representation learning remains an open problem. It is not clear how to generate adversarial examples in the discrete graph domain since the original method is designed for continuous inputs. In this paper, we propose an adversarial training DeepWalk model, which defines the adversarial examples in the embedding space instead of the original discrete relations and obtains adversarial perturbation with fast gradient method. We also leverage the dependencies among nodes based on connectivity patterns in the graph to design perturbations with different L_2 norm constraints, which enables more reasonable adversarial regularization. The training process can be formulated as a two-player game, where the adversarial perturbations are generated to maximize the model loss while the embedding vectors are optimized against such designed noises with stochastic gradient descent method. Although effective as a regularization technique, directly generating adversarial perturbation in embedding space with fast gradient method suffers from interpretability issue, which may restrict its application areas. Further, we manage to restore the interpretability of adversarial examples by constraining the perturbation directions to embedding vectors of other nodes, such that the adversarial examples can be considered as the substitution of nodes in the original discrete graph domain.

Empirical evaluations show the effectiveness of both adversarial and interpretable adversarial training regularization methods by building network embedding method upon DeepWalk. It is worth mentioning that the proposed regularization methods, as a principle, can also be applied to other embedding models with embedding vectors as model parameters such as node2vec and LINE. The main contributions of this paper can be summarized as follows:

- We introduce a novel, succinct and effective regularization technique, namely adversarial training method, for network embedding models which can improve both model robustness and generalization ability.
- We leverage the dependencies among node pairs based on network topology to design perturbations with different L_2 norm constraints for different positive target-context pairs, which enables more flexible and effective adversarial training regularization.
- We also equip the adversarial training method with interpretability for discrete graph data by restricting the perturbation directions to embedding vectors of other nodes, while maintaining its usefulness in link prediction and only slightly sacrificing its regularization ability in node classification.
- We conduct extensive experiments to evaluate the effectiveness of the proposed methods.

2 BACKGROUND

2.1 Framework of Network Embedding

The purpose of network embedding is to transform discrete network structure information into compact embedding vectors, which can be further used to facilitate downstream learning tasks, such as node classification and link prediction. The research problem can be formally formulated as follows: Given a weighted (unweighted) directed (undirected) graph $G(V, E, A)$ ($N = |V|$), with $V = \{v_i\}_{i=1}^{N}$ as the node set, $E = \{e_{ij}\}_{i,j=1}^{N}$ as the edge set, and A as the weighted adjacency matrix with A_{ij} quantifying the strength of the relationship between node v_i and v_j, network embedding is aimed at learning a mapping function $f : V \mapsto U$, where $U \in R^{N \times d}$ ($d \ll N$) is the embedding matrix with the ith row $\boldsymbol{u}_i{}^T$ as the embedding vector of node v_i. Note that for many network embedding models, a context embedding matrix U' will also be learned. For these methods, embedding matrix U is also called target embedding matrix.

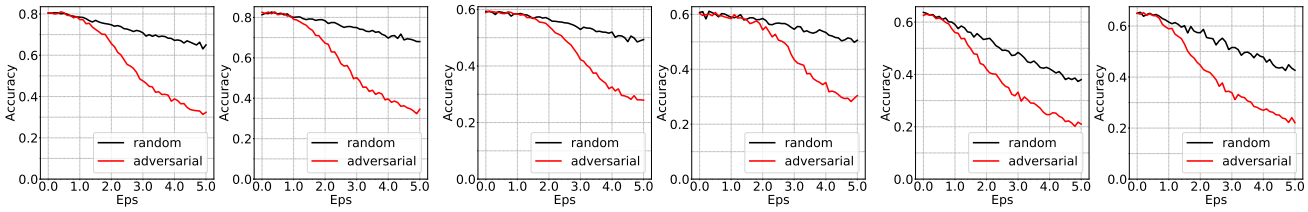
The learning framework of many famous network embedding methods, such as DeepWalk [27], LINE [35] and node2vec [13], can be summarized into two phases: a sampling phase that determines node pairs with strong relationships, and an optimization phase that tries to preserve pairwise relationships in the embedding vectors through the negative sampling approach [22]. In particular, in the first phase, these three methods capture structural information by defining different neighborhood structures, such as random walk explored neighborhood in [13, 27], first-order and second-order proximities in [35]. We denote the generalized neighbors (not restricted to directly connected nodes) of node v_i as $\mathcal{N}(v_i)$, i.e., nodes in this set are closely related with v_i and should be close with v_i in the embedding space. The loss function of this framework can be abstracted as follows:

$$\begin{aligned}\mathcal{L}(G|\Theta) = -\sum_{v_i \in V}\sum_{v_j \in \mathcal{N}(v_i)} \{\log \sigma(s(v_i, v_j|\Theta)) \\ + \sum_{k=1}^{K} \mathbb{E}_{v_k \sim P_k(v)}[\log \sigma(-s(v_i, v_k|\Theta))]\},\end{aligned} \tag{1}$$

where Θ represents model parameters such as target and context embedding matrices, $s(v_i, v_j|\Theta)$ represents the similarity score of node v_i and v_j based on model parameters Θ, and $\sigma(\cdot)$ is the sigmoid function. $P_k(v)$ denotes the distribution for sampling negative nodes, and a simple variant of unigram distribution is usually utilized, i.e., $P_k(v) \propto d_v^{3/4}$, where d_v is the out-degree of node v. Eq. (1) is actually a cross entropy loss with closely related node pair (v_i, v_j) as positive samples and (v_i, v_k) as negative samples, and thus network embedding can be considered as a classification problem.

2.2 Adversarial Training

Adversarial training [12, 34] is a newly proposed effective regularization method for classifiers which can not only improve the robustness of the model against adversarial attacks, but also achieve better generalization performance on learning tasks. It augments the original clean data with the dynamically generated adversarial examples, and then trains the model with the newly mixed examples. Denote the input as $\boldsymbol{x}$ and model parameters as $\boldsymbol{\theta}$. The loss on adversarial examples can be considered as a regularization term in



(a) Cora, training ratio=50%, 80%. (b) Citeseer, training ratio=50%, 80%. (c) Wiki, training ratio=50%, 80%.

Figure 1: Impact of applying adversarial and random perturbations to the embedding vectors learned by DeepWalk on Cora, Citeseer and Wiki on multi-class classification with training ratio as 50% and 80%. Note that "random" represents random perturbations (noises generated from a normal distribution), while "adversarial" represents adversarial perturbations.

the trained classifier $p(y|\cdot)$, which is as follows:

$$-\log p(y|\boldsymbol{x}+\boldsymbol{n}_{adv};\boldsymbol{\theta}), \tag{2}$$

$$where\ \boldsymbol{n}_{adv} = \underset{\boldsymbol{n},\ \|\boldsymbol{n}\|\leq\epsilon}{\arg\min} \log p(y|\boldsymbol{x}+\boldsymbol{n};\hat{\boldsymbol{\theta}}), \tag{3}$$

where $\boldsymbol{n}$ is the perturbation on the input, ϵ represents the norm constraint of $\boldsymbol{n}$, and $\hat{\boldsymbol{\theta}}$ are current model parameters but fixed as constants. We employ L_2 norm in this paper, while L_1 norm has also been used in the literature [12]. Eq. (2) means that the model should be robust on the adversarial perturbed examples. Before each batch training, the adversarial noise $\boldsymbol{n}$ with respect to the input $\boldsymbol{x}$ is firstly generated by solving optimization problem (3) to make it resistant to current model. Since it is difficult to calculate Eq.(3) exactly in general, fast gradient descent method [12] is widely used to obtain the adversarial noise approximately by linearizing $\log p(y|\boldsymbol{x};\boldsymbol{\theta})$ around $\boldsymbol{x}$. Specifically, the adversarial perturbation with L_2 norm constraint can be obtained as follows:

$$\boldsymbol{n}_{adv} = -\epsilon \cdot \frac{\boldsymbol{g}}{\|\boldsymbol{g}\|_2}\ where\ \boldsymbol{g} = \nabla_{\boldsymbol{x}} \log p(y|\boldsymbol{x};\hat{\boldsymbol{\theta}}). \tag{4}$$

It can be easily calculated with backpropagation method.

2.3 Motivation

To improve the generalization ability of network embedding models, two ways have been used: firstly, some denoising autoencoder based methods [5, 8] improve model robustness by adding random perturbation to input data or hidden layers of deep models; secondly, some existing methods [8, 41] regularize embedding vectors from a global perspective through GAN-based method, i.e., encouraging the global smoothness of the distribution of embeddings. In this paper, we aim to introduce a novel, more succinct and effective regularization method for network embedding models, i.e., adversarial training (AdvT) [12]. AdvT generates crafted adversarial perturbations to model inputs and encourages local smoothness for improving model robustness and generalization performance, which can be expected to be more effective than the random perturbation methods [5] and global regularization methods [8, 41]. In the following, we would like to compare the impact of adversarial and random perturbation on embedding vectors to better motivate this new regularization method.

However, it is not clear how to integrate adversarial training into existing network embedding methods. Graph data is discrete, and the continuous adversarial noise can not be directly imposed on the discrete connected information. To bypass this difficulty, we seek to define the adversarial perturbation on embedding vectors instead of the discrete graph domain as inspired by [23]. We define the adversarial perturbation on node embeddings as follows:

$$\boldsymbol{n}_{adv} = \underset{\boldsymbol{n},\ \|\boldsymbol{n}\|\leq\epsilon}{\arg\max}\ \mathcal{L}(G|\hat{\Theta}+\boldsymbol{n}), \tag{5}$$

which can be further approximated with fast gradient method as presented in Eq. (4).

Take DeepWalk [27] with negative sampling loss as an illustrative example. We explore the effect of adversarial perturbations on embedding vectors by adding them to the learned embedding vectors from DeepWalk, and then perform multi-class classification with the perturbed embeddings on several datasets. Besides, we choose random perturbations as the compared baseline, i.e., noises generated from a normal distribution. Figure 1 displays node classification results with varying L_2 norm constraints on the perturbations. We can find that embedding vectors are much more vulnerable to adversarial perturbations than random ones. For example, when ε is set to 2.0, the performance of node classification with training ratio as 80% on Cora drops 3.35% under random perturbation, while that decreases 16.25% under adversarial perturbation which is around 4 times more serious. If the embedding vectors can be trained to be more robust on adversarial noises, we can expect more significant improvements in generalization performance.

3 PROPOSED METHODS

In this section, we first describe the adapted adversarial training method for network embedding models, and present the algorithm based on DeepWalk. Then, we will tackle its interpretability issue by designing a new adversarial perturbation generation method.

3.1 Adversarial Training DeepWalk

Figure 2 shows the framework of DeepWalk with adversarial training regularization. It consists of two phases: a sampling phase that determines node pairs with strong relationships, and an optimization phase that tries to preserve pairwise relationships in the embedding vectors based on negative sampling approach. Note that in this paper we take DeepWalk as the illustrative example, and the proposed framework can be applied to the network embedding methods, such as LINE and node2vec, with the main difference in the sampling phase only.

In the first phase, DeepWalk transforms the network into node sequences by truncated random walk. For each node $v_i \in V$, η

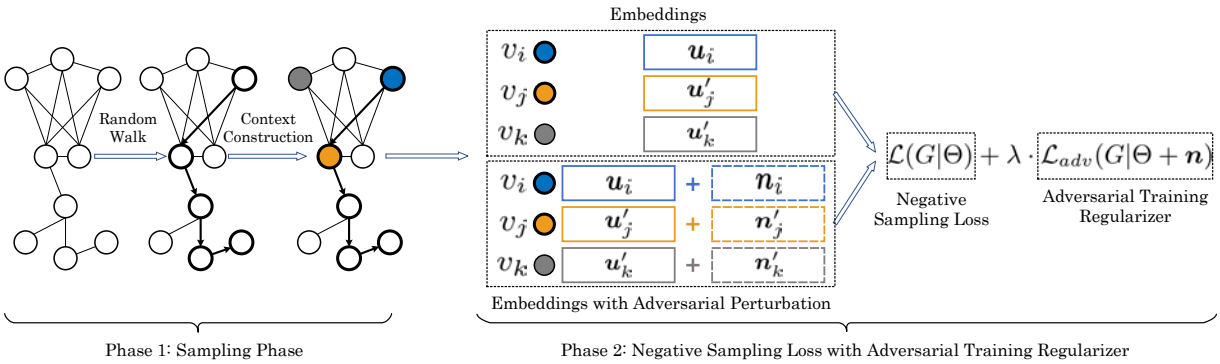


Figure 2: DeepWalk with Adversarial Training Regularization

sequences each with l nodes will be randomly sampled based on network structure with v_i as the starting point. In every walking step, the next node v_j will be sampled from the neighbors of current node v_k with the probability proportional to the edge strength A_{kj} between v_k and v_j. In practice, the alias table method [19] is usually leveraged for node sampling given the weight distribution of neighbors of current node, which only takes $O(1)$ time in a single sampling step. Then in the context construction process, closely related node pairs will be determined based on the sampled node sequences. Denote a node sequence as S with the ith node as s_i. The positive target-context pairs from S is defined as $\mathcal{P} = \{(s_i, s_j) : |i-j| \leq c\}$, where c represents the window size. With the constructed node pairs, the negative sampling loss will be optimized, which is defined as follows:

$$\begin{aligned}\mathcal{L}(G|\Theta) = -\sum_{v_i \in V}\sum_{v_j \in N(v_i)} \{\log \sigma(\boldsymbol{u'}_j^T \cdot \boldsymbol{u}_i) \\ + \sum_{k=1}^{K} \mathbb{E}_{v_k \sim P_k(v)}[\log \sigma(-\boldsymbol{u'}_k^T \cdot \boldsymbol{u}_i)]\},\end{aligned} \tag{6}$$

where (v_i, v_j) is from the constructed positive target-context pairs, and $\boldsymbol{u}_i$ and $\boldsymbol{u'}_j$ are the target embedding of node v_i and context embedding of node v_j respectively.

For the adversarial version of DeepWalk, an adversarial training regularization term is added to the original loss to help learn robust node representations against adversarial perturbations. The regularization term shares the same set of model parameters with the original model, but with the perturbed target and context embeddings as input. Existing methods consider the input examples independently, and impose the unique L_2 norm constraint on all adversarial perturbations [12, 23]. For graph structured data, entities often correlate with each other in a very complicated way, so it is inappropriate to treat all positive target-context relations equally without discrimination. Adversarial regularization helps alleviate overfitting issue, but it may also bring in some noises that can hinder the preservation of structural proximities, i.e., adding noises to those inherently closely-related node pairs will prevent them from having similar embeddings. Thus, we take advantages of the dependencies among nodes to assign different L_2 norm constraints to different positive target-context relations adaptively. Specifically, the more closely two nodes are connected, the smaller the constraint should be. The intuition is that less noises should be added to those node pairs which are inherently strongly-connected in the original network, thus they can be pushed closer in the embedding space with high flexibility, while for those weakly-connected pairs larger constraint can help alleviate the overfitting issue.

We obtain the similarity score of two nodes through computing the shifted positive pointwise mutual information matrix [18]:

$$M_{ij} = \max\{\log(\frac{\hat{M}_{ij}}{\sum_k \hat{M}_{kj}}) - \log(\beta), 0\}, \tag{7}$$

where $\hat{M} = \hat{A} + \hat{A}^2 + \cdots + \hat{A}^t$ captures different high-order proximities, $\hat{A}$ is the 1-step probability transition matrix obtained from A after the row-wise normalization, and β is a shift factor. We set t to 4 and β to $\frac{1}{N}$ in the experiments. Then, the adaptive scale factor for the L_2 norm constraint of the target-context pair v_i and v_j is calculated as below:

$$\Phi_{ij} = 1 - M_{ij} / \max\{M\}, \tag{8}$$

where $\max\{M\}$ represents the maximum entry of matrix M. Since $M_{ij} > 0$ $(\forall i, j)$, $\Phi_{ij} \in [0, 1]$. For those strongly-connected target-context pairs, the adaptive scale factor can help scale down the L_2 norm constraint of the adversarial perturbation, and thus alleviate the negative effect from the noises.

Then, the adversarial training regularizer with scale factor for L_2 norm constraint is defined as follows:

$$\begin{aligned}&\mathcal{L}_{adv}(G|\Theta + \boldsymbol{n}_{adv}) \\ &= -\sum_{v_i \in V}\sum_{v_j \in N(v_i)} \{\log \sigma((\boldsymbol{u'}_j + \Phi_{ij} \cdot (\boldsymbol{n'}_j)_{adv})^T \cdot (\boldsymbol{u}_i + \Phi_{ij} \cdot (\boldsymbol{n}_i)_{adv})) \\ &\quad + \sum_{k=1}^{K} \mathbb{E}_{v_k \sim P_k(v)}[\log \sigma(-(\boldsymbol{u'}_k + (\boldsymbol{n'}_k)_{adv})^T \cdot (\boldsymbol{u}_i + (\boldsymbol{n}_i)_{adv}))]\},\end{aligned} \tag{9}$$

where $(\boldsymbol{n}_i)_{adv}$ and $(\boldsymbol{n'}_j)_{adv}$ represent the original adversarial perturbation for target embedding of node v_i and context embedding of node v_j respectively.

Finally, one key problem is how to compute the adversarial perturbation for the given embedding vector of a node v. Here, we follow the famous adversarial training method directly [12, 34], and generate the perturbation noises to maximize model loss under current model parameters. The adversarial perturbation for node v is defined as follows:

$$\boldsymbol{n}_{adv} = \underset{\boldsymbol{n},\, \|\boldsymbol{n}\| \leq \epsilon}{\arg\max}\, \mathcal{L}(G|\hat{\Theta} + \boldsymbol{n}). \tag{10}$$

Algorithm 1: The adversarial training DeepWalk

Input : graph $G(V, E, A)$, window size c, embedding size d, walks per node η, negative size K, walk length l, adversarial noise level ϵ, adversarial regularization strength λ, batch size b

Output : Embedding matrix U

1 Initialize target and context embeddings with DeepWalk;
2 **while** *not converge* **do**
3 Generate a set of positive target-context pairs $\mathcal{P}$ with random walk based method;
4 **repeat**
5 Sample a batch $\mathcal{B}$ of target-context pairs from $\mathcal{P}$;
6 // Generate adversarial perturbations
7 $\boldsymbol{n}_{adv} = \epsilon \frac{\boldsymbol{g}}{\|\boldsymbol{g}\|_2}$ where $\boldsymbol{g} = \nabla_{\boldsymbol{u}} \mathcal{L}(G|\hat{\Theta})$ for each node v in the batch;
8 // Optimize model parameters
9 Update target and context embeddings by applying gradient descent technique to Eq. (12) ;
10 **until** $[|\mathcal{P}|/b]_+$ *times*;
11 **end**

It can be further approximated with fast gradient method as follows:

$$\boldsymbol{n}_{adv} = \epsilon \frac{\boldsymbol{g}}{\|\boldsymbol{g}\|_2} \ where \ \boldsymbol{g} = \nabla_{\boldsymbol{u}} \mathcal{L}(G|\hat{\Theta}). \tag{11}$$

Therefore, the overall loss for the proposed adversarial training DeepWalk is defined as follows:

$$\mathcal{L}(G|\Theta) + \lambda \cdot \mathcal{L}_{adv}(G|\Theta + \boldsymbol{n}_{adv}), \tag{12}$$

where λ is a hyperparameter to control the importance of the regularization term.

In this paper, we utilize DeepWalk with negative sampling loss as the base model for building the adversarial version of network embedding methods. Since the original implementation is based on the well encapsulated library, which lacks flexibility for further adaption, we re-implement the model with tensorflow [1] and utilize a slightly different training strategy. Specifically, in each training epoch, we independently construct positive target-context pairs with random walk based method, and then optimize model parameters with mini-batch stochastic gradient descent technique. Algorithm 1 summarizes the training procedure for the adversarial training DeepWalk. The model parameters are firstly initialized by training DeepWalk with the method introduced above. For each batch training, adversarial perturbations are generated with fast gradient method for each node in the batch as presented in Line 7. Then, the target and context embeddings will be updated by optimizing the negative sampling loss with adversarial training regularization as shown in Line 9. Asynchronous version of stochastic gradient descent [25] can be utilized to accelerate the training as DeepWalk. Note that we ignore the derivative of $\boldsymbol{n}_{adv}$ with respect to model parameters. The adversarial perturbations can be computed with simple back-propagation method, which enjoys low computational cost. Thus, the adversarial training DeepWalk is scalable as the base model.

3.2 Interpretable Adversarial Training DeepWalk

Adversarial examples refer to examples that are generated by adding viciously designed perturbations with norm constraint to the clean input, which can significantly increase model loss and probably induce prediction error [34]. Take an example from [12] for illustration, a "panda" image with imperceptibly small adversarial perturbation is assigned to be "gibbon" by the well-trained classification model with high confidence, while the original image can be correctly classified. Such adversarial examples can be well interpreted since the perturbations are imposed on the input space. For the adversarial training DeepWalk, adversarial perturbations are added to node embeddings instead of the discrete nodes and connected relations, and thus can not be easily reconstructed in the discrete graph domain. Though effective as a regularizer for improving model generalization performance, it suffers from lack of interpretability, which may create a barrier for its adoption in some real-world applications.

In this section, we propose an interpretable adversarial DeepWalk model by restoring the interpretability of adversarial perturbations. Instead of pursuing the worst perturbation direction only, we restrict the direction of perturbations toward a subset of nodes in the graph in the embedding space, such as the neighbors of the considered node. In this way, the adversarial perturbations in the node embedding space might be interpreted as the substitution of nodes in the original input space, i.e., the discrete target-context relations. However, there might be a certain level of sacrifice on the regularization performance because of the restriction on perturbation directions.

The *direction vector* from node v_t to v_k in the embedding space is defined as follows:

$$\boldsymbol{v}_k^{(t)} = \frac{\tilde{\boldsymbol{v}}_k^{(t)}}{\|\tilde{\boldsymbol{v}}_k^{(t)}\|_2}, \ where \ \tilde{\boldsymbol{v}}_k^{(t)} = \boldsymbol{u}_k - \boldsymbol{u}_t. \tag{13}$$

Denote $V^{(t)} \subseteq V$ ($|V^{(t)}| = T$, $|V^{(t)}| \ll |V|$) as a set of nodes for generating adversarial perturbation for node v_t. We define $V^{(t)}$ as the top T nearest neighbors of node v_t in the embedding space based on current model parameters. To improve model efficiency, we can also obtain $V^{(t)}$ based on the pretrained model parameters, and fix it for all training epochs. We use the latter strategy for experiments in this paper. Denote $\boldsymbol{w}^{(t)} \in \mathcal{R}^T$ as the weight vector for node v_t with $w_k^{(t)}$ representing the weight associated with direction vector $\boldsymbol{v}_k^{(t)}$, where v_k is the kth node in $V^{(t)}$. The interpretable perturbation for v_t is defined as the weighted sum of the direction vectors starting from v_t and ending with nodes in $V^{(t)}$:

$$\boldsymbol{n}(\boldsymbol{w}^{(t)}) = \sum_{k=1}^{T} w_k^{(t)} \boldsymbol{v}_k^{(t)}, \ v_k \in V^{(t)}, \ \forall k = 1, \cdots, T. \tag{14}$$

The adversarial perturbation is obtained by finding the weights that can maximize the model loss:

$$\boldsymbol{w}_{iAdv}^{(t)} = \underset{\boldsymbol{w}^{(t)}, \|\boldsymbol{w}^{(t)}\| \leq \epsilon}{\arg\max} \mathcal{L}_{iAdv}(G|\Theta + \boldsymbol{n}(\boldsymbol{w}^{(t)})), \tag{15}$$

where $\mathcal{L}_{iAdv}$ is obtained by replacing $\boldsymbol{n}_{adv}$ in Eq. (9) with $\boldsymbol{n}(\boldsymbol{w}^{(t)})$. In consideration of model efficiency, the above regularization term is approximated with first-order Taylor series for easy computation as

Table 1: Statistics of benchmark datasets

Name	Cora	Citeseer	Wiki	CA-GrQc	CA-HepTh
$\|V\|$	2,708	3,264	2,363	5,242	9,877
$\|E\|$	5,278	4,551	11,596	14,484	25,973
Avg. degree	1.95	1.39	4.91	2.76	2.63
#Labels	7	6	17	-	-

in [12]. Thus, the weights for constructing interpretable adversarial perturbation for node v_t can be computed as follows:

$$\boldsymbol{w}_{iAdv}^{(t)} = \epsilon \frac{\boldsymbol{g}}{\|\boldsymbol{g}\|_2} \; where \; \boldsymbol{g} = \nabla_{\boldsymbol{w}^{(t)}} \mathcal{L}_{iAdv}(G|\Theta + \boldsymbol{n}(\boldsymbol{w}^{(t)})). \quad (16)$$

Substituting $\boldsymbol{w}_{iAdv}^{(t)}$ into Eq. (14), we can get the adversarial perturbations $\boldsymbol{n}(\boldsymbol{w}_{iAdv}^{(t)})$. Further, by replacing $\boldsymbol{n}_{adv}$ with $\boldsymbol{n}(\boldsymbol{w}_{iAdv}^{(t)})$ in Eq. (9), we can have the interpretable adversarial training regularizer for DeepWalk. The algorithm for interpretable adversarial training DeepWalk is different from Algorithm 1 in the way of generating adversarial perturbations, and thus we do not present it in this paper due to space limitation. Since $|V^{(t)}| \ll |V|$, the computation of adversarial perturbation for one node takes constant time. Therefore, the time complexity of this model is also linear to the number of nodes in the graph as DeepWalk.

4 EXPERIMENTS

In this section, we empirically evaluate the proposed methods through performing link prediction and node classification on several benchmark datasets.

4.1 Experiment Setup

4.1.1 Datasets. We conduct experiments on several benchmark datasets from various real-world applications. Table 1 shows some statistics of them. Note that we do some preprocessing on the original datasets by deleting self-loops and nodes with zero degree. Some descriptions of these datasets are summarized as follows:

- **Cora, Citeseer** [21]: Paper citation networks. Cora consists of 2708 papers with 7 categories, and Citeseer consists 3264 papers including 6 categories.
- **Wiki** [31]: Wiki is a network with nodes as web pages and edges as the hyperlinks between web pages.
- **CA-GrQc, CA-HepTh** [17]: Author collaboration networks. They describe scientific collaborations between authors with papers submitted to General Relativity and Quantum Cosmology category, and High Energy Physics, respectively.

4.1.2 Baseline Models. The descriptions of the baseline models are as follows:

- **Graph Factorization (GF)** [2]: GF directly factorizes the adjacency matrix with stochastic gradient descent technique to obtain the embeddings, which enables it scale to large networks.
- **DeepWalk** [27]: DeepWalk regards node sequence obtained from truncated random walk as word sequence, and then uses skip-gram model to learn node representations. We directly use the publicly available source code with hierarchical softmax approximation for experiments.
- **LINE** [35]: LINE preserves network structural proximities through modeling node co-occurrence probability and node conditional probability, and leverages the negative sampling approach to alleviate the expensive computation.
- **node2vec** [13]: node2vec differs from DeepWalk by proposing more flexible method for sampling node sequences to strike a balance between local and global structural properties.
- **GraRep** [4]: GraRep applies SVD technique to different k-step probability transition matrix to learn node embeddings, and finally obtains global representations through concatenating all k-step representations.
- **AIDW** [8]: AIDW is an inductive version of DeepWalk with GAN-based regularization method. A prior distribution is imposed on node representations through adversarial learning to achieve a global smoothness in the distribution.

Our implemented version of DeepWalk is based on negative sampling approach, thus we denote it as Dwns to avoid confusion. We also include a baseline, namely Dwns_rand, with noises from a normal distribution as perturbations in the regularization term. Following existing work [30], we denote the adversarial training DeepWalk as Dwns_AdvT, and the interpretable adversarial training DeepWalk as Dwns_iAdvT in the rest of the paper.

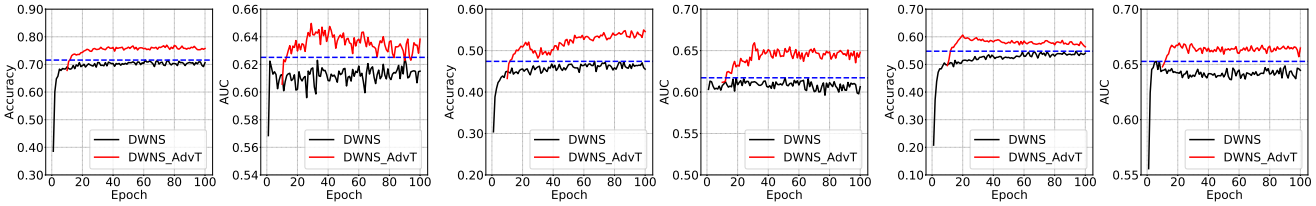
4.1.3 Parameter Settings. For Dwns and its variants including Dwns_rand, Dwns_AdvT and Dwns_iAdvT, the walk length, walks per node, window size, negative size, regularization strength, batch size and learning rate are set to 40, 1, 5, 5, 1, 1024 and 0.001, respectively. The adversarial noise level ϵ has different settings in Dwns_AdvT and Dwns_iAdvT, while Dwns_rand follows the settings of Dwns_AdvT. For Dwns_AdvT, ϵ is set to different value for different datasets. Specifically, ϵ is set to 0.9 for Cora, 1.1 for Citeseer in both link prediction and node classification, and 0.6 and 0.5 for Wiki in node classification and link prediction respectively, while it is set to 0.5 for all other datasets in these two learning tasks. For Dwns_iAdvT, ϵ is set to 5 for all datasets in both node classification and link prediction tasks, and the size of the nearest neighbor set T is set to 5. Besides, the dimension of embedding vectors are set to 128 for all methods.

4.2 Impact of Adversarial Training Regularization

In this section, we conduct link prediction and multi-class classification on adversarial training DeepWalk, i.e., Dwns_AdvT, to study the impact of adversarial training regularization on network representation learning from two aspects: model performance on different training epochs and model performance under different model size.

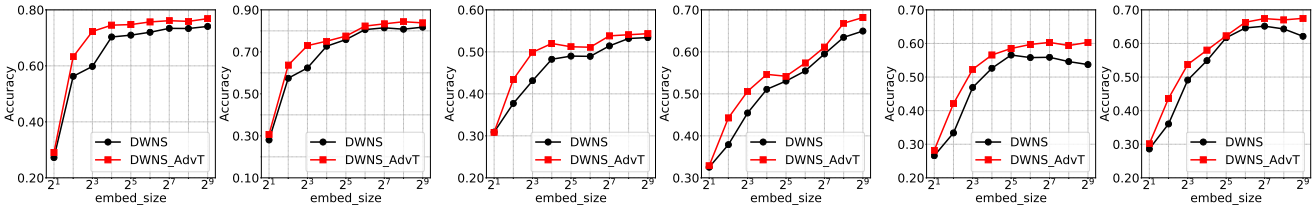
Node classification is conducted with support vector classifier in Liblinear package[1] [10] in default settings with the learned embedding vectors as node features. In link prediction, network embedding is first performed on a sub-network, which contains 80% of edges in the original network, to learn node representations. Note that the degree of each node is ensured to be greater than or equal to 1 during subsampling process to avoid meaningless embedding vectors. We use AUC score as the performance measure, and treat

[1] https://www.csie.ntu.edu.tw/~cjlin/liblinear/



(a) Cora. (b) Citeseer. (c) Wiki.

Figure 3: Training curves of node classification (left, training ratio 10%) and link prediction (right).



(a) Cora, training ratio=10%, 50%. (b) Citeseer, training ratio=10%, 50%. (c) Wiki, training ratio=10%, 50%.

Figure 4: Performance comparison between DWNS and DWNS_AdvT on multi-class classification with training ratio as 10% (left) and 50% (right) respectively under varying embedding size.

link prediction as a classification problem. Specifically, a L_2-SVM classifier is trained with edge feature inputs obtained from the Hadamard product of embedding vectors of two endpoints as many other works [13, 40], positive training samples as the observed 80% edges, and the same number of negative training samples randomly sampled from the network, i.e., node pairs without direct edge connection. The testing set consists of the hidden 20% edges and two times of randomly sampled negative edges. All experimental results are obtained by making an average of 10 different runs.

4.2.1 Training Process. We train DWNS model for 100 epochs, and evaluate the generalization performance of the learned embedding vectors in each epoch with node classification and link prediction on Cora, Citeseer and Wiki. We also conduct similar experiments on DWNS_AdvT for 90 epochs with the model parameters initialized from those of DWNS after $10th$ training epochs. Figures 3 shows the experimental results.

In general, adversarial training regularization can bring a significant improvement in generalization ability to DWNS through the observation of training curves in both node classification and link prediction. Specifically, after 10 training epochs, the evaluation performance has little improvements for all datasets in two learning tasks with further training for DWNS, while adversarial training regularization leads to an obvious performance increase. In Figure 3, the blue line is drew by setting its vertical coordinates as the maximum value of the metrics of DWNS in the corresponding experiments. We can find that the training curve of DWNS_AdvT is continuously above the blue line in different training epochs. Particularly, there is an impressive 7.2% and 9.2% relative performance improvement in link prediction for Cora and Citeseer respectively. We notice that the performance of DWNS_AdvT drops slightly after about 40 training epochs for Cora in link prediction, and about 20 training epochs for Wiki in node classification. The reason might be that some networks are more vulnerable to overfitting, and deeper understanding of this phenomenon needs further exploration.

4.2.2 Performance vs. Embedding Size. We explore the effect of adversarial regularization under different model size with multi-class classification. Figure 4 demonstrates the classification results on Cora, Citeseer and Wiki with training ratio as 10% and 50%. In general, adversarial training regularization is essential for improving model generalization ability. Across all tested embedding size, our proposed adversarial training DEEPWALK can consistently outperform the base model. For two models, when varying embedding size from 2 to 512, the classification accuracy firstly increases in a relatively fast speed, then grows slowly, and finally becomes stable or even drops slightly. The reason is that model generalization ability is improved with the increase of model capacity firstly until some threshold, since more network structural information can be captured with larger model capacity. However, when the model capacity becomes too large, it can easily result in overfitting, and thus cause performance degradation. We notice that the performance improvement of DWNS_AdvT over DWNS is quite small when the embedding size is 2. It is probably because model capacity is the main reason limiting model performance and model robustness is not a serious issue when embedding size is too small.

4.3 Link Prediction

Link prediction is essential for many applications such as extracting missing information and identifying spurious interaction [20]. In this section, we conduct link prediction on five real-world networks,

and compare our proposed methods with the state-of-the-art methods. The experimental settings have been illustrated in Section 4.2. Table 2 summarizes the experimental results.

It can be easily observed that both our proposed methods, including Dwns_AdvT and Dwns_iAdvT, performs better than Dwns in all five datasets, which demonstrates that two types of adversarial regularization methods can help improve model generalization ability. Specifically, there is a 4.62% performance improvement for Dwns_AdvT over Dwns on average across all datasets, and that for Dwns_iAdvT is 4.60%, which are very impressive.

We noticed that AIDW has a poor performance in link prediction. The reasons can be two-folds: firstly, AIDW encourages the smoothness of embedding distribution from a global perspective by imposing a prior distribution on them, which can result in over-regularization and thus cause performance degradation; secondly, AIDW suffers from mode-collapse problem because of its generative adversarial network component, which can also result in model corruption. Besides, Dwns_rand has similar performance with Dwns, which means that the regularization term with random perturbation contributes little to model generalization ability. By comparison, our proposed novel adversarial training regularization method is more stable and effective.

It can be observed that the performance of Dwns_AdvT and Dwns_iAdvT are comparable. Either Dwns_AdvT or Dwns_iAdvT achieves the best results across the five datasets, which shows the remarkable usefulness of the proposed regularization methods. For Cora and CA-GrQc, Dwns_iAdvT has better performance, although we restrict the perturbation directions toward the nearest neighbors of the considered node. It suggests that such restriction of perturbation directions might provide useful information for representation learning.

4.4 Node Classification

Node classification can be conducted to dig out missing information in a network. In this section, we conduct multi-class classification on three benchmark datasets, including Cora, Citeseer and Wiki, with the training ratio ranging from 1% to 90%. Tables 3, 4 and 5 summarize the experimental results.

Firstly, Dwns_rand and Dwns have similar performance in all three datasets. For example, the average improvement of Dwns_rand over Dwns is 0.16% across all training ratios in Wiki, which can be negligible. It validates that random perturbation for the regularization term contributes little to the model generalization performance again. It is understandable, since the expected dot product between any reference vector and the random perturbation from a zero mean gaussian distribution is zero, and thus the regularization term will barely affect the embedding learning.

Secondly, Dwns_AdvT and Dwns_iAdvT consistently outperform Dwns across all different training ratios in the three datasets, with the only exception of Dwns_iAdvT in Citeseer when the training ratio is 3%. Specifically, Dwns_AdvT achieves 5.06%, 6.45% and 5.21% performance gain over Dwns on average across all training ratios in Cora, Citeseer and Wiki respectively, while the improvement over Dwns for Dwns_iAdvT are 2.35%, 4.50% and 2.62% respectively. It validates that adversarial perturbation can provide useful direction for generating adversarial examples, and thus brings significant improvements to model generalization ability after the adversarial training process. For Dwns_iAdvT, it brings less performance gain compared with Dwns_AdvT, which might because the restriction on perturbation direction limit its regularization ability for classification tasks. In this case, there is a tradeoff between interpretability and regularization effect.

Thirdly, AIDW achieves better results than DeepWalk, LINE and GraRep, which shows that global regularization on embedding vectors through adversarial learning can help improve model generalization performance. Our proposed methods, especially Dwns_AdvT, demonstrate superiority over all the state-of-the-art baselines, including AIDW and node2vec, based on experimental results comparison. We can summarize that the adversarial training regularization method has advantages over the GAN-based global regularization methods in three aspects, including more succinct architecture, better computational efficiency and more effective performance contribution.

4.5 Parameter Sensitivity

We conduct parameter sensitivity analysis with link prediction and multi-class classification on Cora, Citeseer and Wiki in this section. Here we only present the results for Dwns_AdvT due to space limitation. Adversarial training regularization method is very succinct. Dwns_AdvT only has two more hyperparameters compared with Dwns, which are noise level ϵ and adversarial regularization strength λ. Note that when studying one hyper-parameter, we follow default settings for other hyper-parameters. The experimental settings of link prediction and node classification have been explained in Section 4.2.

Fig. 5(a) presents the experimental results when varying ϵ from 0.1 to 5.0. For both learning tasks, we can find that the performance in these three datasets first improves with the increase of ϵ, and then drops dramatically after ϵ passing some threshold. It suggests that appropriate setting of ϵ improves the model robustness and generalization ability, while adversarial perturbation with too large norm constraint can destroy the learning process of embedding vectors. Besides, it can be easily noticed that the best settings of ϵ are different for different datasets in general. Specifically, Citeseer has the best results in both link prediction and node classification when $\epsilon = 1.1$, Cora achieves the best results when $\epsilon = 0.9$, while the best setting of ϵ for Wiki is around 0.5. Based on the experimental results on these three datasets only, it seems that the denser the network is, the smaller the best noise level parameter ϵ should be.

We conduct link prediction and node classification on three datasets with the adversarial regularization strength λ from the set $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$. Fig. 5(b) displays the experimental results. For node classification, the best result is obtained when λ is set to around 1, larger values can result in performance degradation. For example, the classification accuracy on Wiki drops dramatically when λ reaches 10, and larger setting produces worse results. For link prediction, the performance is quite consistent among the three datasets. Specifically, when λ increases from 0.001 to 10, the AUC score shows apparent increase for all datasets, and then tends to saturate or decrease slightly. Empirically, 1 is an appropriate value for the adversarial regularization strength λ.

Table 2: AUC score for link prediction

Dataset	Cora	Citeseer	Wiki	CA-GrQc	CA-HepTh
GF	0.550 ± 0.005	0.550 ± 0.002	0.584 ± 0.007	0.593 ± 0.003	0.554 ± 0.001
DeepWalk	0.620 ± 0.003	0.621 ± 0.002	0.658 ± 0.002	0.694 ± 0.001	0.683 ± 0.000
LINE	0.626 ± 0.011	0.625 ± 0.004	0.647 ± 0.010	0.641 ± 0.002	0.629 ± 0.005
node2vec	0.626 ± 0.023	0.627 ± 0.022	0.639 ± 0.010	0.695 ± 0.006	0.667 ± 0.009
GraRep	0.609 ± 0.035	0.589 ± 0.025	0.642 ± 0.045	0.500 ± 0.000	0.500 ± 0.000
AIDW	0.552 ± 0.034	0.606 ± 0.035	0.511 ± 0.019	0.615 ± 0.023	0.592 ± 0.019
Dwns	0.609 ± 0.018	0.609 ± 0.011	0.648 ± 0.007	0.690 ± 0.004	0.662 ± 0.006
Dwns_rand	0.606 ± 0.012	0.608 ± 0.005	0.645 ± 0.010	0.696 ± 0.006	0.662 ± 0.003
Dwns_AdvT	0.644 ± 0.009	**0.656 ± 0.007**	**0.665 ± 0.005**	**0.707 ± 0.004**	**0.692 ± 0.003**
Dwns_iAdvT	**0.655 ± 0.015**	0.653 ± 0.006	0.660 ± 0.002	**0.707 ± 0.004**	0.688 ± 0.004

Table 3: Accuracy (%) of multi-class classification on Cora

%Ratio	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%	20%	30%	40%	50%	60%	70%	80%	90%
GF	24.55	28.87	32.07	33.11	34.45	35.83	38.25	39.05	39.84	39.42	46.14	48.57	50.09	50.85	51.88	52.89	52.34	51.51
DeepWalk	44.63	49.30	52.25	53.05	55.21	59.10	59.26	62.20	63.07	64.60	69.85	74.21	76.68	77.59	77.68	78.63	79.35	79.23
LINE	38.78	49.62	54.51	56.49	58.99	61.30	63.05	64.19	66.59	66.06	70.86	72.25	73.94	74.03	74.65	75.12	75.30	75.76
node2vec	58.02	63.98	66.33	68.07	69.91	69.87	71.41	72.60	73.63	73.96	78.04	80.07	81.62	82.16	82.25	82.85	84.02	84.91
GraRep	54.24	63.58	65.36	68.78	70.67	72.69	72.37	72.70	73.53	74.98	77.48	78.57	79.38	79.53	79.68	79.75	80.89	80.74
AIDW	54.55	63.30	65.86	66.20	67.62	68.61	69.52	71.07	71.44	73.83	77.93	79.43	81.16	81.79	82.27	82.93	84.11	83.69
Dwns	57.72	64.82	67.93	68.50	68.27	70.81	70.72	72.30	72.00	73.20	76.98	79.83	80.56	82.27	82.52	82.92	82.97	84.54
Dwns_rand	56.46	64.87	67.44	68.24	70.38	71.16	71.34	72.67	73.51	73.45	78.04	79.76	81.66	81.72	82.53	83.57	83.51	83.69
Dwns_AdvT	**62.66**	**68.46**	69.91	**73.62**	**74.71**	**75.55**	**76.18**	**76.77**	**77.72**	**77.73**	**80.50**	**82.33**	**83.54**	**83.63**	**84.41**	**84.99**	**85.66**	**85.65**
Dwns_iAdvT	58.67	66.65	**70.17**	70.52	71.42	72.47	74.26	75.32	74.52	76.12	78.88	80.31	81.61	82.80	83.03	83.63	83.75	85.02

Table 4: Accuracy (%) of multi-class classification on Citeseer

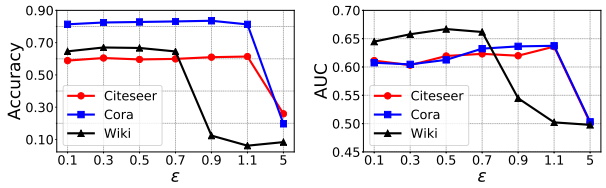
%Ratio	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%	20%	30%	40%	50%	60%	70%	80%	90%
GF	22.63	24.49	25.76	28.21	28.07	29.02	30.20	30.70	31.20	31.48	34.05	35.69	36.26	37.18	37.87	38.85	39.16	39.54
DeepWalk	27.82	32.44	35.47	36.85	39.10	41.01	41.56	42.81	45.35	45.53	50.98	53.79	55.25	56.05	56.84	57.36	58.15	59.11
LINE	29.98	34.91	37.02	40.51	41.63	42.48	43.65	44.25	45.65	47.03	50.09	52.71	53.52	54.20	55.42	55.87	55.93	57.22
node2vec	36.56	40.21	44.14	45.71	46.32	47.47	49.56	49.78	50.73	50.78	55.89	57.93	58.60	59.44	59.97	60.32	60.75	61.04
GraRep	37.98	40.72	43.33	45.56	47.48	47.93	49.54	49.87	50.65	50.60	53.56	54.63	55.44	55.20	55.07	56.04	55.48	56.39
AIDW	38.77	42.84	44.04	44.27	46.29	47.89	47.73	49.61	49.55	50.77	54.82	56.96	58.04	59.65	60.03	60.99	61.18	62.84
Dwns	38.13	42.88	46.60	46.14	46.38	48.18	48.58	48.35	50.16	50.00	53.74	57.37	58.59	59.00	59.53	59.62	59.51	60.18
Dwns_rand	39.29	43.42	42.73	46.00	46.13	48.69	48.15	49.92	50.08	50.84	55.26	58.51	59.59	59.12	60.22	60.62	61.59	60.55
Dwns_AdvT	**41.33**	45.00	**46.73**	**48.57**	**50.37**	**51.06**	**52.07**	**53.09**	**53.73**	**54.79**	**59.21**	**61.06**	**61.26**	**62.56**	**62.63**	**62.40**	**63.05**	**63.73**
Dwns_iAdvT	40.88	**45.53**	46.01	47.10	50.02	50.79	49.59	52.78	51.95	52.26	56.65	59.07	60.27	61.96	62.04	62.20	62.21	63.15

Table 5: Accuracy (%) of multi-class classification on Wiki

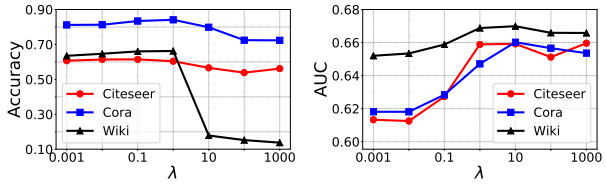
%Ratio	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%	20%	30%	40%	50%	60%	70%	80%	90%
GF	19.76	22.70	27.00	28.41	30.28	31.49	31.87	32.18	34.16	34.25	36.13	37.66	37.43	39.48	40.17	39.83	40.25	41.01
DeepWalk	28.65	32.84	36.66	37.98	40.73	42.94	45.57	45.47	46.06	46.60	54.48	59.05	62.70	64.66	65.95	66.98	68.37	68.78
LINE	32.46	40.84	44.56	49.59	51.11	52.37	54.32	55.72	56.51	57.88	61.08	63.50	64.68	66.29	66.91	67.43	67.46	68.61
node2vec	32.41	41.96	47.32	48.15	50.65	51.08	52.71	54.66	54.81	55.94	59.67	61.11	64.21	65.08	65.58	66.76	67.19	68.73
GraRep	33.38	45.61	49.10	50.92	53.01	54.43	54.84	57.50	57.01	58.57	61.91	63.58	63.77	64.68	65.39	65.92	65.18	67.05
AIDW	35.17	43.05	46.63	51.29	52.40	52.72	55.92	56.78	55.92	57.32	61.84	63.54	64.90	65.58	66.54	65.59	66.58	68.02
Dwns	35.76	42.71	48.08	50.01	50.21	52.26	53.26	53.80	55.27	55.77	59.63	61.98	64.01	64.59	66.99	66.45	67.55	67.51
Dwns_rand	36.12	44.57	46.71	49.15	51.74	53.37	53.22	53.27	54.21	56.33	59.41	61.94	64.07	65.17	66.18	65.64	68.20	67.34
Dwns_AdvT	**38.42**	**45.80**	**50.21**	51.12	**54.29**	**56.43**	**57.12**	**57.82**	**58.60**	**59.97**	**63.33**	**65.32**	**66.53**	**67.06**	**67.69**	**68.94**	**68.35**	**69.32**
Dwns_iAdvT	37.46	45.11	49.14	**51.57**	51.88	54.43	55.42	56.05	55.93	57.81	61.40	63.37	65.71	65.56	67.09	66.81	67.70	68.02

5 RELATED WORK

Network Embedding. Some early methods, such as IsoMap [36] and LLE [28], assume the existence of a manifold structure on input vectors to compute low-dimensional embeddings, but suffer from the expensive computation and their inability in capturing highly non-linear structural information of networks. More recently, some negative sampling approach based models have been proposed, including DeepWalk [27], LINE [35] and node2vec [13], which enjoys two attractive strengths: firstly, they can effectively capture high-order proximities of networks; secondly, they can scale to the widely existed large networks. DeepWalk obtains node sequences with truncated random walk, and learns node embeddings with Skip-gram model [22] by regarding node sequences as sentences. node2vec differs from DeepWalk by proposing more flexible random walk method for sampling node sequences. LINE defines



(a) Noise level ϵ.



(b) Adversarial regularization strength λ.

Figure 5: Impact of hyperparameters on node classification (left, training ratio 50%) and link prediction (right).

first-order and second-order proximities in network, and resorts to negative sampling for capturing them.

Further, some works [4, 26, 39] tried to preserve various network structural properties in embedding vectors based on matrix factorization technique. GraRep [4] can preserve different k-step proximities between nodes independently, HOPE [26] aims to capture asymmetric transitivity property in node embeddings, while N-NMF [39] learns community structure preserving embedding vectors by building upon the modularity based community detection model [24]. Meanwhile, deep learning embedding models [5, 32, 33, 37] have also been proposed to capture highly nonlinear structure. DNGR [5] takes advantages of deep denoising autoencoder for learning compact node embeddings, which can also improve model robustness. SDNE [37] modifies the framework of stacked autoencoder to learn both first-order and second-order proximities simultaneously. DNE-SBP [33] utilizes a semi-supervised SAE to preserve the structural balance property of the signed networks. Both GraphGAN [38] and A-RNE [9] leverage generative adversarial networks to facilitate network embedding, with the former unifies the generative models and discriminative models of network embedding to boost the performance while the latter focuses on sampling high-quality negative nodes to achieve better similariy ranking among node pairs.

However, the above mentioned models mainly focus on learning different network structures and properties, while neglecting the existence of noisy information in real-world networks and the overfitting issue in embedding learning process. Most recently, some methods, including ANE [8] and NetRA [41], try to regularize the embedding learning process for improving model robustness and generalization ability based on generative adversarial networks (GANs). They have very complicated frameworks and suffer from the well-recognized hard training problems of GANs. Furthermore, these two methods both encourage the global smoothness of the embedding distribution, while in this paper we utilize a more succinct and effective local regularization method.

Adversarial Machine Learning. It was found that several machine learning models, including both deep neural network and shallow classifiers such as logistic regression, are vulnerable to examples with imperceptibly small designed perturbations, called adversarial examples [12, 34]. This phenomenon was firstly observed in areas like computer vision with continuous input vectors. To improve model robustness and generalization ability, adversarial training method [12] is shown to be effective. It generates adversarial perturbations for original clean input with the aim of maximizing current model loss, and further approximates the difficult optimization objective with first-order Taylor Series. Such method has also been applied to text classification problem in [23, 30] by defining the perturbation on continuous word embeddings, and recommendation in [15] by generating adversarial perturbations on model parameters. However, to the best of our knowledge, there is no practice of adversarial training regularization for graph representation learning.

For graph structured data, they are fundamentally different from images because of their discrete and indifferentiable characteristics. Some existing works [6, 7, 42] aimed to explore how to generate the adversarial examples in the discrete, binary graph domain, and whether similar vulnerability exists in graph analysis applications. In [7], adversarial attacks are generated by modifying combinatorial structure of graph with a reinforcement learning based method, which is shown to be effective in Graph Neural Network models. Both [42] and [6] designed attack methods to Graph Convolutional Network [16]. Particularly, NETTACK [42] focuses on attributed graph classification problem and FGA [6] tackles network representation learning. However, all of them studied adversarial attack methods without providing any defense algorithms for improving the robustness of existing methods against these attacks. Differently, in this paper, we aim to propose adversarial regularization method for network embedding algorithms to improve both model robustness and generalization ability.

6 CONCLUSION

In this paper, we proposed two adversarial training regularization methods for network embedding models to improve the robustness and generalization ability. Specifically, the first method is adapted from the classic adversarial training method by defining the perturbation in the embedding space with adaptive L_2 norm constraint. Though it is effective as a regularizer, the lack of interpretability may hinder its adoption in some real-world applications. To tackle this problem, we further proposed an interpretable adversarial training method by restricting the perturbation directions to embedding vectors of other nodes, such that the crafted adversarial examples can be reconstructed in the discrete graph domain. Both methods can be applied to the existing embedding models with node embeddings as model parameters, and DeepWalk is used as the base model in the paper for illustration. Extensive experiments prove the effectiveness of the proposed adversarial regularization methods for improving model robustness and generalization ability. Future works would include applying adversarial training method to the parameterized network embedding methods such as deep learning embedding models.

7 ACKNOWLEDGMENTS

Parts of the work were supported by HK ITF UIM/363.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-Scale Machine Learning. In *OSDI*. USENIX Association, 265–283.

[2] Amr Ahmed, Nino Shervashidze, Shravan M. Narayanamurthy, Vanja Josifovski, and Alexander J. Smola. 2013. Distributed large-scale natural graph factorization. In *WWW*. 37–48.

[3] Martín Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein Generative Adversarial Networks. In *ICML*. 214–223.

[4] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning Graph Representations with Global Structural Information. In *CIKM*. 891–900.

[5] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep Neural Networks for Learning Graph Representations. In *AAAI*. 1145–1152.

[6] Jinyin Chen, Yangyang Wu, Xuanheng Xu, Yixian Chen, Haibin Zheng, and Qi Xuan. 2018. Fast Gradient Attack on Network Embedding. *CoRR* abs/1809.02797 (2018).

[7] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. 2018. Adversarial Attack on Graph Structured Data. In *ICML (JMLR Workshop and Conference Proceedings)*, Vol. 80. 1123–1132.

[8] Quanyu Dai, Qiang Li, Jian Tang, and Dan Wang. 2018. Adversarial Network Embedding. In *AAAI*.

[9] Quanyu Dai, Qiang Li, Liang Zhang, and Dan Wang. 2019. Ranking Network Embedding via Adversarial Learning. In *PAKDD*.

[10] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *JMLR* 9 (2008), 1871–1874.

[11] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *NIPS*. 2672–2680.

[12] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and Harnessing Adversarial Examples. *CoRR* abs/1412.6572 (2014).

[13] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *KDD*. 855–864.

[14] Yupeng Gu, Yizhou Sun, Yanen Li, and Yang Yang. 2018. RaRE: Social Rank Regulated Large-scale Network Embedding. In *WWW*. ACM, 359–368.

[15] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial Personalized Ranking for Recommendation. In *SIGIR*. ACM, 355–364.

[16] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[17] Jure Leskovec, Jon M. Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *TKDD* 1, 1 (2007), 2.

[18] Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding as Implicit Matrix Factorization. In *NIPS*. 2177–2185.

[19] Aaron Q. Li, Amr Ahmed, Sujith Ravi, and Alexander J. Smola. 2014. Reducing the sampling complexity of topic models. In *KDD*. 891–900.

[20] Linyuan Lv and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications* 390, 6 (2011), 1150 – 1170.

[21] Andrew McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 2000. Automating the Construction of Internet Portals with Machine Learning. *Inf. Retr.* 3, 2 (2000), 127–163.

[22] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*. 3111–3119.

[23] Takeru Miyato, Andrew M. Dai, and Ian Goodfellow. 2017. Adversarial Training Methods for Semi-Supervised Text Classification. In *ICLR*.

[24] M. E. J. Newman. 2006. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E* 74 (Sep 2006), 036104. Issue 3.

[25] Feng Niu, Benjamin Recht, Christopher Ré, and Stephen J. Wright. 2011. HOGWILD!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent. *Advances in Neural Information Processing Systems* (2011), 693–701.

[26] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric Transitivity Preserving Graph Embedding. In *KDD*. 1105–1114.

[27] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *KDD*. 701–710.

[28] Sam T. Roweis and Lawrence K. Saul. 2000. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science* 290 (2000), 2323–2326. Issue 5500.

[29] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved Techniques for Training GANs. In *NIPS*. 2226–2234.

[30] Motoki Sato, Jun Suzuki, Hiroyuki Shindo, and Yuji Matsumoto. 2018. Interpretable Adversarial Perturbation in Input Embedding Space for Text. In *IJCAI*. 4323–4330.

[31] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective Classification in Network Data. *AI Magazing* 29, 3 (2008), 93–106.

[32] Xiao Shen and Fu-Lai Chung. 2017. Deep Network Embedding with Aggregated Proximity Preserving. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. 40–43.

[33] Xiao Shen and Fu-Lai Chung. 2018. Deep Network Embedding for Graph Representation Learning in Signed Networks. *IEEE Transactions on Cybernetics* (2018).

[34] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *ICLR*.

[35] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW*. 1067–1077.

[36] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. 2000. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science* 290 (2000), 2319–2323. Issue 5500.

[37] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural Deep Network Embedding. In *KDD*. 1225–1234.

[38] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. GraphGAN: Graph Representation Learning with Generative Adversarial Nets. In *AAAI*.

[39] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community Preserving Network Embedding. In *AAAI*. 203–209.

[40] Zhitao Wang, Chengyao Chen, and Wenjie Li. 2017. Predictive Network Representation Learning for Link Prediction. In *SIGIR*. 969–972.

[41] Wenchao Yu, Cheng Zheng, Wei Cheng, Charu C. Aggarwal, Dongjin Song, Bo Zong, Haifeng Chen, and Wei Wang. 2018. Learning Deep Network Representations with Adversarially Regularized Autoencoders. In *KDD*. 2663–2671.

[42] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. 2018. Adversarial Attacks on Neural Networks for Graph Data. In *KDD*. 2847–2856.

Local Matching Networks for Engineering Diagram Search

Zhuyun Dai *
Carnegie Mellon University
zhuyund@cs.cmu.edu

Zhen Fan *
Tsinghua University
fanz15@mails.tsinghua.edu.cn

Hafeezul Rahman
Carnegie Mellon University
hmohamma@andrew.cmu.edu

Jamie Callan
Carnegie Mellon University
callan@cs.cmu.edu

ABSTRACT

Finding diagrams that contain a specific part or a similar part is important in many engineering tasks. In this search task, the query part is expected to match only a small region in a complex image. This paper investigates several local matching networks that explicitly model local region-to-region similarities. Deep convolutional neural networks extract local features and model local matching patterns. Spatial convolution is employed to cross-match local regions at different scale levels, addressing cases where the target part appears at a different scale, position, and/or angle. A gating network automatically learns region importance, removing noise from sparse areas and visual metadata in engineering diagrams.

Experimental results show that local matching approaches are more effective for engineering diagram search than global matching approaches. Suppressing unimportant regions via the gating network enhances accuracy. Matching across different scales via spatial convolution substantially improves robustness to scale and rotation changes. A pipelined architecture efficiently searches a large collection of diagrams by using a simple local matching network to identify a small set of candidate images and a more sophisticated network with convolutional cross-scale matching to re-rank candidates.

KEYWORDS

Diagram Search, Neural IR, Image Retrieval

ACM Reference Format:
Zhuyun Dai, Zhen Fan *, Hafeezul Rahman, and Jamie Callan. 2019. Local Matching Networks for Engineering Diagram Search. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313500

1 INTRODUCTION

In large engineering organizations that manufacture complex products, many engineers at different locations may collaborate to design a product. Searching a corpus of engineering diagrams for similar parts helps staff estimate production costs for new parts based on the costs of similar existing parts; and reduce production costs by using existing parts or replacing several slightly different parts with a new, more widely applicable part.

Engineering diagrams use line drawings and metadata such as part reference numbers to represent the elements of a part or an assembly of parts. Often they are created by computer-aided design (CAD) systems. However, the detailed descriptions stored in CAD systems are unwieldy, often unnecessary, and sometimes too sensitive to be shared widely within an organization. Thus, engineering diagrams – *images* – are the main method of describing parts and assemblies for many tasks. Finding engineering diagrams that contain a specific part or assembly is a type of *image retrieval*.

Image retrieval is a long-studied research area. Most prior research focuses on pictures, and specifically how to represent pictures so that similar pictures are easy to recognize. The recent advances achieved by convolutional neural networks (CNN) create opportunities for generating competitive image representations. Several works investigate taking the output of the network, or aggregating intermediate convolution features, to create an image feature vector that can be used for retrieval via k-nearest neighbour search [1, 2, 7, 11, 25]. Most existing CNN-based image search approaches use a single representation of the entire image; we call this a *global* representation. Global representations have many advantages: they are fast, suppress or generalize unnecessary detail, and describe the main elements of an image. However, they also lose detail, which can be a problem for engineering diagram search.

Engineering diagram search has specific characteristics. Often, the query diagram is expected to match only a small region of an engineering diagram, for example, to find every diagram that uses a specific part. Thus, it is a more *local* matching problem. Engineering diagrams differ from most pictures. Typical engineering diagrams are gray-scale line drawings that have little texture. They have a visual syntax and contain visual metadata elements, for example, locator elements that place the diagram within a larger context; callouts that provide other perspectives; bubbles that provide additional detail; and lines that illustrate relationships among selected parts. It is an open question whether existing image retrieval algorithms based on global, general visual features are effective for engineering diagram search. This paper investigates several global and local approaches for engineering diagram search.

One of our new approaches is an unsupervised ranker called **DI**agram **S**earc**H** with Local Histogram Pooling Matching Network (DISH-HP). DISH-HP models the relevance of a diagram to a query based on the similarity of local regions, using histogram pooling to combine evidence [8]. It first extracts local convolution features from the middle layers of a pretrained network, such as VGG-16

*The first two authors contributed equally.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313500

[26]. Each local feature represents a small region in the original diagram. DISH-HP calculates the similarity scores between every pair of query and document regions. It then groups the region pairs into bins according to their similarity scores. Two diagrams are considered similar if many of the region pairs fall into the high similarity bins. The histogram-based method is efficient so that DISH-HP can scan over all diagrams in a collection to find the most similar candidates.

The second of our new approaches is called Diagram Search with Local Convolutional Matching Network (DISH-Conv). It was inspired by Conv-KNRM, a state-of-the-art neural text ranking model, that uses convolutional neural networks to match text n-grams [3]. The new DISH-Conv model employs spatial convolutional neural networks to build feature vectors for larger regions from small regions, generating multi-scale region representations. A cross-scale matching layer calculates the similarity between regions of different scales, enabling a smaller region in the query to match to a larger region in the candidate diagram. A *gating* network is proposed that automatically detects unimportant regions of a diagram, reducing noisy signals from low-density regions and visual metadata such as callouts. DISH-Conv can be trained end-to-end with relevance signals, so that the convolution, matching, and gating are optimized towards the specific dataset and task.

DISH-HP and DISH-Conv can be used in a pipelined re-ranking architecture. DISH-HP is the first stage that quickly ranks images. The top N candidates are passed to DISH-Conv to refine the ranking. N is limited to a few hundred or a few thousand, so that the second stage is able to use a more accurate ranking model that would be too computationally expensive for the first stage brute-force search.

Evaluation was done on a new diagram search dataset developed by crawling furniture assembly manuals from the Ikea™ website. Queries were generated automatically by extracting query-like regions from diagrams and transforming them in various ways, for example, by changing position, scale, and/or rotation. Experiments on large-scale auto-generated relevance labels show that DISH-HP is more accurate than several unsupervised rankers that use global image representations. Using DISH-Conv to re-rank DISH-HP results improves accuracy substantially. Larger improvements over baseline models are shown on more difficult queries that have rotation and scale changes. Evaluation with a small set of manual relevance assessments further confirms these results. Analysis reveals that the advantage of DISH-Conv is a combination of its abilities to transform rotation changes through the convolutions, match at multiple scale level through the cross-matching layer, and suppress unimportant regions through the gating network.

The rest of this paper is organized as follows. Sections 2 gives a brief overview of related work. Section 3 and 4 introduce the proposed models DISH-HP and DISH-Conv. Section 5 discusses the Ikea diagram search dataset and our experimental methodology. Section 6 reports and discusses experimental results. Section 7 summarizes and concludes the paper.

2 RELATED WORK

Image Search. Until recently, the vast majority of image search approaches were variants of the bag-of-words model [24] based on hand-crafted features, typically SIFT [13]. In the recent decade, out-of-the-box features from pre-trained Convolutional Neural Networks (CNNs) were shown to give state-of-the-art results in many computer vision tasks, including image search. The early CNN-based image retrieval work [2, 7] takes the activation of fully connected layers as *global* descriptors. Later works shift the focus to convolution layers because lower layers preserve more detailed visual information and are more suitable for the search task. The convolution features are extracted from sliding windows at different image regions, hence are usually considered as *local* features. Most prior work used max/sum pooling over the convolution features to produce the single image representation Babenko and Lempitsky [1], Jimenez et al. [11], Kalantidis et al. [12], Tolias et al. [25]. Although local convolution features are used, they are pooled into a single representation, hence the image matching is still performed *globally*. It may raise problems for diagram search where tasks include matching *part* of one diagram to *part* of another diagram.

Patch-based approaches aim to represent a image with multiple image patches to address the problem when the item of interest has appeared on a different scale at an arbitrary position in the relevant image [7, 21]. Patch-based methods are computationally expensive because each individual patch is resized to the same size as the original image and fed forward into a deep convolutional neural network to extract features.

Most CNN-based image retrieval work focused on real-world photographs [1, 2, 11, 25]. They were tested against benchmark datasets consisting of landmark photographs, such as the Oxford Buildings dataset [18], the Paris Dataset [19], and the INRIA Holidays dataset [10]. The effectiveness of CNN based features on engineering diagrams remains to be explored,

Engineering Diagram Search. A lot of work on image-based engineering diagram search were developed before the bag-of-word model [24] was widely applied in image search. These systems often take several steps to first identify lines and shapes and then to calculate similarities between two diagrams. Task specific features and matching criterias were designed, leading to complex systems and extensive feature engineering [6, 15, 20]. A more recent work by Eitz et al. [5] published in 2011 demonstrates the effectiveness of bag-of-words models with SIFT-style features on line-drawings and sketch-based image retrieval. To the best of our knowledge, little exploration has been made on using deep convolution networks for engineering diagram search.

Neural Ranking Models for Text Search. Large progress has been made recently on using deep neural networks for text search. Recent neural ranking models can be categorized as *representation-based* and *interaction-based* [8]. Representation-based models use global representations of the query and document and match in the representation space [9, 22]. They are similar to most image search approaches based on global image representations. Interaction-based models use local interactions between the query and document words and neural networks that learn matching patterns [3, 8, 27]. The DRMM model [8] uses histogram pooling to summarize word-word interactions. It builds a word-word similarity matrix from word embeddings, groups the word-word soft matches into a histogram based on the similarity score, and learns to combine the bins. It allows the model to take into consideration both strong word matches and weaker word matches. The paper also proposed two methods to suppress unimportant query words; one method is

based on term frequency signals and the other is based on the word embedding. The K-NRM model [27] integrates DRMM with the ability to learn customized embeddings. It uses kernel-pooling, an approximation of the histogram with RBF kernels. Kernel-pooling shares the advantage of histogram pooling while also being differentiable so that word embeddings can be learned end-to-end and align with the search task. The Conv-KNRM model [3] extends K-NRM by using convolutional neural networks to build n-gram representations. It uses different sizes of convolutions to build unigram, bigram and trigram embeddings from word embeddings. A *cross-matching* layer compares across n-grams of various lengths in the unified embedding space. It allows the query bi-gram "deep learning" to be matched to the document tri-gram "convolutional neural network". The n-gram matching signals improve ranking accuracy over the unigram models.

3 DISH-HP: LOCAL HISTOGRAM POOLING MATCHING NETWORK

Interaction-based text retrieval models, which match a query to localized regions of a document, are more accurate than representation-based models because i) text queries are not expected to match all, or even most, of a text document, ii) local interactions provide multiple and detailed match signals, and iii) irrelevant regions have no effect. Interaction-based models could be beneficial to diagram search by modeling the interactions of a local region from the query diagram and a local region from the document diagram.

The section describes a diagram search model based on *Local Histogram Pooling Matching Network* (**DISH-HP**). DISH-HP is inspired by the histogram pooling technique proposed by Guo et al. [8]. It models the relevance between two diagrams based on how similar their local regions are.

The input to DISH-HP is a list of local convolution feature vectors extracted from the query diagram Q and the document diagram D. In our experiments, features are extracted from the last convolution layer of the VGG-16 network [26]. This layer divides the image into $R = W \times H = 14 \times 14 = 196$ grids, each corresponding to a 16 pixel $\times$ 16 pixel rectangular region in the diagram. For each of the R regions, a $d = 512$ dimension feature vector is generated. Hence, Q and D are represented by matrices of size $[R, d] = [196, 512]$.

Given the feature matrices of Q and D, DISH-HP collects the similarity scores between every pair of query region and document region. It then groups the region pairs into bins based on the similarity scores and builds a histogram. The histogram is pooled to produce a relevance score, where two diagrams are considered similar if many of the region pairs fall into the high similarity bins. More specifically, DISH-HP builds a similarity matrix M of size $[R, R]$ where the cell $M_{i,j}$ is the cosine similarity between the i-th query feature and the j-th document feature. A histogram is built for each row of the similarity matrix by assigning the similarity values into 10 bins: $B_1 = [0.8, 1]$, $B_2 = [0.6, 0.8)$, ..., $B_{10} = [-1, -0.8)]$. DISH-HP then counts the points in each bin. The counts indicate how many document regions are very similar to the query region (e.g., bin $[0.8, 1]$), how many are somewhat similar (e.g., bin $[0.6, 0.8)$), and how many are dissimilar. The number of points in the first K high-similarity bins $\{B_1, ..., B_K\}$ are taken as the score for the document diagram to match the i-th query region:

$$s(Q_i, D) = \sum_{j=1...K} |B_j|. \quad (1)$$

The final relevance score is the log-sum of counts from every query region:

$$s(Q, D) = \sum_{i=1...R} \log s(Q_i, D). \quad (2)$$

Log-sum squashes very high values into smaller values. It helps to avoid the matching being dominated by a few query regions that are repeatedly matched in the document diagram.

One issue with the above method is that the relevance score is often overwhelmed by noisy match signals from low-density regions. Engineering diagrams are often very sparse; blank or near-blank regions will dominate the high-similarity bins. To address this issue, features from low-density regions are filtered out before computing the similarity matrix. A feature vector is removed if its L2-norm is smaller than a threshold T, with the assumption that lower L2-norm indicates lower density of that region. The L2-norm threshold T is a hyper-parameter to be tuned.

In summary, DISH-HP explicitly models the similarity between local regions from the query and the document. DISH-HP is expected to retrieve more-relevant documents than global matching methods because the local, region-level matching provides more evidence than matching at the global image level. In terms of efficiency, DISH-HP is slightly slower than global methods. DISH-HP needs to compute R^2 cosine similarity values for a document with R local regions; global methods only need to compute a single global cosine similarity. The bottleneck for both global and local method lies in extraction features from the query using a deep convolutions neural network.

4 DISH-CONV: LOCAL CONVOLUTIONAL MATCHING NETWORK

DISH-HP is designed to be simple and efficient in order to be used for initial ranking. However, the simple architecture has several drawbacks that may limit its search effectiveness. First, DISH-HP uses several heuristics, such as which bins are used and how low-density areas are detected. These heuristics may not be optimal. More importantly, the local features are only for a fixed size of image regions (e.g. 16 pixel $\times$ 16 pixel), while in many cases the same part may be drawn at different scales in different diagrams.

This section proposes DISH-Conv, a supervised neural diagram search model based on *Local Convolutional Matching Network* (**DISH-Conv**). The drawbacks of DISH-HP are addressed by using machine-learned weights to replace heuristics and using better network architectures to generate richer features.

DISH-Conv is inspired by Conv-KNRM, a state-of-the-art neural ranking model proposed by Dai et al. [3]. It adapts the use of convolutions and cross-matching from text search to diagram search. The architecture of DISH-Conv is shown in Figure 1. Given local convolution features from a query and a document diagram, it builds multi-scale region representations through spatial convolutions, mutes unimportant region features through a gating network,

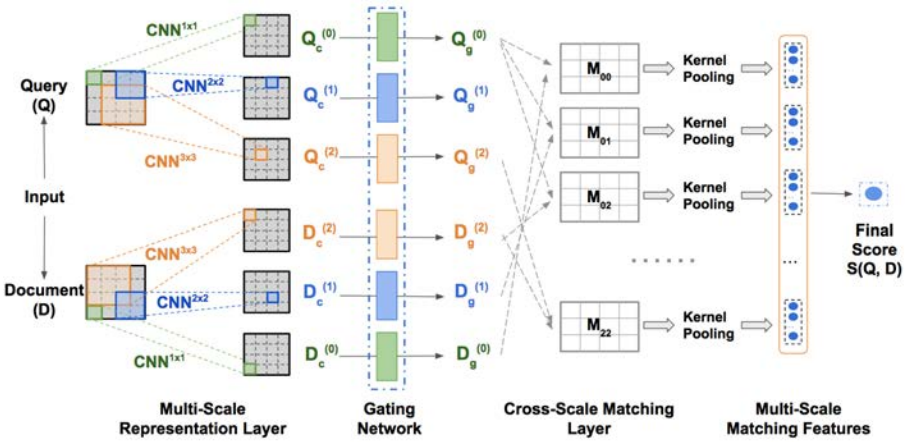


Figure 1: The DISH-Conv architecture. Query and document diagrams are input. The multi-scale representation layer uses spatial convolutions of different kernel sizes to generate features at multiple scales. The gating network computes the importance of each feature, and suppresses features from unimportant areas. The cross-scale matching layer compares the similarity between query regions and document diagram regions of different sizes, and forms the similarity matrices for different scales. The kernel pooling aggregates the similarity matrices into cross-scale matching features. A *learning-to-rank* layer combines all features and generate a final score.

and generates cross-scale local match signals through the cross-matching layer. It then ranks diagrams with the cross-scale match signals using kernel-pooling and learning-to-rank.

4.1 Generating Multi-Scale Representations

DISH-Conv starts with the same process as DISH-HP. Local feature vectors are extracted from the query diagram Q and the document diagram D. The feature vectors are organized in a 3D array of size $[H, W, d]$: $H \times W$ local regions each with a d dimension feature vector.

Intuitively, features extracted from regions of different sizes could improve the robustness on diagrams of different scales. Following this intuition, DISH-Conv uses 2D spatial convolutions to generate features for larger regions. A 2×2 kernel builds representations from 4 adjacent regions, generating a new feature vector whose receptive fields is 4 times larger than the original feature vector. Similar, 3×3 kernels generate representations for even larger regions. We also employ a 1×1 convolution. It has the same receptive fields as the input features, but can transform the features to better fit the ranking task.

The convolutions form a pyramid of region representations at multiple scale levels. In addition to scale, the convolution can also learn other types of transformation. It is able to model rotation because rotation changes are essentially a linear transformation of the pixel coordinates. It can also learn transformation in texture and structure, optimizing the input features toward the specific task and dataset.

4.2 Gating Network

During the development of DISH-HP, we identified that the model should ignore unimportant regions of a diagram. DISH-HP employed a heuristic that compares the L2-norm of local feature vectors to a fixed threshold. However, it is not guaranteed that the features of

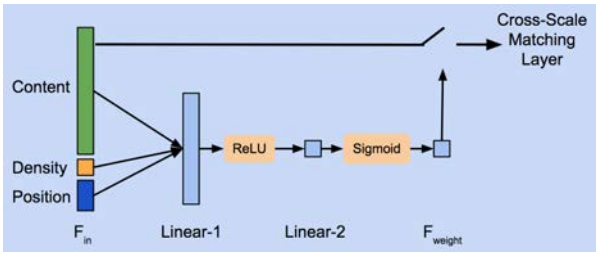


Figure 2: The gating network architecture.

blank areas are always close to zero; and, some high-density areas are also unimportant, for example, callouts and bubbles.

A gating network was developed to automatically learn the importance of local features and to down-weight unimportant features during the matching process (Figure 2). The gating network predicts the importance of a region based on three types of signals: content, position, and density. The content is represented by the region emebedding from the multi-scale representation layer (C features, C is the number of convolution channels). The position features are the coordinates of the center of the perceptive field (2 features). The density feature is the L2-norm as used in DISH-HP (1 feature). These signals are concatenated, generating a $C + 3$ dimension input vector F_{in}, and passed through a 2-layer Perceptron [16]:

$$\vec{h} = ReLU(W_1 \vec{F}_{in} + B_1) \tag{3}$$

$$F_{weight} = \sigma(W_2 \vec{h} + B_2), \tag{4}$$

where W_1, B_1, W_2, and B_2 are the trainable parameters of the Perceptron. σ is the Sigmoid function that converts the free-range network output into the range of $[0, 1]$. The output, $F_{weight} \in [0, 1]$, is multiplied to the original feature vector. F_{weight} close to 1 means that the local feature vector is important and is allowed to pass

through the gate; F_{weight} close to 0 means that the feature vector is unimportant and its effect on the ranker should be muted.

The gating network is trained jointly with the other parts of DISH-Conv, so that it automatically learns to recognize regions that are important for the search task. The importance of a local feature vector is based on not only the density, but more importantly, its content. It allows dense but meaningless regions, such as bubbles, to be detected.

4.3 Ranking with Cross-Scale Match Signals

The multi-scale local region representations, with unimportant regions suppressed by the gating network, are sent to the next cross-scale matching layer. This layer calculates the similarity between each pair of query-document regions. The regions can be taken from different scales. For example, a 16×16 pixel area in the query diagram is compared to 16×16, 32×32, and 48×48 areas in the candidate diagrams (from 1×1, 2×2, and 3×3 convolution). In total, the match is performed at 9 different scales.

For each of the 9 scale combination, the match signals are aggregated through *kernel-pooling* [27]. Kernel-pooling is a soft version of the histogram pooling used in DISH-HP that approximates the histogram function with RBF kernels. It makes the histogram pooling differentiable so that gradients can be back-propagated to the previous convolution layer. The resulting kernel features are similar to the histogram features in DISH-HP. The kernel features from all 9 scale combinations are then combined by a learning-to-rank layer to generate the final score. The learning-to-rank layer assigns different importance to local match signals with different strength (e.g. 'strongly similar', 'weakly similar', 'dissimilar') and at different scales (e.g. 'small query region to small document region' or 'large query region to medium document region'). The output is the predicted relevance score between the query and the document $s(Q, D)$.

DISH-Conv is trained to minimize a standard pairwise loss function:

$$L(Q, P, N) = max(0, \alpha - s(Q, P) + s(Q, N)). \quad (5)$$

The loss function keeps relevant diagrams P closer than any negative diagrams N for each query Q, with a margin α. A query's relevant documents are taken as positive examples. Negative examples are sampled uniformly from the dataset.

4.4 Summary

DISH-Conv is motivated by the interaction-based neural ranking models designed originally to model word-word interaction in text search [3, 8, 27]. DISH-Conv treats local regions as words and models region-region interactions between two diagrams. Its architecture was inspired by the convolution and cross-matching technique proposed by Dai et al. [3]. However, our intuition of using the convolutions and cross-matching is very different from Dai et al. [3]. Our focus is on *scales* and *rotations* while the goal of the original work was to match phrases. We employ spatial convolutions to learn scale-invariant and rotation-invariant transformations, and use cross-matching to generate match signals across multiple scales. A novel gating network is proposed to learn local region importance from position, density, and more importantly, content. Collectively, these elements address several issues identified in DISH-HP, and should enable DISH-Conv to produce more accurate rankings.

5 EXPERIMENTAL METHODOLOGY

This section introduces a new dataset used for experimental evaluation, and the baseline algorithms that were used for comparison.

5.1 Datasets

Although our work is motivated by common tasks in large engineering organizations, there are no standard datasets to use for evaluation because engineering diagrams tend to be proprietary data. A new dataset based on Ikea™ furniture assembly manuals was developed to support experimental evaluation. Although we are unable to redistribute the dataset, others can create a substantially similar dataset based on the descriptions below and additional information posted on our website[1].

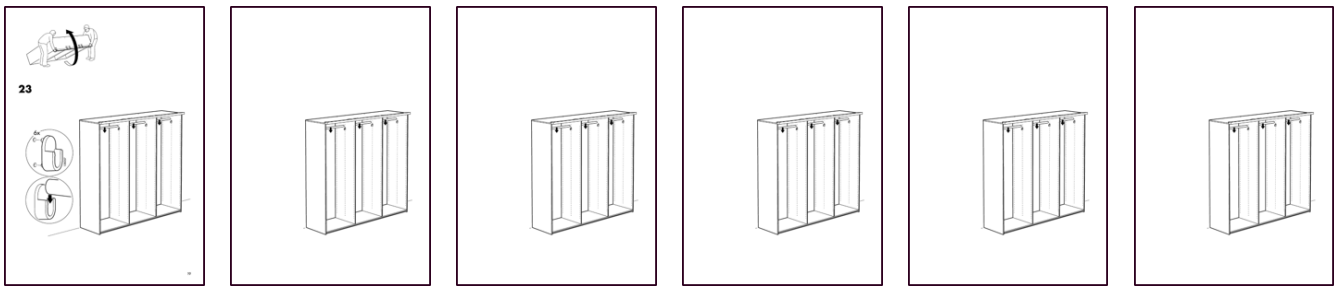
Images: We downloaded product assembly manuals from Ikea™ [2]. Each manual was split into page images. Images that were entirely blank, primarily text, or duplicates of other images were discarded. The result was a corpus of 13,464 furniture assembly diagrams (the Ikea dataset). Each diagram was a black-and-white pdf document. Resolutions of the diagrams vary, with most diagrams around 1000*600 or 600*1000. See Figure 3(a) for an example.

Queries and Relevance Assessments: An automated query generator was developed to create a large set of queries. It tries to locate parts in a diagram that would make reasonable queries. It first selects the zone with relatively high black pixel density as an initial query region, and then gradually expands it by merging the current region with adjacent zones. This process is repeated until the black pixel density of the query region is lower than a threshold. Next, the query region is validated to make sure that it is not too small, too large, entirely black or filled with text. Finally, a query is defined by inserting the validated region into a white background, keeping the image size and the position of the region exactly the same as the original query.

In typical use, queries would not have the same position, scale, and rotation as a region of a relevant diagram. The initial set of queries is named *psr* queries, with lower case letters indicating that the position (p), scale (s), and rotation (r) of the query image are unchanged. Four additional sets of queries were generated to have specific characteristics and increasing difficulty for use in diagnosing weaknesses in retrieval algorithms. The names of these query sets use capitalized letters to mark relative changes. *Psr* queries change the position of the query image, but keep scale and rotation invariant. *pSr* queries change the scale of the query image within the range $[0.5, 2]$. *psR* queries change the rotation 0 to 360 degrees. *PSR* queries change position, scale, and rotation simultaneously. Figures 3b-3f show examples of all five types of queries.

When generating queries, a position, scale, or rotation change is generated randomly within an appropriate range. The transformed query is required to fit entirely within the image boundary so that it does not lose information during the transformation. For position and scale changes, simple rules ensure that the transformed query

[1] A URL will be provided in the camera-ready copy.
[2] https://www.ikea.com/ms/en_US/customer_service/assembly_instructions.html



(a) A relevant document. (b) Query with no change (psr). (c) Query with changed position (Psr). (d) Query with changed scale (pSr). (e) Query with changed rotation (psR). (f) Query with changed position, scale, and rotation (PSR).

Figure 3: Five types of automatically-generated queries for the Ikea dataset.

Table 1: The Ikea Dataset Characteristics

	Description	Number
Corpus	Diagrams from Ikea manuals	13,464
psr	No change	3,649
Psr	Random position change	3,649
pSr	Random scale change (0.5 to 2)	3,649
psR	Random rotation change (0 to 360)	2,988
PSR	Position, scale, & rotation change	3,005

lies within the image boundary. For rotation changes, it is more complex to predict the part boundaries. Therefore, if the rotated query was not entirely within the image boundaries, the change was discarded, and a new random rotation angle generated. If the rotation change failed five times, the image was discarded from the set of queries. Thus, the *psr*, *Psr*, and *pSr* query sets all have the same number of queries (3,649), but the *psR* and *PSR* query sets are somewhat smaller (about 3,000). Table 1 shows the characteristics of the Ikea collection and queries.

Metrics: In the automated query generation process, the source diagram is treated as the query's (only) relevant document. A good retrieval system should rank that document highly. Mean Reciprocal Rank (MRR) is a common metric when there is one relevant document. We also want to know the number of queries for which the relevant document appears at rank 1 and in the top 10 results, thus we also use Recall at cutoff at 1 and 10 (R@1 and R@10). A small set of *manual* labeled queries with multiple relevant documents were studied at Section 6.5.

5.2 Proposed Models and Baselines

Proposed models include two local matching networks, DISH-HP and DISH-Conv. DISH-HP is the unsupervised ranking algorithm proposed in Section 3. DISH-Conv is the supervised local convolutional matching network proposed in Section 4. The proposed models are compared to several baselines, including a bag-of-words image retrieval model based on SIFT features, a neural image retrieval model based on CNN features, and an encoder-decoder network designed specifically for line drawings.

LIRE [14] is an open source image retrieval system based on Lucene. It uses bag-of-words image retrieval models with SIFT features. The indexing and retrieval are unsupervised. We used the open source software [3] to index the images in our corpus and perform the retrieval.

FG-SBIR [17] is a neural retrieval system designed for sketches and line drawings. It is based on an encoder-decoder architecture. The encoder is based on the VGG-16 network [23]; the decoder consists of a chain of upsampling blocks to reconstruct the sketch image from the VGG-16encoder's fully connected layer, helping the encoder to learn a richer representation. The output from the encoder is taken as the image descriptor and used for retrieval. We used the open source implementation [4]. FG-SBIR is a *global* matching approach as the entire image is encoded into a single embedding; local region features are not preserved.

CAM [11] is a state-of-the-art instance retrieval approach based on deep convolutional neural networks. It uses CNN features extracted from an off-the-shelf VGG convolution network trained on ImageNet [4]. For each pre-defined class in ImageNet, CAM generate a class-focused image vector by weighted-summing the CNN features based on their activation to this class. The vectors from all classes are then normalized, whitened, and sum-pooled into a single image vector. Image retrieval is based on the Euclidean distance between the final image vectors. We took the pre-trained CAM model released by the authors [5]. We did not re-train CAM on the Ikea dataset due to lack of class labels. CAM is also a *global* matching approach as local region features are not preserved

The above three baselines used off-the-shelf software. We do not have control over how they process data and extract features. To gain better understanding of the proposed models, we created a set of DISH variants as additional controlled baselines. They differ from DISH-HP and DISH-Conv only in the architectures of the matching networks. The implementation framework, data processing, and feature extraction are controlled to be exactly the same.

DISH-SPoC and DISH-MPoC are two global variants of DISH-HP. They were created to test the effectiveness of local matching against global matching. DISH-SPoC is based on the SPoC descriptor proposed by Babenko and Lempitsky [1], but uses the DISH framework. It takes the same CNN features as DISH-HP, and applies the same filtering on white areas. The filtered CNN features are then aggregated through sum-pooling to generate a single, global image feature vector. The similarity between the query and a document

[3] http://www.lire-project.net/
[4] https://github.com/zoeyangdw/FG-SBIR
[5] https://github.com/imatge-upc/retrieval-2017-cam

diagram is calculated from the cosine similarity between the corresponding vectors. DISH-MPoC is a similar method that replaces the sum-pooling with max-pooling. DISH-SPoC and DISH-MPoC are unsupervised methods.

DISH-Conv-N is DISH-Conv without the spatial convolution; the letter N stands for 'No convolution'. In DISH-Conv-N, the input local CNN features are directly used to calculate region similarities. The model lacks the multi-scale region representation and cross-scale matching in DISH-Conv. DISH-Conv-N can also be viewed as a supervised version of DISH-HP with the hard histogram replaced by soft RBF kernels. DISH-Conv-N is a supervised model.

DISH-Conv-S is a simplified version of DISH-Conv that replaces the spatial convolution with spatial pooling when generating multi-scale region representations. The spatial pooling takes the average of features from smaller regions to generate features for larger regions. Pooling does not have additional parameters, so this model is faster and easier to tune compared to DISH-Conv. DISH-Conv-S separates the effects of convolution from the effects of aggregating small regions into larger regions. It is also a supervised model

Initial Ranking and Re-ranking: We employed a two-stage search pipeline. In the first stage, an initial ranker takes the query and compares it against all diagrams in the corpus, and returns a list of similar diagrams as candidates. In the second stage, a different ranker re-ranks the top 2,000 candidates. DISH-HP, DISH-SPoC and DISH-MPoC are simple models designed for efficient ranking; they were tested on the initial ranking task. DISH-Conv, DISH-Conv-N, and DISH-Conv-S were designed to improve DISH-HP with more complex models; they were tested on the re-ranking task. LIRE, FG-SBIR, and CAM were tested in both stages, providing baselines for both tasks.

5.3 Model Configuration and Implementation

Feature extraction: Off-the-shelf baselines (LIRE, FG-SBIR, and CAM) used their own implementations for feature extraction. All DISH models took the features from an off-the-shelf VGG-16 network trained on ImageNet[6]. Following prior research [21], we extracted features from the last convolution layer. This layer divides the image into $R = W \times H = 14 \times 14 = 196$ grids, each corresponding to a 16 pixel $\times$ 16 pixel rectangular region in the diagram. For each of the 196 regions, a $d = 512$ dimension feature vector is generated. We did not fine-tune the VGG-16 network on engineering diagrams, as the focus of this work is on *matching* local features.

Model Training: Supervised models (DISH-Conv, DISH-Conv-N, and DISH-Conv-S) were trained and tested via 5 fold cross-validation. For each query we sample 10 negative examples. We used the Adam optimizer with batch size 16, learning rate 0.001, and a learning rate decay of 0.5 when validation loss does not decrease. The training stops after 30 epochs or when learning rate drops below $3.12e-5$. The models were implemented in PyTorch.

Hyper Parameters: For DISH-HP, the top $K = 2$ bins were used and low-density area filter threshold T was set as 30. The parameters were chosen through a simple parameter sweep. The kernel pooling layers in DISH-Conv, DISH-Conv-N, and DISH-Conv-S all followed the configurations suggested by prior research [27] and used 11 kernels. The first kernel is the exact match kernel where $\mu = 1, \sigma = 10^{-3}$, or bin $[1, 1]$. The other 10 kernels equally split the cosine range $[-1, 1]$: the μ of bin centers were: $\mu_1 = 0.9, \mu_2 = 0.7, ..., \mu_{10} = -0.9$. The σ of the soft match bins were set to be 0.1. The multi-scale presentation layer in DISH-Conv uses three types of spatial convolutions with different kernel sizes ($1 \times 1, 2 \times 2, 3 \times 3$), 128 output channels, and a stride of 1.

[6]https://download.pytorch.org/models/vgg16-397923af.pth

6 EXPERIMENTAL RESULTS

Two experiments on the Ikea dataset tested the accuracy of the different methods of retrieving diagrams in ranking and re-ranking configurations on large-scale auto-generated relevance labels. Two analyses investigate the behavior and effectiveness of the convolutions and the gating network. An evaluation on a small set of manually-judged queries were conducted to confirm the results from experiments with auto-generated relevance labels.

6.1 Initial Ranking Performance of DISH-HP

DISH-HP is simple, fast, and does not require training. It was tested on the first stage ranking task where the query is compared to all diagrams in the collection. As shown in Table 2, DISH-HP outperforms all other methods by large margins.

Baselines. Among the baselines, only LIRE uses SIFT features; the rest use deep CNN features. LIRE is the least accurate, demonstrating the effectiveness of deep CNN features on engineering diagrams (Table 2). FG-SBIR, which is designed for sketches and line drawings, has higher precision (MRR and Recall@1), but CAM retrieves more relevant diagrams (Recall@10). These results indicate that CNN feature extractors learned from photographs can also extract useful patterns from engineering diagrams.

DISH-SPoC and DISH-MPoC outperforms FB-SBIR and CAM. These four methods are all global matching methods based on CNN features. The main differences of DISH-SPoC and DISH-MPoC with FG-SBIR and CAM are i) they have low-density region filtering, and ii) they use simple sum/max to aggregate local CNN features while FG-SBIR and CAM learn feature aggregation from a dataset. Low-density region filtering is important because these regions bring in noisy signals. The simple pooling functions are task-neutral, while FG-SBIR and CAM are tuned for their training datasets. DISH-MPoC is the strongest baseline.

Local vs. Global. DISH-HP outperforms all other methods by large margins. Like DISH-MPoC, DISH-HP uses deep CNN features and filters blank regions. However, DISH-HP is a local matching method that leverages local region matching evidence. In global methods such as DISH-MPoC, matching is at the entire image level; no local, regional similarity signals are used. They squeeze the image into a 512 dimensional vector. The vector is effective at capturing high-level visual features, such as the shape of a part. However, it loses information at lower granularity levels. As a result, irrelevant diagrams are retrieved, for example, a wheel is retrieved for a round washer because they have similar shapes. Global match methods also missed documents that are partially matched to the query because the other parts in the diagram make the feature vector less similar to the query's feature vector. On the other hand, DISH-HP leverages local similarity signals that tell if the sub-parts between two diagrams are similar. It helps to filter out irrelevant documents and to retrieve partially matched documents.

Table 2: Ranking accuracy on several types of queries. The query is compared to all document diagrams in the Ikea datset.

Method	Invariant (psr)			Position (Psr)			Scale (pSr)			Rotation (psR)			ALL (PSR)		
	MRR	R@1	@10	MRR	R@1	R@10	MRR	R@1	R@10	MRR	R@1	R@10	MRR	R@1	R@10
LIRE	0.19	14%	29%	0.17	9%	19%	0.03	1%	3%	0.00	0	1%	0.00	0%	0%
CAM	0.38	6%	48%	0.31	6%	41%	0.23	5%	32%	0.08	2%	13%	0.04	2%	7%
FG-SBIR	0.44	23%	36%	0.41	19%	35%	0.39	19%	32%	0.10	5%	14%	0.06	3%	8%
DISH-SPoC	0.52	45%	63%	0.48	41%	59%	0.37	30%	48%	0.10	7%	16%	0.07	4%	11%
DISH-MPoC	0.66	58%	76%	0.57	50%	70%	0.44	37%	57%	0.12	9%	18%	0.08	5%	12%
DISH-HP	**0.68**	**59%**	**84%**	**0.65**	**56%**	**82%**	**0.54**	**44%**	**70%**	**0.15**	**11%**	**21%**	**0.10**	**7%**	**14%**

Table 3: Re-ranking accuracy on several types of queries. The top 2,000 documents retrieved from the Ikea dataset by DISH-HP were re-ranked by each method.

Method	Invariant (psr)			Position (Psr)			Scale (pSr)			Rotation (psR)			ALL (PSR)		
	MRR	R@1	R@10	MRR	R@1	R@10	MRR	R@1	R@10	MRR	R@1	R@10	MRR	R@1	R@10
DISH-HP	0.68	59%	84%	0.65	56%	82%	0.54	44%	70%	0.15	11%	21%	0.10	7%	14%
LIRE	0.55	23%	87%	0.66	27%	91%	0.13	2%	46%	0.01	0%	1%	0.01	0	0%
CAM	0.76	60%	88%	0.68	58%	83%	0.60	47%	71%	0.18	12%	22%	0.11	8%	15%
FG-SBIR	0.82	72%	89%	0.76	70%	84%	0.64	58%	72%	0.20	14%	23%	0.11	7%	14%
DISH-Conv-N	0.93	89%	99%	0.83	83%	98%	0.73	65%	85%	0.21	15%	30%	0.09	6%	14%
DISH-Conv-S	**0.94**	**89%**	**99%**	**0.90**	**84%**	**98%**	0.74	65%	86%	0.22	17%	31%	0.10	6%	16%
DISH-Conv	0.91	87%	97%	0.82	74%	94%	**0.78**	**70%**	**90%**	**0.38**	**28%**	**57%**	**0.17**	**10%**	**30%**

Sensitivity to position, scale and rotation changes. DISH-HP is robust to position changes. It had almost the same performance on the position changed queries (Psr) compared to the invariant queries (psr), whereas DISH-MPoC suffered a significant drop when the query is shifted. DISH-HP was relatively sensitive to scale changes: the Recall@10 decreased from 84% to 70% when the query scale was changed (pSr). Rotation changes were more difficult: the Recall@10 was only 21% on rotation changed queries (psR). Queries that simultaneously changed position, scale and rotation (PSR) were the most difficult.

In summary, the experimental results on the baselines show the effectiveness of deep CNN features on engineering diagrams compared to hand-crafted SIFT features. A comparison between models learned on ImageNet and models learned on line drawings indicate that off-the-shelf CNN feature extractors can achieve good Recall, although tuning them for line drawings may increase Precision. Experimental results on DISH-HP demonstrate the power of *local* matching. The local region match signals provide evidence to filter out irrelevant diagrams that share similar shapes with the query but differ in detail. It also enables retrieving relevant diagrams that partially match the query. DISH-HP largely outperforms commonly-used kNN-style global retrieval approaches, including state-of-the-art instance retrieval models. The Recall of DISH-HP was not perfect, especially on scale and rotation changes. But it may serve as good starting point for a retrieval pipeline and to generate training data for more advanced ranking models.

6.2 Re-ranking Performance of DISH-Conv

The second experiment investigated DISH-Conv and its variants. DISH-Conv uses learned weights, multi-scale matching, and convolutional features to improve ranking, especially for queries at different scales and rotations. Due to its greater computational cost, DISH-Conv was only tested as a second stage that re-ranks the top N results returned by a first-stage ranker such as DISH-HP. As a reference, we also tested the baseline methods on the re-ranking task.

As shown in Table 3, LIRE failed to improve the initial ranking of DISH-HP, but both CAM and FG-SBIR were able to improve it. DISH-Conv greatly improved ranking accuracy. In invariant (psr) and position changed queries, over 80% of the relevant documents were ranked first. On queries with scale (pSr) changes, DISH-Conv is able to find 90% of the relevant documents by rank 10, 28% more than DISH-HP. On queries with scale (pSr) changes, DISH-Conv almost tripled Recall@10 of DISH-HP. Together, they lead to higher performance on the most difficult *PSR* queries.

Sources of effectivenss. The two variants, DISH-Conv-N and DISH-Conv-S, help us to understand the sources of effectiveness.

DISH-Conv-N is DISH-Conv without convolution. It can be viewed as a supervised version of DISH-HP , where the hard histogram is replaced by soft RBF kernels, the weight of the kernels is learned, and the heuristic-based region filter is replaced by an end-to-end trained gating network. DISH-Conv-N outperforms DISH-HP, demonstrating that machine-learned kernel weights and machine-learned gating network are more effective than the heuristics.

DISH-Conv-S uses spatial pooling to generate multi-scale representations and matches two diagrams at multiple scale levels. Compared to DISH-Conv-N that only models single-scale match, the pooling-based multi-scale signals in DISH-Conv-S slightly improved the search accuracy for scale changed queries (pSr). Unexpectedly, the model has very high accuracy on position changes (Psr). When a query changes position, the perceptive field receives different content, and the convolution features changes. Pooling features from adjacent regions produces a descriptor for a larger region and compensates for the position change. The results shows

that matching at multiple scales is not only effective for scale invariance, but also improves position invariance.

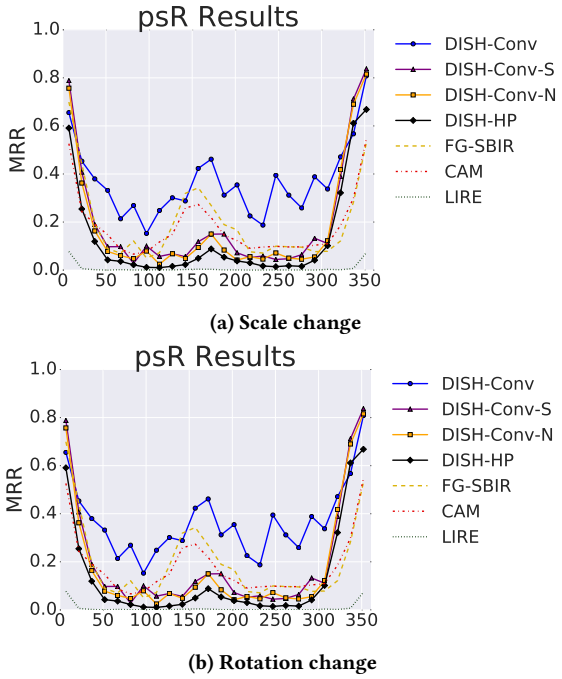
DISH-Conv is the most effective model on queries with scale and rotation changes (pSr, psR and PSR). In contrast to DISH-Conv-S which uses simple average pooling to generate features, DISH-Conv learns a matrix transformation of small-region features to larger-region features. The transformation is more powerful in handling complex changes in scale and rotation. Besides, the parameters are *learned* from the training data, so that they are tuned to fit the characteristics of the diagram dataset and the ranking task. The convolution, the cross-scale matching, together with machine-learned weights, make DISH-Conv more accurate and more robust.

6.3 Scale and Rotation Invariance Analysis

DISH-Conv uses convolutions and cross-matching to handle scale and rotation differences between queries and diagrams. An analysis investigates the effectiveness of these components (Figure 4).

Figure 4a shows each model's sensitivity to scale changes in pSr queries. Most models are best when the scale is unchanged (1.0). Ranking accuracy gradually decreases when the query is scaled down or up. DISH-Conv and DISH-Conv-S are best for all scale changes, due to their multi-scale representations and cross-scale matching. DISH-Conv is more robust to extreme scale changes, demonstrating that convolution is more powerful than pooling.

Figure 4b shows each model's sensitivity to rotation changes in psR queries. All models except DISH-Conv have a bow-like curve. Their MRR scores are high when there is no rotation, but drop to near 0 when the query is rotated ± 50 degrees. MRR improves somewhat near 180 degrees rotation because some queries are symmetric (e.g., rectangles); the symmetry effect is more obvious on FG-SBIR and CAM. Among all models, DISH-HP is the least able to handle rotation, and DISH-Conv is the most robust, although there is still a significant impact. Convolutions in DISH-Conv learn feature transformations that compensate somewhat for the rotation changes.



(a) Scale change

(b) Rotation change

Figure 4: Re-ranking accuracy (MRR) for different amounts of change in query scale and rotation.

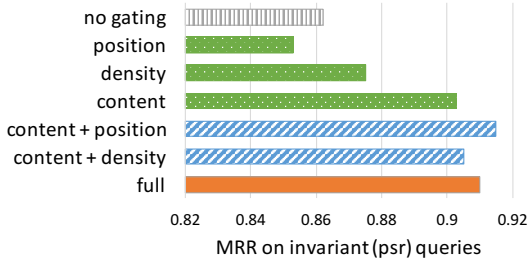


Figure 5: Re-ranking accuracy (MRR) of DISH-Conv on invarinat (psr) queries when gating uses different features.

6.4 Gating Network Analysis

DISH-Conv uses a gating network that estimates the importance of each region based on three types of features: the region's pixel density, estimated by the L2 norm of the feature vector; the region's position, represented by the coordinates of the region center; and the region's content, represented by its feature vector. An ablation study examined the effectiveness of the gating network. Figure 5 compares the accuracy of DISH-Conv with and without the gating network. It also compares gating networks using different types of features.

As shown in Figure 5, the MRR of DISH-Conv drops when the gating network is disabled. Without the gating network, the matches from meaningless areas dominated DISH-Conv's match signals, which produces poor results.

Comparison of gates using position, density, and/or content features indicates that content is the most powerful feature. The content gate learns the importance of a region from its CNN feature vectors, essentially capturing the *meaning* of the region. The position gate predicts a region's importance based on its location in the diagram, assuming that the center of the image is more likely to contain content of interest, which is not necessarily true. Using the position feature alone leads to worse results. The density gate used the L2 norm of the region feature vector, which is similar to the low-density region filter used by DISH-HP. The density gate is moderately better than having no gate. Combining all three types of features (full) provides further improvement, but the best combination omits the density feature, which is subsumed by the content feature.

In summary, identifying important regions and ignoring noisy regions is crucial for matching diagrams. A novel gating network was proposed to detect regions that are unimportant for ranking. Experimental results show that the gating network boosts the ranking accuracy of Local-Conv. The feature ablation study shows that learning to recognize unimportant contents delivers most of the gain, followed by the position of the region within the image.

Table 4: Manual Relevance Assessments statistics

Relevance Judgment Criteria	Avg Doc/Query
3 Same furniture	5.7
2 Likely to be the same furniture	1.3
1 Different, but similar assembly	5.8
0 Not related	63.6

Table 5: Re-ranking accuracy on 50 rotation queries (psR) with manual relevance assessments.

	Auto Labels			Manual Labels		
Method	MRR	R@1	R@10	MRR	R@1	R@10
DISH-HP	0.11	6%	18%	0.17	3%	10%
FG-SBIR	0.13	7%	21%	0.16	4%	10%
DISH-Conv-N	0.15	8%	28%	0.23	6%	14%
DISH-Conv-S	0.15	10%	22%	0.19	5%	11%
DISH-Conv	**0.31**	**22%**	**46%**	**0.37**	**8%**	**22%**

6.5 Manual Relevance Assessments

The experiments above measure search accuracy using large-scale, auto-generated relevance labels, where a query's source diagram is treated as its only relevant document. The last experiment measures accuracy using a small set of manual relevance judgments.

For each of 50 queries, the top 30 diagrams retrieved by Psr, pSr, and psR query variants were judged. 76 diagrams per query were assessed, on average. Three assessors judged the relevance of each diagram on a scale of $0-3$. Table 4 shows the relevance criteria and distribution. Diagrams with labels of 2 and 3 were considered relevant. On average, a query had 7 relevant diagrams.

These assessments were used to evaluate DISH methods and the best baseline FG-SBIR in the re-ranking setting. We report results for rotation (psR) queries because they differentiate the ranking methods the most. As shown in Table 5, the relative order of the ranking methods using manual labels is consistent with that on auto-generated labels: all DISH-Conv methods improved the initial ranking from DISH-HP; DISH-Conv methods were better than FG-SBIR; and DISH-Conv was most accurate. R@1 and R@10 values are lower for manual labels than for auto labels because there are 7× more relevant documents in this condition. MRR values are higher for manual labels, meaning that some top-ranked documents that are considered non-relevant by the auto labels are actually relevant.

In summary, the relative order of the ranking methods does not change when switching from auto labels to manual labels, validating the conclusions drawn from previous experiments.

7 CONCLUSION

Engineering diagram search has specific characteristics in that diagrams are sparse line drawings that contain visual metadata; the query is matched to a small region of a diagram; and the target part can appear at arbitrary position, scale, and rotation. This paper proposes local matching networks that model the interactions between local regions of the query diagram and the document diagrams, providing multiple and detailed match signals. Novel techniques were developed to address the specific issues in engineering diagrams.

This paper first presents an unsupervised model, DISH-HP. It leverages local region features extracted from the middle layer of a deep convolutional neural network. It calculates the similarity scores between query regions and document regions, and uses histogram pooling to combine evidence. The local match signals allow small differences to be recognized. Experimental results showed that matching local, region-level signals leads to superior performance over several retrieval algorithms based on global, image-level features.

The paper then presents a supervised model, DISH-Conv, to improve the robustness to scale and rotation changes. It is inspired by a state-of-the-art neural text ranking model, Conv-KNRM, which uses convolutional networks to generate n-gram embeddings from word embeddings. DISH-Conv adapts that idea and employs spatial convolutions to generate larger-scale representations from small-scale representations, enabling the query and the document diagrams to match at multiple scales. A novel gating network is proposed to improve the search accuracy by automatically detecting unimportant regions of an image. Experimental results show that DISH-Conv improves the ranking of DISH-HP substantially, especially when the target part appears at a different scale and/or is rotated to a different angle in the document diagram.

Our analysis reveals that the power of DISH-Conv is a combination of its ability to transform features through the convolutions, match at multiple scale levels, and suppress unimportant regions through the gating network. The spatial convolution in DISH-Conv generates multi-scale representations that are able to model changes in scale. The convolution is also able to learn a feature transformation that compensates for moderate changes of rotation. The gating network is able to capture the meaning of a region from its feature vector, and suppresses noisy match signals from unimportant regions. The convolution and the gating network are trained end-to-end so that they automatically learn how to transform features and assign importance for the specific task and dataset.

A pipelined system leverages the advantages of both models. DISH-HP efficiently scans the collection to identify a small set of candidate images. DISH-Conv re-ranks the candidates with less noise and richer match signals, delivering more accurate rankings. Together, DISH-HP and DISH-Conv are able to search a large corpus of engineering diagrams efficiently and effectively.

REFERENCES

[1] Artem Babenko and Victor S. Lempitsky. 2015. Aggregating Local Deep Features for Image Retrieval. In *Proceedings of the IEEE International Conference on Computer Vision*. 1269–1277.

[2] Artem Babenko, Anton Slesarev, Alexander Chigorin, and Victor S. Lempitsky. 2014. Neural Codes for Image Retrieval. In *Proceedings of the 13th European Conference on Computer Vision*. 584–599.

[3] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. 126–134.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, 248–255.

[5] Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. 2011. Sketch-Based Image Retrieval: Benchmark and Bag-of-Features Descriptors. *IEEE Transactions on Visualization and Computer Graphics* (2011), 1624–1636.

[6] Manuel J. Fonseca, Alfredo Ferreira, and Joaquim A. Jorge. 2005. Content-based retrieval of technical drawings. *International Journal of Computer Applications in Technology* (2005), 86–100.

[7] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. 2014. Multi-scale Orderless Pooling of Deep Convolutional Activation Features. In *Proceedings of*

the 13th European Conference on Computer Vision. 392–407.

[8] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A Deep Relevance Matching Model for Ad-hoc Retrieval. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management*. 55–64.

[9] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry P. Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *22nd ACM International Conference on Information and Knowledge Management*. 2333–2338.

[10] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2008. Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search. In *Proceedings of the 10th European Conference on Computer Vision*. 304–317.

[11] Albert Jimenez, Jose M. Alvarez, and Xavier Giró i Nieto. 2017. Class Weighted Convolutional Features for Visual Instance Search. In *Proceedings of the British Machine Vision Conference 2017*.

[12] Yannis Kalantidis, Clayton Mellina, and Simon Osindero. 2016. Cross-Dimensional Weighting for Aggregated Deep Convolutional Features. In *Proceedings of the 15th European Conference on Computer Vision*. 685–701.

[13] David G. Lowe. 1999. Object Recognition from Local Scale-Invariant Features. In *Proceedings of the 5th IEEE International Conference on Computer Vision*. 1150–1157.

[14] Mathias Lux and Oge Marques. 2013. *Visual Information Retrieval Using Java and LIRE*. Morgan & Claypool Publishers.

[15] Stefan Müller and Gerhard Rigoll. 1999. Engineering Drawing Database Retrieval Using Statistical Pattern Spotting Techniques. In *Proceedings of the 3rd International Workshop on Graphics Recognition Recent Advances,*. 246–255.

[16] Nils Nilsson. 1965. *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*. McGraw-Hill.

[17] Kaiyue Pang, Yi-Zhe Song, Tony Xiang, and Timothy M. Hospedales. 2017. Cross-domain Generative Learning for Fine-Grained Sketch-Based Image Retrieval. In *Proceedings of the British Machine Vision Conference*.

[18] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. 2007. Object Retrieval with Large Vocabularies and Fast Spatial Matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

[19] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. 2008. Lost in Quantization: Improving Particular Object Retrieval in Large Scale Image Databases. In *Proceedings of the 14th IEEE Conference on Computer Vision and Pattern Recognition*.

[20] Jiantao Pu and Karthik Ramani. 2006. On visual similarity based 2D drawing retrieval. *Computer-Aided Design* 38, 3 (2006), 249–259.

[21] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014. CNN features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 806–813.

[22] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A Latent Semantic Model with Convolutional-Pooling Structure for Information Retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. 101–110.

[23] Karen Simonyan and Andrew Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556* (2014).

[24] Josef Sivic and Andrew Zisserman. 2003. Video Google: A Text Retrieval Approach to Object Matching in Videos. In *Proceedings of the 9th IEEE International Conference on Computer Vision*. 1470–1477.

[25] Giorgos Tolias, Ronan Sicre, and Hervé Jégou. 2015. Particular object retrieval with integral max-pooling of CNN activations. *arXiv Preprint arXiv:1511.05879* (2015).

[26] Visual Geometry Group, Department of Engineering Science, University of Oxford. 2018. Very Deep Convolutional Networks for Large-Scale Visual Recognition. http://www.robots.ox.ac.uk/~vgg/research/very_deep/, Last accessed on 2018-10-15.

[27] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-End Neural Ad-hoc Ranking with Kernel Pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 55–64.

Evaluating Anti-Fingerprinting Privacy Enhancing Technologies

Amit Datta*
Snap Inc.
amit.datta@snap.com

Jianan Lu
University of California, Berkeley
amberljn@berkeley.edu

Michael Carl Tschantz
ICSI
mct@icsi.berkeley.edu

ABSTRACT

We study how to evaluate Anti-Fingerprinting Privacy Enhancing Technologies (AFPETs). Experimental methods have the advantage of control and precision, and can be applied to new AFPETs that currently lack a user base. Observational methods have the advantage of scale and drawing from the browsers currently in real-world use. We propose a novel combination of these methods, offering the best of both worlds, by applying experimentally created models of a AFPET's behavior to an observational dataset. We apply our evaluation methods to a collection of AFPETs to find the Tor Browser Bundle to be the most effective among them. We further uncover inconsistencies in some AFPETs' behaviors.

CCS CONCEPTS

• **Security and privacy → Pseudonymity, anonymity and untraceability**.

KEYWORDS

Privacy enhancing technologies; website fingerprinting; testing

ACM Reference Format:
Amit Datta, Jianan Lu, and Michael Carl Tschantz. 2019. Evaluating Anti-Fingerprinting Privacy Enhancing Technologies. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313703

1 INTRODUCTION

Online data aggregators track consumer activities on the Internet to build behavioral profiles. In addition to using stateful methods, such as cookies, some trackers use stateless tracking mechanisms, also known as browser fingerprinting. A stateless tracker extracts fingerprints from consumers as a collection of several attributes of the browser, operating system, and hardware, typically accessed through Javascript APIs. Fingerprints collected on websites like panopticlick.eff.org and amiunique.org/fp demonstrate that they are sufficiently unique and stable for tracking purposes [15, 36]. The list of attributes that can be used in fingerprints is rapidly increasing [3, 4, 13, 18, 21, 41, 55]. Studies have also uncovered fingerprinting code on popular webpages [3, 4, 18].

Anti-Fingerprinting Privacy Enhancing Technologies (AFPETs) aim to protect consumers against fingerprinting by masking, or spoofing, the values of attributes. Our goal is to find attributes that AFPETs are not masking and to quantify their effects on privacy. We develop a method that compares the trackability of AFPET-modified fingerprints with those of the original fingerprints.

Depending on the goals, AFPET evaluation could depend on the context in which the AFPET is used, accounting for features of other users and non-users, or be a more theoretical assessment of the AFPET's potential, untied to the vagaries of today. For example, if the goal of evaluation is to determine which AFPET to use today, one would want to know how many other users of the AFPET there are since they will form the anonymity set – the group of other users one will blend in with. If instead the goal is to determine which AFPET to fund for further development, the user numbers of today may matter less than the technical or theoretical capabilities of the AFPET. Given that no one AFPET evaluation can match all goals, we will explore points in the space of possible evaluations while focusing more on prospective evaluations.

Methods. First, we consider a more theoretical, experimental analysis that directly looks at an AFPET's ability to mask attributes. This method runs browsers with and without an AFPET installed to determine which attributes the AFPET masks. For this purpose, we develop an experimental framework, PETInspector, which has three components: the *fingerprinting server* (FPServer), which collects fingerprints from visitors, the *client simulator* (ClientSim), which simulates consumers and drives them to FPServer with and without AFPETs, and the *analysis engine* (AnaEng), which compares fingerprints across clients to produce a *mask model* characterizing AFPET behaviors. This tool can be applied to new AFPETs that currently lack a user base. This experimental method does not require access to the source code of AFPETs. However, it does not tell us which attributes are the most important to mask.

Next, we consider a highly context-dependent, observational method. Websites like panopticlick.eff.org and amiunique.org/fp obtain large sets of real-world fingerprints, revealing which are the most trackable (i.e., unique and predictable). In principle, these datasets can be studied to evaluate an AFPET by selecting the fingerprints generated by users of that AFPET and, for each such fingerprint, checking how trackable they are compared to other fingerprints in the dataset. We have implemented the core task of measuring trackablity as a tool, FPInspector, which simulates a simple tracker and computes statistics quantifying anonymity, such as entropy. In practice, however, such observational datasets may contain too few users of an AFPET, especially for new ones, for FPInspector evaluate it. Furthermore, in some cases, it may be difficult to determine which fingerprints correspond to which AFPETs. Thus, utilizing such a dataset requires a more nuanced approach.

Then, we develop a hybrid method combining observational and experimental data to enable the evaluation of AFPETs with low or no usage within the context of browsers used today but without

*The majority of this author's contributions were made at Carnegie Mellon University.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313703

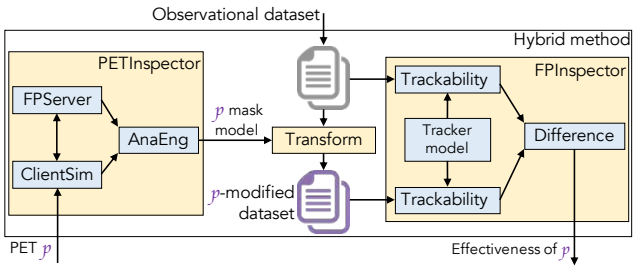


Figure 1: The hybrid AFPET evaluation method takes an observational dataset and a PET, and outputs its effectiveness.

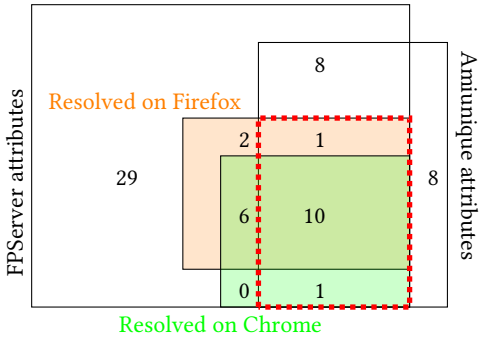


Figure 2: FPServer collects 49 attributes, of which 29 remain unexercised by ClientSim. Amiunique dataset has 28 unique attributes, of which 8 aren't collected by FPServer and 8 are collected but remain unexercised by ClientSim. The red dotted rectangle represents the intersection of attributes exercised by ClientSim and those present in the amiunique dataset, which are used for our hybrid evaluation.

access to the AFPET source code. Our hybrid method combines FPInspector with PETInspector as outlined in Figure 1. It contextualizes the mask model produced by PETInspector by applying it to an observational dataset of real-world fingerprints to produce a counterfactual dataset representing what the browsers would look like to trackers had everyone used the AFPET. By comparing the trackabilities on the two datasets, we evaluate the effectiveness of the AFPET. By parametrically leveraging data from ongoing, large-scale measurement studies, our methods may be adapted for the ever-changing landscape of browsers with little additional work.

Finally, we adjust the hybrid method to take into account the number of users an AFPET has. This shifts the analysis even further in the direction of examining the PET's current abilities.

Results. Using PETInspector, we resolved with high confidence whether 15 AFPETs explicitly claiming to protect against fingerprinting mask 20 attributes of Firefox and 18 attributes of Chrome. We also looked at 11 other popular *blacklisting PETs* (BLPETs), which operate by blacklisting domains known to engage in tracking. While they do not make a claim of protecting against fingerprinting, they should not make matters worse by giving browsers a more unique fingerprint, a property we check them for.

We found that all but the Tor Browser Bundle masked 9 or fewer of the resolved attributes, at least in their default configurations. In particular, we found that Tor left a single attribute, `platform`, unmasked while all others left at least 12 attributes unmasked. PETInspector also uncovered undocumented behaviors and inconsistencies in how some PETs modify various attributes:

- Brave Browser spoofs the `User-Agent` to appear like Chrome, but modifies the `Accept-Language` header, `language` and `plugins` differently than baseline Chrome. This can make Brave users stand out from other Chrome users. We have raised the issue with Brave developers and have received comments from them acknowledging the issue [9, 10].
- While both Privacy Badger and Firefox send the `Dnt` header, only Firefox sets the `doNotTrack` variable in JavaScript's navigator object. As a result, web-services which only use JavaScript to detect the Do Not Track choice will not be able to do so for Privacy Badger users. Furthermore, this inconsistency may make Privacy Badger's users stand out, making them easier to track. We raised this issue with Privacy Badger developers who have since fixed the issue [11].
- HideMyFootprint randomizes the `User-Agent` header, while not modifying the `platform`. This leads to inconsistencies like the `User-Agent` containing *Windows NT 10.0* on a *Linux x86_64* `platform`. Moreover, it sends an additional `Pragma` header, which can make users distinguishable.
- While the Tor Browser Bundle hides the OS by spoofing attributes like `User-Agent` and `cpu class`, one may still infer that information from the `javascript fonts` revealed.

We found 6 AFPETs which masked 4 attributes, but they did not all mask the same set of attributes. To break such ties, we used the hybrid method. We used a pre-existing dataset of over 25,000 real-world fingerprints collected on and provided by the website amiunique.org. [1] Of the 18–20 attributes resolved for each AFPET, only 12 appear in the `amiunique.org` dataset. Figure 2 provides an overview of how we selected attributes for our hybrid evaluation. For these 12 attributes, the hybrid method generates a set of PET-modified fingerprints from the original fingerprints and measures the effectiveness of the 15 AFPETs with FPInspector.

Our hybrid method finds that even with just 12 attributes, 13 AFPETs do not provide much protection over using no PET, decreasing the entropy revealed from about 13 bits without any PET to 11 bits with the AFPET. It finds Tor Browser Bundle (Tor BB) to be most effective, revealing under 3 bits of entropy. Given that some AFPETs only claim to protect a single attribute, these findings do not necessarily show any of them to be broken, but it does provide useful information for users and privacy advocates about their limitations. Furthermore, we found that the AFPET Trace did not mask two attributes that its documentation claimed it did (Table 3).

Recognizing that automation has its limitations, we manually analyzed some of the more interesting findings. We found that some AFPETs performed better when switched out of their default configuration. While we find that some do mask attributes labeled as inconclusive by PETInspector, we did not find any falsely labeled as masked or as unmasked.

[1] Graciously provided by Pierre Laperdrix, one of the creators of amiunique.org.

A source of entropy for Tor BB fingerprints is the revealed screen resolution, which is only partly masked. Tor BB reveals partial information about the screen resolution of its users using a spoofing strategy which depends on the true resolution for usability reasons. We explore a space of alternate spoofing strategies and find some to be just as effective according to our metrics despite being more usable, by utilizing more pixels on average for browsing than Tor.

Interpretation. AFPETs do not claim to protect against all forms of fingerprinting, and, thus, our results should not necessarily be interpreted as finding flaws. Nevertheless, our tool can be useful for AFPET developers. It can test whether they masked the attributes they intended to do so and help ensure that their documentation is correct. Indeed, we found that AFPETs Trace and Tor did not mask all the attributes that their documentation claimed they did.

BLPETs do not claim to protect against fingerprinting, but even they should avoid making browsers more fingerprintable than they already are. For example, we found that Privacy Badger made fingerprinting easier by modifying an attribute in a particular and undocumented way. Despite not making any anti-fingerprinting claims, its developers updated Privacy Badger.

For consumers and their advocates, our results are useful beyond the pre-existing, and sometimes flawed, documentation. In addition to double checking documentation, such consumers may be less concerned with whether PETs meet their specifications than their overall effectiveness, which our tools measure.

Our results are best understood as providing a lower bound on how much room for improvement remains for AFPETs. Our lower bound is sound in that when PETInspector claims that an AFPET leaves an attribute unmasked, it really is not masking it, is not varying the attribute often enough to be effective, or is not masking enough values of the attribute to protect our test browser platforms. Our bound is only a lower bound since, by resolving only the status of 18–20 attributes of each browsing platform, we might label some attributes in need of masking as inconclusive. More attributes can be added to our tools, but the set of possible attributes is open ended and finding platforms that differ in all attributes can be difficult.

Contributions. We make the following main contributions:

- We develop PETInspector to find how 15 AFPETs (and 11 BLPETs) mask 18–20 different attributes. By obtaining a more complete picture of PETs' behaviors, we uncover some inconsistencies and peculiarities (Section 4).
- We develop a hybrid method for evaluating AFPETs from an observational dataset of real-world fingerprints and apply it to evaluate 15 AFPETs. We find Tor BB to be the most effective AFPET among the ones we evaluate (Section 5).
- We adjust the hybrid method to also consider the current number of users each AFPET has (Section 6).
- We explore a space of alternate spoofing strategies for screen resolution by Tor BB and find some which have higher screen utilization than Tor BB, but are just as effective (Section 7).

2 PRIOR WORK

Prior work finds that various attributes are trackable by measuring the uniqueness and predictability of fingerprints collected from real-world browsing platforms [15, 36, 63]. However, few studies evaluate the effectiveness of AFPETs against fingerprinting.

Many prior studies have focused on BLPETs, which use blacklists to block known tracking domains and scripts. Since BLPETs try to prevent the consumer's browser from interacting with trackers, metrics suggestive of successful interactions (e.g., third-party requests sent, cookies placed, etc.) are good indicators of BLPET effectiveness. Studies have evaluated BLPETs by comparing these metrics between browsers with and without the BLPET when visiting popular websites [18, 26, 28, 31, 32, 38, 40, 50]. FPGuard takes a blacklisting strategy to protect against fingerprinting: it uses heuristics to identify fingerprinting domains and blocks them [20]. Gulyás et al. study the tradeoff between a BLPET suppressing some trackers but also leading to the browser having a more unique fingerprint by being a rare browser extension [25].

Most AFPETs protect against fingerprinting by spoofing browser, operating system and hardware characteristics, without blocking specific domains and scripts. AFPETs like the Tor Browser standardize various attribute values [48], whereas others like PriVaricator [45], FP-Block [59], Blink [35], and FPRandom [34] vary them. Metrics for evaluating BLPETs are not able to meaningfully evaluate AFPETs. Some studies have evaluated attribute-varying AFPETs by observing variations in fingerprints when using these AFPETs (e.g., [34, 45]). Vastel et al. look at how AFPETs can introduce inconsistencies between attributes leading to a more unique fingerprint [61]. Our work differs from these and uses a combination of experimental and observational data to more thoroughly evaluate AFPETs.

3 TRACKERS AND PETS

When a user visits a webpage, trackers can have the user's browser execute code that requests information about the user's browsing platform, including their hardware, operating system, and the browser itself. Table 3 provides a list of 49 attributes known to be good candidates for fingerprinting. The tracker can combine multiple attributes $a_1, \ldots, a_n$ to compute a *fingerprint* $id(b) = \langle a_1(b), \ldots, a_n(b) \rangle$ of the browsing platform b where $a_i(b)$ represents the value of attribute a_i for the platform b. A tracker can use fingerprints to identify browsing platforms visiting two websites as being the same one. The more unique the fingerprint is for each user, the fewer false matches the tracker will produce in linking two different users. The more predictable, or stable (ideally, unchanging), the fingerprint is as a user goes from website to website, the fewer matches the tracker will miss.

To protect themselves from fingerprinting, consumers can install AFPETs on their browsing platform to reduce the uniqueness or stability of the platform's fingerprints. Upon installing a PET p, the consumer's browsing platform b is modified to $p(b)$. As a result, the tracker interacts with $p(b)$ and extracts fingerprint $id(p(b))$.

In this study, we look at three types of PETs:

I **Attribute standardizing (AS).** These AFPETs reveal one (full standardization) or one of a small set of possible values (partial standardization) for an attribute. Full standardization makes all AFPET users appear identical, whereas partial standardization makes them appear from a few groups, with respect to that attribute. An AFPET may choose partial over

full standardization if spoofing the attribute value has usability implications.

II **Attribute varying (AV).** These AFPETs vary the value of an attribute so that the values of each user varies across browsing activities. Such variations may affect both the predictability and the uniqueness of the revealed attribute. Laperdrix et al. [34] show that variation AFPETs can vary attributes in a manner that reduces their usability impact.

III **Interaction blocking (IB).** These BLPETs block some or all interactions between the browsing platform and trackers. They rely on a blacklist (e.g., EasyPrivacy) to block interactions matching known tracking patterns. Trackers interacting with browsing platforms with these PETs receive an error message instead of the true fingerprints.

We are primarily interested in evaluating AFPETs that modify the attribute values either by standardizing (I) or varying (II) their values. In some places, we comment on BLPETs that block interactions with known trackers (III). We do so even for BLPETs not claiming to be AFPETs since they are popular, have been the subject of past evaluation studies, have the potential to unintentionally make fingerprints more unique (as we find with Privacy Badger), and can be used as AFPETs. However, we do not directly compare them to the AFPETs since they do not purport to modify any attributes explicitly, and their quality depends upon the quality of their blacklists, necessitating a different form of evaluation.

We leave out of scope PETs that protect against fingerprinting by blocking scripts (e.g., NoScript [29] and ScriptSafe [6]) since they have considerable impact on usability [28]. We also leave out PETs like AdNauseum (adnauseam.io) that do not attempt to prevent tracking but rather to make it pointless by injecting noise into the user's history with fake clicks and website visits.

In this paper, we consider a total of 26 PETs. We assign each PET a unique abbreviation, which we use in some tables. We present the full list of PETs and their abbreviations in Table 1. 23 of the 26 PETs are extensions for Chrome and Firefox, two are full browsers, and one is a browser configuration. 15 of the 26 PETs are AFPETs and purport to either standardize or vary attribute values, while 11 others are popular BLPETs. Some PETs assume mixed strategies.

We went over the documentation of the PETs to uncover how they purport to modify attributes. For all PETs that explicitly document masking an attribute, we place a □ in the corresponding cell in Table 3. Next, we demonstrate how we use our experimental method can check whether the documentation is accurate.

Table 1: List of PETs we study, their abbreviation, and strategy to protection. Most PETs are browser extensions, * indicates full browsers, and ** indicates browser configurations. For AFPETs, we list its number of users.

PET	Abbr.	Strategy	AFPET	Users
Chrome PETs				
CanvasFingerprintBlock [8]	CFB	AS	✓	7.6K
Privacy Extension [54]	PE	AS	✓	915
Brave [12]	BR*	AS+IB	✓	N/A
Canvas Defender [43]	CD_C	AV	✓	20K
Glove [44]	GL	AV	✓	342
HideMyFootprint [1]	HMF	AV+IB	✓	177
Trace [2]	TR	AV+IB	✓	N/A
Adblock Plus [19]	AP_C	IB		
Disconnect [14]	D_C	IB		
Ghostery [23]	GH_C	IB		
Privacy Badger [17]	PB_C	IB		
uBlock Origin [27]	UO_C	IB		
Firefox PETs				
Blend In [49]	BI	AS	✓	858
Blender [39]	BL	AS	✓	1.8K
No Enum. Extensions [51]	NE	AS	✓	N/A
Stop Fingerprinting [46]	SF	AS	✓	1.8K
Tor Browser Bundle [48]	Tor*	AS	✓	≈4M
TotalSpoof [22]	TO	AS	✓	265
Canvas Defender [43]	CD_F	AV	✓	5.3K
CanvasBlocker [30]	CB	AV	✓	27K
Adblock Plus [19]	AP_F	IB		
Disconnect [14]	D_F	IB		
Ghostery [23]	GH_F	IB		
Privacy Badger [17]	PB_F	IB		
Tracking Protection [57]	TP**	IB		
uBlock Origin [27]	UO_F	IB		

4 EXPERIMENTAL EVALUATION OF AFPETS

We now consider an experimental, or test-based, approach to AFPET evaluation conducted with artificial users. These artificial users browse on platforms differing in whether they have an AFPET installed. By comparing fingerprints generated by these artificial users, we infer which attributes the AFPET is masking. We use the degree of masking by each AFPET as an evaluation metric.

4.1 Method

Our experimental framework, PETInspector, is composed of three parts. The *client simulator*, ClientSim, creates and drives experimental browsing platforms, with and without various AFPETs installed, to visit a server. The *fingerprinting server*, FPServer, plays the role of an online tracker and collects fingerprints when the browsing platforms, driven by ClientSim, visit it. The *analysis engine*, AnaEng, compares fingerprints across clients to detect whether an AFPET varies, standardizes, or does not mask the value of an attribute. To observe these behaviors, AnaEng compares the value of the attribute on the browsing platform without any AFPET (i.e., on the baseline browser) with the value when an AFPET is installed.

Client Simulator. ClientSim drives simulated clients using browsing platforms with different configurations to visit FPServer. For each base configuration and AFPET, ClientSim simulates a pair of clients only differing on whether the AFPET is installed, to allow the isolation of the AFPET's effects.

We choose the base configurations to exercise a wide range of attribute values in hopes of triggering an AFPET's masking behavior even when the masking is partial. To exercise more platforms than we have access to, ClientSim simulates browsing platforms either locally on a computer or on pre-configured VirtualBox

virtual machines [62]. ClientSim configures different fonts, timezones, languages, and screen properties using OS and browser functionalities. Some attributes directly depend upon hardware (e.g., `max touch points`) or fixed OS libraries (e.g., `math` attributes) preventing their simulation. Avoid altering others due them possibly interfering the operation of PETs (`DNT enabled`, `openDB`, `indexedDB`, two `storage` attributes, and six `header` attributes), being antiqued (`plugins`), or being a function of lower-level attributes (such as `adBlock installed` and `has lied with`).

After setting up a simulated browsing platform, ClientSim drives browser instances on them using the Selenium Webdriver [52] to FPServer. The browser instances interact with FPServer in a specified pattern of reloads and idling to provide insights about the modification behavior of PETs. In hopes of triggering a PET's ability to mask by varying attribute values, ClientSim drives its browsers across various boundaries that may cause the PET to refresh its spoofed value: *reloads* of a single domain, visits to different *domains* (we give FPServer two domain names), and browsing across *sessions*. We define a session to browsing separated by 45 minutes of down time, following Mozilla's definition of a session as a continuous period of user activity in the browser, where successive events are separated by no more than 30 minutes [60].

Fingerprinting Server. FPServer collects attributes collected by the open-source fingerprinting projects FPCentral [33] and Panopticlick [16], often by reusing their code. We list these attributes in the first column of Table 3. Similar to websites like panopticlick.eff.org and amiunique.org/fp, any browser visiting FPServer's domain can view their fingerprint, while FPServer retains a copy.

Analysis Engine. To check for masking by a PET, AnaEng uses both the fingerprints collected by FPServer from the browsers driven by ClientSim and information directly from ClientSim stating which browser used which PETs and in which configurations. Figure 3 provides an overview of AnaEng. In short, the analysis looks for both masking by standardization and by variation. If it detects standardization or variation for an attribute, it models the attribute as masked in the mask model of the PET that it produces. It models an attribute as unmasked if it is able to thoroughly test it and find neither type of masking. The possible results of the analysis are

(1) Inconclusive: cannot test variation due the baseline browser varying the attribute
(2) Masked: detect AFPET-induced variation by seeing variation with the AFPET but not without the AFPET
(3) Masked: detect AFPET-induced standardization by seeing value change in a stable manner from baseline browser to browser with AFPET
(4) Inconclusive: cannot rule out partial standardization due to lack of browsing platforms that differ enough in the attribute
(5) Unmasked: rule out impactful standardization as unlikely

By *impactful* partial standardization, we mean standardization that affects at least a fraction f of the values. In general, ruling out partial standardization with experiments requires testing for all possible attribute values, a prohibitively expensive, if not impossible, task for many attributes. However, AnaEng can, in reasonable time and with reasonable confidence, rule out impactful partial standardization. To do so, AnaEng estimates the probability of seeing at least

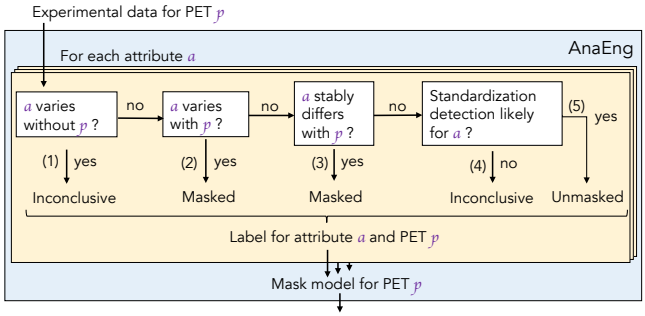


Figure 3: The Analysis Engine (AnaEng) of PETInspector consumes experimental data for a PET and outputs the corresponding mask model.

one changed value given that at least a fraction of them f are being standardized. If this probability is below some threshold α, AnaEng rejects the idea that tool is impactfully standardizing and labels the attribute as unmasked with confidence α. Otherwise, the result is inconclusive since not enough values of the attribute were tested. We use the geometric distribution to estimate likelihood of finding masking given that a fraction f is happening.

4.2 Experiment

Using PETInspector, we performed an initial experiment finding no additional spoofing from AFPETs crossing sessions. Thus, to save time, our main experiment uses only a single session and does not check for the masking of attributes by variation across sessions.

We use ClientSim to simulate six browsing platforms. Three of these are virtual machines running various versions of Linux. We introduce additional changes into these virtual machines to simulate differences in system configurations. Specifically, we install different fonts and browser versions, set up different timezones, and simulate different screen resolutions and languages, The remaining platforms run natively on a Linux desktop, Macbook Pro, and a PC laptop. We perform measurements on Firefox and Chrome browsers. More details on these configurations are in Table 2.

ClientSim drives these experimental browsing platforms to reach FPServer for five reloads of each of the two domain names of FPServer. On each platform, it performs these reloads a total of 28 times: one time each for 26 PETs and one time each for the two baseline browsers. All PETs are left in their default configurations.

4.3 Results

Before commenting on PETs, we make some observations about the baseline browsers. While we did not think of the choice of browsers as affecting the trackability of fingerprints, it turns out that the two browsers have small differences in the attributes shared by them. Aside from the expected difference in the browser name in `User-Agent`, Chrome sets the `cpu class` to *unknown*, the `screen.Depth` to 24, and the `buildID` to *Undefined*, unlike Firefox which reveals different values across browsing platforms. Firefox does not reveal any plugins, while Chrome does, and Chrome's plugins differ across Ubuntu, Debian, and macOS. PETInspector does

Table 2: Configurations of simulated browsing platforms in our main experiment. The last three were regularly used.

#	Type	OS	Addl. Fonts	Resolution	Locale & LANG	Timezone	Browser versions		Notes
							Firefox	Chrome	
1	VM	Ubuntu 16.04	Mordred	450×721×24	ru_RU.UTF-8	GMT+6	56.0	63.0	
2	VM	Debian 8.10	OldLondon	2000×2000×16	de_DE.UTF-8	GMT-3	56.0	63.0	
3	VM	Ubuntu 14.04	(none added)	6000×3000×24+64	ar_SA.UTF-8	GMT-11	56.0	63.0	
4	Local	Ubuntu 16.04	> 40	1920×1080×24	en_EN.UTF-8	GMT-8	56.0	70.0	
5	Local	macOS 10.13	> 145	1440×900×24	en_EN.UTF-8	GMT-8	56.0	70.0	
6	Local	Windows NT 10.0	> 145	1280×720×24	en_EN.UTF-8	GMT-8	56.0 beta	69.0	Touch screen

not find any baseline browser to vary any attributes itself (outcome (1) in Fig. 3).

Turning to PETs, PETInspector produces Table 3, which displays attributes masked or not by AFPETs. We comment on the BLPETs in text. Among the 15 AFPETs, three (Trace, Privacy Extension, and No Enum. Extensions) do not lead to any detectable masking in their default configurations. The remaining 12 AFPETs mask at least one of the collected attributes.

Our experiment also detects undocumented masking of attributes. For example, while Canvas Defender$_C$, Canvas Defender$_F$, CanvasFingerprintBlock, Glove, and CanvasBlocker claim to spoof only the `canvas fingerprint`, we also find them spoofing `webGL` attributes. Similarly, we find undocumented modifications by Brave, Stop Fingerprinting, and TotalSpoof. We also find inconsistencies in the behavior of Brave, Privacy Badger$_C$, Privacy Badger$_F$, HideMyFootprint, and Tor BB, which we discussed in Section 1.

Among the 11 BLPETs, 4 (Disconnect$_F$, Disconnect$_C$, Ghostery$_C$, and Ghostery$_F$) do not lead to any detectable modifications of attributes, 4 (Adblock Plus$_C$, Adblock Plus$_F$, uBlock Origin$_C$, and uBlock Origin$_F$) modify the attribute `adBlock installed`, and 3 (Privacy Badger$_C$, Privacy Badger$_F$, and Tracking Protection) modify Do Not Track attributes. As discussed in the introduction, these BLPETs are not presented as AFPETs, but their modifications can actually make their users more identifiable. Indeed, Privacy Badger was updated in response to our finding.

4.4 Discussion and Limitations

The ranking above may not be suitable for some evaluation goals. For example, some AFPETs were designed to mask a single attribute and does in fact mask it (e.g., Canvas Defender$_C$). Our findings that such AFPETs (or BLPETs) do not mask all attributes should not be interpreted as the PET having a bug. Nevertheless, consumers and advocates seeking effective PETs may find our results useful.

As mentioned above, we may miss some masking of attributes for not testing values that an AFPET standardizes away. Furthermore, we may not detect an AFPET varying an attribute across a boundary that we do not test. Thus, while we can be sure of masking when we find it, we cannot be sure we have found all masking.

Since FPServer extracts fingerprints using first-party scripts, we do not detect masking that is triggered only for third-party scripts.

To an extent, these limitations can be mitigated with more comprehensive experiments using PETInspector. For example, one can modify FPServer to collect additional attributes in both first-party and third-party contexts. Moreover, one can modify ClientSim to detect variations across other boundaries and use more diverse browsing platforms to be more confident about not missing standardization modifications. We will make PETInspector freely available for more extensive experimentation and further development. Our current evaluations demonstrate the benefits of an experimental evaluation method for AFPETs within the current boundaries.

Our experiments may dispute claimed masking (☒ in Table 3) due to the above limitations rather than documentation making spurious claims. AFPETs may mask more attributes when appropriately configured, but users find it difficult to change defaults [37], suggesting our experiments may capture typical use. Next, we perform a manual analysis to understand the effects of configuration and why our results conflict with some AFPETs' documentation.

4.5 Additional Manual Analysis

To address some of the limitations mentioned above, we manually analyze some AFPETs. Specifically, we analyze AFPETs for which we found no evidence of any masking (Trace, Privacy Extension, No Enum. Extensions) and those which made a claim rejected by PETInspector (Trace, Privacy Extension, Stop Fingerprinting, Tor).

PETInspector rejects two claims of masking by Trace. We could not find the source code for Trace, but we installed the extension and manually examined it. Both the documentation and settings panel show canvas fingerprinting being masked by default, despite our studies concluding the opposite. As far as we can tell, Trace really does not mask this attribute despite claiming to. Since running our tests, Trace has been updated from version 1.0.2 to 1.8.6 and it now randomizes the canvas fingerprinting.

As for masking the user-agent, the settings panel of Trace shows that user-agent randomization is off by default, explaining our finding. Turning it on does randomize the user agent.

All of Privacy Extension's masking abilities are off by default. Turning them on does result in standardizing the two attributes in question: the canvas fingerprint and the user-agent.

To analyze No Enum. Extensions, we examined both its source code and documentation. The documentation of No Enum. Extensions only claims to mask plugins, and we found evidence of plugin masking in No Enum. Extensions's source code. PETInspector was inconclusive for this attribute since it was unable to exercise the plugin list for Firefox due to Firefox making the loading of any plugins a manual process. Thus, this instance does not represent a false negative, and instead represents a failure to find a positive.

Table 3: AFPET masks as purported and observed by PETInspector. □ indicates AFPET's documentation purports that the attribute is masked. The remaining symbols represent the possible outputs of PETInspector: + indicates observed masking, × indicates no masking found even when it is likely to detect it, and · indicates inconclusive results. For the results that we manually double checked, we include the outcome of that check as a superscript. Here, × denotes that the PET really does not mask the attribute, + that it does, ×/+ that it is not masked by default but can be with configuration, and ? that the manual analysis was inconclusive. Not shown are attributes that had all inconclusive results and not purported masking (nothing but ·): DNT enabled, IE addBehavior, adBlock installed, h.Connection, h.Dnt, h.Up.-Ins.-Req., indexedDB, math.acosh(1e300), math.asinh(1), math.atanh(05), math.cbrt(100), math.cosh(10), math.expm1(1), math.log1p(10), math.sinh(1), math.tanh(1), **and** openDB.

	Chrome							Firefox							
Attribute	BR	CD_C	CFB	GL	HMF	PE	TR	BI	BL	CD_F	CB	NE	SF	Tor	TO
buildID	·	·	·	·	·	·	·	⊞	⊞	×	×	×	×	⊞	+
canvas fingerprint	⊞	⊞	⊞	⊞	⊞	⊠×/+	⊠×	×	×	⊞	⊞	×	×	⊞	×
cookies enabled	·	·	·	·	·	⊡	·	·	·	·	·	·	·	·	·
cpu class	·	·	·	·	·	·	·	⊞	⊞	×	×	×	+	⊞	+
h.Accept	·	·	·	·	·	⊡	·	·	·	·	·	·	·	⊡	·
h.Accept-Encoding	·	·	·	·	·	·	·	·	·	·	·	·	·	⊡	·
h.Accept-Language	+	×	×	×	×	×	×	×	⊞	×	×	×	×	⊞	×
h.Pragma	·	·	·	·	+	·	·	·	·	·	·	·	·	·	·
h.User-Agent	⊞	×	×	×	⊞	⊠×/+	⊠×	⊞	⊞	×	×	×	×	⊞	⊞
javascript fonts	×	×	×	×	×	×	×	×	×	×	×	×	⊠?	⊞	×
language	+	×	×	×	×	×	×	×	⊞	×	×	×	×	⊞	×
local storage	·	·	·	·	·	⊡	·	·	·	·	·	·	·	·	·
platform	×	×	×	×	×	×	×	⊞	⊞	×	×	×	×	⊠×	+
plugins	⊞	×	×	×	×	×	×	·	·	·	·	⊡+	⊡+	⊡	·
screen.AvailHeight	×	×	×	×	×	×	×	×	×	×	×	×	⊞	⊞	×
screen.AvailLeft	×	×	×	×	×	×	×	×	×	×	×	×	⊞	⊞	×
screen.AvailTop	×	×	×	×	×	×	×	×	×	×	×	×	⊞	⊞	×
screen.AvailWidth	×	×	×	×	×	×	×	×	×	×	×	×	⊞	⊞	×
screen.Depth	·	·	·	·	·	·	·	×	×	×	×	×	⊞	⊞	×
screen.Height	×	×	×	×	×	×	×	×	×	×	×	×	⊞	⊞	×
screen.Left	·	·	·	·	·	·	·	·	·	·	·	·	⊡	⊡	·
screen.Pixel Ratio	×	×	×	×	×	×	×	×	×	×	×	×	⊞	⊞	×
screen.Top	·	·	·	·	·	·	·	·	·	·	·	·	⊡	⊡	·
screen.Width	×	×	×	×	×	×	×	×	×	×	×	×	⊞	⊞	×
session storage	·	·	·	·	·	⊡	·	·	·	·	·	·	·	·	·
timezone	×	×	×	×	×	×	×	×	×	×	×	×	×	⊞	×
touch.event	·	·	·	·	·	·	·	×	×	×	×	×	×	⊞	×
touch.max points	×	×	×	×	×	×	×	·	·	·	·	·	·	·	·
touch.start	·	·	·	·	·	·	·	×	×	×	×	×	×	+	×
webGL.Data Hash	⊞	+	+	+	+	×	×	×	×	+	+	×	×	⊞	×
webGL.Renderer	⊞	+	+	+	+	×	×	×	×	+	×	×	×	⊞	×
webGL.Vendor	⊞	+	+	+	+	×	×	×	×	+	×	×	×	⊞	×

We manually tested Stop Fingerprinting and found that, like No Enum. Extensions, it masks plugins despite PETInspector's inconclusive finding. As for the rejected claim of masking javascript fonts, Stop Fingerprinting may be doing something with the fonts, but not enough to defeat the way FPServer fingerprints them.

Examining Tor BB leads us to believe that a recent update (after Version 7.0.11) accidentally affected its masking of the platform attribute. We also find that during the same time frame, cpuclass and h.User-Agent went from being fully masked to partially masked. We found user complaints about this change in version 8.0a10 [7].

We also confirmed that Privacy Badger did not set the doNotTrack field of the navigator object to match the Dnt header. The code was fixed after we notified the developers of the issue [5].

During our manual examinations, we also looked for artifacts introduced by VMs simulating platforms and found none.

5 HYBRID EVALUATION OF AFPETS

Our experimental method provides a model of how various AFPETs mask fingerprints as well a ranking of AFPETs based on the number of attributes they mask. However, it does not consider how important masking each attribute is. We develop a hybrid method that combines the benefits of the experimental method with an observational method. We start by considering a completely observational method and then discuss how combining it with our experimental method allows us to overcome each of their limitations.

5.1 Sampling

We cannot, in practice, see all the world's browsing platforms and instead must work with a sample. The quality of the metrics computed from the sample depends upon both the nature of the metric and the sample. For example, a random sample will provide a reasonable estimation of the entropy (e.g., [47]). However, estimating the proportion of users in small anonymity sets from even a random sample proves difficult since the length of the tail of the distribution may be unclear from a random sample.

Furthermore, in practice, we must approximate truly random samples of browser platforms from available datasets since we cannot force all users to participate. We do so by using a convenience sample provided to us by the `amiunique` website This sample comprises 25,984 real-world fingerprints collected over a period of 30 days (10/02/2017 to 11/02/2017). Each fingerprint has 32 attributes.

Determining the representativeness of this sample is difficult since it can only be compared to other possibly unrepresentative samples. We compare our sample's distributions to GlobalStat's for desktop users [56]. We find that our sample has a higher proportion of Firefox users (42% vs. 12%) and of Linux users (19% vs. 1%).

5.2 Metrics of Trackability

To measure trackability of fingerprints, we build FPInspector, which consumes a dataset and characterizes how trackable its members are. One such characterization is the anonymity set. An *anonymity set* comprises browsing platforms with identical fingerprints that are, thus, indistinguishable from each other. Thus, the smaller and numerous the anonymity sets, the higher the uniqueness. FPInspector implements various proposed functions (in [15, 63]) over the distribution of anonymity sets for measuring uniqueness.

The first metric which we use to measure uniqueness is entropy. For a set of browser platforms $\mathcal{D} = \{b_i\}_i$, such as those using a particular AFPET, let $\mathcal{D}[id(\cdot)]$ denote the multiset of fingerprints $\{id(b_i)\}_i$ where $id(\cdot)$ is the fingerprinting mechanism. The entropy of these fingerprints is given by

$$\mathsf{ent}(\mathcal{D}[id(\cdot)]) = -\sum_{id_k \in \mathcal{D}[id(\cdot)]} \Pr[id_k] \log_2(\Pr[id_k])$$

where $\Pr[id_k]$ is the probability of observing the fingerprint id_k, which we estimate from the frequency of id_k in $\mathcal{D}[id(\cdot)]$. The higher the entropy, the higher the uniqueness of the fingerprints.

FPInspector also measures the proportion of users in anonymity sets of size less than or equal to 1 ($\mathsf{p}_{-\leq 1}$) and 10 ($\mathsf{p}_{-\leq 10}$). These metrics measure the proportion of browsing platforms hiding in anonymity sets of sizes at most 1 and 10. Higher values of these metrics indicate higher uniqueness of the fingerprints.

FPInspector measures effectiveness of a PET p against fingerprinting mechanism $id(\cdot)$ from the dataset of fingerprints $\mathcal{D}[id(\cdot)]$ in terms of a metric f in $\{\mathsf{ent}, \mathsf{p}_{-\leq 1}, \mathsf{p}_{-\leq 10}\}$ as

$$\mathsf{eff}_{\mathsf{f}}(p, id, \mathcal{D}_p, \mathcal{D}_{\bar{p}}) := \mathsf{f}(\mathcal{D}_{\bar{p}}[id(\cdot)]) - \mathsf{f}(\mathcal{D}_p[id(\cdot)]) \quad (1)$$

where $\mathcal{D}_p$ is a subset of $\mathcal{D}$ using the PET and $\mathcal{D}_{\bar{p}}$ is the rest of $\mathcal{D}$.

5.3 Limitations of Observations Alone

In principle, a highly-context dependent, completely observational method could function by comparing the fingerprints produced by users of each AFPET to determine which are the least trackable. In practice, we face difficulties with obtaining a representative sample of AFPET users and determining which users run which AFPETs.

PET determination. Determining PET use from fingerprints not explicitly containing the information is difficult. This limitation can be overcome by a fingerprinting server designed to collect information about PET use. One approach is to ask visitors about their PETs, but users can be unaware of their own browser's configurations. In some cases, PETs have a distinctive fingerprint that gives away their use, but this would only help us with a subset of PETs. Alternatively, fingerprint collection websites can use automated methods to detect browser extension PETs (e.g., [53, 55]). Unfortunately, our observational data lacks this information.

PET sampling. Even with a fingerprinting server collecting PET information, getting a representative sample of real users with AFPETs to visit the website may be difficult, since there are few AFPET users. This is especially true for new and not yet popular AFPETs. Furthermore, users of AFPETs may be systematically different from users without AFPETs, thereby introducing confounding factors influencing the trackability metrics.

Due to these limitations, we cannot apply FPInspector directly to our dataset. Moreover, the PET sampling limitation may prevent application of this method directly to data collected on even fingerprinting servers designed for PET determination. Thus, we instead use FPInspector in a hybrid evaluation method that avoids the PET determination and sampling problems altogether.

5.4 Overcoming Limitations of Observations

To overcome the difficulty of getting a sample $\mathcal{D}_p$ of browser platforms using a PET p, we construct our own from a sample $\mathcal{D}_{\bar{p}}$ of browser platforms not using p. We then provide both to FPInspector, to evaluate the PET p, as show in Figure 1.

This approach requires that we first get a sample of platforms not using p. We start with the `amiunique` dataset. To convert that dataset of fingerprints into one of platforms, we need a mapping of fingerprints to unique browsing platforms. We approximate this mapping using cookie IDs associated with each fingerprint, similarly to Eckersley [15]. In the dataset, 21,395 fingerprints have a cookie associated with them, of which, 18,295 are unique.

To obtain $\mathcal{D}_{\bar{p}}[id(\cdot)]$, we sanitize the dataset by removing fingerprints with obvious signs of PET use, specifically those with JavaScript disabled and illegitimate screen resolutions. Additionally, we only retain fingerprints from desktop browsers (with Windows, Mac, or Linux OSes) since we only study PETs for desktops. These sanitizations leave 9,493 Chrome and 6,516 Firefox browser fingerprints. We find that these fingerprints reveal 13.002 and 12.359 bits

Table 4: Metrics of trackability (from Section 5.2) for AFPETs

PET	ent	$p_{-\leq 1}$	$p_{-\leq 10}$
Chrome PETs			
no mask	13.002	0.892	0.983
base mask, Privacy Extension, Trace	12.914	0.829	0.982
Canvas Defender$_C$, CFB, Glove	12.306	0.641	0.893
HideMyFootprint	11.77	0.497	0.825
Brave	8.108	0.072	0.262
Firefox PETs			
no mask	12.359	0.875	0.96
base mask, No Enum. Extensions	12.177	0.797	0.949
Blend In, TotalSpoof	12.049	0.747	0.936
CanvasBlocker	12.002	0.7	0.941
Blender	11.875	0.678	0.924
Stop Fingerprinting	11.778	0.726	0.919
Canvas Defender$_F$	11.263	0.483	0.833
Tor BB	4.766	0.01	0.038

of entropy for Chrome and Firefox respectively. These and other metrics are presented in Table 4 in the 'no mask' row.

The mask model from the experimental method provides a way to transform these original fingerprints. We apply the mask model $\hat{p}$ of an PET p produced by PETInspector to the sample $\mathcal{D}_{\bar{p}}$ of platforms without a PET to generate a sample of fingerprints $\mathcal{D}_{\hat{p}}[id(\cdot)]$. This generated sample estimates what the original fingerprints would had looked like had the platforms used the PET p. We use FPInspector to calculate the trackability metrics of the modified fingerprints and unmodified fingerprints. By comparing the metrics of the original and $\hat{p}$-modified fingerprints, we estimate the effectiveness of p.

Of the 49 original attributes, PETInspector provides conclusive characterization for 18 attributes on Chrome and 20 attributes on Firefox. Of these, 12 appear in the `amiunique.org` dataset. For a given PET, we mask these 12 attributes according to the model generated by PETInspector and fully mask the remaining 16 attributes for which the experiment is inconclusive. By fully masking inconclusive attributes, we overestimate the effectiveness of PETs. Thus, we generate PET-modified fingerprints (i.e., $\mathcal{D}_{\hat{p}}[id(\cdot)]$) from the original fingerprints to measure the effectiveness of 15 AFPETs.

5.5 Results

We present the trackability metrics from Section 5.2 in Table 4. The original fingerprints reveal 13.002 and 12.359 bits of entropy for Chrome and Firefox respectively. Applying a *base mask* of all inconclusive attributes reduces them to 12.914 and 12.177 bits.

Our evaluations reveal that all AFPETs but Brave and Tor BB reveal over 11 bits of entropy and hence are marginally better than not using any AFPET at all. For these AFPETs, fewer than 20% of the fingerprints are in anonymity sets of size greater than 10. Brave does better, leaking just over 8 bits of entropy and having over 70% of fingerprints in anonymity sets of size greater than 10. Tor BB performs best since it modifies all the 12 attributes we consider.

5.6 Remaining Limitations

While the hybrid method helps us perform a fine-grained evaluation of AFPETs with few users, it inherits some limitations of the methods on which it builds. From the observational method come the limitations that samples may be biased and that no one metric fully captures the quality of an AFPET. From the experimental method, it inherits the approximate nature of mask models.

In particular, our analysis overestimates the effectiveness of all AFPETs, since we assume any modifications of an attribute by an AFPET renders that attribute useless to a tracker. This may not be the case. For example, Brave spoofs the `User-Agent` and the `Accept-Language` headers to different values than Chrome. Similarly, Tor BB also reveals spoofed values of screen resolution.

We can carry out a tighter evaluation by considering a tracker which can take advantage of the spoofed values. This evaluation requires knowledge of how an AFPET spoofs the attribute. For Tor BB, we perform a manual code analysis to determine how exactly Tor BB deals with screen resolution attributes.[2] We rerun the hybrid analysis on a hand crafted mask model capturing this behavior instead of using the rough model produced by PETInspector. This provides a tighter evaluation for Tor BB that will serve as the basis for our analysis in Section 7.

Finally, the above evaluations are performed on the same set of fingerprints and applies the mask to every fingerprint in the dataset, simulating total adoption of the AFPET. This approach is appropriate evaluations with a long-term prospective, such as selecting an AFPET to fund, since a properly promoted AFPET could become nearly universal in the future. However, those looking to select a AFPET for usage today should be concerned with the number of users each AFPET has since it will affect the size of the anonymity set the AFPET produces. In the next section, we consider a modification of the above method for dealing with this issue.

6 ADJUSTING FOR NUMBER OF USERS

To observe the consequences of having user bases of different sizes, we also evaluate the AFPETs taking into account their popularity. Ideally, we would do this by analyzing fingerprints of all the users of an AFPET. However, not having access to this set of fingerprints, we simulate them by drawing random samples of fingerprints from the `amiunique.org` dataset of size equal to the number of AFPET users and estimate uniqueness metrics on the samples.

Table 1 displays the number of users of each AFPET in our list as of Dec. 2017. The popularity of extensions were obtained from the Firefox add-on library [42] and the Chrome extensions webstore [24]. Tor's popularity was obtained from Tor Metrics [58]. For AFPETs with an undisclosed number of users, such as Brave and Tracking Protection, we are unable to perform this evaluation.

We also do not perform these evaluations for AFPETs with a user base greater than 17,109 (like Tor BB, Canvas Defender$_C$ and CanvasBlocker), since we cannot draw a sample from our dataset of sufficient size. Attempting to draw such a sample by allowing the same fingerprint to be sampled multiple times will overestimate the effectiveness of the PET since such repeats will surely be in the same anonymity set even for PETs that do nothing.

[2] https://gitweb.torproject.org/tor-browser.git/commit/?h=tor-browser-45.8.0esr-6.5-2&id=7b3e68bd7172d4f3feac11e74c65b06729a502b2.

Table 5: Metrics of trackability (from Section 5.2) for AFPETs on samples scaled according to their popularity

PET	#users	ent	$p_{_\leq 1}$	$p_{_\leq 10}$
Chrome PETs				
HideMyFootprint	177.0	7.343	0.901	1.000
Glove	342.0	8.277	0.886	1.000
CanvasFingerprintBlock	7630.0	11.559	0.313	0.899
Firefox PETs				
TotalSpoof	265.0	7.904	0.889	1.000
Blend In	858.0	9.401	0.777	0.983
Stop Fingerprinting	1754.0	9.994	0.641	0.939
Blender	1816.0	10.200	0.614	0.960
Canvas Defender$_F$	5274.0	10.656	0.252	0.845

For all other AFPETs, we compute the mean of the trackability metrics from 100 random samples. Table 5 displays the effectiveness metrics for these AFPETs, sorted according to the entropy. We can see that CanvasFingerprintBlock scores better than HideMyFootprint due to its high popularity, contrary to the original evaluations in Table 4. We also see that the effectiveness of tools with identical effects increases with popularity. For example, TotalSpoof and Blend In both identically modify 12 attributes, but Blend In is more effective than TotalSpoof due to its popularity.

7 APPLICATION: INFORMING AFPET DESIGN

With the ability to accept handcrafted mask models, our hybrid method can help AFPET developers make an informed choice while designing AFPETs. By measuring the effectiveness of hypothetical AFPET designs, developers can compare masking strategies to balance utility with trackability. We carry out such an exploration of alternate designs of Tor BB that mask attributes differently.

Tor BB leaks some information about the screen resolution by only partially standardizing it. Specifically, it resizes new browser windows in quanta (step/bucket sizes) of 200×100 pixels, while capping the window size at 1000×1000 pixels, and uses the client content window size as screen dimensions [48]. As a result all Tor BB users get placed into one of 50 anonymity sets based on the revealed screen dimensions, as long as they do not change the window dimensions manually. We explore the impact of the cap and quanta parameters on the effectiveness of Tor BB.

We use the number of unutilized screen pixels due to a spoofing strategy as a measure of utility loss. We measure two variants: the total number of unutilized pixels (average absolute loss), and the number of unutilized pixels as a percentage of the available pixels (average percentage loss). Increasing the cap parameters and decreasing the quanta parameters reduces this loss.

With the Firefox fingerprints in the `amiunique.org` dataset, we explore parameter settings with the goal of finding a strategy that reduces the utility loss while increasing the effectiveness. We consider alternative cap widths of 1000, 1350, 1550, and 1600 since a higher percentage of fingerprints (25%, 47%, and 51% respectively) have screen widths less than these caps. We retain the cap height of 1000 pixels as more than 50% of the fingerprints remain below that cap. We exhaustively search for all 10,201 quanta in the range 200×100 to 300×200 for all three cap parameters. We set an upper bound of 300×200 as the loss may be too high for low-resolution displays for very high quanta parameters. We find 786 and 291 quanta parameters for cap widths of 1350 and 1550 respectively for which the losses are lower than Tor BB's, but the effectiveness is higher. We display strategies with the least quanta parameters in Table 6. As we increase the cap width to 1600, none of the quanta parameters lead to a higher measure of effectiveness than Tor BB.

Table 6: Comparison of effectiveness and utility-loss of Tor BB's original spoofing strategies with alternate strategies

Cap	Quanta	ent	$p_{_\leq 1}$	$p_{_\leq 10}$	Abs. Loss	% Loss
1000×1000	200×100	2.902	0.001	0.010	870k	50.3%
1350×1000	200×193	2.715	0.001	0.009	729k	42.6%
1350×1000	269×160	2.901	0.000	0.009	728k	42.3%
1550×1000	222×197	2.899	0.000	0.009	666k	40.3%
1550×1000	295×160	2.882	0.000	0.010	636k	37.2%

8 CONCLUSION

We end with some suggestions for AFPET developers and evaluators. We recommend that developers address any attribute that PETInspector flags as unmasked. The entropy results from our hybrid method can aid in determining the order in which to address various unmasked attributes. Given our experimental results, we expect this task will keep the developers of most AFPETs busy. Next, they might want to consider any attributes that PETInspector labeled as inconclusive. After addressing these attributes, they can consider improving how an AFPET spoofs an attribute. As shown in Section 7, not all spoofing is equal. Developers should consider using Tor BB as a starting point for their development and carefully consider the default settings of their AFPET.

The set of fingerprintable attributes are open-ended and will never be fully enumerated, but new attributes can be added to our tools. AFPET evaluators should keep in mind that any one-time evaluation of PETs will quickly become out of date, necessitating the use of automated tools like ours. We encourage developers and advocates (e.g., the EFF) to use automated tools to regularly test the trackability of PETs. Our tool can fill this need, and to this end we open-source the tool here:

https://github.com/tadatitam/pet-inspector

ACKNOWLEDGEMENTS

We thank Pierre Laperdrix for providing us a dataset of real-world fingerprints from amiunique.org, and Milan Ganai for investigating how to automate the use of PETs on Windows. We thank Lay Kuan Loh and Zheng Zong for assistance in exploring the application of information flow experiments to evaluate PETs. We thank Anupam Datta for discussions about this work. We gratefully acknowledge funding support from the National Science Foundation (Grants 1514509 and 1704985). The opinions in this paper are those of the authors and do not necessarily reflect the opinions of any funding sponsor or the United States Government.

REFERENCES

[1] Absolute Double. 2017. HideMyFootprint: Protect your privacy. https://hmfp.absolutedouble.co.uk. (2017). Accessed Dec. 25, 2017.

[2] Absolute Double. 2018. Trace: Browse online without leaving a Trace. https://absolutedouble.co.uk/trace/. (2018). Accessed Jan. 12, 2018.

[3] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. 2014. The web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 674–689.

[4] Gunes Acar, Marc Juarez, Nick Nikiforakis, Claudia Diaz, Seda Gürses, Frank Piessens, and Bart Preneel. 2013. FPDetective: dusting the web for fingerprinters. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 1129–1140.

[5] Alexei "ghostwords". 2018. Support navigator.doNotTrack. Pull request #1861 for the EFForg/privacybadger project on GitHub: https://github.com/EFForg/privacybadger/pull/1861. (July 2018).

[6] Andrew. 2017. Scriptsafe: andryou. https://www.andryou.com/scriptsafe/. (2017). Accessed Dec. 25, 2017.

[7] Anonymous. 2018. Comment 276687 on "New Release: Tor Browser 8.0a10". Tor Blog: https://blog.torproject.org/comment/276424#comment-276424. (Aug. 2018). See responses as well.

[8] appodrome.net. 2017. CanvasFingerprintBlock: Chrome Web Store. https://chrome.google.com/webstore/detail/canvasfingerprintblock/ipmjngkmngdcdpmgmiebdmfbkcecdndc?hl=en. (2017). Accessed Dec. 25, 2017.

[9] Blinded for anonymization. 2018. A bug report to Brave. (Jan. 2018).

[10] Blinded for anonymization. 2018. Another bug report to Brave. (Jan. 2018).

[11] Blinded for anonymization. 2018. A bug report to Privacy Badger. (Jan. 2018).

[12] Brave Browser. 2017. Fingerprint Protection Mode. https://github.com/brave/browser-laptop/wiki/Fingerprinting-Protection-Mode. (2017). Accessed Dec. 19, 2017.

[13] Yinzhi Cao, Song Li, and Erik Wijmans. 2017. (Cross-)Browser Fingerprinting via OS and Hardware Level Features. In *24th Annual Network and Distributed System Security SymposiumNDSS*. http://www.yinzhicao.org/TrackingFree/crossbrowsertracking_NDSS17.pdf

[14] Disconnect. 2017. Disconnect. https://disconnect.me. (2017). Accessed Jan. 12, 2017.

[15] Peter Eckersley. 2010. How unique is your web browser?. In *Privacy Enhancing Technologies*, Vol. 6205. Springer, 1–18.

[16] Electronic Frontier Foundation. 2017. Panopticlick. https://panopticlick.eff.org. (2017). Accessed Dec 12, 2017.

[17] Electronic Frontier Foundation. 2017. Privacy Badger. https://www.eff.org/privacybadger. (2017). Accessed Jan. 13, 2017.

[18] Steven Englehardt and Arvind Narayanan. 2016. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1388–1401.

[19] eyeo GmbH. 2017. Adblock Plus: Surf the web without annoying ads! https://adblockplus.org. (2017). Accessed Dec. 27, 2017.

[20] Amin FaizKhademi, Mohammad Zulkernine, and Komminist Weldemariam. 2015. FPGuard: Detection and prevention of browser fingerprinting. In *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 293–308.

[21] David Fifield and Serge Egelman. 2015. Fingerprinting web users through font metrics. In *International Conference on Financial Cryptography and Data Security*. Springer, 107–124.

[22] fonk. 2017. TotalSpoof Add-on Homepage. http://fonk.wz.cz/totalspoof. (2017). Accessed Dec. 25, 2017.

[23] Cliqz International GmbH. 2017. Ghostery Makes the Web Cleaner, Faster and Safer! https://www.ghostery.com. (2017). Accessed Dec. 27, 2017.

[24] Google. 2017. Chrome web store. https://chrome.google.com/webstore/category/extensions. (Dec. 2017).

[25] Gábor György Gulyás, Dolière Francis Somé, Nataliia Bielova, and Claude Castelluccia. 2018. To Extend or Not to Extend: On the Uniqueness of Browser Extensions and Web Logins. In *Proceedings of the 2018 Workshop on Privacy in the Electronic Society (WPES'18)*. ACM, New York, NY, USA, 14–27. DOI: http://dx.doi.org/10.1145/3267323.3268959

[26] Raymond Hill. 2015. uBlock and others: Blocking ads, trackers, malwares. https://github.com/gorhill/uBlock/wiki/uBlock-and-others%3A-Blocking-ads%2C-trackers%2C-malwares. (May 2015). Accessed July 5, 2017.

[27] Raymond Hill. 2017. uBlock Origin: An efficient blocker for Chromium and Firefox. https://github.com/gorhill/uBlock. (2017). Accessed Dec. 27, 2017.

[28] Muhammad Ikram, Hassan Jameel Asghar, Mohamed Ali Kaafar, Anirban Mahanti, and Balachandar Krishnamurthy. 2017. Towards seamless tracking-free web: Improved detection of trackers via one-class learning. *Proceedings on Privacy Enhancing Technologies* 2017, 1 (2017), 79–99.

[29] InformAction. 2017. NoScript: JavaScript/Java/Flash blocker for a safer Firefox experience! https://noscript.net. (2017). Accessed Dec. 27, 2017.

[30] kkapsner. 2017. CanvasBlocker: A Firefox Plugin to block the canvas-API. https://github.com/kkapsner/CanvasBlocker/. (2017). Accessed Dec. 25, 2017.

[31] Georgios Kontaxis and Monica Chew. 2015. Tracking protection in Firefox for privacy and performance. *arXiv preprint arXiv:1506.04104* (2015).

[32] Balachander Krishnamurthy and Craig E Wills. 2006. Generating a privacy footprint on the internet. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. ACM, 65–70.

[33] Pierre Laperdrix. 2017. Fingerprint Central. https://fpcentral.irisa.fr/. (2017). Accessed Oct 31, 2017.

[34] Pierre Laperdrix, Benoit Baudry, and Vikas Mishra. 2017. FPRandom: Randomizing core browser objects to break advanced device fingerprinting techniques. In *9th International Symposium on Engineering Secure Software and Systems (ESSoS 2017)*.

[35] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. 2015. Mitigating browser fingerprint tracking: multi-level reconfiguration and diversification. In *Proceedings of the 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. IEEE Press, 98–108.

[36] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. 2016. Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints. In *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 878–894.

[37] Pedro Leon, Blase Ur, Richard Shay, Yang Wang, Rebecca Balebako, and Lorrie Cranor. 2012. Why Johnny can't opt out: a usability evaluation of tools to limit online behavioral advertising. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 589–598.

[38] Jonathan R Mayer and John C Mitchell. 2012. Third-party web tracking: Policy and technology. In *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 413–427.

[39] meh. 2017. Blender: Blend in the crowd by faking to be the most common Firefox browser version, operating system and other stuff. https://github.com/meh/blender. (2017). Accessed Dec. 25, 2017.

[40] Georg Merzdovnik, Markus Huber, Damjan Buhov, Nick Nikiforakis, Sebastian Neuner, Martin Schmiedecker, and Edgar Weippl. 2017. Block me if you can: A large-scale study of tracker-blocking tools. In *Proceedings of the 2nd IEEE European Symposium on Security and Privacy (IEEE EuroS&P)*.

[41] Keaton Mowery and Hovav Shacham. 2012. Pixel perfect: Fingerprinting canvas in HTML5. *Proceedings of W2SP* (2012), 1–12.

[42] Mozilla. 2017. Firefox Add-ons. https://addons.mozilla.org/en-US/firefox/. (Dec. 2017).

[43] Multiloginapp. 2017. How Canvas Fingerprint Blockers Make You Easily Trackable. https://multiloginapp.com/how-canvas-fingerprint-blockers-make-you-easily-trackable/. (2017). Accessed Dec 19, 2017.

[44] Net-Comet. 2017. Glove: Chrome Web Store. https://chrome.google.com/webstore/detail/glove/abdgoalibdacpnmknnpkgnfllphboefb?hl=en. (2017). Accessed Dec. 25, 2017.

[45] Nick Nikiforakis, Wouter Joosen, and Benjamin Livshits. 2015. Privaricator: Deceiving fingerprinters with little white lies. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 820–830.

[46] NiklasG. 2017. Stop Fingerprinting: Add-ons for Firefox. https://addons.mozilla.org/en-US/firefox/addon/stop-fingerprinting/. (2017). Accessed Dec. 25, 2017.

[47] Liam Paninski. 2003. Estimation of entropy and mutual information. *Neural computation* 15, 6 (2003), 1191–1253.

[48] Mike Perry, Erinn Clark, Steven Murdoch, and Georg Koppen. 2017. The Design and Implementation of the Tor Browser. https://www.torproject.org/projects/torbrowser/design/#privacy. (2017). Accessed Jul 21, 2017.

[49] Reşat. 2017. Blend In: Add-ons for Firefox. https://addons.mozilla.org/en-US/firefox/addon/blend-in/. (2017). Accessed Dec. 25, 2017.

[50] Franziska Roesner, Tadayoshi Kohno, and David Wetherall. 2012. Detecting and Defending Against Third-party Tracking on the Web. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation (NSDI'12)*. USENIX Association, Berkeley, CA, USA, 12–12. http://dl.acm.org/citation.cfm?id=2228298.2228315

[51] Samy Sadi. 2017. No Enumerable Extensions: Firefox addon that lets you hide installed extensions and avoid being fingerprinted based on them. https://github.com/samysadi/no-enumerable-extensions. (2017). Accessed Jan. 13, 2017.

[52] Sagar Shivaji Salunke. 2014. *Selenium Webdriver in Python: Learn with Examples* (1st ed.). CreateSpace Independent Publishing Platform, USA.

[53] Iskander Sanchez-Rola, Igor Santos, and Davide Balzarotti. 2017. Extension Breakdown: Security Analysis of Browsers Extension Resources Control Policies. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC, 679–694. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/sanchez-rola

[54] Martin Springwald. 2017. Privacy-Extension-Chrome: Provides Privacy for Chrome. https://github.com/marspr/privacy-extension-chrome. (2017). Accessed Dec. 25, 2017.

[55] Oleksii Starov and Nick Nikiforakis. 2017. Xhound: Quantifying the fingerprintability of browser extensions. In *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 941–956.

[56] StatCounter. 2018. StatCounter Global Stats. http://gs.statcounter.com/. (2018). Accessed Feb. 12, 2018.

[57] Mozilla Support. 2017. Tracking Protection. https://support.mozilla.org/en-US/kb/tracking-protection. (2017). Accessed Dec. 27, 2017.
[58] The Tor Project. 2017. Users. Tor Metrics page: https://metrics.torproject.org/userstats-relay-country.html. (Dec. 2017).
[59] Christof Ferreira Torres, Hugo Jonker, and Sjouke Mauw. 2015. FP-Block: usable web privacy by controlling browser fingerprinting. In *European Symposium on Research in Computer Security*. Springer, 3–19.
[60] Hamilton Ulmer. 2010. Browsing Sessions. Mozilla's Blog of Metrics: https://blog.mozilla.org/metrics/2010/12/22/browsing-sessions/. (Dec. 2010).
[61] Antoine Vastel, Pierre Laperdrix, Walter Rudametkin, and Romain Rouvoy. 2018. Fp-Scanner: The Privacy Implications of Browser Fingerprint Inconsistencies. In *27th USENIX Security Symposium (USENIX Security 18)*. USENIX Association, Baltimore, MD, 135–150. https://www.usenix.org/conference/usenixsecurity18/presentation/vastel
[62] Jon Watson. 2008. VirtualBox: Bits and Bytes Masquerading As Machines. *Linux J.* 2008, 166 (Feb. 2008). http://dl.acm.org/citation.cfm?id=1344209.1344210
[63] Ting-Fang Yen, Yinglian Xie, Fang Yu, Roger Peng Yu, and Martin Abadi. 2012. Host Fingerprinting and Tracking on the Web: Privacy and Security Implications.. In *NDSS*.

Heterographic Pun Recognition via Pronunciation and Spelling Understanding Gated Attention Network

Yufeng Diao
School of Computer Science and Technology
Dalian University of Technology, Inner Mongolia University for Nationalities, Dalian, Liaoning, China, diaoyufeng@mail.dlut.edu.cn

Hongfei Lin†
School of Computer Science and Technology
Dalian University of Technology, Dalian, Liaoning, China, hflin@dlut.edu.cn

Liang Yang
School of Computer Science and Technology
Dalian University of Technology, Dalian, Liaoning, China, liang@dlut.edu.cn

Xiaochao Fan
School of Computer Science and Technology
Dalian University of Technology, Xinjiang Normal University, Dalian, Liaoning, China, fxc1982@mail.dlut.edu.cn

Di Wu
School of Computer Science and Technology
Dalian University of Technology, Dalian, Liaoning, China, wudi@dlut.edu.cn

Dongyu Zhang
School of Software
Dalian University of Technology, Dalian, Liaoning, China, zhangdongyu@dlut.edu.cn

Kan Xu
School of Computer Science and Technology
Dalian University of Technology, Dalian, Liaoning, China, xukan@dlut.edu.cn

ABSTRACT

Heterographic pun plays a critical role in human writing and literature, which usually has a similar sounding or spelling structure. It is important and difficult research to recognize the heterographic pun because of the ambiguity. However, most existing methods for this task only focus on designing features with rule-based or machine learning methods. In this paper, we propose an end-to-end computational approach – Pronunciation Spelling Understanding Gated Attention (PSUGA) network. For pronunciation, we exploit the hierarchical attention model with phoneme embedding. While for spelling, we consider the character-level, word-level, tag-level, position-level and contextual-level embedding with attention model. To deal with the two parts, we present a gated attention mechanism to control the information integration. We have conducted extensive experiments on SemEval2017 task7 and Pun of the Day datasets. Experimental results show that our approach significantly outperforms state-of-the-art methods.

† Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13, 2019, San Francisco, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313505

CCS CONCEPTS

• Applied computing • Sociology

KEYWORDS

Heterographic Pun Recognition, Pronunciation, Spelling, Gated Attention

ACM Reference format:

Yufeng Diao, Hongfei Lin, Liang Yang, Xiaochao Fan, Di Wu, Dongyu Zhang and Kan Xu. 2019. Heterographic Pun Recognition via Pronunciation and Spelling Understanding Gated Attention Network. In *Proceedings of WWW '19: The Web Conference (WWW '19), May 13, 2019, San Francisco, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3308558.3313505

1 Introduction

A pun, also called paronomasia, is a form of word play that suggests two or more meanings by exploiting multiple meanings of a word or by replacing a similar sounding word for an intended humorous or rhetorical effect (Tristan and Mladen, 2016). They are widely used in written and spoken literature, intended as jokes. From Shakespeare's works to modern advertisement catchphrases (Tanaka, 1992), puns have been widely applied as a humorous device. In literature, speeches and slogans, puns are also standard rhetorical ploys where can also be used non-humorously. Both humorous and non-humorous

pun, the special form of ambiguity, have been the theme of extensive and attractive works for creating statements with ambiguous and distinct meanings to arise the attractive effect.

The study of puns recognition can be seen as a respectable research topic in traditional linguistics and the cognitive sciences in Nature Language Processing (NLP). Many scholars study on classifying puns by the semantic relationship between the different meaning (polysemy), or similarly pronounced sound (phonology). For example, Redfern (1987) categorizes pun into homographic pun and heterographic pun, which exploits distinct meanings of the same written word and distinct meanings of the similar spoken word, respectively. There are many relevant research rely on this classification system. Our work also think that a pun is consisted of homographic and heterographic pun.

Homographic pun and heterographic pun both have different meanings to produce a humorous or rhetorical effect. Whereas these two types of puns have own special characteristics.

Exp.1 "Getting rid of your boat for another could cause a whole raft of problems."

A homographic pun is characterized by two or more meanings of a word, each of which leads to a different but valid interpretation. In an example 1 shown above, the word "raft" is the pun word which denoting "raft" as a batch or a type of boat.

Exp.2 "A dentist hates having a bad day at the orifice."

A heterographic pun relies on a different kind of senses between two words which have the similar sound and rhyme, or nearly spelling. In an example 2 shown that the pun word "orifice" has a similarly sounding and spelling word "office", which is associated with the previous word "dentist".

In this study, we focus on heterographic pun, defined as puns containing words that sound identical or similar spell to other words, because these puns widely used everywhere and easily obtain in the current works (Tristan and Mladen, 2016). However, the study of heterographic pun recognition in the exiting works has a challenge because of the confused and ambiguous meanings.

To settle the problem, we propose an automatic computational end-to-end Pronunciation Spelling Understanding Gated Attention network (PSUGA) to recognize heterographic pun. Our model takes pronunciation and spelling into account for heterographic recognition. Based on the experiments, the results show that our work achieves the state-of-the-art performance compared to existing methods, which use manual rules or other machine learning models. Here, our contributions are listed as follows.

(i) We apply the hierarchical attention mechanism to understand the latent semantic phoneme representation by CMU pronunciation dictionary for solving the pronunciation of heterographic pun.

(ii) We propose a neural attention network to learn the spelling relationship by character-level, word-level, tag-level, position-level and contextual embeddings for heterographic pun classification.

(iii) We propose the gated attention strategy to exploit the combination of the pronunciation and spelling which is useful for this task. Experimental results on the datasets of SemEval2017 task 7 and Pun of the Day demonstrate that our method outperforms several strong baselines.

2 Related Work

Puns have been discussed in rhetorical and literary criticism since ancient times and have increasingly become a respectable research topic in recent years. However, it is surprising that puns have attracted little attention for computational linguistics and natural language processing (Tristan and Mladen, 2016). In this section, we will introduce the prior work about puns and recognition methods.

Puns contain semantic ambiguous, sentence pattern, similar sound these own characteristics. Hempelmann (2008) studied the automatic recognition of pun target words and humor text generation. Recently, Miller and Gurevych (2015) proposed methods for homographic pun to detect the ambiguous multiple meanings for the word sense disambiguation. Tristan and Mladen (2016) applied a computational method for the detecting of puns in running text and for the isolation of the intended meanings, and evaluated WSD-inspired systems in a dedicated pun identification task.

Some researchers studying puns tend to have a phonological or syntactic puns rather than semantic puns. Zwicky and Zwicky (1986) provided an analysis of the properties of paronomasia puns, especially with regard to the phonological segments. Binsted et al. (1996) applied the pronounciation dictionary (Robinson 1996) and designed the similar sound rules to generate the word pairs of similar pronounce for puns generation. Kao et al. (2015) proposed a computational model of linguistic humor in puns and two information-theoretic measures, ambiguity and distinctiveness, which derived from a simple model of sentence processing. Aaron et al. (2016) considered the semantic relationship between pun and its phonologically similar target, and leveraged automatically learned phone edit probabilities. N-Hance (Sevgili et al., 2017) assumed every pun has a particularly strong association with exactly one other word in the context, and calculated the PMI between every pair of words. Idiom Savant (Doogan et al, 2017) used Google n-grams, word2vec, and CMU Pronouncing Dictionary to recognize heterographic pun. JU_CSE_NLP (Pramanick and Das, 2017) trained a hidden Markov model and cyclic dependency network, using features from a part-of-speech tagger and a syntactic parser. ECNU (Xiu et al., 2017) applied a supervised approach to pun detection by using WordNet and word2vec embeddings. These above methods mostly design useful features, and adopt rule-based or machine learning methods to recognize the heterographic pun. Here we use an end-to-end computable network for recognizing heterographic pun, combined with CMU pronunciation dictionary and spelling relationship.

Compared with pun recognition, pun generation has attracted little attention in the past decades. Hempelmann et al. (2003) modeled the factors to generate the pun and obtain the

measures for pun evaluation based on the theory. Richie et al. (2005) generated pun based on the analysis of humor language. Hong et al. (2009) used to automatic extract the semantic patterns for puns generation and store the knowledge base in template form by using phonetic and semantic linguistic resources.

Some researchers focus on the application of pun in humor. Yang D et al. (2015) recognized the humor and humor anchor on the Pun of the Day dataset by incongruity, ambiguity, interpersonal effect, and phonetic style these useful semantic features. Taylor and Mazlack (2004) proposed an N-gram method with the syntactic context to detect puns for humorous effect in English jokes. Then Taylor (2009) carried out humor recognition depending on Ontological Semantics in the transforming content. Bertero and Fung (2016) used LSTM model to recognize humor based on the phonetic and textual features with the Big Bang Theory Corpus.

Bahdanau et al. (2014) proposed the attention mechanism to select the reference words in an initial language before translating other language for solving the machine translation problem. Yang et al. (2016) applied attention mechanism to settle the document-level classification. There are also some NLP tasks which employing attention mechanism, including machine comprehension (Seo et al., 2017; Yu et al., 2018), text classification (Du et al., 2017; Tan et al., 2018; Shen et al., 2018; Xu et al., 2018; Shen et al., 2018), question answering (Lu et al., 2016; Gui et al., 2017), generation (Zhang et al., 2018; Kim et al., 2018; Fan et al., 2018). In conclusion, attention model can enable to mining the latent and semantic information for improving the performance of NLP tasks.

Hence, we propose an end-to-end approach Pronunciation Spelling Understanding Gated Attention network (PSUGA) for recognizing heterographic pun, combined with CMU pronunciation dictionary, different levels of embedding and attention mechanism.

3 Methods

In this section, we propose our model as Pronunciation Spelling Understanding Gated Attention network (PSUGA). This model can improve the performance by considering both the phonological semantic representation and spelling relationship with character-level, word-level, tag-level, positon-level and contextual embeddings for heterographic puns recognition.

The overall architecture of our model is shown in Figure 1. It consists of two main components: a hierarchical attention convolutional neural network (CNN) as Pronunciation Understanding network, and a multi-level embedding attention network as Spelling Understanding network for text. These two components are combined by a gated attention strategy to recognize the heterographic pun. We describe the details of these two components in the following subsections.

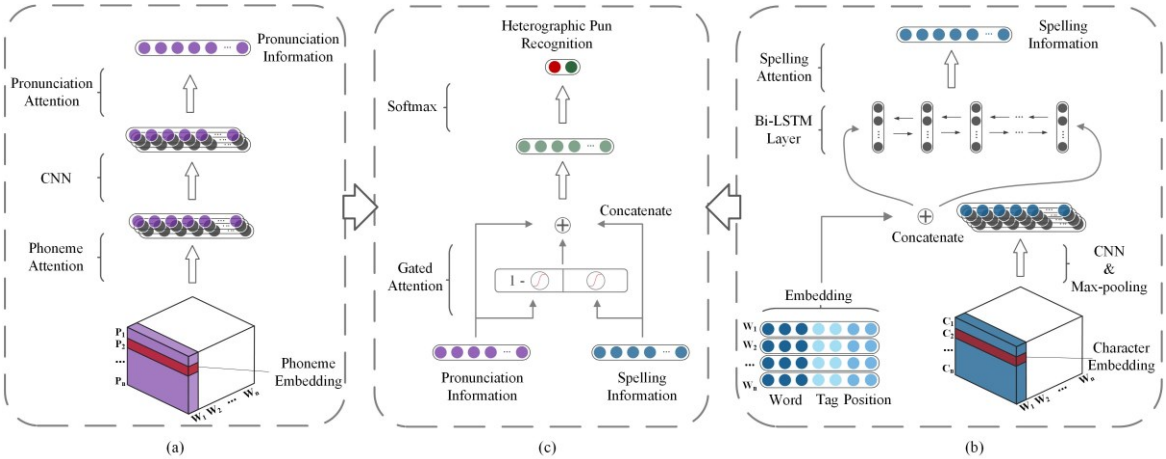


Figure 1: The Framework of Pronunciation Spelling Understanding Gated Attention Network (PSUGA)

3.1 Pronunciation Understanding Network(PUN)

Heterographic pun are mostly fabricated by replacing phonetically similar words. For example, "*What did the grape when it got stepped on? Nothing but it let out a little whine.*" The pun word is "whine", while the similar pronunciation word is "wine" which is associated with the previous word "grape". This example illustrates that one of the main challenges for heteographic pun recognition is identifying multi word expressions between the phonetically similar words.

In order to obtain phoneme representation, we introduce CMU pronouncing dictionary[1] (CMUdict), which developed by Carnegie Mellon University for designing speech synthesizer. It contains 134000 entries. Each entry is pronounced by Arpabet, which is composed of 23 vowels and 31 consonants. For example, the word "Hello" is pronounced as "HH AHO L OW1", where the number indicates stress.

A word can contain a number of phonetic symbols in CMUdict. We use all the pronunciation of an entry for the

[1] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

speech extension, and match any pronunciation as the speech extension of a word. There are two types of speech extensions: the same pronunciation and the similar pronunciation. The former is to look up the word of same sound with this entry in CMUdict, while the latter is to make statistics on pronunciation substitution between original word and target word. Here, we apply the substitution matrix between vowels and vowels, consonants and consonants proposed by Jaech et al. (2016). If the pronunciation is found in CMUdict after phoneme substitution, it can be used as the speech extensions of the original word.

Phoneme Embedding Layer. Phoneme embedding layer is responsible for mapping each word to a high-dimensional vector space. Let $\{x_1,..., x_M\}$ represents the words in the input sentence, where $x_i=\{p_1,...,p_N\}$ represents the phonemes of the word x_i and $p_N \in R^d$ is the d dimensional vector of the N-th phoneme in the target word x_i.

Phoneme Attention Mechanism. Traditional model cannot capture the important parts in sentences. In order to address this problem, we design an attention mechanism that enables model to concentrate on salient phonemes with respect to a specific word. To make full use of pronunciation information, this model uses the phoneme embedding as input to extract phoneme-specific attention signal. The formulas are as follows.

$$u_t = \tanh(w \cdot p_a + b)$$

$$\alpha_{ti} = \frac{\exp(u_{pi}^T p_{vi})}{\sum_i \exp(u_{pi}^T p_{vi})} \quad (1)$$

$$h_t = \sum_i \alpha_{pi} p_{ti}$$

where p_t is the phoneme embedding, p_a and p_v are the weighted matrices in the training process, h_t is the phoneme vector after encoding as input of phoneme embedding layer.

Convolutional Layer. A convolution layer is first used to extract local n-gram features. It involves a filter $w \in R^{ld}$, which is applied to a window of l words to produce a new feature. For example, a window of words $h_{i:\, i+l-1}$ generates the feature map ct as follows,

$$c_t = f(w^{\circ} h_{i:i+l-1} + b) \quad (2)$$

where f is ReLU function, w and b denote the weight matrix and bias respectively, $^{\circ}$ is a convolutional operator which can be considered as 1D inputs to the CNN, l is the length of the filter, h is the phoneme vector after phoneme attention layer, d is the dimension of the phoneme vector, d_l is the dimension of the feature map c_t.

Pronunciation Attention Mechanism. We also apply an attention mechanism, which drives the model to discover the heterographic pun and use the feature map as input to extract pun-specific attention signal. The formulas are as follows.

$$u_o = \tanh(w \cdot c_a + b)$$

$$\alpha_o = \frac{\exp(u_{ci}^T c_{vi})}{\sum_i \exp(u_{ci}^T c_{vi})} \quad (3)$$

$$r_o = \sum_i \alpha_{ci}\, c_{ti}$$

where c_t is the convolutional feature map, c_a and c_v are the weighted matrices in the training process, r_o is the pronuncation understanding vector after encoding, the dimension of r_o is d_l.

3.2 Spelling Understanding Network(SUN)

Heterographic pun can rely on spelling to create the ambiguous effect. For example, "*When the church bought gas for their annual barbecue, proceeds went from the sacred to the propane.*" This pun exploits the spelling similarity between the surface sign "propane" and the latent target "profane".

This task has some challenges, in particular mapping clever near spelling into the intended form. For example, "Tom Swify" (Lessard and Levison, 1992) is one type of pun often found in the data set. Meanwhile, a large of puns is descriptions of things that have been seen and heard in the past. Moreover, heterographic pun is related with the part of speech and position. Miller points out the candidate pun words mainly consist of nouns, verbs, adjectives and adverbs in each pun at the end of the sentence. Our model introduces the useful information as different levels of embedding in the Spelling Understanding Network.

Character Embedding Layer. Our model accepts a sequence of encoded characters as input following Zhang et al. (2015). The encoding is prescribing an alphabet for the input language, and then quantizes each character according to 1-of-m encoding. The alphabet consists of 70 characters, including 26 English letters, 10 digits, 33 other characters and the new line character.

Character embedding layer is responsible for mapping each word to a high-dimensional vector space. Let $\{x_1,..., x_M\}$ represent the words in the input sentence. Following Kim (2014), we obtain the character-level embedding of each word for the intended form with heteographic pun by using Convolutional Neural Networks (CNN). Characters are embedded into vectors, which can be considered as 1D inputs to the CNN model, and the size is the input channel size of CNN model. Then the outputs of the CNN are max-pooled to get a fixed-size vector for each word, where the dimension of outputs is d_c.

Word Embedding Layer. Word embedding layer also maps each word to a high-dimensional vector space for the latent semantic representation. Here we use pre-trained word vectors, GloVe (Pennington et al., 2014) to obtain the fixed word embedding of each word for recognizing heterographic puns. The dimension of GloVe is d_w.

Tag Embedding Layer. Part of speech information is important clue for heterographic pun recognition. Here we define five types of tags: noun, verb, adjective, adverb and other. Tag embedding layer is also capable for representing each word to a high-dimensional semantic vector space. The dimension of tag embedding is d_t.

Position Embedding Layer. Position information can represent the distance for recognizing heterographic pun. Position embedding layer is also able to represent each term into a high-dimensional semantic vector space. The dimension of tag embedding is d_p.

The concatenation of the character-level, word-level, tag-level and position-level embedding vectors is pass to a concatenate operation. The output vector is $Q \in R^{(d_c+d_w+d_t+d_p)\times M}$, where the dimension is $(d_c+d_w+d_t+d_p)$.

Contextual Embedding Layer. We leverage a Long Short-Term Memory Network (LSTM) (Hochreiter and Schmidhuber, 1997) on top of the embeddings provided by the preceding layers to model the temporal interactions between words. LSTM was proposed by Hochreiter and Schmiduber (1997), which had been widely adopted for text processing. There are three gates and one cell in LSTM: an input gate i_t, a forget gate f_t, an output gate o_t and a memory cell c_t. They are all vector in $\mathbb{R}^d$. The equations of transition are:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1}) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + V_f c_{t-1}) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + V_o c_{t-1}) \\ \tilde{c_t} &= \tanh(W_c x_t + U_c h_{t-1}) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c_t} \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \quad (4)$$

where x_t is an input vector at the current time step,σ is the sigmoid function and $\odot$ is the element-wise multiplication operation, $W_{\{i,f,o,c\}}, U_{\{i,f,o,c\}}, V_{\{i,f,o,c\}}$ are learned weight parameters, h_t is the hidden state vector. In LSTM, the hidden state h_t only encodes the front context in a forward direction but not consider the backward context.

We apply a LSTM in both directions, and concatenate the outputs of the two LSTMs (Bi-LSTM) by Graves et al. (2013) to better learn the context with heterographic pun. It has a forward LSTM $\overrightarrow{h}$ and a backward LSTM $\overleftarrow{h}$ to concatenate the hidden states of two LSTMs as the representation of the corresponding word. By doing this, the forward and backward contexts can take into account simultaneously.

Hence, we obtain $E \in R^{2d_e \times M}$ from the input concatenation of the character, word, tag and position vectors Q, where e is the neural units of LSTM model. Note that each column vector of E is 2de-dimensional because the outputs concatenate the forward and backward LSTMs, each with d_u dimensional output.

Spelling Attention Mechanism. In order to discover the important information in spelling understanding, we design an attention mechanism, which enables model to concentrate on salient spelling. To make full use of spelling information, this model uses the contextual embedding as input to extract spelling-specific attention signal. The formulas are as follows.

$$u_f = \tanh(w \cdot e_a + b)$$

$$\alpha_{fi} = \frac{\exp(u_{ei}^T e_{vi})}{\sum_i \exp(u_{ei}^T e_{vi})} \quad (5)$$

$$s_f = \sum_i \alpha_{ei} e_{fi}$$

where e_f is the contextual embedding combining of character-level, word-level and tag-level embeddings, e_a and e_v are the weighted matrices in the training process, s_f is the spelling vector after encoding as the input of contextual embedding layer, the dimension of s_f is $2d_e$.

Gated Attention Mechanism. After learning the pronunciation understanding and spelling understanding networks, the next step is to combine them to get the integrated information. We carry out two strategies as following.

ST1: Averaged Aggregation strategy. We assume that the information of pronunciation part and spelling part are of same importance. The integrated information is computed as:

$$R_{st1} = \alpha * r_o + (1 - \alpha) * s_f \quad (6)$$

where α is 0.5, the dimension of the attention gate is same as r_o and s_f.

ST2: Weighted Aggregation strategy. We assume that the pronunciation information and spelling information are of different importance. We leverage gated attention to model the confidence of clues provided by the two parts. The value of the attention gate is computed as:

$$g = \sigma(w[r_o; s_f] + b) \quad (7)$$

where w is the weight matrix and b is the bias term, σ is the sigmoid function accepting vector as input.

We use 1-g and g as the combination weights of pronunciation side and spelling side to assemble r_o and s_f. Here the attention gate can be seen as a sentinel to control information combination between pronunciation part and spelling part. Note that the dimension of the attention gate is same as r_o and s_f, and the integrated information is computed by weighted sum as:

$$R_{st2} = (g^{\circ} r_o) + ((1 - g)^{\circ} s_f) \quad (8)$$

where $^{\circ}$ stands for element-wise multiplication operation, g is the combination weight, R_{st2} represents the final integrated information of various parts.

Output Layer. Follow previous work, we formulate heteographic pun recognition as a classification problem. In the output layer, we combine r_o, s_f, and R_{st2} as input to a softmax classifier:

$$z_{out} = \text{Softmax}(W \cdot [r_o; s_f; R_{st2}] + b) \quad (9)$$

where $z_{out} \in R^C$ is the vector of predicted probability for heterographic pun. Here C is the number of classes of pun labels, W and b are parameters of the classification layer, activation function is softmax.

3.3 Model Training

Here we introduce the learning and optimization details of our framework. The model can be trained in an end-to-end way by backpropagation, where objective function (loss function) is the cross-entropy loss. Here y is the target distribution and $\hat{y}$ is the predicted distribution. The goal of training model is to minimize the loss function between y and $\hat{y}$ for all examples. The formula is as following.

$$\text{loss}(\theta) = -\sum_i \sum_j y_i^j \log \hat{y}_i^j + \lambda \|\theta\|^2 \quad (10)$$

where i is the index of sentence, j is the index of class, λ is the regularization parameter and θ indicates all parameters.

For optimization method, we adopt mini-batch stochastic gradient descent (SGD) to minimize the objective function. Meanwhile, we add dropout strategy to prevent the co-adaptation of the parameters to settle the overfitting problem.

4 Experiments

In this section, we first introduce the dataset and evaluation metrics. Then we examine the performance of our PSUGA model comparison with existing methods. Finally, we give the detailed analysis of our proposed model.

4.1 Dataset and Evaluation Metrics

We conduct experiments on the widely used Pun of the Day[2] and SemEval2017 Task7[3] dataset.

Pun of the Day. This corpus only contains pun content in the beginning. Then it collects the negative samples from Yahoo!Answer, AP News, New York Times and Proverb to balance the distribution of positive and negative samples for the domain discrepancy.

SemEval2017 Task7. This corpus includes homographic and heterographic pun for recognizing and interpreting pun. We mainly focus on heterographic pun detection in pronunciation and spelling. Each text includes at least one pun. Table 1 shows a detailed statistical distribution of our datasets.

Table 1: Statistics on SemEval2017 Task7 and Pun of the Day.

Dataset	Positive	Negative
Task 7 (Heterographic Pun)	1271	509
Task 7 (Homographic Pun)	1607	643
Pun of the Day	2423	2403

We apply the following standard criteria precision, recall, accuracy and F1-score for evaluation, which also adopted as metrics in SemEval2017 task7 for heterographic pun recognition.

Training Details. In our experiments, our model is tuned with 5-fold cross validation. We apply the GloVe embedding (Pennington et al., 2014) and the dimension is 100. The size of units in LSTM is 100 and dropout is 0.5. We use learning rate decay and early stop in the training process.

4.2 Comparison with Existing Methods

We compare our model with many state-of-the-art methods as follows. The good performing models in SemEval2017 task7 are N-Hance, Idiom Savant, JU_CSE_NLP and ECNU (Miller et al., 2017).

- CNN: CNN without Pronunciation Spelling Understanding Gated Attention network.
- LSTM: LSTM without Pronunciation Spelling Understanding Gated Attention network.
- Bi-LSTM: Bi-LSTM without Pronunciation Spelling Understanding Gated Attention network.
- Bi-LSTM-Attention: Bi-LSTM with single attention mechanism.
- N-Hance (Sevgili et al., 2017): N-Hance system calculates the PMI between every pair of words and uses a certain threshold method to distinguish the pun.
- Idiom Savant (Doogan et al., 2017): This method uses Google n-grams, word2vec, and CMU Pronunciation Dictionary to recognize heterographic pun.
- JU_CSE_NLP (Pramanick and Das, 2017): It trains a hidden Markov and cyclic dependency network, using features from a part-of-speech tagger and syntactic parser.
- ECNU (Xiu et al., 2017): It applies a supervised machine learning approach to detect pun by using WordNet and word2vec embedding.
- PSUGA: Here we both consider the Pronunciation Understanding with hierarchical attention mechanism and phoneme embedding and Spelling Understanding with different level embeddings. They are concatenated by the gated attention mechanism.

The performance is shown in Table 2. Our PSUGA model uses Pun of the Day as the training set to obtain all the parameters, and test in the SemEval2017 task7 dataset for hetegraphic pun recognition. From the results, we have several observations that:

Table 2: Performance of all the methods.

Models	P(%)	R(%)	A(%)	F1(%)
CNN	81.57	80.73	78.61	81.15
LSTM	82.42	81.56	80.61	81.99
Bi-LSTM	82.71	82.34	80.82	82.52
CNN-Attention	81.98	81.26	79.44	81.62
Bi-LSTM-Attention	83.49	82.71	81.30	83.10
N-Hance	77.25	93.00	75.45	84.40
Idiom Savant	87.04	81.90	78.37	84.39
JU_CSE_NLP	73.67	94.02	71.74	82.61
ECNU	78.07	67.61	63.33	72.47
PSUGA	87.92	85.04	82.91	86.46

(1) LSTM has the better performance for heterographic pun recognition compared with CNN (81.99% vs. 81.15%). It shows that LSTM is able to find the global useful features for classification than CNN model. Meanwhile Bi-LSTM outperforms LSTM (82.52% vs. 81.99%), which demonstrated the two parallels of Bi-LSTM has ability to learn more context information.

[2] Pun of the Day: http://www.punoftheday.com/

[3] SemEval2017 Task7:http://alt.qcri.org/semeval2017/task7/

(2) Bi-LSTM-Attention performs slightly better than Bi-LSTM (83.10 % vs. 82.52%). Meanwhile CNN-Attention outperforms CNN (81.62 % vs. 81.15%). It shows that the attention mechanism can capture the semantic important parts to recognize heterographic pun. As we know, attention mechanism enables to improve the performance of NLP tasks.

(3) Among all the approaches, our method behaves best. It achieves the best performance (86.46% of F1) on SemEval2017 task7 and beats the best reported approach N-Hance by 2.06% on F1. It shows that the pronunciation information, spelling information and gated attention mechanism can largely affect the performance for this task.

(4) Compared with other methods, our model tends to achieve the higher precise score and accuracy score but lower recall. We argue it is the different types of additional information, which causes this phenomenon. Our model is able to learn pronunciation and spelling features, and apply the attention mechanism for providing more coherent and related clues. While for rule based methods, they usually need to fit the pun pattern by heuristic rules for obtaining high recall compared with our model.

4.3 Detailed Analysis

We conduct extra experiments to do detailed analysis as following.

4.3.1 Analysis of PSUGA model . In this subsection, we design a series of models to verify the effectiveness of our PSUGA model. First, we do not introduce the pronunciation, spelling and attention mechanism. We design N-Pron-N-Sp model which only using Bi-LSTM for recognizing heterographic pun. Then we implement our model Y-Pron-N-Sp (PUN), which considers pronunciation embedding and hierarchical attention mechanism based on CNN. Next, we design N-Pron-Y-Sp (SUN) model that employing character-level, word-level, tag-level, position-level and contextual-level embeddings to learn the spelling information based on Bi-LSTM and attention model. Then, we design our model Y-Pron-Y-Sp (PSUGA). Table 3 shows the performance of all the models.

Table 3: Analysis of the PSUGA model.

Models	P(%)	R(%)	A(%)	F1(%)
N-Pron-N-Sp	82.71	82.34	80.82	82.52
Y-Pron-N-Sp (PUN)	83.94	83.13	81.74	83.53
N-Pron-Y-Sp (SUN)	85.26	83.92	82.33	84.58
Y-Pron-Y-Sp (PSUGA)	87.92	85.04	82.91	86.46

(1) From Table 3, we can see that N-Pron-N-Sp model achieves the worse performance, which justifies the intuition of exploiting the pronunciation information, spelling information and gated attention mechanism to recognize the heterographic pun.

(2) For Y-Pron-N-Sp and N-Pron-Y-Sp model, they both outperform N-Pron-N-Sp and underperform Y-Pro-Y-Sp. It proves the effectiveness and necessity of the pronunciation part, spelling part, and the gated attention mechanism.

(3) Among all the models, Y-Pron-Y-Sp (PSUGA) beats the three baselines and achieves the best performance. The reason is that our model considers the hierarchical attention mechanism with CMU pronounce embedding, different level of embeddings with spelling information, and effective gated attention mechanism.

4.3.2 Impact of Feature Combinations . In this section, we exploit the effects of feature combinations. The experiments are conducted in the same setting. Results are show in table 4:

According to the results, we find that: (1) Phoneme embedding provides complement improvement for understanding the pronunciation information. (2) Character embedding is effective for understanding spelling. It proves that "Tome Swify", "-ed" these structures can discover the heterogarphic pun. (3) Tag embedding can improve the performance because the part of speech information is useful for this task. (4) While the effectiveness of position embedding is low. An intuition explanation is that when using Bi-LSTM to represent sentence, position information might have been encoded by some extent to the distributed representations for reducing the effect.

Table 4: Performance on Features Combination

Models	Features	F1(%)
PUN	Word Embedding	81.62
	Phoneme Embedding	83.53
SUN	Word Embedding	83.10
	+Character Embedding	83.82
	+ Tag Embedding	83.56
	+Position Embedding	83.37
	+Contextual Embedding	84.09
	All	84.58

4.3.3 Impact of Different Combination Strategies . We come up with the single model and combination model to exploit the effects of different attention strategies. Single model is the basic model to introduce the Pronunciation Understanding network (PUN) and Spelling Understanding network (SUN). Combination model is to assemble PUN and SUN with different strategies. Highway network which proposed by Srivastava et al. (2015) is an effective combination network. ST1 and ST2 are two attention strategies have been described in Section 3.2. Results are shown in table 5.

We find that: (1) all the combination models best the single model PUN and SUN, which justifies the intuition of combination strategy to recognize heterographic pun. (2) Among the combination models, both ST1 and ST2 achieve better results than concatenate and highway. It shows that the attention combination strategy can improve the performance for this task. (3) Besides, ST2 beats ST1 by a large margin (0.89% on F1), showing the advantages of using gated attention strategy to assemble information.

Table 5: Performance on Different Combination Strategies

Models		F1(%)
Single	PUN	83.53
	SUN	84.58
Combination	PSUGA with concatenate	85.09
	PSUGA with highway	85.42
	PSUGA with ST1	85.57
	PSUGA with ST2	86.46

4.3.4 Attention Visualization . Attention weights of two examples are visualized in Figure 2.

In Exp.1, "dentist", "hates" and "orifice" are of higher attention weights, which implies a pun related to part of speech and position. The weight of gated attention strategy is high for spelling understanding, which shows "orifice" and "office" has much similar spelling. In Exp.2, "grape" and "whine" are the words with much attraction in this sentence. The attention weight of pronunciation understanding is high, which implies pronunciation information can capture the semantic learning between "whine" and "wine".

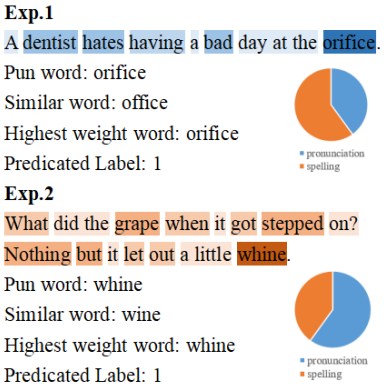


Figure 2: Attention visualization. The word is more important if the color is darker. The pie chart indicates the average value of the gated attention mechanism.

4.3.5 Error Analysis . We also analyze the sentence where our model failed to predict the correct labels of heterographic pun.

Exp.3 When the church bought gas for their annual barbecue, proceeds went from the sacred to the propane.

For example, the true pun label is "negative", but our model predicted its label as "positive". For predicting the label, ambiguity is the main reasons affected the performance. It needs deep semantic analysis of the sentence to identify the ambiguity out. Meanwhile, some background knowledge would be required to predict the label correctly.

5 Conclusion and Future Work

In conclusion, we propose an automatic computational PSUGA model combined with Pronunciation-based hierarchical attention mechanism, Spelling-based different levels embeddings and gated attention strategy to settle with confused and ambiguous meaning. Experimental results show that our method achieves significant improvement over exiting methods and sets a new state-of-the-art performance.

In future work, we would like to find an appropriate way in incorporation the external background knowledge for recognizing the heterographic pun. We also focus on deep semantic analysis, interpretation and generation with heterographic pun. Those are all promising jobs we can pursue in the future.

6. ACKNOWLEDGMENTS

This work is partially supported by grant from the Natural Science Foundation of China (No.61632011, 61572102, 61772103, 61702080, 61602079), the Ministry of Education Humanities and Social Science Project (No.16YJCZH12), the Fundamental Research Funds for the Central Universities (DUT18ZD102), the National Key Research Development Program of China (No. 2016YFB1001103), and China Postdoctoral Science Foundation (No. 2018M631788).

REFERENCES

Tristan M and Mladen T. Towards the automatic detection and identification of English puns[J]. European Journal of Humour Research, 2016 ,4(1):59–75.
Tanaka K. The pun in advertising: A pragmatic approach[J]. Lingua, 1992, 87(1-2): 91-102.
Miller T, Gurevych I. Automatic disambiguation of English puns[C]//ACL. 2015: 719-729.
Yang D, Lavie A, Dyer C, et al. Humor Recognition and Humor Anchor Extraction[C]// EMNLP. 2015: 2367-2376.
Hempelmann CF. Computational humor: Beyond the pun?[J]. The Primer of Humor Research. Humor Research, 2008, (8): 333-360.
Zwicky A, Zwicky E. Imperfect puns, markedness, and phonological similarity: With fronds like these, who needs anemones[J]. Folia Linguistica, 1986, 20(3-4): 493-503.
Robinson, T. The British English example pronounciation dictionary [M], 1996.
Kim Binsted. Machine humour: An implemented model of puns. PH.D. University of Edinburgh, 1996.
Kao J T, Levy R, Goodman N D. A Computational Model of Linguistic Humor in Puns[J]. Cognitive science, 2015.
Jaech A, Koncel-Kedziorski R, Ostendorf M. Phonological Pun-derstanding[C]. Proceedings of NAACL-HLT. 2016: 654-663.
Sevgili Ö, Ghotbi N, Tekir S. N-Hance at SemEval-2017 Task 7: A Computational Approach using Word Association for Puns[C]//Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). 2017: 436-439.
Doogan S, Ghosh A, Chen H, et al. Idiom Savant at Semeval-2017 Task 7: Detection and Interpretation of English Puns[C]//Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). 2017: 103-108.
Pramanick A, Das D. JU_CSE_NLP at SemEval 2017 Task 7: Employing Rules to Detect and Interpret English Puns[C]//Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). 2017: 432-435.
Xiu Y, Lan M, Wu Y. ECNU at SemEval-2017 Task 7: Using Supervised and Unsupervised Methods to Detect and Locate English Puns[C]//Proceedings of

the 11th International Workshop on Semantic Evaluation (SemEval-2017). 2017: 453-456.
Hempelmann C F. Paronomasic puns: Target recoverability towards automatic generation[J]. Dissertation Abstracts International, 2003 ,64(11):4029.
Ritchie. Computational mechanisms for pun generation[J]. Proceedings of the 10th European Workshop on Natural Language Generation, 2005: 8-10.
Hong B A, Ong E. Automatically extracting word relationships as templates for pun generation[C]. The Workshop on Computational Approaches To Linguistic Creativity. Association for Computational Linguistics, 2010:24-31.
Julia M. Taylor and Lawrence J. Mazlack. Computationally recognizing wordplay in jokes. In Proceedings of the 26th Annual Conference of the Cognitive Science Society[C] (CogSci 2004), 2004:1315–1320.
Julia M Taylor. Computational detection of humor: A dream or a nightmare? the ontological semantics approach[C]. In Proceedings of the 2009 ACM International Joint Conference onWeb Intelligence and Intelligent Agent Technology, 2009, (03): 429–432.
Dario Bertero and Pascale Fung. A Long Short-Term Memory Framework for Predicting Humor in Dialogues[J]// Proceedings of NAACL-HLT 2016: 130-135.
Bahdanau D, Cho K, Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate[J]. Computer Science, 2014.
Yang Z, Yang D, Dyer C, et al. Hierarchical Attention Networks for Document Classification[C]//HLT-NAACL. 2016: 1480-1489.
Tan Z, Wang M, Xie J, et al. Deep Semantic Role Labeling with Self-Attention[C]. AAAI. 2018.
Shen T, Zhou T, Long G, et al. Disan: Directional self-attention network for rnn/cnn-free language understanding[C]. AAAI, 2018.
Du J, Xu R, He Y, et al. Stance Classification with Target-specific Neural Attention[C]// Twenty-Sixth International Joint Conference on Artificial Intelligence. 2017:3988-3994.
Seo M, Kembhavi A, Farhadi A, et al. Bidirectional attention flow for machine comprehension[J]. ICLR, 2017.
Yu A W, Dohan D, Luong M T, et al. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension[J]. ICLR, 2018.
Gui L, Hu J, He Y, et al. A question answering approach to emotion cause extraction[J]. arXiv preprint arXiv:1708.05482, 2017.
Lu J, Yang J, Batra D, et al. Hierarchical question-image co-attention for visual question answering[C]//Advances In Neural Information Processing Systems. 2016: 289-297.
Zhang X, Zhao J, LeCun Y. Character-level convolutional networks for text classification[C]//Advances in neural information processing systems. 2015: 649-657.
Kim Y. Convolutional neural networks for sentence classification[J]. arXiv preprint arXiv:1408.5882, 2014.
Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.
Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.
Graves A, Jaitly N, Mohamed A. Hybrid speech recognition with deep bidirectional LSTM[C]//Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on. IEEE, 2013: 273-278.
Srivastava R K, Greff K, Schmidhuber J. Highway networks[J]. arXiv preprint arXiv:1505.00387, 2015.
Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: A simple way to prevent neural networks from overfitting[J]. The Journal of Machine Learning Research, 2014, 15(1): 1929-1958.
Xu C, Paris C, Nepal S, et al. Cross-Target Stance Classification with Self-Attention Networks[J]. arXiv preprint arXiv:1805.06593, 2018.
Zhang H, Goodfellow I, Metaxas D, et al. Self-Attention Generative Adversarial Networks[J]. arXiv preprint arXiv:1805.08318, 2018.
Shen T, Zhou T, Long G, et al. Bi-directional block self-attention for fast and memory-efficient sequence modeling[J]. arXiv preprint arXiv:1804.00857, 2018.
Kim J, Kong D, Lee J H. Self-Attention-Based Message-Relevant Response Generation for Neural Conversation Model[J]. arXiv preprint arXiv:1805.08983, 2018.
Fan A, Lewis M, Dauphin Y. Hierarchical Neural Story Generation[J]. arXiv preprint arXiv:1805.04833, 2018.
Miller T, Hempelmann C F, Gurevych I. SemEval-2017 Task 7: Detection and interpretation of English puns[C]//Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). Vancouver, BC. 2017.
Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.

Evaluating Login Challenges as a Defense Against Account Takeover

Periwinkle Doerfler[†] Maija Marincenko[◇] Juri Ranieri[◇] Yu Jiang[◇]
Angelika Moscicki[◇] Damon McCoy[†] Kurt Thomas[◇]

New York University[†] Google[◇]

ABSTRACT

In this paper, we study the efficacy of login challenges at preventing account takeover, as well as evaluate the amount of friction these challenges create for normal users. These secondary authentication factors—presently deployed at Google, Microsoft, and other major identity providers as part of risk-aware authentication—trigger in response to a suspicious login or account recovery attempt. Using Google as a case study, we evaluate the effectiveness of fourteen device-based, delegation-based, knowledge-based, and resource-based challenges at preventing over 350,000 real-world hijacking attempts stemming from automated bots, phishers, and targeted attackers. We show that knowledge-based challenges prevent as few as 10% of hijacking attempts rooted in phishing and 73% of automated hijacking attempts. Device-based challenges provide the best protection, blocking over 94% of hijacking attempts rooted in phishing and 100% of automated hijacking attempts. We evaluate the usability limitations of each challenge based on a sample of 1.2M legitimate users. Our results illustrate that login challenges act as an important barrier to hijacking, but that friction in the process leads to 52% of legitimate users failing to sign-in—though 97% of users eventually access their account in a short period.

KEYWORDS

account takeover; account recovery; two-factor authentication

ACM Reference Format:
Periwinkle Doerfler, Maija Marincenko, Juri Ranieri, Yu Jiang, Angelika Moscicki, Damon McCoy, Kurt Thomas. 2019. Evaluating Login Challenges as a Defense Against Account Takeover. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313481

1 INTRODUCTION

In response to the threat of wide-scale password theft and account takeover [22], identity providers have re-framed password-only authentication from a binary task of validating password correctness into *risk-aware authentication* [19]. This approach incorporates multiple passive authentication signals with layered proofs of identity. For example, Freeman et al. developed a statistical model of a user's geolocation, browser configuration, and access patterns to detect aberrant login attempts [10]. Google and Microsoft actively deploy similar systems that judge IP address reputation, device reputation, and automation patterns [18, 23]. Of course, with any risk analysis scheme there exists a chance for false positives as users travel and acquire new devices, and networks churn due to DHCP lease expiration. As such, outright blocking any suspicious sign-in would lock a significant volume of legitimate users out of their accounts.

In order to differentiate hijackers from legitimate account holders in edge cases, risk-aware authentication triggers a *challenge* to act as a second proof of identity. Challenges can take many forms: proof of access to a registered device, proof of access to a backup account, knowledge of a shared secret (e.g., a security question), or merely proof of access to a scarce resource (e.g., a CAPTCHA). This layered authentication approach is distinct from two-factor authentication as challenges appear dynamically in response to a perceived threat, thus reducing overall friction on every user sign-in. Additionally, challenges scale to every user by adapting to the security posture of each account, taking advantage of recovery accounts or trusted devices when available. While ideal from a deployment standpoint, it remains unclear how well these varied challenges protect against hijacking or what friction they create for legitimate account holders.

In this paper, we evaluate the security and usability trade-offs of fourteen challenges based on their real-world performance. We begin by building a ground truth corpus of over 350,000 hijacking attempts from increasingly sophisticated actors including automated bots, phishers, and targeted attackers. Then, using login traces from Google, we examine whether risk-aware authentication triggered a challenge for these hijacking attempts, what challenge the algorithm selected, and ultimately whether the hijacker was able to access the account. Additionally, we isolate 1.2 million challenges solved by legitimate users to measure how often challenges temporarily locked out account holders. Critically, our evaluation approach is independent of Google's risk analysis techniques and generalizes to any identity provider.

Our analysis reveals that even weak, knowledge-based challenges can offer hijacking protections against automation to billions of users without requiring any enrollment. That said, the security posture of users plays an important role in protecting against more sophisticated attacks. Users that established a trusted device or a delegated recovery account received up to 10 times better protections against phishing, and to a lesser extent, targeted attacks. We argue that risk-aware authentication and challenges can help extend the viability of passwords in the interim as the community

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313481

considers alternative forms of authentication. For at-risk users looking to mitigate the risk of remote hijacking entirely, we emphasize that security keys offer the best immediate solution.

We summarize our key findings as follows:

- Knowledge-based challenges, such as recalling a secondary email address, prevented over 73% of automated hijacking attempts but only 10% of attacks rooted in phishing.
- On-device prompts, like those shown during two-factor authentication, provide the strongest protection. They blocked 99% of attacks rooted in phishing and 90% of targeted attacks.
- SMS-based challenges provided weaker protections, preventing only 96% of attacks rooted in phishing and 76% of targeted attacks.
- We show that risk-aware authentication in aggregate prevented over 99.99% of automated hijacking attempts and over 92% of attacks rooted in phishing.
- Despite low per-challenge pass rates by legitimate users (13–64% depending on the challenge), over 97% of users gained access to their account within a short period of seeing a challenge. This figure is comparable to password-only authentication.

2 LOGIN & RECOVERY CHALLENGES

Before diving into our study, we outline the fourteen challenges currently deployed at Google that can be triggered at login or during account recovery. We categorize these challenges into four classes as shown in Table 1. While our study focuses on empirical measures of performance, we also frame each challenge's trade-offs between memorability, ease of use, and susceptibility to attacks according to the criteria set down by Bonneau et al. for evaluating web authentication schemes [3]. We extend this taxonomy to include five new criteria. *Frictionless setup*—the challenge does not require a pre-established secret with an identity provider. *No additional network access*—there is no need for a cellular or other network connection. *Resilient to third-party leaks*—the challenge does not involve data that might be revealed by a breach. *Not publicly searchable*—the challenge cannot be gleaned from public information. And lastly *Affinity to user*—the challenge establishes some relationship to a user, rather than just a hard to acquire resource. We discuss each class of challenge according to this criteria and how Google opts to elect one challenge over another during risk analysis.

2.1 Types of challenges

Device-based Challenges. Device-based challenges leverage the ubiquity of mobile devices to establish a second authentication factor for users or, less frequently, access to a hardware security token. For users with mobile devices, identity providers can leverage one-time pads (e.g., a 6-8 digit code that users re-type) generated by an app like Google Authenticator [16] or Duo [7] or via the phone's operating system [11]. Both can operate offline by using a synchronized key between a server and device. Additional delivery mechanisms include on-device prompts where a user clicks "Yes" to confirm a sign-in attempt [7, 13] or SMS where a user re-types a numeric code texted to their device. Both of these require additional network access.

As detailed in Table 1, the primary usability concern of device-based challenges is both access to a mobile device or phone number and the willingness of users to proactively associate these with their account. Sharing a phone number incurs at least some friction in terms of setup, while registering a device or authenticator app requires even more steps. Additionally, while the time-sensitive nature of device challenges obviates some classes of attacks, they are nevertheless vulnerable to man-in-the-middle attacks such as inline phishing. This stems from the inability of apps that generate a code to observe the domain a victim is visiting, something U2F security keys solve. Theft of mobile devices is another risk as it both locks the victim out of their account and gives attackers access to OTP codes. Requiring a PIN or password to access the device partially mitigates this threat.

Delegation-based Challenges. Delegation-based challenges rely on a secondary identity provider to serve as an authority for establishing trust with a user. At Google, this involves sending a one-time code to a secondary email address. Other alternatives include Facebook's delegated recovery [9] where Facebook provides a break-glass, pre-registered secret token that users can employ in lieu of a password at the other identity provider. These schemes require memorizing a username and password for the delegated identity provider. As a single identity (and thus password) can span multiple other providers, we consider backup emails to be partially memorywise effortless. Compared to device-based challenges, these schemes are susceptible to stepping-stone attacks, phishing, or brute force guessing of the secondary account's password, where security is only as strong as the weakest link in the identity chain.

Knowledge-based Challenges. This class of challenges relies on establishing a shared secret with users either at registration time or as the result of actions on a service. Examples include answering security questions, or providing a recovery email or phone number (where no challenge code is sent). Alternatively, identity providers may probe information that a user implicitly provided, such as where the user commonly logs in from or the date they first created an account. Finally, while not presently deployed at Google, challenges may cover a history of the user's interactions, such as selecting the names or faces of contacts the user recently communicated with among decoy contacts [8].

Some knowledge-based schemes provide a frictionless account creation process, but then force users to recall potentially non-obvious interactions in the future. We consider re-usable secrets, such as an email or phone number, to be partially frictionless to establish. Knowledge-based challenges are also vulnerable to any form of observation, including third-party leaks or search indexing, and even fail to protect against throttled guessing when the solution space is biased towards common answers [2]. Nevertheless, these challenges are the only fallback mechanism for establishing some proof of identity when users refuse requests to establish device-based or delegation-based backups.

Resource-based Challenges. Resource challenges do not attempt to establish a proof of identity with users. Instead, they offer a hurdle to prevent automation. These challenges include confirming access to any phone number or (secondary) email address. As such, the primary goal is to restrict throttled guessing. However, attackers

Category	Challenge	Context	Memorywise Effortless	Nothing to Carry	No Additional Network Access	Frictionless Setup	Affinity to User	Resilient to Physical Observation	Resilient to Phishing	Resilient to Third-party Leaks	Not Publicly Searchable	Resilient to Physical Theft	Resilient to Throttled Guessing	Resilient to Unthrottled Guessing	Single Trusted Identity Provider
Device-based	App-based OTP[7, 16]	S	●		●		●	●	◐	●	●	◐	●		●
	Device Settings OTP[11]	L, R	●		●		●	●	◐	●	●	◐	●		●
	Device Prompt[7, 13]	S, L, R	●				●	●	◐	●	●	◐	●	●	●
	Mobile SMS OTP	S, L, R	●			◐	●	●	◐	●	●	◐	●		
	Security key	S	●		●		●	●	●	●	●		●	●	●
Delegation-based	OTP to backup email	L, R	◐	●	●	◐	●			●	●	●			
Knowledge-based	Known pre-registered email	L, R	◐	●	●	◐	●					●	●		●
	Security question	L, R		●	●		●				●	●			●
	Known pre-registered phone number	L, R	◐	●	●	◐	●					●	●		●
	Last login location	L		●	●	●	●			◐	◐	●	●		●
	Account creation date	R		●	●	●	●			●	●	●	●		●
	Printed backup code	S, R	●		●		●			●	●		●		●
Resource-based	OTP to any phone	L	●			●		●	◐	●	●	◐	●		
	OTP to any email address	R	●	●	●	●				●	●	●	●		

Table 1: Challenges presently deployed at Google, broken down by their usability and security concerns. Challenges may appear in multiple contexts: at login due to a suspicious sign-in attempt (L), during recovery when a user forgets their passwords (R), or as a second-factor for accounts with 2FA enabled (S). We indicate fulfillment of a property with a filled circle (●) and partial fulfillment with a half circle (◐). We refer readers to Bonneau et al. for a detailed description of each criteria [3].

can farm these challenges out to unsuspecting users or rely on compromised resources including phones or email accounts.

2.2 Challenge Selection in Practice

Challenge selection at Google is a function of both security posture and context. In the event of a suspicious sign-in attempt, Google's risk analysis engine selects the strongest challenge that an account's legitimate owner should ideally be able to solve. For accounts with an associated device or phone number, the risk engine exclusively allows device-based challenges[1]. Absent a device, the engine falls back to delegation-based challenges, then knowledge-based challenges, and ultimately additional resource challenges. As such, even though a hijacker may know a victim's phone number or backup email, the risk engine will only present such a challenge if no stronger option exists. We present a sample challenge served by Google in Figure 1. Failing a challenge does not allow an actor to subsequently select a weaker class of challenge; for legitimate users, this may mean account lockout until the user can regain access to a trusted network or device. This lock out friction mitigates the risk of downgrade attacks influencing the challenge selection process towards weaker challenges.

While we primarily focus on challenges shown during login and account recovery in this study, we also consider the usability and security of two-factor authentication. We provide a detailed breakdown in Table 1 of the contexts where a challenge may appear. Two-factor authentication always involves a phone or security key to solve a device-based challenge. During recovery, Google, in some cases, might allow users to solve weaker challenges such as providing an account's creation date or confirming access to any email account[2]. The recovery flow mitigates the hijacking risk of such knowledge-based challenges by requiring users to solve multiple, successive challenges and by blocking recovery attempts from devices or networks deemed as high-risk. As with login challenges, Google's recovery process elects the strongest challenge available based on the pre-negotiated secrets and recovery options established by an account holder.

3 DATASETS

In order to evaluate the security and usability of challenges in practice, we had to overcome two hurdles: (1) obtaining a dataset of challenges that hijackers attempted without our findings being biased by Google's risk detection; (2) obtaining a dataset of challenges served to legitimate users where memorability, misplaced devices, and an incentive to access to the account were all in play. We describe our approach to constructing each dataset below and the

[1]The risk engine also makes a judgment to the "liveness" of the device or phone number. This is why phone numbers can be part of an SMS challenge as well as a knowledge challenge. The process for determining liveness is beyond the scope of this paper.

[2]Only for accounts with no two-factor authentication enabled.

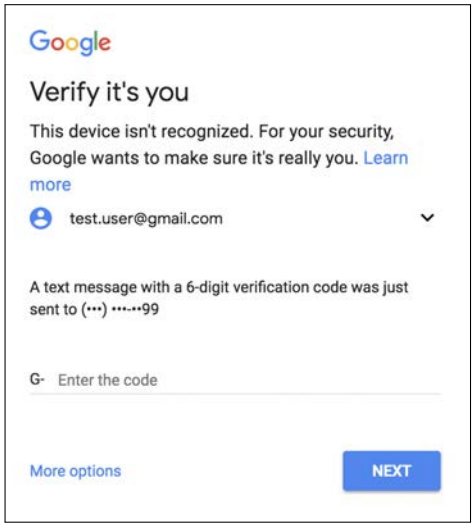


Figure 1: Sample challenge deployed as part of Google's risk engine for login and account recovery. The example shown requests that an account holder confirm an OTP sent via SMS.

Category	Description	Period	Samples
Usability	Challenge to login from familiar device or network.	Jun 19, 2017–Aug 15, 2017	300,000
	Challenge to login from unfamiliar device and network.	Feb 23, 2018–May 31, 2018	300,000
	Recovery challenge from familiar device or network.	Feb 23, 2018–Mar 22, 2018	300,000
	Two-factor challenge from familiar device or network.	Feb 23, 2018–Mar 22, 2018	300,000
Security	Challenge served to suspected automated bot.	Feb 23, 2018–May 31, 2018	300,000
	Challenge served to suspected phishing attack.	Feb 23, 2018–May 31, 2018	50,867
	Challenge served to suspected targeted attack.	Feb 23, 2018–May 31, 2018	453

Table 2: Datasets for studying usability and security performance for challenges.

ethical considerations involved. We provide a high-level overview of the resulting datasets in Table 2.

3.1 Security Datasets

For our study, we leverage three existing threat intelligence feeds that contain credentials stolen by automated bots, phishers, and targeted attackers. We retroactively joined these with records of logins to Google from February 23, 2018–May 31, 2018. For each login, we isolated sign-in attempts from hijackers (described below) and ultimately output whether Google's risk engine served a challenge in response to the hijacking attempt. Our resulting dataset consists of the type of challenge served and one of four outcomes: *passed* indicating the attacker supplied the correct challenge response, *failed* indicating an incorrect response, *skipped* indicating the attacker opted to try and solve an alternative challenge, or *abandoned* indicating the attacker exited the login flow and never supplied a solution.

3.1.1 Challenges to Automated Bots. Automated bots rely on third-party data breaches and brute force password guessing in an attempt to access millions of accounts every day. To detect this activity, Google scores every login using its passive reCAPTCHA system to identify bot activity [12]. We sampled a total of 300,000 login attempts flagged as bot activity, where the target account displayed previous organic sign-in activity (e.g., not a fake account). We manually evaluated metadata for 100 of these sessions and found all were indeed hijacking attempts. We then examined whether Google's risk engine triggered an accompanying challenge for each login. This occurred in every login attempt, resulting in a final dataset of 300,000 challenges. We caution that some bots may only be engaged in checking password validity and not attempt to solve a challenge. As such, we likely underestimate the capabilities of bots to automatically solve challenges. Furthermore, as bot detection plays a role in risk analysis, any estimate of overall hijacking protection provided against bots may be biased towards the types of bots that reCAPTCHA can identify.

3.1.2 Challenges to Phishers. Phishers harvest usernames and passwords, but also additional personal information including IP addresses and phone numbers in an attempt to bypass challenges [5]. Using data from Thomas et al. [22], we collected a sample of 400,000 Google accounts that had their valid password stolen between February 23–May 21, 2018. We then retroactively analyzed all Google sign-in attempts to these accounts during the same period. Only 20.4% of these accounts had any new sign-in activity, indicating phishers either waited on the stolen credentials or that another service may have been the target of interest. For accounts with some sign-in activity, we clustered each login in an effort to identify a single attacker accessing multiple phished accounts. The features we use for clustering are sensitive, but the overall technique identifies hijackers that re-use automation tools or resources when logging in. We note this clustering is independent of risk analysis and thus should not introduce any bias. In total, clustering identified 81,450 sessions as potential hijacking attempts. We manually evaluated metadata for 100 of these sessions and found all were indeed hijacking.

For each login attempt, we then examined whether Google's risk analysis engine triggered a challenge. In total, we identified 50,867 challenges served to phishers. In each case, the risk analysis engine triggered on a suspicious sign-in from a new country previously unassociated with the victim's account. The absence of a challenge in some cases stems from incorrect passwords (the login is rejected and no challenge shown) and because risk analysis will not trigger on repeated, successful accesses (if a phisher successfully accesses the account once, they will not be challenged in the future). We reiterate this analysis was retroactive; there were no additional protections we could provide to the victims at the time of the hijacker's sign-in attempt. In the course of our investigation, if we identified a successfully hijacked account, we forced the account into recovery thus blocking any further access by hijackers.

3.1.3 Challenges to Targeted Attackers. Our last security dataset consists of 214 Google accounts previously identified as victims of targeted attacks between February 23–May 21, 2018 [20]. We lack context for the attack techniques involved in every case, but many relate to spear phishing. We retroactively analyzed each account to isolate the hijacker's attempted login. From this, we identified 453 challenges that Google's risk engine served to attackers. As with phishing, this retroactive analysis precludes the possibility of providing any additional defense to victims at the time of hijacking. In the course of our investigation, if we identified a successfully hijacked account, we forced the account into recovery thus blocking any further access by hijackers.

3.2 Usability Datasets

In order to capture a spectrum of users and contexts, we developed four datasets of challenges to legitimate account holders: challenges to sign-ins from a familiar device or network (via an active experiment), organic challenges shown to sign-ins from an unfamiliar device or network, challenges shown during two-factor authentication, and challenges served during account recovery. As with our security dataset, we ultimately obtain only the type of challenge served and the resulting outcome.

3.2.1 Challenges to Familiar Devices & Networks. In order to gather a sample of challenges solved exclusively by legitimate users, we ran an experiment where we augmented Google's sign-in flow to challenge a random sample of user logins from familiar devices or networks. In the normal course of risk-based authentication, Google would not challenge these logins. We ran our experiment from June 19, 2017–August 15, 2017 and collected a total of 300,000 challenges and responses. We selected the challenge displayed randomly from a set of 9 available challenges.[3]

We ameliorated any risk of lock out by introducing a "Skip" button after a delay of 5 seconds which allowed users to bypass the experiment and continue to their account. Similarly, we allowed users to access their account regardless of the correctness of their response to the challenge. While this may skew results, ethically it presents the least friction to users. We caution that pass rates are thus likely lower than in practice.

3.2.2 Challenges to Low-risk New Devices & Networks. Given the bias of our previous experiment, we also consider a second set of organic challenges and responses served to a random sample of 300,000 user logins from February 23, 2018–May 31, 2018. Google's risk-analysis triggered these challenges due to users signing in from unfamiliar devices and networks, but considered them low risk as there were no other signs of automation or known hijacking access patterns. While the majority of such logins belong to legitimate users—risk analysis is biased towards false positives that lead to friction rather than false negatives that lead to hijacking—it may be that the dataset contains some hijacking attempts. We cannot empirically evaluate how many hijackings appear in this dataset due to a lack of supplementary signals.

3.2.3 Account Recovery Challenges to Familiar Devices & Networks. Account recovery captures users in moments where they have forgotten their password and potentially any other pre-shared secrets. To measure this effect on challenge pass rates, we randomly sampled 300,000 challenges shown organically as part of Google's account recovery flow from February 23, 2018–March 22, 2018. We limit our selection to flows initiated by users from familiar devices and networks to minimize any risk of including hijacker activity.

3.2.4 Two-factor Authentication Challenges to Familiar Devices & Networks. On the other end of the spectrum, users with two-factor authentication are primed to expect a challenge and to have a device on-hand. We randomly sampled 300,000 device-based challenges from February 23, 2018–March 22, 2018 for users with two-factor authentication enabled. We restrict our selection to sign-ins from familiar devices and networks to minimize any risk of including hijacker activity.

3.3 Ethics

Our in situ measurement of challenges raises a number of ethical concerns. While Google does not require IRB approval, we still took care to design our data collection to never capture sensitive private or identifying information related to the legitimate account holders under analysis. Our controlled experiment—the only instance where we introduce new challenges—also minimized any risk of account lock out. When analyzing hijacking attempts, our signals either derive from existing user protections or from auxiliary annotations that preclude real-time usage as a protection. As such, while we may learn of a successful hijacking, it is always after-the-fact. In those cases, we force the affected account into recovery in order to prevent any further damage by hijackers.

3.4 Limitations

Our security dataset is biased towards the class of bots, phishers, and targeted attackers for which we have threat intelligence and thus not a true random sample. Additionally, as attacks and capabilities may evolve over time, our measurements capture only a current snapshot of the effectiveness of challenges at preventing hijacking. Furthermore, as our experiment is in situ, we do not control the challenges these attackers receive. This may result in smaller samples for one type of challenge as compared to another. For example, if a victim has a device associated with their account it precludes showing any other class of challenges. We provide error margins on all measurements to account for small sample sizes. Finally, while we randomly sample Google's entire global user base to assess usability, in practice challenge solution rates likely differ by region. As such, any overall overall passrate statistics will be biased towards the underlying demographic and device accessibility distribution of Google's users.

4 THE SECURITY OF CHALLENGES

We begin our analysis by assessing the performance of individual challenges and ultimately the hijacking risk posed by bots, phishing, and targeted attacks. Our results illustrate that device-based and delegation-based challenges—despite their known limitations—protect against 100% of automated bots and over 92% of attacks rooted in phishing. Furthermore, we show that certain device-based and knowledge-based challenges outperform their category peers. Identity providers should prioritize these challenges when available.

[3]The experiment excludes challenges shown only during recovery or that require two factor authentication via an app or security key.

Category	Challenge	Prevention rate			Skip rate		
		Bot	*Phishing*	*Targeted*	*Bot*	*Phishing*	*Targeted*
Device-based	Authenticator OTP	100% ± 3%	94% ± 7%	–	0% ± 3%	38% ± 7%	–
	Device Prompt	100% ± 1%	99% ± 1%	90% ± 9%	0% ± 1%	45% ± 1%	43% ± 9%
	Device SMS OTP	100% ± 1%	96% ± 1%	76% ± 6%	0% ± 1%	38% ± 1%	12% ± 6%
	Device Settings OTP	100% ± 7%	100% ± 5%	–	0% ± 7%	49% ± 5%	–
	Security Key	100% ± 25%	100% ± 28%	–	0% ± 25%	25% ± 28%	–
Delegation-based	OTP to backup email	100% ± 1%	92% ± 3%	–	0% ± 1%	38% ± 3%	–
Knowledge-based	Account creation date	–	91% ± 5%	100% ± 22%	–	48% ± 5%	5% ± 22%
	Known pre-registered email	73% ± 2%	68% ± 1%	79% ± 26%	0% ± 2%	19% ± 1%	21% ± 26%
	Known pre-registered phone number	100% ± 16%	26% ± 1%	50% ± 20%	0% ± 16%	6% ± 1%	17% ± 20%
	Last login location	100% ± 28%	10% ± 2%	–	0% ± 28%	1% ± 2%	–
	Printed backup code	–	–	–	–	–	–
	Security question	–	93% ± 4%	–	–	38% ± 4%	–
Additional resource	OTP to any email address	–	61% ± 5%	–	–	33% ± 5%	–
	OTP to any phone	100% ± 1%	45% ± 2%	–	0% ± 1%	16% ± 2%	–

Table 3: Hijacking attempts from bots, phishers, and targeted attackers prevented by login or recovery challenges along with error margins for 95% confidence intervals. Device-based challenges prevent 94% of attacks rooted in phishing and 76% of targeted attacks.

4.1 Prevention rate per challenge

The primary security metric for any challenge is whether the information requested cannot be produced by a hijacker. Rogue sources of challenge solutions may come from third-party breaches, phishing, or other forms of observation such as malware on a victim's device. We measure a challenge's prevention rate as:

$$1 - \frac{passed\ challenges}{total\ challenges}$$

We provide a breakdown of prevention rates per challenge and class of attack in Table 3. We omit statistics in scenarios where we have fewer than 10 samples. We include error margins for 95% confidence intervals.

Automated Bots. We find that bots failed to solve 100% of device-based and delegation-based challenges in our sample. This indicates that automation tools lack both real-time access to OTPs and stepping-stone access to the target's secondary accounts. Similarly, bots appear to lack access to expensive-to-scale resources such as phone numbers for receiving an SMS challenge, failing 100% of such challenges. Knowledge-based challenges posed the only hijacking risk, with bots solving 27% of challenges asking for a pre-registered recovery emails. This likely stems from third-party breaches revealing billions of email records and potential recovery details [22]. In the same vein however, these bots do not appear to have access to prior login locations or phone numbers. We note that we cannot calculate the prevention rate of three challenges as they only appear in a recovery context where bots are outright blocked.[4] Our results illustrate that while third-party breach data may be readily available in underground communities, we do not observe its widespread use in hijacking attempts beyond access to valid passwords and secondary identities.

[4]Bots are not exclusively blocked from login in the unlikely event of a false positive, which might otherwise lock a legitimate user out.

Phishing. We find that no single category of challenges is immune to hijacking attempts rooted in phishing. Device-based challenges provided the strongest protection, preventing 94–100% of attacks in our sample. The presence of any successful hijackings indicates that a fraction of attackers can relay OTP codes (or have users click a prompt) within the challenge's freshness window. Given the remote nature of attackers, we believe this likely results from attackers coercing users through a man-in-the-middle attack. Zooming in, we find that phishers correctly solved 4% of SMS challenges compared to 1% of on-device prompts. We believe this statistically significant difference ($p < 0.0001$) results from the context that account holders have about the source of a sign-in. If a user enters their credentials into a man-in-the-middle phishing page, an SMS challenge relays only an OTP code with no additional information. In comparison, an on-device prompt reveals the device type (e.g., phone, desktop), operating system, and geolocation of the phisher who initiated the sign-in attempt. Account holders can use this information to then block the hijacking attempt. Alternatively, it is possible that attackers deceive cellular providers into handing over control of the victim's phone number. There is no equivalent remote attack on device prompts. Further, because there is no overhead for offline codes, the freshness window can be much shorter than for SMS codes. Offline codes from apps are refreshed every 30-60 seconds, whereas SMS based codes may be valid for hours or even days. These longer windows may be another reason that SMS challenges are more susceptible to phishing. In order to avoid all phishing risks, security keys provide the best protection, blocking 100% of attacks in our dataset.

Our phishing dataset also quantifies the risk of relying on secondary identity providers. Here, we find that phishers supplied proof of access to a backup email 8% of the time. Two possible explanations exist: hijackers either phished the OTP code from victims via a man-in-the-middle attack just as they did with device-based challenges, or the attackers had direct access to the backup account. If OTP phishing were the sole technique, we might expect similar prevention rates as OTP codes delivered via SMS (or

Challenge	Incorrect solution rate		
	Bot	*Phishing*	*Targeted*
Account creation date	–	70% ± 10%	–
Known pre-registered email	69% ± 3%	30% ± 3%	50% ± 40%
Known pre-registered phone	–	8% ± 3%	8% ± 27%
Last login location	–	8% ± 3%	–
Printed backup code	–	–	–
Security question	–	78% ± 7%	–

Table 4: Frequency that attackers supply incorrect solutions to challenges, exclusively for knowledge-based challenges.

even higher rates, due to users forgetting their backup account's password). Instead, the lower prevention rates for backup emails suggests that some hijackers have stepping-stone access to backup accounts, reducing the security of the overall scheme to that of the weakest trusted identity provider. Given the propensity of bots to also know a victim's backup email address, we believe it likely this access derives from third-party breaches.

Finally, we find that challenges asking for a user's phone number and last login location prevented only 10–26% of hijacking attempts. This information is likely stolen directly from users via a phishing page, something observed by Thomas et al. in their analysis of phishing kits [22]. We find that challenges asking for an account's creation date, backup email, and security questions prevented 68–93% of hijacking attempts. Security here stems from the inability of victims to recall the correct challenge solution and supply it to the phishing prompt. To illustrate this, we calculate the frequency that hijackers provided an incorrect solution for knowledge-based challenge:

$$1 - \frac{passed\ challenges}{total\ challenges - abandoned - skipped}$$

This metric differs from a challenge's prevention rate as we exclude samples where an attacker fails to provide any solution (e.g., they skip or abandon the challenge). As detailed in Table 4, we find that attackers provided an incorrect solution for a victim's account creation date in 70% of cases and an incorrect solution for a victim's security question in 78% of cases. In contrast, attackers supplied an incorrect phone number and login location for only 8% of such challenges. We confirm issues with memorability in non-phishing contexts later in Section 5.

Targeted Attacks. Our sample of targeted attacks emphasizes the weakness of mobile device-based challenges as a protection against concerted adversaries. We find that SMS challenges blocked only 76% of attacks and that device prompts blocked 90% of attacks. The security of device prompts over SMS is statistically significant ($p < 0.002$). Compared to large-scale phishing, targeted attackers are more adept at stealing SMS OTP codes, likely due to more sophisticated tools and more time available to expend per victim, perhaps employing social engineering as descried by Siadati et al. [21]. Our sample size for knowledge-based challenges is too limited to draw conclusions, but we argue such challenges are unlikely to perform any better against a targeted attacker than in the phishing context where memorability is also a factor. While not shown in Table 3, our sample also includes two targeted attacks on accounts with security keys. The second-factor prevented both attacks. This reiterates the importance of domain validation as part of two-factor authentication rather than just proof of access to a physical resource.

4.2 Overall hijacking prevention

While our focus thus far has been on per-challenge prevention rates, in practice, the protection provided by risk analysis is a holistic function of each user's security posture and the frequency of repeated hijacking attempts. We capture this by measuring the fraction of users in our dataset with any successful sign-in from a hijacker. This includes accounts that never received a challenge, or where multiple challenges may have been involved. As our bot dataset includes only a sample of 300K sign-in attempts, we rely on a separate snapshot of all bot sign-in attempts over a month long window. We find that Google's risk analysis engine blocked over 99.99% of automated hijacking attempts by bots and 92% of attacks rooted in phishing. For targeted attacks, we believe protection is better measured by the number of hijacking attempts by attackers—effectively the window in which an identity provider can trigger more rigorous protections. Successful attacks involved 1–9 challenges, while failed attacks involved 1–12 challenges before the attackers abandoned their efforts. We provide only ranges due to the qualitative nature of the sample (N=214) involved.

5 USABILITY OF CHALLENGES

Layered authentication schemes are counterproductive if legitimate users are unable to access their account from new devices and locations. We measure the friction introduced by each challenge in terms of whether users skip, abandon, or provide incorrect solutions to challenges. We also consider whether, despite failing a challenge, users ultimately retry and gain access to their account. Our results indicate that over 97.3% of users can navigate challenges successfully. For comparison, 95.8% of users can navigate password-only authentication according to the same metric.

5.1 Pass rate per login challenge

In the inverse of the security context, our primary metric for usability is whether legitimate account holders can correctly solve a login challenge, thus avoiding account lockout. We consider two sets of users: those logging in from a familiar device or network and those logging in from a new device and network (which as discussed in Section 3, may include hijacking attempts). For each dataset we measure the overall pass rate per challenge as:

$$\frac{passed\ challenges}{total\ challenges}$$

This metric includes multiple failure modes: users providing an incorrect solution to a challenge, users skipping a challenge in favor of an alternative, and users abandoning a sign-in attempt entirely. To isolate issues related to memorability or mistyping, we calculate the adjusted pass rate per challenge as:

$$\frac{passed\ challenges}{total\ challenges - abandoned - skipped}$$

Category	Challenge	Pass rate		Skip rate		Abandon rate		Adjusted pass rate	
		Familiar	*Unfamiliar*	*Familiar*	*Unfamiliar*	*Familiar*	*Unfamiliar*	*Familiar*	*Unfamiliar*
Device-based	Device Prompt	35%	38%	16%	30%	44%	23%	93%	87%
	Device SMS OTP	45%	62%	28%	12%	25%	20%	98%	93%
	Device Settings OTP	13%	9%	16%	34%	64%	48%	72% ± 1%	56% ± 2%
Delegation-based	OTP to backup email	22%	5%	18%	1%	56%	92%	90% ± 1%	94% ± 3%
Knowledge-based	Known pre-reg. email	48%	66%	6%	8%	35%	14%	83%	86%
	Known pre-reg. phone	64%	65%	5%	10%	27%	13%	96%	86%
	Last login location	62%	77% ± 7%	5%	5% ± 7%	23%	7% ± 7%	88% ± 1%	89% ± 7%
	Security question	37%	48% ± 1%	18%	23% ± 1%	33%	13% ± 1%	78%	77% ± 1%
Additional resource	OTP to any phone	43%	42%	9%	17%	45%	34%	96% ± 1%	92%

Table 5: Usability of individual challenges as assessed by their overall pass rates and detailed failure modes including users skipping, abandoning, or incorrectly answering a challenge.

We isolate the other failure modes by calculating the skip rate and abandon rate per challenge. We present our results in Table 5. Unless otherwise noted, all error margins are within 1% at a 95% confidence interval. We note that challenges related to Google's Authenticator app, security keys, account creation time, and proof of access to any email address are not present as they appear exclusively during two-factor authentication and recovery.

Familiar Device & Network. We find that ready access to a trusted mobile device proved challenging for most users. Only 35% of users successfully responded to an on-device prompt and another 45% successfully provided an SMS OTP. Asking users to access their device settings and supply an OTP code proved even more challenging, with only 13% of users passing the challenge. When confronted with a device-based challenge, 25–64% of users outright abandoned the sign-in attempt while another 16–28% opted to solve an alternative challenge. These failure modes highlight the friction introduced by requiring "something you have" for authentication, despite the ubiquity of mobile devices. All of these device-based methods also rely on action flows that users may never have previously encountered. If we isolate challenges solely to those where users attempted to provide a solution, we find that 93% correctly clicked "Yes" to the on-device prompt and 98% supplied the correct SMS code. In aggregate, our findings illustrate that while device-based security challenges provide the best security, there is a non-negligible risk that users may not have immediate access to their device.

OTP codes sent to a backup email had the second lowest pass rate at 22%. Users abandoned a majority of these–56%–either due to an inability to remember their alternative email and password or the overhead involved. We expect that these rates would be higher for service providers that, unlike Google, are not major email providers. Many users have one primary email address that serves as the backup and recovery mechanism for other web services, and many users use Gmail accounts for this, so we might expect that they have difficulty remembering the credentials to an account they do not necessarily use every day. It's possible that a significant fraction of the accounts in our sample are such accounts. Adjusted pass rates of 90% indicate that users had a harder time copying the supplied code, possibly due to having to memorize the code while switching between browser tabs. In comparison, SMS OTPs had higher adjusted pass rates at 98% likely due to having a second screen.

Knowledge-based challenges exhibited the lowest degree of user friction, with pass rates ranging from 37–64%. If we isolate our analysis solely to memorability, we find that users correctly recalled their phone number 96% of the time and their last login location 88% of the time. Secret questions proved the most difficult for users, with users opting to skip them three times more than other knowledge-based challenges and providing a correct solution in only 78% of cases. As previously noted by Bonneau et al., users provide fake responses to security questions which decreases their recall at a later time [2]. Taken as a whole, knowledge-based challenges provide the best overall usability—though at the expense of failing to prevent over 74% of phishing attacks. These trade-offs highlight the decision identity providers must make when determining whether to minimize hijacking or account lockout.

We find that challenges which require only a single, re-usable secret with nothing to carry such as a phone number or email address provide the best memorability and usability. Users struggle with more complex pre-shared secrets such as passwords at secondary verifiers or security questions. These basic knowledge challenges, though simple for users, are also the easiest for attackers to bypass. In contrast, device-based challenges exhibit slightly higher user friction while providing better security protections.

Unfamiliar Device & Network. Compared to familiar devices and networks, users logging in from unfamiliar devices and networks exhibited 0.3–28% higher overall pass rates with the exception of only three challenges as shown in Table 5. We believe this stems from users having both a higher incentive to gain access from their new device and more context for why a challenge occurred. For example, we find users were 20–209% less likely to abandon a challenge. Skip rates were 4–42% higher, indicating users attempted to navigate towards a challenge they could solve. If we isolate our analysis to issues of memorability, we find that users exhibited lower adjusted pass rates for all device-based challenges and resource-based challenges and similar adjusted pass rates for all knowledge-based challenges (with the exception of their phone number). Interpreting these results, we believe that the lower adjusted pass rates for device-based challenges stems from the same place as the lower abandonment rate—users who would otherwise

Category	Challenge	Pass rate				Adjusted pass rate			
		2FA	*Familiar*	*Unfamiliar*	*Recovery*	*2FA*	*Familiar*	*Unfamiliar*	*Recovery*
Device-based	Authenticator OTP	75%	–	–	46% ± 9%	98%	–	–	94% ± 13%
	Device Prompt	76%	35%	38%	55%	97%	93%	87%	89%
	Device SMS OTP	82%	45%	62%	42%	98%	98%	93%	92%
	Device Settings OTP	–	13%	9%	4% ± 19%	–	72% ± 1%	56% ± 2%	25% ± 49%
	Security Key	67% ± 1%	–	–	–	100% ± 1%	–	–	–
Delegation-based	OTP to backup email	–	22%	5%	22%	–	90% ± 1%	94% ± 3%	87% ± 1%
Knowledge-based	Account creation date	–	–	–	13%	–	–	–	34%
	Known pre-reg. email	–	48%	66%	–	–	83%	86%	–
	Known pre-reg. phone	–	64%	65%	61% ± 1%	–	96%	86%	83% ± 1%
	Last login location	–	62%	77% ± 7%	–	–	88% ± 1%	89% ± 7%	–
	Printed backup code	48% ± 1%	–	–	7% ± 6%	83% ± 2%	–	–	64% ± 18%
	Security question	–	37%	48% ± 1%	22% ± 1%	–	78%	77% ± 1%	38% ± 1%
Additional resource	OTP to any email address	–	–	–	19%	–	–	–	70%
	OTP to any phone	–	43%	42%	–	–	96% ± 1%	92%	–

Table 6: Variations in usability across contexts for two-factor authentication, login challenges, and account recovery. Two-factor authentication users have a higher likelihood of carrying a device, while users in recovery struggle more with knowledge-based challenges.

have abandoned the challenge in our familiar device and network experiment are instead attempting to solve it. Additionally, this dataset likely includes some hijacking activity, pulling down aggregate adjusted pass rates. As such, when interpreting usability, we believe pass rates for both our familiar and unfamiliar context datasets represent a lower bound.

5.2 Spectrum of preparedness

Beyond risk analysis at login, challenges also appear in two-factor authentication and account recovery. Combined, these three contexts form a spectrum of preparedness from users who are primed to expect a challenge to users in a forgetful frame of mind. We examine how these contexts influence overall pass rates and memorability in Table 6.

Users enrolled in two-factor authentication had access to their security key for 67% of sign-in attempts, while another 75–82% had access to their mobile device. These pass rates, while roughly twice as high as the login challenge context, illustrate the limitation of expecting users to have a device on hand. At the other end of the preparedness spectrum, we find that device-based challenges displayed during account recovery exhibited pass rates roughly in line with login challenges. Taken as a whole, while improving user familiarity with SMS OTPs or on-device prompts may help increase their pass rate, device accessibility will cause at least a quarter of users to fail such challenges.

Unfortunately, users who struggled to remember their password were equally likely to struggle with accessing their backup account. We find users passed only 22% of delegation-based challenges during recovery, in line with the pass rates of users in a login challenge context.

Finally, knowledge-based challenges proved even more challenging for users in a recovery context. Here, only 13% of users passed a challenge for their account creation date and 22% their security question. Even when we restrict failures to memory alone, we find adjusted pass rates for both challenges fall below 38%. Users were more apt to produce their phone number–succeeding 61% of the time–at rates in line with users in a login challenge context. Printed backup codes also proved difficult to locate, with only 7% of users passing the challenge during recovery. In contrast, two-factor authentication users who selected to submit a code ultimately succeeded in 48% of cases. Our results highlight how recovery introduces even more usability challenges than a login challenge context. Given the issues of memorization at play, we argue that associating a phone number with an account presents the best path towards helping users who forget their password—though this fails to mitigate the risk of phishing without also requiring an SMS OTP code.

5.3 Overall success rate

While exploring challenges in isolation lets us reason about factors that influence usability, the most critical metric for any layered authentication scheme is whether users can solve at least one available challenge and thus access their account. Here, we examine *all* sign-in attempts to Google over a one month period to determine (1) whether users succeeded in logging in during a single challenged session, and (2) whether any user who failed a challenge ultimately succeeded in accessing their account within one week.[5] For this computation, we segment users into three groups: those with two-factor authentication enabled and accessing their account from a familiar device or network; those attempting to recover access to their account from a familiar device or network; and users signing in from an unfamiliar device and network. We note that we lack an equivalent experiment for login challenges to users signing in from a familiar device or network as this scenario never occurs in situ. For comparison, we also calculate the success rate of password-only authentication for users signing in from familiar devices or networks.

[5]We are unable to provide more granular results due to the computation and anonymization involved.

Source of initial failure	Success rate	
	Same session	*Within one week*
Two-factor authentication	88.2%	98.4%
Login challenge,	47.8%	97.3%
Login challenge, device-based	59.3%	97.9%
Account recovery	49.7%	93.8%
Account recovery, device-based	46.6%	94.6%
Password-only	89.5%	95.8%

Table 7: Success rates of users who fail a challenge to gain access to their account within one week. Login challenges and two-factor authentication exhibit less user friction than passwords.

We caution that overall success rate does not equate to lockout as some users may infrequently access their account. Likewise, for login challenges, hijacking activity will depress per-session success rates.

We present our results in Table 7. We find that two-factor authentication exhibits similar user friction compared to password-only authentication. This holds both per-session and overall. Conversely, just 47.8% of challenged sign-ins result in success, with similar rates of 49.7% for account recovery. However, over the course of a week, users have an opportunity to regain access to a familiar device or network and thus eventually access their account. If we isolate our analysis to device-based challenges only, we find that success rates increased for login challenges, but decrease slightly for users attempting to recover their accounts.

These results illustrate that inherent trade-off of login challenges. Identity providers can significantly reduce the risk of hijacking, but the resulting friction increases the risk of account lockout. However, by allowing users to re-try or to find a trusted device or network, users can eventually access their account at rates similar to password-only authentication. The significantly higher success rate of two-factor authentication indicates that education and habituation play a major role in reducing lock out risks.

6 RELATED WORK

Numerous studies have explored knowledge-based challenges in isolation from a usability perspective. Bonneau et al. extensively examined issues with secret questions as secondary authentication measures [2]. They found that users do not answer questions truthfully—with a mind to increase their own security. Consequently, the authors found users were unable to recall their answers during recovery. Similarly, Jakobsson et al. showed that security questions are more effective when responses are restricted to boolean values rather than free-form responses [15]. While improvements to knowledge-based questions in general help to reduce the risk of lockout, our results emphasize that this class of challenges provides only limited protection against phishing and targeted attacks. However, they do provide value against automated attacks and thus there is merit to exploring usability improvements.

Given the push towards two-factor authentication, recent studies have also explored usability challenges with security keys. Das et al. examined barriers to security key adoption and found users were most concerned with losing their second factor and did not feel they needed the extra layer of security [6]. Lang et al. explored security keys and device OTPs from a practical deployment standpoint and found keys provided both better security and reduced the time that users spent authenticating [17]. We believe that on-device prompts can similarly provide a more seamless authentication experience compared to copying an SMS code, though at present SMS is a more familiar experience for most users. In a similar vein, Siadati et al. examined the phishability of SMS codes and found they could socially engineer 50% of participants into handing over an OTP code [21]. With improvements to warning language included in the text of the SMS, only 8% of participants handed over their OTP code.

Finally, in terms of potential new forms of challenges, research has recently started to explore social challenges. Alomar et al. provide an overview of proposals in this space and potential trade-offs [1]. Brainard et al. proposed vouching by a single additional party [4], where authentication becomes a transitive property. Alternatively, Jain et al. proposed asking users to identify recent activity from friends or to identify a list of friends—more akin to knowledge-based challenges [14]. We are unaware of any large-scale deployments of such schemes, but our taxonomy is comprehensive enough to capture the challenges in this space.

7 CONCLUSION

While the quest to replace passwords continues, authentication challenges offer an effective barrier against remote hijacking threats rooted in password theft. From a practical standpoint, we found that challenges, in conjunction with risk-aware authentication, blocked over 99.99% of automated hijacking attempts and 92% of attacks rooted in phishing at Google. These protections come at a cost of increased failed sign-in attempts from legitimate users, but with eventual success rates at levels similar to password-only authentication. We caution our metrics represent a current snapshot of an arms race. But unlike many cybercrime threats, users can take simple proactive steps to dramatically increase their security. Users who associate a device with their account can reduce their phishing risk by up to 99%. This approach provides similar levels of protection to two-factor authentication while removing the requirement of always having a device on-hand. However, as our data showed, risk-aware authentication cannot reliably protect against repeated, targeted hijacking attempts that involve social engineering. Here, security keys provide the only 100% guarantee of protection against remote password theft. By providing risk-aware authentication as a default protection and security keys as opt-in, users can choose the security and usability trade-off that best works for them.

8 ACKNOWLEDGEMENTS

We thank Elie Bursztein and Tadek Pietraszek for their feedback on developing our study. This work was supported in part by NSF award CNS-1717062 and gifts from Comcast, Google, Linkedin, and YouGov.

REFERENCES

[1] Noura Alomar, Mansour Alsaleh, and Abdulrahman Alarifi. Social authentication applications, attacks, defense strategies and future research directions: a systematic review. *IEEE Communications Surveys & Tutorials*, 2017.

[2] Joseph Bonneau, Elie Bursztein, Ilan Caron, Rob Jackson, and Mike Williamson. Secrets, lies, and account recovery: Lessons from the use of personal knowledge questions at google. In *Proceedings of the International Conference on World Wide Web*, 2015.

[3] Joseph Bonneau, Cormac Herley, Paul C Van Oorschot, and Frank Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *Proceedings of the Symposium on Security and Privacy*, 2012.

[4] John Brainard, Ari Juels, Ronald L Rivest, Michael Szydlo, and Moti Yung. Fourth-factor authentication: somebody you know. In *Proceedings of the Conference on Computer and Communications Security*, 2006.

[5] Marco Cova, Christopher Kruegel, and Giovanni Vigna. There is no free phish: an analysis of "free" and live phishing kits. In *Proceedings of the USENIX Workshop on Offensive Technologies*, 2008.

[6] Sanchari Das, Andrew Dingman, and L Jean Camp. Why johnny doesn't use two factor a two-phase usability study of the fido u2f security key. In *Proceedings of the International Conference on Financial Cryptography and Data Security*, 2018.

[7] Duo Security. Guide to two-factor authentication. https://guide.duo.com/, 2018.

[8] Facebook. Facebook has users identify friends in photos to verify accounts, prevent unauthorized access. https://www.adweek.com/digital/facebook-photos-verify/, 2010.

[9] Facebook. Improving account security with delegated recovery. https://www.facebook.com/notes/protect-the-graph/improving-account-security-with-delegated-recovery/1833022090271267/, 2017.

[10] David Freeman, Sakshi Jain, Markus Dürmuth, Battista Biggio, and Giorgio Giacinto. Who are you? a statistical approach to measuring user authenticity. In *Proceedings of the Network and Distributed System Security Symposium*, 2016.

[11] Google. Confirm your identity using your android device. https://support.google.com/accounts/answer/6046815?hl=en, 2018.

[12] Google. reCAPTCHA v3. https://developers.google.com/recaptcha/docs/v3, 2018.

[13] Google. Sign in faster with 2-step verification phone prompts. https://support.google.com/accounts/answer/7026266?&hl=en, 2018.

[14] Sakshi Jain, Juan Lang, Neil Zhenqiang Gong, Dawn Song, Sreya Basuroy, and Prateek Mittal. New directions in social authentication. In *Proceedings of the Workshop on Usable Security*, 2015.

[15] Markus Jakobsson, Liu Yang, and Susanne Wetzel. Quantifying the security of preference-based authentication. In *Proceedings of the Workshop on Digital Identity Management*, 2008.

[16] Mark Kaufman. Google Authenticator will add a formidable layer of protection to your e-mail account. https://mashable.com/2017/10/29/how-to-set-up-google-authenticator, 2017.

[17] Juan Lang, Alexei Czeskis, Dirk Balfanz, Marius Schilder, and Sampath Srinivas. Security keys: Practical cryptographic second factors for the modern web. In *Proceedings of the International Conference on Financial Cryptography and Data Security*, 2016.

[18] Microsoft. Azure active directory identity protection. https://docs.microsoft.com/en-us/azure/active-directory/active-directory-identityprotection, 2018.

[19] Grzegorz Milka. Anatomy of account takeover. In *Enigma*, 2018.

[20] Ariana Mirian, Joe DeBlasio, Stefan Savage, Geoffrey M. Voelker, , and Kurt Thomas. Hack for hire: Exploring the emerging market for account hijacking. In *Proceedings of The Web Conf*, 2019.

[21] Hossein Siadati, Toan Nguyen, Payas Gupta, Markus Jakobsson, and Nasir Memon. Mind your smses: Mitigating social engineering in second factor authentication. *Computers & Security*, 2017.

[22] Kurt Thomas, Frank Li, Ali Zand, Jacob Barrett, Juri Ranieri, Luca Invernizzi, Yarik Markov, Oxana Comanescu, Vijay Eranti, Angelika Moscicki, et al. Data breaches, phishing, or malware?: Understanding the risks of stolen credentials. In *Proceedings of the Conference on Computer and Communications Security*, 2017.

[23] Kurt Thomas and Angelika Moscicki. New research: Understanding the root cause of account takeover. https://security.googleblog.com/2017/11/new-research-understanding-root-cause.html, 2017.

On Attribution of Recurrent Neural Network Predictions via Additive Decomposition

Mengnan Du, Ninghao Liu, Fan Yang, Shuiwang Ji, Xia Hu
Department of Computer Science and Engineering, Texas A&M University
{dumengnan,nhliu43,nacoyang,sji,xiahu}@tamu.edu

ABSTRACT

RNN models have achieved the state-of-the-art performance in a wide range of text mining tasks. However, these models are often regarded as black-boxes and are criticized due to the lack of interpretability. In this paper, we enhance the interpretability of RNNs by providing interpretable rationales for RNN predictions. Nevertheless, interpreting RNNs is a challenging problem. Firstly, unlike existing methods that rely on local approximation, we aim to provide rationales that are more faithful to the decision making process of RNN models. Secondly, a flexible interpretation method should be able to assign contribution scores to text segments of varying lengths, instead of only to individual words. To tackle these challenges, we propose a novel attribution method, called REAT, to provide interpretations to RNN predictions. REAT decomposes the final prediction of a RNN into additive contribution of each word in the input text. This additive decomposition enables REAT to further obtain phrase-level attribution scores. In addition, REAT is generally applicable to various RNN architectures, including GRU, LSTM and their bidirectional versions. Experimental results demonstrate the faithfulness and interpretability of the proposed attribution method. Comprehensive analysis shows that our attribution method could unveil the useful linguistic knowledge captured by RNNs. Some analysis further demonstrates our method could be utilized as a debugging tool to examine the vulnerability and failure reasons of RNNs, which may lead to several promising future directions to promote generalization ability of RNNs.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Neural networks**.

KEYWORDS

Deep learning interpretation; Recurrent neural network; Text classification; Sentiment analysis

ACM Reference Format:
Mengnan Du, Ninghao Liu, Fan Yang, Shuiwang Ji, Xia Hu. 2019. On Attribution of Recurrent Neural Network Predictions via Additive Decomposition. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313545

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313545

1 INTRODUCTION

The RNNs, such as LSTM [14] and GRU [5], have achieved the state-of-the-art performance in a variety of text classification tasks. These deep text classification models have became increasingly deployed in different web applications, including sentiment classification [42], named entities recognition [32], textual entailment [4], etc. Despite the superior performance, RNNs are often criticized by their lack of interpretability, and are often treated as black-boxes [21]. The lack of understanding of the mechanism behind RNN predictions not only reduces the acceptance from end-users towards the deployed predictors, but also limits ability of system developers in diagnosing the model, searching the reasons for wrong predictions, and improving the model architectures. Therefore, it is highly desirable to explore the interpretability of RNNs, so as to provide insight of how they process text inputs and make inferences therefrom.

It is possible to provide interpretability for RNNs from two perspectives: adding interpretable components to RNN models or performing post-hoc attribution [9]. The former category integrates interpretability directly into the structure of deep models, and usually resorts to attention mechanism [20, 44]. However, the attention components only provide an indirect indicator of contribution scores. For classification tasks, it is not clear which specific class these attention weights contribute to [27]. In contrast, the philosophy of post-hoc attribution is to provide interpretation to predictions made by a pre-trained black-box model without any architectural modification. Post-hoc attribution aims to attribute the prediction of a model to its input features, e.g., words in a text classification task, and produce a heatmap as interpretation, indicating the contribution of each feature to a particular class of interest. In this paper, we follow the post-hoc attribution strategy, as it could provide understandable rationale to the given predicted label from the model while keeping the underlying model status intact.

Although post-hoc attribution has been extensively studied for understanding MLP and CNNs [10], attribution for RNN predictions is still a technically challenging problem. First, one challenge lies in how to guarantee that the interpretations are indeed faithful to the original model. Many previous attribution work, including back-propagation based methods [7, 13], perturbation based methods [16, 19], and local approximation based methods [35, 36], all follow the philosophy of *local interpretation*. That is, they generate interpretable approximation of the original model around the neighborhood of a given prediction to be explained. However, it is not guaranteed that the generated interpretations accurately reflect the decision making process of the original model [6, 12]. In this case, it is hard to tell whether an unexpected interpretation is caused by misbehavior of the model or limitation of the attribution method. Second, it is challenging to develop a flexible attribution method which could generate attribution scores to text

segments (e.g., phrases) of varying lengths. Prior work for RNNs attribution mainly focuses on identifying *word-level contribution scores* [7, 35, 41], which assigns a real-value score for each of the words, indicating the extent to which it contributes to a particular prediction. However, word-level attributions fail to explain why RNNs are successful to process sequences where the order of the data entries matters. Consider an example of sentence sentiment analysis, "I do not dislike cabin cruisers." expresses a neutral opinion. Word-level attribution methods may accurately identify that the word "dislike" has a negative contribution for this prediction, but they fail to capture that the negation word "not" has shifted its polarity and the word combinations "not dislike" have a positive impact for model prediction.

In this paper, we propose a decomposition based attribution method, called REAT (REcurrent ATtribution Method), to provide interpretation for given predictions made by a RNN in a faithful and flexible manner. Through modeling the information flowing process of the hidden state vectors in RNN models, REAT could decompose the final prediction of a RNN into additive contribution of each word in the input text. Since REAT is constructed by directly leveraging the information propagation process from hidden state vectors to the output layer, it enjoys the benefit of high faithfulness to the original RNN model. This method not only can quantify the contribution of each individual word to a prediction, but also could naturally be applied to identify the contribution of word sequences. It thus enables the illumination of how RNNs make use of sequential information, as well as how they capture long-term dependencies. In addition, REAT is widely applicable to different recurrent architectures, including LSTMs and GRUs, and their bidirectional versions. Furthermore, based on the observation that language usually exhibits hierarchical structures, we expand REAT to a hierarchical attribution method, to represent the contributions at different levels of granularity. The major contributions of this paper are summarized as follows:

- We propose a RNN attribution method based on additive decomposition, called REAT, which could provide both word-level and phrase-level attribution scores.
- REAT is applicable to different RNN architectures. We demonstrate its applicability to three standard RNN architectures, including GRU, LSTM and Bidirectional GRU.
- Experimental results on two sentiment analysis datasets validate the faithfulness and interpretability of the proposed method.
- We demonstrate that REAT could be utilized as a debugging tool to analyze the useful linguistic knowledge captured by RNNs and examine the vulnerability and failure reasons of RNNs.

2 PRELIMINARIES

In this section, we start by introducing post-hoc attribution that serves as the basic attribution scheme in this paper. Then we introduce three representative RNN architectures as well as the output layer which transforms the hidden state into probability output.

2.1 Post-hoc Attribution

Notations: Consider a typical multi-class text classification task, a RNN-based classification model can be denoted as $f : X \rightarrow Y$, where X is the text space, and $Y = \{1, ..., C\}$ denotes the set of output classes. The RNN model accepts an instance $\mathbf{x} \in X$ as input, and maps it to an output class: $f(\mathbf{x}) = c \in Y$. Assume the input is composed of a sequence of T words: $\mathbf{x} = \{x_1, ..., x_T\}$ and each word $x_t \in \mathbb{R}^d$ denotes the embedding representation of the t-th word. The high level idea of post-hoc attribution is to attribute the prediction $f(\mathbf{x})$ of a RNN model to its input features $\mathbf{x}$ and output a heatmap indicating the contribution of each feature $x_t \in \mathbf{x}$ to a particular class of interest c. Specifically, we target to generate attributions for a RNN prediction which is specified as follows:

Phrase-level Attribution: We first partition the input text into meaningful phrases (coherent pieces of text) and then attach an attribution score to each individual phrase. Given one index phrase as query. The index set of a phrase that we want to calculate its attribution score is denoted as: $A = \{q, ...r\}$ where $1 \leq q \leq r \leq T$. We indicate the attribution score for the targeting phrase as $S(\mathbf{x}_A)$.

2.2 RNN Architectures

RNNs come in many variants with different architectures, which results in different mapping functions f. In this paper, we discuss three representative RNN architectures that are fundamental and have been widely used in many applications. The common formulations in different RNN architectures motivate the design of our interpretation approach.

LSTM: In a Vanilla RNN, at any time step in a sequence, the hidden state h_t is calculated based on its previous hidden state h_{t-1} and the current input vector x_t:

$$h_t = \tanh(W_{hx}x_t + W_{hh}h_{t-1} + b_h). \tag{1}$$

Comparing to the Vanilla RNN, LSTM makes two major changes. Firstly, LSTM introduces a cell state c_t that serves as an explicit memory. Secondly, instead of simply updating the hidden state h_t in Eq. (1), LSTM uses three gates: input gate i_t, forget gate f_t, and output gate o_t to update the hidden state h_t.

$$\begin{aligned} i_t &= \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \\ f_t &= \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \\ o_t &= \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \\ g_t &= \tanh(W_{gx}x_t + W_{gh}h_{t-1} + b_g) \\ c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\ h_t &= o_t \odot \tanh(c_t). \end{aligned} \tag{2}$$

where each W and b represent weight matrix and bias vector respectively, and $\odot$ denotes element-wise multiplication.

GRU: GRU makes some slight modifications on the basis of LSTM. It only has two gates, i.e., reset gate r_t and update gate u_t. Besides, it merges the cell state and hidden state into a single hidden state h_t. The updating rule of hidden state is denoted as follows.

$$\begin{aligned} r_t &= \sigma(W_{rx}x_t + W_{rh}h_{t-1} + b_r) \\ u_t &= \sigma(W_{ux}x_t + W_{uh}h_{t-1} + b_u) \\ g_t &= \tanh(W_{gx}x_t + r_t \odot W_{gh}h_{t-1} + b_g) \\ h_t &= u_t \odot h_{t-1} + (1 - u_t) \odot g_t. \end{aligned} \tag{3}$$

Bidirectional GRU: This model is constructed by putting two independent GRUs together. The word sequences are fed into one GRU in normal time order, and in reverse time order into another. For each network, the hidden state is updated using the same rule as

Eq. (3). For the sake of brevity, we use subscripts n and r to represent the *normal* and *reverse* network respectively. For classification tasks, the final hidden vector fed into the output layer is constructed by concatenating the hidden vector at time step T for the normal GRU and the hidden vector at time step 1 for the reverse one:

$$h = h_{T,n} \oplus h_{1,r}, \tag{4}$$

where symbol $\oplus$ denotes concatenation operation of two vectors. Both $h_{T,n}$ and $h_{1,r}$ contain the information of the whole text. Hidden state $h_{T,n}$ encodes the information of the normal text (from x_1 to x_T), while $h_{1,r}$ captures the information in the inverse text (from x_T to x_1). In the remaining part of this paper, we use BiGRU to denote Bidirectional GRU.

2.3 RNN Output Layer

To serve the purpose of multi-class text classification, a discriminative layer is added after the activation vector h_T of the last hidden layer at time step T. This layer takes the hidden state h_T as input and turns it into a logit vector z using a weight matrix W:

$$z = Wh_T, \tag{5}$$

and then produces the class probabilities using a softmax layer which converts the logit z_c for a class c into a probability y_c, by comparing it with other logit values:

$$y_c = \text{softmax}(z)_c = \frac{\exp(z_c)}{\sum_{k=1}^{C} \exp(z_k)}. \tag{6}$$

3 METHODOLOGY

In this section, we introduce the proposed attribution method for explaining RNN predictions. We first present a general method for phrase-level RNN prediction attribution, and then apply it to three widely used RNN architectures. Finally, we expand the method to enable hierarchical attribution schemes.

3.1 Phrase-level Attribution Method for Recurrent Models

3.1.1 A Naive Attribution Approach. From the last section, we know that RNNs possess a series of hidden state vectors $\{h_t\}_{t=1,\ldots,T}$, where each vector h_t stores information about the past input blocks, ranging from time step 1 to t. A crucial property of the hidden state vector is that it is updated from time step to time step. Knowing how much information is accumulated at each time step enables us to derive the contribution of that time step towards the final prediction. Intuitively, we denote the response of the RNN model to word $\mathbf{x}_t$, and thereby the information gained at step t, as follows:

$$\tilde{h}_t = g(\mathbf{x}_t) = h_t - h_{t-1}. \tag{7}$$

In this way, we can consider the final hidden state vector h_T to contain information accumulated at T time steps, denoted as: $h_T = \sum_{t=1}^{T} \tilde{h}_t = \sum_{t=1}^{T} (h_t - h_{t-1})$. Then, we can decompose the logit z_c (see Eq. (5)) for target class c using a sequence of factors:

$$z_c = W_c h_T = \sum_{t=1}^{T} W_c (h_t - h_{t-1}). \tag{8}$$

It can be treated as the additive contribution of each word in the input $\mathbf{x}$ to the output logit (unnormalized output probability) of class c. Therefore, the contribution of word $\mathbf{x}_t$ towards the logit z_c can be calculated as:

$$S(\mathbf{x}_t) = W_c (h_t - h_{t-1}). \tag{9}$$

However, this decomposition has a severe shortcoming. The underlying assumption of this formulation is that all evidence accumulated up to time step $t-1$ has been transferred to time t. This violates the updating rules of all the three architectures listed in Sec. 2.2. Actually, it fails to take into consideration the forgetting mechanism of RNNs. For example, both LSTM and GRU have explicit gates to model the forgetting and remembering mechanism, which serves the purpose of controlling information flow and calculating how much proportion of information derived from previous time steps should be kept.

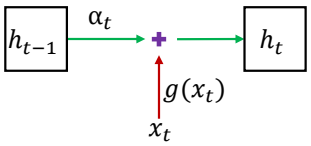


Figure 1: An illustration of the summarized updating rule for the RNN hidden state vectors.

3.1.2 The Proposed Recurrent Attribution Method (REAT). Many variants of RNNs share a similar form of hidden state updates. We summarize a common rule which can be applied to different recurrent network architectures, as illustrated in Fig. 1. The rule maintains a hidden state h_t which summarizes information for past sequences. Also, at each time step t, the hidden state is updated using the following equation:

$$h_t = \alpha_t \odot h_{t-1} + \tilde{h}_t, \tag{10}$$

where $\alpha_t \in [0,1]^{d'}$ so that only partial evidence obtained by RNN from previous $t-1$ steps is brought to the time step t. Here d' is the dimension of hidden state vectors. A higher value of α means that the RNN model preserves more important information from previous time steps. Here $\tilde{h}_t = g(\mathbf{x}_t)$ denotes the evidence that a RNN obtains at time step t, but note that it is no longer defined as in Eq. (7). Some RNN architectures obey this rule exactly, like GRU, while some other architectures follow this rule approximately, such as LSTM. Based on this hidden state updating rule, we can iteratively trace back the generation of h_T and decompose the logit value z_c into the following form:

$$z_c = W_c h_T = \sum_{t=1}^{T} W_c (\tilde{h}_t \odot \prod_{k=t+1}^{T} \alpha_k). \tag{11}$$

Replacing each $\tilde{h}_t$ with $h_t - \alpha_t \odot h_{t-1}$, we can reformulate the additive decomposition as:

$$z_c = \sum_{t=1}^{T} W_c [(h_t - \alpha_t \odot h_{t-1}) \odot \prod_{k=t+1}^{T} \alpha_k]. \tag{12}$$

The main benefit of Eq. (12) comparing to Eq. (11) is that we do not need to know the exact form of $\tilde{h}_t$. Merely knowing the hidden state vector h_t and the updating parameter vector α_t will be sufficient to derive the decomposition. Eq. (12) can be considered as the additive

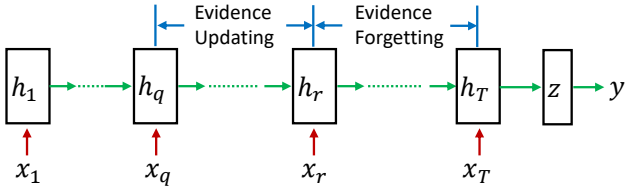


Figure 2: An illustration of the proposed method.

contribution of each word $\mathbf{x}_t$ towards the logit z_c, which is the unnormalized probability for target class c. By taking out the term relevant to time step t, we can derive the contribution value for a single word $\mathbf{x}_t$ at the current time step:

$$S(\mathbf{x}_t) = W_c[\underbrace{(h_t - \alpha_t \odot h_{t-1})}_{Updating} \odot \underbrace{\prod_{k=t+1}^{T} \alpha_k}_{Forgetting}]. \tag{13}$$

The above formulation within the square brackets is the element-wise multiplication of two terms. The left term denotes the updating evidence from time $t-1$ to t, i.e., the contribution to class c by the input word $\mathbf{x}_t$. The right term represents the forgetting mechanism of RNN. The evidence that a RNN has gathered at time step t gradually diminishes as the time increases from $t+1$ to the final time step T. That is to say, only part of the updating evidence will have impacts on the classification task at time step T.

Based on the word-level additive attribution formulation in Eq. (13), we can conveniently derive the phrase-level attribution. For a phrase $\mathbf{x}_A$, where $A = \{q, ...r\}$, $1 \le q \le r \le T$, its attribution score $S(\mathbf{x}_A)$ can be denoted as:

$$S(\mathbf{x}_A) = W_c[\underbrace{(h_r - \prod_{j=q}^{r} \alpha_j \odot h_{q-1})}_{Updating} \odot \underbrace{\prod_{k=r+1}^{T} \alpha_k}_{Forgetting}]. \tag{14}$$

Similar to word-level attribution, phrase-level attribution $S(\mathbf{x}_A)$ also contains two terms. The left term within the square brackets in Eq. (14) represents the updating evidence from time step $q-1$ to time r, while the right term denotes how much percentage of the evidence has been forgotten from time $r+1$ to T. We illustrate this process in Fig. 2. It is worth noting that the key component here to derive phrase-level attribution score for a RNN classifier is to obtain the hidden state vectors and the updating vectors. Usually, only one feed forward operation is needed to derive phrase-level attribution score, which can be implemented efficiently.

3.2 Applications to Specific Architectures

In this section, we apply the proposed idea to three standard RNN architectures, including LSTM, GRU and BiGRU.

3.2.1 GRU Attribution. The hidden state vector updating rule for GRU is written as:

$$h_t = u_t \odot h_{t-1} + (1 - u_t) \odot g_t, \tag{15}$$

which conforms with the paradigm in Eq. (10). Therefore, for a phrase $\mathbf{x}_A$, where $A = \{q, ...r\}$, $1 \le q \le r \le T$, we can directly replace α_t in Eq. (14) with the updating gate vector u_t of the GRU model, and obtain the phrase-level attribution score $S(\mathbf{x}_A)$:

$$S(\mathbf{x}_A) = W_c[(h_r - \prod_{j=q}^{r} u_j \odot h_{q-1}) \odot \prod_{k=r+1}^{T} u_k]. \tag{16}$$

3.2.2 LSTM Attribution. Although it is difficult to directly match the LSTM updating rule of hidden state h_t in Eq. (2) to the paradigm in Eq. (10), the update of the cell state c_t adheres to the expected updating format in Eq. (10). It is denoted as: $c_t = f_t \odot c_{t-1} + i_t \odot g_t$, where f_t is the forgetting gate of LSTM.

Based on the updating rule from the cell state c_t to hidden state: $h_t = o_t \odot tanh(c_t)$, approximately we can obtain:

$$c_t \sim \frac{h_t}{o_t}, \tag{17}$$

where the right term is a element-wise division. Thus we approximate the updating rule of the hidden state vector for LSTM as below:

$$h_t = \frac{f_t \odot o_t}{o_{t-1}} \odot h_{t-1} + \tilde{h}_t. \tag{18}$$

Then, we can further decompose the final hidden state vector h_T into the following formulation:

$$h_T = \sum_{t=1}^{T} (h_t - \frac{f_t \odot o_t}{o_{t-1}} \odot h_{t-1})) \odot \prod_{k=t+1}^{T} \frac{f_k \odot o_k}{o_{k-1}}. \tag{19}$$

For a phrase $\mathbf{x}_A$, where $A = \{q, ...r\}$, $1 \le q \le r \le T$, we can obtain the attribution score $S(\mathbf{x}_A)$:

$$S(\mathbf{x}_A) = W_c[(h_r - \prod_{j=q}^{r} \frac{f_j \odot o_j}{o_{j-1}} \odot h_{q-1}) \odot \prod_{k=r+1}^{T} \frac{f_k \odot o_k}{o_{k-1}}]. \tag{20}$$

3.2.3 BiGRU Attribution. In this section, we illustrate how the proposed method can be applied to bidirectional architectures. The last hidden state $h_{T,n}$ at time step T of the normal GRU uses the identical decomposition as in Sec. 3.2.1. As for the reverse GRU, we use the hidden state vector at time step 1 in order to capture the information from 1 to T, which can be decomposed as follows:

$$h_{1,r} = \sum_{t=1}^{T} (h_{t,r} - u_{t,r} \odot h_{t+1,r}) \odot \prod_{k=1}^{t-1} u_{k,r}. \tag{21}$$

Recall that the final hidden vector fed into the classification layer is the concatenation of the hidden vector at time step T for the normal GRU and the hidden vector at time step 1 for the reverse GRU, i.e., $h = h_{T,n} \oplus h_{1,r}$. As such, the logit value is computed as $z_c = W_c(h_{T,n} \oplus h_{1,r})$. To obtain the attribution score for a phrase $\mathbf{x}_A$, where $A = \{q, ...r\}$, we first calculate the updated hidden evidence for the normal network and the reverse network respectively. We then concatenate these two updated hidden evidence, and multiply it with W_c to produce the contribution scores.

$$\begin{aligned} S(\mathbf{x}_A) = & W_c[((h_{r,n} - \prod_{j=q}^{r} u_{j,n} \odot h_{q-1,n}) \odot \prod_{k=r+1}^{T} u_{k,n}) \oplus \\ & ((h_{q,r} - \prod_{j=q}^{r} u_{j,r} \odot h_{r+1,r}) \odot \prod_{k=1}^{q-1} u_{k,r})]. \end{aligned} \tag{22}$$

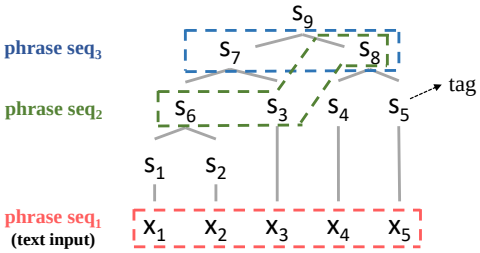


Figure 3: The proposed phrase-based attribution method is naturally compatible with sentence parsing.

	Regular expression
Verb chunks	$< VERB > * < ADV > * < PART > * < VERB > + < PART > *$
Noun chunks	$< DET >?(< NOUN > + < ADP\|CONJ >)* < NOUN > +$

Table 1: Regular expression matching.

3.3 Automatic Attribution Generation

The developed method so far can only calculate the contribution score of a phrase whose beginning and ending in $\mathbf{x}$ are already specified. A natural question that arises is: how to comprehensively analyze and combine the contributions of different parts of an input text, so as to explain the response of RNNs spanning on the whole text. A naive way is to exhaustively select phrases of different length and starting points, and compute their contribution scores respectively. However, such search scheme will span over a lot of redundant and trivially overlapping text segments. In this case, the redundant amount of interpretation information may be overwhelming to users. Traditional word-level interpretation can be seen as a special case of our method by simply setting the phrase length as one, but its flexibility is limited.

To design an interpretation scheme with both desirable understandability and flexibility, we apply text parsing into the attribution method. The general idea is illustrated in Fig. 3. The parse tree is derived from certain existing text parsing algorithm. Here $\mathbf{x}_n$ denotes a word, and $\mathbf{s}_i$ denotes a non-terminal symbol (i.e., a POS tag or a phrase type). Our attribution method will be applied to each of the selected phrase sequences. The phrases within each sequence are semantically meaningful and mutually non-overlapping. In this way, we obtain interpretation of different granularity on a sentence. The interpretation is thus more compatible with the cognition habits of human.

In this work, we partition sentences using regular expression matching [25] to divide each sentence into meaningful phrases. Specifically, we use two regular expressions shown in Tab. 1 to partition the sentence into continuous noun and verb chunks. Take the sentence "The movie does n't serve up lot of laughs." for example. A middle-level phrase sequence obtained from parsing could be "{The movie} {does n't serve up} {lot of laughs}.", which correspond to noun chunk, verb chunk, and noun chunk respectively. Then we feed the sentence into the RNN and the attribution method. The output is a sequence of phrases along with corresponding attribution scores, capturing the most significant phrases contributing to the network prediction during the prediction lifetime. In practice, each original text may end up with 2 (word-level and phrase-level) or 3 (word-level, phrase-level, and clause-level, where clause denotes part of a sentence or a complete sentence containing a verb) hierarchies, and users can obtain a hierarchical interpretation of the prediction.

	SST2	Yelp
# **training texts**	6920	80000
# **development texts**	872	1999
# **test texts**	1821	5492

Table 2: Dataset Statistics

	SST2	Yelp
LSTM	80.4%	92.2%
GRU	81.9%	90.2%
BiGRU	80.9%	93.5%

Table 3: Prediction accuracy of the three RNN architectures on SST2 and Yelp dataset.

4 EXPERIMENTS

In this section, we present the experimental results in order to answer the following research questions (**RQ**s).

- **RQ1** - Does the generated attribution faithfully reflect how the original RNN model makes predictions?
- **RQ2** - Does the proposed RNN attribution method outperform alternative approaches in terms of interpretability?
- **RQ3** - Could hierarchical interpretation comprehensively capture the evidences for prediction at different levels of granularity?
- **RQ4** - How to use the proposed attribution method to analyze the useful insights captured by RNNs?
- **RQ5** - How to use the proposed attribution method to analyze the misbehavior of RNNs and ultimately promote RNNs' generalization capability?

4.1 Datasets

We conduct our experiments on two publicly available sentiment analysis datasets, whose statistics are shown in Tab. 2.

Stanford Sentiment Treebank 2 (SST2) [39] - It contains 2 classes (negative and positive). The numbers of instances for training set, development set and test set are 6920, 872, and 1821 respectively.

Yelp Polarity (Yelp) [46] - It consists of reviews originally extracted from the Yelp Dataset Challenge 2015 data. Zhang et al. [46] constructed the Yelp reviews polarity dataset by considering stars 1 and 2 as negative, and considering 4 and 5 as positive. The numbers of instances for training set and test set are 560,000 and 38,000 respectively. We use a subset of this dataset by filtering out texts whose length is larger than 40, and then randomly select part of the samples from the training set as development set. Ultimately, the dataset contains 80000, 1999, and 5492 instances for training, development, and test set respectively, with each polarity occupying around half of the instances in each set.

4.2 Experimental Setup

For each classification model used in our experiments, it contains a word embedding layer to transform words to fixed length representation vectors, a recurrent network layer to transform word embeddings to hidden state vectors, and a classification layer for output. Specifically, the pre-trained 300-dimensional word2vec word embedding [25] is utilized to initialize the embedding layer. For those words that do not exist in word2vec, we initialize their embedding vectors with some random values. The dimension of hidden state vectors is 150 for both LSTM and GRU, and 300 for BiGRU. The classification layer is composed of a dense layer and a softmax nonlinear transformation. The Adam optimizer [18] is utilized to optimize these models and the learning rate is fixed to 10^{-3}. We train each model for 20 epoches and select the one with the best performance on the development set. Note that we freeze the embedding layer when training all models on SST2, while fine-tune the embedding parameters when training on Yelp. Empirical results show that this can lead to better prediction performance for models on both datasets. We evaluate the test accuracy with the best performing model on the development set, and the performance statistics on SST2 and Yelp dataset are reported on Tab. 3.

4.3 Baseline Methods

We evaluate the proposed REAT method by comparing it with five baseline approaches.

- **Vanilla gradient (VanillaGrad)** [13]: Compute gradients of the output prediction with respect to individual entries in word embedding vectors, and use the L2 norm to reduce each vector of the gradients to a single attribution value, representing the contribution of each single word.
- **Integrated gradient (InteGrad)** [41]: Integrate all Vanilla gradients using a linear interpolation between a baseline input and the original input. Here the baseline input are sentences whose word embedding values are all set to zeros.
- **Gradient times input (GradInput)** [7]: First calculate the gradient of the output with respect to word embedding, and then use dot product of the gradient vector and word embedding vector as the contribution score for a word.
- **LIME** [35]: A model-agnostic interpretation method. It approximates the behavior of a RNN in the neighborhood of a given input using an interpretable white-box model. Here, the interpretable model is a sparse linear model.
- **NaiveREAT**: A simplified variant of the proposed method, as introduced in Sec. 3.1.1. The attribution score for a single word is calculated using Eq. (9).

It is worth noting that except NaiveREAT, the other four baseline methods could only derive word-level contribution scores. To get phrase-level or sentence-level contribution scores, we sum up the scores of all word within a phrase or a sentence.

4.4 Attribution Faithfulness Evaluation

In this section, we evaluate the faithfulness of the attribution methods with respect to the target RNN models. We want to assess whether the attribution results correctly reflect the prediction behavior of RNNs. In general, the faithfulness of an attribution method is evaluated by deleting the sentence of the highest contribution score and observing the prediction changes of the target RNNs [29]. Specifically, the attribution method first produces contribution scores for sentences in the text. Then, it is expected that once the most important sentence is deleted, it will cause the probability value to significantly drop for the target class. Here we define faithfulness score as the metric:

$$S_{\text{faithfulness}} = \frac{1}{N}\sum_{i=1}^{N}(y(\mathbf{x}^{(i)}) - y(\mathbf{x}^{(i)}_{\setminus A})), \tag{23}$$

where A denotes the sentence identified as the most predictive for a prediction, and N is the total number of texts in the dataset. An advantage of this metric is that no knowledge is required of ground truth labels. Theoretically this metric can also be utilized to evaluate word-level and phrase-level attributions. However, empirically, we find some irrelevant words or phrases could also lead to big probability drop, because they cause grammar or syntactic errors instead of really changing the semantics [26, 37]. Therefore, we only use this metric in sentence-level attribution scenarios.

Models	GRU	LSTM	BiGRU
VanillaGrad	0.272	0.243	0.068
InteGrad	0.255	0.253	0.113
GradInput	0.301	0.199	0.178
Lime	0.209	0.188	0.092
NaiveREAT	0.213	0.207	0.114
REAT	**0.311**	**0.318**	**0.196**

Table 4: Comparison about attribution faithfulness between our method and the baseline methods.

We search in SST2 dataset for texts that contain two sentences and use the word "but" as conjunction, and obtain a set with 142 texts. The faithfulness scores for different attribution methods on SST2 dataset are reported in Tab. 4. The proposed method consistently outperforms the baseline methods for all the three architectures. This result demonstrates two advantages of REAT: (1) the generated interpretations are highly faithful to original RNN, (2) the generated phrase-level interpretations are accurate. Since the NaiveREAT method does not consider the forgetting mechanism, it may assign false positive contribution scores for the first sentence in the testing texts. As a result, it is not faithful to the target model and achieves relatively low faithfulness score comparing to REAT. Besides, for the other baseline methods, they can only output word-level attribution scores. Since the word-level scores are not sufficiently faithful to the target model, the sentence-level scores calculated by summing up the word-level scores will further deviate from the prediction of the target RNN model. As a result, these methods have only limited faithfulness performance.

4.5 Attribution Interpretability Evaluation

We evaluate the interpretability of the proposed method to show whether the generated interpretations are reasonable from some fundamental perspectives of human comprehension. We only search for texts that contain both positive adjective words and negative adjective words with obvious sentiment bias. The searching criterion is whether a word belongs to the human annotated lists containing

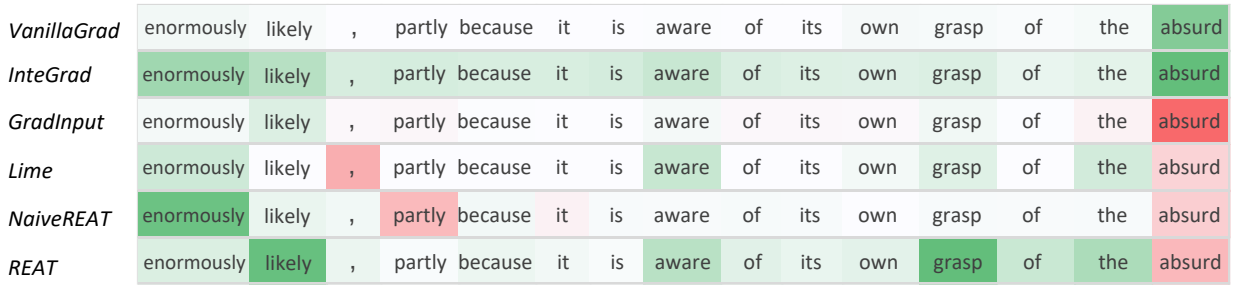


Figure 4: Word-level attribution heatmaps comparing with baseline methods, for a GRU prediction with 99.2% confidence as positive sentiment. Green and red color denote positive and negative contribution of a word to the prediction, respectively.

	SST2			Yelp		
Models	GRU	LSTM	BiGRU	GRU	LSTM	BiGRU
VanillaGrad	41.0	42.3	26.9	57.6	44.4	38.4
InteGrad	44.9	52.6	42.3	58.6	42.4	33.3
GradInput	84.6	80.8	85.9	84.8	82.8	90.9
Lime	73.1	80.8	73.1	74.7	77.8	80.8
NaiveREAT	79.5	87.2	**92.3**	55.6	83.8	86.9
REAT	**85.9**	**89.7**	88.5	**87.9**	**83.8**	**92.9**

Table 5: Interpretability statistical comparison (in percent) of our method with baseline RNN attribution methods.

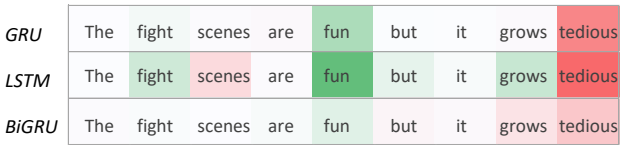


Figure 5: Word-level attribution heatmaps for 3 RNN architectures. Green and red color denote positive and negative contribution of a word to the prediction, respectively.

both positive words[1] and negative words[2] [22]. We obtain a testing set with 78 and 99 samples from SST2 and Yelp respectively. Then we generate a attribution score after feeding each text sample to a RNN, and evaluate the consistency of attribution with human annotations. Here, we focus on analyzing the attribution scores of these words for the positive sentiment side for all testing texts. The interpretation is considered to be a match if the attribution scores for positive words are lager than negative words, otherwise it is treated as a mismatch. Take text "It is ridiculous, of course but it is also refreshing" for example, if the attribution method assigns a higher contribution score to positive word "refreshing" comparing to negative word "ridiculous", we consider it as a match. The final interpretability score is judged by the ratio of matched cases:

$$S_{\text{interpretability}} = \frac{\#\text{match}}{\#\text{match} + \#\text{mismatch}}. \tag{24}$$

We compare the interpretability score of the proposed method with baseline methods on three RNN architectures over SST2 and Yelp dataset. The results are presented in Tab. 5. The proposed method ranks highest for five tasks among all six classification tasks. We observe that the interpretability scores for the NaiveREAT method, which is introduced in Sec. 3.1.1, are unstable across different models and datasets, partly due to the reason that it is not faithful to the original recurrent model. Sometimes, NaiveREAT could accurately capture the contribution score for strong sentiment word, such as for BiGRU prediction on SST2. In other cases, it may assign some false positive and false negative attribution values, leading to lower interpretability scores. Another interesting finding is that the interpretability of InteGrad does not outperform VanillaGrad consistently on all cases. Besides, GradInput yields consistently better performance comparing to VanillaGrad.

4.6 Qualitative Evaluation via Case Studies

We provide several case studies to qualitatively check the effectiveness of the proposed method using heatmaps shown from Fig. 4 to Fig. 6. We use green color to denote positive contribution and red color for negative contribution. Deeper color means higher contribution to the prediction.

4.6.1 Visualization Comparisons with Baseline Methods. We present attribution visualization for a prediction made by GRU with 99.2% confidence for positive sentiment, and compare it with the baseline attribution methods. The results are shown in Fig. 4. The heatmap shows that REAT not only identifies that words "enormously", "likely", "aware", "grasp" have positive contribution for the prediction, but also captures that "absurd" has negative contribution. It is consistent with human comprehension towards this sentence. In contrast, VanillaGrad and InteGrad fail to attribute the negative word; GradInput fails to identify the words that strongly and positively contribute to the prediction, thus can not explain why GRU gives such a high positive prediction score to this text. Also, LIME and NaiveREAT generate some noisy negative scores for irrelevant words, such as "partly", "it" and ",".

4.6.2 Visualizations Under Different RNN Architectures. We compare the word-level attribution results of three RNN architectures. For a given text "The fight scenes are fun but it grows tedious", GRU, LSTM and BiGRU give positive prediction (51.6% confidence), positive prediction (96.2% confidence), and negative prediction (62.7%

[1]https://gist.github.com/mkulakowski2/4289437
[2]https://gist.github.com/mkulakowski2/4289441

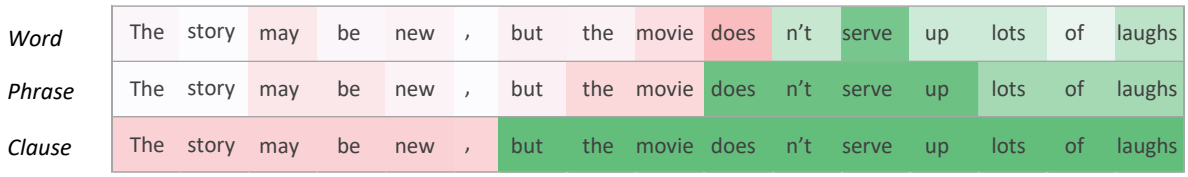


Figure 6: Visualization heatmaps for hierarchical attribution. Green and red color denote positive and negative contribution of a word to the prediction, respectively.

confidence), respectively. We display the attribution heatmaps for positive prediction for all three architectures in Fig. 5. GRU gives nearly the same absolute value of attribution score for "fun" and "tedious". LSTM attributes more words positive contributions than negative words, while BiGRU gives more words negative contribution scores. These attribution heatmaps well reflect the prediction scores, which thus indicates that the interpretations could give users understandable rationale for the predictions.

4.6.3 Visualization for Hierarchical Attribution. We also give the visualization heatmaps of hierarchical attributions in Fig. 6. In this case, we show the attribution scores of a negative sentiment prediction with 99.46% confidence from a LSTM model. For the word-level attribution, "does" has a negative contribution for the prediction, while the combination "does n't serve up" will have a strong positive contribution for the prediction. The clause-level attribution shows that the first clause has a relatively small negative contribution for prediction, while the second clause has a strong positive contribution for prediction. This hierarchical attention thus could represent the contributions at different levels of granularity.

4.7 Linguistic Patterns Analysis

In this section, we apply REAT to analyze the linguistic patterns, in order to comprehend what kind of linguistic knowledge in natural language has been captured by RNNs. We aim to understand whether the linguistic patterns learned are consistent with human cognition. Overall, the analysis of linguistic patterns is implemented by obtaining the contribution score distributions for different syntactic categories. We first tag each word in a text into different part-of-speech (POS) categories. Here, the tagging is performed using the NLTK package[3]. We then proceed by feeding that text to the RNN model and use REAT to generate the contribution score for each word. For each text, the class of interest is its prediction label generated by the RNN model. After that, we summarize the contribution scores for each syntactic category by averaging the scores for all words in that category contained in the dataset.

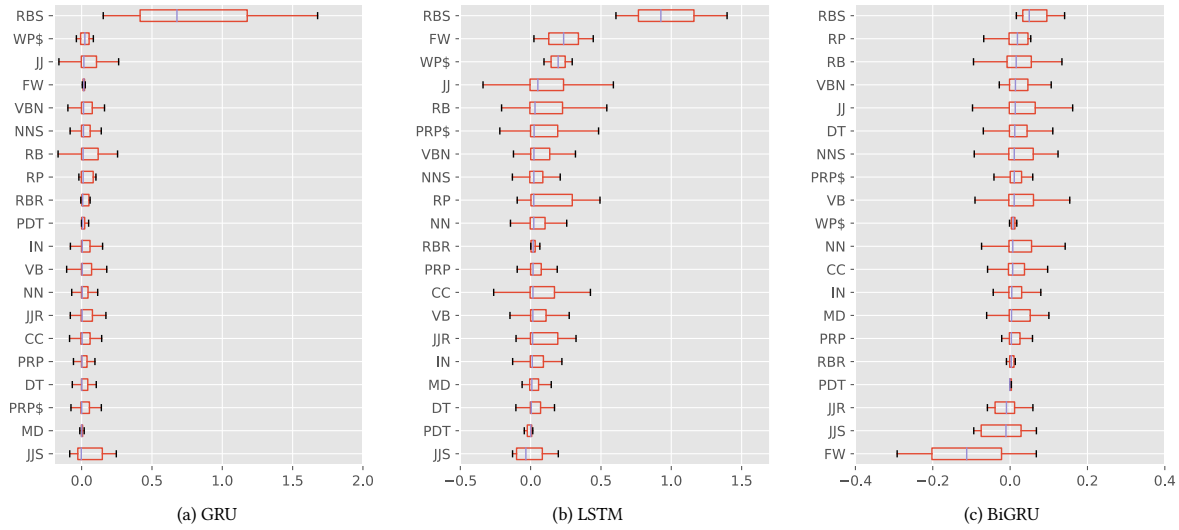
We employ the SST2 test set to analyze linguistic patterns for the three RNN architectures, and use boxplot to illustrate the distribution patterns, as illustrated in Fig. 7. Note that we rank different categories according to the median value. Besides, we combine different verb forms (e.g., past tense and present tense) to a single one, since we observe these verb variants have similar distribution patterns. Also, to capture the primary contribution of each POS category towards the prediction, we omit the outlier points when drawing the boxplot, i.e., those data points that are located outside the fences of the boxplot. An interesting discovery from the boxplots is that, the RBS category, which denotes superlative adverb such as "best" and "most", achieves the highest median value on all three RNN architectures. This is consistent with our human cognition since we indeed rely on these superlative words to express strong emotions. Besides, the adjective category (JJ) ranks relatively high among all the three architectures. The noun category (NN) has a near-zero median for all three architectures. It indicates that these recurrent models are more sensitive to adjectives than nouns, which also makes sense to common cognition habits. In addition, we can observe that the attribution score distribution patterns are similar between GRU and LSTM, while they are very distinct from that of BiGRU. This is partly due to the reason that BiGRU has a bidirectional structure, which will make it use different evidence in making predictions. These findings demonstrate that these RNNs have learned some useful patterns and pay special attention to words that well represent the underlying task.

4.8 Model Misbehavior Analysis

In this section, we apply REAT as a debugging tool to analyze the misbehavior of RNNs. This is motivated by the observation that RNN models may not always meet the expectations of humans, since their performances are still not perfect (see Tab. 3). In this case, the attribution method can be utilized as the tool to analyze their vulnerability and reason the failure cases.

4.8.1 Attribution-based Failure Case Debugging. We use a sample text in the SST2 test set to illustrate this process. The LSTM gives a 99.97% negative sentiment prediction for a text "Schweiger is talented and terribly charismatic, qualities essential to both movie stars and social anarchists". However, this prediction makes no sense to humans, since it is in fact a strong positive sentiment text. Using the proposed attribution method, we find that "terribly" makes the highest contribution for this negative prediction. One possible explanation is that the LSTM model only captures the meaning similar to "terrible", and ignores its other meanings relevant to "extremely", since "terribly" is a polysemant. To validate this hypothesis, we further generate the attribution score distribution for two words "terrible" and "terribly" (see Fig. 8), and find that these two words have consistently negative contribution for positive sentiment. It comes as no surprise, since we use pre-trained word2vec as word embeddings, which are fixed and context-free representations and are limited in performance. Besides, the training set only contains a limited number of apperances of "terribly".

[3]https://www.nltk.org/



(a) GRU (b) LSTM (c) BiGRU

Figure 7: POS category score distributions of three RNN architectures. The x axis denotes attribution scores.

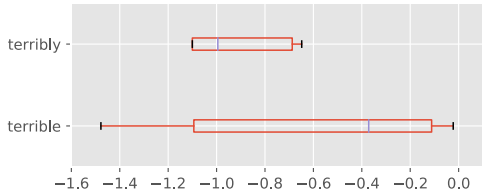


Figure 8: Attribution score distribution for two words.

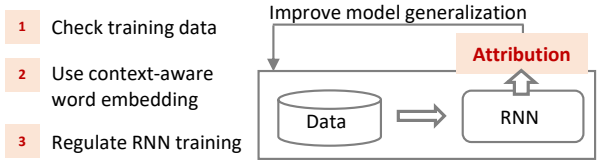


Figure 9: Utilizing attribution to improve models.

This makes the model fail to learn language polysemy and capture the context dependent meaning of "terribly".

4.8.2 Attribution-based Adversarial Attack. Based on the above attribution analysis, we can perform adversarial attacks using interpretable perturbation. Recent work shows that DNNs are fragile, where a small perturbation could dramatically change their prediction results [11, 24, 30]. We change the word "terribly" to its synonyms "extremely" and "very", as shown in Tab. 6. As the result, the prediction for this text shifts from strong negative to strong positive for both conditions, even though the real meaning of the original text is unchanged. Beside this example, such adversarial attack finding applies to other texts in SST2 dataset. We search for texts in test set of SST2, which contain word "extremely" and are originally assigned positive predictions by the LSTM classifier. We change the word "extremely" in the original text to "terribly", as can be seen in Tab. 7. Although the modified texts are semantically identical to our humans, both predictions of these two samples switch from strong positive to strong negative, demonstrating to some extent the vulnerability of this LSTM classifier.

4.8.3 Discussion of Attribution-based Model Generalization Enhancement. The derived attribution method can also be employed to promote the generalization performance of RNN classifiers. Although all the three pretrained RNN classifiers could achieve 100% accuracy on training set of SST2, all of their accuracy on the test set are below 82%. This big gap indicates that the RNN classifiers have overfitted to the training set and may have memorized the artifacts and biases that widely exist in texts. Through examining the failure reasons using the the proposed attribution method, or through attacking the RNN classifiers using attribution-based adversarial samples, we could identify the problem of the RNN classifier. We further propose three directions to promote the generalization ability of RNN classifiers, from the perspective of the training data, word embedding or recurrent model, respectively, as illustrated in Fig. 9. First, if attribution statistic analysis shows that the RNN has learned lots of bias from the training data, then we can check the training data to reduce the data imbalance or data leaking problem to make sure the training data is less biased. Second, if attribution analysis tell us that trained RNN cannot capture context dependent meanings of words, then we can replace context-free and fixed word embeddings, such as word2vec [25] and GloVe [31], to context-aware embeddings, such as the recently proposed ELMo [33], ULMFiT [15] and BERT [8]. Third, if the attribution analysis indicates that the RNN heavily relies on superficial correlations in the training data, then we can add a regularizer to regulate the training behavior of the original RNN model. Bridging attribution with improving model generalization is a challenging topic, which is beyond the length of this paper and will be further investigated in our future research.

	Text	Prediction
Original	Schweiger is talented and terribly charismatic, qualities essential to both movie stars and social anarchists.	Negative (99.97%)
Adversarial	Schweiger is talented and extremely charismatic, qualities essential to both movie stars and social anarchists.	Positive (81.29%)
Adversarial	Schweiger is talented and very charismatic, qualities essential to both movie stars and social anarchists.	Positive (99.53%)

Table 6: Interpretable adversarial attack for a LSTM classifier based on attribution.

	Text	Prediction
Original	Occasionally melodramatic, it 's also extremely effective.	Positive (99.74%)
Adversarial	Occasionally melodramatic, it 's also terribly effective.	Negative (99.00%)
Original	Extremely well acted by the four primary actors, this is a seriously intended movie that is not easily forgotten.	Positive (99.98%)
Adversarial	Terribly well acted by the four primary actors, this is a seriously intended movie that is not easily forgotten.	Negative (87.70%)

Table 7: Interpretable adversarial attack for a LSTM classifier based on attribution.

5 RELATED WORK

Interpretable machine learning is a research topic covering a wide range of directions [3, 9, 17, 23, 43, 45]. In this section, we focus on post-hoc and local interpretation methods which are most relevant to ours and briefly review methods that could provide interpretations for RNN predictions. These attribution methods can be further classified into the following three main categories.

Back-propagation Based Methods This line of work calculates the gradient or its variants of the output with respect to the input, for a particular class of interest, using back-propagation to derive the contribution of words. The philosophy is to identify the words whose variation will most significantly lead to the change of output probability. In the simplest case, the gradient signal is back-propagated, which is first proposed in CNN image classification [38] and later introduced to provide attribution for text classification in NLP. In NLP, gradients are computed with respect to individual entries in word embedding vectors, and then the L2 norm [13] or the dot product of the gradient and the word embedding [7] is calculated to reduce the gradient vector to a scalar, representing the contribution of a single word. Besides gradient signal, some work proposes to back-propagate different signals to the input, such as the relevance of the final prediction score through each layer of the network onto the input layer [1, 2], or only considering the positive gradient signals in the back-propagation process [40].

Perturbation Based Methods The motivation of perturbation based methods is that the most important word for a prediction, once perturbed, will cause the largest probability drop of the output for the target class. We can implement the perturbation in two ways: omission [16, 19] and occlusion [34]. For omission, the word is deleted directly [16]. As for occlusion, the word is replaced with a baseline input. Here, a zero-valued word embedding is utilized [34]. However, both omission and occlusion could make the sentence nonsensical. Since word order is an essential factor of RNN predictions, either omission or occlusion can destroy the word order and thus may trigger the adversarial side of RNN. Therefore, it cannot guarantee that it is meaningful interpretations.

Local Approximation Based Methods These methods are based on the assumption that behaviors of a complex and opaque model around the neighborhood of a given input can be approximated by a simple and interpretable white-box model [35]. A sparse linear model, for example, can be utilized as the locally interpretable model and the weight vector of the linear model is used as the contribution score for a RNN prediction. Sometimes, even the local behavior of a complex model can be extremely non-linear, where linear explanation could lead to poor performance. Therefore, models that are able to capture the non-linear relationship are utilized as the local approximation model. For example, a local attribution method can be designed using if-then rules [36]. Experimental results demonstrate that the generated rules could be utilized to explain both the current instance and some other relevant instances.

The most similar work to ours are two decomposition based methods [27, 28]. However, their work are exclusively designed for LSTM, while our attribution method is widely applicable to different recurrent neural network architectures.

6 CONCLUSION

We propose a new RNN attribution method, called REAT, to provide interpretations for RNN predictions. REAT decomposes the prediction of a RNN as the additive contribution of each words in the input text, in order to faithfully calculate the response of RNN to the input. REAT could also generate phrase-level attribution scores, which can be combined with syntactic parsing algorithms towards attribution at varying granularity. We apply REAT to three standard RNN architectures, including GRU, LSTM and BiGRU. Empirical results on two sentiment analysis datasets validate that interpretations generated by REAT are both interpretable to humans and faithful to the original RNN classifier. We further demonstrate that REAT can reveal the useful linguistic patterns learned by RNNs. In addition, we find that RNN models sometimes capture biases in the training data, which causes them to be vulnerable to adversarial attacks. Leveraging the debugging ability provided by REAT, we are able to identify the problems of RNN, and point out several promising directions to improve RNN's generalization ability. This is a challenging topic and will be investigated in our future research.

ACKNOWLEDGMENTS

The authors thank the anonymous reviewers for their helpful comments. The work is in part supported by NSF grants IIS-1657196, IIS-1718840 and DARPA grant N66001-17-2-4031. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] Leila Arras, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2017. Explaining recurrent neural network predictions in sentiment analysis. *arXiv preprint arXiv:1706.07206* (2017).

[2] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015), e0130140.

[3] Zhengping Che, Sanjay Purushotham, Robinder Khemani, and Yan Liu. 2016. Interpretable deep models for ICU outcome prediction. In *AMIA Annual Symposium Proceedings*. American Medical Informatics Association.

[4] Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. *55th Annual Meeting of the Association for Computational Linguistics (ACL)* (2017).

[5] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[6] Lingyang Chu, Xia Hu, Juhua Hu, Lanjun Wang, and Jian Pei. 2018. Exact and Consistent Interpretation for Piecewise Linear Neural Networks: A Closed Form Solution. *Proceedings of the 24nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)* (2018).

[7] Misha Denil, Alban Demiraj, and Nando de Freitas. 2015. Extraction of salient sentences from labelled documents. *International Conference on Learning Representations (ICLR)* (2015).

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* (2018).

[9] Mengnan Du, Ninghao Liu, and Xia Hu. 2018. Techniques for Interpretable Machine Learning. *arXiv preprint arXiv:1808.00033* (2018).

[10] Mengnan Du, Ninghao Liu, Qingquan Song, and Xia Hu. 2018. Towards Explanation of DNN-based Prediction with Guided Feature Inversion. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)* (2018).

[11] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. *International Conference on Learning Representations (ICLR)* (2015).

[12] Wenbo Guo, Dongliang Mu, Jun Xu, Purui Su, Gang Wang, and Xinyu Xing. 2018. LEMNA: Explaining Deep Learning based Security Applications. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 364–379.

[13] Yotam Hechtlinger. 2016. Interpretation of prediction models using the input gradient. *Conference on Neural Information Processing Systems (NIPS)* (2016).

[14] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[15] Jeremy Howard and Sebastian Ruder. 2018. Fine-tuned Language Models for Text Classification. *56th Annual Meeting of the Association for Computational Linguistics (ACL)* (2018).

[16] Akos Kádár, Grzegorz Chrupała, and Afra Alishahi. 2017. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics* 43, 4 (2017), 761–780.

[17] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. 2016. Visualizing and understanding recurrent networks. *ICLR Workshop* (2016).

[18] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[19] Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220* (2016).

[20] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *International Conference on Learning Representations (ICLR)* (2017).

[21] Zachary C Lipton. 2016. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490* (2016).

[22] Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on World Wide Web (WWW)*. ACM, 342–351.

[23] Ninghao Liu, Mengnan Du, and Xia Hu. 2019. Representation Interpretation with Spatial Encoding and Multimodal Analytics. *ACM International Conference on Web Search and Data Mining (WSDM)* (2019).

[24] Ninghao Liu, Hongxia Yang, and Xia Hu. 2018. Adversarial Detection with Model Interpretation. In *Proceedings of the 24nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM.

[25] Ruslan Mitkov. 2005. *The Oxford handbook of computational linguistics*. Oxford University Press.

[26] Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2017. Adversarial training methods for semi-supervised text classification. *International Conference on Learning Representations (ICLR)* (2017).

[27] W James Murdoch, Peter J Liu, and Bin Yu. 2018. Beyond Word Importance: Contextual Decomposition to Extract Interactions from LSTMs. *International Conference on Learning Representations (ICLR)* (2018).

[28] W James Murdoch and Arthur Szlam. 2017. Automatic rule extraction from long short term memory networks. *International Conference on Learning Representations (ICLR)* (2017).

[29] Dong Nguyen. 2018. Comparing Automatic and Human Evaluation of Local Explanations for Text Classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Vol. 1. 1069–1078.

[30] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *Security and Privacy, 2016 IEEE European Symposium on*. IEEE, 372–387.

[31] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[32] Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. *55th Annual Meeting of the Association for Computational Linguistics (ACL)* (2017).

[33] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)* (2018).

[34] Nina Poerner, Hinrich Schütze, and Benjamin Roth. 2018. Evaluating neural network explanation methods using hybrid documents and morphological prediction. *56th Annual Meeting of the Association for Computational Linguistics (ACL)* (2018).

[35] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 1135–1144.

[36] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-precision model-agnostic explanations. In *AAAI Conference on Artificial Intelligence (AAAI)*.

[37] Motoki Sato, Jun Suzuki, Hiroyuki Shindo, and Yuji Matsumoto. 2018. Interpretable Adversarial Perturbation in Input Embedding Space for Text. *International Joint Conferences on Artificial Intelligence (IJCAI)* (2018).

[38] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. *International Conference on Learning Representations Workshop* (2014).

[39] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing (EMNLP)*. 1631–1642.

[40] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. 2015. Striving for simplicity: The all convolutional net. *International Conference on Learning Representations Workshop* (2015).

[41] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. *International Conference on Machine Learning (ICML)* (2017).

[42] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing (EMNLP)*. 1422–1432.

[43] Fan Yang, Ninghao Liu, Suhang Wang, and Xia Hu. 2018. Towards Interpretation of Recommender Systems with Sorted Explanation Paths. In *IEEE International Conference on Data Mining (ICDM)*.

[44] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

[45] Hao Yuan, Yongjun Chen, Xia Hu, and Shuiwang Ji. 2019. Interpreting Deep Models for Text Analysis via Optimization and Regularization Methods. In *AAAI Conference on Artificial Intelligence (AAAI)*.

[46] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems (NIPS)*. 649–657.

Is a Single Embedding Enough? Learning Node Representations that Capture Multiple Social Contexts

Alessandro Epasto
Google AI
New York, NY
aepasto@google.com

Bryan Perozzi
Google AI
New York, NY
bperozzi@acm.org

ABSTRACT

Recent interest in graph embedding methods has focused on learning a single representation for each node in the graph. But can nodes really be best described by a single vector representation? In this work, we propose a method for learning multiple representations of the nodes in a graph (e.g., the users of a social network). Based on a principled decomposition of the ego-network, each representation encodes the role of the node in a different local community in which the nodes participate. These representations allow for improved reconstruction of the nuanced relationships that occur in the graph – a phenomenon that we illustrate through state-of-the-art results on link prediction tasks on a variety of graphs, reducing the error by up to 90%. In addition, we show that these embeddings allow for effective visual analysis of the learned community structure.

CCS CONCEPTS

• **Information systems → Data mining**; **Social networks**; • **Computing methodologies → Dimensionality reduction and manifold learning**; **Neural networks**; **Learning latent representations**; *Cluster analysis.*

KEYWORDS

graph embeddings; representation learning; polysemous representations

ACM Reference Format:
Alessandro Epasto and Bryan Perozzi. 2019. Is a Single Embedding Enough? Learning Node Representations that Capture Multiple Social Contexts. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313660

1 INTRODUCTION

Learning embedded representations of graphs is a recent and very active area [2, 14, 25, 37, 38, 44]. In a nutshell, an embedding algorithm learns a latent vector representation that maps each vertex v in the graph G to a *single* d dimensional vector. This area has found strong applications, as the embedding representation of nodes leads to improved results in data mining and machine learning tasks, such as node classification [37], user profiling [39], ranking [26], and link prediction [2, 25]. In virtually all cases, the crucial assumption of the embedding methods developed so far is that a *single* embedding vector has to be learned for each node in the graph. Thus, the embedding method can be said to seek to identify the single role or position of each node in the geometry of the graph.

This observation allows us to draw a historical parallel between the very recent research area of graph embedding and the more established field of community detection or graph clustering [22, 24]. Detecting clusters[1] in real world networks is a central topic in computer science, which has an extensive literature. At the beginning of its development, graph clustering has focused mostly on the study of non-overlapping clustering methods [24, 43]. In such methods, each node is assigned to a *single* cluster or community. While the problem of non-overlapping clustering is better understood and has found strong applications and theoretical results, recently, much attention has been devoted to developing *overlapping* clustering methods [16, 20, 40], where each node is allowed to participate in multiple communities. This interest in overlapping comunities is motivated by a number of recent observations of real world networks [1, 16, 19, 29, 30] that show a lack of clear (non-overlapping) community structure.

These findings motivate the following research question: can embedding methods benefit from the awareness of the overlapping clustering structure of real graphs? In particular, can we develop methods where nodes are embedded in multiple vectors, representing their participation in different communities?

In this paper, we provide positive results for these two questions. We develop Splitter, an unsupervised embedding method that allows nodes in a graph to have multiple embeddings to better encode their participation in multiple overlapping communities. Our method is based on recent developments in ego-net analysis, in particular, in overlapping clustering algorithms based on ego-network partitioning [20]. More precisely, we exploit the observation in [16, 19, 20, 40] that cluster structure is easier to identify at the local level. Intuitively, this happens because each node interacts with a given neighbor in usually a single context (even if it is part of many different commmunities in total).

Splitter extends this idea to the case of node embeddings. In particular, we exploit the *persona graph* concept defined by Epasto et al. [20]. This method, given a graph G, creates a new graph G_P (called the persona graph of G), where each node u in G is represented by a series of replicas. These replica nodes, (called the *persona*(s) of u) in G_P, represents an instantiation of the node u in the local community to which it belongs. The method was originally introduced to obtain overlapping clusters. In this paper, we instead show that ego-net based techniques can lead to improvements in

[1]Note that in the paper, we use the terms "cluster" and "community" interchangeably.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313660

embedding methods as well. In particular, we demonstrate that a natural embedding method based on the persona graph outperforms many embedding baselines in the task of link prediction.

To summarize, the contributions of this paper are as follows:

(1) We introduce Splitter, a novel graph embedding method that embeds each node in the graph into multiple embedding vectors, which is based on the analysis of the overlapping structure of communities in real networks.
(2) Our method adds a novel graph regularization constraint to the optimization that enforces consistency of the multiple learned representations for each node.
(3) The method automatically determines the number of embeddings used for each node depending on a principled analysis of the local neighborhood of the node. This does not require the user to specify the number of embeddings as a parameter.
(4) We show experimentally strong improvements over several embedding baselines for the important task of link prediction.
(5) We show how our method enables visual discovery of community membership for nodes that are part of multiple social groups.

2 METHOD

In this section we describe Splitter, our proposed method for learning multiple community-aware representations for a node. First, we start with a review of the preliminaries necessary to understand the work in Section 2.1. Next, in Section 2.2 we discuss our extension for learning multiple node representations. Then we introduce Splitter in Section 2.3, and close with discussing some details of the optimization in Section 2.4.

2.1 Preliminaries

Our method builds upon recent work related to node decomposition [20], and learning node representations with neural networks [35, 37, 38]. Here we describe the basics of both methods.

2.1.1 Notation. We begin with some notation. Let $G = (V, E)$ be an undirected graph[2] consisting of a set V of nodes and a set $E \subset V \times V$ of edges. Let $G[U] = (U, E \cap U \times U)$ be the induced graph of a subset of G's nodes, $U \subset V$. Given a node $u \in V$, we denote its neighborhood as the set of nodes connected to it $N_u = \{v; (u, v) \in E\}$, and its *ego-network* (or ego-net) as the graph induced on the neighborhood $G[N_u]$. We note that the ego-net of u does not include the node u itself in this definition. Finally, let $\mathcal{A}$ be a *non-overlapping clustering* algorithm that given G as an input, returns a partition $\mathcal{A}(G) = (V_1, \ldots, V_t)$ of the vertices V into t disjoint sets (let $\mathrm{np}_{\mathcal{A}}(G) = t$ denote the number of partitions in output.).

2.1.2 Persona Decomposition. The topic of community detection and graph clustering has been of great interest to the community over the last several decades. While much work has focused on finding large clusters, it has been noted that while the community detection problem is hard at scale (the *macroscopic* level), it is relatively trivial when viewed locally (the *microscopic* level) [17, 19]. Using this intuition, a recent proposal from Epasto et al. [20] uses the clusters found in the ego-network of a node (its neighbors and their induced subgraph) as the basis to define a new graph, the *persona graph*, G_P. The nodes in this new graph, which are called *personas*, divide the interactions of each original node in G into several semantic subgroups, which capture different components (or senses) of its network behavior.

More formally, let us assume that we are given a graph G and a clustering algorithm $\mathcal{A}$. The persona decomposition (as proposed in [20]) employs the following algorithm PersonaGraph$(G, \mathcal{A})$ to transform G to its persona graph G_P:

(1) For each node $u \in V$, we use the clustering algorithm $\mathcal{A}$ to partition the ego-net of u. Let $\mathcal{A}(G[N_u]) = \{N_u^1, N_u^2, \ldots, N_u^{t_u}\}$, where $t_u = \mathrm{np}_{\mathcal{A}}(G[N_u])$.
(2) Create a set V' of personas. Each node v_o in V will correspond to t_{v_o} personas (the number of splits of the ego-net of v_o) in V_P, denoted by v_i for $i = 1, \ldots, t_{v_o}$.
(3) Add edges between personas. If $(u, v) \in E$, $v \in N_u^i$ and $u \in N_v^j$, then add an edge (u_i, v_j) to E_P.

After using this procedure, one obtains the persona graph G_P which has some interesting properties. First, every node in G_P is a node from the original graph, split into one or more personas. However, there is no additional connectivity information – the number of edges in G_P is equal to the number of edges in the persona graph. This means that the space required to store G_P is (almost) the same as the original graph. Second, each node in the original graph can be mapped to its corresponding persona(s). However, the community structure of G_P can be wildly different from the original graph. Standard clustering methods, when run on G_P instead genereate overlapping clusterings of G. This phenomena of exposing differing clustering information is visualized further in Section 4.

2.1.3 Graph Embedding. Before introducing our method for learning multiple embeddings for each node, we first review the standard setting of network representation learning, in which a single embedding is learned for each node. The purpose of network embedding is to learn a mapping $\Phi\colon v \in V \mapsto \mathbb{R}^{|V| \times d}$, which encodes the latent structural role of a node in the graph. This, in practice, can be achieved by representing the mapping Φ as a $|V| \times d$ matrix of free parameters that are learned by solving an optimization problem. Perozzi et al. [37] first introduced a modeling of the vertex representation that encodes the node as a function of its co-occurrences with other nodes in short truncated random walks.

More precisely, the method consists of performing multiple random walks over the social graph from each node. The sequences of these walks are then used to extract the co-occurrences of nodes in short sub-windows. These co-occurrences capture the diffusion in the neighborhood around each vertex in the graph, and explore the local community structure around a node. More concretely, the goal of the embedding method is to learn a representation that enables an estimate of the likelihood of a vertex v_i co-occurring with other nodes in the sub-window of a short random walk:

$$\Pr\Big(v_i \mid \big(\Phi(v_1), \Phi(v_2), \cdots, \Phi(v_{i-1})\big)\Big) \tag{1}$$

[2]The method can be also defined for directed graphs in the obvious way; however, we describe it for undirected graphs for simplicity

Notice that the exact computation of this conditional probability is computationally expensive for increasing lengths of the random walks, so DeepWalk uses two techniques to address this challenge. First, the order of the neighboring vertices is ignored. Second, the method reverses the learning task; instead of predicting a missing vertex using the context, it addresses the opposite problem of predicting its local structure using the vertex itself.

These modifications result in the following optimization problem for computing the vertex representations of each node in DeepWalk:

$$\underset{\Phi}{\text{minimize}} \quad -\log \Pr\left(\{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid \Phi(v_i)\right) \tag{2}$$

In the DeepWalk [37] model, the probability of a node v_i co-occurring with v_j is estimated by using a softmax to map the pairwise similarity to a probability space,

$$Pr(v_i|v_j) = \frac{exp(\Phi(v_i) \cdot \Phi'(v_j))}{\sum_{j \in \mathcal{V}} exp(\Phi(v_i) \cdot \Phi'(v_j)} \tag{3}$$

Where $\Phi'(v_i)$ and $\Phi'(v_j)$ represent the "input" and "output" embeddings for node v_i and v_j respectively [34].

2.2 Learning Multiple Node Representations

As discussed so far, network representation learning seeks to learn a function that maps each node to its own representation. Here, we discuss our modifications that were made in light of the fact that we wish to learn one or more representation for each node.

Using the persona decomposition discussed in Section 2.1.2, we can convert the input graph G into the persona graph G_P. From here, it seems like a straightforward application of existing methods to learn one representation for each node $v \in |V_P|$, and as such, learn one or more representation for each original node $u \in |V|$. Unfortunately, this is not the case. The strength of the persona decomposition is also a weakness - it can create a graph that is quite different from that of the original input. In fact, the persona graph can be so different that it may consist of many disconnected components, even if the original graph was connected! Thus, these disconnected components can cause difficulties for representation learning methods. To address these challenges, we propose two improvements to the learning algorithm.

First, we propose adding a constraint, that the persona representations be able to predict their original node in addition to predicting the personas around it in G_P. Specifically, given a persona v_i, we propose to require its representation include a dependency on the node v_o in the original graph G:

$$\Pr\left(v_o \mid \Phi_{G_P}(v_i)\right). \tag{4}$$

To control the strength of our graph regularization, we introduce the parameter λ, which combines with Eq. (2) to yield the following optimization problem:

$$\begin{aligned}\underset{\Phi_{G_P}}{\text{minimize}} \quad & -\log \Pr\left(\{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid \Phi_{G_P}(v_i)\right) \\ & -\lambda \log \Pr\left(v_o \mid \Phi_{G_P}(v_i)\right).\end{aligned} \tag{5}$$

Put another way, this change to the optimization enforces that there are invisible edges to each persona's parent, informing the learning process and regularizing the degree to which the persona representations can deviate. This effectively lets the model reason about the different connected components that may exist after the persona transformation. We note that in practice, we achieved good results on all graphs we considered by simply setting $\lambda = 0.1$.

Secondly, we propose to also make the representation $\Phi_{G_P}(v)$ of a node v's personas depend on its original representation $\Phi_G(v)$ as a prior via initialization. Initializing all personas to the same position in $\mathbb{R}^d$, combined with the regularization term from Eq. (5), constrains the persona embeddings to behave like cohesive parts of a single entity. We note that this does not mean that all of a node's personas end up in the same position at the end of the optimization! Instead, we find that personas with similar roles stay close together, while personas with different roles separate. This is discussed further in Section 4.2, where we examine a visualization of Splitter embeddings. Finally, there is an additional benefit of using the representation of the original graph Φ_G as an initialization – it can help avoid potentially bad random initializations, which can lower task performance [14].

Inference with Multiple Representations. Notice that our method outputs multiple embeddings for each node. To do inference of node features or to predict edges between nodes, one can use standard ML methods to learn a function of the multiple embeddings of each node (or pair of nodes). However, basic aggregations can work too. In our experiments with link prediction, we simply use the maximum dot product over all pairs of embeddings of u and v to predict the likelihood of the pair of nodes being connected.

2.3 Splitter

Using the ideas discussed so far, we present the details of our approach.

2.3.1 Parameters. In addition to an undirected graph $G(V, E)$, the clustering algorithm $\mathcal{A}$ used to obtain the persona graph, as well as the dimensionality of the representations d, our algorithm uses a number of parameters that control the embedding learning process. The first group of parameters deal with sampling G, an essential part of any graph embedding process. Without loss of generality, we describe the parameters to control the sampling process in the notation of Perozzi et. al [37]. Briefly, they are w, the window size to slide over the random walk, t, the length of the random walk to sample from each vertex, and γ the number of walks per vertex. We emphasize that our approach is not limited to simple uniform random walk sampling - any of the more recently proposed graph sampling strategies [25, 35, 38] can be applied in the Splitter model.

The next group of parameters control the optimization. The parameter α controls the learning rate for stochastic gradient descent, and λ effects how strongly the original graph representation regularizes the persona learning.

Finally, an embedding function EmbedFn is used to learn a representation Φ_G of the nodes in the original graph. In order to use the learning algorithm that we specify, this embedding method simply needs to produce representations where the dot-product between vectors encodes the similarity between nodes. The most popular graph embedding methods meet this criteria, including DeepWalk [37], LINE [44], and node2vec [25].

Algorithm 1 SPLITTER. Our method for learning multiple representations of nodes in a graph

Input:
$G(V, E)$, a graph
w, window size
d, embedding size
γ, walks per vertex
t, walk length
α, learning rate
λ, graph regularization coefficient
$\mathcal{A}$, clustering algorithm for the ego-nets
EMBEDFN, a graph embedding method which uses the dot-product similarity (e.g. DeepWalk, node2vec)

Output:
Φ_{G_P} a matrix with one or more representations for each node
$P2N$, a mapping of the rows of Φ_{G_P} to V (the original nodes)

1: **function** SPLITTER(G, EMBEDFN)
2: $G_P \leftarrow$ PERSONAGRAPH$(G, \mathcal{A})$ ▷ Create the persona graph G_P of G using clustering algorithm $\mathcal{A}$ and method in [20].
3: $P2N \leftarrow \emptyset$
4: $\Phi_G \leftarrow$ EMBEDFN(G, w, d, γ, t) ▷ Embed original graph
5: **for each** $v_o \in V$ **do**
6: **for each** $v_j \in$ personas of v_o **do**
7: $\Phi_{G_P}(v_j) \leftarrow \Phi_G(v_o)$ ▷ Initialize j-th persona of v_o
8: $P2N(v_j) \leftarrow v_o$
9: **for** $i = 0$ to γ **do**
10: $O = \text{Shuffle}(V_{G_P})$
11: **for each** $v_i \in O$ **do**
12: $\mathcal{W}_{v_i} = RandomWalk(G_P, v_i, t)$
13: **for each** $v_j \in \mathcal{W}_{v_i}$ **do**
14: **for each** $u_k \in \mathcal{W}_{v_i}[j - w : j + w]$ **do**
15: $J_{G_P}(\Phi_{G_P})_= - \log \Pr(u_k \mid \Phi(v_j))$
16: $J_G(\Phi_{G_P})_= - \log \Pr(P2N(v_j) \mid \Phi(v_j))$
17: $\Phi_{G_P} = \Phi_{G_P} - \alpha * \left(\frac{\partial J_{G_P}}{\partial \Phi_{G_P}} + \lambda \frac{\partial J_G}{\partial \Phi_{G_P}} \right)$
18: **return** Φ_{G_P}, $P2N$

2.3.2 Algorithm. Here, we describe our full algorithm, shown in Algorithm 1 in detail. Lines 2-4 initialize the data structures, create the persona graph, and learn the embedding of the underlying graph. We note that not all of the nodes will necessarily be split; this will depend on each ego-net's structure, as described in Section 2.1.2. Lines 5-8 use the persona graph to initialize the persona representations Φ_{G_P}. The remainder of the algorithm (lines 9-17) details how the persona representations are learned. Line 12 generates the random walks to sample the context of each vertex. WLOG can be graph samples generated in any meaningful way - including uniform random walks [37], skipped random walks [38], or random walks with backtrack [25]. Line 15 calculates the loss due to nodes in the persona graph (how well the persona representation of Φ_{v_j} is able to predict the observed node u_k). Line 16 computes the loss due to the graph regularization (how well the persona representation Φ_{v_j} is able to predict its corresponding original node). Finally, line 18 returns the induced representations Φ_{G_P} and their mapping back to the original nodes $P2N$.

Complexity. The complexity of the algorithm is dominated by two major steps: creating the persona graph and the Skip-gram model learning. Both parts have been analyzed in previous works [14, 20], so we report briefly on the complexity of our algorithm here. Suppose the clustering algorithm $\mathcal{A}$ used on the ego-nets has a complexity of $T(m')$ for analyzing an ego-net of m' edges. Further, suppose the original graph has $\mathcal{T}$ triangles. As such, the persona creation method has a total running time $O(\mathcal{T} + m + \sqrt{m}T(m))$. Moreover, as a worst case, $\mathcal{T} = O(m^{3/2})$ the complexity is $O(m^{3/2} + \sqrt{m}T(m))$. Suppose, for instance, that a linear time clustering algorithm is used; then, the total cost of this phase as a worst case is $O(m^{3/2})$. However, as observed before [20], the algorithm scales much better in practice than this pessimistic bound because the number of triangles is usually much smaller than $m^{3/2}$. The embedding method for a graph of n' total nodes in the persona graph has instead a complexity $O(n'\gamma t w d \log(n')))$, as shown in [14], where d is the number of dimensions, t is the walk length size, w is the window size and γ is the number of random walks used. Notice, that $n' \in O(m)$, as each node u can have at most $O(deg(u))$ persona nodes, so the worst case complexity is: $O(m^{3/2} + \sqrt{m}T(m) + m\gamma t w(d + d\log(m)))$.

2.4 Optimization

As detailed in [37], using representations to predict the probability of nodes (Line 15-16, Algorithm 1) is computationally expensive. To deal with this, we use the hierarchical softmax [34] to calculate these probabilities more efficiently. For completeness's sake, we remark that an alternative optimization strategy exists (using noise contrastive estimation [34]).

Thus, our model parameter set includes both Φ_{G_P} and T, the parameters used internally by the tree in the hierarchical softmax. Further, we use the back-propagation algorithm to estimate the derivatives in lines 15-16, and a stochastic gradient descent (SGD) to optimize the model's parameters. The initial learning rate α for SGD is set to 0.025 at the beginning of the training and then decreased linearly with the number of nodes we have seen so far. Additionally, the parameter λ regularizes the persona embeddings by how much they deviate from the node's original representation.

3 TASK: LINK PREDICTION

In this section, we study how the SPLITTER model proposed so far can be used for the task of link prediction – judging the strength of a potential relationship between the two nodes. We focus on this fundamental application task for the following reasons.

First, we are interested in developing new models that can capture the variation of social behaviors that are expressed in real-world social networks (such as membership in multiple communities). Several recent works suggest that link prediction (or *network reconstruction*) [2, 50] is the best way to analyze an unsupervised network embedding's performance, as it is a primary task (unlike node classification - a secondary task that involves a labeling process that may be uncorrelated with the graph itself). Second, there are several important industrial applications of link prediction on

real networks (e.g. friend suggestion on a social network, product recommendation on an e-commerce site, et cera).

Finally, this task highlights a particular strength of our method. Splitter's ability to model the differing components of a node's social profile (its *personas*) make it especially suitable for the task of link prediction. This addresses a fundamental weakness of most node embedding methods, which effectively treat a node's representation as an average of its multiple senses – a representation that may not make sense in continuous space. Unlike previous work utilizing community information in embeddings [12, 47, 51], we aim to expose the nuanced relationships in a network by sub-dividing the nodes (not the macro-scale community relationships found by joining nodes together).

3.1 Experimental Design

3.1.1 Datasets. We test our Splitter method as well as other baselines on a dataset of five directed and undirected graphs. Our datasets are all publicly available: PPI is introduced [25, 42], while the other datasets are from Stanford SNAP library [28]. For each dataset, in accordance with the standard methodology used in the literature [2, 25], we use the largest weakly connected component of the original graph. We now provide statistics for our dataset.

Directed graphs:

(1) **soc-epinions**: A social network $|V| = 75,877$ and $|E| = 508,836$. Edges represent who trusts whom in the opinion-trust dataset of Epinions.

(2) **wiki-vote**: A voting network $|V| = 7,066$ and $|E| = 103,663$. Nodes are Wikipedia editors. Each directed edges represents a vote for allowing another user becoming an administrator.

Undirected graphs:

(1) **ca-HepTh**: Arxiv's co-author network of High Energy Physics Theory $|V| = 9,877$ and $|E| = 25,998$. Each edge represents co-authorship between two author nodes.

(2) **ca-AstroPh**: Arxiv's co-author network of Astrophysics, $|V| = 17,903$ and $|E| = 197,031$. Each edge represents co-authorship between two author nodes.

(3) **PPI**: Protein-protein interaction graph, $|V| = 3,852$ and $|E| = 20,881$. This represents a natural dataset, where each node is a protein and there is an edge betwen two proteins if they interact (more details in [25]).

3.2 Task

The Link Prediction task follows the methodology introduced in [2, 25], which we briefly detail here. First, the input graph is split into two edge sets, E_{train} and E_{test}, of equal size. The test edges are removed uniformly at random, with the restriction that they do not disconnect the graph. E_{test} is then used as positive examples for a classification task. A corresponding equal sized set of non-existent (random) edges are generated to use as negative examples for testing. The baseline methods are providing the training edges as input, which they use to learn a similarity model (embedded or otherwise). The performance of each method is then measured by ranking the removed edges. Specifically, for each method, we report the ROC-AUC.

3.3 Methods

Here we describe the numerous baseline methods we tested against, including both traditional non-embedding baselines (such as common neighbors) and several embedding baselines. We also detail the application of Splitter's multiple representations of this task.

Non-embedding Baselines: Here, we report standard methods for link prediction that are solely based on the analysis of the adjacency matrix of the graph and in particular, on the immediate neighborhood of the nodes in the graph. These methods take into input E_{train} during inference. Thus, we denote $N(u)$ as the neighbors of u observed in E_{train}. For directed graphs, $N(u)$ only refers to the outgoing edges. In the non-embedding baseline considered, we score an edge (u, v) as a $g(u, v)$, which is a function of $N(u)$ and $N(v)$ only. We consider the following baselines:

(1) **Jaccard Coefficient (J.C.) :**

$$g(u, v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}$$

(2) **Common Neighbors (C.N.):**

$$g(u, v) = |N(u) \cap N(v)|$$

(3) **Adamic Adar (A.A.):**

$$g(u, v) = \sum_{x \in N(u) \cap N(v)} \frac{1}{\log(|N(x)|)}$$

We apply these methods to both the original graph and the persona graph. For the persona graph, we follow a technique similar to our Splitter method to extract a single score from the pairwise similarity (say Jaccard Coefficient) of the multiple persona nodes of a pair of nodes u,v. We also define the Jaccard Coefficient of u, v in the persona graph as the maximum Jaccard Coefficient of a persona node of u and a persona node of v in the persona graph. Similarly, we define the baselines Common Neighbors and Adamic Adar in the persona graph. We report results for using the maximum as aggregation function consistently with our Splitter method, but we experimented as well with many other functions, such as the minimum and the mean and we observed the maximum to perform best.

Embedding Baselines: We also consider the following embedding baselines. These methods take as input E_{train} to learn embedding $\Phi_G(u)$ for every graph node u. During inference, then only the learned embedding are used but *not* the original graph. We compare against these state-of-the-art embedding methods:

(1) Laplacian **EigenMaps** [6] determines the lowest eigenvectors of the graph Laplacian matrix.

(2) **node2vec** [25] learns the embedding by performing random walks on the training edges E_{train} and learning a minimizing skipgram objective (Equation 3).

(3) **DNGR** [11] performs a non-linear (i.e. deep) node embeddings passing a "smoothed" adjancency matrix through a deep auto-encoder. The "smoothing" (called Random Surfing) an alternative to random walks, which effectively has a different context weight from node2vec.

(4) **Asymmetric** [2] is a recent method that learns embeddings by explicitly modeling the edges that appear in the graph.

	Dataset	Non-Embedding Adjacency Methods						Embedding Methods						
		Original graph			Persona graph				Embedding Baselines					Ours
		J.C.	C.N.	A.A.	J.C.	C.N.	A.A.	d	Eigen Maps	node2vec	DNGR	Asymmetric	M-NMF	Splitter
directed	soc-epinions	0.649	0.649	0.647	0.797	0.797	0.797	8	†	0.725	†	0.695	†	**0.972**
								16	†	0.726	†	0.699	†	**0.974**
								32	†	0.714	†	0.700	†	**0.973**
								64	†	0.699	†	0.698	†	**0.970**
								128	†	0.691	†	0.718	†	**0.967**
	wiki-vote	0.579	0.580	0.562	0.860	0.865	0.866	8	0.613	0.643	0.630	0.608	0.886	**0.950**
								16	0.607	0.642	0.622	0.643	0.912	**0.952**
								32	0.600	0.641	0.619	0.683	0.926	**0.953**
								64	0.613	0.642	0.598	0.702	0.932	**0.952**
								128	0.622	0.643	0.554	0.730	0.934	**0.939**
undirected	ca-HepTh	0.765	0.765	0.765	0.553	0.553	0.553	8	0.786	0.731	0.706	0.605	0.852	**0.877**
								16	0.790	0.787	0.780	0.885	0.884	**0.897**
								32	0.795	0.858	0.829	0.884	0.903	**0.909**
								64	0.802	0.886	0.868	0.870	0.912	**0.917**
								128	0.812	0.901	0.897	0.820	0.908	**0.920**
	ca-AstroPh	0.942	0.942	0.944	0.874	0.874	0.874	8	0.825	0.811	0.852	0.592	0.903	**0.959**
								16	0.825	0.833	0.877	0.657	0.935	**0.972**
								32	0.825	0.899	0.917	0.942	0.954	**0.978**
								64	0.824	0.934	0.939	0.936	0.966	**0.982**
								128	0.829	0.955	0.968	0.939	0.974	**0.985**
	PPI	0.766	0.776	0.779	0.698	0.701	0.702	8	0.710	0.733	0.583	0.550	0.739	**0.865**
								16	0.711	0.707	0.687	0.786	0.776	**0.869**
								32	0.709	0.691	0.741	0.794	0.793	**0.869**
								64	0.707	0.671	0.767	0.813	0.817	**0.866**
								128	0.737	0.698	0.769	0.799	0.840	**0.863**

Table 1: We report the ROC-AUC for a link prediction task performed using an ablation test. The columns J.C., C.N. and A.A. stand for the baselines jaccard coefficient, common neighbors and adamic-adar, respectively. The rows represents the datasets (directed and undirected) while the columns represent the methods compared. We compare our Splitter method with three non-embeddings methods (applied to both the original and persona graph) as well four embeddings baselines. For the embedding methods we report results using dimensionality $\{8, 16, 32, 64, 128\}$. We report in bold the highest AUC-ROC for each dimension and dataset. Notice that for Splitter the dimension refers to the size of the embedding of each persona node (i.e. the total embedding size of each node is larger). The next table reports results at parity of space. Results with † indicate lack of completion. We used a machine with 32 GB ram.

Dataset	Avg. Personas per Node $\bar{p}$
soc-epinions	3.03
wiki-vote	4.00
ca-HepTh	2.39
ca-AstroPh	2.53
ppi	4.97

Table 2: Average number of persona nodes per node in original graph.

We compare against the most similar model proposed in the work, the shallow asymmetric model.

(5) **M-NMF** [47] uses a modularity based community detection model to jointly optimize the embedding and community assignment of each node. Unlike Splitter, this method assigns each node to one community, and is based on joining nodes together (not splitting them apart).

We run each of these methods with their suggested default parameters.[3]

During inference with the baselines, we use the embedding of a pair of node u and v to rank the likelihood of the link u, v formed by employing a scoring function that takes in the input the embeddings of the two nodes. To do so, for consistency with previous work, we used the same methodology of [2], which we summarize here. Let Y_u and Y_v be, respectively, the embeddings of u and v. The edge scoring function is defined as follows: for **EigenMaps**,

[3]In order to advance the field, and ensure the reproducibility of our method, we are releasing an implementation of Splitter at https://github.com/google-research/google-research/tree/master/graph_embedding/persona.

it is $-||Y_u - Y_v||$; for **node2vec**, we use the off-shelve binary classification LogisticRegression algorithm of sklearn to lean a model over the Hadamard product of the embeddings of the two nodes; for **DNGR**, we use the bottleneck layer values as the embeddings and the dot product as similarity; for **Asymmetric**, we use the dot product; and for **M-NMF**, similarly to node2vec, we train a model on the Hadamard product of the embeddings.

Our Method (Splitter): In order to use Splitter for link prediction, we need a method to calculate a single similarity score between two nodes (u,v) in the original graph G, each of which may have multiple persona representations. Specifically, as the Splitter embedding model uses the dot-product to encode the similarity between the two node's representations, we need a method to extract a single score from the pairwise similarity of the (potentially) multiple persona nodes. Similar to applying non-embedding baselines to the persona graph, we experimented with a number of aggregation functions (including *min*, *max*, *mean*, etc). The highest performing aggregation function was the maximum, so we define the similarity between the two nodes in G to be the maximum dot-product between any of their constituent personas in G_P.

For learning embeddings with Splitter, we set the random walk length $t = 40$, number of walks per node $\gamma = 10$, and the window size $w = 5$, the initial learning rate $\alpha = 0.025$, and the graph regularization coefficient $\lambda = 0.1$. For EmbedFn, we used node2vec with random walk parameters ($p = q = 1$) which is equivalent to DeepWalk.

3.4 Experimental Results

In the following table, we report the experimental comparison of Splitter with several embedding and non-embedding baselines. For each experiment, we report the AUC-ROC in a link prediction task performed using an ablation test described in Section 3.2. For consistency of comparison, we use the experimental settings (datasets and training/testing splits) of [2] for the baselines. Hence, the baselinesâĂŹ numbers are the same of [2] and are reported for completeness.

We first report in Table 1 the results of Splitter with several dimensionality settings. All results involving Splitter or the adjacency matrix on persona graph baseline uses the connected component method for ego-net clustering. We chose the connected component algorithm for easy replication of the results because it performs very well, as well as due to its previous use in ego-net clustering [20, 40]. In particular [20], showed theoretical results in a graph model for the connected component method at ego-net level.

We first take a look at the adjacency matrix baselines. Here, we consider both the vanilla version of the well-known baselines in the original graph, as well as the application of such baselines on the persona pre-processed graph (with a methodology similar to Splitter, as described above). We observe that simply applying the persona preprocessing to the standard baselines does not consistently improve the results over using them in the original graph. In particular, we only observe improvements in two of the five graphs we analyzed, while sometimes, we see even strong losses in applying this simple pre-processing, especially for our sparsest graphs, such as ca-HepTh. This confirms that the gains observed in our Splitter do not come merely from the pre-possessing.

Now, we consider the embedding methods. In this table, to gain an understanding of Splitter embeddings, we compare different sizes of Splitter embeddings (per persona node) with same size embeddings of other methods (per node). Before delving into the results, a note is important; since each node can have multiple persona embeddings, the total embedding size of Splitter (for the same dimension) can be larger than that of another standard embedding method. For this reason, we will later compare the results at same total embedding size. First, we observe in Table 1 that at the same level of dimensionality, Splitter always outperforms all other baselines. The improvement is particularly significant in the largest graph *epinions* where our method using size 8 embeddings improves AUC-ROC by a factor of 40% (reduction in error of 90%) even when compared with the best baseline with 128 dimensions. Similarly, our method achieves close to optimal performances in two of the other largest graphs, wiki-vote and ca-AstroPh.

As we have mentioned before, our method embeds each node into multiple embeddings (one for each persona node), so for a given dimension d of the embedding, the average embedding size per node is given by $d\bar{p}$, where $\bar{p}$ is the average number of personas per nodes (i.e. the average number of ego-net clusters per node) in the graph. We report the average number of persona nodes for all the graphs in our datasets in Table 2. As we notice, the average number of persona nodes is between 2 and 5 in our datasets (using the connected component ego-network splitting algorithm). We report in Table 3 a different look at the previous results. This time, we compare Splitter's AUC-ROC to other embeddings, allowing for the same (or higher) total embedding space. In our example in Table 3, we fix $d = 16$ for the Splitter method and then we compute the effective average embedding size for each dataset ($\bar{p}16$). Thereafter, we compare our results with the best result for each baseline that uses approximately $16\bar{p}$ dimensions (for fairness of comparison, we actually round $16\bar{p}$ to the next power of 2 and always allow more space for the other methods vs our method). Thus, it is possible to observe that the AUC-ROC of Splitter is higher than that of every other baseline using about $16\bar{p}$ dimensions for all datasets, except once (in ca-HepTh, M-NMF is better, and we observe this is the sparsest graph). This confirms that the method improves over the baselines even after accounting for the increased space due to the presence of multiple embeddings for nodes. We observe the same results for other d besides $d = 16$.

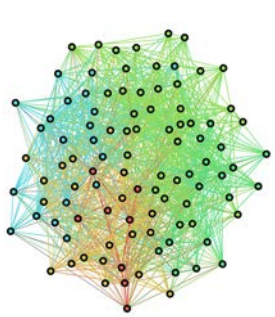
4 TASK: VISUALIZATION

4.1 Synthetic graphs

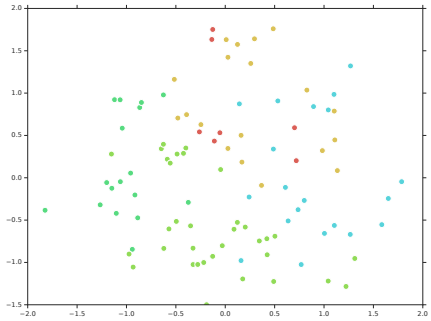
To gain insight on how our embedding method framework operates, we first provide a visualization of a small synthetic graph. The methodology we use is similar to that of [20], which we report for completeness. We created a random graph with planted overlapping communities using the Lanchinetti et al [27] model. We chose this model for consistency with previous ego-net based works [16, 20] and because the model replicates several properties of real-world graphs, such as power law distribution of degrees, varying community sizes and the membership of nodes in varying community numbers. The graph contains 100 nodes and has 9 highly overlapping ground-truth communities.

Dataset	$d_{max} = 16\bar{p}$	Best EigenMaps	Best Node2Vec	Best DNGR	Best Asymmetric	Best M-NMF	Splitter $d = 16$
soc-epinions	48.5	†	0.726	†	0.700	†	**0.974**
wiki-vote	64.0	0.613	0.643	0.630	0.702	0.932	**0.952**
ca-HepTh	38.2	0.802	0.886	0.868	0.885	**0.912**	0.897
ca-AstroPh	40.5	0.824	0.934	0.939	0.942	0.966	**0.972**
ppi	79.5	0.737	0.733	0.769	0.813	0.840	**0.869**

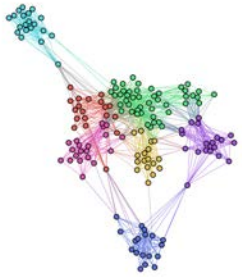
Table 3: AUC-ROC of Splitter with $d = 16$ compared with best baseline allowed larger total embedding space (at approximate space parity).



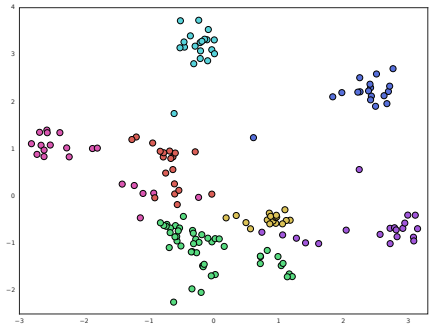
(a) Original Graph



(b) Original Graph Embedding – using M-NMF



(c) Persona Graph



(d) Persona Graph Embedding – using Splitter

Figure 1: Original graph and Persona graph with corresponding embeddings. Notice how the persona graph community structure is clearer than the one in the original graph and this corresponds to more separated embeddings. The colors in the original and persona graph corresponds to community found by a modularity based algorithm. Left side pictures are used with permission from Epasto et al. [20]. (best viewed in color)

We show the results of the visualization in Figure 1. First in Figure 1a and Figure 1c we show a force-directed layout of the original graph and the corresponding persona graph (using Gephi [5] with the same visualization settings). The node coloring corresponds to the discovered communities using a non-overlapping community detection that optimizes modularity [7]. As observed by Epasto et al. [20] on this dataset, the persona graph has a much clearer community structure, finding 8 of the 9 overlapping communities (only 5 communties around found in the original graph).

We now turn our attention to the embeddings output by our method. In Figure 1b, we show a 2D embedding obtained using M-NMF [47] with default settings on the original graph, while in Figure 1d, we show a 2D embedding obtained by our method. As such, it is possible to appreciate how the Splitter embeddings more clearly identify the community structure, as the eight communities found are better separated. In contrast, the M-NMF embeddings do not show a clear separation for this graph, which has highly overlapping communities.

4.2 DBLP Co-Authorship Graph

We then turn our attention to a real-world co-authorship graph using DBLP data. The `4area` graph contains co-authorship relationships extracted from papers in 4 areas of study: Data Mining, Machine Learning, Databases, and Information Retrieval [36]. It is possible to see in Figure 2a and Figure 2b, respectively, a plot of node2vec embeddings and of Splitter embeddings. Notice how in Figure 2b there are 4 areas more or less separated that upon inspection, corresponds to the 4 different fields. A similar observation is



(a) Node2vec – Highlighted Author

(b) Splitter Embeddings – Highlighted Author

Figure 2: Comparison of embedding visualization of a node2vec and Splitter in a DBLP co-authorship graph containing authors from 4 main areas: Data Mining, Machine Learning, Data Base and Information Retrieval. It is possible to observe how the 4 areas correspond to (more or less) well separated regions in the space in both embedding methods. However, standard embeddings force each author that participates in multiple communities to be represented in a single point, while the persona methods allow author-nodes to be represented by multiple embeddings. We show one such embedding for a prolific author with contributions in both the Data Mining and Machine Learning community. Notice how the node2vec embedding results in the author being embedding in the data mining region only (this is best viewed in color).

possible for Figure 2a, representing node2vec embeddings. The key observation in this application scenario is that many authors (in particular the most prolific ones) can contribute to more than one area of study or more than one subarea. However, standard embedding methods force each node to be embedded in only one point in the space, while the Splitter method allows to represent a node as a combination of embeddings (in this setting, we obtain 1.58 personas per node on average). In the Figures 2a and 2b, respectively, we highlight the embeddings learned for one such prolific author, `Jure Leskovec`. Note how this author has one single embedding obtained by node2vec, and multiple embeddings given in output by Splitter. Upon inspection, we observe that the author is embedded in a data mining region by node2vec, surrounded by other prominent authors in Data Mining, such as `Christos Faloutsos`.

However, when observing the representations learned through our Splitter method, we see that this author has a number of persona representations. Moreover, many of the personas reflect different sub-groups of coauthors in Data Mining that our node (e.g. one persona corresponds to co-authorship with other students while at CMU). These personas encode significant portions of the 'average' or 'global' position, which is captured by node2vec. Nevertheless, we also see that a persona ((Jure Leskovec|4)) is now present in the Machine Learning cluster. This illustrates how the representations from Splitter allows the node to span both the Data Mining and Machine Learning region of the space, better characterizing the contributions of the node. Similar observations hold for other authors.

5 RELATED WORK

Our work bridges two very active areas of research: ego-net analysis and graph embeddings. As these are vast and fast growing, we will restrict ourselves to reviewing only the most closely related papers in these two areas.

5.1 Graph embedding

These methods learn one embedding per graph node, with an objective that maximizes (minimizes) the product (distance) of node embeddings if they are 'close' in the input graph. These are most related to our work. In fact, our work builds on the approach introduced by DeepWalk [37], which learns node embeddings using simulated random walks. This idea has been extended to consider node embeddings learned on different variations of random walks [25], as well as other graph similarity measures [45], other loss functions [8], or additional information such as edge labels [15]. These node embeddings have been used as features for various tasks on networks, such as node classification [37], user profiling [39], and link prediction [2, 25, 50]. More recent work in the area has examined preserving the structural roles of nodes in a network [41, 46], learning embeddings for a graph kernel [4], or proposing attention methods to automatically learn the model's hyperparameters [3]. For more information on node embedding, we direct the reader to a recent survey [13].

Moreover, most node embedding methods focus on only learning one representation for each node in the graph. Walklets [38] decompose the hierarchy of relationships exposed in a random walk into a family of related embeddings. However, this is distinctly different from our work, as each node is represented exactly once at each

level of the hierarchy. In addition, the representations are learned independently from each other. HARP [14] is a meta-approach for finding good initializations for embedding algorithms. As a by-product, it produces a series of representations that encode a hierarchical clustering. A number of other works focus on learning community representations, or using communities to inform the node embedding process [12, 47, 51]. Unlike these works, which focus on aggregating nodes into *less* representations, we focus on *dividing* nodes into more representations. This allows our approach, Splitter, to more easily represent prolific nodes that may have overlapping community membership.

5.2 Ego-net analysis

Our work is most closely related to the line of research in social network analysis based on ego-net clustering. From their introduction by Freeman [23] in 1982, ego-nets or ego-networks are a mainstay of social-network analysis [9, 18, 21, 48]. Rees and Gallagher [40] jump-started a rich stream of works [16, 19, 20] that exploit ego-network level clusters to extract important structural information on communities [22], to which a node belongs. They proposed to partition nodes' ego-net-minus-ego graphs in their connected components to find a global overlapping clustering of the graph. Coscia et al. [16] improved over their clustering method by proposing to use a more sophisticated label propagation ego-net partitioning technique. Several authors have since built on such a line of work to improve the scalability and accuracy of ego-net based clustering [10, 19, 20, 33, 33], while others have designed ego-net analysis methods that tackle user metadata on top of the ego-net connectivity [31, 32, 49].

Mostly related to our work is the recent paper by Epasto el al. [20], where they introduce the persona graph method for overlapping clustering. They present a scalable overlapping clustering algorithm based on a local ego-net partition. Their algorithm first creates the ego-nets of all nodes and partition them (in parallel) using any non-overlapping algorithm. These ego-net level partitions are then used to create the persona graph, which is described in more detail in this paper, as this is the basis of our embedding methods. Then, the persona graph is partitioned with another parallel clustering algorithm to obtain overlapping clusters in the original graph.

6 CONCLUSIONS

We introduced Splitter, a novel graph embedding method that builds on recent advances in ego-network analysis and overlapping clustering. In particular, we exploited the recently introduced persona graph decomposition to develop an embedding algorithm that represents nodes in the graph with multiple vectors in a principled way. Our experimental analysis shows strong improvements for the tasks of link prediction and visual discovery and exploration of the community membership of nodes.

Our method draws a connection between the rich and well-studied field of overlapping community detection and the more recent one of graph embedding which we believe may result in further research results. As future work we want to explore more in this direction, focusing on the following challenges: (1) exploiting embeddings for overlapping clustering; (2) studying the effect of this method on web-scale datasets; (3) developing theoretical results on this method; (4) applying our embeddings for classification and semi-supervised learning tasks; and (5) developing the study of learning polysemous node embeddings.

REFERENCES

[1] Bruno Abrahao, Sucheta Soundarajan, John Hopcroft, and Robert Kleinberg. 2014. A separability framework for analyzing community structure. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 8, 1 (2014), 5.

[2] Sami Abu-El-Haija, Bryan Perozzi, and Rami Al-Rfou. 2017. Learning Edge Representations via Low-Rank Asymmetric Projections. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. ACM, New York, NY, USA, 1787–1796.

[3] Sami Abu-El-Haija, Bryan Perozzi, Rami Al-Rfou, and Alexander A. Alemi. 2018. Watch Your Step: Learning Node Embeddings via Graph Attention. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. 9198–9208. http://papers.nips.cc/paper/8131-watch-your-step-learning-node-embeddings-via-graph-attention

[4] Rami Al-Rfou, Dustin Zelle, and Bryan Perozzi. 2019. DDGK: Learning Graph Representations for Deep Divergence Graph Kernels. In *Proceedings of the 2019 World Wide Web Conference on World Wide Web, WWW 2019*.

[5] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. 2009. Gephi: an open source software for exploring and manipulating networks. (2009).

[6] Mikhail Belkin and Partha Niyogi. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*. 585–591.

[7] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[8] Aleksandar Bojchevski and Stephan Günnemann. 2018. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *ICLR* (2018).

[9] R. Burt. 1995. *Structural Holes: The Social Structure of Competition*. Harvard Press.

[10] Nazar Buzun, Anton Korshunov, Valeriy Avanesov, Ilya Filonenko, Ilya Kozlov, Denis Turdakov, and Hangkyu Kim. 2014. EgoLP: Fast and Distributed Community Detection in Billion-Node Social Networks. In *2014 IEEE ICDM Workshops*. 533–540.

[11] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep neural networks for learning graph representations. In *Thirtieth AAAI Conference on Artificial Intelligence*.

[12] Sandro Cavallari, Vincent W Zheng, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. 2017. Learning community embedding with community detection and node embedding on graphs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 377–386.

[13] Haochen Chen, Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2018. A tutorial on network embeddings. *arXiv preprint arXiv:1808.02590* (2018).

[14] Haochen Chen, Bryan Perozzi, Yifan Hu, and Steven Skiena. 2018. Harp: Hierarchical representation learning for networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[15] Haochen Chen, Xiaofei Sun, Yingtao Tian, Bryan Perozzi, Muhao Chen, and Steven Skiena. 2018. Enhanced Network Embeddings via Exploiting Edge Labels. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. 1579–1582.

[16] Michele Coscia, Giulio Rossetti, Fosca Giannotti, and Dino Pedreschi. 2014. Uncovering Hierarchical and Overlapping Communities with a Local-First Approach. *TKDD* (2014).

[17] Michele Coscia, Giulio Rossetti, Fosca Giannotti, and Dino Pedreschi. 2014. Uncovering hierarchical and overlapping communities with a local-first approach. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 9, 1 (2014), 6.

[18] R. I. M. Dunbar and S. G. B. Roberts. 2010. Communication in Social Networks: Effects of Kinship, Network Size and Emotional Closeness. *Personal Relationships* (2010).

[19] Alessandro Epasto, Silvio Lattanzi, Vahab Mirrokni, Ismail Oner Sebe, Ahmed Taei, and Sunita Verma. 2015. Ego-net community mining applied to friend suggestion. *VLDB* 9, 4 (2015), 324–335.

[20] Alessandro Epasto, Silvio Lattanzi, and Renato Paes Leme. 2017. Ego-Splitting Framework: from Non-Overlapping to Overlapping Clusters. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 145–154.

[21] M. Everett and S. P. Borgatti. 2005. Ego network betweenness. *Social Networks* (2005).

[22] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* (2010).

[23] L. C. T. Freeman. 1982. Centered graphs and the structure of ego networks. *Mathematical Social Sciences* (1982).

[24] Michelle Girvan and Mark EJ Newman. 2002. Community structure in social and biological networks. *Proceedings of the national academy of sciences* 99, 12 (2002), 7821–7826.

[25] A. Grover and J. Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[26] Chin-Chi Hsu, Yi-An Lai, Wen-Hao Chen, Ming-Han Feng, and Shou-De Lin. 2017. Unsupervised Ranking using Graph Structures and Node Attributes. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 771–779.

[27] Andrea Lancichinetti and Santo Fortunato. 2009. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E* 80, 1 (2009), 016118.

[28] J. Leskovec and A. Krevl. 2014. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data

[29] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. 2009. Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. *Internet Mathematics* (2009).

[30] Jure Leskovec, Kevin J Lang, and Michael Mahoney. 2010. Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th international conference on World wide web*. ACM, 631–640.

[31] Jure Leskovec and Julian J Mcauley. 2012. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*. 539–547.

[32] Rui Li, Chi Wang, and Kevin Chen-Chuan Chang. 2014. User profiling in an ego network: co-profiling attributes and relationships. In *Proceedings of the 23rd international conference on World wide web*. ACM, 819–830.

[33] Panagiotis Liakos, Alexandros Ntoulas, and Alex Delis. 2016. Scalable link community detection: A local dispersion-aware approach. In *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, 716–725.

[34] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[35] Bryan Perozzi. 2016. *Local Modeling of Attributed Graphs: Algorithms and Applications*. Ph.D. Dissertation. State University of New York at Stony Brook.

[36] Bryan Perozzi, Leman Akoglu, Patricia Iglesias Sánchez, and Emmanuel Müller. 2014. Focused clustering and outlier detection in large attributed graphs. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1346–1355.

[37] B. Perozzi, R. Al-Rfou, and S. Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Knowledge Discovery and Data Mining*.

[38] B. Perozzi, V. Kulkarni, H. Chen, and S. Skiena. 2017. Don't Walk, Skip! Online Learning of Multi-scale Network Embeddings. In *2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*.

[39] Bryan Perozzi and Steven Skiena. 2015. Exact age prediction in social networks. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 91–92.

[40] Bradley S Rees and Keith B Gallagher. 2010. Overlapping community detection by collective friendship group inference. In *2010 International Conference on Advances in Social Networks Analysis and Mining*. IEEE, 375–379.

[41] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 385–394.

[42] C. Stark, B.J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers. 2006. BioGRID: A General Repository for Interaction Datasets. In *Nucleic Acids Research*. https://www.ncbi.nlm.nih.gov/pubmed/16381927

[43] Peter R Suaris and Gershon Kedem. 1988. An algorithm for quadrisection and its application to standard cell placement. *IEEE Transactions on Circuits and Systems* 35, 3 (1988), 294–303.

[44] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1067–1077.

[45] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2018. Verse: Versatile graph embeddings from similarity measures. (2018), 539–548.

[46] Ke Tu, Peng Cui, Xiao Wang, Philip S. Yu, and Wenwu Zhu. 2018. Deep Recursive Network Embedding with Regular Equivalence. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '18)*. ACM, New York, NY, USA, 2357–2366.

[47] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community preserving network embedding. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[48] S. Wasserman and K. Faust. 1994. *Social network analysis: methods and applications*. Cambridge University Press.

[49] Jaewon Yang, Julian J. McAuley, and Jure Leskovec. 2014. Detecting cohesive and 2-mode communities indirected and undirected networks. In *WSDM*.

[50] Ziwei Zhang, Peng Cui, Xiao Wang, Jian Pei, Xuanrong Yao, and Wenwu Zhu. 2018. Arbitrary-Order Proximity Preserved Network Embedding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '18)*. ACM, New York, NY, USA, 2778–2786.

[51] Vincent W Zheng, Sandro Cavallari, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. 2016. From node embedding to community embedding. *arXiv preprint arXiv:1610.09950* (2016).

On-Device Algorithms for Public-Private Data with Absolute Privacy

Alessandro Epasto
Google
New York
aepasto@google.com

Hossein Esfandiari
Google
New York
esfandiari@google.com

Vahab Mirrokni
Google
New York
mirrokni@google.com

ABSTRACT

Motivated by the increasing need to preserve privacy in digital devices, we introduce the *on-device public-private* model of computation. Our motivation comes from social-network based recommender systems in which the users want to receive recommendations based on the information available on their devices, as well as the suggestions of their social contacts, without sharing such information or contacts with the central recommendation system. Our model allows us to solve many algorithmic problems while providing absolute (deterministic) guarantees of the privacy of on-device data and the user's contacts. In fact, we ensure that the private data and private contacts are never revealed to the central system. Our restrictive model of computation presents several interesting algorithmic challenges because any computation based on private information and contacts must be performed on local devices of limited capabilities. Despite these challenges, under realistic assumptions of inter-device communication, we show several efficient algorithms for fundamental data mining and machine learning problems, ranging from k-means clustering to heavy hitters. We complement this analysis with strong impossibility results for efficient private algorithms without allowing inter-device communication. In our experimental evaluation, we show that our private algorithms provide results almost as accurate as those of the non-private ones while speeding up the on-device computations by orders of magnitude.

CCS CONCEPTS

• **Security and privacy** → **Social network security and privacy**; • **Information systems** → **Data mining**; **Social networks**; • **Theory of computation** → *Sketching and sampling*.

KEYWORDS

On-Device Computation; Public-Private Data; Privacy; Social Networks; Graph Algorithms

ACM Reference Format:
Alessandro Epasto, Hossein Esfandiari, and Vahab Mirrokni. 2019. On-Device Algorithms for Public-Private Data with Absolute Privacy. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313677

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313677

1 INTRODUCTION

Digital devices play a significant role in the lives of their users through the connections they provide, the information they assist in consuming, and the online purchases they enable. The wealth of information on such devices (contacts, purchases, etc.) can be potentially used to provide their owner with ever-improving services and a better user experience, but *only* if the user's privacy is respected. Providing the users with useful services while also respecting their privacy is the holy grail of data mining and machine learning and a fundamental endeavor in the research community and in the industry [1, 15, 21–23, 27, 48].

Consider the case of a standard recommender system, such as a service that suggests books to read. The system has a central recommendation provider. The user logs into the system and volunteers her ratings for the books already read and, in exchange, receives suggestions for new books to read. Now, assume that the user would like to receive suggestions based on the books read by her close friends (perhaps the contacts in her mobile phone). In a standard centralized system [12], to obtain such social recommendations, the user needs to authorize access to her list of contacts on the phone.

This has obvious limitations. The user might want to share the book reviews online and with her friends, but at the same time she might not want to share the contact list to the system. The standard centralized system architecture does not allow such a use case, because all the data necessary to solve the task of obtaining the reviews (i.e., the book ratings and the social graph) must be stored and processed in a centralized system.

In this paper, we begin the study of a new model of computation that we call the *On-Device Public-Private* model. This model allows the user to employ all the information available on her device, as well as the user's social contacts, without sharing them with the central system. In particular, we show how to solve several machine learning and data mining problems, in this model, without having to move any private data to the central system and without having to reveal the private connections in the social graph. Our method provides absolute deterministic guarantees of privacy, as no information based on the private data or the private contacts of the user is shared with the central authority.

This model is along the lines of other privacy-preserving schemes that are actively studied and that are reviewed in related workâĂŤfor instance, differential privacy [1, 22–24], federated learning [38], and the recent public-private data models [7, 15]. All these methods allow strong privacy guarantees, and may complement ours, but we observe that they do not directly model the social-graph-based information or they involve a setting where private data or some information based on it may still need to be collected by the central

authority (although in an anonymized or aggregated form). We seek to study a model that allows general algorithmic problems to be solved while harnessing the social network of the users. Our model can be seen as an extension to the recent public-private graph model [15], along the lines of on-device learning settings in which our model, contrary to the public-private graph model, does not require any private information to be shared with the central system, and it also allows metadata on nodes of the network to address applications beyond graph analysis. Finally, we show how to achieve this without relying on insertion of noise to obfuscate the output, which might partially degrade it, while achieving, by design, an even stronger notion of privacy: private data is only used to provide outputs to the data owner.

Our restrictive model of computation presents several interesting algorithmic challenges: since no private information is shared, all computations based on private data must be performed on local devices of limited capabilities. Despite these challenges, we show efficient algorithms in our model for fundamental data mining and machine learning problems, ranging from clustering to recommendation system applications.

In a nutshell, our model works as follows: we allow the user to specify an arbitrary partitioning of the local device data as public or private. Moreover, the user is allowed to specify some contacts as private or public. Only the public data and the public contacts are shared to the central system in accordance with the choice of the user. Then, the user wants to obtain some information (e.g., the results of a recommendation system algorithm) on a dataset consisting of all the data to which the user has access (i.e., her private data, all her contacts whether private or public, as well as all the public information available in the system). All of this must be achieved without revealing the private data and the private contacts to the central authority. We present a formal summary of these results in section 3 after defining necessary preliminaries.

Under the realistic assumption of a limited amount of communication among private contacts, we show positive results for addressing general algorithmic problems with applications in areas including social-network-based recommender systems [42, 46]. We stress that our model, contrary to previous work [15], is not restricted to graph-based problems. In fact, we show results on general machine learning and data mining problems, finding heavy hitter [33] and solving clustering problems (like k-medians, k-means and k-centers) [11, 29].

To further motivate the model, we study a more restrictive case in which the devices cannot communicate directly with each other. In this *no-inter-communication* scenario, we show strong impossibility results for many problems of practical interest, including approximating the distance of users in social networks or counting the connected components of the graph. These impossibility results, in turn, motivate our model that allows minimal communication among devices. On the positive side, we show that, informally, even in this restrictive case, any problem that allows a streaming algorithm with a limited amount of space can be solved efficiently.

The rest of the paper proceeds as follows. In section 2, we introduce our models and review the prior public-private graph models. Then, in section 3, we present our main theoretical results for our model. Next, in section 4, we show empirical evaluation of algorithms in our model. In section 5, we review related work, and finally, in section 6, we present our conclusions.

2 ON-DEVICE PUBLIC-PRIVATE MODEL OF COMPUTATION

In this section, we formally define the *on-device public-private model* of computation. Before defining the model, we review the *public-private graph* model, which was introduced by Chierichetti et al. [15], to which we refer as *centralized* public-private model to distinguish it from the *on-device* version we introduce.

Centralized model. In the (centralized) public-private model of computation, we have a graph $G = (V, E)$, called the *public* graph, where V is the set of nodes representing users of a system and E is the set of public edges connecting them. Let $n = |V|$ and $m = |E|$. Then, for each node $u \in V$, we have an associated *private* graph $G_u = (\{u\} \cup V_u, E_u)$, over the set of nodes $V_u = \{v_1, \ldots, v_k\} \subseteq V$ representing the private edges of the node. Throughout the paper, we assume that the public and private graphs are undirected, unweighted graphs, but the model can be defined [15] in the obvious way for general graphs. We also focus on the most studied instance of the model (known as the star case) where the set E_u consists of a star centered in u (i.e., $E_u = \{u\} \times V_u$), which represents the private neighbors of the node u.

In the centralized public-private model of computation, we have a graph problem to solve (e.g., determining the connected components of the graph) and want to output efficiently the solution for the problem for the graph visible to any arbitrary user u (i.e., the public-private graph of node u, $G \cup G_u$) after performing some efficient preprocessing of the public graph G computed by a central machine. More precisely, let P be the preprocessing algorithm. Ideally, we want P to analyze the public graph G in a time that is polynomial in its size (i.e., $O(\text{poly}(n + m))$), and to output a space-efficient *synopsis* $P(G)$ of the public graph with space $\tilde{O}(n)$.[1] Then, the aim of the query algorithm is to take in input for node u the private graph G_u and to output the solution of the problem for $G \cup G_u$ by using the synopsis $P(G)$, in time and space that is ideally near-linear in $|E_u|$ and poly-log(n). Crucially, we observe that this model assumes that the query algorithm can access the entire synopsis by making queries to it that depend on the private data. Implicitly, this means that the private graph (or some information based on it) resides in the central machine.

On-device model. Motivated by the problem of social-network-based recommendations, in this paper we refine the previous model to enforce that each private graph G_u is stored exclusively on the device owned by the agent u to protect her privacy. We assume that private edges of E_u represent reciprocal private connections of the user u (i.e., $v \in E_u$ implies that $u \in E_v$ and that u and v know each other and they want their connection to be private to them and not shared to the central authority or any other user). Hence, contrary to the centralized model, the query algorithm must be executed on a device of limited computation power and (crucially) cannot make queries to the central machine that depend on the private edges, as this would reveal the private connections to the shared machine.

[1] we use $\tilde{O}(\cdot)$ to neglect polylogarithmic factors.

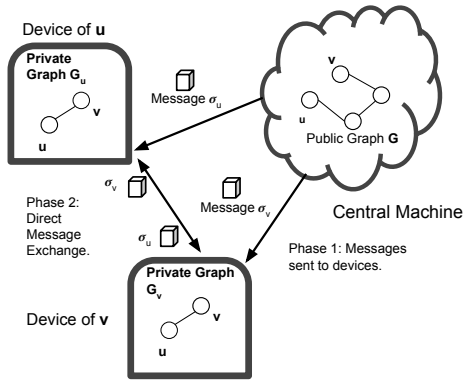


Figure 1: Description of the on-device public private model of computation

To model this, we assume, as in the previous model, that a central machine with high computational power can preprocess the public graph G in polynomial time. Then, contrary to the previous model, we assume that the query algorithm for $G \cup G_u$ is run on the private device of u that is not capable of storing the whole public graph. Indeed, as the complete graph might be large, we restrict the computation power (both space and running time) on each device to be sublinear in the size of the public graph $o(n + m)$, preferably poly-logarithmic in n, and polynomial in the size of the private graph, ideally $\tilde{O}(E_u)$. As this rules out the possibility for the device to store a synopsis of G of size $\Theta(n)$, we allow the device to communicate with the central machine and to the other private neighbors' devices with small-size messages.

The on-device communication protocol. More formally, in the on-device model, we assume that the central machine, after preprocessing the public graph G, sends a single message σ_u to the device corresponding to the node u of (ideally) poly-log(n) bits (we refer to Figure 1 for a pictorial description of the model). No other communication is allowed with the central machine from the device thereafter. Indeed, the shared machine has no information about the private graph G_u, and σ_u just depends on the public graph G and node u, thus preserving completely the privacy of the private contacts in G_u. Next, the device of u is allowed to directly query the device of each private neighbor $v \in E_u$ to receive the *same* message σ_v sent to v by the central machine. We observe that this communication does not involve the central authority and so does not leak any information to it. We also observe that the messages exchanged between a pair of private contacts do not contain private information (as they are obtained from the public machine and shared verbatim with the private contact), so no leakage is possible. Finally, after this exchange, the query algorithm on the device uses the messages received *only* to solve the problem.

For instance, in our book recommendation system example in the introduction, v might be a contact of u in u's favoring secure messaging app.[2] User u may obtain σ_v using the same messaging app to contact v, without thus sharing the app contact list to the book recommendation system provider. Then, u will use the messages received to compute the recommendations on her device.

Guarantees. Notice that, in this model, we protect *by design* all private information from being known by the central authority, as it is never reported to it. Notice, also, that nodes do not share private data with their neighbors (the messages exchanged contain only public data processed by the central machine). Thus, no leakage is possible from neighbors, by design, under the assumptions that the devices are able to establish a trusted connection with their private contacts, and the private contacts will not leak the existence of their connection to others. Notice that even if a private neighbor were to collude with the central machine, the only information that could be obtained by the machine would be the existence of the connection with that neighbor.

Alternative models without communication or a centralized system. Our model requires a single round of small-sized communication among private contacts. This assumption is indeed necessary to solve many important problems, as, in section 3.8, we show strong impossibility results in a model with no-inter-communication between devices. Finally, we observe that the protocol could be simulated without a central authority, if computational constraints were not present, but that would require to process all public data on the user's devices, which is not feasible for large-scale systems.

Metadata. So far, we have described a model which, as in [15], only involves graph information. Recently, another model has been defined that allows public-private (problem-specific) data associated with users as the public-private framework of data summarization [39]. In our on-device framework, as well, we allow both graph data and non-graph metadata on nodes. We assume that each node has a problem-specific public metadata S_u^P that can be shared publicly to everyone and a problem-specific private metadata S_u^V that is only available to u. This allows us to address non-graph-related problems such as clustering and heavy hitters, as we show in our paper. Similar to the graph information, the public metadata can be stored in the central machine, while the private metadata must be processed locally. We observe that more sophisticated access controls (who-can-access-what) can be defined, but we leave this to a future work. To handle the metadata, we seek algorithms that use polynomial time and space in the size of all public data for preprocessing and linear time in the private data of a user for the query algorithm.

3 ALGORITHMIC RESULTS FOR THE ON-DEVICE MODEL

In this section, we report our main algorithmic results. First, we study formally the relationships between the centralized and the on-device model in Section 3.1. Then, we show novel algorithmic results for the heavy hitters problem and k-clustering problem. Finally, we show impossibility results when the devices are not allowed to communicate between private contacts.

[2] In our model we abstract away the details of the communication protocols involved. We only assume that there is a way for the devices to obtain small messages from their private contacts without involving the central authority providing the recommendation system.

3.1 Relationships between the models

Before introducing our main novel algorithmic results, as a warm-up, we present a few basic lemmas to clarify the relationships between the on-device and the centralized model for graph algorithms. In the on-device model, we define Σ as the maximum size of a message exchanged in the protocol, while in the centralized algorithm, we define $|P(G)|$ as the size of the preprocessing synopsis obtained.

First, we prove that the on-device model is at least as hard as the centralized model, because any on-device algorithm can be implemented in the centralized setting.

Lemma 1. *Any on-device public-private graph algorithm with polynomial time preprocessing, $\Sigma \in \tilde{O}(1)$ message size and $\tilde{O}(|E_u|)$ query time, can be implemented in the centralized public-private graph model with polynomial time preprocessing, $|P(G)| \in \tilde{O}(n)$ preprocessing space and $\tilde{O}(|E_u|)$ query time.*

Proof. It is easy to observe that using the central machine, one can simulate the on-device protocol by storing in the synopsis all the messages sent to the nodes and reading them from memory during query time. The total space for the preprocessing is hence $O(n\Sigma)$, while the preprocessing time is clearly polynomial. At query time, the central machine reads the $\tilde{O}(|E_u|)$ messages "sent" to u and simulates the on-device query algorithm in $\tilde{O}(|E_u|)$ time. □

Another simple lemma shows that the contrary is not always true: the on-device mode is *strictly* harder.

Lemma 2. *There exist problems that admit a $|P(G)| \in O(n)$ preprocessing space in the centralized model and for which there is no $o(n)$ message size algorithm in the on-device public-private model.*

The proof is omitted for lack of space, but it is based on standard results in communication theory. All omitted proofs are deferred to the extended version of the paper.

Despite the on-device model being generally harder, we now show that for a significant class of centralized public-private graph algorithms, there is an immediate reduction to the on-device model.

Lemma 3. *Suppose for a given problem there exists a centralized public-private algorithm with synopsis $P(G)$ decomposable in n data-structures, $P(G)_u$ for $u \in V$ such that $|P(G)_u| \in \tilde{O}(1)$ and such that the query algorithm for node u solves the problem in $\tilde{O}(|E_u|)$ time by having access only to $P(G)_u$ and $P(G)_v$ for $v \in E_u$. Then, there exists a $\Sigma \in \tilde{O}(1)$ message size and $\tilde{O}(|E_u|)$ query time algorithm in the on-device model.*

Proof. We observe that the reduction is immediate because the central machine can send message $\sigma_u := P(G)_u$ with space $\tilde{O}(1)$ to each node u, and then the devices can implement the query algorithm using only the messages received from the central machine and their private contacts. □

The previous reductions yield directly algorithms in the on-device model for many problems already studied in the centralized model such as counting the number of connected components, estimating the size of the reachability set of a user, and several centrality estimations (we refer the reader to [15] for details) when the centralized public-private algorithm defined therein has the property required by Lemma 3. Conversely, we observe that this property does not hold for many important algorithms defined in centralized settings, including approximating shortest path distances between nodes u and v and estimating several notions of node similarities between them, because such algorithms require arbitrary access to the centralized synopsis. In particular, the central machine needs to be informed about the node v queried from u. Solving these problems while protecting the privacy of the query in our restrictive model is an interesting open problem.

3.2 Problems addressed

We now introduce our main algorithmic contributions in which we study a class of problems related to recommendation systems in the on-device public-private model. In a social-network recommendation system, there is a set of arbitrary items $\mathcal{S}$ and a graph indicating the social network. We assume the social graph includes a public part G and a private graph G_u for each node u as described above. For each vertex u, we have a set $S_u^P \subseteq \mathcal{S}$ of public items and a set $S_u^V \subseteq \mathcal{S}$ of private items, respectively, that u likes.

We model the problem of implementing a social recommendation system that, given the information in the neighborhood reachable within distance d of a vertex u in the graph visible to u, $G \cup G_u$, returns a subset of items I, with $|I| = k$ to recommend to u. More formally, for a graph $\mathcal{G}$, we let $N_u^d(\mathcal{G})$ be the set of vertices reachable from u within a distance at most d on graph $\mathcal{G}$. We let

$$S_u^d(\mathcal{G}) = S_u^V \cup \bigcup_{v \in N_u^d(\mathcal{G})} S_v^P$$

be the set of items visible to u in $\mathcal{G}$ (i.e., the private items of u plus all public items of nodes reachable from u at distance less than d in the graph $\mathcal{G}$). We will drop $\mathcal{G}$ when we refer to the public graph G. In particular, in our on-device model, we want to approximate the output of certain functions specifying relevant items over the set $S_u^d(G \cup G_u)$ to recommend. We show results for the following problems.

Uniform recommendation function. A uniform recommendation function is a randomized function that returns a subset of items uniformly at random without replacement from $S_u^d(G \cup G_u)$.

Linear recommendation function. Fix a node u, let S_v be $S_v := S_v^P$ for $v \neq u$ and $S_u := S_u^V \cup S_u^P$ for u. Notice that $S_u^d(G \cup G_u) = \bigcup_{v \in N_u^d(G \cup G_u)} S_v$. For a fixed u, item $i \in \mathcal{S}$, and distance d, we define $C_u^i(G \cup G_u)$ as the number of distinct nodes $v \in N_u^d(G \cup G_u)$ for each i that appears in the set S_v defined above. A linear recommendation function is a randomized function that returns a subset of k items I, where the items in I are chosen from $S_u^d(G \cup G_u)$ without replacement with probability proportional to $C_u^i(G \cup G_u)$.

Heavy hitter. Let M be a multiset of $\mathcal{S}$, i.e., a set of items with repetitions. For parameters ϵ and κ, heavy hitter is a function that returns a subset of items I from a multiset M with two properties. 1) I contains all items that appear in at least $\frac{|M|}{\kappa}$ times in M where $|M|$ is the total number of items counting repetitions; and 2) each item $i \in I$ appears at least $\frac{|M|}{\kappa} - \epsilon$ times in M.

For a node u and distance d, we let M be the multiset that contains the union (with repetitions) of S_v for distinct $v \in N_u^d(G \cup G_u)$ (i.e.,

Algorithm 1: General Algorithm.

Input: G, $\forall_{u\in V}(S_u^P, S_u^V, G_u)$, d, k
Output: Output k items for each node u

Pre-processing on central machine using G, $\forall_{u\in V} S_u^P$.
for all $u \in V$, for $d' \le d$ **do**
  Let $\bar{S}_u^{d'}(G) := \bigcup_{v\in N_u^d(G)} S_v^P$.
  Compute $I_u^{d'} := sketch(\bar{S}_u^{d'}(G))$.
  Send to u, $\sigma_u := (I_u^{d-1}, I_u^d)$.
Query algorithm on u's device using G_u, S_u^V.
Receive σ_u from central machine.
Send σ_u to v and receive σ_v from v for all $v \in E_u$.
Compute $I_u^V := sketch(S_u^V)$.
Compose sketches I_u^V, I_v^{d-1} for $v \in E_u$, and I_u^d to obtain sketch I_u.
Extract k items from I_u.

all items visible to u counted with the number of users liked by them). We seek to find all heavy hitters as defined above.

k-Clustering. In this case, each item in $\mathcal{S}$ represents a point in an arbitrary metric space, and the goal is to pick a set of k points $r_1, \ldots, r_k$ that minimizes the cost function.

$$\Big(\sum_{s\in S_u^d(G\cup G_u)} \min_{j=1}^{k} \mathrm{dist}(r_j, s)^\rho \Big)^{1/\rho},$$

for a particular fixed ρ. $\rho = 1$ corresponds to k-median, $\rho = 2$ to k-means and $\rho \to \infty$ to k-center.

In the next sections, we show algorithms with provable theoretical guarantees for all these problems in our model.

3.3 General algorithmic approach

The general approach of our algorithms (reported in Algorithm 1) for the previously mentioned problems is to define a composable sketch function *sketch* over sets of items such that 1) from $sketch(A)$, one can extract k items, approximately solving the problem; 2) $sketch(A)$ has small space $\tilde{O}(k)$; and 3) the sketches computed over two (possibly overlapping) sets of items A and B can be computed easily by composing $sketch(A)$ and $sketch(B)$. Assume that such a function exists; we will use it to compute a sketch $I_u^{d'}$ for the items in the neighborhood at distance at most d' from a node u on the public graph G over the public items S_u^P for all $u \in V$. These sketches will be computed over the public data in the centralized machine in the preprocessing phase for all integers $1 \le d' \le d$ using the public graph G and the public items S_u^P for all $u \in V$. We will send to each individual node u, as the message σ_u, the sketches for the greatest two distances (i.e., $\sigma_u := (I_u^{d-1}, I_u^d)$). Then, the nodes will share the message received from the central machine with their private neighbors and collect their messages. In the query phase, node u will compose: the sketch I_u^d and the sketches $I_v^{d-1} \forall v \in |E_u|$ received, as well as the sketch computed over the private local items S_u^V, to obtain a sketch I_u and to extract from it the recommendations.

Given this general framework, for each specific problem, we only need to show how the preprocessing phase obtains the sketches $I_u^{d'}$ efficiently, how they can be composed, and how to extract the solution from the sketch.

Algorithm 2: Algorithm for uniform recommendation function.

Input: G, $\forall_{u\in V}(S_u^P, S_u^V, G_u)$, d, k, a hash function $h(.)$
Output: Output k uniform sample items for each node u

Obtaining $I^{d'}$'s during preprocessing
For $d' = 0$
for all $u \in V$ **do**
  Sort items in $i \in S_u^P$ based on hash $h(i)$ and let I_u^0 be the first k items by $h(\cdot)$.
for $d' \in \{1 \cdots d\}$ for $u \in V$ **do**
  Let $\hat{I}_u^{d'}$ be the union of $I_u^{d'-1} \cup \bigcup_{v\in N(u)} I_v^{d'-1}$ where $N(u)$ is refers to graph G.
  Sort items in $\hat{I}_u^{d'}$ based on $h(i)$ and let $I_u^{d'}$ be the first k items.
Query algorithm on u's device using same $h(\cdot)$ function.
Let I_u^V be the top k items by $h(\cdot)$ in S_u^V.
Compute $\hat{I}_u$ as union of I_u^V, I_v^{d-1} for $v \in E_u$, and I_u^d.
Extract k items from $\hat{I}_u$ by $h(\cdot)$.

3.4 Uniform recommendation function

We define the sketch for the uniform case. We start with a simple sketching algorithm that does not work. Let us assume the public server defines as sketch I_u^d of uniform and independent sample of items from S_u^d. Suppose u has two neighbors, v' and v''. Let item $i \in S_{v'}^d$, $i \in S_{v''}^d$, and $i' \in S_{v'}^d$, but $i' \notin S_{v''}^d$. In this case, the probability that i exists in either one of the samples of v' or v'' is more than the probability for element i', which is not desired because it does not allow a simple combination of the two samples.

To resolve this issue, we assume that there exists a k-min-wise independent uniform hash function $h : \mathcal{S} \to [0, 1]$ from the items to $[0, 1]$. We assume that the function has shared random seeds that are available to all devices and the central machine (i.e., we can compute it consistently in every device). The algorithm is reported in Algorithm 2, in which we report only the details missing from the general Algorithm 1. To sketch a set of items S, we hash the items in S and keep the k items with minimum values. Two sketches for sets A, B can be easily composed because one can sort the items in $A \cup B$ and return the top k by h. Algorithm 2 shows how to use this property to iteratively construct $I_u^{d'}$ for all nodes based on $I_u^{d'}$ and how to initialize I_u^0, which is the sketch of S_u^P. Then, $I_u^{d'}$ is obtained by combining the sketches of $I_v^{d'-1}$ for the public neighbors of u and the previous sketch $I_u^{d'-1}$ of u. Finally, the last two sketches for d and $d-1$ are sent to each device. To construct uniform recommendations, for any v in the private neighborhood of u, we query v to obtain I_v^{d-1}, construct $\cup_{v'\in E_u\cup G_v} I_{v'}^{d-1} \cup I_u^d$, and compose it with the sketch of the private items in u. Finally, we recommend the k items with minimum hash value from the combined sketch.

The following lemma can be shown.

Theorem 4. *Algorithm 2 obtains k uniform recommendations without replacement for each node in the on-device model with total preprocessing time $\tilde{O}\left(\sum_u |S_u^P| + kd(m+n)\right)$ using $\tilde{O}(k)$ space messages and $\tilde{O}\left(|S_u^V| + k|E_u|\right)$ query time.*

We observe that one could use l_0 sketches [30] on each vertex to provide a uniform recommendation function. However, l_0 samplers

provide independent samples (i.e., with replacements). This can be resolved by using more sketches on each vertex and removing duplicated items, but l_0 samplers are designed to handle deletions of items as well, which is not necessary here.

3.5 Linear recommendation function

In this subsection, we provide a simple technique to recommend k items without replacement from $S_u^d(G \cup G_u)$ with probability proportional to $C_u^i(G \cup G_u)$.

First, we give a simple example to show that unlike the uniform sampling case, several copies of independent samplers are not useful to solve the problem efficiently. Suppose we have only items i and i', and $C_u^i(G \cup G_u) = n$ and $C_u^{i'}(G \cup G_u) = 1$. Note that any sampler that samples 2 items without replacement reports both i and i'. However, sampling one item gives i with probability $\frac{n}{n+1}$, and i' with probability $\frac{1}{n+1}$. Hence, in expectations, we need to make $O(n)$ linear samples with replacement to observe i' once, which is not desirable.

Here again, we use k-min-wise hash functions with shared random bits in a similar fashion to Algorithm 2. The key difference with the algorithm is in the sketch method, which instead of hashing items $\mathcal{S}$ to $[0, 1]$, it hashes pairs of items (i, u) to $[0, 1]$, where i is an item and u is a vertex containing i in its items. We first focus on the preprocessing. For each vertex u, we define $T_u^{d'} = \{(v, i) | v \in N_v^{d'}(G)$ and $i \in S_v^P\}$. We define the sketch $I_u^{d'}$ to be the set of k pairs in $T_u^{d'}$ with minimum hash values and with distinct items (i.e., when an item appears several times, we only keep the pair with lowest hash value for that item). It is easy to see that such sketches can be computed iteratively as in the uniform case. Notice also that at query time for node u, by having access to I_u^d, I_v^{d-1} for $v \in E_u$ and S_u^V, one can construct in the same way a sketch for $S_u^d(G \cup G_u) = \cup_{v \in N_u^d(G \cup G_u)} S_v$, where S_v is defined as $S_v := S_v^P$ for $v \neq u$ and $S_u = S_u^P \cup S_u^V$.

To construct the linear recommendations, we recommend the k distinct items with minimum hash values in the combined sketch of $S_u^d(G \cup G_u)$. Notice that the recommended set is equivalent to the set of the k distinct items with minimum hash values in $T_u^d(G \cup G_u) = \{(v, i) | v \in N_u^d(G \cup G_u)$ and $i \in S_v\}$. Hence, the probability that a pair corresponding to an item i has the minimum hash value is proportional to its frequency C_u^i as desired. After removing the pairs corresponding to the first item, the same argument holds for the second item, and so on. This allows the following theorem to be shown.

Theorem 5. *The previous algorithm obtains k linear recommendations without replacement for each node in the on-device model with total preprocessing time $\tilde{O}\left(\sum_u |S_u^P| + kd(m+n)\right)$ using $\tilde{O}(k)$ space messages and $\tilde{O}\left(|S_u^V| + k|E_u|\right)$ query time.*

3.6 Heavy Hitters

In this subsection, we provide a technique to recommend heavy hitters. Fix a probability $0 < \delta < 1$. Let $s = |\mathcal{S}|$ be the total number of items. We use a similar technique to the linear case, except that we use $H = 12 \log \frac{2s}{\delta} \epsilon^{-2}$ fully independent hash functions. As in the linear recommendations case, each hash function maps a pair (u, i) of user and item to $[0, 1]$. The sketch I_u^d keeps H pairs, one per function, such that for each function, the pair stored is the one with minimum hash value over the items inserted in the sketch. It can be observed that the sketch is composable. The construction of the sketches in the pre-processing and query time is the same as in the linear case. It can be noted that the final sketch I_u^d constructed at query time is a list of $\frac{12 \log \frac{2s}{\delta}}{\epsilon^2}$ items sampled independently, this time with replacement, from $N_u^d(G \cup G_u)$ with probability proportional to $C_u^d(G \cup G_u)$. We recommend an item if and only if it appears at least $\left(\frac{1}{\kappa} - \frac{\epsilon}{2}\right) \frac{12 \log \frac{2s}{\delta}}{\epsilon^2}$ times in the list I_u^d.

Given the result in Theorem 5, the proof can be adapted to show the following bound on the running time and message space.

Lemma 6. *The previous algorithm obtains runs in the on-device model with total pre-processing time $\tilde{O}\left(\sum_u |S_u^P| H + d(m+n)H\right)$ using $\tilde{O}(H)$ space messages and $\tilde{O}\left(|S_u^V| H + |E_u| H\right)$ query time.*

With a little more work, we can also show the following lemma.

Lemma 7. *The algorithm defined above returns the heavy hitter items in the neighborhood of u, with probability $1 - \delta$.*

3.7 k-clustering

Our algorithm in this section is based on the notion of composable core-sets [11]. Bateni et al. [11] show that given a linear space α-approximation algorithm for k-clustering problem for a fixed ρ norm (e.g., $\rho = 1$ for k-median, $\rho = 2$ for k-means, $\rho = \infty$ for k-center), the following procedure provides an $O(\alpha)$-approximate solution of the problem. Arbitrarily decompose the points into *disjoint* sets $U_1, \cdots, U_t$. Inside each set U_i select an approximate solution of size $\tilde{\Theta}(k)$. We call these points representatives. Then, for each point in U_i that is not a representative we move it to the location of the closest representative point in U_i. Now one can represent the new dataset by distinct point locations and their multiplicities, which are composable core-sets. Then, combine the composable core-sets and solve the problem on the new dataset. This is known to provide a $O(\alpha)$-approximation. Notice that the challenge to applying this method to our case is that the points in the sets S_u^P, S_u^V for $u \in V$ are not necessarily disjoint. We show, nevertheless, how to use this technique in our case.

Similar to the technique in previous sections, for each node u we keep a composable core-set of the points in $\bigcup_{v \in N_u^{d'}(G)} S_v^P$ for $d' = d$ and $d' = d - 1$, for each node. We do this, in this section, by first running a BFS search from each node u, gathering the points in S_v^P for $v \in N_u^{d'}(G)$, and finding the corresponding core-sets $I_u^{d'}$ as in [11]. Note that this preprocessing step can be implemented in $\tilde{O}(k|V|)$ space and polynomial time. Then, u receives I_v^{d-1} from each vertex v in its private neighborhood. Node u merges the userâĂŹs core-set I_u^d and the core-sets from private neighbors I_v^{d-1} and solves the problem. Note that core-sets I_v^{d-1} correspond to not necessarily disjoint sets. However, each point appears at most $|E_u| + 1$ times in these core-sets, where $|E_u|$ is the number of private neighbors of u. Therefore, we can show that our approximation factor is increased by a factor of at most $|E_u| + 1$. This implies the following theorem, where, as in [11], we assume there is a polynomial time, linear

space, α-approximation algorithm *Alg* for k-clustering in the norm ρ.

THEOREM 8. *The previous algorithm obtains a $O(\alpha|E_u|)$-approximate k-clustering in the norm ρ, for each node u, in the on-device model with polynomial time preprocessing, using $\tilde{O}(k)$ space messages, and using query time polynomial in $|S_u^V|$, k and $|E_u|$, as desired.*

Specifically, the total pre-processing time is

$$\tilde{O}\left(\sum_u f\left(\left|\bigcup_{v\in N_u^{d'}(G)} S_v^P\right|\right) + m + \sum_u \sum_{v\in N_u^d(G)} \left|S_u^P\right|\right)$$

and query time is $\tilde{O}\left(f\left(\left|S_u^V\right| + k|E_u|\right)\right)$, where $f(.)$ indicates the running time of the offline algorithm *Alg*.

The increase $|E_u|$ in the approximation factor in the previous algorithm is due to the possibility of items appearing in multiple core-sets. In the next subsection we provide a sketching technique that allows us to reduce the weight of the duplicated points from the core-sets. By applying this sketching, our algorithm is (approximately) using the true weight of the points in the core-set. While this heuristic does not improve the approximation factor in the worst case, this theoretically sound heuristic improves the results in practice.

3.7.1 Removing duplicated items. As we mentioned earlier in this subsection, composable core-sets are designed to work with disjoint sets (i.e., a decomposition of the input set). However, in the case of our application, if a vertex v is reachable from two private neighbors of u (via distance less than d), the items (i.e., points) of v appear in both of these neighbors. Hence, we double count the weight of these items on the representative items. If we could explicitly keep the set of the items assigned to the representative items (which we refer to as $A_1, \ldots, A_t$ for simplicity of notation), we could directly indicate the duplicated items and remove their weights from the representative items. However, this might not be practical due to the large size of these sets. In this subsection we provide a technique to sketch these sets such that we can remove the weight of the duplicated items fairly accurately (i.e., with an ϵ error) using a smaller memory.

Next, we provide our tool to deal with sets that are not disjoint for composable core-sets. This tool might be of independent interest.

Let A be a collection of items and let $A_1, \ldots, A_t \subseteq A$ be a collection of subsets of A. Without loss of generality, we assume $|A_1| \geq |A_2| \geq \cdots \geq |A_t|$. Let Δ be an upper bound on the number sets that contain a, for any arbitrary $a \in A$. Let $h(.)$ be a universally shared hash function that hashes $A \to [0,1]$ uniformly at random. We let $\hat{A}_i$ be the subset of A_i with hash value at most $\lambda_i = \frac{6\log(\delta t)\Delta^2}{\epsilon^2}\frac{1}{|A_i|}$. Note that we have $E[|\hat{A}_i|] = |A_i|\frac{6\log(\delta t)\Delta^2}{\epsilon^2}\frac{1}{|A_i|} = \frac{6\log(\delta t)\Delta^2}{\epsilon^2} \in \tilde{O}(\Delta^2)$.

For $i \in \{1, \ldots, t\}$ we define $B_i = \left\{a \middle| a \in A_t \& \nexists_{j>i} a \in A_j\right\}$. This intuitively says that to construct B_is from A_is, we remove duplicated items from the larger sets and keep them in the smaller ones. Similarly, for $i \in \{1, \ldots, t\}$ we define $\hat{B}_i = \left\{a \middle| a \in \hat{A}_t \& \nexists_{j>i} a \in \hat{A}_j\right\}$.

LEMMA 9. *With probability at least $1 - \delta^{-1}$, simultaneously, for all $i \in \{1, \ldots, t\}$ we have*

$$\left||B_i| - \frac{1}{\lambda_i}|\hat{B}_i|\right| \leq \epsilon\frac{|A_i|}{\Delta}.$$

The proof is omitted for lack of space.

3.8 Hardness results for no-inter-communication models

To motivate our model, here we consider a more restrictive case where we do not allow the nodes to exchange the messages σ's with their private contacts. Hence, after the message σ_u is received, the query algorithm has to solve the problem based exclusively on the local private graph G_u and the message σ_u. We call this model *no-inter-communication* model.

We show that it is impossible to solve several basic graph problems in the *no-inter-communication* public-private model, which motivates our more general model and distinguishes the computation power of these two models. We summarize our negative results in the following theorem.

THEOREM 10. *It is information-theoretically impossible to solve the following problems in the* no-inter-communication *public-private model using $o(n)$ size messages:*

- *Approximate the distance of u and v.*
- *Decide if u is connected to v.*
- *Approximate the number of connected components.*

The full proof of all the results is available in the full version. For lack of space, here we describe the intuition behind them. Our hardness results are based on two-party communication complexity techniques. Note that we provide information-theoretic hardness, and hence we do not require computational complexity assumptions like $P \neq NP$.

We use the hardness of the indexing problem [32] and set disjointness [31, 44]. In the indexing problem, we have two parties: namely, Alice and Bob. Alice holds a subset $A \subseteq \{1, 2, \ldots, n\}$ and Bob holds an element a. The goal is for Bob to detect whether or not $a \in A$, using a one-way communication link from Alice to Bob. It is known that $\Omega(n)$ bits of communication are required to solve this problem.

In the set disjointness problem as well, we have two parties, Alice and Bob, each holding a subset of $\{1, 2, \ldots, n\}$. Say Alice holds A and Bob holds B. The goal here is to detect whether or not $A \cap B = \emptyset$, using a two-way communication link. This problem requires $\Omega(n)$ bits of communication as well.

To prove the above theorem, let us define the reachability problem as follow. We have a network G with a vertex r indicated as the root. A vertex v is reachable if there is a path from v to the root. Indeed, the goal is to detect whether each vertex is reachable. The next theorem shows the hardness of solving the reachability problem in the *no-inter-communication* model.

LEMMA 11. *It is not possible to solve the reachability problem in the* no-inter-communication *model using $o(n)$ size messages to the device.*

Next we show the hardness of counting the number of connected components.

Lemma 12. *Let $\epsilon \in (0,1]$ be an arbitrary small constant. It is not possible to approximate the number of connected components with an approximation factor ϵ in the* no-inter-communication *model using $o(n)$ communication with each device.*

Let us define the delay approximation problem as follows. We have an undirected unweighted network G with a vertex r indicated as the root. The delay of a vertex v is equal to the length of the shortest path between v and the root. The delay approximation problem is to approximate the length of the shortest path between each vertex and the root. The next theorem shows the hardness of delay approximation in the *no-inter-communication* model.

Lemma 13. *It is not possible to solve the delay approximation problem with an approximation factor $\omega(\frac{1}{n})$ in the* no-inter-communication *model using $o(n)$ communication with each local device.*

Positive results. Despite the strong impossibility results presented above, we show a reduction from the popular class of streaming algorithms (with $\tilde{O}(1)$ space). This result translates any streaming algorithm to a *no-inter-communication* on-device public-private algorithm and directly provides a few basic algorithms for the *no-inter-communication* model.

Theorem 14. *Consider a problem that admits an (insertion-only) streaming algorithm with $\tilde{O}(1)$ space, T_0 update time per element inserted, and T_1 query time. The problem can be solved in the* no-inter-communication *public-private setting using $\tilde{O}(1)$ message space and $\tilde{O}(|E_u|T_0 + T_1)$ query time.*

One application of this theorem is to estimate the size of the maximum matching in planar graphs (and bounded arboricity graphs). This can be done with $\tilde{O}(1)$ space, $\tilde{O}(1)$ update time, and $\tilde{O}(1)$ query time [37]. In fact, none of the other results presented in this paper can be derived from this theorem.

4 EXPERIMENTS

In this section, we empirically evaluate the efficiency of our algorithms in the on-device public-private model. For experimentation purposes, to ensure replicability of the results, and for privacy reasons, we exclusively use public-available data for all our experiments. All our datasets are obtained from the Stanford's SNAP dataset (http://snap.stanford.edu/data) or from the DBLP public data repository of computer science publication records (https://dblp.uni-trier.de/). Of course, these datasets do not contain private data, so as in previous works [15, 39], we simulate the public-private data split in our dataset. We first describe the datasets used and then show how we simulated the public-private data split.

Uniform sampling, linear sampling, and heavy hitter. To evaluate our algorithms for uniform sampling, linear sampling, and heavy hitters, we use three datasets, referred to as `dblp`, `foods` [35], and `movies` [35]. From these three datasets we extract a collection of user-user relationships (representing the social network of the users) and a collection of user-item pairs (representing the items associated to each user). We first present statistics on these datasets.

- `dblp` contains 1.1×10^6 nodes 8.6×10^6 links and 8.1×10^3 distinct items (for 3.5×10^6 total user-items relationships)
- `foods` contains 2.1×10^5 nodes 1.8×10^7 links and 6.8×10^4 distinct items (for 4.7×10^5 total user-items relationships).
- `movies` contains 9.4×10^5 nodes 1.7×10^8 links and 2.4×10^5 distinct items (for 6.7×10^6 total user-items relationships).

For `dblp` the nodes are authors, and the items associated with a node correspond to the venues in which the author published. The social graph instead represents co-authorship relationships among nodes. For `foods` and `movies`, the nodes are users in a recommender system (see [35] for more details), while the user-items relationships represent the objects each user reviewed. The edges in the social network represent co-reviewing relationships.

k-Clustering. To evaluate our algorithms for the k-clustering problem instead we use a dataset where items are associated with a point in a metric space. We use the `gowalla` dataset [16] which contains 2.0×10^5 nodes 9.5×10^5 links and a total of 6.4×10^6 distinct items. The items represent check-ins corresponding to 2d location data. We use $\rho = \infty$ for the norm, corresponding to the well studied k-center problem. The edges in this dataset are the social connections among users.

Simulating the public-private graphs. To obtain examples of public-private graphs as in [15, 39] we mark independently and uniformly a fraction p of the edges in each dataset as private and remove them from the public graph G while adding each private edge u, v to the private graphs G_u and G_v. We use $p = 1/2$ in the experiments reported, but this coefficient does not qualitatively affect the results obtained. As the most salient aspect of the model is the sketching of the public data in the preprocessing phase we mark all items as public in our experiments, again this does not qualitatively affect the results significantly.

4.1 Results for uniform sampling, linear sampling, and heavy hitter

Here we experiment with our algorithms for uniform sampling, linear sampling, and heavy hitter. In this setting, we use $d = 2$ and $d = 3$ for the radius of the neighborhood around a node for which we extract the suggestions. We chose this range as it is large enough to be algorithmically challenging but not so large that it results in little variability between node suggestions (it is well known that most nodes are at distance 4 to 5 from each other [8, 34]). We experiment with the recommendation number k in the range $[1, 10]$. For heavy hitter we set $\kappa = k$, $\epsilon = \frac{1}{2k}$, and probability of success to 90% (i.e, we upper bound the number of recommendations by k). For stability of the results, when we report the running time, we show the mean of 100 independent executions.

Recall that our aim is to provide to node u suggestions from $S_u^d(G \cup G_u) = S_u^V \cup \bigcup_{v \in N_u^d(G \cup G_u)} S_v^P$ according to the definitions of the specific problem (uniform sampling, linear sampling, and heavy hitter). Notice that our on-device algorithms for uniform sampling and linear sampling are guaranteed to have the same output of a centralized algorithm that analyzes all the data (private and public) in $S_u^d(G \cup G_u)$ and then computes the suggestions. Moreover, our on-device algorithm for heavy hitters has the same output (and hence accuracy) of the centralized sketch-based algorithm running on all the data in $S_u^d(G \cup G_u)$. As such, in our experiments for these problems, we focus on the efficiency of the algorithms. Our main aim is to show that executing the query algorithm for a single

node is highly efficient and can be performed on an inexpensive device. In particular, we seek to show that our preprocessing allows the query time to be much faster than without preprocessing the data. To show this, we make the conservative comparison of the running time of our query algorithm, executed for one node u, with an algorithm that simply loads in memory all data in $S_u^d(G \cup G_u)$ (which is much faster than an algorithm computing the suggestions). This will show the gains that come from the preprocessing. Also, we show the preprocessing algorithm running time to evaluate its scalability in a centralized system. For ease of comparison of the preprocessing and query algorithm running times, we ran all our algorithms (preprocessing and query) as a standard sequential single-core job on the same machine (we used an Intel Xeon CPU E5-1650 v4 with 6 cores at 3.60 GHz, running Debian). Each job uses a single core, but we observe that the preprocessing algorithms lead to an obvious distributed implementation. We observe for the query algorithms very little memory use (less than 1 MB for the data storage), while the preprocessing used less than 10 GB of RAM.



(a) `movies` preprocessing time



(b) `movies` query time

Figure 2: Running time of the preprocessing and query algorithm for heavy hitter.

Preprocessing. We report the running time of our preprocessing algorithms in Figures 2a for the heavy hitter case in `movies`. We omit the figures for the uniform and linear sampling case for lack of space as well as the figures for `dblp`, `foods`. The results are qualitatively similar in all datasets and problems more details are deferred to the full version of the paper.

From the charts, we can make the following observations. Note that as expected, the number of recommendations has only an almost-linear effect on the preprocessing time. The radius d has also a moderate effect on the preprocessing time.

Query time. We now focus on the query time, which is reported in Figure 2b for heavy hitter in `movies` (results for linear and uniform sampling and for other datasets are omitted for lack of space, but are qualitatively very similar). Here we can observe that the diameter, as expected, does not affect the query time. As expected, increasing the number of recommendations has a linear effect on the query time. We observe running times that range from 0.5 to 2.00 milliseconds per query in heavy hitter for the movies dataset (see Figure 2b), showing great efficiency.

Recall that to show a conservative estimate of the speedup of the running time of our query algorithm due to the preprocessing, we compare its mean running time (over 100 random nodes u) with the mean running time of an algorithm that simply loads in memory all data in $S_u^d(G \cup G_u)$. Our experiments for linear sampling, uniform sampling, and heavy hitter shows that the query time for `dblp` obtains a speedup of a factor 10 to 100 for $d = 2$ and a factor 100 to 1000 for $d = 3$ by the preprocessing. Similarly, for `foods` we see a factor 20 to 200 for $d = 2$ and a factor 400 to 4000 for $d = 3$ and for `movies` a factor 200 to 2000 for $d = 2$ and a factor 15000 to 150000 for $d = 3$.

4.2 Results for clustering

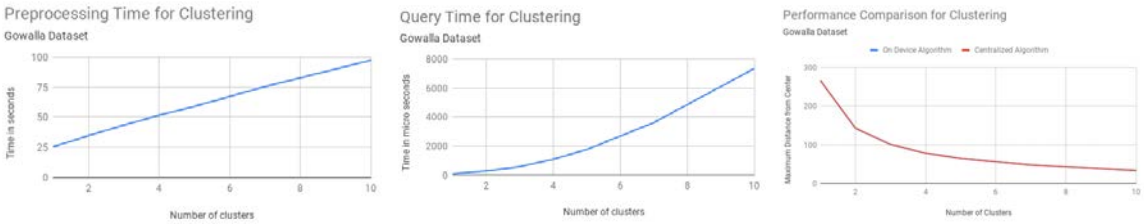
In this section, we show similar experimental results for our algorithm for k-center (i.e., k-clustering with $\rho = \infty$) in the `gowalla` dataset. We set the number of points in the core-set to be k and we set the de-duplication method to use $E[|\hat{A}_i|] = 10|A_i|$. As clustering algorithm, to obtain the core-sets we use the standard k-center greedy algorithm. The running times are reported in Figure 3a and Figure 3b for preprocessing and query algorithm, respectively, for $d = 2$ and $k \in [1, 10]$ clusters. The preprocessing time is again linearly related to k. The average running time per query is less than 9 milliseconds. Here we observe that the running time has a more than linear relationship with k but still is extremely fast for all k. Again, as a benchmark, we compare the running times of the query algorithm with a centralized algorithm: gathering the items on a 2-neighborhood of the query node, we run the same greedy k-center algorithm and observe an increase in speed from a factor of 38 to a factor of 700 times for $d = 2$. Again, this confirms that the preprocessing improves performances significantly.

Then, in Figure 3c, we evaluated the cost of the clustering solution obtained by our algorithm (using the point de-duplication technique) and the cost obtained by a centralized clustering algorithm that runs over the correct dataset without duplicates. We recall that the issue with the composability of the sketches is that the points in the sketches for the private neighbors are not disjoint (notice that this is true even if each user has distinct points, because the sketches include points at distance 2 from the neighbor). For this reason, we introduced a sketching technique to estimate the correct multiplicity of the points in the core-sets. As it is possible to see in the figure, the two lines are almost coincident. In fact, we observe less than 1% variation in the cost of the two solutions, confirming the accuracy of the method beyond the worst case guarantees in this dataset.

5 RELATED WORK

Related work to our paper spans a variety of areas, so we will focus only on the most directly related ones.

Our paper is motivated by the importance of preserving privacy in social-network-based systems. This area has attracted significant attention [21, 26, 27, 48]. Several authors describe attacks and remedies to potential attacks on social-networks' privacy [9,



(a) `gowalla` preprocessing time **(b) `gowalla` query time** **(c) `gowalla` clustering solution cost**

Figure 3: Running time of the preprocessing and query algorithm for clustering and cost of the clustering solution.

47, 48]. Users are increasingly interested in privacy controls of social-network data as Dey et al. [21] observe.

Private-public data model. This interest has motivated many privacy-preserving frameworks in the past. The one most related to our work is the public-private graph framework [15] described in section 2. Chierichetti et al. [15] show how the framework allows multiple graph problems to be solved, including connected components, all-pair distance estimation [19], node similarity scores as social affinity [41], correlation clustering [10], shortest paths [20], and Personalized PageRank computation [25]. Notice that Chierichetti et al. [15] do not explicitly address the issue of keeping the user's private data on the device of limited capacity because it models a centralized setting.

The original framework was defined for graph problems only, however, other non-graph problems have been studied such as the data summarization framework [39], where users possess both private and public data and one wants to summarize the data for personalized recommender systems. The authors show succinct summaries for optimizing submodular user utility functions. Similarly, Archer et al. [7] study indexing problems in public-private graphs, where one wants to retrieve nodes visible to a user. Neither Mirzasoleiman et al. [39] nor Archer et al. [7] define explicitly a notion of on-device computation.

Other privacy-preserving learning models. Related to ours is the rich area of research on differential privacy [1, 22, 23]. Differential privacy was originally proposed as a model for preventing leakage of individuals' information from the output of a computation over private databases. In a nutshell, a differentially private algorithm is guaranteed to have an output that is mostly unaffected by the data of any given user, in a strong probabilistic sense. This strong constraint limits the information over the private data that an attacker can learn. This method has found wide adoption in many areas of computer science, including deep learning [1], for its theoretical elegance and its strong guarantees.

The original applications of differential privacy were in centralized settings, but recently, several studies have used it to protect communication of on-device data to a central authority. An example is Qin et al. [43], who address recommender system problems using differential privacy in a setting not involving social-graph-based information (unlike our paper). More generally, the local differential privacy [24] technique consisting of perturbing local information before sending it to the central authority has received wide attention because it allows the computation of aggregate statistics over user data.

Another related topic is the setting of federated learning [38]. In a federated learning scenario, the users of a system want to optimize a shared machine learning model (e.g., training a single model for all users, using their own private on-device data in a distributed setting without sharing it to the central machine). In this setting, several works [14, 38] show strong theoretical guarantees of privacy based on differential privacy and sophisticated cryptographic protocols [14]. We observe that the setting is different from ours because it does not directly model social-graph-based information, and it focuses on the problem of using every user's data to improve machine learning models for all users.

Contrary to previous work, in our on-device framework, we do not rely directly on differential privacy techniques, as we guarantee, by design, an even stronger notion of privacy because private data is only used to provide output to the data owner. We also observe that our model does not require the insertion of noise in the output, the estimation of output sensitivity [23], or the use of specific cryptographic protocols. However, we believe that the wealth of theoretical results on differential privacy, federated learning, and other areas conceivably can be used to address a wide class of problems in variants of our model, an effort that we leave as an open problem.

Algorithmic problems addressed in our work. Our paper addresses several algorithmic problems in heavy hitters [33] and clustering [4, 29], both of which have a vast literature. Related to our algorithmic techniques is the area of streaming algorithms and streaming data analysis [30, 36, 45]. In particular, we use extensively techniques from sketching [5, 6] that have found applications in graph problems [13, 17, 18, 36, 40]. Particularly related are the notions of composable sketches, which are also related to the concept of composable core-sets [3, 28], mergeable summaries [2], and mapping core-sets for clustering [11].

Finally, to obtain our impossibility results, we use a few important results such as [31, 32, 44].

6 CONCLUSIONS

We introduce a new model for addressing several computational problems while providing strong privacy guarantees for the data and contacts present on a user's device. We believe that our paper further shows how techniques from streaming and sketching can

be used for privacy-preserving computations in on-device computations. As a future work, we would like to further explore the connection of the area of streaming and privacy to address more problems in the on-device public-private model. Also, an interesting area of study is the use of differential-privacy methods in this model.

Acknowledgments. We thank Flavio Chierichetti, Ravi Kumar and Silvio Lattanzi for their work on the public-private graph model that inspired this work.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 308–318.

[2] Pankaj K Agarwal, Graham Cormode, Zengfeng Huang, Jeff M Phillips, Zhewei Wei, and Ke Yi. 2013. Mergeable summaries. *ACM Transactions on Database Systems (TODS)* 38, 4 (2013), 26.

[3] Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. 2004. Approximating extent measures of points. *Journal of the ACM (JACM)* 51, 4 (2004), 606–635.

[4] Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. 2017. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*. Ieee, 61–72.

[5] Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. 2012. Analyzing graph structure via linear measurements. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*. SIAM, 459–467.

[6] Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. 2012. Graph sketches: sparsification, spanners, and subgraphs. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*. ACM, 5–14.

[7] Aaron Archer, Silvio Lattanzi, Peter Likarish, and Sergei Vassilvitskii. 2017. Indexing Public-Private Graphs. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1461–1470.

[8] Lars Backstrom, Paolo Boldi, Marco Rosa, Johan Ugander, and Sebastiano Vigna. 2012. Four degrees of separation. In *Proceedings of the 4th Annual ACM Web Science Conference*. ACM, 33–42.

[9] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. 2007. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 181–190.

[10] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. 2004. Correlation clustering. *Machine learning* 56, 1-3 (2004), 89–113.

[11] MohammadHossein Bateni, Aditya Bhaskara, Silvio Lattanzi, and Vahab Mirrokni. 2014. Distributed balanced clustering via mapping coresets. In *Advances in Neural Information Processing Systems*. 2591–2599.

[12] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. 2013. Recommender systems survey. *Knowledge-based systems* 46 (2013), 109–132.

[13] Paolo Boldi, Marco Rosa, and Sebastiano Vigna. 2011. HyperANF: Approximating the neighbourhood function of very large graphs on a budget. In *Proceedings of the 20th international conference on World wide web*. ACM, 625–634.

[14] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1175–1191.

[15] Flavio Chierichetti, Alessandro Epasto, Ravi Kumar, Silvio Lattanzi, and Vahab Mirrokni. 2015. Efficient algorithms for public-private social networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 139–148.

[16] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1082–1090.

[17] Edith Cohen. 1997. Size-estimation framework with applications to transitive closure and reachability. *J. Comput. System Sci.* 55, 3 (1997), 441–453.

[18] Edith Cohen. 2015. All-distances sketches, revisited: HIP estimators for massive graphs analysis. *IEEE Transactions on Knowledge and Data Engineering* 27, 9 (2015), 2320–2334.

[19] Atish Das Sarma, Sreenivas Gollapudi, Marc Najork, and Rina Panigrahy. 2010. A sketch-based distance oracle for web-scale graphs. In *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 401–410.

[20] Camil Demetrescu and Giuseppe F Italiano. 2004. A new approach to dynamic all pairs shortest paths. *Journal of the ACM (JACM)* 51, 6 (2004), 968–992.

[21] Ratan Dey, Zubin Jelveh, and Keith Ross. 2012. Facebook users have become much more private: A large-scale study. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*. IEEE, 346–352.

[22] Irit Dinur and Kobbi Nissim. 2003. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 202–210.

[23] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.

[24] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. ACM, 1054–1067.

[25] Dániel Fogaras, Balázs Rácz, Károly Csalogány, and Tamás Sarlós. 2005. Towards scaling fully personalized pagerank: Algorithms, lower bounds, and experiments. *Internet Mathematics* 2, 3 (2005), 333–358.

[26] David Garcia. 2017. Leaking privacy and shadow profiles in online social networks. *Science advances* 3, 8 (2017), e1701172.

[27] Ralph Gross and Alessandro Acquisti. 2005. Information revelation and privacy in online social networks. In *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*. ACM, 71–80.

[28] Piotr Indyk, Sepideh Mahabadi, Mohammad Mahdian, and Vahab S Mirrokni. 2014. Composable core-sets for diversity and coverage maximization. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 100–108.

[29] Anil K Jain. 2010. Data clustering: 50 years beyond K-means. *Pattern recognition letters* 31, 8 (2010), 651–666.

[30] Hossein Jowhari, Mert Sağlam, and Gábor Tardos. 2011. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *Proceedings of the thirtieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 49–58.

[31] Bala Kalyanasundaram and Georg Schintger. 1992. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics* 5, 4 (1992), 545–557.

[32] Ilan Kremer, Noam Nisan, and Dana Ron. 1999. On randomized one-round communication complexity. *Computational Complexity* 8, 1 (1999), 21–49.

[33] Kasper Green Larsen, Jelani Nelson, Huy L Nguyên, and Mikkel Thorup. 2016. Heavy hitters via cluster-preserving clustering. *arXiv preprint arXiv:1604.01357* (2016).

[34] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2005. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 177–187.

[35] Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 897–908.

[36] Andrew McGregor. 2014. Graph stream algorithms: a survey. *ACM SIGMOD Record* 43, 1 (2014), 9–20.

[37] Andrew McGregor and Sofya Vorotnikova. 2018. A simple, space-efficient, streaming algorithm for matchings in low arboricity graphs. In *OASIcs-OpenAccess Series in Informatics*, Vol. 61. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[38] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial Intelligence and Statistics*. 1273–1282.

[39] Baharan Mirzasoleiman, Morteza Zadimoghaddam, and Amin Karbasi. 2016. Fast distributed submodular cover: Public-private data summarization. In *Advances in Neural Information Processing Systems*. 3594–3602.

[40] Christopher R Palmer, Phillip B Gibbons, and Christos Faloutsos. 2002. ANF: A fast and scalable tool for data mining in massive graphs. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 81–90.

[41] Rina Panigrahy, Marc Najork, and Yinglian Xie. 2012. How user behavior is related to social affinity. In *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 713–722.

[42] Moon-Hee Park, Jin-Hyuk Hong, and Sung-Bae Cho. 2007. Location-based recommendation system using Bayesian user's preference model in mobile devices. In *International Conference on Ubiquitous Intelligence and Computing*. Springer, 1130–1139.

[43] Zhan Qin, Yin Yang, Ting Yu, Issa Khalil, Xiaokui Xiao, and Kui Ren. 2016. Heavy hitter estimation over set-valued data with local differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 192–203.

[44] Alexander A. Razborov. 1992. On the distributional complexity of disjointness. *Theoretical Computer Science* 106, 2 (1992), 385–390.

[45] Jonathan A Silva, Elaine R Faria, Rodrigo C Barros, Eduardo R Hruschka, Andre CPLF De Carvalho, and João Gama. 2013. Data stream clustering: A survey. *ACM Computing Surveys (CSUR)* 46, 1 (2013), 13.

[46] Frank Edward Walter, Stefano Battiston, and Frank Schweitzer. 2008. A model of a trust-based recommendation system on a social network. *Autonomous Agents and Multi-Agent Systems* 16, 1 (2008), 57–74.

[47] Lingjing Yu, Sri Mounica Motipalli, Dongwon Lee, Peng Liu, Heng Xu, Qingyun Liu, Jianlong Tan, and Bo Luo. 2018. My Friend Leaks My Privacy: Modeling and Analyzing Privacy in Social Networks. In *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*. ACM, 93–104.

[48] Elena Zheleva and Lise Getoor. 2009. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th international conference on World wide web*. ACM, 531–540.

Graph Neural Networks for Social Recommendation

Wenqi Fan
Department of Computer Science
City University of Hong Kong
wenqifan03@gmail.com

Yao Ma
Data Science and Engineering Lab
Michigan State University
mayao4@msu.edu

Qing Li
Department of Computing
The Hong Kong Polytechnic University
csqli@comp.polyu.edu.hk

Yuan He
JD.com
heyuan6@jd.com

Eric Zhao
JD.com
ericzhao@jd.com

Jiliang Tang
Data Science and Engineering Lab
Michigan State University
tangjili@msu.edu

Dawei Yin
JD.com
yindawei@acm.org

ABSTRACT

In recent years, Graph Neural Networks (GNNs), which can naturally integrate node information and topological structure, have been demonstrated to be powerful in learning on graph data. These advantages of GNNs provide great potential to advance social recommendation since data in social recommender systems can be represented as user-user social graph and user-item graph; and learning latent factors of users and items is the key. However, building social recommender systems based on GNNs faces challenges. For example, the user-item graph encodes both interactions and their associated opinions; social relations have heterogeneous strengths; users involve in two graphs (e.g., the user-user social graph and the user-item graph). To address the three aforementioned challenges simultaneously, in this paper, we present a novel graph neural network framework **(GraphRec)** for social recommendations. In particular, we provide a principled approach to jointly capture interactions and opinions in the user-item graph and propose the framework GraphRec, which coherently models two graphs and heterogeneous strengths. Extensive experiments on two real-world datasets demonstrate the effectiveness of the proposed framework GraphRec.

CCS CONCEPTS

• **Information systems** → **Social recommendation**; • **Computing methodologies** → **Neural networks**; **Artificial intelligence**.

KEYWORDS

Social Recommendation; Graph Neural Networks; Recommender Systems; Social Network; Neural Networks

ACM Reference Format:
Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313488

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313488

1 INTRODUCTION

The exploitation of social relations for recommender systems has attracted increasing attention in recent years [18, 28, 30]. These social recommender systems have been developed based on the phenomenon that users usually acquire and disseminate information through those around them, such as classmates, friends, or colleagues, implying that the underlying social relations of users can play a significant role in helping them filter information [23]. Hence, social relations have been proven to be helpful in boosting the recommendation performance [8, 29].

Recent years have witnessed great developments in deep neural network techniques for graph data [15]. These deep neural network architectures are known as Graph Neural Networks (GNNs) [5, 10, 19], which have been proposed to learn meaningful representations for graph data. Their main idea is how to iteratively aggregate feature information from local graph neighborhoods using neural networks. Meanwhile, node information can be propagated through a graph after transformation and aggregation. Hence, GNNs naturally integrate the node information as well as the topological structure and have been demonstrated to be powerful in representation learning [5, 7, 15]. On the other hand, data in social recommendation can be represented as graph data with two graphs. As demonstrated in Figure 1, these two graphs include a social graph denoting the relationships between users, and a user-item graph denoting interactions between users and items. Users are simultaneously involved in both graphs, who can bridge them. Moreover, the natural way of social recommendation is to incorporate the social network information into user and item latent factors learning [37]. Learning representations of items and users is the key to build social recommender systems. Thus, given their advantages, GNNs provide unprecedented opportunities to advance social recommendation.

Meanwhile, building social recommender systems based on GNNs faces challenges. The social graph and the user-item graph in a social recommender system provide information about users from different perspectives. It is important to aggregate information

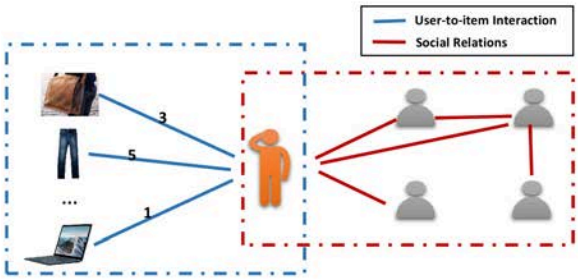


Figure 1: Graph Data in Social Recommendation. It contains two graphs including the user-item graph (left part) and the user-user social graph (right part). Note that the number on the edges of the user-item graph denotes the opinions (or rating score) of users on the items via the interactions.

from both graphs to learn better user representations. Thus, the first challenge is how to inherently combine these two graphs. Moreover, the user-item graph not only contains interactions between users and items but also includes users' opinions on items. For example, as shown in Figure 1, the user interacts with the items of "trousers" and "laptop"; and the user likes "trousers" while disliking "laptop". Therefore, the second challenge is how to capture interactions and opinions between users and items jointly. In addition, the low cost of link formation in online worlds can result in networks with varied tie strengths (e.g., strong and weak ties are mixed together) [36]. Users are likely to share more similar tastes with strong ties than weak ties. Considering social relations equally could lead to degradation in recommendation performance. Hence, the third challenge is how to distinguish social relations with heterogeneous strengths.

In this paper, we aim to build social recommender systems based on graph neural networks. Specially, we propose a novel graph neural network **GraphRec** for social recommendations, which can address three aforementioned challenges simultaneously. Our major contributions are summarized as follows:

- We propose a novel graph neural network GraphRec, which can model graph data in social recommendations coherently;
- We provide a principled approach to jointly capture interactions and opinions in the user-item graph;
- We introduce a method to consider heterogeneous strengths of social relations mathematically; and
- We demonstrate the effectiveness of the proposed framework on various real-world datasets.

The remainder of this paper is organized as follows. We introduce the proposed framework in Section 2. In Section 3, we conduct experiments on two real-world datasets to illustrate the effectiveness of the proposed method. In Section 4, we review work related to our framework. Finally, we conclude our work with future directions in Section 5.

2 THE PROPOSED FRAMEWORK

In this section, we will first introduce the definitions and notations used in this paper, next give an overview about the proposed framework, then detail each model component and finally discuss how to learn the model parameters.

Table 1: Notation

Symbols	Definitions and Descriptions
r_{ij}	The rating value of item v_j by user u_i
$\mathbf{q}_j$	The embedding of item v_j
$\mathbf{p}_i$	The embedding of user u_i
$\mathbf{e}_r$	The opinion embedding for the rating level r, such as 5-star rating, $r \in \{1, 2, 3, 4, 5\}$
d	The length of embedding vector
$C(i)$	The set of items which user u_i interacted with
$N(i)$	The set of social friends who user u_i directly connected with
$B(j)$	The set of users who have interacted the item v_j
$\mathbf{h}_i^I$	The item-space user latent factor from item set $C(i)$ of user u_i
$\mathbf{h}_i^S$	The social-space user latent factor from the social friends $N(i)$ of user u_i
$\mathbf{h}_i$	The user latent factor of user u_i, combining from item space $\mathbf{h}_i^I$ and social space $\mathbf{h}_i^S$
$\mathbf{x}_{ia}$	The opinion-aware interaction representation of item v_a for user u_i
$\mathbf{f}_{jt}$	The opinion-aware interaction representation of user u_t for item v_j
$\mathbf{z}_j$	The item latent factor of item v_j
α_{ia}	The item attention of item v_a in contributing to $\mathbf{h}_i^I$
β_{io}	The social attention of neighboring user u_o in contributing to $\mathbf{h}_i^S$
μ_{jt}	The user attention of user u_t in contributing to $\mathbf{z}_j$
r'_{ij}	The predicted rating value of item v_j by user u_i
$\oplus$	The concatenation operator of two vectors
T	The user-user social graph
R	The user-item rating matrix (user-item graph)
W, **b**	The weight and bias in neural network

2.1 Definitions and Notations

Let $U = \{u_1, u_2, ..., u_n\}$ and $V = \{v_1, v_2, ..., v_m\}$ be the sets of users and items respectively, where n is the number of users, and m is the number of items. We assume that $\mathbf{R} \in \mathbb{R}^{n \times m}$ is the user-item rating matrix, which is also called the user-item graph. If u_i gives a rating to v_j, r_{ij} is the rating score, otherwise we employ 0 to represent the unknown rating from u_i to v_j, i.e., $r_{ij} = 0$. The observed rating score r_{ij} can be seen as user u_i's opinion on the item v_j. Let $\mathcal{O} = \left\{\left\langle u_i, v_j \right\rangle | r_{ij} \neq 0\right\}$ be the set of known ratings and $\mathcal{T} = \left\{\left\langle u_i, v_j \right\rangle | r_{ij} = 0\right\}$ be the set of unknown ratings. Let $N(i)$ be the set of users whom u_i directly connected with, $C(i)$ be the set of items which u_i have interacted with, and $B(j)$ be the set of users who have interacted with v_j. In addition, users can establish social relations to each other. We use $\mathrm{T} \in \mathbb{R}^{n \times n}$ to denote the user-user social graph, where $\mathrm{T}_{ij} = 1$ if u_j has a relation to u_i and zero otherwise. Given the user-item graph **R** and social graph T, we aim

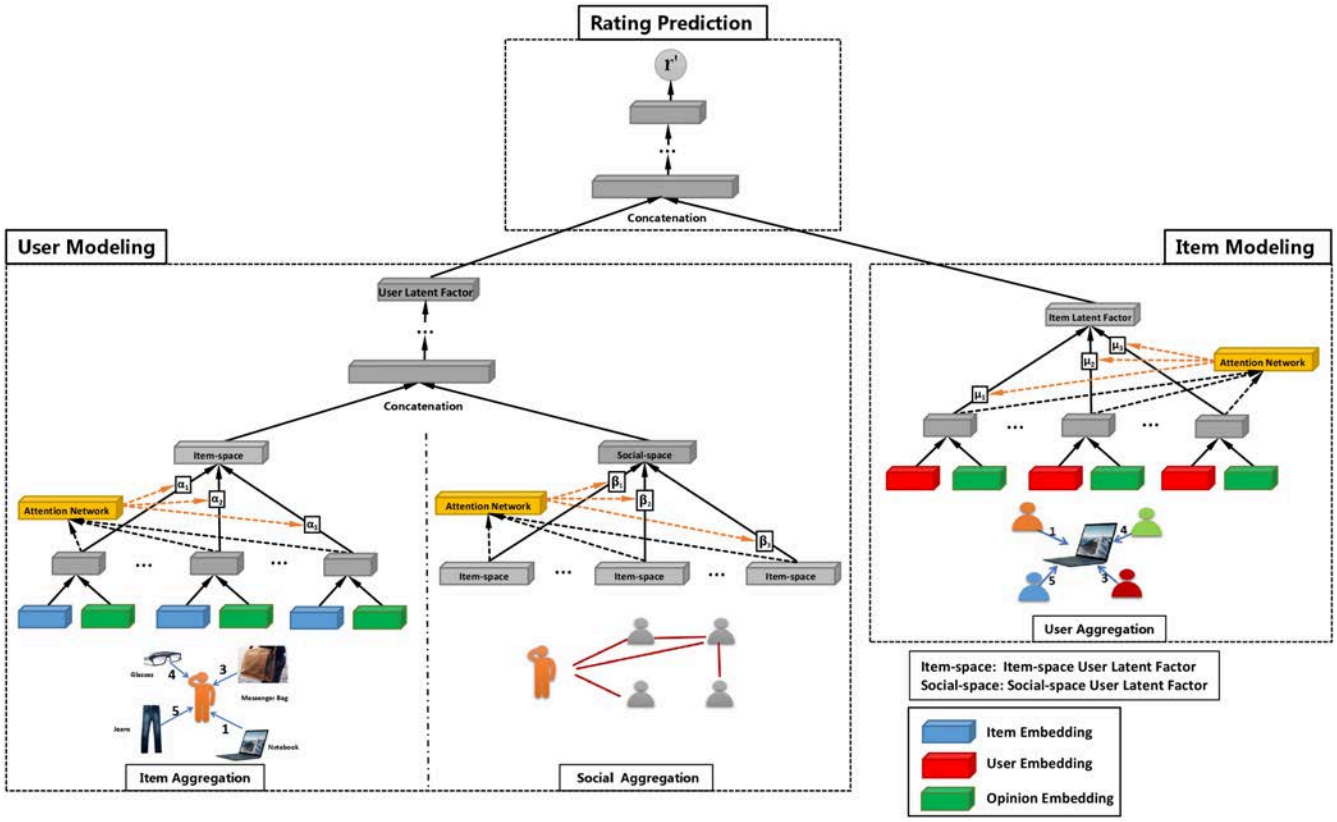


Figure 2: The overall architecture of the proposed model. It contains three major components: user modeling, item modeling, and rating prediction.

to predict the missing rating value in $\mathbf{R}$. Following [11], we use an embedding vector $\mathbf{p}_i \in \mathbb{R}^d$ to denote a user u_i and an embedding vector $\mathbf{q}_j \in \mathbb{R}^d$ to represent an item v_j, where d is the length of embedding vector. More details will be provided about these embedding vectors in the following subsections. The mathematical notations used in this paper are summarized in Table 1.

2.2 An Overview of the Proposed Framework

The architecture of the proposed model is shown in Figure 2. The model consists of three components: user modeling, item modeling, and rating prediction. The first component is user modeling, which is to learn latent factors of users. As data in social recommender systems includes two different graphs, i.e., a social graph and a user-item graph, we are provided with a great opportunity to learn user representations from different perspectives. Therefore, two aggregations are introduced to respectively process these two different graphs. One is item aggregation, which can be utilized to understand users via interactions between users and items in the user-item graph (or item-space). The other is social aggregation, the relationship between users in the social graph, which can help model users from the social perspective (or social-space). Then, it is intuitive to obtain user latent factors by combining information from both item space and social space. The second component is item modeling, which is to learn latent factors of items. In order to consider both interactions and opinions in the user-item graph, we introduce user aggregation, which is to aggregate users' opinions in item modeling. The third component is to learn model parameters via prediction by integrating user and item modeling components. Next, we will detail each model component.

2.3 User Modeling

User modeling aims to learn user latent factors, denoted as $\mathbf{h}_i \in \mathbb{R}^d$ for user u_i. The challenge is how to inherently combine the user-item graph and social graph. To address this challenge, we first use two types of aggregation to learn factors from two graphs, as shown in the left part in Figure 2. The first aggregation, denoted as item aggregation, is utilized to learn item-space user latent factor $\mathbf{h}_i^I \in \mathbb{R}^d$ from the user-item graph. The second aggregation is social aggregation where social-space user latent factor $\mathbf{h}_i^S \in \mathbb{R}^d$ is learned from the social graph. Then, these two factors are combined together to form the final user latent factors $\mathbf{h}_i$. Next, we will introduce item aggregation, social aggregation and how to combine user latent factors from both item-space and social-space.

Item Aggregation. As user-item graph contains not only interactions between users and items but also users' opinions (or rating scores) on items, we provide a principled approach to jointly capture interactions and opinions in the user-item graph for

learning item-space user latent factors $\mathbf{h}_i^I$, which is used to model user latent factor via interactions in the user-item graph.

The purpose of item aggregation is to learn item-space user latent factor $\mathbf{h}_i^I$ by considering items a user u_i has interacted with and users' opinions on these items. To mathematically represent this aggregation, we use the following function as:

$$\mathbf{h}_i^I = \sigma(\mathbf{W} \cdot Aggre_{items}(\{\mathbf{x}_{ia}, \forall a \in C(i)\}) + \mathbf{b}) \tag{1}$$

where $C(i)$ is the set of items user u_i has interacted with (or u_i's neighbors in the user-item graph), $\mathbf{x}_{ia}$ is a representation vector to denote opinion-aware interaction between u_i and an item v_a, and $Aggre_{items}$ is the items aggregation function. In addition, σ denotes non-linear activation function (i.e., a rectified linear unit), and $\mathbf{W}$ and $\mathbf{b}$ are the weight and bias of a neural network. Next we will discuss how to define opinion-aware interaction representation $\mathbf{x}_{ia}$ and the aggregation function $Aggre_{items}$.

A user can express his/her opinions (or rating scores), denoted as r, to items during user-item interactions. These opinions on items can capture users' preferences on items, which can help model item-space user latent factors. To model opinions, for each type of opinions r, we introduce an opinion embedding vector $\mathbf{e}_r \in \mathbb{R}^d$ that denotes each opinion r as a dense vector representation. For example, in a 5-star rating system, for each $r \in \{1, 2, 3, 4, 5\}$, we introduce an embedding vector $\mathbf{e}_r$. For an interaction between user u_i and item v_a with opinion r, we model opinion-aware interaction representation $\mathbf{x}_{ia}$ as a combination of item embedding $\mathbf{q}_a$ and opinion embedding $\mathbf{e}_r$ via a Multi-Layer Perceptron (MLP). It can be denoted as g_v to fuse the interaction information with the opinion information as shown in Figure 2. The MLP takes the concatenation of item embedding $\mathbf{q}_a$ and its opinion embedding $\mathbf{e}_r$ as input. The output of MLP is the opinion-aware representation of the interaction between u_i and v_a, $\mathbf{x}_{ia}$, as follows:

$$\mathbf{x}_{ia} = g_v([\mathbf{q}_a \oplus \mathbf{e}_r]) \tag{2}$$

where $\oplus$ denotes the concatenation operation between two vectors.

One popular aggregation function for $Aggre_{items}$ is the mean operator where we take the element-wise mean of the vectors in $\{\mathbf{x}_{ia}, \forall a \in C(i)\}$. This mean-based aggregator is a linear approximation of a localized spectral convolution [15], as the following function:

$$\mathbf{h}_i^I = \sigma(\mathbf{W} \cdot \left\{ \sum_{a \in C(i)} \alpha_i \mathbf{x}_{ia} \right\} + \mathbf{b}) \tag{3}$$

where α_i is fixed to $\frac{1}{|C(i)|}$ for all items in the mean-based aggregator. It assumes that all interactions contribute equally to understand the user u_i. However, this may not be optimal, due to the fact that the influence of interactions on users may vary dramatically. Hence, we should allow interactions to contribute differently to a user's latent factor by assigning each interaction a weight.

To alleviate the limitation of mean-based aggregator, inspired by attention mechanisms [3, 38], an intuitive solution is to tweak α_i to be aware of the target user u_i, i.e., assigning an individualized weight for each (v_a, u_i) pair,

$$\mathbf{h}_i^I = \sigma(\mathbf{W} \cdot \left\{ \sum_{a \in C(i)} \alpha_{ia} \mathbf{x}_{ia} \right\} + \mathbf{b}) \tag{4}$$

where α_{ia} denotes the attention weight of the interaction with v_a in contributing to user u_i's item-space latent factor when characterizing user u_i's preference from the interaction history $C(i)$. Specially, we parameterize the *item attention* α_{ia} with a two-layer neural network, which we call as the *attention network*. The input to the attention network is the opinion-aware representation $\mathbf{x}_{ia}$ of the interaction and the target user u_i's embedding $\mathbf{p}_i$. Formally, the attention network is defined as,

$$\alpha_{ia}^* = \mathbf{w}_2^T \cdot \sigma(\mathbf{W}_1 \cdot [\mathbf{x}_{ia} \oplus \mathbf{p}_i] + \mathbf{b}_1) + b_2 \tag{5}$$

The final attention weights are obtained by normalizing the above attentive scores using Softmax function, which can be interpreted as the contribution of the interaction to the item-space user latent factor of user u_i as:

$$\alpha_{ia} = \frac{exp(\alpha_{ia}^*)}{\sum_{a \in C(i)} exp(\alpha_{ia}^*)} \tag{6}$$

Social Aggregation. Due to the social correlation theories [20, 21], a user's preference is similar to or influenced by his/her directly connected social friends. We should incorporate social information to further model user latent factors. Meanwhile, tie strengths between users can further influence users' behaviors from the social graph. In other words, the learning of social-space user latent factors should consider heterogeneous strengths of social relations. Therefore, we introduce an attention mechanism to select social friends that are representative to characterize users social information and then aggregate their information.

In order to represent user latent factors from this social perspective, we propose social-space user latent factors, which is to aggregate the item-space user latent factors of neighboring users from the social graph. Specially, the social-space user latent factor of u_i, $\mathbf{h}_i^S$, is to aggregate the item-space user latent factors of users in u_i's neighbors $N(i)$, as the follows:

$$\mathbf{h}_i^S = \sigma(\mathbf{W} \cdot Aggre_{neigbhors}(\left\{\mathbf{h}_o^I, \forall o \in N(i)\right\}) + \mathbf{b}) \tag{7}$$

where $Aggre_{neigbhors}$ denotes the aggregation function on user's neighbors.

One natural aggregation function for $Aggre_{neigbhors}$ is also the mean operator which take the element-wise mean of the vectors in $\left\{\mathbf{h}_o^I, \forall o \in N(i)\right\}$, as the following function:

$$\mathbf{h}_i^S = \sigma(\mathbf{W} \cdot \left\{ \sum_{o \in N(i)} \beta_i \mathbf{h}_o^I \right\} + \mathbf{b}) \tag{8}$$

where β_i is fixed to $\frac{1}{|N(i)|}$ for all neighbors for the mean-based aggregator. It assumes that all neighbors contribute equally to the representation of user u_i. However, as mentioned before, strong and weak ties are mixed together in a social network, and users are likely to share more similar tastes with strong ties than weak ties. Thus, we perform an attention mechanism with a two-layer neural network to extract these users that are important to influence u_i, and model their tie strengths, by relating *social attention* β_{io} with

$\mathbf{h}_o^I$ and the target user embedding $\mathbf{p}_i$, as below,

$$\mathbf{h}_i^S = \sigma(\mathbf{W} \cdot \left\{ \sum_{o \in N(i)} \beta_{io} \mathbf{h}_o^I \right\} + \mathbf{b}) \tag{9}$$

$$\beta_{io}^* = \mathbf{w}_2^T \cdot \sigma(\mathbf{W}_1 \cdot [\mathbf{h}_o^I \oplus \mathbf{p}_i] + \mathbf{b}_1) + b_2 \tag{10}$$

$$\beta_{io} = \frac{exp(\beta_{io}^*)}{\sum_{o \in N(i)} exp(\beta_{io}^*)} \tag{11}$$

where the β_{io} can be seen as the strengths between users.

Learning User Latent Factor. In order to learn better user latent factors, item-space user latent factors and social-space user latent factors are needed to be considered together, since the social graph and the user-item graph provide information about users from different perspectives. We propose to combine these two latent factors to the final user latent factor via a standard MLP where the item-space user latent factor $\mathbf{h}_i^I$ and the social-space user latent factor $\mathbf{h}_i^S$ are concatenated before feeding into MLP. Formally, the user latent factor $\mathbf{h}_i$ is defined as,

$$\mathbf{c}_1 = \left[\mathbf{h}_i^I \oplus \mathbf{h}_i^S \right] \tag{12}$$

$$\mathbf{c}_2 = \sigma(\mathbf{W}_2 \cdot \mathbf{c}_1 + \mathbf{b}_2) \tag{13}$$

$$\cdots$$

$$\mathbf{h}_i = \sigma(\mathbf{W}_l \cdot \mathbf{c}_{l-1} + \mathbf{b}_l) \tag{14}$$

where l is the index of a hidden layer.

2.4 Item Modeling

As shown in the right part of Figure 2, item modeling is used to learn item latent factor, denoted as $\mathbf{z}_j$, for the item v_j by user aggregation. Items are associated with the user-item graph, which contains interactions as well as user's opinions. Therefore, interactions and opinions in the user-item graph should be jointly captured to further learn item latent factors.

User Aggregation. Likewise, we use a similar method as learning item-space user latent factors via item aggregation. For each item v_j, we need to aggregate information from the set of users who have interacted with v_j, denoted as $B(j)$.

Even for the same item, users might express different opinions during user-item interactions. These opinions from different users can capture the characteristics of the same item in different ways provided by users, which can help model item latent factors. For an interaction from u_t to v_j with opinion r, we introduce an opinion-aware interaction user representation $\mathbf{f}_{jt}$, which is obtained from the basic user embedding $\mathbf{p}_t$ and opinion embedding $\mathbf{e}_r$ via a MLP, denoted as g_u. g_u is to fuse the interaction information with the opinion information, as shown in Figure 2:

$$\mathbf{f}_{jt} = g_u([\mathbf{p}_t \oplus \mathbf{e}_r]) \tag{15}$$

Then, to learn item latent factor $\mathbf{z}_j$, we also propose to aggregate opinion-aware interaction representation of users in $B(j)$ for item v_j. The users aggregation function is denoted as $Aggre_{users}$, which is to aggregate opinion-aware interaction representation of users in $\left\{ \mathbf{f}_{jt}, \forall t \in B(j) \right\}$ as:

$$\mathbf{z}_j = \sigma(\mathbf{W} \cdot Aggre_{users}(\left\{ \mathbf{f}_{jt}, \forall t \in B(j) \right\}) + \mathbf{b}) \tag{16}$$

In addition, we introduce an attention mechanism to differentiate the importance weight μ_{jt} of users with a two-layer neural attention network, taking $\mathbf{f}_{jt}$ and $\mathbf{q}_j$ as the input,

$$\mathbf{z}_j = \sigma(\mathbf{W} \cdot \left\{ \sum_{t \in B(j)} \mu_{jt} \mathbf{f}_{jt} \right\} + \mathbf{b}) \tag{17}$$

$$\mu_{jt}^* = \mathbf{w}_2^T \cdot \sigma(\mathbf{W}_1 \cdot [\mathbf{f}_{jt} \oplus \mathbf{q}_j] + \mathbf{b}_1) + b_2 \tag{18}$$

$$\mu_{jt} = \frac{exp(\mu_{jt}^*)}{\sum_{t \in B_{(j)}} exp(\mu_{jt}^*)} \tag{19}$$

This *user attention* μ_{jt} is to capture heterogeneous influence from user-item interactions on learning item latent factor.

2.5 Rating Prediction

In this subsection, we will design recommendation tasks to learn model parameters. There are various recommendation tasks such as item ranking and rating prediction. In this work, we apply the proposed GraphRec model for the recommendation task of rating prediction. With the latent factors of users and items (i.e., $\mathbf{h}_i$ and $\mathbf{z}_j$), we can first concatenate them $\left[\mathbf{h}_i \oplus \mathbf{z}_j \right]$ and then feed it into MLP for rating prediction as:

$$\mathbf{g}_1 = \left[\mathbf{h}_i \oplus \mathbf{z}_j \right] \tag{20}$$

$$\mathbf{g}_2 = \sigma(\mathbf{W}_2 \cdot \mathbf{g}_1 + \mathbf{b}_2) \tag{21}$$

$$\cdots$$

$$\mathbf{g}_{l-1} = \sigma(\mathbf{W}_l \cdot \mathbf{g}_{l-1} + \mathbf{b}_l) \tag{22}$$

$$r'_{ij} = \mathbf{w}^T \cdot \mathbf{g}_{l-1} \tag{23}$$

where l is the index of a hidden layer, and r'_{ij} is the predicted rating from u_i to v_j.

2.6 Model Training

To estimate model parameters of GraphRec, we need to specify an objective function to optimize. Since the task we focus on in this work is rating prediction, a commonly used objective function is formulated as,

$$Loss = \frac{1}{2\,|O|} \sum_{i,j \in O} (r'_{ij} - r_{ij})^2 \tag{24}$$

where $|O|$ is the number of observed ratings , and r_{ij} is the ground truth rating assigned by the user i on the item j.

To optimize the objective function, we adopt the RMSprop [31] as the optimizer in our implementation, rather than the vanilla SGD. At each time, it randomly selects a training instance and updates each model parameter towards the direction of its negative gradient. There are three embedding in our model, including item embedding $\mathbf{q}_j$, user embedding $\mathbf{p}_i$, and opinion embedding $\mathbf{e}_r$. They are randomly initialized and jointly learned during the training stage. We do not use one-hot vectors to represent each user and item, since the raw features are very large and highly sparse. By embedding high-dimensional sparse features into a low-dimensional latent space, the model can be easy to train [11]. Opinion embedding matrix $\mathbf{e}$ depends on the rating scale of the system. For example, for a 5-star rating system, opinion embedding

matrix **e** contains 5 different embedding vectors to denote scores in {1, 2, 3, 4, 5}. Overfitting is a perpetual problem in optimizing deep neural network models. To alleviate this issue, the dropout strategy [26] has been applied to our model. The idea of dropout is to randomly drop some neurons during the training process. When updating parameters, only part of them will be updated. Moreover, as dropout is disabled during testing, the whole network is used for prediction.

3 EXPERIMENT

3.1 Experimental Settings

3.1.1 Datasets. In our experiments, we choose two representative datasets Ciao and Epinions[1], which are taken from popular social networking websites Ciao (http://www.ciao.co.uk) and Epinions (www.epinions.com). Each social networking service allows users to rate items, browse/write reviews, and add friends to their 'Circle of Trust'. Hence, they provide a large amount of rating information and social information. The ratings scale is from 1 to 5. We randomly initialize opinion embedding with 5 different embedding vectors based on 5 scores in {1, 2, 3, 4, 5}. The statistics of these two datasets are presented in Table 2.

Table 2: Statistics of the datasets

Dataset	Ciao	Epinions
# of Users	7,317	18,088
# of Items	10,4975	261,649
# of Ratings	283,319	764,352
# of Density (Ratings)	0.0368%	0.0161%
# of Social Connections	111,781	355,813
# of Density (Social Relations)	0.2087%	0.1087%

3.1.2 Evaluation Metrics. In order to evaluate the quality of the recommendation algorithms, two popular metrics are adopted to evaluate the predictive accuracy, namely Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) [34]. Smaller values of MAE and RMSE indicate better predictive accuracy. Note that small improvement in RMSE or MAE terms can have a significant impact on the quality of the top-few recommendations [16].

3.1.3 Baselines. To evaluate the performance, we compared our GraphRec with three groups of methods including traditional recommender systems, traditional social recommender systems, and deep neural network based recommender systems. For each group, we select representative baselines and below we will detail them.

- **PMF** [24]: **P**robabilistic **M**atrix **F**actorization utilizes user-item rating matrix only and models latent factors of users and items by Gaussian distributions.
- **SoRec** [17]: **So**cial **Rec**ommendation performs co-factorization on the user-item rating matrix and user-user social relations matrix.
- **SoReg** [18]: **So**cial **Reg**ularization models social network information as regularization terms to constrain the matrix factorization framework.
- **SocialMF** [13]: It considers the trust information and propagation of trust information into the matrix factorization model for recommender systems.
- **TrustMF** [37]: This method adopts matrix factorization technique that maps users into two low-dimensional spaces: truster space and trustee space, by factorizing trust networks according to the directional property of trust.
- **NeuMF** [11]: This method is a state-of-the-art matrix factorization model with neural network architecture. The original implementation is for recommendation ranking task and we adjust its loss to the squared loss for rating prediction.
- **DeepSoR** [8]: This model employs a deep neural network to learn representations of each user from social relations, and to integrate into probabilistic matrix factorization for rating prediction.
- **GCMC+SN** [1]: This model is a state-of-the-art recommender system with graph neural network architecture. In order to incorporate social network information into GCMC, we utilize the *node2vec* [9] to generate user embedding as user side information, instead of using the raw feature social connections ($T \in \mathbb{R}^{n \times n}$) directly. The reason is that the raw feature input vectors is highly sparse and high-dimensional. Using the network embedding techniques can help compress the raw input feature vector to a low-dimensional and dense vector, then the model can be easy to train.

PMF and NeuMF are pure collaborative filtering model without social network information for rating prediction, while the others are social recommendation. Besides, we compared GraphRec with two state-of-the-art neural network based social recommender systems, i.e., DeepSoR and GCMC+SN.

3.1.4 Parameter Settings. We implemented our proposed method on the basis of Pytorch[2], a well-known Python library for neural networks. For each dataset, we used $x\%$ as a training set to learning parameters, $(1-x\%)/2$ as a validation set to tune hyper-parameters, and $(1-x\%)/2$ as a testing set for the final performance comparison, where x was varied as $\{80\%, 60\%\}$. For the embedding size d, we tested the value of [8, 16, 32, 64, 128, 256]. The batch size and learning rate was searched in [32, 64, 128, 512] and [0.0005, 0.001, 0.005, 0.01, 0.05, 0.1], respectively. Moreover, we empirically set the size of the hidden layer the same as the embedding size and the activation function as ReLU. Without special mention, we employed three hidden layers for all the neural components. The early stopping strategy was performed, where we stopped training if the RMSE on validation set increased for 5 successive epochs. For all neural network methods, we randomly initialized model parameters with a Gaussian distribution, where the mean and standard deviation is 0 and 0.1, respectively. The parameters for the baseline algorithms were initialized as in the corresponding papers and were then carefully tuned to achieve optimal performance.

3.2 Performance Comparison of Recommender Systems

We first compare the recommendation performance of all methods. Table 3 shows the overall rating prediction error $w.r.t.$ RMSE and

[1] http://www.cse.msu.edu/~tangjili/index.html

[2] https://pytorch.org/

Table 3: Performance comparison of different recommender systems

Training	Metrics	Algorithms								
		PMF	SoRec	SoReg	SocialMF	TrustMF	NeuMF	DeepSoR	GCMC+SN	**GraphRec**
Ciao (60%)	MAE	0.952	0.8489	0.8987	0.8353	0.7681	0.8251	0.7813	0.7697	0.7540
	RMSE	1.1967	1.0738	1.0947	1.0592	1.0543	1.0824	1.0437	1.0221	1.0093
Ciao (80%)	MAE	0.9021	0.8410	0.8611	0.8270	0.7690	0.8062	0.7739	0.7526	0.7387
	RMSE	1.1238	1.0652	1.0848	1.0501	1.0479	1.0617	1.0316	0.9931	0.9794
Epinions (60%)	MAE	1.0211	0.9086	0.9412	0.8965	0.8550	0.9097	0.8520	0.8602	0.8441
	RMSE	1.2739	1.1563	1.1936	1.1410	1.1505	1.1645	1.1135	1.1004	1.0878
Epinions (80%)	MAE	0.9952	0.8961	0.9119	0.8837	0.8410	0.9072	0.8383	0.8590	0.8168
	RMSE	1.2128	1.1437	1.1703	1.1328	1.1395	1.1476	1.0972	1.0711	1.0631

MAE among the recommendation methods on Ciao and Epinions datasets. We have the following main findings:

- SoRec, SoReg, SocialMF, and TrustMF always outperform PMF. All of these methods are based on matrix factorization. SoRec, SoReg, SocialMF, and TrustMF leverage both the rating and social network information; while PMF only uses the rating information. These results support that social network information is complementary to rating information for recommendations.
- NeuMF obtains much better performance than PMF. Both methods only utilize the rating information. However, NeuMF is based on neural network architecture, which suggests the power of neural network models in recommender systems.
- DeepSoR and GCMC+SN perform better than SoRec, SoReg, SocialMF, and TrustMF. All of them take advantage of both rating and social network information. However, DeepSoR and GCMC+SN are based on neural network architectures, which further indicate the power of neural network models in recommendations.
- Among baselines, GCMC+SN shows quite strong performance. It implies that the GNNs are powerful in representation learning for graph data, since it naturally integrates the node information as well as topological structure.
- Our method GraphRec consistently outperforms all the baseline methods. Compared to DeepSoR and GCMC+SN, our model provides advanced model components to integrate rating and social network information. In addition, our model provides a way to consider both interactions and opinions in the user-item graph. We will provide further investigations to better understand the contributions of model components to the proposed framework in the following subsection.

To sum up, the comparison results suggest (1) social network information is helpful for recommendations; (2) neural network models can boost recommendation performance and (3) the proposed framework outperforms representative baselines.

3.3 Model Analysis

In this subsection, we study the impact of model components and model hyper-parameters.

3.3.1 Effect of Social Network and User Opinions. In the last subsection, we have demonstrated the effectiveness of the proposed framework. The proposed framework provides model components to (1) integrate social network information and (2) incorporate users' opinions about the interactions with items. To understand the working of GraphRec, we compare GraphRec with its two variants: GraphRec-SN, and GraphRec-Opinion. These two variants are defined in the following:

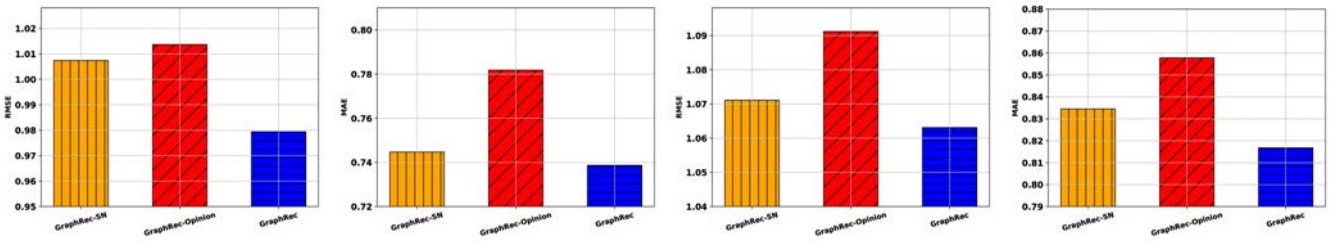
- GraphRec-SN: The social network information of GraphRec is removed . This variant only uses the item-space user latent factor $\mathbf{h}_i^I$ to represent user latent factors $\mathbf{h}_i$; while ignoring the social-space user latent factors $\mathbf{h}_i^S$.
- GraphRec-Opinion: For learning item-space user latent factor and item latent factor, the opinion embedding is removed during learning $\mathbf{x}_{ia}$ and $\mathbf{f}_{jt}$. This variant ignores the users' opinion on the user-item interactions.

The performance of GraphRec and its variants on Ciao and Epinions are given in Figure 3. From the results, we have the following findings:

- **Social Network Information.** We now focus on analyzing the effectiveness of social network information. GraphRec-SN performs worse than GraphRec. It verifies that social network information is important to learn user latent factors and boost the recommendation performance.
- **Opinions in Interaction.** We can see that without opinion information, the performance of rating prediction is deteriorated significantly. For example, on average, the relative reduction on Ciao and Epinions is 3.50% and 2.64% on RMSE metric, and 5.84% and 5.02% on MAE metric, respectively. It justifies our assumption that opinions on user-item interactions have informative information that can help to learn user or item latent factors and improve the performance of recommendation.

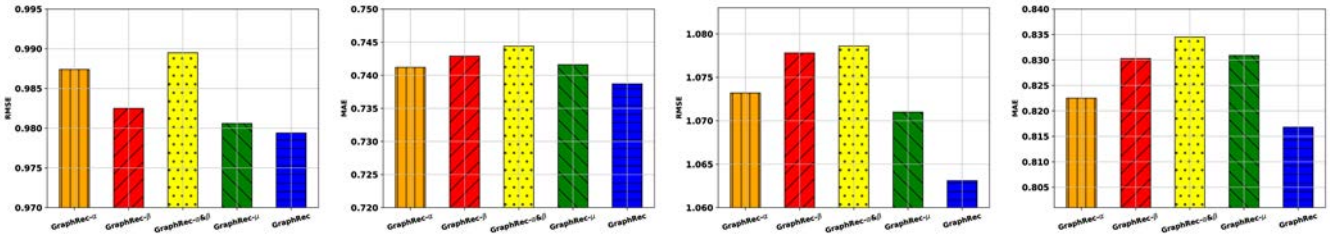
3.3.2 Effect of Attention Mechanisms. To get a better understanding of the proposed GraphRec model, we further evaluate the key components of GraphRec - *Attention mechanisms*. There are three different attention mechanisms during aggregation, including item attention α, social attention β, and user attention μ. We compare GraphRec with its four variants: GraphRec-α, GraphRec-β, GraphRec-$\alpha\&\beta$, and GraphRec-μ. These four variants are defined in the following:

- GraphRec-α: The item attention α of GraphRec is eliminated during aggregating the opinion-aware interaction representation of items. This variant employs the mean-based aggregation function on item aggregation for modeling item-space user latent factors.



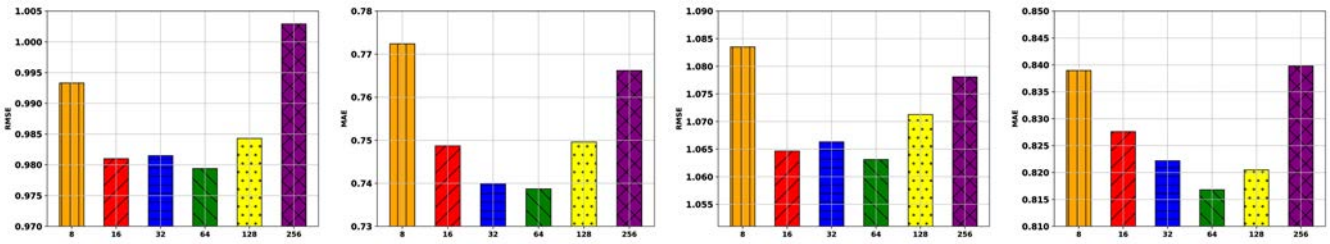
(a) Ciao-RMSE (b) Ciao-MAE (c) Epinions-RMSE (d) Epinions-MAE

Figure 3: Effect of social network and user opinions on Ciao and Epinions datasets.



(a) Ciao-RMSE (b) Ciao-MAE (c) Epinions-RMSE (d) Epinions-MAE

Figure 4: Effect of attention mechanisms on Ciao and Epinions datasets.



(a) Ciao-RMSE (b) Ciao-MAE (c) Epinions-RMSE (d) Epinions-MAE

Figure 5: Effect of embedding size on Ciao and Epinions datasets.

- GraphRec-β: The social attention α is to model users' tie strengths. The social attention α of GraphRec in this variant is eliminated during aggregating user's neighbors. This variant employs the mean-based aggregation function on social aggregation for modeling social-space user latent factors.
- GraphRec-$\alpha\&\beta$: This variant eliminates two attention mechanisms (item attention α and social attention β) on item aggregation and social aggregation for modeling user latent factors.
- GraphRec-μ: The user attention μ of GraphRec is eliminated during aggregating opinion-aware interaction user representation. This variant employs the mean-based aggregation function on user aggregation for modeling item latent factors.

The results of different attention mechanisms on GraphRec are shown in Figure 4. From the results, we have the following findings,

- Not all interacted items (purchased history) of one user contribute equally to the item-space user latent factor, and not all interacted users (buyers) have the same importance to learning item latent factor. Based on these assumptions, our model considers these difference among users and items by using two different attention mechanisms (α and μ). From the results, we can observe that GraphRec-α and GraphRec-μ obtain worse performance than GraphRec. These results demonstrate the benefits of the attention mechanisms on item aggregation and user aggregation.
- As mentioned before, users are likely to share more similar tastes with strong ties than weak ties. The attention mechanism β at social aggregation considers heterogeneous

strengths of social relations. When the attention mechanism β is removed, the performance of GraphRec-β is dropped significantly. It justifies our assumption that during social aggregation, different social friends should have different influence for learning social-space user latent factor. It's important to distinguish social relations with heterogeneous strengths.

To sum up, GraphRec can capture the heterogeneity in aggregation operations of the proposed framework via attention mechanisms, which can boost the recommendation performance.

3.3.3 Effect of Embedding Size. In this subsection, to analyze the effect of embedding size of user embedding $\mathbf{p}$, item embedding $\mathbf{q}$, and opinion embedding $\mathbf{e}$, on the performance of our model.

Figure 5 presents the performance comparison $w.r.t.$ the length of embedding of our proposed model on Ciao and Epinions datasets. In general, with the increase of the embedding size, the performance first increases and then decreases. When increasing the embedding size from 8 to 64 can improve the performance significantly. However, with the embedding size of 256, GraphRec degrades the performance. It demonstrates that using a large number of the embedding size has powerful representation. Nevertheless, if the length of embedding is too large, the complexity of our model will significantly increase. Therefore, we need to find a proper length of embedding in order to balance the trade-off between the performance and the complexity.

4 RELATED WORK

In this section, we briefly review some related work about social recommendation, deep neural network techniques employed for recommendation, and the advanced graph neural networks.

Exploiting social relations for recommendations has attracted significant attention in recent years [27, 28, 37]. One common assumption about these models is that a user's preference is similar to or influenced by the people around him/her (nearest neighbours), which can be proven by social correlation theories [20, 21]. Along with this line, SoRec [17] proposed a co-factorization method, which shares a common latent user-feature matrix factorized by ratings and by social relations. TrustMF [37] modeled mutual influence between users, and mapped users into two low-dimensional spaces: truster space and trustee space, by factorizing social trust networks. SoDimRec [30] first adopted a community detection algorithm to partition users into several clusters, and then exploited the heterogeneity of social relations and weak dependency connections for recommendation. Comprehensive overviews on social recommender systems can be found in surveys [29].

In recent years, deep neural network models had a great impact on learning effective feature representations in various fields, such as speech recognition [12], Computer Vision (CV) [14] and Natural Language Processing (NLP) [4]. Some recent efforts have applied deep neural networks to recommendation tasks and shown promising results [41], but most of them used deep neural networks to model audio features of music [32], textual description of items [3, 33], and visual content of images [40]. Besides, NeuMF [11] presented a Neural Collaborative Filtering framework to learn the non-linear interactions between users and items.

However, the application of deep neural network in social recommender systems is rare until very recently. In particular, NSCR [35] extended the NeuMF [11] model to cross-domain social recommendations, i.e., recommending items of information domains to potential users of social networks, and presented a neural social collaborative ranking recommender system. However, the limitation is NSCR requires users with one or more social networks accounts (e.g., Facebook, Twitter, Instagram), which limits the data collections and its applications in practice. SMR-MNRL [42] developed social-aware movie recommendation in social media from the viewpoint of learning a multimodal heterogeneous network representation for ranking. They exploited the recurrent neural network and convolutional neural network to learn the representation of movies' textual description and poster image, and adopted a random-walk based learning method into multimodal neural networks. In all these works [35] [42], they addressed the task of cross-domain social recommendations for ranking metric, which is different from traditional social recommender systems.

Most related to our task with neural networks includes DLMF [6] and DeepSoR [8]. DLMF [6] used auto-encoder on ratings to learn representation for initializing an existing matrix factorization. A two-phase trust-aware recommendation process is proposed to utilize deep neural networks in matrix factorization's initialization and to synthesize the user's interests and their trust friends' interests together with the impact of community effect based on matrix factorization for recommendations. DeepSoR [8] integrated neural networks for user's social relations into probabilistic matrix factorization. They first represented users using pre-trained node embedding technique, and further exploited k-nearest neighbors to bridge user embedding features and neural network.

More recently, Graph Neural Networks (GNNs) have been proven to be capable of learning on graph structure data [2, 5, 7, 15, 25]. In the task of recommender systems, the user-item interaction contains the ratings on items by users, which is a typical graph data. Therefore, GNNs have been proposed to solve the recommendation problem [1, 22, 39]. sRMGCNN [22] adopted GNNs to extract graph embeddings for users and items, and then combined with recurrent neural network to perform a diffusion process. GCMC [1] proposed a graph auto-encoder framework, which produced latent features of users and items through a form of differentiable message passing on the user-item graph. PinSage [39] proposed a random-walk graph neural network to learn embedding for nodes in web-scale graphs. Despite the compelling success achieved by previous work, little attention has been paid to social recommendation with GNNs. In this paper, we propose a graph neural network for social recommendation to fill this gap.

5 CONCLUSION AND FUTURE WORK

We have presented a Graph Network model (GraphRec) to model social recommendation for rating prediction. Particularly, we provide a principled approach to jointly capture interactions and opinions in the user-item graph. Our experiments reveal that the opinion information plays a crucial role in the improvement of our model performance. In addition, our GraphRec can differentiate the ties strengths by considering heterogeneous strengths of social

relations. Experimental results on two real-world datasets show that GraphRec can outperform state-of-the-art baselines.

Currently we only incorporate the social graph into recommendation, while many real-world industries are associated rich other side information on users as well as items. For example, users and items are associated with rich attributes. Therefore, exploring graph neural networks for recommendation with attributes would be an interesting future direction. Beyond that, now we consider both rating and social information static. However, rating and social information are naturally dynamic. Hence, we will consider building dynamic graph neural networks for social recommendations with dynamic.

ACKNOWLEDGMENTS

The work described in this paper has been supported, in part, by a general research fund from the Hong Kong Research Grants Council (project PolyU 1121417/17E), and an internal research grant from the Hong Kong Polytechnic University (project 1.9B0V). Yao Ma and Jiliang Tang are supported by the National Science Foundation (NSF) under grant numbers IIS-1714741, IIS-1715940 and CNS-1815636, and a grant from Criteo Faculty Research Award.

REFERENCES

[1] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).
[2] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine* 34, 4 (2017), 18–42.
[3] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural Attentional Rating Regression with Review-level Explanations. In *Proceedings of the 27th International Conference on World Wide Web*. 1583–1592.
[4] Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *ACM SIGKDD Explorations Newsletter* 19, 2 (2017), 25–35.
[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*. 3844–3852.
[6] Shuiguang Deng, Longtao Huang, Guandong Xu, Xindong Wu, and Zhaohui Wu. 2017. On deep learning for trust-aware recommendations in social networks. *IEEE transactions on neural networks and learning systems* 28, 5 (2017), 1164–1177.
[7] Tyler Derr, Yao Ma, and Jiliang Tang. 2018. Signed Graph Convolutional Networks. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 929–934.
[8] Wenqi Fan, Qing Li, and Min Cheng. 2018. Deep Modeling of Social Relations for Recommendation. In *AAAI*.
[9] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 855–864.
[10] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.
[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web, WWW*. 173–182.
[12] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* 29, 6 (2012), 82–97.
[13] Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 135–142.
[14] Hamid Karimi, Jiliang Tang, and Yanen Li. 2018. Toward End-to-End Deception Detection in Videos. In *2018 IEEE Big Data*. IEEE, 1278–1283.
[15] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.
[16] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.
[17] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. 2008. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and Knowledge Management*. ACM, 931–940.
[18] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web Search and Data Mining*. ACM, 287–296.
[19] Yao Ma, Suhang Wang, Charu C. Aggarwal, Dawei Yin, and Jiliang Tang. 2019. Multi-dimensional Graph Convolutional Networks. In *Proceedings of the 2019 SIAM International Conference on Data Mining(SDM)*.
[20] Peter V Marsden and Noah E Friedkin. 1993. Network studies of social influence. *Sociological Methods & Research* 22, 1 (1993), 127–151.
[21] Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology* 27, 1 (2001), 415–444.
[22] Federico Monti, Michael Bronstein, and Xavier Bresson. 2017. Geometric matrix completion with recurrent multi-graph neural networks. In *Advances in Neural Information Processing Systems*. 3700–3710.
[23] Paul Resnick and Hal R Varian. 1997. Recommender systems. *Commun. ACM* 40, 3 (1997), 56–58.
[24] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *21th Conference on Neural Information Processing Systems*, Vol. 1. 2–1.
[25] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine* 30, 3 (2013), 83–98.
[26] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.
[27] Jiliang Tang, Charu Aggarwal, and Huan Liu. 2016. Recommendations in signed social networks. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 31–40.
[28] Jiliang Tang, Xia Hu, Huiji Gao, and Huan Liu. 2013. Exploiting local and global social context for recommendation.. In *IJCAI*, Vol. 13. 2712–2718.
[29] Jiliang Tang, Xia Hu, and Huan Liu. 2013. Social recommendation: a review. *Social Network Analysis and Mining* 3, 4 (2013), 1113–1133.
[30] Jiliang Tang, Suhang Wang, Xia Hu, Dawei Yin, Yingzhou Bi, Yi Chang, and Huan Liu. 2016. Recommendation with Social Dimensions. In *AAAI*. 251–257.
[31] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-RMSProp, COURSERA: Neural networks for machine learning. *University of Toronto, Technical Report* (2012).
[32] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Advances in neural Information Processing Systems*. 2643–2651.
[33] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.
[34] Suhang Wang, Jiliang Tang, Yilin Wang, and Huan Liu. 2018. Exploring Hierarchical Structures for Recommender Systems. *IEEE Transactions on Knowledge and Data Engineering* 30, 6 (2018), 1022–1035.
[35] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item silk road: Recommending items from information domains to social users. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 185–194.
[36] Rongjing Xiang, Jennifer Neville, and Monica Rogati. 2010. Modeling relationship strength in online social networks. In *Proceedings of the 19th international conference on World wide web*. ACM, 981–990.
[37] Bo Yang, Yu Lei, Jiming Liu, and Wenjie Li. 2017. Social collaborative filtering by trust. *IEEE transactions on pattern analysis and machine intelligence* 39, 8 (2017), 1633–1647.
[38] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.
[39] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD '18*. ACM, 974–983.
[40] Lili Zhao, Zhongqi Lu, Sinno Jialin Pan, and Qiang Yang. 2016. Matrix Factorization+ for Movie Recommendation. In *IJCAI*. 3945–3951.
[41] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendations with Negative Feedback via Pairwise Deep Reinforcement Learning. In *KDD'18*. ACM, 1040–1048.
[42] Zhou Zhao, Qifan Yang, Hanqing Lu, Tim Weninger, Deng Cai, Xiaofei He, and Yueting Zhuang. 2018. Social-aware movie recommendation via multimodal network learning. *IEEE Transactions on Multimedia* 20, 2 (2018), 430–440.

Knowledge-Enhanced Ensemble Learning for Word Embeddings

Lanting Fang
Southeast University
NKG, CN
lantingf@outlook.com

Yong Luo
Nanyang Technological University
SG
yluo180@gmail.com

Kaiyu Feng
Nanyang Technological University
SG
kaiyufeng@gmail.com

Kaiqi Zhao
Nanyang Technological University
SG
kzhao002@e.ntu.edu.sg

Aiqun Hu
Southeast University
NKG, CN
aqhu@seu.edu.cn

ABSTRACT

Representing words as embeddings in a continuous vector space has been proven to be successful in improving the performance in many natural language processing (NLP) tasks. Beyond the traditional methods that learn the embeddings from large text corpora, ensemble methods have been proposed to leverage the merits from pre-trained word embeddings as well as external semantic sources. In this paper, we propose a knowledge-enhanced ensemble method to combine both knowledge graphs and pre-trained word embedding models. Specifically, we interpret relations in knowledge graphs as linear translation from one word to another. We also propose a novel weighting scheme to further distinguish edges in the knowledge graph with same type of relation. Extensive experiments demonstrate that our proposed method is up to 20% times better than state-of-the-art in word analogy task and up to 16% times better than state-of-the-art in word similarity task.

KEYWORDS

Word embedding; Knowledge graph; Ensemble model

ACM Reference Format:
Lanting Fang, Yong Luo, Kaiyu Feng, Kaiqi Zhao, and Aiqun Hu. 2019. Knowledge-Enhanced Ensemble Learning for Word Embeddings. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313425

1 INTRODUCTION

Word embeddings are low dimensional distributed representations of words in a real-valued vector space. As they are both compact and informative, word embeddings have proven to be effective in improving the performance in many natural language processing (NLP) tasks, e.g., named entity recognition (NER) [32], POS tagging [26] and information retrieval [30]. Most of the existing methods for learning word embeddings are trained on large unlabeled text corpora and many pre-trained word embeddings are available online. Mikolov et al. [18] learn the continuous bag-of-words (CBOW) and Skip-Gram embeddings from Google News corpus. The training objective of CBOW is to combine the embeddings of context words to predict the target word, while Skip-Gram is to use the embedding of each target word to predict its context words. Pennington et al. [24] learn the GloVe embedding set from the common crawled web data. The training objective of GloVe is to let the dot product of the embeddings of two words approximately equal to the logarithm of the count of their co-occurrences.

As word embeddings learned by different methods often capture different statistics from different corpora and these corpora are usually not available publicly, a natural idea for gathering the merits from multiple pre-trained word embeddings is to learn an ensemble model taking the statistics from them. Muromagi et al. [22] propose to combine several word embedding models into a unified one. They prove that an ensemble model can not only cancels out random noise of individual models but also reinforces co-occurrence patterns in the input models. However, the above models are totally context-based and ignore the common sense relations between two words within the embeddings. As a result, these models are incapable of capturing semantic relatedness of words if they do not appear in similar context.

Several studies [3, 7, 28, 34, 36] have been proposed to incorporate external semantic sources. Knowledge graphs, such as WordNet [19] and ConceptNet [29], are considered as one of the most important semantic sources for improving word embeddings. A knowledge graph is a directed graph and can be represented by a set of (entity, *relation*, entity) tuples. Faruqui et al. [7] propose a "retrofitting" method to leverage the synonym, hypernym, hyponym and paraphrase relations in WordNet to learn the ensemble embeddings. Robert et al. [28] propose a more general method "ConceptNet-Numberbatch (CN)" to support other types of relations in ConceptNet. However, these methods underlook the importance of the relations and consider two words are related if they are connected by any relation in the knowledge graph. For instance, relations *EatenBy* and *Hit* in tuples (apple, *EatenBy*, man) and (apple, *Hit*, man) are viewed as equivalent.

In this paper, we propose a knowledge-enhanced ensemble learning model called *RA-Retrofit* for word embeddings. The proposed model is able to leverage all semantic information provided by knowledge graphs as well as the pre-trained word embedding models. Firstly, to use the semantic information provided by knowledge graph, inspired by the efforts on multi-relational data embedding

Yong Luo is also with the Peking University Shenzhen Graduate School, SZX, CN.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313425

[1], we learn embeddings for both nodes (each node represents a word) and edges (each edge represents the relation between two words) in the knowledge graph. The embedding of an edge is interpreted as linear translation from the source word to the target word. In the previous example, the embedding of "man" can be calculated by simply adding the embeddings of word "apple" and relation *EatenBy*. Different from [1], we propose a novel scheme to assign weights to the edges in the loss function. We compute the weight for each edge based on the relevance between the two words connected by the edge: a larger weight for two close related words. For example, we give higher weights for (Internet, *RelatedTo*, Computer) than for (Multipart, *RelatedTo*, Computer) because "Computer" is more related to "Internet" than "Multipart". Secondly, to preserve the information from both the input models and knowledge graphs in the ensemble word embeddings, we propose a joint learning method that minimizes (1) the training loss of the knowledge graph embeddings at the same time. Additionally, we show that *RA-Retrofit* is more general than several existing embedding methods through a theoretical analysis and (2) the difference between the pre-trained word embeddings and the linear projection of the ensemble word embeddings.

In summary, this paper has the following contributions.

- We propose a knowledge-enhanced method to incorporate common sense knowledge into pre-trained embeddings. We learn embeddings for both words and their relations in the knowledge graph and represent the relations as linear translation between the embedding vectors of words. Additionally, we propose a novel weighting scheme for different edges in knowledge graphs by considering the relatedness of the two words of the edge in terms of neighborhood similarity.
- We propose an effective joint learning algorithm that minimizes two losses simultaneously - (1) the loss of acquiring a word's representation vector via relation translation and (2) the loss of recovering the pre-trained word embeddings through linear projection of the ensemble embedding.
- We establish a thorough theoretical analysis to show that several embedding models are restricted version of the RA-Retrofit model proposed in this paper.
- We conduct extensive experiments to show the superiority of the proposed ensemble learning method. Our method significantly outperforms individual pre-training word embeddings (word2vec [18] and Glove [24]) and the state-of-the-art ensemble methods (CN [28] and Meta-Embedding [35]) in both word analogy and word similarity tasks.

2 RELATED WORKS

Our work is related to *word embedding*, *ensemble method*, and *knowledge graph embedding*. We relate research in these areas in turn.

2.1 Word Embedding

Learning word embeddings has been an active research problem for the past couple of years. A host of approaches has been proposed and they have been shown to be successful in improving and simplifying many natural language applications. Most of these word embeddings are trained on large unlabeled text corpora. Continuous bag-of-words (CBOW) and Skip-Gram [18] are two typical embedding methods that either use the context words to predict the target word or vice-versa. GloVe [24] is another unsupervised learning algorithm for learning word embeddings based on context words. It is trained with global word co-occurrence counts and learns a set of word embeddings such that the dot product of the embeddings of two words is approximately equal to the logarithm of their co-occurrence count. The above word embedding methods learn two vectors for each word – a representation vector for the word acting as a target word and a context vector for the word acting as a context word. Usually, only the representation vectors are used for computing similarity.

Different from our proposed approach, the above word embedding models learn the representation only from text corpora. In our proposed approach, we incorporate knowledge graph into our ensemble model. We use the relations in knowledge graph to refine the word embeddings.

2.2 Ensemble Method

Motivated by the fact that different word embedding are trained on different corpora and these corpora are often not publicly available, a group of ensemble methods has been proposed to combine these pre-trained word embeddings. They can be categorized into two types.

The first type of ensemble methods [11, 15, 31, 32, 35] aims to combine multiple embeddings to generate an ensemble embedding with better quality. For instance, Tsuboi et al. [31] incorporate word2vec and GloVe embeddings into a POS tagging system and show that the ensemble model outperforms both individual models. Turian et al. [32] show that the ensemble model that combines CW embeddings [5] and HLBL embeddings [21] outperforms the individual models in natural language applications like named entity recognition and chunking. Luo et al. [15] train the CBOW model with three datasets and propose an ensemble model to combine the word embeddings learned from the three datasets. They show that the ensemble model achieves better performance than using a single word embedding. Hill et al. [11] propose an ensemble method based on machine translation. Yin et al. [35] propose an ensemble approach of combining different public embedding sets to learn meta-embeddings. They show that the meta-embeddings achieve better performance in word similarity and word analogy tasks compared to individual models. These approaches do not incorporate external semantic resources, and thus they are different from our proposed model.

The second type of ensemble methods incorporates external semantic resources to improve word embeddings [7, 28, 36]. One of the most famous semantic resources is knowledge graph, including, but not limited to PPDB [9], WordNet [19] and ConceptNet [28]. The second type of ensemble methods can be further divided into two categories. In the first category, the ensemble methods use semantic resources as a feature when learning the word embeddings [3, 34, 36]. Yu et al. [36] propose a new training objective for learning word embeddings that incorporates prior knowledge. The model extends the objective of word2vec [18] to take prior knowledge about synonyms from semantic resources into account. In the second category, the ensemble methods retrofit the semantic resources into existing word embeddings [7, 28]. Faruqui et al.

[7] introduce a technique known as "retrofitting", which combines embeddings learned from the existing corpus with a source of structured connections between words. Robert et al. [28] modify the retrofitting algorithm, making it not depending on the row order of its input matrix, and allowing it to propagate over the union of the vocabularies. These approaches differ from our proposed model in the aspect that they disregard the types of the relations in the knowledge graph. In contrast, the proposed model interprets relation as linear translation from a word to the other.

2.3 Knowledge graph embedding

The proposed model is also related to knowledge graph embedding. Various types of representation learning methods have been proposed to embed entities and relationships of knowledge graphs into low dimensional vector spaces. Given the representation vectors of entities and relations, these methods define a scoring function on each fact to measure its plausibility. For example, given two entity vectors, the model of Neural Tensor Network [27] represents each relation as a bilinear tensor operator followed by a linear matrix operator. The model of TransE [1], on the other hand, represents each relation as a single vector that linearly interacts with the entity vectors. Likewise, variations on entity representations also exist. RESCAL [23] models triples through bilinear operations over entity and relation representations.

Recently, more and more researchers have started to represent a graph as a document. These approaches sample sequences of nodes from the underlying graph and turn a graph into an ordered sequence of nodes. For example, DeepWalk [25] shows that the distribution of nodes appearing in short random walks is similar to the distribution of words in natural language and employes Skip-Gram to learn the representations of nodes. Node2vec [10] defines a flexible notion of a node's network neighborhood and design a biased random walk procedure. KGloVe [4] presents an approach in that exploits global patterns for creating vector space embeddings, which is inspired by the GloVe model.

Since the adjacency matrix of a knowledge graph is sparse [13], the entity embeddings learned from a knowledge graph cannot provide enough information. In the proposed method, we view the pre-trained word embeddings as distinct descriptions of words and combine it with the entity embeddings.

3 METHOD

In this section, we present an ensemble model that combines pre-trained word embedding models and knowledge graphs. Important notations are summarized in Table 1.

3.1 Problem Formulation

In this subsection we formulate the problem to be investigated. We start with some basic definitions.

Definition 3.1. (**Word Embedding Matrix**): Let V be a vocabulary of words. A word embedding matrix $W \in \mathbb{R}^{|V| \times d}$ contains d-dimensional continuous representation for each word in V. Each row w_v in W is the representation vector for word $v \in V$.

Definition 3.2. (**Knowledge Graph**): A knowledge graph $G(V, E)$ is a directed graph, where V is the set of nodes, or equivalently the vocabulary of words, and E is the set of edges. Each edge in E connects a source word $v \in V$ with target word $u \in V$ via a relation type r. Each edge in E is denoted as a tuple $e = (v, r, u)$, where $v, u \in V$, and $r \in R$ is a type of relation.

Table 1: Notations

Notation	Description
$w_v^i \in \mathbb{R}^{d_i}$	pre-trained word embedding for word v in model i
V	vocabulary
R	relation set
d_i	dimension of the i-th pre-trained word embeddings
$G(V, E)$	knowledge graph G with entity set V and edge set E
(v, r, u)	an edge in E specifying relation r between two words $v, u \in V$
$q_r \in \mathbb{R}^d$	d-dimensional embedding for relation r
$y_v \in \mathbb{R}^d$	the target d-dimensional ensemble word embedding for word v

Consider t pre-trained word embedding matrices $W^1, \ldots, W^t$ and a knowledge graph G on a global vocabulary V, we aim to train (1) an ensemble word embedding matrix Y, where each row $y_v \in \mathbb{R}^d$ is the d-dimensional representation vector for $v \in V$, and (2) the representation vector $q_r \in \mathbb{R}^d$ for each relation $r \in R$.

3.2 Overview

We propose an ensemble model to combine the linguistic information in a set of t pre-trained word embedding models and knowledge graphs. The learning objective of our ensemble model is twofold: (1) To leverage the knowledge graph, the representation vector y_v for word v should be close to the addition of the representation vector of its related word and the relation that connects the two words. (2) To leverage the t word embedding models, the representation vector y_v for word v in the target model should be close to its representation vector w_v^i in the pre-trained word embedding models via linear projection.

To achieve the learning objectives, we first introduce learning from knowledge graph (3.3) and learning from word embedding models (3.4) separately. Then we introduce the weighted global objective function for our ensemble model (3.5), and the corresponding weighting scheme by considering the relevance strength of each edge in the knowledge graph (3.6). To minimize the objective function, we also propose the positive and negative sampling methods (3.7). We elaborate on these issues in turn in the subsequent subsections.

3.3 Learning from Knowledge Graphs

The edges in the knowledge graph reveal a lot of syntactic and semantic relations between the words. A host of work has been proposed to leverage the information in the knowledge graph to improve the word embedding models. Faruqui et al. [7] introduce the "retrofitting" procedure, which refines dense matrices of embeddings to take into account external knowledge from a sparse semantic network. Robert et al. [28] improve the retrofitting algorithm by making it not depending on the row order of its input

matrix and allowing it to propagate over the union of the vocabularies. Both methods define the dissimilarity between two words in the Euclidean space, i.e., dissimilarity between two words is the Euclidean distance between the representation vectors of the two words in the edge. Specifically, given an edge (u, r, v) from the knowledge graph, the dissimilarity between u and v is defined as

$$\phi(u, v) = \| y_u - y_v \|, \tag{1}$$

where y_u and y_v are the learned representation vectors for u and v, respectively.

By using the Euclidean distance, it can bring the embeddings of two similar words closer to each other. However, this dissimilarity measure disregards the types of the relation in the edge. For instance, (obscurity, *RelatedTo*, darkness) and (sleeping, *HasPrerequisite*, darkness) are two edges in the knowledge graph. Though "obscurity" and "sleeping" are both connected to "darkness", they are connected by different types of edges. Apparently, it is not reasonable to encourage the embedding vectors of "obscurity" and "sleeping" to be close to each other.

How can we leverage the type of the relation while we are learning from knowledge graph? To answer the question, we first think about *discovering analogical relations*, which is one of the most popular benchmarks for word embeddings. This task, which is first proposed in [18], shows that the proportional analogies (A is to B as C is to D) can be solved by finding the vector closest to the hypothetical vector calculated as $w_C - w_A + w_B$, e.g. vec("king") - vec("man") + vec("woman") $\approxeq$ vec("queen"). In knowledge graph, proportional analogy (A is to B as C is to D) is relationally equal to edges (A, r, B) and (C, r, D) having the same type of relation. Hence, it is a natural idea to interpret an edge as linear translation operating on the embeddings of the two words. Specifically, given an edge (u, r, v), we represent the relation r with a vector based on its type. The embedding vector of v should be close to the addition of the embeddings of u and r. Therefore, we define the following function to evaluate the dissimilarity for an edge:

$$\phi(u, r, v) = \| y_u + q_r - y_v \|, \tag{2}$$

where y_u and y_v are the learned representation vectors for u and v, respectively, and q_r is the embeddings for relation r. Given two edges (A, r, B) and (C, r, D) from the knowledge graph, Equation 2 encourages the representation vector $y_v = w_D$ (w_B) to be close to the addition of $y_u = w_C$ and q_r (w_A and q_r). Hence, $w_B - w_A + w_C$ will be close to w_D.

In fact, the idea of interpreting edges as translation operating on vectors of nodes has been investigated by efforts on multi-relational data embedding [1], where a model called TransE is proposed to embed entities and relationships of multi-relational data into vectors. Since the TransE model is designed for multi-relational data embedding, we also combine it with other word embedding models to make it suitable for word embedding problem, denoted by TransE+. We will show in Section 5 that the ensemble model proposed in this paper outperforms both TransE and TransE+ in all tasks.

Let (u, r, v) be a positive edge, i.e., a real edge from E. Let (u', r, v') be a negative edge, i.e., (u', r, v') does not exists in E. In order to learn from knowledge graph, we minimize a weighted margin-based ranking criterion over the edges in the knowledge graph:

$$\mathcal{L}_1(u, u', v, v', r) = \mathcal{W}(u, r, v) \max\{m + \phi(u, r, v) - \phi(u', r, v'), 0\}, \tag{3}$$

where m is the margin, $\phi(\cdot)$ is the score function defined in Equation 2, and $\mathcal{W}(u, r, b)$ is the weight of an edge (to be discussed in section 3.6). The intuition is that the dissimilarity of positive edges should be smaller than the negative ones.

3.4 Learning from Word Embedding Models

We consider t word embedding models that are trained with different algorithms and corpora. They can be viewed as distinct descriptions of words. In this section, we first present how to pre-process these word embedding models. Then we present how to learn from these word embedding models.

3.4.1 Aligning Vocabularies. Different word embeddings are trained with different corpora. Moreover, these models apply different pre-processing steps to produce their own vocabularies. For instance, word2vec [18] joins multi-word phrases with underscores while GloVe [24] does not. These pre-processing steps produce different vocabularies for each model. We can combine these models only if their vocabularies are aligned. We next present a unified pre-processing procedure that combines all the pre-processing steps that the word embedding models apply.

The unified pre-processing includes five steps: tokenizing text to separate words from punctuation, joining multi-word phrases with underscores, removing a small list of stopwords from multi-word phrases, folding the text to lowercase, and replacing multiple digits with the character '#'.

This pre-processing procedure is a many-to-one procedure. For instance, "When", "WHEN", and "when" are all in the vocabulary of word2vec. Each of the three terms corresponds a pre-trained representation vector. After the unified pre-processing procedure, they are all transformed to "when" in the new vocabulary. In this case, the term "when" in the new vocabulary is associated with three pre-trained representation vectors. As a result, we need to merge the associated pre-trained vectors for the transformed terms. In this paper, we follow the strategy proposed in a previous work [28] to merge the associated representation vectors by computing a weighted average based on terms' frequencies. Specifically, for a pre-trained word embedding model W, let $v_1, \ldots, v_k$ be the set of terms that will be transformed to a term v after the unified pre-processing step. The merged representation vector for term v is defined as

$$w'_v = \sum_{j=1}^{k} freq(v_j) w_{v_j} \tag{4}$$

where w'_v is the merged representation vector for v, and w_{v_j} is the representation vector of v_i in the pre-trained embedding matrix, and $freq(v_j)$ is the frequency of v_j

In the remaining of the paper, we assume the vocabularies of all pre-trained word embedding models have been aligned. With a slight abuse of notation, we still use w^i_v to denote the merged representation vector of term v in model W^i.

3.4.2 Learning from Pre-trained Word Embedding Models. Given a word $v \in V$ and t word embedding models $W^1, \ldots, W^t$, we use a

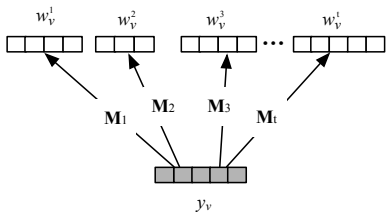


Figure 1: Network for Combining Pre-trained Embeddings

simple neural network to learn the ensemble embeddings, as shown in Figure 1. Let Y be the ensemble embedding model. We define a projection from space Y to space W^i, $i \in [1, t]$ as follows:

$$\hat{w}_v^i = \mathbf{M}_i y_v, \tag{5}$$

where $y_v \in \mathbb{R}^d$ is the embedding of word $v \in V$, $\hat{w}_v^i \in \mathbb{R}^{d_i}$ is the predicted embedding vector of v in W^i and d_i is the dimension of the representation vectors in model W^i, $\mathbf{M}_i \in \mathbb{R}^{d_i \times d}$ is the transformation matrix.

The predicted embedding vector $\hat{w}_v^i$ of word v should be close to the embedding vector w_v^i in model W^i. Thus, we define the objective function for a word v as:

$$\begin{aligned} \mathcal{L}_2(v) &= \sum_{i=1}^{t} \parallel \hat{w}_v^i - w_v^i \parallel \\ &= \sum_{i=1}^{t} \parallel \mathbf{M}_i y_v - w_v^i \parallel, \end{aligned} \tag{6}$$

The intuition behind Equation 6 is that we treat each individual embedding vector as a projection of the ensemble embedding vector. An individual embedding is a description of the word based on the corpus and the model that were used to create it. The ensemble embedding model tries to recover a more comprehensive description of the word when it is trained to predict the individual descriptions.

3.5 Global Objective Function

Now we proceed to give the global objective function. The global objective function is the linear combination of the two aforementioned objective functions, which is defined as follows:

$$\begin{aligned} \mathcal{L}(u, u', v, v', r) =& \mathcal{L}_1(u, u', v, v', r) + \mu \sum_{s \in \{u, u', v, v'\}} \mathcal{L}_2(s) \\ &+ \eta_1 \parallel \Theta_1 \parallel + \eta_2 \parallel \Theta_2 \parallel, \end{aligned} \tag{7}$$

where μ is a parameter to balance the two aforementioned loss functions, $\Theta_1 = \{ y_u, y_{u'}, y_v, y_{v'}, q_r \}$ is the set of embedding vectors of words u, u', v, v' and relation r, $\Theta_2 = \{\mathbf{M}_1, \ldots, \mathbf{M}_t\}$ is the set of transformation matrices that are defined to project the embedding vector in the ensemble model to the space of each individual embedding model.

In the global objective function, the first term $\mathcal{L}_1(u, u', v, v', r)$ makes the ensemble model to learn the information from knowledge graphs. The second term $\mathcal{L}_2(s)$ allows the ensemble model to capture the information in each individual embedding model. The last two terms are L2 regularization to avoid over-fitting, and $\eta_1 > 0$ and $\eta_2 > 0$ are the regularization coefficients.

Algorithm 1 Learning Ensemble Model

Input: Knowledge graph $G = (V, E)$, pre-trained embedding models $W^1, \ldots, W^t$, learning rate γ, ensemble embedding dimension d, component weight μ, η_1, η_2
Output: Ensemble embedding for each words $i \in V$ and relation $r \in R$; transformation parameters $\mathbf{M}_1, \ldots, \mathbf{M}_t$
Initialize ensemble embedding for each $i \in V$, $r \in R$ and parameters $\mathbf{M}_1, \ldots, \mathbf{M}_t$ with uniform distribution $(-\frac{1}{d}, \frac{1}{d})$
repeat
  for $(u, r, v) \in E$ **do**
    $P_N \leftarrow \emptyset$
    for k steps **do**
      $(u', r, v') \leftarrow$ Sample a negative edge
      $P_N \leftarrow (u', r, v')$
    end for
    Extract embeddings of all words in positive and negative edges from $W^1, \ldots, W^t$
    Compute loss function $\mathcal{L}$
    Update gradient $\Theta \leftarrow \Theta + \gamma \frac{\partial}{\partial \Theta} \mathcal{L}$
    Normalize word embeddings
  end for
until *convergence*

The optimization is carried out by stochastic gradient descent. In each step, we update all parameters $\Theta = \Theta_1 \cup \Theta_2$ by calculating the gradient:

$$\Theta \leftarrow \Theta + \gamma \frac{\partial}{\partial \Theta} \mathcal{L}, \tag{8}$$

where γ is the learning rate and $\frac{\partial}{\partial \Theta}$ indicates the gradient of parameters Θ. Finally, we normalize the embedding vectors for word v in the ensemble model Y:

$$\hat{y}_v = \frac{y_v}{max(\parallel y_v \parallel, 1)}. \tag{9}$$

The detailed optimization procedure of ensemble approach is summarized in Algorithm 1. All embedding vectors in the ensemble model are first randomly initialized. At each main iteration of the algorithm, the embedding vectors of the words are first normalized. For each positive edge in the knowledge graph, we first generate k negative edges. We utilize a list P_N to store all negative edges. Then for each individual word embedding model W^i, we extract the pre-trained embedding vectors for the words in the positive and negative edges. After that, we compute the loss function. The parameters are then updated by taking a gradient step with constant learning rate. The algorithm is stopped based on its performance on a validation set.

We also consider the out-of-vocabulary (OOV) words in the vocabulary union i.e., words that not covered by vocabulary intersection. For the words not appearing in the pre-trained embedding model, we initialize the OOV embeddings randomly and use the same mapping formula as for RA-retrofit to connect a ensemble embedding with the pre-trained embeddings. Both ensemble embedding and initialized OOV embeddings are updated during training. For the words not appearing in the graph model, we use equation 6 to learn the ensemble embeddings.

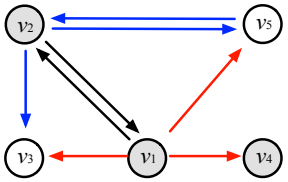


Figure 2: An example of knowledge graph

3.6 Weighting Scheme for an Edge

The relevances between the words of the two edges may be different even if they have same type of relation. For example, although edges (Internet, *RelatedTo*, Computer) and (Multipart, *RelatedTo*, Computer) have the same relation "RelatedTo", the relevance between the words "Internet" and "Computer" should be higher than that between words "Multipart" and "Computer". Intuitively, given an edge (u, r, v) from the knowledge graph, the relevance between words u and v is not only dependent on the type of relation r, but also dependent on the two words themselves. We have shown that we use a representation vector q_r to capture the relation r between two words. We next present how to capture the relevance between the two words in an edge.

Some prior work proposes to use manually defined constraints to distinguish the relevance between the words. For example, Ding et al. [6] propose to impose non-negativity constraints on vector representations, in which only positive samples will be used to learn the representations. Their approach has many limitations. First, it requires expertise to define the most appropriate constraint. Second, it is difficult to compare the relevance between words, especially when the number of words is large. Third, the relevance defined in their approach is highly subjective.

To solve this problem, we propose a method to compute the weight for each edge. Firstly, we define the neighbors of a word v in a knowledge graph as the words that can be connected with v via any relation r. Intuitively, if two words share more common neighbors, they are more likely to be highly relevant. Figure 2 shows an example of the knowledge graph. It can be seen in Figure 2 that v_1 shares more common neighbors with v_2 than with v_4. Specifically, the common neighbors of v_1 and v_2 are v_3 and v_5, while v_1 and v_4 have no common neighbors. Hence, v_1 and v_2 is likely to be more relevant.

Given a word v, let v^N be the set of neighbors of v. We define the weight of an edge (u, r, v) as the Jaccard similarity between the neighbors of u and v:

$$\mathcal{W}(u, r, v) = \epsilon_1 + \frac{|u^N \cap v^N|}{|u^N \cup v^N| + \epsilon_2}, \tag{10}$$

where $\epsilon_2 = 0.001$ is used to prevent division by zero, ϵ_1 is equal to the average value of $|u^N \cap v^N|/(|u^N \cup v^N| + \epsilon_2)$ in all edges such that the weight of an edge is always larger than 0.

3.7 Positive and Negative Sampling

The most frequent words can easily occur hundreds of millions of times. Such words usually provide less information value than the rare words. To counter the imbalance between the rare and frequent words, Mikolov et al. [18] use a simple subsampling approach: Each word v in the training set is discarded with probability computed by the equation:

$$P(v) = 1 - \sqrt{\frac{s}{f(v)}}, \tag{11}$$

where $f(v)$ is the frequency of word v and s is a chosen threshold.

In this paper, we extend the probability equation for edges in the knowledge graph. Given an edge (u, r, v), we define a combine frequency $f(u, v)$ over the corresponding frequency $f(u)$ and $f(v)$. We consider several choices for the combining operator which are summarized in Table 2. We define the subsampling formula for each edge as following:

$$P((u, r, v)) = 1 - \sqrt{\frac{s}{g(u, v)}}, \tag{12}$$

where $g(u, v)$ is the combined frequency of edge (u, r, v) (all combined frequencies are normalized to the interval $[0, 1]$) and s is a chosen threshold (we define $s = 1 \times 10^{-5}$ in this paper).

We next present the negative sampling technique to construct negative edges. Negative sampling assumes that two random words from the vocabulary V should be unrelated with high probability. Given an edge (u, r, v), we generate a set $E'_{(u,r,v)}$ of k negative edges by replacing either u or v by a random word v' from vocabulary V excluding u and v.

$$E'_{(u,r,v)} = \{(u, r, v')|v' \in V \setminus \{u, v\}\} \cup \{(u', r, v)|u' \in V \setminus \{u, v\}\} \tag{13}$$

Table 2: Combining operator $g(u, v)$ for u and v

Id	Operator	Definition
1	Average	$g(u, v) = \frac{f(u)+f(v)}{2}$
2	Max value	$g(u, v) = max(f(u), f(v))$
3	Geometric mean	$g(u, v) = \sqrt{f(u) \times f(v)}$

4 RELATIONSHIP TO OTHER MODELS

In the following we provide a unified view of several embedding models, by showing that they are restricted versions under our framework.

4.1 ConceptNet-Numberbath (CN)

Given an edge (u, r, v), the CN model [28] encourages the embedding vector of u to be close to both the pre-trained embedding vector of u and the embedding vector of its neighbor v. The objective of CN is to minimize:

$$\mathcal{L}_{cn}(u) = \alpha \parallel y_u - w_u \parallel + \beta_{u,v} \parallel y_u - y_v \parallel, \tag{14}$$

where w_u is pre-trained embedding vector of u, y_u and y_v are the embedding vectors of u and v in the ensemble model, respectively.

Proposition 1. CN can be fully recovered by RA-Retrofit embedding when $m = 0$, $\mathcal{W}(u, r, v) = 2\beta_{u,v}$, $t = 1$, $\alpha = \mu$, $\mathbf{M}$ is identity matrix, the number of negative samples $N_s = 0$, and remove the relation embedding q_r and regularization terms.

Proof. For each edge (u, r, v) from the knowledge graph, we remove the regularization terms and set $m = 0, N_s = 0, \mathcal{W}(u, r, v) =$

$2\beta_{u,v}$ and $t = 1$. We can get the objective function of RA-Retrofit as

$$\begin{aligned}\mathcal{L}(u,v,r) =& \mathcal{L}_1(u,v,r) + \mu\mathcal{L}_2(u) + \mu\mathcal{L}_2(v)\\ =& \mathcal{W}(u,r,v)\max\{\phi(u,r,v),0\} + \mu\mathcal{L}_2(u) + \mu\mathcal{L}_2(v)\\ =& 2\beta_{u,v} \parallel y_u + q_r - y_v \parallel + \mu \parallel \mathbf{M}y_u - w_u \parallel\\ &+ \mu||\mathbf{M}y_u - w_v||.\end{aligned} \tag{15}$$

By removing the relation embedding q_r, the objective function $\mathcal{L}$ can be rewrite as

$$\mathcal{L}(u,v) = 2\beta_{u,v} \parallel y_u - y_v \parallel + \mu \parallel y_u - w_u \parallel + \mu \parallel y_v - w_v \parallel . \tag{16}$$

The objective of an edge (u,r,v) to be minimized in CN can be viewed as the addition of the objective functions for u and v:

$$\begin{aligned}\mathcal{L}_{cn}(u,v) =& \mathcal{L}_{cn}(u) + \mathcal{L}_{cn}(v)\\ =& \alpha \parallel y_u - w_u \parallel + \beta_{u,v} \parallel y_u - y_v \parallel + \alpha \parallel y_v - w_v \parallel\\ &+ \beta_{u,v} \parallel y_v - y_u \parallel\\ =& \alpha \parallel y_u - w_u \parallel + 2\beta_{u,v} \parallel y_u - y_v \parallel + \alpha \parallel y_v - w_v \parallel .\end{aligned} \tag{17}$$

When $\alpha = \mu$, this function can be fully recovered by the proposed model. □

4.2 Meta-Embedding (MetaE)

MetaE [35] leverages multiple word embedding models and defines the objective function as:

$$\sum_{k=1}^{t} f_m(\parallel \mathbf{M}_k y_v - w_v^k \parallel + \gamma \parallel \mathbf{M}_k \parallel), \tag{18}$$

where y_v is the learned embedding of word $v \in V$, w_v^k is the embedding of word v in pre-trained embedding model W^k, $\mathbf{M}_k$ is transformation matrix.

Proposition 2. MetaE can be fully recovered by RA-Retrofit embeddings when $\mathcal{W}(u,r,v) = 0$, $f_m = \mu$, $\eta_1 = 0$ and $\eta_2 = f_m \times \gamma$.

Proof. When $\mathcal{W}(u,r,v) = 0$, the objective function of RA-Retrofit can be reduced to

$$\mathcal{L}(u,v) = \mu\mathcal{L}_2(u) + \eta_1 \parallel \Theta_1 \parallel + \eta_2 \parallel \Theta_2 \parallel . \tag{19}$$

For each word v in the edge, the objective of RA-Retrofit to be minimized can be reduced to

$$\begin{aligned}\mathcal{L}(v) =& \mu\mathcal{L}_2(v) + \eta_1 \parallel \Theta_1 \parallel + \eta_2 \parallel \Theta_2 \parallel\\ =& \mu \sum_{k=1}^{t} \parallel \mathbf{M}_k y_v - w_v^k \parallel + \eta_1 \parallel y_v \parallel + \eta_2 \sum_{k=1}^{t} \parallel \mathbf{M}_k \parallel .\end{aligned} \tag{20}$$

When $f_m = \mu$, $\eta_1 = 0$, $\eta_2 = f_m \times \gamma$, the proposed model can recover the meta-embed model. □

4.3 Translating Embeddings (TransE)

TransE [1] is a knowledge graph embedding model. The objective of the TransE is to minimize:

$$\begin{aligned}&\max\{m + \phi(u,r,v) - \phi'(u',r,v'), 0\}\\ &s.t. \quad \phi(u,r,v) = \parallel y_u + q_r - y_v \parallel,\end{aligned} \tag{21}$$

where where y_u and y_v are the learned representation vectors for u and v, respectively, q_r is the embeddings for relation r, $\phi(u,r,v)$ and $\phi'(u',r,v')$ are the scores for positive and negative edges, respectively.

Proposition 3. TransE can be fully recovered by RA-Retrofit when $\mu = \eta_1 = \eta_2 = 0$ and $\mathcal{W}(u,r,v) = 1$.

Proof. It is easy to verify that RA-Retrofit can recover TransE if we set $\mathcal{W}(u,r,v) = 1$ and $\mu = \eta_1 = \eta_2 = 0$. □

5 EXPERIMENTAL STUDY

In this section, we first describe the datasets that are used in the experimental study. Then we present the settings of our experiments. Finally, we investigate the performance of the evaluated methods. All the experiments are done on a computer having Intel(R) Core(TM) i7-6700 CPU @3.40 GHz processor, 6 GB RAM, and running Ubuntu 16.04. Our source codes will be released at https://github.com/fanglanting/RA-retrofit.

5.1 Datasets

We use GloVe[1] and word2vec[2] as the pre-trained embedding vectors for learning. We use ConceptNet (CNet)[3], PPDB[4] and WordNet (WNet)[5] as the knowledge graphs for retrofitting.

5.1.1 Pre-trained embeddings. **Glove** consists of 300 dimensional embedding vectors for 1.9 million words. These vectors were trained on 6 billion words from Wikipedia and English Gigaword. **Word2vec** are trained on 100 billion words of Google news dataset and are of length 300. It consists of embedding vectors for 3 million words.

5.1.2 Knowledge graphs. **CNet** originated as a machine-parsed version of the early crowd-sourcing project called Open Mind Common Sense, and has expanded to include several other data sources. It is a graph whose edges express common sense relationships between two short phrases or words, known as concepts. The edges are labeled from a defined set of relations, such as IsA, HasA, or UsedFor, expressing what relationship holds between the concepts. It contains a total of 2.8 millions of edges, in which 842, 194 edges remain after filtering out the edges whose words do not appear in the vocabularies of both word2vec and GloVe.

PPDB is another resource that is useful for learning about word similarity. It lists pairs of words that are translated to the same word in parallel corpora. Most of its edges are labeled with "Equivalent" and a small part is labeled with other types such as "OtherRelated" and "ForwardEntailment". It contains a total of 7.4 million edges for English language, in which 6 million edges remain in our experiments.

WNet is a lexical database for English. It groups English words into sets of synonyms called synsets, and records a number of relations among these synsets. It explicitly relates concepts with semantically aligned relations such as "hypernyms" and "hyponyms". For example, the word dog is a synonym of canine, a hypernym of puppy and a hyponym of animal. It contains a total of 843, 856 edges, in which 688, 147 edges remain in our experiments.

[1] https://nlp.stanford.edu/projects/glove/
[2] https://code.google.com/archive/p/word2vec/
[3] https://github.com/commonsense/conceptnet5/wiki
[4] http://paraphrase.org/#/download
[5] https://wordnet.princeton.edu/

5.2 Evaluated Methods

We evaluate three types of methods in our experimental study: individual embedding models, ensemble models, and RA-Retrofit model.

Individual embedding models. In the first type of methods, we evaluate the performance of three well-known embedding models: **GloVe** [24], word2vec [18], denoted by **W2V**, and **TransE** [1]. We use the pre-trained vectors for GloVe and word2vec that are published online. We combine CNet, PPDB and WNet as the knowledge graph to train the TransE embeddings.

Ensemble models. In the second type of methods, we consider two state-of-the-art ensemble models **CN** [28] and **MetaE** [35]. We use the published pre-trained vectors for CN directly. We train MetaE to combine word2vec and GloVe with the suggested hyper-parameters [35]. In addition, we adapt the TransE model, denoted by **TransE+**, to make it suitable for embedding words into vector space. Specifically, **TransE+** uses TransE, word2vec and GloVe as the pre-trained embeddings models, and use Eq. (6) to learn the ensemble embedding.

RA-Retrofit model. In the third type of methods, we evaluate the proposed RA-Retrofit model by varying the components in the model. Recall that the RA-Retrofit model combines knowledge graph and pre-trained word embeddings to learn the ensemble model. We vary the knowledge graphs and the pre-trained word embedding models that RA-Retrofit uses and get different variations of RA-Retrofit model. All these variations of RA-Retrofit models are trained by back-propagation with mini-batches. In the experiments, we set the hyper-parameters margin ridge m as 1, parameters μ as 0.01, η_1 and η_2 as 5×10^{-5}, learning rate γ as 0.02, the number of negative samples as 5 and iteration epoch as 100. When retrofitting a single knowledge graph, we set the batch size as 128. When retrofitting multiple knowledge graphs, we set the batch size as 256. In the subsampling process, we use geometric mean as the combine operator.

5.3 Evaluation Tasks

We conduct experiments on three tasks, word visualization, word similarity and word analogy to evaluate the performance of the proposed ensemble model.

The word anology task consists of questions like, "a is to b as u is to ?". In our experimental study, we use the state-of-the-art Google analogy test set [6]. Note that not all questions in the test set are covered by vocabularies of the embedding models. We filter out the uncovered questions and use 17,809 remaining questions in the test set. These questions are divided into two subsets: a semantic subset of size 8,733 and a syntactic subset of size 9,076. Given three words a, b, and u, we find the optimal word v for the analogy "a is to b as u is to v" by using the 3CosMul function [14] defined as follows:

$$v = \arg\max_{y_v} \frac{(\cos(y_v, y_b) + 1) \cdot (\cos(y_v, y_u) + 1)}{2(\cos(y_v, y_a) + 1) + 0.004} \tag{22}$$

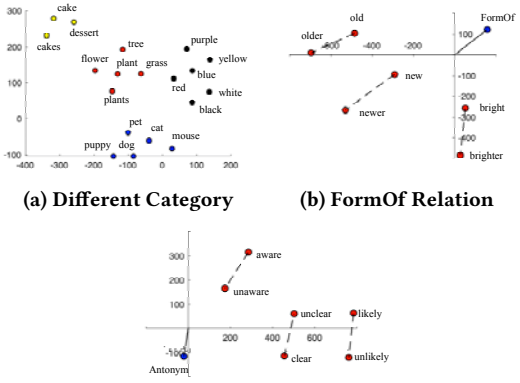
where y_a, y_b, y_u and y_v are the embeddings of word a, b, u and v.

In the word similarity task, we use the following word-similarity gold standards: WordSim-353 (WS353) [8], The Stanford Rare Words (RW) [16], MC-30 (MC) [20], MEN-3000 (MEN) [2] and Stanford Contextual Word Similarity (SCWS) [12]. Each test set contains pairs of words together with their human-assigned similarity scores. We only consider the pairs of words that are also in the vocabularies of the elvauated models. The test set WS353, RW, MC, MEN, and SCWS contains 353, 1707, 30, 3000, and 1971 valid pairs, respectively. For each test set, we calculate cosine similarity between the vectors of two words, and report Spearman's rank correlation coefficient [33] between the rankings produced by our model against the human rankings:

$$\rho = 1 - \frac{6 \sum_{i=1}^{N} \Delta_i^2}{N(N^2 - 1)}, \tag{23}$$

where Δ_i is the difference between the two ranks of each observation and N is the number of observations.

[6] http://download.tensorflow.org/data/questions-words.txt



(a) Different Category (b) FormOf Relation

(c) Antonym Relation

Figure 3: Visualization of word embeddings by t-SNE

5.4 Word Visualization

An intuitive way to understand the performance of a word embedding model is to visualize the embedding vectors. This task allows us to make qualitative assessment of the learned word embeddings.

We project word emebeddings into a 2-dimensional space using the t-SNE toolbox [17], as shown in Figure 3 (a). We can observe from Figure 3 (a) that words sharing similar semantics are closer to each other in the 2-dimensional space than the other words. For example, the words at the right side of Figure 3 (a) are all about color, while the words at the bottom are all related to animal.

To evaluate the analogy feature of the RA-Retrofit embeddings, we randomly select three edges with "FormOf" relation and three edges with "Antonym" relation from the knowledge graphs. Relation "FormOf" represents *A is an inflected form of B and B is the root word of A*. Relation "Antonym" represents *A and B are opposites in some relevant way*. We project the embedding vectors of words and relations in the edges into the 2-dimensional space. The embeddings of the selected edges are reported in Figure 3 (b) and Figure 3 (c). These results show that the embeddings can capture the analogy relationships between the words, i.e. A is to B as C is to D. For instance, vec("aware")-vec("unaware") is similar to vec("likely")-vec("unlikely"). Moreover, we observe that the embedding vector of a relation is similar to the subtraction between the embedding vectors of the two words. For instance, vec("aware")-vec("unaware") is similar to vec("'Antonym").

Table 3: Results on word analogy tasks. If a result is the best result in a block and better than all models in "Ind", then it is bolded. The best result for each benchmark is marked with †.

	Model	Semantic	Syntactic	Full
Ind	W2V	75.51	72.73	74.09
	GloVe	81.17	70.81	75.89
	TransE	31.65	6.17	18.66
Ens	CN	71.54	**76.43**	74.04
	MetaE	**82.10**	70.82	76.35
	TransE+	80.08	74.78	**77.38**
RA-Retrofit	GloVe			
	+PPDB	78.90	**73.89**	76.34
	+WNet	81.59	66.44	73.87
	+CNet	**84.40**	70.02	**77.07**
	+All	82.01	66.91	74.32
	W2V			
	+PPDB	72.56	**75.85**	74.24
	+WNet	78.63	67.76	73.09
	+CNet	**83.17**	69.46	**76.18**
	+All	81.56	68.41	74.86
	W2V + GloVe			
	+PPDB	77.32	75.66	76.47
	+WNet	81.46	68.88	75.05
	+CNet	**85.61**†	73.83	79.61
	+All	85.51	**78.46**†	**81.92**†

Table 4: Results on word similarity tasks. If a result is the best result in a block and better than all models in "Ind", then it is bolded. The best result for each benchmark is marked with †.

	Model	WS353	RW	MC	MEN	SCWS
Ind	W2V	70.35	53.41	80.25	77.56	66.28
	GloVe	76.25	52.00	83.99	81.95	66.14
	TransE	64.35	36.01	76.02	69.26	55.99
Ens	CN	**81.74**	**64.11**†	**88.51**	**86.40**	**73.13**†
	MetaE	78.39	53.67	83.54	83.18	66.37
	TransE+	76.54	54.86	87.93	83.06	66.38
RA-Retrofit	GloVe					
	+PPDB	77.89	54.01	85.61	84.93	67.07
	+WNet	79.88	56.78	86.28	84.19	66.98
	+CNet	81.87	57.00	84.28	85.10	68.14
	+All	**83.60**†	**62.38**	**87.93**	**86.21**	**71.03**
	W2V					
	+PPDB	70.64	53.74	81.78	81.19	66.87
	+WNet	72.23	58.43	**87.75**	80.01	65.92
	+CNet	75.33	59.09	84.79	82.70	67.00
	+All	**77.33**	**63.88**	85.39	**83.29**	**70.53**
	W2V + GloVe					
	+PPDB	76.40	53.85	84.05	83.49	68.31
	+WNet	80.48	61.47	89.11	84.22	69.71
	+CNet	**83.18**	62.24	89.86	**86.95** †	70.93
	+All	81.74	**62.64**	**90.20**†	86.13	**71.20**

5.5 Word Analogy

We next conduct experiments to evaluate the capability of the proposed model for semantic deduction with the word analogy task [18].

Table 3 reports the performance of each model in word analogy tasks. Block "Ind" lists the performance of individual embedding models W2V, GloVe and TransE. Block "Ens" lists the performance of state-of-the-art ensemble models CN, MetaE and TransE+. Block "RA-Retrofit" lists all the variations of the proposed RA-Retrofit models. Specifically, we use GloVe, W2V, and both GloVe and W2V as the pre-trained word embeddings for the RA-Retrofit model, respectively. We vary the knowledge graphs by using PPDB, WNet, CNet, and all three graphs (All).

From block "Ind" and "Ens", we observe that the analogy test accuracy of CN is comparable with W2V and GloVe in the full test set. This is because CN does not utilize the relation information, which is very important to analogy tasks. On the contrary, MetaE outperforms W2V and GloVe w.r.t. analogy test accuracy in the full test set. The results show that combing multiple embeddings can generate an ensemble embedding that contains more analogy information. Moreover, we observe that TransE has the worst accuracy compared with other methods. This is because the semantic information contained in knowledge graphs is not enough for word embeddings. When combined with pre-trained embeddings, TransE+ achieves a better accuracy than the other individual models or state-of-the-art ensemble models.

When the RA-Retrofit model is trained with GloVe or with W2V, we observe that combining knowledge graphs can improve the accuracy compared with each individual embedding model. Among the knowledge graphs, CNet achieves the best increase in the accuracy. This is because CNet contains more edges than WNet, and more relation categories than PPDB (CNet has 37 categories of relations, while PPDB has 6 categories).

When the RA-Retrofit model is trained with both GloVe and W2V, we observe that using all knowledge graphs achieves the best performance. This full model is 20% times better than CN in semantic test set, and 11% times better than W2V, 8% times better than GloVe, 11% times better than MetaE and 6% times better than TransE+ in the full test set.

Based on the observations, we conclude that the word representations generated by the RA-Retrofit model achieves the best performance in the word analogy task.

5.6 Word Similarity

Table 4 reports the empirical results on the word similarity task. We observe that the full RA-Retrofit model (trained with both pre-trained word embeddings and all knowledge graphs) outperforms all individual word embedding models. For instance, the full model is 16% times better than W2V and 7% times better than GloVe in WS353. Additionally, the full RA-Retrofit model performs better than ensemble models MetaE and TransE+. Although CN model generates competitive results compared with the proposed model, its analogy results are much worse than the proposed model. These results show that the proposed model can improve the performance of the word embedding not only at the analogy tasks but also at the word similarity tasks.

According to the results, we conclude that the proposed RA-Retrofit model is capable to generate high quality word embeddings compared with state-of-the-art methods.

Table 5: The effects of various modifications to the RA-Retrofit. Bold indicates the best result.

Model	Semantic	Syntactic	Total
RA-Retrofit-SVD	84.44	77.10	80.70
RA-Retrofit-unweighted	85.17	76.83	80.92
RA-Retrofit-Avg	85.15	77.58	81.29
RA-Retrofit-Max	84.47	78.24	81.30
RA-Retrofit-LogLoss	78.82	70.42	74.54
RA-Retrofit	**85.51**	**78.46**	**81.92**

5.7 Miscellaneous

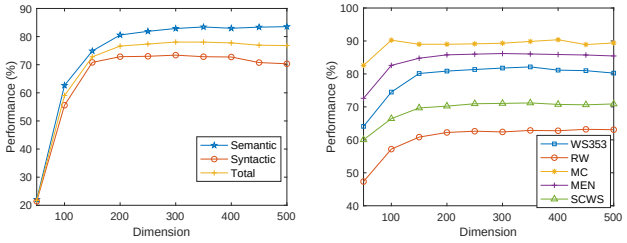
5.7.1 Ablation study for the RA-Retrofit model. We first conduct a set of experiments to justify the design of RA-Retrofit model by varying the components.

Effectiveness of transformation matrix. When learning from pre-trained word embeddings, we use a transformation matrix to project the ensemble embedding of a word such that the projection should be close to the word's pre-trained embedding vector. In this set of experiments, we modify the RA-Retrofit by replacing the transformation matrix with singular value decomposition (SVD). Specifically, we first concatenate the GloVe (300-dimensional) and word2vec (300-dimensional) vectors into 600-dimensional vectors. We then discount redundancy between its features by transforming these 600-dimensional vectors to 300-dimensional vectors with SVD. We encourage the ensemble embedding vector to be close to the decomposed vector. We refer to this modification as *RA-Retrofit-SVD*. The first row in Table 5 reports the accuracy of *RA-Retrofit-SVD*. We observe that using transformation matrix achieves better accuracy than using SVD.

Effectiveness of weight. As discussed in Section 3.6, we propose a scheme to assign weights to each edge. In this set of experiments, we modify the RA-Retrofit by using unweighted global loss to show the effectiveness of our weighting scheme. The modification is referred to as *RA-Retrofit-unweighted* and its accuracy is reported in the second row of Table 5. The results show that the weighting scheme improves the accuracy of the proposed model.

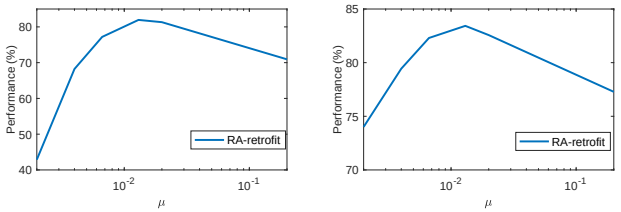
Effectiveness of combining operator. Recall that we discussed subsampling technique in Section 3.7, where we propose several combining operator to combine frequency of an edge. In this set of experiments, we replace the geometric mean in RA-Retrofit with *average* and *max*, respectively. The modifications are referred to as *RA-Retrofit-Avg* and *RA-Retrofit-Max*. The third and fourth rows in Table 5 report the accuracies of the two modifications. We observe that the RA-Retrofit model achieves the best performance when geometric mean is used as the combining operator.

Effectiveness of $\mathcal{L}_1$. As discussed in Section 3.3, we use margin-based ranking criterion to learn the information in the knowledge graphs. Another popular criterion that is often used to learn the knowledge graph embeddings is logistic loss. We modify the RA-Retrofit by changing the margin-based ranking criterion to logistic loss. The modification is referred to as *RA-Retrofit-LogLoss* and its accuracy is reported in the fifth row in Table 5. We observe that the performance drops rapidly. This is because the margin-based ranking criterion tend to assign low scores to positive edges and high scores to negative edges, which makes the learned representations more reasonable.



(a) Word Analogy **(b) Word Similarity**

Figure 4: Effect of Dimensions



(a) Word Analogy (Full) **(b) Word Similarity (WS353)**

Figure 5: Effect of parameter μ

5.7.2 Effects of dimensions and hyper-parameter μ. Figure 4 shows the effects of dimensionality $d \in [50, 500]$ for RA-Retrofit. There are no big differences in the word analogy and similarity tasks when the dimension d is high enough, roughly in the interval $[300, 500]$. In summary, as long as d is chosen to be large enough, the performance of the proposed model is robust.

Figure 5 demonstrates how hype-parameter $\mu \in [0.002, 0.2]$ affects RA-Retrofit in word similarity and word analogy tasks. We observe that the its performance degrades when the value of μ is either too small or too large. The best performance is obtained when μ is set to a value between 0.007 and 0.04. When μ is too small, the result is similar to only using the knowledge graph. When μ is too large, the result is similar to only using the pre-trained word embeddings. This experiment justifies that both knowledge graph and pre-trained word embeddings actually help improving word similarity computation and word analogy.

6 CONCLUSION

In this paper, we proposed a knowledge-enhanced ensemble learning method named RA-Retrofit for word embeddings. RA-Retrofit integrates common sense knowledge into an ensemble model of pre-trained distributional word embeddings to learn high quality word embeddings that is suitable for both word similarity and analogy. Our contributions include a method for translating embedding, equipped by a novel edge weighting scheme for knowledge graph, a joint learning method for optimizing both objectives from knowledge graph and pre-trained embeddings, and a thorough theoretical analysis among RA-Retrofit and other ensemble methods. Extensive experiments demonstrate that our proposed method is up to 20% times better than CN in word analogy task and up to 16% times better than W2V in word similarity task.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*. 2787–2795.

[2] Elia Bruni, Nam-Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *Journal of Artificial Intelligence Research* 49 (2014), 1–47.

[3] Kai-Wei Chang, Wen-tau Yih, and Christopher Meek. 2013. Multi-relational latent semantic analysis. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1602–1612.

[4] Michael Cochez, Petar Ristoski, Simone Paolo Ponzetto, and Heiko Paulheim. 2017. Global rdf vector space embeddings. In *International Semantic Web Conference*. 190–207.

[5] Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*. 160–167.

[6] Boyang Ding, Quan Wang, Bin Wang, and Li Guo. 2018. Improving Knowledge Graph Embedding Using Simple Constraints. *arXiv preprint arXiv:1805.02408* (2018).

[7] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2015. Retrofitting Word Vectors to Semantic Lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1606–1615.

[8] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*. 406–414.

[9] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 758–764.

[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[11] Felix Hill, Kyunghyun Cho, Sebastien Jean, Coline Devin, and Yoshua Bengio. 2014. Embedding word similarity with neural machine translation. *arXiv preprint arXiv:1412.6448* (2014).

[12] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*. 873–882.

[13] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge Graph Completion with Adaptive Sparse Transfer Matrix.. In *AAAI*. 985–991.

[14] Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the eighteenth conference on computational natural language learning*. 171–180.

[15] Yong Luo, Jian Tang, Jun Yan, Chao Xu, and Zheng Chen. 2014. Pre-Trained Multi-View Word Embedding Using Two-Side Neural Network.. In *AAAI*. 1982–1988.

[16] Thang Luong, Richard Socher, and Christopher Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the 17th Conference on Computational Natural Language Learning*. 104–113.

[17] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008), 2579–2605.

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[19] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[20] George A Miller and Walter G Charles. 1991. Contextual correlates of semantic similarity. *Language and cognitive processes* 6, 1 (1991), 1–28.

[21] Andriy Mnih and Geoffrey E Hinton. 2009. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*. 1081–1088.

[22] Avo Muromägi, Kairit Sirts, and Sven Laur. 2017. Linear Ensembles of Word Embedding Models. *arXiv preprint arXiv:1704.01419* (2017).

[23] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data.. In *ICML*, Vol. 11. 809–816.

[24] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing*. 1532–1543.

[25] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 701–710.

[26] Richard Socher, John Bauer, Christopher D Manning, et al. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 455–465.

[27] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[28] Robert Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. 4444–4451.

[29] Robert Speer and Catherine Havasi. 2013. ConceptNet 5: A large semantic network for relational knowledge. In *The People's Web Meets NLP*. Springer, 161–176.

[30] Stefanie Tellex, Boris Katz, Jimmy Lin, Aaron Fernandes, and Gregory Marton. 2003. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. 41–47.

[31] Yuta Tsuboi. 2014. Neural networks leverage corpus-wide information for part-of-speech tagging. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 938–950.

[32] Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*. 384–394.

[33] Arnold D Well and Jerome L Myers. 2003. *Research design & statistical analysis*. Psychology Press.

[34] Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 2014. Rc-net: A general framework for incorporating knowledge into word representations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. 1219–1228.

[35] Wenpeng Yin and Hinrich Schütze. 2016. Learning Word Meta-Embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 1351–1360.

[36] Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Vol. 2. 545–550.

Joint Entity Linking with Deep Reinforcement Learning

Zheng Fang
Institute of Information Engineering, Chinese Academy of Sciences & School of Cyber Security, University of Chinese Academy of Sciences
fangzheng@iie.ac.cn

Yanan Cao*
Institute of Information Engineering, Chinese Academy of Sciences
caoyanan@iie.ac.cn

Dongjie Zhang
Institute of Information Engineering, Chinese Academy of Sciences
zhangdongjie@iie.ac.cn

Qian Li
University of Technology Sydeney
Qian.Li@uts.edu.au

Zhenyu Zhang
Institute of Information Engineering, Chinese Academy of Sciences
zhangzhenyu1996@iie.ac.cn

Yanbing Liu
Institute of Information Engineering, Chinese Academy of Sciences
liuyanbing@iie.ac.cn

ABSTRACT

Entity linking is the task of aligning mentions to corresponding entities in a given knowledge base. Previous studies have highlighted the necessity for entity linking systems to capture the global coherence. However, there are two common weaknesses in previous global models. First, most of them calculate the pairwise scores between all candidate entities and select the most relevant group of entities as the final result. In this process, the consistency among wrong entities as well as that among right ones are involved, which may introduce noise data and increase the model complexity. Second, the cues of previously disambiguated entities, which could contribute to the disambiguation of the subsequent mentions, are usually ignored by previous models. To address these problems, we convert the global linking into a sequence decision problem and propose a reinforcement learning model which makes decisions from a global perspective. Our model makes full use of the previous referred entities and explores the long-term influence of current selection on subsequent decisions. We conduct experiments on different types of datasets, the results show that our model outperforms state-of-the-art systems and has better generalization performance.

CCS CONCEPTS

• **Information systems → Information extraction**.

KEYWORDS

Entity linking, reinforcement learning, joint disambiguation, knowledge base

ACM Reference Format:
Zheng Fang, Yanan Cao, Dongjie Zhang, Qian Li, Zhenyu Zhang, and Yanbing Liu. 2019. Joint Entity Linking with Deep Reinforcement Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313517

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313517

1 INTRODUCTION

Entity Linking (EL), which is also called Entity Disambiguation (ED), is the task of mapping mentions in text to corresponding entities in a given knowledge Base (KB). This task is an important and challenging stage in text understanding because mentions are usually ambiguous, i.e., different named entities may share the same surface form and the same entity may have multiple aliases. EL is key for information retrieval (IE) and has many applications, such as knowledge base population (KBP), question answering (QA), etc.

Existing EL methods can be divided into two categories: local model and global model. Local models concern mainly on contextual words surrounding the mentions, where mentions are disambiguated independently. These methods are not work well when the context information is not rich enough. Global models take into account the topical coherence among the referred entities within the same document, where mentions are disambiguated jointly. Most of previous global models [14, 27, 37] calculate the pairwise scores between all candidate entities and select the most relevant group of entities. However, the consistency among wrong entities as well as that among right ones are involved, which not only increases the model complexity but also introduces some noises. For example, in Figure 1, there are three mentions "France", "Croatia" and "2018 World Cup", and each mention has three candidate entities. Here, "France" may refer to *French Republic*, *France national basketball team* or *France national football team* in KB. It is difficult to disambiguate using local models, due to the scarce common information in the contextual words of "France" and the descriptions of its candidate entities. Besides, the topical coherence among the wrong entities related to *basketball team* (linked by an orange dashed line) may make the global models mistakenly refer "France" to *France national basketball team*. So, how to solve these problems?

We note that, mentions in text usually have different disambiguation difficulty according to the quality of contextual information and the topical coherence. Intuitively, if we start with mentions that are easier to disambiguate and gain correct results, it will be effective to utilize information provided by previously referred entities to disambiguate subsequent mentions. In the above example, it is much easier to map "2018 World Cup" to *2018 FIFA World Cup*

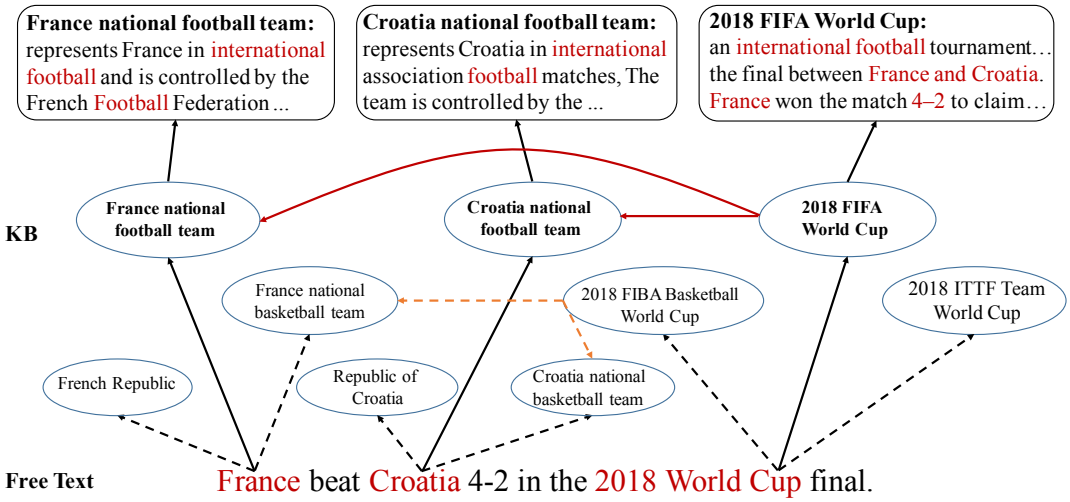


Figure 1: Illustration of mentions in the free text and their candidate entities in the knowledge base. Solid black lines point to the correct target entities corresponding to the mentions and to the descriptions of these correct target entities. Solid red lines indicate the consistency between correct target entities and the orange dashed lines denote the consistency between wrong candidate entities.

based on their common contextual words "France", "Croatia", "4-2". Then, it is obvious that "France" and "Croatia" should be referred to the national football team because football-related terms are mentioned many times in the description of *2018 FIFA World Cup*.

Inspired by this intuition, we design the solution with three principles: (i) utilizing local features to rank the mentions in text and deal with them in a sequence manner; (ii) utilizing the information of previously referred entities for the subsequent entity disambiguation; (iii) making decisions from a global perspective to avoid the error propagation if the previous decision is wrong.

In order to achieve these aims, we consider global EL as a sequence decision problem and propose a deep reinforcement learning (RL) based model, RLEL for short, which consists of three modules: Local Encoder, Global Encoder and Entity Selector. For each mention and its candidate entities, Local Encoder encodes the local features to obtain their latent vector representations. Then, the mentions are ranked according to their disambiguation difficulty, which is measured by the learned vector representations. In order to enforce global coherence between mentions, Global Encoder encodes the local representations of mention-entity pairs in a sequential manner via a LSTM network, which maintains a long-term memory on features of entities which has been selected in previous states. Entity Selector uses a policy network to choose the target entities from the candidate set. For a single disambiguation decision, the policy network not only considers the pairs of current mention-entity representations, but also concerns the features of referred entities in the previous states which is pursued by the Global Encoder. In this way, Entity Selector is able to take actions based on the current state and previous ones. When eliminating the ambiguity of all mentions in the sequence, delayed rewards are used to adjust its policy in order to gain an optimized global decision.

Deep RL model, which learns to directly optimize the overall evaluation metrics, works much better than models which learn with loss functions that just evaluate a particular single decision. By this property, RL has been successfully used in many NLP tasks, such as information retrieval [28], dialogue system [10] and relation classification [12], etc. To the best of our knowledge, we are the first to design a RL model for global entity linking. And in this paper, our RL model is able to produce more accurate results by exploring the long-term influence of independent decisions and encoding the entities disambiguated in previous states.

In summary, the main contributions of our paper mainly include following aspects:

- We are the first to consider EL as a sequence decision problem and innovatively utilize a deep reinforcement learning model in this task.
- The proposed model takes into account both local context and global coherence. In the process of global disambiguation, we make full use of the previous selected entity information and make decisions from a global perspective.
- We evaluate our model on several benchmark datasets and the experimental results show that our model achieves significant improvements over the state-of-the-art methods.

2 METHODOLOGY

The overall structure of our RLEL model is shown in Figure 2. The proposed framework mainly includes three parts: Local Encoder which encodes local features of mentions and their candidate entities, Global Encoder which encodes the global coherence of mentions in a sequence manner and Entity Selector which selects an entity from the candidate set. In the training process, Local Encoder is trained independently because its output is just used as the input of Entity Selector. Entity Selector and Global Encoder are

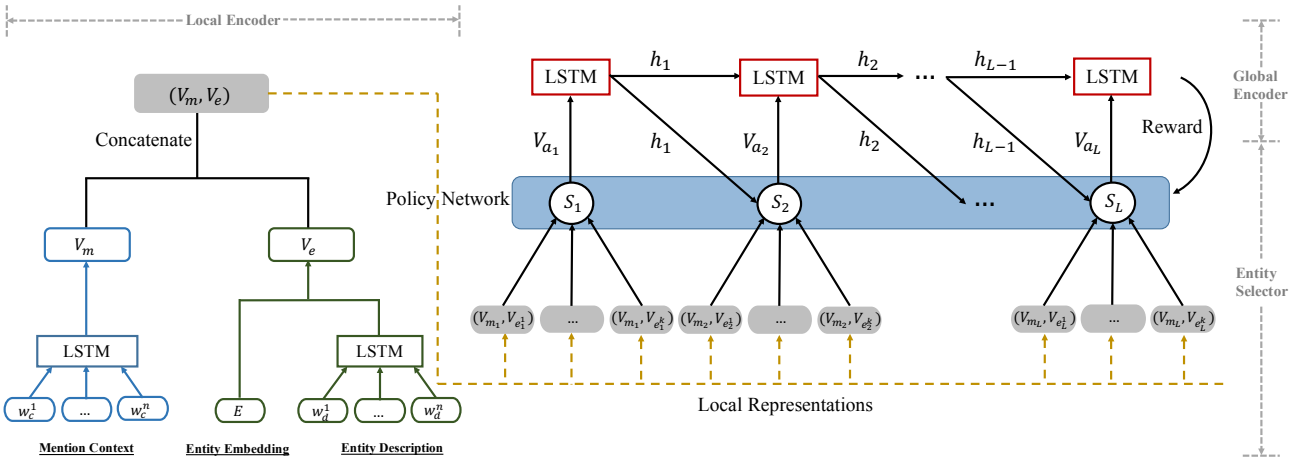


Figure 2: The overall structure of our RLEL model. It contains three parts: Local Encoder, Global Encoder and Entity Selector. In this framework, $(V_{m_t}, V_{e_t^k})$ denotes the concatenation of the mention context vector V_{m_t} and one candidate entity vector $V_{e_t^k}$. The policy network selects one entity from the candidate set, and V_{a_t} denotes the concatenation of the mention context vector V_{m_t} and the selected entity vector $V_{e_t^*}$. h_t represents the hidden status of V_{a_t}, and it will be fed into S_{t+1}.

trained jointly, because these two modules interact on each other. In the following, we will introduce the technical details of these modules.

2.1 Preliminaries

Before introducing our model, we firstly define the entity linking task. Formally, given a document D with a set of mentions $M = \{m_1, m_2, ..., m_k\}$, each mention $m_t \in D$ has a set of candidate entities $C_{m_t} = \{e_t^1, e_t^2, ..., e_t^n\}$. The task of entity linking is to map each mention m_t to its corresponding correct target entity e_t^+ or return "NIL" if there is no correct target entity in the knowledge base. Before selecting the target entity, we need to generate a certain number of candidate entities for model selection.

Inspired by the previous works [29, 31, 42], we use the mention's redirect and disambiguation pages in Wikipedia to generate candidate sets. For those mentions without corresponding disambiguation pages, we use its n-grams to retrieve the candidates [31]. In most cases, the disambiguation page contains many, sometimes even hundreds of entities. To optimize the model's memory and avoid unnecessary calculations, the candidate sets need to be filtered [1, 14, 37]. Here we utilize the XGBoost model [3] as an entity ranker, in which the useful features can be divided into two aspects, the one is string similarity such as the Jaro-Winkler distance between the entity title and the mention, the other is semantic similarity such as the cosine distance between the mention context representation and the entity embedding. Furthermore, we also use the statistical features based on the pageview and hyperlinks in Wikipedia. Empirically, we get the pageview of the candidate entities from the Wikipedia Tool Labs[1] which counts the number of visits on each entity page in Wikipedia. After ranking the candidates based on above features, we take the top k entities as final candidate set for each mention.

[1]The url of the website is: https://tools.wmflabs.org/pageviews/

2.2 Local Encoder

Given a mention m_t and the corresponding candidate set $\{e_t^1, e_t^2, ..., e_t^k\}$, we aim to get their local representation based on the mention context and the candidate entity description. For each mention, we firstly select its n surrounding words, and represent them as word embedding using a pre-trained lookup table [24]. Then, we use Long Short-Term Memory (LSTM) networks to encode the contextual word sequence $\{w_c^1, w_c^2, ..., w_c^n\}$ as a fixed-size vector V_{m_t}. The description of entity is encoded as $D_{e_t^i}$ in the same way. Apart from the description of entity, there are many other valuable information in the knowledge base. To make full use of these information, many researchers trained entity embeddings by combining the description, category, and relationship of entities. As shown in [14], entity embeddings compress the semantic meaning of entities and drastically reduce the need for manually designed features or co-occurrence statistics. Therefore, we use the pre-trained entity embedding $E_{e_t^i}$ and concatenate it with the description vector $D_{e_t^i}$ to enrich the entity representation. The concatenation result is denoted by $V_{e_t^i}$.

After getting $V_{e_t^i}$, we concatenate it with V_{m_t} and then pass the concatenation result to a multilayer perceptron (MLP). The MLP outputs a scalar to represent the local similarity between the mention m_t and the candidate entity e_t^i. The local similarity is calculated by the following equations:

$$\Psi(m_t, e_t^i) = MLP(V_{m_t} \oplus V_{e_t^i}) \tag{1}$$

Where $\oplus$ indicates vector concatenation. With the purpose of distinguishing the correct target entity and wrong candidate entities when training the local encoder model, we utilize a hinge loss that ranks ground truth higher than others. The rank loss function is defined as follows:

$$L_{local} = max(0, \gamma - \Psi(m_t, e_t^+) + \Psi(m_t, e_t^-)) \tag{2}$$

When optimizing the objective function, we minimize the rank loss according to [14, 37]. In this ranking model, a training instance is constructed by pairing a positive target entity e_t^+ with a negative entity e_t^-, where $\gamma > 0$ is a margin parameter. Our purpose is to make the score of the positive target entity e_t^+ be at least a margin γ higher than that of negative candidate entity e_t^-.

With the local encoder, we obtain the representation of mention context and candidate entities, which will be used as the input of the entity selector. In addition, the similarity scores calculated by MLP will be utilized for ranking mentions in the global encoder.

2.3 Global Encoder

In the global encoder module, we aim to enforce the topical coherence among the mentions and their target entities. So, we use an LSTM network which is capable of maintaining the long-term memory to encode the ranked mention sequence. What we need to emphasize is that our global encoder just encode the mentions that have been disambiguated by the entity selector which is denoted as V_{a_t}.

As mentioned above, the mentions should be sorted according to their contextual information and topical coherence. So, we firstly divide all mentions into several segments by the order they appear in the document based on the observation that the topical consistency attenuates along with the distance between the mentions. Then, we sort mentions in a segment from highest to lowest local similarity. As shown in Equation 1, the local similarity of m_i and its corresponding candidate entity e_i^t is defined as $\Psi(m_i, e_i^t)$. On this basis, we define $\Psi_{max}(m_i, e_i^a)$ as the the maximum local similarity between m_i and its candidate set $C_{m_i} = \{e_i^1, e_i^2, ..., e_i^n\}$. The $\Psi_{max}(m_i, e_i^a)$ is used as criterion when sorting mentions. For instance, if $\Psi_{max}(m_i, e_i^a) > \Psi_{max}(m_j, e_j^b)$, we place m_i before m_j. In this order, global encoder memorizes each selected entity and the encoding result will be served as input to the entity selector. Based on the topical coherence between the selected entities, entity selector will perform mention disambiguation from the global perspective.

Before using entity selector to choose target entities, we pretrain the global LSTM network. During the training process, we input not only positive samples but also negative ones to the LSTM. By doing this, we can enhance the robustness of the global encoder module. To pre-train this module, we adopt the following cross entropy loss function .

$$L_{global} = -\frac{1}{n}\sum_{x}\left[y \ln y^{'} + (1-y)\ln(1-y^{'})\right] \tag{3}$$

Where $y \in \{0, 1\}$ represents the real label of the candidate entity. If the candidate entity is correct, $y = 1$; otherwise $y = 0$. $y^{'} \in (0, 1)$ indicates the predicted result of our model. After pre-training the global encoder, we use the entity selector to choose the target entity for each mention and encode these selections.

2.4 Entity Selector

The entity selector module aims to select the target entity from the candidates based on the results of local and global encoder. In the process of sequence disambiguation, each selection will have an impact on subsequent decisions. Therefore, we convert the global entity disambiguation into a reinforcement learning problem and consider the entity selector as an agent. In our model, the agent is designed as a policy network which can learn a stochastic policy and prevents the agent from getting stuck at an intermediate state [39]. Under the guidance of policy, the agent decides which action (choosing one target entity from the candidates) should be taken at each state (current local and global encoding), and will receive a delay reward after all decisions are made. In the following, we will introduce the state, action and reward, and detail how to select target entity via a policy network.

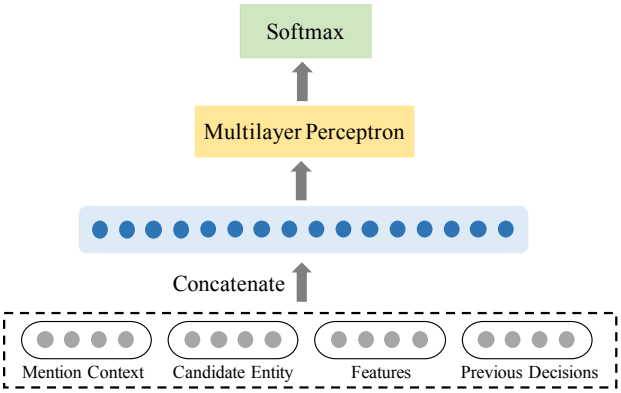


Figure 3: The architecture of policy network. It is a feedforward neural network and the input consists of four parts: mention context representation, candidate entity representation, feature representation, and encoding of the previous decisions.

State. The result of entity selection is based on the current state information. For time t, the state vector S_t is generated as follows:

$$S_t = V_{m_i}^t \oplus V_{e_i}^t \oplus V_{feature}^t \oplus V_{e^*}^{t-1} \tag{4}$$

Where $\oplus$ indicates vector concatenation. $V_{m_i}^t$ and $V_{e_i}^t$ respectively denote the vector of m_i and e_i at time t. For each mention, there are k candidate entities. In order to compare the semantic relevance between the mention and each candidate entity, we use multiple copies of the mention vector. Formally, we extend $V_{m_i}^t \in \mathbb{R}^{1\times n}$ to $V_{m_i}^{t\ '} \in \mathbb{R}^{k\times n}$ and then combine it with $V_{e_i}^t \in \mathbb{R}^{k\times n}$. Since $V_{m_i}^t$ and $V_{e_i}^t$ mainly represent semantic information, we add feature vector $V_{feature}^t$ to enrich lexical and statistical features. These features mainly include the popularity of the entity, the edit distance between the entity description and the mention context, the number of identical words in the entity description and the mention context, etc. In addition, the global feature $V_{e^*}^{t-1}$ is also added to S_t. As mentioned in global encoder module, $V_{e^*}^{t-1}$ is the output of the global LSTM network at time $t-1$, which encodes the mention context and the selected entity from time 0 to $t-1$. Thus, the state S_t consists of current information and previous decisions, containing the semantic features and statistical ones. Next, the concatenated vector will be fed into the policy network to generate action.

Action. According to the status at each time step, we take corresponding action. Specifically, we define the action at time step t is to

select the target entity e_t^* for m_t. The size of action space is the number of candidate entities for each mention, where $a_i \in \{0, 1, 2...k\}$ indicates the position of the selected entity in the candidate list. After performing all actions in the sequence we will get a delayed reward.

Reward. The agent takes the reward value as the feedback of its actions and learns the policy based on it. Since one selection has a long-term impact on subsequent decisions, we give a delay reward rather than an immediate one, reflecting whether the action improves the overall performance or not.

$$R(a_t) = p(a_t) \sum_{j=t}^{T} p(a_j) + (1 - p(a_t))(\sum_{j=t}^{T} p(a_j) + t - T) \quad (5)$$

where $p(a_t) \in \{0, 1\}$ indicates whether the current action is correct or not. If the action is correct, $p(a_t) = 1$; otherwise, $p(a_t) = 0$. Hence $\sum_{j=t}^{T} p(a_j)$ and $\sum_{j=t}^{T} p(a_j) + t - T$ respectively represent the number of correct actions and that of wrong ones from time t to the end of episode. Based on the above definition, our delayed reward can be used to guide the learning of the policy for entity linking.

Policy Network. After defining the state, action and reward, our main challenge is how to choose an action from the action space. To solve this problem, we sample the value of each action using a policy network $\pi_\Theta(a|s)$. The structure of the policy network is shown in Figure 3. The input of the network is the current state, including the mention context representation, candidate entity representation, feature representation, and encoding of the previous decisions. We concatenate these representations and fed them into a multilayer perceptron. For each hidden layer, we generate the output by:

$$h_i(S_t) = Relu(W_i * h_{i-1}(S_t) + b_i) \quad (6)$$

Where W_i and b_i are the parameters of the ith hidden layer, and we use *Relu* as the activation function. After getting the output of the last hidden layer, we feed it into a softmax layer which generates the probability distribution of actions. The probability distribution is generated as follows:

$$\pi(a|s) = Softmax(W * h_l(S) + b) \quad (7)$$

Where the W and b are the parameters of the softmax layer. For each mention in the sequence, we take one action to select the target entity from its candidate set. After performing all decisions in the episode, each action will get an expected reward and our goal is to maximize the expected total rewards. Formally, the objective function is defined as:

$$\begin{aligned} J(\Theta) &= \mathbb{E}_{(s_t, a_t) \sim P_\Theta(s_t, a_t)} R(s_1 a_1 ... s_L a_L) \\ &= \sum_t \sum_a \pi_\Theta(a|s) R(a_t) \end{aligned} \quad (8)$$

Where $P_\Theta(s_t, a_t)$ is the state transfer function, $\pi_\Theta(a|s)$ indicates the probability of taking action a under the state s, and $R(a_t)$ is the expected reward of action a at time step t. According to REINFORCE policy gradient algorithm[38], we update the policy gradient by the equation 9.

$$\Theta \leftarrow \Theta + \alpha \sum_t R(a_t) \nabla_\Theta \log \pi_\Theta(a|s) \quad (9)$$

Algorithm 1 The Policy Learning for Entity Selector

Require: Training data include multiple documents $D = \{D_1, D_2, ..., D_N\}$
Ensure: The target entity for mentions $\Gamma = \{T_1, T_2, ..., T_N\}$

```
Initialize the policy network parameter Θ, global LSTM network parameter Φ;
for D_k in D do
    Generate the candidate set for each mention
    Divide the mentions in D_k into multiple sequences S = {S_1, S_2, ..., S_N};
    for S_k in S do
        Rank the mentions M = {m_1, m_2, ..., m_n} in S_k based on the local similarity;
        for m_k in M do
            Sample the target entity e_k^* for m_k with Θ;
            Input the V_{m_k}^t and V_{e_k^*}^t to global LSTM network;
        end for
        // End of sampling, update parameters
        Compute delayed reward R(a_t) for each action;
        Update the parameter Θ of policy network:
            Θ ← Θ + α Σ_t R(a_t) ∇_Θ log π_Θ(a|s)
    end for
    Update the parameter Φ in the global LSTM network
end for
```

As the global encoder and the entity selector affect each other, we train them jointly after pre-training the two networks. The details of the joint learning are presented in Algorithm 1.

3 EXPERIMENT

In order to evaluate the effectiveness of our method, we train the RLEL model and validate it on a series of popular datasets that are also used by [14, 37]. To avoid overfitting with one dataset, we use both AIDA-Train and Wikipedia data in the training set. Furthermore, we compare the RLEL with some baseline methods, and our model achieves the state-of-the-art results. We implement our models in Tensorflow and run experiments on 4 Tesla V100 GPU.

3.1 Experiment Setup

Datasets. We conduct experiments on several different types of public datasets including news and encyclopedia corpus. The training set is AIDA-Train and Wikipedia datasets, where AIDA-Train contains 18448 mentions and Wikipedia contains 25995 mentions. In order to compare with the previous methods, we evaluate our model on AIDA-B and other datasets. These datasets are well-known and have been used for the evaluation of most existing entity linking systems. The statistics of the datasets are shown in Table 1.

- AIDA-CoNLL [20] is annotated on Reuters news articles. It contains training (AIDA-Train), validation (AIDA-A) and test (AIDA-B) sets.
- ACE2004 [34] is a subset of the ACE2004 Coreference documents.
- MSNBC [9] contains top two stories in the ten news categories(Politics, Business, Sports etc.)

Table 1: Statistics of document and mention numbers on experimental datasets.

Dataset	Doc Num	Mention Num	Mentions Per Doc
AIDA-Train	946	18448	19.5
AIDA-A	216	4791	22.1
AIDA-B	231	4485	19.4
ACE2004	36	251	7.1
MSNBC	20	656	32.8
AQUAINT	50	727	14.5
WNED-CWEB	320	11154	34.8
WNED-WIKI	320	6821	21.3
OURSELF-WIKI	460	25995	56.5

- AQUAINT [25] is a news corpus from the Xinhua News Service, the New York Times, and the Associated Press.
- WNED-CWEB [16] is randomly picked from the FACC1 annotated ClueWeb 2012 dataset.
- WNED-WIKI [16] is crawled from Wikipedia pages with its original hyperlink annotation.
- OURSELF-WIKI is crawled by ourselves from Wikipedia pages.

Training Details. During the training of our RLEL model, we select top k candidate entities for each mention to optimize the memory and run time. In the top k candidate list, we define the recall of correct target entity is R_t. According to our statistics, when k is set to 1, 5, 10, R_t is 0.853, 0.977, 0.993 respectively. Empirically, we choose top 5 candidate entities as the input of our RLEL model. For each candidate entity, in order to get the entity description information and reduce the impact of noise, we use TextRank algorithm [23] to select 15 keywords from its original description in wikipedia. Similarly, we choose 15 words around the mention as its context. In the global LSTM network, if the number of mentions does not reach the pre-defined sequence length, we adopt the mention padding strategy. In short, we use the last mention to pad the sequence until the number of mentions reaches the length.

Hyper-parameter setting. We set the dimensions of word embedding and entity embedding to 300, where the word embedding and entity embedding are released by [30] and [14] respectively. For parameters of the local LSTM network, the number of LSTM cell units is set to 512, the batch size is 64, and the rank margin γ is 0.1. Similarly, in global LSTM network, the number of LSTM cell units is 700 and the batch size is 16. In the above two LSTM networks, the learning rate is set to 1e-3, the probability of dropout is set to 0.8, and the Adam is utilized as optimizer. In the policy network, we set the number of MLP layers to 4 and extend the priori feature dimension to 50.

3.2 Comparing with Previous Work

Baselines. We compare RLEL with a series of EL systems which report state-of-the-art results on the test datasets. There are various methods including classification model [25], rank model [6, 34] and probability graph model [14, 16, 20, 21, 37]. Besides, Cheng *et al.*[5] formulate their global decision problem as an Integer Linear Program (ILP) which incorporates the entity-relation inference.

Table 2: In-KB accuracy result on AIDA-B dataset.

Methods	AIDA-B
Huang and Heck (2015)[21]	86.6%
Chisholm and Hachey (2015)[6]	88.7%
Guo and Barbosa (2016)[16]	89.0%
Globerson *et al.* (2016)[15]	91.0%
Yamada *et al.* (2016)[40]	91.5%
Ganea and Hofmann (2017)[14]	92.2%
Phong and Titov (2018)[37]	93.1%
our	**94.3%**

Globerson *et al.* [15] introduce a multi-focal attention model which allows each candidate to focus on limited mentions, Yamada *et al.*[40] propose a word and entity embedding model specifically designed for EL.

Evaluation Metric. We use the standard Accuracy, Precision, Recall and F1 at mention level (Micro) as the evaluation metrics:

$$Accuracy = \frac{|M \cap M^*|}{|M \cup M^*|} \tag{10}$$

$$Precision = \frac{|M \cap M^*|}{|M|} \tag{11}$$

$$Recall = \frac{|M \cap M^*|}{|M^*|} \tag{12}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{13}$$

where M^* is the golden standard set of the linked name mentions, M is the set of linked name mentions outputted by an EL method.

Results. As same as previous work, we use in-KB accuracy and micro F1 to evaluate our method. We first test the model on the AIDA-B dataset. From Table 2, we can observe that our model achieves the best result. Previous best results on this dataset are generated by [14, 37] which both built CRF models. They calculate the pairwise scores between all candidate entities. Differently, our model only considers the consistency of the selected entities and ignores the relationship between incorrect candidates. The experimental results show that our model can reduce the impact of noise and improve the accuracy of disambiguation. Apart from experimenting on AIDA-B, we also conduct experiments on several different datasets to verify the generalization performance of our model.

From Table 3, we can see that RLEL has achieved relatively good performances on ACE2004, CWEB and WIKI. Previous models [5, 14, 37] achieve better performances on the news datasets such as MSNBC and AQUINT, but their results on encyclopedia datasets such as WIKI are relatively poor. To avoid overfitting with some datasets and improve the robustness of our model, we not only use AIDA-Train but also add Wikipedia data to the training set. In the end, our model achieve the best overall performance.

In most existing EL systems, entities with lower frequency are difficult to disambiguate. For further analysis, we focus on the disambiguation accuracy of the gold entities in AIDA-B dataset with low popularity. We divide the gold entities according to their

Table 3: Compare our model with other baseline methods on different types of datasets. The evaluation metric is micro F1.

Methods	MSNBC	AQUAINT	ACE2004	CWEB	WIKI	Avg
Milne and Witten (2008)[25]	78%	85%	81%	64.1%	81.7%	77.96%
Hoffart and Johannes(2011)[20]	79%	56%	80%	58.6%	63%	67.32%
Ratinov and Lev[34]	75%	83%	82%	56.2%	67.2%	72.68%
Cheng and Roth (2013)[5]	90%	**90%**	86%	67.5%	73.4%	81.38%
Guo and Barbosa (2016)[16]	92%	87%	88%	77%	**84.5%**	85.7%
Ganea and Hofmann (2017)[14]	93.7%	88.5%	88.5%	77.9%	77.5%	85.22%
Phong and Titov (2018)[37]	**93.9%**	88.3%	89.9%	77.5%	78.0%	85.51%
our	92.8%	87.5%	**91.2%**	**78.5%**	82.8%	**86.56%**

Table 4: The micro F1 of gold entities with different pageviews on part of AIDA-B dataset.

Pageview/million	Mention Num	Micro F1
< 0.01	307	91.93%
0.01-0.1	612	86.06%
0.1-1	968	88.97%
1-5	1006	96.03%
5-10	493	96.43%
> 10	825	99.39%

pageviews in wikipedia, and the statistical disambiguation results are shown in Table 4. Since some pageviews can not be obtained, we only count a part of gold entities. The result indicates that our model still works well for low-frequency entities. But for medium-frequency gold entities, our model doesn't work well enough. The most important reason is that other candidate entities corresponding to these medium-frequency gold entities have higher pageviews and local similarities, which makes the model difficult to distinguish.

3.3 Discussion on different RLEL variants

To demonstrate the effects of RLEL, we evaluate our model under different conditions. First, we evaluate the effect of sequence length on global decision making. Secondly, we test whether sorting the mentions have a positive effect on the results. Thirdly, we analyze the results of not adding globally encoding during entity selection. Finally, we compare our RL selection strategy with the greedy choice.

Sequence in different length. A document may contain multiple topics, so we do not add all mentions to a single sequence. In practice, we add some adjacent mentions to the sequence and use reinforcement learning to select entities from beginning to end. To analyze the impact of the number of mentions on joint disambiguation, we experiment with sequences on different lengths. The results on AIDA-B are shown in Figure 4. We can see that when the sequence is too short or too long, the disambiguation results are both very poor. When the sequence length is less than 3, delay reward can't work in reinforcement learning, and when the sequence length reaches 5 or more, noise data may be added. Finally, we choose the 4 adjacent mentions to form a sequence.

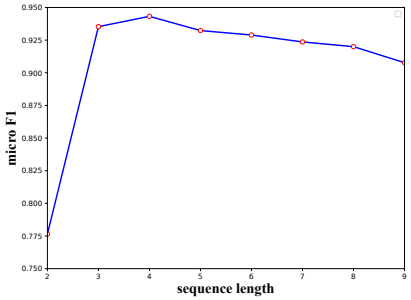
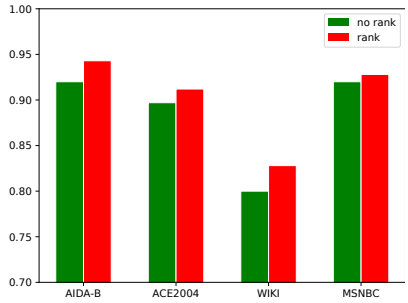


Figure 4: The performance of models with different sequence lengths on AIDA-B dataset.

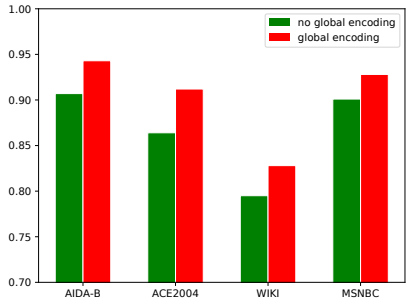
Influence of ranking mentions. In this section, we test whether ranking mentions is helpful for entity selections. We directly input the mentions into the global encoder by the order they appear in the text. We record the disambiguation results and compare them with the method which adopts ranking mentions. As shown in Figure 5a, the model with ranking mentions achieved better performances on most of datasets, indicating that it is effective to place the mention that with a higher local similarity in front of the sequence. It is worth noting that the effect of ranking mentions is not obvious on the MSNBC dataset. The reason is that most of mentions in MSNBC have similar local similarities, so the order of disambiguation has little effect on the final result.

Effect of global encoding. Different from previous global methods which use pairwise similarity between entities to capture their correlation, our model encodes the selected entity information in sequence. To evaluate whether the global encoding contributes to disambiguation, we compare the performance between the model with and without adding the global information. When the global encoding is not taken in to account, the current state only contains the local information, i.e., the mention context representation, candidate entity representation and feature representation. From the results in Figure 5b, we can see that the model with global encoding achieves an improvement of 4% accuracy over the model without global encoding.

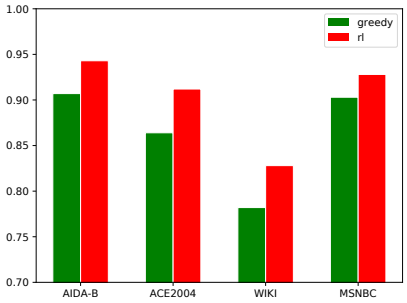
Different entity selection strategies. To illustrate the effectiveness for adopting the reinforcement learning for entity selection, we compare two entity selection strategies like [12]. Specifically, we



a.The influence of ranking mentions for entity selection.



b. The effect of the global encoding for entity selection.



c. Comparison of reinforcement learning selection with greedy choice.

Figure 5: The comparative experiments of RLEL model.

Table 5: Entity selection examples by our RLEL model.

Document Content	Mentions after ranking	Selected Target Entity(is correct)
Australia beat **West Indies** by five wickets in a **World Series** limited overs match at the **Melbourne Cricket Ground**on Friday...	1.Melbourne Cricket Ground 2.World Series 3.West Indies 4.Australia	1.Melbourne Cricket Ground(correct) 2.World Series Cricket(correct) 3.West Indies cricket team(correct) 4.Australia national cricket team(correct)
Instead of **Los Angeles International**, ..., consider flying into **Burbank** or **John Wayne Airport** in **Orange County**, Calif...	1.John Wayne Airport 2.Orange County 3.Los Angeles International 4.Burbank	1.John Wayne Airport(correct) 2.Orange County, California(correct) 3.Los Angeles International Airport(correct) 4.Burbank, California (wrong, the correct is "Hollywood Burbank Airport")

perform entity selection respectively with reinforcement learning and greedy choice. The greedy choice is to select the entity with the largest local similarity from candidate set. But the reinforcement learning selection is guided by delay reward, which has a global perspective. In the comparative experiment, we just replace the RL selection with a greedy choice, keeping the other conditions consistent. From the results in Figure 5c, we can draw a conclusion that our entity selector performs much better than greedy strategies.

3.4 Case Study

Table 5 shows two entity selection examples by our RLEL model. For multiple mentions appearing in the document, we first sort them according to their local similarities, and select the target entities in order by the reinforcement learning model. From the results of sorting and disambiguation, we can see that our model is able to utilize the topical consistency between mentions and make full use of the selected target entity information.

4 RELATED WORK

4.1 Entity Linking

Entity linking can be roughly divided into two major categories: local and global disambiguation. Early studies use local models to resolve mentions independently, they usually disambiguate mentions based on lexical matching between the mention's surrounding words and the entity profile in the reference KB. Various methods have been proposed to model mention's local context ranging from binary classification [25] to rank models [4, 11]. In these methods, a large number of hand-designed features are applied. For some marginal mentions which are difficult to extract features, researchers also exploit the data retrieved by search engines [7, 8] or Wikipedia sentences [36]. However, the feature engineering and search engine based methods are both time-consuming and laborious. Recently, with the popularity of deep learning models, representation learning is utilized to automatically extract semantic features [2, 17]. The entity representations learned by jointly modeling textual contexts and knowledge base are effective in combining multiple sources of information. To make full use of the information contained in representations, we also utilize the pre-trained entity embeddings in our model.

In recent years, with the assumption that the target entities of all mentions in a document shall be related, many novel global models for joint linking are proposed. Assuming the topical coherence among mentions, authors in [13, 33] construct factor graph models, which represent the mention and candidate entities as variable nodes, and exploit factor nodes to denote a series of features. Two recent studies [14, 37] use fully-connected pairwise Conditional Random Field(CRF) model and exploit loopy belief propagation to estimate the max-marginal probability. Moreover, PageRank or Random Walk [16, 19, 42] are utilized to select the target entity

for each mention. The above probabilistic models usually need to predefine a lot of features and are difficult to calculate the max-marginal probability as the number of nodes increases. In order to automatically learn features from the data, Cao *et al.* [1] applies Graph Convolutional Network to flexibly encode entity graphs. However, the graph-based methods are computationally expensive because there are lots of candidate entity nodes in the graph.

To reduce the calculation between candidate entity pairs, Globerson *et al.* [15] proposed a coherence model with an attention mechanism, in which each mention only focuses on a fixed number of mentions. Unfortunately, choosing the number of attention mentions is not easy in practice. Two recent studies [31, 32] finish linking all mentions by scanning the pairs of mentions at most once, they assume each mention only needs to be consistent with one another mention in the document. The limitation of their method is that the consistency information is too sparse, resulting in low confidence. Similar to us, Guo *et al.* [16] also sort mentions according to the difficulty of disambiguation, but they did not make full use of the information of previously referred entities for the subsequent entity disambiguation. Nguyen *et al.* [27] use the sequence model, but they simply encode the results of the greedy choice, and measure the similarities between the global encoding and the candidate entity representations. Their model does not consider the long-term impact of current decisions on subsequent choices, nor does they add the selected target entity information to the current state to help disambiguation.

4.2 Reinforcement Learning

In the last few years, reinforcement learning has emerged as a powerful tool for solving complex sequential decision-making problems. It is well known for its great success in the game field, such as Go [35] and Atari games [26]. Recently, reinforcement learning has also been successfully applied to many natural language processing tasks and achieved good performance [12, 22, 39]. Feng *et al.*[12] used reinforcement learning for relation classification task by filtering out the noisy data from the sentence bag and they achieved huge improvements compared with traditional classifiers. Zhang *et al.* [41] applied the reinforcement learning on sentence representation by automatically discovering task-relevant structures. To implement automatic taxonomy induction from a set of terms, Han *et al.* [18] designed an end-to-end reinforcement learning model to determine which term to select and where to place it on the taxonomy, and this model effectively reduced the error propagation between two phases. Inspired by the above works, we also use reinforcement learning in our framework.

5 CONCLUSIONS

In this paper we consider entity linking as a sequence decision problem and present a reinforcement learning based model. Our model learns the policy on selecting target entities in a sequential manner and makes decisions based on current state and previous ones. By utilizing the information of previously referred entities, we can take advantage of global consistency to disambiguate mentions. For each selection result in the current state, it also has a long-term impact on subsequent decisions, which allows learned policy strategy has a global view. In experiments, we evaluate our method on AIDA-B and other well-known datasets, the results show that our system outperforms state-of-the-art solutions. In the future, we would like to construct an end-to-end model which simultaneously optimize mention ranking and mention disambiguation.

ACKNOWLEDGMENTS

This research is supported by the National Key Research and Development Program of China (No. 2018YFB1004703), the Beijing Municipal Science and Technology Project under grant (No. Z181100002718004), and the National Natural Science Foundation of China grants(No. 61602466).

REFERENCES

[1] Yixin Cao, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Neural Collective Entity Linking. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*. 675–686.

[2] Yixin Cao, Lifu Huang, Heng Ji, Xu Chen, and Juanzi Li. 2017. Bridge Text and Knowledge by Learning Multi-Prototype Entity Mention Embedding. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. 1623–1633.

[3] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 785–794.

[4] Zheng Chen and Heng Ji. 2011. Collaborative Ranking: A Case Study on Entity Linking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*. 771–781.

[5] Xiao Cheng and Dan Roth. 2013. Relational Inference for Wikification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*. 1787–1796.

[6] Andrew Chisholm and Ben Hachey. 2015. Entity Disambiguation with Web Links. *TACL* 3 (2015), 145–156.

[7] Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Stefan Rüd, and Hinrich Schütze. 2016. A Piggyback System for Joint Entity Mention Detection and Linking in Web Queries. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. 567–578.

[8] Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Hinrich Schütze, and Stefan Rüd. 2014. The SMAPH system for query entity recognition and disambiguation. In *ERD'14, Proceedings of the First ACM International Workshop on Entity Recognition & Disambiguation, July 11, 2014, Gold Coast, Queensland, Australia*. 25–30.

[9] Silviu Cucerzan. 2007. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*. 708–716.

[10] Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards End-to-End Reinforcement Learning of Dialogue Agents for Information Access. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. 484–495.

[11] Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity Disambiguation for Knowledge Base Population. In *COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2010, Beijing, China*. 277–285.

[12] Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement Learning for Relation Classification From Noisy Data. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.

[13] Octavian-Eugen Ganea, Marina Ganea, Aurélien Lucchi, Carsten Eickhoff, and Thomas Hofmann. 2016. Probabilistic Bag-Of-Hyperlinks Model for Entity Linking. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. 927–938.

[14] Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep Joint Entity Disambiguation with Local Neural Attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. 2619–2629.

[15] Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2016. Collective Entity Resolution with Multi-Focal Attention. In *Proceedings of the 54th Annual Meeting of the*

Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers.

[16] Zhaochen Guo and Denilson Barbosa. 2018. Robust named entity disambiguation with random walks. *Semantic Web* 9, 4 (2018), 459–479.

[17] Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity Linking via Joint Encoding of Types, Descriptions, and Context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. 2681–2690.

[18] Jiawei Han, Xiang Ren, Jiaming Shen, Yuning Mao, and Xiaotao Gu. 2018. End-to-End Reinforcement Learning for Automatic Taxonomy Induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. 2462–2472.

[19] Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*. 765–774.

[20] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust Disambiguation of Named Entities in Text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*. 782–792.

[21] Hongzhao Huang, Larry P. Heck, and Heng Ji. 2015. Leveraging Deep Neural Networks and Knowledge Graphs for Entity Disambiguation. *CoRR* abs/1504.07678 (2015). arXiv:1504.07678

[22] Ting Liu, William Yang Wang, Yu Zhang, Qingyu Yin, and Weinan Zhang. 2018. Deep Reinforcement Learning for Chinese Zero Pronoun Resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. 569–578.

[23] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*. 404–411.

[24] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013). arXiv:1301.3781

[25] David N. Milne and Ian H. Witten. 2008. Learning to link with wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008*. 509–518.

[26] Volodymyr Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[27] Thien Huu Nguyen, Nicolas R. Fauceglia, Mariano Rodriguez-Muro, Oktie Hassanzadeh, Alfio Massimiliano Gliozzo, and Mohammad Sadoghi. 2016. Joint Learning of Local and Global Features for Entity Linking via Neural Networks. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*. 2310–2320.

[28] Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-Oriented Query Reformulation with Reinforcement Learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. 574–583.

[29] Aasish Pappu, Roi Blanco, Yashar Mehdad, Amanda Stent, and Kapil Thadani. 2017. Lightweight Multilingual Entity Extraction and Linking. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*. 365–374.

[30] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. 1532–1543.

[31] Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. 2017. NeuPL: Attention-based Semantic Matching and Pair-Linking for Entity Disambiguation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*. 1667–1676.

[32] Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. 2018. Pair-Linking for Collective Entity Disambiguation: Two Could Be Better Than All. *CoRR* abs/1802.01074 (2018). arXiv:1802.01074

[33] Chenwei Ran, Wei Shen, and Jianyong Wang. 2018. An Attention Factor Graph Model for Tweet Entity Linking. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. 1135–1144.

[34] Lev-Arie Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and Global Algorithms for Disambiguation to Wikipedia. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*. 1375–1384.

[35] David Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. v. d. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. P. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529, 7587 (2016), 484–489.

[36] Chuanqi Tan, Furu Wei, Pengjie Ren, Weifeng Lv, and Ming Zhou. 2017. Entity Linking for Queries by Searching Wikipedia Sentences. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. 68–77.

[37] Ivan Titov and Phong Le. 2018. Improving Entity Linking by Modeling Latent Relations between Mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. 1595–1604.

[38] Ronald J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* 8 (1992), 229–256.

[39] Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. 564–573.

[40] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*. 250–259.

[41] Tianyang Zhang, Minlie Huang, and Li Zhao. 2018. Learning Structured Representation for Text Classification via Reinforcement Learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. 6053–6060.

[42] Stefan Zwicklbauer, Christin Seifert, and Michael Granitzer. 2016. Robust and Collective Entity Disambiguation through Semantic Embeddings. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*. 425–434.

Measurement and Early Detection of Third-Party Application Abuse on Twitter

Shehroze Farooqi
The University of Iowa

Zubair Shafiq
The University of Iowa

ABSTRACT

Third-party applications present a convenient way for attackers to orchestrate a large number of fake and compromised accounts on popular online social networks. Despite recent high-profile reports of third-party application abuse on popular online social networks, prior work lacks automated approaches for accurate and early detection of abusive applications. In this paper, we perform a longitudinal study of abusive third-party applications on Twitter that perform a variety of malicious and spam activities in violation of Twitter's Terms of Service (ToS). Our measurements spanning over a period of 16 months demonstrate an ongoing arms race between attackers continuously registering and abusing new applications and Twitter trying to detect them. We find that hundreds of thousands of abusive applications remain undetected by Twitter for several months while posting tens of millions of tweets. We propose a machine learning approach for accurate and early detection of abusive Twitter applications by analyzing their first few tweets. The evaluation shows that our machine learning approach can accurately detect abusive application with 92.7% precision and 87.0% recall by analyzing their first seven tweets. The deployment of our machine learning approach in the wild shows that attackers continue to abuse third-party applications despite Twitter's recent countermeasures targeting third-party applications.

CCS CONCEPTS

• **Security and privacy → Social network security and privacy**.

KEYWORDS

Abuse; API; Online Social Networks; Spam; Twitter

ACM Reference Format:
Shehroze Farooqi and Zubair Shafiq. 2019. Measurement and Early Detection of Third-Party Application Abuse on Twitter. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313515

1 INTRODUCTION

Background. Popular social networking sites, including Twitter, allow developers to use third-party applications to enhance user experience. Millions of applications use Twitter's third-party developer platform to support news, gaming, entertainment, analytics, research, and publishing solutions [2]. Third-party Twitter applications (or simply Twitter applications) use OAuth [33] for getting permissions from users to read/write/message on their behalf [11, 15]. Twitter applications have perpetual access to user accounts unless users explicitly revoke their permissions. Naturally, an attacker can control a large number of accounts by tricking users into installing a malicious application [45] or compromising a popular legitimate application [36, 51].

Motivation. Third-party Twitter applications present a convenient way for attackers to orchestrate fake or compromised accounts through Twitter API [12]. Attackers can install third-party applications on fake accounts that they themselves create or buy in bulk from underground marketplaces [35, 58]. Attackers can also trick users (e.g., phishing [27], malicious browser extensions [41]) into installing their applications to compromise accounts. Attackers can even recruit real users on crowdturfing marketplaces to install their applications in exchange for monetary and non-monetary incentives (e.g., free followers) [36, 58, 60]. On several occasions during the last couple of years, Twitter has disclosed large-scale abuse by hundreds of thousands of third-party applications on their platform [7, 10, 45].

Limitations of Prior Art. Prior research has paid little attention to directly mitigate the role of third-party applications in propagating malware and spam on Twitter [34, 43, 52, 54, 55, 55, 57, 61]. While some prior research has reported the spread of malware and spam by third-party Twitter applications [34, 52, 55, 57], most efforts are focused on detecting the *sources* (fake and compromised accounts) and *targets* (retweets) of malicious activities on Twitter. We believe that directly targeting such abusive third-party Twitter applications is crucial for robust detection of increasingly sophisticated malicious activity on Twitter. Our belief is in line with Twitter's recently announced plans to target fake, coordinated, and automated account activities conducted by third-party applications on their platform [9, 10].

Measuring abusive Twitter Applications. In this work, we conduct a 16-month long longitudinal measurement study of abusive third-party applications that perform a variety of malicious and spam activities in violation of Twitter's Terms of Service (ToS) [20]. To collect a comprehensive ground truth of abusive Twitter applications, we retrospectively check whether tweets by third-party applications are removed by users or Twitter's abuse detection systems [5, 8, 58]. Prior work has also leveraged retrospective analysis of deleted/suspended tweets/accounts to study spam and malware campaigns on Twitter [56, 57]. We are able to identify 167,013 abusive third-party Twitter applications through retrospective analysis of tweets collected over a period of 16 months using Twitter's APIs [12, 17]. Our measurements reveal that there is an ongoing

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313515

arms race between attackers registering and abusing new applications and Twitter actively trying to detect and remove them. More specifically, we show that attackers are able to use a large pool of hundreds of thousands of abusive applications to post tens of millions of tweets. Abusive applications often evade detection for several months while posting millions of tweets for astroturfing [32, 52, 55], phone spam [39], and tricking users with deceiving claims to compromise accounts [4, 26, 57].

Early Detection of Abusive Twitter Applications. Accurate and early detection of abusive third-party applications can help in significantly mitigating malware and spam on Twitter. To this end, we propose a machine learning approach for the early detection of abusive third-party Twitter applications by analyzing their first few tweets. Specifically, we extract a variety of user-based (e.g., account age) and tweet-based features (e.g., retweets to tweets ratio) on the first-$k \in \{2, 3, ..., 25\}$ tweets to train a supervised machine learning classification model to distinguish between abusive and benign Twitter applications as early as possible. We implement and evaluate our machine learning approach before and after Twitter's recent countermeasures targeting third-party application abuse [9]. The evaluation shows that our machine learning approach is able to accurately detect abusive applications with 92.7% precision and 87.0% recall. We also show that our machine learning approach detects abusive applications months before Twitter does, during which time they are able to post tens of millions of tweets.

Key Contributions. We summarize our contributions as follows.

• *Longitudinal Measurement Study of Abusive Applications on Twitter.* We perform a longitudinal measurement study to establish a ground truth of abusive Twitter applications that captures diverse malicious and spamming behaviors. We showed that these abusive applications stay undetected for a long time posting tens of millions of tweets despite Twitter's ongoing efforts to detect them.

• *Machine Learning Approach for Early Detection.* We propose a machine learning approach to accurately and early detect these abusive applications by analyzing their first-$k \in \{2, 3, ..., 25\}$ tweets. We select $k = 7$ as a suitable trade-off between classification accuracy and early detection. Our machine learning model detects abusive applications with a precision of 92.7% and a recall of 87.0% using 10-fold cross validation as soon as they post their first seven tweets. We show that our model detects a large fraction of these abusive applications several months before Twitter detects them while they post tens of millions of tweets during this time period.

• *In The Wild Deployment.* The deployment of our machine learning model in the wild shows that attackers are still able to register and abuse third-party applications despite Twitter's new countermeasures [9]. We show that our machine learning model accurately detects these abusive third-party applications as soon as they post their first seven tweets while they evade detection by Twitter for a long time. Finally, we show that our machine learning model detects a large fraction of new abusive applications that are missed by Twitter's existing abuse detection systems.

We have disclosed our findings to Twitter's Site Integrity team, who is actively trying to mitigate abuse of third-party applications on their platform [9]. Our machine learning approach can complement Twitter's abuse detection systems for accurate and early detection of Twitter API abuse by third-party applications.

2 BACKGROUND

In this section, we first provide an overview of third-party application support on popular online social networks. We then discuss our threat model for third-party applications and the prevalence of their abuse on Twitter.

2.1 Third-Party Applications

Online social networks provide APIs to develop third-party applications such as games, entertainment, education, and utilities. To allow third-party application development, online social networks implement authorization frameworks such as OAuth [33]. For example, Twitter uses the OAuth 1.0a authorization framework [33], which enables third-party applications to gain access to Twitter's streaming and REST APIs as well as Twitter's Single Sign-On (SSO) service [12, 19]. When creating a new third-party application, developers have to specify a set of permissions required from users who would install the application on their accounts. OAuth supports both *read* and *write* permissions. The read permissions allow a third-party application to retrieve data (e.g., timeline tweets/posts, list of followers/friends) from a user's account. The write permissions allow a third-party application to perform write actions (e.g., posting tweets/posts, following users or liking pages) on a user's behalf. Popular online social networks such as Twitter and Facebook have millions of third-party applications that are regularly used by hundreds of millions of users [3, 23].

2.2 Threat Model

While third-party applications are widely used for benign purposes, unfortunately, they can also be exploited by attackers to compromise and orchestrate a large number of accounts for nefarious purposes. Prior work has reported several instances of widespread abuse of third-party applications for spreading spam and malware on online social networks [34, 52, 55, 57].

The typical modus operandi of attackers is as follows. Attackers register a new third-party application with the aim of installing it on as many fake/compromised accounts as possible. Attackers install the registered application on fake accounts that they themselves create or buy in bulk from underground marketplaces [35, 58]. Attackers may also compromise an account by tricking its user into installing the application. After installing the application on a sufficient number of accounts, attackers can use the access tokens [1] via the APIs to conduct malicious activities at scale.

The abuse of third-party applications has been shown time and again on popular online social networks [36, 50, 52, 55, 57]. For example, prior work reported the abuse of third-party applications to escalate the reputation of a target account by retweeting/liking/following from compromised/fake accounts on Twitter [52, 55] and Facebook [36]. Prior work has also reported the abuse of third-party applications to run spam or malware campaigns from compromised/fake accounts on Twitter [57] and Facebook [50].

In this paper, we specifically focus on investigating abuse by third-party applications on Twitter. It is noteworthy that Twitter

recently disclosed a large-scale abuse of third-party applications on their platform [10] in the aftermath of a congressional investigation into Russian interference in the 2016 U.S. election [7]. Specifically, Twitter announced that they removed hundreds of thousands of third-party applications that were abusing their API during 2017 through 2018 [10, 45]. Also noteworthy is Twitter's recently announced policy to vet new third-party applications at registration [9]. However, despite Twitter's existing detection systems and new countermeasures, we will show later that attackers continue to abuse third-party applications on Twitter to this day.

2.3 Abusive Twitter Applications

We refer to a third-party Twitter application as "abusive" if it violates Twitter's rules [20]. The violations of these rules mostly include malicious and spammy behaviors such as posting links to malicious content, aggressive following and un-following behavior, abusing reply or mention function, hijack trending topics or hashtags, duplicate updates, etc. Other violations of these rules, unrelated to malicious and spammy behaviors, include the unlawful use of Twitter platform such as illegitimate distribution of copyrighted or hacked material, graphical violence, and harassment.

It is challenging to manually establish the ground truth for third-party Twitter applications (e.g., manual detection of the violation of Twitter's rules) because of the scale and diversity of abusive behavior. To automatically get a comprehensive ground truth of abusive Twitter applications, we retrospectively check whether tweets by third-party applications are removed by users or Twitter's abuse detection systems due to the violation of their rules [20]. Prior work has exploited similar retrospective analysis of deleted tweets and suspended accounts to study spam and malware campaigns on Twitter by fake/compromised accounts [56, 57]. In particular, if all tweets by a Twitter application are removed, it is highly likely that the application is violating Twitter's rules and can be labeled as abusive. In addition to retrospective ground truth labeling of abusive Twitter applications, we further expand our ground truth out-of-band by including follower market applications that compromise accounts by incentivizing users to install the applications in exchange for "free followers" [13, 55]. We next explain our methodology to collect data of these abusive applications.

3 MEASURING ABUSIVE APPLICATIONS

In this section, we conduct a longitudinal measurement study of third-party Twitter applications to demonstrate that abusive applications are able to evade detection by Twitter's abuse detection systems for extended time periods while they post tens of millions of tweets during this time.

3.1 Data collection

We leverage Twitter's streaming API to gather tweets by different third-party Twitter applications and REST API to establish a ground truth of benign and abusive applications.

Streaming API. We rely on Twitter's streaming API to collect publicly available tweets. Twitter's streaming API offers a `sample` stream that returns 1% sample of all public tweets [48]. Each tweet contains the tweet's text and metadata, which includes timestamp, user's screen name, and the source field that contains the name of the application used to post the tweet. We collect 1.5 billion tweets by 112 million users from 456,987 applications from September 2016 to December 2017. We refer to this collection of tweets as the *Twitter sample dataset*.

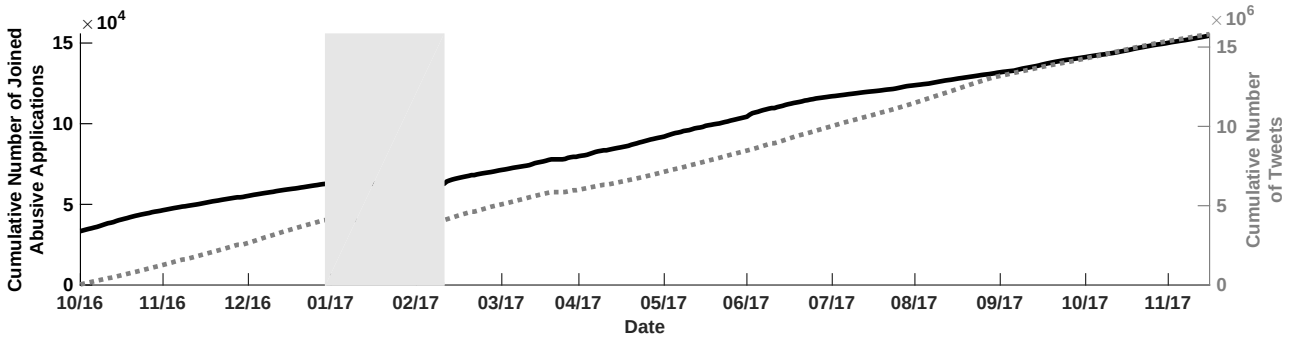
Retrospective Analysis of Removed Tweets. We retrospectively query the current status of tweets of all third-party applications in the Twitter sample dataset to check whether they are removed using Twitter's REST API. We provide sufficient time to Twitter's abuse detection systems to remove tweets by abusive third-party applications. Specifically, we start querying Twitter's REST API to check the deletion status of tweets in August 2018, which is at least 8 months apart from the tweets in the Twitter sample dataset. Note that we cannot query the deletion status of tweets for all applications due to the rate limits imposed by Twitter's REST API [16]. Hence, for each application, we select a random sample of at most 100 tweets whose deletion status is queried using Twitter's REST API. Since 87% of applications have less than or equal to 100 tweets, we sample tweets for only 13% applications and consider all tweets of the remaining applications. In total, we query the deletion status of 12 million tweets posted by 456,987 applications of which 36% tweets are removed. 49% of applications have no removed tweets while 37% applications have all of their tweets removed. To minimize false positives in our labeling, we conservatively label the 37% applications with all of their tweets removed as abusive. We next explain our methodology to identify and crawl follower markets to expand our ground truth of abusive applications.

Follower Markets. We query Google and Twitter search to identify follower markets. First, we search Google using keywords such as "free followers" and "increase followers". We manually analyze search results to identify popular follower markets. Second, we search Twitter using hashtags such as "followers" and "increaseFollowers". We manually analyze URLs in tweets to find follower markets. Using this methodology, we are able to identify 50 follower markets that ask Twitter users to install third-party applications in exchange for "free followers". Our eyeball analysis shows that abusive applications used by follower markets change over time. Therefore, we periodically crawl follower markets to extract the names of their abusive applications. To automate this process, we use Selenium WebDriver [18] to open each follower market website every 15 minutes. Upon clicking the sign-in button to install the application, we are redirected to Twitter's application authorization page. We extract the name of the abusive application, without installing it, from the authorization page. We crawl these 50 follower markets from September 2016 to August 2017 and identify names of 14,150 distinct abusive applications. Out of these 14,150, we find 6,437 abusive applications in our Twitter sample dataset.

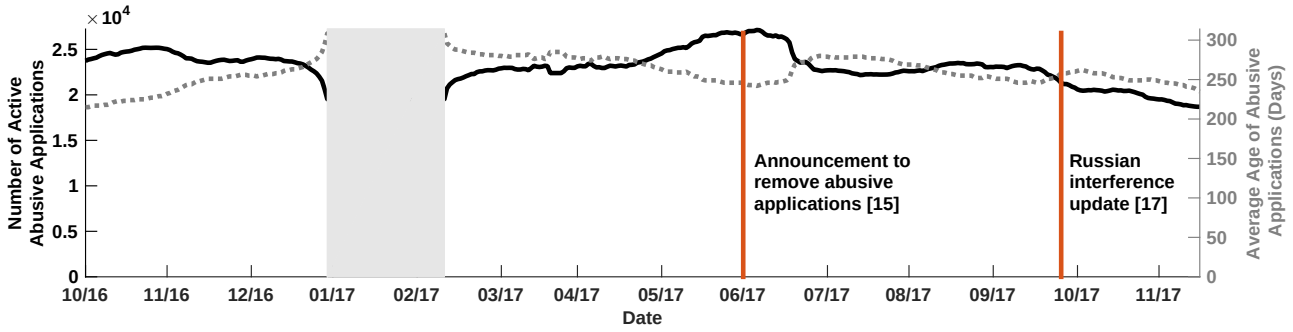
Table 1 summarizes the statistics of third-party Twitter applications in our Twitter sample dataset. In total, we are able to identify 168,227 distinct abusive applications through retrospective analysis of removed tweets and crawling follower markets.

3.2 Arms Race

We investigate the arms race between attackers continuously creating new abusive applications and Twitter trying to detect and remove them [10, 45]. Specifically, we study how quickly Twitter detects and removes abusive applications while they post tens of



(a) Cumulative number of daily joined abusive applications and their tweets count



(b) Count and average of daily active abusive applications

Figure 1: Illustration of the arms race between attackers and Twitter in registering and removing abusive Twitter applications. Attackers use a large pool of abusive applications to post tens of millions of tweets. While Twitter's existing countermeasures detect some abusive applications, attackers always have tens of thousands of active abusive applications that stay undetected for several months.

All Applications	Abusive (Retrospective)	Abusive (Follower markets)	Abusive (Combined)
456,987	167,013	6,437	168,227

Table 1: Summary of abusive applications identified from retrospective analysis of removed tweets and follower markets in our Twitter sample dataset.

millions of tweets observed in the Twitter sample dataset. Figure 1(a) plots the cumulative number of new abusive applications observed every day and the cumulative number of tweets posted by them [1]. We find that attackers use a large pool of applications to post abusive tweets. On average, attackers daily register 392 new abusive applications and post 41,705 tweets. In total, we observe more than 17 million tweets posted by 168,227 abusive applications. Since our Twitter sample dataset is limited to at most 1% sample of daily tweets, the actual number of tweets posted by abusive application is likely higher by roughly two orders of magnitude. Thus, we estimate the number of daily and total tweets posted by abusive applications to be in the order of millions and billions, respectively. Our estimates are close to the recent disclosure by Twitter that mentioned 2.2 billion tweets posted by abusive applications [10].

As Twitter detects and removes some of these abusive applications, we expect attackers to register new applications to make up for the removed applications. To evaluate Twitter's existing abuse detection systems, we plot the distribution of number of daily active abusive applications and their average age in Figure 1(b). Note that we estimate the *age* of an abusive application by calculating the difference between the first time and the last time the application appeared in Twitter sample dataset. We say that an application is removed by Twitter's abuse detection systems [5, 8, 56, 58], when it stops appearing in our dataset. Since there is no definitive way for us to know whether or not an application is removed by Twitter, we optimistically assume that these applications are removed by Twitter. We observe that attackers always have a substantial number of active abusive applications ranging between 18,687-27,132. These active applications stay undetected for a long time with an average age of more than six months.

While recently announced countermeasures by Twitter detect some abusive applications, we note that a vast majority of abusive applications still go undetected for a long time period. Specifically, Twitter announced plans to implement new machine learning based approaches to detect and remove abusive applications in June 2017 [5]. In Figure 1(b), we observe a sharp but small decline in the number of active abusive applications. It is interesting to note that attackers seem to adapt to these new countermeasure and the number of active applications stabilizes by August 2017. We observe another decline in the number of active applications after Twitter announced additional countermeasures against abusive third-party applications in September 2017 [7]. However, we note that a vast

[1]Our data collection stopped during the time period represented by the grey shaded region due to an error in the data collection script.

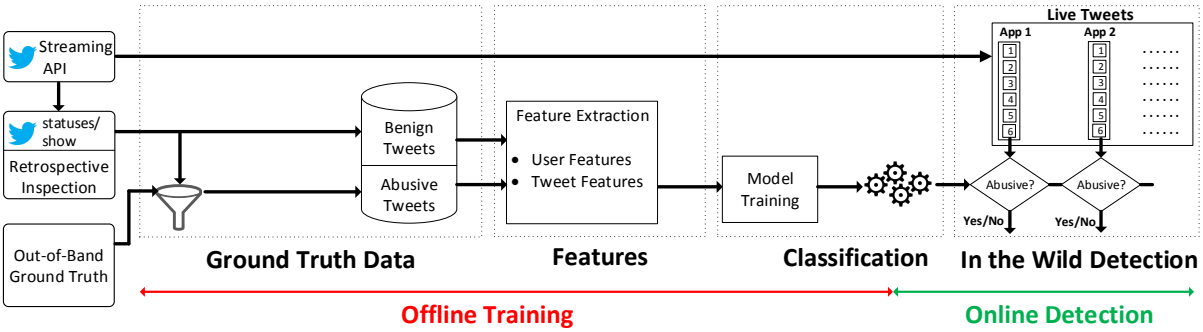


Figure 2: Our proposed approach for early detection of abusive third-party applications on Twitter. In the offline phase, we analyze the first-k tweets of each application to extract user-based and tweet-based features. We then train a supervised machine learning model to classify benign and abusive applications. In the online phase, we use the trained model to detect abusive applications by analyzing their first-k tweets from Twitter's streaming API.

majority of the abusive application still remain active despite Twitter's countermeasures.

It is noteworthy that the applications detected during these sharp declines in Figure 1(b) are relatively new because the average age of active applications increased after the decline. This shows that while Twitter detects some abusive applications, a large number of long-lived abusive applications remain undetected. Specifically, 5,404 abusive applications remain undetected during the 16 month period and are able to post 2.9 million tweets.

Takeaway: Our results indicates an ongoing arms race between attackers and Twitter on the registration and removal of abusive Twitter applications. Essentially, attackers are able to use a large pool of abusive applications to post tens of millions of tweets while being resilient to Twitter's countermeasures. While Twitter's existing countermeasures detect some abusive applications, a large number of abusive applications stay undetected for a long time. As we discuss next, we propose a machine learning approach for the early detection of abusive applications.

4 PROPOSED APPROACH

Since abusive third-party Twitter applications are able to evade detection for a long time, we are interested in detecting these abusive applications as early as possible before they are able to post many tweets. In this section, we present a machine learning approach for early detection of abusive third-party Twitter applications. Figure 2 provides an overview of our approach. In the offline phase, we train a supervised machine learning model that analyzes the first-k tweets of an application to detect abusive applications. More specifically, we extract a variety of user-based and tweet-based features to distinguish between benign and abusive applications. Using a labeled repository of tweets for benign and abusive applications, we then train a supervised machine learning classifier to detect abusive applications. In the online phase, we use the trained supervised machine learning model to detect abusive applications *in the wild* by analyzing their first-k tweets from Twitter's streaming API.

4.1 Ground Truth

Next, we explain our method to establish the ground truth for benign and abusive applications in our Twitter sample dataset. Recall from Section 3.1 that we may strictly label an application as abusive if all of its tweets are removed or benign if none of the tweets are removed. However, this strict definition would result in mislabeling many abusive and benign applications. For instance, a user may remove tweets posted through a benign application due to spelling or grammatical mistakes [22]. Similarly, a user may remove tweets posted through an abusive application after recovering a compromised account [14, 57]. Moreover, Twitter's abuse detection systems may remove a subset of tweets by an abusive application, unrelated to detection of the abusive application [6]. Therefore, we need to relax our labeling criterion from all-or-nothing. To this end, we define two thresholds, α and β, to label an application as abusive or benign, respectively. We label an application as abusive if the percentage of removed tweets is more/less than α/β. On one extreme, we select the value of $\alpha = 90\%$ as 37% of applications have at least 90% of their tweets removed. On the other extreme, we select $\beta = 30\%$ as 58% of applications have less than 30% removed tweets. To reaffirm this selection of β, note that the percentage of removed tweets for several popular benign applications (e.g., Twitter for iPhone, Twitter for Android, Twitter Web Client) is less than 30%. To conclude, we are able to label 95% applications as benign or abusive using 90%-30% α and β selections. Note that we filter applications with only one user/tweet representing less than 2% tweets in our Twitter sample dataset. Overall, our ground truth contains 19,343 benign and 24,588 abusive applications in our Twitter sample dataset.

Next, we present case studies of a select spam and malicious campaigns. We manually analyze the content of a sample of labeled abusive applications to identify these campaigns.

Scam installs. We find several known malicious campaigns that deceive users into installing their abusive applications to compromise accounts and post spam tweets from these accounts. For example, we identify a malicious campaign that claims to inform users who visited their timeline [26, 47]. This is a deceiving claim to trick users into installing abusive applications since Twitter does not provide timeline visit information to third-party applications. As another example, attackers register applications with names that impersonate Twitter (e.g., `Twitter Age Confirmation` and `Twitter Age Verification`) to trick users into installing their abusive applications [4]. We find over 250 abusive applications that are part of such malicious campaigns.

Phone spam campaign. We find a malicious campaign promoting phone spam [39]. In this campaign, spammers mislead victims by making false promises and expect users to contact them on listed phone numbers in the posted tweets. We find 47 abusive applications that are part of this spam campaign.

Astroturfing campaigns. We find several astroturfing campaigns that exploit abusive applications to run their operations. Some examples of these campaigns provide fake followers [55] and retweets [52]. We identify thousands of abusive applications that participate in such astroturfing camapaigns.

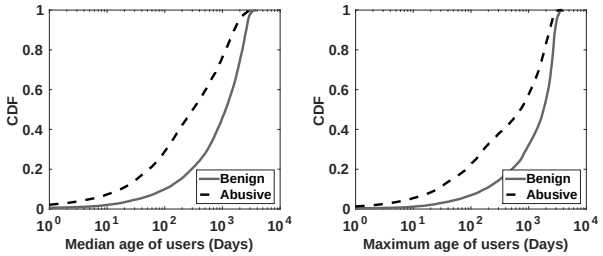
4.2 Features

We extract a comprehensive set of features to capture distinguishing characteristics of benign and abusive applications. Our feature extraction has two key differences compared to prior work. First, while prior research computed per-tweet features to detect spam/malicious tweets or per-user features to detect fake/compromised user accounts, we compute features on a per-application basis to directly detect abusive applications. Second, unlike prior research, we compute features from the first-k tweets of each application for early detection of abusive applications.

We compute a variety of user-based and tweet-based features to detect abusive applications. For user-based features we focus on following characteristics of users: (1) **number of followers*, (2) **number of followings*, (3) average number of followers to followings ratio, (4) ratio of users with default images set as profile, (5) ratio of verified users, (6) **age of user accounts*, (7) **number of tweets*, and (8) average ratio of total tweets to age of user accounts . For tweet-based features we focus on following characteristics of tweets: (1) ***number of user mentions*, (2) ***number of hashtags*, (3) percentage of tweets with hashtags, (4) percentage of unique hashtags, (5) entropy of hashtags, (6) average of retweet-to-tweet ratio, (7) entropy of URLs, (8) percentage of URLs, and (9) percentage of unique URLs. For many of the user-based and tweet-based features, we compute various summary statistics for italicized features across all tweets of an application. Features with * represent mean, median, minimum, and maximum whereas features with ** represent mean, median, minimum, maximum, and standard deviation. In total, we extract 38 user-based and tweet-based features. We next analyze the effectiveness of a select subset of these features in distinguishing between abusive and benign applications.

Account Age. Figure 3(a) plots the distribution of the median age of user accounts of abusive and benign applications. We note that the user accounts of benign applications are significantly older than those of abusive applications. More specifically, 68% abusive applications have median user account age of two years or less. In contrast, 37% benign applications have median user account age of two years or less. We surmise that the median age of user accounts of abusive applications is low because attackers continuously create fresh user accounts which are, sooner or later, suspended due to violation of Twitter rules [56].

Figure 3(b) plots the distribution of maximum user account age for abusive and benign applications. We note that a large fraction of benign applications has at least one account that is very old. Specifically, more than 75% of benign applications have at least one user account aged two years or more. In contrast, more than 50% of



(a) Average (b) Maximum

Figure 3: We observe that the user accounts of benign applications are older than the user accounts of abusive applications.

abusive applications have no user account aged two years or more. While attackers can obfuscate maximum user account age feature by adding an aged user account, the median age of user accounts is more robust to obfuscation because it relies on the whole user account population of an application.

Retweet-to-tweet Ratio. Figure 4 plots the distribution of retweet-to-tweet ratio of abusive and benign applications. We note that a large fraction of abusive applications posts only retweets while this behavior is quite uncommon among benign applications. Specifically, 32% abusive applications have retweet-to-tweet ratio of 1 whereas only 5% benign application have retweet-to-tweet ratio of 1. Such abusive applications are likely being used to artificially boost reputation of tweets [52]. Attackers can try to post original tweets to obfuscate retweet-to-tweet ratio feature. To generate original tweets, attackers can use duplicate content but this will also be detected due to the violation of Twitter rules. Attackers can also recruit crowdturfing workers [44] to generate organic content to obfuscate ratio of retweet-to-tweet feature but it may prohibitively increase their cost for large-scale spam operation [30].

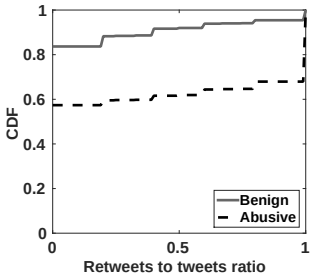


Figure 4: We observe that a large fraction of abusive applications only post retweets while this behavior is quite uncommon among benign applications.

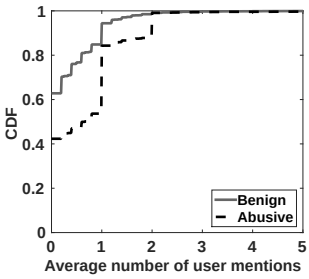


Figure 5: We observe that abusive applications mention more users in their tweets as compared to benign applications.

Number of User Mentions. Figure 5 plots the distribution of average number of user mentions in tweets posted by abusive and benign applications. It is interesting to note that abusive applications mention more users than benign applications. Specifically, 46% abusive applications have an average of one or more user mentions while only 5% benign applications have an average of one or more user mentions. Abusive applications mention more users in tweets to either lure other victims into installing their applications or increase the reach of their tweets. Attackers can try to reduce

or stop mentioning users altogether to manipulate their average number of mentions; however, it would negatively impact their ability to reach to other users.

4.3 Classification

We leverage previously discussed user-based and tweet-based features to train a supervised machine learning model to classify an application as abusive or benign. For classification, we tried several classification algorithms using Python's scikit library and ended up selecting the Random Forest classifier because it outperformed other algorithms. For training the model, we use the ground truth of 19,343 benign and 24,588 abusive applications. We first use 10-fold cross-validation to evaluate the accuracy of our trained Random Forest classification model. We then use our trained Random Forest classification model to detect abusive applications in the wild on new data collected from Twitter's streaming API.

5 EVALUATION

In this section, we evaluate the effectiveness of our machine learning approach in detecting abusive applications as early as possible. First, we use cross-validation to study the classification accuracy for varying values of first-k tweets. We show that our machine learning approach is able to detect abusive applications with very high accuracy several weeks before they are detected by Twitter. Second, we employ our machine learning approach to detect abusive applications based on their first few tweets on a new Twitter dataset, which is collected after Twitter's recently announced countermeasures were implemented. We show that attackers still register new abusive applications that go undetected by Twitter while our machine learning approach detects them quite early.

5.1 Cross-Validation

Early Detection. We train and test a Random Forest classifier on varying values of first-$k \in \{2, 3, ..., 25\}$ tweets using 10-fold cross-validation. For each value of k, we sample from the set of abusive and benign applications with at least k tweets. Since number of benign applications are slightly less than abusive applications, we randomly sample benign applications for each value of k to match the number of abusive applications to create a balanced dataset for cross-validation. We train and test our model using 100 random samples of benign applications for each value of k and report averages and standard deviations of precision and recall metrics. Figure 6 plots precision and recall as a function of k. We observe the best average precision and recall of 94.7% and 89.7% at $k = 25$, respectively. Recall that our objective is to accurately detect abusive applications as early as possible. We observe that precision and recall improve as k increase but they start to plateau beyond $k = 7$ in Figure 6(a). Specifically, precision increases from 90.9% for $k = 2$ to 92.7% for $k = 7$ and recall increases from 83.2% for $k = 2$ to 87.0% for $k = 7$. Therefore, we select $k = 7$ as a suitable tradeoff between early detection and classification accuracy.

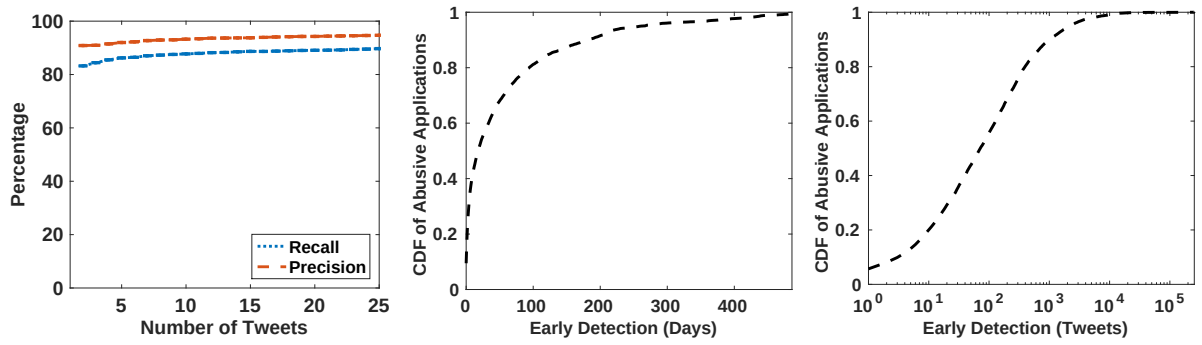
We quantify the early detection of abusive applications correctly detected by our model for $k = 7$ in terms of days and tweets. Early detection in terms of days is defined as the difference between the estimated age of application (defined in Section 3.2) and the age of application when they are detected by our model. Early detection in terms of tweets is defined as the difference between the total number of tweets posted by the applications and the number of tweets posted by applications when they are detected by our model.

Figure 6(b) shows that our trained model is able to detect abusive applications weeks and sometimes months before Twitter does so. Specifically, 42% abusive applications are detected at least a month earlier by our model whereas 21% abusive applications are detected at least 3 months earlier by our model. It is noteworthy that our model has detected 60 abusive applications on the first day of their appearance that otherwise remain undetected throughout the data collection period of 16 months. Figure 6(c) shows that our trained model detects abusive applications before they continue posting hundreds of millions of tweets. Specifically, 45% abusive applications posted 100 or more tweets where 10% abusive applications posted 1000 or more tweets after detection by our model. In total, all abusive applications posted 9,146,439 tweets after detection by our model. Since our Twitter sample dataset is limited to at most 1% sample of daily tweets, the actual number of tweets posted by abusive applications is likely higher by roughly two orders of magnitude. Thus, we estimate that abusive applications posted tweets in the order of hundreds of millions after detection by our model.

We manually analyze a small sample of false positives i.e., benign applications incorrectly classified as abusive by our machine learning model during cross-validation. Our manual analysis uncovers that several applications labeled as benign in our ground truth are in fact abusive. This indicates that our precision is actually more than our estimate reported in Figure 6(a). We surmise that activities of many abusive applications remain undetected by Twitter's existing abuse detection systems during our data collection period; hence, these abusive applications are incorrectly labeled as benign in our ground truth. This also shows that our ground-truth is a conservative estimate of abusive applications. Despite being a conservative estimate, as discussed in Section 4.1, we argue that our ground truth captures a diverse set of abusive behaviors.

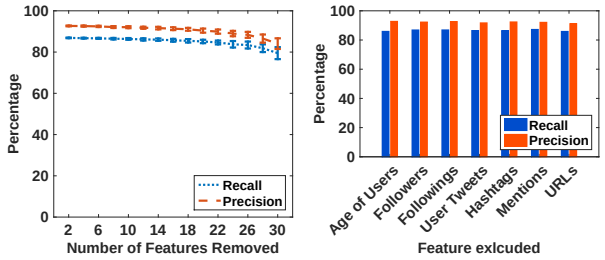
We also acknowledge that abusive applications that post less than 7 tweets will not be detected by our model. However, we argue that these low-activity applications do not pose a significant threat due to their low tweet volume. In other words, while posting fewer tweets allows abusive applications to go undetected, it also limits their ability to conduct abuse on a large scale. Moreover, if needed, we can detect these low-activity abusive applications using our machine learning model trained for smaller values of k with reasonably high precision and recall.

Feature Ablation. We perform an ablation experiment to understand the impact of removing features on the classification accuracy of our model in Figure 7. We randomly remove a varying number of features from 0 to 30 to train and test our model. We repeat this experiment 100 times for each number of removed features. Figure 7(a) shows that precision and recall decreases as we remove more features but this decrease is not substantial. Specifically, the average precision and recall decrease by only 8.6% and 7.4%, respectively, when the number of removed features increase from 2 to 30. We perform another ablation experiment to understand the contribution of individual feature sets of user-based and tweet-based features. Specifically, we remove one feature set at a time to train and test our model. Figure 7(b) shows that the classification accuracy does not



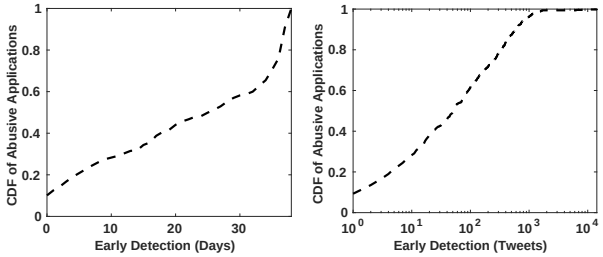
(a) Ten-fold cross validation (b) Early Detection at $k = 7$ (Days) (c) Early Detection at $k = 7$ (Tweets)

Figure 6: We achieve best precision and recall of 94.7% and 89.7%, respectively, at $k = 25$. We select $k = 7$ as a suitable trade-off between early detection and classification accuracy, where we detect 42% of abusive applications at least a month before Twitter during which time these abusive applications post millions of tweets.



(a) Removing features randomly (b) Excluding a feature set

Figure 7: Ablation experiment at $k = 7$ shows that our features are resilient to obfuscation attempts by attackers. More specifically, (a) shows that our feature set is resilient to obfuscation attempts against a combination of features and (b) shows that our classification results are not dependent on individual user-based and tweet-based features.



(a) Early Detection In the Wild (Days) (b) Early Detection In the Wild (Tweets)

Figure 8: The deployment of our machine learning approach in the wild shows that attackers continue to register new applications that go undetected for a long time despite Twitter's new countermeasures while our machine learning approach detects them several weeks before Twitter by analyzing their first seven tweets.

significantly degrade without any individual feature set. The lowest precision of 91.2% and recall of 85.9% is observed when we remove the *URLs* feature set which includes entropy of URLs, unique percentage of URLs, and ratio of URLs to tweets. We conclude that our trained machine learning model is resilient against attempts to obfuscate a specific feature set or a combination of features.

5.2 In the Wild Detection

Next, we show that attackers are able to register new applications and abuse them despite Twitter's most recent countermeasures against them. We also show that our machine learning approach can detect these new abusive applications that are missed by Twitter's countermeasures.

Overview of Twitter's new countermeasures. Recall from Section 3.2 that Twitter has several countermeasures in place to detect and remove abusive applications [5, 10]. However, these countermeasures have not sufficiently deterred attackers from abusing third-party applications. To mitigate the abuse of third-party applications, Twitter recently enforced several new countermeasures [9]. Specifically, Twitter introduced a new policy to review use cases and check policy compliance of new third-party Twitter applications at registration to mitigate abuse. Twitter also introduced new rate limits on the use of POST endpoints (e.g., tweet/retweet, like/follow). Finally, Twitter introduced new ways for users to detect and report abusive applications. Since these countermeasures were implemented after our data collection period that ended in December 2017, we are interested in studying whether they are effective in mitigating abuse of third-party applications. More specifically, we want to find out whether attackers can still register new applications and evade detection and whether our machine learning approach can accurately detect them as early as possible.

In the Wild Deployment. To study the effectiveness of Twitter's newly introduced countermeasures, we use Twitter's streaming API to collect a new tweet dataset during September-October 2018 which is after Twitter's new countermeasures went into effect. To focus on new high-activity applications, we filter the applications that also appeared in our older dataset and those with fewer than seven tweets. We use Twitter's REST API to retrospectively query the deletion status of tweets of 2,225 new applications. Using the approach outlined in Section 3, we find that Twitter removed 532 new applications. In other words, Twitter removed about a quarter (24%) of new applications, which were able to bypass vetting at registration and post hundreds of thousands of tweets before being eventually detected by Twitter.

Next, we use our machine learning model trained on Twitter sample dataset to classify the 2,225 new applications. Our model is able to detect 93% (495 out of 532) of the applications removed by Twitter. It is noteworthy that our model detects these abusive applications that evade detection for weeks while posting hundreds of thousands of tweets. Figure 8 shows that 40% of these abusive applications are detected by our model at least a month before Twitter's new countermeasures. Figure 6(c) shows that 62% of these abusive applications posted at least 100 tweets after detection by our model, which uses only the first-7 tweets. In total, these abusive applications posted 239,485 tweets after detection by our model and before being detected by Twitter's countermeasures. Note that we estimate that abusive applications posted tweets in the order of tens of millions after detection by our model since our tweet collection from Twitter's streaming API is an approximately 1% sample of all tweets. In addition to these 495 abusive applications, our model also classifies 390 new abusive applications that go undetected by Twitter. We manually inspect 10% of these applications sorted in the descending order of our model's detection probabilities. We find that 95% of all the inspected applications are clearly abusive. Among the abusive applications missed by Twitter, we find applications that are part of various spam campaigns such as astroturfing, and profile visit scam [26] (also discussed in Section 4.1).

Takeaway. The deployment of our machine learning approach in the wild shows that attackers continue to register new third-party application and post tens of millions of tweets despite Twitter's new countermeasures. We also show that our machine learning approach accurately and early detect these abusive applications that go undetected by Twitter. We believe that our proposed machine learning approach can complement Twitter's existing efforts for accurate and early detection of abusive third-party applications.

5.3 Limitations & Discussion

Next, we address some limitations of our machine learning approach, discuss its deployment to complement Twitter's existing abuse detection systems, as well as ideas for future extensions.

Evasion and Countermeasures. Like any machine learning based system, our approach is susceptible to evasion if attackers become aware of the details of our machine learning framework. Attackers can attempt to manipulate the features used by our machine learning model to evade detection. However, as demonstrated in Figure 7, our approach is resilient to obfuscation attempts against a particular feature or even different combinations of user-based and tweet-based features. Hence, attackers would need to manipulate multiple feature sets, some of which would likely be cost prohibitive for them. For example, our machine learning model captures the pattern that average account age for abusive applications is less as compared to benign applications. To obfuscate average account age, attackers would need to either discard newer fake/compromised accounts limiting the scale of their operations or purchase "aged" accounts that are reportedly much more expensive than newly created accounts [58]. Even if attackers are able to successfully manipulate multiple features and evade detection, we can periodically retrain our machine learning model using new ground truth to capture the evolving behavior of abusive applications. We can further design new features to better capture the changing behavior of abusive applications since our machine learning framework is readily amenable to the addition of new features as needed. Finally, after becoming aware of our early detection system, attackers can mimic the behavior of benign applications initially and delay abusive activities to evade early detection by our machine learning approach. To address this issue, our machine learning approach can be adapted to continuously monitor an application's tweets in a streaming fashion. Since our work focuses on the early detection of abusive applications, the implementation and evaluation of continuous application monitoring is outside the scope of this paper.

Low-volume Abusive Applications. Our machine learning approach will not be able to detect low-volume abusive applications that post only a few tweets because our machine learning model needs at least seven tweets for detection. First, we surmise that attacker could deliberately post very few tweets to evade detection by our machine learning approach. However, this would significantly reduce the scale of abusive activities, especially given Twitter's revamped application registration process that limits automation [9]. Second, it is also likely that Twitter's existing abuse detection systems [5, 8, 9] are able to detect many abusive applications before they post seven tweets needed by our machine learning approach. In other words, we only observe sophisticated abusive applications via Twitter's streaming API that bypass Twitter's abuse detection systems. Thus, we believe that our machine learning approach nicely complements Twitter's existing countermeasures by early detection of abusive applications that otherwise remain undetected by Twitter.

Handling False Positives While our machine learning approach detects abusive applications with a seemingly non-negligible false positive rate of approximately 6%, we argue that it is sufficiently low to be practical at Twitter's scale. Twitter can employ a review system that allows developers to submit an appeal to review incorrectly flagged applications. Note that Twitter already employs a review process to vet developers [9] which can be extended to review appeals for potentially mislabeled abusive applications. Recall that we observed 2,225 applications through Twitter's streaming API with sufficient activity over the duration of 39 days in September-October 2018. Thus, we argue that a false positive rate of 6% translates into a very manageable 5 false positives per day.

Third-Party Application Abuse on Other Online Social Networks. There have been several high-profile reports of third-party application abuse on popular online social networks including Twitter [10], Facebook [29, 36], and Google+ [46]. Our machine learning approach provides a footprint for other popular online social networks for accurate and early detection of third-party application abuse. Unfortunately, we are unable to investigate abuse of third-party applications on other online social networks due to lack of publicly available data. Nonetheless, online social networks operators can replicate our proposed machine learning framework shown in Figure 2. Most of our features (e.g., age of user accounts, number of followers, number of followings) can be directly translated to other online social networks such as Facebook, Instagram, and Snapchat. Moreover, online social network operators can further design new features based on their proprietary data to better capture behavior of abusive applications on their platform. Our

machine learning framework is readily amenable to addition of new features.

6 RELATED WORK

We divide prior work into three categories. First, we discuss prior work on the abuse of third-party applications on Twitter and Facebook. Second, we discuss prior work on detection of fake or compromised accounts on Twitter. Third, we discuss prior work on the measurement and detection of spam and malicious activities on Twitter.

Third-Party Application Abuse on Twitter. Prior work has reported on the exploitation of third-party Twitter applications for nefarious purposes [31, 34, 52, 55, 57]. Chu et al. [31] reported the abuse of third-party applications by bots on Twitter. Stringhini et al. [55] also reported the abuse of third-party applications by Twitter follower markets. Egele et al. [34] and Thomas et al. [57] each independently reported more than 9,000 abusive third-party applications being used to spread spam on Twitter. Prior research has also used third-party application information to aid detection of compromised accounts [34] and spam [52] on Twitter. Twitter tries to block applications used by attackers; who periodically register new applications to avoid the shutdown. While Twitter has recently removed more than 240,000 abusive applications [8], as we showed in this paper, the cat-and-mouse game between Twitter trying to detect abusive applications and attackers continuously creating new applications is still ongoing. To the best of our knowledge, we are the first to attempt to directly detect abusive third-party Twitter applications.

Third-Party Application Abuse on Facebook. Prior work has also reported on the exploitation of third-party Facebook applications for nefarious purposes [36, 50]. Most closely related to our work is that of Rahman et al. [50]. The authors proposed a machine learning approach to detect abusive third-party applications on Facebook. They found a collusion network of 5,307 abusive Facebook applications that promote each other. A key difference between our work and theirs is that we focus on *early* detection of abusive applications but their detection is post hoc. More recently, Farooqi et al. [36] reported that spammers exploit legitimate third-party Facebook applications to provide fake likes and comments. While they employed temporal clustering and IP rate limits to mitigate the abuse of legitimate third-party Facebook applications, we propose a supervised machine learning approach for early detection of abusive third-party Twitter applications.

Fake/Compromised Accounts. There is a large body of prior work on the detection of fake or compromised accounts in online social networks [21, 24, 25, 28, 34, 40, 42, 43, 49, 53, 54, 59, 61, 62]. First, researchers have leveraged account information such as demographics and number of followers/friends to detect fake or compromised accounts [21, 25, 43, 53, 54]. For example, Stringhini et al. [54] trained machine learning models using account features such as number of friends and messages to detect spamming accounts on Facebook and Twitter. Second, researchers have leveraged social connectivity information to detect fake or compromised accounts [28, 61, 62]. For example, Cai and Jermaine [28] used the latent community model to detect Sybil communities that are linked relatively loosely with the rest of the social graph. Third, researchers have leveraged activity patterns to detect fake or compromised accounts [24, 34, 40, 42, 59]. For example, Egele et al. [34] detected compromised accounts by identifying synchronized changes in account behavior within a short time period. It has been shown time and again that more sophisticated attackers can mimic account information, social connectivity, and activity patterns of real accounts to evade detection by such approaches. While our work is complementary to prior research on the detection of fake or compromised accounts, we believe that it may be more effective to directly target the mechanisms used by attackers to orchestrate fake or compromised accounts. Therefore, we focus on detecting abusive third-party Twitter applications that are used by attackers to control fake or compromised accounts.

Spam/Malware Activities. A large body of prior work has focused on the detection and characterization of spam activities on online social networks [37, 38, 49, 52, 55]. Grier et al. [38] characterized different types of spam activities such as phishing, malware, and scam on Twitter. Gao et al. [37] clustered user activity based on URL and textual similarity to detect and characterize spam campaigns on Facebook. Stringhini et al. [55] trained a machine learning model using activity based features such as the rate of change of followers/followings to detect Twitter follower market customers. Song et al. [52] trained a machine learning model using activity based features such as re-tweet time distribution to detect crowdturfing targets (e.g., tweets) on Twitter. Nilizadeh et al. [49] distinguished between benign and spam activities (e.g., tweets) based on their dissemination patterns in communities that share some topics of interest. While prior research has focused on the detection and characterization of malicious activities in online social networks, we aim to detect abusive applications as early as possible to minimize their malicious activities.

7 CONCLUSION

In this paper, we presented a machine learning based approach for accurate and early detection of abusive third-party applications on Twitter. First, we performed a longitudinal measurement study of abusive third-party Twitter applications over a period of 16 months. Our measurements demonstrated an ongoing arms race between attackers registering and abusing new applications and Twitter actively trying to detect and remove them. Second, since abusive applications go undetected for several months, we proposed a machine learning approach for accurate and early detection of abusive applications by analyzing their first few tweets. The evaluation showed that our machine learning approach can accurately detect abusive applications several months before Twitter's existing abuse detection systems, preventing these abusive applications from posting millions of spam and malicious tweets. Third, the deployment of our machine learning model in the wild showed that attackers continue to abuse third-party applications despite new countermeasures enforced by Twitter. Our machine learning approach can complement Twitter's existing abuse detection systems for accurate and early detection of abusive third-party applications.

ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation under grant numbers 1715152 and 1815131.

REFERENCES

[1] OAuth Core 1.0 Revision A. https://oauth.net/core/1.0a/, June 2009.
[2] One Million Registered Twitter Apps. https://blog.twitter.com/official/en_us/a/2011/one-million-registered-twitter-apps.html, July 2011.
[3] Facebook Apps Leaderboard - AppData. https://web.archive.org/web/20161022132414/http://www.appdata.com/leaderboard/apps, 2016.
[4] Malicious Twitter Applications and Abuse of the Twitter API. https://www.slickrockweb.com/malicious-twitter-applications.php, June 2017.
[5] Our approach to bots and misinformation. https://blog.twitter.com/official/en_us/topics/company/2017/Our-Approach-Bots-Misinformation.html, June 2017.
[6] Twitter is sweeping out fake accounts like never before, putting user growth at risk. https://www.washingtonpost.com/technology/2018/07/06/twitter-is-sweeping-out-fake-accounts-like-never-before-putting-user-growth-risk/, June 2017.
[7] Update: Russian interference in the 2016 US presidential election. https://blog.twitter.com/official/en_us/topics/company/2017/Update-Russian-Interference-in-2016--Election-Bots-and-Misinformation.html, Sept. 2017.
[8] How Twitter is fighting spam and malicious automation. https://blog.twitter.com/official/en_us/topics/company/2018/how-twitter-is-fighting-spam-and-malicious-automation.html, July 2018.
[9] New developer requirements to protect our platform. https://blog.twitter.com/developer/en_us/topics/tools/2018/new-developer-requirements-to-protect-our-platform.html, July 2018.
[10] Update on Twitter's Review of the 2016 U.S. Election. https://blog.twitter.com/official/en_us/topics/company/2018/2016-election-update.html, Jan. 2018.
[11] Application Permission Model - Twitter Developers. https://dev.twitter.com/oauth/overview/application-permission-model, 2019.
[12] Docs - Twitter Developers. https://dev.twitter.com/rest/public, 2019.
[13] "Free followers" apps. https://support.twitter.com/articles/20171936, 2019.
[14] Help with my compromised account. https://help.twitter.com/en/safety-and-security/twitter-account-compromised, 2019.
[15] OAuth with the Twitter API - Twitter Developers. https://dev.twitter.com/oauth, 2019.
[16] Rate limits - Twitter Developers. https://developer.twitter.com/en/docs/basics/rate-limits, 2019.
[17] Sample realtime Tweets - Twitter Developers. https://developer.twitter.com/en/docs/tweets/sample-realtime/overview/GET_statuse_sample, 2019.
[18] Selenium - Web Browser Automation. http://www.seleniumhq.org/, 2019.
[19] Sign in with Twitter. https://dev.twitter.com/web/sign-in, 2019.
[20] The Twitter Rules. https://support.twitter.com/articles/18311, 2019.
[21] A. Aggarwal and P. Kumaraguru. What they do in shadows: Twitter underground follower market. In *13th IEEE Annual Conference on Privacy, Security and Trust (PST)*, 2015.
[22] H. Almuhimedi, S. Wilson, B. Liu, N. Sadeh, and A. Acquisti. Tweets are forever: a large-scale quantitative analysis of deleted tweets. In *ACM CSCW*, 2013.
[23] S. Bennett. 14 percent Use Third Party Apps for Twitter. http://www.adweek.com/digital/twitter-third-party-ads/, Adweek, Aug. 2014.
[24] A. Beutel, W. Xu, Wenkatesan, Chirstopher, and Christos. CopyCatch: Stopping Group Attacks by Spotting Lockstep Behavior in Social Networks. In *WWW*, 2013.
[25] Y. Boshmaf, D. Logothetis, G. Siganos, J. Leria, J. Lorenzo, M. Ripeanu, and K. Beznosov. Integro: Leveraging Victim Prediction for Robust Fake Account Detection in OSNs. In *NDSS*, 2015.
[26] C. Boyd. "Who visits your Twitter profile" spam app brings week of chaos. https://blog.malwarebytes.com/cybercrime/2018/01/who-visits-your-twitter-profile-spam-app- brings-week-of-chaos/, Jan. 2018.
[27] R. Brandom. The Google Docs spam attacks played off Google's most fundamental weakness. https://www.theverge.com/2017/5/4/15544608/google-docs-spam-phishing-email-hack, The Verge, May 2017.
[28] Z. Ca and C. Jermaine. The Latent Community Model for Detecting Sybil Attacks in Social Networks. In *NDSS*, 2012.
[29] C. Cadwalladr and E. Graham-Harrison. Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach. https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-influence-us-election, The Guardian, 2018.
[30] N. Christin. Security Economics: From Game Theory to Field Measurements. https://www.sigmetrics.org/sigmetrics2017/Christin2017_SIGMETRICS_tutorial.pdf SIGMETRICS Tutorial, 2017.
[31] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia. Who is Tweeting on Twitter: Human, Bot, or Cyborg? In *Annual Computer Security Applications Conference (ACSAC)*, 2010.
[32] E. D. Cristofaro, A. Friedman, G. Jourjon, M. A. Kaafar, and M. Z. Shafiq. Paying for Likes?: Understanding Facebook Like Fraud Using Honeypots. In *ACM Internet Measurement Conference (IMC)*, 2014.
[33] E. E. Hammer-Lahav. The OAuth 1.0 Protocol. IETF RFC 5849, April 2010.
[34] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna. COMPA: Detecting Compromised Accounts on Social Networks. In *NDSS*, 2013.
[35] S. Farooqi, M. Ikram, E. D. Cristofaro, A. Friedman, G. Jourjon, M. Kaafar, Z. Shafiq, and F. Zaffar. Characterizing Key Stakeholders in an Online Black-Hat Marketplace. In *IEEE/APWG Symposium on Electronic Crime Research (eCrime)*, 2017.
[36] S. Farooqi, F. Zaffar, N. Leontiadis, and Z. Shafiq. Measuring and Mitigating OAuth Access Token Abuse by Collusion Networks . In *ACM IMC*, 2017.
[37] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Zhao. Detecting and Characterizing Social Spam Campaigns. In *ACM IMC*, 2010.
[38] C. Grier, K. Thomas, V. Paxson, and M. Zhang. @spam: The Underground on 140 Characters or Less. In *ACM CCS*, 2010.
[39] S. Gupta, A. Khattar, A. Gogia, P. Kumaraguru, and T. Chakraborty. Collective Classification of Spam Campaigners on Twitter: A Hierarchical Meta-Path Based Approach. In *WWW*, 2017.
[40] M. Ikram, L. Onwuzurike, S. Farooqi, E. D. Cristofaro, A. Friedman, G. Jourjon, M. A. Kaafar, and Z. Shafiq. Measuring, Characterizing, and Detecting Facebook Like Farms. *ACM Transactions on Privacy and Security*, 20(4):1–28, 2017.
[41] N. Jagpal, E. Dingle, J.-P. Gravel, P. Mavrommatis, N. Provos, M. Rajab, and K. Thoma. Trends and Lessons from Three Years Fighting Malicious Extensions. In *USENIX Security*, 2015.
[42] M. Jian, P. Cui, A. Beutel, C. Faloutsos, and S. Yang. CatchSync: catching synchronized behavior in large directed graphs. In *ACM KDD*, 2014.
[43] K. Lee, J. Caverlee, and S. Webb. Uncovering social spammers: social honeypots + machine learning. In *ACM SIGIR*, 2010.
[44] K. Lee, P. Tamilarasan, and J. Caverlee. Crowdturfers, Campaigns, and Social Media: Tracking and Revealing Crowdsourced Manipulation of Social Media. In *ICWSM*, 2013.
[45] I. Lunden. Twitter revoked API access for 142K apps covering 130M 'low-quality' tweets in 1 week under new terms. https://techcrunch.com/2018/04/25/twitter-axed-142k-apps-violating-tos-in-q1-accounting-for-130m-low-quality-tweets/, Apr. 2018.
[46] D. MacMillan and R. McMillan. Google Exposed User Data, Feared Repercussions of Disclosing to Public. https://www.wsj.com/articles/google-exposed-user-data-feared-repercussions-of-disclosing-to-public-1539017194, The Wall Street Journal, 2018.
[47] A. Mak. Beware: A Twitter Scam That Appeals to Users' Vanity Is Hijacking Accounts. http://www.slate.com/blogs/future_tense/2017/10/10/a_fake_app_is_making_twitter_accounts_post_spam.html, Oct. 2017.
[48] F. Morstatter, J. Pfeffer, H. Liu, and K. Carley. Is the Sample Good Enough? Comparing Data from Twitter's Streaming API with Twitter's Firehose. In *ICWSM*, 2013.
[49] S. Nilizadeh, F. Labreche, A. Sedighian, A. Zand, J. Fernandez, C. Kruegel, G. Stringhini, and G. Vigna. POISED: Spotting Twitter Spam Off the Beaten Paths. In *ACM CCS*, 2017.
[50] M. S. Rahman, T.-K. Huang, H. V. Madhyastha, and M. Faloutsos. FRAppE: Detecting Malicious Facebook Applications. In *ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2012.
[51] J. Russell. Prominent Twitter accounts compromised after third-party app Twitter Counter hacked. https://techcrunch.com/2017/03/15/twitter-counter-hacked/, TechCrunch, Mar. 2017.
[52] J. Song, S. Lee, and J. Kim. CrowdTarget: Target-based Detection of Crowdturfing in Online Social Networks. In *ACM CCS*, 2015.
[53] G. Stringhini, G. Jacob, M. Egele, C. Kruegel, and G. Vigna. EVILCOHORT: Detecting Communities of Malicious Accounts on Online Services. In *USENIX Security Symposium*, 2015.
[54] G. Stringhini, C. Kruegel, and G. Vigna. Detecting Spammers on Social Networks. In *Annual Computer Security Applications Conference (ACSAC)*, 2010.
[55] G. Stringhini, G. Wang, M. Egele, C. Kruegel, G. Vigna, H. Zheng, and B. Y. Zhao. Follow the Green: Growth and Dynamics in Twitter Follower Markets. In *ACM Internet Measurement Conference (IMC)*, 2013.
[56] K. Thomas, C. Grier, V. Paxson, and D. Song. Suspended Accounts in Retrospect: An Analysis of Twitter Spam . In *ACM SIGCOMM*, 2011.
[57] K. Thomas, F. Li, C. Grier, and V. Paxson. Consequences of Connectivity: Characterizing Account Hijacking on Twitter. In *ACM CCS*, 2014.
[58] K. Thomas, D. McCoy, C. Grier, A. Kolcz, and V. Paxson. Trafficking Fraudulent Accounts: The Role of the Underground Market in Twitter Spam and Abuse. In *USENIX Security Symposium*, 2013.
[59] B. Viswanath, M. A. Bashir, M. Crovella, S. Guha, K. P. Gummadi, B. Krishnamurthy, and A. Mislove. Towards Detecting Anomalous User Behavior in Online Social Networks. In *USENIX Security Symposium*, 2014.
[60] G. Wang, C. Wilson, X. Zhao, Y. Zhu, M. Mohanlal, H. Zheng, and B. Y. Zhao. Serf and Turf: Crowdturfing for Fun and Profit. In *WWW*, 2012.
[61] Z. Yang, C. Wilson, T. G. Xiao Wang, B. Zhao, and Y. Dai. Uncovering Social Network Sybils in the Wild. *ACM Transactions on Knowledge Discovery*, 2014.
[62] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. SybilGuard: defending against sybil attacks via social networks. In *ACM SIGCOMM*, 2006.

DPLink: User Identity Linkage via Deep Neural Network From Heterogeneous Mobility Data

Jie Feng[1], Mingyang Zhang[1], Huandong Wang[1], Zeyu Yang[1], Chao Zhang[2], Yong Li[1], Depeng Jin[1]
[1]Beijing National Research Center for Information Science and Technology, Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
[2]Dept. of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA
liyong07@tsinghua.edu.cn

ABSTRACT

Online services are playing critical roles in almost all aspects of users' life. Users usually have multiple online identities (IDs) in different online services. In order to fuse the separated user data in multiple services for better business intelligence, it is critical for service providers to link online IDs belonging to the same user. On the other hand, the popularity of mobile networks and GPS-equipped smart devices have provided a generic way to link IDs, *i.e.*, utilizing the *mobility traces* of IDs. However, linking IDs based on their mobility traces has been a challenging problem due to the highly heterogeneous, incomplete and noisy mobility data across services.

In this paper, we propose *DPLink*, an end-to-end deep learning based framework, to complete the user identity linkage task for heterogeneous mobility data collected from different services with different properties. *DPLink* is made up by a *feature extractor* including a location encoder and a trajectory encoder to extract representative features from trajectory and a *comparator* to compare and decide whether to link two trajectories as the same user. Particularly, we propose a pre-training strategy with a simple task to train the *DPLink* model to overcome the training difficulties introduced by the highly heterogeneous nature of different source mobility data. Besides, we introduce a multi-modal embedding network and a co-attention mechanism in *DPLink* to deal with the low-quality problem of mobility data. By conducting extensive experiments on two real-life ground-truth mobility datasets with eight baselines, we demonstrate that *DPLink* outperforms the state-of-the-art solutions by more than 15% in terms of hit-precision. Moreover, it is expandable to add external geographical context data and works stably with heterogeneous noisy mobility traces. Our code is publicly available[1].

CCS CONCEPTS

• Information systems → Location based services; **Data mining**; **• Human-centered computing → Ubiquitous and mobile computing design and evaluation methods**;

KEYWORDS

deep learning; mobility trajectory; user identity linkage

ACM Reference Format:
Jie Feng[1], Mingyang Zhang[1], Huandong Wang[1], Zeyu Yang[1], Chao Zhang[2], Yong Li[1], Depeng Jin[1] . 2019. *DPLink*: User Identity Linkage via Deep Neural Network From Heterogeneous Mobility Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313424

1 INTRODUCTION

Smartphones and other mobile devices have made it easy for users to access various online services nearly everywhere and at any time. It is very common for a user to have multiple online identifiers (IDs) in different services such as online social networks (OSN), e-commerce services, online games, etc. Service providers have strong motivations to massively mine user data for monetization and optimizing user experience [16]. To capture a more comprehensive understanding of user behavior, it is increasingly intriguing to link user IDs across multiple services to fuse the separated data [37, 40].

To these ends, linking online IDs plays a critical role in the data fusion for better business intelligence. Early research has explored different ways to link user IDs by using service-specific data such as user profile attributes [9] and social graphs [15]. However, these approaches depend on whether these services have the same data type. For example, e-commerce services often do not have social graphs to match with an online social network. Moreover, users may fill in the fake information (*e.g.*, name, gender) in their profiles, which makes the linkage even harder.

In this paper, we explore a more generic approach to link user IDs by leveraging the spatial-temporal locality of user activities. The key intuition is that no matter what online services a user accesses, we can bind them to the user's *physical presence*, which is characterized by time and location. This becomes possible because most online services today have a mobile version with the location as parts of the service (*e.g.*, Uber, Yelp, Twitter). In addition, with some tolerance on granularity, even network accessing related information can be translated into location [14]. Our goal is to link multiple online IDs that belong to the same users across different services. Despite the inspiring prospects and results of user identity linkage, there are several key challenges that remain to be solved:

Heterogeneity nature of mobility data: Due to the different usage behavior of users and various collection mechanisms, the properties of mobility data (*e.g.*, sample rate, time periods) are drastically different across services. For example, the mobility traces

[1]https://github.com/vonfeng/DPLink

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313424

collected by an Internet Service Provider (ISP) are over 3 times denser than those collected by an online social network. Early works simply assume mobility traces of different services have similar sample rate [20] or time period [24], which are sensitive to the heterogeneous mobility data and do not perform well in practice. Due to this challenge, the general trajectory similarity [17, 21] algorithm designed for the trajectory from the same data source also fails to model the correlated relationship between the different data source and can not be effectively applied in the linkage problem.

Poor quality of mobility data: The data quality of collected mobility data is not always so good. *On the one hand*, the collected data only records the time and location information of mobility which is not enough to mine the hidden semantics of it. *On the other hand*, due to the limitation of devices and other artificial reasons [31, 36], the collected data usually contains noisy records to generate significant spatial and temporal mismatches between trajectories from different service. Because of the limitation of algorithms and the lack of proper data, existing approaches [24, 25] ignore the hidden semantics of mobility trajectory. Although, some works [32] propose prior knowledge-based solutions to address the mismatch problem. These solutions require proper manual parameter settings and are difficult to be applied in reality.

In this paper, we propose *DPLink*, an end-to-end deep learning based framework, to achieve linking IDs belonging to the same user for mobility data collected from different services with heterogeneous nature. *DPLink* is consist of two main components: *feature extractor module* and *comparator module*. The *feature extractor module* is designed to extract vector features from input raw trajectory and model relations between the trajectories from the different data source. The following *comparator module*, implemented as a multi-layer feed-forward network, is aimed to yield the final similarity score of the extracted representative trajectory feature vectors.

As the core component of *DPLink*, *feature extractor* contains two level encoders: *location encoder* and *trajectory encoder*. The *location encoder* is designed to integrate multi-dimensional input and extract the low-level feature of isolated locations. The multi-modal embedding based design in location encoder makes *DPLink* expandable to other available geographical features like PoI context. The following *trajectory encoder* is designed to capture the transitional relations of a single trajectory itself and model the correlation between two different trajectories. In the *trajectory encoder*, the transitional relations of single trajectory are captured by a *recurrent encoder*. Then a *selector* is introduced at the end of *trajectory encoder* to force the model to focus on the discriminative parts of trajectory and model the correlations between two different trajectories. With the help of this attention based selector, *DPLink* not only is able to observe the similar parts of two trajectories but also works robustly with noisy and missing trajectory.

Besides, we propose a pre-training mechanism to address the training challenges introduced by the heterogeneity nature of mobility data. Following our proposed training mechanism, our model is first trained to complete a warm-up task as linking trajectory from one mobility data but with different time periods. Due to the regularity of human mobility [10, 27] and the consistency and high quality of used mobility data, this single data based linkage task is much easier for *DPLink* to complete than the linkage task on cross-domain datasets with different quality. Intuitively, this single data based linkage task acts as a simple *auxiliary task* to help the model to first learn about the basic knowledge of the trajectory space. Then, the pretrained network with prior knowledge of the physical world and trajectory data is trained to complete the final *target task*: user identity linkage task on different data collected from different services.

Our contributions can be summarized as follows:

- We are the first to use deep learning techniques in the user identity linkage problem based on the heterogeneous mobility traces collected from different services with different quality. Our model does not assume any property of mobility data and works with heterogeneous mobility data across services.
- We utilize recurrent network with pooling unit as a trajectory encoder to extract transitional features from each single trajectory and introduce attention based selector to capture potential correlations between trajectories from two different data source. With fusing these two features together, our model obtains a comprehensive high-level understanding of trajectory.
- We propose a simple yet effective pre-training mechanism to adapt to the heterogeneity nature of different source mobility data. And the proposed training mechanism greatly improves the performance and robustness of our deep learning model on user identity linkage task on the different data source.
- We perform experiments on two real-life mobility datasets and compare the performance of our model with eight baselines. Extensive results demonstrate that *DPLink* outperforms the state-of-the-art solutions by more than 15%. Moreover, our model succeeds in utilizing additional geographical context data to further improve the performance and works robustly with noisy data.

The rest of this paper is organized as follows. We first formulate the problem in Section 2. Following the formulation, in Section 3, we describe the details of the whole framework of *DPLink*. In Section 4, we apply our model to two real-world mobility datasets and conduct extensive experiments. After systematically reviewing the related works in Section 5, we finally conclude our paper in Section 6.

2 PROBLEM FORMULATION

Let $\mathcal{A}$ represents the set of online account IDs, and S represents the set of different online platforms. Then, $\forall s \in S$, $\mathcal{A}^s$ denotes the set of online account IDs on platform s. Let a tuple $p = (l, t, e)$ represents a location record in location l with semantic label e at time t. Note that locations and times may be recorded at a different granularity and levels of precision in different source mobility data (*e.g.*, GPS coordinates or nearest base station in location record). Thus, without loss of generality, locations and times are divided into bins corresponding to geographical regions (*e.g.*, street block) and intervals of time (*e.g.*, one hour). In this paper, semantic label e is the distribution of point of interests (PoIs). But, we demonstrate that it is not limited to the PoI context, other information like textual context from user review can also be easily applied to our model without too much modification of network. We further define $\mathcal{T}$ as the set of all time bins, $\mathcal{L}$ as the set of all locations, and $\mathcal{E}$ as the set of PoIs.

Given any online ID $u \in \mathcal{A}$, we define its location records as, $\mathcal{R}(u) = \{p_1, p_2, ..., p_n\}$, where n denotes the number of location

Table 1: A list of commonly used notations.

Notat.	Description
$\mathcal{A}$	The set of all online IDs.
S	The set of types (platform) for online IDs.
$\mathcal{A}^s$	The set of online account IDs on platform s.
$\mathcal{T}$	The set of all time slots.
t	time stamp.
$\mathcal{L}$	The set of all regions.
l	location indicator.
$\mathcal{E}$	The set of PoIs.
e	semantic information.
$p = (l, t, e)$	A location record.
x	The embedded location record.
h	The hidden state of recurrent network.
u, v	online user ID.
$\mathcal{R}(u)$	location records sequence for online ID u.
$r(u)$	trajectory slice for online ID u.
$I(u, v)$	Binary variable indicating whether ID u and v belong to the same users.
N	The number of linking candidates for online ID u.

records. Taking the continuity of mobility data into consideration, we further partition the location records into meaningful trajectories $\mathcal{R}(u) = \{r_1(u), r_2(u), ...\}$ with maximum time window T_w (*e.g.*, 1 day). Given a pair of online IDs $u \in \mathcal{A}^1$ and $v \in \mathcal{A}^2$, let a binary variable $I(u, v)$ indicates whether these two IDs belong to the same user when the trajectories of them $r(u), r(v)$ are known,

$$I(u, v) = \begin{cases} 1, & u,v \text{ belong to the same user.} \\ 0, & u,v \text{ belong to different user.} \end{cases}$$

Further, given a target ID u, a list of candidates IDs $v_1, v_2, ..., v_N \subseteq \mathcal{A}^2$ and their trajectories $r(u)$ and $r(v_i)$ for $i = 1, 2, ..., N$, we aim to build a function, which is approximate to the identity function I enough, to find the best matching trajectory $r(v_i)$ and ID v_i.

Following the aforementioned definitions, many researchers [24, 25, 32] proposed insightful algorithms to work out this problem. However, because of the data quality problem and heterogeneity of mobility trajectory, these methods are still far from application in reality. Inspired by the powerful representation ability of deep learning models, we propose a deep learning based framework to address these challenges and aim to achieve better performance

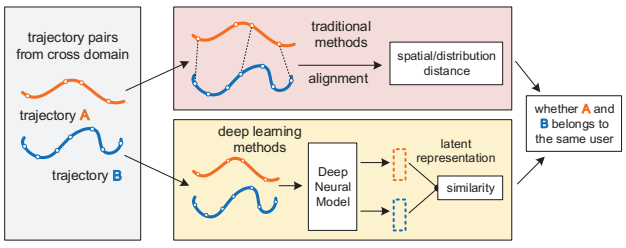


Figure 1: Different working mechanism of traditional methods and our proposed model.

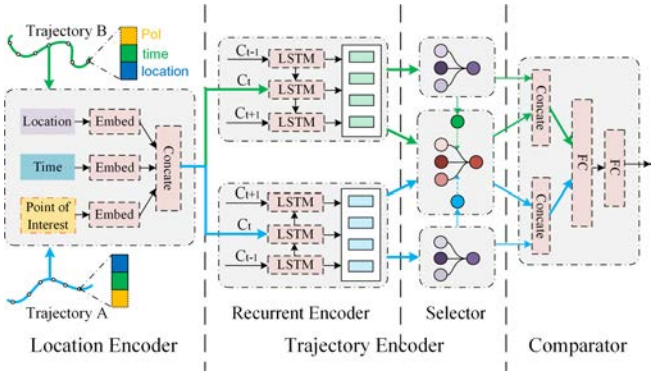


Figure 2: Main architecture of *DPLink* consisting of three major components: location encoder, trajectory encoder (including recurrent encoder and attention based selector), and comparator network.

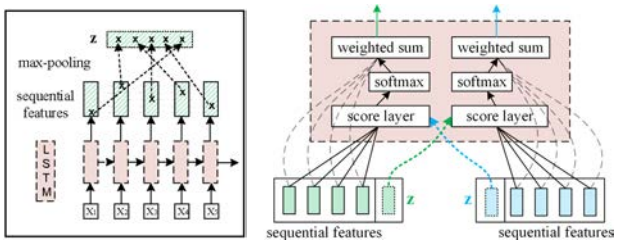
on real-life mobility dataset. The difference in workflows between existing works and our work is shown in Figure 1. In the traditional workflow of existing works, they usually first align the trajectory from the spatial or probabilistic view and then calculate the spatial or distribution distance to obtain a score. However, instead of aligning trajectory directly, we aim to utilize deep learning tool to first obtain the latent representation of mobility trajectory and then calculate the similarity between these two vectors as the similarity of trajectories. The details of our proposed model can be found in the next section.

3 MODEL AND METHOD

The structure of *DPLink* is presented in Figure 2. *DPLink* contains three major components: *location encoder*, *trajectory encoder*, and *comparator network*. The input trajectory pairs are first processed by *location encoder* and then fed into *trajectory encoder* to extract multi-level representative features. Particularly, *DPLink* uses a attention based *selector* at the end of *trajectory encoder* to focus on the similar parts of the trajectory pair and avoid the potentially harmful influence of missing and noisy data. Finally, the generated vector pair, which represents the original trajectory pair, are fed into the *comparator network* to calculate the similarity score of two input trajectories.

3.1 Location Encoder

Location encoder is a multi-modal embedding module, which is designed to reorganize and embed the spatial and temporal features of a trajectory point $p_i = (t_i, l_i)$ into a single vector x_i. Further, due to the flexibility of embedding network, we can also easily model external information e_i like geographical context information in the embedding network. In this way, our model is able to capture the hidden semantic behind the spatial-temporal point, which is difficult to utilize and model for existing methods. We design three sparse linear embedding layers to encode each type of input (*e.g.*, one-hot) into a dense vector representation. Then, we concatenate them together to obtain an ensemble vector x_i. To strengthen the modeling ability of the embedding module, we add *tanh* function



(a) Recurrent encoder. (b) Attention-based selector.

Figure 3: Details of trajectory encoder in *DPLink*.

as the final non-linear activation function. The formulation of the embedding module is as follows,

$$x_i = tanh(W_p p_i + b_p) = tanh([W_t t_i + b_t; W_l l_i + b_l; W_e e_i + b_e]),$$

where W and b denote the learnable parameters of embedding layers, $tanh$ denotes the non-linear activation function, $[;;]$ denotes the concatenate function.

As mentioned above, the location fed into our model is in the form of a one-hot vector. Compared with the original geographic coordinates, one-hot vector lose the information of spatial dependencies, which means that regions close to each other in the physical world can be far away in the one-hot space. However, the spatial dependency is important in measuring trajectory similarity. In order to enable *DPLink* to learn about the spatial adjacency of location, we introduce mobility prediction task to help our model learn the meaning of closeness of location. This task is to predict the next location of an object with knowing its trajectory history. Due to the regularity of mobility, the location points of the trajectory which neared in the time dimension is also neared in the spatial dimension. In other words, the location encoder is enforced by the mobility prediction task to embed the adjacent location in the physical world into the adjacent space in the latent high-dimensional space in a neural network.

It is noted that this multi-modal embedding module is shared by the network to simultaneously process two input trajectories. This sharing mechanism guarantees that two types trajectories from the same geographical space can be projected into another same latent space. Besides, the shared embedding module also greatly reduces the parameters of the whole network.

3.2 Trajectory Encoder

Following the *location encoder* is the *trajectory encoder*, which contains a *recurrent encoder* to extract the transitional level feature of single trajectory and a *selector* to extract correlated level features of trajectory pair.

3.2.1 Recurrent Encoder. Based on the embedded vector representation $\{x_1, x_2, ..., x_n\}$ of original trajectory $\{p_1, p_2, ..., p_n\}$, recurrent encoder is introduced to capture the sequential transitions and model the mobility pattern of a single trajectory. The output encoded trajectory feature is recorded as vector representations $\{h_1, h_2, ..., h_n\}$. As Figure 3(a) shows, the recurrent encoder is made up of a recurrent neural network with max/mean pooling operation. Recurrent neural network is a standard class of neural networks, which is designed with cycle and internal memory to model the sequential information. We use the widely used long short term memory (LSTM) [11] and its popular variation gated recurrent unit (GRU) [5] as the basic unit of our recurrent encoder.

To enhance the ability of sequential pattern modeling, we apply stacked techniques in the recurrent encoder. In practice, we stack several recurrent layers by feeding the output of the former recurrent layer into the next recurrent layer. To overcome the potential overfitting introduced by too many recurrent units, we apply dropout to the output of every recurrent layer.

Since the number of features extracted from the recurrent network is still identical to the length of trajectory, we introduce pooling operation (*e.g.*, max pooling and mean pooling) after recurrent network to obtain fix-length vector representation of trajectory. Actually, this operation is beyond only feature reshaping but also achieves a certain range of performance improvement in our experiment. This is because that pooling operation acts as a simple self-feature filter to select the important features. As Figure 3(a) shows, with the help of recurrent network with max pooling, we model the sequential relations $\{h_1, h_2, ..., h_n\}$ and finally obtain a single feature vector z for each trajectory. And we call this single feature vector z as the *primary representation vector* for trajectory.

3.2.2 Co-Attention based Selector. To handle the data quality problem and enable our model to focus on the critical parts of the trajectory pair for matching and linking, we propose to design a co-attention based selector network. We first briefly introduce the background knowledge of attention mechanism.

Given a query vector q and a series of candidate vectors $\{h_1, h_2, ..., h_n\}$, attention mechanism can be implemented with two steps: 1) to calculate the "correlation" between the query vector q and all these candidate vectors $\{h_1, h_2, ..., h_n\}$; 2) with these normalized "correlation" as weights $\{\alpha_1, \alpha_2, ..., \alpha_n\}$, to calculate the weighted sum y of candidate vectors as the comprehensive representation of them. This weighted sum y is regarded as the summary of the most related parts of candidates $\{h_1, h_2, ..., h_n\}$ for the query q. There are three widely used attention methods: *dot*, *general*, *mlp*. The main difference between these attention implementations is the calculation of "correlation". The formulation of typical attention methods are as follows,

$$y = \sum \alpha_i h_i, \quad \alpha_i = \sigma(f(q, h_i)),$$
$$f_{dot}(q, h_i) = h_i^T q,$$
$$f_{gen}(q, h_i) = h_i^T W q,$$
$$f_{mlp}(q, h_i) = v^T tanh(Wq + Uh_i),$$

where W, U, v are the learnable parameters, f represents the score function, σ is the soft-max function, h_i represents i_{th} candidate vector, q is the query vector and y is the final output.

Figure 3(b) presents the core idea of the co-attention based selector. In the common application of attention mechanism like neural machine translation, there is only one candidate sequence, where the query vector q of attention is naturally constructed from candidates $\{h_1, h_2, ..., h_n\}$, *e.g.*, $q = h_n$. Different from the former, we introduce co-attention mechanism to adapt general attention network for pair input to capture and model the correlation relationship. For the candidates $\{h_1^A, h_2^A, ..., h_n^A\}$ from trajectory A, we

use the *primary representation vector* z_B of trajectory B as the query vector $q_A = z_B$. Meanwhile, we use the *primary representation vector* z_A of trajectory A as the query vector $q_B = z_A$ for the candidates $\{h_1^B, h_2^B, ..., h_m^B\}$ from trajectory B. In this way, we directly connect two trajectories before the final comparator network and give them opportunities to find the related parts of each other. Further, this co-attention based selector will reduce the harmful effects from potential noisy records. After the processing of co-attention network, we obtain the final feature vector of trajectory: y_A for original trajectory A $\{p_1^A, p_2^A, ..., p_n^A\}$ and y_B for original trajectory B $\{p_1^B, p_2^B, ..., p_m^B\}$.

3.3 Comparator Network

The final component of our model is a comparator network, which is implemented as a multilayer feed-forward network. The final layer of the comparator network is a neural unit with *sigmoid* function acting as a logistic regression function to generate the final similarity score. The feature vectors (y_A, z_A) and (y_B, z_B) are independently fused together to obtain the final vector presentation of each trajectory. Then these two vector representations are fed into multi-layer feed-forward network to yield the similarity *score*. From the view of the problem, the comparator network gives two input trajectories a second chance to exchange information and validate each other. From the view of the network, the comparator network can be regarded as a simple but effective classifier for binary classification to judge whether two input vectors belong to the same class. Due to the *sigmoid* function, the output score is a normalized probability which can be optimized by binary cross entropy loss function.

3.4 Training strategy

Our model works in an end-to-end manner without requiring handcrafting features. Since we translate the user identity task into a binary classification problem, we choose binary cross entropy loss as the objective loss function.

$$Loss = -\sum_{i=1}^{n} y_i \log x_i + (1 - y_i) \log(1 - x_i),$$

In the training, the Adaptive Moment Estimation (Adam) algorithm is utilized to optimize the model. Several widely used tricks, such as dropout, L2 regularization, and learning rate schedule, are used to avoid the overfitting problem. The detailed settings for these parameters can refer to the next section.

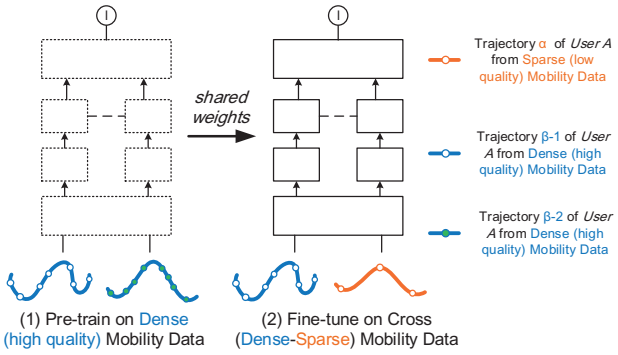


Figure 4: Two step training strategy for user identity linkage task.

However, due to the heterogeneous nature of mobility trajectories from the different service and platform, our model always fails to converge and stops with really poor performance when directly trained from scratch. The intuition that mobility trajectories from different platforms share identical underlying spatial-temporal pattern is the precondition for completing the user identity linkage task. Nevertheless, due to the heterogeneous nature of mobility trajectories from different data source, our model struggles to explore the huge spaces and learn the true knowledge of it. On the one hand, different platforms provide users with various services and users generate different mobility behaviors with various intentions. On the other hand, the sample rate and recording mechanism of collecting data in different platforms are also different. All of these lead to the heterogeneous nature of mobility trajectories, which make it difficult for our model to learn valuable knowledge. Consequently, our original model fails to link trajectories across different mobility data.

Here, we propose to pretrain the whole model with a simple task as warm-up to first obtain the basic knowledge of the physical world and trajectory pattern. Once our model achieves good performance on this simple task, we start to train it with the final user identity linkage task on different mobility data source. This warm-up task is to identify whether two mobility trajectories from the same platform in different periods (*e.g.*, different days) belong to the same user. This warm-up task is proposed based on the observation from the regularity of human mobility [10, 27], people usually go to the same places (*e.g.*, to and from where he needs to go like home and office) and generate similar even the same trajectories in different work days. Furthermore, the trajectory used in the warm-up task are both from the identical high-quality mobility dataset, which is denser and without too many missing records. Due to the regularity of mobility and the high-quality mobility dataset, the warm-up task becomes an easier and directional task to help the model to converge and learn the useful prior knowledge for the future difficult task.

The two-step training strategy is presented in Figure 4. Our training strategy can be regarded as a kind of network-based transfer learning method. The prior knowledge learned from the first linkage task on a single dataset is transferred to the final linkage task (*target task*) on different datasets by sharing partial network weights.

4 PERFORMANCE EVALUATION

In this section, we conduct extensive experiments on two real-world datasets to answer the following research questions:

- **RQ1:** How does *DPLink* perform in the user identity linkage problem on real-life datasets compared with existing algorithms?
- **RQ2:** How does the proposed training mechanism help to improve the performance of *DPLink* on the user identity linkage task?

Table 2: Statistics of the datasets.

Dataset	Users	Records	Locations	Rec./U.	Loc./U.	Duration
Twitter	1228	53337	8975	43	25	21 months
Foursquare	2970	44915	8975	15	12	48 months
ISP	2844	325215	12576	114	19	1 week
Weibo	1761	49651	12576	28	5	1 week

- **RQ3:** Can *DPLink* be applied in data with poor quality (*e.g.*, noisy records)? Is it possible to utilize more external geographical context data (*e.g.*, PoI information) to improve the performance?

In what follows, we first describe the experimental settings, and then answer the above three research questions one by one.

4.1 Experimental Settings

4.1.1 Datasets. We carry out experiments with two real-world cross-domain mobility datasets. The first dataset is the mobile network record dataset provided by one of the largest major Internet Service Provider (ISP) in China and the social network location service records obtained from Weibo[2]. The second dataset, provided by Zhang et al. [41], is the location check-in records from Foursquare[3] and Twitter[4] with the same users. Different generation and collection mechanism of these datasets ensure the representativeness of our experiments. Table 2 summarizes the basic statistical information of datasets. Below, we describe these datasets in detail.

ISP-Weibo. The ISP dataset contains 325215 mobility records that cover the metropolitan area of Shanghai from April 19 to April 26 in 2016. The location records are generated in the base station level when users access cellular network via mobile devices for communication and the Internet. Each trajectory is characterized by anonymized user ID. It contains a series of spatial-temporal points produced by the user, where each point includes a base station ID and a timestamp. We replace the base station ID with a certain longitude-latitude coordinate according to the base station information provided by ISP. The Weibo dataset is generated from ISP dataset. Our collaborators from ISP collected the Weibo sessions from the ISP datasets in the same time window with the permission of Weibo. In Weibo dataset, each mobile trace is characterized by a Weibo ID and contains a series of GPS coordinates that show up in HTTP sessions between the mobile application and the Weibo server. These GPS coordinates are produced when users access the location service in their mobile applications like check-in. Before we access the dataset, the collaborators have mapped the Weibo ID into the same anonymized user ID in the ISP dataset to protect the personal privacy.

Foursquare-Twitter. This dataset contains the trajectories data from Foursqaure, a popular location-based social network (LBSN), and Twitter, a micro-blogs social network around the world. As its primary function, Foursquare provides users various location-related services like location check-ins and posting online reviews. Twitter also provides users with basic location-based services like location check-ins. Provided by Zhang et al. [41], the data is crawled from the web pages of thousands of users who use both platforms with at least one location records during November 2012 in the east coast of U.S.. Based on the crawled raw data, the trajectory is also characterized as a user's account ID and a series of GPS coordinates with timestamp.

[2]https://weibo.com
[3]https://www.foursquare.com/
[4]https://www.twitter.com

Table 3: PoIs category distribution on ISP-Weibo dataset.

Related Function	PoI Categories
Residence	residence, life services.
Entertainment	food, hotel, gym, shopping, leisure.
Business	finance, office building, company, trading area.
Industry	factory, industrial estate, economic development zone.
Education	school, campus.
Scenery spot	scenery spot, tourism development.
Suburb	villages, towns.

Table 4: Typical parameter settings.

Item	Value	Features	input	output
batch size	32	location	10000	200
learning rate	1e-3	time	24	10
drop out	0.3	PoI	20	10
fine-tune lr	0.0005	hidden state	200	200

We crawled 0.75 million point of interests (PoI) of Shanghai from BaiduMap[5] as the additional geographical context of ISP-Weibo dataset. As Table 3 shows, the crawled PoI dataset contains 20 categories and can be classified into 7 region functions. For every base station in ISP dataset, we calculate the distribution of PoIs for it in the surrounding 1 kilometer area. This PoIs distribution serves as a soft function label for the region and describes the potential intention of the user's movement.

It's noted that all the user IDs in our datasets are anonymized to protect user's privacy. Meanwhile, we store the data in a secure local server and only core researchers can access the data with strict non-disclosure agreements.

4.1.2 Baselines. We compare the performance of our model with eight state-of-the-art baselines. Six of them are classic user identity linkage algorithms, another two are deep learning based trajectory representation models.

NFLX: With knowing some external information, Narayanan et al. [23] propose a statistical model based mobility trace similarity score to identify the users in Netflix dataset. NFLX cannot be directly applied into cross-domain linkage problem and we follow the method [24] to adapt it to our problem.

MSQ: Ma et al. [20] incorporate general knowledge in forms of global movement constraints and preferences to identify users from dataset. Specifically, they consider negative square difference between two mobility traces as their similarity score.

[5]https://map.baidu.com/

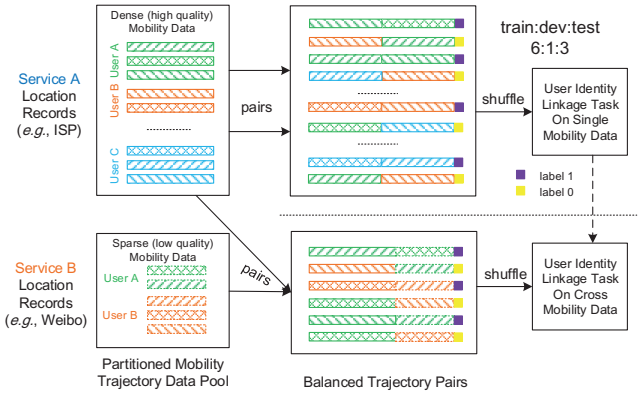


Figure 5: Build training/validation/testing dataset.

HIST: Naini et al. [22] focus on linking users by matching the location histograms of their mobility traces. Firstly, they compute a user's visiting frequency of each location and then define a similarity score of two histograms based on Kullback-Leibler divergence.

LRCF: Goga et al. [8] take the popularity of different locations into consideration. LRCF applies the term frequency-inverse document frequency(TF-IDF) [28] weighting scheme to location visiting histograms and measures mobility trace similarity using a cosine distance.

WYCI: Rossi et al. [25] propose a time based probabilistic user identifying algorithm. They use the frequency of user login in different locations to approximate the probability of visiting these locations. Then they determine if a mobility trace belongs to a user by computing the user's probability to produce the mobility trace.

POIS: POIS [24] algorithm uses the "encountering" events to measure the similarity of two different-domain mobility traces. It assigns every "encountering" event a weight based on statistical model and uses the weighted sum as the similarity measure.

As we are the first deep learning based model for user identity linkage task on different mobility datasets, we compare our model with two deep learning based models for trajectory representation.

TULER: Gao et al. [7] utilize the recurrent network to encode the trajectory into a single vector to identify users in single mobility data. TULER cannot be directly applied to our problem, thus we train two independent TULER models for each mobility data and check whether these two models can identify each input trajectory as the same user.

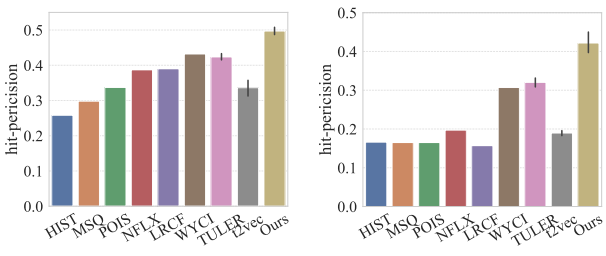
t2vec: Li et al. [17] propose to adapt seq2seq model with a spatial proximity aware loss function to infer and represent the underlying route information of a trajectory for efficient trajectory similarity computation. Following the philosophy of original paper, we adapt it to our problem by using sparse (low sampling) trajectory from one dataset as input and using dense (high sampling) trajectory with same user ID from another dataset as the underlying route to learn the representation of trajectory. Then, we use this representation to link different IDs.

4.1.3 Preprocessing. Figure 5 presents the details of how to build training/validation/testing dataset in our experiment. As Figure 5 shows, the first step is to divide the lengthy location records into continuous mobility trajectories. After this operation, we obtain two independent mobility trajectory pools, where each trajectory is labeled with its user ID, for two different mobility data. As mentioned in the training section, we first pretrain the model with user identity linkage task on single data as a warm-up and then fine-tune the model with user identity linkage task on cross-domain mobility data. Furthermore, we choose the high-quality mobility data with more frequent and longer records, which is called *dense* mobility trajectory data to complete the warm-up linkage task. Another mobility data source with fewer mobility records called *sparse* mobility data is only used in target user linkage task on cross-domain datasets. The mechanism of building data in two training stages are the same, while the main difference is to choose which data to use.

We first introduce the core step of building experiment data for user identity linkage task on single domain data as follows. We randomly choose two trajectories from the dense mobility data pool with identical owner labels to build positive trajectory pair. The basic intuition behind this operation is the regularity of human mobility [10, 27], which means that the mobility trajectory of one user keeps similar in the different periods with extremely high probability. To meet the sample balance requirement of binary classification problem in training stage, for every positive trajectory pair we randomly select two trajectories from dense mobility data pool with different user IDs to form a negative trajectory pair. In the training stage and validation stage, only this one negative sample is enough.

However, in reality, we usually have lots of online ID candidates for matching, and we need to compare them together to find the best matching. To simulate this practical case, in the testing stage, we choose N negative candidates for matching. Particularly, for each online ID, we choose those left online IDs whose trajectory have at least one intersection with it as candidates. Based on the statistics of our data, the average number of these candidates for each online ID is around 20. To meet the requirement of batch size for efficient computation, we select 32 as the default setting for N in the experiment. To build experimental data for user identity linkage task from different data sources (*target task*), we follow the similar steps mentioned before. Different from choosing two input trajectories from the same mobility dataset for warm-up linkage task, we choose one trajectory from dense mobility data and another trajectory from sparse mobility data for final user identity linkage task. In our experiment, the proportion of training, validation and testing data is 6:1:3. It's noted that we divide not only the trajectory but also the users by shuffling the trajectory data with user label. This operation makes sure the generalization of our experiment, where some users may only have few data in the training step and even they only appear in the testing step.

4.1.4 Metrics and Parameter Settings. Our model is implemented on Pytorch platform and the typical parameter settings of our model are presented in Table 4. In the experiments, we select GRU as the default recurrent unit and *dot* attention as the default attention mechanism for efficient computation. The results for LSTM and other basic components are similar. Following the aforementioned evaluation dataset, our model is trained and validated in a binary classification manner, and finally evaluated in the search and ranking manner.



(a) @ISP-Weibo dataset. (b) @Foursquare-Twitter dataset.

Figure 6: The performance of our model and baselines on two real-life datasets.

In the training step, we use the F1 score and AUC to measure the performance of our model, which are both widely used metrics in classification problems. The F1 score is defined as,

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}.$$

AUC, known as the area under the receiver operating characteristic (ROC) curve, is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. With P positive instances and Q negative instances, AUC can be calculated by the following formulation,

$$AUC = \frac{\sum_{i \in +} rank_i - P(1+P)/2}{P \times Q}.$$

In the validation step, we also use the F1 score and AUC as the metrics to cooperate with the early stopping mechanism to find the best parameters and avoid overfitting.

In the testing step, we use hit-precision as the metric to evaluate the performance of algorithms in the user identity linkage task with more than one (*e.g.*, $N = 32$, as mentioned in the preprocessing section) candidate online IDs. After *DPLink* deciding the score for each candidate, we rank these candidates by score and calculate the hit-precision. Hit-precision of top-k candidates is defined as follows,

$$h(x) = \begin{cases} \frac{k-x}{k}, & \text{if } k > x >= 0 \\ 0, & \text{if } x >= k \end{cases}$$

We choose $k = 5$ as the default setting for hit-precision.

4.2 Overall performance on user identity linkage task. (RQ1)

We evaluate our model with eight baselines on ISP-Weibo and Foursquare-Twitter datasets to present the performance on user identity linkage task. The results are presented in Figure 6.

We first analyze the results of ISP-Weibo dataset. Among six classic linkage baselines, WYCI algorithm with the tolerance of different sample rate performs best with 0.432 hit-precision. Besides, the performance of NFLX and LRCF with the tolerance of mismatch from the spatial or temporal view are also similar well. Particularly, the hit-precision of our model is 0.497, which is 15% higher than the best baseline WYCI.

Besides, we compare our model *DPLink* with other two state-of-the-art trajectory representation algorithms: TULER and t2vec. The results demonstrate that our model outperforms both of them more than 17%. TULER is designed for directly identifying the user from single mobility dataset. It fails to obtain better performance due to two potential reasons: 1) TULER cannot identify any new users because the user set is decided in the training step. However, in our experiments, the users in the testing step is not identical to the training step; 2) each TULER only focus on its single mobility dataset and ignores the potential correlation between two different mobility dataset. Based on seq2seq model, t2vec is proposed to infer and represent the underlying route information of a trajectory for efficient trajectory similarity computation. In practice, t2vec needs extremely high-quality dataset (*e.g.*, taxi trajectory dataset with less than 1 minute sampling period) to learn good enough representative function. It fails to handle with the poor quality (low sampling and missing records) and the heterogeneity nature of our cross-domain mobility dataset. *DPLink* is designed for the user identity linkage task and succeeds in dealing with these challenges with special network design and training strategy. On the one hand, the location encoder and trajectory encoder help *DPLink* extract representative features from a single trajectory. On the other hand, the attention based selector enables *DPLink* to capture the correlation between two trajectories with different properties from different mobility data. Finally, the MLP based comparator network acts as a powerful classifier to obtain the final result.

The similar results can also be found in the Foursquare-Twitter dataset. Because of the smaller volume and sparse nature of Foursquare-Twitter dataset, the performance of all the methods is lower. However, due to the powerful representation ability of neural network and transfer learning based training strategy, our model still achieves about 0.4 hit-precision. Compared with the-state-of-the-art methods, the performance gain of our model can be up to 25%. In a word, the evaluation results of two real-life datasets demonstrate the superiority of our model than other the-state-of-the-art methods.

4.3 The effects of the training strategy. (RQ2)

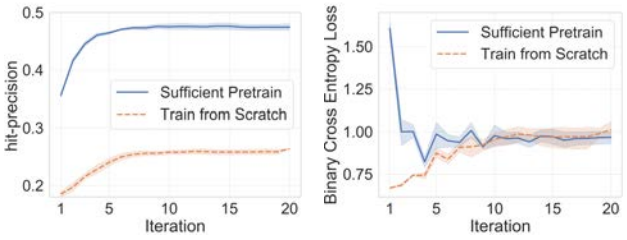
In this section, we conduct experiments to demonstrate the effect of pre-training strategy for *DPLink*. We take the results on ISP-Weibo dataset for example.

Table 5 shows the performance of our model with different training strategies on ISP-Weibo dataset. Obviously, the pretrain step is crucial for better performance. Furthermore, we choose to remove different components of *DPLink* from the sharing weights step to locate the most important component for the performance. As Table 5 shows, removing location embedding and selector weights from the sharing step leads to the worst performance, which denotes the crucial role of location encoder and attention-based selector in modeling. Besides, the effect of removing the recurrent encoder is smaller and the removing the comparator network weights even does not harm the performance at all. These little effects tell us that the transitional relations and classification knowledge are not hard to learn from beginning in this task.

We dive into the training process to understand the importance of pretrain step. As Figure 7(a) shows, after pretraining, the initial performance of our model before fine-tuning on target task can

Table 5: Performance of different training mechanisms on ISP-Weibo dataset. For example, "Full−Location Encoder" denotes that the embedding weights of location encoder is not shared in the pretrain step, which means the location encoder is trained from scratch in user identity linkage task.

Training Strategy	hit-precision (mean+std)	Δ
Train from Scratch	0.257 ± 0.025	-48.3%
Full Pretrain	0.497 ± 0.013	0
Full−Location Encoder	0.404 ± 0.018	-18.7%
Full−Recurrent Encoder	0.438 ± 0.005	-11.9%
Full−Selector	0.395 ± 0.014	-20.5%
Full−Comparator	0.499 ± 0.009	+0.4%



(a) The testing performance curves. (b) The testing loss curves.

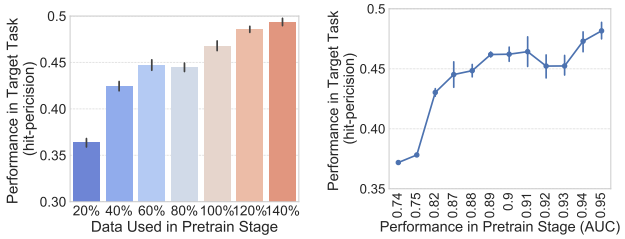
Figure 7: The comparison results between two types training mechanisms on ISP-Weibo dataset.

be higher than 0.35, which is better than the best performance of model trained from scratch. With fine-tuning on target task, the hit-precision of our model can be further improved to 0.49. Figure 7(b) presents the variation trend of testing loss of models with different training mechanism. Due to the heterogeneity of mobility data, the model trained from scratch becomes over-fitted at the very start. Meanwhile, the pretrained model really learns the knowledge about the task and performs better and better.

Figure 8 presents the effects of data volume and model performance of pretrain step on the performance of our model on target task. As Figure 8(a) shows, with more and more data utilized on single domain linkage task in the pretrain step, the final performance of our model on target task also becomes better and better. This observation tells us that enough unlabeled data on a single domain can help to improve the performance of model on target task and reduce the requirement of labeled cross-domain data. Figure 8(b) shows us that better pretrained model on single domain linkage task produces better fine-tune model on target task.

4.4 The expansion capability and stability of *DPLink* model. (RQ3)

In this section, we evaluate whether *DPLink* can utilize external information with the help of multi-modal embedding based location encoder to improve the performance on user identity linkage task and whether it works robustly with noisy records.



(a) The effect of data volume used in pretrain stage. (b) The effect of the performance in pretrain stage.

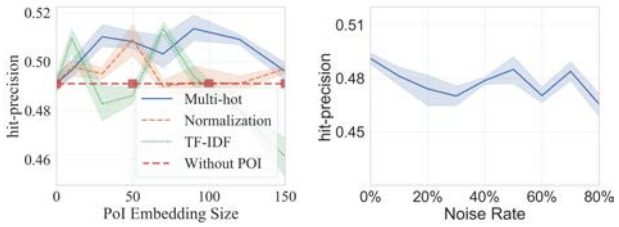
Figure 8: The effects of different pertrain parameters on ISP-Weibo dataset.

To evaluate the expansion capability of multi-modal embedding network, we build three kinds of PoI distribution features: 1) multi-hot distribution, which only describes whether specific type PoI exists in the region; 2) normalized probability distribution of different PoIs in the region; 3) tf-idf distribution, the enhanced version of normalized probability distribution. The evaluation results are presented in Figure 9(a). The results show that *DPLink* is able to apply external PoI information to further improve the performance from 0.49 to more than 0.51. Particularly, the multi-hot feature of PoI distribution performs much better and stable. However, *DPLink* is not limited to only use this kind of PoI distribution feature. Other external information like textual information can also be directly applied to our model after simple pre-processing like word2vec operation.

Figure 9(b) presents the performance variations of *DPLink* with noisy trajectory input. In Figure 9(b), noise rate in 20% means that 20% trajectories in the data are randomly selected to randomly replace one real data point in it with a noisy point. As Figure 9(b) shows, the performance of our model keeps stable with the increase of noise rate, which demonstrates the robustness of *DPLink* . Furthermore, we visualize the attention weights of three kinds of points in the trajectory to verify whether our co-attention based selector really plays an import role in filtering out the harmful effects of noisy records. Three types points are 1) intersection location between two trajectories; 2) noisy point inserted by the program; 3) other location points. In Figure 10, we analyze the average attention weights of three types points in 4 scenarios which include the dense/sparse (high/low quality) mobility data with positive/negative labels. As Figure 10 shows, the average attention weight for intersection point is the highest and the weight for the noisy point is the lowest, which distinctly demonstrates the selecting value of our co-attention based selector.

5 RELATED WORK

Applications of Identity Linkage: A number of applications can benefit from linking IDs across services. For example, Kumar et al. [16] investigated the user migration patterns across social media to provide guidance for online social network design. Zafarani et al. [40] and Yan et al. [37] leveraged linked IDs across social networks for better friend recommendations. Yang et al. [38] leveraged linked IDs across sites for better video recommendations.



(a) The effects of geographical context with different PoI embedding size.

(b) The effects of noisy records with different noise level.

Figure 9: The effects of data quality (geographical context and noisy records) on our model.

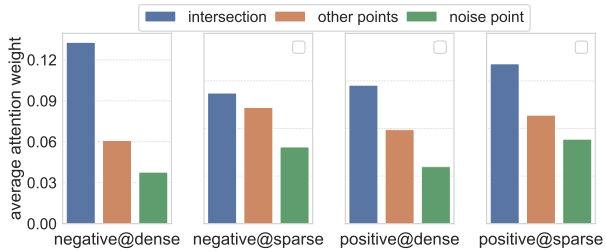


Figure 10: Average attention wights of three types points on testing dataset. For example, "negative@dense" means that the attention weights are from the dense(high quality) mobility trajectory with negative label.

All these works indicate the strong motivations for the service provider to link IDs belonging to the same user. Besides, some researchers [30, 35] tried to protect users from identification attack by matching mobility trajectory.

Identity Linkage using Trajectory Data: A few recent works examine the possibility of linking IDs based on location data [2–4, 13, 20, 22–26, 29, 33, 34]. Mudhakar et al. [29] and Ji et al. [12, 13] focused on linking IDs based on users' graph/network structures. They adapted their algorithms to location trajectories by constructing a "contact graph" to model users encountering with each other. However, these algorithms still require using social network graphs, which are not available in our scenario. In addition, some algorithms are designed to tolerate data noise such as temporal mismatching [23] and spatial mismatching [20]. Wang et al. [32] proposed algorithms with tolerating spatial or temporal mismatches (or both) and modeling user behavior for better linkage performance. Other algorithms implemented de-anonymization attacks based on *individual user's* mobility patterns [22, 25, 26]. Finally, researchers also developed identity linkage algorithms based on "encountering" events [2–4, 24]. By considering the location context (*e.g.*, user population density), it achieved a better performance [4, 24]. However, none of them is able to capture the trajectory dynamics and integrate semantic information simultaneously.

Representation Learning for Trajectories: Recently, deep learning has been used for spatial-temporal data mining [6, 18, 39, 42] such as next location prediction [6, 19] and trajectory embedding [7, 17, 39]. Yao et al. [39] used a recurrent network with manually features to cluster trajectory into several clusters. Gao et al. [7] proposed to identify users in one mobility dataset via a recurrent network encoder. However, the uniqueness of an individual in one dataset does not imply that this user can be easily recognized in another dataset. Besides, this method can not be applied to new users whose data has not been trained with the model. Based on one dense taxi trajectory dataset, Li et al. [17] used a seq2seq model to measure the similarity of sub-trajectories extracted from one dense trajectory, which requires high data quality and fails to model the relationship of trajectories from different datasets.

Due to the limitation of model design, all these existing methods are not suitable for user identity linkage problem for different mobility data. Compared with these methods, our model is not only designed to extract comprehensive trajectory features but also model the correlated relationship between trajectories. This interaction ability is based on the attention mechanism introduced by Bahdanau et al. [1] in neural machine translation task. Feng et al. [6] is the first work to introduce the attention model to predict human mobility. However, instead of achieving mobility prediction, our focus is to measure the trajectory similarity from the different data source to link online IDs.

6 CONCLUSION

In this paper, we investigated the task of user identity linkage by leveraging the power of deep learning. We proposed an end-to-end deep learning framework to link different accounts from heterogeneous mobility data. Proposed model employs location encoder and trajectory encoder to model the complicated single trajectory feature and apply co-attention based selector to focus on discriminative parts when matching two mobility trajectory. Extensive experiments on two real-life mobility datasets show that *DPLink* significantly outperforms eight baselines on the user identity linkage task. Compared with the existing solutions, the proposed model achieves general similarity measurement for heterogeneous mobility data. Besides, it is robust to the noise of trajectory and is easy to extend to external information like PoI distribution.

There are several future directions for our work. First, because of the limitation of the datasets, we only consider simple external geographical context data like PoIs category. In the future, we plan to expand the multi-modal embedding module to process the raw textual information in the check-in data. Second, although the pre-training mechanism works well, it is still not easy to directly train a good model for sparse mobility data. Thus, design more stable network structure and better training mechanism is also an important direction.

7 ACKNOWLEDGMENTS

This work was supported in part by The National Key Research and Development Program of China under grant 2017YFE0112300, the National Nature Science Foundation of China under 61861136003, 61621091 and 61673237, Beijing National Research Center for Information Science and Technology under 20031887521, and research fund of Tsinghua University - Tencent Joint Laboratory for Internet Innovation Technology.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Wei Cao, Zhengwei Wu, Dong Wang, Jian Li, and Haishan Wu. 2016. Automatic user identification method across heterogeneous mobility data sources. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 978–989.

[3] Alket Cecaj, Marco Mamei, and Franco Zambonelli. 2016. Re-identification and information fusion between anonymized CDR and social network data. *Journal of Ambient Intelligence and Humanized Computing* 7, 1 (2016), 83–96.

[4] Wei Chen, Hongzhi Yin, Weiqing Wang, Lei Zhao, and Xiaofang Zhou. 2018. Effective and Efficient User Account Linkage Across Location Based Social Networks. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 1085–1096.

[5] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[6] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. Deepmove: Predicting human mobility with attentional recurrent networks. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web (WWW)*. International World Wide Web Conferences Steering Committee, 1459–1468.

[7] Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Xucheng Luo, and Fengli Zhang. 2017. Identifying human mobility via trajectory embeddings. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*.

[8] Oana Goga, Howard Lei, Sree Hari Krishnan Parthasarathi, Gerald Friedland, Robin Sommer, and Renata Teixeira. 2013. Exploiting innocuous activity for correlating users across sites. In *Proceedings of the 22nd international conference on World Wide Web (WWW)*.

[9] Oana Goga, Patrick Loiseau, Robin Sommer, Renata Teixeira, and Krishna P Gummadi. 2015. On the reliability of profile matching across large online social networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1799–1808.

[10] Marta C Gonzalez, Cesar A Hidalgo, and Albert-Laszlo Barabasi. 2008. Understanding individual human mobility patterns. *Nature* 453, 7196 (2008), 779–782.

[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural computation* 9 8 (1997), 1735–80.

[12] Shouling Ji, Weiqing Li, Neil Zhenqiang Gong, Prateek Mittal, and Raheem A Beyah. 2015. On Your Social Network De-anonymizablity: Quantification and Large Scale Evaluation with Seed Knowledge.. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*.

[13] Shouling Ji, Weiqing Li, Mudhakar Srivatsa, and Raheem Beyah. 2014. Structural data de-anonymization: Quantification, practice, and implications. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1040–1053.

[14] Ethan Katz-Bassett, John P John, Arvind Krishnamurthy, David Wetherall, Thomas Anderson, and Yatin Chawathe. 2006. Towards IP geolocation using delay and topology measurements. In *Proceedings of the ACM SIGCOMM conference on Internet Measurement (IMC)*.

[15] Nitish Korula and Silvio Lattanzi. 2014. An efficient reconciliation algorithm for social networks. *Proceedings of the VLDB Endowment* 7, 5 (2014), 377–388.

[16] Shamanth Kumar, Reza Zafarani, and Huan Liu. 2011. Understanding User Migration Patterns in Social Media. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

[17] Xiucheng Li, Kaiqi Zhao, Gao Cong, Christian S Jensen, and Wei Wei. 2018. Deep Representation Learning for Trajectory Similarity Computation. (2018).

[18] Ziqian Lin, Jie Feng, Ziyang Lu, Yong Li, and Depeng Jin. 2019. DeepSTN+: Context-aware Spatial-Temporal Neural Network for Crowd Flow Prediction in Metropolis. In *AAAI*.

[19] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the Next Location: A Recurrent Model with Spatial and Temporal Contexts. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

[20] Chris YT Ma, David KY Yau, Nung Kwan Yip, and Nageswara SV Rao. 2013. Privacy vulnerability of published anonymous mobility traces. *IEEE/ACM Transactions on Networking (TON)* (2013).

[21] Nehal Magdy, Mahmoud A. Sakr, Tamer Mostafa, and Khaled El-Bahnasy. 2016. Review on trajectory similarity measures. In *IEEE Seventh International Conference on Intelligent Computing and Information Systems*.

[22] Farid M Naini, Jayakrishnan Unnikrishnan, Patrick Thiran, and Martin Vetterli. 2016. Where you are is who you are: User identification by matching statistics. *IEEE Transactions on Information Forensics and Security (TIFS)* (2016).

[23] Arvind Narayanan and Vitaly Shmatikov. 2008. Robust de-anonymization of large sparse datasets. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*.

[24] Christopher Riederer, Yunsung Kim, Augustin Chaintreau, Nitish Korula, and Silvio Lattanzi. 2016. Linking users across domains with location data: Theory and validation. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*. 707–719.

[25] Luca Rossi and Mirco Musolesi. 2014. It's the way you check-in: identifying users in location-based social networks. In *Proceedings of the second ACM Conference on Online Social Networks (COSN)*.

[26] Reza Shokri, George Theodorakopoulos, Jean-Yves Le Boudec, and Jean-Pierre Hubaux. 2011. Quantifying location privacy. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*.

[27] Chaoming Song, Zehui Qu, Nicholas Blumm, and Albert-László Barabási. 2010. Limits of predictability in human mobility. *Science* 327, 5968 (2010), 1018–1021.

[28] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* 28, 1 (1972), 11–21.

[29] Mudhakar Srivatsa and Mike Hicks. 2012. Deanonymizing mobility traces: Using social network as a side-channel. In *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 628–637.

[30] Zhen Tu, Kai Zhao, Fengli Xu, Yong Li, Li Su, and Depeng Jin. 2018. Protecting Trajectory from Semantic Attack Considering k-Anonymity, l-diversity and t-closeness. *IEEE Transactions on Network and Service Management* (2018).

[31] Gang Wang, Sarita Yardi Schoenebeck, Haitao Zheng, and Ben Y. Zhao. 2016. "Will Check-in for Badges": Understanding Bias and Misbehavior on Location-Based Social Networks. In *Proceedings of the International Conference on Web and Social Media (ICWSM)*.

[32] Huandong Wang, Chen Gao, Yong Li, Gang Wang, Depeng Jin, and Jingbo Sun. 2018. De-anonymization of Mobility Trajectories: Dissecting the Gaps between Theory and Practice. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*.

[33] Huandong Wang, Chen Gao, Yong Li, Zhi-Li Zhang, and Depeng Jin. 2017. From Fingerprint to Footprint: Revealing Physical World Privacy Leakage by Cyberspace Cookie Logs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM)*. 1209–1218.

[34] Huandong Wang, Yong Li, Gang Wang, and Depeng Jin. 2018. You Are How You Move: Linking Multiple User Identities From Massive Mobility Traces. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 189–197.

[35] Fengli Xu, Zhen Tu, Yong Li, Pengyu Zhang, Xiaoming Fu, and Depeng Jin. 2017. Trajectory Recovery From Ash: User Privacy Is NOT Preserved in Aggregated Mobility Data. In *Proceedings of the 26th International Conference on World Wide Web (WWW*. 1241–1250.

[36] Fengli Xu, Guozhen Zhang, Zhilong Chen, Jiaxin Huang, Yong Li, Diyi Yang, Ben Y Zhao, and Fanchao Meng. 2018. Understanding Motivations behind Inaccurate Check-ins. *Proceedings of the ACM on Human-Computer Interaction (CSCW)* (2018).

[37] Ming Yan, Jitao Sang, Tao Mei, and Changsheng Xu. 2013. Friend transfer: cold-start friend recommendation with cross-platform transfer learning of social knowledge. In *Proceedings of the International Conference on Multimedia and Expo (ICME)*.

[38] Chunfeng Yang, Huan Yan, Donghan Yu, Yong Li, and Dah Ming Chiu. 2017. Multi-site User Behavior Modeling and Its Application in Video Recommendation. In *Proceedings of the International ACM Conference on Research and Development in Information Retrieval (SIGIR)*.

[39] Di Yao, Chao Zhang, Zhihua Zhu, Jianhui Huang, and Jingping Bi. 2017. Trajectory clustering via deep representation learning. In *International Joint Conference on Neural Networks (IJCNN)*.

[40] Reza Zafarani and Huan Liu. 2014. Finding Friends on a New Site Using Minimum Information. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*.

[41] Jiawei Zhang, Xiangnan Kong, and Philip S. Yu. 2014. Transferring heterogeneous links across location-based social networks. In *WSDM*.

[42] Zefang Zong, Jie Feng, Kechun Liu, Hongzhi Shi, and Yong Li. 2019. DeepDPM: Dynamic Population Mapping via Deep Neural Network. In *AAAI*.

MARINE: Multi-relational Network Embeddings with Relational Proximity and Node Attributes

Ming-Han Feng
Graduate Institute of Networking and Multimedia
National Taiwan University
Taipei, Taiwan
r04944010@ntu.edu.tw

Chin-Chi Hsu
Institute of Information Science
Academia Sinica
Taipei, Taiwan
chinchi@iis.sinica.edu.tw

Cheng-Te Li
Institute of Data Science,
Department of Statistics
National Cheng Kung University
Tainan, Taiwan
chengte@mail.ncku.edu.tw

Mi-Yen Yeh
Institute of Information Science,
Research Center for IT Innovation
Academia Sinica
Taipei, Taiwan
miyen@iis.sinica.edu.tw

Shou-De Lin
Dept. of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan
sdlin@csie.ntu.edu.tw

ABSTRACT

Network embedding aims at learning an effective vector transformation for entities in a network. We observe that there are two diverse branches of network embedding: for homogeneous graphs and for multi-relational graphs. This paper then proposes MARINE, a unified embedding framework for both homogeneous and multi-relational networks to preserve both the proximity and relation information. We also extend the framework to incorporate existing features of nodes in a graph, which can further be exploited for the ensemble of embedding. Our solution possesses complexity linear to the number of edges, which is suitable for large-scale network applications. Experiments conducted on several real-world network datasets, along with applications in link prediction and multi-label classification, exhibit the superiority of our proposed MARINE.

CCS CONCEPTS

• **Information systems** → **Data mining**; *Social recommendation*; • **Computing methodologies** → **Knowledge representation and reasoning**.

KEYWORDS

Homogeneous Network Embedding; Multi-relational Network Embedding; Knowledge Graph embedding

ACM Reference Format:
Ming-Han Feng, Chin-Chi Hsu, Cheng-Te Li, Mi-Yen Yeh, and Shou-De Lin. 2019. MARINE: Multi-relational Network Embeddings with Relational Proximity and Node Attributes. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313715

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313715

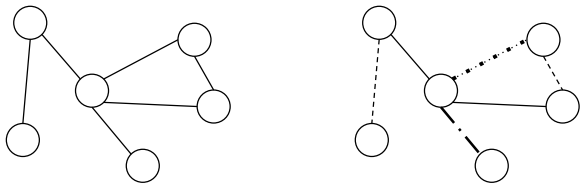


Figure 1: Homogeneous graphs (left) and multi-relational graphs (right). The latter consists of multiple types of link relationships (marked by various types of dashed lines) between node entities.

1 INTRODUCTION

The task of network embedding aims at learning a representation for each entity in a network. The idea is to *embed* some properties of a network into a vector space so that each entity is represented by a low-dimensional feature vector. Such form of representation is more ready for machines to perform further analysis, since most of the machine learning and data mining models accept unstructured feature vectors for training and prediction.

We observe that there are two branches of studies for network embedding. One is to propose an embedding method for *homogeneous graphs* (Figure 1 left) [1, 7, 8, 10, 12, 17, 29–31, 33, 36–38, 40–42, 45, 50–52], in which there is only one single type of nodes and links. The major goal of this branch of work is to preserve certain properties in the linking structure of a network. *Proximity* is the most emphasized property in the embedding of homogeneous graphs. Roughly speaking, proximity estimates the closeness between nodes, and thus a proper network embedding model shall assign similar embeddings to nodes close to each other in a graph. Preserving proximity will allow the model to accurately perform some relevant tasks such as community detection (e.g. closer nodes shall be grouped together) and link discovery (e.g. nodes with common neighbors should be linked). The other branch tries to obtain the embeddings of nodes in a *multi-relational graph*,

i.e., *knowledge graph* (Figure 1 right) [3–5, 9, 11, 13–15, 18–20, 25–28, 34, 35, 39, 44, 47–49]. A multi-relational network assumes there are different types of links (e.g. different relationships between users) in the graph, where key information depicting entities are encoded. Each directed edge can be represented by a triplet (i, k, j), where node i connects to node j via relation k. For example, a node X may connect to a node Y with relation *teacher_of*, but it can link to another node Z with relation *colleague_of*. The majority of the models for multi-relational graph embedding use an individual embedding transformation for each type of edge relation. Then based on certain algebraic operation (usually through addition or multiplication) to combine embeddings of nodes and relations, the multiple relationships in the embedding space can be recovered.

Even though a homogeneous network can be regarded as a special case of multi-relational network with single type of relation, our preliminary experiments have found that existing multi-relational graph embedding methods do not necessary perform well on homogeneous networks. It is probably because most of the existing studies do not emphasize on preserving the proximity information in the network.

Thus, in this paper we propose a multi-relational graph embedding model that preserves not only the relation information but also the proximity information, with the goal to create a unified model that is suitable for embedding both homogeneous and multi-relational networks. Although these two branches of studies respectively propose embedding methods in its own domain, to our knowledge, there has not yet been an effective unified model that works for both homogeneous and heterogeneous scenarios, which is the goal of this paper. Moreover, we also realize that directly applying the existing homogeneous proximity models shall not work for our situation since they do not consider the existence of a variety of relations. Here we propose a novel idea to regularize the importance of each embedding dimension based on the relation in generating the score function for each relation triplet. That is, we argue that some embedding dimensions are more important in predicting a type of relation while others are more important for other relations. For instance, to determine whether an edge belongs to 'classmate' relationship, a latent dimension that represents 'school attended' is probably more important than a latent dimension that represents 'favorite movies'. However, the later shall be more critical to determine the 'friendship' relationship between nodes than the former. To achieve such goal, we propose a tensor-factorization solution to regularize the importance of each embedding dimension based on the type of relations, trained based on the observed relations in the network data.

Besides a general framework to embed either type of the networks, we further intend to incorporate node attribute information into the embedding space. In practice one can usually observe attributes (e.g. demographics or metadata) of nodes in a network. Since such attributes provide semantic depiction about nodes, they shall be considered together in the embeddings of nodes. For instance, in a social network a user node can sometimes be associated with some demographic information such as age or gender. There is no reason to not incorporate such information in the node embedding. For example, we hope the embeddings of two female nodes are closer than the embeddings of two opposite-sex nodes, given the other conditions are similar. Mathematically, such non-graphical information attached to a node can be formulated as a real-valued attribute vector for each node (note that a categorical attribute can always be expanded into multiple binary ones). We notice that some network embedding studies [12, 17, 30, 40, 45, 50, 51] have considered node attributes in learning embeddings, but most of the existing approaches require at least a few labels to guide the learning. Furthermore, a very limited number of multi-relational network embedding methods [18, 19, 48] consider node attributes, as we have not yet seen any benchmark evaluation dataset that contains attributes for nodes in multi-relational networks.

In this paper, we propose the **M**ulti-relation, **A**ttribute, p**R**oximity **I**nformed **N**etwork **E**mbedding (MARINE)[1], a probabilistic network embedding model whose goal is to preserve *the proximity property, the edge relation and the node attributes information* in a low-dimensional embedding space. To achieve the goal, MARINE maximizes the likelihood of not only the existing graph structure but also the observed node attributes. To elaborate, we assign a function to score each (source_embedding, relation_embedding, target_embedding) tuple and try to maximize the difference of scores between the observed tuples and the unobserved ones. The scoring function leverages the modeling of tensor factorization and embedding translation. We show that the tensor factorization component can preserve the proximity information for each type of relation, which aligns well with the goal of most homogeneous network embedding methods. Inspired by the results of word embedding [23], we also consider the embedding translation, realizing the relation information with the relative positions among embeddings. In addition, we require the learned node embeddings to maximize the generation likelihood of node attributes, which assumes the embeddings of two nodes shall be closer if they have similar attributes. Finally, we discover an effective usage of our framework as an ensemble framework for multiple embeddings, and conduct preliminary experiments to verify its usage in this direction.

We evaluate the model based on three different applications with eight real-world datasets. The applications include link prediction, and multi-label classification. We demonstrate that the designed objective based on maximizing the above ideas allows the proposed MARINE to outperform state-of-the-art competitors in both homogeneous networks and multi-relational networks in terms of effectiveness and robustness. With the support of additional attributes, MARINE can further boost its performance. We also provide the visualization on various embedding results as another form of evaluation. Last but not least, we show MARINE of complexity linear in the number of nodes or edges, which enables the efficient implementation in large-scale network data.

We summarize the contributions of this paper, as listed below:

- We propose a unified unsupervised embedding method, MARINE, to enhance the existing solutions for multi-relational network embedding models by preserving the proximity and attributes information in addition to various types of relations.
- Our proximity model is specifically designed for multi-relational data, which allows the dynamic adjustment of importance weights for embedding dimensions given different relations.

[1]The code is available via https://github.com/ntumslab/MARINE

- MARINE enjoys the running time linear to the number of nodes and edges. Our framework also enables the embedding ensembles. We conduct thorough experiments to justify the above claims.
- We empirically conduct several experiments on real-world networks, to examine how MARINE performs in the applications of link prediction and multi-label classification. The results show the superiority capability of MARINE in learning better network representations.

The rest of the paper is organized as follows. Section 2 briefly summarizes related work. Section 3 introduces MARINE in detail. Section 4 reports the experimental results. Finally we discuss and conclude our work in Section 5.

2 RELATED WORK

We first discuss existing studies in homogeneous network embedding. Most of the past models focus on preserving the proximity property as it captures the neighborhood topology information of nodes. Some important network mining tasks, such as link prediction and community detection, can be addressed by discovering proximity relations among nodes. Specifically, between any node pair (i, j), the h-order proximity estimates either the number of h-hop shortest paths from i to j or the probability of a h-hop random walk from i to j. Relevant studies based on proximity assume higher similarity between the embedding vectors of two nodes with high proximity. Singular value decomposition or matrix factorization is commonly used to learn such embeddings [1, 8, 12, 29, 36, 38, 40, 42, 50, 51] These models rely on a matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ pre-computed based on certain proximity estimation. Then let $\boldsymbol{A}$ be approximated by a low-rank matrix multiplication. Some models [7, 17, 41] consider deep neural network structures like auto-encoders to exploit their non-linear mappings between graph topology and embedding vectors. Inspired by recent word embedding algorithms [23], another class of homogeneous graph embedding methods [10, 30, 31, 52] gathers random walk sequences as training set. The similarity of two nodes is expected to be high if their positions are close in most of the random walk sequences. There are some common concerns about the existing approaches. First, several models [8, 12, 40, 42, 50, 51] require to factorize a dense matrix, and thus cannot be scale to large graphs as the complexity reaches $O(N^2)$. Second, some approaches [1, 10, 12, 30, 31, 38, 40–42, 45] assumes symmetric relations, that is, $\boldsymbol{x}_i^\top \boldsymbol{x}_j = \boldsymbol{x}_j^\top \boldsymbol{x}_i$ for edge (i, j), and therefore cannot be used to handle directed graphs. Third, as having been pointed out in LINE [37], it is hard for random walk-based methods to explain explicitly what graph properties are preserved in the embedding space. Our model is developed to address the above issues.

On the other hand, multi-relational network embedding approaches often focus on learning a vector $\boldsymbol{r}_k$ that represents the embedding of edge relation k. Based on different assumptions, previous studies propose various scoring functions given a relation embedding $\boldsymbol{r}_k$ and the embeddings $\boldsymbol{x}_i, \boldsymbol{x}_j$ of two adjacent nodes. It is further assumed that an observed edge (i, k, j) with nodes (i, j) and relation k shall obtain a higher score value than an unobserved edge, and a multi-relational graph can be regarded as a set of multi-relational edges. There are two major classes of the

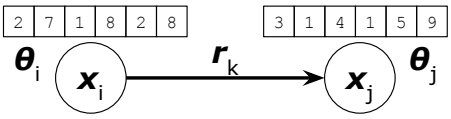


Figure 2: The notations in MARINE. There is an edge with relation k linking from node i to node j, whose attributes are θ_i and θ_j, respectively.

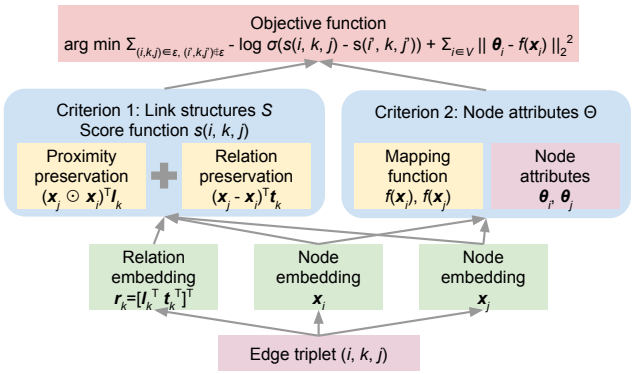


Figure 3: The overview of our proposed MARINE.

scoring functions. One is to build a bilinearity-like form $\boldsymbol{x}_i^\top \boldsymbol{R}_k \boldsymbol{x}_j$ or its non-linear extensions [3, 13, 27, 28, 34, 35, 39, 49] to learn the correlations between nodes (i, j) with respect to a relation k. By viewing an embedding vector as a point in multi-dimensional space, another set of scoring functions aim to constrain the positions of every embedding points [4, 5, 9, 11, 14, 15, 19, 20, 26, 44, 47, 48]. Given the transition scale $\boldsymbol{r}_k$ of a specific edge relation k, let it be fitted by the translation scale from point $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$. In other words, they try to minimize distance $\|f(\boldsymbol{x}_i) + \boldsymbol{r}_k - g(\boldsymbol{x}_j)\|$, where certain mapping functions f and g are added to map the node embedding space to the edge embedding space. We refer readers to a survey paper [25] that introduces recent multi-relational network embedding methods in details. To the best of our knowledge, the existing multi-relational network embedding works has not yet considered the proximity information.

3 METHODOLOGY

3.1 Problem Definition

Since homogeneous graph is a special case of multi-relational graph and undirected graph is a special case of directed graph (i.e. undirected graph can be regarded as a bi-directional directed graph), here our task is formulated over the general directed multi-relational graph. We also assume the nodes are associated with some attributes. Out task can be formulated as:

Definition 3.1. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$ denote a directed multi-relational graph, where $\mathcal{V}$ is a set of nodes, $\mathcal{R}$ is a set of edge relations, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ is a set of edges. Each node $i \in \mathcal{V}$ is appended with attributes represented as a vector $\boldsymbol{\theta}_i \in \mathbb{R}^a$.

Then our goal can be defined as:

Definition 3.2. Given graph $\mathcal{G}$, we would like to learn an embedding vector $\boldsymbol{x}_i \in \mathbb{R}^d$ ($d \ll |\mathcal{V}|$) for each node $i \in \mathcal{V}$ in an *unsupervised* manner.

Furthermore, for each type of edge in a multi-relational graph, we will learn an embedding vector $\boldsymbol{r}_k$ for its relation k. Figure 2 describes the notations. We expect the embedding would capture not only the proximity information but also the the information from link relation as well as the node attributes. In the next sections, we begin to introduce how MARINE learns embeddings with desired properties.

3.2 Design Criteria

Since the learning of embedding is generally unsupervised, here we formulate the task as an optimization problem. As stated previously, we would like to preserve both the link structure and node attributes information. The link structure information can be further decomposed into two components: proximity and relation. We will then formulate the criteria in a probabilistic form:

Criterion 1. **Link Structures** $\mathcal{S}$ preservation: Given a relation $k \in \mathcal{R}$, let an observed edge $(i,k,j) \in \mathcal{E}$ and a non-existing edge $(i',k,j') \notin \mathcal{E}$. Our model would like to assign scores $s(i,k,j) > s(i',k,j')$. Such score reflects the query: *how likely there is a relation of type k from node i point to j*.

Criterion 2. **Node Attributes** Θ preservation: We assume for any two nodes (i,j), the distance between their embeddings $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is positively correlated with the distance between their attributes $(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$.

Given the two design criteria, our task becomes a maximum a posteriori (MAP) problem to find the optimal embedding $\boldsymbol{x}$ and $\boldsymbol{r}$, given $\mathcal{S}$ and Θ. Using Bayes' Rule:

$$\begin{aligned}\arg\max_{\boldsymbol{x},\boldsymbol{r}} & p(\boldsymbol{x},\boldsymbol{r} \mid \mathcal{S},\Theta) \\ &= \arg\max_{\boldsymbol{x},\boldsymbol{r}} \frac{p(\mathcal{S},\Theta \mid \boldsymbol{x},\boldsymbol{r})p(\boldsymbol{x},\boldsymbol{r})}{p(\mathcal{S},\Theta)} \\ &= \arg\max_{\boldsymbol{x},\boldsymbol{r}} \underbrace{p(\mathcal{S},\Theta \mid \boldsymbol{x},\boldsymbol{r})}_{\text{Likelihood}} \underbrace{p(\boldsymbol{x},\boldsymbol{r})}_{\text{Prior}}. \end{aligned} \tag{1}$$

The denominator $p(\mathcal{S},\Theta)$ can be eliminated due to no involvement in the maximization in $\boldsymbol{x},\boldsymbol{r}$. To simplify the analysis, here we assume no prior knowledge about the target embeddings, and hence the prior distribution $p(\boldsymbol{x},\boldsymbol{r})$ is assumed to be *uniform* and can be ignored in the optimization process. Nevertheless, generalization to non-uniform prior is possible with careful design on the form of distribution. To make the joint probability p tractable, we adopt an independence assumption $\mathcal{S} \perp \Theta$ between $\mathcal{S}$ and Θ. We also assume that the edge relation embedding $\boldsymbol{r}$ is independent of to the node attributes given the node embedding, i.e., $\boldsymbol{r} \perp \Theta \mid \boldsymbol{x}$. Therefore we can simplify (1) as:

$$\begin{aligned} p(\mathcal{S},\Theta|\boldsymbol{x},\boldsymbol{r}) &= p_{\mathcal{S}}(\mathcal{S}|\boldsymbol{x},\boldsymbol{r})p_{\Theta}(\Theta|\boldsymbol{x},\boldsymbol{r}) \\ &= \underbrace{p_{\mathcal{S}}(\mathcal{S}|\boldsymbol{x},\boldsymbol{r})}_{\text{Criterion 1}} \underbrace{p_{\Theta}(\Theta|\boldsymbol{x})}_{\text{Criterion 2}}. \end{aligned} \tag{2}$$

We will elaborate $p_{\mathcal{S}}$ and p_{Θ} in Section 3.3 and 3.5.

The overview of our proposed MARINE is presented in Figure 3. Given an edge triplet, MARINE learns their representations based on the observations on link structures $\mathcal{S}$ and node attributes Θ. We will elaborate the details of each component in the following subsections.

3.3 Preserving Link Structures $\mathcal{S}$

$p_{\mathcal{S}}$ in Criterion 1 depends on the link structure. To fulfill this criterion, we formulate the following likelihood function with respect to edges:

$$\begin{aligned} p_{\mathcal{S}}(\mathcal{S} \mid \boldsymbol{x},\boldsymbol{r}) &= \prod_{\substack{(i,k,j)\in\mathcal{E} \\ (i',k,j')\notin\mathcal{E}}} \Pr\left(s(i,k,j) > s(i',k,j') \mid \boldsymbol{x},\boldsymbol{r}\right) \\ &= \prod_{\substack{(i,k,j)\in\mathcal{E} \\ (i',k,j')\notin\mathcal{E}}} \sigma\left(s(i,k,j) - s(i',k,j')\right). \end{aligned} \tag{3}$$

Inspired by [32], (3) can be also approximated by differentiable sigmoid functions $\sigma(z) = \frac{1}{1+\exp(-z)}$. As we maximize the likelihood (3), the gap between $s(i,k,j)$ and $s(i',k,j')$ is enlarged consequently, which satisfies the requirement of Criteria 1. The scoring function s is defined based on the corresponding embeddings $\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_{i'}, \boldsymbol{x}_{j'}, \boldsymbol{r}_k$ (see (4)) to preserve the proximity and relation information, as will be elaborated next.

3.4 Score Function Design

Criterion 1 demands the designed score function s to generate higher scores for an observed edge $(i,k,j) \in \mathcal{E}$, and lower ones for the non-existing counterpart $(i',k,j') \notin \mathcal{E}$. To incorporate both proximity and relation information, we design the score function as follows:

$$s(i,k,j) = \underbrace{(\boldsymbol{x}_j \odot \boldsymbol{x}_i)^\top \boldsymbol{l}_k}_{\text{Proximity preservation}} + \underbrace{(\boldsymbol{x}_j - \boldsymbol{x}_i)^\top \boldsymbol{t}_k}_{\text{Relation preservation}}, \tag{4}$$

where $\odot$ denotes the Hadamard product. We express edge relation embedding $\boldsymbol{r}_k = \begin{bmatrix}\boldsymbol{l}_k^\top & \boldsymbol{t}_k^\top\end{bmatrix}^\top \in \mathbb{R}^{2d}$ as the concatenation of two embedding vectors $\boldsymbol{l}_k \in \mathbb{R}^d, \boldsymbol{t}_k \in \mathbb{R}^d$, to achieve two different preservation targets in (4). The first term of (4) is essentially a tensor factorization to preserve proximity information. The second term implies that translation direction between embeddings shall be closer for the same edge relation. We do not set a relevant weight between two terms of (4), since the weight is absorbed in $\boldsymbol{l}_k$ or $\boldsymbol{t}_k$. In the following paragraphs, we will discuss the insight of (4).

3.4.1 Proximity Preservation. Most previous studies on homogeneous network embedding adopt matrix factorization (see Section 2) to learn embedding. Under such framework, given the embeddings $\boldsymbol{x}_i, \boldsymbol{x}_j$ of node pair (i,j), the existing studies mostly exploit inner product $a_{ij} = \boldsymbol{x}_i^\top \boldsymbol{x}_j = \sum_{u=1}^d x_{iu}x_{ju}$ to fit the proximity-related value a_{ij}. Random walk-based solutions also exploit the inner product operation to learn the similarities among embeddings. Using $\boldsymbol{x}_i^\top \boldsymbol{x}_j$ presumes a strong assumption that each dimension in the embedding space is equally important. Here we argue that for different types of relations, the importance or weight of each dimension to evaluate the proximity quality shall be different. We take a multi-relation social network where nodes are people as the example. Suppose that a single embedding dimension implies "salary" of

a node. Let us compare two different relations: "colleagues" and "friends". This particular dimension shall play a more important role to determine the former relationship than it is to the later one. It is because two colleagues are more likely to earn salary in the similar range than two friends. To model the influence of each dimension of embedding given different relations, we expand two-way matrix factorization into the *three-way tensor factorization*. The axes in this tensor indicate subject node i, edge relation k and object node j. Each binary element is set to 1 for edge $(i,k,j) \in \mathcal{E}$, and 0 for $(i,k,j) \notin \mathcal{E}$ Similar to the assumption in matrix factorization, a low rank d is assumed for this tensor. The score of the embeddings of tuples $\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{l}_k$ is evaluated by a tensor product:

$$\sum_{u=1}^{d} x_{iu} x_{ju} l_{ku} = (\boldsymbol{x}_i \odot \boldsymbol{x}_j)^\top \boldsymbol{l}_k. \tag{5}$$

Note that in our design, the importance of each embedding dimension is regularized by the learned relation embedding $\boldsymbol{l}_k$, meaning that different embedding dimensions can play different roles in determining whether a relation shall exist between two nodes. The next question we would like to answer is that *why such design can satisfy the proximity constraint*. Next we would like to establish the connections of our model to the first and second order proximity defined in homogeneous networks.

Below we first provide formal definitions to such proximity.

Definition 3.3. Given a pair of nodes (i,j), *the first-order proximity* for their embeddings $\boldsymbol{x}_i, \boldsymbol{x}_j$ means a high score between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in the embedding space, if there is an edge adjacent to i and j in the network.

Definition 3.4. Given a pair of nodes (i,j), *the second-order proximity* for their embeddings $\boldsymbol{x}_i, \boldsymbol{x}_j$ means a high score between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in the embedding space, if nodes (i,j) share the same neighbor set in the network.

Previous studies often adopt the inner product $\boldsymbol{x}_i^\top \boldsymbol{x}_j$ or the squared Euclidean distance $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2$ as the score function to achieve proximity preservation. Our design criteria guarantees to *explicitly* preserve the first-order proximity in a homogeneous network. By Criterion 1, given fixed relation k, we have $s(i,k,j) > s(i',k,j')$ for observed edge $(i,k,j) \in \mathcal{E}$ and unobserved edge $(i',k,j') \notin \mathcal{E}$. Modeling Criterion 1 in MARINE likelihood (3) satisfies the first-order proximity. Here we would like to prove that applying (5) to a homogeneous network (or a multi-relation network with single type of relation) can *implicitly* keep the second-order proximity.

Lemma 3.5. *Considering (5), MARINE* ***implicitly*** *preserves the* ***second-order proximity*** *in a homogeneous network.*

Proof. Given a fixed relation k, let two nodes (i,j) have the same set of direct successors. That is, for any node v ($v \neq i, v \neq j$), we have either $(i,k,v) \in \mathcal{E}$, $(j,k,v) \in \mathcal{E}$ or $(i,k,v) \notin \mathcal{E}$, $(j,k,v) \notin \mathcal{E}$. With the progress of training, the objective function eventually will cause observed edges and non-existing edges to become separable in scores. Then there must exist a threshold ϵ such that

$$s(i,k,v) \geq \epsilon \text{ as } (i,k,v) \in \mathcal{E},$$
$$s(i',k,v') < \epsilon \text{ as } (i',k,v') \notin \mathcal{E}$$

, and eventually we can obtain $\epsilon = 0$. If we consider only the tensor factorization part $s(i,k,v) = (\boldsymbol{x}_i \odot \boldsymbol{x}_v)^\top \boldsymbol{l}_k = (\boldsymbol{l}_k \odot \boldsymbol{x}_v)^\top \boldsymbol{x}_i$, then for the above-mentioned node v, we have:

$$s(i,k,v) \cdot s(j,k,v) = (\boldsymbol{l}_k \odot \boldsymbol{x}_v)^\top \boldsymbol{x}_i \boldsymbol{x}_j^\top (\boldsymbol{l}_k \odot \boldsymbol{x}_v) \geq \epsilon^2 = 0,$$

where $\boldsymbol{l}_k \odot \boldsymbol{x}_v \in \mathbb{R}^d$. It implies that matrix $\boldsymbol{x}_i \boldsymbol{x}_j^\top$ is a positive semi-definite matrix that has the non-negative trace property:

$$\boldsymbol{x}_i^\top \boldsymbol{x}_j = \text{tr}(\boldsymbol{x}_i \boldsymbol{x}_j^\top) \geq 0.$$

In practice, with sufficient number of training epochs, our scores (4) can move farther from the zero threshold such that $s(i,k,v) \cdot s(j,k,v) \gg 0$. It could lead to a high inner product $\boldsymbol{x}_i^\top \boldsymbol{x}_j$. In other words, $(\boldsymbol{x}_i, \boldsymbol{x}_j)$ could be close to each other in the embedding space, if nodes (i,j) share direct successors. Similar proof can applied to direct predecessors $(v,k,i), (v,k,j)$. □

3.4.2 Relation Preservation. This part was inspired by a word embedding approach [23] illustrating the distribution of word embeddings in a two-dimensional space. It implicitly reveals the two clusters of words, even though no label is provided to guide the learning. Between two clusters, there exists an implicit relation controlling the translation direction from one point to its adjacent point in the other cluster. In contrast to implicit relation learning in [23], we choose to put an explicit embedding translation term in our score metric (4), thanks to the availability of the relation information in multi-relational networks. Specifically, we expect that given an edge (i,k,j), the vector direction from $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$ shall be close to the relation embedding vector $\boldsymbol{t}_k$. The idea can be modeled by the inner product:

$$(\boldsymbol{x}_j - \boldsymbol{x}_i)^\top \boldsymbol{t}_k. \tag{6}$$

In real-world multi-relational networks, similar type of nodes might connect to similar types of others via similar relation (e.g. sales to customers). They are more likely to be clustered together, as shown in our visualization experiment.

As an additional bonus, the embedding translation term brings an asymmetric similarity metric, that is, $s(i,k,j) \neq s(j,k,i)$ by our definition (4). Notice that the tensor factorization term is symmetric since $(\boldsymbol{x}_i \odot \boldsymbol{x}_j)^\top \boldsymbol{l}_k = (\boldsymbol{x}_j \odot \boldsymbol{x}_i)^\top \boldsymbol{l}_k$. In a directed network, the direction of an edge determines an asymmetric relation between two nodes. Directed graphs are quite popular in real world. For instance, the 'following' information in an online social network is directional, meaning the opposite does not always hold. We observe that a few prior approaches [31, 38, 41] cannot leverage the edge direction information due to symmetric scoring functions. Using the asymmetric nature of the embedding translation term (6), we are allowed to model embeddings for directed graphs.

3.5 Node Attributes Θ

Based on Criteria 2, we need to design p_Θ such that node attributes $\boldsymbol{\theta} \in \mathbb{R}^a$ can affect the learning of embeddings $\boldsymbol{x} \in \mathbb{R}^d$. To satisfy the criterion, we define a bijective mapping function $\boldsymbol{\theta} = f(\boldsymbol{x})$ to bridge $\boldsymbol{\theta}$ and $\boldsymbol{x}$. Thus embedding $\boldsymbol{x}_i = f^{-1}(\boldsymbol{\theta}_i)$ could approach $\boldsymbol{x}_j = f^{-1}(\boldsymbol{\theta}_j)$ if the corresponding attribute vectors $\boldsymbol{\theta}_i \approx \boldsymbol{\theta}_j$. Here

we impose a multi-variate normal distribution for $\boldsymbol{\theta}$:

$$p_\Theta(\Theta \mid \boldsymbol{x}) = \prod_{i \in \mathcal{V}} \mathcal{N}(\boldsymbol{\theta}_i \mid f(\boldsymbol{x}_i), \frac{1}{2\alpha} I) = \prod_{i \in \mathcal{V}} \left(\frac{\pi}{\alpha}\right)^{-\frac{a}{2}} \exp\left(-\alpha \|\boldsymbol{\theta}_i - f(\boldsymbol{x}_i)\|_2^2\right). \quad (7)$$

where $\mathcal{N}(\boldsymbol{\theta} \mid \boldsymbol{\mu}, \Sigma)$ denotes a multi-variate normal distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix Σ. Since the maximum probability density appears in the mean for normal distributions, (7) can model the approximate mapping $\boldsymbol{\theta} \approx f(\boldsymbol{x})$ with uncertainty as variance $\frac{1}{2\alpha}$. Any differentiable bijective functions f are allowed, though in our experiments a naive linear mapping $f(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$ works well, which alleviates the need to tune potential hyperparameters of complex functions f. In the following sections, we use notation $\boldsymbol{\phi} \in \{\boldsymbol{W}, \boldsymbol{b}\}$ as the parameters in function f.

3.6 Objective Function and Optimization

Combining (3) and (7), when minimizing the negative logarithm of (2), we finally aim at solving:

$$\arg\min_{\boldsymbol{x}, \boldsymbol{r}, \boldsymbol{\phi}} L = \sum_{\substack{(i,k,j) \in \mathcal{E} \\ (i',k,j') \notin \mathcal{E}}} -\log \sigma\left(s(i,k,j) - s(i',k,j')\right) + \alpha \sum_{i \in \mathcal{V}} \|\boldsymbol{\theta}_i - f(\boldsymbol{x}_i)\|_2^2. \quad (8)$$

We remove the constant terms in parameters $\boldsymbol{x}, \boldsymbol{r}, \boldsymbol{\phi}$. For large-scale graphs the complexity is a concern, as there may exist quadratic number of edge pairs $((i,k,j),(i',k,j'))$ in (8). Fortunately we can adopt the commonly used *negative sampling* technique to sample a small subset of negative edge pairs for training. Referring to the idea in [32], we sample $Q \in \mathbb{N}$ negative edges $(i',k,j') \notin \mathcal{E}$ for each positive edge $(i,k,j) \in \mathcal{E}$, given relation k fixed. Viewing an edge pair $((i,k,j),(i',k,j'))$ as a training instance, we can apply Stochastic Gradient Descent (SGD) to efficiently update $\boldsymbol{x}, \boldsymbol{r}, \boldsymbol{\phi}$ in differentiable (8). For our objective function $\tilde{L}$, with respect to one training instance $((i,k,j),(i',k,j'))$, SGD optimizes a parameter z using the repetition of following update rule:

$$z \leftarrow z - \eta \frac{\partial \tilde{L}}{\partial z}, \quad (9)$$

where η denotes the learning rate that can be automatically adjusted by certain optimizers like Adam [16]. Here we present the derivatives of objective function $\tilde{L}$ over $\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{l}_k, \boldsymbol{t}_k, \boldsymbol{\phi}$ for SGD optimization.

$$\frac{\partial \tilde{L}}{\partial \boldsymbol{x}_i} = -\sigma\left(s(i',k,j') - s(i,k,j)\right)\left((\boldsymbol{x}_j \odot \boldsymbol{l}_k) - \boldsymbol{t}_k\right) - 2\alpha\left(\boldsymbol{\theta}_i - f(\boldsymbol{x}_i)\right)\frac{\partial f}{\partial \boldsymbol{x}_i} \quad (10)$$

$$\frac{\partial \tilde{L}}{\partial \boldsymbol{x}_j} = -\sigma\left(s(i',k,j') - s(i,k,j)\right)\left((\boldsymbol{x}_i \odot \boldsymbol{l}_k) + \boldsymbol{t}_k\right) - 2\alpha\left(\boldsymbol{\theta}_j - f(\boldsymbol{x}_j)\right)\frac{\partial f}{\partial \boldsymbol{x}_j} \quad (11)$$

$$\frac{\partial \tilde{L}}{\partial \boldsymbol{l}_k} = -\sigma\left(s(i',k,j') - s(i,k,j)\right)\left(\boldsymbol{x}_j \odot \boldsymbol{x}_i\right) \quad (12)$$

$$\frac{\partial \tilde{L}}{\partial \boldsymbol{t}_k} = -\sigma\left(s(i',k,j') - s(i,k,j)\right)\left(\boldsymbol{x}_j - \boldsymbol{x}_i\right) \quad (13)$$

$$\frac{\partial \tilde{L}}{\partial \boldsymbol{\phi}} = -2\alpha\left(\boldsymbol{\theta}_j - f(\boldsymbol{x}_j)\right)\frac{\partial f}{\partial \boldsymbol{\phi}} \quad (14)$$

3.7 Model Analyses

3.7.1 Complexity. The execution time of MARINE is determined by its SGD optimization as discussed previously. Given Q negative sample edges for each positive edge, we have $Q|\mathcal{E}|$ positive-negative edge pairs to optimize our parameters. Therefore the overall time complexity is $O(Q|\mathcal{E}|\omega)$ where ω is the number of epochs. Our experiments show $\omega = 500$ is sufficient for convergence. Since $Q \ll |\mathcal{E}|$ ($Q = 5$ in our experiments) and $\omega \ll |\mathcal{E}|$ in practice, MARINE enjoys time complexity that is linear to the number of positive edges $|\mathcal{E}|$, satisfying the scalability requirement. The training data contain $|\mathcal{E}|$ edges and $a|\mathcal{V}|$ attributes where each node has a attributes. As $a \ll |\mathcal{V}|$, MARINE possesses space complexity $O(|\mathcal{E}| + a|\mathcal{V}|)$ is efficient for practical applications.

3.7.2 Hyparameters. Observing the objective function in (8), there is only one hyperparameter α to be tuned by human, while all the other parameters can be learned during optimization. Note that a more complex mapping function f could lead to more hyperparameters. However our experiment results suggest that a linear mapping function without any hyperparameter can yield satisfiable results.

3.7.3 Embedding Ensemble. Even the external node attributes are not available, (7) can still be effective if we extract node attributes from the input graphs (e.g. degree information). Furthermore, the formula can be used as an embedding ensemble framework. That is, to treat the embedding outcomes of one model as the existing attributes Θ of the nodes. As will be described in the next section, positive experiment results support such idea.

4 EXPERIMENTS

4.1 Experimental Setup

Following a similar setup as most of network embedding studies, we conduct *Link Prediction* and *Multi-label Classification* tasks for evaluations. Three widely compared classic models (DeepWalk, LINE, TransE) and two of the start-of-the-art models (SDNE, ProjE) in network embedding are chosen to compare with MARINE. Note that DeepWalk, LINE and SDNE are models for homogeneous networks, while the other two are models for multi-relational networks. All the embedding dimensions are fixed to **128**; for the methods requiring negative sampling, we set the number of negative samples to **five** times of the positive samples (i.e, $Q = 5$) as mentioned in [10, 37]. Other hyperparameters are set to the default values suggested by the original papers.

- Competitors from homogeneous network embedding:
  - **DeepWalk** [31]: walk length $t = 80$; number of walks per source $\gamma = 10$; window size $w = 10$.

- **LINE** [37]: total number of samples $T = 10$ billion; 64-dimensional first-order proximity embeddings concatenated with 64-dimensional second-order proximity embeddings.
- **SDNE** [41]: first-order proximity weight $\alpha = 0.2$; reconstruction penalty $\beta = 10$; dimensionality of hidden layers $D = [1000]$.

- Competitors from multi-relational network embedding:
  - **TransE** [4]: margin in score function $\gamma = 1$; dissimilarity measurement $d = L_2$.
  - **ProjE** [34]: the original algorithm, denoted as 'ProjE-ori' in our tables, uses 25% of the total entities as negative samples suggested by the authors. However, to be fair in the comparison, we further compare with the performance of five negative samples per training instance (denoted as 'ProjE-Q5'). In addition, the *list-wise* version is selected since it reaches the best performance as reported in the paper.

While training our model, Adam optimizer [16] with suggested values (learning rate 0.001, parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$) is adopted. In all experiments, MARINE is trained for at most 500 epoches. All embeddings are initialized from samples of a uniform distribution ranged in $[-0.05, 0.05]$. For all experiment tasks, MARINE learns the representation without using node attributes, except for the experiments in Section 4.4.

4.2 Unsupervised Link Prediction

We use the same evaluation protocol as proposed in [4]. For each pair of an edge relation k and an object node j, we rank all triplets $(i, k, j)\ \forall\ i$ in the descending order of scores, based on the scoring function $s(i, k, j)$ of each model. The same procedure is then repeated for $(i, k, j)\ \forall\ j$. The performance is evaluated with respect to the hit rate at 10 and Mean Reciprocal Rank (MRR). According to the objective functions in the referenced papers, we review the scoring functions of the baseline models in Table 1.

We use three datasets for link prediction evaluation:

- Datasets used in previous approaches for multi-relational networks:
  - **WordNet (WN18)** [24] is a lexical database for English. It groups English words into sets of synonyms called synsets, and records a number of relations among these synonym sets. The relationships include *hypernyms*, *hyponyms*, *meronym*, and other lexical relations. WN18 has 18 relations, 40,943 nodes, and 141,442 edges.
  - **Freebase (FB15K)** [2] is a large collaborative knowledge base of general facts. For instance, a triplet *(Curtis Armstrong, place_of_birth, Detroit)* indicates that a human entity '*Curtis Armstrong*' and a city entity '*Detroit*' have the relationship '*place_of_birth*'. FB15k has 14,951 nodes, 483,142 edges, and 1,345 relations.
- Datasets used in previous approaches for homogeneous networks:
  - **Protein-Protein Interactions (PPI)** [6] is the dataset used in [10], which is the subgraph of the PPI network for Homo Sapiens. The original network has 3,890 nodes (proteins), 76,584 edges (interactions), and 50 labels representing the biological states. For link prediction task, we randomly remove 10% of existing edges for testing data while ensuring the residual graph is connected.

The experimental results are listed in Table 2. The relatively weak performance of DeepWalk, LINE, and SDNE is expected, since the information of relations (and their direction in edges) is not utilized. The results show that MARINE significantly outperforms all other competitors when 5 negative links are sampled. It is competitive to the state-of-the-art ProjE-ori which samples 25 percentage of the negative samples. We further increase the negative sample size of MARINE to 100, denoted as MARINE-Q100, and found the performance can further be improved. We also realized that our model performs significantly better on homogeneous network PPI, which reflects that the current multi-relational solution might not be the best solution for homogeneous network embedding.

Note that ProjE's performance drops significantly while training with fewer number of negative samples. The requirement of massive negative samples in ProjE results in quadratic training complexity, which is less suitable for large-scale networks.

From Table 2, we can find that the proximity preservation part in MARINE dominantly contributes to the performance. As the relation preservation part used in MARINE, the performance of MARINE in WN18 and FB15K, where edge muti-relations are provided, is relatively better than that in PPI. Besides, the overall performance can be significantly enhanced with the exploitation of both terms together. The relation term mainly focuses on learning different relations, while the link prediction evaluates the model to distinguish different nodes.

4.3 Multi-label Classification

We follow the same evaluation protocol as in [10]: The embeddings generated unsupervisedly by each model are used as the input training features to train a multi-label classification, using one-vs-rest logistic regression with L2 regularization[2]. The training and testing data are split equally, and the procedure is repeated 10 times to get the average. Micro F1-score and Macro F1-score are used to evaluate the performance. The regularization strength parameter $C = 1.0$ is not tuned, leaved as default value.

Three datasets are used for multi-label classification evaluation. The first two networks are homogeneous and the last one contains multiple relations.

- Homogeneous graphs including labels:
  - **PPI** [6] has been introduced in previous subsection.
  - **Wikipedia** [22] is the first billion bytes of the English Wikipedia dump. The processed data [10] builds a co-occurrence network of words, and the labels represent the Part-of-Speech tags. The network has 4,777 nodes, 184,812 edges and 40 labels.
- Multi-relational networks including labels:
  - **Movies** [46] is a complex network containing 41,412 entities with 20 labels (e.g., films, actors, directors) and 134,938 directed edges from 34 social relations (e.g., 'lived-with' and 'married-to').

[2] scikit-learn package LogisticRegression's LibLinear solver is used.

Table 1: Score functions of our competitors. σ here denotes the softmax function.

Model	Score function	Ranking order
DeepWalk	$\boldsymbol{x}_i^\top \boldsymbol{x}_j$	descending
LINE	$\boldsymbol{x}_i^\top \boldsymbol{x}_j$	descending
SDNE	$\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$	ascending
TransE	$\|\boldsymbol{x}_i + \boldsymbol{r}_k - \boldsymbol{x}_j\|_2$	ascending
ProjE	$\sigma(\boldsymbol{x}_i^\top \tanh(\boldsymbol{u} \odot \boldsymbol{x}_j + \boldsymbol{v} \odot \boldsymbol{r}_k + \boldsymbol{b}) + c)$	descending

Table 2: Link Prediction results. Models are evaluated by Hit Rate at top 10 (HR@10) and Mean Reciprocal Rank (MRR).

Dataset	PPI		WN18		FB15k	
	HR@10	MRR	HR@10	MRR	HR@10	MRR
DeepWalk	0.0131	0.0187	0.4165	0.1503	0.0369	0.0187
LINE	0.1726	0.0819	0.0926	0.0347	0.0757	0.0345
SDNE	0.1049	0.0571	0.1951	0.0621	0.0720	0.0291
TransE	0.2101	0.0934	0.9215	0.3678	0.6089	0.3449
ProjE-Q5	0.3905	0.2345	0.8557	0.5901	0.3076	0.1636
MARINE-Q5	**0.8602***	**0.5671***	**0.9253***	**0.6211***	**0.7202***	**0.4861***
ProjE-ori	0.7441	0.5339	0.9389	0.7664	0.6957	0.5126
MARINE-Q100	0.8973	0.6665	0.9402	0.7034	0.7424	0.4044

* outperforms the second-best model at 0.01 level paired t-test.

Table 3: Multi-label classification results with respect to Micro F1-score and Macro F1-score. The last row shows the case MARINE considering LINE's embeddings as attributes (see Section 4.4).

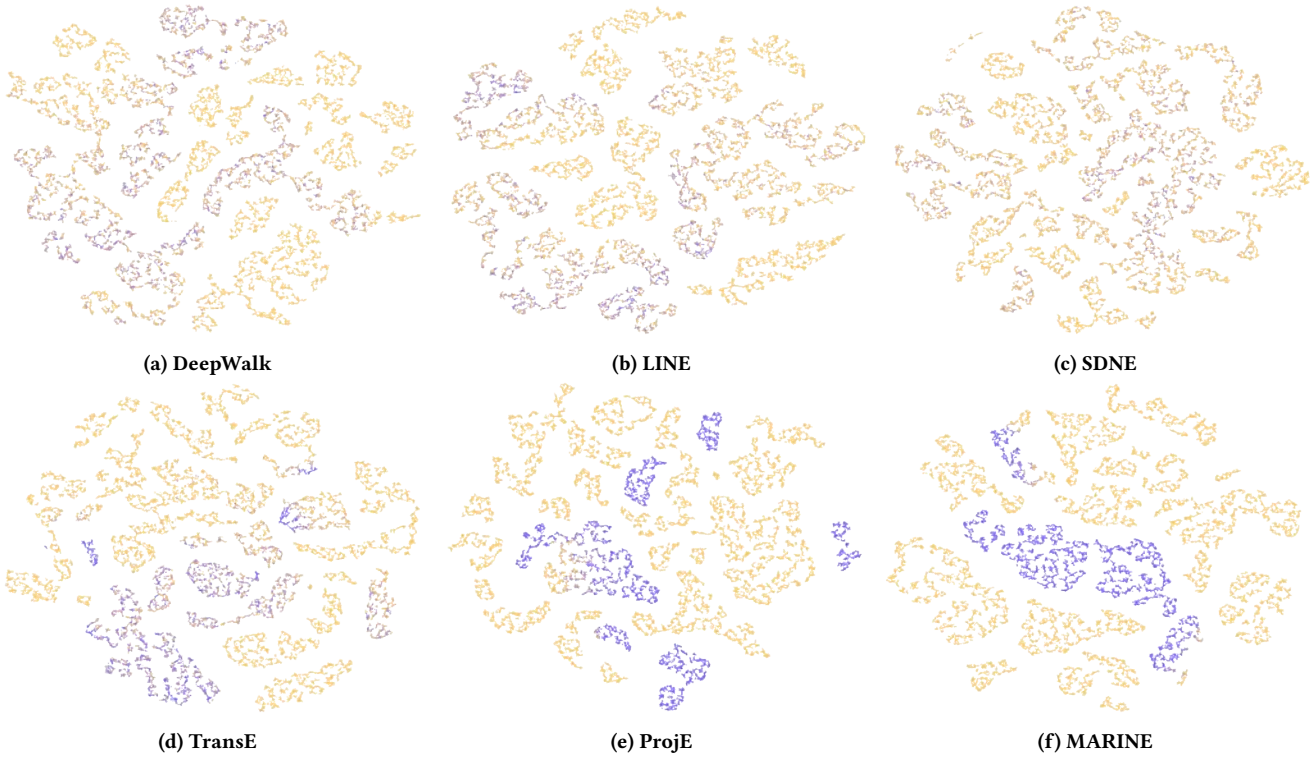
Dataset	PPI		Wikipedia		Movies	
	Micro	Macro	Micro	Macro	Micro	Macro
DeepWalk	0.0839	0.0631	0.3485	0.0595	0.0260	0.0156
LINE	0.0527	0.0332	0.3794	0.0694	0.6958	0.1136
SDNE	0.0021	0.0014	0.2811	0.0215	0.4871	0.0635
TransE	0.0355	0.0209	0.3312	0.0401	0.8932	0.3479
ProjE-Q5	0.0855	0.0683	0.3142	0.0678	0.9315	0.7181
MARINE-Q5	**0.0954***	**0.0723***	**0.4033***	**0.1044***	**0.9511***	**0.7315***
ProjE-ori	0.1083	0.0849	0.4134	0.1338	0.9049	0.6243
MARINE-Q100	0.1064	0.0829	0.4117	0.1219	0.9587	0.7925
Ensemble(MARINE, LINE)	0.1085	0.0822	0.4132	0.1137	0.9502	0.6617

* outperforms the second-best model at 0.01 level paired t-test.

The results are listed in Table 3. Again, given micro F1-score and macro F1-score, it is observed that MARINE significantly outperforms the others. In particular for the movie datasets that contain multiple relations, even MARINE-Q5 can outperform ProjE-ori significantly. For homogeneous networks PPI and Wikipedia, we can see that the most contribution to MARINE performance lies in the proximity preservation part, which consists with the perspective of previous homogeneous network embedding papers. By contrast, in the multi-relational network Movies the relation preservation part devotes more to MARINE in classification problems. We notice that using the original negative sampling strategy, ProjE can slightly outperform our MARINE. However we emphasize that ProjE samples $O(|\mathcal{V}|^2)$ negative node pairs to achieve high performance, which is not scalable at all. Under the same condition of sampling only 5 negative edges per positive edge example, MARINE shall significantly perform better than ProjE.

4.4 Exploiting Node Attributes

In this section, we discuss whether considering the node attributes allows our model to further boost the performance. As the datasets we used previously do not contain attributes, here we exploit the HepTh co-author citation network consisting of 5,501 author and 17,286 paper entities to predict the future citation counts using the network embeddings. The attributes are the ages of papers in HepTh. Following the same setup as [43], we train the embeddings on citations before 1999 and predict the number of citations after 2000. Random Forest Regressor (number of estimators = 32) is used



(a) DeepWalk (b) LINE (c) SDNE

(d) TransE (e) ProjE (f) MARINE

Figure 4: Visualization of HepTh. The orange spots indicate papers, while the purple spots indicate authors.

to train on embeddings for prediction. Five-fold cross-validation experiment is conducted; MAE (Mean Absolute Error) is chosen as the evaluation metric. The baseline usage of attributes relies simply on concatenating the embeddings with the attributes for training. We can see from Table 4 that not all models can benefit from additional attributes, while our model yields significantly better results than simply concatenating attributes with embeddings.

We also want to evaluate the performance of embedding ensemble as described in Section 3.7.3. We try to combine our model with LINE by treating its embedding results as our features, since LINE performs well among classic baseline embedding models. The last row in Table 3 shows that based on the ensemble of MARINE and LINE, we can gain improvement on two of the three datasets. The only exception is for the Movies dataset, probably due to the large performance gap between these two models.

Table 4: MAE evaluation of predicting citation counts. The comparison of the usage of node attributes in each embedding model.

usage	w/o attributes	concatenate
DeepWalk	6.6568	6.1724
LINE	6.8721	6.8994
SDNE	6.0891	6.1112
TransE	6.2739	6.3335
ProjE-Q5	6.5164	6.4319
ProjE-ori	5.9996	5.8402
MARINE-Q5	5.9525	5.8201
MARINE utilizes attributes as Θ		5.7042

4.5 Visualization

Here we want to show how the embedding results are visualized in the low-dimensional space. We apply visualization tool t-SNE [21] on the embeddings learned from, for the ease of presentation, HepTh dataset. All the models learn the embeddings without attributes. As Figure 4 suggests, the models that consider multi-relational networks are able to better distinguishing between different types of nodes. Both MARINE and ProjE can separate different types of nodes, and MARINE further groups the same type of nodes (purple spots) together. It shows that MARINE considers the relations of edges while preserving proximity simultaneously.

5 CONCLUSIONS

In this paper, we consider the two main branches of network embedding models: for homogeneous networks and for multi-relational networks, or knowledge graphs. Our conclusion is that although the existing multi-relational network embedding methods can be adopted to homogeneous networks, the performance can further be boosted while considering the relation-based proximity, as proposed in this paper. Our experimental results suggest that by considering graph proximity, relations, and attributes altogether, it is possible to yield a more robust network embedding model suitable for different types of networks. Finally, comparing with the state-of-the-art

solution ProjE that requires significant amount of negative links sampled, our model enjoys linear time and space complexity during computation.

ACKNOWLEDGMENTS

This work is supported by Ministry of Science and Technology (MOST) of Taiwan under grants 108-2636-E-006-002, 107-2218-E-006-040, 107-2221-E-001-009-MY3, and 106-3114-E-002-008, by Air Force Office of Scientific Research and Asian Office of Aerospace Research and Development (AOARD) under award number FA2386-17-1-4038, and also by Academia Sinica under grant AS-TP-107-M05.

REFERENCES

[1] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J. Smola. [n. d.]. Distributed Large-scale Natural Graph Factorization *(WWW '13)*.
[2] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. [n. d.]. Freebase: a collaboratively created graph database for structuring human knowledge *(SIGMOD '08)*.
[3] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A Semantic Matching Energy Function for Learning with Multi-relational Data. *Machine Learning*.
[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. [n. d.]. Translating embeddings for modeling multi-relational data *(NIPS '13)*.
[5] Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. [n. d.]. Learning Structured Embeddings of Knowledge Bases *(AAAI'11)*.
[6] Bobby-Joe Breitkreutz, Chris Stark, Teresa Reguly, Lorrie Boucher, Ashton Breitkreutz, Michael Livstone, Rose Oughtred, Daniel H. Lackner, JÃijrg BÃđhler, Valerie Wood, Kara Dolinski, and Mike Tyers. 2008. The BioGRID Interaction Database: 2008 update. *Nucleic Acids Research* (2008), D637–D640.
[7] Shaosheng Cao, Wei Lu, and Qiongkai Xu. [n. d.]. Deep Neural Networks for Learning Graph Representations *(AAAI'16)*.
[8] Shaosheng Cao, Wei Lu, and Qiongkai Xu. [n. d.]. GraRep: Learning Graph Representations with Global Structural Information *(CIKM '15)*.
[9] Miao Fan, Qiang Zhou, Emily Chang, and Thomas Fang Zheng. [n. d.]. Transition-based Knowledge Graph Embedding with Relational Mapping Properties *(PACLIC '14)*.
[10] Aditya Grover and Jure Leskovec. [n. d.]. node2vec: Scalable feature learning for networks *(KDD '16)*.
[11] Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. [n. d.]. Learning to Represent Knowledge Graphs with Gaussian Embedding *(CIKM '15)*.
[12] Xiao Huang, Jundong Li, and Xia Hu. [n. d.]. Label Informed Attributed Network Embedding *(WSDM '17)*.
[13] Rodolphe Jenatton, Nicolas Le Roux, Antoine Bordes, and Guillaume Obozinski. [n. d.]. A Latent Factor Model for Highly Multi-relational Data *(NIPS '12)*.
[14] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. [n. d.]. Knowledge Graph Embedding via Dynamic Mapping Matrix. *(ACL '15)*.
[15] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. [n. d.]. Knowledge Graph Completion with Adaptive Sparse Transfer Matrix *(AAAI'16)*.
[16] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv* (2014).
[17] Thomas N. Kipf and Max Welling. [n. d.]. Semi-Supervised Classification with Graph Convolutional Networks *(ICLR '17)*.
[18] Denis Krompaβ, Stephan Baier, and Volker Tresp. [n. d.]. Type-Constrained Representation Learning in Knowledge Graphs *(ISWC'15)*.
[19] Yankai Lin, Zhiyuan Liu, and Maosong Sun. [n. d.]. Knowledge Representation Learning with Entities, Attributes and Relations *(IJCAI'16)*.
[20] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. [n. d.]. Learning Entity and Relation Embeddings for Knowledge Graph Completion *(AAAI'15)*.
[21] Laurens van der Maaten and Geoffrey Hinton. [n. d.]. Visualizing data using t-SNE *(JMLR '08)*.
[22] Matt Mahoney. 2011. Large Text Compression Benchmark. http://www.mattmahoney.net/dc/textdata
[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. [n. d.]. Distributed Representations of Words and Phrases and their Compositionality *(NIPS '13)*.
[24] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* (1995).
[25] Dat Quoc Nguyen. 2017. An overview of embedding models of entities and relationships for knowledge base completion. *ArXiv* (2017).
[26] Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. [n. d.]. STransE: a novel embedding model of entities and relationships in knowledge bases *(NAACL '16)*.
[27] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. [n. d.]. Holographic Embeddings of Knowledge Graphs *(AAAI'16)*.
[28] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. [n. d.]. A Three-way Model for Collective Learning on Multi-relational Data *(ICML'11)*.
[29] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. [n. d.]. Asymmetric Transitivity Preserving Graph Embedding *(KDD '16)*.
[30] Shirui Pan, Jia Wu, Xingquan Zhu, Chengqi Zhang, and Yang Wang. [n. d.]. Tri-party Deep Network Representation *(IJCAI'16)*.
[31] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. [n. d.]. Deepwalk: Online learning of social representations *(KDD '14)*.
[32] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. [n. d.]. BPR: Bayesian personalized ranking from implicit feedback *(UAI '09)*.
[33] Leonardo F.R. Ribeiro, Pedro H.P. Saverese, and Daniel R. Figueiredo. [n. d.]. Struc2Vec: Learning Node Representations from Structural Identity *(KDD '17)*.
[34] Baoxu Shi and Tim Weninger. [n. d.]. ProjE: Embedding Projection for Knowledge Graph Completion *(AAAI '17)*.
[35] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. [n. d.]. Reasoning With Neural Tensor Networks for Knowledge Base Completion.
[36] Han Hee Song, Tae Won Cho, Vacha Dave, Yin Zhang, and Lili Qiu. [n. d.]. Scalable Proximity Estimation and Link Prediction in Online Social Networks *(IMC '09)*.
[37] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. [n. d.]. Line: Large-scale information network embedding *(WWW '15)*.
[38] Lei Tang and Huan Liu. [n. d.]. Relational Learning via Latent Social Dimensions *(KDD '09)*.
[39] Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Falk Brauer. [n. d.]. Random Semantic Tensor Ensemble for Scalable Knowledge Graph Link Prediction *(WSDM '17)*.
[40] Cunchao Tu, Weicheng Zhang, Zhiyuan Liu, and Maosong Sun. [n. d.]. Max-margin Deepwalk: Discriminative Learning of Network Representation *(IJCAI'16)*.
[41] Daixin Wang, Peng Cui, and Wenwu Zhu. [n. d.]. Structural deep network embedding *(KDD '16)*.
[42] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. [n. d.]. Community Preserving Network Embedding *(AAAI'17)*.
[43] Yujing Wang, Yunhai Tong, and Ming Zeng. 2013. Ranking Scientific Articles by Exploiting Citations, Authors, Journals, and Time Information.
[44] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. [n. d.]. Knowledge Graph Embedding by Translating on Hyperplanes *(AAAI '14)*.
[45] Xiaokai Wei, Linchuan Xu, Bokai Cao, and Philip S. Yu. [n. d.]. Cross View Link Prediction by Learning Noise-resilient Representation Consensus *(WWW '17)*.
[46] Gio Wiederhold. 1999. Movies. https://kdd.ics.uci.edu/databases/movies/movies.html
[47] Han Xiao, Minlie Huang, and Xiaoyan Zhu. [n. d.]. TransG : A Generative Model for Knowledge Graph Embedding *(ACL '16)*.
[48] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. [n. d.]. Representation Learning of Knowledge Graphs with Hierarchical Types *(IJCAI'16)*.
[49] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. [n. d.]. Embedding Entities and Relations for Learning and Inference in Knowledge Bases *(ICLR '15)*.
[50] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. [n. d.]. Network Representation Learning with Rich Text Information *(IJCAI'15)*.
[51] D. Zhang, J. Yin, X. Zhu, and C. Zhang. [n. d.]. Homophily, Structure, and Content Augmented Network Representation Learning *(ICDM'16)*.
[52] Chang Zhou, Yuqiong Liu, Xiaofei Liu, Zhongyi Liu, and Jun Gao. [n. d.]. Scalable Graph Embedding for Asymmetric Proximity *(AAAI'17)*.

Gaps in Information Access in Social Networks*

Benjamin Fish
Microsoft Research
benjamin.fish@microsoft.com

Ashkan Bashardoust
University of Utah
ashkanb@cs.utah.edu

danah boyd
Data & Society
danah@datasociety.net

Sorelle A. Friedler
Haverford College
sorelle@cs.haverford.edu

Carlos Scheidegger
University of Arizona
cscheid@cscheid.net

Suresh Venkatasubramanian
University of Utah
suresh@cs.utah.edu

ABSTRACT

The study of influence maximization in social networks has largely ignored disparate effects these algorithms might have on the individuals contained in the social network. Individuals may place a high value on receiving information, e.g. job openings or advertisements for loans. While well-connected individuals at the center of the network are likely to receive the information that is being distributed through the network, poorly connected individuals are systematically less likely to receive the information, producing a gap in access to the information between individuals. In this work, we study how best to spread information in a social network while minimizing this access gap.

We propose to use the maximin social welfare function as an objective function, where we maximize the minimum probability of receiving the information under an intervention. We prove that in this setting this welfare function constrains the access gap whereas maximizing the expected number of nodes reached does not. We also investigate the difficulties of using the maximin, and present hardness results and analysis for standard greedy strategies. Finally, we investigate practical ways of optimizing for the maximin, and give empirical evidence that a simple greedy-based strategy works well in practice.

CCS CONCEPTS

• **Networks** → **Online social networks**; • **Information systems** → *Social recommendation*; • **Theory of computation** → *Graph algorithms analysis.*

KEYWORDS

fairness; influence maximization; social networks

ACM Reference Format:
Benjamin Fish, Ashkan Bashardoust, danah boyd, Sorelle A. Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. 2019. Gaps in Information Access in Social Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313680

*This research was funded in part by the NSF under grants IIS-1633387, IIS-1633724, IIS-1513651, and IIS-1526379.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313680

1 INTRODUCTION

Information flow in networks has been a subject of extensive study. Among the many motivations for the study of how information propagates in a network has been advertising (how can we spread information most effectively on a budget) and clustering (how do groups form and organize in a network).

One of the most important questions in this area is how to maximize influence in a social network. Here the goal is to choose where to place initial sources of information so as to maximize the flow of information via word-of-mouth. First formalized by Kempe, Kleinberg, and Tardos [21], there has been a long series of work in the literature on influence maximization.

However, this work has not typically focused on the impact that the information has on the individuals in the network. For example, one important application of information flow in networks is for recruitment. Social networks like LinkedIn are increasingly used to provide access to jobs and information that can greatly impact an individual's career development. Often just as important as the individuals themselves are the connections *between* individuals – their social networks – in making hiring decisions. This is because information transmitted amongst social networks may accrue amongst the best-connected individuals in the network. As the adage goes, "it's not what you know, but who you know." With more and more of our social life mediated through online networks, the role that networks play in opening up opportunities is increasingly important. This includes not only recruitment, but also advertising and other kinds of marketing.

However, network structure can create haves and have-nots in the game of access. Insiders who are well-connected in the network have easier access to relevant information about opportunities for advancement that can in turn lead to even better connections. Outsiders who lack access to such information will find it much harder to improve their network status. This access gap may lead to a form of inequality that is different from the traditional forms of inequality based on class, race, gender, or other attributes, but nonetheless provides a significant challenge.

Thus, we are concerned with each individual's access to information and not just the number of people reached or the amount of information being distributed. How might we ensure that the access gap in information is reduced? Rather than asking how far we can spread information on a budget, we instead ask which people are getting the information we're spreading.

1.1 Our Work

How can we formulate a notion of equitable access to information in a network, and how might we intervene in a network (on a budget) to minimize the gap in access to information? In particular, we examine how best to add seeds (individuals who start with the information) to a network to minimize this gap in access.

We propose a new measure of access in a network. In contrast to previous work that maximizes the average probability that an individual receives the information (*max reach*), we instead propose to maximize the minimum probability. We formalize access as a *social welfare function* that assigns a real value to the set of utilities received by the individuals, in this case the probabilities of receiving the information. This allows us to evaluate the notions of access themselves: we consider a notion of access to be better if interventions that optimally maximize that notion do not widen the access gap. We show that every notion of access (amongst a wide class of such functions) does to some degree permit the access gap to increase in the worst case. On the other hand, if the access gap increases between two groups of individuals after an intervention, we show that our proposed notion of access at least prohibits situations where the access does not increase at all for the group which started off with less access to the intervention. Perhaps surprisingly, we show in Section 3 that a very large class of natural notions of access (including maximum reach) does not have this very basic prohibition. We desire this because without such a prohibition, in the worst case there's nothing stopping interventions from creating one permanently and significantly advantaged group with access to information and one group without any such access, which we regard as blatantly undesirable.

We show that maximizing the minimum probability is NP-hard, hard even to approximate well, and moreover that a number of standard greedy strategies have asymptotically worst-possible approximation ratios. Nonetheless, we show via experiments that a very simple greedy strategy performs well in practice: namely, choose the seeds to be the vertices currently estimated as having the smallest probabilities of receiving the information. We also demonstrate that by using this strategy, we decrease the correlation between vertices' probability of receiving the information and their location in the network, indicating that our measure of access is not merely a proxy for (static) network structure.

Limitations. We recognize that asking to maximize the minimum probability of access to information ignores the fact that not all individuals in a network might need a particular piece of information. For example, a hiring ad should be spread widely, but only to candidates who are eligible, are in the right geographic areas, and have desirable qualifications. More generally, interventions to improve access to information might themselves cause feedback loops (both virtuous and vicious): our work does not consider those dynamics. Nor does our work consider other notions of utility, like those that take into account the benefits of receiving the information more than once. We leave study of these issues for future work.

In summary, our main contributions are as follows.

- We propose a new measure of information access in a network. We demonstrate that this measure captures certain axiomatically desirable properties of any notion of equal access, and further that existing notions including the well-studied *maximum reach* concept do not.
- We investigate the problem of maximizing access theoretically, presenting hardness results as well as analysis of standard greedy strategies.
- We do a comprehensive empirical evaluation of heuristics for achieving a high level of access, demonstrating that a greedy-based strategy is quite effective at improving equality of access in a network for a given budget of interventions.

1.2 Related Work

Granovetter's seminal work on the strength of weak ties [16] first broached the idea that network position can confer advantages or disadvantages (including in hirings scenarios). Indeed, weak ties can influence success in hiring and careers [15]. In an algorithmic setting, boyd, Levy, and Marwick [4] illustrate how modern social networks like LinkedIn might be vehicles for a more direct propagation of advantage and disadvantage. In that light, our work, which focuses on how to mitigate such effects in the context of information access, falls into the paradigm explored by fairness-aware decision-making in which the goal is to design decision-making systems that ensure the end result is non-discriminatory to individuals or groups of individuals. Our work can be viewed as an attempt to quantify one aspect of *social capital*, a notion introduced by Coleman [7] to capture how social standing within a system could be interpreted as a resource that has utility for an agent. Recently, Benthall and Haynes [3] consider how to use a social network to define racial aspects of social standing, but don't consider interventions in the social network.

Rather than directly model an explicit fair goal for a decision in this setting, via assuming we have access to a sensitive feature like race on which we would focus our attention, we instead model the utility that each individual receives. This formalizes how best to optimize for access to information without necessarily requiring *equal* access. While most of the literature in algorithmic fairness uses equality-based definitions [9–11, 18, 27, 33, 36] (typically either group fairness or individual fairness), the welfare approach to fairness that we use is starting to become more popular. For example, Heidari et al. [19] propose a specific welfare function to use for classification and regression problems.

Our choice of welfare function is based on axiomatic considerations: by determining which functions satisfy specific mathematical criteria used to model gaps in access. The resulting function that seeks to maximize the minimum probability of receiving information bears some resemblance to the *difference principle* outlined by Rawls [30], in that it seeks to intervene so as to provide benefit to the "least-advantaged", here interpreted as those with the least probability of access.

Our work relies on a framework for information propagation that comes from the broad area of *influence maximization*. Influence maximization seeks ways to spread information in a network efficiently using a small collection of *seeds*. The typical measure of information spread used is the expected number of nodes that receive the information (the max reach measure). While influence maximization assigns the same utility to an individual as we do,

the welfare function in that setting is just the sum of the individual utilities. This *utilitarian* approach was initiated by Domingos and Richardson [31] and is formalized as a discrete optimization problem in Kempe, Kleinberg, and Tardos [21]. There is also work into making this process faster [5, 34] or suitable for more general situations, where factors like pricing must be taken into account [2].

A related body of algorithmic work [12, 25, 26] posits that one way to decrease polarization in social networks is to connect people with opposing views by exposing them to new information. Such work differs in focus and approach to modeling from this work because that work is concerned with poor connectivity between communities and we are concerned with individuals who are simply poorly connected.

2 DEFINITIONS

Let G be a graph with n nodes. To describe information flow in G we will use a standard probabilistic model for how information travels – the *independent cascade* (IC) model [21]. In this model, a node either possesses information or not. A set of *seed* nodes start out with the information, and information flow proceeds in rounds. Each newly informed node v informs its neighbors u in the next round i.i.d. with probability of transmission $\alpha_{u,v}$. Once a node is informed, it stays informed, and no longer passes on the message. In this work, we will use the IC model with a fixed probability α of transmission.

Welfare Functions. In the IC model with parameter α, we can associate with each vertex v the probability p_v that v is informed after all information has been passed. We now define a social welfare function $\mu : [0,1]^n \to \mathbb{R}$ to represent how effectively information is spread: it takes as input the probability of infection for each vertex, and outputs the overall welfare.

Definition 1. *The welfare of a set of vertices $V = \{v_1, \ldots, v_{|V|}\}$ in G with seed set S is $\mu_G(S, V) = \mu(p_{v_1}, \ldots, p_{v_{|V|}})$. If V is all n vertices, we abbreviate this as $\mu_G(S)$.*

When the graph is clear from context, we will omit the subscript G and write $\mu(S, V)$ and $\mu(S)$ respectively.

Seed sets represent an *intervention* in the information network. Thus, a primary goal in the study of information flow is to find a *budgeted intervention*: a set of seeds S_+ of size no more than k for a given graph G (possibly with initial seeds S) with maximum welfare

$$S^* = \arg\max_{\substack{S_+ \cup S: \\ |S| \le k}} \mu_G(S_+ \cup S).$$

In other words, S^* is the initial seeds S along with a set of k vertices which maximizes access for G. Later, we will also consider the set of seeds that maximize welfare for a particular set of vertices:

$$S_V = \arg\max_{\substack{S_+ \cup S: \\ |S| \le k}} \mu_G(S_+ \cup S, V).$$

Kempe, Kleinberg, and Tardos [21] and subsequent work use as their welfare function *reach*, the expected number of nodes reached. In our notation, and normalizing to make it conveniently $[0,1]$-valued, this becomes the following:

Definition 2 (Reach). $\mu_{reach}(S, V) = \frac{1}{|V|} \sum_{v \in V} p_v$.

We can easily generalize this to a wider class of notions of welfare. We consider generalized means:

Definition 3 (ϕ-mean). $\mu_\phi(S, V) = \left(\frac{1}{|V|} \sum_{v \in V} p_v^\phi \right)^{1/\phi}$.

Note in the limit, this becomes the geometric mean for $\phi = 0$, the minimum for $\phi = -\infty$, and the maximum for $\phi = +\infty$. In other words, $\mu_{-\infty}(S, V) = \min_{v \in V} p_v$.

We say that a function $\mu_G(S, V) = \mu(x_1, \ldots, x_m)$, each $x_i \in [0,1]$ representing the probability that a node i receives the information, is *monotonically increasing* if $\mu(x_1, \ldots, x_m) \ge \mu(x'_1, \ldots, x'_m)$ when $x_i \ge x'_i$ for all i. A function μ is *strictly monotonically increasing* if $\mu(x_1, \ldots, x_m) > \mu(x'_1, \ldots, x'_m)$ when $x_i \ge x'_i$ for all i and in addition there is some j such that $x_j > x'_j$. μ is *symmetric* if $\mu(x_1, \ldots, x_m) = \mu(x_{\sigma(1)}, \ldots, x_{\sigma(m)})$ for all permutations σ.

In this work, we restrict our attention to symmetric, monotonically increasing welfare functions so that no vertex is privileged above the others and, all else equal, increasing an individual's probability of receiving the information is never undesirable. The ϕ-means are such functions. Moreover, if a continuous welfare function satisfies four natural conditions (symmetry, strictly monotonically increasing, independence of unconcerned agents, and independence of common scale[1]) as a consequence of the Debreu-Gorman theorem [8, 14] the only such welfare functions up to ordering over preferences are the ϕ-means [19, 32], as long as all probabilities are non-zero. In other words, at least in the case of connected undirected graphs, ϕ-means are an extremely wide class of symmetric, monotonically increasing welfare functions, making them a natural class to examine.

3 GAPS IN ACCESS

Optimizing a welfare function is a way to improve access to information in the aggregate. But our concern in this work is whether individuals or subgroups are being left behind in the process. Is it possible that even though an aggregate measure of information access is increasing, the gap in information access between groups is getting larger? In this section, we will focus on evaluating welfare functions with respect to information access properties we would like to ensure.

We now define the *access gap*, which captures how much better some individuals are doing than others.

Definition 4. *The* access gap *of a (non-trivial) partition V, V' of the vertices of a graph G with seed set S under a welfare function μ is*

$$\mu(S, V) - \mu(S, V').$$

Note we only define the access gap over bipartitions, rather than arbitrary subsets. This is to prevent the following situation: Given a partition V_1, V_2, V_3 of G and initial seed set S, suppose $\mu(S, V_1) = \mu(S, V_2)$ are both very large, but $\mu(S, V_3)$ is much smaller. Consider S^*, the optimal seed set for this graph, and suppose now $\mu(S^*, V_1) > \mu(S^*, V_2) = \mu(S^*, V_3)$. We now have a gap between the access of V_1 and V_2, but this gap was a by-product of significantly increasing the

[1] Independence of common scale means that the ordering over alternatives should not change when multiplying each probability by a common positive factor, and independence of unconcerned agents means that the ordering should be independent of a probability that doesn't change, i.e. if $\mu(x, x_1, \ldots, x_m) \ge \mu(x, x'_1, \ldots, x'_m)$, then $\mu(y, x_1, \ldots, x_m) \ge \mu(y, x'_1, \ldots, x'_m)$ for all y.

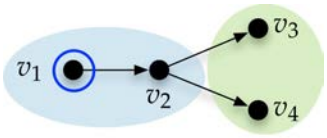


Figure 1: Example showing that the rich can get richer under the optimal intervention. If only one additional seed may be added, it is v_2 for any monotonic welfare measure. Under this intervention, the access gap between $\{v_1, v_2\}$ and $\{v_3, v_4\}$ (the two colored sets) widens.

access of V_3. Since this may well be desirable behavior, we preclude this situation by only considering gaps between bipartitions.

In particular, we want to know when the access gap increases. We call this the *rich getting richer* phenomenon.

Definition 5 (Rich get richer). *In a graph G with initial seeds S under a welfare function μ, we say that the* rich get richer *if there is a (non-trivial) partition V, V' where the optimal intervention S^* satisfies*

$$\mu(S^*, V') - \mu(S^*, V) > \mu(S, V') - \mu(S, V) > 0.$$

Unfortunately, stopping the rich from getting richer in arbitrary graphs may be too much to hope for. Even simple examples show that under many notions of welfare, including all ϕ-means, the rich get richer.

Proposition 3.1. *Suppose μ is symmetric, increasing, and satisfies the following condition: For any $x_1, \ldots, x_m$ in $[0, 1]$, there is some $1 \le \phi < \infty$ such that*

$$\min_i x_i \le \mu(x_1, \ldots, x_m) \le \left(\frac{1}{m}\sum_{i=1}^{m} x_i^{\phi}\right)^{1/\phi}.$$

Then under μ, when $0 < \alpha < \frac{1}{2\phi}$, there exists a graph and initial seed set where the rich get richer.

Note that the upper bound in this third condition is easy to satisfy; it suffices that $\mu(x_1, \ldots, x_m)$ is strictly less than $\max_i x_i$ when not all of the x_i are equal to each other. In addition the assumption that $\phi \ge 1$ is only assumed for the sake of convenience, since $\left(\frac{1}{m}\sum x_i^{\phi}\right)^{1/\phi}$ is monotonic in ϕ.

Proof. Consider the example graph G in Figure 1 and suppose $0 < \alpha < 1$. Let $V = \{v_3, v_4\}$ and $V' = \{v_1, v_2\}$. Note that $p_{v_1} = 1$, $p_{v_2} = \alpha$, and $p_{v_3} = p_{v_4} = \alpha^2$. Then $\mu(S, V) = \mu(p_{v_3}, p_{v_4}) = \mu(\alpha^2, \alpha^2) = \alpha^2$. Yet $\mu(S, V') = \mu(1, \alpha) \ge \alpha > \alpha^2$, so we have $\mu(S, V') > \mu(S, V)$.

What is the optimal seed to add? If we add v_2 to the seeds, then we have $p_{v_1} = p_{v_2} = 1$ and $p_{v_3} = p_{v_4} = \alpha$. Otherwise, if we add v_3 to the seeds, then $p_{v_1} = p_{v_3} = 1$, $p_{v_2} = \alpha$, and $p_{v_4} = \alpha^2$. Note $\mu(1, 1, \alpha, \alpha) \ge \mu(1, \alpha, 1, \alpha^2)$ by symmetry and monotonicity, so without loss of generality the optimal modification is to make v_2 a seed. Then it is easy to calculate $\mu(S^*, V') - \mu(S^*, V) = \mu(1, 1) - \mu(\alpha^2, \alpha^2) = 1 - \alpha^2$. Thus we have the rich getting richer if $1 - \alpha > \mu(1, \alpha) - \alpha^2$. But $\mu(1, \alpha) \le \left(\frac{1+\alpha^{\phi}}{2}\right)^{1/\phi}$, so it suffices to show that $\frac{1+\alpha^{\phi}}{2} < \left(1 - \alpha + \alpha^2\right)^{\phi}$. Then since $\phi \ge 1$ and $0 < \alpha < \frac{1}{2\phi}$,

$$\frac{1+\alpha^{\phi}}{2} \le \frac{1+\alpha}{2} < 1 - \phi\alpha + \phi\alpha^2 \le (1 - \alpha + \alpha^2)^{\phi}.$$

□

Proposition 3.1 holds for all ϕ-means for $\phi < \infty$. We will show in Section 3.1 that the rich get richer not only for the $+\infty$-mean but a whole other class of welfare functions as well (a consequence of Proposition 3.2). Given this, keeping the rich from getting richer appears to be too much to hope for.

3.1 k-imbalance

If we can't keep the rich from getting richer in the worst case, what can we prevent? A particularly concerning case of the rich getting richer is when the access of the worse-off group V doesn't improve at all. That is, a case where $\mu(S, V') > \mu(S, V)$ under the initial seeds S and the rich get richer, but for the set of seeds S^* that maximize welfare $\mu(S^*, V) \le \mu(S, V)$. This might not be so bad if the only way to improve the access of V is to increase the access of V to the point where it is even higher than that of V', so that V' becomes the worse-off group. On the other hand, this situation becomes particularly egregious when in addition $\mu(S_V, V) \le \mu(S, V')$, i.e. the optimal improvement for V still does not improve the access of V to the point where it is larger than the access that V' started out with prior to intervention (recall that S_V – defined in Section 2 – is the seed set that maximizes reach for V). If this can happen when adding k seeds, we will call μ *k-imbalanced*. That is, k-imbalance is a particularly egregious form of the rich getting richer. If μ is not k-imbalanced for any $k > 0$, we will call it *balanced*.

We believe that balance is a natural desideratum because it prevents interventions from never helping the worse-off group at all. Stronger versions of preventing disparity in access may still be preferred, like avoiding the rich from getting richer, so balance may only represent a necessary but not sufficient condition for preventing disparity. In this section, we show a wide class of μ are $\Omega(n)$-imbalanced, but that $\mu_{-\infty}$ is balanced.

Definition 6 (k-imbalance). *A welfare function μ is k-*imbalanced *if there exists a graph G with initial seed set S and partition of the vertices V and V' where the optimal intervention S^* and optimal intervention for S_V under the addition of no more than k seeds satisfies the following:*

(1) $\mu(S, V) < \mu(S_V, V)$ (There is a set of seeds to add that improves the access of V.)
(2) $\mu(S_V, V) \le \mu(S, V')$ (Not only does V' start off with more access than V, but V' starts off with more access than V can possibly achieve.)
(3) $\mu(S^*, V') > \mu(S, V')$ (The access of V' improves.)
(4) $\mu(S^*, V) \le \mu(S, V)$ (The access of V does not improve.)

In other words, a welfare function is imbalanced if

$$\mu(S^*, V) \le \mu(S, V) < \mu(S_V, V) \le \mu(S, V') < \mu(S^*, V').$$

Note that it is immediate that if μ is k-imbalanced for any $k > 0$, then the rich get richer under μ. As k increases, it should be the case that it becomes more difficult to find examples of imbalance, as it is harder to avoid improving the access of V'. Nonetheless, we can show that a wide class of welfare functions, including reach, is $\Omega(n)$-imbalanced:

Proposition 3.2. *Suppose μ is symmetric and strictly increasing. Then μ is $\Omega(n)$-imbalanced.*

Proof. It suffices to consider the simplest case: when there is no communication, i.e. G is the disjoint graph of n vertices. V and V' will each be exactly half of the vertices (for n even). The initial seed set S will be entirely contained in V' and will be size $n/4$. Now we will add an additional $n/4$ seeds. Note first that since μ is symmetric, each of the vertices (with the exception of the initial seeds) are identical. So S_V is any set of $n/4$ additional seeds in V: each additional seed must improve the welfare of V because μ is strictly increasing. But in this case, V and V' become identical, so we have $\mu(S, V) < \mu(S_V, V) \leq \mu(S, V')$. But by symmetry, the optimal seeds to add can be any $n/4$ vertices, in which case we can assume they are all in V'. Thus the welfare of V does not increase while the welfare of V' does. □

It turns out that balance is a useful definition, insomuch as it is actually possible to achieve.

Proposition 3.3. *$\mu_{-\infty}$ is balanced.*

Proof. Suppose $\mu_{-\infty}$ is imbalanced, witnessed by some partition V, V' of G and initial seed set S. Recall imbalance implies that $\mu_{-\infty}(S, V) < \mu_{-\infty}(S, V')$. Then by definition of $\mu_{-\infty}$, the vertex v with minimum probability is in V, i.e. $\mu_{-\infty}(S) = \mu_{-\infty}(S, V)$. Remember S^* maximizes the minimum probability, and $\mu_{-\infty}(S_V, V) > \mu_{-\infty}(S, V)$, so there is at least one graph that increases that minimum probability, which in turn means that S^* does as well. Thus $\mu_{-\infty}(S^*, V) > \mu_{-\infty}(S, V)$, a contradiction. □

On the other hand, $\mu_{-\infty}$ is a special case, and every other ϕ-mean is maximally imbalanced: there exists a graph, initial seed set, and partition of the vertices that verifies the other ϕ-means are imbalanced.

Proposition 3.4. *For $\phi > -\infty, \alpha < 1, \mu_\phi$ is $\Omega(n)$-imbalanced.*

Proof. If $\phi = +\infty$, so μ_ϕ is the maximum probability, then as soon as the graph has at least one seed, then $\mu_\phi(S) = 1$, and any added seeds after that don't change the value, so μ_ϕ is trivially $\Omega(n)$-imbalanced. Otherwise, if $\phi > 0$, μ_ϕ is strictly increasing, and from Proposition 3.2 we know it is $\Omega(n)$-imbalanced. And if $\phi \leq 0$, then μ_ϕ is strictly increasing once all probabilities are non-zero, at which point we use a similar tactic to when μ is strictly increasing, except we will need a connected graph. We will use the star graph, with one central vertex the seed, and all other vertices connected to that seed. In addition there will be some $n/2 - 1$ additional seeds, all in V', which consists of those seeds, the central seed, plus $n/2$ more vertices. V will be the other n vertices, so that G is $2n$ nodes. Our goal will be to add an additional $n/2$ seeds. Remember, since G is connected (all vertices have non-zero probability) μ_ϕ is strictly increasing. Then the optimal graph for V is to add all $n/2$ additional seeds to V, in which case we have $n/2$ vertices with probability 1 and $n/2$ vertices with probability α. But V' in G is exactly the same, so we have $\mu(S, V) < \mu(S_V, V) \leq \mu(S, V')$. However, all non-seeds are isomorphic, so we may assume all $n/2$ new seeds are added to V'. □

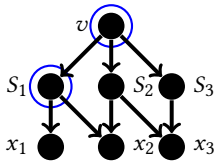


Figure 2: Corresponds with the set cover problem 'Is there $k = 1$ set among $S_1 = \{x_1, x_2\}$, $S_2 = \{x_2, x_3\}$, $S_3 = \{x_3\}$ that cover all elements?' Note that since the answer is no, then the minimum probability is no more than $p_{x_3} = 1 - (1 - \alpha^2)^2$.

We note that one could consider many variations of ϕ-means, including replacing mean with median, maximum with minimum, etc. These variations do not affect the results that we present here. We defer a detailed analysis of these variations to the full version of the paper.

4 MAXIMIN ACCESS

The previous section established $\mu_{-\infty}$ as a better access measure than others, at least when it comes to achieving balance. We now study the problem of maximizing $\mu_{-\infty}$, which we call the *maximin access* problem. We start by showing that this is NP-hard even to approximate well.

Theorem 4.1. *Suppose $\alpha < \frac{\sqrt{5}-1}{2}$. Then choosing k seeds to maximize min access is NP-hard. In this case, the maximin access cannot be approximated better than $O(\alpha)$ and if furthermore $\alpha = O(1/n)$ then the maximum cannot be attained efficiently without an additional $O(\ln n)$ factor seeds.*

Proof. We reduce from Set Cover, where an instance is defined by a collection of subsets $S_1, \ldots, S_m$ over a ground set $U = \{x_1, \ldots, x_n\}$ and an integer k, and the decision problem is whether or not there is a collection of k subsets whose union is U. Further, we can assume $k < n < m$. Given such an instance, we construct a directed graph (example showed in Figure 2). We start with the natural directed bipartite graph corresponding to a set cover instance, where there is a vertex i corresponding with each set S_i and a vertex j corresponding with each element x_j. There is a directed edge from i to j whenever x_j is contained in S_i. We then add a single extra vertex v and directed edges from v to each vertex i corresponding with one of the sets, and ask to maximize the minimum probability by adding $k + 1$ seeds.

Since v has in-degree zero, in order for the maximin access to be greater than zero, v must be chosen as a seed. In this case, since $k < m$, regardless of which seeds are chosen, there is some set S_i such that $p_i = \alpha$. Therefore the maximum min access is no more than α. Without loss of generality, no vertex corresponding to an element x_j need be chosen as a seed. Otherwise, the seed may be moved to any vertex corresponding with a set S_i such that $x_j \in S_i$. The maximin access cannot go down, because we still have $p_j \geq \alpha$.

If there is a set cover, then the maximum min access is at least α: choose the vertices corresponding to the cover for the seeds (plus v), in which case $p_v = 1$, $p_i \geq \alpha$ because they are either seeds or distance one from v, and $p_j \geq \alpha$, because they are distance one from a seed. If there is no set cover, then there is no way to choose the

seeds amongst the S_i such that all vertices are within distance one from a seed. Assume that every element x_j is contained in at most two subsets amongst the S_i (this is now the Vertex Cover problem, an NP-hard special case of Set Cover). So there must be some p_j such that $p_j \leq 1-(1-\alpha^2)^2$. Thus when $\alpha > 1-(1-\alpha^2)^2$, i.e. $\alpha < \frac{\sqrt{5}-1}{2}$, any algorithm that maximizes the min access chooses the set cover if there is one. So any algorithm that has an approximation ratio strictly better than $\frac{1-(1-\alpha^2)^2}{\alpha} = O(\alpha)$ must in fact be exact, and therefore also find the set cover.

Even in the general case of Set Cover, we can still distinguish between when there is and is not a set cover: The existence of a set cover still means the maximin probability is at least α, while the lack of a set cover implies there is at least one vertex with probability no more than $1-(1-\alpha^2)^m$, which is upper-bounded by α when $\alpha = 1/m$. Therefore, since set cover is $O(\ln n)$-inapproximable, we cannot approximate the best k seeds to add without an additional $O(\ln n)$-factor seeds. □

Moreover, if we can find the seeds that maximize the minimum probability, even approximately, we can boost this result to also compute the minimum probability itself approximately. This serves as additional evidence that this problem is hard, as there is no known method to even approximately compute the minimum probability.

Proposition 4.2. *If there is an $\alpha^{f(n)}$-approximation algorithm for maximin access, there is an $\alpha^{2f(n)+2}$-approximation for the minimum access of a vertex in a graph G given a seed s. That is, if the minimum access is p_{min} in G, then we can given an estimate $\hat{p}$ such that*

$$\alpha^{2f(n)+2} p_{min} \leq \hat{p} \leq (1/\alpha)^{2f(n)+2} p_{min}.$$

Proof. Given an instance (G, s, α), we construct a graph G' similar to the one in Figure 5. (We may assume that G is connected.) If the diameter of G is ℓ, add to G a simple undirected path of length ℓ starting from s, and call it G'. Call the end of this path v. In G, $p_{\min} \geq \alpha^\ell$, which means that if we compute the single optimal seed in G', it must be on the path from s to v.

Define x so that $\alpha^{\ell-x} = \alpha^x \cdot p_{\min}$, i.e. $x = \ell/2 - \frac{\log(1/p_{\min})}{2\log(1/\alpha)}$. Then the optimal placement for a seed is at distance k from s, where $\lfloor x \rfloor \leq k \leq \lceil x \rceil$, because for any k we have $p'_v = \alpha^{\ell-k}$ and $p'_{\min} = \alpha^k \cdot p_{\min}$, where p' denotes probabilities in G'.

Suppose that we have a $(1/\alpha)^{f(n)}$-approximation algorithm for maximin access, and it chooses some seed distance k' from s (we may assume that the seed is on the simple path, otherwise we may always choose $k' = 0$). Since it is a $(1/\alpha)^{f(n)}$-approximation on a simple path, k' must be within $f(n)$ of k. Now we can approximate $p_{\min}$ using k' as an estimate of k: We estimate it as $\hat{p} = \alpha^{\ell-2k'}$.

Then $\alpha^{\ell-2k'} \leq \alpha^{\ell-2(k+f(n)} \leq \alpha^{\ell-2(x+1+f(n))}$, and likewise $\alpha^{\ell-2k'} \geq \alpha^{\ell-2(x-1-f(n))}$, so this is within $\alpha^{2f(n)+2}$ of $p_{\min} = \alpha^{\ell-2x}$.

□

4.1 Maximin algorithms

The above results imply that it is hard to maximize $\mu_{-\infty}$ even approximately. Nonetheless, Theorem 4.1 still leaves open the possibility of an α^c-approximation (for fixed number of seeds and $c > 1$). In this section, we present the heuristics we will use, along with a few baselines. We will show in Section 4.2 that, unfortunately, these natural heuristics have a worst-possible approximation ratio (a ratio exponential in n). These results do not preclude good performance in practice, which we discuss in Section 5.

Making our task yet more challenging is that, unlike maximizing reach [21], maximin is not a submodular objective.[2] Nonetheless, it is natural to try a greedy approach, where in each iteration, we add to the seeds the vertex that maximizes the objective function. We refer to this heuristic as **Greedy** (Algorithm 1). To do this, we use the simple approach of estimating each probability p_v for every possible vertex to add to the seed set. (See below for details on how we estimate these probabilities.)

Algorithm 1 Greedy

Input: Graph G, initial seed set S, number of seeds to add k

for k iterations **do**
  for all $j \notin S$ **do**
    $prob \leftarrow \text{ProbEst}(G, S \cup \{j\})$ ▷ Algorithm 4
    $nextMin[j] \leftarrow \min_i prob[i]$ ▷ The minimum probability when the seeds are $S \cup \{j\}$
  $v \leftarrow \arg\min_j nextMin[j]$
  add v to S
return S

We contrast this approach to the faster approach, which we will call **Myopic** (Algorithm 2), whereby we instead in each round choose the vertex with the currently smallest probability as the new seed, without actually evaluating the new value of the objective function.

Algorithm 2 Myopic

Input: Graph G, initial seed set S, number of seeds to add k

$k' \leftarrow k$
if $|S| = 0$ **then**
  Initialize S as the vertex with the highest degree
  $k' \leftarrow k - 1$
for k' iterations **do**
  $prob \leftarrow \text{ProbEst}(G, S)$ ▷ Algorithm 4
  $v \leftarrow \arg\min_i prob[i]$ ▷ pick node with min probability
  add v to S
return S

We also consider a naïve variation (**Naïve Myopic**, Algorithm 3) which, instead of proceeding in rounds, given initial estimates for the probabilities, picks for the seeds the k vertices with the smallest probabilities.

So far, we have omitted how to estimate the probabilities for each vertex. Unfortunately, computing the probability p_v for each vertex exactly is #P-hard [29]. Even computing probabilities of receiving the information with a guaranteed approximation ratio is a long-standing open problem [20]. So in this paper, we use a Monte Carlo method, simulating the IC model a fixed number of times, and estimating the probabilities for each vertex as the

[2] This can be seen using the construction in the proof of Proposition 4.4, starting with one seed in the center of a simple path. Adding one additional seed then does nothing, but adding two seeds increases the minimum probability.

Algorithm 3 Naïve Myopic

Input: Graph G, initial seed set S, number of seeds to add k

$k' \leftarrow k$
if $|S| = 0$ **then**
  Initialize S as the vertex with the highest degree
  $k' \leftarrow k - 1$
$prob \leftarrow \text{ProbEst}(G, S)$ ▷ Algorithm 4
Add to S the k' vertices $i \notin S$ with smallest probability $prob[i]$
return S

percent of times the information reached that vertex under the simulations (Algorithm 4). Of course, this requires having at least one seed, which is not the case in the first round of Myopic and Naïve Myopic. So we always choose the first seed as vertex with the highest degree. This approach for dealing with the first round, as well as estimating the probabilities, provides a simple way to compare these heuristics. As such, for the experiments we also choose the first seed as the highest degree vertex for the Greedy heuristic as well, again to simplify comparison. We leave for future work other approaches for these issues.

Algorithm 4 ProbEst (Monte Carlo probability estimation)

Input: Graph G, seed set S
Parameters: α, Number of simulation rounds R

Initialize $hits[i] \leftarrow 0$ for each i a vertex of G
for R iterations **do** ▷ Simulate the IC model R times
  for all $i \in S$ **do**
    $hits[i]$++ ▷ $hits[i]$ is the number of times i has received the information
  $activeQueue \leftarrow S$ ▷ Keep track of which vertices are currently active
  while $activeQueue$ non-empty **do**
    Dequeue i from $activeQueue$
    for all neighbors j of i **do**
      $transmit \leftarrow$ True with probability α, else False
      if j has not been in $activeQueue$ and $transmit$ **then**
        $hits[j]$++
        Enqueue j to $activeQueue$
$prob[i] \leftarrow hits[i]/R$
return $prob$

An alternative approach that avoids estimating probabilities is to pick seeds that are far from each other, under the intuition that a node far away from the current seeds is likely to have a small p_i and therefore should be picked as the next seed. The resulting heuristic is to pick in each round the node that is furthest from the current set of seeds as the next seed; we call this heuristic **Gonzalez** because of its resemblance to the well-known algorithm for k-center clustering [13].

One could choose other proxies for the utility p_v such as nodes of low degree (or high degree), or nodes that do not contain seeds in a fixed radius ball around them. In our experiments with these heuristics, they were dominated in both quality and performance by the ones mentioned above, and we will not discuss them further.

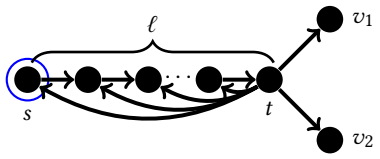


Figure 3: G

4.2 Approximation ratios of maximin algorithms

We now show that Myopic, Naïve Myopic, Greedy, and an exact version of Gonzalez all have approximation ratios that are exponential in n, even if we assume the probabilities required by Myopic, Naïve Myopic, and Greedy can be estimated exactly. This is to emphasize that their poor behavior in the worst case doesn't just stem from the difficulty of approximating the probabilities given a seed set, but the heuristics themselves.

4.2.1 Myopic and Naïve Myopic. Note that in the case $k = 1$, Myopic and Naïve Myopic are identical algorithms. Thus we can show that in this case, both algorithms behave poorly in the worst case, even in the non-trivial case when we start with at least one initial seed.

Proposition 4.3. *Given a graph and non-zero initial seed set, choosing as the seed with smallest p_v yields a solution with approximation ratio no better than $O(\alpha^n)$.*

Proof. Consider the graph G depicted in Figure 3. If we are allowed to add only a single additional seed besides the initial seed set $\{s\}$, then this algorithm will choose to add either v_1 or v_2, because in G they minimize $\min_v p_v$, where $p_{v_1} = p_{v_2} = \alpha^{\ell+1}$. But since we can only reach one of the two, we still have $\min_v p_v = \alpha^{\ell+1}$. But the optimal vertex to add to the seed set is t, where now $\min_v p_v \geq \alpha^2$. Then we get an approximation ratio no better than $O\left(\frac{\alpha^{\ell+1}}{\alpha^2}\right) = O(\alpha^n)$. □

4.2.2 Greedy. We now consider what happens if Greedy is used to choose the k seeds. One problem with Myopic was that, as demonstrated via Figure 3, choosing the vertex with the smallest probability ignores the actual objective function (which in that example is maximized by choosing vertex t). What happens when we attempt to maximize the actual objective function? Again, we assume that for any seed set we are given the exact probabilities instead of approximate probabilities, which we refer to as a *probability oracle*.

Proposition 4.4. *Greedy, even with a probability oracle, has an approximation ratio no better than $O(\alpha^{n/6})$.*

Proof. Consider the simple undirected path of length n, with no initial seeds, where we may add $k = 2$ seeds. The greedy algorithm, in the first iteration, must choose the central vertex (assume n is even). In the second iteration, no vertex can increase the minimum probability, so the minimum probability is $\alpha^{n/2}$. However, the optimal minimum probability is much larger: If the two seeds trisect the path so that they are $n/3$ apart, then no vertex is distance more than $n/3$ from a seed, in which case the minimum probability is at least $\alpha^{n/3}$. □

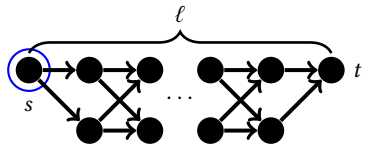


Figure 4: H_ℓ

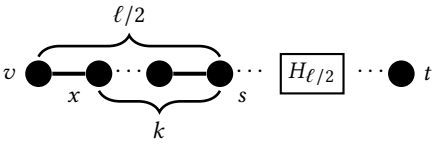


Figure 5: H, which consists of a simple path of length $\ell/2$, whose vertex s is the vertex s of in-degree 0 in $H_{\ell/2}$, depicted in Figure 4.

4.2.3 Minimax distance. Gonzalez is a heuristic to minimize the maximum distance of any vertex from a seed. One motivation behind this algorithm is that in Figure 3, adding an edge from s to t in G takes care of the issues found with Myopic by ensuring that all vertices have distance no more than two from the seed. In general, minimizing the maximum distance exactly is difficult, but even if we could do so, this approach still has a bad approximation ratio.

To show this, we construct a (sparse, max degree two) graph where nonetheless the vertex t furthest away from the seed still has a relatively high probability of receiving the information. This is the case for H_ℓ, shown in Figure 4, that's sufficiently sparse but p_t is large.

Lemma 4.5. *The probability of t being infected in H_ℓ, where each edge has weight $\alpha = 1/2$, is $p_t = \Theta(1/\ell)$.*

Proof. Denote by level k the vertices distance k from s, and by symmetry, the probability of being infected at that level p_k. We want to calculate p_ℓ. Note $p_0 = 1$ and $p_{k+1} = 1 - (1 - \alpha p_k)^2 = p_k - \frac{1}{4}p_k^2$, a variant of the logistic map.

Then $\frac{1}{p_{k+1}} = \frac{1}{\frac{1}{4}p_k(4-p_k)} = \frac{1}{p_k} + \frac{1}{4-p_k}$. Note $1/4 \le \frac{1}{4-p_k} \le 1/3$. Unwinding the recurrence, we get $\frac{1}{p_{k+1}} = \frac{1}{p_0} + \sum_{j=0}^{k} \frac{1}{4-p_j}$, and in particular we have $\frac{1}{p_0} + \frac{k+1}{4} \le \frac{1}{p_{k+1}} \le \frac{1}{p_0} + \frac{k+1}{3}$, i.e. $\frac{1}{p_k} = \Theta(k)$. □

Proposition 4.6. *The algorithm that minimizes the maximum distance from a seed has approximation ratio $O(\sqrt{n}/2^{n/6})$ when $\alpha = 1/2$.*

Proof. Suppose we can choose at most one seed in H, shown in Figure 5. Minimizing the max distance means the seed we use is s, and for sufficiently large ℓ the minimum probability is $p_v = \alpha^{\ell/2}$, at least for $\alpha = 1/2$ (using the previous lemma). However, the optimal seed to use is x, where x is a vertex $k \le \ell/2$ distance from s. Under this seed set, p_v remains the vertex with the minimum probability of getting infected so long as, for some constant c, $\alpha^{\ell/2-k} \le \frac{2c\alpha^k}{\ell}$ (again using the previous lemma). Solving for k to maximize the minimum probability, we get $k = \ell/4 - \frac{\log\left(\frac{\ell}{2c}\right)}{2\log\left(\frac{1}{\alpha}\right)}$. Then the approximation ratio is no better than $\frac{\alpha^{\ell/2+1}}{\alpha^{\ell/2-k+1}} = \alpha^k = \frac{\sqrt{\ell}\alpha^{\ell/4}}{\sqrt{2c}} = O(\sqrt{\ell}(1/2)^{\ell/4})$, and finally note H has $\frac{5}{2}\ell - 4$ edges, $\frac{3}{2}\ell + 1$ vertices, and the maximum in-degree (and out-degree) is two. □

Despite the hardness results of this section, we will show in the next section that these algorithms perform well in practice.

5 EXPERIMENTS

Our experimental evaluation will investigate the following question: does maximizing $\mu_{-\infty}$ create real changes in access? Is this different from the interventions achieved via maximum reach? And how effective are the proposed strategies for optimizing $\mu_{-\infty}$? Since our goal in this paper is to introduce and validate a method for reducing access gaps, we will not focus on achieving the fastest implementations (although we will compare the efficiency of different heuristics).

5.1 Experimental procedure

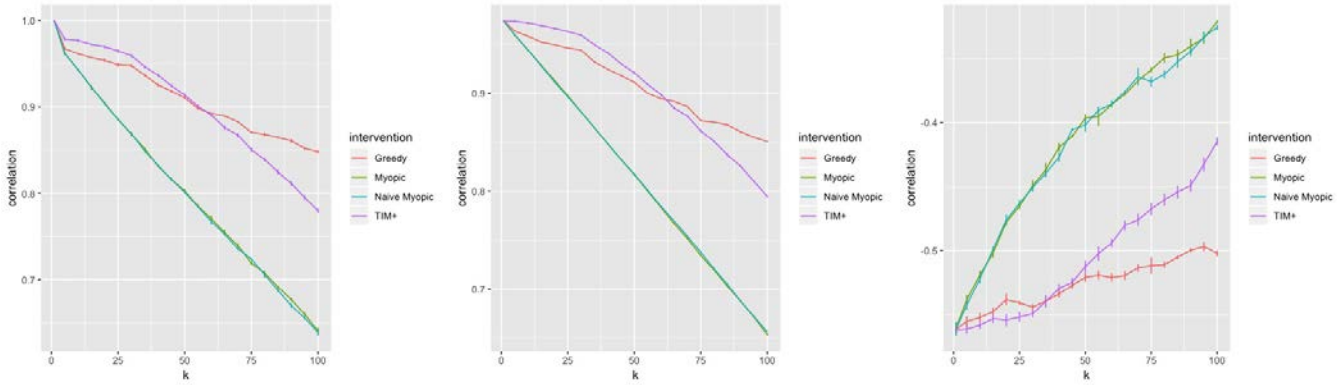
For our evaluation, we used social networks sourced from the SNAP [23] and ICON [6] repositories as described in Table 1.

$\mu_{-\infty}$ is a stringent objective function: it minimally requires having at least one seed in every connected component to achieve non-zero minimum probability, which may require a large number of added seeds if for example there are many disconnected nodes. Since the access gap is maximally large if there is at least one seed and a vertex with $p_v = 0$, we assume that the number of added seeds is large enough to cover all components of the graph. This allows us to add seeds to each component of the graph separately. As a simplifying assumption, in the experiments, we only consider the case (in directed graphs) when the components are strongly connected. In particular, rather than running the heuristics on all of the components, we just use the largest strongly connected component of the graph.

We also varied our intervention size between $k = 1$ and 100, independent of the size of the graph. This is a typical number of seeds used for interventions in the literature, and considering the application – recommending a job position – is a practical intervention size. We varied α – the probability of message transmission across an edge – in the range $\{0.1, 0.2, 0.3, 0.4, 0.5\}$[3]. Above this range information spreads so effectively that all algorithms are indistinguishable. Below this range the utilities p_v obtained are small enough that it is hard for Monte Carlo estimation to distinguish between them. We run 1000 simulations in order to estimate probabilities for any given seed set and repeat each heuristic 20 times, reporting the average result.

As a baseline, we used the algorithm TIM+ [35], which was designed to optimize maximum reach. While this procedure is not a true baseline (it does not directly optimize $\min p_i$), it provides insight into how existing methods for maximum reach might work in this newer setting. We also use as a baseline picking the k seeds uniformly at random (which we will refer to as Random).

[3]We report results for $\alpha \ge 0.3$ for brevity. Behavior below this range was similar.



(a) Initial probabilities (with one seed). (b) Degree of the vertex. (c) Distance from the center.

Figure 6: Correlations between the set of probabilities of access after intervention and three proxies for position in a network in the Arenas graph. Bars correspond to one standard deviation computed over 20 runs of each of the heuristics.

Table 1: Overview of the data sets we use.

Name	Nodes	Edges	Direction
EU [22]	803	24729	Directed
Arenas [1, 17]	1133	5451	Directed
Irvine [28]	1294	19026	Directed
Facebook [24]	4039	24729	Undirected
ca-GrQc [22]	4158	13428	Undirected
ca-HepTh [22]	8638	24827	Undirected

5.2 Maximin and network structure

In practice, what are the effects of using maximin over max reach as the objective? We give evidence that when maximizing reach instead of using maximin, interventions end up strongly reflecting the existing structure of the network. That is, vertices are more likely to become seeds if they are close to the center of the network, where probabilities of receiving the information are *already* high and do not need as many additional interventions.

We show this by measuring the correlation between the probability of receiving information before intervention versus after intervention. We use as a simple proxy for 'before intervention' the probabilities p_v when the vertex with the highest degree is the sole seed. Figure 6a shows the correlation between these two sets of probabilities in the Arenas graph, and indeed the correlation is significantly higher when using TIM+ than when using Myopic.

Assuming every vertex is equally deserving of information, we do not want 'well-positioned' vertices to have an advantage simply because they are well-positioned. Thus, we look at the correlation between the probability of information access after intervention and a few other proxies for position in a network. Figures 6b and 6c show the results for the degrees of the vertices as well as their distances from the center of the graph. Using TIM+, as the distance decreases towards the center or the degree of the node increases, the probabilities of information access increase, leading to a larger (negative) correlation. Again, this effect is lessened by using Myopic, whose resulting probabilities correlate less than TIM+ with both the degree of the vertex and the distance from the center. In other words, Myopic reduces the correlation between vertices' probability of receiving the information and how well connected the vertices are. Naïve Myopic yields very similar results to Myopic, as again seen in Figure 6.

In addition, Myopic changes the distribution of probabilities $\{p_{v_1}, \ldots, p_{v_{|V|}}\}$. Not only does it decrease the number of vertices with very low probability of receiving the information, but it also increases the number of vertices with larger probabilities over a broad range of probabilities, as seen in Figure 7.

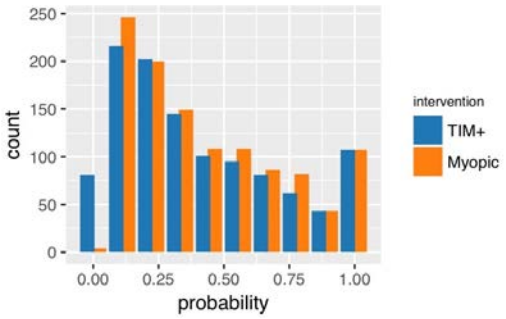
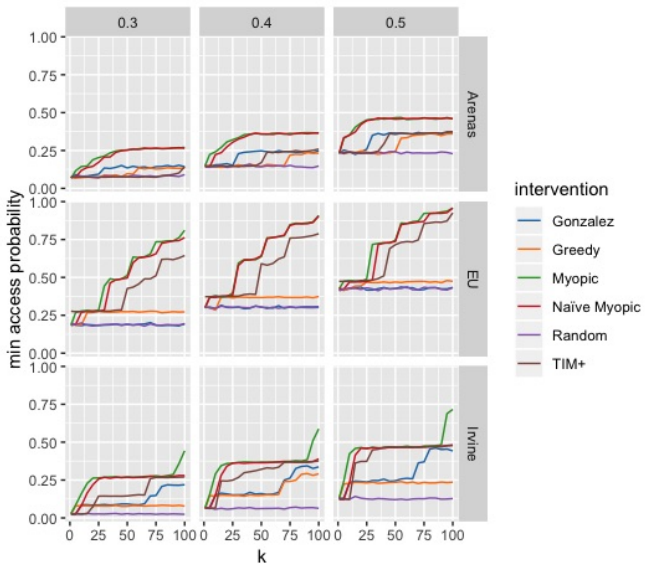


Figure 7: Distribution of probabilities over all vertices in the Arenas graph after adding 100 seeds with $\alpha = 0.1$.

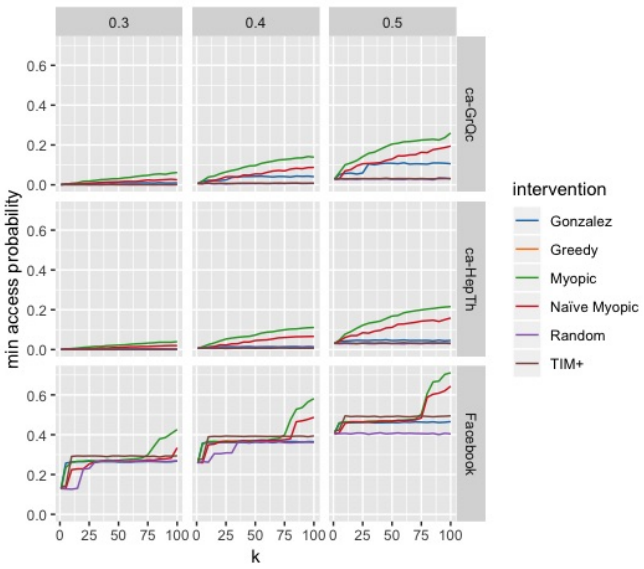
5.3 Heuristic performance

We now study the behavior of the heuristics described in the previous section. We would like to know how they compare in terms of effectiveness (maximizing $\mu_{-\infty}$) and speed.

We present effectiveness results in Figure 8. We omitted the heuristic Greedy when experimenting with larger data sets because it was prohibitively slow. Note that in both charts, the Myopic and Naïve Myopic heuristics consistently outperform the other methods for all ranges of α and intervention size k. The heuristics that do not use estimation are all consistently poor performers, and TIM+ performs well but is consistently dominated. For the smaller data sets, shown in Figure 8a, Greedy also does fairly well.



(a) Smaller data sets.



(b) Larger data sets.

Figure 8: Comparison of the six heuristics with respect to the minimum probability for values of $\alpha = \{0.3, 0.4, 0.5\}$.

Table 2: Speed of each of the heuristics on three data sets for 100 seeds. Times to completion are averaged over 20 runs.

Algorithm	Average time (s)		
	Arenas	EU	Irvine
Random	0.007	0.015	0.012
Gonzalez	0.021	0.031	0.033
Naïve Myopic	0.086	0.208	0.184
TIM+	0.876	1.826	1.046
Myopic	8.910	19.438	16.755
Greedy	507.35	759.296	1399.26

The running time of the heuristics is summarized in Table 2, which shows there is a natural tradeoff between running time and effectiveness. In particular, while the methods that make no use of estimation yield poorer quality results, they run extremely fast because they avoid the expensive step of estimating probabilities. Among the heuristics that estimate probabilities, Naïve Myopic is the fastest, with TIM+ also comparable, while the Myopic heuristic is an order of magnitude more expensive. Greedy is still another order of magnitude slower than Myopic, making it prohibitively expensive to compute in even relatively small graphs.

5.4 Performance on max reach

While the goal of the introduced heuristics is to maximize the minimum information access, it is also valuable to measure them by their average reach $\frac{1}{|V|}\sum_{v \in V} p_v$ to see if they are effective at spreading information to a large number of vertices. We compare the performance of Naïve Myopic and Myopic to TIM+ on this measure over three datasets (see Figure 9). The results show that while Naïve Myopic does not perform well to maximize reach, Myopic appears to outperform TIM+ even though TIM+ was designed for average reach and Myopic was not. This is likely because each seed added by Myopic is guaranteed to increase reach on the graphs, while algorithms that focus on maximizing reach may inadvertently provide access to nodes already reached. However, recall that Myopic is much slower than TIM+ (see Table 2) and so this potential improvement does not come without pitfalls. This tradeoff between average and minimum reach seems worthy of further study.

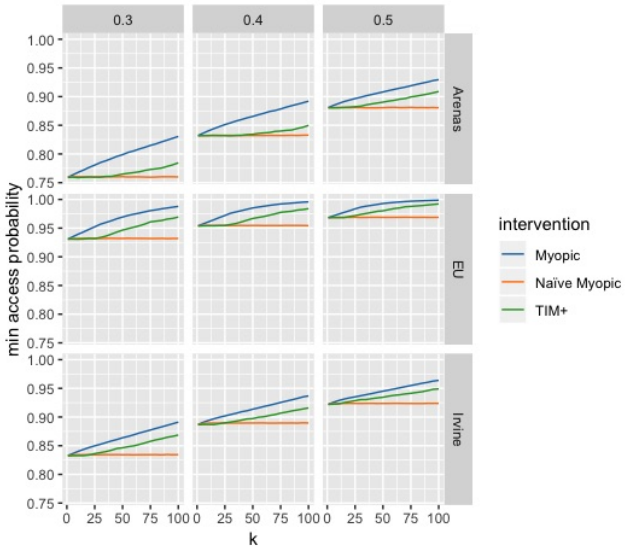


Figure 9: Comparison of three heuristics with respect to reach, the average probability after intervention, for $\alpha = \{0.3, 0.4, 0.5\}$.

REFERENCES

[1] 2017. U. Rovira i Virgili network dataset – KONECT. http://konect.uni-koblenz.de/networks/arenas-email

[2] David Arthur, Rajeev Motwani, Aneesh Sharma, and Ying Xu. 2009. Pricing Strategies for Viral Marketing on Social Networks. In *Internet and Network Economics, 5th International Workshop, WINE 2009, Rome, Italy, December 14-18, 2009. Proceedings*. 101–112.

[3] Sebastian Benthall and Bruce D. Haynes. 2019. Racial categories in machine learning. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. ACM, 289–298.

[4] Danah Boyd, Karen Levy, and Alice Marwick. 2014. The networked nature of algorithmic discrimination. *Data and Discrimination: Collected Essays. Open Technology Institute* (2014).

[5] Wei Chen, Yajun Wang, and Siyu Yang. 2009. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*. 199–208.

[6] Aaron Clauset, Ellen Tucker, and Matthias Sainz. 2016. The Colorado Index of Complex Networks. https://icon.colorado.edu/.

[7] James S Coleman. 1988. Social capital in the creation of human capital. *American journal of sociology* 94 (1988), S95–S120.

[8] Gerard Debreu. 1959. *Topological methods in cardinal utility theory*. Technical Report. Cowles Foundation for Research in Economics, Yale University.

[9] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness Through Awareness. In *Proc. of Innovations in Theoretical Computer Science*.

[10] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. *Proc. 21st ACM KDD* (2015), 259–268.

[11] Benjamin Fish, Jeremy Kun, and Ádám D Lelkes. 2016. A confidence-based approach for balancing fairness and accuracy. In *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 144–152.

[12] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. 2017. Reducing Controversy by Connecting Opposing Views. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*. 81–90.

[13] Teofilo F Gonzalez. 1985. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science* 38 (1985), 293–306.

[14] William M Gorman. 1968. The structure of utility functions. *The Review of Economic Studies* 35, 4 (1968), 367–390.

[15] Mark Granovetter. 1983. The strength of weak ties: A network theory revisited. *Sociological theory* (1983), 201–233.

[16] Mark S Granovetter. 1977. The strength of weak ties. In *Social networks*. Elsevier, 347–367.

[17] R. Guimerà, L. Danon, A. Díaz-Guilera, F. Giralt, and A. Arenas. 2003. Self-similar Community Structure in a Network of Human Interactions. *Phys. Rev. E* 68, 6 (2003), 065103.

[18] Moritz Hardt, Eric Price, Nati Srebro, et al. 2016. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*. 3315–3323.

[19] Hoda Heidari, Claudio Ferrari, Krishna P. Gummadi, and Andreas Krause. 2018. Fairness Behind a Veil of Ignorance: A Welfare Analysis for Automated Decision Making. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*. 1273–1283.

[20] David R. Karger. 1999. A Randomized Fully Polynomial Time Approximation Scheme for the All-Terminal Network Reliability Problem. *SIAM J. Comput.* 29, 2 (1999), 492–514.

[21] David Kempe, Jon M. Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 24 - 27, 2003*. 137–146.

[22] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1, 1 (2007), 2.

[23] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[24] Jure Leskovec and Julian J Mcauley. 2012. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*. 539–547.

[25] Antonis Matakos, Evimaria Terzi, and Panayiotis Tsaparas. 2017. Measuring and moderating opinion polarization in social networks. *Data Min. Knowl. Discov.* 31, 5 (2017), 1480–1505.

[26] Cameron Musco, Christopher Musco, and Charalampos E. Tsourakakis. 2018. Minimizing Polarization and Disagreement in Social Networks. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. 369–378.

[27] Arvind Narayanan. 2018. 21 fairness definitions and their politics. (Feb. 23 2018). Tutorial presented at the Conference on Fairness, Accountability, and Transparency.

[28] Tore Opsahl and Pietro Panzarasa. 2009. Clustering in weighted networks. *Social networks* 31, 2 (2009), 155–163.

[29] J Scott Provan and Michael O Ball. 1983. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM J. Comput.* 12, 4 (1983), 777–788.

[30] J. Rawls. 2009. *A Theory of Justice*. Harvard University Press.

[31] Matthew Richardson and Pedro M. Domingos. 2002. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*. 61–70.

[32] Kevin WS Roberts. 1980. Interpersonal comparability and social choice theory. *The Review of Economic Studies* (1980), 421–439.

[33] Andrea Romei and Salvatore Ruggieri. 2013. A Multidisciplinary Survey on Discrimination Analysis. *The Knowledge Engineering Review* (April 3 2013), 1–57.

[34] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence maximization: near-optimal time complexity meets practical efficiency. In *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*. 75–86.

[35] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence Maximization: Near-optimal Time Complexity Meets Practical Efficiency. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data (SIGMOD '14)*. ACM, New York, NY, USA, 75–86. https://doi.org/10.1145/2588555.2593670

[36] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. 2017. Fairness Constraints: Mechanisms for Fair Classification. In *Artificial Intelligence and Statistics*. 962–970.

Cross-domain Recommendation Without Sharing User-relevant Data

Chen Gao
Beijing National Research Center for Information Science and Technology (BNRist), Department of Electronic Engineering, Tsinghua University
gc16@mails.tsinghua.edu.cn

Xiangning Chen
Beijing National Research Center for Information Science and Technology (BNRist), Department of Electronic Engineering, Tsinghua University

Fuli Feng
School of Computing, National University of Singapore

Kai Zhao
Beijing National Research Center for Information Science and Technology (BNRist), Department of Electronic Engineering, Tsinghua University

Xiangnan He
School of Information Science and Technology, University of Science and Technology of China
xiangnanhe@gmail.com

Yong Li
Beijing National Research Center for Information Science and Technology (BNRist), Department of Electronic Engineering, Tsinghua University
liyong07@tsinghua.edu.cn

Depeng Jin
Beijing National Research Center for Information Science and Technology (BNRist), Department of Electronic Engineering, Tsinghua University

ABSTRACT

Web systems that provide the same functionality usually share a certain amount of items. This makes it possible to combine data from different websites to improve recommendation quality, known as the *cross-domain recommendation* task. Despite many research efforts on this task, the main drawback is that they largely assume the data of different systems can be **fully shared**. Such an assumption is unrealistic — different systems are typically operated by different companies, and it may violate business privacy policy to directly share user behavior data since it is highly sensitive.

In this work, we consider a more practical scenario to perform cross-domain recommendation. To avoid the leak of user privacy during the data sharing process, we consider sharing only the information of the item side, rather than user behavior data. Specifically, we transfer the item embeddings across domains, making it easier for two companies to reach a consensus (e.g., legal policy) on data sharing since the data to be shared is user-irrelevant and has no explicit semantics. To distill useful signals from transferred item embeddings, we rely on the strong representation power of neural networks and develop a new method named as NATR (short for ***N**eural **A**ttentive **T**ransfer **R**ecommendation*). We perform extensive experiments on two real-world datasets, demonstrating that NATR achieves similar or even better performance than traditional cross-domain recommendation methods that directly share user-relevant data. Further insights are provided on the efficacy of NATR in using the transferred item embeddings to alleviate the data sparsity issue.

CCS CONCEPTS

• Information systems → Collaborative filtering; **Recommender systems**; **• Security and privacy → Privacy-preserving protocols**.

KEYWORDS

Cross-domain Recommendation, Privacy Preserving, Deep Learning

ACM Reference Format:
Chen Gao, Xiangning Chen, Fuli Feng, Kai Zhao, Xiangnan He, Yong Li, and Depeng Jin. 2019. Cross-domain Recommendation Without Sharing User-relevant Data. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313538

1 INTRODUCTION

In the current Web ecosystem, it is common that some websites have a certain degree of homogeneity in their functionality and provided information. For example, there are many overlapped hotels on Trip.com and Booking.com, overlapped movies on IMDb and Douban[1], and overlapped products on Amazon and eBay. From the perspective of building recommendation services, it means

[1]Douban.com is a Chinese website that allows registered users to rate movies, music, books, etc.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313538

that the models for such two homogeneous domains are dealing with many items that are the same. This provides opportunities to improve the recommendation quality by enriching data. For example, if domain A does not have sufficient data on some items (*i.e.*, sparse or cold-start items) while the other domain B does have, e.g., a movie first released in US may have many ratings on IMDb but not on Douban, then the recommendation for these items on domain A can be potentially improved by integrating the data of domain B. The task of leveraging auxiliary data from other domains to improve recommendation quality of the target domain is known as the *cross-domain recommendation* [1, 10, 15, 32, 35, 43].

Existing works on cross-domain recommendation have primarily focused on directly aggregating data from multiple domains [1, 15, 32, 35, 43]. In other words, these methods assume that during model training of the target domain, user behavior data of other domains are directly accessible. For example, the representative Collective Matrix Factorization (CMF) [32] method extends Matrix Factorization (MF) by jointly learning user embeddings and item embeddings from the user-item interaction matrix of multiple domains. Despite effectiveness, the assumption that user behavior data can be fully shared across domains is questionable. Typically, different domains (websites) are operated by different companies, and thus it is difficult to let them share user behavior data due to the constraint of company policy.

In this work, we aim to provide a more realistic solution for cross-domain recommendation. To avoid any chance of leaking user privacy, we abandon the sharing of user-relevant data, neither behavior logs nor demographic attributes. However, this will pose challenges to transfer the collaborative filtering (CF) signal from one domain to another, since CF is typically modeled through mining user-item interaction data (e.g., user purchase and click logs). To address this technical challenge, we propose to share the item embeddings, which are learned by reconstructing user-item interaction matrix. The advantages are two-fold: 1) item embeddings can still encode certain CF signal by reflecting item similarities based on user behaviors (e.g., which items are frequently co-rated by users)[2], and 2) item embeddings are latent vectors that have no explicit semantics; as such, the risk of leaking user privacy can be kept to a minimum, which makes it easier for two companies to reach a legal policy for data sharing[3].

Our proposed solution, which has three steps, is illustrated in Figure 1. In the first step, an embedding-based recommender model, MF for example, is trained on the user-item interaction matrix of the auxiliary domain to obtain item embeddings. In the second step, item embeddings of the auxiliary domain are sent to the target domain; note that only the embeddings of overlapped items are necessary to be sent, which are subjected to the data-sharing policy between two companies. Finally, the target domain trains a recommender model with the consideration of the transferred item embeddings. The first two steps are straightforward to implement, and the main challenges lie in how to design a model to effectively incorporate transferred item embeddings in the last step. We summarize the key challenges as follows.

[2]A representative example of using the signal in item embeddings for recommendation is the item-based CF methods [13, 17].

[3]https://www.nytimes.com/interactive/2018/06/03/technology/facebook-device-partners-users-friends-data.html

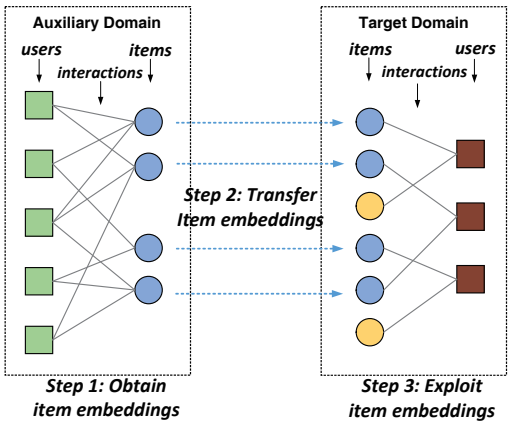


Figure 1: Illustration of our solution for cross-domain recommendation without sharing user-relevant data.

- **Unclear predictive signal of transferred item embeddings**. It is unclear whether and which transferred item embeddings contain useful signal in estimating a user's preference on an item in the target domain. Note that one motivation of conducting cross-domain recommendation is to alleviate the data sparsity problem in the target domain. However, the data sparsity problem may also exist in the auxiliary domain for some items, or the other way round the data in the target domain is already sufficient and does not require extra supplement. As such, it is challenging to distill useful signal from the transferred item embeddings and integrate them into the predictive model of the target domain.
- **Varying importance of transferred item embeddings**. As mentioned, the data of the auxiliary domain is not oracle — it is likely that user behaviors on some items are sparse and are insufficient to learn good embeddings for them. As such, it is a common case that the quality of item embeddings varies, where items of many users behaviors may have good quality and vice versa. Since it is already difficult to judge the quality of learned item embeddings for the auxiliary domain, it becomes even more challenging for the target domain to utilize such unknown- and varied- quality item embeddings well.
- **Embedding dimension discrepancy in latent space**. The data for training in the two domains may be of different scale and have different distribution. Therefore, the optimal embedding size for the two domains may be different. As such, existing cross-domain recommendation solutions that perform regularization on embedding matrices will fail [26]. Moreover, even though we restrict their embedding sizes to be the same, the semantics of their embedding dimensions are different and cannot be directly aligned.

To solve the above mentioned challenges, we design a novel model named **N**eural **A**ttentive-**T**ransfer **R**ecommendation (NATR). Briefly, our proposed method relies on the strong representation power of neural networks and discriminative power of attention

mechanisms to leverage the transferred item embeddings. Specifically, we design 1) a domain-level attention unit to dynamically adjust the importance of the predictive signal of the two domains; 2) an item-level attention unit to determine which embeddings of transferred items are more useful in constructing user representation for further prediction; and; 3) a domain adaption layer to bridge the discrepancy between the embedding space of the two domains. By tailoring our solution for addressing the highlighted challenges, our NATR method demonstrates its strong performance in cross-domain recommendation, and meanwhile preserves user privacy during data sharing.

To summarize, the main contributions of this work are as follows.

- We present a new paradigm for cross-domain recommendation without sharing user-relevant data, in which only item-side data can be shared across domains. To allow the transferring of CF signal, we propose to share the item embeddings which are learned from user-item interactions of the auxiliary domain.
- We propose a new solution NATR to resolve the key challenges in leveraging transferred item embeddings. The two-level attention design allows NATR to distill useful signal from transferred item embeddings, and appropriately combine them with the data of the target domain.
- We conduct extensive experiments on two real-world datasets to demonstrate our proposed method. More ablation studies verify the efficacy of our designed components, and the utility of transferred item embeddings in addressing the data sparsity issue.

The remainder of this paper is as follows. We first formulate the research problem in Section 2. We then elaborate our proposed method in Section 3. We conduct experiments in Section 4, before discussing related work in Section 5. Lastly, we conclude this paper in Section 6.

2 PROBLEM FORMULATION

We first introduce some notations used in the paper. We represent matrices, vectors, and scalars as bold capital letters (*e.g.,* $\mathbf{X}$), bold lower letters (*e.g.,* $\mathbf{x}$), and normal lowercase letters (*e.g.,* x), respectively. If not otherwise specified, all vectors are in a column form; $\mathbf{X}^T$ denotes the transpose of $\mathbf{X}$. We use symbols σ, *ReLU*, and $\odot$ to denote the *sigmoid* function, rectifier function, and element-wise production operation, respectively.

2.1 Cross-domain Recommendation

A typical problem setting of *cross-domain recommendation* is leveraging the data from an *auxiliary* domain to facilitate the recommendation quality in a *target* domain with overlapped items. In the *target domain*, where M and N denote the number of users and items, respectively, we have a user-item interaction matrix $\mathbf{Y}^t \in \mathbb{R}^{M \times N}$ with a binary value at each entry defined as,

$$y^t_{ui} = \begin{cases} 1, & \text{if } u \text{ has interacted with } i; \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Similarly, in the *auxiliary domain*, we have another binary user-item interaction matrix $\mathbf{Y}^a \in \mathbb{R}^{K \times L}$, where K and L are the number of users and items. Note that a portion of L items also occurs in the target domain, which are named as *bridge items*. From the interaction matrices $\mathbf{Y}^t$ and $\mathbf{Y}^a$, the goal of cross-domain recommendation is to learn a predictive function to estimate the likelihood that a given user u will interact with item i in the target domain.

2.2 Cross-domain Recommendation Without Sharing User-relevant Data

Distinct from the typical problem settings of cross-domain recommendation, we abandon the direct sharing of user behavior data (the user-item interaction matrix $\mathbf{Y}^a$ in the auxiliary domain). This is because directly sharing user behavior data may violate the business privacy policy of different companies operating the auxiliary and target domains. Instead, we propose a solution that only transfers the embeddings of bridge items which are offline learned in the auxiliary domain, as illustrated in Figure 1. We define the transferred item embedding matrix $\mathbf{Q}^a = [\mathbf{q}^a_1, \cdots, \mathbf{q}^a_N] \in \mathbb{R}^{D' \times N}$ as,

$$\mathbf{q}^a_i = \begin{cases} \tilde{\mathbf{q}}^a_i, & \text{if item } i \text{ is a bridge item;} \\ \mathbf{0}, & \text{otherwise;} \end{cases} \tag{2}$$

where $\mathbf{0} \in \mathbb{R}^{D'}$ is an all-zero vector and $\tilde{\mathbf{q}}^a_i \in \mathbb{R}^{D'}$ is the offline learned embedding of item i in the auxiliary domain. It should be noted that we organize the transferred item embeddings in $\mathbf{Q}^a$ in the same order of item IDs in the target domain to enable looking up an item embedding with its ID. Here we assume the availability of $\tilde{\mathbf{q}}^a_i$, *i.e.,* the company operating the auxiliary domain has employed an embedding-based recommendation system [19]. Note that the assumption is practical since embedding-based recommendation solutions are widely applied in the industry [4, 5].

After introducing the transferred item embeddings $\mathbf{Q}^a$ from the auxiliary domain, we formulate the problem of cross-domain recommendation without sharing user-relevant data as follows,

Input: The user-item interaction data in the target domain $\mathbf{Y}^t$, and the transferred item embeddings $\mathbf{Q}^a$ from the auxiliary domain.

Output: A predictive model to estimate the likelihood that a user u will interact with an item i in the target domain. Specifically, taking u, i, and $\mathbf{y}^t_u$ which is the interaction history[4] of u in the target domain, as input, the model has to predict,

$$\hat{y}^t_{ui} = f(u, i, \mathbf{y}^t_u), \tag{3}$$

where $\hat{y}^t_{ui} \in [0, 1]$ denotes the probability of interaction between user u and item i.

After obtaining the predictive model, we can use it to score all items for a given user u, and select the top-ranked (*i.e.,* with higher interaction probability) items as the recommendation results for u. It should be noted that there indeed exist user and item attributes in both the auxiliary and target domains. However, to simplify the scenario of the cross-domain recommendation task, we only emphasize the user-item interactions, which is a common setting of existing works [26, 43].

3 PROPOSED METHOD

To solve the problem of cross-domain recommendation without sharing user-relevant data, we rely on the strong representation ability of neural networks and devise a new solution, named NATR,

[4]Note that $\mathbf{y}^t_u$ is the transpose of the u-th row of $\mathbf{Q}^t$.

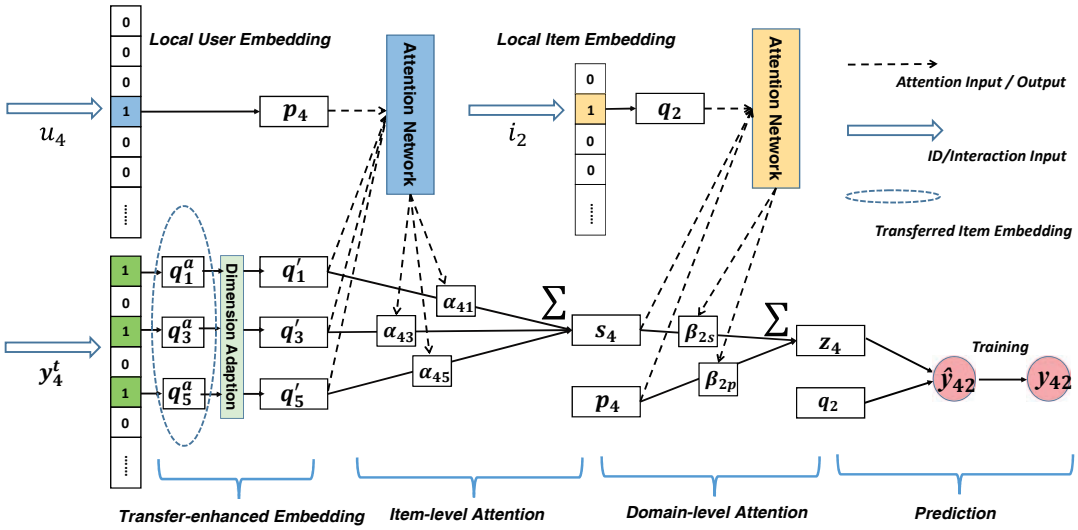


Figure 2: The architecture of our proposed Neural Attentive Transfer Recommendation model. (Here we take the prediction of user-item pair $(4, 2)$ as an example; user 4 has interacted with item 1, 3, and 5 in the target domain; then we utilize q_1^a, q_3^a, and q_5^a transferred from the auxiliary domain to assist predicting y_{42} in the target domain. Note that q^a is the only available auxiliary data in the target domain.)

exploiting the transferred item embeddings. Figure 2 illustrates the architecture of our proposed NATR model, which are made up of the following four layers.

- **Transfer-enhanced Embedding Layer**. We project sparse user and item representations into dense vectors. A dimension-adaption module is adopted to solve the dimension discrepancy problem of transferred item embeddings.
- **Item-level Attentive Layer**. To enrich user representations, we fuse the transferred embedding of items a user interacted with to an *additional user embedding* with an item-level attention unit to model the varying importance of items.
- **Domain-level Attentive Layer**. With a consideration of the diversity across domains, we make use of a domain-level attention unit to control the influence of predictive signals from two domains.
- **Prediction Layer**. Finally, we utilize an inner-product model as the predictive function since our work mainly focuses on devising a framework to exploit the transferred item embeddings.

In the following, we elaborate the details of the aforementioned four layers.

3.1 Transfer-enhanced Embedding Layer

Latent factor model (LFM) is one kind of general framework in collaborative recommender systems, which associates each user and item with real-valued vectors . Considering that LFMs have achieved success in a wide range of recommendation tasks [16, 17, 19, 24, 28], we project sparse user and item representations into real-valued vectors. Specifically, we first encode user ID (u) and item ID (i) into one-hot encodings as follows,

$$\mathbf{v}_u^U = \text{one-hot}(u), \mathbf{v}_i^I = \text{one-hot}(i), \tag{4}$$

where $\mathbf{v}_u^U$ ($\mathbf{v}_i^I$) $\in \mathbb{R}^N$ is a vector with all zero values except the u-th (i-th) entry with value 1. We then project the sparse one-hot encodings ($\mathbf{v}_u^U$ and $\mathbf{v}_i^I$) and multi-hot interaction history ($\mathbf{y}_u^t$) to *local embeddings* and *transferred embeddings*, respectively.

Local embeddings. To project the one-hot user (item) encoding, we employ an embedding layer, which is defined as a fully connected layer without bias term as follows,

$$\mathbf{p}_u = \mathbf{P}^T \mathbf{v}_u^U, \quad \mathbf{q}_i = \mathbf{Q}^T \mathbf{v}_i^I, \tag{5}$$

where $\mathbf{P} \in \mathbb{R}^{N \times D}$ and $\mathbf{Q} \in \mathbb{R}^{M \times D}$ are the parameters to be learned. The obtained embeddings $\mathbf{p}_u$ and $\mathbf{q}_i \in \mathbb{R}^D$ are named as *local embeddings* since they are learned merely with information from the target domain.

Transferred embeddings. In our problem, transferred embeddings of bridge items are the only auxiliary data accessible in the target domain. From the perspective of representative learning, there are two manners to leverage these item embeddings in collaborative filtering: user-based [24, 28] and item-based [13, 17]. Specifically, when predicting the probability that user u will choose item i in the target domain, user-based CF means directly combining embeddings of i of two domains to match $\mathbf{p}_u$ while item-based manner means matching $\mathbf{q}_i$ with transferred embedding of user's historically interacted items. There are two key aspects to make item-based manner a more convincing choice. First, when item i is not a bridge item, user-based CF solution cannot bring any help to prediction. Second, user-based CF can only distill implicit preferences while neglecting the explicit preferences of a user (*i.e.*, the historically

interacted items), while item-based CF can extract explicit preferences through leveraging transferred embeddings of bridge items and multi-hot encoding of historical interactions to enrich user representation.

Therefore, we look up the transferred embedding $\mathbf{q}_j^a$ from $\mathbf{Q}^a$ for each item j with $y_{uj}^t = 1$. Note that $\mathbf{q}_j^a$ will be an all-zero vector if item j is not a bridge item (see Equation 2 for details). As mentioned above, there may exist the challenge of *embedding dimension discrepancy* across two domains, *i.e.,* $D \neq D^{'}$, since the training data in two domains may be of different scale and have different distribution. To solve this problem, we employ a fully-connected layer to adapt the dimension of transferred embeddings, which is formulated as follows,

$$\mathbf{q}_j^{'} = \mathbf{W}_0^T \mathbf{q}_j^a + \mathbf{b}_0, \tag{6}$$

where $\mathbf{W}_0 \in \mathbb{R}^{D^{'} \times D}$ and $\mathbf{b}_0 \in \mathbb{R}^D$ are learnable parameters of mapping matrix and bias. Note that $\mathbf{W}_0$ and $\mathbf{b}_0$ are the only parameters here as the transferred embedding matrix $\mathbf{Q}^a$ is learned offline in the auxiliary domain.

With the above neural components, we project the sparse one-hot and multi-hot encodings into local user and item embeddings, $\mathbf{p}_u$ and $\mathbf{q}_i$, and transferred item embeddings, $\{\mathbf{q}_j^{'} | y_{uj}^t = 1\}$. We introduce our item-based CF solution of leveraging transferred item embeddings detailedly in the following parts.

3.2 Item-level Attentive Layer

The key objective of embedding-based recommendation model is to capture relation between of user and item in the latent space [41], therefore it is critical to explicitly build the relation between transferred item embeddings and local user embedding in our problem. As mentioned above, motivated by item-based CF [13, 17], which encodes the historical interaction behaviors of a user to enrich the user representation, we fuse the transferred item embeddings $\{\mathbf{q}_j^{'} | y_{uj}^t = 1\}$ into an *additional user embedding* $\mathbf{s}_u$. Besides historical interactions, $\mathbf{s}_u$ also contains CF signals transferred from the target domain, which can further enhance user representation. Our first inspiration to calculate $\mathbf{s}_u$ is average pooling, a widely used modeling component in neural networks, formulated as follows,

$$\mathbf{s}_u = \frac{\sum_{\{j | y_{uj}^t = 1\}} \mathbf{q}_j^{'}}{|\mathbf{y}_u^t|}, \tag{7}$$

where $|\mathbf{y}_u^t|$ is the l_1-norm of vector $\mathbf{y}_u^t$, which equals to the number of items user u has interacted with.

However, as mentioned above, there exists another key challenge of *varying importance of embeddings*. Considering that different interacted items have embeddings with varying quality and varying importance to represent the preference of a given user, such naive operation may not work well in real scenario. Therefore, we apply a non-uniform coefficient when fusing the transferred item embeddings:

$$\mathbf{s}_u = \frac{\sum_{\{j | y_{uj}^t = 1\}} \alpha_{uj} \mathbf{q}_j^{'}}{|\mathbf{y}_u^t|}. \tag{8}$$

To model the various item importance in a user-sensitive fashion, here we introduce attention mechanism, which has achieved great success in recommendation tasks [3, 13, 40] and natural language processing [20, 37]. Specifically, the item-level attention unit learns a specific weight α_{uj} for every transferred item embedding $\mathbf{q}_j^{'}$ according to the following formulation,

$$\begin{aligned} \alpha_{uj} &= \frac{e^{a_{uj}}}{\sum_{\{k | y_{uk}^t = 1\}} e^{a_{uk}}}, \\ a_{uj} &= \mathbf{w}_1^T ReLU(\mathbf{p}_u \odot \mathbf{q}_j^{'}) + b_1, \end{aligned} \tag{9}$$

where $\mathbf{w}_1 \in \mathbf{R}^D$ and b_1 denote the weight matrix and bias of a fully connected layer. The input of the item-level attention unit is the interaction between the user and target item, which makes the learned attention score sensitive to the given user.

3.3 Domain-level Attentive Layer

After obtaining the *local* and *additional* user embeddings, our task become learning a prediction function based on these three embeddings. Different from traditional recommendation models, here we have an extra user embedding. To exploit two user embeddings, we fuse them into an *unified embedding*. This is inspired by some cross-domain recommendation models [33, 43], which have demonstrated that fusing embedding vector learned from multi-modal data is a simple but yet effective way to combine signals. Another option is to separately estimate the interaction probability with the two embeddings and fuse the predictions (late fusion). Here we employ early fusion that merges embeddings, allowing us to explicitly capture the interaction between two embeddings.

The *unified embedding* via fusion can be denoted as,

$$\mathbf{z}_u = \beta_{si} \mathbf{s}_u + \beta_{pi} \mathbf{p}_u, \ s.t., \ \beta_s + \beta_p = 1, \tag{10}$$

where β_{si} and β_{pi} are learnable weights for $\mathbf{s}_u$ and $\mathbf{p}_u$, respectively. The aim of β_{si} and β_{pi} is to balance the information from auxiliary and target domain regarding the target item i. In other words, these two weights are item-sensitive. We devise such design to address the key challenge of *unclear predictive signal* which has been mentioned before. That is, in real scenario evaluating different items needs varying amount of auxiliary information. Although we can directly learn β_{si} and β_{pi} by optimizing a final objective function, we rely on the promising representation ability of attention network to model them. Formally, a domain-level attention module is designed as,

$$\begin{aligned} \beta_{si} &= \frac{e^{b_{si}}}{e^{b_{si}} + e^{b_{pi}}}, \ \beta_{pi} = \frac{e^{b_{pi}}}{e^{b_{si}} + e^{b_{pi}}}, \\ b_{si} &= \mathbf{w}_2 ReLU(\mathbf{s}_u \odot \mathbf{q}_i) + b_2, \\ b_{pi} &= \mathbf{w}_2 ReLU(\mathbf{p}_u \odot \mathbf{q}_i) + b_2, \end{aligned} \tag{11}$$

where $\mathbf{w}_2 \in \mathbb{R}^D$ and b_2 are the parameters of the attention network. Note that the input of the attention network is the interaction between user ($\mathbf{s}_u / \mathbf{p}_u$) and item embeddings, which enables the learned attention scores to be sensitive to item i.

3.4 Prediction Layer

After the operation in aforementioned layer, our problem further turns to predict user interaction with two embeddings: unified user embedding vector $\mathbf{z}_u$ and item embedding vector $\mathbf{q}_i$. Here we adopt a predictive function to estimate y_{ui}^t which is the interaction probability between a given pair of user and item (u, i). Since our work mainly focuses on a general framework for cross-domain

recommendation without sharing user-relevant data, we adopt a simple but widely-used inner product model, to estimate the value of y_{ui}^t, which is formulated as

$$\hat{y}_{ui}^t = \sigma(\mathbf{z}_u{}^T \mathbf{q}_i), \tag{12}$$

where σ is the *sigmoid* function. Note that this predictive function can be easily extended to more complicated ones, such as the multi-layer perceptron in [14].

To conclude, with three specially devised layers and a prediction layer, aforementioned three key challenges of our problem are addressed one by one.

3.5 Training

Objective Function. Following the probabilistic optimization framework [14, 28], we first define the likelihood function for an implicit interaction as follows,

$$P_r = \prod_{(u,i)\in\mathcal{Y}_+^t} \hat{y}_{ui}^t \prod_{(u,i)\in\mathcal{Y}_-^t} (1-\hat{y}_{ui}^t), \tag{13}$$

where $\mathcal{Y}_t^+$ denotes the set of observed interactions in interaction matrix of the target domain $\mathbf{Y}^t$ (entries with value of 1), and $\mathcal{Y}_-^t$ denotes negative instances sampled from the unobserved interactions in $\mathbf{Y}^t$ (entries with value of 0). We further take the negative logarithm of the joint probability, and obtain the loss function (*a.k.a, logloss* [29]), which is widely used to optimize recommendation systems with implicit feedbacks [14, 17, 24], to be minimized as follows,

$$\mathcal{L} = -(\sum_{(u,i)\in\mathcal{Y}_+^t} \log \hat{y}_{ui}^t + \sum_{(u,i)\in\mathcal{Y}_-^t} \log(1-\hat{y}_{ui}^t)). \tag{14}$$

To prevent over-fitting, we adopt l_2 regularization on the parameters in the proposed neural model and obtain the overall objective function,

$$\Gamma = \mathcal{L} + \lambda \sum_{\mathbf{\Theta}\in\{\mathbf{P},\mathbf{Q},\mathbf{W}_0,\mathbf{b}_0,\mathbf{w}_1,b_1,\mathbf{w}_2,b_2\}} \|\mathbf{\Theta}\|_F^2. \tag{15}$$

Mini-batch Training. We adopt stochastic gradient descent (SGD) [34], a widely generic solver for neural models, to optimize our proposed NATR model in the mini-batch mode. To construct a mini-batch, we first sample a batch of historical user-item interaction pairs (u, i). For each (u, i), we then adopt a negative sampling technique [24], which is widely used to handle implicit feedbacks in existing researches [14, 28, 43], to randomly select unobserved items $\{i_1^{'}, \cdots i_n^{'}\}$ for user u with a sampling ratio of n. After the sampling, we obtain n triplets $\{(u, i, i_1^{'}), \cdots, (u, i, i_n^{'})\}$ for each instance in the batch. With the constructed mini-batch, we take a gradient step to minimize the objective function.

4 EXPERIMENTS

In this section, we conduct extensive experiments on two real-world datasets to answer the following research questions:

- **RQ1:** How does our proposed NATR model perform compared with the state-of-the-art methods for cross-domain recommendation tasks?
- **RQ2:** Can the proposed NATR alleviate the data sparsity problem in the target domain?
- **RQ3:** What are the effects of the item-level and domain-level attention models in our proposed NATR?

In what follows, we first describe the experimental settings, and then answer the above three research questions.

Table 1: Statistics of our evaluation datasets.

Dataset	Item#	Auxiliary Domain		Target Domain	
		User#	Rec#	User#	Rec#
ML-NF	5,568	31,038	2,269,179	14,630	152,206
TC-IQI	4,851	35,398	314,621	19,999	78,429

4.1 Experimental Settings

4.1.1 Datasets. We experiment with two real-world datasets that both contain implicit interactions from two domains.

- **ML-NF Dataset.** MovieLens and Netflix are two popular platforms with movie recommendation services, in which there are a large portion of overlapped movies. Here we take MovieLens (ML) and Netflix (NF) as the *auxiliary* and *target* domains (*i.e.,* our target is to improve the recommendation performance in NF), respectively. We obtain user-movie interactions in ML and NF from two widely used public movie rating datasets[5][6]. Note that we identify movies with the same name in the two datasets as *bridge items*. Here we conduct whole-string matches to avoid wrong matches as possible. By filtering bridge items and their associated ratings[7], we reserve 5,568 movies, 14,630 ML users and 31,038 NF users in this dataset. Finally, we intentionally transform the rating data into binary (1/0 indicate whether a user has interacted with an item or not) to fit the problem setting of implicit feedback [16].
- **TC-IQI Dataset.** This dataset is collected by [43] to evaluate *cross-domain recommendation* performance of online video contents. In this dataset, there are historical interactions between users and videos from two mainstream video websites, iQiyi[8] (IQI) and Tencent Video[9] (TC) in China. To investigate the performance of facilitating recommendation performance in target domain via leveraging information from auxiliary domain, we regard *IQI* (*TC*) as the *target* (*auxiliary*) domain since interactions in IQI are sparser. Similarly, we filter *bridge items* via exact name matching across videos from these two domains, and only reserve bridge items and interactions associated with them. Note that videos in this dataset are professional production content (PGC) widely available on multiple websites.

After the above pre-processing steps, we obtain two final datasets for performance evaluation, the statistics of which are summarized in Table 1.

[5] https://grouplens.org/datasets/movielens/
[6] https://www.kaggle.com/laowingkin/netflix-movie-recommendation/data
[7] To better evaluate the performance, we follow the setting of previous works [27, 32] and only reserve the bridge items (*i.e.*, neglect items occurring only in one domain). However, our model is also suitable to perform recommendation for those items not overlapped, which has been introduced in detail in Section 3.1
[8] https://www.iqiyi.com
[9] https://v.qq.com

4.1.2 Evaluation Protocols. Following existing works [12, 31], we employ the widely used *leave-one-out* evaluation protocol in the evaluation stage. Similar with [14, 18], given a user in the target domain, we randomly sample 99 items that are not interacted with the user, each method ranks one test item among the 99 sampled items. We then adopt two metrics, *HR* and *NDCG*, which are widely used in the literature of recommendation [6, 14, 39], to report the ranking performance of each method:

- **HR@K:** *Hit Ratio* (HR) measures whether the test item is contained by the top-K item ranking list (1 for yes and 0 for no).
- **NDCG@K:** *Normalized Discounted Cumulative Gain* (NDCG) extends HR by assigning higher scores to the hits at higher positions in the ranking list.

It should be noted that we calculate HR@K and NDCG@K for each test user, and report the average ones over the whole user set.

4.2 Performance Comparison (RQ1)

4.2.1 Baselines. We compare the performance of our proposed NATR with five baselines, which can be divided into two groups: *single-domain* and *cross-domain*. Here *single-domain* methods refer to those which are merely trained with data from the target domain, while *cross-domain* methods jointly consider the data from both the target and auxiliary domains.

The compared *single-domain* methods are introduced as follows:

- **PMF [24].** *Probabilistic Matrix Factorization* (PMF) is a MF based model which exploits negative sampling to handle implicit interaction data. It adopts *logloss* as the loss function and samples several negative items with a ratio when a positive item is fed for training. We tune the learning rate and regularizer and report the best testing performance.
- **GMF.** *Generalized Matrix Factorization* (GMF) is one of the variants of NCF (Neural Collaborative Filtering) [14], which is the state-of-the-art solution for recommendation tasks with implicit feedbacks. This method assign various weights for different dimensions in the dot-product prediction function, which can be regarded as a generalization of vanilla MF. We optimize this model and tune its associated hyper-parameters similarly with the paper.
- **NATR-local.** As mentioned in Section 3.1, our NATR model utilize an item-based CF to leverage transferred item embeddings. Therefore, it is still questionable whether the item-based CF is the only component to improve performance while transferred item embeddings do not help? To demonstrate the effectiveness of transferred embeddings, we degenerate the NATR via adopting local item embeddings rather than transferred item embeddings in the item-level attention unit. Therefore, it is a kind of *single-domain* method. We name it NATR-local and tune it similarly with NATR to report the best performance.

The compared *cross-domain* baselines are as follows.

- **CMF [32].** *Collective Matrix Factorization* (CMF) decomposes the data matrices of multiple interactions simultaneously while sharing embedding vectors of users or items. Here we factorize two interaction matrices from two domains, sharing embedding vectors of those *bridge items*. We carefully tune the weight of two domains, learning rate and regularizer to report the best performance. It is worth mentioning that a recent study on cross-domain recommendation [43] proposed a method named MPF, which adapted vanilla CMF to a special case where all users and items are all overlapped across domains. Apparently this special setting does not fit our problem where only item can be overlapped and regretfully MPF cannot be adapted to our task.
- **ItemCST [26].** *Coordinate System Transfer* (CST) also assumes that both users and items are overlapped and adds two regularization terms in objective functions. Specifically, the two terms set constraints to the embedding distance in two domains for those overlapped users or items. Thus, CST can be adapted to our problem by only reserving item-side regularization term in our task, and we name it as ItemCST. We tune the learning rate and coefficient of regularization term to report the best performance.

  To conclude, CMF is the state-of-art cross-domain recommendation method while facing with high risk of leaking user privacy since it assumes that all interaction data are fully shared. ItemCST is an adapted method from CST and as it only needs the transferred item embeddings to compute the regularization term of item, it preserves user-relevant data.

We implement the baseline methods and our NATR model in TensorFlow[10] It should be noted that we set the embedding size of all compared methods to be 64, which is a typical setting in literature [14, 17]. Our primary experiments also demonstrate that 64 is an embedding size with enough ability to represent the user and item.

4.2.2 Parameter Settings. To determine the optimal hyper-parameters of the method, we construct a validation set via randomly selecting an interacted item for each test user, which has not been selected as the test item. For our NATR, we initialize parameters with a widely used initialization method proposed in [11]. During the training phase, we intentionally set the negative sampling ratio as 4 to construct mini-batches with size of 256 as described in Section 3.5. To optimize the NATR model, we employ the Adagrad [7] optimizer and search its learning rate within {0.001, 0.002, 0.005, 0.01}. In addition, we tune the λ in Equation 15, which balances the loss and regularization terms, in {1e-2, 1e-3, 1e-4,1e-5,1e-6}. As mentioned before, ItemCST and our NATR only rely on the transferred item embeddings as auxiliary data, and in this paper, without loss of generality, we adopt PMF in the auxiliary domain and carefully tune its learning rate and regularizer to obtain item embeddings.

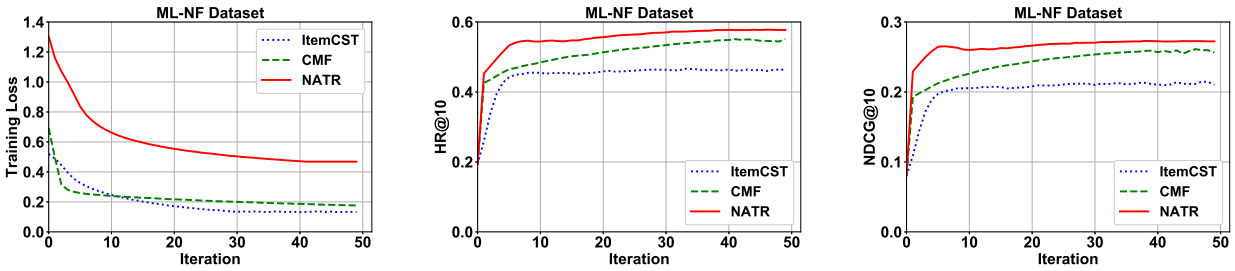
We first compare the top-K recommendation performance with baseline methods. We investigate the top-K performance with K setting to {1, 2, 5, 10}[11]. As described in the evaluation protocols, we test the performance of a ranking list with 100 items. As such, it is reasonable to choose relatively small K [14]. For every method,

[10] https://www.tensorflow.org

[11] Note that HR@K equals to NDCG@K when setting $K = 1$.

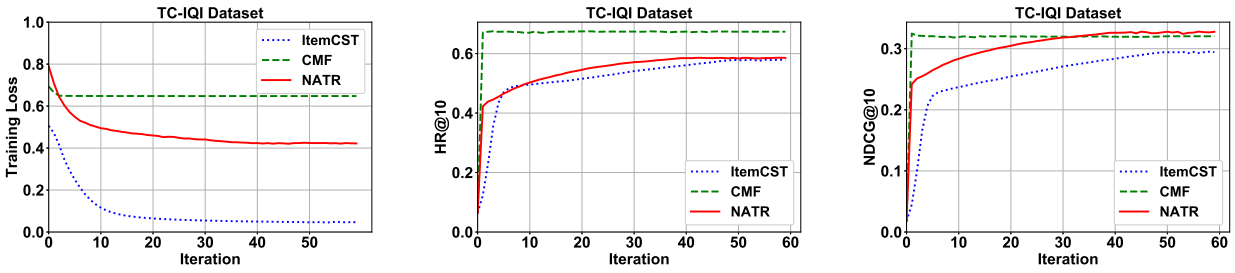
Table 2: Top-K recommendation performance comparison on the ML-NF and TC-IQI datasets (K is set to 1, 2, 5, 10)

			ML-NF Dataset						
Group	**Method**	**User-relevant Data**	**HR(NDCG)@1**	**HR@2**	**NDCG@2**	**HR@5**	**NDCG@5**	**HR@10**	**NDCG@10**
Cross Domain	**NATR**	**Preserved**	**0.1315**	**0.1976**	**0.1403**	**0.3776**	**0.2110**	**0.5781**	**0.2726**
	ItemCST	Preserved	0.0795	0.1475	0.1005	0.3068	0.1670	0.4846	0.2228
	CMF	Shared	0.1023	0.1903	0.1283	0.3675	0.2025	0.5483	0.2593
Single Domain	NATR-local	Preserved	0.0947	0.1769	0.1253	0.3402	0.1894	0.5183	0.2440
	PMF	Preserved	0.0668	0.1162	0.0796	0.2721	0.1375	0.4494	0.1956
	GMF	Preserved	0.0706	0.1174	0.0816	0.2681	0.1410	0.4284	0.1918
			TC-IQI Dataset						
Group	**Method**	**User-relevant Data**	**HR(NDCG)@1**	**HR@2**	**NDCG@2**	**HR@5**	**NDCG@5**	**HR@10**	**NDCG@10**
Cross Domain	**NATR**	**Preserved**	**0.2010**	0.2660	**0.2104**	**0.4513**	**0.2881**	0.6035	**0.3365**
	ItemCST	Preserved	0.1161	0.2129	0.1445	0.4194	0.2309	0.6079	0.2904
	CMF	Shared	0.1649	**0.3101**	0.2101	0.4499	0.2668	**0.6595**	0.3326
Single Domain	NATR-local	Preserved	0.1677	0.2552	0.1776	0.4214	0.2412	0.5864	0.2948
	PMF	Preserved	0.0848	0.1238	0.0945	0.2291	0.1326	0.3309	0.1694
	GMF	Preserved	0.1584	0.2445	0.1729	0.4101	0.2425	0.6021	0.3029



(a) Training Loss (b) HR@10 (c) NDCG@10

Figure 3: Training loss and testing performance of NATR, ItemCST and CMF in each iteration on ML-NF Dataset.

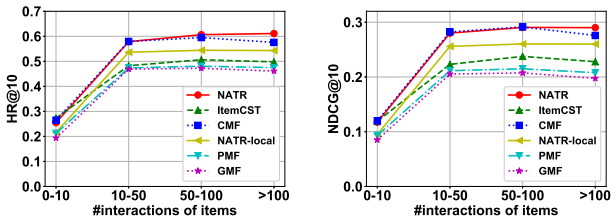


(a) Training Loss (b) HR@10 (c) NDCG@10

Figure 4: Training loss and testing performance of NATR, ItemCST and CMF in each iteration on TC-IQI Dataset.

we carefully tune the hyper-parameters to report the best performance. In Table 2, we report the top-K recommendation performance for the two utilized real-world datasets. We compare our proposed NATR method with three single-domain baselines and two cross-domain ones. From these results, we have the following observations:

- **NATR significantly improves recommendation performance in the target domain.** 1) For those *single domain* methods that are trained with only interaction data from the target domain, the recommendation performance is relatively poor. PMF achieves the worst performance, which can be explained as the limited representation ability of MF model. 2) NATR-local, a degenerative model of the proposed NATR, outperforms PMF and GMF *w.r.t.* all metrics on the ML-NF dataset and most metrics on the TC-IQI dataset, which justifies the effectiveness of explicitly encoding users' historical interactions. 3) Compared with PMF, GMF, and NATR-local, the proposed NATR outperforms the best of them by 28.26% and 39.11% in HR@10 and NDCG@10 for ML-NF dataset and by 0.23% and 11.09% in HR@10 and NDCG@10 for TC-IQI dataset. It demonstrates that leveraging the item embeddings from the auxiliary domain enhances the recommendation quality in the target domain, which



(a) HR@10 (b) NDCG@10

Figure 5: Performance of all methods on items with different number of interaction records on ML-NF dataset.

further indicates that the proposed NATR is a promising solution for the cross-domain recommendation task.

- **NATR performs even better than those cross-domain methods with the risk of leaking user privacy.** We can observe that our proposed NATR model achieves the best performance compared to CMF and ItemCST regarding every evaluation metrics in the NF-ML dataset. For HR@10 and NDCG@10, NATR outperforms the best of them by 18.94% and 4.94%, respectively. We guess the reason of such results is that a joint training with data from two domains (CMF and ItemCST) might distract the loss during the optimization and converge at a status which balances the two domains rather than the optimal status of the target domain. We leave further investigations at the future work. To further study how these methods perform, we present the training loss and testing performance in each interaction in Figure 3 (for ML-NF dataset) and Figure 4 (for TC-IQI dataset). For every method in the two figures, we report the best parameter settings. For both datasets, all methods achieve stable performance after about 50 iterations. With fine hyper-parameter tuning to solve over-fitting, our proposed NATR can effectively outperform ItemCST and achieve similar or even better performance than CMF.
- **NATR effectively distills the CF signal encoded in transferred item embeddings.** NATR-local, a degenerative model of our proposed NATR, only utilizes interaction data from the target domain without exploiting transferred item embddings. Specifically, it replace the transferred item embedding in NATR to local item embeddings. On one hand, the experimental results in Table 2 show that NATR-local achieves better performance than GMF, a competitive method for single domain, demonstrating that taking the explicit preferences of users into consideration can improve recommendation performance. On the other hand, NATR outperforms NATR-local on two datasets which means the combining transferred embeddings are better than only a local CF solution on the target domain. This confirms the utility of transferred item embeddings in encoding CF signal from the auxiliary domain.

To summarize, these comparisons on two real-world datasets verify that our proposed NATR model can effectively leverage transferred item embeddings to improve the recommendation performance in the target domain.

4.3 Data Sparsity Problem (RQ2)

As mentioned in the introduction, one of the primary purposes for cross-domain recommendation is to alleviate item data sparsity problem (*i.e.*, items' records are too few) in the target domain. In particular, for those items with few interactions, of which the embeddings cannot be learned well in the target domain itself, transferred embeddings from the auxiliary domain play a bigger role. To study extensively how our proposed NATR model effectively helps to alleviate the item data sparsity issue, we compare the recommendation performance for items with different level of sparsity.

Specifically, we divide the items to several groups according to number of interaction records in the training set. Note that each group have similar number of items, which make the experimental results more reasonable. Then we apply the evaluation protocol, *leave-one-out*, which is the same with above experiments. For each item, its performance is defined as the average of HR@10 and NDCG@10 when it is in the test set. We compare the proposed NATR model with all five baseline methods in Figure 5. From the results, we can observe that when the interaction records of an item become sparser, the recommendation performance will go worse. For example, in the first group, of which each item has been interacted by only 1-10 users, the best performance of those single-domain methods is about only 0.220 for HR@10 and 0.098 for NDCG@10. Fortunately, with the help of the auxiliary domain, cross-domain methods can achieve better performance for those sparse items. Out of these methods, our proposed NATR model can achieve similar performance compared with CMF and better than ItemCST, which verifies that NATR can serve as a competitive cross-domain recommendation method without sharing user-relevant data.

In summary, our NATR model can improve recommendation performance effectively, no matter the historical records of items are sparse or dense. For items with sparser records, the improvement is more evident and meaningful.

4.4 Impact of Attention Mechanism (RQ3)

In NATR, we utilize the attention mechanism to solve two primary challenges. First, a domain-level attention unit is applied to distill useful signals from transferred item embeddings adn integrate them into the target domain. Second, an item-level attention unit is adopted to handle varying importance of transferred item embeddings. An intuitive question is whether the designed attention unit can really help in our model?

To answer it, we conduct experiments on two degenerative methods of NATR, in which two utilized attention network components are replaced by the simple operation of pooling (*i.e.*, average summation), respectively. We adopt the same evaluation methods with above experiments, and the performance comparison is shown in Table 3. We can observe that when removing item-level attention unit, the top-10 recommendation performance become poor, which

Table 3: Impact of Attention Mechanism.

Dataset	ML-NF	
Methods	**HR@10**	**NDCG@10**
NATR	0.5781	0.2726
Without Item-level Attention	0.5624	0.2655
Without Domain-level Attention	0.5669	0.2722

demonstrate the challenge of varying importance and our attention solution can address it. Besides, removing domain-level attention unit also make the recommendation performance worse. That is, the auxiliary domain cannot be considered as the same with the target domain simply. To conclude, the experimental results demonstrate the necessity of our two specially designed attention units.

In conclusion, extensive experiments on two real-world datasets verify the efficacy of our proposed model, and further studies demonstrate that our model can alleviate data sparsity problem. Moreover, the utility of our specially designed attention networks components is verified.

5 RELATED WORK

In this paper, we propose a solution for cross-domain recommendation without sharing user-relevant data based on neural networks. The closed related work can be divided into cross-domain recommendation and neural network based recommendation.

Cross-domain Recommendation To alleviate cold start and data sparsity issue, cross-domain recommendation is a typical solution which takes data from multiple domains into consideration. With the help of the auxiliary domain, cross-domain recommendation methods can achieve better performance (*i.e.* recommendation accuracy) than single-domain ones. Approaches of cross-domain recommendation can be broadly classified into two types: collaborative and content-based.

Collaborative cross-domain recommendation refers to those approaches utilizing interaction data (rating history, for example) from two domains. Ajit *et al.* [32] proposed a MF based model, CMF (Collective Matrix Factorization), which assumes a common global user factor matrix for all domains, and it factorizes matrices from multiple domains simultaneously. Li *et al.* [22] proposed a model named CBT (Code Book Transfer) which builds a matrix named codebook to represent cluster-level rating pattern and this codebook is shared by two domains. A recent study [43] considered a special task in which both users and items are overlapped, and they proposed a MF based model which assumes part of the user embeddings and whole item embeddings are shared across domains. With a similar setting, Man *et al.* [23] proposed a neural method which employs multi-layer perceptron to adapt user and item embeddings between two domains. Pan *et al.* [26] utilize auxiliary interaction data with a regularization term concerned with overlapped user and item in objective function in MF model. Another category of cross-domain recommendation models is content-based ones, which sharing attributes of user or items from auxiliary domain [1, 8, 45]. Agarwal *et al.* [1] proposed a MF based model in cross-domain recommendation when multi-modal user profiles are available. Elkahky *et al.* [8] transformed user profile and item attributes to dense vectors through deep neural network and matched them in latent space. Zhang *et al.* [45] utilize textual, structure and visual knowledge of items as auxiliary domain to aid building item embedding.

In this paper, we focus on collaborative cross-domain recommendation with auxiliary interaction data, a widely used setting in literature. Specifically, our problem is a typical system-level cross-domain recommendation task, where same items are shared across domains, according to the definition in two surveys [2, 21]. There is a common assumption in existing methods that the whole interaction data can be fully shared across domains, which has the risk of leaking user privacy since various domains may be operated by different companies. In this work, we advocate a more realistic settings that only item-side data can be shared.

Neural Network Based Recommendation. Salakhutdinov *et al.* [30] proposed RBM (Restricted Boltzmann Machines) to predict explicit ratings, which was the first work to apply neural networks to recommender systems. Recently, similar as the research field of CV and NLP, neural networks have achieved great success in recommender systems. Some works [14, 36, 42, 44, 46] relied on neural networks to learn to match function between users and items directly, which can be regarded as the extension of traditional collaborative filtering approaches. He *et al.* [14] proposed a general neural architecture for collaborative filtering, which learns the user-item interaction function via generalized matrix factorization and multi-layer perceptrons. Zhang *et al.* [46] mapped user and item to Hamming space and obtained matching score via neural networks. Tay *et al.* [36] proposed a relational-translation based match function to learn from interactions. Yang *et al.* [42] utilized neural networks to match user-item interaction and user-user relation simultaneously to perform a social recommendation task. Ying *et al.* [44] relied on graph conventional network for collaborative filtering to match huge mount of users and items in real recommender systems. Gao *et al.* [9] introduce multi-task learning to nueral networks to solve the task of recommendation with users' multiple types of behaviors. Besides, some works utilized neural networks to extract the auxiliary information and features in recommender systems, such as textual [25, 47], visual [3, 38], video [5], and hybrid [48].

In this paper, we propose a novel neural model to solve the challenges of extracting useful knowledge from item embedding of the auxiliary domain with the power of neural networks in representative learning.

6 CONCLUSION

In this work, we present a new cross-domain recommendation solution, which can avoid user privacy leakage by transferring only item embeddings from the auxiliary domain. To better exploit the transferred item embeddings, we propose a neural network method named NATR, combining item-level and domain-level attention mechanisms to address the challenges in cross-domain learning. We conduct extensive experiments on two real-world datasets, demonstrating that our NATR method can improve the recommendation performance of the target domain by 18.94%. To the best of our

knowledge, this is the first work that concerns user privacy in cross-domain recommendation, and presents a sound solution to exploit the predictive signal without sharing any user-relevant information.

There are three points about this work that we plan to address in future. First, although our method has taken the dimensionality discrepancy into consideration via a dimension-adaption fully-connected layer, we only empirically verify its effectiveness when two domains have the same embedding size. As such, we will study how different sizes of transferred embeddings would affect the recommendation performance. Second, we will study scalability of our method in industrial scenario where amount of users and items are very huge. Last, since this work only focuses on collaborative cross-domain recommendation, we will study the similar task in content-based scenario.

ACKNOWLEDGMENTS

This work was supported in part by The National Key Research and Development Program of China under grant 2017YFE0112300, the National Nature Science Foundation of China under 61861136003, 61621091 and 61673237, Beijing National Research Center for Information Science and Technology under 20031887521, and research fund of Tsinghua University - Tencent Joint Laboratory for Internet Innovation Technology. This research is also part of NExT++ research which is supported by the National Research Foundation, Prime Minister's Office, Singapore under its IRC@SG Funding Initiative.

REFERENCES

[1] Deepak Agarwal, Bee-Chung Chen, and Bo Long. 2011. Localized factor models for multi-context recommendation. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 609–617.

[2] Iván Cantador, Ignacio Fernández-Tobías, Shlomo Berkovsky, and Paolo Cremonesi. 2015. Cross-domain recommender systems. In *Recommender systems handbook*. Springer, 919–959.

[3] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 335–344.

[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Deep Learning for Recommender Systems Workshop (DLRS)*. 7–10.

[5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *ACM Conference on Recommender Systems (RecSys)*. 191–198.

[6] Jingtao Ding, Guanghui Yu, Xiangnan He, Yuhan Quan, Yong Li, Tat-Seng Chua, Depeng Jin, and Jiajie Yu. 2018. Improving Implicit Recommender Systems with View Data.. In *International Joint Conferences on Artificial Intelligence (IJCAI)*. 3343–3349.

[7] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)* 12, Jul (2011), 2121–2159.

[8] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *International World Wide Web Conference (WWW)*. 278–288.

[9] Chen Gao, Xiangnan He, Dahua Gan, Xiangning Chen, Fuli Feng, Yong Li, Tat-Seng Chua, and Depeng Jin. 2019. Neural Multi-Task Recommendation from Multi-Behavior Data. In *International Conference on Data Engineering (ICDE)*.

[10] Chen Gao, Chao Huang, Yue Yu, Huandong Wang, Yong Li, and Depeng Jin. 2019. Privacy-preserving Cross-domain Location Recommendation. In *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *International Conference on Computer Vision (ICCV)*. 1026–1034.

[12] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial personalized ranking for recommendation. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 355–364.

[13] Xiangnan He, Zhenkui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. NAIS: neural attentive item similarity model for recommendation. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* (2018).

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *International World Wide Web Conference (WWW)*. 173–182.

[15] Liang Hu, Jian Cao, Guandong Xu, Longbing Cao, Zhiping Gu, and Can Zhu. 2013. Personalized recommendation via cross-domain triadic factorization. In *International World Wide Web Conference (WWW)*. 595–606.

[16] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *International Conference on Data Mining (ICDM)*. 263–272.

[17] Santosh Kabbur, Xia Ning, and George Karypis. 2013. Fism: factored item similarity models for top-n recommender systems. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 659–667.

[18] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 426–434.

[19] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).

[20] Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, Vol. 1. 1437–1447.

[21] Bin Li. 2011. Cross-domain collaborative filtering: A brief survey. In *International Conference on Tools with Artificial Intelligence*. 1085–1086.

[22] Bin Li, Qiang Yang, and Xiangyang Xue. 2009. Can Movies and Books Collaborate? Cross-Domain Collaborative Filtering for Sparsity Reduction.. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, Vol. 9. 2052–2057.

[23] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-domain recommendation: an embedding and mapping approach. In *International Joint Conferences on Artificial Intelligence (IJCAI)*. 2464–2470.

[24] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Annual Conference on Neural Information Processing Systems (NIPS)*. 1257–1264.

[25] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based news recommendation for millions of users. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1933–1942.

[26] Weike Pan, Evan Wei Xiang, Nathan Nan Liu, and Qiang Yang. 2010. Transfer learning in collaborative filtering for sparsity reduction. In *The AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 10. 230–235.

[27] Chanyoung Park, Donghyun Kim, Jinoh Oh, and Hwanjo Yu. 2017. Do also-viewed products help user rating prediction?. In *International World Wide Web Conference (WWW)*. 1113–1122.

[28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *The Conference on Uncertainty in Artificial Intelligence (UAI)*. 452–461.

[29] Christian Robert. 2014. Machine Learning, a probabilistic perspective. *CHANCE* 27, 2 (2014), 62–63.

[30] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted Boltzmann machines for collaborative filtering. In *International Conference on Machine Learning (ICML)*. 791–798.

[31] Andriy Shepitsen, Jonathan Gemmell, Bamshad Mobasher, and Robin Burke. 2008. Personalized recommendation in social tagging systems using hierarchical clustering. In *ACM Conference on Recommender Systems (RecSys)*. 259–266.

[32] Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 650–658.

[33] Peijie Sun, Le Wu, and Meng Wang. 2018. Attentive recurrent social recommendation. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 185–194.

[34] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. 2013. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning (ICML)*. 1139–1147.

[35] Jie Tang, Sen Wu, Jimeng Sun, and Hang Su. 2012. Cross-domain collaboration recommendation. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1285–1293.

[36] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent relational metric learning via memory-based attention for collaborative ranking. In *International World Wide Web Conference (WWW)*.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*. 5998–6008.

[38] Suhang Wang, Yilin Wang, Jiliang Tang, Kai Shu, Suhas Ranganath, and Huan Liu. 2017. What your images reveal: Exploiting visual contents for point-of-interest recommendation. In *International World Wide Web Conference (WWW)*. 391–400.

[39] Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. 2018. Tem: Tree-enhanced embedding model for explainable recommendation. In *International World Wide Web Conference (WWW)*. 1543–1552.

[40] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: learning the weight of feature interactions via attention networks. In *International Joint Conferences on Artificial Intelligence (IJCAI)*. 3119–3125.

[41] Jun Xu, Xiangnan He, and Hang Li. 2018. Deep learning for matching in search and recommendation. *International World Wide Web Conference (WWW)* (2018).

[42] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. 2017. Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

[43] Chunfeng Yang, Huan Yan, Donghan Yu, Yong Li, and Dah Ming Chiu. 2017. Multi-site user behavior modeling and its application in video recommendation. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 175–184.

[44] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

[45] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

[46] Yan Zhang, Hongzhi Yin, Zi Huang, Xingzhong Du, Guowu Yang, and Defu Lian. 2018. Discrete deep learning for fast content-aware recommendation. In *International Conference on Web Search and Data Mining (WSDM)*. ACM, 717–726.

[47] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *International Conference on Web Search and Data Mining (WSDM)*. 425–434.

[48] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1059–1068.

Towards Self-Adaptive Metric Learning On the Fly

Yang Gao
The University of Texas at Dallas
Richardson, Texas
yxg122530@utdallas.edu

Yi-Fan Li
The University of Texas at Dallas
Richardson, Texas
yli@utdallas.edu

Swarup Chandra
The University of Texas at Dallas
Richardson, Texas
swarup.chandra@utdallas.edu

Latifur Khan
The University of Texas at Dallas
Richardson, Texas
lkhan@utdallas.edu

Bhavani Thuraisingham
The University of Texas at Dallas
Richardson, Texas
bhavani.thuraisingham@utdallas.edu

ABSTRACT

Good quality similarity metrics can significantly facilitate the performance of many large-scale, real-world applications. Existing studies have proposed various solutions to learn a Mahalanobis or bilinear metric in an online fashion by either restricting distances between similar (dissimilar) pairs to be smaller (larger) than a given lower (upper) bound or requiring similar instances to be separated from dissimilar instances with a given margin. However, these linear metrics learned by leveraging fixed bounds or margins may not perform well in real-world applications, especially when data distributions are complex. We aim to address the open challenge of "Online Adaptive Metric Learning" (OAML) for learning adaptive metric functions on-the-fly. Unlike traditional online metric learning methods, OAML is significantly more challenging since the learned metric could be non-linear and the model has to be self-adaptive as more instances are observed. In this paper, we present a new online metric learning framework that attempts to tackle the challenge by learning a ANN-based metric with adaptive model complexity from a stream of constraints. In particular, we propose a novel Adaptive-Bound Triplet Loss (ABTL) to effectively utilize the input constraints, and present a novel Adaptive Hedge Update (AHU) method for online updating the model parameters. We empirically validates the effectiveness and efficacy of our framework on various applications such as real-world image classification, facial verification, and image retrieval.

CCS CONCEPTS

• **Information systems** → **Similarity measures**; • **Theory of computation** → **Online learning algorithms**; • **Computing methodologies** → *Online learning settings.*

KEYWORDS

Adaptive-Bound Triplet Loss; Adaptive Metric Complexity; Online Metric Learning

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313503

ACM Reference Format:
Yang Gao, Yi-Fan Li, Swarup Chandra, Latifur Khan, and Bhavani Thuraisingham. 2019. Towards Self-Adaptive Metric Learning On the Fly. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313503

1 INTRODUCTION

Learning a good quality similarity metric for images is critical to many large-scale web applications such as image classification, face verification, and image retrieval. These applications are widely used for user identity verification in mobile apps and image recommendation in multi-media search engines. Therefore, having a scalable and high-performance technique for learning a good image similarity metric is becoming increasingly important. Various online metric learning (OML) solutions applying neural networks [4, 6, 12, 14, 19] have been proposed to learn a similarity metric from a stream of constraints. These constraints are typically collected and constructed from user clicks, indicating the similarity and dissimilarity relations among images. Two kinds of constraints, i.e., pairwise and triplet constraints, are widely-accepted in previous studies. A pairwise constraint consists of two similar or dissimilar images, while a triplet constraint is of the form (A, B, C), where image A is similar to image B, but is dissimilar to image C. Compared to offline metric learning approaches, OML algorithms do not require the entire training data to be made available prior to the learning task and hence are more suitable for web applications.

Despite the success of existing OML approaches in many applications, some intrinsic drawbacks have been recently observed. First, most of the existing OML algorithms [6, 12, 14, 19] are designed to learn a pre-selected linear metric (e.g., Mahalanobis distance) by solving a convex optimization problem. These simple metrics are unable to capture the non-linear semantic similarities among instances in complex application scenarios such as image retrieval with fine-grained sub-classes. Second, the performance of existing OML solutions depend heavily on the quality of input constraints, i.e., the true similarity or dissimilarity among example images (training data) provided by the user. These algorithms learn the metric model only from the "hard" constraints within the input and are biased towards them, thereby weakening the model robustness. Here, a constraint is "hard" if the dissimilar images in it is computed as similar by current metric and vice-versa. Moreover, filtering out "hard" constraints from a large volume of candidates is computationally expensive, and is impossible to perform in a real-time setting. In

addition, a real-world input stream usually contains limited number of "hard" constraints because we have little control on the source, i.e., the users. As a result, existing OML approaches may suffer from low constraint utilization in practical applications, leading to poor performance.

These intrinsic drawbacks result in the open challenge of "Online Adaptive Metric Learning" (OAML), i.e., how to learn a metric function with adaptive complexity as more constraints are observed in the stream, while maintaining high constraint utilization during the continuous learning process. This metric function should have a dynamic hypothesis space, ranging from simple linear functions to highly non-linear functions.

In this paper, we propose a novel framework that is able to learn a ANN-based metric from a stream of constraints with full constraint utilization, and more importantly, is able to dynamically adapt its model complexity when necessary. To achieve this, we first amend the common ANN architecture so that the output of every hidden layer is given as input to an independent metric embedding layer (MEL). Each MEL represents an embedding space where similar instances are closer to each other and dissimilar instances are further away from each other. A metric weight is then assigned to each MEL, measuring its importance in the entire metric model. We propose a novel Adaptive-Bound Triplet Loss (ABTL) to evaluate the similarity and dissimilarity errors introduced by the arriving constraint at each online round, and present a novel Adaptive Hedge Update (AHU) method to update the parameters of the metric model along with the metric weights based on these errors. Note that the proposed ABTL eliminates the unique dependency on "hard" constraints in existing methods, thereby improving both the constraint utilization and the quality of learned metrics. We refer to the proposed approach as Online metric learning with Adaptive Hedge Update (OAHU).

The contribution of this paper are as follows:

- We present a novel online metric learning framework - called OAHU - that addresses the challenges in "Online Adaptive Metric Learning" by learning a ANN-based metric with adaptive model complexity and achieving full constraint utilization.
- We theoretically show the regret bound of OAHU and demonstrate the optimal range for parameter selection.
- We empirically evaluate OAHU on real-world image classification, face verification, and image retrieval tasks, and compare its results with existing state-of-the-art online metric learning algorithms.

Rest of this paper is organized as follows. We first review related works in Section 2. Then, we formalize the online metric learning problem and discuss our proposed solution in Section 3. Next, we discuss the empirical evaluation of OAHU in Section 4, and finally conclude in Section 5.

2 RELATED WORK

Existing online metric learning solutions can be broadly classified into two categories: bilinear similarity-based or Mahalanobis distance-based methods.

The bilinear similarity-based methods learn a distance metric given by $d(\boldsymbol{x_1}, \boldsymbol{x_2}) = \boldsymbol{x_1}^T A \boldsymbol{x_2}$. A popular bilinear-based method is OASIS [6]. It is an online dual approach with a passive-aggressive algorithm which computes a similarity metric to solve the image retrieval task. Sparse Online Metric Learning (SOML) [5] is another technique which learns a metric by simplifying the full matrix A by a diagnoal matrix w, i.e., $A = diag(w)$, to handle very high-dimensional cases. To reduce the memory and computational cost, the Sparse Online Relative Similarity (SORS) [1] method adopts an off-diagonal l_1 norm of A as the sparse regularization to pursue a sparse model during the learning process. These three methods are based on triplet constraints where relative similarity/dissimilarity among images are specified.

On the other hand, Mahalanobis distance-based methods focus on learning a distance metric given by $d(\boldsymbol{x_1}, \boldsymbol{x_2}) = (\boldsymbol{x_1} - \boldsymbol{x_2})^T A (\boldsymbol{x_1} - \boldsymbol{x_2})$. Several algorithms have been proposed. Pseudo-metric Online Learning Algorithm (POLA) [22] introduces successive projection operations to learn the optimal metric. Whereas, LEGO [12] learns a Manalanobis metric based on LogDet regularization and gradient descent. RDML [14] uses a regularized online metric learning algorithm. Recently, Li et al. [19] introduced a closed-form solution called OPML, which adopts a one-pass triplet construction strategy to learn the optimal metric with a low computational cost. All these methods, except OPML, are based on pairwise constraints where the similarity or dissimilarity among pairs of images are specified.

In general, triplet constraints are more effective than pairwise constraints [6, 23, 25] because they allow simultaneous learning of similar and dissimilar relations. Therefore, similar to OASIS and OPML, we focus on triplets to learn a high-quality metric. However, in contrast to all the previous online metric learning algorithms, our approach does not depend on the Mahalanobis or bilinear distance since their linearity property [2] may limit expressiveness when exposed to complex data distributions in practical applications. Instead, we introduce a self-adaptive ANN-based metric that flexibly selects an appropriate metric function according to the scenario on which it is deployed.

3 OAHU

3.1 Problem Setting

Given a sequence of triplet constraints $S = \{(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-})\}_{t=1}^T$ that arrive sequentially, where $\{\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-}\} \in \mathcal{R}^d$, and $\boldsymbol{x_t}$ (anchor) is similar to $\boldsymbol{x_t^+}$ (positive) but is dissimilar to $\boldsymbol{x_t^-}$ (negative), the goal of online adaptive metric learning is to learn a model $\boldsymbol{F} : \mathcal{R}^d \mapsto \mathcal{R}^{d'}$ such that $||\boldsymbol{F}(\boldsymbol{x_t}) - \boldsymbol{F}(\boldsymbol{x_t^+})||_2 \ll ||\boldsymbol{F}(\boldsymbol{x_t}) - \boldsymbol{F}(\boldsymbol{x_t^-})||_2$.

The overall aim of the task is to learn a metric model with adaptive complexity while maintaining a high constraint utilization rate. Here, the complexity of $\boldsymbol{F}$ needs to be adaptive so that its hypothesis space is automatically enlarged or shrinked as necessary.

In practical applications, the input triplet stream can be generated as follows. First, some seed triplets can be constructed from user clicks. Then, more triplets are formed by taking transitive closures[1] over these seed triplets. The generated triplets are inserted into the stream in chronological order of creation. In this paper, we assume

[1] If $(\boldsymbol{x_1}, \boldsymbol{x_2})$ and $(\boldsymbol{x_1}, \boldsymbol{x_3})$ are two similar pairs, then $(\boldsymbol{x_2}, \boldsymbol{x_3})$ is a similar pair. If $(\boldsymbol{x_1}, \boldsymbol{x_2})$ and $(\boldsymbol{x_2}, \boldsymbol{x_3})$ are two similar pairs, then $(\boldsymbol{x_1}, \boldsymbol{x_3})$ is a similar pair. If $(\boldsymbol{x_1}, \boldsymbol{x_2})$ is a similar pair and $(\boldsymbol{x_1}, \boldsymbol{x_3})$ is a dissimilar pair, then $(\boldsymbol{x_2}, \boldsymbol{x_3})$ is a dissimilar pair. If $(\boldsymbol{x_1}, \boldsymbol{x_2})$ is a similar pair and $(\boldsymbol{x_2}, \boldsymbol{x_3})$ is a dissimilar pair, then $(\boldsymbol{x_1}, \boldsymbol{x_3})$ is a dissimilar pair.

$\boldsymbol{x}_t^*$: Any anchor, positive or negative instance in a triplet.	E_l: l^{th} metric model in OAHU.
$f^{(l)}(\boldsymbol{x}_t^*)$: Metric embedding of $\boldsymbol{x}_t^*$ generated by E_l.	$\mathcal{L}^{(l)}(\boldsymbol{x}_t, \boldsymbol{x}_t^+, \boldsymbol{x}_t^-)$: The local loss of $(\boldsymbol{x}_t, \boldsymbol{x}_t^+, \boldsymbol{x}_t^-)$ evaluated by E_l.
$\mathcal{L}_{overall}(\boldsymbol{x}_t, \boldsymbol{x}_t^+, \boldsymbol{x}_t^-)$: The total loss of $(\boldsymbol{x}_t, \boldsymbol{x}_t^+, \boldsymbol{x}_t^-)$ evaluated in OAHU.	$\alpha^{(l)}$: Weight of metric model E_l.
$D^{(l)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$: Distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ measured in E_l based on $f^{(l)}(\boldsymbol{x})$.	$d_{sim}^{(l)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ / $d_{dis}^{(l)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$: Similarity/Dissimilarity threshold in E_l for a pair $(\boldsymbol{x}_i, \boldsymbol{x}_j)$.
$\mathcal{T}_{sim}^{(l)}$ / $\mathcal{T}_{dis}^{(l)}$: Upper bound of $d_{sim}^{(l)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ / Lower bound of $d_{dis}^{(l)}(\boldsymbol{x}_i, \boldsymbol{x}_j)$.	$\mathcal{L}_{attr}^{(l)}(\boldsymbol{x}_t, \boldsymbol{x}_t^+)$ / $\mathcal{L}_{rep}^{(l)}(\boldsymbol{x}_t, \boldsymbol{x}_t^-)$: Attractive/Repulsive loss of $(\boldsymbol{x}_t, \boldsymbol{x}_t^+)$ / $(\boldsymbol{x}_t, \boldsymbol{x}_t^-)$ evaluated in E_l.
L: Number of hidden layers in OAHU.	d: Dimensionality of input data.
S_{hidden}: Number of units in each hidden layer in OAHU.	S_{emb}: Dimensionality of learned metric embedding.
β: Discount factor for the update of $\alpha^{(l)}$.	s: Smooth factor for the update of $\alpha^{(l)}$.
η: Learning rate.	$W^{(l)}$ / $\Theta^{(l)}$: Parameter of l^{th} hidden layer/Parameter of l^{th} metric embedding layer.

Table 1: Commonly used symbols and terms

the existence of such triplet stream and focus on addressing the challenges of online metric learning in such scenarios.

3.2 Overview

To learn a metric model with adaptive complexity, we need to address the following question: *when* and *how* to change the "complexity" of metric model in an *online* setting? In this section, we will discuss the details of the proposed framework OAHU that addresses the question in a data-driven manner.

Figure 1 illustrates the structure of our OAHU framework. Inspired by recent works [11, 24], we use an over-complete network and automatically adapt its effective depth to learn a metric function with an appropriate complexity based on input constraints. Consider a neural network with L hidden layers: we connect an independent embedding layer to the network input layer and each of these hidden layers. Every embedding layer represents a space where similar instances are closer to each other and vice-versa. Therefore, unlike conventional online metric learning solutions that usually learns a linear model, OAHU is an ensemble of models with varying complexities that share low-level knowledge with each other. For convenience, let $E_l \in \{E_0, E_1, E_2, \ldots, E_L\}$ denote the l^{th} metric model in OAHU, i.e., the network branch starting from input layer to the l^{th} metric embedding layer, as illustrated in Figure 1. The simplest model in OAHU is E_0, which represents a linear transformation from the input feature space to the metric embedding space. A weight $\alpha^{(l)} \in [0, 1]$ is assigned to E_l, measuring its importance in OAHU.

For a triplet constraint $(\boldsymbol{x}_t, \boldsymbol{x}_t^+, \boldsymbol{x}_t^-)$ that arrives at time t, its metric embedding $f^{(l)}(\boldsymbol{x}_t^*)$ generated by E_l is

$$f^{(l)}(\boldsymbol{x}_t^*) = h^{(l)}\Theta^{(l)} \tag{1}$$

where $h^{(l)} = \sigma(W^{(l)}h^{(l-1)})$, with $l \geq 1, l \in \mathbb{N}$, and $h^{(0)} = \boldsymbol{x}_t^*$.

Here $\boldsymbol{x}_t^*$ denotes any anchor ($\boldsymbol{x}_t$), positive ($\boldsymbol{x}_t^+$) or negative ($\boldsymbol{x}_t^-$) instance based on the definition in Section 3.1, and $h^{(l)}$ represents the activation of l^{th} hidden layer. Note that we also explicitly limit the learned metric embedding $f^{(l)}(\boldsymbol{x}_t^*)$ to reside on a unit sphere, i.e., $||f^{(l)}(\boldsymbol{x}_t^*)||_2 = 1$, to reduce the potential model search space and accelerate the training of OAHU.

During the training phase, for every arriving triplet $(\boldsymbol{x}_t, \boldsymbol{x}_t^+, \boldsymbol{x}_t^-)$, OAHU first retrieves the metric embedding $f^{(l)}(\boldsymbol{x}_t^*)$ from l^{th} metric model according to Eq. 1, and then produces a local loss $\mathcal{L}^{(l)}$ for E_l by evaluating the similarity and dissimilarity errors based on $f^{(l)}(\boldsymbol{x}_t^*)$. Thus, the overall loss introduced by this triplet is given by

$$\mathcal{L}_{overall}(\boldsymbol{x}_t, \boldsymbol{x}_t^+, \boldsymbol{x}_t^-) = \sum_{l=0}^{L} \alpha^{(l)} \cdot \mathcal{L}^{(l)}(\boldsymbol{x}_t, \boldsymbol{x}_t^+, \boldsymbol{x}_t^-) \tag{2}$$

Since the new architecture of OAHU introduces two sets of new parameters, $\Theta^{(l)}$ (parameters of $f^{(l)}$) and $\alpha^{(l)}$, we need to learn $\Theta^{(l)}$, $\alpha^{(l)}$ and $W^{(l)}$ during the online learning phase. Therefore, the final optimization problem to solve in OAHU at time t is given by

$$\begin{aligned} &\underset{\Theta^{(l)}, W^{(l)}, \alpha^{(l)}}{\text{minimize}} && \mathcal{L}_{overall} \\ &\text{subject to} && ||f^{(l)}(\boldsymbol{x}_t^*)||_2 = 1, \forall l = 0, \ldots, L. \end{aligned} \tag{3}$$

To solve the above optimization problem, we propose a novel Adaptive-Bound Triplet Loss (ABTL) in Section 3.3 to estimate $\mathcal{L}^{(l)}$ and introduce a novel Adaptive Hedge Update (AHU) method in Section 3.4 for online updating $\Theta^{(l)}$, $W^{(l)}$ and $\alpha^{(l)}$.

3.3 Adaptive-Bound Triplet Loss

3.3.1 Limitations of Existing Pairwise/Triplet Loss. Existing online metric learning solutions typically optimize a pairwise or triplet loss in order to learn a similarity metric. A widely-accepted pairwise loss [22] and a common triplet loss [19] are shown in Equation 4 and 5 respectively.

$$L(x_t, x_t') = \max\{0, y_t(d^2(x_t, x_t') - b) + 1\} \tag{4}$$

$$L(x_t, x_t^+, x_t^-) = \max\{0, b + d(x_t, x_t^+) - d(x_t, x_t^-)\} \tag{5}$$

Here $y_t \in \{+1, -1\}$ denotes whether x_t is similar (+1) or dissimilar (−1) to x_t' and $b \in \mathcal{R}$ is a user-specified fixed margin. Compared to the triplet loss that simultaneously evaluates both similarity and dissimilarity relations, the pairwise loss can only focus on one of the relations at a time, which leads to a poor metric quality. On the other hand, without a proper margin, the loose restriction of triplet loss can result in many failure cases. Here, a failure case is a triplet that produces a zero loss value, where dissimilar instances are closer than similar instances. As the example shown in Figure 2, for an input triplet (x_t, x_t^+, x_t^-), although x_t^+ is closer to x_t^- and further from x_t, models optimizing triplet loss would incorrectly ignore this constraint due to its zero loss value. In addition, selecting an appropriate margin is highly data-dependent, and requires extensive domain knowledge. Furthermore, as the model improves performance over time, a larger margin is required to avoid failure cases.

Next, we describe an adaptive-bound triplet loss that attempts to solve these drawbacks.

3.3.2 Adaptive-Bound Triplet Loss (ABTL). We define that instances of the same classes are mutually attractive to each other, while instances of different classes are mutually repulsive to each other. Therefore, for any input triplet constraint $(\boldsymbol{x}_t, \boldsymbol{x}_t^+, \boldsymbol{x}_t^-)$, there is an attractive loss $\mathcal{L}_{attr} \in [0, 1]$ between $\boldsymbol{x}_t$ and $\boldsymbol{x}_t^+$, and a repulsive

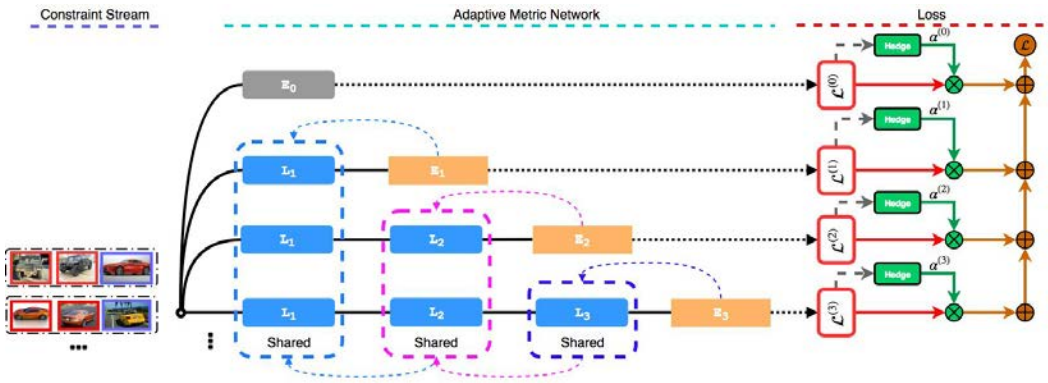


Figure 1: Overview of the OAHU. The chromatic dashed arrows represent update contributions and directions. Each $L_i \in \{L_1, L_2, \ldots\}$ represents a linear transformation layer followed by a ReLU activation. $\{E_0, E_1, \ldots\}$ are the embedding layers connected to corresponding input or hidden layers. Note that E_0 here represents a linear metric model, i.e., a linear transformation from the input feature space to the embedding space.

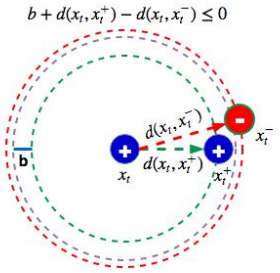


Figure 2: A failure case of triplet loss.

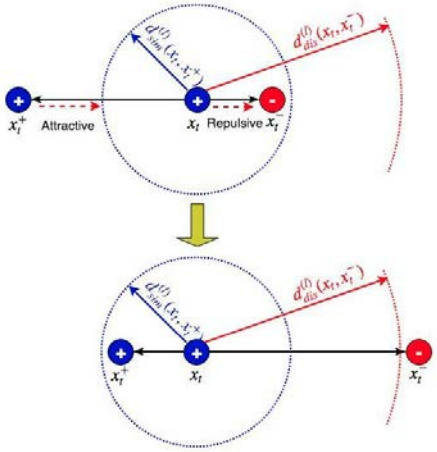


Figure 3: Schematic illustration of ABTL.

loss $\mathcal{L}_{rep} \in [0, 1]$ between $\boldsymbol{x_t}$ and $\boldsymbol{x_t^-}$. In the l^{th} metric model $E_l \in \{E_0, \ldots, E_L\}$, we focus on a distance measure $D^{(l)}(\boldsymbol{x_i}, \boldsymbol{x_j})$ defined as follows:

$$D^{(l)}(\boldsymbol{x_i}, \boldsymbol{x_j}) = ||f^{(l)}(\boldsymbol{x_i}) - f^{(l)}(\boldsymbol{x_j})||_2, \quad \forall l = 0, 1, 2, \ldots, L \tag{6}$$

where $f^{(l)}(\boldsymbol{x})$ denotes the embedding generated by E_l. Since $||f^{(l)}(\boldsymbol{x})||_2$ is restricted to be 1, we have $D^{(l)}(\boldsymbol{x_i}, \boldsymbol{x_j}) \in [0, 2]$. In each iteration, let $D^{(l)}_{orig}(\boldsymbol{x_1}, \boldsymbol{x_2})$ denote the distance between $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ before applying the update to $f^{(l)}$ (i.e., updating $\Theta^{(l)}$ and $\{W^{(j)}\}_{j=0}^{l}$) and $D^{(l)}_{update}(\boldsymbol{x_1}, \boldsymbol{x_2})$ denote the distance after applying the update.

Figure 3 illustrates the main idea of ABTL. The objective is to have the distance $D^{(l)}_{update}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$ of two similar instances $\boldsymbol{x_t}$ and $\boldsymbol{x_t^+}$ to be less than or equal to a similarity threshold $d^{(l)}_{sim}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$ so that the attractive loss $\mathcal{L}^{(l)}_{attr}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$ drops to zero; on the other hand, for two dissimilar instances $\boldsymbol{x_t}$ and $\boldsymbol{x_t^-}$, we desire their distance $D^{(l)}_{update}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$ to be greater than or equal to a dissimilarity threshold $d^{(l)}_{dis}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$, thereby reducing the repulsive loss $\mathcal{L}^{(l)}_{rep}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$ to zero. Equation 7 presents these constraints mathematically.

$$\begin{cases} D^{(l)}_{update}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) \leq d^{(l)}_{sim}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) \\ D^{(l)}_{update}(\boldsymbol{x_t}, \boldsymbol{x_t^-}) \geq d^{(l)}_{dis}(\boldsymbol{x_t}, \boldsymbol{x_t^-}) \end{cases} \tag{7}$$

Unfortunately, directly finding an appropriate $d^{(l)}_{sim}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$ or $d^{(l)}_{dis}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$ is difficult, since it varies with different input constraints. Therefore, we introduce a user-specified hyper-parameter $\tau > 0$ to explicitly restrict the range of both thresholds. Let $\mathcal{T}^{(l)}_{sim}$ denote the upper bound of $d^{(l)}_{sim}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$, and $\mathcal{T}^{(l)}_{dis}$ denote the lower bound of $d^{(l)}_{dis}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$,

$$\begin{cases} \mathcal{T}^{(l)}_{sim} = \tau \geq d^{(l)}_{sim}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) \\ \mathcal{T}^{(l)}_{dis} = 2 - \tau \leq d^{(l)}_{dis}(\boldsymbol{x_t}, \boldsymbol{x_t^-}) \end{cases} \tag{8}$$

Without loss of generality, we propose

$$\begin{cases} d^{(l)}_{sim}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) = a_1 e^{D^{(l)}_{orig}(\boldsymbol{x_t}, \boldsymbol{x_t^+})} + b_1 \\ d^{(l)}_{dis}(\boldsymbol{x_t}, \boldsymbol{x_t^-}) = -a_2 e^{-D^{(l)}_{orig}(\boldsymbol{x_t}, \boldsymbol{x_t^-})} + b_2 \end{cases} \tag{9}$$

where a_1, b_1, a_2 and b_2 are variables need to be determined. Note that $d^{(l)}_{sim}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$ and $d^{(l)}_{dis}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$ are monotonically increasing

and decreasing, respectively. Therefore, $d_{sim}^{(l)}$ preserves the information of original relative similarity, i.e., $d_{sim}^{(l)}(\boldsymbol{x_1}, \boldsymbol{x_2}) < d_{sim}^{(l)}(\boldsymbol{x_1}, \boldsymbol{x_3})$ if $D_{orig}^{(l)}(\boldsymbol{x_1}, \boldsymbol{x_2}) < D_{orig}^{(l)}(\boldsymbol{x_1}, \boldsymbol{x_3})$. This property is critical to many real-world applications that require a fine-grained similarity comparison, such as image retrieval.

The values of a_1, b_1, a_2 and b_2 are then determined by the following boundary conditions:

$$\begin{cases} d_{sim}^{(l)}|_{D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+})=2} \leq \mathcal{T}_{sim}^{(l)} = \tau & d_{sim}^{(l)}|_{D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+})=0} = 0 \\ d_{dis}^{(l)}|_{D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-})=0} \geq \mathcal{T}_{dis}^{(l)} = 2 - \tau & d_{dis}^{(l)}|_{D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-})=2} = 2 \end{cases} \tag{10}$$

Thus we have

$$\begin{cases} d_{sim}^{(l)}\left(\boldsymbol{x_t}, \boldsymbol{x_t^+}\right) = \frac{\tau}{\mathrm{e}^2 - 1}\left(\mathrm{e}^{D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+})} - 1\right) \\ d_{dis}^{(l)}\left(\boldsymbol{x_t}, \boldsymbol{x_t^-}\right) = -\frac{2-(2-\tau)}{1-\mathrm{e}^{-2}}\left(\mathrm{e}^{-D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-})} - 1\right) + (2 - \tau) \end{cases} \tag{11}$$

Provided with $d_{sim}^{(l)}$ and $d_{dis}^{(l)}$, the attractive loss $\mathcal{L}_{attr}$ and repulsive loss $\mathcal{L}_{rep}$ are determined by

$$\begin{cases} \mathcal{L}_{attr}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) = \max\left\{0, \frac{1}{2-c_1} D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) - \frac{c_1}{2-c_1}\right\} \\ \mathcal{L}_{rep}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-}) = \max\left\{0, \frac{-1}{c_2} D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-}) + 1\right\} \end{cases} \tag{12}$$

where $c_1 = d_{sim}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$ and $c_2 = d_{dis}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$.

Therefore, the local loss $\mathcal{L}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-})$, which measures the similarity and dissimilarity errors of the input triplet $(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-})$ in E_l, is the average of $\mathcal{L}_{attr}^{(l)}$ and $\mathcal{L}_{rep}^{(l)}$:

$$\mathcal{L}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-}) = \frac{1}{2}\left(\mathcal{L}_{attr}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) + \mathcal{L}_{rep}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-})\right) \tag{13}$$

Next, we answer an important question: What is the best value of τ? Here, we demonstrate that the optimal range of τ is $(0, \frac{2}{3})$. The best value of τ is thus empirically selected within this range via cross-validation.

Theorem 3.1. *With $\tau \in (0, \frac{2}{3})$, by optimizing the proposed adaptive-bound triplet loss, different classes are separated in the metric embedding space.*

Proof. Let $D(c_1, c_2)$ denotes the minimal distance between classes c_1 and c_2, i.e., the distance between two closest instances from c_1 and c_2 respectively. Consider an arbitrary quadrupole $(x_1, x_2, x_3, x_4) \in Q$ where $\{x_1, x_2\} \in c_1$, $\{x_3, x_4\} \in c_2$, and Q is the set of all possible quadrupoles generated from class c_1 and c_2. Suppose (x_2, x_3) is the closest dissimilar pair among all possible dissimilar pairs that can be extracted from (x_1, x_2, x_3, x_4) [2]. We first prove that the lower bound of $D(c_1, c_2)$ is given by $\min_{(x_1, x_2, x_3, x_4) \in Q} D^{(l)}(x_1, x_4) - D^{(l)}(x_1, x_2) - D^{(l)}(x_3, x_4)$. Due to the triangle inequality property of p-norm distance, we have

$$\begin{aligned} D^{(l)}(x_1, x_4) &\leq D^{(l)}(x_1, x_2) + D^{(l)}(x_2, x_4) \\ &\leq D^{(l)}(x_1, x_2) + D^{(l)}(x_2, x_3) + D^{(l)}(x_3, x_4) \end{aligned} \tag{14}$$

[2] it can always be achieved by re-arranging symbols of x_1,x_2,x_3 and x_4

Therefore,

$$\begin{aligned} D(c_1, c_2) &= \min_{(x_1, x_2, x_3, x_4) \in Q} D^{(l)}(x_2, x_3) \\ &\geq \min_{(x_1, x_2, x_3, x_4) \in Q} D^{(l)}(x_1, x_4) - D^{(l)}(x_1, x_2) - D^{(l)}(x_3, x_4) \end{aligned} \tag{15}$$

By optimizing the adaptive-bound triplet loss, the following constraints are satisfied

$$\begin{cases} D^{(l)}(x_1, x_2) \leq d_{sim}^{(l)}(x_1, x_2) \leq \mathcal{T}_{sim}^{(l)} \\ D^{(l)}(x_3, x_4) \leq d_{sim}^{(l)}(x_3, x_4) \leq \mathcal{T}_{sim}^{(l)} \\ D^{(l)}(x_1, x_4) \geq d_{dis}^{(l)}(x_1, x_4) \geq \mathcal{T}_{dis}^{(l)} \end{cases} \tag{16}$$

Thus

$$\begin{aligned} D(c_1, c_2) &\geq \min_{(x_1, x_2, x_3, x_4) \in Q} D^{(l)}(x_1, x_4) - D^{(l)}(x_1, x_2) - D^{(l)}(x_3, x_4) \\ &\geq \mathcal{T}_{dis}^{(l)} - 2\mathcal{T}_{sim}^{(l)} \\ &= 2 - \tau - 2\tau \\ &= 2 - 3\tau \end{aligned} \tag{17}$$

if $\tau \in (0, \frac{2}{3})$, we have $3\tau < 2$. Therefore,

$$D(c_1, c_2) \geq 2 - 3\tau > 0 \tag{18}$$

Equation 18 indicates that the minimal distance between class c_1 and c_2 is always positive so that these two classes are separated. □

3.3.3 Discussion. Compared to pairwise and triplet loss, the proposed ABTL has several advantages:

- Similar to triplet loss, ABTL simultaneously evaluates both similarity and dissimilarity errors to obtain a high-quality metric.
- The similarity threshold $d_{sim}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$ is always less than or equal to $D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$ and the dissimilarity threshold $d_{dis}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$ is always greater than or equal to $D_{orig}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$. Thus, every input constraint contributes to model update, which eliminates the failure cases of triplet loss and leads to a higher constraint utilization rate.
- The optimal range of τ is provided with a theoretical proof.

3.4 Adaptive Hedge Update (AHU)

With adaptive-bound triplet loss, the overall loss in OAHU becomes

$$\begin{aligned} \mathcal{L}_{overall}(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-}) &= \sum_{l=0}^{L} \alpha^{(l)} \cdot \mathcal{L}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-}) \\ &= \sum_{l=0}^{L} \frac{\alpha^{(l)}}{2}\left(\mathcal{L}_{attr}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) + \mathcal{L}_{rep}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-})\right) \end{aligned} \tag{19}$$

We propose to learn $\alpha^{(l)}$ using the Hedge Algorithm [9]. Initially, all weights are uniformly distributed, i.e., $\alpha^{(l)} = \frac{1}{L+1}, l = 0, 1, 2, \ldots, L$. At every iteration, for each metric model E_l, OAHU transforms the input triplet constraint into E_l's metric embedding space, where the constraint similarity and dissimilarity errors are

evaluated. Thus $\alpha^{(l)}$ is updated based on the local loss suffered by E_l as:

$$\alpha_{t+1}^{(l)} \leftarrow \begin{cases} \alpha_t^{(l)} \beta^{\mathcal{L}^{(l)}} & \beta^{\min_l \mathcal{L}^{(l)}} \log \mathcal{L}^{(l)} > \beta - 1 \\ \alpha_t^{(l)} [1 - (1-\beta)\mathcal{L}^{(l)}] & otherwise \end{cases} \tag{20}$$

where $\beta \in (0,1)$ is the discount factor. We choose to update $\alpha^{(l)}$ based on Equation 20, to maximize the update of $\alpha^{(l)}$ at each step. In other words, if a metric model produces a low/high $\mathcal{L}^{(l)}$ at time t, its associated weight $\alpha^{(l)}$ will gain as much increment/decrement as possible. Therefore, in OAHU, metric models with good performance are highlighted while models with poor performance are eliminated. Moreover, following Sahoo et.al [21], to avoid the model bias issue [7, 10], i.e., OAHU may unfairly prefer shallow models due to their faster convergence rate and excessively reduces weights of deeper models, we introduce a smooth factor s used to set a minimal weight for each metric model. Thus, after updating weights according to Equation 20, the weights are set as: $\alpha_{t+1}^{(l)} = \max(\alpha_{t+1}^{(l)}, \frac{s}{L+1})$. Finally, at the end of every round, the weights α are normalized such that $\sum_{l=0}^{L} \alpha_{t+1}^{(l)} = 1$.

Learning the parameters $\Theta^{(l)}$ and $W^{(l)}$ is tricky but can be achieved via gradient descent. By applying Online Gradient Descent (OGD) algorithm, the update rules for $\Theta^{(l)}$ and $W^{(l)}$ are given by:

$$\begin{aligned} \Theta_{t+1}^{(l)} &\leftarrow \Theta_t^{(l)} - \eta \nabla_{\Theta_t^{(l)}} \mathcal{L}_{overall}(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-}) \\ &= \Theta_t^{(l)} - \eta \frac{\alpha^{(l)}}{2} \left(\nabla_{\Theta_t^{(l)}} \mathcal{L}_{attr}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) + \nabla_{\Theta_t^{(l)}} \mathcal{L}_{rep}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-}) \right) \end{aligned} \tag{21}$$

$$\begin{aligned} W_{t+1}^{(l)} &\leftarrow W_t^{(l)} - \eta \nabla_{W_t^{(l)}} \mathcal{L}_{overall}(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-}) \\ &= W_t^{(l)} - \eta \sum_{j=l}^{L} \frac{\alpha^{(j)}}{2} \left(\nabla_{W_t^{(l)}} \mathcal{L}_{attr}^{(j)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}) + \nabla_{W_t^{(l)}} \mathcal{L}_{rep}^{(j)}(\boldsymbol{x_t}, \boldsymbol{x_t^-}) \right) \end{aligned} \tag{22}$$

Updating $\Theta^{(l)}$ (Equation 21) is simple since $\Theta^{(l)}$ is specific to E_l, and is not shared across different metric models. In contrast, because deeper models share low-level knowledge with shallower models in OAHU, all models deeper than l layers contribute to the update of $W^{(l)}$, as expressed in Equation 22. Algorithm 1 outlines the training process of OAHU.

3.5 Regret Bound

Overall, hedge algorithm enjoys a regret bound of $R_T \leq \sqrt{TlnN}$ [9] where N is the number of experts and T is the number of trials. In our case, N is the number of metric models (i.e., $L+1$) and T is the number of input triplet constraints. The average worst-case convergence rate of OAHU is then given by $O(\sqrt{ln(N)/T})$. Therefore, OAHU is guaranteed to converge after learning from sufficiently many triplets.

3.6 Application of OAHU

In this part, we discuss the deployment of OAHU in different practical applications. Although OAHU can be applied in many applications, we mainly focus on the three most common tasks: classification, similarity comparison, and analogue retrieval. Note that the following discussion describes only one possible way of deployment for each task, and other deployment methods can exist.

Algorithm 1 OAHU: Online Metric Learning with Adaptive Hedge Update

Require: Discount Factor $\beta \in (0,1)$; Smooth Factor s; Control Parameter $\tau \in (0, \frac{2}{3})$; Learning rate η; A randomly initialized ANN with L hidden layers that is parameterized by $\Theta^{(l)}$,$W^{(l)}$ and $\alpha^{(l)}$.

Ensure: $\alpha^{(l)}$, $\Theta^{(l)}$ and $W^{(l)}$

1: Initialize $\alpha^{(l)} = \frac{1}{L+1}, \forall l = 0, 1, \ldots, L$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: Receive a triplet constraint $(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-})$
4: Transform and retrieve the metric embedding $f^{(l)}(\boldsymbol{x_t})$, $f^{(l)}(\boldsymbol{x_t^+})$ and $f^{(l)}(\boldsymbol{x_t^-})$ from each model E_l.
5: Evaluate $\mathcal{L}_{attr}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+})$ and $\mathcal{L}_{rep}^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^-})$ (Eq. 12), $\forall l = 0, 1, \ldots, L$.
6: Compute $L^{(l)}(\boldsymbol{x_t}, \boldsymbol{x_t^+}, \boldsymbol{x_t^-}), \forall l = 0, 1, \ldots, L$ as per Eq. 13.
7: Update $\Theta_{t+1}^{(l)}, \forall l = 0, 1, 2, \ldots, L$ as per Eq. 21.
8: Update $W_{t+1}^{(l)}, \forall l = 0, 1, 2, \ldots, L$ as per Eq. 22.
9: Update $\alpha_{t+1}^{(l)}, \forall l = 0, 1, 2, \ldots, L$ as per Eq. 20.
10: $\alpha_{t+1}^{(l)} = \max(\alpha_{t+1}^{(l)}, \frac{s}{L+1}), \forall l = 0, 1, 2, \ldots, L$.
11: Normalize $\alpha_{t+1}^{(l)}$, i.e., $\alpha_{t+1}^{(l)} = \frac{\alpha_{t+1}^{(l)}}{\sum_{j=0}^{L} \alpha_{t+1}^{(j)}}, \forall l = 0, 1, 2, \ldots, L$.
12: **end for**

3.6.1 Classification. Given a training dataset $\mathcal{D} = \{(\boldsymbol{x_i}, y_i)\}_{i=1}^{n}$ where $\boldsymbol{x_i} \in \mathcal{R}^d$ is an training instance and $y_i \in \{1, 2, \ldots, c\}$ is its associated label, we determine the label $y \in \{1, 2, \ldots, c\}$ of a test instance $\boldsymbol{x} \in \mathcal{R}^d$ as follows:

- For every metric model E_l in OAHU, we find k nearest neighbors of $\boldsymbol{x}$ measured by Eq. 6, each of which is a candidate similar to $\boldsymbol{x}$. Hence, $k(L+1)$ candidates in total are found in OAHU. Let $\mathcal{N} = \{\{(\boldsymbol{x_j^{(l)}}, y_j^{(l)})\}_{j=1}^{k}\}_{l=0}^{L} = \{\mathcal{N}_l\}_{l=0}^{L}$ denotes these candidates where $\boldsymbol{x_j^{(l)}}$ is the j^{th} neighbor of $\boldsymbol{x}$ found in E_l and $y_j^{(l)}$ is its associated label. The similarity score between $\boldsymbol{x}$ and $\boldsymbol{x_j^{(l)}}$ is given by

$$S(\boldsymbol{x}, \boldsymbol{x_j^{(l)}}) = \mathrm{e}^{-\frac{D^{(l)}(\boldsymbol{x_j^{(l)}}, \boldsymbol{x}) - d_{min}}{d_{max} - d_{min}}} \cdot \alpha^{(l)} \tag{23}$$

  where $d_{min} = \min_{\mathcal{N}_l} D^{(l)}(\boldsymbol{x_j^{(l)}}, \boldsymbol{x})$ and $d_{max} = \max_{\mathcal{N}_l} D^{(l)}(\boldsymbol{x_j^{(l)}}, \boldsymbol{x})$.
- Then, the class association score of each class c_i is calculated by:

$$S(c_i) = \sum_{y_j^{(l)} = c_i} S(\boldsymbol{x}, \boldsymbol{x_j^{(l)}}) \tag{24}$$

- The final predicted label y of $\boldsymbol{x}$ in OAHU is determined as:

$$y = \arg\max_{c_i} S(c_i) \tag{25}$$

3.6.2 Similarity Comparison. The task of similarity comparison is to determine whether a given pair $(\boldsymbol{x_1}, \boldsymbol{x_2})$ is similar or not. To

do this, a user-specified threshold $\mathcal{T} \in (0, 1)$ is needed. The steps below are followed to perform the similarity comparison in OAHU.

- First, for every metric model E_l, we compute the distance $D^{(l)}(\boldsymbol{x_1}, \boldsymbol{x_2})$ between $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$. Since $D^{(l)}(\boldsymbol{x_1}, \boldsymbol{x_2}) \in [0, 2]$, we divide $D^{(l)}(\boldsymbol{x_1}, \boldsymbol{x_2})$ by 2 for normalization.
- The similarity probability of $(\boldsymbol{x_1}, \boldsymbol{x_2})$ is then determined by

$$P(\boldsymbol{x_1}, \boldsymbol{x_2}) = \sum_{l=0,\ldots,L} \alpha^{(l)} \cdot p_l \tag{26}$$

  where $p_l = 1$ if $D^{(l)}(\boldsymbol{x_1}, \boldsymbol{x_2})/2 < \mathcal{T}$; Otherwise, $p_l = 0$.
- $(\boldsymbol{x_1}, \boldsymbol{x_2})$ is similar if $P(\boldsymbol{x_1}, \boldsymbol{x_2}) \geq 0.5$ and is dissimilar if $P(\boldsymbol{x_1}, \boldsymbol{x_2}) < 0.5$.

3.6.3 Analogue Retrieval. Given a query item $\boldsymbol{x}$, the task of analogue retrieval is to find k items that are most similar to $\boldsymbol{x}$ in a database $\mathcal{D}$. Note that analogue retrieval is a simplified "classification" task where we only find similar items and ignore the labeling issue.

- We first retrieve the $k(L + 1)$ candidates that are similar to $\boldsymbol{x}$ following the same process in classification task. Let $\mathcal{N} = \{\{\boldsymbol{x_j^{(l)}}\}_{j=1}^{k}\}_{l=0}^{L} = \{\mathcal{N}_l\}_{l=0}^{L}$ denote these candidates. The similarity score between $\boldsymbol{x_j^{(l)}}$ and $\boldsymbol{x}$ is:

$$Sim(\boldsymbol{x_j^{(l)}}, \boldsymbol{x}) = \mathrm{e}^{-\frac{D^{(l)}(\boldsymbol{x_j^{(l)}}, \boldsymbol{x}) - d_{min}}{d_{max} - d_{min}}} \cdot \alpha^{(l)} \tag{27}$$

  where $d_{min} = \min_{\mathcal{N}_l} D^{(l)}(\boldsymbol{x_j^{(l)}}, \boldsymbol{x})$ and $d_{max} = \max_{\mathcal{N}_l} D^{(l)}(\boldsymbol{x_j^{(l)}}, \boldsymbol{x})$.
- Then we sort these items in descending order based on their similarity scores and keep only the top k distinct items as the final retrieval result.

3.7 Time and Space Complexity

Overall, the execution overhead of OAHU mainly arises from the training of the ANN-based metric model, especially from the gradient computation. Assume that the maximum time complexity for calculating the gradient of one layer is a constant C. The overall time complexity of OAHU is simply $O(nL^2C)$, where n is the number of input constraints and L is the number of hidden layers. Therefore, with pre-specified L and ANN architecture, OAHU is an efficient online metric learning algorithm that runs in linear time. Let d, S_{hidden} and S_{emb} denote the dimensionality of input data, number of units in each hidden layer and metric embedding size respectively. The space complexity of OAHU is given by $O\Big(d \cdot S_{emb} + L \cdot S_{hidden}(d + S_{emb}) + \frac{L(L-1)}{2} S_{hidden}^2\Big)$.

3.8 Extendability

As discussed in Section 3.6, OAHU is a fundamental module that can perform classification, similarity comparison, or analogue retrieval tasks independently. Therefore, it can be easily plugged into any existing deep models (e.g., a CNN or RNN) by replacing the classifier, i.e., fully-connected layers, with OAHU. The parameters of those feature-extraction layers (e.g., a convolutional layer) can be fine-tuned by optimizing $\mathcal{L}_{overall}$ directly. However, in this paper, to prove the validity of the proposed approach, we only focus on OAHU itself and leave the deep model extension for future work.

Task	Dataset	# features	# classes	# instances
Image Classification	FASHION-MNIST	784	10	70,000
	EMNIST	784	47	131,600
	CIFAR-10	1024	10	60,000
Face Verification	LFW	73	5749	13,233
Image Retrieval	CARS-196	4096	196	16,185
	CIFAR-100	4096	100	60,000

Table 2: Description of Datasets

4 EMPIRICAL EVALUATION

To verify the effectiveness of our method, we evaluate OAHU on three typical tasks, including (1) image classification (***classification***), (2) face verification (***similarity comparison***), and (3) image retrieval (***analogue retrieval***).

4.1 Baselines

To examine the quality of metrics learned in OAHU, we compare it with several state-of-the-art online metric learning algorithms. (1) LEGO [12] (pairwise constraint): an online metric learning algorithm that learns a Mahalanobis metric based on LogDet regularization and gradient descent. (2) RDML [14] (pairwise constraint): an online algorithm for regularized distance metric learning that learns a Mahalanobis metric with a regret bound. (3) OASIS [6] (triplet constraint): an online algorithm for scalable image similarity learning that learns a bilinear similarity measure over sparse representations. Finally, (4) OPML [19] (triplet constraint): a one-pass closed-form solution for online metric learning that learns a Mahalanobis metric with a low computational cost.

4.2 Implementation

We have implemented OAHU using Python 3.6.2 and Pytorch 0.4.0 library. All baseline methods were based on code released by corresponding authors, except LEGO and RDML. Due to the unavailability of a fully functional code of LEGO and RDML, we use our own implementation based on the authors' description [12, 14]. Hyper-parameter setting of these approaches, including OAHU, varies with different tasks, and will be discussed later for each specific task.

4.3 Image Classification

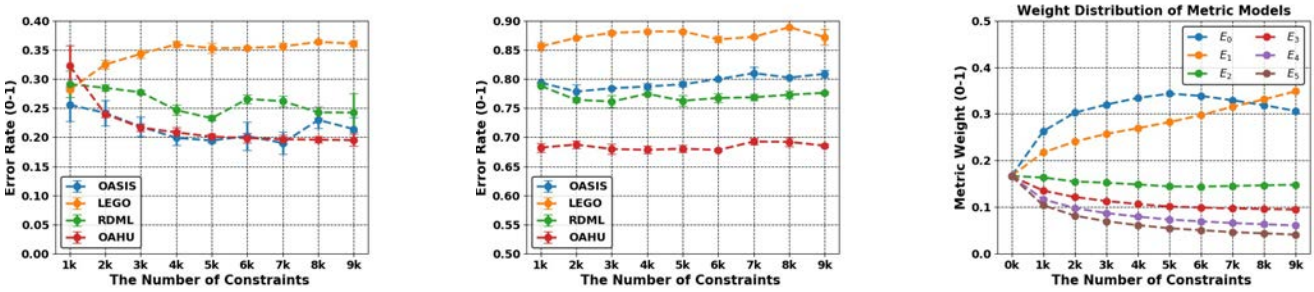
4.3.1 Dataset and Classifier. To demonstrate the validacy of OAHU on complex practical applications, we use three publicly available real-world benchmark image datasets, including **FASHION-MNIST**[3] [26], **EMNIST**[4] [8] and **CIFAR-10**[5] [16] for evaluation. To be consistent with other image datasets, we convert images of CIFAR-10 into grayscale through the OpenCV API [3], resulting in 1024 features. The *k-NN* classifier is employed here, since it is widely used for classification tasks with only one parameter, and is also consistent with the deployment of OAHU discussed in Section 3.6.1. Details of these datasets are listed in "Image Classification" part of Table 2.

4.3.2 Experiment Setup and Constraint Generation. To perform the experiment, we first randomly shuffle each dataset and then divide

[3] https://github.com/zalandoresearch/fashion-mnist
[4] https://www.nist.gov/itl/iad/image-group/emnist-dataset
[5] https://www.cs.toronto.edu/ kriz/cifar.html

Methods	FASHION-MNIST			EMNIST			CIFAR-10			win/tie/loss
	Error Rate	F_1	$\mathcal{U}$	*Error Rate*	F_1	$\mathcal{U}$	*Error Rate*	F_1	$\mathcal{U}$	
LEGO	0.35±0.01 •	0.64±0.00	0.58±0.00	0.84±0.00 •	0.14±0.02	0.66±0.01	0.88±0.00 •	0.05±0.02	0.59±0.00	**3/0/0**
RDML	0.22±0.01 •	0.78±0.00	0.47±0.01	0.64±0.03 •	0.32±0.05	0.50±0.00	0.76±0.01 •	0.24±0.01	0.50±0.01	**3/0/0**
OASIS	0.21±0.02 •	0.79±0.01	0.14±0.00	0.41±0.01 •	0.59±0.00	0.24±0.00	0.79±0.02 •	0.20±0.03	0.47±0.00	**3/0/0**
OPML	0.23±0.00 •	0.78±0.00	0.30±0.01	0.43±0.01 •	0.51±0.02	0.45±0.00	0.83±0.01 •	0.16±0.00	0.48±0.04	**3/0/0**
OAHU	**0.18±0.00**	**0.81±0.00**	**1.00±0.00**	**0.35±0.01**	**0.64±0.01**	**1.00±0.00**	**0.68±0.00**	**0.31±0.00**	**1.00±0.00**	-

Table 3: Comparison of classification performance on competing methods over benchmark image datasets. •/◦ indicates OAHU performs statistically better/worse (0.05 significance level) than the respective method according to the p-values. The statistics of win/tie/loss is also included. Both mean and standard deviation of error rates, constraint utilization $\mathcal{U}$ and macro F_1 scores are reported. 0.00 **denotes a value less than** 0.005**.**



(a) FASHION-MNIST (b) CIFAR-10 (c)

Figure 4: (a) / (b) Error rates of competing methods with increasing number of constraints on FASHION-MNIST / CIFAR-10 dataset. (c) The evolvement of metric weight distribution in OAHU with increasing number of constraints on FASHION-MNIST dataset.

it into development and test sets with the split ratio of 1 : 1. Hyper-parameters of these baseline approaches were set based on values reported by the authors and fine-tuned via 10-fold cross-validation on the development set. Because OPML applies a one-pass triplet construction process to generate the constraints, we simply provide the training data in the development set as the input for OPML. For other approaches including OAHU, we first randomly sample 5,000 seeds (i.e., pairwise or triplet constraints) from the training data and then construct 5,000 more constraints by taking transitive closure over these seeds. Note that the same number of constraints were adopted by the authors of LEGO [12] and OPML [19] when comparing their approaches with baselines. Thus our comparison is fair. In OAHU, we set $\beta = 0.99$, $L = 5$, $S_{hidden} = 100$, $s = 0.1$, $\eta = 0.3$ and $S_{emb} = 50$ as default. The value of τ was found by 10-fold cross-validation on the development set. To reduce any bias that may result from random partition, all the classification results are averaged over 10 individual runs with independent random shuffle processes.

4.3.3 Evaluation Metric. Similar to most of the baselines, e.g., LEGO [12] and OPML [19], we adopt the error rate and macro F_1 score[6] as the evaluation criterion. Moreover, the p-values of student's t-test were calculated to check statistical significance. In addition, the statistics of win/tie/loss are reported according to the obtained p-values. The constraint utilization $\mathcal{U}$, i.e., the fraction of input constraints that actually contribute to model update, is also reported for each method.

4.3.4 Analysis. Table 3 compares the classification performance of OAHU with baseline methods on benchmark datasets. We observe that LEGO and RDML perform poorly on complex image datasets, e.g., EMNIST (47 classes), since they learn a Mahalanobis metric from pairwise constraints, and are biased to either the similarity or dissimilarity relation when updating model parameters. OASIS performs better than OPML because the bilinear metric eliminates the positive and symmetric requirement of the covariance matrix, i.e., A in $d(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1^T A \boldsymbol{x}_2$ which results in a larger hypothesis space [6]. However, OAHU outperforms all the baseline approaches by providing not only significantly lower error rate, but also higher F_1 score, indicating the best overall classification performance among all competing methods.

To illustrate the effect of different numbers of constraints on the learning of a metric, we vary the input constraint numbers[7] for all competing methods except OPML. Here we do not compare with OPML because it internally determines the number of triplets to use via a one-pass triplet construction process. Therefore, explicitly varying the input triplet number for OPML is impossible. Figure 4a and Figure 4b show the comparison of error rates on FASHION-MNIST and CIFAR-10 dataset respectively. The large variance of error rates observed in both OASIS and RDML (i.e., error bars in the figures) indicates the strong correlation of their classification performance with the quality of input constraints. All baseline approaches present an unstable, and sometimes, even worse performance with an increasing number of constraints. This is because these baselines are biased to those "hard" constraints

[6] http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

[7] It is done by first generating 10,000 constraints as discussed in Section 4.3.2 and then uniformly sample the desired number of constraints from them.

leading to positive loss values and incorrectly ignore the emerging failure cases (see Figure 2). In contrast, OAHU provides the best and most stable classification performance with minimal variance, indicating its robustness to the quality of input constraints and better capability of capturing the semantic similarity and dissimilarity relations in data.

OAHU is better mainly because it not only learns from every input constraint by optimizing the adaptive-bound triplet loss (see constraint utilization $\mathcal{U}$ in Table 3), but also automatically adapts the model complexity to achieve better performance (see Figure 4c).

4.4 Face Verification

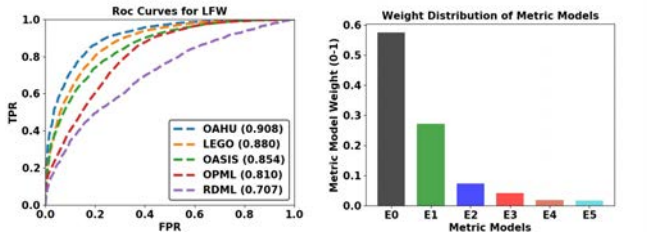
4.4.1 Dataset and Experiment Setup. For face verification, we evaluate our method on Labeled Faces in the Wild Database (LFW) [18]. This dataset has two views: View 1 is used for development purpose (contains a training and a test set), and View 2 is taken as a benchmark for comparison (i.e., a 10-fold cross validation set containing pairwise images). The goal of face verification in LFW is to determine whether a pair of face images belongs to the same person. Since OASIS, OPML, and OAHU are triplet-based methods, following the setting in OPML [19], we adopt the image unrestricted configuration to conduct experiments for a fair comparison. In the image unrestricted configuration, the label information (i.e. actual names) in training data can be used and as many pairs or triplets as one desires can be formulated. We use View 1 for hyper-parameter tuning and evaluate the performance of all competing algorithms on each fold (i.e., 300 intra-person and 300 inter-person pairs) in View 2. Note that the default parameter values in OAHU and baselines were set using the same values as discussed in Section 4.3.2. Attribute features of LFW are used in the experiment so that the result can be easily reproduced and the comparison is fair. Attribute features[8] (73 dimension) provided by Kumar et al. [17] are "high-level" features describing nameable attributes such as race, age etc., of a face image. Details of LFW are described in "Face Verification" part in Table 2.

4.4.2 Constraint Generation and Evaluation Metric. We follow the same process described in Section 4.3.2 to generate constraints for training purpose. However, we add an additional step here: if a generated constraint contains any pair of instances in the validation set (i.e., all pairs in View 1) or the test set (i.e., all folds in View 2), we simply discard it and re-sample a new constraint. This step is repeated until the new constraint does not contain any pair in both sets. In the experiment, 10,000 pair or triplet constraints in total are generated for training.

Note, due to the one-pass triplet generation strategy of OPML, we let itself determine the training triplets. For each test image pair, we first compute the distance between them measured by the learned metric, and then normalize the distance into $[0, 1]$. The widely accepted Receiver Operating Characteristic (ROC) curve and Area under Curve (AUC) are adopted as the evaluation metric.

4.4.3 Analysis. The ROC curves of all competing methods are given in Figure 5a with corresponding AUC values calculated. It can be observed that the proposed approach OAHU outperforms all baseline approaches by providing significantly higher AUC value.

[8]http://www.cs.columbia.edu/CAVE/databases/pubfig/download/lfw_attributes.txt



(a) (b)

Figure 5: (a) ROC Curves of OAHU and contrastive methods on LFW dataset. AUC value of each method is given in bracket. (b) Metric weight distribution of OAHU.



(a) Failure Cases **(b) Successful Cases**

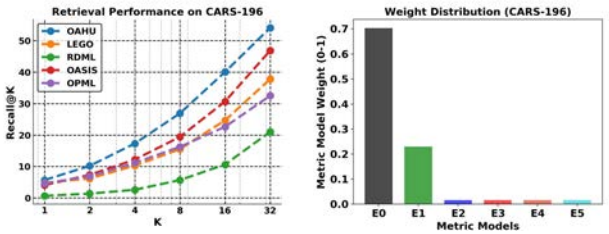
Figure 6: Face verification examples in the test set (View 2) ($\mathcal{T} = 0.55$). For images belonging to same person, $P(\boldsymbol{x}_1, \boldsymbol{x}_2)$ should be close to 1 and for images belonging to different people, $P(\boldsymbol{x}_1, \boldsymbol{x}_2)$ should drop to 0.

As demonstrated in Figure 5b, OAHU is better mainly because it automatically adapts the model complexity to incorporate non-linear models, which help to separate instances that are difficult to distinguish in simple linear models. Figure 6 presents some example comparisons in the test set for both successful and failure cases. Here, the comparison probability $P(\boldsymbol{x}_1, \boldsymbol{x}_2)$ computed based on Eq. 26 is expected to approach 1 if both images are from the same person and should be close to 0 if they belong to different people. Apparently, although OAHU performs much better than baselines, it may still fail in some cases where severe concept drift, e.g. aging and shave, are present.

4.5 Image Retrieval

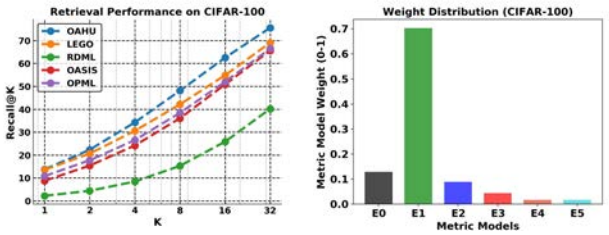
4.5.1 Dataset and Experiment Setup. To demonstrate the performance of OAHU on image retrieval task, we show experiments on two benchmark datasets, which are CARS-196[9] [15] and CIFAR-100[10] [16]. The CARS-196 dataset is divided into 8,144 training images and 8,401 test images, where each class has been split roughly in a 50-50 ratio. Similarly, we randomly split CIFAR-100 into 30,000 training images and 30,000 test images by equally dividing images of each class. For each image in CARS-196 and CIFAR-100, its deep feature, i.e., the output of the last convolutional layer, is extracted from a VGG-19 model pretrained on the ImageNet [20] dataset. Details of these 2 datasets are listed in the "Image Retrieval" part of Table 2. We form the development set of each dataset by including

[9]https://ai.stanford.edu/ jkrause/cars/car_dataset.html
[10]https://www.cs.toronto.edu/ kriz/cifar.html



(a) (b)

Figure 7: (a) Recall@K score on the test split of CARS-196. (b) Metric weight distribution of OAHU on CARS-196.



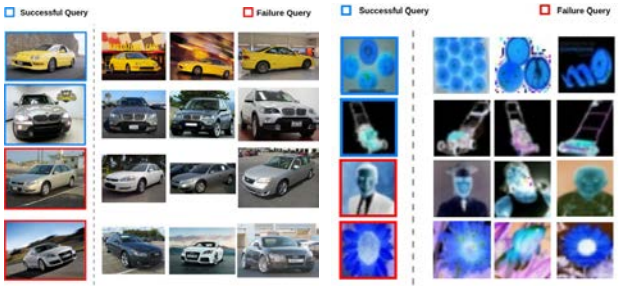
(a) (b)

Figure 8: (a) Recall@K score on the test split of CIFAR-100. (b) Metric weight distribution of OAHU on CIFAR-100.

all its training images. For each dataset, hyper-parameters of baseline approaches were set based on values reported by the authors, and fine-tuned via 10-fold cross-validation on the development set. In OAHU, we set $\beta = 0.99$, $L = 5$, $S_{hidden} = 100$, $s = 0.1$, $\eta = 0.3$ and $S_{emb} = 50$ as default. The value of τ in OAHU was found via 10-fold cross-validation on the development set.

4.5.2 Constraint Generation and Evaluation Metric. With the training images in the development set, we generate the training constraints in the same way as Section 4.3.2. However, in this experiment, we sample 50,000 constraints (25,000 seeds) for each dataset due to the vast amount of classes included in them. The Recall@K metric [13] is applied for performance evaluation. Each test image (query) first retrieves K most similar images from the test set and receives a score of 1 if an image of the same class is retrieved among these K images and 0 otherwise. Recall@K averages this score over all the images.

4.5.3 Analysis. The Recall@K scores on the test set of CARS-196 and CIFAR-100 are shown in Figure 7a and Figure 8a, respectively. We observe that OAHU outperforms all baseline approaches by providing significantly higher Recall@K scores. In addition, the Recall@K score of OAHU grows more rapidly than that provided by most baselines. Figure 7b and Figure 8b illustrate the metric weight distribution of OAHU on CARS-196 and CIFAR-100 respectively. Compared to the baselines that are only linear models, OAHU takes full advantage of non-linear metrics to improve the model expressiveness. Figure 9 shows some examples of successful and failure queries on both datasets. Despite the huge change in the viewpoint,



(a) CARS-196 (b) CIFAR-100

Figure 9: Examples of successful and failure queries on the CARS-196 and CIFAR-100 datasets using our embedding. Images in the first column are query images and the rest are three most similar images. Best viewed on a monitor zoomed in.

configuration, and illumination, our method can successfully retrieve examples from the same class and most retrieval failures come from subtle visual difference among images of different classes.

4.6 Sensitivity of Parameters

The three main parameters in OAHU is the control parameter τ. We vary τ from 0.1 to 0.9 and S_{emb} from 10 to 100 to study their effect on the classification performance. We observe that both classification accuracy (1-error rate) and F_1 score significantly drop if τ is greater than 0.6 but remain unchanged with various τ values if $\tau \leq 0.6$. This observation verifies Theorem 3.1 which states that classes are separated if $\tau \in (0, \frac{2}{3})$. On the other hand, the classification accuracy of OAHU peaks at embedding size $S_{emb} = 50$ where all semantic information are maintained. If we further increase the embedding size, noisy information is going to be introduced which degrades the classification performance. Therefore, we suggest to choose $\tau = 0.1$ and $S_{emb} = 50$ as default.

5 LIMITATIONS AND CONCLUSIONS

In this paper, we propose a novel online metric learning framework, called OAHU, that learns a ANN-based metric with adaptive complexity and a full constraint utilization rate. Compared to the state-of-the-art baselines, OAHU not only obtains the best performance by automatically adapting its complexity to improve the model expressiveness, but also is more robust in terms of the quality of input constraints. Like existing state-of-the-art solutions, OAHU fails when facing severe concept drift data that invalidate the learned similarity metric. We leave the extension of OAHU to severe concept drift scenarios for future work.

6 ACKNOWLEDGMENTS

This material is based upon work supported by NSF award numbers: DMS-1737978, DGE 17236021. OAC-1828467; ARO award number: W911-NF-18-1-0249; IBM faculty award (Research); and NSA awards.

REFERENCES

[1] Charu C. Aggarwal, Zhi-Hua Zhou, Alexander Tuzhilin, Hui Xiong, and Xindong Wu (Eds.). 2015. *2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14-17, 2015*. IEEE Computer Society. http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7373198

[2] Aurélien Bellet, Amaury Habrard, and Marc Sebban. 2013. A Survey on Metric Learning for Feature Vectors and Structured Data. *CoRR* abs/1306.6709 (2013). arXiv:1306.6709 http://arxiv.org/abs/1306.6709

[3] G. Bradski. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).

[4] Casey Breen, Latifur Khan, and Arunkumar Ponnusamy. 2002. Image classification using neural networks and ontologies. In *Proceedings. 13th International Workshop on Database and Expert Systems Applications*. IEEE, 98–102.

[5] Carla E. Brodley and Peter Stone (Eds.). 2014. *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*. AAAI Press. http://www.aaai.org/Library/AAAI/aaai14contents.php

[6] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. 2010. Large Scale Online Learning of Image Similarity Through Ranking. *Journal of Machine Learning Research* 11 (2010), 1109–1135. https://doi.org/10.1145/1756006.1756042

[7] Tianqi Chen, Ian J. Goodfellow, and Jonathon Shlens. 2015. Net2Net: Accelerating Learning via Knowledge Transfer. *CoRR* abs/1511.05641 (2015). arXiv:1511.05641 http://arxiv.org/abs/1511.05641

[8] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. 2017. EMNIST: an extension of MNIST to handwritten letters. *CoRR* abs/1702.05373 (2017). arXiv:1702.05373 http://arxiv.org/abs/1702.05373

[9] Yoav Freund and Robert E. Schapire. 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *J. Comput. Syst. Sci.* 55, 1 (1997), 119–139. https://doi.org/10.1006/jcss.1997.1504

[10] Çaglar Gülçehre, Marcin Moczulski, Francesco Visin, and Yoshua Bengio. 2016. Mollifying Networks. *CoRR* abs/1608.04980 (2016). arXiv:1608.04980 http://arxiv.org/abs/1608.04980

[11] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. 2016. Deep Networks with Stochastic Depth. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*. 646–661. https://doi.org/10.1007/978-3-319-46493-0_39

[12] Prateek Jain, Brian Kulis, Inderjit S. Dhillon, and Kristen Grauman. 2008. Online Metric Learning and Fast Similarity Search. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*. 761–768. http://papers.nips.cc/paper/3446-online-metric-learning-and-fast-similarity-search

[13] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. 2011. Product Quantization for Nearest Neighbor Search. *IEEE Trans. Pattern Anal. Mach. Intell.* 33, 1 (2011), 117–128. https://doi.org/10.1109/TPAMI.2010.57

[14] Rong Jin, Shijun Wang, and Yang Zhou. 2009. Regularized Distance Metric Learning: Theory and Algorithm. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*. 862–870. http://papers.nips.cc/paper/3703-regularized-distance-metric-learningtheory-and-algorithm

[15] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 2013. 3D Object Representations for Fine-Grained Categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*. Sydney, Australia.

[16] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images*. Technical Report.

[17] Neeraj Kumar, Alexander C. Berg, Peter N. Belhumeur, and Shree K. Nayar. 2009. Attribute and simile classifiers for face verification. In *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*. 365–372. https://doi.org/10.1109/ICCV.2009.5459250

[18] Gary B. Huang Erik Learned-Miller. 2014. *Labeled Faces in the Wild: Updates and New Reporting Procedures*. Technical Report UM-CS-2014-003. University of Massachusetts, Amherst.

[19] Wenbin Li, Yang Gao, Lei Wang, Luping Zhou, Jing Huo, and Yinghuan Shi. 2018. OPML: A one-pass closed-form solution for online metric learning. *Pattern Recognition* 75 (2018), 302–314. https://doi.org/10.1016/j.patcog.2017.03.016

[20] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* 115, 3 (2015), 211–252. https://doi.org/10.1007/s11263-015-0816-y

[21] Doyen Sahoo, Quang Pham, Jing Lu, and Steven C. H. Hoi. 2018. Online Deep Learning: Learning Deep Neural Networks on the Fly. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. 2660–2666. https://doi.org/10.24963/ijcai.2018/369

[22] Shai Shalev-Shwartz, Yoram Singer, and Andrew Y. Ng. 2004. Online and batch learning of pseudo-metrics. In *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*. https://doi.org/10.1145/1015330.1015376

[23] Blake Shaw, Bert C. Huang, and Tony Jebara. 2011. Learning a Distance Metric from a Network. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain*. 1899–1907. http://papers.nips.cc/paper/4392-learning-a-distance-metric-from-a-network

[24] Suraj Srinivas and Radhakrishnan Venkatesh Babu. 2016. Learning Neural Network Architectures using Backpropagation. In *Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016*. http://www.bmva.org/bmvc/2016/papers/paper104/index.html

[25] Kilian Q. Weinberger, John Blitzer, and Lawrence K. Saul. 2005. Distance Metric Learning for Large Margin Nearest Neighbor Classification. In *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*. 1473–1480. http://papers.nips.cc/paper/2795-distance-metric-learning-for-large-margin-nearest-neighbor-classification

[26] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. arXiv:cs.LG/cs.LG/1708.07747

Knowledge-aware Assessment of Severity of Suicide Risk for Early Intervention

Manas Gaur
Knoesis Center
Dayton, Ohio
manas@knoesis.org

Amanuel Alambo
Knoesis Center
Dayton, Ohio
amanuel@knoesis.org

Joy Prakash Sain
Knoesis Center
Dayton, Ohio
joy@knoesis.org

Ugur Kursuncu
Knoesis Center
Dayton, Ohio
ugur@knoesis.org

Krishnaprasad Thirunarayan
Knoesis Center
Dayton, Ohio
tkprasad@knoesis.org

Ramakanth Kavuluru
University of Kentucky
Lexington, Kentucky
ramakanth.kavuluru@uky.edu

Amit Sheth
Knoesis Center
Dayton, Ohio
amit@knoesis.org

Randon S. Welton
Department of Psychiatry
Dayton, Ohio
randon.welton@wright.edu

Jyotishman Pathak
Cornell University
New York, NY
jyp2001@med.cornell.edu

ABSTRACT

Mental health illness such as depression is a significant risk factor for suicide ideation, behaviors, and attempts. A report by Substance Abuse and Mental Health Services Administration (SAMHSA) shows that 80% of the patients suffering from Borderline Personality Disorder (BPD) have suicidal behavior, 5-10% of whom commit suicide. While multiple initiatives have been developed and implemented for suicide prevention, a key challenge has been the social stigma associated with mental disorders, which deters patients from seeking help or sharing their experiences directly with others including clinicians. This is particularly true for teenagers and younger adults where suicide is the second highest cause of death in the US. Prior research involving surveys and questionnaires (e.g. PHQ-9) for suicide risk prediction failed to provide a quantitative assessment of risk that informed timely clinical decision-making for intervention. Our interdisciplinary study concerns the use of Reddit as an unobtrusive data source for gleaning information about suicidal tendencies and other related mental health conditions afflicting depressed users. We provide details of our learning framework that incorporates domain-specific knowledge to predict the severity of suicide risk for an individual. Our approach involves developing a suicide risk severity lexicon using medical knowledge bases and suicide ontology to detect cues relevant to suicidal thoughts and actions. We also use language modeling, medical entity recognition and normalization and negation detection to create a dataset of 2181 redditors that have discussed or implied suicidal ideation, behavior, or attempt. Given the importance of clinical knowledge, our gold standard dataset of 500 redditors (out of 2181) was developed by four practicing psychiatrists following the guidelines outlined in Columbia Suicide Severity Rating Scale (C-SSRS), with the pairwise annotator agreement of 0.79 and group-wise agreement of 0.73. Compared to the existing four-label classification scheme (no risk, low risk, moderate risk, and high risk), our proposed C-SSRS-based 5-label classification scheme distinguishes people who are supportive, from those who show different severity of suicidal tendency. Our 5-label classification scheme outperforms the state-of-the-art schemes by improving the graded recall by 4.2% and reducing the perceived risk measure by 12.5%. Convolutional neural network (CNN) provided the best performance in our scheme due to the discriminative features and use of domain-specific knowledge resources, in comparison to SVM-L that has been used in the state-of-the-art tools over similar dataset.

CCS CONCEPTS

• **Human-centered computing** → *HCI design and evaluation methods*; • **Computing methodologies** → **Natural language processing**; **Machine learning**; • **Applied computing** → **Health informatics**.

KEYWORDS

Surveillance and Behavior Monitoring; Reddit; Mental Health; Suicide Risk Assessment; C-SSRS; Medical Knowledge Bases; Perceived Risk Measure; Semantic Social Computing

ACM Reference Format:
Manas Gaur, Amanuel Alambo, Joy Prakash Sain, Ugur Kursuncu, Krishnaprasad Thirunarayan, Ramakanth Kavuluru, Amit Sheth, Randon S. Welton, and Jyotishman Pathak. 2019. Knowledge-aware Assessment of Severity of Suicide Risk for Early Intervention . In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313698

1 INTRODUCTION

According to recent data from the US Centers for Disease Control and Prevention (CDC), suicide is the second leading cause of death for people aged between 10-34 [45] and fourth leading cause for people aged 35-64, escalating the suicide rate in the US by 30% since

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313698

1999[1]. Suicide Prevention Resource Center in the US[2] reports that 45% of people who committed suicide had visited a primary care provider one to two months before their death. These visits were often scheduled for something other than complaints of depression or suicide, suicidal patients may be too embarrassed to bring up suicide. Clinicians often have no prior warning that the patient is currently suicidal or will be developing significant signs of suicidality. Hence, novel strategies are necessary to proactively detect, assess, and enable timely intervention to prevent suicide[3].

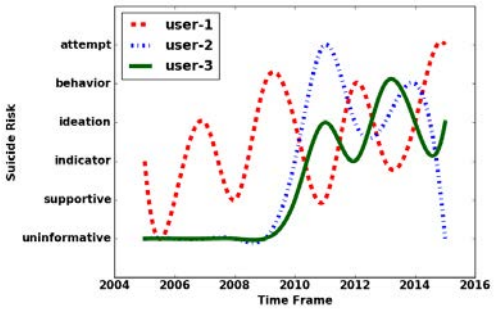


Figure 1: Changing Suicide Risk of 3 Redditors over a period of 11 years

Mental health conditions have been closely linked to suicide [17]. Depression, bipolar and other mood disorders are known to be the main risk factors for suicide, while substance abuse and addiction have been closely linked to suicidal thoughts[4]. SAMHSA[5] reports that people with BPD, Alcoholism, and Drug Addiction are more prone to having suicidal behaviors (e.g., holding gun to the head, driving sharp knife through nerves) and committing suicide. Apart from mental health conditions, there are various other factors exacerbating an individual's urge to commit suicide such as workplace/sexual harassment, religious scripts encouraging self-sacrifice, and heroic portrayal of death in movies. Moreover, popular celebrities who commit suicide can lead to "copycat" suicides or Werther effect [39]. It refers to the contagious influence that a popular figure's suicide can have on an individual, encouraging them to commit suicide. There are several resources for patients to seek help from such as CrisisTextLine, teen line, 7cups.com, imalive.org, and The Trevor Project for LGBTQ. Additional measures are necessary to improve timely intervention [5]. Unobtrusive collection and analysis of social media data can provide a means for gathering insights about an individual's emotions, and suicidal ideation and behavior [33]. A system capable of gleaning digital markers of suicide risk assessment from social media conversations of a patient (see Figure 1) can help a mental health professional (MHP) for making informed decisions as the patients may be reluctant to directly share all the relevant information due to the social stigma associated with mental illness and suicide[22].

There is a significant body of work addressing issues concerning suicide and mental health using social media content. TeenLine, Tumblr, Instagram, Twitter, and Reddit have been common sources of data for research in computational social science [7, 8, 56]. Among these, Reddit has emerged as the most promising one due to the anonymity it affords, its popularity as measured by its content size, and its variety as evident from the diverse subreddits being used for posting that reflects a user's state of mind and mental health disorder, e.g., r/Depression, r/SuicideWatch, r/BipolarSoS. Analysis of the content on Reddit can be leveraged to help an MHP develop an insight into the current situation of an individual, to improve the quality of the diagnosis and intervention strategies if necessary. Shing et al.[54] analyzed the postings of users in SuicideWatch and other related subreddits (e.g., r/bipolarreddit, r/EatingDisorder, r/getting_over_it, and r/socialanxiety) for assessment of suicide risk. The critical opportunity to improve upon these efforts is to utilize reliable domain-specific knowledge sources for understanding the content from a clinical perspective. Specifically, this strategy can augment raw Reddit content to normalize it into a standard medical context and improve the decision-making process of the MHP.

Prior research on suicide risk assessment employs four-label (no risk, low risk, moderate risk, and high risk) classification scheme for categorization of suicidal users [54]. In this research, we provide a C-SSRS-based five-label (supportive, indicator, ideation, behavior, and attempt) classification scheme guided by clinical psychiatrists, which allows the MHP to determine an actionable measure of an individual's suicidality and appropriate care [65]. We compared our 5-label scheme with two other variants: 4-label (indicator, ideation, behavior, and attempt) and (3+1)-label (supportive + indicator, ideation, behavior, and attempt) for monitoring progression and for alerting an MHP as necessary.

Apart from identifying the risk factors of suicide, we can develop approaches to generate answers to the questions from the content in C-SSRS[6], such as (1) Have you wished you were dead or wished you could go to sleep and not wake up? and (2) Have you actually had any thoughts of killing yourself? Our study aims to develop mapping and learning approaches for estimating the suicide risk severity level of an individual, based on his/her posted content [1].

Key Contributions: (1) We develop an annotated gold standard dataset of 500 Reddit users, out of 2181 potentially suicidal users, using their content from mental health-related subreddits. (2) Using domain-specific resources- SNOMED-CT, DataMed, Drug Abuse Ontology (which incorporates DSM-5 [60]) and ICD-10, we created *suicide risk severity lexicon*, curated by MHPs. This enabled us to create a competitive baseline for evaluating our approach. (3) Using four evaluation metrics (graded recall, confusion matrix, ordinal error, and perceived risk measure), we show that the C-SSRS based 5-label classification scheme improves upon the state-of-the-art scheme to characterize suicidality of a user. (4) Our evaluation shows that CNN emerges as a superior model for suicide risk prediction task outperforming the two competing baselines: rule-based and SVM-linear. Technological advancements over the last decade have transformed the health care system with a trend towards real-time monitoring, personal data analysis, and evidence-based diagnosis. Specifically, with the anticipated inclusion of individual's social data and the rapidly growing patient-generated health data [52], MHPs will be better informed about the patient's conditions including their suicidality to enable timely intervention.

[1] https://www.cdc.gov/mmwr/preview/mmwrhtml/mm6217a1.htm
[2] https://www.integration.samhsa.gov/about-us/esolutions-newsletter/suicide-prevention-in-primary-care
[3] https://bit.ly/2QiYqbo
[4] https://www.psychologytoday.com/us/blog/real-healing/201402/suicide-one-addiction-s-hidden-risks
[5] https://www.samhsa.gov/suicide-prevention
[6] https://www.integration.samhsa.gov/clinical-practice/screening-tools#suicide

In Section 2, we review related research. In Section 3, we discuss the resources we use. In Section 4, the critical components of the approach are developed. In Section 5 we give details of experimental design and in Section 6 we discuss our results.

2 RELATED WORK

In this section, we describe prior research related to our study.

2.1 Suicide and Social Media

Jashinsky et al. [28], and Christensen et al. [9] predicted the level of suicide risk for an individual over a period of time using Support Vector Machines (SVM) and the features of Term Frequency-Inverse Document Frequency (TF-IDF), word count, unique word count, average word count per tweet, and average character count per tweet. De Choudhury et al. [17] identified linguistic, lexical, and network features that describe a patient suffering from a mental health condition for predicting suicidal ideation. Analysis of content that contains self-reporting posts on Reddit can provide insights on mental health conditions of users. Utilizing propensity score matching, [17] measured the likelihood of a user sharing thoughts on suicide in the future. Another study from Sueki [58] investigated the linguistic variations among different authors on social media, and observed correlation between suicidal behavior and suicide-related tweets. Furthermore, Cavazos-Rehg et al. [8] performed a qualitative analysis of user's content on Tumblr to better understand discourse of self-harm, suicide, and depression. The study highlights Tumblr as a platform for development of suicide prevention efforts through early intervention. Further, people on social media with mental health conditions, often look for similar people [48].

2.2 Analysis of Suicidal Risk Severity

So far, prior research studied the identification of signals for predicting the suicide risk, mental health conditions leading to suicide [15, 16, 47], psychological state and well-being [50, 51]. Nock et al. [40] reported that ∼9% of people have thoughts of suicide, ∼3% map out their suicidal plans, ∼3% make a suicide attempt and ≤1% people constitute what are known as "suicidal completers". Much information extracted from the content of an individual provide explicit, implicit or ambivalent clues for suicide. These clues can help an MHP assess suicide severity, and better structure the treatment process [11].

Shing et al.[54] used 1.5M posts from 11K users on SuicideWatch subreddit. In the study, experts and crowd-source workers annotated the posts from 245 users using labels defined in [12]. The study evaluates the annotation quality of experts and non-experts and performs risk and suicide screening experiments using linguistic and psycho-linguistic features based on machine/deep learning classifiers. The study fails to bring together different mental health conditions that lead to suicide. Inclusion of supportive users on social media, who are not suicidal, as these constitute the negative samples. Further, the rubric for annotating the dataset was not authoritative, whereas, we utilize C-SSRS endorsed by NIH and SAMHSA.

2.3 Models for Suicide Prediction

In a recent study on predicting suicide attempt in adolescents, Bhat et al. employed deep neural networks for predicting the presence of suicide attempts using >500K anonymized Electronic Health Records (EHR) obtained from California Office of Statewide Health Planning and Development (OSHPD). Through a series of experiments, researchers achieved a true positive rate of 70% and a true negative rate of 98.2% [4]. Another study by Walsh et al. [61] on predicting suicidal attempts using temporal analysis, employs Random Forest (RF) over a cohort of 5167 patients. The study segregates the cohort of patients into 3250 cases and 1917 controls. They achieved an F1-score of 86% with a recall of 95% [61]. The study used binary classification scheme for Electronic Health Records (EHR) dataset, which is not suitable for identifying supportive and indicator users. A transfer learning from social media to EHR can improve its effectiveness [62]. Amini et al. utilized SVM, and decision trees besides RF and Neural Networks (NN), for assessing the risk of suicide in a dataset of individuals from Iran [2]. A recent study by Du et al. [19] used deep learning methods to detect psychiatric stressors leading to suicide. They built binary classifier for identifying suicidal tweets from non-suicidal tweets using Convolutional Neural Networks (CNN). Once suicidal tweets are detected,they performed Named Entity Recognition (NER) using Recurrent Neural Networks (RNN) for tagging psychiatric stressors in a tweet classified as suicidal.

3 BACKGROUND STUDY

We detail the medical knowledge bases underlying the suicide risk severity lexicon used in a baseline (see Section 5.2).

3.1 Domain-specific Knowledge Sources

Medical knowledge bases are resources manually curated by domain experts providing concepts and their relationships for processing the content. As our study aims to assess the severity of at-risk suicidal users, the domain knowledge that corresponds to different levels of suicidality of a patient is crucial. In this work, we employ ICD-10, SNOMED-CT, Suicide Ontology, and Drug Abuse Ontology (DAO) [7] for creating a suicide lexicon to be used in one of our baselines.

Concepts in **SNOMED-CT** are categorized into procedure, observable entity, situation, event, assessment scale, therapy, disorder, and finding and can be extracted using "parents", "children", and "sibling" relationships. For example, Suicide by Hanging [SNOMED ID: 287190007] is a child concept of Suicide [SNOMED ID: 44301001] and sibling concept of Assisted Suicide [SNOMED ID: 51709005], Drug Overdose - Suicide [SNOMED ID: 274228002], Suicide while incarcerated [SNOMED ID: 23546003], and Suicide by self-administered Drug [SNOMED ID: 891003]. **ICD-10** is a medical standard that provides information on patient's health state such as severity, complexity, comorbidities, and complications. Concepts in ICD-10 are categorized into signs, symptoms, abnormal findings, and diagnosis. For example, "suicide attempt" is categorized under a Personal history of self-harm [ICD-10 ID: Z91.5]. It is also categorized under Borderline Personality Disorder [ICD-10 ID: F60.3], Intentional self-harm [ICD-10 ID: X60-X84], and Severe depressive episode with psychotic symptoms [ICD-10 ID: F32.3]. "suicidal ideation" is categorized under Post-traumatic stress disorder[ICD-10 ID: F43.1]. **Suicide Ontology** is an ontology, called "suicideonto"[7] built through text mining and manual curation by domain experts. The ontology contains 290 concepts defining the context of suicide. **Drug Abuse Ontology (DAO)** is a domain-specific hierarchical framework developed by Cameron et al. [7] containing 315 entities (814

[7] https://bioportal.bioontology.org/ontologies/suicideo

instances) and 31 relations defining drug-abuse and mental-health concepts. The ontology has been utilized in analyzing web-forum content related to buprenorphine, cannabis, a synthetic cannabinoid, and opioid-related data [13, 14, 34]. In [21] it was expanded using DSM-5 categories covering mental health and applied for improving mental health classification on Reddit.

3.2 Existing Domain Specific Lexicons

Prior research [6, 38] highlighted the disparity between the informal language used by social media users and the concepts defined by domain experts in medical knowledge bases. Medical entity normalization fills such a gap by identifying phrases (n-grams, or topics) within the content and mapping them to concepts in medical knowledge bases [36]. We use (i) two lexicons, namely, TwADR-L and AskaPatient (see Table 1) to map the social media content to medical concepts [36], and (ii) anonymized and annotated suicide notes made available through Informatics for Integrating Biology and the Bedside (i2b2) challenge to identify content with negative emotions (see Table 2).

Table 1: Existing domain specific lexicons used in this study

Lexicon	#SNOMED Concepts	Max. #phrases per Concept	Sample SNOMED to informal terms mapping
TwADR-L	2172	36	**SNOMED Concept:** Acute depression **Phrases:** 'acute depression', 'just want to finally be happy', 'hated my life', 'depression'
AskaPatient	3051	56	**SNOMED Concept:** Anxiety **Phrases:** 'anxious', 'anxiety issues', 'anxiety', 'anzity'

TwADR-L [36] maps medical concepts in SIDER[8] to their corresponding informal terms used in Twitter. The lexicon has 2172 medical concepts, each of which has up to 36 informal Twitter terms. Each informal term is assigned a single medical concept. AskaPatient[9] [36] maps informal terms from AskaPatient web forum to medical concepts in SNOMED-CT and Australian Medical Terminology [35]. Since this lexicon was created from a web forum, it is more informative compared to TwADR-L. i2b2 Suicide Notes is a dataset generated as a part of the emotion recognition task in 2011 [63]. We have ~2K suicide notes annotated for different emotions, and of them with negative emotions were removed, resulting in 817 suicide notes (see Table 2 for examples).

Table 2: Suicide notes aggregated by emotion labels defined in i2b2.

Emotion Label	#Suicide Notes	Example
abuse	9	My son got married to a horrible woman who does not care curses swears and pushes me around.
anger	69	I have no idea why I could let one person hurt me I loved you for so long but I think I hate you now.
fear	25	In this case you would finally meet defeat so crushing will drain strip you off your courage and hope
guilt	208	God is just and it is true that I am a no good but God will see all that I had to pass through
hopelessness	455	Dear Jane Dont think to badly of me for taking this way out but I am frustrated by taking so much pain
sorrow	51	My heart has been hurt hard and grieving.

[8]http://sideeffects.embl.de
[9]https://www.askapatient.com/

3.3 Suicide Risk Severity Lexicon

Besides the existing lexicons (see Section 3.2), we have built a comprehensive lexicon containing terms related to each level of suicide risk severity (see Table 3). The lexicon was created using Suicideonto[10], DSM-5 [21], and concepts in i2b2 suicide notes. Besides these four severity levels, we consider a separate class of "supportive" users who are not suicidal, but use a similar language. The lexicon was created using the aforementioned medical knowledge bases and slang terms from DAO. The lexicon was validated by the domain experts, and used for annotation and for our baseline (see Section 5.2).

Table 3: Suicide Risk Severity lexicon

Suicide Class	# Terms in a class	Examples
Indicator	1535	Pessimistic character, Suicide of relative, Family history of suicide
Ideation	472	Suicidal thoughts, Feeling suicidal, Potential suicide care
Behavior	146	Planning on cutting nerve, Threatening suicide, Loaded Gun, Drug-abuse
Attempt	124	Previous known suicide attempt, Suicidal deliberate poisoning, Goodbye Attempted suicide by self-administered drug, Suicide while incarcerated.

3.4 Columbia Suicide Severity Rating Scale

Each C-SSRS severity class (ideation, behavior, or attempt) is composed of a set of questions that characterize the respective category. Responses to the questions across the C-SSRS classes eventually determine the risk of suicidality of an individual [44]. One of the challenges researchers face when it comes to dealing with social media content is the disparity in the level of emotions expressed. Since the C-SSRS was originally designed for use in clinical settings, adapting the same metric to a social media platform would require changes to address the varying nature of emotions expressed. For instance, while in a clinical setting, it is typically suicidal candidates that see a clinician; on social media, non-suicidal users may participate to offer support to others deemed suicidal. To address these factors, we have defined two additional classes to the existing C-SSRS scale with three classes. We have provided the description of the five classes in Section 4.4.1.

3.5 Suicide Seed Terms

Not all users in subreddit SuicideWatch (SW) are suicidal. We identify suicidal candidates in subreddit SW by looking into the nature of words used in users' posts.

We analyzed the content of SW subreddit against Zipf-Mandelbrot law to precisely identify terms that are 'prominent' in the online discussion of suicidal thoughts balancing frequency and relevance. In Figure 2, the cyan line follows Zipf-distribution while the green line follows the

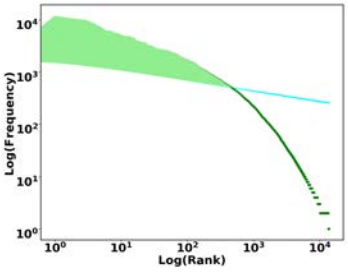


Figure 2: Zipf-Mandelbrot law over SW content for identifying prominent suicide seed terms. Highlighted is the selected region.

[10]https://bioportal.bioontology.org/ontologies/suicideo

Mandelbrot distribution. We are particularly interested in the region of the graph shaded in the top left corner off the cut-off mark between the two lines (light green).

Table 4: Suicide seed terms selected through Figure 2. W: Word, Ph:Phrase, SW: SuicideWatch.

W/Ph.	Freq. W/Ph in SW	W/Ph.	Freq. W/Ph in SW
Commit Suicide	539	Death	215
Hopelessness	130	Gun	577
Sadness	453	Isolation	104
Therapist	194	Trans[11]	96
Kill	577	Sleep	193

This region represents terms in the document that are frequently used by users while also having higher ranks (numerically small values). This effectively eliminates terms that are simply frequently used in the document, but have low ranks. Identified terms were validated by clinical psychiatrists and a curated list of 339 words with a cut-off frequency of 725. A sample list of 10 words is shown in Table 4.

Having identified the suicidally prominent terms, and in conjunction with negation detection technique, we filtered noisy users (users who don't 'positively' use one or more of these terms in their posts) and identified prominently suicidal users.

3.6 Embedding Models

Word embeddings are a set of techniques used to transform a word into a real-valued vector. This allows words with similar meanings to have similar representations and be clustered together in the vector space. Normally, we either generate domain-specific word embeddings local to our problem or employ general purpose word embeddings [32]. We utilize embeddings from ConceptNet[12] (vocabulary= 417193, dimension= 300), a multi-lingual knowledge graph created from expert sources, crowd-sourcing, DBpedia, vocabulary derived from Word2Vec[13] [49], and Glove[14] [43] [57].

4 DATASET CREATION AND ANALYSIS

In this section, we analyze the data, its features and our procedure to identify a small cohort of Redditors that resemble potential candidates for suicidal users (see Figure 3). Our dataset comprises 270,000 users with 8 Million posts from 15 mental health related subreddits; r/StopSelfHarm (SSH), r/selfharm (SLF), bipolar (r/bipolar (BPL), r/BipolarReddit (BPR), r/BipolarSOs, r/opiates (OPT), r/Anxiety (ANX), r/addiction (ADD), r/BPD, r/SuicideWatch (SW), r/schizophrenia (SCZ), r/autism (AUT), r/depression (DPR), r/cripplingalcoholism (CRP), and r/aspergers (ASP) [21]. We used 93K users who actively participated in the SuicideWatch subreddit providing 587466 posts. To further enrich our dataset, we gathered the posts of these users in the remaining 14 subreddits. The timeframe of our dataset is between 2005 and 2016.

4.1 Potential Suicidal Redditors

Subreddit "SuicideWatch" (SW) had nearly 93K redditors as of 2016. To create a representative sample dataset containing users at five-levels of suicide risk, we used seed terms generated using

[11]https://bit.ly/2NEK9bc
[12]http://conceptnet.io
[13]https://code.google.com/archive/p/word2vec/
[14]https://nlp.stanford.edu/projects/glove/

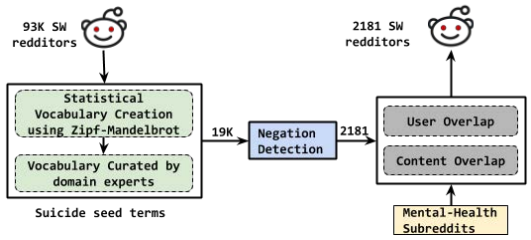


Figure 3: Procedure for generating the annotated dataset. Only 9 mental health subreddits were considered because of significant content overlap (see Figure 4)

Zipf-Mandelbrot (see Section 3.5). We obtained a working set of 19K redditors using such terms. Next, we employed negation detection procedure (see Section 4.2) to eliminate non-suicidal users. Finally, we obtained 2181 users who are potentially suicidal and had participated in other mental health subreddits. For referencing, we denote these users and their content in SW as U^{SW}.

4.2 Negation Detection

Negation detection is a crucial part as the presence of negated sentences can confound a classifier [23]. For example, *I am not going to end my life because I failed a stupid test* is not suicidal, whereas *My daily struggles with depression have driven me to alcohol* reflects user's mental health. The former sentence can give false positive, if we just extract 'going to end my life' as a precursor to a suicide attempt. We employ a negation detection tool and probabilistic context-free grammar that supports negation extraction and negation resolution to improve classifier performance [23].

4.3 User and Content Overlap

As individuals form communities based on shared topics of interest related to mental health conditions [59, 64] in different subreddits, we performed user and content overlap analysis between SW and other mental health subreddits to enrich the contents of users. This analysis provides deeper insight into how potentially suicidal users communicate on problems including causes, symptoms, and treatment solutions. Through user overlap we infer the population level similarity between a mental health subreddit and SW, whereas using content we quantify overlap in context for each user. We calculated the user overlap through the intersection of the users in U^{SW} and i^{th} mental health subreddit (U^{MH_i}). Content overlap was calculated using a cosine similarity measure through domain-specific lexicon, LDA2Vec [37] and ConceptNet.

We leverage the quantified similarity of suicide-related topics between content of the users in U^{SW} and other subreddits (U^{MH_i}), to append the content of users U^{SW}. This procedure will contribute to the holistic nature of the content and enable more discriminative features in the classifier. For example, a post in SW: *I dont think Ive thought about it every day of my entire life. I have for a good portion of it, however, my boyfriend may be able to determine whether I'm worth his time* seems to imply that the user is *non-suicidal*. However, after appending following post taken from "depression" subreddit: *Having a plan for my own suicide has been a long time relief for me as well. I more often than not wish I were dead*, we notice that the user has *suicidal ideations*. As the content in Reddit posts contain slang terms for medical entity, we employed a normalization procedure using standardized lexicons to provide a cleaner interpretation of a

patient's condition, meaningful to a mental health professional or clinician. To perform medical entity normalization, we utilize three lexicons (see Section 3.2), namely, i2b2, TwADR, and AskaPatient, which were created from Medical Records, Twitter, and Web Forum respectively. The normalization used string match.

Content overlap using TwADR-L and AskaPatient: We trained an LDA model with topic coherence over the normalized content to find coherent topics for SW subreddit. Subsequently, using the trained LDA model of SW content, we generated two sets of Topics at user level for U^{SW}, and U^{MH_i}. The topical similarity (TS) was calculated between topics of U^{SW}, and those U^{MH_i}. For the calculation of TS, the user should be present in U^{SW} and U^{MH_i} and should have an average similarity greater than 0.6 (defined empirically). We formalized TS as;

If $u \in U^{SW} \cap U^{MH_i}$, then

$$\left(\frac{1}{|U^{SW}|-1}\sum_{v \in U^{SW} \setminus \{u\}} \cos(\vec{u}^{MH_i}, \vec{v}^{SW})\right) > 0.6 \quad (1)$$

In the above equation, topic vector of users in U^{MH_i} is denoted as $\vec{u}^{MH_i}$ and that of U^{SW} as $\vec{v}^{SW}$.

The resultant column vector contains the similarity between MH_i and SW and has a dimension of **14x1**. Equation 1 used with for two lexicons: TwADR-L and AskaPatient for abstracting the concepts within the reddit posts. To create each column vector, we trained two topic models because TwADR-L lexicon has been created using Twitter and AskaPatient Lexicon using Forum content.

Content Overlap using i2b2: Table 2 shows 6 emotion labels in i2b2 suicide notes dataset. For quantifying the user's content with appropriate emotion label (Table 2), we generated embeddings of content in SW and other MH subreddits for each user using ConceptNet embedding model. We also generated the representations of the emotion labels of the suicide notes through concatenation and dimensionality reduction of the embedding vectors of their corresponding suicide notes [20]. Then, we performed the cosine similarity measure over: (i) embeddings of content from mental health subreddits for each user and the emotion labels, and (ii) embeddings of content from the SW subreddit for each user and the emotion labels. We formalize similarity between i2b2 label and user content embedding as follows:

$$UL(SW, L) = cos(\vec{u}, \vec{l}), u \in U^{SW}, l \in L \quad (2)$$

where UL(SW,L) stores the similarity values between the users in U^{SW} and the emotions labels in i2b2 (L), forming a matrix of dimension 2181 x 6. It is calculated using cosine similarity between the vector of a user ($u \in U^{SW}$) and an emotion label ($l \in L$). Each row of the matrix represents the similarity value for a user embedding generated from all their posts against embedding of each label in i2b2 generated from suicide notes. A similar matrix (using Equation 2) is created for users in other mental health subreddits ($u \in U^{SW} \cap U^{MH_i}$) and emotion labels L. We denote such a matrix as $UL(MH_i, L)$ of dimensions 2181 x 6. $UL(SW, L)$ and $UL(MH_i, L)$ are interpreted as matrices showing to what degree users' contents are close to six emotions. Thereafter, we generate a similarity score ($SS(MH_i, SW)$) as a product of $UL(SW, L)$ and transpose of $UL(MH_i, L)$. Formally we define it as:

$$\text{SS}(MH_i, SW) = \frac{\sum_{\vec{u} \in UL(SW,L)} \sum_{\vec{v} \in UL(MH_i,L)^T} \vec{u}.\vec{v}}{(|U^{SW}|-1)^2} \quad (3)$$

If the users are in U^{SW} and U^{MH_i}, their content will be appended to SW from MH_i only if the content overlap is greater than 0.6 in Equations 1 and 3. The procedure repeated over all MH subreddits and we obtain results shown in Figure 4.

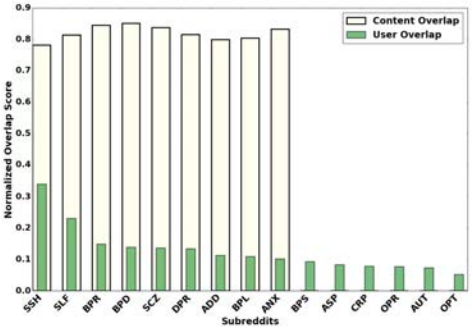


Figure 4: User Overlap and Content Overlap based quantification of influence of other mental health related subreddit to SW. Subreddits *SSH* and *SLF* have the highest content overlap with SW followed by *BPR* and *BPD*.

4.4 Gold Standard Dataset Creation

We describe different classes of suicidality, characterizing users who suffer from mental health conditions or involve themselves in a supportive role on social media. Further, we describe annotated dataset with examples and annotation evaluation using Krippendorff.

4.4.1 5-labels of Suicide Risk Severity: C-SSRS begins with ***Suicidal Ideation*** (*ID*), which is defined as thoughts of suicide including preoccupations with risk factors such as loss of job, loss of a strong relationship, chronic disease, mental illness, or substance abuse. This category can be seen to escalate to ***Suicidal Behavior*** (*BR*), operationalized as actions with higher risk. A user with suicidal behavior confesses active or historical self-harm, or active planning to commit suicide, or a history of being institutionalized for mental health. Actions include cutting or using blunt force violence (self-punching and head strikes), heavy substance abuse, planning for suicide attempt, or actions involving a means of death (holding guns or knives, standing on ledges, musing over pills or poison, or driving recklessly). The last category, an ***Actual Attempt*** (*AT*), is defined as any deliberate action that may result in intentional death, be it a completed attempt or not, including but not limited to attempts where a user called for help, changed their mind or wrote a public "good bye" note. When reviewing users' risk levels for social media adaptation, two additional categories were added to define user behaviors less severe than the above categories.

The first addition was a ***Suicide Indicator*** (*IN*) category which separated those using at-risk language from those actively experiencing general or acute symptoms. Oftentimes, users would engage in conversation in a supportive manner and share personal history while using at-risk words from the clinical lexicon. These users might express a history of divorce, chronic illness, death in the family, or suicide of a loved one, which are risk indicators on the C-SSRS, but would do so relating in empathy to users who expressed ideation or behavior, rather than expressing a personal desire for self-harm. In this case, it was deemed appropriate to flag such users as *IN* because while they expressed known risk factors that could be monitored they would also count as false positives if they were accepted as individuals experiencing active ideation or behavior.

The second additional category was named as ***Supportive*** (*SU*) and is defined as individuals engaging in discussion but with no language that expressed any history of being at-risk in the past

or the present. Some identified themselves as having background in mental health care, while others did not define their motive for interacting at all (as opposed to a family history). Since posting on Reddit is not itself a risk factor, so we give these users a category with even lower risk than those expressing support with a history of risk factors. Any use of language such as a history of depression, or "I've been there" would re-categorize a user as exhibiting suicidal indicator, ideation, or being at greater risk, depending on the language used. These new categories for an adapted C-SSRS should help account for those who communicate in suicide-related forums but were at a low or undefined risk.

4.4.2 Description of the Annotated Dataset: For the purpose of annotation, we randomly picked 500 users from a set of 2181 potential suicidal users. In the annotated data, each user on an average has 31.5 posts within the time frame of 2005 to 2016.

The annotated data comprises of 22% supportive users, 20% users with some suicidal indication but cannot be classified as suicidal, 34% users with suicidal ideation, 15% users with suicidal behaviors, and 9% users have made an attempt (success or fail) to commit suicide. Supportive users constitutes 1/5th of the total data size and prior studies have ignored them. Table 5 shows posts from redditors and their associated suicide risk

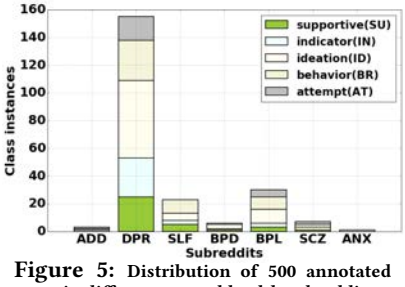


Figure 5: Distribution of 500 annotated users in different mental health subreddits

Table 5: Paraphrased posts from candidate suicidal redditors and associated suicide risk severity level

Always time for you to write your happy ending .... doesnt need to be spelled out with alcohol and Xanax.... keep an open mind	SU
Ive never really had a regular sleep schedule....no energy to hold a conversation....no focus on study....barely eat and sleep....fluffy puppy dog face	IN
Sometimes I literally cant bear to move....my depression....since I was 14....suffering rest of my life....only Death is reserved for me.	ID
Driving a sharp thing over my nerve. Extreme depression and loneliness.... worthless excuse for a life....used everything from wiring to knife blades	BR
I am going to off myself today...loaded gun to my head..determined....huge disappointment....screwed family life....breaks my heart everyday.	AT

severity level. To identify which mental health subreddits (except SW) contributed most to suicidality, we mapped potential suicidal Redditors to their subreddits (see Figure 5).

4.4.3 Evaluation of Annotation: Four practicing clinical psychiatrists were involved in the annotation process. Each expert received 500 users dataset comprising of 15755 posts. We perform two annotation analysis defined for ordinal labels: **(1)** A pair-wise annotator agreement using Krippendorff metric (α) to identify the annotator with highest agreement with others, **(2)** An incremental group wise annotator agreement to find the robustness of the earlier annotator [55]. For group wise agreement, we denote a set of annotators as G with cardinality ($|G|$) range from 2 to 4. α is calculated as $1 - (\frac{D_o(A_j,S)}{D_e})$, where $D_o(A_j, S)$ is observed disagreement and D_e is expected disagreement. The pairwise annotator agreement is a subset of groupwise and we formally define it as:

Table 6: (left). Pairwise annotator agreement, (right). Group wise annotator agreement. A,B,C,and D are annotators.

	B	C	D
A	0.79	0.73	0.68
B	-	0.68	0.61
C	-	-	0.65

	B	B&C	B&C&D
A	0.79	0.70	0.69

$$D_o(A_j, S) = \frac{1}{N.|S|} \sum_{i=1}^{N} \sum_{m \in S} |A_j^i - S_m^i|^2, S \subset G \setminus \{A_j\} \quad (4)$$

$$D_e = \frac{2}{N \cdot |G|(|G|-1)} \sum_{i=1}^{N} \sum_{m,q \in G, m \neq q} |G_m^i - G_q^i|^2 \quad (5)$$

where A_j is the annotator having highest agreement in pairwise α. S is the subset of a group of annotators G that excludes A_j. G_m^i and G_q^i represents the two annotators m and q within the group G^i. i is the index over all the users in the dataset. Results of pairwise and group wise annotators agreement is in Table 6. We observe a substantial agreement between the annotators[15].

5 EXPERIMENTAL DESIGN

5.1 Characteristic Features

Prior research has shown the importance of psycholinguistics, lexical, syntactic, and emotion features in enhancing the efficacy of the classifier [24, 46]. We further improve our feature set with information provided by Reddit. In training our models we used *AFINN*[16], which is a list of words scored for sentiment, emotions, mood, feeling, or attitude. Posts on Reddit may have nearly equal number of upvotes and downvotes making them controversial. We computed *controversiality score (CScore)* as the ratio of the maximum value of the difference, between *upvotes* and *downvotes*, and 1, over *totalvotes*.

$$\text{CSscore} = \max\left(1, \frac{\#upvotes - \#downvotes}{\#totalvotes}\right)$$

We factored in *Intra-Subreddit Similarity* with and without nouns and pronouns as a measure of content similarity of posts between a user and others in a subreddit. To determine the level of personal experience in the social media text, we utilize *First Person Pronouns Ratio* that measures the extent to which a Redditor talks about his/her own experience compared to other Redditors' experience [10]. We used *Language Assessment by Mechanical Turk (LabMT)*, a list of 10,222 words with happiness, rank, internet usage scores, employing strict match and soft match with Reddit posts [18]. On social media, readability is an important factor. We use *height of the dependency parse tree* to measure readability, with parse tree height being proportional to readability [25]. We employ *maximum length of verb phrase* [26] to capture suicidality of individuals. Similarly, *number of pronouns* was used to determine whether they are sharing a direct experience or second hand experience [42]. The value of this feature was high for users classified as supportive or indicator, as these users usually help others. Moreover, *number of sentences* and *number of definite articles* are also discriminative [27].

[15] http://homepages.inf.ed.ac.uk/jeanc/maptask-coding-html/node23.html
[16] http://neuro.imm.dtu.dk/wiki/AFINN

5.2 Baselines

In this study, we use two baselines; (1) 4-class scheme for predicting the suicide risk [54] (2) an empirical baseline based on the suicide risk severity lexicon. We provide details of our lexicon-based empirical baseline. Suicide lexicon developed as a part of the study for initial filtering of users and annotation process is a suitable resource for a baseline. This baseline is a rule-based model for classifying a user based on a strict and soft match criteria according to presence of a concept in the user's content and the suicide risk severity lexicon. For a competitive baseline, we compared this baseline with word-embedding and TF-IDF based approaches for suicide classification [29]. As we also experimented with word-embedding models trained over suicide and non-suicide related content, using compositions of word vectors [3, 32, 41], the baseline based on suicide risk severity lexicon outperformed these competitive approaches.

5.3 Convolutional Neural Network

We have implemented a convolutional neural network (CNN) as proposed in [30] for our contextual classification task [53].

The model takes embeddings of user posts as input and classifies into one of the suicide risk severity levels. We combine embeddings of posts for each user through concatenation, and pass into the model.

$$posts_u = post_{u,1} \bigoplus post_{u,2} \bigoplus ..post_{u,p}.. \bigoplus post_{u,P} \quad (6)$$

$$post_{u,p} = \vec{v}_{u,p,1} \bigoplus \vec{v}_{u,p,2} \bigoplus ..\vec{v}_{u,p,w}.. \bigoplus \vec{v}_{u,p,W} \quad (7)$$

Here $\bigoplus$ represents the concatenation operation of P posts of user u, where each post p of user u ($post_{u,p}$) is the concatenation of vectors of each word w ($\vec{v}_{u,p,w}$) where W is the total number of words in a post. Embeddings of the posts for each user ($posts_u$) have variable length. Hence, we use minimum length padding to make the dimensions of the representations uniform. The model has a convolution layer with filter window {3, 4, 5} and 100 filters for each. After getting the convoluted features, we apply max-pooling and concatenate the representative pooled features. We pass the pooled features through a dropout layer with dropout probability of 0.3, followed by an output softmax layer. The learning rate was set to 0.001 with adam optimizer [31]. While training the model, we have used mini batch of size 4 and trained for 50 epochs. CNN's performance is compared and evaluated in Section 6.

5.4 Evaluation Metrics

We alter the formulation of False Positive (FP) and False Negative (FN) to better evaluate the model performance. FP is defined as the ratio of the number of times the predicted suicide risk severity level (r') is greater than actual level (r^o) over the size of test data (N_T). FN is defined as the ratio of the number of times r' is less than r^o over N_T. Since the numerators of FP and FN involves comparison between r' and r^o suicide risk severity levels, we termed the metrics as graded precision, and recall as graded recall. Ordinal Error (OE) is defined as the ratio of the number of samples where difference between r^o and r' is greater than 1. In our study it represents the model's tendency to label a person as having no-severity or low degree of severity, when he/she is actually at risk.

We formally define FP, FN, and OE as:

$$FP = \frac{\sum_{i=1}^{N_T} I(r'_i > r^o_i)}{N_T},\ FN = \frac{\sum_{i=1}^{N_T} I(r^o_i > r'_i)}{N_T},\ OE = \frac{\sum_{i=1}^{N_T} I(\Delta(r^o_i, r'_i) > 1)}{N_T}$$

where $\Delta(r^o_i, r'_i)$ is the difference between r^o_i and r'_i. r'_i and r^o_i are the predicted and actual response for i^{th} test sample.

5.4.1 Perceived Risk Measure (PRM): It is defined to better characterize the difficulty in classifying a data item while developing a robust classifier in the face of *difficult to unambiguously* annotate datasets. It captures the intuition that if a data item is difficult for human annotators to classify unambiguously, it is unreasonable to expect a machine algorithm to do it well, or in other words, misclassifications will receive reduced penalty. On the other hand, if the human annotators are in strong agreement about a classification of a data item, then we would increase the penalty for any misclassification. This measure captures the biases in the data using disagreement among annotators. Based on this intuition, we define PRM as the ratio of disagreement between the predicted and actual outcomes summed over disagreements between the annotators multiplied by a reduction factor that reduces the penalty if the prediction matches any other annotator. We formally define it as;

$$PRM = \frac{1}{N_T} \sum_{i=1}^{N_T} \left(\frac{1 + \Delta(r'_i, r^o_i)}{1 + \sum_{m,q \in G^i, m \neq q} \Delta(G^i_m, G^i_q)} \cdot \frac{\sum_{m \in G^i} I(r'_i = G^i_m)}{|\mathbf{G^i}|} \right) \quad (8)$$

Where the denominator is the disagreement between G^i_m and G^i_q annotators summed over all annotators in a group $\mathbf{G^i}$ (notations are same as in equation 5). $\frac{\sum_{m \in G^i} I(r'_i = G^i_m)}{|\mathbf{G^i}|}$ is the risk reducing factor calculated as the ratio of agreement of prediction with any of the annotators over the total number of annotators. In cases where r' disagrees with all the annotators in $\mathbf{G}$, the risk reducing factor is set to 1.

6 RESULTS AND ANALYSIS

We evaluate the model performance over different levels of suicide severity. We categorize our experiments into three schemes: **Experiment 1** evaluates the performance of the models over 5 labels (supportive, indicator, ideation, behavior, and attempt); **Experiment 2** evaluates models' performances over 4 labels in which supportive (or negative) samples are removed, and **Experiment 3** comprises labels defined according to 4-label categorization (where supportive and indicator classes are merged into one class : no-risk). Further, for each experiment, the input data is of two forms: **(I1)** Only textual features (TF) represented as vectors of 300 dimensions generated using ConceptNet embeddings , **(I2)** having Characteristics features (CF) (see Section 5.1) and textual features (CF+TF). All experiments were performed with 5 fold hold-out cross-validation. It was defined empirically, observing results at various folds. We show that the proposed 5-label classification scheme has better recall, and the perceived risk measure of the 5-label classification scheme is low compared to other reduced classification schemes. All the experiments have been performed with 5-fold cross validation and results are reported on hold-out test set.

6.1 Experiment 1: 5-Label Classification

For evaluation, we consider five learning models (SVM with Radial Basis Function (SVM-RBF), SVM with Linear Kernel (SVM-L), Random Forest (RF), Feed-Forward Neural Network (FFNN), CNN) that have been used in similar studies (see Section 2.3) over two types of inputs: I1 and I2. For input I1, the baseline is a suicide-lexicon based

classifier which is content-based, and for input I2, the baseline is SVM-L which is the best performing model in Shing et al. [54].

Table 7: Experiment with 5-label Classification

Approach	Input	With Supportive Class			
		Graded Precision	Graded Recall	F-Score	OE
Baseline	text	0.56	0.36	0.44	0.38
SVM-RBF	I1	0.53	0.51	0.52	0.12
	I2	0.57	0.62	0.61	0.12
SVM-L	I1	0.60	0.45	0.52	0.12
	I2	0.77	0.40	0.53	0.09
RF	I1	0.68	0.49	0.57	0.19
	I2	0.62	0.45	0.52	0.11
FFNN	I1	0.45	0.59	0.51	0.15
	I2	0.52	0.63	0.57	0.12
CNN	I1	0.71	**0.60**	0.65	0.10
	I2	0.70	**0.59**	0.64	0.09

Input type I1: Table 7 reports that CNN outperforms the baseline with an improvement of 40% in precision, 5% in recall, and 25% in F-score. Based on small improvement in recall, it is inferred that CNN has a tendency to predict a low risk level (e.g; Supportive) for a user who has an observed high risk (e.g; Behavior). SVM-RBF and SVM-L show an improvement in precision compared to baseline; however, there is 12% and 27% reduction in recall respectively. Further, RF showed a 40% increase in precision at a cost of 16% reduction in recall. On the contrary, FFNN performed relatively well in comparison to baseline concerning recall. Hence, at a fine-grained level of comparison, CNN outperforms the baseline with a considerable improvement in precision and recall. To better characterize the comparison between the models, we analyze them using OE. Such a measure is coarse-grained and focuses more on FN as opposed to acceptable FP. Based on Table 7, we observed that CNN showed the least error based on OE calculation, reporting that 1% of the people have been predicted with a severity level of difference 2 or more compared to observed. Such a measure of evaluation is important because it ignores the biases in the gold standard data. As a result, CNN correctly predicted the severity of 90% of users.

Input Type I2: In comparison to the second baseline, CNN outperforms SVM-L with an improvement of 32% in recall with reduction of 10% in precision. We infer from Table 7 that SVM penalized false positives more than false negatives because of its linearity and i.i.d (independent, identically distributed)[17] assumptions. Whereas, CNN's convoluted representation ignores i.i.d assumptions, the non-linearity induced by ReLU tries to balance FP and FN. It can be seen from recall of SVM-RBF for I2 which is higher than SVM-L. However, SVM-RBF fails to balance FP and FN because of i.i.d considerations. Further, from column OE in Table 7, we infer that CNN predicted a suicidality level >1 compared to observed, for 9% of the users, whereas SVM-L did for 10% of the users.

6.2 Experiment 2: 4-label Classification

To evaluate the models over 4-label classification scheme, we use the same approach as applied in Experiment 1 for the purpose of consistency. In addition, in this experiment, the baseline model created over suicide lexicon disregards supportive labels.

[17] https://bit.ly/2Rw9i5Z

Table 8: Experiment with 4-label Classification

Approach	Input	Without Supportive Class			
		Graded Precision	Graded Recall	F-Score	OE
Baseline	text	0.43	0.57	0.49	0.20
SVM-RBF	I1	0.63	0.47	0.54	0.12
	I2	0.66	0.59	0.62	0.12
SVM-L	I1	0.62	0.53	0.57	0.12
	I2	0.68	0.57	0.61	0.09
RF	I1	0.67	0.41	0.51	0.22
	I2	0.64	0.47	0.54	0.18
FFNN	I1	0.63	0.58	0.60	0.15
	I2	0.67	0.62	0.64	0.12
CNN	I1	0.72	**0.59**	0.65	0.11
	I2	0.70	**0.57**	0.62	0.1

Observing Tables 7 and 8, there is a noticeable improvement in the precision of the models due to reduction in the degree of freedom of the outcome variable (removal of supportive class). Moreover, Tables 7 and 8 show the reduction in recall and an increase in OE. Hence, 5-label scheme supports lower OE for best performing model than does 4-label scheme.

Input Type I1 and Input Type I2: For the content-based input, all the models outperform the baseline in terms of precision, however, only CNN model outperforms baseline in terms of recall. Interestingly, there is a decrease in the recall of the models with non-linear kernel from 5-label to 4-label classification scheme; yet, there is a marginal increase in true positives of SVM-L. It can be inferred that SVM-L is vulnerable to predicting some of indicator users as supportive and ideation users as indicator in experiment 1. However, CNN was able to identify supportive users and most of the classification was centered around ideation and indicator levels; 4-label scheme does not bring in major change in OE for CNN.

6.3 Experiment 3: 3+1 Classification

In this classification scheme, we collapsed the supportive and indicator classes into a common class: "control group". It allows us to create the classification structure as defined in [12]. For this experiment, we considered two top performing models from previous experiments: SVM-L and CNN.

Table 9: Experiment with 3+1-label Classification

Approach	Input	Collapsed Supportive and Indicator Class			
		Graded Precision	Graded Recall	F-Score	OE
SVM-L	I1	0.81	0.54	0.65	0.12
	I2	0.74	0.54	0.63	**0.09**
CNN	I1	0.83	**0.57**	0.676	0.07
	I2	0.85	**0.57**	0.68	**0.06**

Input type I1 and I2: Using such a classification scheme (see Table 9), we observe a significant improvement in precision of SVM-linear and CNN in comparison to previous experiments. Apart from the decrease in the degree of freedom of outcome, the model tries to predict the supportive+indicator and ideation classes as opposed to "behavior" and "attempt". Since supportive+indicator and ideation classes are in majority, they boost the precision of the model. However, the model shows a reduction in recall in this scheme compared to 5-label or 4-label classification scheme. Table 10 shows reduction in OE for CNN from 0.1 to 0.07 for I1 and 0.09 to 0.06 for I2 compared to 5-label classification. It is because 3+1 classification

scheme forces the model to compromise with the popular classes and affect the selection of suitable class. Moreover, through our 5-label classification scheme, we achieved an improvement of 4.2% in graded recall over the (3+1) scheme (see Tables 7 and 9).

6.4 5-label Confusion Matrix Analysis

In this evaluation metric we categorize our suicidality labels into two groups; (1) No-Treatment Groups: Supportive and Indicator User, (2) Treatment Groups: Ideation, Behavior, Attempt.

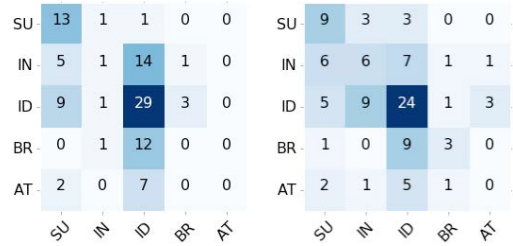


Figure 6: Confusion Matrix of 5-label scheme. (left) CNN, and (right) SVM-L. Y-Axis: True Level, X-Axis: Predicted Level

From Figure 6, out of 36 No-treatment users, CNN correctly classifies 20 users (56%) whereas SVM-L correctly classifies 22 users (61%). However, observing a larger 64 Treatment users, CNN correctly classifies 51 users (80%) whereas SVM-L correctly classifies 46 users (72%). Hence, CNN provides more suitable class for the users compared to SVM-L.

6.5 4-Label Confusion Matrix Analysis

Under the 4-label classification scheme, the No-treatment population involves users annotated as indicator whereas Treatment population contains users annotated as ideation, behavior and attempt. From Figure 7, we observe that CNN correctly classifies 59 out of 64 users (92%) annotated under Treatment whereas SVM-L classifies 53 out of 64 users (83%).

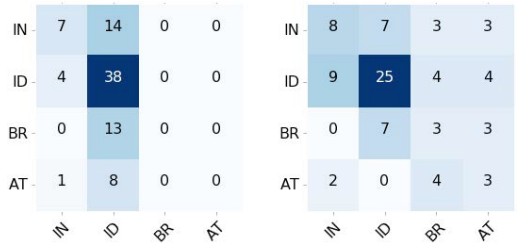


Figure 7: Confusion Matrix of 4-label scheme. (left) CNN, and (right) SVM-L

6.6 3+1 Label Confusion Matrix Analysis

There are 36 users under No-Treatment (supportive and indicator (SU+IN)) group and 64 users under Treatment group. Based on figure 8, we noticed that CNN correctly classifies 26 out of 36 (72%) No-Treatment users whereas SVM-L scored 16 out of 36 (44%). Further, CNN and SVM-L recognized 39 users (61%) and 46 users (72%) in the Treatment group. The decrease in CNN from 80% (5-label) to 61% is attributed to the increase in attempt, behavior, and ideation users classified as No-Treatment.

However, there was no change for SVM-L. But, on comparing 5-label and 3+1 label classification schemes, we observed that collapsing of the supportive and indicator classes can lead to increase in the false positive as SVM-L predicts them as behavior and attempt. There is a reduction in the true positive score for predictive

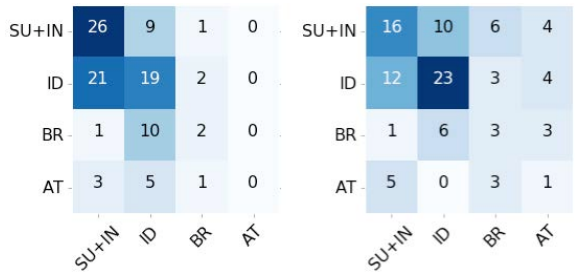


Figure 8: Confusion Matrix of (3+1)-label scheme. (left) CNN, and (right) SVM-L

and actual ideation classes, and users marked as "attempt" have been classified as "supportive and indicator (SU+IN)". As a result, the false negatives of the models have increased. Although this analysis proves the efficacy of 5-label classification over 3+1, CNN being a conservative model, there is a possibility of annotator bias in the data. So below, we perform PRM analysis of SVM-L and CNN over 2 classification schemes: 5-label and 3+1 label.

6.7 Perceived Risk Measure Analysis

Table 10: PRM based comparison of classification schemes

Scheme	Models	PRM
5-Label	CNN	0.14
	SVM-L	0.61
(3+1) Label	CNN	0.16
	SVM-L	0.54

On analyzing models behavior using PRM (Equation 8), Table 10 shows that there is a 12.5% difference between 5-label and 3+1 label classification schemes. Results can be interpreted as: For CNN under 5-label, there is 14% chance that model will provide an outcome that disagrees with every annotator, whereas, for (3+1)-label, it is 16%. Further, we observe that SVM-L has a high risk score compared to CNN in both classification schemes.

7 CONCLUSION

In this study, we presented an approach to predict severity of suicide risk of an individual using Reddit posts, which will allow medical health professionals to make more informed and timely decisions on diagnosis and treatment. A gold standard dataset of 500 suicidal redditors with varying severity of suicidal risk was developed using suicide risk severity lexicon. We then devised a 5-label classification scheme to differentiate non-suicidal users from suicidal ones, as well as suicidal users at different severity levels of suicide risk (e.g., ideation, behavior, attempt). Our 5-label classification scheme outperformed the two baselines. We specifically noted that CNN provided best performance among others including SVM and Random Forest. We make both the gold standard dataset and the suicide risk severity lexicon publicly available to the research community for further suicide-related research.

ACKNOWLEDGEMENT

We acknowledge partial support from the National Science Foundation (NSF) award CNS-1513721: "Context-Aware Harassment Detection on Social Media", National Institutes of Health (NIH) award: MH105384-01A1: "Modeling Social Behavior for Healthcare Utilization in Depression", and National Institute on Drug

Abuse (NIDA) Grant No. 5R01DA039454-02 “Trending: Social media analysis to monitor cannabis and synthetic cannabinoid use”. Dr. Ramakanth Kavuluru was supported by the NIH National Cancer Institute through grant R21CA218231. Any opinions, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF, NIH, or NIDA.

REFERENCES

[1] Amanuel Alambo, Manas Gaur, Usha Lokala, Ugur Kursuncu, Krishnaprasad Thirunarayan, Amelie Gyrard, Randon S Welton, Jyotishman Pathak, and Amit Sheth. 2019. Question Answering for Suicide Risk Assessment using Reddit. *IEEE International Conference on Semantic Computing 2019* (2019).

[2] Payam Amini, Hasan Ahmadinia, Jalal Poorolajal, and Mohammad Moqaddasi Amiri. 2016. Evaluating the high risk groups for suicide: A comparison of logistic regression, support vector machine, decision tree and artificial neural network. *Iranian journal of public health* 45, 9 (2016), 1179.

[3] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2016. A simple but tough-to-beat baseline for sentence embeddings. (2016).

[4] Harish S Bhat and Sidra J Goldman-Mellor. 2017. Predicting Adolescent Suicide Attempts with Neural Networks. *arXiv preprint arXiv:1711.10057* (2017).

[5] J Michael Bostwick, Chaitanya Pabbati, Jennifer R Geske, and Alastair J McKean. 2016. Suicide attempt as a risk factor for completed suicide: even more lethal than we knew. *American journal of psychiatry* 173, 11 (2016), 1094–1100.

[6] Camille Brisset, Yvan Leanza, Ellen Rosenberg, Bilkis Vissandjée, Laurence J Kirmayer, Gina Muckle, Spyridoula Xenocostas, and Hugues Laforce. 2014. Language barriers in mental health care: A survey of primary care practitioners. *Journal of immigrant and minority health* 16, 6 (2014), 1238–1246.

[7] Delroy Cameron, Gary A Smith, Raminta Daniulaityte, Amit P Sheth, Drashti Dave, Lu Chen, Gaurish Anand, Robert Carlson, Kera Z Watkins, and Russel Falck. 2013. PREDOSE: a semantic web platform for drug abuse epidemiology using social media. *Journal of biomedical informatics* 46, 6 (2013), 985–997.

[8] Patricia A Cavazos-Rehg, Melissa J Krauss, Shaina J Sowles, Sarah Connolly, Carlos Rosas, Meghana Bharadwaj, Richard Grucza, and Laura J Bierut. 2016. An analysis of depression, self-harm, and suicidal ideation content on Tumblr. *Crisis* (2016).

[9] Helen Christensen, Philip Batterham, and Bridianne O'Dea. 2014. E-health interventions for suicide prevention. *International journal of environmental research and public health* 11, 8 (2014), 8193–8212.

[10] Kevin Bretonnel Cohen, Dina Demner-Fushman, Sophia Ananiadou, and Jun-ichi Tsujii. 2016. Proceedings of the 15th Workshop on Biomedical Natural Language Processing.

[11] Glen Coppersmith, Ryan Leary, Eric Whyne, and Tony Wood. 2015. Quantifying suicidal ideation via language usage on social media. In *Joint Statistics Meetings Proceedings, Statistical Computing Section, JSM*.

[12] Darcy J Corbitt-Hall, Jami M Gauthier, Margaret T Davis, and Tracy K Witte. 2016. College students' responses to suicidal content on social networking sites: an examination using a simulated facebook newsfeed. *Suicide and Life-Threatening Behavior* 46, 5 (2016), 609–624.

[13] Raminta Daniulaityte, Robert Carlson, Gregory Brigham, Delroy Cameron, and Amit Sheth. 2015. “Sub is a weird drug:” A web-based study of lay attitudes about use of buprenorphine to self-treat opioid withdrawal symptoms. *The American journal on addictions* 24, 5 (2015), 403–409.

[14] Raminta Daniulaityte, Francois R Lamy, G Alan Smith, Ramzi W Nahhas, Robert G Carlson, Krishnaprasad Thirunarayan, Silvia S Martins, Edward W Boyer, and Amit Sheth. 2017. “Retweet to Pass the Blunt”: Analyzing Geographic and Content Features of Cannabis-Related Tweeting Across the United States. *Journal of studies on alcohol and drugs* 78, 6 (2017), 910–915.

[15] Munmun De Choudhury, Scott Counts, and Eric Horvitz. 2013. Social media as a measurement tool of depression in populations. In *Proceedings of the 5th Annual ACM Web Science Conference*. ACM, 47–56.

[16] Munmun De Choudhury, Michael Gamon, Scott Counts, and Eric Horvitz. 2013. Predicting depression via social media. *ICWSM* 13 (2013), 1–10.

[17] Munmun De Choudhury, Emre Kiciman, Mark Dredze, Glen Coppersmith, and Mrinal Kumar. 2016. Discovering shifts to suicidal ideation from mental health content in social media. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. ACM, 2098–2110.

[18] Peter Sheridan Dodds, Kameron Decker Harris, Isabel M Kloumann, Catherine A Bliss, and Christopher M Danforth. 2011. Temporal patterns of happiness and information in a global social network: Hedonometrics and Twitter. *PloS one* 6, 12 (2011), e26752.

[19] Jingcheng Du, Yaoyun Zhang, Jianhong Luo, Yuxi Jia, Qiang Wei, Cui Tao, and Hua Xu. 2018. Extracting psychiatric stressors for suicide from social media using deep learning. *BMC medical informatics and decision making* 18, 2 (2018), 43.

[20] Tianfan Fu, Cheng Zhang, and Stephan Mandt. 2018. Continuous Word Embedding Fusion via Spectral Decomposition. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*. 11–20.

[21] Manas Gaur, Ugur Kursuncu, Amanuel Alambo, Amit Sheth, Raminta Daniulaityte, Krishnaprasad Thirunarayan, and Jyotishman Pathak. 2018. Let Me Tell You About Your Mental Health!: Contextualized Classification of Reddit Posts to DSM-5 for Web-based Intervention. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 753–762.

[22] George Gkotsis, Anika Oellrich, Sumithra Velupillai, Maria Liakata, Tim JP Hubbard, Richard JB Dobson, and Rina Dutta. 2017. Characterisation of mental health conditions in social media using Informed Deep Learning. *Scientific reports* 7 (2017), 45141.

[23] George Gkotsis, Sumithra Velupillai, Anika Oellrich, Harry Dean, Maria Liakata, and Rina Dutta. 2016. Don't Let Notes Be Misunderstood: A Negation Detection Method for Assessing Risk of Suicide in Mental Health Records. In *Proceedings of the Third Workshop on Computational Lingusitics and Clinical Psychology*. 95–105.

[24] David M Howcroft and Vera Demberg. 2017. Psycholinguistic Models of Sentence Processing Improve Sentence Readability Ranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Vol. 1. 958–968.

[25] Yi-Ting Huang, Meng Chang Chen, and Yeali S Sun. 2018. Characterizing the Influence of Features on Reading Difficulty Estimation for Non-native Readers. *arXiv preprint arXiv:1808.09718* (2018).

[26] Nina Hyams and Kenneth Wexler. 1993. On the grammatical basis of null subjects in child language. *Linguistic inquiry* (1993), 421–459.

[27] Zahurul Islam and Alexander Mehler. 2013. Automatic readability classification of crowd-sourced data based on linguistic and information-theoretic features. *Computación y Sistemas* 17, 2 (2013), 113–123.

[28] Jared Jashinsky, Scott H Burton, Carl L Hanson, Josh West, Christophe Giraud-Carrier, Michael D Barnes, and Trenton Argyle. 2014. Tracking suicide risk factors through Twitter in the US. *Crisis* (2014).

[29] Shaoxiong Ji, Celina Ping Yu, Sai-fu Fung, Shirui Pan, and Guodong Long. 2018. Supervised Learning for Suicidal Ideation Detection in Online User Content. *Complexity* 2018 (2018).

[30] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[31] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[32] Ugur Kursuncu, Manas Gaur, Usha Lokala, Anurag Illendula, Krishnaprasad Thirunarayan, Raminta Daniulaityte, Amit Sheth, and I Budak Arpinar. 2018. "What's ur type?" Contextualized Classification of User Types in Marijuana-related Communications using Compositional Multiview Embedding. *IEEE/WIC/ACM International Conference on Web Intelligence (WI)* (2018).

[33] Ugur Kursuncu, Manas Gaur, Usha Lokala, Krishnaprasad Thirunarayan, Amit Sheth, and I Budak Arpinar. 2019. Predictive Analysis on Twitter: Techniques and Applications. In *Emerging Research Challenges and Opportunities in Computational Social Network Analysis and Mining*. Springer, 67–104.

[34] Francois R Lamy, Raminta Daniulaityte, Ramzi W Nahhas, Monica J Barratt, Alan G Smith, Amit Sheth, Silvia S Martins, Edward W Boyer, and Robert G Carlson. 2017. Increases in synthetic cannabinoids-related harms: Results from a longitudinal web-based content analysis. *International Journal of Drug Policy* 44 (2017), 121–129.

[35] Hugo Leroux and Laurent Lefort. 2012. Using CDISC ODM and the RDF Data Cube for the Semantic Enrichment of Longitudinal Clinical Trial Data.. In *SWAT4LS*. Citeseer.

[36] Nut Limsopatham and Nigel Henry Collier. 2016. Normalising medical concepts in social media texts by learning semantic representation. (2016).

[37] Christopher E Moody. 2016. Mixing dirichlet topic models and word embeddings to make lda2vec. *arXiv preprint arXiv:1605.02019* (2016).

[38] Liqiang Nie, Yi-Liang Zhao, Mohammad Akbari, Jialie Shen, and Tat-Seng Chua. 2015. Bridging the vocabulary gap between health seekers and healthcare knowledge. *IEEE Transactions on Knowledge and Data Engineering* 27, 2 (2015), 396–409.

[39] Thomas Niederkrotenthaler, Arno Herberth, and Gernot Sonneck. 2007. The" Werther-effect": legend or reality? *Neuropsychiatrie: Klinik, Diagnostik, Therapie und Rehabilitation: Organ der Gesellschaft Osterreichischer Nervenarzte und Psychiater* 21, 4 (2007), 284–290.

[40] Matthew K Nock, Guilherme Borges, Evelyn J Bromet, Jordi Alonso, Matthias Angermeyer, Annette Beautrais, Ronny Bruffaerts, Wai Tat Chiu, Giovanni De Girolamo, Semyon Gluzman, et al. 2008. Cross-national prevalence and risk factors for suicidal ideation, plans and attempts. *The British Journal of Psychiatry* 192, 2 (2008), 98–105.

[41] Denis Paperno and Marco Baroni. 2016. When the whole is less than the sum of its parts: How composition affects pmi values in distributional semantic vectors. *Computational Linguistics* 42, 2 (2016), 345–350.

[42] James W Pennebaker. 2011. The secret life of pronouns. *New Scientist* 211, 2828 (2011), 42–45.

[43] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on*

empirical methods in natural language processing (EMNLP). 1532–1543.

[44] Kelly Posner, Gregory K Brown, Barbara Stanley, David A Brent, Kseniya V Yershova, Maria A Oquendo, Glenn W Currier, Glenn A Melvin, Laurence Greenhill, Sa Shen, et al. 2011. The Columbia–Suicide Severity Rating Scale: initial validity and internal consistency findings from three multisite studies with adolescents and adults. *American Journal of Psychiatry* 168, 12 (2011), 1266–1277.

[45] Ali Pourmand, Jeffrey Roberson, Amy Caggiula, Natalia Monsalve, Murwarit Rahimi, and Vanessa Torres-Llenza. 2018. Social Media and Suicide: A Review of Technology-Based Epidemiology and Risk Assessment. *Telemedicine and e-Health* (2018).

[46] Hemant Purohit, Andrew Hampton, Valerie L Shalin, Amit P Sheth, John Flach, and Shreyansh Bhatt. 2013. What kind of# conversation is Twitter? Mining# psycholinguistic cues for emergency coordination. *Computers in Human Behavior* 29, 6 (2013), 2438–2447.

[47] Philip Resnik, Anderson Garron, and Rebecca Resnik. 2013. Using topic modeling to improve prediction of neuroticism and depression in college students. In *Proceedings of the 2013 conference on empirical methods in natural language processing*. 1348–1353.

[48] Jo Robinson, Maria Rodrigues, Steve Fisher, and Helen Herrman. 2014. Suicide and social media. *Melbourne, Australia: Young and Well Cooperative Research Centre* (2014).

[49] Xin Rong. 2014. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738* (2014).

[50] H Andrew Schwartz, Johannes Eichstaedt, Margaret L Kern, Gregory Park, Maarten Sap, David Stillwell, Michal Kosinski, and Lyle Ungar. 2014. Towards assessing changes in degree of depression through facebook. In *Proceedings of the Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*. 118–125.

[51] H Andrew Schwartz, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski, Richard E Lucas, Megha Agrawal, Gregory J Park, Shrinidhi K Lakshmikanth, Sneha Jha, Martin EP Seligman, et al. 2013. Characterizing Geographic Variation in Well-Being Using Tweets.. In *ICWSM*. 583–591.

[52] Amit Sheth, Utkarshani Jaimini, and Hong Yung Yip. 2018. How Will the Internet of Things Enable Augmented Personalized Health? *IEEE intelligent systems* 33, 1 (2018), 89–97.

[53] Joongbo Shin, Yanghoon Kim, Seunghyun Yoon, and Kyomin Jung. 2018. Contextual-CNN: A Novel Architecture Capturing Unified Meaning for Sentence Classification. In *Big Data and Smart Computing (BigComp), 2018 IEEE International Conference on*. IEEE, 491–494.

[54] Han-Chin Shing, Suraj Nair, Ayah Zirikly, Meir Friedenberg, Hal Daumé III, and Philip Resnik. 2018. Expert, Crowdsourced, and Machine Assessment of Suicide Risk via Online Postings. In *Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic*. 25–36.

[55] Guillermo Soberón, Lora Aroyo, Chris Welty, Oana Inel, Hui Lin, and Manfred Overmeen. 2013. Measuring crowd truth: Disagreement metrics combined with worker behavior filters. In *CrowdSem 2013 Workshop*.

[56] Shaina J Sowles, Melissa J Krauss, Lewam Gebremedhn, and Patricia A Cavazos-Rehg. 2017. "I feel like I've hit the bottom and have no idea what to do": supportive social networking on Reddit for individuals with a desire to quit cannabis use. *Substance abuse* 38, 4 (2017), 477–482.

[57] Robyn Speer and Joanna Lowry-Duda. 2017. Conceptnet at semeval-2017 task 2: Extending word embeddings with multilingual relational knowledge. *arXiv preprint arXiv:1704.03560* (2017).

[58] Hajime Sueki. 2015. The association of suicide-related Twitter use with suicidal behaviour: a cross-sectional study of young internet users in Japan. *Journal of affective disorders* 170 (2015), 155–160.

[59] C Lee Ventola. 2014. Social media and health care professionals: benefits, risks, and best practices. *Pharmacy and Therapeutics* 39, 7 (2014), 491.

[60] Eduard Vieta and Marc Valentí. 2013. Mixed states in DSM-5: implications for clinical care, education, and research. *Journal of affective disorders* 148, 1 (2013), 28–36.

[61] Colin G Walsh, Jessica D Ribeiro, and Joseph C Franklin. 2017. Predicting risk of suicide attempts over time through machine learning. *Clinical Psychological Science* 5, 3 (2017), 457–469.

[62] Wenbo Wang, Lu Chen, Keke Chen, Krishnaprasad Thirunarayan, and Amit P Sheth. 2017. Adaptive training instance selection for cross-domain emotion identification. In *Proceedings of the International Conference on Web Intelligence*. ACM, 525–532.

[63] Wenbo Wang, Lu Chen, Ming Tan, Shaojun Wang, and Amit P Sheth. 2012. Discovering fine-grained sentiment in suicide notes. *Biomedical informatics insights* 5 (2012), BII–S8963.

[64] Xufei Wang, Lei Tang, Huiji Gao, and Huan Liu. 2010. Discovering overlapping groups in social media. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 569–578.

[65] Andrea N Weber, Maria Michail, Alex Thompson, and Jess G Fiedorowicz. 2017. Psychiatric emergencies: assessing and managing suicidal ideation. *Medical Clinics* 101, 3 (2017), 553–571.

SpeedReader: Reader Mode Made Fast and Private

Mohammad Ghasemisharif
University of Illinois at Chicago
Chicago, IL, USA
mghas2@uic.edu

Peter Snyder
Brave Software
San Francisco, CA, USA
pes@brave.com

Andrius Aucinas
Brave Software
London, UK
aaucinas@brave.com

Benjamin Livshits
Brave Software & Imperial College London
London, UK
ben@brave.com

ABSTRACT

Most popular web browsers include "reader modes" that improve the user experience by removing un-useful page elements. Reader modes reformat the page to hide elements that are not related to the page's main content. Such page elements include site navigation, advertising related videos and images, and most JavaScript. The intended end result is that users can enjoy the content they are interested in, without distraction.

In this work, we consider whether the "reader mode" can be widened to also provide performance and privacy improvements. Instead of its use as a post-render feature to clean up the clutter on a page we propose `SpeedReader` as an alternative multistep pipeline that is part of the rendering pipeline. Once the tool decides during the initial phase of a page load that a page is suitable for reader mode use, it directly applies document tree translation *before the page is rendered*. Based on our measurements, we believe that `SpeedReader` can be continuously enabled in order to drastically improve end-user experience, especially on slow mobile connections. Combined with our approach to predicting which pages should be rendered in reader mode with 91% accuracy, `SpeedReader` achieves average speedups and bandwidth reductions of up to 27× and 84×, respectively. We further find that our novel "reader mode" approach brings with it significant privacy improvements to users. Our approach effectively removes all commonly recognized trackers, issues 115 fewer requests to third parties, and interacts with 64 fewer trackers on average, on transformed pages.

CCS CONCEPTS

• **Human-centered computing** → **Web-based interaction**; • **Information systems** → **Browsers**; **Clustering and classification**; *Content analysis and feature selection*; • **Security and privacy** → *Privacy protections.*

KEYWORDS

Reader Mode; Boilerplate Removal; Web Document Classification; Web Performance; Ad Blocking

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313596

ACM Reference Format:
Mohammad Ghasemisharif, Peter Snyder, Andrius Aucinas, and Benjamin Livshits. 2019. SpeedReader: Reader Mode Made Fast and Private. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313596

1 INTRODUCTION

"Web bloat" is a colloquial term that describes the trend in websites to accumulate size and visual complexity over time. The phenomena has been measured in many dimensions, including total page size [7], page load time [5, 44, 45], memory needed [30], number of network requests [16, 28], amount of scripts executed [26, 34, 37, 39] and third parties contacted [25, 26, 28]. This work suggests that growth in page size and complexity is outpacing improvements in device hardware. All of this has a predictably negative impact on user experience.

Web users and browser vendors have reacted to this "bloat" in a variety of ways, all partially helpful, but with significant downsides. Ad and tracking blockers are a popular and useful tool for reducing the size complexity of sites. Prior work has shown that these tools can be effective in reducing privacy leaks [31], network use, and extend device memory life. Such tools, which use filter lists, are inherently limited in the scope of improvements they can achieve. While these filter lists are large [42], they are small as a proportion of all URLs on the web. Similarly, while these lists are updated often, they are updated slowly compared to URL updates on the web.

Similarly, "reader mode" tools, provided in many popular browsers and browser extensions, are an effort to reduce the growing visual complexity of websites. Such tools attempt to extract the subset of page content useful to users, and remove advertising, animations, boilerplate code, and other non-core content. Current "reader modes" do not provide the user with resource savings since the referenced resources have already been fetched and rendered. The growth and popularity of such tools suggest they are useful to browser users, looking to address the problem of page clutter and visual "bloat".

In this work, we propose a novel strategy called **`SpeedReader`** for dealing with resource and bloat on websites. Our technique provides a user experience similar to existing "reader mode" tools, but with network, performance, and privacy improvements that exceed existing ad and tracking blocking tools, on a significant portion of websites. Significantly, `SpeedReader` differs from existing deployed reader mode tools by operating *before page rendering*,

which allows it to determine which resources are needed for the page's core content before fetching.

How we achieve speedups. SpeedReader achieves its performance improvements through a two-step pipeline:

(1) SpeedReader uses a classifier to determine whether there is a readable subset of the initial, fetched page HTML. This classifier is trained on a labeled corpus of 2,833 websites (see Section 3), and determines whether a page can be display in reader mode with 91% accuracy.
(2) If the classifier has determined that the page is readable, SpeedReader extracts the readable subset of document *before rendering*, using a variety of heuristics developed in prior research [24] and browser vendors [9, 23], and passes the simplified, reader mode document to the browser's render layer. This tree translation step is described in Section 4.

Deployment. Combined with a highly accurate classifier of "readable" pages, the drastic improvements in performance, reduction in bandwidth use and elimination of trackers in reader mode make the approach practical for continuous use. We therefore propose SpeedReader as a sticky feature that a user can toggle to be always on. This approximates the experience of using an e-book reader, but with strengths of content availability on the web. It is also a suitable strategy for content prerendering or prefetching that could be implemented by web browser vendors, automatically delivering graceful performance degradation in poor connectivity areas or on underpowered mobile devices until the rest of the page content can be fetched for a complete render.

Contributions.

- **Novel approach to Reader Mode** - combining machine-learning driven approach to checking whether content can be transformed to text-focused representation for end-user consumption.
- **Applicability** - we demonstrate that 22.0% of web pages are convertible to reader mode style in a dataset of pages reported popular by Alexa. We further demonstrate that 46.27% of pages shared on social networks are *readable*.
- **Privacy** - we demonstrate that using reader mode in the proposed design provides superior privacy protection, effectively removing all trackers from the tested pages, and dramatically reducing communication with third-parties.
- **Ad Blocking** - we show that our unique reader mode approach blocks ads *at least as well* as existing ad blocking tools, blocking 100% of resources labeled as advertising related by EasyList in a crawl of 91,439 pages, without the need to use hand curated, hard-coded filter lists.
- **Speed** - we find that the lightweight nature of reader mode content results in huge performance gains, with up to 27× page load time speedup on average, together with up to 84× bandwidth and 2.4× memory reduction on average.

Paper organization. The rest of this paper is structured as follows. Section 2 provides background information to place SpeedReader in context. Section 3 describes the design, evaluation and accuracy of the classifying step in the SpeedReader pipeline, and Section 4 gives the design and evaluation of the reader mode extraction step in the SpeedReader pipeline. Section 3.3 measures how many websites user encounters that are amenable to SpeedReader, under several browser use scenarios. Section 5 provides some discussion for how our findings can inform future readability, privacy and performance work, Section 6 places this work in the context of prior research, and Section 7 concludes.

2 BACKGROUND

2.1 Terminology

This subsection presents several terms that are not standardized. We present them up front, to ease the understanding of the rest of the work.

Reader mode. We use the term "reader mode" to describe any tool that attempts to extract a useful subset of a website for a simplified presentation. These tools can be either included in the web browser by the browser vendor, added by users through browser extensions, or provided by third parties as a web service. Our use of the term "reader mode" is generic to the concept, and should not be confused with any specific tool.

Classification and transduction. Reader mode tools generally include both a technique for determining whether a page is readable, which we refer to as "classification", and a strategy for converting the initial HTML tree into a simplified reader mode tree, which we refer to as "tree transduction". Though most reader mode tools include both steps within a single tool or library, they are conceptually distinct.

Readable. We use the term "readable" to describe whether a web page contains a subset of content that would be useful to display in a reader mode presentation. Reader mode presentation works best on pages that are text and image focused, and that are mostly static (i.e. few interactive page elements). Examples of such readable pages include articles, blog posts, and news sites. Reader mode presentation does not work well on websites that are highly interactive, or when a page's structure is significant to the page's content. Examples of such non-readable pages include web applications (e.g. Google Mail, Google Maps) or pages that are indexes of other content.

2.2 Existing Reader Modes

Several popular web browsers include reader modes designed to simplify a page's presentation, so that browser users can read the page's contents without distraction of visual clutter such as advertisements, page animations, and unnecessary page boilerplate (e.g. footers, page navigation, comments).

In this section, we give a brief description of several existing reader mode tools, how they're deployed by their authors, and how they are used in the evaluations given in the rest of this paper.

Readability.js. Readability.js [9] is an open source reader mode library, implemented in JavaScript. It is maintained by Mozilla, and is used for the reader mode function in Firefox. The code is closely related to "Readability" [2], an open sourced library developed by Arc90 and used for their now-defunct readability.com web service. Classification works by looking for the element on the page with the highest density of text and link nodes. If the number and density of text and link nodes in that element exceed a given threshold, the library treats the page as readable. Tree transduction works

by normalizing the contents of the text-and-link dense element (to remove styling and other mark up), looking for near-by images for inclusion, and using text patterns in the document that identify the page's author, source and publication date.

Significant to SpeedReader, Readability.js does not consider any display or presentation information when performing either the classification or tree transduction steps. This means that the page does not need to be loaded and rendered to generate a reader mode presentation (though in practice Firefox does not use this library in this way).

Safari Reader View. Safari Reader View is a JavaScript library that implements the reader mode presentation in Safari. Like Readability.js, it is also a fork from Arc90's "Readability", though Apple has changed how the library works in significant ways. In addition to looking for elements with high text and anchor density, Safari Reader View also uses presentation-level heuristics, including where elements appear on the page and what elements are hidden from display by default. Relevant to SpeedReader, this means that Safari Reader View must load a page and at least some of its resources (e.g. images, CSS, JavaScript) to perform either the classification or tree transduction level decisions. Because significant portions of Safari Reader View require a document be fetched and rendered before being evaluated, we do not consider it further in this work (for reasons that are detailed in Sections 3 and 4).

BoilerPipe. BoilerPipe is an academic research project from Kohlschütter *et al.* [24], and is implemented in Java. BoilerPipe has not been deployed directly by any browser vendor. BoilerPipe does not provide functionality for (readability) classification, and assumes that any HTML document contains a readable subset. For tree transduction, BoilerPipe considers number of words and link density features. Like Readability.js, it does not require a browser to load and render a page in order to do reader mode extraction. Their analysis reveals a strong correlation between short text and boilerplate, as well as long text and actual content text (of the textual content) on the Web. Using features with low calculation cost such as number of words enables BoilerPipe to lower the overhead while maintaining high accuracy.

DOM Distiller. DOM Distiller is a JavaScript and C++ library maintained by Google, and used to implement reader mode in recent versions of Chrome. The project is based on BoilerPipe, though has been significantly changed by Google. The classification step in DOM Distiller uses a classifier based approach, and considers features such as whether the page's URL contains certain keywords (e.g. "forum", ".php", "index"), if the page's markup contains Facebook open graph, Google AMP, identifiers, or the number of "/" characters used in the URL's path, in addition to the text-and-link density measures used by Readability.js. At a high level, the tree transduction step also looks at text-and-link dense element in the page, as well as special-cased embedded elements, such as YouTube or Vimeo videos.

DOM Distiller considers some render-level information in both the classification and tree transduction steps. For example, any elements that are hidden from display are not included in the text-and-link density measurements. These render-level checks are a small part of DOM Distiller's strategy. We modified DOM Distiller to remove these display level checks, so that DOM Distiller could be applied to prerendered HTML documents. We note that the evaluation of DOM Distiller in this work uses this modified version of DOM Distiller, and not the version that Google ships with Chrome. We expect this modification to have minimal effects on the discussed measurements, but draw the reader's attention to this change for completeness.

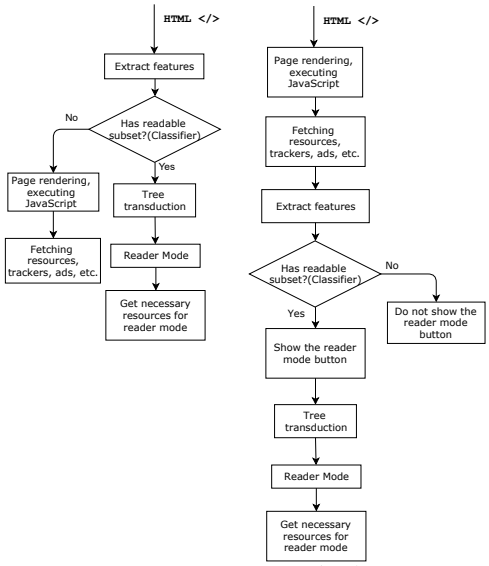


Figure 1: Comparison of SpeedReader (left) with other existing reader modes (right)

2.3 Comparison to SpeedReader

The reader mode functionality shipped with all current major browsers is applied after the document is fully fetched and rendered.[1] This greatly restricts the possible performance, network and privacy improvements existing reader modes can achieve. In fact, in some reader mode implementations we measured, using reader modes *increased* the amount of network used, as some resources were fetched twice, i.e. once for the initial page loading, and then again when presenting images in the reader mode display.

Most significantly, SpeedReader differs from existing reader mode techniques in that it is implemented strictly before the display, rendering, and resource-fetching steps in the browser. SpeedReader can therefore be thought of as a function that sits between the browser's network layer (i.e. takes as input the initial HTML document), and returns either the received HTML (when there is no readable subset), or a greatly simplified HTML document, representing the reader mode presentation (when there is a readable subset). Figure 1 provides a high level comparison of how SpeedReader functions, compared to existing reader modes.

The fact that SpeedReader only considers features available in the initial HTML and URL enables SpeedReader to achieve performance orders of magnitude above existing approaches. Figure 2

[1]While Readability.js does not require that the page be rendered before making reader mode evaluations, in practice Firefox does not expose reader mode functionality to users until after the page is fetched and loaded.

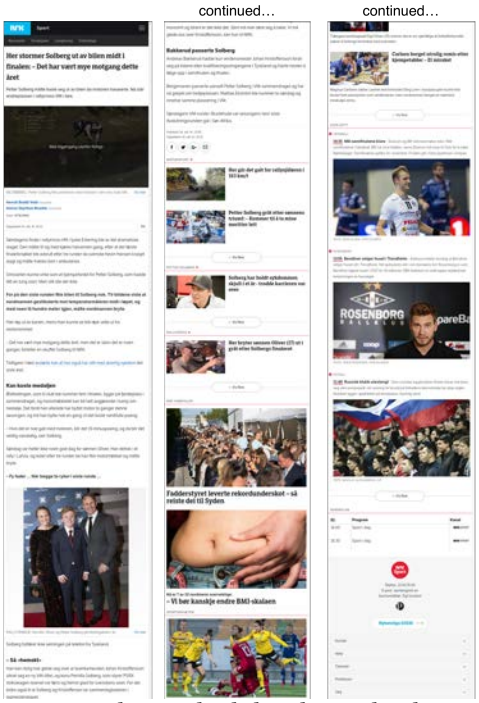


Figure 2: An example page loaded with Google Chrome browser with no modifications

provides a strawman example of a news page as delivered to a standard client: including portal branding and content, but also a range of links to different articles, images and trackers, for a total of 2.7MB of data transferred and 53 scripts executed. Figure 3 demonstrates the functionality of SpeedReader when applying existing reader mode transducers to just the initial HTML document. Therefore, for documents SpeedReader determines are readable, the sources of SpeedReader improvements include:

- Never fetching or executing script or CSS.
- Fetching far fewer images or videos (since images and videos not core to the page's presentation are never retrieved).
- Performing network requests to far fewer third parties (zero, in the common case).
- Saving processing power from not rendering animations, videos or complex layout operations, since reader mode presentations of page content are generally simple.

The above are just some of the ways that SpeedReader is able to achieve considerable performance improvements. The following sections describe how SpeedReader's classification and tree transduction steps were designed and evaluated, and what percentage of websites are amenable to SpeedReader's approach.

3 PAGE CLASSIFICATION

SpeedReader uses a two stage pipeline for generating reader mode versions of websites. This section presents the design and evaluation of the first half of the pipeline, the classification step.



Figure 3: The example page transformed with each of the evaluated SpeedReader transducers

3.1 Classifier Design

The classification step of SpeedReader uses a random forest classifier, trained on a hand-labeled data set of 2,833 websites. Our classifier takes as input a string, depicting an HTML document, and returns a boolean label of whether there is a readable subset of the document. We note that the input to the classifier is the initial HTML returned by the server, and not the final state of the website after JavaScript execution.

Our classifier is designed to execute quickly, since document rendering is delayed during classification. The classifier is trained using 50 estimators, it expands the nodes until all leaves are pure or contain less than 2 samples, and considers 21 features, each selected to be extractable quickly. Selected features include the number of text nodes, number of words, the presence of Facebook open graph or Google AMP markup, and counts for a variety of other tags.

Our classifier considers the following features. We have made the source code for our classifier available publicly as well.[2]

- Counts of the following tags: `<p>`, `<ul>`, `<ol>`, `<dl>`, `<div>`, `<pre>`, `<table>`, `<select>`, `<article>`, `<section>`, `<blockquote>`, `<a>`, `<img>`, `<script>`
- Count of block elements that contain at least 400 words.
- # of words in block elements that match above condition.
- Number of path segments in the URL.
- Boolean determination if the page has any of the following metatags: `amphtml`, `fb_pages`, `og_article`.
- Boolean determination if the page has plaintext match for any of `schema.org` markup for `Article`, `NewsArticle` or `APIReference`.

3.2 Classifier Accuracy

The goal of the classifier in SpeedReader is to predict whether the end result of a page's fetching and execution will result in a readable page, based on the initial HTML of the page. This section describes the data set we used to both train the SpeedReader classifier, and

[2] https://github.com/brave/speedreader-paper-materials.git

Table 1: Description of data set used for evaluating and training "readability" classifiers.

Data set	Number of pages	% Readable
Article pages	956	91.8%
Landing pages	932	1.5%
Random pages	945	22.0%
Total	**2,833**	**38.8%**

Table 2: Accuracy measurements for three classifiers attempting to replicate the manual labels described in Table 1.

Classifier	Precision	Recall
ReadabilityJS	68%	85%
DOM Distiller	90%	75%
SpeedReader Classifier	**91%**	**87%**

to evaluate its accuracy against existing popular, deployed reader mode tools.

Data Set. To assess the accuracy of our classifier, we first gathered 2,833 websites, summarized in Table 1. Our data set is made up of three smaller sets of crawled data, each containing 1,000 URLs, each meant to focus on a different kind of page, with a different expected distribution of readability. 1,000 pages were URLs selected from the RSS feeds of popular news sites (e.g. The New York Times, ArsTechnica), which we expected to be frequently readable. The second 1,000 pages were the landing pages from the Alexa 1K, which we expected to rarely be readable. The final 1,000 pages were selected randomly from non-landing pages linked from the landing pages of the Alexa 5K, which we expected to be occasionally readable. We built a crawler that, given a URL, recorded both the initial HTML response, and a screenshot of the final rendered page (i.e. after all resources had been fetched and rendered, and after JavaScript had executed). We applied our crawler to each of the 3,000 selected URLs. 167 pages did not respond to our crawler, accounting for the difference between the 3,000 selected URLs and the 2,833 pages in our data set.

Finally, we manually considered each of the final page screenshots, and gave each a boolean label of whether there was a subset of page content that was readable. We considered a page readable if it met the following criteria:

(1) The primary utility of the page was its text and image content (i.e. not interactive functionality).
(2) The page contained a subset of content that was useful, without being sensitive to its placement on the page.
(3) The usefulness of the page's content was not dependent on its specific presentation or layout on the website.

This meant that single page applications, index pages, and pages with complex layout were generally labeled as not-readable, while pages with generally static content, and lots of text and content-depicting media, were generally labeled readable. We also share our labeled data,[3] and a guide to the meaning behind the labels,[4] to make our results transparent and reproducible.

Evaluation. We evaluated our classifier on our hand labeled corpus of 2,833 websites, performing a standard ten-fold cross-validation.

[3]https://github.com/brave/speedreader-paper-materials/blob/master/labels.csv
[4]https://github.com/brave/speedreader-paper-materials/blob/master/labels-legend.txt

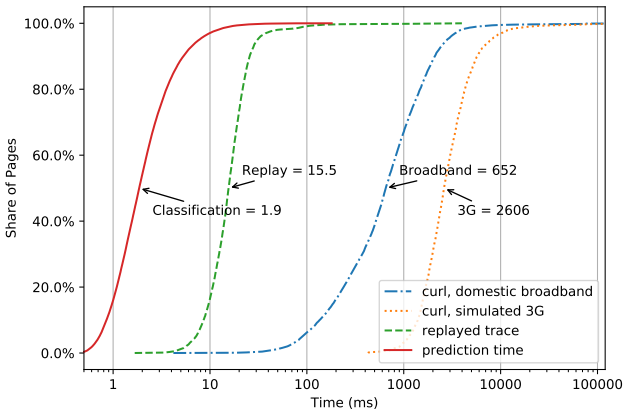


Figure 4: Time to fetch initial HTML document.

For comparison sake, we also evaluated the accuracy of the classification functionality in Readability.js and our modified version of DOM Distiller when applied to the same data set, to judge their ability to predict the final readability state of each document, given its initial HTML. We note that Readability.js is designed to be used this way, but that this prediction point is slightly different than how DOM Distiller is deployed in Chrome. In Chrome, DOM Distiller labels a page as readable based on its final rendered state. This evaluation of DOM Distiller's classification capabilities should therefore not be seen as an evaluation of DOM Distiller's overall quality, but only its ability to achieve the kinds of optimizations sought by `SpeedReader`. Table 2 presents the results of this measurement. As the table shows, `SpeedReader` strictly outperforms the classification capabilities of both DOM Distiller and Readability.js. DOM Distiller has a higher false positive rate than our classifier, while Readability.js has a higher false negative rate.

3.3 Classifier Usability

Problem Statement. Our classifier operates on complete HTML documents, before they are rendered. As a result, the browser is not able to render the document until the entire initial HTML document is fetched. This is different from how current browsers operate, where websites are progressively rendered as each segment of the HTML document is received and parsed. This entails a trade off between rendering delay (since rendering is delayed until the initial HTML document) and network and device resource use (since, when a page is classified as readable, far fewer resources will be fetched and processed).

In this sub-section, we evaluate the rendering delay caused by our classifier, under several representative network conditions. The rendering delay is equal to the time to fetch the entire initial HTML document. We find that the rendering delay imposed is small, especially compared to the dramatic performance improvements delivered when a page is readable (discussed in more detail in Section 4).

Classification Time. We evaluated the rendering delay imposed by our classifier by measuring the time taken to fetch the initial HTML for a page, under different network conditions, and compared it against the time taken for document classification.

Table 3: Measurements of how applicable our readability strategy is under common browser use scenarios.

Measurement	# measured	# readable	% readable
Popular pages	42,986	9,653	22.5%
Unpopular pages	40,908	8,794	21.5%
Total: Random crawl	**83,894**	**18,457**	**22.0%**
Reddit linked	3,035	1,260	41.51%
Twitter linked	494	276	31.2%
RSS linked	506	331	65%
Total: OSN	**4,035**	**1,867**	**46.27%**

First, we determined how long our classifier took to determine if a parsed HTML document was readable. We did so by parsing each HTML string with *myhtml*, a fast, open source, C++ HTML parser [4]. We then measured the execution time taken to extract the relevant features from the document, and to return the predicted label. Our classifier took 2.8 ms on average and 1.9 ms in the median case. Next, we measured the fixed, simulation cost time of serving each web page from a locally hosted web server, which allowed us to account for the fixed overhead in establishing the network connection, and similar unrelated browser book keeping operations. This time was 22.3 ms on average, and 15.5 ms median.

Finally, we selected two network environments to represent different network conditions and device capabilities web users are likely to encounter: a fast, domestic broadband link, with 50 Mbps uplink/downlink bandwidth and 2 ms latency as indicated by a popular network speed testing utility,[5] and a simulated 3G network, created using the operating system's *Network Link Conditioner*.[6] We use a default 3G preset with 780 kbps downlink, 330 kbps uplink, 100 ms packet delay in either direction and no additional packet loss. Downloading the documents on such connection took 1,372 ms / 652 ms (average/median) and 4,023 ms / 2,516 ms for the two cases respectively. Figure 4 summarizes the results of those measurements. Overall, the approximately 2.8 ms taken for an average document classification is a tiny cost compared to just the initial HTML download on reasonably fast connections. It could potentially be further optimized by classifying earlier, i.e. when only a chunk of the initial document is available. Initial tests show promising results, however this adds significant complexity to patching the rendering pipeline and we leave it for future work.

3.4 Applicability to the Web

While subsequent sections will demonstrate the significant performance and privacy improvements provided by SpeedReader, these improvements are only available on a certain type of web document, those that have readable subsets. The performance improvements possible through SpeedReader are therefore bounded by the amount of websites users visit that are readable.

In this subsection, we determine how much of the web is amenable to SpeedReader, by applying our classifier to a sampling of websites, representing different common browsing scenarios. Doing so allows us to estimate the benefits SpeedReader can deliver as well as its relevance to the web. As presented in Table 3, we find that a significant number of visited URLs are readable, suggesting that SpeedReader can deliver significant privacy and performance improvements to users. This subsection continues by describing how we selected URLs in each browsing scenario.

Websites by popularity. We first estimated how many pages hosted on popular and unpopular domains are readable. To do so, we first created two sets of domains, a popular set, consisting of the most popular 5,000 domains, as determined by Alexa, and an unpopular set, comprising a random sample of pages ranked 5,001–100,000. For each domain, we conducted a breadth three, depth three crawl. We first visited the landing page for the domain, and recorded all URLs linked to pages with the same TLD+1 domain. Then we selected up to three URLs from this set, and repeated the above process another time, giving a maximum of 13 URLs per domain, and a total data set of 91,439 pages. The crawl was conducted from AWS IP addresses on 17-20 October 2018.

Social network shared content. We next estimated how much content linked to from online social networks is readable, to simulate a user that spends most of their browsing time on popular online social networks, and generally only browses away to view shared content. We gathered URLs shared from Reddit and Twitter. We gathered links shared on Reddit by using RedditList [32] to obtain top 125 subreddits ranked based on their number of subscribers. We then visited the 25 posts of each popular subreddit and extracted any shared URLs. For Twitter, we extracted shared links from the top 10 worldwide Twitter trends by crawling and extracting shared links from their Tweets.

RSS / feed readers. Finally, we estimated how much content shared from RSS feeds is readable, to simulate a user who finds content mainly through an RSS (or similar) aggregation service. We built a list of RSS-shared content by crawling the Alexa 1K, identifying websites that included RSS feeds, and fetching the five most recent pages of content in each RSS feed.

3.5 Conclusion

In this section we have described how SpeedReader determines whether a page should be rendered in reader mode, based on its initial HTML. We find that SpeedReader outperforms the classification capabilities of existing, deployed reader mode tools. We also find that the overhead imposed by our classification strategy is small and acceptable in most cases, and dwarfed by the performance improvements delivered by SpeedReader, for cases when a page is judged readable.

4 PAGE TREE TRANSDUCTION

This section describes how SpeedReader generates a reader mode presentation of a page, for pages that have been classified as readable. Our evaluation includes three possible reader mode renderings, each presenting a different trade off between amount of media included, performance and privacy improvements.

Generating a reader mode presentation of an HTML document can be thought of as translating one tree structure to another: taking the document represented by the page's initial HTML and generating the document containing a simplified reader mode version. This process of tree mapping is generally known as tree transduction.

[5] speedtest.net - web service that provides analysis of Internet access performance metrics, such as connection data rate and latency

[6] Network Link Conditioner is a tool released by Apple with hardware IO Tools for XCode developer tools to simulate different connection bandwidth, latency and packet loss rates

Table 4: Description of data set used for evaluating the performance implications of different content extraction strategies.

Measurement	Value
Measurement date	17-20 October 2018
# crawled domains	10,000
# crawled pages	91,439
# domains with readable pages	4,931
# readable pages	19,765
% readable pages	**21.62%**

We evaluate Tree transduction by comparing the performance and privacy improvements of the three techniques (Readability.js, DOM Distiller and BoilerPipe) described in detail in Section 2.2.

4.1 Limitations and Bounds

We note that we did not attempt any evaluation of how users perceive or enjoy the reader mode versions of pages rendered by each considered technique. We made this decision for several reasons.

First, two of the techniques (Readability.js and DOM Distiller) are developed and deployed by large browser vendors, with millions or billions of users. We assume that these large companies have conducted their own usability evaluation of their reader mode techniques, and found them satisfactory to users.

Second, the third considered tree transduction technique, Kohlschütter et al's BoilerPipe [24], is an academic work that includes its own evaluation, showing that the technique can successfully extract useful contents from HTML documents. We assume that the authors' evaluation is comprehensive and sufficient, and that their technique can successfully render pages in reader mode presentations. Finally, we are planning to deploy a tree transducer different from existing techniques and a more thorough subjective evaluation of its presentation is left for future study.

4.2 Evaluation Methodology

We compared the performance and privacy improvements achieved through `SpeedReader`'s novel application of three tree transduction techniques: Readability.js, DOM Distiller and BoilerPipe. We conducted this evaluation in three stages.

First, we fetched the HTML of each URL in the random crawl data set outlined in Table 3, again from an AWS IP. The HTML considered here is only the initial HTML response, not the state of the document after script execution. We evaluated whether each of the 91,439 fetched pages that were classified as readable, by applying the `SpeedReader` classifier to each page. We then reduced the data set to the 19,765 pages (21.62%) were readable.

Second, we revisited each URL classified as readable to collect a complete version of the page. To minimize variations in page performance and content during the testing, we collected the "replay archive" for each page using the "Web Page Replay" (WPR) [22] performance tool. WPR is used in Chrome's testing framework for benchmarking purposes and works as a proxy that records network requests or responds to them instead of letting them through to the source depending on whether it works in "record" or "replay" mode.

Finally, we applied each of the three tree transduction techniques to the remaining 19,765 HTML documents, and compared the network, resource use, and privacy characteristics of each transformed page, against the full version of each page. We evaluate performance and privacy characteristics of each page by visiting the URL as replayed from its archive. These findings are described in detail in the next subsections.

Table 5: Performance comparisons of three popular readability tree transducer strategies, as applied to the data set described in Table 4. Values are given as Average, Median. Gain multiplier (×) is calculated for each page load and Average and Median values are reported.

Transducer	Resources (#)		Data (KB)		Memory (MB)		Load Time (ms)	
-	A	M	A	M	A	M	A	M
Default	144	91	2,283	1,461	197	174	1,813	1,069
ReadabilityJS	5	2	186	61	85	79	583	68
Dom Distiller	5	2	186	61	84	79	550	63
BoilerPipe	2	2	101	61	81	77	545	44
Gain (×)								
ReadabilityJS	51	28	84	24	2.4	2.1	20	11
Dom Distiller	52	32	84	24	2.4	2.1	21	12
BoilerPipe	77	48	84	24	2.4	2.1	27	15

We note that using a replay proxy with a snapshot of content often underestimates the costs of a page load. Despite taking care to mitigate the effects of non-determinism by injecting a small script that overrides date functions to use a fixed date and random number generator functions to use a fixed seed and produce a predictable sequence of numbers, it cannot account for all sources of non-determinism. For all requests that the proxy cannot match, it responds with a `Not Found` response. We notice that it results in a small number of requests being missed, primarily those responsible for dynamic ad loading or tracking. It also occasionally interferes with site publisher's custom resource fetching retry logic, where the same request is retried a number of times unsuccessfully, before the entire page load times out and the measurement is omitted.

4.3 Results: Performance

We measured four performance metrics: number of resources requested, amount of data fetched, memory used and page load time. These results are summarized in Table 5 and Figure 5.

We ran all measurements on AWS `m5.large` EC2 instances. For performance measurements, one test was executed at a time, per instance. For each evaluation, we fetched the page from a previously collected record-replay archive, with performance tracing enabled. Once the page was loaded and the performance metrics are recorded, we closed the browser and proxy, and started the next test. No further steps were taken to minimize the likelihood of test VM performance being impacted by interfering workloads on the underlying hardware. For all tests, we used an unmodified Google Chrome browser, version 70.0.3538.67, rendered in Xvfb.[7] Although profiling has overheads of its own [33], in particular for memory use and load times, we used a consistent measurement strategy across all tests, and therefore expect the impact to also be consistent and minor compared to relative performance gains.

We measured a page's load time as the difference between `navigationStart` and `loadEventEnd` events [46] in the main frame (i.e. the time until all sub-resources have been downloaded

[7]While Chrome "headless" mode is available, it effectively employs a different page rendering pipeline with different load time characteristics and memory footprint.

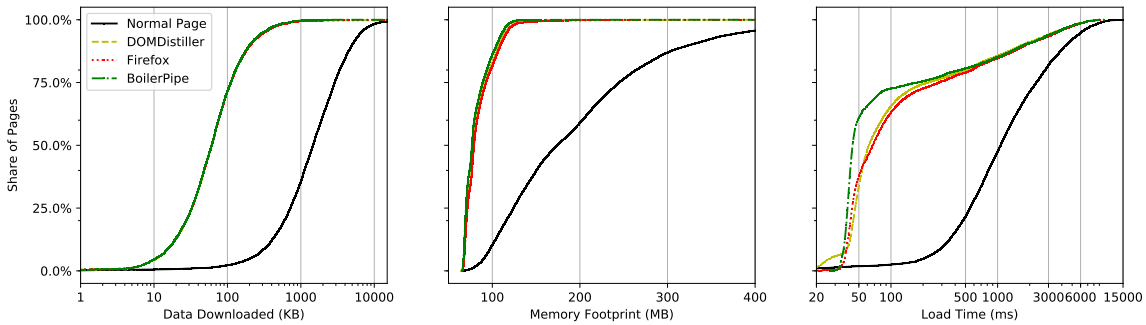


Figure 5: Performance characteristics of the different tree transducer strategies applied, showing the distribution of the key performance metrics.

and the page is fully rendered). Since page content is replayed from a local proxy, network bandwidth and latency variation impact is minimized and the reported load time is a very optimistic figure, especially for bigger pages with more sub-resources as illustrated in Figure 4. Although network cost is still non-zero, the number primarily reflects the time taken to process and render the entire page.

We also recorded the number of resources fetched and the amount of data downloaded during each test. Note that the amount of data downloaded for all of the tree transduction strategies reflects the size of the initial HTML rather than that of the transformed document, as the transformation happens on the client and does not result in additional network traffic. All measured transducers discard the majority of page content (both in page content like text and markup, but also referenced content like images, video files, and JavaScript). Figures 2 and 3 provide an example of how tree transduction techniques simplify page content.

For memory consumption, we measure the overall memory used by the browser and its subprocesses. Google Chrome uses a multi-process model, where each tab and frame may run in a separate process and content of each page also affects what runs in the main browser process. We note that our testing scenario does not consider the case of multiple pages open simultaneously in the same browsing session, as some of the resources are reused. The reported number is therefore that of the entire browser rather than the specific page alone, with some fixed browser runtime overheads.

Memory snapshots are collected with an explicit trigger after the page load is complete with `disabled-by-default-memory-infra` tracing category enabled. Despite including a level of fixed browser memory costs, we still see average memory reduction of up to 2.4× in average or median cases. Overall, depending on the chosen transducer, we show:

- average speedups ranging from 20× to 27×
- average bandwidth savings on the order of 84×
- number of requests is reduced 51× to 77×
- average memory reduction of 2.4×

4.4 Results: Privacy

`SpeedReader` achieves substantial privacy improvements, because it applies the tree transduction step *before* rendering the document, and thus before any requests to third parties have been initiated. The privacy improvements gained by `SpeedReader` are threefold: a reduction in third party requests, a reduction in script execution (an often necessary, though not sufficient, part of fingerprinting online), and a complete elimination of ad and tracking related requests (as labeled by EasyList and EasyPrivacy). This last measure is particularly important, since *92.8%* of the 19,765 readable pages in our data set loaded resources labeled as advertising or tracking related by EasyList and EasyPrivacy [10, 11].

Table 6: Comparisons of the privacy implications of three popular readability tree transducer strategies, as applied to the data set described in Table 4. Values are given as Average and Median values.

Transducer	# third-party		# scripts		Ads & Trackers	
	Avg	Med	Avg	Med	Avg	Med
Default	117	63	83	51	63	24
ReadabilityJS	3	1	0	0	0	0
Dom Distiller	3	1	0	0	0	0
BoilerPipe	1	1	0	0	0	0

This subsection proceeds by both describing how we measured the privacy improvements provided by `SpeedReader`, and the results of that measurement. These findings are presented in Table 6.

We measured the privacy gains provided by `SpeedReader` by first generating reader mode versions of each of the 19,765 readable URLs in our dataset, and counting the number of third parties, script resources, and ad and tracking resources in each generated reader mode page. We determined the number of ad and tracking resources by applying EasyList and EasyPrivacy with an open-source ad-block Node library [21] to each resource URL included in the page. We then compared these measurements to the number of third-parties, script units, and ad and tracking resource requests made in the typical, non-reader mode rendering of each URL.

We found that all three of the evaluated tree transduction techniques dramatically reduced the number of third parties communicated with, and removed all script execution and ad and tracking resource requests from the page. Put differently, `SpeedReader` is able to achieve privacy improvements at least as good, and almost certainly exceeding existing ad and tracking blockers, on readable pages. This claim is based on the observation that ad and tracking blockers do not achieve the same significant reduction in third party communication and script execution as `SpeedReader` achieves.

5 DISCUSSION AND FUTURE WORK

5.1 Reader Mode as a Content Blocker

Most existing reader mode tools function to improve the presentation of page content for readers, by removing distracting content and reformatting text for the browser user's benefit. While the popularity of existing reader modes suggest that this is a beneficial use case, the findings in this work suggest an additional use case for reader modes, blocking advertising and tracking related content.

As discussed in Section 4.4, SpeedReader prevents all ad and tracking related content from being fetched and rendered, as identified by EasyList and EasyPrivacy (Table 6). SpeedReader also loads between 51 and 77 times fewer resources than typical page rendering and reader modes (Table 5), a non-trivial number of which are likely also ad and tracking related. SpeedReader differs fundamentally from existing content blocking strategies. Existing popular tools, like uBlock Origin[20] and AdBlock Plus[15], aim to identify *malicious* or undesirable content, and prevent it from being loaded or displayed; all unlabeled content is treated as desirable and loaded as normal. SpeedReader, and (at last conceptually) reader modes in general, take the opposite approach. Reader modes try to identify *desirable* content, and treat all other page content as undesirable, or, at least, unneeded.

Our results suggest that the reader mode technique can achieve ad and tracking blocking quality *at least* as well as existing content blocking tools, but with dramatic performance improvements. We expect that SpeedReader actually outperforms content blocking tools (as content blockers suffer from false-negative problems, for a variety of reasons), but lack a ground truth basis to evaluate this claim further. We suggest evaluating the content blocking capabilities of reader mode-like tools as a compelling area for future work.

5.2 Comparison to Alternatives

SpeedReader exists among other systems that aim to improve the user experience of viewing content on the web. While a full evaluation of these systems is beyond the scope of this work (mainly because the compared systems have different goals and place different restrictions on users), we note them here for completeness.

AMP. Accelerated Mobile Pages (AMP)[17] is a system developed by Google that improves website performance, in a number of ways. Website authors opt-in to the AMP system by limiting their content to a subset of HTML, JavaScript and CSS functionality, which allows for optimized loading and execution. AMP pages are also served from Google's servers, which provide network level improvements. AMP differs from SpeedReader and other reader mode systems in that users only achieve performance improvements when site authors design their pages for AMP; AMP offers no improvement on existing, traditional websites.

Server-Assisted Rendering. Other browser vendors attempt to improve the user experience by moving page, loading, rendering execution from the client to a server. The client then fetches a rendered version of the page from the server (generally either rendered HTML or as a bitmap). The most popular such system is likely Amazon Silk[1]. While there are significant performance upsides with this thin-client technique, they come with significant downsides too. First, user privacy is harmed, since the rendering-server must manage and observe all client secrets when interacting with the destination server on the client's behalf. Additionally, while the server may be able to improve the loading and rendering of the page, its limited in the kinds of performance improvements it can achieve. Server assisted rendering does not provide any of the presentation simplification or content blocking benefits provided by SpeedReader.

5.3 SpeedReader Deployment Strategies

Always On. SpeedReader as described in this work is designed to be "always on", attempting to provide a readable presentation of every page fetched. Although Safari Reader View also supports an "always on" functionality, it lacks performance and privacy enhancement provided by SpeedReader (Section 2). While this decision maximizes the amount of privacy and performance improvements provided, it entails an overhead while loading each page (Figure 4), which may not be worthwhile in some browsing patterns such as interacting with application-like sites. Additionally, there may be times when users want to maintain a page's interactive functionality (e.g. JavaScript), even when SpeedReader has determined that the page is readable. Ensuring the user's ability to disable SpeedReader would be important in such cases. The system described in this work does not preclude such an option, but only imagines changing the default page loading behavior.[8]

Tree Transduction Improvements. The three evaluated techniques in Section 4, which are adapted from existing tools and research, can provide a reader mode presentation with different performance and privacy improvements. Users of SpeedReader could select which tree transduction technique best suited their needs. However, we expect that ML and similar techniques could be applied to the tree transduction problem, to provide a reader mode presentation that exceeds existing techniques. An improved tree transduction algorithm would achieve equal or greater performance and privacy improvements, while doing a better job of maintaining the meaning and information of the extracted content. We are currently exploring several options in this area, but have found the problem large enough to constitute its own unique work.

6 RELATED WORK

Content Extraction. The problem of removing boilerplate and extracting relevant content from a webpage has been extensively studied. Previous approaches primarily focused on the code structure, visual representation and the link between the two. Lin *et al.* [29] proposed a method to detect content blocks using <TABLE> tags and calculate their entropy to distinguish the informative blocks from the redundant ones. Laber *et al.* [27] proposed a heuristic method for extracting textual sections and title from news articles using <a>, <p> and <title> tags. Other studies have tried to detect *useful* segments in a web page using structural and positional information. Gupta *et al.* [18] introduced a DOM-based method to modify and remove irrelevant DOM nodes to extract the main content. Their

[8] Current browsers and reader modes load all pages in the standard manner, and allow the users to enable a reader mode presentation, while SpeedReader would load pages in the optimized reader mode presentation by default, when possible, and allow users to enable the standard loading behavior.

approach utilized filters to remove DOM nodes with advertisements, and link and text ratio thresholds to remove unwanted table cells. While the proposed rule-based method was simple, it had a poor performance in link rich pages where the main content contained many links. Weninger *et al.* [47] introduced a fast algorithm which calculated the HTML tag ratio of each line to cluster and extract text content. Their algorithm did not perform well on home pages as well as it suffered from high recall and low precision. Cai *et al.* [8] introduced a tag-free vision-based page segmentation algorithm to segment a webpage and extract its web content structure using the link between the visual layout and the content. Fan *et al.* [13] introduced Article Clipper, a web content extractor that leveraged visual cues in addition to HTML tags to extract non-textual and textual content and detect multi-page articles. Their approach underperformed in extracting captions which were links as well as images and captions that were outside of main content.

Heuristic methods are limited by their lack of adaptability. Some have proposed learning based methods to overcome this rigidness. Pasternack and Roth [35] described a semi-supervised algorithm, Maximum Subsequence Segmentation, which tokenized HTML into list of tags, words and symbols, and attempted to classify each block as either "in article" or "out of article" text. Kohlschütter *et al.* [24] developed BoilerPipe to classify text elements using both structural and text features, such as average word length and average sentence length. Sun *et al.* [40] proposed Content Extraction via Text Density (CETD) to extract the text content from pages using a variety of text density measurements. Their method relied on the observation that the amount of text in content sections is large, and the text in boilerplate sections contains more noise and links. Sluban and Grčar [38] introduced an unsupervised and language-independent method for extracting content from streams of HTML pages, by detecting commonalities in page structure. While their method outperformed other open-source content extractor algorithms, it suffered from high memory consumption and poor performance in diverse and small HTML data set.

Wu *et al.* [48] proposed a machine learning model using DOM tree node features such as position, area, font, text and tag properties to select and group content related nodes and their children. In their recent paper, Vogels *et al.* [43] presented an algorithm combining a hidden markov model and a convolutional neural networks (CNNs). Their model first preprocessed an HTML page into a Collapsed DOM (CDOM) tree where each single child parent node was merged with its child. CDOM was then segmented into blocks of main content and boilerplate using sequence labeling of DOM leaves. The features were then used to train two CNNs, obtain potentials and finally find the optimal labeling. Their approach outperformed previous studies on the CleanEval benchmark [3].

Web Complexity. While content extraction has attracted much attention in the scientific literature, fewer studies are conducted to understand website complexity and its impact on page load time and user experience. Gibson *et al.* [14] analyzed webpage template evolution using site-level template detection algorithms and found that templates, with little raw content value, represented 40-50% of the data on the Web and the rate continued to grow at a rate about 6% per year. Butkiewicz *et al.* [7] showed that modern websites, regardless of their popularity, were complex and such complexity could affect user experience. Moreover, their analysis demonstrated that the number of loaded objects and servers could indicate page load time, and both numbers were significant in News websites.

Performance and User Experience. While complexity of webpages can affect page load time, their visual complexity can impact user experience. Harper *et al.*showed that visual complexity in webpages, defined as diversity, density, and positioning of the elements, could increase cognitive load [19] and even have detrimental cognitive and emotional impact on users [41]. In many websites, online advertisements are the only source of income. Nonetheless, online ads, especially intrusive ads, have usability consequences [6]. As Pujol *et al.* [36] observed, 22% of the most active users of a major European ISP use Adblock Plus. As a result, providing the main content in a clutter free page, such as Reader Mode, not only decreases the complexity of a page, but also preserves privacy by limiting the number of requests for third-party services and trackers [12, 25] as well as improves user experience.

7 CONCLUSION

The modern web's progress has led us to the point far beyond Hypertext Markup for document discovery, to having full-fledged, media-rich experiences and dynamic applications. With this growth in capability, there has been a growth in page "bloat", making pages expensive to load, and bringing with it ubiquitous advertising and tracking. In this work, we propose `SpeedReader` as an approach broadening the applicability of "reader mode" browser features to deliver huge improvements to the end-user browsing experience.

Unique among reader mode tools, `SpeedReader` determines if a page is readable based only on the page's initial HTML, before the HTML is parsed and rendered, and before sub-resources are fetched. Our classifier can classify within 2 ms and with 91% accuracy, which makes it practical as an always-on part of the rendering pipeline to transform all suitable pages at load time. We find that `SpeedReader` is widely applicable, and can deliver performance and privacy improvements to 22% of pages on popular and unpopular websites, and a larger proportion of pages linked to from online social networks like Reddit (42%) and Twitter (31%). Since `SpeedReader` makes its modifications before sub-resources are fetched, it uses 84× less network than traditional page rendering (and current reader mode techniques). This results in page load time improvements, important in a range of scenarios from poor connectivity or low-end devices, to expensive data connectivity or simply wanting a clean and simple interaction with primarily textual content. `SpeedReader` also delivers page loading speedups of 20× - 27× and average memory reduction of 2.4×, while maintaining a pleasant, reader mode style user experience. Finally, when `SpeedReader` was applied to 19,765 readable webpages, it prevented 100% of advertising and tracking related resources from being fetched (as labeled by EasyList and EasyPrivacy).

8 ACKNOWLEDGEMENT

This research was supported by Brave Software. We would like to thank David Temkin for his practical feedback and helpful comments on the project. We also would like to thank anonymous reviewers for their time and effort in reviewing this paper.

REFERENCES

[1] Amazon. [n. d.]. Amazon Silk Documentation. docs.aws.amazon.com/silk/index.html

[2] Arc90. [n. d.]. Readability - An Arc90 Lab Experiment. http://ejucovy.github.io/readability/

[3] Marco Baroni, Francis Chantree, Adam Kilgarriff, and Serge Sharoff. 2008. Cleaneval: a Competition for Cleaning Web Pages.. In *LREC*.

[4] Alexander Borisov. [n. d.]. myHTML - Fast C/C++ HTML 5 Parser. Using threads. https://github.com/lexborisov/myhtml

[5] Anna Bouch, Allan Kuchinsky, and Nina Bhatti. 2000. Quality is in the Eye of the Beholder: Meeting Users' Requirements for Internet Quality of Service. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '00)*. ACM, New York, NY, USA, 297–304. https://doi.org/10.1145/332040.332447

[6] Giorgio Brajnik and Silvia Gabrielli. 2010. A review of online advertising effects on the user experience. *International Journal of Human-Computer Interaction* 26, 10 (2010), 971–997.

[7] Michael Butkiewicz, Harsha V. Madhyastha, and Vyas Sekar. 2011. Understanding Website Complexity: Measurements, Metrics, and Implications. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference (IMC '11)*. ACM, New York, NY, USA, 313–328. https://doi.org/10.1145/2068816.2068846

[8] Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. 2003. VIPS: a Vision-based Page Segmentation Algorithm. (November 2003), 28. https://www.microsoft.com/en-us/research/publication/vips-a-vision-based-page-segmentation-algorithm/

[9] Mozilla Corporation. 2018. Readability.js. https://github.com/mozilla/readability

[10] EasyList. 2018. About EasyList. https://easylist.to/pages/about.html

[11] EasyList. 2018. EasyList Github repository. https://github.com/easylist/easylist

[12] Steven Englehardt and Arvind Narayanan. 2016. Online Tracking: A 1-million-site Measurement and Analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*. ACM, New York, NY, USA, 1388–1401. https://doi.org/10.1145/2976749.2978313

[13] Jian Fan, Ping Luo, Suk Hwan Lim, Sam Liu, Parag Joshi, and Jerry Liu. 2011. Article Clipper: A System for Web Article Extraction. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*. ACM, New York, NY, USA, 743–746. https://doi.org/10.1145/2020408.2020525

[14] David Gibson, Kunal Punera, and Andrew Tomkins. 2005. The Volume and Evolution of Web Page Templates. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web (WWW '05)*. ACM, New York, NY, USA, 830–839. https://doi.org/10.1145/1062745.1062763

[15] Eyeo GmbH. 2018. Adblock Plus. https://adblockplus.org/

[16] Utkarsh Goel, Moritz Steiner, Mike P Wittie, Martin Flack, and Stephen Ludin. 2017. Measuring What is Not Ours: A Tale of 3rd Party Performance. In *International Conference on Passive and Active Network Measurement*. Springer, 142–155.

[17] Google. [n. d.]. Accelerated Mobile Pages Project. https://www.ampproject.org

[18] Suhit Gupta, Gail Kaiser, David Neistadt, and Peter Grimm. 2003. DOM-based Content Extraction of HTML Documents. In *Proceedings of the 12th International Conference on World Wide Web (WWW '03)*. ACM, New York, NY, USA, 207–214. https://doi.org/10.1145/775152.775182

[19] Simon Harper, Eleni Michailidou, and Robert Stevens. 2009. Toward a Definition of Visual Complexity As an Implicit Measure of Cognitive Load. *ACM Trans. Appl. Percept.* 6, 2, Article 10 (March 2009), 18 pages. https://doi.org/10.1145/1498700.1498704

[20] Raymond Hill. 2018. uBlock Origin - An efficient blocker for Chromium and Firefox. Fast and lean. https://github.com/gorhill/uBlock

[21] Brave Software Inc. 2018. Brave Ad Block. https://github.com/brave/ad-block

[22] Google Inc. [n. d.]. Catapult - Web Page Replay. https://github.com/catapult-project/catapult.git

[23] Google Inc. 2018. DOM Distiller. https://github.com/chromium/dom-distiller

[24] Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate Detection Using Shallow Text Features. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM '10)*. ACM, New York, NY, USA, 441–450. https://doi.org/10.1145/1718487.1718542

[25] Balachander Krishnamurthy and Craig Wills. 2009. Privacy Diffusion on the Web: A Longitudinal Perspective. In *Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. ACM, New York, NY, USA, 541–550. https://doi.org/10.1145/1526709.1526782

[26] Deepak Kumar, Zane Ma, Zakir Durumeric, Ariana Mirian, Joshua Mason, J. Alex Halderman, and Michael Bailey. 2017. Security Challenges in an Increasingly Tangled Web. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 677–684. https://doi.org/10.1145/3038912.3052686

[27] Eduardo Sany Laber, Críston Pereira de Souza, Iam Vita Jabour, Evelin Carvalho Freire de Amorim, Eduardo Teixeira Cardoso, Raúl Pierre Rentería, Lúcio Cunha Tinoco, and Caio Dias Valentim. 2009. A Fast and Simple Method for Extracting Relevant Content from News Webpages. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM '09)*. ACM, New York, NY, USA, 1685–1688. https://doi.org/10.1145/1645953.1646204

[28] Timothy Libert. 2015. Exposing the Invisible Web: An Analysis of Third-Party HTTP Requests on 1 Million Websites. *International Journal of Communication* 9, 0 (2015). https://ijoc.org/index.php/ijoc/article/view/3646

[29] Shian-Hua Lin and Jan-Ming Ho. 2002. Discovering Informative Content Blocks from Web Documents. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02)*. ACM, New York, NY, USA, 588–593. https://doi.org/10.1145/775047.775134

[30] Georg Merzdovnik, Markus Huber, Damjan Buhov, Nick Nikiforakis, Sebastian Neuner, Martin Schmiedecker, and Edgar Weippl. 2017. Block me if you can: A large-scale study of tracker-blocking tools. In *Security and Privacy (EuroS&P), 2017 IEEE European Symposium on*. IEEE, 319–333.

[31] Georg Merzdovnik, Markus Huber, Damjan Buhov, Nick Nikiforakis, Sebastian Neuner, Martin Schmiedecker, and Edgar Weippl. 2017. Block Me if You Can: A Large-Scale Study of Tracker-Blocking Tools. *Proceedings - 2nd IEEE European Symposium on Security and Privacy, EuroS and P 2017* (2017), 319–333. https://doi.org/10.1109/EuroSP.2017.26

[32] mikesizz. [n. d.]. RedditList - Tracking the top 5000 subreddits. http://redditlist.com/

[33] Thomas Nagele. 2015. *Client-side performance profiling of JavaScript for web applications*. Master Thesis. Radboud University Nijmegen.

[34] Nick Nikiforakis, Luca Invernizzi, Alexandros Kapravelos, Steven Van Acker, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2012. You Are What You Include: Large-scale Evaluation of Remote Javascript Inclusions. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS '12)*. ACM, New York, NY, USA, 736–747. https://doi.org/10.1145/2382196.2382274

[35] Jeff Pasternack and Dan Roth. 2009. Extracting Article Text from the Web with Maximum Subsequence Segmentation. In *Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. ACM, New York, NY, USA, 971–980. https://doi.org/10.1145/1526709.1526840

[36] Enric Pujol, Oliver Hohlfeld, and Anja Feldmann. 2015. Annoyed Users: Ads and Ad-Block Usage in the Wild. In *Proceedings of the 2015 Internet Measurement Conference (IMC '15)*. ACM, New York, NY, USA, 93–106. https://doi.org/10.1145/2815675.2815705

[37] Paruj Ratanaworabhan, Benjamin Livshits, and Benjamin G. Zorn. 2010. JSMeter: Comparing the Behavior of JavaScript Benchmarks with Real Web Applications. In *Proceedings of the 2010 USENIX Conference on Web Application Development (WebApps'10)*. USENIX Association, Berkeley, CA, USA, 3–3. http://dl.acm.org/citation.cfm?id=1863166.1863169

[38] Borut Sluban and Miha Grčar. 2013. URL tree: efficient unsupervised content extraction from streams of web documents. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management (CIKM '13)*. ACM, New York, NY, USA, 2267–2272. https://doi.org/10.1145/2505515.2505654

[39] Peter Snyder, Lara Ansari, Cynthia Taylor, and Chris Kanich. 2016. Browser Feature Usage on the Modern Web. In *Proceedings of the 2016 Internet Measurement Conference (IMC '16)*. ACM, New York, NY, USA, 97–110. https://doi.org/10.1145/2987443.2987466

[40] Fei Sun, Dandan Song, and Lejian Liao. 2011. DOM Based Content Extraction via Text Density. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '11)*. ACM, New York, NY, USA, 245–254. https://doi.org/10.1145/2009916.2009952

[41] Alexandre N. Tuch, Javier A. Bargas-Avila, Klaus Opwis, and Frank H. Wilhelm. 2009. Visual complexity of websites: Effects on users' experience, physiology, performance, and memory. *International Journal of Human-Computer Studies* 67, 9 (2009), 703 – 715. https://doi.org/10.1016/j.ijhcs.2009.04.002

[42] Antoine Vastel, Peter Snyder, and Benjamin Livshits. 2018. Who Filters the Filters: Understanding the Growth, Usefulness and Efficiency of Crowdsourced Ad Blocking. (2018). http://arxiv.org/abs/1810.09160

[43] Thijs Vogels, Octavian-Eugen Ganea, and Carsten Eickhoff. 2018. Web2Text: Deep Structured Boilerplate Removal. In *European Conference on Information Retrieval*. Springer, 167–179.

[44] Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall. 2013. Demystifying Page Load Performance with WProf. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*. USENIX, Lombard, IL, 473–485. https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/wang_xiao

[45] Xiao Sophia Wang, Arvind Krishnamurthy, and David Wetherall. 2016. Speeding up Web Page Loads with Shandian. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*. USENIX Association, Santa Clara, CA, 109–122. https://www.usenix.org/conference/nsdi16/technical-sessions/presentation/wang

[46] Zhiheng Wang. 2012. *Navigation Timing*. W3C Recommendation. W3C. http://www.w3.org/TR/2012/REC-navigation-timing-20121217/.

[47] Tim Weninger, William H. Hsu, and Jiawei Han. 2010. CETR: Content Extraction via Tag Ratios. In *Proceedings of the 19th International Conference on World Wide Web (WWW '10)*. ACM, New York, NY, USA, 971–980. https://doi.org/10.1145/1772690.1772789

[48] Shanchan Wu, Jerry Liu, and Jian Fan. 2015. Automatic Web Content Extraction by Combination of Learning and Grouping. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1264–1274. https://doi.org/10.1145/2736277.2741659

Externalities and Fairness

Mohammad Ghodsi
Sharif University of Technology, IPM - Institute for Research in Fundamental Sciences
Tehran, Iran
ghodsi@sharif.edu

Hamed Saleh
University of Maryland
College Park, Matyland, USA
hameelas@gmail.com

Masoud Seddighin
Sharif University of Technology
Tehran, Iran
mseddighin@ce.sharif.edu

ABSTRACT

One of the important yet insufficiently studied subjects in fair allocation is the externality effect among agents. For a resource allocation problem, externalities imply that the share allocated to an agent may affect the utilities of other agents.

In this paper, we conduct a study of fair allocation of indivisible goods when the externalities are not negligible. Inspired by the models in the context of network diffusion, we present a simple and natural model, namely *network externalities*, to capture the externalities. To evaluate fairness in the network externalities model, we generalize the idea behind the notion of maximin-share (MMS) to achieve a new criterion, namely, *extended-maximin-share* (EMMS). Next, we consider two problems concerning our model.

First, we discuss the computational aspects of finding the value of EMMS for every agent. For this, we introduce a generalized form of partitioning problem that includes many famous partitioning problems such as maximin, minimax, and leximin. We further show that a 1/2-approximation algorithm exists for this partitioning problem.

Next, we investigate on finding approximately optimal EMMS allocations, i.e., allocations that guarantee each agent a utility of at least a fraction of his extended-maximin-share. We show that under a natural assumption that the agents are α-self-reliant, an $\alpha/2$-EMMS allocation always exists. The combination of this with the former result yields a polynomial-time $\alpha/4$-EMMS allocation algorithm.

CCS CONCEPTS

• **Theory of computation → Approximation algorithms analysis**; • **Applied computing → Economics**; • **Networks →** *Network resources allocation*; *Network economics.*

KEYWORDS

Fairness; maximin-share; approximation; externalities; networks

ACM Reference Format:
Mohammad Ghodsi, Hamed Saleh, and Masoud Seddighin. 2019. Externalities and Fairness. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313670

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313670

1 INTRODUCTION

Consider a scenario where there is a collection of m indivisible goods that are to be divided amongst n agents. For a properly chosen notion of fairness, we desire our division to be fair. Motivating examples are dividing the inherited wealth among heirs, dividing assets of a bankrupt company among creditors, divorce settlements, task assignments, etc.

Fair division has been a central problem in Economic Theory. This subject was first introduced in 1948 by Steinhaus [28] in the Polish school of mathematics. The primary model used the metaphor of cake to represent a single divisible resource that must be divided among a set of agents. *Proportionality* is one of the most well-studied notions defined to evaluate the fairness of a cake division protocol. An allocation of a cake to n agents is proportional if every agent feels that his allocated share is worth at least $1/n$ of the entire cake. Despite many positive results regarding proportionality in cake-cutting, moving beyond the metaphor of cake, the problem becomes more subtle. For example, when the resource is a set of indivisible goods, a proportional allocation is not guaranteed to exist for all instances[1].

For allocation of indivisible goods, Budish [10] introduced a new fairness criterion, namely *maximin-share*, that attracted a lot of attention in recent years [3, 6, 7, 14, 22, 27, 29]. This notion is a relaxation of proportionality for the case of indivisible items. Assume that we ask agent i to distribute the items into n bundles, and take the bundle with the minimum value. In such a situation, agent i distributes the items in a way that maximizes the value of the minimum bundle. The maximin-share value of agent i is equal to the value of the minimum bundle in the best possible distribution. Formally, the maximin-share of agent i, denoted by MMS_i, for a set $\mathcal{M}$ of items and n agents is defined as

$$\max_{P=\langle P_1,P_2,\ldots,P_n\rangle\in\Pi} \min_j V_i(P_j),$$

where Π is the set of all partitions of $\mathcal{M}$ into n bundles, and $V_i(P_j)$ is the value of bundle P_j to agent i. In a nice paper, Procaccia and Wang [27] show that in some instances, no allocation can guarantee maximin-share to all the agents, but an allocation guaranteeing each agent 2/3 of his maximin-share always exists. This factor has been recently improved to 3/4 by Ghodsi et al. [14].

Our goal in this paper is to generalize maximin-share to the environment with externalities. Roughly speaking, externalities are the influences (costs or benefits) incurred by other parties. The consequences of various economic activities on third parties are studied by both economists and computer scientists. For resource allocation

[1] For example, consider the case that there are two agents and the resource is a single indivisible item.

problems, externalities imply that the bundle allocated to an agent may affect the utility of the other agents. These externalities can be either positive or negative. In this work, we assume that the externalities are **non-negative**, which is a common assumption in the literature [9, 17, 24].

There are many reasons to consider externalities in an allocation problem. The goods to be divided might exhibit network effects. For example, the value of an XBox to an agent increases as more of his friends also own an XBox, since they can play online. Many merit goods generate positive consumption externalities. In healthcare, individuals who are vaccinated entail positive externalities to other agents around them, since they decrease the risk of contraction. Furthermore, allocating a good to an agent might exert externalities on his friends since they can borrow it.

We wish to take one step toward understanding the impact of externalities in the fair allocation of indivisible items. The messages of our paper can be summarized as follows. **First**, considering the externalities is important: the value of EMMS (the generalization we define to adapt MMS to the environment with externalities) and MMS might have a large gap. In fact, we show that even a small amount of influence can result in an unbounded gap between these two notions. Thus, when the externalities are not negligible, methods that guarantee MMS to all the agents might no longer be applicable. **Second**, with respect to our model and fairness notion, we can approximately maintain fairness in the environment with externalities. In the next section, we give a more detailed description of our results and techniques.

1.1 Our Contribution

We start by proposing a general model to capture the externalities in a fair allocation problem under the additive assumption. Although we present some of our results with respect to this general model, our focus is on a more restricted model, namely *network externalities*, where the influences imposed by the agents can be represented by a weighted directed graph. This model is inspired by the well-studied linear-threshold model in the context of network diffusion.

We suggest the *extended-maximin-share* notion (EMMS) to adapt maximin-share to the environment with externalities. Similar to maximin-share, our extension is motivated by the maximin strategy in a cut-and-choose game. We discuss two aspects of our notion.

First, we discuss the hardness of computing the value of EMMS_i, where EMMS_i is the extended-maximin-share of agent i. We introduce a generalized form of the partition problem that includes many famous partition problems such as maximin, minimax, and leximin partitioning problems. This generalized problem is NP-hard due to a trivial reduction from the classic partition problem. In Section 3, we propose a 1/2-approximation algorithm for computing EMMS_i (Theorem 3.2). In fact, we show that the LPT method, which is a famous greedy algorithm in the context of job scheduling, guarantees 1/2-approximation for the general partition problem. We also reveal several structural properties of such partitions.

Second, we consider the approximate α-EMMS allocations, that is, allocations that guarantee every agent a utility of at least a fraction α of his extended-maximin-share. We define the property of β-self-reliance and show that when the agents are β-self-reliant, there exists an allocation that guarantees every agent i a utility of at least $\beta/2$-EMMS_i (Theorem 4.2). This is our most technically involved result. The basic idea behind our method is as follows: every agent has an expectation value which estimates the utility that he must gain through the algorithm. Initially, the expectation value of agent i is at least $\mathsf{EMMS}_i/2$. In every step of the algorithm, we choose an agent and allocate him a bundle with a value at least as his expectation value. Based on this allocation, we decrease the expectation value of the remaining agents. The process of updating the expectation values is a fairly complex process which we describe in Section 4.2. The analysis of the algorithm is technically involved and heavily exploits the structural properties of the general partitioning problem. Finally, the combination of our existential proof with the 1/2-approximation algorithm for computing EMMS yields a polynomial time $\beta/4$-EMMS allocation algorithm.

1.2 Related Work

Maximin-share has received a lot of attention over the past few years [1–5, 8, 13–15, 18, 22, 27, 29]. The counter-example suggested by Procaccia and Wang [27] refutes the existence of any allocation with the maximin-share guarantee. In addition, Procaccia and Wang propose the first approximation algorithm that guarantees each agent 2/3 of his maximin-share. Recently, Ghodsi et al. [14] improve the approximation ratio to 3/4. For the special case of 3 agents, Procaccia and Wang [27] prove that guaranteeing 3/4 of every agent's maximin-share is always possible. Kurokawa et al. [22] show that when the valuations are drawn at random, an allocation with maximin-share guarantee exists with a high probability, and it can be found in polynomial time.

Other studies generalize maximin-share for different settings. For example, Farhadi et al. [13] generalize maximin-share to the case of asymmetric agents with different entitlements. They introduce the *weighted-maximin-share* (WMMS) criterion and propose an allocation algorithm with a 1/2-WMMS guarantee. Suksompong [29] considers the case that the items must be allocated to groups of agents. Gourvès and Monnot [15] extend maximin-share to the case that the goods collectively received by the agents satisfy a matroidal constraint and propose an allocation with a 1/2 maximin-share guarantee. Ghodsi et al. [14] and Barman et al. [7] consider maximin share guarantee for general valuation functions such as submodular, XOS, and subadditive set functions.

In recent years, considering externalities for different economic activities has received an increasing attention in computer science [4, 8, 9, 17, 19, 20, 23–25]. For example, Haghpanah et al. [17] study auction design in the presence of externalities. Pataki et al. [26] study the influences of the externalities on the Pareto optimal allocation of indivisible goods. Velez studies the fair allocation of indivisible goods and money with externalities [30]. In a more related work, Brânzei et al. [9] consider externalities in the cake cutting problem. They introduce a model for cake cutting with externalities and generalize classic fairness criteria to the case with externalities. Following this work, Li et al. [24] study truthful and fair methods for allocating a divisible resource with externalities.

2 MODEL

Throughout the paper, we assume $\mathcal{M}$ is a set of m indivisible items that must be fairly allocated to a set $\mathcal{N} = [n]$ of agents, where $[n]$

	b_1	b_2	b_3
1	1,5,2	4,6,1	7,1,0
2	3,8,5	1,8,7	4,6,9
3	1,5,3	6,5,7	5,4,1

Table 1: An instance in the general externalities model

denotes the set $\{1, 2, \ldots, n\}$. We introduce our model in Section 2 and our fairness criteria in Section 2.2.

2.1 Modeling the Externalities

We start by proposing a general model to represent the externalities. In the **general externalities** model, we suppose that for every item b, $V_{j,i}(b)$ reflects the utility that agent i receives by allocating b to agent j. In this model, there is no restriction on the value of $V_{j,i}(.)$. Total utility of agent i for an allocation is defined to be the sum of utilities he receives by every item, i.e.,

$$U_i = \sum_{b \in \mathcal{M}} V_{j_b,i}(b)$$

where j_b is the index of the agent whose b is allocated. For example, consider the instance with 3 items and 3 agents demonstrated in Figure 1. In this figure, for each agent i and item b_j, three values are given, where the k'th value shows the utility gained by agent i, if we allocate b_j to agent k. For this instance, if we allocate each item b_i to agent i, then we have:

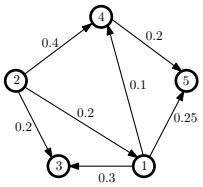
$$U_1 = 1 + 6 + 0 = 7,$$
$$U_2 = 3 + 8 + 9 = 20,$$
$$U_3 = 1 + 5 + 1 = 7.$$

The main focus of the paper is on a more restricted model where the externalities are due to the relationships between agents. For example, friends may share their items with a probability which is a function of their relationship. In this model, each agent i has a valuation function V_i, where for each bundle S, $V_i(S)$ represents the happiness of agent i, if we allocate S to him. We assume that $V_i(.)$ is additive, i.e., for every bundle S,

$$V_i(S) = \sum_{b_j \in S} V_i(b_j).$$

In addition, we consider a directed weighted graph G where for every pair of vertices i and j, the weight of edge $(\overrightarrow{j,i})$, denoted by $w_{j,i}$, represents the influence ratio of agent j on agent i. We refer such a graph as *influence graph*. If we allocate item b to agent j, the utility gained by agent i from this allocation would be $V_i(\{b\}) \cdot w_{j,i}$. For convenience, we define $w_{i,i} = 1$ for every agent i and $w_{j,i} = 0$, for every agent j which is not connected via a directed edge to i.

For instance, consider the example illustrated in Figure 1. For the allocation that allocates every item b_i to agent i ($1 \le i \le 5$), the total utility of the agents would be



	b_1	b_2	b_3	b_4	b_5
1	5	5	2	8	4
2	9	2	5	2	1
3	2	5	3	5	6
4	8	2	4	9	8
5	8	0	2	7	4

Figure 1: An instance in the network externalities model

$$U_1 = 5 + 5 \cdot 0.2 = 6,$$
$$U_2 = 2,$$
$$U_3 = 3 + 5 \cdot 0.2 + 2 \cdot 0.3 = 4.6,$$
$$U_4 = 9 + 8 \cdot 0.1 + 2 \cdot 0.4 = 10.6,$$
$$U_5 = 4 + 8 \cdot 0.25 + 7 \cdot 0.2 = 7.4.$$

We call such a model the **network externalities** model.

Definition 2.1. For the network externalities model, we say agent i is β-self-reliant, if

$$\frac{1}{\sum_{1 \le j \le n} w_{j,i}} \ge \beta.$$

For example in Figure 1, agent 4 is 2/3-self-reliant and agent 2 is 1-self-reliant. In real-world situations, we expect β to be a value close to 1. In other words, we expect an agent to be far more satisfied, when we allocate an item to him rather than allocating it to the other parties. However, being β-self-reliant for $\beta \simeq 1$ doesn't mean that we can ignore the externalities. We discuss more on this in Section 4.

Definition 2.2. For every agent i, we define the *influence vector* of agent i, denoted by $x_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,n}]$ as the vector representing the influences of the agents on agent i in the influence graph, in a non-decreasing order.

As mentioned, we suppose that for every agent i, we have $w_{i,i} = 1$. As an example, for the graph in Figure 1, we have

$$x_1 = [0, 0, 0, 0.2, 1],$$
$$x_2 = [0, 0, 0, 0, 1],$$
$$x_3 = [0, 0, 0.2, 0.3, 1],$$
$$x_4 = [0, 0, 0.1, 0.4, 1],$$
$$x_5 = [0, 0, 0.2, 0.25, 1].$$

We emphasize that throughout the paper, our assumption is that the externalities and the valuations are all non-negative.

2.2 Extended Maximin Share

In this paper, we introduce the extended maximin-share (EMMS) criterion. As mentioned, the maximin-share (MMS) notion was introduced by Budish [10] as a fairness criterion in the division of indivisible items. In Section 1, we gave a formal definition of this notion. The intriguing fact about MMS solution is that it can be motivated by the "cut and choose" game. In this game, an agent divides the items into n bundles and lets other agents choose their bundle first. In the worst-case scenario, the least valued bundle remains, and hence the maximin strategy is to divide the items in a way that the minimum bundle is as attractive as possible.

To extend maximin-share to the case of the agents with externalities, again we consider the worst-case scenario in a "cut and choose" game which incorporates externalities. Suppose that an agent divides the items into n bundles, and other agents somehow distribute these bundles (one bundle to each agent). The maximin strategy of this agent is to divide the items in a way that maximizes his utility in the worst possible scenario (a scenario that minimizes his utility). We define the *extended-maximin-share* of agent i as his maximin value in a "cut and choose" game with externalities.

Formally, let $P = \langle P_1, P_2, \ldots, P_n \rangle$ be a partition of $\mathcal{M}$ into n bundles with the following properties:

- For every i, we have $P_i \neq \emptyset$.
- For every i, j where $i \neq j$, we have $P_i \cap P_j = \emptyset$.
- $\cup_i P_i = \mathcal{M}$.

Furthermore, let $\mathcal{A} : P \rightarrow [n]$ be an allocation function that allocates every set P_i to agent $\mathcal{A}(P_i)$ (one set to each agent). For brevity, when P is clear from the context, we use $\mathcal{A}_i$ instead of $\mathcal{A}(P_i)$ to refer to the agent whom P_i is allocated to. Since exactly one bundle must be allocated to each agent, $\mathcal{A}$ is a bijection. The utility of agent i for an allocation $\mathcal{A}$ is:

$$U_i(\mathcal{A}) = \sum_j V_{\mathcal{A}_j, i}(P_j).$$

For example, consider the instance shown in Figure 2 and let

$$\mathcal{P} = \langle \mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3 \rangle = \langle \{b_1, b_3\}, \{b_2, b_5\}, \{b_4\} \rangle$$

be a partition of the items into three bundles. One possible allocation is to allocate every bundle $\mathcal{P}_i$ to agent i. For this allocation (namely, $\mathcal{A}_\mathcal{P}$), we have

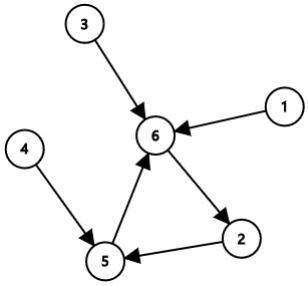
$$\begin{aligned} U_1(\mathcal{A}_\mathcal{P}) &= (5+2) + (6+4) \cdot 0.8 = 15, \\ U_2(\mathcal{A}_\mathcal{P}) &= (9+5) \cdot 0.65 + (2+1) = 12.1, \\ U_3(\mathcal{A}_\mathcal{P}) &= (2+3) \cdot 0.6 + (0+6) \cdot 0.2 + 5 = 9.2. \end{aligned}$$

The worst allocation of P for agent i, denoted by $\mathcal{W}_i(P)$, is the allocation of P that minimizes the utility of agent i:

$$\mathcal{W}_i(P) = \arg \min_{\mathcal{A} \in \Omega_P} U_i(\mathcal{A}),$$

where Ω_P is the set of all $n!$ different allocations of P. For example, a worst allocation of $\mathcal{P}$ for agent 1 is to allocate $\mathcal{P}_3$ to agent 1, $\mathcal{P}_2$ to agent 3, and $\mathcal{P}_1$ to agent 2. The utility of agent 1 for this allocation is $3 + 7 \cdot 0.8 = 8.6$. Similarly, the best allocation of P is defined as:

$$\mathcal{B}_i(P) = \arg \max_{\mathcal{A} \in \Omega_P} U_i(\mathcal{A}).$$



	b_1	b_2	b_3	b_4	b_5
1	5	6	2	3	4
2	9	2	5	2	1
3	2	0	3	5	6

Figure 2: Another instance in the network externalities model

Again, the best allocation of $\mathcal{P}$ for agent 1 is to allocate $\mathcal{P}_2$ to agent 1, $\mathcal{P}_1$ to agent 2, and $\mathcal{P}_3$ to agent 3 which results in the utility of $10 + 7 \cdot 0.8 = 15.6$ for agent 1.

Finally, the extended-maximin-share of agent i, denoted by EMMS_i, is defined as:

$$\mathsf{EMMS}_i = \max_{P \in \Pi} U_i(\mathcal{W}_i(P)),$$

where Π is the set of all partitions of $\mathcal{M}$ into n non-empty subsets. We also define the *optimal partition* of $\mathcal{M}$ for agent i, denoted by O_i, as the partition that determines the value of EMMS_i,

$$O_i = \arg \max_{P \in \Pi} U_i(\mathcal{W}_i(P)).$$

Throughout the paper, when speaking of the network externalities model, we assume that the bundles in $O_i = \langle O_{i,1}, O_{i,2}, \ldots, O_{i,n} \rangle$ are sorted by their non-increasing values for agent i, i.e., for all j, $V_i(O_{i,j}) \geq V_i(O_{i,j+1})$. Thus, we have [2]

$$\mathsf{EMMS}_i = \sum_{1 \leq j \leq n} V_i(O_{i,j}) \cdot x_{i,j}.$$

It can be observed that for agent 1 in Figure 2, we have

$$O_1 = \langle \{b_1, b_3\}, \{b_4, b_5\}, \{b_2\} \rangle$$

and

$$\begin{aligned} \mathsf{EMMS}_1 &= U_1(\mathcal{W}_i(O_i)) \\ &= 7 \cdot 0.8 + 6 = 11.6. \end{aligned}$$

When the agent is clear from the context, for any partition P we use P_j to refer to the j'th valuable bundle of P for that agent.

Finally, an α-EMMS fair allocation problem with the externalities is defined as follows: is there an allocation such that every agent i receives a utility of at least $\alpha \cdot \mathsf{EMMS}_i$?

2.3 Average allocations

We also introduce another fairness notion for the extension of proportionality to the environments with externalities, namely *average-share*.

[2] If it is not clear why this equation holds, see Section 3.

Definition 2.3 (average-share). The average value of item b for agent i, denoted by $\overline{V}_i(\{b\})$, is defined as $\sum_j V_{j,i}(\{b\})/n$. The average-share of agent i is $\overline{V}_i(\mathcal{M}) = \sum_{b\in\mathcal{M}} \overline{V}_i(\{b\})$. Furthermore, an allocation is said to be average, if the total utility of every agent from this allocation would be at least as his average-share.

For the case of network externalities model, we have

$$\overline{V}_i(\mathcal{M}) = V_i(\mathcal{M})/n \cdot \sum_j w_{j,i}. \tag{1}$$

For the agents in Figure 2, we have

$$\overline{V}_1(\mathcal{M}) = (1.8/3) \cdot 20 = 12$$
$$\overline{V}_2(\mathcal{M}) = (1.65/3) \cdot 19 = 10.45$$
$$\overline{V}_3(\mathcal{M}) = (1.8/3) \cdot 16 = 9.6.$$

Average-share plays an important role in analyzing our algorithm for finding the (approximately) optimal partitions in Section 3. As we show in Section 2.4, this notion is stronger than extended maximin-share. However, no approximation of this notion can be satisfied even for very simple scenarios.

2.4 Basic Observations

In Section 2, we introduced two notions: average-share, and extended-maximin-share. For a better understanding of these notions, here we briefly compare them in the **general externalities** model. First, in Lemma 2.4, we prove that average share is stronger than extended maximin-share.

Lemma 2.4. *Average-share implies extended-maximin-share.*

Proof. Let $\mathbb{A} = \{A^1, A^2, \ldots, A^n\}$, where A^k is an allocation of O_i that allocates $O_{i,j}$ to agent

$$j' = ((j + k - 1) \bmod n) + 1.$$

Since in $\mathbb{A}$ each item is allocated to each agent once, $\sum_j U_i(A^j) = \sum_j V_{j,i}(\mathcal{M})$. Thus, the worst allocation in set $\mathbb{A}$ has a utility of at most $\sum_j V_{j,i}(\mathcal{M})/n = \overline{V}_i(\mathcal{M})$ for agent i. As a result, $\mathsf{EMMS}_i = U_i(\mathcal{W}_i(O_i)) \le \overline{V}_i(\mathcal{M})$. □

By a similar argument as in the proof of Lemma 2.4 we can show that for an arbitrary partition P,

$$\overline{V}_i(\mathcal{M}) \le U_i(\mathcal{B}_i(P)).$$

Therefore, for any partition P we have

$$\mathsf{EMMS}_i \le U_i(\mathcal{B}_i(P)).$$

In Lemma 2.5, we show that for $n = 2$, a *cut and choose* method guarantees EMMS_i to both the agents.

Lemma 2.5. *For two agents, the following two step algorithm yields a* 1-EMMS *allocation:*(*i*)*: Ask the first agent to partition the items into his optimal partition* O_1*, and* (*ii*)*: Ask the second agent to allocate* O_1 *(one bundle to each agent).*

Proof. We know $U_2(\mathcal{B}_2(O_1)) \ge \mathsf{EMMS}_2$. Furthermore, since $\mathcal{W}_1(O_1)$ determines the value of EMMS_1, we have:

$$U_1(\mathcal{B}_2(O_1)) \ge \mathsf{EMMS}_1.$$

□

Note that there are instances with two agents such that no approximation of average-share can be guaranteed. For example, when there is only one item with value 1 to both the agents, and no externalities. We later establish another difference between these two notions by providing allocations that approximately guarantee extended maximin-share.

3 COMPUTING EMMS

In this section, we study the problem of computing EMMS_i and O_i. A closer look at the model reveals that the challenges to calculate EMMS are twofold. One is to find the worst allocation of a given partition, and the other is to find a partition that maximizes the utility of the worst allocation. In Lemma 3.1 and Observation 1, we explore the hardness of these problems in the **general externalities** model. We then focus on the network externalities model and give a constant factor approximation algorithm for computing EMMS_i.

Lemma 3.1. *Given a partition* $P = \langle P_1, P_2, \ldots, P_n\rangle$ *of the items in* $\mathcal{M}$*, the worst allocation of* P *for agent* i *can be found in polynomial time.*

Proof. (sketch) Consider a complete bipartite graph $G(X, Y)$ where X represents the bundles of P, and Y represents the agents and there is an edge with weight $V_{j,i}(P_k)$ between every pair $x_k \in X$ and $y_j \in Y$. Finding $\mathcal{W}_i(P)$ is equivalent to finding the min-weight perfect matching in G. Classic network flow algorithms solve this problem in polynomial time [11]. □

Observation 1. *Since finding the maximin partition of a set of items is NP-hard [31], finding the optimal partition of m items and n agents with externalities is also NP-hard.*

Woeginger [31] also show that finding the maximin partition of a set of items without externalities admits a PTAS. However, their method does not directly extend to the case with externalities. To the best of our knowledge, finding an approximately optimal partition for an agent with externalities has not been studied before.

In the case of *network externalities*, our model is easier to deal with. Since the utility of each agent is a convex combination of his valuation, finding the worst allocation $\mathcal{W}_i(P)$ is trivial: consider an n-step allocation algorithm whose every step allocates the most valuable remaining bundle to a remaining agent with the least effect on agent i. Hence,

$$U_i(\mathcal{W}_i(P)) = \sum_j x_{i,j} \cdot V_i(P_j). \tag{2}$$

Recall that x_i (the influence vector of agent i) is non-decreasing, and the bundles in P are sorted in non-increasing order of their values for agent i. This property of the network externalities model allows us to approximate the value of EMMS_i with a constant ratio, using a simple greedy approach. On top of that, it is possible to infer relations between EMMS and some classic partitioning schemes.

Apart from the allocation of bundles, partitioning the items is another challenge to overcome. By definition, an optimal partition is a partition that maximizes Equation (2). Finding an optimal partition for a given vector x_i is in fact, a generalized form of partitioning problems that includes both maximin and minimax partitions. What happens if we partition the items by one of the famous partitioning

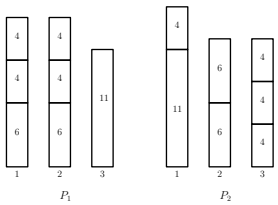


Figure 3: For $x_i = [1, 1, 0]$, minimax (P_1) is optimal, and for $x_i = [1, 0, 0]$, maximin (P_2) is optimal.

schemes such as minimax or maximin? A maximin partition is a partition that maximizes the value of the minimum bundle. It is easy to see that a maximin partition is optimal when $x_i = [1, 0, \ldots, 0]$. Likewise, minimax partition is a partition that minimizes the value of the maximum bundle, and it is the optimal partition when $x_i = [1, 1, \ldots, 1, 0]$. Another example is the leximin partition. A leximin partition first maximizes the minimum bundle, and subject to this constraint, maximizes the second least valued bundle, and so on. Real-world applications of leximin allocations are recently studied by Kurokawa, Procaccia and Shah [21]. For a small enough ϵ, the optimal partition for vector $x_i = [1, \epsilon, \epsilon^2, ..., \epsilon^{n-1}]$ is a leximin partition. See Figure 3 for an illustrative example.

Since none of these partitioning schemes are always optimal, approximating either of them is not desirable. However, the well-known greedy algorithm LPT [3] provides a partition

$$L_i = \langle L_{i,1}, L_{i,2}, \ldots, L_{i,n} \rangle$$

for agent i, such that $U_i(\mathcal{W}_i(L_i))$ is a constant approximation of EMMS_i. LPT is a simple greedy algorithm in the context of job scheduling. This algorithm starts with n empty bundles and iteratively puts the most valuable remaining item into the bundle with the minimum total value. It has been previously established that the partition provided by LPT is a constant approximation for both maximin and minimax partitions [12, 16].

Theorem 3.2. *For the network externalities model, we have*

$$U_i(\mathcal{W}_i(L_i)) \geq \mathsf{EMMS}_i/2. \tag{3}$$

To prove Theorem 3.2, we label some of the items as *huge*. Let $\Delta = V_i(\mathcal{M})/n$, and define Huge items as those items whose values are at least Δ[4]. Denote the set of huge items for agent i by $\mathcal{H}_i$.

Lemma 3.3. *For an instance with no huge item, we have $V_i(L_{i,n}) \geq \Delta/2$.*

Proof. Consider $L_{i,1}$ (the most valuable bundle of L_i for agent i). Trivially, we have $V_i(L_{i,1}) \geq \Delta$, and since there is no huge item, $L_{i,1}$ contains at least two items. On the other hand, according to method of LPT, the items within a bundle arrive in non-increasing order. Therefore, the last item added to $L_{i,1}$ has a value of at most $V_i(L_{i,1})/2$ and the total value of $L_{i,1}$ just before the last item arrives must have been at least $V_i(L_{i,1})/2$. Furthermore, whenever an item is added to a bundle, that bundle has the minimum value among all the bundles. Therefore, $V_i(L_{i,n}) \geq V_i(L_{i,1})/2 \geq \Delta/2$. □

[3]Longest processing time

[4]Notice that Δ and the average share are not necessarily equal.

Since $L_{i,n}$ is the least valued bundle of L_i for agent i,

$$\begin{aligned} U_i(\mathcal{W}_i(L_i)) &\geq \sum_i x_i V_i(L_{i,n}) \\ &\geq \sum_i x_i V_i(L_{i,1})/2 \geq \mathsf{EMMS}_i/2 \end{aligned}$$

Hence, when there is no huge item, regarding Lemma 3.3, Inequality (3) holds. Thus, to prove Theorem 3.2, it only suffices to consider the instances with huge items. Note that, when there are huge items in $\mathcal{M}$, $V_i(L_{i,n}) \geq \Delta/2$ does not necessarily hold. To deal with such situations, we consider some properties for O_i.

Definition 3.4. A partition P is *regular* for agent i, if no item b in some bundle P_j exists, such that $V_i(P_j) > V_i(\{b\}) > V_i(P_n)$ (recall that $V_i(P_n) = \min_j V_i(P_j)$).

Lemma 3.5. *For any partition P, there exists a regular partition P', such that $U_i(\mathcal{W}_i(P)) \leq U_i(\mathcal{W}_i(P'))$.*

Proof. If P is not regular, there exists an item b in some bundle P_j, such that

$$V_i(P_j) > V_i(\{b\}) > V_i(P_n).$$

We modify P as follows: we remove P_j and P_n from P and add two new bundles $A = \{b\}$ and $B = P_j \cup P_n \setminus \{b\}$ to P. Let l and l' be the indices of the newly added bundles in P (note that the bundles are rearranged by their non-increasing values for agent i), such that $j \leq l \leq l' \leq n$ (see Figure 4). We have:

$$V_i(P_j) > \max(V_i(A), V_i(B)) \geq \min(V_i(A), V_i(B)) > V_i(P_n).$$

By this modification, $U_i(\mathcal{W}_i(P))$ increases by a value of at least

$$\begin{aligned} &x_{i,l} \cdot (\max(V_i(A), V_i(B)) - V_i(P_j)) \\ &+ x_{i,l'} \cdot (\min(V_i(A), V_i(B)) - V_i(P_n)), \end{aligned}$$

which is non-negative since

$$V_i(P_j) + V_i(P_n) = \max(V_i(A), V_i(B)) + \min(V_i(A), V_i(B)),$$

and $x_{i,l} \leq x_{i,l'}$. Let

$$\mathcal{L}(P) = \{P_j \mid V_i(P_j) = V_i(P_n)\}.$$

After each modification, either $V_i(P_n)$ increases, or $|\mathcal{L}(P)|$ decreases. Therefore, sequence $(V_i(P_n), V_i(P_{n-1}), \ldots, V_i(P_1))$ increases lexicographically by each move, and hence we eventually end up with a regular partition P' after a finite number of modifications. □

Based on Lemma 3.5, in the rest of this paper we assume that the optimal partitions are regular. Furthermore, in Lemma 3.6 we show that L_i is also regular.

Lemma 3.6. *L_i is regular.*

Proof. For the sake of contradiction, let b be an item in bundle $L_{i,j}$ such that $V_i(L_{i,j}) > V_i(\{b\}) > V_i(L_{i,n})$. Since $V_i(L_{i,j}) > V_i(\{b\})$, $L_{i,j}$ contains at least one other item, say b'. Furthermore, since $V_i(\{b\}) > V_i(L_{i,n})$, after adding b to $L_{i,j}$ no other item can be added to $L_{i,j}$ (recall that in each step of LPT, we add an item to the least valued bundle). This means that b' is added to $L_{i,j}$ before b, which implies $V_i(b') \geq V_i(b)$. But this is a contradiction, because in the step that we add b to $L_{i,j}$, we have $V_i(L_{i,j}) \geq V_i(b') > V_i(L_{i,n})$, which means $L_{i,j}$ is not the minimum bundle in that step. □

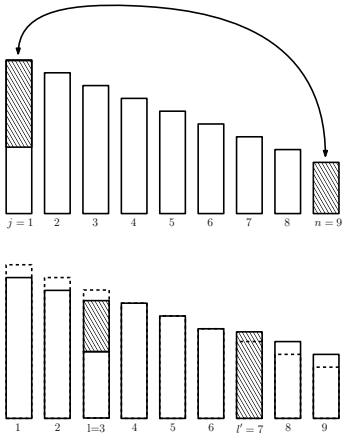


Figure 4: Switching the subsets

In a regular partition P for agent i, any bundle P_j containing a huge item $b \in \mathcal{H}_i$ has no other item. Otherwise, since

$$V_i(P_j) > V_i(\{b\}) \geq \Delta,$$

and $\Delta > V_i(P_n)$, partition P is not regular. This fact about regular partitions (including L_i and O_i) allows us to deal with huge items. We are now ready to prove Theorem 3.2.

Proof of Theorem 3.2. We use induction on the number of agents. For $n = 1$, the statement is trivial. For $n > 1$, we consider two cases. First, if $\mathcal{M}$ contains no huge items, by Lemma 3.3, we have $V_i(L_{i,n}) \geq \Delta/2$. Thus,

$$\begin{aligned} U_i(\mathcal{W}_i(\mathrm{tt}_i)) &= \sum_j V_i(L_{i,j}) \cdot x_{i,j} \\ &\geq \sum_j (\Delta/2) \cdot x_{i,j} \\ &= 1/2 \sum_j \Delta \cdot x_{i,j} \\ &= V_i(\mathcal{M})/2 \sum_j x_{i,j}/n \qquad (\Delta = V_i(\mathcal{M})/n) \\ &= \overline{V}_i(\mathcal{M})/2. \qquad \text{(Equation (1))} \end{aligned}$$

By Lemma 2.4, we have $\overline{V}_i(\mathcal{M}) \geq \mathsf{EMMS}_i$ which implies

$$U_i(\mathcal{W}_i(\mathrm{tt}_i)) \geq \mathsf{EMMS}_i/2.$$

Therefore, it only remains to consider the case that $\mathcal{M}$ contains at least one huge item. Let $B_{\mathcal{H}_i}(O_i)$ and $B_{\mathcal{H}_i}(L_i)$ be the set of the bundles containing huge items in O_i and L_i, respectively. We know that $B_{\mathcal{H}_i}(O_i) = B_{\mathcal{H}_i}(L_i)$, as the bundles in $B_{\mathcal{H}_i}(O_i)$ and $B_{\mathcal{H}_i}(L_i)$ do not contain anything but huge items, and each huge item is the only item within its bundle (recall that both O_i and L_i are regular). In addition, $B_{\mathcal{H}_i}(L_i)$ are the $|\mathcal{H}_i|$ most valuable bundles in L_i. Otherwise, a very similar argument as in the proof of Lemma 3.3 yields $V_i(L_{i,n}) \geq \Delta/2$.

Let $\mathcal{W}'_i(O_i)$ be the worst possible allocation of O_i with the constraint that allocates $|\mathcal{H}_i|$ huge items to the $|\mathcal{H}_i|$ agents with the least influence on agent i. By definition, $U_i(\mathcal{W}'_i(O_i)) \geq U_i(\mathcal{W}_i(O_i))$. Moreover, in both $\mathcal{W}'_i(O_i)$ and $\mathcal{W}_i(L_i)$, huge items are allocated

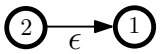


Figure 5: The gap between MMS_i and EMMS_i may be large, even with very small externalities

to the same set of agents, say $\mathcal{N}_{\mathcal{H}_i}$. Now, consider the sub-instance with items $\mathcal{M} \setminus B_{\mathcal{H}_i}(O_i)$ and agents $\mathcal{N} \setminus \mathcal{N}_{\mathcal{H}_i}$. Note that since $V_i(L_{i,n}) < \Delta/2$, the set $\mathcal{M} \setminus B_{\mathcal{H}_i}(O_i)$ (and hence, $\mathcal{N} \setminus \mathcal{N}_{\mathcal{H}_i}$) is non-empty.

By the induction hypothesis, for this sub-instance, Inequality (3) holds. Now, adding huge items and their corresponding agents back, increases the utility of agent i by the same amount for both of the allocations. Thus,

$$U_i(\mathcal{W}_i(L_i)) \geq 1/2 \cdot U_i(\mathcal{W}'_i(O_i)) \geq 1/2 \cdot U_i(\mathcal{W}_i(O_i)).$$

□

4 α-EMMS ALLOCATION PROBLEM

In this section, we focus on the allocations that guarantee every agent i an approximation of EMMS_i. We start this section by comparing EMMS_i to MMS_i.

Let M_i be the maximin partition of $\mathcal{M}$ for agent i. The maximin-share of agent i is equal to the value of the least valued bundle in M_i. Moreover, by definition, we have :

$$\mathsf{EMMS}_i = U_i(\mathcal{W}_i(O_i)) \geq U_i(\mathcal{W}_i(M_i)).$$

This, together with the fact that $U_i(\mathcal{W}_i(M_i)) \geq \mathsf{MMS}_i$ (recall that we have $w_{i,i} = 1$) implies that $\mathsf{EMMS}_i \geq \mathsf{MMS}_i$ always holds. Now, in Lemma 4.1 we show that the gap between EMMS_i and MMS_i could be unbounded even for the instances with 3 agents.

Lemma 4.1. *For any $c \geq 1$, there is an instance with 2 agents, where $\mathsf{EMMS}_1 > c \cdot \mathsf{MMS}_1$.*

Proof. Simply consider the influence graph depicted in Figure 5 and two items b_1 and b_2 such that $V_1(\{b_1\}) = 1$ and $V_1(\{b_2\}) = c/\epsilon$, where ϵ is a small constant less than $1/2$. For this instance, $\mathsf{EMMS}_1 = 1 + c$, and $\mathsf{MMS}_1 = 1$ which means

$$\mathsf{MMS}_1/\mathsf{EMMS}_1 < 1/(1+c).$$

□

The proof of Lemma 4.1 highlights that even for very few externalities, the gap between EMMS and MMS might be large. Thus, the external effects are not negligible even if the impacts of the parties on each other are small.

Our main result is stated in Theorem 4.2. We show that for the network externalities model when all the agents are α-self-reliant, an $\alpha/2$-EMMS allocation always exists. This completely separates extended maximin-share from average-share, since no approximation of the average-share can be guaranteed even for 1-self reliant agents.

Theorem 4.2. *Let $\mathbb{C}$ be an instance such that all the agents are α-self reliant. Then, $\mathbb{C}$ admits an $\alpha/2$-EMMS allocation.*

In the rest of this section, we prove Theorem 4.2 by proposing an $\alpha/2$-EMMS allocation algorithm for the network externalities model with α-self-reliant agents. For brevity, we name our algorithm *Bundle Claiming* (BC) algorithm. It is worth to mention that despite some similarities, this method is fundamentally different from the previous allocation methods for guaranteeing MMS. The main difference is that, here, for each agent, we must keep track of the items allocated to the other parties and based on that, update the utility that each agent must receive to be satisfied. This makes the analysis much more complex.

4.1 Bundle Claiming Algorithm (BC)

In this section, we present the ideas and a general description of Bundle Claiming. First, let us review the definition of EMMS_i. With abuse of notations, we suppose that v_i is a vector representing the values of the bundles in the optimal partition of agent i, i.e., $v_{i,j} = V_i(O_{i,j})$. Recall that the bundles in O_i are sorted by their non-increasing values for agent i. Hence, for all $j < n$, we have $v_{i,j} \geq v_{i,j+1}$. We have $\mathsf{EMMS}_i = \sum_j x_{i,j} v_{i,j}$.

Let $X_i = \sum_j x_{i,j}$. Observation 2 provides an upper bound on the value of $\sum_{j \geq k} x_{i,j} v_{i,j}$.

Observation 2. *For every k, we have $\sum_{j \geq k} x_{i,j} v_{i,j} \leq v_{i,k} \cdot X_i$.*

For example, in an instance with $n = 6$, for $k = 4$, Observation 2 implies:

$$v_{i,4} \cdot X_i \geq v_{i,4} x_{i,4} + v_{i,5} x_{i,5} + v_{i,6} x_{i,6}.$$

Observation 2 is a direct result of the following two facts: first, for all $j > k$, we have $v_{i,k} \geq v_{i,j}$ and second, $\sum_{j>k} x_{i,j} \leq X_i$.

Definition 4.3. For every agent i, we define ℓ_i to be the expectation level of agent i. Agent i with expectation level ℓ_i, has an expectation value of $v_{i,\ell_i}/2$.

In the beginning of the algorithm, the expectation level of all the agents are set to 1. Our algorithm consists of n steps. In each step, we find a bundle B with the minimum number of items that meets the expectation of at least one agent. Bundle B meets the expectation of agent i, if $V_i(B) \geq v_{i,\ell_i}/2$. We allocate B to one of the agents whose expectation is met (we say this agent is satisfied). Next, we update the expectation levels of the remaining agents. The updating process is a fairly complex process which we precisely describe in Section 4.2. Roughly speaking, we update the expectation levels in a way that the following property holds during the algorithm:

> **External-satisfaction property:** Let $\mathcal{S}$ be the set of currently satisfied agents. For every remaining agent i with expectation level ℓ_i, there is a partition of the agents in $\mathcal{S}$ into ℓ_i subsets, namely $N_{i,1}, N_{i,2}, \ldots, N_{i,\ell_i-1}, N_{i,F}$, such that for all $1 \leq j < \ell_i$, the total set of items allocated to the agents in $N_{i,j}$ is worth at least $v_{i,j}/2$ and at most $v_{i,j}$ to agent i, and the total set of items allocated to the agents in $N_{i,F}$ is worth less than $v_{i,\ell_i}/2$ to agent i.

Note that in the updating process, ℓ_i may increase by more than one unit. However, for every remaining agent i, $\ell_i \leq n$ must hold. As we show in Section 4.2, during our algorithm, $\ell_i \leq n$ always holds for every agent i. We use this property to show that if the external-satisfaction property holds for agent i, total amount of externalities incurred by the satisfied agents is at least $\sum_{k<\ell_i} v_{i,k} x_{i,k}/2$. The fact that EMMS_i is calculated with respect to the worst allocation of O_i is the key to prove this inequality.

Consider one step of the algorithm and suppose that a set B of items is allocated to agent i. Since B met the expectation of agent i, $V_i(B) \geq v_{i,\ell_i}/2$. Furthermore, the utility that agent i gained through the externalities of the satisfied agents is at least $\sum_{k<\ell_i} v_{i,k} x_{i,k}/2$. Assuming that agent i is α-self-reliant, his utility is at least

$$\begin{aligned}
&\sum_{k<\ell_i} v_{i,k} x_{i,k}/2 + v_{i,\ell_i}/2 \\
&\geq \sum_{k<\ell_i} v_{i,k} x_{i,k}/2 + \Big(\sum_{k\geq\ell_i} v_{i,k} x_{i,k}/X_i\Big)/2 && \text{(Observation 2)} \\
&\geq (1/2X_i) \sum_k v_{i,k} x_{i,k} && (X_i \geq 1) \\
&\geq \alpha/2 \sum_k v_{i,k} x_{i,k} \\
&= \alpha/2 \mathsf{EMMS}_i. && (4)
\end{aligned}$$

Inequality (4) ensures that the items allocated to agent i satisfy him. Furthermore, we use the external-satisfaction property to prove that the algorithm satisfies all the agents. To show this, it only suffices to prove that in each step of the algorithm there are enough items to meet the expectation of the remaining agents. Consider agent i which has not satisfied yet. The value of the items allocated to the satisfied agents not in $N_{i,F}$ is at most $\sum_{j<\ell_i} v_{i,j}$. Hence, the total value of the remaining items plus the items allocated to the agents in $N_{i,F}$ is at least $\sum_{j\geq\ell_i} v_{i,j}$. Moreover, the value of the items allocated to the agents in $N_{i,F}$ is less than $v_{i,\ell_i}/2$. Thus, the value of the remaining items is at least $v_{i,\ell_i}/2$ which is enough to meet the expectation of agent i. As said before, our algorithm maintains the property that $\ell_i \leq n$ for every remaining agent i.

The details of the bundle claiming algorithm is demonstrated in Algorithm 1. In the next section, we show how to maintain the external-satisfaction property in the algorithm.

Note that, there are only two parts of the BC algorithm whose implementations in polynomial time are not trivial. One is finding a minimum-sized set B meeting the expectation of at least one agent, and the other is the operations we perform in order to maintain the external-satisfaction property. In Observation 3, we show that the first part can be implemented in polynomial time.

Observation 3. *The minimum-sized set that meets the expectation of at least one remaining agent can be found in polynomial time.*

Proof. For every remaining agent i, we find a bundle Γ_i with the minimum size which meets the expectation of agent i as follows: sort the remaining items in their non-increasing values for agent i, and add the items to Γ_i one by one until the bundle meets the expectation of agent i. Finally, it only suffices to select the smallest bundle among these bundles. □

Furthermore, as we show in the next section, the operations we perform to preserve the external-satisfaction property can also be implemented in polynomial time.

ALGORITHM 1: Bundle Claiming algorithm

forall $a_j \in \mathcal{N}$ **do**
  $\ell_j \leftarrow 1$ ▷ Initializing the expectation levels
end
while $\mathcal{N} \neq \emptyset$ **do**
  forall $a_j \in \mathcal{N}$ **do**
    $\Gamma_j \leftarrow$ Minimum sized subset of $\mathcal{M}$, s.t.
    $V_j(\Gamma_j) \geq 1/2 \cdot v_{j,\ell_j}$;
  end
  $i \leftarrow \arg\min_i |\Gamma_i|$;
  Allocate Γ_i to agent i ;
  $\mathcal{N} \leftarrow \mathcal{N} \setminus i$, $\mathcal{M} \leftarrow \mathcal{M} \setminus \Gamma_i$;
  forall $j \in \mathcal{N}$ **do**
    $N_{j,F} \leftarrow N_{j,F} \cup \{i\}$. ;
    while $V_j(N_{j,F}) \geq 1/2 \cdot v_{j,l_j}$ **do**
      $Update(M_j)$ ▷ (see Section 4.2)
    end
  end
end

Finally, using L_i [5] instead of O_i in BC results in an $\alpha/4$-EMMS allocation algorithm.

Corollary 4.4. *Let $\mathbb{C}$ be an instance where for every agent i, $w_{i,i} \geq \alpha$. Then, an $\alpha/4$-EMMS allocation for $\mathbb{C}$ can be found in polynomial time.*

4.2 The External-satisfaction Property

Throughout this section, we suppose that $\mathcal{S}$ is the set of satisfied agents. Furthermore, for each agent i in $\mathcal{S}$, we denote the bundle allocated to him by B_i. We start by giving a detailed explanation of the updating process. As mentioned in the previous section, the external-satisfaction condition must hold during the entire algorithm. To maintain this property in the updating process, for every agent i, we define a mapping M_i that represents the partitioning of $\mathcal{S}$ for agent i (recall the definition of external-satisfaction).

Definition 4.5. For every agent i, we define

$$M_i : \mathcal{S} \rightarrow \{O_{i,1}, O_{i,2}, \ldots, O_{i,n}\} \cup \{F_i\}$$

as a mapping that corresponds each satisfied agent to a bundle in the optimal partition of O_i or to F_i. Furthermore, we define $N_{i,j}$ as the set of agents that are mapped to $O_{i,j}$ in M_i and $N_{i,F}$ as the set of agents mapped to F_i. During the algorithm, we say mapping M_i is valid, if the following conditions hold:

(1) $\forall j < \ell_i \quad \sum_{k \in N_{i,j}} V_i(B_k) \geq v_{i,j}/2$

(2) $\forall j < \ell_i \quad \sum_{k \in N_{i,j}} V_i(B_k) \leq v_{i,j}$

(3) $\sum_{k \in N_{i,F}} V_i(B_k) < v_{i,\ell_i}/2$

During the entire algorithm, mapping M_i must remain valid for every unsatisfied agent i. In the beginning, $\mathcal{S} = \emptyset$ and for every agent i, $\ell_i = 1$ and hence, M_i is valid. In each step of the algorithm, we satisfy an agent i by a bundle B_i. Next, for every unsatisfied agent j, we map agent i to F_j in M_j, i.e., we set $M_j(i) = F_j$. In fact, $N_{j,F}$ corresponds to the satisfied agents that are not mapped to any bundle of O_j in M_j. We use these agents to update ℓ_j. Throughout the algorithm, whenever the total value of the items allocated to the agents in $N_{j,F}$ reaches $v_{j,\ell_j}/2$, M_j becomes invalid and hence, we need to update ℓ_j and M_j to reinstate the validity of M_j. To do so, we pick a subset δ of the agents in $N_{j,F}$ with the minimum size to map them to O_{j,ℓ_j}. Regarding the validity conditions of M_j, total value of the items allocated to the agents in δ must be at least $v_{j,\ell_j}/2$ and at most v_{j,ℓ_j} (we call such subset a *compatible set*). If a compatible set δ exists, we map the agents in δ to O_{j,ℓ_j} in M_j and increase ℓ_j by one. However, there may be some cases that no subset of $N_{j,F}$ is compatible. For such cases, we use the argument in Lemma 4.6.

Lemma 4.6. *Suppose that total value of the items allocated to the agents in $N_{j,F}$ is at least $v_{j,\ell_j}/2$, but $N_{j,F}$ admits no compatible subset. Then, it is possible to modify M_j such that conditions (i) and (ii) remain valid for M_j and $N_{j,F}$ contains at least one compatible subset.*

Using Lemma 4.6 we can modify M_j and then update the mapping. Note that, after increasing ℓ_j, condition (iii) may still be violated. In that case, as long as condition (iii) is violated, we continue updating. Each time we update M_j, value of ℓ_j is increased by one. Since at least one agent is mapped to $O_{j,\ell}$ for each $\ell < \ell_j$, ℓ_j never exceeds n. In the appendix (Algorithm 2), you can find a pseudo-code for this process .

In the last part of this section, we prove Lemma 4.7 which shows that the value of the externalities imposed to agent i by the satisfied agents is lower-bounded by $\sum_{j<\ell_i} x_{i,j} v_{i,j}/2$. As said before, the fact that EMMS_i is defined with regard to the worst allocation of O_i plays a key role in proving Lemma 4.7.

Lemma 4.7. *Consider one step of the algorithm, and let agent i be an arbitrary remaining agent with $\ell_i > 1$. Then, we have*

$$\sum_{j \in \mathcal{S}} w_{j,i} \cdot V_i(B_j) \geq \sum_{j<\ell_i} x_{i,j} \cdot v_{i,j}/2. \tag{5}$$

Proof. We want to show that in every step of the algorithm, for each remaining agent i, Inequality (5) holds. To prove this, we apply a sequence of exchanges between the bundles allocated to the agents in $\bigcup_{j<\ell_i} N_{i,j}$ and show that in every exchange, value of the expression on the left-hand side of Inequality (5) does not increase [6]. Next, we show that after these exchanges, Inequality (5) holds, which means that the Inequality was held for the original allocation.

Let agent j be the agent in $N_{i,1}$ with the least influence on agent i (i.e., minimizes $w_{j,i}$). First, we allocate the bundles that belong to the other agents in $N_{i,1}$ to agent j and remove all the agents but agent j from $N_{i,1}$. Since agent j has the minimum weight (influence) among the agent in $N_{i,1}$, this operation does not increase the left-hand side of Inequality (5).

In addition, let agent j' be the agent with $w_{j',i} = x_{i,1}$. Since agent j' has the minimum weight among all the agents, we have $w_{j',i} \leq w_{j,i}$. Now, let B_j and $B_{j'}$ be the current bundles of agents j

[5] Partitioning provided by LPT algorithm

[6] Note that these exchanges are only to prove this lemma, and not in the algorithm.

and j' (if j' is not satisfied yet, $B_{j'} = \emptyset$). If $V_i(B_{j'}) < V_i(B_j)$, we swap the bundles of j and j'. This operation also does not increase the left-hand side of Inequality (5) since we have $w_{j',i} \leq w_{j,i}$. Finally, we exchange the set that agents j and j' belong to: we remove agent j from $N_{i,1}$, and add agent j' to $N_{i,1}$. In addition, if agent j' previously belonged to $N_{i,r}$ for some r, we add agent j to $N_{i,r}$. This exchange has no effect on the value of $\sum_{j \in S} w_{j,i} \cdot V_i(B_j)$. Furthermore, one can easily observe that after the exchange, condition (ii) holds.

We repeat the same procedure for $N_{i,2}, N_{i,3}, \ldots, N_{i,\ell_i - 1}$. After this sequence of exchanges, each $N_{i,j}$ contains one agent j', where $w_{j',i} = x_{i,j}$. Furthermore, after the exchanges, the second condition for the validity of M_i holds and hence, the value of the items of agent j' for agent i is at least $v_{i,j}/2$. Therefore, total amount of externalities of the satisfied agents is at least $\sum_{j < \ell_i} x_{i,j} v_{i,j}/2$. □

Proof of Lemma 4.6. Let δ be a subset of $N_{j,F}$ with the minimum size that satisfies condition (i). Such a set trivially exists. Since no subset of $N_{j,F}$ is compatible, we have $\sum_{k \in \delta} V_j(B_k) > v_{j,\ell_j}$. By minimality of δ, no proper subset of δ satisfies condition (i). It is easy to observe that this can only happen when δ contains only one agent, say k, with $V_i(B_k) > v_{j,\ell_j}$. We also show in Lemma 4.8 that $|B_k| = 1$; but for now suppose that $|B_k| = 1$ and b is the only item in B_k.

Since O_j is regular[7], there is an index $\ell < \ell_j$, such that bundle $O_{j,\ell} = \{b\}$. We modify M_j as follows: we map agent k to $O_{j,\ell}$ and map the former agents of $N_{j,\ell}$ to F_j. Clearly, conditions (i) and (ii) preserve for M_j after this process. Again, if no subset of $N_{j,F}$ is compatible, we repeat this modification. Each time we modify M_j, the number of indices ℓ for which $O_{j,\ell}$ is mapped to an agent j with $O_{j,\ell} = B_j$ increases by one. Therefore, the process terminates after a finite number of modifications. □

Algorithm 2 illustrates an overview of the update procedure, which completes the BC algorithm.

ALGORITHM 2: Update M_j

```
Resolve = 0
while Resolve == 0 do
    δ ← Minimum sized subset of N_{j,F}, s.t.
      Σ_{k∈δ} V_j(B_k) ≥ 1/2 · v_{j,ℓ_j}
    if Σ_{k∈δ} V_j(B_k) ≤ v_{j,ℓ_j} then
        N_{j,ℓ_j} = δ
        N_{j,F} = N_{j,f} \ δ
        ℓ_j ← ℓ_j + 1
        Resolve+ = 1                          ▷ Resolved
    else
        Let ℓ be an index s.t. O_{j,ℓ} = B_k, where δ = {k}.
        Swap δ (which is a subset of N_{j,F}) with N_{j,ℓ}.   ▷ One
          step closer to resolve
    end
end
```

Lemma 4.8. *Suppose that the total value of the items allocated to the agents in $N_{j,F}$ is at least $v_{j,\ell_j}/2$, but $N_{j,F}$ admits no compatible subset, and let δ be the minimal subset of $N_{j,F}$ that satisfies condition (i). Then, δ contains only one agent, say agent k and $|B_k| = 1$.*

Proof. As mentioned in Lemma 4.6, it is easy to observe that $|\delta| = 1$. Here, we argue that if agent k is the only agent in δ, then $|B_k| = 1$. As a contradiction, let z_1 be the first step of the algorithm that $\delta = \{k\}$, but $|B_k| > 1$. In addition, let z_2 be the step that B_k is allocated to agent k and let ℓ'_j be the expectation level of agent j in step z_2. Trivially, we have $z_2 \leq z_1$. □

Claim 1. *Either $v_{j,n} \geq v_{j,\ell'_j}/2$ or we have $|O_{j,\ell}| = 1$ for all $\ell \leq \ell'_j$.*

Proof of Claim 1. If for some $\ell \leq \ell'_j$, $O_{j,\ell}$ contains more than one item, $O_{j,\ell}$ has a proper subset s such that $V_j(s) \leq v_{j,\ell}/2$. By the same reasoning as Claim 3.5, moving s to bundle $O_{j,n}$ yields a new partition which is at least as good as O_j (See Figure 4). Hence, we can assume w.l.o.g. that $v_{j,n} \geq v_{j,\ell'_j}/2$ holds. □

Regarding Claim 1, we consider two cases.

First, assume that $|O_{j,\ell}| = 1$ for all $\ell \leq \ell'_j$. For this case, at least one of the items in $\bigcup_{\ell \leq \ell'_j} O_{j,\ell}$ is not allocated to any agent before step z_2, and this item singly meets the expectation of agent j. This contradicts the fact that at step z_2, B_k was the minimal set (Note that we supposed $|B_k| > 1$).

Second, assume that $v_{j,\ell_j} \geq v_{j,\ell'_j}/2$. In step z_2, the expectation value of agent j equals $v_{j,\ell'}/2$. Furthermore, $V_j(B_k) > v_{j,\ell_j}$ which means $V_j(B_k) > v_{j,\ell'_j}/2$. On the other hand, $V_j(B_k) < v_{j,\ell'_j}$, otherwise a proper subset of B_k would meet the expectation of agent j in step z_2. Therefore, in step z_2, $\delta = \{k\}$ is the only compatible set for updating M_j and hence, agent k is mapped to O_{j,ℓ'_j}. This also implies that $z_2 \neq z_1$, since we supposed that no compatible subset exists in step z_1.

Furthermore, notice that since $|B_k| > 1$, no item could singly meet the expectation of any agent, including agent j in step z_2. This means that every remaining item in step z_2 has the value less than $v_{j,\ell_j}/2$. On the other hand, in all the modifications before step z_1, the bundle allocated to the agent in δ consists of only one item (z_1 is te first step that the size of the bundle allocated to the agent in δ is more than 1). This means that after step z_2, no modification affects the agents that are mapped to bundles $O_{j,\ell}$ for $\ell \leq \ell'_j$. But this contradicts the fact that agent k is mapped to F_j in step z_1, because agent k is mapped to O_{j,ℓ'_j} and no modification changes $M_j(k)$.

5 CONCLUSION

An exciting open direction is to find approximation allocation algorithms for the general externalities model, with no restriction on the value of $V_{j,i}(\{b_k\})$. Many issues complicate the study of the general model. For example, the approximation algorithm presented for computing the value of EMMS_i is no longer applicable to this model. A good starting point is to study the general model for the cases with a few number of agents, e.g., 3 or 4 agents.

For the network externalities model, one can think of improving the approximation ratio of the allocation. In particular, it would be

[7] Recall the regularity from Definition 3.4

interesting to propose an allocation algorithm with approximation factor independent of the self-reliance of the agents.

Another interesting open direction that might be of independent interest is to find a maximum value β, such that there exists a partition P of the items in which $\mathcal{W}_i(P)$ is a β-approximation of $\mathcal{W}_i(O_i)$ for every agents i. Note that if such a partition exists, any allocation of it to the agents is a β-EMMS allocation.

Finally, it is very impressive to present a *PTAS* for finding the optimal partition for an agent in the network externalities model.

REFERENCES

[1] Georgios Amanatidis, Georgios Birmpas, George Christodoulou, and Evangelos Markakis. 2017. Truthful allocation mechanisms without payments: Characterization and implications on fairness. In *Proceedings of the 2017 ACM Conference on Economics and Computation*. ACM, 545–562.

[2] Georgios Amanatidis, Georgios Birmpas, and Evangelos Markakis. 2016. On truthful mechanisms for maximin share allocations. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. AAAI Press, 31–37.

[3] Georgios Amanatidis, Evangelos Markakis, Afshin Nikzad, and Amin Saberi. 2015. Approximation algorithms for computing maximin share allocations. In *International Colloquium on Automata, Languages, and Programming*. Springer, 39–51.

[4] Nima Anari, Shayan Ehsani, Mohammad Ghodsi, Nima Haghpanah, Nicole Immorlica, Hamid Mahini, and Vahab S Mirrokni. 2010. Equilibrium pricing with positive externalities. In *International Workshop on Internet and Network Economics*. Springer, 424–431.

[5] Haris Aziz, Gerhard Rauchecker, Guido Schryen, and Toby Walsh. 2017. Algorithms for Max-Min Share Fair Allocation of Indivisible Chores.. In *AAAI Conference on Artificial Intelligence*, Vol. 17. 335–341.

[6] Siddharth Barman, Arpita Biswas, Sanath Kumar Krishnamurthy, and Yadati Narahari. 2018. Groupwise maximin fair allocation of indivisible goods. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[7] Siddharth Barman and Sanath Kumar Krishna Murthy. 2017. Approximation algorithms for maximin fair division. In *Proceedings of the 2017 ACM Conference on Economics and Computation*. ACM, 647–664.

[8] Simina Brânzei, Tomasz Michalak, Talal Rahwan, Kate Larson, and Nicholas R Jennings. 2013. Matchings with externalities and attitudes. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 295–302.

[9] Simina Brânzei, Ariel D Procaccia, and Jie Zhang. 2013. Externalities in Cake Cutting.. In *International Joint Conference on Artificial Intelligence*. 55–61.

[10] Eric Budish. 2011. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy* 119, 6 (2011), 1061–1103.

[11] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2009. *Introduction to algorithms*. MIT press.

[12] Bryan L Deuermeyer, Donald K Friesen, and Michael A Langston. 1982. Scheduling to maximize the minimum processor finish time in a multiprocessor system. *SIAM Journal on Algebraic Discrete Methods* 3, 2 (1982), 190–196.

[13] Alireza Farhadi, MohammadTaghi Hajiaghayi, Mohammad Ghodsi, Sebastien Lahaie, David Pennock, Masoud Seddighin, Saeed Seddighin, and Hadi Yami. 2017. Fair Allocation of Indivisible Goods to Asymmetric Agents. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 1535–1537.

[14] Mohammad Ghodsi, MohammadTaghi HajiAghayi, Masoud Seddighin, Saeed Seddighin, and Hadi Yami. 2018. Fair Allocation of Indivisible Goods: Improvements and Generalizations. In *Proceedings of the 2018 ACM Conference on Economics and Computation*. ACM, 539–556.

[15] Laurent Gourvès and Jérôme Monnot. 2017. Approximate Maximin Share Allocations in Matroids. In *International Conference on Algorithms and Complexity*. Springer, 310–321.

[16] Ronald L. Graham. 1969. Bounds on multiprocessing timing anomalies. *SIAM journal on Applied Mathematics* 17, 2 (1969), 416–429.

[17] Nima Haghpanah, Nicole Immorlica, Vahab Mirrokni, and Kamesh Munagala. 2011. Optimal auctions with positive network externalities. In *Proceedings of the 12th ACM conference on Electronic commerce*. ACM, 11–20.

[18] Tobias Heinen, Nhan-Tam Nguyen, Trung Thanh Nguyen, and Jörg Rothe. 2018. Approximation and complexity of the optimization and existence problems for maximin share, proportional share, and minimax share allocation of indivisible goods. *Autonomous Agents and Multi-Agent Systems* (2018), 1–38.

[19] Miho Hong and Jaeok Park. 2018. Core and Top Trading Cycles in a Market with Indivisible Goods and Externalities. (2018).

[20] David Kempe and Mohammad Mahdian. 2008. A cascade model for externalities in sponsored search. In *International Workshop on Internet and Network Economics*. Springer, 585–596.

[21] David Kurokawa, Ariel D Procaccia, and Nisarg Shah. 2015. Leximin allocations in the real world. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*. ACM, 345–362.

[22] David Kurokawa, Ariel D Procaccia, and Junxing Wang. 2016. When can the maximin share guarantee be guaranteed?. In *AAAI Conference on Artificial Intelligence*, Vol. 16. 523–529.

[23] Renato Paes Leme, Vasilis Syrgkanis, and Éva Tardos. 2012. Sequential auctions and externalities. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*. SIAM, 869–886.

[24] Minming Li, Jialin Zhang, and Qiang Zhang. 2015. Truthful cake cutting mechanisms with externalities: do not make them care for others too much!. In *International Joint Conference on Artificial Intelligence*. 589–595.

[25] Vahab S Mirrokni, Sebastien Roch, and Mukund Sundararajan. 2012. On Fixed-Price Marketing for Goods with Positive Network Externalities.. In *WINE*. Springer, 532–538.

[26] Eva Pataki, Andras Sagi, and Kabok Jozef. 2015. Externalities and the optimal allocation of economic resources. In *International Symposium on Intelligent Systems and Informatics*. IEEE, 185–188.

[27] Ariel D Procaccia and Junxing Wang. 2014. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the fifteenth ACM conference on Economics and computation*. ACM, 675–692.

[28] Hugo Steinhaus. 1948. The problem of fair division. *Econometrica* 16, 1 (1948).

[29] Warut Suksompong. 2018. Approximate maximin shares for groups of agents. *Mathematical Social Sciences* 92 (2018), 40–47.

[30] Rodrigo A Velez. 2016. Fairness and externalities. *Theoretical Economics* 11, 1 (2016), 381–410.

[31] Gerhard J Woeginger. 1997. A polynomial-time approximation scheme for maximizing the minimum machine completion time. *Operations Research Letters* 20, 4 (1997), 149–154.

“It’s almost like they’re trying to hide it": How User-Provided Image Descriptions Have Failed to Make Twitter Accessible

Cole Gleason
Carnegie Mellon University
Pittsburgh, Pennsylvania
cgleason@cs.cmu.edu

Patrick Carrington
Carnegie Mellon University
Pittsburgh, Pennsylvania
pcarrington@cmu.edu

Cameron Cassidy
Carnegie Mellon University
Pittsburgh, Pennsylvania
ccassidy@tamu.edu

Meredith Ringel Morris
Microsoft Research
Redmond, Washington
merrie@microsoft.com

Kris M. Kitani
Carnegie Mellon University
Pittsburgh, Pennsylvania
kkitani@cs.cmu.edu

Jeffrey P. Bigham
Carnegie Mellon University
Pittsburgh, Pennsylvania
jbigham@cs.cmu.edu

ABSTRACT

To make images on Twitter and other social media platforms accessible to screen reader users, image descriptions (alternative text) need to be added that describe the information contained within the image. The lack of alternative text has been an enduring accessibility problem since the “alt” attribute was added in HTML 2.0 over 20 years ago, and the rise of user-generated content has only increased the number of images shared. As of 2016, Twitter provides users the ability to turn on a feature that allows descriptions to be added to images in their tweets, presumably in an effort to combat this accessibility problem. What has remained unknown is whether simply enabling users to provide alternative text has an impact on experienced accessibility. In this paper, we present a study of 1.09 million tweets with images, finding that only 0.1% of those tweets included descriptions. In a separate analysis of the timelines of 94 blind Twitter users, we found that these image tweets included descriptions more often. Even users with the feature turned on only write descriptions for about half of the images they tweet. To better understand why users provide alternative text descriptions (or not), we interviewed 20 Twitter users who have written image descriptions. Users did not remember to add alternative text, did not have time to add it, or did not know what to include when writing the descriptions. Our findings indicate that simply making it possible to provide image descriptions is not enough, and reveal future directions for automated tools that may support users in writing high-quality descriptions.

CCS CONCEPTS

• **Human-centered computing** → **Accessibility**.

KEYWORDS

Accessibility, Alternative Text, Image Description, Blind, Twitter

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW ’19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313605

ACM Reference Format:
Cole Gleason, Patrick Carrington, Cameron Cassidy, Meredith Ringel Morris, Kris M. Kitani, and Jeffrey P. Bigham. 2019. “It’s almost like they’re trying to hide it": How User-Provided Image Descriptions Have Failed to Make Twitter Accessible. In *Proceedings of the 2019 World Wide Web Conference (WWW ’19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313605

1 INTRODUCTION

Twitter and other social media platforms have become increasingly visual [24] over the past decade as media such as photos, videos, and GIFs have become more prevalent as content. As visual media makes up a larger portion of total content on a social media platform, such as Twitter, such platforms risk becoming less accessible to people with vision impairments who use screen reading software to access the site [24].

An estimated 39 million people around the world are blind, and many access online sites through screen reader software. They are typical users of social networks [6, 34], but they are not always afforded access to content on social networks like Twitter [24], which are increasingly part of public discourse. Twitter is a platform for members of the media to disseminate and discuss information, as well as users to interact with celebrities in a different format than more traditional media [35]. Many research efforts have examined participation on Twitter in the context of politics around the globe [3], especially during elections [28]. It is important that all visual content on Twitter, including those relating to these topics, be accessible to people with vision impairments so they may have equal participation in public life on the Internet.

Social media platforms have taken different approaches to make visual content more accessible to blind users, although all approaches provide textual descriptions (alternative text) to user-posted images. For instance, Facebook and Instagram both automatically tag each image uploaded to the site using image detection and recognition algorithms [18, 36]. Users can edit and override this text after the image has been posted. Twitter, on the other hand, allows users to add their own descriptions when the image is posted, provided the user has previously enabled that feature.

The automatic alternative text provided by Facebook is always available, but is not yet trustworthy to blind users compared to high-quality alternative text written by humans [36]. However, few users have enabled Twitter’s image description feature for their account, and those that have do not always remember to write alternative

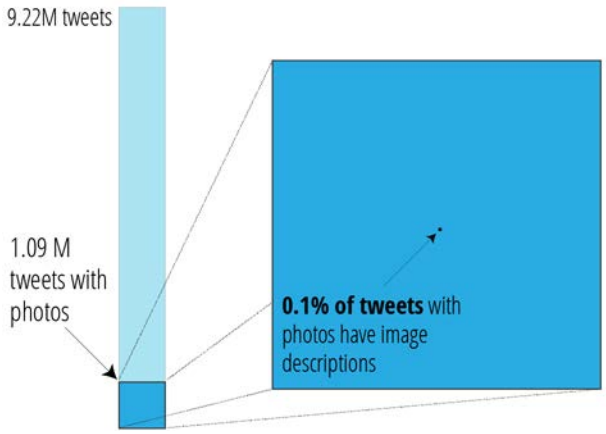


Figure 1: We sampled 9.22 million tweets, collecting 1.09 million with images. Only 0.1% of these tweets contained alternative text for people with vision impairments created using the opt-in image description feature on Twitter.

text. We were interested in further understanding why Twitter users chose to enable and use this feature to better understand what motivates them to provide image descriptions. This understanding will help us improve similar features on social networking sites and increase the number of users providing high-quality alternative text for their social media content.

To understand the current state of alternative text provided on Twitter, we collected a sample of 1.09 million photo tweets and found that only 0.1% contained alternative text. By looking at a sample of posts with alternative text written in English, we found that 83.4% of human-written descriptions were of high quality. We then interviewed 20 Twitter users who had written image descriptions to understand their motivations for writing them.

Our findings suggest that very few users enabled the ability to post alternative text, indicating that Twitter could increase accessibility by turning the feature on by default. Those who do use the feature often do so infrequently, but generally provide alternative text of high-quality (excluding automatic posts from bots). Users who do use the feature could still benefit from training or tools that would help them write better image descriptions. We suggest that researchers or Twitter community members who wish to improve accessibility for the site develop these tools and measure their impact on the accessibility of social media content.

2 RELATED WORK

Our investigation of the accessibility of Twitter is related to prior research on both manual and automatic methods to generate image descriptions, and the usage of alternative text on the web.

2.1 What are Image Descriptions?

Image descriptions, often referred to as alternative text or "alt text", are captions for images online or in other software. Alternative text is most commonly encountered on webpages, as it was added to the HTML 2.0 specification in 1995 [4]. However, at the time it was intended for users who used non-graphical browsers or preferred not to render images when accessing the web. Visually impaired users are not mentioned as an intended audience until the HTML 4.0 specification in late 1997 [9]. Accessibility of images for screen-reader users is one of the most commonly cited reasons to add alternative text to images today, but it is also recommended in case the image does not load for sighted users. Image descriptions are also expected to be added to software in other domains, including mobile phone applications on the iOS [2] and Android [14] platforms. Commercial software can also add alternative text to images in documents, such as Microsoft Word and Adobe Acrobat [1, 22].

Alternative text is not solely meant for human consumption, as various search engines consume it and use it to rank pages [19]. In fact, image descriptions have been used for a number of different applications including "semantic visual search, visual intelligence in chatting robots, photo and video sharing in social media, and aid for visually impaired people to perceive surrounding visual content" [17]. Image labels, captions, and descriptions provide a solid foundation for many of these kinds of applications.

In this paper we focus on alternative text image descriptions for people with vision impairments. Thus, our evaluation revolves around the descriptive quality of captions for a human, as in [25].

2.2 Methods to Generate Alternative Text

Currently, alternative text is primarily created by the developers of the website or authors of the website content. This text is manually written, and authors are recommended to follow the Web Content Accessibility Guidelines [10]. However, as many images on the web are not labelled correctly or at all [5, 16], researchers have sought to automatically generate image descriptions. Both for online accessibility and in situ photos, subsets of individual objects in photos can be labelled using object recognition algorithms [36]. More complex images can be described by scene description models, although this is usually less robust than human labelling. A commercial example of this is Microsoft Seeing AI [21]. However, researchers should remain skeptical of automatically-generated descriptions until they are of consistently high-quality, as prior work has also illuminated that there is a threshold for when image descriptions and captions can do more harm than good [25].

Because many images are complex and difficult to describe, especially for photos with a specific intent, the best alternative text is typically provided by human labelling. As relying on the original content authors to provide accessible images has not proven fruitful, researchers have proposed multiple ways to share alternative text for images between users [12] and collaboratively make websites more accessible without permission from the owner [26]. Guinness *et al.* proposed the Caption Crawler to automatically retrieve alternative text attached to the same image elsewhere on the web through reverse image searches, which achieves a similar goal without active crowdsourcing [15].

2.3 Alternative Text on the Web

The majority of image content on the web does not contain image descriptions. Bigham *et al.* found that less than 40% of significant images on the top 500 high-traffic websites contained alternative text [5]. This motivated the authors to create a tool to generate

alternative text automatically from surrounding web context and optical character recognition. They found that on top-ranked sites by traffic, they automatically generated captions for around 50% of not-described images. In a historical analysis of websites from 1997-2002, Hackett *et al.* found that websites were getting increasing complex and less accessible [16]. Notably, in both this study and Bigham *et al.*, government websites tended to be more accessible than other groups. A more recent (2017) survey of top websites by Guinness *et al.* found 20-35% of images lacking alt text in various categories [15]. Over the last ten years, the number of accessible images on the web has not noticeably changed.

With the rise of social media platforms, a significant amount of image content on the web are now generated by end-users, not website authors. This has lead to a large amount of content being inaccessible, as users did not have the option to add descriptions to their posts. Morris *et al.* found that over 25% of English tweets in June 2015 contained an image, and Twitter did not allow alternative text to be added at the time [24]. The post text itself was not a substitute for alt text, as the authors found that only 11.2% of tweets would serve as good descriptions for their accompanying images. In 2016, Twitter added an opt-in feature for users to write image descriptions for their images, which we examine in detail.

3 THE STATE OF TWITTER ACCESSIBILITY

We sought to quantify the usage of the image description feature across Twitter to understand how the introduction of image descriptions have made Twitter more accessible. We describe the image description feature, in general, followed by overall creation of alternative text by individual, popular, and public accounts.

3.1 Adding Image Descriptions

In 2016, Twitter added a feature to allow users to add image descriptions in their tweets that contain images [27]. By default this feature is disabled. The feature must be enabled by going to *Settings and privacy -> Accessibility* and clicking *Compose image descriptions* [30]. A note below the checkbox says *"Adds the ability to describe images for the visually impaired."* and a link provides more information in Twitter's documentation. Once the feature is enabled, a visual cue appears when a user uploads an image on the website or mobile application (Figure 2). The tweet author may add a description of up to 420 characters to each image (a tweet may contain up to four). Descriptions may not be added to videos or animated GIFs, the other common media formats on the site, and descriptions may not be added or edited after the tweet is posted. Image descriptions can be added to tweets posted through the API using the "ext_alt_text" tag [31], so third party tools may enable image descriptions for their users as well.

3.2 Snapshot of Image Description Usage

We first aimed to measure the amount of image descriptions on Twitter. Using Twitter's public API [32] we collected a sample of public tweets across 5 days in June 2018 for an average duration of 12 hours per day. This resulted in a collection of over 9 million tweets from all languages, including both original tweets and retweets (which may be duplicated). Approximately 1.09 million, or 11.84%, of those were tweets with at least one image. Only 1,144, or 0.1%, of those tweets (with images) contained image descriptions for at least one image.

3.2.1 Original Photo Tweets. Upon further examination of the tweets with images, we noted that 271,330, or about 25%, were original tweets. The other 75% were retweets. Only 177, or 0.07%, of original photo tweets contained image descriptions. When we examined the image descriptions in our sample we noticed that some images had URLs as alternative text (often the source of the image), which we filtered out as they are not descriptive. After removing URLs, there were 166 tweets remaining; 0.06% of the original 271,330 photo tweets.

3.2.2 Retweets. If a tweet contains an image description, retweets of it will still contain the description, but new descriptions cannot be added by other users. Retweeted tweets accounted for 75% of the image tweets in our sample, although many of these were duplicated. We had a total of 820,469 retweets, but only 426,084 were unique (51.2%). For unique retweets, 0.11% contained descriptions, which after removing URLs left only 0.05% or 207 tweets.

3.2.3 Photos vs Tweets. Twitter allows a user to attach up to four images to a single tweet, each with its own description. When examining each photo from original tweets we observed that 336,584 photos were shared, and similar to the other categories only 0.05% contained image descriptions (only 0.03% after filtering out URLs).

The details of our sample and breakdown of our analysis is shown in Table 1. Overall, this suggests that less that 0.05% of the image content on Twitter is accessible to screen reader users.

3.3 Accessibility of Popular Accounts

We wondered if, despite the fact that tweets from randomly-sampled users were mostly inaccessible, perhaps more popular accounts would enable and use the feature regularly. These accounts, run by celebrities or organizations, were more likely to have professionals writing their content. Additionally, if popular accounts were accessible, they would have a greater impact on the overall accessibility of Twitter, as they would appear in more user's timelines.

We collected the image tweets from the top 50 most popular Twitter accounts [33] by number of followers. The tweets collected included every image tweet available between the time the feature was launched and October 2018. We found that only 3 of the 50



Figure 2: The user interface for adding image descriptions on Twitter. The left image shows the "Compose Tweet" window with the "Add Description" pop up circled in blue. The right side is the subsequent window showing the added image with a field at the bottom to add a description.

Table 1: Number of tweets and retweets containing photos and alt-text.

	Day 1	Day 2	Day 3	Day 4	Day 5	Total
Tweets and Photo Tweets						
Total Tweets	1,860,947	1,757,012	1,366,244	2,243,235	1,997,713	9,225,151
Photo Tweets	225,679	189,445	145,778	269,098	261,789	1,091,799
Photo Tweets with alt text	195	199	205	297	248	1,144
Original Photo Tweets						
Original Tweets with photos	57,530	46,793	37,949	65,546	63,512	271,330
Original Photo Tweets with Alt	40	34	24	53	26	177
Photo Retweets						
Photo Retweets	168,149	142,652	107,839	203,552	198,277	820,469
Photo Retweets with alt	155	165	181	244	222	967
Original Photos and Retweeted Photos						
Total Original Photos	71,065	56,736	46,626	81,262	80,895	336,584
Total Retweeted Photos	287,909	241,828	183,244	355,346	349,894	1,418,221

accounts had ever used the feature: the official Twitter account, Bill Gates, and BBC Breaking News. Assuming they enabled the feature just before their first tweet with an image description, these three accounts collectively added descriptions to 14 (8.8%) of 159 image tweets. This assumption does not make much sense for the Twitter account, of course, as it introduced the feature [27]. Only 5 of Twitter's 44 image tweets since the introduction of the feature have included image descriptions. The other 47 popular accounts that never added descriptions for accessibility to their images included prominent news organizations (New York Times, CNN, ESPN), politicians (Barack Obama, Donald Trump, Narendra Modi), and celebrities (Katy Perry, Justin Bieber, Taylor Swift).

Based on prior work that found government websites more accessible [5, 16], we wanted to see if this trend carried across to government accounts on Twitter. Using a list (created by C-SPAN) of 577 accounts associated with members of the current U.S. Congress [8], we performed the same analysis as above. In all tweets since the introduction of the feature, 42 accounts had used the feature at least once, with an average of 3.8% of their image tweets containing a description.

4 ACCESSIBILITY FOR BLIND USERS

Understanding the state of accessible images on Twitter as a whole is valuable to inform how easy it is for a screen reader user to interact with any piece of content on the platform. However, this is not how most users experience Twitter content. They follow a set of users and only see the content authored, retweeted, or liked by the users they follow.

We analyzed the timelines of 94 self-reported blind users to determine if the experience of using Twitter is more accessible for people with vision impairments. To find blind users (who likely use screen readers to access Twitter), we collected a random sample of the accounts that follow the Twitter account of the National Federation of the Blind (@NFB_Voice), a large US-based organization led by blind people. From this sample, we selected an initial 100 users who self-identified as blind or visually impaired in their Twitter profile description. There is no definitive indication that these users also use a screen reader, but we assume that many do.

We wanted to understand if the timelines of these users had the same level of accessibility as Twitter as a whole, or if the accounts they follow posted accessible content more frequently. For each user, we collected all of the accounts they follow, known as "friends" on Twitter[1]. For 6 users, we were unable to retrieve this data, as the accounts they follow were private. For each friend of the 94 remaining accounts, we collected 200 of their most recent tweets (and retweets), or as many as were available. Using these tweets sorted chronologically, we recreated each user's Home timeline for one day in October 2018.

Table 2: Summary of the tweets in one day of recreated Home timeline for 94 blind Twitter users.

	Min	Max	Average	Median
Total friends	6	5,001	720.7	371
Total tweets in timeline	2	29,231	3,379.7	1,554.5
Percent photos in timeline	0.0%	39.7%	18.4%	18.3%
Percent alt Text in photos	0.0%	41.4%	4.6%	2.0%

This is not a perfect recreation of a user's Home timeline, as Twitter does not show every tweet chronologically [29]. Additionally, we could not gather tweets from accounts with higher privacy settings (protected accounts), and some posted tweets may have been deleted by the time we collected them. However, we believe this to be a good approximation of the content these Twitter users would have been exposed to if they logged into Twitter that day.

Overall, we found that the recreated timelines for these users included 18.4% of tweets with photos on average, and 4.6% of photo tweets contained image descriptions. Table 2 contains information about the range of content we observed in these users' timelines, and a visual depiction is shown in Figure 3. In general, these timelines were an order of magnitude more accessible than Twitter as a whole, indicating that these users may be involved in communities with more awareness of the image description feature. An alternative explanation is that these users chose not follow some accounts that post inaccessible images. Regardless of the explanation, while

[1]This does not indicate a mutual relationship, as it does on many other social networks.

these timelines are more accessible than our random sample of Twitter as a whole, they were still largely inaccessible.

5 QUALITY OF IMAGE DESCRIPTIONS

After investigating the prevalence of image descriptions on Twitter and finding low usage of the feature, we were curious about the quality of descriptions that do exist. Do authors write image descriptions in a way that is useful for people with vision impairments? Are the descriptions relevant to the photos?

To answer these questions, we developed a four-point rating scale from "Irrelevant" to "Great description" to assess the quality of image descriptions on Twitter posts (Figure 4). We filtered our sample of original photo tweets with descriptions to those in the English language (based on the "lang" attribute in the Tweet metadata), as we could not effectively assess non-English descriptions. This left 93 tweets from 71 users. To get a larger sample of tweets, we downloaded all tweets from these users with alt text.

5.1 Evaluating Image Descriptions

Prior work by Salisbury *et al.* [25] constructed conversations between crowd workers who were not allowed to view the image in a tweet, and crowd workers who were allowed to and expected to describe the image. From this, they built a set of structured questions to help guide composition of alternative text. We merged these questions and guidance from the Web Content Accessibility Guidelines [10] to develop a rating scale for image descriptions on Twitter. We used a rubric to rate the quality of image descriptions, and examples are shown in Table 4.

5.2 Findings

Two of the researchers used the rubric to redundantly code the quality of 500 photo tweets. We estimated the inter-rater reliability for this set by calculating a weighted Cohen's Kappa of 0.83 for these 500 ratings, which can be seen as strong agreement [20]. One of the researchers then rated an additional 500 photo tweets, resulting in a total of 1000 rated tweets.

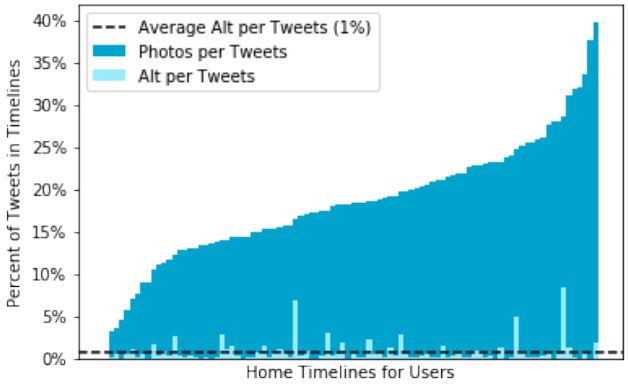


Figure 3: Each timeline analyzed, with the y-axis being percentage of total tweets. Dark blue: photos as a percent of tweets; lighter bars: alt text as a percent of tweets.

Using this set of 1000 rated tweets, we found that 62.6% of alt text rated as irrelevant or somewhat relevant. Only 15.8% rated as "great", with most elements of the image described. More details are available in Table 3.

5.3 Frequency of Use

Some accounts included in our sample used alternative text often, while others only had included a description in a small fraction of the images they tweeted. We looked at all of the images each account had posted since they first used the image description feature. On average, accounts included image descriptions for 60.5% (median = 66.67%) of images tweeted. Many of the accounts who included image descriptions 90-100% of the time were automatically posting content as bots, however. After manually removing 19 of these accounts (Section 5.4, we found that humans tended to write image descriptions for 49.8% of their photos (median = 50%).

5.4 Quality of Bots vs. People

From this analysis, we noticed that many of the tweets with image descriptions were coming from the same accounts, and these accounts were either explicitly bots or exhibited bot-like behavior. The accounts we knew were explicitly bots indicated they were automated in their name or profile description. Some bots generated specific kinds of memes, such as taking images from museum

Table 3: Percentage of image descriptions rated using our rubric for different samples.

Rating	Sample 1	Sample 2 (no bots)	Combined
Irrelevant	32.6%	1.1%	16.9%
Somewhat relevant	30.0%	15.9%	23.0%
Good	21.6%	34.1%	27.9%
Great	15.8%	48.9%	32.4%
Sample Size	1000	1000	2000

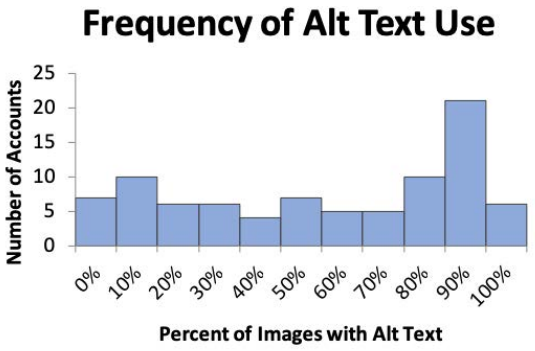
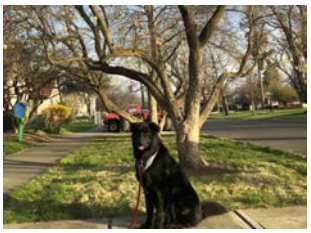


Figure 4: Histogram showing the frequency of image descriptions posted for the accounts in our sample. There is a spike in the 90-100% range that is comprised mostly of bots.

Table 4: Alt-Text Rating Examples. *Links and Usernames are removed from post text for anonymity.

Rating	Image	Post Text & Image Description	Rating and Reason
Irrelevant to image		**Post:** New Music Video - Beach House (@[username]) "Black Car": [link][link] **Alt:** Beach House	We rated the alt-text for this image as **irrelevant** because it does not describe anything in the actual image shown.
Somewhat relevant to image		**Post:** Trailering training class in Henderson yesterday. **Alt:** Trailering training class in Henderson yesterday.	We rated the alt-text for this image as **somewhat relevant** because it relates to the purpose of the image but just repeats the post text without describing more of the image.
Good: some aspects of image described		**Post:** Collecting all the shirts #DNNSummit **Alt:** Three t-shirts from DNN summit.	We rated the alt-text for this image as **good** because it describes most of the things in the image. It could also describe the color and/or what is on each shirt.
Great: almost everything is described		**Post:** "Taking a moment to appreciate beauty. #DogsOfTwitter @[username] #beauty #spring" **Alt:** Styx the dog sits in front of a tree with plastic Easter eggs dangling from the branches. It's very pretty.	We rated the alt-text for this image as **great** because it describes almost everything in the image. This particular description also points out information that might not be immediately apparent visually, either.

archives and superimposing fictional characters on them. Others posted automated information about location-based information such as earthquakes or air-quality, and included maps with their posts. Other accounts did not explicitly state they were bots but exhibited bot-like behavior such as blogs that re-posted articles from their website to drive traffic there from Twitter.

These bots exhibited different levels of accessibility. Some included information that described the entirety of a generated image. Others, such as blog-associated accounts, included human-written alt text for the images in their articles. Overall, however, we saw a general trend of descriptions associated with bot posts receiving a score of "Irrelevant" or "Somewhat relevant".

As we were interested in human-written alternative text, in addition to overall accessibility on Twitter, we sampled an additional random 1000 tweets from our users that excluded the 19 accounts that were explicit or suspected bots. One member of the research team rated this additional sample, finding that just under half of the sample rated as "Great" (48.9%) and 34.1% as "Good". The complete numbers of these samples are available in Table 3.

The difference between these two samples is stark, and indicates that the majority of poor image descriptions may be generated from automated sources. The human tweet authors in our sample described images at a moderately useful level ("good") or higher. Bots are currently important and lively sources of content in the Twitter ecosystem, and their content should be just as accessible to people with vision impairments. These results for human authors are promising, and show that many users write good image descriptions, if they enable and use the feature.

6 DESCRIPTION AUTHOR INTERVIEWS

We interviewed people who had used the image descriptions feature to add alternative text to images in their tweets. Prior work has demonstrated a clear accessibility need on social media platforms from the perspective of people with disabilities [6, 24, 34]. Twitter users that create content must be able to make their images accessible, as the responsibility of making Twitter and other social media platforms accessible should not fall solely on people with disabilities. We were interested in users' motivations for using the feature, their process for composing image descriptions for tweets, and why they may not always add descriptions to their tweets.

6.1 Participants

We interviewed 20 Twitter users who had written at least one tweet including an image and description. Participants were recruited via direct messages or emails to users identified in our sample from Section 5: Quality Ratings. We also recruited participants via an advertisement on Twitter posted by one of the authors. The mean age for participants was 41.6 years (std. dev.= 11.7) with a

Table 5: Participant Demographics.

PID	Age	Gender	Years On Twitter	Occupation
P1	32	male	10	Make and sell jewelry
P2	32	male	10	Client services
P3	45	male	10	Network Administrator
P4	38	woman	9	Translator
P5	48	female	10	Public Engagement
P6	61	female	10	Dir. of IT Accessibility
P7	22	male	7	Research Fellow
P8	25	male	7	Ph.D student
P9	28	female	4	Research Associate
P10	42	female	9	Photographer
P11	44	female	9	Student
P12	32	male	4	Software Engineer
P13	57	female	9	Retired Hydro-geologist
P14	48	male	6	Construction Manager
P15	45-50	trans	9	Scientist
P16	43	male	12	Software Engineering
P17	55	female	12	Marketing
P18	59	male	8	Engineer
P19	48	male	9	Musician, Developer
P20	32	female	9	Marketing

range of time using Twitter from 4 to 12 years (very early adopters). None of the participants reported any visual impairments or use of screen reader technologies. Thus, participants in our study do not represent visually impaired users on Twitter, but rather represent the creators of content that visually impaired users might consume.

6.2 Interview Format

We conducted interviews in person, over the phone, and through Twitter direct messages. The interview consisted of four topics: activation and use of the image description feature, interactions with blind or screen-reader users regarding alt text, process of writing and examples of their alt text, and suggested changes for the image description feature on Twitter. Demographic information is listed in Table 5. The interview sections are described below.

Feature Use. We asked participants three questions regarding their use of the image descriptions feature: (1) why they activated the feature, (2) how they decide when to add descriptions, and (3) and how often they believe they add descriptions to the images they tweet. Our aim was to understand what motivated these users to activate the feature, how actively they provide image descriptions, and for what purpose.

Interactions with Followers. We asked participants about their interactions with followers who were blind or used screen readers. Specifically, we asked about any direct interactions participants had and how many of their followers they believed benefited from image descriptions.

Examples of Image Descriptions. We discussed participants' process for composing image descriptions for their tweets. We first asked participants to describe the elements they included in their descriptions and what they think about when composing them. Next, we chose specific examples from participants' tweets to discuss; one where they wrote alt text and one where they did not. We discussed how they wrote the specific image description for the example and reasons they may not have added descriptions for other images.

Changes to Image Description Feature. Finally, we asked participants to think of one thing they would change about the Twitter image descriptions feature.

Data and Analysis. Each interview was transcribed and analyzed using a theoretical approach to thematic analysis [7] based on the sections of the interview described above. We coded each transcripts based on the topics of the interview. The first and second authors redundantly coded the first five interviews, then discussed and refined the code book before independently coding the remaining 15 transcripts.

6.3 Findings

6.3.1 *Feature Discovery, Motivation, and Use.*

Discovery. Participants discovered the image description feature through a variety of ways. The most mentioned (6) was by the suggestion of someone they were following or a tweet they saw mentioning the capability to add descriptions to images on Twitter.

> *"Because [a specific user I follow] suggested it, I suspect. I can't say for certain. I would never have looked into the accessibility settings if someone hadn't said that this feature was hiding there." – P3*

Some participants (2) discovered the image descriptions feature through announcements, presumably made by Twitter, when the feature was released. Three participants anticipated the release of the image description feature before it was announced. P1 had even requested that Twitter add the capability, almost two years in advance of its release.

> *"I'd been waiting for it to be an option for a while - me and friends had contacted Twitter about it but nothing happened for a while. So as soon as it became available I switched it on...Quite a few of us have disabilities so we try to push for accessibility features like captioning/transcripts or image description." – P1*

Three other participants mentioned they activated the feature because someone else had used alternative text to describe images or they were involved in accessibility related work or communities.

> *"I learned about it through [work] colleagues and wanted folks using screen readers to be able to access media, too." –P15*

Motivation. In terms of why they chose to use the feature, some participants cited that adding alternative text to images was a low effort activity, but would have a high impact on the people it was intended for.

> *"I knew I wanted to use it because it's such a simple small thing that is no effort for me, but may mean a lot to the reader." – P4*

Four participants cited personal or professional connections to someone with a disability as motivation for their use of alternative text. P18 described their professional connections:

"I ran [several] - all accessibility-related accounts. It would have been disingenuous to have the disability community as our majority base of followers and not include alt-text, so we did as soon as it was available. Same for my personal account since I am in that field and was perhaps more aware of it than others might be." – P18

Two ther participants, like P4 and P7, were influenced more personally by their relationship with blind users:

"Maybe subconsciously because a close friend of mine is visually impaired and I used to describe some pictures I find on twitter to him over lunch table conversations but now since we don't meet a lot, this is one way I can keep my friend engaged with my conversations." – P7

Overall, participants demonstrated that, while perhaps not personally connected, providing image descriptions was a matter of "inclusion" that makes things better for everyone.

Habits of Use. With regard to participants' intended use of the feature it became clear that most (12/20) participants explicitly intended to add image descriptions to every image they posted. When we asked how often participants actually used alt text, we found that people experienced varying levels of success:

"Ok I'm not as good at doing alt-text [as] I thought. In my last 10 images, 5 had alt text." – P8

For others, adding the descriptions was a matter of convenience, citing that the applications they used either allowed it or made it straightforward to add the descriptions. Two participants described that when they had time, they would add image descriptions:

"Maybe its really the time which is the factor, when I feel like I am in a rush, I kind of miss adding it to alt-text and instead just describe it in the tweet text."–P7

Two other participants described being reminded by the interface (see figure 2) to add a description.

"The add a description prompt under the photo [which appears once users have activated the ability to write alt text in their accessibility settings] reminds me. I think I do it for every photo I add."–P11

We had two participants, P2 and P19, who created bots that generated original tweets with images that included descriptions. Both bots generated meme-like images by imposing content from online image archives with other images. The alt-text in both cases was just the description of the image that was in the library or museum archive. For these bots, including alternative text as part of the tweet metadata was supported by the Twitter API, thus making the inclusion of a description for every image very convenient.

One participant highlighted that they try to add image descriptions whenever the image is their own photo, whether it is a photo they took or a graphic they created. One participant, P20, mentioned that she switched to using the Twitter web interface or mobile app specifically because another third party tool she was using did not support adding image descriptions. This was similar to P6's professional commitment:

"If I include an image, I add alt text. If I don't feel like bothering, I don't include a picture. Professional pride" – P6

6.3.2 Follower Knowledge and Interaction. Half of participants (10/20) had no knowledge of any of their followers being blind or using screen readers. A majority of participants (11/20) had never been contacted by a blind person regarding their use of alt text.

"I have put stuff in my alt text to try to solicit a response and never gotten one, so I am not sure any of my twitter followers use screen readers." –P15

The other half of participants (10/20) were aware of at least one blind individual among their followers. A few (8) estimated specific numbers between 1 and 40, or "a small percentage". Participants had limited Interactions with their networks regarding alt-text; however, one participant mentioned having discussions where alternative text was mentioned.

6.3.3 Authoring Image Descriptions.

Description Process. We asked participants to share their process for writing image descriptions, both in general and in relation to a specific example of one of their own tweets. Some participants had specific strategies for describing an image. Depending on the intent of the post (mentioned explicitly by 6 participants), a majority of participants (11/20) described writing a general description of the image. One participant imagines trying to explain an image to a friend on the phone.

One participant mentioned describing the colors that appeared in the image. Others described determining the importance of objects, background, and other content in the image that the reader may not be able to ascertain from the main text of the tweet. Participants mentioned transcribing the text included in images or describing the objects, actions, and facial expressions in the image. Based on the content in the image, three participants tried to highlight the focus of the image.

For bot creators and specific content-focused accounts, participants mentioned that they used the image descriptions to convey the purpose of the tweet. For instance, to describe the important visual elements in the image to represent a joke or meme, or to convey why the image makes the joke funny in the context of the post (examples in Figure 5). We encountered two notable examples of content creators that had difficulty writing descriptions, a photographer and a bot creator with accounts that posts memes. P10 grappled with trying to describe the important photographic elements of her images:

"When it's a photograph being shown as a piece of art, that's where it gets difficult - especially since many of my photographs are quite abstract and tend to defy description! I try to touch on the straightforward visual facts of it (what is it a photograph of) but also get across the sense & feel of it where I can. The latter is in some ways more important with my photography. Things like colour tones (is it cool or warm, soft colours or vivid colours), are there any textures, what does it resemble." –P10

P2, who creates Twitter bot accounts, described nuances of conveying jokes presented through memes:

"[The first bot] deviates from my personal answers since its alt-text is the punchline and doesn't describe the visual content of the meme. That can be hard. For [these tweets] the verbosity is the joke, so I used [the alt-text] to convey that verbosity. For



(a)



(b)

Figure 5: The alternative text for (a) was: "Cartoon man in glasses holding book looking at a butterfly labeled 'microbes', asking "Is this ponies" For (b) it was: "Monochrome photograph of a number of dandelion seeds tangled together - much of the focus is soft, the the seed heads and fluff are clearly visible in places. There are hints of blue colour tone in the background."

> *[A different bot account] the joke is both that the character looks intelligent and is making a foolish mistake...the alt text template there mentions the book, the glasses, and then names the two disparate concepts [in the image]." –P2*

All of these examples illustrate a very nuanced process. The approaches vary from person to person and with the intent of the post and image.

Non-Use. Participants cited time constraints as the most common reason they might have missed adding alternative text, either because it was too time consuming, they were in a hurry and forgot (8), or they had to write multiple tweets in succession and missed adding alt-text in the process (5).

Other reasons were that participants simply forgot what the feature was for or that the feature was there. We encountered four participants who mentioned that they relied on the Twitter interface to remind them to add the description.

> *"Actually, looking at twitter on web, the cue for alt-text is "description" which sounds pretty optional at a glance. there's a chance i read "description" and forgot it was for the visually impaired in my rush to tweet and accumulate all the engagement. – P8*

Some participants stated that adding image descriptions from a mobile device was still not possible. However, it is currently possible to add descriptions from the Twitter mobile application. This suggests that either participants did not notice the feature or have not used the application since the capability was added.

6.3.4 Changes. The most common suggestion, mentioned by eight participants, to improve the image description feature on Twitter was to make the descriptions visible to sighted users, especially for their own tweets. P4 states:

> *"[I want] to be able to add the alt text after the tweet is posted. I don't need to edit tweets as a whole, but I would really want to be able to go through all my pics and add alt text. And I wish hovering over images would show what alt text I wrote, like they used to with other images online. Maybe more people would notice/become aware then?"–P4*

The only current process for viewing these descriptions requires the user to view the source code for the web page. This suggestion is also in line with the desired ability to edit the tweet content (and image description) after it has been posted, which is typically not supported on the platform.

The second most common suggestion (by 5 participants) was to make the image description feature "active" by default instead of as a setting that you have to turn on.

> *"I would make it automatically enabled for people so that users don't have to wade through the settings to turn it on. They could be helping so many more people if only they used this feature up front."–P20*

Other suggested changes included improving the interface reminder, addressing bugs, and providing automated support to generate image descriptions. Participant 1 also mentioned the need to increase and "normalize" the use of the feature in a similar manner to captioning. In addition to support from Twitter, one participant (P11) wanted access to individuals with vision impairments to provide guidance on writing good image descriptions. Finally, three participants mentioned increasing the character limit (currently 420 characters) for the alternative text to allow more thorough descriptions, especially for screenshots or pictures of text.

7 DISCUSSION

Our analysis of image description on Twitter revealed that very few people (including popular and government accounts) use the feature, with less than 0.1% of original image tweets having any descriptions at all. The image tweets exposed to blind users contain descriptions slightly more often (4.6%), but still are very inaccessible. The Twitter users that do use the feature author descriptions for about half of the images they tweet, and the descriptions they write tend to be "good" or "great" 83% of the time (bots excluded). Our analysis of interviews with image description writers examined the reasons for use (and non-use) of the feature, and lead us to two paths for improvements: those that Twitter (or similar social platforms) could currently undertake, and those that require further research and additional tools.

7.1 Improvements for the Image Description Feature

Interview participants identified many issues with the image description feature that, if fixed, would lead to higher or better usage. Participants requested that alternative text be visible to them and editable after posting. Interview participants had trouble recalling which images they added descriptions to, and what they wrote. The most common reason they did not add a description was that they forgot when posting quickly, and being able to add image descriptions after the fact would be valuable. Taken further, if users could add image descriptions to retweeted images, volunteers or friends of screen reader users could then make this content accessible.

Another common request was for Twitter to just turn on the ability to author image descriptions for everyone, rather than requiring users to find and enable this capability.

> *"Top wish: it should be turned on by default. It's almost like they're trying to hide it." –P6*

Most participants turned on the feature as soon as they found out about it, and try to include image descriptions for every tweet with images that they post. We agree that the image description feature is hard to find and understand, and Twitter should enable it for everyone to increase accessibility on its site.

However, just enabling the feature for everyone is likely not enough. It is currently designed for people who are familiar with image descriptions for people with vision impairments, and instantly enabling it for everyone could lead to misuse. It may be abused to make tweets appear higher in Twitter or external searches or include spam URLs. We have already observed users including subtle messaging in Twitter image descriptions for followers who know to look for it. Even without intentional abuse, new users who do not understand the purpose of the feature or how to write alt text may not produce high quality image descriptions. Twitter should provide clear on-boarding instructions for users when they first use the feature, explaining its purpose and why users should provide image descriptions for their images. In order to reduce confusion when posting a new image tweet, on image upload Twitter should provide instructions on how to write alt text or a template of structured questions, which prior work has found results in higher quality alt text in other media (*e.g.,* STEM textbook diagrams) [23].

7.2 Additional Tooling and Training for Users

We see two major opportunities for researchers to make social media platforms more accessible through additional tools.

Some content on Twitter is ripe for auto-generation of image descriptions. Automatic captions for generic images has been exemplified by ALT text bot [11], which provides automated alt text in response to tweets containing images. However, researchers could go further in areas where the content format is more constrained. Some photos are just photos of tweets, and could be accessible if linked to the original tweet. Screenshots or photos of text are popular (9.7% of the no-bot sample in Section 5), and robust optical character recognition could make these accessible.

Researchers should also develop tools to help users write better image descriptions. Many users do not know what elements to include in an image, and would benefit from specific instructions, such as the structured questions developed in Salisbury *et al* [25]. Automated tooling could rate how descriptive alt-text is, provide specific instructions based on recognized objects in an image, or even pre-fill the image description with an auto-generated scene description [25, 37]. This may help prompt the user to change or refine the image description before publishing.

7.3 Supporting Authorship through Automated Feedback

To provide a starting point for the goal of creating tools for image description authors, we used the sample rated in Section 5 to create an automatic rater for image descriptions. We merged ratings of "Irrelevant" and "Somewhat relevant" alt text as "low-quality" and ratings of "Good" or "Great" as "high-quality". An Extra Trees classifier [13] was trained on features extracted from a subset of the sample (1,320 tweets). The specific features used were: counts the of parts of speech in the alt text and post text, shared words between the alt text and post text, as well as the length of the alt text. This classifier achieved an overall accuracy on the remaining 680 tweets of 85.3% (AUC = 0.84, precision = 0.83, recall = 0.94), demonstrating it is able to distinguish between much of the alt text quality. The five most important features for this classifier were: number of prepositions in alt text, number of words shared between post text and alt text, length of alt text, number of present verbs in alt text, and number of plural nouns in alt text.

At a very simple level, a classifier like this demonstrates that automatic feedback could be given to users, when they compose their descriptions, on whether it appears to be low or high quality. Specific feedback could focus on how similar the alt text is to the post text, objects in the image that are not mentioned, or a lack of actions (verbs) and objects (nouns) in the written description.

7.4 Limitations

The primary limitation of this work is that we have not comprehensively studied the accessibility of Twitter as a whole, only photo tweets. Other forms of media, such as animated GIFs, videos, polls, and URL previews exist. URL previews in particular can contain alternative text that is pulled from the linked page, but this was not the subject of our analysis.

Our rating scale was developed from prior research on the experience of blind users interacting with alt text, but it was still designed and executed by sighted researchers rating image tweets, Therefore, it does not fully reflect what screen reader users seek when browsing images on Twitter.

8 CONCLUSIONS

Our findings show that image descriptions are very rarely provided on Twitter. This is in large part due to very few users having the feature turned on, but even the users who enabled the ability to provide alternative text descriptions did not always write them. Twitter would likely dramatically improve its overall accessibility if it encouraged all users to provide descriptions and enabled them to do so across all types of media. Access to social media is increasingly important for participation in many aspects of society, including social connection, entertainment, civic participation, and news consumption. By improving descriptions of visual content on social media networks, many users with vision impairments [34] will again have equal access to these vital platforms. As P15 states:

> *"Only built-in accessibility from the start provides more equitable access. Only doing it because/when someone asks for it or because we know a specific individual needs it, puts the burden on the person with the need, and that's not how accessibility should work. And in terms of accessibility online, alt text for static imagery is a low hanging fruit, easy and inexpensive to implement. Failure to do it is just inconsiderate laziness."–P15*

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. (DGE 1745016). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

In addition, this work was developed under a grant from the National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR grant number 90DP0061). NIDILRR is a Center within the Administration for Community Living (ACL), Department of Health and Human Services (HHS).

REFERENCES

[1] Adobe. 2018. Add alternative text. https://www.adobe.com/accessibility/products/acrobat/pdf-repair-add-alternative-text.html

[2] Apple. 2018. Supporting VoiceOver in Your App. https://developer.apple.com/documentation/uikit/accessibility/supporting_voiceover_in_your_app

[3] Julian Ausserhofer and Axel Maireder. 2013. National politics on Twitter: Structures and topics of a networked public sphere. *Information, Communication & Society* 16, 3 (2013), 291–314.

[4] Tim Berners-Lee and Dan Connolly. 1995. HTML 2.0 Specification. *W3C: http://www. w3. org/MarkUp/html-spec* 34 (1995).

[5] Jeffrey P Bigham, Ryan S Kaminsky, Richard E Ladner, Oscar M Danielsson, and Gordon L Hempton. 2006. WebInSight: Making Web Images Accessible. In *Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility - Assets '06*. 181. https://doi.org/10.1145/1168987.1169018

[6] Erin L. Brady, Yu Zhong, Meredith Ringel Morris, and Jeffrey P. Bigham. 2013. Investigating the appropriateness of social network question asking as a resource for blind users. In *Proceedings of the 2013 conference on Computer supported cooperative work - CSCW '13*. ACM Press, New York, New York, USA, 1225. https://doi.org/10.1145/2441776.2441915

[7] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (jan 2006), 77–101. https://doi.org/10.1191/1478088706qp063oa arXiv:arXiv:1011.1669v3

[8] C-SPAN. 2018. @cspan/Members of Congress on Twitter. https://twitter.com/cspan/lists/members-of-congress [Online; accessed 2-November-2018].

[9] W3 Consortium et al. 1998. *HTML 4.0 specification*. Technical Report. Technical report, W3 Consortium, 1998. http://www. w3. org/TR/REC-html40.

[10] W3 Consotrium. [n. d.]. Web Content Accesisbility Guidelines (WCAG) 2.1. https://www.w3.org/TR/WCAG21/

[11] Cameron Cundiff. 2015. alt-text-bot: automatic text descriptions of images on Twitter. https://devpost.com/software/alt-text-bot

[12] Daniel Dardailler. 1997. The ALT-server (" An eye for an alt").

[13] Pierre Geurts, Damien Ernst, and Louis Wehenkel. 2006. Extremely randomized trees. *Machine learning* 63, 1 (2006), 3–42.

[14] Google. 2018. Content labels. https://support.google.com/accessibility/android/answer/7158690?hl=en

[15] Darren Guinness, Edward Cutrell, and Meredith Ringel Morris. 2018. Caption Crawler: Enabling Reusable Alternative Text Descriptions Using Reverse Image Search. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 518, 11 pages. https://doi.org/10.1145/3173574.3174092

[16] Stephanie Hackett, Bambang Parmanto, and Xiaoming Zeng. 2004. Accessibility of Internet Websites Through Time. In *Proceedings of the 6th International ACM SIGACCESS Conference on Computers and Accessibility (Assets '04)*. ACM, New York, NY, USA, 32–39. https://doi.org/10.1145/1028630.1028638

[17] Xiaodong He and Li Deng. 2017. Deep Learning for Image-to-Text Generation: A Technical Overview. *IEEE Signal Processing Magazine* 34, 6 (nov 2017), 109–116. https://doi.org/10.1109/MSP.2017.2741510

[18] Instagram. 2018. Creating a More Accessible Instagram. https://instagram-press.com/blog/2018/11/28/creating-a-more-accessible-instagram/

[19] @JohnMu. 2018. Alt text is extremely helpful for Google Images – if you want your images to rank there. Even if you use lazy-loading, you know which image will be loaded, so get that information in there as early as possible & test what it renders as. https://twitter.com/JohnMu/status/1036901608880254976

[20] Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica: Biochemia medica* 22, 3 (2012), 276–282.

[21] Microsoft. 2017. Seeing AI | Talking camera app for those with a visual impairment. https://www.microsoft.com/en-us/seeing-ai/

[22] Microsoft. 2018. Add alternative text to a shape, picture, chart, SmartArt graphic, or other object. https://support.office.com/en-us/article/add-alternative-text-to-a-shape-picture-chart-smartart-graphic-or-other-object-44989b2a-903c-4d9a-b742-6a75b451c669

[23] Valerie S. Morash, Yue-Ting Siu, Joshua A. Miele, Lucia Hasty, and Steven Landau. 2015. Guiding Novice Web Workers in Making Image Descriptions Using Templates. *ACM Trans. Access. Comput.* 7, 4, Article 12 (Nov. 2015), 21 pages. https://doi.org/10.1145/2764916

[24] Meredith Ringel Morris, Annuska Zolyomi, Catherine Yao, Sina Bahram, Jeffrey P. Bigham, and Shaun K. Kane. 2016. "With most of it being pictures now, I rarely use it". In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. 5506–5516. https://doi.org/10.1145/2858036.2858116

[25] Elliot Salisbury, Ece Kamar, and Meredith Ringel Morris. 2017. Toward Scalable Social Alt Text: Conversational Crowdsourcing as a Tool for Refining Vision-to-Language Technology for the Blind. *Aaai Hcomp 17* Hcomp (2017), 147–156. whttps://www.microsoft.com/en-us/research/wp-content/uploads/2017/08/scalable_social_alttext.pdf

[26] Hironobu Takagi, Shinya Kawanaka, Masatomo Kobayashi, Takashi Itoh, and Chieko Asakawa. 2008. Social Accessibility: Achieving Accessibility Through Collaborative Metadata Authoring. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility (Assets '08)*. ACM, New York, NY, USA, 193–200. https://doi.org/10.1145/1414471.1414507

[27] @todd. 2016. Accessible images for everyone. https://blog.twitter.com/official/en_us/a/2016/accessible-images-for-everyone.html

[28] Andranik Tumasjan, Timm Oliver Sprenger, Philipp G Sandner, and Isabell M Welpe. 2010. Predicting elections with twitter: What 140 characters reveal about political sentiment. *Icwsm* 10, 1 (2010), 178–185.

[29] Twitter. 2018. About your Twitter timeline. https://help.twitter.com/en/using-twitter/twitter-timeline

[30] Twitter. 2018. How to make images accessible for people. https://help.twitter.com/en/using-twitter/picture-descriptions

[31] Twitter. 2018. Post, retrieve and engage with Tweets. https://developer.twitter.com/en/docs/tweets/post-and-engage/api-reference/get-statuses-lookup.html

[32] Twitter. 2018. Sample realtime Tweets. https://developer.twitter.com/en/docs/tweets/sample-realtime/guides/recovery-and-redundancy

[33] Wikipedia contributors. 2018. List of most-followed Twitter accounts — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=List_of_most-followed_Twitter_accounts&oldid=866718146 [Online; accessed 2-November-2018].

[34] Shaomei Wu and Lada A. Adamic. 2014. Visually impaired users on an online social network. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*. ACM Press, New York, New York, USA, 3133–3142. https://doi.org/10.1145/2556288.2557415

[35] Shaomei Wu, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. 2011. Who Says What to Whom on Twitter. In *Proceedings of the 20th International Conference on World Wide Web (WWW '11)*. ACM, New York, NY, USA, 705–714. https://doi.org/10.1145/1963405.1963504

[36] Shaomei Wu, Jeffrey Wieland, Omid Farivar, and Julie Schiller. 2017. Automatic alt-text: Computer-generated image descriptions for blind users on a social network service. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. ACM, 1180–1192.

[37] Yuhang Zhao, Shaomei Wu, Lindsay Reynolds, and Shiri Azenkot. 2017. The Effect of Computer-Generated Descriptions on Photo-Sharing Experiences of People with Visual Impairments. *Proc. ACM Hum.-Comput. Interact.* 1, CSCW, Article 121 (Dec. 2017), 22 pages. https://doi.org/10.1145/3134756

Characterizing Speed and Scale of Cryptocurrency Discussion Spread on Reddit

Maria Glenski
Computer Science and Engineering
University of Notre Dame
Notre Dame, Indiana
mglenski@nd.edu

Emily Saldanha
Data Sciences and Analytics Group
Pacific Northwest National Laboratory
Richland, Washington
emily.saldanha@pnnl.gov

Svitlana Volkova
Data Sciences and Analytics Group
Pacific Northwest National Laboratory
Richland, Washington
svitlana.volkova@pnnl.gov

ABSTRACT

Cryptocurrencies are a novel and disruptive technology that has prompted a new approach to how currencies work in the modern economy. As such, online discussions related to cryptocurrencies often go beyond posts about the technology and underlying architecture of the various coins, to subjective speculations of price fluctuations and predictions. Furthermore, online discussions, potentially driven by foreign adversaries, criminals or hackers, can have a significant impact on our economy and national security if spread at scale.

This paper is the first to qualitatively measure and contrast discussion growth about three popular cryptocurrencies with key distinctions in motivation, usage, and implementation – Bitcoin, Ethereum, and Monero on Reddit. More specifically, we measure how discussions relevant to these coins spread in online social environments – how deep and how wide they go, how long they last, how many people they reach, etc. More importantly, we compare user behavior patterns between the focused community of the official coin subreddits and the general community across Reddit as a whole. Our Reddit sample covers three years of data between 2015 and 2018 and includes a time period of a record high Bitcoin price rise.[1]

Our results demonstrate that while the largest discussions on Reddit are focused on Bitcoin, posts about Monero (a cryptocurrency often used by criminals for illegal transactions on the Dark Web[2]) start discussions that are typically longer and wider. Bitcoin posts trigger subsequent discussion more immediately but Monero posts are more likely to trigger a longer lasting discussion. We find that moderately subjective posts across all three coins trigger larger, longer, and more viral discussion cascades within both focused and general communities on Reddit. Our analysis aims to bring the awareness to online discussion spread relevant to cryptocurrencies in addition to informing models for forecasting cryptocurrency price that rely on discussions in social media.

[1]http://fortune.com/2017/12/17/bitcoin-record-high-short-of-20000/
[2]https://www.deepdotweb.com/2017/09/20/dhs-says-darknet-criminals-switching-bitcoin-monero/

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313702

CCS CONCEPTS

• **Networks** *Online social networks*.

KEYWORDS

discussion spread; information cascades; social networks; Reddit

ACM Reference Format:
Maria Glenski, Emily Saldanha, and Svitlana Volkova. 2019. Characterizing Speed and Scale of Cryptocurrency Discussion Spread on Reddit. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313702

1 INTRODUCTION

Cryptocurrencies are an emerging and disruptive technology. The coins themselves, the underlying software, and the surrounding environment of social and trading behavior are highly volatile and evolve quickly. Cryptocurrencies are disproportionately used by criminals and hackers, and their use has political and economic implications. The way information about these novel distributive technologies spreads across online social platforms shapes online and offline discussions, and could potentially influence peoples' beliefs and decision making. Thus, it is crucial to understand, explain, and anticipate the social behavior and communication patterns in the social environments surrounding cryptocurrencies to understand this phenomena and devise appropriate responses [22, 30, 35].

A preliminary analysis of potential cryptocurrencies of interest identified Bitcoin (BTC), Ethereum (ETH), and Monero (XMR) as the top three cryptocurrencies in terms of both developer interest on GitHub and Bitbucket and community interest on social media platforms – Twitter, Reddit and Facebook as illustrated in Figure 1. For that analysis, we collected data from CoinGecko for 1,742 cryptocurrencies.[3] and compared coins in terms of developer interest features (a measure of activity in public repos on GitHub and Bitbucket) and community interest features (a measure of discussions and popularity on social media platforms) released by CoinGecko. *Bitcoin* is a market-leading cryptocurrency, which laid the foundation for blockchain's transparent digital ledger system (blockchain). *Ethereum* is a cryptocurrency as well as a blockchain development platform, where developers may build their own automated smart contracts. *Monero* is a cryptocurrency similar to Bitcoin but with added security and anonymity features which enable more privacy in transactions. Monero is often used by criminals for illegal transactions on the Dark Web[2].

[3]https://www.coingecko.com/en/coins/all



Figure 1: Cryptocurrencies plotted by developer and community interest according to CoinGecko.

To motivate our analysis we present discussion growth on Reddit between January 2015 and January 2018, covering a period that includes the record high Bitcoin price rise at the end of 2017, in the official r/bitcoin subreddit aligned with related real-world events in Figure 2. We observe a significant increase in Reddit discussion volume during the period of the record Bitcoin price increase. During that time, online discussions could be influenced by adversaries [24] who could potentially manipulate people's opinions [3, 10] regarding buying or selling Bitcoin. The same is true for other cryptocurrencies, for example, we discover high correlations measured as Normalized Mutual Information (NMI) between Reddit discussion volume and coin price for Bitcoin (NMI = 0.64), Ethereum (NMI = 0.89) and Monero (NMI = 0.82).

Our main contribution is a measurement-driven analysis of cryptocurrency discussion spread for three popular coins within two Reddit sub-populations: the *Official Subreddit* communities and a broader *Crypto-Ecosystem* comprised of multiple crypto-related communities. We measure the *speed* of cryptocurrency discussion spread to understand how quickly discussions begin (initial delay or responsiveness), how they grow over time, and how quickly they end (discussion lifetime). We evaluate the *scale* of cryptocurrency discussion spread by measuring discussion volume, audience size, and structural properties of discussion threads *e.g.,* the max-breadth, max-depth, and structural virality [16].

2 RELATED WORK

Information cascades in online social networks (OSNs) are a phenomena in which the idea becomes widely adopted due to influence of social network neighbors and has been widely studied [4, 5, 17, 18, 27, 28, 33]. Researchers presented analyses on how different types of information spread in different social environments e.g., how images spread on Flickr [6, 7], pins (images) on Pinterest [19, 20], videos on YouTube [36], reviews on Yelp [21], URLs on Yahoo [17], posts on Weibo [34], hashtags and URLs on Twitter [26, 31], posts and memes on Facebook [1, 9, 11] etc. Another line of work focused on predicting cascade properties e.g., size, depth [2, 8, 23, 25]

Unlike prior work that focused on how a specific piece of content e.g., an image, URL etc. spreads in a single social environment and what factors drive such information spread, we focus on analyzing threaded discussion cascades relevant to the cryptocurrency domain on Reddit. Our analysis is motivated by the phenomena where online discussion spread could potentially influence (and be influenced) by real-world external events e.g., market crashes and coin prices. The most similar work to ours is [12]. The authors focused on the most popular subreddits and performed the analysis of conversation patterns on Reddit by only considering one measurement of speed (responsiveness), and two scale measurements (volume and virality). We go beyond that: we empirically evaluate and contrast speed and scale of discussion spread related to cryptocurrencies using multiple measurements within the *Official Subreddit* communities and discussions within a broader *Crypto-Ecosystem.*

3 REDDIT DATA

In this study we chose to focus on analyzing discussion spread on Reddit for several reasons. Reddit is a highly popular social news aggregator[4] that allows communities of users to share and discuss information, opinions, and entertainment media. There are 330 million users active in 140 thousand communities on Reddit who post 2.8. million comments daily, and perform 11 million posts and 11 billion page views monthly[5]. Moreover, reconstructing cascades for Twitter and Facebook requires either access to the full data (not just a sample [13]), the collection of all followers to be able to reconstruct the cascades, or the application of other methods for correcting information cascades for missing data [32]. Unlike these social networks, Reddit post and comment data is public[6] and contains full discussion cascades.

Cryptocurrency Official Subreddits. The *Official Subreddit* dataset comprises all discussions posted to r/bitcoin,r/ethereum, and r/monero subreddits over a period of three years from 2015 through 2017. These three subreddits are the official communities for each of our cryptocurrencies of interest and had substantial traffic of, on average, 3.6K, 500, and 380 comments posted each day, for Bitcoin, Ethereum and Monero, respectively. As these communities are explicitly tied to each of the cryptocurrencies, we label each discussion with the subreddit's respective coin. This resulted in a total of 212,302 Bitcoin, 41,792 Ethereum, and 15,035 Monero discussions.

Cryptocurrency Ecosystems. To collect relevant discussions across the Reddit community in general for the *Crypto-Ecosystem* data, we collected the full comment threads for posts that were (a) submitted from January 2015 through August 2017 and (b) contained either keywords related to Bitcoin ('#btc', '#BTC', '#bitcoin', '@bitcoin', 'bitcoin'), Ethereum ('#eth', '#ETH', '#ethereum', '@ethereum', 'ethereum'), or Monero ('#xmr', '#XMR', '#monero', '@monero', 'monero') in the title or text (if the post includes text). This resulted in 38,118 posts and comments for Bitcoin, 8,989 for Ethereum, and 4,381 for Monero that were submitted from January 2015 through August 2017. These discussions are not partitioned by subreddit and, intuitively, discussions can be started by posts that include keywords for multiple coins. To avoid bias from one coin affecting analysis of another, we restrict the dataset to include only those discussions where the initial post included keywords for a single coin.

[4]The 18th most popular site globally and 5th most popular within the U.S. according to Alexa.com: https://www.alexa.com/topsites
[5]https://expandedramblings.com/index.php/reddit-stats/
[6]An archive of Reddit posts and comments is publicly available at https://files.pushshift.io/reddit/ and covers the time period of 2005 through 2018 as of February 2019.

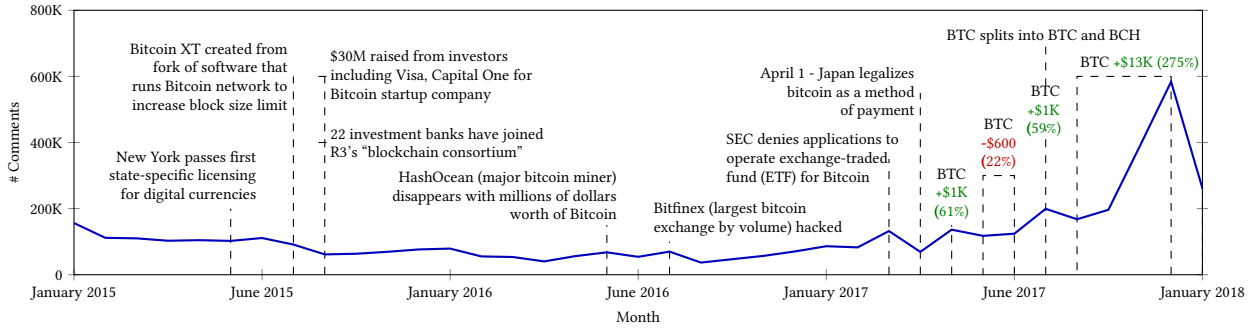


Figure 2: Motivation: discussion growth on Reddit for the r/bitcoin official subreddit related to real-world events and a record high Bitcoin price rise at the end of 2017.

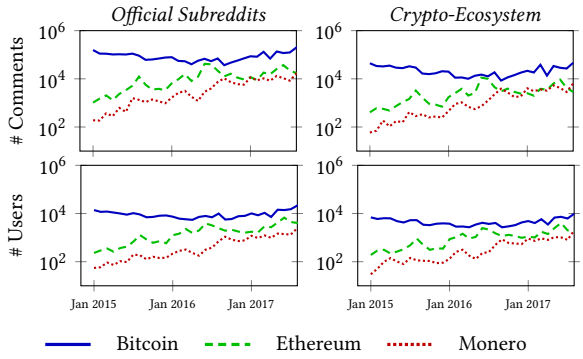


Figure 3: Discussion activity over time as the number of comments (above) and users (below) for each cryptocurrency of interest. Y-axes are presented on a log scale.

This resulted in 35,282 discussions for Bitcoin, 6,949 for Ethereum, and 3,324 for Monero. This restriction did not significantly reduce the sample size overall or for any one coin.

Figure 3 illustrates the volumes of comments posted (above) and users active (below) in cryptocurrency-related discussions each month within two datasets described above. To replicate our results, one could collect comments from the Reddit archive publicly available at https://files.pushshift.io/reddit/ and extract all comments posted to the official subreddits or with related keywords present in the body of the comment, as described above. Interestingly, we see that the volumes of active users and comments posted to the official subreddits, where we did not restrict posts to only those that included the set of keywords, follow very similar trends to the corresponding sets of discussions within the Crypto-Ecosystem dataset.

4 METHODOLOGY

On Reddit, users may contribute to discussions in several ways:

- starting a discussion by authoring a post which must be submitted to a specific community (subreddit),
- posting a comment in reply to a post,
- posting a comment in reply to another comment,
- voting (positively or negatively) on posts or comments.

These discussions can be viewed as tree structures or information cascades (aka comment trees on Reddit) [4, 12, 18] wherein the root node represents the initial post, and other nodes in the tree represent comments to the initial post. We define a discussion cascade as an undirected tree $T = (V, E)$, where V represent all messages (the original post and the follow-up comments in the discussion thread), and E represent the set of edges that connect messages linked by in-reply (or commenting) actions. Figure 4 presents the discussion cascade for an example post and ten comments. We focus on measuring discussion growth using comment trees within focused communities of the official subreddits and across multiple communities in the wider cryptocurrency ecosystems and characterize the speed, scale, and a high-level content analysis of cryptocurrency related discussions on Reddit for three coins of interest – Bitcoin, Ethereum, and Monero. Before studying discussion spread, similar to [5, 27], we discuss the affect of the topological structure of the network, and characterize the population of contributors as outlined below.

First, we examine the *population of contributors* who participated in these discussions in terms of how active they are, *e.g., are we analyzing the behavior of a large population with varied frequencies*

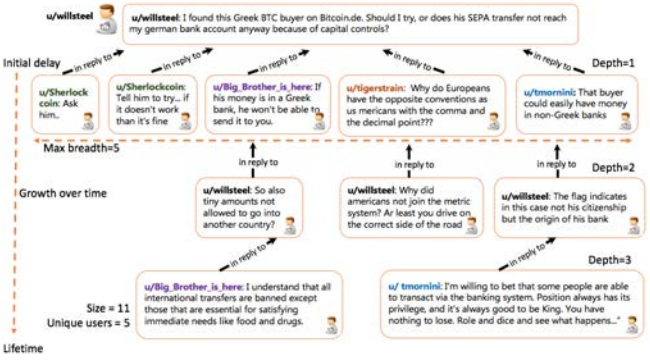


Figure 4: Discussion cascade (aka comment tree on Reddit) illustrated for a post with 10 comments. Discussion speed measurements – initial delay, growth over time, lifetime and discussion scale measurements – volume, audience and structural properties of information cascades are shown.

of activity or a primarily homogenous, densely connected network of highly active users? For that, we look at the mean averages and respective 95% confidence intervals for the volumes of discussions related to each coin that users initiate, *i.e.,* by submitting the initial post, or participating in the discussion through posts and comments.

Next, we examine *topology of user networks* (the forest of cascade trees) constructed created from the reply-links in the discussion threads as shown in Figure 4. We define a reply-link as follows: reply-link <*A*,*B*> exists if user *A* posted a comment in reply to a comment or initial post authored by user *B*. The goal is to determine how closely connected communities contributing to cryptocurrency discussions are, *e.g., are users discussing cryptocurrencies back and forth with the same set of users or engage new users?* The clustering coefficients of the resulting user networks measure the extent to which nodes cluster together in communities within each network.

Finally, we consider the *prevalence of posts that do not trigger any additional commentary* for each coin. We call these posts *ineffective posts*. Conversely, we define *effective posts* as posts that prompt at least one subsequent comment. We compare the relative rate of ineffective posts for each cryptocurrency of interest across both datasets. We examine whether there are consistent patterns in effectiveness of posts related to a given coin or patterns being pairs or sets of coins across datasets. To mitigate bias on measurements of growth and speed analyses from these ineffective posts, we remove them from subsequent analyses in all but one case: when we consider the size of discussions.

4.1 Measuring Speed of Discussion Spread

There are several ways to characterize the speed of discussion growth. In this work we choose to focus on the speed of the three main temporal phases of discussions to characterize discussions related to each coin based on:

(1) how quickly they begin,
(2) the speed with which they grow over time,
(3) how quickly they end.

Initial Delays (aka Responsiveness). The first speed-characteristic we consider the delay between when the initial post is submitted and when the first comment in the discussion is posted in response. This relies on a minimal discussion threshold of two contributions (*i.e.,* one post and one comment). We call this the *initial delay* of the discussion which we operationalize as a measure of the discussion's initial growth rate. As such, we only include discussions that grew beyond the initial post. To compare the average behavior in terms of initial growth, we compare the mean and median values for each coin of interest and determine whether there are statistically significant differences with Mann Whitney U (MWU) tests [29]. Finally, we plot the complementary cumulative distribution functions (CCDFs) for each coin. Complementary cumulative distribution functions plot the percentages of cascades with delays at least as long as a given delay. Through this analysis we characterize how quickly a post triggers a discussion thread.

Growth over time. Next, we consider *the time to reach each depth* in the discussion thread. That is, the delay between submission of the initial, root post and the first comment posted within each depth in the discussion tree. For this measure, we plot the mean average delay to reach each depth with associated 95% confidence intervals for each of the coins. As the mean tends to be influenced by extreme outliers, we also examine the median delay to reach each depth and determine whether there are more robust trends or whether trends remain consistent across both mean and median delays by depth. Similarly, we also compare the mean and median *size by depth*. That is, how quickly (as determined by the depth within the discussion) do the discussions grow at each stage. Through this analysis we identify whether discussions grow consistently or sporadically.

Lifetimes. Thirdly, we consider the *lifetime* which we define as the length of time a discussion is active, *i.e.,* the delay between the submission of the root post and the last comment in the discussion, to characterize how quickly discussions end. Similarly to the analysis we perform related to the initial delays, we compare and contrast the mean and median values and plot the CCDFs for each coin across datasets. Again, we test for significant differences in discussion lifetimes between coins with MWU tests as our approach to partition discussion cascades by coin results in independent sets of discussions.

4.2 Measuring Scale of Discussion Spread

Next, we measure the scale of cryptocurrency discussion growth. We focus on summarizing discussions in terms of the discussion volume, audience size, and structural properties e.g., max breath, max depth and structural virality.

Volume. The first measure of discussion scale we consider is the final size of the discussion, *i.e.,* the final volume of nodes within the discussion tree. Therefore, we define the size as the number of comments plus the initial post that prompted the discussion. As such, *ineffective posts* noted previously are discussions of size 1. We present CCDFs for discussion sizes for each coin and contrast behavior across two datasets. We also present and compare the mean, median, and maximum discussion sizes. MWU tests once again allow us to determine significant differences between coins. Intuitively, we would expect discussions posted to larger subreddits to have a higher maximum size and larger range, in general, for discussions sizes. This is because with larger subreddit sizes (and, thus, more contributors), there are more potential users who may be inclined to contribute to each discussion. However, while in the larger subreddits, there are more potential participants that could drive a discussion to a much larger size, there is also a higher influx of content to the larger subreddits where newer content quickly pushes old discussions down and out of view. This flow of new content may motivate users to join a more recently started or active conversation before they even see the older conversations.

Audience size. We can also consider size in terms of the audience or contributors to the discussion rather than the contributions (posts and comments). We define the population as the number of contributors (users) who participated in a discussion thread by submitting the initial post or a subsequent comment within the thread. We can then summarize the overall population sizes and compare the mean and median values across coins and datasets. Once again, because of the independence of the samples for each coin, we can identify whether significant differences in population sizes exist with MWU tests comparing each of the coins to another.

We also compare the maximum values to consider the upper range of populations who discuss each coin. This analysis provides an overview of the static property, population overall. We also consider the populations within the discussions' cascades and at varying phases of the discussion. To do so, we present the mean and median numbers of users who contribute within a given depth and focus on the initial depths within the discussion: *is there a consistent population contributing regularly or a large population whose contributions are concentrated within a single depth?*

Structural properties. Finally, we consider scale-based characteristics in terms of the structure of the discussion cascades. Here, we compare several static properties of discussion structure:

(1) the *structural virality* or Weiner Index (WI) of the cascade, a measurement of how closely nodes are connected within the discussion [16];
(2) the *max-depth*, *i.e.,* the length of the longest path from the root (initial post in the discussion) to a leaf node in the cascade;
(3) the *max-breadth*, *i.e.,* the largest number of comments within a single depth of the discussion tree.

Again, we compare the CCDF plots across coins and analyze first order statistics for each of the structural properties, with significant differences identified with MWU tests. Max-Depth and Max-Breadth provide insight into the primary dimensions of the cascade individually to identify whether discussions are typically wide or long. Structural virality, on the other hand, provides a more easily interpretable measure of the structure as a whole. Weiner index become minimum when all comments are directly added to the root, and maximum when the tree becomes a chain. In other words, *do discussions follow a "viral" structure of multiple branches off of branches within the discussion or a structure that is more reminiscent of a "broadcast" that is wide but not very deep?* To capture a dynamic measure of the structure as the discussion grew rather than the static properties represented with Max-Depth, Max-Breadth, and structural virality, we also plot the mean average and median breadth by depth. Here, we see how the breadth or width of the discussion cascade grows and shrinks over the initial depths. *Are these patterns of growth monotonic (*i.e., *consistently increasing or consistently decreasing), or do they oscillate between growing and shrinking at increasing depths?*

4.3 Content Subjectivity and Discussion Spread

We focus on the subjectivity of the initial posts and their effect on the discussion spread. To get post subjectivity we rely on TextBlob toolkit[7]. The subjectivity score is a float between 0 and 1, where 0 is very objective and 1 is very subjective. First, we compare the subjectivity of successful versus unsuccessful discussions at a very high level dichotomy: *did the initial post trigger subsequent discussion or did the discussion consist solely of the initial post?* We plot the kernel density estimations (KDE) of what we call *effective posts* that initiated discussions of at least one additional comment and *ineffective posts* that failed to initiate a discussion for each of the coins. We can then compare these KDE plots to identify changes in distributions of subjectivity – *do ineffective and effective posts have significantly different distributions of subjectivity?* Then, to gather a more nuanced view of how the initial post in discussions influence discussion growth, we compare correlations measured using normalized mutual information for each of the static discussion properties (virality, max-depth, max-breadth, size, and population) with the subjectivity of the initial (or "root") post in the discussion.

[7] https://textblob.readthedocs.io/en/dev/

Table 1: Average number of cascades a user initiated, *i.e.,* as the author of the initial post, or participated in as the author of an initial post or subsequent comment.

	Official Subreddits		*Crypto-Ecosystem*	
Coin	Initiated	Participated	Initiated	Participated
Bitcoin	3.79 ± 1.20	11.27 ± 1.19	1.90 ± 0.19	6.74 ± 0.54
Ethereum	3.23 ± 0.96	7.56 ± 1.13	1.80 ± 0.09	4.24 ± 0.42
Monero	2.90 ± 0.75	8.50 ± 1.22	1.84 ± 0.17	5.75 ± 0.76

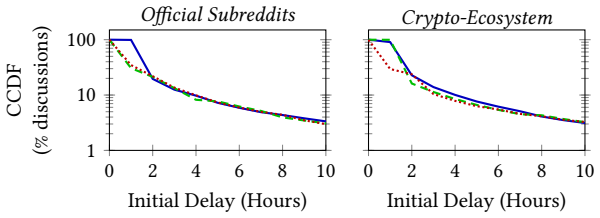
5 RESULTS

Before we measure the speed and scale of information spread in cryptocurrency discussion growth, we first examine several influential characteristics of each dataset and the potential impacts on our analysis. Here, we examine the population of contributors who participated in these discussions, the user-networks for each dataset, and the prevalence of ineffective posts – posts that did not spread.

First, we look at the mean averages and respective 95% confidence intervals for the volumes of discussions related to each coin that users initiate, *i.e.,* by submitting the initial post, or participate in (through posts and/or comments), shown in Table 1. We find that *users are more active within a single subreddit dedicated to* i.e., *the Official Subreddits domain than within discussions focused on the same coin that are started in multiple related subreddits.* Thus, there should be less bias from a single user or group of users on the results of discussion growth in the Crypto-Ecosystem dataset. Intuitively, this composition of contributing users makes sense as users typically browse, subscribe, or otherwise follow specific subreddits based on the topics they cover than individual posts that discuss a topic of interest. Further, a recent study found that users tend to focus their interactions in a small number of subreddits [14].

Next, we consider the networks created with the reply-links to examine how closely connected these communities are. *Are users discussing back and forth with the same set of users or are most interactions with new users?* We find that users interact with, on average, 13 users in the r/Bitcoin subreddit, 8 users in the r/Ethereum and r/Monero subreddits within the Official Subreddits domain. Within the Crypto-Ecosystem domain, users who participate in Bitcoin-related discussions interact with approximately 9 users and we see a similar drop as we do in the Official Subreddits to around 5 and 7 for Ethereum and Monero, respectively. The clustering coefficients are also fairly low for both the Official Subreddits – 0.13 and Crypto-Ecosystem – 0.12 datasets. As shown in Table 1, users, on average, participate in a fairly low number of discussions (between 8 and 11 for Official Subreddits and between 4 and 7 for Crypto-Ecosystem) overall. Users are not densely connected through frequency in discussions or replies to specific users.

Next, we look at the prevalence of posts that do *not* trigger any additional commentary for each coin across datasets. We call these



	Official Subreddits			Crypto-Ecosystem		
coin	mean	median	max	mean	median	max
Bitcoin	4.61	11.17	6.31	5.26	15.25	6.00
Ethereum	7.60	27.60	6.28	8.12	36.37	5.00
Monero	2.85	20.98	4.38	2.56	23.36	3.00

Figure 5: CCDFs for the initial delays of discussions (above) and the mean delay in hours, the median delay in minutes, and the max delay in months (below).

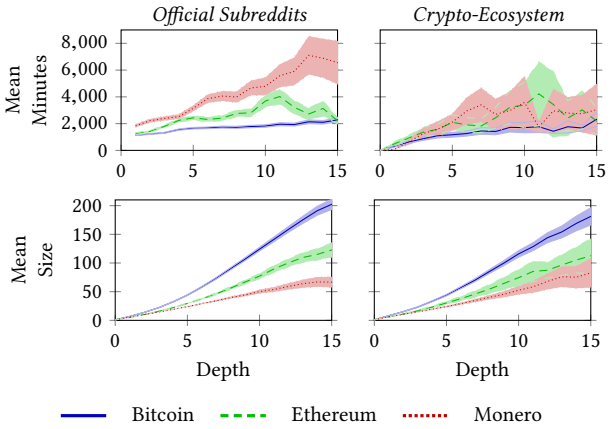


Figure 6: Discussion growth over time measured as the mean average minutes to reach a given depth (above) and the mean size of the discussion when it reaches a given depth (below). 95% confidence intervals are represented with shaded bands.

ineffective posts. We remove these posts from subsequent analyses in all but one case: when we consider the size of discussions. We find an interesting trend across both datasets where Bitcoin and Ethereum have similar rates of ineffective posts (31% and 33%, respectively, for the Official Subreddits and 8% and 9% for the Crypto-Ecosystem) and Monero has much lower rates (13% and 2%). The relative presence of ineffective posts related to the more popular Bitcoin and Ethereum coins are twice to five times as high as found in Monero discussions.

5.1 Speed of Discussion Spread

Here we examine the speed of discussion growth across the three coins of interest and two datasets. We aim to answer three main research questions: *how quickly do discussions begin, grow over time and end*. By answering these three questions we provide a well-rounded overview of both dynamic and static properties of the speed of discussion growth. As we noted previously, we removed what we call *ineffective posts*, posts that failed to trigger a discussion, from the following analysis.

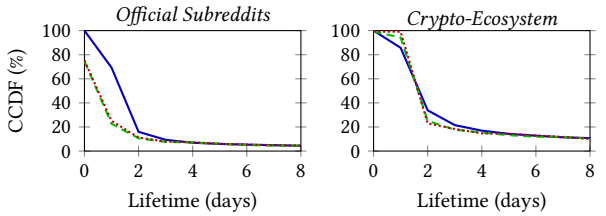
5.1.1 Initial Delays. First, we examine how quickly discussions begin in Figure 5. As a result of the size of the r/bitcoin community and the number of new posts that are submitted to the subreddit, the outliers (e.g. extremely long reaction times from posts that either slowly made their way from the "new queue" to the main subreddit page or were possibly found through the search functionality long after the initial submission) may highly influence mean average delays. On the other hand, the median delay is less influenced by outliers. Bitcoin discussions in the official subreddit have the shortest initial delay of approximately 11 minutes. That is, the median delay between a post submitted to r/bitcoin and that post's first comment is around 11 minutes. In comparison it takes at least 20 minutes for a typical post to Monero to attract its first comment and over 27 minutes for Ethereum. We see that the same ordering is found in the Crypto-Ecosystem. Although there are larger initial delays, we still find that Bitcoin discussions begin the fastest, followed by Monero. Ethereum discussions are the slowest to start. But does this trend follow when we consider growth beyond the first comment?

5.1.2 Growth over Time. Now, we expand beyond the initial delay to the average time it takes for discussions to grow, in general. We examine the speed of discussion growth through the average time to reach each depth, the average size of cascade as each depth is reached and the population size within discussions as they grow.

First, we plot the mean minutes elapsed before discussions reached a given depth in Figure 6 (above) with 95% confidence intervals represented with shaded bands. Although the scale decreases greatly when medians are used instead of mean averages, we find that the trends between the three coins remain consistent. Bitcoin discussions grow the fastest followed by Ethereum then Monero. Although we see some overlap in the 95% confidence intervals when we plot the mean minutes to reach each depth for the Crypto-Ecosystem, the ordering remains consistent.

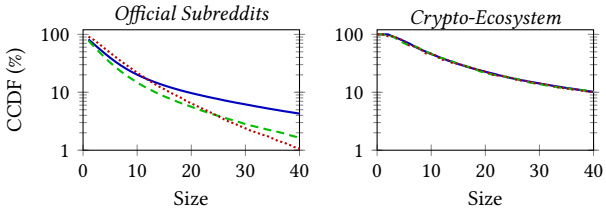
If we consider the size of discussions by the time each depth in the tree is reached, as illustrated in Figure 6 (below), we see that Bitcoin discussions are the largest at each depth and more efficiently increase the number of comments at each depth than Ethereum and Monero across both the datasets. Although the order is flipped (BTC > ETH > XMR), this finding is consistent with the pattern found in average time to reach depth in that Bitcoin shows the fastest growth to largest sizes. This pattern is consistent not only across both datasets but in both median and mean discussion size at each stage of evolution, as determined by the depth in the discussion tree.

5.1.3 Discussion Lifetimes. Finally, we look at how long discussions are active. The median and maximum periods of activity in discussions, *i.e.*, lifetimes, are presented in the Figure 7. We find that Monero discussions have the largest median lifetimes but not the largest maximum lifetimes. Discussions with the largest possible lifetimes are related to the Ethereum cryptocurrency. Interestingly, Bitcoin discussions have the lowest median lifetimes of all three



	Official Subreddits			*Crypto-Ecosystem*		
coin	mean	median	max	mean	median	max
Bitcoin	107.27	6.69	6.43	161.62	13.19	6.00
Ethereum	140.88	11.85	11.12	182.50	17.99	10.00
Monero	182.74	19.18	6.43	178.27	24.11	6.00

Figure 7: CCDFs for the lifetime of discussions (above), the mean, median (in hours), the max lifetimes in months (below).



	Official Subreddits			*Crypto-Ecosystem*		
coin	mean	median	max	mean	median	max
Bitcoin	18.02	6	7,867	20.63	8	2,843
Ethereum	12.74	5	3,816	15.66	7	2,111
Monero	26.98	16	942	17.55	11	692

Figure 8: CCDFs for discussion sizes (above) with the y-axis on a log scale, and the mean, median, and max sizes (below).

coins. In Figure 7, we illustrate the CCDF plots for discussion lifetimes focused on a window of up to eight days which captures 90-95% of all discussions. We see that there is a clear distinction between the complementary cumulative distribution functions for Bitcoin and the other two coins in the Official Subreddits dataset.

Bitcoin has the largest subreddit in terms of content submitted and users who subscribe or contribute, and, as shown in Figure 3 a larger and consistent stream of discussions and comments in both the Official Subreddits and Crypto-Ecosystem datasets. With the consistent flow of new content, users may be more inclined to contribute to a newer discussion than join and continue an older discussion that may no longer be visible because of its score or rank pushing it below a threshold of highly visible ranks. As rank bias influences which content gets voted on and seen by new users, this will also influence which discussions users decide to engage with.

As we noted previously, Monero is the least popular or well-known of the three coins. This is reflected in the smallest subreddit (both in terms of subscribers, contributors, and content) and the smallest sample of posts within the Crypto-Ecosystem. The lower amount and relative frequency, as opposed to the more popular Bitcoin and Ethereum cryptocurrencies, of new content submitted about Monero that would disrupt an existing discussion has probably influenced the larger median lifetimes. With a smaller core community, users who comment in the Monero subreddit may also be more committed to continuing conversations with other contributors, as opposed to users who are tangentially interested in who may visit the larger bitcoin subreddit and make one-off comments on posts they browse. Similarly, the posts related to Monero in the Crypto-Ecosystem dataset may be submitted to smaller subreddits or appeal to a smaller population.

5.2 Scale of Discussion Spread

In this section we present the results on the scale of cryptocurrency discussions, in particular, *how large* discussions grow and the structure of discussions overall and as they develop. In this section, we include the discussions comprised of the *ineffective posts* that we removed from the previous analysis on speed.

Table 2: Discussion audience size measured as median and max volumes of users who participated in each thread.

	Official Subreddits			*Crypto-Ecosystem*		
coin	mean	median	max	mean	median	max
BTC	7.30	3	3,037	11.00	5	685
ETH	5.05	2	707	7.63	4	707
XMR	7.09	5	177	8.63	6	370

5.2.1 Discussion Size. To measure discussion scale we consider the final size of the discussion, *i.e.,* the final volume of nodes within the discussion tree, and plot the CCDFs for discussion sizes for each coin in Figure 8. Across both datasets, the window of sizes up to 40 captures 90 to 99% of all cascades. We see that sizes are more heavily concentrated in lower range of sizes for the Official Subreddits dataset. The CCDF for the r/Bitcoin subreddit rises above the others, indicating more discussions of a larger final size in r/Bitcoin than in r/Ethereum and r/Monero.

When we compare the mean, median, and maximum discussion sizes in the table (below) in Figure 8, we see that the range of r/Bitcoin is much more extended. In contrast, the median size of a Bitcoin cascade is smaller than that of a Monero cascade. Further, Mann Whitney U comparisons of the distributions of each coin's discussion sizes find that, on average, Ethereum discussions are smaller than Bitcoin and Monero discussions ($p < 0.01$), and Bitcoin discussions are smaller than Monero discussions ($p < 0.01$). Although some Bitcoin discussions are much larger than Monero discussions, most are not. Again, this may be due to the size of the community. In the larger subreddits, there are more potential participants that could drive a discussion to a much larger size but there is also a higher influx of content to the larger subreddits where newer content quickly pushes old discussions down and out of view.

5.2.2 Discussion Audience. We can also consider size in terms of the audience or contributors to the discussion rather than the contributions (posts and comments). The overall population sizes are

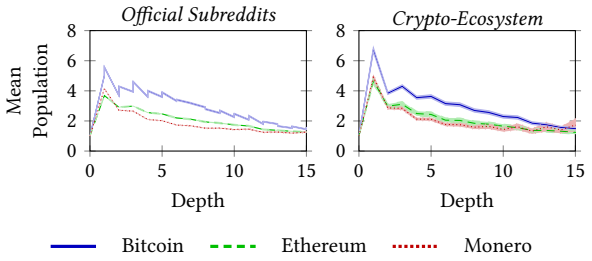


Figure 9: Discussion audience within depth measured as the mean population size at each depth with the 95% confidence interval represented with shaded bands (above), and the median contributor count at each depth (below).

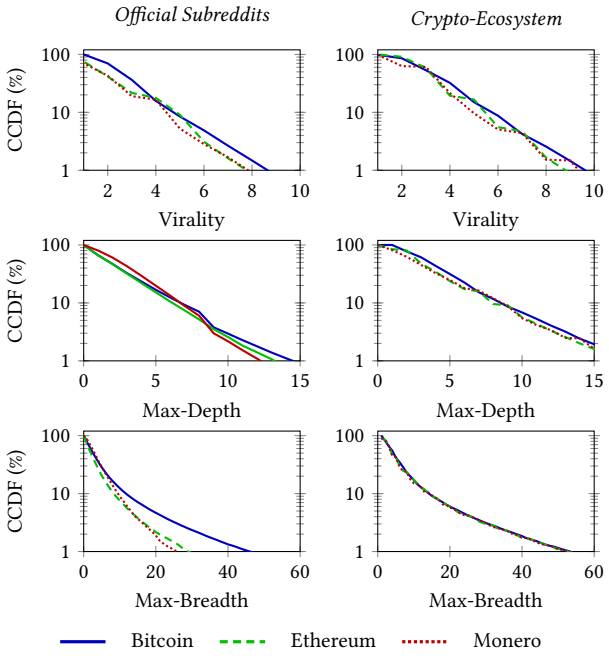
summarized in Table 2. Again, we present the median and maximum overall population (or audience) sizes for discussions related to each coin. The typical discussion is held between 2 and 6 users.

Among the official subreddits, the population sizes are quite similar but in the Crypto-Ecosystem when we consider only those discussions that explicitly reference the coin in the initial post, we see that Bitcoin and Ethereum have a median of five and four users participating, respectively, while Monero has a median of seven users. On the other hand, Monero has the lowest maximum population size while Bitcoin has the largest. The ordering (ETH < BTC < XMR) in terms of typical numbers of participants is also found when we compare the distributions of population sizes with Mann Whitney U tests ($p < 0.01$). As with size of contributions, Monero typically has larger populations in discussions but not the largest possible.

In Figure 9, we observe that the mean population size at each depth differs significantly between coins within the first 15 depths. Across both datasets, Bitcoin discussions have more users participating at each depth. In the Official Subreddits, discussions in r/Ethereum have more users at depths greater than two. That is, discussions in r/Monero on average have slightly more users who respond directly to the initial post than those in r/Ethereum, however this trend is flipped for comments in response to other comments. In the Crypto-Ecosystem dataset, we see an overlap between Ethereum and Monero discussions. When we consider the median population within each depth, we see more similar behavior among the three coins, with spikes above average at several depths for the Bitcoin discussions.

5.2.3 Discussion Structure. Finally, we look at how the scale of the discussion and evolution of the discussion in terms of the structure of the discussion tree. We compare static properties of discussions overall with CCDF plots of virality, max-depth, and max-breadth in Figure 10. As we see in the center row, few discussions have any one given chain grow beyond 5 comments. Less than 1% of discussions have a chain that is at least 15 comments long in the official subreddits, approximately 1-2% has such a chain in the Crypto-Ecosystem dataset.

We highlight the median values for each of the structural properties in Figure 10. We see that discussions tend to have fewer depths with a moderate (at most 3 to 6) number of comments at the most populated depth. Mann Whitney U comparisons of the distributions



	Official Subreddits			*Crypto-Ecosystem*		
	BTC	ETH	XMR	BTC	ETH	XMR
Virality	1.6	1.5	2.3	2.3	2.2	2.8
Max-Depth	1	1	3	3	3	4
Max-Breadth	3	3	6	4	3	4

Figure 10: CCDFs for structural measurements of discussion cascades with a log-scale used for each y-axis (above) and the median values of each property (below).

find that across both datasets, Ethereum has the shortest, narrowest, and least viral of all the coins ($p < 0.01$). That is, these discussions appear more like a broadcast of information than a viral cascade – the initial post is typically followed by a single layer of comments who respond directly to that initial post. Monero discussions have the most "viral" of all discussions, on average, with larger maximum depths and breadths, on average (MWU $p < 0.01$).

When we consider the average comment count (breadth) at each depth however, we see that Bitcoin discussions exhibit larger breadths at each of the initial depths. Figure 11 highlights the mean (above) and median (below) breadths found at each of the initial 15 depths. This illustrates the key differences between coins when we consider discussions as they grow versus the overall discussion after completion. While the maximum breadth of cascades is typically larger in Monero discussions, we see more consistently large breadths in Bitcoin. As we saw in the CCDF plots in Figure 10, Ethereum and Monero discussions exhibit similar patterns of behavior.

5.3 Post Subjectivity and Discussion Spread

In this section, we go beyond the static and dynamic measures of growth and discussion behavior to focus on the *contextual* features of the initial posts that either triggers or fails to trigger a discussion

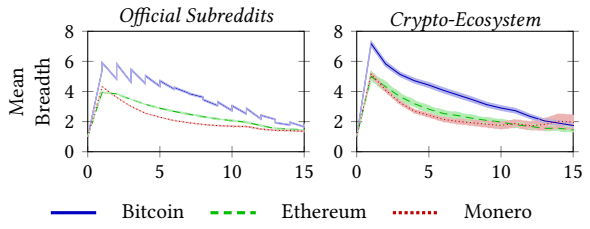


Figure 11: The mean breadth by depth with the 95% confidence interval represented with shaded bands (above), and the median breadth by depth (below).

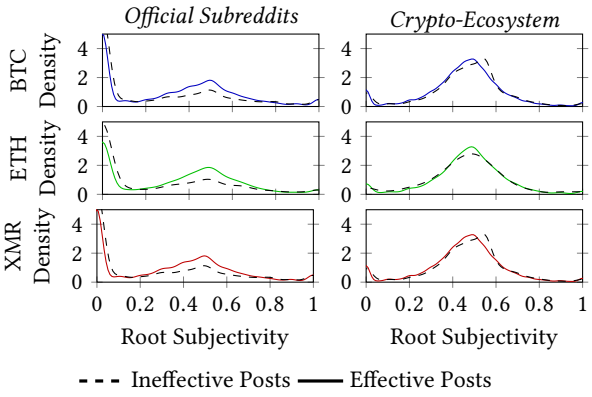


Figure 12: Kernel density estimation plots for *effective posts* that trigger a discussion that includes at least one comment and *ineffective posts* that fail to trigger any subsequent discussion.

to answer the question: *how does the subjectivity of the initial post influence discussion growth?*

First, we compare the subjectivity of effective vs. ineffective discussions at a very high level dichotomy: *did the initial post trigger subsequent discussion or did the discussion consist solely of the initial posts?* We plot the kernel density estimations (KDE) of what we call *effective posts* that initiated discussions of at least one additional comment and *ineffective posts* that failed to initiate a discussion for each of the coins in Figure 12. We see that KDE plots are centered approximately around 0.5, indicating moderate subjectivity, but that ineffective posts tend to skew left, peaking closer to 0.55, for Bitcoin and Monero. Interestingly, ineffective posts about Ethereum appear to remain centered around 0.5 but more evenly across the entire range from 0 to 1, indicating the inclusion of more strongly objective or subjective initial posts.

To gather a more nuanced view of how the initial post in discussions influence the growth, scale, and speed of discussions, we compare normalized mutual information for each of the static discussion properties in Table 3. We also calculated the correlation (Pearson R) of these discussion properties and the subjectivity of the initial post but found significant but low (R<0.05, $p < 0.05$) correlations. Paired with the high normalized mutual information results, we conclude these properties are related but the relationship is not linear. We see that the speed of discussion evolution

Table 3: Normalized Mutual Information between cascade measurements and subjectivity of the root node.

	Official Subreddits			*Crypto-Ecosystem*		
	BTC	ETH	XMR	BTC	ETH	XMR
Virality	0.29	0.36	0.52	0.61	0.68	0.76
Size	0.15	0.21	0.30	0.40	0.49	0.52
# Contributors	0.13	0.18	0.26	0.36	0.43	0.46
Max-Depth	0.10	0.15	0.22	0.29	0.37	0.39
Max-Breadth	0.12	0.17	0.24	0.32	0.40	0.42
Lifetime	0.56	0.68	0.76	0.88	0.92	0.91
Initial Delay	0.37	0.57	0.66	0.75	0.89	0.88

(lifetime and the initial delays) has the greatest mutual information with initial post subjectivity values.

Next, we examined the bivariate kernel density estimations for each of the static discussion cascade measurements and the subjectivity of the initial post (*i.e.,* the "root" of the discussion). We find that the plots within the *Crypto-Ecosystem* dataset illustrate very similar results to the overall distributions of subjectivities shown in Figure 12 at right. Variations in the plots appear to be largely indicative or representative of the differences in the static measurements, highlighted in the previous analyses.

6 SUMMARY

We presented measurement-driven analysis of cryptocurrency discussion spread for three coins of interest on Reddit, and contracted discussion tree patterns in the official coin subreddit and a broader Crypto-Ecosystem on Reddit. Our analysis and novel findings will not only bring the awareness to online discussion spread relevant to cryptocurrencies but will also inform models for forecasting cryptocurrency price that rely on conversations in social media.

We highlight our key findings on *the speed of discussion spread*:

- *Initial Delay* Bitcoin discussions in the official subreddit have the shortest initial delay of approximately 11 minutes. It takes at least 20 minutes for a post to Monero to attract its first comment and over 27 minutes for Ethereum. In the Crypto-Ecosystem delays are larger but coins follow the same trends.
- *Growth Over time* Bitcoin discussions grow the fastest followed by Ethereum then Monero. Bitcoin discussions are the largest at each depth and more efficiently increase the number of comments at each depth than Ethereum and Monero in the official subreddits and Crypto-Ecosystem. Bitcoin shows the fastest growth to largest sizes.
- *Lifetime* Monero discussions have the largest median lifetimes but not the largest maximum lifetimes. Discussions with the largest possible lifetimes are related to the Ethereum cryptocurrency. Interestingly, Bitcoin discussions have the lowest median lifetimes of all three coins.

We highlight our key findings on *the scale of discussion spread*:

- *Volume* On average, Ethereum discussions are smaller than Bitcoin and Monero discussions and Bitcoin discussions are smaller than Monero discussions.
- *Audience Size* The typical discussion is held between 2 and 6 users. In the Crypto-Ecosystem, Bitcoin and Ethereum have

a median of five and four users participating, respectively, while Monero has a median of seven users. ETH < BTC < XMR in terms of typical numbers of participants.

- *Structural properties* Ethereum has the shortest, narrowest, and least viral of all the coins. Monero discussions have the most "viral" of all discussions, on average, with larger maximum depths and breadths. Bitcoin discussions exhibit larger breadths at each of the initial depths.

Future work will extend our preliminary exploration results and further investigate the effect of the original post content on how cryptocurrency discussions spread. More specifically, we will look into content polarity e.g., positive, negative and neutral, novelty, uncertainty and readability. In addition, we will measure user reactions e.g., answer, elaboration, acknowledgement etc. [15] in the discussion cascades influence how discussions spread. Finally, measuring cryptocurrency-related discussion spread before and after the external events of interest e.g., coin price rise, market crash; across a variety of coin types (e.g., popular, stable, scam, etc.); and contrasting signals across multiple social media platforms e.g., Twitter vs. Reddit is of interest.

ACKNOWLEDGMENTS

The research described in this paper was performed at Pacific Northwest National Laboratory, a multiprogram national laboratory operated by Battelle for the U.S. Department of Energy. This work was supported by Defense Advanced Research Projects Agency (DARPA) SocialSim program, under agreement 71177. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government. The Reddit datasets were collected by Leidos, the official data provider for the DARPA SocialSim program.

REFERENCES

[1] Lada A Adamic, Thomas M Lento, Eytan Adar, and Pauline C Ng. 2016. Information evolution in social networks. In *Proceedings of the ninth ACM international conference on web search and data mining*. ACM, 473–482.

[2] Lars Backstrom, Jon Kleinberg, Lillian Lee, and Cristian Danescu-Niculescu-Mizil. [n. d.]. Characterizing and curating conversation threads. In *Proceedings of the sixth ACM international conference on Web search and data mining-WSDMâĂŹ13*. 13.

[3] Eytan Bakshy, Jake M Hofman, Winter A Mason, and Duncan J Watts. 2011. Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 65–74.

[4] Sushil Bikhchandani, David Hirshleifer, and Ivo Welch. 1992. A theory of fads, fashion, custom, and cultural change as informational cascades. *Journal of political Economy* 100, 5 (1992), 992–1026.

[5] Kathleen M Carley. 1999. On the evolution of social and organizational networks. *Research in the Sociology of Organizations* 16, 0 (1999).

[6] Meeyoung Cha, Alan Mislove, Ben Adams, and Krishna P Gummadi. 2008. Characterizing social cascades in flickr. In *Proceedings of the first workshop on Online social networks*. ACM, 13–18.

[7] Meeyoung Cha, Alan Mislove, and Krishna P Gummadi. 2009. A measurement-driven analysis of information propagation in the flickr social network. In *Proceedings of the 18th international conference on World wide web*. ACM, 721–730.

[8] Justin Cheng, Lada Adamic, P Alex Dow, Jon Michael Kleinberg, and Jure Leskovec. 2014. Can cascades be predicted?. In *Proceedings of the 23rd international conference on World wide web*. ACM, 925–936.

[9] Justin Cheng, Lada A Adamic, Jon M Kleinberg, and Jure Leskovec. 2016. Do cascades recur?. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 671–681.

[10] Justin Cheng, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2014. How community feedback shapes user behavior. *arXiv preprint arXiv:1405.1429* (2014).

[11] Justin Cheng, Jon Kleinberg, Jure Leskovec, David Liben-Nowell, Bogdan State, Karthik Subbian, and Lada Adamic. 2018. Do Diffusion Protocols Govern Cascade Growth? *arXiv preprint arXiv:1805.07368* (2018).

[12] Daejin Choi, Jinyoung Han, Taejoong Chung, Yong-Yeol Ahn, Byung-Gon Chun, and Ted Taekyoung Kwon. 2015. Characterizing conversation patterns in Reddit: From the perspectives of content properties and user participation behaviors. In *Proceedings of the 2015 ACM on Conference on Online Social Networks*. ACM, 233–243.

[13] Munmun De Choudhury, Yu-Ru Lin, Hari Sundaram, K Selcuk Candan, Lexing Xie, Aisling Kelliher, et al. 2010. How does the data sampling strategy impact the discovery of information diffusion in social media? *Icwsm* 10 (2010), 34–41.

[14] Maria Glenski, Corey Pennycuff, and Tim Weninger. 2017. Consumers and curators: Browsing and voting patterns on Reddit. *IEEE Transactions on Computational Social Systems* 4, 4 (2017), 196–206.

[15] Maria Glenski, Tim Weninger, and Svitlana Volkova. 2018. Identifying and Understanding User Reactions to Deceptive and Trusted Social News Sources. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 176–181. http://aclweb.org/anthology/P18-2029

[16] Sharad Goel, Ashton Anderson, Jake Hofman, and Duncan J Watts. 2015. The structural virality of online diffusion. *Management Science* 62, 1 (2015), 180–196.

[17] Sharad Goel, Duncan J Watts, and Daniel G Goldstein. 2012. The structure of online diffusion networks. In *Proceedings of the 13th ACM conference on electronic commerce*. ACM, 623–638.

[18] Jacob Goldenberg, Barak Libai, and Eitan Muller. 2001. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters* 12, 3 (2001), 211–223.

[19] Jinyoung Han, Daejin Choi, Jungseock Joo, and Chen-Nee Chuah. 2017. Predicting Popular and Viral Image Cascades in Pinterest. In *ICWSM*. 82–91.

[20] Wenjian Hu, Krishna Kumar Singh, Fanyi Xiao, Jinyoung Han, Chen-Nee Chuah, and Yong Jae Lee. 2018. Who Will Share My Image?: Predicting the Content Diffusion Path in Online Social Networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 252–260.

[21] Muhammad Raza Khan. 2017. Cascading Behavior in Yelp Reviews. *arXiv preprint arXiv:1712.00903* (2017).

[22] Young Bin Kim, Jun Gi Kim, Wook Kim, Jae Ho Im, Tae Hyeong Kim, Shin Jin Kang, and Chang Hun Kim. 2016. Predicting fluctuations in cryptocurrency transactions based on user comments and replies. *PloS one* 11, 8 (2016), e0161197.

[23] Siddharth Krishnan, Patrick Butler, Ravi Tandon, Jure Leskovec, and Naren Ramakrishnan. 2016. Seeing the forest for the trees: new approaches to forecasting cascades. In *Proceedings of the 8th ACM Conference on Web Science*. ACM, 249–258.

[24] Srijan Kumar, Justin Cheng, Jure Leskovec, and VS Subrahmanian. 2017. An army of me: Sockpuppets in online discussion communities. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 857–866.

[25] Andrey Kupavskii, Liudmila Ostroumova, Alexey Umnov, Svyatoslav Usachev, Pavel Serdyukov, Gleb Gusev, and Andrey Kustarev. 2012. Prediction of retweet cascade size over time. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 2335–2338.

[26] Kristina Lerman and Rumi Ghosh. 2010. Information contagion: An empirical study of the spread of news on Digg and Twitter social networks. *Icwsm* 10 (2010), 90–97.

[27] Jure Leskovec, Mary McGlohon, Christos Faloutsos, Natalie Glance, and Matthew Hurst. 2009. Information propagation and network evolution on the web. *DA Project, Machine Learning Dept. Carnegie Mellon University* (2009).

[28] Mei Li, Xiang Wang, Kai Gao, and Shanshan Zhang. 2017. A survey on information diffusion in online social networks: Models and methods. *Information* 8, 4 (2017), 118.

[29] Henry B Mann and Donald R Whitney. 1947. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics* (1947), 50–60.

[30] Ross C Phillips and Denise Gorse. 2017. Predicting cryptocurrency price bubbles using social media data and epidemic modelling. In *Computational Intelligence (SSCI), 2017 IEEE Symposium Series on*. IEEE, 1–7.

[31] Eldar Sadikov and Maria Montserrat Medina Martinez. 2009. Information propagation on Twitter. *CS322 project report* (2009).

[32] Eldar Sadikov, Montserrat Medina, Jure Leskovec, and Hector Garcia-Molina. 2011. Correcting for missing data in information cascades. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 55–64.

[33] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science* 359, 6380 (2018), 1146–1151.

[34] Senzhang Wang, Xia Hu, Philip S Yu, and Zhoujun Li. 2014. MMRate: inferring multi-aspect diffusion networks with multi-pattern cascades. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1246–1255.

[35] Sha Wang and Jean-Philippe Vergne. 2017. Buzz factor or innovation potential: What explains cryptocurrencies' returns? *PloS one* 12, 1 (2017), e0169556.

[36] Honglin Yu, Lexing Xie, Scott Sanner, et al. 2015. The Lifecyle of a Youtube Video: Phases, Content and Popularity.. In *ICWSM*. 533–542.

Goal-setting And Achievement In Activity Tracking Apps: A Case Study Of MyFitnessPal

Mitchell L. Gordon
Stanford University
mgord@cs.stanford.edu

Tim Althoff
University of Washington
althoff@cs.washington.edu

Jure Leskovec
Stanford University
jure@cs.stanford.edu

ABSTRACT

Activity tracking apps often make use of goals as one of their core motivational tools. There are two critical components to this tool: *setting* a goal, and subsequently *achieving* that goal. Despite its crucial role in how a number of prominent self-tracking apps function, there has been relatively little investigation of the goal-setting and achievement aspects of self-tracking apps.

Here we explore this issue, investigating a particular goal setting and achievement process that is extensive, recorded, and crucial for both the app and its users' success: weight loss goals in MyFitnessPal. We present a large-scale study of 1.4 million users and weight loss goals, allowing for an unprecedented detailed view of how people set and achieve their goals. We find that, even for difficult long-term goals, behavior within the first 7 days predicts those who ultimately achieve their goals, that is, those who lose at least as much weight as they set out to, and those who do not. For instance, high amounts of early weight loss, which some researchers have classified as unsustainable, leads to higher goal achievement rates. We also show that early food intake, self-monitoring motivation, and attitude towards the goal are important factors. We then show that we can use our findings to predict goal achievement with an accuracy of 79% ROC AUC just 7 days after a goal is set. Finally, we discuss how our findings could inform steps to improve goal achievement in self-tracking apps.

ACM Reference Format:
Mitchell L. Gordon, Tim Althoff, and Jure Leskovec. 2019. Goal-setting And Achievement In Activity Tracking Apps: A Case Study Of MyFitnessPal. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313432

1 INTRODUCTION

The purpose of an activity tracking app is to help users better understand their behavior. Health-focused apps and fitness devices are the most prevalent, often tracking activities like exercise, eating, and heart rate. Beyond physical health, other types of behaviors are becoming increasingly popular to track as well; for instance, iPhones now come pre-installed with an app that tracks screen time spent in other apps.

In many cases, users wish not only to observe their behavior, but also to improve it [20, 43]. Activity tracking apps often aim to help their users do this, and the guidance they provide can take many forms, ranging from general advice and tips from experts, peer pressure from social networking features, reminders or notifications that ask the user to take a specific timely action, and enforcement of explicitly articulated goals [44]. We think of all these mechanisms as methods to promote behavioral change [4]. They play an important role in people's well-being, akin to the many methods for behavior change in the offline world that one is familiar with, which might include requests or demands from loved ones, medical professionals, bosses, or in the most extreme cases, governmental intervention via fines and criminal punishment.

A nearly universal feature in activity tracking apps, goals are one of the most popular and effective methods for behavioral change [31, 45]. They take many forms. For instance, Apple Watch has three "rings" for the user to fill in, each representing a different daily behavioral goal: one for standing 12 times, one for total calories burned (set by the user), and one for 30 minutes of exercise. Other apps have focused on moderating digital behaviors by setting goals for the amount of time spent on social networking sites [2]. Perhaps the most well-known goal is walking 10,000 steps per day assigned by many fitness-focused activity trackers [7].

Goals in activity tracking apps involve both the setting of a goal (the selection of a desired outcome) and also the achievement of that goal (the process of working towards the selected outcome). Setting a good goal is critically important for users' success: too hard, and the user will become frustrated and give up. Too easy, and they won't achieve as much health benefit as they could have [46]. There has been interesting recent work on goals in activity tracking apps [5, 13, 25, 29, 36, 40, 48], but it remains a topic where there is still much to be understood – particularly on the issue of setting a good goal, since it can be difficult to find records of the process by which goals are both set and subsequently achieved, along with the behaviors that led to those goals being achieved. This has led to the problem that, despite the importance of goal-setting and achievement in activity tracking apps, users often receive limited guidance on selecting a goal that they will actually meet. While most activity trackers use general guidelines from relevant domain expertise combined with demographic information [10] to help users make a choice, after the initial goal is set they often provide limited on-going advice as to whether that goal choice is turning out to have been a good one. Some work has gone further and used current trajectory towards a goal to intervene. For instance, they can notify a user if they are not on track to meet their goal by a certain time frame if they keep that trajectory, or are consistently not meeting daily goals [1, 12, 22, 34, 38]. However, these methods do not consider a holistic view of the many important behaviors that lead to a goal being met, and make simplistic assumptions about linear progressions.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313432

Present Work: Weight Loss Goals in MyFitnessPal. In this paper we consider the goal-setting aspects of activity tracking apps by focusing on a setting where detailed information on both how goals are selected and many related behaviors while attempting to achieve those goals can be studied. Our setting is the process of selecting and working towards weight loss goals in MyFitnessPal. This process has a clearly defined structure: first, upon creating a MyFitnessPal account, all users select weight loss goals, and do so without any guidance from the app. Users optionally provide a free-text reason for their selection of that goal. Users then may then use the app to log their food intake and weight to help them track their progress towards that goal. Meeting their weight loss goal is important for their health: achieving a self-determined weight goal has been linked to long-term weight loss, while the failure to reach a self-determined weight may discourage someone's belief in their ability to control their weight, resulting in abandonment of weight loss behaviors and weight re-gain [14, 27].

MyFitnessPal weight loss goals thus have the key ingredients we need: they take place in a goal-setting and achievement process carried out by users of a activity tracking app who are working towards meeting that goal, users use the app to help guide their lower-level behaviors that will help them to meet that goal, and given the nature of activity tracking, all of this is logged, making the analysis possible. It also serves as an instance of a broader type of goal-setting, familiar from the off-line world as well as the on-line world, in which people aim to lose weight by setting a weight loss goal.

Both the process of goal-setting and the process of losing weight has been studied extensively in medical literature [14]. By contrast, because of our interest in the issue of goal-setting and achievement in activity tracking apps, we study the process from the perspective of early detection of goals that will never be met: we ask which goals users set, how users behave shortly after they set them, and how those behaviors lead to those goals being achieved. Because we are interested in exploring how activity tracking apps may be able to help users select better goals, we focus specifically on behavior that takes place during the first 7 days after a goal is set, and examine whether we might be able to detect early-on that a long-term weight loss goal will never be met. Such early-detection would enable an intervention warning a user that they are working towards a goal they are unlikely to meet, which means we may be able to encourage them to re-think a more realistic goal and prevent them from spending too much time working towards and becoming invested in that goal, only to ultimately become discouraged and re-gain weight because they did not meet it. This could potentially be considered a "just-in-time intervention" [33].

Present work: Main Results. We analyze 2.8 million weight loss goals with 44.6 million weights logged over a period of three years. We begin by validating our dataset through a comparison of the goals the people set in MyFitnessPal to prior studies from before the age of activity trackers, finding that women set more ambitious goals than men and younger adults set more ambitious goals than older adults (Section 4). We also use the unprecedented size of our dataset to show that some findings in prior work are likely due to selection effects. We then analyze goal completion rates and duration, finding that men and older adults are more likely to meet their goals (even when accounting for goal difficulty), but that overall completion rates are low (Section 5).

We then turn to the problem of early detection of goals that will never be met (Section 6). We investigate user's behavior over the first seven days after they set their goal. We find that, contrary to what some prior clinical studies might suggest, there's no such thing as "too much" early weight loss when it comes to meeting your goal (Section 6.1), and that people who log their weight more frequently within the first week are more likely to meet their goals (Section 6.2). We show that more calories reported per day during the first 7 days leads to a lower likelihood of meeting goals, but only for users who are committed loggers; surprisingly, for many users, logging more calories actually indicates a higher propensity to meet a weight loss goal (Section 6.3). We then turn to the motivations behind the goals that people set, introducing a novel application of topic modeling algorithms to identify four primary motivators and show how achievement rates vary with these motivators (Section 6.4).

Finally, we show that whether a user will achieve their target weight can be predicted just based on the initial behavior during the first few days after the target is set. We build a machine learning model to predict target achievement with promising accuracy. We conclude with a discussion of how these results can be translated into actionable suggestions for activity tracking apps as well as traditional offline weight loss programs.

2 RELATED WORK

Goal-setting Theory and Weight Loss. There is a vast body of work discussing the mechanics and psychological aspects of goals, as well as how goals can be used for health-related behaviors. Locke and Latham's seminal work summarizes empirical research on goal-setting theory [27]. Other researchers have focused on how those findings apply specifically to health goals [19, 46].

As with goal-setting theory, there is also a vast body of work discussing the mechanics of weight loss and weight loss goals. As weight loss goals tend to focus on longer term weight loss (as discussed later, we find most people want to lose significant percent of their body weight), literature relating to longer-term weight loss and maintenance, such as Elfhag and Rossner conceptual review of the subject, is most relevant to our work [14]. Other work has focused on characterizing weight loss goals, both in terms of what goals people set and which goals are associated with the best outcomes [16, 24, 26, 27, 49].

Our work first verifies that findings from these smaller-scale offline studies generally apply to our activity tracking dataset in terms of which goals people choose to set and their likelihood of meeting them, while also using the unprecedented scale of our dataset to present new insights. We then extend existing work by specifically investigating early behaviors that predict whether or not a user will meet their weight loss goal in an activity tracking app.

Implementing Goals in Activity Trackers. Activity trackers provide users with an environment to both set goals and track their progress towards their goals. In the best case, they might also be described as systems to help people implement their goals [18]. Some work has looked at different higher-level strategies for

implementing goals in self-trackers, typically through qualitative small-scale studies [8, 11, 32]. Other work has investigated the role that goals play in changing behavior in self-trackers [9, 15, 15, 35]. Rather than exploring the best way to implement goals in activity trackers, our work is a case study of one of the world's most popular implementations: MyFitnessPal.

Adaptive Goals. While goals in self-trackers frequently take into account one's current size and weight, personalized or adaptive goals that take behavior in to account are used sparsely in practice [37]. Some researchers have sought to assign adaptive fitness goals based on a particular user's historical data, in order to improve a goal that is too difficult or too easy. These typically employ simple algorithms that raise goals that the user is easily achieving, and lower goals that the user is struggling to achieve [1, 12, 22, 34, 38]. MyBehavior, in a study of 17 users, suggested exercise or actions to take, but not goals [42]. Hermanny *et al.* explored using heart rate variability to set goals [21]. These adaptive goals can be considered part of a new class of "just-in-time interventions" [33].

Critically, existing work has focused on measuring progress directly related to the goal (e.g. steps for step count goals, calories for calorie goals). Our work builds on a large dataset covering many behaviors related to achieving the goal and findings from work in goal-setting theory to understand whether a goal will be met by taking a holistic view of a user's behavior. We are aware of one other paper which aimed to predict whether a weight loss goal will be met; however their model relied upon users having already been working towards their goal for at least two months and to have logged a large number of weights, meaning it was only applicable to less than 1% of the total number of users in their dataset [47].

3 DATASET DESCRIPTION

This section describes the mechanics of the activity tracking app MyFitnessPal, the dataset used in this paper.

The Mechanics of MyFitnessPal. MyFitnessPal enables its users to track many health-related behaviors. Users can log their food intake, their exercise, and their weight. Users can also set health and behavioral goals. There are three main goals in the app: total weight loss, weekly weight loss, and calories per day. In this paper, we focus on the total weight loss goal: the weight that the user would like to achieve. All users enter a total weight loss goal upon creating a MyFitnessPal account. The app does not provide any guidance when a user is selecting their total weight loss goal. Users also decide upon a weekly weight loss goal between 0 and 2 lbs per week, in increments of half a pound. The default value for this field is 1 lb per week. Using this weekly weight loss goal and demographic information entered by the user (such as current weight and height), the app then automatically assigns the user a calories per day limit. Each food that a user logs is counted against this limit, while exercise is counted as burning off some of those calories.

To log a food item, users type the name of what they ate into the app's search bar, which will then return a list of matches from which the user can choose. Many of these items are branded and will contain full nutritional content supplied by their manufacturer (e.g. a McDonald's Big Mac or Trader Joes Granola). Other generic items, such as eggs or chicken breasts, will also contain nutritional content. If the user does not find a suitable match in the app's database, they can optionally log their item manually and choose which, if any, nutrient content to provide. There are two methods for logging a weight. First, a user can manually enter their current weight into the app whenever they would like. Second, users can connect MyFitnessPal an many internet-connected "smart" scale, which automatically transfer weights they record to the app.

Progress towards the calories per day limit is prominently displayed on the main page of the app. Users can easily view their weight loss progress by clicking on a large icon in the bar at the bottom of the screen.

Table 1: Descriptive statistics from our dataset of MyFitnessPal users.

Characteristics	Value
# of users studied	1,413,431
Observation period	August 2014 - April 2017
Weights logged	44,568,749
Weight goals set	2,803,073
Food items logged	8,785,560,832
Free-text goal justifications	56,014
Mean age	38.69
% users female	71.9
% users underweight	0.34
% users normal weight	29.39
% users overweight	36.28
% users obese	33.99

The Dataset. We use a dataset of 1.4 million MyFitnessPal users over three years, from August 2014 through April 2017. Users set 2.8 million goals, log 44.6 million weights, and eat 8.8 billion food items. Table 1 provides descriptive statistics for our dataset.

In this paper, we focus on weight loss goals and ask the question of which users will ultimately achieve their goal. A user's goal is considered achieved if, at some point in the future after setting the goal, the user logs a weight that is at least as low as that goal. As we are interested in users who showed at least some amount of interest in using the app to track their weight loss progress, we filter to users with at least 7 days between the first and last logged weight of their goal. To remove any minors, extreme outliers, or users who entered likely fake information, we also filter out users who entered a weight over 1000 pounds, an age less than 18 or over 80, and who's weight change goal was more than a 100% difference from of their current weight. This removed 3,681 users users.

4 WHAT WEIGHT GOALS DO PEOPLE CHOOSE?

Performance towards a weight loss goal is, of course, significantly dependant upon the goal's difficulty. This section focuses on goal selection. We aim to validate our dataset by comparing the weight loss goals that users set in MyFitnessPal to the weight loss goals that people set before the age of self-trackers. In particular, we study which goals users choose for themselves; stratifying by demographic information. As mentioned earlier, MyFitnessPal provides no guidance when users are selecting weight targets, which enables

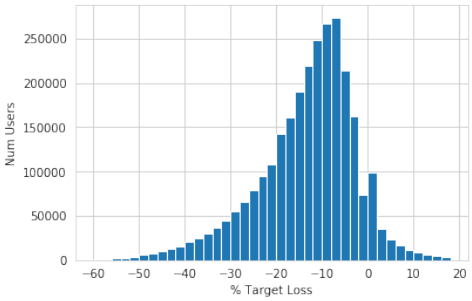


Figure 1: Histogram of weight loss goals as a percent of initial weight. We see a somewhat skewed distribution (skewdness of -0.46), with a peak around 10% weight loss.

an unbiased analysis. While we do note that there is likely a selection bias in terms of who chooses to use MyFitnessPal, towards people who are interested in loosing weight, this bias towards users interested in self-improvement is likely present in many activity tracking apps.

4.1 Aiming Low: Initial Target Selection

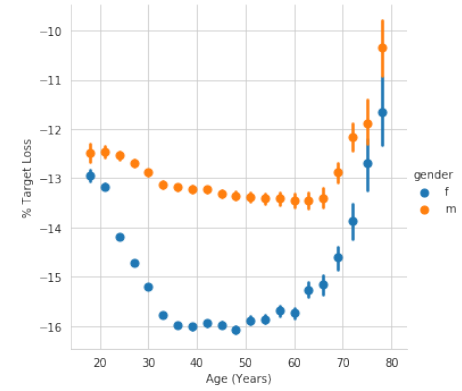
We begin by plotting a histogram of weight loss goals, in terms of percent of current weight, in Figure 1. We find a somewhat skewed (skewedness = -0.46) distribution with a peak around 10% weight loss. We also see that a non-trivial portion of users are actually aiming to gain weight. We remove these users for our later analysis, as this is outside the scope of this paper.

4.2 Declining Ambition: User Age

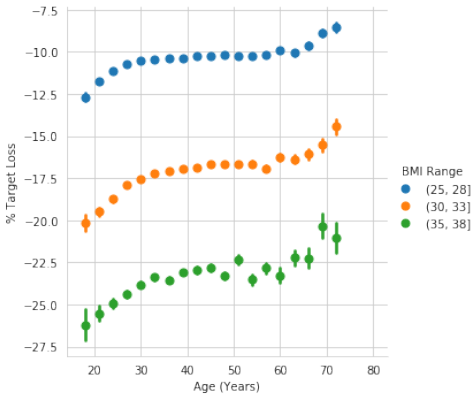
In this section, we ask how ambition changes with age. To answer this question, we begin by plotting the amount of weight users aim to lose (as a percent of current weight) by age. We find a U-shaped curve, shown in Figure 2a. For instance, for women between the ages of 18 to 35, users aim to lose an increasingly large percent of their weight, moving from 13.2% to 15.7% loss (Mann-Whitney U $p < 0.001$). The number then stays relatively constant around 15.7% up until the age of 60. However, after age 60, users aim to lose an increasingly smaller amount of weight, with users at age 75 returning approximately to the same goal as those at age 18 (Mann-Whitney U $p < 0.001$).

To validate our results, we can compare them to a 1989 large-scale telephone-based survey of weight loss goals [49]. To our knowledge, this is the most recent such survey, though notably, it occurred long before the existence of activity tracking apps. The survey reports results from 21,109 American households, asking if they were currently aiming to lose weight. If the survey participant said yes, then the study included that goal in their results. This means that both this survey and our behavioral tracking dataset focus on users aiming to lose weight, thus removing the potential for selection bias on that co-variate. The survey reports results separately for men and women.

Similarly to our results, the survey found a U-shaped curve, but with larger magnitudes. For women between ages 18-29, the survey reports a mean of 17.3 percent loss, compared to our 14.3. Between



(a)



(b)

Figure 2: Weight loss goal as percent of current weight, by age. (A): Without stratification by BMI. We find a U-shaped curve in agreement with existing literature. The U is significantly deeper for women than for men. (B): With this stratification (results shown only for women), we now see that younger adults set the most ambitious goals, but get less ambitious up until the age of 30, from which point goals stay relatively consistent. Then, around 60, older adults set increasingly less ambitious goals. Error bars in all plots correspond to bootstrapped 95% confidence intervals (here, they are mostly too small to be visible).

ages 40-49, the survey reports a mean of 18.1, compared to our 16.0. And between ages 70-79, the survey reports a mean of 14.9, compared to our 13.4. The similar shape of their curve to ours helped to validate our results, while at the same time showing that at all ages, users in MyFitnessPal are slightly less ambitious than those in the 1989 survey.

Having validated our dataset, we now ask, how can we explain this U-shaped curve present in both our work and existing work? The U-shape seems to go against what we might expect, given the fact that younger people tend to be less conservative and take more risks than older people [16]. We note that neither we nor the work we validate our results against have thus far accounted for the fact that age is strongly correlated with whether someone is at a healthy

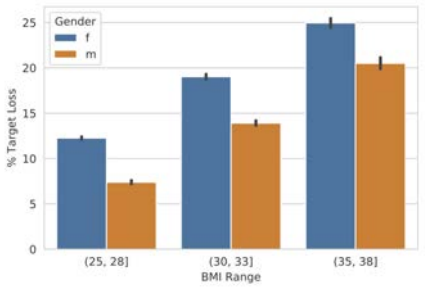


Figure 3: Women set more ambitious goals than men; when controlling for BMI, women aim to lose approximately 5% more weight.

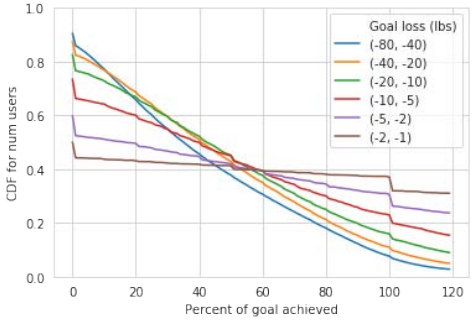


Figure 4: CDF of goal achievement, stratified by goal difficulty. X-axis indicates percent of goal achieved while y-axis indicates the portion of users who have achieved at least that amount of their goal. We find that across the board, goals are unlikely to be achieved. Only 30% of the easiest goals are ever reached, while less than 5% of the most difficult goals are ever reached.

weight, with younger adults more likely to be at a healthy weight. We hypothesize that the U-shaped curve may be due to the fact that younger users will be healthier on average than older users, and thus simply need to lose less to reach a healthy weight. To test this, we take advantage of the unprecedented size of our dataset and stratify our earlier plot by BMI (Body Mass Index, a number based on height and weight used to determine if someone is classified as having a normal weight, overweight, or obese). We do so in Figure 2b. We now find that, when accounting for how healthy a user's existing weight is, we see that younger users actually set significantly more ambitious weight loss goals than older users.

4.3 Gender Matters: User Gender

We now turn to gender differences. Given our findings from the previous section, In Figure 3 we again stratify by BMI to control for selection bias, as men tend to have a higher BMI than women. We find that women set more ambitious goals than men and that, regardless of BMI, this effect size is about a difference of 5% (Mann-Whitney U $p =< 0.0001$). This is somewhat in agreement with the 1989 survey mentioned above, but with a larger magnitude (they saw a difference of at most 4.1%).

5 DO PEOPLE REACH THEIR GOALS?

Next, we explore goal achievement rates along with the amount of time taken for goal achievement. We find that overall, 18.2% of weight loss goals are met. This is remarkably close to an earlier finding from 2005 that approximately 20% of people overweight individuals are successful at long-term weight loss [50].

5.1 How many people reach their goals?

Clearly, goal difficulty should impact goal achievement. We hypothesize that more difficult goals are less likely to be achieved. Therefore, we start by breaking down target achievement as a function of goal size. Figure 4 shows a cumulative distribution function (CDF) with a different line for target size buckets. The x-axis indicates percentage of target reached ($x = 100$ indicates that a target has been achieved). We find that target achievement is relatively low, and decreases as targets become more ambitious. People with the easiest goals (between 1 and 2 percent of their initial weight) are most likely to achieve their targets, with nearly 30% of these users reaching at least 100% of their initial weight loss goal. Of users with the most difficult goals (between 40 and 60 percent of their initial weight), only 10% are likely to achieve their target.

The x-axis of the above plot begins at zero, indicating that a user's weight has not changed. We note that the plotted y-value of our CDF does not begin at 1.0 for any of the goals; this is because a large portion of users end up gaining weight, putting their goal achievement progress to the left of our x-axis starting point. Interestingly, despite that fact that users who set easier goals are more likely to achieve them, we also find that users who set easier goals are more likely to go in the opposite direction and gain weight. For instance we see that only 40% of users with the easiest goals lose weight, while 90% of users with the hardest goals end up losing weight. This finding highlights the importance of not setting a goal that is "too easy" for a user.

We also notice a blip at 100 percent achievement, indicating that while many users either change their target or stop using the app once they've reached their goal, others continue to lose weight even after achieving their target.

We also find that men are more likely to meet their weight loss goals, even when accounting for the fact that men set easier goals: for instance, men who aim to lose 5-10% of their weight succeed 28% of the time, while women who aim to lose 5-10% of their weight succeed 20% of the time.

5.2 How long does it take to reach your target?

Having established how many people reach their targets, we now ask how long it takes them to do so. In Figure 5, we show a CCDF of time taken to achieve a goal as function of goal difficulty. We find that more difficult goals take longer to achieve. For the easiest goals, half of all users who meet their goal do so within 100 days. For the most difficult goals, this number is closer to 300 days.

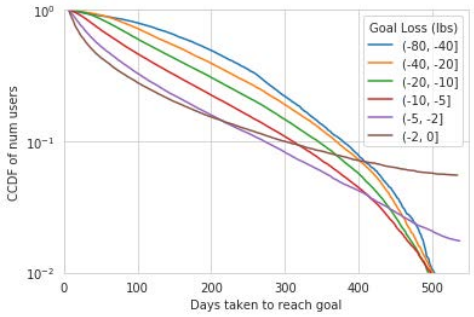


Figure 5: CCDF of days taken to achieve a goal, stratified by goal difficulty. More difficult goals take more time to achieve.

6 EARLY WARNING SIGNS

Having seen that few users ever meet their goals, we now ask what are the early behavioral factors that indicate a user will or won't meet their goal? We ground our analysis in the existing body of literature on short and long term weight loss. We work to both answer open questions in the literature and to determine how existing findings apply specifically to early detection in weight loss target achievement.

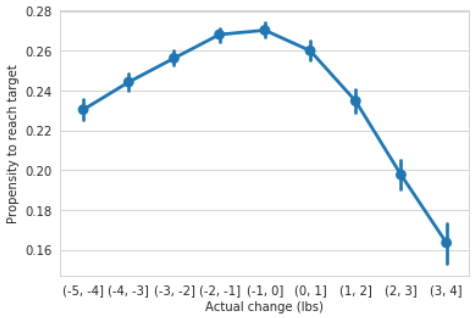
We focus specifically on the first seven days after the first weight of a goal has been logged. We find that even for difficult, long-term goals such as losing twenty pounds, behavioral differences observed during just the first seven days are indicative of large changes – up to double – the propensity to reach that goal.

6.1 Initial Weight Loss Patterns

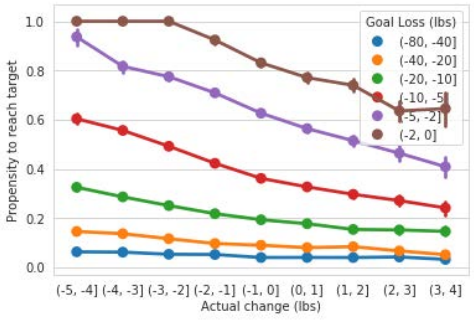
Too much too fast?. Initial weight loss has been identified as a predictor for later weight loss, and also for weight loss maintenance [14]. Some studies have found that the greater the initial weight loss, the better the subsequent outcome [3]. However, other work has found that greater initial weight loss actually predicts future weight gain, explained by the fact that overly quick weight loss is considered unsustainable and clinically unhealthy [30].

Goal-setting theory suggests that early progress towards a goal could serve to provide future motivation, while slow initial progress may cause frustration and ultimately lead to an abandonment of the goal [28]. This taken together with the clinical findings discussed above, make it uncertain as to whether there may be such a thing as "too much" initial progress towards a weight loss goal.

To answer this question, first plot propensity to meet weight loss goal by weight change over the first seven days. Figure 6a shows that we find a U-shaped curve suggested by prior weight loss maintenance literature. On the right, we see that as people gain more weight during the first week, that are less likely to meet their weight loss goals. On the left we see that, as people lose more weight, they are also less likely to meet their goals. We note that the downward trend seems to start around -2 lbs per week, which is exactly what prior literature would suggested: 2 lbs per week of weight loss is often considered the maximum healthy amount [23].



(a)



(b)

Figure 6: Propensity to meet weight loss goal by weight loss over the first week. (A): Without stratification by goal difficulty, we find a U-shaped curve in agreement with existing literature suggesting that "too much" early weight loss is unsustainable and leads to worse longer-term outcomes. (B): With stratification by goal difficulty, we now see that people who lose more weight in the first week are far more likely to reach their goals. Contrary to what medical literature might suggest, there is no such thing as "too much" unsustainable early weight loss which ultimately leads to worse results.

However, we hypothesize that there may be a different reason for this U-shape curve than the idea that "too much" early weight loss is unsustainable and leads to weight gains, as suggested by prior literature. We hypothesize that this U-shaped curve may be due to the fact that people who lose less weight during the first week have also set easier goals on average. To test this, we can stratify our earlier chart by goal difficulty. We show this result in Figure 6b. We now find that there is in fact no amount of initial weight loss which reduces a user's propensity to meet their weight loss goal. Further, see a large difference in propensity to meet goals based on initial weight loss: for instance, a user who loses between 4-5 pounds over the first two weeks of their 20-40 pound loss goal is twice as likely meet that goal as one who loses between 1 and 2 pounds (Mann-Whitney U $p < 0.001$). This finding suggests that follow-up studies to highly cited clinical weight loss papers may be warranted, to determine whether the idea of "too much early weight loss" is due to confounding by motivational reasons rather than physical limitations around unsustainable weight loss

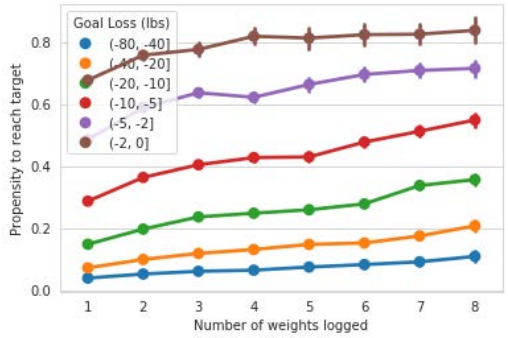


Figure 7: Propensity to achieve goal by number of days logged a weight over the first seven days, stratified by goal difficulty. We see that across all difficulty of goals, people who log weights more frequently over the first week are more likely to achieve their goal.

6.2 Self-monitoring

Self-monitoring means observing oneself and one's behaviour. Indeed, this is the core purpose of an activity tracker. Most activity tracking apps today rely on their users manually choosing log many important behaviors. While step count is typically logged automatically, workouts, food intake, and weights often involve some amount of manual input.

Regular self-monitoring of weight has been linked to long-term weight loss maintenance, which suggests that it may also be important for meeting weight loss goals [6]. In Figure 7, we plot the number of days that a user logged their weight over the first 7 days of a goal. We see that, for all difficulty of weight goals, people who log more regularly over the first seven days are significantly more likely to ultimately meet their goal. For a goal between 40-80 lbs, users who log a weight all 7 days of their first week are more than twice as likely to ultimately meet their goal than those who log only once, moving from 10% to 20% likelihood (Mann-Whitney U $p < 0.001$). Similarly, for a goal between 10 and 20 lbs, we see a change between logging one weight and seven weights from 15% up to 35% (Mann-Whitney U $p < 0.001$).

6.3 Dietary Intake

One of the core features of MyFitnessPal is the ability to log food intake. Obviously, food intake is strongly associated with weight loss. Longer-term weight loss and maintenance has been linked to eating smaller portions and higher quality nutrients [14].

In MyFitnessPal, users log individual food items, where each item has an associated calorie amount (typically populated by MyFitnessPal's food database). We hypothesize that users might fall into two categories: uncommitted loggers, who likely aren't particularly motivated to log their food and don't log everything they eat, and committed loggers, who are motivated to log what they eat. We assign users to groups based on total days logged over the first 7 days: users who logged at least one food item all 7 days are considered a committed logger, and users who did not are considered an uncommitted logger. How likely are these groups to meet their weight loss goals?

Table 2: The four topics discovered from NMF topic model trained on weight loss goal reasons supplied by users when setting their goal. Each topic lists the top ten words associated with it.

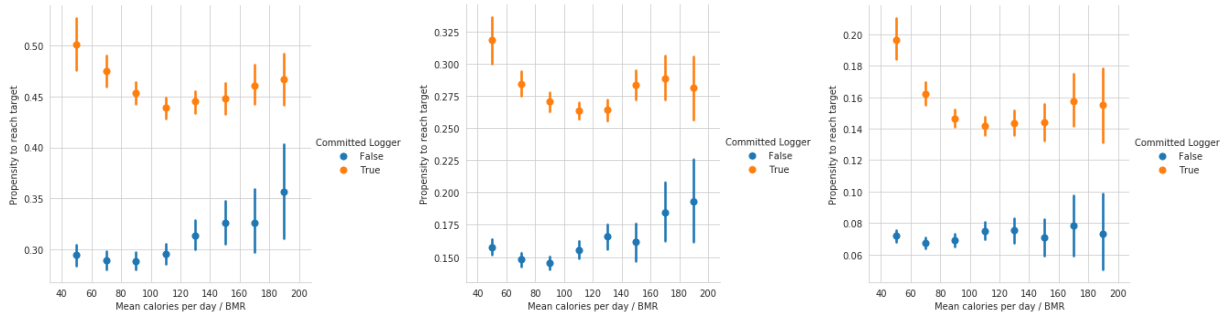
1. Kids role model	2. Lifestyle	3. Health	4. Clothes
good	live	shape	fit
example	life	health	clothes
set	long	weight	old
family	longer	family	able
children	enjoy	need	jeans
kids	happy	tired	wear
daughter	family	time	energy
energy	able	lose	cute
role	kids	kids	size
model	time	energy	comfortable

Figure 8 shows propensity to meet a weight loss goal by mean calories per day over the first 7 days, normalized by each user's Basal Metabolic Rate (BMR, the amount of calories an individual consumes per day at rest), for both committed and uncommitted food loggers. To avoid confounding by goal difficulty, each chart contains only a specific range of goal weight losses. We find that, regardless of amount calories logged, committed food loggers are far more likely to meet their weight loss goals. We also see that, as committed loggers log more calories, they are less likely to meet their goals, which is what we would expect given that eating more calories results in worse weight loss. However, this effect only lasts up until around 110% BMR calories logged, after which more calories logged actually seems to correspond with a higher propensity to meet the goal (though we note that confidence intervals all overlap). This is likely explained by the fact that some of these users are not actually eating more, but rather are extreme loggers who log all food that they eat.

For uncommitted loggers, we see a very slight curve for which more eating indicates a lower propensity to meet a goal, followed by a higher propensity. The difference between the top of the curve and the bottom of the curve is extremely small, which is likely because for these uncommitted loggers, more food loggers mostly is indicative of an increasingly higher commitment to logging, rather than more eating. Overall, we find that logging more calories leads to the expected reduction in propensity to meet a weight loss goal, but only for committed food loggers.

6.4 Motivation

When a user sets their target, MyFitnessPal optionally allowed users to fill in a free text form which asked why they were selecting their goal. We call this a "weight loss reason". Attitude has been found to be important in weight loss maintenance [14], which suggests it may also be important in meeting weight loss goals. We hypothesize that the weight loss reason provided by a user may provide a glimpse into their mindset and attitude towards setting and meeting their goal. In this section, we examine these weight loss reasons to determine whether they are indicative of propensity to meet a weight loss goal.



(a) 5-10 lb goal loss. (b) 10-20 lb goal loss. (c) 20-40 lb goal loss.

Figure 8: Committed food loggers during the first 7 days of a goal are much more likely to meet their weight loss goals than less committed loggers. Further, committed loggers who log more calories are less likely to meet their goal, while uncommitted loggers who log more calories are more likely to meet their goal.

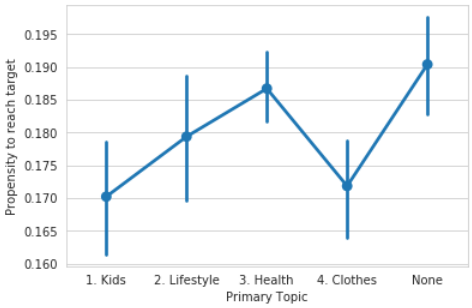


Figure 9: Propensity to achieve goal by motivation. Motivations derived from learning a topic model over free-text goal reasons provided by users and then assigning each goal a primary topic.

After filtering to users who choose to use this field, we end up with 56,014 goals associated with a reason. We first provide a few sample reasons to give a sense of the type of reasons provided by MyFitnessPal users (note that for anonymization, we slightly change the exact text while retaining the meaning). Many are somewhat generic, with users writing that they are setting their goal "To feel better, and have more energy!" and "I would like to get in shape and get back to the weight where I felt comfortable with my body." However, some reasons were quite specific to the user's personal situation, such as "My dad's adopted and doesn't know his family history, but he had a heart attack at age 34 and a RI before age 46. I already have heart issues (which admittedly are well controlled with meds, but still....."

The choice to fill in this field seems likely to be related to propensity to self monitor and commitment to the app which, as discussed earlier, are linked to a higher propensity to achieve one's goal. Surprisingly, we find that users who fill in a weight loss reason are less likely to meet their goals (20.5% vs 17.6%) (Mann-Whitney U $p < 0.001$). However, this difference can likely be explained by the fact that people who supply a weight loss reason aim to lose nearly 50% more weight (33.8 lbs vs 24.6 lbs, Mann-Whitney U $p < 0.001$) and, as we showed earlier, more difficult goals are less likely to be met.

We now ask: what is different about the reasons given by people who meet their goals versus the people who don't? We first find that people who meet their goals use 1.4 fewer words (Mann Whitney U $p < 0.0001$) than those who do meet their goals, at 19.2 words compared to 20.6 words. This is surprising, as one might expect that a longer reason indicates more commitment to the app or more thought-out reasons for losing weight. But how do these reasons differ in terms of the content's subject matter?

Topic Models. Studies have found that people who are motivated to lose weight by a desire for greater self-confidence or superficial considerations such as appearance (e.g. a "healthy narcissism") are more successful at maintaining weight loss than those who are primarily motivated by medical concerns or from pressure by those around them [14]. To determine if we see a similar effect in propensity to meet weight loss goals, we would like to map each weight loss reason to an easily interpretable category. To do this, we propose a novel application of topic modeling algorithms.

Topic models are an unsupervised learning approach to automatically infer interesting patterns in large text corpora [39]. We use this exploratory approach with the goal of understanding what topics are commonly discussed in goal reasons provided by users, with the aim of understanding how they differ in goals which are or are not achieved. Topic modeling has previously been used to extract health-related topics from Twitter [17, 39].

We build our topic model using non-negative matrix factorization (NMF) on extracted term frequency-inverse document frequency (tf-idf) features from our corpus, which helps to select the most useful terms. We use the "English" stopwords list from scikit-learn [41] along with a few words we added manually: want, look, feel, better, healthy, healthier. We use these as stopwords because they were present in far too many articles, to the point that they dominated every topic generated, and in this situation a typical approach to improve topic models is to remove such words. We experiment with selecting a number of topics between k=3 and 20, ultimately selecting k=4 after finding them the most clearly interpretable and differentiated.

Table 2 presents the top 10 terms from each of the four discovered topics, along with our manually assigned topic labels. We assign each goal reason a "primary topic" by selecting the topic with the highest associated score from our NMF model, with a minimum required score of 0.01 (22.4% of goals did not meet this threshold for any of the four topics). We find the topics to be:

(1) **Role model for kids:** users with a reason in this topic discussed how they wished to set a better example for their children. Primary topic for 12.4% of users.
(2) **Lifestyle:** users wanted to live longer, happier lives so that they'd have more time to spend with their family. Primary topic for 12.8% of users.
(3) **Health:** users were concerned about their health and wanted to get in better shape. Primary topic for 36.9% of users.
(4) **Clothes:** reasons in this topic discussed wanting to fit into older clothes that they used to be able to fit in to, or being able purchase and fit into new clothes that they found attractive. Primary topic for 15.4% of users.

Now that we have learned topics for our goal reasons, we can examine which reasons reflect a higher propensity to meet one's goal. To do this. In Figure 9, we plot propensity to meet goals by primary topic. We first note that, unlike in prior sections of this paper, we are unable to stratify by goal difficulty due to the smaller size of this dataset. This means that some results may be explained by the fact that, for instance, people who's primary motivation is fitting in to clothes set slightly easier goals (2 lbs) than those who primary motivation is health. That said, we do find that people primarily motivated by health are more likely to meet their goals than people primarily motivated by clothing, which seemingly goes against what the prior work cited above would suggest.

7 PREDICTING GOAL ACHIEVEMENT

Next, we build on insights from previous sections to predict the likelihood that a user will ever achieve their goal after observing their behavior over the first 7 days.

7.1 Features Used For Learning

To illustrate the predictive power of the different feature sets reported earlier in this paper, we define a series of models, each with a different feature set corresponding to one of the goal-related behaviors. We focus on six types of features:

(1) **Weight loss goal:** In Section 4, we saw propensity to meet a weight loss goal varied significantly with the difficulty of the goal, with easier goals more likely to be met.
(2) **Demographics:** we found that propensity to meet a goal also varied with the user's age and gender, with men and younger adults more likely to meet their goals.
(3) **Initial weight loss:** In Section 5.1, we saw that people who lost more weight over the first 7 days were more likely to meet their goal.
(4) **Self-monitoring:** In Section 5.2, we saw that users who log their weights more frequently over the first 7 days are more likely to meet their goals.
(5) **Calories logged and frequency of logging:** In Section 5.3, we saw that logging more calories indicates a lower likelihood of meeting a goal, but only for committed loggers.
(6) **Motivation:** In Section 5.4, we analyzed free-text motivation provided by users. We use word-level TF-IDF results (for the top 5000 words) as features in this model. This is the same methodology used to train our topic models discussed in Section 5.4.

7.2 Experimental Setup

We aim to predict the likelihood that a user will ever achieve their weight loss goal, after observing their behavior only in the first 7 days. As in Section 6, we filter to users for who there is at least 7 days between the start of their goal and the last weight logged of their goal (though we do not require a minimum number of weight logged during this period). We also remove all users who have already met their goal within the first several days, as it would not be a fair task to predict goal achievement after a user has already achieved their goal. We note that, as shown in Figure 6, most users take far longer than 7 days to meet their goals. We also remove users who didn't last at least 7 days to avoid users who download the app as a curiosity and use it very briefly without ever truly attempting to meet their goal. Further, we filter to users who use the features of the app that we discuss in this paper at least once: specifically, users must log at least one food item, and also must fill in the free-text motivation field when setting their goal. After applying these filters, we arrive at a dataset containing 33,642 goals. (This large reduction from our overall dataset is primarily due to the fact that most users chose not to provide a free-text weight loss reason, as the app provided no motivation for them to do so. Without that restriction, we would have over 900,000 goals in our prediction dataset, and we find that ROC AUC would be only slightly reduced).

We experiment with several classification models, including Linear Regression, Gradient Boosted Trees, Support Vector Machines, and Random Forests. We find that Random Forest models are the most effective for our task, and so we report results from only from those models. Because of the unbalanced dataset (only 18.6% of users achieve their goal) and the trade-off between true and false positive rate associated with prediction, we choose to compare models using the area under the receiver operating characteristic (ROC) curve (AUC) which is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. Thus, a random baseline will score 50% on ROC AUC. We use a 10-fold cross-validation for estimation. We standardize all features to have zero mean and unit variance.

7.3 Results

Figure 10 shows the prediction accuracy of our models. With a model trained on all available features, we achieve a an accuracy of 79% ROC AUC.

We find the most predictive features to be the initial goal and the user's motivation. Strikingly, motivation is significantly more predictive than eating patterns. We note that our motivation model was trained with a simple TF-IDF analysis of the words a user writes in their short free-text goal justification. Future work should explore more sophisticated models and more deeply analyze how free-text motivation for a goal can predict goal achievement rates. Surprisingly, we see that initial weight loss on its own was not

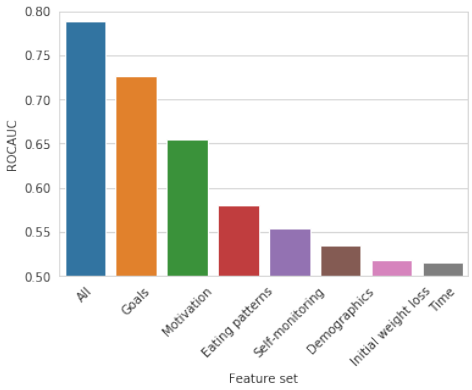


Figure 10: ROC AUC for predicting whether a user will achieve their weight loss goal from observing their behavior over the first seven days after setting it. We report accuracy for each feature set individually and all features together. We reach 79% ROC AUC in a combined model with all features.

a strong predictor of propensity to meet a goal, which is likely explained by the fact that initial loss is somewhat meaningless without knowing the goal that the loss is working towards.

8 DISCUSSION

Goals are a critical component of activity tracking apps. However, long-term goals such as weight loss are rarely met. This paper moves towards addressing this challenge by analyzing user behavior related to both setting and achieving their goals. In particular, we focus on how behavior during the first week after setting a goal predicts whether or not that goal will be achieved. We do so because early detection and notification of goals that are too hard may enable a new class of interventions where users can be encouraged make their goals easier before becoming frustrated and leaving the app. We identify a set of factors which predict goal achievement related to demographics, initial weight loss, self-monitoring, dietary intake, and motivation, summarized in Section 1.

Limitations. There are several important limitations to our work, primarily related to generalizability and causality. First, we note that a weight loss goal is a long-term health goal, which we saw typically takes weeks, months, or even years to achieve. It is an open question as to whether the characteristics of users who achieve short-term goals, such as "walk 10 minutes today", might differ.

We also note that our results are correlational in nature. Ideally, the factors we analyze would represent causal effects to help us understand how to improve weight loss goal achievement. While in our analysis we both build off of prior causal findings and make efforts to reduce potential confounders, the factors that lead to weight loss are enormously complex and causal analysis is often extremely difficult or impossible. However, this is not a necessary requirement for such factors to be useful in machine learning models that predict whether or not a goal is likely to be met. We show that even simple models can predict goal achievement after observing just one week of behavior with reasonably high accuracy.

Implications And Future Work. In addition to uncovering important factors in how people set and achieve their weight loss goals in an activity tracking app, our work also suggests several practical implications. Our work shows that it is possible to predict whether a user will achieve their weight loss goals after observing just their first week of behavior while attempting to meet that goal. This opens the door for activity tracking apps to build a new type of intervention which notifies a user that they seem unlikely to meet their long term goal. Such an intervention might enable users to decide that they should set an easier goal, before they become frustrated and drop out of the app.

Future work should consider several important directions to determine both how to create such an intervention and to determine the effectiveness of such an intervention. First, work should investigate what a activity tracking app do when it recognizes that a goal will never be met, and how users respond to potential interventions. For instance, it is possible than if the user is told too forcefully that they will never meet their goal, perhaps they'll just immediately give up without aiming for an easier goal. On the other hand, if a user is simply assigned an easier goal, it is possible that they will internally still be interested in meeting their first goal, and thus do not truly have an easier one.

9 CONCLUSION

We present a quantitative analysis of the weight loss goals that people set in MyFitnessPal, and the early behaviors that lead to the achievement of those goals. We find that women set more ambitious goals than men, and that younger users set more ambitious goals than older users. We find that most of these goals are never achieved, but that easier goals are far more likely to be achieved than harder ones. We then show that propensity to meet a goal varies significantly by behavior observed during the first week after setting the goal, despite the fact that goals often take months or even years to achieve. We find that, contrary to what findings from medical studies on weight-loss maintenance might suggest, there is no amount of early weight loss which is "too much"; rather more early progress towards a goal predicts a higher propensity to achieve that goal. We also find that people who log more calories are less likely to meet their goal, but only when we account for commitment to logging. We introduce a novel application of topic modeling to show the four primary motivations that users have for selecting their goals. Finally, we show that the results presented in our paper are sufficiently strong to predict, after observing seven days of behavior, whether a user will ultimately meet their goal, with an accuracy of up to 79% ROC AUC. Our findings suggest that activity tracking apps may be able to help their users avoid the pitfalls of failing to meet their goals by asking them them select a more realistic goal after observing their holistic behavior and motivations early-on.

ACKNOWLEDGEMENTS

We thank Claire Donnat, Rok Sosic, Emma Pierson, and MyFitnessPal. M.G. was supported by a Junglee Corporation Stanford Graduate Fellowship. This research has been supported in part by NSF OAC-1835598, ARO MURI, Stanford Data Science Initiative, and Chan Zuckerberg Biohub.

REFERENCES

[1] Marc A Adams, James F Sallis, Gregory J Norman, Melbourne F Hovell, Eric B Hekler, and Elyse Perata. 2013. An adaptive physical activity intervention for overweight adults: a randomized controlled trial. *PloS one* 8, 12 (2013), e82901.

[2] Tim Althoff, Pranav Jindal, and Jure Leskovec. 2017. Online actions with offline impact: How online social networks influence online and offline user behavior. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 537–546.

[3] Arne Astrup and Stephan Rössner. 2000. Lessons from obesity management programmes: greater initial weight loss improves long-term maintenance. *obesity reviews* 1, 1 (2000), 17–19.

[4] Albert Bandura. 1977. Self-efficacy: toward a unifying theory of behavioral change. *Psychological review* 84, 2 (1977), 191.

[5] Thomas Bodenheimer and Margaret A Handley. 2009. Goal-setting for behavior change in primary care: an exploration and status report. *Patient education and counseling* 76, 2 (2009), 174–180.

[6] Lora E Burke, Jing Wang, and Mary Ann Sevick. 2011. Self-monitoring in weight loss: a systematic review of the literature. *Journal of the American Dietetic Association* 111, 1 (2011), 92–102.

[7] Bernard CK Choi, Anita WP Pak, and Jerome CL Choi. 2007. Daily step goal of 10,000 steps: a literature review. *Clinical & Investigative Medicine* 30, 3 (2007), 146–151.

[8] Sunny Consolvo, Predrag Klasnja, David W McDonald, and James A Landay. 2009. Goal-setting considerations for persuasive technologies that encourage physical activity. In *Proceedings of the 4th international Conference on Persuasive Technology*. ACM, 8.

[9] Felicia Cordeiro, Daniel A Epstein, Edison Thomaz, Elizabeth Bales, Arvind K Jagannathan, Gregory D Abowd, and James Fogarty. 2015. Barriers and negative nudges: Exploring challenges in food journaling. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 1159–1162.

[10] Siena F Davis, Marisa A Ellsworth, Hannah E Payne, Shelby M Hall, Joshua H West, and Amber L Nordhagen. 2016. Health behavior theory in popular calorie counting apps: a content analysis. *JMIR mHealth and uHealth* 4, 1 (2016).

[11] Laura Dennison, Leanne Morrison, Gemma Conway, and Lucy Yardley. 2013. Opportunities and challenges for smartphone applications in supporting health behavior change: qualitative study. *Journal of medical Internet research* 15, 4 (2013).

[12] Rod K Dishman, Robert J Vandenberg, Robert W Motl, Mark G Wilson, and David M DeJoy. 2009. Dose relations between goal setting, theory-based correlates of goal setting and increases in physical activity during a workplace trial. *Health education research* 25, 4 (2009), 620–631.

[13] Jeanne M Donaldson and Matthew P Normand. 2009. Using goal setting, self-monitoring, and feedback to increase calorie expenditure in obese adults. *Behavioral Interventions* 24, 2 (2009), 73–83.

[14] Kristina Elfhag and Stephan Rössner. 2005. Who succeeds in maintaining weight loss? A conceptual review of factors associated with weight loss maintenance and weight regain. *Obesity reviews* 6, 1 (2005), 67–85.

[15] Daniel A Epstein, Monica Caraway, Chuck Johnston, An Ping, James Fogarty, and Sean A Munson. 2016. Beyond abandonment to next steps: understanding and designing for life after personal informatics tool use. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 1109–1113.

[16] Gary D Foster, Thomas A Wadden, Renee A Vogt, and Gail Brewer. 1997. What is a reasonable weight loss? Patients' expectations and evaluations of obesity treatment outcomes. *Journal of consulting and clinical psychology* 65, 1 (1997), 79.

[17] Debarchana Ghosh and Rajarshi Guha. 2013. What are we 'tweeting' about obesity? Mapping tweets with topic modeling and Geographic Information System. *Cartography and geographic information science* 40, 2 (2013), 90–102.

[18] Peter M Gollwitzer. 1999. Implementation intentions: strong effects of simple plans. *American psychologist* 54, 7 (1999), 493.

[19] Peter M Gollwitzer and Gabriele Oettingen. 1998. The emergence and implementation of health goals. *Psychology and Health* 13, 4 (1998), 687–715.

[20] Mary Gowin, Marshall Cheney, Shannon Gwin, and Taylor Franklin Wann. 2015. Health and fitness app use in college students: a qualitative study. *American Journal of Health Education* 46, 4 (2015), 223–230.

[21] Katja Herrmanny, Nils Beckmann, Katrin Nachbar, Hanno Sauer, Jürgen Ziegler, and Aysegül Dogangün. 2016. Using Psychophysiological Parameters to Support Users in Setting Effective Activity Goals. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, 1637–1646.

[22] Katja Herrmanny, Jürgen Ziegler, and Aysegül Dogangün. 2016. Supporting users in setting effective goals in activity tracking. In *International Conference on Persuasive Technology*. Springer, 15–26.

[23] Jack F Hollis, Christina M Gullion, Victor J Stevens, Phillip J Brantley, Lawrence J Appel, Jamy D Ard, Catherine M Champagne, Arlene Dalcin, Thomas P Erlinger, Kristine Funk, et al. 2008. Weight loss during the intensive intervention phase of the weight-loss maintenance trial. *American journal of preventive medicine* 35, 2 (2008), 118–126.

[24] Robert W Jeffrey, Rena R Wing, and Randall R Mayer. 1998. Are smaller weight losses or more achievable weight loss goals better in the long term for obese patients? *Journal of consulting and clinical psychology* 66, 4 (1998), 641.

[25] Brian Yoshio Laing, Carol M Mangione, Chi-Hong Tseng, Mei Leng, Ekaterina Vaisberg, Megha Mahida, Michelle Bholat, Eve Glazier, Donald E Morisky, and Douglas S Bell. 2014. Effectiveness of a smartphone application for weight loss compared with usual Care in Overweight Primary Care PatientsA randomized, controlled TrialSmartphone application for weight loss in overweight primary care patients. *Annals of internal medicine* 161, 10_Supplement (2014), S5–S12.

[26] JA Linde, RW Jeffery, RL Levy, NP Pronk, and RG Boyle. 2005. Weight loss goals and treatment outcomes among overweight men and women enrolled in a weight loss trial. *International journal of obesity* 29, 8 (2005), 1002.

[27] Jennifer A Linde, Robert W Jeffery, Emily A Finch, Debbie M Ng, and Alexander J Rothman. 2004. Are unrealistic weight loss goals associated with outcomes for overweight women? *Obesity Research* 12, 3 (2004), 569–576.

[28] Edwin A Locke and Gary P Latham. 2002. Building a practically useful theory of goal setting and task motivation: A 35-year odyssey. *American psychologist* 57, 9 (2002), 705.

[29] Suliman Mansi, Stephan Milosavljevic, Steve Tumilty, Paul Hendrick, Chris Higgs, and David G Baxter. 2015. Investigating the effect of a 3-month workplace-based pedometer-driven walking programme on health-related quality of life in meat processing workers: a feasibility study within a randomized controlled trial. *BMC public health* 15, 1 (2015), 410.

[30] Maureen T McGuire, Rena R Wing, Mary L Klem, Wei Lang, and James O Hill. 1999. What predicts weight regain in a group of successful weight losers? *Journal of consulting and clinical psychology* 67, 2 (1999), 177.

[31] Kathryn Mercer, Melissa Li, Lora Giangregorio, Catherine Burns, and Kelly Grindrod. 2016. Behavior change techniques present in wearable activity trackers: a critical analysis. *JMIR mHealth and uHealth* 4, 2 (2016).

[32] Sean A Munson and Sunny Consolvo. 2012. Exploring goal-setting, rewards, self-monitoring, and sharing to motivate physical activity. In *Pervasive computing technologies for healthcare (PervasiveHealth), 2012 6th international conference on*. IEEE, 25–32.

[33] Inbal Nahum-Shani, Shawna N Smith, Bonnie J Spring, Linda M Collins, Katie Witkiewitz, Ambuj Tewari, and Susan A Murphy. 2017. Just-in-time adaptive interventions (JITAIs) in mobile health: key components and design principles for ongoing health behavior support. *Annals of Behavioral Medicine* 52, 6 (2017), 446–462.

[34] Matthew P Normand. 2008. Increasing physical activity through self-monitoring, goal setting, and feedback. *Behavioral Interventions* 23, 4 (2008), 227–236.

[35] Faryle Nothwehr and Jingzhen Yang. 2006. Goal setting frequency and the use of behavioral strategies related to diet and physical activity. *Health education research* 22, 4 (2006), 532–538.

[36] Ellinor K Olander, Helen Fletcher, Stefanie Williams, Lou Atkinson, Andrew Turner, and David P French. 2013. What are the most effective techniques in changing obese individuals' physical activity self-efficacy and behaviour: a systematic review and meta-analysis. *International Journal of Behavioral Nutrition and Physical Activity* 10, 1 (2013), 29.

[37] Harm op den Akker, Valerie M Jones, and Hermie J Hermens. 2014. Tailoring real-time physical activity coaching systems: a literature survey and model. *User modeling and user-adapted interaction* 24, 5 (2014), 351–392.

[38] Harm op den Akker, Randy Klaassen, Rieks op den Akker, Valerie M Jones, and Hermie J Hermens. 2013. Opportunities for smart & tailored activity coaching. In *Computer-Based Medical Systems (CBMS), 2013 IEEE 26th International Symposium on*. IEEE, 546–547.

[39] Michael J Paul and Mark Dredze. 2014. Discovering health topics in social media using topic models. *PloS one* 9, 8 (2014), e103408.

[40] Erin S Pearson. 2012. Goal setting as a health behavior change strategy in overweight and obese adults: a systematic literature review examining intervention components. *Patient education and counseling* 87, 1 (2012), 32–42.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[42] Mashfiqui Rabbi, Min Hane Aung, Mi Zhang, and Tanzeem Choudhury. 2015. MyBehavior: automatic personalized health feedback from user behaviors and preferences using smartphones. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 707–718.

[43] John Rooksby, Mattias Rost, Alistair Morrison, and Matthew Chalmers Chalmers. 2014. Personal tracking as lived informatics. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. ACM, 1163–1172.

[44] Stephanie Schoeppe, Stephanie Alley, Wendy Van Lippevelde, Nicola A Bray, Susan L Williams, Mitch J Duncan, and Corneel Vandelanotte. 2016. Efficacy of interventions that use apps to improve diet, physical activity and sedentary behaviour: a systematic review. *International Journal of Behavioral Nutrition and Physical Activity* 13, 1 (2016), 127.

[45] Mical Kay Shilts, Marcel Horowitz, and Marilyn S Townsend. 2004. Goal setting as a strategy for dietary and physical activity behavior change: a review of the

literature. *American Journal of Health Promotion* 19, 2 (2004), 81–93.

[46] Victor J Strecher, Gerard H Seijts, Gerjo J Kok, Gary P Latham, Russell Glasgow, Brenda DeVellis, Ree M Meertens, and David W Bulger. 1995. Goal setting as a strategy for health behavior change. *Health education quarterly* 22, 2 (1995), 190–200.

[47] Petar Veličković, Laurynas Karazija, Nicholas D Lane, Sourav Bhattacharya, Edgar Liberis, Pietro Liò, Angela Chieh, Otmane Bellahsen, and Matthieu Vegreville. 2018. Cross-modal recurrent models for weight objective prediction from multimodal time-series data. In *Proceedings of the 12th EAI International Conference on Pervasive Computing Technologies for Healthcare*. ACM, 178–186.

[48] Dong Wen, Xingting Zhang, Xingyu Liu, and Jianbo Lei. 2017. Evaluating the Consistency of Current Mainstream Wearable Devices in Health Monitoring: A Comparison Under Free-Living Conditions. *Journal of medical Internet research* 19, 3 (2017).

[49] David F Williamson, Mary K Serdula, Robert F Anda, Alan Levy, and Tim Byers. 1992. Weight loss attempts in adults: goals, duration, and rate of weight loss. *American Journal of Public Health* 82, 9 (1992), 1251–1257.

[50] Rena R Wing and Suzanne Phelan. 2005. Long-term weight loss maintenance–. *The American journal of clinical nutrition* 82, 1 (2005), 222S–225S.

Link Prediction on N-ary Relational Data

Saiping Guan
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences; School of Computer and Control Engineering, University of Chinese Academy of Sciences
guansaiping@ict.ac.cn

Xiaolong Jin*
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences; School of Computer and Control Engineering, University of Chinese Academy of Sciences
jinxiaolong@ict.ac.cn

Yuanzhuo Wang
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences; School of Computer and Control Engineering, University of Chinese Academy of Sciences
wangyuanzhuo@ict.ac.cn

Xueqi Cheng
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences; School of Computer and Control Engineering, University of Chinese Academy of Sciences
cxq@ict.ac.cn

ABSTRACT

With the overwhelming popularity of Knowledge Graphs (KGs), researchers have poured attention to link prediction to complete KGs for a long time. However, they mainly focus on promoting the performance on binary relational data, where facts are usually represented as triples in the form of (*head entity*, *relation*, *tail entity*). In practice, n-ary relational facts are also ubiquitous. When encountering such facts, existing studies usually decompose them into triples by introducing a multitude of auxiliary virtual entities and additional triples. These conversions result in the complexity of carrying out link prediction concerning more than two arities. It has even proven that they may cause loss of structural information. To overcome these problems, in this paper, without decomposition, we represent each n-ary relational fact as a set of its role-value pairs. We further propose a method to conduct Link Prediction on N-ary relational data, thus called NaLP, which explicitly models the relatedness of all the role-value pairs in the same n-ary relational fact. Experimental results validate the effectiveness and merits of the proposed NaLP method.

CCS CONCEPTS

• **Computing methodologies → Reasoning about belief and knowledge**.

KEYWORDS

Link prediction; n-ary relational fact; knowledge graph; neural network

ACM Reference Format:
Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2019. Link Prediction on N-ary Relational Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313414

*Corresponding author: Xiaolong Jin (jinxiaolong@ict.ac.cn)

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313414

1 INTRODUCTION

Since Google announced Knowledge Graph (KG) in 2012[1], KG has been increasingly popular in these years. And link prediction on KG has caught much attention for a long time, to complete KG and further promote KG based applications. Usually, a KG is represented as a set of binary relational triples in the form of (*head entity*, *relation*, *tail entity*), where n-ary relational facts are decomposed into multiple triples via introducing virtual entities, such as the Compound Value Type (CVT) entities in Freebase [1]. Actually, n-ary relational facts are not in the minority, as an example, in Freebase, more than $\frac{1}{3}$ of its entities are involved in n-ary relational facts [31]. Based on these public KGs of binary form, researchers have developed numerous methods for link prediction [21, 29], which includes the tasks of head entity prediction, relation prediction and tail entity prediction. The popular tensor/matrix based methods [4, 17, 20, 23, 27, 28] and translation based ones [2, 11–13, 15, 16, 30, 32] have been developed and improved for so long. They view triples in a KG as entries in a tensor/matrix, or relational translations between head entities and tail entities. Relatively newly developed neural network based methods [3, 5, 7, 19, 25, 26] use neural network to learn the information propagation of the entities across the edges (relations) of a KG and integrate existing effective methods, or use neural network to model triples directly. Regardless of these various methods, link prediction is essentially reduced to scoring triples by designing different score functions and casted into a ranking/classification task, with valid triples preceding invalid ones in terms of their scores or having different labels from invalid ones.

However, manipulating n-ary relational facts into triples has some deficiencies. Firstly, it makes link prediction involving more than two arities need to consider many triples. Since existing link prediction methods on triples, like the above methods, usually model triples one by one individually and correspondingly conduct link prediction on only a single triple each time, it is intricate to

[1] https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html

carry out link prediction involving more than one triple. Secondly, there is a loss of structural information in some of the conversions from n-ary relational facts to triples [31], which may lead to inaccurate link prediction. Thirdly, the added virtual entities along with the corresponding triples bring in more parameters to be learned.

With these considerations in mind, in this paper, each n-ary relational fact is represented as a set of its role-value pairs instead of being converted into multiple triples. Hence, no additional entities and data are introduced, and all the structural information is retained. Then, we propose a method to handle link prediction on n-ary relational data. Specifically, in this paper, link prediction on n-ary relational data is to predict the missing value or role of a relational fact.

Recently, there are also a few studies for link prediction on n-ary relational data directly. In m-TransH [31], a relation (binary or n-ary relation) is defined by the mappings from a sequence of roles corresponding to this type of relation, to their values. Each specific mapping is an instance of this relation. Then, a translation based method is proposed to model these instances. However, m-TransH does not take the relatedness of the components in the same n-ary relational fact into consideration. RAE [34] further improves m-TransH by complementally modeling the relatedness of values, i.e., the likelihood that two values co-participate in a common instance. Although RAE achieves favorable performance, it does not consider the roles explicitly when evaluating the above likelihood. Actually, under different sequences of roles (corresponding to different relations), the relatedness of two values is greatly different. For example, under the role sequence (*person*, *award*, *point in time*, *together with*) (in this paper, the examples including the role names are all derived from the raw data in Wikidata[2]), *Marie Curie* and *Henri Becquerel* are more related than under the role sequence (*person*, *spouse*, *start time*, *end time*, *place of marriage*), since they won *Nobel Prize in Physics* in *1903* together. To address these problems, our method explicitly models the relatedness of the role-value pairs involved in the same n-ary relational fact. The above example also elucidates its necessity.

In general, the main contributions of this paper can be summarized as follows:

- We advocate a representation form for n-ary relational facts, which represents each n-ary relational fact as a set of its role-value pairs.
- We propose a link prediction method NaLP on n-ary relational data, which captures the relatedness of the role-value pairs in the same n-ary relational fact explicitly.
- Experimental results and further analyses testify the effectiveness and superiority of the proposed NaLP method.

In addition, as publicly available n-ary relational datasets are limited, we build a practical one WikiPeople based on the raw data in Wikidata, which is available on github[3] for further research.

2 RELATED WORKS

According to the fact type that link prediction focuses on, link prediction and its methods can be divided into two categories, i.e., link prediction on binary relational data and link prediction on n-ary relational data.

2.1 Link Prediction on Binary Relational Data

The rapid development of KG has triggered the spring up of a great many methods, including tensor/matrix based methods, translation based methods, and neural network based ones, for link prediction on binary relational data.

Among tensor/matrix based methods [4, 17, 20, 23, 27, 28], the well-known one, RESCAL [23], views a KG as a three-way tensor, where each way corresponds to head entities, relations and tail entities, respectively. And the entry corresponds to the score indicating the validity of the triple formed by the values of the thee ways. If a triple (h, r, t) is observed in the KG, the entry is set to 1, otherwise, set to 0. Through minimizing the reconstruction error of the tensor, the embeddings of entities and relations are learned. Then, the reconstructed tensor from the learned embeddings is used to conduct link prediction, with triples corresponding to entries of high score as valid ones. Similarly, a matrix factorization based method, ComplEx [28], constructs a matrix for each relation, which is decomposed and optimized via minimizing the reconstruction error like RESCAL to learn the embeddings. Differently, complex values are used to define the embeddings, so as to deal with antisymmetric relations effectively.

Translation based methods are derived from the simple but effective method TransE [2]. Its popular translation assumption advocates that valid triples can be viewed as relational translation operations from head entities to tail entities. Based on this assumption, different types of translations and accordingly different score functions measuring the similarity between the relational translation results of head entities and the target tail entities are defined [2, 11–13, 15, 16, 30, 32]. The embeddings of entities and relations are then learned via minimizing score function based loss, which encourages valid triples to have much larger scores than invalid ones. Among these methods, the seminal one, TransE, performs translation operations in a shared space of entities and relations. Further, TransH [30] thinks translation operations should be conducted in relation-specific hyperplanes. Thus, it relates each relation with a hyperplane. Then, entities are projected into the hyperplanes of relations before translations.

Inspired by the excellent performance of neural network in various applications, it has been introduced into link prediction on binary relational data and achieved extremely promising results [3, 5, 7, 19, 25, 26]. Among them, the R-GCN method [25] introduces relational graph convolutional networks as its encoder to get the embeddings of entities, and uses the existing method [33] as its decoder to compute the scores of triples with the embeddings of entities from the encoder and the embeddings of relations randomly initialized. Differently, ConvE [3] models entity prediction directly, which reshapes and concatenates the embeddings of the known entity and relation to apply two-dimensional convolution on the concatenated matrix. Then, after a fully connected projection, it generates the prediction entity. By comparing it with the target entity, ConvE minimizes a binary cross-entropy loss. SENN [7] integrates the prediction tasks of head entities, relations and tail entities in a unified neural network framework based on

[2]https://www.wikidata.org/wiki/Wikidata:Main_Page
[3]https://github.com/gsp2014/WikiPeople

shared embeddings. Fully connected layers are applied to model the prediction processes. ConvKB [19] represents each triple as a three-column matrix, where each column vector corresponds to an element of the triple. This matrix is fed into a convolution layer to generate different feature maps, which are then concatenated into a single feature vector representing the triple. After passing it to a fully connected layer, ConvKB obtains a score indicating the validity of the triple.

2.2 Link Prediction on N-ary Relational Data

As n-ary relational facts are complicated and usually difficult to deal with, most of existing studies convert them into multiple triples, with the introduction of virtual entities and additional triples [1], and research works for link prediction on n-ary relational data directly are relatively much scarcer and newer.

Based on the link prediction method TransH [30] on binary relational data, m-TransH [31] is proposed to generalize it to n-ary relational data. The m-TransH method presents a mathematical definition of relations (including binary and n-ary relations) as the mappings from sequences of roles to their values. Each mapping is an instance of the corresponding relation. Then, the score function of an instance is defined by the weighted sum of the projection results from its values to its relation hyperplane, where the weights are the real numbers projected from its roles. RAE [34] further considers the relatedness of values, i.e., the likelihood that two values co-participate in a common instance, and adds this relatedness loss with a weighted hyper-parameter to the embedding loss of m-TransH. Although with the additional modeling of the relatedness of values, RAE outperforms m-TransH, it does not look into the roles when computing the relatedness. Since roles are also a fundamental part of instances, taking them into consideration may make a difference as illustrated afore.

3 THE NALP METHOD

In this paper, link prediction on n-ary relational data is transformed into estimating whether a relational fact is valid and then casted into a classification task.

In this section, we first demonstrate our representation form of n-ary relational facts, then outline the framework of the proposed NaLP method and elaborate its two key components, respectively. Subsequently, some characteristics of NaLP are discussed. And the details of model training are introduced in the following part.

3.1 The Representation of N-ary Relational Facts

In this paper, we represent each n-ary relational fact as a set of its role-value pairs. As an example, the representation of the fact that *Marie Curie* received *Willard Gibbs Award*, is as follows:

$$\{person : Marie\ Curie, award : Willard\ Gibbs\ Award\}.$$

It is an n-ary relational fact of arity 2, i.e., a binary relational fact, as it has two role-value pairs.

And the facts where some roles have more than one value are represented similarly. For example, the representation of the fact that *Marie Curie* received *Nobel Prize in Physics* in *1903* together with *Henri Becquerel* and *Pierre Curie*, is as follows:

$$\{person : Marie\ Curie, award : Nobel\ Prize\ in\ Physics, point\ in\ time : 1903, together\ with : Henri\ Becquerel, together\ with : Pierre\ Curie\}.$$

It is an n-ary relational fact of arity 5, with 5 role-value pairs.

Formally, given an n-ary relational fact with n roles, and each role r_i having m_i values ($i = 1, 2, \ldots, n$), the representation of this n-ary relational fact is the following set of role-value pairs:

$$\{r_1 : v_1, \ldots, r_1 : v_{m_1}, r_2 : v_{m_1+1}, \ldots, r_2 : v_{m_1+m_2}, \ldots, r_n : v_{m_1+m_2+\cdots+m_{n-1}+1}, \ldots, r_n : v_{m_1+m_2+\cdots+m_n}\}.$$

It is an n-ary relational fact of arity $(m_1 + m_2 + \cdots + m_n)$.

3.2 The Framework of NaLP

Then, we propose a link prediction method NaLP on n-ary relational data. It is designed based on two considerations. On one hand, a role and its value are tightly linked to each other, thus should be bound together. On the other hand, as mentioned in the beginning of Section 3, we aim to judge the validity of a relational fact, i.e., a set of role-value pairs. That is, for a set of role-value pairs, we need to determine whether it is able to form a valid relational fact. Intuitively, all the role-value pairs of a valid relational fact are closely related. And if all the role-value pairs from a set are closely related, then, we assume that this set of role-value pairs has high probability to constitute a valid relational fact. With these considerations in mind, the framework of NaLP is designed as illustrated in Figure 1, which consists of two key components, role-value pair embedding and relatedness evaluation. For clarity, in the figure, we only exemplify one n-ary relational fact in the following form:

$$Rel = \{r_1 : v_1, r_2 : v_2, \ldots, r_m : v_m\},$$

where m is the arity of the relational fact *Rel*. The role set of *Rel* is denoted as $R_{Rel} = \{r_1, r_2, \ldots, r_m\}$ and r_i may be the same to r_j ($i, j = 1, 2, \ldots, m, i \neq j$), as demonstrated in Section 3.1; the value set of *Rel* is denoted as $V_{Rel} = \{v_1, v_2, \ldots, v_m\}$. In what follows, as an example, we illustrate the learning details of this n-ary relational fact.

For the given relational fact *Rel*, the embeddings of its roles in R_{Rel} and values in V_{Rel} are looked up from the embedding matrices $\mathbf{M_R} \in \mathbb{R}^{|R|\times k}$ of roles and $\mathbf{M_V} \in \mathbb{R}^{|V|\times k}$ of values, respectively, where R is the set of all the roles on dataset; V is the set of all the values; k is the latent dimension of the vector space. Following the convention, in what follows, the embeddings are denoted with the same letters but in boldface. As depicted in Figure 1, after passing these embeddings to the role-value pair embedding component, we get the embedding matrix of the m role-value pairs, i.e., *Rel* (see Section 3.3). This resulting embedding matrix is fed into the relatedness evaluation component to compute the pair-wise relatedness of all the role-value pairs, then estimate their overall relatedness, which is used to obtain the evaluation score (see Section 3.4). Particularly, this framework equips NaLP with the ability to be permutation-invariant to the input order of the role-value pairs and deal with relational facts of different arities (see Section 3.5). Subsequently, the evaluation score is used to generate loss (see Section 3.6).

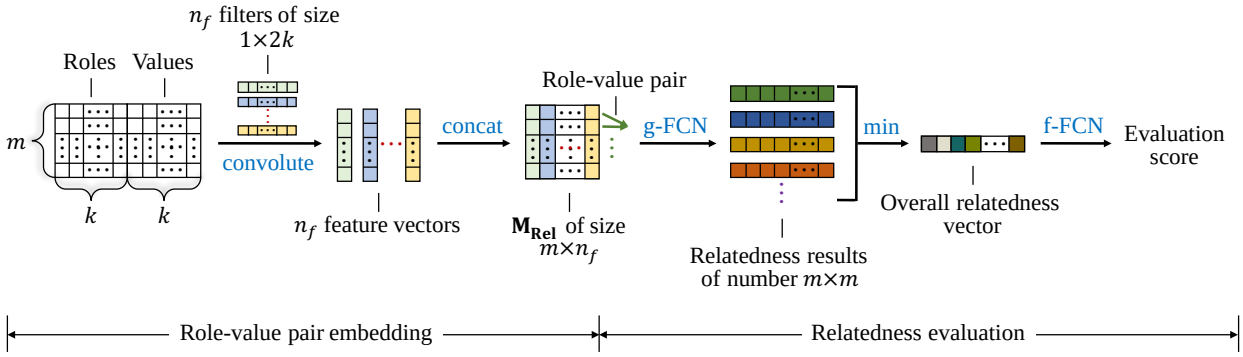


Figure 1: The framework of the proposed NaLP method.

3.3 Role-Value Pair Embedding

The role-value pair embedding component aims to obtain the embeddings of the input role-value pairs. This process contains two sub-processes, i.e., convolution and concatenation, corresponding to "convolute" and "concat" in Figure 1, respectively.

3.3.1 "convolute". As presented in Figure 1, before convolution, the embeddings of the roles r_i and the corresponding values v_i ($i = 1, 2, \ldots, m$) in the relational fact *Rel* are concatenated, and form a matrix of dimension $m \times 2k$, where each role-value pair makes up a row.

Then, this concatenated matrix is passed to the convolution with filter set Ω of n_f filters. To capture the features of the role-value pairs, the dimensions of the filters are all set to $1 \times 2k$. After convolution, we get n_f results of dimension m. And Rectified Liner Units (ReLU) [18] is applied to these results to get n_f feature vectors.

3.3.2 "concat". We concatenate the n_f feature vectors to form a matrix of dimension $m \times n_f$. Then, this matrix can be treated as the embeddings of the m role-value pairs, and each row corresponds to the embedding of one role-value pair. Specifically, each entry of a row encodes the feature of this dimension.

Formally, the embedding matrix $\mathbf{M_{Rel}} \in \mathbb{R}^{m\times n_f}$ of the role-value pairs, i.e., the relational fact *Rel* is defined as:

$$\mathbf{M_{Rel}} = \text{concat}\left(\Upsilon(\text{concat}(\mathbf{R_{Rel}}, \mathbf{V_{Rel}}) * \Omega)\right), \tag{1}$$

where Υ is the ReLU function [18], i.e., $\Upsilon(x) = \max(0, x)$; $*$ denotes the convolution operator; $\text{concat}(\mathbf{R_{Rel}}, \mathbf{V_{Rel}})$ is the aforementioned embedding concatenation process of the role-value pairs before convolution.

3.4 Relatedness Evaluation

This component computes the pair-wise relatedness of all the input role-value pairs, estimates their overall relatedness and obtains the evaluation score, corresponding to "g-FCN", "min" and "f-FCN" in Figure 1, respectively. Before introducing these sub-processes, let's see the principle behind this component.

3.4.1 The Principle. The principle of its design goes deep into determining whether a set of role-value pairs is able to form a valid relational fact. As demonstrated in Section 3.2, this is reduced to computing the overall relatedness of all the role-value pairs in the set. However, the overall relatedness of a set with more than two objects is intricate to measure. When the number of objects is large, corresponding to relational fact of high arity, the evaluation is further complicated. Whereas, computing the relatedness between two objects is relatively much simpler. Thus, how to determine the tricky overall relatedness via the simple relatedness between the role-value pairs?

Straightforwardly, if all the role-value pairs form a closely related set, i.e., a valid relational fact, then every two pairs from the set are greatly related. Thus, for any two pairs, the values of their relatedness feature vector, measuring the relatedness in many different views, are all expected to be sufficiently large. That is, for each feature dimension, the minimum over this dimension of every two pairs is not allowed to be too small. Hence, based on this observation, we apply element-wise minimizing over the pair-wise relatedness of all the role-value pairs to approximately evaluate their overall relatedness. Accordingly, the relatedness evaluation component is designed to contain the computation of the relatedness between the role-value pairs and the overall relatedness of all the role-value pairs, before estimating the final evaluation score.

3.4.2 "g-FCN". The relatedness between the role-value pairs is captured via a Fully Connected Network (FCN) with ReLU as its activation function. In particular, the widely used FCN is also adopted to infer the ways in which two objects are related and achieves marvelous performance in computer vision [8, 24]. In this paper, it turns out that a one-layer FCN already works very well. Thus, only a fully connected layer is adopted in this sub-process.

Concretely, the embeddings of any two role-value pairs are concatenated to form a vector of dimension $2n_f$, and then fed into the fully connected layer with n_{gFCN} nodes. The resulting vector of dimension n_{gFCN} is the relatedness feature vector with entries greater than or equal to 0, each dimension of which indicates the relatedness of the input two role-value pairs in some respect.

3.4.3 "min". As discussed in Section 3.4.1, the overall relatedness of all the role-value pairs is simply approximated by element-wise

minimizing over their pair-wise relatedness. Thus, the overall relatedness $\mathcal{R}_{\mathbf{Rel}}$ of Rel is defined as:

$$\begin{aligned}\mathcal{R}_{\mathbf{Rel}} &= \min_{i,j=1}^{m}\Big(\text{g-FCN}\left(\text{concat}([\mathbf{M}_{\mathbf{Rel}}]_i, [\mathbf{M}_{\mathbf{Rel}}]_j)\right)\Big) \\ &= \min_{i,j=1}^{m}\Upsilon\big(\text{concat}([\mathbf{M}_{\mathbf{Rel}}]_i, [\mathbf{M}_{\mathbf{Rel}}]_j)\mathbf{W}_{\mathbf{gFCN}} + \mathbf{b}_{\mathbf{gFCN}}\big),\end{aligned} \tag{2}$$

where $\min(\cdot)$ is the element-wise minimizing function; $[\mathbf{x}]_i$ is the i-th entry of $\mathbf{x}$; $\mathbf{W}_{\mathbf{gFCN}}$ of dimension $2n_f \times n_{\text{gFCN}}$ and $\mathbf{b}_{\mathbf{gFCN}}$ of dimension n_{gFCN} are the weight matrix and bias vector of g-FCN, respectively. The resulting $\mathcal{R}_{\mathbf{Rel}}$ is a vector of dimension n_{gFCN}, with each entry implying the degree of the overall relatedness in terms of certain feature corresponding to that dimension.

3.4.4 "f-FCN". Then, $\mathcal{R}_{\mathbf{Rel}}$ is used to generate the evaluation score. A one-layer FCN is also found to be sufficient and applied in this sub-process. In this way, the evaluation score $s(Rel)$ of Rel is defined as:

$$\begin{aligned}s(Rel) &= \text{f-FCN}(\mathcal{R}_{\mathbf{Rel}}) \\ &= \mathcal{R}_{\mathbf{Rel}}\mathbf{W}_{\mathbf{fFCN}} + b_{\text{fFCN}},\end{aligned} \tag{3}$$

where $\mathbf{W}_{\mathbf{fFCN}}$ of dimension $n_{\text{gFCN}} \times 1$ and b_{fFCN} are the weight matrix and bias variable of f-FCN, respectively.

3.5 Look into NaLP

Note that, in m-TransH [31] and RAE [34], all relations along with their role sequences are predefined, and the inputs of the models are the instances of these relations, i.e., the sequences of ordered values. Since the i-th value of an input instance corresponds to the i-th role of its relation, the order of the input values are usually not allowed to be changed randomly. Whereas, each input of the proposed NaLP method is a set of role-value pairs, corresponding to a relational fact. Thus, the order of the objects in each input matters in m-TransH and RAE, while it has no impact on NaLP. Actually, NaLP is permutation-invariant to the input order of the role-value pairs and is able to cope with relational facts of different arities. Let's look into these two characteristics.

3.5.1 A Simple Explanation of Permutation Invariance. Suppose that the role-value pairs of Rel are input into NaLP in the original order with the indexes from 1 to m, and a permutation ρ to the indexes results in a new order with the indexes from $\rho(1)$ to $\rho(m)$.

On these grounds, we simply explain the permutation invariance as follows:

- In the process of role-value pair embedding, the embedding matrix $\mathbf{M}_{\rho(\mathbf{Rel})}$ of the permutated role-value pairs $\rho(Rel)$ is the permutation of the original $\mathbf{M}_{\mathbf{Rel}}$, with the $\rho(i)$-th row of $\mathbf{M}_{\mathbf{Rel}}$ placed in the i-th row of $\mathbf{M}_{\rho(\mathbf{Rel})}$, i.e.,

$$[\mathbf{M}_{\rho(\mathbf{Rel})}]_i = [\mathbf{M}_{\mathbf{Rel}}]_{\rho(i)}. \tag{4}$$

- In the process of relatedness evaluation, the overall relatedness $\mathcal{R}_{\rho(\mathbf{Rel})}$ of $\rho(Rel)$ is as follows:

$$\begin{aligned}&\mathcal{R}_{\rho(\mathbf{Rel})} \\ &= \min_{i,j=1}^{m}\Big(\text{g-FCN}\left(\text{concat}([\mathbf{M}_{\rho(\mathbf{Rel})}]_i, [\mathbf{M}_{\rho(\mathbf{Rel})}]_j)\right)\Big) \\ &= \min_{i,j=1}^{m}\Upsilon\left(\text{concat}([\mathbf{M}_{\rho(\mathbf{Rel})}]_i, [\mathbf{M}_{\rho(\mathbf{Rel})}]_j)\mathbf{W}_{\mathbf{gFCN}} + \mathbf{b}_{\mathbf{gFCN}}\right) \\ &= \min_{i,j=1}^{m}\Upsilon\left(\text{concat}([\mathbf{M}_{\mathbf{Rel}}]_{\rho(i)}, [\mathbf{M}_{\mathbf{Rel}}]_{\rho(j)})\mathbf{W}_{\mathbf{gFCN}} + \mathbf{b}_{\mathbf{gFCN}}\right).\end{aligned} \tag{5}$$

That is, we obtain $m \times m$ relatedness results, the same to those in the original version, but with the order of the results changed. Since $\min(\cdot)$ is permutation-equivariant, we have:

$$\mathcal{R}_{\rho(\mathbf{Rel})} = \mathcal{R}_{\mathbf{Rel}}. \tag{6}$$

- Then, the evaluation score $s(\rho(Rel))$ of $\rho(Rel)$ is as follows:

$$\begin{aligned}s(\rho(Rel)) &= \text{f-FCN}(\mathcal{R}_{\rho(\mathbf{Rel})}) \\ &= \mathcal{R}_{\rho(\mathbf{Rel})}\mathbf{W}_{\mathbf{fFCN}} + b_{\text{fFCN}} \\ &= \mathcal{R}_{\mathbf{Rel}}\mathbf{W}_{\mathbf{fFCN}} + b_{\text{fFCN}} \\ &= s(Rel).\end{aligned} \tag{7}$$

- Therefore, NaLP is permutation-invariant to the input order of the role-value pairs.

3.5.2 Handling Relational Facts of Different Arities. Suppose that two batches of relational facts of arities m and m', respectively, are input into NaLP subsequently. In the process of role-value pair embedding, they result in the embedding matrices of the role-value pairs with m and m' rows, respectively. In the process of relatedness evaluation, correspondingly, we obtain $m \times m$ and $m' \times m'$ relatedness results. As $\min(\cdot)$ in Equation (2) returns one vector among all these results of number $m \times m$ or $m' \times m'$, the number of results makes no difference. Thus, NaLP is able to deal with relational facts of different arities.

3.6 Model Training

In the following, we present the loss function and the training process of the proposed NaLP method in detail.

3.6.1 The Loss Function. As above described, we obtain the evaluation score $s(Rel)$ of Rel. Following ComplEx [28], the loss of Rel is defined as:

$$\mathcal{L}(Rel) = \log\left(1 + e^{-I_{Rel}s(Rel)}\right), \tag{8}$$

where $I_{Rel} = 1$, if Rel is a valid relational fact, and $I_{Rel} = -1$, otherwise.

It is straightforward to optimize NaLP with the standard back-propagation. The popular stochastic optimization method Adam [14] with hyper-parameter learning rate λ is used as the optimizer in this paper.

3.6.2 The Training Process. Algorithm 1 presents the training process of the NaLP method.

Before training, the training set is reorganized into several training groups, each of which keeps the relational facts of the same arity (Line 1). In order to guarantee that NaLP masters basic relational facts of low arities before learning relational facts of high arities, we sort the above resulting groups by their arities in ascending order (Line 2). Similar operations are applied to train model on the path queries of length 1 and then all the path queries [9]. Subsequently, the embedding matrices $\mathbf{M}_{\mathbf{R}}$ and $\mathbf{M}_{\mathbf{V}}$ are randomly initialized from uniform distribution, respectively, with $-\frac{1}{\sqrt{k}}$ as minimum and $\frac{1}{\sqrt{k}}$ as maximum; truncated normal distribution with 0.0 as mean and 0.1 as standard deviation is adopted as initializer to initialize all the filters in Ω [19]; $\mathbf{W}_{\mathbf{gFCN}}$ and $\mathbf{W}_{\mathbf{fFCN}}$ are randomly initialized via xavier initializer [6], respectively; $\mathbf{b}_{\mathbf{gFCN}}$ and b_{fFCN} are randomly initialized with zeros (Line 3).

Algorithm 1 The training process of the NaLP method.

Input: Training set T, role set R, value set V, max number of epochs *nepoch*, embedding dimension k, batch size β, number of filters n_f in Ω, the hyper-parameter n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$.
Output: $\mathbf{W}_{\mathbf{R}}$ and $\mathbf{W}_{\mathbf{V}}$, as well as the parameters of NaLP.

1: $T \leftarrow$ group T by the arities of the facts;
2: $T \leftarrow$ sort the groups in T by their arities in ascending order;
3: **Initialize** $\mathbf{M}_{\mathbf{R}}, \mathbf{M}_{\mathbf{V}} \leftarrow \text{uniform}(-\frac{1}{\sqrt{k}}, \frac{1}{\sqrt{k}})$;
$\Omega \leftarrow \text{truncated_normal}(0.0, 0.1)$;
$\mathbf{W}_{\mathbf{gFCN}}, \mathbf{W}_{\mathbf{fFCN}} \leftarrow \text{xavier_initializer}()$;
$\mathbf{b}_{\mathbf{gFCN}}, b_{\text{fFCN}} \leftarrow \text{zeros_initializer}()$;
4: **repeat**
5: **for** each group $T' \in T$ **do**
6: **for** $j = 1$ to $\left\lceil \frac{|T'|}{\beta} \right\rceil$ **do**
7: $S^+ \leftarrow \text{sample}(T', \beta)$; //sample β training facts from T'
8: $S^- \leftarrow \text{neg_sample}(S^+, \beta)$; //obtain β negative samples
9: $S \leftarrow S^+ \cup S^-$;
10: $loss \leftarrow 0$;
11: **for** $\forall Rel \in S$ **do**
12: $\mathbf{R}_{\mathbf{Rel}} \leftarrow \text{look-up}(\mathbf{M}_{\mathbf{R}}, R_{Rel})$; //look up R_{Rel}
13: $\mathbf{V}_{\mathbf{Rel}} \leftarrow \text{look-up}(\mathbf{M}_{\mathbf{V}}, V_{Rel})$; //look up V_{Rel}
14: $\mathbf{M}_{\mathbf{Rel}} \leftarrow$ get the embedding matrix of the role-value pairs following Equation (1);
15: $\mathcal{R}_{\mathbf{Rel}} \leftarrow$ compute the overll relatedness of *Rel* following Equation (2);
16: $s(Rel) \leftarrow$ generate the evaluation score following Equation (3);
17: $\mathcal{L}(Rel) \leftarrow$ compute the loss of *Rel* following Equation (8);
18: $loss \leftarrow loss + \mathcal{L}(Rel)$;
19: **end for**
20: Update the embeddings of the roles and values in S (i.e., the related rows of $\mathbf{M}_{\mathbf{R}}$ and $\mathbf{M}_{\mathbf{V}}$), the filters in Ω, $\mathbf{W}_{\mathbf{gFCN}}$, $\mathbf{W}_{\mathbf{fFCN}}$, $\mathbf{b}_{\mathbf{gFCN}}$, as well as b_{fFCN} via $\nabla loss$;
21: **end for**
22: **end for**
23: **until** the evaluation result on the validation set drops continuously or this process has been iterated for *nepoch* times

During training, Lines 5-22 are repeated until the evaluation result on the validation set drops continuously or reaching the max number of epochs *nepoch*. In this paper, *nepoch* is set to 5000. In each epoch, $\left\lceil \frac{|T'|}{\beta} \right\rceil$ batches of training facts are sampled from each training group T', where $\lceil \cdot \rceil$ is the ceiling function. For each selected training fact, similar to the negative sampling method adopted in TransE [2], we randomly replace one of its values with a random value that the corresponding role holds, or one of its roles is replaced with a relatively lower probability, to obtain a negative sample not contained in the dataset (Line 8). Thereafter, the selected training facts and their negative samples are used to train the model. For each such relational fact *Rel*, the embeddings of its roles and values are looked up from $\mathbf{M}_{\mathbf{R}}$ and $\mathbf{M}_{\mathbf{V}}$, respectively (Lines 12 and 13). Then, they are fed into the role-value pair embedding component to obtain the embedding matrix of the role-value pairs (Line 14). After passing this embedding matrix to the relatedness evaluation component, we obtain the overall relatedness $\mathcal{R}_{\mathbf{Rel}}$ and the evaluation score $s(Rel)$ of *Rel* (Lines 15 and 16). Subsequently, based on $s(Rel)$, loss is computed and added to the loss of the current batch (Lines 17 and 18). Finally, parameter update is performed in batch mode (Line 20).

Specifically, batch normalization [10] is applied after convolution in the role-value pair embedding component to stabilize and accelerate the training process.

4 EXPERIMENTS

In this section, we evaluate the proposed NaLP method through link prediction. Then, overall relatedness analysis is conducted to have a better understanding of NaLP. And some important hyper-parameters are analyzed their impacts on the robustness of NaLP. Before elaborating these experiments, we will first introduce the datasets and metrics, as well as the baselines and experimental settings.

4.1 Datasets and Metrics

4.1.1 Datasets. We conduct our experiments on two datasets. The first one is the public n-ary relational dataset JF17K, which is derived from the popular KG Freebase [1]. All the facts in it are in good quality with the role sequences equal to the predefined standard ones and the corresponding values existing and known. This is usually not the case. And we build a relatively more practical dataset WikiPeople, which is also our second experimental dataset. It is constructed as follows:

- We download the Wikidata dump[4] and extract the facts concerning entities of type *human*.
- Then, these facts are further denoised. For example, facts containing element related to image are filtered out, and facts containing element in {*unknown value*, *no values*} are removed.
- Subsequently, we select the subsets of elements which have at least 30 mentions. The facts related to these elements are kept. Further, each fact is parsed into a set of its role-value pairs.
- The remaining facts are randomly split into training set, validation set and test set by a proportion of 80%:10%:10%.

Actually, WikiPeople is relatively more flexible, where data incompleteness, insert and update are universal. As an example, the role sequence of some n-ary relational facts like "*Marie Curie* received *Nobel Prize in Chemistry* in *1911*" is [*person*, *award*, *point in time*]. Then, the following similar facts in Wikidata[5] correspond to the three cases of data incompleteness, insert and update, respectively:

- *Marie Curie* received *Willard Gibbs Award*.
- *Marie Curie* received *Matteucci Medal* in *1904* with *Pierre Curie*.
  Marie Curie received *Nobel Prize in Physics* in *1903* together with *Henri Becquerel* and *Pierre Curie*.
- *Marie Curie* received *Davy Medal* with *Pierre Curie*.

[4] https://archive.org/details/wikibase-wikidatawiki-20171120
[5] https://www.wikidata.org/wiki/Q7186

In the first example, the value of the role *point in time* is unknown. In the second examples, one or two new roles *together with* are in need. In the third one, the value of the role *point in time* is unknown and a new role *together with* appears. Fortunately, our representation form of relational facts, which represents each relational fact as a set of its role-value pairs, is able to cope with all these cases, as exemplified in Section 3.1.

In real scenario, since there is possible missing in the process of knowledge extraction, and knowledge also grows and updates rapidly with many new contents flowing in, in the dataset derived from one snapshot, the above phenomena of data incompleteness, insert and update are inevitable. Thus, the developed WikiPeople dataset is practical. Note that, in this paper, no special handling is done to values from continuous domain (e.g., point in time, date of birth, date of death, etc.), and they are tackled like other values from discrete domain.

The detailed statistics of the two experimental datasets are displayed in Table 1, where $\#Relation$, $\#Train$, $\#Valid$ and $\#Test$ are the sizes of the relation set, the training set, the validation set and the test set, respectively. Here, $\#Relation$ is counted according to RAE [34] to have a better understanding of the two datasets.

4.1.2 Metrics. We adopt the standard Mean Reciprocal Rank (MRR) and Hits@N as metrics for comparison. These metrics are computed in the similar way of binary dataset [2]. For each test fact, one of its values/roles is removed and replaced by all the values/roles in V/R. These corrupted facts are fed into NaLP to obtain the evaluation scores. Then, these corrupted facts are sorted according to their evaluation scores in descending order. The rank of the test fact is finally stored. This whole procedure is repeated for all the other values/roles of the test fact. Thus, MRR is the average of these reciprocal ranks and Hits@N is the proportion of ranks which are less than or equal to N. In this paper, $N \in \{1, 3, 10\}$, i.e., the results in terms of Hits@1, Hits@3 and Hits@10 are reported. The traditional mean rank (the average of these ranks) is not adopted as a metric, since it is sensitive to outliers [22]. For these chosen metrics, the higher the value of MRR/Hits@N, the better the performance of the prediction.

In the test process, except the test fact, other corrupted facts that may also be valid, i.e., exist in the training/validation/test set, are discarded before sorting the facts.

4.2 Baselines and Experimental Settings

4.2.1 Baselines. Prior works for link prediction on n-ary relational data directly are scarce, and the state-of-the-art works are m-TransH [31] and its enhanced version RAE [34]. Since RAE performs better than its simplified version m-TransH, we only compare the proposed NaLP method with RAE.

4.2.2 Experimental Settings. For the proposed NaLP method, the reported results are given for the best set of hyper-parameters after grid search on the following values, by reference to ConvKB [19]: The embedding dimension $k \in \{50, 100\}$, the batch size $\beta \in \{128, 256\}$, the learning rate $\lambda \in \{5e^{-6}, 1e^{-5}, 5e^{-5}, 1e^{-4}, 5e^{-4}, 1e^{-3}\}$, the number of filters n_f of the convolution in {50, 100, 200, 400, 500}, the hyper-parameter n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$ in {50, 100, 200, 400, 500, 800, 1000, 1200}. The finally adopted optimal settings are: $k = 100$, $\beta = 128$, $\lambda = 5e^{-5}$, $n_f = 200$, $n_{\text{gFCN}} = 1000$ for JF17K, and $k = 100$, $\beta = 128$, $\lambda = 5e^{-5}$, $n_f = 200$, $n_{\text{gFCN}} = 1200$ for WikiPeople.

For the RAE method, as it does not report the ranges of hyper-parameters, we use the following ranges, by reference to its best settings on JF17K: The embedding dimension $k \in \{50, 100\}$, the margin ϵ of the first update step in $\{0.1, 0.5, 1.0, 2.0\}$, its weight hyper-parameter w of the regularization term and its threshold hyper-parameter δ in $\{1e^{-3}, 0.01, 0.1\}$, the batch sizes of the three update steps $\beta_1, \beta_2 = \beta_3 \in \{128, 256, 512, 1000\}$ (note that the second and third update steps share the same batch size), their learning rates $\lambda_1, \lambda_2, \lambda_3 \in \{5e^{-5}, 5e^{-4}, 5e^{-3}, 1e^{-3}, 1.5e^{-3}\}$. The finally adopted optimal settings are: $k = 100$, $w = 0.1$, $\delta = 1e^{-3}$, $\epsilon = 0.5$, $\beta_1 = 512$, $\beta_2 = \beta_3 = 128$, $\lambda_1 = 5e^{-3}$, $\lambda_2 = 1.5e^{-3}$, $\lambda_3 = 5e^{-4}$ for WikiPeople.

Note that, on WikiPeople, the best set of hyper-parameters are determined according to the evaluation results on the validation set in terms of MRR. And, on JF17K, since it lacks a validation set, we tune the hyper-parameters on the test set.

Particularly, we reproduce the experimental results reported in RAE on JF17K and compute more detailed results in terms of the above fine-grained metrics. Note that, before running NaLP on JF17K and RAE on WikiPeople, we need to transform the representation form of facts. It is effortless to transform each value sequence in JF17K to a set of role-value pairs, by adding the corresponding roles. To conduct the inverse transformation on WikiPeople, relations along with their role sequences are defined in advance, then only the value sequence of each fact is stored. Specifically, a new relation is defined when the aforementioned case of data incompleteness/insert/update appears.

4.3 Link Prediction

Link prediction on n-ary relational data has two tasks, i.e., value prediction and role prediction.

4.3.1 Value Prediction. In order to illustrate the performance of the proposed NaLP method on value prediction, we report the experimental results compared with RAE in terms of all the fine-grained metrics in Table 2.

From the results, it is clear that on both datasets, NaLP consistently outperforms RAE in terms of all the four metrics, which indicates the effectiveness and superiority of NaLP. Specifically, on the dataset in good quality, i.e., JF17K, NaLP improves the performance by 0.056 on MRR, 7.1% on Hits@1 and 5.7% on Hits@3. It testifies the strength of NaLP considering the relatedness of role-value pairs rather than only the relatedness of values in RAE. On the relatively more practical dataset, i.e., WikiPeople, the superiority of NaLP is significant. NaLP has considerable increase on all the four metrics, especially 0.166 on MRR, 17.0% on Hits@1 and 18.2% on Hits@3. It is not surprising, since NaLP is capable of better coping with diverse data than RAE. Data incompleteness/insert/update, which is ubiquitous on WikiPeople, is handled elegantly in NaLP, while new relation is defined in RAE, which may lead to data sparsity, when the instances of the new relation are in small number. Hence, it results in the relatively much worse performance of RAE on WikiPeople. Moreover, the overall better results of NaLP regarding more fine-grained metrics, i.e., MRR, Hits@1 and Hits@3,

Table 1: The statistics of the two datasets, JF17K and WikiPeople.

Dataset	$\|V\|$	$\#Relation$			$\#Train$			$\#Valid$			$\#Test$		
		Binary	N-ary	Overall	Binary	N-ary	Overall	Binary	N-ary	Overall	Binary	N-ary	Overall
JF17K	28,645	186	136	322	44,210	32,169	76,379	-	-	-	10,417	14,151	24,568
WikiPeople	47,765	150	557	707	270,179	35,546	305,725	33,845	4,378	38,223	33,890	4,391	38,281

Table 2: Experimental results of value prediction in terms of MRR and Hits@{1, 3, 10}.

Method	JF17K				WikiPeople			
	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
RAE	0.310	0.219	0.334	0.504	0.172	0.102	0.182	0.320
NaLP	**0.366**	**0.290**	**0.391**	**0.516**	**0.338**	**0.272**	**0.364**	**0.466**

Table 3: Detailed experimental results of value prediction on binary and n-ary relational facts in terms of MRR and Hits@{1, 3, 10}.

Method	JF17K								WikiPeople							
	MRR		Hits@1		Hits@3		Hits@10		MRR		Hits@1		Hits@3		Hits@10	
	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary
RAE	0.115	0.397	0.050	0.294	0.108	0.434	**0.247**	0.618	0.169	0.187	0.096	0.126	0.178	0.198	0.323	0.306
NaLP	**0.118**	**0.477**	**0.058**	**0.394**	**0.121**	**0.512**	0.246	**0.637**	**0.351**	**0.283**	**0.291**	**0.187**	**0.374**	**0.322**	**0.465**	**0.471**

Table 4: Experimental results of role prediction in terms of MRR and Hits@{1, 3, 10}.

Dataset	MRR			Hits@1			Hits@3			Hits@10		
	Binary	N-ary	Overall	Binary	N-ary	Overall	Binary	N-ary	Overall	Binary	N-ary	Overall
JF17K	0.811	0.831	0.825	0.738	0.773	0.762	0.872	0.874	0.873	0.928	0.927	0.927
WikiPeople	0.728	0.763	0.735	0.578	0.670	0.595	0.856	0.835	0.852	0.941	0.922	0.938

elucidate that NaLP is more effective in picking exactly valid values out.

To further elaborately evaluate the performance of NaLP on value prediction, we group the test set into binary and n-ary categories according to the arities of the facts. The detailed results on these two categories are presented in Table 3.

It can be seen from Table 3 that on the two datasets, NaLP demonstrates its advantage on both binary and n-ary categories. In more detail, on JF17K, the performance gap between NaLP and RAE is pronounced, especially in terms of MRR, Hits@1 and Hits@3. On WikiPeople, RAE is more significantly left behind by a large margin. These verify that NaLP is able to well deal with value prediction on both binary and n-ary categories. Notably, on JF17K, both RAE and NaLP perform much more poorly on binary category than on n-ary category. To analyze the reason behind, we dig into the dataset. It can be observed from Table 1 that JF17K has less n-ary relations. Although, the ratios between the sizes of n-ary and binary categories on the relation set ($136/186 = 0.731$) and the training set ($32,169/44,210 = 0.728$) are similar and less than 1, JF17K has much more test facts on n-ary category and the above ratio is greater than 1 ($14,151/10,417 = 1.358$). Since the test facts are used to tune the hyper-parameters (due to the lack of validation set), the relatively more facts on n-ary category in the test set encourage the models to do prediction on n-ary category much better. Thus, it is reasonable that both RAE and NaLP obtain worse performance on binary category than on n-ary category in JF17K.

4.3.2 Role Prediction. As there is no other method that conducts role prediction on n-ary relational data, and even RAE is deliberately designed only for value prediction, we conduct role prediction experiment only on the proposed NaLP method. Table 4 demonstrates the detailed experimental results in terms of all the four metrics on binary category, n-ary category and the whole dataset. The experimental results also illustrate the power of NaLP. NaLP achieves excellent results on all the metrics. We attribute this to the reasonable modeling of role-value pairs. Thus, it not only enhances the performance of value prediction, but also benefits role prediction.

4.4 Overall Relatedness Analysis

In the proposed NaLP method, the overall relatedness vector of a set of role-value pairs, i.e., a relational fact, is the crucial intermediate result. We conduct further analyses to dig deep into what it has learned.

According to Section 3.2, a valid relational fact is expected to have an overall relatedness vector of large value, while an invalid relational fact is on the contrary. And how to evaluate the degree

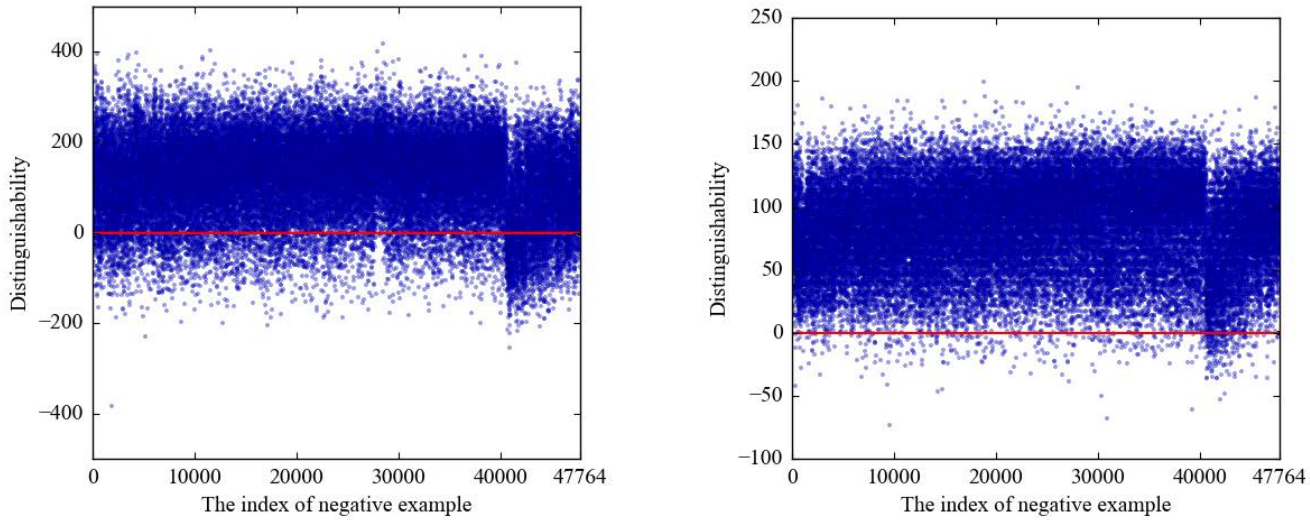


Figure 2: The visualization of the distinguishability results of Case-1 concerning the binary relational fact Fact-1: {*son* : *Michael Douglas*, *father* : *Kirk Douglas*} (left) and Case-2 concerning the n-ary relational fact Fact-2: {*person* : *Marie Curie*, *award* : *Nobel Prize in Physics*, *point in time* : *1903*, *together with* : *Henri Becquerel*, *together with* : *Pierre Curie*} (right).

of largeness? Actually, the relative magnitude between a valid relational fact and its negative samples is relatively more meaningful. Specifically, it is expected that a valid relational fact has larger values in a majority of dimensions of the overall relatedness vector compared to its negative samples, and thus the valid relational fact is distinguishable from its negative samples. Therefore, we propose the following metric to measure this type of distinguishability:

$$d(Rel^+, Rel^-) = \text{sum}\left(\text{sgn}(\mathcal{R}_{\text{Rel}^+} - \mathcal{R}_{\text{Rel}^-})\right) - \text{sum}\left(\text{sgn}(\mathcal{R}_{\text{Rel}^-} - \mathcal{R}_{\text{Rel}^+})\right), \tag{9}$$

where Rel^+ is a valid relational fact, and Rel^- is one of its negative samples; $\mathcal{R}_{\text{Rel}^+}$ and $\mathcal{R}_{\text{Rel}^-}$ are the overall relatedness vectors of Rel^+ and Rel^-, respectively; $\text{sgn}(x)$ is the function that returns 1, if $x > 0$, otherwise, returns 0; $\text{sum}(\cdot)$ is the element-wise sum function. In Equation (9), the left part of the minus sign counts the number of dimensions that $\mathcal{R}_{\text{Rel}^+}$ is larger than $\mathcal{R}_{\text{Rel}^-}$, and the right part is defined similarly. Hence, $d(Rel^+, Rel^-)$ measures the relative amount that $\mathcal{R}_{\text{Rel}^+}$ has more dimensions of larger values than $\mathcal{R}_{\text{Rel}^-}$.

Based on this proposed metric, we select two typical cases from the validation set, one binary case and one n-ary case, which are predicted correctly by NaLP, to carry out overall relatedness analysis. Note that, we only sample cases regarding value prediction, since role prediction is much simpler due to the much smaller role set. The sampled cases are as follows:

- Case-1: Predict *Michael Douglas*, given the role *son* and the remaining role-value pair of the binary relational fact {*son* : *Michael Douglas*, *father* : *Kirk Douglas*}, denoted as Fact-1.
- Case-2: Predict *Nobel Prize in Physics*, given the role *award* and the remaining role-value pairs of the n-ary relational fact {*person* : *Marie Curie*, *award* : *Nobel Prize in Physics*, *point in time* : *1903*, *together with* : *Henri Becquerel*, *together with* : *Pierre Curie*}, denoted as Fact-2.

Similar to the evaluation procedure, we replace *Michael Douglas* in Fact-1 and *Nobel Prize in Physics* in Fact-2 with all the values in V, then these corrupted facts are fed into NaLP to obtain the overall relatedness vectors. Subsequently, the distinguishability metric is computed following Equation (9). Figure 2 depicts these distinguishability results.

As exhibited in Figure 2, most distinguishability results lie in the area above 0. It visually corroborates that the overall relatedness vector of a fact really captures many discriminant features to further estimate the validity of the fact. Therefore, our conjecture on relatedness (see Section 3.2) makes sense to a certain degree. Due to the diversity of relational facts, the dimension of the overall relatedness vector is encouraged to be large. In this way, there is sufficient dimensions to encode various features. This is consistent with the large optimal value of n_{gFCN} (see Section 4.2.2). Specifically, in the left part of Figure 2, corresponding to the binary Case-1, the distinguishability metric of the negative sample indexed as 1844, corresponding to *Kirk Douglas* is extremely small. As the two role-value pairs *son* : *Kirk Douglas* and *father* : *Kirk Douglas* share the same part *Kirk Douglas*, there is no doubt that their relatedness is large, and the negative sample formed by this two role-vale pairs is easy to be judged as valid. NaLP is unable to filter out such situation. Actually, one value usually cannot correspond to two or more roles of the same fact. Thus, some rules may helpful to deal with the above deficiency. We leave it for future work. From the right part of Figure 2, corresponding to the n-ary Case-2, we have the following two observations:

- The negative samples indexed as 9556, 30913 and 39270, corresponding to *Marie Curie*, *Henri Becquerel* and *1903*, obtain the smallest, the second smallest and the third smallest

distinguishability results, respectively. These three values participate in some other roles of Fact-2, which is similar to the situation of *Kirk Douglas* in Case-1.

- Besides the negative samples corresponding to values of type *human*, some negative samples corresponding to values of type *time*, *club* or *institution* also achieve relatively smaller distinguishability results. It is acceptable, since NaLP does not take value type into consideration.

4.5 Hyper-parameter Analysis

To investigate the robustness of the proposed NaLP method, we further analyze the impacts of its important hyper-parameters, i.e., the number of filters n_f in the role-value pair embedding component and the hyper-parameter n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$ in the relatedness evaluation component. Without loss of generality, these experiments are all conducted on WikiPeople concerning value prediction.

4.5.1 The Impact of the Number of Filters n_f. To make the comparison fair, we keep all the hyper-parameters except n_f to be the optimal settings, and only vary the value of n_f in {50, 100, 200, 400, 500}, as mentioned in Section 4.2.2. To be clear, the evaluation results on the validation set with regard to only MRR are illustrated in the left part of Figure 3.

From the left part of Figure 3, it can be observed that the value of n_f affects the results indistinctively when the value lies in {100, 200, 400, 500}. Thus, NaLP is insensitive to the value of n_f, when the value is not too small, and is robust. The best performance with respect to MRR is obtained when n_f is set to 200.

4.5.2 The Impact of the Hyper-parameter n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$. Similarly, we vary the value of n_{gFCN} in {50, 100, 200, 400, 500, 800, 1000, 1200} and adopt all the other optimal hyper-parameter settings. We also demonstrate the evaluation results on the validation set in terms of only MRR in the right part of Figure 3.

In the right part of Figure 3, it turns out that different values of n_{gFCN}, no less than 400, demonstrate insignificant difference on performance and NaLP is thus robust. Among all these setting, $n_{\text{gFCN}} = 1200$ results in slightly better performance.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we represented each n-ary relational fact as a set of its role-value pairs and proposed a method to cope with Link Prediction on N-ary relational data, called NaLP. The design of the framework equips NaLP with the feature of permutation invariance to the input order of the role-value pairs, and the ability to well handle relational facts of different arities. By evaluating the pair-wise relatedness of all the role-value pairs in the same relational fact, NaLP is able to estimate their overall relatedness approximately. The resulting overall relatedness vectors further enable NaLP to pick up many determinant features to decide the validity of the input relational facts. Furthermore, since publicly available n-ary relational datasets are limited, we developed a practical one, WikiPeople, and it was published for further research. Experimental results on the public dataset and the newly developed WikiPeople manifest the merits and superiority of the proposed NaLP method. On value prediction, compared to the state-of-the-art method, NaLP improves all the metrics significantly, especially on WikiPeople. Specifically, NaLP improves the performance even by 7.1% in terms of Hits@1 on the public dataset, and has more than 17.0% increase in terms of Hits@1 and Hits@3 on WikiPeople.

For future work, on one hand, we will explore more expressive neural network models to capture more favorable features for validity evaluation. On the other hand, in this paper, we use only relational facts to conduct link prediction, and in the future we will introduce additional information, such as rules and value types, to further improve the developed model.

ACKNOWLEDGMENTS

The work is supported by the National Key Research and Development Program of China under grant 2016YFB1000902, the National Natural Science Foundation of China under grants 61772501, 61572473, 61572469, and 91646120, and the GFKJ Innovation Program.

REFERENCES

[1] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD'08)*. 1247–1250.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*. 2787–2795.

[3] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D knowledge graph embeddings. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI'18)*. 1811–1818.

[4] Boyang Ding, Quan Wang, Bin Wang, and Li Guo. 2018. Improving Knowledge Graph Embedding Using Simple Constraints. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL'18)*. 110–121.

[5] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'14)*. ACM, 601–610.

[6] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13rd International Conference on Artificial Intelligence and Statistics (AISTATS'10)*. 249–256.

[7] Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2018. Shared embedding based neural networks for knowledge graph completion. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM'18)*. 247–256.

[8] Nicholas Guttenberg, Nathaniel Virgo, Olaf Witkowski, Hidetoshi Aoki, and Ryota Kanai. 2016. Permutation-equivariant neural networks applied to dynamics prediction. *arXiv preprint arXiv:1612.04530* (2016).

[9] Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. 318–327.

[10] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML'15)*. 448–456.

[11] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL'15)*. 687–696.

[12] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge graph completion with adaptive sparse transfer matrix. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI'16)*. 985–991.

[13] Yantao Jia, Yuanzhuo Wang, Hailun Lin, Xiaolong Jin, and Xueqi Cheng. 2016. Locally adaptive translation for knowledge graph embedding. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI'16)*. 992–998.

[14] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling relation paths for representation learning of knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. 705–714.

[16] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*. 2181–2187.

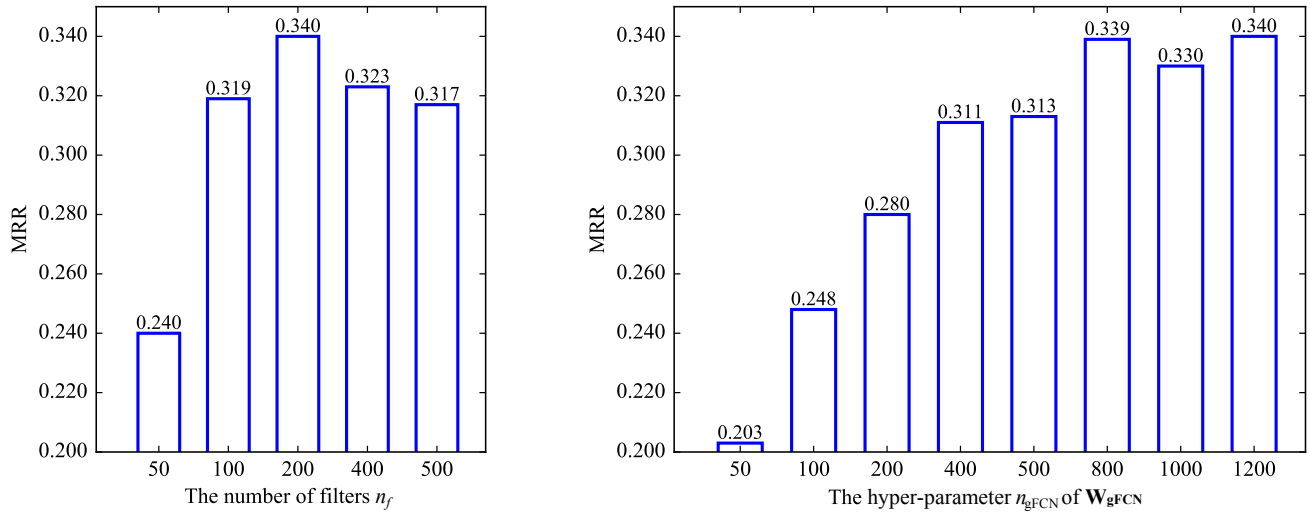


Figure 3: Performance comparison of the proposed NaLP method over different numbers of filters n_f (left) and different hyper-parameters n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$ (right) on WikiPeople concerning value prediction.

[17] Hitoshi Manabe, Katsuhiko Hayashi, and Masashi Shimbo. 2018. Data-dependent Learning of Symmetric/Antisymmetric Relations for Knowledge Base Completion. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI'18)*. 3754–3761.

[18] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML'10)*. 807–814.

[19] Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'18)*, Vol. 2. 327–333.

[20] Maximilian Nickel, Xueyan Jiang, and Volker Tresp. 2014. Reducing the rank in relational factorization models by including observable patterns. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14)*. 1179–1187.

[21] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2016), 11–33.

[22] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI'16)*. 1955–1961.

[23] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning (ICML'11)*. 809–816.

[24] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Tim Lillicrap. 2017. A simple neural network module for relational reasoning. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS'17)*. 4967–4976.

[25] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of the 15th Extended Semantic Web Conference (ESWC'18)*. 593–607.

[26] Baoxu Shi and Tim Weninger. 2017. ProjE: Embedding projection for knowledge graph completion. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI'17)*. 1236–1242.

[27] Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Falk Brauer. 2017. Random semantic tensor ensemble for scalable knowledge graph link prediction. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining (WSDM'17)*. 751–760.

[28] ThÃĩo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning (ICML'16)*, Vol. 48. 2071–2080.

[29] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.

[30] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI'14)*. 1112–1119.

[31] Jianfeng Wen, Jianxin Li, Yongyi Mao, Shini Chen, and Richong Zhang. 2016. On the representation and embedding of knowledge bases beyond binary relations. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. 1300–1307.

[32] Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. TransG: A generative model for knowledge graph embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL'16)*. 2316–2325.

[33] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR'15)*.

[34] Richong Zhang, Junpeng Li, Jiajie Mei, and Yongyi Mao. 2018. Scalable instance reconstruction in knowledge bases via relatedness affiliated embedding. In *Proceedings of the 27th International Conference on World Wide Web (WWW'18)*. 1185–1194.

PrivIdEx: Privacy Preserving and Secure Exchange of Digital Identity Assets.

H. Gunasinghe
Purdue University
huralali@purdue.edu

A. Kundu
IBM Research
akundu@us.ibm.com

E. Bertino
Purdue University
bertino@purdue.edu

H. Krawczyk
IBM Research
hugokraw@us.ibm.com

K. Singh
IBM Research
kapil@us.ibm.com

S. Chari*
IBM Research
snchari@gmail.com

D. Su*
IBM Research
sudong.tom@gmail.com

ABSTRACT

User's digital identity information has privacy and security requirements. Privacy requirements include *confidentiality* of the identity information itself, *anonymity* of those who verify and consume a user's identity information and *unlinkability* of online transactions which involve a user's identity. Security requirements include *correctness*, *ownership assurance* and *prevention of counterfeits* of a user's identity information. Such privacy and security requirements, although conflicting, are critical for identity management systems enabling the exchange of users' identity information between different parties during the execution of online transactions. Addressing all such requirements, without a centralized party managing the identity exchange transactions, raises several challenges. This paper presents a decentralized protocol for privacy preserving exchange of users' identity information addressing such challenges. The proposed protocol leverages advances in blockchain and zero knowledge proof technologies, as the main building blocks. We provide prototype implementations of the main building blocks of the protocol and assess its performance and security.

KEYWORDS

Decentralized identity asset exchange; Privacy preserving; Unlinkability; Counterfeit elimination; Blockchain; ZK-SNARK

ACM Reference Format:
H. Gunasinghe, A. Kundu, E. Bertino, H. Krawczyk, K. Singh, S. Chari*, and D. Su*. 2019. PrivIdEx: Privacy Preserving and Secure Exchange of Digital Identity Assets.. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313574

1 INTRODUCTION

Access to services offered by online service providers (SPs) is controlled by identity verification processes. Such a process requires users to verify different types of identity information, depending on the sensitivity of the service. Some services require to verify individual pieces of identity information, such as email address, phone number and Social Security Number (SSN), and some other services require to verify composite identity information such as driver's license and passport. Certain other services require to perform rigorous due diligence processes to satisfy certain compliance requirements such as Know Your Customer (KYC) compliance in banking/financial services [5]. Similar example scenarios which involve lengthy identity verification processes include, but not limited to, joining a new employer, applying for temporary visa in a foreign country, etc.

The amount of resources and effort that users and SPs have to devote to these identity verification processes vary depending on the type of identity information being verified. For examples, verifying an email address only requires the SP to send an email to the given email address, asking the user to click a link in it, verifying an SSN may require the SP to consume a paid service offered by the SSN authority, and verifying KYC compliance requires the SP to perform background checks on the user to verify the user's status of credit score, Anti-Money Laundering (AML), Counter Terrorist Financing (CTF), Politically Exposed Person (PEP) [5], etc. Once an SP has verified the identity of a user, the package of information associated with such *verified identity* becomes an asset of the SP, which we refer to as *identity asset* (the National Strategy for Trusted Identities in Cyberspace (NSTIC) labels such verified identity information as belonging to LOA (Level Of Assurance) 2+ category [20]). On the other hand, since an identity asset contains personal information about a user, the user also becomes an owner of the identity asset, leading to having two legitimate owners per *identity asset*.

Currently, when a user needs to consume similar services from different SPs, the user is treated as an "alien" by each SP and is required to go through a similar identity verification performed by each SP. These repeated processes not only are costly, but also are inherently error-prone, causing inconvenience to both parties. These issues magnify especially in scenarios involving lengthy identity verification such as consuming financial services from multiple banks, joining multiple employers, applying for temporary visa in multiple countries, etc. If there were a standard protocol through which different SPs could share the same *identity assets* of a user, that would result in substantial cost savings and notable convenience to both parties [5]. The SP who originally performed the identity verification for the user, and hence is one of the owners of the identity asset, can be incentivized for sharing the identity

*this work was done when the author was at IBM Research.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313574

asset, in exchange of a monetary compensation by the subsequent SP(s) that the user interacts with.

Let us consider the following **use case**: *The user Ursula first consumes financial services from bank A where bank A performs identity verification and due diligence steps for KYC compliance on Ursula. Later Ursula needs to consume financial services from bank B as well. Bank B wants to know if Ursula has already performed KYC compliance verification and if so, both Ursula and bank B would like to re-use the corresponding identity asset.* However, in this use case, Ursula would not like to reveal to bank B which bank(s) she has interacted before, and would not like to reveal to bank A, which other bank(s) she is planning to be a customer of. Bank A and bank B themselves would also not like to reveal their identity to each other during potential identity asset exchange, due to competition in business. On the other hand, Ursula would not like the transactions she carries out with different banks based on the same identity asset to be linkable. Prominent privacy protection regulations, such as General Data Protection Regulation (GDPR) [27], also treats a user's transactional patterns as personal data and prohibits tracking such personal data. Therefore, *anonymity of the parties who exchange the identity asset* and *unlinkability of the transactions* are key privacy requirements to be addressed when designing a protocol to facilitate identity asset exchange during online transactions, in addition to protecting *confidentiality* of identity assets from external parties.

The existing identity management protocols, which facilitate sharing users' identity information, do not address all the key privacy requirements. For examples, OpenIDConnect [22], an industry standard widely used by SPs to obtain a user's identity information from an identity provider (IDP), does not preserve anonymity and unlinkability. Two nation-scale brokered identification systems built by the USA and UK governments, namely, Federal Cloud Credential Exchange (FCCX) [21] and GOV.UK Verify [15], respectively, focus on the aforementioned use case and the first privacy requirement, i.e. protecting from each other the anonymity of the parties who exchange users' identity information, in order to preserve users' privacy. Those systems, however, use a government managed broker to mediate the identity exchange transactions, in which case, the identity of the two exchanging parties is revealed to the broker, although the two parties stay anonymous to each other. Brandao et. al [7] have raised certain other privacy concerns on the introduction of such a centralized broker.

One of our goals is to avoid introducing such a centralized broker. Therefore, the proposed protocol is executed in a decentralized identity management ecosystem backed by a permissioned blockchain network (see Section 2.1). Distributed trust implemented on the basis of the consensus protocol through which blockchain peers validate protocol executions eliminates the requirement of a centralized broker. Participants of the decentralized identity management ecosystem invoke the identity exchange transactions with pseudonyms, in order to preserve anonymity and unlinkability. However, when anonymity and unlinkability are enforced in confidential identity asset exchanges, without a mediating centralized party, it is challenging to achieve the required security properties, such as *correctness*, *ownership assurance* and *counterfeits elimination* of the identity assets, and optionally, *financial fairness* of the identity asset exchange transactions, because the participants, appearing with a new pseudonym in each round of the protocol execution, can violate such security properties, as discussed in Section 3. This problem can be related to the challenge of preventing double spending in bitcoin [19] and ZeroCash [4], without a centralized financial institute managing the payment transactions. However, unlike in bitcoin and ZeroCash, whose goal is to prevent double spending of cryptocurrency, which has a single owner at a time, our goal is to enable multiple exchanges of the same identity asset which has two owners, which poses a different set of challenges. We have designed the dedicated phases of the protocol to address such challenges leveraging the power of ZK-SNARKS (see Section 2.2).

Our main contribution is PrivIdEx - a protocol realizing privacy preserving and secure exchanges of identity assets in a decentralized identity management ecosystem, including its: i) design, ii) implementation and iii) analysis and evaluation.

2 PRELIMINARIES

2.1 Permissioned Blockchain

We identify two main parties in a BC [25] network as follows: i) *peers* - they maintain the transaction ledger (i.e. BC) and host the smart contract(s); ii) *participants* - they perform transactions. The consensus algorithm defines rules to be followed by peers when ordering and validating the transactions to be added to the ledger. A *smart contract* defines the business logic for transaction execution and validation, which is invoked by participants and run independently by each peer for executing the transactions.

There are two types of BCs: i) Permissionless BCs - where any one can join the network and write to/read from the BC. Participants' identity is hidden by the random pseudonyms they choose, which results in lack of accountability. Cheating by the peers is avoided and the correctness of the ledger is preserved by employing an expensive consensus algorithm and the assumption that the majority of the computation power of the network is with honest parties. ii) Permissioned BCs - where a trusted certification authority (CA) issues signed X.509 certificates to actors (i.e. peers and participants), which include the public key of a RSA key pair and other identity attributes that determine their permissions (i.e. read/write access to BC). This preserves accountability and enables employing a less costly consensus algorithm. Permissioned BCs are categorized as public and private based on whether read access is controlled or not, respectively. We assume a decentralized identity management ecosystem backed by a permissioned BC when designing the proposed protocol, in particular, Hyperledger Fabric [12], a private permissioned BC.

2.2 ZK-SNARKs

ZK-SANRKs is an efficient construction to prove in zero-knowledge, a satisfying assignment to the class of problems called Quadratic Span Program (QSP) [14]. QSP is an NP-complete problem. According to the principles in complexity theory, for any NP problem L and an NP-complete problem M, there is a reduction function f, which is computable in polynomial time, s.t. $L(x) = M(f(x))$. Accordingly, ZK-SANRKs can be used to prove in zero-knowledge, a satisfying assignment to any NP problem following these high level steps: i) formulate the decision problem D as an NP statement, which is expressed in the following form: Given a set of public inputs X, I know a set of secret inputs W, s.t. condition D holds on X and W

(i.e. the satisfying assignment is constituted by X and W); ii) write an algorithmic program P to solve D; iii) convert P to an arithmetic circuit C; iv) convert C to a QAP (Quadratic Arithmetic Program - a variant of QSP); v) prove/verify satisfiability for the QAP in zero knowledge.

The ZK-SNARKs construction is expressed in following three algorithms: ***i) Generator (G)***: takes as inputs: C and secret parameters λ, and outputs a proving key (pk) and a verification key (vk). This is a one time setup step run by a trusted party, after which λ should be destroyed in order to preserve the soundness of the proofs. (pk, vk) := G(C, λ). ***ii) Prover (P)***: takes as inputs: C, pk and the satisfying assignment - which may have both private inputs w and public inputs x, and outputs the proof Φ. Φ := P (C, pk, w, x). ***iii) Verifier (V)***: takes as inputs: vk, Φ and public inputs x provided by the prover, and outputs the decision d as true, iff (w, x) is a satisfying assignment to P, and false, otherwise. d := V(vk, Φ, x).

In ZK-SNARK (*Zero-Knowledge Succinct Non-interactive Argument of Knowledge*), the *Zero-Knowledge* property enables participants to keep transaction information confidential, and still prove to peers that transactions are valid according to the smart contract. *Succinctness* makes the size of such proofs small (≈2KB) and verification time in the orders of milliseconds, irrespective of the complexity of the business logic defined in the smart contract. The *Non-interactive* property enables multiple peers to verify the proofs independently without interacting with the prover. There are other zero-knowledge proof constructions developed to achieve similar goals without a trusted setup, such as ZK-STARK [3] and Bullet Proof [8]. We use ZK-SNARK since it is more efficient and practical compared to the other constructions.

3 SYSTEM MODEL AND THREATS

3.1 System model

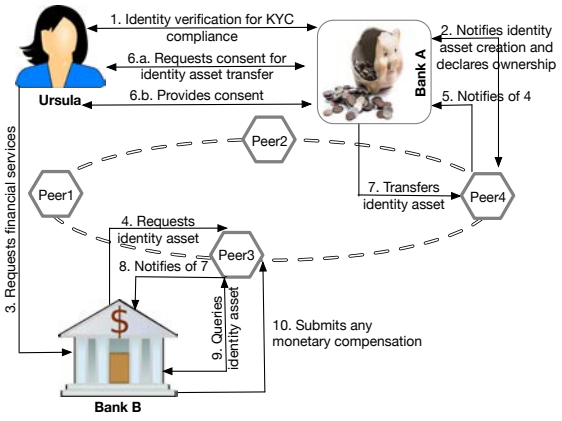


Figure 1: Plain protocol model for decentralized identity asset exchange, with no privacy features. This is the version 0 (V0) in the incremental development of a privacy preserving identity exchange protocol.

The following steps describe the basic flow of identity asset exchange for the use case mentioned in Section 1, which is illustrated in Figure 1: 1) When Ursula consumes financial services for the first time from bank A, identity verification and due diligence are performed by bank A to verify KYC compliance of Ursula, and the resulting identity asset is stored at bank A. Note that the details of how the identity verification is performed is out of scope of the identity exchange protocol. 2) Bank A notifies the identity ecosystem about the identity asset creation and claims its ownership. 3) At a later point in time, Ursula requests financial services from bank B and they discover by some means (either by querying the BC or using Ursula's private records, which we will discuss in Section 4) if the required identity asset is already created for Ursula. 4) If this is the case, Bank B requests from the identity ecosystem the relevant identity asset of Ursula. 5) Bank A receives the request submitted by bank B, via the transaction notification system of the BC. 6.a) If bank A decides to share the identity asset, bank A requests the consent from Ursula to transfer the identity asset to bank B. 6.b) Ursula provides her consent. 7) Bank A transfers the identity asset, along with Ursula's consent, via the identity ecosystem. 8) Bank B receives the notification about the valid identity asset transfer. 9) Bank B queries the identity asset from the ledger. 10) Optionally, if the transferred identity asset is correct, bank B submits a monetary compensation (if it is required, by the policies of the identity ecosystem) to bank A. This could be a bitcoin payment sent to bank A, if the underlying BC supports bitcoin transactions, which bank A can redeem later.

We make the following **three basic assumptions** in the context of decentralized identity asset exchange: 1) There is a criteria to define and verify uniqueness of an identity asset so that if multiple copies of a particular type of identity asset are created by multiple parties using identity information of a given user, they all become digitally identical (e.g. defining a standard format for an identity asset used for a particular identity verification scenario and considering the cryptographic hash (CRH) of the identity asset to be the criteria for verifying uniqueness). 2) If a particular type of identity asset is created for a given user in the identity ecosystem, all the SPs requiring a similar identity asset from the user should re-use it, without re-creating it. 3) The trusted CA, which issues the identity certificates to the actors of the BC network, does not collude with any actor in the BC.

Note that the plain protocol model for identity asset exchange shown in Figure 1 does not include any privacy features, i.e. all the participants appear in their real identities and identity assets are transferred in plain text. Although such a model is not used in a real world deployment, we use it as the baseline (version 0 - V0) protocol to identify what properties should be achieved in order to guarantee that an identity asset exchange protocol is secure, and to analyze the challenges in achieving the same security properties when privacy features are added to the protocol incrementally.

3.2 Threat model for protocol security

In what follows, we first identify the different ways in which an adversary can compromise the security of the plain protocol model. **1) Compromising *correctness***: A malicious identity asset provider (e.g. bank A) transfers an identity asset in step 6, which is different than the one it notified the creation of, in step 2. **2) Compromising *ownership assurance***: A malicious collusion of two of the three parties engaged in an identity asset exchange can compromise the

ownership assurance of one of the two legitimate owners as follows: i) a different user can collude with bank A to impersonate Ursula at bank B; ii) bank A and bank B can collude to transfer the identity asset without Ursula's consent; iii) after bank B obtains the identity asset from bank A, Ursula and bank B can collude to act as the original owners and transfer the identity asset to a different bank. Attacks i) and ii) are possible by sending a fake consent in step 6.b and attack iii) is motivated by any benefits obtained from the identity asset exchange, such as monetary compensation paid by the identity asset consumer to the identity asset provider. **3) Creating *counterfeits*** of an identity asset: Due to the same motivation for attack 2.iii, bank B may execute step 2 using the identity asset received from bank A, hence creating a counterfeit. **4) Compromising *financial fairness***: After receiving the identity asset, bank B can skip step 10.

Accordingly, *correctness*, *ownership assurance*, *counterfeits elimination* and *financial fairness* are key requirements to be addressed, in order to guarantee the security of an identity asset exchange protocol. In what follows, we describe the simple mechanisms that should be incorporated into protocol V0, in order to address those requirements. **1) *Correctness*** : The smart contract that defines the protocol requires from bank A to submit the cryptographic hash (CRH) of the identity asset in step 2. In step 7, bank A submits a pointer (e.g. transaction ID) to step 2 associated with the identity asset being transferred. Then the peers running the smart contract validate correctness by computing the CRH of the transferred identity asset and comparing it with the CRH submitted in step 2 associated with the same identity asset. **2) *Ownership assurance***: The protocol requires: i) from bank A to submit the CRH of the public keys of the two owners in step 2; ii) from Ursula and bank A to sign, using their private keys, the messages sent in step 6.b (consent by the user) and step 7 (identity asset transfer), respectively. Then the peers verify the signatures and confirm that the original owners of the identity asset indeed performed the transfer. Note that an adversary can replay the message sent in step 7. Therefore, the protocol should also require bank B to send a random nonce in the identity asset request message (step 4), which Ursula and bank A should include in the messages they sign in steps 6.b and 7. **3) *Counterfeits elimination***: To ensure that duplicates of an identity asset do not exist, the peers maintain a hash table storing the information submitted in step 2 of the protocol, indexed by the CRH of the identity asset. Each time step 2 is executed for a newly created identity asset, the peers check if the newly submitted CRH already exists in the hash table, in which case the peers reject it as an attempt to create a counterfeit of an identity asset. **4)*Financial fairness*** : It is unlikely that bank B skips step 10 in protocol V0 where it appears with its real identity, as it would damage its reputation. Even if bank B does so, it is easy to take actions against bank B for the dishonest behavior.

3.3 Threat model for users' privacy

In what follows we discuss an adversary's goals in compromising users' privacy (i.e. learning and tracking information that users do not intentionally share) in protocol V0. **1) Compromising *confidentiality* of users' identity information**: The adversary learns the users' identity information from the identity assets transferred in plain text via the BC in executions of step 7. **2) Compromising users' *transactional privacy***: i) the adversary learns the identity of the parties a user interacts with, because in protocol V0, all participants interact with the identity ecosystem using their real identities; ii) the adversary tracks a user's transaction patterns by linking the transactions that the user carries out with different SPs.

The following modifications to protocol V0 address those privacy concerns. **1) *Confidentiality***: Bank A encrypts the identity asset in step 7, using a key known to bank B. **2) *Anonymity*** of the parties whom a user interacts with: All the participants use a pseudonymous certificate issued by the CA, when interacting with the identity ecosystem. **3) *Unlinkability*** of the user's transactions: i) the participants use different pseudonymous certificates in executing step 2 and each round of identity asset transfer (i.e. steps 4 - 10); ii) bank A does not expose the CRH of the identity asset in plain text in step 2; instead it submits a commitment to the CRH of the identity asset. Otherwise, the identity asset consumers (e.g. bank B), can decrypt the identity asset received in step 9, compute its CRH and track the corresponding identity asset creation transaction (e.g. execution of step 2), in order to infer information such as when is the first time the user has consumed a similar service, etc. Note that in the proposed modifications 2 and 3 above, the participants obtain pseudonymous identity certificates from the CA, which do not include any identifiable attributes, but the public key of a new RSA key pair.

3.4 Challenges in preserving users' privacy and ensuring security of the protocol

In what follows we discuss how the aforementioned mechanisms for ensuring security and privacy properties conflict, which raise challenges in developing a privacy preserving and secure identity asset exchange protocol. We discuss such challenges w.r.t. three versions of the incrementally developed protocol, each of which is a result of incorporating privacy features one by one, into V0.

V1-*Confidentiality*: When the identity asset is encrypted, peers cannot verify its *correctness* simply by computing its CRH as in V0. The mechanisms for preserving *ownership assurance*, *counterfeit elimination* and *financial fairness* used in V0 are not affected though.

V2-*Confidentiality and Anonymity*: When the participants execute the protocol with pseudonyms, the *correctness* enforcement mechanism is not affected compared to V1. However, the mechanism to preserve *ownership assurance* in V0/V1 is affected, because there is the threat of a malicious identity consumer (e.g. bank B) sending a 'contract' which they want the identity asset provider and/or the user to sign, instead of a truly random nonce in step 4. Therefore, parties can not give away a signature on a challenge nonce, as a proof of ownership of the private key. There is no effect on the *counterfeit elimination* mechanism used in V0/V1, except that the owners of the identity asset should use the key pair related to their pseudonymous identity in steps 2, 6.b and 7. The *financial fairness* enforcement mechanism used in V0/V1 is affected, because pseudonymous bank B can intentionally skip step 10. Involving the CA to de-anonymize such identity consumers adds lot of overhead. Instead, this should be addressed by the protocol itself.

V3-*Confidentiality, Anonymity and Unlinkability*: It is more challenging to ensure *correctness* when unlinkability is enforced,

because now even the CRH of the identity asset is not exposed in step 2. It is also more challenging to preserve *ownership assurance* than in V2, because now the two owners of the identity asset use different pseudonyms in step 2 and in each round of steps 4-10, in contrast to using a single pseudonym across all transactions as in V2. The hash table based *counterfeit elimination* mechanism is no longer sufficient now, because the hash of the identity asset is not exposed in step 2 and any two parties appearing with new pseudonyms can execute step 2, submitting a commitment to a CRH of any identity asset. Preserving *financial fairness* also needs improved mechanisms because we need to make sure that the underlying monetary payment system also preserves unlinkability; otherwise, parties can be de-anonymized via linkability in the payment system.

Table 1 summarizes how introducing the properties for preserving users' privacy into protocol V0, in an incremental manner, affects the mechanisms for achieving the identified security properties. A checkmark in a given cell indicates that the combination of privacy properties in the given column pose challenges to the mechanism used in the previous version of the protocol, for ensuring the security property in the given row.

Table 1: Effect of introducing the properties for preserving users' privacy, in an incremental manner, on the properties for ensuring security of the protocol.

Properties to ensure the protocol is secure:	Properties to ensure users' privacy:		
	confidentiality	confidentiality + anonymity	confidentiality + anonymity + unlinkability
counterfeit elimination	-	-	✓
correctness	✓	-	✓
ownership assurance	-	✓	✓
financial fairness	-	✓	✓

4 PROTOCOL DESIGN

In what follows, we present the design of the proposed protocol, named PrivIdEx, addressing the aforementioned challenges. As shown in Figure 2, PrivIdEx involves four parties: Identity Asset Provider (IAP), User, Identity Asset Consumer (IAC), and Blockchain (BC). PrivIdEx consists of four phases, each of which serves one or more specific purposes and groups together a set of relevant steps from the identity asset exchange flow shown in Figure 1. Phase 0 is executed only once and phases 1-3 are executed each time an identity asset is exchanged. Note that T represents the transactions posted to the BC by participants, by invoking different functions in the smart contract, W represents the validation steps executed by peers on the transactions and M represents the messages exchanged between two parties offline (i.e. without involvement of the BC). The purpose(s) of each phase is (are) described as follows, with examples from protocol V0.

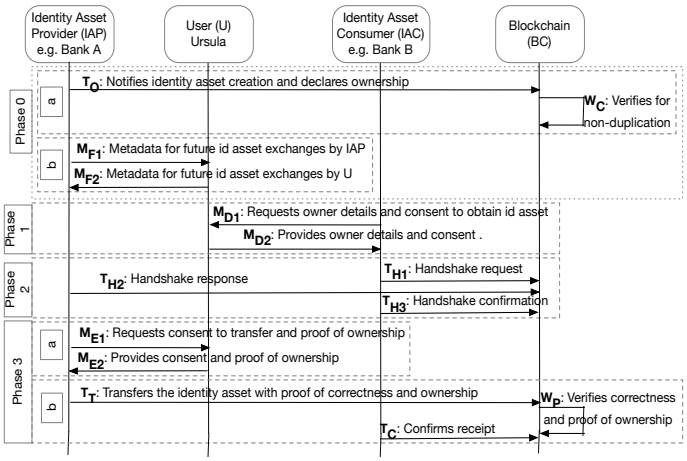


Figure 2: Overview of PrivIdEx design.

Phase 0 consists of two sub phases. Phase 0.a, executed between the IAP and the BC, serves for ownership declaration (T_O) by IAPs for newly created identity assets and verification by peers that such identity assets are not counterfeits. E.g. in protocol V0, the IAP sends the CRH of the public keys of the two owners and the CRH of the identity asset in T_O, which the peers verify in W_C using the hash table of ownership declarations maintained in the BC (see Section 3.2). Phase 0.b, executed between the IAP and the user, allows them to exchange meta data to be used in future identity asset exchange(s).

Phase 1 is executed between the IAC and the user. In M_{D1}, the IAC requests certain details of the IAP that has created the required identity asset (if any), and the user's consent for requesting such identity asset from the corresponding IAP. The meta data saved by the user in phase 0.b is used to construct the user's response (M_{D2}).

Phase 2 consists of a three-way handshake between the IAC and the IAP over the decentralized identity ecosystem. The IAC initiates the handshake by submitting to the BC a message (T_{H1}) addressed to the IAP who owns the identity asset. T_{H1} includes the user's consent received in M_{D2} and is signed by the IAC. All the participants receive a notification about T_{H1} via the BC. The corresponding IAP verifies the user's consent and submits a response handshake message (T_{H2}). The IAC acknowledges T_{H2} by sending a confirmation (T_{H3}). The handshake phase allows the IAP and the IAC to negotiate certain information pertaining to the identity asset exchange, e.g. in protocol V0, the IAC sends the random nonce to be signed by the two owners during the transfer phase, for the proof of ownership. The peers validate a handshake by verifying if T_{H2} and T_{H3} are associated with a corresponding T_{H1} and T_{H2}, respectively.

Phase 3 is where the actual identity transfer happens. In phase 3.a, executed between the IAP and the user, the IAP requests the user, via M_{E1}, to provide consent for transferring the identity asset and the proof of user's ownership of the identity asset. The user responds accordingly via M_{E2}. In phase 3.b, the IAP transfers the identity asset along with M_{E2} and the proof of its ownership of the identity asset, via T_T. The peers verify in W_P, that T_T is associated

with a valid handshake, and verify correctness and proofs of ownership. After receiving the notification about T_T from the BC, the IAC checks the transferred identity asset and posts a confirmation to the BC about the successful receipt of the identity asset in T_C.

In what follows, we present how those different phases are utilized and enhanced to address the challenges for achieving the security properties of the three versions of the incremental design of PrivIdEx (see Section 3.4).

4.1 V1- Confidentiality Preserving Protocol

When confidentiality is enforced, the IAP and the IAC agree on a key K for encrypting the identity asset (A), by integrating the Diffie-Hellman key exchange protocol into the three-way handshake in phase 2. In T_T, the IAP submits the encrypted identity asset: $C = Enc_K(A)$. Due to encryption of the identity asset, *correctness* is the only security property that is more challenging to achieve in V1, compared to V0 (see Section 3.4). To prove correctness, the IAP submits in T_T, the transaction id of T_O associated with A and a zero-knowledge (ZK) proof (Φ_1), proving the knowledge of a satisfying assignment to the NP statement - **NS1**: *Given a cipher text C', a hash value a', I know the following secrets: an identity asset A' and a key K' s.t $a' = CRH(A')$ and $C' = Enc_{K'}(A')$*. The IAP proves the satisfying assignment to NS1 with $A' = A$ and $K' = K$ as secret inputs, $C' = C$, and a' = the CRH of A submitted in the corresponding T_O, as the public inputs. If Φ_1 is verified successfully, the peers accept that T_T encrypts the same identity asset whose ownership was declared in the corresponding T_O.

4.2 V2 - Confidentiality and Anonymity Preserving Protocol

When anonymity is enforced: i) the IAP includes the two owners' pseudonyms in T_O instead of their real identities; ii) the IAP and the user records each other's pseudonym in phase 0.b; iii) the user sends to the IAC, via M_{D2}, the pseudonym of the IAP who created the identity asset; iv) the transactions posted to the BC are signed using the pseudonyms; v) when transaction notifications are received from the BC, each participant checks if the transaction messages are addressed to their pseudonym and responds accordingly. Due to the threats by the pseudonymous participants, *ownership assurance* and *financial fairness* are more challenging to achieve in V2 (see Section 3.4).

To prove ownership in V2, the user and the IAP create ZK proofs (Φ_{2U} and Φ_{2P} respectively) on the NP statement **NS2**: *Given a public key PK, a message M, I know a secret signature S, s.t. RSA_Sig_Verify(M, S, PK) = True*, where S = RSA_Sig(M, private key of PK). RSA_Sig outputs a signature, given a RSA private key, and a message M; RSA_Sig_Verify outputs True iff S is the correct signature for M, using the private key associated with PK. The user and the IAP prove satisfying assignments to NS2 with PK = the public key of the pseudonym included in the T_O associated with the identity asset being transferred, M = the random nonce sent by the IAC via T_{H1}, as public inputs, and S = the signature created on such nonce with the private key associated with the pseudonym, as the secret input. Proving the knowledge of S on the nonce sent by the IAC, without revealing S, avoids the risk of giving a signature on a potential 'contract' (see Section 3.4).

Φ_{2U} and Φ_{2P} are integrated in to the protocol design as follows. The IAP sends to the user, via M_{E1}, the nonce it received from the IAC in T_{H1}. The user sends Φ_{2U} via M_{E2}. Then the IAP creates Φ_1 (see Section 4.1) and Φ_{2P}, and sends T_T to the BC along with the transaction id of T_O associated with A, Φ_1, Φ_{2U} and Φ_{2P}. The peers validate T_T, by verifying correctness and ownership assurance via the ZK-proofs provided in T_T.

To ensure *financial fairness* in V2, the handshake phase is used as follows. The IAP informs the IAC about the required monetary compensation for transferring the identity asset via T_{H2},. If the IAC agrees to pay, it includes in T_{H3} a reference to a bitcoin payment made with a locking condition such as: *'the IAP can can unlock the payment only by using either of these: i) a T_C submitted by the IAC, indicating a successful receipt of the identity asset; ii) a successful W_P by the peers, if a T_C is not submitted after time 't' since the time of T_T'*. Such a locked payment [2, 24] made during the handshake phase prevents a pseudonymous IAC from skipping the required payment. The IAP can redeem the payment only if it transfers the correct identity asset, ensuring financial fairness to both parties.

4.3 V3 - Confidentiality, Anonymity and Unlinkability Preserving Protocol

As per Section 3.4, it is more challenging to achieve all four security properties in V3, compared to V2. In what follows we describe how each phase of the protocol is enhanced to address those challenges. Prior to phase 0.a, the user creates a commitment to the public key of her real identity (U_{pk}): $C_U = commit(U_{PK}, r_u)$ and sends C_U to the IAP. The IAP creates a commitment to the public key of its real identity (P_{pk}): $C_P = commit(P_{PK}, r_p)$ and a commitment to a - the CRH of the newly created identity asset: $C_a = commit(a, r_a)$. The IAP sends T_O to the BC, including the ownership declaration $O = C_U|C_P|C_a$, where | denotes concatenation, signed by a new pseudonym key.

The basic idea of *counterfeit elimination* in V3 is as follows. Let $B = \{a_1, a_2, ..., a_n\}$ be the set of CRH values of the identity assets associated with all the previous valid executions of T_O. This set is represented by a unique polynomial P of degree n, that has a_1, a_2,..., a_n as its roots. The polynomial P is represented as $P(x) = \prod_{i=1}^{n}(x - a_i)$. Let P_i be the i^{th} coefficient of P, for $i = 0, 1, ..., n$. P is initialized as $P(x) = 1$, and its degree increases with each new valid T_O. Hence, at any given time, if a number n of valid identity assets have been created in the identity ecosystem, then there is a number $n + 1$ of P_is. If the evaluation of $P(x)$ with $x = a$ results in zero (i.e. $P(a) = \sum_{i=0}^{n} P_i.a^i = 0$), it implies that the identity asset, whose CRH value is a, is a duplicate of an existing one. In order to preserve unlinkability, peers should only learn if $P(a) = 0$ or not, and nothing else. Therefore, the set of P_is are secretly encoded before being stored in the BC and $P(a)$ is computed in the encoded domain before being revealed to peers.

To prove that the created identity asset is not a counterfeit, the IAP submits T_O to the BC, including O and the following four items:
(I1) The result of computing $P(a)$ in the encoded domain, which is denoted by l, i.e. $l = En(P(a))$.
(I2) A ZK-proof Φ_3, proving that l is correctly computed and that the same a is used to compute both C_a (in O) and l.

(I3) The secretly encoded set of coefficients P'_i of the updated polynomial P', which has a as one of its roots (i.e. $P'(x) = P(x).(x - a)$, and therefore, P'_0 = -$a.P_0$, $P'_i = P_{i-1}$ - $a.P_i$, for $i = 1, ..., n$, and $P'_{n+1} = 1$).

(I4) A ZK-proof Φ_4, proving that I3 is correctly computed, using the same a used in C_a.

Details of the mechanism for counterfeit elimination are as follows. Let E be an additive threshold homomorphic encryption scheme, whose public key is known to everyone, but the secret key is distributed among the peers s.t. a group of at least $t + 1$ of them are required to perform decryption. E is instantiated with the Elgamal encryption scheme over a group G of order q. The public key $h = g^s$, where g a generator in G and s is the private key, which is distributed among the threshold peers. E is initialized with a distributed key generation protocol [13]. The encryption of an element $g' \in G$ is defined as: $E(g') = (g^k, h^k g')$, where $k \in Z_q$ is randomly chosen. An encoding scheme En to encode elements in Z_q is defined based on E as follows. $En(z) = (g^k, h^k g^z)$, where $z \in Z_q$. In fact, $En(z) = E(g^z)$. $En(\cdot)$ is an additively homomorphic encoding of z which allows us to carry out computation on polynomials whose coefficients are presented in encoded form. Moreover, while the value z cannot be recovered in general from $En(z)$, for our purposes we only need to be able to decide for a given $En(z)$ whether z is zero or not. In addition, the secrecy of z is guaranteed by the underlying Elgamal encryption scheme $E(\cdot)$.

Let the set of encoded P_is stored in the BC be $S_n = \{En(P_i) = E(g^{P_i}),$ for $i = 0, 1, ..., n\}$. Details of how the IAP computes I1-I4 are as follows.

(I1) Compute $l = En(P(a))$, given the set S_n:

- Compute a fresh encoding of zero as: $e_0 = En(0) = (g^k, h^k)$ for random $k \in Z_q$, in order to randomize the encoding of l.
- Then l is computed as follows:

$$l = (\prod_{i=0}^{n} (En(P_i)^{a^i})).e_0 \quad (1)$$

(I2) Create Φ_3 on the NP statement - **NS3**: *Given a commitment C_{IA}, an encoding L, and a set S of encoded coefficients of a polynomial P, I know secrets: r, a′, and k′ s.t. $C_{IA} = commit(a', r)$, $L = En(P(a') + 0) = En(P(a')).e_0$ and $e_0 = (g^{k'}, h^{k'})$.*

The IAP proves a satisfying assignment to NS3 with $C_{IA} = C_a$ in O, $L = l$ in (I1), and $S = S_n$ stored in the BC at the time of submitting T_O, as public inputs; and $r = r_a$ and $a' = a$ used in C_a and $k' = k$ used in e_0, as secret inputs.

(I3) Let the set of P'_is in the encoded domain be $S_{n+1} = \{En(P'_i)$ for $i = 0, 1, ..., n + 1\}$. S_{n+1} is computed as follows.

- Compute a fresh encoding of zero as: $e_0 = En(0) = (g^{k_0}, h^{k_0})$ for random $k_0 \in Z_q$, and $En(P'_0) = En((-a).P_0 + 0) = En(P_0)^{-a}.e_0$.
- For $i = \{1, .., n\}$, choose $k_i \in Z_q$ randomly and compute: $e_0^i = En(0)$, and $En(P'_i) = En(P_{i-1} + (-a).P_i + 0) = En(P_{i-1}).En(P_i)^{-a}.e_0^i$.
- Choose $k_1 \in Z_q$ randomly and compute $En(P_{n+1}) = En(1)$.

(I4) Create Φ_4 on the NP statement - **NS4**: *Given a commitment C_{IA}, a set S of encoded coefficients of a polynomial P^n and a set S' of the updated polynomial P^{n+1}, I know secrets: r, a′, k'_0, k'_1 and k'_i for $i = 1, ..., n$ s.t.:*

- *$C_{IA} = commit(a', r)$,*
- *$En(P_0^{n+1}) = En(-a'.P_0^n + 0) = En(P_0^n)^{-a'}.e_0$ and $e_0 = (g^{k'_0}, h^{k'_0})$,*
- *for $i = 1, .., n$, $En(P_i^{n+1}) = En(P_{i-1}^n).En(P_i^n)^{-a'}.e_0^i$ and $e_0^i = (g^{k'_i}, h^{k'_i})$,*
- *$En(P_{n+1}^{n+1}) = (g^{k'_1}, h^{k'_1}.g^1)$.*

The IAP proves a satisfying assignment to NS4 with $C_{IA} = C_a$, $S' = S_{n+1}$ computed in I3 above, $S = S_n$ stored in the BC, as public inputs, and with $a' = a$, $r = r_a$ used in C_a, $k'_0 = k_0$ used in $En(P'_0)$, $k'_1 = k_1$ used in $En(P'_{n+1})$ and $k'_i = k_i$ used in $En(P'_i)$ for $i = 1, .., n$, of I3, as secret inputs.

Once the IAP submits T_0 along with O and I1-I4, the validation W_C (see Figure 2) by peers is executed as follows. If the ZK-proof Φ_3 is successfully verified, each peer i executes the following steps to randomize l: choose $r_i \in Z_q$ randomly, compute $l_i = l^{r_i}$ and broadcast l_i to all the other peers together with a ZK-proof Ω_1^i, proving that the peer knows the value r_i. Then each peer computes l' as the sum of the cipher texts received from all the peers, i.e. $l' = \prod_{i=1}^{m} (l^{r_i}) = E(g^{P(a)(\sum_{i=1}^{m}(r_i))})$, where m is the number of peers. Then peers collectively decode l'. Note that we choose to randomize l and then decode l', instead of just decoding l, due to a potential collusion attack by an IAP and a peer to check if a given value matches the CRH of an already created identity asset. We refer the reader to the technical report [16] for the details of such attack and the definition of the NP statement associated with Ω_1.

If l' does not decode to an encoding of zero (i.e. l' does not decrypt to 1), peers verify the ZK-proof Φ_4. If Φ_4 is successfully verified, peers accept T_O as a valid ownership declaration, which is not associated with a counterfeit, and S_{n+1} as the encoded set of coefficients of the updated polynomial to be stored in the BC. Accordingly, protocol version V3 of PrivIdEx preserves counterfeit elimination, without revealing the CRH values of the identity assets associated with transactions T_O, thereby preserving unlinkability across T_O and T_T associated with the same identity asset. To enable proof of ownership and correctness, while preserving unlinkability, after successful verification of T_O, the peers add the CRH of the ownership declaration O, i.e. f = CRH(O), as a leaf in the Merkle hash tree (MHT) data structure stored in the BC. This marks the end of phase 0.a for protocol V3.

The basic idea of proving ownership during identity asset transfer (phase 3.b) is to prove that the user and the IAP know a *path P* in the MHT from a leaf f, which contains the CRH of a valid ownership declaration O, to the root RT and that the user and the IAP own the private keys associated with the public keys committed in C_U and C_P of such O, respectively. To prove correctness, the IAP proves a similar statement, that is, the IAP knows a path P from f, which contains O with a commitment C_a to a CRH value that matches the CRH value of the identity asset being transferred, to RT. Note that both the owners of the identity asset should prove in zero-knowledge, the knowledge of the same path P in the MHT.

During the identity asset transfer phase, when the IAP requests the user's proof of ownership to the identity asset via M_{E1}, the user creates a ZK-proof Φ_5 proving her ownership and sends it to the IAP via M_{E2}. The IAP then creates a ZK-proof Φ_6 proving its ownership and correctness and transfers the encrypted identity asset along with Φ_5 and Φ_6 via T_T. The NP statements associated with Φ_5 and Φ_6, of which the basic idea has been outlined above, are included in the technical report associated with the paper [16]. Peers verify

Φ_5 and Φ_6 in W_P and confirm that T_T preserves correctness and ownership assurance.

In order to ensure financial fairness while preserving unlinkability, protocol V3 should integrate an anonymous and an unlinkable payment system such as Zerocash [4], which also enables making locked payments described under protocol V2.

5 IMPLEMENTATION AND EXPERIMENTS

In what follows, we present the details of the implementation and experiments on the main building blocks of PrivIdEx. Our goals are two folds: i) understanding the challenges and feasibility of the implementation of some of the most complex building blocks, e.g. ZK-proofs for the NP statements used in PrivIdEx; ii) evaluating circuit size, execution times and storage requirements of ZK-proofs for the NP statements. Experiments were run in a desktop machine running Ubuntu 18.04.1 LTS with 16GB memory and Intel i7-4790 CPU @ 3.6GHz.

We used the ZK-SNARK construction (see Section 2.2) to prove/verify satisfying assignments to the NP statements listed in Section 4. ZK-proofs for the NP statements were created using ZK-SNARKs, following the five steps process listed in Section 2.2. First, the cryptographic primitives in the NP statements were instantiated with specific algorithms. Then the circuits for the NP statements were designed and implemented using the Jsnark [17] framework. The Jsnark framework allows one to write circuits in a format compatible with the ZK-SNARK compilers and provides building blocks called 'gadgets' for designing circuits. In order to compile the circuit into a QAP and to prove/verify in zero-knowledge the satisfiability of the assignment given by the prover, Jsnark interfaces with Libsnark [23] - the widely used library implementing the ZK-SNARK construction. The challenges in this process include, but not limited to: i) gadgets for certain cryptographic primitives, such as Elgamal encryption used in NS4, are not yet available in Jsnark; ii) as different existing Jsnark gadgets accept inputs in different formats, we had to standardize the input formats of these gadgets before wiring them together to form the required circuit.

In circuit 1 (see Figure 3) built for the NP statement NS1 used in V1 of PrivIdEx, the cryptographic hash (CRH) algorithm is instantiated with the widely used SHA-256 and the symmetric encryption algorithm is instantiated with SPECK128 [10], due to its light weight properties. The SPECK128 gadget is wrapped with the gadget implementing symmetric encryption in CBC mode. We evaluated the circuit size (see Table 2), running time (see Figure 4), and storage requirements (see Figure 5) associated with the three algorithms of ZK-SNARKs for circuit 1, by varying the size of the identity asset (A'). Increase in the size of A' increases the size of the proving key and the circuit (i.e., number of constraints in the circuit), which in turn affects the running times of the key generator, which takes the circuit as inputs, and the prover, which takes both the circuit and the proving key as inputs. However, the increase in running time of the prover is much less than that of the key generator, which is good because the prover is run each time an identity asset is exchanged, whereas the key generator is run only at system setup. Proof size, verification key size, and verifier running time are constant irrespective of the size of the secret input A'. Note that for the scope of this paper, we assume a fixed size for the identity assets (those with shorter sizes can use padding) exchanged in a given deployment of PrivIdEx, because it is an overhead to deploy multiple circuits for different sizes. We discuss the mechanisms to allow identity assets of different sizes in the technical report [16].

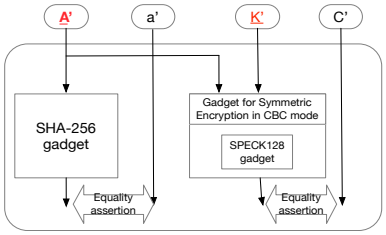


Figure 3: Circuit 1 built for NS1: *Given a cipher text C', a CRH value a', I know the following secrets: an identity asset A' and a key K' s.t $a' = CRH(A')$ and $C' = Enc_{K'}(A')$.*

Size of (A')	64 bytes	128 bytes	256 bytes	512 bytes	1024 bytes
Number of constraints	74,429	126,505	230,675	451,210	855,569

Table 2: Circuit size vs the size of the identity asset (A'), for Circuit 1.

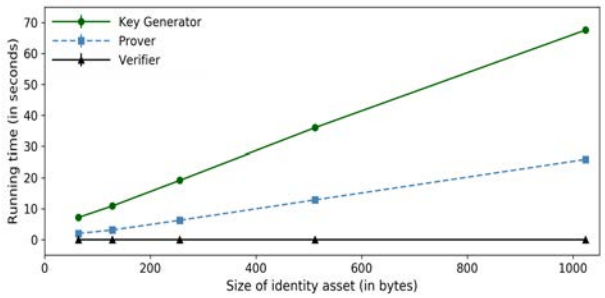


Figure 4: Running time vs the size of the identity asset (A'), for Circuit 1.

In circuit 2 (see Figure 6) built for the NP Statement NS2 used in V2 of PrivIdEx, the signature S is the only secret input, which is created by the provers (i.e. the IAP and the user) locally (i.e. outside of the circuit). Although PK theoretically consists of both RSA modulus and public exponent, only the RSA modulus is given as input PK, and the public exponent is set to a hard coded constant, according to the implementation details of the RSA algorithm [6]. In order to decide the size of the nonce M with optimal trade-off between security and performance of ZK-SNARKs for circuit 2, we evaluated the performance by varying the size of M. However, as shown in Table 3, there was a negligible impact on the performance when the size of the public input M was doubled. Therefore, we decided to use 128 bits as the size of the nonce.

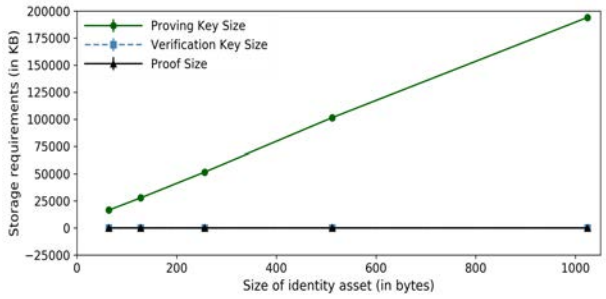


Figure 5: Storage size vs the size of the identity asset (A'), for Circuit 1.

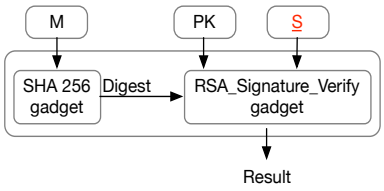


Figure 6: Circuit 2 for NS2: *Given a public key* PK*, a message* M*, I know a secret signature* S*, s.t RSA_Sig_Verify*(M, S, PK) = *True*, where S = RSA_Sig(M, private key of PK).

	64 bits	128 bits
Circuit size (number of constraints)	119,146	119,344
Key gen running time	12.8659(s)	12.8729(s)
Proving key size	32,903(KB)	32,938(KB)
Verification key size	3.2856(KB)	3.597(KB)
Prover running time	3.38(s)	3.3827(s)
Proof size	0.28(KB)	0.28(KB)
Verifier running time	0.0045(s)	0.0045(s)

Table 3: Performance numbers vs the size of the nonce (M), for Circuit 2.

A summary of the insights derived from the above experiments are as follows: 1) Increase in the size of *secret* inputs increases circuit size and prover key size, thereby increasing the running times of the key generator (which is run only once for the entire system lifetime) and the prover (which is run only once at ownership declaration and at each round of exchange of an identity asset). 2) Increase in the size of public inputs has negligible impact on the performance of ZK-SNARKS associated with a given circuit. 3) Proof size, verification key size, and verifier running time are negligibly affected (if at all) by the circuit complexity (e.g. circuit 1 and circuit 2 use different gadgets with varying complexity), size of secret inputs (e.g. experiments on circuit 1,) and size of public inputs (e.g. experiments on circuit 2). This makes using ZK-proofs based on ZK-SNARKs very suitable for use in PrivIdEx to ensure privacy and security properties of identity asset exchange in a decentralized identity ecosystem backed by a BC network, where multiple peers may run the verification algorithms associated with W_C and W_P (see Figure 2). We refer the reader to the technical report [16] for the details of the remaining circuits.

6 SECURITY AND PRIVACY PROOFS

In what follows, we prove that PrivIdEx (V3 - which addresses all three privacy requirements - see Section 4.3) protects against the threats mentioned in the threat models for user privacy (see Section 3.3) and protocol security (see Section 3.2).

The following lemma establishes that an adversary (referred to as Adv1), whose goal is to compromise the user's privacy (see Section 3.3), does not learn any information on the identity asset and the identity of the parties the user interacts with. Based on the information that Adv1 learn from the protocol transcripts of PrivIdEx, Adv1 cannot link different transactions of the same user.

Lemma 6.1. *PrivIdEx (V3) preserves confidentiality of the user's identity asset and anonymity and unlinkability of the user's transactions against Adv1.*

Proof (informal): Based on the security of the Diffie-Hellman key exchange used to establish a key between the IAP and the IAC, during the handshake in phase 2 of the protocol, Adv1 cannot learn the key used to encrypt the identity asset in T_T of phase 3.b (see Figure 2). Hence, *confidentiality* of the identity asset of the user is preserved against Adv1.

Due to the computationally hiding property of the underlying commitment scheme, the actual identities of the IAP and the user are not revealed to Adv1 via commitments: C_P and C_U included in the ownership declaration O of T_O in phase 0.b. Therefore, Adv1 does not learn the identity of the IAP who creates an identity asset for a user as well as the identity of the user for whom the identity asset is created, during phase 0.b. All the transactions posted to the BC include senders' and intended recipients' pseudonyms. A pseudonym P of a participant, i.e. P = CRH(public key in the pseudonymous certificate) is indistinguishable from a random string. Therefore, Adv1, which does not collude with the CA, does not learn the identity of the parties interacting via the BC. Due to the zero-knowledge property of ZK-SANRKs, the identity of which the ownership is proved in Φ_5 and Φ_6 is not revealed to Adv1 in T_T. Anonymity of the underlying payment scheme, which is used to pay any required monetary compensation, ensures that Adv1 does not learn the identity of the IAP or the IAC via the associated payment transactions. Therefore, throughout the protocol execution, *anonymity* of the participants is preserved from Adv1.

The different pieces of information involved in the protocol execution that Adv1 can use to link different transactions of the same user, are as follows: i) identity (i.e. pseudonyms) of the parties; ii) the CRH of the identity asset; iii) cipher text encrypting the identity asset; iv) any payment transactions created to pay monetary compensations for the identity assets. Since new pseudonyms are used by the participants for the execution of each round of the identity asset exchange, Adv1 cannot link such transactions via pseudonyms. Due to the computationally hiding property of the commitment scheme, the CRH of the identity asset is not revealed to Adv1 via commitment: C_h in O of T_O. Due to the encoding scheme En not allowing one to decode the encoded values in l and S_{n+1}, the CRH of the identity asset is not revealed to Adv1 via any of I1-I4 submitted to the BC via T_O. Due to zero-knowledge property of

ZK-SANRKs, the CRH of the identity asset, which is used to prove correctness in Φ_6, is not revealed to Adv1 in T_T. Hence Adv1 cannot link transactions via the CRH of the identity asset. Due to semantic security of the underlying symmetric encryption scheme, Adv1 cannot link the exchange transactions encrypting the same identity asset. Unlinkability of the underlying payment scheme ensures that Adv1 cannot link the identity asset exchange transactions via the associated payment transactions. Therefore, Adv1 does not learn any information helping to link transactions of the same user, hence, unlinkability is preserved against Adv1. ■ We refer the reader to the technical report [16] for the formal proofs of Lemma 6.1.

The following lemma establishes that an adversary (referred to as Adv2), whose goal is to compromise security of the identity asset exchange protocol (see Section 3.2), cannot create a counterfeit of an existing identity asset, transfer a fake identity asset that has not been legitimately created in the identity ecosystem, and claim false ownership to an identity asset.

Lemma 6.2. *PrivIdEx (V3) preserves correctness, ownership assurance and counterfeit elimination against Adv2.*

Proof (informal): Due to the additive homomorphic property of the encoding scheme based on the Elgamal encryption scheme, and the soundness property of ZK-SNARKs (which is based on the knowledge of coefficient assumption) used to create the ZK-proof Φ_3, Adv2 cannot submit an ownership declaration in T_O for an identity asset which is a counterfeit, without failing the validation W_C run by the peers in phase 0.b. Again, due to the soundness of ZK-SNARKs, used to create Φ_5 and Φ_6, we have that: i) an IAP controlled by Adv2 cannot transfer a fake identity asset because the IAP cannot provide commitments to a valid Merkle hash tree path of an ownership declaration O, which contains a commitment to a CRH value that matches the CRH of the identity asset being transferred, in the satisfying assignment to Φ6; ii) a user and an IAP controlled by Adv2 cannot claim false ownership because they cannot provide commitments to a valid Merkle hash tree path of an O, which contains commitments to the CRH of public keys for which they own the private keys, in the satisfying assignments to Φ_5 and Φ6. ■ We refer the reader to the technical report [16] for the formal proofs of Lemma 6.2.

Theorem 6.3. *PrivIdEx preserves the identified privacy properties against Adv1 and the security properties against Adv2.*

Theorem 6.3 follows from lemma 6.1 and Lemma 6.2.

7 RELATED WORK

Identity management research has a rich history. Here we focus on the proposals focusing on exchanging users' identity information between SPs. Next we discuss approaches for privacy enhancing techniques for BC applications and show that such approaches alone cannot address the problem that we focus on. OpenID Exchange (OIX) [11] and OpenID Connect [22] are industry standards which address some form of identity exchange. Such protocols, however, have one central IDP from whom other SPs obtain identity information of a user, and do not address a user's transactional privacy requirements. Identity Mixer [9] is an anonymous credential system which enables users to authenticate to SPs in an unconditionally unlinkable manner, while selectively disclosing users' identity information. Identity Mixer also involves a central IDP from which the user obtains identity tokens; the IDP is known to the SPs while the SPs are not known to the IDP. The USA and UK governments have developed nation-scale identity management systems which enable government identity consumers to obtain users' identity information from third party identity providers, where consumers and providers are anonymous to each other, in order to preserve users' privacy. However, such systems introduce a government managed broker to mediate the identity exchange transactions, which learns the identity of the two exchanging parties, and hence, can track the users' transactions. More recently, decentralized identity management systems have been proposed that leverage BC technology to avoid centralized parties managing users' identity [26]. However, such systems do not address all the privacy requirements that we consider.

Zerocash [4] enables a sender to transfer bitcoins to a recipient in an anonymous and unlinkable manner. PrivIdEx differs from Zerocash in multile respects, including: i) Zerocash prevents double spending of bitcoins whereas PrivIdEx enables transferring the same identity asset as many times as needed by the legitimate owners to different consumers; ii) there is only one anonymous owner for bitcoins at a given time, whereas there are two owners for an identity asset. Hawk [18] is a framework for privacy preserving smart contracts. Hawk alone does not address all the privacy and security requirements of a given use case, such as the one we focus on, which involves multiple phases and repeating interactions among the participants, based on the same identity asset. Zero Knowledge Asset Transfer (ZKAT) [1] by Heperledger Fabric is based on the unspent transaction output (UTXO) model of bitcoin. Hence, it supports exchange of monetary transactions which cannot be double spent, which is different from our use case. Therefore, ZKAT alone is not sufficient to enable privacy preserving and secure identity asset exchange.

8 CONCLUSION

We proposed PrivIdEx - a privacy preserving and secure protocol for identity asset exchange over a decentralized identity ecosystem backed by a permissioned BC network. PrivIdEx enables different SPs that a user interacts with to re-use the identity assets created for the user, eliminating the cost of repeated identity verification and due diligence processes, without having to worry about privacy and security concerns in doing so. Analysis of the threat model, protocol design and implementation and experiments are presented in an *incremental* approach to help readers understand the specific challenges posed when achieving each of the identified privacy properties and the mechanisms developed to address them, which also helps in selectively enabling those properties as required by a given identity ecosystem.

One potential future extension of PrivIdEx is to integrate it with the Identity Mixer based CA in Hyperledger Fabric [1] BC netowork to achieve unlinkability against collusions between the CA and an actor in the BC, so that we can eliminate the third assumption mentioned in Section 3.1. Other relevant future work is to generalize PrivIdEx to facilitate privacy preserving and secure exchange of any confidential digital asset with multiple owners, such as song lyrics,

music, write-ups, e-books, etc., which has not yet been addressed by the existing digital asset exchanging platforms.

Acknowledgement

We thank Ahmed Kosba for clarifications about jsnark and Fabrice Benhamouda for discussions regarding the project. This work is supported by an IBM PhD fellowship award.

REFERENCES

[1] E. Androulaki, S. Cocco, and C. Ferris. 2018. Private and confidential transactions with Hyperledger Fabric. https://developer.ibm.com/tutorials/cl-blockchain-private-confidential-transactions-hyperledger-fabric-zero-knowledge-proof/ Accessed: 1-Nov-2018.

[2] M. Andrychowicz, S. Dziembowski, and D. Malinowski. 2014. Secure Multiparty Computations on Bitcoin. In *IEEE Symposium on Security and Privacy*.

[3] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev. 2018. Scalable, transparent, and post-quantum secure computational integrity. In *Cryptology ePrint Archive: Listing for 2018*.

[4] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. 2014. Zerocash: Decentralized Anonymous Payments from Bitcoin. In *IEEE Symposium on Security and Privacy*.

[5] David Birch. 2016. Putting identity on the blockchain. http://www.chyp.com/putting-identity-on-the-blockchain-part-1-find-a-problem/.

[6] D. Boneh. 1998. Twenty Years of Attacks on the RSA Cryptosystem. https://crypto.stanford.edu/%7Edabo/pubs/papers/RSA-survey.pdf Accessed: 22-Sept-2018.

[7] Luis T.A.N. Brandao, N. Christin, G. Danezis, and Anonymous. 2015. Toward Mending Two Nation-Scale Brokered Identification Systems. In *Proceedings on Privacy Enhancing Technologies*.

[8] B. Bunz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. 2018. Zerocash: Decentralized Anonymous Payments from Bitcoin. In *IEEE Symposium on Security and Privacy*.

[9] J. Camenisch and A. Lysyanskaya. 2001. An Efficient System for Non-Transferable Anonymous Credentials with Optional Anonymity Revocation. In *Proceedings of EUROCRYPT '01*. 93–118.

[10] R. Beaulieu et. al. 2013. The Simon and Speck Families of Lightweight Block Ciphers. https://eprint.iacr.org/2013/404.pdf Accessed: 22-Sept-2018.

[11] Open Identity Exchange. [n. d.]. OIX - Open Identity Exchange. https://www.openidentityexchange.org/ Accessed: 22-Nov-2017.

[12] Hyperledger Fabric. 2018. A Blockchain Platform for the Enterprise. https://hyperledger-fabric.readthedocs.io/en/release-1.3/ Accessed: 16-Oct-2018.

[13] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. 2007. Secure Distributed Key Generation Protocol.. In *J Cryptology*.

[14] R. Gennaro, C. Gentry B. Parno, , and M. Raykova. 2013. Quadratic Span Programs and Succinct NIZKs without PCPs. In *EUROCRYPT*.

[15] GOV.UK. 2018. Introducing GOV.UK Verify. https://www.gov.uk/government/publications/introducing-govuk-verify/introducing-govuk-verify Accessed: 22-Sept-2018.

[16] H. Gunasinghe, A. Kundu, E. Bertino, H. Krawczyk, K. Singh, S. Chari, and D. Su. 2019. PrivIdEx: Privacy Preserving and Secure Exchange of Digital Identity Assets. https://www.cerias.purdue.edu/assets/pdf/bibtex_archive/2019-01.pdf Accessed: 15-Feb-2019.

[17] Ahmed Kosba. 2017. jsnark. https://github.com/akosba/jsnark Accessed: 22-Nov-2017.

[18] A. Kosba, A. Miller, and E. Shi. 2014. Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts.. In *IEEE Symposium on Security and Privacy*.

[19] S. Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System.

[20] Nat. 2010. Is Expressing Levels Enough for LOA2+? https://nat.sakimura.org/2010/09/03/is-expressing-levels-enough-for-loa2/ Accessed: 22-Sept-2018.

[21] United States Postal Office. 2014. FCCX Briefing. https://csrc.nist.gov/csrc/media/events/ispab-june-2014-meeting/documents/ispab_jun2014_fccx-briefing_glair.pdf Accessed: 22-Sept-2018.

[22] OpenID. 2017. Welcome to OpenID Connect. http://openid.net/connect/ Accessed: 22-Nov-2017.

[23] scipr lab. 2017. C++ library for zkSNARKs. https://github.com/scipr-lab/libsnark Accessed: 22-Nov-2017.

[24] Prabath Siriwardena. 2017. A Deeper Look Into Bitcoin Internals. https://medium.facilelogin.com/pay-with-bitcoin-to-play-with-a-fidget-spinner-86b7b43414c0 Accessed: 22-Sept-2018.

[25] Prabath Siriwardena. 2017. Identity on Blockchain (Part I). https://medium.facilelogin.com/identity-on-blockchain-part-i-a59d7abe75c0 Accessed: 22-Sept-2018.

[26] sovrin. 2017. Identity For All. https://sovrin.org/ Accessed: 22-Nov-2017.

[27] European Union. 2016. General Data Protection Regulation. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679 Accessed: 22-Sept-2018.

RED: Redundancy-Driven Data Extraction from Result Pages*

Jinsong Guo
University of Oxford
United Kingdom
jinsong.guo@cs.ox.ac.uk

Valter Crescenzi
University Roma Tre
Italy
crescenz@dia.uniroma3.it

Tim Furche
University of Oxford & Meltwater
United Kingdom
tim.furche@cs.ox.ac.uk

Giovanni Grasso
University of Calabria & Meltwater
Italy
giovanni.grasso@unical.it

George Gottlob
University of Oxford & TU Wien
United Kingdom
georg.gottlob@cs.ox.ac.uk

ABSTRACT

Data-driven websites are mostly accessed through search interfaces. Such sites follow a common publishing pattern that, surprisingly, has not been fully exploited for unsupervised data extraction yet: the result of a search is presented as a paginated list of result records. Each result record contains the main attributes about one single object, and links to a page dedicated to the details of that object.

We present RED, an automatic approach and a prototype system for extracting data records from sites following this publishing pattern. RED leverages the inherent redundancy between result records and corresponding detail pages to design an effective, yet fully-unsupervised and domain-independent method. It is able to extract from result pages all the attributes of the objects that appear both in the result records and in the corresponding detail pages.

With respect to previous unsupervised methods, our method does not require any a priori domain-dependent knowledge (e.g, an ontology), can achieve a significantly higher accuracy while automatically selecting only object attributes, a task which is out of the scope of traditional fully unsupervised approaches. With respect to previous supervised or semi-supervised methods, RED can reach similar accuracy in many domains (e.g., job postings) without requiring supervision for each domain, let alone each website.

CCS CONCEPTS

• **Information systems → Data extraction and integration**.

KEYWORDS

Web Data Extraction; Automatic Wrapper Generation; XPath

ACM Reference Format:
Jinsong Guo, Valter Crescenzi, Tim Furche, Giovanni Grasso, and George Gottlob. 2019. RED: Redundancy-Driven Data Extraction from Result Pages. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313529

*This work is supported by the EPSRC programme grant EP/M025268/1 VADA, and the EU Horizon 2020 grant 809965 LAMBDA.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313529

1 INTRODUCTION AND OVERVIEW

Data-driven websites across many application domains (e.g., real estate agencies, e-commerce) follow a common publishing pattern to provide access to the underlying data: A search interface to query specific types of objects (e.g., products, properties) yields the relevant objects as a list of result records, often paginated into *result pages*. Every result record contains key attributes about one single object, so that the user can get a first impression of the result before deciding to dig further into detailed information. Each result record includes a link to a *detail page*, containing more attributes [24].

To the best of our knowledge, no existing data extraction system has targeted the inherent data intra-site redundancy underlying this popular publishing pattern. Rather, most existing unsupervised approaches [2, 8, 9, 29, 30, 34, 35] can be applied either over a collection of result pages, or over a collection of detail pages. Other approaches rely on the same publishing pattern, but focus only on segmenting the result pages [28], or rely on the much weaker signals arising from the aligning of the labels of the fields on the search form directly against the labels of the data on the detail pages [37]; finally, there are several approaches that focus on the problem of finding redundancy [3, 38] among several sites, but the problem quickly trespasses on that of integrating data coming from autonomous sources [3, 5, 6, 23, 38], a problem that is well known not to have a simple solution [14].

This paper proposes RED, the first data extraction method that leverages this intra-site redundancy to extract data records from result pages. Given a set of result pages and the corresponding set of detail pages from a website, RED infers a collection of *extraction rules* (or simply *rules*). Each one is capable of extracting the values for a certain attribute, e.g., the price of a product, from all the records of a result page. These rules can be applied to every result page of the same website to extract all the relevant data.

We show that exploiting this publishing pattern on a site provides a *low-hanging fruit* opportunity for significantly improving the precision of fully unsupervised data extraction. Existing unsupervised approaches struggle to distinguish relevant data (i.e., object attributes) from noise. Moreover, they are based on a fragile trade-off between the expressiveness of formalism used for describing the extraction rules and the efficiency of the learning task. Conversely, intra-site redundancy distinguishes relevant data that appears both on result records and on detail pages from noise that only appears on either without any supervision, and allows to filter

out noisy candidate extraction rules even when using an expressive formalism to specify them.

Key idea: Exploit the inherent intra-site redundancy between result records and their corresponding detail pages to fully automatically distinguish between slots in the template structure that correspond to actual object attributes.

Figure 1 shows a (simplified) instance of a real estate website that serves as running example. It consists of a result page containing six records along with their corresponding detail pages. The result records and detail pages publish attributes such as Price, Location, number of Beds, and Type of properties. Colored annotations highlight overlapping values for attributes appearing within the records in the result page and the corresponding detail pages. For this example, RED is capable of generating rules extracting the correct values of all attributes for all result records. Figure 2a shows the values extracted by the extraction rules when applied to the example result page in input.

RED generates these rules in three, sequential steps: **(1)** During the *extraction rules generation* step, RED generates two sets of rules, namely the sets of *result rules* and *detail rules*: Figure 2b and Figure 2c show the values extracted by some of the generated rules (shown in Figure 5a and Figure 5b) on the result page and on the detail pages, respectively, for our running example (where r_i (d_i) is the i-th result (detail) rule generated). RED's extraction rules generation algorithm (Section 3) selects the rules from a fragment of XPath [13] expressions that, as empirically verified over many real websites, includes correct extraction rules.

The main challenge in this step is generating a *complete* set of rules efficiently: at least one correct detail rule and one correct result rule is needed for every attribute. It turns out that the completeness of the rules generation algorithm can be achieved only at expense of using a rather expressive family of extraction rules, which means generating many candidate rules, most of which are actually selecting noise.

(2) In the second step, *redundancy seeking*, RED aims to identify pairs of result/detail rules that are considered *redundant*, as they extract the same values for every object. In this quest for redundancy, RED is faced with various challenges. First, in addition to *neat* redundancy between correct rules, there is inevitably a lot of *noisy* redundancy: Some values may occur multiple times across the records (typical if the search is narrow, e.g., only properties in London, or because attributes have a narrow domain, e.g., Beds). Moreover, some values may be repeated several times within the same record or detail page (the third detail page contains a second occurrence of '`Studio`', dotted-blue underlined in Figure 1).

To complicate matters, certain attributes may be optional i.e., not report a value for some result record, such as the Price attribute which is missing for the third and sixth properties of our running example. Both rules r_0 and d_0 select Price values: d_0 is applied to each of the detail pages separately, and can therefore extract the price value if occurring on the page, or it can just report that the value is missing (indicated by *nil*). Conversely, r_0 is applied once on the whole result page (and not on the individual records): it can only extract the price values found on the page, irrespective of which record they belong to. Therefore, taken alone, r_0 is not able to distinguish which records contain a value, and which do not (in our example the third and last record). That leaves r_0 and d_0 mis-"aligned" and without further consideration they would not be considered redundant. This problem has been already tackled by known complex and error prone segmentation techniques for splitting a result page into records [19, 28, 29]. Rather, RED devises an innovative soft-segmentation technique (Section 4.1) which, again, leverages the underlying intra-site redundancy: In particular, RED exploits the presence of navigational links (named *detail links*) that point from each result record to the corresponding detail page. In Figure 1, for each record, one occurrence of the detail link is highlighted (by the departing orange solid arrow). All these detail link occurrences are captured by an extraction rule l, named *link rule*, whose generation is facilitated by the knowledge of urls of the detail pages given in input to RED.

(3) In its final step, named *noise removal*, RED addresses the issue of finding and discarding pairs of rules that exhibit some limited redundancy, but not sufficient to likely be correct extraction rules. RED addresses this challenge by implementing a noise cleaning process (Section 5) that aims at separating the redundancy resulting from the result/detail publishing pattern, from that arising by accident.

Finally, RED outputs all the result extraction rules left after the previous steps along with the special link rule, as shown in Figure 2a for the running example. It is worth noting that, while in this paper we are focusing on data extraction from result page records, RED is perfectly capable of inferring valid extraction rules also for attributes published on the detail pages. Aside of space reasons, we focus on result rules as executing only such rules at a large scale is much cheaper: Once the rules have been generated, no more detail pages have to be crawled, and that is a significant saving, as there are many more detail pages than result pages.

The paper is organized around the main contributions of RED:

- The first formal description of the typical result/detail publishing pattern and the definition of the *Finding Result Attributes Problem* (Section 2);
- A highly accurate, domain independent, and fully unsupervised data extraction method that leverages the redundancy implied by the publishing pattern on real-life websites. It is presented in three steps: a rules generation algorithm to efficiently generate complete sets of extraction rules (Section 3); a technique to measure the redundancy of rules (Section 4), even in the presence of optional attributes (Section 4.1); and finally, a technique to separate and validate regular, attribute redundancy from noisy, accidental redundancy (Section 5);
- A proof-of-concept showing the feasibility of the approach, and the experimental evaluation (Section 6) showing that RED can attain highly accurate results scoring at rule level an F-measure $F_r > 91\%$, and even better results at value level with $F_v > 96\%$. We demonstrate its advantages against state-of-the-art large scale data extraction systems.

The following related problems have been already tackled in literature and are beyond the scope of the present paper: finding deep Web sources [36, 39, 40, 43]; filling the search fields with meaningful values to collect result pages [1, 4, 17, 25, 31–33, 42]; crawling paginated search result pages [21].

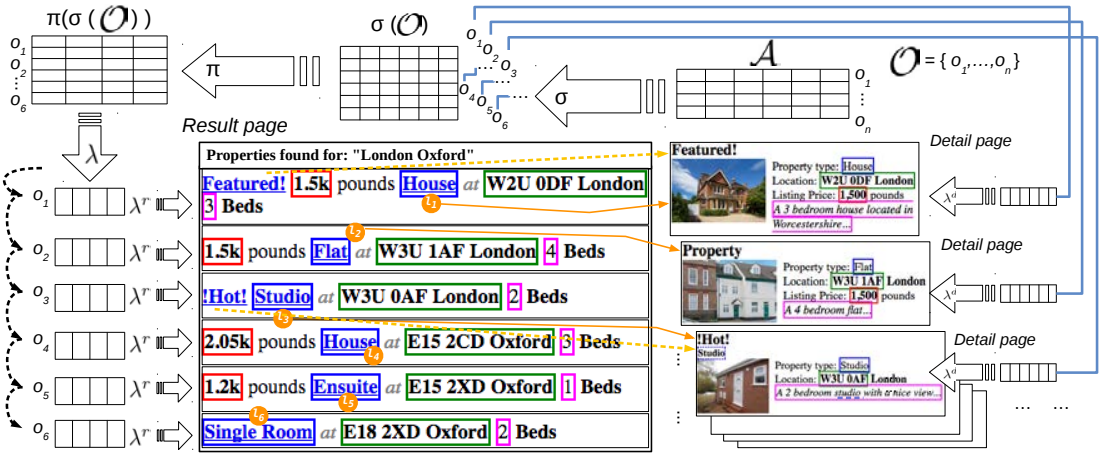


Figure 1: The result/detail publishing pattern.

r_0^b (Price)	r_2 (Type)	r_6 (Location)	r_9 (Beds)	l (Link)
1.5k	House	W2U 0DF London	3	l_1
1.5k	Flat	W3U 1AF London	4	l_2
nil	Studio	W3U 0AF London	2	l_3
2.05k	House	E15 2CD Oxford	3	l_4
1.2k	Ensuite	E15 2XD Oxford	1	l_5
nil	Single Room	E18 2XD Oxford	2	l_6

(a) RED's Output on the Running Example: Annotated rules to extract correct result records.

r_0: ⟨1.5k, 1.5k, 2.05k, 1.2k⟩
r_1: ⟨1.5k, Flat, Studio, House, Ensuite, at⟩
r_2: ⟨House, Flat, Studio, House, Ensuite, Single Room⟩
r_3: ⟨Featured!, 1.5k, 2.05k, 1.2k⟩
r_4: ⟨1.5k, 1.5k, 2.05k, 1.2k⟩
r_5: ⟨at, W3U 1AF Lon..., W3U 0AF Lon..., E15 2CD Ox..., E15 2XD Ox...⟩
r_6: ⟨W2U 0DF Lon..., W3U 1AF Lon..., ...,..., E15 2XD Ox..., E18 2XD Ox...⟩
r_7: ⟨at, W3U 1AF Lon..., W3U 0AF Lon..., E15 2CD Ox..., E15 2XD Ox...⟩
r_8: ⟨W2U 0DF Lon..., W3U 1AF Lon..., E15 2CD Ox..., E15 2XD Ox...⟩
r_9: ⟨3, 4, 2, 3, 1, 2⟩

(b) Values extracted by result rules.

d_0: ⟨1,500, 1,500, *nil*, 2,050, 1,200, *nil*⟩
d_1: ⟨House, Flat, Studio, House, Ensuite, Single Room⟩
d_2: ⟨Featured!, Property, !Hot!, Property, Property, Property⟩
d_3: ⟨pounds, pounds, A 2 bedroom studio..., pounds, pounds, A single...⟩
d_4: ⟨W2U 0DF Lon..., W3U 1AF, W3U 0AF, E15 2CD Ox..., E15 2XD, E18...⟩
d_5: ⟨W2U 0DF Lon..., W3U 1AF, *nil*, E15 2CD Ox..., E15 2XD, *nil*⟩
d_6: ⟨A 3 bedroom house..., A 4 bedroom flat..., A 2 bedroom studio..., ···⟩
d_7: ⟨*nil*, London, London, *nil*, Oxford, Oxford⟩

(c) Values extracted by detail rules.

Figure 2: Running example.

2 PROBLEM DESCRIPTION

The result/detail publishing pattern depicted in Figure 1 assumes that a website publishes data coming from an underlying abstract relation with attributes $\mathcal{A}$ containing all the available information about a set of objects O. In our example, the abstract relation contains one tuple for every published property, each with attributes such as the property's Price, Address, number of Beds, and Type.

Result pages and detail pages are generated by various *scripts* that retrieve data from the abstract relation, embed them into an HTML template, and publish them as web pages. This process performs a sequence of transformations: An initial selection (σ, to recycle a symbol from relational algebra) produces the set of objects to publish in each result page as encoded into HTML source code by the *result page script* λ. Every result page contains a set of *result records*, each produced by applying the *result record script* λ^r to one published object. The corresponding detail page is the result of applying the *detail script* λ^d on the very same object, and is linked from the corresponding result record.

According to the model depicted in Figure 1, even if the detail and result record scripts work on the same set of objects, they may end up publishing different sets of attributes. The detail pages include all the attributes of the abstract relation, whereas the result pages publish only a subset of these attributes selected by a projection operation π.

We now introduce the problem of recovering the values of the attributes published on the result pages. An extraction rule either locates one or several string values from a single page, or it produces a distinct special value *nil* to denote the absence of a value. For our purposes, extraction rules are specified by using XPath expressions [13] belonging to a simple but carefully designed fragment (as detailed in Section 3).

We distinguish two types of extraction rules, named *detail* and *result* rules after the type of pages they are meant to work on.

We call *detail rule* any extraction rule that when applied on a detail page, produces at most one value or *nil*. We use d to denote a detail extraction rule, and we use $d(p)$ to denote the value it extracts from detail page p. Precisely, $d(p)$ is either the XPath *string-value* obtained by applying the rule d on detail page p, or it is *nil* if the XPath expression returns an empty *node-set* [13].

A detail rule for an attribute A is said to be *correct* if it extracts its values from every detail page in the input and extracts *nil* for those pages that do not contain a value for A. We call *noisy* (or *incorrect*) a rule that is not correct: noisy rules mix values of several attributes or they extract part of the underlying HTML template. We distinguish the special case of a *partially-correct* rule for an attribute A, i.e., a rule extracting only correct values of attribute A except for some pages, on which it wrongly extracts *nil*. The value extracted by a detail rule is naturally associated with the object corresponding to the detail page it is applied to.

Similarly, we call *result rule* any extraction rule meant to work on result pages and we denote it by r. A result rule applied on a result page p produces zero, one, or several values denoted $r(p)$. Precisely, r returns the *string-value* for each node in the *node-set* returned by evaluating its XPath expression on the result page p [13].

Result rules, differently from detail rules, cannot produce *nil* values at all, even in the case that they return an empty node-set. Indeed, without knowing the exact boundaries of every result record, the values extracted by a result rule cannot be trivially associated to the record/object they belong to on the result page (Section 4.1 describes a solution to this problem).

Therefore, the notion of *correctness* of a result rule for an attribute A differs from the notion of correctness of a detail rule, even if they are associated with the same attribute: a result rule is correct if it produces exactly the ordered set of non-*nil* values of attribute A as they occur in the source of that page. By *partially-correct* result rule for an attribute A we mean a result rule extracting a strict subset of the correct values of A.

The problem that we aim to solve can be formulated as follows:

Problem (Finding Result Attributes). *Given a set of result pages P_r and the set of corresponding detail pages P_d over the same objects, find the correct values (including nil values) for every attribute A of the abstract relation published in the result pages.*

RED tries to solve this problem by producing a result rule r_A per every attribute A in the result pages, together with an additional result rule, named *link rule* and denoted by l, extracting exactly one occurrence of the link leading to the corresponding detail page per result record: an output result rule r_A associated with an optional attribute A is suitably annotated either r_A^a or r_A^b to specify how the values extracted by r_A should be padded with missing *nil* values, i.e., respectively *after* or *before* occurrences of the detail links.

3 EXTRACTION RULES GENERATION

We introduce an extraction rules generation algorithm working on both result and detail pages. It is a single-parameter algorithm designed to output rules from an XPath fragment whose expressiveness can be easily and effectively tuned by setting the parameter. Its goals are two-fold: on one hand, it aims at being complete, i.e., it has to generate at least a correct rule for every attribute; on the other hand, it should not generate too many noisy rules, because their presence makes the *Finding Result Attributes* problem harder.

The rules generation algorithm includes two main steps: *template analysis*, and *extraction rules enumeration*.

3.1 Template Analysis

The template analysis aims at identifying those nodes in the DOM tree [26] of the input pages that are *template* nodes (e.g., the `<li>` node, and the textual label: '`Price:`' in Figure 3a).[1]

RED's template analysis algorithm is inspired by ExAlg [2], suitably adapted to the result/detail publishing pattern. Our analysis just aims at deciding, for every node in the pages, whether it should be considered as part of the template or not; conversely, ExAlg also solves the complex problem of finding a *full* description of the HTML template.

The analysis on a set of input pages P builds on the notion of *occurrence-vector*, i.e., a vector f of $|P|$ integers indexed by the pages in P, so that $f(p)$ reports how many occurrences of *equivalent* nodes are present in the DOM tree of page p. Two nodes are said to be equivalent if and only if two conditions hold: (i) they are either text nodes with the same value, or they both are element nodes associated with the same element name and attribute names, (ii) their respective parent nodes are equivalent (or they both are the root nodes). Equivalent nodes with the same occurrence-vector are then grouped into the same *equivalence class*.

As observed in [2], by considering a sufficiently large number of pages, the nodes that occur in large and frequently occurring classes are the scaffold of the underlying template. An inherent and significant limitation of this kind of statistical analysis is that the inferred equivalence classes become easily noisy as their support (total number of occurrences) or size (total number of nodes) decreases. Indeed, ExAlg's template inference process described is extremely brittle to the presence of noisy equivalence classes and cannot deal with singleton collections of input pages.

As in the original proposal, we use thresholds on the minimum size and on the minimum support to prevent the algorithm from generating too many noisy classes.[2] However, it is worth noticing that in our setting the template analysis is just an optimization aiming at reducing the number of generated extraction rules and that we have empirically observed that a single result page, usually containing up to tens of records, is already enough for the specific goals of our analysis.

The classes whose occurrence-vector exactly reports the number of objects published per page are called the *root* equivalence classes: their nodes occur exactly once in every detail page and as many times as the number of result records in every result page (an information available as part of RED's input). We classify as template nodes those occurring in the root equivalence classes or in any other equivalence class that appears less frequently than the root equivalent class in every page.[3]

Example 3.1. The root equivalence classes for the running example are ϵ_0^r and ϵ_0^d reported in Figure 4; the resulting template nodes are depicted with gray background in Figure 3. Notice that all their nodes are template nodes. $\epsilon_2^d, \epsilon_3^d, \epsilon_4^d, \epsilon_5^d$ are discarded because their support is too small and indeed they are noisy classes listing values that occur more than once by chance. ϵ_3^r is discarded because it occurs more frequently than the root equivalence class ϵ_0^r. Also nodes in ϵ_1^r, ϵ_2^r are correctly classified as template nodes. Notice that nodes in ϵ_4^r and ϵ_5^r are erroneously classified as template nodes.

The wrong classification of a value as template node will not prevent the generation of correct rules for the corresponding attribute as long as other occurrences of the values of that attribute are correctly classified as target values.

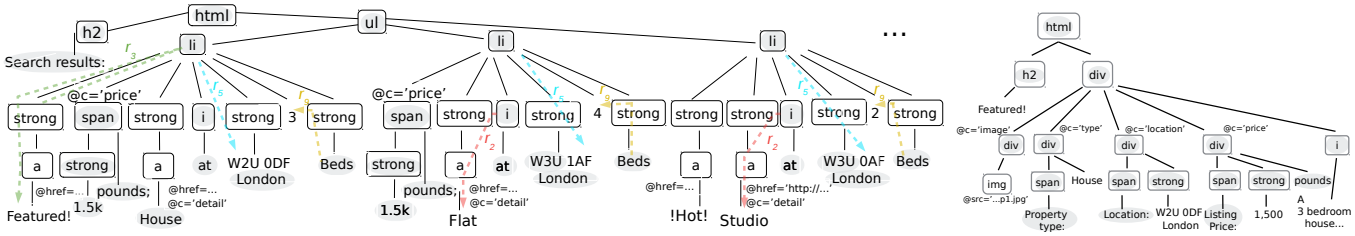
3.2 Extraction Rules Enumeration

The rules generation algorithm builds on the output of the template analysis to enumerate result (respectively, detail) extraction rules associated with tree-paths starting from a uniquely identifiable template node within the result records (resp., detail page), named *pivot*, and reaching a non-template *target value* node.

[1] Text nodes are split in several contiguous siblings nodes by tokenizing at word level.

[2] We consider only equivalence classes whose support is at least 20% of the input objects, but never fewer than 2, and only classes including at least 3 nodes.

[3] These classes are those associated with the *main* data structure of the two types of pages, i.e., the list of records of the result pages and the flat record of the detail pages, or those optional attributes nested within them.



(a) A portion of the DOM tree of the result page.

(b) DOM of the detail page 1.

Figure 3: Running example: DOM of input pages.

class	*nodes*	*occ. vector*
ϵ_0^r	{LI, I, at, A[@class = "detail"], Beds}	[6]
ϵ_1^r	{HTML, UL, H2, Search, results :, . . .}	[1]
ϵ_2^r	{SPAN[@class = "price"], STRONG, pounds}	[4]
ϵ_3^r	{STRONG}	[12]
ϵ_4^r	{House, A, 1.5k, W3U, E15, 2XD}	[2]
ϵ_5^r	{Oxford, London}	[3]
ϵ_0^d	{Property, type :, Location :, . . .}	[1, 1, 1, 1, 1, 1]
ϵ_1^d	{DIV[@class = "price"], SPAN, STRONG, Listing, Price :, pounds}	[1, 1, 0, 1, 1, 0]
ϵ_2^d	{House}	[1, 0, 0, 1, 0, 0]
ϵ_3^d	{1, 500}	[1, 1, 0, 0, 0, 0]
ϵ_4^d	{London}	[1, 1, 1, 0, 0, 0]
ϵ_5^d	{Oxford}	[0, 0, 0, 1, 1, 1]

Figure 4: Running example: template analysis.

r_0 : //span[contains(.,"pounds")]/../*[1]/text()[1]
r_1 : //li/*[2]/*[1]/text()[1]
r_2 : //i/ps::*[1]/*[1]/text()[1]
r_3 : //span[@class="price"]/../*[1]/*[1]/text()[1]
r_4 : //span[@class="price"]/*[1]/text()[1]
r_5 : //li/*[4]/text()[1]
r_6 : //i[contains(.,"at")]/fs::*[1]/text()[1]
r_7 : //i[contains(.,"at")]/*[4]/text()[1]
r_8 : //span[@class="price"]/fs::*[3]/text()[1]
r_9 : //strong[contains(., "Beds")]/ps::text()[1]
l : //a[@class="detail"]

(a) Result candidate rules.

d_0 : //span[contains(.,"Price:")]/../*[2]/text()[1]
d_1 : //span[contains(.,"type:")]/../text()[1]
d_2 : //h2/text()[1]
d_3 : //span[contains(.,"Location:")]/../fs::*[1]/text()[1]
d_4 : //span[contains(.,"Location:")]/fs::*[1]/text()[1]
d_5 : //span[contains(.,"Price:")]/../ps::*[1]/text()[1]
d_6 : //i/text()[1]
d_7 : //span[contains(.,"Location:")]/fs::*[2]/text()[1]

(b) Detail candidate rules.

Figure 5: Running example: extraction rules.

We use as candidate pivot every template node that is either a text, or an element with 'id' or 'class' attribute; every non-template text node is considered as a candidate target value. We then enumerate all the possible tree-paths leading from any pivot node to any target node by hopping over the tree along with a set of predefined XPath step expressions. Namely, we consider expressions capable of moving to the parent element node, to one of the children (either text or element), to the next/previous sibling element.

A couple of additional constraints on the tree-paths are enforced in order to reduce the number of enumerated tree-paths and discard those associated with incorrect rules. First, the path length is bounded by a threshold δ. Second, the analysis considers only a small subset of the input pages which is assumed unbiased.[4]

Each tree-path is translated into an executable extraction rule by appending several XPath step sub-expressions, eventually.

The first XPath step has to match the pivot, and takes one of the following forms depending on its node type: //e[contains(.,"v")] for a textual pivot v having e as parent node; //e[@id="v"] (resp., //e[@class="v"]), for an element pivot e having an id (resp., class) attribute valued v; just //e for any other element pivot.

Then it follows a sequence of XPath step expressions each after one of the hops composing the tree-path from the pivot to the target value: parent, child, following-sibling (abbreviated fs in Figure 5a), and preceding-sibling (ps); each step (except those on the parent axis) is also followed by an XPath positional predicate, e.g., '[4]', and by a node-test: The last step uses text() to target textual values, where all the other steps use '*' to select element nodes.

Example 3.2. Consider rule r_6 shown in Figure 5a: it is based on the 3 steps tree-path 'at ⇝ i ⇝ strong ⇝ W2U 0DF London' from the pivot node 'at' to the Location value. The XPath expression is obtained by combining: the expression selecting the pivot node (//i[contains(.,'at')]); the expression fs::*[1] to move to the next sibling element; and the final expression text()[1] to reach the target text value. A few tree-paths generating the result rules are shown directly in Figure 3 by means of colored dashed lines annotated with the rule and connecting the pivot to the target value across all the nodes in the path.

RED groups the generated rules by extracted values. Within a group producing the same values, only the rule associated with the shortest tree-path is retained, ties are broken by selecting the rule generated earlier. In the running example, r_7 is removed as it extracts the same values as r_5. The rationale to prefer shorter paths is that template nodes far away from the target values are

[4]We use at most 3 detail pages and 1 result page.

progressively less likely to generate rules that work reliably across all records and pages. More sophisticated criteria, such as those considering the robustness of the extraction rule [11, 22] could be adopted, but this is beyond the scope of the present paper.

4 REDUNDANCY SEEKING

RED analyzes the redundancy between all pairs of generated result and detail rules for identifying the likely correct rules. From an abstract point of view, a pair of result/detail rules can be considered *redundant* when the two rules end up extracting values associated with the same attribute of the abstract relation for every object.

Concretely, given one of such pairs composed of one result rule and one detail rule, a pairwise comparison of the extracted values is not trivial as they might extract a different number of values due to the presence of optional attributes. It is necessary to find out the correct alignment of the rules, i.e., where *nil* should be inserted to indicate a missing value within the list of values extracted by the result rule. We introduce a novel *soft segmentation* technique to find the alignment between result rules and detail rules, thus avoiding the tricky problem of finding the exact boundaries of every result record [20, 28]; then, we describe a score function for measuring the redundancy of a pair composed by one aligned result rule and one detail rule.

4.1 Soft Segmentation

The presence of optional attributes in the result pages can lead to the generation of rules that extract fewer values than the number of records. Soft segmentation is RED's technique for finding the correct alignment of the extracted values (w.r.t. the result records).

A prerequisite of the soft segmentation technique is the availability of exactly one detail link per each result record, i.e., that pointing to the corresponding detail page. RED leverages the knowledge of the detail pages, whose urls are assumed given as part of the input, to locate the detail link occurrences within the result records. As all occurrences must be associated with same "slot" of the underlying template, we have to disambiguate all the cases in which several copies of a detail link occur within the same result record, or optional detail links are present (e.g., for example in the first and third featured result records in Figure 1).

We apply the rules generation algorithm described in Section 3 by taking any of the link occurrences as target and generate a set of *link rules*. Only the rules extracting same number of links as the number of result records are considered, and in presence of several candidates, only the rule generated earliest is saved as link rule.

The soft segmentation technique finds out the correct alignment of the values extracted by a result rule w.r.t. detail the link occurrences during a traversal of the DOM tree. If a result rule is correct and extracts exactly the same number of values as the number of result records, the extracted values perfectly interleave with the detail link occurrences, i.e., either the values always occur before the links or they always occur after them. Conversely, in presence of result rules extracting fewer values, a correct alignment is needed.

Example 4.1. The link rule l of our running example is shown in Figure 5a. Let l_i denote the detail link node to the i-th detail page. Consider the correct rule r_9 extracting 6 values, in Figure 2b. It turns out that during an in-order traversal of the DOM tree in Figure 3a, detail links and extracted values interleave perfectly and occur as follows: $\langle l_1, 3, l_2, 4, l_3, 2, l_4, 3, l_5, 1, l_6, 2\rangle$. Instead, detail links and values extracted by r_8 do not: $\langle l_1$, W2U 0DF…, l_2, W3U 1AF…, $\underline{l_3}$, $\underline{l_4}$, E15 2CD…, l_5, E15 2XD…, $l_6\rangle$.

For dealing with these cases, *nil* values should be injected to reestablish the pattern that either every link precedes a value or vice-versa. At most two interleaving sequences of the values, called *alignments*, of a result rule, are possible: Given the result rule r, we use r^a (resp., r^b) to denote the alignment of the rule in which all nodes extracted by r are considered occuring *after* (resp., *before*) the corresponding detail link in every result record.

Example 4.2. Let s_8^a and s_b^b denote two sequences of exactly 12 elements (6 values plus 6 detail links) corresponding the alignment of rule r_8 w.r.t. detail links extracted by l in Figure 5a. Either every value occurs before a link:
$s_8^b = \langle \underline{nil}, l_1$, W2U…, l_2, W3U…, l_3, $\underline{nil}$, l_4, E15 …, l_5, E15…, $l_6\rangle$; or after a link:
$s_8^a = \langle l_1$, W2U…, l_2, W3U…, l_3, $\underline{nil}$, l_4, E15…, l_5, E15…, l_6, $\underline{nil}\rangle$.
We annotate r_8 to specify the relative position of the extracted values w.r.t. detail links so that the *nil* values can be properly inserted. Let p_r denote the result page of the running example:
$r_8^b(p_r) = \langle nil$, W2U 0DF…, W3U 1AF…, nil, E15 2CD…, E15 2XD…$\rangle$
$r_8^a(p_r) = \langle$W2U 0DF…, W3U 1AF…, nil, E15 2CD…, E15 2XD…, $nil\rangle$.

RED tries to enforce an admissible alignment to every produced result rule. Those rules that cannot be aligned are discarded.

Example 4.3. Consider the sequence of values extracted by r_1 and detail links in our running example:
$\langle$1.5k, l_1, Flat, l_2, Studio, l_3, House, l_4, Ensuite, l_5, l_6, at$\rangle$.
The presence of two consecutive links (l_5 and l_6) prevent values and links from interleaving perfectly. Thereby r_1 is removed.

Table 1 shows the alignments for all the result rules in Figure 5a w.r.t. link rule l for our running example.

Table 1: Aligned result rules after soft segmentation.

r_0^b:	$\langle$1.5k, 1.5k, nil, 2.05k, 1.2k, $nil\rangle$
r_2:	$\langle$House, Flat, Studio, House, Ensuite, Single Room$\rangle$
r_3^b:	$\langle$Featured!, 1.5k, nil, 2.05k, 1.2k, $nil\rangle$
r_5^a:	$\langle$at, W3U 1AF…, W3U 0AF…, E15 2CD…, E15 2XD…, $nil\rangle$
r_5^b:	$\langle nil$, at, W3U 1AF…, W3U 0AF…, E15 2CD…, E15 2XD…$\rangle$
r_6:	$\langle$W2U 0DF…,W3U 1AF…,W3U 0AF…,E15 2CD…,E15 2XD…,E18 2XD…$\rangle$
r_8^b:	$\langle nil$, W2U 0DF…, W3U 1AF…, nil, E15 2CD…, E15 2XD…$\rangle$
r_8^a:	$\langle$W2U 0DF…, W3U 1AF…, nil, E15 2CD…, E15 2XD…, $nil\rangle$
r_9:	$\langle$3, 4, 2, 3, 1, 2$\rangle$

The soft segmentation algorithm takes as input a result rule r, the link rule l and a set of result pages. It visits every page by ordering links and values into a sequence of occurrences and checks whether it is possible to inject *nil* values into the sequence to make values (including *nil*) and links to interleave perfectly. If so, it outputs at most two admissible alignments of the result rules, namely r^a and r^b (we simply write r where the alignments coincide).

4.2 Redundancy Score

Given a pair (r^*, d) composed of one aligned result rule (with $* \in \{a, b\}$) and one detail rule, and a set of objects O, the redundancy score of the pair is defined as the average pairwise score between their values:

$$\text{red}(r^*, d) = \sum_{o \in O} \frac{\text{score}(r^*(o), d(o))}{|O|}$$

where $d(o)$ and $r^*(o)$ denote the string extracted for object o by d and r^*, respectively. We then define the following *redundancy score* as a distance function over pairs of values:

$$\text{score}(v_1, v_2) = \begin{cases} 0 & \text{, if } v_1 \text{ substring of } v_2 \text{ or vice versa} \\ JS'(v_1, v_2) & \text{, otherwise} \end{cases}$$

where $JS'(v_1, v_2)$ is a *Jensen-Shannon* string distance modified for handling *nil*. If both v_1 and v_2 are *nil*, it returns 0. If either is *nil*, but not the other, it returns 1. In all other cases, it's a standard string distance. We noticed that the change of formats for the values of an attribute within the same site are very rare in real sites as they also tend to confuse the end user, i.e., the Price attribute is displayed as `1.5k` on a the result record whereas it is presented as `1,500` on the corresponding detail page of the running example in Figure 1. Hence, we observed only negligibly different results by adopting any other popular string distance function [7]. In practice, dealing with the substring cases is more important than the choice of the string distance function.

Example 4.4. Table 2 reports all the pairs scoring less than 0.4 ordered by their redundancy score for our running example. Each pair is composed of an aligned result rule from Table 1, and a detail rule from Figure 5b. The column "*Attributes*" reports whether the rules of the pair are correct, noisy, partially correct, or mis-aligned.

Table 2: Pairs of rules scoring less than 0.4.

pair π	red(π)	*Attributes*
(r_2, d_1)	0	Type correct
(r_6, d_4)	0	Location correct
(r_2, d_6)	0	Type vs Description
(r_8^b, d_7)	0	mis-alig. Location vs partial City
(r_8^a, d_5)	0	partial Location vs partial Location
(r_9, d_6)	0	Beds vs Description
(r_5^a, d_4)	0.333	noisy Location vs Location
(r_0^b, d_0)	0.397	Price correct

5 NOISE REMOVAL

RED uses the redundancy score to filter pairs of rules that are somewhat, but not sufficiently, redundant. Given a redundancy score threshold ρ, we consider as *not redundant* all the pairs of rules having a redundancy score greater than ρ. Unfortunately, many of the remaining pairs can still contain noisy rules: On one hand, there could be an incorrect alignment of the values, due to the presence of too many similar values in result records, e.g., (r_8^b, d_7) in Table 2; on the other hand, even noisy rules happen to be incidentally similar sometimes, e.g., (r_8^a, d_5) and (r_5^a, d_4). This is especially true when the range of the possible values of the compared attributes is rather limited; for example (the number of) Rooms, Beds, and Baths in the real estate domain are all small positive integers. Generally speaking, the correct result rules cannot be trivially separated from the incorrect ones with any fixed value of the threshold.

RED processes the redundant pairs by ascending redundancy score to remove the noisy pairs. First, the *result rule validation* analyzes whether a result rule has been correctly aligned and therefore properly fits in the HTML template of the result pages. Then, the *noise redundancy removal* leverages the redundancy scores to select for each attribute only the best pairs of rules, i.e., those having the lowest redundancy score.

5.1 Result Rules Validation

The result rules validation technique is based on the availability of some correctly aligned result rules as output of the previous processing steps. Beside the link rule, which is assumed to be correct, several other result rules might have only one alignment after the soft segmentation process, for example r_0^b and r_2 in Figure 1. RED leverages these already aligned and therefore inherently validated rules to validate other result rules.

A result rule r is validated against an already correctly-aligned rule g that is assumed correct. We can thereby infer two possible traversal sequences of their nodes corresponding to two positional alignments, and namely: s^b – extracted nodes of r are before those of g; and s^a – extracted nodes of r are after those of g. Then, as described in Section 4.1, we get the real traversal sequence s by traversing the DOM tree. If s does not matches any of s^b and s^a, r is not a valid result rule. We thereby remove r and all the redundant pairs containing r.

Example 5.1. For validating r_8^b based on r_9, RED considers two possible alignments of the values extracted by r_8^b w.r.t. r_9:
⟨3, 4, W2U 0DF..., 2, W3U 1AF..., 3, 1, E15 2CD..., 2, E15 2XD...⟩
⟨3, W2U 0DF..., 4, W3U 1AF..., 2, 3, E15 2CD..., 1, E15 2XD..., 2⟩ .
Neither of these matches the actual sequence in which these values occur on the page: ⟨W2U 0DF..., 3, W3U 1AF..., 4, 2, E15 2CD..., 3, E15 2XD..., 1, 2⟩.
Therefore, r_8^a fails to validate w.r.t. r_9 and is dropped.

5.2 Removing Redundant Noise

Once all the result rules have been validated, the surviving pairs include correct pairs together with partially correct and noisy redundant pairs that are incidentally similar.

The key assumption to remove the noisy pairs in the first category is that correct pairs have a better redundancy score than pairs containing noisy rules. The assumption is exploited in the first step of the algorithm removeRedNoise shown in Listing 1 (lines 2-9) and dealing with already validated result rules.

The second step of the algorithm (lines 10-17) prefers pairs of rules extracting fewer *nil* values: at this stage, all the noisy pairs have already been removed so that we can directly remove the result rules extracting a larger number of *nil*s. Notice that with the exception of the initial ordering (line 11 vs line 3), the two steps are performed by two almost identical loops (lines 2-9 vs lines 10-17). However, they cannot be merged because the latter loop expects as

Listing 1: removeRedNoise

```
Input: A set of redundant pairs Π = {(r, d) : red(r, d) < ϵ}.
Output: A set of correct rule pairs.
1  begin
2      Π′ ← ∅ ;
3      order pairs in Π by increasing redundancy score;
       /* Remove pairs with overlapping result DOM nodes */
4      foreach (r, d) ∈ Π do
5          remove (r, d) from Π ;
6          foreach (r′, d′) ∈ Π do
7              if DOMoverlap(r, r′) then
8                  remove (r′, d′) from Π ;
9          add (r, d) to Π′ ;                    /* saves only the best pair. */
       /* Prefer pairs whose detail rule extracts less nils.  */
10     Π″ ← ∅ ;
11     order Π' by decreasing n. of nil in the detail rules;
12     foreach (r, d) ∈ Π′ do
13         remove (r, d) from Π′ ;
14         foreach (r′, d′) ∈ Π′ do
15             if DOMoverlap(r, r′) then
16                 remove (r′, d′) from Π′ ;
17         add (r, d) to Π″ ;
18     return Π″;
```

input a list of pairs from which the noisy rules have been already removed, as produced by the first loop.

After the two processing steps, the surving pairs are the output result rules that RED classifies as correct.

Example 5.2. Consider again two pairs involving d_4 in Example 4.4, i.e., (r_6, d_4) and (r_5, d_4). The first step of removeRedNoise (lines 2-9) removes the latter pair, i.e., the pair with the worst redundancy score, since r_5 and r_6 extract overlapping nodes and only one of them is assumed to be correct.

Both pairs (r_6, d_4) and (r_8^a, d_5) are 0-scored and r_6 and r_8^a are two overlapping result rules: at most one of them can be correct. Both pairs reach the second loop (lines 10-17). The (r_8^a, d_5) pair is then removed because it contains the largest number of *nil* elements.

Table 3: Pair of rules after RED's noise removal step.

pair π	red(π)	note
(r_2, d_1)	0	Type
(r_6, d_4)	0	Location
(r_2, d_6)	0	Type vs Description
(r_9, d_6)	0	Beds
(r_0^b, d_0)	0.36	Price

Example 5.3. Table 3 reports (ordered by the redundancy score) the pairs after all the steps of the noise removal process. The pairs (r_2, d_1) and (r_2, d_6) are both 0-scored and the correct rule r_2 for Type is included in the output for its redundancy with the Description, an attribute on the detail pages that often "includes" the Type as a substring. So RED output on the running example is the set of aligned result rules r_0^b, r_2, r_6 and r_9 together with the link rule l.

6 EVALUATION

We inspect RED's characteristics and performance through a series of experiments on real-world websites. We first evaluate RED on a dataset derived from the one used in DIADEM [18], a state-of-the-art ontology-based data extraction system. We then apply RED to another dataset comprised of websites from a variety of application domains, a challenging setting for ontology-based data extraction systems that require some supervision on the level of the application domain. We show that, in contrast, RED can attain consistently high precision and recall over many domains.

As for the comparison with other fully unsupervised approaches beside DIADEM, we also considered DEPTA [41], another unsupervised system that, to the best of our knowledge, deals with the most similar setting to ours: unsupervised extraction from result pages, even a single result page.

6.1 Evaluation method

Datasets: We evaluate RED on two datasets, named DIA_DS and RED_DS, including 130 websites in total. DIA_DS consists of 100, still reachable websites from the original DIADEM's dataset, half in the REAL_ESTATE, half in the USED_CARS domain. We use DIADEM to generate wrappers on these sites and collect the detail pages for the dataset, as needed by RED.

RED_DS consists of 30 sites from 10 domains, obtained by randomly picking 3 sites from the *Alexa Top* 100 *Global Sites*. We exclude sites where the detail pages are not hosted on the site itself but rather refer to other sites (typically metasearch portals), as these cases do not fit RED's assumption that detail pages share a common template. Each test-case is related to a single website and it consists of one result page and the corresponding set of detail pages. All the records listed in the result pages are obtained by following the default sorting criteria from the website. Both datasets can be found on https://github.com/redwww/experiments.

Optional Attributes: To validate the need for dealing with optional attributes in result pages, we observed that in our datasets there is a considerable number of sites that contain optional attributes. In detail, 17% of sites in DIA_DS (i.e., 20 sites) and 36% of those in RED_DS (i.e., 13 sites) contain optional attributes. This well illustrates the importance of approaches, such as the soft segmentation technique, dealing with optional attributes in result pages.

Metrics: For each attribute of the result pages in our datasets, we manually crafted the correct XPath expression to extract correct data as golden standard, based on which we computed the number of true positives (tp), false negatives (fn), false positives (fp), precision $P = tp/(tp + fp)$, recall $R = tp/(tp + fn)$, and the F_1−measure, as $F = (2 \cdot P \cdot R)/(P + R)$.

These metrics have been computed both at (macro or) rule level and at (micro) level of extracted values. At rule-level, we consider a rule correct only if it extracts a set of values perfectly matching with that extracted by the golden rule, without any missing value: any rule extracting just a few noisy values or missing a single value, is considered the same as if it were completely wrong. We also calculate all the metrics at value-level by comparing the number of correct/noisy extracted values. In Table 4 and in Table 5 we report the main results for the two considered datasets: P, R, F stand for the precision, recall and F-measure, reported both at rule-level (e.g., F_r) and at value-level (e.g., F_v).

Table 4: RED vs DIADEM Performance.

System		DIADEM	RED	RED
Target Attributes		Ontology	Ontology	All
REAL_ESTATE	P_r	**0.94**	0.91	0.92
	R_r	**0.90**	**0.90**	0.91
	F_r	**0.92**	0.91	0.91
	P_v	**0.98**	**0.98**	0.98
	R_v	0.92	**0.94**	0.93
	F_v	0.95	**0.96**	0.96
USED_CARS	P_r	**0.97**	0.96	0.97
	R_r	**0.96**	**0.96**	0.96
	F_r	**0.96**	**0.96**	0.97
	P_v	**0.97**	0.96	0.97
	R_v	0.94	**0.98**	0.97
	F_v	0.95	**0.97**	0.97

6.2 Comparison with DIADEM

For a fair comparson between RED and DIADEM, we limit the target attributes to those in the ontology used by DIADEM, but also report RED's performance on all available attributes.

Table 4 presents $P/R/F$ results of RED and DIADEM systems over the dataset DIA_DS, both at the rule-level and at the value-level. Generally, RED and DIADEM perform roughly at the same high precision, recall, and F-measure. Considering that RED uses no domain ontology or other prior knowledge about the domain, that is quite remarkable. Moreover, RED can extract *all* the redundant attributes published in the result pages (as shown in the ALL column), not just the ones covered by the DIADEM ontology, achieving similar results. Overall, RED identifies 33% more attributes (about 800 attribute extraction rules) than DIADEM (about 600). The same close performance can be observed at site-level: RED extracts *all* attributes correctly in 73% of the websites, compared to 76% for DIADEM.

6.2.1 Failures & Limitations. The most frequent errors for RED and DIADEM systems can be grouped in two main categories.

Noise: DIADEM leverages its ontology to automatically annotate target pages. Wrong extracted values from DIADEM system are mainly due to misleading annotations. In our experiments, there are 5 sites where it suffers of this problem, e.g., top-lettings.co.uk. DIADEM extracts the template nodes `Available:` always occurring before the attribute AvailableDate and it wrongly takes it as a value of the attribute PropertyStatus. In contrast, RED's most common type of noise are partially correct rules, that extract only a fraction of the correct values.

Missing rules: Where DIADEM mostly misses attributes due to a lack of overlap with the background ontology, RED mostly misses attributes where the sites do not conform to its assumptions and specifically that result attributes also occur on the detail page and that attributes have a large enough domain to show some variability between result records. On site davidtompkins.co.uk the Description values in the result pages do not occur in detail pages. On cotswoldlettings.co.uk the Status of the properties is an optional attribute that can only take the value `SOLD` when present. RED wrongly considers all these values as part of the template.

6.3 Multi-domain evaluation

We further evaluate RED on RED_DS, a dataset covering 10 domains (Coffee, Concerts, Florist, Jewerlry, Job Search, Threatres, Books, Camera, Lighting, Sigars) with 3 randomly selected sites for each application domain. For space reasons, in Table 5 we just report the results for one example domain Cigars,[5] the quality metrics averaged over all the sites computed both at rule and value level, and other aggregated results; namely: total number of records (n) contained in the result page, total number of correct (tp), missing (fn) and incorrect (fp) values/rules produced.

Overall, RED attained a high performance at both rule-level (0.93 of F-measure) and value-level (0.96 of F-measure). At the value-level the precision is remarkable high (0.99).

Among 8 noisy rules in the output produced by RED, only 3 of them are entirely incorrect while all the other noisy rules are partially correct rules. On ncfjobs site, RED seems to perform badly as there are 4 noisy rules, out of which 3 are partially correct rules for Description, and 1 for JobTitle. All these noisy rules survived the noise-removal step because RED rules generation algorithm was not complete, i.e., it could not find correct rules for these attributes.

6.3.1 Comparison with other unsupervised approaches. For comparison with state-of-the-art unsupervised data extraction systems, we ran DEPTA [41] on the dataset RED_DS. This system is also capable of extracting records from result pages and does not require multiple sample pages. Although the algorithm was proposed in 2006, we compared with a runnable version re-implemented in 2012.[6] Since it blindly outputs multiple tables without telling the user which one contains the target data, we manually picked a single table containing the published records and evaluated precision, recall and F-measure at the *column*-level, to get scores comparable with RED's rule-level scores. This yields a negligible advantage for DEPTA as it is usually fairly easy to pick out the dominant table.

We evaluate DEPTA's record identification step over 14 out of 30 sites considered on which the prototype was able to produce a correctly-aligned table. Among these 14 sites, DEPTA achieves a respectable precision of 0.70 and recall of 0.93.

6.4 Robustness to parameter setting

RED depends on two key parameters: δ is the max allowed pivot-to-value distance used during the generation of the rules; ρ is the minimum threshold on the score to consider a pair of result/detail rules as not redundant during the search for correct pairs of rules.

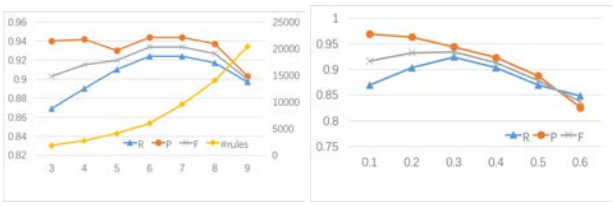
Figure 6a plots the performance of RED on the RED_DS dataset over several δ values (with fixed $\rho = 0.3$). While the precision is only slightly affected, the recall is significantly related to the maximum pivot-to-value distance. The larger the number of generated rules, the higher is the probability of generating correct rules. Unfortunately, it turns out that by generating too many rules, RED may end up introducing so many noisy rules that the probability of a noisy rule to be erroneously considered correct is increased, causing a loss in precision. In our experiments, $\delta = 6$ proved to be the sweet spot.

The other key parameter is the maximum redundancy score, ρ. A pair of result/detail rules is considered not redundant if its redundancy score is above ρ. It significantly affects the noise removal step whose task becomes harder and harder as noisy pairs are injected in the RED's processing pipeline. As shown in Figure 6b (with fixed

[5] A complete table can be found at https://github.com/redwww/experiments
[6] http://seagatesoft.blogspot.it/2012/05/structured-data-extractor.html

Table 5: Results of RED on RED_DS.

Domain	*Site*	n	tp_r	fn_r	fp_r	P_r	R_r	F_r	tp_v	fn_v	fp_v	P_v	R_v	F_v
Cigars	famous-smoke	60	8	1	1	0.88	0.88	0.88	439	15	20	0.95	0.97	0.96
	cigars-of-cuba	9	9	0	1	0.90	1	0.95	71	0	6	0.92	1	0.96
	vipcigars	18	3	0	0	1	1	1	54	0	0	1	1	1
10 domains	*30 sites*	*724*	*145*	*11*	*8*	*0.94*	*0.92*	*0.93*	*3365*	*248*	*28*	*0.99*	*0.93*	*0.96*



(a) Max pivot-value distance δ **(b) Max redundancy score ρ**

Figure 6: Performance by varying the two key parameters.

$\delta = 6$) the precision starts with maximum values, and then it gradually decreases as ρ becomes larger and larger. The noise removal process can still manage to remove most of the noises when ρ is at 0.4 but for larger values of the parameter there is a loss of precision, as the greater number of noisy pairs makes that process less effective. Conversely, too small values for ρ, meaning a too strict criteria for considering a pair of rules as redundant, makes RED wrongly remove some correct but not perfectly redundant pairs, as is evidenced by the recall loss of the plot in Figure 6b at $\rho < 0.3$.

For RED experimental evaluation, we measured RED performance by using the best empirically known values for the two parameters δ and ρ, i.e., $\rho = 0.3$ and $\delta = 6$.

Efficiency: By considering the number of input pages as a constant, RED efficiency can be conveniently analyzed in term of the initial number of rules generated. On RED_DS dataset, it generates an average number of 1259.17 result rules and 4912.10 detail rules, which reduce to 205.50 and 289.03 distinct vectors, respectively, after duplicates removal step; only 88.87 candidate pairs have a redundancy score smaller than ρ. RED achieves an average running time of 72.6 secs on a 64-bit Ubuntu system with an 8-core Intel *i*7 CPU at 3.40GHz, processing most sites in less than 30 seconds. RED running time is spent mostly on executing the generated extraction rules, a step that could be easily parallelised.

7 RELATED WORK

Web data extraction approaches and tools has been proposed both from research community and from industry [15, 16]. Due to the space limitation, we compare RED with the most related work and systems such as DEPTA [41], and DIADEM [18], which tackle with a similar problem as ours.

DEPTA executes two steps on a single result page, (*i*) identifying data records, and (*ii*) aligning values belonging to the same attribute. The first step, which is an enhancement of the MDR algorithm [29], works on a tag tree based on the visual containment relations of different HTML elements. It exploits the similarity of tag strings of nodes (and their descendant nodes) to find the data records. A further partial tree alignment method based on tree-edit distance is applied for the second step. DEPTA has two major drawbacks: First, it is highly sensitive even to small exceptions in the structure of the template (e.g., the record of some featured products may be presented differently from others) and optional nodes (e.g., the discount) in the repeated patterns. Second, due to the lack of any cues for identifying target values, it cannot easily separate target values from template nodes and other noises in the input pages.

One of the state-of-the-art system for domain-specific data extraction from result pages is DIADEM. We limit our discussion to its component most related to RED, i.e., that extracting data from result pages. DIADEM adopts automatic annotators to improve the data identification process, and requires the writing of a domain-dependent ontology which has to be designed and maintained by an expert. We exploit the intra-site highly precise redundancy to make RED more accurately target the relevant data without using any domain dependent feature.

Unsupervised approaches like RoadRunner [8], ExAlg [2], FiVaTech [27], TRINITY [35], and ViDE [30] all are based on the observation that pages providing homogeneous data are generated by using regular HTML templates, so they end up being mostly similar. It turns out that by analyzing the differences among a bunch of pages sharing a common template, it is then possible to reverse-engineer a model of that template, for example by means of regular expressions [2, 9]. However, in practice all these systems are not used for large scale extraction tasks because their performances are highly dependent on how well the input pages satisfy the underlying assumptions, to the point that the output quality level is unpredictable at scale [10]. In contrast, RED does not infer a precise description of the template, and it does not rely on it to find the data as all these latter systems, including ExAlg, do.

The work in [11, 12, 22] addresses the problem of XPath rules generation problem, but starts from a different input, i.e., a set of example nodes most of which are likely correct target nodes. These are usually provided by automatic annotators which are not available for every domain.

The approaches in [5, 23] and in [3] exploit content and/or schema redundancy and focus on template-based sites as RED does. However, they solve a completely different set of problems by adopting techniques based on the availability of data about the same objects across *several* autonomous sites. In other words, they focus on *inter*-site redundancy rather than on *intra*-site redundancy. It is therefore crucially important to align the different formats (e.g., Height expressed in inches on a site vs the same attribute expressed in centimeters on another site), a problem that we show is negligible intra-site. Also, since both approaches work directly on detail pages, they do not address record segmentation problems at all.

In [28] it is tackled the problem of segmenting result pages into records by exploiting links to detail pages exactly as RED. However, differently from RED, the authors do not consider at all the problem of extracting data from the result records once segmented; rather they describe two different approaches for tackling the segmentation problem: one formulates the task as a constraint satisfaction problem (CSP) and the other uses a probabilistic inference setting.

REFERENCES

[1] Manuel Álvarez, Juan Raposo, Alberto Pan, Fidel Cacheda, Fernando Bellas, and Víctor Carneiro. 2007. Crawling the content hidden behind web forms. In *Proc. of ICCSA*. 322–333.

[2] Arvind Arasu and Hector Garcia-Molina. 2003. Extracting Structured Data from Web Pages. In *SIGMOD Conference*. 337–348.

[3] Mirko Bronzi, Valter Crescenzi, Paolo Merialdo, and Paolo Papotti. 2013. Extraction and Integration of Partially Overlapping Web Sources. *PVLDB* 6, 10 (2013), 805–816. http://www.vldb.org/pvldb/vol6/p805-bronzi.pdf

[4] Andrea Calì and Davide Martinenghi. 2010. Querying the deep web. In *Proc. of EDBT*. 724–727.

[5] Shui-Lung Chuang, Kevin Chen-Chuan Chang, and ChengXiang Zhai. 2007. Collaborative Wrapping: A Turbo Framework for Web Data Extraction. In *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*, Rada Chirkova, Asuman Dogac, M. Tamer Özsu, and Timos K. Sellis (Eds.). IEEE Computer Society, 1261–1262. https://doi.org/10.1109/ICDE.2007.368988

[6] Shui-Lung Chuang, Kevin Chen-Chuan Chang, and ChengXiang Zhai. 2007. Context-aware Wrapping: Synchronized Data Extraction. In *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB '07)*. VLDB Endowment, 699–710. http://dl.acm.org/citation.cfm?id=1325851.1325931

[7] William W. Cohen, Pradeep Ravikumar, and Stephen E. Fienberg. 2003. A Comparison of String Distance Metrics for Name-Matching Tasks. In *Proceedings of IJCAI-03 Workshop on Information Integration on the Web (IIWeb-03), August 9-10, 2003, Acapulco, Mexico*, Subbarao Kambhampati and Craig A. Knoblock (Eds.). 73–78. http://www.isi.edu/info-agents/workshops/ijcai03/papers/Cohen-p.pdf

[8] Valter Crescenzi and Giansalvatore Mecca. 2004. Automatic Information Extraction from Large Websites. *J. ACM* 51, 5 (2004), 731–779.

[9] Valter Crescenzi and Paolo Merialdo. 2008. Wrapper Inference for Ambiguous Web Pages. *Applied Artificial Intelligence* 22, 1&2 (2008), 21–52.

[10] Valter Crescenzi, Paolo Merialdo, and Disheng Qiu. 2015. Crowdsourcing large scale wrapper inference. *Distributed and Parallel Databases* 33, 1 (2015), 95–122. https://doi.org/10.1007/s10619-014-7163-9

[11] Nilesh N. Dalvi, Philip Bohannon, and Fei Sha. 2009. Robust web extraction: an approach based on a probabilistic tree-edit model. In *Proc. of SIGMOD*. 335–348.

[12] Nilesh N. Dalvi, Ravi Kumar, and Mohamed A. Soliman. 2011. Automatic Wrappers for Large Scale Web Extraction. *PVLDB* 4, 4 (2011), 219–230.

[13] Steven DeRose and James Clark. 1999. *XML Path Language (XPath) Version 1.0*. W3C Recommendation. W3C. http://www.w3.org/TR/1999/REC-xpath-19991116.

[14] Xin Luna Dong and Divesh Srivastava. 2013. Big Data Integration. *Proc. VLDB Endow.* 6, 11 (Aug. 2013), 1188–1189. https://doi.org/10.14778/2536222.2536253

[15] Ruslan R. Fayzrakhmanov, Emanuel Sallinger, Ben Spencer, Tim Furche, and Georg Gottlob. 2018. Browserless Web Data Extraction: Challenges and Opportunities. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1095–1104. https://doi.org/10.1145/3178876.3186008

[16] Emilio Ferrara, Pasquale De Meo, Giacomo Fiumara, and Robert Baumgartner. 2014. Web data extraction, applications and techniques: A survey. *Knowl.-Based Syst.* 70 (2014), 301–323. https://doi.org/10.1016/j.knosys.2014.07.007

[17] Tim Furche, Georg Gottlob, Giovanni Grasso, Xiaonan Guo, Giorgio Orsi, and Christian Schallhart. 2012. OPAL: automated form understanding for the deep web. In *Proceedings of the 21st international conference on World Wide Web (WWW '12)*. ACM, New York, NY, USA, 829–838. https://doi.org/10.1145/2187836.2187948

[18] Tim Furche, Georg Gottlob, Giovanni Grasso, Xiaonan Guo, Giorgio Orsi, Christian Schallhart, and Cheng Wang. 2014. DIADEM: Thousands of Websites to a Single Database. *PVLDB* 7, 14 (2014), 1845–1856. http://www.vldb.org/pvldb/vol7/p1845-furche.pdf

[19] Tim Furche, Georg Gottlob, Giovanni Grasso, Giorgio Orsi, Christian Schallhart, and Cheng Wang. 2011. Little Knowledge Rules The Web: Domain-Centric Result Page Extraction. In *Proc. of RR*. 61–76.

[20] Tim Furche, Georg Gottlob, Giovanni Grasso, Giorgio Orsi, Christian Schallhart, and Cheng Wang. 2012. AMBER: Automatic Supervision for Multi-Attribute Extraction. *CoRR* abs/1210.5984 (2012).

[21] Tim Furche, Giovanni Grasso, Andrey Kravchenko, and Christian Schallhart. 2012. Turn the page: automated traversal of paginated websites. In *International Conference on Web Engineering*. Springer, 332–346.

[22] Tim Furche, Jinsong Guo, Sebastian Maneth, and Christian Schallhart. 2016. Robust and Noise Resistant Wrapper Induction. In *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, Fatma Özcan, Georgia Koutrika, and Sam Madden (Eds.). ACM, 773–784. https://doi.org/10.1145/2882903.2915214

[23] Pankaj Gulhane, Rajeev Rastogi, Srinivasan H. Sengamedu, and Ashwin Tengli. 2010. Exploiting content redundancy for web information extraction. In *Proc. of WWW*. 1105–1106.

[24] Bin He, Mitesh Patel, Zhen Zhang, and Kevin Chen-Chuan Chang. 2007. Accessing the Deep Web. *Commun. ACM* 50, 5 (May 2007), 94–101. https://doi.org/10.1145/1230819.1241670

[25] Yeye He, Dong Xin, Venkatesh Ganti, Sriram Rajaraman, and Nirav Shah. 2013. Crawling Deep Web Entity Pages. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining (WSDM '13)*. ACM, New York, NY, USA, 355–364. https://doi.org/10.1145/2433396.2433442

[26] Arnaud Le Hors, Philippe Le Hégaret, et al. 2004. *Document Object Model (DOM) Level 3 Core Specification*. W3C Technical Reports. W3C. https://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/.

[27] Mohammed Kayed and Chia-Hui Chang. 2010. FiVaTech: Page-Level Web Data Extraction from Template Pages. *IEEE Trans. Knowl. Data Eng.* 22, 2 (2010), 249–263. https://doi.org/10.1109/TKDE.2009.82

[28] Kristina Lerman, Lise Getoor, Steven Minton, and Craig A. Knoblock. 2004. Using the Structure of Web Sites for Automatic Segmentation of Tables. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, Paris, France, June 13-18, 2004*, Gerhard Weikum, Arnd Christian König, and Stefan Deßloch (Eds.). ACM, 119–130. https://doi.org/10.1145/1007568.1007584

[29] B. Liu, R. Grossman, and Y. Zhai. 2003. Mining Data Records in Web Pages. In *Proc. of SIGKDD*. 601–605.

[30] Wei Liu, Xiaofeng Meng, and Weiyi Meng. 2010. ViDE: A Vision-Based Approach for Deep Web Data Extraction. *IEEE Trans. Knowl. Data Eng.* 22, 3 (2010), 447–460. https://doi.org/10.1109/TKDE.2009.109

[31] Jayant Madhavan, David Ko, Lucja Kot, Vignesh Ganapathy, Alex Rasmussen, and Alon Halevy. 2008. Google's Deep Web Crawl. In *Proc. of the VLDB Endowment (PVLDB)*. 1241–1252.

[32] Jayant Madhavan, David Ko, Łucja Kot, Vignesh Ganapathy, Alex Rasmussen, and Alon Halevy. 2008. Google's deep web crawl. *PVLDB* 1, 2 (2008), 1241–1252.

[33] Ali Mesbah, Arie van Deursen, and Stefan Lenselink. 2012. Crawling Ajax-Based Web Applications through Dynamic Analysis of User Interface State Changes. *ACM Trans. Web* 6, 1, Article 3 (March 2012), 30 pages.

[34] Stefano Ortona, Giorgio Orsi, Marcello Buoncristiano, and Tim Furche. 2015. WADaR: Joint Wrapper and Data Repair. *PVLDB* 8, 12 (2015), 1996–2007. http://www.vldb.org/pvldb/vol8/p1996-ortona.pdf

[35] Hassan A. Sleiman and Rafael Corchuelo. 2014. Trinity: On Using Trinary Trees for Unsupervised Web Data Extraction. *IEEE Trans. Knowl. Data Eng.* 26, 6 (2014), 1544–1556. https://doi.org/10.1109/TKDE.2013.161

[36] Karane Vieira, Luciano Barbosa, Altigran Soares da Silva, Juliana Freire, and Edleno Silva de Moura. 2016. Finding seeds to bootstrap focused crawlers. *World Wide Web* 19, 3 (2016), 449–474. https://doi.org/10.1007/s11280-015-0331-7

[37] Jiying Wang and Fred H. Lochovsky. 2003. Data extraction and label assignment for web databases. In *Proc. of WWW*. 187–196.

[38] Jiying Wang, Ji-Rong Wen, Fred Lochovsky, and Wei-Ying Ma. 2004. Instance-based Schema Matching for Web Databases by Domain-specific Query Probing. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30 (VLDB '04)*. VLDB Endowment, 408–419. http://dl.acm.org/citation.cfm?id=1316689.1316726

[39] Lin Wang, Ammar Hawbani, and Xingfu Wang. 2015. Focused Deep Web Entrance Crawling by Form Feature Classification. In *Big Data Computing and Communications*, Yu Wang, Hui Xiong, Shlomo Argamon, XiangYang Li, and JianZhong Li (Eds.). Springer International Publishing, Cham, 79–87.

[40] Ying Wang, Wanli Zuo, Tao Peng, and Fengling He. 2008. Domain-Specific Deep Web Sources Discovery. In *Fourth International Conference on Natural Computation, ICNC 2008, Jinan, Shandong, China, 18-20 October 2008, Volume 5*, Maozu Guo, Liang Zhao, and Lipo Wang (Eds.). IEEE Computer Society, 202–206. https://doi.org/10.1109/ICNC.2008.350

[41] Yanhong Zhai and Bing Liu. 2006. Structured Data Extraction from the Web Based on Partial Tree Alignment. *TKDE* 18, 12 (2006), 1614–1628.

[42] Zhen Zhang, Bin He, and Kevin Chen-Chuan Chang. 2004. Understanding Web Query Interfaces: Best-effort Parsing with Hidden Syntax. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data (SIGMOD '04)*. ACM, New York, NY, USA, 107–118. https://doi.org/10.1145/1007568.1007583

[43] Feng Zhao, Jingyu Zhou, Chang Nie, Heqing Huang, and Hai Jin. 2016. SmartCrawler: A Two-Stage Crawler for Efficiently Harvesting Deep-Web Interfaces. *IEEE Transactions on Services Computing* 9 (2016), 608–620.

Securing the Deep Fraud Detector in Large-Scale E-Commerce Platform via Adversarial Machine Learning Approach

Qingyu Guo
Nanyang Technological University
qyguo@ntu.edu.sg

Zhao Li
Alibaba Group
lizhao.lz@alibaba-inc.com

Bo An
Nanyang Technological University
boan@ntu.edu.sg

Pengrui Hui, Jiaming Huang
Alibaba Group
{pengrui.hpr,jimmy.hjm}@
alibaba-inc.com

Long Zhang
Alibaba Group
james.zl@alibaba-inc.com

Mengchen Zhao
Nanyang Technological University
zhao0204@ntu.edu.sg

ABSTRACT

Fraud transactions are one of the major threats faced by online e-commerce platforms. Recently, deep learning based classifiers have been deployed to detect fraud transactions. Inspired by findings on adversarial examples, this paper is the first to analyze the vulnerability of deep fraud detector to slight perturbations on input transactions, which is very challenging since the sparsity and discretization of transaction data result in a non-convex discrete optimization. Inspired by the iterative Fast Gradient Sign Method (FGSM) for the L_∞ attack, we first propose the *Iterative Fast Coordinate Method (IFCM)* for discrete L_1 and L_2 attacks which is efficient to generate large amounts of instances with satisfactory effectiveness. We then provide two novel attack algorithms to solve the discrete optimization. The first one is the *Augmented Iterative Search (AIS)* algorithm, which repeatedly searches for effective "simple" perturbation. The second one is called the *Rounded Relaxation with Reparameterization (R3)*, which rounds the solution obtained by solving a relaxed and unconstrained optimization problem with reparameterization tricks. Finally, we conduct extensive experimental evaluation on the deployed fraud detector in TaoBao, one of the largest e-commerce platforms in the world, with millions of real-world transactions. Results show that (i) The deployed detector is highly vulnerable to attacks as the average precision is decreased from nearly 90% to as low as 20% with little perturbations; (ii) Our proposed attacks significantly outperform the adaptions of the state-of-the-art attacks. (iii) The model trained with an adversarial training process is significantly robust against attacks and performs well on the unperturbed data.

CCS CONCEPTS

• **Information systems** → Adversarial retrieval.

KEYWORDS

Online Shopping; Fraud Detection; Adversarial Machine Learning

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313533

ACM Reference Format:
Qingyu Guo, Zhao Li, Bo An, Pengrui Hui, Jiaming Huang, Long Zhang, and Mengchen Zhao. 2019. Securing the Deep Fraud Detector in Large-Scale E-Commerce Platform via Adversarial Machine Learning Approach. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313533

1 INTRODUCTION

With the rapid growth of information technologies, e-commerce has become prevalent nowadays. Large e-commerce platforms serve billions of users and connect them to factories, stores and third-party merchants with numerous products available. The big success of e-commerce also motivates the emergence of fraud transactions to illegally promote items and stores. In order to increase sales, fraudulent sellers turn to third-party malicious service platforms where they can hire human labors to create fake purchases and visits [26, 29, 33], and thus provide a fake impression that the target item is popular. This results in much higher rankings of the items from fraudulent sellers, and causes huge losses to the platforms as it hurts the user experience. Online fraud activities have caused a loss in billions of dollars[1], and it is reported that just in China, the online fraudulent industries involve over 1.6 million human labors and create billions of dollars of illegal incomes[2].

Fraud detection is critical to the development of e-commerce platforms. Recently, with billions of transaction data available, deep learning based models are proposed and deployed to detect fraud transactions on a real-time basis [6, 32, 34]. Inspired by recent findings in the domain of computer vision that deep learning based image classifiers could be fooled by imperceptible noises during test phases [27], we are interested in analyzing the vulnerability of deep fraud detector to slight perturbations on input data. This study has a high practical value for e-commerce platforms as the existence of such adversarial perturbations could indicate the risk of deploying deep fraud detectors. As shown in our motivating example, fraudulent sellers and malicious service providers are able to exploit such risks to bypass the detector with little costs.

In this paper, we aim to investigate the vulnerability of a deployed deep fraud detector being fooled by slight and feasible perturbations. However, the e-commerce domain is different from the computer vision area, since the transaction data are mostly discrete and sparse.

[1] https://www.signifyd.com/blog/2017/10/26/ecommerce-fraud-eight-industries/
[2] http://cn.chinadaily.com.cn/2015-11/05/content_22379527.htm

Few works considering discrete adversarial perturbations either propose a straightforward extension of the existing Fast Gradient Method [9], or take into account only the L_∞ attack [2], while they cannot handle more complex L_1 and L_2 attacks, which are of higher practical interests in our problem.

This paper makes the following contributions.

- First, inspired by the Fast Gradient Sign Method (FGSM), to handle the L_1 and L_2 norm bound constraints, we propose the Iterative Fast Coordinate Method (IFCM) which repeatedly conducts a one-unit descent on a single coordinate. IFCM can efficiently generate a large amount of adversarial perturbations with satisfactory effectiveness and is used in our proposed adversarial training process.
- Second, to address the discrete optimization for adversarial perturbations, we provide two novel attacks. The first attack is called the *Augmented Iterative Search (AIS)*. In each iteration, AIS proceeds to search for "simple" perturbations with one-unit descents on at most two coordinates. AIS also augments the searching process by restricting the space of two-coordinate perturbations based on the *Value of Perturbation (VoP)*. The second attack is called the *Rounded Relaxation with Reparameterization (R3)*, which is an optimization-based attack. R3 first relaxes the discrete optimization with reparameterization trick and rounds the continuous solution to obtain a discrete perturbation.
- Third, we conduct extensive experimental evaluations on the deployed fraud detector from TaoBao, one of the largest e-commerce platforms in the world, with millions of real-world transactions. The experimental results show that (i) The deployed fraud detector is threatened by the crafted adversarial perturbations. Our AIS and R3 attacks successfully decrease the detector's average precision from nearly 90% to as low as 20% with little perturbations. (ii) Our proposed AIS and R3 attacks significantly outperform adaptations of several state-of-the-art attacks, including the C&W attack [4], the Logit-Space attack [2], and the EAD attack [5]. (iii) We propose an adversarial training process with adversarial examples generated by the IFCM. Results show that the robustness of the detection model is significantly improved as we achieve an average precision above 85.9% under all tested attacks. Meanwhile, the average precision on the unperturbed data remains nearly 90%.

2 RELATED WORK

In this section, we review the related works in crafting adversarial examples to attack DNN models and various defense techniques.

2.1 Attacks on DNNs

FGSM [8] and its variant R+FGSM [28] are single step attacks, which perturb the input towards the gradient direction of the loss function. The extensions of FGSM iteratively perform single step attacks with smaller step size, such as BIM [13] and PGD [15]. Other iterative attacks include DeepFool [17], which approximates the decision boundary near the target instance with the linear hyperplane, and UAP [16] to produce a single perturbation which is capable of attacking multiple images. FGSM and R+FGSM are further extended to perturb towards the direction which maximizes the confidence score on the least likely labels for the target image [8, 28].

Besides gradient directional methods, Papernot et al. [21] propose the JSMA attack which uses the Jacobian matrix to get the salient map and modifies the features with high significance to confuse the classifier. C&W attack [3] relaxes the hard constraint on the distortion, and adds a penalty on the L_p norm ($p \in \{0, 2, \infty\}$) of perturbation. EAD [5] improves the transferability of adversarial examples through elastic-net regularization. Several works study the adversarial examples to evade DNN models in real-world applications, including the face recognition system [25], objection detection [11], and malware detection [9].

Most of the works focus on image recognition tasks and consider only continuous perturbations. While Grosse et al. consider the binary input [9] and adapt the fast gradient algorithms to solve their problem in a similar way as our IFCM$_1$, they only consider the L_1 attack. Buckman et al. propose the Gumbel-softmax trick to approximate the one-hot encoding of the discrete feature. However, their attack is suitable for L_∞ distortion bound and fails to consider the constraints that correlate the input features such as L_1 and L_2 norm constraints in our problem.

Our R3 attack is inspired by the optimization-based L_2 attacks [4, 5, 27]. These attacks all handle the L_2 norm constraint by imposing a penalty in the objective function and conducting a binary search to find a suitable constant coefficient of penalty term. This is a reasonable way to find the minimal distortion to successfully craft an adversarial example. However, in our problem, we are asked to find the most effective adversarial instance given the allowed distortion. While one could conduct a binary search to find the suitable constant coefficient that the distortion constraint is satisfied, the neural networks are non-convex in general and thus the binary search could be unstable to always generate a distortion satisfying the given constraint [27]. Our R3 attack utilizes the reparameterization trick to transform the constrained optimization to an equivalent unconstrained problem for which the solution always satisfies the distortion bound.

2.2 Defenses

There are several defense measures to improve the robustness of deep learning based classifiers, such as the defensive distillation [20, 22], which adopts the distillation technique [10] to retrain the same network with the confidence score outputted by the original model as the soft label, feature squeezing [30], detection approaches for adversarial instances [4], and adversarial training [8], a general process based on robust optimization that augments the training data with adversarial examples [8, 15, 35]. Some recent works propose to regularize or mask the gradient of the loss function with respect to inputs [24]. Almost all defenses are shown to be effective only for some attacks [31].

Recently, Athalye et al. pointed out that the defenses based on obfuscated gradients are threatened by their characteristic behaviors which can be exploited to design attacks circumventing them [1]. It is believed that adversarial training does not cause obfuscated gradient [1]. Thus, in this paper, we use adversarial training to achieve robust fraud detection and our experimental evaluation verifies the effectiveness of the strengthened detector.

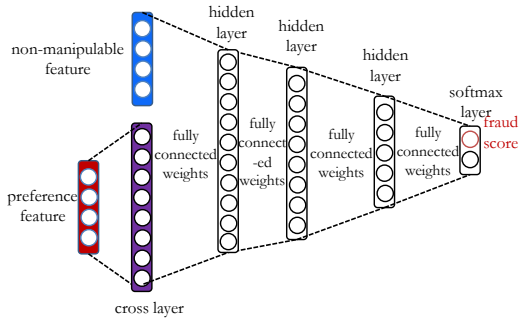


Figure 1: Abstracted architecture of deep fraud detector

```
  localFindResult.pos = paramInt1;
  localFindResult.tOrSubT = 1;
  return localFindResult;
}
j = this.bottomX;
int k = this.bottomY;
int m = randomInt(0, 100);
randomPos();
if (m < 30)
  getUiDevice().drag(j, k, this.topX, this.topY, randomInt(18, 30));
while ((randomInt(0, 100) < 50) && (findByEqual("没有更多宝贝啦") != null))
{
  localFindResult.yes = false;
  localFindResult.pos = (paramInt2 - 1);
  return localFindResult;
```

no more items

Figure 2: A piece of a fraudulent automatic script

3 DEEP FRAUD DETECTION AND ATTACKS

In this section, we first briefly introduce the deep learning model for fraud detection which is used by one of the largest e-commerce platforms in the world. We then present concrete examples to show the possible attacks by malicious sellers and service providers.

3.1 Deep Fraud Detection

Figure 1 shows the architecture of deep neural networks in deployed fraud detection system of TaoBao. The goal of the detector is to predict whether an online transaction is fraudulent. Each input transaction is represented as a feature vector $\mathbf{x}$ which concatenates two classes of features. The first is non-manipulable features, such as scores obtained with label propagation on the user-item transaction graph, to measure the offline maliciousness based on past records. Other non-manipulable features are prices[3]. In this paper, we focus on manipulatable features whose adjustment could have a significant effect on the detector's prediction. Without causing any confusion, we denote a transaction by a vector of only perturbable features. In the context of the investigated fraud detection, the perturbable features are those characterizing the user's preference, called *preference features*, defined as follows.

Let C be the set of all item categories, and $n = |C|$. A preference feature vector is represented by $\mathbf{x} \in \mathbb{N}^n$ with x_i denoting the number of records that the user interacts with items in the i-th category of C in a past period, normally 30 days. Such records include adding an item to the shopping cart, adding an item to "favorite", and so on. $\mathbf{x}$ represents the user's global interest in online shopping. Fraudulent buyers tend to show different preference patterns compared with benign users. Each transaction instance $\mathbf{x}$ is associated with a label $y \in \{0, 1\}$, such that $y = 1$ if the transaction is fraudulent and $y = 0$ otherwise.

All features are first normalized to be within $[0, 1]^n$ with a "smooth" normalization function to tackle the extreme values. Here, the feature value $\mathbf{x}$ is normalized to be $\frac{\log(\mathbf{x}+1)}{\log M}$ (element-wise) where M is the largest number of interactions observed in training data. The normalized preference features are passed to a cross layer [14] at first to exploit the interactions between different categories. The output of the cross layer is concatenated with non-manipulable features. The aggregated feature vector goes to a multi-layer feed-forward neural network, where each hidden-layer consists of rectified linear units as hidden neurons [18]:

$$relu(x) = \max(0, x).$$

The output of the last hidden-layer $\mathbf{z} \in \mathbb{R}^2$, called the *logit*, is then passed to a softmax layer to produce the confidence score on the transaction $\mathbf{x}$ being fraudulent

$$f^{\mathbf{w}}(\mathbf{x}) = \frac{e^{z_1}}{e^{z_0} + e^{z_1}},$$

where $\mathbf{w}$ denotes the parameters of the model, including the weights and biases on hidden neurons. The model is trained over a large-scale data set $\mathcal{D} = \{\mathbf{x}^i, y^i\}_{i=1}^N$ by minimizing the regularized total loss function $L^{\mathbf{w}}(\mathbf{x}, y)$ over $\mathcal{D}$

$$\min_{\mathbf{w}} \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} [L^{\mathbf{w}}(\mathbf{x}, y)] + \lambda \|\mathbf{w}\|^2, \tag{1}$$

where λ is the regularization coefficient. The commonly used loss function in classification tasks is the cross-entropy loss

$$L^{\mathbf{w}}(\mathbf{x}, y) = -\mathbb{1}_{\{y=1\}} \log f^{\mathbf{w}}(\mathbf{x}) - \mathbb{1}_{\{y=0\}} \log (1 - f^{\mathbf{w}}(\mathbf{x})),$$

and the stochastic gradient descent is widely applied to optimize the parameter $\mathbf{w}$ [7, 23].

3.2 Attacks to Fraud Detector

With billions of customers served on e-commerce platforms, the profit from fraud transactions could be huge for malicious sellers as they significantly promote the rankings of target items. Driven by the huge profit, a lot of malicious service platforms provide sophisticated tactics to bypass the strict detection and conduct fraud transactions. One common tactic is to mimic benign users' behavior such as using automatic visit script to visit different items before purchasing the target item, and hiring human labors to chat with the merchants before making a transaction. Figure 2 shows an example of automatic script to bypass the fraud detection, which is a piece of Java code as part of a mobile malicious service App which creates random perturbation to a target transaction such as dragging and browsing to behave like humans, intentionally visiting items from some categories, etc. According to domain experts, malicious platforms can create around 200,000 visits in per day and promote over 1000 items from malicious sellers. Since we verify that the vulnerability to slight perturbations exists for a fraud detector, the malicious platforms could exploit those threats to adapt the fraud transaction data in order to bypass the detection.

[3]Prices are hard to perturb in practice and the empirical study shows that slight perturbations on prices cause negligible changes.

Formally, let δ be the vector of perturbation on fraud instance $\mathbf{x}$ where δ_i is the amount of changes on x_i, and δ is discrete, i.e., $\delta \in \mathbb{Z}^n$. Assume $\delta \geq \mathbf{0}$ since it is more practical for realistic adversaries to add additional records of interactions via measures such as aforementioned automatic scripts[4]. In the domain of computer vision, the adversarial perturbations need to satisfy constraints that the perturbations are small enough, so that the ground truths of the adversarial instances remain the same as the original ones [8]. We also impose similar restrictions on δ, in order to capture the *feasibility* of perturbations in the sense that adversaries can conduct the perturbation with little effort. Such restrictions are expressed by the L_p norm bound on perturbations, and the common choices of p are $\{0, 1, 2, \infty\}$. Thus, the set of feasible δ bounded by L_p norm is defined as follows:

$$\Delta_q^p := \{\delta \mid \delta \in \mathbb{Z}^n, \|\delta\|_p \leq q, \delta \geq \mathbf{0}\}, \tag{2}$$

where q is the maximal distortion in L_p norm, and

$$\|\delta\|_0 = \sum_{i=1}^{n} \mathbb{1}_{\{\delta_i > 0\}}, \qquad \|\delta\|_1 = \sum_{i=1}^{n} |\delta_i|,$$

$$\|\delta\|_2 = \sqrt{\sum_{i=1}^{n} \delta_i^2}, \qquad \|\delta\|_\infty = \max_i |\delta_i|.$$

We focus on attacks bounded by L_1 and L_2 norms, which are suitable to describe realistic perturbations on the discrete and sparse preference features. L_0 attack is not practical as it allows the adversary to add infinitely many records on one category. L_∞ attack is not reasonable as the preference features are sparse in the sense that interactions on only few categories are recorded. Thus, even an adversarial perturbation with L_∞ norm of 1 violates the sparse nature of preference features.

For a suitable distortion metric L_p, our objective is defined as follows. For a deep fraud detector with model parameter $\mathbf{w}$, given a fraud transaction instance $\mathbf{x}$ in testing or deploying phase, we aim to find the adversarial example, an instance $\mathbf{x}^{adv}$ crafted with a feasible perturbation $\delta \in \Delta_q^p$, i.e., $\mathbf{x}^{adv} = \mathbf{x} + \delta$, to maximally decrease the fraudulent score on $\mathbf{x}^{adv}$. This can be done by solving the following optimization problem:

$$\min_{\delta \in \Delta_q^p} \quad f^{\mathbf{w}}(\mathbf{x} + \delta) \tag{3}$$

In the following sections, we present our three novel L_1 and L_2 attack algorithms. We first provide an *Iterative Fast Coordinate Method (IFCM)* which is efficient enough to generate a large number of adversarial examples with satisfactory attacking effectiveness. Therefore, IFCM is used in the defense measure of adversarial training. To evaluate the robustness of the detector more accurately, we provide two stronger attacks. One is the search-based attack called *Augmented Iterative Search (AIS)*, and the second one is optimization-based attack called *Rounded Relaxation with Reparameterization (R3)*. Though AIS and R3 cannot generate millions of adversarial examples for adversarial training, their scalability is sufficient to evaluate the robustness of the model, and both algorithms outperform the adaption of the state-of-the-art methods significantly in attacking effectiveness. Empirical evaluation shows that with adversarial training on samples generated by IFCM, the model is significantly more robust against stronger attacks such as AIS and R3.

[4]Note that, our attacks can be easily adapted to allow negative perturbations.

4 FAST ATTACK: IFCM

In this section, we propose a fast attack algorithm called *Iterative Fast Coordinate Method* (IFCM), which is inspired by the iterative gradient sign method [13]. Given the sparsity and discretilization of input data, IFCM conducts multiple gradient descents, each with step size 1 along a single coordinate. The descending coordinate is selected in different ways when generating L_1 and L_2 attacks respectively. In IFCM for L_1 attack (IFCM_1), the coordinate with the smallest partial derivative is chosen. While in IFCM for L_2 attack (IFCM_2), the selection of descending coordinate considers not only the value of partial derivative along that coordinate, but also the increment of L_2 norm caused by one unit descent along the coordinate. Thus, following the greedy algorithm for the knapsack problem, IFCM_2 selects the coordinate with the highest *Value of Perturbation* (VoP) which is defined later. The two variants of IFCM are illustrated as follows.

4.1 IFCM_1

Algorithm 1: Iterative Fast Coordinate Method for L_1 attack (IFCM_1)

input : Fraud instance $\mathbf{x}$, q, detection model parameter $\mathbf{w}$
output: Adversarial example $\mathbf{x}^{adv}$ with L_1 norm bounded distortion

1 $\mathbf{x}^{adv} \leftarrow \mathbf{x}$,$\delta \leftarrow \mathbf{0}_n$;
2 **while** $\sum_{i=1}^{n} \delta_i < q$ **do**
3 Compute the gradient $\nabla_{\mathbf{x}} f^{\mathbf{w}}(\mathbf{x}^{adv})$ with back-propagation;
4 $i_{min} = \arg\min_i \frac{\partial f^{\mathbf{w}}(\mathbf{x}^{adv})}{\partial x_i}$;
5 **if** $\frac{\partial f^{\mathbf{w}}(\mathbf{x}^{adv})}{\partial x_{i_{min}}} > 0$ **then**
6 **return** $\mathbf{x}^{adv}$;
7 $x_{i_{min}}^{adv} \leftarrow x_{i_{min}}^{adv} + 1$,$\delta_{i_{min}} \leftarrow \delta_{i_{min}} + 1$;
8 **return** $\mathbf{x}^{adv}$.

As illustrated by Algorithm 1, at each iteration, IFCM_1 computes the gradient $\nabla_{\mathbf{x}} f^{\mathbf{w}}(\mathbf{x}^{adv})$ of the fraudulent score with the current adversarial example (Line 3), and selects the coordinate i_{min} with the minimal partial derivative (Line 4). If the partial derivative of $f^{\mathbf{w}}(\mathbf{x}^{adv})$ with $x_{i_{min}}$ is negative, a step of coordinate descent along the coordinate i_{min} is conducted with step size 1. Otherwise, when the gradient $\nabla_{\mathbf{x}} f^{\mathbf{w}}(\mathbf{x}^{adv})$ is positive along every coordinate, or the maximal distortion q is reached, the algorithm terminates.

4.2 IFCM_2

The main difference between IFCM_2 and IFCM_1 lies in the choice of descending coordinate. IFCM_1 selects the coordinate by comparing the partial derivative, while the coordinate chosen in IFCM_2 is decided by *Value of Perturbation* (VoP). VoP is inspired by the greedy algorithm for knapsack problem. For each coordinate i, a unit descent on such coordinate decreases the fraudulent score $f^{\mathbf{x}^{adv}}$ by

Algorithm 2: Iterative Fast Coordinate Method for L_2 attack ($IFCM_2$)

input : Fraud instance $\mathbf{x}, q$, detection model parameter $\mathbf{w}$
output : Adversarial example $\mathbf{x}^{adv}$ with L_1 norm bounded distortion

1 $\mathbf{x}^{adv} \leftarrow \mathbf{x}, \delta \leftarrow \mathbf{0}_n$;
2 **while** $\sqrt{\sum_{i=1}^{n} \delta_i^2} < q$ **do**
3 Compute the gradient $\nabla_{\mathbf{x}} f^{\mathbf{w}}(\mathbf{x}^{adv})$ with back-propagation;
4 Compute VoP_i for $i \in [n]$;
5 $i_{max} = \arg\max_i VoP_i$;
6 **if** $VoP_{i_{max}} \leq 0$ **then**
7 **return** $\mathbf{x}^{adv}$;
8 $x_{i_{max}}^{adv} \leftarrow x_{i_{max}}^{adv} + 1, \delta_{i_{max}} \leftarrow \delta_{i_{max}} + 1$;
9 **return** $\mathbf{x}^{adv}$.

an estimated amount of $-\frac{\partial f^{\mathbf{w}}(\mathbf{x}^{adv})}{\partial x_i}$ (the *value*), and increases the L_2 norm of current perturbation δ by $\sqrt{\|\delta\|_2^2 + 2\delta_i + 1} - \|\delta\|_2$ (the *weight*). VoP_i is defined as the estimated decrease on the fraudulent score per unit increase of L_2 norm via increasing δ_i

$$VoP_i = \begin{cases} \frac{-\frac{\partial f^{\mathbf{w}}(\mathbf{x}^{adv})}{\partial x_i}}{\sqrt{\|\delta\|_2^2 + 2\delta_i + 1} - \|\delta\|_2} & \text{if } \sqrt{\|\delta\|_2^2 + 2\delta_i + 1} \leq q \\ -\infty & \text{otherwise.} \end{cases}$$

5 SEARCH-BASED ATTACK: AIS

In order to further evaluate the vulnerability of deep fraud detector to adversarial perturbation, we propose an attack method called *Augmented Iterative Search* (AIS) which shows superior performance in fooling the detector and outperforms all benchmarks significantly in our experimental evaluation. Since the size of perturbation space is a combinatorial number, for example, $|\Delta_q^1| = C_{n+q-1}^q$, the exhaustive search is intractable. Thus, AIS repeatedly searches for a "simple" perturbation which brings the largest decrease on the objective value of optimization (3) and accumulates those "simple" perturbations to form the final solution. The simplest perturbation perturbs only one coordinate with a single unit, which is the one with L_1 norm of 1, termed "single-unit" perturbation. AIS goes further by taking into account the perturbations which perturb two entries with non-zero changes, one with 1 and one with -1[5], termed "tuple-unit" perturbations. However, the number of feasible tuple-unit perturbations is of quadratic order $O(n^2)$ with n, which makes it impractical to exhaustively search all of them. Thus, AIS augments the search process of the tuple-unit perturbations by restricting the space of such perturbations, as we explain later.

5.1 AIS_1

Similarly as IFCM, we implement two variants of AIS, AIS_1 and AIS_2, for L_1 and L_2 attacks respectively. AIS_1 is illustrated in Algorithm 3. δ^t denotes the accumulative perturbations until iteration t and $\delta^0 = \mathbf{0}$. At each iteration t, the set of all single-unit perturbations is denoted by $\Delta_t^- \cup \Delta_t^+$ where Δ_t^- contains the single-unit perturbations which have value -1 on the only non-zero entry and the adoption of $\delta^- \in \Delta_t^-$ decreases the L_1 norm of accumulative perturbation δ^{t-1} by 1. Similarly, $\delta^+ \in \Delta_t^+$ has value 1 on the only non-zero entry.

$$\Delta_t^- = \{\delta \in \mathbb{Z}^n \mid \|\delta\|_1 = 1, \|\delta^{t-1} + \delta\|_1 = \|\delta^{t-1}\|_1 - 1, \delta^{t-1} + \delta \geq \mathbf{0}\}$$
$$\Delta_t^+ = \{\delta \in \mathbb{Z}^n \mid \|\delta\|_1 = 1, \|\delta^{t-1} + \delta\|_1 = \|\delta^{t-1}\|_1 + 1\}. \quad (4)$$

The set of all tuple-unit perturbations is denoted by $\Delta_t^{\pm}$

$$\Delta_t^{\pm} = \{\delta^+ + \delta^- \mid \delta^+ \in \Delta_t^+, \delta^- \in \Delta_t^-\}$$

However, the size of $\Delta_t^{\pm}$ is $mn - m$ where m is the number of non-zero entries in δ^{t-1}, and in the worst case, $\|\Delta_t^{\pm}\|$ can be quadratic with n. To augment the search of tuple-unit perturbation, AIS_1 replaces $\Delta_t^{\pm}$ with its restricted subset $\tilde{\Delta}_{t,k}^{\pm}$ with much smaller size. In order to do so, we first define the value of a "simple" perturbation δ with respect to current solution $\mathbf{x}^{adv}$ as the marginal decrease on the objective value of (3) due to the adoption of δ

$$V(\mathbf{x}^{adv}, \delta) = f^{\mathbf{w}}(\mathbf{x}^{adv}) - f^{\mathbf{w}}(\mathbf{x}^{adv} + \delta)$$

The restricted set $\tilde{\Delta}_{t,k}^{\pm}$ is defined as follows

$$\tilde{\Delta}_{t,k}^{\pm} = \{\delta^+ + \delta^- \mid \delta^+ \in \tilde{\Delta}_{t,k}^+, \delta^- \in \tilde{\Delta}_{t,k}^-\} \quad (5)$$

where $\tilde{\Delta}_{t,k}^+ \subset \Delta_t^+$ and $\tilde{\Delta}_{t,k}^- \subset \Delta_t^-$ contain the top k single-unit perturbations among Δ_t^+ and Δ_t^- respectively ordered by values in the descending order. The intuition behind $\tilde{\Delta}_{t,k}^{\pm}$ is that a tuple-unit perturbation is more likely to have higher marginal decrease on $f^{\mathbf{w}}(\mathbf{x}^{adv})$ if it consists of the single-unit perturbations with high values. The size of $\tilde{\Delta}_{t,k}^{\pm}$ is of order $O(k^2)$, and the choice of k balances the scalability and solution quality.

The search operation in AIS_1 is as follows. Given the L_1 norm bound of distortion, a reasonable policy is to process the spaces of perturbations Δ_t^-, Δ_t^+ and $\tilde{\Delta}_{t,k}^{\pm}$ separately with different priorities. Δ_t^- is assigned the highest priority as the perturbation within it can decrease the L_1 norm of the accumulative perturbation. Thus, if there exists one single-unit perturbation $\delta^- \in \Delta_t^-$ with positive value, it is returned and added to δ^{t-1}, the accumulative perturbation so far (Lines 4–6). If Δ_t^- contains no single-unit perturbation with positive value, $\tilde{\Delta}_{t,k}^{\pm}$ is processed. Similarly, the tuple-unit perturbation $\delta^{\pm}$ with positive value is returned and δ^t is updated (Lines 8–10). Finally, if both Δ_t^- and $\tilde{\Delta}_{t,k}^{\pm}$ cannot decrease the objective $f^{\mathbf{w}}(\mathbf{w}(\mathbf{x}^{adv}))$, AIS_1 searches for δ^+ with positive value from Δ_t^+, which increases the L_1 norm of accumulative perturbation by 1 (Lines 14–16).

5.2 AIS_2

The key difference between AIS_2 and AIS_1 is that, in AIS_1, a single-unit perturbation would always change the L_1 norm of the accumulative perturbation by 1, and the L_1 norm of the accumulative perturbation is unchanged when a tuple-unit perturbation is applied. However, this is not always the case when producing L_2 attacks. Therefore, the value of a "simple" perturbation needs to take into account not only the decrease of fraudulent score $f^{\mathbf{w}}(\mathbf{x}^{adv})$, but also the change of the L_2 norm of accumulative perturbation. Given

[5]Notice that the value of -1 does not necessarily violate the non-negative requirement on δ, which is the accumulative perturbation.

Algorithm 3: Augmented Iterative Search (AIS) for L_1 attack

input : Fraud instance $\mathbf{x}$, q, maximum iteration K,detection model parameter $\mathbf{w}$, k

output : Adversarial example $\mathbf{x}^{adv}$ with L_1 norm bounded distortion

$\mathbf{x}^{adv} \leftarrow \mathbf{x}$,$\delta^0 \leftarrow \mathbf{0}_n$;
for $t \leftarrow 1$ **to** K **do**
  Compute Δ_t^-, Δ_t^+ and $\tilde{\Delta}_{t,k}^{\pm}$ with (4) and (5);
  Solve $\min_{\delta \in \Delta_t^-} V(\mathbf{x}^{adv}, \delta)$ and get δ^-;
  if $V(\mathbf{x}^{adv}, \delta^-) > 0$ **then**
    $\delta^t \leftarrow \delta^{t-1} + \delta^-$, $\mathbf{x}^{adv} \leftarrow \mathbf{x}^{adv} + \delta^-$;
  else
    Solve $\min_{\delta \in \tilde{\Delta}_t^{\pm}} V(\mathbf{x}^{adv}, \delta)$ and get $\delta^{\pm}$;
    if $V(\mathbf{x}^{adv}, \delta^{\pm}) > 0$ **then**
      $\delta^t \leftarrow \delta^{t-1} + \delta^{\pm}$, $\mathbf{x}^{adv} \leftarrow \mathbf{x}^{adv} + \delta^{\pm}$;
    else
      if $\|\mathbf{x}^{adv} - \mathbf{x}\|_1 = q$ **then**
        return $\mathbf{x}^{adv}$
      Solve $\min_{\delta \in \Delta_t^+} V(\mathbf{x}^{adv}, \delta)$ and get δ^+;
      if $V(\mathbf{x}^{adv}, \delta) > 0$ **then**
        $\delta^t \leftarrow \delta^{t-1} + \delta^+$, $\mathbf{x}^{adv} \leftarrow \mathbf{x}^{adv} + \delta^+$;
      else
        return $\mathbf{x}^{adv}$;
return $\mathbf{x}^{adv}$

the current solution $\mathbf{x}^{adv}$ of optimization (3) and a feasible "simple" perturbation δ, we define the *Value of Perturbation (VoP)* as follows:

$$VoP(\mathbf{x}^{adv}, \delta) = \begin{cases} \frac{\Delta f^{\mathbf{w}}(\mathbf{x}^{adv}, \delta)}{d_2(\mathbf{x}^{adv}, \delta) + \epsilon} & \text{if } \|\mathbf{x}^{adv} + \delta\|_2 \geq \|\mathbf{x}^{adv}\|_2 \\ -d_2(\mathbf{x}^{adv}, \delta) \cdot \Delta f^{\mathbf{w}}(\mathbf{x}^{adv}, \delta) & \text{o.w.} \end{cases}$$

where ϵ is a small constant to avoid being divided by 0,

$$d_2(\mathbf{x}^{adv}, \delta) = \|\mathbf{x}^{adv} + \delta\|_2 - \|\mathbf{x}^{adv}\|_2,$$

and

$$\Delta f^{\mathbf{w}}(\mathbf{x}^{adv}, \delta) = f^{\mathbf{w}}(\mathbf{x}^{adv}) - f^{\mathbf{w}}(\mathbf{x}^{adv} + \delta),$$

Similar with AIS_1, we start from the definition of the single-unit perturbations, which form Δ_t^- and Δ_t^+. Δ_t^- is the same as defined in (4) while Δ_t^+ is a bit different, considering the L_2 norm constraint.

$$\Delta_t^+ = \{\delta \in \mathbb{Z}^n \mid \|\delta\|_1 = 1, \|\delta^{t-1} + \delta\|_1 = \|\delta^{t-1}\|_1 + 1, \|\delta^{t-1} + \delta\|_2 \leq q\}.$$

The restricted set $\tilde{\Delta}_{t,k}^{\pm}$ follows the definition in (5) where $\tilde{\Delta}_{t,k}^+ \subset \Delta_t^+$ and $\tilde{\Delta}_{t,k}^- \subset \Delta_t^-$ contain the k single-unit perturbations with top-k *VoP*s among Δ_t^+ and Δ_t^- respectively. AIS_2 searches the"simple" perturbation in a way that, the perturbations which can decrease the L_2 norm of accumulative perturbation are evaluated first. For this aim, we divide $\tilde{\Delta}_{t,k}^{\pm}$ into two sets, $\tilde{\Delta}_{t,k}^{\pm,m}$ which consists of perturbations whose adoption can decrease the L_2 norm of accumulative perturbation, and $\tilde{\Delta}_{t,k}^{\pm,p}$, the complementary set of $\tilde{\Delta}_{t,k}^{\pm,m}$.

$$\tilde{\Delta}_{t,k}^{\pm,m} = \{\delta \in \tilde{\Delta}_{t,k}^{\pm} \mid \|\delta^{t-1} + \delta\|_2 < \|\delta^{t-1}\|_2\} \quad \tilde{\Delta}_{t,k}^{\pm,p} = \tilde{\Delta}_{t,k}^{\pm} \setminus \tilde{\Delta}_{t,k}^{\pm,m}.$$

Algorithm 4 illustrates the details of AIS_2. First, AIS_2 processes

Algorithm 4: Augmented Iterative Search (AIS) for L_2 attack

input : Fraud instance $\mathbf{x}$, q, maximum iteration K,detection model parameter $\mathbf{w}$, k

output : Adversarial example $\mathbf{x}^{adv}$ with L_2 norm bounded distortion

1 $\mathbf{x}^{adv} \leftarrow \mathbf{x}$,$\delta^0 \leftarrow \mathbf{0}_n$;
2 **for** $t \leftarrow 1$ **to** K **do**
3 Compute Δ_t^-, Δ_t^+ and $\tilde{\Delta}_{t,k}^{\pm}$;
4 Solve $\min_{\delta \in \Delta_t^- \cup \tilde{\Delta}_{t,k}^{\pm,m}} VoP(\mathbf{x}^{adv}, \delta)$ and get δ^-;
5 **if** $VoP(\mathbf{x}^{adv}, \delta^-) > 0$ **then**
6 $\delta^t \leftarrow \delta^{t-1} + \delta^-$, $\mathbf{x}^{adv} \leftarrow \mathbf{x}^{adv} + \delta^-$;
7 **else**
8 Solve $\min_{\delta \in \Delta_t^+ \cup \tilde{\Delta}_{t,k}^{\pm,p}} VoP(\mathbf{x}^{adv}, \delta)$ and get δ^+;
9 **if** $VoP(\mathbf{x}^{adv}, \delta^+) > 0$ **then**
10 $\delta^t \leftarrow \delta^{t-1} + \delta^+$, $\mathbf{x}^{adv} \leftarrow \mathbf{x}^{adv} + \delta^+$;
11 **else**
12 **return** $\mathbf{x}^{adv}$
13 **return** $\mathbf{x}^{adv}$

$\Delta_t^- \cup \tilde{\Delta}_{t,k}^{\pm,m}$, with perturbations decreasing the L_2 norm of the accumulative perturbation. If there exists one "simple" perturbation $\delta^- \in \Delta_t^- \cup \tilde{\Delta}_{t,k}^{\pm,m}$ with positive *VoP*, it is returned and added to δ^{t-1}, the accumulative perturbation so far (Lines 4–6). Otherwise, $\Delta_t^+ \cup \tilde{\Delta}_{t,k}^{\pm,p}$ is proceeded (Lines 8–10). The algorithm terminates if no "simple" perturbation within $\Delta_t^- \cup \Delta_t^+ \cup \tilde{\Delta}_{t,k}^{\pm}$ has positive *VoP* (Line 12) or the maximum iteration K is reached.

6 OPTIMIZATION-BASED ATTACK: R3

In this section, we propose a novel attack framework called *Rounded Relaxation with Reparameterization (R3)*. R3 solves the following relaxed optimization problem of (3) and rounds the continuous solution to get the discrete perturbation.

$$\begin{aligned} \min_{\delta \in \mathbb{R}_{+,0}^n} \quad & f^{\mathbf{w}}(\mathbf{x} + \delta) \\ \text{s.t.} \quad & \|\delta\|_p \leq q. \end{aligned} \tag{6}$$

The main problem here is that the constrained non-convex optimization is hard to solve. Though one can apply the projected gradient descent to ensure the solution is always feasible, unlike the box-constraint in the common L_∞ attacks, the projection operation is challenging under the complex L_1 and L_2 norm constraints. Thus, we propose the reparameterization tricks to transform the constrained optimization (6) to the equivalent unconstrained optimization by exploiting the property of L_1 and L_2 norms.

For the L_1 attack, we notice that the unit L_1 norm ball $\mathcal{B}_1 = \{\delta \in \mathbb{R}_{+,0}^n \mid \|\delta\|_1 = 1\}$ is the set of all n-dimensional probability distribution vectors. Thus, we can resolve the unit L_1 norm constraint in (6) with Softmax reparameterization as follows. Let $\mathbf{z} \in \mathbb{R}^n$ be

the variable to reparameterize δ.

$$\delta_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \quad \forall i \in [n]$$

To generalize it to the L_1 norm within $[0, q]$, we define $\mathbf{r} \in \mathbf{R}^n$ and reparameterize δ as follows

$$\delta_i = q \cdot \frac{(tanh(r_i) + 1) \cdot e^{z_i}}{2 \sum_{j=1}^{n} e^{z_j}} \quad \forall i \in [n] \tag{7}$$

One can easily verify that δ satisfies the L_1 norm constraint and every feasible δ can be reparameterized with suitable $\mathbf{r}$ and $\mathbf{z}$.

The L_2 norm constraint of δ can also be resolved with similar reparameterization trick using $\mathbf{r}$ and $\mathbf{z}$.

$$\delta_i = q \cdot \frac{(tanh(r_i) + 1) \cdot e^{z_i}}{2\sqrt{\sum_{j=1}^{n} e^{2z_j}}} \quad \forall i \in [n] \tag{8}$$

With reparameterization, we solve the following unconstrained optimization which is equivalent to (6).

$$\min_{\mathbf{r}, \mathbf{z} \in \mathbb{R}^n} \quad f^{\mathbf{w}}(\mathbf{x} + \delta) \tag{9}$$

where δ is in the forms of (7) and (8) for L_1 and L_2 attacks respectively.

We use the Adam optimizer to conduct gradient descent to solve (9). After the continuous solution δ is obtained, we round it into a discrete solution satisfying the L_p norm bound of q. The rounding process works as follows. We pick a threshold value η from $[0, 1]$. Let $\tilde{\delta}$ be the solution of (9). For each entry $\tilde{\delta}_i$, if the residual value $\tilde{\delta}_i - \lfloor \tilde{\delta}_i \rfloor$ is larger than η, we set δ_i be $\lfloor \tilde{\delta}_i \rfloor + 1$ and otherwise, δ_i is assigned with $\lfloor \tilde{\delta}_i \rfloor$. The binary search is conducted on possible threshold values and output the final solution δ such that no feasible δ with $\|\delta\|_p \leq q$ can be obtained with higher threshold value. As there are at most n different residual values, the binary search can be done within $\log(n)$ rounds.

7 ADVERSARIAL TRAINING FOR IMPROVING ROBUSTNESS

In order to improve the robustness of the deep fraud detector, we incorporate adversarial examples into the training process (Algorithm 5), and solve the following robust optimization

$$\min_{\mathbf{w}} \mathop{\mathbb{E}}_{(\mathbf{x}, y) \sim \mathcal{D}} [\max_{\delta \in \Delta_{q_1}^1 \cup \Delta_{q_2}^2} L^{\mathbf{w}}(\mathbf{x} + \delta, y)] \tag{10}$$

where we omit the regularization term. Adversarial training (Algorithm 5) is a common practice to solve (10). The idea is to incorporate the adversarial examples into the training process and to approximate the optimization of the inner problem in (10). Noticing that both L_1 and L_2 attacks are considered in this paper, at each iteration, for each fraud instance in minibatch $\mathcal{B}$ of training data, we uniformly pick p from $\{1, 2\}$ and generate one adversarial example with IFCM$_p$. All the generated adversarial examples are grouped into $\mathcal{B}^{adv}$, which is concatenated with $\mathcal{B}$ to form the new minibatch training data $\mathcal{B}'$. A training step is then conducted on $\mathcal{B}'$ with the following loss function

$$\begin{aligned} \tilde{L}^{\mathbf{w}}(\mathbf{x}, y) =& \mathbb{1}_{\{y=0\}} L^{\mathbf{w}}(\mathbf{x}, y) + (1 - \gamma) \mathbb{1}_{\{y=1\}} L^{\mathbf{w}}(\mathbf{x}, y) \\ &+ \gamma \mathbb{1}_{\{y=1\}} L^{\mathbf{w}}(\mathbf{x}^{adv}, y) \end{aligned} \tag{11}$$

where γ is the parameter balancing the loss of the original data and adversarial examples.

Algorithm 5: Adversarial training with IFCM

```
Randomly initialize the network for fraud detection;
repeat
    Read minibatch B ⊂ D from training set D;
    For each fraud instance in B, randomly selects p ∈ {1, 2}
      and create an adversarial example with IFCM_p, grouped
      into B^adv;
    Concatenate B^adv with B as B';
    Conduct one training step over B' with loss L̃ in (11);
until training converged;
```

8 EXPERIMENTAL EVALUATION

In this section, we present extensive experimental evaluation to illustrate the performance of our methods.

8.1 Experimental Settings

8.1.1 Dataset and Data Analysis. Our dataset is provided by TaoBao, one of the largest e-commerce platforms in the world. The dataset contains all transaction data in a period of 30 days. Each transaction is encoded as a feature vector to represent its statistics and characteristics, including various prices, the number of historical purchases on an item, the offline metrics on fraudulent suspicious of the buyer and item, and the statistics on a user's preference, depicted by the records of interactions on all categories in the past one month. We randomly sample 1.5 million fraud transactions and 1.5 million benign instances from the data in the first 25 days to form a balanced training set. We uniformly pick 300 thousand transactions from the records of the last 5 days as the test data, where the proportion of fraud instances to benign ones is around 1 : 30.

We provide some statistics on the preference features to give a better impression on the fraud detection problem. One key statistic is the L_1 norm of a transaction instance $\mathbf{x}$, that is, the total number of interactions recorded in the past month. We notice that in the fraud transactions, 66.8% have zero interaction, i.e., $\|\mathbf{x}\|_1 = 0$[6], while the percentage is only 3.9% in the benign transactions. According to the explanation from domain experts in TaoBao, those fraud instances with zero L_1 norms are likely to be generated by large number of simple and cheap robots or scripts by the platforms, which are also easier to detect. The remaining fraud instances with non-zero L_1 norms correspond to the sophisticated scripts, such as the one in Figure 2, which makes complicated operations to behave like humans. Table 1 presents the distribution of $\|\mathbf{x}\|_1$ of instances with non-zero L_1 norms, as well as the mean value of $\|\mathbf{x}\|_1$.

There are several points we would like to make regarding these statistics. First, though it seems reasonable to have two separate classifiers, one for instances with zero L_1 norm and the other for the remaining instances, empirical attempts show that it provides

[6]We would like to remind the reader that for simplicity, we only consider the preference features to encode an instance, while in the deployed model, there are other non-zero features.

Type	$\|\mathbf{x}\|_1 \leq 100$	$100 < \|\mathbf{x}\|_1 \leq 200$	$200 < \|\mathbf{x}\|_1 \leq 300$	$\|\mathbf{x}\|_1 > 300$	mean($\|\mathbf{x}\|_1$)
fraud	33.87%	16.43%	10.85%	38.85%	**377**
benign	45.33%	23.56%	12.8%	18.31%	**208**

Table 1: Statistics on fraud and benign instances with non-zero L_1 norms

little improvement and brings extra vulnerability to adversarial attack. Thus, we use a single classifier as the detector for all instances. Second, despite the high amount of fraud instances with zero L_1 norms, the perturbation on those instances is meaningful as it indicates the potential that the adversary improves their simple bots with extra functions. Third, we also evaluate our attacks on fraud instances with non-zero L_1 norms separately, and the slight perturbation can still bypass the detector with high chance. For example, with $q = 20$ in L_1 attack, which corresponds to the distortion ratio of only around 5% given that the mean value of $\|\mathbf{x}\|_1$ is 377 for fraud instances with non-zero L_1 norms, the adversarial examples generated by AIS_1 can fool the detector with probability over 50%.

8.1.2 Model Training and Attack Settings. The deep fraud detector model is trained with the following parameters. The training batch size is 500 and the model is trained over 30000 batches. Adam [12] optimizer is used to conduct the stochastic gradient descent and the learning rate is set to be 0.0002. The regularization parameter α is set to be 0.01. The adversarial training is conducted with adversarial examples generated by $IFCM_1$ and $IFCM_2$, with $q = 30$ and $q = 6$ respectively. To test the attacking effectiveness of various algorithms, we consider $q \in \{10, 20, 30\}$ for the L_1 attack and $q \in \{4, 5, 6\}$ for the L_2 attack. In evaluation of the detector strengthened with adversarial training, we also try larger values of q, such as $q \in \{10, 20, 30, 40, 50\}$ for the L_1 attack and $q \in \{4, 5, 6, 7\}$ for the L_2 attack.

8.1.3 Baselines. We adapt several state-of-the-art continuous attacks to generate discrete perturbations, including FGSM [8], C&W attack [4], Logit-Space attack [2], and EAD attack [5].

FGM_1: FGM_1 is a single step L_1 attack method which selects the q coordinates with the minimal partial derivatives and adds one unit perturbation on each of them.

C&W attack: C&W attack is an optimization-based method primarily for continuous L_0, L_2, and L_∞ attacks on image classifiers. We adapt it to generate L_1 attack and use rounding techniques to obtain the discrete perturbation satisfying the norm constraint. C&W uses the Change of Variables (CoV) to replace δ with $\frac{q}{2}(tanh(\mathbf{w})+1)$ (element-wise) and solves the following continuous optimization

$$\min_{\mathbf{w}\in\mathbb{R}^n} c \cdot f(\mathbf{x} + \frac{q}{2}(tanh(\mathbf{w}) + 1)) + \|\frac{q}{2}(tanh(\mathbf{w}) + 1)\|_p$$

where $f(\mathbf{x}^{adv})$ is suggested to be defined as the difference of the logits, i.e., $f(\mathbf{x}^{adv}) = \max(z_1^{adv} - z_0^{adv}, 0)$ [4] with $\mathbf{z}^{adv}$ being the output of the last hidden-layer when the adversarial example $\mathbf{x}^{adv} = \mathbf{x} + \frac{q}{2}(tanh(\mathbf{w}) + 1)$ is the input. We replace it with $f(\mathbf{x}^{adv}) = z_1^{adv} - z_0^{adv}$ as z_1 tends to be larger than z_0 in our tested detector and the suitable threshold is around 0.9, rather than 0.5. Constant c is selected with binary search to ensure that $f(\mathbf{x}^{adv})$ is minimized to the extend while $\|\frac{q}{2}(tanh(\mathbf{w}) + 1)\|_p \leq q$ is still satisfied[7]. The optimization is solved with Adam optimizer. We pick the hyper-parameters that perform the best in solving the optimization in our tests, including the learning rate of 0.1 with 1000 gradient descent steps. 16 binary searches are conducted on $c \in [0, 100]$. The continuous solution δ is rounded off in a similar way as R3 attack. After the binary search returns the optimal c, we solve the optimization with 5 random starting-points of $\mathbf{w}$ and pick the best rounded discrete solution.

C&W-LS attack: Logit-Space attack [2] uses one-hot encoding to represent the discrete feature and applies the Softmax reparameterization to approximate the one-hot encoding. Formally, δ_i is represented by $Softmax(\mathbf{w}_i) \cdot (0, 1, ..., q)$ and the optimized variable is changed to $W = [\mathbf{w}_1, ..., \mathbf{w}_n]$. The optimization is similar to C&W attack, except that an entropy penalty is imposed to penalize δ_i being fractional. That is, C&W-LS solves the following optimization

$$\min_{W\in\mathbb{R}^{n\times(q+1)}} c\cdot f(\mathbf{x}+U(0,1,...,q)^T)+\|U(0,1,...,q)^T\|_p+\beta\cdot\sum_{i\in[n]} H(\mathbf{u}_i)$$

where $U = [\mathbf{u}_1, ..., \mathbf{u}_n]$, $\mathbf{u}_i = Softmax(\mathbf{w}_i)$, i.e., $u_{ij} = \frac{e^{w_{ij}}}{\sum_{j'=0}^{q} e^{w_{ij'}}}$ for $j \in \{0, 1, ..., q\}$, and $H(\mathbf{u}_i) = -\sum_{j=0}^{q} u_{ij} \log(U_{ij})$ is the entropy for $\mathbf{u}_i$ regarded as a distribution, which is minimized when $\mathbf{u}_i$ is one-hot. The implementation follows the same settings as C&W attack. We try different values of β and choose the best one in evaluation, which is 0.1 for L_1 attack and 0.5 for L_2 attack.

EAD attack: EAD attack [5] is a continuous L_1 attack on image classifier. It is also optimization-based attack which solves the following optimization problem with elastic-net regularization

$$\min_{\delta\in\mathbb{R}^n_{0,+}} c \cdot f(\mathbf{x} + \delta) + \beta \cdot \|\delta\|_1 + \|\delta\|_2^2$$

We try different values of β and choose the best one in evaluation, which is 5.0.

We also try to round other attack algorithms such as DeepFool [17] for discrete L_2 attack. However, since DeepFool aims to find the successful attack with the smallest L_2 distortion and thus imposes no norm restriction on perturbation. The rounding solution of DeepFool performs poorly. We also implement the random attack, which has little effect on degrading the performance of the classifier. Therefore, the results of these attempts are not presented in the paper.

8.1.4 Evaluation Metrics. We evaluate various attacks and adversarial training based on two metrics, *success rate* and *Average Precision (AP)*. Success rate measures the proportion of fraud instances which are correctly recognized by the detector with pre-defined threshold η and mis-classified under the attack. AP is commonly

[7] Though in [4], the binary search on c aims to ensure the minimal value of c such that δ successfully fools the detector, it performs worse than our implementation when rounding the solution to the discrete one with L_p norm bound q. Thus, we implement C&W attack in the way explained here.

Attack	Original model			Model with adversarial training				
	$q = 10$	$q = 20$	$q = 30$	$q = 10$	$q = 20$	$q = 30$	$q = 40$	$q = 50$
unattack	0.892	0.892	0.892	0.89	0.89	0.89	0.89	0.89
FGM_1	0.817	0.798	0.787	0.882	0.885	0.892	0.902	0.901
C&W	0.763	0.516	0.442	0.889	0.892	0.897	0.897	0.9
C&W-LS	0.765	0.572	0.575	0.889	0.889	0.889	0.885	0.887
EAD	0.767	0.573	0.393	0.887	0.891	0.895	0.896	0.896
$IFCM_1$	0.767	0.578	0.409	0.879	**0.876**	**0.875**	0.876	0.876
R3	0.745	0.463	0.251	0.881	0.877	**0.875**	**0.873**	**0.859**
AIS_1	**0.726**	**0.434**	**0.238**	**0.88**	**0.876**	**0.875**	0.874	0.873

Table 2: *Average Precision (AP)* of the original model and the model with adversarial training under L_1 attacks

Attack	Original model			Model with adversarial training			
	$q = 4$	$q = 5$	$q = 6$	$q = 4$	$q = 5$	$q = 6$	$q = 7$
unattack	0.892	0.892	0.892	0.89	0.89	0.89	0.89
C&W	0.71	0.492	0.238	0.882	0.889	0.893	0.887
C&W-LS	0.66	0.447	0.256	0.888	0.893	0.897	0.9
$IFCM_2$	0.66	0.485	0.343	**0.877**	**0.876**	0.876	0.877
R3	0.68	0.463	**0.205**	0.894	0.895	**0.875**	0.877
AIS_2	**0.57**	**0.344**	0.207	0.878	0.877	0.876	**0.875**

Table 3: *Average Precision (AP)* of the original model and the model with adversarial training under L_2 attacks

Attack	$\|\mathbf{x}\| = 0$			$\|\mathbf{x}\| > 0$		
	$q = 10$	$q = 20$	$q = 30$	$q = 10$	$q = 20$	$q = 30$
FGM_1	5.4%	11.1%	13.4%	22.4%	34.1%	41%
C&W	10.1%	73.4%	75.4%	25.9%	47.4%	53.6%
C&W-LS	9.6%	45.5%	58.7%	25.7%	43.6%	44.7%
EAD	9.7%	58.5%	76.5%	26.6%	49%	60.7%
$IFCM_1$	9.7%	58.4%	90.4%	26.3%	49.4%	64.6%
R3	14.1%	81.4%	97.6%	27.9%	53.4%	**66.5%**
AIS_1	**19.3%**	**84.9%**	**97.8%**	**28.7%**	**54.1%**	66.4%

Table 4: *Success rate* of L_1 attacks

Attack	$\|\mathbf{x}\| = 0$			$\|\mathbf{x}\| > 0$		
	$q = 4$	$q = 5$	$q = 6$	$q = 4$	$q = 5$	$q = 6$
C&W	21.7%	64.4%	94.3%	37%	55.3%	68.7%
C&W-LS	28.2%	82.7%	95.3%	39.3%	58.7%	68.3%
$IFCM_2$	23.7%	78.4%	96.5%	39.2%	58.4%	69.8%
R3	28.8%	72%	94.5%	37.3%	56%	69.9%
AIS_2	**59.7%**	**89.8%**	**98.7%**	**45%**	**61.4%**	**71%**

Table 5: *Success rate* of L_2 attacks

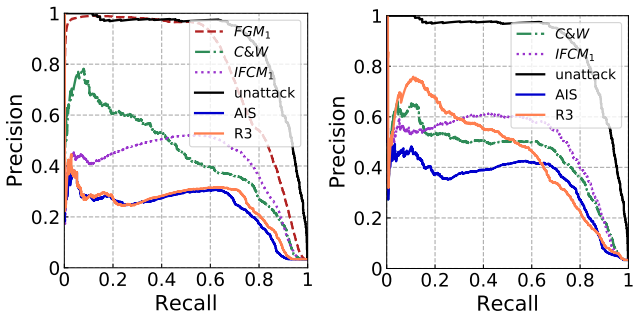
adopted to measure the performance of a classifier, which is defined by the enclosing area of the Precision-Recall curve (PR-curve). For a threshold η, the precision and recall are defined as follows [19]

$$P(\eta) = \frac{tp}{tp + fp} \quad R(\eta) = \frac{tp}{tp + fn},$$

where tp, fp and fn denote the true positive rate, false positive rate and false negative rate respectively. A higher value of AP intuitively means that the detector can recall a large proportion of fraud instances and meanwhile mis-classifies a few benign instances. A stronger attack should be able to degrade the AP to a lower value. The effectiveness of adversarial training can also be measured by AP by verifying whether the obtained model can pertain the AP under strong attacks. PR-curve is also useful for the selection of threshold in deployment. In practice, we choose the threshold η^* which equals the precision and recall, and we use η^* when evaluating the success rate of attacks. One thing to notice is that, an attack degrading the AP to a lower value doesn't mean it has higher success rate at η^*, since AP is an average measure of performance over all threshold values. We now present our key experimental results.

8.2 Average Precision

Tables 2 and 3 show the AP of the original detector and the one obtained via adversarial training under various L_1 and L_2 attacks respectively. Figure 3 shows the PR-curves of the detector under L_1 attacks with $q = 30$ and L_2 attacks with $q = 5$. We denote by *unattack* the trials without adversarial perturbation, corresponding to the performance on the original test data. The results show that (i) The deployed detector is extremely vulnerable to adversarial perturbations, as with $q = 20$ and $q = 5$ for AIS_1 and AIS_2 respectively, the AP is decreased from nearly 0.9 to 0.434 and 0.344. The numbers further decrease to 0.238 and 0.207 for AIS_1 attack with $q = 30$ and AIS_2 attack with $q = 6$ respectively. (ii) Our attack methods always achieve the largest degrade of AP in all settings, and significantly outperform baselines, especially AIS and R3, which show the similar performance. (iii) With adversarial training, the detector becomes significantly more robust to adversarial perturbations, as we consider larger q values 40, 50 for L_1 attack and 7 for L_2 attack, and the model can still achieve an AP above 0.859 in all



(a) L_1 attack, $q = 30$ (b) L_2 attack, $q = 5$

Figure 3: PR-curves.

Attack	L_1 attack			L_2 attack		
	$q = 10$	$q = 20$	$q = 30$	$q = 4$	$q = 5$	$q = 6$
FGM_1	0.005	0.005	0.005	-	-	-
IFCM	0.037	0.072	0.103	0.057	0.088	0.125
C&W	372.5	399.8	480.1	285.4	317.9	329.2
C&W-LS	290.1	313.6	455.9	299.5	301.3	333.2
EAD	71.8	87.4	89.5	-	-	-
R3	143.9	147.4	161.2	145.2	147.2	159.9
AIS	39.9	88.9	129.1	74.4	103.8	137.9

Table 6: Average runtime of generating one adversarial example (in seconds)

settings. Besides, the performance on the original test data is not degraded due to adversarial training, which shows the necessity of adversarial training[8].

8.3 Success Rate

Tables 4 and 5 show the success rates of tested L_1 and L_2 attacks respectively on the original model with threshold value η^*. To better understand the effectiveness of attacks, we separate the fraud instances based on the L_1 norm and randomly pick 1000 fraud instances with zero L_1 norm and 1000 fraud instances with non-zero L_1 norm. The results show that (i) Both types of fraud instances are highly vulnerable to adversarial perturbations. For instances with zero L_1 norm, with distortion bounds of 20 in L_1 norm and 5 in L_2 norm, our *AIS* attack can craft the instance to bypass the detector with probabilities over 84%. This shows the potential threat for the deployed system as the real-world adversaries and platforms may exploit this property and design more sophisticated scripts and bots with only tens of additional operations to bypass the detector with high changes. While for instances with non-zero L_1 norm, with distortion bounds of 20 in L_1 norm, which corresponds to an average distortion ratio of only 5% given the average L_1 norm for this type of instance being 377, our AIS_1 attack can fool the detector with probability 54.1%. (ii) Our attack methods achieve the highest success rates in all settings, and outperform baselines more significantly for instances with zero L_1 norm.

8.4 Feasibility of Attacks

8.4.1 Feasibility of Adversarial Perturbations. The perturbations generated here are not artificial noises on the input. Instead, there do exist such cases where the malicious service platform intentionally creates such perturbations. Recall that in the motivating scenario, we show an automatic script (Figure 2) by a malicious service provider. The script can conduct operations to pretend it as a human. One interesting pattern noticed by domain experts is that, when the script scrolls the item list on the website or mobile Apps, it will not directly visit the target item to create fake impression. Instead, it is designated with certain probability of visiting another item from certain categories and return to the item list. Imagine that fraudulent sellers and providers recognize the adversarial perturbations to bypass the detector. They can easily design effective scripts or even hire human labors to conduct the perturbation. Furthermore, we also sample adversarial perturbations and the domain experts verify that they are doable by the fraudulent sellers or malicious service providers.

8.4.2 Scalability of Attacks. Table 6 lists the average runtime of generating one adversarial example with various attacks. Notice that different attacks have varying proportions of computations able to be parallelized. Thus, the runtime of crafting a single instance cannot reflect the runtime of processing a batch of samples. From the adversary's perspective, our attacks are feasible as they can generate an adversarial examples within several minutes. Besides, one extra note on AIS is that it belongs to the *black-box* attacks as no information of the architecture of the detector is required. Thus, we argue that AIS could generate adversarial instances that are practical for real-world adversaries to craft.

[8]The proposed adversarial training process has been adopted to train the deep fraud detector for deployment since the Double 11 Shopping Festival of 2018.

9 CONCLUSION

This paper reveals the potential risk of deploying deep learning models in the task of fraud detection. We propose a fast attack called IFCM to generate large amount of effective adversarial examples, and two novel and more powerful attacks, search-based AIS attack and optimization-based R3 attack. We conduct extensive evaluations on a deployed deep fraud detector from TaoBao, one of the largest e-commerce platforms in the world with large-scale real-world transaction data. Results show that (i) Our proposed attacks can bypass the deployed detector with a high probability and can successfully degrade the Average Precision (AP) of the detector from nearly 90% to as low as 20% with slight perturbations. (ii) Our proposed attacks significantly outperform the state-of-the-art attack methods adapted to the domain of fraud detection, with respect to both the success rate and the degrading of AP of the detector in all testing trials. (iii) Our proposed adversarial training with the mixture of adversarial examples from both the L_1 and L_2 attacks significantly improves the robustness of the model, and pertains the AP of nearly 90% on the unperturbed test data.

ACKNOWLEDGMENTS

This research was supported by MOE, NRF, and NTU.

REFERENCES

[1] Anish Athalye, Nicholas Carlini, and David A. Wagner. 2018. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In *Proceedings of the 35th International Conference on Machine Learning (ICML'18)*. 274–283.

[2] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. 2018. Thermometer Encoding: One Hot Way to Resist Adversarial Examples. (2018).

[3] Nicholas Carlini, Guy Katz, Clark Barrett, and David L Dill. 2017. Provably Minimally-Distorted Adversarial Examples. *arXiv* 1709 (2017).

[4] Nicholas Carlini and David A. Wagner. 2017. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE Symposium on Security and Privacy (SP'17)*. 39–57.

[5] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. 2018. EAD: Elastic-Net Attacks to Deep Neural Networks via Adversarial Examples. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI'18)*. 10–17.

[6] Kang Fu, Dawei Cheng, Yi Tu, and Liqing Zhang. 2016. Credit Card Fraud Detection Using Convolutional Neural Networks. In *Neural Information Processing - 23rd International Conference (ICONIP'16)*. 483–490.

[7] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep Learning*. Vol. 1.

[8] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and Harnessing Adversarial Examples. *CoRR* abs/1412.6572 (2014).

[9] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick D. McDaniel. 2016. Adversarial Perturbations Against Deep Neural Networks for Malware Classification. *CoRR* abs/1606.04435 (2016).

[10] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the Knowledge in a Neural Network. *CoRR* abs/1503.02531 (2015).

[11] Earlence Fernandes Tadayoshi Kohno Bo Li Atul Prakash Amir Rahmati Ivan Evtimov, Kevin Eykholt and Dawn Song. 2017. Robust Physical-World Attacks on Deep Learning Models. *arXiv preprint arXiv:1707.08945* (2017).

[12] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[13] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2016. Adversarial Examples in the Physical World. *CoRR* abs/1607.02533 (2016).

[14] Lingqiao Liu, Chunhua Shen, and Anton van den Hengel. 2015. The Treasure Beneath Convolutional Layers: Cross-Convolutional-Layer Pooling for Image Classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15)*. 4749–4757.

[15] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards Deep Learning Models Resistant to Adversarial Attacks. *CoRR* abs/1706.06083 (2017).

[16] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. 2017. Universal Adversarial Perturbations. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. 86–94.

[17] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 2574–2582.

[18] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML'10)*. 807–814.

[19] David L Olson and Dursun Delen. 2008. *Advanced Data Mining Techniques*.

[20] Nicolas Papernot and Patrick D. McDaniel. 2016. On the Effectiveness of Defensive Distillation. *CoRR* abs/1607.05113 (2016).

[21] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2016. The Limitations of Deep Learning in Adversarial Settings. In *IEEE European Symposium on Security and Privacy (EuroS&P'16)*. 372–387.

[22] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks. In *IEEE Symposium on Security and Privacy (SP'16)*. 582–597.

[23] Herbert Robbins and Sutton Monro. 1985. A stochastic approximation method. In *Herbert Robbins Selected Papers*. 102–109.

[24] Andrew Slavin Ross and Finale Doshi-Velez. 2018. Improving the Adversarial Robustness and Interpretability of Deep Neural Networks by Regularizing Their Input Gradients. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI'18)*. 1660–1669.

[25] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. 2016. Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS'16)*. 1528–1540.

[26] Ning Su, Yiqun Liu, Zhao Li, Yuli Liu, Min Zhang, and Shaoping Ma. 2018. Detecting Crowdturfing "Add to Favorites" Activities in Online Shopping. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web (WWW'18)*. 1673–1682.

[27] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2013. Intriguing Properties of Neural Networks. *CoRR* abs/1312.6199 (2013).

[28] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick D. McDaniel. 2017. Ensemble Adversarial Training: Attacks and Defenses. *CoRR* abs/1705.07204 (2017).

[29] Haiqin Weng, Zhao Li, Shouling Ji, Chen Chu, Haifeng Lu, Tianyu Du, and Qinming He. 2018. Online E-Commerce Fraud: A Large-Scale Detection and Analysis. In *34th IEEE International Conference on Data Engineering (ICDE'18)*.

[30] Weilin Xu, David Evans, and Yanjun Qi. 2018. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In *25th Annual Network and Distributed System Security Symposium (NDSS'18)*.

[31] Xiaoyong Yuan, Pan He, Qile Zhu, Rajendra Rana Bhat, and Xiaolin Li. 2017. Adversarial Examples: Attacks and Defenses for Deep Learning. *CoRR* abs/1712.07107 (2017).

[32] Ashkan Zakaryazad and Ekrem Duman. 2016. A Profit-Driven Artificial Neural Network (ANN) with Applications to Fraud Detection and Direct Marketing. *Neurocomputing* 175 (2016), 121–131.

[33] Mengchen Zhao, Zhao Li, Bo An, Haifeng Lu, Yifan Yang, and Chen Chu. 2018. Impression Allocation for Combating Fraud in E-commerce Via Deep Reinforcement Learning with Action Norm Penalty. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI'18)*. 3940–3946.

[34] Panpan Zheng, Shuhan Yuan, Xintao Wu, Jun Li, and Aidong Lu. 2018. One-Class Adversarial Nets for Fraud Detection. *CoRR* abs/1803.01798 (2018).

[35] Stephan Zheng, Yang Song, Thomas Leung, and Ian J. Goodfellow. 2016. Improving the Robustness of Deep Neural Networks via Stability Training. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*. 4480–4488.

Predictive Crawling for Commercial Web Content

Shuguang Han
Google AI, 1600 Amphitheatre Parkway, Mountain View, CA, USA
hanshuguang@google.com

Bernhard Brodowsky
Google Zurich, Brandschenkestrasse 110, 8002 Zurich, Switzerland
bernhardb@google.com

Przemek Gajda
Google Zurich, Brandschenkestrasse 110, 8002 Zurich, Switzerland
pgajda@google.com

Sergey Novikov
Google Zurich, Brandschenkestrasse 110, 8002 Zurich, Switzerland
sergeyn@google.com

Michael Bendersky
Google AI, 1600 Amphitheatre Parkway, Mountain View, CA, USA
bemike@google.com

Marc Najork
Google AI, 1600 Amphitheatre Parkway, Mountain View, CA, USA
najork@google.com

Robin Dua
Google AI, 1600 Amphitheatre Parkway, Mountain View, CA, USA
robindua@google.com

Alexandrin Popescul
Google AI, 1600 Amphitheatre Parkway, Mountain View, CA, USA
apopescul@google.com

ABSTRACT

Web crawlers spend significant resources to maintain freshness of their crawled data. This paper describes the optimization of resources to ensure that product prices shown in ads in a context of a shopping sponsored search service are synchronized with current merchant prices. We are able to use the predictability of price changes to build a machine learned system leading to considerable resource savings for both the merchants and the crawler. We describe our solution to technical challenges due to partial observability of price history, feedback loops arising from applying machine learned models, and offers in cold start state. Empirical evaluation over large-scale product crawl data demonstrates the effectiveness of our model and confirms its robustness towards unseen data. We argue that our approach can be applicable in more general data pull settings.

CCS CONCEPTS

• **Information systems → Web crawling**; *E-commerce infrastructure*; *Search engine architectures and scalability*.

KEYWORDS

Predictive Crawling, Product Search, Commercial Content Change Dynamics

ACM Reference Format:
Shuguang Han, Bernhard Brodowsky, Przemek Gajda, Sergey Novikov, Michael Bendersky, Marc Najork, Robin Dua, and Alexandrin Popescul. 2019. Predictive Crawling for Commercial Web Content. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313694

This paper is published under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 International (CC-BY-NC-ND 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY-NC-ND 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313694

1 INTRODUCTION

An important characteristic of web content is the constancy of its change. While the dynamics of change are well understood in the context of generic web pages [2, 4, 5], commercial content (e.g., product listings, e-commerce sites, etc.) dynamics are relatively less explored. We see two major differences between the generic web content and commercial content. First, commercial content tends to have different change dynamics as it is closely related to marketing and economic factors such as supply and demand. Second, commercial content contains richer domain-specific metadata, e.g., merchant and brand, which provides us with more facets to understand its change dynamics. Besides, since there are billions of products advertised and sold online, often from small independent merchants, tracking and predicting of commercial content change at scale is an important but hard research challenge. Thus, in this paper, we focus on the change dynamics for commercial content.

In particular, we study commercial content change in the context of building an effective product search service. Figure 1 illustrates a high level architecture for such service. Product search results are based on the *product offers* that are listed on *merchants'* websites. These merchants are distributed and independent, and their websites cannot be searched in real-time due to latency constraint. Thus, a local database is used to store local copies of offer attributes. This paper studies the dynamics of price changes since price is one of the most important attributes of online products. The local database is then used by the search interface when users access offer details. The crawler plays a crucial role in our search service, as it attempts to keep the local database fresh. The ultimate goal for the crawler is to ensure that the prices seen by users (from the local database) match the prices shown on merchants' websites.

To achieve this goal, we need to update the local database immediately after an offer changes its price. One potential way is encouraging merchants to push price change information through merchant feeds. Our production system indeed receives such information. This mechanism alone, however, in our experience is insufficient to maintain maximum product offer freshness. Typically, merchant feeds are pushed according to specific schedules,

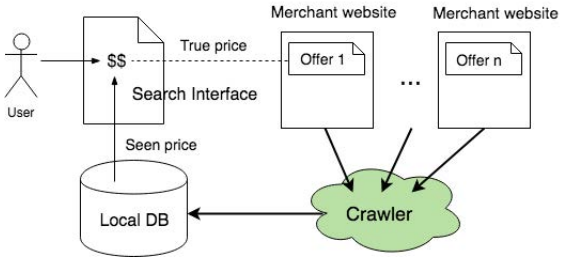


Figure 1: Overall architecture for our product search service.

but offers may change attributes regardless of the feed upload schedule. Creating a quality feed and delivering it continuously carries a substantial cost for merchants. Therefore, significant efforts are still required to build an efficient offer crawler.

An alternative solution is to repeatedly crawl merchants' websites for updates. Production crawlers often need to handle billions of offers, and scheduling frequent re-crawls for all them is impractical. To deal with that, our production crawler selects a subset of offers based on certain heuristics, and then crawls them periodically using a *Batch Scheduler*. Meanwhile, to compensate for important offers that might not be selected, we include an *InstaCrawl* component to crawl offers immediately after user clicks. Although this brings reasonable database freshness, offer selection heuristics in the *Batch Scheduler* are hard to build, and we overspend substantial resources on *InstaCrawl* because most offers do not change prices.

Previous development of generic web crawlers encounters a similar issue; researchers and practitioners try to solve it by predicting content change and schedule crawls accordingly [19, 24]. Most of existing studies developed their models based on web pages' past change frequencies [12, 13, 16, 20, 25]. There are two issues with this approach. First, it cannot handle the web pages with no or little crawl history, i.e., cold start. Second, it cannot deal with feedback loops — adjustments to the crawl strategy can distort the distribution of change frequency features so that the trained model may not work anymore. Later studies discovered that the content of a web page can help alleviate the cold start problem since similar pages share similar change patterns [28, 30].

Inspired by prior work, in this paper, we apply the predictive crawling approach to the domain of crawling commercial content. Informed by the latest advances in deep learning, we propose a state-of-the-art neural network based method to predict price change at scale. Our method combines both change dynamics as well as content and metadata information based features in a unified framework which enables us to effectively handle cold start and feedback loops of change frequency features.

We further develop new crawling strategies based on the predictive models and examine their effectiveness in the production environment. In this paper, we focus on optimizing *InstaCrawl* because the corresponding offers are more likely to be accessed by users. Particularly, we use the predictive model to reduce crawls for offers that are unlikely to change prices. Our experiments show that the new crawling strategies succeed in saving significant amount of resources. Since we use metadata and content features which are relatively static and independent from crawl history, our methods save resources even in the cold-start scenarios, for offers with little or no prior change history. In this paper, we intend to keep our models generalizable to other crawling components. For example, our price change model can also be applied to the *Batch Scheduler* for selecting more appropriate offers for periodical re-crawls.

To summarize, the main contributions of our work are as follows:

- To motivate our predictive models, we provide a comprehensive, global-scale study of factors that affect online price change – the first such published study to the best of our knowledge (Section 3).
- We develop a state-of-the-art and scalable price change prediction model that combines both price change frequency and metadata features (Section 4), and further demonstrate both its effectiveness and robustness in a cold-start setting (Section 5).
- We utilize the price change prediction model for making crawl decisions, and demonstrate that it can save a significant amount of crawl resources in production setting (Section 6).

2 RELATED WORK

A large body of work addressing freshness in pull systems has been done in the context of web crawlers (e.g. [6, 12, 23]). The topic started attracting significant interest in industry and research in the 1990s together with the rise of web search engines. Crawler optimization to address significant resource demands became an active area of research too.

A number of studies, for example [11, 13, 31], focused on change detection and estimation of frequency of change with sampling based policies. Cho and Garcia-Molina [8, 9] analyzed web content change patterns and proposed several design choices for incremental crawlers. Among them, the authors discovered that web pages changed with different frequency, and the overall freshness of a crawled corpus can be improved by 10%-23% through adjusting download frequency for different pages. Brewington and Cybenko [4] presented statistical analysis of web page modifications, introduced a freshness metric and estimated the download rate to maintain a desired level of freshness. The works [19] and [2] analyzed degree to which change correlates with other page properties, described finer-grained change statistics within different regions in the page, across various types of web pages and user visitation patterns. Tan and Mitra [30] used unsupervised learning to focus crawling within groups of pages that share similar change patterns.

Also, in the context of web page crawling, Radinsky and Bennett [28] extended the prediction model by combining change frequency features with the content of pages and their similarity to other pages. However, the use of change frequency features brings several important issues. Such features are calculated from download history and are sensitive to changes in crawl frequency. Changes to the crawling strategy will significantly affect the distribution of crawl history features, whereas the goal for change prediction is to apply such a model to reduce unnecessary crawling. Radinsky and Bennett's model can potentially avoid such issues since it also considers the content of web pages. However, the authors did not examine the performance after excluding the change frequency features. Also, they assume a full observation of crawl history when computing the change frequency which are unavailable most of the time.

Having a reliable prediction of when a web page will change is not enough for building an effective crawler. A key component in addition to estimating change probability is establishing an update policy. Optimality and scalability of such policies for various metrics under different assumptions is a subject of a number of studies. A recent work [3] presented a tractable near-optimal randomized strategy that can be computed in near-linear time. Eckstein et al. [18] studied the monitoring of web page changes under politeness constraint. Cho and Garcia-Molina [10] introduced freshness and age metrics and proposed several crawler policies assuming that change frequency follows a Poisson distribution. Also under the Poisson assumption, Coffman et al. [15] studied crawler optimality and proved an optimal formula for page refresh frequency, and Grimes et al. [20] studied optimality of a combined cost model for staleness and crawl resources when estimating refresh rates. We show, in this paper, that prices are not equally likely to change across different times of day and days of the week, thus deviating from the Poisson model. Aligned with our findings, Calzarossa and Tesser [5] analyzed temporal change patterns for news websites and showed that the changes are time dependent with significant hourly and daily fluctuations.

The work from Olston and Pandey [25] addressed crawler optimality which focused on persistent content and attempted avoiding ephemeral content likely to be obsolete by the time it reaches the index. Lefortier et al. [21], on the other hand, studied situations when engines do want to present fresh results even over ephemeral content. Wolf et al. [32] studied crawling strategies prioritized to refresh content that is more likely to surface as a result of a user query. Pandey et al. [27] presented a probabilistic change behaviour model, resource allocation and scheduling mechanism to answer continuous queries over dynamic web content and describe its more subtle differences compared to general web crawling.

A related area of research studies policies for refreshing web content caches. For example, Cohen and Kaplan [16] proposed policies for proactive validation of cache content using historic access patterns such as frequency and recency of access.

Overall, existing studies did conduct extensive research on both crawler optimization and content change prediction. However, they mainly focused on the crawling of generic web pages, and, to the best of our knowledge, very few of them studied the crawling for commercial content. Our work is the first to provide a large-scale analysis of the dynamics of commercial web content change, and develop scalable and effective content change prediction models.

3 ANALYZING PRICE CHANGE AT SCALE

Before diving into the details of building machine learning models for predictive crawling, we first provide a large-scale analysis of price change patterns for online offers. In particular, we focus on temporal, geographical and content patterns.

3.1 The Salticus Dataset

In order to conduct an unbiased analysis that avoids feedback loops [29], we sampled one million online offers and scheduled hourly crawls for all of them. The sample is a stratified mix of two sample types: (a) random uniform from the entire corpus of offers; (b) click weighted, to better represent popular offers that receive clicks. We download hourly snapshots of these offer pages from May 1st 2018 to the middle of September 2018, and use this crawl data for the following analysis, as well as for our predictive machine learning models. We refer to these snapshots as the *Salticus*[1] *dataset* in the remainder of this paper.

3.2 Temporal Patterns

We analyze two types of temporal patterns. First, we examine whether there are certain preferred time periods (e.g., weekday or weekend, morning or evening) for merchants to change prices. Second, we analyze how long it takes for an offer to change its price. Other temporal patterns such as changes over different days of the month and months of the year can also be interesting. However, since we only have data for a few months, such analysis might not be representative statistically.

Figure 2 illustrates the price change probability over the different days of the week and hours of the day (in merchant local time). We find that merchants are more likely to change prices at midnight. The peak of midnight change is probably due to automated updates. For days of the week, changes are more likely to happen on weekdays than on weekends. The non-uniform temporal patterns suggest that time is a useful factor to determine price change. Therefore, both day of the week and hour of the day are applied in our machine learning models. It is worth noting that our data spans from May to September which does not overlap with major holiday seasons. We expect different temporal patterns during holidays.



Figure 2: Probability of price change for different days of the week and hours of the day (local time). For each horizontal axis unit (e.g., Mon.), its probability equals to the number of downloaded offer pages with price change (on Mon.) over the total number of downloaded offer pages (on Mon.).

Figure 3 plots the distribution of time intervals between price changes. The horizontal axis denotes the time interval (in hours) for an offer to change its price; the vertical axis denotes the percentage of price changes — the amount of price changes within a given time interval over the total number of price changes in our dataset. As an example, suppose our dataset consists of two offers, A and B. Offer A changes its price at $t_0 + 1h$ and changes again at $t_0 + 3h$. Offer B changes its price at $t_0 + 1h$. Therefore, the total number of price changes is three, with two coming from A and one from B. The probability of change for $1h$ time interval is $2/3$, and the probability of change for $2h$ interval is $1/3$. Note that for an offer,

[1]Named after a spider genus, popularly known as "zebra spider".

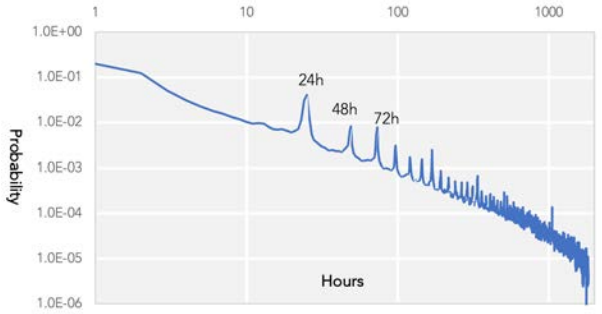


Figure 3: Distribution of time intervals between two price changes (log-log plot). The probability for a given time interval measures how many of the price changes in the Salticus dataset happened in the given interval.

the first observation of its price does not count as a change since we do not know its prior price.

Figure 3 conveys several messages. We find that among all price changes, a considerable amount of them happen within a short time. This might come from the products whose prices depend on external factors such as price fluctuations of financial instruments. One illustrative example is gold coin prices, which change dynamically together with the gold spot price. Second, we see multiple local peaks at 24 hours, 48 hours, etc. This may be due to the automatic price updates, which was also discussed in Figure 2.

3.3 Geographical Patterns

The Salticus dataset contains online product offers from across the world. This enables us to explore price change patterns for different countries. Figure 4 provides the change probability over 20 countries. Each country is represented by its ISO code.[2] The probability is computed in the same manner as described in Section 3.2. We find that different countries exhibit varying change patterns. Among the 20 countries we considered, India (IN), Mexico (MX) and Brazil (BR) are the countries with the highest price change rate whereas the United States (US), Japan (JP) and Indonesia (ID) are the lowest. This might be related to economic factors such as inflation rates in a particular country. We find a 0.4629 Pearson correlation between countries' price change rates and their inflation rates (as of October 2018). The variation of price changes across countries motivates including country as a feature in our machine learning model.

3.4 Content Patterns

This section further explores the price change patterns for different types of products. In our repository, each offer is associated with a product category.[3] A product category such as *Apparel & Accessories > Clothing* defines the product area and the subcategory of an offer. With such information, we then analyze the price change probability in each top-level category of the taxonomy. Again, the probability is computed similarly to Section 3.2.

[2]ISO codes are defined at https://en.wikipedia.org/wiki/ISO_3166-1

[3]We adopt the public Google Shopping Product Taxonomy for our online offers. A full schema is at: http://www.google.com/basepages/producttype/taxonomy.en-US.txt.



Figure 4: Price change probability in different countries. Probability is computed similarly to Figure 2.

Figure 5 presents the price change probabilities for different taxonomy categories. *Software* has the highest change rate, as it is affected by a highly dynamic *Virtual Currency* subcategory. *Electronics* (e.g., phones, tablets) and *Media* (e.g., books, music) are more dynamic than others. Offers in *Business & Industrial* and *Religious & Ceremonial* categories have the lowest change rates. Again, the non-uniform price change rate across the different categories implies that offer metadata can provide meaningful information when building machine learning models, which we take into account when designing our model as described in the next section.

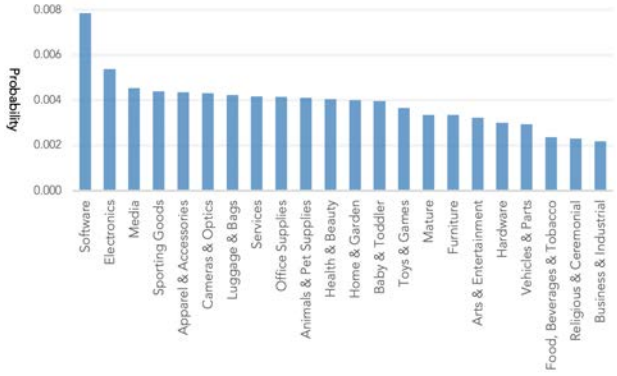


Figure 5: Price change probability over product categories. Probability is computed similarly to Figure 2.

4 METHODOLOGY

We model the price change prediction as a classification problem, for which we want to predict, for any individual offer, whether its price will change in the next K hours. Here, we do not consider the degree of change since our ultimate goal is to build an effective crawler that re-crawls an offer's page as long as there is a change. Therefore, we do not model it as a regression problem.

To build such a model, a set of training examples is required. Each example contains a list of features and a corresponding label. A positive label means that the offer's price changes within the next K hours; otherwise, a negative label is assigned. If a price x changes to y and then changes back to x, we still count it as a positive label because there are indeed price changes (twice) within the K hours. The Salticus dataset described in Section 3.1 is used for deriving our training, validation and testing examples.

4.1 Training/Testing Data

4.1.1 Generating Training Examples. The Salticus dataset contains two types of crawl events — the hourly scheduled crawls described in Section 3.1, and the regular crawls from the production system. Each crawl event is associated with a timestamp t, which allows us to split our data by time and simulate the moment we need to make a *<crawl, not crawl>* decision in the production system.

Specifically, for each simulated *prediction time* t, the future crawls after t provide full observability (on an hourly basis) of the price information in the next K hours. We use the offer information up to t to generate features, and the next K hours to generate ground-truth binary labels reflecting the price change (or lack thereof), thereby creating a single training example. By shifting t and repeating this process, we create a set of training examples.

In this paper, the *prediction time* t comes from both hourly crawls and production crawls. Using hourly crawls in addition to production crawls provides multiple benefits. First, it helps to generate enough training examples for rarely crawled offers in the production crawler. Second, it guarantees that we have an unbiased training data, with prediction times uniformly spread out over all 24 hours a day, and 7 days a week.

Finally, we observe non-uniform distributions for crawling time — crawls for popular offers are aggregated densely around the same time. To avoid generating too many similar examples, we restrict different ts to be at least 15 minutes away from each other.

The entire training generation process is illustrated in Figure 6. Instead of choosing the immediate follow-up crawl event as the next t, we skip two regular crawls and one hourly crawl since they are crawled less than 15 minutes from the last prediction time.

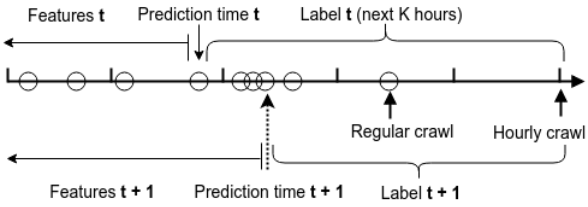


Figure 6: Training data generation process for timestamps t and $t + 1$. A vertical line denotes a Salticus hourly crawl; a circle represents a production crawl.

4.1.2 Seen and Unseen Offers. The one million Salticus offers used for model development only make up a small fraction of active offers. To understand whether our trained models can be generalized to offers that were never seen from the training data, we divide the Salticus dataset into seen and unseen offers. This is achieved by holding out 25% (i.e., 250K) of the one million offers (denoted *unseen offers*) and the remaining 750K offers are used as *seen offers*. Combining with the time-based split of training and testing data, the evaluation setup is illustrated in Figure 7. Our predictive models are only trained with the 750K offers.

Here, we use the unseen offers to simulate the production setting where our models do not observe any historical information for most of the offers. Therefore, we expect better model performances for the seen offers comparing to the unseen offers.

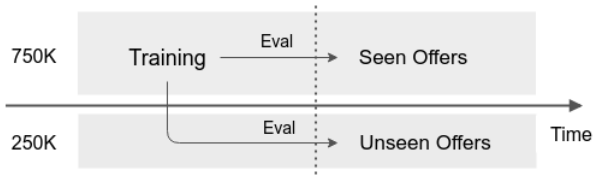


Figure 7: Overview of the training/evaluation setup.

4.1.3 Training/testing Setup. Our training/validation/testing datasets are split based on time. Particularly, crawling events from May 2018 to July 2018 are used for training, data from August 1 to August 5 is used for validation, and data from the rest of August is for testing. In total, we extract around 1.3 billion training examples, 84 million validation examples and 620 million testing examples. As expected, most of them did not contain price changes (i.e., negative price change label). In this paper, we set $K = 6$ hours, for which we observe around 1 : 48 positive/negative ratio for all training, validation and testing datasets. The validation dataset is used for model selection, and the reported results are based on the chosen model's performance on the testing data.

4.2 Features

We consider two types of features in this paper. First, we employ the historical change frequency features that are commonly adopted in previous studies [9, 28]. Second, we propose using metadata and other offer-related information that may be predictive of a price change even for offers with limited or no prior history.

4.2.1 Offer Change Frequency. We first extract a set of price change frequency features pertaining to each individual offer. We define the change frequency as the number of price changes per hour. Depending on the amount of crawling history being used, we extract the following three types of change frequency features — change frequency within the most recent day, week and month. We extract the time since the last price change and use it as a feature, as well.

4.2.2 Product Category Change Frequency. Data sparsity is a critical issue for offer-level change frequency. For new offers and rarely-crawled offers, the change frequency is either unavailable or unreliable. Radinsky and Bennett [28], in the context of web search, proposed to learn from similar web pages when encountering the sparsity issue. Similarly, in the context of commercial content, two offers may share similar change patterns if they come from the same product category. As a result, we further compute the change frequencies at the product category level and use them as features.

4.2.3 Metadata Features. The change frequency features are highly dependent on crawl history. This introduces issues when the history information is unavailable or is subject to change. To abate these issues, we introduce a number of history-independent features in our model. As shown in Section 3, offers from different product categories and different countries tend to exhibit different change patterns. This motivates us to extract the metadata information of an offer and use it as a source of features. Besides the product category and country, we include brand, condition, language, merchant and web page title of the offer. Details of the metadata features are presented in Table 1. Our previous analysis in Figure 2 illustrates

diverse price change patterns for different days of a week and hours of a day. Thus, the current day and current hour (of the prediction time) are also included as predictive features.

Compared to the offer and product change frequency features, metadata-based features have several benefits. First, they alleviate the cold-start issue since metadata is readily available for both existing and brand-new offers and product categories. Second, they are robust with respect to unexpected changes in the crawl history. Third, as we show next, metadata-based features can be combined with change frequency features for further model improvement.

Table 1: Features used in baselines and the proposed models.

Price change frequency for offer O	
Frequency (1 day)	Change frequency within last day
Frequency (1 week)	Change frequency within last week
Frequency (1 month)	Change frequency within last month
Most recent change	Time since the most recent change
Price change frequency for offers in same product category C	
Frequency (1 day)	Change frequency within last day
Frequency (1 week)	Change frequency within last week
Frequency (1 month)	Change frequency within last month
Most recent change	Avg. time since the most recent change
Metadata and related information for offer O	
Brand	Brand id
Condition	Condition: new, used or refurbished
Country	Country code
Day of Week	Day of week for the prediction time
Hour of Day	Hour of day for the prediction time
Language	Offer page language
Merchant	Merchant id
Offer title	Offer page title
Product category	Product category

4.2.4 Full Observability VS. Partial Observability. As mentioned, the change frequency features are highly sensitive to the amount of available crawl history. In the best case, if we can afford constant (e.g., hourly as in our dataset) crawls, the estimates for change frequency features will be more accurate. However, production crawlers usually cannot achieve the full observability, both due to the crawling costs as well as the politeness constraints [23].

In this paper, we experiment with two types of observability — **full observability** where we assume the observation of hourly crawls, and **partial observability** in which we only observe crawls from the existing crawling strategy. The former tells us the upper bound, whereas the latter can help us to estimate the predictive power for the change frequency features under existing crawling strategy. It is worth noting that the partial observability is sensitive to the change of crawling strategy, and adjustments to the strategy will further affect the observability, while the full observability scenario is not practical at the scale of the entire corpus.

4.3 Models

4.3.1 Baselines. In this paper, we adopt a set of baselines purely based on the change frequency features, in which we assume that the probability an offer changes its price in the next K hours equals to the historic price change frequency. Depending on the level of granularity, we include two baselines: an offer-level price change baseline and a product category level price change baseline. The former predicts an offer's price change based on its own history, whereas the latter employs the change history of its product category. Moreover, we include a third baseline combining both offer-level and category-level features in a linear model. These simple but effective baselines were commonly adopted in previous studies [8, 9, 28, 30].

4.3.2 Proposed Models. As discussed, we model the prediction task as a binary classification problem to classify whether an offer's price will change in the next K hours. To incorporate both numerical change frequency features and metadata information in one unified framework, in this paper, we adopt a feed-forward deep neural network (DNN) model.

The change frequency features, if included, are treated as numerical dense features. Metadata information is modeled by sparse features, which are embedded into a low-dimensional space [22]. Here, an embedding is a condensed vector (low-dimensional vector) representation of metadata information. Semantically, similar metadata would have similar embeddings located closely in the vector space. The combination of dense frequency features and embedded metadata features in a single neural network architecture enables both *memorization* and *generalization*, a desired property for a predictive machine learning model [7].

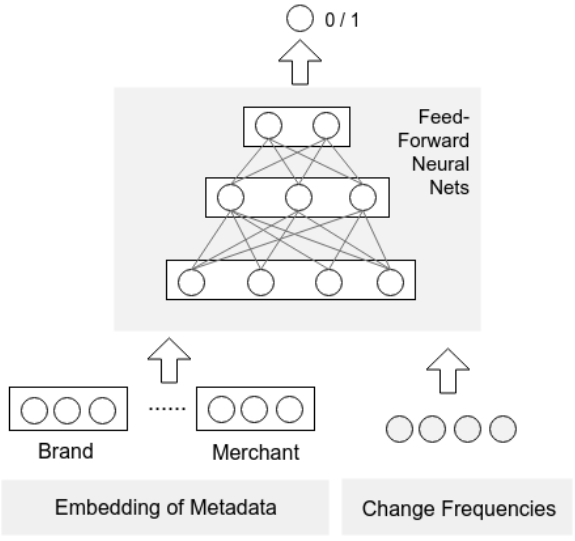


Figure 8: An illustration of our model architecture.

We use the TensorFlow DNNClassifier for model implementation [1], where we set three hidden layers with 256, 128 and 64 hidden units in each layer. We adopt ReLU (Rectified Linear Unit) as the

activation function for hidden units, and choose the Adagrad algorithm to optimize the cross-entropy loss. To deal with over-fitting, we adopt both L1 and L2 regularization and set both to 0.001.

To summarize, this paper includes the following baselines (the first three) and proposes two models for predicting price changes.

- Frequency (offer): offer-level change frequency
- Frequency (category): category-level change frequency
- Frequency (combined): combines offer-level and category-level change frequencies in a linear model
- DNN (metadata): DNN using only metadata information
- DNN (metadata + frequency): DNN using both metadata and offer + product category change frequencies.

5 PREDICTING PRICE CHANGE

This section reports the comparison results between the proposed models and baselines. Our evaluation adopts the **AUC** metric (Area Under Receiver Operating Characteristic Curve) to measure the performance of different models and baselines. A value of 0.5 means a random guess whereas 1.0 corresponds to a perfect prediction.

5.1 Evaluation on Seen Offers

Table 2 reports the testing AUCs for different predictive models on seen offers (see Section 4.1.2 for its definition). Here, we try to assess model quality on predicting the future for the offers being seen from the training set. The AUC values are obtained in the following way — firstly, we evenly split our testing data into 20 subsets; then, we evaluate on each subset; finally, we compute the averages and standard deviations over the 20 subsets. In Table 2, the top nine rows refer to baselines and the last two correspond to the proposed models. Overall, all of these models achieved AUCs above 0.5, meaning that both change frequency and offer metadata information have a predictive power. They can also be combined to further boost the prediction performance.

Table 2: Testing AUC (standard deviation) for different prediction models on seen offers.

Model \ Observability	Partial	Full
Frequency (category, 1 day)	0.560 (0.003)	0.573 (0.004)
Frequency (category, 1 week)	0.577 (0.004)	0.584 (0.004)
Frequency (category, 1 month)	0.570 (0.003)	0.583 (0.003)
Most recent (category)	0.569 (0.004)	0.582 (0.003)
Frequency (offer, 1 day)	0.642 (0.002)	0.721 (0.003)
Frequency (offer, 1 week)	0.769 (0.003)	0.815 (0.003)
Frequency (offer, 1 month)	0.791 (0.003)	0.832 (0.003)
Most recent (offer)	0.789 (0.003)	0.827 (0.003)
Frequency (combined)	0.791 (0.004)	0.847 (0.003)
DNN (metadata)	0.842 (0.005)	
DNN (metadata + frequency)	**0.875 (0.002)**	**0.883 (0.003)**

5.1.1 Full Observability VS. Partial Observability. Table 2 shows that the change frequency features computed from fully-observable crawl history outperform the ones derived from partially-observable history (details about observability are in Section 4.2.4). In terms of the baseline *Frequency (offer, 1 day)*, having fully observable history boosts AUC by as much as 12.3%. This implies the importance of collecting more crawling history. However, such an implication goes against our ultimate goal — building predictive models to lower crawling frequency. We can resonably expect that after reducing crawls from production crawlers, the performance of the partial observability baselines will decrease.

Compared to the baselines, our proposed models have the following advantages. First, the *DNN (metadata)* model does not use any price change feature and thus is not subject to the change of crawl history observability. Second, for the *DNN (metadata + frequency)* model, its performance difference under full and partial observability is small compared to the difference in the baselines. Both of them demonstrate the robustness of our proposed models.

5.1.2 Baselines VS. Proposed Models. As for baselines, we observe that offer-level price change features significantly outperform the category-level features. This is expected since the product category only provides a high-level abstraction for an offer whereas specific characteristics for each offer are missing. However, category features would be helpful when offer-level features are absent, e.g., for new offers without history. In this experiment, we only see a marginal improvement when combining offer and category features. This is because there are no new offers in the seen offers.

Comparing to the baselines, our *DNN (metadata)* model significantly outperforms the best baseline using the partial observability history, and works equivalently well as the best baseline using the full observability history. Since *DNN (metadata)* does not use any change frequency feature, it is robust towards new offers. Moreover, once the change frequency features are available, they can be simply incorporated in the *DNN (metadata + frequency)* model to yield even better performance.

5.2 Generalizability to Unseen Offers

To understand whether our models can be generalized to other offers, this section replicates the evaluation in Section 5.1 but assesses over 250K unseen offers (see its definition in Section 4.1.2). This section examines how our proposed models would generalize to the production system, which often handles billions of offers. Note that an unseen offer is different from a brand-new offer. The former refers to an offer that was not observed in the training data, whereas the latter denotes an offer without crawl history. For the unseen offers in this paper, we can still compute their offer-level and category-level price change frequency features.

Table 3 reports the AUCs of different predictive models for unseen offers. The baselines are roughly the same as Table 2, whereas the proposed models drop slightly. This aligns with our expectation since some metadata information from the unseen offers might never be observed from the training data. Surprisingly, even with partially observable history, the proposed models still outperform all baselines and maintain the AUCs above 0.80. With fully observable history, although the pure metadata feature does not achieve

the AUC as the best baseline, it can be combined with the price change features and eventually produces the best performance.

It is worth noting that although we do not specifically examine our model performances over the brand-new offers, we still expect that we can achieve reasonable performances using *DNN (metatda)* since this model does not use any crawl history information. In this case, offer level baselines would fail and the *DNN (metadata + frequency)* model will be downgraded to *DNN (metadata)*.

Table 3: Testing AUC (standard deviation) for different prediction models on unseen offers.

Model \ Observability	Partial	Full
Frequency (category, 1 day)	0.554 (0.002)	0.567 (0.002)
Frequency (category, 1 week)	0.572 (0.002)	0.579 (0.002)
Frequency (category, 1 month)	0.565 (0.002)	0.579 (0.002)
Most recent (category)	0.564 (0.002)	0.578 (0.002)
Frequency (offer, 1 day)	0.642 (0.002)	0.720 (0.003)
Frequency (offer, 1 week)	0.766 (0.003)	0.817 (0.003)
Frequency (offer, 1 month)	0.788 (0.003)	0.833 (0.003)
Most recent (offer)	0.787 (0.003)	0.828 (0.003)
Frequency (combined)	0.797 (0.003)	0.849 (0.003)
DNN (metadata)	0.803 (0.003)	
DNN (metadata + frequency)	**0.854 (0.003)**	**0.862 (0.003)**

5.3 Summary

Overall, the evaluation results from the above two sections clearly demonstrate the effectiveness of our proposed models, which not only work well on the seen offers but also show generalizability to unseen offers. Besides, we discover that metadata information is an important predictive feature. Such a feature is relatively static and easily accessible across different types of offers, effective in the cold start setting, and is not subject to the change of crawl history availability. Our proposed DNN approach provides an effective way to utilize the metadata information, and further enables the incorporation of additional features.

6 PREDICTIVE RESOURCE ALLOCATION

Experiments in Section 5 demonstrate the feasibility of building machine learning models to predict price change. However, it remains unclear how such models can be integrated with production crawlers. This section attempts to answer the question.

As mentioned in the Introduction, there are two important components in our production crawler — the *Batch Scheduler* that downloads offer pages periodically, regardless of whether or not the offers are accessed by users, and the *InstaCrawl* that follows past user clicks and downloads offer pages based on their click frequencies. The design of *InstaCrawl* relies on the observation that the past click is a good predictor of the future click [14]. Indeed, in our production crawler, 95% of user-clicked offers were also accessed in the past week. Besides, the *InstaCrawl* is also in line with previous studies which account for user experience when developing user-centric crawling quality metrics [26, 32].

In this paper, we focus on the optimization for *InstaCrawl* because the corresponding offers are more likely to be accessed by users. However, we believe that our model can also be applied to the *Batch Scheduler*, which is one of our future work directions. To be specific, our predictive models will be used for reducing unnecessary *InstaCrawl* crawls that do not yield new updates to the price information. This is achieved in the following way. For each crawl request, we skip it if our model predicts that the price will not change; otherwise, we send the request for crawling. Since our current production crawler harvests all requests and the proposed models only target to crawl a subset of them, we can use existing production logs for simulation.

It is worth noting that there is a trade-off between spending resource and improving the price information freshness. If there are unlimited resources (for both crawlers and merchant servers), one can always reach the optimal freshness. When crawling resources are limited, more stale prices will be shown. However, some approaches will be able to make better trade-offs than others. In this section, we compare the trade-offs of our proposed model with the commonly-adopted baselines in the literature.

6.1 Experimental Setup

6.1.1 Overview. As discussed, with production crawler logs, we can simulate different strategies to utilize the predictive models. To understand the effectiveness of each strategy, we need a test collection that contains all of the regular production crawls for simulation, as well as the true prices for evaluation. The Salticus dataset (see Section 3.1) is such a repository.

The simulation process is illustrated in Figure 9. Each circle denotes an *InstaCrawl* request c_i from our production crawler, and each vertical line corresponds to a Salticus hourly crawl s_k. When developing a predictive crawling strategy, we apply the decision (to crawl or not) based on a machine learning prediction for each c_i. If the decision is to crawl, the price ($\text{price}_{\text{local}}$) will be updated; otherwise, it stays unchanged. The hourly crawl s_k is used to compute the ground-truth price ($\text{price}_{\text{true}}$).

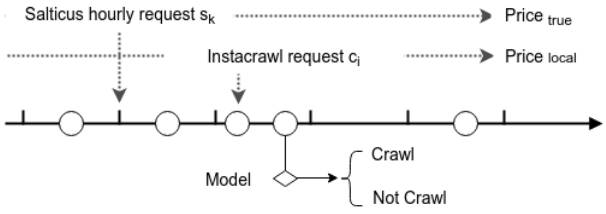


Figure 9: An illustration of applying machine learning predictions in a production crawler.

6.1.2 Crawl Decisions. Converting a model prediction to a crawl decision is a critical component in Figure 9. The simplest way is to define a threshold. A request with prediction score above the threshold will be crawled; otherwise, it will be skipped. However, a hard decision may result in some offers never being crawled. Therefore, we adopt a probabilistic decision strategy — the crawl decision is made stochastically and proportionally to the prediction

score. In this way, offers with low prediction scores but high click rate still have a chance to be crawled.

The skewedness of prediction scores makes it difficult to directly use the original scores. As a result, we rank scores and map them into n quantiles (with the first quantile mapping to the lowest score). For the i-th quantile, we use the probability in Equation 1 for making the stochastic decision. This is a simple heuristic that makes the probability linearly proportional to its quantile — a larger quantile will result in a higher probability. We do believe that there are more principled ways to determine the probability; however, we only seek for the simplest way for the proof of concept in this paper. Here, γ is a resource factor, which can be tuned to achieve different resource saving percentages.

$$p_i = \min\left(1.0, \frac{i}{n} \cdot \gamma\right), \quad \gamma \geqslant 0 \tag{1}$$

6.1.3 Compared Models. We compare the following crawling strategies. First, we include a baseline that makes uniform-probability (p_u) decisions for all *InstaCrawl* requests. Different p_u can yield different resource consumption rates. For example, setting $p_u = 0.8$ will crawl any request with the probability of 0.8, and thus can save 20% of resources. The uniform baseline provides us with a basic understanding of the task difficulty.

Based on the evaluation results from Section 5, we further include the following three models: the best baseline *Frequency (combined)*, the best model *DNN (metadata + frequency)* and *DNN (metadata)*. *DNN (metadata)* does not use crawl history and thus can handle cold-start and the change of crawl history observability. The change frequency features are sensitive to crawl history observability, different observability levels will yield different feature values (see Section 5.1.1). In this section, we use partially observable history, i.e., downloaded pages from *InstaCrawl*, but not the hourly Salticus crawls. Note that when a new crawling strategy is applied, observability will change further since earlier no-crawl decisions will affect the availability of crawl history. Despite that, *DNN (metadata + frequency)* still gives us the upper bound performance.

6.1.4 Evaluation Metrics. Our project operates on the trade-off between resource and performance. When crawling resources are reduced within the same strategy, the performance will always drop. However, some models might be able to make better trade-offs than others. Therefore, comparing the trade-offs among different strategies is the primary focus for evaluation. Of course, with saved resources, one can schedule more crawls for unpopular pages, which may further improve the overall performance. However, we mainly focus on evaluating *resource savings* in this paper, and leave the *resource re-allocation* evaluation for future work.

In our evaluation, resource usage (denoted by R) within a time period $[t_s, t_e]$ is defined to be proportional to the number of crawling requests that are eventually sent for page download at this time period. To be specific, suppose that we have N *InstaCrawl* requests and C ($C \leq N$) of them are finally sent for crawling. R will be proportional to C. Since N is a constant during our evaluation of different crawling strategies, we use it for normalization, i.e., $R = C/N$. Note that reducing crawls will not only reduce the load of our production crawler but also significantly reduce the load for the merchant servers. Too many requests at the same time might even cause our production crawlers to be blocked.

For performance, we adopt the commonly-used freshness metric (denoted by F) [9, 28]. A page is considered to be fresh if its latest local copy of crawl result aligns with its real-world content. Since the *InstaCrawl* requests are correlated with the number of user clicks, we can treat each c_i in Figure 9 as a user click. Then, we compute the freshness for each click c_i **before** we actually crawl it. This way, we actually examine whether a user will see the right content after click. As shown in Equation 3, the overall freshness F is computed by aggregating the freshness of all clicks in $[t_s, t_e]$.

$$f(c_i) = \begin{cases} 1, & \text{if price}_{\text{local}} = \text{price}_{\text{true}} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$F(t_s, t_e) = \frac{1}{N} \sum\nolimits_{t_{c_i} \in [t_s, t_e]} f(c_i) \tag{3}$$

6.2 Resource Savings

In this section, we first describe a resource-freshness trade-off for the uniform baseline, aiming to provide a basic understanding of the problem space. Then, we show how the trade-off can be improved using different machine learning based crawling strategies.

6.2.1 Freshness-Resource Trade-offs. Unlike the evaluation in Section 5, this section attempts to assess the accumulated effects of a number of machine learning decisions over time. However, first we will forego machine learning, and try to better understand the trade-off between freshness and resource utilization in our application. On one side of the spectrum, will the freshness be 100% if we harvest every regular crawl request in Figure 9? On the other side of the spectrum, what level of freshness can be maintained if we always decide *not to crawl*. Between these extremes, we can control the freshness-resource trade-off based on the uniform probability p_u (i.e., the uniform baseline in Section 6.1.3).

Accordingly, we continuously compute the freshness for different uniform crawling strategies every six hours (i.e., t_e - t_s = 6h) starting from August 1st, and plot them in Figure 10. Here, our experiments assume everything is synchronized at the beginning, so each line starts from freshness 1.0. In Figure 10, $p_u = 0.0$ (the bottom line) denotes that we never crawl, and thus the resource consumption is 0%. $p_u = 1.0$ (the topmost line) means that we crawl at every click c_i and the resource consumption is 100%.

We observe that the first 10% of resource (p_u from 0.0 to 0.1) improves freshness significantly by 0.381, whereas a further resource investment only provides a marginal increase — an additional 90% of resources (p_u from 0.1 to 1.0) only lifts the freshness by 0.029. This clearly shows that uniform crawling heavily overspends resources, particularly after $p_u > 0.1$. To reach the same level of crawl quality, other strategies might require much less resource. Therefore, the major focus of the following section is to explore the ways of optimizing resource utilization using our proposed models.

6.2.2 Resource Saving at Different Freshness Levels. To align with Section 5, we also evaluate on both seen and unseen offers. For each offer type, we compare resource utilization at fixed freshness levels for different crawling strategies. Instead of reporting values from one specific time period, we aggregate and report evaluation

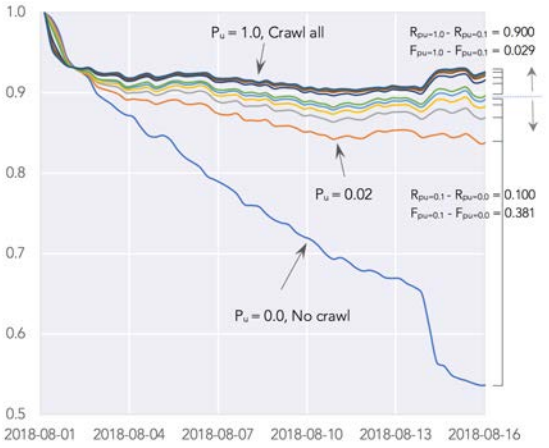


Figure 10: The change of freshness over time for a set of uniform crawling strategies. Each line denotes a p_u, which corresponds to 0.0, 0.02, 0.04, 0.06, 0.08, 0.1, 0.3, 0.5, 0.7, 0.9 and 1.0, respectively (bottom to top).

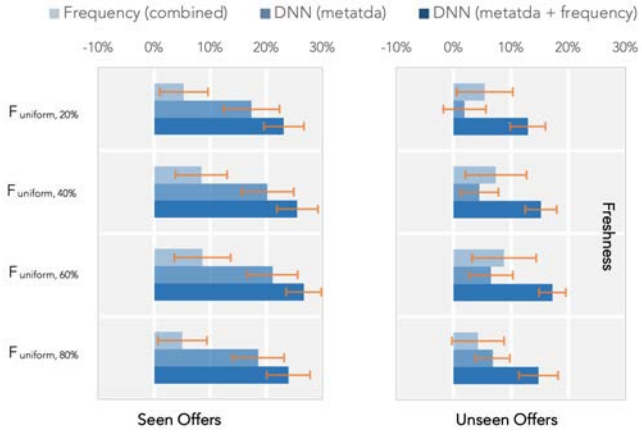


Figure 11: Resource savings (with 95% confidence interval) for different crawl strategies at target freshness levels. The horizontal axis denotes % of resources saved compared to the uniform strategy; the vertical axis denotes freshness. Freshness levels ($F_{uniform,\ X\%}$) are selected to cover 20%, 40%, 60% and 80% resource for the uniform strategy.

metrics over ten time periods five days each, i.e., August 5 to August 9, August 10 to August 14, and so on. Here, we start from August 5 because the impact from initialization fades away (we assume a freshness of 1.0 in the beginning of our simulation) and the performance becomes more stable.

In our evaluation, the freshness levels are chosen based on the uniform strategy, where we cover the freshness corresponding to the resource usage of 20%, 40%, 60% and 80% for the uniform strategy. We name them as $F_{uniform,\ 20\%}$, $F_{uniform,\ 40\%}$, $F_{uniform,\ 60\%}$ and $F_{uniform,\ 80\%}$. For each predictive crawling strategy, the amount of resources needed for a freshness level is obtained through tuning γ in Equation 1. Then, for the given crawling strategy, we measure its *resource saving* using the relative percentage of resources that can be saved compared to the uniform strategy.

Figure 11 plots the resource savings for each crawling strategy. Here, 0% means that a crawling strategy utilizes the same resource as the uniform strategy. Compared to the uniform baseline, all other models show the ability of spending less resources while maintaining the same freshness. In particular, *DNN (metadata + frequency)* achieves the best performance, which saves as much as 27% resources for seen offers and 17% for unseen offers. *DNN (metadata)* also outperforms the *Frequency (combined)* baseline on seen offers and performs equivalently on unseen offers. This aligns with our findings in Section 5.2, demonstrating the effectiveness of our proposed machine learning models. Note that there remains a big discrepancy between the seen and unseen offers for our proposed models. This is consistent with our findings in Section 5.2 and is due to the mismatch of metadata in these two different types of offers. In the future, we will explore more effective ways to bridge this gap between seen and unseen offers.

Again, as discussed in Section 5.2, since *DNN (metadata)* does not use any crawl history information, we expect that the resource saving performance for the unseen offers can be generalized to cold-start offers with no or very little history information. Overall, in this case, *DNN (metadata)* model can still save around 5% of resources whereas all frequency baselines will fail.

7 CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

In this paper, we study predictive resource allocation in the context of building a production crawler for commercial content. In particular, we focus on the problem of price change prediction for commercial offers. We start our study in Section 3 via a detailed analysis of price changes on a global scale. To the best of our knowledge, this is the first such publicly available analysis. In addition, the predictability of price change dynamics that we discover, motivates the further development of predictive price change models.

In Section 4, we propose a deep neural network based approach for predicting future price changes. The model goes beyond the change frequency features commonly used in prior work, by incorporating metadata information of online offers. There are multiple contributions of this approach. First, to the best of our knowledge, this is the first published attempt to build predictive crawlers in the context of commercial web content and price change prediction. Second, we update the existing predictive models [28, 30] with a state-of-the-art machine learning approach, which incorporates both numerical frequency features and content-based sparse features within a unified framework. Experimental results in Section 5 demonstrate the effectiveness of our approach. Moreover, the incorporation of metadata enables our model to deal with the cold-start problem and avoid feedback loops.

Beyond simply predicting price changes, we further study models integration with a production crawler in Section 6. Evaluation with production crawl logs demonstrate that our models can make better resource-freshness trade-off decisions than the frequency-only baselines both for seen and unseen offers. Furthermore, through

the use of metadata information, our models can save resources even for offers with no prior price change history.

7.2 Future work

Despite the promising results we obtained, our study still has several limitations, which will be the focus of our future work.

First, we observe that models built with solely metadata information perform less well on unseen offers. This can be partially solved by increasing the size of the full-observability Salticus dataset. However, for truly new offers, an exploration of more advanced approaches for a better modeling of metadata information will be required. Besides, we see evidence that more frequent model retraining increases the wins and plan to adopt it in the future.

Second, our use of machine learning scores for predictive resource allocation may be sub-optimal. Our experiment adopts a simple quantile-based conversion from prediction score to sampling probability (see Equation 1). This can be improved by adopting a more direct optimization approach. In addition, we may also explore ways to productively re-allocate the crawl resources saved by our methods in future work.

Finally, in this paper, we made the simplifying assumption that price changes (and crawls) can be modeled independently, and do not utilize any sequence information. This allows for simpler model deployment, and, as demonstrated in Sections 5 and 6, already outperfoms the existing baselines. Thus, an interesting direction for future work is applying sequence based deep learning models (e.g., RNN, LSTM [17]), to further improve the model performance.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)*, Vol. 16. 265–283.

[2] Eytan Adar, Jaime Teevan, Susan T Dumais, and Jonathan L Elsas. 2009. The web changes everything: understanding the dynamics of web content. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*. ACM, 282–291.

[3] Yossi Azar, Eric Horvitz, Eyal Lubetzky, Yuval Peres, and Dafna Shahaf. 2018. Tractable near-optimal policies for crawling. *Proceedings of the National Academy of Sciences* 115, 32 (2018), 8099–8103.

[4] Brian E Brewington and George Cybenko. 2000. How dynamic is the web? 1. *Computer Networks* 33, 1-6 (2000), 257–276.

[5] Maria Carla Calzarossa and Daniele Tessera. 2015. Modeling and predicting temporal patterns of web content changes. *Journal of Network and Computer Applications* 56 (2015), 115–123.

[6] Carlos Castillo. 2005. Effective web crawling. In *Acm SIGIR Forum*, Vol. 39. ACM, 55–56.

[7] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 7–10.

[8] Junghoo Cho and Hector Garcia-Molina. 2000. The Evolution of the Web and Implications for an Incremental Crawler. In *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB '00)*. Morgan Kaufmann Publishers Inc., 200–209.

[9] Junghoo Cho and Hector Garcia-Molina. 2000. Synchronizing a database to improve freshness. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data (SIGMOD '00)*, Vol. 29. ACM, 117–128.

[10] Junghoo Cho and Hector Garcia-Molina. 2003. Effective page refresh policies for web crawlers. *ACM Transactions on Database Systems (TODS)* 28, 4 (2003), 390–426.

[11] Junghoo Cho and Hector Garcia-Molina. 2003. Estimating frequency of change. *ACM Transactions on Internet Technology (TOIT)* 3, 3 (2003), 256–290.

[12] Junghoo Cho, Hector Garcia-Molina, and Lawrence Page. 1998. Efficient crawling through URL ordering. *Computer Networks and ISDN Systems* 30, 1-7 (1998), 161–172.

[13] Junghoo Cho and Alexandros Ntoulas. 2002. Effective change detection using sampling. In *Proceedings of the 28th International Conference on Very Large Data Bases*. VLDB Endowment, 514–525.

[14] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. Click models for web search. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 7, 3 (2015), 1–115.

[15] Edward G Coffman Jr, Zhen Liu, and Richard R Weber. 1998. Optimal robot scheduling for web search engines. *Journal of scheduling* 1, 1 (1998), 15–29.

[16] Edith Cohen and Haim Kaplan. 2001. Refreshment policies for web content caches. In *IEEE INFOCOM 2001 - The Conference on Computer Communications - Twentieth Annual Joint Conference of the IEEE Computer and communications Societies*, Vol. 3. IEEE, 1398–1406.

[17] Li Deng. 2014. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing* 3 (2014).

[18] Jonathan Eckstein, Avigdor Gal, and Sarit Reiner. 2008. Monitoring an information source under a politeness constraint. *INFORMS Journal on Computing* 20, 1 (2008), 3–20.

[19] Dennis Fetterly, Mark Manasse, Marc Najork, and Janet Wiener. 2003. A large-scale study of the evolution of web pages. In *Proceedings of the 12th International Conference on World Wide Web*. ACM, 669–678.

[20] Carrie Grimes, Daniel Ford, and Eric Tassone. 2008. Keeping a Search Engine Index Fresh: Risk and optimality in estimating refresh rates for web pages. *Proc. INTERFACE* (2008).

[21] Damien Lefortier, Liudmila Ostroumova, Egor Samosvat, and Pavel Serdyukov. 2013. Timely crawling of high-quality ephemeral new content. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*. ACM, 745–750.

[22] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[23] Marc Najork and Allan Heydon. 2002. High-performance web crawling. In *Handbook of Massive Data Sets*. Springer, 25–45.

[24] Christopher Olston and Marc Najork. 2010. Web crawling. *Foundations and Trends® in Information Retrieval* 4, 3 (2010), 175–246.

[25] Christopher Olston and Sandeep Pandey. 2008. Recrawl scheduling based on information longevity. In *Proceedings of the 17th International Conference on World Wide Web*. ACM, 437–446.

[26] Sandeep Pandey and Christopher Olston. 2005. User-centric web crawling. In *Proceedings of the 14th International Conference on World Wide Web*. ACM, 401–411.

[27] Sandeep Pandey, Krithi Ramamritham, and Soumen Chakrabarti. 2003. Monitoring the dynamic web to respond to continuous queries. In *Proceedings of the 12th International Conference on World Wide Web*. ACM, 659–668.

[28] Kira Radinsky and Paul N Bennett. 2013. Predicting content change on the web. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*. ACM, 415–424.

[29] D Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison. 2015. Hidden technical debt in machine learning systems. In *Advances in Neural Information Processing Systems*. 2503–2511.

[30] Qingzhao Tan and Prasenjit Mitra. 2010. Clustering-based incremental web crawling. *ACM Transactions on Information Systems (TOIS)* 28, 4 (2010), 17.

[31] Qingzhao Tan, Ziming Zhuang, Prasenjit Mitra, and C Lee Giles. 2007. Efficiently detecting webpage updates using samples. In *International Conference on Web Engineering*. Springer, 285–300.

[32] Joel L Wolf, Mark S Squillante, PS Yu, Jay Sethuraman, and Leyla Ozsen. 2002. Optimal crawling strategies for web search engines. In *Proceedings of the 11th International Conference on World Wide Web*. ACM, 136–147.

Generating Titles for Web Tables

Braden Hancock*
Stanford University
Stanford, CA
bradenjh@cs.stanford.edu

Hongrae Lee
Google
Mountain View, CA
hrlee@google.com

Cong Yu
Google
New York, NY
congyu@google.com

ABSTRACT

Descriptive titles provide crucial context for interpreting tables that are extracted from web pages and are a key component of search features such as tabular featured snippets from Google and Bing. Prior approaches have attempted to produce titles by selecting existing text snippets associated with the table. These approaches, however, are limited by their dependence on suitable titles existing *a priori*. In our user study, we observe that the relevant information for the title tends to be scattered across the page, and often—more than 80% of the time—does not appear verbatim anywhere in the page. We propose instead the application of a sequence-to-sequence neural network model as a more generalizable approach for generating high-quality table titles. This is accomplished by extracting many text snippets that have potentially relevant information to the table, encoding them into an input sequence, and using both copy and generation mechanisms in the decoder to balance relevance and readability of the generated title. We validate this approach with human evaluation on sample web tables and report that while sequence models with only a copy mechanism or only a generation mechanism are easily outperformed by simple selection-based baselines, the model with both capabilities performs the best, approaching the quality of crowdsourced titles while training on fewer than ten thousand examples. To the best of our knowledge, the proposed technique is the first to consider text-generation methods for table titles, and establishes a new state of the art.

KEYWORDS

title generation, web tables, table summarization, table understanding, sequence-to-sequence model, text generation, pointer-generator network, natural language generation

ACM Reference Format:
Braden Hancock, Hongrae Lee, and Cong Yu. 2019. Generating Titles for Web Tables. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313399

1 INTRODUCTION

Modern search engines no longer simply return links to relevant web pages. Where possible, they enhance the search experience by extracting and directly displaying the information that a user is seeking. Often, this information is best displayed in a semi-structured form, such as a table or list; for example, in response to the query "U.S. states by population 2018," a relevant table showing population by state is more effective than a sentence with the same information. It is now a major feature in mainstream search engines, e.g., [27, 29] and Figure 1 shows examples of tabular results in Google and Bing.

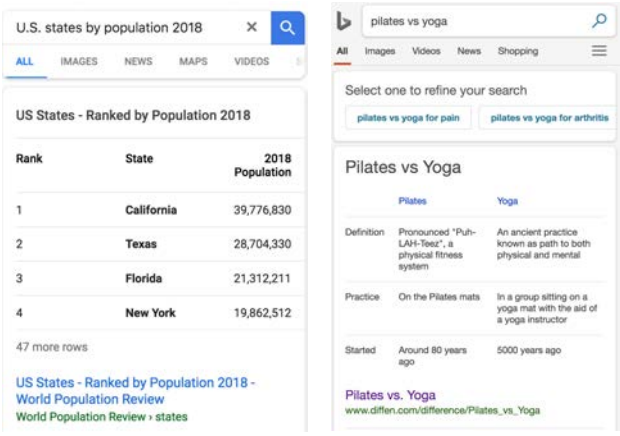


Figure 1: Example tabular results in search engines

However, separating a table from its original context removes important clues that help a user interpret its contents and trust its relevance. In our experience, the most effective way of providing this crucial context is by accompanying the tables with corresponding descriptive titles.

Consider, for example, the table shown in Figure 2. If this table were shown in isolation, it would be difficult to say for certain what it is about. If the search query that led to this table being returned is known (e.g., "Rocky Franchise Awards"), this helps with interpretation but still leaves many questions unanswered, such as: Does this table provide the exact information I was looking for or is it just the most relevant table that could be found? Are the awards in the table for one *Rocky* film or the entire *Rocky* franchise? Are these all from the same award source or from many sources? When a descriptive title is added, understanding of the table's contents improves, as does trust in the table's relevance to the query. The goal is for the user to have enough information from the displayed result (including the title) to know whether or not it is relevant to his or her information need.

Beyond user experience, the ability to generate titles for or otherwise summarize the content of semi-structured data enables other use cases as well. In a search engine, it plays a crucial role in supporting web applications such as table snippets [2]. From an information extraction standpoint, titles generated offline may be used as an

*BH completed the majority of this work while at Google.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313399

Award	Result	Nominee
Best Picture	Won	Robert Chartoff and Irwin Winkler
Best Director	Won	John G. Avildsen
Best Actor	Nominated	Sylvester Stallone
Best Actress	Nominated	Talia Shire

6 more rows

Figure 2: A descriptive title provides crucial context for understanding the contents of a table when it is displayed out of its original context. Consider, for example, the experience of trying to interpret or assess the relevance of this extracted table with versus without its title: "*Rocky* Academy Award Nominations (1977)."

additional signal when assessing the relevance of the summarized content with respect to user queries. With the increasing popularity of hands-free interfaces such as Google Home [17] and Amazon Echo [12], descriptive titles also provide a way for such devices to summarize a table's content in a more understandable way than reading the data row-by-row.

1.1 Challenges

In practice, automatically producing descriptive titles is quite challenging. First of all, the semantics of a table tend to be distributed among a variety of elements on a web page. For example, consider the table shown in Figure 3: the page title includes the name of the person being discussed, the section headings describe the topic of the table, and the column headers give the schema of the table contents. Any single one of these text strings would be insufficiently precise to serve as a table title, but together they include all the necessary information for a human to interpret its contents. This raises the question of how to best represent this collection of text snippets for machine learning.

Second, a title must sound natural; i.e., it should sound like something a human would write and not a "keyword soup." Clearly for the example in Figure 3, simply concatenating the text from each of the table metadata components would result in a title with poor readability.

Third, tables tend to describe specific relationships between specific entities, much more so than unstructured data such as free text. Consequently, descriptive titles for tables tend to include many proper nouns and rare or out-of-vocab (OOV) tokens, which are well-documented stumbling blocks for natural language generation algorithms [1, 19, 28].

Finally, the variety of tables on the web is immense [8, 14]. Some tables are relational ("horizontal") while others are entity-based ("vertical") [2]. A table may have five rows or 50. It may have an explicit caption and three section headings or none of either. It may contain the primary content of a web page or be purely supplemental, or it may be used entirely for formatting purposes and not actually be expressing any relationship at all. This tremendous diversity renders many heuristic-based approaches inadequate, as evidenced in part by the unsuccessful attempts to heuristically bootstrap training data which we report in Section 3.

Ultimately, a generated title must satisfy two properties: relevance and readability. A **relevant** title is one which correctly describes what information the table contains; it should be neither too vague ("Scores") nor too specific ("The box score of the baseball game that the Houston Astros won against the Los Angeles Dodgers in 10 innings in Game 5 of the 2017 World Series"). A **readable** title is one which sounds natural to a human reader. It need not be a complete sentence (indeed, most natural titles are not), but it should be fairly straightforward for a human to parse.

1.2 Conventional Approaches

To address readability, a common approach is to treat the title production task as a selection problem–given all the text snippets on the page, identify the one most likely to make a good title for a table in question. By selecting existing text, such approaches avoid the complexity of learning a language model for concatenating words and phrases in a natural way. However, to succeed, these approaches require a high quality title to appear somewhere verbatim on the web page, which is rarely the case. As will be discussed in Section 3, in more than 80% of cases in our user study, users did not consider any extracted text snippets as the best representation of the table. The caption tag, which is supposed to provide a title of the table, is used in widely different ways across table authors, and moreover, is present in only less than a few percentage of web tables.

When existing text snippets *are* relevant to a table, they often each contain only a portion of the ideal title for that table. Consider, for example, the titles shown in Figure 3 and Figure 5—all three titles contain text from at least three separate locations on the page. This highlights the need for compositionality in any title generation approach, which most IR-based approaches do not support.

Another approach for title production would be to mine query logs for past user queries that led to that table and use these as candidate titles [5]. Using human-generated text increases the likelihood of the title being compositional (since it is not restricted to text snippets from the page), but once again, the success of the approach is dependent on a high quality title existing *a priori* for selection. For tables with few corresponding queries, success is unlikely. Furthermore, for web pages with multiple tables in them, the task of linking a query that led to that page with the most relevant table on the page (if any) is non-trivial.

1.3 Proposed Approach

In this paper, we propose a new framework for generating high quality table titles. Our approach utilizes a sequence-to-sequence neural network model with both a copy mechanism and a generation mechanism. By generating titles instead of selecting or ranking existing text strings, this approach is capable of composing high quality title strings even when none exist in the source text. At the same time, rare and OOV tokens are retrievable thanks to the copy mechanism (see Figure 3, for example). Furthermore, as a high-capacity machine learning model, with sufficient training data the model is able to learn and utilize more nuanced patterns than most

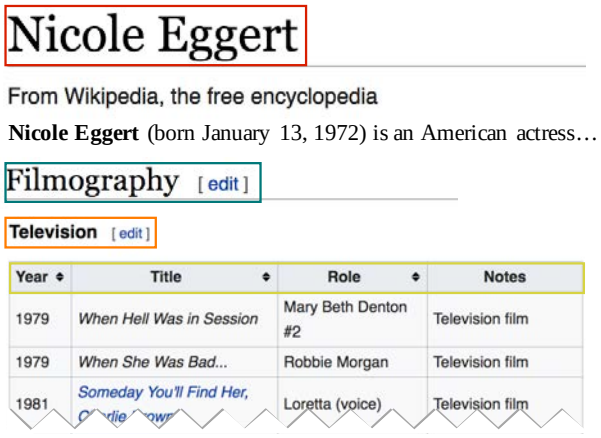


Figure 3: On the left is the top portion of a table from the web along with preceding section headings (some content has been removed for clarity). On the right are the input string (split into multiple lines for readability) and output of the trained title generation network. The double underscores around "nicole" and "eggert" denote that these are out-of-vocab (OOV) tokens (i.e., they were not present in the training data, and can therefore only be copied, not generated). Note that the generated title is composed from multiple fields (but not all fields), irrelevant tokens such as "[edit]" and "- wikipedia" are not copied, and while "role" may help to suggest the schema of this table, the token "roles" was actually generated, reflecting a learned pattern that titles tend to refer to plural objects.

heuristic-based approaches, making it more robust to the variety of tables.

The contributions of this paper are as follows:

(1) We identify the key components for successful table title generation (semi-structured data representation, composition from multiple sources, and handling rare and OOV tokens) and propose techniques for dealing with each in the context of web tables.
(2) We explore qualitatively and quantitatively the value of having both copy and generation mechanisms in a model for table title generation, finding that both are necessary to surpass simple baseline performance.
(3) We propose a framework for generating high quality titles for web tables and find that both by automated metrics (ROUGE scores) and human evaluation on a held-out test set, our model outperforms selection-based baselines and sequence-to-sequence models with either a generation mechanism or a copy mechanism, falling just slightly below human-generated titles on relevance and readability.

2 WEB TABLE TITLE GENERATION

2.1 Web Tables

The web is filled with tables. In a seminal 2008 study, researchers at Google extracted 14.1 billion HTML tables from Google's general-purpose web crawl and released initial findings on the properties and challenges associated with this corpus [4]. Since that publication, hundreds of papers have been published on various applications associated with web tables, including table detection, table classification, relation extraction, knowledge base construction, data cleaning, query answering using tables, and many others [2, 16, 23, 33, 36, 37, 39]. To this important body of work we add our own contribution of title generation for web tables.

Figure 3 shows excerpts from the top portion of a web page that has a table in it. As a human, it is easy to recognize that this table shows television appearances by Nicole Eggert. Note, however, that only one of the words in the target title for this table actually occurs within the table. Instead, the meaning of the table is distributed among many different elements on the page. This raises the important question of what context should be considered when creating a title for a table.

One option would be to include the entire web page. This would be guaranteed to include any text that could possibly be relevant for titling a table; unfortunately, it would almost certainly also include a significant amount of unrelated text, decreasing the signal-to-noise ratio of the input. The task of identifying pertinent information in a document is its own open research problem of significant complexity. Additionally, the sequential nature of encoding information with a sequence-to-sequence models means that training time is proportional to the length of the sequence being encoded. Computation overheads for encoding tens or hundreds of thousands of tokens as input is likely prohibitive for our task.

Empirically, we know that the most relevant information for titling a table tends to come from a relatively small set of potential text sources. One obvious source of such text is the contents of the *caption* tag within a table element. However, less than 5% of tables for which we would like to generate titles have explicit captions (including the one in Figure 3). Other relevant sources outside the table itself include the page title or section headings. While these certainly have greater coverage than captions, they also introduce new sources of ambiguity. Both page titles and section headings

Table 1: Metadata fields considered for inclusion in the input to the title generation model. Field names followed by * were included for the experiments reported in Section 4. Where multiple values are found, we maintain the list of values in reading order.

Field	Description
Page title*	Tokens inside the <title> tag nested in the <head> tag of the web page
Section headings*	Tokens in <h1>, <h2>, etc. tags with increasing priority, starting with the nearest
Table captions*	Captions inside <caption> tags
Spanning headers*	Headers in <th> tags that span all columns
Column headers*	Headers in <th> tags
Prefix text	Up to 200 tokens preceding the table until a new table or section boundary
Suffix text	Up to 200 tokens following the table until a new table or section boundary
Table rows	Text inside a <tr> tag, comma delimited with other tags (e.g., <td>) removed

may be only loosely related to the content of an associated table or they may refer to multiple tables on the page which actually have important distinctions between them.

In addition to these text snippets which tend to show up in their own set of HTML tags (e.g., <title> or <h1>), spans of unstructured text in sentences surrounding a table may also include important contextual clues. For example, in Figure 3, the first sentence on the page mentions that Nicole Eggert is an actress, which clearly pertains to the displayed table that lists television shows in which she has performed. While relevant text that alludes to the table may theoretically appear anywhere in the document, to reduce complexity and runtime, we use the heuristic of looking only at the unstructured text immediately preceding or following a table.

2.2 Representing Semi-Structured Data

Once we know what information we would like to include in the input to our model, the question arises of how to represent this information. Drawing on the rich and rapidly expanding body of work on encoder-decoder frameworks, we chose to model these structured fields as a sequence and found that table title generation lends itself well to such an approach. The output—a title in natural language—is clearly sequential in nature. And the input, a combination of textual elements associated with the table, exhibits a sequential nature on a couple different levels. First, the text within each field is ordered as it reads left to right. And second, as demonstrated in Figures 3 and 5, the metadata fields that we collect for each table tend to increase in specificity as they get nearer to the table. Just as they were intended to be read in that order on the web page, they tend to remain in that order in high quality titles.

For each table in our dataset, we collected the following metadata fields (which are also listed in Table 1 and described in greater detail below): page title, section headings, table captions, spanning headers, column headers, prefix text, suffix text, and table rows (the contents of the table, one row at a time). Then, to convert these fields (key-value pairs) into a sequence, we iterated through the fields in the order just given and concatenated their contents, preceding each value (one or more tokens) with a special field token of the form "#[field name]". The model treats these field tokens the same as any other, eventually learning to encode information differently based on what field it is reading and to never generate these field tokens in the decoder, since they never appear in the training data titles. We now describe in greater detail how each of these fields were collected.

Page title: Page titles are found inside the <title> tag in the <head> section of a web page. This title is not directly displayed on the web page in a browser, so many web pages repeat the page title as a section heading for the user to see (as in Figure 3). Given their applicability to the contents of the entire page, page titles proved to be one of the most consistently relevant fields for generated titles. While each page has only one page title, many page titles effectively concatenate multiple titles with delimiters, as in "Chocolate Gifts | Artisan Truffles | Gourmet Chocolate".

Section headings: We chose to rely primarily on explicit header tags (<h1>, <h2>, etc.) for section headings, using the numeric levels of the headers as proxies for parent-child and sibling relationships between sections. Rather than storing all headers preceding a table, we worked backward through the document, starting at the beginning of the table and ending at the beginning of the document. In this traversal, we saved only the first header (closest to the table) at each level that had higher priority (lower numeric value) than any headers collected so far (i.e., if a heading of type <h3> had already been observed, then all section headings of type <h#> where # $\geq$ 3 were skipped from then on). Because not all web pages use these tags to denote headers, however, (relying instead on font size, bold formatting, centering, etc.), not all section headings that a human might identify were always collected. The problem of identifying document structure from visual information is an open research problem.

Table captions: While table captions (in explicit <caption> tags) tended to be highly relevant, they were relatively rare ($< 5\%$[1] of tables) and rarely sufficient as a title. Instead, captions often either provided the final piece of information in a long chain of text snippets that together form an adequate title (as in Figure 5), or they represented a long caption (sometimes multiple sentences) as in, for example, scientific literature. Regardless, if a table had a caption, some portion of it tended to show up in both crowd-sourced and generated titles.

Column headers: Column headers are table cells denoted with <th> tags (instead of the <td> tags for table data). Most tables had only one row of column headers, but not all. Column headers tended to be short, often just one or two words each.

[1]The statistics in the paper are from samples in our internal corpus unless specified.

2016 Olympic Medal Count

Rank	Country	Total Medals
1	United States	121
2	Great Britain	67
3	China	70
4	Russia	55
5	Germany	42

GDP by Country (Millions)

Rank	Country	GDP ($ Millions)
1	United States	18.6
2	China	11.2
3	Japan	4.9
4	Germany	3.5
5	United Kingdom	2.6

Global Average Internet Speeds

Rank	Country	Speed (Mb/s)
1	South Korea	26.7
2	Sweden	19.1
3	Norway	18.8
4	Japan	17.4
5	Netherlands	17.0

Figure 4: The contents of non-header rows of tables contribute surprisingly little information toward their titles tables. Consider these three tables with very similar non-header contents but very different topics and titles.

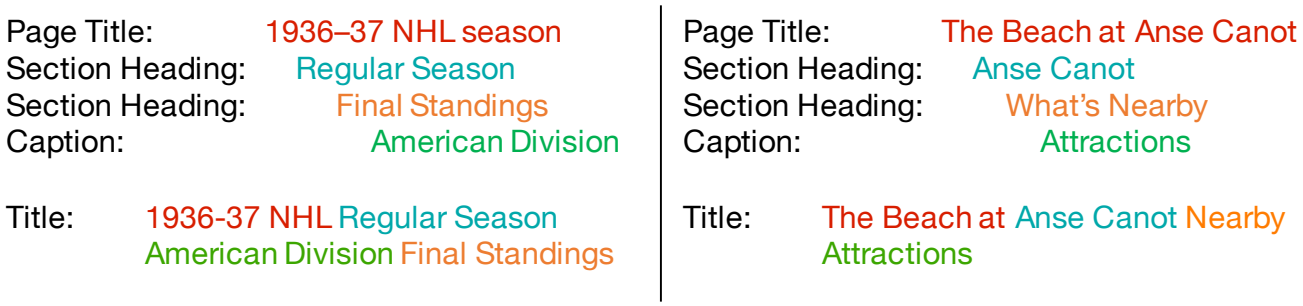


Figure 5: For the majority of tables, the ideal title is composed from multiple associated text snippets, rather than selected from among them.

Spanning headers: Spanning headers are a special type of column header—one which spans all columns in the table. Some tables use these effectively as table titles, just placed inside the table instead of above it. Spanning headers were even rarer than captions, showing up in < 1% of tables.

Prefix/suffix text: Prefix and suffix text are the tokens immediately before and after the table, respectively. They consisted of up to 200 tokens, stopping short at the boundary of another table or another section, since such boundaries tend to signal a change in topic. In practice, we found the prefix and suffix text to be quite noisy, rarely providing relevant title components not already included in the other fields, while significantly increasing the size of the input sequence. Consequently, these fields were not included in the dataset used to train the models for which we report results.

Table rows: Table rows refer to contents of all non-header rows in a table. We were surprised at first to observe that including table rows actually hurt model performance, but upon closer inspection, we recognized that individual table records rarely include unique information that appears in a good title, and topical information expressed in the columns is generally captured by the column headers. Thus, the final dataset did not include the table rows field. Figure 4 illustrates this finding with three tables that have very similar contents in their non-header rows, none of which contribute much to their respective titles.

2.3 Supporting Compositionality

Because of the inherent complexity of generating new text, our initial attempts to solve the problem of titling tables were selection-based. However, we soon found that the upper bound of such an approach was far too low, due to the frequency with which no adequate title could be found in a single continuous span of text. This observation was further confirmed during the collection of the crowdsourced dataset we describe in Section 3. For each human-generated title that we collected, we had the worker denote whether the optimal title that they generated occurred anywhere on the page in its entirety; 83% of the time, it did not.

As an illustration of this trend, consider the two collections of text snippets shown in Figure 5. Each corresponds to a single table whose human-generated title is also given. For both tables, all four snippets are relevant, but it is only when these pieces are composed that a sufficiently correct title is created.

2.4 Handling Rare and OOV Tokens

Successfully handling rare and OOV tokens is a challenge for most text generation tasks. This problem is exacerbated for the task of table title generation, due to the relational nature of tables; Tables tend to describe specific entities and quantities rather than abstract ideas. A generic description like "people shopping at an outdoor

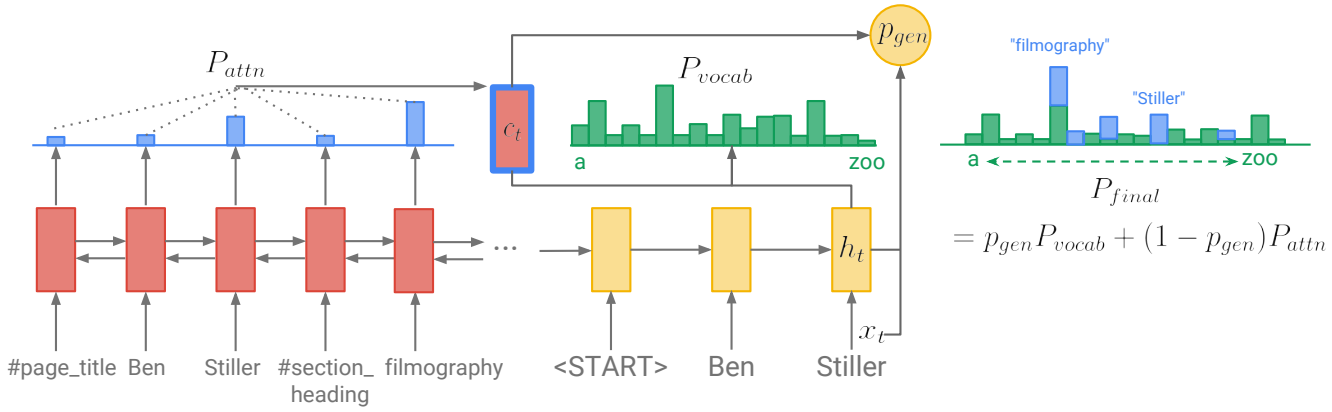


Figure 6: At each decode step, the pointer-generator network calculates the scalar value p_{gen}, which determines the relative contributions of the *vocabulary distribution* (tokens that can be generated) and the *attention distribution* (tokens that can be copied from the input sequence). Beam search is performed over the *final distribution*.

market" may work well for images, but generic titles like "countries and numbers" for tables in Figure 4 would not be helpful. We observe the proportion of specific entities to generic terms is much higher in the very information-dense format of tables. For example, in the validation split of our crowdsourced dataset (approx. 1000 titles), 90% of titles contained one or more proper nouns, and 45% contained one or more OOV words (for a vocab based on over 8000 tables plus titles).

To overcome this stumbling block, we included a copy mechanism in the decoder of our sequence-to-sequence model. A copy mechanism allows a decoder network to not only generate tokens from a known vocabulary, but also occasionally copy a token directly from the input. The parameters controlling when a token is copied versus generated are also learned. Consequently, the model is able to learn specific patterns from the training data that suggest that the following word in a title is likely to be a rare or OOV word that should be copied rather than generated. For example, after producing the tokens “Action Films Starring,” with high probability, the next generated token will be the name of an actor or actress, of which there are many, and whose embeddings are likely very close to one another. However, that actor’s or actress’s name will almost certainly appear in the metadata associated with the table (e.g., in the page title or a section heading). In this case, copying over one of the unknown tokens that is being used like a name in the input (where “being used like a name” is a soft pattern the network naturally learns) is far more likely to succeed than generating a token.

2.5 Model Selection

The model architecture we used was a pointer-generator network, first introduced by [25] as a tool for abstractive summarization of documents into a few sentences. The pointer-generator network utilizes a bi-LSTM encoder and LSTM decoder with attention. At each decode step, the scalar value $p_{gen} \in (0, 1)$ is calculated with a linear layer as follows:

$$p_{gen} = \sigma(W_c c_t + W_h h_t + W_x x_t) \quad (1)$$

where c_t is the context vector (a weighted sum of encoder hidden states, using the attention distribution), h_t is the decoder hidden state, x_t is the decoder input, and W_c, W_h, and W_x are learnable parameters. The scalar p_{gen} is used to weight the relative contributions of the vocabulary distribution P_{vocab} (i.e., the distribution that is sampled in a traditional sequence-to-sequence generation model) and the attention distribution P_{attn} to the final distribution P_{final}, which is defined for each token w in the union of the vocabulary and the input tokens:

$$P_{final}(w) = p_{gen}P_{vocab}(w) + (1 - p_{gen})P_{attn}(w) \quad (2)$$

From this distribution, the decoder samples with beam search the next token to produce (as shown in Figure 6).

Our loss function was the average negative log likelihood of the generated sequence. We used the Adagrad optimizer [9] with an initial learning rate of 0.15, gradient clipping of 2.0, and early stopping on a validation set to control overfitting. Word embeddings (128-dimensional) were randomly initialized and learned as the model was trained. LSTM hidden states (256-dimensional) were initialized with random uniform initialization up to magnitude 0.02. We trained with mini-batch size of 64 and decoded using beam search (beam size = 8) with a minimum decode length of 4 tokens and a maximum of 20 tokens. The input text was truncated at 150 tokens. We did not threshold the vocabulary, which was approximately 18k in size. During training we calculated ROUGE-1, ROUGE-2, and ROUGE-L scores as proxies for title quality. We report these numbers as well as readability and relevance scores provided by human evaluators in Section 4.

2.6 Reducing Token Duplication

An artifact of using the negative log-likelihood objective function is that we often saw the same token being generated back-to-back. Intuitively, we can understand this as the algorithm being quite confident that a particular token will show up somewhere in the true title, but because there are many ways to paraphrase a functionally equivalent title, it does not know precisely where to place that token to minimize loss. Consequently, it generates the same

token in multiple places to increase its chance of being correct in one of them (e.g., it generates "Highest Salaries NBA NBA NBA" instead of "Highest Salaries in the NBA" or "Highest Salaries of NBA Players"). While this may decrease the loss function, it also decreases readability.

The original pointer-generator network paper proposed a *coverage* mechanism for reducing duplication of phrases in generated summaries [25]. The coverage mechanism works by summing the attention distribution over all previous decoder timesteps and penalizing those tokens with high values. We find that with titles being much shorter than article summaries, a simple heuristic serves the same purpose while simplifying the model and requiring fewer parameters to learn: we simply forbid the algorithm from generating the same token twice in one title by zeroing out its probability in the final distribution in all future decode steps. While this heuristic is not foolproof (e.g., "La La Land Awards"), we find that over 95% of the titles in our dataset have no repeated tokens, and many with repeated tokens can easily be paraphrased by the model to avoid the repetition while remaining relevant. Including this heuristic yielded an instant 4.5 point boost in ROUGE score. With more training data, a more sophisticated approach for reducing token duplication may yield further improvements.

3 DATASET CREATION

3.1 Crowdsourced Dataset

The crowdsourced dataset consists of 10,102 web tables scraped from the tables returned as featured snippets [27] to user queries on Google over a span of five months from January-May 2017.

The tables were each shown in their original context to three trained crowdworkers, who were asked to provide a descriptive title for that table and to mark whether that title occurred verbatim anywhere on the page or was composed. Based on a manual inspection of the candidate titles for 100 random tables in our dataset, we found qualitatively that the most informative and relevant title (of the three provided by crowdworkers) was composed 83% of the time. The three candidate titles for each table were then aggregated using the following heuristic: if two or more of the title are identical, accept that title; otherwise, select the title with the most tokens (since the most common failure mode for generated titles was a lack of detail). From a sample of 50 tables, this heuristic chose what we considered the best title 67% of the time. Finally, the titled dataset was split 80/10/10 into train/validation/test splits.

The dataset contains tables from 1384 different domain names, where two major domains were wikipedia.org (72.6%), and espn.com (0.8%). The dataset covers diverse topics including news (e.g., bbc.co.uk, times.com), health (e.g., webmd.com, nih.gov), entertainment (e.g., boxofficemojo.com, allmusic.com), and corporations (e.g., microsoft.com, ibm.com). The average accepted title length is 40.9 characters or 6.8 tokens. The total vocabulary size is 17,862.

3.2 Heuristic-based Datasets

The first heuristic-based approach we tried was proposed in [5]: mining *query logs* for queries that led to pages with tables on them and using the queries as candidate titles for those tables. In addition to general quality issues with the queries as titles, we found it difficult to reliably link a given query to the precise table on a page it is most relevant to when multiple tables are found on the same page. Restricting the dataset to those tables that are the dominant tables on a page biased the dataset toward tables that are the central topic of the page, introducing an over-reliance on the page title as a source of title tokens.

Next we tried using *table captions* as titles for those tables that had them, but as discussed in Section 2.3 and demonstrated in Figure 5, these often contained only the last link in chain of necessary text spans for generating a good title. Hoping to identify a set of more specific captions, we tried filtering the dataset to *captions with three or more entities*. This had the result of overrepresenting certain domains (e.g., scientific papers, code documentation, large online catalogs) that dominated the training set. With all of these caption-based approaches, we faced the additional issue that by removing the caption from the training input (to prevent the model from learning to simply copy over the caption word-for-word), we sometimes removed the only mention of essential information required to generate a proper title. Our last attempt was to use as titles the *table captions with non-stopwords from multiple different sources* on the page and all of whose tokens could be found in at least one other input field. This, unfortunately, resulted in a dataset that was too small to use.

From these attempts, it became clear to us that given the immense variety of web tables, we would be hard-pressed to find any heuristic that could generate a training set of sufficient size without introducing significant harmful bias. We consequently commenced with collecting the crowdsourced dataset described above.

4 EXPERIMENTAL EVALUATION

We validated the quality of our approach with a human evaluation on 200 tables from the held-out test split of the dataset. We created two baselines that simply select the page title or nearest section heading, respectively, with a few obvious post-processing steps, such as removing "[edit]" from wikipedia section headings. We ran the pointer-generator network in *Copy Only* mode (p_{gen} hardcoded to 0 so no tokens are generated), *Generate Only* mode (p_{gen} hardcoded to 1 so all tokens are generated, as in a standard sequence-to-sequence model), and without modification (*Copy + Generate*). Finally, we also evaluated the quality of the best crowdsourced title for each table as an effective upper bound.

Evaluators assessed the readability and relevance of each title on a 3-point scale with the following interpretations: for readability, 1 = Poor, 2 = Medium, 3 = Well. For relevance, 1 = Needs Not Met, 2 = Needs Somewhat Met, 3 = Needs Fully Met. Experimental results are shown in Table 2.

Prior to collecting a dataset via crowdsourcing (Sec. 3.1), we attempted to bootstrap a training dataset using four different heuristics (Sec. 3.2). In each case, we found that the heuristic we used biased the training data enough to significantly hinder its ability to generalize. Consequently, the results reported in this paper all correspond to models trained with the crowdsourced data. Interestingly, while *Copy + Generate* had the strongest performance, *Copy Only* and *Generate Only* had the weakest, with each performing much better in one of the two metrics than the other, as shown in Figure 2. Since both readability and relevance are required in any real-world use case, it makes sense that historically, simpler selection-based

Table 2: Comparison of relevance and readability (from human evaluation) and ROUGE metrics for 200 titles. Results are grouped by category (baselines, neural networks, crowdsourced) and shown graphically in Figure 7. The crowdsourced titles have no ROUGE scores because they are the ground truth.

Model	Relevance	Readability	ROUGE-1	ROUGE-2	ROUGE-L
Page Title	2.25	2.41	0.510	0.369	0.461
Section Heading	2.29	2.56	0.476	0.315	0.411
Generate Only	1.08	2.25	0.168	0.064	0.151
Copy Only	1.97	1.26	0.384	0.221	0.240
Copy + Generate	**2.56**	**2.66**	**0.647**	**0.485**	**0.574**
Crowdsourced	2.72	2.74			

approaches have been preferred to generation-based ones. Combining the two mechanisms, however, results in a model that performs better than both single-mechanism models in all metrics.

Generate Only: We observe that the *Generate Only* model produced much more readable output than *Copy Only*, but scored extremely low on relevance. The failure of *Generate Only* seems to stem from the fact that because of the prevalence of rare and OOV tokens, the model was unable to learn sufficiently precise embeddings to convey the required information to the decoder. It did, however, appear to capture general topics. For example, when trying to produce the title "Jyotii Sethi Filmography," it produced "List of Movies with Sue Perkins" instead—both describe a list of a particular actress's movies, but clearly the model lacked the exposure to the tokens "Jyotii Sethi" to either include them in its vocabulary or learn precise enough embeddings for them.

Copy Only: On the other hand, the *Copy Only* model produced more relevant output—often producing very specific words pertaining to the table at hand—but had a much harder time composing text snippets in a readable way. Intuitively, this is because the metadata (e.g., section headings, column headers) associated with semi-structured data such as lists and tables often lack the prepositions, conjunctions, and other transition words necessary to combine multiple pieces of information. As a result, titles often consisted of snippets concatenated in unnatural ways.

Copy + Generate: Examining the output of the *Copy + Generate* model, we found many instances where a combination of both copied tokens and generated tokens was used to great effect. For example, many tables in our dataset report movie or television appearances of an individual. A common title pattern for these is "Ash King Filmography" (if the table is about Bollywood singer Ash King). Instead, for that table, the *Copy + Generate* model produced the title "Filmography of Ash King". While it may have been a mistake to generate "Filmography" first in the title, because it can generate filler words and had learned a reasonable language model, the model was able to recover from this misstep by adding the preposition "of", which actually did not occur anywhere in the input. This simple example illustrates the power of combining these two mechanisms in a single model.

It is worth noting that while the absolute difference between *Copy + Generate* and the *Section Heading* baseline is not large, the learned model cuts the gap to human-generated performance by more than 50% in both readability and relevance. This corresponds to a quite noticeable difference from a reader's standpoint. Additionally, the task of generating focused natural language is one in which performing moderately is not too difficult, but performing well enough to produce genuinely usable text is very challenging.

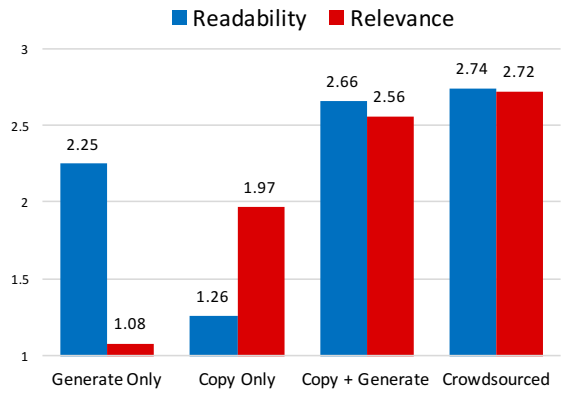


Figure 7: The poor performances of the *Generate Only* and *Copy Only* variants highlight the necessity of having both mechanisms in a single model to produce high quality titles.

5 RELATED WORK

The approach of flattening structured fields into a sequence was also recently taken by [6], who used it for the related task of generating one-sentence biographies from Wikidata facts. Instead of employing a copy mechanism, they add an autoencoder objective to their sequence-to-sequence model to encourage relevance in generated text. Other attempts to summarize structured web content include generating text summaries from individual knowledge base triples [32], generating descriptions of restaurants from structured attributes using rule-based [24] or learned [34] templates, and generating titles for web pages with e-commerce search results given a set of attributes shared by all items on the page [20].

In addition to pointer-generator networks, multiple other types of networks with copy mechanisms have been proposed [10, 11, 15, 31], each with minor differences affecting when and how the network chooses to generate versus copy.

Another way to attempt to deal with rare and OOV words is to learn over sub-word units, using character-level embeddings [7, 13, 18], byte-pair encodings [26], or WordPieces [35], for example. One reason why these methods often work well is because they are somewhat robust to variation in prefixes or suffixes of words, making it easier to recognize the similarity between words like "slower" and "slowly" compared to a word-based approach with independent embeddings for each. In the table title domain, however, we found that the most common source of OOV words was uncommon named entities, which reduces the benefit of sub-word embeddings.

For long document summarization, [22] also reduced repetition by forbidding duplicate n-grams (in their case, trigrams). They also add decoder attention (to increase awareness of which tokens have already been generated) and intra-temporal attention on the input (similar to the coverage mechanism of [25]), which penalizes input tokens that have received high attention scores in the past in hopes of limiting the number of time steps that a given token can influence.

The question of how to best evaluate Natural Language Generation (NLG) systems is one of ongoing debate. While automated methods such as BLEU and ROUGE scores are fast and cheap, these have been found to show surprisingly weak correlation with human ratings [21] for many tasks and domains. (We see this in our own dataset as well, where the *Page Title* baseline outperforms *Section Heading* in all three ROUGE metrics, yet underperforms it in both human evaluation metrics). In general, human evaluation is still the preferred method for assessing final system quality where possible [3]. Nevertheless, as they are a standard point of comparison, we report ROUGE scores as well.

6 FUTURE WORK

Surprisingly, the pointer-generator network produces effective titles when trained with just 8k examples, a much smaller dataset than is typically used in text generation tasks. Empirically, this appears to be because the model focuses more on learning when and where to "copy and paste" than it does on learning accurate embeddings for all words in the vocabulary. Using the fact that words of similar parts of speech or entity types will tend to be used in the same ways in titles, in follow-up work we intend to explore the benefits of injecting extra syntactic/semantic information (such as part of speech or named entity tags) directly into the input stream. We anticipate that this will further improve performance while maintaining low training data requirements.

As we reported in this paper, we found prefix text and suffix text to have too low signal-to-noise ratios (i.e., too many irrelevant words compared to relevant ones) to justify their presence in the input to our model. However, certain key phrases in these fields seem to strongly indicate that the surrounding words will be relevant for the title—phrases such as "the following table" and "as shown below". We suspect that a model better equipped to handle long input sequences (such as an LSTM with skipping [38] or a Transformer [30]) may be able to better utilize fields such as these, resulting in a richer input sequence from which to generate a title. Additionally, one might take advantage of the increasingly prevalent structured metadata embedded in webpages in formats such as RDFa, Microdata and JSON-LD.

7 CONCLUSION

In this work, we presented a new framework for generating titles for web tables. We found that for the majority of web tables, the ideal title must be composed, rather than selected, from the text snippets associated with the table. We trained a sequence-to-sequence model to generate titles from input sequences consisting of the flattened key-value pairs of metadata associated with web tables. The best performing model utilized a copy mechanism to make titles relevant (often copying rare and OOV words directly from the input), and a generation mechanism to make titles readable (drawing on a learned language model for titles). In a human evaluation, the model with both copy and generation mechanisms outperformed all other models, approaching the quality of crowdsourced titles. We believe this establishes a new state of the art for table title generation.

ACKNOWLEDGMENTS

The authors would like to thank Quoc Le and Jialu Liu for their helpful feedback over the course of this project.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR* (2015).
[2] Sreeram Balakrishnan, Alon Y. Halevy, Boulos Harb, Hongrae Lee, Jayant Madhavan, Afshin Rostamizadeh, Warren Shen, Kenneth Wilder, Fei Wu, and Cong Yu. 2015. Applying WebTables in Practice. In *CIDR*.
[3] Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of NLG systems. Association for Computational Linguistics.
[4] Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment* 1, 1 (2008), 538–549.
[5] Fernando Chirigati, Jialu Liu, Flip Korn, You Will Wu, Cong Yu, and Hao Zhang. 2016. Knowledge exploration using tables on the web. *Proceedings of the VLDB Endowment* 10, 3 (2016), 193–204.
[6] Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from Wikidata. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1* (2017), 633–642.
[7] Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A character-level decoder without explicit segmentation for neural machine translation. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (2016), 1693–1703.
[8] Eric Crestan and Patrick Pantel. 2011. Web-scale table census and classification. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 545–554.
[9] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.
[10] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (2016), 1631–1640.
[11] Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (2016), 140–149.
[12] Amazon.com Inc. 2019. Amazon Echo. https://www.amazon.com/Amazon-Echo-And-Alexa-Devices/b?ie=UTF8&node=9818047011. (2019). Accessed: 2019-02-09.
[13] Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics* (2017).
[14] Oliver Lehmberg, Dominique Ritze, Robert Meusel, and Christian Bizer. 2016. A large public corpus of web tables containing time and context metadata. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 75–76.
[15] Liunian Li and Xiaojun Wan. 2018. Point Precisely: Towards Ensuring the Precision of Data in Generated Texts Using Delayed Copy Mechanism. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1044–1055.

[16] Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. 2010. Annotating and Searching Web Tables Using Entities, Types and Relationships. *VLDB* 3, 1-2 (2010), 1338–1347.

[17] Google LLC. 2019. Google Home. https://store.google.com/us/product/google_home. (2019). Accessed: 2019-02-09.

[18] Minh-Thang Luong and Christopher D Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (2016), 1054–1063.

[19] Minh-Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare word problem in neural machine translation. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing* (2015), 11–19.

[20] Prashant Mathur, Nicola Ueffing, and Gregor Leusch. 2017. Generating titles for millions of browse pages on an e-Commerce site. In *Proceedings of the 10th International Conference on Natural Language Generation*. 158–167.

[21] Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why We Need New Evaluation Metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2231–2242.

[22] Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A Deep Reinforced Model for Abstractive Summarization. *arXiv preprint arXiv:1705.04304* (2017).

[23] Rakesh Pimplikar and Sunita Sarawagi. 2012. Answering Table Queries on the Web Using Column Keywords. *VLDB* 5, 10 (2012), 908–919.

[24] Yevgeniy Puzikov and Iryna Gurevych. 2018. E2E NLG Challenge: Neural Models vs. Templates. In *Proceedings of INLG*.

[25] Abigail See, Peter Liu, and Christopher Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. 1073–1083.

[26] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (2015), 1715–1725.

[27] Danny Sullivan. 2018. A reintroduction to Google's featured snippets. (2018). https://www.blog.google/products/search/reintroduction-googles-featured-snippets/

[28] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[29] Microsoft The Bing Team. 2017. Bing launches new intelligent search features, powered by AI. (2017). https://blogs.bing.com/search/2017-12/search-2017-12-december-ai-update

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[31] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*. 2692–2700.

[32] Pavlos Vougiouklis, Hady Elsahar, Lucie-Aimée Kaffee, Christophe Gravier, Frederique Laforest, Jonathon Hare, and Elena Simperl. 2018. Neural wikipedian: Generating textual summaries from knowledge base triples. *Journal of Web Semantics* (2018).

[33] Chi Wang, Kaushik Chakrabarti, Yeye He, Kris Ganjam, Zhimin Chen, and Philip A. Bernstein. 2015. Concept Expansion Using Web Tables. In *WWW*.

[34] Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2018. Learning Neural Templates for Text Generation. *arXiv preprint arXiv:1808.10122* (2018).

[35] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR* (2016).

[36] Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, and Surajit Chaudhuri. 2012. InfoGather: Entity Augmentation and Attribute Discovery by Holistic Matching with Web Tables. In *SIGMOD'12*. 97–108.

[37] Mohan Yang, Bolin Ding, Surajit Chaudhuri, and Kaushik Chakrabarti. 2014. Finding Patterns in a Knowledge Base using Keywords to Compose Table Answers. *VLDB* 7, 14 (2014).

[38] Adams Wei Yu, Hongrae Lee, and Quoc V Le. 2017. Learning to Skim Text. *arXiv preprint arXiv:1704.06877* (2017).

[39] Cong Yu. 2012. Towards a High Quality and Web-Scalable Table Search Engine. In *KEYS'12*. 1–1.

Understanding the Effects of the Neighbourhood Built Environment on Public Health with Open Data

Apinan Hasthanasombat
University of Cambridge
United Kingdom
ah953@cam.ac.uk

Cecilia Mascolo
University of Cambridge
United Kingdom
cm542@cam.ac.uk

ABSTRACT

The investigation of the effect of the built environment in a neighbourhood and how it impacts residents' health is of value to researchers from public health policy to social science. The traditional methods to assess this impact is through surveys which lead to temporally and spatially coarse grained data and are often not cost effective. Here we propose an approach to link the effects of neighbourhood services over citizen health using a technique that attempts to highlight the cause-effect aspects of these relationships. The method is based on the theory of *propensity score matching with multiple 'doses'* and it leverages existing fine grained open web data. To demonstrate the method, we study the effect of sport venue presence on the prevalence of antidepressant prescriptions in over 600 neighbourhoods in London over a period of three years. We find the distribution of effects is approximately normal, centred on a small negative effect on prescriptions with increases in the availability of sporting facilities, on average. We assess the procedure through some standard quantitative metrics as well as matching on synthetic data generated by modelling the real data. This approach opens the door to fast and inexpensive alternatives to quantify and continuously monitor effects of the neighborhood built environment on population health.

CCS CONCEPTS

• **Information systems** → *Spatial-temporal systems*; *Data mining*; *Web mining*.

KEYWORDS

Open data; Population health; Causal inference, Propensity score.

ACM Reference Format:
Apinan Hasthanasombat and Cecilia Mascolo. 2019. Understanding the Effects of the Neighbourhood Built Environment on Public Health with Open Data. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313701

1 INTRODUCTION

The effects of urban neighbourhood characteristics on health outcomes of its residents is an ongoing research topic [2]. With over half of the population living in urban areas and a projected increase to 68% by 2050 [31], understanding exactly how these neighbourhood aspects impact different disease outcomes is a significant public health concern. Traditional research into the risk factors of many diseases focus mainly on 'direct causes' - those that are close to the biological mechanisms that are responsible for the development of the disease. For instance, high cholesterol and coronary heart disease. However, in order to deliver maximum impact when designing interventions that prevent diseases in the population at large, it has been argued that we should also focus on what puts people at *'risks of risks'* [15, 27] - the factors that lead people to be exposed to the direct causes of disease in the first place. These, which we call 'higher level causes', for instance social-economic status (SES) or the availability of services and amenities in a neighbourhood environment, are useful from a public health point of view, as they can be influenced directly by policy, as opposed to an individual's diet or lifestyle choices.

Previous work on neighbourhood effects on health mainly consists of observational cross-section studies, with the SES of the neighbourhood as the most studied variable, and Body Mass Index (BMI) as the most studied health outcome [2]. We find two main issues. Firstly, a large proportion of these works rely on traditional data sources such as surveys which can be quite expensive at scale and therefore lead to limited geographical and time span coverage. This focus on traditional data sources also means aspects such as the availability of services and amenities in the neighbourhood are neglected, and studies to understand its impact on residents' health are few, even though the built environment was found to have a consistent association with for example, depression [18], amongst neighbourhood structural features. Some previous work have for instance looked at food venue density and BMI [34], and tobacco store density and life expectancy [8]. Secondly, these discussions have mainly revolved around associations, and the daring leap to attempt at a causal conclusion are few. Since these physical aspects of a neighbourhood are controllable and can be planned, this has enormous potential to help inform public policy decisions. The data offered by projects such as Open Street Map (OSM), combined with the growing availability of causal methodology, turns this potential closer to a reality at scale.

In this paper we try to leverage this available open data offering fine grained spatio-temporal granularity and a causality framework which takes us closer to eliciting causal effects. There are advantages to this approach. The use of open data offers a scalable and cost effective solution in addition to the granularity in comparison with survey methodologies. Additionally, physical characteristics are better defined and quantifiable compared to measures such as SES. Specifically, and to illustrate the approach, we investigate the availability of sporting facilities and its potential causal effect on

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313701

antidepressant prescriptions across over 600 neighbourhoods in London over three years.

In summary, the contributions of this paper are:

(1) A general cost effective approach to study the influence of the built environment and neighbourhood characteristics, and the mechanisms that underlie this influence such as access to tobacco, certain types of food, or sports, over population health.
(2) We work on a specific example, exploiting urban open data to show that the effect of sporting facilities on antidepressant prescriptions in London.
(3) Our analysis, using data from Open Street Map (OSM), the UK National Health Service (NHS) prescriptions and the census, shows that the effect of sport venues over prescriptions for antidepressants follows an approximate normal distribution centred at a negative effect for prescriptions with increases in sporting facilities, on average.
(4) We assess our approach by examining the results from the procedure for balance, as well as comparing with results from the null model and of synthetic datasets modelled based on real data, where we have knowledge of the real effect.

This approach opens the door to new methodologies to study urban health and to offer urban planners mechanisms to study these geo-temporal processes at a fine grained scale. We first give a general overview and intuition behind the approach, followed by a more detailed explanation. Section 5 and 6 then describes the results of our analysis on a specific case study and our evaluation. We conclude by discussing the implications of our investigations, its limitations, and potential.

2 RELATED WORK

The study of neighbourhood effects on health has grown rapidly over the last two decades [22] [2]. This ranges from grocery store density and BMI [34] to tobacco store density and life expectancy [8]. The neighbourhood variable under study can be split largely into two categories. The tangible aspects of the neighbourhood, such as green spaces, amenities, or venues offering services, and the intangible aspects of the neighbourhood, comprising mostly of social aspects such as social cohesion, social disadvantage, to perceived safety.

The research area dates back to Durkheim's study of suicide in the 19th centuary [5] but the most recent concrete indication of any plausible causal effect is shown by a rare randomised controlled study conducted in the United States which shows that disadvantaged families moving to neighbourhoods with lower poverty rates lead to long term improvements on mental and physical health [17].

Non experimental studies in the literature predominantly feature social characteristics of a neighbourhood [2][22]. However, these intangible variables are often difficult to define, hard to measure and quantify, as opposed to the physical characteristics. Additionally, the traditional methods of surveys lends itself well to the study of social aspects as opposed to the physical, which may explain the lack of studies in the latter area.

Reviews of neighbourhood effects research suggests that apart from a few key studies such as the randomised experiment, evidence is inconclusive regarding many health outcomes [18, 22], and the literature has had its fair share of discussion around the validity of causal estimates from non experimental data [21][12]. Additionally, even though association studies are plentiful, there has been a lack of proposed mechanisms for which high level neighbourhood characteristics can influence health, and of the ones that are proposed, hardly any are related to the tangible aspects of the neighbourhood [4]. This study examines one such potential mechanism, given that the built environment is noted to be consistently associated with depression [18].

There is a glaring opportunity to leverage non-conventional data sources to help characterise neighbourhoods, and an appropriate methodology that will allow this to be done continuously at scale. Alternative data sources, such as those from mobile phone call detail records, social networks such as Twitter, Foursquare, Google Street View and other web data, have been increasingly used in a wide range of applications from estimating poverty levels [30], measuring social diversity [10], to auditing the neighbourhood environment [29].

There exists studies that investigate how these data sources can be used to understand and improve public health. For instance, the association between human lifestyles and chronic diseases, and its use in predicting diseases evolution in urban areas [34]. Another study used a model based on twitter data to predict disease prevalence, and found associations between the different geographical risk factors (e.g tobacco use, exercise) based on phone interview data, with the model prediction [24]. However these are constrained to associations and predictions. Traditional studies attempting to establish causal links on the other hand, such as that between physical activity, fast food environments, BMI, body fat percentage, and waist circumference, noted lack of data on the food environment to be a limitation [19].

In short, we see three potential improvements to the state of neighbourhood effects research: More focus on physical characteristics of the neighbourhood, a deeper investigation into the potential physical causal mechanisms by which the neighbourhood effects health, and scalability of methodology to continuously monitor effects leveraging non-traditional data sources.

This paper illustrates an alternative approach which addresses these concerns and contributes further to the discussion.

3 APPROACH OVERVIEW

We now describe the overall approach. We are interested in studying specific characteristics of neighbourhoods, say the presence of particular services - for instance sporting facilities - over a population health outcome, such as antidepressant prescriptions. The neighbourhood characteristic here is then called the *treatment*. This stems from the fact that we are adopting a causal view; we want to find the effect that applying the treatment would have on the outcome for the average neighbourhood.

Geographical Separation. The units under study are then not specific individuals, but rather neighbourhoods. The adopted definition of a neighbourhood in this study is that of a *ward*, an administrative geographical separation, of which, for instance, London has approximately 625. At the beginning of the calendar year, each ward is considered treated with a particular *dose* of treatment, where a dose is the binned value of the absolute value of the treatment variable.

Randomized Controlled Trials. In an ideal scenario, the best approach to estimate the causal effect of the neighbourhood treatment to the outcome is to conduct a randomised controlled trial (RCT). We randomly select half of the wards and strip them of the treatment variable as control, and apply the same amount of treatment to the rest. Why this is infeasible is obvious, but what is the next best alternative?

We run an *observational study*, which aims to achieve the same goal as a RCT, but without data from a randomised experiment. To do this we first look at the intuition behind why RCTs work in the first place. The key point is that randomisation of treatment assignment to each neighbourhood ensures that all of the variables that may also be influencing the outcome are equal in expectation as the number of neighbourhoods increases. This means that the outcomes of the two groups, those exposed to treatment and the control, are comparable because the only difference between the groups is the treatment status. An important point here is that we do not have to know all the variables that also affect the outcome; the randomisation ensures these are balanced even if unobserved.

The observational study conducted here attempts to achieve the same goal as the RCT - construct two groups where the outcome is comparable with the only difference being the treatment status. However, since we no longer have randomisation, two problems arise. Firstly, there could be variables that also affect the outcome, which are only balanced in the groups if the treatment assignment is randomised. Additionally, we also could have variables that also affect the treatment status of the units, as treatment is not randomly assigned. Which means even if we were able to construct two groups with balance on the variables that affected the outcome, the groups may be imbalanced with respect to the variables that determined treatment, and the effect will not be representative of the average effect on the population. The variables that potentially affects both the treatment, outcome, or each other are called *confounding variables* or *confounders*[1]. The game here is then to construct two groups such that on average, the confounding variables are balanced, but the treatment assignment is different.

Matching. To achieve this, we use a procedure called *matching*. For each control unit, if we were able to find another unit in the population with the same set of values for all confounding variables but instead has received a different treatment, then the difference between the outcomes of the pair can be calculated and averaged over all pairs in the population. This is called exact matching. However finding an exact match is not always possible, so instead we find, for each unit, a match that would minimise the overall difference between the average value for all confounders between the two groups. This is the basic idea behind the approach.

Propensity Score. There are two additional modifications. Firstly, since there could be many confounding variables, high-dimensional matching between the units could be potentially difficult, so we match instead on a single measure called the *propensity score*, which is a function of the confounding variables and is shown to achieve balance, on average, for the confounding variables [28]. Secondly, instead of a binary treatment status, treated or control, there are multiple possible treatment 'doses', where the dose is defined by the bin in which the treatment variable falls into. The two constructed groups now consists of the 'low dose' group and the 'high dose' group, because units can only be matched if they received a different dose of treatment. The balancing nature of the propensity score still applies in this case under some conditions.

At a Glance. In summary, we estimate the causal effects by emulating the results of a randomised controlled trial using a matching procedure based on propensity scores to balance the confounding variables between the high dose and low dose treatment group. The difference between the outcomes of these two groups is interpreted as the average causal effect of the treatment on the population.

[1]While in some literatures the word confounder is used to refer to a particular type of confounding variable - the common cause between treatment and outcome - here we use it interchangeably with confounding variable

4 METHODOLOGY

In this section we go into more detail on various parts of the methodology before we describe its application to a specific example.

As briefly mentioned previously, the aim is to attempt to emulate a randomised controlled experiment - which means that the confounding variables between the 'treated' and 'control' group in the binary case, or high and low dose group in the multiple treatment case, should be balanced on average.

To achieve this, the wards are matched in such a way that minimises the difference between the confounding variables over all matched pairs, and maximises the difference between the treatment 'dose' e.g the binned value of available sport venues in a ward, in each pair.

In this section we first describe matching in the case of a binary treatment, then introduce the need for the propensity score, followed by how this matching can be achieved with multiple treatment levels.

4.1 Matching with Binary Treatment

Finding a match for a treated unit with precisely the same values for all variables in the set of confounders is called exact matching. In practice, especially with more than a handful of confounders, this is infeasible.

Given a confounder x, if an exact match cannot be found, the nearest value to the confounder could be used instead. However, where there is more than one confounder, the matching procedure uses the sum of the difference between all of the confounders, and aims to find a matching that minimizes this total sum. This is called nearest neighbour matching. We find a set of pairs of units M, given a set of p confounders to minimise the distance

$$\sum_{(i,j)\in M} \sum_{p} |x_p^i - x_p^j|$$

where x_p^i denotes the pth confounder of unit i. This optimisation problem can be solved as a graph matching problem. Given a graph $G = (V, E)$ a matching is a subset $M \subset E$ such that for all $(u_1, v_1), (u_2, v_2) \in M$, $u_1 \neq v_1 \neq u_2 \neq v_2$ i.e no two edges share a common vertex.

A couple of definitions on graph matching. A *perfect matching* is a matching that includes all the vertices of a graph. This is only possible when the number of vertices are even. A near-perfect matching is a matching when only one vertex is excluded from the matching. This happens when the number of vertices are odd. A matching is called *maximum* if it contains the largest number

of edges, which in turn implies it contains the largest number of vertices. A perfect and near-perfect match is always a maximum matching. The problem of finding maximum matchings can be extended to weighted graphs, where each edge is associated with a distance. We can then ask for a maximum matching that minimises the total sum of all edge weights. Such a matching is called an *optimal matching*. Our problem can be framed as an optimal matching problem with a fully connected graph, where the vertices represent the wards, and the edge weights are the difference between the values of the confounders between each potentially matched pair.

4.2 The Propensity Score

Even with nearest neighbour matching, the high dimensionality of the confounders can still be a problem. With many confounders, the optimal match may still yield large overall distances in the matching, and the confounders in the treatment and control group could still remain imbalanced, if the sample size is small relative to the dimension of the confounders. The theory of the propensity score resolves this issue by stating that, instead of matching on all of the confounders, which may possibly be high dimensional, if the matching was performed based on a value called the propensity score, a one dimensional scalar value, then the confounders in the treatment and control group should still be balanced on average, in the limit of a large sample [28]. This was intitially shown to be true in the case of binary treatment. The estimation of the propensity score is discussed next in the context of multiple treatments.

4.3 Matching with Multiple Treatment Levels

If we let z denote the level of treatment received, the discussion so far has been constrained to the case where there are only two possible 'doses' of treatment, that is, no treatment at all $z = 0$, and a single value for treatment $z = 1$. For our particular application, we have to consider extensions of this to the case where there are many possible levels of treatment.

The propensity score is unknown, and has to be estimated. Additionally, the balancing nature of the propensity score is only proved in the case where there were two levels of treatment. Joffe et al [11] showed that the balancing property of the propensity score could be extended to the case with varying doses of treatment. This means that matching based on a scalar propensity score will still balance observed confounders when there are different doses of treatment. This is precisely the case when $P(z|x) = P(z|b(x))$, where x are the confounders and $b(x)$ is the propensity score. McCullaugh's ordered logistic model [20] would satisfy this criteria.

In light of this, the propensity score for any unit i corresponds to $\beta^T_{x_i}$ in the model

$$log\frac{P(z_i) \geq d}{P(z_i) < d)} = \theta_d + \beta^T \mathbf{x_i} \tag{1}$$

where d is the different doses of treatment, and can take values $= 2, 3, .., D$. Note that here, the distribution of treatment levels given the confounders depends only on $\beta^T_{\mathbf{x}}$, which means $P(z|x) = P(z|b(x))$, and makes it a balancing score in the case with multiple treatments. An estimate of this value can be obtained using maximum likelihood, and is denoted $\hat{\beta}^T_{\mathbf{x_i}}$.

Another difference when performing matching with doses is that we would like matched pairs to have different levels of treatment. Similar to the case with binary treatment when control units are only allowed to match with treated units and vice versa.

To maximise the difference between the treatment levels of the matched pairs, instead of matching directly with the difference of the estimated propensity score, a modified version of the distance metric that incorporates the treatment dosage, adopted from Lu et al [16] is used. The distance between two units i and j, denoted $d_{i,j}$ is then defined as

$$d_{(i,j)} = \frac{|\hat{\beta}^T_{\mathbf{x_i}} - \hat{\beta}^T_{\mathbf{x_j}}| + \epsilon}{|z_i - z_j|} \tag{2}$$

Where ϵ is an extremely small positive number. The role of ϵ is to deal with edge cases when either the potential match has the exact same confounders or treatment dosage.

In the case with two treatments, matching can be solved using an optimal bipartite matching algorithm, such as the Hungarian algorithm [13]. This is because the vertices of the graph can be divided into two disjoint sets, those which represent units who received treatment, and those which did not, as units from the same group are not allowed to match, and has no edge in the graph. Here where we have multiple doses of treatment, and each unit can potentially be matched with any other unit, and is called an optimal non-bipartite matching problem. The algorithm used to solve the this problem can be attributed to Edmonds [6]. This algorithm generalises his original blossoming algorithm that constructs maximum matchings, to construct a maximum weighted matching on a graph.

We can now construct a graph, with each vertex representing a unit, and edges connected to other units where the weight is the distance given by equation 2. Using Edmonds algorithm we then can obtain an optimal matching M. Given a set of matched pairs M, and let y denote the outcome and z the treatment, the average treatment effect (ATE) per dose can then be calculated by

$$ATE = \sum_{(i,j)\in M} \frac{y_i - y_j}{z_i - z_j} \tag{3}$$

5 PROOF OF CONCEPT APPLICATION

We demonstrate the feasibility of using the matching procedure described previously to investigate sporting facility availability and its effect on antidepressant prescriptions on the average neighbourhood in London. We first describe the datasets, then the confounding variables, apply the matching procedure and assess the results.

5.1 Datasets

To obtain the data required to perform the matching, we combine eight datasets. These include *venue*, *prescription*, *GP*, and *drug* data, *ward boundaries* and *demographics*, the *Greater London area boundary*, and finally *postcodes* in London. All of these datasets are open.

Venue Data. The OpenStreetMap (OSM) project is a crowd-sourcing platform that aims to create a freely available and editable map of the world. The map contains data on a wide array of geographical features, such as shops, cycle routes, benches to waterways, which

are contributed by users and moderated before being added to the project. These features can be given a tag, where a list of official tags are maintained [7].

Prescription Data. The UK National Health Service (NHS) maintains an open dataset that contains all of the prescriptions, broken down at the practice level, across the whole of England. This dataset specifies which drug, identified using the British National Formulary[2] Code (BNF Code), and the quantity prescribed in GP's across England.

GP and Drug Data. Each month a BNF Code to chemical name mapping is published which allows the drug prescribed to be identified. The NHS also maintains a medical practice dataset which allows a practice to be located using its postcode from the practice code.

Geographical Boundaries and Demographics. The London data store [1] is a repository for data on different geographical separations in London and contains statistics such as population, housing benefits, crime rates, access to nature, to higher education results. Additionally, the geographical separations at the ward, borough or LSOA level is also available in GeoJson format.

Finally, The office of national statistics (ONS) maintains a postcode dataset that allows mapping between a postcode and different administrative or electoral areas, or latitude and longitude.

5.2 Confounding Variables

The confounding variables considered can be broken down into three categories. First of these is the population structure. London, as with many developed cities, is a commercially active area and different parts of the city attract people at different stages of their lives. The data segregates the population into those with age 0-15 (children), 16-64 (working age) and 64+ (retired). Plotting these values as a percentage of the population of each ward on a choropleth clearly shows that central London attracts the majority of the working age group, retirees mostly occupy the outskirts and children are mostly even distributed, albeit with higher concentration in more deprived areas. This is shown in Figure 1.

Additionally, the number of full time and part time employees in an area helps us characterise whether the area is more residential or commercial, which affects both sporting facilities and prescription numbers.

The second is the availability of green spaces. Despite London being one of the greenest capital cities on the planet, there are major discrepancies in the availability of green spaces. The demographic dataset captures this with two values, % area that is green space, and % homes with deficient access to nature. Access to nature not only has been linked with mental health [3] but its availability also determines the type of sporting venue that is available.

Finally, with SES being so commonly linked with a variety of health outcomes [2, 22], we also include several measures of deprivation, ranging from housing benefit claims to job seekers allowance claims. The full set of confounders used are shown in Table 1.

Of course, there are many points of discussion surrounding how we select confounding variables and decide on the causal structure and we defer to Section 7.1 where we touch on the key points. The results of applying the matching procedure is considered next.

Table 1: The set of considered confounders.

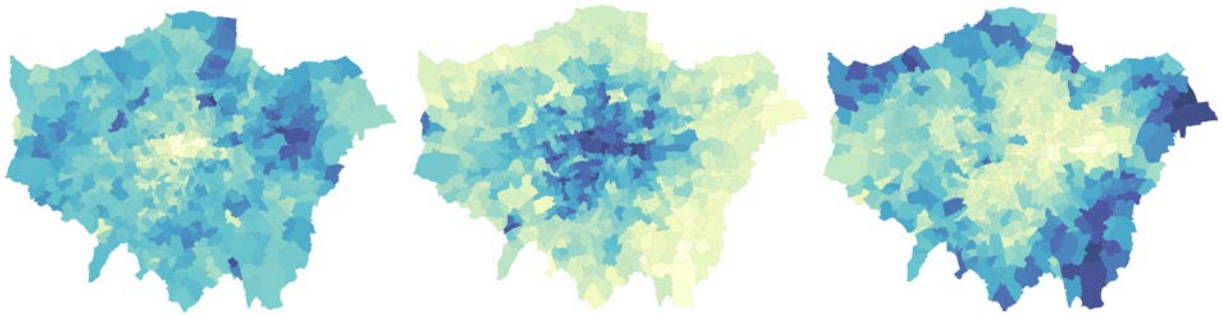
Population Structure
Ages 0-15
Ages 16-64
Ages 64+
Full-time Employees
Part-time Employees
Green Spaces
% Area Green Space
% Homes with Deficient Access to Nature
Deprivation
DWP [3] Working-age client group (Rates)
Employment and Support Allowance Claimants
Housing Benefit Rates
Income Support Claimants
Incapacity Benefit Claimants
Job Seeker's Allowance Claimant Rates

5.3 The Treatment Effect

In this section we apply the matching procedure as described in Section 4 using the datasets and confounders identified above, to understand the impact of sport venues over antidepressant prescription. Doing this requires the following steps:

(1) Using the OSM data dump (the 'planet file'), the open source Osmium Tool, a boundary file of the Greater London Area and boundaries for London wards, we extracted all venues related to sports between the years 2011-2013 and mapped them to each ward in London using a geometry library. The venues were identified as sports-related through a list of key:value tags, the ones which are included were distilled from the official tag page [23] and are shown in Table 2. The extracted venues and its frequency are represented using the word-cloud shown in figure 2.
(2) To determine prescriptions, the prescription data was filtered by BNF code for antidepressants, the postcode data was filtered to only those in London, the GP data is then used to filter for London GPs and obtain a list of GP IDs with its corresponding prescription numbers for 2011-2013. The GPs were then mapped to each ward similarly.
(3) For each unit (ward) we now have the treatment (sport venues), outcome (antidepressant prescriptions) and associated confounding variables from the demographic dataset. We can now estimate the propensity score, $\beta_{x_i}^T$ in equation 1, calculate the distance using equation 2 and construct a graph to perform optimal non-bipartite matching with Edmonds algorithm.
(4) Given a matching M, we could calculate the average treatment effect as in equation 3. This is shown along with the distribution of the effects in the matched pairs in Table 3.

[2] A pharmaceutical reference book published by the British Medical Association
[3] Department of Work and Pensions



(a) Ages 0-15 (b) Ages 16-64 (c) Ages 65+

Figure 1: A choropleth map revealing systematic bias in age structures in London. Darker colours indicate a higher density.

Table 2: Objects tagged with the key and values listed here were included in the extraction.

Key	Values Included
Sports	All Values
Leisure	Pitch Golf Course Swimming Pool Sports Centre Horse Riding Track
Club	Sports



Figure 2: Tag values from extracted sports venues, size is proportional to frequency

The units here are the change in antidepressant prescription per person, per dosage of sports venue in a ward in London. We can see that the ATE across all three years are negative. This suggests that there is a negative effect. If we compare this with the treatment effect distribution from matching on the null model - where the sport venue counts per wards were randomised - also shown in Figure 3, this also suggests that there is an effect in the negative direction.

Table 3: The distribution of the treatment effects in the matched pairs across 3 years, the average distribution on the null model, across ten runs, is shown below.

Year	ATE	Min	25th %	75th %	Max
2011	-0.461	-91.579	-7.651	5.774	65.911
2012	-1.117	-66.628	-7.948	7.912	55.853
2013	-0.529	-90.034	-7.573	6.454	72.320
Null Model					
2011	0.178	-62.699	-7.147	6.971	62.811
2012	0.494	-68.740	-7.245	8.495	62.291
2013	-0.085	-64.700	-8.180	8.161	71.384

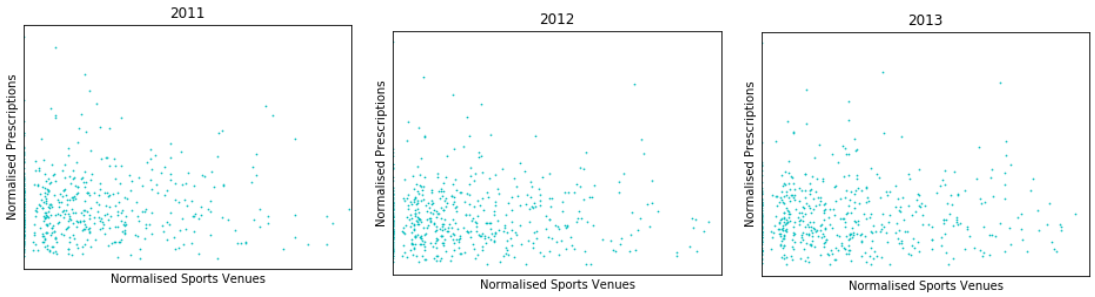
Specifically, a unit increase in dosage level is expected, in the long run, to decrease the number of antidepressant prescriptions per person by anywhere between 0.461 to 1.117. With an average prescription level per person value of 23.51 across all wards across three years, this amounts to a decrease of 1.96-4.75% per year. If we looked at the simple correlation between the two variables, we could not discern as much, as shown in Figure 3.

However this result deserves a deeper discussion. With such a relatively small magnitude, a more conservative interpretation would be that of a small negative, or no effect. This is due to the many possible uncertainties, both in the data and assumptions, that will be discussed in Section 7.2. If we look at the entire distribution of effects we can see that it spans a wide range of values, with the distribution skewed towards the negative direction, albeit slightly. This is shown in Figure 4.

In the next section we will explore some more aspects related to these results and assess their validity.

6 ASSESSING THE INFERRED EFFECT

How do we assess the reliability of the inferred effects? Since we do not have the 'true' effects to compare with, the effectiveness of the matching procedure in this instance is evaluated using three metrics: the dose difference, the confounder balance, and performance on a synthetic dataset.



(a) Spearman 0.0286, p-value 0.493 (b) Spearman 0.00765, p-value 0.855 (c) Spearman 0.0155, p-value 0.712

Figure 3: Simple correlation between the normalised number of sports venues and prescription across wards in London. This paints a very different picture to the obtained effects after matching.

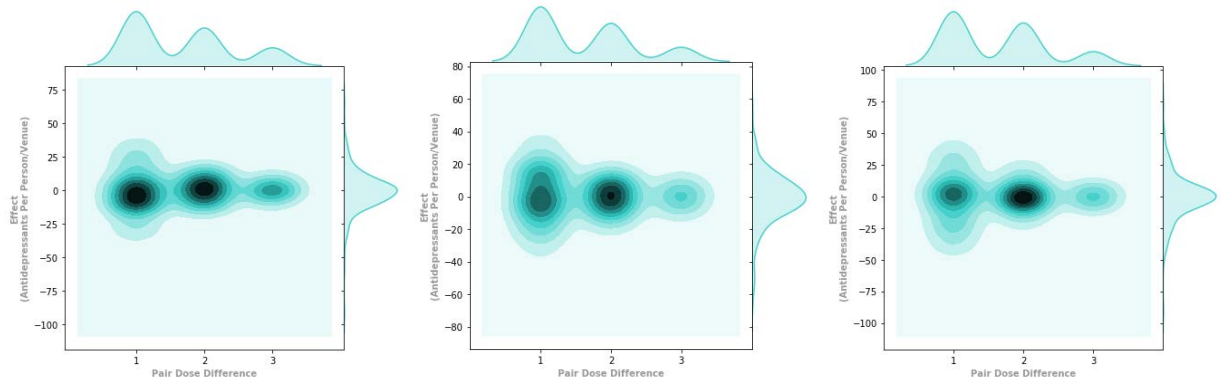


Figure 4: Plot of the joint distribution between dose difference and effect of each matched ward pair. From left to right: 2011, 2012, 2013.

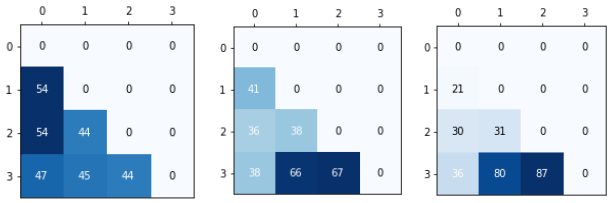
Comparing the results to alternative methods used to elicit causal effects such as difference-in-difference or regression discontinuity would also be informative, however the available data does not satisfy the conditions for their use.

6.1 Dose Difference

One of the intentions behind the form of the distance metric used, given by Equation 2, is to maximise the difference in treatment doses of each matched pair. This leads to reduced noise when computing the in-pair treatment effect.

Figure 5 shows the distribution of doses in each matched pair across the span of three years. Rows represent the dose of the unit with the higher level of treatment in the pair, and the columns represent the dose of the unit with the lower treatment. Darker colours indicate higher number of pairs with these doses.

Here one of the aims is to create pairs where the doses are not equal, we can see that this is achieved judging from the diagonals of the heatmap across three years. In fact, the difference in dosage is larger than 1 level in 50.7%, 49.0% and 51.2%, respectively. With approximately 35% of pairs having a difference in dose level of 2 and 15% with a difference of 3 dose levels across three years. The matching procedure has indeed created pairs with different treatment levels.



(a) 2011 (b) 2012 (c) 2013

Figure 5: Distribution of within-matched-pair doses between the high dose unit and low dose unit. Rows represent the high dose unit and column the low dose unit.

6.2 Confounder Balance

The original aim in any matching procedure is to produce different groups of units that have, on average, similar values for the confounders, such that the treatment effect is representative of the population. Table 4 shows the average value of the confounders between the high dose unit and low dose unit in a pair.

Here we can see that most of the confounders have successfully been balanced across the two groups. The exception is perhaps the normalised full time and part time employee rate, which has a difference of over 5 and 1.4 percent respectively. This is seen clearly

Table 4: The average normalised values of the confounders in the high dose and low dose groups across all matched pairs in 2011. Other years gave similar results.

Confounder	High	Low
Population Aged 0-15	19.611	19.760
Population Aged 16-64	68.991	69.182
Population Aged 65+	11.398	11.058
Greenspace Area	27.272	26.448
Homes with Deficient Access to Nature	25.751	25.727
Employment & Support Allowance Claims	1.034	1.026
Housing Benefit Claims	12.871	12.863
Income Support Claims	3.393	3.374
Incapacity Benefit Claims	2.882	2.807
DWP Working Age Clients	14.363	14.307
Part-Time Employees	13.229	14.688
Full-Time Employees	34.089	39.827
Job Seekers Allowance Claims	5.623	5.569

by plotting the difference between the two groups as a percentage of the average of the original values, shown in blue in Figure 6.

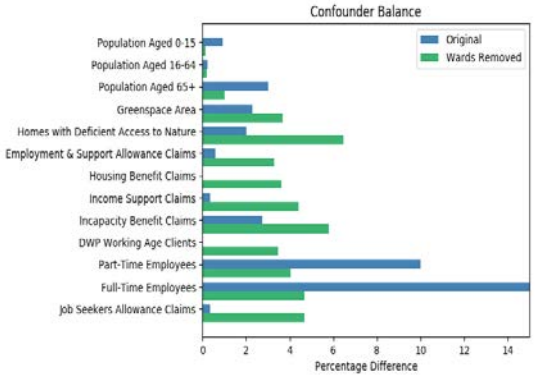


Figure 6: The percentage difference between the confounders of the high and low dose groups with respect to the average of the original values. The green bars show the balance after the wards of St James, West End, and the City of London was excluded from the matching procedure.

On further investigation this can be attributed to a couple of extreme values for these two confounders in the ward data, which can be seen clearly in Figure 7, with particular wards having over three times the value of the remaining wards.

The three wards with the highest values for both confounders were predictably large commercial areas. These were the St James and West End wards of the Westminster area, containing Mayfair, Soho, large business districts, and the City of London. We may be tempted to remove these wards in the matching procedure, in an attempt to achieve more balance within the confounders. At first glance, doing so may seem a good idea, shown in green in Figure 6.

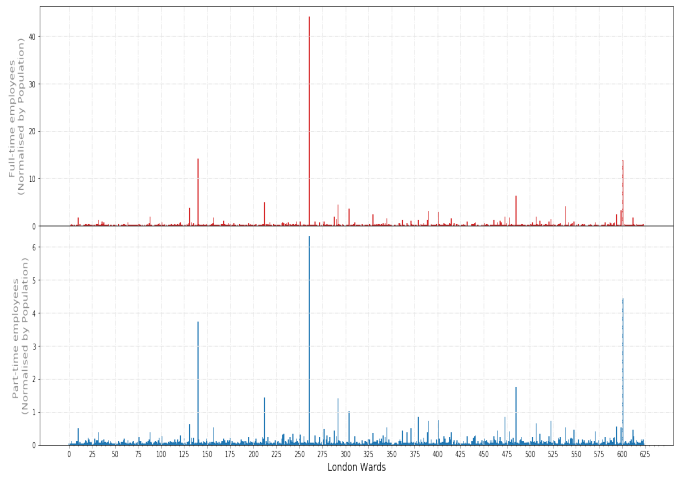


Figure 7: A plot of full time and part time employees, normalised by population, for the wards in London. Here we clearly see at least three outliers.

However, removing the wards, all of which are highly commercial areas of the city, induces a bias, as evidenced by the increased imbalance across the other confounders. The reason behind this may be that removing key business districts means that other business areas are forced to match with less similar areas, perhaps residential ones, generating additional imbalance. Additionally, remove key business districts mean that the average effect is now no longer representative of London as a whole, but rather London without the wards of St James, West End, and the City of London. Therefore for the entirety of London, we have to take into account the way that the balancing errors may have influenced the obtained effect in interpreting said effect.

6.3 Generating Synthetic Data

The most common benchmark in the literature for the evaluation of the causal effect estimate is the use of synthetic data [32][14][9][33]. Here, instead of constructing a completely arbitrary artificial dataset, we model the observed dataset and use this model to generate synthetic data. This means that we can generate datasets that mimic the real data, but where we are able to specify the underlying mechanisms - how the confounders affect both the treatment and the outcomes, and how the treatment affects the outcome. As a result, we know the 'true' causal effect of the treatment. We can then assess how well the method recovers this true effect.

More precisely, if we wanted to generate N units $i = 1, ..., N$ (these are artificial 'wards' in our case) each with p confounders (13 here) where we denote the confounders of unit i as $\mathbf{x_i} = x_{i1}, ..., x_{ip}$, we first have to generate values for all confounders for each N units. We then specify two functions, $f_Z(\mathbf{x_i})$ that takes in the confounders and generates a treatment (normalised sporting facilities) for a unit, and $f_Y(z_i, \mathbf{x_i})$ that take in the confounders and the treatment to generate an outcome (normalised antidepressant prescriptions) for that unit.

The distribution of the confounders from the real data is show in Figure 8. We model the confounders using a truncated normal

distribution except for part time employees, full time employees, and deficient access to nature, which we model using an exponential distribution. The parameters of the confounder distributions was estimated using maximum likelihood.

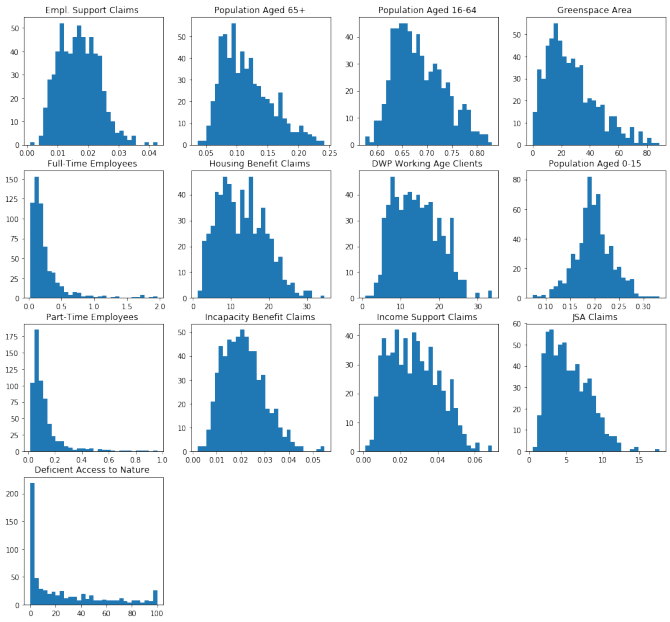


Figure 8: Distribution of normalised confounders in London wards.

Similarly looking at the distribution of sporting facilities, we then constructed $f_Z(\mathbf{x})$ as an exponential distribution:

$$f_Z(\mathbf{x}) = P(Z = z|\mathbf{x_i}) = \begin{cases} \frac{1}{f(\mathbf{x_i})} e^{\frac{z}{f(\mathbf{x_i})}} & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

where

$$f(\mathbf{x_i}) = \frac{\sum_p x_{ip}}{\alpha}$$

and α is tuned such that the treatments mimic the observed distribution.

The outcome for each unit, Y_i, is determined by $f_Y(z_i, \mathbf{x_i})$ and is constructed as a truncated normal distribution where if $Y > 0$

$$Y \sim \text{Normal}(\alpha \cdot \sum_p x_{ip}, \beta \cdot \sum_p x_{ip}) + \gamma z_i$$

where the values for α and β were tuned such that the data generated is within the observed range of outcomes in the real data, and z_i is the treatment for unit i. Here, we know that the treatment effect is γ per dose by construction.

Figure 9 shows a comparison between the observed distribution of the treatment and outcomes, and the distribution generated by the mechanism just described.

It is important to acknowledge that the functions chosen to generate the data is one of many possible functions that could potentially be the underlying mechanism behind this process. It is virtually impossible to tell which function is more likely than the other, besides the fact that it is able to generate data that closely mirrors the observed data.

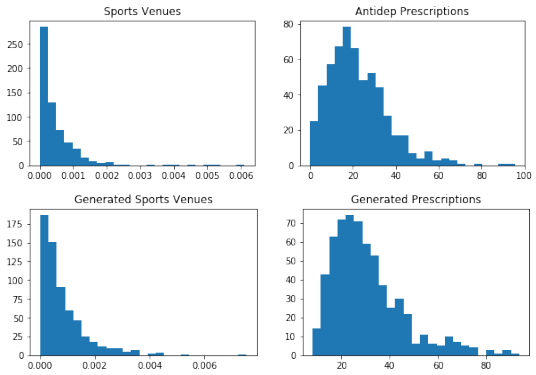


Figure 9: Comparison between normalised sports venues and antidepressant prescriptions in the data and the generated dataset.

6.4 Matching on Synthetic Data

Now that we could generate units along with its confounders, treatment, and outcome, we can run the same matching procedure used on the real dataset on the synthetic dataset. Since we know the true ATE of the synthetic dataset by construction, we can evaluate how the matching procedure performs.

The synthetic process was run to generate a total of 6250 units, each batch of 625 having a different set of confounders. This is done four times for four different 'true' ATE values, two for each direction, one with a small effect and one large effect. The results of the treatment effects calculated from the matchings are shown in Table 5.

Table 5: Treatment effects for the synthetic dataset. Each row was obtained from 6250 generated units, with each group of 625 containing a different set of confounders.

ATE	Est. ATE	Min	25th %	75th %	Max
-1	-0.292	-75.2	-7.20	6.17	63.2
-10	-9.30	-73.7	-15.9	-2.75	87.5
1	1.45	-70.5	-5.30	8.55	70.6
10	10.4	-69.9	3.48	17.3	68.5

Here we can see a similar outcome to the treatment effects obtained from the real dataset. The effects span a wide range, however the distribution is skewed towards the direction of the true effect. In the case where the magnitude of the 'true' effect is large, over 75% of all estimated effects lie in the correct direction.

The estimated effects also lie in the correct direction. We can see that in larger true effects, the matching procedure is able to give a better estimate. Where the effect is small, the inferred effects are less accurate. This gives some indication to how we should interpret the results obtained from real data, and since the obtained effects are small, we should err more on the side of a conservative interpretation.

7 DISCUSSION

We now turn our attention to discuss the potential sources of error in this approach. First we address the issue of confounding variable selection which was deferred from section 5.2 and then address the sources of error more generally.

7.1 Confounding Variable Selection

In the current causal inference literature there does not exist a method of testing which variables are relevant to a causal model for a particular problem from a set of possible variables. The closest that we have come across in this respect are methods in causal discovery [26], which at the current best can be used to find the equivalence class of directed acyclic graphs that could represent the class of models that potentially produced the observed distribution of variables, under some restricting assumptions, and given a pre-specified set of variables.

Therefore the current best option in determining confounding variables related to a causal question is to use existing subject matter knowledge, or literature relevant to the problem at hand. This is a fundamental problem of causal inference, where we cannot be certain whether we have identified all variables relevant to the cause and effect relationship at hand. It may be the case that in the future it is discovered, hypothetically, that people who smoke are severely more likely to develop depression, in which case the proportion of smokers in an area, or some variable that closes correlates with it, should additionally be considered a confounder.

An additional point to note is that of selecting key variables and edges that would have enough of an effect to be worth considering. Precisely how large is 'enough' still remains an open issue. People who have pets may indeed be slightly less prone to develop severe depression, but is the effect noticeable enough? Does the distribution of people who own pets consistently different in various areas? Systematic ways to answer these types of questions still remain unknown.

7.2 Sources of Error and Open Problems

The challenge with inferring cause is the amount of care that must be taken to eliminate all sources of errors throughout the entire process that could impact the results. In this project we by no means claim that this has been achieved, and discuss the potential sources of errors that we have conceived of here. The errors can be classified into three major categories; those that emerge from the assumptions about the causal structure, from the data itself, and from simplifying assumptions.

The most glaring source is that of the causal structure. Here the inferred effects is only a good estimate if we have included all of the confounders that are indeed relevant to the problem. This means that there are no unobserved factors that could influence the treatment and outcome. This is otherwise known as the strong ignorability assumption in Rubin's causal model, or called identifiability in Pearl's framework [25], and we have discussed above.

The second source of error are those from the data. Does the dataset of venues include all existing sports venues in London? If not, are they missing at random, or does it depends on some factor, such as the area? If it is the latter case, how do we assess this bias in exclusion from the dataset, and how do we incorporate this into an uncertainty of the estimated effect? The same issue applies with the prescription data. A potential direction forward is to assume the observed data is some fraction of reality and find the thresholds, under some additional assumptions, where the results significantly deviate from those obtained here.

Additionally, can the numbers in the dataset be taken at face value? For instance, the statistics on the demographics of a ward may have itself been generated by a statistical model. Even if we had the uncertainty attached to any particular value, how do we incorporate this uncertainty when inferring the effect?

The final major source of error are the simplifying assumptions. For instance, does the number of antidepressant prescriptions represent the rate of depression in an area well? Or does the exclusion of private prescriptions in the data create enough of a bias? The same question about incorporating uncertainties, if these were known, applies here. In general, issue of propagating uncertainties through some causal inference procedure is an interesting and open issue.

8 CONCLUSIONS

In this paper we have illustrated an approach to investigate the causal effects of the built neighbourhood environment on population health using open data at scale. This is important for several reasons. Firstly, it provides the potential to investigate the physical underlying mechanism of how a particular aspect of the neighbourhood influences health. Many higher level neighbourhood characteristics such as income or SES have been associated with multiple health outcomes, but how exactly does this affect health remains a question. Is it because people with higher SES eat healthily, exercise more often, or smoke less? Looking at the available services in a neighbourhood allows us to tackle these types of potential mechanisms. Secondly, the physical environment has the advantage of being better defined and quantifiable as opposed to more abstract measures such as general deprivation. Additionally, it provides a low cost and fast alternative to traditional methods of data collection, often with extra granularity. Lastly, these features of a neighbourhood are controllable from a policy point of view, as opposed to individual lifestyle choices, and is perhaps a more effective choice from a public health intervention standpoint.

Specifically, we have investigated the extent to which the availability of sporting venues have a causal role to play on the prescription of antidepressants in the neighbourhoods of London. Is the strength of evidence obtained and the magnitude of the indicated effect large enough to warrant a mass campaign encouraging sporting facilities? Probably not. However, it does give us insight into some important issues. With the preliminary knowledge of the possible effect size being rather modest, decision makers can now decide whether pursuing this question further, to obtain a better estimate of the effect, for example by collecting additional data or investigating to what extent the assumptions hold, is worth pursuing. It may be the case that the resources available should be put into other issues that are now known to be more cost effective, given the evidence obtained here. Or similarly, this method can be adopted to investigate related questions on venues and health.

It is hoped that this contributes to the conversation about urbanisation, and more generally, data-driven approaches in research tackling social and policy questions.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Damon Wischik for helpful discussions, and also thank the anonymous reviewers for their valuable and helpful comments.

A. Hasthanasombat's work is supported through the Cambridge Trust and King's College Cambridge, and he would like to also thank the Department of Computer Science and Technology for their additional support.

REFERENCES

[1] 2018. London Datastore. https://data.london.gov.uk/dataset/lsoa-atlas

[2] Mariana C. Arcaya, Reginald D. Tucker-Seeley, Rockli Kim, Alina Schnake-Mahl, Marvin So, and S. V. Subramanian. 2016. Research on neighborhood effects on health in the United States: A systematic review of study characteristics. *Social Science & Medicine* 168 (Nov. 2016), 16–29. https://doi.org/10.1016/j.socscimed.2016.08.047

[3] Jo Barton, Rachel Bragg, Carly Wood, and Jules Pretty. 2016. *Green exercise: Linking nature, health and well-being*. Routledge.

[4] Alexandra Blair, Nancy A. Ross, Geneviève Gariepy, and Norbert Schmitz. 2014. How do neighborhoods affect depression outcomes? A realist review and a call for the examination of causal pathways. *Social Psychiatry and Psychiatric Epidemiology* 49, 6 (June 2014), 873–887. https://doi.org/10.1007/s00127-013-0810-z

[5] Emile Durkheim. 2005. *Suicide: A study in sociology*. Routledge.

[6] Jack Edmonds. 1965. Maximum matching and a polyhedron with 0, 1-vertices. *Journal of Research of the National Bureau of Standards B* 69 (1965), 125–130.

[7] Open Street Map Foundation. 2018. Map Features - OpenStreetMap Wiki. https://wiki.openstreetmap.org/wiki/Map_Features#Route

[8] Panagis Galiatsatos, Cynthia Kineza, Seungyoun Hwang, Juliana Pietri, Emily Brigham, Nirupama Putcha, Cynthia S. Rand, Meredith McCormack, and Nadia N. Hansel. 2018. Neighbourhood characteristics and health outcomes: evaluating the association between socioeconomic status, tobacco store density and health outcomes in Baltimore City. *Tobacco Control* 27, e1 (July 2018), e19–e24. https://doi.org/10.1136/tobaccocontrol-2017-053945

[9] Xing Sam Gu and Paul R. Rosenbaum. 1993. Comparison of Multivariate Matching Methods: Structures, Distances, and Algorithms. *Journal of Computational and Graphical Statistics* 2, 4 (1993), 405–420. https://doi.org/10.2307/1390693

[10] Desislava Hristova, Matthew J. Williams, Mirco Musolesi, Pietro Panzarasa, and Cecilia Mascolo. 2016. Measuring Urban Social Diversity Using Interconnected Geo-Social Networks. In *Proceedings of the 25th International Conference on World Wide Web - WWW '16*. ACM Press, Montréal, Québec, Canada, 21–30. https://doi.org/10.1145/2872427.2883065

[11] Marshall M. Joffe and Paul R. Rosenbaum. 1999. Invited Commentary: Propensity Scores. *American Journal of Epidemiology* 150, 4 (Aug. 1999), 327–333. https://doi.org/10.1093/oxfordjournals.aje.a010011

[12] Markus Jokela. 2014. Are Neighborhood Health Associations Causal? A 10-Year Prospective Cohort Study With Repeated Measurements. *American Journal of Epidemiology* 180, 8 (Oct. 2014), 776–784. https://doi.org/10.1093/aje/kwu233

[13] Harold W Kuhn. 1956. Variants of the Hungarian method for assignment problems. *Naval Research Logistics Quarterly* 3, 4 (1956), 253–258.

[14] Sheng Li, Nikos Vlassis, Jaya Kawale, and Yun Fu. 2016. Matching via Dimensionality Reduction for Estimation of Treatment Effects in Digital Marketing Campaigns.. In *IJCAI*. 3768–3774.

[15] Bruce G. Link and Jo Phelan. 1995. Social Conditions As Fundamental Causes of Disease. *Journal of Health and Social Behavior* (1995), 80–94. https://doi.org/10.2307/2626958

[16] Bo Lu, Elaine Zanutto, Robert Hornik, and Paul R. Rosenbaum. 2001. Matching with doses in an observational study of a media campaign against drug abuse. *J. Amer. Statist. Assoc.* 96, 456 (2001), 1245–1253.

[17] Jens Ludwig, Greg J. Duncan, Lisa A. Gennetian, Lawrence F. Katz, Ronald C. Kessler, Jeffrey R. Kling, and Lisa Sanbonmatsu. 2012. Neighborhood Effects on the Long-Term Well-Being of Low-Income Adults. *Science* 337, 6101 (Sept. 2012), 1505–1510. https://doi.org/10.1126/science.1224648

[18] C. Mair, A. V. Diez Roux, and S. Galea. 2008. Are neighbourhood characteristics associated with depressive symptoms? A review of evidence. *Journal of Epidemiology & Community Health* 62, 11 (Nov. 2008), 940–946. https://doi.org/10.1136/jech.2007.066605

[19] Kate E. Mason, Neil Pearce, and Steven Cummins. 2018. Associations between fast food and physical activity environments and adiposity in mid-life: cross-sectional, observational evidence from UK Biobank. *The Lancet Public Health* 3, 1 (Jan. 2018), e24–e33. https://doi.org/10.1016/S2468-2667(17)30212-8

[20] Peter McCullagh. 1980. Regression Models for Ordinal Data. *Journal of the Royal Statistical Society. Series B (Methodological)* 42, 2 (1980), 109–142. http://www.jstor.org/stable/2984952

[21] J. Michael Oakes. 2004. The (mis)estimation of neighborhood effects: causal inference for a practicable social epidemiology. *Social Science & Medicine* 58, 10 (May 2004), 1929–1952. https://doi.org/10.1016/j.socscimed.2003.08.004

[22] J. Michael Oakes, Kate E. Andrade, Ifrah M. Biyoow, and Logan T. Cowan. 2015. Twenty Years of Neighborhood Effect Research: An Assessment. *Current Epidemiology Reports* 2, 1 (March 2015), 80–87. https://doi.org/10.1007/s40471-015-0035-7

[23] OSM. 2018. Key:sport - OpenStreetMap Wiki. https://wiki.openstreetmap.org/wiki/Key:sport

[24] Michael J Paul and Mark Dredze. 2011. You Are What You Tweet: Analyzing Twitter for Public Health. (2011), 8.

[25] Judea Pearl. 2009. *Causality*. Cambridge University Press. Google-Books-ID: f4nuexsNVZIC.

[26] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. 2017. *Elements of Causal Inference: Foundations and Learning Algorithms*. MIT Press. Google-Books-ID: uie9AQAACAAJ.

[27] Jo C. Phelan, Bruce G. Link, Ana Diez-Roux, Ichiro Kawachi, and Bruce Levin. 2004. "Fundamental causes" of social inequalities in mortality: a test of the theory. *Journal of Health and Social Behavior* 45, 3 (Sept. 2004), 265–285. https://doi.org/10.1177/002214650404500303

[28] Paul R. Rosenbaum and Donald B. Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70, 1 (April 1983), 41–55. https://doi.org/10.1093/biomet/70.1.41

[29] Andrew G. Rundle, Michael D. M. Bader, Catherine A. Richards, Kathryn M. Neckerman, and Julien O. Teitler. 2011. Using Google Street View to Audit Neighborhood Environments. *American Journal of Preventive Medicine* 40, 1 (Jan. 2011), 94–100. https://doi.org/10.1016/j.amepre.2010.09.034

[30] Chris Smith-Clarke and Licia Capra. 2016. Beyond the Baseline: Establishing the Value in Mobile Phone Based Poverty Estimates. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 425–434. https://doi.org/10.1145/2872427.2883076

[31] United Nations Social and Economic Affairs Division. 2018. World Urbanization Prospects.

[32] Wei Sun, Pengyuan Wang, Dawei Yin, Jian Yang, and Yi Chang. 2015. Causal Inference via Sparse Additive Models with Application to Online Advertising. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, Austin, Texas, 297–303. http://dl.acm.org/citation.cfm?id=2887007.2887049

[33] Fani Tsapeli, Mirco Musolesi, and Peter Tino. 2017. Non-parametric causality detection: An application to social media and financial data. *Physica A: Statistical Mechanics and its Applications* 483 (Oct. 2017), 139–155. https://doi.org/10.1016/j.physa.2017.04.101

[34] May C. Wang, Soowon Kim, Alma A. Gonzalez, Kara E. MacLeod, and Marilyn A. Winkleby. 2007. Socioeconomic and food-related physical characteristics of the neighbourhood environment are associated with body mass index. *Journal of Epidemiology and Community Health* 61, 6 (June 2007), 491–498. https://doi.org/10.1136/jech.2006.051680

Distributed Tensor Decomposition for Large Scale Health Analytics

Huan He
Emory University
Atlanta, Georgia
huan.he@emory.edu

Jette Henderson
CognitiveScale
Austin, Texas
jette.henderson@gmail.com

Joyce C. Ho
Emory University
Atlanta, Georgia
joyce.c.ho.emory.edu

ABSTRACT

In the past few decades, there has been rapid growth in quantity and variety of healthcare data. These large sets of data are usually high dimensional (*e.g.* patients, their diagnoses, and medications to treat their diagnoses) and cannot be adequately represented as matrices. Thus, many existing algorithms can not analyze them. To accommodate these high dimensional data, tensor factorization, which can be viewed as a higher-order extension of methods like PCA, has attracted much attention and emerged as a promising solution. However, tensor factorization is a computationally expensive task, and existing methods developed to factor large tensors are not flexible enough for real-world situations.

To address this scaling problem more efficiently, we introduce `SGranite`, a distributed, scalable, and sparse tensor factorization method fit through stochastic gradient descent. `SGranite` offers three contributions: (1) *Scalability*: it employs a block partitioning and parallel processing design and thus scales to large tensors, (2) *Accuracy*: we show that our method can achieve results faster without sacrificing the quality of the tensor decomposition, and (3) *FlexibleConstraints*: we show our approach can encompass various kinds of constraints including l_2 norm, l_1 norm, and logistic regularization. We demonstrate `SGranite`'s capabilities in two real-world use cases. In the first, we use Google searches for flu-like symptoms to characterize and predict influenza patterns. In the second, we use `SGranite` to extract clinically interesting sets (i.e., phenotypes) of patients from electronic health records. Through these case studies, we show `SGranite` has the potential to be used to rapidly characterize, predict, and manage a large multimodal datasets, thereby promising a novel, data-driven solution that can benefit very large segments of the population.

CCS CONCEPTS

• **Information systems → Web mining**; **Data extraction and integration**; • **Computing methodologies → Factor analysis**; **Canonical correlation analysis**; **MapReduce algorithms**; • **Applied computing → Health informatics**.

KEYWORDS

Web Mining; User-Generated Content; Health Analytics; Tensor Decomposition; Distributed Algorithm; Apache Spark

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313548

ACM Reference Format:
Huan He, Jette Henderson, and Joyce C. Ho. 2019. Distributed Tensor Decomposition for Large Scale Health Analytics. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313548

1 INTRODUCTION

Increasingly large amounts of health-related data are released on the Internet and have great potential for enabling better disease surveillance and disease management. As a motivating example, search activities on diseases such as influenza can be used and correlated with actual influenza surveillance data. Estimation of influenza-like illness (ILI) rates is a well-studied task [25, 32], Google Flu Trends, while flawed, demonstrated a link between influenza related search queries and the Centers for Disease Control and Prevention's (CDC) ILI rates[19]. Similarly, programs such as the National Institute of Health's All for Us NIH [2], are looking to gather data and make it publicly available to researchers to enable precision medicine. Extracting influenza patterns or clinical characteristics from such high-dimensional data can pose challenges, even before considering whether the data has been appropriately labeled.

A vast majority of the algorithms for disease surveillance or disease prediction adopt a supervised learning approach, but the need for labels can limit the possible scope of the task. However, unsupervised learning methods such as tensor factorization have been successfully applied in many application domains including social network analysis [29, 30, 39] and health analytics [15, 16, 18, 22, 36]. Tensors can succinctly represent high-dimensional data, including various representations of time or different sources of data. For example, an existing work showed that factorizing a tensor that grouped ILI historical statistics by year, week, and region could tease out patterns that are commonly based on the weeks that influenza is highest, deliver insight into the degree to which regions are similar or different from one another in terms of influenza, and capture the changes in ILI intensity from one year to the next [13]. Moreover, a variety of constraints can be placed on the learned latent factors to extract meaningful patterns and reduce overfitting. Yet, efficient tensor decomposition of large datasets in the presence of such constraints can be challenging.

In this paper, we propose `SGranite`, a distributed tensor decomposition framework that can incorporate a variety of regularization terms to constrain the latent factors. In particular, we show that integrating three forms of regularization terms can achieve easier-to-interpret factors, provide robustness in the presence of noise, and map to existing domain knowledge. Moreover, `SGranite` is very fast and scalable. Using a Spark-based implementation, we demonstrate the ability to decrease computation time by distributing both

the data as well as the parameters without sacrificing accuracy. To promote reproducibility, our code is open-sourced and available on GitHub[1].

The contributions of our work can be summarized as follows:

- **Flexibility:** Our framework supports a variety of meaningful constraints such as sparsity, diversity, and distinguishability.
- **Scalability:** Our scalability analysis of `SGranite` on a large tensor constructed from healthcare data achieves near linearity speed-up as we scale to the number of machines. Moreover, our framework achieves at least a 4× speed-up compared to an existing state-of-the-art distributed tensor factorization method.
- **Accuracy:** Our empirical results in two health-related case studies show that incorporating the variety of constraints improves interpretability and robustness compared to the standard decomposition models.

Table 1 summarizes the contributions in the context of existing works.

2 BACKGROUND AND NOTATION

This section briefly introduces tensors and a popular tensor decomposition model. We refer the reader to [24, 34] for comprehensive overviews of practical tensor decompositions.

2.1 Tensors

Tensors are generalizations of matrices and vectors to higher dimensions. An N-way tensor is denoted as $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and each cell of the tensor represents the interactions between N types of data. Each dimension of the tensor is referred to as a *mode*. Tensors can be unfolded or flattened as a matrix, which is called *matricization*. $\mathbf{X}_{(n)}$ denotes the matricization of tensor $\mathcal{X}$ along mode-n. Table 2 lists the operations and symbols used in this paper.

Definition 2.1. A rank-one N-way tensor is the outer product of N vector

$$\mathcal{X} = a^{(1)} \circ a^{(2)} \circ \cdots \circ a^{(N)}.$$

Each element of a rank-one tensor is the product of the corresponding vector elements (i.e., $x_{i_1 i_2 i_3 \cdots i_n} = a^{(1)}_{i_1} \circ a^{(2)}_{i_2} \circ \cdots \circ a^{(N)}_{i_N}$).

2.2 Tensor Decompositions

The CANDECOMP / PARAFAC (CP) model [7, 14] is one of the most popular and well-studied tensor decomposition method. In CP decomposition, the tensor is factored into a sum of rank-one tensors:

$$\mathcal{X} \approx \mathcal{M} = \sum_{r=1}^{R} \mathbf{A}^{(1)}(:, r) \circ \mathbf{A}^{(2)}(:, r) \circ \cdots \circ \mathbf{A}^{(n)}(:, r),$$

where $\mathbf{A}^{(n)}(:, r)$ is the rth column of $\mathbf{A}^{(n)}$. CP-decomposition can be also expressed as $[[\lambda; \mathbf{A}^{(1)}; \mathbf{A}^{(2)}; \cdots; \mathbf{A}^{(n)}]]$, where λ is a vector of the weights λ_r. Several benefits of the CP decomposition includes its intuitive output structure, uniqueness property that makes the model reliable to interpret, and the ability to learn a model even with relatively small amount of observations [24]. Figure 1 provides an example of the CP decomposition for an influenza-based tensor, where each rank-one tensor represents a pattern over time for a group of states and a set of search queries.

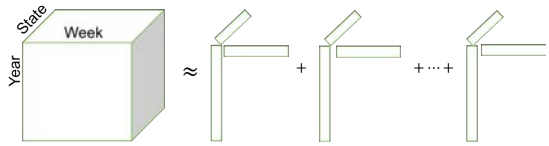


Figure 1: An example of CP decomposition for influenza search data. A tensor is constructed of time series data is decomposed into the weighted sum of rank-one tensors based on the minimization of an objective function. Each rank-one tensor, formed by taking the outer product of factor vectors, constitutes a latent factor.

The CP decomposition, $\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \cdots, \mathbf{A}^{(n)}$, is performed by minimizing the loss between $\mathcal{X}$ and $\mathcal{M}$ as defined by an objective function. The form of the objective function is determined by assumptions about how the data in the tensor was generated. A standard method of fitting a CP decomposition is the least squares formulation, which assumes the random variation in the tensor data follows a Gaussian distribution. Unfortunately, this may not be appropriate for count data, which is common in many applications including those considered in this paper [17].

A more appropriate objective function for count data assumes the underlying distribution of the data is Poisson [8]. This assumption results in an objective function that minimizes the Kullback-Leibler (KL) divergence:

$$f(\mathcal{M}) = \sum_i m_i - x_i \log m_i.$$

There have been several recent works to accelerate the computational speed of the CP decomposition. FlexiFact partitions the tensor into smaller tensors that are then decomposed in parallel using Apache Hadoop. While their method scales well with the input size, the model is not well-suited for count data and performs excessive disk operations. More recently, DisTenC, a Spark-based distributed tensor completion algorithm was proposed that regularizes the trace norm of the tensor [11]. Similar to FlexiFact, it is designed for numeric data and does not support any regularization or constraints on the factors.

3 SGRANITE

We propose `SGranite`, a distributed and flexible constrained CP model, to impose a variety of constraints on the latent factors. Our algorithm uses distributed stochastic gradient descent (DSGD) approaches to scale the CP decomposition on count data to huge datasets. `SGranite` has the following benefits:

- Simultaneously supports multiple constraints on the factor matrices.
- Learns patterns even when data cannot be stored on a single server.
- Maintains computational efficiency across a large number of workers.

[1] https://github.com/hehuannb/SGranite-WWW

Table 1: A comparison of the supported features between SGranite versus state of arts methods

Feature	SGranite	CP-APR [8]	Granite [16] (fit using SGD)	FlexiFact[6]	DisTenC [11]
Scalable	Yes	No	Yes	Yes	Yes
Memory Efficient	Yes	No	Yes	No	Yes
Time efficient	Yes	No	No	Yes	Yes
Appropriate for count data	Yes	Yes	Yes	No	No
Works with constraints	Yes	No	Yes	Yes	No

Table 2: Table of symbols and their associated definitions

Symbol	Definition
$\mathcal{X}$, $\mathbf{X}$, $\mathbf{x}$, x	Tensor, Matrix, Column Vector, Scalar
$\mathbf{X}_{(n)}$	n-mode matricization of a tensor $\mathcal{X}$
$\mathbf{X}(\mathbf{r},:)$	rth row of $\mathbf{X}$
$\mathbf{X}(:,\mathbf{r})$	rth column of $\mathbf{X}$
$\mathbf{A}^{(n)}$	nth factor matrix
$\|a\|_2$, $\|\mathbf{A}\|_F$	l_2 norm, Frobenius norm
$*$	Hadamard (elementwise) product
$\circ$	outer product
$\otimes$	Kronecker product
$\odot$	Khatri-Rao product (column-wise $\otimes$)

A distributed framework for incorporating a variety of constraints in CP decomposition is appealing for several reasons including the ability to extract patterns from large datasets that cannot be readily stored on a centralized server, to encode prior knowledge, to improve interpretability, and to democratize high-dimensional learning by running on standard commodity servers.

In this section, we will first provide a general overview and then formulate the optimization problem.

3.1 General Optimization Problem

SGranite, builds on several existing nonnegative CP decomposition algorithms to model sparse count data using the Poisson distribution [8, 16]. Let $\mathcal{X}$ denote an observed tensor constructed from count data with size $I_1 \times I_2 \times \cdots \times I_N$ and M represent a same-sized tensor of Poisson parameters for $\mathcal{X}$. In addition to KL divergence, we introduce generalized constraints on the factor matrices, $\mathcal{R}(\mathbf{A}^{(n)})$ to the objective function. Thus, the optimization problem is defined as:

$$\begin{aligned} \min f(\mathcal{M}) &= \sum_{\vec{i}} (m_{\vec{i}} - x_i \log m_{\vec{i}}) + \sum_k \underbrace{\beta_k \mathcal{R}_k\left(\mathbf{A}^{(n)}\right)}_{\text{regularization terms}} \\ \text{s.t. } \mathcal{M} &= [\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}, \cdots, \mathbf{A}^{(N)}]\!] \\ \lambda_r &\geq 0, ||\mathbf{a}_r^{(n)}||_1 = 1, \ \forall r \\ \mathbf{A}^{(n)} &\in [0,1]^{I_n \times R}, \ \forall n \end{aligned} \tag{1}$$

The Poisson parameters, $\mathbf{m}$, can be determined by minimizing the negative log-likelihood of the observed data $\mathbf{x}$. We also maintain the stochasticity (i.e., elements sum to 1) and non-negativity constraints (i.e., factor elements and weights, or λ, must be non-negative) that were introduced in the original CP-APR model [8].

3.2 Example of Useful Regularization Terms

Equation 1 supports a variety of regularization items, $\mathcal{R}(\mathbf{A}^{(n)})$. While we describe three forms of special regularizations that are useful for analyzing health data, SGranite was developed to handle any regularization that is either smooth and differentiable or has an easy-to-compute proximal operator [31].

3.2.1 Diversity on $\mathcal{A}^{(n)}$. For analyzing flu patterns or clinical characteristics of patient subgroups, it is preferable for the rank-one factor components to be distinct from each other. This allows domain experts to more easily interpret the patterns. While several mechanisms for encouraging diversity have been proposed [16, 21, 36], we adopt the angular penalty term in [16] that encourages diversity between rank-one tensors by penalizing overlapping elements. There are two benefits to this regularization. It does not require prior knowledge to construct a similarity matrix that is used in [21]. Similarly, it does not require the discovered patterns to be orthogonal to one another [36], which may be too restrictive. Under angular regularization, any element that has large values in multiple columns in the factor matrix are penalized. Thus, the angular penalty for the n^{th} factor matrix, $\mathbf{A}^{(n)}$, is formulated as follows:

$$\mathcal{R}_k\left(\mathbf{A}^{(n)}\right) = \sum_{r=1}^{R} \sum_{p=1}^{r} \max\left(0, \frac{(a_p^n)^T a_r^n}{\|a_p^n\|_2 \|a_r^n\|_2} - \theta_n\right)^2$$

3.2.2 Sparsity and Smoothness on $\mathcal{A}^{(n)}$. Sparsity and smoothness constraints have been introduced in a wide range of applications to improve interpretability and increase robustness to noise. Our framework supports a general class of ℓ_p penalties including simplex constraint term ($||\mathbf{a}_r||_1 = 1, a_{ir} \in [0,1]$); ℓ_2 regularization on the weight and the first factor matrix, $\lambda\mathcal{A}^{(1)}$ to mitigate overfitting to large count data; and the ℓ_0-norm regularization which caps the number of non-zeros elements in the factor.

We first consider the simplex constraint term, which can yield sparse factors while providing a probabilistic interpretation. For the n^{th} factor matrix, $\mathbf{A}^{(n)}$, we restrict the elements to lie on the ℓ_1-ball of diameter s, where s is a user-specified parameter, such that:

$$\mathcal{R}_k\left(\mathbf{A}^{(n)}\right) = \sum_{r=1}^{R} (s - ||\mathbf{a}_r^{(n)}||_1)$$

When $s = 1$, this results in the projection of the factor onto the probabilistic (or canonical) simplex [9]. By decreasing s to be less than 1, the resulting factors will be sparser.

The ℓ_2-norm regularization was introduced in [16] to encourage terms in the factor matrix vectors to be similar-sized. Together with the simplex projection, the interaction of these two regularizations achieved further sparsity by driving specific elements to 0 more quickly in a similar manner to the elastic net regularization [41].

$$\mathcal{R}_k\left(\mathbf{A}^{(n)}\right) = \sum_{r=1}^{R} \|a_r^{(n)}\|_2$$

The ℓ_0-norm regularization, introduced in [4], is an alternative to the simplex projection that limits the number of non-zero elements. While its usage in Equation 1 results in a non-convex optimization problem, the hard thresholding properties can yield easy to interpret factors (top-k elements). To perform hard-thresholding on the n^{th} factor matrix, $\mathbf{A}^{(n)}$, the regularization term is:

$$\mathcal{R}_k\left(\mathbf{A}^{(n)}\right) = \sum_{r=1}^{R} \|a_r^{(n)}\|_0$$

3.2.3 Discriminative Factors. In some scenarios, the discovered patterns should be discriminative of a certain outcome of interest. For example, we may want to use the clinical characteristics to predict things like mortality or whether or not the patient is likely to be readmitted in 30 days. [21] introduced a logistic regression regularization that encouraged the derivation of latent factors that can distinguish in-hospital mortality outcomes. `SGranite` also adopts the regularization term to derive discriminative latent factors when such information exists. Without loss of generality, we assume that the first mode has labeled records. Then the discriminative regularization is of the form:

$$\mathcal{R}_k\left(\mathbf{A}^{(1)}\right) = \log P(\mathcal{A}^{(1)}, y|\theta)$$

The probability of a sample $\mathbf{a}(i,:)$ (i^{th} row in $\mathbf{A}^{(1)}$) having the outcome of interest, $P(\mathcal{A}^{(1)}, y|\theta)$, is obtained by training a logistic regression model on the factor matrix $\mathbf{A}^{(1)}$.

3.2.4 Sparse, Diverse, and Discriminative Patterns. To demonstrate the flexibility of `SGranite`, we introduce all three forms of regularization into our final optimization problem. Thus, the final objective function is:

$$\begin{aligned} f(\mathcal{M}) = &\sum_{\vec{i}} (m_{\vec{i}} - x_i \log m_{\vec{i}}) + \\ &\beta_1 \sum_{n=1}^{N} \sum_{r=1}^{R} \sum_{p=1}^{r} \max\left(0, \frac{(a_p^n)^T a_r^n}{\|a_p^n\|_2 \|a_r^n\|_2} - \theta_n\right)^2 + \\ &\beta_2 \sum_{n=1}^{N} \sum_{r=1}^{R} (s - ||\mathbf{a}_r^{(n)}||_2) + \beta_3 \log P(\mathcal{A}^{(1)}, y|\theta) \end{aligned} \tag{2}$$

3.3 SGD Updates

This section provides details of how to solve our optimization problem efficiently (Equation 2). `SGranite` uses an alternating minimization approach, cycling through each mode while fixing all the other modes. For each mode, the resulting subproblem is solved using stochastic gradient descent (SGD). To derive the SGD updates, we first re-write the objective function as a scalar-valued function of the parameter vector y using the same approach as [3]. The parameter vector y represents the vectorization of the factor matrices, with the weights $\boldsymbol{\lambda}$ absorbed into the first factor matrix.

$$y = \begin{bmatrix} vec(\boldsymbol{\lambda}\mathbf{A}^{(1)}) \\ vec(\mathbf{A}^{(2)}) \\ \vdots \\ vec(\mathbf{A}^{(n)}) \end{bmatrix}$$

As a result, the gradients of the objective function can be formed by vectorizing the partial derivatives with respect to each component of this parameter vector:

$$\nabla f(y) = \left[vec(\frac{\partial f}{\partial \mathbf{A}^{(1)}}) \cdots vec(\frac{\partial f}{\partial \mathbf{A}^{(n)}}) \right]$$

For notational convenience, we also represent the matricized form of the tensor decomposition as:

$$[\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}, \cdots, \mathbf{A}^{(N)}]\!]_{(n)} = \boldsymbol{\lambda}\mathbf{A}^{(n)}(\mathbf{A}^{(-n)})^T$$

where

$$\mathbf{A}^{(-n)} = \mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)}.$$

Thus, the partial derivatives of Equation 2 with respect to the factor matrix, $\mathbf{A}^{(n)}$ are the following:

$$\begin{aligned} \frac{\partial f}{\partial A_r^{(n)}} = &\left[1 - X_{(n)} \oslash Z_{(n)}\right] a_r^{(-n)} + \\ &\beta_1 \sum_{p \neq r} \max\left(0, g(a_r^{(n)}, a_p^{(n)})\right) \frac{\partial g(a_r^{(n)}, a_p^{(n)})}{\partial \mathbf{a}_r^{(n)}} + \\ &\beta_2 \mathbf{a}_r^{(n)} + \beta_3 y \frac{1}{1 + \exp(\mathbf{y}\mathbf{A}_r^{(1)})} \theta \end{aligned} \tag{3}$$

We refer the reader to [16, 21] for the detailed derivation of the gradients.

For large datasets, the calculation of the derivatives simultaneously for all modes is computationally expensive. Thus, `SGranite` uses an SGD approach to avoid storing the entire tensor in memory. For faster convergence, we adopt a variant of SGD named Adaptive Moment Estimation (Adam) to adaptively update the learning rate [23]. Our preliminary experiments on a single machine showed that SGD with Adam converged faster and more accurately than using a fixed learning rate.

Algorithm 1 SGD updating process

1: **for** $l = 1 : L$ **do**
2: Randomly select n samples
3: Calculate the gradients for samples using Equation 3
4: Compute the decaying averages of past and past squared gradients
5: Take a step using averaged gradients
6: **end for**

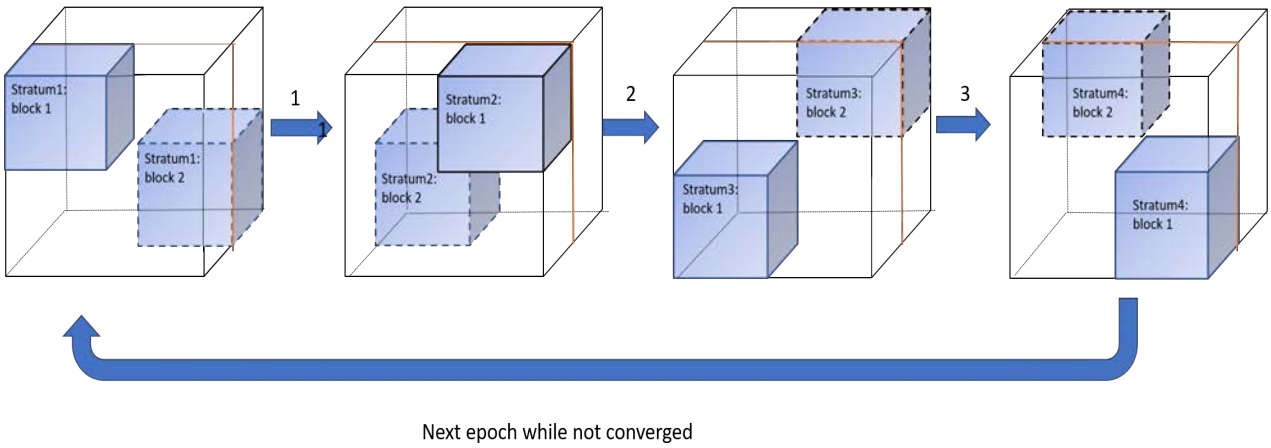


Figure 2: A graphical example of our SGranite: Suppose there are 2 workers, we will have 8 blocks and 4 strata after partition. We run this process iteratively until convergence. In each epoch, start from strata one, each worker runs SGD for its own assigned block in parallel. Check the convergence until all strata are iterated. We repeat above algorithm again if the stopping criteria is not satisfied. All intermediate results are saved as Resilient Distributed Datasets (RDD) collections and cached in memory.

3.4 Parallel Algorithm using Spark

Although SGD scales well to sparse data, we would like to distribute the computation to achieve results faster. FlexiFact proposed distributing the computation by dividing the tensor such that no two blocks share any elements (along with any dimension) [6]. Thus, the SGD algorithm can be run in parallel on each block without sacrificing accuracy. We refer the reader to [6, 12] for detailed proof of convergence. SGranite uses the same approach to distribute the non-zero elements of the count tensor using this tensor partition. However, we note several main differences between our framework and Flexifact: (1) support for sparse, count data by using an appropriate objective function (KL divergence), (2) flexibility to incorporate a variety of constraints beyond sparsity and non-negativity, and (3) distributed computation using Apache Spark.

Unlike SGranite, FlexiFact uses the Hadoop Map-Reduce platform to distribute the data collection across multiple nodes. Unfortunately, a Hadoop workflow spends an exorbitant amount of time on disk operations, as it needs to read and write intermediary results on the disk. On the other hand, Apache Spark [40] has been proposed as an alternative that eliminates unnecessary disk operations for iterative algorithms. By performing the data analytic operations in-memory and in near real-time, Spark can achieve lower computation times. Thus, SGranite is developed using Spark to distribute the computation.

3.4.1 Tensor Partition. First, we define a stratum as a set of independent blocks, and we denote the number of blocks in each stratum by d. Suppose we have d available workers, in order to iterate all regions of $\mathcal{X}$, we need d^3 blocks and thus d^2 strata. For a stratum s we have d blocks $Z_i^{(s)}$ for $i = 0, 1, \cdots, n-1$. A detailed partition function for a size of $I \times J \times K$ tensor $\mathcal{X}$ is provided below:

$$\begin{aligned} b_i &= (i\lceil I/d \rceil, (i+1)\lceil I/d \rceil) \\ b_j &= (j\lceil J/d \rceil, (j+1)\lceil J/d \rceil) \\ b_k &= (k\lceil K/d \rceil, (k+1)\lceil K/d \rceil) \\ j_{s,i} &= (j+s) \\ k_{s,i} &= (j+s) \mod d \\ Z_i^{(s)} &= \mathcal{X}_{(b_i, b_{j_{s,i}}, b_{k_{s,i}})} \end{aligned} \tag{4}$$

Figure 2 provides an example of how to divide a count tensor for 2 available workers.

3.4.2 Block Parallelization. Prior to introducing how SGranite iteratively solves the optimization problem in parallel, we introduce some definitions. A full epoch is defined as when the algorithm has seen all the d^3 blocks in the tensor. Since we need d^2 strata to cover all the blocks, we need to perform d^2 inner iterations. Therefore, we refer to each stratum training as a single inner iteration. Thus, in SGranite, the computation of each stratum is performed sequentially in each epoch. But for each stratum, we run SGD on the d^2 blocks in parallel. After each inner iteration, we update the factor matrices and use them as the initialization for the next stratum. Figure 3 provides an example of training using a single stratum. Upon the completion of an epoch (all strata have been run), the factor matrices are combined from all the workers, and then re-normalized for identifiability. The normalization can be performed for a user-specified mode, otherwise it defaults to the first mode. Convergence is checked between epochs by measuring the changes in the KL divergence to see if it is below a given tolerance. The details for the parallel-version of SGranite is described in Algorithm 2.

3.4.3 Spark Implementation Details. The non-zero elements of the count tensor are stored in a list using the coordinate format and

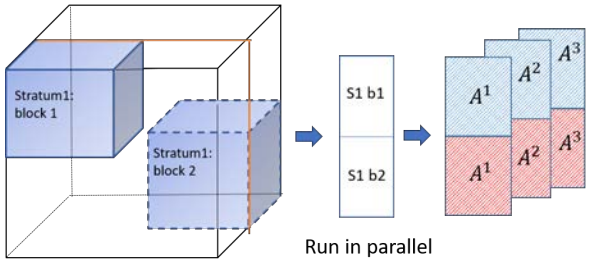


Figure 3: A graphical example of one stratum training: Given one stratum of training data and factor matrices $A^{(1)}, A^{(2)}, A^{(3)}$, we run SGD on each block in parallel. Then factor matrices $A^{(1)}, A^{(2)}, A^{(3)}$ are updated and used as the initialization for the next stratum training.

Algorithm 2 `SGranite`

Randomly initialize $[\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}; \cdots; \mathbf{A}^{(N)}]\!]$
Partition the tensor and construct d^2 stratas using 4
for $m = 1 : M$ **do**
  for $l = 1 : d^2$ **do**
    Assign each block in lth strata to a worker
    Each worker runs Algorithm 1 in parallel
    Update the factors $[\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}; \cdots; \mathbf{A}^{(N)}]\!]$
  end for
  Gather results from each worker
  Normalize factor matrices according to the specified mode
  if converged **then**
    break
  end if
end for
Return $[\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}; \cdots; \mathbf{A}^{(N)}]\!]$

loaded as Resilient Distributed Datasets (RDDs), and then it is shared throughout our cluster as a broadcast variable. A broadcast variable in Spark is immutable, meaning that it cannot be changed later on. This may seem inconvenient but it truly suits our case since we only need to read values from the tensor to calculate gradients in each iteration.

We do not broadcast factor matrices since we need to update them in each iteration. Due to our partition function, each worker has a chance to update factors matrices with different boundaries. The best way is to partition factor matrices using Block ID. In this way, we can reduce the memory and communication cost. Specifically, we applied `map` and `aggregateByKey` functions to partition the factor matrices into blocks. The function `map` transforms each entry of the sparse tensor into an element in the RDD whose key is a block ID. Then `aggregateByKey` groups each block together and persists in memory. In each inner iteration, we use `groupWith` to build a stratum partitioned using `partitionBy` and then use `mapPartitions` to assign tasks to each node.

We found storing factor matrices RDDs and partitioned result in a significant acceleration, but not doing this will cause virtual memory issues in our experiments. Our experiments suggest such a design will enable us to obtain better speed-up and scalability.

4 EXPERIMENTS AND RESULTS

In this section, we first provide descriptions for two real-world health datasets. We then give an overview of baseline methods and provide qualitative and quantitative results.

4.1 Datasets

We use the following two publicly available datasets:

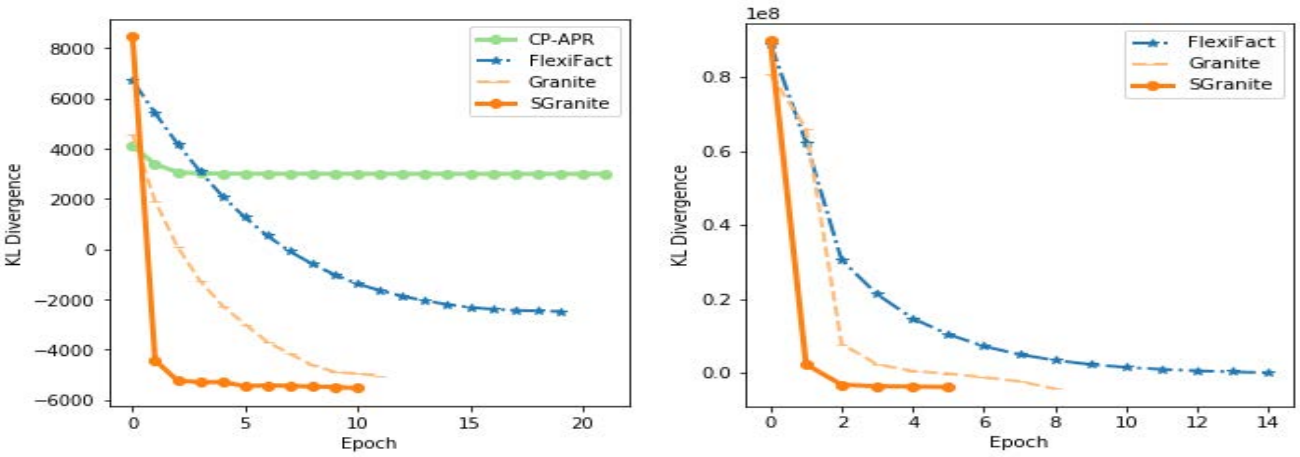
- Influenza: Using Google Flu Trends historical data[2] from 2003 to 2015, we generated a tensor to uncover temporal influenza patterns that are unique and similar across multiple states. For each region in the United States, we collected the number of search queries related to influenza on a weekly basis over 11 years. The resulting tensor is 12 regions × 52 weeks × 11 years. Although the data quality has been shown to be low Olson, Donald R et al. [28], this dataset is used to demonstrate the feasibility of `SGranite` on search data.
- MIMIC-III [20]: MIMIC-III is large database containing de-identified health data associated with approximately sixty thousand admissions of critical care unit patients from the Beth Israel Deaconess Medical Center collected between 2001 and 2012. For each patient, we extract medications and the International Classification of Diseases (ICD-9) diagnosis codes. ICD-9 codes are aggregated using Clinical Classification Software (CCS) categories[3], a standard preprocessing step in healthcare analysis. Similarly, medications are grouped using the Anatomical Therapeutic Chemical (ATC) Classification via the RxNorm RESTful Web API, a web service developed by the National Library of Medicine[4]. The aggregation step results in a 38159 patient × 234 diagnosis × 511 medication tensor.

4.2 Baseline Approaches

We will compare `SGranite` to both centralized and distributed CP decomposition methods.

- CP-APR [8]: The first algorithm proposed for modeling sparse count data using a Poisson distribution. There is no support for constraints, and the updates are performed using multiplicative updates. The algorithm has been ported to Python by the authors of [16].
- Granite [16]: A centralized extension of CP-APR that incorporates the angular penalty, ℓ_2, and the simplex projection as regularization terms. The authors shared a python implementation of Granite fit using SGD.
- FlexiFact [6]: A distributed algorithm based on the DSGD approach that factorizes a coupled tensor and matrix using a similar partition method. However, it uses least squares as an objective and only supports non-negativity and ℓ_1 constraints. For a fair comparison, we implemented the algorithm in Spark according to the paper.

[2] https://www.google.org/flutrends
[3] The mapping from ICD-9 to CCS can be found at https://www.hcup-us.ahrq.gov/toolssoftware/ccs/ccs.jsp.
[4] Details for the RxNorm RESTful Web API can be found at https://mor.nlm.nih.gov/download/rxnav/RxNormAPIREST.html.



(a) Influenza **(b) MIMIC**

Figure 4: Comparison of two distributed and two non-distributed CP models using KL divergence. SGranite converges in less epochs than the other methods. The negative KL divergence arises from the fact that the observed values are not probability measurements.

4.3 Implementation and Hardware Details

SGranite is implemented in Python and the source code is publicly available[5]. The experiments were conducted on AWS. The cluster has one master and three worker nodes. Each node has four virtual cores and 16 GB of RAM. Results in our paper were reported using 4 workers.

4.3.1 Hyperparameter tuning. The logistic regularization penalty β_3, simplex projection β_2 and the angular penalty β_1 were set as 0.06, 0, and 0.02, respectfully. θ_n for angular penalty for each mode, the learning rate, and batch sizes were selected as 0.9, 0.0001, and 200 respectively. All hyperparameters were chosen based on a grid search over values.

4.4 Results

4.4.1 Scalability and Accuracy. First, we assess the quality of the approximation (measured by KL divergence) for SGranite and the other baseline methods. Figure 4 shows the KL divergence as the function of the number of epochs on both datasets. For the centralized algorithms (CP-APR and Granite), each epoch corresponds to a full iteration. The plots demonstrate that SGranite converges at least 4× faster than FlexiFact and also faster than the centralized algorithms. Moreover, the quality of the approximation is better than any of the existing methods. This suggests that SGD-based methods may help escape undesirable local minima (compared to CP-APR). The figure also highlights the importance of appropriately modeling the data distribution as opposed to using the least-squares loss (FlexiFact) may not yield the best approximation.

Next, we evaluate the scalability of our algorithm with respect to the number of workers. We calculate the speed-up as the ratio between the total execution time and the sequential execution time. Figure 5 demonstrates the speed-up of SGranite with respect to the number of workers. As can be seen in the figure, the speed up for the MIMIC tensor is very close to the ideal speed-up, as it is relatively large. However, there is a limited improvement on the Influenza tensor, a small dataset. This is due to the communication cost that is incurred in coordinating the different nodes. We note that because SGranite caches the updated factor matrices in memory to minimize disk accesses between consecutive iterations, it is able to scale to a large dataset and a large number of workers. Since this speed-up would not be possible on a system like Hadoop, we do not provide a comparison with FlexiFact.

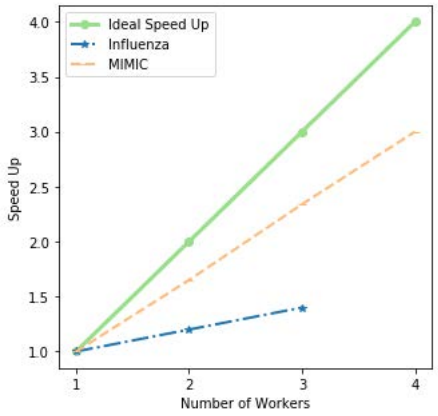
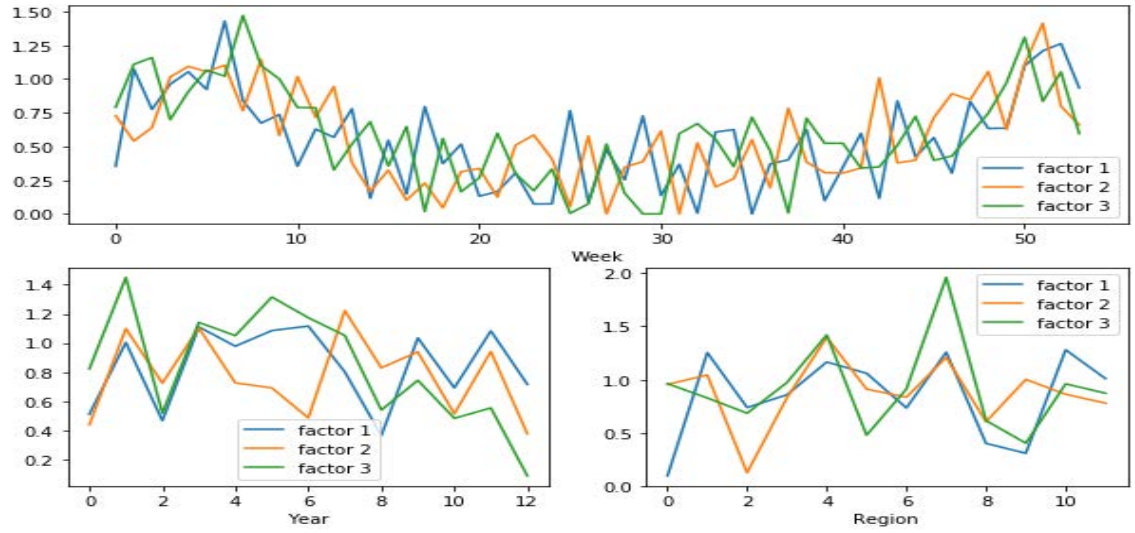


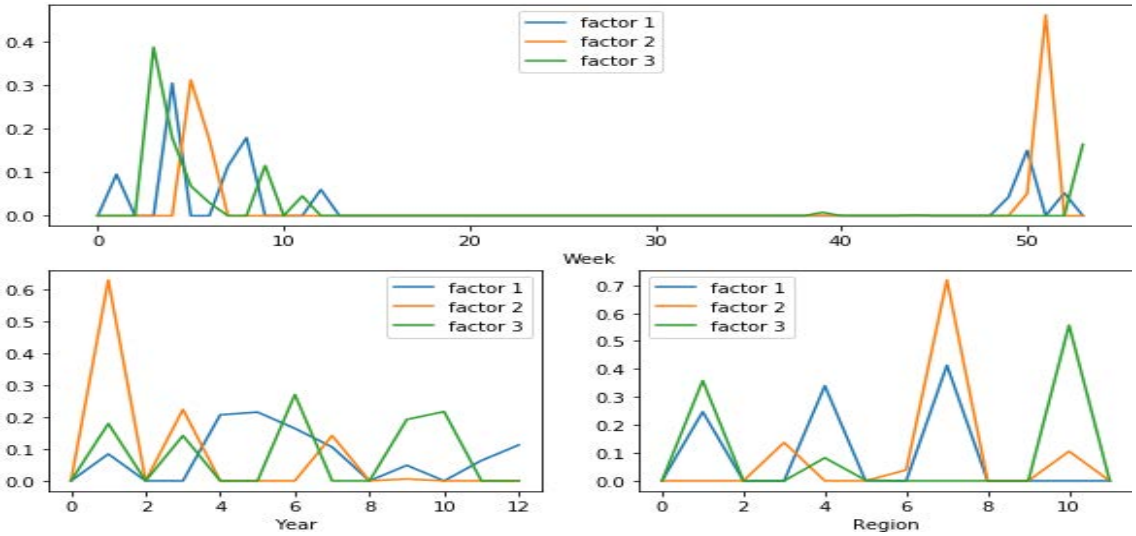
Figure 5: The speed-up curve for both datasets. It shows analysis of large datasets will gain an obvious speed up by using SGranite.

4.4.2 Qualitative and Quantitative Assessment of the Constraints. To examine the impact of the constraints, we first compare the results from SGranite on the influenza dataset both with and without the angular and simplex regularization terms. Figure 6a shows the

[5]https://github.com/hehuannb/SGranite-WWW



(a) No angular penalty and simplex projection ($\beta_1 = \beta_2 = 0$)



(b) Angular penalty and simplex projection constraints

Figure 6: A comparison of the learned latent factors with and without constraints using $R = 3$. Year from 2003 to 2015

latent factors learned without the regularization terms, and Figure 6b shows the latent factors learned with the regularization terms. From these plots, we observe that the learned factors without regularization are highly correlated and can be difficult to distinguish from one another. In Figure 6a, it is hard to discern any noticeable pattern across the weeks and the different regions. In comparison, Figure 6b demonstrates the potential of incorporating both diversity and sparsity. We can observe that factor 2 predominantly captures the peak influenza season that occurs both towards the end of December and in mid-February in region 7, whereas factor 3 is slightly delayed and captures the influenza trend in regions 1 and 10. Furthermore, all three factors capture the peak in influenza season that occurs in late December and early February through March.

Next, we quantitatively assessed the impact of the logistic regression and angular penalty on the MIMIC-III dataset. To evaluate the discriminative power and distinctiveness of the learned factors, we used the in-hospital mortality cohort similar to that proposed in [21]. We used 37,000 patients, including all 5,014 patients who died during admission. We split our dataset into 80% training and 20% testing. We measured the discrimination on the test set using the area under the receiver operating characteristic curve (AUC). Distinctiveness is measured using the average overlap or the degree of overlapping between latent factors. It is defined as the average of cosine similarities between all latent factor pairs:

$$\text{Avg Overlap} = \frac{\sum_{r_1}^{R} \sum_{r_2 > r_1}^{R} \cos(a_{r_1}^{(2)}, a_{r_2}^{(2)}) + \cos(a_{r_1}^{(3)}, a_{r_2}^{(3)})}{R(R-1)}$$

Table 3 summarizes the AUC, total computation time (or running time), and the average overlap. We observe that SGranite can not only accelerate the tensor decomposition but also provides better prediction than other baseline methods. Moreover, the average overlap is smaller than Granite even without the angular constraints. This suggests that the partition function may also have some beneficial impact in terms of reducing overlapping factors. Moreover, incorporating the angular constraints further helps the discriminative ability of the model. This suggests that adding diversity constraints to yield less correlated latent features may also help the resulting predictive model. Therefore, SGranite supports a variety of flexible constraints and yields improved predictive performance.

Model	AUC	Time	Avg Overlap
CP-APR [8]	0.63	> 1 hour	0.3
FlexiFact [6]	0.65	35 mins	0.37
Granite [16]	0.67	> 1 hour	0.3
SGranite ($\beta_1 = 0$)	0.68	**20 mins**	0.1
SGranite ($\beta_1 > 0$)	**0.71**	25 mins	**0.07**

Table 3: Table of AUC, running time, and average overlapping using different methods. The highest AUC value means extracted phenotypes have stronger discrimination. The lowest running time indicates our distributed method can significantly accelerate the computation time. Compared to CP-APR and FlexiFact, adding angular penalty improved the distinction significantly.

4.4.3 Case Study 1: Flu Patterns. We provide a further qualitative assessment of our learned latent patterns from the influenza dataset. First, we comment on the ability to capture the overall flu season trends. Although flu season can vary across region to region, the flu season is typically between October through May (week 43 to week 22) [1]. We observe this phenomenon even with and without angular penalty constraints as illustrated in Figure 6. The variance in region and slight shifts in the week are further evident when angular penalty and simplex projection constraints are present (Figure 6b). We can see that some of the regions are present only in 1 of the factors. Moreover, slight shifts along the week are observed (top chart), depending on which latent factor with the higher elements occurring between weeks 48 and 13. This provides further confirmation that each region will have slightly different times when influenza will be more prominent.

We also assessed the learned flu patterns with FlexiFact, the other distributed CP algorithm that supports non-negativity and sparsity. Figure 7 presents the learned latent factors using FlexiFact. We observe that the peak level regions that are discovered using SGranite are more consistent with the CDC influenza positive test results, shown in Figure 8. The peaks that are discovered by FlexiFact are inconsistent with the observed CDC reports. FlexiFact factors suggest two different peaks, one between weeks 8-10 and one 18-20, whereas the CDC reports note a peak around 7-9 and by week 20, it has mostly died down. Moreover, we observe that the FlexiFact latent factors are more difficult to interpret as the region and year factors are fairly correlated. We also compared with the learned factors from a previous study [13] and found our learned patterns were more consistent with the observed results. This suggests that the incorporation of constraints not only improves interpretability but also provides robustness to noise.

4.4.4 Case Study 2: Phenotypes. We conducted a second case study to examine SGranite's ability to extract discriminative and distinct clinical characteristics from the MIMIC III dataset. The identification of clinical phenotypes from EHR data can help advance our understanding of disease risk and drug response as well as support the practice of precision medicine on a national scale [33, 38].

For clinicians, diversity is important to discover rare phenotypes in a patient population. Moreover, diverse phenotypes are likely easier to implement, as a clinician may find it difficult to rank-order or apply phenotypes that have substantial overlap. In addition, discriminative phenotypes are better predictors of mortality (shown in Table 3) and thus can be used to assist the decision-making process.

Table 4 presents the learned phenotypes that are important where importance is determined based on the magnitude of the phenotypes (or λ_r). Thus, these are the three sets of patient characteristics at which diagnosis and medication are dominant. First, we observe that the learned phenotypes have limited number of overlapping elements. In table 4, the most significant phenotype (λ_1) captures acute complications with heart diseases which can be riskier. In particular, acute respiratory distress syndrome has a mortality rate of 30-50% and is associated with long hospital stays [27]. The third phenotype (λ_3) captures more chronic diseases such as heart valve disorder, leukemias, and osteoarthritis. In addition, we observe that most medication codes in Table 4 are associated with diagnosis codes above. For example, potassiuman chloride and practolol are commonly used to lower blood pressure in hypertensives [10, 37]. An ACE inhibitor is used primarily for the treatment of hypertension and congestive heart failure [26]. And magnesium carbonate has shown to be effective for chronic kidney diseases and intracranial injury [5, 35].

5 CONCLUSION

In this paper, we presented a distributed, diverse, non-negative tensor decomposition framework that supports a variety of constraints including an angular penalty to encourage diversity and a simplex projection to encourage sparsity while scaling to large tensors. By imposing such regularization terms, SGranite successfully extracts meaningful latent factors in two real-world use cases. Moreover, by using Spark, SGranite successfully reduces processing time by dramatically reducing the workload and high communication cost. In addition, SGranite improves binary prediction tasks by incorporating logistic supervision into the fitting process. In the future, we plan to develop a distributed algorithm that can handle linear regression problem and also an extension that can use outside data sources as the guidance information.

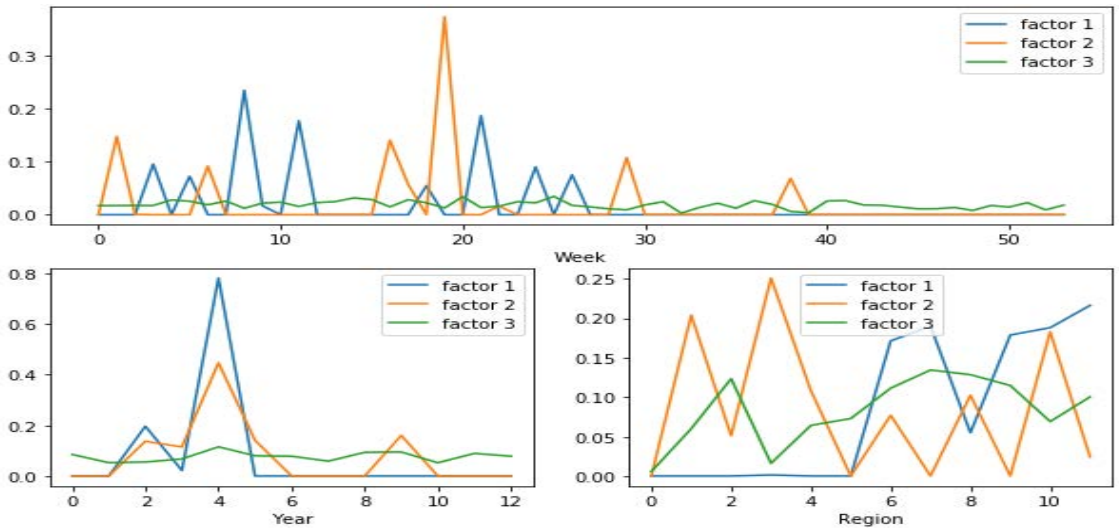


Figure 7: Latent factors obtained using Flexifact. Year from 2003 to 2015

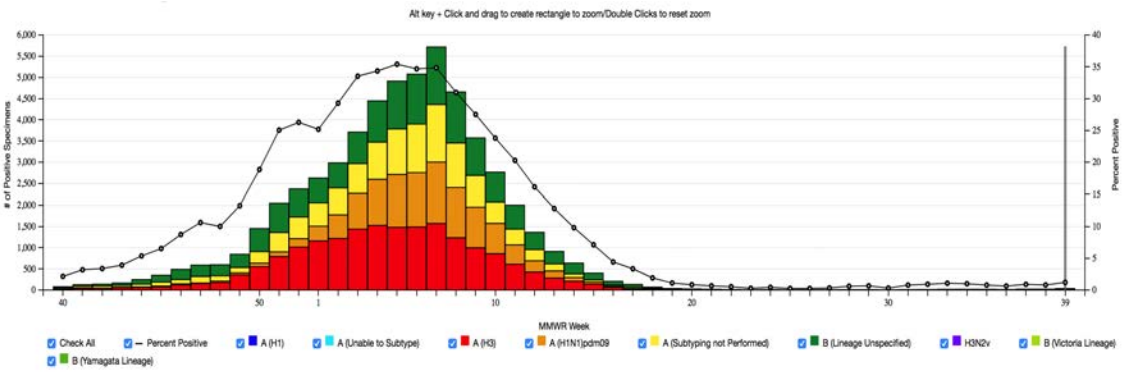


Figure 8: This figure is downloaded from CDC, it shows the actual influenza positive tests reported to CDC in 2010-2011, week ending Oct 01, 2001.

Phenotype 1: $\lambda_1 = 87$	**Phenotype 2:** $\lambda_1 = 79$	**Phenotype 3:** $\lambda_3 = 77$
Respiratory distress syndrome	Coronary atherosclerosis & other heart disease	Heart valve disorders
Acute cerebrovascular disease	Intracranial injury	Intracranial injury
Coronary atherosclerosis & other heart disease	Congestive heart failure; nonhypertensive	Leukemias
Hypertension w/ complications & secondary hypertension	Acute and unspecified renal	Osteoarthritis
Potassium chloride	Oxyphenisatine	Practolol
Sodium chloride	Adrenergics, inhalants	Magnesium carbonate
Omeprazole	ACE inhibitors	Anilides
Potassium chloride	Nitroprusside	Sodium chloride

Table 4: Table of top 3 phenotypes with high λ. Upper four rows are diagnosis codes and four rows below are medication codes correspondingly

6 ACKNOWLEDGEMENT

This work was supported by the National Science Foundation award IIS-#1838200 and the National Institute of Health award 1K01LM012924-01.

REFERENCES

[1] [n. d.]. Flu Season. https://en.wikipedia.org/wiki/Flu_season

[2] [n. d.]. National Institutes of Health. https://allofus.nih.gov/

[3] Evrim Acar, Daniel M Dunlavy, and Tamara G Kolda. 2011. A scalable optimization approach for fitting canonical tensor decompositions. *Journal of Chemometrics* 25, 2 (Feb. 2011), 67–86.

[4] Ardavan Afshar, Ioakeim Perros, Evangelos E Papalexakis, Elizabeth Searles, Joyce C Ho, and Jimeng Sun. 2018. COPA: Constrained PARAFAC2 for sparse & large datasets. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 793–802.

[5] M F Arango and J H Mejia-Mantilla. 2006. Magnesium for acute traumatic brain injury.

[6] Alex Beutel, Partha Pratim Talukdar, Abhimanu Kumar, Christos Faloutsos, Evangelos E Papalexakis, and Eric P Xing. 2014. FlexiFaCT - Scalable Flexible Factorization of Coupled Tensors on Hadoop. *SDM* (2014).

[7] J Douglas Carroll and Jih-Jie Chang. 1970. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. *Psychometrika* 35, 3 (1970), 283–319.

[8] Eric C Chi and Tamara G Kolda. 2012. On tensors, sparsity, and nonnegative factorizations. *SIAM J. Matrix Anal. Appl.* 33, 4 (2012), 1272–1299.

[9] John C Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. 2008. Efficient projections onto the l1-ball for learning in high dimensions. *ICML* (2008).

[10] Tommaso Filippini, Federica Violi, Roberto D'Amico, and Marco Vinceti. 2017. The effect of potassium supplementation on blood pressure in hypertensive subjects: A systematic review and meta-analysis. *International Journal of Cardiology* 230 (March 2017), 127–135.

[11] Hancheng Ge, Kai Zhang, Majid Alfifi, Xia Hu, and James Caverlee. 2018. DisTenC: A aistributed algorithm for scalable tensor completion on Spark. In *Proceedings of the 2018 IEEE 34th International Conference on Data Engineering*. 137–148.

[12] Gemulla, Rainer, Nijkamp, Erik, Haas, Peter J, and Sismanis, Yannis. 2011. *Large-scale matrix factorization with distributed stochastic gradient descent.* ACM, New York, New York, USA.

[13] Michael Joseph Haass, Mark Hilary Van Benthem, and Edward M Ochoa. 2014. *Tensor analysis methods for activity characterization in spatiotemporal data.* Technical Report.

[14] Harshman, R A. 1970. Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multimodal factor analysis. (1970).

[15] Jette Henderson, Ryan Bridges, Joyce C Ho, Byron C Wallace, and Joydeep Ghosh. 2017. PheKnow-Cloud: A Tool for Evaluating High-Throughput Phenotype Candidates using Online Medical Literature. *AMIA Joint Summits on Translational Science proceedings. AMIA Joint Summits on Translational Science* 2017 (2017), 149–157.

[16] Jette Henderson, Joyce C Ho, Abel N Kho, Joshua C Denny, Bradley A Malin, Jimeng Sun, and Joydeep Ghosh. 2017. Granite - Diversified, Sparse Tensor Factorization for Electronic Health Record-Based Phenotyping. *ICHI* (2017).

[17] Jette Henderson, Bradley A Malin, Joshua C Denny, Able N Kho, Jimeng Sun, Joydeep Ghosh, and Joyce C Ho. 2019. CP Tensor Decomposition with Cannot-Link Intermode Constraints. In *SDM*.

[18] Joyce C Ho, Joydeep Ghosh, and Jimeng Sun. 2014. Marble - high-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization. *KDD* (2014).

[19] Ginsberg J, Mohebbi MH, Patel RS, Brammer L, Smolinski MS, and Brilliant L. 2008. Detecting influenza epidemics using search engine query data. *Nature* 457, 7232 (2008).

[20] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data* 3 (2016), 160035.

[21] Yejin Kim, Robert El-Kareh, Jimeng Sun, Hwanjo Yu, and Xiaoqian Jiang. 2017. Discriminative and Distinct Phenotyping by Constrained Tensor Factorization. *Scientific Reports* 7, 1 (April 2017), 1114.

[22] Yejin Kim, Jimeng Sun, Hwanjo Yu, and Xiaoqian Jiang. 2017. Federated Tensor Factorization for Computational Phenotyping. *KDD* (2017), 887–895.

[23] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv.org* (Dec. 2014), arXiv:1412.6980. arXiv:cs.LG/1412.6980

[24] Kolda, Tamara G and Bader, Brett W. 2009. Tensor Decompositions and Applications. *SIAM Rev.* 51, 3 (2009), 455–500.

[25] V Lampos, A C Miller, S Crossan, C Stefansen Scientific reports, and 2015. [n. d.]. Advances in nowcasting influenza-like illness rates using search query logs. *nature.com* ([n. d.]).

[26] ECK Li, B S Heran, JM Wright Cochrane Database of, and 2014. [n. d.]. Angiotensin converting enzyme (ACE) inhibitors versus angiotensin receptor blockers for primary hypertension. *cochranelibrary.com* ([n. d.]).

[27] FCCP Nathaniel Marchetti, DO. 2018. Acute Respiratory Distress Syndrome (ARDS) | CHEST Foundation. Retrieved January 2018 from https://foundation.chestnet.org/patient-education-resources/acute-respiratory-distress-syndrome-ards/

[28] Olson, Donald R, Konty, Kevin J, Paladini, Marc, Viboud, Cécile, and Simonsen, Lone. 2013. Reassessing Google Flu Trends Data for Detection of Seasonal and Pandemic Influenza - A Comparative Epidemiological Study at Three Geographic Scales. *PLoS Computational Biology* 9 (2013), e1003256–.

[29] Evangelos E Papalexakis. 2016. Automatic Unsupervised Tensor Mining with Quality Assessment. In *Proceedings of the 2016 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 711–719.

[30] Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. 2015. ParCube - Sparse Parallelizable CANDECOMP-PARAFAC Tensor Decomposition. *TKDD* 10, 1 (2015), 1–25.

[31] Neal Parikh, Stephen Boyd, et al. 2014. Proximal algorithms. *Foundations and Trends® in Optimization* 1, 3 (2014), 127–239.

[32] Philip M Polgreen, Yiling Chen, David M Pennock, and Forrest D Nelson. 2008. Using Internet Searches for Influenza Surveillance. *Clinical Infectious Diseases* 47, 11 (2008), 1443–1448.

[33] R L Richesson, W E Hammond, M Nahm Journal of the, and 2013. [n. d.]. Electronic health records based phenotyping in next-generation clinical trials: a perspective from the NIH Health Care Systems Collaboratory. *academic.oup.com* ([n. d.]).

[34] Sidiropoulos, Nicholas D, De Lathauwer, Lieven, Fu, Xiao, Huang, Kejun, Papalexakis, Evangelos E, and Faloutsos, Christos. 2016. Tensor Decomposition for Signal Processing and Machine Learning. *CoRR* stat.ML, 13 (2016), 3551–3582.

[35] David M Spiegel, Beverly Farmer, Gerard Smits, and Michel Chonchol. 2007. Magnesium Carbonate Is an Effective Phosphate Binder for Chronic Hemodialysis Patients: A Pilot Study. *Journal of Renal Nutrition* 17, 6 (2007), 416–422.

[36] Yichen Wang, Robert Chen, Joydeep Ghosh, Joshua C Denny, Abel N Kho, You Chen, Bradley A Malin, and Jimeng Sun. 2015. Rubik - Knowledge Guided Tensor Factorization and Completion for Health Data Analytics. *KDD* (2015), 1265–1274.

[37] Gavin WK Wong, Heidi N Boyda, and James M Wright. 2016. Blood pressure lowering efficacy of beta-1 selective beta blockers for primary hypertension. *Cochrane Database of Systematic Reviews* (2016).

[38] Pranjul Yadav, Michael Steinbach, Vipin Kumar, and György J Simon. 2018. Mining Electronic Health Records (EHRs) - A Survey. *ACM Comput. Surv.* (2018).

[39] Shihao Yang, Mauricio Santillana, John S Brownstein, Josh Gray, Stewart Richardson, and S C Kou. 2017. Using electronic health records and Internet search information for accurate influenza forecasting. *BMC Infectious Diseases* 17, 1 (May 2017), 89.

[40] M Zaharia, R S Xin, P Wendell, T Das Communications of the, and 2016. [n. d.]. Apache spark: a unified engine for big data processing. *dl.acm.org* ([n. d.]).

[41] Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *J. R. Stat. Soc. Series B Stat. Methodol.* 67, 2 (2005), 301–320.

Debiasing Vandalism Detection Models at Wikidata

Stefan Heindorf
Paderborn University
heindorf@uni-paderborn.de

Yan Scholten
Paderborn University
yascho@mail.uni-paderborn.de

Gregor Engels
Paderborn University
engels@uni-paderborn.de

Martin Potthast
Leipzig University
martin.potthast@uni-leipzig.de

ABSTRACT

Crowdsourced knowledge bases like Wikidata suffer from low-quality edits and vandalism, employing machine learning-based approaches to detect both kinds of damage. We reveal that state-of-the-art detection approaches discriminate anonymous and new users: benign edits from these users receive much higher vandalism scores than benign edits from older ones, causing newcomers to abandon the project prematurely. We address this problem for the first time by analyzing and measuring the sources of bias, and by developing a new vandalism detection model that avoids them. Our model FAIR-S reduces the bias ratio of the state-of-the-art vandalism detector WDVD from 310.7 to only 11.9 while maintaining high predictive performance at 0.963 ROC_{AUC} and 0.316 PR_{AUC}.

ACM Reference Format:
Stefan Heindorf, Yan Scholten, Gregor Engels, and Martin Potthast. 2019. Debiasing Vandalism Detection Models at Wikidata. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313507

1 INTRODUCTION

Knowledge bases play an important role in modern information systems. For instance, web search engines use them to enrich search results, conversational agents to answer factual questions, and fake news detectors for fact checking. Collecting knowledge at scale still heavily relies on crowdsourcing: Google acquired the open Freebase project to bootstrap its proprietary "Knowledge Graph" until Freebase was shut down and succeeded by Wikidata, the free knowledge base of Wikimedia. Other prominent open knowledge bases like Yago and DBpedia also depend on crowdsourcing by extracting knowledge from Wikipedia. As crowdsourcing knowledge has a long history, so does the fight against damage caused by vandals and other users, which may propagate to information systems using the knowledge base, potentially reaching a wide audience.

From its humble beginnings with manual review and rule-based detection bots, damage control at Wikipedia has grown into an intricate sociotechnical system, where man and machine work together to review edits and to maintain the integrity of its articles. While Wikipedia's damage control system grew alongside the encyclopedia for more than a decade, Wikidata gained momentum at a much faster pace: Transferring Wikipedia's system to Wikidata without a second thought (see Figure 1 for an illustration), adapting its procedures and the machine learning technologies employed [27, 49], has given rise to discrimination.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313507

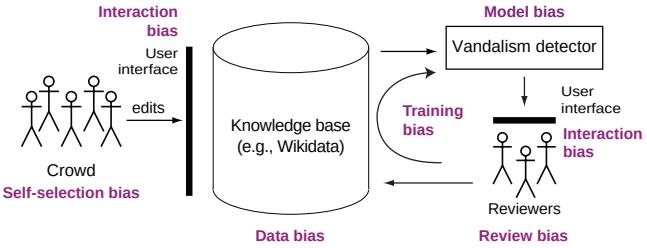


Figure 1: Wikidata's damage control system along potential sources of bias, which may accumulate in a vicious cycle.

In this paper, we reveal for the first time that state-of-the-art vandalism detectors employed at Wikidata are heavily biased against certain groups of contributors. For example, *benign* edits of new users receive vandalism scores over 300 times higher than *benign* edits of older user accounts. Such a widespread discrimination of certain user groups (especially that of anonymous editors) undermines the founding principles on which Wikimedia's projects are built:[1] maintaining a neutral point of view, the ability of anyone to edit articles, and the creation of a welcoming environment. The discrimination of anonymous users by registered users has long been recognized and the problem has been tackled through continuous community outreach.[2] But when discrimination gets encoded into automatic decision-making at Wikimedia, this aggravates the problem. For example, it has been previously found that new contributors whose edits are automatically reverted are much more likely to withdraw from the project [21, 22, 50].

Besides raising awareness, we carefully analyze different sources of bias in Wikidata's damage control system. Based on these insights, we develop two new machine learning models and demonstrate that bias can be significantly reduced compared to the state-of-the-art. Our model FAIR-E uses graph embeddings to check the content's correctness without relying on biased user features. Our model FAIR-S systematically selects the best-performing hand-engineered features under the constraint that no user features are used. Furthermore, we experiment with different transformations of the state-of-the-art vandalism detector WDVD: post-processing scores, reweighting training samples, and combining approaches via ensembles. We evaluate our approaches on a subset of the standardized, large-scale Wikidata Vandalism Detection Corpus 2016 [27], comparing our results to others from the literature.

In what follows, after discussing related work in Section 2, we analyze in Section 3 the sources of bias for Wikidata's damage control system. Section 4 introduces our new detection models designed to mitigate biases. Sections 5 and 6 detail the evaluation data and our comparative evaluation, respectively, and Section 7 discusses limitations and practical implications.

[1] https://meta.wikimedia.org/wiki/Founding_principles
[2] https://en.wikipedia.org/wiki/Wikipedia:IPs_are_human_too

2 RELATED WORK

Despite the long-standing insight that machine learning is prone to incur bias, the body of work addressing this problem is surprisingly small. Only recently Baeza-Yates [2] compiled an overview of biases found on the web that may induce discrimination, including an analysis of Wikipedia's editor elite and its gender gap. Romei and Ruggieri [48], Zliobaite [67], and Pedreschi et al. [45] collect measures to quantify bias in machine learning models which we employ in our analysis. Otherwise, only a handful of papers explicitly attempt to mitigate biases in machine learning applications: Zhao et al. [65] and Bolukbasi et al. [5] tackle gender bias, Wang et al. [57], Wilkie and Azzopardi [59] and Yang and Stoyanovich [61] tackle biases in information retrieval systems, Torralba and Efros [53] investigate bias in image classification, and Dixon et al. [13] propose methods to measure and mitigate bias in text classification. Algorithmic attempts at bias mitigation include naive Bayes [7], SVMs [62], decision trees [30], and random forests [47]. The trade-off between accuracy and fairness is explored by Berk et al. [4], Kleinberg et al. [33], Corbett-Davies et al. [10], and Chouldechova [8], while performance measures for imbalanced datasets such as PR_{AUC} and ROC_{AUC} are not considered. Berk et al. [4], Hardt et al. [23], and Dwork et al. [16] discuss different notions of fairness including equality of opportunity and statistical parity. More closely related to our work, Halfaker et al. [21, 22] and Schneider et al. [50] find that newcomer retention at Wikimedia projects is severely affected by overzealous reversion of their edits. Passing part of the blame to automatic vandalism detectors, no remedies are proposed.

Vandalism at Wikipedia is defined as "editing deliberately intended to obstruct or defeat the project's purpose."[3] Though this definition excludes damage caused unintentionally, the task of a vandalism detector, human or other, is to identify both kinds of damage so they can be dealt with according to severity. The literature and practitioners often speak of vandalism detection while actually operationalizing damage detection in general [31]; current vandalism detectors are quite incapable of discerning user intent, which often even challenges human reviewers. Vandalism detection was first proposed for Wikipedia, employing text mining and user reputation features to detect damaging edits to articles [46]. More than a dozen approaches have been proposed since (e.g., [1, 29, 54, 56]), giving rise to a rich set of features, many of which have been transferred into the models currently at work. By comparison, vandalism detection models for Wikidata are still in their infancy [25, 27, 43, 49, 52], as evidenced by the facts that their feature sets are almost entirely the same as those employed for Wikipedia, and that hardly any feature quantifies Wikidata's actual content, i.e., the knowledge encoded as subject-predicate-object triples. Our approach omits the biased user reputation features and introduces novel graph embeddings, drawing from, and advancing related work on fact checking and link prediction.

Fact checking pertains to checking the correctness of a fact and can be divided into approaches using internal and external knowledge. Regarding the former, Ciampaglia et al. [9] compute the shortest path between a fact's subject and its object, considering different weighted (transitive) predicates. Shi and Weninger [51] determine paths between subject and object that help to distinguish correct from incorrect predicates between them, employing a one-hot encoding and logistic regression. We experimented with path features, too, but found a simpler graph embedding to be superior for debiasing. Nishioka and Scherp [43] analyze the evolution of knowledge graphs to verify changes, employing basic features like the age of entities and predicates and the in- and out-degree of entities. Approaches using external knowledge include that of Wu et al. [60], who check the correctness of facts from news articles by generating SQL queries to find contradictory information in a relational database. Lehmann et al. [37] check the correctness of a given subject-predicate-object triple via web search, taking into account the trustworthiness of websites found.

Link prediction is the task of predicting missing predicates between pairs of subjects and objects. Nickel et al. [41] survey corresponding approaches and distinguish explicit and latent features. The former category includes path rankings used to complete existing knowledge bases [36, 40, 51] and validation knowledge extracted from web sources [14, 15, 28]. Gardner and Mitchell [20] simplified and sped up path ranking. We experimented with their best features, but found a simpler graph embedding to be superior for debiasing. Other approaches do not try to complete a knowledge base but classify the local completeness of entities within [11, 18]. Link prediction approaches with latent features are based on matrix factorization [42], neural embeddings [14], and translational embeddings [6, 38, 55, 58]. Special kinds of link prediction include type prediction of the type of an entity [19, 39, 44] and predicting obligatory attributes of entities of a given type [35].

Altogether, fact checking and link prediction are complementary in that the former's goal is to identify incorrect knowledge, whereas the latter's goal is to introduce correct knowledge that is missing. Both are often evaluated on artificial data, whereas we apply them to vandalism detection at scale on real-world data from Wikidata. Moreover, both tasks typically assume a static knowledge base, while vandalism detection presumes a constant stream of edits. We adapt the methods borrowed from both tasks accordingly.

3 BIAS ANALYSIS

Wiktionary defines 'discrimination' as "*treatment* of an individual or group to their disadvantage" and 'bias' as "*inclination* towards something; predisposition, partiality, prejudice [...]." If a given system acts discriminatorily, we call it biased, and we ask which of its components cause its bias and how it can be controlled to render its behavior non-discriminatory. In this section, we carry out the first bias analysis of Wikidata's damage control system.

Several potential sources of bias can be identified (see Figure 1): *model bias* refers to the vandalism detection model used, and may result from its selection of features and its learning algorithm. The user interface of monitoring tools may introduce *interaction bias* by directing reviewer attention to high-scoring edits first, enclosing them in a "filter bubble" and reinforcing inherent *reviewer bias*, i.e., the prejudices that humans sometimes hold. Analyzing the reviewers' decisions, models are adjusted directly, by creating rules and features, or indirectly, by using their decisions as labeled training data, causing *training bias* and *data bias*. Furthermore, biased reviewer decisions may lead to *self-selection bias* among Wikidata's volunteers, ousting non-conformists.

[3] https://en.wikipedia.org/wiki/Wikipedia:Vandalism

The discriminatory nature of *Wikipedia's* damage control system has been previously shown [21, 22, 50]: the rise of (semi-)automatic reviewing tools caused more newcomer contributions to be considered damaging, severely affecting retention. Although policies have been adjusted to prevent such discrimination, the vandalism detection models have not been redesigned. Our analysis shows that, at least for *Wikidata*, this is insufficient. In what follows, we go beyond previous analyses by shedding light on the main sources of bias, the detection models and the human reviewers, and by taking into account anonymous users, the antecedents of newcomers.

3.1 Measuring Bias

Every edit of an item at Wikidata results in a new revision i of that item, and the task of a vandalism detector c is to compute a vandalism score for each new revision as soon as it arrives, so that $c(i) \approx Pr(i = \text{vandalism} \mid x_i)$, where x_i is i's feature vector. The scores are then used for two modes of operation. In fully automatic mode, the revisions exceeding a score threshold are automatically reverted without any human intervention. In semi-automatic mode, revisions are ranked by vandalism score and manually reviewed.

To measure the bias of a detector producing continuous scores, roughly following Kleinberg et al. [33] and Zemel et al. [63], we compare the average scores of *benign* edits by two disjoint user groups. The more the difference deviates from 0 or the ratio deviates from 1, the more biased the classifier is. By convention, we call one group "protected" (i.e., to be protected from discrimination). Let I denote the set of all revisions, then given the ground truth whether a revision i is vandalism or benign, we get the following contingency table, where $A, \ldots, H$ denote the corresponding subsets of I, from which we derive the difference (Diff.) and the Ratio between the protected group and the remainder as bias measures:

		Truth benign	Truth vand.	∪
Protected	yes	A	B	C
	no	D	E	F
∪		G	H	I

$$\text{Diff.} = \frac{1}{|A|}\sum_{i\in A} c(i) - \frac{1}{|D|}\sum_{i\in D} c(i)$$

$$\text{Ratio} = \frac{1}{|A|}\sum_{i\in A} c(i) \;/\; \frac{1}{|D|}\sum_{i\in D} c(i)$$

Table 1a shows examples, comparing the vandalism score of the Wikidata Vandalism Detector (WDVD) for the creation of a given subject-predicate-object triple by a registered user with that of creating the same triple anonymously. Regarding the first example, at a score difference of 0.0907, an anonymous user receives a score that is 900.2 times higher than that of a registered user.

To measure the bias of semi-automatic detection, which produces binary scores, we utilize the odds ratio, which is invariant to changes in class distribution in the dataset. Let $p_1 = \sum_{i\in A} c(i)/|A|$ be the proportion of benign revisions from protected users that are considered vandalism and let $p_2 = \sum_{i\in D} c(i)/|D|$ be the proportion of benign revisions from other users that are considered vandalism:

$$\text{Odds ratio} = \frac{p_1/(1-p_1)}{p_2/(1-p_2)}.$$

Further measures have been proposed, including the closely related risk difference and risk ratio [7, 48, 67], but also measures based on individuals instead of groups [16], the latter demanding that similar edits receive similar scores. For lack of a task-specific similarity measure, we resort to the aforementioned measures.

3.2 Biases of Vandalism Models

There are two kinds of vandalism detectors for Wikidata: (1) the rule-based Wikidata Abuse Filter (FILTER) [27], which tags revisions as per user-defined rules,[1] and (2) machine learning-based detectors. The latter include the state-of-the-art approach WDVD [27] and variants thereof developed during the WSDM Cup 2017 [24], the "Objective Revision Evaluation Service" (ORES) [49] deployed at Wikimedia, and our new approach FAIR proposed in this paper.[2] The WDVD approach outperformed all other approaches in the recent WSDM Cup 2017 in terms of PR_{AUC} [25]. Table 1b shows their distribution of features across feature categories. Apparently, all previous approaches rely on contextual, and especially user-related features, such as account age and whether an edit was made by a registered or an anonymous user. Moreover, hardly any content features capture Wikidata's statements, but only textual portions of an item. This already hints at a high potential for bias.

Analyzing the vandalism scores of all approaches applied on the Wikidata Vandalism Corpus 2016 [25] reveals that all except FAIR exhibit significant biases. Table 1c exemplifies the average vandalism scores obtained for anonymous and registered users from which we compute the bias measures found in Table 1d (top row pair). Similarly, the bias against newcomers and country of origin is computed (remainder of the table). Being anonymous or a newbie raises one's average vandalism score of benign edits by a factor of up to 311 under WDVD, and by a factor of 40-133 under ORES and FILTER. With FAIR, we reduce the bias ratio to 11.9. There is comparably small bias against country of origin, and even a bias in favor of Japanese users, which can be explained by their low vandalism prevalence [31]. These values must be taken with a grain of salt, since geolocation is available only for anonymous users, who are a priori discriminated because of this fact. In what follows, we hence focus on mitigating the bias against anonymous users.

Upon close inspection, the many user-related features employed by WDVD, ORES, and FILTER turned out to be the causes for their biases. For example, the feature `isRegisteredUser`, that is employed by both WDVD and ORES, is a simple feature with high predictive performance. But it causes *benign* edits by anonymous users to have much higher scores than *benign* edits by registered users: 9.00% of edits by anonymous users constitute vandalism and only 0.03% of edits by registered users (in the training and validation set of WDVC-2016-Links). The feature is not able to discriminate *benign* and *vandalizing* edits based on the actual content of an edit, thus assigning roughly 300 times higher scores to *all* edits by anonymous users. Similarly, ORES includes the age of a user (`userAge`), and WDVD includes the numbers of revisions and items edited by a user (`userFrequency`, `cumUserUniqueItems`), as well as geolocation of IP addresses (e.g., `userCountry`, `userCity`). Regarding FILTER's rules, Table 1e lists the top-most rules fired which exploit user information: "new user changing sth."; "new user removing sth."; and "possible vandalism." Compared to the top ones not doing so, these rules act highly biased against anonymous users. Edits to which these rules apply receive an average vandalism score of 28.8%, which is a high number given the large class imbalance; the majority of wrong decisions affect anonymous users (69.1%), again

[1] We convert tags to continuous scores by computing the empirical vandalism probability of all past revisions with a given tag.
[2] For brevity, we report only on the best variant, FAIR-S, in this bias analysis.

Table 1: Bias analyis: (a) Examples for model bias. (b) Overview of features by model. (c) Average vandalism scores for anonymous and registered users.[4] (d) Bias measurements against protected users as score difference (top rows) and score ratio (bottom rows). (e) Rules of the Wikidata Abuse Filter (FILTER) and their biases.[5] (f) Biases by Wikidata reviewers. Among 1,100 benign edits, 145 were incorrectly reverted by human reviewers with disproportionately many affecting anonymous editors.

(a)

Revision	User	Score	Diff.	Ratio
⟨Guido Westerwelle, place of death, Cologne⟩				
313453592	Anonymous	0.0908	0.0907	900.2
313455460	Registered	0.0001		
⟨Alejandro Cuello, occupation, actor⟩				
325717121	Anonymous	0.2912	0.2763	19.6
318143388	Registered	0.0149		

(b)

Feature	WDVD	ORES	FILTER	FAIR-S
Content	27	5	0	4
Character	11	0	0	0
Word	9	3	0	0
Sentence	4	1	0	0
Statement	3	1	0	4
Context	20	8	1	10
User	10½	2	½	0
Item	2	2	0	6
Revision	7½	4	½	4

(c)

Users	WDVD	ORES	FILTER	FAIR-S
Anonymous	0.1215	0.1144	0.0978	0.0337
Registered	0.0004	0.0009	0.0014	0.0028

(d)

Protected	WDVD	ORES	FILTER	FAIR-S
Benign edits by *all* users				
Anonymous	0.121	0.114	0.096	0.031
	310.7	133.1	69.2	11.9
Newcomer (1h since 1st edit)	0.138	0.109	0.109	0.037
	172.7	72.4	63.7	13.5
Newcomer (1d since 1st edit)	0.101	0.085	0.085	0.026
	170.9	68.3	58.2	10.2
Newcomer (7d since 1st edit)	0.060	0.053	0.053	0.015
	118.8	48.5	40.7	6.1
Benign edits by *anonymous* users				
Origin USA	0.053	0.091	0.053	0.021
	1.5	1.9	1.6	1.6
Origin Mexico	0.243	0.132	0.109	0.059
	3.1	2.2	2.1	2.8
Origin Spain	0.167	0.097	0.085	0.049
	2.5	1.9	1.9	2.5
Origin Japan	-0.080	-0.053	-0.001	-0.028
	0.3	0.5	1.0	0.2

(e)

FILTER Rules	Performance (*all* edits) Total	Vand.	Prob.	Bias (*benign* edits) Total	Anon.	Prob.
w/ user information	21,225	6,113	28.8%	15,112	10,445	69.1%
New user changing sth.	14,452	4,313	29.8%	10,139	6,525	64.4%
New user removing sth.	5,609	1,031	18.4%	4,578	3,560	77.8%
Possible vandalism	1,164	769	66.1%	395	360	91.1%
w/o user information	12,612	849	6.7%	11,763	739	6.3%
Self-referencing	4,030	141	3.5%	3,889	236	6.1%
Removal of gender	3,408	83	2.4%	3,325	95	2.9%
Unexpected gender	3,381	612	18.1%	2,769	292	10.5%
Miscellaneous	1,793	13	0.7%	1,780	116	6.5%
No Rules	6,970,017	15,688	0.2%	6,954,329	140,580	2.0%

(f)

Users	Manually reviewed benign edits (n=1,103)			
	Observed behavior		Expected behavior	
	Reverted (incorrect)	Non-reverted (correct)	Reverted (incorrect)	Non-reverted (correct)
Anonymous	103	23	16.6	109.4
Registered	42	935	128.4	848.6

a large number compared to the small percentage of overall edits by anonymous users. FILTER rules can only be created by Wikidata's (at the time of writing) about 60 administrators,[3] who are the most active users and who carry out lots of other maintenance tasks, too.

For the new approach FAIR-S, we omit all user features, thus significantly reducing its bias. The remaining bias can be explained by unsuspicious features that are (slightly) correlated with the omitted feature `isRegisteredUser`. This effect is referred to as *indirect discrimination* [45] or *redlining* [7]. In general, it is undesirable to make all groups have exactly the same average scores as small differences might be justified by hidden confounding variables and it would be an overreaction often called *affirmative action* [48, 66].

3.3 Biases of Wikidata Reviewers

Another potential source of bias might be the human reviewers engaged in Wikidata's semi-automatic damage control system. If a reviewer encounters a poor edit, it is reverted, and otherwise typically nothing happens. To investigate the degree to which revert decisions may be biased, we used a manually annotated dataset, compiled as part of the Wikidata Vandalism Corpus 2015 (WDVC-2015) [26]. It consists of two random samples of 1,000 edits each that were rollback reverted by Wikidata reviewers and that were not reverted, respectively. These edits were carefully annotated with respect to whether they constitute vandalism, but unlike Wikidata's reviewers, the annotator was given no knowledge about the registration status (anonymous or registered) of the editor.

Among the 2,000 edits, 1,103 are *benign* edits, 145 of which were *incorrectly* reverted. Table 1f breaks down the observed behavior of Wikidata reviewers as a contingency table of registration status over revert decision correctness. Reverting a benign edit is an incorrect decision, whereas not reverting it is correct. From the marginals of this matrix (not shown), we obtain the expected behavior under the assumption of independence. We find that revert decisions by Wikidata reviewers are heavily biased against anonymous editors, since they are disproportionately often affected by incorrectly reverted edits: 103 edits by anonymous users are *incorrectly* reverted although we would expect only 16.6 edits considering the relatively small number of 126 edits by anonymous editors. Fisher's test shows that the bias against anonymous users is statistically highly significant ($p < 0.0001$). Figure 2 shows how the reviewer bias developed over time. Choosing time intervals so that an approximately equal amount of the 1,103 benign edits fell into each interval, the odds ratio severely increased over time, suggesting that bias against anonymous users among reviewers kept growing.

Only Wikidata's administrators and a few other powerful users with the so-called rollback rights can perform rollback reverts. Even when accounting for human fallibility, it appears that, on this sample of edits, anonymous editors were discriminated. We do not suspect any malicious intent of Wikidata reviewers. Rather, the fact that many true vandalism edits originate from anonymous editors may have led them to ingest some form of prejudice whenever an anonymous edit looks odd while skipping edits by registered editors without closer inspection. However, even if more vandalism originates from anonymous editors than from registered ones, this is not a sufficient justification to treat benign anonymous editors unfairly. After all, a revert decision should be based on content alone, and not on who the content comes from.

[3] https://www.wikidata.org/wiki/Special:ListGroupRights

[4] Tables c, and d were computed on the test set of the vandalism corpus WDVC-2016-Links. Vandalism scores of the models FAIR-S, WDVD, ORES, FILTER were obtained as described in Sections 4 and 6 and calibrated to approximate vandalism probabilities.

[5] Computed on the whole WDVC-2016-Links dataset.

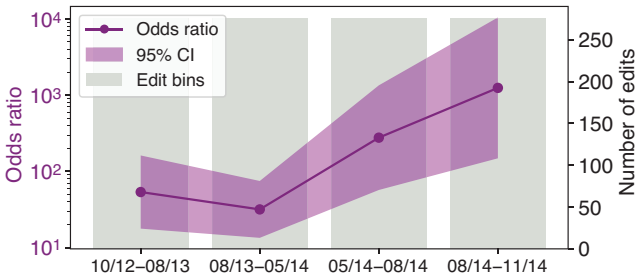


Figure 2: Reviewer bias against anonymous users as odds ratio over time, averaged over equisized edit bins.

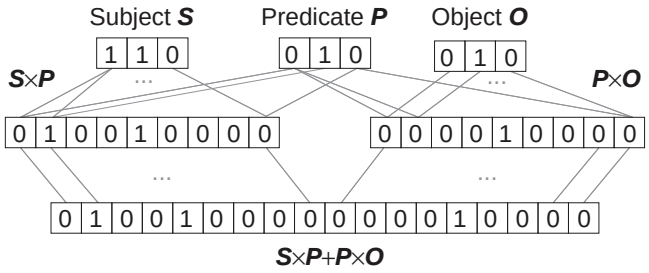


Figure 3: Example of an embedding of a subject-predicate-object triple in $n := 3$-dimensional predicate space (top), the outer products $S \times P$ and $P \times O$ (middle), and the final $2n^2 = 18$-dimensional embedding (bottom).

4 FAIR—UNBIASED VANDALISM DETECTION

Our bias analysis indicates that the main source of bias in state-of-the-art vandalism detectors are user-related features. That, together with the lack of features to characterize the actual content of Wikidata, creates a situation where bias and discrimination thrives. This section introduces FAIR, a new vandalism detector for Wikidata. We tackle the problem (1) by devising new content features based on explicit graph embeddings that protect Wikidata's primary asset, its subject-predicate-object triples between entities (FAIR-E), and, (2) by careful feature selection from the existing vandalism detectors with an eye on bias (FAIR-S).

4.1 FAIR-E: Graph Embeddings for Wikidata

Unlike for Wikipedia, where the full power of text mining can be unleashed to detect vandalism in its articles, the content of knowledge bases is much more difficult to be represented, which may explain the lack of corresponding features in Wikidata's vandalism detectors. We propose a novel graph embedding FAIR-E that results in a particularly low bias. To the best of our knowledge, this kind of graph embedding has not been used before, neither for Wikidata vandalism detection, nor elsewhere.

Editing Wikidata means editing subject-predicate-object triples. The user interface of Wikidata enforces that every edit affects exactly one triple,[6] so that representing an edit boils down to representing the edited triple. We encode the subject, predicate, and object of a triple in predicate space and capture all pairwise interactions between them (see Figure 3 for an illustration). For the subject, we encode all *outgoing* predicates as a binary vector S. The predicate is directly encoded as one-hot encoding P. For the object, we encode all *incoming* predicates as a binary vector O. The whole triple is then encoded as concatenations of pairwise combinations: $S \times P + P \times O$, where the +-operator denotes the concatenation of vectors and $\times$ the outer product, i.e., all combinations of elements. Let n be the number of predicates, then a triple vector has $2n^2$ dimensions. In Wikidata, there are about $n = 6,000$ predicates. To avoid overfitting, we restrict the set of predicates: We compute all predicates that were present in the item graph at the end of the training set and we set all other predicates in our embedding vectors to zero, thus effectively removing predicates to attributes such as strings and numbers and removing predicates that have been deleted earlier. Moreover, we restrict the representation to $n := 100$ predicates for S, P, and O, respectively. Using a suitable training dataset, we compute the top 100 predicates according to frequency for S, P, and O independently, and then proceed to computing the combined representation, yielding a 20,000-dimensional vector. As machine learning algorithm, we employ logistic regression.

To give an example how the embedding works: Suppose we want to represent the newly created triple ⟨Alejandro Cuello (Q15924626), occupation (P106), actor (Q33999)⟩. The subject is a person with predicates "date of birth" (P569), "sex or gender" (P21), and "country of citizenship" (P27), the triple predicate is "occupation" (P106), and the object has incoming predicates "occupation" (P106), "field of work" (P101), and "position held" (P39). Our model learns that the triple predicate "occupation" goes well with the incoming object predicates "occupation", "field of work" and "position held." In contrast, the triple ⟨Steve Jobs (Q19837), instance of (P31), animal (Q729)⟩ is highly unusual. Adding another "instance of" (P31) relationship to a subject with subject predicates "country of citizenship" (P27) and "sex or gender" (P21) often points to vandalism; as does adding an "instance of" relationship to an object with incoming object predicate "parent taxon" (P171).

Variants. We experimented with multiple variants of our approach, including different values for n, different interactions between subject, predicate, and object (see Section 6), and predicates to attributes. We experimented with taking the *outgoing* predicates of the object instead of the *incoming* predicates, too. However, the above encoding outperformed all other variants on the validation set in terms of predictive performance. Varying the strength of L2 regularization had little effect on predictive performance but had a large effect on bias, so we disabled regularization to minimize bias (setting scikit-learn's parameter $C = 10000$). We also experimented with the path ranking algorithm [14, 20], but the resulting model did not outperform the above model.

Limitations. If a subject-predicate-object triple uses a predicate not among the ones selected, its embedding is the zero vector, which was the case for about 15% of triples in our dataset. Our logistic regression classifier assigns the same, small vandalism probability to all these cases (determined by the intercept of logistic regression). For triples updated in a revision, we only consider the new version of the triple. Our embedding does not distinguish additions and removals of triples and we leave it for future work to incorporate such a distinction in a way that improves performance. As the set and distribution of predicates changes over time, retraining the classifier may be required from time to time.

[6] Wikidata's API allows for editing multiple triples at once, which is rather the exception. In that case, we use the main triple from Wikidata's automatically generated comment.

4.2 FAIR-S: Selecting Unbiased Features

As an alternative approach to debiasing vandalism detectors, we systematically evaluated candidate feature subsets for subject-predicate-object triples from the union of all features developed for previous detectors and call our model FAIR-S: We used the features of WDVD [27], the features of ORES [25, 49], and a few new features, including the ones from FAIR-E, while intentionally omitting user-related features and features not targeting subject-predicate-object triples between entities. Features were added to the set until predictive performance on our validation set did not improve anymore. Table 2 shows the resulting feature set, which constitutes a local optimum in the space of possible feature subsets, since removing any of the features or adding further features from our candidate set decreases performance in terms of ROC_{AUC}. Seven features from Heindorf et al. [27], three features from Sarabadani et al. [49], and four new features were selected. We use a random forest with 32 trees and a maximal depth of 16 for our experiments.[7] In a pilot study, we experimented with other algorithms including logistic regression, neural networks, and gradient boosted decision trees, carefully tuning their hyperparameters, yet, corroborating previous findings, random forests outperformed them all in this setting [27, 49]. Below, the features are described in more detail.

Subject. We characterize a subject by how many different users have edited it (`subjectLogCumUniqueUsers`), how many edits have been performed on it (`subjectLogFrequency`), how many labels and aliases it has (`subjectNumberOfLabels`, `subjectNumberOfAliases`), the number of words in its English label (`subjectLabelWordLength`), and how often the predicate has been edited for this subject (`subjectPredicateCumFrequency`). These features signal how popular a subject is (`subjectLogCumUniqueUsers`), how large a subject is (`subjectNumberOfLabels`), and how complex a subject is (`subjectLabelWordLength`). The number of words serves as a proxy for the complexity of an item. Items with one-word labels might be about organizations or places, items with two words are typically persons, and items with many words are complex topics. Other forms of encoding from our candidate features did not yield any improvement. In general, we found that subject features are prone to overfitting, since there are about 2.5 million different subjects and most subjects have never been vandalized.

Predicate. We represent each predicate by the number of times it appears in the training set (`predicateFrequency`). Other forms of encoding from our candidate feature set did not yield improvements.

Object. In contrast to subject features, object features are less prone to overfitting since objects are reused many times. We capture the popularity of an object by the number of revisions it appears in (`objectFrequency`) and the number of times an object has been edited in combination with a given predicate (`objectPredicateCumFrequency`). Moreover, one of the features of FAIR-E was included: We compute the set of all incoming predicates of an object in the knowledge graph and take this set as a feature (`objectPredicateEmbedFrequency`), the idea being that the incoming predicates of an object capture in what context and for what purposes it should be used. To avoid overfitting, we restrict this set to the top 100 most frequent predicates on the training set.

[7] We use a fixed seed for feature selection. Due to some low-performing and redundant features, the feature set is not stable for different seeds. But even slightly different feature sets yield similar predictive performance and bias values.

Table 2: Features of our vandalism detector FAIR-S and their performance scores and biases on our test dataset, computed with a random forest with 32 trees and maximal depth 16.

Feature Group / Feature	Reference	Performance ROC_{AUC}	Performance PR_{AUC}	Bias Diff	Bias Ratio
Subject	—	0.907	0.071	0.01199	4.05
`subjectLogCumUniqueUsers`	[27]	0.901	0.052	0.01005	3.51
`subjectLogFrequency`	[27]	0.880	0.042	0.00795	2.95
`subjectNumberOfLabels`	[49]	0.859	0.031	0.00321	1.76
`subjectNumberOfAliases`	[49]	0.761	0.026	0.00198	1.47
`subjectLabelWordLength`	—	0.721	0.008	0.00165	1.38
`subjectPredicateCumFrequency`	—	0.687	0.008	0.00096	1.22
Predicate	—	0.729	0.011	0.00196	1.46
`predicateFrequency`	[27]	0.729	0.011	0.00196	1.46
Object	—	0.729	0.026	0.00234	1.55
`objectPredicateEmbedFrequency`	—	0.682	0.013	0.00188	1.44
`objectFrequency`	[27]	0.650	0.010	0.00107	1.25
`objectPredicateCumFrequency`	—	0.613	0.007	0.00014	1.03
Edit Features	—	0.889	0.062	0.00926	3.32
`editProportionOfTriplesAdded`	[49]	0.866	0.022	0.00418	2.00
`editSubactionFrequency`	[27]	0.851	0.020	0.00381	1.91
`editPrevActionFrequency`	[27]	0.629	0.024	0.00297	1.70
`editActionFrequency`	[27]	0.575	0.005	0.00052	1.12

Edit. We characterize the change of an edit with four features: the edit operation, such as create, update, or remove (`editActionFrequency`, `editSubactionFrequency`), the previous action performed on the same item (`editPrevActionFrequency`), and the size of the edit operation relative to the current size of the item (`editProportionOfTriplesAdded`).

Variants. Both for the subject and the object, we experimented with using their *super types* according to Wikidata's *instance of* hierarchy as a feature. Neither of these features improved predictive performance nor bias. For the object, similar information is already captured with our predicate embedding (e.g., in `objectPredicateEmbedFrequency`). Also, a bag-of-words model of the subject's and the object's textual labels and descriptions did not help.

5 EVALUATION DATA

This section describes the Wikidata datasets used in our evaluation, how they have been pre-processed, split into training, validation, and test sets, as well as their characteristics. Moreover, we investigate the proportion of model bias due to data bias.

5.1 Wikidata Revision History and Graph

We derive our dataset from the Wikidata Vandalism Corpus 2016 (WDVC-2016) [25]. This dataset ranges from October 2012 to June 2016 and contains all human edits along with labels whether the edit is considered vandalism. The labels were automatically obtained by analyzing the use of the rollback tool of Wikidata, which is explicitly meant to revert vandalism [26, 27]. A manual analysis revealed that 86% of reverted revisions are indeed vandalism and only about 1% of the non-reverted revisions. We derive a subset of this dataset, called WDVC-2016-Links, that contains all edits pertaining to the knowledge *graph*, i.e., the actual content of

Table 3: Evaluation data: (a) Derivation of the new dataset WDVC-2016-Links from WDVC-2016. (b) Datasets for training, validation, and test in terms of vandalism triples, total triples, subjects, predicates, and objects. (c) The vandalism corpus WDVC-2016-Links broken down by domain. (d) Estimation of the true bias without noisy labels. We perform stratified sampling on the test dataset and manually annotate the samples to estimate the true bias. (e) Wikidata vandalism corpus WDVC-2016-Links over time by domain for all edits, and, (f) for vandalism edits only.

(a)

Filtering steps	Edits	Prop.
WDVC-2016	82,679,918	100 %
w/o semi-automatic editing tools	35,462,023	43 %
w/o labels, descriptions, aliases	26,959,949	33 %
w/o sitelinks	17,371,199	21 %
w/o qualifiers, references	15,883,276	19 %
w/o attributes	11,631,335	14 %
w/o item creation, merges, misc	7,002,290	8 %
WDVC-2016-Links	7,002,290	8 %

(b)

Dataset	Vand.	Edits	Subjects	Pred.	Objects
Training	16,608	5,890,968	2,187,993	747	790,960
Validation	2,604	550,798	201,102	610	131,666
Test	2,429	560,524	212,902	603	131,479

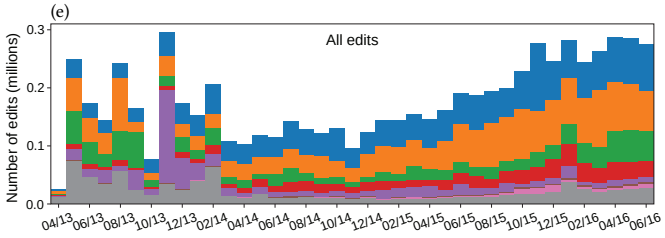
(c)

Domain	Performance (*all* edits)			Bias (*benign* edits)		
	Total	Vand.	Prob.	Total	Anon.	Prob.
Generic	1,863,273	5,755	0.31%	1,857,518	36,174	1.95%
Person	1,905,257	10,539	0.55%	1,894,718	46,671	2.46%
Organization	67,890	312	0.46%	67,578	2,601	3.85%
Events	96,115	68	0.07%	96,047	639	0.67%
Works	618,464	1,598	0.26%	616,866	32,252	5.23%
Terms	680,904	437	0.06%	680,467	2,279	0.33%
Place	951,589	1,628	0.17%	949,961	16,857	1.77%
Others	818,798	1,304	0.16%	817,494	13,841	1.69%

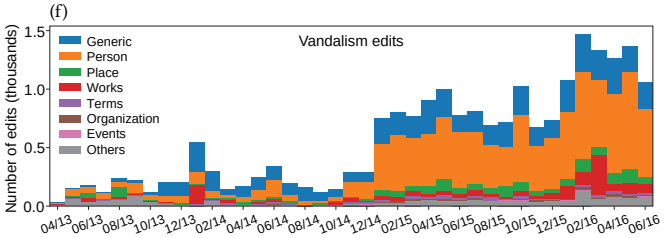
(d)

Corpus			Gold			WDVD	True Bias
User	Truth	n	Truth	n	Weight	∅ Score	wt. sum
Anon.	vand.	2,061	benign	19	1.98%	0.279443	0.005526
			vand.	231	n/a	n/a	n/a
	benign	10,270	benign	189	98.02%	0.079500	0.077928
			vand.	61	n/a	n/a	n/a
Reg.	vand.	368	benign	29	0.01%	0.032105	0.000003
			vand.	221	n/a	n/a	n/a
	benign	547,825	benign	239	99.99%	0.000579	0.000579
			vand.	11	n/a	n/a	n/a
Sum		560,524		1,000		Diff.	0.083
						Ratio	143.4

(e)



(f)



the knowledge base. Table 3a details the successive filtering steps we applied. We omit edit operations by semi-automatic editing tools like Wikidata Game because they perform little vandalism and we believe tailored quality checks should rather be directly built into them. Moreover, we remove edits affecting labels, descriptions, aliases, sitelinks, qualifiers, references, attributes, and special operations for item creation and item merging, yielding a dataset consisting of edits affecting links between entities.

While the Wikidata vandalism corpus contains the data in incremental form, i.e., edit by edit, for our novel content-based features, we need to represent the data as a graph. For these features, we base our computation on the static Wikidata graph from February 29, 2016, i.e., the graph capturing the data ahead of the time interval covered by the validation set.[8]

5.2 Datasets for Training, Validation, and Test

Following previous work, we split the dataset by time to enable classification of edits as soon as they arrive without exploiting information "from the future." Table 3b shows the data splits along key statistics on triples. Overall, the dataset contains 21,641 vandalism cases among 7 million edits; a class imbalance of 0.3%. Our training set ranges from April 18, 2013, to February 29, 2016, consisting of 6 million edits. For efficiency, we derive triples from Wikidata's automatically generated edit comments, which were not available before April 2013. The two-month periods for validation and testing range from March to April and May to June, respectively, each comprising more than 550,000 edits. The large number of subjects renders subject feature prone to overfitting.

[8] https://archive.org/download/wikidata-json-20160229

5.3 Corpus Analysis

Table 3c breaks down the vandalism corpus by domain. We categorize edits by domain according to Wikidata's classification system.[9] In both absolute and relative terms, the most vandalism edits occur on persons such as Barack Obama, Christiano Ronaldo, and Justin Bieber, potentially causing high vandalism scores for all edits in this category. At the same time, among all categories, persons have the highest number of *benign* edits by anonymous users which still should receive low vandalism scores. This illustrates the inherent difficulty to achieve high predictive performance and low bias at the same time. An unbiased vandalism model must discriminate vandalism from non-vandalism without discriminating certain editor groups. Biases are still relatively small in this case, compared to the abuse filter in Table 1e.

The plots (e) and (f) in Table 3 overview our corpus both in terms of time and domains. The number of edits per month increases, as does the number of vandalism cases. The drop of manual edits around March 2014 can be explained by emerging automatic tools that automate routine data import and maintenance tasks, thus temporarily reducing manual edit operations. The increasing vandalism around January 2015 coincides with the redesign of Wikidata's user interface, driving vandals from the header area of labels, descriptions, and aliases to the body area containing links and statements. The person domain grows fastest, both in terms of benign and vandalism edits. Overall, the distribution of vandalism across domains remains relatively stable over time, with the exception of two outliers: an increase of vandalism affecting the works domain in January 2013 and March 2016.

[9] https://www.wikidata.org/w/index.php?title=Wikidata:List_of_properties&oldid=552806031

5.4 Estimating Model Bias Without Data Bias

To investigate the effect of the noisy automatic vandalism labels on bias, we carry out a manual analysis: Drawing a stratified sample of 1,000 edits with respect to registration status (anonymous vs. registered) and automatic ground truth (vandalism vs. benign), we obtain gold labels by manual review of 250 edits per stratum, enabling an estimation of the true bias. The reviewer was kept unaware to which stratum a given edit belonged. Table 3d summarizes our findings and shows the average vandalism scores of WDVD for the data subsets we found to be *truly benign* and that form the basis for the bias computation. The WDVD scores were obtained by calibrating its original scores with isotonic regression on the large-scale corpus ground truth. As a result, WDVD's estimated bias without noisy labels is a score difference of 0.083 and a score ratio of 143.4. This is about half the bias measured on the automatically annotated corpus (0.121 and 310.7, respectively), which still necessitates countermeasures to reduce it. We leave creating a larger unbiased dataset that allows for training, calibrating, and evaluating (unbiased) vandalism detectors to future work.

6 EVALUATION AND BIAS OPTIMIZATION

We carried out a series of experiments to minimize the bias of our models while maintaining a competitive predictive performance. Table 4a gives an overview of the results we obtained for FAIR-E and FAIR-S, the state-of-the-art baselines WDVD, ORES, and FILTER, and variants thereof obtained by post-processing scores, reweighting training samples, and combining models in an ensemble. FAIR-E exhibits low bias at reasonable predictive performance; FAIR-S achieves higher predictive performance at a higher bias.

The literature has investigated trade-offs between different notions of fairness and predictive performance [4, 8, 10, 33], indicating that the two optimization goals cannot be satisfied at the same time, dependent on the notion of fairness and performance measure. Nevertheless, a sustainable model must strike a balance between fairness and predictive performance to ensure the long-term success of an online community. The state-of-the-art in vandalism detection has not yet reached its full potential in this respect.

In what follows, we briefly recap the baseline models and the prediction performance measures, and then detail the aforementioned experiments. To ensure reproducibility, the code base underlying our research is published alongside this paper.[10] While feature extraction is carried out in Java, our experiments are implemented in Python using scikit-learn, version 0.19.1. To calibrate classifier scores before computing bias, we use isotonic regression.

6.1 Baselines and Performance Measures

For the final evaluation on the test set, all baselines were trained on both the training and the validation set of WDVC-2016-Links, i.e., on about 6 million revisions ranging from April 2013 to April 2016. Unless otherwise indicated, the same hyperparameters and machine learning algorithms as reported in Heindorf et al. [25] were used: WDVD employs multiple instance learning on top of bagging and random forests. It uses 16 random forests, each build on 1/16 of the training dataset with the forests consisting of 8 trees, each having a maximal depth of 32 with two features per split using the default Gini split criterion. ORES uses a random forest with 80 decision trees considering 'log2' feature per split using the 'entropy' criterion. FILTER uses a random forest with scikit-learn's default hyperparameters, i.e., 10 decision trees without a maximal depth.

We employ performance measures for both predictive performance and bias performance, the latter using the bias measures introduced in Section 3. Predictive performance is measured in terms of area under curve of the receiver operating characteristics (ROC_{AUC}) and area under the precision-recall curve (PR_{AUC}). Both curves capture predictive performance across different operating points, where each point on one of the curves has a corresponding point on the other [12]. However, corresponding points are weighted differently when computing the area under the curves: ROC_{AUC} emphasizes operating points at high recall ranges and PR_{AUC} emphasizes points at high precision ranges. Hence, ROC_{AUC} is best suited to analyze semi-automatic vandalism detectors, where revisions are ranked and manually checked by reviewers, whereas, PR_{AUC} is best suited to analyze fully automatic detection systems, where vandalism is reverted without human intervention.

6.2 Debiasing via Feature Engineering

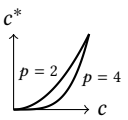
We obtain our debiased models FAIR-E and FAIR-S via feature engineering. For FAIR-E, we employ graph embeddings capturing the content of an edit in contrast to meta data such as user reputation, thus obtaining a bias ratio of only 5.6 on the test set compared to over 300 of the state-of-the-art approach WDVD. We experimented with different variants of what feature interactions to consider and Table 4b overviews our results on the validation set. In general, we observe that, with increasing model complexity, both predictive performance and bias increase. Given the relatively low bias, we opt to optimize predictive performance and choose the model $S \times P + P \times O$ as FAIR-E, taking into account all pairwise interactions between subject and predicate as well as predicate and object.

Based on feature selection and explicitly omitting user features, FAIR-S achieves a bias ratio of only 11.9 at 0.316 PR_{AUC} and 0.963 ROC_{AUC}. Plot (c) in Table 4 shows the trade-offs between bias and predictive performance obtainable by adding or by removing single features from the set of candidate features and outlines the Pareto front on the validation set. For example, removing the feature `objectPredicateCumFrequency` from FAIR-S yields the left-most point with both lower predictive performance and lower bias. We choose the best predictive model without user features as FAIR-S. Compared to WDVD, which consists of 47 features, FAIR-S consists of only 14 features, thus simplifying the model, reducing its runtime, and rendering its decisions much more explainable.

6.3 Debiasing via Post-Processing Scores

As an alternative to feature engineering, we can obtain comparable results in terms of bias and predictive performance by post-processing the scores of WDVD. We scale uncalibrated scores $c(i)$ from sample i of the protected class, while leaving those from the unprotected class unchanged, thus obtaining $c^*(i)$:

$$c^*(i) = \begin{cases} c(i)^p, & \text{revision } i \text{ from anon. user} \\ c(i), & \text{revision } i \text{ from reg. user} \end{cases}$$



[10] http://www.heindorf.me/wdvd

Table 4: Evaluation results: (a) Performance and bias of approaches on the test set of the vandalism corpus WDVC-2016-Links. (b) Optimization of FAIR-E on the validation set by testing all pairwise feature interactions between subject (S), predicate (P), and object (O) in predicate space. (c) Optimization of FAIR-S on the validation set. The plot shows ROC_{AUC} over bias of FAIR-S (purple) after adding (green) or removing (orange) a feature. The dotted line shows non-dominated points on the Pareto front. (d) Error analysis with respect to bias on our golden dataset.

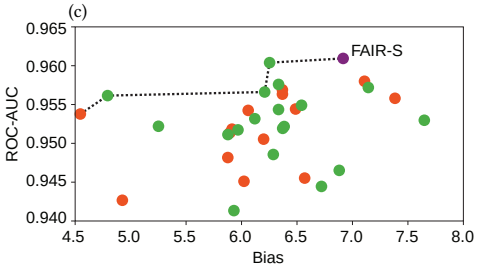
(a)

Debiasing Experiment Model	Performance		Bias	
	PR_{AUC}	ROC_{AUC}	Diff	Ratio
Feature engineering				
FAIR-E	0.177	0.865	0.016	5.6
FAIR-S	0.316	0.963	0.031	11.9
Post-processing scores				
WDVD with p=3.88	0.230	0.966	0.015	5.3
WDVD with p=3.22	0.340	0.976	0.030	11.8
Weighting training samples				
WDVD with $\alpha = 8.1$	0.160	0.963	0.015	5.3
WDVD with $\alpha = 4.3$	0.314	0.973	0.030	11.5
Combining models				
FAIR-E + WDVD	0.229	0.974	0.033	11.7
FAIR-E + ORES	0.238	0.967	0.033	11.8
FAIR-E + FILTER	0.224	0.953	0.033	11.8
Baselines				
WDVD	0.547	0.990	0.121	310.7
ORES	0.434	0.965	0.114	133.1
FILTER	0.302	0.924	0.096	69.2

(b)

Features	Performance		Bias	
	PR_{AUC}	ROC_{AUC}	Diff	Ratio
$\boldsymbol{S \times P + P \times O}$	**0.112**	**0.848**	**0.0081**	**2.90**
$S \times P + S \times O$	0.095	0.741	0.0071	2.65
$S \times O + P \times O$	0.068	0.721	0.0057	2.28
$S \times P \times O$	0.073	0.675	0.0064	2.45
$S \times P$	0.025	0.718	0.0021	1.45
$S \times O$	0.028	0.629	0.0023	1.50
$P \times O$	0.047	0.733	0.0031	1.69
$S + P + O$	0.068	0.739	0.0042	1.94
$S + P$	0.038	0.690	0.0036	1.78
$S + O$	0.034	0.719	0.0024	1.52
$P + O$	0.068	0.739	0.0042	1.94
S	0.020	0.643	0.0012	1.26
P	0.008	0.659	0.0008	1.17
O	0.011	0.663	0.0004	1.09

(c)



(d)

Users	Manually reviewed edits (n=1,000)			
Gold Truth:	Vandalism (n=524)		Benign (n=476)	
FAIR-S:	Vandalism (correct)	Benign (incorrect)	Vandalism (incorrect)	Benign (correct)
Anonymous	250	42	73	135
Registered	173	59	28	240

After scaling, we apply isotonic regression to calibrate the scores to represent probabilities, i.e., $c^*(i) \approx Pr(i = \text{vandalism} \mid x_i)$, in order to prevent bias being lower just due to a smaller interval of scores. By experimentally varying the parameter $p > 1$, we obtain classifiers that have approximately the same bias as our models FAIR-E and FAIR-S, allowing for performance comparisons. As Table 4a shows, we can achieve approximately the same predictive performance as with feature engineering. The advantage of post-processing scores is that the model does not have to be retrained. However, the model's decisions are also much more difficult to understand. It is less clear what effect single features have, thus hindering an intuitive explanation to Wikidata editors whose edits have been reverted. Apart from the polynomial scaling function introduced above, we experimented with other families of functions, including linear, fractional, and exponential functions. Polynomial scaling outperformed all other variants in terms of PR_{AUC}.

6.4 Debiasing via Weighting Training Samples

As another option to reduce biases, we experimented with reweighting training samples. We start by assigning each of the four groups of protected/unprotected, benign/vandalism revisions the same weight. Then we increase the weight of benign edits by protected users such that it becomes prohibitively expensive for the algorithm to make mistakes on this set of revisions. Symmetrically, we increase the weight of vandalism edits by unprotected users. Thus, using the notation from Section 3, the weight of each training sample i is obtained by:

$$w(i) = \begin{cases} \alpha \cdot \frac{1}{|A|}, & \text{revision } i \in A \\ 1 \cdot \frac{1}{|B|}, & \text{revision } i \in B \\ 1 \cdot \frac{1}{|D|}, & \text{revision } i \in D \\ \alpha \cdot \frac{1}{|E|}, & \text{revision } i \in E \end{cases}$$

		Truth benign	Truth vand.	∪
Protected	yes	A	B	C
	no	D	E	F
∪		G	H	I

By varying the constant $\alpha > 1$, we obtain classifiers with varying trade-offs between bias and predictive performance. For the weighting experiment in Table 4a, we reduce the maximal tree depth of WDVD from 32 to 16 to increase the number of training samples per leaf, giving the weighting a larger effect. We experimentally determine α to achieve approximately the same bias as our models FAIR-E and FAIR-S, allowing for performance comparisons. However, predictive performance is lower than that obtained by feature engineering or post-processing scores. We believe the reason is that reweighting leads to overfitting on certain subsets of the data and the effect on bias is rather indirect by reducing the misclassification rate on these subsets, instead of tackling bias directly. Hence, we suggest to rather use feature engineering or post-processing scores. Besides, we also tried to employ separate constants α_1 and α_2 per group, but no better bias and predictive performance was achieved, so we resorted to the basic weighting scheme.

6.5 Debiasing via Combining Models

In yet another debiasing attempt, we train ensembles of models with high predictive performance and low-bias models to derive a model with characteristics in between. A weighted average between the scores of the two models is computed. Table 4a shows the results: While it is possible to use ensembles to reconcile predictive performance with bias, this method appears to be inferior to feature engineering and post-processing scores.

6.6 Debiasing Error Analysis

Table 4d shows the results of an error analysis for FAIR-S. The continuous scores of FAIR-S were converted to binary decisions, so that predicted vandalism prevalence equals the vandalism prevalence in the golden set from Table 3d. The table shows, that 73 benign edits by anonymous users were classified as vandalism. Manually investigating these edits, we find that most of the edited triples are

updates (53), rather than removals (11) or creations (9). In most cases, the subject is a human. Predicates frequently affected are "occupation" (P106, 10 cases), "instance of" (P31, 8), "country of citizenship" (P27, 6), and "sex or gender" (P21, 6). It might be beneficial to develop vandalism detectors tailored to update operations, e.g., by comparing the old and new object of a triple. Also specific features targeting problematic predicates might help. Beyond intrinsic plausibility checks, also double-checks in external databases or web search engines seem promising.

7 DISCUSSION

Biases in machine learning are still widely neglected by researchers and practitioners in machine learning and data science, who primarily focus on optimizing predictive performance, disregarding fairness constraints that are essential for the long-term success of online communities. Approaches that discriminate against sex, race, religion, or that are otherwise biased against minorities—or even against majorities[11]—jeopardize an online community's long-term goal of promoting a feeling of fairness, security, and belonging. We hope that our endeavor helps towards increasing awareness.

Our bias definition emphasizes fair treatment of *benign* edits, which is also known as *equality of opportunity* [23], and which we believe is well-aligned to the goal of newcomer retention. Fair treatment of both *benign* and *vandalism* edits would correspond to the stricter fairness notion *equalized odds* [23], making it potentially more challenging to find a good balance between predictive performance and fairness. Another fairness notion that is sometimes mentioned by the media, lawmakers, and the literature is *statistical parity* [16, 17], requiring that the same proportions of anonymous and registered edits are classified as vandalism. This neglects that the two populations might have different vandalism prevalences, and Dwork et al. [16] argue that it is an inadequate notion of fairness. First results [4, 8, 10, 33] suggest that there is a trade-off between fairness and predictive performance for many notions of fairness, requiring difficult trade-offs when the set of features is fixed. We believe decisions about an edit should focus on the edit's "content", instead of the reputation of the user who submitted it. But even without features prone to incur bias, there can still be bias against certain groups, since seemingly harmless features might correlate with biased features. For example, in Table 2, features related to edit operations and subjects introduce (a small) bias against anonymous edits. However, forcing two groups to have exactly the same score distribution might introduce reverse discrimination, also known as affirmative action [3, 48, 66], since there might be hidden confounding variables justifying certain differences. In the future, causal modeling [32, 34, 64] might help to decide what causal relations to consider.

Wikidata makes for an interesting case study to analyze and mitigate biases as it has one of the largest online communities and provides opportunities to pay particular attention to the content rather than the user reputation. We believe some lessons learned in this project can be transferred to other projects, too. It is common practice to identify malicious edits via meta data such as geolocation of IP addresses, age of user account, or browser information such as user language. While those features are simple to obtain, they do not directly judge the quality of an edit and harm well-intentioned users. Our graph embeddings serve as an example of how features purely judging the content of an edit help to reduce unintentional biases. Perhaps surprisingly, we can obtain comparable performance-bias trade-offs by artificially scaling scores after a biased model has been trained. On the one hand, this approach might serve as an easy route to debias existing models. On the other hand, we feel that it is rather a "black-box approach" and might be less suitable for understanding predictions and explaining them to editors.

Finally, we would like to point out some limitations and directions for future research. Our model FAIR-E was specifically designed for edits affecting links between entities and extending it to attributes, labels, descriptions, and aliases might be challenging, with the exception of our model FAIR-S. Here, such an extension is rather straightforward as it selects features from WDVD which is capable of classifying all edits in Wikidata. Our analysis is based on a dataset derived from the rollback actions of Wikidata reviewers and we observed some reviewer bias in this dataset. We leave it to future work to create a large-scale, unbiased dataset. For example, the Wikimedia Foundation might decide to hide user information from Wikidata reviewers, forcing them to purely consider the content of an edit. However, this might make the review process more expensive as reviewers cannot skip or quickly skim large amounts of edits by registered users anymore. Besides fair vandalism detectors, further means to increase newcomer retention might include vandalism detectors explaining their decisions, improved user interfaces, onboarding programs for newcomers, increased social interactions, and gamification—all accompanied by data-driven processes such as A/B testing or reinforcement learning.

8 CONCLUSION AND OUTLOOK

Machine learning models ingest biases through training data and features—sometimes even aggravating them—fueling a vicious cycle of reinforcing biases in a larger system. In this work, we developed a vandalism detector for Wikidata's damage control system that does not contribute significant bias of its own. Compared to the state-of-the-art, it considerably reduces bias against edits by anonymous and newly registered editors. We achieve this result by omitting user-related features and by developing features that purely encode the content of an edit, rather than any meta information. In the future, we plan to investigate biases at further online platforms and to develop a general framework for bias mitigation, e.g., by employing evolutionary algorithms to explore the Pareto front of non-dominated models in bias-performance space.

ACKNOWLEDGMENTS

This work was supported by the German Research Foundation (DFG) within the Collaborative Research Center "On-The-Fly Computing" (CRC 901). Special thanks go to the anonymous reviewers for their helpful comments.

[11] Anonymous *edits* are a clear minority in our dataset since only about 2.4% of edits are by anonymous editors. However, anonymous *editors* might possibly be in the majority, since the number of distinct IP addresses from anonymous editors is about 2.5 times higher than the number of distinct user accounts from registered editors (our data does not reveal how often the same anonymous editor uses a different IP address).

REFERENCES

[1] B. T. Adler, L. de Alfaro, S. M. Mola-Velasco, P. Rosso, and A. G. West. 2011. Wikipedia Vandalism Detection: Combining Natural Language, Metadata, and Reputation Features. In *CICLing*. Springer, 277–288.

[2] R. Baeza-Yates. 2018. Bias on the Web. *Commun. ACM* 61, 6 (2018), 54–61.

[3] S. Barocas and A. D. Selbst. 2016. Big data's disparate impact. *Cal. L. Rev.* 104 (2016), 671.

[4] R. Berk, H. Heidari, S. Jabbari, M. Kearns, and A. Roth. 2018. Fairness in Criminal Justice Risk Assessments: The State of the Art. *Sociological Methods & Research* (2018).

[5] T. Bolukbasi, K. Chang, J. Y. Zou, V. Saligrama, and A. T. Kalai. 2016. Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. In *NIPS*. 4349–4357.

[6] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NIPS*. 2787–2795.

[7] T. Calders and S. Verwer. 2010. Three naive Bayes approaches for discrimination-free classification. *Data Min. Knowl. Discov.* 21, 2 (2010), 277–292.

[8] A. Chouldechova. 2017. Fair Prediction with Disparate Impact: A Study of Bias in Recidivism Prediction Instruments. *Big Data* 5, 2 (2017), 153–163.

[9] G. L. Ciampaglia, P. Shiralkar, L. M. Rocha, J. Bollen, F. Menczer, and A. Flammini. 2015. Computational Fact Checking from Knowledge Networks. *PLOS ONE* 10, 6 (2015), 1–13.

[10] S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq. 2017. Algorithmic Decision Making and the Cost of Fairness. In *KDD*. ACM, 797–806.

[11] F. Darari, S. Razniewski, R. E. Prasojo, and W. Nutt. 2016. Enabling Fine-Grained RDF Data Completeness Assessment. In *ICWE*. Springer, 170–187.

[12] J. Davis and M. Goadrich. 2006. The Relationship Between Precision-Recall and ROC Curves. In *ICML*. ACM, 233–240.

[13] L. Dixon, J. Li, J. Sorensen, N. Thain, and L. Vasserman. 2018. Measuring and Mitigating Unintended Bias in Text Classification. In *AIES*. ACM, 67–73.

[14] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. 2014. Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion. In *KDD*. ACM, 601–610.

[15] X. L. Dong, E. Gabrilovich, K. Murphy, V. Dang, W. Horn, C. Lugaresi, S. Sun, and W. Zhang. 2016. Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources. *IEEE Data Eng. Bull.* 39, 2 (2016), 106–117.

[16] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. S. Zemel. 2012. Fairness Through Awareness. In *ITCS*. ACM, 214–226.

[17] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In *KDD*. ACM, 259–268.

[18] L. Galárraga, S. Razniewski, A. Amarilli, and F. M. Suchanek. 2017. Predicting Completeness in Knowledge Bases. In *WSDM*. ACM, 375–383.

[19] A. Gangemi, A. G. Nuzzolese, V. Presutti, F. Draicchio, A. Musetti, and P. Ciancarini. 2012. Automatic Typing of DBpedia Entities. In *ISWC*. 65–81.

[20] M. Gardner and T. M. Mitchell. 2015. Efficient and Expressive Knowledge Base Completion Using Subgraph Feature Extraction. In *EMNLP*. ACL, 1488–1498.

[21] A. Halfaker, R. S. Geiger, J. T. Morgan, and J. Riedl. 2013. The Rise and Decline of an Open Collaboration System: How Wikipedia's Reaction to Popularity is Causing Its Decline. *American Behavioral Scientist* 57, 5 (2013), 664–688.

[22] A. Halfaker, A. Kittur, and J. Riedl. 2011. Don't Bite the Newbies: How Reverts Affect the Quantity and Quality of Wikipedia Work. In *Int. Sym. Wikis*. 163–172.

[23] M. Hardt, E. Price, and N. Srebro. 2016. Equality of Opportunity in Supervised Learning. In *NIPS*. 3315–3323.

[24] S. Heindorf, M. Potthast, H. Bast, B. Buchhold, and E. Haussmann. 2017. WSDM Cup 2017: Vandalism Detection and Triple Scoring. In *WSDM*. ACM, 827–828.

[25] S. Heindorf, M. Potthast, G. Engels, and B. Stein. 2017. Overview of the Wikidata Vandalism Detection Task at the WSDM Cup 2017. In *WSDM Cup 2017 Notebook Papers*.

[26] S. Heindorf, M. Potthast, B. Stein, and G. Engels. 2015. Towards Vandalism Detection in Knowledge Bases: Corpus Construction and Analysis. In *SIGIR*. ACM, 831–834.

[27] S. Heindorf, M. Potthast, B. Stein, and G. Engels. 2016. Vandalism Detection in Wikidata. In *CIKM*. ACM, 327–336.

[28] A. Jain and P. Pantel. 2010. FactRank: Random Walks on a Web of Facts. In *COLING*. Tsinghua University Press, 501–509.

[29] S. Javanmardi, D. W. McDonald, and C. V. Lopes. 2011. Vandalism Detection in Wikipedia: A High-Performing, Feature-Rich Model and its Reduction Through Lasso. In *Int. Sym. Wikis*. ACM, 82–90.

[30] F. Kamiran, T. Calders, and M. Pechenizkiy. 2010. Discrimination Aware Decision Tree Learning. In *ICDM*. IEEE Computer Society, 869–874.

[31] J. Kiesel, M. Potthast, M. Hagen, and B. Stein. 2017. Spatio-Temporal Analysis of Reverted Wikipedia Edits. In *ICWSM*. AAAI Press, 122–131.

[32] N. Kilbertus, M. Rojas-Carulla, G. Parascandolo, M. Hardt, D. Janzing, and B. Schölkopf. 2017. Avoiding Discrimination through Causal Reasoning. In *NIPS*. 656–666.

[33] J. M. Kleinberg, S. Mullainathan, and M. Raghavan. 2017. Inherent Trade-Offs in the Fair Determination of Risk Scores. In *ITCS*, Vol. 67. 43:1–43:23.

[34] M. J. Kusner, J. R. Loftus, C. Russell, and R. Silva. 2017. Counterfactual Fairness. In *NIPS*. 4069–4079.

[35] J. Lajus and F. M. Suchanek. 2018. Are All People Married?: Determining Obligatory Attributes in Knowledge Bases. In *WWW*. ACM, 1115–1124.

[36] N. Lao, T. M. Mitchell, and W. W. Cohen. 2011. Random Walk Inference and Learning in A Large Scale Knowledge Base. In *EMNLP*. ACL, 529–539.

[37] J. Lehmann, D. Gerber, M. Morsey, and A. N. Ngomo. 2012. DeFacto - Deep Fact Validation. In *ISWC*. Springer, 312–327.

[38] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *AAAI*. AAAI Press, 2181–2187.

[39] A. Melo, H. Paulheim, and J. Völker. 2016. Type Prediction in RDF Knowledge Bases Using Hierarchical Multilabel Classification. In *WIMS*. ACM, 14:1–14:10.

[40] E. Minkov, W. W. Cohen, and A. Y. Ng. 2006. Contextual Search and Name Disambiguation in Email Using Graphs. In *SIGIR*. ACM, 27–34.

[41] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich. 2016. A Review of Relational Machine Learning for Knowledge Graphs. *Proc. IEEE* 104, 1 (2016), 11–33.

[42] M. Nickel, V. Tresp, and H. Kriegel. 2012. Factorizing YAGO: Scalable Machine Learning for Linked Data. In *WWW*. ACM, 271–280.

[43] C. Nishioka and A. Scherp. 2018. Analysing the Evolution of Knowledge Graphs for the Purpose of Change Verification. In *ICSC*. IEEE Computer Society, 25–32.

[44] H. Paulheim and C. Bizer. 2013. Type Inference on Noisy RDF Data. In *ISWC*. Springer, 510–525.

[45] D. Pedreschi, S. Ruggieri, and F. Turini. 2009. Measuring Discrimination in Socially-Sensitive Decision Records. In *SDM*. SIAM, 581–592.

[46] M. Potthast, B. Stein, and R. Gerling. 2008. Automatic Vandalism Detection in Wikipedia. In *ECIR*. Springer, 663–668.

[47] E. Raff, J. Sylvester, and S. Mills. 2018. Fair Forests: Regularized Tree Induction to Minimize Model Bias. In *AIES*. ACM, 243–250.

[48] A. Romei and S. Ruggieri. 2014. A multidisciplinary survey on discrimination analysis. *Knowledge Eng. Review* 29, 5 (2014), 582–638.

[49] A. Sarabadani, A. Halfaker, and D. Taraborelli. 2017. Building Automated Vandalism Detection Tools for Wikidata. In *WWW (Companion Volume)*. ACM, 1647–1654.

[50] J. Schneider, B. S. Gelley, and A. Halfaker. 2014. Accept, decline, postpone: How newcomer productivity is reduced in English Wikipedia by pre-publication review. In *OpenSym*. ACM, 26:1–26:10.

[51] B. Shi and T. Weninger. 2016. Discriminative predicate path mining for fact checking in knowledge graphs. *Knowl.-Based Syst.* 104 (2016), 123–133.

[52] C. H. Tan, E. Agichtein, P. Ipeirotis, and E. Gabrilovich. 2014. Trust, but Verify: Predicting Contribution Quality for Knowledge Base Construction and Curation. In *WSDM*. ACM, 553–562.

[53] A. Torralba and A. A. Efros. 2011. Unbiased Look at Dataset Bias. In *CVPR*. IEEE Computer Society, 1521–1528.

[54] K. Tran and P. Christen. 2013. Cross Language Prediction of Vandalism on Wikipedia Using Article Views and Revisions. In *PAKDD*. Springer, 268–279.

[55] Q. Wang, Z. Mao, B. Wang, and L. Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Trans. Knowl. Data Eng.* 29, 12 (2017), 2724–2743.

[56] W. Y. Wang and K. McKeown. 2010. "Got You!": Automatic Vandalism Detection in Wikipedia with Web-based Shallow Syntactic-Semantic Modeling. In *COLING*. Tsinghua University Press, 1146–1154.

[57] X. Wang, M. Bendersky, D. Metzler, and M. Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *SIGIR*. ACM, 115–124.

[58] Z. Wang, J. Zhang, J. Feng, and Z. Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *AAAI*. AAAI Press, 1112–1119.

[59] C. Wilkie and L. Azzopardi. 2017. Algorithmic Bias: Do Good Systems Make Relevant Documents More Retrievable?. In *CIKM*. ACM, 2375–2378.

[60] Y. Wu, P. K. Agarwal, C. Li, J. Yang, and C. Yu. 2014. Toward Computational Fact-Checking. *PVLDB* 7, 7 (2014), 589–600.

[61] K. Yang and J. Stoyanovich. 2017. Measuring Fairness in Ranked Outputs. In *SSDBM*. ACM, 22:1–22:6.

[62] M. B. Zafar, I. Valera, M. Gomez-Rodriguez, and K. P. Gummadi. 2017. Fairness Beyond Disparate Treatment & Disparate Impact: Learning Classification without Disparate Mistreatment. In *WWW*. ACM, 1171–1180.

[63] R. S. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. 2013. Learning Fair Representations. In *ICML (3) (JMLR Workshop and Conference Proceedings)*, Vol. 28. JMLR.org, 325–333.

[64] L. Zhang and X. Wu. 2017. Anti-discrimination learning: a causal modeling-based framework. *I. J. Data Science and Analytics* 4, 1 (2017), 1–16.

[65] J. Zhao, T. Wang, M. Yatskar, V. Ordonez, and K. Chang. 2017. Men Also Like Shopping: Reducing Gender Bias Amplification using Corpus-level Constraints. In *EMNLP*. ACL, 2979–2989.

[66] I. Zliobaite. 2015. On the relation between accuracy and fairness in binary classification. *CoRR* abs/1505.05723 (2015).

[67] I. Zliobaite. 2017. Measuring discrimination in algorithmic decision making. *Data Min. Knowl. Discov.* 31, 4 (2017), 1060–1089.

Message Distortion in Information Cascades

Manoel Horta Ribeiro*
UFMG
manoelribeiro@dcc.ufmg.br

Kristina Gligorić
EPFL
kristina.gligoric@epfl.ch

Robert West
EPFL
robert.west@epfl.ch

ABSTRACT

Information diffusion is usually modeled as a process in which immutable pieces of information propagate over a network. In reality, however, messages are not immutable, but may be morphed with every step, potentially entailing large cumulative distortions. This process may lead to misinformation even in the absence of malevolent actors, and understanding it is crucial for modeling and improving online information systems. Here, we perform a controlled, crowdsourced experiment in which we simulate the propagation of information from medical research papers. Starting from the original abstracts, crowd workers iteratively shorten previously produced summaries to increasingly smaller lengths. We also collect control summaries where the original abstract is compressed directly to the final target length. Comparing cascades to controls allows us to separate the effect of the length constraint from that of accumulated distortion. Via careful manual coding, we annotate lexical and semantic units in the medical abstracts and track them along cascades. We find that iterative summarization has a negative impact due to the accumulation of error, but that high-quality intermediate summaries result in less distorted messages than in the control case. Different types of information behave differently; in particular, the conclusion of a medical abstract (i.e., its key message) is distorted most. Finally, we compare extractive with abstractive summaries, finding that the latter are less prone to semantic distortion. Overall, this work is a first step in studying information cascades without the assumption that disseminated content is immutable, with implications on our understanding of the role of word-of-mouth effects on the misreporting of science.

CCS CONCEPTS

• **Human-centered computing → Empirical studies in collaborative and social computing**.

KEYWORDS

Information cascades; message distortion; information distortion

ACM Reference Format:
Manoel Horta Ribeiro, Kristina Gligorić, and Robert West. 2019. Message Distortion in Information Cascades. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313531

* Work done during an internship at EPFL. Code/data: `github.com/epfl-dlab/mdic`

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313531

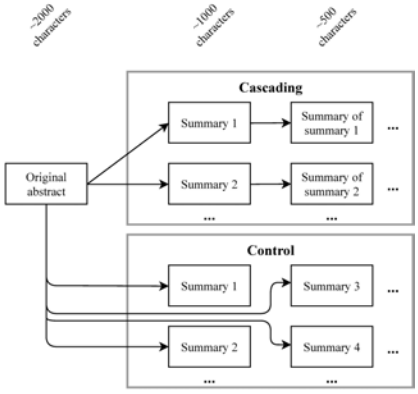


Figure 1: Schema of the crowdsourced experiment for simulating information cascades. In the cascading setting, workers summarize texts iteratively, reducing the number of characters hop by hop. In the control setting, workers always summarize the original text for all target lengths.

1 INTRODUCTION

The spread of information, online and offline, is a noisy process. As a message is passed on from person to person, or from platform to platform, errors creep in, and facts are distorted, oftentimes to an extent such that, after a few hops of propagation, downstream messages may be entirely different from—or even contradict—the original message. This way, valuable information may turn into harmful misinformation, even without purposeful interference.

Misconceptions and sensationalism add their part to compound the problem, as frequently observed in the context of health-related topics such as dieting and vaccination. For example, in 2006, the first results of the *Women's Health Initiative Dietary Intervention* trial were published. The trial found little impact of diets lower in fat and higher in fruits, vegetables, and grains in the incidence of certain diseases in women between 50 and 79 years old [25]. Shortly after its publication, a sequence of press releases and news stories increasingly distorted the nuanced and cautious findings of the study, overlooking methodological caveats and benefits found [35]. Throughout the diffusion process, news overwhelmingly reported that food and nutrition had little to do with health and disease [9].

This anecdote portrays how information may be distorted as it propagates over the news and social media. These distortion processes are overlooked in the existing literature on information diffusion [21], which treats information disseminated through a network as consisting of immutable pieces of content (e.g., *memes* [34] or topics [10]). Previous research in communication studies [14, 22] indicates, however, that the way information is altered, interpreted, or framed along its diffusion may have a significant impact.

Two orthogonal factors are at play during message propagation:

(1) **Word of mouth:** Information commonly spreads in a cascading fashion, from person to person, or from platform to platform, rather than directly from the original source to every person or platform.
(2) **Summarization:** When an original message is passed on, it is frequently compressed, focusing on the essence while omitting unnecessary details.

Both factors can introduce errors. First, word-of-mouth propagation usually takes place on noisy channels (unless messages are forwarded unmodified, e.g., via retweets), and when an error occurs, it is passed on via what we term the **telephone effect,** named after the *telephone game,* in which *"players form a line, and the first player comes up with a message and whispers it to the ear of the second person in the line. The second player repeats the message to the third player, and so on. When the last player is reached, they announce the message they heard to the entire group. The first person then compares the original message with the final version. Although the objective is to pass around the message without it becoming garbled along the way, this usually ends up happening"* [1]. Second, summarization can be seen as lossy compression and thus induces an additional loss of information, which we term the **summary effect.**

Consider, e.g., this three-hop cascade: a ten-page medical research paper is promoted in a one-page university press release, which is picked up by a half-page newspaper article, which is finally mentioned in a tweet with 280 characters. It is clear that the tweet at the end of the cascade will be different from the original research paper. What is less clear is whether the difference stems from the fact that the message was passed on three times (the telephone effect), or from the fact that ten pages were compressed to 280 characters (the summary effect). Disentangling the telephone and summary effects is difficult when working with observationally collected information cascades as studied in prior work, e.g., URLs spreading via tweets [50] or quotations spreading via news articles and blog posts [34]. Moreover, assembling an appropriate dataset in the first place is challenging, too. For instance, in the case of the aforementioned *Women's Health* trial, it is hard even to identify the structure of cascades, i.e., the graph that specifies from which other node each node received the message. If an article misreported the findings, it is unclear if the author read the original scientific report and misunderstood it, or if they read other articles that had already introduced the error. Also, the level of coverage may vary widely: a special feature on women's health in a tabloid may briefly mention the trial, whereas an editorial in *Science* may be dedicated to it entirely. Meaningfully comparing such different formats is hard.

Present work: an experimental framework for studying message distortion in information cascades. These challenges associated with observational settings motivated us to adopt an experimental approach. In our experimental design, inspired by the telephone game, we aim to track the distortion of messages as they propagate hop by hop. Starting from an original message, we simulate an information cascade by asking a crowd worker to shorten the original message to a prescribed target length while preserving the essential information. The resulting summary is then passed on to another worker, who is asked to condense it to an even shorter target length, and so forth. This way, we obtain chains of **cascading summaries.** Along the chains, the original message is distorted by the telephone and summary effects. To tease the two effects apart, we also collect **control summaries,** of the same lengths as the cascading summaries, but produced by directly summarizing the original message to the respective target length, without any intermediate summaries. This setup is depicted in Fig. 1. Cascading summaries are subject to both the telephone and the summary effects, whereas control summaries are only subject to the summary effect, so the difference in error rates in cascading vs. control summaries can be ascribed to the telephone effect.

This experimental design allows us to address the following **research questions:**

RQ1: Measuring the telephone effect. What part of information distortion is due to the cascading of messages (telephone effect), rather than to length restrictions (summary effect)?
RQ2: Information persistence. Given that a piece of information has already survived k summarization steps, how likely is it to survive one more? What factors impact its survival?
RQ3: Extractive vs. abstractive summarization. Broadly, there are two ways of summarizing text, (1) by subselecting keyphrases (*extractive*), (2) by paraphrasing essential information (*abstractive*). How effective are these strategies in mitigating distortion introduced by the telephone effect?

Application to medical information. Motivated by the importance of medical information, we apply the above framework to a scenario where original messages are abstracts of papers published in the *New England Journal of Medicine* (NEJM). Automatically identifying the key facts contained in a medical abstract is an open challenge [45], and so is determining the presence vs. absence of those facts in subsequent summaries. Hence, in order to reliably address the above research questions, we manually annotate lexical and semantic units in the abstracts and track them along cascades.

We find that, overall, cascading summarization has a detrimental effect due to the accumulation of error. High-quality intermediate summaries, however, can have a positive effect, by isolating the essential information and discarding noise, entailing less distorted subsequent messages than in the control case. Different types of information behave differently; in particular, the conclusion of a medical abstract—the most critical information —is distorted most: the conclusion is correctly represented in cascading summaries by about 25 percentage points less, compared to control summaries, for a fixed target length. Moreover, we find that the prior knowledge of the crowd workers impact the persistence of information in the control setting, but not in the cascading setting. Finally, comparing extractive with abstractive summaries, we find that extractive summaries (where more keyphrases are preserved) are less prone to semantic distortion.

Implications. Beyond the special case of medical information, our work has general implications for the study of information cascades in a setting where information is not immutable and atomic but subject to distortion and omission. In this context, modeling the distortion of content through its diffusion in a network can help us understand the nature of viral content and the processes through which biased or erroneous information arises from a correct source. Furthermore, these insights could be used to purposefully create content that is less prone to distortion as it is diffused.

Table 1: Papers used and associated topics: vaccination (VA), breast cancer (BC), cardiovascular diseases (CD), nutrition (NU).

Paper	Topic	Paper	Topic
A population-based study of measles, mumps, and rubella vaccination and autism [36]	VA	Effect of rosiglitazone on the risk of myocardial infarction and death from cardiovascular causes [44]	CD
Response to a monovalent 2009 influenza A (H1N1) vaccine [20]	VA	Azithromycin and the risk of cardiovascular death [48]	CD
First results of phase 3 trial of RTS, S/AS01 malaria vaccine in African children [51]	VA	Global sodium consumption and death from cardiovascular causes [41]	CD
Waning protection after fifth dose of acellular pertussis vaccine in children [31]	VA	Effect of sibutramine on cardiovascular outcomes in overweight and obese subjects [29]	CD
Adjuvant exemestane with ovarian suppression in premenopausal breast cancer [46]	BC	Changes in diet and lifestyle and long-term weight gain in women and men [42]	NU
Effect of screening mammography on breast-cancer mortality in Norway [30]	BC	Comparison of weight-loss diets with different compositions of fat, protein, and carbohydrates [52]	NU
Exemestane for breast-cancer prevention in postmenopausal women [19]	BC	Primary prevention of cardiovascular disease with a Mediterranean diet [15]	NU
Effect of three decades of screening mammography on breast-cancer incidence [7]	BC	Association of coffee drinking with total and cause-specific mortality [17]	NU

2 RESEARCH DESIGN

We describe our experimental design for collecting cascading and control summaries (Sec. 2.1), the dataset of cascades of medical information (Sec. 2.2), and our method for extracting keyphrases and facts from abstracts and tracking them along cascades (Sec. 2.3).

2.1 Collecting information cascades

We perform a controlled experiment to simulate an information diffusion process through text summarization tasks. Starting from an original text of length l_0, we leverage crowd workers to shorten it to a sequence of prescribed target lengths $l_1 > l_2 > \cdots > l_n$. More specifically, workers were given the following instructions:

> You will be given a short text (l_k characters) with medicine-related information. Your tasks are these:
>
> (1) Read the text carefully.
> (2) Write a summary of the text. Your summary should
> (a) convey the most important information in the text, as if you are trying to inform another person about what you just read;
> (b) contain between $l_{k+1} - \Delta_{k+1}$ and $l_{k+1} + \Delta_{k+1}$ characters.
>
> We expect high-quality summaries and will manually inspect some of them. Copy-pasting is disabled.

Cascading and control summaries differ with respect to the input given to workers. In cascading summaries, a crowd worker whose task is to produce a summary of length l_k is given as input a summary of length l_{k-1} that was previously produced by another crowd worker. Only for target length l_1 is the original text used as input. In control summaries, on the contrary, the input is always the original text, regardless of the target length.

For each original text, target length, and condition, we collect multiple summaries. Cascading summaries form a set of chains rooted in the original abstracts, whereas control summaries are flat broadcast trees. This is depicted in Fig. 1.

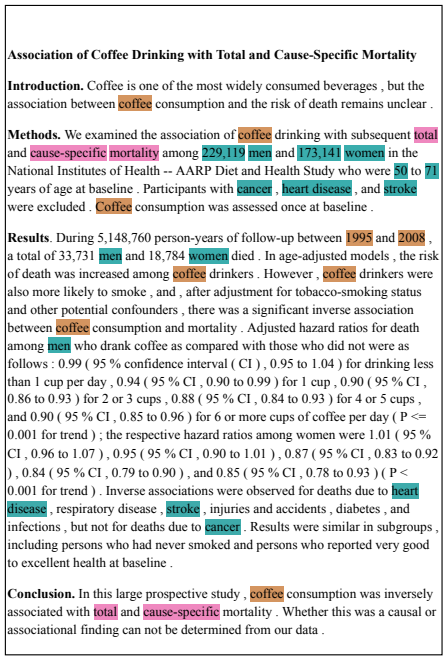
Cascading summaries are affected by both the telephone and the summary effect (as defined in Sec. 1), whereas control summaries are affected only by the summary effect, such that comparing the two cases allows us to quantify the telephone effect. One might argue that the telephone effect could be isolated in a more straightforward fashion by using a constant target length throughout, asking workers to simply rephrase the input, and thus eliminating the summary effect altogether. This, however, would allow for trivial solutions on behalf of crowd workers: nothing would keep them from simply copying the input, such that by eliminating the summary effect, the telephone effect would also vanish. This shortcoming could be addressed using a different research design, e.g., by showing workers the input text for a certain amount of time, then hiding it and asking them to reproduce it from memory. This setup, however, is more complex to implement, requires workers to spend idle time before finishing the task, and runs the risk of cheating (e.g., even when disabling copy–paste, users might take a screenshot or a picture). Also, for longer original texts (such as research paper abstracts), summarization is a step users would naturally perform in real cascades. For all these reasons, we adopt the summarization-based paradigm. Due to budget constraints, we create cascades such that each node has one descendant, since otherwise, the cost of the experiment would increase exponentially.

2.2 Dataset: cascades of medical information

In our concrete application of the above experimental design, we collect cascades whose root nodes consist of medical abstracts from the *New England Journal of Medicine* (NEJM). We select four research fields of public interest (vaccination, breast cancer, cardiovascular diseases, and nutrition), and choose 4 impactful papers per field, for a total of 16 abstracts, listed in Table 1. Analyzing the number of links to these papers in a large corpus of blog posts and news pieces, we select papers that were widely discussed online.

For each abstract, we generate 8 chains of cascading summaries, so that for each target length, each abstract had 8 summaries associated with it. In parallel, we collect another 8 independent control summaries per target length, totaling $16 \times 8 \times 2 = 256$ summaries per target length. Original abstracts were $l_0 \approx 2000$ characters long, and we consider five target lengths (which we also refer to as *hops*): $l_1 = 1000$, $l_2 = 500$, $l_3 = 250$, $l_4 = 125$, and $l_5 = 64$ characters. We allow slacks $\Delta_1 = 100$, $\Delta_2 = 50$, $\Delta_3 = 25$, $\Delta_4 = 13$ and $\Delta_5 = 9$, for each budget respectively.

For several reasons, we enforce that workers only summarize one text per hop per abstract. Firstly, we do not want a single worker to be involved in several summarization chains, as this would imply that a single unskilled or malicious worker could jeopardize the quality of all the chains. Secondly, we do not want information to leak from one summary to another. As different chains are related to the same paper, it could be that workers understood the text better if they summarized multiple summaries originating from the same paper. Due to the latter reason, we also limit workers to do summaries related to a paper in different hops 36 hours apart.



Association of Coffee Drinking with Total and Cause-Specific Mortality

Introduction. Coffee is one of the most widely consumed beverages , but the association between coffee consumption and the risk of death remains unclear .

Methods. We examined the association of coffee drinking with subsequent total and cause-specific mortality among 229,119 men and 173,141 women in the National Institutes of Health -- AARP Diet and Health Study who were 50 to 71 years of age at baseline . Participants with cancer , heart disease , and stroke were excluded . Coffee consumption was assessed once at baseline .

Results. During 5,148,760 person-years of follow-up between 1995 and 2008 , a total of 33,731 men and 18,784 women died . In age-adjusted models , the risk of death was increased among coffee drinkers . However , coffee drinkers were also more likely to smoke , and , after adjustment for tobacco-smoking status and other potential confounders , there was a significant inverse association between coffee consumption and mortality . Adjusted hazard ratios for death among men who drank coffee as compared with those who did not were as follows : 0.99 (95 % confidence interval (CI) , 0.95 to 1.04) for drinking less than 1 cup per day , 0.94 (95 % CI , 0.90 to 0.99) for 1 cup , 0.90 (95 % CI , 0.86 to 0.93) for 2 or 3 cups , 0.88 (95 % CI , 0.84 to 0.93) for 4 or 5 cups , and 0.90 (95 % CI , 0.85 to 0.96) for 6 or more cups of coffee per day (P <= 0.001 for trend) ; the respective hazard ratios among women were 1.01 (95 % CI , 0.96 to 1.07) , 0.95 (95 % CI , 0.90 to 1.01) , 0.87 (95 % CI , 0.83 to 0.92) , 0.84 (95 % CI , 0.79 to 0.90) , and 0.85 (95 % CI , 0.78 to 0.93) (P < 0.001 for trend) . Inverse associations were observed for deaths due to heart disease , respiratory disease , stroke , injuries and accidents , diabetes , and infections , but not for deaths due to cancer . Results were similar in subgroups , including persons who had never smoked and persons who reported very good to excellent health at baseline .

Conclusion. In this large prospective study , coffee consumption was inversely associated with total and cause-specific mortality . Whether this was a causal or associational finding can not be determined from our data .

Coarse category	Fine category	Facts	Keyphrases
Participants	Sex	The study was performed in women and men.	men; women
	Age	The participants were between 50 to 71 years of age at baseline.	50; 71
	Condition	Participants with cancer, heart disease and stroke were excluded.	cancer; heart disease; stroke
	Location	N/A	
	Sample size	There were 229,119 men and 173,141 women among the participants.	229,119; 173,141
Intervention	General	The study assessed the impact of coffee consumption.	coffee
	Duration	The study followed up individuals between 1995 and 2008.	1995; 2008
	Intensity	N/A	
	Control	N/A	
Outcomes	General	The study measured long-term total and cause-specific mortality.	total; cause-specific; mortality
	Effect strength	N/A	
	Adverse effects	N/A	
Conclusion	General	Coffee consumption was inversely associated with total and cause-specific mortality in non-smokers.	We annotate no keyphrases for this category.

Figure 2: *Left:* one of the 16 abstracts used in this study. *Right:* the categories employed, as well as facts and keyphrases associated with the abstract. Keyphrases are highlighted in the abstract.

To assess participants' level of domain knowledge, we use a questionnaire for each topic. For the topics *Breast cancer* and *Vaccination,* we use online quizzes approved by University of Rochester medical reviewers [2, 5]. For the topic *Cardiovascular diseases*, we use the sections *Epidemiology* and *Risk factors* of Bergman et al. [6]. For the topic *Nutrition,* we use the first section on expert nutrition advice of the UCL Nutrition Knowledge Questionnaire [32].

Table 2: Values used for annotating facts.

Fact value	Explanation
A	The fact is entirely captured in the text, omitting only insignificant details.
B	The essence of the fact is captured in the text, but a significant amount of detail was omitted.
C	The fact is not, or only insufficiently, captured in the text.
D	The fact contradicts the original text.

2.3 Annotating and tracking information

Studying information distortion requires quantifying the completeness and truthfulness of a summary: which facts from the original text are present in a given summary, and which facts have been omitted or even contradicted?

To address this challenge, we exploit the fact that medical abstracts are highly structured: key information can be attributed to a few main categories (e.g., *Participants* or *Intervention*), which may be further decomposed into multiple lower-level ones [45]. For example, *Participants* may be decomposed into subcategories like *Age, Sex,* and *Condition.* We develop two methods for tracking information related to these categories, which we explain in detail.

We annotate **keyphrases** (e.g., "mortality") in the abstracts with the categories they belong to (e.g., *Outcomes*). The phrases are then matched in subsequent summaries. Tracking keyphrases is computationally simple, but runs the risk of low recall: a fact may be expressed in a summary, but in words different from those in the input text. In other words, semantic completeness does not require lexical completeness. Moreover, it may be the case that the keyphrase is present, but the actual information is lost, e.g., if the summary contradicts the source text but shares keyphrases with it.

We also extract **facts,** short sentences stating the essential information a text conveys about a specific category. For example, for *Participants/Sample size,* a fact may be *"There were 229,119 men and 173,141 women among the participants"*. Facts of a given category may partially overlap with others; e.g., in the above example, it is mentioned that the study was performed with men and women, information which may also be present in the fact for the *Participants/Sex* category. We present a full abstract with annotated keyphrases and facts in Fig. 2. On the right-hand side, we show the hierarchical categories for both keyphrases and facts, inspired by the categories proposed in Nye et al. [45]. Notice that *Conclusion/General* is a category for which we have only facts but not keyphrases. This is the case as the conclusion often involves keyphrases from various categories; e.g., the statement *"Coffee consumption was inversely associated with total and cause-specific mortality in nonsmokers"* contains keyphrases about the intervention (*coffee*) and the outcome (*cause-specific; mortality*).

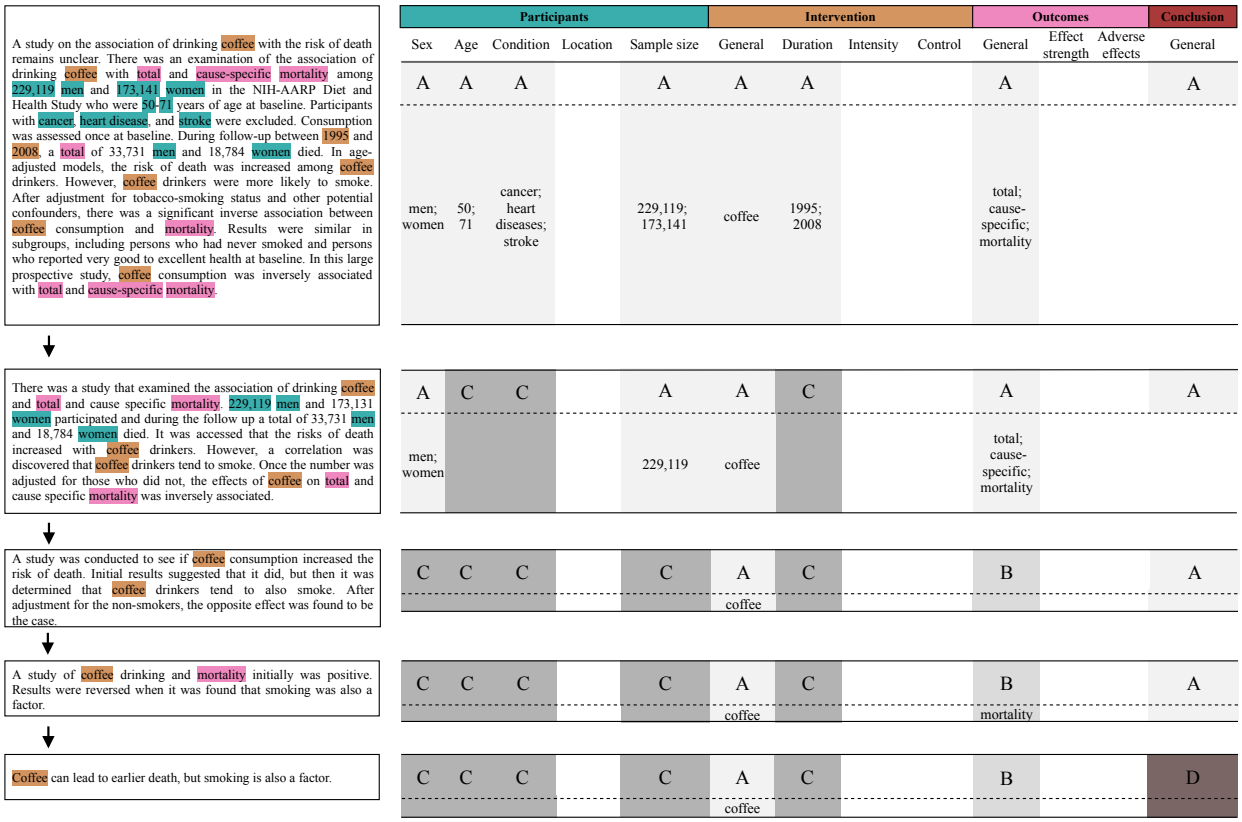


Figure 3: *Left:* a real summarization cascade obtained from crowd workers. *Right:* the annotated facts and the matched keyphrases for each of the summaries. Fact values (*A, B, C, D*) are explained in Table 2.

Tracking facts along the cascade is more complex and ambiguous than tracking keyphrases. For each summary, we assign each fact a **fact value** (*A, B, C,* or *D*). The meaning of these values is defined in Table 2. We attributed values to facts via crowdsourcing, instructing crowd workers as follows:

> You will be given several statements (around 9) and a series of short texts (of different sizes) related to medicine. Your tasks are these:
>
> (1) Read the statement and the texts carefully.
> (2) Judge how well the statement is captured by the different texts, choosing between *A, B, C,* and *D* (as explained in Table 2).
>
> The above will be repeated for several statements (the texts will be the same). Quality checks will be performed on the answers.

Attributing values to the facts is subjective: it is often not trivial to decide between *A* and *B*, or between *B* and *D*; e.g., the information that the study involved 1,067 participants can be distorted along the cascade by stating that the study involved a thousand participants. Is this fact partially preserved, as the order of magnitude is still conveyed, or does this contradict the initial fact, since $1{,}067 \neq 1{,}000$?

We take several steps to ensure annotation quality: *(i)* We add a qualification test where we explain the task thoroughly, gave examples, and assessed workers' ability to understand the subtleties of the task. *(ii)* We show texts of different sizes at once to ensure that workers could compare what a piece of information would look like in several lengths, making the annotation of a given fact consistent across several summaries. *(iii)* We introduce quality checks (facts that are wrong for all statements), allowing us to filter workers who failed often. *(iv)* Lastly, we assign three workers per fact and manually review all facts, giving emphasis to those where not all three workers had annotated a given fact with the exact same value (33% of all cases).

Example. We present a summary cascade on the left-hand side of Fig. 3. The summaries refer to the abstract from Fig. 2. The findings are nuanced: coffee is associated with an increased risk of death; yet, when accounting for confounding variables, such as smoking, the association is reversed. In the cascade, as the summaries become shorter, this subtlety becomes increasingly difficult to grasp. In the last summary, we eventually even arrive at a contradiction.

Annotated keyphrases and extracted facts are shown on the right-hand side of Fig. 3. Firstly, notice that the sudden contradiction we just described is captured, as the label *D* was assigned to the *Conclusion/General* fact in the last summary. Another interesting transition here is how the *Outcomes/General* fact shifts from *A* to *B* in the third summary: both previous summaries indicated that the summary measured total and cause-specific mortality, whereas the third one simply indicates that it measured the risk of death, which is less complete.

To see the limitations of tracking keyphrases and how they are overcome by tracking facts, consider the second summary, which

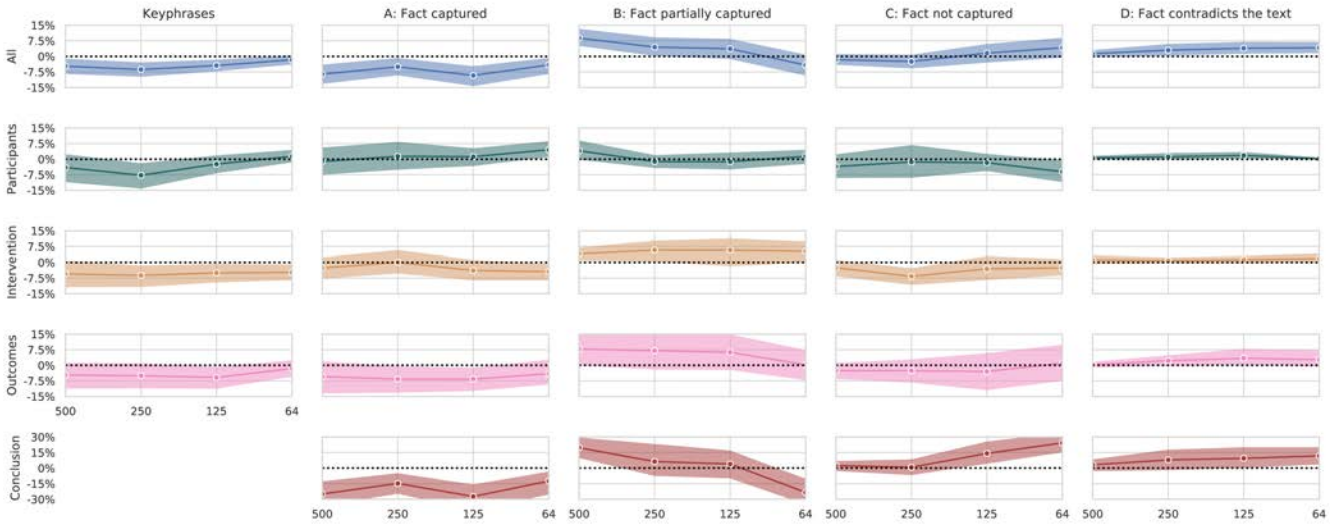


Figure 4: Differences in percentage of preserved keyphrases (column 1) and fact values (columns 2–5) between cascading and control summaries for fact categories (rows) and hops (x-axes), with bootstrapped 95% confidence intervals.

contains a typo: it says that the study was performed in 173,131 women instead of 173,141. Also, the worker used the expression "risk of death", instead of "mortality". These examples show that keyphrases can be difficult to track when there are typos or when expressions are rephrased.

3 RESULTS

3.1 RQ1: Strength of the telephone effect

In our first analysis, we compare the distortion of information in the cascading and control conditions.

Differences across hops. We calculate the percentage of keyphrases in the summaries of a given hop and the percentage of facts with a given value for all papers in the cascading and control settings. We depict the difference between the two settings in percentage points in Fig. 4. Values above 0% mean that the percentage of keyphrases, or of facts of a given value, is higher in the cascading summaries. We observe several interesting patterns. Cascading summaries preserve fewer keyphrases in all categories at all hops. This difference decreases in the last hops, perhaps because they require users to adopt more abstractive summarization strategies in both settings. The percentage of facts labeled as *A* is lower for cascading than for control summaries. This effect is particularly strong for the *Outcomes* and *Conclusion* categories. In the latter, the difference is greater than 15%. The difference in the percentage of facts labeled as *D* has an increasing trend. Here too, the effect is strongest for the *Outcomes* and *Conclusion* categories. Lastly, facts labeled as either *B* or *C* do not present a clear trend for the *Participants, Intervention,* and *Outcomes* categories. For the *Conclusion* category, however, the difference in *B*-valued facts decreases along the hops, while the percentage of *C*-valued facts increases. Overall, these differences suggest that the "telephone effect" impacts the quality of summaries significantly, particularly harming the essential facts—the outcomes and conclusion of a study.

Category distribution. We also analyze how the distribution of categories differs for the distinct settings (Table 3). Consider the percentage of facts labeled as *A* that belong to the *Participants* and *Conclusion* categories. For cascading summaries, the percentage of *Participants* facts consistently stays above 29%, while this percentage drops significantly to 14% for control summaries. For the *Conclusion* category, we have the inverse: the percentage of *A* statements in the *Conclusion* category decreases in the cascading group and remains stable (above 10%) in the control. This suggests that the type of message that ends up getting spread differs between the cascading and control summaries: cascading summaries seem to "remember" less pertinent facts about participants, whereas control summaries preserve more crucial facts about study conclusions.

Table 3: Distribution over categories (*Participants, Intervention, Outcomes, Conclusion*) for fact values (*A, B, C, D*). Results for control setting in bold.

Tgt. len.		500	250	125	64
A	**P**	38%/**35%**	36%/**32%**	32%/**25%**	29%/**14%**
	I	34%/**33%**	36%/**34%**	41%/**37%**	50%/**52%**
	O	17%/**18%**	16%/**18%**	18%/**20%**	17%/**21%**
	C	10%/**14%**	12%/**16%**	8%/**18%**	4%/**13%**
B	**P**	23%/**23%**	17%/**23%**	14%/**19%**	13%/**11%**
	I	24%/**26%**	27%/**21%**	25%/**20%**	24%/**15%**
	O	35%/**39%**	37%/**37%**	38%/**37%**	39%/**36%**
	C	18%/**12%**	19%/**19%**	23%/**25%**	24%/**38%**
C	**P**	46%/**47%**	48%/**46%**	47%/**47%**	45%/**48%**
	I	24%/**24%**	27%/**29%**	28%/**29%**	28%/**29%**
	O	27%/**28%**	23%/**23%**	20%/**21%**	21%/**21%**
	C	2%/**1%**	3%/**2%**	5%/**2%**	6%/**2%**
D	**P**	22%/**20%**	21%/**20%**	18%/**0%**	2%/**0%**
	I	26%/**20%**	15%/**15%**	13%/**17%**	15%/**0%**
	O	19%/**27%**	24%/**20%**	24%/**0%**	24%/**18%**
	C	33%/**33%**	41%/**40%**	45%/**83%**	59%/**82%**

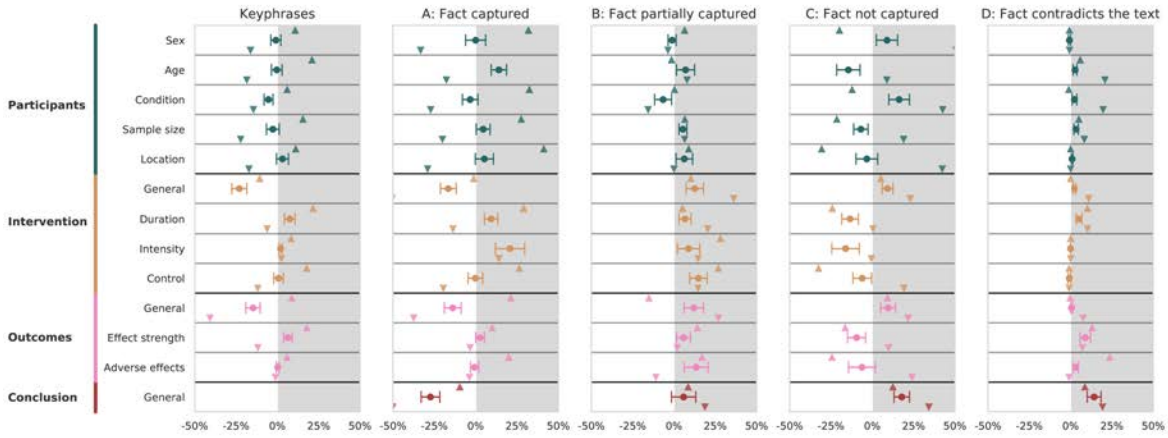


Figure 5: Differences in the percentage of facts and keyphrases between cascading and control summaries for fine-grained categories (averaged over all target lengths). On the positive side of the x-axis (in gray), the percentage of facts/keyphrases is higher in the cascading scenario. Each category is associated with three values (with bootstrapped 95% CIs): The circle ∘ represents the actual difference between the cascading and control settings. The upward △ (downward ▽) triangle represents the difference between the successor of the best (worst) cascading summary from the previous hop and control.

A closer look. We now zoom into the fine-grained categories to drill deeper with regard to the differences between the cascading and control settings. We study the difference between the percentage of keyphrases and facts, averaged across hops, as depicted by the circles (∘) in Fig. 5. We find that the differences are largest in what can be considered the most important facts. For example, the categories *Conclusion/General*, *Intervention/General,* and *Outcomes/General* have the biggest difference for facts labeled as *A* and for keyphrases (recall that there are no conclusion-related keyphrases). The percentage of facts labeled as *C* and *D* for these categories is also higher in the cascading scenario. This is interesting as one might expect that the telephone effect should impact peripheral categories such as *Duration* the most, whereas the opposite is the case. Analyzing the telephone effect from this perspective, we find that iterative summarization creates a "tunnel vision" effect, where less important information often moves to the fore when multiple summarization steps are involved.

The bright side of the telephone effect. Cascading summaries clearly have the disadvantage of propagating errors. However, they also have a potential advantage: given that one person did an excellent job at summarizing a text, it may be that the text they created is more straightforward for the next person to understand and thus to summarize. To determine the existence and strength of such a hypothetical positive effect, we proceed as follows. For each paper and each hop in the cascading setting, we consider only the summary whose source was the "best" of all summaries in the previous hop, where we consider the best summary to be the one with the largest number of *A*-labeled facts. We then compare the percentage of facts and keyphrases per fine-grained category between these summaries and the ones in the control setting as previously done. We show the values for this comparison for each fine-grained category as an upward triangle (△) in Fig. 5. Similarly, we also show the values for an analogous comparison where the source summary was the "worst" summary, depicted as a downward triangle (▽).

By inspecting facts labeled as *A*, we find that the "best-ancestor" scenario preserves facts better or similarly to the control setting (except for the *Conclusion/General* and *Intervention/General*category). This shows that a summary can, in fact, be a better reference text than the original abstract itself. A potential cause for this improvement might be that, although the telephone setting causes distortions, it also lowers the cognitive load, as summarizing jargon-filled 2000-character abstract is significantly harder than summarizing a 500-character text that was written by someone whose knowledge on the subject may be closer to one's own. On the other hand, the "worst-ancestor" scenario highlights the dangers of the telephone effect, where one sloppy summarization may distort the facts present in the text and harm subsequent summaries.

3.2 RQ2: Hop-to-hop information persistence

Next, we calculate the *conditional chance of survival* for keyphrases and facts across hops. That is, given that a fact or a keyphrase is present in a given hop k, what is the chance that it will also be present in hop $k+1$? In the case of keyphrases, for a given summarization process and category, we define the survival probability as the percentage of keyphrases that continue to exist in the summary, out of those that already exist in the reference text. For instance, if there are three keyphrases in the *Outcomes* category in hop 1, and two remain at hop 2, the probability of survival for this category from hop 1 to hop 2 is 2/3. In the case of facts, we consider the number of statements of a given category that are fully (fact value *A*) or partially (*B*) preserved in the source text and continue to be so in the resulting summary. The mean values along with a linear regression line for different categories can be seen in the first column

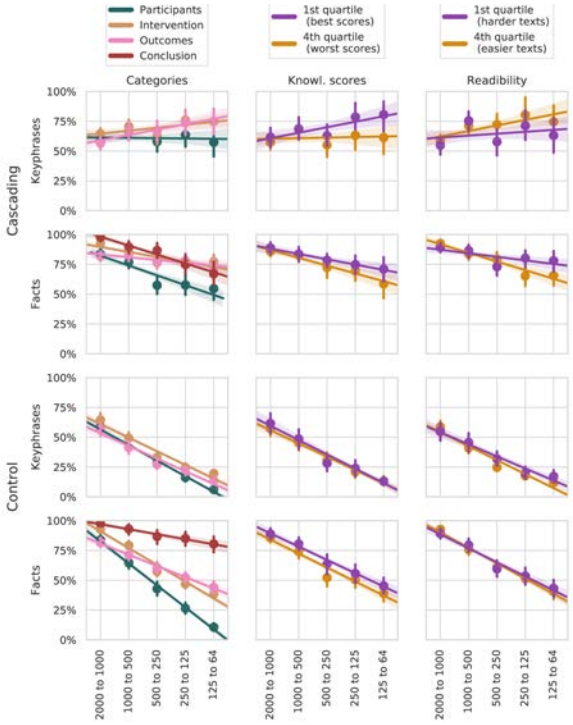


Figure 6: Probabilities of surviving one more hop for keyphrases/facts, alongside regression lines. Each row represents a combination of keyphrases/facts and a setting (cascading/control). In the first column, we compare the survival chance across different categories. In the other two, we compare it for different values of workers' questionnaire scores and of the readability index of the original abstract.

of Fig. 6. Interestingly, we observe that, while by construction the target length of texts decreases exponentially hop by hop, survival probabilities decrease or increase only linearly.

Cascading vs. control. We continue to inspect the first column of Fig. 6. The probability of survival of keyphrases increases for cascading summaries across hops for all coarse categories. This makes sense intuitively: if a keyphrase was selected for a summary, it is likely that it is relevant and thus will be selected again. In the control scenario, keyphrases always exist in the reference text (because it is the original abstract), so the probability of survival decreases as target lengths decrease. Analyzing facts, however, we find a significant difference between our experimental conditions: the probability of survival for facts in the control *and* cascading settings decreases across hops, implying that different dynamics govern distortions associated with lexical vs. semantic elements of a text. Whereas survival probabilities increase for keyphrases in cascading summaries, they decrease for facts. We conjecture that some salient keyphrases have an inherent "fitness" for survival, but that this is less true for abstract facts, which may be more or less salient, and thus "fit", in their concrete surface forms.

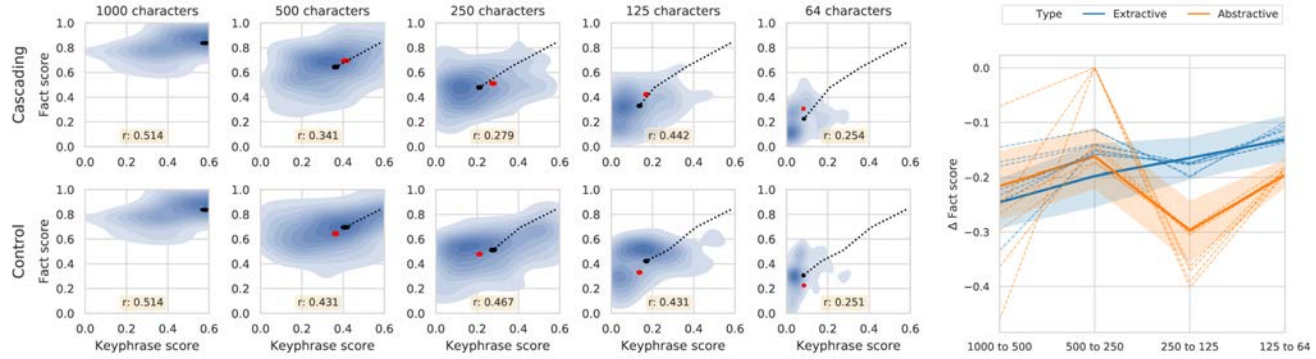
Categories. Moreover, we can also find differences in the dynamics of distortion across different categories. We observe that categories have different survival probabilities, especially for facts. For control summaries, the survival chance of *Conclusion* facts is higher than for the facts of other categories. Also, for *Conclusion*, the survival chance in cascading summaries is lower than for control summaries. This suggests that using the original abstract as a reference makes it more likely that the reader will understand the conclusion. Notice that this is different from what we observed in Sec. 3.1, as there, it could be the case that the conclusion prevailed in the control setting simply because it got lost in the cascade. Here we see that, even when the conclusion fact is present in a given summary in the cascading setting, it is less likely that it will survive.

Knowledge questionnaire. We consider the influence of the worker's knowledge questionnaire score (Sec. 2.2) on the level of distortion. At each hop, we rank the texts according to the score of the workers who summarized them on the questionnaire of the topic associated with the original abstract, and then compare the survival chance of facts and keyphrases of two extremes: the texts summarized by workers whose score is in the first quartile (who performed the best in the test) and the texts summarized by workers whose score is in the fourth quartile (who performed the worst). This is shown in the second column of Fig. 6. We use Chow's test [13] to assess whether the coefficients of the regression are significantly different, finding that, in the control setting, workers who scored better in the test lost significantly fewer facts than those who did poorly ($p < 0.01$). For cascading summaries, keyphrases survive more for workers that scored well in the text ($p < 0.05$), whereas the difference in the survival of facts is not significant. A hypothesis for this disparity is that in-depth knowledge on the subject is essential to read the original abstract (as in the control case), but that, once the text has been summarized by someone else (who may not possess that knowledge), the noise is already introduced, and the effect of the level of knowledge fades away.

Readability. We also consider the influence of the number of difficult words *in the original abstract* on the survival of facts and keyphrases. We order texts according to the percentage of words that *(i)* have more than one syllable and *(ii)* are not in a list of the most frequent English words (as done for readability metrics such as the Dale-Chall Readability Score [12]). We compare the cascades of the four most readable abstracts (first quartile) with the four least readable (fourth quartile). This is depicted in the third column of Fig. 6. We find a different effect to what we observed when inspecting the knowledge questionnaires: facts in more readable abstracts do not a have a significantly different of survival in any setting according to Chow's test. Keyphrases survive more for more readable abstracts in the cascading setting only ($p < 0.05$). Altogether, these results suggest that the telephone effect alters not only the facts and keyphrases of the summaries, but also how other factors influence the distortion processes over these facts and keyphrases.

3.3 RQ3: Extractive vs. abstractive strategies

Another aspect that may influence distortion effects in information cascades is the summarization strategy employed. Here we compare *abstractive* and *extractive* summarization strategies. To be able to do so, we have to distinguish abstractive and extractive summaries, as well as successful and unsuccessful ones. To this end, we define



(a) (b)

Figure 7: (a) Kernel density estimates of summaries according to their fact scores and keyphrase scores for different hops in cascading and control settings. Ellipses mark centroids (height and width: 95% CIs; black: experimental setting in question; red: other experimental setting). (b) Loss of fact score per summarization step for extractive vs. abstractive texts in the cascading setting. Lines correspond to multiple combinations of $\alpha \in [0.4, 0.7]$ and $\beta \in [0.0, 0.15]$ (95% CIs plotted for $\alpha = 0.3, \beta = 0.05$).

keyphrase scores and *fact scores* for each summary. The keyphrase score is defined as the percentage of keyphrases that the summary retained from the original abstract, whereas for fact scores we take a weighted average of the percentages of *A*- and *B*-valued facts retained from the original abstract (assigning weight 1 to *A*-valued facts and weight 0.5 to *B*-valued facts). With these definitions, we can plot summaries as points in a two-dimensional plane. We depict the kernel density estimate of these points in Fig. 7a. In this figure, we consider points with higher fact scores (y-axis) to be better summaries, and points with higher keyphrase scores (x-axis) to be more extractive. This way, the figure allows us to visually track the quality and extractiveness of summaries.

In each plot of Fig. 7a, we display a black ellipsis marking the centroid of the summary cloud (the height and width of the ellipsis represent 95% confidence intervals), and a red ellipsis marking the centroid for the other experimental setting. We also plot the hop-by-hop trajectory of the centroid as a dotted line. Notice the diagonal trajectory: summaries go from better and more extractive to worse and less extractive. Comparing the centroids for the cascading and the control settings, we also see that the centroid of the cascading setting always lies southwest of that of the control setting: cascading summaries are worse as well as less extractive.

In all scenarios, the fact scores are correlated with keyphrase scores (Pearson's r between 0.25 and 0.51), as can be seen in Fig. 7a. This makes it particularly hard for us to fairly assess summarization strategies, as it may be that the worst summaries are considered "abstractive" simply because they do not contain keyphrases (while also not containing the facts). Thus, our analysis has to condition the comparison in a way that we compare summaries that are similar in "goodness", but that differ in the level of extractiveness (which we capture using the defined scores).

We proceed as follows: for each hop and abstract in the cascading setting, we match summaries in pairs. Summaries are only matched if *(i)* the relative difference in their keyphrase score is *greater* than α and *(ii)* the relative difference in their fact score is *less* than β. We greedily match each summary once. Within each pair, we consider the summary with the higher keyphrase score to be more extractive (while of similar quality). We then compare how the fact score of each summary in the pair decreases with the subsequent summarization. The idea is that if extractive texts lose more of their fact score than abstractive texts, this is evidence that extractive summarization is less effective (and vice versa).

We show the results for each of summarization step in Fig. 7b for several values of α and β. The plot suggests that even when we account for the correlation between keyphrases and facts, we observe that extractive summaries are beneficial to latter hops, but similar in the first ones. In our data, thus, although abstracts were often full of jargon and technicalities, trying to change important lexical parts of the text (which we tagged as keyphrases) seems to be detrimental Our analysis here is limited as we are considering the survival of keyphrases to be a proxy for extractiveness, which may not necessarily hold in all cases.

4 DISCUSSION

4.1 Summary of findings and implications

In this paper, we propose an experimental framework for studying message distortion in information cascades and assess the diffusion of information from selected medical abstracts. Our analyses suggest that information cascades, as captured by iterative summarizations, distort the examined medical texts lexically and semantically. Facts and keyphrases frequently are not captured or even contradict the original text. The "telephone effect" impacts the most essential information the most, in particular the conclusion of abstracts. Overall, the content of the message after cascading summarization differs considerably from the content after direct summarization.

The telephone effect is, however, nuanced. Firstly, it is not necessarily bad: good summaries may serve as stepping stones toward better downstream summaries. Moreover, cascading summaries attenuate the impact of the complexity of the original text as well as the impact of users' topic-specific knowledge. Our findings suggest

that influential platforms or users may have a disparate impact on the quality of information being spread, resonating with the narrative that the rise of online social networks, where messages written by anyone can have far-reaching impact, has diminished the quality of the information [26].

We also investigated the success of abstractive vs. extractive summarization in information cascades. Even when accounting for the correlation between keyphrases and facts, we still find that extractive summarization performs better. This insight has implications on how scientists and the press should cooperate to convey research to the general public. Our findings suggest that scientific media coverage should make an effort not to distort the key terms with which authors use in their research.

Lastly, although keyphrases and facts are correlated, the conditional probability of survival of keyphrases increases hop by hop in information cascades, whereas it decreases for facts—a finding that should be taken into consideration when modeling the diffusion of information using keyphrases as proxies for associated facts.

4.2 Related work

Similar methodologies. Previous work has considered similar experimental settings to study distinct phenomena. Mesoudi and Whiten [38] propose a setting similar to our iterative cascades (among others) to study cultural transmission and evolution. Moussaïd et al. [40] use the equivalent of what we call a one-hop iterative cascade to study the propagation of risk information. A similar analysis of how textual content mutates was done in an observational setting by Adamic et al [3]. The researchers studied how memes evolve through Facebook analyzing the adaptation of textual features, as well as their impact on the propagation of the meme.

Word of mouth and customer behavior. Word of mouth is an important phenomenon driving customer behavior. Prior work has studied how the level of customer satisfaction impacts engagement in word-of-mouth information diffusion [4] and how effects of word of mouth together with other factors impact customer persuasion [24]. Word-of-mouth effects have also been analyzed in a networked framework to demonstrate different roles played by weak and strong social ties and the relational properties of homophily on referral behavior [23].

***Bona fide* vs. intentional information distortion.** Agents involved in word of mouth distort the information even given best intents. In contrast, a rich body of work seeks to detect, model, and prevent intentional message distortion, e.g., the dissemination of misinformation, and fake news [11, 33, 54]. Our experimental setting abstracts away from the complex social media landscape with heterogeneous agents including bots and trolls [16, 55], in order to understand the fundamental patterns that govern information distortion. On the middle ground between best intent and intentional distortion, media outlets distort information in more a more nuanced way [39]. In *agenda setting*, more attention is allocated to stories that fit a biased narrative, and in *framing*, the facts are more subtly distorted due to how they are presented.

Telephone and summarization effects. The telephone effect and the summarization effect have been studied from a linguistic standpoint. Breck and Cardie [8] note that facts, events, and opinions appearing in news articles are often known only second- or third-hand. Agents reporting them resort to using two kinds of expression that can filter information: perspective and speech expressions. They propose a learning approach that correctly determines the hierarchical structure of such information filtering expressions emerging due to the telephone effect. More recently, Gligorić et al. [18] found that length constraints cause Twitter users to summarize their content. This process significantly alters linguistic aspects of the text, such as the use of abbreviations, contracted forms, and articles.

Science communication. Media coverage of science has been investigated by communication research. The focus of the field is on scientists' attitude towards the media and on patterns of interactions with journalists and other key players such as news organizations and science information professionals [47, 56]. Evidence suggests that most scientists consider visibility in the media important and responding to journalists a professional attitude that is reinforced by universities and other science organizations. Exaggeration in the news is strongly associated with exaggeration in press releases, and improving the accuracy of academic press releases could represent a key opportunity for reducing misleading health-related news [53].

Summarization and paraphrasing. Finally, the summarization task at the heart of our experimental design is tightly related to the classic natural language processing tasks of paraphrasing [28, 49, 57] and summarization [37, 43]. Prior research introduced the distinction between abstractive and extractive summarization approaches upon which we rely in some of our analyses [27].

4.3 Limitations and future work

Our experimental setup allows us to finely measure and characterize message distortion effects, which would be difficult in observational setups. Still, it remains unclear to what extent our findings generalize to a wider spectrum of real-world information cascades, where three important distinctions are likely to have an impact: *(i)* the sequence-of-laypeople mechanism used here may differ from the way information diffuses in the real world; *(ii)* different kinds of information (e.g., news) may be subject to different dynamics; *(iii)* the assumed *bona fide* scenario may not hold in all cases.

Future work will face the challenge of tracking message distortion in real-world cascades (such as the one described in Sec. 1). By taking advantage of methods and insights developed in this paper, we plan to establish the extent to which our findings reflect the complex universe of social media, thus removing the *bona fide* assumption. Moreover, we will explore whether the observed effects can be modeled mathematically to predict which lexical and semantic units of a text that are likely to be distorted.

ACKNOWLEDGEMENTS

We gratefully acknowledge support from CNPq, Atmosphere, a Google Research Award for Latin America (Manoel Horta Ribeiro), and a Google Faculty Research Award (Robert West). We would also like to thank Lucy Li for helpful discussions.

REFERENCES

[1] 2018. Chinese whispers. https://en.wikipedia.org/w/index.php?title=Chinese_whispers&oldid=865329255 Page Version ID: 865329255.

[2] Levy Adam S and Kim Stump-Sutliff. [n. d.]. Breast Cancer Quiz - Health Encyclopedia - University of Rochester Medical Center. https://web.archive.org/web/20181116150802/https://www.urmc.rochester.edu/encyclopedia/content.aspx?contenttypeid=40&contentid=BreastCancerCancerBrQuiz

[3] Lada A. Adamic, Thomas M. Lento, Eytan Adar, and Pauline C. Ng. 2016. Information Evolution in Social Networks. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM '16)*. ACM, New York, NY, USA, 473–482.

[4] Eugene W. Anderson. 1998. Customer Satisfaction and Word of Mouth. *Journal of Service Research* 1, 1 (Aug. 1998), 5–17.

[5] Lentnek Arnold and Rita Sather. [n. d.]. Immunization Quiz - Health Encyclopedia - University of Rochester Medical Center. https://web.archive.org/web/20190222050418/https://www.urmc.rochester.edu/encyclopedia/content.aspx?contenttypeid=40&contentid=ImmunQuiz

[6] Hannah E. Bergman, Bryce B. Reeve, Richard P. Moser, Sarah Scholl, and William M. P. Klein. 2011. Development of a Comprehensive Heart Disease Knowledge Questionnaire. *American journal of health education / American Alliance for Health, Physical Education, Recreation, and Dance* 42, 2 (March 2011), 74–87.

[7] Archie Bleyer and H. Gilbert Welch. 2012. Effect of three decades of screening mammography on breast-cancer incidence. *New England Journal of Medicine* 367, 21 (2012), 1998–2005.

[8] Eric Breck and Claire Cardie. 2004. Playing the Telephone Game: Determining the Hierarchical Structure of Perspective and Speech Expressions. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)*. Association for Computational Linguistics, Stroudsburg, PA, USA.

[9] Jane E. Brody. 2015. The More We Learn on Nutrition, the More We Ignore. https://well.blogs.nytimes.com/2015/10/12/the-more-we-learn-on-nutrition-the-more-we-ignore/

[10] Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. 2010. Emerging Topic Detection on Twitter Based on Temporal and Social Terms Evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining (MDMKDD '10)*. ACM, New York, NY, USA, 4:1–4:10.

[11] Stefano Cresci, Fabrizio Lillo, Daniele Regoli, Serena Tardelli, and Maurizio Tesconi. 2018. $FAKE: Evidence of Spam and Bot Activity in Stock Microblogs on Twitter. In *Twelfth International AAAI Conference on Web and Social Media*.

[12] Edgar Dale and Jeanne S. Chall. 1948. A Formula for Predicting Readability: Instructions. *Educational Research Bulletin* 27, 2 (1948), 37–54.

[13] Christopher Dougherty. 2016. *Introduction to Econometrics* (fifth edition, new to this edition: ed.). Oxford University Press, Oxford, New York.

[14] Robert M. Entman. 1993. Framing: Toward Clarification of a Fractured Paradigm. *Journal of Communication* 43, 4 (Dec. 1993), 51–58.

[15] Ramón Estruch, Emilio Ros, Jordi Salas-Salvadó, Maria-Isabel Covas, Dolores Corella, Fernando Arós, Enrique Gómez-Gracia, Valentina Ruiz-Gutiérrez, Miquel Fiol, and José Lapetra. 2013. Primary prevention of cardiovascular disease with a Mediterranean diet. *New England Journal of Medicine* 368, 14 (2013), 1279–1290.

[16] Claudia I. Flores-Saviaga, Brian C. Keegan, and Saiph Savage. 2018. Mobilizing the Trump Train: Understanding Collective Action in a Political Trolling Community. In *Twelfth International AAAI Conference on Web and Social Media*.

[17] Neal D. Freedman, Yikyung Park, Christian C. Abnet, Albert R. Hollenbeck, and Rashmi Sinha. 2012. Association of coffee drinking with total and cause-specific mortality. *New England Journal of Medicine* 366, 20 (2012), 1891–1904.

[18] Kristina Gligorić, Ashton Anderson, and Robert West. 2018. How Constraints Affect Content: The Case of Twitter's Switch from 140 to 280 Characters. In *Twelfth International AAAI Conference on Web and Social Media*.

[19] Paul E. Goss, James N. Ingle, José E. Alés-Martínez, Angela M. Cheung, Rowan T. Chlebowski, Jean Wactawski-Wende, Anne McTiernan, John Robbins, Karen C. Johnson, and Lisa W. Martin. 2011. Exemestane for breast-cancer prevention in postmenopausal women. *New England Journal of Medicine* 364, 25 (2011), 2381–2391.

[20] Michael E. Greenberg, Michael H. Lai, Gunter F. Hartel, Christine H. Wichems, Charmaine Gittleson, Jillian Bennet, Gail Dawson, Wilson Hu, Connie Leggio, and Diane Washington. 2009. Response to a monovalent 2009 influenza A (H1N1) vaccine. *New England Journal of Medicine* 361, 25 (2009), 2405–2413.

[21] Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A. Zighed. 2013. Information Diffusion in Online Social Networks: A Survey. *SIGMOD Rec.* 42, 2 (July 2013), 17–28.

[22] Stuart Hall. 2009. Encoding/Decoding. In *Media and Cultural Studies: Keyworks*. John Wiley & Sons. Google-Books-ID: I8dPhB88Sx4C.

[23] Thorsten Hennig-Thurau, Kevin P. Gwinner, Gianfranco Walsh, and Dwayne D. Gremler. 2004. Electronic word-of-mouth via consumer-opinion platforms: What motivates consumers to articulate themselves on the Internet? *Journal of Interactive Marketing* 18, 1 (Jan. 2004), 38–52.

[24] Paul M. Herr, Frank R. Kardes, and John Kim. 1991. Effects of Word-of-Mouth and Product-Attribute Information on Persuasion: An Accessibility-Diagnosticity Perspective. *Journal of Consumer Research* 17, 4 (March 1991), 454–462.

[25] Barbara V. Howard, Linda Van Horn, Judith Hsia, JoAnn E. Manson, Marcia L. Stefanick, Sylvia Wassertheil-Smoller, Lewis H. Kuller, Andrea Z. LaCroix, Robert D. Langer, Norman L. Lasser, Cora E. Lewis, Marian C. Limacher, Karen L. Margolis, W. Jerry Mysiw, Judith K. Ockene, Linda M. Parker, Michael G. Perri, Lawrence Phillips, Ross L. Prentice, John Robbins, Jacques E. Rossouw, Gloria E. Sarto, Irwin J. Schatz, Linda G. Snetselaar, Victor J. Stevens, Lesley F. Tinker, Maurizio Trevisan, Mara Z. Vitolins, Garnet L. Anderson, Annlouise R. Assaf, Tamsen Bassford, Shirley A. A. Beresford, Henry R. Black, Robert L. Brunner, Robert G. Brzyski, Bette Caan, Rowan T. Chlebowski, Margery Gass, Iris Granek, Philip Greenland, Jennifer Hays, David Heber, Gerardo Heiss, Susan L. Hendrix, F. Allan Hubbell, Karen C. Johnson, and Jane Morley Kotchen. 2006. Low-Fat Dietary Pattern and Risk of Cardiovascular Disease: The Women's Health Initiative Randomized Controlled Dietary Modification Trial. *JAMA: Journal of the American Medical Association* 295, 6 (2006), 655–666.

[26] Lee Howell. 2018. Opinion | Only You Can Prevent Digital Wildfires. *The New York Times* (Oct. 2018). https://www.nytimes.com/2013/01/09/opinion/only-you-can-prevent-digital-wildfires.html

[27] Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A Unified Model for Extractive and Abstractive Summarization using Inconsistency Loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 132–141.

[28] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional Neural Network Architectures for Matching Natural Language Sentences. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2042–2050.

[29] W. Philip T. James, Ian D. Caterson, Walmir Coutinho, Nick Finer, Luc F. Van Gaal, Aldo P. Maggioni, Christian Torp-Pedersen, Arya M. Sharma, Gillian M. Shepherd, and Richard A. Rode. 2010. Effect of sibutramine on cardiovascular outcomes in overweight and obese subjects. *New England Journal of Medicine* 363, 10 (2010), 905–917.

[30] Mette Kalager, Marvin Zelen, Frøydis Langmark, and Hans-Olov Adami. 2010. Effect of screening mammography on breast-cancer mortality in Norway. *New England Journal of Medicine* 363, 13 (2010), 1203–1210.

[31] Nicola P. Klein, Joan Bartlett, Ali Rowhani-Rahbar, Bruce Fireman, and Roger Baxter. 2012. Waning protection after fifth dose of acellular pertussis vaccine in children. *New England Journal of Medicine* 367, 11 (2012), 1012–1019.

[32] N. Kliemann, J. Wardle, F. Johnson, and H. Croker. 2016. Reliability and validity of a revised version of the General Nutrition Knowledge Questionnaire. *European Journal of Clinical Nutrition* 70, 10 (Oct. 2016), 1174–1180.

[33] Adam Kucharski. 2016. Post-truth: Study epidemiology of fake news. *Nature* 540 (Dec. 2016), 525.

[34] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. 2009. Meme-tracking and the Dynamics of the News Cycle. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*. ACM, New York, NY, USA, 497–506.

[35] Lauren Lissner, Lluis Serra Majem, Maria Daniel Vaz de Almeida, Christina Berg, Roger Hughes, Geoffrey Cannon, Inga Thorsdottir, John Kearney, Jan-Å Gustafsson, Joseph Rafter, Lbrahim Elmadfa, and Nick Kennedy. 2006. The Women's Health Initiative. What is on trial: nutrition and chronic disease? Or misinterpreted science, media havoc and the sound of silence from peers? *Public Health Nutrition* 9, 2 (April 2006), 269–272.

[36] Kreesten Meldgaard Madsen, Anders Hviid, Mogens Vestergaard, Diana Schendel, Jan Wohlfahrt, Poul Thorsen, Jørn Olsen, and Mads Melbye. 2002. A population-based study of measles, mumps, and rubella vaccination and autism. *New England Journal of Medicine* 347, 19 (2002), 1477–1482.

[37] Inderjeet Mani. 2009. Summarization evaluation: an overview. In *Proceedings of the NTCIR Workshop*, Vol. 2.

[38] Mesoudi Alex and Whiten Andrew. 2008. The multiple roles of cultural transmission experiments in understanding human cultural evolution. *Philosophical Transactions of the Royal Society B: Biological Sciences* 363, 1509 (Nov. 2008), 3489–3501.

[39] Fred Morstatter, Liang Wu, Uraz Yavanoglu, Stephen R. Corman, and Huan Liu. 2018. Identifying Framing Bias in Online News. *Trans. Soc. Comput.* 1, 2 (June 2018), 5:1–5:18.

[40] Mehdi Moussaïd, Henry Brighton, and Wolfgang Gaissmaier. 2015. The amplification of risk in experimental diffusion chains. *Proceedings of the National Academy of Sciences* (April 2015), 201421883.

[41] Dariush Mozaffarian, Saman Fahimi, Gitanjali M. Singh, Renata Micha, Shahab Khatibzadeh, Rebecca E. Engell, Stephen Lim, Goodarz Danaei, Majid Ezzati, and John Powles. 2014. Global sodium consumption and death from cardiovascular causes. *New England Journal of Medicine* 371, 7 (2014), 624–634.

[42] Dariush Mozaffarian, Tao Hao, Eric B. Rimm, Walter C. Willett, and Frank B. Hu. 2011. Changes in diet and lifestyle and long-term weight gain in women and men. *New England Journal of Medicine* 364, 25 (2011), 2392–2404.

[43] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Berlin,

Germany, 280–290.

[44] Steven E. Nissen and Kathy Wolski. 2007. Effect of rosiglitazone on the risk of myocardial infarction and death from cardiovascular causes. *New England Journal of Medicine* 356, 24 (2007), 2457–2471.

[45] Benjamin Nye, Junyi Jessy Li, Roma Patel, Yinfei Yang, Iain Marshall, Ani Nenkova, and Byron Wallace. 2018. A Corpus with Multi-Level Annotations of Patients, Interventions and Outcomes to Support Language Processing for Medical Literature. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 197–207.

[46] Olivia Pagani, Meredith M. Regan, Barbara A. Walley, Gini F. Fleming, Marco Colleoni, István Láng, Henry L. Gomez, Carlo Tondini, Harold J. Burstein, and Edith A. Perez. 2014. Adjuvant exemestane with ovarian suppression in premenopausal breast cancer. *New England Journal of Medicine* 371, 2 (2014), 107–118.

[47] Hans Peter Peters. 2013. Gap between science and media revisited: Scientists as public communicators. *Proceedings of the National Academy of Sciences of the United States of America* 110, Suppl 3 (Aug. 2013), 14102–14109.

[48] Wayne A. Ray, Katherine T. Murray, Kathi Hall, Patrick G. Arbogast, and C. Michael Stein. 2012. Azithromycin and the risk of cardiovascular death. *New England Journal of Medicine* 366, 20 (2012), 1881–1890.

[49] Philip Resnik, Olivia Buzek, Chang Hu, Yakov Kronrod, Alex Quinn, and Benjamin B. Bederson. 2010. Improving Translation via Targeted Paraphrasing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 127–137.

[50] Tiago Rodrigues, Fabrício Benevenuto, Meeyoung Cha, Krishna Gummadi, and Virgílio Almeida. 2011. On Word-of-mouth Based Discovery of the Web. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference (IMC '11)*. ACM, New York, NY, USA, 381–396.

[51] S. Clinical Trials Partnership RTS. 2011. First results of phase 3 trial of RTS, S/AS01 malaria vaccine in African children. *New England Journal of Medicine* 365, 20 (2011), 1863–1875.

[52] Frank M. Sacks, George A. Bray, Vincent J. Carey, Steven R. Smith, Donna H. Ryan, Stephen D. Anton, Katherine McManus, Catherine M. Champagne, Louise M. Bishop, and Nancy Laranjo. 2009. Comparison of weight-loss diets with different compositions of fat, protein, and carbohydrates. *New England Journal of Medicine* 360, 9 (2009), 859–873.

[53] Petroc Sumner, Solveiga Vivian-Griffiths, Jacky Boivin, Andy Williams, Christos A. Venetis, Aimée Davies, Jack Ogden, Leanne Whelan, Bethan Hughes, Bethan Dalton, Fred Boy, and Christopher D. Chambers. 2014. The association between exaggeration in health related science news and academic press releases: retrospective observational study. *BMJ* 349 (Dec. 2014), g7015.

[54] Sebastian Tschiatschek, Adish Singla, Manuel Gomez Rodriguez, Arpit Merchant, and Andreas Krause. 2018. Fake News Detection in Social Networks via Crowd Signals. In *Companion Proceedings of the The Web Conference 2018 (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 517–524.

[55] Onur Varol, Emilio Ferrara, Clayton A. Davis, Filippo Menczer, and Alessandro Flammini. 2017. Online Human-Bot Interactions: Detection, Estimation, and Characterization. In *Eleventh International AAAI Conference on Web and Social Media*.

[56] Michael F. Weigold. 2001. Communicating Science: A Review of the Literature. *Science Communication* 23, 2 (Dec. 2001), 164–193.

[57] Wenpeng Yin and Hinrich Schütze. 2015. Convolutional Neural Network for Paraphrase Identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, 901–911.

Auditing the Partisanship of Google Search Snippets

Desheng Hu
Northeastern University
hu.des@husky.neu.edu

Shan Jiang
Northeastern University
sjiang@ccs.neu.edu

Ronald E. Robertson
Northeastern University
rer@ccs.neu.edu

Christo Wilson
Northeastern University
cbw@ccs.neu.edu

ABSTRACT

The text snippets presented in web search results provide users with a slice of page content that they can quickly scan to help inform their click decisions. However, little is known about how these snippets are generated or how they relate to a user's search query. Motivated by the growing body of evidence suggesting that search engine rankings can influence undecided voters, we conducted an algorithm audit of the political partisanship of Google Search snippets relative to the webpages they are extracted from. To accomplish this, we constructed lexicon of partisan cues to measure partisanship and construct a set of left- and right-leaning search queries. Then, we collected a large dataset of Search Engine Results Pages (SERPs) by running our partisan queries and their autocomplete suggestions on Google Search. After using our lexicon to score the machine-coded partisanship of snippets and webpages, we found that Google Search's snippets generally amplify partisanship, and that this effect is robust across different types of webpages, query topics, and partisan (left- and right-leaning) queries.

CCS CONCEPTS

• **Information systems** → **Summarization**; • **Social and professional topics** → **Political speech**; • **Human-centered computing** → *User interface design.*

KEYWORDS

snippet generation; Google Search; partisan echo chamber; algorithm auditing

ACM Reference Format:
Desheng Hu, Shan Jiang, Ronald E. Robertson, and Christo Wilson. 2019. Auditing the Partisanship of Google Search Snippets. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313654

1 INTRODUCTION

Search engines are one of, if not the most important tool used by people seeking information on the web. Surveys have repeatedly found that people reach for search engines first when they have online information needs [18] and that this happens on a daily basis [19]. These observations are especially true with respect to breaking news, where people report getting news from search engines even more frequently than social media [51, 54]. Surveys and ethnographic studies have also observed that search engines are specifically used for "fact-checking," possibly because search engines are viewed as neutral and trusted sources of information [4, 18, 19, 30, 55, 65].

When conceptualizing the role of search engines in information seeking, it is critical to recognize that they are no longer just intermediaries that transfer users from queries to websites via "10 blue links" [46]. Instead, modern search engines are rich media platforms unto themselves, often presenting links to third-party websites alongside images, videos, maps, algorithmically curated "knowledge," and social media posts [58]. This additional control over how information is presented — beyond ranking and filtering — can further increase the capacity of search engines to shape users' behavior and preferences. For example, recent studies have found that different types of queries often return different types of results, and these different result types (e.g., embedded Twitter results) can surface partisan or low-quality links in highly ranked positions [16, 57, 58].

One particularly important feature found on modern Search Engine Result Pages (SERPs) are *search snippets*. Snippets appear below links to webpages and provide a brief summary of the content of the given page. These summaries provide users with additional context about a webpage, beyond what can be gleaned from their title. Originally, snippets were drawn from HTML `meta description` tags embedded in webpages. Over time, however, search engines have incorporated *document summarization algorithms* [64] to dynamically produce snippets for pages that lack meta-data, as well as combat intentionally misleading meta-data [24, 28, 63].

Snippets are a critical — and, we argue, under-scrutinized — facet of modern search engines. *First*, snippets have a direct influence on the links that users choose to visit. For example, an inaccurate or misleading snippet may cause a user to select a different link. *Second*, even if a user does not click a given link (or *any* of the links on a SERP), they may still read and be influenced by the snippets. In much the same way people scan the titles and opening paragraphs of newspaper articles [25, 27], users may scan the titles and snippets of webpages in a SERP.

In this study, we present the first analysis of *partisan cues* in snippets on Google Search. We define partisan cues as words or phrases that are disproportionately used by a specific political group to signal group membership and frame specific issues. For example, we observe that the phrases "gun rights" and "gun violence" are used by right- and left-leaning US politicians, respectively, to

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313654

engage with their fellow partisans and frame the gun control debate. We focus on partisan cues in SERPs returned for political queries because (1) news-seeking and fact-checking are common and socially-important use cases for Google Search, and (2) behavioral experiments have shown that the political valence of search rankings can have dramatic real-world impacts on voting behavior [22, 23].

To implement our study, we trained a lexicon of partisan unigrams and bigrams drawn from speeches given by US politicians using Natural Language Processing (NLP) techniques [26]. This lexicon serves two purposes. *First*, we manually curated a subset of the bigrams to use as queries on Google Search. In total, we chose 1,050 right- and left-leaning terms, as well as 3,520 names of US politicians, to use as queries. We queried Google Search for these terms and all of their *autocomplete* suggestions between October 13–30, 2018, ultimately producing a dataset of 88,745 SERPs and 541,437 unique webpages that were linked from the SERPs. *Second*, we used the lexicon to score the partisanship of text in our dataset from left- (-1) to right-leaning (1). We then compared the partisanship score of each snippet to the score of the webpage it was drawn from to understand whether Google Search is amplifying, dampening, or even inverting the incidence of partisan cues in snippet text relative to the original document.

Our study makes the following contributions and observations:

- We present the first large-scale analysis of machine-coded partisanship in Google Search snippets, covering 4,570 political queries and their autocomplete suggestions.
- We audit the behavior of Google Search's document summarization algorithm, and find that snippets tend to be drawn from text that is near the beginning of webpages. We further observe that the algorithm leverages visible text and textual meta-data (such as alt-text on images) from webpages.
- Overall, we find that 54–58% of snippets amplify partisanship, depending on the fraction of our lexicon that is used for scoring, i.e., the snippets contain stronger partisan cues on average than the corresponding webpage they were synthesized from. This finding remains consistent across SERPs from left- and right-leaning queries and pages with and without structured meta-data that may influence Google Search's document summarization algorithm [28, 29].
- Surprisingly, we find that 19–24% of snippets have inverse partisanship than the corresponding webpage.
- We identify 31 websites where Google Search consistently produces snippets that differ from the underlying webpages in terms of the machine-coded partisanship, with high statistical significance. These websites include prominent news and social media services.

We believe that it is highly unlikely that Google has intentionally engineered their document summarization algorithm to amplify partisan cues. Instead, a more likely explanation for our findings is that journalistic practice encourages the use of partisan terms and quotes from partisan politicians in the introduction (and meta-data) of articles, which are also the types of text favored by the summarization algorithm. We hope this study will foster additional research and public debate about the role of search snippets in web search, and foster the development of document summarization algorithms that are intentionally designed to align with societal expectations and democratic norms.

Outline. Our study is organized as follows: in § 2 we motivate our study and survey related work. In § 3 we introduce our datasets and partisan scoring metrics. We analyze our dataset in § 4 and conclude with limitations and discussion in § 5.

2 BACKGROUND AND RELATED WORK

Snippets are a ubiquitous and important feature on modern search engines like Google Search. The goal of snippets is to help users assess the relevance of linked documents without having to click through each one [64, 67]. Snippets appear below links in SERPs and typically show one to three lines of text summarizing the linked document. The sentences or fragments that compose a given snippet may be extracted from different parts of source document, including visible text and invisible meta-data such as the HTML `meta description` tag, alt-text on HTML `img` tags, and various Microformat and Microdata structured meta-data languages [28, 29]. Search engines often generate snippets dynamically at query-time to select document text that is relevant to the current user's query [3, 64].

Document Summarization. Automated snippet generation is a variant of extractive summarization (in contrast to abstractive summarization [10]) that has been actively researched for decades. The earliest text summarization algorithms relied on word frequency distributions [45], sentence location [21], and per-sentence scoring techniques [52] to select text for the summary. In 1998, methods that tailored summaries to specific queries were invented [64], which continue to influence the design of modern search engines [5, 68]. More recent research has used machine learning [36, 69] and neural network-based methods [11, 50] to achieve better summarization performance.

Studying Web Search Engines. Prior work has examined various qualitative aspects of snippets. There are studies on whether the length of snippets impact user experience and the informativeness of search results [13, 38], as well as how snippet length should be customized for mobile web search [37]. Collins-Thompson et al. studied the relationship between user experience and the reading levels, and found that dwell-time improved when the reading level of the snippet matched the underlying document [12]. In our context, this suggests that user satisfaction may be maximized when the partisan cues in a snippet accurately reflect the partisanship of the corresponding document.

To our knowledge, no prior studies have examined the partisanship of snippets. Although there are several studies that have *audited* [15, 59] Google Search, the focus of these studies has been examining how Google Search's personalization algorithm impacts the links that appear in SERPs [32, 39, 58], or the partisanship of links in SERPs [16, 57].

The Impact of Snippets. Our study of partisan cues in snippets is motivated by work that examined how people read and interpret the news. *First*, studies have shown that people often do not read news articles in full; instead, they scan the headlines [25, 27]. Eye-tracking studies suggest that the same behavior is true with respect

to snippets, since they attract a major portion of users' attention when they browse SERPs [13, 31], but ultimately users may only click a single link (if any) to view in full. *Second*, news headlines have the power to shape users' perceptions of stories, even if they read the full articles [20, 25]. We hypothesize that the same effect may be true of snippets: the partisan cues highlighted in a snippet may influence users' perceptions of the linked document even if it has a different partisan slant. Furthermore, in cases where a user does not click any links on a SERP, partisan cues in the snippets may still influence the user's perception of the query term itself.

Scoring Partisanship. Identifying ideological bias in text is difficult and many methods have been proposed to address it. Studies on political communication have revealed that subtle differences in word choice can signal an ideological leaning [9, 17, 35]. For example, US Republicans say "death tax" while Democrats say "estate tax," and there are no ideologically neutral alternative phrases [41].

Early approaches to political text analysis typically used human raters to manually code political statements [7, 8, 40] or relied on manually constructed dictionaries of terms to automatically label statements [14, 33, 34, 43, 53, 61, 62]. In this study, we leverage NLP techniques that automatically generate lexicons of partisan terms using politician's public speeches as a convenient, voluminous, and well-labeled (i.e., because the partisanship of the speaker can be determined based on their party affiliation and voting record) training dataset [26, 60]. These techniques obviate the need for tedious human labeling. Using the lexicons, the *machine-coded* partisanship of a document can be quantified by looking at the number and strength of polarized terms in the document [42, 44].

Unlike prior work on partisanship of SERPs, our goal is to examine the partisanship of text rather than of web domains [16, 57]. Thus, existing datasets that score the partisanship of web domains are not pertinent to our study [1, 2, 6, 49, 56].

3 DATASET AND METHODOLOGY

In this section, we present the datasets and methods that we use in our study. *First*, we introduce our lexicon of partisan unigrams and bigrams, including the raw data and methods we used to construct it. We use this lexicon to inform our selection of queries on Google Search, as well as score the partisanship of text. *Second*, we explain the process we used to select queries for Google Search, crawl SERPs and webpages, and extract data from these web documents. *Third*, we present our metrics for computing partisanship and comparing the partisanship between snippets and webpages.

3.1 Lexicon Construction

The basis for our study is a lexicon of unigrams and bigrams[1] that represent *partisan cues*, i.e., each phrase is primarily used by partisan people with a specific ideological stance. We use this lexicon for two purposes: to select ideological queries for Google Search (see § 3.2) and to score the partisanship of text (see § 3.3).

We use a method developed by Gentzkow and Shapiro to construct our lexicon [26]. In brief, this method counts the frequency of n-grams in documents that have reliable partisan labels (e.g., right and left), and then scores each n-gram based on the relative, normalized frequency of its use by individuals on each side of the partisan divide.

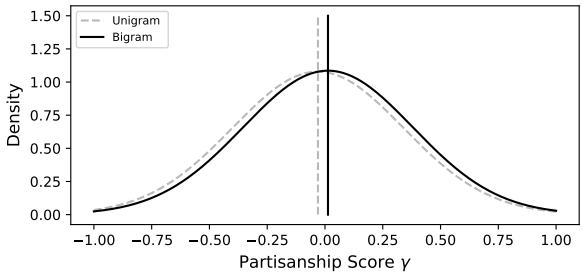


Figure 1: Distribution of partisanship scores γ of unigrams and bigrams in our Vote Smart lexicon.

For our study, we crawled a corpus of documents from votesmart.org, which is a website that catalogues information about politicians, on August 11, 2018. We chose Vote Smart because prior work has shown that its data is useful for understanding politicized language [66]. Specifically, we gathered transcripts of speeches and letters from 20 Democrats and 19 Republicans, including Donald Trump, Hillary Clinton, all congressional leaders from the US House and Senate, and several other prominent US congressional representatives. We selected this set of politicians because they generate a large volume of documents due to their public presence, and as leaders they tend to set the agendas for their respective parties.

Before constructing our lexicon, we balanced and cleaned the corpus. *First*, we chose to use documents dated between January 2008 and August 2018, as this covered the current and former US presidential administrations. We found that using older documents polluted the final lexicon with phrases and issues that were no longer salient in 2018, meaning they did not return relevant results when queried on Google Search. *Second*, we randomly downsampled the documents each year to ensure that each side had an equal number of total documents. *Third*, we cleaned (e.g., case-folding, lemmatization, etc.) and tokenized the corpus using NLTK.[2]

Next, we extracted all unigrams and bigrams from the cleaned corpus and scored them. We computed the probability of each unigram and bigram g appearing in Republican- and Democrat-authored text as $Pr_R(g)$ and $\Pr_D(g)$, respectively. We then computed the partisan bias γ of each g as

$$\gamma(g) = \frac{Pr_R(g) - Pr_D(g)}{Pr_R(g) + Pr_D(g)}. \quad (1)$$

γ scores range from $[-1, 1]$, with 1 (-1) indicating that a phrase is used exclusively by Republicans (Democrats).

Finally, we obtained our lexicon by filtering out unigrams and bigrams that appear <50 times, as these are unlikely to be recognized by people as partisan. After filtering, our lexicon contains 69,901 total terms, with 13,464 unigrams and 56,437 bigrams. Example terms and their scores include: "LGBT" (-0.92), "climate crisis" (-0.91), "equal pay" (-0.90), "gun safety" (-0.85), "burdensome regulation"

[1] We do not consider n-grams with $n > 2$ in this study because we believe that the majority of commonly used partisan cues can be expressed in one or two words.

[2] NLTK: www.nltk.org/

Table 1: Example root queries organized by topic. Queries are color coded as right- and left-leaning by partisanship score.

Topic	Root Queries
Gun	gun control, conceal weapon, gun rights, traffic firearm, gun trace, illegal gun, crime gun, gun traffic, gun dealer, gun lobby, gun safety, gun violence
Election	military voter, secret ballot, voter assistance, ballot access, voter suppression, election act, election infrastructure, election security, eligible voter, secure election, interference 2016 election
President	president health, executive overreach, pipeline president, mandate president, assure president, president unconstitutional, truth president, border president, president hillary, initiative president, president emergency, president assad
Pregnancy	elective abortion, largest abortion, abortion provider, pay abortion, perform abortion, abortion coverage, abortion clinic, fund abortion, unintended pregnancy, emergency contraception, access contraception
Tax	territorial tax system, flat tax, first 24000 tax free, tax mandate, repeal death tax, burdensome taxes, tax scam, tax giveaway, tuition tax, manufacture tax, tax wealthiest, offshore tax haven
American	burden american, regulation american, american conservative, american strength, american liberty, american patriot, american hostage, american spirit, eligible american, american caucus, wealthiest american, richest american, muslim american, american democracy, reignite american, american privacy
Statutes	scrub act, retirement act, reins act, act unilaterally, raise the wage act, honest ads act, undermine affordable care act, equality act, real id act, refinance act, second chance act, pay act
Medicare	raid medicare, medicare trustee, care entitlement, higher copays, medicare advantage, medicaid patient, medicare negotiate, affordable child, cost prescription, cut medicaid, improve affordable, benefit affordable
LGBT	lgbt rights, lgbt american, gay bisexual, lesbian gay, lgbt people, lgbt individual, lgbt community, discrimination lgbt
Obamacare	obamacare mandate, obamacare premium, obamacare plan, problem obamacare, failure obamacare, obamacare employer, obamacare break, obamacare disaster

(0.81), "illegal immigrant" (0.86), "bureaucrat" (0.89), "obamacare" (0.91), and "death tax" (0.98).

Figure 1 shows the score distribution for terms in the lexicon. As expected, the scores are normally distributed, with most terms having low partisanship. The mean score for unigrams and bigrams are -0.03 and 0.01, respectively, demonstrating that the final lexicon is well-balanced.

3.2 Data Collection and Extraction

The next step in our methodology is constructing a dataset of snippets from Google Search SERPs along with the corresponding webpages. Constructing this dataset involved choosing the queries for Google Search, crawling, and finally extracting data from the resulting webpages.

Query Selection. Our ultimate goal in this study is to compare the prevalence of partisan cues in Google Search snippets to their corresponding webpages. To achieve this, we need a large sample of politically-relevant SERPs from Google Search, which necessitates having a set of politically-relevant queries.

We built our set of *root queries* from two sources: (1) the names of US politicians and (2) partisan bigrams from our lexicon. For the former, we gathered the names of 3,520 US politicians, including prominent members of the current and former presidential administrations, members of the US House and Senate, and state governors. For the latter, we had two independent human labelers[3] examine all bigrams in the lexicon with scores < -0.5 or > 0.5 and manually curate terms that were specific enough to be valid queries on Google Search. For example, the bigrams "spend percentage" and "defend proud" both have a score of 1 in the lexicon (i.e., exclusive use by Republicans), but the labelers felt that these bigrams (and others like them) were not useful as queries, since they were unlikely to return results that linked to news and political webpages. Additionally, the labelers expanded bigrams in cases where additional words were necessary to transform them into valid queries. For example, the labelers expanded the bigrams "sanctity life" and "operation choke" (both with a score of 1) to "sanctity of life" and "operation choke point". Because this labeling task is highly subjective, we accepted all of the labelers' suggestions and then randomly downsampled to produce an ideologically balanced set of 1,050 partisan queries.

Clustering. To facilitate analysis of how partisan cues vary by topic, we clustered the 1,050 partisan root queries using a word-embedding technique [48]. Specifically, we used the pre-trained word vector *wiki-news-300d-1M.vec* [47] to map each root query into a vector space. Then we averaged the vectors corresponding to the words in each root query and clustered the mean vectors using k-means. The authors manually examined all 41 resulting clusters and selected 24 that were well-formed, i.e., all of the included root queries were related to a single, well-defined topic. Table 1 shows a subset of these topical clusters and the queries they contained.

Crawling and Query Expansion. We queried Google Search for all 4,570 root queries between October 13–30, 2018. We also queried Google Search for all of the *autocomplete suggestions* for each root query. This is a useful technique for expanding our root queries because the suggestions are drawn from real user queries, i.e., the suggestions provide a window into an more ecologically-valid sample of queries [57, 58]. Although some of the autocomplete suggestions may not be politically-relevant, the corresponding SERPs have minimal impact our analysis because they typically do not contain terms from our lexicon. In total, we ran 88,745 queries and collected an equal number of SERPs from Google Search.

Data Extraction. Since our focus is on the relationship between search snippets and webpage content, we parsed all of the SERPs in our dataset to retrieve snippets and their associated webpage links. We filtered out links in the SERPs that did not have snippets, linked to back to *.google.com, or were part of a non-standard *component* (e.g., maps, videos, images, etc.) [57, 58]. Next, we crawled all the links. Finally, for each (snippet, webpage) tuple, we filtered out cases were the webpage contained <300 characters, which removed 114,020 tuples from our dataset. Ultimately, our dataset contained 820,795 (snippet, webpage) tuples.

In most of our analysis in § 4 we focus on the *page text* in webpages. We define page text as the string content that exists between HTML tags in the body of the HTML document. Page

[3]Both labelers were US citizens by birth and were familiar with US politics.

Table 2: Lexicon size and number of (snippet, webpage) pairs that are scorable under different thresholds.

Threshold	Unigrams	Bigrams	Scorable Pairs Count	Scorable Pairs %
5%	638	2692	43311	5.28
10%	1276	5378	137067	16.70
15%	1914	8076	272745	33.23

text excludes everything in the document head, HTML tags, tag attributes, JavaScript, and Cascading Style Sheets.

3.3 Scoring Snippets and Webpages

The next step in our methodology is computing a partisan bias score for each snippet and webpage using our lexicon. Let U and B be the sets of unigrams and bigrams in our lexicon, and $\gamma(g)$ is the partisan score of n-gram g. For a given sample of text (either a snippet or webpage) produced in response to a given query, we compute its partisan bias score Γ using the following steps:

(1) Use NLTK to clean and tokenize the text and the query, producing lists of tokens T and Q, respectively.
(2) Compute $T' = \{\forall t \in T \mid t \notin Q\}$, i.e., remove all of the tokens $q \in Q$ from T. This step ensures fairness: 1,050 of our queries are explicitly partisan, and query terms are highly likely to appear in the corresponding snippets and webpages. It would be unfair to explicitly query for partisan words then use those same words to score the resulting samples.
(3) Compute $S_B = \{\forall t \in T' \mid t \in B\}$, i.e., the list of bigrams in T'. Compute $S_U = \{\forall t \in (T' - S_B) \mid t \in U\}$, i.e., list of unigrams in T' after the bigrams have been removed (to avoid double counting cases where unigrams and bigrams have overlapping words). Compute $S = S_B \uplus S_U$. This process prioritizes bigrams over unigrams since the former tend to have stronger semantic meanings [60].
(4) Compute the partisan bias score Γ of S as

$$\Gamma(S) = \frac{\sum_{s \in S} \gamma(s)}{|S|}, \tag{2}$$

i.e., the mean bias score of all n-grams found in T'.

Γ scores range from $[-1, 1]$, with -1 (1) indicating that the partisan cues in the snippet or webpage lean heavily to the left (right).

Note that we have intentionally chosen to normalize Γ scores using the total number of partisan queues in the text, rather than the total number of tokens in the text. We chose this formulation because webpage text tends to be orders of magnitude longer than snippet text. If we normalized by the number of tokens, then webpages would be guaranteed to have much lower scores than snippets, and thus it would not be reasonable to compare the scores for a snippet and the corresponding webpage.

Relative Bias Scores. In most of the analysis in § 4, our focus is on comparing the Γ scores of (snippet, webpage) pairs. To compare the magnitude and direction of the two Γ scores, we defined a *relative political bias score* Δ computed as

$$\Delta(\Gamma_s, \Gamma_w) = \begin{cases} \Gamma_s - \Gamma_w & \text{if } \Gamma_w \geq 0 \\ \Gamma_w - \Gamma_s & \text{if } \Gamma_w < 0 \end{cases} \tag{3}$$

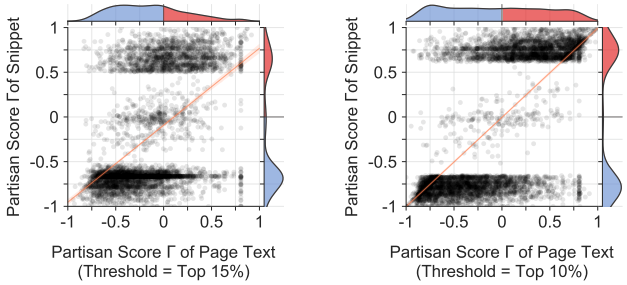


Figure 2: Partisan scores Γ of (snippet, webpage) pairs under the top 15% and top 10% lexicon thresholds.

where Γ_s and Γ_w are the Γ scores of the snippet and webpage, respectively. Δ scores exist in the range $[-2, 1]$, with three distinct subranges that have different interpretations.

- $[0, 1]$: Δ scores in this range indicate that the snippet had more partisan cues than the webpage, thus we say that the snippet *amplified* partisan bias.
- $[-2, -1]$: Δ scores in this range indicate that the snippet and webpage had opposite polarity (i.e., one was right-leaning and the other left-leaning, or vice versa), thus we say that the snippet *flipped* the partisan bias.
- $[-1, 0]$: Δ scores in this range are ambiguous. They could indicate that the snippet had fewer partisan cues than the webpage, thus the snippet *decreased* partisan bias, or they could indicate that the snippet flipped the partisan bias.

In § 4, we present both the distribution of Δ scores, as well as the percentage of cases where snippets amplified, decreased, or flipped partisan bias, to resolve ambiguity. Furthermore, to be conservative in our analysis, we also interpret Δ scores in the range $[-0.1, 0.1]$ as *unchanged*, i.e., the snippet and webpage had sufficiently similar Γ scores that a person is unlikely to notice substantive differences between their use of partisan cues.

Thresholding the Lexicon. As shown in Figure 1, the vast majority of terms in our lexicon have low γ scores, i.e., they are not strongly partisan, and are unlikely to be recognized as partisan by a human being. Since we are only interested in strongly partisan cues, we chose three thresholds $x = [5\%, 10\%, 15\%]$ to prune our lexicon. For threshold x, we select the top x left-leaning and top x right-leaning n-grams in the lexicon.

Table 2 shows the number of unigrams and bigrams that remain in the lexicon after thresholding, as well as the the number and percentage of all (snippet, webpage) pairs that are scorable at a given threshold. We say that a (snippet, webpage) pair is not scorable if Γ is undefined for either, i.e., the snippet or the webpage contains zero n-grams from the given lexicon once the words in the query itself are removed.

As expected, Table 2 shows that the percentage of our dataset that is scorable shrinks rapidly as the threshold becomes stricter. In § 4, we only analyze the datasets at the 10% and 15% thresholds because the scorable sample at 5% is too small.

4 ANALYSIS

In this section, we present our analysis comparing the partisan cues in snippets and the corresponding webpages. Throughout our analysis, we use relative political bias Δ scores to compare the partisanship of (snippet, webpage) pairs, and we present results derived from two lexicons containing the top 15% and 10% of partisan terms from our Vote Smart lexicon (see § 3.3). We begin by examining the overall distribution of Δ scores, and then examine various subsets of our full dataset to determine if Δ scores vary by search query, website, and webpage structure.

4.1 Overall Results

Figure 2 shows the Γ scores for each (snippet, webpage) pair in our dataset under the 15% and 10% lexicon thresholds. As expected, the distribution of scores for snippets is strongly bimodal because the average snippet only includes 1–2 partisan cues and they are likely to share the same polarity (e.g., both lean right or both lean left). In contrast, the distribution of scores for webpages is more evenly distributed, since the average webpage contains 30–52 partisan cues depending on the lexicon threshold. The Spearman rank correlation between the snippet's and webpage's Γ scores reveal moderate correlation with coefficients 0.50 and 0.63 at $p < 0.001$ for the top 15% and top 10% lexicon thresholds, respectively.

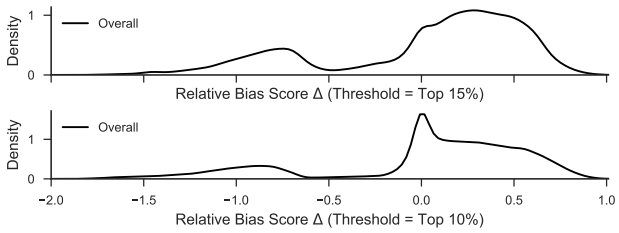
Figure 3a shows the distribution of Δ scores calculated over our entire dataset of (snippet, webpage) pairs under the top 15% and top 10% lexicons. The distributions have similar, bimodal shape regardless of lexicon: the bulk of the volume is in the $[0, 1]$ range, indicating that the snippets have stronger partisan cues than the corresponding webpages. The remaining volume tends to fall in the $[-1.5, -0.5]$, indicating snippets that either (1) have weaker partisan cues than the corresponding webpage or (2) have opposite partisan cues than the corresponding webpage (i.e., the political polarity has flipped).

The "Overall" bar in Figure 4 presents the fractions of (snippet, website) pairs where the snippet *amplified*, *decreased*, *flipped*, or had *unchanged* partisanship relative to the corresponding webpage. Recall that we conservatively consider Δ scores in the range $[-0.1, 0.1]$ to be unchanged, since the relative difference in partisanship between the snippet and webpage are unlikely to be noticeable in practice. We observe that 54–58% of snippets amplified partisan cues, depending on the lexicon threshold. Cases where polarity flipped account for 19–24% of the total, while cases where the snippet had weaker partisan cues only account for 3–4% of the total.

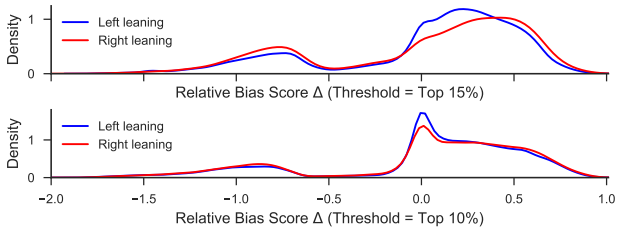
4.2 By Location

The overall results in § 4.1 surprised us: they suggest that Google Search's document summarization algorithm constructs snippets that consistently amplify partisanship. To investigate why this might be the case, we examine where Google's algorithm tends to extract snippets from in webpages, and whether this relates to the use of partisan cues.

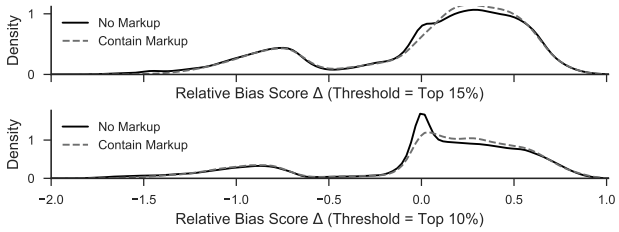
Figure 5 shows the distribution of locations within webpages that snippets were extracted from in our dataset. We split the analysis between snippets that were extracted entirely from one location, and snippets that were composed of several pieces that were extract



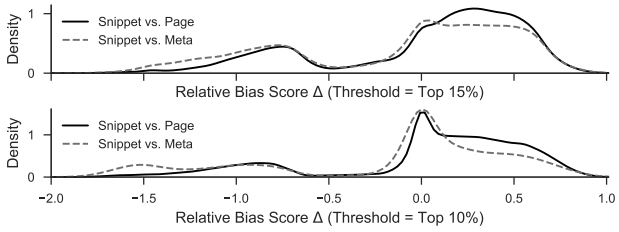
(a) All (snippet, webpage) pairs



(b) Split by query polarity



(c) Split by webpages with and without structured meta-data



(d) Snippets compared to webpage text and meta-data

Figure 3: Distribution of relative political bias scores Δ under the top 15% and top 10% lexicon thresholds. We present the overall score distribution, as well as distributions where the (snippet, webpage) pairs are split by various criteria.

from different locations in the webpage[4]. We observe that the algorithm has a strong preference for text that is earlier in documents. This may interact poorly with the "inverted pyramid" journalism

[4]In this analysis, we ignore snippets that were extracted entirely from webpage meta-data

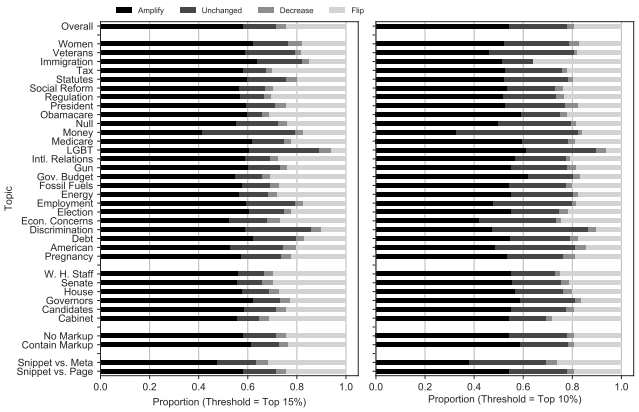


Figure 4: Fraction of cases where the snippet amplified, decreased, flipped, or had unchanged partisanship relative to the corresponding webpage. We present stacked bars for all (snippet, webpage) pairs, pairs within various query topics, pairs with webpages containing and not containing structured meta-data as well as snippets versus meta-data and versus the page text. We present results under the top 15% and top 10% lexicon thresholds.

format, which places the most specific language and jargon at the head of the story.

To further investigate the relationship between snippet location and partisanship, we calculated Δ scores between snippets and subsets of text from the corresponding webpage, where each subset contains the initial f percentage of page text. We show these results in Figure 6, varying f from 5% to 100% of the webpage text. We find that the fraction of cases where the snippet amplified the machine-coded partisanship increases as we consider larger percentages of webpage text, while the fraction of unchanged cases decreases precipitously. This suggests that either: (1) highly partisan terms at the beginning of the page, which get selected by the summarization algorithm, are eventually averaged out by less partisan terms later in the page, or (2) partisan terms of one polarity at the beginning of the page are counteracted by terms with opposite polarity later in the page. Either way, the choice to algorithmically privilege specific text snippets often leads to a mismatch between the machine-coded partisanship of the snippet versus the entire document.

4.3 By Root Query

Next, we examine how Δ scores vary by root query. To make our analysis tractable (recall that we ran 88,745 queries on Google Search), we clustered our root queries by (1) topic (see § 3.2 and Table 1) in the case of partisan phrases, and (2) job in the case of politician's names (e.g., US Senator, state governor, etc.). In total, we examine 24 topical clusters of partisan phrases and 6 clusters of politicians. Additionally, we aggregate each root query together with its associated autocomplete suggestions.

Figure 4 shows the breakdown of (snippet, webpage) relationships for different clusters of root queries. With few exceptions, the individual clusters tend to exhibit the same characteristics as the overall dataset. Notable exceptions include: the "Money" topic has

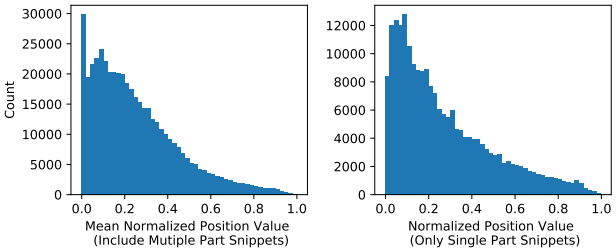


Figure 5: Distribution of the positions in webpages that snippets were extracted from, normalized over the total webpage length.

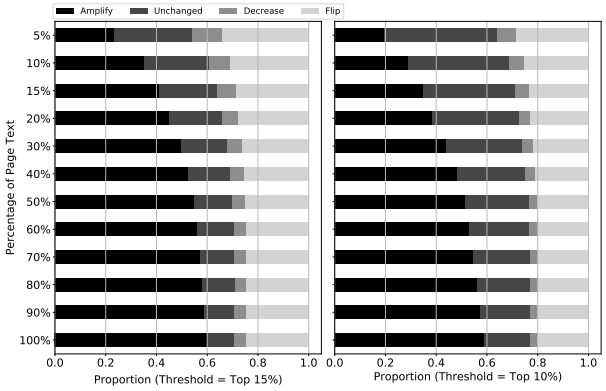


Figure 6: Fraction of cases where the snippet amplified, decreased, flipped, or had unchanged partisanship relative to subsets of text from the corresponding webpage, always starting from the top of the page. We present results under the top 15% and top 10% lexicon thresholds.

the lowest fraction of amplifying (33–41%) and highest fraction of unchanged (38–50%) snippets; under the 15% threshold, the "Immigration" and "Obamacare" topics have the largest fractions of amplifying (64%) and flipped (31%) snippets, respectively; under the 10% threshold, the "Government Budgeting" and "Immigration" topics have the largest fractions of amplifying (62%) and flipped (34%) snippets, respectively.

4.4 By Query Polarity

Next, we examine how Δ scores vary by query polarity. Specifically, we take our clusters of the root queries and divide them into left- and right-leaning queries. For partisan phrases, this is done using their γ score (see § 3.1); for politician's names, this is done using their party affiliation. Note that for this analysis, we omit six topical clusters that only contained root queries of one polarity (e.g., "LGBT", see Table 1).

Figure 3b shows the distribution of Δ scores for (snippet, webpage) pairs, split between left- and right-leaning queries. The distributions exhibit roughly the same bimodal patterns as the overall

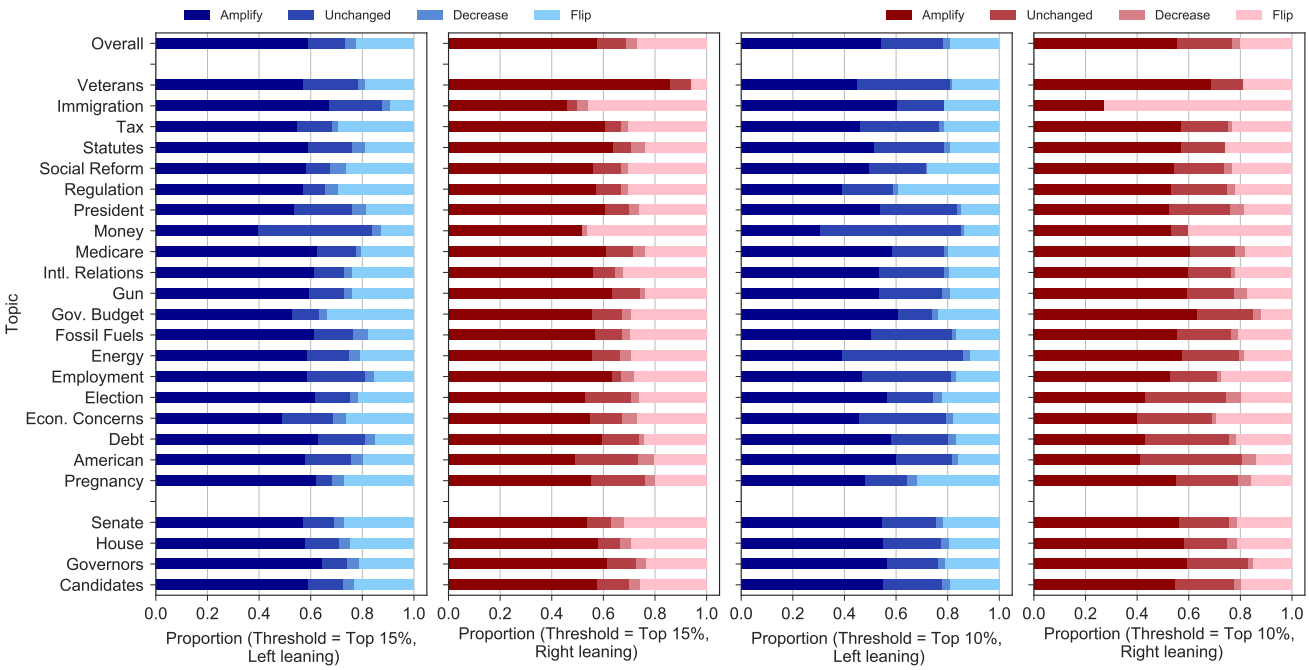


Figure 7: Fraction of cases where the snippet amplified, decreased, flipped, or had unchanged partisanship relative to the corresponding webpage, split by query polarity. We present stacked bars for all (snippet, webpage) pairs and pairs within various query topics. We present results under the top 15% and top 10% lexicon thresholds.

distributions shown in Figure 3a. With respect to each other, we used the Kolmogorov-Smirnov test to compare the left- and right-leaning distributions under each lexicon threshold, respectively, and found that they are significantly different (top 15%: $D = 0.06^{***}$; top 10%: $D = 0.03^{***}$). However, the differences between the means of the distributions are very small (top 15%: 0.02; top 10%: 0.004), meaning that the differences in the distributions have little substantive impact.

Figure 7 shows the fractions of (snippet, webpage) relationships broken down by root query and query polarity, and under different lexicon thresholds. The "Overall" bars show that left- and right-leaning queries behave similarly: 54–59% of snippets from left-leaning queries amplify partisan cues, versus 55–58% of snippets from right-leaning queries; 19–22% of snippets from left-leaning queries flip polarity, versus 20–27% of snippets from right-leaning queries. However, as we previously observed there are exceptions. For example, in the "Immigration" topic, snippets in SERPs generated by left-leaning queries amplify partisan cues 61–67% of the time, versus 27–46% of the time for right-leaning queries. Similarly, in the "Money" topic, snippets generated by left-leaning queries amplify partisan cues 30–40% of the time, versus 52–53% of the time for right-leaning queries.

4.5 By Website

Next, we analyze how Δ scores vary by website, i.e., we group the (snippet, webpage) pairs by the domain hosting the webpage. For each website that had a sample size >100, we use the Wilcoxon V signed-rank test to compare the snippets' Γ scores and webpages' Γ scores, and further applied a Boneferroni correction for multiple hypothesis testings. For the top 15% lexicon we tested 235 websites, versus 113 for the 10% threshold. Ultimately, there were 31 domains whose snippets' Γ scores were significantly different from its webpages' Γ scores under both of the lexicon thresholds. We highlight these websites in Table 3, along with the count of (snippet, webpage) pairs for each websites. We organize the websites by type, including twelve news sites, seven political websites, three social media websites, and three law-related websites. We could not categorize the last six websites.

The results in Table 3 demonstrate that there are particular websites where Google Search's summarization algorithm consistently derives snippets with different machine-coded partisanship than the webpage text. This is unsurprising in the case of social media websites, since each page may contain political content from different people, and thus a single snippet may not capture all of their views. The results are more surprising in the case of news and political websites. It is difficult to say whether these findings are caused by (1) some unanticipated, but consistent, interaction between the content or formatting on these websites and Google Search's summarization algorithm, or (2) whether this reflects an intentional effort by the authors of these websites to "optimize" their appearance on Google Search by highlighting partisan cues.

4.6 By Structured Meta-data

According to Google's documentation, its snippet generation algorithm may leverage structured meta-data from within a webpage when summarizing it [28, 29]. This meta-data can be as simple as

Table 3: Websites that have significant difference between text scores and snippet scores *Note:* $^{*}p < 0.05$; $^{**}p < 0.01$; $^{***}p < 0.001$.

Website	Wilcoxon V	Count
247sports.com	$7.78 \cdot 10^{3***}$	255
bloomberg.com	$6.56 \cdot 10^{4***}$	749
businessinsider.com	$1.59 \cdot 10^{5***}$	995
cnn.com	$3.72 \cdot 10^{5***}$	1503
dallasnews.com	$3.09 \cdot 10^{3***}$	198
espn.com	$3.50 \cdot 10^{4**}$	429
forbes.com	$3.05 \cdot 10^{5***}$	1356
fortune.com	$1.14 \cdot 10^{4**}$	259
projects.propublica.org	$2.41 \cdot 10^{3***}$	170
realclearpolitics.com	$4.26 \cdot 10^{4***}$	534
reuters.com	$5.95 \cdot 10^{4***}$	590
washingtonexaminer.com	$2.76 \cdot 10^{4***}$	397
ballotpedia.org	$2.92 \cdot 10^{6***}$	6795
congress.org	$4.28 \cdot 10^{4**}$	477
eeoc.gov	$5.37 \cdot 10^{3***}$	222
gop.gov	$4.12 \cdot 10^{3***}$	289
govtrack.us	$7.52 \cdot 10^{4***}$	626
numbersusa.com	$7.28 \cdot 10^{2***}$	141
votesmart.org	$4.54 \cdot 10^{5***}$	1901
twitter.com	$3.57 \cdot 10^{7***}$	12597
youtube.com	$2.07 \cdot 10^{5***}$	3931
yelp.com	$7.32 \cdot 10^{4***}$	679
law.cornell.edu	$3.11 \cdot 10^{3***}$	211
pview.findlaw.com	$1.71 \cdot 10^{4*}$	301
texasbar.com	$9.50 \cdot 10^{2***}$	118
beenverified.com	$3.15 \cdot 10^{4***}$	414
doctor.webmd.com	$3.33 \cdot 10^{4***}$	504
healthgrades.com	$1.22 \cdot 10^{5***}$	995
legacy.com	$1.40 \cdot 10^{6***}$	3335
ratemyprofessors.com	$7.87 \cdot 10^{3**}$	217
yellowpages.com	$2.17 \cdot 10^{4***}$	374

HTML `meta` tags, or as complex as structured meta-data markups like Microformats[5], RDFa, and Microdata. Google is intentionally ambiguous about when and whether meta-data is used to generate snippets to prevent gaming of the algorithm by Search Engine Optimization (SEO) services.

In this section, we examine whether pages that contain structured meta-data data have snippets that are more or less consistent with the webpage content. Page authors could use structured meta-data to help supply Google Search's snippet generation algorithm with information that is more representative of the webpage's content; alternatively, they could use meta-data to misrepresent the webpages content (e.g., by making it seem more partisan and inflammatory) to attract visitors.

To implement this comparison, we split our dataset into two sets of (snippet, webpage) pairs, based on whether the webpage contained structured meta-data. We specifically looked for ten kinds of Microformat meta-data, including specific attributes for objects like *People and Organizations*, *Blog Posts*, *Locations*, and *Events*. 17% of webpages in our dataset contained Microformat meta-data; of these webpages, the top objects included *People and Organizations* (62%), *Blog Posts* (30%), and *Locations* (2%).

Figure 3c shows the distribution of Δ scores for (snippet, webpage) pairs, split between webpages that do and do not contain structured meta-data. As above, we used the Kolmogorov-Smirnov test to compare the two distributions under each lexicon threshold, respectively, and found that they are significantly different (top 15%: $D = 0.04^{***}$; top 10%: $D = 0.06^{***}$). However, the differences between the means of the distributions was very small (0.03), again suggesting that the differences in the distributions has little substantive impact.

Figure 4 shows the fractions of (snippet, webpage) relationships broken down by webpages containing and not containing structured meta-data. Webpages with markup have a slightly greater fraction of snippets that amplify bias (59–61% versus 54–58%) and slightly lower fraction of snippets that leave partisanship unchanged (12–20% versus 14–24%). Based on these results, structured meta-data does not appear to make a widespread difference on the machine-coded partisanship of Google Search snippets.

Next, we examine the relationship between snippets and meta-data from a different perspective. Thus far, all of our analysis has compared the partisan cues in snippets to the partisan cues in the text of the webpage (see § 3.2). However, in practice we observe that Google Search's snippet generation algorithm often extracts sentences and phrases from meta-data in the webpage, such as the as the HTML `meta description` tag and alt-text on HTML `img` tags. This is an important distinction, because this meta-data is typically not visible to the user. This creates situations where the snippet for a webpage may be extracted from text that reader of that page may never see.

To analyze the differences between visible page text and invisible meta-data, we compare the Γ score of each snippet to the Γ scores of the page text and meta-data of the corresponding webpage.[6] To implement this comparison, we extracted all of the meta-data from the webpages in our corpus.

Figure 3d shows the distribution of Δ scores for (snippet, webpage) pairs when we compare snippets versus meta-data and versus the page text. We ran the Kolmogorov-Smirnov test on these distributions and found them to be significantly different (top 15%: $D = 0.11^{***}$; top 10%: $D = 0.04^{***}$). However, unlike all of our previous comparisons, in this case the differences between the means of the distributions were large (top 15%: 0.13; top 10%: 0.20), indicating that there are qualitative differences between meta-data and page text with respect to the use of partisan cues.

Figure 4 explores this further by presenting the fractions of cases where snippets amplified, decreased, or flipped partisanship relative to the webpage text and meta-data. When compared to meta-data, snippets were less likely to amplify partisan cues (38–48% versus 54–58%), more likely to leave partisanship unchanged (16–31% versus 14–23%), and more likely to flip polarity (26–31% versus 19–24%). There are two potential explanations for this finding: (1) Google Search's snippet generation algorithm may prefer to use phrases extracted from meta-data, thus leading to closer alignment between the partisan cues in snippets and meta-data; or (2) webpage authors may use more partisan cues in meta-data than in the page text (the average number of partisan cues in webpage meta-data in our dataset is 22–33, depending on the fraction of our lexicon that is used for scoring).

[5]Microformats: http://microformats.org/

[6]Note that the results for "Snippet vs. Page" in Figure 3d and Figure 4 are identical to the "Overall" results; we repeat them to make visual comparison easier.

5 DISCUSSION

In this work, we present the first evaluation of partisan cues in Google Search snippets. Our work relies on a dataset of 820,795 unique (snippet, webpage) pairs that were gathered from Google Search based on 88,745 unique queries for partisan phrases, names of US politicians, and all of their autocomplete suggestions. We quantify the partisan cues in our dataset using a lexicon of highly partisan unigrams and bigrams extracted from political speeches hosted by Vote Smart.

The main takeaway of our study is that Google Search snippets tend to amplify partisan cues relative to the original webpages (see Figure 4). This may have serious implications for how people consume political information from Google Search: if a person "scans the headlines" on a SERP, they may be left with a more partisan impression than if they read the source documents [25, 27]. Furthermore, partisan cues in snippets may prime users with a particular framing that will color their perception of the webpages if they click through and read them [20, 25]. Field-studies should be conducted to determine whether these hypothesized effects actually hold true in practice.

One important, unresolved question is *why* Google Search's snippet generation algorithm tends to produce snippets that amplify partisan cues. It is extremely unlikely that this is an intentional design decision on Google's part; rather, this is likely an unintended, emergent effect. Our analysis of the behavior of Google Search's snippet generation algorithm in § 4.2 and § 4.6 suggests that the cause may be interactions between writing style and webpage structure: the snippet algorithm prefers text at the beginning of pages and within meta-data, while page authors tend to use more partisan cues in these locations. Additional algorithm auditing work should be conducted to more fully explore the implementation of Google Search's snippet generation algorithm.

A second, equally important question is: is Google's current snippet generation algorithm acceptable, or should it be altered (and if so, how)? Generating partisan snippets is not necessarily bad design, if the objective is to make the partisan polarity of webpages that appear in SERPs very clear to users before they click the links. Arguably, altering the snippet generation algorithm to eschew or replace partisan cues will lead to snippets that fail to accurately reflect the content of the underlying page, which may worsen the user experience if it leads to users being unsatisfied with pages they visit. That said, it is unclear whether the snippet is the best place to implicitly convey the partisanship of webpages, since the domain of the website may already be sufficient (e.g., US users recognize the differences between Fox News and MSNBC, irrespective of any particular article).

Conversely, it can be argued that reducing or eliminating partisan cues in snippets may have overall benefits for democratic society, i.e., by reframing political hot-topics using non-partisan language that does immediately trigger framing effects and confirmation bias in readers. This could be accomplished by shifting from extractive to abstractive summarization. This design change may be especially important with respect to Google Search, since it is widely believed to be neutral, and it is the first place most web users go to investigate news and politics [4, 18, 19, 30, 55, 65]. We argue that Google, and other search engines, should engage in a transparent and deliberate effort to design how their summarization algorithms interact with news and political content, rather than allowing the algorithm to blindly function as-is, without reflection or critique.

Applications. Our methods for scoring partisan cues in text have applications beyond measurement and auditing. *First*, our methods could be used to help improve the quality of document summarization algorithms, e.g., by identifying snippets that are more closely aligned with the overall political valence of a document. *Second*, our methods could be used to inform the design of document summarization algorithms that are specifically tuned to increase diversity, e.g., by finding document with a particular political valence, or by intentionally extracting snippets from documents that have reduced or flipped polarity.

Limitations. Our study has several limitations. *First*, the Γ scores that we calculate for individual webpages are not strongly correlated with the overall partisanship of the associated websites, as measured using existing scales like AllSides and others [1, 2, 6, 57]. This is expected: the webpages we sample from any given website are not necessarily representative of the website as a whole.

Second, our approach for parsing text from webpages is overly permissive: text that is not relevant for our analysis (e.g., navigation bars, page footers, etc.) will also be extracted and included in our analysis. In practice, there is no perfect tool for extracting "content" from webpages. We do not expect this additional text to have a substantive impact on our analysis.

Third, our method for scoring partisan cues does not consider contextual semantics. For example, we do not consider negation or sarcasm. Unfortunately, this is a fundamental limitation of current NLP techniques that impacts many contemporaneous studies [57].

Fourth, our lexicon may be incomplete, e.g., there may be partisan cues that are used in the media but were not repeated sufficiently by politicians for them to emerge in our lexicon. The potential for false negatives informed our decision not to analyze (snippet, webpage) pairs containing zero partisan cues from our lexicon, since it would have been presumptuous to conclude that these samples contained "zero partisanship."

Fifth, our study is not longitudinal, and it is unclear whether our results generalize over time.

Finally, people may not recognize all of the n-grams in our lexicon as partisan, or they may not react to them in proportion to their γ score (i.e., people may judge them to be more partisan or less partisan). Extrapolating further, this means that people may not make the same judgments about partisan cues in snippets and webpages that we do in our analysis. Additional fieldwork is necessary to validate the n-grams in our lexicon and their scores, as well as evaluate people's perceptions of partisan cues in situ on SERPs.

ACKNOWLEDGMENTS

This research was supported in part by NSF grant IIS-1553088. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

REFERENCES

[1] AllSides. 2018. Media Bias Ratings. AllSides. https://www.allsides.com/media-bias/media-bias-ratings

[2] Eytan Bakshy, Solomon Messing, and Lada A Adamic. 2015. Exposure to ideologically diverse news and opinion on Facebook. *Science* 348, 6239 (2015), 1130–1132.

[3] Lorena Leal Bando, Falk Scholer, and Andrew Turpin. 2010. Constructing Query-biased Summaries: A Comparison of Human and System Generated Snippets. In *Proceedings of the Third Symposium on Information Interaction in Context (IIiX '10)*. ACM, New York, NY, USA, 195–204. https://doi.org/10.1145/1840784.1840813

[4] Edelman Berland. 2017. 2017 Edelman Trust Barometer. http://www.edelman.com/trust2017/ Accessed: 2017-03-07.

[5] Sasha Blair-Goldensohn, Kerry Hannan, Ryan McDonald, Tyler Neylon, and Jeffrey C Reynar. 2011. Phrase based snippet generation. US Patent 8,010,539.

[6] Ceren Budak, Sharad Goel, and Justin M Rao. 2016. Fair and balanced? Quantifying media bias through crowdsourced content analysis. *Public Opinion Quarterly* 80, S1 (2016), 250–271.

[7] Ian Budge, Hans-Dieter Klingemann, et al. 2001. *Mapping policy preferences: estimates for parties, electors, and governments, 1945-1998*. Vol. 1. Oxford University Press on Demand.

[8] Ian Budge, David Robertson, Derek Hearl, et al. 1987. *Ideology, strategy and party change: spatial analyses of post-war election programmes in 19 democracies*. Cambridge University Press.

[9] Jonathan Charteris-Black. 2006. Politicians and Rhetoric: The Persuasive Power of Metaphor.

[10] Wei-Fan Chen, Matthias Hagen, Benno Stein, and Martin Potthast. 2018. A User Study on Snippet Generation: Text Reuse vs. Paraphrases. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 1033–1036.

[11] Jianpeng Cheng and Mirella Lapata. 2016. Neural Summarization by Extracting Sentences and Words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, 484–494.

[12] Kevyn Collins-Thompson, Paul N Bennett, Ryen W White, Sebastian De La Chica, and David Sontag. 2011. Personalizing web search results by reading level. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 403–412.

[13] Edward Cutrell, Zhiwei Guan, and Ed Cutrell. 2007. What are you looking for? An eye-tracking study of information usage in web search. , 407–416 pages. https://www.microsoft.com/en-us/research/publication/what-are-you-looking-for-an-eye-tracking-study-of-information-usage-in-web-search/

[14] Miranda De Vries, Daniela Giannetti, and Lucy Mansergh. 2001. Estimating policy position from the computer coding of political texts. *Estimating the policy positions of political actors* (2001), 193–216.

[15] Nicholas Diakopoulos. 2014. Algorithmic accountability reporting: On the investigation of black boxes. *Tow Center for Digital Journalism, Columbia University* (2014).

[16] Nicholas Diakopoulos, Daniel Trielli, Jennifer Stark, and Sean Mussenden. 2018. I Vote For—How Search Informs Our Choice of Candidate. In *Digital Dominance: The Power of Google, Amazon, Facebook, and Apple*, M. Moore and D. Tambini (Eds.). 22.

[17] Daniel Diermeier, Jean-François Godbout, Bei Yu, and Stefan Kaufmann. 2012. Language and ideology in Congress. *British Journal of Political Science* 42, 1 (2012), 31–55.

[18] William H. Dutton and Grant Blank. 2013. *Cultures of the Internet: The Internet in Britain: Oxford Internet Survey 2013 Report*. Technical Report. Oxford Internet Institute, University of Oxford. http://oxis.oii.ox.ac.uk/wp-content/uploads/2014/11/OxIS-2013.pdf

[19] William H. Dutton, Bianca Christin Reisdorf, Elizabeth Dubois, and Grant Blank. 2017. Search and Politics: A Cross-National Survey. (2017).

[20] Ullrich K. H. Ecker, Stephan Lewandowsky, Ee Pin Chang, and Rekha Pillai. 2014. The effects of subtle misinformation in news headlines. *Journal of Experimental Psychology: Applied* 20, 4 (2014), 323–335.

[21] H. P. Edmundson. 1969. New Methods in Automatic Extracting. *J. ACM* 16, 2 (April 1969), 264–285. https://doi.org/10.1145/321510.321519

[22] Robert Epstein and Ronald E Robertson. 2015. The search engine manipulation effect (SEME) and its possible impact on the outcomes of elections. *Proceedings of the National Academy of Sciences* 112, 33 (2015), E4512–E4521.

[23] Robert Epstein, Ronald E. Robertson, David Lazer, and Christo Wilson. 2017. Suppressing the Search Engine Manipulation Effect (SEME). *Proceedings of the ACM: Human-Computer Interaction* 1, 2 (November 2017).

[24] Vanessa Fox. 2009. Google Search Now Supports Microformats and Adds "Rich Snippets" to Search Results. Search Engine Land. https://searchengineland.com/google-search-now-supports-microformats-and-adds-rich-snippets-to-search-results-19055.

[25] Renae Franiuk, Jennifer L. Seefelt, and Joseph A. Vandello. 2008. Prevalence of Rape Myths in Headlines and Their Effects on Attitudes Toward Rape. *Sex Roles* 58, 11–12 (June 2008), 790–801.

[26] Matthew Gentzkow and Jesse M Shapiro. 2010. What drives media slant? Evidence from US daily newspapers. *Econometrica* 78, 1 (2010), 35–71.

[27] M. Glenski, C. Pennycuff, and T. Weninger. 2017. Consumers and Curators: Browsing and Voting Patterns on Reddit. *IEEE Transactions on Computational Social Systems* 4, 4 (Dec 2017), 196–206.

[28] Google. 2018. Customizing Results Snippets. Google Custom Search Developer Documentation. https://developers.google.com/custom-search/docs/snippets.

[29] Google. 2018. Providing Structured Data. Google Custom Search Developer Documentation. https://developers.google.com/custom-search/docs/structured_data.

[30] Jeffrey Gottfried and Elisa Shearer. 2016. News use across social media platforms. Pew Research Center. http://www.journalism.org/2016/05/26/news-use-across-social-media-platforms-2016/

[31] Laura A. Granka, Thorsten Joachims, and Geri Gay. 2004. Eye-tracking Analysis of User Behavior in WWW Search. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '04)*. ACM, New York, NY, USA, 478–479. https://doi.org/10.1145/1008992.1009079

[32] Aniko Hannak, Piotr Sapieżyński, Arash Molavi Kakhki, Balachander Krishnamurthy, David Lazer, Alan Mislove, and Christo Wilson. 2013. Measuring Personalization of Web Search. In *WWW*.

[33] Roderick P Hart. 1984. *Verbal style and the presidency: A computer-based analysis*. Academic Pr.

[34] Roderick P Hart. 2001. Redeveloping DICTION: theoretical considerations. *Progress in communication sciences* (2001), 43–60.

[35] Roderick P Hart, Jay P Childers, and Colene J Lind. 2013. *Political tone: How leaders talk and why*. University of Chicago Press.

[36] Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive Summarization using Continuous Vector Space Models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*. Association for Computational Linguistics, 31–39. https://doi.org/10.3115/v1/W14-1504

[37] Michael Kaisser, Marti A Hearst, and John B Lowe. 2008. Improving search results quality by customizing summary lengths. *Proceedings of ACL-08: HLT* (2008), 701–709.

[38] Jaewon Kim, Paul Thomas, Ramesh Sankaranarayana, Tom Gedeon, and Hwan-Jin Yoon. 2017. What snippet size is needed in mobile web search?. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*. ACM, 97–106.

[39] Chloe Kliman-Silver, Aniko Hannak, David Lazer, Christo Wilson, and Alan Mislove. 2015. Location, Location, Location: The Impact of Geolocation on Web Search Personalization. In *Proceedings of the Internet Measurement Conference (IMC 2015)*. Tokyo, Japan.

[40] Hans-Dieter Klingemann, Richard I Hofferbert, and Ian Budge. 1994. *Parties, policies, and democracy*. Westview Press.

[41] George Lakoff. 2010. *Moral politics: How liberals and conservatives think*. University of Chicago Press.

[42] Michael Laver, Kenneth Benoit, and John Garry. 2003. Extracting policy positions from political texts using words as data. *American Political Science Review* 97, 2 (2003), 311–331.

[43] Michael Laver and John Garry. 2000. Estimating policy positions from political texts. *American Journal of Political Science* (2000), 619–634.

[44] Wei-Hao Lin, Eric Xing, and Alexander Hauptmann. 2008. A joint topic and perspective model for ideological discourse. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 17–32.

[45] H. P. Luhn. 1958. The Automatic Creation of Literature Abstracts. *IBM J. Res. Dev.* 2, 2 (April 1958), 159–165. https://doi.org/10.1147/rd.22.0159

[46] Harry McCracken. 2016. How Google Search Got Past "10 Blue Links". Fast Company. https://www.fastcompany.com/3060044/how-google-search-got-past-10-blue-links.

[47] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in Pre-Training Distributed Word Representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

[48] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[49] Amy Mitchell, Jeffrey Gottfried, Jocelyn Kiley, and Katerina Eva Matsa. 2014. Political Polarization & Media Habits. Pew Research Center's Journalism Project.

[50] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In *AAAI*. AAAI Press, 3075–3081.

[51] Nic Newman, Richard Fletcher, Antonis Kalogeropoulos, David A. L. Levy, and Rasmus Kleis Nielsen. 2017. *Reuters Institute Digital News Report 2017*. Technical Report. Reuters Institute for the Study of Journalism.

[52] C. D. Paice. 1990. Constructing Literature Abstracts by Computer: Techniques and Prospects. *Inf. Process. Manage.* 26, 1 (April 1990), 171–186. https://doi.org/10.1016/0306-4573(90)90014-S

[53] PAUL PENNINGS and JAN KLEINNIJENHUIS. 2003. Measurement of party positions on the basis of party programmes, media coverage and voter perceptions. In *Estimating the policy position of political actors*. Routledge, 182–202.

[54] Media Insight Project. 2014. The Personal News Cycle: How Americans Choose to Get Their News. Media Insight Project. https://www.americanpressinstitute.org/publications/reports/survey-research/how-americans-get-news/

[55] Kristen Purcell, Joanna Brenner, and Lee Rainie. 2012. *Search engine use 2012*. Technical Report. Pew Research Center's Internet and American Life Project.

[56] Filipe N. Ribeiro, Lucas Henrique, Fabricio Benevenuto, Abhijnan Chakraborty, Juhi Kulshrestha, Mahmoudreza Babei, and Krishna P. Gummadi. 2018. Media Bias Monitor: Quantifying Biases of Social Media News Outlets at Large-Scale. In *Proceedings of the AAAI Intl. Conference on Web and Social Media (ICWSM)*. Stanford, USA.

[57] Ronald E Robertson, Shan Jiang, Kenneth Joseph, Lisa Friedland, David Lazer, and Christo Wilson. 2018. Auditing Partisan Audience Bias within Google Search. *Proceedings of the ACM: Human-Computer Interaction* 2, CSCW (November 2018).

[58] Ronald E Robertson, David Lazer, and Christo Wilson. 2018. Auditing the Personalization and Composition of Politically-Related Search Engine Results Pages. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 955–965.

[59] Christian Sandvig, Kevin Hamilton, Karrie Karahalios, and Cedric Langbort. 2014. Auditing algorithms: Research methods for detecting discrimination on internet platforms. In *Proceedings of "Data and Discrimination: Converting Critical Concerns into Productive Inquiry", a Productivereconference at the 64th Annual Meeting of the International Communication Association*.

[60] Yanchuan Sim, Brice DL Acree, Justin H Gross, and Noah A Smith. 2013. Measuring ideological proportions in political speeches. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 91–101.

[61] Philip J Stone, Robert F Bales, J Zvi Namenwirth, and Daniel M Ogilvie. 1962. The general inquirer: A computer system for content analysis and retrieval based on the sentence as a unit of information. *Behavioral Science* 7, 4 (1962), 484–498.

[62] Philip J Stone, Dexter C Dunphy, and Marshall S Smith. 1966. The general inquirer: A computer approach to content analysis. (1966).

[63] Danny Sullivan. 2008. Google Testing Enhanced Listings, "Pagelinks" & Auto-Spelling Correction. Search Engine Land. .

[64] Anastasios Tombros and Mark Sanderson. 1998. Advantages of Query Biased Summaries in Information Retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98)*. ACM, New York, NY, USA, 2–10. https://doi.org/10.1145/290941.290947

[65] Francesca Tripodi. 2018. Searching for Alternative Facts: Analyzing Scriptural Inference in Conservative News Practices. Data & Society.

[66] Oren Tsur, Dan Calacci, and David Lazer. 2015. A frame of mind: Using statistical models for detection of framing and agenda setting campaigns. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 1629–1638.

[67] Andrew Turpin, Yohannes Tsegay, David Hawking, and Hugh E. Williams. 2007. Fast Generation of Result Snippets in Web Search. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '07)*. ACM, New York, NY, USA, 127–134. https://doi.org/10.1145/1277741.1277766

[68] Alexandre A Verstak and Anurag Acharya. 2012. Generation of document snippets based on queries and search results. US Patent 8,145,617.

[69] Kam-Fai Wong, Mingli Wu, and Wenjie Li. 2008. Extractive Summarization Using Supervised and Semi-supervised Learning. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1 (COLING '08)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 985–992. http://dl.acm.org/citation.cfm?id=1599081.1599205

To Return or to Explore: Modelling Human Mobility and Dynamics in Cyberspace

Tianran Hu
University of Rochester
thu@cs.rochester.edu

Yinglong Xia
FutureWei Technologies, Inc.
yinglong.xia.2010@ieee.org

Jiebo Luo
University of Rochester
jluo@cs.rochester.edu

ABSTRACT

With the wide adoption of multi-community structure in many popular online platforms, human mobility across online communities has drawn increasing attention from both academia and industry. In this work, we study the statistical patterns that characterize human movements in cyberspace. Inspired by previous work on human mobility in physical space, we decompose human online activities into ***return*** and ***exploration*** – two complementary types of movements. We then study how people perform these two movements, respectively. We first propose ***a preferential return model*** that uncovers the preferential properties of people returning to multiple online communities. Interestingly, this model echos the previous findings on human mobility in physical space. We then present ***a preferential exploration model*** that characterizes exploration movements from a novel online community-group perspective. Our experiments quantitatively reveal the patterns of people exploring new communities, which share striking similarities with online return movements in terms of underlying principles. By combining the mechanisms of both return and exploration together, we are able to obtain an overall model that characterizes human mobility patterns in cyberspace at the individual level. We further investigate human online activities using our models, and discover valuable insights on the mobility patterns across online communities. Our models explain the empirically observed human online movement trajectories remarkably well, and more importantly, sheds better light on the understanding of human cyberspace dynamics.

CCS CONCEPTS

• **Information systems** → **Social networking sites**; *World Wide Web*; • **Applied computing** → **Sociology**.

KEYWORDS

Cyberspace, Online Communities, Human Mobility, Human Dynamics, Preferential Return, Preferential Exploration

ACM Reference Format:
Tianran Hu, Yinglong Xia, and Jiebo Luo. 2019. To Return or to Explore: Modelling Human Mobility and Dynamics in Cyberspace. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313686

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313686

1 INTRODUCTION

In physical space, human movements follow a set of underlying principles. Much research work has been devoted to human mobility in physical space on various topics such as individual and group level mobility patterns [11, 39], the temporal-spatial properties of human mobility [17], and the relation between individual mobility and social connections [9]. With the rapid advances in information and communication technologies (ICT), especially the smartphones, the majority of the population in modern societies around the world are spending a significant and increasing amount of their daily time and activities in cyberspace, and in doing so connecting to multiple online communities.

Uncovering the statistical patterns that characterize the trajectories of human movements across online communities is not only key to many research areas such as computational sociology [4, 51], social media [19, 35], and complex networks [36], but also of great importance for the design of multi-community platforms [20]. Much work has been done on modelling online trajectories, including learning sequential patterns from user traces [31], detecting information spreading among communities [21, 43], and extracting human mobility patterns in cyberspace [18]. However, these studies are usually descriptive and limited at a coarse user-group granularity, while a finer grained understanding in the underlying rules that people follow when they move across online communities is still missing. To address the deficiency, we employ a popular framework in physics for studying individual movements across real word locations [32, 40]. To be specific, we decompose online activities to two basic types of movements across communities – ***return*** and ***exploration***. We then study based on what mechanisms individuals perform these two types of movements.

We start our work with investigating the relation between return and exploration. When an individual performs a move in cyberspace, she either returns to a previously visited online community, or explores a new community. Previous work on human online activities suggests that, at each move, users are not choosing between return and exploration randomly [2]. One evidence is that the probability of people visiting new communities shows a decreasing tendency over time instead of staying constant [54], indicating an evolving relation between return and exploration. We take a close look at this evolving relation, and quantitatively show that the probability of exploring new communities decreases exponentially with the amount of unique communities previously visited. Generally speaking, this finding implies that the more communities an individual has visited, the more likely she will return to a visited community at her next move, rather than exploring a new one. Interestingly, this relation in cyberspace is similar to the relation between human physical return and exploration in real world, which is originally reported in the preferential return model in physics [40].

With the relation between return and exploration being modelled, we proceed to investigate how humans perform return movement among visited communities. Reportedly, people visit online communities following Zipf's law [7], i.e., an individual's preferred communities receive exponentially more visits than the less preferred. This observation indicates that the probabilities of returning are not evenly distributed among visited communities. Our experiments show that the probability of an individual returning to a community is proportional to the visit frequency she pays to this community. In other words, people are more intended to return to their frequently visited communities. Consequently, the frequently visited communities get even more returns (i.e., the rich get richer), and result in the exponential imbalance. This finding, again, echos the property of people returning to offline locations in physics preferential return model. Therefore, without causing confusion, we also refer to our model for online return movement (including its relation with exploration) as the ***preferential return model***.

We then study the mechanism how humans explore the cyberspace. Usually, people explore online communities driven by their personal demands and interests [12, 53]. This suggests that people are more likely to explore new communities somewhat related to the communities they have visited before, rather than randomly select a unvisited one. In fact, previous literature indicates that the online communities an individual visits can be categorized into ***community-groups*** [13, 29]. In other words, with a high probability, a newly visited community can be classified into one of the previously explored community-groups. Following this idea, we investigate the exploration movement from a community-group perspective. To be specific, given the group label of each online community as well as a user's exploration trajectory, we study how she selects new communities from her previously explored community-groups, or opens a new community-group to explore.

Our experiments quantitatively indicate that exploration movement at community-group level share a remarkable resemblance with return movement, and also follows a ***preferential principle***. We first focus on the probability that an individual explores a community in a community-group that she has never explored before. We discover that this probability also evolves over time, and decreases exponentially with the total amount of community-groups the user has explored. Next, we examine the probability of a user exploring a new community from her previously explored community-groups. Similar to return movement, our results show that people pay more attention to their preferred community-groups. To be specific, with much higher probabilities people select new communities from the preferred community-groups to explore. To the best of our knowledge, this is the first time that such a quantitative mechanism of human exploration in cyberspace being reported. We refer to the mechanism as the ***preferential exploration model***.

Through the lens of both *preferential return* and *preferential exploration* models, we are able to not only see a big picture of how people move across online communities, but also uncover more insights in human mobility patterns in cyberspace. For example, we uncover that some communities attract more people to return. The communities of this type are usually of relatively small sizes, and concentrate on niche topics. Meanwhile, we also find that the members of some other communities are more intended to explore. The communities of this type, differently, are usually of larger sizes, and their topics are broader and more interactive. In fact, our analysis indicates that more than 67% of the variance of user intention to explore new communities can be explained using user visit history. It is also of high research value to compare human dynamics in physical space and cyberspace. Interestingly, our experiments indicate that humans are more likely to explore in the physical world, while in cyberspace they are more likely to return to their familiar communities.

The main contributions of this paper are summarized as follows.

- We systematically study human mobility in cyberspace. In particular, we present a ***preferential return model*** and a ***preferential exploration model***. The models explain empirical evidences remarkably well, and synergistically characterize the statistical patterns of human movements across online communities.
- We further investigate online activities at an individual level using our models. Our findings reveal previously undiscovered insights on human cyberspace dynamics, such as the relations between user activities and the community features, the differences between human online and offline movements, and so on.

2 DATA COLLECTION AND PREPROCESSING

We collect our data from Reddit, one of the most visited websites in the world[1]. Reddit has been used as the data sources in many computational sociology studies [47, 50], due to its high popularity and almost complete data availability[2]. In addition, we select Reddit for our study for two other reasons. 1) Multi-community structure – Reddit is organized into thousands of communities (i.e. subreddits), and users are allowed to post in any communities at will. Such a multi-community setting breeds a large amount of activities across communities [15, 21, 51], and therefore makes the data source ideal for studying human online movements. 2) Complete user traces – all the user activities (except for the ones deleted by users) on the platform are available. In other words, up to a time point, we are able to collect complete user online trajectories from their first activities. Given that human online mobility patterns are evolving over time [16], using complete user traces for our study avoids possible information lose and leads to more precise analysis.

In this work, we take a user post in a Reddit community as a "check-in" activity in the community, and a movement between communities consists of two consecutive posts. Please note that the two consecutive posts can be located in the same community. We first download all the user posts on Reddit from 2012 July to 2014 Dec. This initial dataset contains totally 1 billion records, posted by 9 million users in 0.2 million online communities. We first filter out communities with less than 100 users, resulting in 17 thousand active communities. We then find users who did not join Reddit until 2013 Jan (i.e. remove all the user who have activities (posts) in the first six months in our data), and extract the trajectories of these users across active communities. By doing so, we ensure that the extracted trajectories are complete (i.e. starting from the users'

[1] en.wikipedia.org/wiki/Reddit
[2] Reddit data is publicly available, and free for download at www.reddit.com/3bxlg7

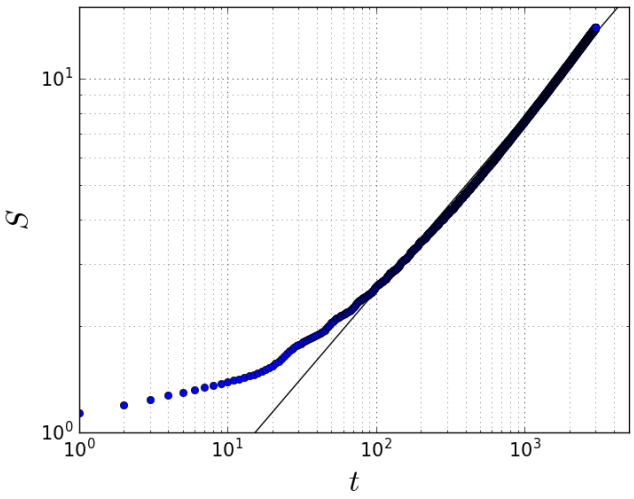


Figure 1: Distribution of the number of visited unique communities S over time t in hours. The fitted straight line indicates that $S \sim t^{\mu}$, where μ is estimated to be 0.42.

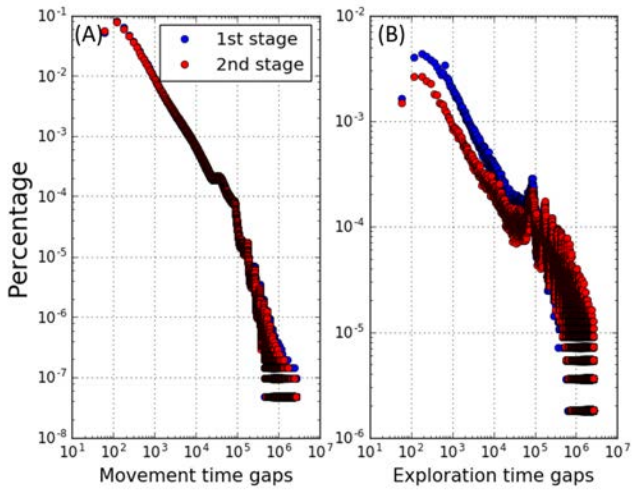


Figure 2: The distributions of time gaps in seconds between all movements (A) and explorations (B) for two stages. Both plots are shown in log-log scale. Please note that the two curves mostly overlap in plot (A), indicating little difference between the first and second stages in terms of the distribution of time gaps between all movements. While (B) reveals that the time gaps between explorations are significantly shorter in the first stage.

first activities) up to 2014 Dec[3]. We further filter out inactive users who posted less than 100 activities or joined less than five online communities. We also follow previous work to exclude non-human accounts by detecting those accounts that contain certain terms such as "Bots", "Moderator" and so on, as well as deleted users [18]. In total, our final dataset contains 16 million activities, forming the online trajectories of 57 thousand users. We will specify in the corresponding sections if more constraints are applied to the data.

3 PREFERENTIAL RETURN MODEL

In this section, we introduce our ***preferential return model***. We first model the probability that a user explores a new online community at each move, denoted as P_{new}. Since return and exploration are complementary, the probability of the user returning to a visited community is naturally $1 - P_{new}$. We find evidence that P_{new} decreases over time, and the decreasing tendency can be well modelled with the number of unique communities the user has visited. We then model, given that the user choose to return, the probability that she returns to a specific visited community i, denoted as Π_i.

3.1 Relation between Return and Exploration

According to our experiments, as well as several previous studies [8, 18, 48], the number of unique communities visited by a user, denoted as S, follows

$$S \sim t^{\mu}, \tag{1}$$

where t is the time the user has spent wandering in cyberspace (Figure 1). The parameter μ is found to be smaller than one [18], and empirically estimated to be around 0.4 from our data. The fact that $\mu < 1$ indicates that the exploration of new communities of users shows a decreasing tendency over time. Two possible reasons could cause the tendency: 1) the probability of user exploring new communities, P_{new}, decreases, or 2) P_{new} stays constant, but the user activity levels reduce over time (i.e. performing fewer movements). To find out, we evenly divide user trajectories into two stages according to the number of communities explored. In other words, given that a user has explored S unique communities in total, she explored $S/2$ in each stage, respectively. We examine the time gaps between explorations, as well as the time gaps between all the moves for each stage, respectively. Our experiments show that, in terms of the distributions of the time gaps between all the moves, little difference is observed between two stages (Figure 2 (A)). In other words, statistically, user activity levels are almost unchanged over time. However, the time gaps between explorations in the later stage are significantly longer than in the former stage (Figure 2 (B)). Clearly, these evidences suggest that although users stay active at the same level, the probabilities of exploring new communities decrease over time.

To model the decreasing probability, we associate P_{new} with S. The idea is rooted from our assumption that the more communities a user has visited, the less likely she might want to explore a new one in her next move. Therefore, we formally model P_{new} as

$$P_{new} = \rho S^{-\gamma}. \tag{2}$$

We then quantitatively validate this formula using empirical data. However, it is not practically feasible to estimate P_{new} for every move a user makes. Fortunately, P_{new} can be also viewed as the

[3]We are aware of that our filtering method could still include some users who joined Reddit before 2012 July, but we assume that the amount of such users are few and have little impact on our experiments.

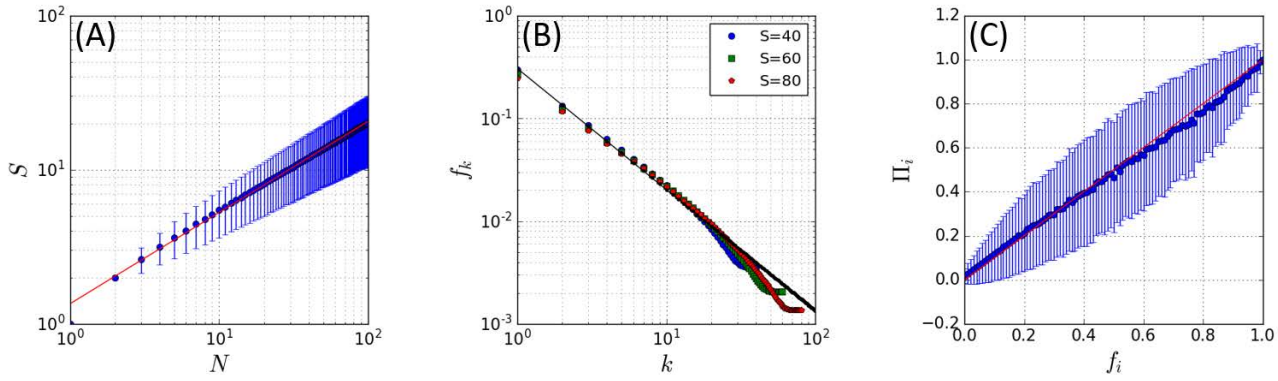


Figure 3: (A) The distribution of the number of uniquely visited communities S over the number of total movements N averaged on all the users, and the error bars indicate the standard deviations. The fitted line in log-log scale indicates that power-law distribution of S over N, as suggested in Formula (4). (B) The relation between visit frequency f_k and visit ranking k for three user groups of different S. The plot shows that Zipf's law holds true, regardless of the number of unique communities a user has visited. (C) The relation between Π_i and f_i averaged on all the user, and the error bars show the standard deviations. The fitted line has a slope of 0.96 and an intercept of 0.005, indicating the equivalence of Formula (6) holds true.

change rate of S over the total number of moves, denoted as N [40][4]. Formally, we can rewrite the equivalence as

$$P_{new} = \frac{\mathrm{d}S}{\mathrm{d}N}. \tag{3}$$

Combining Formulas (3) and (2), we can derive our hypothesized relation between S and N:

$$S = (1+\gamma)(\rho N)^{1/(1+\gamma)}. \tag{4}$$

Since both S and N can be observed from data, we therefore validate our modelling on P_{new} (i.e. Formula 2) by empirically showing that Formula (4) holds true.

Formula (4) suggests a power-law distribution of S over N, i.e. $log(S) \propto log(N)$. We calculate the Pearson Correlation Coefficient between $log(S)$ and $log(N)$ for each user in our dataset. Our experiments report that for more than 96% of the users, the coefficients are larger than 0.9 (p-values < 0.001). In other words, Formula (4) holds true for most users. We further plot this distribution averaged on all the users in Figure 3 (A), and clearly observe a straight line in log-log scale. Therefore, our hypothesis on P_{new} is validated as well. To be specific, we can conclude that, as the number of visited communities increases, the probability of an individual exploring new communities decreases exponentially. Practically, suggested by Formula (4), we are able to determine γ and ρ for each user from the parameters of the straight line in log-log scale[5], and study human mobility in cyberspace at an individual granularity (Section 6).

3.2 Probability of Returning to a Community

Modelling P_{new} allows us to predict how likely an individual would explore a new community, or return to a visited community. We next study the probability of an individual returning to a specific visited community i, i.e. Π_i. It is reported that the frequency f_k of a user's kth most visited community follows Zipf's law [18, 54], and formally,

$$f_k \sim k^{-\zeta}. \tag{5}$$

Figure 3 (B) shows that the this law also appears in our data. The Zipf's law indicates users' imbalanced probabilities of returning to visited communities. Such a "the rich get richer" observation inspires us to model Π_i as the proportion of visits paid to this community previously, denoted as f_i. Formally,

$$\Pi_i = f_i. \tag{6}$$

Theoretically, given P_{new} being modelled as $\rho S^{-\gamma}$, this hypothesis on Π_i can lead to the Zipf's law property in Formula (5), providing evidence for the rationale of Formula (6)[6]. Empirically, We further validate this formula using our data. Since it is infeasible to directly estimate Π_i for each move, we make an approximation in our experiments. To be specific, we cut user trajectories into slices of length L. Within each slice, we estimate Π_i as the percentage of visits to community i. Then within the previous slice, we compute f_i, the frequency to the same community. We set L as 100 for a robust estimate for Π_i[7], and conduct our experiments on totally 23 thousand users whose trajectories cover at least two slices (i.e. total movements N large than 200). We plot the relation between Π_i and f_i in Figure 3 (C). From the plot, we can observe that the relation is linear with a slope very close to 1, indicating that Formula (6) holds true. An interesting observation is the spindle-shaped errors of the fitted straight line – the equivalence between Π_i and f_i is more stable when f_i approaches either end (1 or 0). This is because

[4]Generally speaking, this is because the probability of visiting a new community at each move is equal to the "amount of change" of S brought by each move. For more information, please refer to the supplementary material of [40]

[5]Simply, the slope of the straight line is $1/(1+\gamma)$, and the intercept is $log(\rho)/(1+\gamma) - log(1+\gamma)$.

[6]A detailed proof is beyond the scope of this paper. Please refer to [40] for more information.

[7]Naturally, larger values of L lead to more precise estimates of Π_i. However, since Π_i and f_i are estimated within two different slices, larger L also introduces more temporal differences to the slices, and therefore, introduces larger errors to our validation. We experiment with L from a reasonable range of 50 to 200, and obtain similar results.

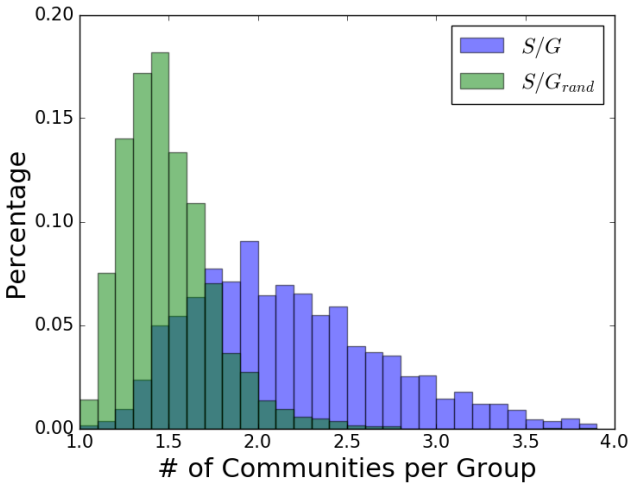


Figure 4: Histogram of the average numbers of online communities a user explored per community-group under our group assignments (S/G) versus random group assignments (S/G_{rand}).

our approximation might introduce some amounts of noises into the fitting, unless the user previously rarely visits a community ($f_i \to 0$) or visits the community very frequently ($f_i \to 1$).

4 PREFERENTIAL EXPLORATION MODEL

The preferential return model characterizes the statistic patterns of ***return*** movement in cyberspace. However, this model does not describe if a user chooses to explore, which new community she will visit. In this section, we introduce our *preferential exploration model* for characterizing ***exploration*** movement. The model is inspired by the fact that the communities a user visit usually can be categorized into several community-groups [12, 53]. Table 1 shows the exploration traces of three randomly selected users from our data. For example, the thrird user in the table explored $S = 28$ unique online communities, while most of these communities are mainly from five community-groups. To be specific, community *sushi*, *asianeats* (asian eats), *tipofmytongue* (tip of my tongue), and *ramen* are clearly related to ***Food***. Similarly, *Boxing*, *kungfu*, *loseit* (lose it), and so on are all communities on ***Exercises***. Such a grouping property suggests that the explorations of a user are not independent – previously explored community-groups could provide further information for later explorations. Following this idea, we investigate exploration movement from a community-group perspective. In particular, we first categorize all the online communities into community-groups. Then we study how users select unvisited communities from community-groups at each exploration.

4.1 Extracting Community-groups

In this work, we adopt a dendrogram-based method to extract community-groups, as suggested in many community detection studies [22, 38]. We apply Pointwise Mutual Information (PMI) as the similarity metric for clustering [42]. To be specific, for two online community C_i and C_j, let U_{C_i} and U_{C_j} denote their user sets, respectively. We compute the similarity between C_i and C_j, as $pmi(U_{C_i}, U_{C_j})$, and formally,

$$pmi(U_{C_i}, U_{C_j}) = log\frac{|U_{C_i} \cap U_{C_j}|}{|U_{C_i} \cup U_{C_j}|}, \tag{7}$$

where $|\cdot|$ denotes the size of a set. PMI computes the similarity between communities from a user co-occurrence perspective. Compared with language-based metrics that are usually adopted for measuring online community similarity [43, 45], PMI better quantifies the likelihood of users moving across communities. For example, *nyc* and *boston* are two typical city-specific communities on Reddit. Since the visitors to city-specific communities are mainly local residents [18], typical users of *nyc* or *boston* are unlikely to pay visit to the other community. Therefore, the similarity between *nyc* and *boston* should be relatively low from a movement perspective, as indicated by a low value of $pmi(nyc, boston)$. On the contrary, according to our experiments, language-based metrics, such as similarities between doc2vec embedding [23], suggest high similarity between the two communities, because of their similarly city-related topics. We set the number of community-groups as 300 for clustering[8], and extract flat group structures for online communities, i.e., each online community is assigned to only one community-group. Table 2 shows some examples of the extracted community-groups. We define that a community-group is explored by a user if the user explores one or more online communities of the group.

We further examine our community-group assignments by comparing the results with random group assignments. Specifically, we randomly cluster all the communities into the same number of groups, and the distribution of group sizes (i.e. number of online communities in each group) is also set to the same as our group assignments. Let G denote the number of community-groups a user explored under our assignments and G_{rand} denote the number of random community-groups a user explored. If our assignments capture the grouping property of user explorations, i.e., user explorations concentrate on several community-groups, then the average number of online communities a user visited per community-group, i.e. S/G, should be significantly larger than S/G_{rand}. We compute both S/G and S/G_{rand} for each user, and conduct a t-test to compare the two values. The test result indicates that a user explores significantly more online communities per community-group under our assignments than random group assignments with a p-value < 0.001. Therefore, our community-group assignments are validated (see Figure 4 for more details).

4.2 Exploring New Community-groups

With the group labels being assigned to online communities, we study the probability of a user exploring a community in a community-group that she has never explored before. We denote this probability as Q_{new}, and the probability of exploring a community in her explored community-groups is thus $1 - Q_{new}$. Similar to our inference on P_{new}, the probability Q_{new} equals to the change rate of the number of unique community-groups a user explores, G, over the number of unique communities she visits, S. Formally, the

[8]Using different settings of community-group numbers leads to same statistic patterns of exploration movement, but with slightly different parameters.

Table 1: Exploration traces of three randomly selected users from our dataset. We also indicate the community-groups each user explored. The community-groups are automatically extracted from all the online communities using a clustering method, and the titles are manually labeled. We use different colors and superscripts in the exploration traces to label which community-group an online community belongs to. The online communities that do not share group membership with any other communities in a trace are colored in black, and are not denoted with superscripts. Also, please note that *AUS.*, *RELP.*, and *EXER.* are short for Australia, Relationships, and Exercises, respectively.

User	*Exploration Traces of Online Communities*	*Comm-groups*
1	*espnyankees*[1] → *NYYankees*[1] → *Libertarian*[2] → *nyjets*[1] → *rangers*[1] → *funny* → *baseball*[1] → *nfl*[1] → *politics*[2] → *nba*[1] → *ShitPoliticsSays*[2] → *NYKnicks*[1].	1. ***Sports*** 2. ***Politics***
2	*subaru*[1] → *Android*[2] → *Autos*[1] → *S2000*[1] → *sysadmin* → *CarAV*[1] → *DIY*[3] → *Documentaries* → *BuyItForLife*[3] → *Ford*[1] → *commandline*[2] → *thewholecar*[1] → *cars*[1] → *UserCars*[1] → *fixit*[3] → *Nexus5*[2] → *programming*[2] → *Shitty_Car_Mods*[1] → *HomeNetworking*[3] → *DesignMyRoom*[3].	1. ***Cars*** 2. ***Tech*** 3. ***DIY***
3	*sushi*[1] → *melbourne*[2] → *todayilearned* → *AustralianPolitics*[2] → *books* → *relationship_advice*[3] → *Boxing*[4] → *amateur_boxing*[4] → *Tobacco*[5] → *ForeverAlone*[3] → *asianeats*[1] → *bodybuilding*[4] → *tipofmytongue*[1] → *kungfu*[4] → *homegym*[4] → *wallpaper* → *ForeverAloneDating*[3] → *Cigarettes*[5] → *MeetPeople*[3] → *loseit*[4] → *CigarMarket*[5] → *dating*[3] → *circlejerkaustralia*[2] → *MakeNewFriendsHere*[3] → *Zippo*[5] → *ramen*[1] → *weightlifting*[4] → *MMA*[4].	1. ***Food*** 2. ***Aus.*** 3. ***RELP.*** 4. ***EXER.*** 5. ***Smoke***

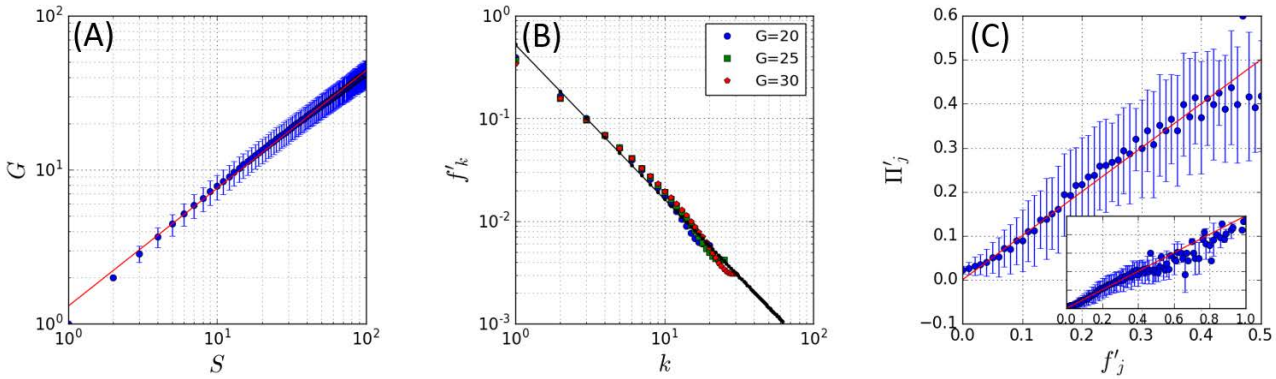


Figure 5: (A) The distribution of the number of unique explored community-groups G over the number of explored communities S averaged over all the users. The plot is in log-log scale, and the error bars indicate the standard deviations. (B) Zipf's law between the exploration frequency to community-groups, f'_k, and the exploration ranking k. (C) The relation between Π'_j and f'_j. The main plot and the inset plot shows the ranges of [0, 0.5] and [0, 1] for f'_j, respectively. Both plots are averaged over all the users, and the error bars show the standard deviations.

equivalence is

$$Q_{new} = \frac{dG}{dS}. \tag{8}$$

We therefore investigate the relation between G and S from user trajectories. Very interestingly, we also discover a power-law distribution of G over S, i.e. $log(G) \propto log(S)$. Specifically, the Pearson Correlation Coefficients between $log(G)$ and $log(S)$ of over 91% users are higher than 0.9, with p-values smaller than 0.001. We plot this distribution in log-log scale in Figure 5 (A).

Given the power-law distribution, as well as Formula (8), we are able to infer the expression of Q_{new}, and formally,

$$Q_{new} = \rho' G^{-\gamma'}. \tag{9}$$

In other words, the probability of exploring a new community-group exponentially decreases with the number of unique groups previously explored. This finding makes interesting but obvious sense – a user's exploration in cyberspace gradually converges to the several categories (i.e., community-groups) she is interested in, and consequently, the more community-groups she has explored, the less likely she will explore a new one. Furthermore, the power-law distribution of G over S can be simply parameterized as

$$G = (1 + \gamma')(\rho' S)^{1/(1+\gamma')}. \tag{10}$$

Please note that, the relation between S and G can be empirically estimated from user trajectories. Therefore, Formula (10) allows us to compute both ρ' and γ' for each user, and to quantify individual exploration patterns (Section 6).

Table 2: Five extracted community-groups using an automatic clustering method with manually labeled titles. The first three groups –*Food*, *Tech*, and *Cars* in the table, are already mentioned in Table 1, and we show more online communities from these community-groups here. We also list two new community-groups – *Wine* and *Asia* in the table.

Title	*Online Communities*
Tech	*Fedora, storage, SQLServer, webdesign, PowerShell, exchangeserver, ipv6, gnu, aws, hacking, git, Cisco, cyberlaws, redhat, Database ,django, docker,* etc.
Cars	*Porsche, BMW, racing, 350z, 4x4, spotted, formula1, RX7, infiniti, LandRover, ATV, Drifting, driving, TopGear, , RatRod, Jeep, HotWheels, carporn, E30.* etc.
Food	*JapaneseFood, grilling, IndianFood, Pizza, recipes steak, BBQ, KoreanFood, streeteats, Cheese, HotPeppers, grilledcheese, slowcooking, meat, hotsauce, Bacon,* etc.
Wine	*tequila, Gin, alcohol, cocktails, liquor, rum, Scotch, vodka, Whiskyporn, beer, wine, bourbon, drunk, worldwhisky, brewing, whisky, beerwithaview,* etc.
Asia	*Korean, China, Osaka, japannews, VietNam, Bangkok, japanesemusic, Tokyo, seoul, JapanTravel, Chinese, taiwan, beijing, Chinavisa, shanghai, cambodia,* etc.

4.3 Exploring Previous Community-groups

Next, we investigate how users select new communities to explore from the community-groups explored before. We start with the exploration frequency a user exercises in each community-group, denoted as f'. For the first time, we uncover that Zipf's law also applies to f'. Generally speaking, users perform exponentially more explorations in their preferred community-groups, resulting in very imbalanced exploration frequencies among community-groups. Formally, the exploration frequency f'_k of the kth most explored community-group satisfies

$$f'_k \sim k^{-\zeta'}. \tag{11}$$

Figure 5 (B) shows the Zipf's law for the users of different numbers of explored community-groups. The exponentially uneven exploration frequency indicates that "the rich get richer" is also reflected in user exploration at the community-group level. Therefore, we naturally model the probability a user explores a community in her explored community-group j, denoted as Π'_j, as the exploration frequency of this community-group f'_j, and formally,

$$\Pi'_j = f'_j. \tag{12}$$

We empirically validate $\Pi'_j = f'_j$ by employing the same method for validating $\Pi_i = f_i$. Specifically, we divide user exploration trajectories into slices of length L'. Within a slice, we estimate Π'_j as the percentage of explorations to community-group j, and within the previous slice we compute f'_j for the same community-group. Although a large L' makes the estimations more precise for users who have sufficiently long exploration trajectories, it also significantly reduces the total number of users for our experiments. Therefore, we make a trade-off, and set $L' = 20$. We then filter out the users who explore fewer than 40 communities ($S < 40$), and retain 4 thousand users. We plot our results in Figure 5 (C). Despite the relatively large errors caused by the value of L', we still see the approximate equivalence between Π'_j and f'_j, which validates Formula (12). Note that the errors increase as f'_j increases, especially for $f'_j > 0.5$ (see the inset of Figure 5 (C)). This is due to the fact that a user can explore at most one community-group with a frequency larger than 0.5. Meanwhile, the total number of users is relatively small. Hence the errors for $f'_j > 0.5$ are relatively large.

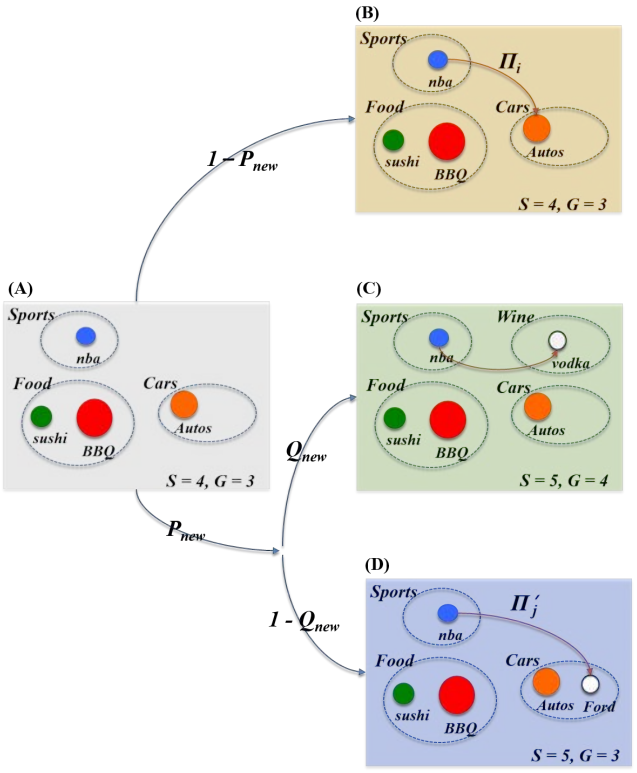


Figure 6: An example of an imaginary user's movements across online communities, illustrating our overall model. From the initial stage (A), the user first chooses between returning to a previously visited community (B) or exploring a new one. If the user chooses to explore, she further decides between exploring a new community-group (C) or selecting a new community from a previously explored community-group (D).

5 OVERALL MODEL

Combining both ***preferential return model*** and ***preferential exploration model***, we are able to cover two basic types of movements across online communities, and obtain an overall model for human mobility in cyberspace.

5.1 An illustrative example of our model

We summarize our overall model using an example of an imaginary user moving across online communities (Figure 6):

- At the initial stage (Figure 6 (A)), the user has visited four communities, belonging to three communities groups ($S = 4$ and $G = 3$): *nba* – ***Sports*** group, *sushi* and *BBQ* –***Food*** group, and *Autos* – ***Cars*** group.
- The user first chooses to perform either exploration or return, with probability P_{new} and $1 - P_{new}$, respectively. Our model indicates that probability P_{new} can be modelled as $\rho S^{-\gamma}$.
- If she chooses to return, she returns to community i with probability Π_i, and the probability can be modeled as the visit frequency to this community f_i. Figure 6 (B) illustrates that the user returns to her previously visited community – *Autos*. Both S and G are unchanged in this case.
- Otherwise, she further chooses to explore an unvisited community in a new or an explored community-group, with probability Q_{new} and $1 - Q_{new}$, respectively. We model Q_{new} as $\rho' G^{-\gamma'}$ in our model. Figure 6 (C) shows that the user explores community *vodka* in a new community-group ***Wine***. In this case, both S and G are increased by one.
- Furthermore, the probability of the user exploring a new community in a previously explored community-group j, denoted as Π'_j, is proportional to the exploration frequency of this community-group f'_j. Figure 6 (D) shows that the user explores a new community *Ford* in her explored community-group ***Cars***. Therefore, S increases to 5, and G remains 3.

5.2 The Preferential Principle

This overall model uncovers two interesting resemblances with respect to human dynamics: 1) the resemblance between human movements in cyber and physical spaces, and 2) the resemblance between online return and exploration movements. First, according to the physical preferential model [30], the relation between return and exploration in physical world shares the same form of Formula (2). Meanwhile, the probability of returning to a physical location is also proportional to the previous visit frequency of this location (Formula 6). In other words, although the two spaces are fundamentally different, people follow a similar "preferential principle" in both spaces – the probability of visiting new communities/locations decreases over time, and people pay more and more visits to the communities/locations they are fond of. Second, it is worth noting the similar mathematical forms of our preferential return and exploration models. The similarities further suggest that the "preferential principle" not only governs how people return to visited online communities, but also applies to the explorations to new communities. To be specific, people explore new community guided by their preference to community-groups – they gradually lose interests in exploring new community-groups, and perform more and more explorations to their preferred groups.

Such a "preferential principle" in both cyber and physical spaces, and for both return and exploration movements, leads to the frequently observed Matthew effect [26] ("the rich get richer" phenomenon) in human mobility patterns. Given the fact that Matthew effect is also observed in many other areas, such as economy and sociology [33, 41], it is worth questioning that if this principle is one of the fundamental rules of human dynamics, and applicable to other human activities. We will save these very interesting extensions for our future work.

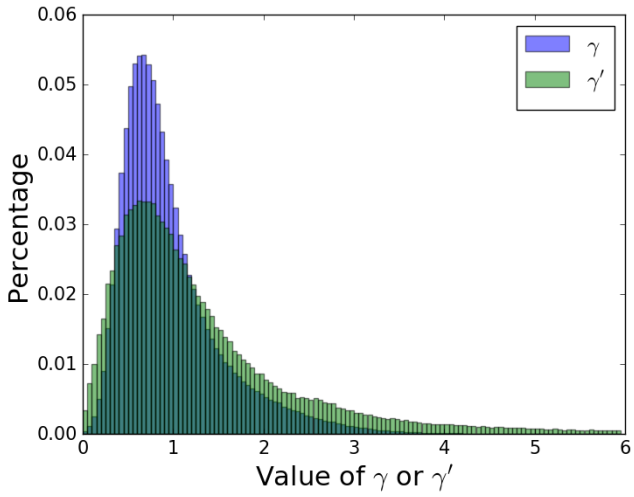


Figure 7: Histogram of γ versus γ' for all the users .

6 HUMAN DYNAMICS IN CYBERSPACE

Our models reveal that user intentions to explore new online communities and community-groups both decrease over time. From Formulas (4) and (10), we can separately derive

$$log(S) = 1/(1+\gamma)log(N) + const, \tag{13}$$

$$log(G) = 1/(1+\gamma')log(S) + const. \tag{14}$$

Apparently, parameter γ quantifies how strong users' intentions are to explore new communities. A larger γ indicates a weaker intention – the increase of number of unique communities S is slower as the total movements N increases. Similarly, parameter γ' quantifies how strong users' intentions are to explore new community-groups. Therefore, in this section, we focus on γ and γ', and study human dynamics in cyberspace.

6.1 Intentions to Explore

We compute both γ and γ' for all the users in our dataset. Our results report that the mean and median of γ are 0.97 and 0.82, respectively; and the mean and median of γ' are 2.52 and 1.17, respectively. We then conduct a t-test between γ and γ', and the test result indicates that γ' is significantly larger than γ (t-statistic = 41.9, and p-value < 0.001). Figure 7 shows the distributions of both parameters. Recall that, in physical space, the relation between return and exploration are the same as in cyberspace. According to the previous research [40], the intention to explore new places in physical space, parameterized by γ_p, is estimated around 0.2. Therefore, we have

$$\gamma_p < \gamma < \gamma' \tag{15}$$

Formula (15) reveals previously unknown insights of human intention to explore. On one hand, from $\gamma_p < \gamma$, we learn that people are more likely to explore in physical space than in cyberspace. One possible reason for the difference lies in the evolution of physical

Table 3: Top five positively and negatively correlated online communities to γ. We list the communities along with their user sizes and the values of the coefficients.

Subreddit	*User Size*	*Coefficient*
Top five positively correlated online communities		
Albuquerque	11,970	0.49
letsgofish	3,211	0.39
tulsa	10,324	0.37
deephouse	23,674	0.33
TankPorn	47,351	0.33
Top five negatively correlated online communities		
TrueAskReddit	158,669	-0.38
poker	84,276	-0.34
baseball	822,279	-0.34
Needafriend	68,294	-0.31
AskHistorians	836,844	-0.29

locations and online communities. Online communities are self-updating – people communicate and interact in communities, and keep posting new contents [24]. On the contrary, physical locations are usually associated with fixed functionality, and change much slower over time. Therefore, people can still be exposed to new contents even if they return frequently to a visited communities in cyberspace, but they might feel bored if they do not explore new places in physical space. On the other hand, the intention to explore new community-groups is the lowest among three, suggested by the large value of γ'. This result is consistent with the psychology theories that describe human cognitive upper limit of handling interests [27, 28]. In other words, when people have explored sufficient community-groups to fulfill their interests, they mainly focus on exploring the new communities within the explored groups instead of opening up new community-groups.

In a nutshell, human dynamics in cyberspace shares many similar principles with physical space while maintaining several unique characteristics.

6.2 Relations between Intentions and Contents

With the knowledge of user intentions to explore, a natural follow-up question is if the intentions are related to the communities or community-groups a user visits. For example, we wonder if people with lower intentions visit different communities from people with higher intentions. To answer the questions, we conduct two linear regression analyses. The dependent valuables of the analyses are γ and γ' of users, and the independent valuables are the online communities and community-groups the users visit for the two analyses, respectively. The two analyses report quite different results. The first analysis reports a high r^2 of over 0.67. This suggests that more than 67% of the variance of γ can be explained by the online communities a user visits. In contrast, the second analysis reports a quite low r^2 of 0.11, indicating that the intention of exploring new community-groups can be barely explained by users' visit histories of community-groups. The analysis results probably suggest human's different cognitive views of single online community and community-groups. We leave the investigation of such differences for the future work.

Furthermore, from the results of the linear regression analysis of γ, we are able to learn the communities that are negatively and positively correlated with γ. Specifically, for each independent valuable (i.e., online community), the linear regression calculates a coefficient and a p-value. The coefficient indicates how correlated an independent valuable is to the dependent valuable, and the p-value suggests how significant the correlation is. We first filter out all the online communities whose p-values are larger than 0.05, and obtain 381 communities positively correlated with γ, and another 427 communities negatively correlated. We list the top five most positively and negatively correlated communities in Table 3. Please note that, positive correlations to γ imply that people who visit these communities prefer to return, and oppositely, negative correlations imply that their visitors prefer to explore.

We can observe that positively and negatively correlated communities are quite different in several aspects. First, positive correlated communities are usually of smaller sizes. This difference can be directly read from the table, and is also confirmed by a t-test (t-statistic = 3.8, p-value < 0.001). Second, the two types of communities are of quite divergent topics. The positively correlated communities usually focus on specific and niche topics. For example, the first (*Albuquerque*) and the third (*tulsa*) are local communities of two American cities[9]. In fact, many other city-specific communities also appear to be positively correlated. The topics of the other three positively correlated communities are also quite specific. For example, *TankPorn* is a community for sharing pictures of tanks, and *letsgofish* focuses on Miami Marlins – an American professional baseball team based in Miami, Florida. While the topics of negatively correlated communities are broader and more interactive. For example, *TrueAskReddit* and *AskHistorians* are both Q&A communities. Meanwhile, *Needafriend* is a community for building up friendships. The contents in these communities involve much more interactions between users. The remaining two communities are both related to general topics – discussions on *poker* and *baseball*.

7 DISCUSSION & FUTURE WORK

Different from previous studies on human online activities, in this work we employ a novel mobility perspective, and decompose online activities into return and exploration movements. Our models indicate that both online return and exploration movements follow a preferential principle. Such a finding further suggests several valuable research questions. Since return movement in physical space reportedly also exhibits the same properties, a question worth studying is if humans explore physical space in a similar preferential way. Furthermore, we also wonder if the principle could extend to other types of human activities. For example, we can study if certain human daily activities, e.g., listening to music, can also be modelled the same way. To be specific, it is possible that people re-listen to their preferred music many times (similar to return movements); and the more music they have listened, the less likely they will seek for new music (similar to exploration movements).

[9]The two cities are Albuquerque, New Mexico, and Tulsa, Oklahoma, respectively.

Moreover, given the fact that the preferential returns are never observed in animal mobility [5], we are also curious if such a preferential principle is one of the fundamental, and meanwhile, unique principles of human dynamics.

Our findings in this paper also imply rich practical applications. For example, we uncover that human explorations in cyberspace are guided by an underlying group structure and related to previously explored community-groups. Meanwhile, our models quantify individual intentions to explore both online communities (γ) and community-groups (γ'). This knowledge could be applied to predicting users' next exploration, and implementing a more sophisticated recommendation system. For example, it is possible to deploy more personalized recommendations based on user exploration intentions. Specifically, for users with a low γ but a high γ', the system can focus on recommending communities within their explored groups; and for users with a low γ and also a low γ', the system may recommend more communities from new community-groups.

Furthermore, our preferential exploration model is built on a flat community-group structure in cyberspace. We extract the group structure from data using an automatic clustering method. Our experiments report that different settings of the number of groups result in slightly different parameters (i.e., γ' and ρ'), but the statistical patterns of exploration in cyberspace are consistent. It is worth noting that many previous studies suggest that the group structure revealed in human activities are in fact hierarchical and overlapping [30, 38]. Flat community-group structures with different group numbers can be regarded as rooted from the same general hierarchy but with different cutting thresholds, and are special cases of overlapping group structures. These facts suggest a more ubiquitous possibility of modelling human online exploration – humans explore communities following an implicit "road-map" of cyberspace. To be specific, in this new model, a global hierarchy defines the positional relations and the cost of moving across online communities. In other words, the hard boundaries of community-groups are replaced by continuous "distances" in cyberspace. We can then naturally associate the probability of exploration with a distance measure and obtain a more generic model without manually setting any thresholds.

8 RELATED WORK

Since Rotman et al. first suggest that people interact and share purpose in online groups, and therefore form sociological communities [37], there has been much work studying human activities across online communities. It is common that people join multiple communities on online platforms [6], and user participation in multiple communities reportedly benefits the survival of the communities [55]. Danescu-Niculescu-Mizil et al. study the interaction between users and online communities, and discover that there are different stages of users joining online communities from a linguistic perspective [10]. Following a similar approach, Tan et al. investigate the language change of users while joining multiple communities, and report that people tend to visit less and less popular communities [44]. Furthermore, how communities affect user actives are also studied. For example, Hamilton et al. study the user loyalty to multiple communities, and reveal the relation between user engagement and community characteristics [15]. Kumar et al. report that communities interact, and even conflict with each others, and most conflicts are started by a small number of specific online communities [21].

Our work is also closely related to the work studying human mobility. Many models have been developed in physics to investigate the mechanisms of the movements across locations in real word [14, 46, 49]. For example, Barabasi et al. propose a priority queue mechanism to model [1], and Ravasz et al. suggest a hierarchical organization [34] to explain the observed burstiness property of human mobility. Song et al. first propose the preferential return model for quantifying the return and exploration movements in physical world [40]. Such a model is then extended by much follow-up work [3, 25]. For example, Pappalardo et al. develop a "returner and explorer dichotomy" for differentiating people of divergent mobility patterns [32].

The similarities between human activities across online communities and movements in physical space have been discussed in some recent work [52, 54]. For example, Hu et al. introduce an analogy between physical space and cyberspace, and leverage framework designed from movements across locations in real world to study mobility patterns across online communities [18]. However, these studies mostly focus on statistical resemblances, instead of investigating the underlying rules that lead to the similarities [2]. Different from the previous work, in this work we empirically show that the mechanisms of movements in both spaces also share interesting similarities.

9 CONCLUSIONS

Given the increasing involvement and significance of cyberspace in people's daily lives, it is important to model the underlying patterns and principles of human online activities. In this work, we have systematically studied human activities across online communities from a mobility perspective. We first propose the preferential return model, which characterizes how people choose between returning to a previously visited community or exploring a new one. Then the model quantifies, if the user decides to return, how likely she is going to return to a specific community. With the return movements being modelled, we further propose the preferential exploration model. This model analyzes human online exploration movements from a novel community-group perspective, and quantitatively describes the processes of people selecting unvisited communities from community-groups. The two models are substantially validated by statistically solid evidences on a large real world data set. Combining these two models, we are able to reveal an interesting preferential principle of human cyberspace dynamics. Generally speaking, the principle suggests that people gradually lose interest in exploring new physical or cyber locations, and more and more often return to the location they are fond of. Such a principle implies a rich variety of future work (as stated in Section 7).

We have further applied our models to analyzing human online trajectories at the individual level, and reveal previously undiscovered insights about human mobility in cyberspace. We quantify user intention to explore in cyberspace and find that, interestingly, people are more likely to explore in physical world. Moreover, we investigate the relations between online communities and user intention to explore. Our experiments report that people who prefer

to return tend to visit communities of smaller size and specific topics, while people who prefer to explore tend to visit communities of larger size and interactive topics.

REFERENCES

[1] Albert-Laszlo Barabasi. 2005. The origin of bursts and heavy tails in human dynamics. *Nature* 435, 7039 (2005), 207.

[2] Hugo S Barbosa, Fernando B de Lima Neto, Alexandre Evsukoff, and Ronaldo Menezes. 2016. Returners and Explorers Dichotomy in Web Browsing Behavior: A Human Mobility Approach. In *Complex Networks VII*. Springer, 173–184.

[3] Marc Barthélemy. 2011. Spatial networks. *Physics Reports* 499, 1-3 (2011), 1–101.

[4] Václav Belák, Samantha Lam, and Conor Hayes. 2012. Cross-Community Influence in Discussion Fora. *ICWSM* 12 (2012), 34–41.

[5] Clifford T Brown, Larry S Liebovitch, and Rachel Glendon. 2007. Lévy flights in Dobe Ju/âĂŹhoansi foraging patterns. *Human Ecology* 35, 1 (2007), 129–138.

[6] Jeffrey Chan, Conor Hayes, and Elizabeth M Daly. 2010. Decomposing Discussion Forums and Boards Using User Roles. *ICWSM* 10 (2010), 215–218.

[7] Anna Chmiel, Kamila Kowalska, and Janusz A Hołyst. 2009. Scaling of human behavior during portal browsing. *Physical Review E* 80, 6 (2009), 066122.

[8] Anna Chmiel, Julian Sienkiewicz, Mike Thelwall, Georgios Paltoglou, Kevan Buckley, Arvid Kappas, and Janusz A Hołyst. 2011. Collective emotions online and their influence on community life. *PloS one* 6, 7 (2011), e22207.

[9] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1082–1090.

[10] Cristian Danescu-Niculescu-Mizil, Robert West, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. No country for old members: User lifecycle and linguistic change in online communities. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 307–318.

[11] Yves-Alexandre De Montjoye, César A Hidalgo, Michel Verleysen, and Vincent D Blondel. 2013. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports* 3 (2013), 1376.

[12] Santo Fortunato, Alessandro Flammini, Filippo Menczer, and Alessandro Vespignani. 2006. Topical interests and the mitigation of search engine bias. *Proceedings of the National Academy of Sciences* 103, 34 (2006), 12684–12689.

[13] Michelle Girvan and Mark EJ Newman. 2002. Community structure in social and biological networks. *Proceedings of the national academy of sciences* 99, 12 (2002), 7821–7826.

[14] Marta C Gonzalez, Cesar A Hidalgo, and Albert-Laszlo Barabasi. 2008. Understanding individual human mobility patterns. *nature* 453, 7196 (2008), 779.

[15] William L Hamilton, Justine Zhang, Cristian Danescu-Niculescu-Mizil, Dan Jurafsky, and Jure Leskovec. 2017. Loyalty in online communities. In *Proceedings of the... International AAAI Conference on Weblogs and Social Media. International AAAI Conference on Weblogs and Social Media*, Vol. 2017. NIH Public Access, 540.

[16] Lei Hou, Xue Pan, Qiang Guo, and Jian-Guo Liu. 2014. Memory effect of the online user preference. *Scientific reports* 4 (2014), 6560.

[17] Tianran Hu, Eric Bigelow, Jiebo Luo, and Henry Kautz. 2017. Tales of two cities: Using social media to understand idiosyncratic lifestyles in distinctive metropolitan areas. *IEEE Transactions on Big Data* 3, 1 (2017), 55–66.

[18] Tianran Hu, Jiebo Luo, and Wei Liu. 2018. Life in the" Matrix": Human Mobility Patterns in the Cyber Space. *ICWSM* (2018).

[19] Sanjay Ram Kairam, Dan J Wang, and Jure Leskovec. 2012. The life and death of online groups: Predicting group growth and longevity. In *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 673–682.

[20] Robert E Kraut and Paul Resnick. 2012. *Building successful online communities: Evidence-based social design*. Mit Press.

[21] Srijan Kumar, William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2018. Community interaction and conflict on the web. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 933–943.

[22] Peter Langfelder, Bin Zhang, and Steve Horvath. 2007. Defining clusters from a hierarchical cluster tree: the Dynamic Tree Cut package for R. *Bioinformatics* 24, 5 (2007), 719–720.

[23] Jey Han Lau and Timothy Baldwin. 2016. An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation. In *Proceedings of the 1st Workshop on Representation Learning for NLP*. 78–86.

[24] Zhiyuan Lin, Niloufar Salehi, Bowen Yao, Yiqi Chen, and Michael S Bernstein. 2017. Better When It Was Smaller? Community Content and Behavior After Massive Growth.. In *ICWSM*. 132–141.

[25] Xin Lu, Linus Bengtsson, and Petter Holme. 2012. Predictability of population displacement after the 2010 Haiti earthquake. *Proceedings of the National Academy of Sciences* 109, 29 (2012), 11576–11581.

[26] Robert K Merton. 1968. The Matthew effect in science: The reward and communication systems of science are considered. *Science* 159, 3810 (1968), 56–63.

[27] George A Miller. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review* 63, 2 (1956), 81.

[28] Ulric Neisser. 2014. *Cognitive psychology: Classic edition*. Psychology Press.

[29] Mark EJ Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical review E* 69, 2 (2004), 026113.

[30] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435, 7043 (2005), 814.

[31] Pietro Panzarasa, Tore Opsahl, and Kathleen M Carley. 2009. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *Journal of the American Society for Information Science and Technology* 60, 5 (2009), 911–932.

[32] Luca Pappalardo, Filippo Simini, Salvatore Rinzivillo, Dino Pedreschi, Fosca Giannotti, and Albert-László Barabási. 2015. Returners and explorers dichotomy in human mobility. *Nature communications* 6 (2015), 8166.

[33] Alexander M Petersen, Woo-Sung Jung, Jae-Suk Yang, and H Eugene Stanley. 2011. Quantitative and empirical demonstration of the Matthew effect in a study of career longevity. *Proceedings of the National Academy of Sciences* 108, 1 (2011), 18–23.

[34] Erzsébet Ravasz, Anna Lisa Somera, Dale A Mongru, Zoltán N Oltvai, and A-L Barabási. 2002. Hierarchical organization of modularity in metabolic networks. *science* 297, 5586 (2002), 1551–1555.

[35] Daniel M Romero, Brendan Meeder, and Jon Kleinberg. 2011. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th international conference on World Wide Web*. ACM, 695–704.

[36] Martin Rosvall and Carl T Bergstrom. 2008. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences* 105, 4 (2008), 1118–1123.

[37] Dana Rotman, Jennifer Golbeck, and Jennifer Preece. 2009. The community is where the rapport is–on sense and structure in the youtube community. In *Proceedings of the fourth international conference on Communities and technologies*. ACM, 41–50.

[38] Huawei Shen, Xueqi Cheng, Kai Cai, and Mao-Bin Hu. 2009. Detect overlapping and hierarchical community structure in networks. *Physica A: Statistical Mechanics and its Applications* 388, 8 (2009), 1706–1712.

[39] Filippo Simini, Marta C González, Amos Maritan, and Albert-László Barabási. 2012. A universal model for mobility and migration patterns. *Nature* 484, 7392 (2012), 96.

[40] Chaoming Song, Tal Koren, Pu Wang, and Albert-László Barabási. 2010. Modelling the scaling properties of human mobility. *Nature Physics* 6, 10 (2010), 818.

[41] Alan T Sorensen. 2007. Bestseller lists and product variety. *The journal of industrial economics* 55, 4 (2007), 715–738.

[42] Qi Su, Kun Xiang, Houfeng Wang, Bin Sun, and Shiwen Yu. 2006. Using pointwise mutual information to identify implicit features in customer reviews. In *International Conference on Computer Processing of Oriental Languages*. Springer, 22–30.

[43] Chenhao Tan. 2018. Tracing Community Genealogy: How New Communities Emerge from the Old. *arXiv preprint arXiv:1804.01990* (2018).

[44] Chenhao Tan and Lillian Lee. 2015. All who wander: On the prevalence and characteristics of multi-community engagement. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1056–1066.

[45] Trang Tran and Mari Ostendorf. 2016. Characterizing the language of online communities and its relation to community reception. *arXiv preprint arXiv:1609.04779* (2016).

[46] Alexei Vázquez, Joao Gama Oliveira, Zoltán Dezsö, Kwang-Il Goh, Imre Kondor, and Albert-László Barabási. 2006. Modeling bursts and heavy tails in human dynamics. *Physical Review E* 73, 3 (2006), 036127.

[47] Alex Wang, William L Hamilton, and Jure Leskovec. 2016. Learning linguistic descriptors of user roles in online communities. In *Proceedings of the First Workshop on NLP and Computational Social Science*. 76–85.

[48] Chunyan Wang, Mao Ye, and Bernardo A Huberman. 2012. From user comments to on-line conversations. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 244–252.

[49] Ye Wu, Changsong Zhou, Jinghua Xiao, Jürgen Kurths, and Hans Joachim Schellnhuber. 2010. Evidence for a bimodal distribution in human communication. *Proceedings of the national academy of sciences* (2010).

[50] Amy X Zhang, Bryan Culbertson, and Praveen Paritosh. 2017. Characterizing online discussion using coarse discourse sequences. In *Proceedings of the Eleventh International Conference on Web and Social Media. AAAI Press*.

[51] Justine Zhang, William L Hamilton, Cristian Danescu-Niculescu-Mizil, Dan Jurafsky, and Jure Leskovec. 2017. Community identity and user engagement in a multi-community landscape. In *Proceedings of the... International AAAI Conference on Weblogs and Social Media. International AAAI Conference on Weblogs and Social Media*, Vol. 2017. NIH Public Access, 377.

[52] Zhi-Dan Zhao, Zi-Gang Huang, Liang Huang, Huan Liu, and Ying-Cheng Lai. 2014. Scaling and correlation of human movements in cyberspace and physical

space. *Physical Review E* 90, 5 (2014), 050802.
[53] Zhi-Dan Zhao, Zimo Yang, Zike Zhang, Tao Zhou, Zi-Gang Huang, and Ying-Cheng Lai. 2013. Emergence of scaling in human-interest dynamics. *Scientific Reports* 3 (2013), 3472.
[54] Zhi-Dan Zhao and Tao Zhou. 2012. Empirical analysis of online human dynamics. *Physica A: Statistical Mechanics and its Applications* 391, 11 (2012), 3308–3315.
[55] Haiyi Zhu, Robert E Kraut, and Aniket Kittur. 2014. The impact of membership overlap on the survival of online communities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 281–290.

MiST: A Multiview and Multimodal Spatial-Temporal Learning Framework for Citywide Abnormal Event Forecasting

Chao Huang
University of Notre Dame
chuang7@nd.edu

Chuxu Zhang
University of Notre Dame
czhang11@nd.edu

Jiashu Zhao
JD.com
zhaojiashu1@jd.com

Xian Wu
University of Notre Dame
xwu9@nd.edu

Dawei Yin
JD.com
yindawei@acm.org

Nitesh V. Chawla
University of Notre Dame
nchawla@nd.edu

ABSTRACT

Citywide abnormal events, such as crimes and accidents, may result in loss of lives or properties if not handled efficiently. It is important for a wide spectrum of applications, ranging from public order maintaining, disaster control and people's activity modeling, if abnormal events can be automatically predicted before they occur. However, forecasting different categories of citywide abnormal events is very challenging as it is affected by many complex factors from different views: (i) dynamic intra-region temporal correlation; (ii) complex inter-region spatial correlations; (iii) latent cross-categorical correlations. In this paper, we develop a Multi-View and Multi-Modal Spatial-Temporal learning (MiST) framework to address the above challenges by promoting the collaboration of different views (spatial, temporal and semantic) and map the multi-modal units into the same latent space. Specifically, MiST can preserve the underlying structural information of multi-view abnormal event data and automatically learn the importance of view-specific representations, with the integration of a multi-modal pattern fusion module and a hierarchical recurrent framework. Extensive experiments on three real-world datasets, *i.e.*, crime data and urban anomaly data, demonstrate the superior performance of our MiST method over the state-of-the-art baselines across various settings.

CCS CONCEPTS

• **Information systems → Spatial-temporal systems**.

KEYWORDS

Abnormal Event Forecasting, Deep Neural Networks; Spatial-temporal Data Mining

ACM Reference Format:
Chao Huang, Chuxu Zhang, Jiashu Zhao, Xian Wu, Dawei Yin, and Nitesh V. Chawla. 2019. MiST: A Multiview and Multimodal Spatial-Temporal Learning Framework for Citywide Abnormal Event Forecasting. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313730

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW'19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313730

1 INTRODUCTION

Citywide abnormal events, like crimes (*e.g.*, robbery, felony assault) and urban anomalies (*e.g.*, blocked driveway, noise), may pose tremendous risks to public safety if not timely handled. According to national statistics, urban anomalies cost Americans more than $100 billion direct and indirect loss in each of recent years [39]. Hence, accurate and reliable prediction of abnormal events is in pressing need for data-driven decision makers to alleviate huge life and economic losses. For example, in disaster control, by forecasting future abnormal events, local governments can design better transportation planning and mobility management strategies to prevent severe social riots [34, 48]. Furthermore, in public order maintaining services, understanding latent occurrence patterns of abnormal events at each geographical region in a city is highly important for people's activities modeling and place recommendation tasks [20, 40]. In this paper, we aim to predict abnormal events of different categories at each region of a city ahead of time, thereby leading to significant improvement in societal welfare.

There exist prior studies on detecting geographical anomalies using spatial-temporal data [7, 17, 26]. Most of these studies identify the abnormal events by analyzing the historic traces or movement patterns of the studied objects using statistical and data mining approaches. However, instead of forecasting events in the future, these approaches typically identify them only after they have occurred, which may lead to significant information delay and lack of preparedness to handle the anomalies in an efficient way.

We identify three key challenges of modeling such abnormal event data from multiple views, which motivate the model design:

First, the distributions of abnormal events in the urban space are highly dynamic and vary greatly from one region to anther. In such cases, the abnormal event occurrences are no longer independent across different locations and it is crucial to take the spatial correlations into consideration for the abnormal event forecasting task. In addition, when modeling dynamic pairwise spatial correlations, probabilistic graphical models [23, 24] may fail due to their heavy computational cost with massive parameters based on prior assumed distributions.

Second, the occurrence pattern of abnormal events often involve underlying factors which are dynamically evolving over time. For example, crime causality on weekdays may differ from weekends. Conventional time series forecasting techniques, such as Autoregressive Integrated Moving Average (ARIMA) [29] and Support Vector Regression (SVR) [30], are mostly limited to linear models which depend solely on the single-level periodic patterns. Thus,

these methods can hardly be responsible to temporal dynamics across different time slots with respect to the occurrences of abnormal events.

Third, explicit and implicit influences among different abnormal event categories are ubiquitous in real life. For example, traffic jam at a specific region may be triggered by a robbery occurring in the same region, due to the crowd aggregation and increased patrol in response to the robbery. Therefore, the occurrence of a specific category of abnormal event may stem from not only spatial correlations between regions as well as temporal dependencies across time slots, but also the influences among different categories.

Motivated by the aforementioned challenges, this work proposes a general and flexible framework–Multi-View Deep Spatial-Temporal Networks (MiST)–for learning the predictive structure from the relationships within the multi-view abnormal event data. Specifically, at the first phase, we develop a context-aware recurrent framework to capture temporal dynamics of abnormal event data in different views and provide automatic view-specific representations. At the second phase, to jointly preserve the inter-region correlations and the cross-category influences together with the encoded temporal patterns of multi-dimensional data, we propose a pattern fusion module based on attention mechanisms to promote the collaboration of different views, and automatically capture the contribution of correlated regions, time slots and categories from the corresponding view in the predictive model. To supercharge MiST modeling of summarized occurrence patterns with time-ordered structural information and non-linearities, a conclusive recurrent network module is designed to model the sequential patterns of fused embedding vectors at the final phase. The final conclusive latent representations are fed into a fully connected neural networks to predict the abnormal events in future time slots.

In summary, we highlight our contributions as follows:

- We introduce a new multi-view and multi-modal spatial-temporal learning framework MiST for predicting abnormal events of different categories at each region of a city. MiST maps all the spatial, temporal, and semantic units into a latent space to preserve their cross-modal correlations.
- We develop a multi-modal pattern fusion module that is tailored to cooperate with a hierarchical recurrent framework for learning the latent region-time-category interactions shared in multi-view data, and automatically adjust the correlations from each view in assisting the prediction task.
- Through extensive experiments performed on three real-world abnormal event datasets collected from NYC and Chicago, we show that MiST consistently surpasses several state-of-the-art baselines across various settings.

The rest of this paper is organized as follows. We formally present the problem in Section 2. Section 3 introduces the proposed framework. The experimental results are presented in Section 4. Section 5 summarizes the related works. We finally conclude our work in Section 6.

2 PROBLEM FORMULATION

In this section, we first introduce the preliminaries and the studied problem. Then, we present the overview of our developed framework.

2.1 Preliminaries

Definition 1. **Geographical Region**. *We partition a city with a grid-based map segmentation. In particular, we divide the whole city into $I \times J$ disjointed grids which consists of I rows and J columns with the longitude and latitude information. Each grid is referred to as a geographical region denoted by $r_{i,j}$, where i and j is the index of row and column, respectively. In this work, we use regions as the minimal units to study the abnormal event forecasting problem.*

We define the geographical region set as $R = (r_{1,1}, ..., r_{i,j}, ..., r_{I,J})$, and suppose that there are L abnormal event categories ($C = (c_1, ..., c_l, ..., c_L)$), where C is the category set indexed by l. Given a time window T, we also split T as non-overlapping and consequent time slots ($T = (t_1, ..., t_k, ..., t_K)$), where K denotes the number of time slots which is indexed by k.

Definition 2. **Abnormal Event Data Source**. *Given a region $r_{i,j}$, we use $\mathbf{Y}_{i,j} = (\boldsymbol{y}^1_{i,j}, ..., \boldsymbol{y}^l_{i,j}, ..., \boldsymbol{y}^L_{i,j}) \in \mathbb{R}^{L \times K}$ to denote the observations of all categorical abnormal events during K past time slots at region $r_{i,j}$. In particular, $\boldsymbol{y}^l_{i,j} \in \mathbb{R}^K$ belongs to category c_l and represents the event observed at region $r_{i,j}$ from time slot t_1 to t_K. In $\boldsymbol{y}^l_{i,j}$, each element $y^{l,k}_{i,j}$ is set to 1 if abnormal event of category c_j is observed at region $r_{i,j}$ in time slot t_k and 0 otherwise.*

Problem Statement. Given the historical abnormal event data source of all categories across geographical regions R in a city from time slot t_1 to t_K, the objective of this work is to learn a predictive framework which infers the unknown event occurrence of each category c_l at each region $r_{i,j}$ in h future time slots. Formally, we aim to compute: $(y^{l,(K+h)}_{i,j} | \mathbf{Y}_{i,j} = (\mathbf{y}^1_{i,j}, ..., \mathbf{y}^L_{i,j}));\ i, j \in [1, ..., I], [1, ..., J]$.

2.2 Framework Overview

Our developed MiST is a multi-layer representation learning framework as shown in Figure 1. Before presenting the details, we first introduce the model input and then elaborate the motivations to design our framework MiST from spatial-temporal-categorical views.

Definition 3. **Event Context Tensor**. *Given a target region $r_{i,j}$, we model different categorical abnormal events of its spatially nearby regions in time slot t_k using an event context tensor, $\mathcal{A}^k_{i,j} \in \mathbb{R}^{I \times J \times L}$ with three dimensions denoting I grid rows, J grid columns, and L categories, respectively. Specifically, given the time slot t_k, each element in $\mathcal{A}^k_{i,j,l}$ is set as 1 if there exists abnormal events of category c_l from the corresponding region $r_{i,j}$ in the $I \times J$ grid map. Otherwise, we set the element in $\mathcal{A}^k_{i,j,l}$ as 0.*

Context-aware Recurrent Framework. To characterize intra-region correlations in terms of dynamic properties of abnormal event distributions from temporal view, we develop our context encoder based on Long short-term memory (LSTM) networks to learn latent representations for the elements of the flattened vector from event context tensor $\mathcal{A}$ in each time slot. The representations learned from our LSTM encoder give MiST the ability to model the evolving time-dependent nature of abnormal events by well capturing not only their local temporal contexts but also multi-level periodical patterns.

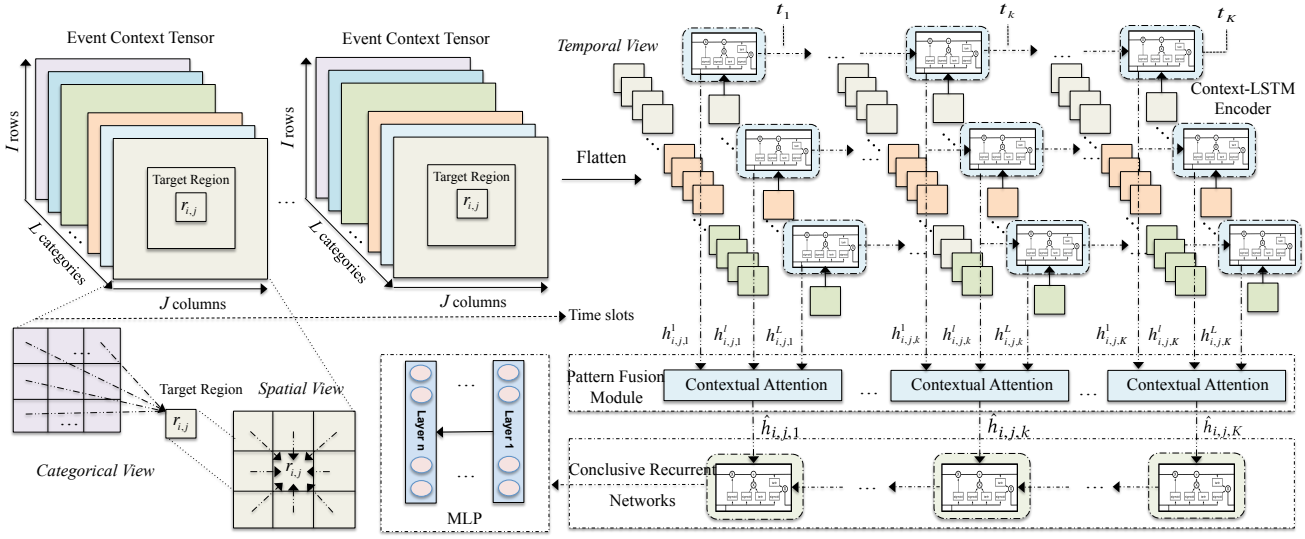


Figure 1: The Multiview and Multimodal Spatial-Temporal Learning Framework.

Multi-Modal Pattern Fusion Module. To jointly capture the inter-region and cross-category correlations in terms of abnormal event distributions, we propose a deep fusion module to model inherent occurrence patterns of abnormal events across surrounding geographical regions and different categories simultaneously. Given a specific time slot t_k, we supply the learned hidden representation vectors of each time-ordered sequence of tensor $\mathcal{A}^k_{i,j}$ to an attention mechanism in order to generate summarized embedding vectors from spatial-categorical views.

Conclusive Recurrent Networks. Relying on the hidden representations generated from spatial-temporal-categorical views, we then develop a conclusive recurrent networks to effectively capture the sequential patterns of cross-modal correlations between location, time, and event category. The final multi-view sequence representations from spatial-temporal-categorical dimensions are stored in the last states of the conclusive recurrent network cells, which provides a guidance for the prediction during the occurrence probability decoding phase.

3 METHODOLOGY

In this section, we present the details of our MiST framework.

3.1 Context-aware Recurrent Framework

In the architecture of MiST, we first employ Long Short-Term Memory (LSTM) networks to encode the complex intra-region correlations with respect to the distributions of abnormal events across time slots (from t_1 to t_K). In particular, LSTM consists of one cell state, and three controlled gates to update the cell state by performing write, read and reset operations, respectively. Formally, the updating functions for hidden states $\mathbf{h}^t_{i,j,l}$ and $\mathbf{c}^t_{i,j,l}$ corresponding to region $r_{i,j}$ and category c_l in t-th time step of input encoded sequence with length T in our LSTM encoder are presented as:

$$
\begin{aligned}
\mathbf{i}^t_{i,j,l} &= \sigma(\mathbf{W}_i \cdot \mathbf{h}^{t-1}_{i,j,l} + \mathbf{V}_i \cdot x^t_{i,j,l} + b_i) \\
o^t_{i,j,l} &= \sigma(\mathbf{W}_o \cdot \mathbf{h}^{t-1}_{i,j,l} + \mathbf{V}_o \cdot x^t_{i,j,l} + \mathbf{b}_o) \\
\mathbf{f}^t_{i,j,l} &= \sigma(\mathbf{W}_f \cdot h^{t-1}_{i,j,l} + V_f \cdot x^t_{i,j,l} + b_f) \\
\widetilde{c^t_{i,j,l}} &= \phi(\mathbf{W}_c \cdot \mathbf{h}^{t-1}_{i,j,l} + \mathbf{V}_c \cdot x^t_{i,j,l} + \mathbf{b}_c) \\
\mathbf{c}^t_{i,j,l} &= \mathbf{f}^t_{i,j,l} \odot \mathbf{c}^{t-1}_{i,j,l} + \mathbf{i}^t_{i,j,l} \odot \widetilde{\mathbf{c}^t_{i,j,l}} \\
\mathbf{h}^k_{i,j,l} &= \mathbf{o}^t_{i,j,l} \odot \phi(\mathbf{c}^t_{i,j,l})
\end{aligned} \tag{1}
$$

where $\mathbf{W}_* \in \mathbb{R}^{d_s \times d_s}$ represents the transformation matrix from the previous states (*i.e.*, $\mathbf{c}^{t-1}_{i,j,l}$ and $\mathbf{h}^{t-1}_{i,j,l}$) to LSTM cell and $\mathbf{V}_* \in \mathbb{R}^{d_x \times d_s}$ are the transformation matrices from input to LSTM cell. Here, d_x and d_s denotes the dimension of input vectors and hidden states, respectively. Furthermore, $\mathbf{b}_* \in \mathbb{R}^{d_s}$ is defined as a vector of bias term. $\sigma(\cdot)$ and $\phi(\cdot)$ represents the sigmoid and tanh function, respectively. The $\odot$ operator denotes the element-wise product. We denote the input gate, output gate and forget gate as $\mathbf{i}^t_{i,j,l}$, $\mathbf{o}^t_{i,j,l}$, and $\mathbf{f}^t_{i,j,l}$, respectively. For simplicity, we denote Eq. 1 as $\mathbf{h}^t_{i,j,l} = \text{LSTM}(*, \mathbf{c}^{t-1}_{i,j,l}, \mathbf{h}^{t-1}_{i,j,l})$ in the following subsections. While there exist other variations of recurrent neural networks, *e.g.*, gated recurrent unit (GRU), we choose LSTM for its general expressions.

3.2 Multi-Modal Pattern Fusion Module

While directly applying the recurrent neural network to solve the abnormal event forecasting problem is intuitive, the general recurrent neural network can do little to handle the complicated influencing factors from other geographical regions and event categories. Hence, we further employ an attention mechanism to adaptively capture the dynamic correlations from spatial and categorical view. Attention mechanisms are proposed to infer the importance of different parts of the training data, and let the learning algorithms focus on the most informative parts. Attention mechanism aims

to free the encoder-decoder architecture from the fixed-length internal representation by introducing a context vector to model the relevance. In addition, in order to differentiate regions and categories in the fusion process, we incorporate region embedding $\mathbf{e}_{r_{i,j}} \in \mathbb{R}^{d_e}$ and category embedding $\mathbf{e}_{c_j} \in \mathbb{R}^{d_e}$ as contextual information into the attention mechanism. Formally, the attention computation formulations in MiST are given:

$$\boldsymbol{\eta}^k_{i,j,l} = \tanh(\mathbf{W}^k(\mathbf{h}^k_{i,j,l}, \mathbf{e}_{r_{i,j}}, \mathbf{e}_{c_j}) + \mathbf{b}^k)$$
$$\alpha^k_{i,j,l} = \frac{\exp(\boldsymbol{\eta}^k_{i,j,l})}{\sum_{i,j\in G}\sum_{l=1}^{L}\exp(\boldsymbol{\eta}^k_{i,j,l})} \tag{2}$$

where we term the size of hidden representation vector in attention networks as *attention dimensionality* which is denoted as S. $\mathbf{W}^k \in \mathbb{R}^{d_s \times S}$ and $\mathbf{b}^k) \in \mathbb{R}^{d_s}$ are respectively the weight matrix and bias vector that project the input into hidden layers to get $\eta^k_{i,j,l}$ as hidden representation of $\mathbf{h}^k_{i,j,l}$. Then, we measure the importance of hidden representations of each region $r_{i,j}$ in the grid map $G = I \times J$ ($(i,j) \in G$) and category c_l ($c_l \in C$), and get a normalized importance weight $\alpha^k_{i,j,l}$ through a softmax function. The attention weights of our fusion module are jointly determined by the input spatial-categorical features and the encoded historical hidden states in the context-LSTM encoder. After obtaining the attention weights, the output hidden representation vector at time slot t_k is derived as:

$$\mathbf{q}^k = \sum_{i,j\in G}\sum_{l=1}^{L}\alpha^k_{i,j,l}\mathbf{h}^k_{i,j,l} \tag{3}$$

where $\mathbf{q}^k$ is the summarized concatenation of view-specific representations $\mathbf{h}^k_{i,j,l}$, describing what kind of influential factors will be considered as more important for abnormal event occurrences at target region $r_{i,j}$. In the training process of MiST, the developed the deep fusion module with attention mechanism is parametrized as a feed-forward neural network which can be trained with the whole neural network. Our proposed approach is very general, which can automatically learn the correlation weights of different views.

3.3 Conclusive Recurrent Network

So far we have developed two components of MiST to (i) model dynamic intra-region correlations from temporal views with the context-LSTM encoder; (ii) capture complex inter-region and cross-categorical correlations from spatial-categorical views with deep fusion module. With the above learned weights as coefficients, the summarized representation $\mathbf{q}^k$ are calculated as the weighted combinations of the view-specific representations with the voting weights (*i.e.*, $\alpha^k_{i,j,l}$) of different views.

To integrate the complex patterns encoded from spatial-categorical factors with the temporal patterns under the MiST framework, we propose to encode the multi-dimensional patterns with recurrent neural networks and model the complex location-time-category interactions with latent-space representations. In this paper, we apply LSTM as the recurrent unit and formulate it as follows:

$$\boldsymbol{\xi}_k = \text{LSTM}(\mathbf{q}_{k-1}, \boldsymbol{\xi}_{k-1}) \tag{4}$$

The joint embedding $\boldsymbol{\xi}$ maps all spatial, temporal, and categorical units into a common latent space. The developed conclusive recurrent networks provide a flexible way to let different views to collaborate with each other. By combining this spatial and categorical contextual signals with the current recurrent temporal status, our MiST framework could predict future abnormal events based on not only the sequential relation but also the spatial correlations across regions and co-occurrence relationships across categories.

3.4 Forecasting and Model Inference

Finally, we utilize the Multilayer Perceptron (MLP) component to decode the occurrence probability by capturing the non-linear dependencies between elements in hidden vectors. Formally, we represent MLP as follows:

$$\psi_1 = \phi(\mathbf{W}_1 \cdot \lambda_1 + \mathbf{b}_1)$$
$$\cdots\cdots$$
$$\psi_N = \phi(\mathbf{W}_n \cdot \lambda_n + \mathbf{b}_n)$$
$$y^{l,k}_{i,j} = \sigma(\mathbf{W}' \cdot L_n + \mathbf{b}') \tag{5}$$

where N represents the number of hidden layers in MLP (indexed by n). For the ψ_n layer, $\mathbf{W}_n$ and $\mathbf{b}_n$ represents the weight matrix and bias vector, respectively. We use *ReLU* (denoted as $\phi(\cdot)$) as the activation function of the fully connected layers and further specify the activation function as *sigmod* (denoted as $\sigma(\cdot)$) to output the abnormal event occurrence probability of category c_l at region $r_{i,j}$ in time slot t_k, *i.e.*, $y^{l,k}_{i,j}$.

In general, our abnormal event occurrence forecasting can be regarded as a classification problem. We utilize cross entropy as the metric in our loss function which is defined as follows:

$$\mathscr{L} = -\sum_{(i,j,l,k)\in S} y^{l,k}_{i,j}\log\hat{y}^{l,k}_{i,j} + (1 - y^{l,k}_{i,j})\log(1 - \hat{y}^{l,k}_{i,j}) \tag{6}$$

where $\hat{y}^{l,k}_{i,j}$ denotes the estimated probability of the l-th category of abnormal events occur at region $r_{i,j}$ in k-th time slot. Here, S is the set of abnormal events in the training process. The model parameters can be derived by minimizing the loss function. In this work, we use Adam optimizer [16] to learn the parameters of MiST.

4 EVALUATION

In this section, we conduct extensive experiments on three real-world abnormal event datasets (*i.e.*, crime and urban anomaly data) collected from New York City (NYC) and Chicago to verify the effectiveness of our developed *MiST* framework. We then analyze the experimental results and demonstrate the accuracy promotion by comparing it with various baselines. We perform experiments to answer the following questions:

- **Q1**: Compared with the state-of-the-art forecasting techniques, can our *MiST* framework achieve comparable accuracy in predicting both citywide crimes and anomalies across different cities?
- **Q2**: Does *MiST* consistently outperform other algorithms with respect to different time periods?

Algorithm 1: The Learning Process of MiST Model.

Input: Historical Abnormal Event Sequences Y, Neighbor Region Set G, Category Set L, Sequence Length T, and Batch Size b_{size}.

Paras: Embedding Matrices $e_r \in \mathbb{R}^{(I*J)\times d_e}$, $e_c \in \mathbb{R}^{C\times d_e}$, and Other Hidden Parameters θ.

Initialize all parameters;

// Sample a minibatch of size b_{size}.

foreach $\mathsf{T}_{\text{batch}} = \text{sample}(X, b_{\text{size}})$ **do**

  foreach $\langle i, j, l, t\rangle \in \mathsf{T}_{\text{batch}}$ **do**

    foreach $\langle i', j'\rangle \in G[(i, j)]$ **do**

      $R_{i,j} = e_r[(i, j) :]$;

      // Lookup region embeddings

      foreach $l' \in L$ **do**

        $C_l = e_c[l, :]$;

        // Lookup category embeddings

        $c^0_{i',j',l'} = c^0, h^0_{i',j',l'} = h^0$// init. hidden states

        for $t' \leftarrow (k - s)$ **to** $(k - 1)$ **do**

          $c^t_{i',j',l'}, h^t_{i',j',l'} = \text{LSTM}(X[i', j', l', t'], c^{t-1}_{i',j',l'}, h^{t-1}_{i',j',l'})$;

          // Eq.1

        end

      end

    end

    $\hat{T}_k = \text{Attention}(h^k_{:,:,:}, [h^k_{:,:,:}; R_{:,:}; C_{:}])$ $\hat{y}^t_{i,j,l} = \text{MLP}(\hat{T}_k)$

    $y^t_{i,j,l} = X[i, j, l, t]$ Update loss $\mathcal{L}$

  end

  Update all parameters w.r.t $\mathcal{L}$;

end

- **Q3**: How does our *MiST* model work for forecasting abnormal events with different categories compared with state-of-the-art techniques?
- **Q4**: How is the performance of *MiST* variants with different combinations of key components in the joint framework?
- **Q5**: How is the performance of *MiST* with different spatial and temporal scales?
- **Q6**: How do different hyperparameter settings affect the forecasting performance of *MiST*?
- **Q7**: How is the interpretation of our *MiST* framework in capturing dynamic importance weights across spatial and category dimensions when predicting citywide abnormal events?

4.1 Data Description

4.1.1 **Data Statistics.** We collect three abnormal event datasets with two different types from NYC and Chicago, *i.e.*, two crime data and one urban anomaly data, and separately perform experiments of predicting the occurrences of each categorical urban crimes and anomalies at each geographical region in a city. The basic statistics of the three datasets are shown in Table 1. In our experiments, we focus on several key categories and consider other categories as external type. We also show the geographical distributions of abnormal events with respect to different categories and time periods in Figure 2. Their details are introduced as follows:

Table 1: Dataset Statistics.

Data Source	**Crime Reports from NYC**	
Time Span	From Jan, 2015 to Dec, 2015	
Category	Burglary	Robbery
Number of Instances	14,967	16,886
Category	Felony Assault	Grand Larceny
Number of Instances	20,189	41,873
Data Source	**Anomaly Reports from NYC**	
Time Span	From Jan, 2014 to Dec, 2014	
Category	Noise	Illegal Parking
Number of Instances	134,690	57,374
Category	Blocked Driveway	Building/Use
Number of Instances	74,698	24,319
Data Source	**Crime Reports from Chicago**	
Time Span	From Jan, 2015 to Dec, 2015	
Category	Burglary	Robbery
Number of Instances	13,103	9,633
Category	Narcotics	Theft
Number of Instances	21,607	56,695
Category	Assault	Battery
Number of Instances	16,992	48,824
Category	Criminal Damage	Deceptive Practice
Number of Instances	28,590	14,084

- **NYC Crime Data (NYC-C)**. This crime dataset records multiple categories of crime reports (*e.g.*, burglary, robbery) in NYC and each crime record is in the format of (crime category, latitude, longitude, timestamp). This data was collected from Jan 2015 to Dec 2015.
- **NYC Urban Anomaly Data (NYC-A)**. This dataset spanning from Jan 2014 to Dec, 2014 is collected from 311 non-emergency services in New York City which records urban anomalies of different categories (*i.e.*, noise, illegal parking). Each reported anomaly record is formatted as (anomaly category, latitude, longitude, timestamp).
- **Chicago Crime Data (CHI-C)**. This crime data was collected from Chicago during 2015 to Dec 2015, which documents different categorical crime reports (*e.g.*, theft, narcotics). The data format is similar as that of NYC crime data.

4.2 Experimental Settings

4.2.1 **Parameter Settings.** In our experiments, we leverage Adam as the optimizer in the learning process of *MiST* and implemented *MiST* with TensorFlow architecture. We set the hidden state dimensionality d_s as 32 and embedding size d_e as 32. The attention dimension and the number of MLP layers is set to 32 and 3, respectively. Additionally, the batch size and learning rate is set to 64 and 0.001, respectively.

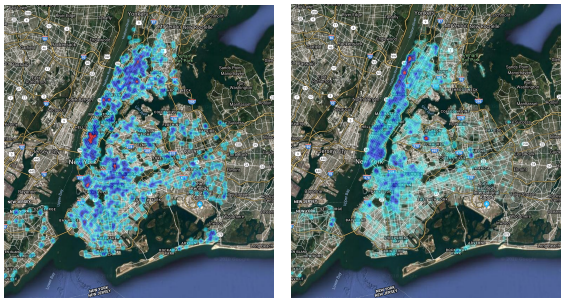
Table 2: Crime forecasting results across different categories in terms of *Macro-F1* and *Micro-F1*.

City	Month	Metrics	SVR	ARIMA	LR	ST-RNN	GRU	RDN	HRN	ARM	*MiST*
NYC-C	Aug	Macro-F1	0.3960	0.4090	0.4325	0.5070	0.5099	0.5102	0.4893	0.5031	**0.5384**
		Micro-F1	0.3010	0.3210	0.4108	0.4530	0.4605	0.4620	0.4650	0.3298	**0.5050**
	Sep	Macro-F1	0.4160	0.4190	0.4526	0.4891	0.4910	0.5015	0.4856	0.4868	**0.5365**
		Micro-F1	0.3210	0.3370	0.3881	0.4415	0.4439	0.4582	0.4627	0.3099	**0.5054**
	Oct	Macro-F1	0.4180	0.4190	0.4345	0.4903	0.4906	0.4907	0.4748	0.4907	**0.5110**
		Micro-F1	0.3210	0.3300	0.3727	0.4340	0.4330	0.4444	0.4451	0.3046	**0.4739**
	Nov	Macro-F1	0.4280	0.4290	0.4246	0.4705	0.4719	0.4896	0.4945	0.4908	**0.5066**
		Micro-F1	0.3280	0.3430	0.3768	0.4045	0.4370	0.4389	0.4478	0.3017	**0.4772**
	Dec	Macro-F1	0.4240	0.4150	0.3875	0.4683	0.4805	0.4806	0.4639	0.4815	**0.4920**
		Micro-F1	0.3320	0.3320	0.3450	0.3977	0.4147	0.4303	0.4117	0.3001	**0.4563**
CHI-C	Aug	Macro-F1	0.5004	0.5121	0.4838	0.5349	0.5189	0.5278	0.5391	0.5382	**0.5896**
		Micro-F1	0.3108	0.3302	0.3812	0.3949	0.4203	0.3728	0.3886	0.3871	**0.4663**
	Sep	Macro-F1	0.5159	0.5164	0.4733	0.4898	0.4796	0.5145	0.5095	0.5037	**0.5637**
		Micro-F1	0.3293	0.3412	0.3913	0.4292	0.4227	0.3551	0.3487	0.3408	**0.4649**
	Oct	Macro-F1	0.5141	0.5201	0.4671	0.4622	0.5341	0.5292	0.5354	0.5290	**0.5793**
		Micro-F1	0.3275	0.3432	0.2948	0.3665	0.4007	0.3781	0.3882	0.3817	**0.4695**
	Nov	Macro-F1	0.5111	0.5113	0.4941	0.5383	0.5181	0.5270	0.5330	0.5283	**0.5634**
		Micro-F1	0.3314	0.3417	0.3819	0.4168	0.3795	0.3877	0.3971	0.3764	**0.4564**
	Dec	Macro-F1	0.5034	0.5048	0.4802	0.5337	0.5221	0.5338	0.5103	0.5285	**0.5805**
		Micro-F1	0.3271	0.3343	0.3528	0.4053	0.3993	0.4067	0.3682	0.3987	**0.4783**
NYC-A	Aug	Macro-F1	0.6181	0.6072	0.5788	0.6455	0.6008	0.6496	0.6561	0.6511	**0.7119**
		Micro-F1	0.5405	0.5192	0.5614	0.5925	0.5441	0.5937	0.5942	0.5752	**0.6802**
	Sep	Macro-F1	0.6177	0.6204	0.5681	0.6797	0.6484	0.6733	0.6820	0.6670	**0.7244**
		Micro-F1	0.5353	0.5302	0.5268	0.6154	0.6115	0.6051	0.6032	0.5912	**0.6773**
	Oct	Macro-F1	0.6294	0.6303	0.5901	0.6625	0.6424	0.6711	0.6812	0.6672	**0.7278**
		Micro-F1	0.5436	0.5412	0.5385	0.5957	0.6072	0.6013	0.6173	0.5900	**0.6979**
	Nov	Macro-F1	0.6136	0.6214	0.5915	0.6559	0.6423	0.6541	0.6690	0.6463	**0.7034**
		Micro-F1	0.5349	0.5452	0.5670	0.5851	0.5891	0.5813	0.6007	0.5654	**0.6663**
	Dec	Macro-F1	0.5947	0.5849	0.5556	0.6135	0.6231	0.6178	0.6370	0.6247	**0.6895**
		Micro-F1	0.5126	0.5234	0.5212	0.5265	0.5495	0.5323	0.5183	0.5218	**0.6309**

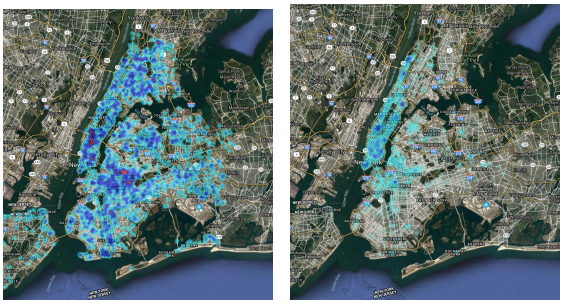
4.2.2 **Baseline Methods.** *MiST* is measured against the following baseline methods from five categories: (i) the conventional time series forecasting methods (*i.e.*, SVR and ARIMA); (ii) conventional supervised learning algorithm (*i.e.*, LR); (iii) recurrent neural network and its variants for spatial-temporal data forecasting (*i.e.*, ST-RNN and GRU); (iv) advanced neural network models for time series and sequence modeling (*i.e.*, RDN, HRN and ARM). The details are presented as follows:

- **Support Vector Regression (SVR)** [3]: it is a supervised learning model for time series regression analysis based on kernel functions which are characterized by the margin and the number of support vectors.
- **Auto-Regressive Integrated Moving Average (ARIMA)** [15]: it is a conventional time series prediction model for understanding and predicting future values in a time series which is in conjunction with stationary and linear transformations.
- **Logistic Regression (LR)** [11]: it is a statistical model which forecasts each region's abnormal event occurrences based on temporal features (e.g., the day of a week and the month of a year) extracted from historical crime logs.
- **Spatial-Temporal Recurrent Neural Network (ST-RNN)**[25]: it utilizes the recurrent neural networks to capture the spatial and periodical temporal contexts in modeling time-ordered data from each time intervals.
- **Gated Recurrent Framework (GRU)** [6]: it is a time-ordered sequence modeling approach based on gated recurrent unit and encodes the input time series into latent representations.
- **Recurrent Deep Networks (RDN)** [12]: this method is a recurrent deep network framework to consider long-term dependencies in recurrent neural networks, and aims to predict spatial events via jointly modeling normal dynamic temporal patterns.
- **Attentive Hierarchical Recurrent Networks (HRN)** [14]: it is a attention-based hierarchical recurrent framework that is capable of capturing the dynamic sequential patterns and their inherent interrelationships with other ubiquitous data.
- **Attention-based Recurrent Model (ARM)** [9]: this method leverages recurrent framework and attention mechanisms to interpret the relations between the current and past values in predicting spatial-temporal mobility data.

4.2.3 **Evaluation Protocols.** In our experiments, we chronologically split the dataset into training (6.5 months), validation (0.5 month), and test (1 month) sets. The validation set was used for tuning hyper-parameters and the final performance comparison was conducted on the test set. We partitioned New York City and



(a) Crime-Burglary-Aug (b) Anomaly-Noise-Aug



(c) Crime-Burglary-Nov (d) Anomaly-Noise-Nov

Figure 2: Distributions of abnormal events in NYC.

Chicago into 248 and 189 disjointed geographical regions, respectively, where each region is a $2km \times 2km$ grid, following the similar settings in [32, 43]. Based on the region partition results, we can map each abnormal event (*i.e.*, report of crime or urban anomaly) into an individual geographical region and generate the inputs for *MiST* framework. In this work, We adopted two types of evaluation metrics to fully evaluate all methods:

- (i) We use *Marco-F1* and *Micro-F1* [10, 33] to evaluate the prediction accuracy across different crime categories. These metrics indicate the overall performance across different classes. The mathematical definitions of Marco-F1 and Micro-F1 are given: Micro-F1 $= \frac{1}{J} \cdot \sum_{j=1}^{J} \frac{2 \cdot TP_j}{2 \cdot TP_j + FN_j + FP_j}$ and Macro-F1 $= \frac{2 \cdot \sum_{j=1}^{J} TP_j}{2 \cdot \sum_{j=1}^{J} TP_j + \sum_{j=1}^{J} FN_j + \sum_{j=1}^{J} FP_j}$. where J is the total number of event categories. A higher Micro-F1 and Macro-F1 score indicates better performance.
- (ii) we use *F1-score* (harmonic mean to balance precision and recall) and *AUC* [8] to evaluate the accuracy in predicting an individual category of abnormal event occurrence. A higher F1-score and AUC value reflects a better forecasting accuracy.

To ensure the fairness of performance comparison for all compared methods, our experiments are conducted by forecasting occurrences of abnormal events in consecutive days in the time period of test set. In the evaluation results, the average performance is reported across all days in the test time period.

4.3 Performance Comparison (Q1, Q2 and Q3)

4.3.1 **Overall Comparison (Q1).** Table 2 shows both the crime and urban anomaly forecasting accuracy across categories on different cities in terms of Macro-F1 and Micro-F1. From the evaluation results, we summarize three key observations as follows:

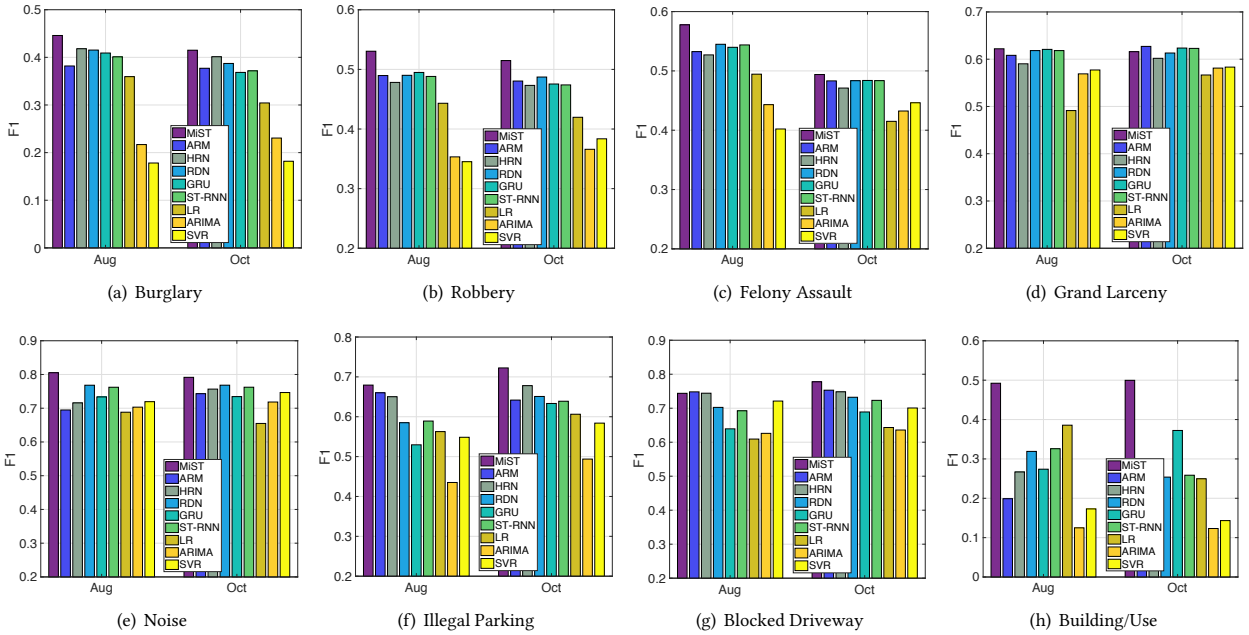
First and foremost, *MiST* is significantly better than other different types of neural network-based methods. For example, for predicting crimes in Chicago, *MiST* achieves relatively 9.6% and 30.9% improvements over the best performed baseline (*i.e.*, RDN) in terms of Macro-F1 and Micro-F1 on September. This sheds lights on the benefit of our multi-view model which jointly considers the spatial, temporal, and semantic relationships.

Second, the performance of neural network-based methods are superior to that of conventional time series forecasting techniques and supervised learning approach. This is due to the factors that: (1) conventional time series forecasting methods only emphasize one fixed temporal pattern, rather than the time-evolving temporal dependencies; (2) the neural network-based methods could capture complex inherent structures of multi-dimensional spatial-temporal data in a non-linear manner, which is more advantageous.

Lastly, there is no obvious winner among the recurrent network network variants (*i.e.*, ST-RNN and GRU) and deep sequential data modeling solutions (*i.e.*, RDN, HRN and ARM). This again confirms that only considering the data dependencies from temporal dimension is insufficient to predict both crime and urban anomaly occurrence well. In contrast, *MiST* dynamically associates weights for latent spatial, temporal and categorical correlations, which shows remarkable flexibility and superiority.

4.3.2 **Forecasting Accuracy v.s. Time Periods (Q2).** To make thorough evaluation, we conduct comparison experiment of *MiST* and all baselines across different training and test time periods. We can note that the best performance is consistently achieved by *MiST* with different forecasting time periods. In addition, another observation is that the performance gain between *MiST* and other baselines is relatively stable when sliding the training and testing time windows, which reflects the robustness of our model in learning the dynamic abnormal event distributions over time.

4.3.3 **Forecasting Accuracy v.s. Categories (Q3).** Furthermore, we investigate the effectiveness of *MiST* in predicting each individual category of abnormal events and report the evaluation results in Figure 4 and Figure 3 for crimes in Chicago and crimes as well as urban anomalies in NYC. We can observe that our *MiST* achieves the best performance in all cases. For example, *MiST* outperforms the best performed baseline (*i.e.*, RDN) by 13.2% on average in forecasting burglaries and 12.4% on average in forecasting illegal parking in NYC. Additionally, another interesting observation is that obvious improvements can also be obtained by *MiST* in predicting abnormal events of sparser categories, *e.g.*, 84.1% average relatively improvement over the state-of-the-art approach ST-RNN for building/use prediction. This observation indicates that *MiST* is capable of effectively modeling latent correlations from different contextual modalities (region, time, category), to alleviate the data scarcity problem in forecasting sparse anomalies.



(a) Burglary (b) Robbery (c) Felony Assault (d) Grand Larceny

(e) Noise (f) Illegal Parking (g) Blocked Driveway (h) Building/Use

Figure 3: Forecasting results for individual category in terms of *F1-score* on NYC crime and anomaly data.

4.4 Component-Wise Evaluation of *MiST* (Q4)

In addition to comparing *MiST* with state-of-the-art techniques, we also aim to get a better understanding of key components of *MiST* by studying the effect of different view components. We denote the full version of our method as as *MiST*.

- **Spatial-View + Temporal View** *MiST*-st: This variant captures both spatial and temporal dependencies in predicting abnormal events, *i.e.*, does not consider cross-categorical influences.
- **Category-View + Temporal View** *MiST*-ct: This variant considers both categorical and temporal views, *i.e.*, without considering inter-region spatial correlations.
- **Temporal View** *MiST*-t: For this variant, we only use LSTM with temporal attention mechanism to consider the temporal dimension of muli-view data, *i.e.*, without considering contextual signals from both spatial and categorical views.

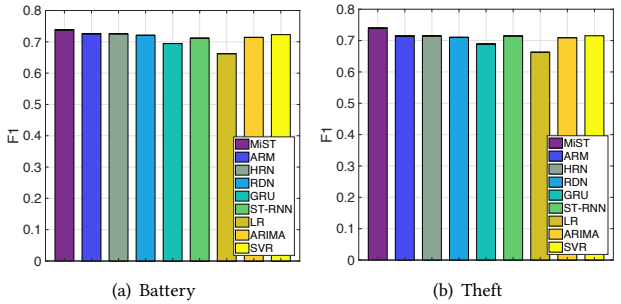
We report the evaluation results on crime dataset in Figure 5. We can notice that the full version of our developed framework *MiST* achieves the best performance in all cases. As such, it is necessary to build a joint framework to capture spatial view (inter-region spatial correlations), temporal view (intra-region temporal correlations), as well as categorical view (cross-categorical dependencies) simultaneously for forecasting citywide abnormal events.

4.5 Effect of Spatial and Temporal Scale (Q5)

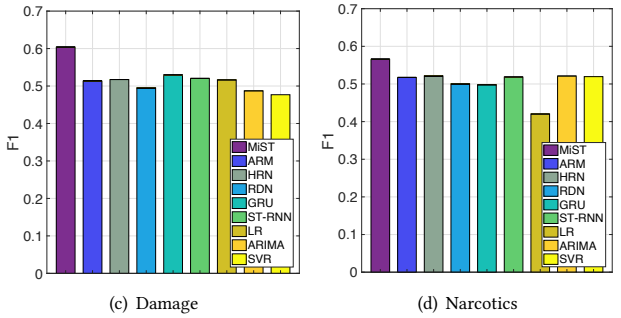
In our evaluations, we further investigated the effect of spatial and temporal scale in *MiST*, *i.e.*, the geographical scale of grid map $G = I \times J$ in the event context tensor $\mathcal{A}$ ($I = J$ in our experiments) and the length of encoded sequence T in our developed recurrent framework. The evaluation results are presented in Figure 6. There are two interesting insights from the result: (i) Figure 6(a) reveals that increasing I and J improves performance, which is not surprising since we consider a larger geographical space to learn jointly representations from mutli-views. Furthermore, as the spatial scale of the event context tensor increases, the performance slightly increases but mainly remains stable when $I = J = 11$. One potential reason is that when considering larger geographical coverage, more parameters need to be learned. As a result, the training of *MiST* becomes harder. (ii) As depicted in Figure 6(b), the performance of *MiST* becomes better as the encoded sequence length T increases and tends to saturate once T reaches 10. Due to space limit, we only report the results of crime data on Oct. Similar results can be observed on urban anomaly dataset and other time periods.

4.6 Hyperparameters Studies (Q6)

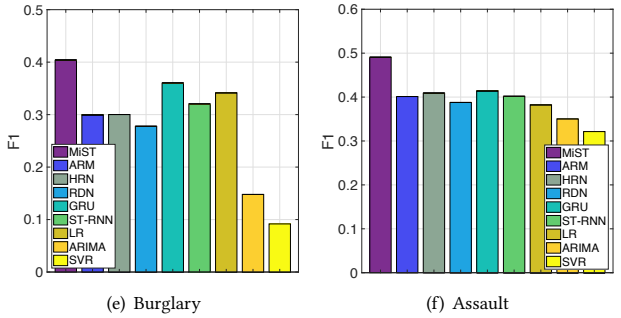
To investigate the robustness of *MiST* framework, we examine how the different choices of parameters affect the performance of *MiST* in the prediction performance. Except for the parameter being tested, we set other parameters at the default values. Figure 7 and Figure 8 shows the evaluation results as a function of one selected parameter when fixing others. Overall, we observe that *MiST* is not strictly sensitive to these parameters in two tasks and is able to reach high performance under a cost-effective parameter choice, which demonstrates the robustness of *MiST*. Furthermore, we can observe that the increase of prediction performance saturates as the representation dimensionality reaches around 32. This is because: at the beginning, a larger value of hidden state dimensionality d_s and attention dimension S brings a stronger representation power for the recent framework and attention network, but the further increase of dimension size of latent representations to encode abnormal event occurrence patterns from multi-views might lead to the overfitting issue. In our experiments, we set the dimension



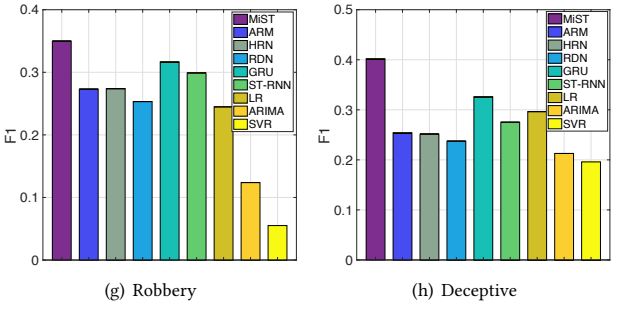
(a) Battery (b) Theft



(c) Damage (d) Narcotics

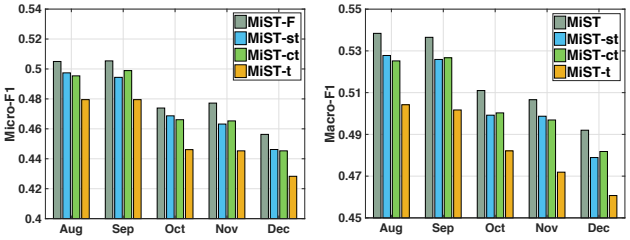


(e) Burglary (f) Assault



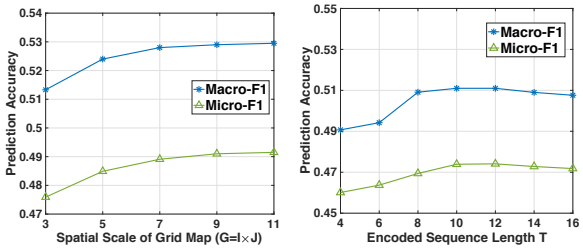
(g) Robbery (h) Deceptive

Figure 4: Forecasting results for individual category in terms of *F1-score* on Chicago crime data.



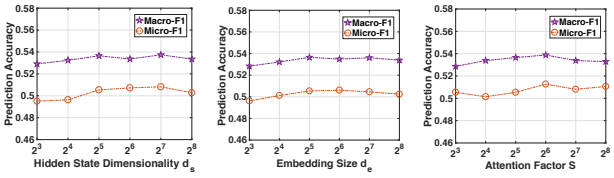
(a) Micro-F1 (b) Macro-F1

Figure 5: Evaluation on MiST variants.



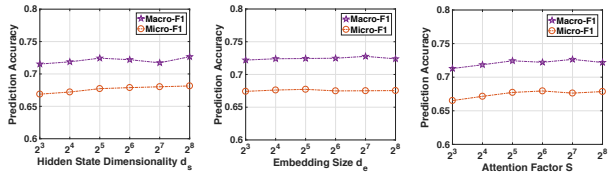
(a) Spatial Scale Effect (b) Temporal Scale Effect

Figure 6: Effect of spatial and temporal scale in *MiST* on crime data in terms of *Macro-F1* and *Micro-F1* on Oct.



(a) Hidden State Dimensionality d_s (b) Embedding Size d_e (c) Attention Dimension S

Figure 7: Parameter sensitivity of *MiST* in predicting crimes in NYC in terms of *Macro-F1* and *Micro-F1* on Sep.



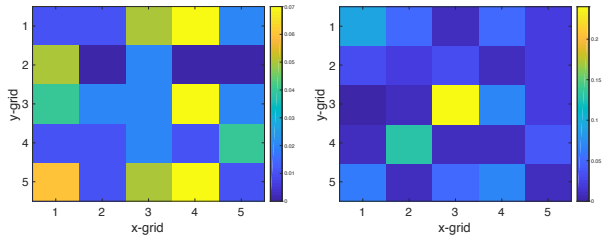
(a) Hidden State Dimensionality d_s (b) Embedding Size d_e (c) Attention Dimension S

Figure 8: Parameter sensitivity of *MiST* in predicting urban anomalies in NYC in terms of *Macro-F1* and *Macro-F1* on Sep.

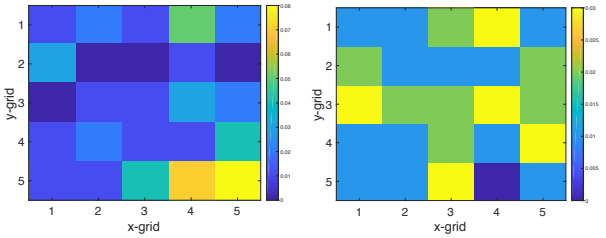
size as 32 due to the consideration of the trade-off between the effectiveness and computational cost.

4.7 Case Study (Q7)

Apart from the superior forecasting performance, another key advantage of *MiST* is its ability in interpreting the importance weights of spatial and categorical correlations in predicting a specific category of abnormal event at a region. To demonstrate this, we perform case studies to show the explainability of our framework by visualizing the attention weights obtained from *MiST* in predicting burglaries with a 5×5 grid map (the target region is in the middle of this map) in NYC, as shown in Figure 9. We could observe that



(a) Robbery (b) Burglary



(c) Felony Assault (d) Grand Larceny

Figure 9: The importance weights of across regions and crime categories learned from *MiST* in predicting burglaries with a 5×5 grid map in NYC.

MiST enables the dynamic modeling of correlations between the target region and other regions, as well as latent influences between the target event category (burglary) and other categories.

5 RELATED WORK

We introduce the research work which are related to our study.

Ubiquitous Sensing Applications. Ubiquitous sensing has emerged as a new sensing paradigm that empowers average people to contribute their observations and measurements about the physical world. Numerous novel applications have been developed to address various challenges in smart cities applications [4, 18, 21, 32, 36, 37, 41, 42], such as intelligent transportation [4], privacy-preserving [37], business assessment [21] and human activity modeling [41]. However, the abnormal event prediction problem in ubiquitous sensing scenario remain as a challenging problem to be solved. In this paper, we develop an end-to-end model to tackle this problem by addressing the unique challenges.

Spatial Anomaly Detection and Event Forecasting. There exists a good amount of work on the topics *geographical anomaly detection* [7, 17, 44, 46, 47]. For example, Doan *et al.* detected anomalies by modelling the behaviours of pedestrian flows across multiple locations [7]. Zheng *et al.* identified the anomalies from spatial-temporal data using multiple data sources [47]. However, all above solutions identified the geographical anomalies *after they happen* which might lead to the inefficiency to handle the anomalies beforehand. Additionally, while several recent studies aim to predict social events using social media data [27, 31, 45], these work only consider obvious social events which can be identified based on the external content (*e.g.*, keywords) from social media platforms (*e.g.*, Twitter). In contrast, this paper solves a new problem of different categories of abnormal event prediction at each region of a city with a multi-view and multi-modal deep neural network architecture.

Sequential Data Modeling. A relevant body of work in the machine learning and data mining communities have investigated modeling sequential data [9, 12, 13, 35, 38, 48]. For example, Autoregression-based models (*e.g.*, ARIMA) have been proposed to study the problem of time series prediction [19]. Compared to traditional time series data, temporally-ordered abnormal event sequence has its own characteristics, *e.g.*, complex spatial-temporal correlations and dynamic multi-dimensional interactions. Recently, Recurrent neural networks (RNNs) become an effective tool due to their success in sequence modeling [1]. For example, Hu *et al.* proposed to capture sequential structures of subevents with LSTM networks [12]. Wu *et al.* developed a RNN based model to predict future behavioral trajectories of users in online recommendation platforms [35]. However, these works can only capture temporal dependency in time series, which ignore the various characteristics of geolocation data from multi-views. To overcome this problem, our MiST framework aims to forecast the abnormal event data in urban space via modeling latent correlations from spatial-temporal-categorical views.

Attention Mechanism. The attention mechanism has been proposed to strengthen not only the ability of RNN in capturing the long-range dependencies but also the interpretability in image analysis [5] and nature language processing [22]. Later, a number of studies developed attention-based models to encode hidden states in different time series applications, such as patients' health information prediction [28] and online personalized recommendation [2]. Inspired by the above work, this work proposes a deep fusion module based on attention mechanism. Different from the above attention-based methods which only focus on sequential patterns, we propose a novel multi-view attention model which fuses view-specific representations on different dimensions.

6 CONCLUSION

This paper explored a new neural network architecture MiST to explicitly model the dynamic patterns of citywide abnormal events from spatial-temporal-categorical views. We integrated recurrent neural networks with a multi-modal pattern fusion module to model spatial-temporal correlations. We evaluated our new MiST framework on different types of real-world datasets and the results showed that our approach achieves better performance when competing with several baselines. There are several future directions for our work. *First*, we are interested in extending the current method to identify the causalities for the occurrences of different categorical abnormal events, which is useful for suggesting relevant public policy making. For future work, it is promising to explore the factors underlying abnormal event occurrences and how different categorical event propagate with respect to spatio-temporal dimensions. *Second*, in this paper, we evaluated our method on three real-world datasets, *i.e.*, crime and anomaly data, due to limited data availability. In fact, the framework of MiST is general, and it is feasible to apply our model on other multi-dimensional time-stamped sequence data.

ACKNOWLEDGMENTS

This work is supported in part by the Army Research Laboratory under Cooperative Agreement Number W911NF- 09-2-0053 and the National Science Foundation (NSF) grants CNS- 1629914 and IIS-1447795. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *NIPS*. 1171–1179.

[2] Da Cao, Xiangnan He, Lianhai Miao, Yahui An, Chao Yang, and Richang Hong. 2018. Attentive Group Recommendation. In *Special Interest Group on Information Retrieval (SIGIR)*.

[3] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: a library for support vector machines. *Transactions on Intelligent Systems and Technology (TIST)* 2, 3 (2011), 27.

[4] Longbiao Chen, Daqing Zhang, Leye Wang, Dingqi Yang, et al. 2016. Dynamic cluster-based over-demand prediction in bike sharing systems. In *International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp)*. ACM, 841–852.

[5] Jongwon Choi, Hyung Jin Chang, Jiyeoup Jeong, Yiannis Demiris, and Jin Young Choi. 2016. Visual tracking using attention-modulated disintegration and integration. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 4321–4330.

[6] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, et al. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[7] Minh Tuan Doan, Sutharshan Rajasegarar, Mahsa Salehi, Masud Moshtaghi, and Christopher Leckie. 2015. Profiling pedestrian distribution and anomaly detection in a dynamic environment. In *International Conference on Information and Knowledge Management (CIKM)*. ACM, 1827–1830.

[8] Yuxiao Dong, Jing Zhang, Jie Tang, Nitesh V Chawla, and Bai Wang. 2015. Coupledlp: Link prediction in coupled networks. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 199–208.

[9] Jie Feng, Yong Li, Chao Zhang, and etc. 2018. DeepMove: Predicting Human Mobility with Attentional Recurrent Networks. In *International World Wide Web Conference (WWW)*. International World Wide Web Conferences Steering Committee, 1459–1468.

[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 855–864.

[11] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. 2013. *Applied logistic regression*. Vol. 398. John Wiley & Sons.

[12] Linmei Hu, Juanzi Li, Liqiang Nie, Xiao-Li Li, and Chao Shao. 2017. What Happens Next? Future Subevent Prediction Using Contextual Hierarchical LSTM.. In *International Conference on Artificial Intelligence (AAAI)*. 3450–3456.

[13] Chao Huang, Xian Wu, and Dong Wang. 2016. Crowdsourcing-based urban anomaly prediction system for smart cities. In *International on Conference on Information and Knowledge Management (CIKM)*. ACM, 1969–1972.

[14] Chao Huang, Junbo Zhang, Yu Zheng, and Nitesh V Chawla. 2018. DeepCrime: Attentive Hierarchical Recurrent Networks for Crime Prediction. In *International Conference on Information and Knowledge Management (CIKM)*. ACM, 1423–1432.

[15] Mehdi Khashei, Mehdi Bijari, and Gholam Ali Raissi Ardali. 2009. Improvement of auto-regressive integrated moving average models using fuzzy logic and artificial neural networks (ANNs). *Neurocomputing* 72, 4-6 (2009), 956–967.

[16] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Viet Duc Le, Hans Scholten, and Paul Havinga. 2013. Flead: Online frequency likelihood estimation anomaly detection for mobile sensing. In *International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp)*. ACM, 1159–1166.

[18] Ruirui Li, Jyun-Yu Jiang, Chelsea J-T Ju, and Wei Wang. 2019. CORALS: Who Are My Potential New Customers? Tapping into the Wisdom of Customers' Decisions. In *International Conference on Web Search and Data Mining (WSDM)*. ACM, 69–77.

[19] Xiaolong Li, Gang Pan, Zhaohui Wu, and etc. 2012. Prediction of urban human mobility using large-scale taxi traces and its applications. *Frontiers of Computer Science* 6, 1 (2012), 111–121.

[20] Defu Lian, Kai Zheng, Yong Ge, Longbing Cao, Enhong Chen, and Xing Xie. 2018. GeoMF++: scalable location recommendation via joint geographical modeling and matrix factorization. *Transactions on Information Systems (TOIS)* 36, 3 (2018), 33.

[21] Jianxun Lian, Fuzheng Zhang, Xing Xie, and Guangzhong Sun. 2017. Restaurant survival analysis with heterogeneous information. In *International World Wide Web Conference (WWW)*. ACM, 993–1002.

[22] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, Vol. 1. 2124–2133.

[23] Christoph Ling, Jerome Kunegis, Sergej Sizov, and Steffen Staab. 2014. Detecting no-gaussian geographical topics in tagged photo collections. In *International Conference on Web Search and Data Mining (WSDM)*. ACM, 603–612.

[24] Fu-Yanjie Yao Zijun Xiong Hui Liu, Bin. 2013. Learning gepgrapihcal preferences for point-of-interest recommendation. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 1043–1051.

[25] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the Next Location: A Recurrent Model with Spatial and Temporal Contexts.. In *International Conference on Artificial Intelligence (AAAI)*. 194–200.

[26] Yu Liu, Baojian Zhou, Feng Chen, and David Cheung. 2017. Graph topic scan statistic for spatial event detection. In *International Conference on Information and Knowledge Management (CIKM)*. ACM, 489–498.

[27] Zhiwei Liu, Yang Yang, Zi Huang, and etc. 2017. Event early embedding: predicting event volume dynamics at early stage. In *Special Interest Group on Information Retrieval (SIGIR)*. ACM, 997–1000.

[28] Fenglong Ma, Radha Chitta, Jing Zhou, Quanzeng You, Tong Sun, and Jing Gao. 2017. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 1903–1911.

[29] Bei Pan, Ugur Demiryurek, and Cyrus Shahabi. 2012. Utilizing real-world transportation data for accurate traffic prediction. In *ICDM*. IEEE, 595–604.

[30] Goce Ristanoski, Wei Liu, and James Bailey. 2013. Time seriess forecasting using distribution enhanced linear regression. In *PAKDD*. Springer, 484–495.

[31] Minglai Shao, Jianxin Li, Feng Chen, and etc. 2017. An efficient approach to event detection and forecasting in dynamic multivariate social media networks. In *WWW*. ACM, 1631–1639.

[32] Hongjian Wang, Daniel Kifer, Corina Graif, and Zhenhui Li. 2016. Crime rate inference with big data. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 635–644.

[33] Jinpeng Wang, Gao Cong, Wayne Xin Zhao, and Xiaoming Li. 2015. Mining User Intents in Twitter: A Semi-Supervised Approach to Inferring Intent Categories for Tweets.. In *International Conference on Artificial Intelligence (AAAI)*. 318–324.

[34] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. 2018. IntelliLight: a reinforcement learning approach for intelligent traffic light control. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM.

[35] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent recommender networks. In *International Conference on Web Search and Data Mining (WSDM)*. ACM, 495–503.

[36] Xian Wu, Yuxiao Dong, Chao Huang, Jian Xu, Dong Wang, and Nitesh V Chawla. 2017. Uapd: Predicting urban anomalies from spatial-temporal data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*. Springer, 622–638.

[37] Dingqi Yang, Daqing Zhang, Bingqing Qu, and Philippe Cudré-Mauroux. 2016. PrivCheck: privacy-preserving check-in data publishing for personalized location based services. In *International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp)*. ACM, 545–556.

[38] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *International Conference on Artificial Intelligence (AAAI)*.

[39] Rose Yu, Yaguang Li, Ugur Demiryurek, Cyrus Shahabi, and Yan Liu. 2017. Deep learning: a generic approach for extreme condition traffic forecasting. In *SIAM International Conference on Data Mining (SDM)*. SIAM, 777–785.

[40] Chao Zhang, Liyuan Liu, Dongming Lei, and etc. 2017. TrioVecEvent: embedding-based online local event detection in geo-tagged tweet streams. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 595–604.

[41] Chao Zhang, Keyang Zhang, Quan Yuan, and etc. 2017. Regions, periods, activities: uncovering urban dynamics via cross-modal representation learning. In *International World Wide Web Conference (WWW)*. ACM, 361–370.

[42] Fuzheng Zhang, Nicholas Jing Yuan, David Wilkie, Yu Zheng, and Xing Xie. 2015. Sensing the pulse of urban refueling behavior: A perspective from taxi mobility. *Transactions on Intelligent Systems and Technology (TIST)* 6, 3 (2015), 37.

[43] Junbo Zhang, Yu Zheng, and Dengkang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *International Conference on Artificial Intelligence (AAAI)*.

[44] Xuchao Zhang, Liang Zhao, Arnold P Boedihardjo, Chang-Tien Lu, and Naren Ramakrishnan. 2017. Spatiotemporal event forecasting from incomplete hyper-local price data. In *International Conference on Information and Knowledge Management*

(CIKM). ACM, 507–516.

[45] Liang Zhao, Junxiang Wang, and Xiaojie Guo. 2018. Distant-supervision of heterogeneous multitask learning for social event forecasting with multilingual indicators. International Conference on Artificial Intelligence (AAAI).

[46] Guanjie Zheng, Susan Brantely, Thomas Lauvaus, and Li Zhenhui. 2017. Contextual spatial outlier detection with metric learning. In *International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 2161–2170.

[47] Yu Zheng, Huichu Zhang, and Yong Yu. 2015. Detecting collective anomalies from multiple spatio-temporal datasets across different domains. In *International Conference on Advances in Geographic Information Systems (SIGSPATIAL)*. ACM.

[48] Xian Zhou, Yanyan Shen, Yanmin Zhu, and Linpeng Huang. 2018. Predicting multi-step citywide passenger demands using attention-based neural networks. In *International Conference on Web Search and Data Mining (WSDM)*. ACM, 736–744.

Who Watches the Watchmen: Exploring Complaints on the Web

Damilola Ibosiola
Queen Mary University of London
d.i.ibosiola@qmul.ac.uk

Ignacio Castro
Queen Mary University of London
i.castro@qmul.ac.uk

Gianluca Stringhini
Boston University
gian@bu.edu

Steve Uhlig
Queen Mary University of London
steve.uhlig@qmul.ac.uk

Gareth Tyson
Queen Mary University of London
g.tyson@qmul.ac.uk

ABSTRACT

Under increasing scrutiny, many web companies now offer bespoke mechanisms allowing any third party to file complaints (*e.g.,* requesting the de-listing of a URL from a search engine). While this self-regulation might be a valuable web governance tool, it places huge responsibility within the hands of these organisations that demands close examination. We present the first large-scale study of web complaints (over 1 billion URLs). We find a range of complainants, largely focused on copyright enforcement. Whereas the majority of organisations are occasional users of the complaint system, we find a number of bulk senders specialised in targeting specific types of domain. We identify a series of trends and patterns amongst both the domains and complainants. By inspecting the availability of the domains, we also observe that a sizeable portion go offline shortly after complaints are generated. This paper sheds critical light on *how* complaints are issued, *who* they pertain to and *which* domains go offline after complaints are issued.

CCS CONCEPTS

• **Information systems** → **World Wide Web**; *Web complaints*; Web search engines; Web indexing; Infringing content distribution; • **General and reference** → Measurement.

KEYWORDS

Web Measurement; Web Complaints; Copyright Infringement; Takedown notices

ACM Reference Format:
Damilola Ibosiola, Ignacio Castro, Gianluca Stringhini, Steve Uhlig, and Gareth Tyson. 2019. Who Watches the Watchmen: Exploring Complaints on the Web. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313438

1 INTRODUCTION

The web has proven a powerful platform for the large-scale distribution of content. Notoriously difficult to regulate, individual web organisations have been frequently left to decide how to best handle issues related to legal, regulatory and even moral matters, *e.g.,* moderation of online discourse, removal of copyright infringing content, mitigation of online harassment. This has generated significant societal attention, with major companies like Google and Twitter coming under increasing public scrutiny [9, 24]. Consequently, large web organisations have begun to implement bespoke mechanisms to allow third parties to register complaints. For example, Google's complaint system allows anybody to issue notices requesting the removal of specific results from their search listings, whereas Twitter enables users to report posts they believe to be infringing policy. Although a valuable tool in the wider landscape of web governance, this places considerable responsibility within the hands of these organisations, who must decide which complaints are legitimate and how they should be dealt with (often referred to as self regulation [30]). Yet, to date, we have little evidence regarding how these complaint procedures are handled, how successful they are, or who they are targeted against and by whom.

We argue that the scale of these complaints and their impact on the wider society's perception of the web, warrant detailed investigation. There are multiple interesting questions, such as who generates complaints? Whom do these complaints pertain to? What are the characteristics of domains that receive complaints? Does content remain online after complaints? With these questions in mind, this paper presents the *first large-scale study of web complaints*. To this end, we have gathered complaints from hundreds of transparency reports made available by organisations including Google, Vimeo, Bing and Twitter, and reflected in the Lumen database (§2). These reports expose detailed information about web complaints covering over 1 billion URLs ranging from copyright infringement to governmental notices.

We start our analysis by characterising the nature and scale of organisations that generate notices (complainants). Despite the presence of numerous complaint categories, copyright notices largely dominate, with 98.6% of all complaints (§3.1). A critical minority of organisations play a remarkably prominent role in this, with the top 10 alone contributing 41% of all notices sent (§3.2). Our analysis reveals that the majority of *notices* are generated by large content-based organisations (*e.g.,* NBC, Fox). Despite this, we find that the majority of notice senders are occasional users of the complaint system: 94% of complainants issue fewer than 100 notices. These different groups tend to rely on different types of complaints. For example, large copyright enforcers (*e.g.,* Rivendell) generate millions of copyright notices, whereas governmental agencies (*e.g.,* Roskomnadzor) issue a far smaller set of targeted court and governmental

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313438

notices. This leads us to focus on the categories of websites that these different types of notices pertain to (§3.3). We find that notices are heavily biased towards certain types of website. For example, websites categorised as 'File Sharing' and 'Elevated Exposure' are hugely over-represented amongst our complaint dataset. Similarly, individual notice senders tend to target specific topics, *e.g.,* 60% of complaints by *Cam Model Protection* target adult websites, whereas 55% by *NBCUniversal* target Entertainment, shedding light on the priorities of organisations utilising the complaint services.

This leads us to investigate if equivalent dynamism exists on the part of the reported websites themselves (§4). We find that a small (and constantly evolving) set of websites dominate the ranking of the most reported, with a few domains that remain prominent throughout our entire measurement period: the top 1% of domains accumulate 63% of complaints (§4.1). These include many obscure websites, which are unlikely to be widely known, *e.g.,* 22 out of the top 30 most frequently reported websites are not even in the Alexa Top 100K, and the correlation between the popularity of domains in terms of notices and Alexa rank is just 0.13. Deeper analysis reveals that these trends are dictated by the activity of a small set of *extremely* aggressive notice senders (§4.3), whose 'bursty' behaviour creates high levels of instability.

We finally inspect the availability of the webpages that are complained about, with the conjecture that websites receiving many complaints may be more likely to go offline (§4.4). We confirm that many are highly ephemeral: 22% of all domain names soon get taken offline (NXDOMAIN), whereas our HTTP liveness checks show that a further 19% of all resources fail to return 200 OK responses within 1 week of a complaint. We correlate these with a number of factors to find that the 'success' rate of complainants differs dramatically, with the most successful (Rivendell) seeing 55% of its complaints acted upon in contrast to others complainants where the figure is below 1%.

2 METHODOLOGY & DATASET

This section presents our data collection methodology. This has two goals: (*i*) To provide vantage into a broad set of web complaints, covering enough domains to provide meaningful insight; and (*ii*) To annotate these complaints with sufficient metadata to shed light on domain activities and topics.

2.1 Website Complaints

First, we detail our methodology for gathering complaints issued about websites. Naturally, it is impossible to get complaints issued to *all* websites, because the vast majority do not make this information available. Hence, we focus on sites that expose transparency reports, *e.g.,* Bing, Twitter, Google, Vimeo. These reports list complaints received by the organisations, including relevant metadata. To gather these, we utilise Lumen[1] — a database run by the Berkman Klein Center for Internet & Society. It aggregates and publishes transparency report data pertaining to complaints issued towards 223 organisations. The exact purpose of each complaint differs, *e.g.,* a complaint to Bing will normally request the removal of search results, whilst complaints to Vimeo will concern the removal of videos. However, each notice (*i.e.,* complaint) includes the following standard fields:

- *Notice Type:* the category of notice which has been reported. For example, a Digital Millennium Copyright Act (DMCA), trademark or data protection *etc.*
- *Notice Sender:* or complainant, is the organisation who submitted the notice.
- *Notice Recipient:* the web publisher or service provider where the infringing notice is sent to.
- *Notice Principal:* in cases of a copyright-related notice, this is the person or organisation that holds the copyright on the content reported.
- *Infringing URL(s):* the list of URL(s) that the notice sender is requesting to be dealt with. Note that this is not necessarily a set of URLs owned by the recipient, *e.g.,* Bing may receive complaints requesting the removal of a third-party URL from its search results.

We collected all complaints from the Lumen database between 01/01/2017 and 31/12/2017 using their API. In total, we extract 1,054,248,823 URL complaints from 38,523 notice senders.

2.2 Website Metadata

Once we extract the reported URLs, we compile further metadata. This section explains the metadata collected.

Website Categorisation. We classify each domain using the VirusTotal API.[2] This API has been used in a wide set of research, and is known to provide good accuracy [14, 16, 31]. The API provides a classification for each domain in our dataset, *e.g.,* games, education, file sharing, blogs *etc.* We later use this to understand the types of complaints generated. Due to the usage limitations of the API, we only categorise the top 240K domains with the most reported URLs. Note that 22% of these could not be categorised by VirusTotal. We further annotate each domain with its Alexa ranking to gain insight into its global popularity.

DNS Probes. We utilise the Domain Name Service (DNS) to map the domains to their respective DNS records on 29/07/2018. We performed queries (IPv4), which yield 849,023 responses and 206,863 IP addresses. We use this data to check if the domain name is still live.

Webpage Probes. For each URL, we download its HTML and parse it to extract all embedded domains. This allows us to identify mirrors of websites hosted on multiple domains/URLs, by comparing the HTML contents. Tests are performed from a university campus, where we have confirmed no web filtering is performed. In total, we collect the HTML structure for 770,737 webpages.

Liveness Checks. We also perform periodic checks on the domains and URLs to verify if the websites are still active (*i.e.,* returning an HTTP 200 status code). This allows us to explore the potential efficacy of organisations seeking to remove content. Due to the sheer scale of the complaints (>1 billion URLs), we only perform checks for 2M URLs complained about between 14/07/2018 to 17/07/2018. Upon recording a complaint from Lumen, we added its URL to a queue and began weekly checks that executed between 18/07/2018

[1] https://lumendatabase.org/

[2] https://www.virustotal.com/

to 14/08/2018. We record the HTML response and HTTP status code, alongside whether or not the TCP handshake timed out. Each week, we exclude URLs that have already been deleted.

2.3 Limitations & Ethical Considerations

We emphasise that Lumen only provides vantage into 223 complaint recipients, obviously a small fraction of the world's online organisations. These consist primarily of: *Google*–79%, *Bing*–15.6%, *Twitter*–3.8%, *Periscope*–0.8% and *Vimeo*–0.4%. We do not assert that our findings generalise beyond these organisations, although the scale of these five companies indicates that the insights are highly valuable nevertheless. We should also highlight that our research covers websites that may participate in illegal activity (*e.g.*, copyright infringement). We restrict this collection to downloading the webpage HTML and checking server liveness, ensuring that *no* measurements involve participating in a websites activities, *e.g.*, registering user accounts or downloading content. Note that this may still result in advertisement revenue being generated by the website. Unfortunately, this was impossible to avoid considering the nature of our measurements. That said, we limit ourselves to accessing URLs a small number of times (< 10), minimising any potential revenue.

3 COMPLAINTS & NOTICE SENDERS

In this section, we investigate the senders and receivers of complaints. Specifically, we are interested in exploring *who* sends complaints and *what* they pertain to.

3.1 What Types of Complaints Exist?

We identify complaints from 38,523 unique senders, covering 1.05 Billion URLs, which are hosted across 1,090,173 domains. Individual complaints tend to contain multiple URLs, with an average of 560 URLs and 31 domains per notice. In terms of the *types* of notices, there is a remarkable skew towards DMCA complaints. This is partly driven by the prominence of search engine recipients within our data. Table 1 presents a breakdown of the types of complaints across the entire dataset. We find that DMCA notices make up 98.6% (1.05B URLs) of the dataset and a similar share of domains (97.8%). DMCA notices are a US-enforceable complaint which covers takedown notices for (allegedly) copyright infringing content. The senders largely appear to be third party organisations who act on behalf of the actual copyright owners: just 9% of notice senders listed are also the principal. Measured by frequency, DMCA notices are followed by Defamation (52K), Court Orders (29K) and Government Requests (2.7K), covering nearly a third of the recipients (31%). We also observe a number of less popular complaint types, such as Law Enforcement Requests, Data Protection and Trademark infringements. Whereas these make up less than 0.001% of the dataset, they cover more than 20% of the recipients.

3.2 Who Are the Notice Senders?

The previous section suggests that notice senders are more diverse than simply the owners of copyright material, and that different reporting practices coexist. To shed light on these aspects, we next inspect the entities behind the notices submitted. We find that the distribution of notices is *highly* skewed towards a few extremely

Table 1: Dataset summary with the percentages of notice types, and the corresponding share of URLs, senders, principals and domains.

Notice Type	% of notices	% of URLs	% of senders	% of principals	% of domains
DMCA	98.6	99.99	94.46	99.87	97.78
Defamation	0.95	<0.01	0.15	<0.01	1.50
Court Order	0.19	<0.01	4.87	0.07	0.41
Government Request	0.15	<0.01	0.15	0.02	0.13
Private Information	<0.01	<0.01	0.071	-	0.003
Data Protection	<0.01	<0.01	-	-	<0.001
Law Enforcement Request	<0.01	<0.01	0.02	0.03	<0.001
Trademark	<0.01	<0.01	0.02	-	<0.001
Other	0.08	<0.01	0.26	<0.01	0.16
Total	1,885,267	1,054,248,823	38,523	20,686	1,090,173

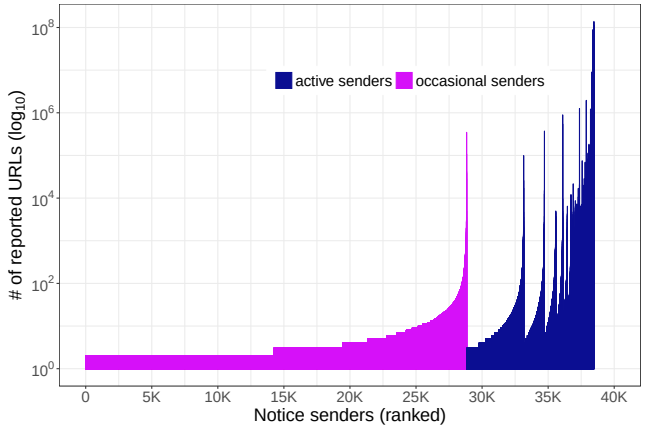


Figure 1: Number of reported URLs per complainant. The X-axis present each complainant, sorted by the two metrics: (*i*) number of days a complainant reports on; and (*ii*) notices generated by each complainant.

active senders. The top 10% of notice senders report over 1 billion URLs, in stark contrast to just 550K by the bottom 90%.

Figure 1 presents the number of reported URLs from each notice sender. The X-axis is sorted by two metrics: (*i*) the number of days that a sender generates notices on; and (*ii*) the number of notices sent (note that Y-axis is in log scale). The distribution is highly unbalanced, with a large majority of notice senders (94%) issuing fewer than 100 complaints. The figure reveals two broad categories of complainants: (*i*) *active*, who send complaints on multiple days and (*ii*) *occasional*, only generating notices on a single day. Whilst the *active* group represents just 25% of all complainants, it is responsible for almost all notices (99.92%). In contrast, the *occasional* senders, consisting of the remaining 75%, collectively contribute just 0.08% of the total number of complaints. It can also be observed that the curve in Figure 1 is not monotonic. This is because the number of notices issued each day can vary. Whereas occasional senders, by definition, *only* issue complaints on a single date, some send multiple notices. Although the daily average is just 1.4 notices, there are some occasional senders who send a large number of complaints in a single burst. For example, Idreto (an occasional sender) generated 350K notices on a single day. As a result, even

Table 2: Top 10 complainants (by # of reported URLs).

Notice Sender	% of reported URLs	% of notice	% of reported domains	# of reported days
Rivendell	13.17	1.65	4.97	357
Aiplex	9.76	1.88	1.30	364
BPI	8.60	2.52	2.72	355
Apdif Mexico	8.52	0.55	0.16	208
Mg Premium	7.77	0.47	0.49	341
Apdif Brasil	7.56	0.52	0.39	244
Remove Your Media	7.28	0.92	2.45	346
Mark Monitor	5.29	1.13	5.12	365
Fox Group Legal	4.36	0.14	2.26	355
Protek Media, S.C.	4.20	0.31	0.67	365
Total	806,358,505	188,158	224,076	

occasional senders can have a significant impact on the overall ranking of reported domains.

We now inspect the types of complaints generated by these two groups of notice senders. Table 2 presents the top 10 complainants. We find that *active* senders are dominated by copyright enforcement and trade organisations *e.g.,* the British Phonographic Industry (BPI), Apdif Brasil, Apdif Mexico, *etc.* These organisations represent hundreds of music recording companies in their respective jurisdictions. Also within this group are copyright protection agencies such as Muso, Aiplex, Mark Monitor and Entura who specialise in tracking pirated content. These companies aggregate complaints from many different copyright holders, and act as enforcers on their behalf. This partly explains their broad coverage and ability to produce large volumes of complaints. In contrast, *occasional* senders are predominantly made up of small organisations and private entities. We see that the categories of *Private Information* and *Trademark* are particularly dominated by *occasional* notice senders, making up 94% and 70% respectively. Conversely, *active* senders hold sway among other notice types, *i.e.,* DMCA. These striking differences between the categories of sender highly differing strategies based on the nature of the complaints, with the ability of a small hub of complainants to dominate the wider system.

3.3 What Topics Do Complaints Target?

To better understand the main drivers behind the complaints, we investigate which categories of websites notices senders complain about. Figure 2 presents the percentage breakdown of complaints, domains and complainants based on the reported domain category. Although it is clear that these classifications contain noise, we believe they offer useful insight into wider activity. To better understand how this relates to the general web ecosystem, we also depict the distribution of the Alexa Top 100K websites.

There is a remarkable divergence between our dataset and Alexa, highlighting a clear bias towards complaints for certain types of websites. The largest fraction of domains are categorised as business (for both Alexa and Lumen), which includes websites such as `4shared.com`, `mangapark.me` and `gorillavid.in`.[3] The most over-represented categories in the Lumen dataset are Elevated Exposure[4] (8% *vs.* 0.3% for Alexa), Entertainment (13% *vs.* 3%), File

[3]We note that this category covers a number of business facing websites, including those engaged in Hollywood copyright theft.
[4]This category refers to malicious websites that camouflage their true nature

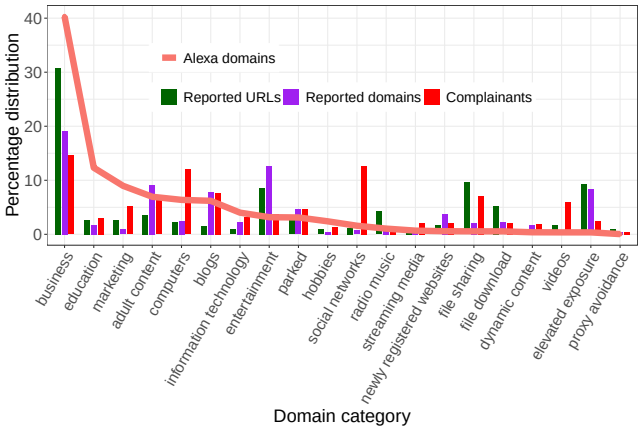


Figure 2: Percentage of reported URLs, domains and complainants for the 20 most reported domain categories *vs.* the share of Alexa Top 100K per category.

Download (2% *vs.* 0.6%), File Sharing (2% *vs.* 0.6%), Blogs (8% *vs.* 6%), and Adult Content (9% *vs.* 7%). While File downloading and sharing are expected, the other ones are less intuitive. In contrast, categories that are under-represented include Education (2% for Lumen *vs.* 12% for Alexa) and Marketing (1% *vs.* 9%).

These trends suggest that notice senders tend to focus on individual categories. To explore this, Figure 3 presents the fraction of complaints that target each of these top 20 categories for the top 30 complainants. In-line with our conjecture, several complainants exhibit significant bias towards a single category. For example, 42% of MG Premium's complaints target adult websites, 45% of NBCUniversal's focus on Entertainment, and 34% of Rico's is Business, demonstrating the prevalence of a high degree of specialisation. Considering the practice of using web crawling to collect links [29], this specialisation makes sense as it allows individual organisations to streamline their activities (based on which sites they have crawlers for).

Briefly, we also note that these categories tend to attract different *types* of notices too. For example, we find that the majority of Data Protection (60%) notices are logged against adult domains. This aligns with the well known high litigiousness of the adult content industry [21]. In contrast, 10% of complaints catagorised as Defamation pertain to news. The remaining notice types (DMCA, Government, and Private Information) are typically made about business domains, representing 31%, 33% and 60% for each category respectively; with the exception of Court Order notices where most complaints are regarding shopping domains (27%). These clear trends confirm that the majority of notice senders are quite focused in their activities, with clear specialisation.

4 WEBSITES & WEBPAGES

Whereas the previous section explored the complaints and complainants, we next inspect the websites (domains) and webpages (URLs) which the complaints pertain to. Specifically, we are interested in understanding which websites gain most attention, and their availability.

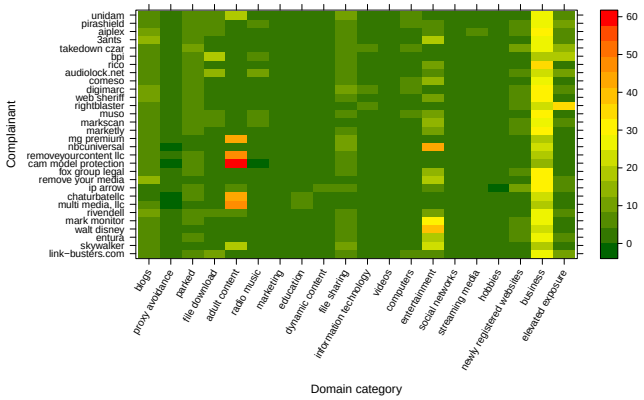


Figure 3: Distribution of (top 20) domain categories (in %) reported by (top 30) complainants.

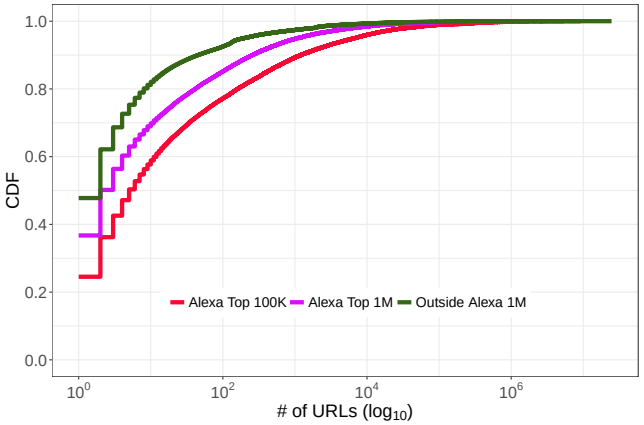


Figure 4: CDF of the number of complaints per domain. Domains are classified by their Alexa Rank.

4.1 How 'Hot' Are Websites?

We start by inspecting the number of complaints generated about each website. Figure 4 presents a CDF showing the distribution of notices per domain. We separate the data depending on the domain popularity, based on its position in the Alexa ranking. Again, we observe a noticeable skew: it is clear that there are a small number of 'hot' websites that gain the most attention from notice senders. In particular, the most reported domains (the top 1% by the number of reported URLs), receive 63% of all complaints. To provide insight into the characteristics of these domains, Table 3 summarises the Top 10 in terms of the number of complaints. A range of uncommon websites are seen within this list. Although, overall, 60% of notices relate to websites in the Alexa Top 100K, only 4 of the most reported 10 domains, and 8 out of the top 30 are within this Alexa rank (we find a Pearson correlation of just 0.13 between these two rankings). The reasons behind these relatively obscure sites gaining 'notoriety' are quite diverse, but they do share one common attribute: their position in the ranking is typically driven by a single complainant that repeatedly targets them. In fact, 9 out of the top 10 domains receive at least 50% of their complaints from a single organisation. This is a general pattern across all domains: we find that 82% of them receive at least half of their complaints from a single complaint. To visualise this dominance by a few complainants in each domain, we depict in Figure 5 the percentage of URLs reported by the top 3 complainants of each domain (in green). The rest of complainants (in red) tend to contribute little to nothing. The vast majority of domains receive nearly all complaints from just a tiny set of senders. For example, `mp3toys.xyz` receives 99.9% of complaints from a single party (Apdif Brasil). Upon closer inspection, it is clear that this organisation uses an algorithm to 'guess' potential infringing URLs based on song titles. Although we envisage that these URLs are checked for liveness before complaints are generated, we note that `mp3toys.xyz` dynamically generates 200 OK HTTP responses for *any* URL requested, likely disrupting any liveness checks. These type of bulk sending activities do not appear to be rare occurrences. For example, the same organisation reported over 17M URLs for file hosing domain `4shared.com`, despite the website only hosting 2M pages [25]. As well as confirming that these highly active senders are heavily automated, it also suggests that rigorous procedures are not always followed.

4.2 Are Domains Unique?

The above is based on unique domain counts, however, we also posit that some of these domains may actually host the same content, or even resolve to the same IP address. To test this, we turn to our DNS and webpage probes, which downloaded the HTML from all domains. We extract their `<title>` and all metadata tags. In cases, where two domains' tags match, we assume they host the same page. We term these *replicas*. The vast majority of domains host different content. Fewer than 0.01% of domains have any replicas. From those that do, 73% refer to the same IP address, indicating that the web server operator has simply created multiple domain names. Just 2% of these have matching second-level domains (with a different TLD), whereas the remainder actually are entirely distinct. We conjecture that this may be an evasion tactic to avoid DNS-based blocking schemes. We also observe certain outliers; in the most extreme case, we find that `1fichier.com` uses 3,838 different domain names, which map to 78 IP addresses. This is a file sharing service well known for hosting illegal content. Another key driving force in the case of these extreme examples is the presence on numerous unblocker websites in our dataset. These websites essentially operate as proxies, generating third level domains for any website requested. For example, `s-s.www.cats.com.prx2.unblocksites.co` provides access to `www.cats.com` via `unblocksites.co`. Table 4 presents the top 10 (in terms of reported URLs) of these unblocking services. Remarkably, `unblocksites.co` actually constitutes 10% of all reported URLs. In other words, we find that many complainants target these unblocker sites, by replicating their complaints for both the origin domain and the unblocked version. Understanding an exploring these is a key area of our future work.

4.3 How Stable are Domain Rankings?

The previous sections indicate that complaints about a domain are often dominated by a single sender. We conjecture that this

Table 3: Top 10 domains with most complaints and the number of TLDs associated with each domain, times it appeared in the top 10 (in terms of complaints) per month, domain category, major complainant (with the share of complaints), number of days receiving complaints and its Alexa ranking position.

Domain	# of TLDs	# of times in top 10	Domain category	# of complainants	Major complainant (s) (% of complaints)	# of days reported	Alexa Rank
mp3toys.xyz	10	3	elevated exposure	10	Apdif Brasil (99.9%)	109	-
4shared.com	15	4	filesharing	40	Apdif Brasil (99.8%)	246	-
googlevideo.com	1	7	entertainment	84	Comeso (78.2%), Remove Your Media (17.3%)	239	-
mangapark.me	3	4	business	18	Remove Your Media (99.1%)	28	3,901
gorillavid.in	3	3	business	16	Fox Group Legal (55%), Mark Monitor (25.2%), Vobile (17.1%)	365	6,778
tvad.me	1	2	business	14	Fox Group Legal (59.9%), Mark Monitor (39.9%)	94	45,766
israbox.vip	101	5	media file download	19	Rivendell (99.6%)	46	-
uploaded.net	3	2	filesharing	216	Rivendell (31.7%), Skywalker(18.1%) Mark Monitor (14.9%), Fox Group Legal (12.3%)	365	662
genteflowmp3.uno	11	2	media file download	6	Apdif Mexico (99.6%)	149	-
deep-warez.org	1	2	radiomusic	43	Rivendell (98%)	345	183,305

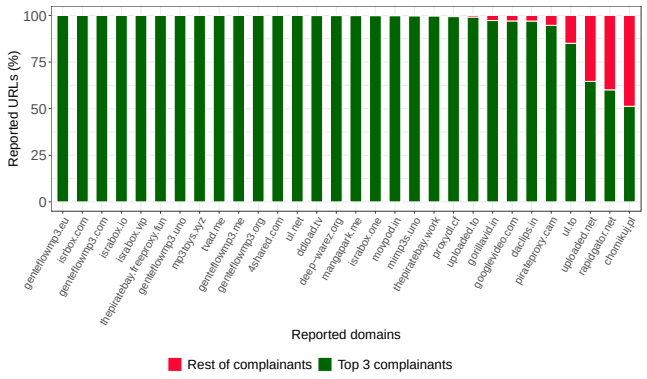


Figure 5: Percentage of reported URLs by the top 3 complainants for each domain (on X-axis) *vs.* the rest of complainants.

Table 4: Top 10 unblocking services present in our dataset.

Unblocking Site	% of Reported URLs	% of Domains	Unblocking Site Category	Alexa Rank
unblocksites.co	10	0.13	uncategorised	12,252
freeproxy.fun	2.4	0.02	webproxy	26,517
unblocked.lol	2	0.04	proxy avoidance	8,552
unblockall.xyz	0.9	0.02	proxy avoidance	820,291
proxydude.xyz	0.6	0.007	elevated exposure	-
immunicity.gold	0.5	0.03	proxy avoidance	-
unblockall.org	0.5	0.01	business	4,947
unblocked.cam	0.5	0.02	proxy avoidance	
unblocker.cc	0.4	0.009	proxy avoidance	15,535
unlockproj.club	0.4	0.02	uncategorised	

dominance may result in significant temporal instability. This can be represented as a ranked list, capturing the domains most frequently reported on each day. To explore this, Figure 6 presents the count of domains that are reported on x days. 81% of domains are complained about on fewer than 5 days, with only 0.2% being complained about on more than 300 days. This suggests a high degree of instability, with the make-up of complaints changing on a daily basis. In line with our previous findings, the majority of these infrequently complained about domains are outside the Alexa Top 1M, whereas 87% of domains which are complained about on over 300 days are within the top 1M. In Figure 6 this can be seen as an upturn on the right-hand side of the graph.

To explore this trend, We next calculate the statistical variance of each domain's daily complaint count to understand how the number of daily complaints change. Figure 7 presents a CDF showing the per domain variance over the entire dataset (for domains with complaints on multiple days). About 15% of domains have a variance of zero because they receive the same number of complaints each day they are reported; 94% of these are reported under six times. However, many remaining domains exhibit significant daily variance: 7% of domains have a variance greater than 1K and, remarkably, 0.9% of domains even have a daily variance greater than 1M. This means that the number of complaints to a domain varies heavily on a day-to-day basis.

Closer inspection reveals that this is driven by *extremely* aggressive complainants who periodically inject large sets of (sometimes repeat) notices. For example, for domains with variance greater than 1M, we find that 92% receive an average of 116 duplicates from the same sender. To better highlight this aggressive activity, Figure 8 presents timeseries measuring the number of daily complaints for several example domains. We select the top two domains with the largest variance in each Alexa ranking category (top 100K,

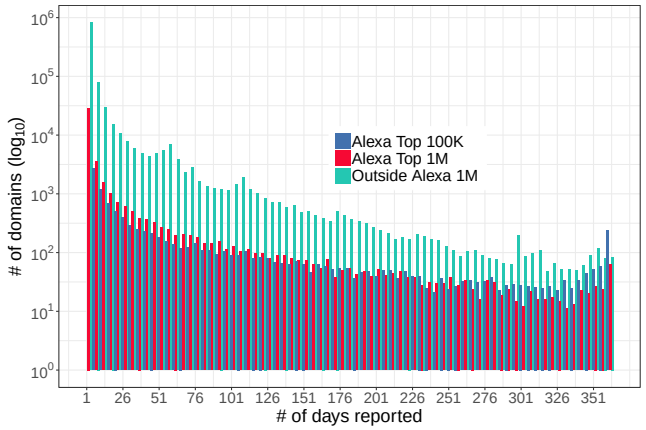


Figure 6: Number of days receiving complaints for each domain in a given Alexa rank.

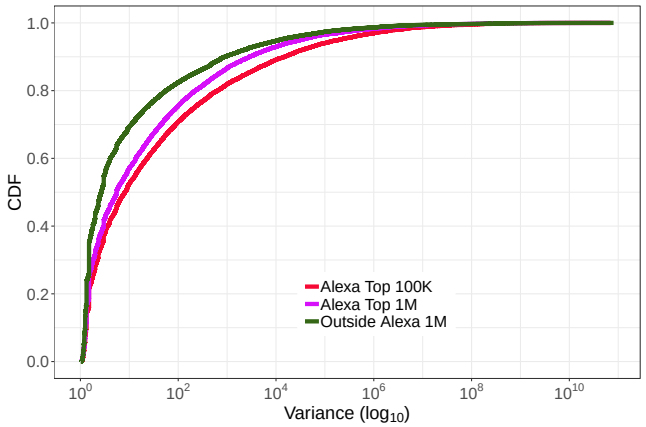


Figure 7: CDF of the variance of the number of complaints per day for each domain in a given Alexa rank.

top 1M, outside Alexa). Significant instability exists across each of these domains, with labels in the figure highlighting the individual sender causing each spike. For instance, the top ranked domain, `mp3toys.xyz`, receives an average of 500K daily complaints from Apdif Brasil between Jan & Feb, yet this collapses to below 6 per day in March. Similarly, `mp3taringa.net` receives a significant number of complaints between 03/Jan – 05/Jan by Apdif Mexico, but following this the numbers collapse to almost zero. We posit that these senders are not always careful in the complaints they generate but, rather, send bulk notices, leaving the recipient to make sense of the content.

4.4 Are Reported Websites Alive?

We finally inspect the availability of reported domains and webpage resources. As it is impossible to draw causal links between a notice being issued and the removal of content, we limit our analysis to inspecting the availability of URLs, rather than inferring the reason for their (un)availability.

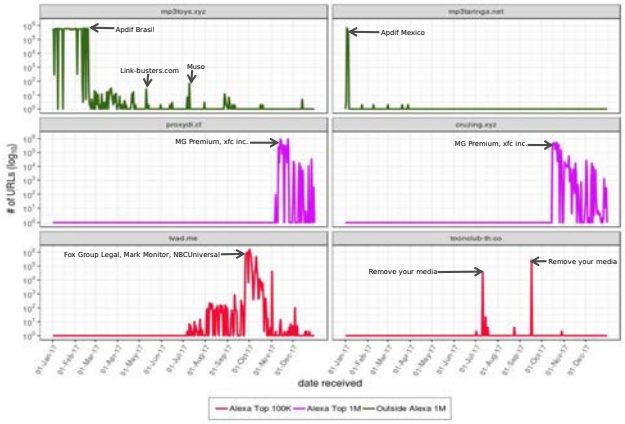


Figure 8: Timeseries with complaints to selected domains and complainants causing complaints' bursts.

Understanding Domain Liveness. We first check the liveness of each domain's DNS record using our DNS dataset. This reveals that 22% of reported domains return an NXDOMAIN response.[5] These domains account for over 183M (17%) of infringing URLs, with just 3% (1,426) of them belonging to domains that rank within Alexa's Top 1M. As different domain registrars may have differing policies regarding the removal of records, we next group domains by their TLD, and check the likelihood of domains being taken offline; Figure 9 presents the results. We plot both the number of websites (domains) that are unavailable, as well as the number of specific URLs that become unavailable (because the domain is offline). For context, we plot the density of Alexa top 1M domains that have each TLD. The majority of TLDs with a high percentage of NXDOMAIN responses do not frequently occur in the top Alexa rankings. Instead, we see that the majority of domains are from the recent wave of new generic TLDs [23]. The most extreme is `.lol`, where 98% of domains are NX; it is noteworthy that this TLD is operated by Uniregistry, which has been accused of predominantly hosting spam [1]. These trends indicate that the usage and behaviours across these TLDs are quite different, with some far more likely to contain unavailable domains.

Understanding URL Availability. Next, we turn to our periodic HTTP liveness checks to see if resources (URLs) are still alive (for those domains that do not return NXDOMAIN). We see that the number of 200 (OK) responses decline slowly but steadily over the 4 week period that we monitored. After week 4, 22% of URLs are inactive (*i.e.,* non-200). This trend, however, is relatively shallow with the majority of URLs (19%) becoming inactive in the first week after the notice has been observed in Lumen.[6] We also note that the statuses returned evolve across the four weeks. In week 1, the number of HTTP 4XX responses is 168K, yet in week 2 we only observe a further 12K webpages responding with 4XX. Instead, the

[5]This is returned when a domain name does not exist on the authoritative name server any longer.
[6]Note that it is also possible that the URL was not live when the complaint was generated. Unfortunately, we cannot check this.

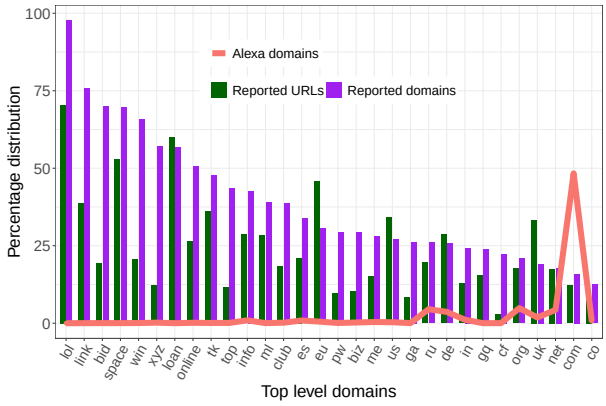


Figure 9: Unavailable domains per TLD and URLs thus becoming inaccessible (as a % from the total) *vs.* the share of Alexa domains for each TLD.

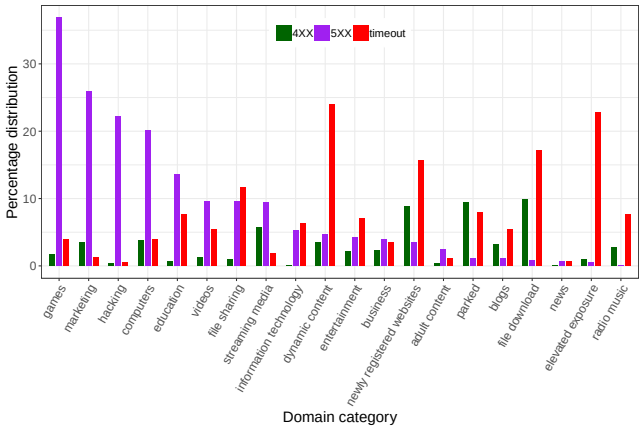


Figure 10: URLs going offline (*i.e.,* non-$2XX$ response) for the top-20 most reported domain categories (in %).

number of HTTP 5XX and TCP timeouts increase, suggesting that these websites go through several stages that start with removing of content (therefore returning a 4XX) before total shutdown of the web server. The latter makes sense, as it may be unnecessary to continue running a server if most content has been removed.

Understanding Category Availability. We continue our analysis by inspecting which category of URLs are most likely to go offline. Figure 10 breaks down all URLs into their categories and presents the share within each group that reports non-200 HTTP responses after complaints are generated. Certain categories are significantly more likely to return non-200 responses. For example, 10% of URLs classified as File Download return a 404 response; similar traits are also seen across Parked (9%) and Newly Registered Website (9%). In contrast, 24% of URLs classified as *Dynamic Content* (*i.e.,* websites that generate different material for each visit) return a TCP connection timeout, *i.e.,* the web server is no longer online. Other examples of categories with high numbers of URLs that timeout include *Elevated Exposure* (22%), and *File Download* (17%). The category which contains fewest unavailable sites is *News*, where only 1.34% of URLs become non-200. This indicates that the robustness of sites differs substantially across categories. That said, it is reasonable that domains more clearly engaging in suspicious activity are most likely to become unavailable.

Understanding Complainant Success An obvious question is which complainants are most likely to see their reported URLs deleted. To measure this, we calculate the percentage of reported URLs from each notice sender that later sees the URL resource returning a non-200 response. Figure 11 presents the weekly percentage of complaints that return a non-200 response after each weekly liveness check for the top 10 senders (based on the complainant with most not 2XX responses). The websites targeted by these different notice senders have very different availability properties. Complaints from rights enforcement organisations (*e.g.,* Rivendell, MarkScan, AudioLock.net) appear more effective compared to trade organisations (*e.g.,* British Phonographic Organisation).

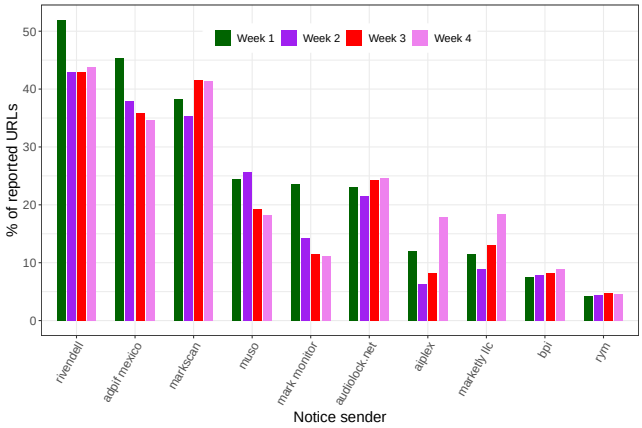


Figure 11: Weekly share (non-cumulative) of URLs with a non $2XX$ response for the top 10 complainants.

For example, about 53% of complaints in the first week of submission from Rivendell return non 2XX response code whilst just 8% from British Phonographic Organisation return same. These results confirms that the efficacy of these different organisations differs greatly, and that the websites they pursue have extremely different characteristics in terms of resilience to takedown.

5 RELATED WORK

There are two main areas of related work: (*i*) studies that rely on Lumen for exploring web complaints; and (*ii*) studies that have explored illegal online activities more generally.

Web Complaint Studies. There have been numerous studies into web complaints, however, these largely appear in legal journals, and rarely contain data-driven analysis. Those that do are mostly restricted to manual inspection of small sets of notices. For example, Heins *et al.* [10] examined 320 takedown notices to determine if the takedown process is fairly used by reporting organisations. They discovered that 20% of notice senders send weak copyright claims,

an assertion our results agree with. Urban *et al.* [28] examined nearly 900 notices in an attempt to discover the primary reporting organisations that file them. They saw that business entities and corporations were the main users of takedown notices. Our work differs significantly from these two studies. Whereas they study a small sample of notices, we offer far broader vantage into web complaints, characterising the activities across notices regarding over 1 billion URLs. Closely related is the Right To Be Forgotten (RTBF), established in 2014 within the European Union. This allows individuals to request the de-listing of personal information from search engines, which is "inaccurate, inadequate, irrelevant or excessive". Bertram *et al.* [2] inspected the RTBF requests issued to Google. This work is complementary to our own, as we focus on different complaint mechanisms, *i.e.,* Lumen does not cover RTBF. This is evident from the contrast between types of URLs in [2] *vs.* our study, *e.g.,* 33% of RTBF URLs relate to social media, and 20% to news. This can be compared against §3.3, where we find a greater propensity towards Entertainment and Business URLs (driven by the prominence of copyright enforcement notices).

Illegal Web Activities. George *et al.* [8] examined the challenges that comes with the ease of sharing User Generated Content (UGC), highlighting the roles played by hosting providers and proposing stronger legislation to address the illegal sharing of UGC. Wong [33] suggested a more flexible legislation, whilst Sawyer [22] recommended that platforms that share UGC should develop solutions to mitigate against the sharing of infringing material. Clay and Lucas discussed how a UGC platform (YouTube) has been exploited for such purpose [3, 11]. Raman *et al.* also identified pirated content being shared via Facebook Live [19]. To prevent the sharing of infringing content on such platforms, Dutta *et al.* proposed a signature-based detection to mitigate against infringing material remaining accessible online [6].

Peer-to-Peer networks and file hosters are also a frequently used to disseminate illicit or illegal material. Despite several anti-piracy efforts through the injection of fake content on BitTorrent portals [4, 5, 7, 15] and the shutdown of file hosters services [18], about 90% of files shared using BitTorrent protocol are judged to be infringing [20, 32]. Furthermore, 80% of files shared through file hosters are also in the same category [17]. Ibosiola *et al.* measured the availability of illegal content on streaming cyberlockers [13]. They found that the majority of copyright infringing content is hosted on a small number of platforms. A large portion of complaints in our data also pertain to adult content. While there have been several studies on online adult content [26, 27], few focus on illegal adult content [12]. Our work is orthogonal as we primarily focus on the complaints that are related to these activities.

6 DISCUSSION AND FUTURE WORK

This paper has explored the nature of web complaints. With increasing scrutiny on illegal and illicit web activities, and the recent ability to streamline complaints against different stakeholders, this study offers a critical input into the wider ongoing debate about web governance and the use of so-called self-regulation [30].

Summary of Findings. We have found a large and complex ecosystem dominated by a small set of complainants. While there are a large number of organisations (38,523) that generate over 1 billion reported URLs, the top 10 complainants alone contribute over 41% of all notices. It therefore appears that these complaint procedures have become the dominion of a small group of large and very active organisations. Dominant players consists of a mix of influential copyright owners (*e.g.,* Fox) and third-parties specialised in pursuing copyright infringers (*e.g.,* Rivendell). Bursts of complaints are common with most of the complaints towards each domain originating from 2–3 complainants, driving the unusual instability we see in the rankings. Complainants are highly specialised in terms of the types of notices they generate and the domains they target. This leaves some domain categories (*e.g.,* File Sharing) regularly reported, and others rarely seen (*e.g.,* Education). Surprisingly, many of the most frequently reported domains are quite obscure, and fail to score highly in popularity rankings. Finally, we find that complaints do seem to matter. Many domain names are soon taken offline and 22% of the URLs are inaccessible within just 4 weeks of us observing the complaints. Hence, it is clear that we shed light on a highly dynamic environment from the perspective of domain operators too.

Societal and Legal Implications. Web governance and the (mis)use of web complaint mechanisms have important social and legal implications. Transparency is critical and, as a society, it is important to know how and why information is filtered. This is particularly the case as we have found that these mechanisms might not be always used wisely, *e.g.,* with some complainants generating hundreds of repeat notices, and seemingly auto-generated URLs (§ 4.3). We argue that this might overwhelm recipients, who will not necessarily have the resources to deal with these large numbers. As these highly centralised models of operation have the potential for misuse, understanding the activity of senders is therefore critical.

Our results further suggest that there is opportunity to improve and streamline the procedures. For notice recipients, the filtering of invalid complaints would no doubt be a valuable innovation. That said, we do not discount the veracity of many complaints, and therefore developing mechanisms to support this process from the perspective of (legitimate) notice senders would also be worthwhile. This, of course, should not be done at the expense of website operators, who should always be given paths to recourse. Developing techniques that automate the above three things is important. Arguably, Lumen and other similar transparency platforms, can play a powerful role in this process.

Future Work. There are a number of lines of future work. First, we hope to expand our access to more diverse datasets. Within the paper, we have not investigated the role that recipients might play within the nature of complaints observed. This is likely to open up new lines of interesting investigation. Our analysis has also revealed traits of a cat-and-mouse game, with complainants bulk sending notices, and websites replicating themselves across multiple domains and TLDs. Exploring the temporal attributes of this game will no doubt reveal a number of yet unseen behaviours. Last, we also wish to explore if search engines cease to index URLs that are complained about. Quantifying this forms a key strand of our future work.

REFERENCES

[1] 2016. Famous Four rubbish Spamhaus Worst TLD league. http://domainincite.com/20164-schilling-famous-four-rubbish-spamhaus-worst-tld-league.

[2] Theo Bertram, Elie Bursztein, Stephanie Caro, Hubert Chao, Rutledge Chin Feman, Peter Fleischer, Albin Gustafsson, Jess Hemerly, Chris Hibbert, Luca Invernizzi, et al. [n. d.]. Three years of the Right to be Forgotten. *Under Submission* ([n. d.]).

[3] Andrew Clay. 2011. Blocking, tracking, and monetizing: YouTube copyright control and the downfall parody. Institute of Network Cultures: Amsterdam, 219–233.

[4] Rubén Cuevas, Michal Kryczka, Angel Cuevas, Sebastian Kaune, Carmen Guerrero, and Reza Rejaie. 2013. Unveiling the incentives for content publishing in popular bittorrent portals. *IEEE/ACM Transactions on Networking* (2013).

[5] Ruben Cuevas, Michal Kryczka, Roberto González, Angel Cuevas, and Arturo Azcorra. 2014. TorrentGuard: Stopping scam and malware distribution in the BitTorrent ecosystem. *Computer Networks* (2014).

[6] Rabindranath Dutta and Kamal Chandrakant Patel. 2008. Detecting copyright violation via streamed extraction and signature analysis in a method, system and program.

[7] Reza Farahbakhsh, Angel Cuevas, Ruben Cuevas, Reza Rejaie, Michal Kryczka, Roberto Gonzalez, and Noel Crespi. 2013. Investigating the reaction of BitTorrent content publishers to antipiracy actions. *13th IEEE International Conference on Peer-to-Peer Computing, IEEE P2P 2013 - Proceedings* (2013), 1–10.

[8] Carlisle George and Jackie Scerri. 2007. Web 2.0 and User-Generated Content: legal challenges in the new frontier. *Journal of Information, Law and Technology* (2007).

[9] G Anthony Giannoumis. 2014. Regulating web content: the nexus of legislation and performance standards in the United Kingdom and Norway. *Behavioral sciences & the law* 32, 1 (2014), 52–75.

[10] Marjorie Heins, Waldman Michael, and Goldberg Deborah. 2005. Will Fair Use Survive? Free Expression in the Age of Copyright Control. *Brennan Center for Justice* (2005).

[11] Lucas Hilderbrand. 2007. YouTube: Where cultural memory and copyright converge. *FILM QUART* 61, 1 (2007), 48–57.

[12] Ryan Hurley, Swagatika Prusty, Hamed Soroush, Robert J Walls, Jeannie Albrecht, Emmanuel Cecchet, Brian Neil Levine, Marc Liberatore, Brian Lynn, and Janis Wolak. 2013. Measurement and analysis of child pornography trafficking on P2P networks. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 631–642.

[13] Damilola Ibosiola, Benjamin Steer, Alvaro Garcia-Recuero, Gianluca Stringhini, Steve Uhlig, and Gareth Tyson. 2018. Movie Pirates of the Caribbean: Exploring Illegal Streaming Cyberlockers. *Association for the Advancement of Artificial Intelligence(ICWSM)* (2018).

[14] Muhammad Ikram, Rahat Masood, Gareth Tyson, Mohamed Ali Kaafar, Noha Loizon, and Roya Ensafi. 2019. The Chain of Implicit Trust: An Analysis of the Web Third-party Resources Loading. *arXiv preprint arXiv:1901.07699* (2019).

[15] Sebastian Kaune, Ruben Cuevas Rumin, Gareth Tyson, Andreas Mauthe, Carmen Guerrero, and Ralf Steinmetz. 2010. Unraveling bittorrent's file unavailability: Measurements and analysis. In *Peer-to-Peer Computing (P2P), 2010 IEEE Tenth International Conference on*. IEEE, 1–9.

[16] Dae Wook Kim, Peiying Yan, and Junjie Zhang. 2015. Detecting fake anti-virus software distribution webpages. *Computers & Security* 49 (2015), 95–106.

[17] Tobias Lauinger, Kaan Onarlioglu, Abdelberi Chaabane, Engin Kirda, William Robertson, and Mohamed Ali Kaafar. 2013. Holiday pictures or blockbuster movies? Insights into copyright infringement in user uploads to one-click file hosters. *Lecture Notes in Computer Science* 8145 LNCS (2013), 369–389.

[18] Tobias Lauinger, Martin Szydlowski, Kaan Onarlioglu, Gilbert Wondracek, Engin Kirda, and Christopher Kruegel. 2013. Clickonomics: Determining the Effect of Anti-Piracy Measures for One-Click Hosting. *Network and Distributed System Security Symposium* (2013), 1–14.

[19] Aravindh Raman, Gareth Tyson, and Nishanth Sastry. 2018. Facebook (A) Live?: Are Live Social Broadcasts Really Broad casts?. In *Proceedings of Web Conference*.

[20] Layton Robert and Watters Paul. 2010. Investigation into the extent of infringing content on BitTorrent networks. *Internet Commerce Security Laboratory* (2010).

[21] Matthew Sag. 2014. Copyright trolling, an empirical study. *Iowa L. Rev.* 100 (2014), 1105.

[22] Michael S Sawyer. 2009. Filters, Fair Use & Feedback: User Generated Content Principle and the DMCA. *Berkeley Technology and Law Journal* (2009).

[23] Daniela Michele Spencer. 2014. Much Ado About Nothing: ICANN's New GTLDs. *Berkeley Tech. LJ* 29 (2014), 865.

[24] Zack Stiegler. 2013. *Regulating the web: network neutrality and the fate of the open internet*. Rowman & Littlefield.

[25] Ernesto TorrentFreak. 2016. Google Asked to Remove 50 Million 4shared Links. https://torrentfreak.com/google-asked-to-remove-50-million-4shared-links-161104/

[26] Gareth Tyson, Yehia Elkhatib, Nishanth Sastry, and Steve Uhlig. 2013. Demystifying porn 2.0: A look into a major adult video streaming website. In *Proceedings of Internet measurement conference*. 417–426.

[27] Gareth Tyson, Yehia Elkhatib, Nishanth Sastry, Steve Uhlig, et al. 2015. Are people really social in porn 2.0?. In *ICWSM*.

[28] Jennifer Urban and Laura Quilter. 2005. Efficient Process or Chilling Effects? Takedown Notices under Section 512 of the Digital Millennium Copyright Act. *Santa Clara Computer & High Technology Law Journal* 22, 4 (2005), 621–693.

[29] Jennifer M Urban, Joe Karaganis, and Brianna L Schofield. 2016. Notice and Takedown in Everyday Practice. *Berkeley Center for Law & Technology* (2016).

[30] Joshua Urist. 2006. Who'S Feeling Lucky-Skewed Incentives, Lack of Transparency, and Manipulation of Google Search Results under teh DMCA. *Brook. J. Corp. Fin. & Com. L.* 1 (2006), 209.

[31] Xiaolei Wang, Sencun Zhu, Dehua Zhou, and Yuexiang Yang. 2017. Droid-AntiRM: Taming Control Flow Anti-analysis to Support Automated Dynamic Analysis of Android Malware. In *Proceedings of the 33rd Annual Computer Security Applications Conference*. ACM, 350–361.

[32] Paul A. Watters, Robert Layton, and Richard Dazeley. 2011. How much material on BitTorrent is infringing content? A case study. *Information Security Technical Report* (2011).

[33] Mary W S Wong. 2009. "Transformative" User-Generated Content in Copyright Law: Infringing Derivative Works or Fair Use? *Vand. J. Ent. & Tech. Law* (2009).

Nonlinear Diffusion for Community Detection and Semi-Supervised Learning

Rania Ibrahim
Computer Science Department,
Purdue University
West Lafayette, Indiana, USA
ibrahimr@purdue.edu

David F. Gleich
Computer Science Department,
Purdue University
West Lafayette, Indiana, USA
dgleich@purdue.edu

ABSTRACT

Diffusions, such as the heat kernel diffusion and the PageRank vector, and their relatives are widely used graph mining primitives that have been successful in a variety of contexts including community detection and semi-supervised learning. The majority of existing methods and methodology involves linear diffusions, which then yield simple algorithms involving repeated matrix-vector operations. Recent work, however, has shown that sophisticated and complicated techniques based on network embeddings and neural networks can give empirical results superior to those based on linear diffusions. In this paper, we illustrate a class of nonlinear graph diffusions that are competitive with state of the art embedding techniques and outperform classic diffusions. Our new methods enjoy much of the simplicity underlying classic diffusion methods as well. Formally, they are based on nonlinear dynamical systems that can be realized with an implementation akin to applying a nonlinear function after each matrix-vector product in a classic diffusion. This framework also enables us to easily integrate results from multiple data representations in a principled fashion. Furthermore, we have some theoretical relationships that suggest choices of the nonlinear term. We demonstrate the benefits of these techniques on a variety of synthetic and real-world data.

CCS CONCEPTS

• **Mathematics of computing** → **Graph algorithms**; • **Computing methodologies** → *Cluster analysis*; • **Theory of computation** → *Semi-supervised learning*.

KEYWORDS

Nonlinear Diffusion; Community Detection; Graph-based Semi-Supervised Learning

ACM Reference Format:
Rania Ibrahim and David F. Gleich. 2019. Nonlinear Diffusion for Community Detection and Semi-Supervised Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313483

1 INTRODUCTION

Diffusion methods for graph mining work by modeling a process where we conceptually inject a *dye* at a particular node or set of nodes in the graph and observe how the *dye* diffused over the edges of the graph.[1] How the *dye* diffuses reveals latent structure in the graph. For instance, if the *dye* stays nearby the source, this indicates that there is a community nearby the source. If the *dye* starts at nodes with a particular label, then under homophily (where neighbors share similar properties), the *dye* will concentrate at nodes likely to share the same label. In this way, the method can be used for semi-supervised learning. Because this conceptual view of diffusions is general, a graph diffusion can be realized in a large number of ways including smooth functions on graphs [57], random walks [38], PageRank equations [37], and linear algebra relaxations of combinatorial graph theory [15, 35]. Consequently, diffusion methods have been widely applied to a large variety of graph mining tasks [13, 18, 20, 24, 32, 38]. In the majority of these results, diffusion methods gave a performance that was comparable to the state of the art, with many possible ways to adjust the computational primitive to improve the results and to scale up to massive problems.

More recently, however, a new class of techniques emerged that raised the state of the art performance on semi-supervised learning problems. These are based on node embeddings along with subsequent machine learning algorithms [22, 39] or from more complicated graph-based variations on neural networks [26, 56]. These methods are impressive because they have established new benchmarks for performance on learning with graph-based data. A downside to these methods, however, is that they can be hard to reason about and can make it difficult to understand what may be causing poor performance on new problems.

Diffusions, on the other hand, are highly intuitive, simple procedures modeled on the *dye* description in the introduction. The vast majority of existing work on diffusions in graphs considers linear diffusions, those where the dynamics can be fully captured by a single matrix operator, such as a random-walk operator or a Laplacian matrix, or a regularized or perturbed version of either [19]. In this paper, we consider nonlinear diffusions, which arise in more

[1] We use the term *dye* here as an intuitive concept only and mimic the ideas of dyes and diffusions in medical diagnostics and physics.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313483

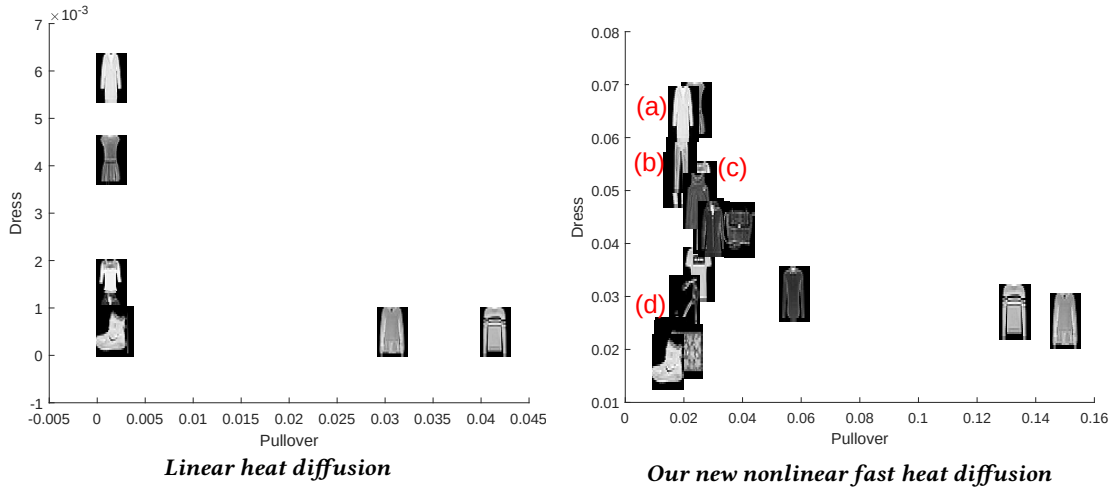


Figure 1: Comparison between linear diffusion based on the linear heat kernel diffusion (left) and our nonlinear diffusion (right) on Fashion MNIST dataset where we seeded the diffusions on a few pictures of pullover sweaters and dresses. This gives a coordinate for each image in the graph based on its diffusion value and these diffusion values reflect how much it looks like a pullover or dress. We draw the same 20 examples for linear diffusion and nonlinear diffusion. The linear diffusion concentrates the results nearby the seeds (most images are on top of each other) where the nonlinear diffusion captures our intuitive notion that there is a diversity of images that look like dresses or pullovers. For example, in the right figure, with respect to the y-axis, the sorting of the diffusion values reflect the semantic similarity between the clothes. Marker (a) points to dresses, marker (b) points to trousers (long as dress), marker (c) points to tops and finally, marker (d) points to shoes.

complicated physics and ecology settings. For instance, the porous media equation [23, 29] describes diffusion through a physical substance that can capture and store the diffusing substance (such as an isentropic gas), slowing the diffusion in a nonlinear fashion. The fast diffusion equation [48] describes diffusion in plasma's [6], and also the diffusion of materials in silicon when the medium is clustered [25]. These diffusions are fairly well understood from an abstract mathematical perspective (e.g. see [7, 48, 49]). With only a few exceptions, however, these ideas have not been applied to graph mining tasks. (See related work for more discussion of what has been done with respect to graphs and networks.) A related type of nonlinearity, which involves using a nonlinear transmission process which diffuses *dye* according to some nonlinear rule, is captured with a nonlinear Laplacian operator such as the p-Laplacian [1, 8, 9]. These have been studied as ways to generalize spectral clustering (see Section 4 for more about the differences). Additionally, it was shown in [36] that using the traditional Laplacian in semi-supervised learning results in labeling function that is spiky at the labeled points and flat everywhere else. Bridle et al. [8] solve this problem using the p-Laplacian in an energy minimization problem and we inherit this advantage when we use the p-Laplacian in the diffusion process.

In this paper, we show that by using values from these nonlinear diffusions, we are able to match state of the art performance benchmarks in semi-supervised learning. This helps to retain the intuitive reasoning behind the methods. As an example of how nonlinear diffusions help to solve problems, see Figure 1, where we show the solution of the standard heat kernel diffusion and nonlinear diffusion for a semi-supervised learning problem on the Fashion MNIST dataset [54]. In this case, we form a graph based on the images themselves (see more details in Section 6.4). Then we diffuse from a few images of a pull-over sweater and a dress with the heat kernel diffusion and our nonlinear diffusion. We show each image (node in the graph) in the after the diffusion has run for some time for the kernel diffusion compared with the nonlinear fast diffusion. As the figure shows, using the nonlinear diffusion causes the points to spread out far more than what is shown in the heat kernel. This illustrates how these advanced diffusions can give a much wider spread of values compared with standard diffusions and enable better classification. Avoiding these *spiked* solutions was also a motivation of past work on nonlinear diffusions [8].

More generally, introducing simple nonlinearities often improves the performance of other machine learning tasks. For example, nonlinear dimensionality reduction techniques [41, 46] outperforms linear techniques by assuming that the data lies in a nonlinear manifold. Nonlinear activation functions enhance the results of neural networks [16], and simple nonlinear transformations of the results of spectral clustering improve results as well [51]. Hence, our work shows yet another way that simple nonlinearities can improve the performance of graph mining tasks.

Finally, one of the great advantages to diffusion methods is that they offer theoretical guarantees of performance through Cheeger inequalities [11–13, 35, 55]. These relate the output from the diffusion to nearby bottlenecks in the graph, formalized as low conductance sets. (See [56] for examples on how general this perspective can be to capture a large number of notions of *conductance*.) In this paper, we have results that suggest using a nonlinear diffusion using the power function with power value close to 0 should give

the best performance for low-conductance set detection through new analysis of the sweepcut procedure (Theorem 5.8).

Our experimental results (Section 6) show that nonlinear diffusion outperforms heat diffusion in community detection by up to 35% in F1-measure and it outperforms related work in graph-based semi-supervised learning in most datasets. In the latter case, we show how our diffusion framework can be adapted to work on multiple datasets (Section 3.2).

Summary of contributions and outline of the paper.

- We introduce two methods for nonlinear diffusion, one using nonlinear functions and another one using nonlinear Laplacian operators (Section 2).
- We place these diffusions in a framework where we can integrate multiple data representations and to self-train itself (Section 3).
- We show that nonlinear diffusion with power function should give smaller conductance sets with p approaches 0. (Section 5).
- We experimentally compare nonlinear diffusion methods to related work in community detection and graph-based semi-supervised learning (Section 6).

2 EVALUATING NONLINEAR DIFFUSIONS

2.1 Notation and Background

We use the following notation throughout the rest of the paper. Scalars are denoted by small letters (e.g., m, n), sets are shown in capital letters (e.g., X, Y), vectors are denoted by small bold letters (e.g., $\mathbf{f}, \mathbf{g}$), and matrices are denoted by capital bold letters (e.g., $\boldsymbol{A}, \boldsymbol{L}$).

Let $G = (V, E)$ be an undirected graph, where $n = |V|$ is the number of nodes in the graph. Let $\boldsymbol{A}$ be the associated adjacency matrix of the graph G and $\boldsymbol{D}$ be the diagonal matrix of degrees, where $\boldsymbol{D}_{ii} = \mathbf{d}(i)$, is the degree of the vertex i. (In the case of a weighted graph, then the weighted degrees are the sum of the edge weights.) Additionally, $\mathbf{e}_S$ is the vector of all ones for entries $i \in S$ and zero otherwise.

Conductance. Conductance is one of the popular quality measures that quantifies how good is the set S as a partition of the graph G [43]. Given a subset S of the vertices V, we define the volume of set S as $\mathrm{vol}(S) = \sum_{v \in S} \mathbf{d}(v)$ and the cut of set S as $\mathrm{cut}(S) = \{\{u, v\} \in E : u \in S \text{ and } v \in \bar{S}\}$. Furthermore, the conductance of the set S is defined as:

$$\phi(S) = \frac{|\mathrm{cut}(S)|}{\mathrm{minvol}(S)},$$

where $\mathrm{minvol}(S) = \min(\mathrm{vol}(S), \mathrm{vol}(V - S))$. For weighted graphs, the volume uses the weighted degrees and the $|\mathrm{cut}(S)|$ is the sum of edge weights cut.

The heat kernel diffusion The heat kernel diffusion [13] from a seed set S results in a solution to the dynamical system:

$$\frac{\partial \mathbf{u}}{\partial t} = -\boldsymbol{L}\mathbf{u} \qquad \mathbf{u}(0) = \frac{\mathbf{e}_S}{|S|},$$

where $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{A}\boldsymbol{D}^{-1}$ is the random walk Laplacian with eigenvalues $\lambda \in [0, 2]$.[2]

[2] Note that the Fiedler vector used in spectral clustering is $\mathbf{f} = \boldsymbol{D}\mathbf{x}_2$, where $\mathbf{x}_2$ is the eigenvector associated with the second smallest eigenvalue of $\boldsymbol{L}$.

We can state a closed-form solution via the matrix exponential:

$$\mathbf{u}(t) = \exp(-t\boldsymbol{L})\mathbf{e}_S/|S|,$$

which has been shown to have a number of nice properties [13, 27].

In the remainder of this section, we will describe two methods to encode nonlinearity in the heat kernel diffusion. The first method is by first applying a nonlinear function g to grow or decay the values at each node and then transmitting the heat according to the heat diffusion differential equation. The second method is by using a nonlinear Laplacian operator that grows or decays the amount transferred on each edge.

2.2 Nonlinear Diffusion with Growth or Decay

The following dynamical system describes a general notion of nonlinear diffusion (e.g. [49]) when applied to a graph or network:

$$\frac{\partial \mathbf{u}}{\partial t} = -\boldsymbol{L}g(\mathbf{u}) \qquad \mathbf{u}(0) = \frac{\mathbf{e}_S}{|S|}.$$

Here g is the nonlinear function, S is the set of seed nodes, and $\mathbf{u}(t) \in \mathbb{R}^n$ is the diffusion value at each node at time t. One representative choice of g is the power function $g(\mathbf{u}) = \mathbf{u}^p$ (element-wise). Using the power function is motivated by the porous media equation ($p > 1$) [47] and the fast diffusion equation ($p < 1$) [7] (see discussion of these in the introduction). Setting $p = 1$ gives the linear heat kernel diffusion [13].

In nonlinear diffusion, each node will apply the function g to its current diffusion value before transmitting the heat to its neighbor. If the values of $\mathbf{u} \in [0, 1]$, then if $p > 1$, the diffusion at each node is reduced first according to both the parameter p and the diffusion value at the node before the node can transmit any heat. This models the diffusion medium *capturing* the value being diffused. Hence, this will keep the diffusion more localized around the seed nodes. However, if $p < 1$, then the diffusion at each node will be magnified first before the node starts transmitting the value, which will help the diffusion explore wider regions. This models settings where large values diffuse faster.

Due to the effect of the nonlinearity, nonlinear diffusion does not have a closed form solution as the heat kernel diffusion, therefore we approximate the solution of the dynamical system by using a simple forward Euler integration, which can be derived by approximating the time-derivative $\partial\mathbf{u}/\partial t \approx (\mathbf{u}(t + h) - \mathbf{u}(t))/h$.[3] Hence, we have the evolution:

$$\mathbf{u}(t + h) = \mathbf{u}(t) - h\boldsymbol{L}\mathbf{u}(t)^p.$$

Here $\mathbf{u}(t + h)$ is the diffusion value at time $t + h$. Because the notion of "time" for this system is arbitrary, we only study the solution after k steps of the evolution process with a fixed value of h. The solution $\mathbf{u}$ for the dynamical systems is non-negative as mentioned in [4] and [47]. However, as we approximate the dynamical system using forward Euler method, the values can be negative. Getting non-negative numerical solutions is difficult as discussed in [44] and therefore, to ensure that $\mathbf{u}$ values are between 0 and 1, we truncate values less than zero to be zero and values greater than one to be one. Algorithm 1 gives a simple pseudocode to do this.

[3] The forward Euler method is often problematic in dynamical systems in physics because there are conservation laws that need to be respected in the approximation. Because the diffusion framework here is an abstraction, we are less worried about accuracy in terms of the ODE, which makes forward Euler a simple, pragmatic choice.

We have ongoing analysis to study accuracy and convergence in the integration problem in more detail that are beyond the scope of this paper; among these results, our analysis gives a direct proof of non-negativity.

Algorithm 1: Nonlinear Diffusion with Growth or Decay

```
NonlinearDiffusionWithGrowthOrDecay (L, k, h, g, s)
  Input : Random walk Laplacian L, number of steps k, step
          length h, nonlinear function g and initial diffusion
          values s.
  Output: f
u = s ;
for i ← 0 to k do
    u = u − hLg(u) ;
    truncate values of u to be between 0 and 1
end
f = g(u);
```

2.3 Nonlinear Diffusion via Nonlinear Transfer

Like the heat kernel diffusion, we can also define nonlinear diffusion using nonlinear transfer. One way to do this is via the p-Laplacian [1, 8, 9]. The p-Laplacian operator [8, 9] is defined as:

$$(\nabla_p \mathbf{f})_i = \sum_{j \in V} w_{ij} \phi_p(\mathbf{f}_i - \mathbf{f}_j),$$

where $\phi_p : \mathbb{R} \to \mathbb{R}$ is

$$\phi_p(x) = |x|^{p-1}\text{sign}(x),$$

and $\mathbf{f} = \boldsymbol{D}^{-1}\mathbf{u}$. This results in a nonlinear Laplacian operator as in the following setting:

$$\frac{\partial \mathbf{u}}{\partial t} = \nabla_p \mathbf{u}.$$

We can write the result of the p-Laplacian on a vector $\mathbf{u}$ using the incidence matrix of the graph. Let $\boldsymbol{N}$ be the edges-by-nodes, unweighted incidence matrix and $\boldsymbol{W}$ be a diagonal matrix of edge-weights that corresponds to the order of edges in the incidence matrix.[4] Then

$$\phi_p(\mathbf{f}_i - \mathbf{f}_j) = |\boldsymbol{N}\boldsymbol{D}^{-1}\mathbf{u}|^{p-1}. * \text{sign}(\boldsymbol{N}\boldsymbol{D}^{-1}\mathbf{u})$$
$$\nabla_p \mathbf{f} = \boldsymbol{N}^T\boldsymbol{W}\,|\boldsymbol{N}\boldsymbol{D}^{-1}\mathbf{u}|^{p-1}. * \text{sign}(\boldsymbol{N}\boldsymbol{D}^{-1}\mathbf{u}).$$

Here, we use the element-wise multiplication .∗, absolute value and power. Additionally, we use the observation that $\phi_p(\boldsymbol{N}\boldsymbol{D}^{-1}\mathbf{u})$ has the same sign as $\boldsymbol{N}\boldsymbol{D}^{-1}\mathbf{u}$. Hence when we multiply by $\boldsymbol{N}^T\boldsymbol{W}$ the arbitrary sign of the difference of edges in the incidence matrix cancels and we scale by the appropriate weight because of $\boldsymbol{W}$. Note that if $p = 2$, then the equation corresponds to the heat kernel diffusion. Algorithm 2 shows the pseudocode for the nonlinear diffusion using the p-Laplacian.

[4]For the unweighted incidence matrix $\boldsymbol{N}$ and an unweighted graph, $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{A}\boldsymbol{D}^{-1} = \boldsymbol{N}^T\boldsymbol{N}\boldsymbol{D}^{-1}$. For a weighted graph, we have $\boldsymbol{L} = \boldsymbol{N}^T\boldsymbol{W}\boldsymbol{N}\boldsymbol{D}^{-1}$. The matrices $\boldsymbol{N}$ and $\boldsymbol{W}$ are easy to build from the raw list of edges. If `ei`, `ej`, `ew` are length $|E| = m$ arrays with the source, destination, and weight of each edge, then $\boldsymbol{N}$ = `sparse([1:m;1:m],[ei;ej],[ones(m,1);-ones(m,1)],m,n)` and $\boldsymbol{W}$ = `diag(ew)` in pseudo-Matlab or Julia notation.

Algorithm 2: Nonlinear Diffusion using the p-Laplacian

```
NonlinearTransfer (N, w, D, k, h, p, s)
  Input : Unweighted incidence N, edge weights w, degree
          matrix D, number of steps k, step length h, nonlinear
          diffusion parameter p and initial diffusion values s.
  Output: f
u = s ;
for i ← 0 to k do
    q = ND^{-1}u ;
    u = u − hN^T[w. ∗ (|q|)^{p−1}. ∗ sign(q)] ;
    truncate values of u to be between 0 and 1
end
f = D^{-1}u;
```

3 USING NONLINEAR DIFFUSIONS

Given an input graph G, we can use the solution vector $\mathbf{f}$ that results from either of the two nonlinear diffusions instead of any of the vectors used in semi-supervised learning [19, 24, 35, 57] or local community detection [13, 27, 55] as a straightforward substitution in these algorithmic pipelines. All of these pipelines feature a *diffuse-and-round* perspective where the results of multiple diffusions are turned into estimated classes in the case of semi-supervised learning or rounded to clusters in the case of community detection. We elaborate more on community detection in Section 5.

To be more precise, for semi-supervised learning, we assume we have $S_1, \ldots, S_c$ as c-seed sets that represent samples of c classes. The goal of the problem is to infer the class label $1, \ldots, c$ for the remaining nodes $V - \cup S_i$. The most interesting case is when $|S_i|$ is small and constant, such as 5 or 10 and independent of the graph size. In which case, we diffuse from S_i using our nonlinear diffusion to compute a vector $\mathbf{f}_i$. With all the vectors $\mathbf{f}_1, \ldots, \mathbf{f}_c$, we assign classes based on their rank in the diffused solutions [19]. That is, node j is the $r_1, \ldots, r_c$th largest node in $\mathbf{f}_1, \ldots, \mathbf{f}_c$, then we assign its class to $\text{argmin}_i\, r_i$. We can also pick the class based on the largest value of node j in any of the diffusions $[\mathbf{f}_i]_j$, which we call value-based rounding [57].

Complexity. The runtime complexity of our diffusions is linear in the number of edges of the graph, because the major piece of computation involves k-matrix vector products – multiplying by a Laplacian or incidence matrix is linear in the number of edges – for each of the c classes.

For the case of semi-supervised learning, our experiments and final methodology incorporate two additional ideas. First, we can use feedback from the diffusion process itself to suggest new training labels [10] (Section 3.1). Second, we show how to easily incorporate multiple data representations into our diffusion framework (Section 3.2).

3.1 Self-Training with Diffusion Methods

In semi-supervised learning, a self-trained diffusion means that the diffusion method incorporates feedback from its results. For instance, if the results of a diffusion $\mathbf{f}_i$ suggest that a subset of node R is extremely like to have label i, then we would re-run the diffusion using seeds $S_i \cup R$. This process is repeated either for a specific

number of iterations or until no significant change is observed in the diffusion method. A confidence score for each label can be calculated based on the diffusion values. The higher the diffusion value, the more confident the diffusion is about its decision. As self-training depends on the decision of the diffusion, if the diffusion yields useful results, then the quality of the diffusion will enhance over time, otherwise, the quality will decrease. Therefore, there is no guarantee that self-training will always enhance the quality of the diffusion, however, in practice it often does.

3.2 Integrating multiple data representations

In many real-world learning problems, the underlying data has multiple representations. For example in citation graphs, there is a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ which captures the relationship between publications based on citations but there are also metadata associated with each paper including authors, titles, and keywords. It is essential to make use of all the available representations in order to develop a better understanding of the data. Consequently, we show how to use 2 different types of relationships between our objects to do prediction. The extension to even more types is simple.

The idea is to construct multiple graphs based on each data representation. For example in citation graphs, we would have $\boldsymbol{A}$ the citation graph, and also $\boldsymbol{B}$, which is a nearest neighbor graph based on some similarity function over the metadata features. Then, we want to run diffusions in $\boldsymbol{A}$ and $\boldsymbol{B}$ simultaneously and our goal is to have all the diffusions concentrated in the same set of nodes. In order to accomplish this goal, each diffusion will try to pull the other diffusions to be similar to it. This changes the nonlinear diffusion differential equation (and similarly the heat kernel diffusion) to:[5]

$$\mathbf{u}_{k+1}^A = \mathbf{u}_k^A - h\boldsymbol{L}_A g(\mathbf{u}_k^A) - \sigma(\mathbf{u}_k^A - \mathbf{u}_k^B)$$
$$\mathbf{u}_{k+1}^B = \mathbf{u}_k^B - h\boldsymbol{L}_B g(\mathbf{u}_k^B) - \sigma(\mathbf{u}_k^B - \mathbf{u}_k^A).$$

Here $\mathbf{u}_k^A$ is the diffusion value for the diffusion running over $\boldsymbol{A}$ at step k and we describe the update to get to the next step. The impact of the parameter σ is the strength of the coupling between the solutions. If σ is large, then we will move the solutions together. If σ is small, then they will evolve with only weak coupling.

4 RELATED WORK

Diffusion methods for clustering, semi-supervised learning, and a variety of other problems have a lengthy history, which has been extensively discussed in various surveys [10, 18, 34]. These methods are all based on linear notions of diffusion. We highlight two important connections we use in the remainder of the paper: spectral clustering and personalized PageRank. For a useful broad perspective on diffusion, we also recommend Ghosh et al. and Yan et al. [17, 55].

Spectral clustering is based on looking for minimum energy modes of the graph Laplacian. These minimum energy modes also can be interpreted as bottlenecks in a diffusion over the nodes. Since this is a linear matrix operator, these minimum energy modes correspond to the small eigenvectors. Specifically, the Fiedler vector is the second smallest generalized eigenvector $\boldsymbol{L}\boldsymbol{D}\mathbf{f} = \lambda \boldsymbol{D}\mathbf{f}$. This eigenvector gives the smallest non-trivial minimizer of the energy function $E(\mathbf{f}) = \mathbf{f}^T \boldsymbol{L}\boldsymbol{D}\mathbf{f}/(\mathbf{f}^T \boldsymbol{D}\mathbf{f})$. The Fiedler vector provably identifies a set of small conductance in the graph [15].

Personalized PageRank (PPR) is a diffusion method that models a random walk with reset process [37]. As a diffusion, it corresponds to the infinite time limit of

$$\frac{d\mathbf{x}}{dt} = (1-\alpha)(\mathbf{e}_S - \mathbf{x}) - \alpha \boldsymbol{L}\mathbf{x},$$

following results on the dynamics of PageRank [21]. PageRank can also be derived as a locally biased analogue of spectral clustering [19], and it is commonly used both for semi-supervised learning [57] and community detection [2], where it also identifies a set of small conductance in the graph. We compare against PPR in our experiments.

4.1 Nonlinear diffusions in graphs.

All of the existing work on nonlinear diffusion for graphs uses the p-Laplacian [1] and seeks to generalize spectral clustering inspired approaches. For instance, Buhler and Hein [9] seek to generalize the Fiedler vector to the p-Laplacian by finding low-energy states. Likewise, Brindel and Zhu [8] seek low-energy states of the p-Laplacian that fix the value of specific nodes to use for semi-supervised learning. Our perspective is different and we seek to generalize methods based on their diffusion interpretations rather than the energy minimization perspectives. For linear diffusions, the two are essentially equivalent because the eigenvectors determine the long-term behavior of the diffusion, but the generalizations are rather different for nonlinear operations.

4.2 Relationship to Neural Networks

Our work can also be interpreted in the context of graph-based neural networks [42]. These view a graph, or a matrix derived from a graph, as defining a layer in a neural network. Hence, we can interpret our diffusion as the output of a graph neural network with a fixed set of weights and a variable number of layers (depending on k). However, in addition to the graph structure, there are other sources of information that are not utilized in these related work like features per node. To use both the graph structure along with the feature matrix, both Planetoid [56] and Graph Convolutional Networks (GCN) [26] were proposed to allow neural networks to jointly learn from the feature matrix along with the graph structure. We compare against both of these methods in our experiments. (In this work, we extend the diffusion methods to learn from different sources of information.)

Compared to neural networks, nonlinear diffusion has two advantages, the first one is that it is more interpretable, as it follows a dynamical equation. Having a dynamical equation makes it easier to reason about its decisions and to develop theoretical results like developing Cheeger-type inequalities. The second advantage is that it reduces the number of parameters (we do no optimization over the weights of the layers) used in neural networks and therefore it is a simpler model.

4.3 Further relationships

Another line of work studies *fractional Laplacians*, which model the diffusion dynamics with fractional powers of the Laplacian matrix [40]. As such, these are still linear diffusions, albeit on a

[5] The equations for the p-Laplacian formulation are changed in an analogous fashion.

non-linear transformation of the Laplacian matrix. Our approach to integrating the results of multiple representations bears some relationship with recent work on centrality in temporal networks [45], although the motivation and applications are distinct. Finally, there are methods that use flow techniques for semi-supervised learning [50, 52]. These techniques offer fundamentally different tradeoffs. In our view, they are often best used to refine the results of diffusion algorithms to achieve better downstream performance.

5 NONLINEAR DIFFUSION ANALYSIS

The goal of this section is to prove our main theoretical result, which guides how we would pick a nonlinear diffusion to find small conductance sets in a graph. The precise method we study is as follows. We use Algorithm 1 to diffuse from a seed set S to produce a diffusion vector $\mathbf{f}$. (Note that $\mathbf{f}$ depends on the seed set, the value h, and the number of steps k.) Then we perform a sweep-cut procedure over the vector $\mathbf{f}$ to identify a set of small conductance. The sweep-cut procedure works by building a permutation $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ that sorts $\mathbf{f}$ from large elements to small elements, and then evaluates the conductance of each sorted prefix $S_i = (\sigma_1, \ldots, \sigma_i)$. We return the set $S = \operatorname{argmin}_i \phi(S_i)$ where $\phi(S_i)$ is the conductance.

This proof of our main results proceeds in two stages. First, we show a relationship between the sweep over any vector $\mathbf{f}$ and the set of minimal conductance (Section 5.2) that shows how well an "approximate indicator" vector behavior. Second, we will use the properties of nonlinear diffusion to guide our choice for nonlinear diffusion.

5.1 f-weighted Conductance

Before we begin, we fix a variety of notation that we will use throughout the proof and introduce a specialized notion of conductance, that we term f-weighted conductance. Our objective is to develop an inequality that relates the smallest conductance of any set, ϕ^*, in the graph G to the conductance of performing sweepcut over the vector $\mathbf{f}$, which results in conductance $\phi(\mathbf{f})$. Additionally, we want this inequality to maintain a linear relationship between ϕ^* and $\phi(\mathbf{f})$. This will require quantifying how well $\mathbf{f}$ looks like an indicator vector over the optimal set.

Using $\mathbf{f}$, we will direct each undirected edge $e = (u, v)$ in the graph G, such that it points from the node with higher $\mathbf{f}$ value to the node with lower $\mathbf{f}$ value. This re-direction will encode the sorted order of the nodes and will ensure that $\mathbf{f}(u) - \mathbf{f}(v) \geq 0$ for all $(u, v) \in E$. To encode this re-direction using matrices, we will define $\boldsymbol{N}_\mathbf{f} \in \{0, 1, -1\}^{m \times n}$ to be the $\mathbf{f}$-oriented incidence matrix, where for each edge $e = (u, v)$, $\boldsymbol{N}_\mathbf{f}(e, u) = 1$ and $\boldsymbol{N}_\mathbf{f}(e, v) = -1$ if and only if $\mathbf{f}(u) \geq \mathbf{f}(v)$. We will use $x \sim y$ to iterate over the directed edges. Additionally, assume that the vertices are numbered descending according to $\mathbf{f}$, so that we have $\mathbf{f}(v_1) \geq \mathbf{f}(v_2) \geq \ldots \geq \mathbf{f}(v_n)$ and let us first normalize $\mathbf{f}$ so that its values are between 0 and 1.

To accomplish our objective, we need to define a relationship between the set S^* of smallest conductance in the graph and the set obtained by performing a sweepcut over $\mathbf{f}$. Therefore, we define the f-weighted conductance $\phi^*(\mathbf{f})$ as:

$$\phi^*(\mathbf{f}) = \frac{|\mathrm{cut}_\mathbf{f}(S^*)|}{\mathrm{minvol}_\mathbf{f}(S^*)},$$

where $|\mathrm{cut}_\mathbf{f}(S^*)| = (\mathbf{e}^T_{S^* \to \bar{S}^*} + \mathbf{e}^T_{\bar{S}^* \to S^*})\boldsymbol{N}_\mathbf{f}\mathbf{f}$, $\mathbf{e}_{S \to \bar{S}}$ is the indicator vector where only edges pointing from S to $\bar{S}$ have value one and other edges have value zero, and $\mathrm{minvol}_\mathbf{f}(S^*) = \min(\sum_{u \in S^*} \mathbf{f}(u)\mathbf{d}(u), \mathrm{vol}(G) - \sum_{u \in S^*} \mathbf{f}(u)\mathbf{d}(u))$. If $\mathbf{f} = \mathbf{e}_{S^*}$, then $\phi^*(\mathbf{f}) = \phi^*$.

5.2 A General Sweepcut Inequality

To derive a general sweepcut inequality, we will derive an upper bound and a lower bound for $\phi^*(\mathbf{f})$ and using these bounds, we will be able to relate $\phi(\mathbf{f})$ to ϕ^* linearly. Additionally, let us define W as the set of nodes that have nonzero $\mathbf{f}$ values.

Lemma 5.1. *If $vol(W) \leq minvol(S^*)$, then:*

$$\phi^*(\mathbf{f}) \geq \frac{\phi(\mathbf{f})\, vol_\mathbf{f}(W) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{minvol_\mathbf{f}(S^*)},$$

Proof. The result is largely algebraic:

$$\begin{aligned}
\phi^*(\mathbf{f}) &= \frac{|\mathrm{cut}_\mathbf{f}(S^*)|}{\mathrm{minvol}_\mathbf{f}(S^*)} \\
&= \frac{(\mathbf{e}^T_{S^* \to \bar{S}^*} + \mathbf{e}^T_{\bar{S}^* \to S^*})\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\mathrm{minvol}_\mathbf{f}(S)} \\
&= \frac{\mathbf{e}^T\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\mathrm{minvol}_\mathbf{f}(S^*)} \\
&= \frac{\sum_{x \sim y}(\mathbf{f}(x) - \mathbf{f}(y)) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\mathrm{minvol}_\mathbf{f}(S)}.
\end{aligned}$$

Note that due to numbering of the vertices descending according to $\mathbf{f}$, $(\mathbf{f}(x) - \mathbf{f}(y))$ is always a non negative number. Moreover, using $\sum_{x \sim y}(\mathbf{f}(x) - \mathbf{f}(y)) = \sum_{i=1}^{n-1}(\mathbf{f}(x_i) - \mathbf{f}(x_{i+1}))|\mathrm{cut}(S_i)|$ from [13], where S_i is the set of nodes labeled v_1 to v_i, we get:

$$\begin{aligned}
\phi^*(\mathbf{f}) &= \frac{\sum_{i=1}^{n-1}(\mathbf{f}(x_i) - \mathbf{f}(x_{i+1}))|\mathrm{cut}(S_i)| - (\mathbf{e}^T_{S^* \to S^*} + \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*})\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\mathrm{minvol}_\mathbf{f}(S^*)} \\
&\geq \frac{\phi(\mathbf{f})\sum_{i=1}^{n-1}(\mathbf{f}(x_i) - \mathbf{f}(x_{i+1}))\,\mathrm{minvol}(S_i) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\mathrm{minvol}_\mathbf{f}(S^*)} \\
&\quad - \frac{\mathbf{e}^T_{\bar{S}^* \to \bar{S}^*}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\mathrm{minvol}_\mathbf{f}(S^*)}.
\end{aligned}$$

As $\mathrm{vol}(W) \leq \mathrm{minvol}(S^*)$, we get:

$$\begin{aligned}
\phi^*(\mathbf{f}) &\geq \frac{\phi(\mathbf{f})\sum_{x \in W}\mathbf{f}(x)\mathbf{d}(x) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\mathrm{minvol}_\mathbf{f}(S^*)} \\
&\geq \frac{\phi(\mathbf{f})\mathrm{vol}_\mathbf{f}(W) - \mathbf{e}^T_{S^* \to S^*}\boldsymbol{N}_\mathbf{f}\mathbf{f} - \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*}\boldsymbol{N}_\mathbf{f}\mathbf{f}}{\mathrm{minvol}_\mathbf{f}(S^*)}.
\end{aligned}$$

□

Lemma 5.2. *We also have*

$$\phi^*(\mathbf{f}) \leq \phi^* \frac{minvol(S^*)}{minvol_\mathbf{f}(S^*)}.$$

Proof. By definition,

$$\phi^*(\mathbf{f}) = \frac{|\mathrm{cut}_\mathbf{f}(S^*)|}{\mathrm{minvol}_\mathbf{f}(S^*)}.$$

Because $\mathbf{f} \in [0,1]$, then the maximum value for $\mathbf{f}(i) - \mathbf{f}(j)$ is 1 for any edge (i,j) and therefore we get:

$$\phi^*(\mathbf{f}) \leq \frac{|\text{cut}(S^*)|}{\text{minvol}_{\mathbf{f}}(S^*)} = \phi^* \frac{\text{minvol}(S^*)}{\text{minvol}_{\mathbf{f}}(S^*)}.$$

□

Theorem 5.3. *If we run a sweep-cut procedure on* $\mathbf{f}$ *where* $0 \leq f_i \leq 1$ *and* $vol(W) \leq minvol(S^*)$*, then:*

$$\phi(\mathbf{f}) \leq \phi^* \frac{minvol(S^*)}{vol_{\mathbf{f}} W} + \frac{\mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f}}{vol_{\mathbf{f}}(W)}.$$

Proof. By using the two previous lemmas and re-arranging the terms, we can prove the theorem. □

Note that the previous generalized sweepcut inequality depends on $\mathbf{f}$, W and S^*. If $\mathbf{f}$ is far from highlighting the set S^* or $\text{vol}(W)$ is small, then this bound will reduce to the trivial bound $\phi(\mathbf{f}) \leq 1$. In the next section, we will use this generalized sweepcut inequality in addition to nonlinear diffusion properties to show that if the power function is used, then using small powers yields smaller conductance sets.

5.3 Implications for Nonlinear Diffusion

Lemma 5.4. *The sum property of nonlinear diffusion with growth or decay is:*

$$\mathbf{e}^T \mathbf{u}_t = \mathbf{e}^T \mathbf{u}_0 = 1.$$

Proof. By multiply both sides of the differential equation for nonlinear diffusion with growth or decay by $\mathbf{e}^T$ and using $\mathbf{e}^T \boldsymbol{L} = 0$, we get $\frac{\partial \mathbf{u}}{\partial t} = 0$. Therefore, the sum of the entries of $\mathbf{u}$ is always one. □

We will set $\mathbf{f} = g(\mathbf{u})$ and start our proofs using the general sweepcut inequality proposed in the previous section.

Lemma 5.5. *If* g *satisfies* $g(\mathbf{u}(x)) - g(\mathbf{u}(y)) \leq \mathbf{u}(x) - \mathbf{u}(y)\ \forall x, y \in W$*, then:*

$$\mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} \leq |W|.$$

Proof. By noting that $\mathbf{f}$ is nonzero only at the set W, we get:

$$\mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} \leq \mathbf{e}^T_{W \to W} \boldsymbol{N}_{\mathbf{f}} \mathbf{f}.$$

If g satisfies $g(\mathbf{u}(x)) - g(\mathbf{u}(y)) \leq \mathbf{u}(x) - \mathbf{u}(y)\ \forall x, y \in W$, then:

$$\begin{aligned} \mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} &\leq \mathbf{e}^T_{W \to W} \boldsymbol{N}_{\mathbf{f}} \mathbf{u} \\ &\leq \sum_{x \in W} \mathbf{u}(x) |N(x) \cap W|. \end{aligned}$$

Here $N(x)$ is the set of nodes that are neighbors to x in the graph.

$$\mathbf{e}^T_{S^* \to S^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} + \mathbf{e}^T_{\bar{S}^* \to \bar{S}^*} \boldsymbol{N}_{\mathbf{f}} \mathbf{f} \leq \sum_{x \in W} \mathbf{u}(x) |W| = |W|.$$

□

Lemma 5.6.

$$vol_{\mathbf{f}}(W) \geq min_{i \in W} \mathbf{f}_i\, vol(W).$$

Proof.

$$\text{vol}_{\mathbf{f}}(W) = \mathbf{e}_W^T \boldsymbol{D} \mathbf{f} \geq \min_{i \in W} \mathbf{f}_i \mathbf{e}_W^T \boldsymbol{D} \mathbf{e}_W = \min_{i \in W} \mathbf{f}_i \text{vol}(W).$$

□

Theorem 5.7. *Sweepcut Inequality for Nonlinear Diffusion with growth or decay:*

If g *satisfies* $g(\mathbf{u}(x)) - g(\mathbf{u}(y)) \leq \mathbf{u}(x) - \mathbf{u}(y)\ \forall x, y \in W$ *and* $vol(W) \leq minvol(S^*) \leq v$*, then:*

$$\phi(\mathbf{f}) \leq \phi^* \frac{v}{min_{i \in W} \mathbf{f}_i\, vol(W)} + \frac{|W|}{min_{i \in W} \mathbf{f}_i\, vol(W)}.$$

Proof. We can prove the theorem using the two previous lemmas. □

Theorem 5.8. *If* $minvol(S^*) \leq v$*,* $g(\mathbf{u}) = \mathbf{u}^p$*,* p *tends to zero,* $\mathbf{d}_{avg}(W) \geq \frac{\alpha}{\phi^*}$*, and* $vol(W) \leq minvol(S^*)$*, then the nonlinear diffusion sweepcut inequality becomes:*

$$\phi(\mathbf{f}) \leq (\frac{v}{vol(W)} + \frac{1}{\alpha})\phi^*.$$

Proof. By noting that, as p tends to zero, $\mathbf{u}$ tends to an indicator and therefore, the condition $g(\mathbf{u}(x)) - g(\mathbf{u}(y)) \leq \mathbf{u}(x) - \mathbf{u}(y) \forall x, y \in W$ is always satisfied. In this case, $\mathbf{f}$ will tend to be $\mathbf{e}_W$ and the following properties applies for $\mathbf{e}_W$. $\min_{i \in W} \mathbf{f}_i \text{vol}(W)$ will tend to $\text{vol}(W)$ and we can observe that:

$$\frac{|W|}{\min_{i \in W} \mathbf{f}_i \text{vol}(W)} = \frac{|W|}{\text{vol}(W)} = \frac{|W|}{|W| \mathbf{d}_{\text{avg}}(W)} = \frac{1}{\mathbf{d}_{\text{avg}}(W)}.$$

As $\mathbf{d}_{\text{avg}}(W) \geq \frac{\alpha}{\phi^*}$ and re-arranging the terms, we can prove the theorem. Note that we can control $\text{vol}(W)$ using k, the number of steps, as increasing k increases $\text{vol}(W)$. □

At its heart, this theorem states that if the result of the diffusion is close to an indicator vector, then we will have some relationship between the set of smallest conductance and the result. This suggests using the power function with small power as it will tend to encourage "indicator-like" behavior for any number of time-steps. If the graph has an average degree of $\Omega(\frac{1}{\phi^*})$, then the nonlinear diffusion conductance will be close to the smallest conductance in the graph by a multiplicative factor and the larger the average degree of the graph, the smaller the multiplicative factor. Additionally, we can use this sweepcut inequality to tell us how close the nonlinear diffusion conductance is to the smallest conductance in the graph by simply computing the constants. On the other hand, this theorem only provides coarse guidance and essentially requires a large degree of knowledge of the optimal set to yield non-trivial results.

6 EXPERIMENTS

The goal of our experiments is to show the advantage of our nonlinear diffusion process in a variety of settings. We first show that it gives better F1-values in LFR synthetic graphs compared with linear diffusions, where we get the best results with small p as hinted at by our theory results. Second, in community detection on SNAP graphs, the non-linear diffusion gives consistently higher $F1$ scores on all graphs except one. Third, on graph-based semi-supervised learning, we show that nonlinear diffusion improves substantially on recently developed methods based

on neural networks in two of three tests. Finally, we conclude our experiments by trying to understand why nonlinear diffusion gives better results. Thus, we conduct a case study in the Fashion MNIST data and argue that the reason nonlinear diffusions perform better is because they produce vectors with higher dispersion than linear diffusions. Our experimental code is available at: https://github.com/RaniaSalama/Nonlinear_Diffusion

6.1 Synthetic Data

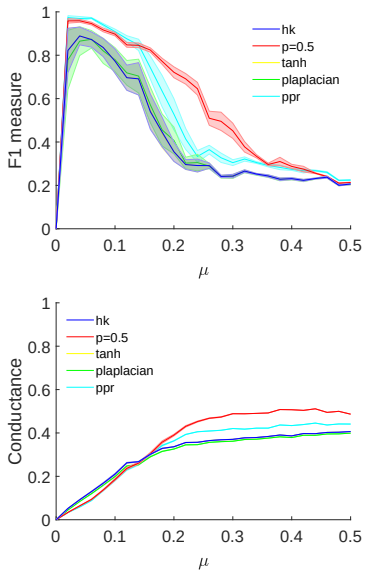


Figure 2: The F1 measure and the conductance of various techniques on LFR synthetic datasets. The bands show the variation over 100 trials.

We use the LFR model [28] to generate the synthetic datasets as it is a widely used model in evaluating community detection algorithms. The parameters used for the model are $n = 1000$, average degree is 10, maximum degree is 50, minimum community size is 20 and maximum community size is 100. LFR node degrees and community sizes are distributed according to the power law distribution, we set the exponent for the degree sequence to 2 and the exponent for the community size distribution to 1. We vary μ the mixing parameter from 0.02 to 0.5 with step 0.02 and run the heat kernel diffusion (hk), personalized PageRank (ppr), nonlinear diffusion using power function with $p = 0.5$, nonlinear diffusion using tanh function and nonlinear diffusion using p-Laplacian. In all the experiments, we set $k = 100$ and $h = 0.001$. For each graph, we start from a random seed node and we repeat the experiment 100 times. Figure 2 shows the mean and the variance for the F1 measure and the conductance while varying μ. As shown in the figure, nonlinear diffusion using power function with $p = 0.5$ has a better F1 measure than the heat kernel diffusion and has lower conductance than it for $\mu = 0.02$ to 0.14. The conductance for the nonlinear diffusion gets slightly higher after $\mu = 0.14$ but the increase in F1 measure is much higher. For example, for $\mu = 0.24$, the increase in conductance of $p = 0.5$ over the heat kernel diffusion is $\approx 9\%$, while the increase in F1 measure is $\approx 30\%$.

6.2 Local Community Detection

Local community detection is the task of finding the community when given a member of that community. In this task, we start the diffusion from the given node. Table 1 shows the community detection results for heat kernel diffusion (hk), personalized PageRank (ppr), nonlinear diffusion using the power function as the nonlinear function (power) with $p = 0.5$, nonlinear diffusion using the tanh function (tanh) and nonlinear diffusion using the p-Laplacian operator (p-Lap) with $p = 1.9$. In all nonlinear diffusions, we set $k = 10, h = 0.001$. We have followed the same evaluation settings as mentioned in [27]. In these experiments, we identify 100 communities from the ground truth such that each community has a size greater than 10, then for all algorithms, we start from each node in the community and finally report the result from the node that yields the best F1 measure. For nonlinear diffusion we truncate the diffusion values that are less than ε to be zero and the diffusion values that are greater than one to be one, where ε is designed to give 100 nodes in the final community, where we grow larger if necessary.

Table 1 shows datasets from SNAP repository [30], their characteristics, the mean F1 measure and the mean set size for the heat kernel diffusion, the personalized PageRank and the nonlinear diffusions. These results show that nonlinear diffusion can return a larger set than the heat kernel diffusion with higher or comparable F1 score. For example, in YouTube dataset, the detected communities for nonlinear diffusion have a higher F1 measure by around 35%.

6.3 Graph-based Semi-supervised Learning

Semi-supervised learning is the task of learning from a small set of labeled examples and a very large set of unlabeled examples. In these experiments, we use three citation networks, which are Cora, Citeseer and Pubmed. Table 2 shows the characteristic of these datasets. We run two coupled nonlinear diffusions as in Section 3.2, one using the adjacency matrix and another one using an r nearest neighbor graph constructed from the feature matrix using radial basis function with width equal 1. Following the exact methodology of [26], we choose the hyperparameters based on a train-validate-test split where we pick the parameters based on the ones that have the best classification accuracy on the validation set and then show the evaluation on the test set. The parameters are the number of nearest neighbors r, the number of steps k, the value of the power function p_1 for the adjacency diffusion and p_2 for the nearest neighbor diffusion, the value of σ, and value or rank rounding. For reproducibility, the parameters we find are given in a footnote.[6]

[6]For all the experiments, we set $h = 0.001$. Let r be the number of nearest neighbors used in the graph construction. For the power function, in Cora, we set the $r = 100$, $k = 400$, $\sigma = 0.55$, $p_1 = 0.6$, $p_2 = 0.45$ and used value-based rounding; in Citeseer we set $r = 200$, $k = 400$, $\sigma = 0.51$, $p_1 = 0.55$, $p_2 = 0.55$ and used rank-based rounding; and in Pubmed we set $r = 2000$, $k = 300$, $\sigma = 0.65$, $p_1 = 0.7$, $p_2 = 0.6$ and used rank-based rounding. For nonlinear diffusion using tanh function, in Cora we set $r = 100$, $k = 450$, $\sigma = 1.9$ and used value-based rounding; in Citeseer we set $r = 200$, $k = 450$, $\sigma = 1.9$ and used rank-based rounding; and in Pubmed wet set $r = 2000$, $k = 400$, $\sigma = 1.8$ and used rank-based rounding. Finally, for p-Laplacian, for Cora, we set $r = 100$, $p_1 = 1.9$, $p_2 = 1.7$, $\sigma = 0.05$, $k = 400$; for Citeseer,

Table 1: Community detection results.

	|V|	|E|	F1 measure					Set size				
			hk	ppr	power	tanh	p-Lap	hk	ppr	power	tanh	p-Lap
DBLP	317,080	1,049,866	0.364	0.273	**0.462**	0.401	0.403	156	4,529	599.59	113.5	136.21
Amazon	334,863	925,872	**0.608**	0.415	0.559	0.505	0.476	145	5,073	479.76	34.70	74.20
YouTube	1,134,890	2,987,624	0.128	0.078	**0.470**	0.388	0.422	137	5,833	89.46	680.60	215.82
LiveJournal	3,997,962	34,681,189	0.138	0.104	0.251	0.258	**0.269**	156	299	104.29	773.35	717.49

Table 2: Characteristics of datasets used in graph-based semi-supervised learning experiments.

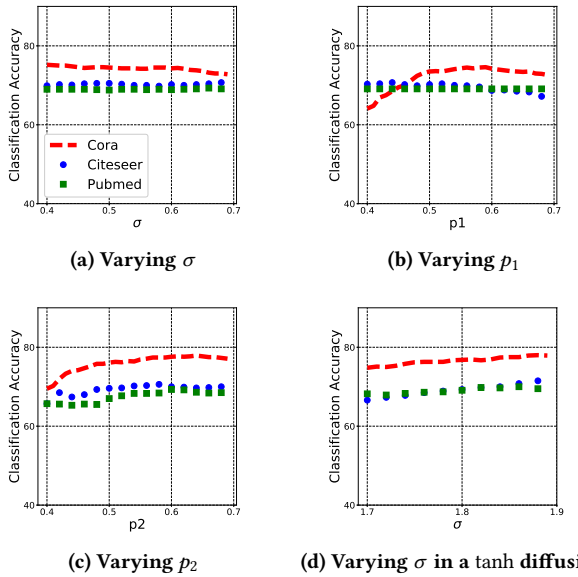
	|V|	|E|	|Classes|	d	Label rate
Citeseer	3,327	4,732	6	3,703	0.036
Cora	2,708	5,429	7	1,433	0.052
Pubmed	19,717	44,338	3	500	0.003

Table 3: Classification accuracy for graph-based semi-supervised learning experiments.

	Cora	Citeseer	Pubmed
ManiReg [3]	59.5	60.1	70.7
SemiEmb [53]	59.0	59.6	71.1
LP [58]	68.0	45.3	63.0
DeepWalk [39]	67.2	43.2	65.3
ICA [33]	75.1	69.1	73.9
Planetoid [56]	75.7	64.7	77.2
GCN [26]	81.5	70.3	**79.0**
GCN + self learning [31]	80.2	67.8	76.9
hk	61.4	53.8	63.3
power	74.2	70.6	70.0
power + self-learning	71.3	68.9	70.3
tanh	77.9	**73.4**	71.0
tanh + self-learning	**82.9**	72.2	73.0
p-Lap	79.8	68.7	73.4
p-Lap + self-learning	79.2	67.8	73.4

Table 3 shows the classification accuracy of nonlinear diffusion compared to other graph-based semi-supervised learning related work. As shown in the table nonlinear diffusion using tanh function has the best classification accuracy in Cora and Citeseer. However, GCN has the best classification accuracy in Pubmed. Note that,

we set $r = 300$, $p_1 = 1.8$, $p_2 = 1.7$, $\sigma = 0.7$, $k = 1000$; and for Pubmed, we set $r = 100$, $p_1 = 1.9$, $p_2 = 1.7$, $\sigma = 0.05$, $k = 400$.



(a) Varying σ **(b) Varying p_1**

(c) Varying p_2 **(d) Varying σ in a tanh diffusion**

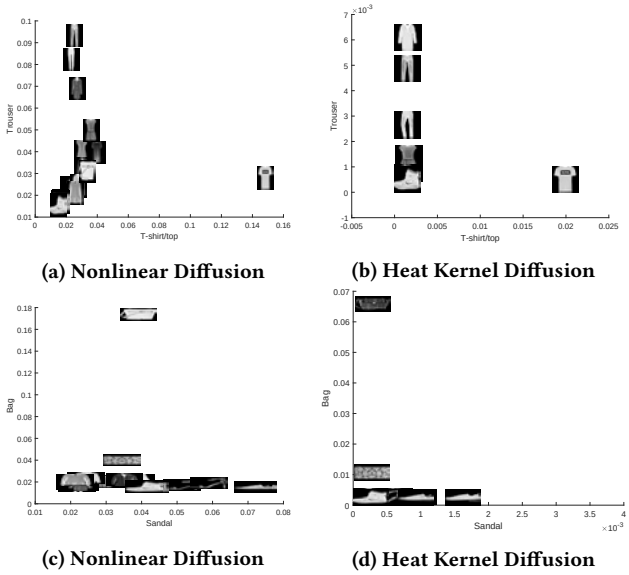
Figure 3: As we vary parameters, the accuracy remains high for many choices.

nonlinear diffusion uses less parameters as we do no optimization over the weights of the layers and therefore it is the simplest model with good accuracy. Additionally, the dynamics of the nonlinear diffusion is described by a differential equation and therefore it is easier to reason about and to develop bounds for its performance.

To further analyze the stability of nonlinear diffusion when the parameters change, Figure 3 shows changing σ, p_1 and p_2 for nonlinear diffusion using power function from 0.4 to 0.7 with step of 0.02 and changing σ for nonlinear diffusion using tanh function from 1.7 to 1.9 with step 0.02. As shown in the figure, nonlinear diffusion is robust to small changes in the parameters.

6.4 Fashion MNIST Case Study

Our final experiment is a case study of the Fashion MNIST dataset to understand why nonlinear diffusion outperforms linear diffusion. The Fashion MNIST dataset is a collection of Zalando's clothing images [54] with 70,000, 28 by 28 grayscale images. Each example is assigned a label from 10 classes, which are: T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag and Ankle Boot.



(a) Nonlinear Diffusion (b) Heat Kernel Diffusion

(c) Nonlinear Diffusion (d) Heat Kernel Diffusion

Figure 4: Comparison between the heat kernel diffusion and the nonlinear diffusion on Fashion-MNIST dataset.

Table 4: Heat kernel diffusion and nonlinear diffusion error rates on Fashion-MNIST dataset (k = 1000, h = 0.001, labels per class = 10, and r = 100).

	hk	power
T-shirt/top	0.466 ± 0.211	**0.362 ± 0.228**
Trouser	0.180 ± 0.084	**0.061 ± 0.017**
Pullover	0.609 ± 0.140	**0.582 ± 0.208**
Dress	0.547 ± 0.171	**0.481 ± 0.275**
Coat	0.516 ± 0.199	**0.502 ± 0.220**
Sandal	0.546 ± 0.165	**0.499 ± 0.256**
Shirt	**0.795 ± 0.109**	0.820 ± 0.098
Sneaker	0.217 ± 0.116	**0.196 ± 0.137**
Bag	0.528 ± 0.118	**0.403 ± 0.130**
Ankle boot	0.325 ± 0.138	**0.188 ± 0.190**
Total	0.473 ± 0.030	**0.409 ± 0.040**

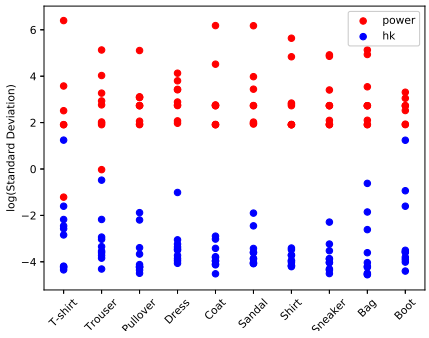


Figure 5: Dispersion comparison between the heat kernel diffusion values and nonlinear diffusion.

We construct a r-nearest neighbor graph from the data. Then, we randomly select 10 examples per class to start the semi-supervised learning task. Finally, we repeat the experiments 10 times and report the error rate over the 70,000 examples. This means that we only use around 0.14% of examples for the learning task.

Table 4 shows the error rates for the heat kernel diffusion and the nonlinear diffusion using power function with $p = 0.5$ for each different classes. These results show that nonlinear diffusion has a lower error rate than the heat kernel diffusion in all class except the Shirt class. Additionally, overall the error rate for the nonlinear diffusion was lower by around 7%.

To further study the difference between the two techniques, Figure 4 (a, b) compares values of the diffusion as in the introduction (Figure 1). Specifically, these figures are run on a random sample of 1000 images where we start from a single T-shirt/top example in the x-axis and the diffusion values when we start from a single coat example in the y-axis. Then we show 2 randomly chosen pictures from each class (so 20 pictures total). As shown in the figures, the heat kernel diffusion is extremely spiky for the different examples, while nonlinear diffusion is able to distinguish between different examples to provide more information to downstream operators. Figure 4 (c, d) shows the same comparison for Sandals and Bags.

We did one final evaluation to understand this advantage. Our conjecture is that the nonlinear diffusion produces values with far more dispersion over the range than the heat kernel. In Figure 5, we draw the log of the standard deviation of the diffusion values when we start the diffusion from each class for each trial. As shown in the figure, the nonlinear diffusion values show much higher standard deviations than the heat kernel diffusion, which supports this point.

7 DISCUSSION AND FUTURE WORK

In this paper, we have shown that nonlinear diffusion is competitive with embedding techniques and significantly outperform classical diffusions in community detection and semi-supervised learning. As a simple diffusion model, there are ample opportunities to accelerate the computation using a wide variety of techniques and we have not made that a focus of our work. For instance, there are Monte Carlo [14] and localized algorithms [2, 27] for linear diffusions that yield sublinear complexity and millisecond-level runtimes on billion edge graphs. In comparison, the complexity of our current implementation is linear (the work is k matrix-vector products). Note that, in our experience, the time required to evaluate these diffusions is dominated by the initial data load, ingestion, and r-nearest neighbor computation so we do not view this as problematic, but for large numbers of classes, faster runtime may be useful. We also note that our approach can easily be combined with recent work on higher-order analysis of graphs based on motifs [5]. Specific things we will explore in the future include how to choose the nonlinear function g in an automated fashion by studying the properties of target sets in the graph. Additionally, we hope to provide improved algorithms to evaluate the nonlinear ODEs themselves.

Acknowledgments. Both authors would like to acknowledge our colleagues who provided initial feedback on this paper. This work is supported by NSF awards CCF-1149756, IIS-1422918, IIS-1546488; NSF Center for Science of Information STC, CCF-0939370, DARPA SIMPLEX, and the Sloan Foundation.

REFERENCES

[1] S. Amghibech. 2003. Eigenvalues of the Discrete p-Laplacian for Graphs. *Ars Comb.* 67 (2003).

[2] Reid Andersen, Fan Chung, and Kevin Lang. 2006. Local Graph Partitioning using PageRank Vectors. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, IEEE, Washington, D.C., USA, 475–486. http://www.math.ucsd.edu/~fan/wp/localpartition.pdf

[3] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research* 7, Nov (2006), 2399–2434.

[4] Philippe Bénilan and Michael G Crandall. 1981. The continuous dependence on φ of solutions of u t- $\Delta\varphi$ (u)= 0. *Indiana University Mathematics Journal* 30, 2 (1981), 161–177.

[5] Austin Benson, David F. Gleich, and Jure Leskovec. 2016. Higher-order organization of complex networks. Submitted. *Science* 353, 6295 (2016), 163–166. https://doi.org/10.1126/science.aad9029

[6] James G Berryman and Charles J Holland. 1978. Nonlinear diffusion problem arising in plasma physics. *Physical Review Letters* 40, 26 (1978), 1720.

[7] M. Bonforte, J. Dolbeault, G. Grillo, and J. L. Vazquez. 2010. Sharp rates of decay of solutions to the nonlinear fast diffusion equation via functional inequalities. *Proceedings of the National Academy of Sciences* 107, 38 (sep 2010), 16459–16464. https://doi.org/10.1073/pnas.1003972107

[8] Nick Bridle and Xiaojin Zhu. 2013. p-voltages: Laplacian regularization for semi-supervised learning on high-dimensional data. In *Eleventh Workshop on Mining and Learning with Graphs (MLG2013)*. ACM, New York, NY, USA.

[9] Thomas Bühler and Matthias Hein. 2009. Spectral clustering based on the graph p-Laplacian. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, ACM, New York, NY, USA, 81–88.

[10] O Chapelle and B Sch. 2006. olkopf, and A. Zien, Semi-Supervised Learning, vol. 2.

[11] Fan Chung. 2005. Laplacians and the Cheeger inequality for directed graphs. *Annals of Combinatorics* 9, 1 (2005), 1–19.

[12] Fan Chung. 2007. Four proofs for the Cheeger inequality and graph partition algorithms. In *Proceedings of ICCM*, Vol. 2. ScienTech, Mason, OH, USA, 378.

[13] Fan Chung. 2007. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences* 104, 50 (2007), 19735–19740.

[14] Fan Chung and Olivia Simpson. 2013. Solving Linear Systems with Boundary Conditions Using Heat Kernel Pagerank. In *Algorithms and Models for the Web Graph*. Springer, 203–219.

[15] Fan R. L. Chung. 1992. *Spectral Graph Theory*. American Mathematical Society, Providence, RI, USA.

[16] George Cybenko. 1989. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems* 2, 4 (1989), 303–314.

[17] Rumi Ghosh, Shang-hua Teng, Kristina Lerman, and Xiaoran Yan. 2014. The interplay between dynamics and networks: centrality, communities, and cheeger inequality. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, ACM, New York, NY, USA, 1406–1415.

[18] David F. Gleich. 2015. PageRank beyond the Web. *SIAM Rev.* 57, 3 (August 2015), 321–363. https://doi.org/10.1137/140976649

[19] David F Gleich and Michael W Mahoney. 2015. Using local spectral methods to robustify graph-based learning algorithms. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, ACM, New York, NY, USA, 359–368.

[20] David F. Gleich and Michael W. Mahoney. 2016. Mining large graphs. In *Handbook of Big Data*, Peter Bühlmann, Petros Drineas, Michael Kane, and Mark van de Laan (Eds.). CRC Press, Boca Raton, Florida, USA, 191–220. https://doi.org/10.1201/b19567-17

[21] David F. Gleich and Ryan A. Rossi. 2014. A Dynamical System for PageRank with Time-Dependent Teleportation. *Internet Mathematics* 10, 1–2 (June 2014), 188–217. https://doi.org/10.1080/15427951.2013.814092

[22] Aditya Grover and Jure Leskovec. 2016. Node2Vec: Scalable Feature Learning for Networks. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 855–864. https://doi.org/10.1145/2939672.2939754

[23] FM Henderson and RA Wooding. 1964. Overland flow and groundwater flow from a steady rainfall of finite duration. *Journal of Geophysical Research* 69, 8 (1964), 1531–1540.

[24] Thorsten Joachims. 2003. Transductive Learning via Spectral Graph Partitioning. In *ICML*. ACM, New York, NY, USA, 290–297. http://www.aaai.org/Papers/ICML/2003/ICML03-040.pdf

[25] J. R. KING. 1988. Extremely High Concentration Dopant Diffusion in Silicon. *IMA Journal of Applied Mathematics* 40, 3 (1988), 163–181. https://doi.org/10.1093/imamat/40.3.163

[26] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* abs/1609.02907 (2016).

[27] Kyle Kloster and David F Gleich. 2014. Heat kernel based community detection. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, ACM, New York, NY, USA, 1386–1395.

[28] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. 2008. Benchmark graphs for testing community detection algorithms. *Physical review E* 78, 4 (2008), 046110.

[29] LS Leibenzon. 1930. The motion of a gas in a porous medium. Complete Works, Vol. 2, Acad. *Sciences URSS, Moscow,(Russian)* 63 (1930), 8–9.

[30] Jure Leskovec and Andrej Krevl. 2014. {SNAP Datasets}:{Stanford} Large Network Dataset Collection. http://snap.stanford.edu/data.

[31] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. AAAI Press, Palo Alto, CA, USA, 3538–3545.

[32] Yixuan Li, Kun He, David Bindel, and John E. Hopcroft. 2015. Uncovering the Small Community Structure in Large Networks: A Local Spectral Approach. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. International World Wide Web Conferences Steering Committee, Geneva, Switzerland, 658–668. https://doi.org/10.1145/2736277.2741676

[33] Qing Lu and Lise Getoor. 2003. Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. ACM, New York, NY, USA, 496–503.

[34] Ulrike Luxburg. 2007. A tutorial on spectral clustering. *Statistics and Computing* 17, 4 (December 2007), 395–416. https://doi.org/10.1007/s11222-007-9033-z

[35] Michael W. Mahoney, Lorenzo Orecchia, and Nisheeth K. Vishnoi. 2012. A Local Spectral Method for Graphs: With Applications to Improving Graph Partitions and Exploring Data Graphs Locally. *Journal of Machine Learning Research* 13 (August 2012), 2339–2365. http://www.jmlr.org/papers/volume13/mahoney12a/mahoney12a.pdf

[36] Boaz Nadler, Nathan Srebro, and Xueyuan Zhou. 2009. Semi-supervised learning with the graph laplacian: The limit of infinite unlabelled data. *Advances in neural information processing systems* 22 (2009), 1330–1338.

[37] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report 1999-66. Stanford University. http://dbpubs.stanford.edu:8090/pub/1999-66

[38] Jia-Yu Pan, Hyung-Jeong Yang, Christos Faloutsos, and Pinar Duygulu. 2004. Automatic multimedia cross-modal correlation discovery. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, New York, NY, USA, 653–658. https://doi.org/10.1145/1014052.1014135

[39] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, ACM, New York, NY, USA, 701–710.

[40] A. P. Riascos and José L. Mateos. 2014. Fractional dynamics on networks: Emergence of anomalous diffusion and Lévy flights. *Physical Review E* 90, 3 (sep 2014). https://doi.org/10.1103/physreve.90.032809

[41] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *science* 290, 5500 (2000), 2323–2326.

[42] F. Scarselli, M. Gori, Ah Chung Tsoi, M. Hagenbuchner, and G. Monfardini. 2009. The Graph Neural Network Model. *IEEE Transactions on Neural Networks* 20, 1 (jan 2009), 61–80. https://doi.org/10.1109/tnn.2008.2005605

[43] Satu Elisa Schaeffer. 2007. Graph clustering. *Computer Science Review* 1, 1 (2007), 27–64. https://doi.org/10.1016/j.cosrev.2007.05.001

[44] Lawrence F Shampine, Skip Thompson, JA Kierzenka, and GD Byrne. 2005. Nonnegative solutions of ODEs. *Appl. Math. Comput.* 170, 1 (2005), 556–569.

[45] Dane Taylor, Sean A. Myers, Aaron Clauset, Mason A. Porter, and Peter J. Mucha. 2017. Eigenvector-Based Centrality Measures for Temporal Networks. *Multiscale Modeling & Simulation* 15, 1 (jan 2017), 537–574. https://doi.org/10.1137/16m1066142

[46] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *science* 290, 5500 (2000), 2319–2323.

[47] JL Vázquez. 2005. Smoothing and decay estimates for nonlinear parabolic equations of porous medium type. *preprint* (2005).

[48] Juan Luis Vázquez. 2007. Perspectives in nonlinear diffusion: between analysis, physics and geometry. In *Proceedings of the International Congress of Mathematicians Madrid, August 22–30, 2006*. American Mathematical Society, Providence, RI, USA, 609–634.

[49] Juan Luis Vázquez. 2017. The Mathematical Theories of Diffusion: Nonlinear and Fractional Diffusion. In *Nonlocal and Nonlinear Diffusions and Interactions: New Methods and Directions*. Springer International Publishing, New York, NY, USA, 205–278. https://doi.org/10.1007/978-3-319-61494-6_5

[50] Luke N. Veldt, David F. Gleich, and Michael W. Mahoney. 2016. A Simple and Strongly-Local Flow-Based Method for Cut Improvement. In *International Conference on Machine Learning*. ACM, New York, NY, USA, 1938–1947. http://jmlr.org/proceedings/papers/v48/veldt16.html

[51] James Voss, Mikhail Belkin, and Luis Rademacher. 2016. The Hidden Convexity of Spectral Clustering. In *AAAI*. AAAI Press, Palo Alto, CA, USA, 2108–2114. https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12307/11849

[52] Di Wang, Kimon Fountoulakis, Monika Henzinger, Michael W. Mahoney, and Satish Rao. 2017. Capacity Releasing Diffusion for Speed and Locality. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Doina Precup and Yee Whye Teh (Eds.), Vol. 70. PMLR, International Convention Centre, Sydney, Australia, 3598–3607. http://proceedings.mlr.press/v70/wang17b.html

[53] Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. 2012. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*. Springer, New York, NY, USA, 639–655.

[54] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *CoRR* abs/1708.07747 (2017).

[55] Xiaoran Yan, Shang-hua Teng, Kristina Lerman, and Rumi Ghosh. 2016. Capturing the interplay of dynamics and networks through parameterizations of Laplacian operators. *PeerJ Computer Science* 2 (2016), e57.

[56] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In *International Conference on Machine Learning*. ACM, New York, NY, USA, 40–48.

[57] Denny Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *Advances in neural information processing systems*. MIT Press, Cambridge, MA, USA, 321–328.

[58] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*. ACM, New York, NY, USA, 912–919.

Bayesian Exploration with Heterogeneous Agents

Nicole Immorlica
Microsoft Research
nicimm@microsoft.com

Jieming Mao
University of Pennsylvania
maojm517@gmail.com

Aleksandrs Slivkins
Microsoft Research
slivkins@microsoft.com

Zhiwei Steven Wu
University of Minnesota
zsw@umn.edu

ABSTRACT

It is common in recommendation systems that users both consume and produce information as they make strategic choices under uncertainty. While a social planner would balance "exploration" and "exploitation" using a multi-armed bandit algorithm, users' incentives may tilt this balance in favor of exploitation. We consider Bayesian Exploration: a simple model in which the recommendation system (the "principal") controls the information flow to the users (the "agents") and strives to incentivize exploration via information asymmetry. A single round of this model is a version of a well-known "Bayesian Persuasion game" from [24]. We allow heterogeneous users, relaxing a major assumption from prior work that users have the same preferences from one time step to another. The goal is now to learn the best *personalized* recommendations. One particular challenge is that it may be impossible to incentivize some of the user types to take some of the actions, no matter what the principal does or how much time she has. We consider several versions of the model, depending on whether and when the user types are reported to the principal, and design a near-optimal "recommendation policy" for each version. We also investigate how the model choice and the diversity of user types impact the set of actions that can possibly be "explored" by each type.

KEYWORDS

bayesian exploration, incentivizing exploration, heterogeneous agents

ACM Reference Format:
Nicole Immorlica, Jieming Mao, Aleksandrs Slivkins, and Zhiwei Steven Wu. 2019. Bayesian Exploration with Heterogeneous Agents. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313649

1 INTRODUCTION

Recommendation systems are ubiquitous in online markets (*e.g.*,Netflix for movies, Amazon for products, Yelp for restaurants, etc.), high-quality recommendations being a crucial part of their value proposition. A typical recommendation system encourages its users to submit feedback on their experiences, and aggregates this feedback in order to provide better recommendations in the future. Each user plays a dual rule: she consumes information from the previous users (indirectly, via recommendations), and produces new information (*e.g.*,a review) that benefits future users. This dual role creates a tension between exploration, exploitation, and users' incentives.

A social planner – a hypothetical entity that controls users for the sake of common good – would balance "exploration" of insufficiently known alternatives and "exploitation" of the information acquired so far. Designing algorithms to trade off these two objectives is a well-researched subject in machine learning and operations research. However, a given user who decides to "explore" typically suffers all the downside of this decision, whereas the upside (improved recommendations) is spread over many users in the future. Therefore, users' incentives are skewed in favor of exploitation. As a result, observations may be collected at a slower rate, and suffer from selection bias (*e.g.*,ratings of a particular movie may mostly come from people who like this type of movies). Moreover, in some natural but idealized examples (*e.g.*,[30, 33]) optimal recommendations are never found because they are never explored.

Thus, we have a problem of *incentivizing exploration*. Providing monetary incentives can be financially or technologically unfeasible, and relying on voluntary exploration can lead to selection biases. A recent line of work, started by [30], relies on the inherent *information asymmetry* between the recommendation system and a user. These papers posit a simple model, termed *Bayesian Exploration* in [34]. The recommendation system is a "principal" that interacts with a stream of self-interested "agents" arriving one by one. Each agent needs to make a decision: take an action from a given set of alternatives. The principal issues a recommendation, and observes the outcome, but cannot direct the agent to take a particular action. The problem is to design a "recommendation policy" for the principal that learns over time to make good recommendations *and* ensures that the agents are incentivized to follow this recommendation. A single round of this model is a version of a well-known "Bayesian Persuasion game" [24].

Our scope. We study Bayesian Exploration with agents that can have heterogenous preferences. The preferences of an agent are encapsulated in her *type*, *e.g.*,vegan vs meat-lover. When an agent takes a particular action, the outcome depends on the action itself (*e.g.*,the selection of restaurant), the "state" of the world (*e.g.*,the qualities of the restaurants), and the type of the agent. The state is persistent (does not change over time), but initially not known; a Bayesian prior on the state is common knowledge. In each round, the agent type is drawn independently from a fixed and known distribution. The principal strives to learn the best possible recommendation for each agent type.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313649

We consider three models, depending on whether and when the agent type is revealed to the principal: the type is revealed immediately after the agent arrives (*public types*), the type is revealed only after the principal issues a recommendation (*reported types*),[1] and the type is never revealed (*private types*). We design a near-optimal recommendation policy for each modeling choice.

Explorability. A distinctive feature of Bayesian Exploration is that it may be impossible to incentivize some agent types to take some actions, no matter what the principal does or how much time she has. For a more precise terminology, a given type-action pair is *explorable* if this agent type takes this action under some recommendation policy in some round with positive probability. This action is also called *explorable* for this type. Thus: some type-action pairs might not be explorable. Moreover, one may need to explore to find out which pairs are explorable. The set of explorable pairs is interesting in its own right as they bound the welfare of a setting. Recommendation policies cannot do better than the "best explorable action" for a particular agent type: an explorable action with a largest reward in the realized state.

Comparative statics for explorability. We study how the set of all explorable type-action pairs (*explorable set*) is affected by the model choice and the diversity of types. First, we find that for each problem instance the explorable set stays the same if we transition from public types to reported types, and can only become smaller if we transition from reported types to private types. We provide a concrete example when the latter transition makes a huge difference. Second, we vary the distribution $\mathcal{D}$ of agent types. For public types (and therefore also for reported types), we find that the explorable set is determined by the support set of $\mathcal{D}$. Further, if we make the support set larger, then the explorable set can only become larger. In other words, *diversity of agent types helps exploration*. We provide a concrete example when the explorable set increases very substantially even if the support set increases by a single type. However, for private types the picture is quite different: we provide an example when *diversity hurts*, in the same sense as above. Intuitively, with private types, diversity muddles the information available to the principal making it harder to learn about the state of the world, whereas for public types diversity helps the principal refine her belief about the state.

Our techniques. As a warm-up, we first develop a recommendation policy for public types. In the long run, our policy matches the benchmark of "best explorable action". While it is easy to prove that such a policy exists, the challenge is to provide it as an explicit procedure. Our policy focuses on exploring all explorable type-action pairs. Exploration needs to proceed gradually, whereby exploring one action may enable the policy to explore another. In fact, exploring some action for one type may enable the policy to explore some action for another type. Our policy proceeds in phases: in each phase, we explore all actions for each type that can be explored using information available at the start of the phase. Agents of different types learn separately, in per-type "threads"; the threads exchange information after each phase.

[1]Reported types may arise if the principal asks agents to report the type after the recommendation is issued, *e.g.*,in a survey. While the agents are allowed to misreport their respective types, they have no incentives to do that.

An important building block is the analysis of the single-round game. We use information theory to characterize how much state-relevant information the principal has. In particular, we prove a version of *information-monotonicity*: the set of all explorable type-action pairs can only increase if the principal has more information.

As our main contribution, we develop a policy for private types. In this model, recommending one particular action to the current agent is not very meaningful because the agents' type is not known to the principal. Instead, one can recommend a *menu*: a mapping from agent types to actions. Analogous to the case of public types, we focus on explorable menus and gradually explore all such menus, eventually matching the Bayesian-expected reward of the best explorable menu. One difficulty is that exploring a given menu does not immediately reveal the reward of a particular type-action pair (because multiple types could map to the same action). Consequently, even keeping track of what the policy knows is now non-trivial. The analysis of the single-round game becomes more involved, as one needs to argue about "approximate information-monotonicity". To handle these issues, our recommendation policy satisfies only a relaxed version of incentive-compatibility.

In the reported types model, we face a similar issue, but achieve a much stronger result: we design a policy which matches our public-types benchmark in the long run. This may seem counterintuitive because "reported types" are completely useless to the principal in the single-round game (whereas public types are very useful). Essentially, we reduce the problem to the public types case, at the cost of a much longer exploration.

Discussion. This paper, as well as all prior work on incentivizing exploration, relies on very standard yet idealized assumptions of Bayesian rationality and the "power to commit" (i.e., principal can announce a policy and commit to implementing it). A recent paper [23] attempts to mitigate these assumptions (in a setting with homogeneous agents). However, some form of the "power to commit" assumption appears necessary to make any progress.

We do not attempt to *elicit* agents' types when they are not public, in the sense that our recommendation to a given agent is not contingent on anything that this agent reports. However, our result for reported types is already the best possible, in the sense that the explorable set is the same as for public types, so (in the same sense) elicitation is not needed.

Related work. Bayesian Exploration with homogenous agents was introduced in [30], and largely resolved: for optimal policy in the case of two actions and deterministic utilities [30], for explorability [34], and for regret minimization and stochastic utilities [33].

Bayesian Exploration with heterogenous agents and public types is studied in [33], under a very strong assumption which ensures explorability of all type-action pairs, and in [34], where a fixed tuple of agent types arrives in each round and plays a game. [34] focus on explorability of joint actions. Our approach for the public-type case is similar on a high level, but simpler and more efficient, essentially because we focus on type-action pairs rather than joint actions.

A very recent paper [16] (ours is independent work) studies incentivizing exploration with heterogenous agents and private types, but allows monetary transfers. Assuming that each action

is preferred by some agent type, they design an algorithm with a (very) low regret, and conclude that *diversity helps* in their setting.

Several papers study "incentivizing exploration" in substantially different models: with a social network [6]; with time-discounted utilities [10]; with monetary incentives [16, 19]; with a continuous information flow and a continuum of agents [15]; with long-lived agents and "exploration" separate from payoff generation [29, 31, 32]; with fairness [25]. Also, seminal papers [11, 27] study scenarios with long-lived, exploring agents and no principal.

Recommendation policies with no explicit exploration, and closely related "greedy algorithm" in multi-armed bandits, have been studied recently [7, 26, 36, 37]. A common theme is that the greedy algorithm performs well under substantial assumptions on the diversity of types. Yet, it suffers $\Omega(T)$ regret in the worst case.[2]

Exploration-exploitation tradeoff received much attention over the past decades, usually under the rubric of "multi-armed bandits", see books [13, 14, 21]. Absent incentives, Bayesian Exploration with public types is a well-studied problem of "contextual bandits" (with deterministic rewards and a Bayesian prior). A single round of Bayesian Exploration is a version of the Bayesian Persuasion game [24], where the signal observed by the principal is distinct from the state. Exploration-exploitation problems with incentives issues arise in several other scenarios: dynamic pricing, *e.g.*,[5, 9, 28], dynamic auctions [8], advertising auctions [3, 4, 18], human computation [20, 22, 38], and repeated actions, *e.g.*,[1, 2, 12].

2 MODEL AND PRELIMINARIES

Bayesian Exploration is a game between a principal and T agents. The game consists of T rounds. Each round $t \in [T]$ proceeds as follows: a new agent t arrives, receives a message m_t from the principal, chooses an action a_t from a fixed action space $\mathcal{A}$, and collects a reward $r_t \in [0,1]$ that is immediately observed by the principal. Each agent t has a *type* $\theta_t \in \Theta$, drawn independently from a fixed distribution $\mathcal{D}$, and an *action space* $\mathcal{A}$ (same for all agents). There is uncertainty, captured by a "state of nature" $\omega \in \Omega$, henceforth simply the *state*, drawn from a Bayesian prior $\mathcal{P}$ at the beginning of time and fixed across rounds. The *reward* $r_t = u(\theta_t, a_t, \omega) \in [0,1]$ of agent t is determined by its type θ_t, the action $a_t \in \mathcal{A}$ chosen by this agent, and the state ω, for some fixed and deterministic *reward function* $u : \Theta \times \mathcal{A} \times \Omega \to [0,1]$. The principal's messages m_t are generated according to a randomized online algorithm π termed "recommendation policy". Thus, an *instance* of Bayesian Exploration consists of the time horizon T, the sets $\mathcal{A}, \Theta, \Omega$, the type distribution $\mathcal{D}$, the prior $\mathcal{P}$, and the reward function u.

The knowledge structure is as follows. The type distribution $\mathcal{D}$, the Bayesian prior $\mathcal{P}$, the reward function u, and the recommendation policy are common knowledge. Each agent t knows her own type θ_t, and observes nothing else except the message m_t. We consider three model variants, depending on whether and when the principal learns the agent's type: the type is revealed immediately after the agent arrives (*public types*), the type is revealed only after the principal issues a recommendation (*reported types*), the type is not revealed (*private types*).

Let H_t denote the *history* observed by the principal at round t, immediately before it chooses its message m_t. Hence, it equals $\{(r_1, \theta_1), \ldots, (r_{t-1}, \theta_{t-1}), \theta_t\}$ for public types, $\{(r_1, \theta_1), \ldots, (r_{t-1}, \theta_{t-1})\}$ for reported types, and $\{r_1, \ldots, r_{t-1}\}$ for private types.[3] Formally, this is the input to the recommendation policy in each round t. Borrowing terminology from the Bayesian Persuasion literature, we will often refer to the history as the *signal*. We denote the set of all possible histories (signals) at time t by $\mathcal{H}_t$.

The recommendation policy π, the type distribution $\mathcal{D}$, the state distribution $\mathcal{P}$, and the reward function u induce a joint distribution $\mathcal{D}(\Omega, \mathcal{H}_t)$ over states and histories, henceforth called the *signal structure* at round t. Note that it is known to agent t.

We are ready to state agents' decision model. Each agent t, given the realized message m, chooses an action a_t so as to maximize her *Bayesian-expected reward*

$$\mathbb{E}[r_t] \equiv \mathop{\mathbb{E}}_{(\omega, H_t) \sim \mathcal{D}(\Omega, \mathcal{H}_t)} \left[\mathop{\mathbb{E}}_{m_t \sim \pi(H_t)} [u(\theta_t, a_t, \omega) \mid m_t = m] \right].$$

Given the instance of Bayesian Exploration, the goal of the principal is to choose a policy π that maximizes (Bayesian-expected) *total* reward, *i.e.*,$\sum_{t=1}^{T} \mathbb{E}[r_t]$.[4]

We assume that the sets $\mathcal{A}$, Θ and Ω are finite. We use ω_0 as the random variable for the state, and write $\Pr[\omega]$ for $\Pr[\omega_0 = \omega]$. Similarly, we write $\Pr[\theta]$ for $\Pr[\theta_t = \theta]$.

Bayesian-incentive compatibility. For public types, we assume the message m_t in each round is a recommended action $a \in \mathcal{A}$ which, for convenience, we sometimes write as $m_t(\theta_t)$. For private and reported types, we assume that the message m_t in each round is a *menu* mapping types to actions, i.e., $m_t : \Theta \to \mathcal{A}$. We further assume π is Bayesian incentive-compatible.

Definition 2.1. Let $\mathcal{E}_t$ be the event that the agents have followed principal's recommendations before round t, *i.e.*,$a_s = m_s(\theta_s)$ for all rounds $s < t$. The recommendation policy π is *Bayesian incentive compatible* (*BIC*) if for all rounds t and messages m such that

$$\Pr_{(\omega, H_t) \sim \mathcal{D}(\Omega, \mathcal{H}_t)} [m = \pi(H_t) \mid \mathcal{E}_t] > 0,$$

it holds that for all types θ and actions a,

$$\mathbb{E}\left[u(\theta, m(\theta), \omega) - u(\theta, a, \omega) \mid m_t = m, \mathcal{E}_t \right] \geq 0, \qquad (1)$$

where the expectation is over $(\omega, H_t) \sim \mathcal{D}(\Omega, \mathcal{H}_t)$.

The above assumptions are without loss of generality, by a suitable version of Myerson's "revelation principle".

Explorability and benchmarks. For public types, a type-action pair $(\theta, a) \in \Theta \times \mathcal{A}$ is called *eventually-explorable* in state ω if there is some BIC recommendation policy that, for T large enough, eventually recommends this action to this agent type with positive probability. Then action a is called *eventually-explorable* for type θ and state ω. The set of all such actions is denoted $\mathcal{A}_{\omega,\theta}$.

Likewise, for private types, a menu is called *eventually-explorable* in state ω if there is some BIC recommendation policy that eventually recommends this menu with positive probability. The set of all such menus is denoted $\mathcal{M}_\omega$.

[2] This is a well-known folklore result in various settings; *e.g.*,see [35, 37].

[3] For randomized policies, the history also contains policy's random seed in each round.

[4] While the principal must commit to the policy given only the problem instance, the policy itself observes the history and thus can adapt recommendations to inferences about the state based on the history. See Example 3.2.

Our benchmark is the best eventually-explorable recommendation for each type. For public and private types, resp., this is

$$\mathtt{OPT}_{\mathtt{pub}} = \sum_{\theta\in\mathbf{\Theta},\,\omega\in\mathbf{\Omega}} \Pr[\omega]\cdot\Pr[\theta]\cdot \max_{a\in\mathcal{A}_{\omega,\theta}} u(\theta,a,\omega). \tag{2}$$

$$\mathtt{OPT}_{\mathtt{pri}} = \sum_{\omega\in\mathbf{\Omega}} \Pr[\omega]\cdot \max_{m\in\mathcal{M}_\omega} \sum_{\theta\in\mathbf{\Theta}} \Pr[\theta]\cdot u(\theta,m(\theta),\omega). \tag{3}$$

We have $\mathtt{OPT}_{\mathtt{pub}} \geq \mathtt{OPT}_{\mathtt{pri}}$, essentially because any BIC policy for private types can be simulated as a BIC policy for public types. We provide an example (Example 3.2) when $\mathtt{OPT}_{\mathtt{pub}} > \mathtt{OPT}_{\mathtt{pri}}$.

3 COMPARATIVE STATICS

We discuss how the set of all eventually-explorable type-action pairs (*explorable set*) is affected by the model choice and the diversity of types. The explorable set is all information that can possibly be learned in the public-types model. All else equal, settings with larger explorable set have greater or equal total expected reward, both in benchmark (2) and in our approximation guarantees. For private types, the exploration set provides an "upper bound" on the information available to the principal, because the principal does not directly observe the agent types.

Explorability and the model choice. Fix an instance of Bayesian Exploration. Let $\mathcal{A}^{\mathtt{pub}}_\omega$ and $\mathcal{A}^{\mathtt{pri}}_\omega$ be the explorable set for a given state ω, for public and private types, respectively.[5] We will show in Section 4.3 that the explorable set for reported types is $\mathcal{A}^{\mathtt{pub}}_\omega$, too.

Claim 3.1. $\mathcal{A}^{\mathtt{pri}}_\omega \subseteq \mathcal{A}^{\mathtt{pub}}_\omega$.

The idea is that one can simulate any BIC recommendation policy for private types with a BIC recommendation policy for public types; we omit the details.

Interestingly, $\mathcal{A}^{\mathtt{pri}}_\omega$ can in fact be a *strict* subset of $\mathcal{A}^{\mathtt{pub}}_\omega$:

Example 3.2. There are 2 states, 2 types and 2 actions: $\mathbf{\Omega} = \mathbf{\Theta} = \mathcal{A} = \{0,1\}$. States and types are drawn uniformly at random: $\Pr[\omega=0] = \Pr[\theta=0] = \frac{1}{2}$. Rewards are defined as follows:

Table 1: Rewards $u(\theta,a,\omega)$ when $\omega=0$ and $\omega=1$.

	$a=0$	$a=1$
$\theta=0$	$u=3$	$u=4$
$\theta=1$	$u=2$	$u=0$

	$a=0$	$a=1$
$\theta=0$	$u=2$	$u=0$
$\theta=1$	$u=3$	$u=4$

Claim 3.3. *In Example 3.2, $\mathcal{A}^{\mathtt{pri}}_\omega$ is a strict subset of $\mathcal{A}^{\mathtt{pub}}_\omega$.*

Proof. Action 0 is preferred by both types initially. Thus in the first round, the principal must recommend action 0 in order for the policy to be BIC. Hence type-action pairs $\{(0,0),(1,0)\}$ are eventually-explorable in all models.

In the second round, the principal knows the reward of the first-round agent. When types are public or reported, the reward together with the type is sufficient information for the principal to learn the state. Moving forward, the principal can now recommend the higher-reward action for each type (either directly or, in the case of reported types, through a menu). Thus, type-action pair $(0,1)$ is eventually-explorable when $\omega=0$ and, similarly, type-action pair $(1,1)$ is eventually-explorable when $\omega=1$.

For private types, samples from the first-round menu (which, as argued above, must recommend action 0 for both types) do not convey any information about the state, as they have the same distribution in both states. Therefore, action 1 is not eventually-explorable, for either type and either state. □

Explorability and diversity of agent types. Fix an instance of Bayesian Exploration with type distribution $\mathcal{D}$. We consider how the explorable set changes if we modify the type distribution $\mathcal{D}$ in this instance to some other distribution $\mathcal{D}'$. Let $\mathcal{A}_\omega$ and $\mathcal{A}'_\omega$ be the corresponding explorable sets, for each state ω.

For public and reported types, we show that the explorable set is determined by the support set of $\mathcal{D}$, denoted $\mathtt{support}(\mathcal{D})$, and can only increase if the support set increases:

Claim 3.4. *Consider Bayesian Exploration with public types. Then:*
(a) if $\mathtt{support}(\mathcal{D}) = \mathtt{support}(\mathcal{D}')$ then $\mathcal{A}_\omega = \mathcal{A}'_\omega$.
(b) if $\mathtt{support}(\mathcal{D}) \subset \mathtt{support}(\mathcal{D}')$ then $\mathcal{A}_\omega \subseteq \mathcal{A}'_\omega$.

Proof Sketch. Consider public types (the case of reported types then follows by arguments in Section 4.3). Let π be a BIC recommendation policy for the instance with type distribution $\mathcal{D}$ and suppose π eventually explores type-action pairs $\mathcal{A}_\omega$ for this instance and state ω. Consider the instance with type distribution $\mathcal{D}'$. Extend π to a policy π' as follows: let T' be the subsequence of T for which $\mathcal{D}(\theta_t) > 0$. If $t \notin T'$, then recommend the action a that maximizes agent t's Bayesian-expected reward. If $t \in T'$, then consider the sub-history $H \equiv H_t^{T'}$ restricted to T' and recommend action $a \sim \pi(H)$. Then π' is BIC for the instance with type distribution $\mathcal{D}'$. Furthermore, π' eventually explores the same set of type-action pairs $\mathcal{A}_\omega$ for this modified instance as well (and possibly more) as every history that occurs with positive probability in the original instance occurs as a sub-history in the modified instance with positive probability as well. □

For private types, the situation is more complicated. More types can help for some problem instances. For example, if different types have disjoint sets of available actions (more formally: say, disjoint sets of actions with positive rewards) then we are essentially back to the case of reported types, and the conclusions in Claim 3.4 apply. On the other hand, we can use Example 3.2 to show that more types can hurt explorability when types are private. Recall that in this example, for private types only action 0 can be recommended. Now consider a less diverse instance in which only type 0 appears. After one agent in that type chooses action 0, the state is revealed to the principal. For example, when the state $\omega=0$, action 1 can be recommended to future agents. This shows that, in this example, explorable set increases when we have fewer types.

4 PUBLIC TYPES

In this section, we develop our recommendation policy for public types. Throughout, $\mathtt{OPT} = \mathtt{OPT}_{\mathtt{pub}}$.

Theorem 4.1. *Consider an arbitrary instance of Bayesian Exploration with public types. There exists a BIC recommendation policy*

[5] Equivalently, $\mathcal{A}^{\mathtt{pri}}_\omega$ is the set of all type-action pairs $(\theta, m(\theta))$ that appear in some eventually-explorable menu $m \in \mathcal{M}_\omega$ in state ω with private types.

with expected total reward at least $(T - C) \cdot \texttt{OPT}$*, for some constant* C *that depends on the problem instance but not on* T*. This policy explores all type-action pairs that are eventually-explorable for a given state.*

4.1 A single round of Bayesian Exploration

Signal and explorability. We first analyze what actions can be explored by a BIC policy in a single round t of Bayesian Exploration for public types, as a function of the history. Throughout, we suppress θ and t from our notation. Let S be a random variable equal to the history at round t (referred to as a *signal* throughout this section), s be a realization of S, and $\mathcal{S} = \mathcal{D}(\Omega, \mathcal{H})$ be the signal structure: the joint distribution of (ω, S). Note different policies induce different histories and hence different signal structures. Thus it will be important to be explicit about the signal structure throughout this section.

Definition 4.2. Consider a single-round of Bayesian Exploration when the principal receives signal S with signal structure $\mathcal{S}$. An action $a \in \mathcal{A}$ is called *signal-explorable for a realized signal* s if there exists a BIC recommendation policy π such that $\Pr[\pi(s) = a] > 0$. The set of all such actions is denoted as $\texttt{EX}_s[\mathcal{S}]$. The *signal-explorable set*, denoted $\texttt{EX}[\mathcal{S}]$, is the random subset of actions $\texttt{EX}_S[\mathcal{S}]$.

Information-monotonicity. We compare the information content of two signals using the notion of conditional mutual information (see [17] for background). Essentially, we show that a more informative signal leads to the same or larger explorable set.

Definition 4.3. We say that signal S *is at least as informative* as signal S' if $I(S'; \omega \mid S) = 0$.

Intuitively, the condition $I(S'; \omega_0 | S) = 0$ means if one is given random variable S, one can learn no further information from S' about ω_0. Note that this condition depends not only on the signal structures of the two signals, but also on their joint distribution.

Lemma 4.4. *Let* S, S' *be two signals with signal structures* $\mathcal{S}, \mathcal{S}'$*. If* S *is at least as informative as* S'*, then* $\texttt{EX}_{s'}[\mathcal{S}'] \subseteq \texttt{EX}_s[\mathcal{S}]$ *for all* s', s *such that* $\Pr[S = s, S' = s'] > 0$.

Proof. Consider any BIC recommendation policy π' for signal structure $\mathcal{S}'$. We construct π for signal structure $\mathcal{S}$ by setting $\Pr[\pi(s) = a] = \sum_{s'} \Pr[\pi'(s') = a] \cdot Pr[S' = s' \mid S = s]$. Notice that $I(S'; \omega_0 \mid S) = 0$ implies S' and ω_0 are independent given S, i.e $\Pr[S' = s' \mid S = s] \cdot \Pr[\omega_0 = \omega \mid S = s] = \Pr[S' = s', \omega_0 = \omega \mid S = s]$ for all s, s', ω. Therefore, for all s' and ω,

$$\begin{aligned}
&\textstyle\sum_s \Pr[S' = s' \mid S = s] \cdot \Pr[\omega_0 = \omega, S = s] \\
&\quad = \textstyle\sum_s \Pr[S' = s' \mid S = s] \cdot \Pr[\omega_0 = \omega \mid S = s] \cdot \Pr[S = s] \\
&\quad = \textstyle\sum_s \Pr[S' = s', \omega_0 = \omega \mid S = s] \cdot \Pr[S = s] \\
&\quad = \textstyle\sum_s \Pr[S = s, S' = s', \omega_0 = \omega] \\
&\quad = \Pr[\omega_0 = \omega, S' = s'].
\end{aligned}$$

Therefore π' being BIC implies that π is also BIC. Indeed, for any $a, a' \in \mathcal{A}$ and $\theta \in \mathbf{\Theta}$, by plugging in the definition of π,

$$\begin{aligned}
&\textstyle\sum_{\omega, s} \Pr[\omega_0 = \omega, S = s] \cdot (u(\theta, a', \omega) - u(\theta, a, \omega)) \cdot \Pr[\pi(s) = a] \\
&= \textstyle\sum_{\omega, s'} \Pr[\omega_0 = \omega, S' = s'] \cdot (u(\theta, a', \omega) - u(\theta, a, \omega)) \cdot \Pr[\pi'(s') = a] \\
&\geq 0.
\end{aligned}$$

Finally, for any s', s, a such that $Pr[S' = s', S = s] > 0$ and $\Pr[\pi'(s') = a] > 0$, we have $\Pr[\pi(s) = a] > 0$. This implies $\texttt{EX}_{s'}[\mathcal{S}'] \subseteq \texttt{EX}_s[\mathcal{S}]$. □

Max-Support Policy. We can solve the following LP to check whether a particular action $a_0 \in \mathcal{A}$ is signal-explorable given a particular realized signal $s_0 \in \mathcal{X}$. In this LP, we represent a policy π as a set of numbers $x_{a,s} = \Pr[\pi(s) = a]$, for each action $a \in \mathcal{A}$ and each feasible signal $s \in \mathcal{X}$.

maximize x_{a_0, s_0}

subject to:

$$\begin{aligned}
&\textstyle\sum_{\omega \in \Omega, s \in \mathcal{X}} \Pr[\omega] \cdot \Pr[s \mid \omega] \cdot \\
&(u(\theta, a, \omega) - u(\theta, a', \omega)) \cdot x_{a,s} \geq 0 && \forall a, a' \in \mathcal{A} \\
&\textstyle\sum_{a \in \mathcal{A}} x_{a,s} = 1, && \forall s \in \mathcal{X} \\
&x_{a,s} \geq 0, && \forall s \in \mathcal{X}, a \in \mathcal{A}
\end{aligned}$$

Since the constraints in this LP characterize any BIC recommendation policy, it follows that action a_0 is signal-explorable given realized signal s_0 if and only if the LP has a positive solution. If such solution exists, define recommendation policy $\pi = \pi^{a_0, s_0}$ by setting $\Pr[\pi(s) = a] = x_{a,s}$ for all $a \in \mathcal{A}, s \in \mathcal{X}$. Then this is a BIC recommendation policy such that $\Pr[\pi(s_0) = a_0] > 0$.

Definition 4.5. Given a signal structure $\mathcal{S}$, a BIC recommendation policy π is called *max-support* if $\forall s \in \mathcal{X}$ and signal-explorable action $a \in \mathcal{A}$ given s, $\Pr[\pi(s) = a] > 0$.

It is easy to see that we obtain max-support recommendation policy by averaging the $\pi^{a,s}$ policies defined above. Specifically, the following policy is BIC and max-support:

$$\pi^{\max} = \frac{1}{|\mathcal{X}|} \sum_{s \in \mathcal{X}} \frac{1}{|\texttt{EX}_s[\mathcal{S}]|} \sum_{a \in \texttt{EX}_s[\mathcal{S}]} \pi^{a,s}. \tag{4}$$

Maximal Exploration. We design a subroutine `MaxExplore` which outputs a sequence of actions with two properties: it includes every signal-explorable action at least once, and each action in the sequence marginally distributed as $\pi^{\max}$. The length of this sequence, denoted L_θ, should satisfy

$$L_\theta \geq \max_{(a,s) \in \mathcal{A} \times \mathcal{X} \text{ with } \Pr[\pi^{\max}(s) = a] \neq 0} \frac{1}{\Pr[\pi^{\max}(s) = a]}. \tag{5}$$

This step is essentially from [34]; we provide the details below for the sake of completeness. The idea is to put $C_a = L_\theta \cdot \Pr[\pi^{\max}(S) = a]$ copies of each action a into a sequence of length L_θ and randomly permute the sequence. However, C_a might not be an integer, and in particular may be smaller than 1. The latter issue is resolved

by making L_θ sufficiently large. For the former issue, we first put $\lfloor C_a \rfloor$ copies of each action a into the sequence, and then sample the remaining $L_\theta - \sum_a \lfloor C_a \rfloor$ actions according to distribution $p^{\text{res}}(a) = \frac{C_a - \lfloor C_a \rfloor}{L_\theta - \sum_a \lfloor C_a \rfloor}$. For details, see Algorithm 1.

Algorithm 1 Subroutine MaxExplore

Input: type θ, signal S and signal structure $\mathcal{S}$.
Output: a list of actions α
Compute $\pi^{\max}$ as per (4)
Initialize $Res = L_\theta$.
for each action $a \in \mathcal{A}$ **do**
  $C_a \leftarrow L_\theta \cdot \Pr[\pi^{\max}(S) = a]$
  Add $\lfloor C_a \rfloor$ copies of action a into list α.
  $Res \leftarrow Res - \lfloor C_a \rfloor$.
  $p^{\text{res}}(a) \leftarrow C_a - \lfloor C_a \rfloor$
$p^{\text{res}}(a) \leftarrow p^{\text{res}}(a)/Res$, $\forall a \in \mathcal{A}$.
Sample Res many actions from distribution according to p^{res} independently and add these actions into α.
Randomly permute the actions in α.
return α.

Claim 4.6. *Given type θ and signal S, MaxExplore outputs a sequence of L_θ actions. Each action in the sequence marginally distributed as $\pi^{\max}$. For any action a such that $\Pr[\pi^{\max} = a] > 0$, a shows up in the sequence at least once with probability exactly 1. MaxExplore runs in time polynomial in L_θ, $|\mathcal{A}|$, $|\Omega|$ and $|\mathcal{X}|$ (size of the support of the signal).*

4.2 Main Recommendation Policy

Algorithm 2 is the main procedure of our recommendation policy. It consists of two parts: *exploration*, which explores all the eventually-explorable actions, and *exploitation*, which simply recommends the best explored action for a given type. The exploration part proceeds in phases. In each phase l, each type θ gets a sequence of L_θ actions from MaxExplore using the data collected before this phase starts. The phase ends when every agent type θ has finished L_θ rounds. We pick parameter L_θ large enough so that the condition (5) is satisfied for all phases l and all possible signals $S = S_l$. (Note that L_θ is finite because there are only finitely many such signals.) After $|\mathcal{A}| \cdot |\Theta|$ phases, our recommendation policy enters the exploitation part. See Algorithm 2 for details.

There is a separate thread for each type θ, denoted $\texttt{thread}(\theta)$, which is called whenever an agent of this type shows up; see Algorithm 3. In a given phase l, it recommends the L_θ actions computed by MaxExplore, then switches to the best explored action. The thread only uses the information collected before the current phase starts: the signal S_l and signal structure $\mathcal{S}_l$.

The BIC property follows easily from Claim 4.6. The key is that Algorithm 2 explores all eventually-explorable type-action pairs.

The following lemma compares the exploration of Algorithm 2 with l phases and some other BIC recommendation policy with l rounds. Notice that a phase in Algorithm 2 has many rounds.

Lemma 4.7. *Fix phase $l > 0$ and the sequence of agent types $\theta_1, ..., \theta_T$. Assume Algorithm 2 has been running for at least $\min(l, |\mathcal{A}| \cdot |\Theta|)$ phases. For a given state ω, if type-action pair (θ, a) can be explored by some BIC recommendation policy π at round l with positive probability, then such action is explored by Algorithm 2 by the end of phase $\min(l, |\mathcal{A}| \cdot |\Theta|)$ with probability 1.*

Algorithm 2 Main procedure for public types

Initialization: signal $S_1 = \mathcal{S}_1 = \perp$, phase count $l = 1$, index $i_\theta = 0$ for each type $\theta \in \Theta$.
for rounds $t = 1$ to T **do**
  if $l \le |\mathcal{A}| \cdot |\Theta|$ **then**
    {Exploration}
    Call thread $\texttt{thread}(\theta_t)$.
    if every type θ has finished L_θ rounds in the current phase ($i_\theta \ge L_\theta$) **then**
      Start a new phase: $l \leftarrow l + 1$.
      Let S_l be the signal for phase l:
        the set of all observed type-action-reward triples.
      Let $\mathcal{S}_l$ be the signal structure for S_l
        given the realized type sequence $(\theta_1, ..., \theta_t)$.
  else
    {Exploitation}
    Recommend the best explored action for agent type θ_t.

Algorithm 3 Thread for agent type θ: $\texttt{thread}(\theta)$

if this is the first call of $\texttt{thread}(\theta)$ of the current phase **then**
  Compute a list of L_θ actions $\alpha_\theta \leftarrow \text{MaxExplore}(\theta, S_l, \mathcal{S}_l)$.
  Initialize the index of type θ: $i_\theta \leftarrow 0$.
$i_\theta \leftarrow i_\theta + 1$.
if $i_\theta \le L_\theta$ **then**
  Recommend action $\alpha_\theta[i_\theta]$.
else
  Recommend the best explored action of type θ.

Proof. We prove this by induction on l for $l \le |\mathcal{A}| \cdot |\Theta|$. Base case $l = 1$ is trivial by Claim 4.6. Assuming the lemma is correct for $l - 1$, let's prove it's correct for l.

Let $S = S_l$ be the signal of Algorithm 2 by the end of phase $l - 1$. Let S' be the history of π in the first $l - 1$ rounds. More precisely, $S' = (R, H_1, ..., H_{l-1})$, where R is the internal randomness of policy π, and $H_t = (\Theta_t, A_t, u(\Theta_t, A_t, \omega_0))$ is the type-action-reward triple in round t of policy π.

The proof plan is as follows. We first show that $I(S'; \omega_0 | S) = 0$. Informally, this means the information collected in the first $l - 1$ phases of Algorithm 2 contains all the information S' has about the state w_0. After that, we will use the information monotonicity lemma to show that phase l of Algorithm 2 explores all the action-type pairs π might explore in round l.

First of all, we have

$$\begin{aligned} I(S'; \omega_0 \mid S) &= I(R, H_1, ..., H_{l-1}; \omega_0 \mid S) \\ &= I(R; \omega_0 \mid S) + I(H_1, ..., H_{l-1}; \omega_0 \mid S, R) \\ &= I(H_1, ..., H_{l-1}; \omega_0 \mid S, R). \end{aligned}$$

By the chain rule of mutual information, we have

$$I(H_1, ..., H_{l-1}; \omega_0 \mid S, R) = I(H_1; \omega_0 \mid S, R) + \cdots + I(H_{l-1}; \omega_0 \mid S, R, H_1, ..., H_{l-2}).$$

For all $t \in [l-1]$, we have

$$\begin{aligned} &I(H_t; \omega_0 \mid S, R, H_1, ..., H_{t-1}) \\ &\quad = I(\Theta_t, A_t, u(\Theta_t, A_t, \omega_0); \omega_0 \mid S, R, H_1, ..., H_{t-1}) \\ &\quad = I(\Theta_t; \omega_0 \mid S, R, H_1, ..., H_{t-1}) \\ &\qquad + I(A_t, u(\Theta_t, A_t, \omega_0); \omega_0 \mid S, R, H_1, ..., H_{t-1}, \Theta_t) \\ &\quad = I(A_t, u(\Theta_t, A_t, \omega_0); \omega_0 \mid S, R, H_1, ..., H_{t-1}, \Theta_t). \end{aligned}$$

Notice that the suggested action A_t is a deterministic function of randomness of the recommendation policy R, history of previous rounds $H_1, ..., H_{t-1}$ and type in the current round Θ_t. Also notice that, by induction hypothesis, $u(\Theta_t, A_t, \omega_0)$ is a deterministic function of $S, R, H_1, ..., H_{t-1}, \Theta_t, A_t$. Therefore we have

$$I(H_t; \omega_0 \mid S, R, H_1, ..., H_{t-1}) = 0, \qquad \forall t \in [l-1].$$

Then we get $I(S'; \omega_0 \mid S) = 0$.

By Lemma 4.4, we know that $\texttt{EX}[\mathcal{S}'] \subseteq \texttt{EX}[\mathcal{S}]$. For state ω, there exists a signal s' such that $\Pr[S' = s' \mid \omega_0 = \omega] > 0$ and $a \in \texttt{EX}_{s'}[\mathcal{S}']$. Now let s be the realized value of S given $\omega_0 = \omega$, we know that $\Pr[S' = s' \mid S = s] > 0$, so $a \in \texttt{EX}_s[\mathcal{S}]$. By Claim 4.6, we know that at least one agent of type θ in phase l of Algorithm 2 will choose action a.

Now consider the case when $l > |\mathcal{A}| \cdot |\Theta|$. Define ALGto be the variant of Algorithm 2 such that it only does exploration (removing the if-condition and exploitation in Algorithm 2). For $l > |\mathcal{A}| \cdot |\Theta|$, the above induction proof still work for ALG, i.e. for a given state ω, if an action a of type θ can be explored by a BIC recommendation policy π at round l, then such action is guaranteed to be explored by ALGby the end of phase l. Now we are going to argue that ALGwon't explore any new action-type pairs after phase $|\mathcal{A}| \cdot |\Theta|$. Call a phase exploring if in that phase ALGexplores at least one new action-type pair. As there are $|\mathcal{A}| \cdot |\Theta|$ type-action pairs, ALGcan have at most $|\mathcal{A}| \cdot |\Theta|$ exploring phases. On the other hand, once ALGhas a phase that is not exploring, because the signal stays the same after that phase, all phases afterwards are not exploring. So, ALGdoes not have any exploring phases after phase $|\mathcal{A}| \cdot |\Theta|$. For $l > |\mathcal{A}| \cdot |\Theta|$, the first $|\mathcal{A}| \cdot |\Theta|$ phases of Algorithm 2 explores the same set of type-action pairs as the first l phases of ALG. □

Proof of Theorem 4.1. Algorithm 2 is BIC by Claim 4.6. By Lemma 4.7, Algorithm 2 explores all the eventually-explorable type-actions pairs after $|\mathcal{A}| \cdot |\Theta|$ phases. After that, for each agent type θ, Algorithm 2 recommends the best explored action: $\arg\max_{a \in \mathcal{A}_{\omega,\theta}} u(\theta, a, \omega)$ with probability exactly 1.[6]

Therefore Algorithm 2 gets reward OPT except rounds in the first $|\mathcal{A}| \cdot |\Theta|$ phases. It remains to prove that the expected number of rounds in exploration (i.e. first $|\mathcal{A}| \cdot |\Theta|$ phases) does not depend on the time horizon T. Let N_l be the duration of phase l. Recall that the phase ends as soon as each type has shown up at least L_θ times. It follows that $\mathbb{E}[N_l] \le \sum_{\theta \in \Theta} \frac{L_\theta}{\Pr[\theta]}$. So, one can take $C = |\mathcal{A}| \cdot |\Theta| \cdot \sum_{\theta \in \Theta} \frac{L_\theta}{\Pr[\theta]}$. □

[6]This holds with probability exactly 1, provided that our algorithm finishes $|\mathcal{A}| \cdot |\Theta|$ phases. If some undesirable low-probability event happens, *e.g.*,if all agents seen so far have had the same type, our algorithm would never finish $|\mathcal{A}| \cdot |\Theta|$ phases.

4.3 Extension to Reported Types

We sketch how to extend our ideas for public types to handle the case of reported types. We'd like to simulate the recommendation policy for public types, call it π_{pub}. We simulate it separately for the exploration part and the exploitation part. The exploitation part is fairly easy: we provide a menu that recommends the best explored action for each agent types.

In the exploration part, in each round t we guess the agent type to be $\hat{\theta}_t$, with equal probability among all types.[7] The idea is to simulate π_{pub} only in *lucky rounds* when we guess correctly, *i.e.*,$\hat{\theta}_t = \theta_t$. Thus, in each round t we simulate the l_t-th round of π_{pub}, where l_t is the number of lucky rounds before round t. In each round t of exploration, we suggest the following menu. For type $\hat{\theta}_t$, we recommend the same action as π_{pub} would recommend for this type in the l_t-th round, namely $\hat{a}_t = \pi_{\text{pub}}^{l_t}(\hat{\theta}_t)$. For any other type, we recommend the action which has the best expected reward given the "common knowledge" (information available before round 1) and the action $\hat{a}_t$. This is to ensure that in a lucky round, the menu does not convey any information beyond action $\hat{a}_t$. When we receive the reported type, we can check whether our guess was correct. If so, we input the type-action-reward triple back to π_{pub}. Else, we ignore this round, as if it never happened.

Thus, our recommendation policy eventually explores the same type-action pairs as π_{pub}. The expected number of rounds increases by the factor of $|\Theta|$. Thus, we have the following theorem.

Theorem 4.8. *Consider Bayesian Exploration with reported types. There exists a BIC recommendation policy whose expected total reward is at least* $(T - C) \cdot \texttt{OPT}_{\text{pub}}$*, for some constant* C *that depends on the problem instance but not on* T*. This policy explores all type-action pairs that are eventually-explorable for public types.*

5 PRIVATE TYPES

Our recommendation policy for private types satisfies a relaxed version of the BIC property, called δ*-BIC*, where the right-hand side in (1) is $-\delta$ for some fixed $\delta > 0$. We assume a more permissive behavioral model in which agents obey such policy.

The main result is as follows. (Throughout, $\texttt{OPT} = \texttt{OPT}_{\text{pri}}$.)

Theorem 5.1. *Consider Bayesian Exploration with private types, and fix* $\delta > 0$*. There exists a* δ*-BIC recommendation policy with expected total reward at least* $(T - C \log T) \cdot \texttt{OPT}$*, where* C *depends on the problem instance but not on time horizon* T*.*

The recommendation policy proceeds in phases: in each phase, it explores all menus that can be explored given the information collected so far. The crucial step in the proof is to show that:

(P1) the first l phases of our recommendation policy explore all the menus that could be possibly explored by the first l rounds of any BIC recommendation policy.

[7]We guess the types uniformly, rather than according to their probabilities, because our goal is to explore each type for certain number of rounds. Guessing a type according to its probability will only make rare types appear even rarer.

The new difficulty for private types comes from the fact that we are exploring menus instead of type-actions pairs, and we do not learn the reward of a particular type-action pair immediately. This is because a recommended menu may map several different types to the chosen action, so knowing the latter does not immediately reveal the agent's type. Moreover, the full "outcome" of a particular menu is a distribution over action-reward pairs, it is, in general, impossible to learn this outcome exactly in any finite number of rounds. Because of these issues, we cannot obtain Property (P1) exactly. Instead, we achieve an approximate version of this property, as long as we explore each menu enough times in each phase.

We then show that this approximate version of (P1) suffices to guarantee explorability, if we relax the incentives property of our policy from BIC to δ-BIC, for any fixed $\delta > 0$. In particular, we prove an approximate version of the information-monotonicity lemma (Lemma 4.4) which (given the approximate version of (P1)) ensures that our recommendation policy can explore all the menus that could be possibly explored by the first l rounds of any BIC recommendation policy.

5.1 A Single round of Bayesian Exploration

Recall that for a random variable S, called *signal*, the signal structure is a joint distribution of (ω, S).

Definition 5.2. Consider a single-round of Bayesian Exploration when the principal has signal S from signal structure $\mathcal{S}$. For any $\delta \geq 0$, a menu $m \in \mathcal{M}$ is called δ-signal-explorable, for a given signal s, if there exists a single-round δ-BIC recommendation policy π such that $\Pr[\pi(s) = m] > 0$. The set of all such menus is denoted as $\mathtt{EX}_s^\delta[\mathcal{S}]$. The δ-signal-explorable set is defined as $\mathtt{EX}^\delta[\mathcal{S}] = \mathtt{EX}_S^\delta[\mathcal{S}]$. We omit δ in $\mathtt{EX}^\delta[\mathcal{S}]$ when $\delta = 0$.

Approximate Information Monotonicity. In the following definition, we define a way to compare two signals approximately.

Definition 5.3. Let S and S' be two random variables. We say random variable S is α-approximately informative as random variable S' about state ω_0 if $I(S'; \omega_0 | S) = \alpha$.

Lemma 5.4. *Let S and S' be two random variables and $\mathcal{S}$ and $\mathcal{S}'$ be their signal structures. If S is $(\delta^2/8)$-approximately informative as S' about state ω_0 (i.e. $I(S'; \omega_0|S) \leq \delta^2/8$), then $\mathtt{EX}_{s'}[\mathcal{S}'] \subseteq \mathtt{EX}_s^\delta[\mathcal{S}]$ for all s', s such that $\Pr[S = s, S' = s'] > 0$.*

Proof. For each signal realization s, denote

$$D_s = \mathbf{D}_{\mathsf{KL}}\left(((S', \omega_0) \mid S = s) \quad \| \quad (S'|S = s) \times (\omega_0 \mid S = s) \right).$$

We have $\sum_s \Pr[S = s] \cdot D = I(S'; \omega_0|S) \leq \delta^2/8$.

By Pinsker's inequality, we have

$$\begin{aligned}
&\sum_s \Pr[S = s] \cdot \sum_{s', \omega} |\Pr[S' = s', \omega_0 = \omega | S = s] \\
&\qquad - \Pr[S' = s'|S = s] \cdot \Pr[\omega_0 = \omega | S = s]| \\
&\leq \textstyle\sum_s \Pr[S = s] \cdot \sqrt{2 \ln(2) \cdot D_s} \\
&\leq \sqrt{2 \textstyle\sum_s \Pr[S = s] \cdot D_s} \leq \delta/2.
\end{aligned}$$

Consider any BIC recommendation policy π' for signal structure $\mathcal{S}'$. We construct π for signature structure $\mathcal{S}$ by setting

$$\Pr[\pi(s) = m] = \textstyle\sum_{s'} \Pr[\pi'(s') = m] \cdot Pr[S' = s'|S = s].$$

Now we check π is δ-BIC. For any $m, m' \in \mathcal{M}$ and $\theta \in \mathbf{\Theta}$,

$$\begin{aligned}
&\textstyle\sum_{\omega, s} \Pr[\omega_0 = \omega] \cdot \Pr[S = s|\omega_0 = \omega] \\
&\qquad \cdot (u(\theta, m(\theta), \omega) - u(\theta, m'(\theta), \omega)) \cdot \Pr[\pi(s) = m] \\
&= \textstyle\sum_{\omega, s, s'} \Pr[\omega_0 = \omega, S = s] \cdot \Pr[S' = s'|S = s] \cdot \Pr[\pi'(s') = m] \\
&\qquad \cdot (u(\theta, m(\theta), \omega) - u(\theta, m'(\theta), \omega)) \\
&\geq \textstyle\sum_{\omega, s, s'} \Pr[\omega_0 = \omega, S = s, S' = s'] \cdot \Pr[\pi'(s') = m] \\
&\qquad \cdot (u(\theta, m(\theta), \omega) - u(\theta, m'(\theta), \omega)) \\
&\qquad - 2 \cdot \sum_{\omega, s, s'} |\Pr[\omega_0 = \omega, S = s] \cdot \Pr[S' = s'|S = s] \\
&\qquad - \Pr[\omega_0 = \omega, S = s, S' = s']| \\
&= \sum_{\omega, s'} \Pr[\omega_0 = \omega, S' = s'] \cdot \Pr[\pi'(s') = m] \\
&\qquad \cdot (u(\theta, m(\theta), \omega) - u(\theta, m'(\theta), \omega)) \\
&\qquad - 2 \cdot \sum_s \Pr[S = s] \cdot \sum_{s', \omega} |\Pr[S' = s', \omega_0 = \omega | S = s] \\
&\qquad - \Pr[S' = s'|S = s] \cdot \Pr[\omega_0 = \omega | S = s]| \\
&\geq 0 - 2 \cdot t \tfrac{\delta}{2} = -\delta.
\end{aligned}$$

We also have for any s', s, m such that $\Pr[S' = s', S = s] > 0$ and $\Pr[\pi'(s') = m] > 0$, we have $\Pr[\pi(s) = m] > 0$. This implies $\mathtt{EX}_{s'}[\mathcal{S}'] \subseteq \mathtt{EX}_s^\delta[\mathcal{S}]$. □

Max-Support Policy. We can solve the following LP to check whether a particular menu $m_0 \in \mathcal{A}$ is signal-explorable given a particular realized signal $s_0 \in \mathcal{X}$. In this LP, we represent a policy π as a set of numbers $x_{m,s} = \Pr[\pi(s) = m]$, for each menu $m \in \mathcal{M}$ and each feasible signal $s \in \mathcal{X}$.

maximize x_{m_0, s_0}

subject to:

$$\begin{aligned}
&\sum_{\omega \in \Omega, s \in \mathcal{X}} \Pr[\omega] \cdot \Pr[s|\omega] \cdot (u(\theta, m(\theta), \omega) - u(\theta, m'(\theta), \omega) + \delta) \\
&\qquad \cdot x_{m,s'} \geq 0 \qquad \forall m, m' \in \mathcal{M}, \theta \in \mathbf{\Theta} \\
&\textstyle\sum_{m \in \mathcal{M}} x_{m,s} = 1, \qquad \forall s \in \mathcal{X} \\
&x_{m,s} \geq 0, \qquad \forall s \in \mathcal{X}, m \in \mathcal{M}
\end{aligned}$$

Since the constraints in this LP characterize any δ-BIC recommendation policy, it follows that menu m_0 is δ-signal-explorable given realized signal s_0 if and only if the LP has a positive solution. If such solution exists, define recommendation policy $\pi = \pi^{m_0, s_0}$ by setting $\Pr[\pi(s) = m] = x_{m,s}$ for all $m \in \mathcal{M}, s \in \mathcal{X}$. Then this is a δ-BIC recommendation policy such that $\Pr[\pi(s_0) = m_0] > 0$.

Definition 5.5. Given a signal structure $\mathcal{S}$, a recommendation policy π is called the δ-max-support policy if $\forall s \in \mathcal{X}$ and δ-signal-explorable menu $m \in \mathcal{M}$ given s, $\Pr[\pi(s) = m] > 0$.

It is easy to see that we obtain δ-max-support recommendation policy by averaging the $\pi^{m,s}$ policies define above. Specifically, the following policy is a δ-BIC and δ-max-support policy.

$$\pi^{max} = \frac{1}{|\mathcal{X}|} \sum_{s \in \mathcal{X}} \frac{1}{|\mathsf{EX}_s^\delta[\mathcal{S}]|} \sum_{m \in \mathsf{EX}_s^\delta[\mathcal{S}]} \pi^{m,s}. \tag{6}$$

Maximal Exploration. Let us design a subroutine, called MaxExplore, which outputs a sequence of L menus. We are going to assume $L \geq \max_{m,s} \frac{B_m(\gamma_0)}{\Pr[\pi^{max}(s)=m]} \cdot \gamma_0$ is defined in Algorithm 5 of Section 5.2. B_m is defined in Lemma 5.7.

The goal of this subroutine MaxExplore is to make sure that for any signal-explorable menu m, m shows up at least $B_m(\gamma_0)$ times in the sequence with probability exactly 1. On the other hand, we want that the menu of each specific location in the sequence has marginal distribution same as π^{max}.

Algorithm 4 Subroutine MaxExplore

Input: signal S, signal structure $\mathcal{S}$.
Output: a list of menus μ
Compute π^{max} as per (6).
Initialize $Res = L$.
for each menu $m \in \mathcal{M}$ **do**
  $C_m \leftarrow L \cdot \Pr[\pi^{max}(S) = m]$.
  Add $\lfloor C_m \rfloor$ copies of menu m into list μ.
  $Res \leftarrow Res - \lfloor C_m \rfloor$.
  $p^{Res}(m) \leftarrow C_m - \lfloor C_m \rfloor$
$p^{Res}(m) \leftarrow p^{Res}(m)/Res, \forall m \in \mathcal{M}$.
Sample Res many menus from distribution according to p^{Res} independently and add these menus into μ.
Randomly permute the menus in μ.
return μ.

Similarly as the MaxExplore in Section 4, we have the following:

Claim 5.6. *Given realized signal S, MaxExplore outputs a sequence of L menus. Each menu in the sequence marginally distributed as π^{max}. For any menu m such that $\Pr[\pi^{max} = m] > 0$, m shows up in the sequence at least $B_m(\gamma_0)$ times with probability exactly 1. MaxExplore runs in time polynomial in L, $|\mathcal{M}|$, $|\Omega|$, $|\mathcal{X}|$ (size of the support of the signal).*

Menu Exploration. If an agent in a given round follows a given menu m, an action-reward pair is revealed to the algorithm after the round. Such action-reward pair is called a *sample* of the menu m. Let $D_m(\omega)$ denote the distribution of this action-reward pair for a fixed state ω (with randomness coming from the agent arrivals).

We compute an estimate Δ_m of $D_m(\omega_0)$. This estimate is a *triple-list*: an explicit list of (action, reward, positive probability) triples.

Lemma 5.7. *For any $\gamma > 0$, we can compute a triple-list Δ_m which is a function of $B_m(\gamma) = O(\ln 1/\gamma)$ samples of menu m such that*

$$\forall \omega \in \Omega \quad \Pr[\Delta_m \neq D_m(\omega) \mid \omega_0 = \omega] \leq \gamma.$$

Proof. Let U be the union of the support of $D_m(\omega)$ for all $\omega \in \Omega$. For each $u \in U$ (u is just a sample of the menu), define

$$q(u, \omega) = \Pr_{v \sim D_m(\omega)}[v = u].$$

Let δ_m be small enough such that for all ω, ω' with $D_m(\omega) \neq D_m(\omega')$, there exists $u \in U$, such that $|q(u, \omega) - q(u, \omega')| > \delta_m$.

Now we compute Δ_m as follows: Take $B_m(\gamma) = \frac{2}{\delta_m^2} \ln\left(\frac{2|U|}{\gamma}\right)$ samples and set $\hat{q}(u)$ as the empirical frequency of seeing u. And set Δ_m to be some $D_m(\omega)$ such that for all $u \in U$, $|q(u, \omega) - \hat{q}(u)| \leq \delta_m/2$. Notice that if such ω exists, Δ_m will be unique. If no ω satisfies this, just pick Δ_m to be an arbitrary $D_m(\omega)$.

Now let's analyze $\Pr[\Delta_m \neq D_m(\omega)]$. Let's fix the state $\omega_0 = \omega$. By Chernoff bound, for each $u \in U$,

$$\Pr[|q(u, \omega) - \hat{q}(u)| > \delta_m/2] \leq 2 \exp\left(-2 \cdot (\delta_m/2)^2 \cdot B_m(\gamma)\right) \leq \gamma/|U|.$$

By union bound, with probability at least $1 - \gamma$, we have for all $u \in U$, $|q(u, \omega) - \hat{q}(u)| \leq \delta_m/2$. This implies $\Delta_m = D_m(\omega)$. □

5.2 Main Recommendation Policy

In this subsection, we develop our main recommendation policy, Algorithm 5 (see pseudo-code), which explores all the eventually-explorable menus and then recommends the agents the best menu given all history. We pick L to be at least

$$\max_{m, s: \Pr[\pi(s)=m]>0} \frac{B_m(\gamma_0)}{\Pr[\pi(s) = m]}$$

for all π that might be chosen as π^{max} by Algorithm 5.

It is easy to check by Claim 5.6 that for each agent, it is δ-BIC to follow the recommended action if previous agents all follow the recommended actions. Therefore we have the following claim.

Claim 5.8. *Algorithm 5 is δ-BIC.*

Lemma 5.9. *For any $l > 0$, assume Algorithm 5 has at least $\min(l, |\mathcal{M}|)$ phases. For a given state ω, if a menu m can be explored by a BIC recommendation policy π at round l (i.e. $\Pr[\pi^l = m] > 0$), then such menu is guaranteed to be explored B_m times by Algorithm 5 by the end of phase $\min(l, |\mathcal{M}|)$.*

Proof. The proof is similar to Lemma 4.7. We prove by induction on l for $l \leq |\mathcal{M}|$.

Let S be the signal of Algorithm 5 in phase l. Let S' be the history of π in the first $l - 1$ rounds. More precisely, $S' = R, H_1, ..., H_{l-1}$. Here R is the internal randomness of π and

$$H_t = (M_t, A_t, u(\Theta_t, M_t(\Theta_t), \omega_0))$$

is the menu and the action-reward pair in round t of π.

Let $\mathcal{M}'$ to be the set of menus explored in the first $l - 1$ phases of Algorithm 5. By the induction hypothesis, we have $\forall t \in [l - 1]$, $M_t \subseteq \mathcal{M}'$. Then:

$$\begin{aligned} &I(S'; \omega_0 | S) = I(R, H_1, ..., H_{l-1}; \omega_0 | S) \\ =&I(R; \omega_0 | S) + I(H_1, ..., H_{l-1}; \omega_0 | S, R) = I(H_1, ..., H_{l-1}; \omega_0 | S, R). \end{aligned}$$

By the chain rule of mutual information, we have

$$\begin{aligned} &I(H_1, ..., H_{l-1}; \omega_0 | S, R) \\ =&I(H_1; \omega_0 | S, R) + I(H_2; \omega_0 | S, R, H_1) + \cdots + I(H_{l-1}; \omega_0 | S, R, H_1, ..., H_{l-2}). \end{aligned}$$

Algorithm 5 Main procedure for private types

Initialize: signal $S_1 = \mathcal{S}_1 = \perp$, phase $l = 1$.
{ S_l and $\mathcal{S}_l$ are the signal and signal structure in phase l. }
Set $\gamma_1 = \min\left(\frac{\delta^2}{16|\mathcal{M}|\log(|\Omega|)}, \left(\frac{\delta^2}{32|M|}\right)^2\right)$ and $\gamma_2 = \frac{1}{T|\mathcal{M}|}$.
Set $\gamma_0 = \min(\gamma_1, \gamma_2)$.
for rounds $t = 1$ to T **do**
  if phase $l \le |\mathcal{M}|$ **then**
    {**Exploration**}
    if $t \equiv 1 \pmod{L}$ **then**
      Start a new phase:
      $\mu \leftarrow \text{MaxExplore}(S_l, \mathcal{S}_l)$ {compute a list of L menus}
    Suggest menu $\mu[(t-1) \mod L + 1]$ to the agent.
    if $t \equiv 0 \pmod{L}$ **then**
      End of a phase:
      for each explored menu m in the previous phase **do**
        use $B_m(\gamma_1)$ samples to compute Δ_m from Lemma 5.7
      If no state $\omega \in \Omega$ is consistent with Δ_m (*i.e.,*$\Delta_m = D_m(\omega)$) for all explored menus m **then**
        pick any state ω, and
        set $\Delta_m \leftarrow D_m(\omega)$ for all explored menus m.
      {to ensure that #signals is bounded by $|\Omega|$.}
      $l \leftarrow l + 1$.
      Set $S_l = \{ \Delta_m$: all explored menus $m \}$.
      Set $\mathcal{S}_l$ to be the signal structure of S_l.
  else
    {**Exploitation**}
    if this is the first exploitation round **then**
      for each menu m explored during exploration **do**
        use $B_m(\gamma_2)$ samples to compute Δ_m from Lemma 5.7.
      Set $S_l = \{ \Delta_m$: all explored menus $m \}$.
    Suggest the menu which consists of the best action of each type conditioned on S_l and the prior.

For all $t \in [l-1]$, we have

$$\begin{aligned}
&I(H_t; \omega_0 \mid S, R, H_1, ..., H_{t-1}) \\
=&I(M_t, A_t, u(\Theta_t, M_t(\Theta_t), \omega_0); \omega_0 \mid S, R, H_1, ..., H_{t-1}) \\
=&I(A_t, u(\Theta_t, M_t(\Theta_t), \omega_0); \omega_0 \mid S, R, H_1, ..., H_{t-1}, M_t) \\
\le&I(D_{M_t}; \omega_0 \mid S, R, H_1, ..., H_{t-1}, M_t).
\end{aligned}$$

The second last step comes from the fact that M_t is a deterministic function of $R, H_1, ..., H_{t-1}$. The last step comes from the fact that $(A_t, u(\Theta_t, M_t(\Theta_t), \omega_0))$ is independent with ω_0 given D_{M_t}.

Then we have

$$\begin{aligned}
&I(D_{M_t}; \omega_0 \mid S, R, H_1, ..., H_{t-1}, M_t) \\
=&\textstyle\sum_{m \in \mathcal{M}'} \Pr[M_t = m] \cdot I(D_m; \omega_0 \mid S, R, H_1, ..., H_{t-1}, M_t = m) \\
\le&\textstyle\sum_{m \in M'} \Pr[M_t = m] \cdot I(D_m; \omega_0 \mid \Delta_m, M_t = m). \\
\le&\textstyle\sum_{m \in M'} \Pr[M_t = m] \cdot H(D_m \mid \Delta_m, M_t = m).
\end{aligned}$$

The last step comes from the fact that

$$I(D_m; (S \backslash \Delta_m), R, H_1, ..., H_{t-1} \mid \omega_0, \Delta_m, M_t = m) = 0.$$

By Lemma 5.7, we know that $\Pr[D_m \ne \Delta_m \mid M_t = m] \le \gamma_1$. By Fano's inequality, we have

$$\begin{aligned}
&H(D_m \mid \Delta_m, M_t = m) \le H(\gamma_1) + \gamma_1 \log(|\Omega| - 1) \\
&\le 2\sqrt{\gamma_1} + \gamma_1 \log(|\Omega| - 1) \le \frac{\delta^2}{16|\mathcal{M}|} + \frac{\delta^2}{16|\mathcal{M}|} = \frac{\delta^2}{8|\mathcal{M}|}.
\end{aligned}$$

Therefore we have

$$I(H_t; \omega_0 \mid S, R, H_1, ..., H_{t-1}) \le \frac{\delta^2}{8|\mathcal{M}|}, \forall t \in [l-1].$$

Then we get $I(S'; \omega_0 \mid S) \le \delta^2/8$.

By Lemma 5.4, we know that $\mathtt{EX}_{s'}[\mathcal{S}'] \subseteq \mathtt{EX}_s^{\delta}[\mathcal{S}]$. By Claim 5.6, we know that phase l will explore menu m at least $B_m(\gamma_0)$ times.

When $l > |\mathcal{M}|$, the proof follows from the same argument as the last paragraph of the proof of Lemma 4.7. □

Proof of Theorem 5.1. By Claim 5.8, Algorithm 5 is δ-BIC.

By Lemma 5.9, for each state ω, Algorithm 5 explores all the eventually-explorable menus (i.e. $\mathcal{M}_\omega$) by the end of $|\mathcal{M}|$ phases.

After that, by Lemma 5.7 and $\gamma_2 = \frac{1}{T|\mathcal{M}|}$, for a fixed state ω, we know that with probability $1 - 1/T$, $\delta_m = D_m$ for all $m \in \mathcal{M}_\omega$. In this case, the agent of type θ gets expected reward at least $u(\theta, m^*(\theta), \omega)$ where menu $m^* = \arg\max_{m \in \mathcal{M}_\omega} \sum_{\theta \in \Theta} \Pr[\theta] \cdot u(\theta, m(\theta), \omega)$. Taking average over types, the expected reward per round should be at least $(1 - 1/T) \cdot \max_{m \in \mathcal{M}_\omega} \sum_{\theta \in \Theta} \Pr[\theta] \cdot u(\theta, m(\theta), \omega)$.

The expected number of rounds of the first $|\mathcal{M}|$ phases is $|\mathcal{M}| \cdot L = O(\ln(T))$. Therefore, Algorithm 5 has expected total reward at least $T \cdot \mathtt{OPT} - T \cdot (1/T) - O(\ln(T)) = T \cdot \mathtt{OPT} - O(\ln(T))$. □

REFERENCES

[1] Kareem Amin, Afshin Rostamizadeh, and Umar Syed. 2013. Learning Prices for Repeated Auctions with Strategic Buyers. In *26th Advances in Neural Information Processing Systems (NIPS)*. 1169–1177.

[2] Kareem Amin, Afshin Rostamizadeh, and Umar Syed. 2014. Repeated Contextual Auctions with Strategic Buyers. In *27th Advances in Neural Information Processing Systems (NIPS)*. 622–630.

[3] Moshe Babaioff, Robert Kleinberg, and Aleksandrs Slivkins. 2015. Truthful Mechanisms with Implicit Payment Computation. *J. of the ACM* 62, 2 (2015), 10. Subsumes the conference papers in *ACM EC 2010* and *ACM EC 2013*.

[4] Moshe Babaioff, Yogeshwer Sharma, and Aleksandrs Slivkins. 2014. Characterizing Truthful Multi-armed Bandit Mechanisms. *SIAM J. on Computing (SICOMP)* 43, 1 (2014), 194–230. Preliminary version in *10th ACM EC*, 2009.

[5] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. 2018. Bandits with Knapsacks. *J. of the ACM* 65, 3 (2018). Preliminary version in *FOCS 2013*.

[6] Gal Bahar, Rann Smorodinsky, and Moshe Tennenholtz. 2016. Economic Recommendation Systems. In *16th ACM Conf. on Electronic Commerce (EC)*.

[7] Hamsa Bastani, Mohsen Bayati, and Khashayar Khosravi. 2018. Mostly Exploration-Free Algorithms for Contextual Bandits. *CoRR arXiv:1704.09011* (2018). Working paper.

[8] Dirk Bergemann and Maher Said. 2011. Dynamic Auctions: A Survey. In *Wiley Encyclopedia of Operations Research and Management Science*. John Wiley & Sons.

[9] Omar Besbes and Assaf Zeevi. 2009. Dynamic Pricing Without Knowing the Demand Function: Risk Bounds and Near-Optimal Algorithms. *Operations Research* 57 (2009), 1407–1420. Issue 6.

[10] Kostas Bimpikis, Yiangos Papanastasiou, and Nicos Savva. 2018. Crowdsourcing Exploration. *Management Science* 64 (2018), 1477–1973. Issue 4.

[11] Patrick Bolton and Christopher Harris. 1999. Strategic Experimentation. *Econometrica* 67, 2 (1999), 349–374.

[12] Mark Braverman, Jieming Mao, Jon Schneider, and Matt Weinberg. 2018. Selling to a No-Regret Buyer. In *ACM Conf. on Economics and Computation (ACM EC)*. 523–538.

[13] Sébastien Bubeck and Nicolo Cesa-Bianchi. 2012. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends in Machine Learning* 5, 1 (2012).

[14] Nicolò Cesa-Bianchi and Gábor Lugosi. 2006. *Prediction, learning, and games*. Cambridge Univ. Press.

[15] Yeon-Koo Che and Johannes Hörner. 2018. Optimal design for social learning. *Quarterly Journal of Economics* (2018). Forthcoming. First published draft: 2013.

[16] Bangrui Chen, Peter I. Frazier, and David Kempe. 2018. Incentivizing Exploration by Heterogeneous Users. In *Conf. on Learning Theory (COLT)*. 798–818.

[17] I. Csiszar and J. Körner. 2011. *Information theory: coding theorems for discrete memoryless systems*. Cambridge University Press.

[18] Nikhil Devanur and Sham M. Kakade. 2009. The Price of Truthfulness for Pay-Per-Click Auctions. In *10th ACM Conf. on Electronic Commerce (EC)*. 99–106.

[19] Peter Frazier, David Kempe, Jon M. Kleinberg, and Robert Kleinberg. 2014. Incentivizing exploration. In *ACM Conf. on Economics and Computation (ACM EC)*. 5–22.

[20] Arpita Ghosh and Patrick Hummel. 2013. Learning and incentives in user-generated content: multi-armed bandits with endogenous arms. In *Innovations in Theoretical Computer Science Conf. (ITCS)*. 233–246.

[21] John Gittins, Kevin Glazebrook, and Richard Weber. 2011. *Multi-Armed Bandit Allocation Indices*. John Wiley & Sons.

[22] Chien-Ju Ho, Aleksandrs Slivkins, and Jennifer Wortman Vaughan. 2016. Adaptive Contract Design for Crowdsourcing Markets: Bandit Algorithms for Repeated Principal-Agent Problems. *J. of Artificial Intelligence Research* 55 (2016), 317–359. Preliminary version appeared in *ACM EC 2014*.

[23] Nicole Immorlica, Jieming Mao, Aleksandrs Slivkins, and Steven Wu. 2018. Incentivizing Exploration with Unbiased History. Under submission, available at `https://arxiv.org/abs/1811.06026`.

[24] Emir Kamenica and Matthew Gentzkow. 2011. Bayesian Persuasion. *American Economic Review* 101, 6 (2011), 2590–2615.

[25] Sampath Kannan, Michael J. Kearns, Jamie Morgenstern, Mallesh M. Pai, Aaron Roth, Rakesh V. Vohra, and Zhiwei Steven Wu. 2017. Fairness Incentives for Myopic Agents. In *ACM Conf. on Economics and Computation (ACM EC)*. 369–386.

[26] Sampath Kannan, Jamie Morgenstern, Aaron Roth, Bo Waggoner, and Zhiwei Steven Wu. 2018. A Smoothed Analysis of the Greedy Algorithm for the Linear Contextual Bandit Problem. In *Advances in Neural Information Processing Systems (NIPS)*.

[27] Godfrey Keller, Sven Rady, and Martin Cripps. 2005. Strategic Experimentation with Exponential Bandits. *Econometrica* 73, 1 (2005), 39–68.

[28] Robert D. Kleinberg and Frank T. Leighton. 2003. The Value of Knowing a Demand Curve: Bounds on Regret for Online Posted-Price Auctions. In *IEEE Symp. on Foundations of Computer Science (FOCS)*.

[29] Robert D. Kleinberg, Bo Waggoner, and E. Glen Weyl. 2016. Descending Price Optimally Coordinates Search. Working paper. Preliminary version in *ACM EC 2016*.

[30] Ilan Kremer, Yishay Mansour, and Motty Perry. 2014. Implementing the âĂIJWisdom of the CrowdâĂİ. *J. of Political Economy* 122 (2014), 988–1012. Issue 5. Preliminary version in *ACM EC 2014*.

[31] Annie Liang and Xiaosheng Mu. 2018. Overabundant Information and Learning Traps. In *ACM Conf. on Economics and Computation (ACM EC)*. 71–72.

[32] Annie Liang, Xiaosheng Mu, and Vasilis Syrgkanis. 2018. Optimal and Myopic Information Acquisition. In *ACM Conf. on Economics and Computation (ACM EC)*. 45–46.

[33] Yishay Mansour, Aleksandrs Slivkins, and Vasilis Syrgkanis. 2015. Bayesian Incentive-Compatible Bandit Exploration. In *15th ACM Conf. on Economics and Computation (ACM EC)*.

[34] Yishay Mansour, Aleksandrs Slivkins, Vasilis Syrgkanis, and Steven Wu. 2016. Bayesian Exploration: Incentivizing Exploration in Bayesian Games. In *16th ACM Conf. on Economics and Computation (ACM EC)*.

[35] Yishay Mansour, Aleksandrs Slivkins, and Steven Wu. 2018. Competing Bandits: Learning under Competition. In *9th Innovations in Theoretical Computer Science Conf. (ITCS)*.

[36] Manish Raghavan, Aleksandrs Slivkins, Jennifer Wortman Vaughan, and Zhiwei Steven Wu. 2018. The Externalities of Exploration and How Data Diversity Helps Exploitation. In *Conf. on Learning Theory (COLT)*. 1724–1738.

[37] Sven Schmit and Carlos Riquelme. 2018. Human Interaction with Recommendation Systems. In *Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*. 862–870.

[38] Adish Singla and Andreas Krause. 2013. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *22nd Intl. World Wide Web Conf. (WWW)*. 1167–1178.

Diversity and Exploration in Social Learning

Nicole Immorlica
Microsoft Research
nicimm@microsoft.com

Jieming Mao
University of Pennsylvania
maojm517@gmail.com

Christos Tzamos
University of Wiscosin-Madison
tzamos@wisc.edu

ABSTRACT

In consumer search, there is a set of items. An agent has a prior over her value for each item and can pay a cost to learn the instantiation of her value. After exploring a subset of items, the agent chooses one and obtains a payoff equal to its value minus the search cost. We consider a sequential model of consumer search in which agents' values are correlated and each agent updates her priors based on the exploration of past agents before performing her search. Specifically, we assume the value is the sum of a common-value component, called the quality, and a subjective score. Fixing the variance of the total value, we say a population is more diverse if the subjective score has a larger variance. We ask how diversity impacts average utility. We show that intermediate diversity levels yield significantly higher social utility than the extreme cases of no diversity (when agents under-explore) or full diversity (when agents are unable to learn from each other) and quantify how the impact of the diversity level changes depending on the time spent searching.

KEYWORDS

social learning, heterogeneous agents, Weitzman index

ACM Reference Format:
Nicole Immorlica, Jieming Mao, and Christos Tzamos. 2019. Diversity and Exploration in Social Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313679

1 INTRODUCTION

Members of a society often leverage the experiences of others to inform their own choices. A high school student contemplating colleges might ask former high school students about their decision-making processes: what schools were considered, why the chosen one was selected over the others, and so on. A tourist contemplating restaurants might read reviews of former diners on review websites: what was the overall impression of the restaurant, what aspects contributed to this impression, etc. This sort of communal learning is highly valuable. By listening to experiences from the past, society can avoid repeating the mistakes of the past. However, this benefit comes at a potentially significant cost. As each individual optimizes their own choice, they might naturally select a tried-and-true option over an unknown one. As a result, unknown options, which could have great potential, are never discovered.

The classic field of *social learning*, initiated by [Ban92, BHW92], models this problem in the following way. Agents arrive sequentially and choose an action (e.g., search strategy) that results in some payoff based on the state of the world (e.g., the values of the items). Each agent receives a private signal correlated with the state of the world (e.g., priors on item values) and also observes chosen actions of past agents. The literature derives conditions under which the society may "herd" on a sub-optimal action, ignoring private signals and instead exploiting information gleaned from the history of actions.

The nascent field of *incentivizing exploration* tries to use information design or mechanism design to prevent the exploitation of historical information. In this line of work, there is a principal (the review website in the case of restaurants for instance), a sequence of agents (tourists in the restaurant example), and a set of options (the restaurants themselves). Each option has an inherent value, which is common to all agents, and the goal is to discover the highest-value option as quickly as possible. The principal either explicitly pays agents to try new options [FKKK14], or sends signals that cleverly obfuscate the past experiences in a way that convinces agents to try new options [KMP13, MSS15]. While these approaches successfully mitigate the dangers of learning from the past, they require the existence of a principal to coordinate society's search.

In this paper, we study decentralized exploration. Our central hypothesis is that societies perform sufficient search, even without a coordinating principal, when there are *sufficiently diverse preferences*. Intuitively, this diversity should be 1) high enough that agents occasionally explore even when past agents had great experiences, and 2) low enough that there is still value in observing the past. In this case, then society can perhaps learn to select good options, both by learning from each individual agent as she investigates her options before making a choice, and by learning from the sequence of agents who make different choices than their peers. We ask, is there a simple model and large range of parameters in which we can observe this effect?

We explore our hypothesis in the context of a standard model of consumer search, iterated over time. In a single round of consumer search, an agent is presented with a set of items. The agent knows the distribution of her value for each item and may pay a fixed cost to learn its instantiation. The resulting utility of the agent is her value for the item she selects minus the costs she paid to learn her values. For example, a high school student contemplating colleges might know her approximate value for each and, based on this information, pick a subset to visit prior to selecting one. In our iterated version of this model, what an agent knows about her values depends on what options previous agents considered. More precisely, before performing her search, each agent updates her priors on the values observed by previous agents during their searches. In the college choice problem, this corresponds to a high

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313679

school student listening to the trip reports of students in prior years before selecting which subset of schools to visit.

The degree to which historical information impacts an agent's search depends on the amount of correlation, or *diversity*, in the agents' preferences. In order to study the impact of diversity on average utility, we introduce a model in which each option has a common value component, drawn IID from a zero-mean Gaussian with variance $1 - \sigma^2$, and each agent/option pair has a private value component, also drawn IID from a zero-mean Gaussian with variance σ^2. Thus the total value of each option for each agent, the sum of the common and private value components, is also a zero-mean Gaussian with a fixed variance of 1. The variance parameter, σ^2, can be thought of as the diversity level. For $\sigma^2 = 0$, all the value of an option comes from the common value component, whereas for $\sigma^2 = 1$, all the value comes from the private value component. Since the distribution of option values remains constant as we change the diversity level, we are able to compare the average value obtained by societies of different diversity levels.

Our theoretical and simulation results show that diversity helps in this simple model. In particular, intermediate levels of diversity always yield higher average utilities than the extremes of zero diversity ($\sigma^2 = 0$), in which preferences are perfectly correlated, or total diversity ($\sigma^2 = 1$), in which preferences are independent across agents. We further investigate the comparison between intermediate levels of diversity. Our main result, Theorem 4.3, Theorem 4.4 and simulations, identify three distinct behaviors: For very short time horizons, less diverse societies outperform more diverse ones as the benefits of the added exploration have yet to take effect. For very long time horizons, less diverse societies again outperform more diverse ones as even the less diverse societies eventually perform sufficient exploration. But for a very large range of intermediate time horizons, we see more diverse societies outperform less diverse ones. This is because the less diverse societies tend to exploit historical information at the expense of exploration, similar to the phenomenon of herding in the social learning literature.

In addition to our aforementioned results, in Section 5, we extend our results to a more general setting that allows the quality of an item to get a very high value D with tiny probability $\frac{1}{D}$. This setting models situations where even though the society might be very diverse and agents have almost no common value, in the unlikely event that some clearly superior option with value D appears, they all agree that its value is high. We use this model to capture settings where the item values are not very concentrated, as Gaussians are, but have heavier probability tails. These happen in many natural situations where breakthroughs with significantly high value occur and the whole society agrees that they are valuable. We show that for such settings, high diversity is extremely beneficial for the social good as it increases incentives for exploration and breakthroughs occur faster. The literature on incentivizing exploration [FKKK14, HKQ15] usually refers to these settings as "Diamonds in the rough" and often uses this setting to provide lower bounds. In contrast, we show diverse societies excel in these settings.

Techniques. To establish our results, we need to be able to describe the optimal behavior of agents and bound the society's average expected utility. As we argue, the optimal behavior of the agents is characterized in [Wei79]. As shown in that paper, agents define an index for each item and then search them in decreasing order of index. Unfortunately, this index is defined implicitly, making it difficult to explicitly determine the utilities obtained by the agents. A crucial result that allows us to perform our analyses is that the average utility is approximately equal to the maximum quality of an item that agents will have explored by the end of the time horizon. This result enables us to focus on bounding the maximum quality of explored items. To provide an upper bound on this value, we consider an an alternative simpler scenario where agents, instead of following the optimal policy and computing indices of the inferred distributions of values, they use a plain threshold stopping rule. We show that there exists a coupling of the two processes such that the simpler process always explores more items and then proceed to bound the highest quality achievable when using the simple threshold stopping rule for every agent. We also give a matching lower bound by bounding the number of items that agents with a given diversity level could explore and then bound the expectation of the maximum value that these many normally distributed items may have.

We show that our bounds are asymptotically tight. This is required in order to be able to perform comparisons across different diversity levels as even a constant approximation of the expected average utility would not suffice to show a clear separation as average utilities of different diversity levels can be very close multiplicatively. To prove these sharp bounds, we rely on strong concentration results for Gaussian distributions.

2 RELATED WORK

Our work is very related to the line of work in Bayesian social learning. The classic results of [Ban92, BHW92] show that in social learning with homogeneous payoffs, agents may ignore their private signals in favor of information they infer from previous agents. When this happens, society herds on sub-optimal actions. A recent strand of literature, including [ADLO11, AMMO17, GPR06, LS15a, LS15b], explore the impact of heterogeneous preferences on social learning, generally observing that heterogeneous preferences help learning when the society is well-connected. The primary difference between these works and our paper is the structure of the actions and information sets. With the exception of [GPR06], these works assume that the action of each agent is binary and the state of world is binary. On the contrary, in our paper, the action of each agent is a search strategy and the state of world (the qualities of items) is sampled from independent Gaussian distributions. This choice allows us to study the impact of diversity in a specific setting of interest, as motivated by the literature on consumer search initiated by [Wei79]. For the dispersion of information, [ADLO11, LS15a, LS15b] assume agents observe a subset of past actions based on the social network and [AMMO17, GPR06] assume agents observe the full history or summary statistics of the reviews of past agents where a review is a discrete valued function of an agent's utility. In our paper, we assume agents observe actions of all previous agents and qualities/values of items explored by all previous agents.

Incentivizing exploration has been studied in the multi-armed bandit setting. In this line of research, it is assumed that myopic agents arrive sequentially and each pulls an arm at a time. As each agent only wants to maximize its own reward, exploration

might not be done sufficiently without a principal coordinating exploration. [KMP13, MSS15] study the setting where the outcomes of the actions are only observed by the principal and the principal convinces myopic agents to explore by sending signals containing partial information of the past outcomes. [FKKK14] considers a different setting where the myopic agents observe all the previous outcomes and the principal is allowed to use money transfers to incentivize exploration. A great amount of follow-up work has been done in various settings, including [CH15, HKQ15, MSSW16, KKM$^+$17, PBS18].

There is an emerging line of work studying "greedy algorithms" in multi-armed bandits[BBK18, KMR$^+$18, RSVW18]. Greedy algorithms can be viewed as a form of social learning and these papers show that near-optimal performance can be achieved under certain diversity conditions.

[SR17] considers myopic agents with diverse preferences in the multi-armed bandit setting. They show that if subjective preferences are known to agents beforehand, myopic agents will explore all the arms even without a principal coordinating exploration. They consider a setting of qualities and subjective preferences such that the difference between any two qualities is smaller than 1 and the subjective preferences have random binary values. For each arm, this setting guarantees the existence of agents who prefer the arm no matter what quality is. Exploration is therefore guaranteed in the long run.

[ASGM17] provides simulation results (without theoretical guarantees) showing that certain amount of preference diversity can lead to an increase in the average utility in a model that is closely related to our model. The main difference to our model is that they assume each agent searches items according to popularity levels of items and stops searching once an item has value larger than some threshold. On the contrary, we assume each agent performs the optimal search.

3 MODEL

We consider a setting with infinitely many items[1] of unknown value. Agents arrive sequentially in T rounds. The agent that arrives in round t has a value $v_{i,t}$ for item i that is the sum of two terms: an objective score q_i for the item and an individual subjective score $s_{i,t}$. The objective scores (qualities) q_i are the same for all agents and are drawn from a Gaussian $\mathcal{N}(0, 1-\sigma^2)$ with mean zero and variance $1-\sigma^2$ for some $\sigma \in [0,1]$. We refer to this score as the quality of item i. The subjective scores $s_{i,t}$ vary among agents and are drawn independently from a Gaussian $\mathcal{N}(0, \sigma^2)$ with mean zero and variance σ^2. The total value of an item, $v_{i,t} = q_i + s_{i,t}$, is thus distributed according to a Gaussian $\mathcal{N}(0,1)$ with mean zero and variance one.

Definition 3.1 (Diversity Level). *We define the* diversity *of a population of agents to be the parameter σ (the standard deviation of the subjective score).*

When the diversity is 0, all the value of an item is derived from its objective score, i.e., $v_{i,t} = q_i$, and so agents' values are perfectly correlated. When diversity is 1, all the value of an item is derived from its subjective score, i.e., $v_{i,t} = s_{i,t}$, and so agents' values are independent. Intermediate diversities interpolate between these two extremes.

Initially, the agent only knows a prior $\mathcal{F}_{i,t}$ over her value for each item i, which is a function of the initial Gaussian priors and her observations from past agents' searches. The agent may learn the objective score q_i and subjective score $s_{i,t}$ of an item i, and hence its value $v_{i,t} = q_i + s_{i,t}$, by paying a search cost c, where c is a fixed constant smaller than $\frac{1}{\sqrt{2\pi}}$ (chosen so that the agents are incentivized to explore at least one item). A search strategy of an agent is a mapping from observations to either a choice from among previously explored items, or an item to explore next. The utility $u_t(\sigma, \{\mathcal{F}_{i,t}\})$ of the agent for a given search strategy and priors is her expected value for the chosen item i^*, $v_{i^*,t}$, minus her total expected search cost, i.e., c times the expected number of items she explores. The chosen item i^* has to be an explored item.

The optimal search strategy σ^* for an agent in this setting was characterized by [Wei79] in a slightly more general setting called the *Pandora's problem*. This strategy assigns a score, called the *Weitzman index*, to every item and then explores items with positive indices in decreasing order of their indices until the observed value is greater than the following item's index.

Definition 3.2 (Weitzman Index). *The* Weitzman index $x^*_{i,t}$ *is defined as* $\mathbb{E}_{v_{i,t}\sim\mathcal{F}_{i,t}}[\max(v_{i,t} - x^*_{i,t}, 0)] = c$.

We assume that all agents use the optimal strategy and explore items in decreasing order of Weitzman index. We denote by x^* the Weitzman index of an item that has not been explored previously, i.e. the solution to $\mathbb{E}_{v\sim\mathcal{N}(0,1)}[\max(v - x^*, 0)] = c$. The choice of $c < \frac{1}{\sqrt{2\pi}}$ guarantees that $x^* > 0$ and thus every agent will explore at least one item.

Once an item is explored (but not necessarily chosen) by some agent, agents in subsequent rounds obtain information about that item. There are multiple choices we can make about exactly what information agents learn; the right choice is dictated by the setting and whether agents can infer each others' subjective biases. In this paper, we will focus on a "revealed quality" model, where, after an item i is explored by some agent, its quality q_i is revealed to all agents in future rounds. In this model, agents update their prior for a previously explored item i to $F_{i,t} = \mathcal{N}(q_i, \sigma^2)$ before computing its Weitzman index $x^*_{i,t}$. We also consider a "revealed value" model, where after an item i is explored by some agent in round t, only its value to that agent $v_{i,t} = q_i + s_{i,t}$ is revealed to future agents. In this model, the optimal strategy of the agents becomes more complicated. Given the past values of agents for a given item, the current agent forms a posterior distribution of the item's quality. This posterior can be expressed as a Gaussian distribution with mean and variance that depend on the average past observed values and the number of times that item gets explored. We can show similar results in the "revealed value" model. We omit them because of the space limit.

Our goal is to understand how the expected average utility changes depending on the diversity level σ of the society. The impact of diversity will vary with the time spent searching, and so we define the average utility with respect to a time horizon T. Let $\mathcal{H}_t$ be a history indicating the information obtained about the

[1] This assumption allows us to avoid edge cases in our analyses; our results carry over to settings with finitely many items so long as there are sufficiently many of them (in particular, more than the logarithm of the time horizon).

items explored in the first $t-1$ rounds and $\mathcal{F}(\mathcal{H}_t)$ be the induced priors for the agent arriving in round t.

Definition 3.3 (Average Utility). *We define* $A(\sigma, T) = \frac{1}{T}\sum_{t=1}^{T} E_{\mathcal{H}_t}[u_t(\sigma, \mathcal{F}(\mathcal{H}_t))]$ *to be the* average expected utility *over agents in the first T rounds when the diversity level is σ.*

3.1 Justification of Our Model

Here we provide the rationale behind the model choices.

Decomposition in Common and Private Values The assumption that agent values decompose in independent common and private components is commonly used in the empirical auction literature. Going back to the work of [LV98] who showed how to perform the decomposition based on deconvolution, it has been shown that it predicts the agent behavior quite accurately in real-world data [Kra11, Ask10].

Normally Distributed Values The choice of normal distributions for the common and private values enables expressing a wide range of diversity levels with a single parameter. The constant sum of variances property allows for fair comparisons as in all cases agents have the same distribution of values for every option. Normal distributions appear in practice; e.g., the distribution of meta-critic scores for movies on Rotten Tomatoes closely fits a normal distribution; and the distribution of the common value in the US offshore oil and gas auctions has been shown to fit a normal distribution using econometrics [STZ17].

Pandora's Box Search Model The Weitzman search strategy is the optimal policy that an agent can follow to maximize his expected utility. It has been successfully applied to real-world applications in the marketing literature [BKM16, KAB10, HC17].

4 HOW DIFFERENT DIVERSITY LEVELS COMPARE

There are two opposing forces that affect average utility – exploration and exploitation. When the society is not very diverse, agents tend to exploit previously explored options as it is quite costly for them to search for new alternatives that could potentially give them a higher value. On the other hand, diverse societies tend to explore more, but cannot exploit as much since information collected in the past is less valuable to them.

Focusing solely on exploration or exploitation can come at a significant cost. In the extreme case where the society is not at all diverse (i.e., $\sigma = 0$) and focuses on exploitation, we can show the society's average expected utility, $A(0, T)$, can be upper bounded by a constant. In particular, $A(0, T)$ does not depend on the number of rounds T.

Lemma 4.1. *For any T, $A(0, T) \leq \sqrt{\ln\left(\frac{1}{2\pi c^2}\right)} + 2$.*

To prove Lemma 4.1, we first bound by a constant the Weitzman index x^* of an unsearched item, i.e. one whose quality is not yet revealed. We then argue that once some item with sufficient quality has been identified (with quality $q_i > x^* + c$), no other item will be explored in future rounds. We conclude the proof by upper-bounding the expected quality of such an item conditional on its quality being higher than $x^* + c$. This directly gives a bound for the average utility of the agents. The proof of Lemma 4.1 is given in Appendix B.

In the other extreme case where agents are fully diverse, the society's average expected utility, $A(1, T)$, is even worse. We show that it is always the case that a fully diverse population obtains lower utility than a non-diverse population and so, in particular, its utility is also bounded by a constant independent of T.

Lemma 4.2. *For any T, $A(1, T) \leq A(0, T)$.*

To prove Lemma 4.2, we observe that the first agent in any society performs the same search no matter the diversity level and so achieves the same expected utility. When the diversity level is 0, the expected utility improves across rounds, while when the diversity level is 1, the expected utility stays the same across rounds, implying the result. The proof of Lemma 4.2 is given in Appendix B.

In both of these extreme cases, the average expected utility is bounded by a constant that depends on the search cost c but not on the number of rounds. However, as we show, any intermediate diversity level $\sigma \in (0, 1)$ can be significantly better than those extreme cases as the average expected utility in those cases grows with the number of rounds. Thus, it becomes clear that societies with intermediate diversity levels achieve higher average expected utility than those with extreme diversity levels.

Our main result is a characterization of how different diversity levels $\sigma, \sigma' \in (0, 1)$ compare for different numbers of rounds T. We identify three regions, depending on the number of rounds, that affect which diversity level is better than the other.

When the number of rounds T is small, the smaller diversity level σ' achieves higher average expected utility. This is because the benefit of broader past explorations is not as significant compared to knowing the values of a smaller subset of items more accurately. We show numerically in Section 4.1 that this is indeed the case and that initially, for very few rounds, almost no diversity is preferable.

When T takes intermediate values, the value of exploring a larger number of options becomes more significant than higher accuracy on fewer options and thus the higher diversity level σ is preferred. Theorem 4.3 shows the existence of this region under a technical assumption which requires a small gap between σ' and σ.

Theorem 4.3 (Intermediate T). *Let $\sigma, \sigma' \in (0, 1)$ be two diversity levels with $\sigma' < \sigma \cdot e^{-O\left(\frac{1}{1-\sigma^2}\right)}$. There exists an interval (T_1, T_2) such that $A(\sigma, T) > A(\sigma', T)$ for all $T \in (T_1, T_2)$, i.e. the large diversity level σ outperforms the small diversity level σ'. Here T_1 and T_2 depend on σ and σ'.*

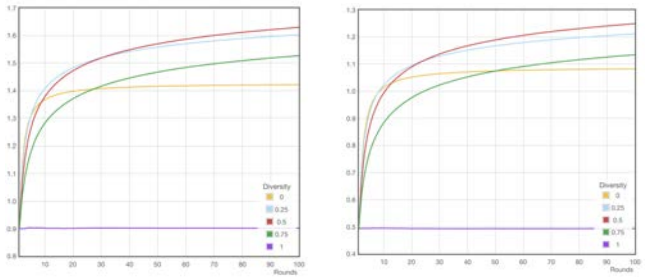
When T is very large, the smaller diversity σ' level becomes again preferable to the higher level σ as in both cases a comparably large number of options have been explored in which case it is again more beneficial to know the value of the explored items more accurately. This region is described in Theorem 4.4.

Theorem 4.4 (Very large values of T). *Let $\sigma, \sigma' \in (0, 1)$ be two diversity levels with $\sigma' < \sigma$. There exists an interval (T_3, ∞) such that $A(\sigma', T) > A(\sigma, T)$ for all $T \in (T_3, \infty)$, i.e. the small diversity level σ' outperforms the large diversity level σ. Here T_3 depends on σ and σ'.*

Theorems 4.3 and 4.4 show that the diversity level σ is preferable to a small diversity level σ' for intermediate number of rounds but as the number of rounds increases the smaller diversity level becomes more socially beneficial. The proofs of the theorems are presented in Section 4.2.

We note that even though $A(\sigma', T)$ is guaranteed to be larger than $A(\sigma, T)$ for large values of T in the third region, this will happen at extremely large values of T. In fact, in all simulations presented in the next sub-section, we ran up to 10^6 rounds for different values of c, we did not observe any transition from the second region where $A(\sigma', T) < A(\sigma, T)$ to the third region where $A(\sigma', T) > A(\sigma, T)$. This is because the behavior of the $A(\sigma, T)$ is tightly characterized by the upper-bound we will show in Proposition 4.5, i.e., $\sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln(T))}$, which shows that the dependence on T is doubly-logarithmic and thus extremely large values of T are needed to reach the third region. Thus, for all practical purposes, we observe in simulations that the optimal diversity level increases monotonically with the number of rounds.

4.1 Behavior for small T - Numerical Simulations



(a) $c = 0.1$ **(b)** $c = 0.2$

Figure 1: Simulations of the average expected utility for different number of rounds and diversity levels.

Before we proceed with the proof of Theorems 4.3 and 4.4, we present some simulations that show the evolution of society's expected average expected utility with the number of rounds T in the "revealed quality" model. Figure 1, shows the plots $A(\sigma, T)$ for different number of rounds T up to 100 and different diversity levels $\sigma \in \{0, 0.25, 0.5, 0.75, 1\}$. We present two plots, one for $c = 0.1$ and one for $c = 0.2$. For each of the different scenarios, we run 10^5 independent runs.

As can be seen from the figure, all different diversity levels start at the same point for $T = 1$. The full diverse case $\sigma = 1$ remains constant at that value achieving the lowest possible utility than all diversity levels as every agent searches from scratch every time without reusing any information from the past. The case without diversity ($\sigma = 0$) on the other hand, initially has an increasing average social utility but as we've shown in Lemma 4.1, approaches a constant value after few rounds.

Diversity levels $\sigma = 0.25, 0.5$, and 0.75, achieve lower expected utility than the no-diversity case in the first rounds but eventually all outperform it as described by Theorem 4.3. The lowest diversity level $\sigma = 0.25$ manages to do this faster but also gets outperformed by the higher diversity level $\sigma = 0.5$ in later rounds. The diversity level 0.75 also outperforms 0.25 after $\sim 10^3$ rounds (not shown in the graph) but doesn't seem to outperform 0.5 even after simulating for 10^6 rounds. This is an indication that the gap in diversity levels required by Theorem 4.3 is necessary for statement to hold.

4.2 Behavior for intermediate and large values of T

We now move on to prove Theorems 4.3 and 4.4. Our main technical contribution that drives the proof of those Theorems is a sharp characterization of the average expected utility for any fixed diversity level $\sigma \in (0, 1)$. Proposition 4.5 gives a closed form expression for the average expected utility which is tight up to additive constants.

Proposition 4.5. *For any $\sigma \in (0, 1)$, and any $T > 0$,*

$$A(\sigma, T) = \sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln T)} \pm O_c(1).$$

The proof of Proposition 4.5 can be found in Appendix B. The main observation in establishing the statement is that the expected utility $A(\sigma, T)$ is well approximated by the maximum quality of the revealed items after T rounds (Lemma B.3). Using this observation, we obtain the result by giving upper and lower bounds to the best item that is revealed in the end which in turn is closely related to the number of revealed items. To give the lower-bound (Lemma B.4), we bound the rate that new items are revealed by bounding the probability that all available options are unsatisfactory for a given agent, i.e. when his true value is negative for all currently explored items. For the upper-bound (Lemma B.5), we show that the probability that a new item is explored is exponentially small with the number of high quality items already known. Thus as we show, the number of items n explored after T rounds is logarithmic in T. We obtain the bound on the maximum quality of a revealed item by noting that the maximum of n gaussian random variables with variance $1-\sigma^2$ is approximately $\sqrt{1-\sigma^2} \cdot \sqrt{2\ln n}$. This gives us the approximate expression of the expected utility $A(\sigma, T)$.

Given the closed form expression of Proposition 4.5, we can easily obtain Theorems 4.3 and 4.4 as corollaries.

Proof of Theorem 4.3:

By Proposition 4.5, there exists a constant C_1 such that

$$\left| A(\sigma, T) - \sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln(T))} \right| \leq C_1.$$

For notation convenience, define constant $C_2 = \sqrt{2\ln(2)}$. Set $T_1 = \exp\left(\frac{1}{\sigma^2} \exp\left(\frac{(2C_1+C_2)^2}{2(1-\sigma^2)}\right)\right)$ and $T_2 = \exp\left(\frac{1}{\sigma'^2}\right)$.

For any $T < T_2$, we can upper bound $A(\sigma', T)$ by constant $C_1 + C_2$:

$$\begin{aligned} A(\sigma', T) &\leq \sqrt{1-\sigma'^2} \cdot \sqrt{2\ln(1+\sigma'^2 \ln(T))} + C_1 \\ &< \sqrt{2\ln\left(1+\sigma'^2 \cdot \frac{1}{\sigma'^2}\right)} + C_1 \leq C_2 + C_1. \end{aligned}$$

For any $T > T_1$, we can lower bound $A(\sigma, T)$ by constant $C_1 + C_2$:

$$\begin{aligned} A(\sigma,T) &\geq \sqrt{1-\sigma^2}\cdot\sqrt{2\ln(1+\sigma^2\ln(T))} - C_1 \\ &> \sqrt{1-\sigma^2}\cdot\sqrt{2\ln\left(1+\sigma^2\cdot\frac{1}{\sigma^2}\exp\left(\frac{(2C_1+C_2)^2}{2(1-\sigma^2)}\right)\right)} - C_1 \\ &> 2C_1 + C_2 - C_1 = C_1 + C_2. \end{aligned}$$

Therefore for any $T \in (T_1, T_2)$, $A(\sigma, T) > A(\sigma', T)$. In order to make sure $T_1 < T_2$, we need $\sigma' < \sigma \cdot \exp\left(-\frac{(2C_1+C_2)^2}{4(1-\sigma^2)}\right)$. □

Proof of Theorem 4.4:

For any $\sigma' < \sigma$, we have that $\lim_{T\to\infty} \frac{A(\sigma,T)}{A(\sigma',T)} = \frac{\sqrt{1-\sigma^2}}{\sqrt{1-\sigma'^2}} < 1$. This immediately implies that for any $\sigma' < \sigma$, there exists a large enough T_3 such that $A(\sigma, T) < A(\sigma', T)$ for all $T > T_3$, as required by Theorem 4.4.

□

5 AN EXTENSION WITH RARE BUT HIGH REWARD ITEMS

We extend our results to a setting where agents' values are correlated in a more structured way. In this extension, we introduce an additional term of objective value in the quality of an item, i.e. we write q_i as $d_i + r_i$ where d_i takes value D with probability $p = \Theta(D^{-1})$ and 0 otherwise while r_i is sampled from $\mathcal{N}(0, 1-\sigma^2)$ as before. We call an item with $d_i = D$ a "diamond" due to its rarity and high value. We assume that $c \geq p \cdot D$ so that the exploration cost is larger than the expected value of an item.

In this diamond model, diversity gives significant help in early rounds. Our main result in this model is that the existence of a diamond can significantly change the behavior and average utilities of agents of different diversity levels.

Theorem 5.1. *Consider any constant diversity level $\sigma < 1$. If the number of rounds T satisfies $\omega(D) \leq T \leq \exp(o(D))$, then $A(\sigma, T) = o(1) \cdot A(1, T)$ as the diamond value D increases.*

In contrast to the previous section, where exploration in the full diverse setting is meaningless as future agents cannot benefit from the experience of their predecessors, Theorem 5.1 shows that if there is even a tiny probability that a very desirable and socially beneficial alternative is identified, diversity helps significantly. Therefore, we show that in societies where even fully diverse agents can agree on a clearly superior outcome which might occur with very small probability, larger diversity levels work in society's favor as they help identify this option faster.

The full proof of Theorem 5.1 can be found in Appendix B. Here we show some important steps of the proof.

To prove Theorem 5.1, we first show that when the diversity level is 1 and $T = \omega(1/p)$, the expected average utility is close to the diamond value D.

Lemma 5.2. *$A(1, T)$ is at least $(1 - o(1))D$ for $T = \omega(1/p)$.*

The main idea behind the proof of Lemma 5.2 is to show that at least one new item is searched in each round before a diamond is discovered and thus in $T = \omega(1/p)$, the diamond will be found. The proof of Lemma 5.2 can be found in Appendix B.

Next we proceed to bound the expected average utility for any diversity level $\sigma < 1$, when $T \leq \exp(o(D))$, $A(\sigma, T)$. We start with the extreme case when $\sigma = 0$. The proof is similar to the proof of Lemma 4.1 and is given in Appendix B.

Lemma 5.3. *$A(0, T)$ is at most $D \cdot p + O(1)$.*

The case for intermediate diversity level $\sigma \in (0, 1)$ is more complicated. We give an upper bound on the average utility in the appendix (Lemma B.9).

A TOOLS FOR GAUSSIAN DISTRIBUTIONS

In this section, we list several facts and lemmas about Gaussian distributions. Throughout this document, we use Φ to refer to the *CDF* of the Gaussian $\mathcal{N}(0, 1)$ and φ to refer to the density function of the Gassian $\mathcal{N}(0, 1)$.

Fact A.1. *Let $\Phi(t)$ be the CDF of standard Gaussian distribution then for $t > 0$,*

$$\frac{1}{\sqrt{2\pi}}\cdot\frac{t}{t^2+1}\cdot\exp(-t^2/2) \leq 1-\Phi(t) \leq \min\{\frac{1}{\sqrt{2\pi}t}, 1\}\exp(-t^2/2).$$

Fact A.2 ([Bar08]). *Let $\Phi(t)$ be the CDF of standard Gaussian distribution. Let $\varphi(t)$ be the density of standard Gaussian distribution. For any $t > 0$,*

$$\frac{\varphi(t)}{1-\Phi(t)} \leq \frac{t}{2} + \frac{\sqrt{t^2+4}}{2}.$$

The following two lemmas can be proved by standard techniques about Gaussian distributions. We omit their proofs because of space limit.

Lemma A.1. *For any $T > 0$, let $x_1, ..., x_T$ be sampled from $\mathcal{N}(0, \sigma^2)$ independently. Then we have*

$$Pr\left[\min_{i\in[T]}\frac{x_1+\cdots+x_i}{i} \leq -2\sigma\right] \leq 1/4.$$

Lemma A.2. *For any $x^*, T > 0$, let $x_1, ..., x_T$ be sampled from $\mathcal{N}(0, \sigma^2)|_{\geq x^*}$ independently. Then we have*

$$\mathbb{E}\left[\max_{i\in[T]} x_i\right] \leq \sigma\sqrt{2\ln(2T)} + x^*.$$

B OMITTED PROOFS

B.1 Ommitted Proofs of Lemma 4.1 and Lemma 4.2

Lemma B.1 (Restatement of Lemma 4.1). *For any T, $A(0, T) \leq \sqrt{\ln\left(\frac{1}{2\pi c^2}\right)} + 2$.*

Proof. The proof plan is to first bound by a constant the Weitzman index x^* of an unsearched item, i.e. one whose quality is not yet revealed. We then argue that once some item with sufficient quality has been identified (with quality $q_i > x^* + c$), no other item will be explored in future rounds. We conclude the proof by upper-bounding the expected quality of such an item conditional on its quality being higher than $x^* + c$. This directly gives a bound for the average utility of the agents.

We start by bounding the Weitzman index x^* of an unsearched item. For an unsearched item, the distribution of the value is $\mathcal{N}(0, 1)$.

The Weitzman index x^* satisfies $\mathbb{E}[\max(X - x^*, 0)] = c$ where $X \sim \mathcal{N}(0, 1)$. Using Fact A.1 on Gaussian distributions, we see

$$\begin{aligned} c &= \mathbb{E}[\max(X - x^*, 0)] \\ &= \mathbb{E}[X - x^* | X - x^* > 0] \cdot Pr[X - x^* > 0] \\ &= \left(-x^* + \frac{\varphi(x^*)}{\Phi(-x^*)}\right) \cdot \Phi(-x^*) \\ &= (-x^*)(1 - \Phi(x^*)) + \varphi(x^*) \\ &\leq (-x^*)\left(\frac{\varphi(x^*)}{x^*} - \frac{\varphi(x^*)}{(x^*)^3}\right) + \varphi(x^*) \\ &= \frac{\varphi(x^*)}{(x^*)^2} = \frac{1}{\sqrt{2\pi} \cdot (x^*)^2 \cdot \exp((x^*)^2/2)}. \end{aligned}$$

This implies an upper bound on the Weitzman index of an unsearched item:

$$x^* \leq \max\left(1, \sqrt{\ln\left(\frac{1}{2\pi c^2}\right)}\right).$$

When an item has been searched with observed quality q_i, since the diversity level is 0, the values of this item for future agents are determinstically q_i. Therefore the Weitzman index of this item is simply $q_i - c$.

Once some item has quality $q_i > c + x^*$, its Weitzman index is larger than x^* which means it will be searched before any unsearched items. And once this item is searched in a round, we know its realized value is q_i which is larger than x^*. Therefore no more unsearched items will be searched.

Let Q be the quality of the first item with quality larger than $c + x^*$. We have $\mathbb{E}[Q] = \mathbb{E}_{X \sim \mathcal{N}(0,1)}[X | X > c + x^*]$. We know that starting from the round this item is searched, each agent's utility is at most $Q - c$ and before this item is ever searched, each agent's utility is at most $c + x^* - c \leq Q - c$. Therefore we can bound the average utility as following:

$$\begin{aligned} A(0, T) &\leq \mathbb{E}[Q - c] \leq \mathbb{E}_{X \sim \mathcal{N}(0,1)}[X | X > c + x^*] - c \\ &= \frac{\varphi(c + x^*)}{1 - \Phi(c + x^*)} - c \\ &\leq \frac{x^* - c + \sqrt{(x^* + c)^2 + 4}}{2} \text{ (by Fact A.2)} \\ &\leq x^* + 1 \\ &\leq \sqrt{\ln\left(\frac{1}{2\pi c^2}\right)} + 2. \end{aligned}$$

□

Lemma B.2 (Restatement of Lemma 4.2). *For any T, $A(1, T) \leq A(0, T)$.*

Proof. First notice that when the diversity level is 1, values are completely independent of the history and have the same distributions across agents. Therefore, the expected utilities of all agents are the same. Therefore $A(1, T)$ equals the expected utility of the first agent.

When the diversity level is 0, consider the following strategy for agents except the first agent: just search the item chosen by the first agent and choose it. By using this strategy, we know all agents will get at least the utility of the first agent. By the optimality of Weitzman index strategy, we know $A(0, T)$ will be at least the expected utility of the first agent.

Finally, notice that the expected utility of the first agent does not depend on the diversity level. This implies $A(0, T) \geq A(1, T)$. □

B.2 Omitted Proof of Proposition 4.5

We proceed to show Proposition 4.5. We first relate the average expected utility $A(\sigma, T)$ to the expectation of the positive part of the highest quality explored during the first T rounds when the diversity level is σ, denoted by $M(\sigma, T)$. Here the positive part of x means $\max(x, 0)$. In Lemma B.3, we show that $A(\sigma, T)$ is sandwiched between $M(\sigma, \gamma T)$ and $M(\sigma, T)$. Subsequently, we provide sharp bounds on $M(\sigma, T)$ and show that $M(\sigma, \gamma T)$ and $M(\sigma, T)$ are very close for large T.

Lemma B.3. *For any diversity level $\sigma \in (0, 1)$ and any $\gamma \in (0, 1)$,*

$$(1 - \gamma)\left(M(\sigma, \gamma T) - c\right) \leq A(\sigma, T) \leq M(\sigma, T) + O_c(1).$$

Proof. Let's first prove the left hand side. For each agent after round γT, a sub-optimal strategy is to explore and pick the item with highest quality searched before the end of round γT and leave. This gives utility $M(\sigma, \gamma T) - c$ in expectation. By the optimality of the Weitzman index strategy, agents after round γT should at least get $M(\sigma, \gamma T) - c$ in expectation. Again by the optimality of the Weitzman index strategy, agents in the first γT rounds should get utility at least 0 in expectation. Thus, we have $A(\sigma, T) \geq (1 - \gamma)\left(M(\sigma, \gamma T) - c\right)$.

Now let's consider the right hand side. The main idea is to upper bound the expected utility of the last round T by $M(\sigma, T) + O_c(1)$.

Let n be the number of items explored in the first T rounds, with qualities $q_1, ..., q_n$. In round T, the agent's expected utility will be at least as large as the expected utility of agents in previous rounds as more information is revealed. Let j be the item where the T-th agent stops his search. We fix j, n and $q_1, .., q_n$ in this argument and expectations are taken over randomness of subjective scores of the T-th agent. T-th agent's value for any previously explored option is less than x_j^*, the Weitzman index of item j, as he chose to explore item j. Moreover, since he decided to stop at j, his value at j, $q_j + s_{j,T}$ must be higher than x_{j+1}^*, the Weitzman index of the next item to search in round T. After the search, the T-th agent might take item j or some other items with value smaller than x_j^*. Thus, his expected utility, is at most $\max(q_j + s_{j,T}, x_j^*) - c$ and $s_{j,T}$ is sampled from $\mathcal{N}(0, \sigma^2)$ conditioned on $q_j + s_{j,T} > x_{j+1}^*$.

Using Lemma A.2, we can upper bound the expected utility of the agent in round T by

$$\begin{aligned} \mathbb{E}[\max(q_j + s_{j,T}, x_j^*)] &\leq \mathbb{E}_{X \sim \mathcal{N}(0, \sigma^2)}[q_j + X | q_j + X > x_j^*] \\ &\leq q_j + \mathbb{E}_{X \sim \mathcal{N}(0, \sigma^2)}[X | X > \max(x_j^* - q_j, 0)] \\ &\leq q_j + \max(x_j^* - q_j, 0) + O(1) \text{ (by Lemma A.2)} \end{aligned}$$

Recall x^* is the Weitzman index of an unexplored item. If item j has been searched before round T, we have $x_j^* = x^*$. We can simply get

$$\mathbb{E}[\max(q_j + s_{j,T}, x_j^*)] \leq \max(q_j, 0) + x^* + O(1).$$

If item j hasn't been searched before round T, we bound the difference between x_j^* and q_j. The definition of Weitzman index

gives us,

$$c = \mathbb{E}_{X\sim\mathcal{N}(0,\sigma^2)}[\max(X-(x_j^*-q_j),0)] \le \mathbb{E}_{X\sim\mathcal{N}(0,1)}[\max(X-(x_j^*-q_j),0)].$$

Note that the last expression is strictly less than c when $x_j^*-q_j > x^*$ implying that $x_j^*-q_j \le x^*$. Then we get

$$\mathbb{E}[\max(q_j+s_{j,T},x_j^*)] \le q_j + x^* + O(1)$$

To sum up, no matter item j has been searched or not before round T, the expected utility of the agent in round T is at most $M(\sigma,T)+x^*+1$. Since the agent in round T has expected utility at least as high as expected utilities of agents in previous rounds, we conclude that $A(\sigma,T) \le M(\sigma,T)+x^*+O(1)$. □

In Lemma B.4, we give a lower bound on $M(\sigma,T)$. The main idea of the proof is to give a lower bound on the number of searched items and then show that $M(\sigma,T)$ is roughly the expected value of the maximum of that many independent Gaussians. The lower bound on the number of searched items can be achieved by considering events (indicated as W_t in the proof) when all the previously searched items have low values (caused by very negative subjective scores).

Lemma B.4. *For any diversity level $\sigma \in (0,1)$, and any $T > \exp(13/\sigma^2)$,*

$$M(\sigma,T) \ge \sqrt{1-\sigma^2}\cdot\sqrt{2\ln(\sigma^2\ln T)} - O(1).$$

Proof. To prove the statement, we fix a sufficiently large parameter n and we show that with high probability after a sufficiently large number of rounds T, two events will happen:

- The maximum quality among the first n items is at least $h = \sqrt{1-\sigma^2}\cdot\sqrt{2}\cdot(\sqrt{\ln n}-1)$,
- At least one agent will explore an item with quality higher than h.

We first bound the probability that the maximum quality of the first n items is less than h. As the qualities of all items are independent, this is a standard calculation about the maximum of independent Gaussian random variables. Setting $g = \frac{h}{\sqrt{1-\sigma^2}}$, we have:

$$\begin{aligned}
&Pr\left[\max(q_1,\ldots,q_n)\le h\right]\\
&= (Pr\left[q_1<h\right])^n = \Phi\left(\frac{h}{\sqrt{1-\sigma^2}}\right)^n\\
&\le \left(1-\frac{\exp\left(-\frac{1}{2}\cdot g^2\right)}{\sqrt{2\pi}}\cdot\frac{g}{g^2+1}\right)^n && \text{from Fact A.1}\\
&\le \exp\left(-n\cdot\frac{\exp\left(-\frac{1}{2}\cdot g^2\right)}{\sqrt{2\pi}}\cdot\frac{g}{g^2+1}\right)\\
&\le \exp\left(-\frac{\exp\left(2\sqrt{\ln n}\right)}{e\sqrt{\pi}}\cdot\frac{\sqrt{\ln n}-1}{2(\sqrt{\ln n}-1)^2+1}\right) \le \frac{1}{\sqrt{\ln n}} && \text{for } n\ge 5.
\end{aligned}$$

Assuming that there is some item with quality higher than h among the first n, we now bound the probability that some agent will explore it. Let W_t be the indicator of the event that in round t, for all items $i\in[n]$ with $q_i \le h$, $s_{i,t} < -h$. It is clear that if $W_t = 1$ for some t, $v_{i,t} < 0$ for all items with quality less than h so the agent t will keep exploring until he finds the item with quality higher than h.

We lower bound the probability that $W_t = 1$. We have that:

$$\begin{aligned}
Pr[W_t=1] &\ge \left(Pr\left[s_{i,t}<-h\right]\right)^n = \left(1-\Phi\left(\frac{h}{\sigma}\right)\right)^n\\
&\ge \exp\left(-\frac{1}{2}\cdot\left(\frac{h}{\sigma}\right)^2\cdot n\right) \ge \exp\left(-\frac{n\ln n}{\sigma^2}\right).
\end{aligned}$$

Setting $n = \frac{Z}{\ln(Z)}$ for $Z = \sigma^2\ln T$, we get that $Pr[W_t=1] \ge \frac{\ln Z}{T}$. Thus, the probability that no agent will find the high quality item is at most: $Pr[\forall t, W_t = 0] \le \left(1-\frac{\ln Z}{T}\right)^T \le \frac{1}{Z} \le \frac{1}{\sqrt{\ln n}}$.

Thus, overall we have that if $n \ge 5$

$$\begin{aligned}
M(\sigma,T) &\ge (1-\frac{2}{\sqrt{\ln n}})h \ge \sqrt{1-\sigma^2}\cdot\sqrt{2}(\sqrt{\ln n}-3)\\
&\ge \sqrt{1-\sigma^2}\cdot\sqrt{2}(\sqrt{\ln Z}-4) && \text{for } Z > n \ge 5\\
&\ge \sqrt{1-\sigma^2}\cdot\sqrt{2\ln(\sigma^2\ln T)} - 6.
\end{aligned}$$

Thus, setting T so that $n \ge 5$, we get the required bound. □

In addition to this lower bound, in Lemma B.5, we provide an upper bound on $M(\sigma,T)$. The main idea of the proof is to consider items with good qualities and show that the chance of searching a new item decreases exponentially with the number of such items. We perform the analysis using a coupling argument to an imaginary setting in which a larger number of items is always explored.

Lemma B.5. *For any diversity level $\sigma\in(0,1)$,*

$$M(\sigma,T) \le \sqrt{1-\sigma^2}\cdot\sqrt{2\ln(1+\sigma^2\ln(T))} + O_c(1).$$

Proof. Let $I(t)$ be a random variable that indicates the number of explored items after the agent in the t-th round has finished exploring, label items in the order in which they were first explored, and let $q_1, q_2, \ldots, q_{I(T)}$ be random variables that denote the quality of the items explored in the first T rounds. Then, in this notation, $M(\sigma,T)$, the expected positive part of the maximum quality of the explored items at the end of round T, is $\mathbb{E}[\max(0,\max_{i\le I(T)} q_i)]$.

To bound this expression, we define the following alternative exploration procedure that is significantly easier to analyze. We will show that there is always a coupling between the original and the alternative processes so that more items are always explored in the latter. Let x^* be the Weitzman index that satisfies $\mathbb{E}[\max(X-x^*,0)] = c$ when $X\sim\mathcal{N}(0,1)$. In our alternative process, every agent goes through every item i starting from $i=1$ and, whenever $q_i \ge x^*+c+1$, flips a coin that has probability $\Phi(1/\sigma)$ of coming heads. If the coin comes heads he stops, otherwise he continues on to the next item $i+1$. Let $I'(t)$ be the maximum number of items that have been explored by the agents in the first t rounds of this alternative procedure.

We define a coupling such that $I(t) \le I'(t)$. In the original process, we have that every agent t has value $q_i + s_{i,t}$ for item i. In the alternative process, if $q_i \ge x^*+c+1$, we couple the coin with the event that $s_{i,t} \ge -1$ which also has probability $\Phi(1/\sigma)$. Assuming inductively that $I(t-1) \le I'(t-1)$, we will show that $I(t) \le I'(t)$. This follows immediately if $I(t) \le I'(t-1)$ since we have that

$I'(t-1) \le I'(t)$. Otherwise, when $I(t) > I'(t-1)$, the original agent in the t'th round must have explored new item(s). Therefore, he must have $s_{i,t} \le -1$ for all items $i < I(t)$ for which $q_i \ge x^* + c + 1$ as he must have explored all of those first before reaching item $I(t)$. By definition of the coupling, the alternative agent will not stop in any item i such that $i < I(t)$ and thus $I'(t) \ge I(t)$.

Thus, $M(\sigma, T) = \mathbb{E}[\max(0, \max_{i \le I(T)} q_i)] \le \mathbb{E}[\max(0, \max_{i \le I'(T)} q_i)]$. We complete the proof by bounding $\mathbb{E}[\max(0, \max_{i \le I'(T)} q_i)]$ in Lemma B.6. □

Lemma B.6. *Let $I'(T)$ be as defined in the proof of Lemma B.5. Then*

$$\mathbb{E}[\max(0, \max_{i \le I'(T)} q_i)] \le \sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln T)} + O_c(1).$$

Proof. For notation convenience, define $p_k = \Pr[|\{i | i \le I'(T) \wedge q_i \ge x^* + c + 1\}| = k]$. Letting m_k denote the expected maximum of k random variables distributed according to $\mathcal{N}(0, 1-\sigma^2)|_{\ge x^*+c+1}$, we have that $\mathbb{E}[\max(0, \max_{i \le I'(T)} q_i)] \le \sum_k m_k p_k$. From Lemma A.2, we have that $m_k \le \sqrt{1-\sigma^2} \cdot \sqrt{2\ln(2k)} + x^* + c + 1 \le \sqrt{1-\sigma^2} \cdot \sqrt{2\ln k} + x^* + c + 3$ and thus we get that

$$\begin{aligned}
&\mathbb{E}[\max(0, \max_{i \le I'(T)} q_i)] \\
\le &\sqrt{1-\sigma^2} \sum_k \sqrt{2\ln k} \cdot p_k + (x^* + c + 3) \\
\le &\sqrt{1-\sigma^2} \cdot \sum_{k \ge z} \sqrt{2\ln k} \cdot p_k + \sqrt{1-\sigma^2} \cdot \sqrt{2\ln z} + (x^* + c + 3)
\end{aligned}$$

for all $z \ge 1$.

We now bound $p_k = \Pr[|\{i | i \le I'(T) \wedge q_i \ge x^* + c + 1\}| = k]$, i.e. the probability that k items with qualities higher than $x^* + c + 1$ are explored during the T rounds. We note that this can happen if in at least one round an agent manages to get k coin flips to land tails before stopping the exploration. Every agent achieves this with probability at most $(1 - \Phi(1/\sigma))^k \le e^{-\frac{k}{2\sigma^2}}$ and thus for all T rounds, the probability that k items with value higher than $x^* + c + 1$ are explored is at most $T \cdot e^{-\frac{k}{2\sigma^2}}$ by a union bound. We thus get that:

$$\begin{aligned}
&\mathbb{E}[\max(0, \max_{i \le I'(T)} q_i)] \\
\le &\sqrt{1-\sigma^2} \cdot \left(\sqrt{2\ln z} + T \sum_{k \ge z} e^{-\frac{k}{2\sigma^2}} \sqrt{2\ln k} \right) + (x^* + c + 3)
\end{aligned}$$

Setting $z = 7(1 + \sigma^2 \ln T)$ we have that

$$\begin{aligned}
T \sum_{k \ge z} e^{-\frac{k}{2\sigma^2}} \sqrt{2\ln k} &\le e^{-7} e^{\frac{z}{7\sigma^2}} \sum_{k \ge z} e^{-\frac{k}{2\sigma^2}} \sqrt{2\ln k} \\
&\le e^{-7} e^{\frac{z}{7}} \sum_{k \ge z} e^{-\frac{k}{2}} \sqrt{2\ln k} \le e^{-7} e^{\frac{z}{7}} e^{-\frac{z}{4}} \le e^{-7}
\end{aligned}$$

and we obtain

$$\mathbb{E}[\max(0, \max_{i \le I'(T)} q_i)] \le \sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln T)} + x^* + c + 5.$$

□

Finally, by combining Lemma B.3, B.4 and B.5, we prove Proposition 4.5.

Proof of Proposition 4.5: Let's first prove the upper bound on $A(\sigma, T)$. By Lemma B.3 and Lemma B.5, we have

$$\begin{aligned}
A(\sigma, T) &\le M(\sigma, T) + O_c(1) \\
&\le \sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln(T))} + O_c(1)
\end{aligned}$$

Now let's prove the lower bound on $A(\sigma, T)$. Notice that if $T \le \exp(13/\sigma^2)$,

$$\sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln(T))} \le \sqrt{2\ln(1+13)} = O(1),$$

and therefore the lower bound is trivial. We only need to prove the lower bound when $T > \exp(13/\sigma^2)$.

By Lemma B.3 and Lemma B.4, we have

$$\begin{aligned}
A(\sigma, T) &\ge \left(1 - \frac{1}{\sqrt{T}}\right) (M(\sigma, \sqrt{T}) - c) \\
&\ge \left(1 - \frac{1}{\sqrt{T}}\right) \left(\sqrt{1-\sigma^2} \cdot \sqrt{2\ln\left(\sigma^2 \ln\left(\sqrt{T}\right)\right)} - O(1) - c \right) \\
&\ge \sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln(T)) - 2} - O_c(1) \\
&\quad - \frac{\sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln(T))}}{\sqrt{T}} \\
&\ge \sqrt{1-\sigma^2} \cdot \sqrt{2\ln(1+\sigma^2 \ln(T))} - O_c(1).
\end{aligned}$$

□

B.3 Omitted Proofs of Section 5

Lemma B.7 (Restatement of Lemma 5.2). *$A(1, T)$ is at least $(1 - o(1))D$ for $T = \omega(1/p)$.*

Proof. When diversity level is $\sigma = 1$, we have $q_i = d_i$. So q_i's realized value is either 0 or D. For each explored item, if it has quality 0, its Weitzman index will be lower than the Weitzman index of an unexplored item. This means that before an item with quality D is explored, each agent will explore at least one new item. Therefore, at round t, the probability of having seen a diamond is at least $1 - (1-p)^t$. Once a diamond has been found, each successive agent will get utility at least $D - c$. Therefore, for any t, $A(1, T) \ge (1 - (1-p)^t)(1 - t/T)(D - c)$.

We want to pick $t = \omega(1/p)$ and $t/T = o(1)$. Choosing $t = \sqrt{T \cdot (1/p)}$ suffices. We get $A(1, T) \ge (1 - o(1))(1 - o(1))(D - c) = (1 - o(1))D$. □

Lemma B.8 (Restatement of Lemma 5.3). *$A(0, T)$ is at most $D \cdot p + O(1)$ in the diamond model.*

Proof. The proof plan is similar to the proof of Lemma 4.1. We first bound by a constant the Weitzman index x^* of an unsearched item, i.e. one whose quality is not yet revealed. We then argue that once some item with sufficient quality has been identified (with quality $q_i > x^* + c$), no other item will be explored in future rounds. We conclude the proof by upper-bounding the expected quality of such an item conditional on its quality being higher than $x^* + c$. This directly gives a bound for the average utility.

We start by the Weitzman index of an unsearched item. Let x^* be its Weitzman index. Let $X \sim \mathcal{N}(0,1)$, by the defintion of Weitzman index, $c = p \cdot \mathbb{E}[\max(X + D - x^*, 0)] + (1-p) \cdot \mathbb{E}[\max(X - x^*, 0)]$.

We give upper bounds on $\mathbb{E}[\max(X + D - x^*, 0)]$ and $\mathbb{E}[\max(X - x^*, 0)]$ separately. For $\mathbb{E}[\max(X + D - x^*, 0)]$, since $x^* > 0$, we have

$$\begin{aligned}
&\mathbb{E}[\max(X + D - x^*, 0)] \le \mathbb{E}[\max(X + D, 0)] \\
=&\mathbb{E}[X + D] + (\mathbb{E}[-X | -X \ge D] - D) \cdot \Pr[-X \ge D] \\
=&D + \left(\frac{\varphi(D)}{1 - \Phi(D)} - D\right)(1 - \Phi(D)) \\
\le&D + \left(\frac{D}{2} + \frac{\sqrt{D^2+4}}{2} - D\right) \cdot \frac{1}{\sqrt{2\pi}\exp(D^2/2)D} \\
\le&D + \frac{1}{\sqrt{2\pi}\exp(D^2/2)D^2}.
\end{aligned}$$

For $\mathbb{E}[\max(X - x^*, 0)]$, we have

$$\begin{aligned}
&\mathbb{E}[\max(X - x^*, 0)] = \mathbb{E}[X - x^* | X - x^* > 0] \cdot Pr[X - x^* > 0] \\
= &\left(-x^* + \frac{\varphi(x^*)}{\Phi(-x^*)}\right) \cdot \Phi(-x^*) = (-x^*)(1 - \Phi(x^*)) + \varphi(x^*) \\
\le &(-x^*)\left(\frac{\varphi(x^*)}{x^*} - \frac{\varphi(x^*)}{(x^*)^3}\right) + \varphi(x^*) = \frac{\varphi(x^*)}{(x^*)^2} \\
= &\frac{1}{\sqrt{2\pi} \cdot (x^*)^2 \cdot \exp((x^*)^2/2)}.
\end{aligned}$$

To sum up we get $c \le p \cdot D + \frac{p}{\sqrt{2\pi}\exp(D^2/2)D^2} + \frac{1-p}{\sqrt{2\pi}\cdot(x^*)^2\cdot\exp((x^*)^2/2)}$. For notation convenience, define $\alpha = \frac{c - p\cdot D - \frac{p}{\sqrt{2\pi}\exp(D^2/2)D^2}}{1-p}$. We have $\sqrt{2\pi} \cdot (x^*)^2 \cdot \exp((x^*)^2/2) \le \frac{1}{\alpha}$. As $c - p \cdot D = \Omega(1)$ and $D = \omega(1)$, we have $\alpha = \Omega(1)$. This implies an upper bound on the Weitzman index of an unsearched item: $x^* \le \max\left(1, \sqrt{\ln\left(\frac{1}{2\pi\alpha^2}\right)}\right) = O(1)$.

When an item has been searched with observed quality q_i, since the diversity level is 0, the values of this item for future agents are determinstically q_i. Therefore the Weitzman index of this item is simply $q_i - c$.

Once some item has quality $q_i > c + x^*$, its Weitzman index is larger than x^* which means it will be searched before any unsearched items. And once this item is searched in a round, we know its realized value is q_i which is larger than x^*. Therefore no more unsearched items will be searched.

Let Q be the quality of the first item with quality larger than $c + x^*$. We have $\mathbb{E}[Q] = \mathbb{E}[q_i | q_i > c + x^*]$. We know that starting from the round this item is searched, each agent's utility is at most $Q - c$ and before this item is ever searched, each agent's utility is at most $c + x^* - c \le Q - c$. Therefore we can bound the average utility as following:

$$\begin{aligned}
A(0,T) &\le \mathbb{E}[Q - c] \le \mathbb{E}[q_i | q_i > c + x^*] - c \\
&\le D \cdot p + \frac{\varphi(c + x^*)}{1 - \Phi(c + x^*)} - c \\
&\le D \cdot p + \frac{x^* - c + \sqrt{(x^* + c)^2 + 4}}{2} \\
&\le D \cdot p + x^* + 1 \text{ (by Fact A.2)} \\
&= D \cdot p + O(1).
\end{aligned}$$

□

We proceed to show Lemma B.9 which is the main technical part behind the proof of Theorem 5.1 and gives a bound on the expected maximum quality of the items explored in T rounds, i.e. $M(\sigma, T)$. This implies an upper bound on the expected average utility $A(\sigma, T)$ using an identical argument to Lemma B.3. We omit the proof because of space limit. It is very similar to the proof of Lemma B.5.

Lemma B.9. *For any diversity level $0 < \sigma < 1$, in the diamond model,*

$$\begin{aligned}
&A(\sigma, T) \le M(\sigma, T) \\
\le&\sqrt{1 - \sigma^2} \cdot \sqrt{2\ln(1 + \sigma^2 \ln(T))} + \frac{pD(2\sigma^2 \ln(D) + 2\sigma^2 \ln(T))}{1 - \Phi\left(\frac{x^* + c + 1}{\sqrt{1-\sigma^2}}\right)} + O_c(1).
\end{aligned}$$

where x^* is the Weitzman index of an unsearched item.

Proof of Theorem 5.1: Given Lemma B.9, we can now complete the proof of Theorem 5.1. We know that x^* (the Weitzman index of an unsearched item) is constant as shown in the proof of Lemma 5.3. So for a constant $\sigma < 1$, we know that $\frac{x^* + c + 1}{\sqrt{1-\sigma^2}}$ is also a constant and therefore $\Phi\left(\frac{x^* + c + 1}{\sqrt{1-\sigma^2}}\right)$ is bounded away from 1. Plugging $T = \exp(o(D))$ into Lemma B.9, we get that $A(\sigma, T) = o(D)$. As $A(1, T) = (1 - o(1))D$ by Lemma 5.2, the theorem follows. □

REFERENCES

[ADLO11] Daron Acemoglu, Munther A. Dahleh, Ilan Lobel, and Asuman Ozdaglar. Bayesian learning in social networks. *Review of Economic Studies*, 78(4):1201–1236, 2011.

[AMMO17] Daron Acemoglu, Ali Makhdoumi, Azarakhsh Malekian, and Asuman Ozdaglar. Fast and slow learning from reviews. Working Paper 24046, National Bureau of Economic Research, November 2017.

[ASGM17] Pantelis P. Analytis, Hrvoje Stojic, Alexandros Gelastopoulos, and Mehdi Moussaïd. Diversity of preferences can increase collective welfare in sequential exploration problems. *CoRR*, abs/1703.10970, 2017.

[Ask10] John Asker. A study of the internal organization of a bidding cartel. *American Economic Review*, 100(3):724–62, June 2010.

[Ban92] Abhijit V. Banerjee. A simple model of herd behavior. *QUART. J. ECONOM*, 107(3):797–818, 1992.

[Bar08] Árpád Baricz. Mills' ratio: Monotonicity patterns and functional inequalities. *Journal of Mathematical Analysis and Applications*, 340(2):1362 – 1370, 2008.

[BBK18] Hamsa Bastani, Mohsen Bayati, and Khashayar Khosravi. Mostly exploration-free algorithms for contextual bandits. *CoRR arXiv:1704.09011*, 2018. Working paper.

[BHW92] Sushil Bikhchandani, David Hirshleifer, and Ivo Welch. A theory of fads, fashion, custom, and cultural change in informational cascades. *Journal of Political Economy*, 100(5):992–1026, 1992.

[BKM16] Bart J. Bronnenberg, Jun B. Kim, and Carl F. Mela. Zooming in on choice: How do consumers search for cameras online? *Marketing Science*, 35(5):693–712, 2016.

[CH15] Yeon-Koo Che and Johannes Horner. Optimal design for social learning. Levine's bibliography, UCLA Department of Economics, 2015.

[FKKK14] Peter Frazier, David Kempe, Jon Kleinberg, and Robert Kleinberg. Incentivizing exploration. In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, EC '14, pages 5–22, New York, NY, USA, 2014. ACM.

[GPR06] Jacob K. Goeree, Thomas R. Palfrey, and Brian W. Rogers. Social learning with private and common values. *Economic Theory*, 28(2):245–264, Jun 2006.

[HC17] Elisabeth Honka and Pradeep Chintagunta. Simultaneous or sequential? search strategies in the u.s. auto insurance industry. *Marketing Science*, 36(1):21–42, 2017.

[HKQ15] Li Han, David Kempe, and Ruixin Qiang. Incentivizing exploration with heterogeneous value of money. In Evangelos Markakis and Guido Schäfer, editors, *Web and Internet Economics*, pages 370–383, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[KAB10] Jun B. Kim, Paulo Albuquerque, and Bart J. Bronnenberg. Online demand under limited consumer search. *Marketing Science*, 29(6):1001–1023, 2010.

[KKM+17] Sampath Kannan, Michael Kearns, Jamie Morgenstern, Mallesh Pai, Aaron Roth, Rakesh Vohra, and Zhiwei Steven Wu. Fairness incentives for myopic agents. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, EC '17, pages 369–386, New York, NY, USA, 2017. ACM.

[KMP13] Ilan Kremer, Yishay Mansour, and Motty Perry. Implementing the "wisdom of the crowd". In *Proceedings of the Fourteenth ACM Conference on Electronic Commerce*, EC '13, pages 605–606, New York, NY, USA, 2013. ACM.

[KMR+18] Sampath Kannan, Jamie Morgenstern, Aaron Roth, Bo Waggoner, and Zhiwei Steven Wu. A smoothed analysis of the greedy algorithm for the linear contextual bandit problem. 2018.

[Kra11] Elena Krasnokutskaya. Identification and estimation of auction models with unobserved heterogeneity. *The Review of Economic Studies*, 78(1):293–327, 2011.

[LS15a] Ilan Lobel and Evan Sadler. Information diffusion in networks through social learning. *Theoretical Economics*, 10(3):807–851, 2015.

[LS15b] Ilan Lobel and Evan Sadler. Preferences, homophily, and social learning. *Operations Research*, 64(3):564–584, 2015.

[LV98] Tong Li and Quang Vuong. Nonparametric estimation of the measurement error model using multiple indicators. *Journal of Multivariate Analysis*, 65(2):139 – 165, 1998.

[MSS15] Yishay Mansour, Aleksandrs Slivkins, and Vasilis Syrgkanis. Bayesian incentive-compatible bandit exploration. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, EC '15, pages 565–582, New York, NY, USA, 2015. ACM.

[MSSW16] Yishay Mansour, Aleksandrs Slivkins, Vasilis Syrgkanis, and Zhiwei Steven Wu. Bayesian exploration: Incentivizing exploration in bayesian games. In *Proceedings of the 2016 ACM Conference on Economics and Computation, EC '16, Maastricht, The Netherlands, July 24-28, 2016*, page 661, 2016.

[PBS18] Yiangos Papanastasiou, Kostas Bimpikis, and Nicos Savva. Crowdsourcing exploration. *Management Science*, 64(4):1727–1746, 2018.

[RSVW18] Manish Raghavan, Aleksandrs Slivkins, Jennifer Wortman Vaughan, and Zhiwei Steven Wu. The externalities of exploration and how data diversity helps exploitation. pages 1724–1738, 2018.

[SR17] S. Schmit and C. Riquelme. Human Interaction with Recommendation Systems: On Bias and Exploration. *ArXiv e-prints*, March 2017.

[STZ17] Vasilis Syrgkanis, Elie Tamer, and Juba Ziani. Inference on auctions with weak assumptions on information. *CoRR*, abs/1710.03830, 2017.

[Wei79] Martin Weitzman. Optimal search for the best alternative. *Econometrica*, 47(3):641–54, 1979.

Alleviating Users' Pain of Waiting: Effective Task Grouping for Online-to-Offline Food Delivery Services

Shenggong Ji
Southwest Jiaotong University, China
shenggongji@163.com

Yu Zheng*
JD Urban Computing Business Unit, China
msyuzheng@outlook.com

Zhaoyuan Wang
Southwest Jiaotong University, China
wang_zhaoyuan@foxmail.com

Tianrui Li*
Southwest Jiaotong University, China
trli@swjtu.edu.cn

ABSTRACT

Ordering take-out food (a.k.a. takeaway food) on online-to-offline (O2O) food ordering and delivery platforms is becoming a new lifestyle for people living in big cities, thanks to its great convenience. Web users and mobile device users can order take-out food (i.e. obtain *online food ordering services*) on an O2O platform. Then the O2O platform will dispatch food carriers to deliver food from restaurants to users, i.e. providing users with *offline food delivery services*. For an O2O food ordering and delivery platform, improving food delivery efficiency, given the massive number of food orders each day and the limited number of food carriers, is of paramount importance to reducing the length of time users wait for their food. Thus, in this paper, we study the food delivery task grouping problem so as to improve food delivery efficiency and alleviate the pain of waiting for users, which to the best of our knowledge has not been studied yet. However, the food delivery task grouping problem is challenging, given two reasons. First, the food delivery efficiency is affected by multiple factors, which are non-trivial to formulate and jointly consider. Second, the problem is a typical NP-hard problem and to find near-optimal grouping results is not easy. To address these two issues, we propose an effective task grouping method. On one hand, we provide formal formulations for the factors affecting the food delivery efficiency, and provide an objective to organically combine these factors such that it can better guide the task grouping. On the other hand, we propose heuristic algorithms to efficiently obtain effective task grouping results, consisting of a greedy algorithm and a replacement algorithm. We evaluate our task grouping method using take-out food order data from web users and mobile device users on a real-world O2O food ordering and delivery platform. Experiment results demonstrate that our task grouping method can save ~16% (87 seconds) of average waiting time for each user, comparing with many baseline methods. It indicates that our method is able to significantly improve the food delivery efficiency and can provide better food delivery services for users.

*Yu Zheng and Tianrui Li are the correspondence authors of this paper. Yu Zheng is also with JD Intelligent City Research, China, and Xidian University, China.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313464

CCS CONCEPTS

• **Information systems → Spatial-temporal systems**; **Data mining**; • **Human-centered computing → Ubiquitous and mobile computing**;

KEYWORDS

Mobile applications, O2O food delivery services, task grouping, graph edge partition, optimization

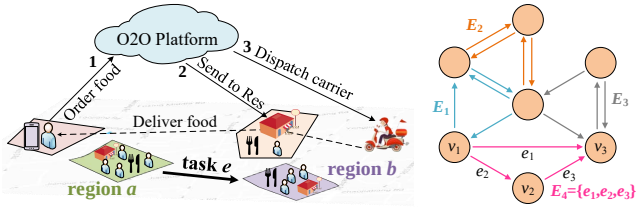
ACM Reference Format:
Shenggong Ji, Yu Zheng, Zhaoyuan Wang, and Tianrui Li. 2019. Alleviating Users' Pain of Waiting: Effective Task Grouping for Online-to-Offline Food Delivery Services. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313464

1 INTRODUCTION

Advance in modern web and mobile device technologies is spurring the prevalence of ordering take-out food (takeaway food) on online-to-offline (O2O) food ordering and delivery platforms (O2O platforms for short). As shown in Figure 1(a), web users or mobile device users can order take-out food on an O2O platform (step 1), through a web or an App. After receiving the food orders, the O2O platform will send them to corresponding restaurants (step 2), and will dispatch food carriers to deliver the food from restaurants to users (step 3), i.e. providing offline food delivery services for users. Thus, users are not necessary to go to restaurants in person, just waiting and enjoying the delivery of food from food carriers. Thanks to its great convenience, there are massive people ordering take-out food on O2O platforms every day. For example, in China, the number of daily food orders from users on one O2O platform has already reached 10 million [25, 26]. As a result, for O2O food ordering and delivery platforms, given the limited number of food carriers and the massive number of users each day, how to improve the food delivery efficiency such that the waiting time of the massive users can be reduced has become an urgent problem.

A feasible way to improve the food delivery efficiency is grouping delivery tasks in a city into groups (i.e. the task grouping problem). One delivery task is not one specific food order from a user, but all food orders from a region to another. For the example in Figure 1(a), there is a delivery task e from region a to region b, since there exist users in region b ordering food from restaurants in region a. A region contains many users and restaurants. If we see regions as vertices and delivery tasks as directed edges, we can construct

a delivery task graph, like the one in Figure 1(b). In Figure 1(b), tasks (edges) are grouped into four task groups, i.e. E_1, E_2, E_3 and E_4, which are highlighted using different colors. A food carrier will be assigned to only one task group such that he/she can be very familiar with the locations of users and restaurants in the task group. In other words, by task grouping, food carriers in each task group can avoid wasting time on finding the locations of users and restaurants, and the routes between them, thus improving the food delivery efficiency and reducing users' waiting time.



(a) Online-to-Offline Food Delivery Services (b) Task Graph

Figure 1: Food delivery task grouping problem.

It should be noted that the task grouping problem is not a real-time problem. For an O2O food ordering and delivery platform, delivery tasks are grouped just from time to time, e.g. once per month. For example, the task grouping result of this month can be obtained based on users' food orders in the last few months. Having the task grouping result, we can allocate food carriers to each task group and the food orders (arriving in real time) can be delivered by food carriers in corresponding task group. Clearly, *the food carrier allocation problem* (i.e. how many food carriers are needed for each task group?) and *the real-time food carrier dispatching problem* (i.e. which food carrier should be dispatched when a food order arrives in real time?) also significantly affect the food delivery efficiency. In this work, we focus on the task grouping problem since it is the foundation of the allocation and dispatching problem. These two problems will be studied in our future work.

In fact, our task grouping problem can be seen as a graph edge partition problem, i.e. a graph vertex-cut problem [5, 17, 30]. However, previous graph edge partition algorithms (heuristic algorithms mainly) are designed for distributed graph computation platforms [5, 10, 11], whose aim is to minimize the number of total replicated vertices over different edge groups. Therefore, due to the different scenarios and the different objectives, previous graph edge partition algorithms cannot work well to our task grouping problem. As a consequence, a new task grouping method is in great need for the O2O food ordering and delivery platforms.

However, our task grouping problem is challenging, due to the following two reasons.

First, to improve the food delivery efficiency, our task grouping should consider multiple factors, e.g. the *shareability* and the *empty run time* of each task group. Besides the joint consideration of these two factors, to mathematically formulate and derive the shareability and the empty run time is also non-trivial. Specifically, the *shareability* indicates to what extent food orders in a task group can be shared (jointly delivered). The *empty run time* refers to the empty cruising time before carriers deliver next food orders. For example, the task group E_4 marked by the pink color in Figure 1(b) may have high shareability, since carriers can jointly deliver orders in task e_1 together with orders in task e_2 or e_3, saving the total delivery time, if these orders come at close time. However, the task group has high empty run time. For instance, after a food carrier delivers an order to region v_3, if the next food order is in task e_1 (or e_2), the food carrier needs to go back to region v_1, *in empty*, to deliver the food order. Apparently, to improve the food delivery efficiency, we expect task groups with *high* shareability and *low* empty run time.

Second, the task grouping problem can be easily proved to be a NP-hard problem [5, 17, 30]. As a result, it is also non-trivial to design an effective and efficient algorithm to obtain near-optimal task grouping solutions.

To address these issues, in this paper, we propose a method to do the task grouping for O2O food ordering and delivery platforms. Specifically, our contributions are as below.

- To the best of our knowledge, we are the first to study the delivery task grouping problem for O2O food ordering and delivery platforms in real world. We formally define the food delivery task grouping problem.
- We provide formal formulations for the shareability and the empty run time. Besides, we propose a method to organically incorporate the shareability and the empty run time into one objective such that it can better guide our task grouping.
- We propose effective heuristic algorithms to solve the task grouping problem, including a greedy algorithm and a replacement algorithm. The greedy algorithm is to output a fast and good task grouping solution, while the replacement algorithm is to further refine the obtained grouping solution.
- We evaluate our task grouping method using the food order data from web users and mobile device users on an O2O food ordering and delivery platform in real world. Compared with previous graph edge partition methods, our task grouping method is able to save at least 16% (87 seconds) of average travel time for each food order, significantly reducing the waiting time of users.

2 OVERVIEW

2.1 Preliminary

Definition 1 (*Region*): A region v is a small piece of a city, containing some restaurants and users. Note that considering each individual restaurant (or user) is not necessary, given that some restaurants (or users) are geographically very close and thus can be seen as a unity. A city is comprised of many regions and we denote all regions in a city by V. Regions V in a city are obtained by region segmentation using road networks, which will be detailed later in Section 3.1. An example for the obtained regions V in a city can be found in Figure 8(b).

Definition 2 (*Delivery Task*): A delivery task, denoted by $e = (o, d, n^1, \cdots, n^P)$, is a directed edge from region o to region d, containing all food orders from restaurants of region o to users of region d. $n^p, p = 1, \cdots, P$ denotes the number of food orders in the delivery task in time period p, as demonstrated in Figure 2. A period can be with 15 minutes. P is the number of total time periods under consideration for the task grouping (e.g. one month). We can denote all delivery tasks in a city by E. The extraction of all delivery

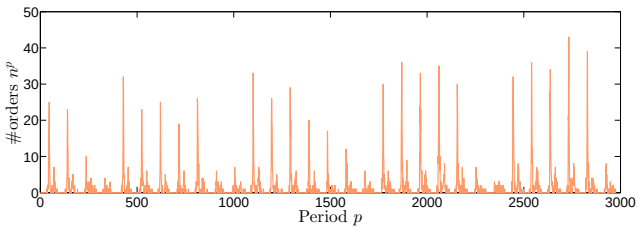
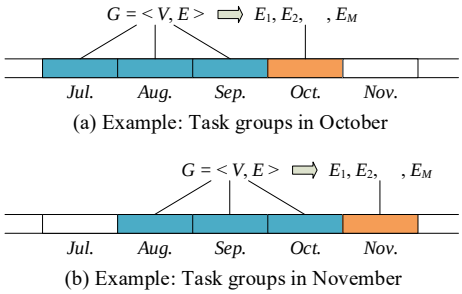


Figure 2: An example for a food delivery task's $n^1, \cdots, n^P$.



(a) Example: Task groups in October

(b) Example: Task groups in November

Figure 3: Examples for task grouping.

Different with the edges on distributed graph computation platforms [5, 10, 11], our edges (tasks) contain clear spatial and temporal properties. Specifically, o and d refer to the spatial locations of a task, while $n^1, \cdots, n^P$ denote the temporal frequency of food orders (from web users and mobile device users) from region o to region d (Figure 2). Thus, the task grouping problem in our scenario can be more difficult.

Definition 3 (*Task Graph*): A task graph is a directed graph, denoted by $G = \langle V, E \rangle$, with all regions V as vertices and all delivery tasks E as directed edges. G is naturally obtained once V and E have been extracted.

2.2 Problem Definition

M-task grouping problem: given a constructed task graph $G = \langle V, E \rangle$ and the number M of task groups, the task grouping problem aims to group all delivery tasks in E into M groups, i.e. $E_m, m = 1, \cdots, M$, such that the following conditions (or objectives) can be satisfied:

(1) Each task group E_m is a subset of E, i.e. $E_m \subset E, \forall m$;
(2) All tasks are grouped into the M groups, i.e. $\bigcup_{m=1}^{M} E_m = E$;
(3) Task groups do not overlap, i.e. $E_{m_1} \cap E_{m_2} = \emptyset, \forall m_1 \neq m_2$;
(4) Each task group E_m has high shareability;
(5) Each task group E_m has low empty run time.

As mentioned in the Introduction, our task grouping is not conducted in real time, but just from time to time. For the examples in Figure 3, the task grouping is conducted every month. Actually, since the distribution of food orders won't change dramatically, the task grouping can even be conducted every quarter/half of a year. In Figure 3(a), the task groups in October are obtained based on the task graph constructed using food order data in the previous three months, i.e. July, August and September. The obtained task groups are applied in October, i.e. each arriving food order are delivered by a food carrier in the corresponding task group. That is, the food carrier allocation and real-time dispatching (defined in the Introduction) in October are based on the obtained task groups. Similarly, in Figure 3(b), at the beginning of November, the task grouping is re-conducted based on the task graph constructed using food order data in August, September and October.

Our task grouping problem is actually a graph edge partition problem, which has already been proved to be NP-hard [2, 30]. As mentioned before, due to the spatial and temporal properties of the edges (tasks) in our task graph and the different objectives 4 and 5, our task grouping problem can be more difficult.

2.3 Framework

Figure 4 demonstrates the framework of our task grouping method, consisting of three components: pre-processing, shareability and empty run time, and task grouping algorithm. Below, we introduce each component in detail.

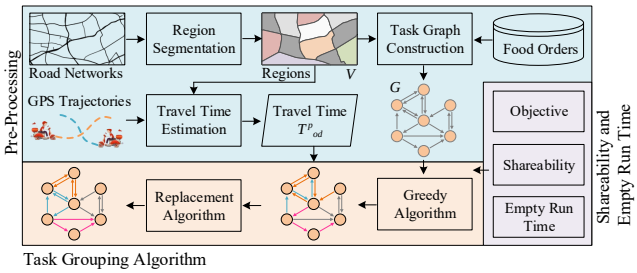


Figure 4: The framework of our task grouping method.

As shown in Figure 4, the pre-processing component has three small components: region segmentation, task graph construction, and travel time estimation. Firstly, the region segmentation is to segment a city into non-overlapping regions V. In detail, a city is segmented based on the road structure of the city (i.e. road networks), e.g. Figure 8(a, b). Next, with the segmented regions V, we construct the task graph $G = \langle V, E \rangle$. The construction process uses the historical food order records collected from web users and mobile device users in real world. From these food order records, we can know whether there are food orders from one region to another and the number of the food orders in each time period. That is, all delivery tasks E between any two regions can be extracted. Then, based on V and E, task graph G can be constructed. The constructed task graph G will be transferred to the task grouping algorithm component for grouping. The third small component is the travel time estimation, which is to estimate the travel time of food carriers between any two regions in V in each time period p. The estimation is based on the GPS trajectories of food carriers collected from real world. The estimated travel time are also necessary in the task grouping algorithm component.

The shareability and empty run time component is to formally formulate the shareability and the empty run time, which are then combined as one objective. The objective will be used to guide the grouping of tasks. In general, given a grouping result, we utilize the total delivery time of food orders to indicate the shareability of

the grouping result. The smaller the total delivery time, the higher shareability the grouping result. Thus, our objective is formulated as the sum of the delivery time and the empty run time. The delivery time and the empty run time of a grouping result will consider the number of food orders in each task in each period.

The task grouping algorithm component details our algorithms for grouping tasks into M task groups, including a greedy algorithm and a replacement algorithm. The greedy algorithm assigns tasks to each task group E_m in a greedy manner based on the objective, and will output a task grouping result $E_1, \cdots, E_m$. The replacement algorithm is to further improve the grouping result of the greedy algorithm. Specifically, we try to replace the owner (one task group) of a task with another task group, with the expectation that the total delivery time and the total empty run time can be reduced after the replacement. The replacement algorithm stops when the objective cannot be improved any more, and then our ultimate task grouping result is returned.

3 PRE-PROCESSING

3.1 Region Segmentation

In this subsection, we segment a city into non-overlapping regions V, which includes two steps. At the first step, we segment the city into regions according to the structure of road networks, using the region segmentation method proposed in [29]. By this method, the segmented regions have semantic meanings, e.g. a region can be a small block of a city. This is an important advantage of this segmentation method over the grid-based method [9, 18]. However, since road networks in a city are too dense, some regions obtained from the first step are very small, e.g. the widths and lengths of some regions are less than 200 meters. Therefore, at the second step, we recursively merge those small and geographically adjacent regions into relatively big regions, until the size of the smallest region reaches a threshold (e.g. 400 meters × 400 meters).

3.2 Task Graph Construction

With segmented regions V, we can construct the task graph based on users' food order records over a long time in real world. Specifically, for each food order (see the data format in the Section 6), we can get the location of the restaurant and the location of the user, which can be mapped into regions. For example, if the restaurant is in region o and the user is in region d, we have a task $e = (o, d, n^1, \cdots, n^P)$. Iterating over the entire food order records, we can obtain all tasks E. Besides the location information, we also have the time stamp at which a food order is ordered, which can be mapped to a corresponding time period. Thus, for each task $e \in E$, we can get the number of food orders in each period p, i.e. the $e.n^p$. Finally, based on the constructed V and E, we can then obtain the task graph $G = \langle V, E \rangle$ in a city.

3.3 Travel Time Estimation

In this subsection, we estimate the travel time $T^p_{o,d}$ of food carriers traveling from region $o \in V$ to region $d \in V$ in each period p, using the GPS trajectories of food carriers in real world. Since a period is too short (e.g. 15 minutes), we estimate the travel time from o to d in each hour, based on which we can get $T^p_{o,d}$. In detail, our estimation process has the following steps. First, we detect and remove stay points from the GPS trajectories of food carriers, using the stay-point detection algorithm proposed in [32]. In real world, food carriers may stay still to wait for users to fetch their food, or to wait for restaurants to prepare the food. Thus, we need to remove these stay points, before estimating the travel time of food carriers. Second, we map the GPS trajectories (without stay points) onto road segments, using the map-matching algorithm proposed in [28]. After the map-matching, we can obtain the travel time of carriers on each road segment. Note that there may exist some missing data, we can fill them based on the spatial and temporal correlation of travel time between road segments [4, 6, 24, 31]. Finally, the travel time between any two regions can be estimated as the minimum travel time of a path (a sequence of road segments), concatenating the centers of these two regions [24]. Note that the estimated travel time is necessary for the calculation of the shareability and the empty run time, when we conduct the task grouping.

4 SHAREABILITY AND EMPTY RUN TIME

4.1 Objective

The shareability of a grouping result $E_1, \cdots, E_M$ can be indicated by the total delivery time needed to deliver all food orders in all tasks. The smaller the total delivery time, the higher the shareability. For a simple example in Figure 5, there are four tasks. Tasks e_1, e_2 can be easily shared since their origins and destinations are both highly close, and food orders in e_1, e_2 have a similar arrival pattern (Figure 5(c)). That is, food orders in e_1, e_2 that come in the same time period can be delivered jointly, using just around one share of delivery time. So can tasks e_3, e_4. Thus, if we group e_1, e_2 as a task group and e_3, e_4 as another group, as Figure 5(a), the shareability of the grouping is high. In the meantime, the total delivery time to deliver all food orders is small, since jointly delivering food orders in e_1, e_2 (or e_3, e_4) can save a lot of delivery time. In contrast, as Figure 5(b), if we group e_1, e_3 as a group and e_2, e_4 as another group, the shareability of the grouping is low. Meanwhile, the total delivery time is large, since jointly delivering food orders in e_1, e_3 (or e_2, e_4) cannot save the total delivery time. Therefore, we can use the total delivery time to indicate the shareability. It should be noted that for a task group, in addition to the tasks' origins and destinations, the shareability is also affected by tasks' number of food orders in each time period (Figure 5(c)). The formal formulation of the shareability is in Section 4.2.

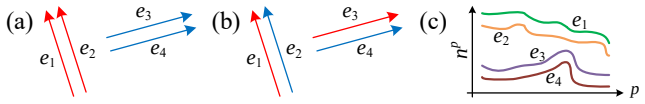


Figure 5: An example for the shareability.

The total delivery time for delivering all food orders is denoted by $DT(E_1, \cdots, E_M)$ and is actually the aggregation of delivery time in each task group E_m, i.e.

$$DT(E_1, \cdots, E_M) = \sum_{m=1}^{M} DT(E_m), \tag{1}$$

where $DT(E_m)$ refers to the total delivery time needed to deliver food orders in task group E_m.

The total empty run time of a grouping result $E_1, \cdots, E_M$ is denoted by $ET(E_1, \cdots, E_M)$, which is the aggregation of empty run time in each task group E_m:

$$ET(E_1, \cdots, E_M) = \sum_{m=1}^{M} ET(E_m), \tag{2}$$

where $ET(E_m)$ denotes the empty run time of task group E_m.

For our task grouping problem, we expect a grouping result with smaller delivery time (i.e. higher shareability) and smaller empty run time. Therefore, we can combine them as one objective, i.e. the total travel time, which is the sum of the total delivery time and the total empty run time. Thus, the total travel time $TT(E_1, \cdots, E_M)$ is

$$TT(E_1, \cdots, E_M) = \sum_{m=1}^{M} TT(E_m), \tag{3}$$

where $TT(E_m)$ is the aggregation of $DT(E_m)$ and $ET(E_m)$:

$$TT(E_m) = DT(E_m) + ET(E_m). \tag{4}$$

To obtain our objective function in Equation 3, we need to mathematically formulate the delivery time (shareability) $DT(E_m)$ and the empty run time $ET(E_m)$ in each task group E_m. Thus, for the rest of this section, we detail how to mathematically formulate the delivery time $DT(E_m)$ and the empty run time $ET(E_m)$ for each task group m.

4.2 Delivery Time $DT(E_m)$ (Shareability)

When calculating the delivery time of food orders in a task group E_m, we will consider jointly delivering (sharing) food orders, such that the delivery time can denote the shareability. However, in real world, only food orders that arrive within a time period (e.g. 15 minutes) can be shared, since jointly delivering food orders in different time periods would let users wait too long for their food. That is, jointly delivering food in different time periods is not user-friendly and will lead to poor user experience. Thus, the $DT(E_m)$ is actually the aggregation of the delivery time of food orders in each time period p. That is, $DT(E_m)$ is

$$DT(E_m) = \sum_{p=1}^{P} DT(E_m, p), \tag{5}$$

where P denotes the number of total time periods, and $DT(E_m, p)$ refers to the delivery time of food orders in task group E_m in period p. A period considered in this paper is with 15 minutes.

For food orders in a task group E_m in a period p, there may exist many strategies to jointly deliver them. For example, as shown in Figure 6, there are three tasks in task group E_m, i.e. e_1, e_2, e_3. In period p, e_1 is with one order, e_2 with two and e_3 with one. The possible joint delivery solutions include: 1) jointly delivering the order in e_1 with an order in e_2 and jointly delivering another order in e_2 with the order in e_3; 2) jointly delivering the order in e_1 with the order in e_3 and jointly delivering the two orders in e_2; etc. The problem now is to find a delivery solution such that it can be used for the calculation of our needed delivery time.

Definition 4 (*Delivery Time*): The delivery time $DT(E_m, p)$ is defined as the minimum delivery time among all possible delivery

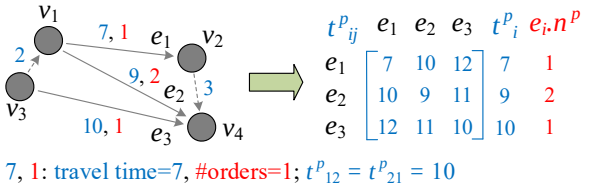


Figure 6: A simple example for delivery time $DT(E_m, p)$.

solutions such that it can indicate the shareability. As a consequence, to obtain the $DT(E_m, p)$, we need to find the optimal delivery solution. In math, it is

$$DT(E_m, p) = \min_{y_i, x_{ij}} \sum_{i=1}^{|E_m|} \left(y_i \cdot t_i^p + \sum_{j=i}^{|E_m|} x_{ij} \cdot t_{ij}^p \right)$$
$$s.t. \begin{cases} 1 \cdot y_i + \left(2 \cdot x_{ii} + 1 \cdot \sum_{j=1, j \neq i}^{|E_m|} x_{ij}\right) = e_i.n^p, \forall i \\ x_{ij} = x_{ji}, \forall i, j \\ y_i, x_{ij} \in \mathbb{Z}^+ \cup \{0\}, \forall i, j \end{cases} \tag{6}$$

y_i and x_{ij} are the decision variables. y_i refers to the number of food orders in task e_i being delivered *individually*. x_{ij} refers to the number of food orders in task e_i and e_j that are delivered jointly. t_i^p denotes the travel time needed for delivering an order in task e_i, i.e. $t_i^p = T^p_{e_i.o, e_i.d}$. For example, in Figure 6, $t_1^p = T^p_{v_1, v_2} = 7$. t_{ij}^p is the minimum travel time needed to jointly deliver an order in e_i and an order in e_j, and $t_{ij}^p = t_{ji}^p$ holds. For example, in Figure 6, $t_{12}^p = t_{21}^p = 10$ means that the minimum travel time to jointly deliver an order in e_1 and an order in e_2 is 10 minutes (by taking the route $v_1 \rightarrow v_2 \rightarrow v_4$, i.e. picking up two orders in region v_1 and then delivering the one to region v_2 and another one to v_4). $e_i.n^p$ denotes the number of orders in e_i in period p. $|E_m|$ denotes the number of tasks in group E_m.

In this optimization problem, the objective is to find the optimal delivery solution, i.e. y_i and x_{ij}, with the minimum delivery time. The first constraint requires that all orders in each task e_i should be delivered. Note that if $x_{ii} = 1$, two orders in task e_i are delivered (denoted by $2 \cdot x_{ii}$). The second constraint indicates that the sharing of orders between two tasks is symmetric. The last constraint means that y_i, x_{ij} are nonnegative integers.

From the optimization problem 6, we can find that the shareability of a task group is affected both by tasks' origins and destinations and by tasks' number of food orders in a time period, as aforementioned. The more tasks having close origins and destinations, the higher the shareability. The more food orders that come at the same time period, the higher the shareability.

The optimization problem can be efficiently solved using some off-the-shelf optimization solvers, e.g. Gurobi Optimizer [16], Microsoft Solver Foundation [13], etc. For the example in Figure 6, the solution is $x_{21} = x_{12} = 1, x_{23} = x_{32} = 1$ and thus $DT(E_m, p) = (T^p_{v_1, v_2} + T^p_{v_2, v_4}) + (T^p_{v_3, v_1} + T^p_{v_1, v_4}) = 21$. That is, the optimal delivery solution is to jointly deliver an order in e_2 with the order in e_1 (taking route $v_1 \rightarrow v_2 \rightarrow v_4$) and to jointly deliver another order in e_2 with the order in e_3 (taking route $v_3 \rightarrow v_1 \rightarrow v_4$).

4.3 Empty Run Time $ET(E_m)$

The empty run time for task group E_m is denoted by $ET(E_m)$. Similar to $DT(E_m)$, $ET(E_m)$ is also the aggregation of the empty run time $ET(E_m, p)$ in each period p, i.e.

$$ET(E_m) = \sum_{p=1}^{P} ET(E_m, p). \tag{7}$$

The empty run time $ET(E_m, p)$ is the total time intervals between the delivery units in period p.

Definition 5 (*Delivery Unit*): A delivery unit contains one food order in task e_1^s and one food order in e_2^s, and is denoted by $du = (e_1^s, e_2^s, o, d)$, where o and d denote the origin and destination of the delivery unit, respectively. For the example in Figure 6, the solution for $DT(E_m, p)$ contains two delivery units. The first unit is to jointly deliver an order in e_1 and e_2 by taking the route $v_1 \to v_2 \to v_4$. Therefore, $du_1 = (e_1, e_2, v_1, v_4)$. The second unit is to jointly deliver an order in e_2 and e_3 by taking the route $v_3 \to v_1 \to v_4$, that is, $du_2 = (e_2, e_3, v_3, v_4)$. Note that if a delivery unit contains only one food order, e_2^s is null and we denote $e_2^s = *$.

After a carrier finishes a delivery unit du_i, to deliver next delivery unit du_j, the carrier needs to spend some time traveling from du_i's destination $du_i.d$ to du_j's origin $du_j.o$, *in empty* (without carring any food). That is, the travel time from $du_i.d$ to $du_j.o$ is the empty run time. For example, as the first row in Figure 7, from delivery unit du_1 to du_2, there is empty run time $T^p_{v_4,v_3}$.

Definition 6 (*Empty Run Time*): Given the last delivery unit du_0 in period $p-1$ and the n delivery units $du_1, \cdots, du_n$ obtained from the solution for $DT(E_m, p)$ in this period p, the empty run time $ET(E_m, p)$ is the minimum time needed to travel between delivery units $du_1, \cdots, du_n$, starting from du_0. In math, it is

$$ET(E_m, p) = \min_{(i_1, i_2, \cdots, i_n) \in \text{Perm}(1,2,\cdots,n)} \sum_{j=0}^{n} T^p_{du_{i_j}.d, du_{i_{j+1}}.o} \tag{8}$$

where $\text{Perm}(1, 2, \cdots, n)$ denotes all permutations of set $\{1, 2, \cdots, n\}$ and $i_0 = 0$. $T^p_{du_{i_j}.d, du_{i_{j+1}}.o}$ is the travel time from du_{i_j}'s destination $du_{i_j}.d$ to $du_{i_{j+1}}$'s origin $du_{i_{j+1}}.o$, i.e. the empty run time between the i_j-th delivery unit and i_{j+1}-th delivery unit.

In summary, the delivery time $DT(E_m, p)$ is the total travel time of all delivery units in period p, while the empty run time $ET(E_m, p)$ is the minimum travel time needed to travel between all delivery units. For example, for the delivery units obtained in Figure 6, i.e. the $du_1 = (e_1, e_2, v_1, v_4)$ and $du_2 = (e_2, e_3, v_3, v_4)$ in Figure 7, $DT(E_m, p) = (T^p_{v_1,v_2} + T^p_{v_2,v_4}) + (T^p_{v_3,v_1} + T^p_{v_1,v_4})$. Assume that the last delivery unit in period $p-1$ is $du_0 = (e_1, *, v_1, v_2)$. We have $\text{Perm}(1, 2) = \{(1, 2), (2, 1)\}$, corresponding to the two rows in Figure 7. If the optimal solution is $(i_1, i_2) = (2, 1)$, i.e. the second row, then $ET(E_m, p) = T^p_{v_2,v_3} + T^p_{v_4,v_1}$.

The optimization problem 8 is a NP-hard problem. In fact, the problem can be seen as a traditional traveling salesman problem [19], which is to find the shortest route to visit a given list of cities. Specifically, a delivery unit in our problem can be seen as a city in the traveling salesman problem and the distance between two cities is the travel time from the destination of a delivery unit to the origin of another delivery unit. Thus, to solve the NP-hard optimization problem 8, we can apply an efficient and effective nearest neighbour

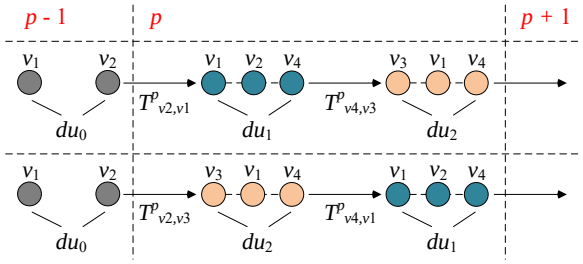


Figure 7: A simple example for empty run time $ET(E_m, p)$.

algorithm [7, 19, 20], which has been widely used in the traveling salesman problem. Specifically, assuming the current delivery unit is du_i, the algorithm selects the nearest (in terms of travel time) unvisited delivery unit du_j as the next delivery unit. That is, we select du_j such that

$$T^p_{du_i.d, du_j.o} \le T^p_{du_i.d, du_k.o}, \text{for each unvisited } du_k.$$

In this way, starting from du_0, we can obtain the solution for the optimization problem 8 efficiently and the empty run time $ET(E_m, p)$. The time complexity is $O(n)$, where n is the number of delivery units in period p.

An assumption underlying the above formulation is that there exists only one carrier in each task group E_m. This is due to that when conducting task grouping, we don't know the number of carriers in each task group E_m. The allocation of carriers to task groups is after the task grouping. Actually, if we are given the number of carriers in each task group, the problem can be formulated as a multiple travelling salesman problem [1] and corresponding empty run time can be similarly calculated.

5 TASK GROUPING ALGORITHM

5.1 Greedy Algorithm

The greedy algorithm is demonstrated in Algorithm 1. In general, the greedy algorithm is comprised of M rounds (Line 3-11) and at each round m, we assign tasks to task group E_m. Below, we detail the processes of assigning tasks to E_m at each round m.

Algorithm 1 Greedy Algorithm

```
1:  procedure GREEDY(E, M)
2:      E^u = E                                   ▷ Initial unassigned tasks
3:      for m = 1, ··· , M
4:          E_m = ∅
5:          e_* ← a random task from E^u
6:          E_m = E_m ∪ {e_*},  E^u = E^u \ {e_*}
7:          while |E_m| < α|E|/M or E^u ∩ Nei(E_m) ≠ ∅
8:              for e ∈ E^u ∩ Nei(E_m)
9:                  Compute MTR(e|E_m)            ▷ Equation 9
10:             e_** = arg min_{e∈E^u∩Nei(E_m)} MTR(e|E_m)
11:             E_m = E_m ∪ {e_**}, E^u = E^u \ {e_**}
12:     return E_1, ··· , E_M
```

Initially, E_m is an empty set and is assigned with a random task e_* selected from unassigned tasks E^u (Line 3-6). E^u is initialized

as E (Line 2) and tasks assigned will be removed from E^u. *Second*, we search each unassigned task e from E_m's neighborhood tasks $Nei(E_m)$ and compute the *marginal travel time ratio* $MTR(e|E_m)$ (Equation 9) of assigning e to E_m (Line 8-9). *Third*, the task e_{**} with the minimum marginal time ratio is assigned to E_m (Line 10-11), making E_m be with high shareability and low empty run time. We repeat the Step 2 and Step 3, until the number of tasks in E_m surpasses $\alpha|E|/M$ or there does not exist unassigned task nearby E_m (Line 7). The first stop criterion is to balance the size of each task group (e.g. $\alpha = 1.1$). The reason that we only consider the neighborhood tasks of E_m is that usually tasks not in $Nei(E_m)$ have much bigger marginal travel time ratios than tasks in $Nei(E_m)$. Thus, tasks not in $Nei(E_m)$ can be pruned and the total running time can be largely reduced. Below, we introduce the definitions of the marginal travel time ratio and the neighborhood tasks.

Definition 7 (*Marginal Travel Time Ratio*): For task group E_m and a task e, the marginal travel time ratio of assigning task e to task group E_m is defined as:

$$MTR(e|E_m) = \frac{TT(\{e\} \cup E_m) - TT(E_m)}{TT(\{e\})}, \tag{9}$$

where $TT(E_m)$ denotes the total travel time needed (including delivery time and empty run time) to deliver food orders in task group E_m as defined in Equation 4. Similarly, $TT(\{e\} \cup E_m)$ denotes the total travel time for delivering food orders in tasks $\{e\} \cup E_m$. That is, $MTR(e|E_m)$ is the marginal travel time of food orders after assigning e to E_m, divided by the original travel time needed to deliver food orders in task e. Apparently, to minimize the total travel time (our objective), we assign the task with the minimum $MTR(e|E_m)$ to the task group E_m each time, i.e. Line 10-11 in Algorithm 1.

Definition 8 (*Neighborhood Task*): A task e is a neighborhood task of task group E_m, if both e's origin $e.o$ and e's destination $e.d$ are geographically close to E_m, i.e.

$$T^p_{e.o,E_m} \le t_{nei} \text{ and } T^p_{e.d,E_m} \le t_{nei}.$$

$T^p_{e.o,E_m}$ denotes the minimum of the travel time from e's origin $e.o$ to regions in E_m. t_{nei} is a given threshold, e.g. 5 minutes.

5.2 Replacement Algorithm

The replacement algorithm, shown in Algorithm 2, is to improve the grouping result of the greedy algorithm. The main idea of the replacement algorithm is to replace the owner of a task from one task group with another task group, with the expectation that the total travel time can be reduced (our objective). The replacement algorithm stops (Line 3) when the number of continuous failed replacement attempts n_{cur} reaches the threshold n_{th} (e.g. $n_{th} = 100$). For each replacement process (Line 4-11), we have the following four steps.

First, we randomly select a task e from a random task group E_{m_1} and we remove e from E_{m_1} (Line 4-5). *Second*, for each task group E_m, we compute the *marginal travel time* $MT(e|E_m)$ (Equation 10) if assigning task e to group E_m (Line 6-7). *Third*, we find the task group E_{m^*} with the minimum marginal travel time, and assign task e to group E_{m^*} (Line 8-9). *Fourth*, if the task group E_{m^*} is exactly the task group E_{m_1}, this replacement process fails, since the total travel time is not reduced. As a result, the number of continuous failed attempts increases, i.e. $n_{cur} = n_{cur} + 1$ (Line 10). If the task group

Algorithm 2 Replacement Algorithm

```
1:  procedure REPLACE(E_1, ··· , E_M)
2:      n_cur = 0                                   ▷ Initial failed times
3:      while n_cur < n_th
4:          e ← a random task from a random E_{m_1}
5:          E_{m_1} = E_{m_1} \ {e}
6:          for m = 1, ··· , M
7:              Compute MT(e|E_m)                   ▷ Equation 10
8:          m* = arg min_{m=1,···,M} MT(e|E_m)
9:          E_{m*} = E_{m*} ∪ {e}
10:         if m* = m_1: n_cur = n_cur + 1
11:         else: n_cur = 0
12:     return E_1, ··· , E_M
```

E_{m^*} is not the task group E_{m_1}, the replacement process succeeds and the total travel time has been reduced successfully. Thus, we reset the n_{cur} as $n_{cur} = 0$ (Line 11).

Definition 9 (*Marginal Travel Time*): The marginal travel time $MT(e|E_m)$ of assigning e to E_m is defined as:

$$MT(e|E_m) = TT(\{e\} \cup E_m) - TT(E_m), \tag{10}$$

where $TT(E_m)$ denotes the total travel time for task group E_m as defined in Equation 4. Thus, selecting the task group E_{m^*} with the minimum travel time for the task e is helpful for minimizing the total travel time (our objective).

6 EVALUATION

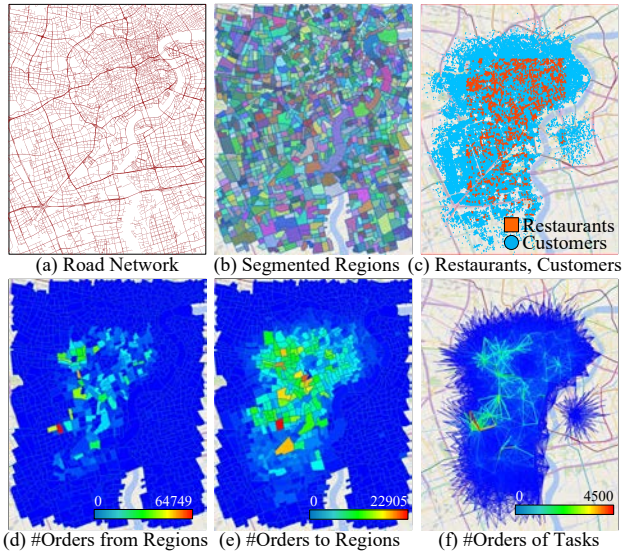
6.1 Datasets and Pre-processing Results

To evaluate our task grouping method, we use datasets collected from a real-world O2O food ordering and delivery platform. Specifically, our datasets include road network data, food order record data, and food carriers' GPS trajectory data. Below, we detail these datasets and our pre-processing results.

6.1.1 Road network. We use the road network data in city of Shanghai, China. The network contains 333,766 intersections and 440,922 road segments, as shown in Figure 8(a). As detailed before, our region segmentation is based on the road network. Specifically, using the road network, we segment the area in Figure 8(a) into regions, and after merging small and adjacent regions, we obtain 1,000 regions (i.e. $|V| = 1000$), as demonstrated in Figure 8(b). Besides, the road network data is also needed when we estimate the travel time of food carriers between any two regions.

6.1.2 Food order records. We collect one month of food order record data from users in this area. The food order records start from December 1 to December 31, 2016. For each food order record, we have the following information: 1) the time stamp at which the user ordered the food; 2) the restaurant information (ID, latitude, longitude); 3) the user information (ID, latitude, longitude); 4) the ID of the food carrier delivering the food order; and etc.

Temporal distribution of food orders. In total, our food order record dataset contains 1,852,439 food orders in total. In average, each day is with around 59,756 orders. Note that a time period is with 15 minutes, thus we have 2,976 time periods for the 31 days of food order records, i.e. $P = 2976$. The number of food orders in



(a) Road Network (b) Segmented Regions (c) Restaurants, Customers

(d) #Orders from Regions (e) #Orders to Regions (f) #Orders of Tasks

Figure 8: Regions, tasks, and food orders in an area of Shanghai city, China.

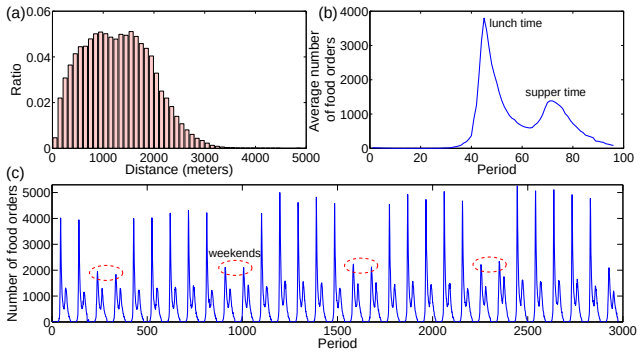


Figure 9: Statistics of food orders from real-world users.

each time period is demonstrated in Figure 9(c), and Figure 9(b) presents the average number of food orders in each period of a day. Clearly, the number of food orders has a clear daily pattern, i.e. there exist two peaks (a big one and a small one) each day. The big peak appears during the lunch time while the small peak appears during the supper time. On the other hand, there also exists a weekly pattern, i.e. weekends have much less food orders than weekdays, as shown in Figure 9(c). It implies that people more often order take-out food during their working hours, especially at noon on weekdays.

Spatial distribution of food orders. Together with the segmented regions V, the food order record data can be used to construct the task graph. Based on the food order records, using the method proposed in Section 3.2, we extract 2,980 tasks, i.e. $|E| = 2980$. And then we can construct the task graph $G = \langle V, E \rangle$, based on the obtained V and E.

The tasks (edges) are demonstrated in Figure 8(f). Task (edges) are denoted by lines in different colors based on the number of food orders in each task. That is, different colors correspond to different numbers of food orders, as the color bar shows. For example, a task denoted by red color has around 4500 food orders in the 31 days, while a task in blue color has only a few food orders.

Similar to the Figure 8(f), we can also draw the spatial distribution of food orders in each region, as shown in Figure 8(d) and (e). Specifically, Figure 8(d) demonstrates the number of food orders from each region (restaurants), while Figure 8(e) shows the number of food orders to each region (users).

Figure 8(c) shows the geographical locations of restaurants and users. Figure 9(a) presents the distribution of the geographical distances between restaurants and users. As depicted in the figure, for most food orders, the distances between restaurants and users are within 3,000 meters. This is due to that the O2O platform usually recommends restaurants within 3 kilometers to users on its web and App.

6.1.3 GPS trajectories of food carriers. The GPS trajectories of food carriers are also collected in December 2016. For each GPS record, we have a food carrier ID, a latitude, a longitude, and a time stamp. In total, there are 4,774 food carriers with 115,602,018 GPS records collected. Based on the GPS trajectory data, using the method proposed in Section 3.3, we can estimate the travel time of food carriers traveling between any two regions in V, in each time period p.

6.2 Metrics

To evaluate our task grouping method, we use four metrics, which are the average delivery time $\overline{DT}$, the average empty run time $\overline{ET}$, the average travel time $\overline{TT}$ for each food order, and the average number of regions that each task group covers. Formally, the four metrics can be formulated as

$$\overline{DT} = \frac{DT}{n_{total}}, \quad \overline{ET} = \frac{ET}{n_{total}}, \quad \overline{TT} = \frac{TT}{n_{total}}, \quad \overline{\#R} \tag{11}$$

where DT, ET and TT correspond to the total delivery time in Equation 1, the total empty run time in Equation 2 and the total travel time in Equation 3, respectively. n_{total} denotes the number of total food orders in all tasks. The smaller the first three metrics, the better a grouping result. For a task grouping result with a smaller $\overline{\#R}$, carriers in a task group can be more familiar with the locations of users and restaurants. Thus, the smaller the $\overline{\#R}$, the better.

6.3 Baselines

To demonstrate the effectiveness of our task grouping method, we compare our method with many state-of-the-art baseline methods, including NE [30], DBH [27], HDRF [17], RAND [5], and Oblivious [5, 17]. Although these baseline methods are originally designed for undirected graph edge partition problems for distributed graph computation platforms [5, 10, 11], they can be used or can be slightly modified to partition directed graph edges for our task grouping problem.

6.4 Effectiveness

We conduct extensive experiments to compare our task grouping method with the above baselines. Specifically, we use the task graph G constructed from our datasets and set $M = 300, 400, 500,$

Table 1: Comparisons with baselines (unit for $\overline{DT}, \overline{ET}, \overline{TT}$: seconds).

Task Grouping Method	$M = 300$				$M = 400$				$M = 500$			
	$\overline{DT}$	$\overline{ET}$	$\overline{TT}$	$\overline{\#R}$	$\overline{DT}$	$\overline{ET}$	$\overline{TT}$	$\overline{\#R}$	$\overline{DT}$	$\overline{ET}$	$\overline{TT}$	$\overline{\#R}$
NE	275	260	535	9.9	277	264	541	7.9	279	268	547	6.6
DBH	276	294	570	8.8	276	280	556	6.6	276	276	552	5.4
HDRF	283	267	550	7.3	285	264	549	6.0	285	258	543	5.1
RAND	292	611	903	18.4	292	603	895	13.9	292	606	898	11.3
Oblivious	284	261	545	7.5	285	252	537	5.9	285	250	535	4.9
Greedy (ours)	270	191	461	8.8	273	198	471	6.8	276	200	476	6.3
Greedy+Replace (ours)	268	180	448	8.5	271	185	456	7.0	273	186	459	6.0

i.e. grouping all tasks into 300, 400, 500 groups. To better evaluate the performance of the greedy algorithm and the replacement algorithm in our method, we first consider only the greedy algorithm (Greedy) and then add the replacement algorithm (Greedy+Replace). Experiment results are summarized in Table 1.

In terms of the first three metrics, our task grouping method defeats all baseline methods. The best baseline method is the NE method [30]. The improvement of our method over baseline methods is significant. For example, when $M = 300$, our greedy algorithm (461 seconds) has already defeated the baseline algorithms by at least 74 seconds (comparing with 535 seconds of NE method). If combined with the replacement algorithm, our method needs just 448 seconds for each food order, saving 87 seconds for each order. That is, our method can save at least 16% of average travel time for each order, comparing with the baseline methods, i.e. significantly reducing the waiting time of users.

In terms of the average number $\overline{\#R}$ of regions covered by each task group, each task group of our method covers less than 9 regions, which is a proper value. From Table 1, we can find that the smaller the M, the bigger the $\overline{\#R}$. Thus, to make carriers familiar with the locations of users and restaurants in a task group, grouping more task groups (a relatively bigger M) is better.

In the meantime, we can find that the average delivery time, empty run time, and travel time will slightly increase with the increasing of M. This is due to that the larger the M, the smaller the size of each task group, and the more difficult to improve shareability and to reduce empty run time. However, this does not mean a smaller M is better since we expect a bigger M to reduce $\overline{\#R}$. Thus, the selection of M should be a trade-off between $\overline{TT}$ and the $\overline{\#R}$ for O2O platforms in real world.

6.5 Time Efficiency

In this subsection, we study the time efficiency of our task grouping method. Our task grouping method is realized in C# and is performed on a single computer with 4 Intel Xeon E3-1225 3.31 GHz cores and 8 GB RAM. The running time of our task grouping method is presented in Figure 10. We can see that our task grouping method is efficient, using just at most 1.5 hours. It should be noted that given a city, the task grouping is only conducted from time to time, e.g. once per month or even per quarter/half of a year, as mentioned in the Introduction. Besides, our task grouping method can easily run in parallel. When more machines are used, the running time can be significantly reduced. Thus, our task grouping method is efficient enough for the applications of real-world O2O food ordering and delivery platforms.

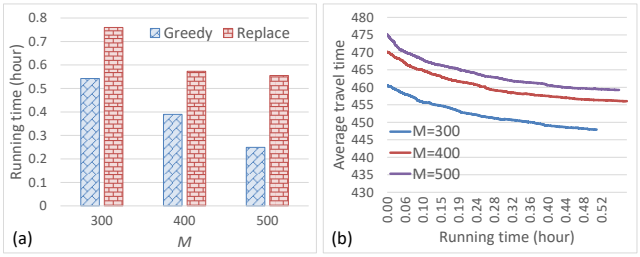


Figure 10: Running time of our task grouping method.

6.6 Balance of Task Grouping Results

The balance of each task group is also an important index. The amounts of tasks, regions, and food orders in different task groups are expected to be balancing. To this end, for a task grouping result, we use entropy to indicate the balance of tasks, regions and food orders in each task group. More specifically, we define the entropy of tasks for a task grouping result as $Ent^t = -\sum_{m=1}^{M} p_m^t \log_2 p_m^t$. p_m^t denotes the ratio of tasks in group m, i.e. the number of tasks in group m divided by the total number of tasks. Similarly, in terms of regions and food orders, we can define $Ent^r = -\sum_{m=1}^{M} p_m^r \log_2 p_m^r$ and $Ent^o = -\sum_{m=1}^{M} p_m^o \log_2 p_m^o$. p_m^r and p_m^o refer to the ratio of regions and the ratio of food orders in group m, respectively. Entropies of different task grouping methods are presented in Figure 11. As shown in the figure, our method, together with HDRF, RAND and Oblivious, achieves balancing task grouping results, with entropies approaching the optimal entropy $\log_2 300 \approx 0.829$.

7 RELATED WORK

7.1 Previous Research on Take-out Food

Previous research on take-out food (takeaway food) is conducted mainly from the socioeconomic perspective [14, 15, 23] or from the perspective of human health [21, 22]. In general, previous research mainly focuses on the influence of take-out food consumption on health. For example, in [15], Miura et al. studied the relationship between the consumption of take-out food with the socioeconomic differences in fruit and vegetable intake. Based on their analysis, they found that less educated people are more likely to consume

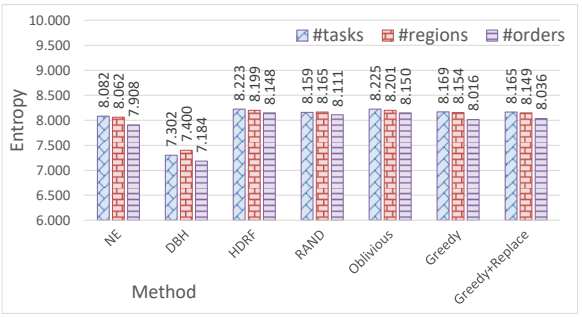


Figure 11: Entropies of tasks, regions, and orders. $M = 300$.

less healthy take-out food, containing fewer fruit and vegetables, and that consuming less healthy take-out food may result in the socioeconomic differences in fruit and vegetable intake. For another example, in literature [22], Smith et al. investigated the relationship between the take-out food consumption with diet quality and abdominal obesity. This work concluded that consuming take-out food twice a week or more may lead to poor diet quality and moderate abdominal obesity in young men and women.

Different with the above work, our work aims to improve the take-out food delivery efficiency for O2O take-out food ordering and delivery platforms, from the perspective of computer science, using data-driven methods. Consuming take-out food frequently may lead to some health concerns, however more and more healthy take-out food can be ordered. Since massive food orders are issued each day, to improve the food delivery efficiency is essential for O2O platforms. To the best of our knowledge, we are the first to study the take-out food from this perspective.

7.2 Graph Edge Partition

From the perspective of the problem formulation, our task grouping problem is formulated as a graph edge partition problem (a.k.a. graph vertex-cut problem) [2, 5, 12, 17, 27, 30]. Previous graph edge partition problems are mainly studied for large-scale distributed graph computation platforms, e.g. PowerGraph [5], GraphLab [10], Pregel [11]. Specifically, they partition a large-scale computation graph to different machines and each machine runs a part of the graph in parallel, so as to speed up the whole computation process. Their objective is to minimize the number of replicated vertices in different machines so as to lower the communication cost between machines. Many heuristic algorithms [5, 12, 17, 27, 30] have been proposed in recent years for graph edge partition problems. For example, in [30], Zhang et al. proposed a neighborhood expansion-based heuristic algorithm to do the graph edge partition. In [5], Gonzalez et al. proposed a random algorithm and some greedy algorithms. In [17], Petroni et al. presented a high-degree replicated first algorithm. Besides the graph edge partition problems, graph vertex grouping problems (a.k.a. graph edge-cut problems) [2, 3, 8] were also widely studied, which is to partition vertices (instead of edges) in a graph into a given number of groups.

As discussed before, although our task grouping problem is a graph edge partition problem, due to the different scenarios and objectives, previous graph edge partition algorithms cannot apply well to our problem. Actually, our experiment results have also validated that our proposed task grouping algorithm can achieve much better performance than the previous graph edge partition algorithms (see Table 1).

8 CONCLUSION AND FUTURE WORK

In this paper, we proposed a task grouping method for O2O take-out food ordering and delivery platforms, consisting of two main contents. First, we formulated the factors affecting the food delivery efficiency and provided a combined objective to guide our task grouping. Second, we presented effective heuristic algorithms for grouping tasks, including a greedy algorithm and a replacement algorithm. Using our task grouping method, the food delivery efficiency can be highly improved. According to the experiment results, our method is able to group tasks into task groups with high shareability (i.e. low delivery time) and low empty run time. Comparing with the previous graph edge partition algorithms, our method is able to save at least 16% (87 seconds) of average travel time for each food order, which is a significant improvement. As a result, with our research, O2O take-out food ordering and delivery platforms can largely improve their food delivery efficiency, thus providing better food delivery services for users. In the future, as mentioned in the Introduction, we plan to study the food carrier allocation problem and the real-time food carrier dispatching problem. Note that these two problems also significantly affect the food delivery efficiency of O2O food ordering and delivery platforms.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant Nos. 61672399, U1609217, 61773324), the Fundamental Research Funds for the Central Universities (Nos. 22071004005040177), and Doctoral Innovation Fund Program of Southwest Jiaotong University.

REFERENCES

[1] Tolga Bektas. 2006. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega* 34 (2006), 209–219.
[2] Florian Bourse, Marc Lelarge, and Milan Vojnovic. 2014. Balanced graph edge partition. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*. 1456–1465.
[3] Aydin Buluç, Henning Meyerhenke, Ilya Safro, Peter Sanders, and Christian Schulz. 2016. Recent Advances in Graph Partitioning. In *Algorithm Engineering - Selected Results and Surveys*. 117–158.
[4] C. De Fabritiis, R. Ragona, and G. Valenti. 2008. Traffic estimation and prediction based on real time floating car data. In *Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems*.
[5] Joseph E. Gonzalez, Yucheng Low, Haijie Gu, Danny Bickson, and Carlos Guestrin. 2012. PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs. In *Proceedings of the 10th USENIX Symposium on Operating Systems Design and Implementation*. 17–30.
[6] E. Jenelius and H. N. Koutsopoulos. 2013. Travel time estimation for urban road networks using low frequency probe vehicle data. *Transportation Research Part B: Methodological* 53 (2013), 64–81.
[7] David S. Johnson and Lyle A. McGeoch. 1997. The Traveling Salesman Problem: A Case Study in Local Optimization. *Local Search in Combinatorial Optimisation* (1997), 215–310.
[8] George Karypis and Vipin Kumar. 1998. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. *SIAM J. Scientific Computing* 20, 1 (1998), 359–392.
[9] John Krumm and Eric Horvitz. 2007. Predestination: Where Do You Want to Go Today? *IEEE Computer* 40 (2007), 105–107.
[10] Yucheng Low, Joseph Gonzalez, Aapo Kyrola, Danny Bickson, Carlos Guestrin, and Joseph M. Hellerstein. 2012. Distributed GraphLab: A Framework for Machine Learning in the Cloud. *PVLDB* 5, 8 (2012), 716–727.

[11] Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik, James C. Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. 2010. Pregel: a system for large-scale graph processing. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD*. 135–146.

[12] Daniel W. Margo and Margo I. Seltzer. 2015. A Scalable Distributed Graph Partitioner. *PVLDB* 8, 12 (2015), 1478–1489.

[13] Microsoft. 2019. *Microsoft solver foundation reference manual.* https://msdn.microsoft.com/en-us/library/ff524499(v=vs.93).aspx.

[14] Kyoko Miura, Katrina Giskes, and Gavin Turrell. 2009. Socioeconomic differences in takeaway food consumption and their contribution to inequalities in dietary intakes. *Journal of Epidemiology & Community Health* 63, 10 (2009), 820–826.

[15] Kyoko Miura, Katrina Giskes, and Gavin Turrell. 2011. Contribution of take-out food consumption to socioeconomic differences in fruit and vegetable intake : a mediation analysis. *Journal of The American Dietetic Association* 111, 10 (2011), 1556–1562.

[16] Gurobi optimization. 2019. *Gurobi optimizer reference manual.* http://www.gurobi.com.

[17] Fabio Petroni, Leonardo Querzoni, Khuzaima Daudjee, Shahin Kamali, and Giorgio Iacoboni. 2015. HDRF: Stream-Based Partitioning for Power-Law Graphs. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015*. 243–252.

[18] Jason W. Powell, Yan Huang, Favyen Bastani, and Minhe Ji. 2011. Towards Reducing Taxicab Cruising Time Using Spatio-Temporal Profitability Maps. In *Advances in Spatial and Temporal Databases - 12th International Symposium, SSTD*. 242–260.

[19] César Rego, Dorabela Gamboa, Fred Glover, and Colin Osterman. 2011. Traveling salesman problem heuristics: Leading methods, implementations and latest advances. *European Journal of Operational Research* 211, 3 (2011), 427–441.

[20] Daniel J. Rosenkrantz, Richard Edwin Stearns, and Philip M. Lewis II. 1974. Approximate Algorithms for the Traveling Salesperson Problem. In *15th Annual Symposium on Switching and Automata Theory*. 33–42.

[21] Helmut Schröder, Montserrat Fíto, and Maria Isabel Covas. 2007. Association of fast food consumption with energy intake, diet quality, body mass index and the risk of obesity in a representative Mediterranean population. *British Journal of Nutrition* 98 (2007), 1274–1280.

[22] Kylie J Smith, Sarah A McNaughton, Seana L Gall, Leigh Blizzard, Terence Dwyer, and Alison J Venn. 2009. Takeaway food consumption and its associations with diet quality and abdominal obesity: a cross-sectional study of young adults. *International Journal of Behavioral Nutrition and Physical Activity* 6, 29 (2009).

[23] Gavin Turrell and Katrina Giskes. 2008. Socioeconomic disadvantage and the purchase of takeaway food: a multilevel analysis. *Appetite* 51, 1 (2008), 69–81.

[24] Yilun Wang, Yu Zheng, and Yexiang Xue. 2014. Travel time estimation of a path using sparse trajectories. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 25–34.

[25] Wikipedia. 2019. *Online take-out food ordering platform 1 in China.* https://en.wikipedia.org/wiki/Ele.me.

[26] Wikipedia. 2019. *Online take-out food ordering platform 2 in China.* https://en.wikipedia.org/wiki/Meituan-Dianping.

[27] Cong Xie, Ling Yan, Wu-Jun Li, and Zhihua Zhang. 2014. Distributed Power-law Graph Computing: Theoretical and Empirical Analysis. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014*. 1673–1681.

[28] Jing Yuan, Yu Zheng, Chengyang Zhang, Xing Xie, and Guangzhong Sun. 2010. An Interactive-Voting Based Map Matching Algorithm. In *Eleventh International Conference on Mobile Data Management, MDM*. 43–52.

[29] Nicholas Jing Yuan, Yu Zheng, and Xing Xie. 2012. Segmentation of Urban Areas Using Road Networks. *MSR-TR* 65 (July 2012).

[30] Chenzi Zhang, Fan Wei, Qin Liu, Zhihao Gavin Tang, and Zhenguo Li. 2017. Graph Edge Partitioning via Neighborhood Heuristic. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*. 605–614.

[31] Yu Zheng. 2015. Trajectory Data Mining: An Overview. *ACM Transaction on Intelligent Systems and Technology* 6, 3 (2015), 29:1–29:41.

[32] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. 2012. Mining interesting locations and travel sequences from GPS trajectories. In *Proceedings of the 18th International Conference on World Wide Web, WWW*. 17–30.

CommunityGAN: Community Detection with Generative Adversarial Nets

Yuting Jia, Qinqin Zhang, Weinan Zhang, Xinbing Wang
Shanghai Jiao Tong University
{hnxxjyt,zqq.lh.377,wnzhang,xwang8}@sjtu.edu.cn

ABSTRACT

Community detection refers to the task of discovering groups of vertices sharing similar properties or functions so as to understand the network data. With the recent development of deep learning, graph representation learning techniques are also utilized for community detection. However, the communities can only be inferred by applying clustering algorithms based on learned vertex embeddings. These general cluster algorithms like K-means and Gaussian Mixture Model cannot output much overlapped communities, which have been proved to be very common in many real-world networks. In this paper, we propose CommunityGAN, a novel community detection framework that jointly solves overlapping community detection and graph representation learning. First, unlike the embedding of conventional graph representation learning algorithms where the vector entry values have no specific meanings, the embedding of CommunityGAN indicates the membership strength of vertices to communities. Second, a specifically designed Generative Adversarial Net (GAN) is adopted to optimize such embedding. Through the minimax competition between the motif-level generator and discriminator, both of them can alternatively and iteratively boost their performance and finally output a better community structure. Extensive experiments on synthetic data and real-world tasks demonstrate that CommunityGAN achieves substantial community detection performance gains over the state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → **Clustering**; **Social networks**.

KEYWORDS

Community Detection; Graph Representation Learning; Generative Adversarial Nets

ACM Reference Format:
Yuting Jia, Qinqin Zhang, Weinan Zhang, Xinbing Wang. 2019. CommunityGAN: Community Detection with Generative Adversarial Nets. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313564

1 INTRODUCTION

Network is a powerful language to represent relational information among data objects from social, natural and academic domains [32]. One way to understand network is to identify and analyze groups of vertices which share highly similar properties or functions. Such groups of vertices can be users from the same organization in social networks [16], proteins with similar functionality in biochemical networks [12], and papers from the same scientific fields in citation networks [20]. The research task of discovering such groups is known as the community detection problem [36].

Graph representation learning, also known as network embedding, which aims to represent each vertex in a graph as a low-dimensional vector, has been consistently studied in recent years. The application of deep learning algorithms including Skip-gram [8, 19] and convolutional network [10] has improved the efficiency and performance of graph representation learning dramatically. Moreover, Generative Adversarial Nets (GAN) has also been introduced for learning better graph representation [25]. The learned vertex representation vectors can benefit a wide range of network analysis tasks including link prediction [6, 26], recommendation [34, 35], and node classification [23].

However, there still exists many limitations for the application of such embedding in overlapping community detection problems because of the dense overlapping of communities. Generally, the useful information of vertex embedding vectors is the relevant distance of these vectors, while the specific value in vertex embedding vectors has no meanings. Thus, given the representation vectors of the vertices, one has to adopt other algorithms like logistic regression to accomplish the real-world application tasks. To detect communities, a straightforward way is to run some clustering algorithms in the vector space. However, in some real-world datasets, one vertex may belong to tens of communities simultaneously [29, 30], while most clustering algorithms cannot handle such dense overlapping. In recent years, some researchers try to perform the community detection and network embedding simultaneously in a unified framework [3, 27] but still fail to solve the dense overlapping problem.

In this paper, we propose *CommunityGAN*, a novel community detection framework that jointly solves overlapping community detection and graph representation learning. The input-output overview of CommunityGAN is shown in Figure 1. With CommunityGAN, we aim to learn network embeddings like AGM (Affiliation Graph Model) through a specifically designed GAN. AGM [30, 31] is a framework which can model densely overlapping community structures. It assigns each vertex-community pair a nonnegative factor which represents the degree of membership of the vertex to the community. Thus, the strengths of membership from a vertex to all communities compose the representation vector of it.

Moreover, in recent years, motifs have been proved as essential features in community detection tasks [11]. Thus in this paper, unlike most previous works considering relationship between only

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313564

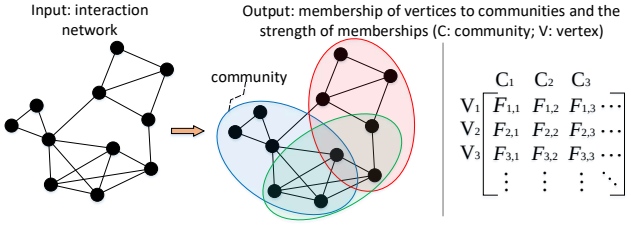


Figure 1: The input-output overview of CommunityGAN.

two vertices (the relationship between a center vertex and one of the other vertices in a window), we try to generate and discriminate motifs. Specifically, CommunityGAN trains two models during the learning process: 1) a generator $G(s|v_c)$, which tries to generate the most likely vertex subset s to compose a specified kind of motif; 2) a discriminator $D(s)$, which attempts to distinguish whether the vertex subset s is a real motif or not. In the proposed CommunityGAN, the generator G and the discriminator D act as two players in a minimax game. Competition in this game drives both of them to improve their capability until the generator is indistinguishable from the true connectivity distribution.

The contributions of our work are threefold:

- We combine AGM and GAN in a unified framework, which achieves both the outstanding performance of GAN and the direct vertex-community membership representation in AGM.
- We study the motif distribution among ground-truth communities and analyze how they can help improve the quality of detected communities.
- We propose a novel implementation for motif generation called *Graph AGM*, which can generate the most likely motifs with graph structure awareness in a computationally efficient way.

Empirically, two experiments were conducted on a series of synthetic data, and the results prove: 1) the ability of CommunityGAN to solve dense overlapping problem; 2) the efficacy of motif-level generation and discrimination. Additionally, to complement these experiments, we evaluate CommunityGAN on two real-world scenarios, i.e., community detection and clique prediction, using five real-world datasets. The experiment results show that CommunityGAN substantially outperforms the state-of-the-art methods in the field of community detection and graph representation learning. Specifically, CommunityGAN outperforms baselines by 7.9% to 21.0% in terms of F1-Score for community detection. Additionally, in 3-clique and 4-clique prediction tasks, CommunityGAN improves AUC score to at least 0.990 and 0.956 respectively. The superiority of CommunityGAN relies on its joint advantage of particular embedding design, adversarial learning and motif-level optimization.

2 RELATED WORK

Community Detection. Many community detection algorithms have been proposed from different perspectives. One direction is to design some measure of the quality of a community like modularity, and community structure can be uncovered by optimizing such measures [17, 28]. Another direction is to adopt the generative models to describe the generation of the graphs, and the communities can be inferred by fitting graphs to such models [9, 36]. Moreover, some models focus on the graph adjacency matrix and output the relationship between vertices and communities by adopting matrix factorization algorithms on the graph adjacency matrix [14, 31]. These models often consider the dense community overlapping problem and detect overlapping communities. However, the performance of these methods are restricted by performing pair reconstruction with bi-linear models.

Graph representation learning. In recent years, several graph representation learning algorithms have been proposed. For example, DeepWalk [19] shows the random walk in a graph is similar to the text sequence in natural language. Thus it adopts Skip-gram, a word representation learning model [15], to learn vertex representation. Node2vec [8] further extends the idea of DeepWalk by proposing a biased random walk algorithm, which provides more flexibility when generating the sampled vertex sequence. LINE [24] first learns the vertex representation preserving both the first and second order proximities. Thereafter, GraRep [2] applies different loss functions defined on graphs to capture different k-order proximities and the global structural properties of the graph. Moreover, GAN has also been introduced into graph representation learning. GraphGAN [25] proposes a unified adversarial learning framework, which naturally captures structural information from graphs to learn the graph representation. ANE [5] utilizes GAN as a regularizer for learning stable and robust feature extractor. However, all of the above algorithms focus on general graph representation learning. For community detection tasks, we have to adopt other clustering algorithms on vertex embeddings, which cannot handle the dense community overlapping problem. Compared to above mentioned methods, our CommunityGAN can output the membership of vertices to communities directly.

Unified framework for graph representation learning and community detection. In recent years, some unified frameworks for both graph representation learning and community detection have been proposed. Wang et al. [27] developed a modularized nonnegative matrix factorization algorithm to preserve both vertex similarity and community structure. However, because of the complexity of matrix factorization, this model cannot be applied on many real-world graphs containing millions of vertices. Cavallari et al. [3] designed a framework that jointly solves community embedding, community detection and node embedding together. However, although the community structures and vertex embeddings are optimized together, the community structures are still inferred through a clustering algorithm (e.g., Gaussian Mixture Model) based on the vertex embedding, which means the dense community overlapping problem cannot be avoided.

3 EMPIRICAL OBSERVATION

In this section, data analysis of the relationship between motifs and communities is presented, which motivates the development of CommunityGAN. We first describe the network datasets with

Table 1: Dataset statistics. V: number of vertices, E: number of edges, T: number of 3-clique (triangle), C: number of communities, A: community memberships per vertex.

Dataset	V	E	T	C	A
LiveJournal	4.0M	34.9M	178M	310K	3.09
Orkut	3.1M	120M	528M	8.5M	95.93
Youtube	1.1M	3.0M	3.1M	30K	0.26
DBLP	0.43M	1.3M	2.2M	2.5K	2.57
Amazon	0.34M	0.93M	0.67M	49K	14.83

Table 2: The occurrence probability of cliques for vertices sampled from all vertices or from one community. R: from all vertices. C: from one community.

Dataset	2-Clique		3-Clique		4-Clique	
	R	C	R	C	R	C
LiveJournal	4E-6	0.80	2E-11	0.18	0	0.08
Orkut	2E-5	0.82	8E-11	0.18	0	0.06
Youtube	4E-6	0.73	0	0.10	0	0.02
DBLP	1E-5	0.52	1E-10	0.31	0	0.23
Amazon	6E-6	0.53	0	0.18	0	0.05

ground-truth communities and then present our empirical observations.

3.1 Datasets

We study the five networks proposed by Yang and Leskovec [30], where LiveJournal, Orkut and Youtube are all online social networks, while DBLP and Amazon are collaboration network and product network respectively. Table 1 provides the statistics of these networks.

3.2 Empirical Observations

Now we present our empirical observations by answering two critical questions. How do the communities contribute to the generation of motifs? What is the change in motif generation with communities overlapping?

First, we investigate the relationship between motif generation and community structure. For each time, we randomly select one community. Then we sample 2/3/4 vertices from this community and judge whether they could compose a motif or not. We repeat this process for billions of times and get the occurrence probability of motifs in one community. We compare this value with the motif occurrence probability in the whole network. Because, in this paper, we mostly focus on a particular kind of motifs (clique), we only demonstrate the occurrence probability of 2/3/4-vertex cliques. As shown in Table 2, the average occurrence probabilities of cliques for vertices in one community are much higher than that for vertices randomly selected from the whole network. This observation demonstrates that the occurrence of motifs is strongly correlated to the community structure.

Next, we study the influence of community overlapping on the generation of motifs. Yang and Leskovec [31] have conducted the

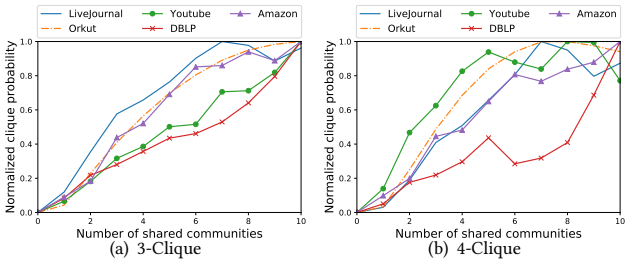


Figure 2: Normalized probability for vertices to be a clique as a function of number of shared communities. Probabilities are scaled so that maximum value is one.

relationship of 2-clique (edge) probability with community overlapping that the more communities two vertices have in common, the higher probability for them to be 2-clique. In this paper, we further study the influence of community overlapping on the generation of 3-clique and 4-clique. As shown in Figure 2, similar to 2-clique, the probability curve increases in the overall trend as the number of shared communities increases. Such observation accords with the base assumption of AGM framework that vertices residing in communities' overlaps are more densely connected to each other than the vertices in a single community. So we can extend the AGM from edge generation to clique generation, and the details will be explained in §4.4.

4 METHODOLOGY

Notation. In this paper, all vectors are column vectors and denoted by lower-case bold letters like $\mathbf{d}_v$ and $\mathbf{g}_v$. Calligraphic letters are used to represent sets like $\mathcal{V}$ and $\mathcal{E}$. And for simplicity, the corresponding normal letters are used to denote their size like $V = |\mathcal{V}|$. The details of notations in this paper are listed in Table 3.

In this section, we introduce the framework of CommunityGAN and discuss the implementation and optimization of the generator and discriminator in CommunityGAN.

4.1 CommunityGAN Framework

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{M})$ represent the studied graph, where $\mathcal{V} = \{v_1, \ldots, v_V\}$ is a set of vertices, $\mathcal{E} = \{e_{ij}\}_{i,j=1}^{V}$ is the set of edges among the vertices, and $\mathcal{M}$ is the set of motifs in graph $\mathcal{G}$. In this paper, notably, we only focus on a particular kind of motifs: cliques. Moreover, the D and G are also designed to discriminate and generate cliques. For a given vertex v_c, we define $M(v_c)$ as the set of motifs in the graph covering v_c, whose size is generally much smaller than the total number of vertices V. Moreover, we define conditional probability $p_{true}(m|v_c)$ as the preference distribution of motifs covering v_c over all other motifs in $\mathcal{M}$. Thus $M(v_c)$ can be seen as a set of observed motifs drawn from $p_{true}(m|v_c)$. Given the graph $\mathcal{G}$, we aim to learn the following two models: **Generator** $G(s|v_c; \theta_G)$, which tries to approximate $p_{true}(m|v_c)$, and generate (or select) subsets of vertices from $\mathcal{V}$ covering v_c most likely to be real motifs; and **Discriminator** $D(s, \theta_D)$, which aims to estimate the probability that a vertex subset s is a real motif, i.e., comes from $\mathcal{M}$.

Table 3: Notations used in this paper

Symbol	Description
$\mathcal{G}$	The studied graph
$\mathcal{V}$, $\mathcal{E}$, $\mathcal{M}$, $\mathcal{C}$	Set of vertices, edges, motifs, communities in graph $\mathcal{G}$
V, E, M, C	Number of vertices, edges, motifs, communities in graph $\mathcal{G}$
$M(v_c)$	Set of motifs covering v_c in graph $\mathcal{G}$
$N(v_c)$	Set of neighbors of vertex v_c in graph $\mathcal{G}$
s	The subset of vertices from $\mathcal{V}$
$p_{true}(m\|v_c)$	The preference distribution of motifs covering v_c over all other motifs in $\mathcal{M}$
$G(s\|v_c;\theta_G)$	Generator trying to generate vertex subsets from $\mathcal{V}$ covering v_c most likely to be real motifs
$D(s,\theta_D)$	Discriminator aiming to estimate the probability that a vertex subset s is a real motif
$\mathbf{d}_v$, $\mathbf{g}_v \in \mathbb{R}_+^C$	Nonnegative C-dimensional representation vector of vertex v in discriminator D and generator G, respectively
θ_D, $\theta_G \in \mathbb{R}_+^{V\times C}$	The union of all $\mathbf{d}_v$ and $\mathbf{g}_v$, respectively
$F_{uc} \in \mathbb{R}_+$	The nonnegative strength for vertex u to be affiliated to community c used in AGM
$p_c(v_1, v_2, \ldots, v_m)$	The probability of m vertices v_1 to v_m to be a clique through of community c
$p(v_1, v_2, \ldots, v_m)$	The probability of m vertices v_1 to v_m to be a clique through of any one community
$\odot(F_{v_1}, F_{v_2}, \ldots, F_{v_m})$	Sum of the entrywise product from F_{v_1} to F_{v_m}, i.e., $\sum_{c=1}^{C}\prod_{i=1}^{m} F_{v_i c}$
$G_v(v_{s_m}\|v_{s_1}, \ldots, v_{s_{m-1}})$	The vertex generator based on selected vertices $v_{s_1}, \ldots, v_{s_{m-1}}$ in vertex subset generation process
$p_{v_v}(v_i\|v_c)$	The relevance probability of v_i given v_c with root v_v in random walk process

The generator G and discriminator D are combined by playing a minimax game: generator G would try to perfectly approximate $p_{true}(m|v_c)$ and generate the most likely vertex subsets similar to real motifs covering v_c to fool the discriminator, while discriminator D would try to distinguish between ground-truth motifs from $p_{true}(m|v_c)$ and the ones generated by G. Formally, G and D act as two opponents in the following two-player minimax game with the joint value function $V(G, D)$:

$$\min_{\theta_G}\max_{\theta_D} V(G,D) = \sum_{c=1}^{V}\Big(\mathbb{E}_{m\sim p_{true}(\cdot|v_c)}[\log D(m;\theta_D)] + \mathbb{E}_{s\sim G(s|v_c;\theta_G)}[\log(1-D(s;\theta_D))]\Big). \tag{1}$$

Based on Eq. (1), the optimal G and D can be learned by alternatively maximizing (D) and minimizing (G) the value function $V(G, D)$. The CommunityGAN framework is illustrated in Figure 3. Discriminator D is trained with positive samples from $p_{true}(\cdot|v_c)$ and negative samples from $G(\cdot|v_c;\theta_G)$, and generator G is updated with policy gradient technique [22] under the guidance of D (detailedly described later in this section). Competition between G and D drives both of them to improve until G is indistinguishable from the true motif distribution p_{true}.

4.2 CommunityGAN Optimization

Given positive samples from true motif distribution p_{true} and negative samples from the generator G, the discriminator aims to maximize the log-probability of correctly classifying these samples, which could be solved by gradient ascent if D is differentiable for θ_D, i.e.,

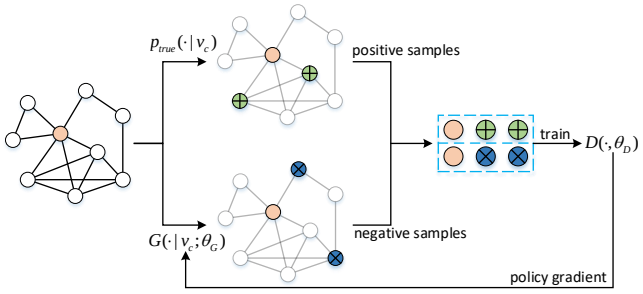


Figure 3: Framework of CommunityGAN

$$\nabla_{\theta_D} V(G,D) = \sum_{c=1}^{V}\Big(\mathbb{E}_{m\sim p_{true}(\cdot|v_c)}[\nabla_{\theta_D}\log D(m;\theta_D)] + \mathbb{E}_{s\sim G(s|v_c;\theta_G)}[\nabla_{\theta_D}\log(1-D(s;\theta_D))]\Big). \tag{2}$$

In contrast to the discriminator, the objective of the generator is to minimize the log-probability that the discriminator correctly distinguishes samples from G. Because the sampling of s is discrete, we follow [21, 25, 33] to compute the gradient of $V(G, D)$ with respect to θ_G by policy gradient:

$$\begin{aligned}\nabla_{\theta_G} V(G,D) &= \nabla_{\theta_G}\sum_{c=1}^{V}\mathbb{E}_{s\sim G(\cdot|v_c)}[\log(1-D(s))] \\ &= \sum_{c=1}^{V}\mathbb{E}_{s\sim G(\cdot|v_c)}[\nabla_{\theta_G}\log G(s|v_c)\log(1-D(s))].\end{aligned} \tag{3}$$

4.3 A Naive Implementation of D and G

A naive implementation of the discriminator and generator is based on sigmoid and softmax functions, respectively.

For the discriminator D, intuitively we can define it as the multiplication of the sigmoid function of the inner product of every two vertices in the input vertex subset s:

$$D(s) = \prod_{(u,v)\in s,\, u\neq v}\sigma(\mathbf{d}_u^{\top}\cdot\mathbf{d}_v), \tag{4}$$

where $\mathbf{d}_u, \mathbf{d}_v \in \mathbb{R}^k$ are the k-dimensional representation vectors for discriminator D of vertices u and v respectively, and θ_D is the union of all $\mathbf{d}_v$'s.

For the implementation of G, to generate a vertex subset s covering vertex v_c, we can regard the subset as a sequence of vertices $(v_{s_1}, v_{s_2}, \ldots, v_{s_m})$ where $v_{s_1} = v_c$. Then the generator G can be defined as follows:

$$\begin{aligned}&G(s|v_c) \\ =&G_v(v_{s_2}|v_{s_1})G_v(v_{s_3}|v_{s_1}, v_{s_2})\cdots G_v(v_{s_m}|v_{s_1}, \ldots, v_{s_{m-1}}).\end{aligned} \tag{5}$$

Notably, in Eq. (5), the generation of v_{s_m} is based on vertices from v_{s_1} to $v_{s_{m-1}}$, not only on $v_{s_{m-1}}$. Because if we generate v_{s_m} based only on $v_{s_{m-1}}$, it will be very likely that vertex v_{s_m} and the other vertices belong to different communities. For example, vertices from v_{s_1} to $v_{s_{m-1}}$ are students in one university, while vertex v_{s_m} is the parent of $v_{s_{m-1}}$. Simply, we can know the probability of the vertex subset s being a clique will be very low. Thus, we generate v_{s_m} based on all the vertices from v_{s_1} to $v_{s_{m-1}}$.

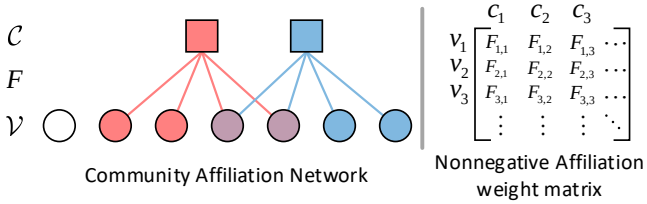


Figure 4: AGM framework. $\mathcal{V}$: vertices; C: communities; F: affiliation of vertices to communities.

For the implementation of the vertex generator G_v, straightforwardly, we can define it as a softmax function over all other vertices, i.e.,

$$G_v(v_{s_m}|v_{s_1},\ldots,v_{s_{m-1}}) = \frac{\exp(\sum_{i=1}^{m-1}\mathbf{g}_{v_{s_m}}^\top \mathbf{g}_{v_{s_i}})}{\sum_{v\notin(v_{s_1},\ldots,v_{s_{m-1}})}\exp(\sum_{i=1}^{m-1}\mathbf{g}_v^\top \mathbf{g}_{v_{s_i}})}, \tag{6}$$

where $\mathbf{g}_v \in \mathbb{R}^k$ is the k-dimensional representation vectors for generator G_v of vertex v, and θ_G is the union of all $\mathbf{g}_v$'s.

4.4 Graph AGM

Sigmoid and softmax function provide concise and intuitive definitions for the motif discrimination in D_s and vertex generation in G_v, but they have three limitations in community detection task: 1) To detect community, after learning the vertex representation vectors based on Eq. (4) and (6), we still need to adopt some clustering algorithms to detect the communities. According to [29], the overlap is indeed a significant feature of many real-world social networks. Moreover, in [30], the authors showed that in some real-world datasets, one vertex might belong to tens of communities simultaneously. However, general clustering algorithms cannot handle such dense overlapping. 2) The calculation of softmax in Eq. (6) involves all vertices in the graph, which is computationally inefficient. 3) The graph structure encodes rich information of proximity among vertices, but the softmax in Eq. (6) completely ignores it.

To address these problems, in CommunityGAN we propose a novel implementation for the discriminator and generator, which is called Graph AGM.

AGM (Affiliation Graph Model) [30, 31] is based on the idea that communities arise due to shared group affiliation, and views the whole network as a result generated by a community-affiliation graph model. The framework of AGM is illustrated in Figure 4, which can be either seen as a bipartite network between vertices and communities, or written as a nonnegative affiliation weight matrix. In AGM, each vertex could be affiliated to zero, one or more communities. If vertex u is affiliated to community c, there will be a nonnegative strength F_{uc} of this affiliation. For any community $c \in C$, it connects its member vertices u, v with probability $1-\exp(-F_{uc}\cdot F_{vc})$. Moreover, each community c creates edges independently. If the pair of vertices u, v are connected multiple times through different communities, the probability is $1-\exp(-\sum_c F_{uc}\cdot F_{vc})$. So that the probability that vertices u, v are connected (through any possible communities) is $p(u,v) = 1-\exp(-F_u^\top \cdot F_v)$, where F_u and F_v are the nonnegative C-dimensional affiliation vectors for vertices u and v respectively.

We extend AGM from edge generation to motif generation. For any m vertices v_1 to v_m, we assume that the probability of them to be a clique through of community c is defined as $p_c(v_1, v_2, \ldots, v_m) = 1-\exp(-\prod_{i=1}^m F_{v_i c})$. Then the probability that these m vertices compose a clique through any possible communities can be calculated via

$$\begin{aligned} p(v_1, v_2, \ldots, v_m) &= 1-\prod_c (1-p_c(v_1, v_2, \ldots, v_m)) \\ &= 1-\exp(-\odot(F_{v_1}, F_{v_2}, \ldots, F_{v_m})), \end{aligned} \tag{7}$$

where $\odot(F_{v_1}, F_{v_2}, \ldots, F_{v_m})$ means the sum of the entrywise product from F_{v_1} to F_{v_m}, i.e., $\sum_{c=1}^C \prod_{i=1}^m F_{v_i c}$.

Then the discriminator, which was defined as the product of sigmoid in a straightforward way, can be redefined as

$$D(s) = 1-\exp(-\odot(\mathbf{d}_{v_1}, \mathbf{d}_{v_2}, \ldots, \mathbf{d}_{v_m})), \tag{8}$$

where $\mathbf{d}_v \in \mathbb{R}^C$ is the nonnegative C-dimensional representation vectors of vertex v for discriminator D, and θ_D is the union of all $\mathbf{d}_v$'s.

Moreover, the generator G_v can be redefined as the softmax function over all other possible vertices to compose a clique with $m-1$ chosen vertices:

$$\begin{aligned} &G_v(v_{s_m}|v_{s_1},\ldots,v_{s_{m-1}}) \\ &= \frac{1-\exp(-\odot(\mathbf{g}_{v_{s_1}},\ldots,\mathbf{g}_{v_{s_m}}))}{\sum_{v\notin(v_{s_1},\ldots,v_{s_{m-1}})} 1-\exp(-\odot(\mathbf{g}_{v_{s_1}},\ldots,\mathbf{g}_{v_{s_{m-1}}},\mathbf{g}_v))}, \end{aligned} \tag{9}$$

where $\mathbf{g}_v \in \mathbb{R}^C$ is the nonnegative C-dimensional representation vectors of vertex v for generator G_v, and θ_G is the union of all $\mathbf{g}_v$'s. With this setting, the learned vertex representation vectors $\mathbf{g}_v$ will represent the affiliation weight between vertex v and communities, which means we need no additional clustering algorithms to find the communities and the first aforementioned limitation is omitted.

To further overcome the other two limitations, inspired by [25], we design graph AGM as follows. To calculate $G_v(v_{s_m}|v_{s_1},\ldots,v_{s_{m-1}})$, we first assume there is a virtual vertex v_v which is connected to all the vertices in the union of neighbors of vertices from v_{s_1} to $v_{s_{m-1}}$, i.e., $N(v_v) = N(v_{s_1}) \cup \cdots \cup N(v_{s_{m-1}})$ where $N(v)$ represents the set of neighbors of vertex v. Moreover, we assign $\mathbf{g}_{v_v} = \mathbf{g}_{v_{s_1}} \circ \cdots \circ \mathbf{g}_{v_{s_{m-1}}}$, i.e., the representation vector of v_v is the entrywise product of the representation vectors of vertices from v_{s_1} to $v_{s_{m-1}}$. For simplicity, we substitute $G_v(v_{s_m}|v_{s_1},\ldots,v_{s_{m-1}})$ with $G_v(v_{s_m}|v_v)$. Then we adopt random walk starting from the virtual vertex v_v based on a well designed probability distribution (Eq. (10)). During the process of random walk, if the currently visited vertex is v and generator G decides to visit v's previous vertex, then v is chosen as the generated vertex and the random walk process stops. The whole process for generating a vertex subset with the size of 3 is illustrated in Figure 5.

Moreover, in the random walk process, we wish the walk path is always relevant to the root vertex v_v for maximizing the probability of the generated vertex subset to be a motif. So that for a given vertex v_c and one of its neighbors $v_i \in N(v_c)$, we define the relevance probability of v_i given v_c as

$$p_{v_v}(v_i|v_c) = \frac{1-\exp(-\odot(\mathbf{g}_{v_i}, \mathbf{g}_{v_c}, \mathbf{g}_{v_v}))}{\sum_{v_j\in N(v_c)} 1-\exp(-\odot(\mathbf{g}_{v_j}, \mathbf{g}_{v_c}, \mathbf{g}_{v_v}))}, \tag{10}$$

which is actually a softmax function over $N(v_c)$ for composing clique with vertices v_v and v_c.

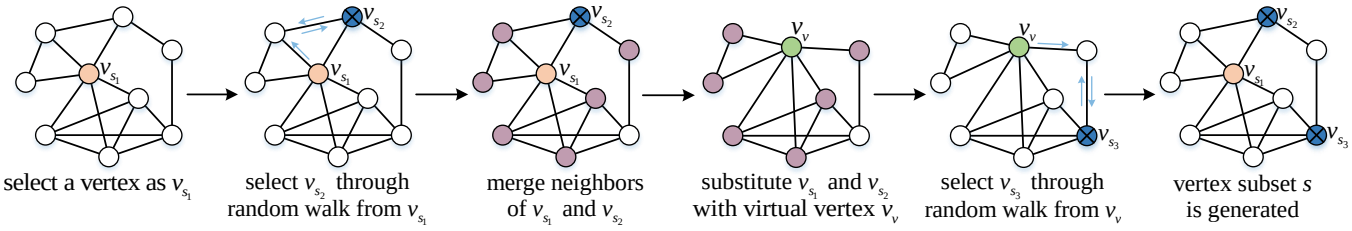


Figure 5: The generation process for vertex subset s. Blue arrows indicate the path of random walk. At the two blue vertices, because the vertex generator decides to visit the previous vertex, the random walk finishes and the blue vertices are selected.

Algorithm 1: CommunityGAN framework

Input: number of communities c, size of discriminating samples m, size of generating samples n.
Output: generator $G(s|v_c;\theta_G)$, discriminator $D(s,\theta_D)$.

```
Initialize and pre-train G(s|v_c;θ_G) and D(s,θ_D);
while CommunityGAN not converge do
    for G-steps do
        Generate n vertex subsets from G(s|v_c;θ_G) for each vertex v_c based on the random walk process;
        Update θ_G according to Eq. (3), (5), (10) and (11);
    end
    for D-steps do
        Sample m positive vertex subsets from p_true and s negative vertex subsets from G(s|v_c;θ_G) for each vertex v_c;
        Update θ_D according to Eq. (2) and (8);
    end
end
return G(s|v_c;θ_G) and D(s,θ_D);
```

If we denote the path of random walk as $P_r = (v_{r_1}, v_{r_2}, \ldots, v_{r_n})$ where $v_{r_1} = v_v$, the probability for selecting this path will be $p_{v_v}(v_{r_{n-1}}|v_{r_n}) \cdot \prod_{i=1}^{n-1} p_{v_v}(v_{r_{i+1}}|v_{r_i})$. In the policy gradient, we regard the selection of this path as an action and the target is to maximize its reward from D. Thus, although there may be multiple paths between v_{r_1} and v_{r_n}, if we have selected the path P_r, we will optimize the policy gradient on it and neglect other paths. In other words, if we select the path P_r, we assign $G_v(v_{s_m}|v_v)$ as follows:

$$G_v(v_{s_m}|v_v) = p_{v_v}(v_{r_{n-1}}|v_{r_n}) \cdot \prod_{i=1}^{n-1} p_{v_v}(v_{r_{i+1}}|v_{r_i}). \tag{11}$$

Finally, the overall solution of CommunityGAN is summarized in Algorithm 1.

4.5 Other Issues

Model Initialization. We have two methods to initialize the generator G and the discriminator D. The first is that we can deploy AGM model on the graph to learn a community affiliation vector F_i for each vertex v_i, and then we can set $g_{v_i} = d_{v_i} = F_i$ directly. The second is that we can use locally minimal neighborhoods [7] to initialize θ_G and θ_D. We can regard each vertex v_i along with its neighbors $N(v_i)$, denoted as $C(v_i)$, as a community. Community $C_(v_i)$ is called locally minimal if $C_(v_i)$ has lower conductance than all the $C_(v_j)$ for vertices v_j who are connected to vertex v_i. Gleich and Seshadhri [7] have empirically showed that the locally minimal neighborhoods are good seed sets for community detection algorithms. For a node v_i who belongs to a locally minimal neighborhood c, we initialize $F_{v_i c} = 1$, otherwise $F_{v_i c} = 0$.

The second method can save the training time of AGM on the graph. However, we find that the performance of such initialization is a little lower than the first initialization. To achieve best performance, in this paper, we choose to adopt the first initialization method. Moreover, in the efficiency analysis experiment (refer to §6.4), the training time of CommunityGAN includes the time of the pre-training process.

Determining community membership. After learning parameters θ_G and θ_D, we need to determine the "hard" community membership of each node. We achieve this by thresholding θ_G and θ_D with a threshold δ. The basic intuition is that if two nodes belong to the same community c, then the probability of having a link between them through community c should be larger than the background edge probability $\epsilon = 2E/V(V-1)$. Following this idea, we can obtain the threshold δ as below:

$$1 - \exp(-\delta^2) = \epsilon \Rightarrow \delta = \sqrt{-\log(1-\epsilon)} \tag{12}$$

With δ obtained, we consider vertex v_i belonging to community c if $\mathbf{g}_{v_i c} \geq \delta$ or $\mathbf{d}_{v_i c} \geq \delta$ for generator G and discriminator D respectively.

Choosing the number of communities. We follow the method proposed in [1] to choose the number of communities C. Specifically, we reserve 20% of links for validation and learn the model parameters with the remaining 80% of links for different C. After that, we use the learned parameters to predict the links in validation set and select the C with the maximum prediction score as the number of communities.

5 SYNTHETIC DATA EXPERIMENTS

In this section, in order to evaluate the effectiveness of CommunityGAN, we conduct a series of experiments based on synthetic datasets. Specifically, we conduct two experiments to prove: 1) the ability of CommunityGAN to solve the dense overlapping problem; 2) the efficacy of motif-level generation and discrimination. To simulate the real-world graphs with ground-truth overlapping communities, we adopt CKB Graph Generator - a method that can generate large random social graphs with realistic overlapping community structure [4] - to generate such synthetic graphs. The

Table 4: Synthetic Graphs statistics. V: number of vertices, E: number of edges, C: number of communities, A: average number of community memberships per vertex, P: percentage of communities that have overlapping with others

Graph	V	E	C	A	P
ckb-290	1078	29491	226	2.76	100%
ckb-280	1044	30055	252	3.03	100%
ckb-276	1023	38775	339	3.74	100%
ckb-273	1018	28025	372	3.94	100%
ckb-270	1007	30740	371	4.20	100%
ckb-267	1123	65813	381	4.57	100%

code of CommunityGAN (including pre-training model) and demo datasets are available online[1].

5.1 Evaluation Metric

The availability of ground-truth communities allows us to quantitatively evaluate the performance of community detection algorithms. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{M})$, we denote the set of ground-truth communities as C^* and the set of detected communities as $\hat{C}$, where each ground-truth community $c_i^* \in C^*$ and each detected community $\hat{c}_i \in \hat{C}$ are both defined by a set of their member nodes. We consider the average F1-Score as the metric to measure the performance of the detected communities $\hat{C}$. To compute the F1-Score between detected communities and ground-truth communities, we need to determine which $c_i^* \in C^*$ corresponds to which $\hat{c}_j \in \hat{C}$. We define final F1-Score as the average of the F1-score of the best-matching ground-truth community to each detected community, and the F1-score of the best-matching detected community to each ground-truth community:

$$F1(C^*, \hat{C}) = \frac{1}{2}\left(\frac{1}{C^*}\sum_{c_i^* \in C^*} \max_{\hat{c}_j \in \hat{C}} F1(c_i^*, \hat{c}_j) + \frac{1}{\hat{C}}\sum_{\hat{c}_i \in \hat{C}} \max_{c_j^* \in C^*} F1(c_j^*, \hat{c}_i)\right) \tag{13}$$

The higher F1-Score means that the detected communities are more accurate and have better qualities.

5.2 Datasets

To prove CommunityGAN's ability to solve the dense overlapping problem, we utilize the CKB Graph Generator to generate synthetic graphs with ground-truth communities of different overlapping levels. All the parameters of CKB are set as default except two: 'power law exponent of user-community membership distribution (β_1)' and 'power law exponent of community size distribution (β_2)' [4]. By setting $\beta_1 = \beta_2 = 2.9, 2.8, 2.76, 2.73, 2.7, 2.67$, we can get a series of graphs with different levels of community overlapping. The detailed statistics of these graphs are shown in Table 4. We can see that the percentages of overlapped communities (denoted by P) are all 100%, indicating that all the communities overlap more or less with others. Besides, with the decrease of β_1 and β_2, the average number of community memberships per vertex (denoted by A) increases, which means the density of community overlapping is becoming higher and higher.

[1]https://github.com/SamJia/CommunityGAN

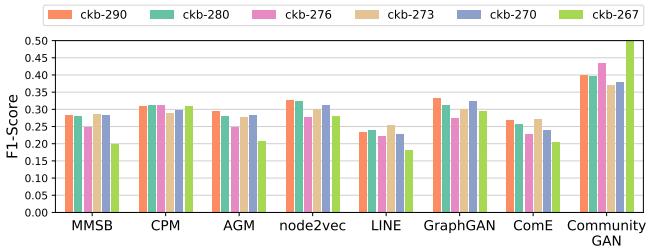


Figure 6: Performance of methods on the synthetic graphs.

5.3 Comparative Methods

We compare CommunityGAN with 7 baseline methods, of which 3 are traditional overlapping community detection methods (MMSB, CPM and AGM), 3 are recent graph representation learning methods (node2vec, LINE and GraphGAN), and 1 combines graph representation learning as well as community detection (ComE).

MMSB [1] is one of the representatives of overlapping community detection methods based on dense subgraph extraction.

CPM [18] builds up the communities from k-cliques and allows overlapping between the communities in a natural way.

AGM [31] is a framework which can model densely overlapping community structures.

Node2vec [8] adopts biased random walk and Skip-Gram to learn vertex embeddings.

LINE [24] preserves the first-order and second-order proximity among vertices in the graph.

GraphGAN [25] unifies generative and discriminative graph representation learning methodologies via adversarial training in a minimax game.

ComE [3] jointly solves the graph representation learning and community detection problem.

5.4 Experiment Setup

For all the experiments, we perform stochastic gradient descent to update parameters with learning rate 0.001. In each iteration, m and n are both 5, i.e., 5 positive and 5 negative vertex subsets will be sampled or generated for each vertex (refer to Algorithm 1), and then we update G and D on these vertex subsets for 3 times. The final learned communities and vertex representations are $\mathbf{g}_v$'s. Hyperparameter settings for all baselines are as default.

For CommunityGAN, CPM, AGM, MMSB and ComE, which directly output the communities for vertices, we fit the network into them to detect communities directly. For node2vec, LINE and GraphGAN, which can only output the representation for vertices, we fit the network into them and apply K-means to the learned embeddings to get the communities. Because of the randomness of K-means, we repeat 5 times and report the average results. Moreover, the number of ground-truth communities is set as the input parameters to the community detection models and K-means.

5.5 Ability of CommunityGAN to Solve Dense Overlapping Problem

For each method and each dataset, we calculate the average value of F1-Score. The performance of 8 methods on the series of graphs is

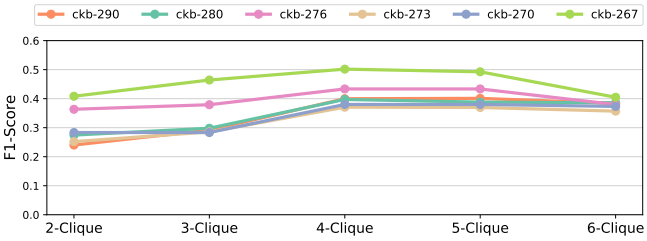


Figure 7: Performance of CommunityGAN with different size of cliques on the synthetic graphs.

shown in Figure 6. As we can see: 1) CommunityGAN significantly outperforms all the baselines on all the synthetic graphs. 2) Although CommunityGAN is based on AGM and utilizes AGM as the pre-train method, with the minimax game competition between the discriminator and generator, CommunityGAN achieves huge improvements compared to AGM. 3) On the whole, the performance of 4 graph representation learning methods (node2vec, LINE, GraphGAN, ComE) falls with the increase of overlapping density among communities, while 3 traditional overlapping community detection methods (MMSB, CPM, AGM) keep relatively steady. Among these 3 methods, the only exception is that the performance of MMSB and AGM drops largely on ckb-267. One possible reason is that the average degree of ckb-267 increases significantly compared to other graphs, and thus the performance of the two methods are affected. However, although based on AGM, CommunityGAN still performs very well on the graph ckb-267 with the well designed positive/negative sampling and the minimax game between the generator and discriminator.

5.6 Efficacy of Motif-Level Generation and Discrimination

To evaluate the efficacy of the motif-level generation and discrimination, we tune the size of cliques in the training process of CommunityGAN. Figure 7 shows the performance of CommunityGAN with different sizes of cliques on the series of synthetic graphs. We can find that, generally, cliques with a size of 4 or 5 can reach the best performance. It means that, compared to edges and triangles, the larger cliques in graphs correspond more to the community structure. In other words, community detection methods should not only focus on the edges in the graph, but also pay more attention to the motifs, especially larger motifs. However, we can see that in most graphs the performance falls when CommunityGAN generates and discriminates 6-vertex cliques. Our explanation is that too-large cliques are rare in graphs, which can only cover a relatively small number of nodes. Because of these uncovered nodes, few positive samples can be used in the training process, which in turn causes CommunityGAN to fail to learn appropriate representation vectors.

To prove such an explanation, we have counted the number of cliques and number of vertices covered by such cliques, as shown in Table 5. We can see that no matter whether the number of cliques increases or not, the number of covered vertices does not change very much from 2-clique to 5-clique. Thus CommunityGAN can benefit from the larger cliques to gather better community structure, and then reaches the best performance when clique size is 4 or 5. However, when the size of clique reaches 6, the number of covered vertices drops dramatically, and thus the performance of CommunityGAN decreases.

Table 5: Statistics of cliques in the synthetic graphs.

Number of cliques

Model	2-clique	3-clique	4-clique	5-clique	6-clique
ckb-290	29491	105585	82617	31919	14354
ckb-280	30055	115047	96661	33735	9539
ckb-276	38775	175639	189300	126720	76247
ckb-273	28025	122566	166760	163765	165575
ckb-270	30740	140547	203114	211604	214018
ckb-267	65813	377686	580950	811522	1097762

Number of covered vertices

Model	2-clique	3-clique	4-clique	5-clique	6-clique
ckb-290	1078	1078	1078	1065	595
ckb-280	1044	1044	1044	1034	617
ckb-276	1023	1023	1023	1018	714
ckb-273	1018	1018	1018	1012	798
ckb-270	1007	1007	1007	1005	797
ckb-267	1123	1123	1123	1120	907

5.7 Discussion

With the two experiments on the series of synthetic graphs, we can draw the following conclusions:

- With the same special vector design as AGM, CommunityGAN holds the ability to solve the dense overlapping problem. With the increase of overlapping among communities, the performance of CommunityGAN can maintain steadiness and does not fall like the baseline methods.
- With the minimax game competition between the discriminator and generator, CommunityGAN gains significant performance improvements based on AGM.
- With proper sizes, the motif-level generation and discrimination can help models learn better community structures compared to edge-level optimization.

6 REAL-WORLD SCENARIOS

To complement the previous experiments, in this section, we evaluate the performance of CommunityGAN on a series of real-world datasets. Specifically, we choose two application scenarios for experiments, i.e., community detection and clique prediction.

6.1 Datasets

We evaluate our model using two categories of datasets, one with ground-truth communities and one without.

Datasets with ground-truth communities.[2] Because of the training time of some baselines, we only sample three subgraphs with 100 ground-truth communities as the experiment networks from these three large networks.

[2]http://snap.stanford.edu/data/#communities

Table 6: F1-Score and NMI on community detection.

F1-Score			
Model	Amazon	Youtube	DBLP
MMSB	0.366	0.124	0.104
CPM	0.214	0.058	0.318
AGM	0.711	0.158	0.398
node2vec	0.550	0.288	0.265
LINE	0.532	0.170	0.208
GraphGAN	0.518	0.303	0.276
ComE	0.562	0.213	0.240
CommunityGAN	**0.860**	**0.327**	**0.456**

NMI			
Model	Amazon	Youtube	DBLP
MMSB	0.068	0.031	0.000
CPM	0.027	0.000	0.066
AGM	0.635	0.025	0.059
node2vec	0.370	0.071	0.068
LINE	0.248	0.070	0.027
GraphGAN	0.417	0.049	0.083
ComE	0.413	**0.091**	0.059
CommunityGAN	**0.853**	**0.091**	**0.153**

Amazon is collected by crawling Amazon website. The vertices represent products; the edges indicate the frequently co-purchase relationships; the ground-truth communities are defined by the product categories in Amazon. This graph has 3,225 vertices and 10,262 edges.

Youtube is a network of social relationships of Youtube website users. The vertices represent users; the edges indicate friendships among the users; the user-defined groups are considered as ground-truth communities. This graph has 4,890 vertices and 20,787 edges.

DBLP is a co-authorship network from DBLP. The vertices represent researchers; the edges indicate the co-author relationships; authors who have published in a same journal or conference form a community. This graph has 10,824 vertices and 38,732 edges.

Datasets without ground-truth communities.[3]

arXiv-AstroPh is from the e-print arXiv and covers scientific collaborations between authors with papers submitted to the Astro Physics category. The vertices represent authors, and the edges indicate co-author relationships. This graph has 18,772 vertices and 198,110 edges.

arXiv-GrQc is also from arXiv and covers scientific collaborations between authors with papers submitted to the General Relativity and Quantum Cosmology categories. This graph has 5,242 vertices and 14,496 edges.

6.2 Community Detection

The community detection experiment is conducted on three networks with ground-truth community memberships, which allows us to quantify the accuracy of community detection methods by evaluating the level of correspondence between detected and ground-truth communities.

[3]http://snap.stanford.edu/data/#canets

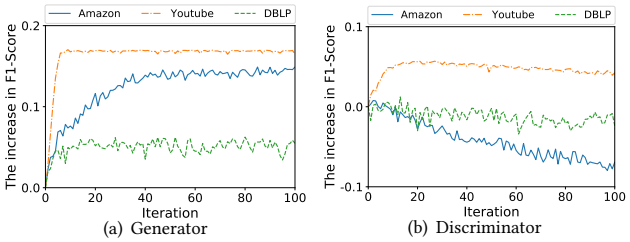


Figure 8: Learning curves of the generator and the discriminator of CommunityGAN in community detection.

Setup. In the community detection experiment, all the baselines and setups are the same as the synthetic data experiments, whose details refer to §5.4.

Evaluation Metric. Besides the F1-Score described in synthetic dataset experiments (refer to §5.1), to comprehensively evaluate the performance of methods, we also introduce the Normalized Mutual Information (NMI) in the community detection experiment. NMI is a measure of similarity borrowed from information theory. Later, it is extended to measure the quality of overlapping communities. Similar to F1-Score, the higher NMI means the better performance. Please refer to [13] for details.

Results. Table 6 shows the F1-Score and NMI of 8 methods on 3 networks, respectively. As we can see: 1) Performance of CPM, MMSB and LINE is relatively poor in community detection, which means they cannot capture the pattern of community structure in graphs well. 2) Node2vec, ComE and GraphGAN perform better than CPM, MMSB and LINE. One possible reason is that node2vec and GraphGAN adopt the random walk to capture the vertex context which is similar to community structure in some way, and ComE solves the network embedding and community detection task in one unified framework. 3) CommunityGAN outperforms all the baselines (including its pre-train method AGM) in community detection task. Specifically, CommunityGAN improves F1-Score by 7.9% (on Youtube) to 21.0% (on Amazon) and improves NMI by at most 34.3% (on Amazon). Our explanation is that the consideration of dense community overlapping provides CommunityGAN a higher learning flexibility than the non-overlapping or sparse overlapping community detection baselines. Moreover, the minimax game between the generator and discriminator drives CommunityGAN to gain significant performance improvements based on AGM.

To intuitively understand the learning stability of CommunityGAN, we further illustrate the learning curves of the generator and the discriminator on the three datasets in Figure 8. Because the F1-Score on the three datasets differs very much, we only demonstrate the relative increase in the score. As we can see, the generator performs outstandingly well after convergence, while the performance of the discriminator falls a little or boosts at first and then falls. Maybe it is because that the discriminator aims to distinguish

Table 7: AUC on clique prediction.

arXiv-AstroPh			
Model	2-clique	3-clique	4-clique
AGM	0.919	0.987	0.959
node2vec	0.579	0.514	0.544
LINE	0.918	0.980	0.963
GraphGAN	0.799	0.859	0.855
ComE	0.904	0.951	0.953
CommunityGAN	**0.923**	**0.990**	**0.970**

arXiv-GrQc			
Model	2-clique	3-clique	4-clique
AGM	0.900	0.980	0.871
node2vec	0.632	0.569	0.534
LINE	**0.969**	0.989	0.880
GraphGAN	0.756	0.880	0.728
ComE	0.924	0.962	0.914
CommunityGAN	0.904	**0.993**	**0.956**

ground truth from generated samples better. However, the generated samples are all around the center vertex, which causes the discriminator to lose some global discrimination ability.

6.3 Clique Prediction

In clique prediction, our goal is to predict whether a given subset of vertices is a clique. Therefore, this task shows the performance of graph local structure extraction ability of different graph representation learning methods.

Setup. In the clique prediction experiment, because some traditional community detection methods (including MMSB and CPM) cannot predict the existence of edges among vertices, these methods are omitted in this experiment. To analyze the effect of motif generation and discrimination in CommunityGAN, in this experiment, we have evaluated the prediction for 2-clique (same to edge), 3-clique and 4-clique. With the size of cliques determined, we randomly hide some cliques, which cover 10% of edges, in the original graph as ground truth, and use the left graph to train all graph representation learning models. After training, we obtain the representation vectors for all vertices and use logistic regression method to predict the probability of being clique for a given vertex set. Our test set consists of the hidden vertex sets (cliques) in the original graph as the positive samples and the randomly selected non-fully connected vertex sets as negative samples with the equal number.

Results. We use arXiv-AstroPh and arXiv-GrQc as datasets, and report the results of AUC in Table 7. As we can see, even though in 2-clique (edge) prediction CommunityGAN does not always get the highest score, CommunityGAN outperforms all the baselines in both 3-clique and 4-clique prediction. For example, on arXiv-GrQc, CommunityGAN achieves gains of 0.40% to 74.52% and 4.60% to 79.03% in 3-clique and 4-clique prediction, respectively. This indicates that though CommunityGAN is designed for community detection, with the design of motif generation and discrimination,

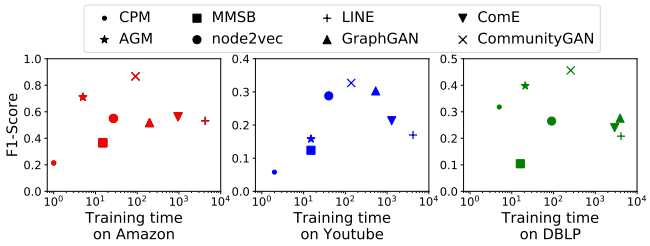


Figure 9: Performance as training time (Sec.).

it can still effectively encode the information of clique structures into the learned representations.

6.4 Efficiency analysis

In this paper, we propose Graph AGM for efficiently generating the most likely motifs with graph structure awareness. Because of the random walk process, in which the exact time complexity is not easy to infer, we evaluate the efficiency of CommunityGAN by directly comparing the training time with baselines. In this evaluation, the number of threads is set as 16 if the model supports parallelization and other parameters for baselines are as default. Figure 9 illustrates the performance and training time. Notably, the training time of CommunityGAN includes the time of the pre-training process. Even though CommunityGAN is not the fastest model, the training time of CommunityGAN is still acceptable and its performance significantly outperforms the faster models.

7 CONCLUSION

In this paper we proposed CommunityGAN that jointly solves the overlapping community detection and graph representation learning. Unlike the embedding of general graph learning algorithms in which the vector values have no meanings, the embedding in CommunityGAN indicates the membership strength of vertices to communities, which enables CommunityGAN to detect densely overlapped communities. Then a specifically designed GAN is adopted to optimize such embedding. Through the minimax game of motif-level generator and discriminator, both of them can boost their performance and finally output better community structures. We adopted CKB Graph Generator to create a series of synthetic graphs with ground-truth communities. Two experiments were conducted on these graphs to prove the ability of CommunityGAN to solve dense overlapping problem and its efficacy of motif generation and discrimination. Additionally, to complement the experiments on the synthetic datasets, we did experiments on five real-world datasets in two scenarios, where the results demonstrate that CommunityGAN substantially outperforms baselines in all experiments due to its specific embedding design and motif-level optimization.

8 ACKNOWLEDGEMENTS

The work is supported by National Key R&D Program of China (2018YFB1004702), National Natural Science Foundation of China (61532012, 61829201, 61702327, 61772333), Shanghai Sailing Program (17YF1428200). Weinan Zhang and Xinbing Wang are the corresponding authors.

REFERENCES

[1] Edo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. 2009. Mixed membership stochastic blockmodels. In *Advances in Neural Information Processing Systems*. 33–40.

[2] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning Graph Representations with Global Structural Information. In *CIKM*. ACM, New York, NY, USA, 891–900. https://doi.org/10.1145/2806416.2806512

[3] Sandro Cavallari, Vincent W Zheng, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. 2017. Learning community embedding with community detection and node embedding on graphs. In *CIKM*. ACM, 377–386.

[4] Kyrylo Chykhradze, Anton Korshunov, Nazar Buzun, Roman Pastukhov, Nikolay Kuzyurin, Denis Turdakov, and Hangkyu Kim. 2014. Distributed Generation of Billion-node Social Graphs with Overlapping Community Structure. In *Complex Networks V*, Pierluigi Contucci, Ronaldo Menezes, Andrea Omicini, and Julia Poncela-Casasnovas (Eds.). Springer International Publishing, Cham, 199–208.

[5] Quanyu Dai, Qiang Li, Jian Tang, and Dan Wang. 2017. Adversarial Network Embedding. *CoRR* abs/1711.07838 (2017). arXiv:1711.07838 http://arxiv.org/abs/1711.07838

[6] Sheng Gao, Ludovic Denoyer, and Patrick Gallinari. 2011. Temporal link prediction by integrating content and structure information. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 1169–1174.

[7] D. F. Gleich and C. Seshadhri. 2012. Neighborhoods are good communities. In *KDD'12*.

[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. *CoRR* abs/1607.00653 (2016). arXiv:1607.00653 http://arxiv.org/abs/1607.00653

[9] Zhiting Hu, Junjie Yao, Bin Cui, and Eric Xing. 2015. Community level diffusion extraction. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. ACM, 1555–1569.

[10] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR* abs/1609.02907 (2016). arXiv:1609.02907 http://arxiv.org/abs/1609.02907

[11] Christine Klymko, David F. Gleich, and Tamara G. Kolda. 2014. Using Triangles to Improve Community Detection in Directed Networks. *CoRR* abs/1404.5874 (2014). arXiv:1404.5874 http://arxiv.org/abs/1404.5874

[12] Nevan J Krogan, Gerard Cagney, Haiyuan Yu, Gouqing Zhong, Xinghua Guo, Alexandr Ignatchenko, Joyce Li, Shuye Pu, Nira Datta, Aaron P Tikuisis, et al. 2006. Global landscape of protein complexes in the yeast Saccharomyces cerevisiae. *Nature* 440, 7084 (2006), 637–643.

[13] Andrea Lancichinetti, Santo Fortunato, and János Kertész. 2009. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics* 11, 3 (2009), 033015.

[14] Ye Li, Chaofeng Sha, Xin Huang, and Yanchun Zhang. 2018. Community Detection in Attributed Graphs: An Embedding Approach. In *AAAI Conference on Artificial Intelligence*. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17142

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *CoRR* abs/1310.4546 (2013). arXiv:1310.4546 http://arxiv.org/abs/1310.4546

[16] Mark EJ Newman. 2004. Detecting community structure in networks. *The European Physical Journal B-Condensed Matter and Complex Systems* 38, 2 (2004), 321–330.

[17] Mark EJ Newman. 2006. Modularity and community structure in networks. *Proceedings of the national academy of sciences* 103, 23 (2006), 8577–8582.

[18] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435, 7043 (2005), 814–818.

[19] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. *CoRR* abs/1403.6652 (2014). arXiv:1403.6652 http://arxiv.org/abs/1403.6652

[20] Yiye Ruan, David Fuhry, and Srinivasan Parthasarathy. 2013. Efficient community detection in large networks using content and links. In *Proceedings of the 22nd international conference on World Wide Web*. 1089–1098.

[21] John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. 2015. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems*. 3528–3536.

[22] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*. 1057–1063.

[23] Jiliang Tang, Charu Aggarwal, and Huan Liu. 2016. Node classification in signed social networks. In *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 54–62.

[24] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. *CoRR* abs/1503.03578 (2015). arXiv:1503.03578 http://arxiv.org/abs/1503.03578

[25] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. GraphGAN: Graph Representation Learning With Generative Adversarial Nets. In *AAAI Conference on Artificial Intelligence*. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16611

[26] Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. 2018. SHINE: signed heterogeneous information network embedding for sentiment link prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 592–600.

[27] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community Preserving Network Embedding.. In *AAAI*. 203–209.

[28] Ju Xiang, Tao Hu, Yan Zhang, Ke Hu, Jian-Ming Li, Xiao-Ke Xu, Cui-Cui Liu, and Shi Chen. 2016. Local modularity for community detection in complex networks. *Physica A: Statistical Mechanics and its Applications* 443 (2016), 451–459.

[29] Jierui Xie, Stephen Kelley, and Boleslaw K. Szymanski. 2011. Overlapping Community Detection in Networks: the State of the Art and Comparative Study. *CoRR* abs/1110.5813 (2011). arXiv:1110.5813 http://arxiv.org/abs/1110.5813

[30] Jaewon Yang and Jure Leskovec. 2012. Community-affiliation graph model for overlapping network community detection. In *2012 IEEE 12th International Conference on Data Mining*. IEEE, 1170–1175.

[31] Jaewon Yang and Jure Leskovec. 2013. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 587–596.

[32] Jaewon Yang, Julian McAuley, and Jure Leskovec. 2014. Detecting cohesive and 2-mode communities indirected and undirected networks. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 323–332.

[33] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient.. In *AAAI*. 2852–2858.

[34] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 283–292.

[35] Chuxu Zhang, Lu Yu, Yan Wang, Chirag Shah, and Xiangliang Zhang. 2017. *Collaborative User Network Embedding for Social Recommender Systems*. 381–389. https://doi.org/10.1137/1.9781611974973.43 arXiv:https://epubs.siam.org/doi/pdf/10.1137/1.9781611974973.43

[36] Hongyi Zhang, Irwin King, and Michael R. Lyu. 2015. Incorporating Implicit Link Preference into Overlapping Community Detection. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, 396–402. http://dl.acm.org/citation.cfm?id=2887007.2887063

Semantic Text Matching for Long-Form Documents

Jyun-Yu Jiang*
Department of Computer Science, University of California, Los Angeles, CA, USA
jyunyu@cs.ucla.edu

Mingyang Zhang
Google Inc., Mountain View, CA, USA
mingyang@google.com

Cheng Li
Google Inc., Mountain View, CA, USA
chgli@google.com

Michael Bendersky
Google Inc., Mountain View, CA, USA
bemike@google.com

Nadav Golbandi
Google Inc., Mountain View, CA, USA
nadavg@google.com

Marc Najork
Google Inc., Mountain View, CA, USA
najork@google.com

ABSTRACT

Semantic text matching is one of the most important research problems in many domains, including, but not limited to, information retrieval, question answering, and recommendation. Among the different types of semantic text matching, long-document-to-long-document text matching has many applications, but has rarely been studied. Most existing approaches for semantic text matching have limited success in this setting, due to their inability to capture and distill the main ideas and topics from long-form text.

In this paper, we propose a novel Siamese multi-depth attention-based hierarchical recurrent neural network (SMASH RNN) that learns the long-form semantics, and enables long-form document based semantic text matching. In addition to word information, SMASH RNN is using the document structure to improve the representation of long-form documents. Specifically, SMASH RNN synthesizes information from different document structure levels, including paragraphs, sentences, and words. An attention-based hierarchical RNN derives a representation for each document structure level. Then, the representations learned from the different levels are aggregated to learn a more comprehensive semantic representation of the entire document. For semantic text matching, a Siamese structure couples the representations of a pair of documents, and infers a probabilistic score as their similarity.

We conduct an extensive empirical evaluation of SMASH RNN with three practical applications, including email attachment suggestion, related article recommendation, and citation recommendation. Experimental results on public data sets demonstrate that SMASH RNN significantly outperforms competitive baseline methods across various classification and ranking scenarios in the context of semantic matching of long-form documents.

*Work done while interning at Google.

This paper is published under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 International (CC-BY-NC-ND 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW'19, May 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY-NC-ND 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313707

CCS CONCEPTS

• **Information systems** → **Document representation**; **Document structure**; • **Computing methodologies** → *Information extraction.*

KEYWORDS

Semantic text matching; long documents; hierarchical document structures; attention mechanism; recurrent neural networks.

ACM Reference Format:
Jyun-Yu Jiang, Mingyang Zhang, Cheng Li, Michael Bendersky, Nadav Golbandi, and Marc Najork. 2019. Semantic Text Matching for Long-Form Documents. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, Article 4, 12 pages. https://doi.org/10.1145/3308558.3313707

1 INTRODUCTION

Semantic text matching estimates semantic similarity between a source and a target text pieces (e.g., query-to-document match, question-to-paragraph match, etc.). Correctly modeling semantics in text matching has long been the "holy grail" of textual information retrieval. The difficulties of semantic matching are two-fold: first, semantics of words and phrases can be ambiguous; second, when the text is long, semantics of individual words, phrases and sentences can be buried in complex document structures. While the prior research mainly focuses on the first set of difficulties, in this paper we tackle the second challenge; namely that of dealing with complex long-form documents.

To better understand the effects of document length in semantic text matching, in Figure 1 we show applications of semantic text matching across the length spectrum of source and target texts. As shown on the upper left of this figure, ad-hoc retrieval tasks like web search [7] use short source queries while targeting long-form documents; on the lower left, short text retrieval tasks like Twitter search [9, 41] use short source queries and target short documents; on the lower right, the tasks of document classification like sentiment analysis aim to categorize long-form documents into a limited set of classes [25, 30]. Interestingly, the upper right part of the figure is relatively less explored in the semantic text match setting, and, as we empirically demonstrate, many of the previously proposed semantic matching methods deteriorate when source and target documents become longer. This poses an important research challenge, since semantic text matching for long-form documents

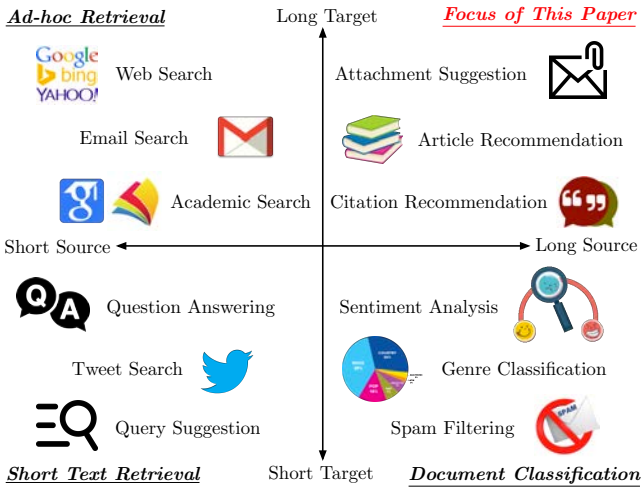


Figure 1: Applications across different lengths of source and target documents in text retrieval. In this work, we focus on the tasks with long sources and long targets.

can benefit a myriad of applications, such as related article recommendation, email attachment suggestion, citation recommendation, etc.

One of the most conventional approaches to semantic text matching is to compute a vector as the representation for each document, such as bag-of-words and latent Dirichlet allocation models [5, 18], and then apply typical similarity metrics to compute the matching scores. Unfortunately, the performance of traditional approaches is unsatisfactory, as they often fail to capture the semantic document structure. The advances in deep learning in recent years provide the opportunity to understand complex natural language and to learn better long-form document representations. For instance, recurrent neural networks (RNNs) treat a document as a sequence of words and derive a document representation based on the information along the sequence [32, 33]; convolutional neural networks (CNNs) obtain document representations by preserving local patterns [23, 38, 54].

However, although existing deep learning approaches significantly advanced the field of semantic text matching, they mainly focus on short documents and have apparent drawbacks when dealing with long-form documents. First, the core topic or idea can be hard to identify and extract from a complex narrative of a long-form document. Some of the previous studies [10, 52, 54] exploit attention mechanism [29] to distill important words from sentences, but valuable information can still be diluted within a large number of sentences and paragraphs in long-form documents. Second, the complex structural information of long-form documents have not yet been taken into account. Most of the existing approaches rely on word-level knowledge to compute text similarity. Structural information like relations among sentences and paragraphs is often disregarded. Third, the semantics of a document can drift over the course of a long narrative. For example, it is not uncommon to find documents in which the writer moves across a spectrum of subjects in the span of several passages. Neither RNN or CNN can naturally capture or follow semantic drifts like this. RNN-based approaches can attain confusing document representations by sequentially processing sentences with different semantics. CNN-based approaches can deteriorate when trying to pool and filter different semantics.

Natural language documents generally follow hierarchical structures [35] to help people read and understand them. Therefore, it is vital to utilize these structures in order to train machine learning models which can fully capture the semantics of long-form documents. Most generally, a document can be represented as a hierarchy of paragraph, sentence and word sequences. Different paragraphs and sentences in a document can have different semantic meaning and importance. The most similar research to this work is to use sentence-level information for document classification [10, 52]. As we will show in the experimental section, for long-form documents, sentence-level based document representations are still unsatisfactory because sentences in the same document may be associated with different importance and diverse semantics. On the contrary, a deep understanding of document structure can effectively boost semantic text matching.

In this paper, we propose Siamese multi-depth attention-based hierarchical RNN (SMASH RNN) to address the problem of long-form document semantic matching. Under the two-tower structure of Siamese network [32], each tower of the proposed model is a multi-depth attention-based hierarchical RNN (MASH RNN). MASH RNN, as the major component of our model, can derive comprehensive document representations from multiple levels of document structure. For example, word-, sentence-, and paragraph-level knowledge of a document can be derived by three attention-based hierarchical RNNs with different depths. To generate comprehensive document presentations, MASH RNN concatenates representations from all of these document levels, aiming to capture both concrete low-level observations and abstract high-level insights. Combining the document representations for both source and target documents from MASH RNN, SMASH RNN estimates semantic matching scores based on the representations of the source and the target documents, and an extra fully connected layer.

Our contributions can be summarized as follows:

- To the best of our knowledge, this paper is the first work to extensively exploit document structure for better document representations, in the context of improving the state-of-the-art performance of the long-form document semantic text matching models.
- We propose the SMASH RNN framework for long-form document semantic text matching. MASH RNN, the major component of SMASH RNN, learns document representations from multiple abstraction levels of the document structure.
- Experiments were conducted on publicly available datasets for three different applications: email attachment suggestion, related article recommendation, and citation recommendation. Experiment results demonstrate the effectiveness of SMASH RNN. We also provide an in-depth experiment analysis to prove the robustness of our proposed framework.

The rest of the paper is organized as follows. First, we present related work in Section 2. Then, the problem statement and SMASH RNN are described in Section 3. We show experiment results and provide an in-depth analysis in Section 4. Finally, some conclusions are given in Section 5.

2 RELATED WORK

In this section, we first give the background of semantic text matching and indicate the difference between our approach and previous studies. Deep document classification is then introduced as a related task. Last but not least, we discuss the attention mechanism in deep learning, which plays an important role in SMASH RNN.

2.1 Semantic Text Matching

To measure the similarity between documents, it is intuitive for traditional approaches to compare the words in the documents. For example, Mihalcea et al. [31] compute the word-to-word similarity while Wu et al. [49] exploit the vector space model with the term frequency-inverse document frequency (TF-IDF). However, words in the documents are extremely sparse. In addition, the semantics between individual words also cannot be captured, so these approaches usually obtain unsatisfactory results. Although some works attempt to leverage the semantics in external resources such as knowledge bases [42] or alleviate the data sparseness by Latent Semantic Analysis (LSA) [53], traditional approaches are still limited by discrete words.

The recent development of deep learning in natural language processing provides a new opportunity for semantic text matching. After using deep learning models to encode the documents in distributed representations, the Siamese structure [8] for metric learning is usually applied to learn the similarity information. There is a plethora of deep learning models for encoding documents. For instance, some studies [2, 27, 32, 33, 45, 46] exploit RNNs to model the sequential information of documents while other work [19, 38, 46, 54] applies CNNs to capture the important local messages in documents for semantic text matching.

However, most of the previous studies are not designed for long-form documents. For example, RNNs can lose important messages while passing information along a very long sequence of words. The local features learned by CNNs can be inadequate to represent the complex semantics of long-form documents.

Another line of work maps the term positions of the source and the target texts to create a matching matrix [27, 45, 46, 50, 54] or computes local interactions with the target text for each word in the source text [16]. This works well for scenarios like ad-hoc retrieval, where the source queries are short. For a scenario where both the source and the target are long documents, a matching matrix would be time-consuming to compute, and is too large to fit in memory.

In addition, the document structures that could be much beneficial are generally ignored by most of the previous semantic text matching approcaches. Note that although Liu et al. [27] mention the term "hierarchical structure," they focus on the structures in sentences so that long-form documents still cannot be handled. In other words, their proposed methods are incapable to deal with long-form documents. Our approach differs from this work in that the hierarchical document structures with different depths are explicitly considered in the model, thereby directly benefiting the model to learn the semantics of different levels in the document structure.

2.2 Deep Document Classification

Instead of simultaneously modeling two documents, deep document classification that considers a single document at a time with deep learning models can be considered as a related task. Similar to semantic text matching, deep learning models can also be utilized to encode documents into distributed representations for document classification. For example, both CNNs [13, 25, 39] and RNNs [10, 28, 52] can do document classification for many applications such as sentiment analysis. Devlin et al. [12] propose bidirectional encoder representations from transformers (BERT). However, all of these works focus on short text classification, especially for sentences and documents with few sentences. Even though BERT quickly became the state-of-the-art for many NLP tasks, its multi-head attention component requires an unrealistically enormous amount of both memory and computation time for long-form documents. Moreover, most of their approaches only take word-level information into account and ignore the structure of long-form documents. HAN [52] and NSC [10] with the exactly same architecture are the only two studies using a two-layer RNN with attention to consider sentence-level information for document classification. On one hand, a document narrative can contain a large sequence of loosely related sentences. Grouping of those sentences into paragraphs or sections is not handled by these models. On the other hand, these models only consider the sentence-level knowledge, but the word-level information that would be diluted in the two-layer model structure can be also important. Therefore, existing approaches of deep document classification cannot appropriately capture the semantics of long-form documents.

2.3 Attention Mechanism in Deep Learning

The attention mechanism [4, 29] has already become one of the most important techniques in deep learning since its huge success in machine translation [44]. Given an input sequence, the attention mechanism infers the importance of each position with a learnable context vector. Besides machine translation, the attention mechanism benefits many applications, such as caption generation [51], document classification [10, 52], and question answering [40]. In this paper, the attention mechanism is exploited to extract the important knowledge in each individual branch of the hierarchical structure for a long-form document.

3 SEMANTIC TEXT MATCHING FOR LONG-FORM DOCUMENTS WITH SMASH RNN

In this section, we first formally define the objective of this paper, and then present the proposed framework, Siamese multi-depth attention-based hierarchical RNN (SMASH RNN), to address the problem of long-form document semantic text matching.

3.1 Problem Statement

In this paper, we focus on long-form documents that can be represented as a hierarchical compositions of word. Note that hierarchy depths for documents can be vary, depending on the highest level of abstraction for a document structure. To facilitate readability, we assume that there are three levels in hierarchy – paragraphs,

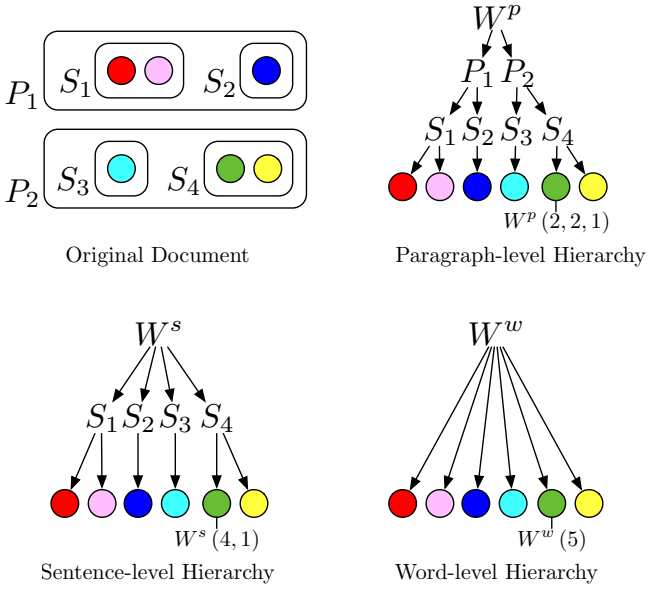


Figure 2: The illustration of hierarchical structures with different depths for an example document. P_k and S_j are the structures of paragraphs and sentences. The positions of a word in the structures depend on the depths of hierarchies.

sentences and words. However, it is important to note that our proposed method is general enough to handle variable depths of hierarchies. Figure 2 gives an illustration of hierarchical structures with different depths for a document d. Words in the d can be fitted into three hierarchical structures, i.e., W^p, W^s, and W^w, with depth 3 (paragraph-level), depth 2 (sentence-level), and depth 1 (word-level) respectively. More precisely, $W^p\,(k, j, i)$ is the i-th word in the j-th sentence of the k-th paragraph given a paragraph-level hierarchy; $W^s\,(j, i)$ is the i-th word in the j-th sentence given a sentence-level hierarchy; $W^w\,(i)$ is the i-th word for the word-level hierarchy with a depth of 1, which is just a long sequence. Note that the bottom level words in the three different hierarchies are exactly identical while their annotations vary according to hierarchy depth and document structure.

Given a source document d_s and a set of candidate documents D_c, our goal is to estimate semantic similarity $\hat{y} = Sim(d_s, d_c)$ between the source document d_s and every candidate document $d_c \in D_c$ so that the target documents semantically matched to the source document have higher semantic similarity scores.

3.2 Framework Overview

Figure 3 and 4 illustrate the framework of our proposed Siamese multi-depth attention-based hierarchical RNN (SMASH RNN). Under the Siamese structure [32], each SMASH RNN has two multi-depth attention-based hierarchical RNN (MASH RNN) towers. For each document, MASH RNN derives an informative representation based on the knowledge from different levels of document structure. For each level, an attention-based hierarchical RNN (with corresponding level depth) is constructed as an encoder to generate representations for that level. For example, the paragraph-level encoder produces paragraph-level representations with a depth-3 encoder while the sentence-level encoder produces sentence-level representations with a depth-2 encoder. The final document representation is then acquired by concatenating the representations of different levels, comprehensively covering the knowledge in all document structure levels. To estimate semantic similarity for semantic text matching, SMASH RNN adopts the Siamese structure with two MASH RNN towers. Given representations generated by MASH RNN for both the source and target documents, a fully-connected layer with nonlinearity infers a probabilistic score to examine the semantic relation between two documents with a sigmoid function [34].

In Section 3.3, we formally define MASH RNN. In Section 3.4, we put two MASH RNN towers together to define SMASH RNN.

3.3 MASH RNN for Document Representation

Most of the previous studies only exploit word-level information to represent documents [25, 32, 38, 54]. To investigate deeper structures, some work [10, 52] uses sentence-level information to represent documents. However, only using sentence-level information could lead to a loss of word level information. More than that, a long-form document usually contains quite a number of sentences, and its structure is usually deeper than sentence level. The deeper document structure was not modeled in previous studies for semantic text matching.

In this paper, we propose to model documents with information from different document structure levels. To ease the discussion, we focus on three levels of document structure – paragraph, sentence, and word-level, as mentioned in Section 3.1.

The computation of encoders in MASH RNN follows a bottom-up principle with bidirectional recurrent neural networks (Bi-RNNs) with attention. Take the paragraph-level encoder as an example. Given the j-th sentence in the k-th paragraph in the paragraph-level hierarchy W^p, we first embed words in sentence to vectors through a word embedding layer as follows:

$$\boldsymbol{x}^{\boldsymbol{p}}_{\boldsymbol{k,j,i}} = emb(W^p(k, j, i)), 1 \le i \le L^p_{k,j},$$

where $L^p_{k,j}$ is the length of the sentence, and the word embedding layer $emb(\cdot)$ embeds the words into vectors with an embedding matrix. To encode the sentence, a Bi-RNN reads the sequence of embedding vectors during both the forward pass and the backward pass. In the forward pass, the Bi-RNN creates a sequence of forward hidden states

$$\overrightarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j}}} = \left[\overrightarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j,1}}}, \overrightarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j,2}}}, \cdots, \overrightarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j,L^p_{k,j}}}}\right],$$

where $\overrightarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j,i}}} = \text{RNN}\left(\overrightarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j,i-1}}}, \boldsymbol{x}^{\boldsymbol{p}}_{\boldsymbol{k,j,i}}\right)$ is generated by a dynamic function such as LSTM [21] or GRU [11]. Here we use GRU instead of LSTM because it requires fewer parameters [24]. The backward pass processes the input sequence in reverse order and generates the backward hidden states as

$$\overleftarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j}}} = \left[\overleftarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j,1}}}, \overleftarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j,2}}}, \cdots, \overleftarrow{\boldsymbol{h}^{\boldsymbol{p}}_{\boldsymbol{k,j,L^p_{k,j}}}}\right],$$

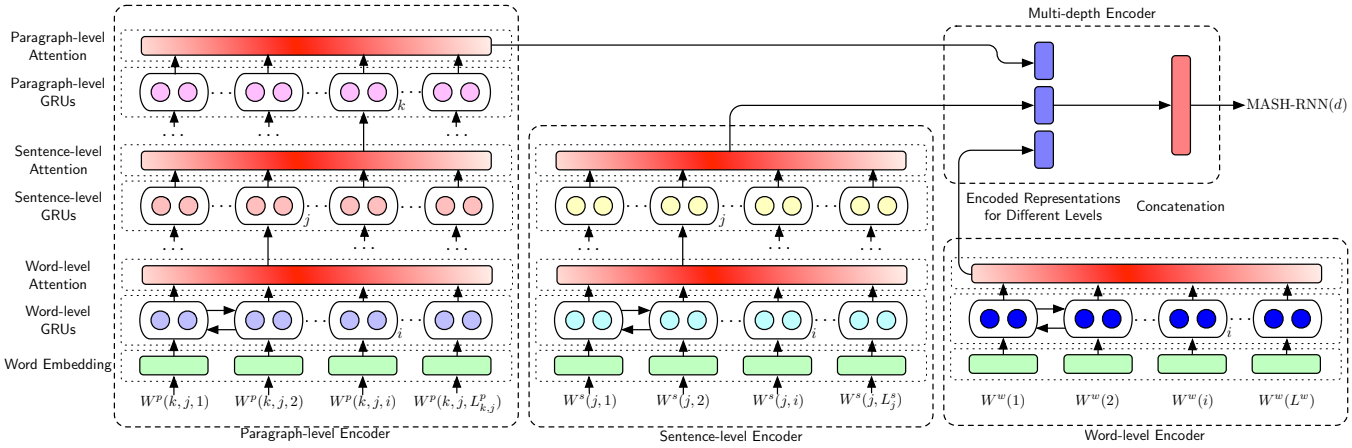


Figure 3: The schema of multi-depth attention-based hierarchical RNN (MASH RNN).

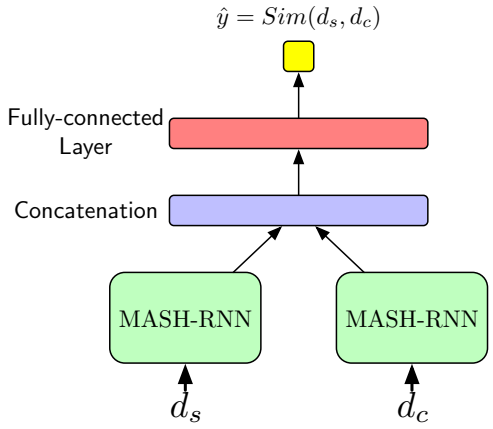


Figure 4: The architecture of Siamese multi-depth attention-based hierarchical RNN (SMASH RNN).

where $\overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k,j,i}}} = \text{RNN}\left(\overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k,j,i+1}}}, \boldsymbol{x}^p_{\boldsymbol{k,j,i}}\right)$. The forward and backward hidden states are concatenated as a hidden representation for words in the sentence

$$\boldsymbol{h}^p_{\boldsymbol{k,j}} = \left[\boldsymbol{h}^p_{\boldsymbol{k,j,1}}, \boldsymbol{h}^p_{\boldsymbol{k,j,2}}, \cdots, \boldsymbol{h}^p_{\boldsymbol{k,j,L^p_{k,j}}}\right],$$

where $\boldsymbol{h}^p_{\boldsymbol{k,j,i}} = \left[\overrightarrow{\boldsymbol{h}^p_{\boldsymbol{k,j,i}}}; \overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k,j,i}}}\right]$.

Since each word can have unequal importance in the sentence, the attention mechanism [4] is applied to extract and aggregate hidden representations that are more important. More precisely, the representation $\boldsymbol{h}^p_{\boldsymbol{k,j,i}}$ will be transformed by a fully-connected hidden layer to measure the importance $\alpha^p_{k,j,i}$ as follows:

$$\alpha^p_{k,j,i} = \frac{\exp\left(\boldsymbol{u}^p_{\boldsymbol{k,j,i}} \cdot \boldsymbol{u}^p_{\boldsymbol{w}}\right)}{\sum_{i'} \exp\left(\boldsymbol{u}^p_{\boldsymbol{k,j,i'}} \cdot \boldsymbol{u}^p_{\boldsymbol{w}}\right)},$$

in which $\boldsymbol{u}^p_{\boldsymbol{k,j,i}} = \tanh\left(\mathcal{F}^p_w\left(\boldsymbol{h}^p_{\boldsymbol{k,j,i}}\right)\right)$; $\mathcal{F}^p_w(\cdot)$ is a fully-connected layer; tanh is the activation for the convenience of similarity computation; $\boldsymbol{u}^p_{\boldsymbol{w}}$ is a vector to measure the word importance. The normalized importance $\alpha^p_{k,j,i}$ can be further obtained through a softmax function. Finally, the representation of the j-th sentence in the k-th paragraph can be represented as the weighted sum of the hidden representations as follows:

$$\boldsymbol{s}^p_{\boldsymbol{k,j}} = \sum_i \alpha^p_{k,j,i} \boldsymbol{h}^p_{\boldsymbol{k,j,i}}.$$

With the representations of sentences, the Bi-RNN is applied again to learn the paragraph representations because a paragraph can be considered as a sequence of sentences. Given the representations of sentences in the k-th paragraph $\boldsymbol{s}^p_{\boldsymbol{k,j}}$, a Bi-RNN generates the forward and backward hidden states of sentences as follows:

$$\overrightarrow{\boldsymbol{h}^p_{\boldsymbol{k}}} = \left[\overrightarrow{\boldsymbol{h}^p_{\boldsymbol{k,1}}}, \overrightarrow{\boldsymbol{h}^p_{\boldsymbol{k,2}}}, \cdots, \overrightarrow{\boldsymbol{h}^p_{\boldsymbol{k,L^p_k}}}\right], \overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k}}} = \left[\overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k,1}}}, \overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k,2}}}, \cdots, \overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k,L^p_k}}}\right],$$

where L^p_k is the number of sentences in the k-th paragraph; $\overrightarrow{\boldsymbol{h}^p_{\boldsymbol{k,j}}} = \text{RNN}\left(\overrightarrow{\boldsymbol{h}^p_{\boldsymbol{k,j-1}}}, \boldsymbol{s}^p_{\boldsymbol{k,j}}\right)$; $\overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k,j}}} = \text{RNN}\left(\overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k,j+1}}}, \boldsymbol{s}^p_{\boldsymbol{k,j}}\right)$. The hidden representations for the sentences in the paragraph can then be represented as

$$\boldsymbol{h}^p_{\boldsymbol{k}} = \left[\boldsymbol{h}^p_{\boldsymbol{k,1}}, \boldsymbol{h}^p_{\boldsymbol{k,2}}, \cdots, \boldsymbol{h}^p_{\boldsymbol{k,L^p_k}}\right],$$

where $\boldsymbol{h}^p_{\boldsymbol{k,j}} = \left[\overrightarrow{\boldsymbol{h}^p_{\boldsymbol{k,j}}}; \overleftarrow{\boldsymbol{h}^p_{\boldsymbol{k,j}}}\right]$. With the attention mechanism, the importance of each sentence can be measured as follows:

$$\alpha^p_{k,j} = \frac{\exp\left(\boldsymbol{u}^p_{\boldsymbol{k,j}} \cdot \boldsymbol{u}^p_{\boldsymbol{s}}\right)}{\sum_{j'} \exp\left(\boldsymbol{u}^p_{\boldsymbol{k,j'}} \cdot \boldsymbol{u}^p_{\boldsymbol{s}}\right)},$$

where $\boldsymbol{u}^p_{\boldsymbol{k,j}} = \tanh\left(\mathcal{F}^p_s\left(\boldsymbol{h}^p_{\boldsymbol{k,j}}\right)\right)$; $\mathcal{F}^p_s(\cdot)$ and $\boldsymbol{u}^p_{\boldsymbol{s}}$ are the fully-connected layer and the vector for the importance measurement. Finally, the representation of the k-th paragraph can be represented as

$$\boldsymbol{p}^p_{\boldsymbol{k}} = \sum_j \alpha^p_{k,j} \boldsymbol{h}^p_{\boldsymbol{k,j}}.$$

As the top structure in the paragraph-level hierarchy, the document can be treated as a sequence of paragraphs, so we can also utilize the Bi-RNN to generate paragraph-level document representations. Given the representations of paragraphs in the document $\boldsymbol{p}_{\boldsymbol{k}}^{\boldsymbol{p}}$, the forward and backward hidden states of paragraphs in the Bi-RNN are generated as follows:

$$\overrightarrow{\boldsymbol{h}^{\boldsymbol{p}}} = \left[\overrightarrow{\boldsymbol{h}_1^{\boldsymbol{p}}}, \overrightarrow{\boldsymbol{h}_2^{\boldsymbol{p}}}, \cdots, \overrightarrow{\boldsymbol{h}_{L^p}^{\boldsymbol{p}}}\right], \overleftarrow{\boldsymbol{h}^{\boldsymbol{p}}} = \left[\overleftarrow{\boldsymbol{h}_1^{\boldsymbol{p}}}, \overleftarrow{\boldsymbol{h}_2^{\boldsymbol{p}}}, \cdots, \overleftarrow{\boldsymbol{h}_{L^p}^{\boldsymbol{p}}}\right],$$

where L^p is the number of paragraphs in the document; $\overrightarrow{\boldsymbol{h}_{\boldsymbol{k}}^{\boldsymbol{p}}} = \text{RNN}\left(\overrightarrow{\boldsymbol{h}_{\boldsymbol{k-1}}^{\boldsymbol{p}}}, \boldsymbol{p}_{\boldsymbol{k}}^{\boldsymbol{p}}\right)$; $\overleftarrow{\boldsymbol{h}_{\boldsymbol{k}}^{\boldsymbol{p}}} = \text{RNN}\left(\overleftarrow{\boldsymbol{h}_{\boldsymbol{k+1}}^{\boldsymbol{p}}}, \boldsymbol{p}_{\boldsymbol{k}}^{\boldsymbol{p}}\right)$. The hidden representations for the paragraphs can be represented as

$$\boldsymbol{h}^{\boldsymbol{p}} = \left[\boldsymbol{h}_1^{\boldsymbol{p}}, \boldsymbol{h}_2^{\boldsymbol{p}}, \cdots, \boldsymbol{h}_{L^p}^{\boldsymbol{p}}\right],$$

where $\boldsymbol{h}_{\boldsymbol{k}}^{\boldsymbol{p}} = \left[\overrightarrow{\boldsymbol{h}_{\boldsymbol{k}}^{\boldsymbol{p}}}; \overleftarrow{\boldsymbol{h}_{\boldsymbol{k}}^{\boldsymbol{p}}}\right]$. The importance of each paragraph can then be measured with the attention mechanism as follows:

$$\alpha_k^p = \frac{\exp\left(\boldsymbol{u}_{\boldsymbol{k}}^{\boldsymbol{p}} \cdot \boldsymbol{u}_{\boldsymbol{p}}^{\boldsymbol{p}}\right)}{\sum_{p'} \exp\left(\boldsymbol{u}_{\boldsymbol{k}'}^{\boldsymbol{p}} \cdot \boldsymbol{u}_{\boldsymbol{p}}^{\boldsymbol{p}}\right)},$$

in which $\boldsymbol{u}_{\boldsymbol{k}}^{\boldsymbol{p}} = \tanh\left(\mathcal{F}_p^p\left(\boldsymbol{h}_{\boldsymbol{k}}^{\boldsymbol{p}}\right)\right)$; $\mathcal{F}_p^p(\cdot)$ and $\boldsymbol{u}_{\boldsymbol{p}}^{\boldsymbol{p}}$ are the fully-connected layer and the vector for the importance measurement. The paragraph-level document representation can finally be represented as the output of the paragraph-level encoder as follows:

$$\boldsymbol{d}^{\boldsymbol{p}} = \sum_k \alpha_k^p \boldsymbol{h}_{\boldsymbol{k}}^{\boldsymbol{p}}.$$

Besides the paragraph-level encoder and paragraph-level representation $\boldsymbol{d}^{\boldsymbol{p}}$, with shallower depths of hierarchy W^s and W^w, we create sentence-level and word-level encoders and generate sentence-level and word-level representations $\boldsymbol{d}^{\boldsymbol{s}}$ and $\boldsymbol{d}^{\boldsymbol{w}}$.

In order to capture both concrete low-level observations and abstract high-level insights, MASH RNN uses representations from different document levels. Specifically, MASH RNN returns final document representation as a concatenation of representations generated by multi-depth encoders:

$$\boldsymbol{d} = \left[\boldsymbol{d}^{\boldsymbol{p}}; \boldsymbol{d}^{\boldsymbol{s}}; \boldsymbol{d}^{\boldsymbol{w}}\right].$$

3.4 SMASH RNN for Semantic Text Matching

The Siamese structure associated with two identical sub-networks were shown to be effective in measuring the affinity between representations of two documents modeled in the same hidden space [32, 38, 47, 54]. To address the problem of semantic text matching for long-form documents, we propose the Siamese multi-depth attention-based hierarchical RNN (SMASH RNN) using a Siamese structure that fuses the outputs of two MASH RNNs.

Figure 4 illustrates the structure of SMASH RNN for estimating the semantic similarity between a source document d_s and a candidate document d_c. To tackle two sub-networks, there are several approaches, such as using an attention matrix [54] or a similarity matrix [38]. However, long-form documents require an enormous number of parameters with these methods. Here we propose to utilize the concatenation of representations with a fully-connected layer to learn an appropriate way for computing the semantic similarity. More formally, given $\boldsymbol{d}_{\boldsymbol{s}}$ and $\boldsymbol{d}_{\boldsymbol{c}}$, which are the representations generated by MASH RNN for two documents, the final feature vector can be represented as $\boldsymbol{x}_{\boldsymbol{f}} = [\boldsymbol{d}_{\boldsymbol{s}}; \boldsymbol{d}_{\boldsymbol{c}}]$. The semantic similarity between two documents can be computed as follows:

$$\boldsymbol{u}_{\boldsymbol{f}} = \text{ReLU}\left(\mathcal{F}_d\left(\boldsymbol{x}_{\boldsymbol{f}}\right)\right),$$

$$\hat{y} = \sigma\left(\mathcal{F}_f\left(\boldsymbol{u}_{\boldsymbol{f}}\right)\right),$$

where $\mathcal{F}_d(\cdot)$ and $\mathcal{F}_f(\cdot)$ are two fully-connected hidden layers; ReLU$(\cdot)$ is the activation function for the hidden layer; $\sigma(\cdot)$ is the logistic sigmoid function for generating probabilistic scores as similarities [17].

Note that although some of the previous studies [23, 33, 38, 53] suggest to exploit predefined similarity functions, such as the Manhattan distance and the cosine similarity, to directly measure the affinity between representations, we found that predefined similarity functions do not work with long-form documents. This can be because the predefined functions are too simple to estimate the complex semantic relations among long-form documents.

3.5 Learning and Optimization

The task of semantic text matching can be modeled as a binary classification problem. Given a tuple of training data (d_s, d_c, y), where y is a Boolean value showing whether two documents are semantically matched, SMASH RNN optimizes the binary cross-entropy [20] between the estimated probabilistic score $\hat{y}$ and the gold standard y. More formally, the loss function can be written as

$$-(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})).$$

Moreover, the architecture of SMASH RNN allows end-to-end learning [55] that directly trains the full model from scratch using the existing data for a given task. Neither additional data engineering nor multi-step optimization are required.

4 EXPERIMENTS

In this section, we conduct extensive experiments and in-depth analysis with large-scale real-world datasets.

4.1 Experimental Settings

In the experiments, we verify the performance of SMASH RNN in three real-world applications of semantic text matching for long-form documents, including (1) email attachment suggestion, (2) related article recommendation, and (3) citation recommendation. More precisely, these experimental tasks are not only practical but also involving different types of long-form documents for validating the robustness of SMASH RNN. The experimental settings about the datasets employed in each task are described in the following corresponding subsection.

Evaluation. The performance of semantic text matching is mainly evaluated with standard classification metrics, including accuracy, precision, recall, and F1-score [15]. For the task of email attachment suggestion, following prior work [43], we also conduct a ranking experiment, which is evaluated with three standard information

retrieval metrics, including precision at 1 (P@1), mean reciprocal rank (MRR), and mean average precision (MAP) [3].

Implementation Details. The model is implemented in Tensorflow [1]. The Adam optimizer [26] is applied to optimize the parameters with an initial learning rate of 10^{-5}. The batch size is set as 32. Note that

a large batch size can lead to enormous memory consumption because long-form documents consists of

Note that because a long-form document can have an enormous amount of words, the size of a batch cannot be so large due to the limitation of memory usage. The number of hidden neurons in GRUs and hidden layers is set as 128, and the number of dimensions for word embeddings is 256.

Baseline Methods. For all of the tasks, we compare with the following baseline methods with and without using document structures to evaluate the performance of SMASH RNN.

- *RNN-based approach* (RNN) [32, 33] exploits an RNN to model each document as a word sequence. The Siamese structure is then applied to measure the relations between documents. It is the representative of the approaches based on word-level RNNs.
- *CNN-based approach* (CNN) [25] integrates the word embeddings into an embedding matrix and applies several convolutional filters to extract representative features with a pooling layer that covers the whole document. More precisely, a Siamese structure infers the semantic similarity with local features learned by convolutional filters. It can be also treated as the representative of the approaches based on word-level CNNs.
- *CNN with a matching matrix* (DeepQA) [38] enhances the CNN-based approach for question answering. In addition to the convolutional features, DeepQA uses a matching matrix [6] to measure an asymmetric similarity between the features of two documents with a noisy channel [14]. After joining the convolutional features of two documents and their asymmetric similarity, a fully-connected layer generates the semantic similarity. It can also be treated a CNN-based method using word-level information.
- *Hierarchical Attention Network* (HAN) [52] is the only baseline method that considers the document structure information. For each sentence, HAN applies an attention-based RNN to generate a feature vector, thereby deriving the sentence-level document representations with the other RNN. Note that HAN only obtains the sentence-level features for a document. The paragraph-level information is ignored, and the word-level knowledge can be diluted after being passed in the hierarchical model. More specifically, HAN is a special case of MASH RNN using only sentence-level encoders.

For the Siamese structure, although many studies use similarity functions to combine features, such as Manhattan distance [32] and cosine similarity [33], we found all of these methods perform worse than the methods using concatenation with a hidden layer for measuring the relations between long-form documents. Hence, we simply modify the baseline models by applying the Siamese structure shown in Section 3.4 instead of similarity functions to aggregate two document representations. Note that we do not compare with previous works incapable of processing long-form documents, such as ABCNN [54], DRMM [16], and BERT [12]. ABCNN learns an attention matrix for arbitrary position mappings, which is too large to be fitted in memory for long-form documents. DRNN requires an enormous amount of both memory and computational time to compute the local interaction between every words in the source document and the target document. BERT is also memory-inefficient and time-consuming for long-form documents, and its pretraining can only handle at most 512 tokens for a document.

For simplicity, our proposed framework SMASH RNN is denoted as SMASH in experimental results. In addition, P, S, and W in the tables indicate paragraph-, sentence-, and word-level hierarchies utilized in MASH RNN.

Table 1: The statistics of examples in the training, validation, and testing datasets for email attachment suggestion.

Dataset	Training	Validation	Testing
Period of Emails	18 months	2 months	1 month
Number of Examples	49,102	6,950	3,650

4.2 Task 1: Email Attachment Suggestion

The first application of semantic text matching for long-form documents in the experiments is email attachment suggestion. Given the content of an email and a candidate document, the goal of email attachment suggestion to classify whether the document should be an attachment for the email. If the system can precisely discriminate the attachments, the system will be able to automatically suggest those attachments, thereby saving users from spending lots of time on searching and attaching documents. This task is also evaluated as a ranking problem that aims to rank candidate documents for a given email.

Experimental Datasets. We adopt the largest publicly available Avocado Research Email Collection [36] as the experimental dataset for email attachment suggestion. The dataset is a collection of emails and attachments taken from a defunct information technology company. The emails sent within the densest 21-month period from January 2000 to September 2001 are selected for experiments. To partition the emails into training, validation, and testing sets, the first 18-month emails are the training data while the following 2-month emails are utilized for validation. The remaining 1-month emails are treated as the testing set. For attachments, we simply remove non-natural language attachments such as programming code or electronic business cards, to construct a pool of candidate attachments. For each pair of an email and its attachment, we extract the pair as a positive example. For each positive example, we randomly sample an irrelevant document from the candidate pool as a negative example to form a balanced dataset. Finally, there are 26,589 attachments in the pool of candidate attachments after filtering non-retrievable files. Table 1 shows the statistics of examples in the experimental datasets for email attachment suggestion.

Experimental Results. Table 2 shows the classification performance of the baseline methods and the proposed SMASH method with combinations of hierarchies used in MASH RNN.

For the baseline methods, the methods using only the word-level knowledge, i.e., RNN, CNN, and DeepQA, have similar performance. HAN surpasses all of the other baseline methods because it considers the sentence-level knowledge. As the proposed method in this

Table 2: The classification performance of email attachment suggestion. All improvements of our methods denoted as (*) are significant differences over the HAN method at 99% level in both of a paired t-test and a permutation test.

Method	Accuracy	Precision	Recall	F1
RNN [32, 33]	0.5594	0.5772	0.4439	0.5018
CNN [25]	0.5612	0.5694	0.5024	0.5338
DeepQA [38]	0.5618	0.5990	0.3740	0.4605
HAN [52]	0.5900	0.6096	0.5003	0.5496
SMASH (P)	0.5878	0.5895	0.5783*	0.5839*
SMASH (P+S)	0.6363*	0.6225*	0.6927*	0.6557*
SMASH (P+S+W)	**0.6718***	**0.6440***	**0.7686***	**0.7008***

Table 3: The ranking performance of email attachment suggestion. All improvements of our methods denoted as (*) are significant differences over the HAN method at 99% level in both of a paired t-test and a permutation test.

Method	P@1	MRR	MAP
RNN [32, 33]	0.1571	0.3557	0.3515
CNN [25]	0.1451	0.3499	0.3475
DeepQA [38]	0.1774	0.3634	0.3567
HAN [52]	0.1827	0.3724	0.3658
SMASH (P)	0.1534	0.3653	0.3638
SMASH (P+S)	0.2444*	0.4537*	0.4480*
SMASH (P+S+W)	**0.2692***	**0.4900***	**0.4845***

paper, SMASH significantly outperforms all of the baseline methods. More precisely, SMASH using all-level knowledge achieves 13.86% and 27.51% improvements against HAN in the metrics of accuracy and F1-score. While using only the paragraph-level hierarchy, SMASH (P) has a similar accuracy and a better F1-score compared to HAN. After accordingly adding sentence- and word-level knowledge, SMASH (P+S) and SMASH (P+S+W) have further improved performance. It demonstrates that all of the hierarchies in different levels are beneficial for email attachment suggestion. In addition, each of the hierarchies holds different information, so they can jointly enhance SMASH.

Aside from classification, the predicted semantic similarity as a numeric value can be also utilized for ranking candidate attachments. For each email, we randomly sample irrelevant documents to construct a list with ten candidate attachments. Table 3 demonstrates the ranking performance of different methods. The experimental results are consistent with the classification experiments. As a result, SMASH achieves 47.35% and 32.45% improvements of P@1 and MAP against HAN. It indicates that the semantic similarities generated by SMASH not only discriminate the attachments but are also predictive of their relative relevance scores.

4.3 Task 2: Related Article Recommendation

The second application in the experiments is related article recommendation. Given a pair of long-form articles, the goal of related article recommendation is to classify if the target article is relevant to the source article. This application can contribute to many real-world scenarios. For example, when a user reads a long Wikipedia page, the system can automatically push other related pages for a more comprehensive coverage of the topic; it can be also helpful for recommending news articles about related events.

Table 4: The statistics of examples in the training, validation, and testing datasets for related article recommendation.

Dataset	Training	Validation	Testing
% of Source Articles	80%	10%	10%
Number of Examples	65,948	8,166	8,130

Table 5: The classification performance of related article recommendation. All improvements of our methods denoted as (*) are significant differences over the HAN method at 99% level in both of a paired t-test and a permutation test.

Method	Accuracy	Precision	Recall	F1
RNN [32, 33]	0.7430	0.7328	0.7647	0.7484
CNN [25]	0.6714	0.7204	0.5601	0.6302
DeepQA [38]	0.7366	0.7197	0.7749	0.7463
HAN [52]	0.8089	0.7521	**0.9234**	0.8290
SMASH (P)	0.8047	0.7499	0.9120	0.8230
SMASH (P+S)	**0.8219***	**0.7987***	0.8677	**0.8318***
SMASH (P+S+W)	0.8144*	0.7626*	0.9137	0.8313*

Experimental Datasets. The Wikipedia [48] is adopted for the experiments on related article recommendation. 10% of the entity pages in Wikipedia are randomly sampled to build the corpus. Since there is no gold standard about the relatedness of Wikipedia articles, we use links as a source of weak supervision. First, we assume that similar articles have similar sets of outgoing links. Based on this assumption, the Jaccard similarity [22] between the outgoing links of two articles measures their *pseudo similarity*. The article pairs with pseudo similarities greater than a threshold 0.5 are considered as the positive examples. Moreover, we define the article that has the lexicographically smaller URL as the source article of the pair, for the convenience of partitioning datasets and avoiding duplicate examples. For each positive example, we randomly sample a mismatched article from the outgoing links of the source article to generate a negative example. Note that the mismatched articles are not sampled from the entire corpus because those pages will be too irrelevant to make the task challenging enough for differentiating the performance of the various methods. The examples with 80% of the source articles are the training set when each of the validation and testing sets is generated by 10% of the articles. Table 4 shows the statistics of the experimental datasets for related article recommendation.

Experimental Results. Table 5 demonstrates the performance of the different methods for related article recommendation. CNN-based methods, i.e., CNN and DeepQA, perform worse than other methods using RNNs. It may be due to the fact that Wikipedia articles are more structural so that convolutional features and the pooling layer cannot capture the document organization. RNN is slightly better than DeepQA since it captures the word dependencies in documents. HAN that considers the sentence-level structure

Table 6: The statistics of examples in the training, validation, and testing datasets for citation recommendation.

Dataset	Training	Validation	Testing
% of Source Papers	80%	10%	10%
Number of Examples	169,346	20,742	20,358

Table 7: The performance of citation recommendation. All improvements of our methods against the HAN method are significant differences at 99% level in both of a paired t-test and a permutation test.

Method	Accuracy	Precision	Recall	F1
RNN [32, 33]	0.7352	0.7358	0.7339	0.7348
CNN [25]	0.7309	**0.8048**	0.6097	0.6938
DeepQA [38]	0.7410	0.7579	0.7084	0.7323
HAN [52]	0.7813	0.7454	**0.8544**	0.7962
SMASH (P)	0.7739	0.7532*	0.8149	0.7828
SMASH (P+S)	**0.8068***	0.8019*	0.8150	**0.8084***
SMASH (P+S+W)	0.8058*	0.8038*	0.8092	0.8065*

is still the best baseline method for the task. Similar to the results of the previous task, the performance of HAN and SMASH with only the paragraph-level hierarchy is similar. When SMASH considers multi-level knowledge, it outperforms all of the baseline methods. It is also worth mentioning that although SMASH (P+S+W) is better than HAN, it does not improve SMASH (P+S) by adding the word-level hierarchy. It indicates the limitations of word-level information for capturing the semantics of long articles in Wikipedia.

4.4 Task 3: Citation Recommendation

Last but not least, the third application in the experiments is citation recommendation for academic papers. Given the content of an academic paper and the other paper as a candidate citation, we aim to classify if the candidate should be cited by the paper. This application can benefit researchers in exploring relevant studies for the paper. It not only accelerates the process of doing research and paper-writing but also prevents missing related works.

Experimental Datasets. Here we adopt the AAN Anthology Network Corpus [37] for the experiments. The corpus consists of the descriptions of 23,766 papers written by 18,862 authors in 373 venues and 124,857 citations in a citation network. In addition, the text of some papers are available in the corpus. For each paper with available text, the paper and each of its citations with text in the corpus is treated as a positive example. For each positive example, a irrelevant paper is randomly sampled to create a negative example for constructing a balanced dataset. To prevent the leakage of ground truth, the *References* sections are manually removed. We also remove the abstract sections to increase the difficulty of the task. The dataset is partitioned by the source papers of examples. The training set includes the examples generated by 80% of the source papers when each of the validation and testing sets is generated by 10% of the source papers, respectively. Table 6 demonstrates the statistics of the experimental datasets for citation recommendation.

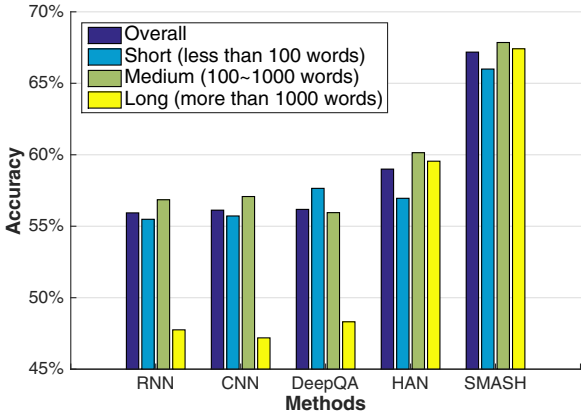


Figure 5: The classification performance of different methods across different document lengths in email attachment suggestion. SMASH uses all-level hierarchies.

Experimental Results. Table 7 shows the performance of the methods for citation recommendation. The experimental results are consistent with the results of the previous two tasks. Three baseline methods with only word-level knowledge have similar performance; CNN without using the asymmetric similarity matrix and considering word dependencies performs slightly worse than the other methods. HAN with the sentence-level information still outperforms all of the other baseline methods. After exploiting the knowledge of multi-level hierarchies, SMASH surpasses all of the baseline methods. Similar to the results for related article recommendation, the word-level hierarchy does not enhance the performance after considering the paragraph- and sentence-level knowledge. It depicts that high-level structural information is much more important for the task of citation recommendation.

4.5 Robustness Analysis and Discussion

After evaluating the effectiveness of SMASH RNN in different real-world applications, we analyze its robustness in this section.

4.5.1 Document length analysis. First, we verify that the improvements achieved by SMASH are consistent across different lengths of documents with the task of email attachment suggestion. Here we simply categorize document lengths into three groups, including short (less than 100 words), medium (100 to 1,000 words), and long (more than 1,000 words). For each testing example, we classify the example into the three length groups based on the maximum of the length of the source and candidate documents. Figure 5 and 6 show the classification and ranking performance of different methods for email attachment suggestion. Although HAN has a similar performance to other baseline methods for short and medium documents, it significantly improves both accuracy and MAP for the long-form documents by taking the sentence-level knowledge into account. It demonstrates the importance of understanding the document structure to deal with long-form documents. The improvements of SMASH against the baseline methods, including HAN, are significant and consistent across different document lengths. This demonstrates that hierarchies utilized in SMASH faithfully capture

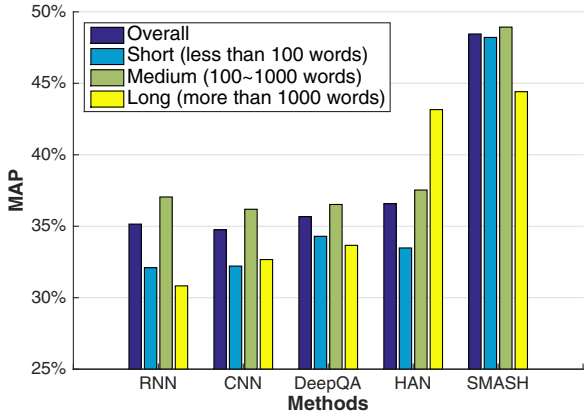


Figure 6: The ranking performance of different methods across different document lengths in email attachment suggestion. SMASH uses all-level hierarchies.

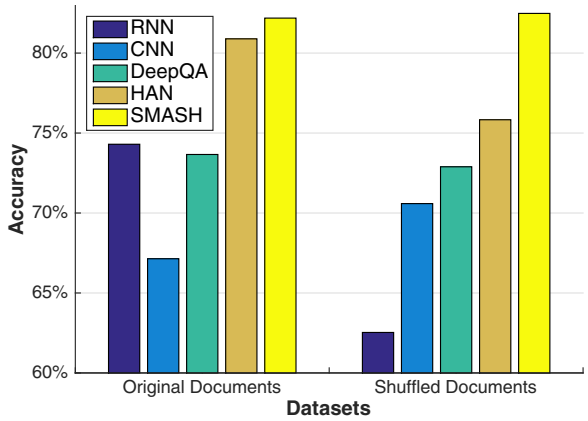


Figure 7: The classification performance of different methods with the original and shuffled documents in related article recommendation. SMASH uses the paragraph- and sentence-level hierarchies.

the document structure and semantics through multiple levels of abstraction. Hence, SMASH is robust across the spectrum of document lengths, and is capable of modeling both short as well as long-form documents.

4.5.2 Robustness to document perturbation. An interesting observation is that although SMASH achieves a 13.86% improvement of accuracy against HAN for email attachment suggestion, the improvements for the other two tasks, while statistically significant, are much smaller (1.6% and 3.1%, respectively). We assume this is because the opening words in Wikipedia pages (descriptions) and academic papers (introductions) are highly informative, and therefore the baselines can solve the problem using only positional information. To verify the assumption, we shuffle the paragraphs of Wikipedia articles to make the important texts randomly distributed in the documents for related article recommendation. Figure 7 shows the performance for related article recommendation with original and shuffled documents. RNN and HAN perform worse with shuffled documents because they cannot identify the critical parts in different positions. CNN and DeepQA have consistent performances because the convolutional features are independent to positional information. These improvements are also consistent in several trails with different random seeds for shuffling documents.

The performance of SMASH is also consistent because the model diagnoses the documents at different levels with attention. Moreover, the improvement against the best baseline method, HAN, increases significantly for the shuffled documents, and is more in line with the improvements achieved for the email attachment suggestion task. This leads us to conclude that SMASH is robust to the positioning of the core document information, and can perform equally well both for documents with well-defined introductions, as well as for complex, non-linear narratives.

5 CONCLUSIONS

In this paper, we propose to address the problem of semantic text matching for long-form documents, which has been less explored in the previous studies. In order to model the complex semantics of long-form documents, MASH RNN is introduced to generate document representations with hierarchical structures at different levels, as well as the attention mechanism in deep learning. Our model, SMASH RNN, is then formulated as a Siamese structure that aggregates the representations of the source and candidate documents derived by two MASH RNNs.

In addition to formulating the theoretical framework, we also demonstrate the practical potential of semantic text matching for long-form documents by providing three real-world applications, including email attachment suggestion, related article recommendation, and citation recommendation. Extensive experiments demonstrate that our proposed approach significantly outperforms several competitive baseline methods within different categories in both of the classification and ranking tasks. Moreover, we also show the robustness of SMASH RNN across the spectrum of documents lengths and perturbations.

Our conclusions can be summarized as follows: (1) semantic text matching for long-form documents is impactful, with numerous useful applications; (2) the usage of hierarchical document structure is essential for semantic text matching, especially for modeling long-form documents; (3) SMASH RNN can accurately capture the complicated semantics of long-form documents, even if the important messages may occur at any position, and at any level of the document structure.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. TensorFlow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI'16)*. USENIX Association, 265–283.

[2] Hadi Amiri, Philip Resnik, Jordan Boyd-Graber, and Hal Daumé III. 2016. Learning text pair similarity with context-sensitive autoencoders. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL'16)*, Vol. 1. ACL, 1882–1892.

[3] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. 1999. *Modern information retrieval*. Vol. 463. ACM.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint*

arXiv:1409.0473 (2014).
[5] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research* 3, Jan (2003), 993–1022.
[6] Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014. Open question answering with weakly supervised embedding models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 165–180.
[7] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems* 30, 1-7 (1998), 107–117.
[8] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a" siamese" time delay neural network. In *Advances in Neural Information Processing Systems (NIPS'94)*. 737–744.
[9] Michael Busch, Krishna Gade, Brian Larson, Patrick Lok, Samuel Luckenbill, and Jimmy Lin. 2012. Earlybird: Real-time search at twitter. In *Proceedings of 2012 IEEE 28th International Conference on Data Engineering (ICDE'12)*. IEEE, 1360–1369.
[10] Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. 2016. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP'16)*. 1650–1659.
[11] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).
[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* (2018).
[13] Cicero dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING'14)*. ACL, 69–78.
[14] Abdessamad Echihabi and Daniel Marcu. 2003. A noisy-channel approach to question answering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistic (ACL'03)*. ACL, 16–23.
[15] George Forman. 2003. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research* 3, Mar (2003), 1289–1305.
[16] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM'16)*. ACM, 55–64.
[17] Jun Han and Claudio Moraga. 1995. The Influence of the Sigmoid Function Parameters on the Speed of Backpropagation Learning. In *Proceedings of the international workshop on artificial neural networks: From natural to artificial neural computation*. Springer-Verlag, 195–201.
[18] Taher H Haveliwala, Aristides Gionis, Dan Klein, and Piotr Indyk. 2002. Evaluating strategies for similarity search on the web. In *Proceedings of the 11th International Conference on World Wide Web (WWW'02)*. ACM, 432–442.
[19] Hua He, Kevin Gimpel, and Jimmy Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. ACL, 1576–1586.
[20] Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313, 5786 (2006), 504–507.
[21] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.
[22] Paul Jaccard. 1912. The distribution of the flora in the alpine zone. 1. *New phytologist* 11, 2 (1912), 37–50.
[23] Jyun-Yu Jiang, Francine Chen, Yan-Ying Chen, and Wei Wang. 2018. Learning to disentangle interleaved conversational threads with a Siamese hierarchical network and similarity ranking. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'18)*. ACL, 1812–1822.
[24] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on Machine Learning (ICML '15)*. 2342–2350.
[25] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. ACL, 1746–1751.
[26] Diederik P Kingma and Jimmy Lei Ba. 2015. Adam: Amethod for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR'15)*.
[27] Bang Liu, Ting Zhang, Fred X Han, Di Niu, Kunfeng Lai, and Yu Xu. 2018. Matching Natural Language Sentences with Hierarchical Sentence Factorization. In *Proceedings of the 2018 World Wide Web Conference (WWW'18)*. ACM, 1237–1246.
[28] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, 2873–2879.
[29] Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. ACL, 1412–1421.
[30] Prem Melville, Wojciech Gryc, and Richard D Lawrence. 2009. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'09)*. ACM, 1275–1284.
[31] Rada Mihalcea, Courtney Corley, Carlo Strapparava, et al. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the Twentieth AAAI Conference on Artificial Intelligence (AAAI'06)*, Vol. 6. AAAI Press, 775–780.
[32] Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, 2786–2792.
[33] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. 2016. Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*. 148–157.
[34] Alexandru Niculescu-Mizil and Rich Caruana. 2005. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML'05)*. 625–632.
[35] Geoffrey Nunberg. 1990. *The linguistics of punctuation*. Number 18. Center for the Study of Language (CSLI).
[36] Douglas Oard, William Webber, David Kirsch, and Sergey Golitsynskiy. 2015. Avocado research email collection. *Philadelphia: Linguistic Data Consortium* (2015).
[37] Dragomir R. Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. 2013. The ACL anthology network corpus. *Language Resources and Evaluation* (2013), 1–26. https://doi.org/10.1007/s10579-012-9211-2
[38] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'15)*. ACM, 373–382.
[39] Aliaksei Severyn and Alessandro Moschitti. 2015. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'15)*. ACM, 959–962.
[40] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems (NIPS'15)*. 2440–2448.
[41] Jaime Teevan, Daniel Ramage, and Merredith Ringel Morris. 2011. # TwitterSearch: a comparison of microblog search and web search. In *Proceedings of the Fourth ACM International Conference on Web search and Data Mining (WSDM'11)*. ACM, 35–44.
[42] George Tsatsaronis, Iraklis Varlamis, and Michalis Vazirgiannis. 2010. Text relatedness based on a word thesaurus. *Journal of Artificial Intelligence Research* 37 (2010), 1–39.
[43] Christophe Van Gysel, Bhaskar Mitra, Matteo Venanzi, Roy Rosemarin, Grzegorz Kukla, Piotr Grudzien, and Nicola Cancedda. 2017. Reply with: Proactive recommendation of email attachments. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM'17)*. ACM, 327–336.
[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS'17)*. 5998–6008.
[45] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations.. In *AAAI*, Vol. 16. AAAI Press, 2835–2841.
[46] Chenglong Wang, Feijun Jiang, and Hongxia Yang. 2017. A hybrid framework for text modeling with convolutional RNN. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'17)*. ACM, 2061–2069.
[47] Shuohang Wang and Jing Jiang. 2017. A compare-aggregate model for matching text sequences. (2017).
[48] Wikipedia. 2001. Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/
[49] Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. 2008. Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems* 26, 3 (2008), 13.
[50] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 55–64.
[51] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning (ICML'15)*. 2048–2057.
[52] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'16)*. ACL, 1480–1489.

[53] Wen-tau Yih, Kristina Toutanova, John C Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL'11)*. ACL, 247–256.

[54] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics* 4 (2016), 259–272.

[55] Xiang Zhang and Yann LeCun. 2015. Text understanding from scratch. *arXiv preprint arXiv:1502.01710* (2015).

RealGraph: A Graph Engine Leveraging the Power-Law Distribution of Real-World Graphs

Yong-Yeon Jo
Hanyang University
Seoul, Korea
jyy0430@hanyang.ac.kr

Myung-Hwan Jang
Hanyang University
Seoul, Korea
sugichiin@hanyang.ac.kr

Sang-Wook Kim*
Hanyang University
Seoul, Korea
wook@hanyang.ac.kr

Sun-Ju Park
Yonsei University
Seoul, Korea
boxenju@yonsei.ac.kr

ABSTRACT

As the size of real-world graphs has drastically increased in recent years, a wide variety of graph engines have been developed to deal with such big graphs efficiently. However, the majority of graph engines have been designed without considering the power-law degree distribution of real-world graphs seriously. Two problems have been observed when existing graph engines process real-world graphs: inefficient scanning of the sparse indicator and the delay in iteration progress due to uneven workload distribution. In this paper, we propose RealGraph, a single-machine based graph engine equipped with the hierarchical indicator and the block-based workload allocation. Experimental results on real-world datasets show that RealGraph significantly outperforms existing graph engines in terms of both speed and scalability.

CCS CONCEPTS

• **Information systems** → *Graph-based database models*; *Database management system engines*; *Computing platforms*.

KEYWORDS

Graph engine, single machine, real-world graph, power-law degree distribution

ACM Reference Format:
Yong-Yeon Jo, Myung-Hwan Jang, Sang-Wook Kim, and Sun-Ju Park. 2019. RealGraph: A Graph Engine Leveraging the Power-Law Distribution of Real-World Graphs. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313434

1 INTRODUCTION

Graphs are widely used data structures for representing relationships between objects in various real-world domains, such as the web, social networks, protein networks, etc. Recently, with their increasing usage, the size of real-world graphs has increased rapidly. Such real-world graphs are called *big graphs*. To meet the demand for efficient handling of real-world big graphs, various graph engines have been developed [1, 6, 18, 25, 30]. These graph engines provide a simple programming interface to help programmers easily develop and execute graph algorithms for big graphs [14, 21].

Since all graph algorithms could be iteratively performed, existing graph engines adopt an *iterative processing* method, where an iteration starts to proceed *after* all the nodes to be processed at its previous iteration are completely processed [8, 19, 24]. Each iteration consists of two phases. At the first phase, the graph engine scans the *indicator* consisting of a bit vector [14, 29] to identify the nodes to be processed. At the second phase, it processes the identified nodes in parallel through multi-threading. Both phases of the iteration in existing graph engines, however, have been developed without careful considerations on the degree distribution of real-world graphs.

Most real-world graphs follow the *power-law degree distribution*, which is characterized by very few nodes having a very high degree (i.e., *hub nodes*), while most nodes (i.e., non-hub nodes) having a low degree [15]. The power-law degree distribution of real-world graphs causes following two phenomena when existing graph engines execute graph algorithms. First, the distribution of the nodes processed at iterations is skewed where the majority of nodes are processed at less than first half iterations while only a small number of nodes are processed at the remaining iterations[1]. Second, at some iterations, the nodes with very large degree disparity (i.e., hub and non-hub nodes) are processed together.

These phenomena cause the following two problems, which slows down the performance of the graph engines. The problem related to the first phase of the iteration is *inefficient scanning of the indicator*. Existing graph engines use a single flat indicator [14, 25, 29, 30]. They scan the whole indicator *linearly* at each iteration to identify the nodes to be processed. In the majority of iterations, the number of nodes to be processed tends to be very small, which makes the indicator very sparse in those iterations. Linear scanning of the very sparse indicator causes unnecessary computations and leads to the performance degradation of the graph engines.

The problem related to the second phase of the iteration is the delay for the progress of the next iteration due to *uneven workload*

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313434

[1]Note that this phenomenon does not happen in the graph algorithms that processes *all nodes* at each iteration, such as PageRank.

allocation over threads. Existing graph engines adopt the node-based workload allocation, where each thread processes an individual node with its edges [6, 14, 18, 30]. Since the workload of each thread is proportional to the number of edges, the node-based allocation produces huge difference in the workload between the threads that process hub nodes and the others that process non-hub nodes. This causes the threads that process non-hub nodes to wait for the ones that process hub nodes to complete, which delays the start of the next iteration.

To solve these two problems, we propose a new single-machine based graph engine[2] called *RealGraph*. RealGraph is designed based on the principles of well-known database systems [3, 4, 7] to manage big data efficiently. Furthermore, When storing graph data, RealGraph preserves *data locality* by adopting *efficient data layout* [12] that stores the data accessed together in adjacent storage space. In addition, we propose two new techniques considering the characteristics of real-world graphs to execute the graph algorithms efficiently.

First, we propose the *hierarchical indicator* that consists of multi-level bit vectors designed to skip easily the ranges of the indicator to be unnecessarily scanned. The lowest-level bit vector is the same as the flat indicator in existing graph engines. A higher-level bit vector compresses some range of its lower-level bit vector using a single bit. A bit set as 1 of each level bit vector denotes that a corresponding range at its lower-level bit vector should be scanned. The process starts from the highest-level bit vector; it scans the range of the lower-level bit vector corresponding to the bit (if each higher-level bit vector has a bit set as 1). Unlike the existing flat indicator, the hierarchical indicator reduces unnecessary scanning of the sparse indicator dramatically.

Second, we propose the block-based workload allocation which assigns *a block* to each thread, instead of a node and its edges in the node-based one. A block is a fixed-size standard I/O unit of the graph engine, which stores multiple nodes and their edges together. Since the block size is fixed, the total number of edges stored in each block is almost the same. Thus, the graph engine can allocate almost the same workload to each thread. This reduces the waiting time among threads at every iteration, which eventually accelerates the start of the next iteration.

We show the effectiveness of the proposed techniques through extensive experiments. The results indicate that RealGraph improves the performance considerably when using the proposed techniques together rather than individually. We also demonstrate the performance of RealGraph by comparing it with five well-known single-machine based graph engines [14, 21, 25, 29, 30] and four distributed-system based graph engines [1, 9, 16, 26]. For experiments, we used seven graph algorithms and six real-world graphs. For all graph algorithms and datasets, RealGraph shows the best performance in terms of processing speed and scalability compared with existing graph engines.

The contributions of this paper are as follows.

- First, we point out the problems that occur when existing graph engines deal with real-world graphs.
- Second, we propose two techniques for efficient processing of real-world graphs: (1) the hierarchical indicator to reduce unnecessary scanning and (2) the block-based workload allocation to balance the workload among threads.
- Third, we propose RealGraph, a single-machine based graph engine employing with these techniques.
- Finally, we verify the superiority of RealGraph through extensive experiments.

The organization of the paper is as follows. Section 2 addresses our motivation. Section 3 overviews our RealGraph. Section 4 explains the hierarchical indicator. Section 5 describes the block-based workload allocation. Section 6 shows experimental results. Finally, Section 7 summarizes and concludes the paper.

2 MOTIVATION

This section describes the execution procedure of a graph algorithm in existing graph engines, explains the characteristics of real-world graphs, and points out the problems when existing graph engines process real-world graphs.

The graph engine is a general-purpose software framework that aims to process various graph algorithms, rather than targeting specific graph algorithms [6, 9, 13, 14, 16–19, 21, 25, 26, 29, 30]. Although some graph algorithms could be processed asynchronously without synchronization between threads, all graph algorithms can be executed synchronously where threads are synchronized at regular intervals [8]. Therefore, most graph engines adopt *iterative processing* based on synchronous execution[3], where all the nodes to be processed at each iteration are processed completely and then the next iteration starts [24].

At each iteration, the graph engine executes a graph algorithm with two phases [13, 14, 25]: (1) identifying the nodes to be processed and (2) processing the identified nodes and their edges. In the first phase, the graph engine identifies the nodes to be processed by scanning the *indicator* consisting of a bit vector[4] of the length equal to the number of nodes [14, 25, 30]. If the bit is set as 1, it means that the node corresponding to the bit is to be processed at the current iteration. In the second phase, threads process the identified nodes and their edges in parallel. The existing graph engines adopt the node-based workload allocation that assigns a node and its edges to each thread [6, 18, 30].

However, both phases were designed without careful consideration of the degree distribution of real-world big graphs. Note that most real-world graphs show the power-law degree distribution, consisting of a small number of hub nodes and a very large number of non-hub nodes [15].

[2]Existing graph engines can be classified into two categories: single-machine based ones and distributed-system based ones.

[3]Some graph engines adopt an asynchronous execution based processing [10, 24].

[4]For the indicator composed of a bit vector, the space required for representing a node is a single bit. Therefore, even when dealing with a big graph with a large number of nodes, the graph engine requires only a small, fixed space. The downside of using a bit vector is that many bits are left unused when only a small number of nodes are processed in an iteration. A *linked list* may be considered as an alternative data structure. The linked list, however, requires much more space to represent a single node (e.g., 4 bytes in the case of using an integer for a node). In the iteration that processes only a small number of nodes, there is an advantage of using only a very small space for the nodes. But there would be a significant overhead of allocating and releasing a number of nodes for a linked list at each iteration. In particular, if the number of nodes that need to be processed in an iteration exceeds the given memory space, the graph engine could not run. This is the reason why most graph engines employ the simple and small bit vector for the indicator.

For example, Figure 1 shows the degree distribution of the Yahoo dataset used in our experiments, where the x-axis represents the degree of a node, and the y-axis does the number of nodes having the degree. We observe this dataset also follows the power-law degree distribution. This power-law degree distribution of the real-world graph causes the following phenomenon at each phase, which degrades the performance of graph engines.

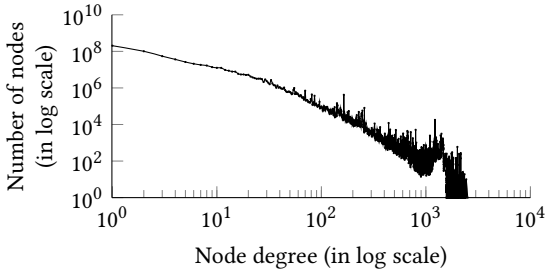


Figure 1: Degree distribution of the Yahoo dataset.

First, the distribution of the nodes processed at iterations is skewed. Since hub nodes are connected to many other nodes, they are highly likely to be accessed at early iterations in general regardless of the starting node. Also, a large number of nodes connected to the hub nodes are accessed together in the following iteration. As a result, the majority of nodes tend to be processed at early iterations, and only a small number of nodes that are multiple hop away from the hub nodes are likely to be handled at remaining iterations.

Figure 2 shows the cumulative number of the nodes processed up to each iteration, when the breadth first search (BFS) is performed on all the datasets used in experiments. Since the total number of iterations is different for each dataset, the x-axis represents the ratio of the number of processed iterations to that of the entire iterations. The y-axis indicates the cumulative percentage of the processed nodes. At early 30% or less of iterations, almost 90% of nodes are processed, and fewer than 10% nodes are processed over the remaining 70% of iterations. It is also observed that only a few nodes are processed across at more than half of the remaining iterations.

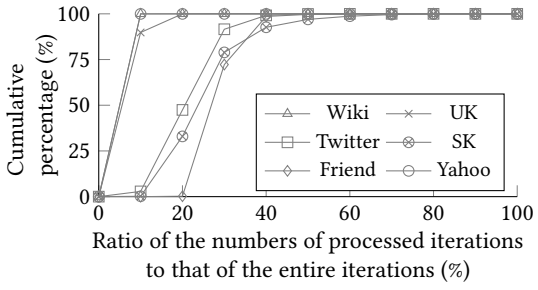


Figure 2: Cumulative number of the nodes processed up to each iteration when BFS is performed.

This phenomenon makes the indicator *very sparse in most iterations*. Since the existing graph engines linearly scan the entire indicator, the scanning becomes very inefficient. As the graph size increases, this linear scanning becomes even more impractical. In fact, with 1.4 billion nodes in the Yahoo dataset, the largest dataset used in our experiments, the scanning overhead for the indicator takes up to 50% of the total BFS execution time.

Secondly, the degrees of the nodes processed at each iteration vary significantly. In particular, the degree difference between nodes is very large at the iterations where hub nodes and non-hub nodes are processed together. Figure 3 shows the degree difference between the nodes processed at each iteration, when BFS is performed on all the datasets. The results for the first 100 iterations are shown[5]. The x-axis represents the iteration number, and the y-axis shows the ratio of the minimum node degree to the maximum node degree in log scale. We observe the existence of significant degree difference at every iteration, and in extreme cases the difference is up to 10^6 times more.

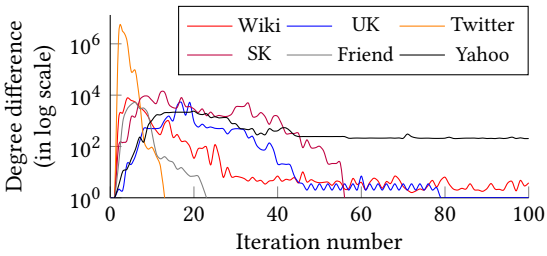


Figure 3: Degree difference between the nodes processed at each iteration when BFS is performed.

The degree difference between nodes causes uneven distribution of workload among threads in existing node-based workload allocation. Because the workload of threads is proportional to the degree of the assigned node in the node-based allocation [14, 25, 29] the workload among threads varies, which in turn delays the completion of the iteration. This is particularly true at the iteration which processes the hub nodes, because the threads with non-hub nodes should wait until the ones with the hub nodes to complete their executions. This delays the start of the next iteration, which eventually slows down the graph engine.

These observations motivate us to develop our RealGraph, a graph engine equipped with our two novel techniques: (1) efficient scanning of the sparse indicator and (2) uniform workload allocation over threads to minimize the delay for the progress of the next iteration.

3 OVERVIEW OF REALGRAPH

In this section, we describe the general architecture of our RealGraph and the efficient data layout for graph storage in RealGraph. In the following Sections 4 and 5, we describe how RealGraph solves the problems mentioned in Section 2.

3.1 Architecture

RealGraph employs, as a programming model, the *vertex-centric programming model* [14, 30] that performs a graph algorithm per the node with its edges. As a memory model, RealGraph employs the *external-memory model* [21, 29] that utilizes external storage when graph data exceeds the main memory.

[5]We show the results for the first 100 iterations, since the numbers of total iterations are different across datasets.

As shown in Figure 4, RealGraph consists of four layers: storage management layer, buffer management layer, object management layer, and thread management layer. The lower three layers manage memory and storage space. They are designed on the basis of well-known database storage systems, such as WiSS [7], Exodus [4], and Shore [3]. The thread management layer executes graph algorithms and is implemented with two techniques proposed in this paper. On top of them, we also provide a web-based user interface[6]. Using the user interface, anyone could execute a graph algorithm provided by RealGraph on their own graphs and download its result. The roles and components of each layer except for the interface are as follows.

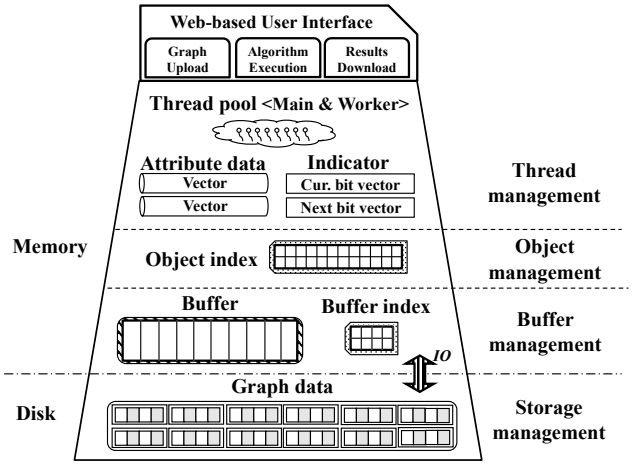


Figure 4: Architecture of RealGraph.

The storage management layer manages storage space. The graph data is divided into blocks and stored in storage space. A block is the standard I/O unit, and each block contains one or more objects. Here, an object is composed of a node and its edges[7]. If the size of an object exceeds the block size, it is stored across several blocks. This layer also manages I/Os of these blocks.

The buffer management layer manages the space for blocks in the main memory. It consists of a buffer and a buffer index. A buffer is a space with a given size in main memory and keeps blocks within that space. The buffer index has the indices for blocks loaded in buffer, which helps to access the blocks in the buffer quickly. This index is always loaded in the main memory.

The object management layer manages the location information for blocks and objects of the graph data. There is an object index which contains the indices of objects and blocks stored in storage device. It is always loaded in the main memory.

The thread management layer performs a graph algorithm. It consists of three components: thread pool, attribute data, and two indicators (i.e., current and next). The thread pool has two types of threads: worker threads that actually perform graph algorithms and one main thread that manages worker threads. The attribute data stores intermediate and final results for either nodes or edges obtained when performing a graph algorithm (e.g., nodes visited in BFS, node ranks in PageRank). The attribute data consists of several attribute vectors, each of which has the same number of elements as that of nodes. The element of each vector is the attribute corresponding to the node. If the size of the attribute data exceeds that of a given memory, they are divided into several chunks and stored in the secondary storage; each chunk required by a graph algorithm is loaded into memory. The current/next indicator stores the information about the nodes to be processed at the current/next iteration. Both indicators are always loaded in the main memory.

On the basis of the above architecture, each iteration of a graph algorithm is executed as follows. First, the main thread identifies the nodes to be processed at the current iteration by scanning the *current* indicator. Next, using the object index, the main thread finds the locations of the blocks including the objects corresponding to the identified nodes, and checks if the block exists in the buffer. Then, the main thread assigns each block to be processed to each worker thread in order. If it does not, the worker thread requests the block to the storage management layer and loads it into the buffer. After loading, the worker thread executes the function of a graph algorithm and updates the results in the corresponding attribute data. Then, it sets the next indicator such that the nodes connected to the processed nodes in the current iteration are to be processed at the next iteration. Finally, before the start of the next iteration, the next indicator is switched to the current indicator, and the current indicator is initialized to be used as the next indicator.

3.2 Efficient data layout

A good placement of nodes and edges in storage (i.e., data layout) would improve both CPU and I/O performances of a graph engine. Existing graph engines, however, have aimed at maximizing the utilization of computing resources through system optimization, overlooking the importance of data layout [14, 25].

Recognizing this, we have borrowed the concept of efficient data layout [12] and applied it to RealGraph. The main idea for this data layout is, the data accessed together in executing graph algorithms are placed in the same or adjacent storage space. Since RealGraph stores objects (each having a node and its edges) in blocks in the ascending order of node IDs as other graph engines [21, 29], we assign consecutive node IDs to the nodes likely to be accessed together at the same or successive iteration(s) by a graph algorithm. This idea makes the objects accessed together in a graph algorithm to be stored in the same or adjacent block(s).

Based on reference [12] to assign consecutive node IDs to nodes to be accessed together in various graph algorithms, breadth first search (BFS) is used as follows: BFS is performed from an arbitrary starting node in a graph given; the nodes accessed by BFS are recorded one by one in the order; the node ID used in a graph engine is determined by this order. The data layout obtained by this idea makes the nodes and edges accessed together at the same or successive iteration(s) of graph algorithms placed in the same or adjacent block(s) successfully.

This data layout transformation is performed at the preprocessing step where RealGraph stores a graph[8]. The data layout transformation is required only once for each graph to be stored in

[6] http://166.104.110.75:20080

[7] Since RealGraph uses the adjacency list, each object is stored as a node and the edges pointing to its adjacent nodes.

[8] Most graph engines also perform the preprocessing step when storing a graph. For example, GraphChi divides a graph into shards and then stores them in its preprocessing step.

RealGraph, and the graph stored in this new data layout is beneficial to the performance of many graph algorithms running on RealGraph. This implies that the transformation overhead is shared by many graph algorithms running on RealGraph.

Reference [12] reports that the data layout determined by BFS enables a graph engine not only to reduce the number of block accesses but also to face more sequential accesses of blocks, rather than random accesses, thereby improving its performance considerably.

Figure 5 shows two data layouts. A data layout has nine blocks represented by rectangles. A circle represents a node and its edges in a block. The numbers in circles indicate the order of access requests by a given graph algorithm. For processing, a graph engine should load 8 nodes into the main memory in Figure 5. In this case, by changing into the efficient data layout from existing data layout, we improve the I/O and CPU performances of RealGraph.

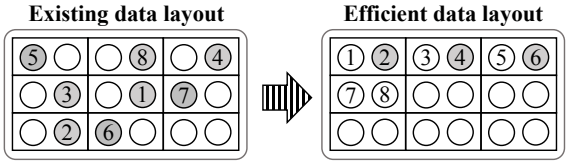


Figure 5: Data layout transformation.

When performing graph algorithms, worker threads need frequent access to blocks and attribute data. Even when it is not possible to load all of the blocks into memory, since the objects to be processed together are more likely to be located in the same or successive blocks, the number of I/O requests would be decreased and the sequential access rate could be increased. In Figure 5, RealGraph accesses only four blocks sequentially, instead of 8 blocks randomly. Similarly, the I/O performance would be increased when entire attribute data could not be loaded into memory, since the same or successive chunks of attribute data are more likely to be updated. In both cases, it is more likely to re-access the data already loaded in the main memory, which results in the increased cache hit ratio and improves the CPU performance. In addition, since adjacent bits are more likely to be set at each iteration in the current indicator, the cache hit ratio for the indicator would be increased.

Through experiments, we evaluate the performance of the efficient data layout when applied to RealGraph and also examine the synergy effects when the efficient data layout is used in combination with our two techniques in the following sections.

4 HIERARCHICAL INDICATOR

As explained in Section 2, linear scanning of the sparse indicator is very inefficient. For efficient scanning, we propose a hierarchical indicator that skips efficiently the ranges that do not need to be scanned.

The hierarchical indicator is made up of two or more bit vectors, where the upper-level bit vector determines the ranges to be scanned in the lower-level bit vector. Each level bit vector is divided into a set of ranges with a fixed length. The lowest-level bit vector is the same as the flat indicator used in existing graph engines. Each higher-level bit vector consists of a number of bits as many as the number of ranges in each lower-level bit vector. Each bit of the higher-level bit vector refers to the corresponding range of its lower-level bit vector. Both the current and next indicators are configured in the same way.

Figure 6 shows the hierarchical indicator consisting of three level bit vectors with a range length of 3. The top-level bit vector consists of three bits. The middle-level bit vector has three ranges with 9 bits. The bottom-level bit vector has a total of 9 ranges with 27 bits.

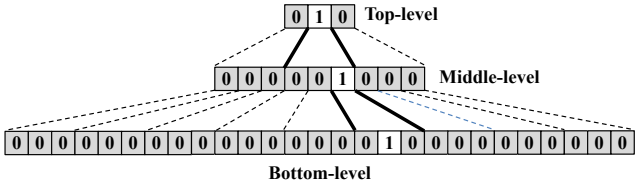


Figure 6: 3-level hierarchical indicator.

The current and next hierarchical indicators are used as follows. At each iteration, the main thread identifies the nodes to be processed in the *top-down manner* from the current hierarchical indicator. It scans each higher-level bit vector to identify the bits set as 1, and determines the ranges to be scanned of the lower-level bit vector. Then, it scans those ranges only. The main thread repeats this process and finds the nodes to be processed corresponding to the bits set as 1 in the lowest-level bit vector. By using the current hierarchical indicator, the main thread in RealGraph could skip all the ranges of the lower-level bit vector corresponding to the bits with 0 of the higher-level bit vector.

In Figure 6, the main thread scans the top-level bit vector and identifies the second bit set as 1. Only the second range of the middle-level bit vector is scanned, and the sixth bit with 1 is found. The sixth range of the bottom-level bit vector is scanned, and node 17 is identified as the node to be processed at the current iteration. In this example, the main thread scans only 9 bits, instead of all 27 bits of the bottom-level bit vector.

For the next hierarchical indicator, the worker threads set the nodes to be processed at the next iteration in the *bottom-up manner*. Each worker thread sets the bits corresponding to the nodes to be processed in the lowest-level bit vector as 1. Then, it sets the bit of the higher-level bit vector corresponding to the range including the bit set as 1 of the lower-level bit vector. This process is repeated until the top-level bit vector is reached.

Suppose Figure 6 is the next hierarchical indicator where the worker thread sets the 17th bit in the bottom-level bit vector. It also sets the sixth bit in the middle-level bit vector corresponding to the sixth range including the 17th bit in the bottom-level bit vector. Finally, it sets the second bit of the top-level bit vector corresponding to the second range containing the sixth bit in the middle-level bit vector.

To construct the hierarchical indicator, two parameters are required: the range length and the height of the hierarchical indicator. Since the height is automatically determined once the range length is fixed, the hierarchical indicator could be configured by determining the range length.

Figure 7 shows the result of BFS by changing the range length on the Yahoo dataset. Figure 7-(a) shows the scanning time, and Figure 7-(b) shows the memory usage and the height of the hierarchical indicator in levels. The experimental results show that as

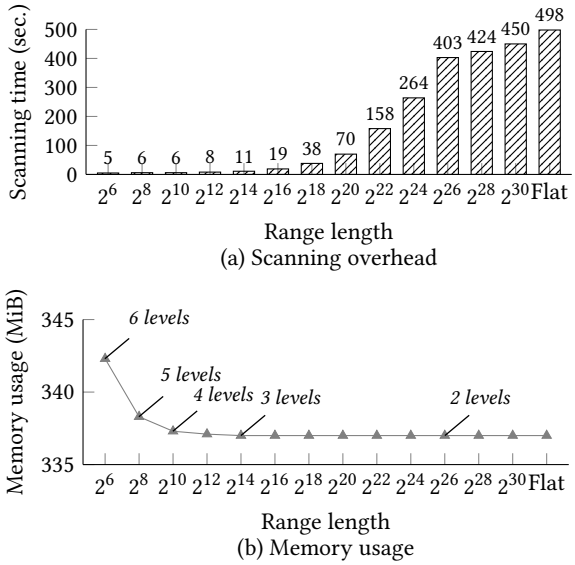


Figure 7: Scanning overhead and memory usage of the hierarchical indicator.

the range length increases, the scanning time increases, and the memory usage decreases. When considering both scanning time and memory usage, the range length of 2^{10} bits is the most efficient. The hierarchical indicator with the range length of 2^{10} bits uses only 0.1% more memory than the flat indicator employed in existing graph engines and reduces the scanning time dramatically to one-hundredth. In subsequent experiments, therefore, we use 2^{10} bits for the range length of the hierarchical indicator.

5 BLOCK-BASED WORKLOAD ALLOCATION

As mentioned in Section 2, the degree difference at each iteration causes uneven workload distribution among threads in the node-based workload allocation. This delays the progress of an iteration, which leads to degradation of the overall performance of the graph engine.

To solve the delay for the progress of the next iteration, we propose the block-based workload distribution which assigns a *block* to each thread, rather than an object (i.e., a node with its edges). A block may contain multiple objects of different sizes, but the sum of their sizes is almost the same. In RealGraph, threads are given with almost equal workloads due to the block-based workload distribution, which reduces the disparity of the completion times between threads.

Since the block size affects the performance of RealGraph, we examine the appropriate block size for RealGraph.

Figure 8 shows the execution time of PageRank[9] with 20 iterations, starting at 64KiB and increasing the block size by 4 times on the Yahoo dataset. The x-axis represents the block size, and the y-axis represents the execution time. The result confirms that the 1024KiB block shows the best performance. Thus, we use 1024KiB as the block size for RealGraph in the following experiments.

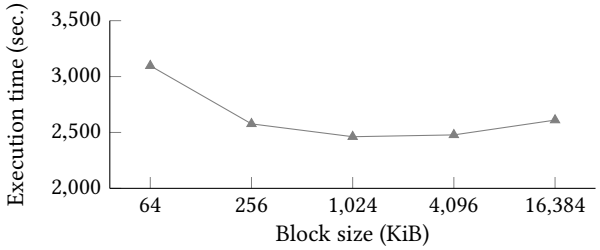


Figure 8: Execution time of PageRank with different block sizes.

To verify the superiority of the block-based workload allocation compared to the node-based one, we apply both node and block allocation techniques to RealGraph and measure the workload of each thread (i.e., the number of processed edges) at each iteration while performing PageRank on the Yahoo dataset. Figure 9 shows the result where the x-axis represents the iteration number, and the y-axis represents the standard deviation of the workload among threads. There exist a few cases where the standard deviation of the workload in the block-based allocation is larger than that in the node-based allocation. In overall, however, the block-based allocation has a much smaller standard deviation than the node-based one. This indicates the block-based allocation distributes the workload more evenly across threads.

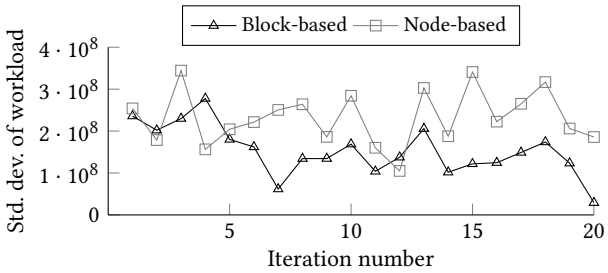


Figure 9: Standard deviation of workload over threads at each iteration

Note that the block-based workload allocation requires additional *thread synchronization* when handling a large object. A single object stored across *multiple blocks*, such as the object for a hub node, is handled by *multiple threads* in RealGraph. If multiple threads that handle a single object need to update the result of the same attribute, thread synchronization is required, making the threads progress in sequence. Since thread synchronization is an operation with large overhead, frequent thread synchronization could degrade the performance of the graph engine. If many objects are stored over several blocks, the performance of RealGraph could slow down.

Fortunately, in real-world graphs, these objects occupy only a small fraction. For all the datasets used in the experiment, when the block size is 1024KiB, only 0.0003% of the objects are stored in several blocks on average. Because the frequency of thread synchronization is very rare during the execution of the graph algorithm,

[9]BFS accesses *part of* nodes at each iteration, which results in random accesses to blocks. The random access of blocks influences some other factors such as buffer replacement policy to affect the performance of the graph algorithm. PageRank, on the other hand, causes sequential accesses to blocks, because it sequentially accesses all the nodes at each iteration. With the sequential access of blocks, other factors are less likely to affect the performance of the graph algorithms. Therefore, in this experiment, we use PageRank to observe the change of performance according to the block size as much as possible.

the performance degradation of RealGraph due to the thread synchronization is negligible.

6 PERFORMANCE EVALUATION

This section demonstrates the superiority of RealGraph through experiments. Section 6.2 validates the techniques applied to RealGraph. Section 6.3 compares the performance of RealGraph with existing single-machine based graph engines. Section 6.4 compares the performance of RealGraph with those of existing distributed-system based graph engines.

6.1 Experimental Setup

We employed five state-of-the-art single-machine based graph engines: GraphChi [14], TurboGraph [25], GridGraph [30], FlashGraph [29], and X-Stream [21]. These engines and RealGraph were performed on PCs with Intel i7-7700K, 1TiB SSD, and 64GiB memory. We also compared RealGraph with four distributed-system based graph engines: PowerGraph [9], GraphLab [16], GraphX [26], and Giraph [1].

As experimental datasets, we used six real-world graphs of varying sizes. Table 1 shows the specifications of each dataset, where *#Nodes* is the number of nodes, *#Edges* is the number of edges, and *Size* is the data size. The descriptions of datasets are as follows[10]. *Wiki* is the hyperlinked data of wiki pages in English Wikipedia. *UK* is the dataset representing the relationships of web pages having the .uk domain. *Twitter* is the dataset with the relationships of tweets collected on Twitter. *SK* is the dataset with the relationships of web pages having the .sk domain. *Friend* is the dataset with all links among users in the online gaming social network, Friendster. *Yahoo* is the hyperlinked data of web pages in Yahoo! AltaVista.

Table 1: Graph datasets

Datasets	Real-world graphs					
	Wiki	UK	Twitter	SK	Friend	Yahoo
#Nodes (M)	12	39	61	50	68	1,413
#Edges (B)	0.37	0.93	1.4	1.9	2.5	6.6
Size (GiB)	5.7	16	24	32	44	114

We employed seven graph algorithms widely used in various fields: breadth first search (BFS) [22], weakly connected component (WCC) [23], betweenness centrality (BC) [2], PageRank [20], random walk with restart (RWR) [28], belief propagation (BP) [11], and sparse matrix and vector multiplication (SpMV) [27]. Since BFS, WCC, and BC are the algorithm that accesses *part of* nodes at each iteration, the indicator is used. For the remaining four graph algorithms (i.e., PageRank, RWR, BP, and SpMV), the indicator is not used, because they access all the nodes at each iteration [11]. Since BFS, WCC, and BC differ in the number of iterations according to the datasets, they are performed until their termination conditions are satisfied. For the other four graph algorithms (i.e., PageRank, RWR, BP, and SpMV), the number of iterations is fixed as 20.

[10] Wiki: http://konect.uni-koblenz.de/networks/wikipedia_link_en, UK: http://law.di.unimi.it/webdata/uk-2005, Twitter: an.kaist.ac.kr/traces/WWW2010, SK: http://law.di.unimi.it/webdata/sk-2005, Friend: https://archive.org/details/friendster-dataset-201107, Yahoo: http://webscope.sandbox.yahoo.com

[11] There are many other graph algorithms, such as single source shortest path, depth first search, community detection, and triangle counting that use indicators for their executions. However, since existing graph engines do not provide them, we were not able to include them in our experiments.

6.2 Evaluation of proposed techniques

In this section, we verify the effectiveness of the techniques applied to RealGraph: Hierarchical indicator (H), Block-based workload allocation (B), and efficient Data layout (D). We performed two algorithms — BFS as a representative algorithm for accessing *part of* nodes at each iteration and PageRank as a representative algorithm for accessing *all* nodes at each iteration — on the Yahoo dataset, the largest one in experimental datasets. We set the number of threads of RealGraph as 8. The memory size is limited to 16GiB for the executions of out-of-core algorithms.

Figure 10 shows the results. The *x*-axis represents RealGraph with each technique applied. In the *x*-axis, Null means RealGraph with none of the proposed techniques (i.e., employing the flat indicator and node-based workload allocation), and each alphabet represents RealGraph equipped with the technique referring to its initial. For example, HB indicates RealGraph with the hierarchical indicator (H) and the block-based workload allocation (B). Ω represents RealGraph with all techniques applicable to each algorithm. The *y*-axis represents the execution time. And now, we describe the result in a way that compares the performance of RealGraph including any proposed technique with that of Null.

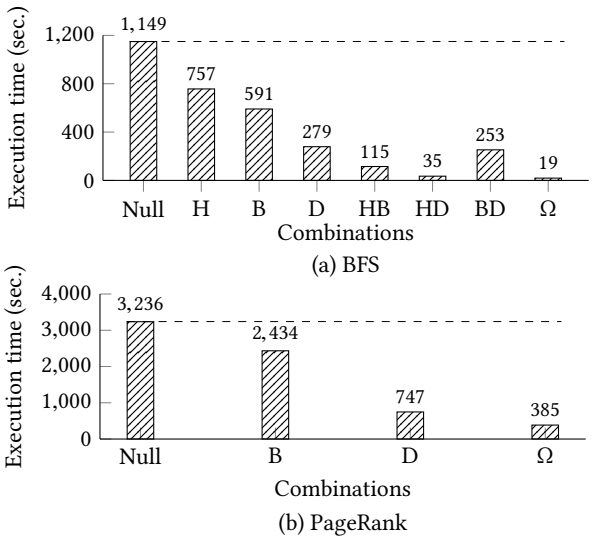


Figure 10: Performance of proposed techniques.

Figure 10-(a) shows the result of BFS. The hierarchical indicator (H) improves the performance about 1.5 times by reducing the scanning overhead, the block-based workload allocation (B) about 1.9 times by the nice load balancing, and the efficient data layout (D) about 4 times by increasing the degree of sequential access and the cache hit for attribute data and blocks, respectively.

When using both the hierarchical indicator and the block-based workload allocation (HB), the performance of RealGraph is improved 9.9 times. This is because the scanning overhead of the indicator at each iteration is reduced and the delay of the progress for the next iteration is prevented by even workload distribution. In addition, the main thread communicates with worker threads

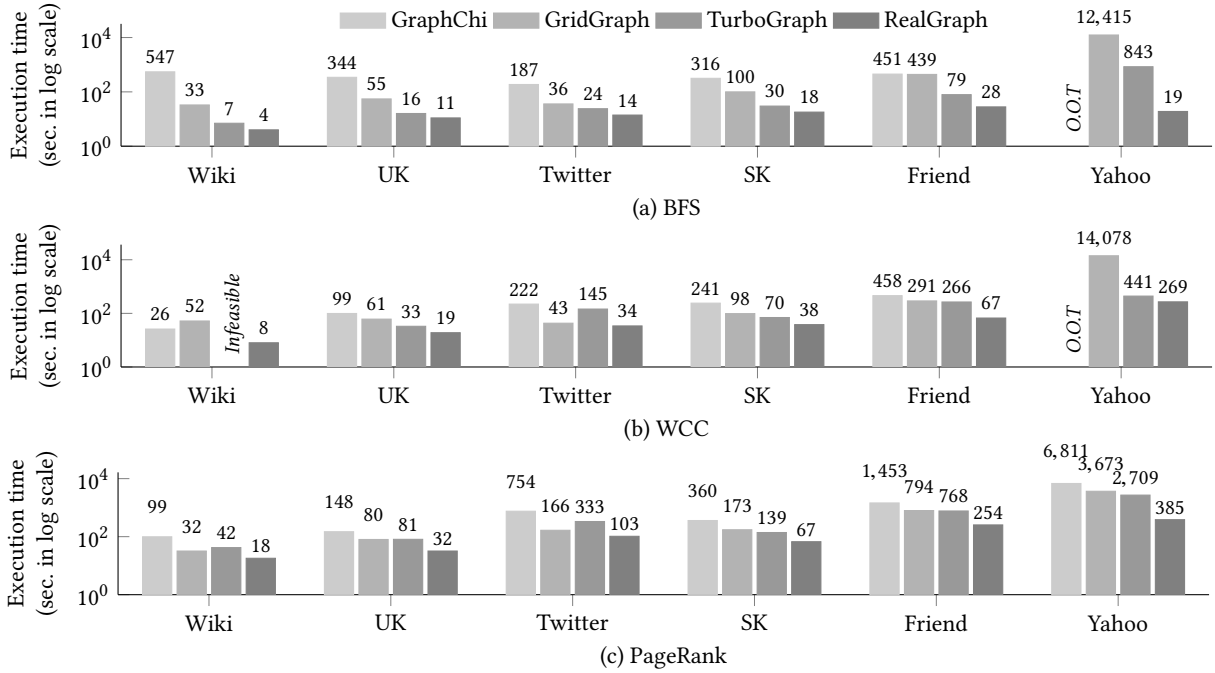


Figure 11: Performance comparison with graph engines with the same model.

per each block (rather than each node), which reduces the communication overhead. This further improves the performance of RealGraph.

The synergy is greater when combining the hierarchical indicator and the efficient data layout (HD). As mentioned in Section 3.2, the performance of RealGraph is improved by the efficient data layout when handling attribute data and graph data. Moreover, the number of ranges to be scanned in the current indicator decreases when efficient data layout is employed, because the bits corresponding to nodes to be processed tend to be clustered in a few ranges with efficient data layout. The cache hit ratio in the next indicator is also increased, because the bits in the same or adjacent ranges are likely to be set. This results in the *dramatic* performance improvement of RealGraph by 32.8 times.

The combination of the block-based workload allocation and the efficient data layout (BD) also shows synergy, improving the performance of RealGraph by 4.4 times. The magnitude, however, is somewhat smaller than the previous two cases (HB and HD). The reason is as follows. The efficient data layout aggregates the nodes to be processed into one or adjacent blocks. If an iteration processes a small number of nodes, the number of blocks to be processed may be less than the number of threads in RealGraph. Since each thread processes a block by the block-based workload allocation, all the given threads could have not been fully utilized in this case.

Finally, the combination of all three technologies (Ω) enhances the performance of RealGraph by 60 times.

Figure 10-(b) shows the results of PageRank. In PageRank where the indicator is not used, we could apply the block-based workload allocation (B) and the efficient data layout (D) to RealGraph. Each technique improves the performance by 1.3 and 4.3 times, respectively. The performance of RealGraph is improved 8.4 times when using both techniques together (Ω). The efficient data layout increases the probability that the nodes with sequential node IDs and the nodes connected to them take adjacent node IDs. If these nodes are processed by a single thread, the cache hit ratio for attribute data can be further increased. Because the block-based workload allocation assigns each thread the nodes in a single block with sequential node IDs, the performance is further improved by the synergy between the block-based allocation and the efficient data layout.

Through above experiments, we showed that each technique of RealGraph is effective, and that their combinations create a significant synergy effect. We also demonstrated that these techniques are all effective regardless of the type of graph algorithms.

6.3 Comparisons with single-machine based graph engines

In this experiment, we evaluate the performance of RealGraph by comparing it with existing single-machine based graph engines. Section 6.3.1 evaluates the performance of the single-machine based graph engines following the *same model* as RealGraph. Section 6.3.2 evaluates the performance of the single-machine based graph engines adopting *different models* from RealGraph. In these two sections, we used BFS, WCC, and PageRank, which are commonly provided by all graph engines. Section 6.3.3 shows the performance of other graph algorithms provided by different graph engines such as BC, RWR, BP, and SpMV.

6.3.1 Single-machine based graph engines with the same model. We employed TurboGraph [25], GridGraph [30], and GraphChi [14], all of which follow the same model as RealGraph. They adopt the vertex-centric model and the external-memory model. Since

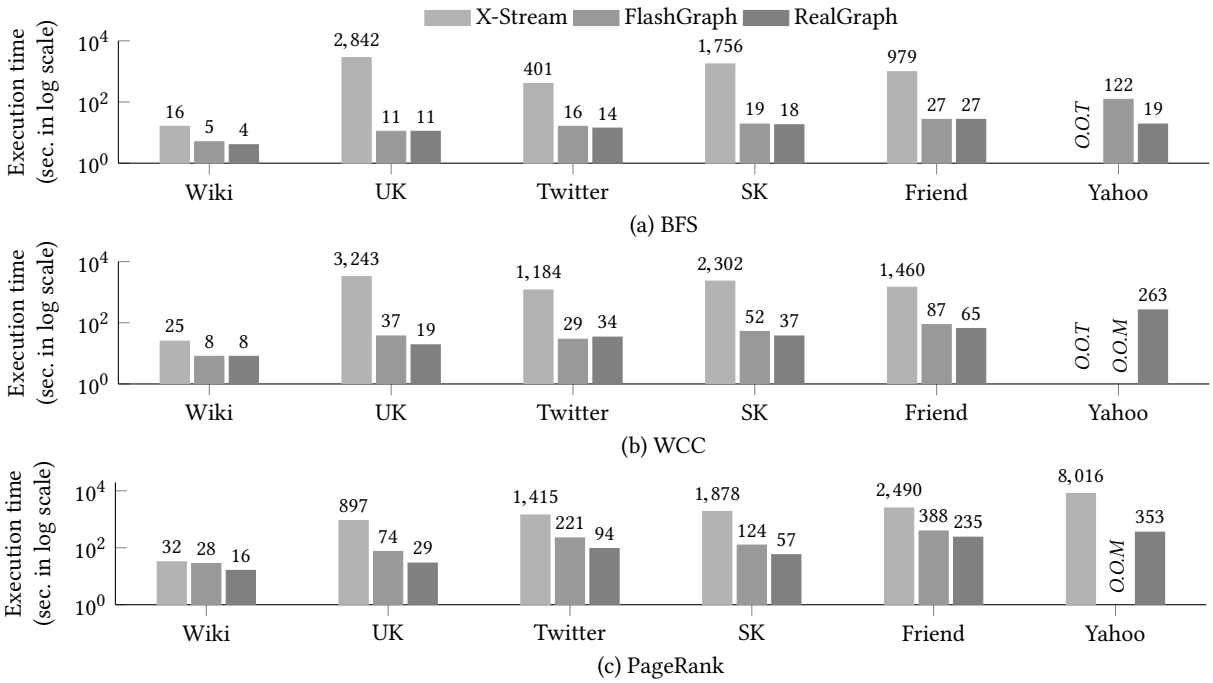


Figure 12: Performance comparison with graph engines with the different models.

TurboGraph fixed the number of threads identically as six, we set the number of threads of all the graph engines to six.

Figure 11 shows the results. The x-axis represents the graph datasets, and the y-axis represents the execution time in log scale. The experimental results show that overall execution time is in order of GraphChi, GridGraph, TurboGraph, and RealGraph. RealGraph provides the best performance *universally* for all graph algorithms and for all datasets. In particular, RealGraph is better than TurboGraph, the best performer among existing ones, up to about 44 times in the best case (i.e., BFS on the Yahoo dataset). We note that TurboGraph does not work in the Wiki dataset (i.e., infeasible), and GraphChi could not get the results of BFS and WCC on the Yahoo dataset within 24 hours (i.e., out of time, O.O.T).

6.3.2 Single-machine based graph engines with different models. In this experiment, we employed two graph engines with different models from that of RealGraph: FlashGraph [29] and X-Stream [21]. FlashGraph adopts the vertex-centric model and the semi-external memory model. X-Stream adopts the edge-centric model and the external-memory model. The number of threads was set identically as 8. We performed BFS, WCC, and PageRank, as in the previous experiments.

Figure 12 shows the experimental results. The x-axis represents the graph datasets, and the y-axis represents the execution time in log scale. RealGraph has the same or better performance than FlashGraph except for WCC on the Twitter dataset. In particular, the performance of RealGraph is up to six times better than that of FlashGraph (i.e., BFS on the Yahoo dataset). Since the graph engine with the semi-external memory model does not work if whole attribute data cannot be loaded into the given memory, FlashGraph could not perform PageRank and WCC on the Yahoo dataset (i.e., out of memory, O.O.M), while RealGraph performs all the algorithms.

Compared with X-Stream, RealGraph has always better performance for all graph algorithms and all datasets. In the case of PageRank on the Yahoo dataset, RealGraph outperforms X-Stream up to 250 times. Since the graph engines with the edge-centric model sequentially read the entire graph data at each iteration, it is inefficient for the algorithms that need to process only a few nodes at each iteration such as BFS and WCC. X-Stream does not work on BFS and WCC on the Yahoo dataset within 24 hours (i.e., O.O.T), while RealGraph works in maximum 360 seconds. These results show that RealGraph has superior scalability and better performance compared to two graph engines with different models.

6.3.3 Performance on other graph algorithms. This experiment shows that RealGraph is effective for other graph algorithms provided by different graph engines. BC is provided by RealGraph, GraphChi, and X-Stream; RWR is provided by RealGraph, FlashGraph, and X-Stream; BP is provided by RealGraph, GridGraph, and X-Stream; SpMV is provided by RealGraph, GraphChi, and X-Stream. TurboGraph does not provide any additional graph algorithms except for the common graph algorithms (i.e., BFS, PageRank,

Table 2: Execution times of other graph algorithms

Graph engines	Graph algorithms			
	BC	RWR	BP	SpMV
RealGraph	125.7	313.9	3017.3	112.2
TurboGraph	-	-	-	-
GridGraph	-	-	-	226.9
GraphChi	-	6082.6	8418.3	-
FlashGraph	O.O.M	-	-	-
X-Stream	O.O.T	8016.3	O.O.M	538.3

(Execution time unit: sec.)

Table 3: Performance comparison of RealGraph with distributed-system based graph engines

Graph engines	Graph algorithms		System environment			
	WCC	PageRank	Machine	#Cores	Mem. Size	Storage
RealGraph	34	94	Single PC with Intel i7	4	16G	SSD
PowerGraph	-	72	64 Amazon EC2 cc1.4xlarge	8	23G	HDD
GraphLab	244	249	16 Amazon EC2 m2.4xlarge	8	68G	HDD
GraphX	251	419	16 Amazon EC2 m2.4xlarge	8	68G	HDD
Giraph	200	596	16 Amazon EC2 m2.4xlarge	8	68G	HDD

(Execution time unit: sec.)

and WCC). The number of threads of all graph engines was identically set as 8. We performed the algorithms on the largest Yahoo dataset.

Table 2 shows the execution times of the algorithms on the graph engines. The hyphen (-) means that the execution time is not measured because the graph engine does not provide the corresponding algorithm. The experimental results show that RealGraph has the best performance for all graph algorithms, which proves that RealGraph are effective in various domains using real-world graphs.

6.4 Comparisons with distributed-system based graph engines

Finally, we compared RealGraph with four distributed-system based graph engines: PowerGraph [9], GraphLab [16], GraphX [26], and Giraph [1]. Since prior research [9, 26] has evaluated the performance of these four distributed-system based graph engines, we *quote* their results for comparison while measuring the performance of our RealGraph. Table 3 presents the performance comparison between RealGraph and distributed-system based graph engines for WCC and PageRank which are the only graph algorithms whose performances on the four distributed-system based graph engines are commonly reported [9, 26]) on the Twitter dataset. Table 3 also shows the system environment of each graph engine. where *Machine* indicates the number of and type of the machines, and for each machine, *#Cores* means the number of cores, *Mem. Size* does the size of memory, and *Storage* does the type of storage. The hyphen (-) means that no result is available.

Under typical configuration, the distributed-system based graph engine uses 16 or more machines, each of which has 8 cores and 20GiB or more memory. In other words, they have computing power *much* higher than RealGraph. Nonetheless, the result shows that RealGraph equipped with low computing power performs better than distributed-system based engines except PowerGraph.

When executing PageRank, PowerGraph employs 64 machines, each of which is equipped with 8 cores, while RealGraph uses a single machine with 4 cores. That is, PowerGraph uses 128 times more cores than RealGraph, notwithstanding other computing resources. The performance of PowerGraph, however, is only 25% better than that of RealGraph. Simply put, RealGraph achieves reasonable performance using much smaller computing resources.

The reasons are as follows. The graph algorithm is performed through multiple iterations, producing intermediate results proportional to the number of nodes at each iteration. To proceed to the next iteration, each machine in the distributed graph engine should maintain the same intermediate results. Thus, intermediate results should be synchronized through communication among machines, which may cause more overhead than processing of the graph algorithm [5]. We claim that RealGraph is more effective if multiple machines are not required and the given a graph can reside within a single machine.

7 CONCLUSIONS

Most real-world graphs follow the power-law degree distribution, which with a small number of hub nodes having a high degree and most nodes having a low degree. We have observed two important problems that occur when existing single-machine based graph engines process such real-world graphs: (1) inefficient scanning of the sparse indicator and (2) the delay for the progress of the iterations due to uneven workload distribution.

In order to solve these problems, we developed a new graph engine, RealGraph, designed on the basis of well-known database systems. We applied not only an efficient data layout with good data locality, but also following two techniques that efficiently process real-world graphs into RealGraph.

First, we proposed a *hierarchical indicator* where the higher-level indicator compresses a range of a lower-level indicator using a single bit. This helps to skip unnecessary ranges of each lower-level indicator by scanning each higher-level indicator, which provides efficient indicator scanning at every iteration.

Second, we devised a *block-based workload allocation* where RealGraph assigns blocks to threads. Since a block is a fixed-size standard I/O unit, the total size of objects in each block is almost the same. This prevents the delay for the progress of the next iteration.

Through extensive experiments, we verified the superiority of RealGraph. We first evaluated the effectiveness of each proposed technique. Each technology improves the performance of RealGraph, and the combination of technologies further improves its performance with synergy. For various graph algorithms and real-world graphs, RealGraph has the best performance in terms of speed and scalability compared to existing single-machine based and distributed-system based graph engines.

ACKNOWLEDGMENTS

This work was supported by (1) the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (MSIT) (NRF-2017R1A2B3004581) and (2) Next-Generation Information Computing Development Program through NRF funded by MSIT (NRF-2017M3C4A7069440). Also, we appreciate Samsung Electronics' university program [Flash Solutions for Emerging Applications] that significantly helps train our lab. members.

REFERENCES

[1] Ching Avery. 2011. Giraph: Large-scale graph processing infrastructure on hadoop. In *Hadoop Summit*. 5–9.

[2] Ulrik Brandes. 2001. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology* 25, 2 (2001), 163–177.

[3] Michael J Carey, David J DeWitt, Michael J Franklin, Nancy E Hall, Mark L McAuliffe, Jeffrey F Naughton, Daniel T Schuh, Marvin H Solomon, CK Tan, Odysseas G Tsatalos, Seth J White, and Michael J Zwilling. 1994. Shoring up persistent applications. In *Proceedings of the ACM international conference on management of data (SIGMOD)*. 383–394.

[4] Michael J Carey, David J DeWitt, Joel E Richardson, and Eugene J Shekita. 1986. *Object and file management in the EXODUS extensible database system.* University of Wisconsin-Madison. Computer Sciences Department.

[5] Rong Chen, Xin Ding, Peng Wang, Haibo Chen, Binyu Zang, and Haibing Guan. 2014. Computation and communication efficient graph processing with distributed immutable view. In *Proceedings of the international symposium on high-performance parallel and distributed computing (HPDC)*. 215–226.

[6] Yuze Chi, Guohao Dai, Yu Wang, Guangyu Sun, Guoliang Li, and Huazhong Yang. 2016. NXgraph: An efficient graph processing system on a single machine. In *Proceedings of the IEEE international conference on data engineering (ICDE)*. 409–420.

[7] H-T Chou, David J Dewitt, Randy H Katz, and Anthony C Klug. 1985. Design and implementation of the Wisconsin storage system. *Software: Practice and Experience* 15, 10 (1985), 943–962.

[8] Martin Erwig. 1992. Graph algorithms= iteration+ data structures?. In *International workshop on graph-theoretic concepts in computer science*. 277–292.

[9] Joseph E Gonzalez, Yucheng Low, Haijie Gu, Danny Bickson, and Carlos Guestrin. 2012. PowerGraph: Distributed graph-parallel computation on natural graphs. In *Proceedings of the USENIX symposium on operating systems design and implementation (OSDI)*. 17–30.

[10] Minyang Han and Khuzaima Daudjee. 2015. Giraph unchained: barrierless asynchronous parallel execution in pregel-like graph processing systems. *Proceedings of the VLDB Endowment* 8, 9 (2015), 950–961.

[11] Min-Hee Jang, Christos Faloutsos, Sang-Wook Kim, U Kang, and Jiwoon Ha. 2016. Pin-trust: Fast trust propagation exploiting positive, implicit, and negative information. In *Proceedings of the ACM international conference on information and knowledge management (CIKM)*. 629–638.

[12] Yong-Yeon Jo, Jiwon Hong, Myung-Hwan Jang, Jae-Geun Bang, and Sang-Wook Kim. 2016. Data Locality in graph engines: Implications and preliminary experimental results. In *Proceedings of the ACM international conference on information and knowledge management (CIKM)*. 1885–1888.

[13] Min-Soo Kim, Kyuhyeon An, Himchan Park, Hyunseok Seo, and Jinwook Kim. 2016. GTS: A fast and scalable graph processing method based on streaming topology to GPUs. In *Proceedings of the ACM international conference on management of data (SIGMOD)*. 447–461.

[14] Aapo Kyrola, Guy E Blelloch, and Carlos Guestrin. 2012. GraphChi: Large-scale graph computation on just a pc. In *Proceedings of the USENIX symposium on operating systems design and implementation (OSDI)*. 31–46.

[15] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data* 1, 1 (2007), 1–41.

[16] Yucheng Low, Danny Bickson, Joseph Gonzalez, Carlos Guestrin, Aapo Kyrola, and Joseph M Hellerstein. 2012. Distributed GraphLab: A framework for machine learning and data mining in the cloud. *Proceedings of the VLDB Endowment* 5, 8 (2012), 716–727.

[17] Lingxiao Ma, Zhi Yang, Han Chen, Jilong Xue, and Yafei Dai. 2017. Garaph: Efficient GPU-accelerated graph processing on a single machine with balanced replication. In *Proceedings of the USENIX annual technical conference (ATC)*. 195–207.

[18] Steffen Maass, Changwoo Min, Sanidhya Kashyap, Woonhak Kang, Mohan Kumar, and Taesoo Kim. 2017. Mosaic: Processing a trillion-edge graph on a single machine. In *Proceedings of the ACM european conference on computer systems (EuroSys)*. 527–543.

[19] Grzegorz Malewicz, Matthew H Austern, Aart JC Bik, James C Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. 2010. Pregel: A system for large-scale graph processing. In *Proceedings of the ACM international conference on management of data (SIGMOD)*. 135–146.

[20] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.

[21] Amitabha Roy, Ivo Mihailovic, and Willy Zwaenepoel. 2013. X-Stream: Edge-centric graph processing using streaming partitions. In *Proceedings of the ACM symposium on operating systems principles (SOSP)*. 472–488.

[22] Robert Sedgewick and Kevin Wayne. 2011. *Algorithms*. Addison-wesley professional.

[23] Kenji Suzuki, Isao Horiba, and Noboru Sugie. 2003. Linear-time connected-component labeling based on sequential local operations. *Computer Vision and Image Understanding* 89, 1 (2003), 1–23.

[24] Guozhang Wang, Wenlei Xie, Alan J Demers, and Johannes Gehrke. 2013. Asynchronous large-scale graph processing made easy. In *Proceedings of biennial conference on innovative data systems research (CIDR)*. 3–6.

[25] Wook-Shin, Sangyeon Lee, Kyungyeol Park, Jeong-Hoon Lee, Min-Soo Kim, Jinha Kim, and Hwanjo Yu. 2013. TurboGraph: A fast parallel graph engine handling billion-scale graphs in a single PC. In *Proceedings of the ACM international conference on knowledge discovery and data mining (KDD)*. 77–85.

[26] Reynold S Xin, Joseph E Gonzalez, Michael J Franklin, and Ion Stoica. 2013. GraphX: A resilient distributed graph system on spark. In *Proceedings of the international workshop on graph data management experiences and systems (GRADE)*. 1–6.

[27] Xintian Yang, Srinivasan Parthasarathy, and Ponnuswamy Sadayappan. 2011. Fast sparse matrix-vector multiplication on GPUs: Implications for graph mining. *Proceedings of the VLDB Endowment* 4, 4 (2011), 231–242.

[28] Hilmi Yildirim and Mukkai S Krishnamoorthy. 2008. A random walk method for alleviating the sparsity problem in collaborative filtering. In *Proceedings of the ACM conference on recommender systems (RecSys)*. 131–138.

[29] Da Zheng, Disa Mhembere, Randal Burns, Joshua Vogelstein, Carey E Priebe, and Alexander S Szalay. 2015. FlashGraph: Processing billion-node graphs on an array of commodity SSDs. In *Proceedings of the USENIX conference on file and storage technologies (FAST)*. 45–58.

[30] Xiaowei Zhu, Wentao Han, and Wenguang Chen. 2015. GridGraph: Large-scale graph processing on a single machine using 2-level hierarchical partitioning. In *Proceedings of the USENIX annual technical conference (ATC)*. 375–386.

(Mis)Information Dissemination in WhatsApp: Gathering, Analyzing and Countermeasures

Gustavo Resende
Universidade Federal de Minas Gerais, Computer Science Department, Brazil
gustavo.jota@dcc.ufmg.br

Philipe Melo
Universidade Federal de Minas Gerais, Computer Science Department, Brazil
philipe@dcc.ufmg.br

Hugo Sousa
Universidade Federal de Minas Gerais, Computer Science Department, Brazil
hugosousa@dcc.ufmg.br

Johnnatan Messias
Max Planck Institute for Software Systems (MPI-SWS), Germany
johnme@mpi-sws.org

Marisa Vasconcelos
IBM Research
marisaav@br.ibm.com

Jussara M. Almeida
Universidade Federal de Minas Gerais, Computer Science Department, Brazil
jussara@dcc.ufmg.br

Fabrício Benevenuto
Universidade Federal de Minas Gerais, Computer Science Department, Brazil
fabricio@dcc.ufmg.br

ABSTRACT

WhatsApp has revolutionized the way people communicate and interact. It is not only cheaper than the traditional Short Message Service (SMS) communication but it also brings a new form of mobile communication: the group chats. Such groups are great forums for collective discussions on a variety of topics. In particular, in events of great social mobilization, such as strikes and electoral campaigns, WhatsApp group chats are very attractive as they facilitate information exchange among interested people. Yet, recent events have raised concerns about the spreading of misinformation in WhatsApp. In this work, we analyze information dissemination within WhatsApp, focusing on publicly accessible political-oriented groups, collecting all shared messages during major social events in Brazil: a national truck drivers' strike and the Brazilian presidential campaign. We analyze the types of content shared within such groups as well as the network structures that emerge from user interactions within and cross-groups. We then deepen our analysis by identifying the presence of misinformation among the shared images using labels provided by journalists and by a proposed automatic procedure based on Google searches. We identify the most important sources of the fake images and analyze how they propagate across WhatsApp groups and from/to other Web platforms.

CCS CONCEPTS

• **Networks** → **Online social networks**; • **Mathematics of computing** → *Network flows*; • **Information systems** → *Mobile information processing systems*; *Chat*.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313688

KEYWORDS

WhatsApp groups, misinformation, information dissemination, social network structure, fake images

ACM Reference Format:
Gustavo Resende, Philipe Melo, Hugo Sousa, Johnnatan Messias, Marisa Vasconcelos, Jussara M. Almeida, and Fabrício Benevenuto. 2019. (Mis)Information Dissemination in WhatsApp: Gathering, Analyzing and Countermeasures. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313688

1 INTRODUCTION

WhatsApp is a world-wide popular messaging app with more than 1.5 billion active users [2] which is currently the main messaging app in many countries, including India, Brazil, and Germany. Nearly everyone with a smartphone uses WhatsApp in Brazil (about 120 million active users [15]) to keep in touch with friends and family, do business, as well as read the news.

There are key features in WhatsApp that make this app unique. First, any communication within the app is end-to-end encrypted, meaning that messages, photos, videos, voice messages, documents, status updates, and calls are only seen by those involved in the communication. Second, WhatsApp allows users to easily create and organize chat groups. These groups, which are limited to 256 members, are by default private, as group administrators decide who can join them. However, a group manager may choose to share the link to join it in websites or social networks. In such a case, anyone with access to the link can join the group, which becomes, from a practical perspective, publicly accessible. Finally, WhatsApp provides features for viral spreading, allowing users to broadcast an initial message to 256 contacts or groups or forward content to 20 contacts or groups[1].

[1]The message forwarding was limited to 5 groups in India and 20 in the rest of the world along the period this work was developed. Currently, the limit has been updated to 5 worldwide.

Recent events have raised serious concerns that WhatsApp can become a fertile ground for groups interested in disseminating misinformation, especially as part of articulated political campaigns. In 2018, unfounded allegations disseminated over WhatsApp have fueled mob lynching in India that killed more than 20 people in a two-month window [9]. The 2018 Brazilian elections experienced an information war organized within WhatsApp where false rumors, manipulated photos, decontextualized videos, and audio hoaxes have become campaign ammunition and went viral on the platform with no way to monitor their full reach or origin [15].

This paper provides a large scale investigation of information dissemination within *WhatsApp groups*. We focus on *political-oriented* publicly accessible groups as we expect greater user engagement in topics of stronger social impact. We also offer a first look into *misinformation* dissemination within *WhatsApp*. More specifically, we tackle the following research questions.

RQ1: What kind of content is shared in WhatsApp *publicly accessible* groups? Are there fake news or misinformation in these messages? **RQ2:** What is the interplay between WhatsApp and other Web platforms (i.e. social networks such as Twitter, forums, and websites) in the dissemination of political content and, in particular, misinformation?

To answer these questions, we first identify publicly accessible groups related to Brazilian politics in WhatsApp, by searching the Web and other social networks such as Twitter and Facebook for invitation links to WhatsApp groups. These groups are suitable for activism and political engagement, making them a potential target of misinformation campaigns that might attempt to maximize the audience of a story with misinformation by sharing it with people that are engaged in supporting political candidates. We joined those groups and gathered the content shared within them for time periods corresponding to two major social mobilization events in Brazil: (i) a national truck drivers' strike[2] (May 21^{st} to June 2^{nd}, 2018); and (ii) the first round of the 2018 Brazilian general elections campaign (August 16^{th} to October 7^{th}, 2018), with 141 and 364 groups monitored, respectively.

We start our investigation by first analyzing the content shared and the user interactions within the monitored WhatsApp groups to understand how users disseminate information in such environments. Our results show that images are often the most shared type of media, and they usually carry satires, news, and activism-related content. We also show that WhatsApp has a network configuration similar to many other online social networks (e.g., Twitter or Facebook) which connects thousands of users. Thus, it has the potential to make any information become viral. Moreover, we analyze how the content dissemination crosses the boundary between WhatsApp and other Web platforms.

Further, we explore the presence of misinformation campaigns in the monitored groups. We identify misinformation in image content by relying on two sources: (i) a Brazilian fact-checking agency; and (ii) a proposed automatic procedure that exploits the results of Google searches to identify images that appear in well-known fact-checking websites. Our results show a considerable number of images checked as containing misinformation, which was largely disseminated in the monitored groups. We assess the occurrence of such images in Web domains and Twitter accounts then analyze the network of misinformation propagation. Our analyses reveal that a few groups are the most responsible for disseminating images with misinformation. Moreover, by comparing the timestamps when an image first appeared on WhatsApp (as captured by our data) and on other Web applications, we find that WhatsApp was the primary source of 30% of the identified images containing misinformation.

The remainder of the paper is organized as follows. Next section discusses related work. Then, Section 3 details our methodology and summarizes the data gathered. Section 4 characterizes the content, the network structure of the groups we monitored and how information propagates in such groups. Section 5 presents our main results on the dissemination of misinformation in those groups. Finally, Section 6 concludes the paper.

2 RELATED WORK

Fake news have encountered a suitable means for fast, cheap, and easy dissemination in social media systems. Indeed, these platforms have been a main vehicle for public opinion manipulation and fake news dissemination [19, 21, 25]. Studies during the 2016 U.S. presidential election campaign observed a strong correlation of the number of visits to fake news websites (i.e., sites that deliberately publish hoaxes and misinformation) and aggregate voting patterns at state and county levels [7].

Social bots are one of the most common types of manipulation attack, emulating real users, posting content and interacting with real users and other bots [5, 16]. They were used on Twitter during the 2016 U.S. presidential campaign to manipulate discussion [1] reaching about 20% of all posts about presidential elections. Another misinformation campaign was observed during the 2017 French presidential election, in which bots' posts with unauthentic documents about a candidate quickly spread on Twitter two days before the final voting [4]. Facebook was also a target of misinformation spread aiming at influencing American voters during the 2016 presidential campaign. Using Facebook Ads platform, groups linked to the Russian Intelligence Research Agency (IRA) bought about 3,000 ads linked to 470 user accounts targeting voters from the swing states [12, 21]. Since then, Facebook has performed measures to mitigate fake news dissemination such as removing fake accounts related to political movements and working directly with fact-checking websites [13]. However, fake news is not only disseminated exclusively by social bots or through ads but also by real users. A recent study analyzed over 126,000 cascades of fact-checked news stories on Twitter, finding that fake news was 70% more likely to be retweeted than true stories, and humans are more likely to spread fake news than bots [25].

More recently, WhatsApp has also become a powerful tool to influence people during political campaigns, especially in countries in South America, Africa, and Southeast Asia. In these locations, WhatsApp groups constitute the majority of online communication, enabling politicians to reach a larger share of voters, even those residing in areas with precarious Internet connectivity. Other features that make communication via WhatsApp attractive for this kind of marketing are: it is cheaper, messages are often not contextualized and it can be used to target small groups with specific messages

[2]This was a massive event, with strong political connotation, that affected the whole country, paralyzing many economic and social sectors of the nation.

[20, 23]. This was observed in Brazil, where family groups were responsible for 51% of the dissemination of fake news on WhatsApp during the period of the 2018 presidential elections [10]. Because of the end-to-end encryption, it is hard to track the dissemination of misinformation in such platforms. Yet, recent efforts have gathered and analyzed data from WhatsApp chat groups [8, 20], focusing on textual interactions in these groups. Our present effort provides a deeper understanding of the content exchanged in WhatsApp, unveiling, among other findings, the spread of misinformation campaigns through images in the platform.

3 METHODOLOGY

In this section, we present how we gathered data of WhatsApp groups and the methods used to process and analyze it.

3.1 Data Collection

As a first step of our data collection, we had to identify a considerable number of publicly accessible groups. To that end, we used the URL pattern `"chat.whatsapp.com"`, which is commonly used in invitations to join WhatsApp groups, as a search query and submitted it to Google, Twitter, and Facebook search engines. We restricted our search space to groups related to Brazilian politics, by including in each search query a word from a dictionary related to the 2018 Brazilian elections[3]. This dictionary contains the name of politicians, political parties, as well as words associated with political extremism. Finally, we performed a manual inspection of the collected group names to filter out those unrelated to politics. In total, we found 3,444 distinct links for publicly accessible groups, out of which only 1,828 were valid (i.e., unbroken).

As a second step, we selected a number of valid groups to monitor. This monitoring involves joining each group using a cell phone. Thus, the number of groups (see Section 3.3) monitored was constrained by the available devices and their resources (memory). We joined each selected group using our available cell phones and a tool developed by Garimella *et al.* [8]. We then periodically downloaded all data shared in each group and stored them in a database. Specifically, stored data can be grouped into: images, videos, audio messages, external links, and text messages. From each message, we extracted its group name (i.e., the group the message was posted), a group ID, a user ID, and timestamp. That is, we mapped telephone numbers and user names onto unique user identifiers[4], discarding the original information afterwards. For the media messages, we also downloaded their respective files and used their filenames as a reference to the message.

As we will show in Section 3.3, images are the most frequent type of *media* content in our collected data, as well as an important source of misinformation. Thus, we delved further into the images shared on the monitored groups and developed a tool to collect Web pages in which those images have appeared. The tool exploits the capability of searching for images provided by Google search, where a user can submit an image as query, and obtain as result webpages that include matching images along with their post dates. As will be discussed in Section 4, this information allows us to analyze temporal sharing patterns such as the time interval between first appearance of an image on WhatsApp and on the Web.

In order to explore the content veracity of the images shared in the groups, we extended the tool to automatically identify whether each image had been previously checked as *fake* by a number of fact-checking websites. We further elaborate on this process in Section 5.2.

Table 1: Overview of our datasets.

	Truck Drivers' Strike	Election Campaign
#Groups	141	364
#Total Users	5,272	18,725
#Total Messages	121,781	789,914
#Text Messages	95,424	591,162
#Images	11,610	110,954
#Videos	9,752	73,310
#Audios	4,995	14,488
#URLs	11,728	92,654

3.2 Data Limitation

To our knowledge, this work is the first effort that aims to explore the political debate in WhatsApp. Also, it proposes a methodology to infer which identified publicly accessible groups are related to politics. Unfortunately, we are not aware of an approach that would allow us to assess the representativeness of our data as even the total number of groups available in the country is *not* of public knowledge. We emphasize, though, that all sensitive information (i.e., user names and phone numbers) were *not* stored in our dataset.

3.3 Events Captured in our Dataset

Our data collection focuses on the time period of two major social mobilization events in Brazil: (i) a national truck drivers' strike (May 21^{st} to June 2^{nd} 2018); and (ii) the first round of the 2018 Brazilian presidential elections campaign (August 16^{th} to October 7^{th} 2018). The truck drivers' strike was a stoppage of autonomous truck drivers all over Brazil. The strikers began a mobilization through their leaders in social networks, expressing against frequent adjustments and without minimum predictability in fuel prices, especially diesel, made by the state-owned company Petrobras[5]. The shutdown and blockades of highways in 24 states and in the Federal District caused the unavailability of food and medicine around the country, shortages and high gasoline prices, with long queues to fuel. We analyze the data collected for each aforementioned period as a separate dataset, contrasting our findings across periods.

Table 1 provides an overview of our two datasets, showing the total number of messages shared as well as the number of messages per type of content (text[6], image, video and audios). Note that most shared messages are indeed textual content, but image is the most frequent type of *media* content in both datasets, reaching roughly 10% and 15% of all content shared in the monitored groups during the strike and election period, respectively. Note also the large number of links to websites (last row) present in the text messages.

[3] https://goo.gl/PdwAfV

[4] Throughout the paper we refer to such identifiers as users. Yet, they are indeed unique telephone numbers, as we are not able to identify multiple devices of the same user.

[5] http://www.petrobras.com.br/en/

[6] Only messages that are entirely composed of textual content are counted as text messages.

We shared a sample of our dataset to a set of journalists which led to wide press coverage. Particularly, a BBC story[7] manually analyzed the popular content from the truck drivers' strike data and suggested that it was organized in a decentralized way through WhatsApp groups. Audios were often used to call truck drivers to join and remain in the strike, while videos and images were used to disseminate scenes of the strike and its consequences in different cities. Similarly, another BBC story[8] analyzed popular content collected during a week of the electoral period, reporting presence of misinformation in different kinds of content, but mainly in images. They reported audios describing conspiracy theories, manipulated photos, fake polls, attacks to the traditional media and to famous personalities, as well as the instigation of hate, especially against LGBT and feminists.

4 MESSAGE CONTENT AND DISSEMINATION

In this section, we characterize the content of the messages, the network structure of the groups as well as the temporal patterns of message sharing in our datasets.

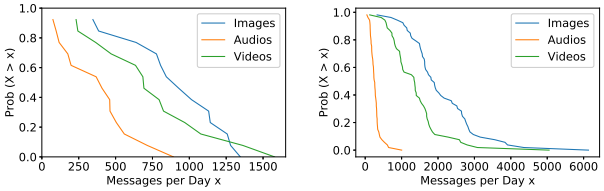
4.1 Media Content and URLs

As our first analysis, we focus on the content shared, notably media content (audios, videos, and images). We also briefly discuss the presence of URLs in textual messages. The discussion is based on Table 1 as well as on Figure 1, which shows the complementary cumulative distribution of the number of messages of different media types shared per day, across all monitored groups.

4.1.1 URLs and Webpage domains. As shown in Table 1, a total of 11,728 URLs were shared (as part of text messages) in the monitored groups during the truck drivers' strike period. During the election campaign, this number reached 92,654 URLs. They correspond to 8% and 9% of the total amount of messages we gathered for each period, respectively. The nature of these URLs varies from links to news websites, blogs, entertainment, and other social networks to even links to other WhatsApp groups. Also, 69% of all URLs shared during the truck drivers' strike period are unique. This fraction drops to 45% during the election campaign, indicating less diversity and more repetition of the links during that period.

4.1.2 Audios. Our datasets contain 4,995 and 14,488 audio messages during the truck drivers' strike and election campaign periods, respectively, which correspond to 4% and 2% of all messages gathered during each respective period. As shown in Figure 1, audio is the least frequent media type shared in the monitored groups. For example, during the truck drivers' strike, up to 450 audios were shared daily, in 60% of the days (see Figure 1(a)). During the election campaign period, up to 500 audios were shared in the same fraction of the monitored days. Note however that, this number reached a peak two days before the election day, with 1,002 audio messages shared on all monitored groups.

4.1.3 Videos. In total, 9,752 videos were shared during the truck drivers' strike, which corresponds to 8% of all messages shared during the period. Also, according to Figure 1, up to 800 videos were shared daily in 60% of the days, reaching up to 1,578 videos on a single day. During the election campaign period, 73,310 videos (9% of all messages) were shared in total, with 1,300 videos being shared daily in 60% of the days (up to 5,052).



(a) Truck Drivers' Strike (b) Election Campaign

Figure 1: Numbers of daily messages with media content shared on all groups.

4.1.4 Images. As mentioned, images represent the most popular media content shared on the monitored groups, with a total of 11,610 images (almost 10% of all messages) and 110,954 images (15%) shared during the truck drivers' strike and election campaign periods, respectively. According to Figure 1, this higher frequency happens on a daily basis. Up to 1,000 and 2,000 images were shared in 60% of the days during the truck drivers' strike and during the election campaign, respectively. Even more, in about 5% of the days, the number of images shared on the groups on a single day exceeded 1,350 and 6,100 in the two periods. For the election campaign (the latter), these days are in the week before the election day.

Given the popularity of images, we next deepen our analysis of the dissemination of type of content in the monitored groups.

4.2 Characterizing WhatsApp Images

A WhatsApp group is usually meant to be a space for discussions about a specific subject such as politics, education, games. However, the content shared itself may diverge from the group subject given the will of their participants. For example, as in other Web systems, WhatsApp groups are susceptible to spam activity (e.g. advertisements or inappropriate content). To understand the kinds of images shared on our selected groups, we first categorize the images by performing content labeling and analyze the distribution of images across categories. We also discuss the appearance of the same images on other websites and social networks.

4.2.1 Content Labeling. We asked three volunteers to label a sample of the most shared images during each period. The sample from the truck drivers' strike period contains a selection of the top-20 most shared images on each day, with a total 220 images. For the election campaign period, the sample contains the top-100 most shared images, considering the whole monitored period. In order to identify duplicates of the same image, we used the *Perceptual Hashing (pHash)* algorithm [17] to calculate a fingerprint for each image. We were then able to group images having the same hash-values based on human eye perception as duplicates.

A taxonomy guideline document with instructions was given to the volunteers with the following directions: (i) observe an image and, read the text on it, if available; (ii) if there is a text, check the existence of any citation to a website or other source; (iii) check if the following content types are present in this post: *Political*

[7] https://www.bbc.com/portuguese/brasil-44325458
[8] https://www.bbc.com/portuguese/brasil-45666742

Table 2: Image categories used to label the sampled images.

Category	Description
Political	Information about a candidate or party
News	News information with a quote
Advertising	Advertisement of product, service or company
Satire	Humorous content regarding current events
Inappropriate	Illicit products, violence, hate speech or pornographic content
Activism	Popular movements and protests
Opinion	Expression of a personal opinion or comment
Others	Image does not fit in any other category

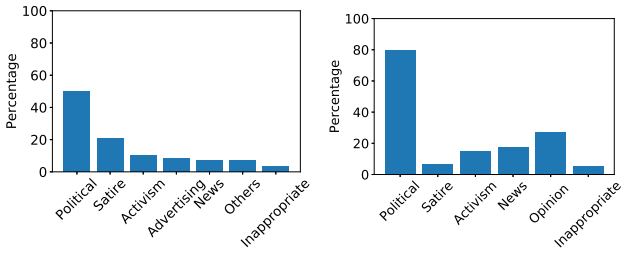
Content; News; Advertising; Opinion; Satire; Activism; (iv) identify possible inappropriate, offensive or even illegal content by checking for the presence of *Dissemination of Hate; Violence; or Promotion of Illicit Products as Inappropriate Content*; (v) you can classify a post with more than one category (e.g., News and Political Content) or none of them; and finally (vi) if you cannot fit the image in any of the listed categories or are unable to establish its category, label the image as *Others*. Table 2 lists the categories used to label the sampled images.

After each of the three volunteers annotated each image according to the categories in Table 2, we measured the inter-annotator agreement in terms of the *Fleiss's* κ [6]. We assumed that consensus was reached if the null hypothesis of negative or no agreement $\kappa = 0$ can be rejected. Since the same image may fit more than one category, we applied the test individually for each category, averaging the κ scores obtained. The test result for the *Others* category indicated a poor agreement for the sample collected during the election campaign period. This result was then disregarded. Overall, we obtained moderate agreement among the annotators for both periods, with average κ equal to 0.6 and 0.42 for the truck drivers' strike and election campaign samples, respectively. This result is reasonable given that some categories are very broad and distinctions are somewhat blurred. In the following, we assume that an image belongs to a category if at least two of the annotators agreed upon that category.

Figure 2 shows the distributions of the image categories in each sample. In both periods, most images are related to politics (50% during the truck drivers' strike and 80% during the election campaign). The large fractions of satire and activism during the truck drivers' strike are also worth noting: we observed a lot of memes about the movement and messages supporting the truck drivers. A small fraction of images shared during that period contains advertisement. In contrast, images with opinions from personalities were much more popular during the election campaign, corresponding to 20% of all images in our sample. News and images with activism were also frequently shared during that period, but no image in our sample contain explicit advertisements.

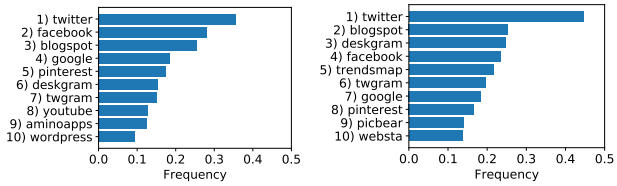
4.2.2 WhatsApp Images on other Websites. We now analyze the extent to which the images shared on our monitored WhatsApp groups have also appeared on other websites including social networks and blogs. We do so by searching for the observed images using the Google Images search engine, as discussed in Section 3.1.

Figure 3 shows the most popular domains returned by Google Images for the images shared in each monitored period. Notice that online social networks like Twitter, Facebook, Aminoapps, and Pinterest are among the most frequent domains where the images were



(a) Truck Drivers' Strike (b) Election Campaign

Figure 2: Distributions of image categories.



(a) Truck Drivers' Strike (b) Election Campaign

Figure 3: Most popular domains for images shared on WhatsApp publicly accessible groups.

Table 3: Sharing of images on monitored WhatsApp groups.

	Truck Drivers' Strike			Election Campaign		
	Mean	Std. Dev	Max	Mean	Std. Dev	Max
#Images per Group	91	143.250	1,011	345	580.531	4,320
#Users per Group	17	21.745	110	34	41.347	211
#Images per User	5	10.831	197	10	32.322	1,612
#Shares for Image (total)	1	1.978	58	1	4.512	125

also posted, for both periods. Similarly, image apps like Deskgram and Twgram as well as Blogspot are popular domains, especially the latter, suggesting that a large fraction of image content shared on WhatsApp groups may indeed have blogs as possible sources. TrendsMap, a website that shows visualizations of the trends on Twitter, was popular for images shared during the election campaign period, while more than 10% of the images shared during the truck drivers' strike period also appeared on YouTube, as thumbnails of videos.

4.3 Network Structure

In this section, we analyze sharing patterns of images on the selected WhatsApp groups by studying the structure of the networks that emerge from the participation of users in different groups. To better understand this network structure, we first discuss some key measures related to the sharing of images within each group. For each period, Table 3 presents averages, standard deviations, and maximum values of the numbers of images shared by each user and within each group as well as number of users sharing images in each group and the total number of times each image was shared (across all groups). Overall, all averages are larger for the election campaign period. Although some differences may be (partially) credited to a longer monitoring period, we note that the intensity of image sharing per user was indeed higher during the campaign, with an average twice as larger and a peak eight times as larger than in the strike period. Similarly, we do observe a significant

Table 4: Network metrics for WhatsApp graphs.

	#Nodes	#Edges	Avg. Degree	Diameter	APL*	Density	LCC**
Group Network Truck Drivers' Strike	136	55	0.809	8	2.975	0.006	25
Group Network Election Campaign	333	842	5.057	8	3.459	0.015	206
User Network Election Campaign	10,860	492,217	90.91	9	3.952	0.008	8,934

*Average Path Length. **Largest Connected Component.

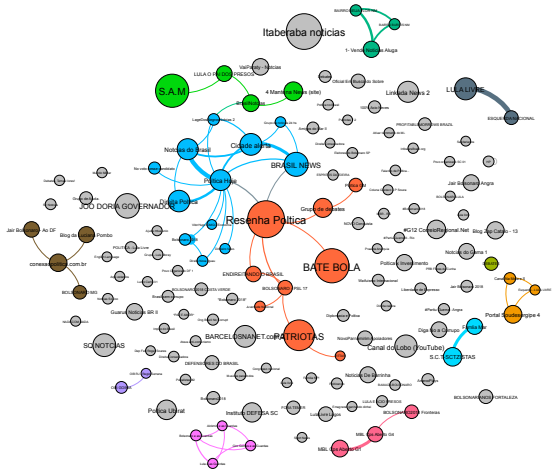
increase in the number of users sharing images in the groups. Note however that most images are shared only a few times (once, on average), as only a few images are widely shared. Interestingly, we found that the groups with the largest number of images shared during the strike (*"Resenha Política"* with 1,011 images) and during the election campaign (*"#BOLSONARO PRESIDENTE"* with 4,320 images) are indeed the groups with the largest numbers of users sharing this type of content (110 and 211, respectively).

We modeled the interactions across groups by means of two network models, one at the group level and one at the user level. That is, we built a *group network* where each node represents one monitored group and edges are added connecting groups that have at least one member in common sharing image content. Figure 4 shows the group networks built for each monitored period. The size of each node represents the number of users who shared image content in the group. Although many groups are somewhat isolated or weakly connected to the rest, we do note the presence of several clusters of groups which are strongly interconnected by sharing many members in common. This may facilitate the flow of information across group boundaries.

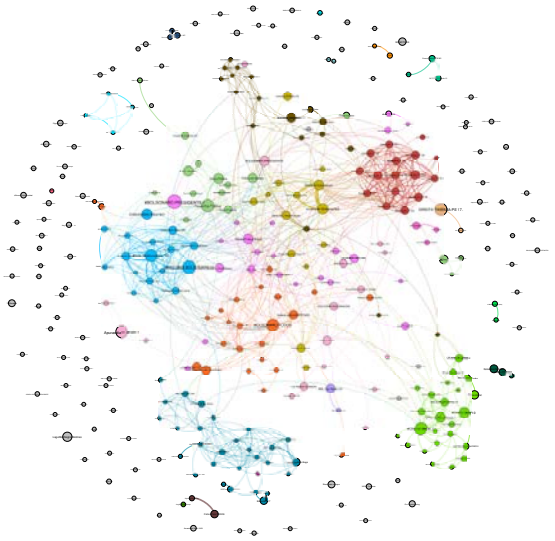
We also modeled the relationship between users by building a *user network* where each node is a user and an edge is added between two nodes if the corresponding users have shared image content in at least one group in common. Node size represents the number of groups in which the user shared images. The user networks are naturally larger and harder to visualize. For illustration purposes, Figure 5 shows a subgraph of the network built for the election campaign, with 5,700 nodes. The network structure of the groups is evidenced by the clusters formed. We note a large number of users blending together connecting to each other inside those groups. Most users indeed form a single cluster, connecting mostly to other members of the same community. On the other hand, there are also a few users who serve as bridges between two or more groups linked by multiple users at the same time. Furthermore, a few users work as big central hubs, connecting multiple groups simultaneously. Lastly, some groups have a lot of users in common, causing these groups to be strongly inter-connected, making it even difficult to distinguish them.

To better understand the properties of these graphs, Table 4 shows various network metrics computed for the group and user networks. It presents numbers of nodes and edges, average node degree, network diameter, average path length (APL), network density, and the size of the largest connected component (LCC).

Focusing on the group networks, we see that the graph for the election campaign is more complex and more densely connected, with more clusters (i.e., communities) of groups and edges emerging



(a) Truck Drivers' Strike Network



(b) Election Campaign Network

Figure 4: Group networks (nodes are groups and edges connect groups with users in common).

between them, and a larger fraction of nodes belonging to the largest connected component (62%). Despite such differences, the average path length between the groups is only slightly larger (3.46, against 2.97 in the strike period). Also, although network density (ratio of number of edges in the graph to the maximum number of edges possible) is low for both periods (under 2%), it is higher during the election campaign. We observe similar properties in the user network with a small average shortest path length (3.95) and higher largest connected component (82% of the users).

Therefore, as illustrated in Figures 4 and 5 and in Table 4, WhatsApp is more than just a mobile network that provides end-to-end encrypted communication between two users. It exhibits network properties very similar to many other social networks such as Twitter or Facebook, connecting thousands of users and having the potential to make a piece of information become viral.

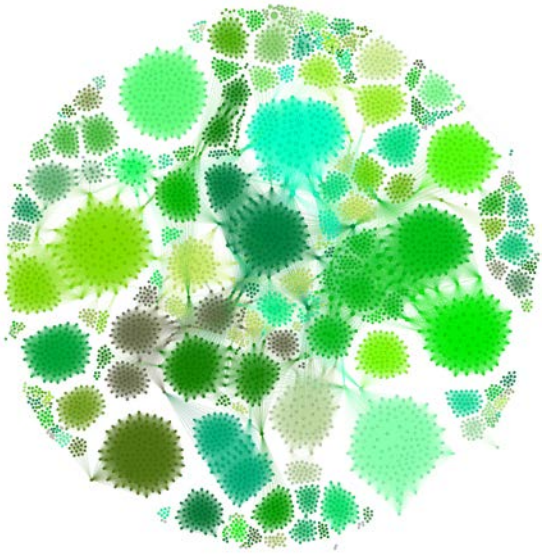


Figure 5: User network (nodes are users and edges connect users with group in common. Subgraph of election period).

4.4 Propagation Dynamics

We analyze the propagation dynamics of images within the monitored WhatsApp groups by means of two metrics, namely lifetime and burst time. The lifetime of an image is the time interval between the first and the last time the image was shared, as captured by our datasets. Burst time is the time interval between consecutive shares of the same image, irrespective of the group. In our analysis, we first identify the set of images shared during each monitored period and then compute their lifetimes and burst times considering an extended monitored period from April 23^{rd} to October 22^{nd} 2018. We also analyze the propagation dynamics of images as they cross the boundaries of WhatsApp, appearing elsewhere on the Web.

4.4.1 Lifetimes. As shown in Figure 6(a), the distributions of lifetimes are mostly similar[9]: around 20% of the images have lifetimes under 1 hour, and 40% of them have lifetimes under 20 hours. Yet, many images last quite longer on the system: more than 30% of the images have lifetimes exceeding 100 hours.

4.4.2 Burst Times. Figure 6(b) shows the distributions of burst times. Once again, both distributions are mostly similar, although there is a tendency of images being reshared within shorter time intervals during the election campaign. For example, in 40% of the cases, images were reshared within up to 120 and 100 minutes during the strike and election campaign periods, respectively. Yet, some images were reshared very sporadically: in around 20% of the cases, burst times exceed 5 and 2 days during the same periods.

4.4.3 Propagation to and from the Web. We also analyze the propagation of images across the boundaries between WhatsApp and the Web. Specifically, we analyze the difference between the time an image was first shared on a monitored group and the time when it was indexed by Google. The latter is taken as an estimate of the time it first appeared on the Web. A positive difference suggests that the image was first shared in one of the monitored groups and then published on the Web. A negative difference may suggest the image was first posted on the Web[10]. Figure 6(c) shows the cumulative distribution of such time differences for images shared during the two monitored periods. Note that, in both cases, the time differences are indeed negative for most images (80%). Yet, 14% of the images were posted on the same day on the Web and on the WhatsApp groups, and 6% were first shared on WhatsApp.

[9]We note that the somewhat shorter lifetimes for images shared during the election campaign may be a side effect of the interruption of monitoring.

5 MISINFORMATION ON WHATSAPP

In this section, we look at the presence of misinformation in the images shared on WhatsApp groups. First, we discuss two techniques used to identify misinformation in the images in our datasets. We analyze their characteristics and propagation, and compare these images with the rest of our WhatsApp data.

5.1 Labeling with a Fact-Checking Agency

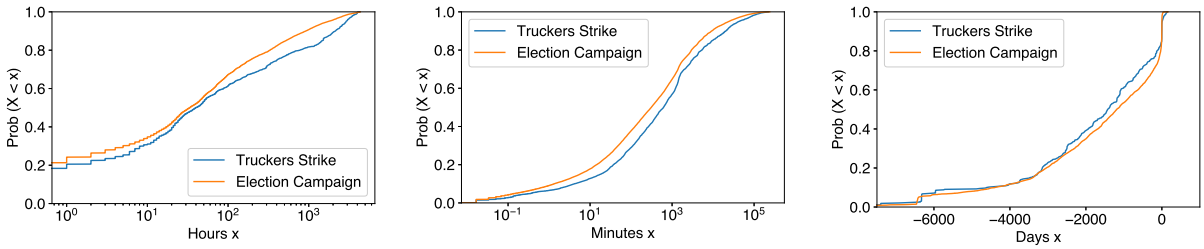
We created a list of the most shared images during the election campaign period and gave them to one of the most important fact-checking agencies in Brazil, *Lupa* [11]. They checked the veracity of each of these images following a methodology similar to other fact-checking agencies around the world (e.g., the American *Politifact*[12] and the Argentinian *Chequeado*[13]). They first analyzed where these images contained factual information as opposed to opinions since it is not possible to check the latter. Out of a total of 61 images, 47 were marked as factual. Out of these factual images, they found that 22 had already been checked by other fact-checking agencies: 17 images had been checked as containing misinformation, and only 5 images had been checked as true. These results show an expressive number of images with misinformation in WhatsApp during the 2018 Brazilian elections. In terms of percentages, 36.2% of the images with factual information where checked as containing misinformation, whereas 53.2% of them include misleading and inconclusive content (not supported by public information), and only 10.6% were verified as true. Examples of images checked as misinformation are shown in Figure 7.

Figure 7(a) is an edited image of the Brazilian former president Dilma Rousseff, who was impeached in 2016 [22], alongside Fidel Castro, former president of Cuba. At the time this picture of Castro was taken, Dilma was 11 years old. Thus, the image is clearly fake. It was the most popular image in the analyzed period. Figure 7(b) is an edited image of the former Brazilian president Lula, imprisoned for corruption at the time of monitoring [3, 24], meeting the aggressor responsible for stabbing the then presidential candidate Jair Bolsonaro during a campaign rally [18]. The intention of the image was to associate Lula with the attack against Bolsonaro.

5.2 An Automatic Methodology for Finding Misinformation

Recently, Facebook has announced partnerships with many third-party fact-checking organizations, through which Facebook demote or reduce the visibility of links rated as false [14]. This kind of

[10]Our analysis is constrained by the view of WhatsApp provided by our datasets.
[11]https://piaui.folha.uol.com.br/lupa
[12]https://www.politifact.com
[13]https://chequeado.com



(a) Lifetimes (b) Burst times (c) Interval between first share on WhatsApp and Web

Figure 6: Distributions of temporal properties characterizing propagation dynamics of images.



(a) Fabricated image of former president Dilma Rousseff next to Fidel Castro. (b) Fabricated image of Bolsonaro's aggressor next to former president Lula.

Figure 7: Images checked as containing misinformation by both fact-checking methodologies.

partnership neglects misinformation in images, as fact-checkers only provide rates to links containing stories with misinformation. Next, we provide a strategy to connect the false stories found in images shared with external links that appear on the Web, providing a simple way for Facebook to demote links containing images with misinformation identified on WhatsApp.

First, we identified the main fact-checking agencies in Brazil[14]. We then automatized the process of searching each image shared on the WhatsApp groups on the Web by using the Google Image search. Given the search results for an image, we checked whether any of the returned pages belong to one of the fact-checking domains. If so, we parsed the fact-checking page and automatically labeled the image as fake or true depending on how the image was tagged on the fact-checking page.

We applied this methodology to all images in both datasets. We found only 2 images with misinformation in the truck drivers' strike dataset and 70 images containing misinformation in the elections dataset. Thus, we restricted our focus to the election campaign period. We compared the 70 images with misinformation identified by the automatic process with the 17 images checked as fake by Lupa (see previous section), obtaining an overlap of only 2 images. Thus we built a single dataset of 85 images with misinformation identified by official fact-checking agencies.

In the following, we analyze the images in this dataset, focusing on other websites where they also appear. We also compare properties of these images with those of the other images shared during the election campaign period. To distinguish between them we refer to the former as misinformation and to the later as unchecked, since the veracity of their content was not necessarily checked. We cannot guarantee the absence of misinformation in the unchecked images, given that such an assertion is restricted by the availability of checked facts. Yet, we expect that we were able to catch most images containing misinformation in our dataset, especially those with greater impact on users, as they most probably were identified by the fact checkers.

[14]Fact-checking agencies: *Boatos.org*: https://www.boatos.org; *e-Farsas*: http://www.e-farsas.com; *Comprova*: https://projetocomprova.com.br; *Lupa*: https://piaui.folha.uol.com.br/lupa; *Globo G1*: http://g1.globo.com/fato-ou-fake; and *Aos Fatos*: https://aosfatos.org.

Table 5: Overview of images shared during election campaign period: misinformation versus unchecked content.

	Misinformation	Unchecked
#Groups in which images were shared	157	351
#Users who shared images	624	10,339
#Unique images	85	69,590
#Total shares of images	1,168	109,791

5.3 Images with misinformation on WhatsApp and on the Web

Table 5 presents a comparison of the images with misinformation and with unchecked content shared during the election campaign, showing the numbers of distinct images, users who shared those images, groups in which those images were shared and total number of shares. Note that, even though the number of distinct images with misinformation is small (85), these images summed up 1,168 shares posted by 624 different users in 157 different groups. Despite representing less than 1% of all images shared, these images appeared in 44% of the monitored groups in the period of the election campaign, effectively reaching a large user population. Also, note that nearly 5.7% of all users shared images with misinformation.

5.3.1 Network of propagation in WhatsApp groups. We analyzed the propagation of images with misinformation on the WhatsApp groups by building a network model representing the groups in which the images with misinformation *first appeared*. Specifically, we built a directed graph where each node represents a group and a directed edge from node A to node B was added if the same image with misinformation was first shared in group A and then appeared in group B. To build this graph we considered only groups in which at least 2 distinct images with misinformation were shared during the period. The weight of an edge is defined as the number of images containing misinformation that were first shared in a group and

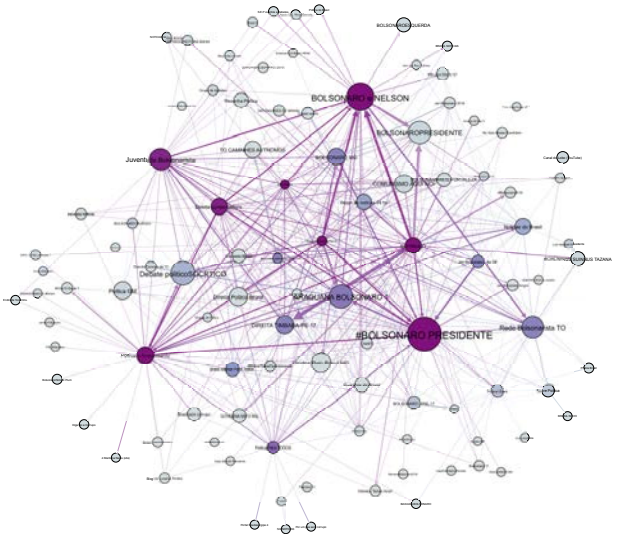


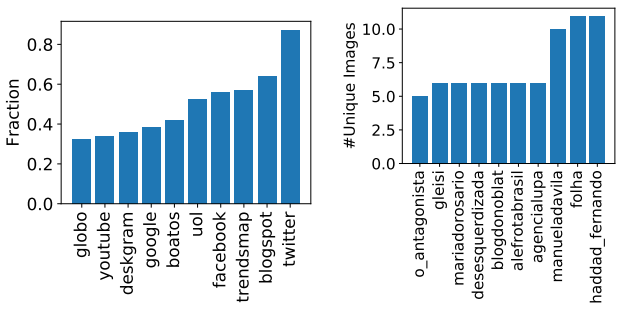
Figure 8: Network of misinformation propagation on WhatsApp groups (election campaign period).

then co-occurred in the other. The size of a node represents the number of images with misinformation posted on that particular group while the color represents the sum of the outgoing edges, that is, the total number of images that were "first seen" in that group and then spread to the rest of the network.

Figure 8 shows the network of propagation of images with misinformation in the monitored WhatsApp groups during the election campaign. Note that some nodes are darker (larger out-degree) than others, suggesting they are the main "seeds" of the images with misinformation in the graph. It is worth noting that the group in which the largest number of images with misinformation first appeared (largest node) is indeed the group with the largest number of users and largest number of images shared in general. Yet, we note that some large nodes have very light colors (e.g. *"ARAGUANA BOLSONARO 1"* and *"BOLSONAROPRESIDENTE"*), meaning that although many images containing misinformation were shared in them, they acted more as receptors than seeds, since their out-degrees are small. These results seem to suggest that fewer groups are responsible for the spreading of a large fraction the images with misinformation in WhatsApp.

5.3.2 Presence on the Web. We also analyzed the presence of images with misinformation on the Web. Figure 9(a) shows that these images frequently appeared in other social networks and blogs, notably Twitter, as also observed for images in general (Figure 3). Figure 9(b) shows the Twitter accounts that most shared these images, presenting the number of different images with misinformation tweeted by each account. Among them, there are some official journalistic accounts (*folha* and *agencialupa*) and official accounts of the presidential candidate Fernando Haddad and his vice, Manuela d'Avila. We note these profiles posted images containing misinformation with the purpose of repudiating them, acting as fact-checking accounts. Yet, some other accounts acted as misinformation broadcasters by spreading it further through the network.

5.3.3 Propagation Dynamics. Recall that, in Section 4.4 we analyzed image propagation dynamics by characterizing their lifetimes



(a) Most popular domains. (b) Most popular Twitter accounts.

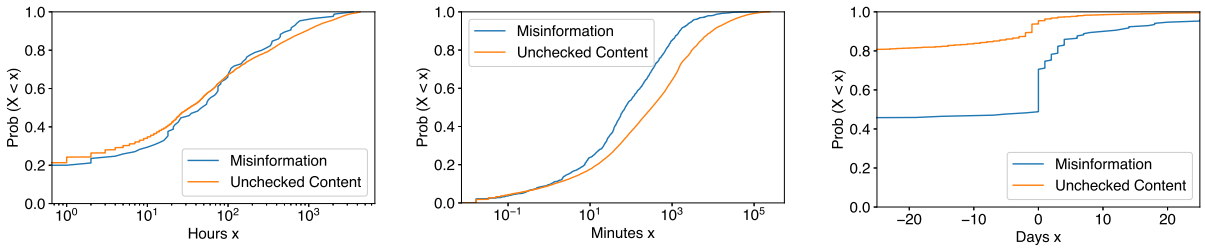
Figure 9: Images with misinformation on the Web.

and burst times within WhatsApp groups as well as the time interval between their first appearance in WhatsApp and on the Web. We here revisit this analysis comparing the same metrics for images with misinformation and images with unchecked content, focusing on the election campaign period. The results are shown in Figure 10. For each metric, we compared the two distributions using the Kolmogorov-Smirnov test [11] with 95% confidence level, with the null hypothesis that two samples have the same distribution.

We found no statistical difference between the distributions of lifetimes of images with misinformation and images with unchecked content (p-value of 0.78). For both types of content, around 70% of the images remain in the system for up to 100 hours. In contrast, the distributions of burst times are statistically different (p-value of 2e-30). Burst times tend to be shorter for images with misinformation, suggesting a faster propagation of this type of content. For example, in 60% of the cases, an image with misinformation is reshared within 100 minutes. The fraction of such burst times reduces to 40% for images with unchecked content.

Similarly, the distributions of the time interval between first appearance on WhatsApp and on the Web are also clearly different (p-value of 2.4e-47). The vast majority (95%) of images with unchecked content were first posted on the Web (negative intervals). Only 3% of them were shared first on the monitored groups (positive intervals) whereas 2% appeared on both Web and WhatsApp on the same day. In contrast, only 45% of the images with misinformation were shared first on the Web, 20% of them were shared on both platforms on the same day, and 35% were shared first on the WhatsApp group. These results seem to suggest that WhatsApp acted as a source of images with misinformation during the election campaign period.

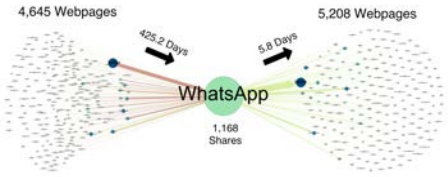
To further investigate the sharing of image content on the monitored WhatsApp groups and on the Web, we propose a visualization by means of a directed network, as shown in Figure 11. The network contains a central node representing WhatsApp (i.e., the monitored groups); the other nodes represent Web domains in which the images shared on WhatsApp also appeared. A directed edge from a node/domain to the central node implies that an image first appeared on that domain and later it was shared on WhatsApp. A directed edge from the central node to a node/domain implies the opposite. Thus, to improve readability, we plot nodes representing domains in which the images appeared before being shared on WhatsApp to the left of the central node, and nodes representing domains in which the images appeared after being shared on



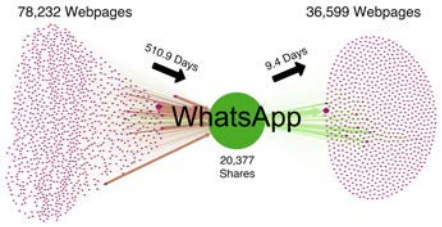
(a) Lifetime (b) Burst Time (c) Interval between first share on WhatsApp and Web

Figure 10: Temporal properties of propagation of images with misinformation versus images with unchecked content.

WhatsApp to its right. The size of a node representing a domain captures the number of webpages in that domain in which images shared on WhatsApp appeared. The color of an edge represents the average time difference between the first appearance of an image on WhatsApp and on the specific domain, considering all images posted on that domain (green is faster than red). We emphasize that this representation captures the temporal ordering of the first appearance of an image within WhatsApp and on the Web, as captured by our dataset. Although it may provide hints about the propagation of image across the boundaries between WhatsApp and the Web, we cannot claim they map exactly the actual information flow.



(a) Images with misinformation.



(b) Images with unchecked content.

Figure 11: Network representation of images shared on WhatsApp and on the Web.

Figure 11 shows the network representations for images with misinformation and images with unchecked content. In addition to the network itself, each figure shows, for each group of domains, the total numbers of pages containing shared images as well as the average time interval between the first appearance of an image on the Web and on WhatsApp. We note that images that were first published on the Web take much longer to reach the WhatsApp groups (more than a year) than the other way around (only a few days) for both types of images. The average time interval is 73 times longer for images with misinformation and 54 times longer for images with unchecked content. Also, in general, images with misinformation cross the boundaries between WhatsApp and the Web much more quickly: 425 days from the Web to WhatsApp and less than 6 days from WhatsApp to the Web, on average (as opposed to 511 and 9 days, respectively, for images with unchecked content). Moreover, the numbers of domains (and webpages) on both sides of the central node are much more balanced for images with misinformation. This suggests, once again, that images with misinformation are much more often spread from the WhatsApp groups to the rest of the Web than images with unchecked content.

6 CONCLUDING REMARKS

This work presents an analysis of messages shared on publicly accessible WhatsApp groups related to politics during two major events in Brazil. We found that images are the most popular type of media content shared on this platform during both periods analyzed. Moreover, by manually labeling these images, we found the frequent presence of satire, activism, and personal opinions, and much of this content came from other social networks. We also analyzed the temporal patterns of the propagation of these images within WhatsApp groups, finding that a large fraction of images remained being shared on the platform for quite some time (more than 4 days), and often within short time intervals (a couple of hours). We also found that most images shared on WhatsApp were actually posted first elsewhere on the Web. As a complement, we characterized the network structure of the monitored WhatsApp groups, showing how they connect with each other and offering insights into how information may propagate across them and to/from the Web.

We also proposed a methodology to automatically identify images with misinformation, and used it to investigate the sharing of this type of content in the monitored groups. We characterized the propagation dynamics of these images, contrasting it with the patterns observed for the other (unchecked) image contents. We found that images with misinformation tend to be reshared within shorter time intervals and are much more often shared first on WhatsApp and then on the Web. This observation suggests that WhatsApp may have been a relevant source of images with misinformation to the Web during the analyzed period.

As a final contribution, we also designed and deployed the **WhatsApp Monitor**[15], a Web-based system to help the top Brazilian official fact-checking agencies and journalists. Our system displays the most popular content shared in the monitored publicly

[15] http://www.whatsapp-monitor.dcc.ufmg.br

accessible groups on a daily basis. This allows journalists to get an idea about critical content that is worth being fact-checked. We emphasize that sensitive information such as user names and phone numbers are discarded and thus are not shown by our system.

To our knowledge, this is the first effort to build a system of its kind. Our system has already been used by more than a hundred journalists with an editorial line and by three fact-checking agencies which explicitly mentioned our system as a data source. We hope this system can be useful during future election campaigns and other major events in Brazil and other countries.

More broadly, we expect this study to drive follow-up investigations covering other types of content as well as delving further into the interplay between WhatsApp groups and the Web as channels for information propagation.

ACKNOWLEDGMENTS

This work was partially supported by the project FAPEMIG-PRONEX-MASWeb, Models, Algorithms and Systems for the Web, process number APQ-01400-1, as well as grants from CNPq, CAPES, and Fapemig.

REFERENCES

[1] Alessandro Bessi and Emilio Ferrara. 2016. Social bots distort the 2016 U.S. Presidential election online discussion. *First Monday* 21, 11 (2016).

[2] Josh Constine. 2018. WhatsApp hits 1.5 billion monthly users. $19B? Not so bad. https://techcrunch.com/2018/01/31/whatsapp-hits-1-5-billion-monthly-users-19b-not-so-bad/ "[Online; posted on 31-Jan-2018]".

[3] Shasta Darlington. 2018. As 'Lula' Sits in Brazil Jail, Party Nominates Him for President. *N. Y. Times* (Aug 2018). https://www.nytimes.com/2018/08/05/world/americas/lula-brazil-election-luiz-inacio-lula-da-silva.html

[4] Emilio Ferrara. 2017. Disinformation and social bot operations in the run up to the 2017 French presidential election. *First Monday* 22, 8 (2017).

[5] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2016. The Rise of Social Bots. *Commun. ACM* 59, 7 (June 2016), 96–104.

[6] Joseph L. Fleiss, Bruce Levin, and Myunghee Cho Paik. 2013. *Statistical methods for rates and proportions.* John Wiley & Sons.

[7] Adam Fourney, Miklos Z. Racz, Gireeja Ranade, Markus Mobius, and Eric Horvitz. 2017. Geographic and Temporal Trends in Fake News Consumption During the 2016 US Presidential Election. In *Proc. of the CIKM.*

[8] Kiran Garimella and Gareth Tyson. 2018. WhatsApp, Doc? A First Look at WhatsApp Public Group Data. In *Proc. of the ICWSM.*

[9] Vindu Goel, Suhasini Raj, and Priyadarshini Ravichandran. 2018. How WhatsApp Leads Mobs to Murder in India. https://www.nytimes.com/interactive/2018/07/18/technology/whatsapp-india-killings.html "[Online; posted on 18-Jul-2018]".

[10] Juliana Gragnani. 2018. Pesquisa inédita identifica grupos de família como principal vetor de notícias falsas no WhatsApp. https://www.bbc.com/portuguese/brasil-43797257 [Online; posted on 20-April-2018].

[11] Frank J. Massey Jr. 1951. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association* 46, 253 (1951), 68–78.

[12] Young Mie Kim, J. Hsu, D. Neiman, C. Kou, L. Bankston, S. Kim, R. Heinrich, R. Baragwanath, and G. Raskutti. 2018. The Stealth Media? Groups and Targets behind Divisive Issue Campaigns on Facebook. *Political Communication* 0, 0 (2018), 1–27.

[13] Mallory Locklear. 2018. Researchers say Facebook's anti-fake news efforts might be working. https://www.engadget.com/2018/09/14/facebook-fake-news-efforts-working/ [Online; posted on 14-Sep-2018].

[14] Tessa Lyons. 2018. Hard Questions: How Is Facebook's Fact-Checking Program Working? https://newsroom.fb.com/news/2018/06/hard-questions-fact-checking. "[Online; posted on 14-Jun-2018]".

[15] Matheus Magenta, Juliana Gragnani, and Felipe Souza. 2018. How WhatsApp is being abused in Brazil's elections. https://www.bbc.com/news/technology-45956557 "[Online; posted on 24-Oct-2018]".

[16] Johnnatan Messias, Lucas Schmidt, Ricardo Rabelo, and Fabrício Benevenuto. 2013. You followed my bot! Transforming robots into influential users in Twitter. *First Monday* 18, 7 (July 2013).

[17] Vishal Monga and Brian L. Evans. 2006. Perceptual image hashing via feature points: performance evaluation and tradeoffs. *IEEE Transactions on Image Processing* 15, 11 (2006), 3452–3465.

[18] Dom Phillips. 2018. Jair Bolsonaro: Brazil presidential frontrunner stabbed at campaign rally. *the Guardian* (Sep 2018). https://www.theguardian.com/world/2018/sep/06/brazil-jair-bolsonaro-far-right-presidential-candidate-stabbed

[19] Julio C. S. Reis, André Correia, Fabrício Murai, Adriano Veloso, and Fabrício Benevenuto. 2019. Supervised Learning for Fake News Detection. *IEEE Intelligent Systems* 34, 2 (2019).

[20] Gustavo Resende, Johnnatan Messias, Márcio Silva, Jussara Almeida, Marisa Vasconcelos, and Fabrício Benevenuto. 2018. A System for Monitoring Public Political Groups in WhatsApp. In *Proceedings of the 24th Brazilian Symposium on Multimedia and the Web (WebMedia '18).* ACM, New York, NY, USA, 387–390.

[21] Filipe N. Ribeiro, Koustuv Saha, Mahmoudreza Babaei, Lucas Henrique, Johnnatan Messias, Fabrício Benevenuto, Oana Goga, Krishna P. Gummadi, and Elissa M. Redmiles. 2019. On Microtargeting Socially Divisive Ads: A Case Study of Russia-Linked Ad Campaigns on Facebook. In *Proceedings of the ACM Conference on Fairness, Accountability, and Transparency (FAT*'19).* Atlanta, USA.

[22] Simon Romero. 2016. Dilma Rousseff Is Ousted as Brazil's President in Impeachment Vote. *N. Y. Times* (Aug 2016). https://www.nytimes.com/2016/09/01/world/americas/brazil-dilma-rousseff-impeached-removed-president.html

[23] Tactical Tech. 2018. WhatsApp: The Widespread Use of WhatsApp in Political Campaigning in the Global South. https://ourdataourselves.tacticaltech.org/posts/whatsapp/ "[Online; posted on 03-Jul-2018]".

[24] David Treece. 2018. Brazil's ex-president Lula imprisoned to keep him out of the election. https://www.theguardian.com/world/2018/jun/08/brazils-ex-president-lula-imprisoned-to-keep-him-out-of-the-election-letters

[25] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science* 359, 6380 (2018), 1146–1151.

The Block Point Process Model for Continuous-time Event-based Dynamic Networks

Ruthwik R. Junuthula
University of Toledo
Toledo, OH, USA
rjunuth@utoledo.edu

Maysam Haghdan*
ETH Zürich
Zürich, Switzerland
maysam.haghdan@inf.ethz.ch

Kevin S. Xu
University of Toledo
Toledo, OH, USA
kevin.xu@utoledo.edu

Vijay K. Devabhaktuni
University of Toledo
Toledo, OH, USA
Vijay.Devabhaktuni@utoledo.edu

ABSTRACT

We consider the problem of analyzing timestamped relational events between a set of entities, such as messages between users of an on-line social network. Such data are often analyzed using static or discrete-time network models, which discard a significant amount of information by aggregating events over time to form network snapshots. In this paper, we introduce a block point process model (BPPM) for continuous-time event-based dynamic networks. The BPPM is inspired by the well-known stochastic block model (SBM) for static networks. We show that networks generated by the BPPM follow an SBM in the limit of a growing number of nodes. We use this property to develop principled and efficient local search and variational inference procedures initialized by regularized spectral clustering. We fit BPPMs with exponential Hawkes processes to analyze several real network data sets, including a Facebook wall post network with over 3,500 nodes and 130,000 events.

CCS CONCEPTS

• **Mathematics of computing** → **Random graphs**; *Probabilistic representations*; Probabilistic reasoning algorithms; Stochastic processes; • **Computing methodologies** → *Learning latent representations*; Spectral methods.

KEYWORDS

block Hawkes model; event-based network; continuous-time network; timestamped network; relational events; stochastic block model; Hawkes process; asymptotic independence

ACM Reference Format:
Ruthwik R. Junuthula, Maysam Haghdan, Kevin S. Xu, and Vijay K. Devabhaktuni. 2019. The Block Point Process Model for Continuous-time Event-based Dynamic Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313633

*Research was conducted while author was at the University of Toledo.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313633

Sender	Receiver	Time
1	2	0.1
2	3	0.4
3	2	0.6
1	2	1.2
1	3	1.3
2	1	1.6

(a) Event table

$$A^{[0,1)} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$A^{[1,2)} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

(b) Adjacency matrix

Figure 1: Two representations of a continuous-time event-based dynamic network. The adjacency matrices discard the exact times and ordering of events.

1 INTRODUCTION

Many application settings involve analysis of timestamped relational event data in the form of triplets (sender, receiver, timestamp), as shown in Figure 1a. Examples include analysis of messages between users of an on-line social network, emails between employees of a company, and transactions between buyers and sellers on e-commerce websites. These types of data can be represented as dynamic networks evolving in continuous time due to the fine granularity on the timestamps of events and the irregular time intervals at which events occur.

Statistically modeling these types of relations and their dynamics over time has been of great interest, especially given the ubiquity of such data in recent years. Most prior work has involved modeling these relations using network representations, with nodes representing senders and receivers, and edges representing events. Such representations often either discard the timestamps altogether, which transforms the dynamic network into a static network, or aggregate events over time windows to form network snapshots evolving in discrete time as in Figure 1b. There have been numerous statistical models proposed for static networks dating back to the 1960s [12, 18], and more recently, for discrete-time networks [5, 15, 17, 26, 32, 39–42], but comparatively less attention has been devoted to continuous-time networks of timestamped relations.

The development of such continuous-time or point process-based network models [2, 7, 8, 11, 27, 38] appears to have progressed separately from recent advances in static and discrete-time network models. There have been many recent developments on estimation

for static network models such as the stochastic block model (SBM), including the development of consistent estimators such as regularized spectral clustering [31, 34]. Although some continuous-time models have drawn their inspiration from static network models, to the best of our knowledge, there has not been prior work connecting the two types of models, and in particular, examining whether provably accurate estimators for static network models can be used to estimate continuous-time models.

In this paper we introduce the block point process model (BPPM) for continuous-time event-based dynamic networks, inspired by the SBM for static networks. Our main contributions are as follows:

- We demonstrate an asymptotic equivalence between our proposed BPPM and the SBM in the limit of growing number of nodes, which allows us to use provably accurate and efficient estimators for the SBM, such as regularized spectral clustering, as a starting point to fit BPPMs.
- We develop efficient local search and variational inference procedures for the BPPM initialized by regularized spectral clustering on an aggregated adjacency matrix.
- We fit the BPPM to several real network data sets, including a Facebook network with over 3,500 nodes and 130,000 events and demonstrate that it is more accurate at predicting future interactions compared to discrete-time SBMs.

2 BACKGROUND

We consider dynamic networks evolving in continuous time through the observation of events between pairs of nodes at recorded timestamps, as shown in Figure 1a. We assume that events are directed, so we refer to the two nodes involved in an event as the sender and receiver (although the model we propose can be trivially modified to handle undirected events by reducing the number of parameters). Such event data can be represented in the form of a matrix E where each row is a triplet $\mathbf{e} = (u, v, t)$ denoting an event from node u to node v at timestamp t. Let N denote the total number of nodes in the network, and let T denote the time of the last interaction, so that the interaction times are all in $[0, T]$.

From an event matrix E, one can obtain an adjacency matrix $A^{[t_1,t_2)}$ over any given time interval $[t_1, t_2)$ such that $0 \leq t_1 < t_2 \leq T$. To simplify notation, we drop the time interval from the adjacency matrix, i.e. $A = A^{[t_1,t_2)}$. In this adjacency matrix, $a_{ij} = 1$ if there is at least one event from node i to node j in $[t_1, t_2)$, and $a_{ij} = 0$ otherwise. For example, Figure 1b shows two adjacency matrices constructed by aggregating events from the event table shown in Figure 1a over $[0, 1)$ and $[1, 2)$.

2.1 The Stochastic Block Model

Most statistical models for networks consider an adjacency matrix rather than event-based representation; many commonly used models of this type are discussed in the survey by Goldenberg et al. [12]. One model that has received significant attention is the *stochastic block model* (SBM), which is defined as follows (adapted from Definition 3 in Holland et al. [18]):

Definition 2.1 (Stochastic block model). Let A denote a random adjacency matrix for a static network, and let $\mathbf{c}$ denote a class membership vector. A is generated according to a stochastic block model with respect to the membership vector $\mathbf{c}$ if and only if,

(1) For any nodes $i \neq j$, the random variables a_{ij} are statistically independent.
(2) For any nodes $i \neq j$ and $i' \neq j'$, if i and i' are in the same class, i.e. $c_i = c_{i'}$, and j and j' are in the same class, i.e. $c_j = c_{j'}$, then a_{ij} and $a_{i'j'}$ are identically distributed.

The classes in the SBM are also commonly referred to in the literature as blocks. The class membership vector $\mathbf{c}$ has N entries where each entry $c_i \in \{1, \ldots, K\}$ denotes the class membership of node i, and K denotes the total number of classes. Recent work has focused on estimating the class memberships from the adjacency matrix A. In this setting, spectral clustering (and regularized variants) has emerged as an efficient estimator that has theoretical accuracy guarantees [22, 31, 33–35], scales to large networks with thousands of nodes, and is generally not sensitive to initialization.

2.2 Related Work

Most existing work on modeling dynamic networks has considered a discrete-time representation, where the observations consist of a sequence of adjacency matrices. This observation model is ideally suited for network data collected at regular time intervals, e.g. weekly surveys. In practice, however, dynamic network data is often collected at much finer levels of temporal resolution (e.g. at the level of a second or millisecond), in which case it likely makes more sense to treat time as continuous rather than discrete. In order to apply discrete-time dynamic network models to such data, it must first be pre-processed by aggregating events over time windows to form network snapshots, and this technique is used in many real data experiments [15, 27, 40–42]. For example, an aggregated representation of the network in Figure 1a with time window of 1 is shown in Figure 1b.

Aggregating continuous-time network data into discrete-time snapshots presents several challenges. One would ideally choose the time window to be as short as possible for the maximum temporal resolution. However, this increases the number of snapshots, and accordingly, the computation time (typically linear in the number of snapshots). More importantly, models fit using shorter time windows can lead to worse predictors than models fit using longer time windows because the models often assume short-term memory, such as the Markovian dynamics in many discrete-time SBMs [17, 26, 39–42]. We demonstrate some of these practical challenges in an experiment in Section 6.2.2.

Another line of research that has evolved independently of discrete-time network models involves the use of point processes to estimate the structure of a latent network from observations at the nodes [10, 13, 16, 24, 36]. These models are often used to estimate networks of diffusion from information cascades. Such work differs from the setting we consider in this paper, where we directly observe events *between pairs of nodes* and seek to model the dynamics of such event sequences.

There have been several other models proposed using point processes to model continuous-time event-based networks [2, 7, 8, 11, 23, 27, 38], which is the setting we consider in this paper. These models are typically fit using Markov chain Monte Carlo (MCMC) methods, which do not scale to large networks with thousands of nodes. The BPPM that we propose in this paper is a simpler version of the Hawkes IRM [2]. The relational event model (REM) [7] is

Algorithm 1 Generative process for BPPM

1: **for** node $i = 1$ **to** N **do**
2: Sample class c_i from categorical distribution with parameter vector $\boldsymbol{\pi}$
3: **for** block pair $b = 1$ **to** p **do**
4: **loop**
5: Sample next event time t_b from bth point process
6: **if** $t_b > T$ **then**
7: **break**
8: Randomly select nodes $i \in b_1, j \in b_2$ to form an edge from node i to node j at time t_b

related to the BPPM in that it is also inspired by the SBM and shares parameters across nodes in the network in a similar manner. We discuss the Hawkes IRM and REM in greater detail and compare them to our proposed model in Section 3.4.

3 THE BLOCK POINT PROCESS MODEL

3.1 Model Specification

We propose to model continuous-time dynamic networks using a generative point process network model. Motivated by the SBM for static networks, we propose to divide nodes into K classes or blocks and to associate a univariate point process with each pair of node blocks $b = (b_1, b_2) \in \{1, \ldots, K\}^2$, which we refer to as a *block pair*. Let $p = K^2$ denote the total number of block pairs. Let $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_K\}$ denote the class membership probability vector, where π_q denotes the probability that a node belongs to class q. We call our model the *block point process model* (BPPM). The generative process for the BPPM for a network of duration T time units is shown in Algorithm 1.

The BPPM is a very general model—notice that we have not specified what type of point process to use in the model (we discuss this in Section 3.3). The proposed BPPM is less flexible than existing point process network models such as the Hawkes IRM and the REM (we compare the BPPM to these models in Section 3.4), but its simplicity enables theoretical analysis of the model. We then use the findings of our analysis to develop principled and efficient inference procedures that scale to large networks with thousands of nodes and hundreds of thousands of events. The proposed inference procedures, which we discuss in Section 4, take advantage of the close relationship between the BPPM and the SBM, which we discuss next.

3.2 Asymptotic Equivalence with the Stochastic Block Model

The BPPM is motivated by the SBM, where the probability of forming an edge between two nodes depends only the classes of the two nodes. Given the relation between the point process and adjacency matrix representations discussed in Section 2, a natural question is whether there is any equivalence between the BPPM and the SBM. Specifically, does an adjacency matrix $A = A^{[t_1, t_2)}$ constructed from an event matrix E generated by the BPPM follow an SBM? As far as we know, this connection between point process and static network models has not been previously explored in the literature.

We first note that A meets criterion 2 (identical distribution within a block pair) in Definition 2.1 due to the random selection of node pair for each event in step 8 of Algorithm 1. To check criterion 1 (independence of all entries of A), we first note that entries a_{ij} and $a_{i'j'}$ in different block pairs, i.e. $(c_i, c_j) \neq (c_{i'}, c'_j)$, depend on different independent point processes (unlike in the Hawkes IRM), so a_{ij} and $a_{i'j'}$ are independent.

Next, consider entries a_{ij} and $a_{i'j'}$ in the same block pair $b = (c_i, c_j) = (c_{i'}, c_{j'})$. In general, these entries are dependent so that criterion 1 is not satisfied.[1] For example, if a Hawkes process [21] is used in step 5 of Algorithm 1, then $a_{i'j'} = 1$ indicates that at least one event was generated in block pair b, i.e. there was at least one jump in the intensity of the process. This indicates that the probability of another event is now higher, so the conditional probability $\Pr(a_{ij} = 1 | a_{i'j'} = 1)$ should be higher than the marginal probability $\Pr(a_{ij} = 1)$. Thus a_{ij} and $a_{i'j'}$ are *dependent*, so A does *not follow* an SBM!

We denote the *deviation from independence* using the terms δ_0 and δ_1 defined by

$$\delta_0 = \Pr(a_{ij} = 0 | a_{i'j'} = 0) - \Pr(a_{ij} = 0) \tag{1}$$

$$\delta_1 = \Pr(a_{ij} = 0 | a_{i'j'} = 1) - \Pr(a_{ij} = 0). \tag{2}$$

If $\delta_0 = \delta_1 = 0$, then the two adjacency matrix entries are independent. If $\delta_0 \neq 0$ or $\delta_1 \neq 0$, then the two entries are dependent, with smaller values of $|\delta_0|, |\delta_1|$ indicating less dependence. The following theorem bounds these values.

Theorem (Asymptotic Independence Theorem)**.** *Consider an adjacency matrix A constructed from the BPPM over some time interval $[t_1, t_2)$. Then, for any two entries a_{ij} and $a_{i'j'}$ both in block pair b, the deviation from independence given by δ_0, δ_1 defined in* (1), (2) *is bounded in the following manner:*

$$|\delta_0|, |\delta_1| \leq \min\{1, \mu_b / n_b\} \tag{3}$$

where μ_b denotes the expected number of events in block pair b in $[t_1, t_2)$, and n_b denotes the size of block pair b. In the limit as $n_b \to \infty$, $\delta_0, \delta_1 \to 0$ provided μ_b grows at a slower rate than n_b. Thus a_{ij} and $a_{i'j'}$ are asymptotically independent for growing n_b.

The proof of the Asymptotic Independence Theorem is provided in Appendix A. We evaluate the tightness of the bound in (3) via simulation in Section 5.1. Since it depends only on the expected number of events μ_b and not the distribution, it is likely to be loose in general but applies to any choice of point process.

The Asymptotic Independence Theorem states that the deviation given by δ_0, δ_1 is non-zero in general for fixed n_b, so the entries a_{ij} and $a_{i'j'}$ are dependent, but the dependence decreases as the size of a block (and thus, a block pair) grows. This can be achieved by letting the number of nodes N in the network grow while holding the number of classes K fixed. In this case, the sizes of block pairs would be growing at rate $O(N^2)$, so the asymptotic behavior should be visible for networks with thousands of nodes. Thus, an adjacency matrix constructed from the BPPM approaches an SBM in the limit of a growing network! To the best of our knowledge, this is the first such result linking networks constructed from point process models and static network models. It is also practically useful in

[1] An exception is the case of a homogeneous Poisson process, for which the entries are independent by the splitting property.

that it allows us to leverage recent work on provably accurate and efficient inference on the SBM for the BPPM.

3.3 Choice of Point Process Model

Any temporal point process can be used to generate the event times in the BPPM. We turn our attention to a specific point process: the Hawkes process [21], which is a self-exciting process where the occurrence of events increases the probability of additional events in the future. The self-exciting property tends to create clusters of events in time, which are empirically observed in many settings. Prior work has suggested that Hawkes processes with exponential kernels provide a good fit to many real social network data sets, including email and conversation sequences [14, 25] and re-shares of posts on Twitter [43]. Hence, we also adopt the exponential kernel, which has intensity function

$$\lambda(t) = \lambda^{\infty} + \sum_{t_i < t} \alpha e^{-\beta(t - t_i)},$$

where λ^{∞} denotes the background rate that the intensity reverts to over time, α denotes the jump size for the intensity function, β denotes the exponential decay rate, and the t_i's denote times of events that occurred prior to time t. We refer to this model as the *block Hawkes model* (BHM).

3.4 Relation to Other Models

The block Hawkes model we consider is a simpler version of the Hawkes IRM, which couples a non-parametric Bayesian version of the SBM called the infinite relational model (IRM) [20] with mutually-exciting Hawkes processes. By utilizing mutually-exciting Hawkes processes, the Hawkes IRM allows for reciprocity of events between block pairs. Similar to the BHM, node pairs in a block pair are selected at random to form an edge. The authors use MCMC-based inference that scales only to very small networks.

The BHM simplifies the Hawkes IRM by using a fixed number of classes K and univariate rather than multivariate Hawkes processes. The use of univariate Hawkes processes is crucial because it allows for independence between block pairs, which we used in the analysis in Section 3.2 to demonstrate an asymptotic equivalence with an SBM. We use this asymptotic equivalence in Section 4 to devise an efficient inference procedure that scales to networks with thousands of nodes.

The BHM also has similarities with the relational event model (REM) [7], which associates a non-homogeneous Poisson process with each pair of nodes, where the intensity function is piecewise constant with knots (change points) at the event times. Different node pairs belonging to the same block pair are governed by the same set of parameters. The REM also incorporates other edge formation mechanisms within block pairs such as reciprocity and transitivity, similar to an exponential random graph model. The authors also use MCMC for inference.

4 INFERENCE PROCEDURE

The observed data is in the form of triplets $\mathbf{e}_s = (u_s, v_s, t_s)$ for each event s denoting the nodes u_s, v_s involved and the timestamp t_s. Consider an event matrix E where each row corresponds to an event in the form of a triplet $\mathbf{e}_s$. Fitting the BPPM involves estimating both the unknown classes or blocks for each node and the point process parameters θ_b for each block pair b from E. In the case of an exponential Hawkes process, the parameters are given by

$$\theta_b = \left(\alpha_b, \beta_b, \lambda_b^{\infty}\right)$$

for each block pair b. Let $\theta = \{\theta_b\}_{b=1}^{p}$ denote the set of point process parameters over all p block pairs.

Exact inference is impractical for all but the smallest networks due to the discrete class memberships $\mathbf{c}$. Thus, we consider two approximate inference methods: a greedy local search (Section 4.1) and variational inference (Section 4.2). Both approaches are iterative and converge to a local maximum and are thus sensitive to the choice of initialization, which we discuss in Section 4.3.

4.1 Local Search

Consider a conditional likelihood function for the point process parameters θ given the values of the class memberships $\mathbf{c}$ that determine the block pairs. Let $E^{(b)}$ denote rows of E corresponding to events involving block pair $b = (b_1, b_2)$; that is, rows $\mathbf{e}_s$ where $u_s \in b_1$ and $v_s \in b_2$. The p row blocks $E^{(b)} = [\mathbf{u}^{(b)}, \mathbf{v}^{(b)}, \mathbf{t}^{(b)}]$ form a partition of the rows of matrix E. Let m_b denote the number of events observed in block pair b. Let n_b denote the size of block pair b, i.e. the number of node pairs or possible edges in block pair b, which is given by $|b_1|(|b_1| - 1)$ if $b_1 = b_2$ and $|b_1||b_2|$ otherwise. The conditional log-likelihood function is given by

$$\begin{aligned}
\log \Pr(E|\theta, \mathbf{c}) &= \log \prod_{b=1}^{p} \Pr\left(\mathbf{u}^{(b)}, \mathbf{v}^{(b)}, \mathbf{t}^{(b)} \middle| \theta_b, \mathbf{c}\right) \\
&= \log \prod_{b=1}^{p} \Pr\left(\mathbf{t}^{(b)} \middle| \theta_b\right) \prod_{s \in b} \Pr(u_s, v_s | \mathbf{c}) \\
&= \log \prod_{b=1}^{p} \Pr\left(\mathbf{t}^{(b)} \middle| \theta_b\right) \left(\frac{1}{n_b}\right)^{m_b} \\
&= \sum_{b=1}^{p} \left[\log \Pr\left(\mathbf{t}^{(b)} \middle| \theta_b\right) - m_b \log n_b\right] \qquad (4)
\end{aligned}$$

where the expression for $\Pr(u_s, v_s|\mathbf{c})$ follows from the random selection of nodes in step 8 of the BPPM generative process. The term $\log \Pr\left(\mathbf{t}^{(b)}|\theta_b\right)$ is simply the log-likelihood of the point process model parameters given the timestamps of events in block pair b. For the block Hawkes model, this term can be expressed in the following form [21, 30]:

$$\begin{aligned}
\log \Pr\left(t_{(1)}, \ldots, t_{(m)} | \alpha_b, \beta_b, \lambda_b^{\infty}\right) = \sum_{s=1}^{m} \Bigg\{ & \frac{\alpha_b}{\beta_b}\left[e^{-\beta_b\left(t_{(m)} - t_{(s)}\right)} - 1\right] \\
& + \log\left[\lambda_b^{\infty} + \alpha_b \sum_{r=1}^{s-1} e^{-\beta_b\left(t_{(s)} - t_{(r)}\right)}\right] \Bigg\} - \lambda_b^{\infty} t_{(m)}, \qquad (5)
\end{aligned}$$

where $t_{(s)}$ denotes the sth event corresponding to block pair b. (5) can be written in a recursive form as shown in [29].

The conditional log-likelihood (4) requires knowledge of class memberships $\mathbf{c}$, which are used to partition the event matrix into row blocks $E^{(b)}$ and thus affect both terms in (4) through $\mathbf{t}^{(b)}$, m_b, and n_b. However, in practice, class memberships are unknown, so we must maximize (4) over all possible class assignments.

We use a local search (hill climbing) procedure, which is also often referred to as label switching or node swapping in the network science literature [19, 44] to iteratively update the class assignments to reach a local maximum in a greedy fashion. Recent work has found that such greedy algorithms are competitive with more computationally demanding estimation algorithms in both the static SBM [4] and discrete-time dynamic SBM [5, 41] while scaling to much larger networks. At each iteration, we swap a single node to a different class by choosing the swap that increases the log-likelihood the most. For each possible swap, we evaluate the log-likelihood by partitioning events using to the new class assignments, obtaining the maximum-likelihood estimates of the point process model parameters, and substituting these estimates along with the new class assignments into (4). For the block Hawkes model, we maximize (5) with respect to $(\alpha_b, \beta_b, \lambda_b^\infty)$ for each block using a standard interior point optimization routine [3].

Each iteration of the local search considers $N(K-1)$ possible swaps. Computing the log-likelihood for each swap involves iterating over the timestamps of all M events. Thus, each iteration of the local search has time complexity $O(KMN)$, which is linear in both the number of nodes and events, allowing it to scale to large networks. We verify this time complexity experimentally in Section 5.3. The local search is easily parallelized by evaluating each possible swap on a separate CPU core. We terminate the local search procedure when no swap is able to increase the log-likelihood, indicating that we have reached a local maximum.

4.2 Variational Inference

Variational inference is commonly used as an optimization-based alternative to MCMC and scales to much larger data sets. We implement a mean-field variational inference approach for the BPPM to approximate the intractable posterior distribution by a fully factorizable variational distribution. To reduce the Kullback-Leibler (KL) divergence between the true posterior and the mean-field approximation, we derive the evidence lower bound (ELBO) using an approach similar to the derivation in [6] for a static SBM.

Let Z denote an $N \times K$ class membership matrix, where the notation $z_{iq} = 1$ is equivalent to $c_i = q$, both denoting that node i is in class q. We use a K-dimensional multinomial distribution for each row $\mathbf{z}_i = [z_{i1}, z_{i2}, \ldots, z_{iK}]$ of Z resulting in the following variational distribution:

$$R_E(Z) = \prod_{i=1}^{N} \text{Multinomial}(\mathbf{z}_i | \boldsymbol{\tau}_i), \tag{6}$$

where $\boldsymbol{\tau}_i$ denotes the variational parameter for node i. Unlike for a static SBM, we don't have a closed-form update equation for the block Hawkes model, so we optimize the ELBO using coordinate ascent. The derivation of the ELBO and the variational expectation-maximization algorithm are provided in Appendix B.

4.3 Spectral Clustering Initialization

In order to ensure that the local search or variational inference procedures do not get stuck in poor local maxima, it is important to provide a good initialization. Methods used to initialize class estimates in static and discrete-time SBMs include variants of k-means clustering [4, 26, 27] and spectral clustering [40, 41]. Variational inference is often executed with multiple random initializations, although some structured approaches have been used successfully in practice for certain models. Given the close relationship between the proposed BPPM and the SBM discussed in Section 3.2, we use a spectral clustering initialization, which is much faster and more principled than the typical approach of multiple random initializations. Spectral clustering is an attractive choice because it scales to large networks containing thousands of nodes and has theoretical performance guarantees applicable to the BPPM, as we discuss next.

Algorithm 2 Regularized spectral clustering algorithm used to initialize the local search in the BPPM inference procedure

Require: Adjacency matrix A, number of classes K, regularization parameter $\tau \geq 0$ (Default: τ = average node degree)

1: Compute diagonal matrices O^τ with entries $o_{ii}^\tau = \sum_j a_{ij} + \tau$ and P^τ with entries $p_{jj}^\tau = \sum_i a_{ij} + \tau$
2: Compute regularized graph Laplacian $L = (O^\tau)^{-1/2} A (P^\tau)^{-1/2}$
3: Compute singular value decomposition of L
4: $\tilde{\Sigma} \leftarrow$ diagonal matrix of K largest singular values of L
5: $(\tilde{U}, \tilde{V}) \leftarrow$ left and right singular vectors for $\tilde{\Sigma}$
6: $\tilde{Z} \leftarrow [\tilde{U}, \tilde{V}]$ {concatenate left and right singular vectors}
7: Normalize each row of $\tilde{Z}$ to have magnitude of 1
8: $\hat{\mathbf{c}} \leftarrow$ k-means clustering on rows of $\tilde{Z}$
9: **return** $\hat{\mathbf{c}}$

Recent work has demonstrated that applying spectral clustering (or a regularized variant) to a network generated from an SBM results in consistent estimates of class assignments as the number of nodes $N \to \infty$ [22, 31, 33–35]. These theoretical guarantees typically require the expected degrees of nodes to grow polylogarithmically with the number of nodes so that the network is not too sparse. Networks that satisfy this requirement belong to the polylog degree regime. On the other hand, the Asymptotic Independence Theorem shows an asymptotic equivalence between the BPPM and SBM provided that there are not too many events, i.e. the network is not too dense.

In the polylog degree regime, the ratio

$$\frac{\mu_b}{n_b} = O\left(\frac{N \,\text{poly}(\log N)}{N^2}\right) \to 0 \text{ as } N \to \infty,$$

so the network is not too dense, and the Asymptotic Independence Theorem holds. Thus, spectral clustering should provide an accurate estimate of the class assignments in the polylog degree regime, which is commonly observed in real networks such as social networks. Since we consider directed relations, we use a regularized spectral clustering algorithm for directed networks (pseudocode provided in Algorithm 2) to initialize the local search. It is a variant of the DI-SIM co-clustering algorithm [34] modified to produce a single set of clusters for directed graphs by concatenating scaled left and right singular vectors in a manner similar to Sussman et al. [35]. For variational inference, we require an initialization on the variational parameters $\boldsymbol{\tau}_i$ rather than a hard clustering solution, so we set $\boldsymbol{\tau}_i$ to the ith row of $\tilde{Z}$ to represent a soft clustering initialization.

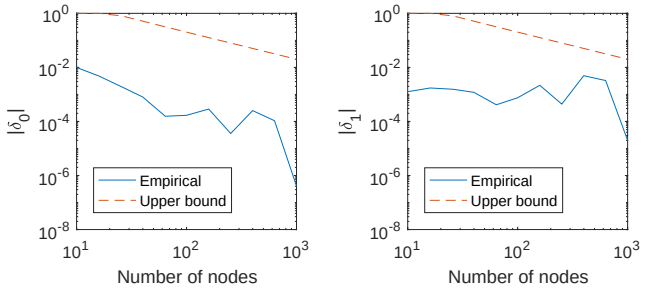


Figure 2: Comparison of empirical deviation from independence with theoretical upper bound. Empirical deviations are well within the theoretical bound.

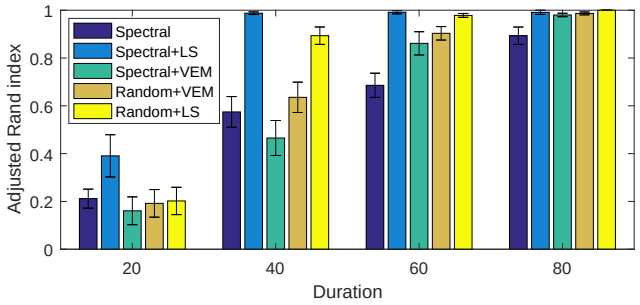


Figure 3: Mean adjusted Rand indices (± standard error) for class estimation simulation experiment with varying durations. Spectral clustering with local search produces the most accurate class estimates, while the spectral initialization for variational EM does not appear to work well.

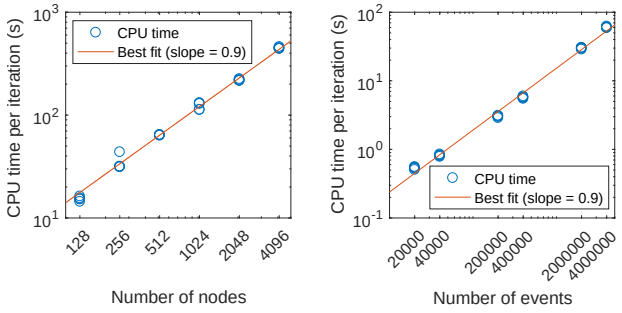


Figure 4: CPU time per iteration of local search inference in seconds with varying number of nodes and events.

5 SIMULATED DATA EXPERIMENTS

5.1 Deviation from Independence

The Asymptotic Independence Theorem demonstrates that pairs of adjacency matrix entries in the same block pair are dependent, but that the dependence is upper bounded by (3), and that the dependence goes to 0 for growing blocks. To evaluate the slackness of the bounds, we simulate networks from the block Hawkes model (BHM). Since δ_0 and δ_1 depend only on the size of the blocks, we simulate networks with a single block and let the number of nodes N grow from 10 to 1,000. For each number of nodes, we simulate 100,000 networks from the block Hawkes model for a duration of $T = 20$ time units. We choose the Hawkes process parameters to be $\alpha = 5N$, $\beta = 10N$, and $\lambda^{\infty} = 0.5N$. The expected number of events $\mu = NT = 20N$, which grows with N and is slower than the growth of the size $n = N(N-1)$ of the block pair, so the Asymptotic Independence Theorem applies.

We evaluate the absolute difference between the empirical marginal probability $\widehat{\Pr}(a_{ij} = 0)$ and the empirical conditional probabilities $\widehat{\Pr}(a_{ij} = 0|a_{i'j'} = 0)$ and $\widehat{\Pr}(a_{ij} = 0|a_{i'j'} = 1)$. The empirical deviation from independence is shown to be well below the upper bound in Figure 2. The bound (3) in the Asymptotic Independence Theorem depends only on the mean number of events, so it is somewhat loose when applied to the block Hawkes model.

5.2 Class Estimation

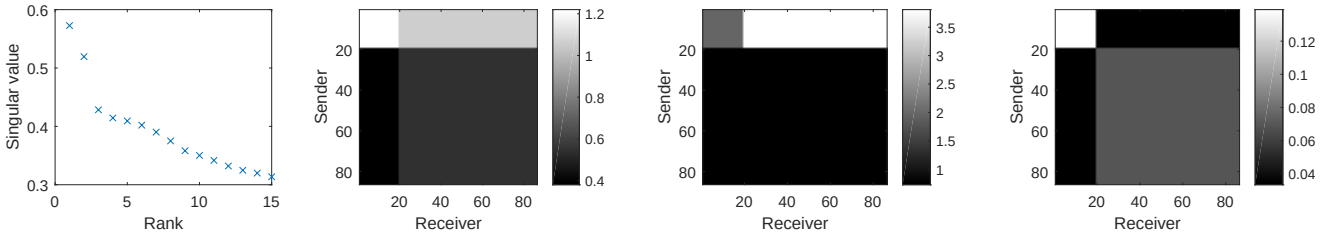
This simulation experiment is based on the synthetic network generator from Newman and Girvan [28], where all the diagonal block pairs have the same parameters, and the off-diagonal block pairs have the same parameters, but different from the diagonal block pairs. We generate networks with 128 nodes and 4 classes from the block Hawkes model using Algorithm 1 with varying durations from 20 to 80 time units. We generate 10 networks for each duration, with Hawkes process parameters α and β being 0.6 and 0.8, respectively, for all block pairs. The baseline rates λ^{∞} are 1.8 for diagonal block pairs and 0.6 for off-diagonal block pairs. Classes are estimated using 5 methods: spectral clustering, local search initialized with spectral clustering, variational EM initialized with spectral clustering, variational EM with 10 random initializations, and local search with 10 random initializations.

The results shown in Figure 3 demonstrate that local search initialized with spectral clustering (Spectral+LS) is the most accurate. Conversely, the spectral clustering initialization for variational EM (Spectral+VEM) results in significantly worse results. Variational EM and local search using 10 random initializations (Random+VEM and Random+LS) take significantly longer than Spectral+LS; however, they are also less accurate, so we use Spectral+LS in the remainder of this paper.

5.3 Scalability of Local Search

We evaluate the scalability of the proposed local search inference procedure by generating networks with varying number of nodes and events from the block Hawkes model. When varying the number of nodes, we choose a time duration of 1,200 time units and set the Hawkes process parameters $(\alpha, \beta, \lambda^{\infty})$ to $(1.6, 2, 1.2)$ for diagonal block pairs and $(0.6, 0.8, 0.6)$ for off-diagonal block pairs. When varying the number of events, we choose a network with 128 nodes and set the Hawkes process parameters $(\alpha, \beta, \lambda^{\infty})$ to $(0.6, 0.8, 1.8)$ for diagonal block pairs and $(0.6, 0.8, 0.6)$ for off-diagonal block pairs. We keep the number of classes fixed to be 4 in both settings and simulate 5 networks for each configuration.

The CPU times per iteration for both varying number of nodes and events are shown in Figure 4 along with best-fit lines for a



(a) Singular values (b) α: Jump sizes (c) β: Jump decay rates (d) λ^{∞}: Background intensity

Figure 5: Block Hawkes model fit on Reality Mining data with 2 blocks. Nodes 1-15 belong to block 1 and the rest to block 2. (a) First 15 singular values of the adjacency matrix. (b)-(d) Hawkes process parameter estimates.

power law relationship (beginning with 512 nodes for the varying nodes experiment). The best-fit line has slope 0.9 in both cases, confirming the linear time complexity both in terms of the number of nodes N and the number of events M, which is as expected according to the computed time complexity in Section 4.1. All CPU times are recorded on a Linux workstation using 18 Intel Xeon processor cores operating at 2.8 GHz.

6 REAL DATA EXPERIMENTS

6.1 MIT Reality Mining

We analyze the MIT Reality Mining data [9], using start times of phone calls as events from the caller to the recipient. Nodes in this data set correspond to students and staff at MIT. The data were collected over approximately 10 months beginning in August 2004. We remove all nodes who do not either make or receive a call at any point in the data trace, resulting in a network with 86 nodes.

We observe a significant gap after the second largest singular value of the regularized graph Laplacian, as shown in Figure 5a so we choose $K = 2$ blocks. From examining the block Hawkes model parameter estimates shown in Figure 5b-5d, block pair $(1, 1)$ has both higher background intensity λ^{∞} and longer bursts due to high α and low β. Thus, not only does it appear to be a community, but phone calls within the community tend to happen in prolonged bursts. On the other hand, block pair $(2, 2)$ has higher background intensity than the off-diagonal block pairs but without the large jump sizes, indicating a lack of bursty behavior compared to block pair $(1, 1)$. Block pair $(1, 2)$ has high values for both α and β, indicating large bursts of short duration. However, since block pair $(1, 2)$ has a lower background intensity than $(2, 2)$, the overall density of the block pair in the aggregated adjacency matrix is not necessarily higher. Indeed, block pair $(2, 2)$ has a density of 0.041, while block pair $(2, 1)$ has a density of 0.034. Thus, the continuous-time model of the network enables greater analysis of the dynamics of interactions over both short and long periods of time, and the findings may be quite different from those of static and discrete-time network representations.

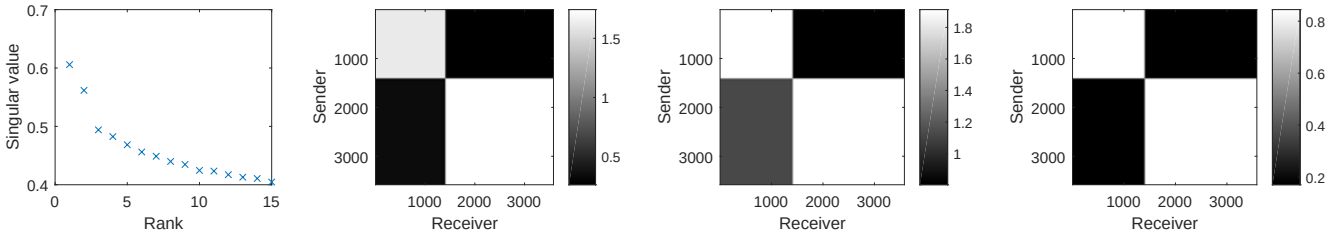
6.2 Facebook Wall Posts

6.2.1 Model-Based Exploratory Analysis. We analyze the Facebook wall post data collected by Viswanath et al. [37], which contains over $60,000$ nodes. We consider events between January 1, 2007 and January 1, 2008. We remove nodes with degree less than 10, resulting in a network with $137,170$ events among $3,582$ nodes.

We select a model with $K = 2$ blocks, as suggested by the singular values of the regularized graph Laplacian shown in Figure 6a. The parameters inferred from the BHM fit on the Facebook data are shown in Figure 6b-6d. The diagonal block pairs have larger values of background intensity λ^{∞}, indicating that the blocks form communities. This finding could also have been yielded by static and discrete-time SBMs. The diagonal block pairs also have higher values of jump sizes α, indicating that wall posts between members of a community are more bursty. A portion of the Hawkes process intensity function for block pair $(1, 1)$ is shown in Figure 7. In addition to diurnal patterns, one can observe bursty periods of wall posts throughout the day. This finding could not have been obtained from static and discrete-time SBMs. From the values of α on the diagonal block pairs, we see that wall posts within block 2 are more bursty, with higher jump sizes and roughly equal jump decay rates compared to posts within block 1. By observing the values of α on off-diagonal block pairs, we notice that there isn't much asymmetry, but the decay rate β exhibits asymmetry. Specifically, events from block 1 to 2 have longer sustained bursts than events from block 2 to 1 due to the lower value of β_{12} compared to β_{21}.

6.2.2 Comparison with Discrete-Time SBM. To compare our proposed continuous-time BHM with a discrete-time SBM [41], we consider the task of predicting the time to the next event in each block pair. We believe that this is a fair comparison because both the BHM and discrete-time SBM require that pairs of nodes in the same block pair have identical edge probabilities. We split the data into two sets, with the first 8 months of data for training and the last 4 months for testing. During each week of the test set, we attempt to predict the time to the next event in each block pair. Afterwards, we update the model with all events during that week. This results in 16 predictions in the test set for each block pair.

The BHM directly models event times, so we use the expected next event time for each block as the prediction. The discrete-time SBM does not directly model event times, so we multiply the expected number of time snapshots that will elapse before the next edge formation, which is geometrically distributed, by the snapshot length and then subtract half the snapshot length (to center the prediction within the snapshot) to get a prediction for the next event time. Since the prediction for the discrete-time SBM is dependent on the snapshot length, we test several different snapshot lengths.



(a) Singular values (b) α: Jump sizes (c) β: Jump decay rates (d) λ^{∞}: Background intensity

Figure 6: Block Hawkes model parameter estimates on Facebook wall post data with 2 blocks. Nodes 1-1421 belong to block 1 and the rest to block 2. (a) First 15 singular values of the adjacency matrix. (b)-(d) Hawkes process parameter estimates.

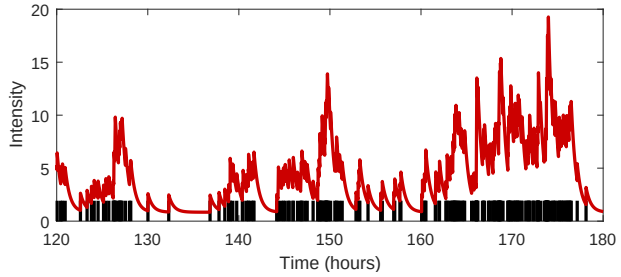


Figure 7: Hawkes process intensity function for events in block pair (1,1) of the Facebook data. Event times are denoted by black ticks along the horizontal axis.

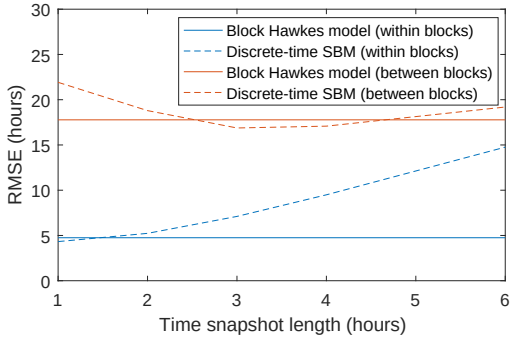


Figure 8: Prediction RMSE in hours for block Hawkes model and discrete-time SBMs on Facebook data.

We estimate class assignments for both models using regularized spectral clustering with 2 classes (with no local search iterations) so that differences in class estimates between the two models do not play a role in the accuracy. We believe this is a valid comparison because spectral clustering is used as the initialization to local search in the inference procedure for both models, as discussed in Section 4.1 for the BHM and in [41] for the discrete-time SBM.

We evaluate the accuracy of the predictions by computing the root mean-squared error (RMSE) between predicted event times and actual event times for the first event in each block pair during a week. Since the blocks form communities, we expect events to arrive much more frequently within blocks. Thus we separate the evaluation into within-block and between-block prediction RMSE. As shown in Figure 8, the accuracy of the discrete-time SBM is highly dependent on the snapshot length. For snapshots of 6 hours and longer, the loss in temporal resolution is the main contributor to the high RMSE. The shorter snapshots such as 1 hour and 2 hours have excellent temporal resolution for within-block prediction but are less accurate for between-block prediction. Choosing 3-hour long snapshots results in the most accurate between-block predictions. Due to the different rates of events within and between blocks, the discrete-time representation must trade off between within-block and between-block prediction accuracy when choosing the snapshot length. Using our continuous-time BHM, we avoid this complex problem of choosing the snapshot length and produce more accurate predictions (in total RMSE) than the discrete-time model for any snapshot length, as shown in Figure 8.

7 CONCLUSION

In this paper, we introduced the block point process model (BPPM) for dynamic networks evolving in continuous time in the form of timestamped events between nodes. Our model was inspired by the well-known stochastic block model (SBM) for static networks and is a simpler version of the Hawkes IRM. We demonstrated that adjacency matrices constructed from the BPPM follow an SBM in the limit of a growing number of nodes. To the best of our knowledge, this is the first result of this type connecting point process network models with adjacency matrix network models. Additionally we proposed a principled and efficient algorithm to fit the BPPM using spectral clustering and local search that scales to large networks and apply it to analyze several real networks.

A PROOF OF ASYMPTOTIC INDEPENDENCE THEOREM

We begin with a well-known lemma on the difference of powers that will be used both to upper and lower bound the deviation from independence.

Lemma A.1 (Difference of powers). *For a real number $x > 1$ and integer $m \geq 1$, we have the following identity:*

$$m(x-1)^{m-1} \leq x^m - (x-1)^m \leq mx^{m-1}. \tag{7}$$

Proof. The proof follows straightforwardly from factorizing a difference of powers. Specifically, for real numbers $x > y > 0$ and

integer $m \geq 1$,

$$x^m - y^m = (x-y)\sum_{i=0}^{m-1} x^{m-1-i}y^i. \tag{8}$$

If $x > 1$ and $y = x - 1$, then (8) becomes

$$x^m - (x-1)^m = \sum_{i=0}^{m-1} x^{m-1-i}(x-1)^i.$$

There are m terms in the summation. The largest term is x^{m-1}, and the smallest term is $(x-1)^{m-1}$. Thus, we can upper and lower bound the sum by mx^{m-1} and $m(x-1)^{m-1}$, respectively, to arrive at (7). □

The next lemma will be used in the upper bound.

Lemma A.2. *For a real number $x > 1$ and integer $m \geq 1$,*

$$\left(\frac{n-1}{n}\right)^m \geq 1 - \frac{m}{n}.$$

Proof.

$$\begin{aligned}
\left(\frac{n-1}{n}\right)^m - 1 &= \frac{(n-1)^m - n^m}{n^m} \\
&= -\frac{n^m - (n-1)^m}{n^m} \\
&\geq -\frac{mn^{m-1}}{n^m} \qquad (9) \\
&= -\frac{m}{n}, \qquad (10)
\end{aligned}$$

where (9) follows from applying Lemma A.1. Adding 1 to both sides of (10), we arrive at the desired result. □

We now prove the Asymptotic Independence Theorem.

Proof of Asymptotic Independence Theorem. First compute the marginal probability $\Pr(a_{ij} = 0)$. $a_{ij} = 0$ implies that no events between nodes i and j occurred. To compute this probability, we first compute the conditional probability given that the number of events in block pair b is m_b. To simplify notation, we drop the subscript b from m_b and n_b in the remainder of the proof, so the conditional probability can be written as

$$\Pr(a_{ij} = 0|m) = \left(\frac{n-1}{n}\right)^m,\ m \geq 0,$$

where the equality follows by noting that, conditioned on m total events in block pair b, the number of events between nodes i and j follows a binomial distribution with m trials and success probability $1/n$. The $1/n$ success probability is due to step 8 of the generative process of the BPPM, which involves selecting node pairs randomly to receive an event. By the Law of Total Probability, the marginal probability is

$$\begin{aligned}
\Pr(a_{ij} = 0) &= \sum_{m=0}^{\infty} p(m)\Pr(a_{ij} = 0|m) \\
&= \sum_{m=0}^{\infty} p(m)\left(\frac{n-1}{n}\right)^m, \qquad (11)
\end{aligned}$$

where the probability mass function $p(m)$ denotes the probability that m events in block pair b occurred.

Next consider the joint probability $\Pr(a_{ij} = 0, a_{i'j'} = 0)$. As before, condition on the number of events m. The conditional joint probability is

$$\Pr(a_{ij} = 0, a_{i'j'} = 0|m) = \left(\frac{n-2}{n}\right)^m,\ m \geq 0,$$

because the number of events for each node pair in block pair b follow a multinomial distribution with m trials and all event probabilities equal to $1/n$. By the Law of Total Probability,

$$\Pr(a_{ij} = 0, a_{i'j'} = 0) = \sum_{m=0}^{\infty} p(m)\left(\frac{n-2}{n}\right)^m. \tag{12}$$

We first lower bound δ_0 by noting that

$$\begin{aligned}
\delta_0 &= \Pr(a_{ij} = 0|a_{i'j'} = 0) - \Pr(a_{ij} = 0) \\
&\geq \Pr(a_{ij} = 0, a_{i'j'} = 0) - \Pr(a_{ij} = 0) \\
&= \sum_{m=0}^{\infty} p(m)\left[\left(\frac{n-2}{n}\right)^m - \left(\frac{n-1}{n}\right)^m\right] \qquad (13) \\
&= -\sum_{m=0}^{\infty} p(m)\frac{(n-1)^m - (n-2)^m}{n^m} \\
&\geq -\sum_{m=0}^{\infty} p(m)\frac{m(n-1)^{m-1}}{n^m} \qquad (14) \\
&= -\frac{1}{n}\sum_{m=0}^{\infty} mp(m)\left(\frac{n-1}{n}\right)^{m-1} \\
&\geq -\frac{1}{n}\sum_{m=0}^{\infty} mp(m) \qquad (15) \\
&= -\frac{\mu}{n}, \qquad (16)
\end{aligned}$$

where (13) follows from (11) and (12), (14) follows from Lemma A.1, and (15) follows by observing that $\left(\frac{n-1}{n}\right)^{m-1} \leq 1$.

We now upper bound δ_0 by noting that

$$\begin{aligned}
\delta_0 &= \Pr(a_{ij} = 0|a_{i'j'} = 0) - \Pr(a_{ij} = 0) \\
&\leq 1 - \Pr(a_{ij} = 0) \\
&= 1 - \sum_{m=0}^{\infty} p(m)\left(\frac{n-1}{n}\right)^m \\
&\leq 1 - \sum_{m=0}^{\infty} p(m)\left(1 - \frac{m}{n}\right) \qquad (17) \\
&= 1 - \sum_{m=0}^{\infty} p(m) + \frac{1}{n}\sum_{m=0}^{\infty} mp(m) \\
&= 1 - 1 + \frac{\mu}{n} \\
&= \frac{\mu}{n}, \qquad (18)
\end{aligned}$$

where (17) follows from Lemma A.2.

Next we lower bound δ_1 by noting that

$$\begin{aligned}
\delta_1 &= \Pr(a_{ij}=0|a_{i'j'}=1) - \Pr(a_{ij}=0) \\
&= \frac{\Pr(a_{ij}=0, a_{i'j'}=1) - \Pr(a_{ij}=0)\Pr(a_{i'j'}=1)}{\Pr(a_{i'j'}=1)} \\
&\geq \Pr(a_{ij}=0, a_{i'j'}=1) - \Pr(a_{ij}=0)\Pr(a_{i'j'}=1) \\
&= \Pr(a_{ij}=0) - \Pr(a_{ij}=0, a_{i'j'}=0) \\
&\quad - \Pr(a_{ij}=0)[1-\Pr(a_{i'j'}=0)] \\
&= -[\Pr(a_{ij}=0, a_{i'j'}=0) - \Pr(a_{ij}=0)\Pr(a_{i'j'}=0)] \\
&= -\Pr(a_{i'j'}=0)[\Pr(a_{ij}=0|a_{i'j'}=0) - \Pr(a_{ij}=0)] \\
&= -\Pr(a_{i'j'}=0)\delta_0 \\
&\geq -\delta_0 \\
&\geq -\frac{\mu}{n}, \qquad (19)
\end{aligned}$$

where (19) follows from (18).

Finally we upper bound δ_1 using the same approach as for δ_0:

$$\begin{aligned}
\delta_1 &= \Pr(a_{ij}=0|a_{i'j'}=1) - \Pr(a_{ij}=0) \\
&\leq 1 - \Pr(a_{ij}=0) \\
&\leq \frac{\mu}{n}, \qquad (20)
\end{aligned}$$

where the final inequality is obtained from (18).

Combining (16), (18), (19), and (20), and noting that the maximum deviation between two probabilities is 1, we arrive at the desired result $|\delta_0|, |\delta_1| \leq \min\{1, \mu/n\}$, which completes the proof. □

B VARIATIONAL INFERENCE DETAILS

We begin by deriving the complete-data log-likelihood for the block point process model.

$$\begin{aligned}
&\log \Pr(E, Z|\theta, \boldsymbol{\pi}) \\
&\quad = \log \Pr(Z|\boldsymbol{\pi}) + \log \Pr(\mathbf{u}, \mathbf{v}, \mathbf{t}|Z, \theta) \\
&\quad = \log \Pr(Z|\boldsymbol{\pi}) + \log \Pr(\mathbf{u}, \mathbf{v}|Z) + \log \Pr(\mathbf{t}|\mathbf{u}, \mathbf{v}, Z, \theta) \qquad (21)
\end{aligned}$$

The first term in (21) represents the prior class probability and is given by

$$\log \Pr(Z|\boldsymbol{\pi}) = \log\left(\prod_{i=1}^{N}\prod_{q=1}^{K} \pi_{iq}^{z_{iq}}\right) = \sum_{i=1}^{N}\sum_{q=1}^{K} z_{iq}\log(\pi_{iq}).$$

The second term in (21) represents the uniform distribution of events to nodes inside a block pair and is given by

$$\begin{aligned}
\log \Pr(\mathbf{u}, \mathbf{v}|Z) &= \log\left[\prod_{q=1}^{K}\prod_{l=1}^{K}\left(\frac{1}{n_{ql}}\right)^{m_{ql}}\right] \\
&= -\sum_{q=1}^{K}\sum_{l=1}^{K} m_{ql}\log(n_{ql}) \\
&= -\sum_{q=1}^{K}\sum_{l=1}^{K} \mathbf{z}_{:q}^T W \mathbf{z}_{:l}\log(\mathbf{z}_{:q}^T\mathbf{z}_{:l}),
\end{aligned}$$

where m_{ql} and n_{ql} are defined in the same manner as m_b and n_b in Section 4.1. The last equality represents these two terms as a function of Z, where $\mathbf{z}_{:q}$ denotes the qth column of $\mathbf{z}$, and a weighted adjacency matrix W where the weight w_{ij} corresponds to the total number of events from node i to node j. The third term in (21) depends on the choice of point process; for the block Hawkes model, it is given by

$$\begin{aligned}
\log \Pr(\mathbf{t}|\mathbf{u}, \mathbf{v}, Z, \theta) = \sum_{q=1}^{K}\sum_{l=1}^{K}\Bigg(\sum_{s=1}^{M}\Bigg\{ & z_{u_s q} z_{v_s l}\frac{\alpha_{ql}}{\beta_{ql}}\left[e^{-\beta_{ql}(t_M - t_s)} - 1\right] \\
& + \log\left[\lambda_{ql}^{\infty} + \sum_{r=1}^{s-1} z_{u_r q} z_{v_r l}\alpha_{ql} e^{-\beta_{ql}(t_s - t_r)}\right]\Bigg\} - t_M \lambda_{ql}^{\infty}\Bigg),
\end{aligned}$$

where u_s and v_s denote the sender and receiver nodes for event s. The complete-data log-likelihood (21) is intractable since changing the class membership of one node might affect the class memberships of other nodes, so all possibilities of Z need to be considered.

We derive a mean-field variational inference procedure in which we approximate the posterior with the fully factorizable variational distribution given by (6). We now try to find the best distribution from the family of multinomial distributions that can get us closest in KL divergence to the posterior. But calculating KL divergence requires calculating the intractable posterior, so we instead maximize the evidence lower bound (ELBO), which is equivalent to minimizing KL divergence [1]. The ELBO is given by

$$\text{ELBO}(R_E) = \mathbb{E}\left[\log \Pr(E, Z|\theta, \boldsymbol{\pi})\right] - \mathbb{E}\left[\log R_E(Z)\right].$$

Now expand and calculate expectation of individual terms in the complete-data log-likelihood (21). The expectation of the log of the summations in the second and third terms in (21) can be bounded using Jensen's inequality:

$$\mathbb{E}[f(x)] \geq f(\mathbb{E}(x)).$$

After bounding the log sums using Jensen's inequality and calculating the expectations with respect to R_E, we arrive at the simplified expression for the ELBO:

$$\begin{aligned}
\mathcal{E}(R_E) = & \sum_{i=1}^{N}\sum_{q=1}^{K}\tau_{iq}\log(\pi_{iq}) - \sum_{q=1}^{K}\sum_{l=1}^{K}\boldsymbol{\tau}_{:q}^T W \boldsymbol{\tau}_{:l}\log(\boldsymbol{\tau}_{:q}^T\boldsymbol{\tau}_{:l}) \\
& + \sum_{q=1}^{K}\sum_{l=1}^{K}\Bigg(\sum_{s=1}^{M}\Bigg\{\tau_{u_s q}\tau_{v_s l}\frac{\alpha_{ql}}{\beta_{ql}}\left[e^{-\beta_{ql}(t_M - t_s)} - 1\right] + \log\Bigg[\lambda_{ql}^{\infty} \\
& + \sum_{r=1}^{s-1}\tau_{u_r q}\tau_{v_r l}\alpha_{ql}e^{-\beta_{ql}(t_s - t_r)}\Bigg]\Bigg\} - t_M\lambda_{ql}^{\infty}\Bigg) + \sum_{i=1}^{N}\sum_{q=1}^{K}\tau_{iq}\log(\tau_{iq}) \qquad (22)
\end{aligned}$$

where $\boldsymbol{\tau}_{:q}$ denotes the vector of variational parameters for class q over all N nodes. The variational expectation-maximization (VEM) algorithm alternates between a variational E step, in which we use coordinate ascent to optimize the ELBO (22) over the variational parameters $\boldsymbol{\tau}_i$ for each node i, and an M step, in which we optimize the ELBO over the Hawkes process parameters $\left(\alpha_{ql}, \beta_{ql}, \lambda_{ql}^{\infty}\right)$ for all block pairs $(q, l) \in \{1, \ldots, K\}^2$.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation grants IIS-1755824 and DMS-1830412.

REFERENCES

[1] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. 2017. Variational inference: A review for statisticians. *J. Amer. Statist. Assoc.* 112, 518 (2017), 859–877.

[2] Charles Blundell, Jeff Beck, and Katherine A. Heller. 2012. Modelling Reciprocating Relationships with Hawkes Processes. In *Advances in Neural Information Processing Systems 25*. 2600–2608.

[3] Richard H. Byrd, Jean Charles Gilbert, and Jorge Nocedal. 2000. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming* 89, 1 (2000), 149–185.

[4] Etienne Côme and Pierre Latouche. 2015. Model selection and clustering in stochastic block models based on the exact integrated complete data likelihood. *Statistical Modelling* 15, 6 (2015), 564–589.

[5] Marco Corneli, Pierre Latouche, and Fabrice Rossi. 2016. Exact ICL maximization in a non-stationary temporal extension of the stochastic block model for dynamic networks. *Neurocomputing* 192 (2016), 81–91.

[6] Jean-Jacques Daudin, Franck Picard, and Stéphane Robin. 2008. A mixture model for random graphs. *Statistics and Computing* 18, 2 (2008), 173–183.

[7] Christopher DuBois, Carter T. Butts, and Padhraic Smyth. 2013. Stochastic blockmodeling of relational event dynamics. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*. 238–246.

[8] Christopher DuBois and Padhraic Smyth. 2010. Modeling relational events via latent classes. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 803–812.

[9] Nathan Eagle, Alex Pentland, and David Lazer. 2009. Inferring friendship network structure by using mobile phone data. *Proceedings of the National Academy of Sciences* 106 (2009), 15274–15278.

[10] Mehrdad Farajtabar, Yichen Wang, Manuel Gomez Rodriguez, Shuang Li, Hongyuan Zha, and Le Song. 2015. COEVOLVE: A joint point process model for information diffusion and network co-evolution. In *Advances in Neural Information Processing Systems 28*. 1945–1953.

[11] Eric W. Fox, Martin B. Short, Frederic P. Schoenberg, Kathryn D. Coronges, and Andrea L. Bertozzi. 2016. Modeling e-mail networks and inferring leadership using self-exciting point processes. *J. Amer. Statist. Assoc.* 111, 514 (2016), 564–584.

[12] Anna Goldenberg, Alice X. Zheng, Stephen E. Fienberg, and Edoardo M. Airoldi. 2010. A survey of statistical network models. *Foundations and Trends® in Machine Learning* 2, 2 (2010), 129–233.

[13] Eric C. Hall and Rebecca M. Willett. 2016. Tracking dynamic point processes on networks. *IEEE Transactions on Information Theory* 62, 7 (2016), 4327–4346.

[14] Peter F. Halpin and Paul De Boeck. 2013. Modelling dyadic interaction with Hawkes processes. *Psychometrika* 78, 4 (2013), 793–814.

[15] Qiuyi Han, Kevin S. Xu, and Edoardo M. Airoldi. 2015. Consistent estimation of dynamic and multi-layer block models. In *Proceedings of the 32nd International Conference on Machine Learning*. 1511–1520.

[16] Xinran He, Theodoros Rekatsinas, James Foulds, Lise Getoor, and Yan Liu. 2015. HawkesTopic: A joint model for network inference and topic modeling from text-based cascades. In *Proceedings of the 32nd International Conference on Machine Learning*. 871–880.

[17] Qirong Ho, Le Song, and Eric P. Xing. 2011. Evolving cluster mixed-membership blockmodel for time-varying networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*. 342–350.

[18] Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. 1983. Stochastic blockmodels: First steps. *Social Networks* 5, 2 (1983), 109–137.

[19] Brian Karrer and M. E. J. Newman. 2011. Stochastic blockmodels and community structure in networks. *Physical Review E* 83, 1 (2011), 016107.

[20] Charles Kemp, Joshua B. Tenenbaum, Thomas L. Griffiths, Takeshi Yamada, and Naonori Ueda. 2006. Learning systems of concepts with an infinite relational model. In *Proceedings of the 21st National Conference on Artificial Intelligence*. 381–388.

[21] Patrick J. Laub, Thomas Taimre, and Philip K. Pollett. 2015. Hawkes processes. *arXiv preprint arXiv:1507.02822* (2015). http://arxiv.org/abs/1507.02822

[22] Jing Lei and Alessandro Rinaldo. 2015. Consistency of spectral clustering in stochastic block models. *The Annals of Statistics* 43, 1 (2015), 215–237.

[23] Scott W. Linderman and Ryan P. Adams. 2014. Discovering latent network structure in point process data. In *Proceedings of the 31st International Conference on Machine Learning*. 1413–1421.

[24] Scott W. Linderman and Ryan P. Adams. 2015. Scalable Bayesian inference for excitatory point process networks. *arXiv preprint arXiv:1507.03228* (2015). https://arxiv.org/abs/1507.03228

[25] Naoki Masuda, Taro Takaguchi, Nobuo Sato, and Kazuo Yano. 2013. Self-exciting point process modeling of conversation event sequences. In *Temporal Networks*. Springer, 245–264.

[26] Catherine Matias and Vincent Miele. 2017. Statistical clustering of temporal networks through a dynamic stochastic block model. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 79, 4 (2017), 1119–1141.

[27] Catherine Matias, Tabea Rebafka, and Fanny Villers. 2018. A semiparametric extension of the stochastic block model for longitudinal networks. *Biometrika* 105, 3 (2018), 665–680.

[28] M. E. J. Newman and M. Girvan. 2004. Finding and evaluating community structure in networks. *Physical Review E* 69, 2 (2004), 026113.

[29] Yosihiko Ogata. 1981. On Lewis' simulation method for point processes. *IEEE Transactions on Information Theory* 27, 1 (1981), 23–31.

[30] T. Ozaki. 1979. Maximum likelihood estimation of Hawkes' self-exciting point processes. *Annals of the Institute of Statistical Mathematics* 31 (1979), 145–155.

[31] Tai Qin and Karl Rohe. 2013. Regularized spectral clustering under the degree-corrected stochastic blockmodel. In *Advances in Neural Information Processing Systems 26*. 3120–3128.

[32] Riccardo Rastelli. 2017. Exact integrated completed likelihood maximisation in a stochastic block transition model for dynamic networks. *arXiv preprint arXiv:1710.03551* (2017). https://arxiv.org/abs/1710.03551

[33] Karl Rohe, Sourav Chatterjee, and Bin Yu. 2011. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics* 39, 4 (2011), 1878–1915.

[34] Karl Rohe, Tai Qin, and Bin Yu. 2016. Co-clustering directed graphs to discover asymmetries and directional communities. *Proceedings of the National Academy of Sciences* 113, 45 (2016), 12679–12684.

[35] Daniel L. Sussman, Minh Tang, Donniell E. Fishkind, and Carey E. Priebe. 2012. A consistent adjacency spectral embedding for stochastic blockmodel graphs. *J. Amer. Statist. Assoc.* 107, 499 (2012), 1119–1128.

[36] Long Tran, Mehrdad Farajtabar, Le Song, and Hongyuan Zha. 2015. NetCodec: Community detection from individual activities. In *Proceedings of the SIAM International Conference on Data Mining*. 91–99.

[37] Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P. Gummadi. 2009. On the evolution of user interaction in Facebook. In *Proceedings of the 2nd ACM Workshop on Online Social Networks*. ACM, 37–42.

[38] Lu Xin, Mu Zhu, and Hugh Chipman. 2017. A continuous-time stochastic block model for basketball networks. *The Annals of Applied Statistics* 11, 2 (2017), 553–597.

[39] Eric P. Xing, Wenjie Fu, and Le Song. 2010. A state-space mixed membership blockmodel for dynamic network tomography. *The Annals of Applied Statistics* 4 (2010), 535–566.

[40] Kevin S. Xu. 2015. Stochastic block transition models for dynamic networks. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*. 1079–1087.

[41] Kevin S. Xu and Alfred O. Hero III. 2014. Dynamic stochastic blockmodels for time-evolving social networks. *IEEE Journal of Selected Topics in Signal Processing* 8, 4 (2014), 552–562.

[42] Tianbao Yang, Yun Chi, Shenghuo Zhu, Yihong Gong, and Rong Jin. 2011. Detecting communities and their evolutions in dynamic social networks—a Bayesian approach. *Machine Learning* 82, 2 (2011), 157–189.

[43] Qingyuan Zhao, Murat A. Erdogdu, Hera Y. He, Anand Rajaraman, and Jure Leskovec. 2015. SEISMIC: A self-exciting point process model for predicting tweet popularity. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1513–1522.

[44] Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. 2012. Consistency of community detection in networks under degree-corrected stochastic block models. *The Annals of Statistics* 40, 4 (2012), 2266–2292.

OUTGUARD: Detecting In-Browser Covert Cryptocurrency Mining in the Wild

Amin Kharraz
University of Illinois
Urbana-Champaign

Zane Ma
University of Illinois
Urbana-Champaign

Paul Murley
University of Illinois
Urbana-Champaign

Charles Lever
Georgia Institute of Technology

Joshua Mason
University of Illinois
Urbana-Champaign

Andrew Miller
University of Illinois
Urbana-Champaign

Nikita Borisov
University of Illinois
Urbana-Champaign

Manos Antonakakis
Georgia Institute of Technology

Michael Bailey
University of Illinois
Urbana-Champaign

ABSTRACT

In-browser cryptojacking is a form of resource abuse that leverages end-users' machines to mine cryptocurrency without obtaining the users' consent. In this paper, we design, implement, and evaluate OUTGUARD, an automated cryptojacking detection system. We construct a large ground-truth dataset, extract several features using an instrumented web browser, and ultimately select seven distinctive features that are used to build an SVM classification model. OUTGUARDachieves a 97.9% TPR and 1.1% FPR and is reasonably tolerant to adversarial evasions. We utilized OUTGUARDin the wild by deploying it across the Alexa Top 1M websites and found 6,302 cryptojacking sites, of which 3,600 are new detections that were absent from the training data. These cryptojacking sites paint a broad picture of the cryptojacking ecosystem, with particular emphasis on the prevalence of cryptojacking websites and the shared infrastructure that provides clues to the operators behind the cryptojacking phenomenon.

CCS CONCEPTS

• **Security and privacy → Browser security**; *Web application security*.

KEYWORDS

Browser Security; Web Security; Cryptojacking;

ACM Reference Format:
Amin Kharraz, Zane Ma, Paul Murley, Charles Lever, Joshua Mason, Andrew Miller, Nikita Borisov, Manos Antonakakis, and Michael Bailey. 2019. OUTGUARD: Detecting In-Browser Covert Cryptocurrency Mining in the Wild. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3308558.3313665

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313665

1 INTRODUCTION

The financial success of cryptocurrencies, which are valued at over 280 billion USD in market capitalization[1], has drawn the attention of malicious actors, who attempt to accumulate wealth through illegitimate means. One such method is cryptojacking, which entails hijacking of a victim's processing power to mine cryptocurrency without consent. Cryptojacking has been reported in several forms, including botnets [21, 55] and compromised cloud computing infrastructure [43] that mine cryptocurrencies to benefit the attackers.

Recently, a new strain of cryptojacking has emerged: in-browser cryptojacking, where JavaScript cryptocurrency miners are deployed through websites. In-browser cryptojacking can be enabled primarily by two developments. First, researchers designed specialized proof-of-work mining algorithms (e.g., Cryptonight [6]) that make CPU mining reasonably competitive. Second, in-browser JavaScript performance has been improved through new web standards (e.g., WebAssembly) and general increases in end-user computational power. These advances, coupled with the broad deployment capabilities of the web, have led to reports of in-browser cryptojacking on vulnerable websites [36], public Wi-Fi networks [10], ad-networks [37], and a popular news site [35].

Existing detection mechanisms for in-browser cryptojacking (henceforth simply referred to as cryptojacking) are nascent and primarily centered around blacklisting [1, 12, 49] and CPU usage profiling [45]. Blacklisting is a reactive technique that struggles to keep up with rapidly evolving web threats [32]. CPU usage profiling is imprecise and can mislabel online games, videos, and other processing intensive JavaScript web applications. As we demonstrate in Section 3.4.2, this method suffers from high false positive rates of up to 13% on a small-scale dataset. Recently, L1 cache measurements have been utilized to effectively detect cryptojacking incidents [30]. While this technique is an improvement over the status quo, it is limited by the need for manual, case-by-case reverse engineering of WebAssembly code. Furthermore, reliance on the state of the L1 cache is prone to noise from other running processes and would affect overall detection accuracy.

This paper presents OUTGUARD, an open-source[2] cryptojacking detection system that uses a machine learning classifier to identify

[1] Market cap for the top 100 cryptocurrencies as of July 2018, according to [11].
[2] Both code and data are available at https://github.com/teamnsrg/outguard

cryptojacking at scale. As a first step to building OUTGUARD, we systematically constructed a labeled dataset by scanning the Alexa Top 1M domains with a specially instrumented browser and recording each site's resources, provenance, and JavaScript execution traces. Applying existing cryptojacking detection tools to the scan data yielded a set of 3,006 cryptojacking execution traces from 12 different families. We assembled these cryptojacking traces into two datasets: a balanced dataset with equal numbers of cryptojacking and benign website traces, and an imbalanced dataset that more closely represents the real-world frequency of cryptojacking.

Starting with a wide range of features that pertain to JavaScript execution, network behavior, and DOM characteristics, we carefully analyzed the labeled data to ultimately narrow down to seven features in four feature categories: browser-based, mining-based, load-based, and network-based. We tested two different types of machine learning models, Support Vector Machines (SVM) and Random Forests (RF), and ultimately chose SVM as the classifier for OUTGUARDbecause it achieves 97.9% TPR and 1.1% FPR.

Effective threat detection systems inevitably prompt malicious actors to adopt detection evasion techniques. We examined the resilience of OUTGUARDto multiple forms of adversarial evasion by removing each feature category and measuring the overall detection accuracy. Our analysis showed that while an adversary can potentially evade some of the features used in the learning process, such evasion techniques severely cripple cryptojacking operations and curtail an adversary's profits. For instance, removing mining-based and network-based features (e.g., WebAssembly) is feasible, but their removal makes cryptojacking less attractive to adversaries.

We deployed OUTGUARDin the wild on the Alexa Top 1M domains and discovered 6,302 cryptojacking sites, of which 3,600 had not been detected by existing techniques. Since in-browser cryptomining, in theory, is not necessarily malicious if user consent is obtained, we performed additional analysis to verify that, in reality, nearly all cryptomining sites did not ask for user consent.Through the lenses of both the Alexa Top 1M domains and passive DNS data, cryptojacking appears predominantly in the long tail of lower popularity sites that promote illicit content such as torrents or copyrighted video streaming. Our analysis also exposed 69 incidents of cryptojacking loaded through advertisement web pages. These incidents constituted unintentional cryptojacking, which occurred without the direct involvement of website owners/publishers.

We examined the delivery of cryptojacking and coordination of cryptojacking monetization. We classified cryptojacking incidents into 14 major families based on their source code origin and found a competitive blend of five families that, in total, accounted for more than 70% of cryptojacking occurrences. We also detected 486 cryptojacking incidents using 24 less well-known cryptomining services. These services provide pool mining services that charge as little as 1% of total mining revenue. Finally, we conducted a campaign analysis and identified 35 cryptojacking campaigns which ranged in size from 2 to 121 websites. These websites deliver different types of cryptojacking libraries such as CoinHive, JSeCoin, and Crypto-loot.

Given the extent of the identified cryptojacking websites, we envision multiple deployment scenarios for disrupting cryptojacking operations in the wild. The output of OUTGUARDcould be used to augment blacklists or anti-malware mechanisms in popular browsers. The proposed detection model could also be used as a browser extension that preemptively monitors the content of the visited websites and notifies the user if the website loads a cryptojacking library.

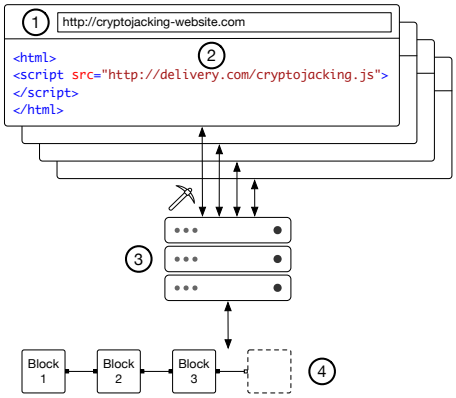


Figure 1: Cryptojacking overview. Websites (1) include cryptojacking code from a delivery server (2). Victim browsers execute mining tasks assigned from a mining pool/proxy (3), which ultimately adds new blocks to the blockchain (4).

2 BACKGROUND AND RELATED WORK

2.1 Cryptojacking Workflow

Modern cryptocurrencies are based on blockchain consensus, which was first presented and popularized by Bitcoin [41]. The blockchain is maintained and extended by miners, which typically compete to compute a proof-of-work puzzle that must be satisfied by new blocks. This proof-of-work serves as both a denial-of-service deterrent and an economic indication of commitment to the consensus protocol. Initial proof-of-work protocols, such as Bitcoin's Hashcash, benefit substantially from custom hardware optimizations, and most mining now occurs on custom mining rigs.

The development of Cryptonight [6], a new proof-of-work algorithm that significantly reduces the advantages of custom hardware, positioned GPUs, custom FPGAs/ASICs, and CPUs back into the same ballpark. In fact, Hashcash relies on brute-forcing SHA-256, which is easily parallelized and reliant on computational bandwidth, making it amenable to customized ASIC hardware. Cryptonight, on the other hand, is a memory-bound protocol (2Mb per core) that relies on random access to slow memory and each new block relies on the results of the prior block. One likely side effect of this development is cryptojacking, which allows an adversary to exploit a victim's computing power to mine cryptocurrencies without consent. Figure 1 outlines the cryptojacking workflow, which involves several entities, starting with user-facing cryptojacking websites. These websites are loaded through a user's JavaScript-enabled web browser and embed a cryptojacking JavaScript library, either through intentional design or hacker injection. The cryptojacking library is delivered from a delivery server, which can reside on the same domain as the base website or, more commonly, is loaded through a separate delivery server host.

Once the cryptojacking code has begun execution, it connects to a mining pool proxy in order to receive mining tasks. A proxy server

is often employed between the mining pool and cryptojacking miners to limit the number of connections and further subdivide mining tasks assigned by the mining pool. Prior to mining, each mining operator (who may control more than one miner) will have registered a mining account with the mining pool, so that she can receive payments for her cryptojacking efforts. Some form of account identifier is usually transmitted from the cryptojacking client to the mining pool to correctly map mining output to the correct mining account. These communications typically occur via Stratum [2], a JSON RPC protocol for mining pool coordination that is cryptocurrency-agnostic. The mining pool monitors the blockchain and assigns mining tasks to each miner that connects to it. These tasks are a low threshold version of the actual proof-of-work puzzle, so that the mining pool can record and reward incremental mining effort. Occasionally, when one of the completed tasks constitutes a successful block, the mining pool adds the block to the blockchain and claims the associated mining reward for itself. Then, periodically, the mining pool will distribute a portion of this reward among its mining accounts in proportion to the work accomplished; the rest is kept as a mining pool fee. These fees are typically around 10–30%, but can be as low as 1% or 2% [50].

The description provided thus far has focused on the components involved in the cryptojacking workflow. In reality, many of these components are controlled by a shared operator. To elucidate the common mappings between components and operators, we present two scenarios: full-service cryptojacking and do-it-yourself (DIY) cryptojacking. For full-service cryptojacking such as CoinHive, a website owner or hijacker only needs to create a CoinHive account and then insert a script inclusion tag her website. In this scenario, CoinHive operates the delivery server and the mining pool / proxy. For a DIY use case, the publisher may host her own delivery server and have the cryptojacking code connect to a public mining pool.

2.2 Related Work

Techniques seeking to gain financial profit through unauthorized access to a victim computer are numerous. There is a large body of work studying adversarial attempts to profit from end-user machines. Ransomware [25, 27, 29, 44], malicious advertisements [7, 34, 56], click fraud [19], web-based social engineering attacks [5, 8, 9, 26, 33, 51, 53], online scams [28, 39], and Potentially Unwanted Programs (PUPs) [31, 42, 48] are only a few examples of these activities in the wild. In the following paragraphs, we explain more closely related work on cryptojacking operations.

A subset of prior work on cryptojacking [16, 40, 46] mainly focuses on the impacts of unwanted mining on user machines (e.g., CPU utilization), and provides an estimate of network size and revenue for CoinHive. In this paper, we mainly focus on less noisy features that can facilitate automatic detection. Very recently, Konoth et al. [30] and Hong et al. [20] concurrently and independently performed systematic analyses on cryptojacking incidents. Hong et al. [20] proposed two techniques by statically searching for a set of fixed signatures in cryptojacking code as well as identifying patterns in the execution stack. The authors demonstrated the insufficiency of blacklists and provide insight on a set of techniques (e.g., obfuscation) that are currently used by adversaries to bypass blacklists. While there are some similarities between their detection approach and ours (e.g., identifying specific patterns), OUTGUARDmainly relies on dynamic behaviors of cryptojacking libraries that are the core components of in-browser cryptojacking, and that are quite unlikely to be used in the same way in benign programs. We elaborate on the feature set in Section 3, and show that the system still achieves high detection coverage if adversaries utilize different levels of obfuscation techniques.

The prior work most relevant to ours is MineSweeper [30], which introduces a static analysis technique to study the wasm code used by cryptojacking code. While this approach is an improvement over the status quo, it would be desirable to complement it with a more generic approach that requires minimal human intervention. OUTGUARDdoes not require manual analysis, and does not rely on cryptographic primitive identification. This was a conscious design choice to avoid possible false positive cases, as acknowledged by the authors [30]. MineSweeper also shows that CPU cache L1 event monitoring can be quite useful in detecting cryptojacking websites by analyzing two cryptojacking families with 100% CPU usage. While WebAssembly provides significant freedom to manage the memory layout, an adversary may not be able to selectively load mining operations in the L1 or L3 cache in order to bypass the proposed detection. However, our analysis reveals that cryptojacking libraries impose different L1 cache event loads as a function of the CPU threshold, leaving threshold-based approaches still vulnerable to future evasion. Furthermore, measuring CPU cache events is a very expensive operation, as the entire process of inferring the state of the CPU cache is very sensitive to noise and to the global status of other active applications on the test machine. This makes approaches similar to MineSweeper less likely to scale. OUTGUARDdoes not have this limitation, and the detection accuracy of OUTGUARDis not impacted by the type of cryptographic function or CPU-related features (see Section 3). Thus, OUTGUARDis a more generic solution to the same problem space.

3 OUTGUARD

Because cryptojacking is still in its infancy, existing detection mechanisms are still sparse and relatively naive: NoCoin [45], minerBlock [4], and CoinBlockerLists [1] rely on open-source, user-updated blacklists to match domains or filenames. Building of an automated cryptojacking detection system in this nascent landscape requires a full construction process: building a ground-truth dataset, creating features, selecting features, training a classification model, and finally evaluating the model under benign and adversarial inputs. In this section, we describe OUTGUARD's architecture, shown in Figure 2, and detail its implementation and performance.

3.1 Building Ground Truth

Currently, no reliable labeled datasets for cryptojacking websites exist. To begin building our training dataset, we first aggregated the frequently updated public blacklist patterns in CoinBlockerLists [1] as well as the blacklist patterns used in popular browser extensions NoCoin [45] and minerBlock [4]. We scanned the Alexa Top 1M websites using our instrumented Chromium browser and collected 3,006 websites that contained JavaScript libraries or delivery servers matching the blacklist patterns.

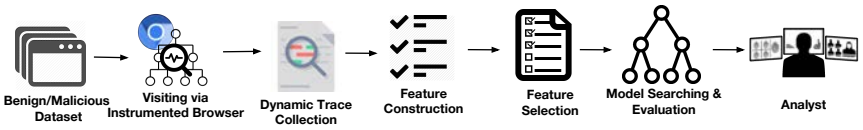


Figure 2: A high-level view of OUTGUARD's architecture.

Table 1: The training set of cryptojacking libraries we collected. Wappalyzer recognized all the libraries collected, but lacks market share (MS%) data for most of the families at the time of writing. Our labeled dataset of 3,006 websites covers 12 of 14 (85%) cryptojacking libraries.

Code Family	Currency	MinerBlock	Wappalyzer (MS%)	Labeled Dataset (%)
JSEcoin	JSECoin	✓	✓(30.6%)	1,358 (45.2%)
CoinHive	Monero	✓	✓(48.2%)	1,124 (37.4%)
CoinHive Captcha	Monero	✓	✓	132 (4.4%)
DeepMiner	Monero	✓	✓	88 (2.9%)
Coinimp	Monero	✓	✓	68 (2.3%)
Crypto-Loot	Monero	✓	✓	55 (1.8%)
ProjectPoi	Monero	-	✓(19.2%)	43 (1.4%)
Coinlab	Monero	✓	✓	43 (1.4%)
Cloudcoins	Monero	✓	✓	42 (1.4%)
Moneromine	Monero	✓	✓	28 (0.9%)
Coinhave	Monero	-	✓	15 (0.5%)
Monero.cc	Monero	✓	✓	10 (0.3%)
Webmine	Monero	-	✓	-
Inwemo	Monero	-	✓	-
Total	-	10 families	14 families (100%)	3,006 (100%)

To further label the cryptojacking libraries, we used Wappalyzer [52] version 5.2, a web technology detection tool that lists the libraries used by a given website. Wappalyzer labels cryptojacking JavaScript (JS) code by examining its origin. To create our labeled dataset, we incorporated the same methodology by mapping URLs to cryptojacking family names. For example, if the code originated from `http//cdn.cloudcoins.co`, we labeled the code as an instance of the Cloudcoin family. As shown in Table 1, we found 12 cryptojacking families in our labeled data, which cover 100% of the cryptojacking libraries known to MinerBlock. We did not find any instances of two out of the 14 total cryptojacking libraries known to Wappalyzer: *Inwemo* and *Webmine*. Inwemo is a seemingly defunct advertising replacement product [22], and Webmine is a similar product based in the Czech Republic.

To build a ground-truth set of non-cryptojacking websites, we randomly selected websites from the Alexa Top 100K. Even though there have been a few reported cases of high popularity sites exhibiting cryptojacking [13, 35], these websites are generally well-maintained and less likely to be subject to new security incidents (as later indicated in Section 5). The non-cryptojacking dataset covers a variety of real-world websites, including dynamic and high-profile websites such as news, legitimate video streaming, and popular companies' websites as well as static websites. This also contains a wide range of JS usage, including high-usage sites (e.g., cnn.com). We combined the ground truth cryptojacking and non-cryptojacking websites into two training datasets: a balanced dataset and an imbalanced dataset. The balanced dataset (BD) contains equal numbers of 2,000 cryptojacking and 2,000 benign websites. The cryptojacking websites were selected randomly and include members of all the 12 labeled cryptojacking families. The motivation behind the imbalanced dataset (ID) is that, in reality, there are significantly more benign websites than cryptojacking websites, and a balanced distribution may bias the accuracy of the classifier. To analyze the performance of the classifier, we built the ID with an imbalance ratio of 1 to 10, such that it contains 2,700 cryptojacking websites and 27,000 benign pages.

3.2 Feature Selection

After carefully examining a variety of cryptojacking websites in our labeled data, we developed a set of features that are broadly characteristic of cryptojacking libraries. Starting with the raw trace data from our data collection, we considered almost all the related features in rendering and JavaScript engine that could be useful in attributing the runtime behavior of JS code. However, we narrowed down our list to 12 features, as we observed those 12 features can potentially reflect cryptojacking behaviors more accurately. We investigated the following features, which we suspected would be strong indicators of cryptojacking, but that did not prove to be distinctive cryptojacking features.

- *JS engine execution time*: We expected JS execution time to dominate a cryptojacking website, but found that this feature is very noisy; many websites continuously execute JS.
- *JS compilation time*: Cryptojacking did not display significant differences in JS compilation time.
- *Garbage collection*: Garbage collection is often a loose proxy for memory usage, but we didn't find differentiating behavior for cryptojacking libraries.
- *Iframe resource loads*: Cryptojacking code is often loaded in an iframe, but rampant advertising iframes drowned out meaningful data.
- *CPU usage*: Cryptomining relies directly on CPU resources, but many cryptojacking libraries throttle CPU usage and are hard to distinguish from JavaScript-heavy sites.

Ultimately, by applying our domain knowledge and evaluating different potential features, we converged on seven features that represent the JS runtime execution, networking, JS event loads, and cryptomining properties of cryptojacking libraries. These features, with the potential exception of WebSockets, are essential for the profitable operation of cryptojacking campaigns. We show that all seven features are nearly ubiquitous across all 12 families of cryptojacking libraries in our labeled training set. We expand on each feature below.

Web Workers. Web workers are a modern JS feature first specified by W3C in 2009 [23] for executing JS files in background threads, independent of a site's main execution thread. Compared to non-cryptojacking usage of web workers for processor-intensive tasks (e.g., image processing during photo upload), we found that cryptojacking libraries are more likely to utilize *multiple* web worker threads for running several mining tasks concurrently. We observed that almost all 12 families utilize at least 3 web workers to perform mining. OUTGUARD's instrument browser dynamically extracts the number of threads as a feature for cryptojacking detection.

Parallel Tasks. We observed that benign websites typically create threads for specific purposes (e.g., caching or streaming video). Our analysis shows that in all the cryptojacking websites, in addition to running multiple web worker threads, cryptojacking libraries tend to run the *same* tasks in each thread, repeatedly invoking the same sequence of method calls. OUTGUARDoutputs the number of identical tasks as a numeric value for detection.

WebAssembly. WebAssembly, or wasm [54], is a new web standard introduced in 2015 that allows web developers to specify assembly-like code that executes on modern browsers at near-native speeds. The performance of WebAssembly makes it an attractive target for processing intensive cryptomining, and we found that all cryptojacking libraries in our training set use WebAssembly. OUTGUARDdetects the usage of WebAssembly by monitoring the exchange of wasm modules between the main execution thread and web workers. This detection technique, which is oblivious to the communication obfuscation found in some variants of CoinHive, results in a boolean feature that indicates the presence or absence of WebAssembly.

WebSockets. WebSockets are a low-overhead, full-duplex communication channel over TCP that was standardized in 2011. Our analysis shows that most mining modules in a browser establish a WebSocket connection to a remote mining pool/proxy. In addition to overcoming some limitations of JavaScript APIs in accessing the TCP socket, WebSocket provides a faster communication channel than standard TCP, as it eliminates the sending of additional HTTP headers for each client-side request. We also observed that cryptojacking libraries use WebSocket APIs to receive data from the mining pool which reduces bandwidth usage. OUTGUARDsearches for `webSocketCreate` events in its JS execution traces and outputs a numeric feature value for the number of WebSocket connections.

Hash Algorithms. As shown in Table 1, Monero is the most common cryptocurrency mined by cryptojacking libraries. Monero mining is a proof-of-work (PoW) task that requires the repeated calculation of the CryptoNight hashing algorithm, which can be calculated efficiently on consumer CPUs. OUTGUARDidentifies CryptoNight hashing by searching for specific function names and binary signatures in JS execution call stacks. For instance, *JSECoin* and some *CoinHive* variants call functions named `module._cryptonight_hash` and `CryptonightWASMWrapper.hash`, respectively. On the other hand, CoinHave embeds and executes binary wasm hashing functions inline for which we also developed detection signatures. This detection mechanism outputs a boolean feature.

PostMessage Event Load. Web workers constantly have to send the results of mining operations to the main process by sending the job id, the nonce, and the result of each operation. This information is then sent to mining proxies over WebSocket connections in order to manage mining tasks. The communication between the renderer process and the web workers is handled via PostMessage. As the number of web workers is significantly high in cryptojacking websites, and PostMessages are the only form of communication with web workers in most modern browsers, a large number of PostMessage events in site visits can be an indication of cryptojacking. We report the total number of PostMessage events as a numeric feature.

MessageLoop Event Load. Chromium performs task management among threads by using MessageLoop. When a WebAssmebly module is loaded into a web worker thread, it starts cryptographic operations inside loops. These operations are pushed to the MessageLoop of the thread to be sequentially processed. While use of MessageLoop does not by itself necessarily mean that a website is performing cryptojacking operations, a significantly large number of MessageLoop events during a short site visit, along with other features, can be an indication of cryptojacking. We report the number of MessageLoop events as a numeric value.

3.3 Prototype Implementation

Automatic recording of the dynamic behavior of client-side JavaScript code, such as resource usage, code provenance, and network communication, is a non-trivial challenge. In order to extract these features from websites, we instrumented Chromium and leveraged the Chromium Debugging Protocol [17] to access nearly all the functionality of DevTools [18]. More specifically, the crawler collected browser request and response traces, JavaScript execution traces, and the source code of both inline and dynamically loaded JavaScript code. This set of information was sufficient to attribute specific JavaScript code to the requested computational

Table 2: 10-fold cross-validation on Support Vector Machines (SVM) and Random Forest (RF), using balanced (BD) and imbalanced (ID) datasets. We selected SVM as the classifier for OUTGUARDbecause of its superior 96.2–97.9% TPR and 0.2–1.1% FPR.

Metric	SVM		Random Forest	
	BD	ID	BD	ID
TPR	96.2%	97.9%	95.6%	97.1%
FPR	0.2%	1.1%	1.3%	2.8%
AUC	97.8%	98.4%	96.9%	98.1%

resources, the functions being executed, any external code referenced by the function, and the execution time spent per function. This trace data includes all features we discussed with respect to the feature selection phase. We deployed the OUTGUARDcrawler on 40 virtual machines. For each website, the crawler stayed on the visited website for 45 seconds and interacted with the page by programmatically scrolling to the bottom of the page to simulate user behavior. We empirically found that 45 seconds was sufficient time for a cryptojacking website to load a cryptojacking library, communicate with a mining pool, and run mining tasks. Our analysis on 1,000 randomly selected websites shows that the run-time overhead of the instrumentation layer is approximately 2%.

The second module of OUTGUARDis a classifier that analyzes the collected data and detects cryptojacking websites by incorporating the features described in Section 3.4. To construct the detection model, we tested multiple classification algorithms and found that an SVM classifier produced the best detection results on the training dataset. To construct the classification model, we used the SVM implementation provided by scikit-learn [47]. We evaluate the classifier in Section 3.4.

3.4 Evaluation

In this section, we examine OUTGUARD's detection accuracy on the ground-truth training dataset and compare it with CPU utilization, an existing cryptojacking detection heuristic. To identify the best machine learning algorithm for automatic classification of cryptojacking libraries, we ran a 10-fold cross-validation on our BD and ID training datasets by using the scikit-learn [47] implementations of Support Vector Machines (SVM) and Random Forests (RF), based on prior work that showed these two classification algorithms work well on balanced and imbalanced datasets [28]. As shown in Table 2, both models perform above 95% TPR and below 3% FPR. We selected SVM as the default classifier for OUTGUARDbecause it slightly outperforms the RF model across both training sets.

The SVM model chosen for OUTGUARDis highly selective, achieving TPR rates of 96.2% and 97.9% for our BD and ID datasets, respectively. Our analysis of the detected cases revealed that OUTGUARDachieved a false positive rate (FP) of 1.1% (with 32 false detections) on an imbalanced dataset. Further manual analysis revealed that all the false positive cases were a set of parked domains that inserted 29 JavaScript scripts, on average, including trackers and advertisements. These websites were falsely labeled as cryptojacking because multiple trackers were using WebSocket connections for communications, and some of these scripts created multiple threads for client-side data processing. Our manual analysis of the false negative cases revealed that OUTGUARDwas unable to detect some variants of CoinImp because they did not contain indicative function calls in their execution traces, and they used fewer than three web workers.

3.4.1 Feature Analysis. We divided the features we selected in Section 3.2 into four categories in order to reason about subsequent analysis, such as feature ranking and classification evasion, from a feature function perspective. The categories are (1) runtime execution-based features such as WebAssembly usage; web worker quantity, and running of parallel tasks for mining activities; (2) a network-based feature which describes the type of communication between the mining module and the mining pool; (3) event load-based features such as MessageLoop and PostMessage Load, and (4) a mining-based feature that looks for hash algorithm fingerprints.

To determine the significance of each incorporated feature, we performed *recursive feature elimination* (RFE) using OUTGUARD's SVM model on the BD dataset. The ranking procedure begins with all seven features. At each subsequent step, a feature with the minimum weight is removed, and the FP and TP rates are re-calculated to measure the contribution of the removed feature. Table 3 displays the ranks of all features, in descending order of importance. For convenience, we derived a score ratio by dividing each feature score with the maximum score. We found that the five most predominant detection features target the dynamic characteristics of cryptojacking, making the system more resilient to common evasions such as code obfuscation. This matches our intuition that such behaviors contribute to successful mining operation – avoiding these techniques would likely impact revenue generation.

3.4.2 Evaluating the Effectiveness of CPU Utilization. We compared OUTGUARD's classification model against a CPU usage model which is a detection heuristic utilized by several reports [14, 38]. We collected the traces of all websites in our BD training dataset that consumed at least 50% of the total CPU capacity. One might expect a 50% CPU usage threshold to be reasonable for detecting cryptojacking given that we observed cryptojacking libraries from all 12 families establish CPU thresholds of at least 50% of the total CPU capacity. The CPU usage heuristic was applied to the BD. Out of the 2,000 non-cryptojacking websites in BD, CPU usage heuristic misclassified 263 as cryptojacking websites (false positive rate = 13.1%). Manual inspection of a handful of these cases revealed non-cryptojacking websites that introduced high CPU overhead due to a large number of requests and responses, constant JavaScript compilation, and frequent dynamic changes in the DOM during script loads. For example, a visit to cnn.com, a high-profile news website, generated an execution trace with 151 unique domain names that were used to retrieve JavaScript resources and other content (e.g., fonts, images, and styles).

Our comparative analysis demonstrates that it is not uncommon for non-cryptojacking websites to exhibit high CPU utilization, even for websites that typically do not require elevated CPU usage in order to function. CPU utilization can be a helpful feature for analysis of cryptojacking at small scales, but it is a less effective

Table 3: Feature importance ranking in OUTGUARD. Features are grouped into categories based on runtime execution (E), mining (M), Event Load (L), and network (N) behavior.

Rank	Feature	Category	Score Ratio	New?
1	No. Web Workers	E	100%	✓
2	Creating Identical Tasks	E	93.7%	✓
3	PostMessage Event Load	L	91.2%	✓
4	Using WebAssembly	E	85.8%	[20, 30]
5	MessageLoop Event Load	L	82.3%	✓
6	Using Hash Functions	M	76.4%	[20]
7	No. WebSocket Connections	N	48.2%	[30]

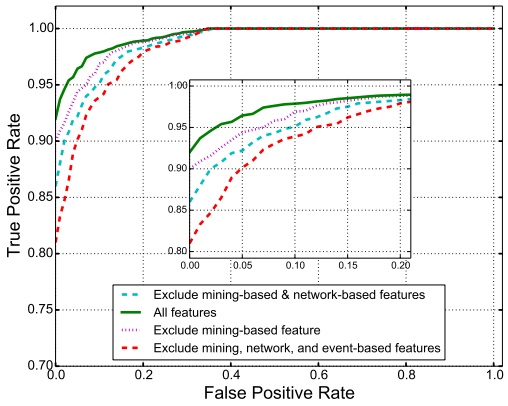


Figure 3: Detection results of OUTGUARDon the labeled dataset when different evasive scenarios were used.

feature for automatic detection at scale. CPU usage is a noisy signal—relying on this heuristic increases the likelihood of false positive cases when hundreds of thousands of websites are being analyzed.

3.5 Adversarial Considerations

Cryptojacking operators, like malware authors, are likely to exercise techniques to evade OUTGUARDby changing their implementation strategies. To address potential evasion scenarios, we evaluated the efficacy of OUTGUARD's SVM model with specific feature categories selectively removed. Figure 3 summaries the results of the experiment. The solid green curve shows the ROC curve of OUTGUARDwhen it incorporates all the features into the detection model. The dotted purple curve represents the ROC curve in the absence of mining-based features. Evasion of this feature from the detection model is a realistic possibility as our approach to detect hashing algorithm usage is somewhat brittle because it is based on collecting in-line binaries that are statically included in cryptojacking libraries or on finding references to a specific hash function in execution traces. Cryptojacking operators can re-generate cryptojacking binaries by adding dead code or by renaming functions, much as evasive malware does, to bypass mining-based feature. In this scenario, the detection performance of OUTGUARDdegrades, but it still achieves a relatively high detection accuracy, with 93% TPR and 1.2% FPR. The mining-based feature is important, but as suggested by the RFE analysis, it is not precariously important, since OUTGUARDstill achieves high detection results without it.

In the next step, we excluded the network-based feature with the assumption that a cryptojacking library may forgo a websocket connection to the mining pool/proxy server and connect through other means instead. For instance, the code may use standard TCP connections or JSONP or other communication techniques. In this case, OUTGUARDwill rely on event loads and runtime execution features to identify cryptojacking activity. The dashed blue ROC curve illustrates OUTGUARD's performance with mining-based and network-based features excluded. The system still achieves 92% TPR and 1.5% FPR. In the final experiment, we removed the event load features under the assumption that adversaries may try to limit the number of threads, and thereby impact the generated events between the web workers and the host process. The red ROC curve illustrates OUTGUARD's performance. In this case, OUTGUARDrelies solely on runtime execution features to identify cryptojacking activity. Use of runtime features (e.g., WebAssembly) in cryptojacking libraries is vital from profitability perspective, as these libraries have to incorporate such browser functionalities to use system resources effectively. Failing to do so would make the in-browser cryptomining not practical or less attractive for generating revenue. So in this final scenario, the classification accuracy has decreased significantly with the loss of networking features, down to 87% with 1.1% FPR.

Note that while an adversary can evade network or event load features, doing so will likely have a significant effect on the efficacy of mining operations, as the mining module requires low-overhead connections in order to communicate with mining pools. Furthermore, as discussed in Section 2, in-browser cryptojacking relies on multiple external parties to operate. Consequently, network-level changes requires compatibility with the rest of the parties involved in the ecosystem (e.g., pools and proxies). Moreover, minimizing of JS event loads to evade the corresponding feature set is not a trivial task, as the entire operation, including task management and task execution, significantly depends on the communication between the host process and the mining threads. Overall, we argue that a tool similar to OUTGUARDcan provide robust detection of cryptojacking websites and potentially increase the operational costs of cryptojacking or reduce the generated revenue of such operations.

4 REAL-WORLD DEPLOYMENT

We demonstrated the feasibility of real-world deployment of OutGuardby performing an analysis of the Alexa Top 1M websites. OutGuarddetected 5,873 instances of cryptojacking. Using this data, we characterized the broader cryptojacking ecosystem by examining the websites on which cryptojacking occurs, and the infrastructure supporting its mining and monetization. We later performed a scan of the Alexa Top 600K websites to supplement the case study.

For the real-world application of OutGuard, we enabled Google Safe Browsing in our instrumented browser to avoid falsely reporting cryptojacking that might be prevented by default browser protection mechanisms. That allowed us to represent realistically the cryptojacking exposure of an end-user browsing the web. We also modified OutGuardto access 3 random links on each Alexa Top webpage to simulate user interaction/navigation on the page. The measurements were conducted over two time ranges: the first scan took place from February 12 to April 15, 2018, on the Alexa Top 1M websites, and the second scan took place from October 20 to October 30, 2018, on the Alexa Top 600K websites. In total, OutGuardreported 6,302 cryptojacking websites across 5,128,117 URLs, as shown in Table 4.

4.1 Obtaining User Consent

Mining cryptocurrency in a browser is not inherently malicious, and in-browser cryptomining has been presented as a justifiable business model alternative to advertising and captcha services. The lack of user consent is the distinguishing factor between legitimate in-browser cryptomining from cryptojacking.

We performed a manual analysis of 100 randomly selected websites that contained the JSECoin library by visiting the websites and checking whether the opt-in system was enabled on them. 22% of the websites were not reachable during the manual check (e.g., `mxqproject.com, masir.us`). We found that in the remaining 78% of websites, the JSECoin cryptojacking library was automatically executed without user consent (e.g., `media9.pk, seriesenlinea.net`). Furthermore, on late 2017, CoinHive introduced a version of cryptomining code called AuthedMine, which requires user permission to turn the browser into a Monero miner. We did not find any instance of this library either in the training or in the large-scale deployment of OutGuard, which mirrors other reports [24]. This experiment suggests that website operators are likely to be reluctant to use libraries with explicit opt-in forms, and that most detected cases do, in fact, constitute cryptojacking.

4.2 Detection Coverage Evaluation

In total, OutGuarddetected 6,302 cryptojacking websites. 3,600 of the detected sites in our two experiments were newly detected websites not observed in the training dataset. Since we do not have ground truth labeled data in the large-scale experiment, we cannot provide precision-recall analysis of the detection results. However, manual analysis shows that OutGuardfalsely reported 14 websites as cryptojacking because they were loading several frames and iframes to embed videos and pop-ups. In addition to creating multiple threads, we observed several WebSocket connections with remote servers to load streaming content in these websites. We consider these cases false positives. The manual analysis of the trace files indicated that the remaining newly detected websites were true positives. We cross-checked the output of OutGuardagainst dedicated security solutions such as MinerBlock [12] and Antiminer [49]. Table 4 summarizes the results of the experiment. Our examination of the filtering lists in these extensions showed that these extensions are not specifically designed to identify cryptojacking websites. Rather, they primarily look for addresses of known mining pool proxies (e.g., `xmrpool.net`). As shown in Table 4, OutGuardhad approximately 40% more coverage than MinerBlock and 41.5% more coverage than AntiMiner.

While disrupting the communication between the cryptojacking libraries and the mining pools is an effective mitigation technique, this approach is less likely to be very successful in identifying different variants of cryptojacking families. For example, the naive versions of cryptojacking families make queries to a specific address that can easily be detected and blocked. However, we observed several cases of Crypto-loot, JSECoin, and CoinImp, which use legitimate cloud deployment services (e.g., `now.sh`) for this purpose. Unsurprisingly, it is impractical to block a domain name or IP address that belongs to cloud services without affecting a large number of other legitimate applications. Consequently, these extensions are limited to domains that do not map to cloud services, and can therefore be evaded fairly easily. We posit that a solution such as OutGuardthat monitors the dynamic behavior of websites and JavaScript libraries is more likely to prevent cryptojacking incidents in practice. With that being said, these dedicated security solutions can benefit from using OutGuardby automatically updating their corresponding filter lists.

5 CHARACTERIZING CRYPTOJACKING WEBSITES

Armed with the results of OutGuard's deployment in the wild, we first seek to answer the question of where cryptojacking occurs. Reports of cryptojacking websites have painted a myriad landscape of sites infected via web vulnerabilities [36], popular websites with cryptojacking advertisements [13], and suspicious e-commerce websites [15]. We will now take a comprehensive look at the 6,302 detected cryptojacking websites, investigating their popularity, categories, and lifetimes.

5.1 Website Popularity and Type

We look up each cryptojacking website's Alexa ranking as a metric for the relative number of visits for each cryptojacking website. 220 websites (3.5%) fall within the top 100K, 602 (9.6%) appear between the top 100K and top 1M, and the remaining 5,480 (87%) are outside of the top 1M. This suggests that cryptojacking occurs primarily in the long tail of lower-popularity websites, as hypothesized in Section 3.1 during the discussion of training data construction.

In order to better address the 5,480 sites outside of the Alexa Top 1M, we also explored passive DNS data as a means to estimate the relative popularity of the cryptojacking domains. Through the lens of one of the largest U.S. ISPs, we collected the contents and daily lookup volume of DNS resource records (RRs) for all successful DNS resolutions. We extracted all passive DNS RRs that match the

Table 4: Number of cryptojacking incidents detected by Outguard. 3,600 (57%) of the detected cryptojacking websites were new detections.

Results	Experiment 1	Experiment 2	Total
Target crawl size	Alexa 1 Million	Alexa 600K	-
URLs visited	3,798,433	1,329,684	5,128,117
Unique cryptojacking domains	5,873	429	6,302
New detections	3,171	429	3,600 (57%)
Overlap with MinerBlock	-	-	3,776 (60%)
Overlap with AntiMiner	-	-	3,684 (58.5%)

seed list of 5,873 cryptojacking domains from the initial Alexa Top 1M real-world deployment and also included RRs with subdomains of the seed list. This dataset consisted of 595.6 million RRs collected from April 2016 to March 2018.

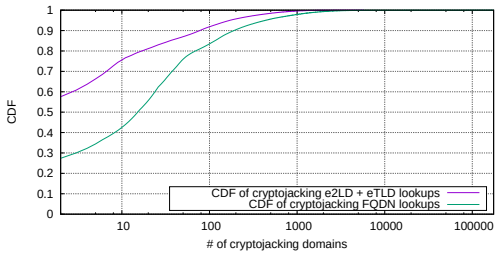


Figure 4: Distribution of passive DNS lookups for cryptojacking FQDNs and e2LD+eTLD. By both metrics, `coinhive.com` accounts for the most lookups, 21.5% and 51.7%, respectively.

We constructed the initial seed domain list for passive DNS data collection from websites that have embeded cryptojacking; however, some domains and their subdomains (e.g. `coinhive.com`) are also used for cryptojacking delivery servers or mining pool/proxy servers that coordinate cryptojacking miners. This muddles the passive DNS dataset with multiple use cases, but grouping passive DNS by effective second-level domain (e2LD) and effective TLD (eTLD), as shown in Table 5, makes it easier to distinct between cryptojacking websites and cryptojacking delivery servers or mining pools.

To characterize the ecosystem of websites that have embeded cryptojacking, we categorized the top 100 e2LD+eTLDs, which accounted for 92% of all passive DNS lookups. Granted, domains dually purposed to be both cryptojacking websites and cryptojacking administration sites have inflated lookup counts, so we examined only the number of domains for each category and did not evaluate the lookup-weighted category composition. In Table 6 we found that the top cryptojacking domains were typically sites with illicited content such as torrent hosting sites, video download/streaming sites, and malicious websites. Assuming that most of these cryptojacking incidents were not the result of website compromise, these results suggest that cryptojacking is used by criminal actors more successfully than lawful actors.

One particularly interesting mechanism for cryptojacking delivery is inclusion through an advertisement on a website through an embedded iframe. Through this form of indirect delivery, cryptojacking can achieve broad distribution, as demonstrated by DoubleClick's reported inclusion of cryptojacking on Youtube [13].

While we did not detect cryptojacking directly included in advertisements, we did find 69 cryptojacking incidents in landing pages that resulted from clicking on an advertisement. We measured these occurrences by cross-referencing the provenance of websites derived from clicked links with a list of ad domains provided by EasyList (`easylist.to`) and Ghostery (`www.ghostery.com`). We observed several instrances of *crypto–loot* and *JSECoin* libraries in our dataset. These instances were mainly delivered via specific ad-networks such as `adnxs.com` (33%),`spotxchange.com` (23.1%), and `stickyads.com` (14.5%). While the number of advertisement-based cryptojacking incidents is relatively small, they represent a fundamentally different type of cryptojacking: *unintentional* cryptojacking incidents that are dynamically loaded during page loads.

5.2 Website Lifetime

We continuously monitored all cryptojacking websites from our first deployment, which observed 5,873 websites by crawling them every ten days for 3 months to approximate their active lifetime. We labeled a cryptojacking website as inactive if we did not observe any sign of cryptojacking operations in a given trace file. We performed 9 crawls over 3 months from May 20 to July 25, 2018 and found that 32% of the cryptojacking websites became inactive in less than 10 days. These websites were mainly hosted on a set of cloud servers and were likely managed by the same group of operators. While we observed a steady decrease in the number of active cryptojacking operations over time, there were 619 (10.5%) websites—mostly video streaming, gaming, and application download sites—that stayed active for the entire measurement period. An additional 1,552 (26.4%) websites hosted cryptojacking libraries for a moderate duration of 3–9 weeks. In total, 68 (3.1%) of the 2,171 active cryptojacking websites were among the Alexa Top 100K websites, which impact thousands of users. In fact, this analysis suggests that current preventions solutions are unable or unwilling to blacklist active cryptojacking websites including the ones with higher ranking for several weeks.

6 OPERATIONAL INSIGHTS

As a second case study, we dig into the operations of cryptojacking. We cluster the identified cryptojacking families, reveal previously unknown CoinHive-like services, extract mining account identifiers to group cryptojacking incidents. This aggregated data begins to

Table 5: Grouping the top passive DNS lookups by FQDN and e2LD+eTLD highlights that the expanded passive DNS dataset includes two types of domains: cryptojacking administration domains (*) and cryptojacking website domains(†).

FQDN	Lookups (%)	e2LD+eTLD	Lookups (%)
coinhive.com*†	128.31M (21.5%)	coinhive.com*†	308.09M (51.7%)
cnhv.co*†	34.83M (5.8%)	cnhv.co*†	34.83M (5.8%)
moonbit.co.in†	15.53M (2.6%)	adsptp.com†	19.10M (3.2%)
www.adsptp.com†	13.60M (2.3%)	watchfree.to†	17.30M (2.9%)
1480876790.rsc.cdn77.org†	12.87M (2.2%)	moonbit.co.in†	15.61M (2.6%)
www.rcyclmnr.com†	12.24M (2.1%)	rcyclmnr.com†	13.41M (2.3%)
www.alluc.ee†	9.35M (1.6%)	cdn77.org†	12.87M (2.2%)
www.watchfree.to†	9.12M (1.5%)	alluc.ee†	12.10M (2.0%)
ws008.coinhive.com*	8.69M (1.5%)	crypto-loot.com*†	10.18M (1.7%)
ws013.coinhive.com*	8.68M (1.5%)	recycloped.com†	6.77M (1.3%)
ws014.coinhive.com*	8.68M (1.5%)	pornxs.com†	6.51M (1.1%)
ws012.coinhive.com*	8.67M (1.5%)	faucethub.io†	3.86M (0.6%)
ws011.coinhive.com*	8.67M (1.5%)	baiduccdn1.com†	3.85M (0.6%)
ws009.coinhive.com*	8.67M (1.5%)	graaam.com†	3.83M (0.6%)
ws010.coinhive.com*	8.67M (1.5%)	mycashbar.com†	3.44M (0.6%)
ws030.coinhive.com*	8.39M (1.4%)	otohits.net†	2.67M (0.4%)
www.recycloped.com*	6.77M (1.1%)	jsecoin.com*†	2.53M (0.4%)
ws020.coinhive.com*	6.73M (1.1%)	kickass.cd†	2.23M (0.4%)
ws016.coinhive.com*	6.69M (1.1%)	oyunuoyna.com†	2.18M (0.4%)
ws018.coinhive.com*	6.69M (1.1%)	xxgasm.com†	2.14M (0.4%)
175,202 others	263.83M (44.3%)	*5,807 others*	112.19M (18.8%)
Total	595.67M (100%)	*Total*	595.67M (100%)

Table 6: Categories for the top 100 most popular cryptojacking e2LD+eTLDs.

Category	#	Category	#
torrent	21	software	2
video	15	forum/chatroom	2
unknown	11	education	2
malicious	9	crypto faucet	2
porn	7	pyrotechnics	1
news	7	mining pool	1
cryptojacking	7	manga	1
gaming	4	image upload	1
music	3	automobile	1
traffic exchange	2	advertising	1

reveal the higher-level behavior of the individual and organizational actors that run cryptojacking.

6.1 The Prevalence of Cryptojacking Families

To identify cryptojacking families that each cryptojacking website belongs to, we analyzed the downloaded wasm modules to see whether we can link cryptojacking families together. Although the names of wasm modules might vary for different cryptojacking operators, the content of the wasm module is less likely to differ for a given family. This suggests a relatively stagnant set of wasm binaries, which are typically compiled from C/C++. To assign a label to a given website, we considered two features: (1) the address of the server that hosted the cryptojacking library (similar to Wappalyzer), and (2) the hash of the wasm module used by the library. As displayed in Table 7, we found that these two metrics were sufficient to label 5816 (92.3%) of the detected websites into the 14 known cryptojacking families encountered during training data labeling. We utilized the raw number of discovered cryptojacking domains for each cryptojacking family as a proxy for the family's market share and found a broad spectrum of cryptojacking families with significant deployment. There is no single dominant entity, with the top 3 libraries, CoinHive, JSECoin, and Crypto-loot, accounting for 60.7% of the market. In total, 50% of all cryptojacking sites used non-standard hosts for their cryptojacking libraries, suggesting that the hosting server regexes found in certain blacklists are inherently reactive in their detection capabilities, and even if updated regularly, they can be easily thwarted.

We performed another experiment on the remaining 486 (7.7%) cryptojacking incidents and clustered them based on the address of the remote server that hosted the cryptojacking library. For example, Moonify (e.g., `moonify.min.js`) or Grindcash (`gridcash.js`) are CoinHive-like mining services which were not in our training dataset, but OUTGUARDidentified instances of these libraries in our large-scale experiment. We identified 24 previously unseen coinhive like mining services. Table 8 shows a subset of these mining services as well as the main mining pool.

Table 7: The number of cryptojacking websites and their associated cryptojacking families in the Alexa Top 1 Million websites in 50% of the cases, the cryptojacking library was loaded from a remote server.

Cryptojacking Family	Occurrences	Dynamically Loaded
CoinHive	2,206 (35%)	227 (10.3%)
JSEcoin	1,271 (20.2%)	839 (66%)
Crypto-loot	766 (12.2%)	737 (96.3%)
ProsectPoi	348 (5.5%)	228 (65.5%)
CoinHave	321 (5.1%)	170 (53%)
CoinImp	227 (3.6%)	207 (91%)
CoinLab	202 (3.2%)	202 (100.0%)
DeepMiner	176 (2.8%)	162 (92%)
WebMine	139 (2.2%)	111 (80%)
Cloudcoins	57 (0.9%)	43 (76%)
MoneroMiner	53 (0.85%)	7 (13%)
CoinHive Captcha	28 (0.45%)	4(13%)
Inwemo	12 (0.19%)	11 (97%)
Monero.cc	10 (0.15%)	10 (100%)
Previously-Unknown Libraries	486 (7.7%)	214 (44%)
Total	6,302 (100%)	3,172(50%)

Table 8: A subset of previously unknown cryptojacking services. 486 cryptojacking incidents were conducted by 24 new CoinHive-like services.

New Mining Servies	Occurrences	Main Mining Pool
WebXMR	208	xmrm.pw
CoinPot	81	coinpot.co
Coinblind	32	coinblind.com
Punchhub	21	punchsub.net
VidMiner	18	estream.nu
AJCryptominer	15	ajcryptominer.com
CoinNebula	11	coinnebula.com
Ad-miner	11	ad-miner.com
Cryptonoter	9	cryptonoter.com
Lightminer	8	lightminer.co
gridcash	8	gridcash.net
DigXMR	8	digxmr.com
Minercry	7	minercry.pt
Moonify	7	moonify.io
Grindcash	5	ulnawoyyzbljc.ru

6.2 Campaign Analysis

Our analysis show that obtaining the *Site Key* is possible by searching in network traces and JS source code. While mining services allow generating several Site Keys per user, in practice, we found several instances of individual Site Keys being used on multiple websites simultaneously. We found 386 cryptojacking websites under 35 clusters based on unique Site Keys ranging in size from 2 domains to 121 domains. Table 9 illustrates a subset of these Site Keys. Our analysis shows that 16 out of 35 detected campaigns were using less expensive or free CoinHive-like mining services.

To get a flavor for the behavior of a cryptojacking campaign, we took a closer look at the largest CoinHive campaign. The 121 clustered domains were all registered under anonymous WHOIS services and pointed to a set of 23 IP addresses found in 15 ASes, which were mostly cloud-hosting services. The IPs were globally distributed in 7 countries including the US, Germany, Australia, Great Britain, China, Canada, and Switzerland. From Censys.io [3] we were able to determine that all IPs pointed to HTTP webservers running either openresty, nginx, or apached. The campaign's usage of multiple anonymous WHOIS services, globally distributed cloud hosting, and large number of fungible, human-meaningless domains (e.g. 5565925.com, 2zo.xyz, jipa.gdn,etc.) hint at evasion techniques characteristic of other web-based malware or PUPs.

7 DISCUSSION AND LIMITATIONS

In this paper, we construct and evaluate Outguard, a resilient system for detecting cryptojacking that finds nearly twice as many cryptojacking websites as existing blacklist solutions. As we demonstrated, a large fraction of websites have short-lived cryptojacking operations. Therefore, regular scans of the Alexa Top 1M or other domain lists using a system similar to Outguardcan automatically update blacklists with the latest cryptojacking domains and remove stale domains.

While the evaluation of Outguardon real-world dataset shows that it works well in practice, it has three potential limitations. First, Outguardis subject to bias due to the degree of completeness of the public blacklists. Like other machine learning approaches, the effectiveness of Outguarddepends on the quantity of the dataset that we used to generate the model. The hash function detection mechanism currently relies on static function name/fingerprints that can be circumvented by an evasive cryptojacking operator. We expect this to have limited impact based on our feature ranking

Table 9: A subset of cryptojacking campaigns and their corresponding mining pools. 16 out of 35 detected cryptojacking campaigns were using less expensive or free CoinHive-like mining services.

Campaign ID	No. of Sites	Main Mining service
pfr5E2eLd0VduZ1wKaesPFnccU9d2GZi	121	coinhive.com
Yq2af5ZaYuELhmPUH524q7sjiaCNCzXr	117	cloudfront.net
FGZZeDbNg5AMpILuz8fKWI4UJqdNIxce	23	xmrpool.xyz
l8Un4BkrkTfKQOEoCRyTtGMfhJFlibNu	16	semipool.com
XcgzWRYl8Lh3r3ElNSznHhP9HAcYtd4g	13	minemonero.pro
Dx8uX9lNo7a3kmGe55vO6bhWMquRd9YG	10	supportxmr.com
15217	7	aalbbh84.info
b7fbc3b2e3d88d9d766dc5ed27db53983ec759be76e2	5	crypto-loot.com
14285	4	bhzejltg.info

analysis. In general, OUTGUARDshould be viewed as a lower bound estimate of the the full cryptojacking ecosystem.

Second, note that OUTGUARDis a supervised learning technique. Cryptojacking operators may be able exploit this fundamental limitation and change their strategies to evade some of the features of the detection model. For example, the operators may find alternatives to using web sockets or web worker services. We plan to explore the practicality and profitability costs of these evasions to cryptojacking operators in future work. While not foolproof, OUTGUARDis likely to detect a large number of evasion scenarios as shown in Section 3.5. We believe that OUTGUARDraises the bar for adversaries to avoid cryptojacking detection and increases the development cost required to carry out these operations.

Finally, as mentioned earlier, cryptojacking occurs in many forms beyond in-browser mining. In this paper, we focused on in-browser cryptojacking as it is very easy for cryptojacking operators to develop and deploy. In contrast to other common attacks, in-browser cryptojacking does not require building specific binaries or creating drive-by download websites, which are more likely to be flagged by modern anti-malware solutions. Other forms of cryptojacking operations, such as cryptojacking browser extensions, desktop software, and mobile apps, are outside the scope of this study.

8 CONCLUSION

We find that in-browser cryptojacking libraries use browser resources in a very similar way. We show that if these traits are accurately modeled, it is possible to automatically identify a large number of in-browser cryptojacking operations. Towards that end, we developed a tool called OUTGUARD, which is able to detect a significant number of cryptojacking websites and potentially disrupt the malicious operations of cryptojacking operators. Consequently, we conclude that defending against cryptojacking operations at scale is possible with minimal human intervention.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation (NSF) under grant CNS-1518741 award. We would like to thank the anonymous reviewers for their helpful comments.

REFERENCES

[1] [n. d.]. CoinBlockerLists. https://github.com/ZeroDot1/CoinBlockerLists. Accessed: 07-29-2018.
[2] [n. d.]. Stratum mining protocol. https://en.bitcoin.it/wiki/Stratum_mining_protocol. Accessed: 07-29-2018.
[3] 2018. Censys. https://censys.io/.
[4] 2018. MinerBlock Extension. https://github.com/xd4rker/MinerBlock/blob/master/assets/filters.txt.
[5] Sherly Abraham and InduShobha Chengalur-Smith. 2010. An overview of social engineering malware: Trends, tactics, and implications. *Technology in Society* 33 (2010), 188–196.
[6] David Andersen. 2014. Mining Money with Monero ... and CPU vector intrinsics. https://da-data.blogspot.com/2014/08/minting-money-with-monero-and-cpu.html.
[7] Sajjad Arshad, Amin Kharraz, and William Robertson. 2016. Include Me Out: In-Browser Detection of Malicious Third-Party Content Inclusions. In *Proc. 20th International Conference on Financial Cryptography and Data Security (FC)*. 441–459.
[8] Juan Caballero, Chris Grier, Christian Kreibich, and Vern Paxson. 2011. Measuring Pay-per-install: The Commoditization of Malware Distribution. In *Proceedings of the 20th USENIX Conference on Security (SEC'11)*. USENIX Association, Berkeley, CA, USA, 13–13.
[9] Robert B. Cialdini. 2009. *Influence: Science and Practice.* Writers of the Round Table Press, Boston.
[10] Catalin Cimpanu. 2017. Cryptojackers Found on Starbucks WiFi Network, GitHub, Pirate Streaming Sites. https://www.bleepingcomputer.com/news/security/cryptojackers-found-on-starbucks-wifi-network-github-pirate-streaming-sites/.
[11] CoinMarketCap. 2018. Top 100 Cryptocurrencies by Market Capitalization. https://coinmarketcap.com/.
[12] CryptoMineDev. 2018. MinerBlock. https://chrome.google.com/webstore/detail/minerblock/emikbbbebcdfohonlaifafnoanocnebl?hl=en.
[13] Dan Goodin. 2018. Now even YouTube serves ads with CPU-draining cryptocurrency miners. https://arstechnica.com/information-technology/2018/01/now-even-youtube-serves-ads-with-cpu-draining-cryptocurrency-miners/.
[14] David Maciejak. 2017. Cryptojacking: Digging for Your Own Treasure. https://www.fortinet.com/blog/threat-research/cryptojacking-digging-for-your-own-treasure.html.
[15] Willem de Groot. 2017. Cryptojacking found on 2496 online stores. https://gwillem.gitlab.io/2017/11/07/cryptojacking-found-on-2496-stores/.
[16] Shayan Eskandari, Andreas Leoutsarakos, Troy Mursch, and Jeremy Clark. 2018. A first look at browser-based Cryptojacking. *CoRR* (2018).
[17] Google Chromium. 2017. Chrome debugging protocol viewer. https://chromedevtools.github.io/devtools-protocol/.
[18] Google Development. 2018. Extending DevTools. https://developer.chrome.com/extensions/devtools.
[19] Hamed Haddadi. 2010. Fighting online click-fraud using bluff ads. *ACM SIGCOMM Computer Communication Review* 40 (2010), 21–25.
[20] Geng Hong, Zhemin Yang, Sen Yang, Lei Zhang, Yuhong Nan, Zhibo Zhang, Min Yang, Yuan Zhang, Zhiyun Qian, and Haixin Duan. 2018. How You Get Shot in the Back: A Systematical Study about Cryptojacking in the Real World. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1701–1713.
[21] Danny Yuxing Huang, Hitesh Dharmdasani, Sarah Meiklejohn, Vacha Dave, Chris Grier, Damon McCoy, Stefan Savage, Nicholas Weaver, Alex C Snoeren, and Kirill Levchenko. 2014. Botcoin: Monetizing Stolen Cycles. In *Proc. Network and Distributed System Security Symposium (NDSS)*.
[22] Product Hunt. 2018. Inwemo. https://www.producthunt.com/posts/inwemo.

[23] Ian Hickson. 2009. W3C Working Draft: Web Workers. https://www.w3.org/TR/2009/WD-workers-20090423/.

[24] Jerome Segura. 2018. The state of malicious cryptomining. https://blog.malwarebytes.com/cybercrime/2018/02/state-malicious-cryptomining/.

[25] Amin Kharraz, Sajjad Arshad, Collin Mulliner, William Robertson, and Engin Kirda. 2016. UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware. In *Proceeding of 25th USENIX Security Symposium*. 192–203.

[26] Amin Kharraz, Engin Kirda, William Robertson, Davide Balzarotti, and Aurelien Francillon. 2014. Optical Delusions: A Study of Malicious QR Codes in the Wild. In *Proceeding of 44th International Conference of IEEE/IFIP Dependable Systems and Networks (DSN)*. 192–203.

[27] Amin Kharraz, William Robertson, Davide Balzarotti, Leyla Bilge, and Engin Kirda. 2015. Cutting the gordian knot: A look under the hood of ransomware attacks. In *Proceeding of 12th International Conference of Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*. 3–24.

[28] Amin Kharraz, William Robertson, and Engin Kirda. 2018. Surveylance: Automatically Detecting Online Survey Scams. In *Proceeding of IEEE Symposium on Security and Privacy (S&P)*. 70–86.

[29] Eugene Kolodenker, William Koch, Gianluca Stringhini, and Manuel Egele. 2017. PayBreak: Defense Against Cryptographic Ransomware. In *Proceeding of ACM on Asia Computer and Communications Security (Asia CCS)*. 599–611.

[30] Radhesh Krishnan Konoth, Emanuele Vineti, Veelasha Moonsamy, Martina Lindorfer, Christopher Kruegel, Herbert Bos, and Giovanni Vigna. 2018. MineSweeper: An In-depth Look into Drive-by Cryptocurrency Mining and Its Defense. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1714–1730.

[31] Platon Kotzias, Leyla Bilge, and Juan Caballero. 2016. Measuring PUP Prevalence and PUP Distribution through Pay-Per-Install Services.. In *Proceeding of 25th USENIX Security Symposium*.

[32] Marc Kührer, Christian Rossow, and Thorsten Holz. 2014. Paint It Black: Evaluating the Effectiveness of Malware Blacklists. In *Proceeding of 17th International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*. 1–21.

[33] Bum Jun Kwon, Jayanta Mondal, Jiyong Jang, Leyla Bilge, and Tudor Dumitras. 2015. The dropper effect: Insights into malware distribution with downloader graph analytics. In *Proceeding of 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 1118–1129.

[34] Zhou Li, Kehuan Zhang, Yinglian Xie, Fang Yu, and XiaoFeng Wang. 2012. Knowing your enemy: understanding and detecting malicious web advertising. In *Proceedinf of 2012 ACM SIGSAC conference Computer and Communications Security (CCS)*. 674–686.

[35] Lindsey O'Donnell. 2018. Cryptojacking Attack Found on Los Angeles Times Website. https://threatpost.com/cryptojacking-attack-found-on-los-angeles-times-website/130041/.

[36] Lindsey O'Donnell. 2018. Cryptojacking Campaign Exploits Drupal Bug, Over 400 Websites Attacked. https://threatpost.com/cryptojacking-campaign-exploits-drupal-bug-over-400-websites-attacked/131733/.

[37] Chaoying Liu and Joseph C. Chen. 2018. Malvertising Campaign Abuses GoogleâĂŹs DoubleClick to Deliver Cryptocurrency Miners. https://blog.trendmicro.com/trendlabs-security-intelligence/malvertising-campaign-abuses-googles-doubleclick-to-deliver-cryptocurrency-miners/.

[38] Malwarebytes Inc. 2017. A look into the global drive-by cryptocurrency mining phenomenon . https://go.malwarebytes.com/rs/805-USG-300/images/Drive-by_Mining_FINAL.pdf.

[39] Najmeh Miramirkhani, Oleksii Starov, and Nick Nikiforakis. 2017. Dial One for Scam: A Large-Scale Analysis of Technical Support Scams. In *Proceeding of Network and Distributed System Security Symposium (NDSS)*.

[40] Marius Musch, Christian Wressnegger, Martin Johns, and Konrad Rieck. 2018. Web-based Cryptojacking in the Wild. *arXiv preprint arXiv:1808.09474* (2018).

[41] Satoshi Nakamoto. 2009. Bitcoin: A peer-to-peer electronic cash system. http://www.bitcoin.org/bitcoin.pdf.

[42] Terry Nelms, Roberto Perdisci, Manos Antonakakis, and Mustaque Ahamad. 2016. Towards Measuring and Mitigating Social Engineering Software Download Attacks. In *Proceedings of 25th USENIX Security Symposium*.

[43] Lily Hay Newman. 2018. Hack Brief: Hackers Enlisted Tesla's Public Cloud To Mine Cryptocurrency. https://www.wired.com/story/cryptojacking-tesla-amazon-cloud/.

[44] Patrick Traynor Nolen Scaife, Henry Carter and Kevin Butler. 2016. CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*.

[45] Rafael Keramidas. 2018. No Coin. https://github.com/keraf/NoCoin/blob/master/src/js/background.js.

[46] Jan Rüth, Torsten Zimmermann, Konrad Wolsing, and Oliver Hohlfeld. 2018. Digging into Browser-based Crypto Mining. *arXiv preprint arXiv:1808.00811* (2018).

[47] scikit-learn. 2018. scikit-learn: machine learning in Python. https://github.com/scikit-learn/scikit-learn.

[48] Kurt Thomas, Juan A. Elices Crespo, Ryan Rasti, Jean-Michel Picod, Cait Phillips, Marc-André Decoste, Chris Sharp, Fabio Tirelo, Ali Tofigh, Marc-Antoine Courteau, Lucas Ballard, Robert Shield, Nav Jagpal, Moheeb Abu Rajab, Panayiotis Mavrommatis, Niels Provos, Elie Bursztein, and Damon McCoy. 2016. Investigating Commercial Pay-Per-Install and the Distribution of Unwanted Software. In *USENIX Security Symposium*.

[49] tunghobrens. 2018. AntiMiner – No 1 Coin MinerBlock. https://chrome.google.com/webstore/detail/anti-miner-no-1-coin-mine/ibhpgkhoicjhklmbhdoeikeggbeejonj?hl=en.

[50] Jordan Tuwiner. [n. d.]. Bitcoin Mining Pools. https://www.buybitcoinworldwide.com/mining/pools/. accessed: 01-29-2019.

[51] Phani Vadrevu, Babak Rahbarinia, Roberto Perdisci, Kang Li, and Manos Antonakakis. 2013. Measuring and detecting malware downloads in live network traffic. In *Proceeding of European Symposium on Research in Computer Security*. Springer, 556–573.

[52] Wappalyzer. [n. d.]. MinerBlock. https://www.wappalyzer.com/download. accessed: 01-23-2019.

[53] Barton Whaley. 1982. Toward a general theory of deception. *The Journal of Strategic Studies* 5 (1982), 178–192.

[54] World Wide Web Consortium. 2018. WebAssembly Specifications. https://webassembly.github.io/spec/.

[55] James Wyke. 2012. ZeroAccess Botnet – Mining and Fraud for Massive Financial Gain. https://www.sophos.com/en-us/medialibrary/PDFs/technical%20papers/Sophos_ZeroAccess_Botnet.pdf.

[56] Apostolis Zarras, Alexandros Kapravelos, Gianluca Stringhini, Thorsten Holz, Christopher Kruegel, and Giovanni Vigna. 2014. The dark alleys of madison avenue: Understanding malicious advertisements. In *Proceeding of 2014 ACM Internet Measurement Conference (IMC)*. 373–380.

From Small-scale to Large-scale Text Classification

Kang-Min Kim
Korea University
Seoul, South Korea
kangmin89@korea.ac.kr

Yeachan Kim
Korea University
Seoul, South Korea
yeachan@korea.ac.kr

Jungho Lee
Korea University
Seoul, South Korea
leejungho@korea.ac.kr

Ji-Min Lee
Korea University
Seoul, South Korea
jm94318@korea.ac.kr

SangKeun Lee
Korea University
Seoul, South Korea
yalphy@korea.ac.kr

ABSTRACT

Neural network models have achieved impressive results in the field of text classification. However, existing approaches often suffer from insufficient training data in a large-scale text classification involving a large number of categories (e.g., several thousands of categories). Several neural network models have utilized multi-task learning to overcome the limited amount of training data. However, these approaches are also limited to small-scale text classification. In this paper, we propose a novel neural network-based multi-task learning framework for large-scale text classification. To this end, we first treat the different scales of text classification (i.e., large and small numbers of categories) as multiple, related tasks. Then, we train the proposed neural network, which learns small- and large-scale text classification tasks simultaneously. In particular, we further enhance this multi-task learning architecture by using a gate mechanism, which controls the flow of features between the small- and large-scale text classification tasks. Experimental results clearly show that our proposed model improves the performance of the large-scale text classification task with the help of the small-scale text classification task. The proposed scheme exhibits significant improvements of as much as 14% and 5% in terms of micro-averaging and macro-averaging F1-score, respectively, over state-of-the-art techniques.

CCS CONCEPTS

• **Information systems → Clustering and classification**; • **Computing methodologies → Multi-task learning**; **Neural networks**.

KEYWORDS

Large-scale Text Classification; Multi-task Learning; Deep Neural Networks

ACM Reference Format:
Kang-Min Kim, Yeachan Kim, Jungho Lee, Ji-Min Lee, and SangKeun Lee. 2019. From Small-scale to Large-scale Text Classification. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313563

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313563

1 INTRODUCTION

Large-scale text classification seeks to classify arbitrary texts into a few semantically relevant classes or categories. There is a wide range of applications of large-scale text classification, including web search personalization [6], contextual advertising [5, 23], topical web search [4], recommender systems [2, 33], etc. The success of these applications is highly dependent on the quality of text classification. Usually, large-scale text classification requires a sufficiently large taxonomy of topical categories to capture various topics in arbitrary texts. In addition, it is necessary to collect a large amount of training data for each category in the taxonomy.

Although neural network models have shown promising results on text classification tasks [17, 21, 42], they perform poorly in large-scale text classification [16], which is often involved in a few or several thousands of categories. This is largely attributed to the fact that the training data for each category is relatively insufficient and distributed unevenly among classification categories.

To handle the large-scale text classification, several works [23, 34] have represented categories in large-scale taxonomy by using an explicit representation model [38] based on bag-of-words or bag-of-phrases. They exhibit a robust performance in the large-scale text classification. One advantage of the explicit representation model used in these studies [23, 34] is that it easily utilizes the taxonomy structure to share training data among categories with "is-a" relationships. However, it may be difficult to apply this method in a neural network that implicitly represents text. In addition, the performance of these approaches is limited to associated knowledge bases.

In the neural network-based text classification, several studies often utilize a multi-task learning framework with other related classification tasks to cope with the limited amount of training data [24–26, 39, 41]. Multi-task learning learns multiple related tasks simultaneously to extract common features and significantly improves the performance of a single task. One characteristic of these multi-task architectures is they share some lower layers to catch common features. Non-shared layers, the remaining layers are split into multiple specific tasks.

In this paper, we propose a multi-task learning framework based on the convolutional neural network (CNN) to handle the large-scale text classification. We apply multi-task learning by treating

Table 1: Statistics of taxonomies

	Original ODP taxonomy	Large-scale taxonomy	Small-scale taxonomy
No. categories	802,379	2,735	13
No. webpages	3,624,444	76,167	1,032,411
Avg. webpages	4.5	27.8	79,416
Min. webpages	1	2	18,021
Max. webpages	1,276	915	171,955

small- and large-scale text classification tasks as two related tasks. The multi-task framework converts the large-scale text classification task to a small-scale text classification task and jointly learns both tasks. This multi-task learning framework with different scales is motivated by our hypothesis that the useful features learned from the small-scale classification task are highly expected to be useful for the large-scale classification task. In particular, the large-scale classification task is further enhanced by task-dependent features from the small-scale classification task through a gate mechanism.

We demonstrate the efficacy of our model on a group of text classification tasks with different scales. Experimental results show that the joint learning of small- and large-scale classification yields significantly improved results in the large-scale text classification task with the help of the small-scale text classification task. In summary, our contributions are three-fold:

- We propose a convolutional neural network (CNN) based multi-task learning framework to handle the large-scale text classification, which shares useful features between small- and large-scale classification.
- We incorporate a gate mechanism into the proposed CNN-based multi-task learning, which helps to selectively utilize useful features of a small-scale classification task in large-scale classification task.
- We demonstrate the efficacy of the proposed methodology through extensive experiments. The performance evaluation clearly shows that our approach significantly outperforms strong baselines in terms of micro- and macro-averaging F1-score.

The remainder of this paper is organized as follows. We briefly describe the knowledge taxonomy and convolutional neural networks for text classification in Section 2. Section 3 describes the multi-task learning framework for large-scale text classification. We present the performance evaluation results and in-depth analysis in Sections 4 and 5, respectively. We discuss related work in Section 6 and conclude the paper in Section 7.

2 PRELIMINARY

2.1 Building Taxonomy

For the large-scale text classification, the first step of our methodology is to build the taxonomy of topics by utilizing the Open Directory Project (ODP) [1]. The ODP is a large-scale and tree-structured web directory with a maximum of 15 levels, where approximately 4 million webpages are classified into 0.8 million categories by volunteer editors. We utilize only subsets of ODP categories rather than entire sets of them to build the large-scale taxonomy, since we observe that there are many useless or too-specific categories. Specifically, we apply three heuristic rules suggested in the work [23] to remove those categories. To construct the small-scale taxonomy, we use only top-level categories from the large-scale taxonomy.

The large-scale taxonomy is well organized to classify arbitrary documents into various topics (e.g., *Top/Sports/Baseball/Major_League/Teams/Los_Angeles_Dodgers/News_and_Media*). However, the training data for each category in large-scale taxonomy is relatively insufficient and distributed unevenly among categories. On the other hand, the small-scale taxonomy represents highly abstracted topics (*Top/Sports*, *Top/Health*, *Top/Computers*, *Top/Business*, etc.), but has sufficient training data (i.e., containing the training data of all descendant categories from the top-level category to the leaf category) whose distribution is relatively balanced. Table 1 shows the statistics of taxonomies.

2.2 Convolutional Neural Network for Text Classification

For the text classification, many studies have utilized neural network models, such as recurrent neural networks [15] and convolutional neural networks [17, 37, 42]. In particular, long short-term memory network (LSTM) [12], which is a type of recurrent neural networks, has achieved impressive results on various natural language processing tasks [13, 28, 36]. However, due to the large number of parameters, LSTM usually requires sufficient training data. Therefore, LSTM may perform poorly in large-scale text classification where training data is insufficient. We adopt convolutional neural networks (CNN) which have been very successful on text classification tasks [17, 37, 42] and need relatively small parameters. The component of CNN-based text classifier usually consists of four layers (i.e., embedding layer, convolution layer, pooling layer and fully connected layer).

2.2.1 Embedding Layer. The embedding layer transforms the document into a vector of embeddings, denoted as $\mathbf{w}_{1:n} \in \mathbb{R}^{nk}$ as the input of the network, where n and k are the number of words and the dimension of word embedding, respectively. A document of length n is represented as follows:

$$\mathbf{w}_{1:n} = \mathbf{w}_1 \oplus \mathbf{w}_2 \oplus \ldots \oplus \mathbf{w}_n \tag{1}$$

where $\oplus$ is the concatenation operator. $\mathbf{w}_i \in \mathbb{R}^k$ is the word vector corresponding to the i-th word in the document. We insert zero-paddings to the both sides of these word representations to handle variable-sized documents and circumvent the border effect [8].

[1] https://curlie.org

2.2.2 Convolution Layer. The function of the convolution layer is to extract features from the input. Using a filter $\mathbf{f} \in \mathbb{R}^{hk}$, a convolution operation on h consecutive word vectors starting from i-th produces a feature. For example, a feature c_i is generated from a window of words $\mathbf{w}_{i:i+h-1}$ as follows:

$$c_i = \mathbf{ReLU}(\mathbf{f} \cdot \mathbf{w}_{i:i+h-1} + \mathbf{b}) \tag{2}$$

where $\mathbf{ReLU}(\cdot)$ is the element-wise rectified linear unit function [31] and $\mathbf{b} \in \mathbb{R}$ is a bias term. Repeating the convolution operations for each possible window of words in the document $\mathbf{w}_{1:h}, \mathbf{w}_{2:h+1}, ..., \mathbf{w}_{n-h+1:n}$, we obtain a feature map $\mathbf{c}$ as follows:

$$\mathbf{c} = [c_1, c_2, ..., c_{n-h+1}] \tag{3}$$

2.2.3 Pooling Layer. The function of the pooling layer is to further abstract the features $\mathbf{c} \in \mathbb{R}^{n-h+1}$ by aggregating the scores for each filter. There are two choices for the pooling operation, i.e., average and max. In this paper, we apply a max-over-time pooling operation [8] over the feature map and take the maximum value $\hat{\mathbf{c}} = max\{\mathbf{c}\}$ as the feature corresponding to this particular filter. The idea behind max pooling is to capture the most important feature, which has the highest value, for each feature map. With pooling layers, we induce a fixed-length vector from feature maps. To obtain multiple features, we use multiple filters with different window sizes. For m different filters, we simply concatenate the results of each filter and obtain the output vector of the pooling layer $\mathbf{z} \in \mathbb{R}^m$ as follows:

$$\mathbf{z} = [\hat{\mathbf{c}}_1, \hat{\mathbf{c}}_2, ..., \hat{\mathbf{c}}_\mathbf{m}] \tag{4}$$

2.2.4 Fully Connected Layer. The features from the pooling layers are passed to a fully connected softmax layer whose output is the probability distribution over labels. We apply dropout [35] as a mean of regularization by randomly setting to zero a proportion of elements of the feature vector. The parameters of the network are trained to minimize the cross-entropy of the predicted and true distributions.

$$L(\hat{y}, y) = -\sum_{i=1}^{N}\sum_{j=1}^{M} y_i^j log(\hat{y}_i^j), \tag{5}$$

where y_i^j and $\hat{y}_i^j$ are the ground-truth label and prediction probabilities, respectively, while N and M denotes the number of training samples and the class number, respectively.

3 METHODOLOGY

3.1 Multi-task Learning Framework for Large-scale Text Classification

Most neural network methods often suffer from large-scale text classification [16], since training documents for each category are insufficient. Motivated by the success of multi-task learning on text classification [25, 26, 39, 41], we propose a new CNN-based multi-task learning framework for the large-scale text classification. We apply multi-task learning by treating small- and large-scale classification tasks as two related tasks. The proposed framework leverages supervised data from the small-scale classification task, which has relatively sufficient training data whose distribution is balanced. Figure 1 shows our proposed model.

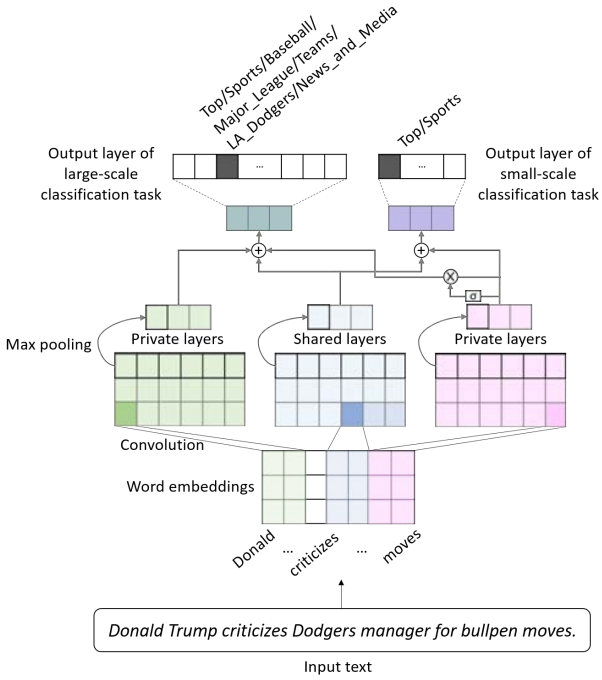


Figure 1: Illustration of architecture for multi-task CNN

In the proposed CNN architecture, the small- and large-scale classification tasks share the same embedding, convolution, and pooling layer. Each task, however, keeps its own private convolution, pooling and fully connected layer as well. Thus, the proposed model maintains two feature spaces: private (i.e., task-dependent) and shared (i.e., task-invariant) feature spaces. We denote the features obtained from the shared pooling layer as $\mathbf{z}_{sh}$. The features obtained from the private pooling layers of the small- and large-scale classification tasks are denoted as $\mathbf{z}_s$ and $\mathbf{z}_l$, respectively.

Although the shared layers help small- and large-scale classification tasks share useful common features, some useful and shareable features may be unidentified. In addition, the private features of small-scale classification task may be higher quality than the private features of large-scale classification task, because the small-scale classification task has relatively sufficient training data. Therefore, we propose a gate mechanism, which enables the large-scale classification task to borrow private features from the small-scale classification task. The gate mechanism controls the flow of features from the small-scale classification task to the large-scale classification task (we shall investigate both symmetric and asymmetric gate mechanisms through experiments in Section 4). When the large-scale classification task l borrows the features from the small-scale classification task s, a gate $\mathbf{g}$ selects the helpful features as follows:

$$\mathbf{g} = \sigma(\mathbf{W}_{s\to l}\mathbf{z}_s + \mathbf{b}_{s\to l}) \tag{6}$$

where σ denotes the non-linear activation of sigmoid, which guarantees the values of $\mathbf{g}$ in the range [0,1]. $\mathbf{W}_{s\to l}$ is the weight which needs to be learned, $\mathbf{b}_{s\to l}$ is a bias term. For the large-scale

Table 2: Statistics of datasets

		Training dataset	Test dataset	Development dataset
ODP (large-scale)	No. categories	2,735	2,735	2,477
	No. webpages	50,246	23,444	2,477
ODP (small-scale)	No. categories	13	13	13
	No. webpages	996,531	23,354	12,526
NYT	No. articles	-	120	-

classification task, the final features $\mathbf{z}'_l$ is calculated by fusing $\mathbf{z}_{sh}$, $\mathbf{z}_s$ and $\mathbf{z}_l$ as follows:

$$\mathbf{z}'_l = \mathbf{z}_l + \mathbf{z}_{sh} + \mathbf{g} \odot \mathbf{z}_s \tag{7}$$

where $\odot$ denotes element-wise multiplication. The features from the small-scale classification task are merged into the large-scale classification task after the selection by gates.

For the small-scale classification task, the final features $\mathbf{z}'_s$ are calculated by adding $\mathbf{z}_{sh}$ and $\mathbf{z}_s$ as follows:

$$\mathbf{z}'_s = \mathbf{z}_s + \mathbf{z}_{sh} \tag{8}$$

In the last layer of each task, vector representations $\mathbf{z}'_l$ and $\mathbf{z}'_s$ are ultimately fed into corresponding fully connected softmax layers to fit the number of classes, which emits the prediction of probability distribution for tasks l and s, respectively.

$$\hat{y_l} = softmax(\mathbf{W}_l \mathbf{z}'_l + \mathbf{b}_l) \tag{9}$$

$$\hat{y_s} = softmax(\mathbf{W}_s \mathbf{z}'_s + \mathbf{b}_s) \tag{10}$$

where $\hat{y_l}$ and $\hat{y_s}$ are predictive results, $\mathbf{W}_l$ and $\mathbf{W}_s$ are the weights of the fully connected layers, while $\mathbf{b}_l$ and $\mathbf{b}_s$ are the bias terms.

Given the predictions of all tasks, a global loss function forces the model to minimize the cross-entropy of the predicted and true distributions for all the tasks as follows:

$$\Phi = \lambda_l L(\hat{y_l}, y_l) + \lambda_s L(\hat{y_s}, y_s) \tag{11}$$

where λ_l and λ_s are the weights for the large- and small-scale classification task, respectively.

3.2 Training

3.2.1 Joint Training. Following [7, 25], the training of the small- and large-scale text classification tasks is achieved in a stochastic manner by looping over the tasks:

(1) Select a random task.
(2) Select a random training example from this task.
(3) Update the parameters for this by taking a gradient step with respect to this example.
(4) Go to 1.

3.2.2 Fine Tuning. For the proposed model, there are shared layers for the small- and large-scale text classification tasks. Thus, after the joint learning phase, we use a fine tuning strategy to further optimize the performance for the large-scale text classification task.

3.2.3 Over-sampling. The distribution of the ODP training dataset for the large-scale text classification task is highly skewed. We apply an over-sampling, which duplicates instances of under-represented classes until the dataset gets balanced, to lessen the impact of the imbalanced dataset [11, 20].

4 EXPERIMENTS

We experiment with a large-scale text classification task to verify the efficacy of the proposed multi-task learning framework.

4.1 Datasets

4.1.1 ODP. We use the RDF dump from the original ODP dataset released on January 8, 2017. As previously mentioned, to obtain a well-organized ODP taxonomy, we apply heuristic rules and build our own large-scale taxonomy with 2,735 categories. Thus, the final training dataset used in our experiments consists of 50,246 webpages. To construct the small-scale classification dataset, we use only 13 top-level categories from the large-scale taxonomy by excluding two categories, *Top/News* and *Top/Adult*, which contain fewer than 100 webpages. Thus, the training dataset used in the small-scale classification task consists of 996,531 webpages.

The ODP test dataset consists of webpages collected from the original ODP. The webpages in each category are randomly divided into a training set, a test set and a development set. In particular, we build two kinds of ODP test datasets. In the large-scale classification task, we collect 23,444 webpages from 2,735 ODP categories in large-scale taxonomy, while collecting 23,354 webpages from 13 ODP categories in the small-scale classification task.

4.1.2 NYT. In addition to the ODP test datasets, we select six categories related to the New York Times as the source for our second test dataset: *art*, *business*, *food*, *health*, *politics*, and *sports*. We randomly collect 20 news articles from each of these categories. Table 2 shows the statistics of datasets.

4.2 Evaluation Metrics

For the ODP datasets, we use the F_1 measure [40] as the classification performance metric, which is the balanced harmonic mean of precision and recall. We use two averaging methods to compute the F_1 measure: macro-averaging (Ma-F_1) and micro-averaging (Mi-F_1). In macro-averaging, F_1 value is measured for each category and then a simple average of all F_1 measures is computed. Conversely, micro-averaging computes the F_1 measure over all test documents. That is, macro-averaging gives equal weight to each category, whereas micro-averaging gives equal weight to each document classification decision. Thus, Mi-F_1 measure depends on the performance of categories with a large number of test documents,

while Ma-F_1 measure is strongly influenced by the performance of categories with fewer test documents.

For the NYT test dataset, we use precision at k. Three participants manually assess the top-k ODP categories obtained by the text classifiers according to three scales: relevant, somewhat relevant, and not relevant. For example, if two categories from the top-5 categories are relevant and one category is somewhat relevant, then the precision at 5 is measured as 0.6.

4.3 Comparisons with State-of-the-art Models

We evaluate the performance of eleven methods. We first adopt the explicit representation-based text classification [23] as the state-of-the-art for large-scale text classification. Other baselines include CNN, LSTM, fastText, and paragraph vector-based text classifiers, which are state-of-the-art methods on multi-class text classification. We also choose four baselines for neural network-based multi-task learning. In our experiments, we compare the following methods:

- *MC* (baseline): This is a text classification method using the explicit representation model based on bag-of-words, called merge centroid, which utilizes enriched training data for each category based on documents classified into their descendants [23].
- *CNN* (baseline): This is the convolutional neural network-based text classification method [17]. We set the hyperparameters to be the same as those of the proposed model.
- *LSTM* (baseline): This is the long short-term memory network-based text classification method [15]. We set the hidden dimension of the LSTM layer to 800, while the other hyperparameters (i.e., mini-batch size, dropout rate, regularization weight, and learning rate) are set to be the same as those of the proposed model.
- *BiLSTM* (baseline): This is the bidirectional long short-term memory network-based text classification method. We set the hidden dimension of each LSTM layer to 800, while the other hyperparameters are set to be the same as those of the proposed model.
- *PV* (baseline): This is the text classification method using paragraph vectors [22]. The learned vector representations have 300 dimensions. We represent categories by averaging the document embeddings for each document in a category.
- $fastText$ (baseline): This is a simple n-gram based neural network model for text classification [14]. The learned vector representations have 300 dimensions.
- *MT-DNN* (baseline): This is a multi-task learning model with DNN, in which a hidden layer is shared [27]. We set the hidden dimension of a shared layer to 300, while the other hyperparameters are set to be the same as those of the proposed model.
- *MT-CNN* (baseline): This is a multi-task learning model with CNN, in which lookup-tables are shared partially while other layers are task-specific [7]. We set the number of filters of private convolution layers to 300, while the other hyperparameters are set to be the same as those of the proposed model.
- *FS-LSTM* (baseline): This is a multi-task learning model with LSTM, in which a single LSTM layer is shared [26]. We set the hidden dimension of a shared LSTM layer to 300, while the other hyperparameters are set to be the same as those of the proposed model.
- *SP-LSTM* (baseline): This is a multi-task learning model with LSTM, in which each task has a private LSTM layer and shares a shared LSTM layer [26]. We set the hidden dimension of both the shared and private LSTM layers to 300, while the other hyperparameters are set to be the same as those of the proposed model.
- *SPG-CNN*: This is our proposed multi-task learning with CNN, in which each task has a private convolution layer and shares a shared convolution layer. In addition, the large-scale classification task borrows private features from the small-scale classification task through a gate mechanism.

Table 3: Detail of word embeddings

	Corpus	Dimension	|V|
Word2Vec [30]	Google News	300	3,000,000
GloVe [32]	Wikipedia	300	400,000
fastText [3]	Wikipedia	300	∞
GWR [18]	Google News	300	∞

4.4 Hyperparameters and Training

We implement the proposed model and competitor methods using TensorFlow frameworks [1] and train models in a single machine equipped with an Intel Core i5 processor, 32GB of RAM, and an NVIDIA GeForce GTX 1080 Ti with 11GB of RAM.

The networks are trained with backpropagation and the gradient-based optimization is performed using the Adam update rule [19]. The embedding layers for all of the neural networks based models (i.e., apart from *MC*) are initialized with the four pre-trained word embeddings, which are summarized in Table 3. We adopt the publicly available Word2Vec [2] and GloVe [3] models, which are popular word embeddings techniques. However, Word2Vec and GloVe models cannot represent out-of-vocabulary (OOV) words, which do not appear in the pre-trained word embedding. Thus, we additionally adopt fastText [3] (this is a word representation technique, being different from $fastText$ [14] that belongs to the baselines) and GWR [18], a character-based word representation approach, to handle OOV words. These approaches allow our framework to effectively expand the vocabulary in the large-scale text classification. The word embeddings are fine-tuned during training to improve performance. The other parameters are initialized by Xavier [9].

The hyperparameters that achieve the best performance on the development set will be chosen for the final evaluation. The final hyperparameters are as follows. For all datasets we use the ReLU, filter windows of 2, 3 and 4 with 300 filters each, mini-batch size of 64, dropout rate of 0.6, regularization weight of 0.001, and learning rate of 0.001. We set both λ_l and λ_s to 0.5.

[2] https://code.google.com/archive/p/word2vec/
[3] https://nlp.stanford.edu/projects/glove/

4.5 Experimental Results

We first compare a few gate mechanisms with the ODP dataset (2,735 categories). In Table 4, *SP-CNN* denotes the multi-task learning model with CNN by the shared and private layers architecture without the gate mechanism. *SPG-CNN*$_{s2l}$ denotes *SP-CNN* using the gate mechanism that controls the flow of features from the small- to the large-scale text classification task, while the gate mechanism in *SPG-CNN*$_{l2s}$ controls the flow of features from the large- to the small-scale classification task. In *SPG-CNN*$_{bi}$, the gate mechanism is used in two-way. We observe that *SPG-CNN*$_{s2l}$ outperforms *SP-CNN*, *SPG-CNN*$_{l2s}$, and *SPG-CNN*$_{bi}$. In particular, our experimental results show that *SPG-CNN*$_{l2s}$ performs worse than *SP-CNN*. These results demonstrate that only the flow of features from the small- to the large-scale classification task is effective at the large-scale text classification. Thus, we adopt *SPG-CNN*$_{s2l}$ in the rest of experiments, which is simply denoted as *SPG-CNN* hereafter.

Table 4: Comparison of symmetric and asymmetric gate mechanisms on the ODP dataset (2,735 categories)

	Mi-F_1	Ma-F_1
SP-CNN	0.550	0.430
SPG-CNN$_{s2l}$	**0.556**	**0.448**
SPG-CNN$_{l2s}$	0.509	0.429
SPG-CNN$_{bi}$	0.551	0.442

We also compare the performance of different pre-trained word embeddings in *SPG-CNN*. Table 5 shows the results of various word embeddings. We observe that *SPG-CNN* performs the best when the embedding layer is initialized with fastText. The publicly available Word2Vec and GloVe embeddings are 4.3% and 2.8% behind fastText embeddings in terms of Ma-F_1, respectively. GWR embeddings perform close to fastText embeddings. One possible reason that fastText and GWR, character-based word embedding approaches, perform better than other popular word-based embeddings is that a large number of OOV words in pre-trained word embeddings lead to the suboptimal performance of large-scale text classification. In the remaining experiments, the embedding layers of all the neural networks based models are initialized with fastText.

Table 5: Comparison of different word embeddings on the ODP dataset (2,735 categories)

	Word embedding	Mi-F_1	Ma-F_1
SPG-CNN	Word2Vec [30]	0.525	0.405
	GloVe [32]	0.530	0.420
	fastText [3]	**0.556**	**0.448**
	GWR [18]	0.550	0.435

Table 6 summarizes the experimental results for large-scale text classification on the ODP test dataset with 2,735 target classes. We observe that *SPG-CNN* outperforms all the baselines. *SPG-CNN* performs better than *MC* over 14% and 5% on average in terms of Mi-F_1 and Ma-F_1, respectively. We also observe that *SPG-CNN* exhibits

Table 6: Large-scale classification performance on the ODP dataset (2,735 categories)

	Mi-F_1	Ma-F_1
MC [23]	0.486	0.426
CNN [17]	0.528	0.413
LSTM [15]	0.430	0.316
BiLSTM	0.495	0.376
PV [22]	0.337	0.234
$fastText$ [14]	0.460	0.371
MT-DNN [27]	0.373	0.260
MT-CNN [7]	0.512	0.435
FS-LSTM [26]	0.545	0.413
SP-LSTM [26]	0.541	0.418
SPG-CNN	**0.556**	**0.448**

Table 7: Small-scale classification performance on the ODP dataset (13 categories)

	Mi-F_1	Ma-F_1
MC [23]	0.757	0.689
CNN [17]	0.804	0.754
LSTM [15]	0.810	0.763
BiLSTM	0.824	0.779
PV [22]	0.738	0.612
$fastText$ [14]	0.727	0.568
MT-DNN [27]	0.793	0.731
MT-CNN [7]	0.800	0.739
FS-LSTM [26]	0.826	0.781
SP-LSTM [26]	**0.828**	**0.782**
SPG-CNN	0.804	0.755

improvement compared to *MT-DNN*, *MT-CNN*, *FS-LSTM* and *SP-LSTM* by over 72%, 3%, 8%, and 7% in terms of Ma-F_1, respectively. Our experimental results show that neural network-based single-task methods except CNN (i.e., *LSTM*, *BiLSTM*, *PV*, and $fastText$) perform worse than *MC* in terms of Mi-F_1 and Ma-F_1. It turns out that *CNN* performs the best among the single-task neural models and better than *MC* over 9% on average in terms of Mi-F_1. However, *CNN* performs worse than *MC* in terms of Ma-F_1. The average performance of multi-task learning models with different scales (i.e., *MT-DNN*, *MT-CNN*, *FS-LSTM*, *SP-LSTM*, and *SPG-CNN*) achieves an improvement of 10.8% and 10.9% over the average performance of single-task learning models in terms of Mi-F_1 and Ma-F_1, respectively. This experimental result indicates that the utilization of multi-task learning with different scales is indeed effective in the large-scale text classification.

Table 7 shows the evaluation results for small-scale text classification on the ODP test dataset with 13 target categories. We observe that *SP-LSTM* outperforms *SPG-CNN*, as well as other baselines. These results are consistent with the results of previous work [24–26], reporting the superior performance of LSTM-based text classifier trained on sufficient training data whose distribution is balanced. We observe that all deep neural network-based

single-task methods (i.e., *CNN*, *LSTM*, and *BiLSTM*) exhibit better performances than *MC* in the small-scale text classification. From Table 6 and 7, we confirm that neural network-based single-task methods are indeed limited to the small-scale text classification. We also observe that the average performance of multi-task learning models with different scales achieves the improvement of 4% and 10% over that of single-task learning models in terms of Mi-F_1 and Ma-F_1, respectively. This result demonstrates that multi-task learning is also helpful to the small-scale text classification task.

Table 8 shows evaluation results on the NYT test dataset. Again, *SPG-CNN* outperforms *CNN*, *LSTM*, *BiLSTM*, *PV*, *fastText*, *MT-DNN*, *MT-CNN*, *FS-LSTM*, and *SP-LSTM* by over 49%, 86%, 52%, 122%, 34%, 18%, 6%, 34%, and 29% in terms of precision at k on average, respectively. We also observe that the average performance of multi-task learning models with different scales achieves the improvement of 41% over that of single-task learning models on average. We also perform the t-test for the classification results, and find that *SPG-CNN* results are statistically significant with $p < 0.01$.

Table 8: Classification performance on the NYT dataset (2,735 categories)

	Precision at k				
	1	2	3	4	5
CNN [17]	0.325	0.279	0.231	0.208	0.193
LSTM [15]	0.225	0.221	0.189	0.177	0.175
BiLSTM	0.308	0.275	0.219	0.204	0.200
PV [22]	0.208	0.171	0.161	0.150	0.138
fastText [14]	0.292	0.296	0.281	0.260	0.242
MT-DNN [27]	0.383	0.317	0.300	0.285	0.272
MT-CNN [7]	0.417	0.354	0.333	0.325	0.310
FS-LSTM [26]	0.292	0.296	0.281	0.260	0.242
SP-LSTM [26]	0.325	0.288	0.281	0.267	0.262
SPG-CNN	**0.458**	**0.392**	**0.353**	**0.325**	**0.312**

We measure the training time for *SPG-CNN* and other neural network-based multi-task learning models. For a fair comparison, we use the same hardware resources, and record the training time for a single epoch on ODP dataset. In Table 9, we observe that *SPG-CNN* has better classification performance while learning faster than *FS-LSTM* and *SP-LSTM* by 1.2x and 1.26x, respectively. Although *MT-DNN* and *MT-CNN* are 1.98x and 1.18x faster than *SPG-CNN*, respectively, *MT-DNN* and *MT-CNN* perform worse than *SPG-CNN*. These results confirm that *SPG-CNN* is effective with a reasonable cost of training time.

5 ANALYSIS

We qualitatively examine the classified categories to analyze why the utilization of the small-scale classification task with the gate mechanism improves the performance of large-scale text classification. We take the NYT articles related to topics "Health care policies" and "Nutritions" as examples. First, from Table 10, we observe that *SPG-CNN* and *SP-CNN* classify the given documents into highly relevant categories. More specifically, from Table 10(a),

Table 9: Training time for a single epoch on ODP dataset compared to multi-task learning models

	Training time	Mi-F_1	Ma-F_1
MT-DNN [27]	**2h 7m**	0.373	0.260
MT-CNN [7]	3h 34m	0.512	0.435
FS-LSTM [26]	5h 3m	0.545	0.413
SP-LSTM [26]	5h 17m	0.541	0.418
SPG-CNN	4h 12m	**0.556**	**0.448**

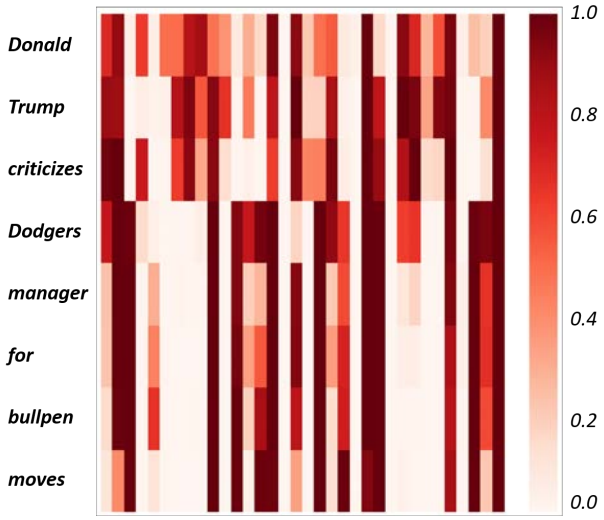


Figure 2: Visualization of the gate's activation. Y-axis represents the input words in chronological order, while X-axis represents gates.

we observe that *SPG-CNN* returns the "Social issues" related categories in top-3. *SP-CNN* also returns two "Social issues" related categories and a category related to "Insurance" that are somewhat associated with the article. In contrast, *CNN* often misclassifies top-level category (e.g., *Top/Computers/...*). In Table 10(b), we also observe that all the categories in top-3 returned by *SPG-CNN* are about "Nutritions" or "Medicine research", while *SP-CNN* returns two relevant categories and an irrelevant category. *CNN* returns a relevant category only on top-1.

We also observe that *SP-CNN* has the same small-scale classification results as *SPG-CNN* (i.e., *Top/Society*, *Top/Kids_and_Teens*, and *Top/Health* in Table 10(a), while *Top/Shopping*, *Top/Health*, and *Top/Science* in Table 10(b)). Nevertheless, it often returns categories that are not related to small-scale classification results (e.g., *Top/Business/Financial_Services/Insurance/Carriers*) and produces worse classification results than *SPG-CNN*. These results clearly demonstrate that the proposed gate mechanism successfully improves the performance of the large-scale text classification by effectively utilizing useful features of the small-scale classification task.

Table 10: Illustrations of Classified Categories

(a) "States' Policies on Health Care Exclude Some of the Poorest. The refusal by about half the states to expand Medicaid will leave millions of poor people ineligible for government-subsidized health insurance under President Obama's health care law even as many others with higher incomes receive federal subsidies to buy insurance." in NYT dataset

Method	Scale	Rank	Category
SPG-CNN	large-scale	1	***Top/Society/Issues/Poverty***
		2	***Top/Society/Issues/Health***
		3	***Top/Society/Issues/Children,_Youth_and_Family/Advocacy_Groups***
	small-scale	1	***Top/Society***
		2	*Top/Kids_and_Teens*
		3	***Top/Health***
SP-CNN	large-scale	1	***Top/Society/Issues/Poverty***
		2	***Top/Society/Issues/Health***
		3	*Top/Business/Financial_Services/Insurance/Carriers*
	small-scale	1	***Top/Society***
		2	*Top/Kids_and_Teens*
		3	***Top/Health***
CNN [17]	large-scale	1	***Top/Society/Issues/Health***
		2	***Top/Home/Personal_Finance/Insurance***
		3	*Top/Computers/Software/Industry-Specific/Insurance*

(b) "Fish Oil Claims Not Supported by Research. Fish oil is now the third most widely used dietary supplement in the United States, after vitamins and minerals, according to a recent report from the National Institutes of Health. At least 10 percent of Americans take fish oil regularly, most believing that the omega-3 fatty acids in the supplements will protect their cardiovascular health." in NYT dataset

Method	Scale	Rank	Category
SPG-CNN	large-scale	1	***Top/Shopping/Health/Nutrition***
		2	***Top/Shopping/Health/Nutrition/Supplements***
		3	***Top/Health/Medicine/Research***
	small-scale	1	***Top/Shopping***
		2	***Top/Health***
		3	***Top/Science***
SP-CNN	large-scale	1	***Top/Shopping/Health/Nutrition/Supplements***
		2	*Top/Shopping/Health/Alternative/Herbs*
		3	***Top/Health/Medicine/Research***
	small-scale	1	***Top/Shopping***
		2	***Top/Health***
		3	***Top/Science***
CNN [17]	large-scale	1	***Top/Shopping/Health/Nutrition/Supplements***
		2	*Top/Business/Industrial_Goods_and_Services/Calibration_and_Testing/Ultrasonic_Testing*
		3	*Top/Recreation/Pets/Fish_and_Aquaria*

In addition, we analyze the gate activation in *SPG-CNN*, which controls the flow of features from small-scale classification task to large-scale classification task. We select an article title "Donald Trump criticizes Dodgers manager for bullpen moves", which is related to both "Issues" and "Sports". In Figure 2, we plot the gate activation as each word is processed. The small- and large-scale classification tasks of *SPG-CNN* return *Top/Society* and *Top/Society/Issues/Warfare_and_Conflict* when processing up to the third word (or criticizes), while *Top/Sports* and *Top/Sports/Baseball/Major_League/Teams/ Los_Angeles_Dodgers/News_and_Media* when the last word is processed. Interestingly, from Figure 2, we observe that different neurons are activated as predicted categories get changed. These results show how the large-scale text classification task in *SPG-CNN* selectively utilizes useful features of the small-scale classification task in practice.

6 RELATED WORK

For the large-scale text classification, many works have been proposed to handle data sparsity on knowledge bases. McCallum et al. firstly addressed data sparsity on a hierarchical taxonomy [29]. They adopted a statistical technique, called *shrinkage*, to estimate the parameters of data-sparse child categories with their data-rich

ancestor categories. In [10, 23], they proposed a large-scale text classification method called merge-centroid (MC). MC utilizes enriched training data for each category based on webpages classified into their ancestor and/or descendants in the ODP. In another line of work [34], they enriched semantic information, such as phrases and related terms, in the ODP documents by utilizing another knowledge base, Wikipedia. However, these approaches only consider term weighting approaches, without considering the importance of semantic similarity between words.

Neural network based models have utilized multi-task learning to overcome the problem of insufficient training data. In [7], they proposed multi-task learning with convolutional neural networks for several traditional NLP tasks. This study utilized a shared representation for input words, while we utilize a shared convolution layer and private convolution layers to construct the multi-task learning framework. A related work [25] proposed three different models of sharing information with recurrent neural networks (RNN). In another line of work [26], the authors augmented RNN-based multi-task learning by using adversarial training. Another work [39] proposed a CNN-based architecture for multi-task learning, which shares features via a gated sharing unit instead of a shared layer. However, none of the above research deals with the large-scale text classification. To the best of our knowledge, our current work is one of only a few works that apply multi-task learning to the large-scale text classification by treating different scales of text classification.

7 CONCLUSION

In this paper, we have proposed a novel CNN-based multi-task learning framework to handle the large-scale text classification. Specifically, we have incorporated the gate mechanism into the multi-task CNN to selectively utilize useful features of small-scale classification task. Our approach involves two steps. We first convert the large-scale classification task to a small-scale classification task. Secondly, we train the proposed CNN-based multi-task learning model, which simultaneously learns small- and large-scale classification tasks with the gate mechanism. We have verified the large-scale text classification performance of our methodology using real-world datasets. The experimental results confirm that our scheme significantly outperforms several strong baseline methods. We plan to integrate explicit and implicit representation techniques to further enhance the large-scale text classification.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their valuable comments. This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2018R1A2A1A05078380).

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-scale Machine Learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI'16)*. 265–283.

[2] Bahram Amini, Roliana Ibrahim, Mohd Shahizan Othman, and Mohammad Ali Nematbakhsh. 2015. A Reference Ontology for Profiling Scholar's Background Knowledge in Recommender Systems. *Expert Syst. Appl.* 42, 2 (Feb. 2015), 913–928.

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics* 5 (2017), 135–146.

[4] Andrei Broder, Marcus Fontoura, Evgeniy Gabrilovich, Amruta Joshi, Vanja Josifovski, and Tong Zhang. 2007. Robust classification of rare queries using web knowledge. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'07)*. 231–238.

[5] Andrei Broder, Marcus Fontoura, Vanja Josifovski, and Lance Riedel. 2007. A Semantic Approach to Contextual Advertising. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'07)*. 559–566.

[6] Paul Alexandru Chirita, Wolfgang Nejdl, Raluca Paiu, and Christian Kohlschütter. 2005. Using ODP metadata to personalize search. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'05)*. 178–185.

[7] Ronan Collobert and Jason Weston. 2008. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML'08)*. 160–167.

[8] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* 12 (Nov. 2011), 2493–2537.

[9] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10)*. 249–256.

[10] Jongwoo Ha, Jung-Hyun Lee, Won-Jun Jang, Yong-Ku Lee, and SangKeun Lee. 2014. Toward Robust Classification Using the Open Directory Project. In *Proceedings of the 2014 International Conference on Data Science and Advanced Analytics (DSAA'14)*. 607–612.

[11] Haibo He and Edwardo A. Garcia. 2009. Learning from Imbalanced Data. *IEEE Trans. on Knowl. and Data Eng.* 21, 9 (Sept. 2009), 1263–1284.

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780.

[13] Rie Johnson and Tong Zhang. 2016. Supervised and Semi-supervised Text Categorization Using LSTM for Region Embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning (ICML'16)*. 526–534.

[14] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL'17)*. 427–431.

[15] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An Empirical Exploration of Recurrent Network Architectures. In *Proceedings of the 32nd International Conference on Machine Learning (ICML'15)*. 2342–2350.

[16] Kang-Min Kim, Dinara Aliyeva, Byung-Ju Choi, and SangKeun Lee. 2018. Incorporating Word Embeddings into Open Directory Project based Large-scale Classification. In *Proceedings of the 22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'18)*. 376–388.

[17] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. 1746–1751.

[18] Yeachan Kim, Kang-Min Kim, Ji-Min Lee, and SangKeun Lee. 2018. Learning to Generate Word Representations using Subword Information. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING'18)*. 2551–2561.

[19] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[20] Bartosz Krawczyk. 2016. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence* 5 (2016), 221–232.

[21] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*. 2267–2273.

[22] Quoc V Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML'14)*. 1188–1196.

[23] Jung-Hyun Lee, JongWoo Ha, Jin-Yong Jung, and SangKeun Lee. 2013. Semantic Contextual Advertising based on the Open Directory Project. *ACM Trans. on the Web* 7, 4 (Nov. 2013), 24:1–24:22.

[24] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Deep Multi-Task Learning with Shared Memory for Text Classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP'16)*. 118–127.

[25] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent Neural Network for Text Classification with Multi-Task Learning. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. 2873–2879.

[26] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial Multi-task Learning for Text Classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL'17)*. 1–10.

[27] Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. 2015. Representation Learning Using Multi-Task Deep Neural Networks for Semantic Classification and Information Retrieval. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'15)*. 912–921.

[28] Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. 1412–1421.

[29] Andrew McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. 1998. Improving Text Classification by Shrinkage in a Hierarchy of Classes. In *Proceedings of the 15th International Conference on Machine Learning (ICML'98)*. 359–367.

[30] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of the 27th Conference on Neural Information Processing Systems (NIPS'13)*. 3111–3119.

[31] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML'10)*. 807–814.

[32] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. 1532–1543.

[33] Woo-Jong Ryu, Jung-Hyun Lee, Kang-Min Kim, and SangKeun Lee. 2017. meCurate: Personalized Curation Service Using a Tiny Text Intelligence. In *Proceedings of the 26th International Conference on World Wide Web Companion (WWW'17 Companion)*. 269–272.

[34] HaeYong Shin, GeunJae Lee, Woo-Jong Ryu, and SangKeun Lee. 2017. Utilizing Wikipedia Knowledge in Open Directory Project-based Text Classification. In *Proceedings of the 32nd Symposium on Applied Computing (SAC'17)*. 309–314.

[35] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* 15, 1 (Jan. 2014), 1929–1958.

[36] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 28th Conference on Neural Information Processing Systems (NIPS'14)*. 3104–3112.

[37] Jin Wang, Zhongyuan Wang, Dawei Zhang, and Jun Yan. 2017. Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*. 2915–2921.

[38] Zhongyuan Wang and Haixun Wang. 2016. Understanding Short Texts. In *the 54th Annual Meeting of the Association for Computational Linguistics (Tutorial) (ACL'16)*.

[39] Liqiang Xiao, Honglun Zhang, and Wenqing Chen. 2018. Gated Multi-Task Network for Text Classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'18)*. 726–731.

[40] Yiming Yang. 1999. An evaluation of statistical approaches to text categorization. *Inf. Retr.* 1, 1 (May 1999), 69–90.

[41] Honglun Zhang, Liqiang Xiao, Yongkun Wang, and Yaohui Jin. 2017. A generalized recurrent neural architecture for text classification with multi-task learning. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*. 3385–3391.

[42] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In *Proceedings of the 29th Conference on Neural Information Processing Systems (NIPS'15)*. 649–657.

Dual Neural Personalized Ranking

Seunghyeon Kim
Sungkyunkwan University
South Korea
kimsh1509@skku.edu

Jongwuk Lee*
Sungkyunkwan University
South Korea
jongwuklee@skku.edu

Hyunjung Shim
Yonsei University
South Korea
kateshim@yonsei.ac.kr

ABSTRACT

Implicit user feedback is a fundamental dataset for personalized recommendation models. Because of its inherent characteristics of sparse one-class values, it is challenging to uncover meaningful user/item representations. In this paper, we propose *dual neural personalized ranking* (*DualNPR*), which fully exploits both user- and item-side pairwise rankings in a unified manner. The key novelties of the proposed model are three-fold: (1) DualNPR discovers mutual correlation among users and items by utilizing both user- and item-side pairwise rankings, alleviating the data sparsity problem. We stress that, unlike existing models that require extra information, DualNPR naturally augments both user- and item-side pairwise rankings from a user-item interaction matrix. (2) DualNPR is built upon deep matrix factorization to capture the variability of user/item representations. In particular, it chooses raw user/item vectors as an input and learns latent user/item representations effectively. (3) DualNPR employs a dynamic negative sampling method using an exponential function, further improving the accuracy of top-N recommendation. In experimental results over three benchmark datasets, DualNPR outperforms baseline models by 21.9–86.7% in hit rate, 14.5–105.8% in normalized discounted cumulative gain, and 5.1–23.3% in the area under the ROC curve.

CCS CONCEPTS

• **Information systems → Collaborative filtering**.

KEYWORDS

Implicit feedback; top-N recommendation; dual representation

ACM Reference Format:
Seunghyeon Kim, Jongwuk Lee, and Hyunjung Shim. 2019. Dual Neural Personalized Ranking. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313585

1 INTRODUCTION

With the enormous success of e-commerce and online streaming service (*e.g.*, Amazon, eBay, Spotify, and Netflix), recommendation systems have gained much attention in academia and industry [24, 27, 35]. As one of the most successful techniques, collaborative filtering (CF) has been widely used for building a personalized recommendation model. The underlying goal of CF is to predict user preferences on unrated items by discovering hidden correlations among users/items. Concretely, CF is formulated to address either *rating prediction* or *top-N recommendation*. While rating prediction minimizes the error of predicting a user's ratings for her unrated items, top-N recommendation finds a ranked list of the favorite top-N items among unrated items.

Depending on the inherent characteristics of datasets, the recommendation models exploit *explicit* or *implicit* feedback. Earlier approaches have employed user's explicit feedback (*e.g.*, 5-star ratings). While explicit feedback is useful to understand relative user preferences among rated items, it is not always available in practice. By contrast, implicit feedback (*e.g.*, purchase histories, bookmarks, and browsing logs) does not require a user to annotate ratings but is naturally obtained from the user's behavior. Therefore, it is much easier to collect than explicit feedback, and thereby it is more feasible for real-world applications. In this paper, we aim to develop a new recommendation model for the top-N recommendation from implicit feedback.

Existing recommendation models with implicit feedback commonly suffer from the following challenges: (1) Implicit feedback is extremely sparse (*i.e.*, *data sparsity*). In most real datasets, the sparsity is about 96–99%, which leads to it being difficult for the recommendation models to learn meaningful latent user/item representations. (2) The latent user/item representations are usually induced from simple matrix factorization, which assumes that a user-item interaction matrix is represented by a linear combination of latent user vectors/item vectors. When handling highly complicated real-life datasets, this linear model might be too simple to disentangle the latent user/item vectors. (3) While implicit feedback only provides positive data for labling positive experiences, the correct label of unknown feedback is ambiguous (*i.e.*, *data ambiguity*). Unknown feedback is interpreted as a mixture of negative and unlabeled positive feedback. Therefore, it is necessary to distinguish negative feedback from unknown feedback.

To overcome these challenges, various recommendation models have been proposed in the past decade [24]. Among them, Bayesian personalized ranking (BPR) [23] is one of the most representative recommendation models. Unlike previous approaches [10, 14, 19, 25] that directly predict user preferences for unrated items, BPR minimizes a pairwise ranking loss between positive and negative feedback. In this way, it adequately addresses the data sparsity problem. Because BPR still relies on simple matrix factorization, it is incapable of correctly disentangling complex latent user/item representations. In addition, it does not pay much attention to how to distinguish negative feedback from unknowns; BPR regards all unknowns as negative feedback.

Recent studies attempt to inherit the advantage of BPR at handling data sparsity while improving the modeling power. Several

*Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313585

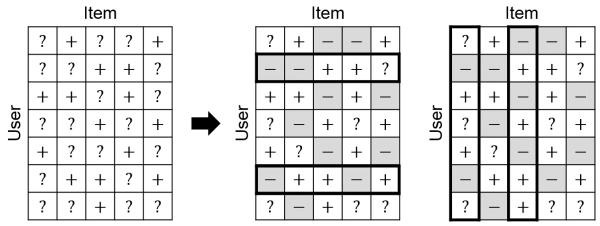


Figure 1: Example of user- and item-side pairwise rankings (center and right) elicited from a user-item interaction matrix (left), where gray color indicates negative feedback from unknown feedback. For instance, user 2 prefers item 3 to item 1 (center) and item 1 is preferred by user 3 over user 6 (right).

studies [4, 6, 18] have integrated the BPR model with deep neural networks (DNNs) to provide more expressiveness (*i.e.*, additional parameters) and flexibility (*i.e.*, non-linear activation functions) in discovering hidden correlations among users/items. Most recently, neural personalized ranking (NPR) [18] has been proposed to generalize the BPR model using DNNs. However, it focused solely on incorporating auxiliary information into DNNs and has not yet studied deep matrix factorization (DMF) to learn latent user/item representations as well as negative sampling to choose negative items from unknown feedback.

In this paper, we propose *dual neural personalized ranking* (*DualNPR*), which incorporates both user- and item-side pairwise rankings into DMF. (Figure 3 depicts the detailed architecture of the proposed model.) The key novelties of the proposed model are summarized as follows.

(1) **Utilizing dual pairwise rankings**: To alleviate the data sparsity problem, we introduce a novel data representation to *fully* exploit both user- and item-side pairwise rankings in a unified manner. Figure 1 illustrates the user- and item-side pairwise rankings collected from a toy user-item interaction matrix. While original BPR only utilizes user-side pairwise ranking, it is intuitive to employ both user- and item-side pairwise rankings to better understand correlations among users/items; it is particularly beneficial to investigate their *mutual* correlations. The use of item-side pairwise ranking is also attractive because we can augment pairwise rankings without requiring any side information. To the best of our knowledge, none of the existing work utilizes both user- and item-side pairwise rankings. Moreover, the idea of adding item-side pairwise ranking can be easily extended to improve the performance of BPR and its variants such as BPR with heterogeneous implicit feedback [20] and NPR [18].

(2) **Incorporating BPR into DNNs**: Motivated by DMF [34], our model adopts multiple hidden layers in DNNs to increase the model capacity, and learns more meaningful latent user/item representations. We believe that our model architecture helps capture the variability of user/item representations and mutual correlations among users/items. In particular, our model takes *raw* user/item vectors as an input. Although this idea is simple, user/item vectors with similar patterns are enforced to be placed as close as possible in latent space. Moreover, we can leverage a denoising method [16, 33, 36] for raw input vectors to make our model more generalized.

(3) **Dynamic negative sampling**: We design *dynamic negative sampling* using an exponential function to distinguish negative items from unknown feedback. Without much computational overhead, our negative sampling dynamically maximizes the gradient gap between positive and negative feedback in training our model. Consequently, it helps improve the quality of our model as well as achieve stable learning convergence.

We conduct extensive experiments with public benchmark datasets including Amazon and MovieLens datasets. Experimental results show that the proposed model significantly outperforms competitive models by 21.9–86.7% in terms of hit rate (HR), 14.5–105.8% in normalized discounted cumulative gain (NDCG), and 5.1–23.3% in the area under the ROC curve (AUC). Furthermore, our model shows an improvement of 35.3–43.3% in HR when both using user- and item-side triplets. In addition, our dynamic negative sampling outperforms existing sampling methods by 3.3–4.6% in HR.

2 PRELIMINARIES

In this section, we first introduce basic notations and formulate the top-N recommendation problem. Then, we review the most relevant existing work such as BPR [23] and NPR [18].

Problem Statement. Let $\mathcal{U}$ be a set of m users, and $\mathcal{I}$ be a set of n items. We are given a user-item interaction matrix $\mathbf{R} \in \{0,1\}^{m \times n}$, where $r_{ui} \in \mathbf{R}$ indicates implicit feedback from user $u \in \mathcal{U}$ for item $i \in \mathcal{I}$. Specifically, $r_{ui} = 1$ means that user feedback is *observed* (or *known*), and it is interpreted as positive feedback. Meanwhile, $r_{ui} = 0$ indicates *unobserved* (or *unknown*) feedback, and it is interpreted as a mixture of negative and unlabeled positive feedback. Given user u, $\mathcal{I}_u^+ = \{i \in \mathcal{I} | r_{ui} = 1\}$ is defined as a set of items with positive feedback by user u. Likewise, $\mathcal{U}_i^+ = \{u \in \mathcal{U} | r_{ui} = 1\}$ is defined as a set of users with positive feedback for item i. For unobserved feedback, $\mathcal{I}_u^- = \mathcal{I} \backslash \mathcal{I}_u^+$ and $\mathcal{U}_i^- = \mathcal{U} \backslash \mathcal{U}_i^+$ denote a set of items and a set of users, respectively.

In this paper, we aim to optimize a top-N recommendation list using pair-wise user/item preferences (*i.e.*, a pair of positive and negative users/items). To formalize this problem, a set D_u of *user-side triplets* is defined as follows:

$$\mathcal{D}_u = \{(u,i,j) | u \in \mathcal{U} \wedge i \in \mathcal{I}_u^+ \wedge j \in \mathcal{I}_u^-\}, \tag{1}$$

where (u,i,j) represents that user u prefers i to j. The pairwise relationship is denoted as $i \succ_u j$. Likewise, a set of *item-side triplets* is defined as follows:

$$\mathcal{D}_i = \{(i,u,v) | i \in \mathcal{I} \wedge u \in \mathcal{U}_i^+ \wedge v \in \mathcal{U}_i^-\}, \tag{2}$$

where (i,u,v) means that item i is preferred by user u over user v, denoted as $u \succ_i v$. Although these relationships are useful resources for learning recommendation models, existing studies [4, 6, 18, 23] built upon BPR solely leverage user-side triplets from a user-item interaction matrix.

Bayesian Personalized Ranking (BPR). Because implicit user feedback only provides sparse one-class values for positive feedback, it is inherently difficult to uncover the relative preferences of users. This is also known as the one-class collaborative filtering (OCCF) problem [10, 19]. To address this problem, BPR [23] is formulated to

minimize a pairwise ranking loss such that all items with positive feedback from a target user are ranked higher than the rest of the items.

Specifically, BPR aims to maximize the posterior probability $p(\Theta|i \succ_u j)$, where Θ is a set of learning parameters. According to Bayes' rule, the posterior probability is defined as:

$$p(\Theta|i \succ_u j) \propto p(i \succ_u j|\Theta)p(\Theta). \tag{3}$$

Then, the likelihood function for Θ is defined as:

$$p(i \succ_u j|\Theta) = \sigma(\hat{r}_{ui} - \hat{r}_{uj}), \tag{4}$$

where $\sigma(\cdot)$ is the sigmoid function and $\hat{r}_{ui}$ and $\hat{r}_{uj}$ are the predicted scores for items i and j by user u, respectively. In BPR, $\hat{r}_{ui}$ is computed by a linear combination of latent user vectors and item vectors [10, 14, 19, 25].

$$\hat{r}_{ui} = \mathbf{p_u}^T \mathbf{q_i}, \tag{5}$$

where $\mathbf{p_u} \in \mathbb{R}^{d \times 1}$ is a latent user vector and $\mathbf{q_i} \in \mathbb{R}^{d \times 1}$ is a latent item vector. (Symbols in bold font are vectors and matrices.) In other words, the model parameters of BPR are $\Theta = \{\mathbf{P}, \mathbf{Q}\}$; $\mathbf{P}$ and $\mathbf{Q}$ indicate a set of latent user vectors and a set of latent item vectors, respectively. The prior density follows a normal distribution $p(\Theta) \sim \mathcal{N}(0, \lambda_\Theta \mathbf{I})$, where λ_Θ is a model-specific parameter and $\mathbf{I}$ is the identity matrix.

Because of the monotone transformation property of the natural log function, maximum likelihood estimation is equivalent to minimizing the negative log likelihood. Finally, based on Bayesian inference, the objective function of BPR aims to minimize a pairwise ranking loss for all user-side triplets.

$$\underset{\Theta}{\operatorname{argmin}} \sum_{(u,i,j) \in \mathcal{D}_u} -\ln\left(\sigma(\hat{r}_{ui} - \hat{r}_{uj})\right) + \Omega(\Theta), \tag{6}$$

where Ω is Frobenius regularization for parameters Θ.

Neural Bayesian Personalized Ranking (NPR). To benefit from the great success of DNNs [15], there are several extensions of BPR using DNNs. First, visual BPR [6] combines the BPR model with auxiliary visual features learned from convolutional neural networks (CNN). Also, Bayesian personalized ranking deep neural network [4] introduces CNN to infer latent user representations from social graphs. In other words, DNNs have been exploited as a feature extractor for additional contexts and inputs for graphs, but have not been extended to BPR to learn complex user/item representations.

Most recently, NPR [18] is proposed to generalize BPR using DNNs. Figure 2 illustrates the architecture of NPR by integrating BPR with DNN. Specifically, it takes a user-side triplet (u, i, j) as an input, where user u and a pair of items i and j are represented by *one-hot* vectors. For instance, in Figure 1, user 2 prefers item 3 to item 1. In this case, $\mathbf{u}$, $\mathbf{i}$, and $\mathbf{j}$ are represented by $[0, 1, 0, 0, 0, 0]$, $[0, 0, 1, 0, 0]$, and $[1, 0, 0, 0, 0]$, respectively. Because there are m users and n items in $\mathbf{R}$, one-hot input vectors $\mathbf{u}$, $\mathbf{i}$, and $\mathbf{j}$ are represented by m-, n-, and n-dimensional vectors, respectively, and the three embedding vectors are computed by:

$$\mathbf{p_u} = \mathbf{W}\mathbf{u}, \quad \mathbf{q_i} = \mathbf{W}'\mathbf{i}, \quad \mathbf{q_j} = \mathbf{W}''\mathbf{j}, \tag{7}$$

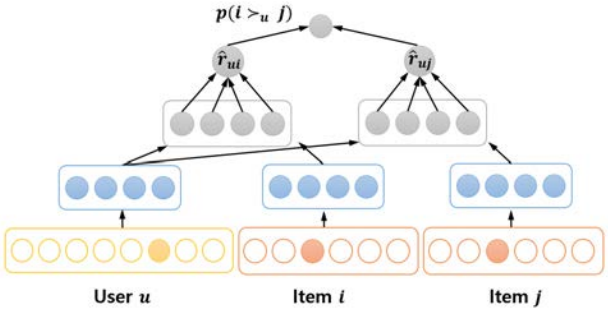


Figure 2: Model architecture of neural personalized ranking (NPR). Note that the user/item is represented by an one-hot vector.

where $\mathbf{W}$, $\mathbf{W}'$, and $\mathbf{W}''$ are embedding matrices for user u and items i and j, respectively. Because they transform the input vector to latent user/items vectors, they can correspond to learning parameters $\mathbf{P}$ and $\mathbf{Q}$ in Eq. (5).

As NPR is vertically symmetric for a user-item pair, we mainly explain the substructure for user u and item i in Figure 2. Unlike existing simple matrix factorization, $\hat{r}_{ui}$ is calculated by a non-linear matrix factorization model.

$$\hat{r}_{ui} = a(\mathbf{w}^\mathbf{T}(\mathbf{p_u} \odot \mathbf{q_i}) + b), \tag{8}$$

where $\odot$ is an element-wise product and $\mathbf{w} \in \mathbb{R}^{d \times 1}$ is a d-dimensional weight vector for latent dimensions and b is the bias term. In addition, $a(\cdot)$ indicates the activation function. In [18], the ReLU function is used as the activation function to capture non-linear user-item interactions. It is reported that Eq. (8) not only identifies the variability of user-item interactions by relaxing an equal weight requirement but also offers non-linear flexibility.

Except for computing $\hat{r}_{ui}$ and $\hat{r}_{uj}$ in Eq. (8), the objective function of NPR is very similar to Eq. (6) in original BPR. It is expected that non-linear matrix factorization used in NPR is effective for representing user/item vectors in a more flexible manner. As a result, NPR can achieve a better quality for top-N recommendation than BPR. However, it does not address the following issues: (1) item-side triplets are also a useful training resource, (2) more generalized DMF can be incorporated to represent latent user/item vectors, and (3) negative sampling is one of the key components for learning BPR and its variants.

3 PROPOSED MODEL

In this section, we propose *dual neural personalized ranking* (*DualNPR*), which complements the drawbacks of BPR and its variants. The key technical contributions of DualNPR can be summarized as follows: (1) We adopt a novel data representation to unify both user- and item-side triplets. Such a unified representation can be utilized for all pairs in user- and item-side rankings. Because the representation can also capture mutual correlations among users/items, we can effectively alleviate the data sparsity problem. (2) Our model is built upon a DMF model with multiple hidden layers to obtain the variability of user/item representations. In this process, raw user/item vectors are taken as an input. The input vectors encode

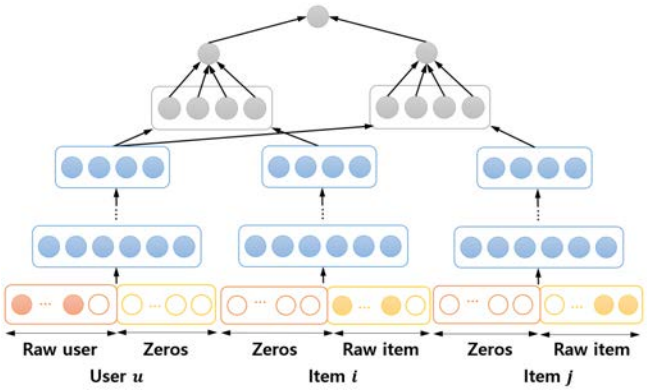


Figure 3: Model architecture of DualNPR. The user/item is represented by a raw vector concatenated with a zero vector.

richer information than conventional one-hot vector representations. Additionally, it is advantageous to apply a denoising scheme for input vectors. (3) We employ a new dynamic negative sampling strategy to choose negative feedback from unknown feedback effectively. This sampling method helps to improve the quality of our model as well as achieve stable learning convergence. In short, DualNPR can be interpreted as a more generalized BPR model built upon DNN without any additional information.

3.1 Model Architecture

Figure 3 illustrates the network architecture of DualNPR, integrating both user- and item-side triplets. Specifically, it consists of the following components: *input layer*, *embedding layer*, *scoring layer*, and *output layer*.

Input Layer: As depicted in Figure 3, while the NPR model uses a one-hot vector representation for user u, our model defines each user u by a *raw* vector across all items. Similarly, each item i is represented by a raw vector across all users. In Figure 1, for instance, when $user_2$ prefers $item_3$ to $item_1$, $\mathbf{u}$, $\mathbf{i}$, and $\mathbf{j}$ are represented by $[0, 0, 1, 1, 0]$, $[0, 1, 0, 1, 0, 1, 1]$, and $[0, 0, 1, 0, 1, 0, 0]$, respectively. While the similarity between two one-hot vectors is difficult to quantify, the similarity between two raw vectors is intuitive and able to be measured by distance metrics. That is, we can expect that the distance between latent vectors should have a similar tendency to the distance between corresponding raw vectors; the similar raw vectors result in similar latent vectors. Moreover, we can apply a denoising scheme for the input layer to increase the robustness of the proposed model as in existing work [16, 33, 36]. Formally, each input vector for a user-side triplet (u, i, j) is represented by:

$$\mathbf{u} = \mathbf{R}_{u*}, \quad \mathbf{i} = \mathbf{R}_{*i}, \quad \mathbf{j} = \mathbf{R}_{*j}, \tag{9}$$

where $\mathbf{R}_{u*} \in \{0,1\}^{n\times 1}$, $\mathbf{R}_{*i} \in \{0,1\}^{m\times 1}$, and $\mathbf{R}_{*j} \in \{0,1\}^{m\times 1}$ are a row vector for user u and column vectors for items i and j in a user-item matrix $\mathbf{R}$, respectively.

We now explain a *dual* data representation to exploit both user and item vectors in a unified manner. To represent user/item vectors, existing recommendation models [7, 8, 18, 34, 36] using DNN usually define user/item representation by different layers with user/item embedding matrices. In contrast to existing models, we adopt a single, unified input vector representation, which is defined by the following $(m + n)$-dimensional concatenated vector.

$$[\mathbf{u}; \mathbf{i}] \in \{0, 1\}^{m+n}. \tag{10}$$

To represent user vector $\mathbf{u}$ with the concatenated vector, the elements for the item vector are set as zeros. Likewise, to represent item vector $\mathbf{i}$, the elements for the user vector are set as zeros. Formally:

$$\mathbf{u} = [\mathbf{R}_{u*}; \mathbf{0}], \quad \mathbf{i} = [\mathbf{0}; \mathbf{R}_{*i}], \quad \mathbf{j} = [\mathbf{0}; \mathbf{R}_{*j}]. \tag{11}$$

This dual representation is capable of sharing one embedding layer instead of using two embedding layers for users/items. We believe that such unified embedding is a key factor to better formulate the personalized ranking problem because it learns mutual correlations across users/items.

Embedding Layer: To learn latent user/item vectors, we use one or multiple hidden layers as used in DMF [34]. For a user-side triplet (u, i, j), the first layer is represented by:

$$l_1 = a(\mathbf{W_1} x + b_1) \ \text{ for } \ x \in \{\mathbf{u}, \mathbf{i}, \mathbf{j}\}, \tag{12}$$

where $\mathbf{W_1} \in \mathbb{R}^{(m+n)\times d}$ is an embedding matrix for users/items, b_1 is the bias term, and $a(\cdot)$ is the activation function. Given (u, i, j), we have three input vectors, $\mathbf{u}$, $\mathbf{i}$ and $\mathbf{j}$. Each is a $(m+n)$-dimensional vector and all three vectors share the same embedding layer. We can also employ multiple intermediate hidden layers.

$$l_i = a(\mathbf{W_i} l_{i-1} + b_i) \ \text{ for } \ i = 2, \ldots, L. \tag{13}$$

Through the multiple hidden layers, (u, i, j) is represented by three low-dimensional vectors in a latent space. We denote the three embedding vectors as $\Phi(\mathbf{u})$, $\Phi(\mathbf{i})$, and $\Phi(\mathbf{j})$.

$$\Phi(\mathbf{u}) = a(\ldots a(\mathbf{W}_2 a(\mathbf{W_1}[\mathbf{R}_{u*}; \mathbf{0}] + b_1) + b_2) \ldots), \tag{14}$$

$$\Phi(\mathbf{i}) = a(\ldots a(\mathbf{W}_2 a(\mathbf{W_1}[\mathbf{0}; \mathbf{R}_{*i}] + b_1) + b_2) \ldots), \tag{15}$$

$$\Phi(\mathbf{j}) = a(\ldots a(\mathbf{W}_2 a(\mathbf{W_1}[\mathbf{0}; \mathbf{R}_{*j}] + b_1) + b_2) \ldots), \tag{16}$$

where $a(\cdot)$ is used as the ReLU function at each layer. From Eq (7) in NPR, $\Phi(\mathbf{u})$, $\Phi(\mathbf{i})$, and $\Phi(\mathbf{j})$ correspond to $\mathbf{p_u}$, $\mathbf{q_i}$, and $\mathbf{q_j}$, respectively, indicating a latent user vector and two latent item vectors.

The key advantage of sharing one embedding layer for both users and items is to being able to leverage an item-side triplet (i, u, v) in a unified manner. That is, one embedding layer through concatenated input vectors can be used for both user- and item-side triplets. As a result, the embedding layer can model mutual information among users/items, which effectively increases the modeling capacity with a smaller number of parameters in comparison with NPR.

Scoring Layer: Similar to traditional latent factor models [10, 14, 19], the scoring layer is used to predict a preference score for user-item interactions. Formally:

$$\hat{r}_{ui} = g\Big(\Phi(\mathbf{u}), \Phi(\mathbf{i})\Big), \tag{17}$$

where $g(\cdot, \cdot)$ is the scoring function for a user-item pair. In general, this function can be represented by a (non-linear) inner product and cosine similarity. In this paper, we investigate the following scoring functions:

- **Inner product**: As the simplest method, this is computed by the inner product between two vectors.

$$\hat{r}_{ui} = \Phi(\mathbf{u})^T \Phi(\mathbf{i}). \tag{18}$$

- **Cosine similarity**: This is computed by the normalized inner product. This function is also used for a deep structured semantic model (DSSM) [11] and a DMF model [34].

$$\hat{r}_{ui} = \frac{\Phi(\mathbf{u})^T \Phi(\mathbf{i})}{||\Phi(\mathbf{u})||\ ||\Phi(\mathbf{i})||}. \tag{19}$$

- **Non-linear inner product**: As in NPR [18], this is computed with a non-linear weighted inner product by appending weights for latent dimensions.

$$\hat{r}_{ui} = a(\mathbf{w}^T (\Phi(\mathbf{u}) \odot \Phi(\mathbf{i})) + b). \tag{20}$$

One might ask whether we should consider more complex scoring functions to improve the modeling power. One of the representative recommendation models using DNNs, neural collaborative filtering [8] successfully employs a complex scoring function using multi-layer perception. Such a deeper network can be useful for capturing non-linear and complex relationships between latent user/item vectors. However, in the current form of our model, we have kept relatively simple scoring functions. This is because we can employ multiple hidden layers for learning latent user/item vectors. Having a complex scoring model may deteriorate the overall quality of the proposed model by incurring the overfitting problem of DNN. Therefore, we skip using complex scoring functions. (In future work, we will study how to optimize the scoring function.)

Output Layer: Given two predicted scores $\hat{r}_{ui}$ and $\hat{r}_{uj}$, we minimize a pairwise ranking loss for Θ. For user- and item-side triplets, the likelihood functions are defined as:

$$p(i >_u j|\Theta) = \sigma(\hat{r}_{ui} - \hat{r}_{uj}) = \sigma\Big(g\big(\Phi(\mathbf{u}), \Phi(\mathbf{i})\big) - g\big(\Phi(\mathbf{u}), \Phi(\mathbf{j})\big)\Big), \tag{21}$$

$$p(u >_i v|\Theta) = \sigma(\hat{r}_{iu} - \hat{r}_{iv}) = \sigma\Big(g\big(\Phi(\mathbf{i}), \Phi(\mathbf{u})\big) - g\big(\Phi(\mathbf{i}), \Phi(\mathbf{v})\big)\Big). \tag{22}$$

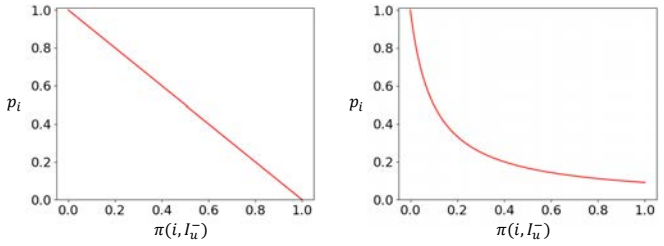
Objective Function: Finally, we introduce the objective function to deal with both user- and item-side pairwise rankings. Similar to classical BPR, our objective function aims to minimize the relative loss for pairwise rankings. In clear contrast, our objective function is to minimize both user- and item-side ranking losses.

$$\begin{aligned} \underset{\Theta}{\text{argmin}} \sum_{(u,i,j)\in\mathcal{D}_u} & -\ln\left(\sigma(\hat{r}_{ui} - \hat{r}_{uj})\right) + \\ \lambda \sum_{(i,u,v)\in\mathcal{D}_i} & -\ln\left(\sigma(\hat{r}_{iu} - \hat{r}_{iv})\right) + \Omega(\Theta), \end{aligned} \tag{23}$$

where $\sigma(\hat{r}_{ui} - \hat{r}_{uj})$ and $\sigma(\hat{r}_{iu} - \hat{r}_{iv})$ are computed by Eq. (21) and Eq. (22), respectively, and λ is the weight to control the importance of item-side triplets. $\Omega(\Theta)$ is the Frobenius regularization term for our parameters $\Theta = \{\mathbf{W}_1, \ldots, \mathbf{W}_L, b_1, \ldots, b_L\}$.

3.2 Dynamic Negative Sampling

One of the main components in BPR is to choose negative items from unknown feedback. The original BPR model [23] is based on a *uniform* negative sampling method. Given user u, it first determines the number of negative items. In general, the number of negative items is proportional to the number of positive items for user u, *i.e.*, $|\mathcal{I}_u^-| = \alpha \cdot |\mathcal{I}_u^+|$. (Usually, α is set to 2–5.) Then, negative items are randomly selected from a set of all unrated items with an *equal* probability. Although this is straightforward, the uniform sampling method is ineffective for improving the quality of BPR.



(a) Linear function (b) Exponential function ($\beta = 10$)

Figure 4: Comparison of linear and exponential functions in the Amazon Music dataset.

To address this problem, a dynamic negative sampling strategy [22, 37] is proposed to maximize *gradient gaps* between positive and negative items. Intuitively, it chooses the negative items that have a substantial prediction loss at each learning step. For the current model, the probability of choosing a negative item is based on a ranked list of all unrated items. That is, the importance $\pi(i, \mathcal{I}_u^-)$ of negative item $i \in \mathcal{I}_u^-$ for dynamic negative sampling is derived by the ranking of negative items.

$$\pi(i, \mathcal{I}_u^-) = \frac{rank(i)}{|\mathcal{I}_u^-|}, \tag{24}$$

where $rank(i)$ is the ranking order of item i in $\mathcal{I}_u^-$. If item i has the highest ranking in $\mathcal{I}_u^-$, $\pi(i, \mathcal{I}_u^-)$ is $1/|\mathcal{I}_u^-|$. That is, when an item is ranked highest, $\pi(i, \mathcal{I}_u^-)$ is lowest. Then, the probability p_i of selecting negative item $i \in \mathcal{I}_u^-$ is computed by a *linear* function.

$$p_i \propto 1 - \pi(i, \mathcal{I}_u^-). \tag{25}$$

We modify this dynamic negative sampling by replacing the linear function with an exponential function. Figure 4 illustrates the linear and exponential function for dynamic negative sampling. Under this adaption, the items with high rankings have a much higher probability than negative items. The probability of selecting item i is rewritten as follows:

$$p_i \propto \frac{1}{\exp(\beta\pi(i, \mathcal{I}_u^-))}, \tag{26}$$

where β is the parameter used to control the slope of the exponential function. Note that our dynamic sampling can also be performed for both user- and item-side triplets.

3.3 Training and Prediction

Algorithm 1 describes the detailed procedure for training the proposed model. To update the parameters on each layer, we use back-propagation by adopting mini-batch gradient descent. For each epoch, we shuffle a training dataset for the user- and item-side triplets. In Section 4.1, we will explain the detailed parameter settings for training our model.

Algorithm 1: Training procedure of DualNPR.

Input: A user-item matrix $\mathbf{R}$, the number of epochs T
Parameters: $\Theta = \{\mathbf{W}_1, \ldots, \mathbf{W}_L, b_1, \ldots, b_L\}$

Randomly initialize parameters Θ.
for $t \leftarrow 1$ **to** T **do**
  for $u \in \mathcal{U}$ *and* $i \in \mathcal{I}$ **do**
    Set negative samples from $\mathcal{I}_u^-$ and $\mathcal{U}_i^-$ using Eq. (26).
  end
  Randomly sample a mini-batch $\mathcal{S} \subseteq \mathcal{D}_u$ and $\mathcal{S}' \subseteq \mathcal{D}_i$.
  for $(u, i, j) \in \mathcal{S}$ **do**
    Compute $\sigma(\hat{r}_{ui} - \hat{r}_{uj})$ using Eq. (21).
  end
  for $(i, u, v) \in \mathcal{S}'$ **do**
    Compute $\sigma(\hat{r}_{iu} - \hat{r}_{iv})$ using Eq. (22).
  end
  Update Θ with the pairwise ranking error in Eq. (23).
end

After training our model, we need to calculate the scores for all unrated items $\mathcal{I}_u^-$ for user u for top-N recommendation. Given user u and every unrated item $i \in \mathcal{I}_u^-$, the user's preference score $\hat{r}_{ui}$ is computed from the proposed model. In this case, we simply feed a user-item pair (u, i) to a substructure of the proposed model. Because our model shares the same learning parameters for a triplet input, we can simply obtain predicted score $\hat{r}_{ui}$ by using the substructure. After all unrated items for user u are passed through our model, they are sorted by descending order. Finally, we select the top-N items with the highest scores.

3.4 Extensions to BPR Variants

We explain how to extend our dual representation to several BPR variants. First, BPR with confidence (BPRC) [31] adds a confidence weight for each implicit feedback on top of the BPR framework. The objective function is defined as:

$$\underset{\Theta}{\operatorname{argmin}} \sum_{(u,i,j)\in\mathcal{D}_u} -\ln\left(c_{uij}\sigma(\hat{r}_{ui} - \hat{r}_{uj})\right) + \Omega(\Theta), \tag{27}$$

where c_{uij} is a confidence weight for (u, i, j). Our idea of dual representation can be extended to BPRC; adding the confidence for user- and item-side triplets can effectively achieve such an extension. Let c_{uij} and c_{iuv} denote confidence weights for (u, i, j) and (i, u, v), respectively. The objective of BPRC with our dual representation is formulated as:

$$\begin{aligned}\underset{\Theta}{\operatorname{argmin}} &\sum_{(u,i,j)\in\mathcal{D}_u} -\ln\left(c_{uij}\sigma(\hat{r}_{ui} - \hat{r}_{uj})\right) + \\ &\lambda \sum_{(i,u,v)\in\mathcal{D}_i} -\ln\left(c_{iuv}\sigma(\hat{r}_{iu} - \hat{r}_{iv})\right) + \Omega(\Theta).\end{aligned} \tag{28}$$

We further extend our model to adaptive BPR (ABPR) [20] with different implicit feedback. The objective of ABPR with our dual representation is written as follows:

$$\begin{aligned}\underset{\Theta}{\operatorname{argmin}} &\sum_{(u,i,j)\in\mathcal{D}_u} -\ln\left(c_{uij}\sigma(\hat{r}_{ui} - \hat{r}_{uj})\right) + \\ &\lambda_{\mathcal{E}} \sum_{(u,i,j)\in\mathcal{E}_u} -\ln\left(c'_{uij}\sigma(\hat{r}_{iu} - \hat{r}_{iv})\right) + \Omega(\Theta),\end{aligned} \tag{29}$$

where $\mathcal{E}_u$ is a set of user-side triplets collected from another implicit feedback dataset, and $\lambda_{\mathcal{E}}$ is the regularization term for $\mathcal{E}_u$. Analogous to Eq. (28), we simply add item-side triplets to two different implicit feedback datasets in Eq. (29).

Lastly, we discuss how to apply our dual representation to NPR and contextual NPR. For NPR, we can replace input vectors in Eq. (7) with concatenated user-item vectors. Then, their input vectors are modified by:

$$\mathbf{p_u} = \mathbf{W}[\mathbf{u}; \mathbf{0}], \quad \mathbf{q_i} = \mathbf{W}[0; \mathbf{i}], \quad \mathbf{q_j} = \mathbf{W}[0; \mathbf{j}], \tag{30}$$

where $\mathbf{W}$ indicates one embedding matrix for users/items. (In Section 4, we will evaluate the extension of NPR by adding our dual representation.)

Similarly, our dual representation can also be extended to contextual NPR. Let $\mathbf{e}_u^c$ and $\mathbf{e}_i^c$ denote a user context vector and an item context vector, respectively. For input vectors in Eq. (10), we append two contextual vectors to a user-item input vector, *i.e.*, $[\mathbf{u}; \mathbf{e}_u^c; \mathbf{i}; \mathbf{e}_i^c]$. Then, a user vector is represented by setting an item vector and its context as zeros. Likewise, an item vector is represented by setting a user vector and its context as zeros.

$$\mathbf{u} = [\mathbf{R}_{u*}; \mathbf{e}_u^c; \mathbf{0}; \mathbf{0}], \quad \mathbf{i} = [\mathbf{0}; \mathbf{0}; \mathbf{R}_{*i}; \mathbf{e}_i^c], \quad \mathbf{j} = [\mathbf{0}; \mathbf{0}; \mathbf{R}_{*j}; \mathbf{e}_j^c] \tag{31}$$

Under this dual representation, we can also share one embedding layer for users/items with contexts.

Table 1: Statistics of three real-world datasets.

Statistics	AMusic	MLLatest	ML1M
# of users	2,831	671	6,040
# of items	13,410	9,066	3,706
# of ratings	63,064	100,004	1,000,208
Sparsity (%)	99.83%	98.36%	95.53%

4 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate the effectiveness of the proposed model. First, we describe the data preparation and experimental setup. Then, we compare DualNPR with several competitive models, including the state-of-the-art recommendation model. Also, we carry out an ablation study to investigate the effect of dual representation, that of negative sampling, that of scoring functions, and finally that of DMF.

4.1 Experimental Setup

Datasets. We perform extensive experiments over three real-world datasets including Amazon Music (AMusic), MovieLens Latest (MLLatest) and MovieLens 1M (ML1M). These are all publicly available

Table 2: Performance comparisons between DualNPR and other baseline models (best in bold).

Datasets	Models	Measures						
		HR@25	HR@50	HR@100	NDCG@25	NDCG@50	NDCG@100	AUC
AMusic	BPR	0.0797	0.1200	0.1754	0.0311	0.0342	0.0477	0.6866
	NPR	0.0454	0.0684	0.1111	0.0163	0.0207	0.0276	0.7118
	Dual+NPR	0.0678	0.1053	0.1590	0.0254	0.0326	0.0413	0.8000
	DualNPR	**0.1488**	**0.2136**	**0.2992**	**0.0580**	**0.0704**	**0.0843**	**0.8780**
	Gain (%)	+86.7	+78.0	+70.6	+86.5	+105.8	+76.7	+23.3
MLLatest	BPR	0.1770	0.2248	0.2820	0.0798	0.0889	0.0982	0.7325
	NPR	0.1463	0.2054	0.2879	0.0572	0.0731	0.0819	0.8365
	Dual+NPR	0.1580	0.2221	0.2683	0.0579	0.0703	0.0779	0.6639
	DualNPR	**0.2158**	**0.2903**	**0.3851**	**0.0914**	**0.1056**	**0.1210**	**0.9244**
	Gain (%)	+21.9	+29.1	+33.8	+14.5	+18.8	+23.2	+10.5
ML1M	BPR	0.1488	0.2181	0.3166	0.0589	0.0723	0.0882	0.8370
	NPR	0.1454	0.2181	0.3172	0.0567	0.0706	0.0866	0.8672
	Dual+NPR	0.1454	0.2197	0.3113	0.0583	0.0726	0.0873	0.8240
	DualNPR	**0.2292**	**0.3290**	**0.4556**	**0.0994**	**0.1185**	**0.1390**	**0.9115**
	Gain (%)	+54.0	+49.7	+43.6	+68.8	+63.2	+57.6	+5.1

on their websites [1] [2]. Because we are interested in the recommendation models with implicit feedback, all of the observed feedback was regarded as positive feedback. For Movielens datasets, we do not apply any preprocessing. For AMusic, because it is notorious for high sparsity, we filtered out unreliable ratings; users who had less than 10 ratings and items that had been rated by less than five users were eliminated as has been done in existing studies [8, 33, 34]. Table 1 reports the detailed statistics of the three datasets.

Competitive Models. We compared DualNPR with two existing models and NPR with our dual representation.

- **BPR** [23]: This model minimizes a pairwise ranking loss for user-side triplets using simple matrix factorization.
- **NPR** [18]: This is a state-of-the-art BPR variant that combines BPR with DNNs.
- **Dual+NPR**: Using our dual representation, we extend NPR to utilize both user- and item-side triplets.

Although several models [7, 8, 34, 36] utilize DNNs for personalized ranking, these models belong to the point-wise approach, which directly predicts the absolute rating. In this paper, we choose BPR and NPR as the competitive models because both of them aim to optimize a pairwise ranking loss. Also, because some BPR variants [4, 6] have exploited DNNs as a feature extractor for additional contexts and inputs for graphs, it is inappropriate to compare our model with them.

Evaluation Protocol. To assess the accuracy of top-N recommendation, we adopted the *leave-one-out evaluation*, which has been widely used in the literature [7, 8, 34]. We held-out one random user-item interaction as the test item for each user, and the rest of user-item interactions were used for the training dataset. Unlike sampling-based evaluation tasks [7, 8, 34] that randomly choose 100 items that have not been rated by the user, we choose all items unrated by each user as the candidate items for top-N recommendation. Because there are many unrated items for each user, our evaluation protocol should handle much more challenging cases for top-N recommendation. We believe that our evaluation protocol is more suitable for conducting a fair evaluation for top-N recommendation because it essentially prevents sampling bias during evaluation.

Evaluation Measures. For each model, we reported the accuracy of top-N recommendation with three metrics: hit rate (HR), normalized discounted cumulative gain (NDCG) and area under the ROC curve (AUC). These metrics have been widely used in existing studies [7, 8, 23, 34, 36]. The size of the ranked list N is chosen as 25, 50, and 100. While the HR@N intuitively measures whether a test item is present in the top-N list, the NDCG@N puts more weight on highly ranked hit items than others in the top-N list. Lastly, AUC accounts for the relative rank of the test item among all unrated items. Essentially, AUC is optimized by BPR [23]. Note that all three measures are averaged across all users. The higher they are, the more accurate a recommendation model is.

Implementation Details. We implemented DualNPR and other existing models using TensorFlow. We randomly initialized model parameters with Xavier initializer [5]. For all baseline models, the number of latent dimensions was set to 128. The optimizer for the baseline models was mini-batch Adagrad. We optimized the models by adjusting the batch size of 32 and the learning rate of 0.005. The regularization term for the learning parameters was set to 0.0001. For DualNPR, the weight λ of user-item triplets was set to 0.6 (AMusic) and 0.8 (MLLatest and ML1M). All experiments were performed on a desktop with 128 GB memory and 2 Intel Xeon Processors (E5-2630 v4, 2.20 GHz, 25M cache), and all models were trained using 4 Nvidia GeForce GTX 1080Ti.

[1] https://grouplens.org/datasets/movielens
[2] http://jmcauley.ucsd.edu/data/amazon/

Table 3: Effects of dual representation in DualNPR (best in bold).

Datasets	Dual rep.	Measures						
		HR@25	HR@50	HR@100	NDCG@25	NDCG@50	NDCG@100	AUC
AMusic	No	0.0783	0.1227	0.1856	0.0291	0.0376	0.0478	0.8103
	Yes	**0.1122**	**0.1707**	**0.2511**	**0.0451**	**0.0561**	**0.0691**	**0.8706**
	Gain (%)	+43.3	+39.1	+35.3	+55.0	+49.2	+44.6	+7.4
MLLatest	No	0.1431	0.2066	0.2918	0.0571	0.0692	0.0829	0.8964
	Yes	**0.2158**	**0.2903**	**0.3851**	**0.0914**	**0.1056**	**0.1210**	**0.9244**
	Gain (%)	+50.8	+40.5	+32.0	+60.1	+52.6	+46.0	+3.1
ML1M	No	0.2090	0.3034	0.4236	0.0859	0.1042	0.1234	0.9042
	Yes	**0.2292**	**0.3290**	**0.4556**	**0.0994**	**0.1185**	**0.1390**	**0.9115**
	Gain (%)	+9.7	+8.4	+7.6	+15.7	+13.7	+12.6	+0.8

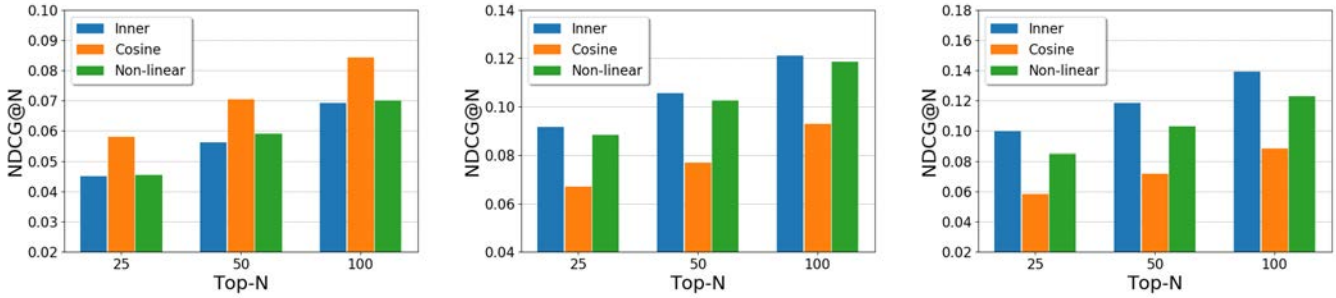
Table 4: Effects of negative sampling in DualNPR (best in bold).

Datasets	Functions	Measures						
		HR@25	HR@50	HR@100	NDCG@25	NDCG@50	NDCG@100	AUC
AMusic	Random	0.0988	0.1495	0.2239	0.0389	0.0486	0.0606	0.8535
	Linear	0.1441	0.2043	0.2894	0.0567	0.0683	0.0821	0.8756
	Exponential	**0.1488**	**0.2136**	**0.2992**	**0.0580**	**0.0704**	**0.0843**	**0.8780**
	Gain (%)	+3.3	+4.6	+3.4	+2.3	+3.1	+2.7	+0.3
MLLatest	Random	0.1878	0.2626	0.3589	0.0747	0.0891	0.1046	0.9090
	Linear	0.2137	0.2858	0.3802	**0.0941**	**0.1079**	**0.1232**	0.9217
	Exponential	**0.2158**	**0.2903**	**0.3851**	0.0914	0.1056	0.1210	**0.9244**
	Gain (%)	+1.0	+1.6	+1.3	-2.9	-2.1	-1.8	+0.3
ML1M	Random	0.1612	0.2427	0.3501	0.0641	0.0798	0.0971	0.8813
	Linear	0.2173	0.3110	0.4319	0.0915	0.1098	0.1294	0.9040
	Exponential	**0.2292**	**0.3290**	**0.4556**	**0.0994**	**0.1185**	**0.1390**	**0.9115**
	Gain (%)	+5.5	+5.8	+5.5	+8.6	+7.9	+7.4	+0.8

4.2 Experimental Results

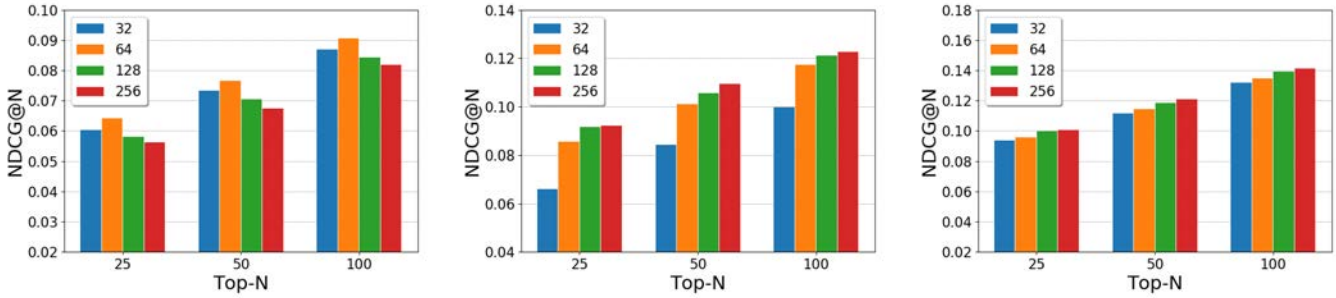
DualNPR vs. Other Competitors. We evaluate the effectiveness of DualNPR in comparison with two baseline models and NPR with dual representation. In DualNPR, by default, while we used cosine similarity as the similarity function for AMusic, we used inner product as the similarity function for MLLatest and ML1M. As shown in Table 2, we found three key observations: (1) Regardless of datasets and measures, DualNPR significantly outperformed the baseline models. DualNPR was consistently better than the baseline models by 21.9–86.7% in HR, 14.5–105.8% in NDCG, and 5.1–23.3% in AUC. For the experimental results of NPR and BPR, our results show a similar tendency as reported in [18]. This achievement is meaningful in the sense that we have not added any auxiliary information. (2) Modifying NPR with our dual representation is beneficial at improving the accuracy of the original NPR. In AMusic, we observed that Dual+NPR was improved by 43.1–49.3% in HR and 49.6–57.4% in NDCG, and 12.3% in AUC, respectively. Although Dual+NPR is less accurate than DualNPR, it significantly outperformed NPR. (3) Lastly, DualNPR consistently outperforms Dual+NPR: 36.5–57.6% in HR, 50.2–70.5% in NDCG, and 10.6–39.2% in AUC. This improvement comes from DMF and negative sampling. Therefore, we can conclude that our choice of DMF and negative sampling is effective to improve the accuracy of top-N recommendation.

Effects of Dual Representation. We investigated the effects of dual representation in DualNPR. For three datasets, we consistently used inner product as the similarity function. As shown in Table 3, item-side triplets are useful to improve the accuracy of DualNPR: 7.6–50.8% in HR, 12.6–60.1% in NDCG, and 0.8–7.4% in AUC. Analyzing the results across multiple datasets, we found that the improvement owing to dual representation varies on the characteristics of the datasets. To demonstrate the significance of improvement, we computed the gain, which is the absolute difference of accuracy divided by the baseline accuracy. On average, the gain of MLLatest was 41.1% in HR, 52.9% in NDCG, and 3.1% in AUC. Likewise, the average gain of AMusic was 39.3% in HR, 49.6% in NDCG, and 7.4% in AUC. Meanwhile, the average gain of ML1M was 8.6% in HR, 14.0% in NDCG, and 0.8% in AUC. We observed that the performance gain in AMusic and MLLatest were much higher than those of ML1M. We believe that such a gap in improvement is strongly correlated with the sparsity of the datasets; AMusic (99.83%) and MLLatest (98.86%) are much more sparse than ML1M (95.53%). Therefore, we confirmed that our dual representation is beneficial to address the data sparsity problem.



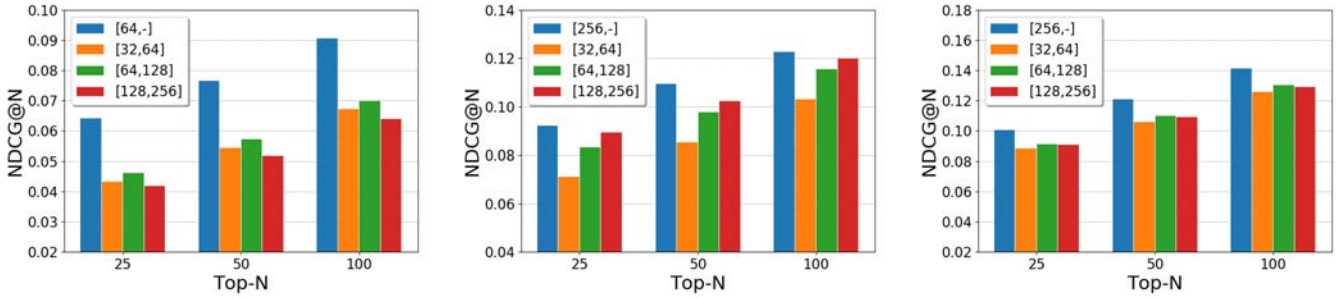
(a) AMusic (b) MLLatest (c) ML1M

Figure 5: Comparisons of varying scoring functions in DualNPR.



(a) AMusic (b) MLLatest (c) ML1M

Figure 6: Comparisons of varying the size of embedding dimensions in DualNPR.



(a) AMusic (b) MLLatest (c) ML1M

Figure 7: Comparisons of varying the depth of hidden layers in DualNPR.

Effects of Negative Sampling. Our dynamic negative sampling method using an exponential function was compared with two baselines: random sampling and dynamic negative sampling using a linear function. As shown in Table 4, our sampling strategy is most effective at improving the accuracy of DualNPR for all three different datasets: 1.0–5.8% in HR, -2.9–8.6% in NDCG, and 0.3–0.8% in AUC. This means that our dynamic sampling with an exponential function tends to increase the gradient gap between positive and negative items, by selectively choosing negative items with high rankings from unobserved feedback. Compared to random sampling, our sampling method shows significant improvement gains by 7.3–50.6% in HR, 15.7–55.1% in NDCG, and 1.7–3.4% in AUC. Investigating these results over different datasets, we observe that our sampling strategy is more useful in handling the sparser dataset, such as AMusic. This justifies our choice of using the exponential function as it handles the sparse dataset more adequately, and improves the accuracy of top-N recommendation.

Effects of DMF. To determine the main components of DMF, we investigated various scoring functions, the size of embedding dimensions for users/items, and the depth of hidden layers. We observed several interesting results that were opposed to our expectations.

First, Figure 5 depicts the accuracy over three scoring functions in three datasets. It was observed that both inner product (MLLatest and ML1M) and cosine similarity (AMusic) are better than a non-linear inner product. We find that the choice of optimal scoring function varies on the dataset; it significantly influenced the accuracy. Although the non-linear scoring function is less useful in the three datasets, we still believe that the non-linear scoring function can be useful for other large-scale datasets.

Next, Figure 6 illustrates the accuracy at four different sizes of embedding dimension over the three datasets. The higher the dimension, the lower the accuracy. When the dimension was 64 in AMusic and 256 in both MLLatest and ML1M, we achieved the highest accuracy. The optimal size of the embedding dimensions tends to be closely related to the number of model parameters and the amount of positive feedback for each dataset. Note that the size of input representation multiplied by the size of embedding dimension decides the number of parameters in our model. Hence, when the size of input vector for three datasets was 16k, 9.7k, and 9.7k, respectively, the number of model parameters for three datasets and ML1M is 1M, 2.4M, and 2.4M, respectively. The optimal model parameters for AMusic are much less than MLLatest and ML1M because the amount of positive feedback is much less than other datasets. In other words, this is because, given the limited amount of positive feedback, increasing the model parameters can incur an overfitting problem.

Lastly, Figure 7 depicts the accuracy at various depths of hidden layers over the three datasets. The deeper the layer's depth, the lower the accuracy of DualNPR. From these observations, we conjecture that the correlations among users/items may not be highly complex and non-linear. In future work, we will investigate the effects of DMF over various large-scale datasets.

5 RELATED WORK

Much of the attention on personalized recommendation systems [24, 27] has shifted toward collaborative filtering (CF) utilizing implicit user feedback for top-N recommendation. Because an implicit dataset provides extremely sparse one-class feedback, it is inherently challenging to infer hidden user preferences from implicit feedback. This is also known as the one-class collaborative filtering (OCCF) problem [19]. To address this problem, existing work can be categorized into two groups: *point-wise* and *pair-wise* approaches.

5.1 Point-wise Approach

The point-wise approach aims at minimizing the absolute differences between actual values and predicted values. One of the popular models is weighted matrix factorization [10, 19]. Recently, with the success of DNNs [15], several studies [12, 35] have attempted to combine matrix factorization with DNNs such as autoencoders (AE) [16, 26, 28, 33], multi-layer perceptions (MLP) [1, 2, 8, 34, 36], convolutional neural networks (CNN) [7, 13], and recurrent neural networks (RNN) [9, 32]. Specifically, Wang et al. [28] combined a latent factor model with the autoencoders and Kim et al. [13] proposed a hybrid CF model by extracting textual features using CNN. Besides, Cheng et al. [1] and Covington et al. [2] proposed a wide-and-deep model where user features, item features, and additional contents are concatenated as input vectors. For sequential recommendation, Hidasi et al. [9] and Wu et al. [32] proposed RNN-based recommendation models. Although they outperform existing simple matrix factorization models, they have not optimized pairwise ranking for top-N recommendation.

5.2 Pair-wise Approach

The pair-wise approach aims to minimize a pairwise ranking loss in the ranked list. This approach is also used in other research domains such as computer vision [29, 30], information retrieval [11] and, natural language processing [3, 17]. Among recommendation models, the most representative model is BPR [23]. The objective function of BPR is to minimize a pairwise ranking loss of user-side triplets elicited from implicit user feedback. We categorize the variants of BPR into the following two groups.

BPR from Multiple Implicit Feedback. Pan et al. [20] extended BPR to accommodate different implicit feedback by adapting the confidence level of pairwise preferences. Later on, Qiu et al. [21] investigated the correlation between auxiliary feedback and target feedback and proposed an integrated model by considering the characteristics of different implicit feedback. Although this is valid for improving the quality of original BPR, they did not necessarily require heterogeneous implicit feedback. In contrast, our model naturally augments user- and item-side pairwise rankings from a user-item interaction matrix.

Integrating BPR with DNNs: As a hybrid approach, He et al. [6] proposed a visual BPR, where CNN is used for extracting visual features. Ding et al. [4] combined BPR with DNNs by extracting user features from social networks using CNN. Most recently, Niu et al. [18] proposed NPR by combining BPR with DNNs, and extended NPR to incorporate contextual information. These existing studies in this category [4, 6, 18] are most relevant to ours because they also utilize DNNs to improve the modeling power of BPR. However, we believe that our model can be viewed as a more generalized BPR model in the sense that we handle both user- and item-side pairwise rankings without any additional information. In particular, our model can be easily extended to other BPR variants.

6 CONCLUSION

In this paper, we have introduced a new CF model that uses implicit user feedback for top-N recommendation. To achieve this goal, we have proposed DualNPR that fully exploits both user- and item-side pairwise rankings using DNNs. The proposed model addresses the following key challenges faced in the existing work. (1) DualNPR overcomes the data sparsity problem by leveraging both user- and item-side triplets collected from a user-item interaction matrix. (2) DualNPR leverages DMF to learn the variability of latent user/item representations. (3) DualNPR effectively chooses negative feedback from unknowns by introducing dynamic negative sampling with an exponential function.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2018R1A2B6009135, NRF-2018R1A5A1081213, and NRF-2018R1A5A 1060031).

REFERENCES

[1] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *International Workshop on Deep Learning for Recommender Systems DLRS@RecSys*. 7–10.

[2] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *ACM International Conference on Recommender Systems (RecSys)*. 191–198.

[3] Arpita Das, Harish Yenala, Manoj Kumar Chinnakotla, and Manish Shrivastava. 2016. Together we stand: Siamese Networks for Similar Question Retrieval. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

[4] Daizong Ding, Mi Zhang, Shao-Yuan Li, Jie Tang, Xiaotie Chen, and Zhi-Hua Zhou. 2017. BayDNN: Friend Recommendation with Bayesian Personalized Ranking Deep Neural Network. In *ACM International Conference on Information and Knowledge Management (CIKM)*. 1479–1488.

[5] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics, AISTATS*. 249–256.

[6] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *International Conference on Artificial Intelligence (AAAI)*. 144–150.

[7] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer Product-based Neural Collaborative Filtering. In *International Joint Conference on Artificial Intelligence (IJCAI)*. 2227–2233.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *International Conference on World Wide Web (WWW)*. 173–182.

[9] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based Recommendations with Recurrent Neural Networks. *CoRR* abs/1511.06939 (2015).

[10] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *IEEE International Conference on Data Mining (ICDM)*. 263–272.

[11] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry P. Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *ACM International Conference on Information and Knowledge Management (CIKM)*. 2333–2338.

[12] Alexandros Karatzoglou and Balázs Hidasi. 2017. Deep Learning for Recommender Systems. In *ACM International Conference on Recommender Systems (RecSys)*. 396–397.

[13] Dong Hyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional Matrix Factorization for Document Context-Aware Recommendation. In *ACM International Conference on Recommender Systems (RecSys)*. 233–240.

[14] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 426–434.

[15] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

[16] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep Collaborative Filtering via Marginalized Denoising Auto-encoder. In *ACM International Conference on Information and Knowledge Management (CIKM)*. 811–820.

[17] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. 2016. Learning Text Similarity with Siamese Recurrent Networks. In *International Workshop on Representation Learning for NLP (Rep4NLP@ACL)*. 148–157.

[18] Wei Niu, James Caverlee, and Haokai Lu. 2018. Neural Personalized Ranking for Image Recommendation. In *ACM International Conference on Web Search and Data Mining (WSDM)*. 423–431.

[19] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan M. Lukose, Martin Scholz, and Qiang Yang. 2008. One-Class Collaborative Filtering. In *IEEE International Conference on Data Mining (ICDM)*. 502–511.

[20] Weike Pan, Hao Zhong, Congfu Xu, and Zhong Ming. 2015. Adaptive Bayesian personalized ranking for heterogeneous implicit feedbacks. *Knowl.-Based Syst.* 73 (2015), 173–180.

[21] Huihuai Qiu, Yun Liu, Guibing Guo, Zhu Sun, Jie Zhang, and Hai Thanh Nguyen. 2018. BPRH: Bayesian personalized ranking for heterogeneous implicit feedback. *Inf. Sci.* 453 (2018), 80–98.

[22] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *ACM International Conference on Web Search and Data Mining (WSDM)*. 273–282.

[23] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *International Conference on Uncertainty in Artificial Intelligence (UAI)*. 452–461.

[24] Francesco Ricci, Lior Rokach, and Bracha Shapira (Eds.). 2015. *Recommender Systems Handbook*. Springer.

[25] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *Annual Conference on Neural Information Processing Systems (NIPS)*. 1257–1264.

[26] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. AutoRec: Autoencoders Meet Collaborative Filtering. In *International Conference on World Wide Web Companion (WWW)*. 111–112.

[27] Koen Verstrepen, Kanishka Bhaduri, Boris Cule, and Bart Goethals. 2017. Collaborative Filtering for Binary, Positiveonly Data. *SIGKDD Explorations* 19, 1 (2017), 1–21.

[28] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1235–1244.

[29] Liwei Wang, Yin Li, and Svetlana Lazebnik. 2016. Learning Deep Structure-Preserving Image-Text Embeddings. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5005–5013.

[30] Liwei Wang, Yin Li, and Svetlana Lazebnik. 2017. Learning Two-Branch Neural Networks for Image-Text Matching Tasks. *CoRR* abs/1704.03470 (2017).

[31] Sheng Wang, Xiaobo Zhou, Ziqi Wang, and Ming Zhang. 2012. Please spread: recommending tweets for retweeting with implicit feedback. In *International Workshop on Data-driven User Behavioral Modelling and Mining from Social Media (DUBMMSM)*. 19–22.

[32] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent Recommender Networks. In *ACM International Conference on Web Search and Data Mining (WSDM)*. 495–503.

[33] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. 2016. Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. In *ACM International Conference on Web Search and Data Mining (WSDM)*. 153–162.

[34] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep Matrix Factorization Models for Recommender Systems. In *International Joint Conference on Artificial Intelligence (IJCAI)*. 3203–3209.

[35] Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep Learning based Recommender System: A Survey and New Perspectives. *CoRR* abs/1707.07435 (2017).

[36] Shuai Zhang, Lina Yao, Aixin Sun, Sen Wang, Guodong Long, and Manqing Dong. 2018. NeuRec: On Nonlinear Transformation for Personalized Ranking. In *International Joint Conference on Artificial Intelligence (IJCAI)*. 3669–3675.

[37] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *International ACM SIGIR conference on research and development in Information Retrieval (SIGIR)*. 785–788.

Studying Preferences and Concerns about Information Disclosure in Email Notifications

Yongsung Kim*
Northwestern University
Evanston, IL
yk@u.northwestern.edu

Adam Fourney
Microsoft Research
Redmond, WA
adamfo@microsoft.com

Ece Kamar
Microsoft Research
Redmond, WA
eckamar@microsoft.com

ABSTRACT

People receive dozens, or hundreds, of notifications per day and each notification poses some risk of accidental information disclosure in the presence of others; onlookers may see notifications on a mobile phone lock screen, on the periphery of a desktop or laptop display. We quantify the prevalence of these accidental disclosures in the context of email notifications, and we study people's relevant preferences and concerns. Our results are compiled from a retrospective survey of 131 respondents, and a contextual-labeling study where 169 participants labeled 1,040 meeting-email pairs. We find that, for 53% of people, at least 1 in 10 email notifications poses an information disclosure risk, and the real or perceived severity of these risks depend both on user characteristics and the meeting or email attributes. We conclude by exploring machine learning for predicting people's comfort levels, and we present implications for the design of future social-context aware notification systems.

CCS CONCEPTS

• **Security and privacy** → **Social aspects of security and privacy**; • **Human-centered computing** → *Empirical studies in ubiquitous and mobile computing*.

KEYWORDS

Notifications, information disclosure, privacy, virtual assistants

ACM Reference Format:
Yongsung Kim, Adam Fourney, and Ece Kamar. 2019. Studying Preferences and Concerns about Information Disclosure in Email Notifications. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313451

1 INTRODUCTION

Estimates suggest that people receive dozens, or hundreds, of notification messages per day [21, 30, 31] delivered to a range of connected devices that people carry with them, or that are ever-present in the environment (e.g., wearables, smartphones, computers or – increasingly – internet of things devices). Extensive prior research has explored the productivity costs of mal-timed notifications [13, 20, 21], but little is known about the privacy cost of such messages; people are frequently in the presence of others when notifications arrive, and, in these contexts, each notification poses some risk of accidental information disclosure. For example, email notifications often reveal the sender and subject fields of messages, and may be visible above the lock screen of a mobile phone (Figure 1a), on the periphery of a co-located desktop or laptop (Figure 1b), on a projection screen during a presentation, or on a smart TV [38]. As Susan Farrell writes in [9]:

> "By making smart devices ubiquitous, we've exposed ourselves to computer-assisted embarrassment. We must expand our usability methods to cover not only the isolated user in one context of use, but also the social user, who interacts with the system in the presence of others."

With the ultimate goal of designing more contextually-relevant notification strategies, we sought to understand the information disclosure risks arising from notifications received when people are in the presence of others. Specifically, we ask the following research questions:

- **RQ1:** How often do email notifications pose an information disclosure risk? I.e., How often are emails received while in the presence of others, and how often do these emails contain sensitive information?
- **RQ2:** To what degree are people's preferences and concerns dependent on user characteristics (personal preferences, job role, etc.) versus being context-dependent with respect to the content disclosed by the notification, and the people in the room?

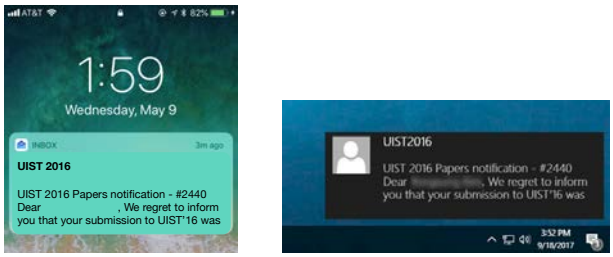


Figure 1: An example of an email notification on an iPhone (a) and on a computer with Windows 10 (b). The notification reveals the email's sender, subject, and first sentence from the message body. When in the presence of others (e.g, during meetings), users may be uncomfortable receiving notifications and revealing these data to onlookers.

*Work done while at Microsoft Research.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313451

- **RQ3:** To what degree can a machine-learned system anticipate information disclosure risks? If such predictions are possible, what features are important?

To answer RQ1 and RQ2, we conducted two studies in an enterprise environment grounded in the scenario of a person receiving an email notification on their notification-capable devices while attending a meeting. We focus on emails because they are the primary source of notifications in this environment: 86% of professionals cite email as their primary means of communication, and an average 112.5 billion business emails were exchanged per day in 2015 [14]. Moreover, email often carries notifications for other services (Twitter, Facebook, Slack), suggesting that our findings are likely to provide insights about notifications generated by other communication platforms.

We first conducted an exploratory retrospective online survey (N=131) that asked respondents about hypothetical situations involving recent emails and meetings. From this survey we gained an understanding about the existence and prevalence of the accidental information disclosure problem, and we learned about people's preferences and concerns.

Findings from the retrospective survey guided design decisions for the second study, which provides a deeper, data-centric investigation into information disclosure risks in enterprise settings. This study employed a custom-built labeling tool, to collect labels and preferences for email-meeting pairs. The labeling tool integrated directly with participants' email and calendaring accounts (N=169), allowing the tool to systematically identify emails that were actually received during meetings, focusing labeling on non-hypothetical cases. For each email-meeting pair, the tool automatically extracts features summarizing high-level characteristics of the user, the content of the email message, and the properties of the meeting; and, it does so in a manner that maintains participant anonymity.

Finally, to answer RQ3, we leverage the data collected by the aforementioned contextual labeling tool to explore feature sets and machine learning algorithms to predict information disclosure risks. Here we explore: (a) features that are associated with users (e.g. job title); (b) features that describe a particular email message (e.g. number of recipients in an email); and (c) features that describe a meeting instance (e.g., whether or not a manager is present).

The rest of the paper is structured as follows: We review related work, then discuss the findings of the exploratory retrospective survey. We describe the second study, which involved the deployment of a custom contextual-labeling tool, then describe relevant findings. Finally, we explore machine learning algorithms to predict people's comfort levels, and we present implications for the design of future social-context aware notification systems.

2 RELATED WORK

2.1 Productivity Cost in Notifications

Previous works in notifications mainly focused on the negative impact of interruptions on productivity, and seek to identify low-cost interruptible moments to send notifications. Studies show that the cognitive load of current tasks [6, 20], task transitions [11, 16], physical activities, user interactions on devices [27], and other context such as time and location [29] can be used to identify interruptible moments for notifications. In office settings, for example, Fogarty et al. [13] show that sensor-based statistical models can be used to predict one's interruptibility by leveraging features such as: phone use, ambient noise (e.g., to detect conversations), mouse and keyboard usage. Likewise, Horvitz et al. [18] use computer activity, calendar information, audio and video signals, and location data to predict users' availability. Notification policies then use inferred levels of availability and interruptibility to determine the ideal time to deliver notifications. Research has also recognized that there can be costs associated with delaying notifications, suggesting that such policies should consider both the cost and benefits of delays [17, 19].

Our work is distinguished in that we study the privacy costs associated with notifications, as opposed to productivity costs. As such, the scenarios we consider are privacy-sensitive, and this has implications for our methodology. We have taken careful steps to respect respondents' privacy in our studies.

2.2 Privacy Cost in Notifications

2.2.1 Privacy Attitudes and Behaviors. A first step in understanding the potential privacy cost of ill-timed notifications is to understand people's general attitudes towards privacy. Pioneering work in this space includes the Westin Privacy Segmentation Index [39], that categorizes people into three groups: privacy fundamentalist, marginally concerned, and pragmatist. *Privacy fundamentalists* are those who are very concerned about their privacy and very reluctant to share any of their information. *Marginally concerned* are generally willing to share details or data about themselves. *Privacy pragmatists* are people who are somewhat concerned about their privacy, but are willing to compromise some privacy for convenience.

Numerous follow-up studies have adopted the Westin Index [1–3, 28], but have reported that people's preferences are nuanced and context sensitive, with complex interactions between the types of disclosures and the audiences who bear witness. For example, Ackerman et al. show that people are generally comfortable sharing their favorite TV shows, favorite snacks, and even email addresses with websites, but are much less comfortable when there is a chance that the website could share the information with others in an identifiable way, or if the information were instead provided by a child under their care [1]. Likewise, Olson et al. show that, even among one's trusted inner circles, one's willingness to disclose information varies greatly depending on the nature of the information being shared [28]. For example, respondents were often uncomfortable disclosing work-related documents with family members, or health information (e.g., pregnancy status) with co-workers.

Adding to this complex motif, past work reports that privacy attitudes are not always correlated with behaviors [2, 26, 33]. For example, participants' behavior in an online shopping scenario was different from their self-reported privacy preferences [33], and one's privacy attitude was a poor predictor of whether participants would share location information [5]. Like privacy attitudes, privacy behaviors are context-specific [5, 12, 40] and multi-dimensional [23].

In this paper we recognize the potential for these complex context-dependent attitudes and behaviors. We designed both of our studies to collect a multitude of signals that characterize how information, audiences, and contexts interact with one another to create situations in which people are uncomfortable (or comfortable) receiving

notifications. Additionally, we designed our second study such that it grounds data collection on specific historical instances rather than asking people about general or hypothetical scenarios and attitudes.

2.2.2 Difference in Disclosures and Privacy Management Strategies. In addition to understanding privacy attitudes in a broader context, it is also important to understand how notifications may differ from the scenarios explored in past work. Here, prior work has mainly examined intentional sharing of information with websites or social networking services (SNSs) [2–4, 36, 37]. Granted that when people disclose their information to these services they don't have perfect knowledge of the consequences, with few exceptions, they have agency in deciding which information to disclose. As such, people often perform an informal risk-benefit analysis before taking actions that may have privacy implications [7, 8, 25, 40]. People also enact preventive strategies, including self-censoring, managing access control groups, and taking actions to conceal their identity [24, 37].

Agency and control is not available in cases where people receive push notifications. Instead, the notification scenario bears resemblance to the scenario in which people are tagged in photos shared to SNSs without consent [24] or posts being unintentionally shared on SNSs as a consequence of *if-this-then-that (IFTTT)* [34]. Here, users can often take corrective actions by either untagging the photos or deleting the contents before more people see the posts. With push notifications, disclosures are instantaneous, and it is unclear what corrective actions can be taken after the event.

A similar situation arises with the accidental disclosure of web browsing history [15]. Through a survey of 155 people, Hawkey et al. found that, as before, comfort level is related to one's level of control (e.g., control over the mouse and keyboard). Moreover, people's comfort is higher when disclosures are to spouses or close friends, and lower with colleagues.

Most closely related to our work is on examining the privacy costs of receiving notifications on a smart TV when watching in a social setting [38]. From a survey of 167 participants, Weber et al. assessed people's comfort levels with different notification variants, and found that the comfort level is higher when the notification reveals less information (e.g. notification indicators). Our work is inspired by these findings, but both extends them to an enterprise setting, and considers a richer set of contexts (e.g. meeting type or location, social structure of people in the room, etc.).

2.2.3 Limitations and Trade-offs in Existing Notification Strategies. Finally, we note that existing mobile devices and applications provide some minimal options for managing notifications. Typically, such options are binary (e.g. turning on and off notifications, enabling notifications for "important" emails in Gmail, and enabling notifications to show previews). Recognizing the privacy risks of notifications, some applications (e.g., WhatsApp) and mobile devices (e.g., iPhone X) have disabled notification previews by default, preferring instead to provide generic indicators (e.g., "You have a new message.") These solutions are context independent, and employ a one-size fits all strategy. Users must accept the trade-offs between minimizing privacy risks (e.g., suppressing notifications) and maintaining timely access to information. In contrast, our work takes into consideration both user contexts and notification contents.

As more and more internet-connected devices and services become capable of displaying information via notifications, there is a need to better understand people's preferences and concerns about notification-induced information disclosures. There is also a need to develop designs and strategies that can adapt both to a notification's content and the user's context.

3 STUDY #1: EXPLORATORY RETROSPECTIVE SURVEY

As a first step, we describe the results of an exploratory retrospective survey which was designed to gain our initial understanding of the following: (1) How often notifications pose an information disclosure risk? (2) How do features of the notifications, meetings and individuals contribute to the perceived risk? As noted in the introduction, the methods employed here allow a broad investigation covering numerous contexts (e.g., personal vs. work, email topics and themes, etc.)

3.1 Procedure

The survey began by collecting basic demographic information including education, job role, age, gender, notification-capable device use, average number of meetings in a day. It then asked participants to answer questions about a notification scenario, which was evolved slowly over several sections of the questionnaire, as follows:

(1) Respondents were first asked to consider their most recent meeting. Respondents answered questions about their role in the meeting, their relation to each of the meeting's attendees, and the meeting's time and location, and the types of notification-capable devices that were present (including Desktop PC, Laptop, Smartphone, Smartwatch, and Smart speaker).
(2) The survey then asked respondents to open their primary work email inbox, consider their 10 most recent emails (excluding the survey invitation), and imagine a scenario in which they received notifications for those emails during the aforementioned meeting. Respondents were asked about the age of the 10^{th} email.[1] Furthermore, we asked participants to imagine that their notification-capable devices' screens were visible to meeting attendees, and answer how many email notifications (out of the ten emails) they would have been uncomfortable sharing with the people in the room. We refer to this scenario as a *hypothetical information disclosure event*, or *HIDE*.
(3) The questions above were then repeated for the 10 most recent emails in respondents' personal email inboxes.
(4) Respondents were then asked to select one email (could be from either work or personal email inbox), if applicable, that they would have been most uncomfortable receiving in the scenario outlined above. We refer to this email as the *HIDE email*. Respondents were asked to rate their comfort level in

[1]We decided to only rely on a respondent's 10 most recent emails to (a) convey a simple and consistent sampling criteria in the retrospective survey, and (b) minimize response and counting errors by allowing respondents to observe their emails within a single window pane (the decision made after pilot testing). To partially compensate for this limitation, we additionally asked the age of 10^{th} email, which enables us to calculate the incoming email rate.

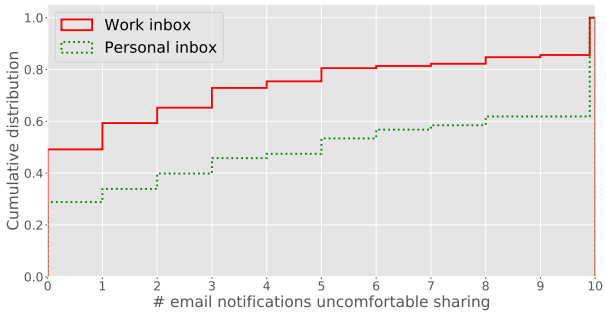


Figure 2: Cumulative distribution of the number of work (red, solid) and personal (green, dotted) emails that people felt uncomfortable sharing with meeting attendees.

sharing the HIDE email notification with the people in the room with a 5-point Likert scale (1: Very uncomfortable, 5: Very comfortable). Respondents then were asked to describe general features of the HIDE email, including their relation to its sender, the number of recipients, the type of the inbox (work or personal), the type of content contained therein, and to explain, in broad terms, why receiving notifications for this email would be uncomfortable.

The survey was deployed by emailing a random sample of 800 employees within a large IT corporation. There were 21 email delivery failures, for various technical reasons, resulting in 779 individuals successfully receiving the invitation. We describe our findings next.

3.2 Results

3.2.1 Participants. We received 118 completed, and 13 partially completed responses (response rate = 17%, completion rate = 90%). Of the 131 total respondents, 85 were male (65%), 44 were female (34%), 2 preferred not to answer. Ages were distributed as follows: 18 – 24 years old (5%), 25 – 34 (24%), 35 – 44 (31%), 45 – 54 (27%), 55 – 64 (8%), ≥ 65 (2%), and 2% declined to answer.

Participants reported occupying a diverse set of job roles including: program managers (28%), software developers (19%), marketing and sales people (8%), and IT support staff (8%). The remaining 49 individuals (37%) worked in diverse roles such as administrative assistants, data scientists, designers, attorneys or other roles in the legal department, and human resources.

3.2.2 Prevalence of the information disclosure risk. One key finding from the retrospective survey, and a partial answer to our first research question, is that the majority of respondents (53.4%) reported that *at least one* of their ten most recent work emails would have resulted in an uncomfortable disclosure of information had the email arrived during their most recent meeting. This increases significantly to 73.3% when respondents were asked to consider the ten most recent emails delivered to their personal accounts (two tailed difference of proportions test, $p << 0.001$, $Z = 3.33$). While these findings demonstrate the potential for risk, we cannot be sure how many emails were actually received during the meetings. We address this limitation in the second study.

3.2.3 Three groups of respondents. Figure 2 extends the above analysis by presenting the cumulative distribution for work emails (red) and for personal emails (green) that participants indicated they would be uncomfortable receiving in their most recent meeting. There are two notable features of these distributions: first, there is a sharp rise at $x = 10$ emails, with 17% of respondents noting they would be uncomfortable receiving *any* work email notification in the presence of others. This increases to 40% for personal emails. We also note that the y-intercepts of the two curves are rather high: 47% of respondents reported that *none* of their ten most recent work emails were sensitive. This decreases to 25% for personal emails. The remaining 36% and 35% of respondents were more selective, and reported that *some* of their work and personal emails were sensitive, respectively.

On the surface, these three groups bear some resemblance to Westin's three categories, but we are sensitive to the fact that privacy concerns are complex and multidimensional. The responses we collected may be dependant on user characteristics (e.g. personal preferences, the nature of their occupation), contents in the notifications, and the contexts and their relationship with the people in the meeting. For example, one participant explained that she was uncomfortable sharing the information in notifications due to the nature of her occupation:

> "*I deal with a lot of privileged and confidential information on an hourly, daily basis. I am not able to share the information and it should not be visible to others.*"–P122

We could also see that some of the respondents are less concerned about the email notifications divulging information because they personally are less concerned about the contents of notifications being shared with the other people in the room. As P81 reported:

> "*...But I'm generally not embarrassed by who and what I am. Plus, humor is a good way to diffuse why I get the spam I get.*"

Also, worth noting that the floor and ceiling effects we observe may simply reflect limitations in our ability to sample the most/least sensitive contexts, contents, and emails for some users. Our second study will use more sophisticated sampling method to address such limitations.

To sum up, the distribution of responses indicate the three distinct groups of respondents. While some of the open-ended survey responses – like the ones above – highlight the relationship between respondents' occupation, personality and demographics [32] on notification preferences, some indicate that their preferences depend on the context. To understand this relationship, in the next sections, we extend the analysis by examining the content disclosed by notifications, and the people attending the meeting.

3.2.4 Notification fields and their content. Given prior research [28], we suspect that one's comfort level depends both on the types of information disclosed by notifications, and on the audience witnessing the disclosure. Addressing the former, participants were asked to select one email that they would have been uncomfortable sharing in their most recent meeting from among their 10 most recent emails. As noted earlier, we refer to this as a hypothetical information disclosure event (HIDE). Participants were asked to answer questions about their HIDE emails, and to describe, in their

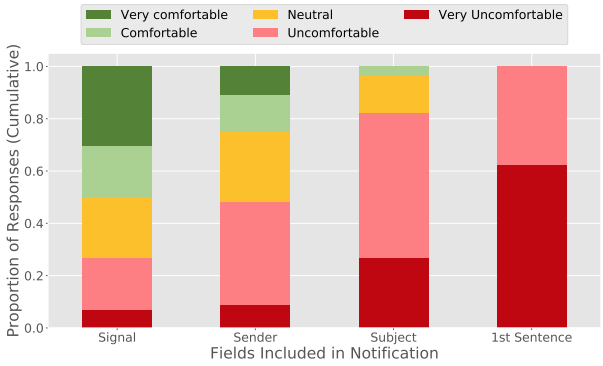


Figure 3: Distribution of respondent comfort levels for sharing different fields included in typical email notifications for HIDE emails.

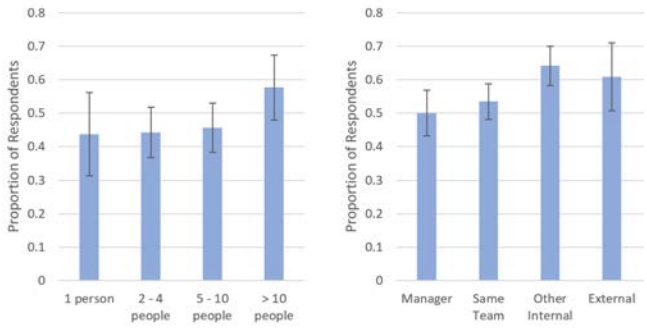


Figure 4: Proportion of respondents reporting HIDE emails, partitioned by the number of people attending the meeting (left) and the attendees' relationships to the respondent (right). Error bars show standard error; none of the pairwise differences are statistically significant at the $\alpha = 0.05$ level.

own words, the types of information that rendered a notification sensitive. We limit the remaining discussion to the 62 respondents (47%) who both identified a HIDE email, and who fully completed this portion of the survey questionnaire. We describe the general properties of HIDE emails, then analyze open-ended responses.

In most cases (72.6%), respondents elected to describe HIDE emails that were delivered to their work email inboxes – perhaps reflecting that the survey invitation was sent during business hours. Among these emails, most were sent by work colleagues (75.6%), including: direct superiors (15.6%), team members (28.9%), and other members of the organization (28.9%). Conversely, external senders included: clients or customers (8.9%), family and friends (4.4%), and one instance each from a doctor, and from an insurance company. Conversely, when HIDE emails were delivered to personal inboxes, most were from friends and family (76.5%), but also included messages from: doctors, banking institutions and external recruiters.

In the majority of cases (62.9%) respondents were the only recipient of their HIDE emails – though there were 11 cases (12.9%) where the email was sent to 5 or more people (via the to: or cc: lines). This finding agrees with the intuition that emails sent only to one person are, perhaps, more likely to contain private information. We explore this hypothesis further in the second part of this paper.

As noted above, we asked participants to describe, in broad terms, what types of information rendered these emails sensitive. Open-ended responses were analyzed, and themes identified, following open coding practices [22]. Among the top themes were: unspecified personal life details (14 instances), unspecified confidential work documents (11 instances), details of ongoing projects (10 instances), health information (7 instances), personal successes or failures (7 instances), financial communications (4 instances), and messages that mention people attending the meeting (3 instances). These themes largely overlap those identified by Olson et al. in [28].

As email notifications can reveal numerous email fields (Figure 1) including: sender, subject, and the first sentence of the email body, we asked participants to rate how comfortable would they have been if the people in the room saw the different email fields. Each field-type reveals a different class of information, and, to varying degrees, poses an information disclosure risk. For example, the sender field reveals that a respondent is in correspondence with a particular individual, while the subject reveals the topic of discussion. Correspondingly, among HIDE emails 25% of survey respondents were comfortable with email notifications that revealed the sender's identity, while only 3.6% were comfortable with notifications that reveal the email subject. This difference is highly statistically significant ($p = 0.001$, $Z = 3.20$). Figure 3 provides further details, breaking down respondent comfort levels by field-type.

3.2.5 Audience features. We explored whether meeting properties (e.g., location, attendance, etc.) impacted the proportion of respondents who identified a HIDE email among their 10 most recent messages. Though this analysis revealed no statistically significant results, we present our findings to: (1) characterize the meetings our respondents attend, (2) offer points of comparison with the results from our second study, and (3) identify weak trends that may yet become features in machine-learned models.

The majority of respondents reported that their most recent meeting occurred in a conference room (52.7%), though people also reported hosting meetings in their own offices (22.9%), attending meetings in someone else's office (15.3%), and attending meetings in other common spaces such as a lounge or atrium (8.4%). Among respondents whose meetings occurred in conference rooms, 50.7% were able to identify an email they would have been uncomfortable receiving in that context. Likewise, among respondents meeting in offices, 46.6% were able to identify such an email. These differences are not statistically significant ($p = 0.61$, $Z = 0.509$).

When asked how many people attended the respondent's most recent meeting, the mode response was "5 – 10 other people" (35.1%). Other response categories included: "2-4 other people" (32.8%), "10 or more people" (14.5%), and "1:1 meetings" (12.2%). Again, we examine the proportion of respondents who were able to identify an email they would have been uncomfortable receiving in each of these contexts. This proportion monotonically increases with the size of the meeting (Figure 4, left), but the pairwise differences are not significant.

Finally, we report that meetings were attended by team members (65.6%), direct superiors (41.2%), other members of the respondent's organization (51.1%), as well as people external to the organization (17.6%). In each case, we examine the proportion of respondents who were able to identify an email they would have been uncomfortable

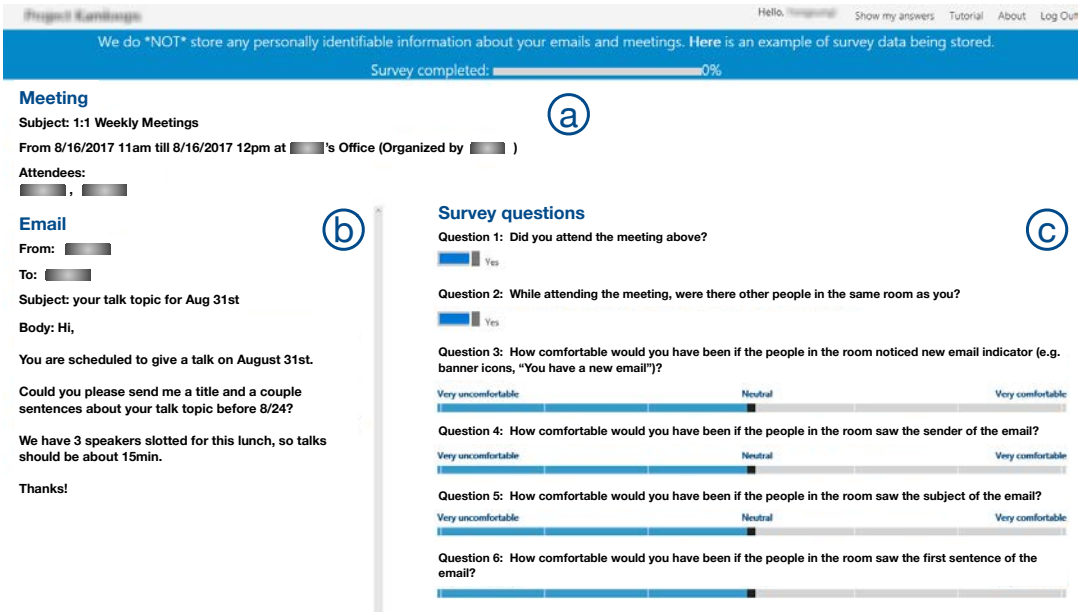


Figure 5: Interface for survey questions about email-meeting pairs. (a) Meeting pane that shows a randomly chosen meeting, time and location of the meeting, attendees, and subject of the meeting; (b) Email pane that shows a randomly chosen email that was received during the meeting, the sender, recipients, subject, and body of the email; (c) survey questions about their comfort level in sharing different fields of the email, type of the email, preferences for different devices and disclosure level.

receiving in the meeting. Figure 4 (right) shows that the proportion increases as the meeting's attendee list grows beyond one's own team. Though this agrees with intuition, the differences are not statistically significant. We revisit this observation later (§4).

In summary, results from the exploratory retrospective survey provide a key set of initial insights about the prevalence of the information disclosure risks posed by notifications, and about how email topics and fields may contribute to this risk. Moreover, we found that respondents could be clustered into three distinct groups based on the proportion of emails they received that would result in uncomfortable notifications.

While the survey was designed to ground responses in respondents' actual emails and meetings, it is limited in two important ways. First, to convey a simple and consistent sampling criteria and to minimize response or counting errors, the survey asked respondents to comment only on a single meeting, and on their single most-sensitive email. Second, it asked respondents to consider *hypothetical* situations in which they receive the most-sensitive email during the most recent meeting. Together, these survey design choices limit our ability to measure actual incidence rates, and to model the overall distribution of risk. To address these limitations, we developed and deployed a custom-built contextual labeling tool, which we describe in the next section.

4 STUDY #2: CONTEXTUAL LABELING STUDY

Based on the insights we gained from the exploratory retrospective study, we developed and deployed a tool (Figure 5) to extract various features and collect labelled data for learning a context-dependent predictive model of disclosure risk. The tool allows participants to: view emails that they received during meetings, view the details of those meetings, and rate their comfort-levels in receiving the corresponding email notification in that context. We described the procedure, apparatus, and results below.

4.1 Procedure and Apparatus

The contextual-labeling study was deployed within the same large IT company as the initial retrospective survey. Participants shared a common computing environment. In particular, they stored their emails and calendars in a common email and calendar web service. This homogeneous environment greatly simplified the administration of the study, and the implementation of the tool.

4.1.1 Procedure. Participants were recruited by emailing a random sample of 4000 employees, distinct from the 800 who were contacted for the retrospective survey. The invitation email described the study's purpose and procedure, and provided sufficient information to validate the authenticity of the invitation (e.g., links to internal systems and pages documenting the experiment, and the results of both an internal review process, and IRB review). Crucially, the invitation also included a link to the web application that hosted the labeling tool.

Upon navigating to the web application, participants were first shown the purpose of the study (i.e. characterizing people's preferences about information disclosed by desktop and mobile notifications that arrive when the intended recipient is in the presence of others). Then, participants were asked to authenticate to the tool using their corporate credentials, and to grant the tool time-limited access to their corporate email and calendaring accounts. Once

participants were authenticated, they were presented with a brief tutorial of the labeling tool, and its three regions:

- The top region (Figure 5a) displayed a recent meeting, randomly selected from a day. Visible fields included: meeting subject, time, location, organizer, and a list of attendees.
- The left region (Figure 5b) displayed a randomly selected email that arrived during the meeting. Visible fields included: the sender, recipient list (the 'to:' and 'cc:' lines), subject, and email body.
- The right region (Figure 5c) contained a short survey, where participants could answer questions about the email-meeting pair. Questions asked about participants' comfort levels in having notifications disclose various fields of the email to the people attending the meeting with a 7-point Likert Scale (1: Very uncomfortable, 7: Very comfortable).[2]

To collect data across a wide range of email-meeting pairs, the tool samples one email-meeting pair per day, moving backward in history one day at a time until 10 pairs are labeled. In each day, a meeting was randomly selected and then an email that was received during the meeting was randomly selected as well. If participants indicated that they did not attend a scheduled meeting, or that the meeting was conducted via teleconference, the tool selected another email-meeting pair for labeling.

Upon inputting labels for 10 email-meeting pairs, participants were presented with a debriefing page, and an optional invitation to take part in a raffle for one of three $50 Amazon.com gift cards. This sweepstakes was conducted in appreciation for their participation.

In addition to collecting user-provided labels and preferences, the labeling tool collected high-level features of the emails and meetings (Table 1). Importantly, the study was conducted completely anonymously: users were assigned random session ID, the participation sweepstakes was conducted on a separate unconnected system, and the features were chosen to be non-personally identifiable. We provide more details about these features in the next section.

4.2 Feature Extraction

Table 1 presents a list of features automatically collected by the labeling tool. Features are broadly categorized into three groups. User features are those that are associated with the participant, but not with a particular email-meeting pair. For example, this class includes the average number of emails a person received during meetings in the past 7 days, their depth in the organization chart, and their job title. To preserve anonymity, we used k-anonymization ($k = 50$) at the organizational level for full job titles, and separately, for job-title bigrams. Email features are those that describe a particular email message. Examples of these features include the number of recipients, Linguistic Inquiry and Word Count (LIWC, [35]) feature vectors, mentions of people, locations, and organizations, and whether the email is machine-generated. For LIWC categories, we tokenized the body, categorized each token into the pre-defined psychological and linguistic categories in LIWC, then computed the percentage of tokens in each category relative to the email body as a whole. We used the Stanford named entity recognizer [10] to detect mentions of people, places and organizations. To check whether or not an email is autogenerated, we used a simple heuristic to check whether or not an email contains 'Unsubscribe.' Meeting features describe the meeting instance, and include details such as the number of attendees, and whether a person's manager is in attendance. Organizational relationships were computed by cross referencing the attendee information from the calendar and the company's organizational chart as follows: for each attendee, we checked whether or not the attendee was the person's manager, direct report, above or below the org chart. Finally, we include one hybrid feature which counts the number of meeting attendees mentioned in the email.

[2]Since our exploratory survey results exhibited ceiling and floor effects, we extended the scale to 7 points to allow for more nuanced responses.

	Feature	Description
User	jobTitle	k-anonymized job title
	orgDepth	depth in the organizational chart
	numEmails	number of emails received in the past week
	numMeetings	number of meetings scheduled in the past week
	avgEmlPerMtg	average number of emails received in meetings
	numMtgWithEml	number of meetings interrupted by emails
Email	numRecipients	number of recipients in the email
	numDistList	number of distribution lists as the recipient
	numThreads	number of threads in the email
	isAutogenerated	is autogenerated email
	numPplMentioned	number of attendees mentioned in the email
	numAttachment	number of attachments
	attachment[type]	type of attachment in the email
	isSenderInternal	is the sender internal to the organization
	numSubjectWords	number of words in the email subject
	numBodyWords	number of words in the email body
	entity[type]	mentions of people, locations and organizations
	LIWC[cat]	feature vector over email body
Meeting	location	meeting location
	numAttendees	number of people attending the meeting
	isManagerPresent	is the person's manager present
	numDirectReports	number of direct reports present
	numOrgAbove	number of attendees above in the org chart
	numOrgBelow	number of attendees below in the org chart
	numExternal	number of attendees external to organization
	numSubjectWords	number of words in the meeting subject
	numBodyWords	number of words in the meeting body
	LIWC[cat]	feature vector over meeting body

Table 1: A list of user, email and meeting features that the labeling tool automatically computed for each email-meeting pair labeled by participants.

4.3 Results

We sought to replicate the analyses we conducted in the exploratory retrospective survey, when possible. As we will show, the consistency of their results helps bolster our confidence in their validity and reliability. Now, we discuss our main findings of this study.

4.3.1 Participants. In total, we received 1,040 meeting-email pairs labeled by 169 participants. Similar to the retrospective survey, job roles were diverse. The largest two categories included software developers (21.3%) and program managers (13.6%). An additional 52 individuals (30.8%) occupied various roles including: marketing managers, attorneys, sales specialists, business planners, etc. Finally, the job roles of 58 participants (34.3%) were filtered by k-anonymization.

	10 Uncomfort	Mixed	10 Comfort
Labeling Study	23%	36%	41%
Study1 (work)	17%	36%	47%
Study1 (personal)	40%	35%	25%

Table 2: Distribution of three groups of respondents in both the retrospective and contextual-labeling study. "10 Uncomfort" indicates respondents who labelled all 10 emails as uncomfortable, "10 Comfort" indicates all 10 emails as comfortable, and "Mixed" indicates a mixture of both comfortable and uncomfortable emails.

In addition to collecting job demographics, the labeling tool collected general measurements of the participants' calendars and email inboxes. For performance reasons, these coarse measurements are constrained to examining users' 300 most recent emails, together with the last 7 days of their calendar. 75.15% of participants' inboxes contained fewer than 300 emails received during this 7-day window (average: 103.7), allowing a full measurement of email-meeting co-occurrences during the week. These participants' 7-day calendars contained an average of 13.84 meetings (SD: 9.04). On average participants received emails during 7.35 (53%) of these meetings (SD: 6.10). In other words, slightly more than half of all meetings were interrupted by email.

A similar analysis can be performed for the remaining 24.85% who received more than 300 emails. Here, the results cover a variable time frame that is necessarily shorter than a week. For these participants, their most recent 300 inbox emails co-occurred with an average of 8.74 meetings (SD: 6.14). These participants' 7-days calendars contained an average of 18.83 meetings (SD: 11.66), suggesting that *at least* 46% of their meetings are interrupted by email.

The remaining analysis considers specific email-meeting pairs that are sampled from participant's calendars and inboxes. Unlike the above-mentioned aggregate measures, the sampling procedure is not constrained by the 7-day, 300-email, limit.

4.3.2 Prevalence of the information disclosure risk. As detailed in the procedure section, participants were asked to answer questions about 10 email-meeting pairs. To generate these pairs, we randomly sampled a meeting, then randomly sampled an email received during the meeting. Mirroring the analysis of the retrospective survey, 90 out of 169 people (53.3%) had at least one email whose notification they rated as uncomfortable sharing (i.e., they selected a comfort rating of <= 4 on the 7-point Likert scale for sharing the sender, subject and first sentence of the email). This proportion is nearly identical to that which was found in the retrospective survey for emails delivered to work inboxes (53.4%, Figure 2).

4.3.3 Three groups of respondents. Earlier, when we analyzed the results of the retrospective survey, we observed that respondents fell into three groups based on their responses to a hypothetical scenario in which they received notifications of their 10 most-recent emails while attending their most recent meeting. In this labeling study, participants were asked to label 10 emails known to have actually arrived during 10 distinct meetings. This allows a more ecologically valid analysis of this phenomenon. For comparison purposes we transform the 7-point Likert scale to a binary scale. Here scores > 4 (neutral) affirm that the user is comfortable with the notification. Results are presented in Table 2, together with the distribution reported in our retrospective survey. The results show that 23% of the respondents were uncomfortable sharing any of 10 emails in the respective meetings ("10 Uncomfort") , while 41% of the respondents were comfortable sharing all of the 10 emails ("10 Comfort") , and 36% of them were uncomfortable sharing some emails ("Mixed") . Notably, the distribution of users is roughly consistent across both the retrospective survey and the labeling tool (for work inboxes). This suggests that, although the retrospective survey involved a hypothetical situation, its findings closely match those of real-world scenarios. The distribution of the groups also resembles the classical trichotomy of the Westin Index; but, again, we are sensitive to the fact that these preferences can be highly nuanced and context-sensitive. To that end, we explore contextual factors below, as well as later in the section 5.

4.3.4 Email and meeting properties. When reporting the results of the retrospective survey, we examined various properties of emails and meetings that might indicate, or themselves constitute, an information disclosure risk. We now reexamine those criteria.

Number of email recipients: In the retrospective survey, we found that the number of people in an email's recipient list may influence users' comfort levels. Specifically, we observed that the majority of HIDE emails (62.9%) had only a single recipient. We can report a similar statistic for data collected via the labeling tool but must first filter the data such that they are directly comparable: In the retrospective survey, users were asked to discuss the most sensitive email from among those they would be uncomfortable sharing in a meeting. When we apply the same criteria to label-tool data, we find that 46.8% of emails have only a single recipient.

The labeled data also allows for a more deliberate examination of this phenomenon. This is because it contains examples of both sensitive and non-sensitive email-meeting pairs. Over all 1040 pairs, 406 (39%) emails were delivered to a single recipient (i.e., contains no other recipients in either the 'to:' or 'cc:' fields). For 131 (32.3%) of these emails, participants indicated that they would be uncomfortable with meeting attendees seeing the resultant email notifications. This number falls to 23.2% for emails delivered to multiple individuals. This difference is statistically significant ($Z = 3.228$, $p = 0.001$), suggesting that, when multiple people are in an email thread, the likelihood of the email containing sensitive information may be lower.

Number of external meeting attendees: Results from the earlier retrospective survey also suggested that meeting attendance might influence how comfortable people are with sharing their email notifications.[3] Specially, the presence of people outside of one's team or organization *might* increase levels of discomfort. Again, the labeled-data allows for a more ecologically valid and sensitive analysis: Of the 1040 email-meeting pairs, 298 (27.8%) were attended by people from outside of the organization.[4] Participants

[3]Results from the retrospective survey were not statistically significant, but suggested a possible trend worth further investigation (Figure 4).

[4]External to the organization via the *numExternal* feature. Note, the organization chart used in the labeling tool was not sufficiently fine-grained to determine if a fellow employee was a member of a different team.

Classifier	AUC	Pr	Re	F1
Boosted tree ensemble	0.85	0.71	0.63	0.67
kNN	0.59	0.39	0.23	0.29
SVM	0.58	0.78	0.03	0.06

Table 3: Results of the classifiers that predict if a respondent would be uncomfortable revealing a given email notification in a particular meeting context (i.e., an email-meeting pair).

Top 10	Top 11-20
U.avgEmlPerMtg (1)	M.numExternal (0.57)
U.numEmails (0.86)	E.numThreads (0.54)
U.numMtgWithEml (0.78)	E.LIWC[posemo] (0.53)
M.orgChartAbove (0.73)	E.numSubjectWords (0.49)
E.LIWC[money] (0.7)	U.orgDepth (0.47)
M.LIWC[negemo] (0.62)	M.LIWC[bio] (0.47)
M.numSubjectWords (0.6)	U.SENIORPROGMNG (0.45)
E.numBodyWords (0.58)	E.numCC (0.44)
M.LIWC[posemo] (0.58)	E.entity[org] (0.43)
U.numMeetings (0.58)	E.numDistList (0.41)

Table 4: Top 20 most informative features for the boosted tree ensemble classifier. The number in the parentheses indicates the information gain, normalized such that the most informative feature scores a 1.0. The prefixes 'U', 'E', and 'M', denote user, email, and meeting features, respectively.

reported that in 95 cases (32.3%), they would be uncomfortable sharing the email notification with meeting attendees. This proportion falls to 24.4% for meetings in which all attendees are fellow employees of the same organization. This difference is statistically significant ($Z = 2.78$, $p = 0.005$), suggesting that meeting attendance may indeed influence comfort levels about email notifications.

Together, these findings reinforce and extend our answers to the first two research questions: people are often interrupted by emails when in meetings, in a sizable minority of cases people are uncomfortable sharing the resultant email notifications with the people in the room (RQ1). These levels of discomfort may depend on individual characteristics, as well as on features of the meetings and emails (RQ2). This, in turn, hints at the possibility of using machine learning to predict when email notifications pose an information disclosure risk. We examine this in the next section, and, in doing so, answer our final research question (RQ3).

5 PREDICTING COMFORT LEVEL

To further explore the problem space, we developed binary classifiers that, given an email-meeting pair, decide if the delivery of an email notification would result in an uncomfortable situation. In constructing these classifiers, we both: (1) gain a deeper understanding of how combinations of user, email and meeting features may contribute to one's concerns about email notifications, and (2) explore modeling decisions and requirements that can lead to context-dependent predictions accurate enough to be used to manage users' notifications in real-world settings.

5.1 Prediction Results

For training and evaluation we use the 1040 labeled examples collected in the second study, above. Training and evaluation is done using 10-fold cross-validation, stratified the data such that, for each user, 7 labeled data points are included in the training set, and 3 labeled data points are included in the test split, and optimized to maximize the area under the receiver operating characteristic curve (AUC). We train boosted tree ensemble, SVM with linear kernel, and k-nearest neighbor (k=3) classifiers.

Detailed results are presented in Table 3. There are a few points to note. Our best performing classifier, the boosted tree ensemble, achieved an AUC of 0.85 which is well above chance. However, with our best performing model achieving a precision of 0.71, and a recall of 0.62, the classifiers are likely to be less useful in high-risk applications. Nevertheless, our explorations reveal that generic features provide some information about the information disclosure risks of an email-meeting pair.

For example, we looked at the 20 most informative features with our best performing classifier, the boosted tree ensemble (Table 4). In addition to the user features that might be indicative of potential exposure (the number of emails a user receives in a week, the average emails per meeting, the number of meetings), and email features that are related to email contents and length, we found features related to meeting context. Notably, the fourth most informative feature was one that counted how many meeting attendees were above the recipient in the organizational chart. Again, we also found that people were more uncomfortable sharing emails when the meeting description contains strong negative or positive emotion words categorized by LIWC [35].

To this end, we have answered our research questions. In the next section, we offer design implications, then conclude with a brief discussion of limitations and future work.

6 DESIGN IMPLICATIONS

In this section, we discuss some of the implications for designing contextually-relevant notification systems and policies.

Personalization matters. Both our exploratory retrospective survey and the contextual labeling study revealed three general and distinct groups of respondents. If a user can be quickly characterized as *Always Comfortable* or as *Always Uncomfortable*, then the notifications policies are rather straightforward: unconditionally allow all notifications for the unconcerned, and turn off notifications for those *Always Uncomfortable* unless the system is certain the user is alone. However, there exists a much richer strategy space for the *Mixed Comfortable* group, where various notification actions can be designed by hiding certain fields of messages, or delivering notifications only to certain devices.

Even simple context is helpful. As discussed above, for some people an effective notification policy might need only know if a person is alone, and examining a user's calendar may serve as an acceptable approximation. More sophisticated policies might consider whether an email was delivered to multiple people, or whether a meeting will be attended by people from outside the organization. These possibilities make clear that interesting notification policies can be developed from simple, easy-to-compute, signals.

Better sensing is likely to help. Both the retrospective survey and the contextual-labeling study included questions that could not be automatically answered based on calendar appointments and emails alone. For example, we asked participants if they attended meetings, and whether meetings were conducted via teleconference. With richer sensing capabilities, it is possible that these features could be reasoned about automatically.

Choose your notification fields wisely. Finally, our studies revealed that some notification fields are less sensitive than others. However, these preferences weren't universal, and varied even within a single user's responses. For example, an email's sender field may pose little risk in some cases (e.g., an email from a known collaborator), but pose significant risk in others (e.g., an email from a medical specialist). Systems should leverage these preferences, but also be expressive enough to allow exceptions. Systems should also scale their services, opting for more conservative notification strategies when there is uncertainty in predicting comfort level.

7 ETHICAL SAFEGUARDS FOR OUR STUDIES

In our studies, we sought to ensure participant anonymity and to avoid collecting any personally identifiable information. To make participation anonymous, we assigned participants random session identifiers in the labeling tool study, and hosted participation sweepstakes on a separate unconnected system. To preserve anonymity in feature extraction, for example, we used k-anonymization ($k = 50$) for full job titles and job-title bigrams. To avoid collecting personally identifiable information, we asked participants to describe their emails in broad terms in open-ended questions and cautioned participants not to include any personally identifiable information. Further in the labeling tool study, we chose email and meeting features that were not personally identifiable. For example, instead of collecting exact words that an email contained, we only collected LIWC feature vectors of the email. Finally, due to the sensitive nature of the topic we investigate, we also allowed participants to see the data being collected, and allowed them to delete their responses should they change their mind during or after the study.

8 DISCUSSION AND LIMITATIONS

In this paper, we characterized information disclosure risks that arise when people receive notifications in the presence of others. Specifically, we focused on emails that arrive during meetings. We employed multiple lines of evidence, including a large retrospective survey, and data collected in a second study via a purpose-built labeling tool. We report similar findings in both datasets, and are encouraged by this consistency.

Nevertheless, we caution readers against overgeneralizing our findings. Both the survey and labeling tool were deployed within a single large U.S.-based information technology company. Though respondents occupied a wide variety of job roles, it remains to be demonstrated that our findings generalize to other companies, company cultures, and industries. For example, it seems likely that people working in financial, legal, and medical industries may be more sensitive to information disclosure risks. It is also possible that preferences and concerns may vary by country and culture. Thankfully the *features* we described in this paper are very general, and are likely to be available in other companies and industry settings. As such, we expect replication efforts to be rather straightforward.

We would also like to extend our analysis to personal emails and contexts (e.g., social gatherings, etc). The retrospective survey revealed that people perceive a higher risk when personal emails arrive during work meetings. One wonders: Is the same true for social gatherings? How might work notifications be perceived in non-work contexts? We believe this line of investigation will be fruitful future research.

Likewise, our studies focused exclusively on email notifications. This choice was deliberate and practical; as noted in the introduction, email is often a carrier for other types of notifications (e.g., social networks). In the future, we hope to explore the privacy risks of other notification types, including: instant messages, calendar appointments, reminders, and other information proactively displayed by virtual assistants. Studying these notifications requires deeper technical integration with devices and platforms. Our studies were also exclusively performed from the perspective of the notification recipient. We recognize that the senders of emails are also subject to disclosure risks. When such risks are present, message originators can take some limited preventive actions to mitigate risks (e.g., by using the email subject to indicate that an email is sensitive, or by adding blank lines to the beginning of a message).

One goal of this paper is to assess the basic feasibility of machine learned classifiers, and in doing so, to identify features that help predict notification comfort levels. Our top classifier achieved an AUC of 0.85, and its strongest features capture aspects of notifications that were deemed important in our survey. Moving forward, we hope to improve accuracy, and this likely requires the collection of more data—our sample of 1040 labeled email-meeting pairs is rather small for state-of-the-art classifiers, and this data scarcity is confounded by the fact that the labels are likely correlated; the 1040 labeled pairs represent only 169 individuals. With more data, we hope to improve prediction accuracy enough to implement real-time notification policies for virtual assistants.

Finally, we recognize that there are costs to delaying notifications and hiding notification fields [17, 19]. Our studies did not measure these costs, and cannot directly experiment with formal notification policies. Nevertheless, we feel our analysis has showcased the need and opportunity to develop such policies, and has provided hints about which features and properties are likely to be informative.

9 CONCLUSION

In this paper, we report results from a retrospective survey and a larger contextual labeling study. Our research necessitated that we ask users to discuss sensitive scenarios. Specifically, we asked user to describe sensitive emails, and to discuss why they would feel uncomfortable receiving notifications for those messages when in the presence of others. To this end, we designed our studies to carefully respect participant privacy, and we believe these considerations were instrumental in allowing us to recruit a combined total of 300 individuals. From these individuals, we learned:

- (RQ1) Email notifications indeed pose an information disclosure risk.
- (RQ2) The real or perceived severity of these risks depend both on user characteristics (e.g. the nature of occupation)

and attributes of the meeting or email (e.g., the number of recipients or attendees).
- (RQ3) Machine-learned models can learn attributes, patterns and signals associated with risky email-meeting pairs. Here, user-level features are more informative than generic meeting or email-level features.

Taken together, our findings present a rich picture of notifications, as viewed through the lens of privacy. We hope that our findings will inform the design of future notification systems.

REFERENCES

[1] Mark S. Ackerman, Lorrie Faith Cranor, and Joseph Reagle. 1999. Privacy in e-Commerce: Examining User Scenarios and Privacy Preferences. In *Proceedings of the 1st ACM Conference on Electronic Commerce (EC '99)*. ACM, New York, NY, USA, 1–8. https://doi.org/10.1145/336992.336995

[2] Alessandro Acquisti and Ralph Gross. 2006. Imagined communities: Awareness, information sharing, and privacy on the Facebook. In *International workshop on privacy enhancing technologies*. Springer, 36–58.

[3] Bettina Berendt, Oliver Günther, and Sarah Spiekermann. 2005. Privacy in e-commerce: stated preferences vs. actual behavior. *Commun. ACM* 48, 4 (2005), 101–106.

[4] Camille Cobb and Tadayoshi Kohno. 2017. How Public Is My Private Life?: Privacy in Online Dating. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1231–1240. https://doi.org/10.1145/3038912.3052592

[5] Sunny Consolvo, Ian E. Smith, Tara Matthews, Anthony LaMarca, Jason Tabert, and Pauline Powledge. 2005. Location Disclosure to Social Relations: Why, when, & What People Want to Share. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '05)*. ACM, New York, NY, USA, 81–90. https://doi.org/10.1145/1054972.1054985

[6] Edward Cutrell, Mary Czerwinski, and Eric Horvitz. 2001. Notification, Disruption, and Memory: Effects of Messaging Interruptions on Memory and Performance. IOS Press, 263–269.

[7] Paul Dourish and Ken Anderson. 2006. Collective information practice: emploring privacy and security as social and cultural phenomena. *Human-computer interaction* 21, 3 (2006), 319–342.

[8] Janna Lynn Dupree, Richard Devries, Daniel M. Berry, and Edward Lank. 2016. Privacy Personas: Clustering Users via Attitudes and Behaviors Toward Security Practices. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5228–5239. https://doi.org/10.1145/2858036.2858214

[9] Susan Farrell. 2016. Computer-Assisted Embarrassment. https://www.nngroup.com/articles/embarrassment.

[10] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL '05)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 363–370. https://doi.org/10.3115/1219840.1219885

[11] Joel E. Fischer, Chris Greenhalgh, and Steve Benford. 2011. Investigating Episodes of Mobile Phone Activity As Indicators of Opportune Moments to Deliver Notifications. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '11)*. ACM, New York, NY, USA, 181–190. https://doi.org/10.1145/2037373.2037402

[12] Martin Fishbein and Icek Ajzen. 1975. *Belief, attitude, intention and behavior: An introduction to theory and research.*

[13] James Fogarty, Scott E. Hudson, Christopher G. Atkeson, Daniel Avrahami, Jodi Forlizzi, Sara Kiesler, Johnny C. Lee, and Jie Yang. 2005. Predicting Human Interruptibility with Sensors. *ACM Trans. Comput.-Hum. Interact.* 12, 1 (March 2005), 119–146. https://doi.org/10.1145/1057237.1057243

[14] The Radicati Group. 2015. Email Statistics Report, 2015-2019.

[15] Kirstie Hawkey and Kori M. Inkpen. 2006. Keeping Up Appearances: Understanding the Dimensions of Incidental Information Privacy. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. ACM, New York, NY, USA, 821–830. https://doi.org/10.1145/1124772.1124893

[16] Joyce Ho and Stephen S Intille. 2005. Using context-aware computing to reduce the perceived burden of interruptions from mobile devices. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 909–918.

[17] Eric Horvitz, Carl Kadie, Tim Paek, and David Hovel. 2003. Models of Attention in Computing and Communication: From Principles to Applications. *Commun. ACM* 46, 3 (March 2003), 52–59. https://doi.org/10.1145/636772.636798

[18] Eric Horvitz, Paul Koch, Carl M. Kadie, and Andy Jacobs. 2002. Coordinate: Probabilistic Forecasting of Presence and Availability. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI'02)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 224–233. http://dl.acm.org/citation.cfm?id=2073876.2073903

[19] Eric Horvitz, Paul Koch, Raman Sarin, Johnson Apacible, and Muru Subramani. 2005. Bayesphone: Precomputation of Context-Sensitive Policies for Inquiry and Action in Mobile Devices. In *User Modeling 2005*. Springer, Berlin, Heidelberg, 251–260. http://link.springer.com/chapter/10.1007/11527886_33 DOI: 10.1007/11527886_33.

[20] Shamsi T. Iqbal, Piotr D. Adamczyk, Xianjun Sam Zheng, and Brian P. Bailey. 2005. Towards an Index of Opportunity: Understanding Changes in Mental Workload During Task Execution. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '05)*. ACM, New York, NY, USA, 311–320. https://doi.org/10.1145/1054972.1055016

[21] Shamsi T Iqbal and Eric Horvitz. 2010. Notifications and awareness: a field study of alert usage and preferences. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*. ACM, 27–30.

[22] Harry Hochheiser Jonathan Lazar, Jinjuan Heidi Feng. 2010. *Research Methods In Human-Computer Interaction* (1 ed.). Wiley & Sons.

[23] Bart P. Knijnenburg, Alfred Kobsa, and Hongxia Jin. 2013. Dimensionality of information disclosure behavior. *International Journal of Human-Computer Studies* 71, 12 (Dec. 2013), 1144–1162. https://doi.org/10.1016/j.ijhcs.2013.06.003

[24] Airi Lampinen, Vilma Lehtinen, Asko Lehmuskallio, and Sakari Tamminen. 2011. We'Re in It Together: Interpersonal Management of Disclosure in Social Network Services. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 3217–3226. https://doi.org/10.1145/1978942.1979420

[25] Yuan Li. 2012. Theories in online information privacy research: A critical review and an integrated framework. *Decision Support Systems* 54, 1 (2012), 471–481.

[26] Patricia A. Norberg, Daniel R. Horne, and David A. Horne. 2007. The Privacy Paradox: Personal Information Disclosure Intentions versus Behaviors. *Journal of Consumer Affairs* 41, 1 (June 2007), 100–126. https://doi.org/10.1111/j.1745-6606.2006.00070.x

[27] Tadashi Okoshi, Julian Ramos, Hiroki Nozaki, Jin Nakazawa, Anind K. Dey, and Hideyuki Tokuda. 2015. Reducing Users' Perceived Mental Effort Due to Interruptive Notifications in Multi-device Mobile Environments. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 475–486. https://doi.org/10.1145/2750858.2807517

[28] Judith S. Olson, Jonathan Grudin, and Eric Horvitz. 2005. A Study of Preferences for Sharing and Privacy. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05)*. ACM, New York, NY, USA, 1985–1988. https://doi.org/10.1145/1056808.1057073

[29] Veljko Pejovic and Mirco Musolesi. 2014. InterruptMe: Designing Intelligent Prompting Mechanisms for Pervasive Applications. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14)*. ACM, New York, NY, USA, 897–908. https://doi.org/10.1145/2632048.2632062

[30] Martin Pielot, Karen Church, and Rodrigo de Oliveira. 2014. An In-situ Study of Mobile Phone Notifications. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services (MobileHCI '14)*. ACM, New York, NY, USA, 233–242.

[31] Alireza Sahami Shirazi, Niels Henze, Tilman Dingler, Martin Pielot, Dominik Weber, and Albrecht Schmidt. 2014. Large-scale Assessment of Mobile Notifications. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 3055–3064.

[32] Kim Bartel Sheehan. 2002. Toward a typology of Internet users and online privacy concerns. *The Information Society* 18, 1 (2002), 21–32.

[33] Sarah Spiekermann, Jens Grossklags, and Bettina Berendt. 2001. E-privacy in 2nd generation E-commerce: privacy preferences versus actual behavior. In *Proceedings of the 3rd ACM conference on Electronic Commerce*. ACM, 38–47.

[34] Milijana Surbatovich, Jassim Aljuraidan, Lujo Bauer, Anupam Das, and Limin Jia. 2017. Some Recipes Can Do More Than Spoil Your Appetite: Analyzing the Security and Privacy Risks of IFTTT Recipes. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1501–1510. https://doi.org/10.1145/3038912.3052709

[35] Yla R. Tausczik and James W. Pennebaker. 2010. The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. *Journal of Language and Social Psychology* 29, 1 (2010), 24–54. https://doi.org/10.1177/0261927X09351676 arXiv:https://doi.org/10.1177/0261927X09351676

[36] Sadegh Torabi and Konstantin Beznosov. 2016. Sharing Health Information on Facebook: Practices, Preferences, and Risk Perceptions of North American Users. In *Twelfth Symposium on Usable Privacy and Security (SOUPS)*.

[37] Jessica Vitak and Jinyoung Kim. 2014. "You Can'T Block People Offline": Examining How Facebook's Affordances Shape the Disclosure Process. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '14)*. ACM, New York, NY, USA, 461–474. https://doi.org/10.1145/2531602.2531672

[38] Dominik Weber, Sven Mayer, Alexandra Voit, Rodrigo Ventura Fierro, and Niels Henze. 2016. Design Guidelines for Notifications on Smart TVs. In *Proceedings of the ACM International Conference on Interactive Experiences for TV and Online Video (TVX '16)*. ACM, New York, NY, USA, 13–24. https://doi.org/10.1145/2932206.2932212

[39] Alan F Westin. 1991. Harris-Equifax consumer privacy survey 1991. *Atlanta, GA: Equifax Inc* (1991).

[40] Allison Woodruff, Vasyl Pihur, Sunny Consolvo, Lauren Schmidt, Laura Brandimarte, and Alessandro Acquisti. 2014. Would a privacy fundamentalist sell their DNA for $1000... if nothing bad happened as a result? The Westin categories, behavioral intentions, and consequences. In *Symposium on Usable Privacy and Security (SOUPS)*, Vol. 5. 1.

RiSER: Learning Better Representations for Richly Structured Emails

Furkan Kocayusufoglu*
University of California, Santa Barbara
furkan@cs.ucsb.edu

Ying Sheng
Google
yingsheng@google.com

Nguyen Vo
Google
nguyenvo@google.com

James B. Wendt
Google
jwendt@google.com

Qi Zhao
Google
zhaqi@google.com

Sandeep Tata
Google
tata@google.com

Marc Najork
Google
tata@google.com

ABSTRACT

Recent studies show that an overwhelming majority of emails are machine-generated and sent by businesses to consumers. Many large email services are interested in extracting structured data from such emails to enable intelligent assistants. This allows experiences like being able to answer questions such as "What is the address of my hotel in New York?" or "When does my flight leave?". A high-quality email classifier is a critical piece in such a system. In this paper, we argue that the rich formatting used in business-to-consumer emails contains valuable information that can be used to learn better representations. Most existing methods focus only on textual content and ignore the rich HTML structure of emails. We introduce *RiSER* (Richly Structured Email Representation) – an approach for incorporating both the structure and content of emails. RiSER projects the email into a vector representation by jointly encoding the HTML structure and the words in the email. We then use this representation to train a classifier. To our knowledge, this is the first description of a neural technique for combining formatting information along with the content to learn improved representations for richly formatted emails. Experimenting with a large corpus of emails received by users of Gmail, we show that RiSER outperforms strong attention-based LSTM baselines. We expect that these benefits will extend to other corpora with richly formatted documents. We also demonstrate with examples where leveraging HTML structure leads to better predictions.

CCS CONCEPTS

• **Information systems → Email**; • **Computing methodologies → Artificial intelligence**;

*Work done while interning at Google.

This paper is published under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 International (CC-BY-NC-ND 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY-NC-ND 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313720

KEYWORDS

Email Representation; HTML Structure Encoding; Email Classification

1 INTRODUCTION

Classifying documents into two or more target classes is a core problem in many web applications, and has received attention from several research communities. Identifying the sentiment in text documents [43], detecting spam in email [12] and the web [36], and fake news detection [42] are well-known applications. Even enterprise applications like legal discovery [40] make use of the core abstraction of document classification.

Most existing approaches focus on short plain-text documents like tweets, reviews, and posts in online discussion forums. However, several interesting applications are on corpora where the documents are longer, and often have rich HTML markup structure. Such corpora have not received as much research attention, but are important for many applications. For instance, web pages have rich HTML structure; classifying them [39] is useful for several applications like focused crawling, faceted search, and even as a signal for web page search ranking.

Email is another area that contains longer documents with rich HTML markup structure. Recent studies [31] have shown that most email is richly formatted and machine-generated rather than plain-text sent by humans. Several interesting applications beyond spam-detection rely on learning good classifiers over email. This includes foldering [19], automatic prioritization [15], and even information extraction [2, 14, 41]. In particular, these studies show that learning a high-precision classifier over richly formatted emails is a key ingredient to extracting structured data from email that is then used to power several intelligent experiences [14, 41].

In this paper, we tackle the problem of learning a good representation (embedding) for structured documents like richly formatted emails. We focus on evaluating these embeddings for the task of classifying an email into one of k target classes. Recent literature has shown that for relatively short documents, a recurrent neural network with an attention mechanism [48] is a very strong baseline for learning good representations. With richly formatted documents, most existing techniques ignore the formatting information except

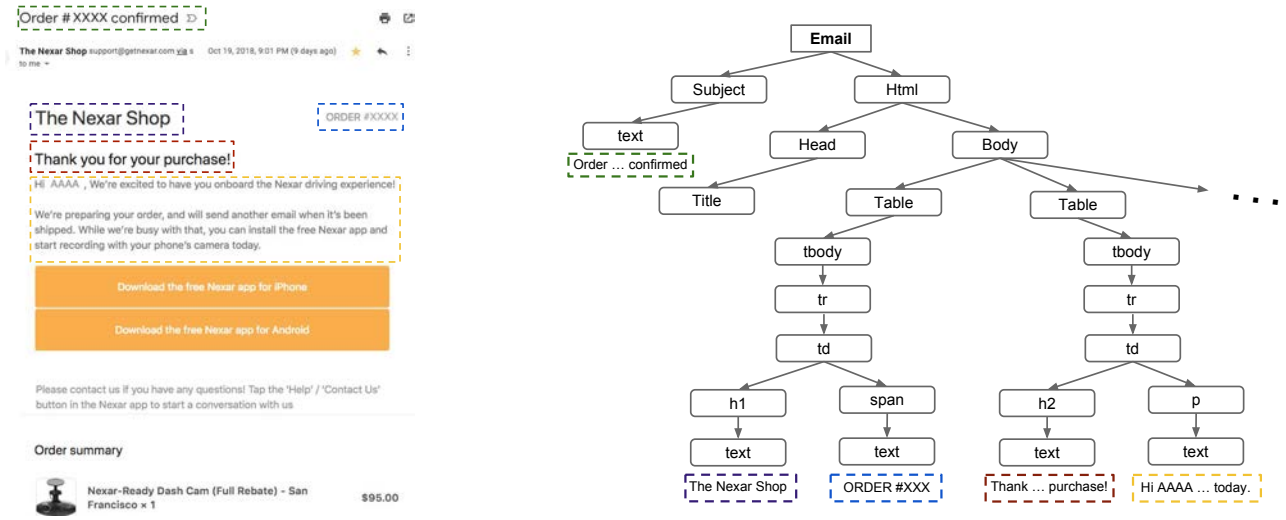


Figure 1: An example *purchase* email (left) and a simplified DOM tree (right) representing a portion of the email. For clarity, we trim long trivial HTML sequences such as nested *div* tags. Note that we extend the DOM tree beyond the email message body to include a branch representing the subject.

for assuming that documents are hierarchically organized [48] into sentences and paragraphs. We illustrate in Section 2 that the rich structure of HTML-formatted email can be a valuable signal for the relative importance of various pieces of information in the document. We introduce a neural architecture called RiSER (Richly Structured Email Representation) to take advantage of the structure of the email in addition to the content. RiSER projects the input email into a vector representation by jointly encoding the HTML structure of the email along with the terms in the email. It consists of a 2-level LSTM to first construct a structural encoding and combines this with an encoding of the terms for which there is a second LSTM layer. This provides a simple way to encode the formatting and layout information as opposed to manually engineering structural and visual features.

Experiments on real data from Gmail corpus show that the structure does indeed contain useful information and that RiSER is able to exploit this information to outperform a strong baseline that only considers textual information. While we focus on emails, the techniques introduced in RiSER are generic and can be used for any document corpus containing richly formatted information such as HTML web pages and PDF documents.

We make the following key contributions in this paper:

- We identify that structural information in richly formatted HTML emails is a valuable signal that is ignored by existing classification approaches focused on textual content.
- We propose a 2-level neural architecture called RiSER that jointly learns to encode both the structure and the content of the email by examining the sequence of HTML tags that each text term is associated with.
- We demonstrate through experiments on two classification tasks on data from Gmail that RiSER learns enhanced email representations.

The rest of this paper is organized as follows: Section 2 provides more detailed background on the information available in the structure and markup, explains how we represent the input document, and lays out the problem definition for applications like information extraction. Section 3 introduces our two-level neural architecture that jointly encodes the HTML markup and the textual content of documents to learn a good representation. Section 4 reports on the performance of this architecture on two real-world classification tasks using data from Gmail. We compare the performance of this architecture with several baselines including ones that try to leverage manual feature-engineering to represent aspects of this information. The experimental results show that the models with the proposed architecture significantly increase recall at high precision compared to previous baselines by up to 8.6%. Section 5 summarizes areas of related work from the data mining, NLP, and ML research communities. Finally, Section 6 concludes the paper with a summary and potential directions for additional research.

2 EMAIL STRUCTURE

We focus on business-to-consumer (B2C) emails, the vast majority of which are machine-generated instantiations of predefined templates, accounting for up to 90% of all email traffic on the internet [30].

Figure 1 (*left*) depicts an example email from the author's inbox that contains a purchase confirmation. Unlike personal emails sent by humans, B2C emails like these often have rich HTML structure. This structure introduces a complex hierarchy which can be represented by a Document Object Model (DOM) tree [27], such as the one illustrated in Figure 1 (*right*). Each node in this tree represents an HTML tag in the message body, and its children consist of the HTML tags nested within it. The text of the message resides at the leaf nodes. Note that we have extended this DOM tree beyond the

standard model to also include the subject of the email as a child node of the email root. A typical commercial email like the example in Figure 1 may contain around 2,000 leaf nodes. In fact, we have observed some messages, albeit few, that have as many as 10,000 leaf nodes and a depth of up to 200.

The use of rich HTML structure draws the attention of the human eye to different parts of the email. We find that the categorically similar emails often share similar structure. For example, bill reminder emails will format a "pay now" link close to the middle of the email, while hotel confirmation emails often contain tables of check-in and check-out information. This manifests as tables, headers, footers, highlighted and emboldened text in the markup. The key hypothesis we test in this paper is whether the markup in such richly formatted emails offers additional signal over just the textual content.

The remainder of this section describes how we convert this tree representation into a sequence of features that are suitable as input for the classification models described in the following section. The sequence consists of three types of features: *textual features*, *annotation features*, and *structural features*.

The textual features consist of a sequence of the first 200 terms (including punctuation) extracted from the DOM tree (including the subject branch) via in-order traversal. Limiting ourselves to 200 terms allows us to fully represent most emails without the risk of allowing very long emails to derail model training.

The annotation features indicate whether a particular text span contains an annotation. Several examples such as *Date*, *Price*, and *Time* are listed in Table 1. A term may correspond to one or more annotations. These annotations are extracted through a variety of methods including dictionary lookups and regular expression matching. Libraries of these annotators were developed over several years to support more traditional rule-based information extraction tasks. An entity detection library, for example, is also available as a Google Cloud API [18]. Readers can refer to Sheng et al. [41] for more detailed discussion on these annotators. In this work, we provide a way to incorporate the signals from these annotations.

Annotation Types	
Address	Location
Alphabet-Number	Price
Confirmation Number	Telephone Number
Date	Time
Establishment	Tracking Number

Table 1: Annotation features.

Finally, we represent the structure of the document using XPaths [7]. An XPath is the sequence of nodes (HTML tags) extending from the root of the DOM tree to a leaf node. For example, the XPath of the term "ORDER" in the example email in Figure 1 is */html/body/table/tbody/tr/td/span/text*. All the terms in a paragraph (in a *p* node), for instance, might share the same sequence of HTML tags for their XPath feature.

We considered several alternative approaches to represent document structure. For example, one can identify the XPath patterns for text in key HTML tags like headers, tables, lists and then use these to mark each term. This approach is labor-intensive and more importantly ad-hoc and may not take into account patterns than an engineer has not visually inspected and considered important. For our problem, we are interested in an approach that can be applied to all the emails from various domains with different structural patterns without having to manually engineer structural features. Another possible representation is to use a visual blocks structure using external parsers [11]. Visual blocks are tree structures based on the visual layout of the page. Such an approach would depend on an external parser to produce features that can then be represented as part of the input example. We chose to use the XPath since it is simple, and provides a detailed representation of the markup. This allows us to directly test the hypothesis of whether the information in the markup is valuable for learning a better representation. We expect that using visual features from a sub-system that understands document layout could be interesting future work.

A complete set of features representing a single email document thus consists of a stream of up to 200 terms, the set of annotations that each term is a part of, and the XPath that each term resides at. Figure 2 denotes the features corresponding to three terms – "Shop", "ORDER" and "#XXXX" from the email in Figure 1. The terms "Shop" and "ORDER" do not contain any annotation, but "#XXXX" has two annotations: *Confirmation Number* and *Alphabet-Number*. The bottom row contains the XPath in the DOM tree at which each of these terms appear.

We describe the architecture that combines these three features in Section 3 below, and explore the utility of different combinations of these features in Section 4.

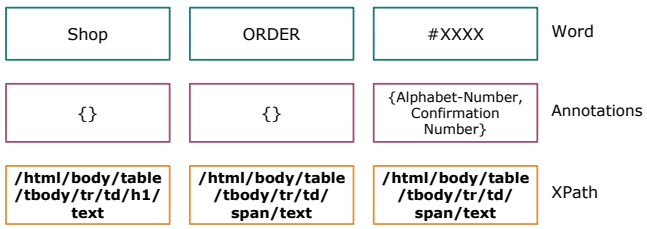


Figure 2: Feature sets of three terms "Shop", "ORDER" and "#XXXX" from the top of the email in Figure 1.

3 MODEL ARCHITECTURE

In this section, we present our email representation framework – RiSER. The proposed model consists of two main components: an XPath encoder and a word encoder. At a high level, the XPath encoder models an email's DOM structure by encoding the tags along the XPath to a leaf node using an LSTM layer [21] while the word encoder combines word embeddings with the corresponding XPath encodings and additional features to learn an enhanced representation of the email. The enhanced email representation is then used as a feature for email classification. Figure 3 shows an overview of the model architecture. We describe the details of the framework and its components in the following sections.

3.1 XPath Encoder

Recall from Section 2 that an XPath is a sequence of HTML tags from root to leaf of an email's DOM tree. The proposed XPath encoder

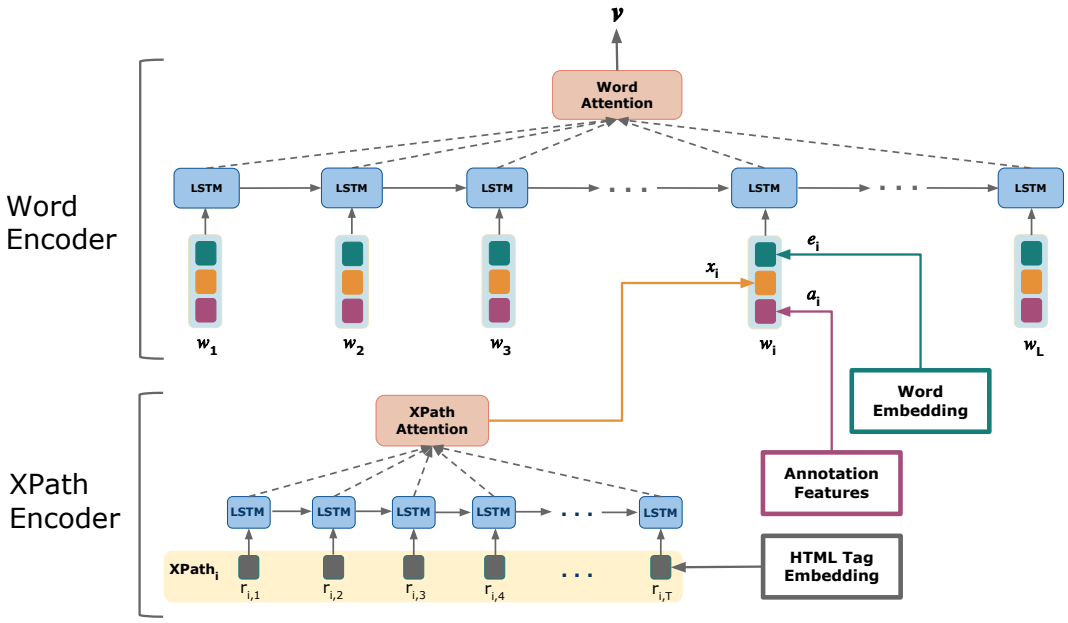


Figure 3: Overview of the RiSER architecture. The XPath Encoder iteratively encodes the tags in the XPath for each term in the email's DOM tree. Later, each XPath encoding (x_i) is combined with the corresponding word embedding (e_i) and annotation features (a_i) of the term to form a word representation (w_i). The Word Encoder then processes these word representations ($\{w_1, w_2, \cdots, w_L\}$) to learn an enhanced email representation (v). *(best viewed in color)*

in Figure 3 summarizes this sequence with a vector of fixed length. In this work, we use an LSTM encoder with an additional attention mechanism [48] to represent the sequential elements (HTML tags), however, one is free to choose any standard sequence encoder to model the sequential input. More precisely, let the $XPath_i$ be a sequence of HTML tags with embeddings $[r_{i1}, \cdots, r_{iT}]$. In practice, we randomly initialize these embeddings and update them during training. We pass these embeddings through an LSTM layer to produce output vectors:

$$h_{it} = \overrightarrow{LSTM}(r_{it}), t \in [1, T] \tag{1}$$

Typical practices for extracting a final output representation from an LSTM layer include selecting the final output or averaging across all outputs. However, we observe that XPath sequences can be very long and highly repetitive. Furthermore, not all HTML tags may contribute equally to the email structure. For instance, HTML tags such as `strong`, `em` or `big` can be more informative than other much more common tags, such as `div`. Hence, we apply an attention mechanism [48] over the outputs of the LSTM layer in order to (1) extract such informative tags from long sequences and reward their contributions to the XPath representation, and (2) simultaneously reduce the impact of highly repetitive tags.

The final XPath encoding vector is computed as follows. First, we feed the LSTM output h_{it} through a fully connected layer with trainable weight matrix W_r and bias vector b_r, as well as a *tanh* activation function to produce the hidden vector u_{it}:

$$u_{it} = \tanh(W_r h_{it} + b_r) \tag{2}$$

Next we calculate an importance weight α_{it} for each HTML tag r_{it} in the sequence as a normalized similarity score between the hidden vector u_{it} and a trainable structure vector u_r:

$$\alpha_{it} = \frac{\exp(u_{it}^T u_r)}{\sum_i \exp(u_{it}^T u_r)} \tag{3}$$

Finally, we compute the *XPath encoding* vector x_i as a weighted sum of all the LSTM outputs:

$$x_i = \sum_t \alpha_{it} h_{it} \tag{4}$$

The proposed XPath encoder has the following two desirable properties. First, as we show in Section 4.5, the XPath encoding vectors capture information that is highly valuable for an email classifier. Second, the XPath encoder captures the relative hierarchy of an email's structure, i.e. two XPath sequences will have similar embeddings in the vector space if they correspond to closer leaf nodes in the DOM tree since they share long common subsequences. In the next section, we explain how the XPath encoder interacts with the word encoder.

3.2 Word Encoder

The word encoder is responsible for learning a rich representation of the email which not only summarizes the email's content but also encodes its HTML structure. In order to accomplish this goal, we go

beyond the word-embeddings-based representations and capture additional features with structural information from the email. We create our rich word representation vector w_i by leveraging the following three components:

- *Word embeddings.* We construct our vocabulary with the most frequent 20,000 words in our data. Each word in the vocabulary is mapped to a word embedding vector through an embedding look-up matrix. Out-of-vocabulary (OOV) words are mapped to an *unknown* vector. In this work, we randomly initialize the word embeddings and jointly train them with the model. One can also use pre-trained word embeddings such as Word2Vec [35], Glove [38], or fastText [9].
- *XPath encodings.* The corresponding XPath of each word is passed through the XPath encoder to produce an XPath encoding. These XPath encodings implicitly capture the position of the word in the DOM tree. Since XPath sequences can be very long, we only keep the first T tags of the sequence and omit the rest. Note that T is a hyper-parameter in our framework.
- *Annotation vectors.* We represent the annotations a word corresponds to as a 10-D binary vector. Each dimension (0/1) indicates whether a word is annotated by a particular markup shown in Table 1. These features turn out to be very helpful, as we will show in Section 4.4.

To form improved word representations, the word embeddings, XPath encodings, and annotation vectors are concatenated for each term and fed into a fully connected layer with *tanh* non-linearity. Formally, given an email with L words, we represent each word in the sequence as:

$$w_i = \tanh(W_s[e_i, x_i, a_i]), i \in [1, L] \tag{5}$$

where $e_i \in \mathbb{R}^k$ is the word embedding vector, $x_i \in \mathbb{R}^l$ is the corresponding XPath encoding calculated in Equation (4), and a_i is the 10-dimensional binary vector representing the annotations mentioned in Section 2. $W_s \in \mathbb{R}^{m*m}$ is a trainable weight matrix, where $m = k + l + 10$.

With $[w_1, w_2, \cdots, w_L]$ we can now compute an email representation vector by using another LSTM layer to encode the sequence and apply an additional attention mechanism [48] over that layer:

$$h_i = \overrightarrow{LSTM}(w_i), i \in [1, L] \tag{6}$$

$$u_i = \tanh(W_w h_i + b_w) \tag{7}$$

$$\alpha_i = \frac{\exp(u_i^T u_w)}{\sum_i \exp(u_i^T u_w)} \tag{8}$$

$$v = \sum_i \alpha_i h_i \tag{9}$$

where v is the email representation vector that summarizes both the email structure and semantics.

3.3 Email Classification

We use the email representation computed above as a feature in email classification. More specifically, we use a linear layer to convert the email representation vector to a real-valued vector of size $|C|$, where C is the set of classes to predict. A final softmax layer is applied to obtain the normalized probabilities over the labels:

$$p = \text{softmax}(W_c v + b_c) \tag{10}$$

The training objective of our framework is cross-entropy loss:

$$loss = -\sum_{d \in D} \sum_{c \in C} p_c^g(d) \cdot log(p_c(d)) \tag{11}$$

where D is the set of training emails, C is the collection of email classes, $p_c(d)$ is the probability of predicting email d as class c, and $p_c^g(d)$ is a binary value indicating whether c is the correct label. The entire model is trained through back-propagation with respect to all parameters. Training details are further explained in Section 4.3.

4 EXPERIMENTS

This section presents the results from several experiments on data from a large Gmail corpus to illustrate the effectiveness of the RiSER architecture. We focus on two binary classification tasks for this purpose. The first task is one of detecting if an email contains a bill. This model is used by the email service to determine if it wants to proactively remind the user when a bill is due. The second task is one of detecting if an email contains a hotel reservation confirmation. This model is also used to support intelligent applications such as travel planning. We refer to these two as the *Bill* and *Hotel* tasks. We pick these two tasks for illustrative purposes. The overall system that these classifiers are a part of includes several other classifiers (and extractors) for flights, calendar appointments, shipping confirmations, etc.

The experiments are designed to compare the performance of different RiSER variants (introduced in Section 4.2) against a strong baseline and to understand the incremental advantage of incorporating the annotations and XPath embeddings. Before describing the experimental setting, we first describe how the datasets are constructed including the source for the labels and sampling techniques used to deal with biases.

4.1 Dataset

The training and testing datasets are constructed by sampling from the Gmail corpus. During the course of this work, user privacy was protected through strict data access controls and data preprocessing to avoid training on sensitive data. Nobody involved with this project had access to visually inspect any of the data. Only terms matching the dictionary of top 20,000 terms are used, while the rest are replaced by an *unknown* identifier. Any text spans that contain potentially private data (such as addresses, dates, and phone numbers) are denoted by the corresponding annotation feature, and the underlying terms are replaced by the *unknown* identifier.

4.1.1 Labels. Ground truth labels for this dataset are derived from three sources: Microdata [20], manually defined parsers, and rule-based extractors. Microdata is a standard that enables senders to explicitly label and mark up their outgoing emails with structured information. For example, when sending a confirmation email to a customer, hotels can include markup to indicate that their emails are reservation confirmations, and even specify details such as confirmation numbers, addresses, check-in and check-out out dates and times, etc. While Microdata markup is precise, it is not widely

adopted by all email senders, so the volume of annotations tends to be low.

Manually defined parsers are designed to extract category-specific fields from emails in lieu of Microdata. These are created on a per-sender basis and hand-crafted based on several instances of emails donated by users for this very purpose. For the purposes of email classification, we use the presence of a successful extraction from an email as a positive label for the category.

Note that we prioritize construction of parsers for high volume senders, since (a) parser construction is laborious, thus focusing on larger senders yields more labeled samples, and (b) the donated corpus is relatively small and thus contains a limited number of samples from low volume senders. Manually defined parsers tend to be highly precise but brittle, since small changes to the emails can often break the parsers, requiring human involvement and new donated emails to fix them.

Rule-based extractors consist of a manually engineered set of traditional information extraction techniques, such as dictionary lookups and regular expressions, that operate across senders. These are also constructed by leveraging the donated email corpus for both development and validation. These tend to have higher recall than the approaches above, but often at the cost of lower precision. Similar to the manually defined parsers, we use the presence of a successful extraction of an email as a positive label for the email.

4.1.2 Sampling. Despite drawing labeled examples from the three sources of ground truth mentioned above, the vast majority of emails remain unlabeled, and even among those that are labeled, the distribution is skewed towards higher volume senders. Thus, training classifiers using a uniformly random sample of data often leads to overfitting and poor generalization to smaller senders.

Domain-based sampling attenuates this issue by limiting the number of samples observed from high volume senders and boosting those from smaller senders, however this method can be too coarse for larger senders that have their own internal skew—e.g. the majority of amazon.com email might be classified as *purchase orders*, while a small fraction are *account memberships* or *bill reminders*.

We find a balance between these two methods by stratifying emails by the templates from which they are instantiated. This ensures that an even number of emails are represented from the *purchase order*, *account membership*, and *bill reminder* templates, along with emails from much smaller senders.

These templates are inferred through a one-time template generation process that clusters similarly structured emails into groups that are likely to have been instantiated from a single template. These techniques range from clustering on the sender and subject of an email [3] to clustering on the structure of the email body [2, 4]. In this work, we use structural clustering techniques based on a locality sensitive hash of the email body similar to the technique described in [41].

We split the data into train and test sets with a ratio of 9:1 while ensuring that samples belonging to the same sender domain appear in only one of these sets to prevent the effects of memorization. For the training data, we downsample the negative examples to yield a positive-to-negative ratio of about 1:100. We keep the original ratio in the test set. To put this into perspective, the resulting training sets have approximately *150M* total samples for the Bill task and *31M* samples for the Hotel task.

4.2 Experiment Configuration

We train and evaluate four variants of the RiSER architecture, each incorporating a different combination of the textual, annotation, and structural features described in Section 3.2.

- **RiSER-W** uses the Word Encoder only and represents words using the *word embeddings* component only. That is, in Equation 5, the word representation w_i is represented by the word embedding e_i only.
- **RiSER-WA** uses the Word Encoder and represents words with the *word embeddings* and *annotation features*.
- **RiSER-WX** uses both the Word Encoder and the Xpath Encoder and represents words with the *word embeddings* and *XPath encodings*.
- **RiSER-WXA** uses both the Word Encoder and the XPath Encoder and represents words with the *word embeddings*, *XPath encodings*, and *annotation features*. This variant is the one displayed in Figure 3.

The RiSER-W variant is simply an LSTM-based recurrent model with an attention mechanism, as it only includes word embeddings in its input layer. Recent studies in the literature have shown that attention-based LSTM models [29, 48] are excellent for modeling text documents and achieve state-of-the-art performance in document classification tasks by outperforming linear models, SVMs, and feed-forward neural networks. Therefore we consider RiSER-W as a baseline model to compare against the remaining RiSER variants.

Note that one can employ more complex neural architectures than just the RiSER-W architecture to improve email classification performance. However the focus of this work is to demonstrate the advantages of exploiting the rich formatting structure of emails through our proposed XPath Encoder and overall architecture. Moreover, we aim to demonstrate that textual content can be better utilized when combined with structural information for learning email representations.

We compare the performance of the RiSER-WA, RiSER-WX and RiSER-WXA variants against the baseline RiSER-W model to determine whether structural information and annotation features can improve the learned representations. We test this hypothesis by classifying emails into two semantic categories, *Bills* and *Hotels*. While we could have posed this as a multi-class classification problem, practical convenience of improving one model without affecting any other models motivated us to pose these as separate *binary classification* tasks. The general idea of leveraging structure to learn a better representation holds for multi-class, multi-label, or even regression tasks.

4.3 Hyper-parameters and Training

For all four variants, we perform grid search over the word embedding dimension ({50, 100, 200}), the word encoder LSTM output (hidden state) dimension ({64, 128, 256}), the initial learning rate ({0.01, 0.001, 0.0001}), the batch size ({50, 100, 200}), the word encoder dropout rate ({0, 0.25, 0.5}), and the optimizer type (AdaGrad [16] and Adam [24]). For the Adam optimizer, we use the default settings suggested by the authors ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$).

For the RiSER-WX and RiSER-WXA models we also search over the following additional hyper-parameters: the HTML tag embedding dimension ({25, 50, 100}), the XPath encoder LSTM output dimension ({16, 32, 64}), the XPath encoder dropout rate ({0, 0.25, 0.5}), and the maximum XPath length ({20, 30, 40}).

We use the norm clipping trick with a *threshold* of 5.0 in all four models to avoid the exploding gradient problem.

For each model, we use the Vizier [17] platform to select the best combination of hyper-parameters based on AUC-PR after a maximum of 1M training steps. The selected hyper-parameters for each model are shown in Table 2. We later continue training the selected models until they converge and report their highest numbers on the test data in the next section.

BILL				
Hyper-parameter	RiSER-W	RiSER-WA	RiSER-WX	RiSER-WXA
batch size	200	200	200	200
learning rate	0.001	0.0001	0.0001	0.001
optimizer type	Adam	Adam	Adam	Adam
word embedding dim	200	100	100	100
WE LSTM output dim	128	128	128	128
WE dropout rate	0	0.25	0.25	0
tag embedding dim	N/A	N/A	25	25
max XPath length	N/A	N/A	20	20
XE LSTM output dim	N/A	N/A	64	16
XE dropout rate	N/A	N/A	0.5	0.25

HOTEL				
Hyper-parameter	RiSER-W	RiSER-WA	RiSER-WX	RiSER-WXA
batch size	200	100	200	200
learning rate	0.001	0.0001	0.0001	0.001
optimizer type	Adam	Adam	Adam	Adam
word embedding dim	200	100	100	200
WE LSTM output dim	128	64	64	128
WE dropout rate	0.5	0.25	0.25	0.25
tag embedding dim	N/A	N/A	50	25
max XPath length	N/A	N/A	20	30
XE LSTM output dim	N/A	N/A	16	64
XE dropout rate	N/A	N/A	0.5	0.25

Table 2: Selected hyper-parameters for the RiSER variants for the Bill and Hotel classification tasks. WE stands for Word Encoder and XE stands for XPath Encoder.

4.4 Results and Discussion

Classifier performance is often measured using area under the receiver operating characteristic curve (AUC-ROC). However, when dealing in datasets with high class skew—which is often the case for email—the area under the precision-recall curve (AUC-PR) provides a much fairer representation of model performance. In practice, even the AUC-PR metric is inadequate given that applications for email generally require extremely high precision. For example, misclassifying an important email as spam is an incredibly bad user experience.

Thus, we evaluate our models based on their *recall at a fixed level of high precision*. By requiring that all models reach a prespecified precision threshold, we minimize bad user experiences. Fixing the precision then allows us to evaluate different models by the extent of their coverage at that level of precision. That being said, we also report the AUC-PR metric for clarity. The experimental results on both datasets are summarized in Table 3.

BILL				
Model	R@P=0.85	R@P=0.90	R@P=0.95	AUC-PR
RiSER-W	76.4	71.5	57.4	85.4
RiSER-WA	76.8	73.0	63.1	85.9
RiSER-WX	76.6	73.5	64.1	86.2
RiSER-WXA	78.0	73.7	66.0	85.6

HOTEL				
Model	R@P=0.85	R@P=0.90	R@P=0.95	AUC-PR
RiSER-W	95.9	95.0	92.4	97.1
RiSER-WA	97.3	95.7	93.2	98.0
RiSER-WX	96.3	95.5	93.5	97.2
RiSER-WXA	97.4	96.3	92.9	97.7

Table 3: Recall at a fixed level of precision and AUC-PR performance metrics for four RiSER variants trained for the Bill and Hotel classification tasks.

4.4.1 **Recall-at-fixed-precision (R@P).** Results show that our RiSER-WXA variant overall gives the best performance across both datasets. The improvement in performance depends on the data type. For more complex dataset, such as Bill, the RiSER-WXA variant outperforms the baseline model (RiSER-W) by 2.2% on R@P(0.9), and by 8.6% on R@P(0.95) metrics. Note that recall tends to decrease drastically for the baseline model as the precision threshold increases, while RiSER-WXA demonstrates more robust performance. This leads to a larger gap between both models' recall at higher precision thresholds. On the other hand, improvements in the Hotel task are smaller compared to the Bill task, leading to 1.3% on R@P(0.9), and by 0.5% on R@P(0.95) metrics. We also report the experimental results of RiSER-WA and RiSER-WX variants in order to better evaluate the effects of each component.

Effect of annotation features. Our experiments show that using crafted annotation features on top of word embeddings provides important performance gains over the baseline. With respect to the R@P(0.95) metric, the RiSER-WA variant outperforms the RiSER-W variant by 5.7% on the Bill task, and 0.8% on the Hotel task. Similarly, the gap between recall is smaller (1.5% and 0.7%) as we decrease the precision threshold.

We believe the reason behind the performance boost that comes with the annotation features is that they allow us to recover meanings in words that cannot be captured with the defined vocabulary. To be more specific, emails include dates, numbers, addresses such that most of these terms are anonymized due to privacy constraints and are embedded with the *unknown* vector. Thus, including the annotation features allows our framework to capture these additional signals, which intuitively improves the model's ability to correctly classify an email. As a toy example, an email without a date is most likely not a hotel reservation.

Effect of email structure. Combining XPath encodings with word embeddings leads to an improvement of 6.7% in the Bill task, and 1.1% in the Hotel task with respect to the R@P(0.95) metric. More importantly, by comparing the performance of RiSER-W and RiSER-WX, we verify that incorporating the *HTML structure* into the learning process leads to *improved email representations*. In

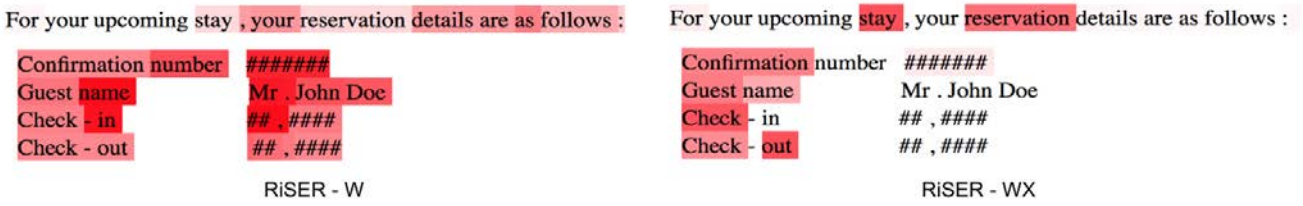


Figure 4: Word attention weights on an example donated Hotel email using the RiSER-W and RiSER-WX variants. Brighter red highlighting indicates a higher attention weight for the corresponding term relative to the other terms in the email.

Section 4.5, we illustrate this with some example emails in which the inclusion of HTML structure results in better classification.

While improvements of the order of 5% to 10% may seem small, they represent a significant reduction in "bad experiences" for users of Gmail. Consider, for example, a service that extracts structured data from these emails to remind people when their bills are due or answers questions about their hotel reservations. The improved models directly translate to improved recall for both these assistive experiences.

4.4.2 **AUC-PR.** The AUC-PR metric generally improves for RiSER-WA, RiSER-WX and RiSER-WXA variants compared to the baseline variant on both classification tasks. However, the improvements are not always in line with the improvements to the recall-at-fixed-precision metric. Surprisingly, unlike the R@P metric, the RiSER-WXA variant does not have the highest AUC-PR for either task. For the Bill task, the RiSER-WX variant performs the best (86.2%), while for the Hotel task, the RiSER-WA variant has the highest AUC-PR (98.0%).

4.5 Qualitative Analysis

In order to get an intuition into how RiSER utilizes the HTML structure of an email, we visualize and compare the attention weights of two architecture variants: one that excludes the XPath encodings and one that includes the XPath encodings. Here, we use the RiSER-W and RiSER-WX variants trained for the Hotel classification task. We illustrate the resulting attention weights of the two models applied to a single donated email example in Figure 4. Note that we obfuscate key information for privacy reasons.

In comparing the term attention weights of the RiSER-W and RiSER-WX models, we find that the latter is much better at attending to terms that are semantically relevant to the hotel reservation class. For example, terms such as `'stay'`, `'reservation'`, `'confirmation'`, and `'guest'` are attended to while the RiSER-W model attends to nearly all words. In fact, it attends more to those that are less semantically relevant in this example.

To get an intuition as to why structural information helps these models attend differently to these terms, we inspect the attention weights of the XPath tags in addition to the term attention weights. We visualize these weights for a donated email example in Figure 5. Red highlighting indicates term attention weights while yellow highlighting indicates the XPath's HTML tag attention weights.

Our first observation is that the use of structure helps the model understand the boundaries of different parts of the email. In this particular example the model attends to terms that belong to the subject more than the remainder of the text stream. Note that the word attention weights drop suddenly at the boundary of the subject and the beginning of the message body. We observed this pattern across many donated examples.

Our second observation is that the model also attends more to HTML tags that are often used to highlight important text. In this example, the model pays particular attention to meaningful tags such as `strong` (which indicates if a word is bold-ed), `h1`/`h2`/`h3` (which indicate headers), and `ul` and `li` (which indicate list items). These types of markup are generally used to draw the user's attention to specific text in a rendered email, often containing important information. Similarly, note that the `center` tag, which aligns all inner HTML to the center of the display, is also highly attended to.

In summary, from inspecting the attention weights of the RiSER architecture with and without XPath encodings, we observe that including structural information improves attention at the term

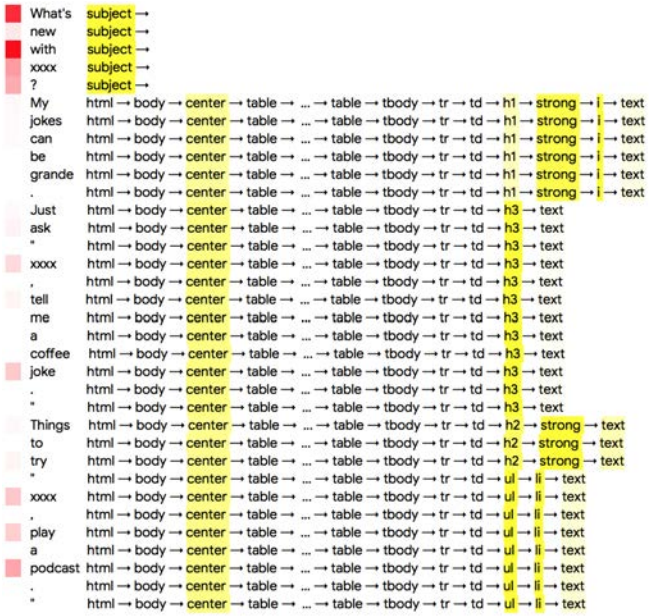


Figure 5: Term and XPath attention weights. Red highlighting indicates the term attention weights. Yellow highlighting indicates the XPath's HTML tag attention weights (row normalized). Brighter colors indicate higher attention weights.

level by helping the model learn about boundaries in documents (e.g. subject vs. message body), as well as learn to focus on parts that the original sender intended to draw the user's attention to (e.g. `strong`, `center`, etc.).

5 RELATED WORK

Our work builds on contributions from several research areas including natural language processing, data mining, web mining, and deep learning. Here we discuss related work in two main categories: text representation and email representation.

There is a rich history of work in text representation [33, 47] for various tasks like sentiment classification [37] (e.g. positive or negative reviews) and genre detection (e.g. sports, news, entertainment, etc.). More recently, the availability of large amounts of text data has led to unsupervised approaches for learning a good distributed representation for words. Techniques like Word2Vec [34] and Glove [38] have produced pre-trained embeddings for words that can be combined with more complex neural approaches.

The success of these approaches are later combined with the power of recurrent neural networks (RNNs) which led others to learn distributed representations of a larger piece of text like sentences [26], paragraphs [28], and documents [25]. Several variants of RNNs like convolutional-gated RNN [45], tree-LSTM [44] and Quasi-RNN [10] are proposed as alternative architectures. Apart from recurrent neural networks, methods employing convolutional networks [23, 49] have also produced compelling results for text classification tasks.

Recently, authors in [48] have argued that the hierarchical structure of documents helps to learn better representations. Acknowledging that the importance of words or sentences may vary depending on the context, they use an attention mechanism [5] while hierarchically encoding words into sentences and sentences into documents. In addition to the hierarchical structure, discourse structure [32] in the text, which represents the linguistic organization of a text as a tree, has also been studied in recent document representation work [8, 22, 29].

Each of these lines of research are based solely on the textual content assuming that the documents are plain-text. In this paper, we focus on machine-generated emails which have a rich HTML structure. In addition to leveraging the textual context, we encode the structural markup to learn a better representation. Note that in our work, the term "structure" corresponds to a formatting structure of an email, rather than the linguistic structure mentioned above. Combining our framework with the aforementioned architectures is possible and makes for promising future work.

Learning good email representations has been an area of interest for multiple research groups [2, 14, 41]. In addition to information extraction, email classifiers are useful for other applications like automatic foldering [6, 19], spam classification [13], and message priority ranking [1].

Lastly, systems like Fonduer [46] tackle richly formatted documents for the task of Knowledge Base Construction (KBC). Fonduer shows that augmenting textual features with structural and visual features produced by a library that parses the HTML markup and providing it to a strong baseline like a bi-directional LSTM with attention can indeed provide a significant improvement over just using the text features. Our results agree with the findings in Fonduer – formatting information does indeed help us learn a better representation for an email. In contrast to the approach of engineering a set of structural features (examples from Fonduer include HTML tag of the term, HTML tag of the parent, tabular features like row number, column number, n-grams from all the cells in the same row) our approach simply relies on learning an encoding using the sequence of HTML tags that appear in the path from the root to the node containing the term. Our results show that this is a powerful approach when combined with well-known building blocks like LSTMs and an attention mechanism. While we do not implement the nearly 40 structural, tabular, and visual features implemented in Fonduer, an interesting piece of future work is to study if XPath encodings as learned by our model can be as effective as the sophisticated features manually engineered in Fonduer.

6 CONCLUSION

In this paper, we study the problem of learning representations for richly structured emails. Considering the fact that most of today's emails are machine generated, we argue the rich formatting used in these emails contains highly valuable information that can be used to learn better representations. To address this gap in the literature, we proposed a novel framework called RiSER. Our framework combines three components from the email in order to learn an enhanced email representation: 1) textual content, 2) HTML structure, 3) manual annotation features. We show the effectiveness of our approach by evaluating the individual components as well as the full framework on email classification task across two datasets from a large email service. The experimental results and visualizations further indicate that our framework is learning an improved representation for emails and is capable of capturing valuable information buried in the email's HTML structure.

We are exploring multiple avenues of future research. We expect that leveraging structure will be valuable for other challenging tasks like information extraction from emails and web pages. We are considering ways to extend the RiSER framework to other types of documents with rich structure. For instance, PDFs and scanned images contain many informative layout and structural signals. However such documents do not contain XPath-style markup. We are exploring alternative ways to represent the layout and formatting information from such documents.

REFERENCES

[1] Douglas Aberdeen, Ondrej Pacovsky, and Andrew Slater. 2010. The learning behind Gmail priority inbox. In *LCCC: NIPS 2010 Workshop on Learning on Cores, Clusters and Clouds*.

[2] Manoj K Agarwal and Jitendra Singh. 2018. Template Trees: Extracting Actionable Information from Machine Generated Emails. In *Proceedings of the 30th International Conference on Database and Expert Systems Applications (DEXA)*. 3–18.

[3] Nir Ailon, Zohar S Karnin, Edo Liberty, and Yoelle Maarek. 2013. Threading machine generated email. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining (WSDM)*. 405–414.

[4] Noa Avigdor-Elgrabli, Mark Cwalinski, Dotan Di Castro, Iftah Gamzu, Irena Grabovitch-Zuyev, Liane Lewin-Eytan, and Yoelle Maarek. 2016. Structural clustering of machine-generated mail. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM)*. 217–226.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[6] Ron Bekkerman, Andrew McCallum, and Gary Huang. 2005. *Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora.* Center for Intelligent Information Retrieval report IR-418. University of Massachusetts.

[7] Anders Berglund, Scott Boag, Don Chamberlin, Mary F Fernández, Michael Kay, Jonathan Robie, and Jérôme Siméon. 2007. XML Path Language (XPath). *World Wide Web Consortium (W3C)* (2007).

[8] Parminder Bhatia, Yangfeng Ji, and Jacob Eisenstein. 2015. Better Document-level Sentiment Analysis from RST Discourse Parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2212–2218.

[9] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606* (2016).

[10] James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2016. Quasi-recurrent neural networks. *arXiv preprint arXiv:1611.01576* (2016).

[11] Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. 2003. Extracting content structure for web pages based on visual representation. In *Asia-Pacific Web Conference*. Springer, 406–417.

[12] Godwin Caruana and Maozhen Li. 2012. A survey of emerging approaches to spam filtering. *ACM Computing Surveys (CSUR)* 44, 2 (2012), 9.

[13] Gordon V Cormack. 2008. Email spam filtering: A systematic review. *Foundations and Trends® in Information Retrieval* 1, 4 (2008), 335–455.

[14] Dotan Di Castro, Iftah Gamzu, Irena Grabovitch-Zuyev, Liane Lewin-Eytan, Abhinav Pundir, Nil Ratan Sahoo, and Michael Viderman. 2018. Automated Extractions for Machine Generated Mail. In *Companion Proceedings of the The Web Conference 2018 (WWW)*. 655–662.

[15] Dotan Di Castro, Zohar Karnin, Liane Lewin-Eytan, and Yoelle Maarek. 2016. You've got mail, and here is what you could do with it!: Analyzing and predicting actions on email messages. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining (WSDM)*. 307–316.

[16] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)* 12, Jul (2011), 2121–2159.

[17] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D Sculley. 2017. Google Vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1487–1495.

[18] Google. 2019. Analyzing Entities. https://cloud.google.com/natural-language/docs/analyzing-entities.

[19] Mihajlo Grbovic, Guy Halawi, Zohar Karnin, and Yoelle Maarek. 2014. How many folders do you really need?: Classifying email into a handful of categories. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM)*. 869–878.

[20] Ramanathan V Guha, Dan Brickley, and Steve Macbeth. 2016. Schema. org: evolution of structured data on the web. *Commun. ACM* 59, 2 (2016), 44–51.

[21] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[22] Yangfeng Ji and Noah Smith. 2017. Neural discourse structure for text categorization. *arXiv preprint arXiv:1702.01829* (2017).

[23] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[24] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[25] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*. 2267–2273.

[26] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*. 1188–1196.

[27] Philippe Le Hégaret, Ray Whitmer, and Lauren Wood. 2005. Document Object Model (DOM). http://www.w3.org/DOM.

[28] Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057* (2015).

[29] Yang Liu and Mirella Lapata. 2018. Learning structured text representations. *Transactions of the Association of Computational Linguistics* 6 (2018), 63–75.

[30] Yoelle Maarek. 2016. Is Mail The Next Frontier In Search And Data Mining?. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining (WSDM)*. 203–203.

[31] Yoelle Maarek. 2017. Web Mail is not Dead!: It's Just Not Human Anymore. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*. 5–5.

[32] William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse* 8, 3 (1988), 243–281.

[33] Andrew McCallum, Kamal Nigam, et al. 1998. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, Vol. 752. 41–48.

[34] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[35] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS)*. 3111–3119.

[36] Marc Najork. 2009. Web spam detection. In *Encyclopedia of Database Systems*. Springer, 3520–3523.

[37] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 79–86.

[38] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[39] Xiaoguang Qi and Brian D Davison. 2009. Web page classification: Features and algorithms. *ACM Computing Surveys (CSUR)* 41, 2 (2009), 12.

[40] Herbert L Roitblat, Anne Kershaw, and Patrick Oot. 2010. Document categorization in legal electronic discovery: computer classification vs. manual review. *Journal of the American Society for Information Science and Technology (JASIST)* 61, 1 (2010), 70–80.

[41] Ying Sheng, Sandeep Tata, James B Wendt, Jing Xie, Qi Zhao, and Marc Najork. 2018. Anatomy of a Privacy-Safe Large-Scale Information Extraction System Over Email. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. 734–743.

[42] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter* 19, 1 (2017), 22–36.

[43] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1631–1642.

[44] Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* (2015).

[45] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. 1422–1432.

[46] Sen Wu, Luke Hsiao, Xiao Cheng, Braden Hancock, Theodoros Rekatsinas, Philip Levis, and Christopher Ré. 2018. Fonduer: Knowledge base construction from richly formatted data. In *Proceedings of the 2018 International Conference on Management of Data (SIGMOD)*. 1301–1316.

[47] Yiming Yang and Xin Liu. 1999. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 42–49.

[48] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. 1480–1489.

[49] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS)*. 649–657.

Reputation Deflation Through Dynamic Expertise Assessment in Online Labor Markets

Marios kokkodis
Boston College
kokkodis@bc.edu

ABSTRACT

Current reputation systems in online labor markets (e.g., Freelancer, PeoplePerHour) experience three major shortcomings: (1) reputation inflation (i.e., reputation scores are inflated to "above average" values) (2) reputation attribution (i.e., attribution of reputation scores to individual skills is unfeasible) and (3) reputation staticity (i.e., reputation scores are uniformly averaged over time). These shortcomings render online reputation systems uninformative, and sometimes even misleading. This work proposes a data-driven approach that deflates reputation scores by solving the problems of reputation attribution and saticity. The deflating process starts with projecting any random set of skills to a set of competency dimensions. For each competency dimension, a Hidden Markov Model estimates a contractor's current (but latent) competency-specific expertise. Aggregation of competency-specific estimates provides expertise predictions for any given set of required skills. Empirical analysis on 61,330 completed tasks from a major online labor market shows that the resulting estimates are deflated and they better predict contractor performance. These results suggest a series of implications for online (labor) markets and their users.

CCS CONCEPTS

• **Information systems → Data analytics**.

KEYWORDS

Reputation deflation, Expertise assessment, Digital markets, Online markets, HMM, Word embeddings

ACM Reference Format:
Marios kokkodis. 2019. Reputation Deflation Through Dynamic Expertise Assessment in Online Labor Markets. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313479

1 INTRODUCTION

Online labor markets such as Peopleperhour and Freelancer connect buyers with contractors around the globe to accomplish a diverse set of tasks. These tasks span multiple categories, including web development, graphic design, accounting and sales and marketing. On par with other online platforms, online labor markets have experienced an accelerated growth in the past decade [43]. A typical transaction in these markets starts with a buyer posting a job description. Contractors who are looking for opportunities submit their job applications to relevant jobs. The buyer then chooses to hire one (or more) contractor(s) from the pool of applicants. Once hired, the contractors complete the task and receive the prearranged compensations.

To identify the best candidate for each opening, buyers assess a series of observed and latent characteristics of the available contractors. Observed characteristics include the contractors' skills, work histories, and certifications. Latent characteristics include the contractors' true knowledge and abilities. The existence of latent characteristics, the heterogeneity that appears in the observed ones [26], and the interactions between the two create an uncertain environment of information asymmetry [4].

To reduce information asymmetry and help buyers make better-informed hiring decisions, online labor markets have developed reputation systems. Contractors get rated for the tasks they complete, and these ratings become part of their online resumes. Buyers then consider the contractors' past performance as a signal for future performance. In theory, such reputation mechanisms should inform and facilitate buyers' choices. In practice however, current reputation systems experience three major shortcomings:

- Reputation inflation: In online labor markets, raters feel pressure to rate "above average" [14]. At the same time, contractors who receive low ratings are unable to get hired. As a result, they leave the market and look for alternatives, or rejoin with different credentials [20]. These two synergic forces result in a highly inflated rating distribution [14, 19].
- Reputation attribution: Current reputation systems in online labor markets provide a uni-dimensional accumulated reputation score for each contractor [41]. Because these markets support heterogeneous distributions of skills and qualifications [26], accurate attribution of accumulated reputation scores to individual skills is unfeasible. For instance, consider a contractor who completes a task that requires `java`, `SQL` and `python` and receives a feedback score 9/10. Does this score capture the contractor's performance on `java`, on `SQL`, on `python`, or on any combinations of those skills?
- Reputation staticity: Over time, contractors evolve, as they gain expertise on skills that they repeatedly use. Current reputation systems however uniformly average received ratings to provide an aggregate score, thereby assuming that the latent expertise of a contractor does not change over time [27].

The end result of these shortcomings is that current reputation scores are often uninformative and sometimes even misleading.

This work proposes a data-driven framework that directly solves reputation attribution and reputation staticity, and, indirectly, deflates the reputation distribution. The framework first incorporates

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313479

word embeddings to project skills into a set of competency dimensions. For each one of these competency dimensions a Hidden Markov Model (HMM) uses observed signals to dynamically estimate a contractor's competency-specific level of expertise. These estimates are a solution to reputation attribution, as they map expertise to specific skillsets. Because the HMMs allow contractors to dynamically evolve to various expertise states, they provide *current* estimates of expertise (solution to reputation staticity). Aggregation of competency-specific expertise estimates provides expertise predictions for any given set of required skills (HMM-W2V reputation).

In theory, compared to current reputation scores in online labor markets, HMM-W2V reputation should be closer to the true expertise of contractors on a given set of skills. Since these markets attract millions of contractors of various abilities, HMM-W2V reputation is likely to form a deflated, Paretian distribution [38]. Empirical evaluation of the HMM framework on 61,330 completed tasks from a major online labor market shows that, compared to current reputation scores, HMM-W2V reputation is (1) deflated, and (2) a better predictor of contractor performance on a any given set of skills.

This work is the first to propose a data-driven solution that dynamically estimates contractors' expertise in online labor markets. Competency-specific estimates provide deflated scores for the dynamic expertise of contractors on any given set of skills. These estimates can improve an online labor market's transaction efficacy through better and more relevant candidate recommendations [30]. Better-informed buyers are likely to make more successful hiring decisions (through accurate opening-specific expertise estimates) and as a result continue using the platform. At the same time, the market can better understand the demand and supply of experienced contractors in every competency dimension, and create any necessary interventions. Finally, the proposed framework generalizes beyond online labor markets, and it is directly applicable to other markets that suffer from reputation inflation, such as Amazon, Yelp, and TripAdvisor.

2 RESEARCH CONTEXT

Digital markets signal the quality of their services (or products) through online reputation systems. The strong economic impact of these systems on product selection and quality assessment has been repeatedly verified in multiple contexts [1, 3, 12, 13, 23, 29, 44]. Next, we discuss the importance of reputation signals in the context of an online labor market.

2.1 Reputation in online labor markets

Buyers place a very significant weight on contractors' reputation when making hiring decisions [45], and they are willing to trade-off reputation and price to accept higher bids posted by more reputable contractors [36]. This trade-off is also observed on the contractor side, where the contractor's past performance has a significant impact on future wages [7]. Reputation can be a significant predictor of success for fixed-price contracts [32], and it can substantially improve contractors' subsequent employment outcomes [39]. Furthermore, reputation can capture latent qualities of a contractor that transfer across multiple task categories that require a diverse set of skills [27]. Overall, these works show that reputation drives both hiring decisions and subsequent task outcomes.

Table 1: Data summary

	Obs.	Min	Mean	Median	Max	StD
Hourly wage	61,330	3	13.6	11.1	219	10.6
Skills per user	671,340	1	9.6	9	61	5.9
Skills per opening	61,330	1	2.4	2	21	1.7
Applications	671,340					
Hires	61,330					
Openings	61,330					
Skills ($\mathbb{R}$)	215					
Unique skillsets	17,449					

Contractors come from 182 countries. The dataset spans 4 years.

2.2 Reputation inflation

Response bias, i.e., who chooses to rate a service, drives the overall rating of a service more so than the actual service characteristics [35]. One type of response bias is acquisition bias, which recognizes that buyers typically choose services that they expect to like [18]. This predisposition of buyers introduces a bias in the sample of users who are able to rate a service, which partially explains the observation that overall ratings tend to be positively inflated on most digital platforms [19].

Reputation inflation is present in online labor markets as well [14]. Even state-of-the art bilateral feedback systems [8] suffer from inflation [14]. However, acquisition bias might not be the only driving force of this inflation. One reason for this is "Customer death" [20]: users that receive low feedback scores leave the marketplace since they cannot get hired, and they either create new accounts and rejoin, or join alternative marketplaces to use. An alternative explanation is that raters likely feel pressure to assign good ratings due to inefficiencies of current reputation systems [14]. The proposed dynamic framework in this work tackles these inefficiencies from a design science perspective: it uses information from multiple observed signals in an online labor market to create a deflated, informative reputation.

2.3 Reputation as a signal of expertise

In theory, reputation systems resolve information asymmetries by providing an accurate estimate of the service (or product) quality [12]. In the context of online labor markets, service quality is bound to the expertise of a contractor on a given set of skills. Hence, ideally, reputation scores in online labor markets should signal a contractor's level of expertise. In practice however, due to reputation inflation reputation attribution and reputation staticity, current reputation scores fail to accurate assess a contractor's expertise.

The focal framework of this study relies on accurate skillset-specific expertise assessment. Expertise is latent, and its assessment is a very hard task due to lack of observable signals [26]. Item Response Theory [16] is the classic test-based approach for assessing

expertise on a given set of skills. Other approaches use observational data to identify experts (in online communities) through analyses of networks [21, 40, 46] and static user profiles [6].

In the context of an online labor market there are multiple observed signals that draw a more complete picture of a contractor's expertise. Such signals include the total number of completed jobs and the contractor's premium (hourly compensation). Furthermore, through observing contractors over multiple years, online labor markets become aware of dynamic changes in user profiles (e.g., new skill acquisitions, wage fluctuations, etc.). The proposed framework incorporates these observed signals to estimate the up-to-date expertise of a contractor on any given set of skills.

3 REPUTATION INFLATION IN AN ONLINE LABOR MARKET

The focal data forms a snapshot of 671,340 job applications that led to 61,330 completed tasks by 17,510 contractors from a major online labor market, `GigWork` (pseudonym). `GigWork` supports a diverse set of tasks, including software and web development, writing and translation, sales and marketing, data science, etc. The dataset spans four years, and it considers only contractors who have completed two or more tasks. For these contractors the dataset includes complete information, including any new skill acquisition, and the complete set of their job applications.

Table 1 summarizes the dataset. Overall, the contractor profiles and job openings span across 215 skills. The contractors come from 182 different countries, and they complete tasks for which they earn between $3 and $219 per hour.

On `GigWork`, buyers rate contractors after the completion of a task with a score $Y \in \{0, 1/9, 2/9, \ldots, 9/9\}$. The platform supports a state of the art reputation system: The buyer and the contractor each gets two weeks to leave their feedback score – however this is a double blind process: neither party learns its own rating before leaving a rating for the other party. Figure 1 shows the resulting accumulated feedback distribution on `GigWork`. The mean of this distribution is 0.96 and the median 0.98. Simply put, most of the contractors appear to have an almost perfect reputation. Hence, it is fair to say that `GigWork` suffers from reputation inflation.

At the same time, the heterogeneity in terms of skills and qualifications on the platform in combination with the fact that feedback scores are assigned uniformly (reputation attribution) adds to the noise of the inflated distribution. Figure 2 shows that skills accumulate vastly different feedback scores: from `translation`, with median 1 and mean 0.91, to `twitter-marketing` with median 7/9 and mean 0.74. This variation suggests that there is a need for skill-specific reputation.

4 PROBLEM FORMULATION AND METHODOLOGY

Independent contractors in an online labor market are on the lookout for suitable opportunities. When they find good matches, contractors submit job applications and (some of them) eventually get hired. Once hired, they complete the assigned tasks and receive a pre-arranged compensation along with the respective performance feedback scores. Due to inefficiencies of current reputation systems,

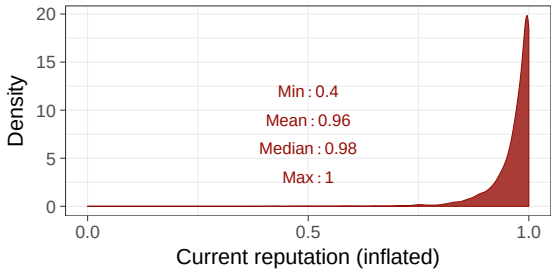


Figure 1: Reputation inflation on `GigWork`. Accumulated contractor feedback scores are skewed to the right. The mean is 0.96 and the median 0.98.

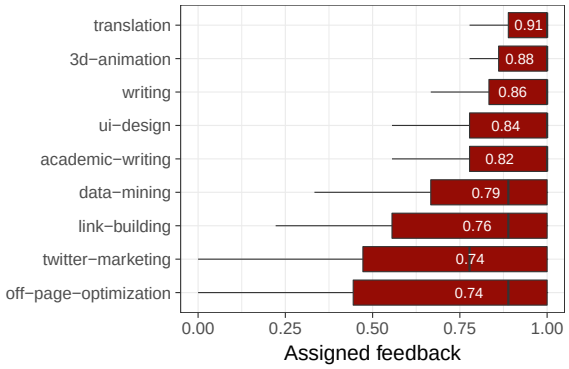


Figure 2: Boxplots of assigned feedback scores for a subset of skills. Each boxplot includes the mean feedback score for the respective skill.

performance feedback scores are inflated (Figure 1), and have high variability across different skills (Figure 2).

Problem definition: *Given a contractor's history on the market, our goal is to estimate a score that captures the current, deflated and true contractor's expertise on any given set of skills.*

The proposed framework (`HMM-W2V-framework`) consists of three components. The first component focuses on decomposing skills into competency dimensions that allows the framework to generalize and accommodate any arbitrary number of skills. The second component presents a dynamic model that combines multiple signals to estimate a contractor's current, competency-specific expertise. The final component aggregates competency-specific predictions to get a current estimate of the contractor's expertise on any arbitrary set of skills. Figure 3 draws the interconnections of these three components, which we describe in detail next.

4.1 Component A: Skills decomposition

Buyers can request contractors with any arbitrary combination of skills. This creates a space of tens of thousands of available combinations of unique skillsets (Table 1). Theoretically, we could directly estimate the expertise of a contractor on any observed

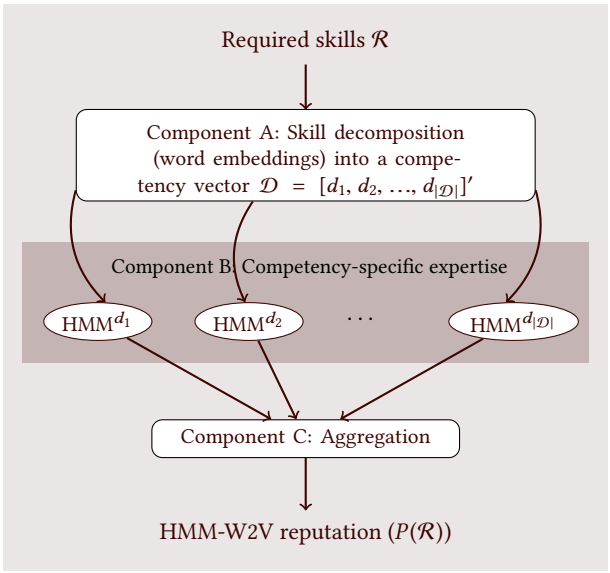


Figure 3: The three components of the `HMM-W2V-framework`. Component A maps any set of skills to competency dimensions through word embeddings. Component B provides dynamic models (HMM) that make competency-specific and up-to-date expertise estimates. Component C aggregates these estimates to provide a skillset-specific expertise assessment.

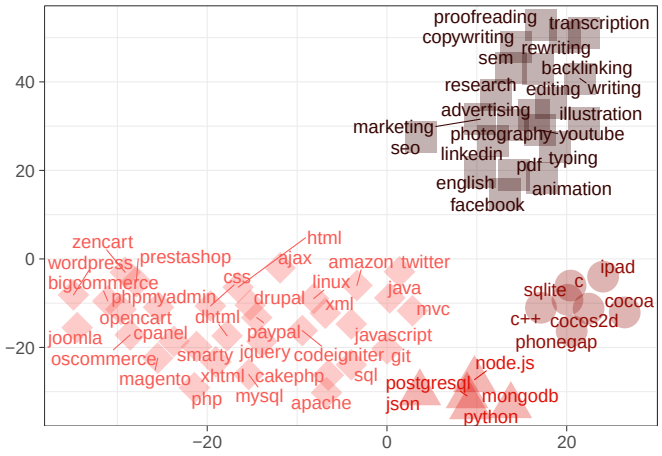


Figure 4: Visualization of mapped skills through W2V in a reduced, two dimensional space through stochastic neighbor embedding [17]. Contextually similar skills appear close to each other (e.g., `c++, c, sqlite`), while contextually dissimilar skills appear far away from each other (e.g., `mongodb` vs. `proofreading`).

skillset. There are three major drawbacks of this approach: first, independent of the size of the data analyzed, considering only skillset-specific observations will result in very sparse training datasets. Second, expertise is transferable [25, 27] and many skillsets are highly correlated. Third, an entrance of new skills would require retraining of the complete framework from scratch.

To overcome these shortcomings we use a distributed representation of words model (word embeddings–W2V) [2, 33] that projects individual skills into a set of competency dimensions. W2V embed words from a vocabulary into a lower dimensional space (i.e., competency dimensions), in which semantically similar words appear close to each other, while semantically dissimilar words appear far away from each other [33]. In the context of an online labor market, a "skill" maps to a "word," and a "skillset" to a "document." Based on this representation, W2V projects contextually similar skills close to each other in the $|\mathcal{D}|$-dimensional space of competencies. (The actual number of competency dimensions $|\mathcal{D}|$ is an input parameter.)

Figure 4 shows how a randomly selected subset of skills maps into a reduced two-dimensional space. The plot reveals hidden contextual similarities between skills; For instance it shows that buyers who request `c++` usually also request `sqlite`, while buyers who request `python` usually also request `mongodb`. The plot hence indicates that `python` is contextually closer to `mongodb` than `sqlite`.

The `HMM-W2V-framework` maps any observed skillset $\mathcal{R}$ into a $|\mathcal{D}|$-dimensional vector space of competencies (Figure 3). To do so, it aggregates competency-specific scores of each skill in a given skillset. Specifically, a skillset $\mathcal{R} = \{R_1, R_2, \ldots, R_{|\mathcal{R}|}\}$, $R_k \in \mathbb{R}$ maps to an aggregated representation as follows:

$$w^d_{\mathcal{R}} = \sum_{R \in \mathcal{R}} \text{W2V}^d(R), \quad \forall d \in \mathcal{D}\,, \tag{1}$$

where $w^d_{\mathcal{R}}$ is the d–competency score of skillset $\mathcal{R}$, and $\text{W2V}^d(R)$ is the W2V score for skill R in dimension d. (Alternatively, we could use a distributed memory model (document embeddings, D2V) [31]. In practice, W2V outperformed D2V in this context.)

4.2 Component B: Competency-specific expertise

At any given time (t), each participating contractor has a level of expertise in each available competency dimension $d \in \mathcal{D}$. This expertise is latent (unobserved). However, for each completed task with required skills $\mathcal{R}$ we observe a feedback score (Y_t) that the contractor receives, which maps into competency-specific scores $Y^d_t = w^d_{\mathcal{R}} Y_t, \forall d \in \mathcal{D}$. These scores ($Y^d_t$) form up-to-date proxies of the contractor's expertise in each competency dimension $d \in \mathcal{D}$.

In addition to latent, a contractor's expertise dynamically evolves with time. Contractors gain (or lose) expertise in any competency dimension through learning new skills, improving their knowledge in others, or by completely abandoning skills that the market deems obsolete. The `HMM-W2V-framework` formulates this evolution through a structured Hidden Markov Model (HMM [15, 28]). At any point in time t, contractors operate from a latent, competency-specific state, which determines their propensity to perform with score Y^d_t. Depending on these performance scores on completed tasks, contractors stochastically transition to new latent states. The framework assumes a set of $\mathcal{S}^d$ latent states that describe K^d different levels of expertise for each competency d, $\mathcal{S}^d$ =

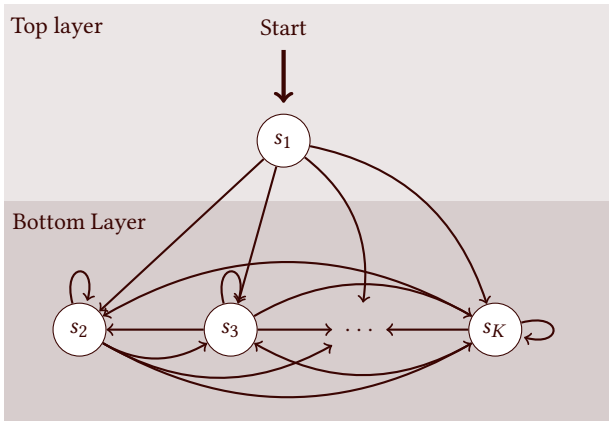


Figure 5: Two-layered structure of each competency-specific HMM. The top layer includes an initial state s_1, which clusters all contractors when they first join the platform. Once contractors start completing tasks, they emit observations and transition to an appropriate state in the bottom layer.

$\{s_1, s_2, \ldots, s_{K^d}\}$. The rest of the paper assumes that the unit of time (t) is a new completed task.

HMM structure: Every new contractor who joins the platform has a certain level of expertise across the competency dimensions $d \in \mathcal{D}$. As the contractor completes new tasks on the platform, we observe signals of the contractor's latent expertise. To capture this behavior, the `HMM-W2V-framework` imposes a two-layer structure: In the first layer, it assume only an initial latent state s_1, where all new contractors land. This state makes an average initial estimate of a contractor's competency-specific expertise. (Alternatively, contractors could land stochastically to one of the second layer states. This would add noise to the estimates, and as a result it could potentially hurt the performance of the framework.) Once the contractors complete their first job and emit an observation (i.e., $Y_1^d, \quad \forall d \in \mathcal{D}$), they stochastically transition to one of the $K^d - 1$ states of the second layer. Figure 5 presents the structure with the two layers and derives the possible transitions from each state.

To define an HMM for a given competency dimension d, we need (1) a vector of initial state probabilities $\boldsymbol{\pi}^d$, (2) a transition matrix T^d that stores the transition probabilities between states, and (3) an emission matrix E^d that describes the state-specific probability distributions for observations Y_t^d. Since every new contractor lands in state s_1, the initial probability vector of the HMM is the following:

$$\boldsymbol{\pi}^d = [1, 0, 0, ..., 0]' \ . \tag{2}$$

A contractor's history provides multiple observable signals that drive transitions to new expertise states (e.g., received feedback scores in previous tasks, total wages received, hiring rates, number of completed tasks). Such historical attributes define a vector $\boldsymbol{Z}_t^d$, which directly affects the transition probability matrix T^d. Formally, we define the transition probability of a given contractor for a given dimension to move from state s_k to state s_l at time t as follows:

$$\lambda^{d,s_k s_l}_{\boldsymbol{\gamma}_{kl}^d \boldsymbol{Z}_{t-1}^d} = \Pr(S_t^d = s_l | S_{t-1}^d = s_k; \boldsymbol{\gamma}_{kl}^d, \boldsymbol{Z}_{t-1}^d) = \text{softmax}(\boldsymbol{\gamma}_{kl}^d \boldsymbol{Z}_{t-1}^d) \ . \tag{3}$$

In the previous Equation, $\boldsymbol{\gamma}_{kl}^d$ is the vector of coefficients of state s_k that define the weights of $\boldsymbol{Z}_{t-1}^d$ in estimating the transition probability to state s_l.

Because of the HMM structure (Figure 5), the transition matrix is not completely filled with elements of Equation 3. Specifically, at time t, the transition matrix has the following form:

$$T^d(\boldsymbol{\Gamma}^d, \boldsymbol{Z}_{t-1}^d) = \begin{bmatrix} 0 & \lambda^{d,s_1 s_2}_{\boldsymbol{\gamma}_{12}^d \boldsymbol{Z}_{t-1}^d} & \cdots & \lambda^{s_1 s_{K^d}}_{\boldsymbol{\gamma}_{1K^d}^d \boldsymbol{Z}_{t-1}} \\ 0 & \lambda^{d,s_2 s_2}_{\boldsymbol{\gamma}_{22}^d \boldsymbol{Z}_{t-1}^d} & \cdots & \lambda^{d,s_2 s_{K^d}}_{\boldsymbol{\gamma}_{2K^d}^d \boldsymbol{Z}_{t-1}^d} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \lambda^{d,s_{K^d} s_2}_{\boldsymbol{\gamma}_{K^2}^d \boldsymbol{Z}_{t-1}^d} & \vdots & \lambda^{s_{K^d} s_{K^d}}_{\boldsymbol{\gamma}_{K^d K^d}^d \boldsymbol{Z}_{t-1}^d} \end{bmatrix} , \tag{4}$$

where $\boldsymbol{\Gamma}^d = [\boldsymbol{\gamma}_{11}^d, \boldsymbol{\gamma}_{12}^d, \ldots, \boldsymbol{\gamma}_{K^d K^d}^d]'$.

Similarly, current, opening-specific observed characteristics (e.g., bidding price or whether or not a contractor has been invited to apply) drive the observed emissions of the HMM (matrix E^d). These characteristics form a vector $\boldsymbol{X}_t^d$ which affects the elements of the emission matrix. Formally, the conditional probability of observing Y_t^d given the current state of the contractor S_t^d is:

$$\Pr(Y_t^d | S_t^d = s_k; \boldsymbol{\theta}_k^d, \boldsymbol{X}_t^d) = f^d(\boldsymbol{\theta}_k^d \boldsymbol{X}_t^d) \ , \tag{5}$$

where $f^d(.)$ is a continuous probability distribution (e.g., Gaussian), and $\boldsymbol{\theta}_k^d$ is the parameter vector of the continuous distribution for state k and competency d. The vector of complete parameter vectors $\boldsymbol{\Theta}^d$ (for all states $s_k \in \mathcal{S}^d$) is as follows:

$$\boldsymbol{\Theta}^d = [\boldsymbol{\theta}_1^d, \boldsymbol{\theta}_2^d, ..., \boldsymbol{\theta}_{K^d}^d]' . \tag{6}$$

HMM identification: Given the structure of the HMM the focus turns on estimating the parameter vectors $\boldsymbol{\Theta}^d, \boldsymbol{\Gamma}^d$. To do so, we maximize the conditional probability of the set of observations given the HMM. (For simplicity, in the following analysis we drop the superscript d. However, keep in mind that this estimation process happens independently for each dimension $d \in \mathcal{D}$.)

Let us assume that we have the following sequence of M observations for a given contractor i:

$$\boldsymbol{Y}_i = Y_{i1}, Y_{i2}, \ldots, Y_{iM} \ . \tag{7}$$

These observations correspond to a sequence of input vectors:

$$\boldsymbol{X}_{1:M} = \boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_M \ . \tag{8}$$

Furthermore, let us assume that $\boldsymbol{Y}_i$ is the result of a sequence of latent states, $\boldsymbol{S}_i$:

$$\boldsymbol{S}_i = S_{i1}, S_{i2}, \ldots, S_{iM} \ , \tag{9}$$

where $S_{im} \in \mathcal{S}$. This sequence of states is affected by the sequence of historic vectors $\boldsymbol{Z}_{1:M-1}$:

$$\boldsymbol{Z}_{1:M-1} = \boldsymbol{Z}_1, \boldsymbol{Z}_2, \ldots, \boldsymbol{Z}_{M-1} \ . \tag{10}$$

Figure 6 shows these sequences along with their interactions. Based on the structure of the graph, we get the conditional likelihood of observing sequence $\boldsymbol{Y}_i$:

$$\Pr(\boldsymbol{Y}_i | \boldsymbol{S}_i; \boldsymbol{\Theta}, \boldsymbol{X}_{1:M}) = \prod_{t=1}^{M} \Pr(Y_{it} | S_{it}; \boldsymbol{\Theta}, \boldsymbol{X}_t) \ , \tag{11}$$

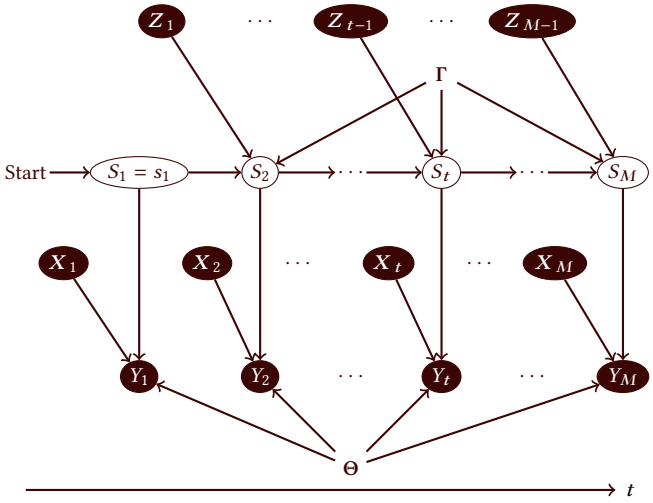


Figure 6: Temporal evolution of the HMM. The structure of the latent state sequence S_i, the observed sequence of outcomes Y_i, the parameter vectors Θ, Γ, and the sequences of input vectors $X_{1:M}, Z_{1:M-1}$ for a given contractor i. For better readability we have dropped the contractor subscript i and the competency superscript d. As with traditional probabilistic graphical models, latent states are in clear ellipses, and observed features in shaded ones.

where Equation 5 estimates the right hand side. From Figure 6, the conditional probability of observing the sequence S_i is:

$$\Pr(S_i|\Gamma, Z_{1:M-1}) = \boldsymbol{\pi}(S_1) \prod_{t=2}^{M} \Pr(S_{it}|S_{it-1}; \Gamma, Z_{t-1}) \ , \tag{12}$$

where $\boldsymbol{\pi}(S_1)$ is the the prior probability of being at state S_1. Equation 3 estimates these transition probabilities. Since the structure of the HMM (Figure 5) imposes that every new contractor lands in state s_1 ($\boldsymbol{\pi}(S_1 = s_1) = 1$), the previous equation becomes:

$$\Pr(S_i|\Gamma, Z_{1:M-1}) = \prod_{t=2}^{M} \Pr(S_{it}|S_{it-1}; \Gamma, Z_{t-1}) \ . \tag{13}$$

Based on this analysis and the graph in Figure 6, the likelihood of this sequence of observations for contractor i is as follows:

$$\begin{aligned} l(Y_i; \Theta, \Gamma) &= \Pr(Y_i|\Theta, \Gamma, X_{1:M}, Z_{1:M-1}) \\ &= \sum_{\forall S_i} \Pr(Y_i, S_i|\Theta, \Gamma, X_{1:M}, Z_{1:M-1}) \\ &\overset{\text{Figure 6}}{=} \sum_{\forall S_i} \Pr(Y_i|S_i; \Theta, X_{1:M}) \Pr(S_i|\Gamma, Z_{1:M-1}) \\ &= Pr(Y_{i1}|S_{i1}; \Theta, X_1) \\ &\times \sum_{\forall S_i} \prod_{t=2}^{M} \Pr(Y_{it}|S_{it}; \Theta, X_t) \\ &\times \Pr(S_{it}|S_{it-1}; \Gamma, Z_{t-1}), \end{aligned} \tag{14}$$

where we used the structure of the HMM to decompose the joint probability of $\Pr(Y_i, S_i|\Theta, \Gamma, X_{1:M}, Z_{1:M-1})$. Then, the complete likelihood for a dataset with N contractors is:

$$L(\Theta, \Gamma) = \prod_{i=1}^{N} l(Y_i; \Theta, \Gamma) \ . \tag{15}$$

Maximization of this likelihood estimates the parameters Θ, Γ. We do this numerically through the limited memory Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [9]. (In practice we minimize the negative log-likelihood.)

Table 2: Descriptive statistics for the attributes in vectors X, Z, and the outcome variable Y_{it}.

		Mean	Median	StD	Min	Max
	Observed outcome (Y_{it})	0.86	1	0.22	0	1
Vector Z_t	Current reputation (FB_{it})	0.96	0.98	0.05	0.4	1
	Total money earned ($)	202	44	883	3	56,433
	Completed jobs	3.14	2	2.17	2	39
	Hiring rate	0.20	0.09	0.26	0.002	1
Vector X_t	Invited	0.3	0	0.46	0	1
	Hourly bid ($)	11.93	8.89	12.31	3	400
	Buyer's reputation	0.93	1	0.13	0	1
	Rehires	0.02	0	0.1	0	1

4.3 Component C: Aggregation

This process happens independently for each competency dimension $d \in \mathcal{D}$. As a result, for a given contractor i, each HMM estimates a current (t) competency-specific expertise (p_{it}^d). To predict the expertise of a contractor for a given opening with a set of required skills $\mathcal{R}$ the `HMM-W2V-framework` aggregates the available competency-specific estimates. Let us define a vector with all the competency-specific weights for the opening at hand (Equation 1),

$$\boldsymbol{w}_{\mathcal{R}} = [w_{\mathcal{R}}^1, w_{\mathcal{R}}^2, ..., w_{\mathcal{R}}^{|\mathcal{D}|}]' \ , \tag{16}$$

and a vector with the competency-specific expertise estimates,

$$\boldsymbol{p}_{it} = [p_{it}^1, p_{it}^2, ..., p_{it}^{|\mathcal{D}|}]' \ . \tag{17}$$

The aggregated expertise for skillset $\mathcal{R}$ is then the inner product of the two:

$$P_{it}(\mathcal{R}) = \langle \boldsymbol{w}_{\mathcal{R}} \cdot \boldsymbol{p}_{it} \rangle \ . \tag{18}$$

5 IMPLEMENTATION DETAILS AND RESULTS

Next, we discuss the implementation details and the performance of the `HMM-W2V-framework`.

5.1 Features

Figure 6 shows how vectors $\mathbf{Z}_t$ and $\mathbf{X}_t$ affect the transitions and the emissions of the HMM respectively. What variables compose these two vectors?

Transition probabilities to new states are driven by the framework's current expertise estimate of any given contractor. Even

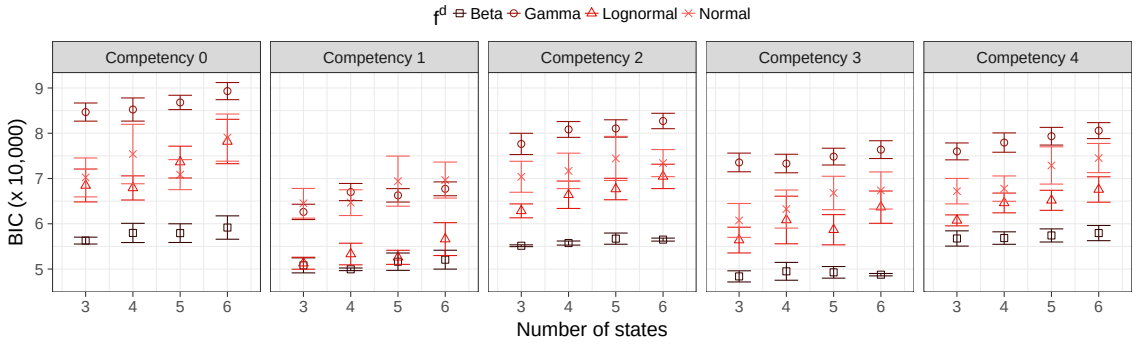


Figure 7: Number of states selection and choice of function f^d, for the five competency dimensions $d \in \mathcal{D}$. Error bars represent 95% confidence intervals.

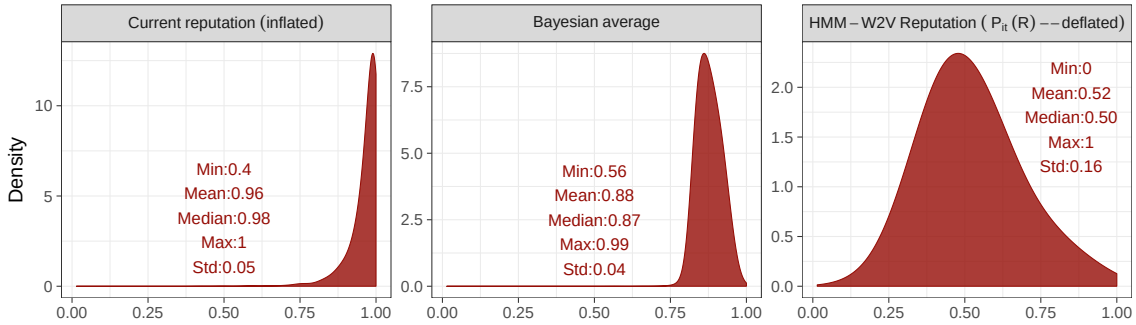


Figure 8: Reputation deflation: Distributions of the inflated current reputation (left), Bayesian average (middle), and HMM-W2V reputation (right).

Table 3: Regression analysis of the `HMM-W2V-framework` and the two baselines.

DV: Observed rating	(A1)	(A2)	(A3)	(A4)	(A5)	(A6)	(A7)	(A8)
Current reputation	-0.004	-0.097***					-0.017*	-0.108***
	(0.01)	(0.02)					(0.01)	(0.02)
Current reputation (squared)		0.121***						0.124***
		(0.03)						(0.03)
Bayesian average			0.047	-0.090			0.070	0.302
			(0.04)	(0.36)			(0.05)	(0.36)
Bayesian average (squared)				0.084				-0.169
				(0.22)				(0.22)
HMM-W2V reputation					0.175***	0.561***	0.177***	0.567***
					(0.02)	(0.11)	(0.02)	(0.11)
HMM-W2V reputation (squared)						-0.313***		-0.315***
						(0.08)		(0.08)
R^2	0.000	0.001	0.000	0.000	0.010	0.010	0.010	0.012

Robust standard errors in parentheses. The constant term is estimated but omitted from the table. ($^{***}p-$value < 0.001, $^{**}p-$value < 0.01, $^{*}p-$value < 0.05).

though actual expertise is latent, online labor markets provide observed signals that are correlated with expertise. Such signals form vector $\mathbf{Z}_t$, and include the accumulated feedback score, the total amount of received wages, the hiring rate of the contractor, and the total number of completed jobs. For each competency $d \in \mathcal{D}$, the `HMM-W2V-framework` customizes these signals by weighing them with $w^d_{\mathcal{R}}$. (Note that these signals are estimated at time $t-1$, i.e., at the completion of task $t-1$.)

Similarly, emission probabilities are affected by other, opening-specific characteristics that are not directly related to the latent expertise of the worker (feature vector $\mathbf{X}_t$). Such characteristics are whether or not the contractor has been invited to apply to the specific opening, the bidding price of the contractor, the number of times that a contractor has previously worked with the buyer at hand, and the focal employer's reputation.

Table 2 shows the descriptive statistics of the variables that form vectors $\boldsymbol{X}_t, \boldsymbol{Z}_t$, as well as the outcome variable Y_t.

5.2 Modeling choices and parameter estimation

For the purposes of this study, we pick the number of competency dimensions to be $|\mathcal{D}| = 5$. (In practice, engineers can experiment with various competency dimensions – however, to keep the analysis focused, we only present results for $|\mathcal{D}| = 5$.) We split the dataset into training (66%) and test sets (34%) based on contractors (i.e., the sequence of outcomes of each contractor can only be in either the test or the training set, but not in both). We use the training set to estimate the parameters of the framework and to choose appropriate functions. We then use the test set to evaluate the performance of the framework.

Function f^d (Equation 5) and the number of states (K^d) for each competency are fundamental choices for the implementation of the framework. To make the best possible choices, we estimate the configurations that yield the smallest Bayesian information criterion (BIC) scores in the training set [37]. In particular, the framework considers the following continuous probability distributions for modeling function f^d:

$$f^d \in \{\text{ Gaussian, Gamma, Beta, LogNormal}\}\ \forall d \in \mathcal{D}\ , \qquad (19)$$

and the following set of number of states:

$$K^d \in \{2, 3, ..., 6\}, \forall d \in \mathcal{D}\ . \qquad (20)$$

For each combination in $\{K^d \times f^d\}$, we estimate the the set of parameters $\boldsymbol{\Theta}^d, \boldsymbol{\Gamma}^d$ that maximize the likelihood of Equation 15. Because the optimization process depends on the initialization of $\boldsymbol{\Theta}^d, \boldsymbol{\Gamma}^d$, it is prone to stuck in local maxima. To increase the likelihood of reaching a potential global maximum, we conduct a search of 1,000 random initializations for each combination in $\{K^d \times f^d\}$. Finally, for each competency dimension $d \in \mathcal{D}$ we pick the configuration that yields the lowest BIC score.

Figure 7 shows the BIC scores for a series of configurations, for each one of the five competencies we consider. In terms of the emission function f^d, Beta seems to perform statistically significantly better than the other three choices, for competencies 0,2,3,4. In competency 1, Beta performs on par with the Lognormal distribution. In terms of number of states selection, none of the available choices is statistically significantly better than the rest (given f^d =Beta). The configurations that yielded the minimum BIC scores are:

- Competency-0: $K^0 = 6$ states, f^0 =Beta emissions.
- Competency-1: $K^1 = 3$ states, f^1 =Beta emissions.
- Competency-2: $K^2 = 4$ states, f^2 =Beta emissions.
- Competency-3: $K^3 = 6$ states, f^3 =Beta emissions.
- Competency-4: $K^4 = 4$ states, f^4 =Beta emissions.

After estimating the parameters and choosing the states k^d and functions f^d, the `HMM-W2V-framework` is ready to estimate the latent expertise of the contractors in the test set.

5.3 Results

Because contractors' levels of expertise are latent, the evaluation of the `HMM-W2V-framework` is not straightforward. Recall that the main goal of this study is to estimate scores that (1) are not inflated (like the current reputation scores), and (2) they capture the up-to-date expertise of contractors on any arbitrary set of skills. Testing whether the `HMM-W2V-framework` deflates reputation is straightforward through estimating and plotting the resulting distribution of $P_{it}(\mathcal{R})$. To evaluate whether the information of HMM-W2V reputation is more accurate than current reputation, we use as ground truth the observed outcomes of each task in the test set.

Baselines: Two baseline algorithms form a benchmark for the HMM-W2V reputation. The first one, is the current accumulated feedback score of `GigWork`. The second baseline is a Bayesian average of current reputation, which is frequently used for ranking items with star ratings [10, 34]. The Bayesian average for contractor i at time τ is as follows:

$$\text{Bayesian average:} BA_{i\tau} = \frac{h_{\text{tasks}} h_{\text{fb}} + \sum_{t=1}^{\tau-1} Y_{it}}{h_{\text{tasks}} + (\tau - 1)}\ , \qquad (21)$$

where h_{tasks} and h_{fb} are priors for the number of completed tasks per contractor and the contractor outcomes respectively. We set these priors to the mean values of the training set.

Reputation deflation: Figure 8 compares the distributions of the two baselines and the HMM-W2V reputation. Visually, the two baseline distributions are shifted to the right, while the focal HMM-W2V reputation distribution is almost centered at 0.5. While the two baselines take values in [0.4,1] (current reputation) and [0.56.0.99] (Bayesian average), HMM-W2V reputation deflates these scores to [0,1]. The mean of current reputation is 0.96, while Bayesian average brings this mean down to 0.88. However, the mean of HMM-W2V reputation is down to 0.52. Finally, while the two baselines have low variation (Standard deviation around 0.05), the deflated distribution clearly ranks contractors in the available space, with a standard deviation (0.16) 3.2 times higher than the baselines.

HMM-W2V reputation accuracy: The deflated distribution would have had very little value if the deflated HMM-W2V reputation was not a better (than the baselines) representation of the latent contractor expertise. Four metrics test this:

- Regression analysis: Does HMM-W2V reputation better explain the observed outcomes Y_{it} than the two baselines?
- Predictive analysis: Does HMM-W2V reputation better predict the observed outcomes Y_{it} than the two baselines?

Table 4: Predictive performance of the `HMM-W2V-framework` compared to the two baselines.

	Predictive analysis		Ranking correlations		Mutual Information
	MAE	MSE	Kendall τ	Spearman's rank	
Current reputation	0.17 ± 0.007	0.061 ± 0.001	0.07^{***}	0.05^{***}	0.006
Bayesian average	0.18 ± 0.007	0.061 ± 0.001	0.01^{**}	0.01^{**}	0.006
HMM-W2V reputation	0.14 ± 0.003	0.057 ± 0.001	0.12^{***}	0.09^{***}	0.016
Improvement over baselines	16%-17%	7%	71%-1100%	80% - 800%	166%

(95% Confidence intervals for MAE and MSE. $^{***}p-$value < 0.001, $^{**}p-$value < 0.01.)

- Ranking correlation: Does HMM-W2V reputation rank contractors better than the two baselines?
- Mutual information: Does HMM-W2V reputation and the observed outcomes Y_{it} yield higher mutual information than the two baselines?

For the regression analysis we estimate the following specification:

$$Y_{it} = c_1 + \boldsymbol{c_2}\text{poly}(P_{it}(\mathcal{R}), 2) + \boldsymbol{c_3}\text{poly}(FB_{it}, 2) + \boldsymbol{c_4}\text{poly}(BA_{it}, 2)\ , \quad (22)$$

where poly(x,2) estimates the first and second order of variable x. Table 3 shows the results. These models use the complete test set to estimate the necessary parameters. Columns (A1,A2) show models that consider only current reputation scores (FB_{it}). Columns (A3,A4) show models that consider only Bayesian average scores (BA_{it}). Columns (A5,A6) show models that consider only HMM-W2V reputation scores ($P_{it}(\mathcal{R})$). The comparison shows that HMM-W2V reputation has higher explanatory power than the two baselines, as it yields more than 10 times greater R^2, while coefficients $\boldsymbol{c_2}$ are consistently statistically significant ($p < 0.001$). Models (A7,A8) show how the six variables behave when we estimate the complete specification of Equation 22. Again, the coefficients of HMM-W2V reputation remain large and statistically significant ($p < 0.001$).

Models A2, A4 and A6 in Table 3 test the predictive performance of the three reputation scores. Table 4 shows the 10-fold cross validation Mean Absolute Error (MAE) and Mean Squared Error (MSE) for the three models. The last row shows the percentage improvement of the HMM-W2V reputation over the two baselines. For both metrics HMM-W2V reputation outperforms the two baselines, showing an improvement (decrease in errors) between 7% and 17%.

A correct expertise assessment should rank contractors according to their likelihood of performing good in any given opening with required skills $\mathcal{R}$. Two ranking correlation coefficients (Kendall τ [22], Spearman's rank [42]) test the ordinal associations between expertise estimates and observed outcomes. Table 4 shows the clear superiority of HMM-W2V reputation over the two baselines.

Finally, mutual information measures the dependence between two variables [11]. Observed outcomes should be dependent to an accurate expertise estimate. Table 4 shows that observed outcomes are 166% more dependent to HMM-W2V reputation compared to the two baselines.

Overall, the empirical analysis in this section shows a clear superiority of the HMM-W2V reputation over the two baselines, both in terms of deflating reputation scores and estimating the latent contractor expertise (accuracy). This promising performance of the `HMM-W2V-framework` suggests important implications for online (labor) markets and their users. We discuss these next.

6 IMPLICATIONS

Deflated reputation scores help (1) contractors to differentiate, (2) buyers to make better-informed and faster (reduced search cost [5]) decisions, and (3) the market to build more accurate recommendation algorithms but also to better understand the supply distributions across latent competencies. When contractors can be accurately differentiated, the quality of the supply side of the market naturally increases. High-quality contractors are more likely to keep participating, when low-quality contractors could potentially give space to newcomers. When buyers make better-informed (and faster) decisions that lead to better outcomes become more likely to keep participating in the market. At the same time, markets can improve the performance of their recommendation algorithms by using HMM-W2V reputation as an additional feature. Better recommendations imply higher income and higher transaction efficacy [24, 30]. Finally, through studying the resulting HMM-W2V reputation, markets can get a better picture of the distribution of skills and abilities, and intervene where they think appropriate (e.g., through targeted advertising on skills that there is a deficit).

Besides the direct implications to online labor markets, the `HMM-W2V-framework` can be adjusted and implemented in any online market that has similar rating distributions for products or services. Amazon for instance could deploy the proposed framework to deflate product ratings. In this scenario, the dimensions could be latent product characteristics, and the required skills could be the preferences of the buyer across these latent product characteristics. Similarly, reputation platforms such as Yelp and TripAdvisor could use this framework, as they could decompose venue features to latent dimensions and ask for user preferences across these dimensions.

In conclusion, we presented a framework that deflates reputation in online labor markets through solving reputation attribution and reputation staticity. Analysis of real hiring decisions from an online labor market showed a clean superiority of the proposed framework over current reputation and a Bayesian average baseline.

ACKNOWLEDGMENTS

The author thanks professors Robert Fichman, Panagiotis G. Ipeirotis, Konstantinos Pelechrinis and Sam Ransbotham for their constructive comments and guidance.

REFERENCES

[1] Panagiotis Adamopoulos. 2013. What makes a great MOOC? An interdisciplinary analysis of student retention in online courses. In *International Conference on Information Systems*. AIS.

[2] Panagiotis Adamopoulos, Anindya Ghose, and Vilma Todri. 2018. The Impact of User Personality Traits on Word of Mouth: Text-Mining Social Media Platforms. *Information Systems Research* 29, 3 (2018), 612–640.

[3] Panagiotis Adamopoulos and Alexander Tuzhilin. 2015. The business value of recommendations: A privacy-preserving econometric analysis. In *International Conference on Information Systems*. AIS.

[4] George A Akerlof. 1970. The market for "lemons": Quality uncertainty and the market mechanism. *The Quarterly Journal of Economics* (1970), 488–500.

[5] J Yannis Bakos. 1997. Reducing buyer search costs: Implications for electronic marketplaces. *Management Science* 43, 12 (1997), 1676–1692.

[6] Krisztian Balog and Maarten De Rijke. 2007. Determining expert profiles (with an application to expert finding). In *Proceedings of the 20th International Joint Conference on Artifical intelligence*. Morgan Kaufmann Publishers Inc., 2657–2662.

[7] Rajiv D Banker and Iny Hwang. 2008. Importance of measures of past performance: Empirical evidence on quality of e-service providers. *Contemporary Accounting Research* 25, 2 (2008), 307–337.

[8] Gary Bolton, Ben Greiner, and Axel Ockenfels. 2013. Engineering trust: reciprocity in the production of reputation information. *Management Science* 59, 2 (2013), 265–285.

[9] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. 1995. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing* 16, 5 (1995), 1190–1208.

[10] Mung Chiang. 2012. *Networked Life: 20 Questions and Answers*. Cambridge University Press.

[11] Thomas M Cover and Joy A Thomas. 2012. *Elements of information theory*. John Wiley & Sons.

[12] Chrysanthos Dellarocas. 2003. The digitization of word of mouth: Promise and challenges of online feedback mechanisms. *Management Science* 49, 10 (2003), 1407–1424.

[13] Chrysanthos Dellarocas, Guodong Gao, and Ritu Narayan. 2010. Are Consumers More Likely to Contribute Online Reviews for Hit or Niche Products? *Journal of Management Information Systems* 27, 2 (Oct. 2010), 127–158.

[14] Apostolos Filippas, John J. Horton, and Joseph Golden. 2018. Reputation Inflation. *Working paper* (2018).

[15] Anindya Ghose and Vilma Todri. 2016. Towards a digital attribution model: Measuring the impact of display advertising on online consumer behavior. *MIS Quarterly* 40, 4 (2016), 889–910.

[16] R.K. Hambleton, H. Swaminathan, and H.J. Rogers. 1991. *Fundamentals of Item Response Theory*. SAGE Publications. http://books.google.com/books?id=cmJU9SHCzecC

[17] Geoffrey E Hinton and Sam T Roweis. 2003. Stochastic neighbor embedding. In *Advances in neural information processing systems*. 857–864.

[18] Nan Hu, Paul A Pavlou, and Jie Zhang. 2017. On self-selection biases in online product reviews. *MIS Quarterly* 41, 2 (2017), 449–471.

[19] N. Hu, J. Zhang, and P.A. Pavlou. 2009. Overcoming the J-shaped distribution of product reviews. *Commun. ACM* 52, 10 (2009), 144–147.

[20] Kinshuk Jerath, Peter S Fader, and Bruce GS Hardie. 2011. New perspectives on customer "death" using a generalization of the Pareto/NBD model. *Marketing Science* 30, 5 (2011), 866–880.

[21] Pawel Jurczyk and Eugene Agichtein. 2007. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the sixteenth ACM Conference on Information and Knowledge Management*. ACM, 919–922.

[22] Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika* 30, 1/2 (1938), 81–93.

[23] Marios Kokkodis. 2012. Learning from positive and unlabeled Amazon reviews: Towards identifying trustworthy reviewers. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*. ACM, 545–546.

[24] Marios Kokkodis. 2018. Dynamic Recommendations for Sequential Hiring Decisions in Online Labor Markets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 453–461.

[25] Marios Kokkodis and Panagiotis G. Ipeirotis. 2013. Have you done anything like that?: predicting performance using inter-category reputation. In *Proceedings of the sixth ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 435–444.

[26] Marios Kokkodis and Panagiotis G. Ipeirotis. 2014. The utility of skills in online labor markets. *Proceedings of the AIS International Conference on Information Systems* (2014).

[27] Marios Kokkodis and Panagiotis G. Ipeirotis. 2016. Reputation transferability in online labor markets. *Management Science* 62, 6 (2016), 1687–1706.

[28] Marios Kokkodis and Theodoros Lappas. 2016. Realizing the Activation Potential of Online Communities. In *International Conference on Information Systems*.

[29] Marios Kokkodis and Theodoros Lappas. 2016. The Relationship Between Disclosing Purchase Information and Reputation Systems in Electronic Markets. In *International Conference on Information Systems*.

[30] Marios Kokkodis, Panagiotis Papadimitriou, and Panagiotis G. Ipeirotis. 2015. Hiring behavior models for online labor markets. In *Proceedings of the eighth ACM International Conference on Web Search and Data Mining (WSDM)*. ACM.

[31] Q Le and T Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*. 1188–1196.

[32] Mingfeng Lin, Yong Liu, and Siva Viswanathan. 2018. Effectiveness of Reputation in Contracting for Customized Production: Evidence from Online Labor Markets. *Management Science* 64, 1 (2018), 345–359.

[33] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[34] Evan Miller. 2014. Ranking items with star ratings. http://www.evanmiller.org/ranking-items-with-star-ratings.html. Accessed: 2018-10-24.

[35] Wendy W Moe and David A Schweidel. 2012. Online product opinions: Incidence, evaluation, and evolution. *Marketing Science* 31, 3 (2012), 372–386.

[36] Antonio Moreno and Christian Terwiesch. 2014. Doing business with strangers: Reputation in online service marketplaces. *Information Systems Research* 25, 4 (2014), 865–886.

[37] Kevin P Murphy. 2012. *Machine learning: a probabilistic perspective*. The MIT Press.

[38] Ernest O'Boyle Jr and Herman Aguinis. 2012. The best and the rest: Revisiting the norm of normality of individual performance. *Personnel Psychology* 65, 1 (2012), 79–119.

[39] Amanda Pallais. 2014. Inefficient hiring in entry-level labor markets. *The American Economic Review* 104, 11 (2014), 3565–3599.

[40] Konstantinos Pelechrinis, Vladimir Zadorozhny, Velin Kounev, Vladimir Oleshchuk, Mohd Anwar, and Yiling Lin. 2015. Automatic evaluation of information provider reliability and expertise. *World Wide Web* 18, 1 (2015), 33–72.

[41] PeoplePerHour. 2018. Hire expert freelancers. https://www.peopleperhour.com/hire-freelancers. Accessed: 2018-10-22.

[42] Charles Spearman. 1904. The proof and measurement of association between two things. *The American journal of psychology* 15, 1 (1904), 72–101.

[43] Statista. 2018. Quarterly share of e-commerce sales of total U.S. retail sales from 1st quarter 2010 to 2nd quarter 2018. https://www.statista.com/statistics/187439/share-of-e-commerce-sales-in-total-us-retail-sales-in-2010/. Accessed: 2018-10-09.

[44] Vilma Todri and Panagiotis Adamopoulos. 2014. Social commerce: An empirical examination of the antecedents and consequences of commerce in social network platforms. In *International Conference on Information Systems*.

[45] Hema Yoganarasimhan. 2013. The value of reputation in an online freelance marketplace. *Marketing Science* 32, 6 (2013), 860–891.

[46] Jun Zhang, Mark S Ackerman, and Lada Adamic. 2007. Expertise networks in online communities: structure and algorithms. In *Proceedings of the 16th International Conference on World Wide Web*. ACM, 221–230.

Learning from On-Line User Feedback in Neural Question Answering on the Web

Bernhard Kratzwald
ETH Zurich
Zurich, Switzerland
bkratzwald@ethz.ch

Stefan Feuerriegel
ETH Zurich
Zurich, Switzerland
sfeuerriegel@ethz.ch

ABSTRACT

Question answering promises a means of efficiently searching web-based content repositories such as Wikipedia. However, the systems of this type most prevalent today merely conduct their learning once in an offline training phase while, afterwards, all parameters remain static. Thus, the possibility of improvement over time is precluded. As a consequence of this shortcoming, question answering is not currently taking advantage of the wealth of feedback mechanisms that have become prominent on the web (e. g., buttons for liking, voting, or sharing).

This is the first work that introduces a question-answering system for (web-based) content repositories with an on-line mechanism for user feedback. Our efforts have resulted in QApedia – a framework for on-line improvement based on shallow user feedback. In detail, we develop a simple feedback mechanism that allows users to express whether a question was answered satisfactorily or whether a different answer is needed. Even for this simple mechanism, the implementation represents a daunting undertaking due to the complex, multi-staged operations that underlie state-of-the-art systems for neural questions answering. Another challenge with regard to web-based use is that feedback is limited (and possibly even noisy), as the true labels remain unknown. We thus address these challenges through a novel combination of neural question answering and a dynamic process based on distant supervision, asynchronous updates, and an automatic validation of feedback credibility in order to mine high-quality training samples from the web for the purpose of achieving continuous improvement over time.

Our QApedia framework is the first question-answering system that continuously refines its capabilities by improving its now dynamic content repository and thus the underlying answer extraction. QApedia not only achieves state-of-the-art results over several benchmarking datasets, but we further show that it successfully manages to learn from shallow user feedback, even when the feedback is noisy or adversarial. Altogether, our extensive experimental evaluation, with more than 2,500 hours of computational experiments, demonstrates that a feedback mechanism as simple as a binary vote (which is widespread on the web) can considerably improve performance when combined with an efficient framework for continuous learning.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313661

CCS CONCEPTS

• **Information systems** → **Question answering**; • **Computing methodologies** → *Lifelong machine learning*; *Online learning settings.*

KEYWORDS

Question answering; User feedback; Distant Supervision; On-line learning

ACM Reference Format:
Bernhard Kratzwald and Stefan Feuerriegel. 2019. Learning from On-Line User Feedback in Neural Question Answering on the Web. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313661

1 INTRODUCTION

Question-answering (QA) systems promise an efficient way to search content repositories, as both questions and answers are presented in natural language [38]. Use cases can include the vast majority of freely-available materials that have been developed on the web. Hence, question answering has been developed for traditional knowledge bases and knowledge graphs [e. g., 1, 2, 14, 41, 45] but also for unstructured open-domain content such as Wikipedia [3, 8, 11] and social media [7], as well as various domain-specific applications including Stack Overflow [4].

Content-based question answering widely assumes that answer extraction mechanisms are static; see the systems in [8, 11] as illustrative examples. This is in contrast to QA for knowledge bases, where a key objective involves building up the knowledge base itself [12, 16], as well as continuously updating the underlying decision rules for answering [2]. However, in the context of QA for linguistic content, such dynamic mechanisms are, to the best of our knowledge, non-existent.

This paper suggests a framework for content-based QA in order to bolster QA systems with dynamic capabilities: our framework aids continuous learning on the basis of a simple mechanism for user feedback. In fact, user feedback plays a prominent role in web-based systems, as a large proportion of websites now offer tools for social interaction. Examples include buttons for liking or sharing content (e. g., as on Instagram), as well as voting with an array of categorical emotions (e. g., as on Facebook) [25]. There have been previous attempts to utilize user interactions in classical information retrieval [5, 27] or knowledge bases [1], and yet there is a dearth of content-based QA systems that leverage user interactions.

Extending content-based QA with user feedback promises several benefits. When interacting with a QA system, the correct answer is unknown to a user, and yet it is fairly easy to judge whether

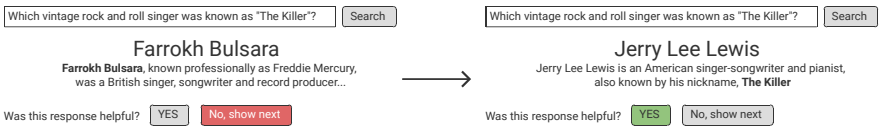


Figure 1: Illustrative sketch of shallow user feedback in question-answering for natural language. A simple credibility check is often sufficient in order to judge whether an answer makes sense in the given context.

an answer makes sense in the given context (see Fig. 1). In addition, shallow feedback in the form of a binary vote can be collected at low cost. It is especially low-cost in comparison to reverting to a human annotator in order to retrieve the correct label. Finally, the prevalence of feedback mechanisms on the web ensures that such user interactions have become widely intuitive.

Contributions: This work proposes QApedia: a neural question-answering framework for encyclopedic content that continuously improves on the basis of on-line user feedback. To the best of our knowledge, QApedia represents the first content-based QA system that improves over time. Our feedback framework advances the status quo of static QA systems while being specifically tailored to web-based settings:

(1) **Feedback for dynamic knowledge.** Content on the web is subject to considerable time variability and, hence, a QA system must adapt to this dynamic nature. To facilitate this, we develop an effective feedback mechanism so that the abilities of the QA system can successfully continue to improve over time. Our framework directly incorporates user feedback in an end-to-end loop: collected feedback is fed back into the system in an on-line fashion. As a key challenge during implementation, we must overcome the problem of catastrophic forgetting that is known in neural networks and thus also neural QA. For this reason, we develop a tailored form of distant supervision with asynchronous updates.

(2) **Shallow feedback.** We only require shallow user feedback in the form of a simple up or down vote, which is nowadays common on the web. We specifically refrain from asking users to report the exact answers, as users might not know these answers or else be reluctant to report them; instead, it is sufficient for our framework to receive a simple credibility check. Receiving only limited feedback – and not necessarily the correct solution – requires a specialized adaptation of distant supervision to our setting.

(3) **Noisy and adversarial feedback.** User feedback in web-based settings is often noisy or even adversarial. Our framework must therefore be designed so that, despite errors in user feedback, it maintains its performance (or even continues to improve) and is thus especially robust. This is achieved by incorporating a validation procedure, based on knowledge mining, during which the credibility of user feedback is checked.

Our findings demonstrate that our QApedia framework successfully manages to learn from on-line feedback. It not only adapts to the feedback provided in the on-line setting, but it also maintains the abilities it has acquired through previous training, thus overcoming the issue of catastrophic forgetting. Our results yield a considerable improvement: the user feedback ensures that the performance over time no longer remains static but, even when evaluating the QA system with question-answer pairs from a different domain, the number of correct answers continues to increase over time on the order of 10–20 percentage points. For instance, in one dataset, fewer than 60,000 user interactions with shallow feedback were sufficient to double the percentage of exact answers. These performance improvements are even maintained in the case of noisy and adversarial feedback. Furthermore, catastrophic forgetting in a naïve QA system decreases the ratio of exact answers by 5 percentage points, whereas our QApedia framework largely maintains the original performance.

2 RELATED WORK

Question Answering: Question answering can be divided into two main paradigms, namely systems that operate in relation to structured knowledge and those that rely upon (primarily unstructured) textual content (or both, as in [15]).

QA systems for structured knowledge [e. g., 1, 2, 14, 41, 45] derive answers from knowledge bases, ontologies, or knowledge graphs. Structured knowledge bases augment web search and sometimes even serve as substitutes; see, for instance, Wikidata or Google Knowledge Graph. Explicit structures entail the benefit of simplifying the process of answer extraction, yet they are incomplete and limited to rigid (and often pre-defined) schemata and, therefore, lack the same flexibility as running text.

QA systems for content in natural language [e. g., 9, 11, 16, 31, 42] overcome some of the drawbacks of raw knowledge bases, as they extract answers directly from an underlying corpus of unstructured text documents. Hence, they find widespread application in mining web-based content such as Wikipedia or other online encyclopedias [3, 8, 11]. The content-based approach greatly contributes to overall flexibility, especially when such systems leverage the growing body of knowledge in web-based content repositories. Hence, QA systems for (web-based) content repositories constitute the focus of this work. Yet prior systems for question answering for content repositories have been designed as static systems: all decision rules are determined once and are static thereafter, thus curbing any form of continuous improvement.

Neural Question Answering for Content: Content-based QA systems commonly proceed through multiple phases [24]: they first select a subset of documents (or paragraphs) that are considered relevant and then extract the final answer. Answer extraction has traditionally been based on linguistic rules or pattern matching [13, 20, 31, 37], whereas deep neural networks [11, 17, 35, 42] have evolved only recently as the state-of-the-art. This is later reflected by our implementation, in which we combine several of the recent advances from neural question answering, such as attention [40, 43] and adaptive retrieval [23], in order to obtain a reference system that represents the status quo.

Learning from User Feedback: Improving (i. e., learning) from user feedback is a well-studied problem in, for instance, information retrieval and web-based search [5, 27]. Applications in these disciplines include user feedback that is leveraged in order to improve conversational agents [36], assess the relevance of answers in search queries for knowledge graphs [39], or identify incorrect queries to databases [18]. While [34] makes use of shallow binary feedback for machine translation, it does not account for noisy or adversarial feedback.

Closest to our research is the seminal work by Abujabal et al. [1] that distinguishes the relevance of a set of potential answers based on user feedback in the context of question answering for a structured knowledge base. Yet the latter work addresses a different objective and, in contrast to our research, thus entails marked differences. First, the aforementioned feedback mechanism from [1] is tailored to knowledge bases, while we contribute to the specific methodological foundation of content-based question answering. Second, the authors' work integrates user feedback into ontology-based representations, while we extract answers from natural language based on deep neural networks. Third, the methodology in [1] utilizes user feedback at strategic intervention points (i. e., when decision rules are unknown), whereas we develop a framework in which on-line feedback is directly integrated end-to-end. That is, user feedback is collected continuously across potentially all user interactions and then directly incorporated at the most effective point of the QA system such that the overall performance of the QA task improves. Furthermore, our work presents – to the best of our knowledge – the first attempt in the field of question answering that adapts to the particular characteristics of user feedback on the web, namely the possibility of errors and even malicious feedback; therefore, we evaluate our proposed framework with user feedback that is noisy or even adversarial.

Never-Ending Learning: In never-ending learning [10, 30], models continue to learn new tasks and/or improve upon existing tasks over time, even after their initial estimation. Never-ending learning represents an especially challenging undertaking in combination with neural networks, as neural networks are inherently prone to so-called catastrophic forgetting [22]. Catastrophic forgetting describes the phenomenon whereby models forget what they have learned earlier; i. e., while learning new tasks, the performance on tasks that have already been trained decreases dramatically.

In prior literature, applications of never-ending learning to question answering are scarce. To the best of our knowledge, there is only one such approach, the NEQA system [1]. This system is designed to address the specific needs of knowledge bases and is thus not applicable to content-based QA as in our research. More precisely, the NEQA system accumulates new rules on the basis of which it extract answers from a knowledge base. In contrast, we pursue a different objective as we develop a data-driven approach for updating the deep neural network that is nowadays common in content-based QA: both training data and model parameters are treated as dynamic components. For this purpose, we need to derive a specific framework involving distant supervision combined with asynchronous updating.

Distant Supervision: Distant supervision (i. e., enriching weakly labeled data with labels that are potentially noisy) has been previously used for neural question answering, but for purposes other than aiding never-ending learning. Previous use cases include, for example, locating correct answers in text passages [28] or for data augmentation [11, 19]. In this work, we contribute two-fold: (1) in contrast to prior works, we apply distant supervision to on-line feedback mechanisms for question answering in relation to content repositories. (2) We suggest a nascent setting in which on-line user feedback can be subject to errors, thus necessitating a novel feedback loop whereby user feedback is validated against the current content repository as a prerequisite for performing distant supervision.

3 QApedia: CONTINUOUS IMPROVEMENT FROM USER FEEDBACK

The architecture of our proposed QApedia framework (see Fig. 2) consists of three modules: a *content base* stores the knowledge for answering questions. Examples include, e. g., Wikipedia or other content repositories in natural language. This module also manages the datasets that are used for learning the question-answering capabilities. The *question-answering module* provides the actual system that takes a question as input and then suggests an answer. Its implementation adheres to the state-of-the-art of neural QA as detailed in Section 4. The module for *continuous improvement* represents our core contribution and ensures that dynamic learning takes place. It thus receives feedback through user interactions and, on this basis, updates the content base and, subsequently, the neural network inside the question-answering module.

3.1 Feedback Mechanism through User Interactions

3.1.1 Motivation Behind Shallow User Feedback. The QApedia framework achieves its dynamic learning based on user interactions that are simple in nature: a user is merely asked to assess the overall fit of an answer in the given context and should then respond with a simple up- or down-vote. This binary feedback system was specifically chosen as it promises inherent benefits. First, shallow user feedback can be collected at low cost. Accordingly, it does not require expert users or rigorous fact-checking through which the correct answer is determined. Instead, we strive for a simple integration whereby the context of an answer is assessed, and we thus expect that user feedback might include a small portion of erroneous assessments. Similar feedback mechanisms have become prevalent in web-based use cases [33, 34], e. g., when users are asked to classify whether a translation was helpful. Hence, users should be familiar with this arrangement, which ensures straightforward use.

Our design, based on shallow feedback, also comes with limitations: if the QA system comes up with the wrong answer, it will likely receive a negative vote. However, it does not necessarily obtain the correct answer, which can be used as a label. This poses challenges – beyond those that accompany web-based use – that we must carefully address in our design.

Another challenge arises from the web-based nature of the user inputs: there is no guarantee that user feedback is correct. Some users might be better at judging the suitability of an answer in a given context than others. Some users might, however, just click a random feedback option or even test the system's capability by intentionally responding with an erroneous answer. Hence, the

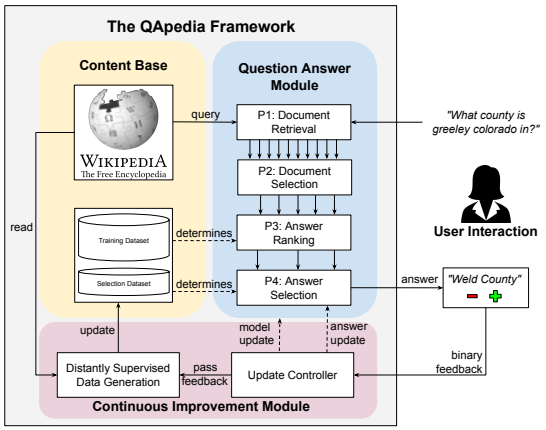


Figure 2: Architecture of our QApedia framework.

system must adapt to noisy (i. e., partially random) and adversarial (i. e., explicitly wrong) feedback.

3.1.2 Feedback Collection. The QA system takes a question q as input and then returns a triplet $\langle q, a, p \rangle$, where a refers to the answer and p to the surrounding paragraph that can be used for assessing the fit (see the example presentation in Fig. 1). Let ψ denote the (optional) feedback, i. e., an up-vote or a down-vote. Depending on the feedback, the system proceeds as follows:

- *Positive feedback* is directly passed to the module for continuous improvement. The module first performs a credibility check to validate the answer a with an up-vote against the content base. It then generates a new data sample $\langle q, a, p \rangle$ (or, more precisely, $\langle q, a, p^* \rangle$ with an improved p^* through distant supervision as described in Section 3.3) and, based on this, updates the datasets in the content base. The updated training sets are then used at regular intervals for re-estimating the neural network inside the QA module.
- *Negative feedback* triggers a process in which the current answer is rephrased and an alternative answer a' is displayed to the user. For this purpose, the QA module is queried in order to generate the next-best but different answer. We achieve this by choosing the next answer in the ranked list and ensuring that it is not a duplicate of the original answer. Eliminating duplicates is necessary, since the candidate answers during answer ranking are often identical. Users can repeatedly provide negative feedback, where ρ denotes the maximum frequency that we use in our experimental evaluation, whereas such a limit would not be enforced in practice.

The user feedback is directly fed into the content base through distant supervision and thus follows a human-in-the-loop process.

3.2 Content Base

Our QApedia framework performs question answering for content repositories. As part of our experiments, we later adhere to earlier research [3, 8, 11] and utilize Wikipedia as a widespread baseline. Without loss of generality, our framework can also be used in relation to all other knowledge bases that contain unstructured materials in (English) natural language.

Our model estimation further relies upon two datasets with labeled questions, answers, and paragraphs. These datasets are time-dependent and thus dynamic in order to enable continuous learning. Then, the training dataset $\mathcal{D}^t_{\text{train}}$ at time t is needed for estimating the parameters inside the question-answering module. A second dataset $\mathcal{D}^t_{\text{selection}}$ at time t is used for model selection. Both datasets contain triplets $\langle q, a, p \rangle$ with a question q, an answer a, and the corresponding paragraph p (i. e., in which the answer a is located).

It is important to point out that the dynamic, time-dependent nature of our datasets $\mathcal{D}^t_{\text{train}}$ and $\mathcal{D}^t_{\text{selection}}$ is integral for incorporating user feedback at runtime. This allows us to include user feedback in the form of new data samples. Finally, this time dependency is in stark contrast to prior literature on content-based QA [e. g., 11, 42], which builds upon corpora that are static. In such examples, the QA models are trained once during an off-line phase, whereas we implement the datasets for training and model selection as dynamic components updated through on-line feedback.

3.3 Continuous Improvement Module

The continuous improvement module is responsible for three tasks. First, user feedback is not directly fed into the content base but subject to a customized process with *distant supervision*. This entails an internal credibility check of the user feedback against the content base in order to deal with feedback that is noisy or adversarial. Second, the credibility check produces aggregated evidence with high-quality data samples that is eventually integrated into the content base. Third, the actual updates of the QA module can be triggered at arbitrary time intervals, for which we draw upon *asynchronous model updates*. As part of this, all trainable parameters inside the QA module are re-estimated based on the current state of the content base.

Prior work on distant supervision in question answering [11, 19] assumes that the provided answer to a question is always correct, whereas a web-based setting does not allow for this assumption and we thus adapt to incorrect feedback via credibility validation.

3.3.1 Credibility Validation. QApedia improves under noisy and adversarial feedback thanks to a careful design: we do not directly integrate triples $\langle q, a, p \rangle$ with a positive up-vote into the training data but instead rely upon a novel validation procedure with distant supervision, which filters erroneous or malicious feedback. As a credibility check, we mine so-called evidence $\mathcal{E} = \{p_1, \ldots, p_n\}$ from the content base that supports the user feedback. Only if this evidence surpasses a certain threshold do we treat the user feedback as credible.

Evidence is mined as follows. We utilize the QA system – more precisely, the mechanisms for document retrieval and document selection – in order to retrieve text passages $\mathcal{E}$ from the content base that are likely to contain the answer to question q. The resulting evidence $\mathcal{E}$ is further processed by a set of filtering rules, so that we obtain only a subset $\mathcal{E}_{\text{filtered}} \subseteq \mathcal{E}$. Without difficulty, we can reject all passages that fail to contain the assumed answer a; i. e., $a \notin p$. These passages do not provide any strong evidence for a being an answer of q. Furthermore, we filter passages that are very short, since we noticed that these commonly contain summarizing statements or headlines but are rarely extensive enough to contain

Algorithm 1: Updating procedure for continuous improvement of QA systems from shallow user feedback.

Input: User feedback ψ for question-answer-paragraph triple $\langle q, a, p \rangle$, where $\psi \in \{0, \uparrow, \downarrow\}$ denotes either an up-vote, down-vote, or no vote

$\mathcal{E}_{\text{filtered}} \leftarrow \emptyset$
if $\psi = \downarrow$ **then**
  Trigger answer selection to refine the displayed answer
else if $\psi = \uparrow$ **then**
  Initialize evidence $\mathcal{E} \leftarrow$ document_selection(q) by querying for q
  for $p \in \mathcal{E}$ **do**
    if $a \in p$ **and** $len(p) > min_len$ **and** $NE(q) \subseteq NE(p)$ **then**
      Score the evidence $s \leftarrow score(p, q, a)$ based on a sliding window
      if $s \geq 2$ **then**
        $\mathcal{E}_{\text{filtered}} \leftarrow \mathcal{E}_{\text{filtered}} \cup \{\langle q, s \rangle\}$;
      end
    end
  end
  if $|\mathcal{E}_{\text{filtered}}| >$ threshold τ **then**
    Determine best paragraph $p^* \leftarrow \max \mathcal{E}_{\text{filtered}}$ in the evidence (with the score from document retrieval as a tie breaker)
    Determine randomly which corpus $c \leftarrow corp(q, a)$ to update
    if $c =$ train **then**
      $\mathcal{D}^{t+1}_{\text{train}} = \mathcal{D}^{t}_{\text{train}} \cup \{\langle q, a, p^* \rangle\}$;
    else
      $\mathcal{D}^{t+1}_{\text{selection}} = \mathcal{D}^{t}_{\text{selection}} \cup \{\langle q, a, p^* \rangle\}$;
    end
  end
end

the answer. Accordingly, we check that the paragraph length $len(p)$ exceeds a minimum length (here we use min_len = 25). Following [11, 19], we further filter paragraphs that do not contain all of the named entities that were found in the question.[1] Let $NE(p)$ refer to the named entities – we then ensure that $NE(q) \subseteq NE(p)$. Finally, we score the quality of each of the remaining passages by counting the bi-gram overlap between the question and a 15-word window around the found answer in the passage. We reject all passages where the score is below a threshold. In our experiments, we follow [11] and enforce at least two overlaps of bi-grams. All remaining passages $\mathcal{E}_{\text{filtered}}$ are considered as strong evidence for the correctness of the feedback.

The decision of whether or not an item of feedback is credible is made based on the number of remaining text passages, i. e., $|\mathcal{E}_{\text{filtered}}| \geq \tau$. The hyperparameter τ controls the trade-off between a fast improvement (i. e., more samples will be generated) on the one hand, and a more restrictive filtering of noise on the other hand (i. e., more feedback will be rejected). We demonstrate the effects of rigorous filtering in our experiments on noisy and adversarial users.

3.3.2 Data Generation via Distant Supervision. If the user feedback is found to be credible, we do not add $\langle q, a, p \rangle$ directly into our content base; instead, we generate a new data sample $\langle q, a, p^* \rangle$ with replaced paragraph p^* from the evidence. The aim behind this is to find a high-quality paragraph p^* which contains more information than the original paragraph p that received the up-vote. To achieve this, we make direct use of the aforementioned credibility validation and the evidence set $\mathcal{E}_{\text{filtered}}$ accumulated in that phase. In detail, we use the highest scoring text passage $p^* \in \max_p \mathcal{E}_{\text{filtered}}$, i. e., we take the paragraph with the highest bi-gram overlap. If there are multiple passages with the same overlap, we use the retrieval score as a tiebreaker. This results in a new data sample $\langle q, a, p^* \rangle$, which is added to the dataset either for training or model selection, i. e.,

$$\mathcal{D}^{t+1}_{\text{train}} \leftarrow \mathcal{D}^{t}_{\text{train}} \cup \{\langle q, a, p^* \rangle\} \quad \text{or} \tag{1}$$

$$\mathcal{D}^{t+1}_{\text{selection}} \leftarrow \mathcal{D}^{t}_{\text{selection}} \cup \{\langle q, a, p^* \rangle\}. \tag{2}$$

The choice is made randomly, while maintaining the original ratio of the splits between training and model selection set (i. e., 90 % vs. 10 %). Furthermore, we assert that no question is in both datasets at the same time and that every dataset contains no duplicate question-answer pairs.

3.3.3 Asynchronous Model Updates. At desired time steps t, the continuous improvement module invokes an asynchronous model update. Here the idea is to use the current training sets from the content base, i. e., $\mathcal{D}^{t}_{\text{train}}$ and $\mathcal{D}^{t}_{\text{selection}}$ at time t in order to re-estimate the neural QA system. We use the above training data $\mathcal{D}^{t}_{\text{train}}$ with mini-batch gradient descent for 45 epochs. After each epoch, we evaluate the performance on the selection dataset $\mathcal{D}^{t}_{\text{selection}}$. Finally, we select the model with the highest-scoring performance on the selection dataset and use it to replace our current model.

The combination of distant supervision and data-driven learning of model parameters proves to be very robust against noisy or even adversarial feedback, as it provides two lines of defense. The first line of defense is our method for generating new data samples, since it filters most of the noise before it can enter the system. The second line of defense is the update mechanism itself. Its data-driven nature tolerates a small fraction of noise by definition, since noisy gradients are averaged out as long as the corpus contains enough correct samples. This is in strong contrast to a rule-based system, where even one wrong rule is guaranteed to generate wrong answers.

4 THE QUESTION-ANSWERING MODULE

The question-answering module within our framework for learning from on-line feedback provides the functional core that is responsible for extracting the displayed answers from the web-based content repository. Its structure is designed such that it combines recent advances in the field of neural question answering that are regarded as state-of-the-art in order to yield a reference implementation. Specifically, it builds upon attention mechanisms [40, 43], adaptive retrieval [23], and neural answer extraction at paragraph level [11, 42], concepts that have been proposed only recently. Consistent with the literature, the question-answering module operates along four phases (see Fig. 2), which we summarize briefly in the following: In **(P1) Document Retrieval**, all documents from the content repository are scored based on their relevance to the question. In **(P2) Document Selection**, after ranking all documents by their relevance, only a small subset of documents is selected, while all others are discarded. In **(P3) Answer Ranking**, a deep neural network is used to suggest the locations of candidate answers. In **(P4) Answer Selection**, finally, the probabilities are used in order to determine the best answer over all paragraphs and documents. The resulting answer is also the one that is presented to the user or leveraged for additional computations inside the proposed framework for continuous improvement from on-line feedback.

[1] We used the Stanford CoreNLP toolkit: https://nlp.stanford.edu/software/

4.1 P1: Document Retrieval

During information retrieval, all documents in the content repository are ranked based on their relevance to the input question q. Our actual implementation is analogous to the neural question answering from [11], which is widely used as a baseline. In detail, the approach from [11] represents each question q and all documents as vectors with hashed bigram counts [44] that are weighted by tf-idf. Then a dot product between the vectors of the question and each document is computed in order to obtain the relevance scores. This process is further accelerated by constructing all vectors using an inverted index in order to allow for fast retrieval.

4.2 P2: Document Selection

During document selection, the content repository is filtered for N documents that entail the highest relevance scores. It is often suboptimal to select only a single document (i. e., $N = 1$) since the recall can improve by choosing a larger subset (i. e., $N > 1$). However a larger subset also increases the proportion of noise and thus the possibility of eventually selecting a wrong answer. As a remedy, we follow the adaptive approach in [23], where the actual size n is determined based on a trainable classifier. Such an adaptive document selection is known to be especially suited to settings in which the size of the content repository is variable [23]. This adaptive selection represents an important element in our continuous learning framework, in which the dataset for training is intentionally designed to be dynamic and thus grows over time.

4.3 P3: Answer Ranking

The selected documents are further split into paragraphs and, in order to obtain greater accuracy in practice, all subsequent steps are performed at paragraph-level, as suggested by [11]. The objective is then to determine the probability of a candidate answer based on the following deep neural network. For this purpose, we compute the probability distributions over all start and end positions. These are given by p^s and p^e, respectively.

The architecture consists of an embedding layer, separate encoding layers for both the question and paragraphs, as well as a subsequent prediction layer. The network architecture adapts a simplified version of the document reader that was used in [11]; however, it is extended by additional encoding layers similar to [42] in order to yield a good predictive power while maintaining a reasonable runtime.

Input: The neural network for answer extraction receives input in the form of a paragraph $p = \langle p_1, \ldots, p_l \rangle$ of length l and a question $q = \langle q_1, \ldots, q_k \rangle$ of length k, where $p_1, \ldots, p_l$ and $q_1, \ldots, q_k$ denote the tokens therein.

Embedding Layer: The embedding layer maps each token onto a vector representation.[2] A question q is then given by an $n \times k$ matrix $\boldsymbol{Q} = [\boldsymbol{q}_1, \ldots, \boldsymbol{q}_k]$, where n is the embedding dimension and $\boldsymbol{q}_i$ (in bold) denotes the embedding vector that corresponds to the original token q_i, for $i = 1, \ldots, k$. Analogously, we translate the paragraph p into an $n \times l$ matrix $\boldsymbol{P} = [\boldsymbol{p}_1, \ldots, \boldsymbol{p}_l]$. Following [11, 26], we further compute a paragraph-aligned representation of the question $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_l]$ using question-to-query attention $\boldsymbol{u}_i = \sum_{j=1}^{k} \alpha_{i,j}\, \boldsymbol{q}_j$ with

$$\alpha_{i,j} = \frac{\exp\left(f(\boldsymbol{p}_i)^T\, f(\boldsymbol{q}_j)\right)}{\sum_{j'=1}^{k} \exp\left(f(\boldsymbol{p}_i)^T\, f(\boldsymbol{q}_{j'})\right)} \quad \text{for } i = 1, \ldots, l, \tag{3}$$

where $f(\cdot)$ refers to a fully-connected neural network layer that we implemented with ReLU activation.

Encoding Layer for Paragraphs: A multi-layered bidirectional long short-term memory (BiLSTM) is used in order to encode the previous embeddings. That is, we apply BiLSTMs to both the paragraph and the paragraph-aligned representations in order to concatenate their hidden states along the sequence of tokens, i. e.,

$$\boldsymbol{H}^P = \text{BiLSTM}_1(\boldsymbol{P}) \quad \text{and} \quad \boldsymbol{H}^U = \text{BiLSTM}_2(\boldsymbol{U}), \tag{4}$$

where $\boldsymbol{H}^P$ and $\boldsymbol{H}^U$ are matrices of size $2mh$-by-l with h being the hidden size of the LSTM and m the number of bi-directional layers (hence the 2). Notably, these are separate models – namely, BiLSTM_1 and BiLSTM_2 – which do not share any parameters. Both are finally combined into a joint encoding $\boldsymbol{G}$ of the paragraph by stacking the hidden states, their element-wise interactions [42], and exact match features [11]. More specifically, this is accomplished with the help of an additional multi-layer BiLSTM given by

$$\boldsymbol{G} = \text{BiLSTM}_3\left(\left[\boldsymbol{H}^P; \boldsymbol{H}^U; \boldsymbol{H}^P \odot \boldsymbol{H}^U; \boldsymbol{z}\right]\right), \tag{5}$$

where $\boldsymbol{G}$ is again a $2mh \times l$ matrix, $\odot$ refers to the element-wise multiplication and exact match features are represented by an l-dimensional binary vector $\boldsymbol{z} = [z_1, \ldots, z_l]$ with $z_i = 1_{[p_i \in q]}$. In other words, z_i equals one if the i-th token of a paragraph is included in the question.

Encoding Layer for Questions: Questions are encoded by feeding their embeddings into a multilayer BiLSTM network, yielding a matrix of size $2mh$-by-k that is given by

$$\boldsymbol{H}^Q = \text{BiLSTM}_4(\boldsymbol{Q}). \tag{6}$$

In order to obtain a representation $\boldsymbol{r}$ of questions that is independent of their length, we use so-called self-attention [43] which sums over all k tokens via

$$\boldsymbol{r} = \sum_{i=1}^{k} \beta_i\, \boldsymbol{h}_i^Q \quad \text{with} \quad \beta_i = \frac{\exp(\boldsymbol{w}^T\, \boldsymbol{q}_i)}{\sum_{i'=1}^{k} \exp(\boldsymbol{w}^T\, \boldsymbol{q}_{i'})}, \tag{7}$$

where $\boldsymbol{w}$ is a trainable parameter and $\boldsymbol{h}_i^Q$ is the i-th column vector of $\boldsymbol{H}^Q$.

Prediction Layer: Following [11], we obtain scores proportional ($\propto$) to the probabilities for which each token represents the start and the end position of a candidate answer. This is computed via

$$\boldsymbol{p}^s \propto \exp(\boldsymbol{G}^T \boldsymbol{W}_s \boldsymbol{r}) \quad \text{and} \quad \boldsymbol{p}^e \propto \exp(\boldsymbol{G}^T \boldsymbol{W}_e \boldsymbol{r}), \tag{8}$$

respectively, such that $\boldsymbol{W}_s$ and $\boldsymbol{W}_e$ denote weight matrices of size $2mh$-by-$2mh$ that can be trained. For any subspan a of a paragraph p that ranges from token a_s to a_e, we can now determine the probability of answering a question: this probability is proportional to the dot product between $p^s_{a_s}$ and $p^e_{a_e}$. Notably, we refrain from normalizing these scores in order to ensure comparability across paragraphs and even documents.

[2] We utilized pre-trained, case-sensitive Glove embeddings, the so-called common crawl from http://nlp.stanford.edu/data/glove.840B.300d.zip.

4.4 P4: Answer Selection

The phase for answer selection aims at determining the best answer out of all candidates, i. e., $\max_a \mathbb{P}(a \mid q, p)$. Hence, one simply optimizes over the joint probabilities of start and end positions that are given by $\boldsymbol{p}^s$ and $\boldsymbol{p}^e$, respectively. Consistent with [11], an additional constraint is enforced in order to ensure that the end token comes after the start token, as well as guaranteeing that the final answer has a maximum length (here: 15 tokens, as suggested in [11]). This yields the following optimization problem

$$\max_{a_s, a_e} \left(p^s_{a_s}\right)^T p^e_{a_e} \quad \text{s. t.} \quad a_s \leq a_e \leq a_s + 15, \tag{9}$$

which selects the final answer that is displayed to the user.

In the case of negative feedback, the answer selection module is triggered to refine the answer by selecting the next best scoring answer that is different from the down-voted answer shown before.

5 EXPERIMENTAL SETUP

5.1 Content Repository

Our QA system draws upon the English-language Wikipedia[3] as the central source of knowledge in order to answer questions. The English-language Wikipedia currently consists of more than five million documents containing more than eight million words and, therefore, provides an excellent basis for question answering that spans various disciplines and topics. Wikipedia is freely available and has an active community that keeps it up-to-date. By following a content-based paradigm for QA, we directly query Wikipedia articles and extract answers from these. We also use Wikipedia when validating the credibility of user feedback, a process in which we aggregate evidence in a distantly-supervised fashion that should validate the correctness of the feedback.

5.2 Question-Answer Corpora

We use two different question-answer corpora during our experiments in order to highlight the dynamic nature (i. e., dependent on time step t): one to initialize our system (at $t = 0$) and one to simulate user feedback after deployment (for $t > 0$); see Fig. 3. This split into two different phases represents an experimental setup that is especially challenging since both corpora stem from separate domains, each with a question-answer style that is distinctly different. Hence, it allows us to benchmark both the ability of continuous learning in relation to new question-answer pairs and its robustness against forgetting when samples stem from the initial domain.

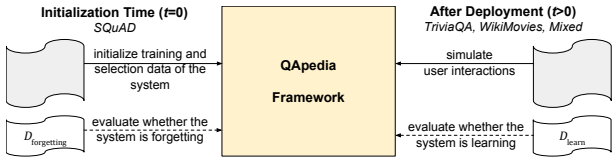


Figure 3: Overview of different corpora used to initialize our QApedia framework, as well as to simulate users.

[3]We used the preprocessed dump from https://github.com/facebookresearch/DrQA.

During *initialization time* ($t = 0$), we use the Stanford Question Answering Corpora (SQuAD) [32] to initialize the training data $\mathcal{D}^{t=0}_{\text{train}}$ and the selection data $\mathcal{D}^{t=0}_{\text{selection}}$ within the framework, and to estimate the model parameters. SQuAD includes more than 100,000 question-answer-paragraph samples with paragraphs exclusively extracted from Wikipedia. We keep a hold-out fraction of about 10,000 SQuAD samples, which we call $\mathcal{D}_{\text{forgetting}}$, in order to re-evaluate the performance over time and test whether our system is prone to catastrophic forgetting. Our hypothesis is that a system that is initialized on SQuAD and robust against catastrophic forgetting does not decrease in its initial performance on the SQuAD corpus (e. g., $\mathcal{D}_{\text{forgetting}}$).

After *deployment* ($t > 1$), we simulate user interactions on another corpus with a substantially different question-answer style. To show the robustness of our results, we run three independent experiments using the TriviaQA [19] corpus, the WikiMovies [29] corpus, and a third corpus that combines questions from TriviaQA (25 %) and WikiMovies (50 %) with the WebQuestions [6] corpus (25 %). In all three experiments, these corpora are split into a simulation part $\mathcal{D}^{t>0}_{\text{sim}}$ which provides the samples that represent our simulated users and a hold-out part $\mathcal{D}_{\text{learn}}$ in order to evaluate the extent to which continuous learning is achieved. Our hypothesis is: a system with our continuous improvement module increases its initial performance on $\mathcal{D}_{\text{learn}}$ due to learning and, at the same time, its performance remains robust against forgetting when evaluated on $\mathcal{D}_{\text{forgetting}}$.

5.3 Simulation of User Interactions

User interactions are simulated based on the following routine:

Initialization: use the SQuAD corpus to initialize the training data $\mathcal{D}^{t=0}_{\text{train}}$ and selection data $\mathcal{D}^{t=0}_{\text{selection}}$

Deployment: repeat for time steps $t = 1, \ldots$

- **(a) Simulation:** simulate $N_\phi = 10,000$ user interactions with up- and down-votes based on the corpus $\mathcal{D}^t_{\text{sim}}$ that is used after deployment
- **(b) Distant supervision:** validate the credibility of the user feedback and, if necessary, generate new data samples from the user feedback
- **(c) Dataset update:** merge training and selection data from the previous time step $t - 1$ and the new data samples into $\mathcal{D}^t_{\text{train}}$ and $\mathcal{D}^t_{\text{selection}}$
- **(d) Model update:** re-estimate the parameters inside the neural QA based on $\mathcal{D}^t_{\text{train}}$ and $\mathcal{D}^t_{\text{selection}}$ at time t
- **(e) Learning effect:** evaluate the new QA model on $\mathcal{D}_{\text{learn}}$
- **(f) Robustness against forgetting:** evaluate the new QA model on a hold-out fraction $\mathcal{D}_{\text{forgetting}}$ of the SQuAD corpus

The above procedure allows us to study the overall effect of continuous learning on $\mathcal{D}_{\text{learn}}$, as well as the robustness against forgetting on the SQuAD dataset $\mathcal{D}_{\text{forgetting}}$. We again point out that all evaluations in the following are conducted on hold-out data that has been used neither for training, simulating users nor model selection.

5.4 Clairvoyant, Noisy, and Adversarial Feedback

We study QApedia under different types of user responses:

(1) Clairvoyant users always provide feedback that is accurate. Hence, these users respond to a correct answer from the QA system with an up-vote and, conversely, with a down-vote if it is wrong. These users underlie our initial experiments in order to confirm continuous learning, while subsequent experiments control for the possibility of errors in the user feedback.
(2) Noisy feedback tests the system when a certain portion of user responses deviate from the ground-truth. Here we assume that a certain share of users (i. e., experts) respond with the correct answer, while others provide random feedback with probability ε.
(3) Adversarial feedback refers to feedback that is intentionally incorrect; i. e., up-votes if an answer is wrong and down-votes if an answer is correct. Here we again assume users that follow the ground truth and others that respond adversarially with probability ε.

For noisy and adversarial users, the total proportion of incorrect up-votes is not only determined by ε, but also by the performance of the system itself. For instance, if a system answers 20 % of the questions correctly and 10 % of the users vote randomly whereas 90 % provide correct feedback, then approximately one out three up-votes will be noisy. In our experiments, we chose a system-specific initialization of $\varepsilon = 0.05$, so that 18 % of the initial up-votes are noisy when averaged over all users. Due to its probabilistic nature, this number can vary across individual batches of user simulation and reach, e. g., up to 30 % in some of them.

Users are allowed to provide negative feedback repeatedly. Let ρ refer to the maximum frequency with which negative feedback is provided. In our experiments, we set $\rho = 1$ in order to yield a challenging setup and later experiment with $\rho = 3$ as part of a sensitivity analysis.

5.5 Implementation Details

As part of the answer ranking (P3), we train deep neural networks using mini-batch gradient descent over the content repository, i. e., the question-answer-paragraph triples, with a batch size of 32. We implemented a loss function $\mathcal{L}$ following [11]. Let a^* refer to the ground-truth answer, which spans from a_s^* to a_e^*. Given predictions $\boldsymbol{p}^s$ and $\boldsymbol{p}^e$ from the neural network, we then minimize the negative log-likelihood

$$\mathcal{L} = -\log \frac{\exp(p^s_{a^*_s})}{\sum_{j=1}^{l} \exp(p^s_j)} - \log \frac{\exp(p^e_{a^*_e})}{\sum_{j=1}^{l} \exp(p^e_j)}. \tag{10}$$

For this purpose, we utilize the AdaMax optimizer [21] with gradient clipping set to 10. All LSTM layers entail 128 hidden units and are trained with a recurrent dropout of 0.4. For all encoding layers, we use three BiLSTM layers, e. g., $m = 2$.

6 RESULTS

We conducted more than 2,500 hours of experiments and report results over more than 1.5 million simulated user interactions with 150 model re-estimations. We divide the results into three sections. First, we illustrate the learning process given shallow user feedback. Second, we check the robustness of our approach against forgetting. Third, we simulate noisy and adversarial users and evaluate whether our system still manages to improve under these challenging conditions.

6.1 Performance Improvements from Shallow User Feedback

The objective behind our first set of experiments is to determine the overall performance improvements from shallow user feedback. For this reason, we study the influence of user feedback when users always provide the correct response and, hence, consider results in the absence of noise or errors. This type of user interactions enables us to obtain an upper bound to the performance.

Fig. 4 reports our key results. In detail, we measure the fraction of correctly answered questions, e. g., where the predicted answer matches exactly to one of the ground truth answers. Per design, a traditional QA system fails to incorporate user feedback and is thus limited to a static performance. By contrast, our plots consistently derive considerable performance improvements from user feedback. Furthermore, we compare QApedia to a baseline without distant supervision in which we append samples with positive feedback directly into the training set.

- **WikiMovies:** Prior to the inclusion of user feedback, the systems obtain 13.3 % exact matches, which is identical to the performance of a static QA system. Our learning framework successfully manages to improve the performance over time. Distant supervision with shallow feedback increases the share of exact matches up to 35.5 % – an increase of 22.2 percentage points. On the other hand, the exact matches from the baseline without distant supervision amount to 30.9 %, which is inferior to QApedia by 4.6 percentage points. As determined later, the difference between this baseline and the performance of our system is attributable to the quality improvement of samples during the distantly supervised data generation process.
- **TriviaQA:** Distant supervision with $\rho = 3$ yields the best performance overall. Choosing $\rho = 1$ at first resembles the baseline without distant supervision in terms of performance but, after sufficient user interactions, outperforms it.
- **Combined dataset:** The combined dataset shows similar results. The performance differences are small, and yet the QApedia with $\rho = 3$ is still ranked first.

Evidently, continuous learning can be efficiently achieved through shallow user feedback using only a fraction of the original dataset. Choosing a larger τ yields much richer information, and yet both variants show little difference in performance, suggesting that simple feedback is already sufficient.

6.2 Robustness Against Forgetting in Neural QA

Our neural QA system could theoretically – as is the case for all other neural networks – be prone to catastrophic forgetting and we thus evaluate whether this issue is attenuated by our design. Fig. 5 compares the performance across user interactions: we measure the robustness of how well a QA system can answer questions regarding the initialization corpus (i. e., its hold-out part $\mathcal{D}_{\text{forgetting}}$) over time

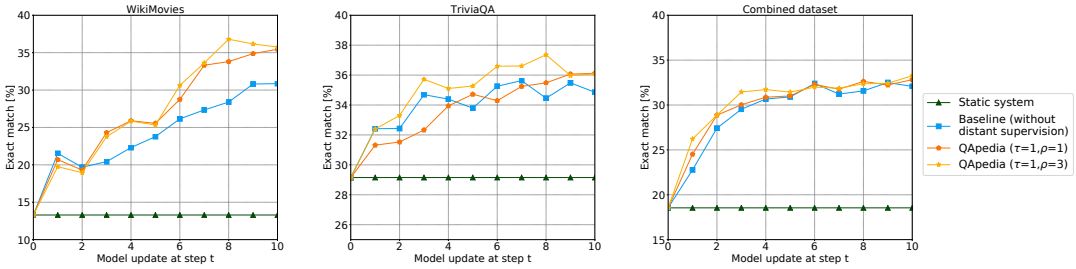


Figure 4: In a clairvoyant setting where correct feedback is provided, our QApedia achieves considerable performance improvements of up to 20 percentage points in terms of exact matches. The primary baseline represents the static QA system (i. e., horizontal line). For comparison, a trivial update framework without distant supervision is also reported.

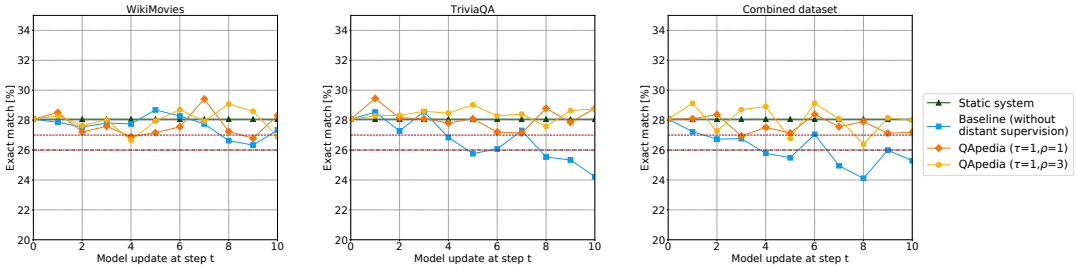


Figure 5: In a clairvoyant setting where only correct feedback is provided, our QApedia is considerably less prone to forgetting than alternative baselines. For better readability, a threshold is plotted at a 2-percentage-point decrease in performance in order to aid comparisons.

as user feedback is included. Small fluctuations are natural due to the data-driven learning method.

- **WikiMovies:** All system configurations reveal fairly similar patterns. Overall, our QApedia framework performs best with distant supervision and $\rho = 3$ repeated answers. This configuration not only shows continuous improvements for the feedback dataset, but only once attains a performance decrease of more than 1 % relative to the initial system and, moreover, reveals a gradual upward trend. The baseline without distant supervision performs less stably in this setting but, contrary to the next two datasets, does not show forgetting.
- **TriviaQA:** The performance of the baseline without distant supervision declines dramatically: the system's performance drops from 28.1 % exact matches down to only 24.2 %. In contrast, distant supervision again obtains a fairly stable performance.
- **Combined dataset:** The findings point in the expected directions, and yet we see that the mixed original of question-answer pairs in this dataset presents a challenging undertaking and results in considerable volatility. Again, the baseline without supervision hints at structural patterns of forgetting, as it regularly falls short by 2 % or more. Both distantly supervised approaches behave similarly and never undercut the 2 % mark.

Overall, our approach of distant supervision, and especially utilizing a larger value of ρ (i. e., 3 as opposed to 1), can reduce the risk of forgetting. Since the user feedback was simulated correctly (without noise), we credit this for the improved data generation via distant supervision. To obtain a better understanding, we quantitatively analyzed 200 question-answer-paragraph triplets generated from positive feedback by QApedia and the baseline without distant supervision. In the clairvoyant setting, all answers are correct, and yet we still see a distinct difference in the quality of the paragraphs. We found that, for QApedia, only about 4 % of data samples were uninformative (i. e., although the answer is correct contained within the paragraph, the paragraph alone is not sufficient to answer the question). For the baseline without distant supervision, the fraction of uninformative samples was considerably higher and amounted to 13 %.

6.3 Performance Under Noisy User Feedback

Being robust to user feedback with noise is a key requirement for use cases on the web. This experiment thus studies the effect of noise on the overall performance with noisy users as described in Section 5.4. We fix ρ to 1, and vary τ between 1 and 2. As described above, τ controls how rigorously we filter feedback during the credibility validation.

We report the experiment based on the WikiMovies dataset in Fig. 6. The total improvement declines only slightly as a result of the presence of noisy users: while it was up to 22.2 percentage points in the previous experiment based on feedback that was always correct, we now still achieve an increase in exact matches by 20.6 percentage points ($\tau = 2$) and 18.1 percentage points ($\tau = 1$). More importantly, the baseline (where positive feedback is directly added to the training set) sees an improvement of merely 9.8 percentage points, which is less than half of the performance gain from our QApedia framework. We further see that the QApedia

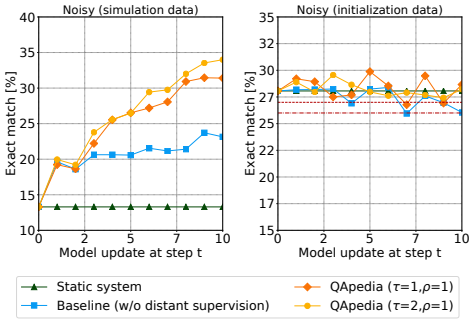


Figure 6: The proposed QApedia framework addresses a web-based use case where user feedback can be subject to noise, but remains fairly robust. Reported is the WikiMovies dataset.

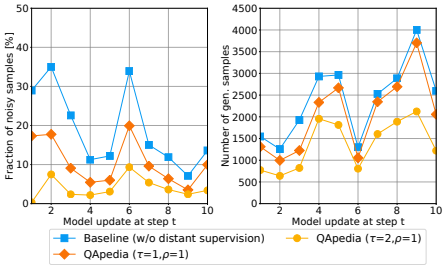


Figure 7: Our QApedia framework with automated credibility check for answers is considerably more robust than the baseline without distant supervision: our approach for the WikiMovies dataset removes more samples that received incorrect feedback (left) and thus results in fewer such samples that are added to the training set (right). This behavior is reinforced when choosing a stricter credibility check (e. g., $\tau = 2$).

framework, especially with stricter filtering, is also less prone to forgetting as compared to the baseline.

We now provide descriptive analysis concerning the update mechanism. We report (see Fig. 7 left) the relative frequency of incorrect samples that are added to the training data $\mathcal{D}_{\text{train}}$. Here we find that the baseline without distant supervision generates new training samples of which, on average, 19,2 % are wrong. Distant supervision decreases this share significantly. It plummets to 10.5 % for distant supervision with $\tau = 1$ and to 3.5 % for $\tau = 2$. Fig. 7 (right) reports the number of total samples that are generated (correct and incorrect). Here a higher τ decreases the number of samples added to the training set, but also results in fewer samples with incorrect feedback.

6.4 Robustness to Adversarial User Feedback

We repeat the previous experiments with users among whom an average of 20 % of the feedback is adversarial, i. e., users up-vote wrong answers and down-vote correct answers. The performance is depicted in Fig. 8 and resembles the previous findings under noise. The ratio of exact matches in our proposed approach increases by

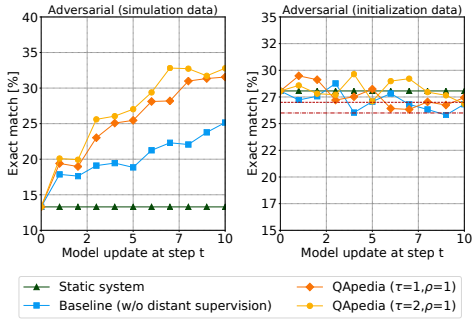


Figure 8: The proposed QApedia framework addresses a web-based use case where user feedback can be subject to adversarial inputs, but remains fairly robust.

19.4 percentage points, which is significantly larger than for the baseline without distant supervision (i. e., 11.8-percentage-point gain). Adversarial feedback leads to a larger variation in performance of the initialization task, where QApedia (in contrast to the baseline) again registers a better mean value. This is an interesting observation, since individual re-estimation steps can entail feedback of which 30 % is wrong. Altogether, these findings concerning adversarial feedback highlight the robustness of QApedia, especially with more rigorous credibility validation ($\tau = 2$).

7 DISCUSSION

This work presents QApedia: the first question-answering system for encyclopedic content repositories that gradually improves over time by harnessing shallow user feedback. A careful design was necessary in order to obtain state-of-the-art performance and, moreover, tailor the framework to the specific requirements of on-line feedback mechanisms on the web. We identified the following levers as especially pertinent. First, continuous learning from feedback can bolster the performance considerably; e. g., the proportion of exact matches can rise by up to 20 percentage points. We could have theoretically built our system such that it asks users to provide the correct labels as feedback (i. e., active crowd sourcing); however, we specifically decided upon shallow binary feedback, since, from a user perspective, this is considerably less costly. Second, our results establish that the problem of catastrophic forgetting can be effectively solved. Here it is important that all parts can efficiently deal with training sets that are dynamic in size and over time. For this reason, we utilize document retrieval, which adapts to the corpus size, and implement a novel distant supervision based on asynchronous updates. Third, we present a demanding setup with extensive adversarial feedback, and yet it does not necessarily diminish performance. Rather, our framework yields improvements of up to 30 %. This highlights the relevance of a simple credibility check.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful comments. We appreciate the help of Ryan Grabowski in editing our manuscript with regard to language. We gratefully acknowledge the support of NVIDIA Corporation, who donated of the TITAN Xp GPUs used for this research.

REFERENCES

[1] Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2018. Never-Ending Learning for Open-Domain Question Answering over Knowledge Bases. In *The Web Conference (WWW)*. 1053–1062.

[2] Abdalghani Abujabal, Mohamed Yahya, Mirek Riedewald, and Gerhard Weikum. 2017. Automated Template Generation for Question Answering over Knowledge Graphs. In *International Conference on World Wide Web (WWW)*. 1191–1200.

[3] David Ahn, Valentin Jijkoun, Gilad Mishne, Karin Müller, Maarten de Rijke, Stefan Schlobach, M. Voorhees, and L. Buckland. 2004. Using Wikipedia at the TREC QA Track. In *Text REtrieval Conference (TREC)*.

[4] Muhammad Asaduzzaman, Ahmed Shah Mashiyat, Chanchal K. Roy, and Kevin A. Schneider. 2013. Answering questions about unanswered questions of stack overflow. In *Working Conference on Mining Software Repositories*. 97–100.

[5] Michael Bendersky, Xuanhui Wang, Donald Metzler, and Marc Najork. 2017. Learning from User Interactions in Personal Search via Attribute Parameterization. In *ACM International Conference on Web Search and Data Mining (WSDM)*. 791–799.

[6] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic Parsing on Freebase from Question-Answer Pairs. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1533–1544.

[7] Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. 2008. Finding the right facts in the crowd: factoid question answering over social media. In *International Conference on World Wide Web (WWW)*. 467.

[8] Davide Buscaldi and Paolo Rosso. 2006. Mining knowledge from Wikipedia for the question answering task. In *International Conference on Language Resources and Evaluation (LREC)*.

[9] YongGang Cao, Feifan Liu, Pippa Simpson, Lamont Antieau, Andrew Bennett, James J. Cimino, John Ely, and Hong Yu. 2011. AskHERMES: An online question answering system for complex clinical questions. *Journal of Biomedical Informatics* 44, 2 (2011), 277–288.

[10] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, JR., and Tom M. Mitchell. 2010. Toward an Architecture for Never-Ending Language Learning. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*.

[11] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to Answer Open-Domain Questions. In *Association for Computational Linguistics (ACL)*. 1870–1879.

[12] Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew McCallum, Tom Mitchell, Kamal Nigam, and Seán Slattery. 2000. Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence* 118, 1-2 (2000), 69–113.

[13] Abdessamad Echihabi, Ulf Hermjakob, Eduard Hovy, Daniel Marcu, Eric Melz, and Deepak Ravichandran. 2008. How To Select An Answer String? In *Advances in Open Domain Question Answering*. 383–406.

[14] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*. 1156–1165.

[15] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A. Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John Prager, Nico Schlaefer, and Chris Welty. 2010. Building Watson: An Overview of the DeepQA Project. *AIMag (AI Magazine)* 31, 3 (2010), 59.

[16] Sanda Harabagiu, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunescu, Roxana Girju, Vasile Rus, and Paul Morarescu. 2000. FALCON: Boosting Knowledge for Answer Engines. 479–488.

[17] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems (NIPS)*. 1693–1701.

[18] Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. 2017. Learning a Neural Semantic Parser from User Feedback. In *Association for Computational Linguistics (ACL)*. 963–973.

[19] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Association for Computational Linguistics (ACL)*. 1601–1611.

[20] Michael Kaisser and Tilman Becker. 2004. Question Answering by Searching Large Corpora with Linguistic Methods. In *Text REtrieval Conference*.

[21] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[22] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America* 114, 13 (2017), 3521–3526.

[23] Bernhard Kratzwald and Stefan Feuerriegel. 2018. Adaptive Document Retrieval for Deep Question Answering. In *Empirical Methods in Natural Language Processing (EMNLP)*.

[24] Bernhard Kratzwald and Stefan Feuerriegel. 2019. Putting Question-Answering Systems into Practice: Transfer Learning for Efficient Domain Customization. In press. (2019).

[25] Eun-Ju Lee and Yoon Jae Jang. 2010. What Do Others' Reactions to News on Internet Portal Sites Tell Us? Effects of Presentation Format and Readers' Need for Cognition on Reality Perception. *Communication Research* 37, 6 (2010), 825–846.

[26] Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant. 2016. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436* (2016).

[27] Liangda Li, Hongbo Deng, Anlei Dong, Yi Chang, Ricardo Baeza-Yates, and Hongyuan Zha. 2017. Exploring Query Auto-Completion and Click Logs for Contextual-Aware Web Search and Query Suggestion. In *International Conference on World Wide Web (WWW)*. 539–548.

[28] Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Association for Computational Linguistics (ACL)*. 1736–1745.

[29] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-Value Memory Networks for Directly Reading Documents. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1400–1409.

[30] T. Mitchell, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, W. Cohen, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, J. Welling, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, and M. Gardner. 2018. Never-ending learning. *Commun. ACM* 61, 5 (2018), 103–115.

[31] Dragomir Radev, Weiguo Fan, Hong Qi, Harris Wu, and Amardeep Grewal. 2005. Probabilistic question answering on the web. *Journal of the Association for Information Science and Technology* 56, 6 (2005), 571–583.

[32] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Empirical Methods in Natural Language Processing (EMNLP)*. 2383–2392.

[33] Gerard Salton and Chris Buckley. 1990. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science* 41, 4 (1990), 288–297.

[34] Avneesh Saluja, Ian Lane, and Ying Zhang. 2012. Machine translation with binary feedback: a large-margin approach. In *Proceedings of the Tenth Biennial Conference of the Association for Machine Translation in the Americas*.

[35] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603* (2016).

[36] Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. 2018. Building a Conversational Agent Overnight with Dialogue Self-Play. *arXiv preprint arXiv:1801.04871* (2018).

[37] Dan Shen and Dietrich Klakow. 2006. Exploring correlation of dependency relation paths for answer extraction. In *Annual Meeting of the Association for Computational Linguistics*. 889–896.

[38] R. F. Simmons. 1965. Answering English questions by computer: A survey. *Commun. ACM* 8, 1 (1965), 53–70.

[39] Yu Su, Shengqi Yang, Huan Sun, Mudhakar Srivatsa, Sue Kase, Michelle Vanni, and Xifeng Yan. 2015. Exploiting Relevance Feedback in Knowledge Graph Search. In *ACM International Conference on Knowledge Discovery and Data Mining (KDD)*. 1135–1144.

[40] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-To-End Memory Networks. *Neural Information Processing Systems (NIPS)* (2015), 2440–2448.

[41] Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. 2012. Template-based question answering over RDF data. In *Conference on World Wide Web (WWW)*.

[42] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. 2018. R3: Reinforced Ranker-Reader for Open-Domain Question Answering. In *Conference on Artificial Intelligence (AAAI)*.

[43] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated Self-Matching Networks for Reading Comprehension and Question Answering. In *Annual Meeting of the Association for Computational Linguistics*. 189–198.

[44] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. 2009. Feature hashing for large scale multitask learning. In *International Conference on Machine Learning (ICML)*.

[45] Mohamed Yahya, Klaus Berberich, Shady Elbassuoni, and Gerhard Weikum. 2013. Robust question answering over the web of linked data. In *ACM International Conference on Information & Knowledge Management (CIKM)*. 1107–1116.

Blockchain Mining Games with Pay Forward

Elias Koutsoupias
Department of Computer Science, University of Oxford
elias@cs.ox.ac.uk

Philip Lazos
Department of Computer Science, University of Oxford
plazos@gmail.com

Foluso Ogunlana
Department of Computer Science, University of Oxford
ConsenSys
foogunlana@yahoo.com

Paolo Serafino
Department of Computer Science, University of Oxford
paolo.serafino@cs.ox.ac.uk

ABSTRACT

We study the strategic implications that arise from adding one extra option to the miners participating in the bitcoin protocol. We propose that when adding a block, miners also have the ability to pay forward an amount to be collected by the first miner who successfully extends their branch, giving them the power to influence the incentives for mining. We formulate a stochastic game for the study of such incentives and show that with this added option, smaller miners can guarantee that the best response of even substantially more powerful miners is to follow the expected behavior intended by the protocol designer.

CCS CONCEPTS

• **Theory of computation → Algorithmic game theory and mechanism design**.

KEYWORDS

Mechanism Design; Nash Equilibrium; Blockchain Mining; Stochastic Games

ACM Reference Format:
Elias Koutsoupias, Philip Lazos, Foluso Ogunlana, and Paolo Serafino. 2019. Blockchain Mining Games with Pay Forward. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313740

1 INTRODUCTION

Bitcoin, currently the most-widely used cryptocurrency, was introduced in a 2009 white paper [12] by the mysterious Satoshi Nakamoto as a form of decentralized, distributed, peer-to-peer digital currency. The backbone of the bitcoin protocol is the *blockchain*, a distributed ledger that (ideally) takes the form of a chain where bitcoin *transactions* are stored into *blocks*. Blocks are created by special nodes of the bitcoin network called miners that: (*i*) collect and validate the set of transactions to be included in a block and (*ii*) solve a crypto-puzzle (the so-called *proof of work*) that cryptographically links the newly created (*mined*) block to the tail of the existing blockchain. The main purpose of the blockchain is to solve the problem of *distributed consensus*, where the consensus to be obtained is on the history (and relative order) of the bitcoin transactions.

Since mining new blocks is a computationally intensive (and expensive) task, miners need a reward scheme to keep mining blocks. Bitcoin has two main reward schemes for miners: (*i*) *transaction fees*, that are left by bitcoin users on a voluntary basis (and hence vary in frequency and size) and *coinbase* (or block) rewards, which are constituted by a fixed set amount (currently 12.5Ƀ) of newly minted bitcoins. Coinbase rewards are the only way by which bitcoins can ever be created, and the protocol specifies that (to avoid inflation) coinbase rewards be decreasing over time, until the limit of 21 million bitcoins are created: at that point the only incentive scheme left for miners will be transaction fees. Currently, most of the rewards for mining a block are coinbase.

In a distributed setting populated by selfish miners, however, the blockchain will hardly be a chain at all, but rather a *tree*. For instance, network delays may lead miners to add newly mined blocks to different blocks that they believed to be the tail of the chain. For this reason, newly mined blocks are not part of the consensus until sufficient time has passed since their creation (i.e., d levels of other blocks are added to the blockchain – currently $d = 100$), and block rewards become available to miners only then. Worse still, even though the bitcoin protocol prescribes that miners should mine from the last known block in the chain (the so-called Frontier strategy), self-interested miners may try to create a *fork* by intentionally adding a sequence of blocks (a *branch*) constituting a parallel history of the transactions, in an attempt to reap more block rewards or to double-spend bitcoins (once per each branch of the fork).

It should be clear from now that the bitcoin protocol is rife with game-theoretic issues. Nakamoto [12] analyzed in a simple model, providing rough estimates showing that if a large majority of miners follow Frontier, then their chain will be the longest and contain the agreed upon history.

Since Nakamoto's paper, there has been significant work about the strategies of miners, mainly under the assumption that the reward per block is fixed (see for example [5, 8, 16]). In particular, these *mining games* have been systematically evaluated through the lens of game-theory by Kiayias et al. [8]. They considered mining games in which miners can influence the blockchain in two ways: by strategically choosing to mine at different branches in an attempt to overtake the longest branch and by withholding their mined blocks and releasing them at the right time, wasting

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313740

everyone else's computational power. In [8, 16], it was shown by both formal proofs and simulations that the Nakamoto protocol is stable when no miner has computation power more than 0.33 of the total. Furthermore, it was also shown that when the miners do not withhold blocks, the protocol is stable if no miner has computation power more than 0.42.

This raises the issue of *fairness* and *compliance* in the bitcoin protocol. By deviating from the protocol and mining strategically, large miners could affect the network in two ways. By increasing their own rewards disproportionately to their computational power they limit the mining rewards claimed by smaller miners, which is unfair. Also, their constant forking in an attempt to add more blocks upsets the protocol and can cause delays, as well as negate transactions that have been already processed.

In this work, we propose a slight modification to the protocol whereby miners can entice other miners to mine at their block by adding a *pay-forward* amount. This pay-forward amount is collected by the miner of the next block, thus providing incentives to other miners to try to extend the tree from this particular branch. When we extend the available strategies for the miners by adding the ability to pay-forward, we reclaim some stability by ensuring that the attacking (computationally powerful) miner is incentivized to follow the protocol, even if he still reaps more than his fair share of rewards.

The main technical contribution is the study of mining games in which the strategies are extended by pay-forward. We show that with this extension, the stability of the protocol increases significantly: the computational power needed by a dishonest miner to disrupt the blockchain is substantially higher. This shows an interesting trade-off for small miners: on one hand they lose the pay-forward amount but on the other hand, they provide the right incentives to large miners to play Frontier and thus secure that their block will end up in the longest chain.

1.1 Our Results

We consider two types of *stochastic games*, whose states are rooted trees. The game is played in discrete time-steps. At the beginning of each time-step, every miner chooses a block and tries to mine from it. Each player i has probability p_i to mine a new block, proportional to his computational power. In the end, the new block may be added to the blockchain or kept hidden and released later. After many rounds of playing, the utility of each player is the fraction of bitcoins he owns (from block and pay-forward rewards) over the total value that has been mined in the longest chain.

Let p be the computational power of the strongest player, called Miner 1. If p is large enough, his best response, given that everyone else is playing Frontier, may be a different strategy. We find thresholds on p and w, the pay forward amount of every other player, to guarantee that mining at the end of the longest chain is a best response for Miner 1. As in [8], we consider two variants. In the immediate release case, any mined block has to be added to the blockchain immediately for other players to use. In this case, without pay forward rewards, it was proven in [8] that Frontier is a best response for Miner 1 for $p < h$, where $0.361 \leq h \leq 0.455$. Experimentally, they also showed that there is always a deviating strategy for $h \approx 0.42$. In the strategic release case, the newly mined blocks are public knowledge, but can only be used by other players when their creators decide to *release them.* As before, it was shown in [8] that there is a threshold $\hat{h} \geq 0.308$ (experimentally shown to be approximately 0.33 [8, 16]) such that $p < \hat{h}$ leads to Frontier as a best response for Miner 1.

We improve these thresholds to $h = 0.5$ and $\hat{h} \geq 0.344$ by showing that there exists some w (as a function of p) that guarantees Miner 1's best response is to play Frontier when all remaining miners play Frontier and pay forward w. In addition, we experimentally show that for $h \leq 0.44$ that outcome is also a pure Nash equilibrium for the immediate release case, while for the strategic release case we find that $\hat{h} \approx 0.38$. We also devise a linear program to calculate the minimum value of w for any given p.

1.2 Related Work

Bitcoin, originally introduced by Nakamoto in [12] and followed by several cryptocurrencies, such as Litecoin, Ethereum and Ripple, initiated the research on blockchains. Nakamoto's original paper analyses double spending attacks while Rosenfeld provides a more detailed analysis [15]. Bonneau et al. [2] and Tschorsch and Scheuermann [20] provided extensive surveys of the research and challenges in cryptocurrencies.

Kroll et al. [10] was one the first papers to consider the economics of Bitcoin mining, assuming that participants behave according to their incentives. Eyal and Sirer [7] showed that the security of bitcoin is not guaranteed by a majority of honest miners as was previously assumed. They gave a specific mining strategy and argued that a pool of miners, with at least a 1/3 of the total processing power, can get extra profit regardless of the block propagation characteristics of the network. With sufficiently favorable block propagation, this threshold of processing power falls to 0. Extending this, Sapirshtein et al. [16] provided systematic analysis of the space of selfish mining strategies based on computational results.

A similar approach was taken by Kiayias et al. [8] who provided a framework for studying the strategic considerations made by Bitcoin miners. They formulated two abstract stochastic games and proved rigorous bounds on the threshold of computational power below which the honest strategy is a Nash Equilibrium. Carlsten et al. [3] extended the previous approaches by investigating mining games when the reward for miners varies and comes mainly from transaction fees. They observed that the random block arrival times lead to high variance in rewards and they considered strategies that lead to instability. They showed via simulation that an equilibrium exists, but it has the counter intuitive and undesirable effect of a growing backlog of unprocessed transactions.

Several other studies examine possible attacks on the protocol and suggest adaptations to ensure its security. Rosenfeld [14] and Courtois and Bahack [4] discuss pool mining attacks. Eyal [5] introduces an attack in which mining pools infiltrate one another resulting in a pool game. Lewenberg et al. [11] also provide a game theoretic analysis of pool mining. Babaioff et al. [1] consider Sybil attacks on the network and propose a reward scheme to prevent miners from hiding transactions in competition with other miners.

There is also considerable work on the performance and scalability of blockchain inspired networks. Sompolinsky and Zohar's

[18] Greedy-Heaviest-Observed-Sub-Tree rule is an alternative consensus mechanism. Eyal [6] proposes Bitcoin-NG which increases the throughput of Bitcoin. Poon and Dryja's Lightning Network [13] scales via off-chain transactions and hashed commitments. Sompolinsky and Zohar's PHANTOM [19] achieves near unlimited transaction throughput. SPECTRE [17], by Sompolinsky et al, fully orders the transactions in blocks using recursive elections and can used in combination with PHANTOM. Kiayias et al. developed Ouroboros[9], a proof of stake network with provable security, utilising a secure multi-party coin flipping protocol.

2 MODEL AND NOTATION

The model contains subtleties in the abstractions that at first glance might be overlooked by readers who are too familiar with the bitcoin protocol. We do our best to point out any of those instances, but be especially vigilant when reading about the way payoffs are calculated.

The bitcoin mining game with pay forward is an abstraction of the actual protocol, simplified in a way that can only accentuate its game theoretic issues (i.e. negative results hold a fortiori for the actual protocol) while being open to rigorous analysis. The parameters of the game are:

(1) the number n of miners (or players).
(2) the probabilities $p_1, \ldots, p_n$, representing the *hash power* of each miner, which is the probability that they solve the crypto-puzzle. They are proportional to the computational power of each miner and such that: $p_i \geq 0$ for each i and $\sum_{i=1}^{n} p_i = 1$.
(3) the depth d after which mined blocks are 'paid', i.e. their coinbase becomes usable by the respective miner. Without loss of generality we mostly consider $d = \infty$ which gives the attacker extra power.

We assume that each block has a *fixed reward*, which by appropriate scaling is equal to 1. This is reasonable, as the large majority of rewards come from the fixed coinbase rewards. The fluctuating transaction fees (while not insignificant) are still comparatively small.

During the execution of the protocol, miners add their blocks to the blockchain one at a time[1]. Their goal is to maximize the fraction of their blocks on the longest branch of the blockchain, as the longest branch represents the consensus and all other branches are pruned and the computational power spent for their creation is wasted. This does not mean that all the miners try to extend the longest branch at every time step, because some times they might benefit by either adding a block to a shorter branch or withholding it for later.

Definition 2.1. A public state is a rooted tree. Every node is labelled by one of the players and the amount of money he pays-forward. The nodes represent mined blocks and the label indicates the player who mined the block. The pay-forward amount is collected by the next miner in the longest branch. Every level of the tree has at most one node labelled i because there is no reason for a player to mine twice the same level.

A private state of player i is similar to the public state except it may contain more nodes called private nodes and labelled by i. The public tree is a subtree of the private tree and has the same root.

The incomplete information case (where the p_i's or private states are unknown) is significantly more complicated. In this work we only consider the full information case where even the private states are common knowledge (but only become part of the public state when their respective miners add them). We consider two variants:

Immediate-release model Every mined block is added to the public tree immediately.

Strategic-release model Mined blocks can be withheld: every miner is aware of their existence, but they can only mine from them once they are added to the public tree.

The second model has no counterpart in practice, but it serves as an intermediate step between the full and incomplete information models and allows us to study issues around strategic release of blocks. If a strategy is not dominant in this model it cannot be dominant in the incomplete information setting.

Definition 2.2 (Strategy). The strategy of miner i can be fully characterized by three functions $\mu_i, r_i, \mathrm{PF}_i$:

- the mining function μ_i selects a block of the public state to mine from
- the release function r_i which is the rooted tree he releases to the public. It is a subtree of his private state that contains the public state.
- the pay-forward function PF_i which is the amount of money to be left for the next miner.

Both functions depend on the public and private states of every player.

The original suggested strategy in [12] is Frontier.

Definition 2.3 (Frontier). A miner follows strategy Frontier if he releases mined blocks immediately, always mines at the deepest node in the blockchain and pays forward 0 B.

In the following, we refer to Frontier(w) as the strategy that mines and releases like Frontier but always pays forward w.

The game is played in *phases*. At each phase exactly one miner will mine a block. Then he will choose if he wants to release it, which may trigger a cascade of releases from other players. Eventually, once no one else wants to release anything the public knowledge is updated and the next phase begins. Even if no one releases a block, miners know when the phase ends since everything is public knowledge.

To incentivise miners to mine new blocks, payments are necessary. When a block is added, it is unclear if it is going to remain on the longest branch permanently. To remedy this, blocks are paid for only after their branch is increased by d blocks, after which time it is safe to assume it will stay in the longest branch. Blocks are paid through coinbase, transaction fees and pay-forward. Currently, coinbase rewards (which can be claimed at $d = 100$) far outweigh

[1] Due to the decentralized nature of bitcoin, it is possible for more than one miners to mine a block simultaneously. This event does occur by chance, but it is very rare and it is not something a miner can plan for. For this reason, we assume that exactly one block is mined at every step, but most of our analysis could be easily extended to the case in which each miner succeeds with probability p_i independently.

transaction fees. However, by design Bitcoin will eventually do away with coinbase rewards to limit inflation, at which point only transaction fees will remain, leading to potential instability [3]. We only consider coinbase rewards in this work. For game theoretic analysis we rigorously define payments:

Definition 2.4 (Payments). For some nodes of the tree, the miners who discovered them will get a payment (coinbase rewards are normalized to 1). The payments comply with the following rules:

- the blocks that receive a payment must form a path from the root. This immediately adds the restriction that at every level of the tree exactly one node receives payment.
- among the blocks of a single level that satisfy the above condition, the first one which succeeds in having a descendant d generations later receives payment.

Since only one block per level is paid for and they form the longest branch, the utility of a miner in the long run is defined as the fraction of pay-forward and coinbase rewards he obtained minus the pay-forward he paid in that branch, over it's length.

When a block is paid for, strategic miners will ignore any branch that starts at an earlier node. This limits the shape of the public state to a long path (the *trunk*) with ignored *stale* branches dangling from it. Only at the end of the trunk we may find multiple active branches, competing to reach length d and render the others stale. We will also consider the case where $d = \infty$. Here there is the possibility that two competing branches will go on forever, but since one of them will have less computational power, a case of gamblers ruin makes this impossible.

3 IMMEDIATE RELEASE

In this section we show that if Frontier(w) (for some value of w) is followed by all miners but one, the remaining miner's best response is Frontier, provided his hash power is $p < 0.5$. Since only one miner might deviate, we can view this situation as Miner 1 being the potentially deviant large miner with hash power $p < 0.5$ and Miner 2 all small miners combined into one, which we call the honest miners, since they follow the same strategy.

In essence, every honest player mines at the deepest node and always pays forward w. This amount propagates across the small miners forming Miner 2 but Miner 1 can claim it for himself, leading to an interesting stochastic game where Miner 1 is incentivise to mine the longest chain, since the rewards don't accumulate.

The blockchain itself is a rooted tree, but after pruning abandoned branches we are left with one long, undisputed, main path followed by two branches of length a and b, mined by Miner 1 and Miner 2 respectively. Therefore, the possible states of the game have the form (a, b, c), where $a \leq b + 1$ since Miner 2 always mines the longest branch (except briefly, right when Miner 1's branch overtakes the longest) and c is either 0 or 1, to indicate whether the *last block before the fork* contained a pay forward reward.

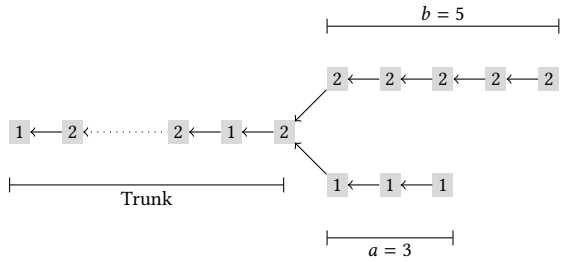


Figure 1: A typical state tree. The trunk represents the blocks whose rewards have already been collected. The current state $(3, 5, 1)$ of the game is represented by the blocks mined by Miner 1 and Miner 2 and $c = 1$ because Miner 2 controls the block before the fork.

The set of states (a, b, c) can be partitioned as follows:

(1) **Winning states**: the set W of states where Miner 2 capitulates and starts mining from the tip of Miner 1's branch. This happens exactly when $a = b+1$, therefore $W = \{(b+1, b, c) \mid b \geq 0 \text{ and } c \in \{0, 1\}\}$. After Miner 1 overtakes, the new state of the game is $(0, 0, 0)$ since they both mine at the same point.
(2) **Capitulation states**: the set C of states where Miner 1 capitulates, abandons his branch and mines from a block of Miner 2's branch, thus moving the game to state $(0, s, 1)$ for some s s.t. $0 \leq s < b$. We say that Miner 1 capitulates *at* (a, b, c) and *to* $(0, s, 1)$. Clearly, after capitulating there would be a pay forward reward available.
(3) **Mining states**: the set M of states where Miner 1 and Miner 2 mine their respective branches.

Miner 1 can capitulate to any state $(0, s, 1)$ and will always choose the one that maximizes his payoff. Since he is rational, when capitulating from state (a, b, c) he would only go to states $(0, s, 1)$ with $s < b$, otherwise he would be undercutting his own tentative branch. The strategy Frontier has $M = \{(0, 0, 0), (0, 0, 1)\}$, $C = \{(0, 1, 0), (0, 1, 1)\}$ and always capitulates to $(0, 0, 1)$.

Let $g_k(a, b, c)$ denote the optimal expected gain of Miner 1 starting from state (a, b, c) when the longest chain is extended by k levels. Knowing Miner 1 will never pay-forward, we can recursively define:

$$g_k(a,b,c) = \begin{cases} g_{k-1}(0,0,0) + a + w \cdot c & \text{if } a = b+1 \\ \max \begin{cases} \max_{s=0,\ldots,b-1} g_k(0,s,1) \\ p g_k(a+1,b,c) + (1-p) g_{k-1}(a,b+1,c) \end{cases} & \end{cases}. \tag{1}$$

The first line represents Miner 2 capitulating to Miner 1's chain. The second line containing the max has two cases. In the first case, Miner 1 capitulates to the optimal point within Miner 2's chain. In the second, they both mine and extend their chains with probability p and $(1-p)$ respectively. Notice that the first term is $pg_k(a+1, b, c)$ and not $pg_{k-1}(a+1, b, c)$. Since $a < b + 1$, if Miner 1 adds a block the deepest level of the blockchain does not change, thus k remains the same.

Clearly, Miner 1 eventually will add some block to the chain and the game will restart at state $(0, 0, 0)$. Therefore, we expect that

the initial state (a, b, c) has no effect asymptotically. We define the expected gain per level $g^\star$ as:

$$g^\star = \lim_{k\to\infty} \frac{g_k(a, b, c)}{k}. \tag{2}$$

We can decompose $g^\star$ into two separate quantities:

$$g^\star = q_M + q_{\text{PF}} \cdot w, \tag{3}$$

where q_M is the expected fraction of blocks he mined and q_{PF} the expected fraction of blocks mined and containing pay forward rewards.

Lemma 3.1. *If Miner 1 follows Frontier, we have $q_M = p$ and $q_{PF} = p(1-p)$.*

Proof. If Miner 1 is honest then the probability that he mined any arbitrary block is p. That block also contains pay forward rewards if the one that immediately preceded it was mined by Miner 2, which occurs with probability $p(1-p)$. □

We are ready to state the main theorem of this section.

Theorem 3.2. *For every $p < 0.5$, there exists $w \geq 0$ large enough so that if every miner but one follows Frontier(w), the best response of the remaining miner with hash power p is Frontier.*

Proof. We will show that q_{PF} attained by Frontier is at least as large as that of any optimal strategy, and in particular it is larger for $0 < p < 0.5$. □

Lemma 3.3. *If Frontier is not the optimal strategy and $0 < p < 0.5$, we have that $p(1-p) > q_{PF}$. For $p = 0$ or $p = 0.5$ we have equality.*

Proof. Assuming Frontier is not an optimal strategy, we consider different cases, depending on the actions of the optimal strategy starting from state $(0, 1, c)$. □

Clearly, we have that $g_k(0, 1, 1) \geq g_k(0, 1, 0)$ for all k. To see this, consider what happens if we apply exactly the same sequence of actions starting from $(0, 1, 0)$ and from $(0, 1, 1)$. As the only difference between the two states is the initial pay forward amount, if we start from $(0, 1, 1)$ the reward will be the higher (in expectation) by some fraction of w. Therefore, if the optimal strategy capitulates at $(0, 1, 1)$ it must also capitulate at $(0, 1, 0)$. This is exemplified in Figure 2, where blocks with a red outline represent those mined by the optimal strategy, and red arrows show how the action of the optimal strategy in one case implies the same action in the other.

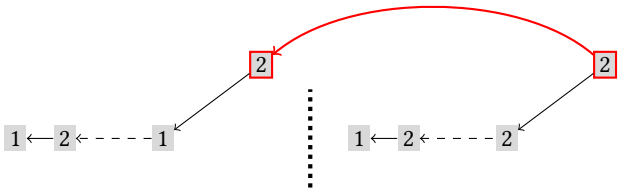


Figure 2: Mining like Frontier

But in this case, the optimal strategy capitulates at exactly the same states as Frontier, as shown in Figure 2. Since the blockchain starts empty, at $(0, 0, 0)$ this strategy would behave exactly the same as Frontier and would therefore not be optimal.

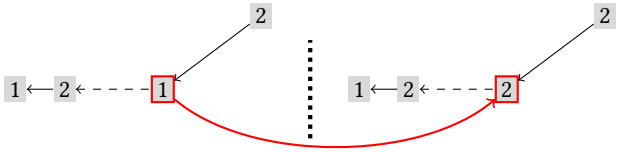


Figure 3: Case 1

In Case 1 (shown in Figure 3), we assume that the optimal strategy mines at $(0, 1, 0)$. As before, since $(0, 1, 1)$ is a more beneficial state it would mine from there as well. We can therefore assume that the optimal strategy always capitulates to states $(0, s, 1)$ with $s \geq 1$. Blocks are permanently added to the blockchain only after a capitulation. Given that, in order to compute q_{PF} for Miner 1, we construct the Markov chain shown in Figure 4, with states $(0, 0, 0)$ and $(0, s, 1)$ indicating that Miner 2 or Miner 1 capitulated respectively. Each transition is labelled with a pair where the first element is the transition probability, whereas the second element is the minimum number of blocks added to the chain when the transition occurs.

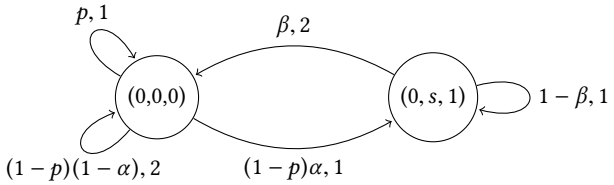


Figure 4: Case 1 Markov chain

Starting from $(0, 0, 0)$, with probability p Miner 1 adds a block and the game restarts. With probability $1 - p$ we move to state $(0, 1, 0)$ and the optimal strategy has probability α of losing the race and capitulating to some state $(0, s, 1)$. From there, the optimal strategy has probability β of winning and returning to $(0, 0, 0)$.

As q_{PF} is an asymptotic quantity, we are interested in the fraction of pay forward rewards per block mined at the stationary distribution. Let π be the stationary probability of state $(0, s, 1)$. Then:

$$\pi = \frac{\alpha - \alpha \cdot p}{\alpha - \alpha \cdot p + \beta} \tag{4}$$

In the actual game, multiple blocks are potentially mined in each edge transition, and only the first block might claim a pay forward reward. On every transition at least one block is permanently added, *except from* $(0, s, 1)$ *to* $(0, 0, 0)$ *and* $(0, 0, 0)$ *to itself through transition* $(1-p)(1-\alpha)$. In these cases at least two blocks are added, since Miner 1 must mine at least two blocks to be ahead of Miner 2. From $(0, s, 1)$ to $(0, 0, 0)$ is the only transition where Miner 1 wins a block containing pay forward. Dividing the blocks with pay-forward over the total amount mined at the stationary distribution:

$$q_{\text{PF}} \leq \frac{\pi\beta}{(1-\pi)(2(1-\alpha)(1-p) + \alpha(1-p) + p) + \pi(2\beta + 1 - \beta)} \tag{5}$$

We now need to set the parameters α and β optimally to maximize the upper bound on q_{PF}. We take the derivative of the upper bound of q_{PF} with respect to α and β to get:

$$\frac{dq_{\text{PF}}}{d\alpha} = \frac{\beta^2(2-p)(1-p)}{(\alpha(p-1)+\beta(p-2))^2} \geq 0 \tag{6}$$

and

$$\frac{dq_{\text{PF}}}{d\beta} = \frac{\alpha^2(1-p)^2}{(\alpha(p-1)+\beta(p-2))^2} \geq 0. \tag{7}$$

From [8, Lemma 1], we know that the probability that Miner 1 reaches a winning state starting from (a, b) is at most $(p/(1-p))^{(b-a+1)}$. Therefore, the optimal strategy must have

$$\alpha \leq 1 \text{ and } \beta \leq \left(\frac{p}{1-p}\right)^{s+1} \leq \left(\frac{p}{1-p}\right)^2 \tag{8}$$

Since q_{PF} is an increasing function of α, β, the upper bound is obtained by setting them to their highest possible value, leading to:

$$q_{\text{PF}} \leq \frac{(1-p)p^2}{1-(1-p)p(3-2p)}. \tag{9}$$

Compared to the honest outcome $p(1-p)$, we have:

$$p(1-p) - \frac{(1-p)p^2}{1-(1-p)p(3-2p)} = \frac{(1-p)p(1-2p)}{1-(1-p)p(3-2p)} \geq 0, \tag{10}$$

with equality obtained only for $p = 0$ and $p = 0.5$. We also plot this result at the end of the proof.

In the last case, the optimal strategy capitulates at $(0, 1, 0)$ but mines at $(0, 1, 1)$.

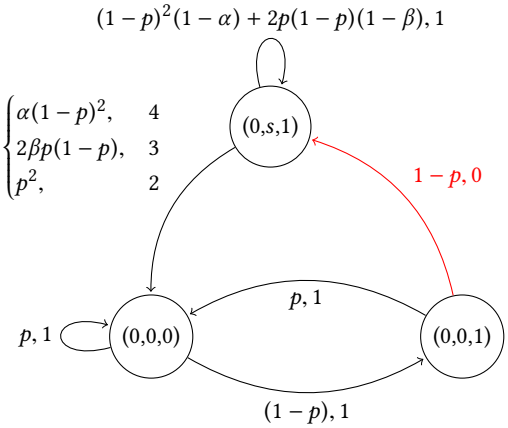


Figure 5: Case 2 Markov chain

The Markov chain in this case (Figure 5) is trickier. As before each state represents a capitulation: $(0, 0, 0)$ for Miner 2 and $(0, 0, 1)$, $(0, s, 1)$ with $s \geq 1$ for Miner 1. However, the transition indicated with the red arrow is *not* a capitulation: it is merely a move to a state that Miner 1 could also capitulate *to*. To more accurately model the transitions of state $(0, s, 1)$, we consider multiple moves ahead. Specifically, three outcomes could happen before the optimal strategy considers capitulating. With probability p^2, Miner 1 could add two blocks and move to $(0, 0, 0)$. Miner 2 could also add two blocks and move to $(0, s + 2, 1)$, with probability $(1-p)^2$. Then, Miner 1 wins with probability α. Finally, Miner 1 and Miner 2 could both add one block leading to $(1, s + 1, 1)$ with probability $2p(1-p)$. From there, Miner 1 wins with probability β.

Carefully adjusting for the more complicated transitions, the same analysis as before holds. For both cases, we plot the difference $p(1-p) - q_{\text{PF}}$ in Figure 6.

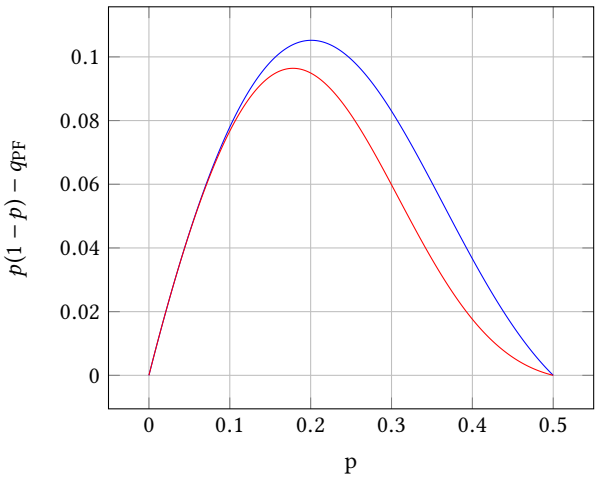


Figure 6: Advantage of Frontier in collecting q_{PF} for Case 1 (blue) and Case 2 (red)

Therefore, by (3) and Lemma 3.1 we have that

$$g^\star = q_M + q_{\text{PF}} \cdot w \leq p + (1-p) \cdot w \Rightarrow w \geq \frac{q_M - p}{(1-p) - q_{PF}}$$

and since $(1-p) > q_{PF}$ for $0 < p < 0.5$ there must be some value of w large enough to make Frontier the best response. Unfortunately, this result is not enough to calculate w. Giving crude upper bounds on q_M or q_{PF} independently is not too hard, we can use [8, Lemma 1] from example. However, because the trade-off between the two is hard to establish and the actual blockchain does not use $d = \infty$, it is more useful to develop an algorithm that finds the minimum w required for a specific d, to any degree of accuracy.

3.1 Calculating the optimal w for finite d

In reality, the values of w needed are not too large. We could attempt to find the minimum w necessary by computing $g_k(0, 0, 0)/k$ for large values of k and comparing with $p + p(1-p)w$, but a naive implementation would take $O(k^3)$ time. To simplify this search we can define a potential as in [8]

$$\phi(a, b, c) = \lim_{k\to\infty} g_k(a, b, c) - k \cdot g^\star, \tag{11}$$

that captures the advantage of Miner 1 at different states. This quantity is bounded, because irrespective of the initial state (a, b, c) the game repeats: in particular, when Miner 1 gets ahead of Miner 2 they both restart at state $(0, 0, 0)$. Therefore, the gain per k asymptotically converges to $g^\star$ and ϕ is well defined. Following recurrence (1), we have:

$$\phi(a,b,c) = \begin{cases} \phi(0,0,0) + a + c \cdot w - g^\star & \text{if } a = b+1 \\ \max \begin{cases} \max_{s=0,\ldots,b-1} \phi(0,s,1) \\ p \cdot \phi(a+1,b,c) \\ \quad +(1-p) \cdot (\phi(a,b+1,c) - g^\star) \end{cases} & \end{cases}, \quad (12)$$

where we set $\phi(0,0,0) = 0$. Finding a ϕ that satisfies these constraints is even harder. However, if we truncate the game at depth d, it becomes more feasible through linear programming. Notice that (12) (which holds for $d = \infty$) does not explicitly define the value of $\phi(a,b,c)$: the recursion is unbounded. By truncating the game at d we limit the available states to $a, b \le d$. Specifically, Miner 1 *has* to capitulate when $b = d$.

The exact value of d has a small effect on the game. In particular, Miner 1 has more options as d increases, since he does not have to capitulate early. Therefore, the minimum w needed for compliance is a slightly increasing function of d. The effect of this is quite muted however; the probability of having two branches with length greater than d, without Miner 1 winning or capitulating, decreases exponentially in d. In this section we select $d = 8$, as it is large enough to show the trend, but not too large to render the LP computationally infeasible.

Relaxing the two maxes by inequalities, we get the following LP:

$$\begin{aligned} &\text{minimize} \quad g + \tfrac{1}{D} \sum_{a=0}^{d} \sum_{b=0}^{d} \sum_{c=0}^{1} \phi(a,b,c) \\ &\text{subject to} \\ &\quad \phi(b+1,b,c) \ge \phi(0,0,0) + b + 1 + c \cdot w - g \\ &\qquad \text{for } b < d, c \le 1 \\ &\quad \phi(a,b,c) \ge \phi(0,s,1) \\ &\qquad \text{for } a \le b < d, s < b, c \le 1 \\ &\quad \phi(a,b,c) \ge p \cdot \phi(a+1,b,c) + (1-p)(\phi(a,b+1,c) - g) \\ &\qquad \text{for } a \le b < d, c \le 1 \end{aligned},$$

where $\phi(a,b,c) \ge 0$ and $D \gg d^2$ is a normalizing factor to keep the sum of states insignificant compared to g. The constraints are straightforward enough, but the objective is a minimization, which might appear odd at first. The second term of the sum is only there to ensure that feasible solutions only contain potential functions ϕ that tightly satisfy (12). Since many states could be unreachable (for example, if w is high enough so that Miner 1 plays Frontier, the only mining states would be $(0,0,0)$ and $(0,0,1)$), their exact value does not affect g and we need this 'extra' term to keep them in check, just to bias the LP towards certain feasible solutions. We minimize g to use the following lemma:

Lemma 3.4. *For any g, ϕ pair satisfying (12) we have:*

$$g_k(a,b,c) \le \phi(a,b,c) + k \cdot g.$$

Proof. We use induction on k. Clearly, for $k = 0$ we have $g_0(a,b,c) = 0 \le \phi(a,b,c)$. For fixed k we do strong induction on b and backwards induction on a. Starting with $b = 0$, we begin the induction on $a = b + 1$:

$$\begin{aligned} g_k(1,0,c) &= g_{k-1}(0,0,0) + 1 + c \cdot w \\ &\le \phi(0,0,0) + (k-1)g + 1 + c \cdot w \\ &= \phi(0,0,0) + 1 + c \cdot w - g + k \cdot g \\ &= \phi(1,0,c) + k \cdot g \end{aligned}$$

and

$$\begin{aligned} g_k(0,0,c) &= p \cdot g_k(1,0,c) + (1-p) g_{k-1}(0,1,c) \\ &\le p(\phi(1,0,c) + k \cdot g) + (1-p)(\phi(0,1,c) + (k-1)g) \\ &= p \cdot \phi(1,0,c) + (1-p)(\phi(0,1,c) - g) + k \cdot g \\ &= \phi(0,0,c) + k \cdot g. \end{aligned}$$

For $b > 0$ the proof works the same, starting from $g_k(b+1,b,c)$ as the base case for a. For $a < b + 1$:

$$\begin{aligned} g_k(a,b,c) &= \max \begin{cases} \max_{s=0,\ldots,b-1} g_k(0,s,1) \\ p g_k(a+1,b,c) + (1-p) g_{k-1}(a,b+1,c) \end{cases} \\ &\le \max \begin{cases} \max_{s=0,\ldots,b-1} \phi(0,s,1) + kg \\ p \cdot (\phi(a+1,b,c) + kg) \\ \quad +(1-p)(\phi(a,b+1,c) + (k-1)g) \end{cases} \\ &= k \cdot g + \max \begin{cases} \max_{s=0,\ldots,b-1} \phi(0,s,1) \\ p \cdot \phi(a+1,b,c) \\ \quad +(1-p)(\phi(a,b+1,c) - g) \end{cases} \\ &= \phi(a,b,c) + kg, \end{aligned}$$

where strong induction was used for the capitulation case. □

Using this lemma, we can find the minimum value of w that leads to Frontier being a best response for different values of p by using binary search on w, checking that the ϕ produced from the LP satisfies (12) and that $g = p + p(1-p)w$. By the definition of $g^\star$:

$$g^\star = \lim_{k\to\infty} \frac{g_k(a,b,c)}{k} \le \lim_{k\to\infty} \frac{\phi(a,b,c) + k \cdot g}{k} = p + p(1-p)w,$$

For $d = 8$, we obtain the graph in fig. 7.

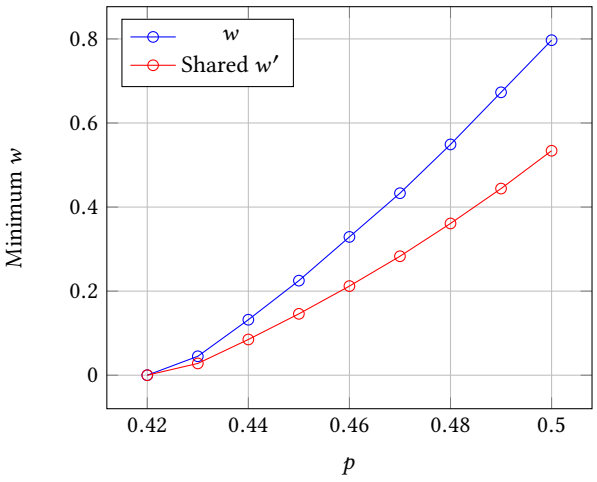


Figure 7: Minimum w for the immediate release case.

It is not a mistake that $w < 1$ for $p = 0.5$. The reason is that d is finite. For $d = \infty$, the strategic miner with $p \geq 0.5$ never has a reason to capitulate, no matter w: the probability that his branch eventually overtakes the honest one is always quite high. However, the probability that he will win the race *within d steps* is much smaller, leading to this result.

If not all honest miners are required to pay forward exactly w, they can share the costs. In particular, one strategy which also makes Miner 1 comply with the protocol is for the first honest miner after Miner 1 to pay forward w' and every other honest miner to pay forward $2w'$, until Miner 1 adds a block and claims the reward for himself, where the process restarts. As we can see from the Figure 7, although the total amount paid forward is larger (as $2w' \geq w$), the maximum contribution of any honest miner is $w' \leq w$. We can verify compliance is indeed a best response for Miner 1 by using a slightly modified LP.

3.2 Pure Nash Equilibria for small miners

So far, we have only considered the best responses of Miner 1, given a society of honest, identical miners and identified the minimum value w needed to ensure his compliance. The natural question is to consider if this also a best response from the other miners. Clearly, when treated as a group this is not their best response. If they do not pay forward any amount, Miner 1 would have an average gain of $g' \geq p$, by mining strategically. By paying forward enough, they need to *increase* his gain to $g^\star \geq g'$, as he can always employ strategic mining while they also pay forward. Therefore, it is more costly for Miner 2 to pay forward w and ensure compliance than to pay forward nothing and allow strategic mining.

We will argue however that under certain conditions paying forward can be a pure Nash equilibrium, if Miner 2 consists of many *small* miners that are treated individually. A miner is considered *small* if his chance of mining a block is negligible. Of course at every step some small miner will mine a block with probability $1 - p$. To see why paying forward is an equilibrium, consider the options of a small miner at the *exact* moment when he mines a block: he can either pay forward and guarantee it is included in the chain or keep everything for himself, but risk getting undercut by Miner 1. Being a small miner, his payoff from mining another block in the future is insignificant compared to the current reward at stake. Formally, the following strategy, called StrictFrontier(w), is part of a PNE:

Definition 3.5 (StrictFrontier). Every miner (except Miner 1) pays forward w and mines at the end of the longest chain that contains blocks with pay forward values either w or 0

Miner 1's strategy is to mine at the frontier without paying forward, as before. Note, that paying forward some value other than w or 0 *cannot* be a best response, since that block will be ignored by the honest miners, who have the majority of the computational power[2]

Theorem 3.6. *For $d = 8$ and any $p \leq 0.44$, there exists w such that Miner 1 playing Frontier and every other small miner playing StrictFrontier(w) is a pure Nash equilibrium.*

[2]The reason StrictFrontier accepts pay forward of 0 is just to incentivize Miner 1 to cooperate. Without it, every block would have the same pay forward, negating any effects this may have and reducing to the classic bitcoin protocol.

Proof sketch. From the perspective of the small miner, he has two opportunities *not* to pay forward: after another small miner or after Miner 1. We can modify the LP from Section 3.1 to contain both these deviations by the small miner. Examining the tightness of the constraints, we can identify the mining states of Miner 1. Specifically, for $p \leq 0.44$, we observe that in both cases, after the small miner adds his block without paying forward, Miner 1 immediately forks the chain behind the small miner and starts mining in parallel. In particular, states $(0, 1, c)$, $(1, 1, c)$, $(1, 2, c)$ and $(2, 2, c)$ become mining states and $(0, 3, c)$, $(1, 3, c)$, $(2, 3, c)$ capitulation states, for $c \in \{0, 1\}$. Essentially he mines while one step behind and capitulates if the honest miners add 2 more blocks. Overall, he overtakes the honest branch with probability at least $p^2 + (1 - p)p^3$.

Therefore, the expected payoff of the small miner is at most $(1 + w)(1 - p^2 - (1 - p)p^3)$, compared to at least $1 - w$ which would have been his payoff had he paid forward w (since in this case he would always keep his mined block). Plugging in the p, w pairs found in Figure 7 we verify that StrictFrontier(w) is indeed a best response. □

It would be very interesting to see if there exists a strategy profile that is a PNE where Miner 1 plays Frontier for $0.45 < p < 0.5$. The analysis seems quite challenging and would likely require different techniques to analyze the emerging stochastic game. Ideally, every small miner comprising Miner 1 would use the same w, but it could be the case that better bounds could be obtained if the pay forward amount use depends on the pay forward received from the previous block.

4 STRATEGIC RELEASE

Similarly to the immediate release case, we identify the maximum hash power p such that if all miners but one follow Frontier(w), the remaining miner's best response is Frontier if his hash power is less than p. As before, it is enough to consider a two player game where Miner 1 is the deviant miner with hash power $p < 0.5$ and Miner 2 represents all small miners, as they follow the same strategy.

The blockchain retains its structure: it still contains a long trunk of blocks followed by two branches, corresponding to the released blocks of Miner 1 and Miner 2. Since Miner 1 could also have more unreleased blocks in his branch the state is now a quadruple (a_r, a, b, c) where a is the number of blocks in Miner 1's branch, of which a_r have been released, and b is the number of blocks in Miner 2's branch. Also, contrary to the immediate release case we can have $a > b + 1$.

Since Miner 2 follows the Frontier(w) strategy, he will capitulate if $a_r \geq b + 1$ and continue mining his branch for $a_r < b + 1$. Therefore, without loss of generality we can assume that at state (a_r, a, b, c) if $a < b + 1$ then $a_r = a$ and if $a \geq b + 1$ then $a_r = b$, otherwise Miner 2 would immediately capitulate and the game would continue at state $(0, a - a_r, 0, 0)$. Therefore, we can encode the states of the mining game with strategic release by the triplet (a, b, c), where $a_r = \min(a, b)$.

As before, we need to recursively define the gain of Miner 1 after the longest branch is extended by k levels. When $a < b + 1$, the actions of Miner 1 are exactly the same as in the immediate release case: he can either capitulate to $(0, s, 1)$ or mine. However,

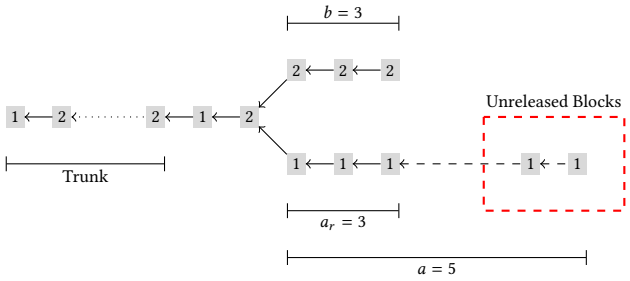


Figure 8: This tree represents state $(3, 5, 3, 1)$. Miner 1 has already mined 5 blocks ahead of the fork, but has only released 3 of them. Miner 2 knows this because the game is complete information. However, he cannot mine there until Miner 1 releases his blocks. Miner 1 can strategically release blocks right when Miner 2's branch is about to become the main one, thus wasting Miner 1's computational power.

for $a \geq b+1$ he can release one more block to force Miner 2 to capitulate and continue at state $(a-b-1, 0, 0)$.

$$\hat{g}_k(a,b,c) = \max \begin{cases} k+b+c\cdot w & \text{if } a \geq k+b \\ \max \begin{cases} \max_{s=0,\ldots,b-1} \hat{g}_k(0,s,1) \\ p\hat{g}_k(a+1,b,c) \\ \quad +(1-p)\hat{g}_{k-1}(a,b+1,c) \\ \hat{g}_{k-1}(a-b-1,0,0) \\ \quad +b+1+c\cdot w \end{cases} \end{cases} \tag{13}$$

where the last term applies for $a \geq b+1$. Equivalently, we can define $g_k(a,0,c) = -\infty$ for all k and $a < 0$. The first term is necessary, as without it the recurrence is ill-defined and Miner 1 can keep mining forever. Once he reached the horizon k he can safely release all his blocks. We define the expected gain per level $\hat{g}^\star$ and potential $\hat{\phi}$ as we did in (2) for the immediate release. Note that the case where $a = k+b$ does not appear, since ϕ is the asymptotic advantage as $k \to \infty$ and a, b are bounded.

$$\hat{\phi}(a,b,c) = \max \begin{cases} \max_{s=0,\ldots,b-1} \hat{\phi}(0,s,1) \\ p\hat{\phi}(a+1,b,c) + (1-p)(\hat{\phi}(a,b+1,c) - \hat{g}^\star) \\ \hat{\phi}(a-b-1,0,0) + b + 1 + c\cdot w - \hat{g}^\star \end{cases}. \tag{14}$$

Releasing blocks only causes Miner 2 to capitulate if $a \geq b+1$, so we set $\phi(a,0,c) = -\infty$ for $a < 0$. Before moving on, we need a useful inequality of the *immediate release* potential ϕ, to use for bounding the advantage of some states relative to others.

Lemma 4.1. *For nonnegative integers a, b and $\ell \in \{0,1\}$ we have that:*

$$\phi(a+\ell, b+\ell, 1) \leq \phi(a,b,0) + (\ell+w)\cdot\left(\frac{p}{1-p}\right)^{b-a+1}. \tag{15}$$

Proof. We first need to establish that for $c \in \{0,1\}$

$$g_k(a+\ell, b+\ell, c) \leq g_k(a,b,c) + \ell\cdot\left(\frac{p}{1-p}\right)^{b-a+1} \tag{16}$$

and

$$g_k(a+\ell, b+\ell, 1) \leq g_k(a,b,0) + (\ell+w)\cdot\left(\frac{p}{1-p}\right)^{b-a+1}. \tag{17}$$

Suppose that from state (a,b) Miner 1 follows the same strategy as state $(a+\ell, b+\ell)$. This is possible, as winning or capitulating depends only on the difference $b-a$. Let $\bar{g}_k(a,b,c)$ be the gain from playing this suboptimal strategy and $\bar{r}(a,b)$ the probability of winning from state (a,b). Clearly, we have:

$$g_k(a,b,c) \geq \bar{g}_k(a,b,c) = g_k(a+\ell, b+\ell, c) - \ell\cdot\bar{r}(a,b). \tag{18}$$

Given that $\bar{r}(a,b) \leq (p/(1-p))^{b-a+1}$ from [8, Lemma 1] we get the first inequality.

For the second, let $r(a,b)$ be the probability of winning from (a,b) with the optimal strategy. As before:

$$g_k(a,b,1) = g_k(a,b,0) + w\cdot r(a,b) \leq g_k(a,b,0) + w\cdot\left(\frac{p}{1-p}\right)^{b-a+1}, \tag{19}$$

which we substitute in (16).

To complete the proof, we take limits in (16) and use the definition of ϕ. □

In [8] it was shown that for $p \leq 0.361$ Frontier is a best response, therefore (for $w=0$) there exists a ϕ such that for $g^\star = p$ (12) is satisfied, meaning that honest mining maximizes the fraction of blocks added by Miner 1. By Lemma 3.3, increasing w can only strengthen this result, therefore for $p \leq 0.361$ and any $w \leq 1$ there exists ϕ that satisfies (12) for $g^\star = p + p(1-p)w$. Using this, we extend the potential to states (a,b,c) with $a > b+1$:

$$\bar{\phi}(a,b,c) = \begin{cases} \phi(a,b,c) & \text{if } a \leq b+1 \\ a\lambda + b\mu + \kappa + c\cdot w & \text{otherwise} \end{cases}, \tag{20}$$

where $\lambda = \frac{(p-1)^2(1-pw)}{1-2p}$, $\mu = \frac{p(p-(p-1)^2 w)}{2p-1}$ and $\kappa = \frac{(p-1)p(pw-1)}{2p-1}$. The constants have been selected so that $\bar{\phi}(a,b,c) = p\bar{\phi}(a+1,b,c) + (1-p)(\bar{\phi}(a,b+1,c) - \hat{g}^\star)$ and $\bar{\phi}(b+1,b,c) = b+1+c\cdot w - \hat{g}^\star = \phi(b+1,b,c)$, for $\hat{g}^\star = p + p(1-p)w$ which is the honest gain. Doing this, we can use known results about ϕ for smaller states while having a tight enough closed form of $\hat{\phi}$ on states which are unlikely to be reached.

Following the notation of [8] we define $\bar{\phi}_M$ for when Miner 1 continues to mine, $\bar{\phi}_R$ when Miner 1 releases some blocks and $\bar{\phi}_C$ when he capitulates.

$$\begin{aligned} \bar{\phi}_M(a,b,c) &= p\cdot\bar{\phi}(a+1,b,c) + (1-p)(\bar{\phi}(a,b+1,c) - \hat{g}^\star) \\ \bar{\phi}_R(a,b,c) &= \bar{\phi}(a-b-1,0,0) + b + 1 + c\cdot w - \hat{g}^\star \\ \bar{\phi}_C(a,b,c) &= \max_{s=0,\ldots,b-1} \bar{\phi}(0,s,1). \end{aligned}$$

Theorem 4.2. *For every $p < 0.344$, there exists $w \geq 0$ large enough so that if every miner but one follows Frontier(w), the best response of the remaining miner with hash power p is Frontier.*

Proof. We need to show that $\bar{\phi}$ is a valid potential and satisfies (14) for $\hat{g}^\star = p + p(1-p)w$, or equivalently:

$$\bar{\phi}(a,b,c) = \max\{\bar{\phi}_M(a,b,c), \bar{\phi}_R(a,b,c), \bar{\phi}_C(a,b,c)\}$$

□

Lemma 4.3. *The potential $\bar{\phi}$ and $\hat{g}^\star = p + p(1-p)w$ satisfy recurrence (14) for $p \leq 0.344$ (root of the polynomial $-p^4 + 3p^3 - 7p^2 + 5p - 1$) and some $w < 1$.*

Claim 1. *For states* (a, b, c) *with* $a < b + 1$*:*

$$\bar{\phi}(a, b, c) = \phi(a, b, c) = \max\{\bar{\phi}_M(a, b, c), \bar{\phi}_R(a, b, c), \bar{\phi}_C(a, b, c)\}.$$

In this case $\bar{\phi}_R(a, b, c) = -\infty$ and therefore $\bar{\phi}(a, b, c) = \phi(a, b, c)$ which satisfies (12) and (14) by definition, as releasing is only possible for $a \geq b + 1$.

Claim 2. *For states* (a, b, c) *with* $a > b + 1$*:*

$$\bar{\phi}(a, b, c) = \bar{\phi}_M(a, b, c) = \max\{\bar{\phi}_M(a, b, c), \bar{\phi}_R(a, b, c), \bar{\phi}_C(a, b, c)\}.$$

By definition, $\bar{\phi}(a, b, c) = \bar{\phi}_M(a, b, c)$. We have:

$$\bar{\phi}(a, b, c) - \bar{\phi}_R(a, b, c) = \frac{b(2 - 4p) + (1 - p)(2 - 3p)(1 - pw)}{1 - 2p} > 0.$$

Since $p \leq 0.344$ we know that $\phi(a, b, c)$ corresponds to the potential of the honest mining strategy, in the immediate release case. Therefore:

$$\begin{aligned}\bar{\phi}_C(a, b, c) &= \max_{s=0,\ldots,b-1} \bar{\phi}(0, s, 1) \\ &= \max_{s=0,\ldots,b-1} \phi(0, s, 1) = \phi(0, 0, 1) \leq w \cdot \frac{p}{1 - p}\end{aligned}$$

as $(0, 0, 1)$ and $(0, 0, 0)$ are the only mining states and by using Lemma 4.1for the last inequality. Also:

$$\begin{aligned}\bar{\phi}_R(a, b, c) &= \bar{\phi}(a - b - 1, 0, 0) + b + 1 + c \cdot w - \hat{g}^\star \\ &\geq 1 - p(1 - p)w \\ &\geq w \cdot \frac{p}{1 - p} \\ &\geq \bar{\phi}_C(a, b, c),\end{aligned}$$

for $p < 0.344$.

Claim 3. *For states* $(b + 1, b, c)$*:*

$$\begin{aligned}\bar{\phi}(b + 1, b, c) &= \bar{\phi}_R(b + 1, b, c) \\ &= \max\{\bar{\phi}_M(b + 1, b, c), \bar{\phi}_M(b + 1, b, c), \bar{\phi}_C(b + 1, b, c)\}.\end{aligned}$$

Exactly as before, we have $\bar{\phi}_R(b + 1, b, c) = b + 1 + c \cdot w - \hat{g}^\star \geq \bar{\phi}_C(b + 1, b, c)$. For $\bar{\phi}_M(b + 1, b, c)$:

$$\begin{aligned}\bar{\phi}_M(b + 1, b, c) &= p\bar{\phi}(b + 2, b, c) + (1 - p)\left(\bar{\phi}(b + 1, b + 1, c) - g\right) \\ &\leq p((b + 2)\lambda + b\mu + \kappa + c \cdot w) \\ &\quad + (1 - p)\left((b + 1 + c \cdot w)\frac{p}{1 - p} - g\right),\end{aligned}$$

by the definition of $\bar{\phi}$ and Lemma 4.1. Then:

$$\begin{aligned}&\bar{\phi}_R(a, b, c) - \bar{\phi}_M(a, b, c) \geq \\ &\quad b(1 - 2p)^2 + (p - 1)\left(p^3 w - p^2(w + 1) + p(2c \cdot w + 5) - c \cdot w\right) \\ &\quad + 1.\end{aligned}$$

Since $b \geq 0$ we only need:

$$(p - 1)\left(p^3 w - p^2(w + 1) + p(2c \cdot w + 5) - c \cdot w\right) + 1 \geq 0.$$

Setting $w = 1$, it holds for $p \leq 0.344$ for $c = 0$ and $p \leq 0.442$ for $c = 1$. For smaller values of p it is not necessary to set w to it's highest value.

Now, we can use the equivalent of Lemma 3.4(whose proof has very minor differences) for the strategic release case to get that:

$$\begin{aligned}\hat{g}^\star = \lim_{k\to\infty} \frac{\hat{g}_k(a, b, c)}{k} &\leq \lim_{k\to\infty} \frac{\bar{\phi}(a, b, c) + k \cdot (p + p(1 - p)w)}{k} \\ &= p + p(1 - p)w.\end{aligned}$$

This shows that his gain is at most what he would get by playing Frontier, hence a best response.

Through a procedure similar to Section 3.1, adjusting the LP for this case, we obtain the following graph for $d = 8$ for the minimum value of w required.

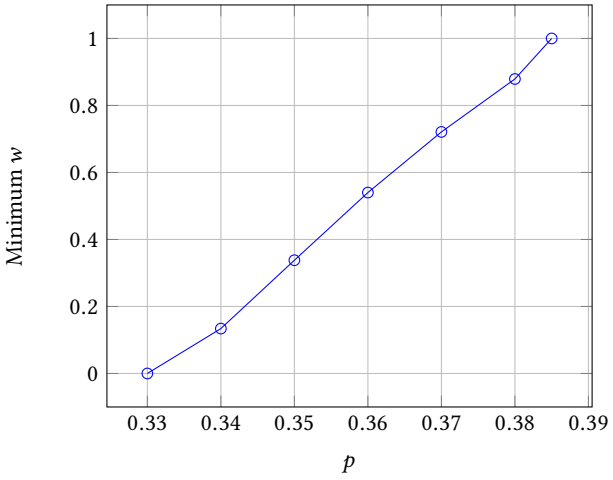


Figure 9: Minimum values of w for the strategic release case.

Contrary to Figure 7, for $p = 0.385$ we have $w \approx 1$. As $d \to \infty$, we reach $w = 1$ for $p \approx 0.38$.

ACKNOWLEDGMENTS

This work was generously supported by the ERC Advanced Grant 321171 (ALGAME).

REFERENCES

[1] Moshe Babaioff, Shahar Dobzinski, Sigal Oren, and Aviv Zohar. 2012. On Bitcoin and Red Balloons. In *Proceedings of the 13th ACM Conference on Electronic Commerce (EC '12)*. ACM, New York, NY, USA, 56–73. https://doi.org/10.1145/2229012.2229022

[2] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. 2015. SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy*. 104–121. https://doi.org/10.1109/SP.2015.14

[3] Miles Carlsten, Harry Kalodner, S Matthew Weinberg, and Arvind Narayanan. 2016. On the Instability of Bitcoin Without the Block Reward. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 154–167.

[4] Nicolas T. Courtois and Lear Bahack. 2014. On Subversive Miner Strategies and Block Withholding Attack in Bitcoin Digital Currency. *CoRR* abs/1402.1718 (2014). arXiv:1402.1718 http://arxiv.org/abs/1402.1718

[5] I. Eyal. 2015. The Miner's Dilemma. In *2015 IEEE Symposium on Security and Privacy*. 89–103. https://doi.org/10.1109/SP.2015.13

[6] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert Van Renesse. 2016. Bitcoin-NG: A Scalable Blockchain Protocol.. In *NSDI*. 45–59.

[7] Ittay Eyal and Emin Gün Sirer. 2018. Majority is Not Enough: Bitcoin Mining is Vulnerable. *Commun. ACM* 61, 7 (June 2018), 95–102. https://doi.org/10.1145/3212998

[8] Aggelos Kiayias, Elias Koutsoupias, Maria Kyropoulou, and Yiannis Tselekounis. 2016. Blockchain Mining Games. In *Proceedings of the 2016 ACM Conference on Economics and Computation (EC '16)*. ACM, New York, NY, USA, 365–382. https://doi.org/10.1145/2940716.2940773

[9] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. 2017. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. In *Advances in Cryptology – CRYPTO 2017*, Jonathan Katz and Hovav Shacham (Eds.). Springer International Publishing, Cham, 357–388.

[10] Joshua A Kroll, Ian C Davey, and Edward W Felten. 2013. The economics of Bitcoin mining, or Bitcoin in the presence of adversaries. In *Proceedings of WEIS*, Vol. 2013. 11.

[11] Yoad Lewenberg, Yoram Bachrach, Yonatan Sompolinsky, Aviv Zohar, and Jeffrey S. Rosenschein. 2015. Bitcoin Mining Pools: A Cooperative Game Theoretic Analysis. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 919–927. http://dl.acm.org/citation.cfm?id=2772879.2773270

[12] Satoshi Nakamoto. [n. d.]. Bitcoin: A peer-to-peer electronic cash system,âĂİ http://bitcoin.org/bitcoin.pdf.

[13] Joseph Poon and Thaddeus Dryja. 2016. The bitcoin lightning network: Scalable off-chain instant payments. *draft version 0.5* 9 (2016), 14.

[14] Meni Rosenfeld. 2011. Analysis of Bitcoin Pooled Mining Reward Systems. *CoRR* abs/1112.4980 (2011). arXiv:1112.4980 http://arxiv.org/abs/1112.4980

[15] Meni Rosenfeld. 2014. Analysis of hashrate-based double spending. *arXiv preprint arXiv:1402.2009* (2014).

[16] Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. 2016. Optimal selfish mining strategies in bitcoin. In *International Conference on Financial Cryptography and Data Security*. Springer, 515–532.

[17] Yonatan Sompolinsky, Yoad Lewenberg, and Aviv Zohar. 2016. SPECTRE: A Fast and Scalable Cryptocurrency Protocol. *IACR Cryptology ePrint Archive* 2016 (2016), 1159.

[18] Yonatan Sompolinsky and Aviv Zohar. 2015. Secure high-rate transaction processing in bitcoin. In *International Conference on Financial Cryptography and Data Security*. Springer, 507–527.

[19] Yonatan Sompolinsky and Aviv Zohar. 2018. PHANTOM: A Scalable BlockDAG Protocol. *IACR Cryptology ePrint Archive* 2018 (2018), 104.

[20] F. Tschorsch and B. Scheuermann. 2016. Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies. *IEEE Communications Surveys Tutorials* 18, 3 (thirdquarter 2016), 2084–2123. https://doi.org/10.1109/COMST.2016.2535718

ContraVis: Contrastive and Visual Topic Modeling for Comparing Document Collections

Tuan V. M. Le
H. John Heinz III College
Carnegie Mellon University
tuanminhlv@yahoo.com

Leman Akoglu
H. John Heinz III College
Carnegie Mellon University
lakoglu@andrew.cmu.edu

ABSTRACT

Given posts on 'abortion' and posts on 'religion' from a political forum, how can we find topics that are discriminative and those in common? In general, (1) how can we *compare and contrast* two or more different ('labeled') document collections? Moreover, (2) how can we *visualize* the data (in 2-d or 3-d) to best reflect the similarities and differences between the collections?

We introduce (to the best of our knowledge) the first *contrastive and visual* topic model, called ContraVis, that jointly addresses both problems: (1) contrastive topic modeling, and (2) contrastive visualization. That is, ContraVis learns not only latent topics but also embeddings for the documents, topics and labels for visualization. ContraVis exhibits three key properties by design. It is (*i*) **Contrastive:** It enables comparative analysis of different document corpora by extracting latent discriminative and common topics across labeled documents; (*ii*) **Visually-expressive:** Different from numerous existing models, it also produces a visualization for all of the documents, labels, and the extracted topics, where proximity in the coordinate space is reflective of proximity in semantic space; (*iii*) **Unified:** It extracts topics and visual coordinates *simultaneously* under a joint model.

Through extensive experiments on real-world datasets, we show ContraVis 's potential for providing visual contrastive analysis of multiple document collections. We show *both qualitatively and quantitatively* that ContraVis significantly outperforms both unsupervised and supervised state-of-the-art topic models in contrastive power, semantic coherence and visual effectiveness.

CCS CONCEPTS

• **Computing methodologies** → **Topic modeling**; • **Mathematics of computing** → *Dimensionality reduction.*

KEYWORDS

contrastive topic models; visualization; comparative text mining

ACM Reference Format:
Tuan V. M. Le and Leman Akoglu. 2019. ContraVis: Contrastive and Visual Topic Modeling for Comparing Document Collections. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313617

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313617

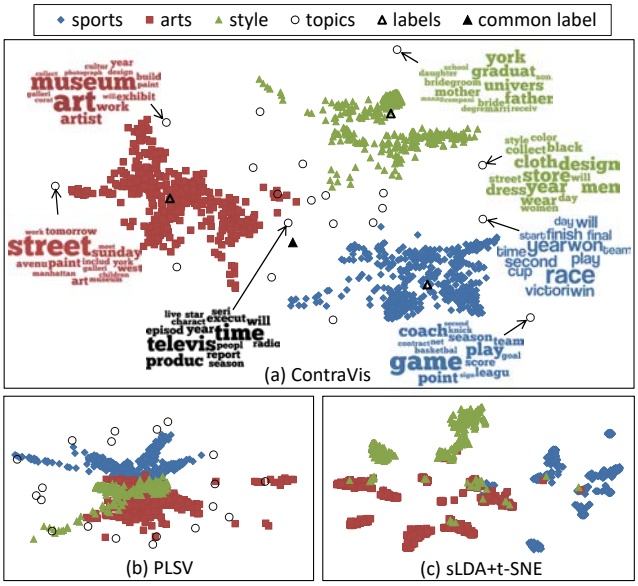


Figure 1: (best in color) Contrastive analysis of `sports`, `arts`, `style` documents from NYTnews (K=20 topics); (top) ContraVis (supervised joint method) not only finds latent discriminative topics but also gives visual embedding of documents and topics. Two discriminative topics per label shown in wordclouds (w/ respective colors) + one common topic (black); (bottom) *PLSV* (unsupervised joint method) gives visual embedding of documents and topics. However, the topics are not as discriminative, indicated by a mixing of labels. *sLDA*+tSNE (supervised topic modeling followed by embedding, document-only) has a mixing of labels and gives no means to visually explore topics. (See details in §5)

1 INTRODUCTION

Topic modeling is widely used for identifying latent themes (=topics) in a large collection (or corpus) of documents. Topics are simply groups of words that best represent the information in the corpus. As such, they aid in sense-making by characterizing the corpus at a high level, and can also be used for several downstream tasks like indexing, search and classification.

Vast majority of topic models are designed to extract topics from a *single* corpus. In many cases, on the other hand, one is interested in comparing and contrasting documents from *multiple* corpora. These corpora could be from different time periods, e.g. science articles 1960-80 vs. 1980-2000; spatial locations, e.g. news articles from 'Asia' vs. 'USA' vs. 'Europe'; or different sub-populations, e.g. essays on neuroscience vs. psychology.

Our Goal. In this work, we focus on the latter area, broadly known as comparative text mining [28]. Specifically, we address the *cross-collection modeling and visualization* problems, motivated by the following questions. How can we find hidden topics from *two or more* document collections? What are the *common and discriminative* topics among these collections? Further, how can we *visualize* the documents and topics to easily explore, compare and contrast the collections? In this paper, we refer to the task of finding common and discriminative topics as contrastive topic modeling as these topics help to contrast document collections. In addition, the visualization produced by embedding document collections and their contrastive topics is referred to as contrastive visualization. Ideally, a contrastive visualization can help users to easily spot the discriminative and common topics. In this work, we are interested in scatterplot visualization where documents and topics are embedded in a 2-d or 3-d visualization space. §6 provides a brief review of other visualization forms for visual comparison.

Previous Approaches. Topic models like *LDA* [2] and its variants can be used to find hidden topics from document collections. However, since these models are unsupervised, the extracted topics may not be discriminative. For contrastive topic modeling, we can leverage the labels of the document collections to employ supervised topic models [9, 16, 20, 21, 30]. Although the learned topics may be discriminative, these models are not designed for visualization. One can perform a post-hoc embedding using dimensionality reduction methods such as t-SNE [14] for visualization. This two-phase approach, however, has two different objective functions (one for topic modeling and one for visualization) that are optimized separately, which may result in poor visualization. Recently, new models that follow a joint approach have been developed; tying together the two steps using a single objective function [7, 10–12]. This type of unified approach is pioneered by *PLSV* [7], which jointly models topics and visualization by locating documents and topics in the same visualization space. However, *PLSV* is unsupervised and therefore the extracted topics and visualization may not be discriminative for contrastive analysis. We compare to these previous approaches through extensive experiments in §4 and §5.

Our Model. We introduce a supervised joint technique called ContraVis that addresses both contrastive topic modeling and contrastive visualization. It *simultaneously* learns (a) latent topics among multiple corpora for sensemaking and (b) 2-d or 3-d embeddings for documents, identified topics, and labels (where each corpus is represented with a unique label) for visual analysis. The topics capture both the common themes across the corpora as well as discriminative themes specific to each corpus/label. Moreover, the embeddings in the visual coordinate system reflect the semantic proximities between (*i*) the documents and topics and (*ii*) the topics and labels. To the best of our knowledge, ContraVis is the *first supervised joint technique for simultaneous contrastive topic modeling and visual embedding*.

Example. To demonstrate ContraVis's novel aspects, we show Figure 1 on an example real-world dataset from NYTnews (see Section 4.1) where we contrast three labeled corpora: `sports`, `arts` and `style`. In (b) and (c), we use unsupervised and supervised topic models, *PLSV* [7] and *sLDA* [16] respectively, to extract topics and topic probability distributions of the labeled documents. For *sLDA*, we also perform a post-hoc embedding of the documents to 2-d based on their topic representations using t-SNE [14]. In contrast, we show the output of our ContraVis in (a). Besides extracting topics, ContraVis readily produces a visual embedding (i.e., no post-hoc embedding is necessary). Moreover, it embeds not only the documents but also the labels (triangles) and the topics (circles). This provides a *holistic* view of the data. In (a), one can see the top few representative words or wordclouds per topic, like other models. What is more: (1) one can also see how documents relate to different topics and visually infer each document's probability distribution over topics, and (2) how topics relate to different labels (=corpora) which facilitates identifying the common and discriminative ones.

To summarize, our model exhibits the following key properties, which constitute the main contributions of our work.

- **Contrastive power:** We propose a new topic model called ContraVis for the comparative analysis of multiple document corpora. It quantifies its identified topics in terms of their relevance to different corpora, revealing common and discriminative ones.
- **Visual-expressiveness:** ContraVis produces embeddings, to visualize (all of) documents, labels, and topics in 2- or 3-d. Proximity in embedding space is reflective of proximity in semantic space, which helps easily infer (*i*) most dominant topics per document and (*ii*) most relevant/discriminative topics per label.
- **Unified nature:** ContraVis estimates the topics and the embedding coordinates simultaneously within a single, joint model. Thus, it is a more holistic model for comparative text analysis than existing models that require post-hoc embeddings.

We evaluate the semantic, contrastive, and visualization quality of ContraVis both qualitatively and quantitatively on diverse real-world datasets. We show the superiority of ContraVis and its potential as a method for visual contrastive analysis of multiple document collections, from diverse domains and from different sub-populations, space, or time.

Reproducibility: We share the source code for ContraVis and all of our public-domain datasets at https://github.com/tuanlvm/ContraVis.

2 PROBLEM DEFINITIONS

In this problem we aim to build a model to compare and contrast documents across different corpora, e.g. three collections of posts respectively on 'gun control', 'taxes', and 'military'. Documents belonging to a specific corpus can be labeled by its name. As such, the input is a collection of single-label documents $\mathcal{D} = \{(d_1, s_1), \ldots, (d_N, s_N)\}$, where $s_n \in \mathcal{L}$ is the label of document d_n and $\mathcal{L}$ is the set of unique labels from a finite domain. For output, our model seeks to produce a comparison that is interpretable; by inferring common topics and discriminative ones for each label. In addition, for an effective exploratory and contrastive analysis, we propose to use an visualization where documents, topics and labels are embedded in the same space such that the visual proximities between them reflect their semantic similarities and differences. Given these objectives, we give our problems formally.

Table 1: Notation used in text.

	Notation	Description
Input Corpus	$\mathcal{D}$	document corpus
	N	number of documents, $\|\mathcal{D}\| = N$
	$\mathcal{V}$	vocabulary, $\|\mathcal{V}\| = W$
	d_n	a specific document
	M_n	number of words in document d_n
	$\mathcal{L}$	set of unique labels, $\|\mathcal{L}\| = L$
	s_n	single label of document d_n
	c	common label, which is applied to all documents
	Λ_n	$\Lambda_n := \{s_n, c\}$
Topics	K	total number of topics
	z	a specific latent topic
	β_z	word distribution of topic z
	$\boldsymbol{\beta}$	collection of β_z's for all topics
Visualization	d	visualization dimension (2 or 3)
	x_n	latent coordinate of doc. d_n in visualization space
	ϕ_z	latent coordinate of topic z in visualization space
	μ_l	latent coordinate of label l in visualization space
	$\boldsymbol{\chi}$	collection of x_n's for all documents
	$\boldsymbol{\Phi}$	collection of ϕ_z's for all topics
	$\boldsymbol{\mu}$	collection of μ_l for all labels

Given *a collection of single-label documents $\mathcal{D}$ containing words from a finite vocabulary $\mathcal{V}$ and labels from set $\mathcal{L}$, number of topics K, and visualization dimension d;*

Problem 1 (Contrastive Topic Modeling). **Find**
(i) K latent topics, and word probability distributions of topics,
(ii) topic distributions of documents, and
(iii) topic and word distributions of labels.

Problem 2 (Contrastive Document Visualization). **Find** *d-dimensional visualization coordinates for (i) N documents, (ii) K topics, and (iii) L labels,*

such that *the spatial proximities between documents, topics, and labels are reflective of (or proportional to) the topic-document and topic-label probability distributions above.*

In our formulation, we introduce a new label c as a common label, which is applied to all documents. Hereafter, we denote $\mathcal{L} := \{\mathcal{L}, c\}$ and $\Lambda_n := \{s_n, c\}$ as the set of labels for each d_n. We use label c to capture the common topics among documents. Note that we infer K latent topics overall and find the probability distribution of these topics over different labels. This provides a "soft assignment" of topics to labels, from which we can deduce the relevance of each topic to each label, as well as mutual and common topics across labels. This reduces the parameter complexity of our model; as compared to finding K_l mutual topics per label $l \in \mathcal{L}$ and K_c common topics, which would require $L + 1$ hyperparameters to choose, we only set K (details in §3.2.2).

3 PROPOSED CONTRAVIS MODEL

We introduce ContraVis, a new, visually-expressive contrastive topic model that jointly addresses Problem 1 and Problem 2. Specifically, ContraVis introduces an innovative generative process that ties together the probability distributions with the visualization coordinates, as we describe next.

3.1 Generative Process

Besides inferring topic-word and document-topic probability distributions, our additional objective is to learn latent visualization coordinates x_n of each document d_n and ϕ_z for each topic. In addition, each label l is also assumed to have a latent coordinate μ_l in the same visualization space. Let $\boldsymbol{\mu} = \{\mu_l\}_1^L$. The label distribution of d_n is expressed by its Euclidean distances to labels as follows:

$$\mathrm{P}(l|x_n, \boldsymbol{\mu}) = \frac{\exp(-\frac{1}{2}||x_n - \mu_l||^2)}{\sum_{l'=1}^{L} \exp(-\frac{1}{2}||x_n - \mu_{l'}||^2)} \quad (1)$$

According to Eq. (1), $\mathrm{P}(l|x_n, \boldsymbol{\mu})$ is high when x_n is close to μ_l. Since we observe the label s_n of document d_n, d_n should be placed close to μ_{s_n} in the visualization space so that $\mathrm{P}(s_n|x_n, \boldsymbol{\mu})$ is high. Note that each document also has a probability $\mathrm{P}(c|x_n, \boldsymbol{\mu})$ of belonging to the common label c. By introducing the common label, we can model the overlap among labels while still aiming to separate them for extracting discriminative topics.

For each document, we introduce a label-dependent topic distribution $\mathrm{P}(z|l, x_n, \boldsymbol{\Phi})$, which is determined by:

$$\mathrm{P}(z|l, x_n, \boldsymbol{\Phi}) = \frac{\exp(-\frac{1}{2}||\mu_l - \phi_z||^2)\exp(-\frac{1}{2}||x_n - \phi_z||^2)}{\sum_{z'=1}^{K} \exp(-\frac{1}{2}||\mu_l - \phi_{z'}||^2)\exp(-\frac{1}{2}||x_n - \phi_{z'}||^2)} \quad (2)$$

We can see that for a label l, x_n has high $\mathrm{P}(z|l, x_n, \boldsymbol{\Phi})$ when it is close to topic ϕ_z and ϕ_z is close to label μ_l. Therefore, as x_n is to be placed close to its observed label s_n, topics that best describe d_n and are discriminative for s_n are also to be placed close to s_n.

The steps of the generative process of ContraVis are as follows:

(1) For each label $l \in \mathcal{L}$ (including common label c):
  (a) Draw l's coordinate: $\mu_l \sim \mathrm{Normal}(0, \sigma_0^{-1} I)$
(2) For each topic $z = 1, \ldots, K$:
  (a) Draw z's word distribution: $\beta_z \sim \mathrm{Dirichlet}(\lambda)$
  (b) Draw z's coordinate: $\phi_z \sim \mathrm{Normal}(0, \varphi^{-1} I)$
(3) For each document d_n, where $n = 1, \ldots, N$:
  (a) Draw d_n's coordinate: $x_n \sim \mathrm{Normal}(\mu_{s_n}, \gamma^{-1} I)$
  (b) For each word $w_{nm} \in d_n$:
    (i) Draw a label: $l \sim \mathrm{Multi}(\{\mathrm{P}(l|x_n, \boldsymbol{\mu})\}_{l=1}^{L})$
    (ii) Draw a topic: $z \sim \mathrm{Multi}(\{\mathrm{P}(z|l, x_n, \boldsymbol{\Phi})\}_{z=1}^{K})$
    (iii) Draw a word: $w_{nm} \sim \mathrm{Multi}(\beta_z)$
  (c) Set Λ_n = unique labels of w_{nm}

Above, I is the $d \times d$ identity matrix. σ_0, φ, and γ are hyperparameters that control the variance of the Normal distributions. Similar to LDA [2], ContraVis associates each topic with a word probability distribution β_z where $\mathrm{P}(\beta_z) = \frac{\Gamma\left((\lambda+1)W\right)}{\Gamma(\lambda+1)^W} \prod_{w=1}^{W} \beta_{zw}^{\lambda}$ and λ is the Dirichlet prior. Coordinates of topics and labels are drawn from spherical Normal distributions with mean at zero. For each document, we assume that its coordinate is drawn from Normal distributions with mean at μ_{s_n}. Note that this does not ensure that its generated label will be s_n. For each word w_{nm} in document d_n, first we draw a label l and draw a topic z from $\mathrm{P}(z|l, x_n, \boldsymbol{\Phi})$, w_{nm} is then drawn from multinomial β_z. Finally, labels of d_n are unique labels of d_n's words. Figure 2 illustrates the corresponding graphical model for the generative process of our model.

3.2 ContraVis Inference

The parameters of ContraVis include the word distributions of topics $\boldsymbol{\beta} = \{\beta_z\}_{z=1}^{K}$, document coordinates $\boldsymbol{\chi} = \{x_n\}_{n=1}^{N}$, topic coordinates $\boldsymbol{\Phi} = \{\phi_z\}_{z=1}^{K}$, and label coordinates $\boldsymbol{\mu} = \{\mu_l\}_{l=1}^{L}$ (including common label c). Let $\Omega = \langle \boldsymbol{\beta}, \boldsymbol{\chi}, \boldsymbol{\Phi}, \boldsymbol{\mu} \rangle$. In addition to the model

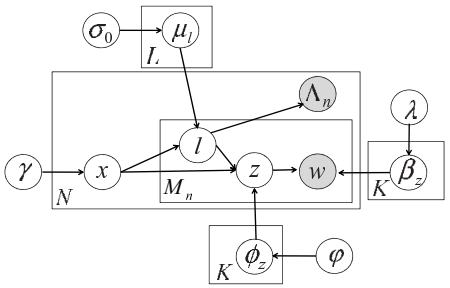


Figure 2: Graphical model representation of ContraVis's generative process (See §3.1).

parameters, the label and topic assignment of words in the documents are unobserved latent variables (See (3bi) and (3bii) of the generative process in §3.1). Therefore, we use the EM algorithm to infer Ω from $\mathcal{D}$ based on maximum a posteriori estimation. Data likelihood can be written as follows (hyperparameters omitted for simplicity).

$$\begin{aligned}\mathrm{P}(\mathcal{D}|\Omega) &= \sum_{\mathcal{L}_\mathcal{W}}\sum_{\mathcal{Z}_\mathcal{W}} \mathrm{P}(\mathcal{D},\, \mathcal{L}_\mathcal{W},\, \mathcal{Z}_\mathcal{W}|\Omega) = \prod_{n=1}^{N} \mathrm{P}(\Lambda_n)\mathrm{P}(d_n|x_n,\, \boldsymbol{\mu},\, \boldsymbol{\Phi},\, \boldsymbol{\beta}) \\ &= \prod_{n=1}^{N}\Big[\mathrm{P}(\Lambda_n)\prod_{m=1}^{M_n}\sum_{l=1}^{L}\sum_{z=1}^{K}\Big[\mathrm{P}(l|x_n,\, \boldsymbol{\mu})\mathrm{P}(z|l,\, x_n,\, \boldsymbol{\Phi})\mathrm{P}(w_{nm}|\beta_z)\Big]\Big]\end{aligned} \tag{3}$$

where $\mathcal{L}_\mathcal{W}, \mathcal{Z}_\mathcal{W}$ are respectively label assignments and topic assignments to all the words $\mathcal{W}$ in $\mathcal{D}$, and

$$\mathrm{P}(\Lambda_n) = \prod_{m=1}^{M_n}\sum_{l\in\Lambda_n}\mathrm{P}(l|x_n,\, \boldsymbol{\mu}) \tag{4}$$

The EM algorithm seeks to find the MLE of the marginal likelihood by iteratively applying E and M steps that we outline below.

E step: The conditional expectation of the data loglikelihood with priors under the current estimate of the parameters $\hat{\Omega}$ is

$$\begin{aligned}C(\Omega|\hat{\Omega}) &= \mathrm{E}_{(\mathcal{L}_\mathcal{W},\mathcal{Z}_\mathcal{W}|\mathcal{D},\hat{\Omega})}[\log \mathrm{P}(\mathcal{D},\, \mathcal{L}_\mathcal{W},\, \mathcal{Z}_\mathcal{W}|\Omega)] \\ &= \sum_{n=1}^{N}\sum_{m=1}^{M_n} log \sum_{l\in\Lambda_n}\mathrm{P}(l|x_n,\, \boldsymbol{\mu}) \\ &+ \sum_{n=1}^{N}\sum_{m=1}^{M_n}\sum_{l=1}^{L}\sum_{z=1}^{K}\mathrm{P}(l,\, z|n,\, m,\, \hat{\Omega})\log\Big[\mathrm{P}(l|x_n,\, \boldsymbol{\mu})\mathrm{P}(z|l,\, x_n,\, \boldsymbol{\Phi})\mathrm{P}(w_{nm}|\beta_z)\Big] \\ &+ \sum_{n=1}^{N}\log\mathrm{P}(x_n) + \sum_{z=1}^{K}\log\mathrm{P}(\phi_z) + \sum_{z=1}^{K}\log\mathrm{P}(\beta_z) + \sum_{l=1}^{L}\log\mathrm{P}(\mu_l)\end{aligned} \tag{5}$$

where

$$\mathrm{P}(x_n) = \Big(\frac{\gamma}{2\pi}\Big)^{\frac{D}{2}}\exp\Big(-\frac{\gamma}{2}\parallel x_n - \mu_{s_n}\parallel^2\Big), \tag{6}$$

$$\mathrm{P}(\phi_z) = \Big(\frac{\varphi}{2\pi}\Big)^{\frac{D}{2}}\exp\Big(-\frac{\varphi}{2}\parallel \phi_z\parallel^2\Big), \tag{7}$$

$$\mathrm{P}(\mu_l) = \Big(\frac{\sigma_0}{2\pi}\Big)^{\frac{D}{2}}\exp\Big(-\frac{\sigma_0}{2}\parallel \mu_l\parallel^2\Big). \tag{8}$$

We set hyper-parameters $\varphi = 0.1N$ as used in *PLSV* [7], $\gamma = L^2$ and $\sigma_0 = 0.1N$ which work well in practice. The conditional distribution of label l and topic z given the m'th word in the n'th document under current estimate of the parameters is given as

$$\mathrm{P}(l,\, z|n,\, m,\, \hat{\Omega}) = \frac{\mathrm{P}(l|\hat{x}_n)\mathrm{P}(z|l,\, x_n,\, \hat{\boldsymbol{\Phi}})\mathrm{P}(w_{nm}|\hat{\beta}_z)}{\sum_{l'=1}^{L}\sum_{z'=1}^{K}\mathrm{P}(l'|\hat{x}_n)\mathrm{P}(z'|l',\, x_n,\, \hat{\boldsymbol{\Phi}})\mathrm{P}(w_{nm}|\hat{\beta}_{z'})}. \tag{9}$$

M step: We update the entries of each β_z as follows:

$$\hat{\beta}_{zw} = \frac{\sum_{n=1}^{N}\sum_{m=1}^{M_n}\mathbb{1}(w_{nm}=w)\mathrm{P}(z|n,\, m,\, \hat{\Omega}) + \lambda}{\sum_{w'=1}^{W}\sum_{n=1}^{N}\sum_{m=1}^{M_n}\mathbb{1}(w_{nm}=w')\mathrm{P}(z|n,\, m,\, \hat{\Omega}) + \lambda W} \tag{10}$$

where $\mathbb{1}(X) = 1$ if X is true and 0 otherwise. As for the other parameters, x_n, ϕ_z, μ_l's, we do not have closed form solutions. We update them using the quasi-Newton method L-BFGS [17] with the following gradients of $C(\Omega|\hat{\Omega})$ w.r.t x_n, ϕ_z, μ_l.

$$\begin{aligned}\frac{\partial C(\Omega|\hat{\Omega},\, g)}{\partial \phi_z} &= \sum_{n=1}^{N}\sum_{m=1}^{M_n}\sum_{l=1}^{L}\Big(\mathrm{P}(l|n,\, m,\, \hat{\Omega})\mathrm{P}(z|l,\, x_n,\, \boldsymbol{\Phi}) - \mathrm{P}(l,\, z|n,\, m,\, \hat{\Omega})\Big)(2\phi_z - \mu_l - x_n) \\ &\quad - \varphi\phi_z\end{aligned} \tag{11}$$

$$\begin{aligned}\frac{\partial C(\Omega|\hat{\Omega},\, g)}{\partial x_n} &= M_n\Big(\sum_{l=1}^{L}(x_n-\mu_l)\mathrm{P}(l|x_n,\, \boldsymbol{\mu}) - \sum_{l=1}^{\Lambda_n}(x_n-\mu_l)\frac{\mathrm{P}(l|x_n,\, \boldsymbol{\mu})}{\sum_{l=1}^{\Lambda_n}\mathrm{P}(l|x_n,\, \boldsymbol{\mu})}\Big) \\ &\quad - \gamma(x_n - \mu_{s_n}) + \sum_{m=1}^{M_n}\sum_{l=1}^{L}\Big(\mathrm{P}(l|x_n,\, \boldsymbol{\mu}) - \mathrm{P}(l|n,\, m,\, \hat{\Omega})\Big)(x_n-\mu_l) \\ &\quad + \sum_{m=1}^{M_n}\sum_{l=1}^{L}\sum_{z=1}^{Z}\Big(\mathrm{P}(l|n,\, m,\, \hat{\Omega})\mathrm{P}(z|x_n,\, l,\, \boldsymbol{\Phi}) - \mathrm{P}(l,\, z|n,\, m,\, \hat{\Psi})\Big)(x_n-\phi_z)\end{aligned} \tag{12}$$

$$\begin{aligned}\frac{\partial C(\Omega|\hat{\Omega},\, g)}{\partial \mu_l} &= -\sigma_0\mu_l + \sum_{n=1,\, l=s_n}^{N}\gamma(x_n-\mu_{s_n}) - \sum_{n=1,\, l\notin\Lambda_n}^{N}M_n\mathrm{P}(l|x_n,\, \boldsymbol{\mu})(x_n-\mu_l) \\ &\quad + \sum_{n=1,\, l\in\Lambda_n}^{N}M_n\Big(\frac{\mathrm{P}(l|x_n,\, \boldsymbol{\mu})}{\sum_{l'=1}^{\Lambda_n}\mathrm{P}(l'|x_n,\, \boldsymbol{\mu})} - \mathrm{P}(l|x_n,\, \boldsymbol{\mu})\Big)(x_n-\mu_l) \\ &\quad + \sum_{n=1}^{N}\sum_{m=1}^{M_n}\Big(\mathrm{P}(l|n,\, m,\, \hat{\Omega}) - \mathrm{P}(l|x_n,\, \boldsymbol{\mu})\Big)(x_n-\mu_l) \\ &\quad + \sum_{n=1}^{N}\sum_{m=1}^{M_n}\sum_{z=1}^{Z}\Big(\mathrm{P}(l|n,\, m,\, \hat{\Omega})\mathrm{P}(z|x_n,\, l,\, \boldsymbol{\Phi}) - \mathrm{P}(l,\, z|n,\, m,\, \hat{\Omega})\Big)(\mu_l-\phi_z)\end{aligned} \tag{13}$$

Our estimated model parameters, $\Omega = \langle\boldsymbol{\beta}, \boldsymbol{\chi}, \boldsymbol{\Phi}, \boldsymbol{\mu}\rangle$, respectively capture topic word distributions (Problem 1 *(i)*) as well as document, topic, and label coordinates (Problem 2). Using Ω we can also compute the topic probability distribution of a document d_n (Problem 1 *(ii)*) and topic and word distributions of a label l (Problem 1 *(iii)*), respectively as follows.

$$\mathrm{P}(z|d_n,\, \Omega) = \sum_{l=1}^{L}\mathrm{P}(l|x_n)\mathrm{P}(z|l,\, x_n,\, \boldsymbol{\Phi}) \tag{14}$$

$$\mathrm{P}(z|l,\, \Omega) = \frac{\sum_{n=1}^{N}\mathrm{P}(d_n)\mathrm{P}(l|x_n)\mathrm{P}(z|l,\, x_n,\, \boldsymbol{\Phi})}{\sum_{n=1}^{N}\mathrm{P}(d_n)\mathrm{P}(l|x_n)} \tag{15}$$

$$\mathrm{P}(w|l,\, \Omega) = \frac{\sum_{n=1}^{N}\sum_{z=1}^{K}\mathrm{P}(d_n)\mathrm{P}(l|x_n)\mathrm{P}(z|l,\, x_n,\, \boldsymbol{\Phi})\mathrm{P}(w|\beta_z)}{\sum_{n=1}^{N}\mathrm{P}(d_n)\mathrm{P}(l|x_n)},\ \mathrm{P}(d_n) = \frac{1}{N}$$

3.2.1 Quantifying Common and Discriminative Topics.

Based on $p(z|l)$ as in Eq. (15), we define $cmn_score = p(z|c)$ and $disc_score = \frac{p(z|l)}{max_{l'\neq l}p(z|l')}$ to identify common and discriminative topics, respectively. Visually, common topics are those close to the common label and discriminative topics of a label are those near to that label but far from the other labels in the embedding space.

3.2.2 Choosing K.

In practice, setting K is challenging for the contrastive task for guessing the total number of relevant topics across corpora is nontrivial. Topic models are typically evaluated by either measuring performance on some external task, such as document classification, or by estimating the probability of unseen

held-out documents. On average, a better model produces a higher probability of held-out documents.

In our case, we aim to find K that gives rise to a model that explains (or represents) the original data $\mathcal{D}$ the best. To this end, we fix topic word distributions as well as the topic and label coordinates, and pick K (among a set of alternatives) that maximizes the following marginal likelihood of the input documents.

$$\mathrm{P}(\mathcal{D}|\boldsymbol{\mu}, \Phi, \boldsymbol{\beta}) = \prod_{n=1}^{N} \int_{x_n} \mathrm{P}(\Lambda_n)\, \mathrm{P}(d_n|x_n, \boldsymbol{\mu}, \Phi, \boldsymbol{\beta})\, \mathrm{P}(x_n) dx_n \,, \quad (16)$$

where $\mathrm{P}(d_n|x_n, \boldsymbol{\mu}, \Phi, \boldsymbol{\beta})$ is as given in Eq. (3). Here we treat input documents as though they are unseen and integrate out the unknown variables (their coordinates) to compute their likelihood.

The integral in Eq. (16) is not tractable, as such, we use importance sampling [27] to estimate the marginal likelihood, where we sample S (=500) samples of x_n from $P(x_n)$. The marginal likelihood is then approximated as:

$$\mathrm{P}(\mathcal{D}|\boldsymbol{\mu}, \Phi, \boldsymbol{\beta}) \simeq \prod_{n=1}^{N} \Big[\frac{1}{S} \sum_{s} \mathrm{P}(\Lambda_n)\, \mathrm{P}(d_n|x_n^{(s)}, \boldsymbol{\mu}, \Phi, \boldsymbol{\beta}) \Big] \,. \quad (17)$$

3.2.3 **Computational Complexity.** In ContraVis inference algorithm, E-step is performed in $O(NLKW)$ where we evaluate Eq. (9) for all words of each document. For M-step, the most expensive task is to perform the L-BFGS algorithm. As we know, for each iteration, L-BFGS has a computational cost of $O(qp)$ plus the cost to evaluate $C(\Omega|\hat{\Omega})$ in Eq. (5) and its gradients [17]. Here, q is the number of steps stored in the memory and $p = (N + L + K)d$ is the number of variables. The evaluations of $C(\Omega|\hat{\Omega})$ and its gradients can be performed in $O(NLKW)$. Since modest values of q are often between 3 and 20 [17] and the number of iterations in L-BFGS is often small, M-step can be performed at a cost of $O(NLKW)$. Therefore, ContraVis inference has an overall computational cost of $O(NLKW)$ per iteration, which is linear in the number of documents, labels, topics and vocabulary size.

4 QUANTITATIVE EVALUATION

We evaluate ContraVis both quantitatively (in this section) and qualitatively (in §5, through many case studies). Before detailing our experiment results, we describe the datasets we used for evaluation, which come from three different domains: news articles, political posts, and employee peer reviews.

4.1 Dataset Description

NYTnews consists of all the news articles published at the New York Times from January 1987 to June 2007.[1] Articles are labeled by online sections where they are published to. We omit articles published in more than one online section and select the following $L = 9$ categories: `sports`, `business`, `arts`, `style`, `travel`, `technology`, `real-estate`, `magazine`, and `health`. We also use time information to contrast articles, like `technology` news from 1990's vs. 2000's.

PolForum contains posts from a political forum, which is organized into various threads of subject areas for discussion.[2] Posts are labeled by the thread under which they appear. In total there are $L = 8$ different threads; in order of frequency: `abortion`, `religion`, `taxes`, `economy`, `military`, `gun-control`, `environment`, and `race`.

PeerReview contains peer-reviews among employees of a ride-sharing company (proprietary dataset). We group all the reviews each employee received into a document and label documents by their department. The company has $L = 4$ major departments: `operations`, `engineering`, `people-analytics`, and `finance`.

Table 2: Dataset statistics. Descriptions given in §4.1.

	#labels L	#documents N	# words W
NYTnews	9	515252	8666
PolForum	8	18732	6774
PeerReview	4	5336	4032

Table 2 shows the size of each dataset after preprocessing. In our experiments, for each dataset, we sample 250 documents from each label uniformly at random and construct 10 such collections with different samples. We report results averaged across 10 collections.

In this section, we evaluate ContraVis on three different *quantitative* tasks, and compare it to existing topic models rigorously. The tasks capture both semantic and visualization quality of the competing methods. We present quantitative performance on all datasets; NYTnews with 9, PolForum with 8, and PeerReview with 4 labels.

Before detailing our empirical results, we describe the baseline methods that we compared to.

4.2 Comparison Methods

Table 3 lists comparative methods and their properties. We compare ContraVis to both unsupervised and supervised topic models. ContraVis makes use of document labels for topic modeling and visualization, as such it is in the supervised category.

(1) (Supervised) *sLDA*[3][16] (s for supervised)
(2) (Supervised) *DiscLDA*[4] [9] (Disc for discriminative)
(3) (Unsupervised): *LDA*[5] [2]

The above baselines do not produce an explicit visual embedding of the documents (nor of the topics or labels), unlike ContraVis. For our quantitative tasks that evaluate visualization quality, we pipeline these methods with dimensionality reduction. Specifically, we obtain the documents' topic representation and use t-SNE [14] to embed them into 2-d. (See for e.g., Fig. 8b)

(4) (Unsupervised & Joint (Topics+Vis)): *PLSV*[6] [7]
(5) (Supervised & Joint (Topics+Vis)): ContraVis [this paper].

Table 3: Comparison of methods by properties.

Properties vs. Methods	*LDA*	*sLDA*, *DiscLDA*	*PLSV*	ContraVis
Contrastive (Supervised)		✔		✔
Visually-expressive			✔	✔
Unified/Jointly modeled			✔	✔

4.3 Task 1: Discrimination in 2-d

With this work, we set out to not only perform topic modeling but also visualization that enables contrastive analysis—which sets our work apart. As such, the first quantitative task involves contrastive

[1] Publicly available at https://catalog.ldc.upenn.edu/ldc2008t19
[2] Publicly available at https://github.com/tuanlvm/ContraVis/tree/master/data
[3] We use author implementation at https://github.com/blei-lab/class-slda. This is a variant of *sLDA* with a categorical response [5].
[4] We obtained the implementation from the author Prof. Simon Lacoste-Julien. In our experiments, we use *DiscLDA* with fixed transformation matrix T.
[5] http://scikit-learn.org (sklearn.decomposition.LatentDirichletAllocation)
[6] We use the implementation at https://github.com/tuanlvm/SEMAFORE

power in the visualization space. Intuitively, a good contrastive visualization should well separate the documents with common topics from those belonging to discriminative topics. Moreover, it should be easy for users to spot these documents in the visualization. With this intuition, we design two subtasks for evaluation, named *Common* and *Discriminative*, described as follows.

Setup: To setup, we identify the top three most frequent labels from each dataset, which we refer to as C, B, and A (without loss of generality). For the *Common* task, we sample 250 documents each from label A and B, and 100 from C. We split C documents in half and mix with those from A and B. We consider 300 documents (250 A + 50 C) as class 1 and the other 300 (250 B + 50 C) as class 2. We then apply the methods to compare class 1 and class 2. The goal is to identify the documents with common topics among the two classes, i.e. the hidden label C ones which are considered as ground truth for the *Common* task.

For the *Discriminative* task, we sample 500 documents from C, and 50 each from A and B. We consider 300 documents (250 C + 50 A) as class 1 and the other 300 (250 C + 50 B) as class 2. We then apply the methods to compare class 1 and class 2. This time the goal is to identify the documents with discriminative topics among the two classes, i.e. the hidden label A and B ones which are considered as ground truth for the *Discriminative* task.

We quantify the ranking performance of the methods. Concretely, we rank the documents by their distance to the common label c (in 2-d) for our ContraVis (in increasing order for *Common*, and decreasing order for *Discriminative*). Since the baseline methods do not have such a c, we first find the mean coordinate of all documents from each class and take the mean of these means to designate as the common label's coordinate. We repeat the experiments for 10 randomly selected samples per dataset, and report average precision and recall on each subtask. Basically, a good contrastive visualization will have a better ranking performance because it separates well the documents with common topics from the ones with discriminative topics.

Results: Figure 3 shows the precision versus recall curves for all methods on each dataset for the *Common* task, and Figure 4 the respective plots for the *Discriminative* task. We see that ContraVis produces a better ranking for both tasks and outperforms the competing methods.

More concretely, Table 4 and Table 5 provide the mean average precision (area under PR curves) respectively for both tasks. We find that ContraVis is significantly better than the baselines in many cases according to the paired Wilcoxon signed rank test. An example illustration on NYTnews is given in Figure 5 comparing ContraVis with *sLDA* for these two subtasks. These results show the contrastive power of ContraVis in the visualization space. In addition, the ContraVis's ranking performance indicates that users can easily spot the similarities or differences among document collections by simply observing the distances between documents and the common label c, which enables visual contrastive analysis. Many case studies will be presented in §5 to show the potential of ContraVis for contrasting and comparing document collections.

4.4 Task 2: Word-to-Label Relevances

We find that the ContraVis's contrastive power in visualization space (§4.3) does not come at the expense of discriminative power and semantic coherence. To show this, we measure semantic discrimination quality, particularly, how well ContraVis can identify the most relevant words to a label (or class) as compared to the baseline methods. For instance, humans would frequently use words like 'life' and 'baby' when talking about `abortion`, and the words 'god' and 'faith' for `religion` (See Figure 10).

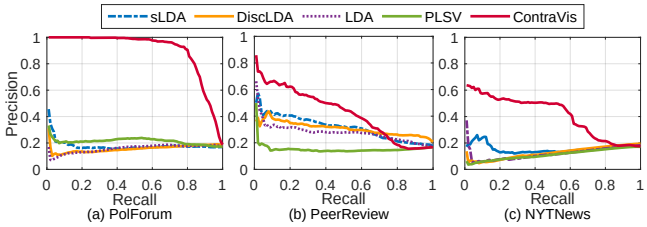


Figure 3: Precision-Recall curves (avg'ed over 10 samples) for task *Common*.

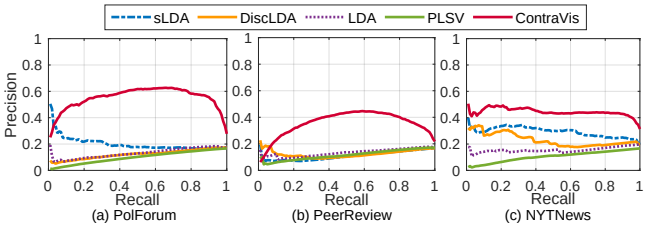


Figure 4: Precision-Recall curves (avg'ed over 10 samples) for task *Discriminative*.

Table 4: Mean Average Precision (MAP) (avg'ed across 10 samples) ± stan.dev. on task *Common*. ▲ (p<0.001) and △ (p<0.005) are cases where ContraVis is significantly better than the baseline w.r.t. the paired Wilcoxon signed rank test.

	ContraVis	*sLDA*	*DiscLDA*	*LDA*	*PLSV*
PolForum	**0.897**±0.03	0.178±0.04▲	0.157±0.08▲	0.158±0.03▲	0.213±0.14▲
PeerReview	**0.423**±0.16	**0.314**±0.10	**0.307**±0.08	**0.276**±0.09	0.152±0.02▲
NYTnews	**0.410**±0.24	0.156±0.03△	0.126±0.03▲	0.121±0.02△	0.112±0.01▲

Table 5: Mean Average Precision (MAP) of methods (avg'ed across 10 samples) ± stan.dev. on task *Discriminative*.

	ContraVis	*sLDA*	*DiscLDA*	*LDA*	*PLSV*
PolForum	**0.548**±0.07	0.201±0.08▲	0.122±0.02▲	0.132±0.02▲	0.099±0.00▲
PeerReview	**0.356**±0.07	0.111±0.01▲	0.125±0.03▲	0.134±0.03▲	0.116±0.01▲
NYTnews	**0.444**±0.13	**0.293**±0.18	**0.232**±0.19	0.156±0.04▲	0.105±0.01▲

Ground truth: There is no existing repository of word-to-subject-matter relevances that we could directly use as ground truth. Therefore, we use 3 different means to create a ranked list of words by relevance to each label.

1) tf-idf: We rank the words that appear in documents of a certain label by their total tf-idf values. As such, this ranking is obtained based on in-corpus information.

2) Google 5-gram frequencies: We take a large collection of 5-grams generated by Google Inc. from around 1 trillion word tokens of text from public Web pages.[7] For each label l (e.g. `religion`), we count the number of times the word pair (w, l) co-occurs across the 5-grams, and sort the words (w's) by their frequency.

[7] https://catalog.ldc.upenn.edu/ldc2006t13

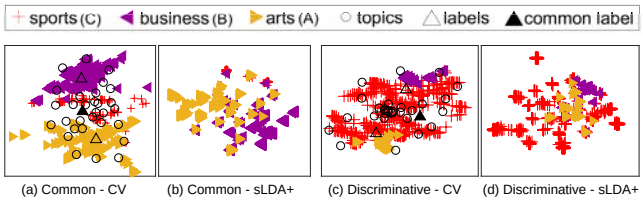


Figure 5: (best in color) Visualization of documents from NYTnews (K=30); *Common* task: (a) ContraVis (CV) places docs with common topics (sports) near the common label, (b) *sLDA+*: it is not as easy to spot common docs; *Discriminative* task: (c) CV places docs with discr. topics (business, arts) far from the common label as well as one another, (d) *sLDA+*: it is not easy to spot these docs. + is short for +t-SNE.

3) Yahoo 5-gram frequencies: We obtain a 3rd ranking of words by label relevance based on Yahoo 5-grams[8], in the same fashion as the above.

Setup: We quantify the word-to-label relevances of models by computing $p(w|l)$, which is given in Eq. (3.2) for ContraVis. For unsupervised models, *LDA* and *PLSV*, we compute $p(w|l) = \sum_z p(w|z)p(z|l)$, where $p(w|z)$ is readily output by the model and $p(z|l) = \frac{\sum_{d_n:s_n=l} p(z|d_n)}{\sum_{z=1}^{K}\sum_{d_n:s_n=l} p(z'|d_n)}$. For *sLDA* and *DiscLDA*, $p(w|l) \propto \sum_{n=1}^{N}\sum_{z=1}^{K} p(w|z)p(z|d_n)p(l|d_n)p(d_n)$ with $p(d_n) = \frac{1}{N}$, where $p(l|d_n)$ for *sLDA* is computed using softmax [5], and is approximated for *DiscLDA* using bridge sampling [9].

We measure the quality of the top t words in a given model's ranking, based on the Normalized Discounted Cumulative Gain: $NDCG = \frac{DCG@t}{IDCG@t}$. $DCG@t = \sum_{i=1}^{t} \frac{rel_i}{\log_2(i+1)}$ (the higher, the better), where rel_i is the "true" relevance of top ith word. We obtain rel_i from the ground truth using corresponding word's respective score used for ranking (i.e., frequencies or tf-idf). $IDCG@t$ is the "ideal" DCG, that of the top t words of the ground truth itself.

Note that Yahoo and Google rankings contain many out-of-corpus words since they are based on external corpora, where we expect all methods to obtain lower $NDCG$ relative to the tf-idf ground truth with the ranked list of in-corpus words.

Results: Figure 6 shows $NDCG@t$ performance of the methods for $t \in \{10, 20, \ldots, 100\}$ on each dataset based on the tf-idf in-corpus ground truth. ContraVis consistently ranks among the top across all t thresholds, closely followed by *sLDA*. To quantify the differences more concretely, we perform the paired Wilcoxon signed rank test across 10 samples and all labels of each dataset, as presented in Table 6 for all three ground truth. As one can see, ContraVis significantly outperforms the competing methods in many cases. On NYTnews there is no significant difference between ContraVis and *sLDA*, but, ContraVis is significantly superior to *sLDA* on the other two datasets across all ground truth.

Pairwise significance tests according to the paired Wilcoxon signed rank test reveal the (total or partial) orderings in Table 7. We find that ContraVis is significantly superior to all methods in many cases and ties with *sLDA* and *LDA* in some cases. We wrap up with Table 8 that provides qualitative evidence to the quality of $p(w|l)$ rankings by ContraVis. It ranks words like 'white' and 'black' at top for l : `race` from PolForum, 'candidate' and 'recruit' for l : `people-analytics` from PeerReview, and 'building' and 'house' for l : `real-estate` from NYTnews, among others, which all agree with human intuition.

[8]https://webscope.sandbox.yahoo.com/catalog.php?datatype=l

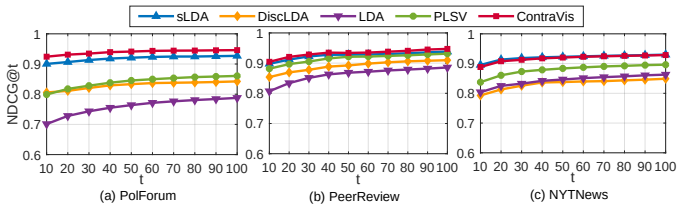


Figure 6: $p(w|l)$ ranking quality w.r.t. tf-idf ground-truth and $NDCG@t$ at varying t: number of top words. Results are averaged across 10 samples and all labels for each dataset.

Table 6: $p(w|l)$ ranking quality w.r.t $NDCG@100$ avg'ed across 10 samples and all labels from each dataset ($K = 30$). Symbols ▲ (p<0.001) and △ (p<0.005) denote the cases where ContraVis is significantly better than the baseline w.r.t. the paired Wilcoxon signed rank test.

	ContraVis	*sLDA*	*DiscLDA*	*LDA*	*PLSV*
PolForum					
tf-idf	**0.9460**±0.00	0.9270±0.01▲	0.8424±0.01▲	0.7878±0.07▲	0.8605±0.01▲
Google	**0.5494**±0.02	0.5151±0.03▲	0.4681±0.02▲	0.3770±0.06▲	0.4562±0.01▲
Yahoo	**0.5635**±0.02	0.5291±0.03△	0.4806±0.01▲	0.3875±0.06▲	0.4686±0.02▲
PeerReview					
tfidf	**0.9471**±0.00	0.9382±0.00▲	0.9096±0.01▲	0.8855±0.04▲	0.9313±0.00▲
Google	**0.2388**±0.01	0.2128±0.02▲	0.1805±0.02▲	0.2001±0.02▲	0.1971±0.02▲
Yahoo	**0.2461**±0.01	0.2193±0.02▲	0.1855±0.02▲	0.2061±0.02△	0.2022±0.02▲
NYTnews					
tf-idf	**0.9286**±0.00	**0.9300**±0.01	0.8490±0.01▲	0.8628±0.04▲	0.8962±0.01▲
Google	**0.2483**±0.03	**0.2384**±0.00	0.1947±0.01▲	**0.2348**±0.03	0.2119±0.01△
Yahoo	**0.2574**±0.03	**0.2479**±0.02	0.2062±0.01▲	**0.2450**±0.03	0.2219±0.01△

Table 7: Relative $p(w|l)$ ranking quality comparison w.r.t. $NDCG@100$ ($K = 30$). $\gg$ denotes p<0.001, $>$ denotes p<0.005 and $\approx$ denotes indifference w.r.t. the paired Wilcoxon signed rank test over 10 samples and all labels per dataset.

Dataset	GTruth	Relative Order
PolForum	tf-idf	CV $\gg$ *sLDA* $\gg$ *PLSV* $\gg$ *DiscLDA* $>$ *LDA*
	Google	CV $\gg$ *sLDA* $>$ *DiscLDA* $\approx$ *PLSV* $\gg$ *LDA*
	Yahoo	CV $>$ *sLDA* $>$ *DiscLDA* $\approx$ *PLSV* $\gg$ *LDA*
PeerReview	tf-idf	CV $\gg$ *sLDA* $\gg$ *PLSV* $\gg$ *DiscLDA* $\approx$ *LDA*
	Google	CV $\gg$ *sLDA* $>$ *PLSV* $\gg$ *DiscLDA*; CV$\gg$*LDA*
	Yahoo	CV $\gg$ *sLDA* $>$ *PLSV* $\gg$ *DiscLDA*; CV$>$*LDA*
NYTnews	tf-idf	CV $\approx$ *sLDA* $\gg$ *PLSV* $\gg$ *DiscLDA*; CV$\gg$*LDA*
	Google	CV $\approx$ *sLDA* $>$ *PLSV* $>$ *DiscLDA*; CV$\approx$*LDA*
	Yahoo	CV $\approx$ *sLDA* $>$ *PLSV* $>$ *DiscLDA*; CV$\approx$*LDA*

4.5 Task 3: Classification in 2-d

Since learning contrastive visualization is supervised, we can expect that the labels should be well separated in the visualization space. However, we find it is still valuable to show what is the extent that ContraVis separates the labels compared to the baselines. To investigate this, we perform k Nearest Neighbors (kNN) classification with visualization coordinates as inputs, i.e., each document will be assigned to the dominant label among its k nearest neighbors in

Table 8: Top 5 most relevant/discriminative words for a selection of three labels from each dataset w.r.t. $p(w|l)$ by ContraVis closely agree with human understanding.

Dataset	Label	Top 5 Words
PolForum	abortion	abort, human, life, peopl, person
	taxes	tax, govern, incom, pay, will
	race	black, white, peopl, race, racist
PeerReview	operations	help, citi, idea, busi, peopl
	engineering	help, project, engin, issu, good
	people-analytics	candid, help, recruit, hire, manag
NYTnews	sports	game, team, play, season, year
	business	compani, year, percent, will, market
	real-estate	build, hous, year, number, apart

the visualization space. The accuracy is the fraction of documents with predicted label matching the truth. We report the accuracy on each dataset, averaged across 10 collections. Intuitively, higher accuracy means that labels are more separated in the visualization space.

Results: Figure 7 shows kNN accuracy of all methods on each dataset, for varying (left) number of topics K and (right) number of nearest neighbors k and an example illustration on NYTnews is given in Figure 8 comparing ContraVis with *DiscLDA* (the second best performing baseline on this task). We note the near-perfect accuracy that ContraVis achieves across all datasets and settings which may imply that ContraVis aggressively separate the labels to learn discriminative topics and thus the learned topic model may be too overfitted. However, as discussed in §6, ContraVis focuses on comparing document collections, which aims to learn a discriminative topic model that best explains the similarities and differences among these document collections. Its focus is not on prediction tasks such as classification or regression. Therefore, overfitting may not be a big issue to ContraVis, but conversely, in some cases it may be a good trait (e.g., when we try to extract the differences between two very similar labels).

5 QUALITATIVE EVALUATION

In this section, we evaluate ContraVis qualitatively through many case studies where we apply our method to contrast and compare 2 or 3 document collections. These case studies represent common application scenarios. Our case studies are designed to answer two questions: (1) How well does ContraVis capture the underlying discriminative and common topics? Do these topics agree with human intuition? and (2) How good are the visualizations that ContraVis produces?

5.1 NYTnews

We perform our first case study on NYTnews using three labels (`sports`, `arts`, `style`). As shown in Figure 1(a), ContraVis well-separates the collections and the learned topics are also discriminative. For `sports`, we see topics about basketball and racing game. Topics exclusive to `arts` include 'museum', 'art', 'exhibit' and 'street', 'art', 'paint'. For `style`, we see a topic on fashion ('cloth', 'design', 'women') and another one about lifestyle ('graduate', 'university', 'father'). Figures 1(b) and 1(c) show the visualizations by *PLSV* and *sLDA*+t-SNE which are not as comprehensive and documents are not as well separated.

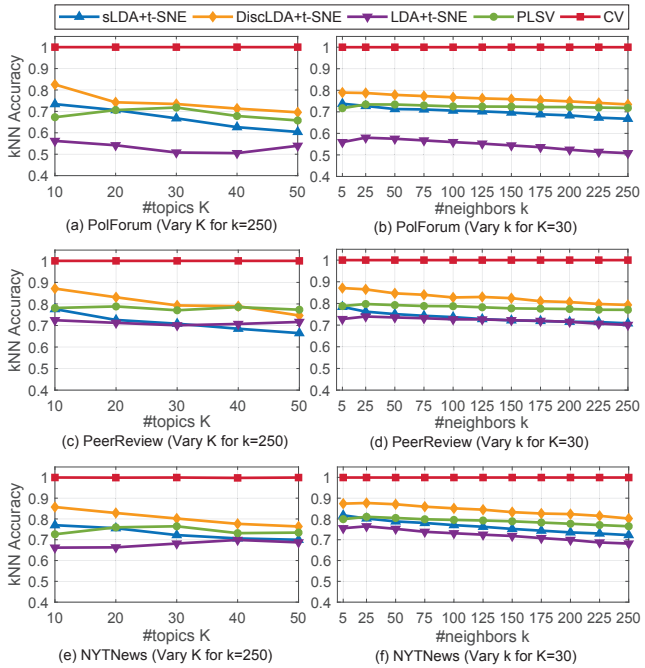


Figure 7: kNN classification accuracy in 2-d (avg'ed across 10 random samples per dataset) on (from top to bottom) PolForum, PeerReview, and NYTnews. ContraVis (denoted as CV) consistently and significantly outperforms competition in all settings.

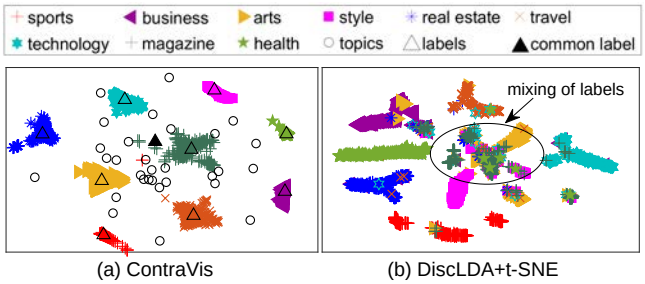


Figure 8: (best in color) 2-d visualization of documents from NYTnews ($K = 30$); (left) ContraVis embedding well-separates documents with different labels, in contrast (right) *DiscLDA*+tSNE embedding is not as pure.

The next study compares documents across time. Specifically, we apply ContraVis to contrast `technology` articles in NYTnews from 1990's vs. 2000's. As shown in Figure 9, in 1990's, tech-news talked about 'website', 'travel', 'find', 'book', etc. Notice that WWW was invented in 1989, after which computers started being used for finding information, booking travel, etc. In contrast, in 2000's news started to involve 'video', 'game', and 'xbox', that is around the period when Xbox was introduced in 2001. In both time periods, tech-news discuss about 'system', 'machine', 'power', and 'technology'. The number of articles also reflects the development of technology. We have only 79 articles on `technology` in 1990-1997, with a lot more in 2000 and onward.

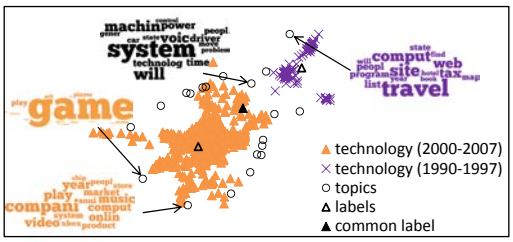


Figure 9: (best in color) Comparing `technology` articles from different time periods in NYTnews using ContraVis (K=20).

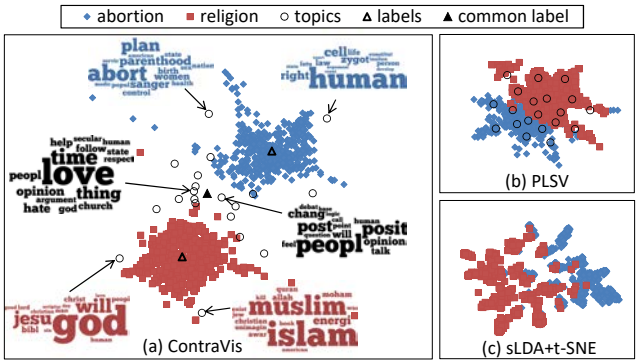


Figure 10: (best in color) Contrastive visualization of `abortion` and `religion` posts from PolForum (K=20); (left) ContraVis embedding well-separates the collections and two most discriminative topics per label and two common topics shown are intuitive, while (right) *PLSV* and *sLDA*+tSNE embeddings are not on par.

5.2 PolForum

Next, we apply ContraVis to see what topics are discussed in different threads of PolForum. Figure 10 shows a contrastive visualization of `abortion` vs. `religion` posts. *PLSV* and *sLDA*+t-SNE embeddings in Figure 10(b) and 10(c) do not provide a well visual comparison of the two threads. ContraVis embedding in Figure 10(a), by contrast, shows separated clusters. Also shown are two most discriminative topics for each thread as well as two common topics. Abortion topics are intuitive; with words like 'abort', 'plan', 'parenthood' and 'human', 'cell', 'zygote'. Under `religion`, people talk about Christianity and Islam, respectively involving words 'god', 'jesus', 'bible' and 'islam', 'muslim', 'allah'. They share common topics around 'peopl', 'opinion', 'position', etc.

Figure 11 shows another use case, comparing posts from `taxes`, `military` and `gun-control` threads. We see that a common topic has frequently used words such as 'govern', 'right', 'rule' and 'power'. Also shown are one discriminative topic for each thread, which are right on subject and coherent. For example, topic exclusive to `gun-control` has representative words 'gun', 'furious', 'law', 'firearm' and 'atf'. ATF is a United States law enforcement agency for regulating use of firearms. In the embedding, `taxes` and `gun-control` posts are close to each other which suggests that they may be more related to each other than they are to `military`.

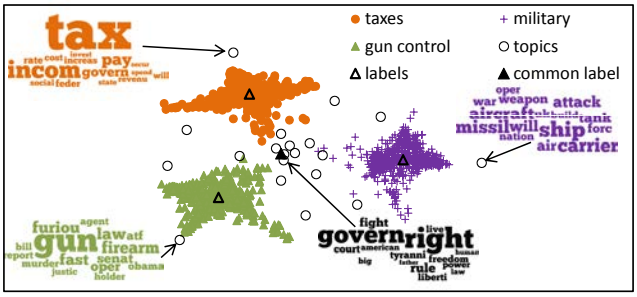


Figure 11: (best in color) Comparing posts on `gun-control`, `taxes`, & `military` from PolForum using ContraVis (K=20).

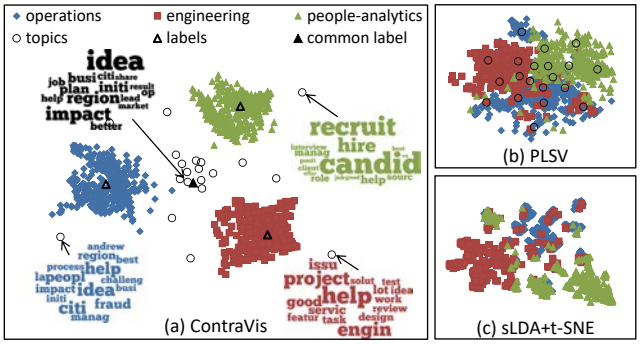


Figure 12: (best in color) Contrastive visualization of reviews for employees from different departments in PeerReview (K=20); (left) In ContraVis embedding, departments are well-separated and disc. topics reflect the role of each department; (right) *PLSV* and *sLDA*+tSNE embeddings mix the labels. *sLDA*+tSNE embedding is doc-only (no topics).

5.3 PeerReview

Finally, we present a case study where we compare reviews of employees from different departments in PeerReview. Since documents are reviews, we expect to see very frequent words like 'help(ful)', 'good' and 'idea' in the topics. In spite of that, ContraVis still can detect discriminative topics which are closely related to the role of each department. For example, from ContraVis embedding in Figure 12(a), one can tell that `people-analytics` department is responsible for recruiting, `engineering` deals with project design and testing, and `operations` is responsible for managing business in different regions. These insights cannot be gained from the embeddings by *PLSV* and *sLDA*+t-SNE in Figure 12(b) and 12(c).

Through several case studies from different domains above, we have shown that ContraVis finds meaningful topics that agree with human understanding. In addition, it provides an intuitive visualization that helps quickly identify common and discriminative topics as well as their relationship to documents and labels. Overall, ContraVis shows promise for exploratory and comparative text analysis, and is superior to existing topic models like *PLSV* (unsupervised joint method) and *sLDA* which necessitates post-hoc embedding with subpar results.

6 RELATED WORK

Unsupervised Topic Modeling and Visualization: There are various unsupervised topic models [2, 6, 22]. *LDA* [2] is arguably the most popular one that represents each document as a mixture of topics and each topic as a probability distribution over words. For visualization, one needs to pipeline these with dimensionality reduction methods such as t-SNE [14], LargeVis [26]. Recently, methods that jointly model topics and visualization using a single generative model have been developed. This line of research is pioneered by *PLSV* [7] and is further developed by later works that incorporate the neighborhood graph [12], use document network/hyperlinks information [10], or model documents' spherical representation [11]. Similar to ContraVis, these methods assume that all documents and topics are represented as points in the same visualization space. The main differences between ContraVis and these methods are: (1) ContraVis is a supervised method where labels are used to extract the discriminative topics and produce contrastive visualization; (2) ContraVis assumes that each document has a label distribution and labels are also located as points in the visualization space. The distances from documents to labels encode the label distributions; (3) ContraVis introduces label-dependent topic distributions for each document. Basically, a document tends to pick topics belonging to its label for generating its words; (4) ContraVis explicitly models common topics by introducing the common label which is shared across documents. By modeling label distributions and label-dependent topic distributions, ContraVis can effectively extract discriminative and common topics. We extensively compare to *LDA*+t-SNE and *PLSV*. Other joint methods [10–12] are not directly comparable to ContraVis, as they use additional input such as neighborhood information or the document network. We can extend ContraVis to incorporate such information if available.

Supervised Topic Modeling and Visualization: Several supervised topic models have been proposed such as *sLDA* [16], *DiscLDA* [9] for single-label documents and *LabeledLDA* [20], *PLDA* [21] for multi-label and partially labeled documents respectively. Recently, *MedLDA* integrates the max-margin principle into supervised topic models by optimizing the goodness of fit of the learned topic model and its prediction accuracy on a max-margin classifier [29]. These models use document side information such as categories or review rating scores to steer the model learning towards extracting more discriminative topics for prediction tasks such as classification or regression. Unlike these supervised topic models, ContraVis focuses on the task of comparing document collections whose objective is to extracting common and discriminative topics for exploring both the similarities and the differences among these collections. Its focus is not on document classification tasks. Moreover, ContraVis also jointly produces a contrastive visualization, which enable visual comparison of document collections. For supervised topic models mentioned above, one needs to pipeline these methods with a post-hoc embedding using a dimensionality reduction technique (like t-SNE [14]), which is not effective for visually contrasting document collections as we demonstrated in our experiments. Among these models, *sLDA* and *DiscLDA* are more closely related to our work as they are for single-label documents and in principle we can extend ContraVis to integrate the max-margin mechanism as in *MedLDA*. In our experiments, we extensively compared to two state-of-the-art supervised topic models for single-label documents, *sLDA*+t-SNE and *DiscLDA*+t-SNE.

Comparative Text Mining: There have been works that directly model common and discriminative topics for comparative text mining. CCMix [28] introduces a probabilistic mixture model to identify k common themes across all collections and k collection-specific themes for each collection. ccLDA [19] improves over CCMix [28] by introducing an LDA-based analog, which automatically learns the probability of using the common vs. collection-specific word distributions while generating documents. While ccLDA maintains common and collection-specific word probability vectors for each topic, differential topic models [4] propose to use a hierarchy of topics across collections. Recently, there are works that propose using spectral methods [30] and nonnegative matrix factorization [8] for comparative text mining. However, they mainly focus on scenarios involving two collections while our method can handle multiple corpora. Moreover, although these methods can infer common and discriminative topics across collections, they do not provide any means for visual comparison, unlike ContraVis.

Visualization for Comparison: ContraVis uses scatterplot visualization to display the relationship between documents, topics, and labels. There exist work that propose other visualization forms for making comparisons. For example, DiTop-View, in which topics are represented as glyphs on a 2D plane, is used for comparing document collections [18]. Buddy plots, on the other hand, represent each document as a row and other documents are encoded as circular glyphs along the row for comparison of topic models [1]. TextDNA [25] use configurable colorfields to visualize word usage patterns across different text collections. Other work focus on comparing two or more documents using word clouds [3, 13]. Compared to these visualization forms, scatterplot by ContraVis provides an effective way to get an overview of multiple corpora as it explicitly shows the relationship between documents, topics and labels. A challenge with scatterplots is that they are susceptible to overdraw (overlapping glyphs); for very large document corpora one could employ several tackling strategies [15].

We note that work on summarizing and extracting phrases for contrasting opinions in text documents [23, 24] are different in focus, without specific emphasis on topic modeling or visual analysis.

7 CONCLUSION

We propose ContraVis[9], a new supervised topic model for contrasting and visualizing multiple document collections. Our model jointly learns common and discriminative topics as well as embeddings of documents, topics, and labels for visualization. Due to its joint nature of contrastive topic modeling and visualization, the learned topics and their relationships to documents and labels are reflected faithfully in the visualization, which facilitates exploring and comparing the collections. To the best of our knowledge, ContraVis is the first to jointly address the contrastive topic modeling and visualization problems. We conduct comprehensive experiments on real-life datasets and the results show that our method significantly outperforms the existing techniques in terms of contrastive power as well as visual and semantic coherence.

[9] Code and public-domain datasets available at https://github.com/tuanlvm/ContraVis

ACKNOWLEDGMENTS

This research is sponsored by NSF CAREER 1452425 and IIS 1408287. We thank Daniel Bay from Uber Technologies Inc. for helping us in our data analysis. Any conclusions expressed in this material are of the authors and do not necessarily reflect the views, expressed or implied, of the funding parties or our industrial partners.

REFERENCES

[1] Eric Alexander and Michael Gleicher. 2016. Task-driven comparison of topic models. *IEEE transactions on visualization and computer graphics* 22, 1 (2016).
[2] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research* 3, Jan (2003), 993–1022.
[3] Quim Castella and Charles Sutton. 2014. Word storms: Multiples of word clouds for visual comparison of documents. In *Proceedings of the 23rd international conference on World wide web*. ACM, 665–676.
[4] Changyou Chen, Wray L. Buntine, Nan Ding, Lexing Xie, and Lan Du. 2015. Differential Topic Models. *IEEE Trans. Pattern Anal. Mach. Intell.* 37, 2 (2015).
[5] Wang Chong, David Blei, and Fei-Fei Li. 2009. Simultaneous image classification and annotation. In *CVPR*. 1903–1910.
[6] Thomas Hofmann. 1999. Probabilistic latent semantic analysis. In *UAI*. 289–296.
[7] Tomoharu Iwata, Takeshi Yamada, and Naonori Ueda. 2008. Probabilistic latent semantic visualization: topic model for visualizing documents. In *KDD*. 363–371.
[8] Hannah Kim, Jaegul Choo, Jingu Kim, Chandan K Reddy, and Haesun Park. 2015. Simultaneous discovery of common and discriminative topics via joint nonnegative matrix factorization. In *KDD*. 567–576.
[9] Simon Lacoste-Julien, Fei Sha, and Michael I Jordan. 2009. DiscLDA: Discriminative learning for dimensionality reduction and classification. In *NIPS*. 897–904.
[10] Tuan M. V. Le and Hady W. Lauw. 2014. Probabilistic latent document network embedding. In *ICDM*. 270–279.
[11] Tuan M. V. Le and Hady W. Lauw. 2014. Semantic visualization for spherical representation. In *KDD*. 1007–1016.
[12] Tuan M. V. Le and Hady W. Lauw. 2016. Semantic visualization with neighborhood graph regularization. *Jour. of AI Res.* 55 (2016), 1091.
[13] Tuan M. V. Le and Hady W. Lauw. 2016. Word Clouds with Latent Variable Analysis for Visual Comparison of Documents. In *IJCAI*.
[14] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.
[15] Adrian Mayorga and Michael Gleicher. 2013. Splatterplots: Overcoming overdraw in scatter plots. *IEEE transactions on visualization and computer graphics* 19, 9 (2013), 1526–1538.
[16] Jon D Mcauliffe and David M Blei. 2008. Supervised topic models. In *NIPS*.
[17] J. Nocedal and S. J. Wright. 2006. *Numerical Optimization*. Springer.
[18] Daniela Oelke, Hendrik Strobelt, Christian Rohrdantz, Iryna Gurevych, and Oliver Deussen. 2014. Comparative exploration of document collections: a visual analytics approach. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library.
[19] Michael Paul. 2009. Cross-collection topic models: Automatically comparing and contrasting text. *Urbana* 51 (2009), 61801.
[20] Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP*. 248–256.
[21] Daniel Ramage, Christopher D Manning, and Susan Dumais. 2011. Partially labeled topic models for interpretable text mining. In *KDD*. 457–465.
[22] Joseph Reisinger, Austin Waters, Bryan Silverthorn, and Raymond J Mooney. 2010. Spherical topic models. In *ICML*. 903–910.
[23] Xiang Ren, Yuanhua Lv, Kuansan Wang, and Jiawei Han. 2017. Comparative Document Analysis for Large Text Corpora.. In *WSDM*. ACM, 325–334.
[24] Zhaochun Ren and Maarten de Rijke. 2015. Summarizing Contrastive Themes via Hierarchical Non-Parametric Processes.. In *SIGIR*. ACM, 93–102.
[25] Danielle Albers Szafir, Deidre Stuffer, Yusef Sohail, and Michael Gleicher. 2016. Textdna: Visualizing word usage with configurable colorfields. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 421–430.
[26] Jian Tang, Jingzhou Liu, Ming Zhang, and Qiaozhu Mei. 2016. Visualizing large-scale and high-dimensional data. In *Proceedings of the 25th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 287–297.
[27] Hanna M Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. 2009. Evaluation methods for topic models. In *ICML*. 1105–1112.
[28] ChengXiang Zhai, Atulya Velivelli, and Bei Yu. 2004. A Cross-collection Mixture Model for Comparative Text Mining. In *KDD*. 743–748.
[29] Jun Zhu, Amr Ahmed, and Eric P Xing. 2012. MedLDA: maximum margin supervised topic models. *Journal of Machine Learning Research* 13, Aug (2012), 2237–2278.
[30] James Y. Zou, Daniel J. Hsu, David C. Parkes, and Ryan Prescott Adams. 2013. Contrastive Learning Using Spectral Methods.. In *NIPS*. 2238–2246.

Classifying Extremely Short Texts by Exploiting Semantic Centroids in Word Mover's Distance Space

Changchun Li, Jihong Ouyang, Ximing Li*
College of Computer Science and Technology, Jilin University, China
Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, China
changchunli93@gmail.com,ouyj@jlu.edu.cn,liximing86@gmail.com

ABSTRACT

Automatically classifying extremely short texts, such as social media posts and web page titles, plays an important role in a wide range of content analysis applications. However, traditional classifiers based on bag-of-words (BoW) representations often fail in this task. The underlying reason is that the document similarity can not be accurately measured under BoW representations due to the extreme sparseness of short texts. This results in significant difficulty to capture the generality of short texts. To address this problem, we use a better regularized word mover's distance (RWMD), which can measure distances among short texts at the semantic level. We then propose a RWMD-based centroid classifier for short texts, named RWMD-CC. Basically, RWMD-CC computes a representative semantic centroid for each category under the RWMD measure, and predicts test documents by finding the closest semantic centroid. The testing is much more efficient than the prior art of K nearest neighbor classifier based on WMD. Experimental results indicate that our RWMD-CC can achieve very competitive classification performance on extremely short texts.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning algorithms**;
• **Information systems** → **Clustering and classification**.

KEYWORDS

Extremely Short Texts; Regularized Word Mover's Distance; Semantic Centroid; Hypothesis Margin

ACM Reference Format:
Changchun Li, Jihong Ouyang, Ximing Li. 2019. Classifying Extremely Short Texts by Exploiting Semantic Centroids in Word Mover's Distance Space. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313397

1 INTRODUCTION

The amount of extremely short texts available online, such as social media posts and web page titles, has exploded. Automatic classification of extremely short texts is significant for a wide range of content analysis applications, e.g., user interest profiling [41] and query suggestion [31] etc. However, it is acknowledged as a very challenging task [17, 19, 32] due to the sparseness of extremely short texts, where a single short text, e.g., Twitter posts, often contains even less than 10 words.

Table 1: Examples of document similarity. All values are computed after a removal of the stopwords. *BoW*: the cosine of the BoW representation. *WMD*: the normalized inverse of WMD using a pre-trained *Glove* word embeddings.

Document	*BoW*	*WMD*
I will attend the research meeting. *The conference invited me to participate, I will go.*	0.0	0.89
The relentless pursuit of perfection makes things better. *The constant striving for perfection improves the situation.*	0.13	0.92

The major difficulty in extremely short text classification is that there are much less informative word co-occurrences among short texts under the bag-of-words (BoW) representations, therefore the existing algorithms are mainly based on feature expansion [8, 12, 18, 20, 21, 25, 29, 32, 35, 38–40]. The representative works include: the method of [29] uses search engines to obtain auxiliary contexts, and applies traditional classifiers to those expanded short texts; the work in [25] expands document features by hidden topics learnt from a big reference Wikipedia corpus, where the latent Dirichlet allocation (LDA) [3] topic model is used as a hidden topic excavator. Although these feature expansion methods empirically improve the classification performance, they are heavily domain-dependent. For example, the hidden topics learnt from reference corpora may be inconsistent with the current dataset, making the concatenation features less discriminative.

In some sense, we can analyze the difficulty of extremely short text classification with BoW representations from a just different perspective of document similarity misalignment. Because the short texts are extremely sparse, i.e., containing very few words, even semantically close text pairs may merely share any same word by no means, resulting in inaccurate similarity measure under their BoW representations (Examples are shown in Table 1). In this situation, it is intractable to find the generality of short texts from the same category and the difference of ones from different categories, raising up significant difficulty to build the decision plane of classifiers. Fortunately, recently a word mover's distance (WMD) [15] has been developed, which measures the optimal transport from

*Corresponding author: liximing86@gmail.com

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313397

the embedded words of one document to those of another one, a special case of earth mover's distance (EMD) [26]. The WMD can effectively measure semantic distances among documents using word embeddings [16, 23], therefore it must be a good choice to short texts, instead of distances, as well as similarities, of BoW. Reviewing examples in Table 1, the WMD can effectively measure short text distances even when they share no any same word.

Driven by the success in WMD, we aim at developing short text classifiers using it. A straightforward previous classifier with WMD is the K nearest neighbour (KNN) with a prefetch and prune scheme [15]. It has shown very competitive classification performance, however, the KNN classifier based on WMD is much testing-inefficient especially for big corpora, where for prediction of a test sample one must compute its WMDs with all training samples. Motivated by this, we would like to train a representative semantic centroid for each category, and predict a test sample by comparing the WMDs between it and these semantic centroids, so as to reduce the testing complexity to the category number. We propose a novel **R**egularized **W**ord **M**over's **D**istance **C**entroid **C**lassifier (**RWMD-CC**), of which the objective is built on hypothesis margin with WMD. In RWMD-CC, a problem is that the objective with WMD must be intractable to compute, because the WMD itself is a optimal transport problem. To solve this, we borrow the idea from regularized WMD [6] for efficient optimization, i.e., computing the gradient of the objective. Experimental results indicate that our RWMD-CC outperforms the state-of-the-art baseline classifiers of short texts.

The major contributions of this paper are outlined as follows:

- We develop a both effective and efficient RWMD-CC method for short text classification.
- We build a loss function using hypothesis margin that is based on the structural risk minimization principle, and borrow the idea from regularized WMD for efficient optimization.
- Empirical results indicate that our RWMD-CC outperforms the state-of-the-arts, and is much more efficient on testing than the existing WMD-based methods.

2 PRELIMINARIES

Formally, the word mover's distance (WMD) [15] is a special case of the earth mover's distance (EMD) [26, 27], also named Wasserstein distance [4]. Therefore, we first introduce the EMD, and then show the WMD in detail.

2.1 Optimal Transport and Earth Mover's Distance

The earth mover's distance is the special case of optimal transport distance. Essentially, the optimal transport distance [34] is a measure of the distance between two probability measures over a finite set $\mathcal{V}$ of size $|\mathcal{V}| = V$, i.e., p_1 and p_2. Given a cost function $c : \mathcal{V} \times \mathcal{V} \to \mathcal{R}$, the optimal transport distance measures the optimal (i.e., cheapest) way to transport the mass in probability measure p_1 to match that in p_2:

$$W_c\left(p_1, p_2\right) = \inf_{\gamma \in \Pi(p_1, p_2)} \int_{\mathcal{V}\times\mathcal{V}} c\left(v_1, v_2\right) \gamma\left(dv_1, dv_2\right), \qquad (1)$$

where $\Pi(p_1, p_2)$ denotes the set of joint probability measures on $\mathcal{V} \times \mathcal{V}$ with p_1 and p_2 as marginals. When the cost function c is a metric or the p-th power with $p \geq 1$ of one over $\mathcal{V}$, Eq.1 is called the earth mover's distance, also Wasserstein distance.

In terms of the discrete measures, both marginals are in the simplex, i.e., $p_1,\ p_2 \in \Delta^{\mathcal{V}} = \{\mathbf{x} \in \mathcal{R}_+^V \mid \mathbf{x}^\intercal \mathbf{1} = 1\}$, where $\mathbf{1}$ is the all-one vector. The EMD thus can be written as follows:

$$W_c\left(p_1, p_2\right) = \inf_{\gamma \in \Pi(p_1, p_2)} \langle \gamma, C \rangle \qquad \gamma \mathbf{1} = p_1,\ \gamma^\intercal \mathbf{1} = p_2, \qquad (2)$$

where $\langle \cdot, \cdot \rangle$ stands for the Frobenius dot-product, $C \in \mathcal{R}^{V \times V}$ is the distance matrix computed by cost function c, also a metric. The time complexity of calculating the EMD between two V-dimensional discrete distributions scales at least in $O(V^3 \log(V))$ [24].

2.2 Word Mover's Distance

The WMD can be seemed as a special case of the discrete EMD, used for measuring distances among documents. Given a document d, we can represent it as a normalized V-dimensional term frequency vector:

$$tf_d = \left(\frac{tf_{d1}}{\sum_{v=1}^V tf_{dv}}, \frac{tf_{d2}}{\sum_{v=1}^V tf_{dv}}, \cdots, \frac{tf_{dV}}{\sum_{v=1}^V tf_{dv}} \right), \qquad (3)$$

where tf_{dv} is the term frequency of word v in document d, and V is the number of unique words.

One can consider that the above normalized term frequency vector as a discrete multinomial distribution over words, also a simplex. In this case, given two documents d_i and d_j, we can use the EMD to measure the distance between them, leading to the WMD, i.e., $W_c\left(tf_{d_i}, tf_{d_j}\right)$.

Here, the cost function is the distance or similarity of the corresponding word embeddings of words occurring in the documents, and also a metric over word embeddings. By using word embeddings, the WMD can take semantic information of words into consideration. In this sense, the WMD actually measures the optimal transport from one document to any other at the semantic level.

Regularized WMD. The authors of [6] formulated a smoothed version of EMD by introducing an entropic regularization term to the objective of EMD. This makes the objective strictly convex, so as to efficiently be solved by Sinkhorn-Knopp algorithm [30]. Following this, we can obtain a regularized WMD (RWMD), a.k.a. dual-Sinkhorn divergence [6], formulated by:

$$W_c^\lambda\left(tf_{d_i}, tf_{d_j}\right) = \left\langle \gamma_\lambda^*, C \right\rangle, \qquad (4)$$

where

$$\gamma_\lambda^* = \underset{\gamma \in \Pi(tf_{d_i}, tf_{d_j})}{\arg\min} \langle \gamma, C \rangle - \frac{1}{\lambda} H(\gamma),\ \gamma \mathbf{1} = tf_{d_i},\ \gamma^\intercal \mathbf{1} = tf_{d_j}, \qquad (5)$$

where $H(\cdot)$ denotes the entropy, and $\lambda > 0$ is a regularization parameter. With λ larger, the closer the regularized WMD is to the original EMD. Cuturi [6] shows that with Sinkhorn-Knopp algorithm [30] the regularized WMD can be solved in $O(V^2)$ time, which is significantly lower than computing the original EMD that costs at least in $O(V^3 \log(V))$ [24].

3 RWMD-BASED CENTROID CLASSIFIER

In this section, we describe the proposed RWMD-CC method for short texts.

3.1 Formulation

For clarity, we first give a formulation of the text classification task.

We use $x_d \in \Delta^{\mathcal{V}}$ and $y_d \in \mathcal{Y} = \{1, 2, \cdots, L\}$ to respectively denote the normalized term frequency vector and the label of short text d, where $\mathcal{V}$ is the set of unique words with size V and L is the number of labels. Then, a training dataset of D short texts is described by $S = \{x_d, y_d\}_{d=1}^{d=D}$. Our goal is to learn a classifier from S for predicting future short texts, i.e., learning a prediction function $f(x|\theta) \rightarrow y$ with parameter θ.

3.2 RWMD-CC

We begin with the centroid-based classification methodology. The mechanism is that given L category centroids $\{\theta_l \in \Delta^{\mathcal{V}}\}_{l=1}^{l=L}$, we predict any future short text x' by finding its closest category centroid.

The goal now is to learn these L category centroids from the training dataset S. Inspired by the prior art [33] and based on the structural risk minimization principle, we design a loss function of RWMD-CC using hypothesis margin, which involves both training errors (> 0) and inverse training margins (< 0). With a squared Frobenius norm regularizer, the loss function with repsect to θ is defined to be:

$$\begin{aligned}\mathcal{L}(\theta) = &\frac{1}{D}\left\{\sum_{d=1}^{D}\left|W_c^{\lambda}\left(g(x_d), g(\theta_R)\right) - W_c^{\lambda}\left(g(x_d), g(\theta_M)\right)\right|_{-}\right. \\ &\left. + \sum_{d=1}^{D}\eta\left|W_c^{\lambda}\left(g(x_d), g(\theta_R)\right) - W_c^{\lambda}\left(g(x_d), g(\theta_M)\right)\right|_{+}\right\} \\ &+ \frac{\lambda'}{2}\|\theta\|_F^2,\end{aligned} \tag{6}$$

where $|z|_{-} = min\{z, 0\}$, $|z|_{+} = max\{z, 0\}$; θ_R and θ_M denote the nearest category centroids to x_d under the RWMD measure with same and different label, respectively; $\eta \geq 1$ is a constant parameter to balance the training errors and inverse training margins in the loss function; $\|\cdot\|_F^2$ is the squared Frobenius norm, and $\lambda' \in [0, 1]$ is a regularization parameter. Here, we introduce $g(\cdot)$ with the constant importance vector $\mathbf{w}$:

$$g(x) = \frac{\mathbf{w} \circ x}{\mathbf{w}^{\intercal} x},$$

where $\circ$ represents the Hadamard product, to reflect the importance of words for distinguishing the categories in different classification tasks and datasets.

With the category centroids $\hat{\theta}$ obtained by optimizing $\mathcal{L}(\theta)$, for any future short text x', the prediction function of RWMD-CC is defined by:

$$y = \underset{l=1,\cdots,L}{\arg\min}\, W_c^{\lambda}\left(g(x'), g(\hat{\theta}_l)\right). \tag{7}$$

Besides, to obtain normalized centroids and avoid constrained conditions, we use the softmax transformation $\mathfrak{s}(\mu)$ for centroids θ with auxiliary parameters μ:

$$\theta = \mathfrak{s}(\mu), \qquad \theta_{lv} = \frac{\exp(\mu_{lv})}{\sum_{i=1}^{V}\exp(\mu_{li})}. \tag{8}$$

Combining Eq.6 and Eq.8, we reach the following final loss function with respect to the auxiliary parameters μ:

$$\mathfrak{L}(\mu) = \mathcal{L}\Big(\mathfrak{s}(\mu)\Big). \tag{9}$$

3.3 Optimization

We use gradient descent to minimize the loss function $\mathfrak{L}(\mu)$. In the following, we first show the optimization process, and then provide the implementation details.

3.3.1 The Optimization Algorithm of RWMD-CC. In gradient descent, given the gradient $\nabla\mathfrak{L}(\mu)$ and the learning rate φ, the parameters μ are updated by $\mu \leftarrow \mu - \varphi\nabla\mathfrak{L}(\mu)$. In the following, we mainly describe the computation of the gradient $\nabla\mathfrak{L}(\mu)$ and then outline the full algorithm.

The gradient $\nabla\mathfrak{L}(\mu)$. After some simple derivations, we show the gradient $\nabla\mathfrak{L}(\mu)$ as follows:

$$\begin{aligned}\nabla\mathfrak{L}(\mu) &= \frac{\partial\mathcal{L}\Big(\mathfrak{s}(\mu)\Big)}{\partial\mathfrak{s}(\mu)} \times \frac{\partial\mathfrak{s}(\mu)}{\partial\mu} \\ &= \theta \circ \frac{\partial\mathcal{L}(\theta)}{\partial\theta} - \theta\left(\theta^{\intercal}\frac{\partial\mathcal{L}(\theta)}{\partial\theta}\mathbf{I}_L\right),\end{aligned} \tag{10}$$

where $\mathbf{I}_L$ is the identity matrix with dimension L. The calculation of $\nabla\mathfrak{L}(\mu)$ only depends on $\frac{\partial\mathcal{L}(\theta)}{\partial\theta}$ that is given below.

[Computation of $\frac{\partial\mathcal{L}(\theta)}{\partial\theta}$]

For convenience, we re-write $\mathcal{L}(\theta)$ in Eq.6 as

$$\mathcal{L}(\theta) = \frac{1}{D}\left(\sum_{d=1}^{D}\delta\Big(h(x_d, \theta)\Big)h(x_d, \theta)\right) + \frac{\lambda'}{2}\|\theta\|_F^2, \tag{11}$$

where

$$\delta(z) = \begin{cases}\eta & \text{if } z \geq 0, \\ 1 & \text{if } z < 0;\end{cases}$$

$$h(x_d, \theta) = W_c^{\lambda}\left(g(x_d), g(\theta_R)\right) - W_c^{\lambda}\left(g(x_d), g(\theta_M)\right).$$

Consequently, the gradient of $\mathcal{L}(\theta)$ with respect to each category centroid θ_l can be computed by:

$$\frac{\partial\mathcal{L}(\theta)}{\partial\theta_l} = \frac{1}{D}\left(\sum_{d=1}^{D}\delta\Big(h(x_d, \theta)\Big)\frac{\partial h(x_d, \theta)}{\partial\theta_l}\right) + \lambda'\theta_l, \tag{12}$$

where

$$\frac{\partial h(x_d, \theta)}{\partial\theta_l} = \begin{cases}\frac{\partial W_c^{\lambda}\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)} \times \frac{\partial g(\theta_l)}{\partial\theta_l} & \text{if } \theta_R = \theta_l \\ -\left(\frac{\partial W_c^{\lambda}\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)} \times \frac{\partial g(\theta_l)}{\partial\theta_l}\right) & \text{if } \theta_M = \theta_l,\end{cases} \tag{13}$$

and

$$\frac{\partial g(\theta_l)}{\partial\theta_l} = \frac{\text{diag}(\mathbf{w}) - g(x)\mathbf{w}^{\intercal}}{\mathbf{w}^{\intercal}x}.$$

The computation of $\frac{\partial\mathcal{L}(\theta)}{\partial\theta_l}$ requires $\frac{\partial W_c^{\lambda}\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)}$, which, unfortunately, is intractably solved from the regularized WMD in Eq.4 directly. Inspired by the recent works on fast earth mover's distance computation [6, 7, 10], we change the regularized WMD

$W_c^\lambda\left(g(x_d), g(\theta_l)\right)$ to its dual problem and compute the gradient $\frac{\partial W_c^\lambda\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)}$ by solving the dual problem with well-known Sinkhorn-Knopp algorithm which is also used in [6, 7, 43]. We now show the details.

[Computation of $\frac{\partial W_c^\lambda\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)}$]

We first give the dual problem of the regularized WMD $W_c^\lambda\left(g(x_d), g(\theta_l)\right)$. Following [7], the Lagrange duality of regularized WMD $W_c^\lambda\left(g(x_d), g(\theta_l)\right)$ with dual variables $\alpha, \beta \in \mathcal{R}^V$ is given by:

$$^dW_c^\lambda\left(g(x_d), g(\theta_l)\right) = \sup_{(\alpha,\beta)} \Bigg(\alpha^\intercal g(x_d) + \beta^\intercal g(\theta_l) - \sum_{i,j} \frac{e^{-\lambda\left(C(i,j)-\alpha_i-\beta_j\right)}}{\lambda} \Bigg). \tag{14}$$

From [2], we know that the dual optimal β^* of Eq.14 is, in fact, the gradient $\frac{\partial W_c^\lambda\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)}$ that we need:

$$\frac{\partial W_c^\lambda\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)} = \beta^*. \tag{15}$$

Then, below we give the details of how to compute the dual optimal β^*. Following Proposition 2 of [7], when given $\mathbf{K} = e^{-\lambda C}$ which is the element-wise exponential of $-\lambda C$, there exists a pair of vectors $(\mathbf{u}, \mathbf{v}) \in \mathcal{R}^V \times \mathcal{R}^V$ such that the optimal solutions of regularized WMD, i.e., γ_λ^* of Eq.4 in Section 2.2 and its dual in Eq.14, i.e., β^*, are respectively given by

$$\gamma_\lambda^* = \mathrm{diag}(\mathbf{u})\mathbf{K}\mathrm{diag}(\mathbf{v}),\ \beta^* = -\frac{\log(\mathbf{v})}{\lambda} + \frac{\log(\mathbf{v})^\intercal \mathbf{1}}{\lambda V}\mathbf{1}. \tag{16}$$

where the term $\frac{\log(\mathbf{v})^\intercal \mathbf{1}}{\lambda V}\mathbf{1}$ is used to normalize β^* [7]. Besides, based on the Sinkhorn's theorem [30], the pair $(\mathbf{u}, \mathbf{v})$ can be recovered as a fixed point of the Sinkhorn map $(\mathbf{u}, \mathbf{v}) \leftarrow (g(x_d) \oslash \mathbf{K}\mathbf{v}, g(\theta_l) \oslash \mathbf{K}^\intercal \mathbf{u})$, where $\oslash$ represents the element-wise division. This fixed point iteration can also be written as a single update:

$$\mathbf{v} \leftarrow g(\theta_l) \oslash \left(\mathbf{K}^\intercal \left(g(x_d) \oslash \mathbf{K}\mathbf{v}\right)\right).$$

Hence, with vector $\mathbf{v}$, the gradient $\frac{\partial W_c^\lambda\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)}$ can be computed by combining Eq.15 and Eq.16:

$$\frac{\partial W_c^\lambda\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)} = \beta^* = -\frac{\log(\mathbf{v})}{\lambda} + \frac{\log(\mathbf{v})^\intercal \mathbf{1}}{\lambda V}\mathbf{1}. \tag{17}$$

In summary, we outline the detailed computation procedure of the gradient of the regularized WMD in *Algorithm 1*.

In regularized WMD, λ is the entropic regularization parameter as mentioned in Section 2.2. In our experiments, we use $\lambda = 10$ following [13].

Full algorithm. In summary, we outline the optimization procedure of RWMD-CC in *Algorithm 2*.

3.3.2 Implementation Details.

Algorithm 1 Gradient of the regularized WMD

Input: $g(x_d)$, $g(\theta_l)$, matrix $\mathbf{K}$, regularization parameter $\lambda > 0$;
Output: $\frac{\partial W_c^\lambda\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)}$.

Initialize $\mathbf{v} = \mathbf{1}$;
while $\mathbf{v}$ has not converged **do**
  $\mathbf{v} \leftarrow g(\theta_l) \oslash \left(\mathbf{K}^\intercal \left(g(x_d) \oslash \mathbf{K}\mathbf{v}\right)\right)$;
end while
$\frac{\partial W_c^\lambda\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)} \leftarrow -\frac{\log(\mathbf{v})}{\lambda} + \frac{\log(\mathbf{v})^\intercal \mathbf{1}}{\lambda V}\mathbf{1}$;

Algorithm 2 Optimization for RWMD-CC

Input: Distance matrix C, training dataset $S = \{x_d, y_d\}_{d=1}^{d=D}$, importance vector $\mathbf{w}$, regularization parameters λ, $\lambda' > 0$, and learning rate φ;
Output: $\mu = \{\mu_l\}_{l=1}^{l=L}$.

Initialize θ;
Calculate μ by the inverse of Eq.8 with θ;
Calculate $g(x_d)$ for $d = 1, \ldots, D$;
for $t = 1$ **to** N_{iter} **do**
  Calculate θ using Eq.8 with μ;
  Calculate $g(\theta_l)$ for $l = 1, \ldots, L$;
  for $d = 1$ **to** D **do**
    for $l = 1$ **to** L **do**
      Calculate $\frac{\partial W_c^\lambda\left(g(x_d), g(\theta_l)\right)}{\partial g(\theta_l)}$ using *Algorithm 1*;
      Calculate $\frac{\partial h(x_d, \theta)}{\partial \theta_l}$ using Eq.13;
    end for
  end for
  Calculate $\frac{\partial \mathcal{L}(\theta)}{\partial \theta_l}$ using Eq.12 for $l = 1, \ldots, L$;
  Calculate $\nabla\mathfrak{Q}(\mu)$ using Eq.10;
  $\mu \leftarrow \mu - \varphi\nabla\mathfrak{Q}(\mu)$;
end for

Initialization of category centroids θ. In RWMD-CC, we use *Class-Feature-Centroid* (CFC) [11] centroids as the initialization values of category centroids θ. In CFC centroids, the unnormalized mass of word v in category l is given by

$$\theta_{lv} = b^{\frac{DF_v^l}{D_l}} \bullet \log(\frac{L}{CF_v}), \tag{18}$$

where DF_v^l is word v's document frequency in category l; D_l is the number of documents in category l; CF_v is the number of categories containing word v; and b is a constant larger than one. Here, we set b to $e - 1.0$ (≈ 1.718) following [11].

Setting of importance vector $\mathbf{w}$. The constant importance vector $\mathbf{w}$ in RWMD-CC is set by using *balanced distributional concentration* (bdc) [37]. In bdc, the weight $\mathbf{w}_v$ of word v is given by:

$$\mathbf{w}_v = 1 + \frac{\sum_{l=1}^L \frac{p(v|l)}{\sum_{l=1}^L p(v|l)} \log\left(\frac{p(v|l)}{\sum_{l=1}^L p(v|l)}\right)}{\log(L)},$$

where $p(v|l) = \frac{TF_v^l}{TF^l}$ is the proportion of word v in its relevant category l, and TF_v^l, TF^l denote the frequency of word v in category l and the frequency sum of all words in category l, respectively.

3.4 Stochastic Optimization

The computation of the gradient $\frac{\partial \mathcal{L}(\theta)}{\partial \theta}$ needs to pass all training documents per-iteration and may be expensive for big datasets. To address this, we propose a stochastic optimization version of RWMD-CC, namely ***RWMD-CC***$_s$.

In RWMD-CC$_s$, we approximate noisy gradient of each category centroid θ_l by using a randomly selected subset S' per-iteration, instead of the whole training set S:

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta_l} \approx \frac{1}{D_{mini}} \left(\sum\nolimits_{\{x_d, y_d\} \in S'} \delta\Big(h(x_d, \theta)\Big) \frac{\partial h(x_d, \theta)}{\partial \theta_l} \right) + \lambda' \theta_l, \tag{19}$$

where $D_{mini} = |S'|$ is the mini-batch size. Since the size of S' is greatly less than that of S, i.e., $|S'| = D_{mini} \ll |S| = D$, the computation of gradients in RWMD-CC$_s$ is more effective than that in RWMD-CC, and we use RWMD-CC$_s$ in our experiments.

With the gradient $\frac{\partial \mathcal{L}(\theta)}{\partial \theta}$ approximated by Eq.19, $\nabla \mathfrak{L}(\mu)$ is calculated by Eq.10. In RWMD-CC$_s$, rather than using constant learning rate, we use *Adam* algorithm [14], which computes adaptive learning rate for each parameter, to update μ with the gradient $\nabla \mathfrak{L}(\mu)$.

3.5 Time Complexity

We now discuss the time complexity of RWMD-CC and RWMD-CC$_s$ for both training and testing phases, respectively. For clarity, we revisit some notations: L the number of category centroids, D the number of training documents, D_{mini} the mini-batch size in RWMD-CC$_s$, V the number of unique words; and introduce a new notation D_{avg} the average length of documents.

Training. For simplicity, we only discuss the per-iteration time complexity. In detail, the main time consuming part is the calculation of the gradient of the RWMD (step 9 in Algorithm 2). In our algorithms, we obtain the gradient of the RWMD between a document and a category centroid via the Sinkhorn-Knopp algorithm [6, 10], whose time complexity is $O(D_{avg}V)$. For RWMD-CC, we need to repeat this process between every document of training dataset and every category centroid, i.e., Eq.12, hence require DL times. Consequently, the per-iteration takes $O(DLD_{avg}V)$ time. Since one document in short texts only contains few words, e.g., $D_{avg} = 3.2$ in *Trec* dataset, each iteration of the training phase can be done in minutes. For RWMD-CC$_s$, we use the subset of training dataset with size D_{mini} to approximate the gradient of the objective, i.e., Eq.19, thus the time complexity of per-iteration is $O(D_{mini}LD_{avg}V)$, and per-iteration can be done in seconds.

Testing. When predicting a test document, both our RWMD-CC and RWMD-CC$_s$ only compute the RWMDs between the test document and all category centroids following the prediction function Eq.7. Hence the testing time complexity of our algorithms is linear with respect to L. However, for predicting a test document, WMD-based KNN classifiers, such as WMD+KNN, have to compare the WMDs between this test document and all documents of training datasets, so the testing time complexity of WMD-based KNN classifiers is a linear function of D. Obviously, our RWMD-CC and RWMD-CC$_s$ are much more efficient than WMD-based KNN classifiers. Actually, we can consider the centroid as a representative pseudo-document for each category. In this sense, the centroid classifier is a special case of K nearest neighbour (KNN) classifier with these pseudo-documents where $K = 1$.

We will further demonstrate the real run-time of training and testing phases in the experiment.

3.6 Generalization Error of RWMD-CC

Given a set of i.i.d. samples $S = \{x_d, y_d\}_{d=1}^{d=D}$, let $\hat{\theta}$ be the empirical risk minimizer

$$\hat{\theta} = \arg\min_{\theta \in \Theta} \Big\{ \hat{\mathbb{E}}_S \Big[\delta\Big(h(x, \theta)\Big) h(x, \theta) \Big] = \frac{1}{D} \left(\sum\nolimits_{d=1}^{D} \delta\Big(h(x_d, \theta)\Big) h(x_d, \theta) \right) \Big\}. \tag{20}$$

Further suppose $\Theta = \mathfrak{s} \circ (\Theta^o \times \cdots \times \Theta^o)$ is the composition of the softmax transformation $\mathfrak{s}$ and a L-valued function space with a base hypothesis space Θ^o of functions mapping into $\mathcal{R}^V$. The softmax transformation $\mathfrak{s}$ guarantees that θ_l lies in the simplex $\Delta^{\mathcal{V}}$.

Theorem 3.1. *For any $\tau > 0$, with probability at least $1 - \tau$, it holds that*

$$\mathbb{E}\Big[\delta\Big(h(x, \hat{\theta})\Big) h(x, \hat{\theta})\Big] \le \inf_{\theta \in \Theta} \mathbb{E}\Big[\delta\Big(h(x, \theta)\Big) h(x, \theta)\Big] + \eta\left(256V\sqrt{L^3}\|C\|_\infty \Re_D(\Theta^o) + 64\sqrt{L^3}\frac{\log V}{\lambda} + \|C\|_\infty \sqrt{\frac{2\log(1/\tau)}{D}}\right) + \|C\|_\infty \sqrt{\frac{2\log(1/\tau)}{D}} \tag{21}$$

with the constant $\|C\|_\infty = \max_{ij} C_{ij}$. $\Re_D(\Theta^o)$ is the Rademacher complexity [1] measuring the complexity of the hypothesis space Θ^o.

Remark. From Theorem 3.1, there is a constant error, which is introduced by regularized WMD, in risk bound. We can see that the constant error will vanish as $\lambda \to \infty$, which also coincides with the risk bound for the EMD with the standard convergence rate $O(1/\sqrt{D})$. However, as shown in [6], with λ larger, computing regularized WMD costs more time in same dimension. These mean that the value of λ reflects the trade-off between computation speed and approximation accuracy to some degree. Here, we set $\lambda = 10$ following [13].

4 RELATED WORK

In this section, we review some most related works on short text classification and classification using WMD.

4.1 Short Text Classification

Previous works on short text classification are mainly based on feature expansion. They can be roughly divided into two classes. One is to extract auxiliary context information using search engines [8, 29, 32]. Traditional classification algorithms are then applied to such expanded short texts, which are deemed as long pseudo-documents. An limitation is that such algorithms must require high quality search results. The other is to expand document features by

employing external knowledgebases such as Wikipedia and WordNet [12, 21, 25, 35]. For example, [25] uses hidden topics of LDA learnt from Wikipedia to enrich document-level features. These algorithms perform well, however, they may be problematic when the pre-trained topics and concepts are inconsistent with the current dataset for certain applications.

Recently, some deep learning-based classifiers have been proposed [9, 36, 42]. To our knowledge, an early representative work is the character to sentence convolutional neural network (CharSCNN) [9]. As suggested by the name, the CharSCNN is based on CNN, and it exploits from character- to sentence-level information to perform short text classification. Other deep learning classifiers are also based on the CNN architecture, and the pre-trained word embeddings are used as the input.

Additionally, many researchers have developed generic short text topic models, e.g., biterm topic model (BTM) [5] and word network topic model [44]. They learn topic representations of short texts and then use a traditional algorithm, e.g., SVMs, as a downstream classifier. Such methodology empirically performed well on classification evaluations of short texts.

4.2 Classification Using WMD

Since the WMD is a measure of document distance, a most straightforward classification method is WMD+*K*NN [15]. A recent work [13] incorporates the supervision (i.e., labels) into a WMD learning procedure, leading to supervised WMD (S-WMD). Empirically, S-WMD+*K*NN performed better than WMD+*K*NN. In contrast to those methods, our RWMD-CC computes a representative centroid for each category. Roughly, RWMD-CC can reduce the number of calculating WMDs from D to L when predicting each test document.

5 EXPERIMENTS

In this section, we empirically compare our RWMD-CC against the existing baseline algorithms of short texts.

In early experiments, we have evaluated the performance of RWMD-CC and RWMD-CC$_s$ (the stochastic optimization version of RWMD-CC) on all the datasets, and found that the performance gap between RWMD-CC and RWMD-CC$_s$ is very small. That is, the stochastic optimization does not affect the performance of RWMD-CC. Hence we only show RWMD-CC$_s$ in the experiments.

5.1 Experimental Settings

Datasets. We choose three datasets of short texts, including *Trec*[1], *biomedical*[2] and *StackOverFlow*[2]. For all datasets, we remove the standard stopwords, words with term frequency less than 5, and empty documents. The statistics of datasets are shown in Table 2. Besides, we employ pre-trained *GloVe*[3] word embeddings and randomly set embeddings of the words that are not included in the vocabulary of pre-trained word embeddings. We use the cosine distance, i.e., $1 - cos(w_i, w_j)$, which is used to measure the distance between any word embeddings pair $\{w_i, w_j\}$, as the cost function c in the regularized WMD to compute the distance matrix C.

[1]http://trec.nist.gov/data/tweets/
[2]https://github.com/jacoxu/STC2
[3]https://nlp.stanford.edu/projects/glove/. The 100-dimensional vectors trained on Wikipedia 2014 and Gigaword 5 are used.

Table 2: Statistics of datasets. *#doc*: the number of documents. *#word*: the number of unique words. D_{avg}: the average document length. *#label*: the number of categories. *Rate*: the word cover rate of GloVe word embeddings.

Dataset	*#doc*	*#word*	D_{avg}	*#label*	*Rate* (%)
Trec	5,864	1,058	3.2	6	99.3
biomedical	19,970	4,472	7.3	20	94.7
StackOverFlow	19,768	2,651	3.9	20	78.7

Baseline algorithms. We choose four existing algorithms for comparison. Details are given as below.

- **TFIDF + *K*NN**: We use tf-idfs as the representations of short texts, and then perform *K*NN as a downstream classifier with counting the similarities between short texts by cosine measure.
- **BTM + SVMs**: We first learn the topic representations of short texts using BTM [5], and then perform SVMs[4] as a downstream classifier.
- **WMD + *K*NN**: This is the first use of WMD for classification in its original paper [15]. The code is available on the net[5].
- **S-WMD + *K*NN**: S-WMD [13] is a supervised extension of WMD, and also uses stochastic optimization. The code is available on the net[6].

For the *K*NN-based methods, the neighborhood number K is tested from $\{1, \ldots, 19\}$. For BTM+SVMs, WMD+*K*NN and S-WMD+*K*NN, we use default parameter settings in their papers and codes, except the mini-batch size of S-WMD+*K*NN, which is set to 256.

Evaluation metrics. The comparison is based on two popular classification metrics, i.e., macro-averaging F1 and micro-averaging F1. The macro-averaging F1 (Macro-F1) and micro-averaging F1 (Micro-F1) are the two different types of average of F1 score and used to measure the classification performance for the whole corpus. The F1 score is a combined form of precision (π) and recall (ρ):

$$\pi_l = \frac{TP_l}{TP_l + FP_l}, \quad \rho_l = \frac{TP_l}{TP_l + FN_l}, \quad F1_l(\pi_l, \rho_l) = \frac{2\pi_l\rho_l}{\pi_l + \rho_l},$$

where TP_l (True Positives) is the number of documents assigned correctly to category l; FP_l (False Positives) is the number of documents that do not belong to category l but are assigned incorrectly to it by classifier; FN_l (False Negatives) is the number of documents that belong to category l but are assigned incorrectly to other ones by classifier. Specifically, Micro-F1 and Macro-F1 are the F1 score of the whole dataset and the average F1 score of all categories, respectively.

Parameter settings. For RWMD-CC$_s$, in our experiments, its parameters are empirically set as follows: $\lambda = 10$, $b = e - 1.0$, $\eta = 60.0$, $\lambda' = 0.001$, $D_{mini} = 256$ and $N_{iter} = 50$. All the experiments are carried on a Linux server with Intel Xeon 3.10 GHz CPU and 16G memory.

[4]http://scikit-learn.org/stable/
[5]https://github.com/mkusner/wmd
[6]https://github.com/gaohuang/S-WMD

Table 3: Results of classification accuracy (mean±std) with 5-fold cross validation. The best results of short text classifiers are highlighted in boldface. "‡" means that the gain of RWMD-CC$_s$ is statistically significant at 0.05 level.

Metric	Dataset	BoW		WMD		
		TFIDF + *KNN*	BTM + SVMs	WMD + *KNN*	S-WMD + *KNN*	RWMD-CC$_s$
Micro-F1	*Trec*	0.8281 ± 0.0063‡	0.6223 ± 0.0122‡	0.9389 ± 0.0086	**0.9543 ± 0.0031**	0.9425 ± 0.0081
	biomedical	0.5596 ± 0.0072‡	0.5129 ± 0.0071‡	0.6063 ± 0.0212‡	0.6255 ± 0.0056‡	**0.6763 ± 0.0057**
	StackOverFlow	0.6613 ± 0.0083‡	0.5880 ± 0.0053‡	0.7914 ± 0.0113‡	0.7847 ± 0.0130‡	**0.8470 ± 0.0062**
Macro-F1	*Trec*	0.8477 ± 0.0092‡	0.6074 ± 0.0178‡	0.9444 ± 0.0063	**0.9585 ± 0.0018**	0.9428 ± 0.0087
	biomedical	0.5534 ± 0.0077‡	0.5195 ± 0.0065‡	0.5971 ± 0.0225‡	0.6226 ± 0.0052‡	**0.6677 ± 0.0060**
	StackOverFlow	0.6616 ± 0.0079‡	0.6095 ± 0.0044‡	0.7899 ± 0.0111‡	0.7842 ± 0.0130‡	**0.8460 ± 0.0057**

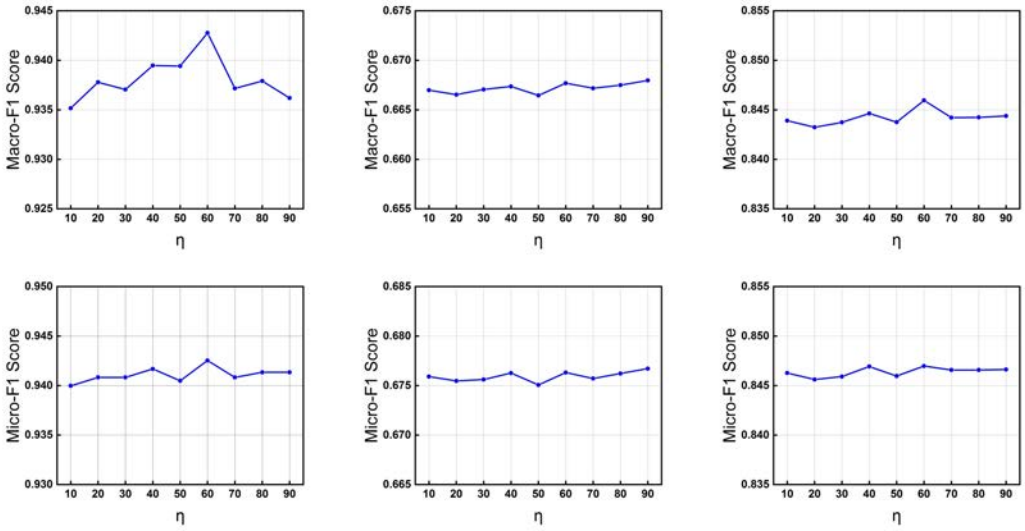


Figure 1: Classification performance by varying η on *Trec* (left), *biomedical* (middle) and *StackOverFlow* (right).

5.2 Performance Comparison

For all methods, we perform a 5-fold cross validation and report the average scores. The classification results over all datasets are shown in Table 3. Overall, our RWMD-CC$_s$ performs better than the baselines in most cases. Some observations are made as follows:

Comparing RWMD-CC$_s$ against baselines: First, RWMD-CC$_s$ consistently outperforms BoW-based methods on Micro-F1 and Macro-F1 scores, especially the scores of RWMD-CC$_s$ are even about 0.18 higher than those of TFIDF+*K*NN on *StackOverFlow*. Second, RWMD-CC$_s$ performs better than other WMD-based methods on *biomedical* and *StackOverFlow*, e.g., about 0.06 higher than S-WMD+*K*NN on *StackOverFlow*. RWMD-CC$_s$ only has a small decrease compared with S-WMD+*K*NN on *Trec*. The possible reason is the category-imbalanced nature of *Trec*, in which the number of documents of each category is {1300, 1309, 95, 1288, 1005, 867}, and thus some documents of other categories may be misclassified into the category, which contains fewer documents, for centroid-based classifiers.

Comparing WMD-based methods against BoW-based methods: WMD-based methods (including WMD+*K*NN, S-WMD+*K*NN and RWMD-CC$_s$) significantly dominate BoW-based methods (including TFIDF+*K*NN and BTM+SVMs) on two evaluation metrics on all three datasets. For example, the Micro-F1 scores of WMD+*K*NN beat TFIDF+*K*NN by 0.11, 0.05 and 0.13 on *Trec*, *biomedical*, and *StackOverFlow*, respectively. It means that for short text classification the WMD-based classifiers are much better than the classifiers based on BoW representations. The result is expected since the WMD can effectively measure semantic distances among documents with word embeddings and hence reduce the document similarity misalignment, especially for short texts. This is also an evidence of that the WMD is a better distance measure than the distances or similarities based on BoW.

Observation about word embeddings: The classification performance of RWMD-CC$_s$ is insensitive to the word embeddings. For instance, on *StackOverFlow* the performance of RWMD-CC$_s$ is still very competitive even the cover rate of pre-trained GloVe word

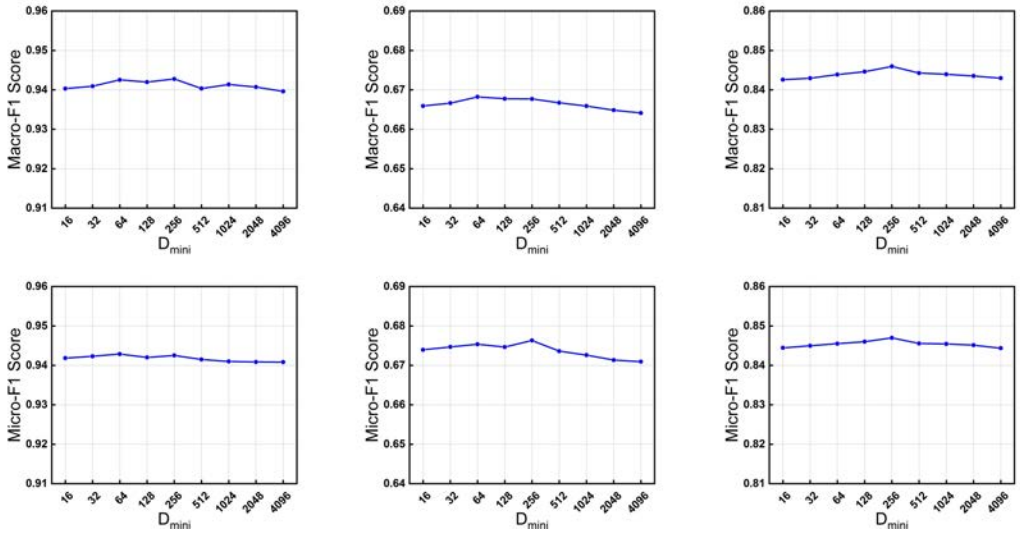


Figure 2: Classification performance by varying D_{mini} on *Trec* (left), *biomedical* (middle) and *StackOverFlow* (right).

Table 4: Time cost of WMD+*K*NN, S-WMD+*K*NN and RWMD-CC$_s$ on *Trec* (top section), *biomedical* (middle section) and *StackOverFlow* (bottom section). Init. Time: time cost during initializing parameters. Iter. Time: per-iteration time in the training phase. Test Time: time cost of predicting all test documents. h: hours, m: minutes, s: seconds.

Method	Init. Time	Iter. Time	Test Time
WMD+*K*NN	–	–	8m 49s
S-WMD+*K*NN	21m 57s	2s	8m 17s
RWMD-CC$_s$	–	1s	18s
WMD+*K*NN	–	–	1h 46m
S-WMD+*K*NN	4h 34m	3s	1h 35m
RWMD-CC$_s$	–	3s	5m 38s
WMD+*K*NN	–	–	3h 4m
S-WMD+*K*NN	4h 52m	4s	2h 44m
RWMD-CC$_s$	–	2s	3m 40s

embeddings is only 78.7%. Therefore, we argue that RWMD-CC$_s$ is practical even big collections of word embeddings lack.

5.3 Efficiency Comparison

We compare the running time of the three WMD-based methods, including RWMD-CC$_s$, WMD+*K*NN and S-WMD+*K*NN.

We compare these algorithms in initializing time, per-iteration training time and testing time. To be fair, we use the following settings: (a) for all datasets, we randomly select 20% from full dataset as the test set and remain 80% as the training set; (b) for all methods, we set the entropic regularization parameter λ of regularized WMD to 10; (c) specifically for RWMD-CC$_s$ and S-WMD+*K*NN, we set the mini-batch size to 256 in their training phases. All experiments are carried with **single thread** on a Linux server with Intel Xeon 3.10 GHz CPU and 16G memory. Table 4 shows the running time results averaged on 100 runs. Some observations are made as follows:

Comparison of testing time: RWMD-CC$_s$ is significantly faster than WMD+*K*NN and S-WMD+*K*NN when predicting test documents, e.g., WMD+*K*NN and S-WMD+*K*NN cost about 3 hours on *StackOverFlow* dataset, as comparison, RWMD-CC$_s$ merely 4 minutes. This is consistent with the time complexity analysis mentioned in the section 3.5. That is because when predicting a test document, RWMD-CC$_s$ only calculates the WMDs between this test document and the category centroids, WMD+*K*NN and S-WMD+*K*NN, in contrast, have to compute the WMDs between it and all documents of training datasets. We thus argue that RWMD-CC$_s$ is a more practical choice, especially for real world applications, where fast response is required.

Comparison of training time: Thanks to stochastic optimization, both the training phases of RWMD-CC$_s$ and S-WMD+*K*NN spend very little time. Therefore, the training of RWMD-CC$_s$ is also efficient for big corpora.

Comparison of initializing time: S-WMD+*K*NN takes huge time during initializing parameters, e.g., about 5 hours on *StackOverFlow* dataset, in contrast, our RWMD-CC$_s$ spends no time for initialization. This again demonstrates RWMD-CC$_s$ is more practical.

5.4 Evaluation of Parameters

In this subsection, we empirically evaluate two crucial parameters of RWMD-CC$_s$, including the trade-off parameter η and mini-batch size D_{mini}.

We first evaluate the impact of different η values over the set $\{10, 20, \cdots, 90\}$. The results are shown in Figure 1. Roughly, both Macro-F1 and Micro-F1 scores of RWMD-CC$_s$ have a small change in all settings with the η value varying from 10 to 90. That is to say,

RWMD-CC$_s$ is insensitive to η, making it more robust in real world applications. Besides, the best scores are achieved at $\eta = 60$ in most cases. We thus fix $\eta = 60$ in our experiments, and suggest choosing the value of η from 50 to 70 for short texts in practice.

Then, we examine the classification performance of different D_{mini} values over the set $\{16, 32, \cdots, 4096\}$. As the results shown in Figure 2, we argue that RWMD-CC$_s$ is insensitive to the mini-batch size D_{mini}. Furthermore, the best performance is achieved when $D_{mini} = 256$. Thus, in the experiments, we use $D_{mini} = 256$ as the default setting of RWMD-CC$_s$, which is also the suggested value in practice.

6 CONCLUSION

In this paper, we propose a RWMD-CC method for extremely short text classification. Our RWMD-CC is based on the popular document distance, e.g., WMD, which measures the distance at the semantic level. We learn a representative semantic centroid for each category, and predict any text sample depending on the WMDs between this sample and all semantic centroids. To achieve this, we build a loss function with hypothesis margin that is based on the structural risk minimization principle. For fast optimization, we follow the spirit of the regularized WMD and stochastic optimization. Experimental results indicate that our RWMD-CC outperforms the state-of-the-art baseline methods, and more importantly it is much more efficient than the existing WMD-based KNN methods during prediction.

A STATISTICS LEARNING BOUNDS

We establish the proof of Theorem 3.1 in this section. For simpler notation, given a sequence $S = \{x_d, y_d\}_{d=1}^{d=D}$ of i.i.d. training samples, we denote the empirical risk $\hat{R}_S$ and expected risk R as

$$\hat{R}_S(\theta) = \hat{\mathbb{E}}_S\Big[\delta\Big(h(x,\theta)\Big)h(x,\theta)\Big] = \frac{1}{D}\Big(\sum\nolimits_{d=1}^{D}\delta\Big(h(x_d,\theta)\Big)h(x_d,\theta)\Big),$$
$$R(\theta) = \mathbb{E}\Big[\delta\Big(h(x,\theta)\Big)h(x,\theta)\Big]. \tag{22}$$

where

$$\delta(z) = \begin{cases}\eta & \text{if } z \geq 0,\\ 1 & \text{if } z < 0;\end{cases} \qquad h(x_d,\theta) = W_c^{\lambda}\Big(x_d,\theta_R\Big) - W_c^{\lambda}\Big(x_d,\theta_M\Big),$$

θ_R and θ_M denote the nearest category centroids to x_d under RWMD measure with same label and different label, respectively.

Lemma A.1 (Lemma B.1 of [10]). *Let $\hat{\theta}$, $\theta^* \in \Theta$ be the minimizer of the empirical risk $\hat{R}_S$ and expected risk R, respectively. Then*

$$R(\hat{\theta}) \leq R(\theta^*) + 2\sup_{\theta\in\Theta}|R(\theta) - \hat{R}_S(\theta)|.$$

To bound the risk for $\hat{\theta}$, we need to establish a uniform concentration bound for the loss function $\mathcal{L}(\theta)$ in Eq.11. Therefore, we define a loss function space induced by the hypothesis space Θ as

$$\mathcal{H} = \Big\{\mathfrak{h} : x \mapsto \delta\Big(h(x,\theta)\Big)h(x,\theta) : \theta \in \Theta\Big\}. \tag{23}$$

The uniform concentration bound depends on the "complexity" of $\mathcal{H}$, which is measured by the empirical *Rademacher complexity* defined below.

Definition A.2 (Rademacher Complexity [1]). Let $\mathcal{F}$ be a family of mapping from $\mathcal{Z}$ to $\mathcal{R}$, and $S = (z_1, \ldots, z_D)$ a fixed sample from $\mathcal{Z}$. The empirical Rademacher complexity of $\mathcal{F}$ with respect to S is defined as

$$\hat{\mathfrak{R}}_S(\mathcal{F}) = \mathbb{E}_\sigma\Big[\sup_{f\in\mathcal{F}}\frac{1}{D}\sum_{d=1}^{D}\sigma_d f(z_d)\Big], \tag{24}$$

where $\sigma = (\sigma_1, \ldots, \sigma_d, \ldots, \sigma_D)$, with σ_d's independent uniform random variables taking values in $\{+1, -1\}$. σ_d's are called the Rademacher random variables. The Rademacher complexity is defined by taking expectation with respect to the samples S,

$$\mathfrak{R}_D(\mathcal{F}) = \mathbb{E}_S\Big[\hat{\mathfrak{R}}_S(\mathcal{F})\Big].$$

Theorem A.3. *Let $\mathcal{H} = \Big\{\mathfrak{h} : x \mapsto \delta\Big(h(x,\theta)\Big)h(x,\theta) : \theta \in \Theta\Big\}$. For any $\tau > 0$, with probability at least $1 - \tau$, the following holds for all $\mathfrak{h} \in \mathcal{H}$,*

$$\mathbb{E}[\mathfrak{h}] - \hat{\mathbb{E}}_S[\mathfrak{h}] \leq 2\mathfrak{R}_D(\mathcal{H}) + (\eta+1)\|C\|_\infty\sqrt{\frac{\log(1/\tau)}{2D}}, \tag{25}$$

where $\mathfrak{R}_D(\mathcal{H})$ is Rademacher complexity of the loss function space $\mathcal{H}$ associated to Θ, and $\|C\|_\infty = \max_{ij} C_{ij}$.

Following the definition of $\mathcal{H}$, $\mathbb{E}[\mathfrak{h}] = R(\theta)$ and $\hat{\mathbb{E}}_S[\mathfrak{h}] = \hat{R}_S(\theta)$. Hence, a uniform control for the deviation of the empirical risk from the expected risk is given by the Theorem A.3.

Theorem A.4 (McDiarmid's Inequality). *Let $S = \{X_1, \ldots, X_D\}$ be i.i.d. random variables with size D from $\mathcal{X}$. Assume there exists $C > 0$ such that $f : \mathcal{X}^D \to \mathcal{R}$ satisfies the following stability condition*

$$|f(x_1, \ldots, x_d, \ldots, x_D) - f(x_1, \ldots, x'_d, \ldots, x_D)| \leq C \tag{26}$$

for all $d = 1, \ldots, D$ and any $x_1, \ldots, x_D, x'_d \in \mathcal{X}$. Then for any $\varepsilon > 0$, denoting $f(X_1, \ldots, X_D)$ by $f(S)$, it holds that

$$\mathbb{P}(f(S) - \mathbb{E}[f(S)] \geq \varepsilon) \leq \exp\Big(-\frac{2\varepsilon^2}{DC^2}\Big). \tag{27}$$

Lemma A.5. *Let the constant $\|C\|_\infty = \max_{ij} C_{ij}$ is the maximum in the distance matrix C, then $0 \leq W_c^{\lambda}(\cdot,\cdot) \leq \|C\|_\infty$.*

Proof. For any x and θ_l, let γ_λ^* be the relaxed transport plan that solves (5), then

$$W_c^{\lambda}(x,\theta_l) = \Big\langle\gamma_\lambda^*, C\Big\rangle \leq \|C\|_\infty\sum_{v_1,v_2}\gamma_{v_1,v_2} = \|C\|_\infty.$$

□

Lemma A.6. *Let the constant $\|C\|_\infty = \max_{ij} C_{ij}$ is the maximum in the distance matrix C, then $-\|C\|_\infty \leq h(x,\theta) \leq \|C\|_\infty$, $-\|C\|_\infty \leq \delta\Big(h(x,\theta)\Big)h(x,\theta) \leq \eta\|C\|_\infty$.*

Lemma A.6 can be easily proved following the definitions of $\delta(z)$ and $h(x,\theta)$ with Lemma A.5.

Proof of Theorem A.3. Inspired by the proof of Theorem B.3 of [10], the Theorem A.3 can be easily proved by using Lemma A.6 and Theorem A.4. The details can be found in the proof of Theorem B.3 of [10]. □

Now, we establish the Rademacher complexity $\mathfrak{R}_D(\mathcal{H})$. For each instance (x_d, y_d) from S, we denote the category centroids θ_l with the same label y_d as θ_l^+ and the ones with a different label than y_d as θ_l^-, then $\theta_R \in \{\theta_l^+\}$ and $\theta_M \in \{\theta_l^-\}$. Thus, we can re-write the function $h(x,\theta) = W_c^\lambda(x,\theta_R) - W_c^\lambda(x,\theta_M)$ as

$$h(x,\theta) = \left(\min_{\theta_l^+}\left\{W_c^\lambda(x,\theta_l^+)\right\} - \min_{\theta_l^-}\left\{W_c^\lambda(x,\theta_l^-)\right\}\right). \tag{28}$$

Then, we define a function $\mathfrak{m} : [-\|C\|_\infty, \|C\|_\infty] \to \mathcal{R}$ and a real function space $\mathcal{F}$ as follows:

$$\mathfrak{m} : z \mapsto z\delta(z), \tag{29}$$

$$\mathcal{F} = \left\{\mathfrak{f} : (x,\vartheta) \mapsto W_c^\lambda(x,\vartheta)\right\}. \tag{30}$$

Theorem A.7. *Let the function $\mathfrak{m}$ and the function space $\mathcal{F}$ be defined as above. For the loss function space $\mathcal{H}$, it holds that*

$$\mathfrak{R}_D(\mathcal{H}) \le 16\eta\sqrt{L^3}\mathfrak{R}_D(\mathcal{F}). \tag{31}$$

Theorem A.8 (Rademacher Vector Contraction Inequality [22]). *Let $\mathcal{F}$ be a class of real functions, and $\mathcal{H} \subset \mathcal{F} = \mathcal{F}_1 \times \cdots \times \mathcal{F}_K$ be a K-valued function class. If $\mathfrak{t} : \mathcal{R}^K \mapsto \mathcal{R}$ is a L_m-Lipschitz continuous function and $\mathfrak{t}(\mathbf{0}) = 0$, then*

$$\hat{\mathfrak{R}}_S(\mathfrak{t} \circ \mathcal{H}) \le \sqrt{2}L_m \sum_{k=1}^{K} \hat{\mathfrak{R}}_S(\mathcal{F}_k). \tag{32}$$

Proof of Theorem A.7. With the function $\mathfrak{m}$, the loss function space $\mathcal{H} = \left\{\mathfrak{h} : x \mapsto \delta\Big(h(x,\theta)\Big)h(x,\theta) : \theta \in \Theta\right\}$ can be written as $\mathcal{H} = \mathfrak{m} \circ \mathcal{G}$, where $\mathcal{G} = \left\{\mathfrak{g} : x \mapsto h(x,\theta) : \theta \in \Theta\right\}$ is a function space.

Now, we first show the function $\mathfrak{m}$ is Lipschitz continuous with constant η, then give $\mathfrak{R}_D(\mathcal{G})$, which is the Rademacher complexity of $\mathcal{G}$.

With the definition of function $\mathfrak{m}$ and $\delta, \eta \ge 1$, it holds that

$$|\mathfrak{m}(z_1) - \mathfrak{m}(z_2)| = \left\{\begin{array}{ll} |\eta z_1 - \eta z_2| = \eta|z_1 - z_2| & \text{if } z_1, z_2 \ge 0 \\ |\eta z_1 - z_2| = \eta z_1 + (-z_2) & \text{if } z_1 \ge 0, z_2 < 0 \\ |z_1 - \eta z_2| = (-z_1) + \eta z_2 & \text{if } z_1 < 0, z_2 \ge 0 \\ |z_1 - z_2| & \text{if } z_1, z_2 < 0 \end{array}\right\} \le \eta|z_1 - z_2|.$$

Thus, $\mathfrak{m}$ is Lipschitz with constant η and $\mathfrak{m}(0) = 0$.

As next step, we establish the Rademacher complexity $\mathfrak{R}_D(\mathcal{G})$. Following Eq.28, the functions $\mathfrak{g}$ contained in $\mathcal{G}$ can be expressed as the sum of two terms of identical form, one multiplied by -1, such as

$$x \mapsto \min_{\theta_l}\left\{W_c^\lambda(x,\theta_l)\right\}, \tag{33}$$

where the minimum is taken over at most L terms, L denotes the number of category centroids. Let $\mathcal{M}$ be the function space, in which the form of functions is Eq.33. Note that $\mathfrak{R}_D(\sum_{k=1}^K \mathcal{F}_k) \le \sum_{k=1}^K \mathfrak{R}_D(\mathcal{F}_k)$ holds for all real function classes $\mathcal{F}_k$ by Theorem 12 of [1] due to the triangle inequality. Further, following Theorem 12 of [1] multiplying a function space by -1 does not change its Rademacher complexity. Therefore, we can upper bound $\mathfrak{R}_D(\mathcal{G})$ by twice the complexity of the function space $\mathcal{M}$:

$$\mathfrak{R}_D(\mathcal{G}) \le 2\mathfrak{R}_D(\mathcal{M}). \tag{34}$$

Following [28], the function that calculates the minimum of L values as follows,

$$(z_1, \ldots, z_L) \mapsto \min\{z_1, \ldots, z_L\},$$

is Lipschitz continuous with constant $\sqrt{8L}$. We apply Theorem A.8 to the $\sqrt{8L}$-Lipschitz continuous function min and the function space

$$\underbrace{\mathcal{F} \times \cdots \times \mathcal{F}}_{L \text{ copies}}.$$

It holds

$$\hat{\mathfrak{R}}_S(\mathcal{M}) \le \sqrt{2}\sqrt{8L}\sum_{l=1}^{L}\hat{\mathfrak{R}}_S(\mathcal{F}) = 4\sqrt{L^3}\hat{\mathfrak{R}}_S(\mathcal{F}). \tag{35}$$

Then, following Theorem 12 of [1] and combining Eq.34 with Eq.35, we have

$$\mathfrak{R}_D(\mathcal{H}) = \mathfrak{R}_D(\mathfrak{m} \circ \mathcal{G}) \le 2\eta\mathfrak{R}_D(\mathcal{G}) \le 16\eta\sqrt{L^3}\mathfrak{R}_D(\mathcal{F}).$$

□

Before establishing $\mathfrak{R}_D(\mathcal{F})$, we provide some useful properties of regularized WMD.

Lemma A.9 (Lemma 2 of [43]). *For two discrete measures $p_1,\ p_2 \in \Delta^{\mathcal{V}}$, the regularized WMD $W_c^\lambda(p_1,p_2)$ and the EMD $W_c(p_1,p_2)$ satisfy the following relationship,*

$$W_c(p_1,p_2) \le W_c^\lambda(p_1,p_2) \le W_c(p_1,p_2) + \frac{2}{\lambda}\log V,$$

where $V = |\mathcal{V}|$.

Then, we define a function space $\mathcal{F}_{EMD}$ corresponding to the family of the EMD as follows:

$$\mathcal{F}_{EMD} = \left\{\mathfrak{f}_{EMD} : (x,\vartheta) \mapsto W_c(x,\vartheta)\right\}.$$

From Theorem 4 of [43], the Rademacher complexities of $\mathcal{F}$ and $\mathcal{F}_{EMD}$ satisfy

$$\mathfrak{R}_D(\mathcal{F}) \le \mathfrak{R}_D(\mathcal{F}_{EMD}) + \frac{\log V}{\lambda}. \tag{36}$$

In the following, we relate the Rademacher complexity of $\mathcal{F}_{EMD}$ with the hypothesis space Θ^o by using the similar processing method in [10] and give the proof of Theorem 3.1.

Proof of Theorem 3.1. Consider a function space $\mathcal{F}_{EMD}$ with a softmax transformation $\mathfrak{s}$

$$\mathcal{F}_{EMD} = \left\{f_{EMD} : (x,\mu) \mapsto W_c(\mathfrak{s}(x),\mathfrak{s}(\mu))\right\},$$

From Proposition B.10 of [10], we know that the $\mathcal{R}^V \times \mathcal{R}^V \mapsto \mathcal{R}$ map $\iota : (z,z') \mapsto W_c(\mathfrak{s}(z),\mathfrak{s}(z'))$ is a $2\sqrt{2}\|C\|_\infty$-Lipschitz continuous function and $\iota(\mathbf{0},\mathbf{0}) = 0$. We apply Theorem A.8 to the $2\sqrt{2}\|C\|_\infty$-Lipschitz continuous function ι, and it holds

$$\hat{\mathfrak{R}}_S(\mathcal{F}_{EMD}) \le \sqrt{2}\cdot 2\sqrt{2}\|C\|_\infty V\hat{\mathfrak{R}}_S(\Theta^o). \tag{37}$$

The conclusion of Theorem 3.1 follows by combining Eq.37 with Eq.36, Theorem A.7, Theorem A.3 and Lemma A.1. □

ACKNOWLEDGMENTS

We would like to acknowledge support for this project from the National Natural Science Foundation of China (NSFC) (No.61876071, No.61602204).

REFERENCES

[1] Peter L. Bartlett and Shahar Mendelson. 2002. Rademacher and Gaussian Complexities: Risk Bounds and Structural Results. *Journal of Machine Learning Research* 3 (2002), 463–482.

[2] Dimitris Bertsimas and John N Tsitsiklis. 1997. *Introduction to Linear Optimization*. Athena Scientific Belmont, MA.

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3 (2003), 993–1022.

[4] Vladimir Igorevich Bogachev and Aleksandr Viktorovich Kolesnikov. 2012. The Monge-Kantorovich Problem: Achievements, Connections, and Perspectives. *Russian Mathematical Surveys* 67, 5 (2012), 785–890.

[5] Xueqi Cheng, Xiaohui Yan, Yanyan Lan, and Jiafeng Guo. 2014. BTM: Topic Modeling over Short Texts. *IEEE Transactions on Knowledge and Data Engineering* 26, 12 (2014), 2928–2941.

[6] Marco Cuturi. 2013. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. In *Neural Information Processing Systems*. 2292–2300.

[7] Marco Cuturi and Arnaud Doucet. 2014. Fast Computation of Wasserstein Barycenters. In *International Conference on Machine Learning*. 685–693.

[8] Honghua (Kathy) Dai, Lingzhi Zhao, Zaiqing Nie, Ji-Rong Wen, Lee Wang, and Ying Li. 2006. Detecting Online Commercial Intention (OCI). In *International Conference on World Wide Web*. 829–837.

[9] Cícero Nogueira dos Santos and Maíra Gatt. 2014. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In *International Conference on Computational Linguistics*. 69–78.

[10] Charlie Frogner, Chiyuan Zhang, Hossein Mobahi an Mauricio Araya-Polo, and Tomaso Pogglo. 2015. Learning with a Wasserstein Loss. In *Neural Information Processing Systems*. 2053–2061.

[11] Hu Guan, Jingyu Zhou, and Minyi Guo. 2009. A Class-Feature-Centroid Classifier for Text Categorization. In *International Conference on World Wide Web*. 201–210.

[12] Xia Hu, Nan Sun, Chao Zhang, and Tat-Seng Chua. 2009. Exploiting Internal and External Semantics for the Clustering of Short Texts Using World Knowledge. In *ACM International Conference on Web Search and Data Mining*. 353–362.

[13] Gao Huang, Chuan Guo, Matt J. Kusner, Yu Sun, Kilian Q. Weinberger, and Fei Sha. 2016. Supervised Word Mover's Distance. In *Neural Information Processing Systems*. 4862–4870.

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From Word Embeddings To Document Distances. In *International Conference on Machine Learning*. 957–966.

[16] Ximing Li, Jinjin Chi, Changchun Li, Jihong Ouyang, and Bo Fu. 2016. Integrating Topic Modeling with Word Embeddings by Mixtures of vMFs. In *International Conference on Computational Linguistics*. 151–160.

[17] Ximing Li, Changchun Li, Jinjin Chi, and Jihong Ouyang. 2018. Short Text Topic Modeling by Exploring Original Documents. *Knowledge and Information Systems* 56, 2 (2018), 443–462.

[18] Ximing Li, Changchun Li, Jinjin Chi, Jihong Ouyang, and Chenliang Li. 2018. Dataless Text Classification: A Topic Modeling Approach with Document Manifold. In *ACM International Conference on Information and Knowledge Management*. 973–982.

[19] Ximing Li, Yue Wang, Ang Zhang, Changchun Li, Jinjin Chi, and Jihong Ouyang. 2018. Filtering out the Noise in Short Text Topic Modeling. *Information Sciences* 456 (2018), 83–96.

[20] Ximing Li and Bo Yang. 2018. A Pseudo Label based Dataless Naive Bayes Algorithm for Text Classification with Seed Words. In *International Conference on Computational Linguistics*. 1908–1917.

[21] Yuhua Li, David McLean, Zuhair A. Bandar, James D. O'Shea, and Keeley Crockett. 2006. Sentence Similarity Based on Semantic Nets and Corpus Statistics. *IEEE Transactions on Knowledge and Data Engineering* 16, 8 (2006), 1138–1150.

[22] Andreas Maurer. 2016. A Vector-contraction Inequality for Rademacher Complexities. In *International Conference on Algorithmic Learning Theory*. 3–17.

[23] Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. 2013. Linguistic Regularities in Continuous Space Word Representations. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2292–2300.

[24] Ofir Pele and Michael Werman. 2009. Fast and Robust Earth Mover's Distances. In *International Conference on Computer Vision*. 460–467.

[25] Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi. 2008. Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections. In *International Conference on World Wide Web*. 91–100.

[26] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. 1998. A Metric for Distributions with Applications to Image Databases. In *International Conference on Computer Vision*. 59–66.

[27] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. 2000. The Earth Mover's Distance as a Metric for Image Retrieval. *International Journal of Computer Vision* 40, 2 (2000), 99–121.

[28] Petra Schneider, Michael Biehl, and Barbara Hammer. 2009. Adaptive Relevance Matrices in Learning Vector Quantization. *Neural Computation* 21, 12 (2009), 3532–3561.

[29] Dou Shen, Rong Pan, Jian-Tao Sun, Jeffrey Junfeng Pan, Kangheng Wu, Jie Yin, and Qiang Yang. 2006. Query Enrichment for Web-query Classification. *ACM Transactions on Information Systems* 24, 3 (2006), 320–352.

[30] Richard Sinkhorn. 1967. Diagonal Equivalence to Matrices with Prescribed Row and Column Sums. *The American Mathematical Monthly* 74, 4 (1967), 402–405.

[31] Yang Song, Dengyong Zhou, and Li wei He. 2012. Query Suggestion by Constructing Term-Transition Graphs. In *ACM International Conference on Web Search and Data Mining*. 353–362.

[32] Aixin Sun. 2012. Short Text Classification Using Very Few Words. In *ACM SIGIR Conference on Research and Development in Information Retrieval*. 1145–1146.

[33] Songbo Tan and Xueqi Cheng. 2007. Using Hypothesis Margin to Boost Centroid Text Classifier. In *ACM Symposium on Applied Computing*. 398–403.

[34] Cédric Villani. 2008. *Optimal Transport: Old and New*. Springer Berlin Heidelberg.

[35] Fang Wang, Zhongyuan Wang, Zhoujun Li, and Ji-Rong Wen. 2014. Concept-based Short Text Classification and Ranking. In *ACM International Conference on Information and Knowledge Management*. 1069–1078.

[36] Peng Wang, Bo Xu, Jiaming Xu, Guanhua Tian, Cheng-Lin Liu, and Hongwei Hao. 2016. Semantic Expansion Using Word Embedding Clustering and Convolutional Neural Network for Improving Short Text Classification. *Neurocomputing* 174, Part B (2016), 806–814.

[37] Tao Wang, Yi Cai, Ho-fung Leung, Zhiwei Cai, and Huaqing Min. 2015. Entropy-Based Term Weighting Schemes for Text Categorization in VSM. In *IEEE International Conference on Tools with Artificial Intelligence*. 325–332.

[38] Yang Wang, Xuemin Lin, Lin Wu, Wenjie Zhang, Qing Zhang, and Xiaodi Huang. 2015. Robust Subspace Clustering for Multi-view Data by Exploiting Correlation Consensus. *IEEE Transactions on Image Processing* 24, 11 (2015), 3939–3949.

[39] Yang Wang, Lin Wu, Xuemin Lin, and Junbin Gao. 2018. Multiview Spectral Clustering via Structured Low-Rank Matrix Factorization. *IEEE Transactions on Neural Networks and Learning Systems* 29, 10 (2018), 4833–4843.

[40] Yang Wang, Wenjie Zhang, Lin Wu, Xuemin Lin, Meng Fang, and Shirui Pan. 2016. Iterative Views Agreement: An Iterative Low-rank based Structured Optimization Method to Multi-view Spectral Clustering. In *International Joint Conference on Artificial Intelligence*. 2153–2159.

[41] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. 2010. TwitterRank: Finding Topic-sensitive Influential Twitterers. In *ACM International Conference on Web Search and Data Mining*. 261–270.

[42] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In *Neural Information Processing Systems*. 649–657.

[43] Peng Zhao and Zhi-Hua Zhou. 2018. Label Distribution Learning by Optimal Transport. In *AAAI Conference on Artificial Intelligence*. 4506–4513.

[44] Yuan Zuo, Jichang Zhao, and Ke Xu. 2016. Word Network Topic Model: A Simple but General Solution for Short and Imbalanced Texts. *Knowledge and Information Systems* 48, 2 (2016), 379–398.

Large Scale Semantic Indexing with Deep Level-wise Extreme Multi-label Learning

Dingcheng Li, Jingyuan Zhang, Ping Li
Cognitive Computing Lab (CCL), Baidu Research
{lidingcheng,zhangjingyuan03,liping11}@baidu.com

ABSTRACT

Domain ontology is widely used to index literature for the convenience of literature retrieval. Due to the high cost of manual curation of key aspects from the scientific literature, automated methods are crucially required to assist the process of semantic indexing. However, it is a challenging task due to the huge amount of terms and complex hierarchical relations involved in a domain ontology. In this paper, in order to lessen the curse of dimensionality and enhance the training efficiency, we propose an approach named *Deep Level-wise Extreme Multi-label Learning and Classification (Deep Level-wise XMLC)*, to facilitate the semantic indexing of literatures. Specifically, Deep Level-wise XMLC is composed of two sequential modules. The first module, deep level-wise multi-label learning, decomposes the terms of a domain ontology into multiple levels and builds a special convolutional neural network for each level with category-dependent dynamic max pooling and macro F-measure based weights tuning. The second module, hierarchical pointer generation model merges the level-wise outputs into a final summarized semantic indexing. We demonstrate the effectiveness of Deep Level-wise XMLC by comparing it with several state-of-the-art methods on automatic labeling of MeSH, on literature from PubMed MEDLINE and automatic labeling of AmazonCat13K.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by classification**; **Transfer learning**; **Classification and regression trees**; **Neural networks**.

KEYWORDS

deep level-wise extreme multi-label learning and classification, online macro F-measure optimization, pointer generation

ACM Reference Format:
Dingcheng Li, Jingyuan Zhang, Ping Li. 2019. Large Scale Semantic Indexing with Deep Level-wise Extreme Multi-label Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313636

1 INTRODUCTION

With the explosively growing amount of scientific literature, semantic indexing methods with high efficiency are required to build retrieval systems. Even with effective techniques, the semantic indexing process still involves manual curation of key aspects from scientific literature. In order to summarize the main topics of articles, domain experts are usually invited to manually index articles with keywords that are selected from the domain ontology.

In the medical field, MEDLINE [3, 19, 27] is perhaps the world largest biomedical literature database, and Medical Subject Headings (MeSH) is the domain ontology for indexing articles in MEDLINE. It has greatly improved the experience of medical literature search by mapping queries to MeSH headings. For example, the query *teen drug use* is mapped to the MeSH headings *Adolescent* and *Substance–Related Disorders*. Currently, most of the mapping rules as well as the final indexing of medical literature from MEDLINE are manually generated by domain experts. It is expensive and time-consuming for the human-labeling process of semantic indexing. Automated methods are therefore crucially desired.

The task of automated curation, however, faces significant challenges. First of all, an article is often labeled with multiple keywords or concepts. In addition, the domain ontology involves hundreds of thousands or even millions of labels. Those labels are typically organized in hierarchical structures that are represented in the form of a forest. It is a non-trivial task to simultaneously deal with massive labels, data samples and complex hierarchical structures.

In this paper, we consider the task of automated semantic indexing as an extreme multi-label learning and classification (XMLC) problem. Different from the traditional multi-class [15, 16, 34], XMLC allows for the co-existence of millions of labels for each data sample. Recently, several approaches are proposed to deal with XMLC, including FASTXML [29], LOMTrees [9], SLEEC [4], robust Bloom filters [10], label partitioning [33], fast label embeddings [25] and several deep learning methods, Hierarchical multi-label classification using local neural networks [7], DXML [20] and XML-CNN [37]. To a good extent, those methods have achieved good progress in handling XMLC. However, the curse of dimensionality (for which we refer to the huge label space) and the high demand of hand-crafted feature engineering are two major barriers for further improving the effectiveness and efficiency.

In order to address these two issues, we propose a novel framework named *Deep Level-wise Extreme Multi-label Learning and Classification (Deep Level-wise XMLC)* to deal with the problem of large scale semantic indexing. Deep Level-wise XMLC consists of two sequential modules. The first module is a level-wise multi-label classification model. It addresses the curse of dimensionality effectively by decomposing massive labels (in a higher dimensional space) into multiple levels (in a lower dimensional space). For each level, a convolutional neural network is constructed with two novel designs: One design is a category-based dynamic max pooling strategy aiming at capturing both label co-occurrences and categorical

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313636

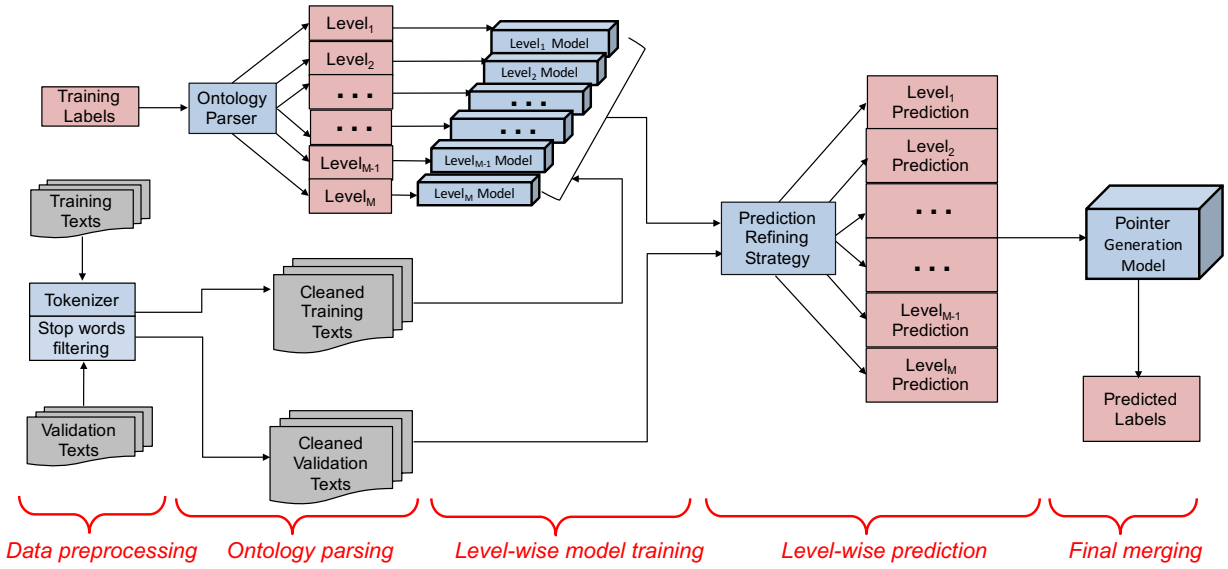


Figure 1: System architecture of the proposed framework. From left to right, there are five steps or blocks. In the data preprocessing stage, as usual, tokenization and filtering are made to obtain clean texts. Training and validation data are randomly selected according some proportions. Different from usual NLP tasks, an extra step of ontology parsing on the training labels is needed so that labels will be split into multiple levels based on their ontological hierarchies. In the third stage, the neural model described in the methodology section is employed to train level-wise models. Then, in the testing stage, testing data is fed into the trained level-wise models for label predictions or tagging after the testing data is preprocessed in a similar fashion. In the last stage, a final merging is made with a pointer generation model so that some less related labels are sifted out.

relations among labels. It helps connect the level-wise classification models tightly. The other design is a prediction refining strategy based on macro F-measure optimization, which enables the module to automatically select the labels in an incremental manner. The second module of Deep Level-wise XMLC is a hierarchical pointer generation model that merges predicted labels for each level into final summarized semantic indexing by the way of copying and generation mechanism [31]. As a whole, Deep Level-wise XMLC avoids the high cost of human interferences by learning semantic indexing without any feature engineering. We show the entire system architecture in Figure 1.

In summary, our major contributions are summarized as follows:

- We propose *Deep Level-wise XMLC* to learn large scale semantic indexing. It divides labels into multiple levels to lessen the curse of dimensionality while improving the training efficiency.
- We introduce a new strategy with category-dependent dynamic max pooling to capture both co-occurrences and categorical relations among labels.
- We explore a prediction refining technique derived from macro F-measure optimization to intelligently select the best labels in an online fashion.
- We develop a hierarchical pointer generation model to merge the level-wise outputs into the final summarized semantic indexing.
- We demonstrate the effectiveness of Deep Level-wise XMLC by comparing it with several state-of-the-art methods on automatic labeling of MeSH from MEDLINE, as well as AmazonCat13K [1], which is the XMLC dataset with similar nature as MeSH.

[1]https://manikvarma.github.io

2 METHODOLOGY

There are two primary challenges in XMLC. Firstly, the number of labels in one dataset can be more than 10,000 or even as large as one million. Secondly, one data sample can be indexed with multiple labels, the number typically ranging from one to several dozens.

In this paper, we propose Deep Level-wise XMLC to handle these two challenges by decomposing labels of each data sample into multiple levels. Five steps are involved in Deep Level-wise XMLC as shown in Figure 1. Firstly, the input raw texts for training and validations are processed with NLP preprocessors. Words are tokenized and stop words are filtered out. Secondly, the training labels are parsed into level-wise labels with ontology parsers. Thirdly, level-wise models are trained with Deep Level-wise XMLC. Fourthly, level-wise prediction is made on the datasets with refining strategies. Finally, a point generation model is trained to merge level-wise models into a unified label set.

The following subsections will focus on 1) our deep level-wise multi-label learning framework and 2) the pointer generation model to merge labels of all levels into one unified label set.

2.1 Deep level-wise multi-label learning

Formally, the problem can be defined as follows. Given a set of input pairs $\cup_{i=1}^{N}\{\mathbf{x}_i, \mathbf{y}_i\}$, Deep Level-wise XMLC decomposes them into M levels and trains M neural models on the training data. The whole label set is denoted as $\mathcal{L}$ and $|\mathcal{L}|$ refers to the total number of labels in $\mathcal{L}$. Each $\mathbf{y}_i$ is a multi-hot vector with length $|\mathcal{L}|$. Each model at level m predicts the most probable K labels, $\{\hat{y}_1^m, ..\hat{y}_j^m, ..., \hat{y}_K^m\}$ on each data sample. K is determined with a refining strategy. In the

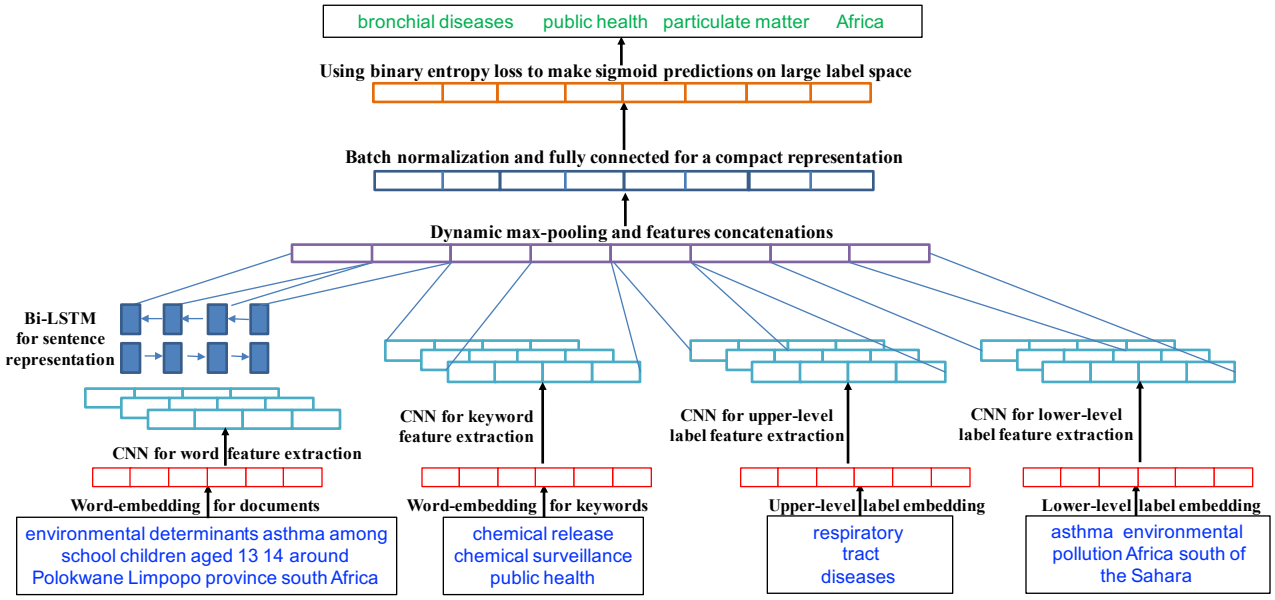


Figure 2: Neural structure of the proposed deep multi Label learning framework. As shown in this figure, the inputs involve four parts: word embedding for documents, word embedding for key words, upper-level and lower level embedding. Embeddings are pretrained with documents, keywords and labels for each document of the training data. Note that label embeddings are made with the same source, namely, the corresponding ontology. After the embedding representations are obtained, all of them are fed into CNN for feature extractions. In order to keep the order for documents, Bi-LSTM is employed for sentence representation. Then, dynamic max-pooling and batch normalization follow those feature extractions. Finally, binary entropy loss is computed to make sigmoid predictions on each label. We can also view this network structure as a multi-task learning.

end, a pointer generation model is trained to merge the predicted $\{\hat{\mathbf{y}}_i^1, ..., \hat{\mathbf{y}}_i^m, ..., \hat{\mathbf{y}}_i^M\}$ of M levels for each data sample $\mathbf{x}_i$ into one unified label set $\mathbf{y}_i$.

2.1.1 Feature embedding construction. We construct models in a level-wise manner. A neural model is built at each level with four parallel inputs as shown in Figure 2. The four inputs include documents, keywords and level-related information. They provide diverse information for the construction of more discriminative features. We employ CNNs to learn a rich number of feature representations. Document and keyword embedding are learned through CNN directly. The other two embeddings, upper-level and lower-level label embeddings, are learned from the embedding of prediction results from upper and lower levels. Specifically, two steps are involved. Firstly, similar to word embedding for input texts and keywords, we employ Gensim to train label embeddings from the annotated MeSH. Secondly, in both training and testing, predicted labels for some documents at some levels will be utilized as input features for their upper level or lower level. The two embeddings can not only help capture level-wise dependencies, but also deal with label imbalance issues in XMLC [4, 37]. In this way, both label co-occurrences and the knowledge from their upper and lower levels can help enhance the representation learning of rare labels.

For example, in MeSH, *Lymphangioma* is a rare label and it can not be easily represented by itself. With the information of its upper level MeSH, *Lymphatic Vessel Tumors* and lower level MeSH *Lymphangioma*, *Cystic Lymphangioma* can be better represented in the embedding space.

After the four embeddings are learned, they are concatenated and delivered into the max-pooling layer.

Due to the order information, raw tokens/words cannot be directly concatenated to the embeddings of keywords, upper and lower level labels. Hence, we construct a Bi-LSTM for raw tokens/words over their CNN features to keep the language order information [14] before concatenation.

2.1.2 The objective function of the learning framework. After the embedding concatenation, a dynamic max pooling layer, a binary cross-entropy loss over sigmoid output and a hidden bottleneck layer are employed. The loss function L of the binary cross-entropy objective is formulated as

$$L = \tag{1}$$

$$\underset{f}{\text{argmin}} -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{|\mathcal{L}|} y_{ij}\log(\sigma(f_j(\mathbf{x}_i)) + (1-y_{ij})\log(1-\sigma(f_j(\mathbf{x}_i))),$$

where $\sigma(f_j(\mathbf{x}_i)) = \frac{1}{1+e^{-f_j(\mathbf{x}_i)}}$ and $f_j(\mathbf{x}_i)$ denote the output layer function. In addition, $f_j(\mathbf{x}_i) = \mathbf{w}_o g_h(\mathbf{w}_h[P(\mathbf{c}_1), ..., P(\mathbf{c}_\iota))])$. Here $\mathbf{w}_h \in R^{h\times(\iota p)}$ and $\mathbf{w}_o \in R^{|\mathcal{L}|\times h}$ are weight matrices associated with the bottleneck layer and output layer, g_h is the element-wise activation functions, e.g., *sigmoid* or *tanh* applied to the bottleneck layer and ιp is the product of ι and p at the dynamic max pooling layer. ι refers to the number of features fed into pooling layers and

p refers to pooling numbers. Both are determined by the number of features in $\mathbf{x_i}$. In addition, $\mathbf{c}_i$ is the vector of convolutional features after the pooling operation $P(.)$ from lower layers.

2.1.3 Categorical-oriented dynamic max pooling. In traditional CNN models for text classification, a max-over-time [11] scheme is often adopted, as intuitively the maximum element of a feature map should take the most important information, i.e., $P(\mathbf{c}) = \max\{\mathbf{c}\}$. This approach, however, exhibits a severe drawback. Using only one value to represent the whole feature map may miss information when the input document includes multiple topics. For multiple-label learning tasks, multiple pooling can capture richer information [20]. In this work, we dynamically perform pooling as $P(\mathbf{c}) = [\max\{\mathbf{c}_{(1:\frac{l}{p})}\}, ..., \max\{\mathbf{c}_{(l-\frac{l}{p}+1:l)}\}] \in R^p$, where $\mathbf{c}_{(1:\frac{l}{p})}$ refers to the sub-vector of $\mathbf{c}$ starting from index 1 to $\frac{l}{p}$. Previous work used a fixed p. If p is set too large, redundant features may be included. If it is set too small, relevant features may be missing.

In our work, level-wise related information, i.e., categorical information of labels is incorporated into the neural structures to help select p dynamically. Specifically, we tune p with the distribution of the label levels. For example, in MeSH, all terms are divided into 16 categories, like *Anatomy*, *Organisms*, *Diseases* and etc. Each category involves diverse subcategories and each label involves different distributions. Based on the distribution, we assign different weights to determine the p. The larger the weight of the category is, the larger the p is. The weight of the category or the label is initialized from the training data.

2.1.4 Refining predictions with macro F-measure maximization. With embeddings and dynamic max pooling, the network can make level-wise predictions as shown in Figure 1. At each level, we select the top-K predicted labels for each data sample. However, a fixed K may yield high recall but low precision. In this work, we refine the predictions with a more flexible weight adjustment strategy.

We apply the online F-measure optimization (OFO) [6, 13] for the weight adjustment. With OFO, a dynamic balance of precision and recall can be achieved. Formally, the OFO algorithm optimizes the binary F-measure through threshold tuning in an online fashion.

$$F_{ij}^t = \frac{2\sum_{l=1}^{i} y_{lj}^t \hat{y}_{lj}^t}{\sum_{l=1}^{i} y_{lj}^t + \sum_{l=1}^{i} \hat{y}_{li}^t} = \frac{2\alpha_{ij}^t}{\beta_{ij}^t}, \tag{2}$$

with $\alpha_{ij}^t = \sum_{l=1}^{i} y_{lj}^t \hat{y}_{lj}^t$ and $\beta_{ij}^t = \sum_{l=1}^{i} y_{lj}^t + \sum_{l=1}^{i} \hat{y}_{li}^t$. Here y_{lj}^t is the j-th label of the l-th data sample. F_{ij}^t is the accumulated F-score from the first to the i-th data sample on label y_j at iteration t.

Due to the incremental property, the threshold of OFO is updated by two rules. At the same iteration, the threshold λ_{ij}^t is updated as $\lambda_{ij}^t = \alpha_{(i-1)j}^t / \beta_{(i-1)j}^t$. At different iterations, it is updated as $\lambda_{ij}^t = \alpha_{Nj}^{(t-1)} / \beta_{Nj}^{(t-1)}$. Given the i-th data sample, OFO refines predicted labels as $\hat{y}_{ij}^t = \left[\hat{\eta}(\mathbf{x}_{ij}^t) > \lambda_{ij}^t\right]$, where $\hat{\eta}(\mathbf{x}_{ij}^t)$ refers to the prediction probability of $\mathbf{x}_i$ on label y_j at iteration t. The optimal F-measure $\hat{F}(\lambda)$ is twice the value of the optimal threshold $\hat{\lambda}$ as $\hat{F}(\lambda) = 2\hat{\lambda}$. Since our refining mechanism is dynamic, level-wise and incremental, the optimal threshold $\hat{\lambda}$ will not be fixed until the end of training. It will be saved as a parameter for testing.

2.2 Pointer generation model for final merging

After we have level-wise outputs, we need to merge those outputs into one unified label set. However, we cannot simply combine them together because a simple concatenation will lead to a much larger number of labels than the gold standard labels. In this paper, we propose a filtering method to remove some level-wise labels to make sure that the final distributions of predicted labels are consistent with the gold standard ones. Inspired by the text summarization [23, 31], we treat each level-wise predication as one sentence and the gold standard as a summarized output. We take into consideration the hierarchical relations of labels among levels during decoding, encoding and attention states [23, 31].

2.2.1 Hierarchical pointer generation model. Inspired by [31], we design our hierarchical pointer generation model by allowing both copying labels from the level-wise predictions and generating labels from the whole label set.

The model is composed of five parts as in Figure 3. The first part, i.e., the input, is about the level-wise predictions from level 1 to level M. In the second part, the input is encoded to M sequences of hidden states. Each encoded hidden state reflects the inner relations of the predicted labels at a certain level. We represent the *encoder hidden state* as $\mathbf{e}^\tau = \mathbf{v}^T \tanh(\mathbf{w}_h \gamma^\tau + \mathbf{w}_s \mathbf{s}^\tau + b_{attn})$. Here $\mathbf{s}^\tau$ and γ^τ are the predicted *label sequence vector* and the *context vector* surrounding the predicted labels, respectively. The terms $\mathbf{v}$, $\mathbf{w}_h$, $\mathbf{w}_s$ and b_{attn} are weight parameters. The *context vector* is about the co-occurrences of labels.

In the third part, *attention generators* are derived from the *encoder hidden state* to generate an attention distribution $\mathbf{a}^\tau$ and a context vector γ^τ at time step τ. $\mathbf{a}^\tau$ is calculated as $\mathbf{a}^\tau = softmax(\mathbf{e}^\tau)$. The attention distribution is a probability distribution over the predicted level-wise labels. It is used to produce γ^τ as a hierarchical weighted sum of the *encoder hidden states*: $\gamma^\tau = \sum_q \mathbf{w}_q \mathbf{a}_q^\tau \gamma_q$.

In Figure 3, each attention generator is named as a coverage vector, which shows how much focus is given to labels of each level. As is known, summarizations may lead to repetitions. Hence, the same label may be generated more than once as well. The well-designed coverage vector plays the role of judging whether the label is a duplicate or not. If not a duplicate, the label with a high attention has more chance of being decoded as one right label. If a duplicate, the mechanism at 2.2.3 will filter the label out.

In the fourth part, i.e., the *decoder hidden states*, the generation probability $p_{gen} \in [0, 1]$ for time step τ is calculated from the context vector γ_τ, the encoder state $\mathbf{s}_\tau$ and the decoder input $\mathbf{y}^\tau$ as:

$$p_{gen} = \sigma(\mathbf{w}_h \gamma^\tau + \mathbf{w}_s \mathbf{s}^\tau + \mathbf{w}_y \mathbf{y}^\tau + b_{ptr}), \tag{3}$$

where $\mathbf{w}_h$, $\mathbf{w}_s$, $\mathbf{w}_y$ and b_{ptr} are weight parameters. Here p_{gen} is used as a soft switch to choose between *generating* a label from the whole label set by *sampling* from the label distribution, $p_{\mathcal{L}}$ (see how $p_{\mathcal{L}}$ is calculated in 2.2.2) or *copying* a label from the input sequences by sampling from the attention distribution $\mathbf{a}^\tau$.

With the above four parts, the proposed hierarchical pointer generation model can be trained to generate the final summarized semantic indexing. In the fifth part, we learn the probability of generating the final labels. Given a training pair $(\cup_{m=1}^{M} \mathbf{y}^m, \mathbf{y})$, we compute the conditional probability $p(\cup_{m=1}^{M} \mathbf{y} | y, \theta)$ to estimate the

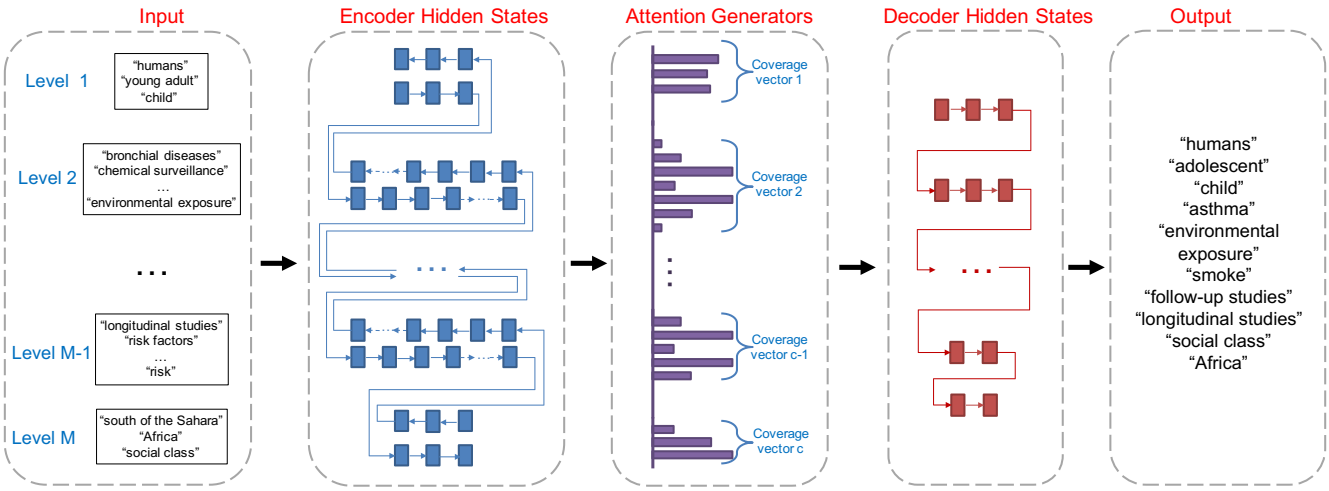


Figure 3: Pointer generation model for final merging. This step follows the level-wise neural structure learning. Five components can be seen from left to right. The input is the predicted level-wise labels. They are organized as sequences level by level. Then, they are encoded into hidden states. Based on coverage vectors described in the methodology, attentions are generated. Then, decoder is working to generate outputs with reduced sizes.

labels of the terms of the probability chain rule as

$$p(\mathbf{y}|\cup_{m=1}^{M}\mathbf{y}^m;\theta)=\prod_{\delta=1}^{|\mathcal{L}|}p_\theta(y_\delta|y_1,...,y_{\delta-1};\theta), \tag{4}$$

where $\mathbf{y}=\{\mathbf{y}_1,...,\mathbf{y}_{|\mathcal{L}|}\}$ is a sequence of $|\mathcal{L}|$ vectors. The parameters of the model are learned by maximizing the conditional probabilities for the training set as $\theta=\underset{\theta}{\text{argmax}}\sum_{\mathbf{y},\cup_{m=1}^{M}}\log p(\mathbf{y}|\cup_{m=1}^{M}\mathbf{y}^m;\theta)$, where the sum is over training examples.

2.2.2 Sequence-to-sequence probability calculation. The above procedure finally will produce the label vocabulary distribution $p_{\mathcal{L}}$:

$$p_{\mathcal{L}}=softmax(\mathbf{v}'(\mathbf{v}[s_t,h_{*t}]+b)+b'), \tag{5}$$

where $\mathbf{v}$, $\mathbf{v}'$, b and b' are learnable parameters. For a specific label, we can obtain from $p(y_{ij})=p_{\mathcal{L}}(y_{ij})$. The loss function is the negative log likelihood of the target label $\hat{y}_{ij}$. The following example illustrates the procedure of probability calculations for one label given other labels.

Suppose we have a label set $\{Nutritional\ and\ Metabolic\ Diseases\}$. The hierarchical relations are shown in Figure 4. The left words in the figure are acronyms for the labels. For example, $Wolfram\ Syndrome$ is shortened with their initial letters as ws.

Given $context=\{nmd,md,dm,dmt1\}$, we follow the hierarchical relations among those labels to calculate $p(\mathbf{e}_{ws}|context)$ as

$$p(\mathbf{e}_{ws}|context)=(1-\sigma(b_{ptr}+\mathbf{e}_{nmd}^T\mathbf{h}_{nmd})\sigma(b_{ptr}+\mathbf{e}_{md}^T\mathbf{h}_{md})$$
$$\sigma(b_{ptr}+\mathbf{e}_{dm}^T\mathbf{h}_{dm})\sigma(b_{ptr}+\mathbf{e}_{dmt1}^T\mathbf{h}_{dm1})). \tag{6}$$

2.2.3 Mechanism to avoid duplications. A problem for pointer generation model or sequence-to-sequence model is that it may copy terms from input multiple times [24, 30]. In our task, we do not need any repetitive terms since each label should be unique. Previous works [31, 32] adapts a coverage mechanism to avoid repetition. Namely, if labels have been seen in the output, the probability of generating them will become low. We take similar approaches by combining a coverage mechanism into the whole pointer generation model. Specifically,

$$\varkappa^m=\sum_{\mathbf{y}^m=0}^{|\varkappa|}\varkappa^{\mathbf{y}^m}, \tag{7}$$

where, $\varkappa$ refers to a coverage vector, $\mathbf{y}^m$ refers to m-th level.

Our coverage vectors are composed of a set of vectors for all levels. For each coverage vector, $\varkappa^{\mathbf{y}^m}$ is an unnormalized distribution over the level-wise inputs that represents the degree of coverage that those labels have received from the attention mechanism so far. Since we order labels with levels, there should be no repetitions at different sections of levels, this mechanism aims at removing all duplicate labels found in different sections and also avoid duplicates within the same level. $w_\varkappa\varkappa^{\mathbf{y}^m}$ is added to the attention mechanism and a $covloss_{\mathbf{y}^m}$ is also added to the total loss function of the pointer generation model as the penalty for any duplications.

3 EXPERIMENTAL RESULTS

In this section, we evaluate the effectiveness of our Deep Level-wise XMLC with MEDLINE dataset labeled with MeSH and AmazonCat13K. As described in Section 1, MEDLINE is the largest biomedical literature database in the world and Medical Subject Headings (MeSH) is the domain ontology for tagging articles in MEDLINE. The other dataset we use, AmazonCat13K, is one of benchmark datasets for developing extreme classification algorithms. It involves 13330 labels, with all labels hierarchically organized, similar to MeSH. The dataset scale, the expert labeling and the hierarchical nature provide a perfect testbed for our proposed framework.

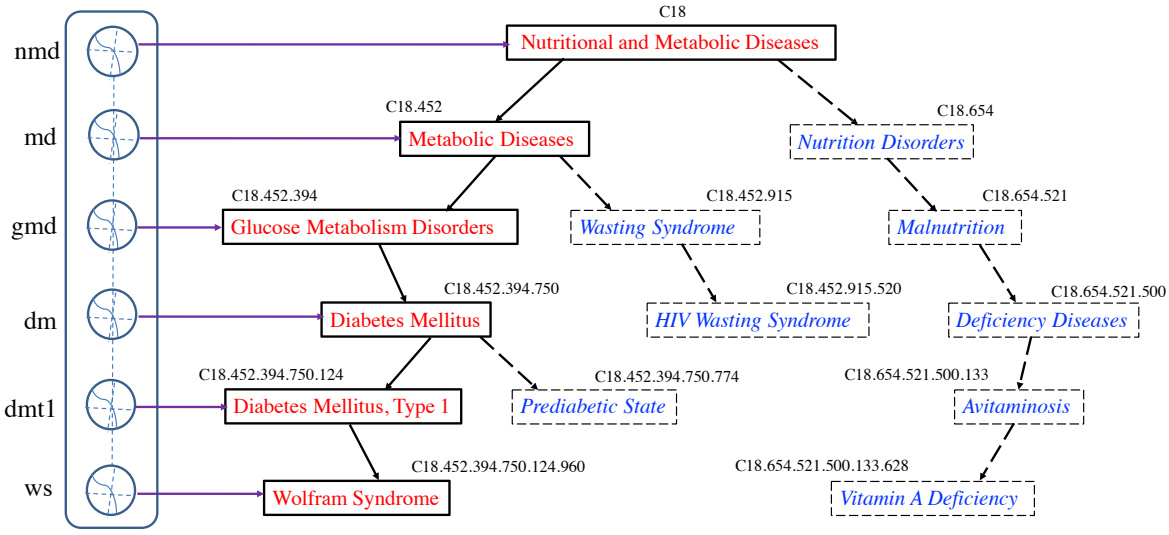


Figure 4: Example of sequence-to-sequence probability calculation. The calculation of the example is described in section 2.2.2. A little bit clarification may be necessary is on those short forms. On the left, for space saving and for better illustration of the process, Sigmoid symbols are drawn there with those short forms, which are the initial letters of the mesh terms.

3.1 Data setting and preprocessing

The total number of MeSH labels in MEDLINE is 26,000, among which 60% appear more than 1000 times. We remove those MeSH labels occurring less than 10 times in the experiment. The MEDLINE has 26 million articles with abstracts. 90% of these articles have about 20 MeSH labels. 4 to 16 MeSH labels are assigned to 82% of articles. In MeSH, 3.5 million abstracts have both MeSH labels and keywords. The ontology of MeSH labels can be decomposed into 7 levels, where the lowest level (the 7th level) includes the most specific MeSH labels while the highest level (the 1st level) has the most general and abstract MeSH labels. For articles with only MeSH labels of the lowest level, we expand them by the following method. Starting from labels at the lowest level, we find out all labels of their upper levels. We construct 7 datasets for the proposed Deep Level-wise XMLC framework.

Meanwhile, 102,167 abstracts with MeSH labels from all the 7 levels are put aside for testing. The statistics of the dataset at each level is shown in Table 1. It can be observed that the middle levels have the largest number of labels while the highest level has only 83 labels and the lowest level has 2445 labels. Similar trend can been found for data volumes. Two million articles have labels from level 2, 3 and 4 while less than one million articles have labels from level 1, 6 and 7.

For AmazonCat13K, we cannot directly use their preprocessed dataset since Deep Level-wise XMLC requires text data. Meanwhile, we need divide the data based on their level-wise categories. It is found that all labels can be decomposed into 9 levels. Somewhat differently, if a document from AmazonCat13K has lower labels, it must have higher labels while a document from MeSH is not necessarily so. Therefore, it is straightforward to find a common set for testing for AmazonCat13K (simply use documents with lower categories). In order to keep a reasonable pool of testing data, we ignore documents which have levels higher than 6 (only 9990, 385 and 31 documents for level 7,8, and 9, respectively).

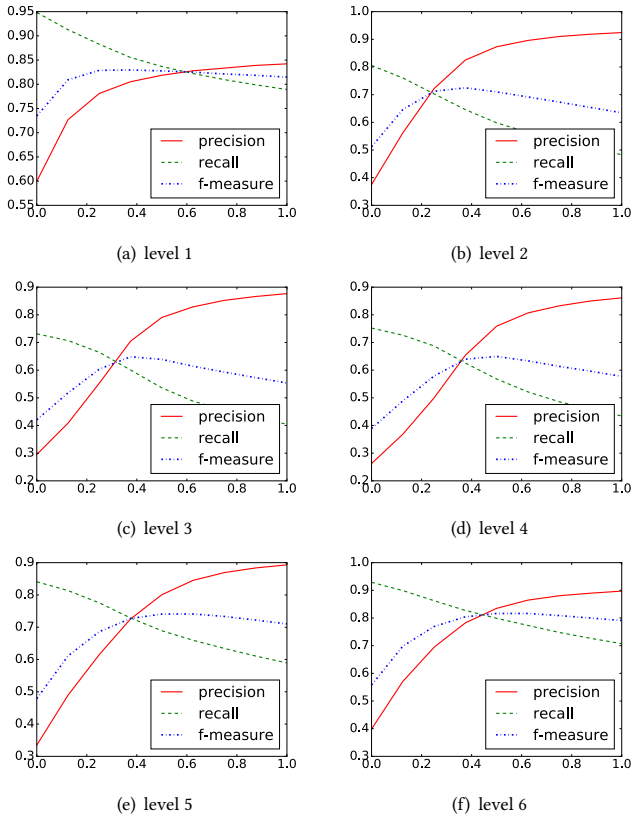
Table 1: The statistics of the datasets. For each level, there are different data volumes. Papers in Medline, do not necessarily imply that they can be tagged higher level MeSH terms even if they are tagged with lower level MeSH terms.

Levels	Data Volumes		The number of labels	
	MeSH	AmazonCat13K	MeSH	AmazonCat13K
Level 1	969,233	858,795	83	44
Level 2	2,444,854	812,249	1,382	362
Level 3	2,405,321	549,326	4,484	2,281
Level 4	2,182,885	427,378	6,568	6,181
Level 5	1,522,195	178,042	5,750	5,372
Level 6	906,873	71,041	3,895	1,998
Level 7	402,794	9,990	2,445	441

In the experiments, for the MEDLINE articles and keywords, at each level, we first train an individual neural network according to the first component of Deep Level-wise XMLC. The trained model is employed to make predictions on the testing data for each level. Then the predicted level-wise labels as well as the gold standard labels from the training data are utilized by the pointer generation model for the final merging. Likewise, we train level-wise model for AmazonCat14K except that the latter do not have keywords.

3.2 Evaluation metrics

In extreme multi-label classification datasets, even though there are usually huge label spaces, only limited number of relevant labels for each document. This means that it is important to present a short ranked list of relevant labels for each test document. The



(a) level 1 (b) level 2

(c) level 3 (d) level 4

(e) level 5 (f) level 6

Figure 5: Macro precision, recall and F-measure obtained by OFO with Deep Level-wise XMLC for the first six levels.

evaluation thus focuses on the quality of such ranked lists with emphasis on the relevance of the top portion of each list. In our work, however, we use two evaluation metrics for the purpose of comparisons with the two sources of datasets. The medical community prefers to use precision, recall and F-score while those from the general domains prefer precision at K (P@K) and the Normalized Discounted Cummulated Gains (NDCG@K or G@K for short).

Specifically, given a predicted label list $\mathbf{y}_1^m = \{y_1, y_2, ..., y_K\}$ with top K items at level m, precision, recall and F-score are defined as follows,

$$MiP = \frac{\sum_{i=1}^{N} c(K, i, y_1^K)}{\sum_{i=1}^{N} K_i}, \qquad MaP = \frac{1}{N}\sum_{i=1}^{N} \frac{c(K, i, y_1^K)}{K_i}, \tag{8}$$

$$MiR = \frac{\sum_{i=1}^{N} c(K, i, y_1^K)}{\sum_{i=1}^{N} AK_i}, \qquad MaR = \frac{1}{N}\sum_{i=1}^{N} \frac{c(K, i, y_1^K)}{AK_i}, \tag{9}$$

$$MiF = \frac{2 * MiP * MiR}{MiP + MiR}, \qquad MaF = \frac{2 * MaP * MaR}{MaP + MaR}, \tag{10}$$

where N is the number of data samples and $c(K, i, y_1^K)$ is the number of correct labels among the top K ranked labels; AK_i is the total number of the gold standard labels for article i; The difference between micro measures and macro measures lies in the calculation of the predicted probabilities. For micro measures, the probability calculation is not done until all correct predictions are added together, while for macro measures, the probability calculation will be done for each article and in the end, an average is used as the macro scores. We report both measures in order to see how accurate the model is for a single article and for an entire dataset.

In contrast, the definition of P@K and NDCG@K is,

$$P@k = \frac{1}{k}\sum_{l \in r_k(\hat{\mathbf{y}})} y_l \tag{11}$$

$$DCG@k = \sum_{l \in r_k(\hat{\mathbf{y}})} \frac{y_l}{log(l+1)} \tag{12}$$

$$NDCG@k = \frac{DCG@k}{\sum_{l=1}^{min(k, \|\vec{y}\|)} \frac{y_l}{log(l+1)}} \tag{13}$$

where $\vec{y} \in \{0, 1\}^L$ is denoted as the vector of true labels of a document and $\hat{\vec{y}} \in R^L$ as the system-predicted score vector for the same document. We use $k = 1, 3, 5$, following the convention of P@K and NDCG@K.

3.3 Parameter settings

For the neural network of Deep Level-wise XMLC, we use the rectified linear units. The filtering windows are set to 3, 4, 5. The dropout rate is set to 0.5 and the L2 constraint is set to 3. The mini-batch size is set to 256. The embedding dimensions vary for different features. For Mesh, word embedding for medical words involves 500,000 unique tokens, keyword embedding involves over 100,000 phrases and label embedding 26,000 MeSH terms. Gensim is employed to train the embedding with 300 as the dimension. For AmazonCat13K, pretrained GoogleNews-vectors-negative300.bin is utilized with 3 million tokens and 300 as the dimension. The values for other hyperparameters are chosen via a grid search on a smaller validation set from the training data.

3.4 Performance with online F-measure optimization

As discussed in Section 2.1.4, the online macro F-measure optimization (OFO) is integrated into the proposed framework. In order to show the effectiveness of OFO, we report the macro precision, recall and F-score for the first 6 levels in Figure 5 for MeSH. Due to space limit, we do not show the result of the 7th level and that of AmazonCat13K in this paper. Similar performances can be obtained for them. We can observe that OFO helps achieve a balance between macro precision and recall. It is further observed that the optimal F-score is different at different levels. If we always select the top K ($k = 10$ in our experiment) for the level-wise prediction, we cannot obtain the best F-score though the recall at each level can be as high as around 80%. The precision can be as low as or less than 20%. The reason is that after we divide MeSH labels of each article into 7 levels, most of articles have only 2 to 5 labels at each level. This means that even if all of labels are within the top 10, the precision is only from 20% to 50% although the recall can be 100%. In this case, the F-scores are not high either. The OFO greatly removes less relevant labels so that the number of labels in the final prediction set of each level ranges from 2 to 5 as well. Meanwhile, most of

Table 2: The level-wise performance for micro measures with Top K and OFO. This table aims at showing the incremental improvements on the micro measures when the new features added stepwise for each level. Meanwhile, for each level, results for TopK without optimization and with optimization as OFO are shown there as well.

Embeddings	Levels	Level 1		Level 2		Level 3		Level 4		Level 5		Level 6		Level 7	
MEDLINE Collections	Measures	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO
	MiP	23.32	55.83	29.32	90.79	22.96	79.82	20.63	76.80	20.15	46.45	22.18	55.40	20.29	66.63
	MiR	95.68	84.77	65.17	37.57	38.32	35.10	37.21	37.65	49.85	62.31	72.24	73.42	83.64	82.34
	MiF	37.50	67.32	40.44	53.15	28.72	48.75	26.54	50.53	28.70	53.22	33.93	63.15	32.66	73.66
MEDLINE Collections & keywords		Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO
	MiP	31.45	72.45	36.53	88.10	31.56	52.90	29.36	76.80	31.25	84.81	32.88	69.43	33.43	77.19
	MiR	92.56	79.56	66.72	43.03	58.53	51.09	46.31	37.65	52.75	45.11	74.43	69.18	86.48	80.63
	MiF	46.95	75.84	47.21	57.82	41.00	51.98	35.94	50.53	39.24	58.90	45.61	69.30	48.21	78.87
MEDLINE Collections & keywords & upper & lower labels		Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO
	MiP	39.52	81.01	43.64	69.95	40.65	68.63	38.45	63.62	40.33	73.31	41.94	76.64	45.38	84.94
	MiR	86.75	73.93	63.27	56.05	55.33	43.70	52.31	46.76	51.14	53.07	69.45	65.88	77.36	77.30
	MiF	54.30	**77.33**	51.65	**62.23**	46.87	**53.40**	44.32	**53.90**	45.09	**61.57**	52.30	**70.85**	57.20	**80.94**
AmazonCat13K		Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO
	MiP	72.21	94.17	48.57	61.49	30.43	32.51	20.99	21.34	20.37	20.88				
	MiR	99.56	92.24	97.42	99.55	87.47	99.86	72.03	99.92	69.24	99.91				
	MiF	83.71	**93.19**	64.82	**76.02**	45.15	**49.05**	32.51	**35.17**	31.48	**34.54**				

Table 3: The level-wise performance for macro measures with Top K and OFO. This table aims at showing the incremental improvements on the micro measures when the new features added stepwise for each level. Meanwhile, in a smilar fashion to Table 2, for each level, results for TopK without optimization and with optimization as OFO are shown there as well.

Embeddings	Levels	Level 1		Level 2		Level 3		Level 4		Level 5		Level 6		Level 7	
MEDLINE Collections	Measures	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO
	MaP	23.32	59.96	29.32	56.18	22.96	81.40	20.63	36.84	20.15	48.95	22.18	69.94	20.29	78.73
	MaR	96.26	94.83	65.17	76.03	47.00	43.99	44.18	72.64	56.18	81.53	75.90	48.65	85.73	65.01
	MaF	37.55	73.47	40.44	64.61	30.85	57.11	28.13	48.89	29.66	61.12	34.32	57.38	32.81	71.22
MEDLINE Collections & keywords		Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO
	MaP	33.54	92.56	37.69	85.33	33.84	82.88	32.63	80.68	34.52	61.53	33.92	69.44	36.34	76.97
	MaR	91.35	70.23	68.27	55.80	61.33	48.79	48.94	52.16	55.77	77.55	72.94	86.25	82.88	92.67
	MaF	49.07	79.87	48.57	67.48	43.61	61.42	39.15	63.36	42.64	68.62	42.64	79.63	46.30	84.09
MEDLINE Collections & keywords & upper & lower labels		Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO	Top K	OFO
	MaP	42.33	80.53	45.46	82.52	41.46	70.48	38.99	75.90	43.49	84.52	43.49	86.46	49.34	92.09
	MaR	87.77	85.53	62.37	64.59	58.34	60.06	52.35	56.80	52.35	65.96	70.44	77.32	78.63	83.58
	MaF	57.11	**82.96**	52.59	**72.46**	48.47	**64.85**	46.05	**64.97**	44.69	**74.10**	53.78	**81.64**	60.63	**87.63**

the correct predictions are still kept in the prediction set. Evidently, this tuning strategy greatly boosts the performance.

3.5 Level-wise Performance

As discussed in Section 2, our framework decomposes the task of XMLC into level-wise model constructions. Therefore, in this section, we report level-wise prediction results in order to see the intermediate developments and improvements of the whole model.

As is shown in Figure 2, the level-wise neural models learn label embeddings from the MEDLINE collections, keywords and the predicted labels from the upper or lower levels. Here we report the performance of the level-wise neural models with different embeddings. We further show the effectiveness of OFO by comparing with the level-wise neural models by fixing the top K labels for each level. We test on different K from 1 to 10 and find that the best performance is achieved when K is 5.

Table 2 reports the micro performance for the level-wise model with OFO and top K fixing strategies. Here K is set to 5 for the best results. We also show the performance for the macro measurement in Table 3. We can see that OFO always perform better than the strategy of fixing top K, no matter on the micro or the macro measurements.

Table 2 and Table 3 also report the level-wise prediction with three different embeddings for MeSH. Although the evaluation for AmazonCat13K dataset is not based on F-scores, we also report micro measures for AmazonCat13K to show the advantages of OFO. After all, the result of P@K and NDCG@K for it is computed on the filtered output with OFO. From this result, we can see an evident incremental trend for all seven levels. Namely, with keywords and predicted MeSH terms of upper and lower levels added, the prediction sees quick improvements accordingly. It is not hard to see that in general, macro results are better than micro results. Among

them, the third level and the fourth level of MeSH while the fourth and the fifth level of AmazonCat13K yield worse results than others while level 1 obtain much better results for both datasets. This is understandable considering the larger number of labels for third and the fourth (4,484 and 6,568 for MeSH while 6,181 and 5,372 for AmazonCat13K respectively).

3.6 Performance of final merging

The proposed Deep Level-wise XMLC will merge the level-wise predictions into one unified label set with a pointer generation model. In this section, we further compare with five state-of-the-art approaches to demonstrate the effectiveness of the pointer generation model, including MTIDEF [12], MeSH Now [22], MeSHLabeler, MeSHRanker [21] and Deep Mesh [28] for MeSH results. All these existing systems make heavy use of feature engineering. In contrast, Deep Level-wise XMLC uses limited external resources. For AmazonCat13K, we report results of XML-CNN, the state of the art systems on this benchmark dataset.

Let us start from MeSH labeling. After we achieve the level-wise results, the hierarchical pointer generation model is trained with predicted results from all levels as the input and the gold standard labels as the output. For model training, the inputs can be organized with each label as an independent unit or with labels of the same level as one unit (known as *sentence* in the summarization community). Hence, two pointer generation models are trained, with the former known as Deep Level-wise XMLC$_{label}$ and with the latter as Deep Level-wise XMLC$_{level}$. For comparison, we also report results which adds results of all levels together and then filter less relevant labels by their prediction probabilities and by the label distributions in the gold standard (Deep Level-wise XMLC$_{sampling}$).

Table 4: Performance of Deep Level-wise XMLC for MeSH dataset. From the bold numbers, we can see that the best performances come from Deep Level-wise XMLC. It is obvious to see that level based and dynamic pooling obtain better performance than label based and dynamic pooling.

Methods	MaP	MaR	MaF
No Level-wise XMLC			
MTIDEF	49.39	51.40	50.37
BC_D2V-TFIDF	47.41	46.33	46.86
MeSHRanker	53.64	54.13	53.89
MeSHLabeler	54.50	51.72	50.54
DeepMesh	53.80	55.05	54.42
MeSH Now	51.28	53.72	52.47
Deep Level-wise XMLC without pointer generation network			
Sampling	49.56	51.21	50.37
Max pooling	51.56	65.52	57.70
Deep Level-wise XMLC with pointer generation network			
Dynamic pooling & label-based	**61.20**	57.21	59.14
Dynamic pooling & level-based	53.22	**70.65**	**60.77**

As shown in Table 4, both Deep Level-wise XMLC$_{label}$ and Deep Level-wise XMLC$_{level}$ outperform other systems much on macro measures in precisions, recalls as well as F-scores. Due to the page limit, we do not report the micro measures, which have similar trends.

By involving embeddings from MEDLINE collections and keywords, Deep Level-wise XMLC$_{label}$ and Deep Level-wise XMLC$_{level}$ achieve much better performances than all other existing cutting edge frameworks. To our interest, it seems that different organizations of the inputs will lead to different performances in precision and recall although F scores are quite similar. Deep Level-wise XMLC$_{label}$ achieves better precision while Deep Level-wise XMLC$_{level}$ better recall. This seems to indicate that our hierarchical pointer generation model takes into considerations the correlations between labels within the unit. Therefore, Deep Level-wise XMLC$_{level}$, which has longer input unit, obtains better recall. Yet, it also includes more false positives, thus reducing its precision. In contrast, Deep Level-wise XMLC$_{label}$ wins in precision probably it considers more smaller units and then misses more true positives.

Meanwhile, we can see that Deep Level-wise XMLC$_{sampling}$ obtains much poorer results than most of existing systems. This shows that the hierarchical pointer generation model plays essential roles in reaching the optimal performances in the end. Besides, we also report results of Deep Level-wise XMLC$_{level}$ with max pooling. By default, all of our systems work with dynamic max pooling. Evidently, the result shows that dynamic max pooling gains advantages over the usual max pooling strategies.

Table 5: Performance of Deep Level-wise XMLC for AmazonCat13K. We use the version with pointer generation network and dynamic pooling & level-based. As stated in 3.1, in order to extend our methodology from medical field to more general ones, we test our model on AmazonCat13K as well. For those who use AmazonCat13K, they prefer reporting precision@K and NDCG@K. We also list the performance of XML-CNN for AmazonCat13K for comparisons.

	P@1	P@3	P@5	G@1	G@3	G@5
XML-CNN	95.06	79.86	63.91	95.06	89.48	87.06
Deep Level-wise XMLC	96.52	83.72	67.89	97.48	92.32	87.52

For AmazonCat13K, the result is given in Table 5. We also copy the state of the art results from XML-CNN from their work [20] in Table 5. We can see higher results from our work. Due to the differences of the testing data, we cannot tell which work is better from those results (our testing dataset are extracted from the raw text data with labels of each level for better evaluations). Their work is tested on the standard test dataset prepared by data collectors. Yet, it at least shows that our model can achieve results as comparable to the best model.

4 RELATED WORK

4.1 Tree-based methods

Due to the huge number of labels, the prediction of XMLC may involve high cost in both time and space. Tree-based methods make efforts to reduce both training and testing cost. For example, the label partitioning by sub-linear ranking (LPSR) method attempts to reduce the prediction time by learning a hierarchy over a base

classifier [17, 18, 33]. Argrawal et al., 2013 [1] proposes a method as the multi-label random forest (MLRF), which seeks to learn an ensemble of randomized trees instead of relying on the learning of a base classifier. FastXML [29] is proposed to learn a hierarchy not over the label space but over the feature space. It defines the set of labels active in a region to be the union of the labels of all training points present in that region. At each node of the hierarchy, an NDCG-based objective is optimized. Namely, at each node, a hyperplane is induced and it splits the set of documents in the current node into two subsets. Predictions are made by returning the ranked list of the most frequently occurring labels in all the leaf nodes. Recently, Mu et al [26] develops multi-label classifications for social streams based on ensemble random forests. It integrates a base learner and a label-based learner to learn hierarchical labels. However, these approaches suffer from high cost of training due to the dimensionality of both label space and feature space.

4.2 Embedding methods

Embedding methods attempt to overcome the intractability issue brought by the huge number of labels by projecting label vectors onto a low dimensional space and thus reducing the number of labels. The assumption is that the label matrix is low-rank. Due to its strong theoretical foundations and the ability to handle label correlations, embedding methods have proved to be the most popular approach for tackling XMLC problems [2, 5, 8, 10]. In particular, the recently proposed embedding method SLEEC [4] greatly increases the accuracy after they incorporates the non-linear neighborhood constraints in the low-dimensional embedding space for training and use a simple k-nearest neighbor (k-NN) clustering in the embedding space for testing. In our approach, we take steps further by exploring level-wise label embedding to improve the predictions of our neural structure.

4.3 Max-margin method

Max-margin method is also employed to handle multi-label classification. Yen et al [36] proposes a model named as PD-Sparse. Essentially, a linear classifier is learned for each label with $L1$ and $L2$ norm penalty on the weight matrix associated with this label. This results in a sparse solution in both the primal and dual spaces. Their fully-Corrective Block-Coordinate Frank-Wolfe training algorithm achieves sub-linear training time w.r.t the number of primal and dual variables while getting better performance than 1-vs-all SVM and logistic regression on multi-label classification, with significantly reduced training time and model size. However, same as 1-vs-all SVM, the method is algorithmically not scalable to extreme multi-label learning.

4.4 Deep learning based method

Deep learning based method has also been used for multi-label learning [20, 37]. Zhang et al. [37] incorporate label space embedding into feature embedding. Specifically, they construct an adjacency matrix for labels A and derive the label graph matrix with the equation $M = (A + A^2)/2$. Then, for each nonzero entry in the matrix, a tuple composed of the index p, q and M_{pq} is fed to a label embedding network to train a compound network together with the word-embedding. In the prediction stage, k-NN search is performed in the low-dimensional feature representation to find similar samples from training datasets. The average of the k-NN's labels is set as final label prediction. [20] proposes to take multi-label co-occurrence patterns into the nueral network objective to improve the classification performance. They also propose to employ dynamic max pooling to capture rich information from different regions of the document and an additional hidden bottleneck layer to reduce model size. Moreover, a binary cross-entropy loss over sigmoid output is tailored to XMLC. However, these methods are not applicable for data with complex hierarchical labels like ours since the decomposition of label hierarchies reduce the label space greatly. In addition, Yan et al [35] proposes Boltzmann CNNs-based hybrid learning network to handle biomedical literature classification. Their work is enriched with data sequence embeddings. This design is not good for huge label space. Their experiments only focus on classes fewer than 2,000 MeSH labels. The work mostly close to ours is hierarchical multi-label classification network (HMCN) proposed by [7]. Their HMCN is claimed to be capable of simultaneously optimizing local and global loss functions for discovering local hierarchical class-relationships and global information from the entire class hierarchy while penalizing hierarchical violations. But their work has higher computational complexity due to the utilization of fully feed-forward layers. Even if they simplify their network with an LSTM-like model with shared weights, it still has high computation burden. It seems that is why they only report works on datasets of at most about 4000 labels.

5 CONCLUSION

In this work, we propose deep learning based level-wise framework to handle extreme multi-label learning and classification, named as Deep Level-wise XMLC. The innovation of Deep Level-wise XMLC includes a few points. Firstly, we develop a split model training mechanism, with which, labels are divided into multiple levels so that the curse of dimensionality and training cost are both lessened to a large degree. Secondly, category-dependent dynamic max pooling and weights adjustments with macro F-measure are integrated into the neural architecture so that the final predictions fit more to the distributions of the levels and their hierarchical relations. Thirdly, we successfully design a hierarchical pointer generation model to merge level-wise outputs into one unified label prediction.

The results show that Deep Level-wise XMLC achieves the state-of-the-art results by utilizing MEDLINE collections, keywords and predicted labels from upper and lower levels are utilized. The results for AmazonCat13K also show that Deep Level-wise XMLC is generic enough to handle diverse datasets.

In this work, it is not hard to see that Deep Level-wise XMLC can be conveniently transferred to tasks, like large scale semantic indexing for constructing more efficient and accurate information retrieval engines and reducing expensive manual expert efforts as shown in this work.

In future work, more experiments will be done on diverse datasets. Besides, model improvements can be made on designing more robust loss functions as well as adding more layers for handling feature refinements or weight adjustments and meanwhile improving running efficiency.

REFERENCES

[1] Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web (WWW)*. Rio de Janeiro, Brazil, 13–24.

[2] Krishnakumar Balasubramanian and Guy Lebanon. 2012. The Landmark Selection Method for Multiple Output Prediction. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*. Edinburgh, Scotland, UK, 283–290.

[3] William A Baumgartner Jr, K Bretonnel Cohen, Lynne M Fox, George Acquaah-Mensah, and Lawrence Hunter. 2007. Manual curation is not sufficient for annotation of genomic databases. *Bioinformatics* 23, 13 (2007), i41–i48.

[4] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. 2015. Sparse local embeddings for extreme multi-label classification. In *Advances in neural information processing systems (NIPS)*. Montreal, Canada, 730–738.

[5] Wei Bi and James Kwok. 2013. Efficient multi-label classification with many labels. In *International Conference on Machine Learning (ICML)*. Atlanta, GA, 405–413.

[6] Róbert Busa-Fekete, Balázs Szörényi, Krzysztof Dembczynski, and Eyke Hüllermeier. 2015. Online F-measure optimization. In *Advances in Neural Information Processing Systems (NIPS)*. Montreal, Canada, 595–603.

[7] Ricardo Cerri, Rodrigo C Barros, and André CPLF De Carvalho. 2014. Hierarchical multi-label classification using local neural networks. *J. Comput. System Sci.* 80, 1 (2014), 39–56.

[8] Yao-Nan Chen and Hsuan-Tien Lin. 2012. Feature-aware label space dimension reduction for multi-label classification. In *Advances in Neural Information Processing Systems (NIPS)*. Lake Tahoe, NV, 1529–1537.

[9] Anna E Choromanska and John Langford. 2015. Logarithmic time online multiclass prediction. In *Advances in Neural Information Processing Systems (NIPS)*. Montreal, Canada, 55–63.

[10] Moustapha M Cisse, Nicolas Usunier, Thierry Artieres, and Patrick Gallinari. 2013. Robust bloom filters for large multilabel classification tasks. In *Advances in Neural Information Processing Systems (NIPS)*. Lake Tahoe, NV, 1851–1859.

[11] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[12] Minlie Huang, Aurélie Névéol, and Zhiyong Lu. 2011. Recommending MeSH terms for annotating biomedical articles. *Journal of the American Medical Informatics Association* 18, 5 (2011), 660–667.

[13] Kalina Jasinska, Krzysztof Dembczynski, Róbert Busa-Fekete, Karlson Pfannschmidt, Timo Klerx, and Eyke Hullermeier. 2016. Extreme F-measure maximization using sparse probability estimates. In *International Conference on Machine Learning (ICML)*. New York, NY, 1435–1444.

[14] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar, 1746–1751.

[15] Dingcheng Li, Jingyuan Zhang, and Ping Li. 2018. Representation Learning for Question Classification via Topic Sparse Autoencoder and Entity Embedding. In *2018 IEEE International Conference on Big Data (IEEE Big Data)*. Seattle, WA, 126–133.

[16] Dingcheng Li, Jingyuan Zhang, and Ping Li. 2019. TMSA: A Mutual Learning Model for Topic Discovery and WordEmbedding. In *In Proceedings of the SIAM conference on Data Mining (SDM)*. Calgary, Canada.

[17] Ping Li. 2009. Abc-boost: Adaptive base class boost for multi-class classification. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*. Montreal, Canada, 625–632.

[18] Ping Li. 2010. Robust LogitBoost and Adaptive Base Class (ABC) LogitBoost. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence (UAI)*. Catalina Island, CA, 302–311.

[19] DA Lindberg. 1999. Internet access to the National Library of Medicine. *Effective clinical practice: ECP* 3, 5 (1999), 256–260.

[20] Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep Learning for Extreme Multi-label Text Classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. Tokyo, Japan, 115–124.

[21] Ke Liu, Shengwen Peng, Junqiu Wu, Chengxiang Zhai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2015. MeSHLabeler: improving the accuracy of large-scale MeSH indexing by integrating diverse evidence. *Bioinformatics* 31, 12 (2015), i339–i347.

[22] Yuqing Mao and Zhiyong Lu. 2017. MeSH Now: automatic MeSH indexing at PubMed scale via learning to rank. *Journal of biomedical semantics* 8, 1 (2017), 15.

[23] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer Sentinel Mixture Models. *CoRR* abs/1609.07843 (2016). arXiv:1609.07843 http://arxiv.org/abs/1609.07843

[24] Haitao Mi, Baskaran Sankaran, Zhiguo Wang, and Abe Ittycheriah. 2016. Coverage Embedding Models for Neural Machine Translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Austin, TX, 955–960.

[25] Paul Mineiro and Nikos Karampatziakis. 2015. Fast label embeddings via randomized linear algebra. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*. Porto, Portugal, 37–51.

[26] Xin Mu, Feida Zhu, Yue Liu, Ee-Peng Lim, and Zhi-Hua Zhou. 2018. Social Stream Classification with Emerging New Labels. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*. Melbourne, Australia, 16–28.

[27] David Newman, Sarvnaz Karimi, and Lawrence Cavedon. 2009. Using Topic Models to Interpret MEDLINE's Medical Subject Headings. In *AI 2009: Advances in Artificial Intelligence*, Ann Nicholson and Xiaodong Li (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 270–279.

[28] Shengwen Peng, Ronghui You, Hongning Wang, Chengxiang Zhai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2016. DeepMeSH: deep semantic representation for improving large-scale MeSH indexing. *Bioinformatics* 32, 12 (2016), i70–i79.

[29] Yashoteja Prabhu and Manik Varma. 2014. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*. New York, NY, 263–272.

[30] Baskaran Sankaran, Haitao Mi, Yaser Al-Onaizan, and Abe Ittycheriah. 2016. Temporal Attention Model for Neural Machine Translation. *CoRR* abs/1608.02927 (2016). arXiv:1608.02927 http://arxiv.org/abs/1608.02927

[31] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*. Vancouver, Canada, 1073–1083.

[32] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling Coverage for Neural Machine Translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*. Berlin, Germany, 76–85.

[33] Jason Weston, Ameesh Makadia, and Hector Yee. 2013. Label partitioning for sublinear ranking. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*. Atlanta, GA, 181–189.

[34] Ting-Fan Wu, Chih-Jen Lin, and Ruby C Weng. 2004. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research* 5, Aug (2004), 975–1005.

[35] Yan Yan, Xu-Cheng Yin, Chun Yang, Sujian Li, and Bo-Wen Zhang. 2018. Biomedical literature classification with a CNNs-based hybrid learning network. *PloS one* 13, 7 (2018), e0197933.

[36] Ian En-Hsu Yen, Xiangru Huang, Pradeep Ravikumar, Kai Zhong, and Inderjit Dhillon. 2016. Pd-sparse: A primal and dual sparse approach to extreme multiclass and multilabel classification. In *International Conference on Machine Learning (ICML)*. New York, NY, 3069–3077.

[37] Wenjie Zhang, Junchi Yan, Xiangfeng Wang, and Hongyuan Zha. 2018. Deep Extreme Multi-label Learning. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval (ICMR)*. Yokohama, Japan, 100–107.

Current Flow Group Closeness Centrality for Complex Networks*

Huan Li
Fudan University
Shanghai, China
huanli16@fudan.edu.cn

Richard Peng
Georgia Institute of Technology
Atlanta, Georgia, United States
rpeng@cc.gatech.edu

Liren Shan
Fudan University
Shanghai, China
13307130150@fudan.edu.cn

Yuhao Yi
Fudan University
Shanghai, China
yhyi15@fudan.edu.cn

Zhongzhi Zhang
Fudan University
Shanghai, China
zhangzz@fudan.edu.cn

ABSTRACT

The problem of selecting a group of vertices under certain constraints that maximize their joint centrality arises in many practical scenarios. In this paper, we extend the notion of current flow closeness centrality (CFCC) to a set of vertices in a graph, and investigate the problem of selecting a subset S to maximizes its CFCC $C(S)$, with the cardinality constraint $|S| = k$. We show the NP-hardness of the problem, but propose two greedy algorithms to minimize the reciprocal of $C(S)$. We prove the approximation ratios by showing the monotonicity and supermodularity. A proposed deterministic greedy algorithm has an approximation factor $(1 - \frac{k}{k-1} \cdot \frac{1}{e})$ and cubic running time. To compare with, a proposed randomized algorithm gives $(1 - \frac{k}{k-1} \cdot \frac{1}{e} - \epsilon)$-approximation in nearly-linear time, for any $\epsilon > 0$. Extensive experiments on model and real networks demonstrate the effectiveness and efficiency of the proposed algorithms, with the randomized algorithm being applied to massive networks with more than a million vertices.

CCS CONCEPTS

• **Theory of computation → Graph algorithms analysis**; **Discrete optimization**; • **Information systems → Data mining**.

KEYWORDS

Social Networks, Centrality, Combinatorial Optimization, Spectral Graph Theory

ACM Reference Format:
Huan Li, Richard Peng, Liren Shan, Yuhao Yi, and Zhongzhi Zhang. 2019. Current Flow Group Closeness Centrality for Complex Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313490

*Huan Li, Liren Shan, Yuhao Yi, and Zhongzhi Zhang are with Shanghai Key Laboratory of Intelligent Information Processing, School of Computer Science, Fudan University, Shanghai 200433.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313490

1 INTRODUCTION

A fundamental problem in network science and graph mining is to identify crucial vertices [35, 38]. It is an important tool in network analysis and found numerous applications in various areas [44]. The first step of finding central vertices is to define suitable indices measuring relative importance of vertices. Over the past decades, many centrality measures were introduced to characterize and analyze the roles of vertices in networks [5, 8, 9, 53]. Among them, a popular one is closeness centrality [3, 4]: the closeness of a vertex is the reciprocal of the sum of shortest path distances between it and all other vertices. However, this metric considers only the shortest paths, and more importantly, neglects contributions from other paths. Therefore it can produce some odd effects, or even counterintuitive results [7]. To avoid this shortcoming, Brandes and Fleischer presented current flow closeness centrality [11] based on electrical networks [19] , which takes into account contributions from all paths between vertices. Current flow based closeness has been shown to better discriminate vertices than its traditional counterparts [7].

While most previous works focus on measures and algorithms for the importance of individual vertices in networks [38], the problem of determining a group of k most important vertices arises frequently in data mining and graph applications. For example, in social networks, retailers may want to choose k vertices as promoters of product, such that the number of the potentially influenced customers is maximized [31]. Another example is P2P networks, where one wants to place resources on a fixed number of k peers so they are easily accessed by others [25]. In order to measure the importance of a group of vertices, Everett and Borgatti [21] extended the idea of individual centrality to group centrality, and introduced the concepts of group centrality, for example, group closeness. Recently, some algorithms have been developed to compute or estimate group closeness [6, 13, 56]. However, similar to the case of individual vertices, these notions of group centrality also disregard contributions from paths that are not shortest.

In this paper, we extend current flow closeness of individual vertices [7] by proposing current flow closeness centrality (CFCC) for group of vertices. In a graph with n vertices and m edges, the CFCC $C(S)$ of a vertex group $S \subset V$ is equal to the ratio of n to the sum of effective resistances between S and all vertices u in $V \setminus S$. We then consider the optimization problem: how can we find a group

S^* of k vertices so as to maximize $C(S^*)$. We solve this problem by considering an equivalent problem of minimizing the reciprocal of $C(S^*)$. We show that the problem is NP-hard in Section 4, but also prove that the problem is an instance of supermodular set function optimization with cardinality constraint in Section 5. The latter allows us to devise greedy algorithms to solve this problem, leading to two greedy algorithms with provable approximation guarantees:

(1) A deterministic algorithm with a $(1-\frac{k}{k-1}\cdot\frac{1}{e})$ approximation factor and $O(n^3)$ running time (Section 6);
(2) A randomized algorithm with a $(1-\frac{k}{k-1}\cdot\frac{1}{e}-\epsilon)$-approximation factor and $\widetilde{O}(km\epsilon^{-2})$ running time[1] for any small $\epsilon > 0$ (Section 7).

A key ingredient of our second algorithm is nearly linear time solvers for Laplacians and symmetric, diagonally dominant, M-matrices (SDDM) [16, 48], which has been used in various optimization problems on graphs [17, 30, 41].

We perform extensive experiments on some networks to evaluate our algorithm, and some of their results are in Section 8. These results show that both algorithms are effective. Moreover, the second algorithm is efficient and is scalable to large networks with more than a million vertices.

1.1 Related Works

There exist various measures for centrality of a group of vertices, based on graph structure or dynamic processes, such as betweenness [18, 22, 39, 55], absorbing random-walk centrality [37, 40, 57], and grounding centrality [14, 46]. Since the criterion for importance of a vertex group is application dependent [24], many previous works focus on selecting (or deleting) a group of k vertices (for some given k) in order to optimize related quantities. These quantities are often measures of vertex group importance motivated by the applications, including minimizing the leading eigenvalue of adjacency matrix for vertex immunization [12, 51], minimizing the mean steady-state variance for first-order leader-follower noisy consensus dynamics [15, 45], maximizing average distance for identifying structural hole spanners [47, 54], and others.

Previous works on closeness centrality and related algorithms are most directly related to our focus on the group closeness centrality in this paper. The closeness centrality for an individual vertex was proposed [3] and formalized [4] by Bavelas. For a given vertex, its closeness centrality is defined as the reciprocal of the sum of shortest path distances of the vertex to all the other vertices. Everett and Borgatti [21] extended the individual closeness centrality to group closeness centrality, which measures how close a vertex group is to all other vertices. For a graph with n vertices and m edges, exactly computing the closeness centrality of a group of vertices involves calculating all-pairwise shortest path length, the time complexity of the state-of-the-art algorithm [28] for which is $O(nm + n^2 \log n)$. To reduce the computation complexity, various approximation algorithms were developed. A greedy $O(n^3)$ algorithm with approximation ratio $\left(1-\frac{k}{k-1}\cdot\frac{1}{e}\right)$ was devised [13], and a sampling algorithm that scales better to large networks, but without approximation guarantee was also proposed in the same paper. Very recently, new techniques [6] have been developed to speed up the greedy algorithm in [13] while preserving its theoretical guarantees.

Conventional closeness centrality is based on the shortest paths, omitting the contributions from other paths. In order to overcome this drawback, Brandes and Fleischer introduced current flow closeness centrality for an individual vertex [11], which essentially considers all paths between vertices, but still gives large weight to short paths. Our investigation can be viewed as combining this line of current based centrality measures with the study of selecting groups of k vertices.

[1]We use the $\widetilde{O}(\cdot)$ notation to hide the poly log factors.

2 PRELIMINARIES

In this section, we briefly introduce some useful notations and tools for the convenience of description of our problem and algorithms.

2.1 Notations

We use normal lowercase letters like a, b, c to denote scalars in $\mathbb{R}$, normal uppercase letters like A, B, C to denote sets, bold lowercase letters like $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ to denote vectors, and bold uppercase letters like $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ to denote matrices. We write $\boldsymbol{a}_{[i]}$ to denote the i^{th} entry of vector $\boldsymbol{a}$ and $\boldsymbol{A}_{[i,j]}$ to denote entry (i, j) of matrix $\boldsymbol{A}$. We also write $\boldsymbol{A}_{[i,:]}$ to denote the i^{th} row of $\boldsymbol{A}$ and $\boldsymbol{A}_{[:,j]}$ to denote the j^{th} column of $\boldsymbol{A}$.

We write sets in matrix subscripts to denote submatrices. For example, $\boldsymbol{A}_{[I,J]}$ denotes the submatrix of $\boldsymbol{A}$ with row indices in I and column indices in J. To simplify notation, we also write $\boldsymbol{A}_{-i}$ to denote the submatrix of $\boldsymbol{A}$ obtained by removing the i^{th} row and i^{th} column of $\boldsymbol{A}$. For example, for an $n \times n$ matrix $\boldsymbol{A}$, $\boldsymbol{A}_{-n}$ denotes the submatrix $\boldsymbol{A}_{[1:n-1,1:n-1]}$.

Note that the precedence of matrix subscripts is the lowest. Thus, $\boldsymbol{A}_{-n}^{-1}$ denotes the inverse of $\boldsymbol{A}_{-n}$ instead of a submatrix of $\boldsymbol{A}^{-1}$.

For two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, we write $\boldsymbol{A} \preceq \boldsymbol{B}$ to denote that $\boldsymbol{B} - \boldsymbol{A}$ is positive semidefinite, i.e., $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \le \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x}$ holds for every real vector $\boldsymbol{x}$.

We use $\boldsymbol{e}_i$ to denote the i^{th} standard basis vector of appropriate dimension, and $\mathbf{1}_S$ to denote the indicator vector of S.

2.2 Graphs, Laplacians, and Effective Resistances

We write $G = (V, E, w)$ to denote a positively weighted undirected graph with n vertices, m edges, and edge weight function $w : E \to \mathbb{R}^+$. The Laplacian matrix $\boldsymbol{L}$ of G is defined as $\boldsymbol{L}_{[u,v]} = -w(u, v)$ if $u \sim v$, $\boldsymbol{L}_{[u,v]} = \deg(u)$ if $u = v$, and $\boldsymbol{L}_{[u,v]} = 0$ otherwise, where $\deg(u) \stackrel{\text{def}}{=} \sum_{u\sim v} w(u, v)$ is the weighted degree of u and $u \sim v$ means $(u, v) \in E$. Let $w_{\max}$ and $w_{\min}$ denote, respectively, the maximum weight and minimum weight among all edges. If we orient each edge of G arbitrarily, we can also write it's Laplacian as $\boldsymbol{L} = \boldsymbol{B}^T\boldsymbol{W}\boldsymbol{B}$, where $\boldsymbol{B}_{m\times n}$ is the signed edge-vertex incidence matrix defined by $\boldsymbol{B}_{[e,u]} = 1$ if u is e's head, $\boldsymbol{B}_{[e,u]} = -1$ if u is e's tail, and $\boldsymbol{B}_{[e,u]} = 0$ otherwise, and $\boldsymbol{W}_{m\times m}$ is a diagonal matrix with $\boldsymbol{W}_{[e,e]} = w(e)$. It is not hard to show that quadratic forms of $\boldsymbol{L}$ can be written as $\boldsymbol{x}^T\boldsymbol{L}\boldsymbol{x} = \sum_{u\sim v} w(u, v)\left(\boldsymbol{x}_{[u]} - \boldsymbol{x}_{[v]}\right)^2$, which immediately implies that $\boldsymbol{L}$ is positive semidefinite, and $\boldsymbol{L}$ only has one zero eigenvalue if G is a connected graph.

The following fact shows that submatrices of Laplacians are always positive definite and inverse-positive.

Fact 2.1. *Let $\boldsymbol{L}$ be the Laplacian of a connected graph and let $\boldsymbol{X}$ be a nonnegative, diagonal matrix with at least one nonzero entry. Then, $\boldsymbol{L}+\boldsymbol{X}$ is positive definite, and every entry of $(\boldsymbol{L}+\boldsymbol{X})^{-1}$ is positive.*

Let $0 = \lambda_1 < \lambda_2 \leq \ldots \leq \lambda_n$ be eigenvalues of $\boldsymbol{L}$ of a connected graph G, and $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n$ be the corresponding orthonormal eigenvectors. Then we can decompose $\boldsymbol{L}$ as $\boldsymbol{L} = \sum_{i=2}^{n} \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^T$ and define its pseudoinverse as $\boldsymbol{L}^{\dagger} = \sum_{i=2}^{n} \frac{1}{\lambda_i} \boldsymbol{v}_i \boldsymbol{v}_i^T$.

It is not hard to verify that if $\boldsymbol{L}$ and $\boldsymbol{H}$ are Laplacians of connected graphs supported on the same vertex set, then $\boldsymbol{L} \preceq \boldsymbol{H}$ implies $\boldsymbol{H}^{\dagger} \preceq \boldsymbol{L}^{\dagger}$.

The pseudoinverse of Laplacian matrix can be used to define effective resistance between any pair of vertices [32].

Definition 2.2. For a connected graph $G = (V, E, w)$ with Laplacian matrix $\boldsymbol{L}$, the effective resistance between vertices u and v is defined as $R_{\text{eff}}(u, v) = (\boldsymbol{e}_u - \boldsymbol{e}_v)^T \boldsymbol{L}^{\dagger} (\boldsymbol{e}_u - \boldsymbol{e}_v)$.

The effective resistance between two vertices can also be expressed in term of the diagonal elements of the inverse for submatrices of $\boldsymbol{L}$.

Fact 2.3 ([27]). $R_{\text{eff}}(u, v) = \left(\boldsymbol{L}_{-u}^{-1}\right)_{[v,v]} = \left(\boldsymbol{L}_{-v}^{-1}\right)_{[u,u]}$.

2.3 Current Flow Closeness Centrality

The current flow closeness centrality was proposed in [11]. It is based on the assumption that information spreads efficiently like an electrical current.

To define current flow closeness, we treat the graph G as a resistor network via replacing every edge e by a resistor with resistance $r_e = 1/w(e)$. Let $v_{st}(u)$ denote the voltage of u when a unit current enters the network at s and leaves it at t.

Definition 2.4. The current flow closeness $C(u)$ of a vertex u is defined as $C(u) = n/\sum_{v \in V} (v_{uv}(u) - v_{uv}(v))$.

It has been proved [11] that the current flow closeness of vertex u equals the ratio of n to the sum of effective resistances between u and other vertices.

Fact 2.5. $C(u) = n/\sum_{v \in V} R_{\text{eff}}(u, v)$.

Actually, current flow closeness centrality is equivalent to information centrality [50].

2.4 Supermodular Functions

We now give the definitions for monotone and supermodular set functions. For simplicity, we write $S + u$ to denote $S \cup \{u\}$ and $S - u$ to denote $S \setminus \{u\}$.

Definition 2.6 (Monotonicity)**.** A set function $f : 2^V \to \mathbb{R}$ is monotone if $f(S) \geq f(T)$ holds for all $S \subseteq T$.

Definition 2.7 (Supermodularity)**.** A set function $f : 2^V \to \mathbb{R}$ is supermodular if $f(S) - f(S + u) \geq f(T) - f(T + u)$ holds for all $S \subseteq T \subseteq V$ and $u \in V$.

3 CURRENT FLOW CLOSENESS OF A GROUP OF VERTICES

We follow the idea of [11] to define current flow closeness centrality (CFCC) of a group of vertices.

To define current flow closeness centrality for a vertex set $S \subseteq V$, we treat the graph G as a resistor network in which all vertices in S are grounded. Thus, vertices in S always have voltage 0. For a vertex $u \notin S$, let $v_{uS}(v)$ be the voltage of v when a unit current enters the network at u and leaves it at S (i.e. the ground). Then, we define the current flow closeness of S as follows.

Definition 3.1. Let $G = (V, E, w)$ be a connected weighted graph. The current flow closeness centrality $C(S)$ of a vertex group $S \subseteq V$ is defined as $C(S) = n/\sum_{u \in V} (v_{uS}(u) - v_{uS}(S)) = n/\sum_{u \in V} v_{uS}(u)$.

Note that there are different variants of the definition of CFCC for a vertex group. For example, we can use$(n - |S|) / (\sum_{u \in V} v_{uS}(u))$ as the measure of CFCC for a vertex set S. Definition 3.1 adopts the standard form as the classic closeness centrality [13].

We next show that $C(S)$ is in fact equal to the ratio of n to a sum of effective resistances as in Fact 2.5.

Let $u \in V$ be a fixed vertex. Suppose there is a unit current enters the network at u and leaves it at S. Let $\boldsymbol{v} \in \mathbb{R}^n$ be a vector of voltages at vertices. By Kirchhoff's Current Law and Ohm's Law, we have $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{e}_u - \boldsymbol{i}_S$, where $\boldsymbol{i}_S$ denotes the amount of current flowing out of S. Since vertices in S all have voltage 0, we can restrict this equation to vertices in $V - S$ as $\boldsymbol{L}_{[V-S, V-S]} \boldsymbol{v}_{[V-S]} = \boldsymbol{e}_u$, which leads to $\boldsymbol{v}_{[V-S]} = \boldsymbol{L}_{[V-S, V-S]}^{-1} \boldsymbol{e}_u$. This gives the expression of voltage at u as $v_{uS}(u) = \boldsymbol{e}_u^T \boldsymbol{v} = \boldsymbol{e}_u^T \boldsymbol{L}_{-S}^{-1} \boldsymbol{e}_u$. Now we can write the CFCC of S as

$$C(S) = \frac{n}{\sum_{u \in V} v_{uS}(u)} = \frac{n}{\sum_{u \in V - S} \boldsymbol{e}_u^T \boldsymbol{L}_{-S}^{-1} \boldsymbol{e}_u} = \frac{n}{\text{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right)}.$$

Note that the diagonal entry $\left(\boldsymbol{L}_{-S}^{-1}\right)_{[u,u]}$ of $\left(\boldsymbol{L}_{-S}^{-1}\right)$ is exactly the effective resistance $R_{\text{eff}}(u, S)$ between vertex u and vertex set S [15], with $R_{\text{eff}}(u, S) = 0$ for any $u \in S$. Then we have the following relation governing $C(S)$ and $R_{\text{eff}}(u, S)$.

Fact 3.2. $C(S) = n/\text{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right) = n/\sum_{u \in V} R_{\text{eff}}(u, S)$.

Being able to define CFCC of a vertex set raises the problem of maximizing current flow closeness subject to a cardinality constraint, which we state below.

Problem 1 (Current Flow Closeness Maximization, CFCM). *Given a connected graph $G = (V, E, w)$ with n vertices, m edges, and edge weight function $w : E \to \mathbb{R}^+$ and an integer $1 \leq k \leq n$, find a vertex group $S^* \in V$ such that the CFCC $C(S^*)$ is maximized, that is* $S^* \in \underset{S \subseteq V, |S|=k}{\arg\max} \; C(S)$.

4 HARDNESS OF CURRENT FLOW CLOSENESS MAXIMIZATION

In this section, we prove that Problem 1 is NP-hard. We will give a reduction from vertex cover on 3-regular graphs (graphs whose vertices all have degree 3), which is an NP-complete problem [23]. The decision version of this problem is stated below.

Problem 2 (Vertex Cover on 3-regular graphs, VC3). *Given a connected 3-regular graph $G = (V, E)$ and an integer k, decide whether or not there is a vertex set $S \subset V$ such that $|S| \leq k$ and S is a vertex cover of G (i.e. every edge in E is incident with at least one vertex in S).*

We then give the decision version of Problem 1.

Problem 3 (Current Flow Closeness Maximization, Decision Version, CFCMD). *Given a connected graph $G = (V, E, w)$, an integer k, and a real number $r \in \mathbb{R}$, decide whether or not there is a vertex set $S \subset V$ such that $|S| \leq k$ and $C(S) \geq r$.*

To give the reduction, we will need the following lemma.

Lemma 4.1. *Let $G = (V, E, w)$ be a connected 3-regular graph with all edge weights being 1 (i.e. $w(e) = 1$ for all $e \in E$). Let $S \subset V$ be a nonempty vertex set, and $k = |S|$. Then, $C(S) \leq 3n/(n-k)$ and the equality holds if and only if S is a vertex cover of G.*

Proof. We first show that if S is a vertex cover of G then $C(S) = 3n/(n-k)$. When S is a vertex cover, $V \setminus S$ is an independent set. Thus, $\boldsymbol{L}_{-S}$ is a diagonal matrix with all diagonal entries being 3. So we have

$$C(S) = n/\mathrm{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right) = n/\mathrm{Tr}\left((\mathrm{diag}(3,\ldots,3))^{-1}\right) = 3n/(n-k).$$

We then show that if S is not a vertex cover of G then $C(S) < 3n/(n-k)$. When S is not a vertex cover, $V \setminus S$ is not an independent set. Thus, $\boldsymbol{L}_{-S}$ is a block diagonal matrix, with each block corresponding to a connected component of $G[V \setminus S]$, the induced graph of G on $V \setminus S$. Let $T \subseteq V \setminus S$ be a connected component of $G[V \setminus S]$ such that $|T| > 1$. Then, the block of $\boldsymbol{L}_{-S}$ corresponding to T is $\boldsymbol{L}_{[T,T]}$. For a vertex $u \in T$, let the u^{th} column of $\boldsymbol{L}_{[T,T]}$ be $\begin{pmatrix} 3 \\ -\boldsymbol{a} \end{pmatrix}$. Then, we can write $\boldsymbol{L}_{[T,T]}$ into block form as

$$\boldsymbol{L}_{[T,T]} = \begin{pmatrix} 3 & -\boldsymbol{a}^T \\ -\boldsymbol{a} & \boldsymbol{A} \end{pmatrix},$$

where $\boldsymbol{A} \stackrel{\text{def}}{=} \boldsymbol{L}_{[T-u,T-u]}$. By blockwise matrix inversion we have

$$\left(\boldsymbol{L}_{[T,T]}^{-1}\right)_{[u,u]} = 1/\left(3 - \boldsymbol{a}^T\boldsymbol{A}^{-1}\boldsymbol{a}\right).$$

Since $\boldsymbol{L}_{[T,T]}^{-1}$ is positive definite, we have $1/(3 - \boldsymbol{a}^T\boldsymbol{A}^{-1}\boldsymbol{a}) > 0$ and hence $\boldsymbol{a}^T\boldsymbol{A}^{-1}\boldsymbol{a} < 3$. Since T is a connected component, $\boldsymbol{a}$ is not a zero vector, which coupled with the fact that $\boldsymbol{A}^{-1}$ is positive definite gives $\boldsymbol{a}^T\boldsymbol{A}^{-1}\boldsymbol{a} > 0$. Thus, $1/(3 - \boldsymbol{a}^T\boldsymbol{A}^{-1}\boldsymbol{a}) > 1/3$. Since this holds for all $u \in T$, we have $\mathrm{Tr}\left(\boldsymbol{L}_{[T,T]}^{-1}\right) > |T|/3$. Also, since T can be any connected component of $G[V \setminus S]$ with at least two vertices, and a block of an isolate vertex in $G[V \setminus S]$ contributes a 1/3 to $\mathrm{Tr}\left((\boldsymbol{L}_{-S})^{-1}\right)$, we have for any S which is not a vertex cover of G

$$\mathrm{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right) > |V \setminus S|/3 = (n-k)/3,$$

which implies $C(S) = n/\mathrm{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right) < 3n/(n-k)$. This completes the proof. □

The following theorem then follows by Lemma 4.1.

Theorem 4.2. *Maximizing current flow closeness subject to a cardinality constraint is NP-hard.*

Proof. We give a polynomial reduction

$$p : \{(G = (V, E), k)\} \to \{(G = (V, E, w), k, r)\}$$

from instances of VC3 to instances of CFCMD. For a connected 3-regular graph $G = (V, E)$ with n vertices, we construct a weighted graph $G' = (V, E, w_1)$ with the same vertex set and edge set and an edge weight function $w_1 : E \to \{1\}$ mapping all edges to weight 1. Then, we construct a reduction p as

$$p\left((G = (V, E), k)\right) = \left(G' = (V, E, w_1), k, 3n/(n-k)\right).$$

By Lemma 4.1, p is a polynomial reduction from VC3 to CFCMD, which implies that CFCM is NP-hard. □

5 SUPERMODULARITY OF THE RECIPROCAL OF CURRENT FLOW GROUP CLOSENESS

In this section, we prove that the reciprocal of current flow group closeness, i.e., $\mathrm{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right)/n$, is a monotone supermodular function. Our proof uses the following lemma, which shows that $\boldsymbol{L}_{-S}^{-1}$ is entrywise supermodular.

Lemma 5.1. *Let $u, v \in V$ be an arbitrary pair of vertices. Then, the entry $\left(\boldsymbol{L}_{-S}^{-1}\right)_{[u,v]}$ is a monotone supermodular function. Namely, for vertices $u, v \neq w$ and nonempty vertex sets $S \subseteq T \subseteq V$ such that $u, v, w \notin T$,*

$$\left(\boldsymbol{L}_{-S}^{-1}\right)_{[u,v]} \geq \left(\boldsymbol{L}_{-T}^{-1}\right)_{[u,v]}$$

and

$$\left(\boldsymbol{L}_{-S}^{-1}\right)_{[u,v]} - \left(\boldsymbol{L}_{-(S+w)}^{-1}\right)_{[u,v]} \geq \left(\boldsymbol{L}_{-T}^{-1}\right)_{[u,v]} - \left(\boldsymbol{L}_{-(T+w)}^{-1}\right)_{[u,v]}.$$

To prove Lemma 5.1, we first define a linear relaxation $\boldsymbol{H} : [0,1]^n \to \mathbb{R}^{n\times n}$ of $\boldsymbol{L}_{-S}$ as

$$\left(\boldsymbol{H}(\boldsymbol{x})\right)_{[u,v]} = \begin{cases} \boldsymbol{L}_{[u,u]} & \text{if } u = v, \\ \left(1 - \boldsymbol{x}_{[u]}\right) \cdot \left(1 - \boldsymbol{x}_{[v]}\right) \cdot \boldsymbol{L}_{[u,v]} & \text{if } u \neq v. \end{cases} \tag{1}$$

Remark 5.2. We remark the intuition behind this relaxation $\boldsymbol{H}(\boldsymbol{x})$. Let $\boldsymbol{x}_{[=1]}$ denote the indices of entries of $\boldsymbol{x}$ equal to one, and let $\boldsymbol{x}_{[<1]}$ denote the indices of entries of $\boldsymbol{x}$ less than one. Then, by the definition in (1), we can write $\boldsymbol{H}(\boldsymbol{x})$ into a block diagonal matrix as

$$\boldsymbol{H}(\boldsymbol{x}) = \begin{pmatrix} \left(\boldsymbol{H}(\boldsymbol{x})\right)_{[\boldsymbol{x}_{[=1]},\boldsymbol{x}_{[=1]}]} & \boldsymbol{0} \\ \boldsymbol{0} & \left(\boldsymbol{H}(\boldsymbol{x})\right)_{[\boldsymbol{x}_{[<1]},\boldsymbol{x}_{[<1]}]} \end{pmatrix},$$

where $\left(\boldsymbol{H}(\boldsymbol{x})\right)_{[\boldsymbol{x}_{[=1]},\boldsymbol{x}_{[=1]}]}$ is itself a diagonal matrix. This means that if $\boldsymbol{x} = \boldsymbol{1}_S$ for some nonempty vertex set $S \subseteq V$, the following statement holds:

$$\left(\boldsymbol{H}^{-1}(\boldsymbol{1}_S)\right)_{-S} = \boldsymbol{L}_{-S}^{-1}.$$

The condition that every entry of $\boldsymbol{x}$ is in $[0,1]$ coupled with Fact 2.1 also implies that all submatrices of $\boldsymbol{H}(\boldsymbol{x})$ are positive definite and inverse-positive.

Now for vertices $u, v \neq w$ and nonempty vertex set $S \subseteq V$ such that $u, v, w \notin S$, we can write the marginal gain of a vertex w as

$$\left(\boldsymbol{L}_{-S}^{-1} - \boldsymbol{L}_{-(S+w)}^{-1}\right)_{[u,v]} = \left(\boldsymbol{H}^{-1}(\boldsymbol{1}_S) - \boldsymbol{H}^{-1}(\boldsymbol{1}_{S+w})\right)_{[u,v]}. \tag{2}$$

We can further write the matrix on the rhs of (2) as an integral by

$$\begin{aligned}
&\boldsymbol{H}^{-1}(\mathbf{1}_S) - \boldsymbol{H}^{-1}(\mathbf{1}_{S+w}) = -\left.\boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)\right|_0^1 \\
&= -\int_0^1 \frac{\mathrm{d}\boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)}{\mathrm{d}t}\mathrm{d}t \\
&= \int_0^1 \boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)\frac{\mathrm{d}\boldsymbol{H}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)}{\mathrm{d}t}\boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)\mathrm{d}t \\
&= \int_0^1 \boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)\left(-t\boldsymbol{H}(\mathbf{1}_V - \boldsymbol{e}_w)\right)\boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)\mathrm{d}t, \quad (3)
\end{aligned}$$

where the second equality follows by the identity

$$\frac{\mathrm{d}\boldsymbol{A}^{-1}}{\mathrm{d}x} = -\boldsymbol{A}^{-1}\frac{\mathrm{d}\boldsymbol{A}}{\mathrm{d}x}\boldsymbol{A}^{-1}$$

for any invertible matrix $\boldsymbol{A}$. To prove Lemma 5.1, we will also need the following lemma, which shows the entrywise monoticity of $\boldsymbol{H}^{-1}(\boldsymbol{x})$.

Lemma 5.3. *For $0 \le t \le 1$, the following statement holds for any vertices $u, v \ne w$ and nonempty vertex sets $S \subseteq T \subseteq V$ such that $u, v, w \notin T$:*

$$\left(\boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)\right)_{[u,v]} \ge \left(\boldsymbol{H}^{-1}(\mathbf{1}_T + t\cdot\boldsymbol{e}_w)\right)_{[u,v]}.$$

Proof. For simplicity, we let $\boldsymbol{S} \stackrel{\text{def}}{=} \boldsymbol{H}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)$ and $\boldsymbol{T} \stackrel{\text{def}}{=} \boldsymbol{H}(\mathbf{1}_T + t\cdot\boldsymbol{e}_w)$. We also write $P \stackrel{\text{def}}{=} V\setminus T$, $Q \stackrel{\text{def}}{=} V\setminus S$, and $F \stackrel{\text{def}}{=} T\setminus S$. Due to the block diagonal structures of $\boldsymbol{S}$ and $\boldsymbol{T}$, we have

$$\left(\boldsymbol{S}^{-1}\right)_{[u,v]} = \left(\boldsymbol{S}^{-1}_{[Q,Q]}\right)_{[u,v]} \quad (4)$$

and

$$\left(\boldsymbol{T}^{-1}\right)_{[u,v]} = \left(\boldsymbol{T}^{-1}_{[P,P]}\right)_{[u,v]}. \quad (5)$$

Since $\boldsymbol{S}$ and $\boldsymbol{T}$ agree on entries with indices in P, we can write the submatrix $\boldsymbol{S}_{[Q,Q]}$ of $\boldsymbol{S}$ in block form as

$$\boldsymbol{S}_{[Q,Q]} = \begin{pmatrix} \boldsymbol{T}_{[P,P]} & \boldsymbol{S}_{[P,F]} \\ \boldsymbol{S}_{[F,P]} & \boldsymbol{S}_{[F,F]} \end{pmatrix}.$$

By blockwise matrix inversion, we have

$$\begin{aligned}
&\left(\boldsymbol{S}^{-1}_{[Q,Q]}\right)_{[P,P]} \\
=&\boldsymbol{T}^{-1}_{[P,P]} + \boldsymbol{T}^{-1}_{[P,P]}\boldsymbol{S}_{[P,F]}\left(\boldsymbol{S}^{-1}_{[Q,Q]}\right)_{[F,F]}\boldsymbol{S}_{[F,P]}\boldsymbol{T}^{-1}_{[P,P]} \\
=&\boldsymbol{T}^{-1}_{[P,P]} + \boldsymbol{T}^{-1}_{[P,P]}\left(-\boldsymbol{S}_{[P,F]}\right)\left(\boldsymbol{S}^{-1}_{[Q,Q]}\right)_{[F,F]}\left(-\boldsymbol{S}_{[F,P]}\right)\boldsymbol{T}^{-1}_{[P,P]},
\end{aligned}$$

where the second equality follows by negating both $\boldsymbol{S}_{[P,F]}$ and $\boldsymbol{S}_{[F,P]}$. By definition the matrix $-\boldsymbol{S}_{[P,F]}$ is entrywise nonnegative. By Fact 2.1, every entry of $\boldsymbol{S}^{-1}_{[Q,Q]}$ and $\boldsymbol{T}^{-1}_{[P,P]}$ is also nonnegative. Thus, the matrix

$$\boldsymbol{T}^{-1}_{[P,P]}\left(-\boldsymbol{S}_{[P,F]}\right)\left(\boldsymbol{S}^{-1}_{[Q,Q]}\right)_{[F,F]}\left(-\boldsymbol{S}_{[F,P]}\right)\boldsymbol{T}^{-1}_{[P,P]}$$

is entrywise nonnegative, which coupled with (4) and (5), implies $\left(\boldsymbol{S}^{-1}\right)_{[u,v]} \ge \left(\boldsymbol{T}^{-1}\right)_{[u,v]}$. □

We are now ready to prove Lemma 5.1.

Proof of Lemma 5.1. By definition the matrix $-t\boldsymbol{H}(\mathbf{1}_V - \boldsymbol{e}_w)$ is entrywise nonnegative when $t \ge 0$. By Fact 2.1, the matrix $\boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)$ is also entrywise nonnegative when $0 \le t \le 1$. Thus, the derivative in (3) is entrywise nonnegative, which implies the the monotonicity of $\left(\boldsymbol{L}^{-1}_{-S}\right)_{[u,v]}$ for any pair of vertices $u, v \in V$.

We then prove the supermodularity, i.e.,

$$\left(\boldsymbol{L}^{-1}_{-S} - \boldsymbol{L}^{-1}_{-(S+w)}\right)_{[u,v]} \ge \left(\boldsymbol{L}^{-1}_{-T} - \boldsymbol{L}^{-1}_{-(T+w)}\right)_{[u,v]} \quad (6)$$

for any $S \subseteq T \subseteq V$ and $u, v \notin T$. Lemma 5.3, coupled with the fact that $\boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)$ and $-t\boldsymbol{H}(\mathbf{1}_V - \boldsymbol{e}_w)$ are both entrywise nonnegative, gives the entrywise monotonicity of the derivative in (3) as

$$\begin{aligned}
&\left(\boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)\left(-t\boldsymbol{H}(\mathbf{1}_V - \boldsymbol{e}_w)\right)\boldsymbol{H}^{-1}(\mathbf{1}_S + t\cdot\boldsymbol{e}_w)\right)_{[u,v]} \\
\ge&\left(\boldsymbol{H}^{-1}(\mathbf{1}_T + t\cdot\boldsymbol{e}_w)\left(-t\boldsymbol{H}(\mathbf{1}_V - \boldsymbol{e}_w)\right)\boldsymbol{H}^{-1}(\mathbf{1}_T + t\cdot\boldsymbol{e}_w)\right)_{[u,v]}.
\end{aligned}$$

Integrating both sides of the above inequality with respect to t on the interval $[0, 1]$ gives (6). □

The following theorem then directly follows by Lemma 5.1.

Theorem 5.4. *The reciprocal of current flow group centrality, i.e., $\mathrm{Tr}\left(\boldsymbol{L}^{-1}_{-S}\right)/n$, is a monotone supermodular function.*

We note that [15] has previously proved that $\mathrm{Tr}\left(\boldsymbol{L}^{-1}_{-S}\right)$ is monotone and supermodular by using the connection between effective resistance and commute time for random walks. However, our proof is fully algebraic. Moreover, we present a more general result that $\boldsymbol{L}^{-1}_{-S}$ is entrywise supermodular.

By Theorem 5.4, one can obtain a $(1 - \frac{k}{k-1}\cdot\frac{1}{e})$-approximation to the optimum $\mathrm{Tr}\left(\boldsymbol{L}^{-1}_{-S^*}\right)$ by a simple greedy algorithm, by picking the vertex with the maximum marginal gain each time [42]. However, since computing $\mathrm{Tr}\left(\boldsymbol{L}^{-1}_{-S}\right)$ involves matrix inversions, a naive implementation of this greedy algorithm will take $O(kn^4)$ time, assuming that one matrix inversion runs in $O(n^3)$ time. We will show in the next section how to implement this greedy algorithm in $O(n^3)$ time using blockwise matrix inversion.

6 A DETERMINISTIC GREEDY ALGORITHM

We now consider how to accelerate the naive greedy algorithm. Suppose that after the i^{th} step, the algorithm has selected a set S_i containing i vertices. We next compute the marginal gain $\mathrm{Tr}\left(\boldsymbol{L}^{-1}_{-S_i}\right) - \mathrm{Tr}\left(\boldsymbol{L}^{-1}_{-(S_i+u)}\right)$ of each vertex $u \notin S_i$.

For a vertex $u \notin S_i$, let $\begin{pmatrix} d_u \\ -\boldsymbol{a}\end{pmatrix}$ denote the u^{th} column of the submatrix $\boldsymbol{L}_{-S_i}$. Then we write $\boldsymbol{L}_{-S_i}$ in block form as $\boldsymbol{L}_{-S_i} = \begin{pmatrix} d_u & -\boldsymbol{a}^T \\ -\boldsymbol{a} & \boldsymbol{A}\end{pmatrix}$, where $\boldsymbol{A} \stackrel{\text{def}}{=} \boldsymbol{L}_{-(S_i+u)}$. By blockwise matrix inversion, we have

$$\boldsymbol{L}^{-1}_{-S_i} = \begin{pmatrix} \frac{1}{s} & \frac{1}{s}\boldsymbol{a}^T\boldsymbol{A}^{-1} \\ \frac{1}{s}\boldsymbol{A}^{-1}\boldsymbol{a} & \boldsymbol{A}^{-1} + \frac{1}{s}\boldsymbol{A}^{-1}\boldsymbol{a}\boldsymbol{a}^T\boldsymbol{A}^{-1}\end{pmatrix}, \quad (7)$$

where $s = d_u - \boldsymbol{a}^T \boldsymbol{A}^{-1} \boldsymbol{a}$. Then the marginal gain of u can be further expressed as

$$\begin{aligned}
&\operatorname{Tr}\left(\boldsymbol{L}_{-S_i}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-(S_i+u)}^{-1}\right) \\
=&\left(\boldsymbol{L}_{-S_i}^{-1}\right)_{[u,u]} + \sum_{v \in V \setminus (S_i+u)} \left(\left(\boldsymbol{L}_{-S_i}^{-1}\right)_{[v,v]} - \left(\boldsymbol{L}_{-(S_i+u)}^{-1}\right)_{[v,v]}\right) \\
=&\frac{1}{s} + \frac{1}{s}\operatorname{Tr}\left(\boldsymbol{A}^{-1}\boldsymbol{a}\boldsymbol{a}^T\boldsymbol{A}^{-1}\right) = \frac{1}{s} + \frac{1}{s}\boldsymbol{a}^T\boldsymbol{A}^{-1}\boldsymbol{A}^{-1}\boldsymbol{a} \\
=&\left(\boldsymbol{e}_u^T \boldsymbol{L}_{-S_i}^{-2} \boldsymbol{e}_u\right) / \left(\boldsymbol{e}_u^T \boldsymbol{L}_{-S_i}^{-1} \boldsymbol{e}_u\right),
\end{aligned}$$

where the second equality and the fourth equality follow by (7), while the third equality follows by the cyclicity of trace.

By (7), we can also update the inverse $\boldsymbol{L}_{-S_i}^{-1}$ upon a vertex u by

$$\boldsymbol{L}_{-(S_i+u)}^{-1} = \left(\boldsymbol{L}_{-S_i}^{-1} - \left(\boldsymbol{L}_{-S_i}^{-1} \boldsymbol{e}_u \boldsymbol{e}_u^T \boldsymbol{L}_{-S_i}^{-1}\right) / \left(\boldsymbol{e}_u^T \boldsymbol{L}_{-S_i}^{-1} \boldsymbol{e}_u\right)\right)_{-u}.$$

At the first step, we need to pick a vertex u_1 with minimum $\sum_{v \in V} R_{\text{eff}}(u_1, v)$, which can be done by computing $\sum_{v \in V} R_{\text{eff}}(u, v)$ for all $u \in V$ using the relation [10]

$$\sum\nolimits_{v \in V} R_{\text{eff}}(u, v) = n\left(\boldsymbol{L}^{\dagger}\right)_{[u,u]} + \operatorname{Tr}\left(\boldsymbol{L}^{\dagger}\right).$$

We give the $O(n^3 + kn^2)$-time algorithm as follows.

$S_k = \textsc{ExactGreedy}\,(G, \boldsymbol{L}, k)$

(1) Compute $\boldsymbol{L}^{\dagger}$ by inverting $\boldsymbol{L}$ in $O(n^3)$ time.
(2) $S_1 \leftarrow \{u_1\}$ where $u_1 = \arg\min_{u \in V} n\left(\boldsymbol{L}^{\dagger}\right)_{[u,u]} + \operatorname{Tr}\left(\boldsymbol{L}^{\dagger}\right)$.
(3) Compute $\boldsymbol{L}_{-S_1}^{-1}$ in $O(n^3)$ time.
(4) Repeat the following steps for $i = 1, \ldots, k-1$:
  (a) $u_{i+1} \leftarrow \arg\max_{u \in (V \setminus S_i)} \frac{\boldsymbol{e}_u^T \boldsymbol{L}_{-S_i}^{-2} \boldsymbol{e}_u}{\boldsymbol{e}_u^T \boldsymbol{L}_{-S_i}^{-1} \boldsymbol{e}_u}$, $S_{i+1} \leftarrow S_i + u_{i+1}$
  (b) Compute $\boldsymbol{L}_{S_{i+1}}^{-1}$ in $O(n^2)$ time by

$$\boldsymbol{L}_{-(S_i+u_{i+1})}^{-1} = \left(\boldsymbol{L}_{-S_i}^{-1} - \frac{\boldsymbol{L}_{-S_i}^{-1} \boldsymbol{e}_{u_{i+1}} \boldsymbol{e}_{u_{i+1}}^T \boldsymbol{L}_{-S_i}^{-1}}{\boldsymbol{e}_{u_{i+1}}^T \boldsymbol{L}_{-S_i}^{-1} \boldsymbol{e}_{u_{i+1}}}\right)_{-u_{i+1}}.$$

(5) Return S_k.

The performance of ExactGreedy is characterized in the following theorem.

Theorem 6.1. *The algorithm $S_k = \textsc{ExactGreedy}(G, \boldsymbol{L}, k)$ takes an undirected positive weighted graph $G = (V, E, w)$ with associated Laplacian $\boldsymbol{L}$ and an integer $2 \le k \le n$, and returns a vertex set $S_k \subseteq V$ with $|S_k| = k$. The algorithm runs in time $O(n^3)$. The vertex set S_k satisfies*

$$\operatorname{Tr}\left(\boldsymbol{L}_{-u^*}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-S_k}^{-1}\right) \ge \left(1 - \frac{k}{k-1} \cdot \frac{1}{e}\right)\left(\operatorname{Tr}\left(\boldsymbol{L}_{-u^*}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right)\right)$$

where

$$S^* \overset{\text{def}}{=} \arg\min_{|S| \le k} \operatorname{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right), \quad u^* \overset{\text{def}}{=} \arg\min_{u \in V} \sum_{v \in V} R_{\text{eff}}(u, v).$$

Proof. The running time is easy to verify. We only need to prove the approximation ratio.

By supermodularity, for any $i \ge 1$

$$\operatorname{Tr}\left(\boldsymbol{L}_{-S_i}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-S_{i+1}}^{-1}\right) \ge \frac{1}{k}\left(\operatorname{Tr}\left(\boldsymbol{L}_{-S_i}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-S*}^{-1}\right)\right),$$

which implies

$$\operatorname{Tr}\left(\boldsymbol{L}_{-S_{i+1}}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right) \le \left(1 - \frac{1}{k}\right)\left(\operatorname{Tr}\left(\boldsymbol{L}_{-S_i}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right)\right).$$

Then, we have

$$\begin{aligned}
&\operatorname{Tr}\left(\boldsymbol{L}_{-S_k}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right) \\
\le&\left(1 - \frac{1}{k}\right)^{k-1}\left(\operatorname{Tr}\left(\boldsymbol{L}_{-S_1}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right)\right) \\
\le&\frac{k}{k-1} \cdot \frac{1}{e}\left(\operatorname{Tr}\left(\boldsymbol{L}_{-S_1}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right)\right),
\end{aligned}$$

which coupled with $\operatorname{Tr}\left(\boldsymbol{L}_{-S_1}^{-1}\right) = \operatorname{Tr}\left(\boldsymbol{L}_{-u^*}^{-1}\right)$ finishes the proof. □

7 A RANDOMIZED GREEDY ALGORITHM

The deterministic greedy algorithm ExactGreedy has a time complexity $O(n^3)$, which is still not acceptable for large networks. In this section, we provide an efficient randomized algorithm, which achieves a $(1 - \frac{k}{k-1} \cdot \frac{1}{e} - \epsilon)$ approximation factor in time $\widetilde{O}(km\epsilon^{-2})$.

To further accelerate algorithm ExactGreedy we need to compute the marginal gains

$$\operatorname{Tr}\left(\boldsymbol{L}_{-S_i}^{-1}\right) - \operatorname{Tr}\left(\boldsymbol{L}_{-(S_i+u)}^{-1}\right) = \frac{\boldsymbol{e}_u^T \boldsymbol{L}_{-S_i}^{-2} \boldsymbol{e}_u}{\boldsymbol{e}_u^T \boldsymbol{L}_{-S_i}^{-1} \boldsymbol{e}_u} \tag{8}$$

for all $u \in V$ and a vertex set $S_i \subseteq V$ more quickly. We also need a faster way to compute

$$\sum\nolimits_{v \in V} R_{\text{eff}}(u, v) = n\left(\boldsymbol{L}^{\dagger}\right)_{[u,u]} + \operatorname{Tr}\left(\boldsymbol{L}^{\dagger}\right) \tag{9}$$

for all $u \in V$ at the 1st step. We will show how to solve both problems in nearly linear time using Johnson-Lindenstrauss Lemma and Fast SDDM Solvers. Our routines are motivated by the effective resistance estimation routine in [33, 49].

Lemma 7.1 (Johnson-Lindenstrauss Lemma [29]). *Let $\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_n \in \mathbb{R}^d$ be fixed vectors and $0 < \epsilon < 1$ be a real number. Let q be a positive integer such that*

$$q \ge 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \ln n$$

and $\boldsymbol{Q}_{q \times d}$ be a random matrix obtained by first choosing each of its entries from Gaussian distribution $N(0, 1)$ independently and then normalizing each of its columns to a length of 1. With high probability, the following statement holds for any $1 \le i, j \le n$:

$$(1 - \epsilon)\left\|\boldsymbol{v}_i - \boldsymbol{v}_j\right\|^2 \le \left\|\boldsymbol{Q}\boldsymbol{v}_i - \boldsymbol{Q}\boldsymbol{v}_j\right\|^2 \le (1 + \epsilon)\left\|\boldsymbol{v}_i - \boldsymbol{v}_j\right\|^2.$$

Lemma 7.2 (Fast SDDM Solvers [16, 48]). *There is a routine $\boldsymbol{x} = \textsc{Solve}(\boldsymbol{S}, \boldsymbol{b}, \epsilon)$ which takes a Laplacian or an SDDM matrix $\boldsymbol{S}_{n \times n}$ with m nonzero entries, a vector $\boldsymbol{b} \in \mathbb{R}^n$, and an error parameter $\delta > 0$, and returns a vector $\boldsymbol{x} \in \mathbb{R}^n$ such that*

$$\left\|\boldsymbol{x} - \boldsymbol{S}^{-1}\boldsymbol{b}\right\|_{\boldsymbol{S}} \le \delta\left\|\boldsymbol{S}^{-1}\boldsymbol{b}\right\|_{\boldsymbol{S}}$$

holds with high probability, where $\|\boldsymbol{x}\|_{\boldsymbol{S}} \overset{\text{def}}{=} \sqrt{\boldsymbol{x}^T \boldsymbol{S}\boldsymbol{x}}$, and $\boldsymbol{S}^{-1}$ denotes the pseudoinverse of $\boldsymbol{S}$ when $\boldsymbol{S}$ is a Laplacian. The routine runs in expected time $\widetilde{O}(m \log(1/\delta))$.

7.1 Approximation of (9)

To approximate (9) we need to approximate all diagonal entries of $\boldsymbol{L}^\dagger$, i.e., $\boldsymbol{e}_u^T\boldsymbol{L}^\dagger\boldsymbol{e}_u$ for all $u \in V$. We first write $\boldsymbol{e}_u^T\boldsymbol{L}^\dagger\boldsymbol{e}_u$ in an Euclidean norm as

$$\begin{aligned}\boldsymbol{e}_u^T\boldsymbol{L}^\dagger\boldsymbol{e}_u &= \boldsymbol{e}_u^T\boldsymbol{L}^\dagger\boldsymbol{L}\boldsymbol{L}^\dagger\boldsymbol{e}_u = \boldsymbol{e}_u^T\boldsymbol{L}^\dagger\boldsymbol{B}^T\boldsymbol{W}\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_u\\ &= \boldsymbol{e}_u^T\boldsymbol{L}^\dagger\boldsymbol{B}^T\boldsymbol{W}^{1/2}\boldsymbol{W}^{1/2}\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_u = \left\|\boldsymbol{W}^{1/2}\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_u\right\|^2.\end{aligned}$$

Then we use Johnson-Lindenstrauss Lemma to reduce the dimensions. Let $\boldsymbol{Q}_{q\times m}$ be a random Gaussian matrix where $q = \left\lceil 4(\epsilon^2/2-\epsilon^3/3)^{-1}\ln n\right\rceil$. By Lemma 7.1,

$$\left\|\boldsymbol{W}^{1/2}\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_u\right\|^2 \approx_{1+\epsilon} \left\|\boldsymbol{Q}\boldsymbol{W}^{1/2}\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_u\right\|^2$$

holds for all $u \in V$ with high probability. Here we can use sparse matrix multiplication to compute $\boldsymbol{Q}\boldsymbol{W}^{1/2}\boldsymbol{B}$, and Fast SDDM solvers to compute $\boldsymbol{Q}\boldsymbol{W}^{1/2}\boldsymbol{B}\boldsymbol{L}^\dagger$.

We give the routine for approximating (9) as follows.

$\{r_u\}_{u\in V} = \textsc{ERSumsEst}\,(G, \boldsymbol{L}, \epsilon)$

(1) Set $\delta = \frac{\epsilon}{9n^2}\left(\frac{(1-\epsilon/3)w_{\min}}{(1+\epsilon/3)w_{\max}}\right)^{1/2}$.
(2) Generate a random Gaussian matrix $\boldsymbol{Q}_{q\times m}$ where $q = \left\lceil 4((\epsilon/3)^2/2-(\epsilon/3)^3/3)^{-1}\ln n\right\rceil$.
(3) Compute $\left(\boldsymbol{Q}\boldsymbol{W}^{1/2}\boldsymbol{B}\right)_{q\times n}$ by sparse matrix multiplication in $O(qm)$ time.
(4) Compute an approximation $\widetilde{\boldsymbol{Z}}_{q\times n}$ to $\boldsymbol{Z}_{q\times n} \stackrel{\text{def}}{=} \boldsymbol{Q}\boldsymbol{W}^{1/2}\boldsymbol{B}\boldsymbol{L}^\dagger$ by
$$\widetilde{\boldsymbol{Z}}_{[i,:]} \leftarrow \textsc{Solve}(\boldsymbol{L}, \left(\boldsymbol{Q}\boldsymbol{W}^{1/2}\boldsymbol{B}\right)_{[i,:]}^T, \delta)^T$$
where $1 \le i \le q$.
(5) $r_u \leftarrow \left\|\widetilde{\boldsymbol{Z}}\right\|_F^2 + n\left\|\widetilde{\boldsymbol{Z}}\boldsymbol{e}_u\right\|^2$ for all $u \in V$ and return $\{r_u\}_{u\in V}$.

The performance of ERSumsEst is characterized in the following lemma.

Lemma 7.3. *The routine ERSumsEst runs in time $\widetilde{O}(m)$. For $0 < \epsilon \le 1/2$, the $\{r_u\}_{u\in V}$ returned by ERSumsEst satisfies*

$$(1-\epsilon)\sum_{v\in V} R_{\text{eff}}(u,v) \le r_u \le (1+\epsilon)\sum_{v\in V} R_{\text{eff}}(u,v)$$

with high probability.

7.2 Approximation of (8)

We first approximate the numerator of (8), which can be recast in an euclidian norm as $\boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-2}\boldsymbol{e}_u = \left\|\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\right\|^2$. We then once again use Johnson-Lindenstrauss Lemma to reduce the dimensions. Let $\boldsymbol{P}_{p\times n}$ be a random Gaussian matrix where $p = \left\lceil 4(\epsilon^2/2-\epsilon^3/3)^{-1}\ln n\right\rceil$. By Lemma 7.1, we have that for all $u \in V$,

$$(1-\epsilon)\left\|\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\right\|^2 \le \left\|\boldsymbol{P}\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\right\|^2 \le (1+\epsilon)\left\|\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\right\|^2,$$

where $\boldsymbol{P}\boldsymbol{L}_{-S_i}^{-1}$ can be computed using Fast SDDM Solvers.

We continue to approximate the denominator of (8). Since $\boldsymbol{L}_{-S_i}$ is an SDDM matrix, we can express it in terms of the sum of a Laplacian and a nonnegative diagonal matrix as $\boldsymbol{L}_{-S_i} = \boldsymbol{B}'^T\boldsymbol{W}'\boldsymbol{B}' + \boldsymbol{X}$. Then we can write

$$\begin{aligned}&\boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u = \boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{L}_{-S_i}\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\\ =&\boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-1}\left(\boldsymbol{B}'^T\boldsymbol{W}'\boldsymbol{B}' + \boldsymbol{X}\right)\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\\ =&\boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{B}'^T\boldsymbol{W}'\boldsymbol{B}'\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u + \boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{X}\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\\ =&\left\|\boldsymbol{W}'^{1/2}\boldsymbol{B}'\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\right\|^2 + \left\|\boldsymbol{X}^{1/2}\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\right\|^2. \qquad (10)\end{aligned}$$

Let $\boldsymbol{Q}_{q\times m}$ and $\boldsymbol{R}_{r\times n}$ be random Gaussian matrices where

$$q = r = \left\lceil 4(\epsilon^2/2-\epsilon^3/3)^{-1}\ln n\right\rceil.$$

By Lemma 7.1, we have for all $u \in V$

$$(10) \approx_{1+\epsilon} \left\|\boldsymbol{Q}\boldsymbol{W}'^{1/2}\boldsymbol{B}'\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\right\|_2^2 + \left\|\boldsymbol{R}\boldsymbol{X}^{1/2}\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u\right\|_2^2.$$

Here $\boldsymbol{Q}\boldsymbol{W}'^{1/2}\boldsymbol{B}'\boldsymbol{L}_{-S_i}^{-1}$ and $\boldsymbol{R}\boldsymbol{X}^{1/2}\boldsymbol{L}_{-S_i}^{-1}$ can be computed using Fast SDDM Solvers.

We give the routine for approximating (8) as follows.

$\{g_u\}_{u\in V} = \textsc{GainsEst}\,(G, \boldsymbol{L}, S_i, \epsilon)$

(1) Set $\delta_1 = \frac{w_{\min}\epsilon}{27w_{\max}n^4}\left(\frac{(1-\epsilon/9)}{(1+\epsilon/9)n}\right)^{1/2}$ and $\delta_2 = \delta_3 = \frac{1}{4n^4w_{\max}}\left(\frac{\epsilon w_{\min}^{3/2}}{9n}\right)^{1/2}$.
(2) Generate random Gaussian matrices $\boldsymbol{P}_{p\times n}, \boldsymbol{Q}_{q\times m}, \boldsymbol{R}_{r\times n}$ where
$$p, q, r = \left\lceil 4((\epsilon/9)^2/2-(\epsilon/9)^3/3)^{-1}\ln n\right\rceil.$$
(3) Let $\boldsymbol{X} = \text{Diag}\left(\boldsymbol{L}_{-S_i}\boldsymbol{1}\right)$ and denote $\boldsymbol{L}_{-S_i} - \boldsymbol{X}$ by $\boldsymbol{B}'^T\boldsymbol{W}'\boldsymbol{B}'$.
(4) Compute $\boldsymbol{Q}\boldsymbol{W}'^{1/2}\boldsymbol{B}'$ and $\boldsymbol{R}\boldsymbol{X}^{1/2}$ by sparse matrix multiplication in $O(pm)$ time.
(5) Compute approximations $\widetilde{\boldsymbol{Z}}^{(1)}$ to $\boldsymbol{Z}^{(1)} \stackrel{\text{def}}{=} \boldsymbol{P}\boldsymbol{L}_{-S_i}^{-1}$, $\widetilde{\boldsymbol{Z}}^{(2)}$ to $\boldsymbol{Z}^{(2)} \stackrel{\text{def}}{=} \boldsymbol{Q}\boldsymbol{W}'^{1/2}\boldsymbol{B}'\boldsymbol{L}_{-S_i}^{-1}$, and $\widetilde{\boldsymbol{Z}}^{(3)}$ to $\boldsymbol{Z}^{(3)} \stackrel{\text{def}}{=} \boldsymbol{R}\boldsymbol{X}^{1/2}\boldsymbol{L}_{-S_i}^{-1}$ by
(a) $\widetilde{\boldsymbol{Z}}^{(1)}_{[i,:]} \leftarrow \textsc{Solve}(\boldsymbol{L}_{-S_i}, \boldsymbol{P}^T_{[i,:]}, \delta_1)^T$,
(b) $\widetilde{\boldsymbol{Z}}^{(2)}_{[i,:]} \leftarrow \textsc{Solve}(\boldsymbol{L}_{-S_i}, \left(\boldsymbol{Q}\boldsymbol{W}'^{1/2}\boldsymbol{B}'\right)^T_{[i,:]}, \delta_2)^T$, and
(c) $\widetilde{\boldsymbol{Z}}^{(3)}_{[i,:]} \leftarrow \textsc{Solve}(\boldsymbol{L}_{-S_i}, \left(\boldsymbol{R}\boldsymbol{X}^{1/2}\right)^T_{[i,:]}, \delta_3)^T$.
(6) $g_u \leftarrow \frac{\left\|\widetilde{\boldsymbol{Z}}^{(1)}\boldsymbol{e}_u\right\|^2}{\left\|\widetilde{\boldsymbol{Z}}^{(2)}\boldsymbol{e}_u\right\|^2 + \left\|\widetilde{\boldsymbol{Z}}^{(3)}\boldsymbol{e}_u\right\|^2}$ for all $u \in V$ and return $\{g_u\}_{u\in V}$.

Lemma 7.4. *The routine GainsEst runs in time $\widetilde{O}(m)$. For $0 \le \epsilon \le 1/2$, the $\{g_u\}_{u\in V}$ returned by GainsEst satisfies*

$$(1-\epsilon)\frac{\boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-2}\boldsymbol{e}_u}{\boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u} \le g_u \le (1+\epsilon)\frac{\boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-2}\boldsymbol{e}_u}{\boldsymbol{e}_u^T\boldsymbol{L}_{-S_i}^{-1}\boldsymbol{e}_u}$$

with high probability.

We now give the $\widetilde{O}(mk)$-time greedy algorithm as follows.

S_k = ApproxGreedy $(G, \boldsymbol{L}, k, \epsilon)$

(1) $\{r_u\}_{u\in V} \leftarrow$ ERSumsEst $(G, \boldsymbol{L}, \epsilon/3)$
(2) $S_1 \leftarrow \{u_1\}$ where $u_1 = \arg\min_{u\in V} r_u$.
(3) Repeat the following steps for $i = 1, \ldots, k-1$:
  (a) $\{g_u\}_{u\in V} \leftarrow$ GainsEst $(G, \boldsymbol{L}, S_i, \epsilon/2)$
  (b) $S_{i+1} \leftarrow S_i + u_{i+1}$ where
$$u_{i+1} = \arg\max_{u\in(V\setminus S_i)} g_u.$$
(4) Return S_k.

The performance of ApproxGreedy is characterized in the following theorem.

Theorem 7.5. *The algorithm $S_k =$ ApproxGreedy$(G, \boldsymbol{L}, k, \epsilon)$ takes an undirected positively weighted graph $G = (V, E, w)$ with associated Laplacian $\boldsymbol{L}$, an integer $2 \le k \le n$, and an error parameter $0 < \epsilon \le 1/2$, and returns a vertex set $S_k \subseteq V$ with $|S_k| = k$. The algorithm runs in time $\widetilde{O}(km)$. With high probability, the vertex set S_k satisfies*

$$(1+\epsilon)\mathrm{Tr}\left(\boldsymbol{L}_{-u^*}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S_k}^{-1}\right) \ge \left(1 - \frac{k}{k-1}\cdot\frac{1}{e} - \epsilon\right)\left((1+\epsilon)\mathrm{Tr}\left(\boldsymbol{L}_{-u^*}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right)\right),$$

where

$$S^* \stackrel{\text{def}}{=} \arg\min_{|S|\le k} \mathrm{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right), \quad u^* \stackrel{\text{def}}{=} \arg\min_{u\in V} \sum_{v\in V} R_{\text{eff}}(u, v).$$

Proof. The running time is easy to verify. It follows by the running times in Lemma 7.3 and 7.4.

We now prove the approximation ratio. The main difference between ApproxGreedy and ExactGreedy is that at each step, ExactGreedy picks a vertex with maximum marginal gain, while ApproxGreedy picks a vertex with at least $\frac{1-\epsilon/2}{1+\epsilon/2} \ge 1-\epsilon$ times maximum marginal gain. By supermodularity we have for any $i \ge 1$

$$\mathrm{Tr}\left(\boldsymbol{L}_{-S_i}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S_{i+1}}^{-1}\right) \ge \frac{1-\epsilon}{k}\left(\mathrm{Tr}\left(\boldsymbol{L}_{-S_i}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S*}^{-1}\right)\right),$$

which implies

$$\mathrm{Tr}\left(\boldsymbol{L}_{-S_{i+1}}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right) \le \left(1 - \frac{1-\epsilon}{k}\right)\left(\mathrm{Tr}\left(\boldsymbol{L}_{-S_i}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right)\right).$$

Then, we have

$$\begin{aligned}
&\mathrm{Tr}\left(\boldsymbol{L}_{-S_k}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right) \\
&\le \left(1 - \frac{1-\epsilon}{k}\right)^{k-1}\left(\mathrm{Tr}\left(\boldsymbol{L}_{-S_1}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right)\right) \\
&\le \frac{k}{k-1}\cdot\frac{1}{e^{(1-\epsilon)}}\left(\mathrm{Tr}\left(\boldsymbol{L}_{-S_1}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right)\right) \\
&\le \frac{k}{k-1}\cdot\left(\frac{1}{e} + \epsilon\right)\left(\mathrm{Tr}\left(\boldsymbol{L}_{-S_1}^{-1}\right) - \mathrm{Tr}\left(\boldsymbol{L}_{-S^*}^{-1}\right)\right).
\end{aligned}$$

By Lemma 7.3, we obtain

$$\mathrm{Tr}\left(\boldsymbol{L}_{-S_1}^{-1}\right) \le \frac{1+\epsilon/3}{1-\epsilon/3}\mathrm{Tr}\left(\boldsymbol{L}_{-u^*}^{-1}\right) \le (1+\epsilon)\,\mathrm{Tr}\left(\boldsymbol{L}_{-u^*}^{-1}\right),$$

which plus the above inequality finishes the proof. □

8 EXPERIMENTS

In this section, we study the performance of our algorithms by conducting experiments on some classic network models and real-world networks taken from KONECT [34] and SNAP [36]. We run our experiments on the largest components of these networks. Related information of these networks is shown in Table 1, where networks are shown in increasing order of their numbers of vertices.

Table 1: Information of datasets. For a network with n vertices and m edges, we use n' and m' to denote the number of vertices and edges in its largest connected component, respectively.

Network	n	m	n'	m'
Zachary karate club	34	78	34	78
Windsufers	43	336	43	336
Contiguous USA	49	107	49	107
Barabási-Albert	50	94	50	94
Watts-Strogatz	50	100	50	100
Erdös-Rényi	50	95	50	95
Regular ring lattice	50	100	50	100
Dolphins	62	159	62	159
David Copperfield	112	425	112	425
Jazz musicians	198	2742	195	1814
Virgili	1,133	5,451	1,133	5,451
Euroroad	1,174	1,417	1,039	1,305
Protein	1,870	2,277	1,458	1,948
Hamster full	2,426	16,631	2,000	16,098
ego-Facebook	2,888	2,981	2,888	2,981
Vidal	3,133	6,726	2,783	6,007
Powergrid	4,941	6,594	4,941	6,594
Reactome	6,327	147,547	5,973	145,778
ca-HepTh	9,877	25,998	8,638	24,806
PG-Privacy	10,680	24,316	10,680	24,316
CAIDA	26,475	53,381	26,475	53,381
ego-Twitter	81,306	1,342,296	81,306	1,342,296
com-DBLP	317,080	1,049,866	317,080	1,049,866
roadNet-PA	1,087,562	1,541,514	1,087,562	1,541,514
com-Youtube	1,134,890	2,987,624	1,134,890	2,987,624
roadNet-TX	1,379,917	1,921,660	1,351,137	1,879,201
roadNet-CA	1,965,206	2,766,607	1,957,027	2,760,388

We implement our algorithms in Julia to facilitate interactions with the SDDM solver contained in the Laplacian.jl package[2]. All of our experiments were run on a Linux box with 4.2 GHz Intel i7-7700 CPU and 32G memory, using a single thread.

8.1 Accuracy of routine GainsEst

We first show the accuracy of routine GainsEst. We use quantity JLfactor to denote $q/\ln n$ in GainsEst. We run GainsEst with different JLfactors on network arXiv High Energy Physics - Theory collaboration network (ca-HepTh). And we set $S_i = \{u^*\}$ where

$$u^* \stackrel{\text{def}}{=} \arg\min_{u\in V} \sum\nolimits_{v\in V} R_{\text{eff}}(u, v).$$

[2]https://github.com/danspielman/Laplacians.jl

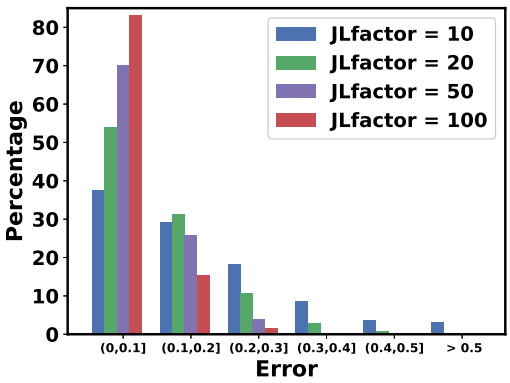


Figure 1: Error distribution for routine GainsEst with different JLfactors on network ca-HepTh.

For each JLfactor, we compute the relative error of gain of each vertex $u \neq u^*$, given by

$$\frac{\left|g_u - \left(\boldsymbol{e}_u^T \boldsymbol{L}_{-u^*}^{-2} \boldsymbol{e}_u\right) / \left(\boldsymbol{e}_u^T \boldsymbol{L}_{-u^*}^{-1} \boldsymbol{e}_u\right)\right|}{\left(\boldsymbol{e}_u^T \boldsymbol{L}_{-u^*}^{-2} \boldsymbol{e}_u\right) / \left(\boldsymbol{e}_u^T \boldsymbol{L}_{-u^*}^{-1} \boldsymbol{e}_u\right)}.$$

We then draw the distribution of relative errors with different JLfactors in a histogram. The results are shown in Figure 1. We observe that the errors become small when JLfactor gets larger. Moreover, almost all errors are in the range $(0, 0.5]$ when JLfactor ≥ 20.

We set JLfactor $= 20$ in all other experiments. We will show that this is sufficient for our greedy algorithm to obtain good solutions empirically.

8.2 Effectiveness of Greedy Algorithms

We show the effectiveness of our algorithms by comparing the results of our algorithms with the optimum solutions on four small model networks (Barabási-Albert (BA) [2], Watts-Strogatz (WS) [52], Erdös-Rényi (ER) [20], and a regular ring lattice [52]) and four small realistic networks (Dolphins, Contiguous USA, Zachary karate club, and Windsurfers). We are able to compute the optimum solutions on these networks because of their small sizes.

For each $k = 1, 2, \ldots, 6$, we first find the set of k vertices with the optimum current flow closeness by brute-force search. We then compute the current flow closeness of the vertex sets returned by ExactGreedy and ApproxGreedy. Also, we compute the solutions returned by the random scheme, which chose k vertices uniformly at random. The results are shown in Figure 2 and 3. We observe that the current flow closenesses of the solutions returned by our two greedy algorithms and the optimum solution are almost the same. This means that the approximation ratios of our greedy algorithms are significantly better than their theoretical guarantees. Moreover, the solutions returned by our greedy algorithms are much better than those returned by the random scheme.

We then demonstrate the effectiveness of our algorithms further by comparing it to the random scheme, as well as two other schemes, Top-degree and Top-cent, on six larger networks. Here the Top-degree scheme chooses k vertices with highest degrees, while the Top-cent scheme chooses k vertices with highest current flow closeness centrality according to Definition 2.4. A comparison of results for these five algorithms is shown in Figure 4. We observe

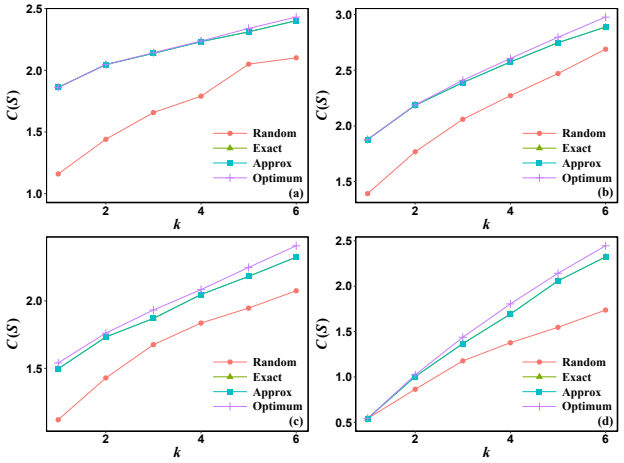


Figure 2: Current flow closeness of vertex sets returned by ExactGreedy, ApproxGreedy, random and optimum strategies on four models: BA (a), WS (b), ER (c), and a regular ring lattice (d).

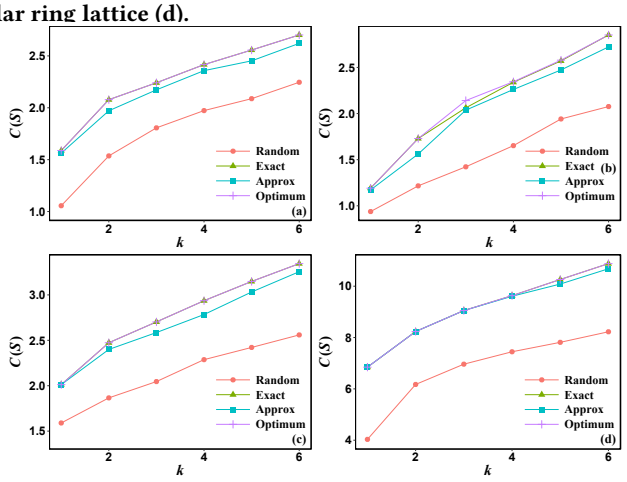


Figure 3: Current flow closeness of vertex sets returned by ExactGreedy, ApproxGreedy, random and optimum strategies on four networks: Dolphins (a), Contiguous USA (b), Zachary karate club (c), and Windsurfers (d).

that our two greedy algorithms obtain similar approximation ratios, and both outperform the other three schemes (random, Top-degree, and Top-cent).

8.3 Efficiency of Greedy Algorithms

We now show that algorithm ApproxGreedy runs much faster than ExactGreedy, especially on large-scale networks. We run our two greedy algorithms on a larger set of real-world networks. For each network, we use both greedy algorithms to choose $k = 10$ vertices, and then compare their running times as well as the solutions they returned. For networks with more than 30000 vertices, we compute the current flow closeness of the vertex set returned by ApproxGreedy by combining Fast Laplacian solvers and Hutchinson's trace estimation [1, 26]. We list the results in Table 2. We observe that the ratio of the running time of ExactGreedy to

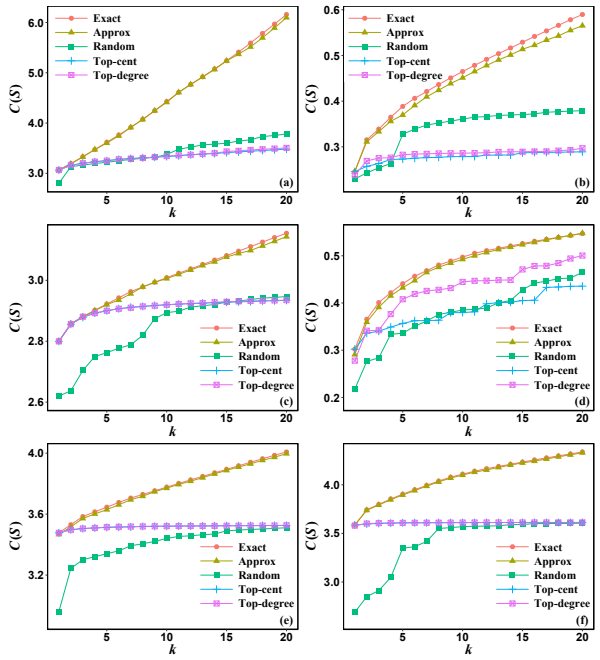


Figure 4: Current flow closeness of vertex sets returned versus the number k of vertices chosen for the five algorithms on David Copperfield (a), Euroroad (b), U. Rovira i Virgili (c), US power grid (d), Hamsterster full (e), and Reactome (f).

Table 2: The average running times and results of ExactGreedy (Exact) and ApproxGreedy (Approx) algorithms on a larger set of real-world networks, as well as the ratios of running times of ExactGreedy to those of ApproxGreedy, and ratios of current-flow closenesses achieved by ApproxGreedy to those achieved by ExactGreedy.

Network	Time (seconds)			Current flow closeness		
	Exact	Approx	Ratio	Exact	Approx	Ratio
Protein	1.35	1.34	1.01	0.7451	0.7388	0.9915
Vidal	8.04	3.40	2.36	1.3179	1.3138	0.9968
ego-Facebook	13.26	2.56	5.17	1.0195	1.0195	1.00
ca-Hepth	196.67	15.80	12.44	1.5340	1.5267	0.9952
PG-Privacy	367.71	17.25	21.31	0.7155	0.7141	0.9980
CAIDA	5530.32	30.64	464.96	1.3948	1.3945	0.9997
ego-Twitter	-	562.27	-	-	5.5052	-
com-DBLP	-	1700.13	-	-	1.6807	-
com-Youtube	-	6374.02	-	-	1.0382	-
roadNet-PA	-	9802.54	-	-	0.3089	-
roadNet-TX	-	13671.05	-	-	0.2260	-
roadNet-CA	-	22853.20	-	-	0.2630	-

that of ApproxGreedy increases rapidly when the network size becomes larger, while the vertex sets they returned have approximately the same current flow closeness. Moreover, ApproxGreedy is able to solve current flow closeness maximization for networks with more than 10^6 vertices in a few hours, whereas ExactGreedy fails because of its high time and space complexity.

9 CONCLUSIONS

In this paper, we extended the notion of current flow closeness centrality (CFCC) to a group of vertices. For a vertex group S in an n-vertex graph with m edges, its CFCC $C(S)$ equals the ratio of n to the sum of effective resistances between S and all other vertices. We then considered the problem of finding the set S^* of k vertices with an aim to maximize $C(S^*)$, and solved it by considering an equivalent problem of minimizing $\mathrm{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right)/n$. We showed that the problem is NP-hard, and proved that the objective function is monotone and supermodular. We devised two approximation algorithms for minimizing $\mathrm{Tr}\left(\boldsymbol{L}_{-S}^{-1}\right)/n$ by iteratively selected k vertices in a greedy way. The first one achieves a $(1-\frac{k}{k-1}\cdot\frac{1}{e})$ approximation ratio in time $O(n^3)$; while the second one obtains a $(1-\frac{k}{k-1}\cdot\frac{1}{e}-\epsilon)$ approximation factor in time $\widetilde{O}(mk\epsilon^{-2})$. We conducted extensive experiments on model and realistic networks, the results of which show that both algorithms can often give almost optimal solutions. In particular, our second algorithm is able to scale to huge networks, and quickly gives good approximate solutions in networks with more than 10^6 vertices. In future works, we plan to introduce and study the betweenness group centrality based on current flow [43], which takes into account all possible paths.

ACKNOWLEDGMENTS

This work was supported in part by the National Key R & D Program of China (No. 2018YFB1305104), the National Natural Science Foundation of China (No. 61872093), the National Science Foundation (No. 1637566), and Shanghai Municipal Science and Technology Major Project (No. 2018SHZDZX01) and ZJLab.

REFERENCES

[1] Haim Avron and Sivan Toledo. 2011. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *J. ACM* 58, 2 (2011), 8:1–8:34.
[2] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512.
[3] Alex Bavelas. 1948. A mathematical model for group structures. *Hum. Oran.* 7, 3 (1948), 16–30.
[4] Alex Bavelas. 1950. Communication patterns in task-oriented groups. *J. Acoust. Soc. Am.* 22, 6 (1950), 725–730.
[5] Michele Benzi and Christine Klymko. 2015. On the limiting behavior of parameter-dependent network centrality measures. *SIAM J. Matrix Anal. Appl.* 36, 2 (2015), 686–706.
[6] Elisabetta Bergamini, Tanya Gonser, and Henning Meyerhenke. 2018. Scaling up Group Closeness Maximization. In *Proceedings of the Twentieth Workshop on Algorithm Engineering and Experiments (ALENEX '18)*. SIAM, 209–222.
[7] Elisabetta Bergamini, Michael Wegner, Dimitar Lukarski, and Henning Meyerhenke. 2016. Estimating Current-Flow Closeness Centrality with a Multigrid Laplacian Solver. In *Proceedings of the 7th SIAM Workshop on Combinatorial Scientific Computing (CSC '16)*. SIAM, 1–12.
[8] Paolo Boldi and Sebastiano Vigna. 2014. Axioms for centrality. *Internet Math.* 10, 3-4 (2014), 222–262.
[9] Francesco Bonchi, Gianmarco De Francisci Morales, and Matteo Riondato. 2016. Centrality measures on big graphs: Exact, approximated, and distributed algorithms. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. ACM, 1017–1020.
[10] Enrico Bozzo and Massimo Franceschet. 2013. Resistance distance, closeness, and betweenness. *Soc. Networks* 35, 3 (2013), 460–469.
[11] Ulrik Brandes and Daniel Fleischer. 2005. Centrality Measures Based on Current Flow. In *22nd Annual Symposium on Theoretical Aspects of Computer Science, Proceedings (STACS '05)*. Springer, 533–544.
[12] Chen Chen, Hanghang Tong, B Aditya Prakash, Charalampos E Tsourakakis, Tina Eliassi-Rad, Christos Faloutsos, and Duen Horng Chau. 2016. Node immunization on large graphs: Theory and algorithms. *IEEE Trans. Knowl. Data Eng.* 28, 1 (2016), 113–126.

[13] Chen Chen, Wei Wang, and Xiaoyang Wang. 2016. Efficient maximum closeness centrality group identification. In *Databases Theory and Applications - 27th Australasian Database Conference, Proceedings (ADC '16)*. Springer, 43–55.

[14] Andrew Clark, Qiqiang Hou, Linda Bushnell, and Radha Poovendran. 2017. A submodular optimization approach to leader-follower consensus in networks with negative edges. In *Proceedings of American Control Conference (ACC '17)*. IEEE, 1346–1352.

[15] Andrew Clark and Radha Poovendran. 2011. A submodular optimization framework for leader selection in linear multi-agent systems. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC '11)*. IEEE, 3614–3621.

[16] Michael B. Cohen, Rasmus Kyng, Gary L. Miller, Jakub W. Pachocki, Richard Peng, Anup Rao, and Shen Chen Xu. 2014. Solving SDD linear systems in nearly $m \log^{1/2} n$ time. In *Proceedings of the 46th annual ACM Symposium on Theory of Computing (STOC '14)*. ACM, 343–352.

[17] Samuel I. Daitch and Daniel A. Spielman. 2008. Faster approximate lossy generalized flow via interior point algorithms. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC '08)*. ACM, 451–460.

[18] Shlomi Dolev, Yuval Elovici, Rami Puzis, and Polina Zilberman. 2009. Incremental deployment of network monitors based on group betweenness centrality. *Inform. Process. Lett.* 109, 20 (2009), 1172–1176.

[19] Peter G Doyle and J Laurie Snell. 1984. *Random Walks and Electric Networks*. Mathematical Association of America.

[20] Paul Erdös and Alfréd Rényi. 1959. On random graphs, I. *Publ. Math. Debrecen* 6 (1959), 290–297.

[21] Martin G Everett and Stephen P Borgatti. 1999. The centrality of groups and classes. *J. Math. Sociol.* 23, 3 (1999), 181–201.

[22] Martin Fink and Joachim Spoerhase. 2011. Maximum betweenness centrality: approximability and tractable cases. In *Algorithms and Computation - 5th International Workshop, Proceedings (WALCOM '11)* . Springer, 9–20.

[23] Gerd Fricke, Stephen T. Hedetniemi, and David Pokrass Jacobs. 1998. Independence and Irredundance in k-Regular Graphs. *Ars Comb.* 49 (1998).

[24] Rumi Ghosh, Shang-hua Teng, Kristina Lerman, and Xiaoran Yan. 2014. The interplay between dynamics and networks: centrality, communities, and cheeger inequality. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, 1406–1415.

[25] Christos Gkantsidis, Milena Mihail, and Amin Saberi. 2006. Random walks in peer-to-peer networks: algorithms and evaluation. *Perform. Evaluation* 63, 3 (2006), 241–263.

[26] MF Hutchinson. 1989. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Commun. Stat. Simul. Comput.* 18, 3 (1989), 1059–1076.

[27] N Sh Izmailian, R Kenna, and FY Wu. 2013. The two-point resistance of a resistor network: a new formulation and application to the cobweb network. *J. Phys. A* 47, 3 (2013), 035003.

[28] Donald B. Johnson. 1977. Efficient algorithms for shortest paths in sparse networks. *J. ACM* (1977).

[29] William Johnson and Joram Lindenstrauss. 1984. Extensions of Lipschitz mappings into a Hilbert space. *Contemp. Math.* 26, 189-206 (1984), 1.

[30] Jonathan A. Kelner, Gary L. Miller, and Richard Peng. 2012. Faster approximate multicommodity flow using quadratically coupled flows. In *Proceedings of the 44th Symposium on Theory of Computing Conference (STOC 2012)*. ACM, 1–18.

[31] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*. ACM, 137–146.

[32] Douglas J Klein and Milan Randić. 1993. Resistance distance. *J. Math. Chem.* 12, 1 (1993), 81–95.

[33] Ioannis Koutis, Alex Levin, and Richard Peng. 2016. Faster Spectral Sparsification and Numerical Algorithms for SDD Matrices. *ACM Trans. Algorithms* 12, 2 (2016), 17:1–17:16.

[34] Jérôme Kunegis. 2013. Konect: the koblenz network collection. In *22nd International World Wide Web Conference, Companion Volume (WWW '13)* . ACM, 1343–1350.

[35] Amy N Langville and Carl D Meyer. 2012. *Who's# 1?: the science of rating and ranking*. Princeton University Press.

[36] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data. (June 2014).

[37] Rong-Hua Li, Jeffrey Xu Yu, Xin Huang, and Hong Cheng. 2014. Random-walk domination in large graphs. In *Proceedings of IEEE 30th International Conference on Data Engineering (ICDE '14)*. IEEE, 736–747.

[38] Linyuan Lü, Duanbing Chen, Xiao-Long Ren, Qian-Ming Zhang, Yi-Cheng Zhang, and Tao Zhou. 2016. Vital nodes identification in complex networks. *Phys. Rep.* 650 (2016), 1–63.

[39] Ahmad Mahmoody, Charalampos E Tsourakakis, and Eli Upfal. 2016. Scalable betweenness centrality maximization via sampling. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, 1765–1773.

[40] Charalampos Mavroforakis, Michael Mathioudakis, and Aristides Gionis. 2015. Absorbing random-walk centrality: Theory and algorithms. In *Proceedings of IEEE International Conference on Data Mining (ICDM '15)*. IEEE, 901–906.

[41] Gary L. Miller and Richard Peng. 2013. Approximate Maximum Flow on Separable Undirected Graphs. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '13)*. SIAM, 1151–1170.

[42] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions - I. *Math. Program.* 14, 1 (1978), 265–294.

[43] Mark E. J. Newman. 2005. A measure of betweenness centrality based on random walks. *Soc. Networks* 27, 1 (2005), 39–54.

[44] Mark E. J. Newman. 2010. *Networks: An Introduction*. Oxford University Press.

[45] Stacy Patterson and Bassam Bamieh. 2010. Leader selection for optimal network coherence. In *Proceedings of 49th IEEE Conference on Decision and Control (CDC '10)*. IEEE, 2692–2697.

[46] Mohammad Pirani and Shreyas Sundaram. 2014. Spectral properties of the grounded Laplacian matrix with applications to consensus in the presence of stubborn agents. In *Proceedings of American Control Conference (ACC '14)*. IEEE, 2160–2165.

[47] Mojtaba Rezvani, Weifa Liang, Wenzheng Xu, and Chengfei Liu. 2015. Identifying top-k structural hole spanners in large-scale social networks. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM '15)*. ACM, 263–272.

[48] Daniel A. Spielman and Shanghua Teng. 2014. Nearly Linear Time Algorithms for Preconditioning and Solving Symmetric, Diagonally Dominant Linear Systems. *SIAM J. Matrix Anal. Appl.* 35, 3 (2014), 835–885.

[49] Daniel A. Spielman and Nikhil Srivastava. 2011. Graph Sparsification by Effective Resistances. *SIAM J. Comput.* 40, 6 (2011), 1913–1926.

[50] Karen Stephenson and Marvin Zelen. 1989. Rethinking centrality: Methods and examples. *Soc. Networks* 11, 1 (1989), 1–37.

[51] Hanghang Tong, B Aditya Prakash, Charalampos Tsourakakis, Tina Eliassi-Rad, Christos Faloutsos, and Duen Horng Chau. 2010. On the vulnerability of large graphs. In *Proceedings of IEEE 10th International Conference on Data Mining (ICDM '10)*. IEEE, 1091–1096.

[52] Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of small-world networks. *Nature* 393, 6684 (1998), 440.

[53] Scott White and Padhraic Smyth. 2003. Algorithms for estimating relative importance in networks. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*. ACM, 266–275.

[54] Wenzheng Xu, Mojtaba Rezvani, Weifa Liang, Jeffrey Xu Yu, and Chengfei Liu. 2017. Efficient Algorithms for the Identification of Top-k Structural Hole Spanners in Large Social Networks. *IEEE Trans. Knowl. Data Eng.* 29, 5 (2017), 1017–1030.

[55] Yuichi Yoshida. 2014. Almost linear-time algorithms for adaptive betweenness centrality using hypergraph sketches. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, 1416–1425.

[56] Junzhou Zhao, John Lui, Don Towsley, and Xiaohong Guan. 2014. Measuring and maximizing group closeness centrality over disk-resident graphs. In *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*. ACM, 689–694.

[57] Pengpeng Zhao, Yongkun Li, Hong Xie, Zhiyong Wu, Yinlong Xu, and John CS Lui. 2017. Measuring and Maximizing Influence via Random Walk in Social Activity Networks. In *Database Systems for Advanced Applications - 22nd International Conference, Proceedings, Part II (DASFAA '17)* . Springer, 323–338.

Semi-Supervised Graph Classification: A Hierarchical Graph Perspective

Jia Li
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
Tencent AI Lab
lijia@se.cuhk.edu.hk

Yu Rong
Tencent AI Lab
Shenzhen
yu.rong@hotmail.com

Hong Cheng
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
hcheng@se.cuhk.edu.hk

Helen Meng
Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
hmmeng@se.cuhk.edu.hk

Wenbing Huang
Tencent AI Lab
Shenzhen
hwenbing@126.com

Junzhou Huang
Tencent AI Lab
Shenzhen
joehhuang@tencent.com

ABSTRACT

Node classification and graph classification are two graph learning problems that predict the class label of a node and the class label of a graph respectively. A node of a graph usually represents a real-world entity, e.g., a user in a social network, or a protein in a protein-protein interaction network. In this work, we consider a more challenging but practically useful setting, in which a node itself is a graph instance. This leads to a hierarchical graph perspective which arises in many domains such as social network, biological network and document collection. For example, in a social network, a group of people with shared interests forms a user group, whereas a number of user groups are interconnected via interactions or common members. We study the node classification problem in the hierarchical graph where a "node" is a graph instance, e.g., a user group in the above example. As labels are usually limited in real-world data, we design two novel semi-supervised solutions named SEmi-supervised grAph cLassification via Cautious/Active Iteration (or SEAL-C/AI in short). SEAL-C/AI adopt an iterative framework that takes turns to build or update two classifiers, one working at the graph instance level and the other at the hierarchical graph level. To simplify the representation of the hierarchical graph, we propose a novel supervised, self-attentive graph embedding method called SAGE, which embeds graph instances of arbitrary size into fixed-length vectors. Through experiments on synthetic data and Tencent QQ group data, we demonstrate that SEAL-C/AI not only outperform competing methods by a significant margin in terms of accuracy/Macro-F1, but also generate meaningful interpretations of the learned representations.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313461

CCS CONCEPTS

• **Mathematics of computing → Graph algorithms**; • **Information systems → Social networks**; • **Computing methodologies → Supervised learning by classification**.

KEYWORDS

hierarchical graph; graph embedding; semi-supervised learning; active learning

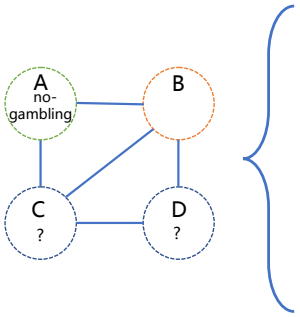
ACM Reference Format:
Jia Li, Yu Rong, Hong Cheng, Helen Meng, Wenbing Huang, and Junzhou Huang. 2019. Semi-Supervised Graph Classification: A Hierarchical Graph Perspective. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313461

1 INTRODUCTION

Graph has been widely used to model real-world entities and the relationship among them. Two graph learning problems have received a lot of attention recently, i.e., node classification and graph classification. Node classification is to predict the class label of nodes in a graph, for which many studies in the literature make use of the connections between nodes to boost the classification performance. For example, [25] enhances the recommendation precision in LinkedIn by taking advantage of the interaction network, and [27] improves the performance of document classification by exploiting the citation network. Graph classification, on the other hand, is to predict the class label of graphs, for which various graph kernels [3, 9, 29, 30] and deep learning approaches [22, 23] have been designed. In this work, we consider a more challenging but practically useful setting, in which a node itself is a graph instance. This leads to *a hierarchical graph in which a set of graph instances are interconnected via edges.* This is a very expressive data representation, as it considers the relationship between graph instances, rather than treating them independently. The hierarchical graph model applies to many real-world data, for example, a social network can be modeled as a hierarchical graph, in which a user group is represented by a graph instance and treated as a node in the

hierarchical graph, and then
nected via interactions or co
a document collection can b
which a document is regarde
set of documents are interco
In this paper, we study *grap*
which predicts the class labe
graph.

One challenge in this pr
a much too complicated inp
this challenge, we design a
embeds a graph instance of a
All graph instances in the h
embedding vectors which ar
cation. Specifically, the embe
classifier called IC from grap
vectors and predicted class
Another classifier HC at the
bedding vectors and their c
predicted class probabilities of all graph instances. To enforce a consistency between the two classifiers, we define a disagreement loss to measure the degree of divergence between the predictions by them and aim to minimize the disagreement loss.



Another challenge is that the amount of available class labels is usually very small in real-world data, which limits the classification performance. To address this challenge, we take a semi-supervised learning approach to solving the graph classification problem. We design an iterative algorithm framework which takes turns to build or update classifiers IC and HC. We start with the limited labeled training set and build IC, which produces the embedding vectors of graph instances. HC takes the embedding vectors as input and produces predictions. We cautiously select a subset of predicted labels by HC with high confidence to enlarge the training set. The enlarged training set is then fed into IC in the next iteration to update its parameters in the hope of generating more accurate embedding vectors and predictions. HC further takes the new embedding vectors for model update and class prediction. This is our proposed solution, called SEmi-supervised grAph cLassification via Cautious Iteration (SEAL-CI), to the graph classification problem.

We also extend this iterative algorithm to the active learning framework, in which we iteratively select the most informative instances for manual annotation, and then update the classifiers with the newly labeled instances in a similar process as described above. This method is called SEAL-AI in short.

Our contributions are summarized as follows.

- We study semi-supervised graph classification from a hierarchical graph perspective, which, to the best of our knowledge, has not been studied before. Our proposed solutions SEAL-C/AI achieve superior classification performance to the state-of-the-other graph kernel and deep learning methods, even when given very few labeled training instances.
- We design a novel supervised, self-attentive graph embedding method called SAGE to embed graphs of arbitrary size into fixed-length vectors, which are used as a common form of input for classification. The embedding approach not only simplifies the representation of a hierarchical graph greatly, but also provides meaningful interpretations of the underlying data in two forms: 1) embedding vectors of graph instances, and 2) node importance in a graph instance learned through a self-attentive mechanism that differentiates their contribution in classifying a graph instance.
- We evaluate SEAL-C/AI on both synthetic graphs and Tencent QQ group data. From the social networking platform Tencent QQ, we select 37,836 QQ groups with 18,422,331 unique anonymized users and classify them as "game" or "non-game" groups. SEAL-C/AI achieve a Macro-F1 score of 70.8% and 73.2% respectively with only 2.6% labeled instances. They both outperform other competing methods by a large margin.

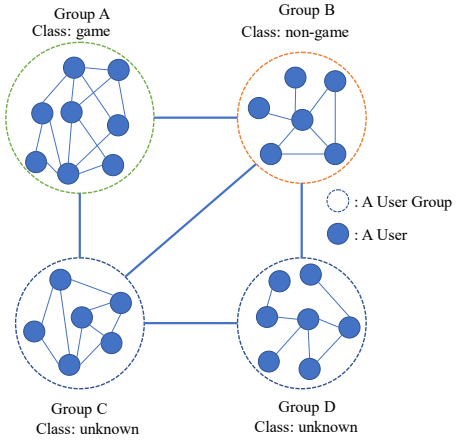


Figure 1: A hierarchical graph with four graph instances A, B, C, D, each of which corresponds to a user group in a social network.

The remainder of this paper is organized as follows. Section 2 gives the problem definition and Section 3 describes the design of SEAL-C/AI. We report the experimental results in Section 4 and discuss related work in Section 5. Finally, Section 6 concludes the paper.

2 PROBLEM DEFINITION

We denote a set of objects as $O = \{o_1, o_2, \ldots, o_N\}$ which represent real-world entities. We use ϕ attributes to describe properties of objects, e.g., age, gender, and other information of people.

We use a *graph instance* to model the relationship between objects in O, which is denoted as $g = (V, A, X)$, $V \subseteq O$ is the node set and $|V| = n$, A is an $n \times n$ adjacency matrix representing the connectivity in g, and $X \in \mathbb{R}^{n \times \phi}$ is a matrix recording the attribute values of all nodes in g.

A set of graph instances G can be interconnected, and the connectivity between the graph instances is represented by an adjacency matrix Θ. The graph instances and their connections are modeled as a *hierarchical graph*.

A graph instance $g \in G$ is a *labeled graph* if it has a class label, represented by a vector $y \in \{0, 1\}^c$, where c is the number of classes. A graph instance is *unlabeled* if its class label is unknown. Then G can be divided into two subsets: labeled graphs G_l and

unlabeled graphs G_u, where $G = G_l \cup G_u$, $|G_l| = L$ and $|G_u| = U$. In this paper, we study the problem of ***graph classification***, which determines the class label of the unlabeled graph instances in G_u from the available class labels in G_l and the hierarchical graph topological structure. As the amount of available class labels is usually very limited in real-world data, we take a semi-supervised learning approach to solving this problem.

Figure 1 depicts a hierarchical graph in the context of a social network. A, B, C, D denote four user groups. Group A has the class label of "game", B has the label of "non-game", while the class labels of C and D are unknown. These four groups are connected via some kind of relationships, e.g., interactions or common members. The internal structure of each user group shows the connections between individual users. From this hierarchical graph, we want to determine the class labels of groups C and D.

3 METHODOLOGY

3.1 Problem Formulation

In our problem setting, we have two kinds of information: graph instances and connections between the graph instances, which provide us with two perspectives to tackle the graph classification problem. Accordingly, we build two classifiers: a classifier IC constructed for graph instances and a classifier HC constructed for the hierarchical graph, both of which make predictions for unlabeled graph instances in G_u.

For both classifiers, one goal is to minimize the supervised loss, which measures the distance between the predicted class probabilities and the true labels. Another goal is to minimize a disagreement loss, which measures the distance between the predicted class probabilities by IC and HC. The purpose of this disagreement loss is to enforce a ***consistency*** between the two classifiers.

Formally, we formulate the graph classification problem as an optimization problem:

$$\min \zeta(G_l) + \xi(G_u), \tag{1}$$

where $\zeta(G_l)$ is the supervised loss for the labeled graph instances, and $\xi(G_u)$ is the disagreement loss for the unlabeled graph instances.

Specifically, $\zeta(G_l)$ includes two parts:

$$\zeta(G_l) = \sum_{g_i \in G_l} (\mathcal{L}(y_i, \psi_i) + \mathcal{L}(y_i, \gamma_i)), \tag{2}$$

where ψ_i is a vector of predicted class probabilities by IC, and γ_i is a vector of predicted class probabilities by HC. $\mathcal{L}(\cdot, \cdot)$ is the cross-entropy loss function.

The disagreement loss $\xi(\cdot)$ is defined as:

$$\xi(G_u) = \sum_{g_i \in G_u} D_{KL}(\gamma_i || \psi_i), \tag{3}$$

where $D_{KL}(\cdot||\cdot)$ is the Kullback-Leibler divergence, $D_{KL}(P||Q) = \sum_j P_j \log\left(\frac{P_j}{Q_j}\right)$. In the following subsections, we describe our design of classifiers IC and HC, and our approach to minimizing the supervised loss and the disagreement loss.

3.2 Design of Classifiers

Classifier IC takes a graph instance as input. As different graph instances have different numbers of nodes, IC is expected to handle graph instances of arbitrary size. Classifier HC takes the hierarchical graph as input, in which individual graph instances are the "nodes". This is a much too complicated input for a classifier. To deal with the above challenges, we propose to embed a graph instance $g_i \in G$ into a fixed-length vector e_i via IC first. Then HC can take as input the embedding vectors of graph instances and the adjacency matrix Θ. In particular, IC takes as input the adjacency matrix A_i and attribute matrix X_i of an arbitrary-sized graph instance g_i, and outputs an embedding vector e_i and a vector of predicted class probabilities ψ_i, i.e., $(e_i, \psi_i) = \text{IC}(A_i, X_i)$. HC takes the embedding vectors $E = \{e_i\}_{i=1}^{L+U}$ and Θ, and outputs the predicted class probabilities $\Gamma = \{\gamma_i\}_{i=1}^{L+U}$, i.e., $\Gamma = \text{HC}(E, \Theta)$. In the following, we illustrate the design of IC which performs discriminative graph embedding, and then the design of HC which performs graph-based classification.

3.2.1 Discriminative graph embedding. Our graph embedding task is to produce a fixed-length discriminative embedding vector of a graph instance. In the literature, graph representation techniques have recently shifted from hand-crafted kernel methods [33] to neural network based end-to-end methods, which achieve better performance in graph-structured learning tasks. In this vein, we adopt neural network methods for the graph embedding task, for which, however, we identify three challenges:

- *Size invariance*: How to design the neural network structure to flexibly take an arbitrary-sized graph instance and produce a fixed-length embedding vector?
- *Permutation invariance*: How to derive the representation regardless of the permutation of nodes?
- *Node importance*: How to encode the importance of different nodes into a unified embedding vector?

In particular, the third challenge is *node importance*, i.e., different nodes in a graph instance have different degrees of importance. For example, in a "game" group the "core" members should be more important than the "border" members in contributing to the derived embedding vector. We need to design a mechanism to learn the node importance and then encode it in the embedding vector properly.

To this end, we propose a self-attentive graph embedding method, called SAGE, which can take a variable-sized graph instance, and combine each node to produce a fixed-length vector according to their importance within the graph. In SAGE, we first utilize a multi-layer GCN [16] to smooth each node's features over the graph's topology. Then we use a self-attentive mechanism to learn the node importance and then transform a variable number of smoothed nodes into a fixed-length embedding vector, as proposed in [18]. Finally, the embedding vector is cascaded with a fully connected layer and a softmax function, in which the label information can be leveraged to discriminatively transform the embedding vector e into ψ. Figure 2 depicts the overall framework of SAGE.

Formally, we are given the adjacency matrix $A \in \mathbb{R}^{n \times n}$ and the attribute matrix $X \in \mathbb{R}^{n \times \phi}$ of a graph instance g as input. In the preprocessing step, the adjacency matrix A is normalized:

$$\hat{A} = \tilde{D}^{-\frac{1}{2}}(A + I_n)\tilde{D}^{-\frac{1}{2}}, \tag{4}$$

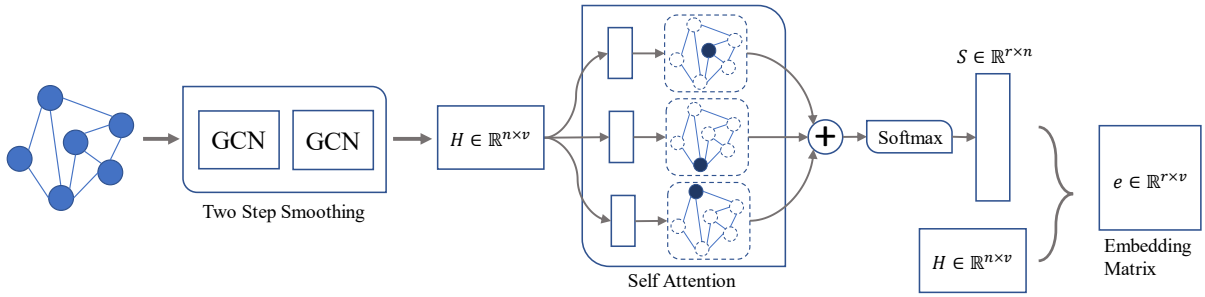


Figure 2: The supervised self-attentive graph embedding method SAGE.

where I_n is the identity matrix and $\tilde{D}_{ii} = \sum_m (A + I_n)_{im}$. Then we apply a two-layer GCN network:

$$H = \hat{A}\,\mathrm{ReLU}(\hat{A}XW^0)W^1. \tag{5}$$

Here $W^0 \in \mathbb{R}^{\phi \times h}$ and $W^1 \in \mathbb{R}^{h \times v}$ are two weight matrices. GCN can be considered as a Laplacian smoothing operator for node features over graph structures, as pointed out in [17]. Then we get a set of representation $H \in \mathbb{R}^{n \times v}$ for nodes in g. Note that the representation H does not provide node importance, and it is size variant, i.e., its size is still determined by the number of nodes n. So next we utilize the self-attentive mechanism to learn node importance and encode it into a unified graph representation, which is size invariant:

$$S = \mathrm{softmax}\Big(W_{s2}\tanh(W_{s1}H^T)\Big), \tag{6}$$

where $W_{s1} \in \mathbb{R}^{d \times v}$ and $W_{s2} \in \mathbb{R}^{r \times d}$ are two weight matrices. The function of W_{s1} is to linearly transform the node representation from a v-dimensional space to a d-dimensional space, then nonlinearity is introduced by tying with the function tanh. W_{s2} is used as r views of inferring the importance of each node within the graph. It acts like inviting r experts to give their opinions about the importance of each node independently. Then softmax is applied to derive a standardized importance of each node within the graph, which means in each view the summation of all the node importance is 1.

After that, we compute the final graph representation $e \in \mathbb{R}^{r \times v}$ by multiplying $S \in \mathbb{R}^{r \times n}$ with $H \in \mathbb{R}^{n \times v}$:

$$e = SH. \tag{7}$$

e is size invariant since it does not depend on the number of nodes n any more. It is also permutation invariant since the importance of each node is learned regardless of the node sequence, and only determined by the task labels.

One potential risk in SAGE is that r views of node importance may be similar. To diversify their views of node importance, a penalization term is imposed:

$$P = \left\| SS^T - I_r \right\|_F^2. \tag{8}$$

Here $\left\| \cdot \right\|_F$ represents the Frobenius norm of a matrix. We train the classifier in a supervised way with the task at hand, in the hope of minimizing both the penalization and the cross-entropy loss.

To summarize, we use SAGE to construct the instance-level classifier IC. It produces not only the estimated class probability vector ψ, but also a graph embedding e, which is the input for classifier HC described in the next subsection.

3.2.2 Graph-based classification. Given the graph embedding $E = \{e_i\}_{i=1}^{L+U}$ and the adjacency matrix $\Theta \in \mathbb{R}^{(L+U)\times(L+U)}$, our next task is to infer the parameters of classifier HC and derive the predicted probabilities $\Gamma = \{\gamma_i\}_{i=1}^{L+U}$. This problem falls into the setting of traditional graph-based learning where E can be treated as the set of node features. Recently neural network based approaches such as [16, 34] have demonstrated their superiority to traditional methods such as ICA [27]. In this context we make use of GCN [16] again for the consideration of efficiency and effectiveness. In the following, we consider a two-layer GCN and apply preprocessing by $\hat{\Theta} = \tilde{D_\Theta}^{-\frac{1}{2}}(\Theta + I_{L+U})\tilde{D_\Theta}^{-\frac{1}{2}}$. Then the model becomes:

$$\Gamma = HC(E, \Theta) = \mathrm{softmax}\Big(\hat{\Theta}\,\mathrm{ReLU}(\hat{\Theta}EW_\Theta^0)W_\Theta^1\Big), \tag{9}$$

where $W_\Theta^0 \in \mathbb{R}^{(rv)\times M}$ is an input-to-hidden weight matrix with M feature maps and $W_\Theta^1 \in \mathbb{R}^{M \times c}$ is a hidden-to-output weight matrix. The softmax function is applied row-wise and we get Γ. With Γ and Ψ we can compute the supervised loss in problem (2) and the disagreement loss in problem (3).

3.3 The Proposed SEAL-CI Model

In this subsection, we present our method to minimize the objective function (1). In real-world scenarios, the number of labeled graph instances L can be quite small compared to the number of unlabeled instances U. In this context, neural network based classifiers such as IC may suffer from the problem of overfitting. To mitigate this, we have both the disagreement loss (3) and the supervised loss (2) included in the objective function (1). The disagreement loss can be regarded as a regularization to prevent overfitting.

Problem (1) is a mixed combinatorial and continuous optimization problem. The supervised loss (2) includes two parts, $\mathcal{L}(y_i, \psi_i)$ and $\mathcal{L}(y_i, \gamma_i)$, i.e., the supervised loss of IC and HC. $\mathcal{L}(y_i, \gamma_i)$ depends on classifier IC to provide accurate graph embedding. All these issues make the problem highly non-convex. As such, we use the idea of iterative algorithm to alternate minimizing the supervised loss of IC and HC, and minimizing the disagreement loss by trusting a subset of predictions by HC in the next iteration of graph embedding by IC.

To be more specific, we combine the graph embedding algorithm in Section 3.2.1 and graph-based classification algorithm in Section 3.2.2 into one iterative algorithm. We build IC to produce graph embedding E^t for all graph instances in iteration t, and then feed E^t into HC to get the predicted probabilities Γ^t. We then make use of Γ^t to update the parameters of IC and generate E^{t+1}, which is

Algorithm 1: SEAL-CI

Input: A, X, Θ.
Output: Ψ^t, Γ^t.

1 Initial: $G_{tmp} = \emptyset, G_l^0 = G_l, t = 0$;
2 **while** $t\lambda \leq U$ **do**
3 $\quad \mathcal{W}^{t+1} \leftarrow \arg\min \zeta(G_l^t|\mathcal{W}^t)$;
4 $\quad \Psi^{t+1}, E^{t+1} \leftarrow \mathsf{IC}(A, X|\mathcal{W}^{t+1})$;
5 $\quad \Gamma^{t+1} \leftarrow \mathsf{HC}(E^{t+1}, \Theta|\mathcal{W}^{t+1})$;
6 $\quad G_{tmp} \leftarrow h(t\lambda, \Gamma_{G_u}^{t+1})$;
7 $\quad G_l^{t+1} \leftarrow G_l \cup G_{tmp}$;
8 $\quad G_{tmp} = \emptyset$;
9 Return Ψ^t, Γ^t;

then used as the input of HC in iteration $t+1$. Figure 3 depicts the overall framework of this iterative process. Although this method may not reach the global optimum, similar setting [20, 27] has been proven to be effective.

3.3.1 How to utilize Γ^t? To update the graph embedding vectors, a naive approach is feeding the whole set of Γ^t for the parameter update in IC, which is the idea of the original ICA [27]. However, not all Γ^t are correct in their predictions. The false predictions may lead the update of embedding neural network to the wrong direction. To this end, we make use of the idea of [20], a variant of the original ICA, and cautiously exploit a subset of Γ^t to update the parameters of IC in each iteration. Specifically, in iteration t, we choose the $t\lambda$ most confident predicted labels while ignoring the less confident predicted labels. This operation continues until all the unlabeled samples have been utilized. To further improve the efficiency, the parameters of IC are not re-trained but fine-tuned based on the parameters obtained in the previous iteration. This algorithm is called SEmi-supervised grAph cLassification via Cautious Iteration (SEAL-CI) and is presented in Algorithm 1. Note here $\mathcal{W}$ is the set of all the parameters of IC and HC. In line 6, the training set for IC has been enlarged by $t\lambda$ instances and it is done by "committing" these instances' labels from their maximum probability. In other words, the newly enrolled training instances are found by:

$$h(\lambda, \Gamma) = \text{top}(\max_{\gamma \in \Gamma} \gamma, \lambda). \tag{10}$$

Here function $\text{top}(\cdot, \lambda)$ is used to select the top λ instances and function $\max \gamma$ is used to select the maximum value in the probability vector γ.

3.4 The Proposed SEAL-AI Model

Our proposed model is easy to extend to the active learning scenario. In case further annotation is available, we can perform active learning and ask for annotations with a budget of B. Denote the set of graph instances being annotated as G_B, then the objective function in the active learning setting is re-written as:

$$\begin{aligned} &\min f(G|B, \mathcal{W}) \\ &\text{s.t.} \quad |G_B| \leq B, \end{aligned} \tag{11}$$

where $f(G|B, \mathcal{W}) = \zeta(G_l \cup G_B|\mathcal{W}) + \xi(G_u \setminus G_B|\mathcal{W})$. This is still a mixed combinatorial and continuous optimization problem. It is very hard to infer the model parameters and the active learning set G_B simultaneously. By definition, the active learning set G_B is intractable unless the model parameters are completely inferred. To solve this chicken-and-egg problem, we decompose the objective function into two sub-steps: parameter optimization and candidate generation. Then we optimize $f(G|B, \mathcal{W})$ iteratively. This algorithm is called SEmi-supervised grAph cLassification via Active Iteration (SEAL-AI) and is shown in Algorithm 2.

Algorithm 2: SEAL-AI

Input: A, X, Θ.
Output: Ψ^t, Γ^t.

1 Initial: $G_{tmp} = \emptyset, G_B^0 = \emptyset, G_l^0 = G_l, G_u^0 = G_u, t = 0$;
2 **while** $|G_B^t| \leq B$ **do**
3 $\quad \mathcal{W}^{t+1} \leftarrow \arg\min \zeta(G_l^t|\mathcal{W}^t)$;
4 $\quad \Psi^{t+1}, E^{t+1} \leftarrow \mathsf{IC}(A, X|\mathcal{W}^{t+1})$;
5 $\quad \Gamma^{t+1} \leftarrow \mathsf{HC}(E^{t+1}, \Theta|\mathcal{W}^{t+1})$;
6 $\quad G_{tmp} \leftarrow \arg\min_{|G_{tmp}|=k} \xi(G_u^t \setminus G_{tmp}|\mathcal{W}^{t+1})$;
7 $\quad G_B^{t+1} \leftarrow G_B^t \cup G_{tmp}$;
8 $\quad G_l^{t+1} \leftarrow G_l^t \cup G_{tmp}$;
9 $\quad G_u^{t+1} \leftarrow G_u^t \setminus G_{tmp}$;
10 $\quad G_{tmp} = \emptyset$;
11 Return Ψ^t, Γ^t;

At the beginning of this iterative process, we optimize the supervised loss $\zeta(G_l|\mathcal{W})$ based on current labeled graphs in G_l (line 3 in Algorithm 2). In active learning, the choice of candidate generator is a key component. We exploit the idea of ALFNET [1] and choose the candidate graph instances G_{tmp} by maximizing the decrease of the current disagreement loss based on the new parameter obtained in the first step (line 6 in Algorithm 2). At last we label G_{tmp} and update G_B, G_l and G_u respectively (line 7-9 in Algorithm 2).

It is worth noting that from the hard example mining perspective, the disagreement score is an excellent criterion for the active learning setting. Specifically, we choose the candidates by first calculating the distribution divergence of (γ_i, ψ_i) from $\Gamma_u = \{\gamma_i\}_{i=1}^U$ and $\Psi_u = \{\psi_i\}_{i=1}^U$:

$$z(\psi_i, \gamma_i) = D_{KL}(\gamma_i||\psi_i). \tag{12}$$

Then we choose k instances with the largest KL divergence. Intuitively, the KL divergence between ψ_i and γ_i can be viewed as the conflict of two supervised models. A large KL divergence indicates that one of the models gives wrong predictions. To this end, the instances with a large KL divergence are more informative to help the algorithm converge more quickly.

3.5 Complexity Analysis

We analyze the computational complexity of our proposed methods. Here we only focus on Algorithm 1, since Algorithm 2 is almost the same except the step of selecting candidate graph instances to the training set. In Algorithm 1, the intensive parts in each iteration contain the updates of IC and HC as well as the selection of candidate instances. We discuss each part in details below.

Regarding IC, the core is to compute the activation matrix H in Eq. (5) where the matrix-vector multiplications are up to $O(E_1\phi)$

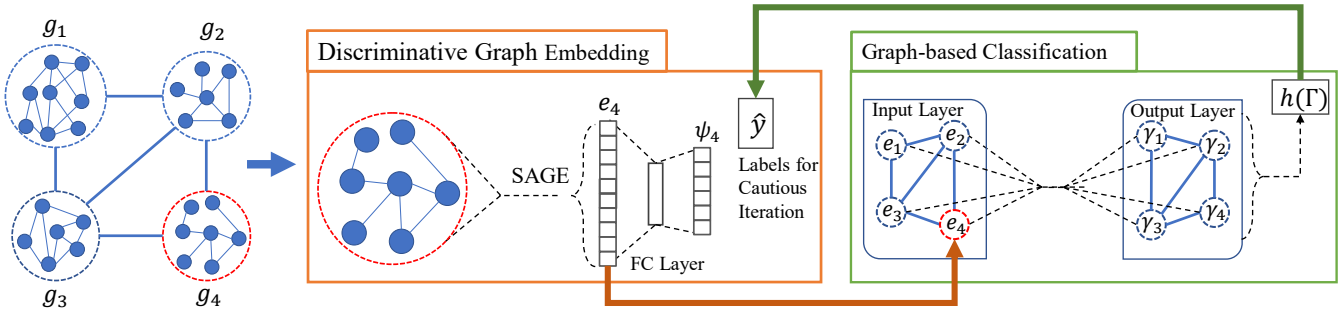


Figure 3: Schematic diagram of the learning framework SEAL-CI. There are two subroutines: discriminative graph embedding (in the orange box) and graph-based classification (in the green box).

flops for one input graph instance; here E_1 denotes the number of edges in the graph instance and ϕ is the input feature dimension. Thus, it leads to the complexity of $O(E_1(L+U)\phi)$ by going through all $L+U$ graph instances.

Next, the computation by HC in Eq. (9) requires $O(E_2 rv)$ flops in total, where E_2 denotes the number of links between graph instances. Then in candidate selection, performing comparisons between all unlabeled graph instances has a complexity of $O(L+U)$ given the outputs of two classifiers IC and HC.

Overall, the complexity of our method is $O(E_1(L+U)\phi + E_2 rv)$ which scales linearly in terms of the number of edges in each graph instance (i.e., E_1), the number of links between graph instances (i.e., E_2) and the number of graph instances (i.e., $(L+U)$). Thus, our method is computationally comparable to the GCN-based method [16], and more efficient than PSCN [23] that is quasi-linear with respect to the numbers of nodes and edges.

4 EXPERIMENTS

We first validate the effectiveness of our graph embedding algorithm SAGE on two data sets: PROTEINS and D&D. Then we evaluate our SEAL-C/AI methods on both synthetic and Tencent QQ group data sets.

4.1 Performance of SAGE

We use two benchmark data sets, PROTEINS and D&D, to evaluate the classification accuracy of SAGE, and compare it with the state-of-the-art graph kernels and deep learning approaches. PROTEINS [4] is a graph data set where nodes are secondary structure elements and edges represent that two nodes are neighbors in the amino-acid sequence or in 3D space. D&D [7] is a set of structures of enzymes and non-enzymes proteins, where nodes are amino acids, and edges represent spatial closeness between nodes. Table 1 lists the statistics of these two data sets.

4.1.1 Baselines and Metrics. The baselines include four graph kernels and two deep learning approaches:

- the shortest-path kernel (SP) [3],
- the random walk kernel (RW) [9],
- the graphlet count kernel (GK) [30],
- the Weisfeiler-Lehman subtree kernel (WL) [29],
- PATCHY-SAN (PSCN) [23], and
- graph2vec [22].

Table 1: Statistics of PROTEINS and D&D

	PROTEINS	D&D
Max number of nodes	620	5748
Avg number of nodes	39.06	284.32
Number of graphs	1113	1178

Table 2: Accuracy of different classifiers

Approach	PROTEINS	D&D
SP	75.07±0.54%	-
RW	74.22±0.42%	-
GK	71.67±0.55%	78.45±0.26%
WL	72.92±0.56%	77.95±0.70%
PSCN	75.89±2.76%	77.12±2.41%
graph2vec	73.30±2.05%	-
SAGE	**77.26**±2.28%	**80.88**±2.33%

We follow the experimental setting as described in [23], and perform 10-fold cross validation. In each partition, the experiments are repeated for 10 times. The average accuracy and the standard deviation are reported. We list results of the graph kernels and the best reported results of PSCN according to [23].

For SAGE, we use the same network architecture on both data sets. The first GCN layer has 128 output channels, and the second GCN has 8 output channels. We set $d = 64$, $r = 16$, and the penalization term coefficient to be 0.15. The dense layer has 256 rectified linear units with a dropout rate of 0.5. We use minibatch based Adam [15] to minimize the cross-entropy loss and use He-normal [11] as the initializer for GCN. For both data sets, the only hyperparameter we optimized is the number of epochs.

4.1.2 Results. Table 2 lists the experimental results. As we can see, SAGE outperforms all the graph kernel methods and the two deep learning methods by 1.27% – 5.59% in accuracy. This shows that our graph embedding method SAGE is superior.

Table 3: Statistics of generated graph instances

Type	Number	Nodes	Edges	Density
Watts-Strogatz	351	173	347	2.3%
Tree	217	127	120	1.5%
Erdős-Rényi	418	174	3045	20%
Barbell	818	169	2379	16.3%
Bipartite	426	144	1102	10.6%
Barabási-Albert	298	173	509	3.4%
Path	180	175	170	1.1%

The node and edge numbers and density are the average for each type of graph.

4.2 SEAL-C/AI on Synthetic Data

We evaluate the performance of SEAL-C/AI on synthetic data. We first give a description of the synthetic generator, then visualize the learned embeddings and analyze the self-attentive mechanism on the generated data. Finally we compare our methods with baselines in terms of classification accuracy.

4.2.1 Synthetic Data Generation. The benchmark data set Cora [19] contains 2708 papers which are connected by the "citation" relationship. We borrow the topological structure of Cora to provide the skeleton (i.e., edges) of our synthetic hierarchical graph. Then we generate a set of graph instances, which serve as the nodes of this hierarchical graph. Since there are 7 classes in Cora, we adopt 7 different graph generation algorithms, that is, Watts-Strogatz [32], Tree graph, Erdős-Rényi [8], Barbell [13], Bipartite graph, Barabási-Albert graph [2] and Path graph, to generate 7 different types of graph instances, and connect them in the hierarchical graph.

Specifically, to generate a graph instance g, we randomly sample a number from $[100, 200]$ as its size n. Then we generate its structure and assign the class label according to the graph generation algorithm. In this step, the parameter p in Watts-Strogatz, Erdős-Rényi, Bipartite graph and Barabási-Albert graph is randomly sampled from $[0.1, 0.5]$, the branching factor for Tree graph is randomly sampled from $[1, 3]$. At last, to make this problem more challenging, we randomly remove 1% to 20% edges in the generated graph g. The statistics of the generated graph instances are listed in Table 3.

4.2.2 Visualization. To have a better understanding of the synthesized graph instances, we split all 2708 graph instances into two parts. 1708 instances are used for training and the remaining 1000 instances are used for testing. We apply SAGE on the training set and derive the embeddings of the 1000 testing instances. We then project these learned embeddings into a two-dimensional space by t-SNE [31], as depicted in Figure 4. Each color in Figure 4 represents a graph type. As we can see from this two-dimensional space, the geometric distance between the graph instances can reflect their graph similarity properly.

We then examine the self-attentive mechanism of SAGE. We calculate the average attention weight across r views and normalize the resulting attention weights to sum up to 1. From the testing instances, we select three examples: a Tree graph, an Erdős-Rényi graph and a Barbell graph, for which SAGE has a high confidence (> 0.9) in predicting their class label. The three examples are depicted in Figure 5, where a bigger node implies a larger average attention weight, and a darker color implies a larger node degree. On the left is a Tree graph, in which most of the important nodes learned by SAGE are leaf nodes. This is reasonable since leaves are discriminative features to distinguish Tree graph from the other 6 types of graphs. In the center is an Erdős-Rényi graph. We cluster these nodes into 5 groups by hierarchical clustering [14], and see that SAGE tends to highlight those nodes with large degrees within each cluster. On the right is a Barbell graph, in which SAGE pays attention to two kinds of nodes. The first kind is those nodes that connect a dense graph and a path, and the second kind is the nodes that are on the path.

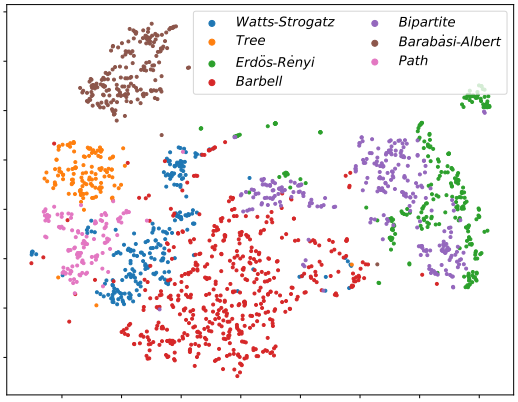


Figure 4: Two-dimensional visualization of graph embeddings generated from the synthesized graph instances using SAGE. The nodes are colored according to their graph types.

4.2.3 Baselines and Metrics. We use 6 approaches as our baselines:

- GK-SVM/GCN [30], which calculates the graphlet count kernel (GK) matrix, then GK-SVM feeds the kernel matrix into SVM [12] whereas GK-GCN feeds the kernel vector of each graph instance to GCN.
- WL-SVM/GCN [29], which is similar as above but using the Weisfeiler-Lehman subtree kernel (WL).
- graph2vec-GCN [22], which embeds the graph instances by graph2vec and then feeds the embeddings to GCN.
- cautious-SAGE-Cheby, which is similar to SEAL-CI except that we replace GCN with Cheby-GCN [6].
- active-SAGE-Cheby, which is similar to SEAL-AI except that we replace GCN with Cheby-GCN [6].
- SAGE, which ignores the connections between graph instances and treats them independently.

We use 300 graph instances as the training set for all methods except SEAL-AI and active-SAGE-Cheby, for which only 140 graphs are used as labeled graph instances at hand and then $B = 160$ is set for active learning. We use 1000 graph instances as the testing set. We run each method 5 times and report its average accuracy. The number of epochs for graph2vec is 1000 and the learning rate is 0.3. To avoid overfitting of SAGE on this small data set, we use a relatively small number of neurons. The first GCN layer has 32 output channels and the second GCN layer has 4 output channels.

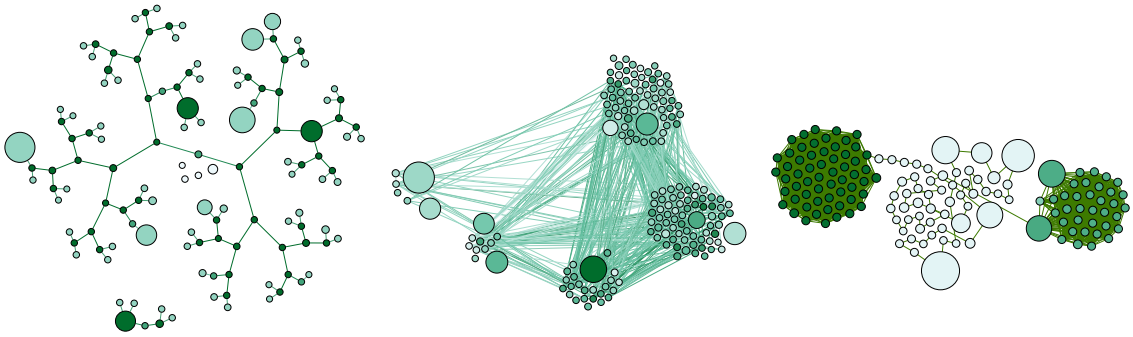


Figure 5: Attention of graph embeddings on 3 different types of graphs (left: Tree graph; middle: Erdős-Rényi graph; right: Barbell graph). A bigger node indicates a larger importance, and a darker color implies a larger node degree.

We set $d = 32$ and $r = 10$. The dense layer has 48 units with a dropout rate of 0.3. We set $M = 16$ in HC.

4.2.4 Results. Table 4 shows the experimental results for semi-supervised graph classification. Among all approaches, SEAL-C/AI achieve the best performance. In the following, we analyze the performance of all methods categorized into 4 groups.

Group *1: Both GK-SVM and WL-SVM outperform their GCN-based counterparts, indicating that SVM is more effective than GCN with the computed kernel matrix. All the embedding-based methods perform better than these two kernel methods, which proves that embedding vectors are effective representations for graph instances and are suitable input for graph neural networks.

Group *2: graph2vec-GCN achieves 85.2% accuracy, which is comparable to that of SAGE, but lower than that of SEAL-C/AI. One possible explanation is that graph2vec is an unsupervised embedding method, which fails to generate discriminative embeddings for classification. Another possibility is that there is no iteration in this method, and the 300 training instances do not include very informative ones. These limitations of graph2vec are also motivations for us to design the supervised embedding method SAGE and the iterative framework in SEAL-CI.

Group *3: cautious-SAGE-Cheby outperforms SAGE by only 0.8%, which is not remarkable considering that it exploits many more training instances. The accuracy of active-SAGE-Cheby is 3.3% lower than that of SEAL-AI, which means that Cheby-GCN is inferior to GCN.

Group *4: Both SEAL-CI and SEAL-AI outperform SAGE significantly, which proves the effectiveness of our hierarchical graph based perspective and the iterative algorithm for graph classification. SEAL-AI outperforms SEAL-CI only slightly, by 1.2%. This shows, although SEAL-CI can make use of more training samples, it is still influenced by the misclassified cases of GCN.

4.2.5 Influence of the number of labeled training instances. We examine how the number of labeled training instances affects the performance of our methods. We train SAGE and SEAL-CI with a label size of $\{140, 180, 220, 260, 300\}$. We train SEAL-AI with 140 labeled instances and then set the budget B for active learning at $\{0, 40, 80, 120, 160\}$. Thus the three methods have the same number of labeled training instances. We set $\lambda = 40$ in SEAL-CI and $k = 10$ in SEAL-AI. We run all methods 5 times, and plot their average accuracy in Figure 6. As we can see from Figure 6, when the number of labeled training instances is 140, SEAL-CI performs best since it can utilize more training samples. As the number of labeled training instances increases, the performance of SEAL-AI improves dramatically. SEAL-AI catches up with SEAL-CI at 260 labeled training instances and outperforms SEAL-CI at 300 labeled training instances. It validates that SEAL-AI can make use of the iterations to find informative and accurate training samples. Meanwhile SEAL-CI trusts the prediction of GCN conditionally on its confidence, which may bring some noise to the learning process. SEAL-C/AI outperform SAGE in all cases, which makes sense because SEAL-C/AI make good use of the hierarchical graph setting and consider the connections between the graph instances for classification.

Table 4: Comparison of different methods on the synthetic data set for semi-supervised graph classification

	Algorithm	Accuracy
*1	**GK-SVM/GCN**	77.8%/73.4%
	WL-SVM/GCN	83.4%/75.5%
*2	**graph2vec-GCN**	85.2%
*3	**cautious-SAGE-Cheby**	86.5%
	active-SAGE-Cheby	89.1%
*4	**SAGE**	85.7%
	SEAL-CI	91.2%
	SEAL-AI	**92.4**%

4.3 SEAL-C/AI on Tencent QQ Group

In this section, we evaluate SEAL-C/AI on Tencent QQ group data. We describe the characteristics of this data set and then present the experimental results. Finally, we have some open discussions on how to construct a hierarchical graph from real-world data.

4.3.1 Data Description. Tencent QQ is a social networking platform in China with nearly 800 million monthly active users[1]. There

[1]https://www.tencent.com/en-us/articles/17000391523362601.pdf

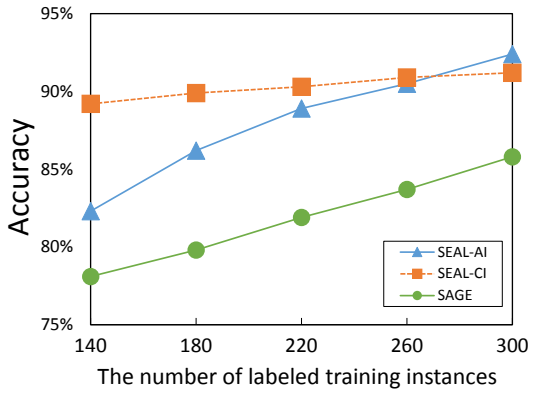


Figure 6: Accuracy with different number of labeled training instances on synthetic data for semi-supervised graph classification.

Table 5: Statistics of collected Tencent QQ groups

Class label	Number	Nodes	Edges	Density
game	1,773	147	395	5.48%
non-game	36,063	365	1586	3.28%

The node and edge numbers and density are the average for each type of QQ group.

are around 100 million active online QQ groups. In this experiment, we select 37,836 QQ groups with 18,422,331 unique anonymized users. For each user, we extract seven personal features:

- number of days ever since the registration day;
- most frequently active area code in the past 90 days;
- number of friends;
- number of active days in the past 30 days;
- number of logging in the past 30 days;
- number of messages sent in the past 30 days;
- number of messages sent within QQ groups in the past 30 days.

We have 298,837,578 friend relationships among these users. 1,773 groups are labeled as "game" and the remaining groups are labeled as "non-game".

We construct the hierarchical graph from this Tencent QQ group data as follows. A user is treated as an object, and a QQ group as a graph instance. The users in one group are connected by their friendship. The attribute matrix X is filled with the attribute values of the users. The statistics of the graph instances are listed in Table 5. We build the hierarchical graph from the graph instances via common members across groups. That is, if groups A and B have more than one common member, we connect them.

4.3.2 Baselines and Metrics. We use the same set of baselines as in Section 4.2.3. 1000 graph instances are used as labeled training instances for all methods except SEAL-AI and active-SAGE-Cheby, for which only 500 are used as labeled training instances at hand and then B is set to 500 for active learning. We use 10,000 instances for testing for all methods. We run each method 3 times and report its average accuracy. The hyperparameters of SAGE are the same as the settings in Section 4.1.1. Since the class distribution is quite imbalanced in this data set, we report the Macro-F1 instead of accuracy.

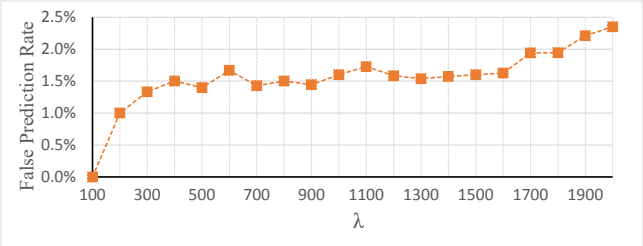


Figure 7: The false prediction rate of GCN with λ in SEAL-CI.

Table 6: Comparison of different methods on Tencent QQ group data for semi-supervised graph classification

	Algorithm	Macro-F1
*1	**GK-SVM**	48.8%
	WL-SVM	47.8%
*2	**graph2vec-GCN**	48.1%
*3	**cautious-SAGE-Cheby**	64.3%
	active-SAGE-Cheby	66.7%
*4	**SAGE**	54.7%
	SEAL-CI	70.8%
	SEAL-AI	**73.2%**

4.3.3 Results. Table 6 shows the experimental results. SEAL-C/AI outperform GK, WL and grah2vec by at least 12% in Macro-F1. Within our framework, GCN is better than Cheby-GCN for about 6%. SEAL-AI outperforms SEAL-CI by 2.4%. Next we provide the reason why SEAL-AI outperforms SEAL-CI on this data set. Figure 7 shows the false prediction rate (i.e., the percentage of misclassified instances) within the λ most confident predictions of GCN. As we can see, the false prediction rate increases as λ increases and it reaches 2.4% when $\lambda = 2000$. In the framework of SEAL-CI, as the iteration goes on, we shall bring in more noise to the parameter update of SAGE, while all the training samples in SEAL-AI are informative and correct. This explains why SEAL-AI outperforms SEAL-CI on this Tencent QQ group data.

4.3.4 Visualization. We provide visualization of a "game" group and its neighborhood in Figure 8. The left part is the ego network of the center "game" group. In the one-hop neighborhood of this "game" group, there are 10 "game" groups and 19 "non-game" groups. "Game" groups are densely interconnected with a density of 34.5%, whereas "non-game" groups are sparsely connected with a density of 8.8%. The much higher density among "game" groups validates that common membership is an effective way to relate them in a hierarchical graph for classification. The right part depicts the internal structure of the ego "game" group with 22 users. A bigger node indicates a larger importance, and a darker green color implies a larger node degree. These 22 members are loosely connected and there are no triangles. This makes sense because in reality online "game" groups are not acquaintance networks. Regarding

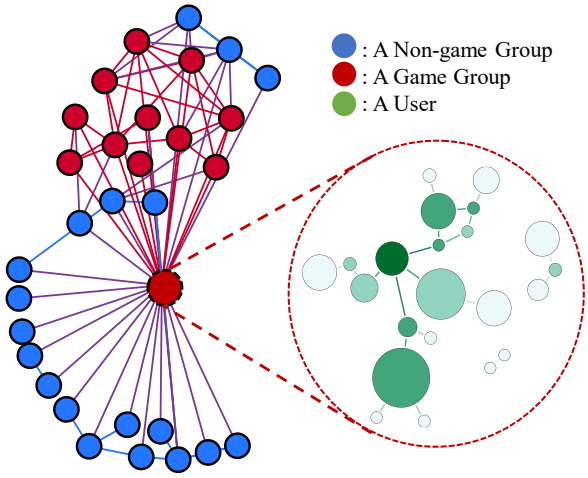


Figure 8: The ego network of a "game" group. The left side is the ego network, in which "game" groups are in red and "non-game" groups are in blue. The right side is the internal structure of the ego "game" group, in which a bigger node indicates a larger importance, and a darker color implies a larger node degree.

the learned node importance, node 1 has the highest importance as it is the second active member and has a large degree in this group. Node 16 is also important since it has the highest degree in this group. The "border" member 5 has a big attention weight since it has the largest number of days ever since the registration day and is quite active in this group.

4.3.5 Discussion. How to construct a hierarchical graph from raw data is an open question. In the above experiment, we connect two QQ groups if they have more than one common member (i.e., > 1). When we change the threshold, it directly affects the edge density in the hierarchical graph, and may influence the classification performance. For example, if we connect two QQ groups when they have one common member or more (i.e., ≥ 1), the edge density is 2.8% compared with 0.27% in the first setting. A proper setting of this threshold is data dependent, and can be determined through a validation set.

5 RELATED WORK

This work is related to semi-supervised classification of networked data, variable-sized graph embedding and active learning.

Most work on semi-supervised learning for networked data aims to utilize the network structure to boost the learning performance. The assumption is that network context can provide additional information that is not covered by node attributes. Ever since the pioneer work of Sen et al. [27], Iterative Classification Algorithm (ICA) has become a paradigm for networked data with limited annotations. In ICA, for each node a local classifier takes the estimated labels of its neighborhood and its own features as input, and outputs a new estimated label. The iteration continues until adjacent estimations stabilize. In ALFNET [1], the authors first cluster the network nodes into several groups, and design a content-only classifier CO and a collective classifier CC. Based on the disagreement score of CO and CC in each iteration, a candidate instance set is generated from different clusters and labeled. Then both CO and CC are re-trained using the labeled set until convergence. One main difference between ICA and ALFNET is that ICA does not require human intervention while ALFNET needs human annotation in case labels of the candidate set are not available.

Recent work has focused on using deep learning neural networks to further improve the performance. [34] leverages both network context and node features by jointly training node embedding to predict the class label and the context of the network. Later Kipf and Welling [16] simplify the loss design by only considering the supervised loss while network context is exploited by the GCN operator. Our problem setting is different from all of the above, as the node is no longer a fixed-size feature vector but a variable-size graph. It can be regarded as a generalization of the previous setting, and cannot be handled by existing solutions effectively.

Representation learning on graphs has been proposed to transform instances in topological space into fixed-size vectors in Euclidean space in which geometric distance reflects their structural similarity. There are two trends on this topic, one of which is a shift from node embedding [10, 24] to whole graph embedding. [33] uses CBOW and skip-gram model [21], previously proven to be successful in natural language processing, to learn a new graph kernel. Meanwhile, some other methods focus on generating graph embeddings by integrating node embeddings. [23] proposes a spatial-based graph CNN operator and then concatenates these obtained node representations by imposing a problem-specific node ordering. [6] defines a "graph coarsening" operation by first clustering the node representations and then applying a max-pooling operation. However, all these methods need some preprocessing steps such as node ordering or clustering, which is not a necessity from a data-driven perspective. Another trend is a shift from unsupervised embedding [21] to supervised embedding [5, 18], which provides better performance for downstream classification tasks. In this sense, our embedding method SAGE performs whole graph embedding in a supervised way.

Active learning has been integrated in many collective classification methods [1, 28] to find the most informative samples to be labeled. However, research that generalizes active learning with deep semi-supervised learning is still lacking. The closest work is [35] in which the authors utilize active learning to incrementally fine-tune a CNN network for image classification. Our solution SEAL-AI is different in the sense that the informative samples selected by active learning are used to update the parameters of the graph embedding network, whose output is then fed into HC in an iterative framework.

6 CONCLUSION

In this paper, we study semi-supervised graph classification from a hierarchical graph perspective. The hierarchical graph is a much too complicated input for classification, thus we first design a supervised, self-attentive graph embedding method SAGE to embed graph instances into fixed-length vectors, which are a common input form for classification. We build two classifiers IC and HC at the

graph instance level and the hierarchical graph level respectively to fully exploit the available information. Our semi-supervised solutions SEAL-C/AI adopt an iterative framework to update IC and HC alternately with an enlarged training set. Experimental results on synthetic graphs and Tencent QQ group data show that SEAL-C/AI outperform other competitors by a significant margin in accuracy/Macro-F1, and they also generate meaningful interpretations of the learned representations for graph instances.

ACKNOWLEDGMENTS

The authors would like to thank Tencent Security Platform Department for discussions and suggestions. The work described in this paper was supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China [Project No.: CUHK 14205618], Tencent AI Lab Rhino-Bird Focused Research Program GF201801 and the CUHK Stanley Ho Big Data Decision Analytics Research Centre.

REFERENCES

[1] M. Bilgic, L. Mihalkova, and L. Getoor. 2010. Active learning for networked data. In *ICML*. 79–86.
[2] B. Bollobás and O. M. Riordan. 2003. Mathematical results on scale-free random graphs. *Handbook of graphs and networks: from the genome to the internet* (2003), 1–34.
[3] K. M. Borgwardt and H.-P. Kriegel. 2005. Shortest-path kernels on graphs. In *ICDM*. 74–81.
[4] K. M. Borgwardt, C. S. Ong, S. Schönauer, S.V.N. Vishwanathan, A. J. Smola, and H.-P. Kriegel. 2005. Protein function prediction via graph kernels. In *ISMB*. 47–56.
[5] H. Dai, B. Dai, and L. Song. 2016. Discriminative embeddings of latent variable models for structured data. In *ICML*. 2702–2711.
[6] M. Defferrard, X. Bresson, and P. Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*. 3844–3852.
[7] P. D. Dobson and A. J. Doig. 2003. Distinguishing enzyme structures from non-enzymes without alignments. *Journal of molecular biology* 330, 4 (2003), 771–783.
[8] P. Erdős and A Rényi. 1960. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci* 5, 1 (1960), 17–60.
[9] T. Gärtner, P. Flach, and S. Wrobel. 2003. On graph kernels: Hardness results and efficient alternatives. In *Learning theory and kernel machines*. Springer, 129–143.
[10] A. Grover and J. Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*. 855–864.
[11] K. He, X. Zhang, S. Ren, and J. Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*. 1026–1034.
[12] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their Applications* 13, 4 (1998), 18–28.
[13] M. Herbster and M. Pontil. 2007. Prediction on a graph with a perceptron. In *NIPS*. 577–584.
[14] S. C. Johnson. 1967. Hierarchical clustering schemes. *Psychometrika* 32, 3 (1967), 241–254.
[15] D. P. Kingma and J. Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.
[16] T. N. Kipf and M. Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.
[17] Q. Li, Z. Han, and X. Wu. 2018. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. In *AAAI*. 3538–3545.
[18] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. 2017. A Structured Self-attentive Sentence Embedding. In *ICLR*.
[19] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore. 2000. Automating the construction of internet portals with machine learning. *Information Retrieval* 3, 2 (2000), 127–163.
[20] L. K. McDowell, K. M. Gupta, and D. W. Aha. 2007. Cautious inference in collective classification. In *AAAI*. 596–601.
[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. 3111–3119.
[22] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal. 2017. graph2vec: Learning Distributed Representations of Graphs. *CoRR* abs/1707.05005 (2017). arXiv:1707.05005
[23] M. Niepert, M. Ahmed, and K. Kutzkov. 2016. Learning Convolutional Neural Networks for Graphs. In *ICML*. 2014–2023.
[24] B. Perozzi, R. Al-Rfou, and S. Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*. 701–710.
[25] R. Ramanath, H. Inan, G. Polatkan, B. Hu, Q. Guo, C. Ozcaglar, X. Wu, K. Kenthapadi, and S. C. Geyik. 2018. Towards Deep and Representation Learning for Talent Search at LinkedIn. In *CIKM*. 2253–2261.
[26] F. Rousseau, E. Kiagias, and M. Vazirgiannis. 2015. Text categorization as a graph classification problem. In *ACL-IJCNLP*. 1702–1712.
[27] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–106.
[28] B. Settles. 2012. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6, 1 (2012), 1–114.
[29] N. Shervashidze, P. Schweitzer, E. J. v. Leeuwen, K. Mehlhorn, and K. M. Borgwardt. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research* 12, Sep (2011), 2539–2561.
[30] N. Shervashidze, S.V.N. Vishwanathan, T. Petri, K. Mehlhorn, and K. M. Borgwardt. 2009. Efficient graphlet kernels for large graph comparison. In *AISTATS*. 488–495.
[31] L. v. d. Maaten and G. Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.
[32] D. J. Watts and S. H. Strogatz. 1998. Collective dynamics of âĂŸsmall-worldâĂŹ networks. *Nature* 393 (1998), 440–442.
[33] P. Yanardag and S.V.N. Vishwanathan. 2015. Deep Graph Kernels. In *KDD*. 1365–1374.
[34] Z. Yang, W. W. Cohen, and R. Salakhutdinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *ICML*. 40–48.
[35] Z. Zhou, J. Shin, L. Zhang, S. Gurudu, M. Gotway, and J. Liang. 2017. Fine-Tuning Convolutional Neural Networks for Biomedical Image Analysis: Actively and Incrementally. In *CVPR*. 4761–4772.

Efficient Ridesharing Order Dispatching with Mean Field Multi-Agent Reinforcement Learning

Minne Li
University College London

Zhiwei (Tony) Qin
DiDi Research America

Yan Jiao
DiDi Research America

Yaodong Yang
University College London

Zhichen Gong
University College London

Jun Wang
University College London

Chenxi Wang
DiDi Research

Guobin Wu
DiDi Research

Jieping Ye
DiDi Research

ABSTRACT

A fundamental question in any peer-to-peer ridesharing system is how to, both effectively and efficiently, dispatch user's ride requests to the right driver in real time. Traditional rule-based solutions usually work on a simplified problem setting, which requires a sophisticated hand-crafted weight design for either centralized authority control or decentralized multi-agent scheduling systems. Although recent approaches have used reinforcement learning to provide centralized combinatorial optimization algorithms with informative weight values, their single-agent setting can hardly model the complex interactions between drivers and orders. In this paper, we address the order dispatching problem using multi-agent reinforcement learning (MARL), which follows the distributed nature of the peer-to-peer ridesharing problem and possesses the ability to capture the stochastic demand-supply dynamics in large-scale ridesharing scenarios. Being more reliable than centralized approaches, our proposed MARL solutions could also support fully distributed execution through recent advances in the Internet of Vehicles (IoV) and the Vehicle-to-Network (V2N). Furthermore, we adopt the mean field approximation to simplify the local interactions by taking an average action among neighborhoods. The mean field approximation is capable of globally capturing dynamic demand-supply variations by propagating many local interactions between agents and the environment. Our extensive experiments have shown the significant improvements of MARL order dispatching algorithms over several strong baselines on the accumulated driver income (ADI), and order response rate measures. Besides, the simulated experiments with real data have also justified that our solution can alleviate the supply-demand gap during the rush hours, thus possessing the capability of reducing traffic congestion.

KEYWORDS

Multi-Agent Reinforcement Learning; Mean Field Reinforcement Learning; Order Dispatching

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313433

ACM Reference Format:
Minne Li, Zhiwei (Tony) Qin, Yan Jiao, Yaodong Yang, Zhichen Gong, Jun Wang, Chenxi Wang, Guobin Wu, and Jieping Ye. 2019. Efficient Ridesharing Order Dispatching with Mean Field Multi-Agent Reinforcement Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313433

1 INTRODUCTION

Real-time ridesharing refers to the task of helping to arrange one-time shared rides on very short notice [4, 10]. Such technique, embedded in popular platforms including Uber, Lyft, and DiDi Chuxing, has greatly transformed the way people travel nowadays. By exploiting the data of individual trajectories in both space and time dimensions, it offers more efficiency on traffic management, and the traffic congestion can be further alleviated as well [18].

One of the critical problems in large-scale real-time ridesharing systems is how to dispatch orders, i.e., to assign orders to a set of active drivers on a real-time basis. Since the quality of order dispatching will directly affect the utility of transportation capacity, the amount of service income, and the level of customer satisfaction, therefore, solving the problem of order dispatching is the key to any successful ride-sharing platform. In this paper, our goal is to develop an intelligent decision system to maximize the accumulated driver income (ADI), i.e., the value of all the orders served in a single day, with the ability to scale up to a large number of drivers and robust to potential hardware or connectivity failures. The challenge of order dispatching is to find an optimal trade-off between the short-term and long-term rewards. When the number of available orders is larger than that of active drivers within the order broadcasting area (shown as the grey shadow area in the center of Fig. 1b), the problem turns into finding an optimal order choice for each driver. Taking an order with a higher price will contribute to the immediate income; however, it might also harm the ADI in the long run if this order takes the driver to a sparsely populated area. As illustrated in Fig. 1a, considering the two orders starting from the same area but to two different destinations, a neighboring central business district (CBD) and a distant suburb. A driver taking the latter one may have a higher one-off order price due to the longer travel distance, but the subsequent suburban area with little demand could also prevent the driver from further sustaining income. Dispatching too many such orders will, therefore, reduce the number of orders taken and harm the long-term ADI. The problem becomes more serious

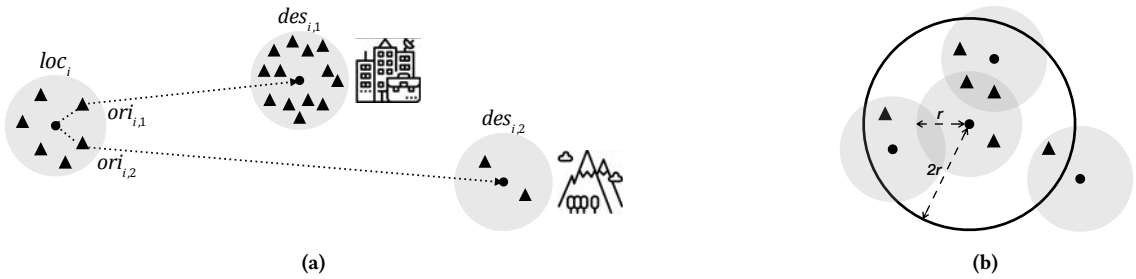


Figure 1: The order dispatching problem. (a) Two order choices (black triangle, departing from $ori_{i,1}$ and $ori_{i,2}$ respectively) within a driver i's (black dot, located at loc_i) order receiving area (grey shadow), where one ends at a neighboring CBD (located at $des_{i,1}$) with high demand, and the other ends at a distant suburb (located at $des_{i,2}$) with low demand. Both orders have the same pick-up distance for driver i. (b) Details within a neighborhood, where the radius of the order receiving area and the neighborhood (black circle) are r and $2r$ respectively.

particularly during the peak hours when the situation in places with the imbalance between vehicle supply and order demand gets worse. As such, an intelligent order dispatching system should be designed to not only assign orders with high prices to the drivers, but also to anticipate the future demand-supply gap and distribute the imbalance among different destinations. In the meantime, the pick-up distance should also be minimized, as the drivers will not get paid during the pick-up process; on the other hand, long waiting time will affect the customer experience.

One direction to tackle the order dispatching challenge has been to apply hand-crafted features to either centralized dispatching authorities (e.g., the combinatorial optimization algorithm [28, 45]) or distributed multi-agent scheduling systems [3, 40], in which a group of autonomous agents that share a common environment interact with each other. However, the system performance relies highly on the specially designed weighting scheme. For centralized approaches, another critical issue is the potential "single point of failure" [23], i.e., the failure of the centralized authority control will fail the whole system. Although the multi-agent formulation provides a distributed perspective by allowing each driver to choose their order preference independently, existing solutions require rounds of direct communications between agents during execution [30], thus being limited to a local area with a few agents.

Recent attempts have been made to formulate this problem with centralized authority control and model-free reinforcement learning (RL) [33], which learns a policy by interacting with a complex environment. However, existing approaches [41] formulate the order dispatching problem with the single-agent setting, which is unable to model the complex interactions between drivers and orders, thus being oversimplifications of the stochastic demand-supply dynamics in large-scale ridesharing scenarios. Also, executing an order dispatching system in a centralized manner still suffers from the reliability issue mentioned above that is generally inherent to the centralized architecture.

Staying different from these approaches, in this work we model the order dispatching problem with multi-agent reinforcement learning (MARL) [7], where agents share a centralized judge (the critic) to rate their decisions (actions) and update their strategies (policies). The centralized critic is no longer needed during the execution period as agents can follow their learned policies independently, making the order dispatching system more robust to potential hardware or connectivity failures. With the recent development of the Internet of Things (IoT) [5, 11, 44] and the Internet of Vehicles (IoV) [22, 42], the fully distributed execution could be practically deployed by distributing the centralized trained policy to each vehicle through the Vehicle-to-Network (V2N) [1]. By allowing each driver to learn by maximizing its cumulative reward through time, the reinforcement learning based approach relieves us from designing a sophisticated weighting scheme for the matching algorithms. Also, the multi-agent setting follows the distributed nature of the peer-to-peer ridesharing problem, providing the dispatching system with the ability to capture the stochastic demand-supply dynamics in large-scale ridesharing scenarios. Meanwhile, such fully distributed executions also enable us to scale up to much larger scenarios with many more agents, i.e., a scalable real-time order dispatching system for ridesharing services with millions of drivers.

Nonetheless, the major challenge in applying MARL to order dispatching lies in the changing dynamics of two components: the size of the action set and the size of the population. As illustrated in Fig. 1b, the action set for each agent is defined as a set of neighboring active orders within a given radius r from each active driver (shown as the shadow area around each black dot). As the active orders will be taken and new orders keep arriving, the size and content of this set will constantly change over time. The action set will also change when the agent moves to another location and arrives in a new neighborhood. On the other hand, drivers can also switch between online and offline in the real-world scenario, the population size for the order dispatching task is therefore also changing over time.

In this paper, we address the two problems of variable action sets and population size by extending the actor-critic policy gradient methods. Our methods tackle the order dispatching problem within the framework of centralized training with decentralized execution. The critic is provided with the information from other agents to incorporate the peer information, while each actor behaves independently with local information only. Our major contributions are listed as follows:

- We propose the MARL-based *independent order dispatching* (IOD) algorithm to solve the issue of changing action set, where we use the vectorized features of each order as the

network input to generate a set of ranking values to choose an order.
- We propose the *cooperative order dispatching* (COD) algorithm, an extension of IOD to resolve the variable population size, where we adopt the mean field approximation to transform the interactions between agents to the pairwise interaction between an agent and the average response from a sub-population in the neighborhood.
- We provide the convergence proof of mean field reinforcement learning algorithms with function approximations to justify our algorithm in theory.

Experiments on the large-scale simulator show that compared with variant multi-agent learning benchmarks, the mean field multi-agent reinforcement learning algorithm gives the best performance towards the accumulated driver income, the order response rate, and the average pick-up distance. Besides the state of the art performance, our solution also enjoys the advantage of distributed execution, which has lower latency and is easily adaptable to be deployed in the real world application.

2 METHOD

In this section, we first illustrate our definition of order dispatching as a Markov game, and discuss two challenges when applying MARL to this game. We then propose a MARL approach with the independent Q-learning, namely the independent order dispatching algorithm (IOD), to solve this game. By extending IOD with mean field approximations, which capture dynamic demand-supply variations by propagating many local interactions between agents and the environment, we finally propose the cooperative order dispatching algorithm (COD).

2.1 Order Dispatching as a Markov Game

2.1.1 Game Settings. We model the order dispatching task by a Partially Observable Markov Decision Process (POMDP) [20] in a fully cooperative setting, defined by a tuple $\Gamma = \langle \mathcal{S}, \mathcal{P}, \mathcal{A}, \mathcal{R}, \mathcal{O}, N, \gamma \rangle$, where $\mathcal{S}, \mathcal{P}, \mathcal{A}, \mathcal{R}, \mathcal{O}, N, \gamma$ are the sets of states, transition probability functions, sets of joint actions, reward functions, sets of private observations, number of agents, and a discount factor respectively. Given two sets $\mathcal{X}$ and $\mathcal{Y}$, we use $\mathcal{X} \times \mathcal{Y}$ to denote the Cartesian product of $\mathcal{X}$ and $\mathcal{Y}$, i.e., $\mathcal{X} \times \mathcal{Y} = \{(x, y) | x \in \mathcal{X}, y \in \mathcal{Y}\}$. The definitions are given as follows:

- N: N homogeneous agents identified by $i \in \mathcal{I} \equiv \{1, ..., N\}$ are defined as the active drivers in the environment. As the drivers can switch between online and offline via a random process, the number of agents N could change over time.
- $\mathcal{S}, \mathcal{O}$: At each time step t, agent i draws private observations $o_i^t \in \mathcal{O}$ correlated with the true environment state $s^t \in \mathcal{S}$ according to the observation function $\mathcal{S} \times \mathcal{I} \to \mathcal{O}$. The initial state of the environment is determined by a distribution $p_1(s^1) : \mathcal{S} \to [0, 1]$. A typical environmental state s includes the order and driver distribution, the global timestamp, and other environment dynamics (traffic congestion, weather conditions, etc.). In this work, we define the observation for each agent i with three components: agent i's location loc_i, the timestamp t, and an on-trip flag to show if agent i is available to take new orders.
- $\mathcal{P}, \mathcal{A}$: In the order dispatching task, each driver can only take active orders within its neighborhood (inside a given radius, as illustrated in Fig. 1b). Hence, agent i's action set $\mathcal{A}_i$ is defined as its own active order pool based on the observation o_i. Each action candidate $a_{i,m} \in \mathcal{A}_i$ is parameterized by the normalized vector representation of corresponding order's origin $ori_{i,m}$ and destination $des_{i,m}$, i.e., $a_{i,m} \equiv [ori_{i,m}, des_{i,m}]$. At time step t, each agent takes an action $a_i^t \in \mathcal{A}_i$, forming a set of joint driver-order pair $\mathbf{a}^t = \mathcal{A}_1 \times ... \times \mathcal{A}_N$, which induces a transition in the environment according to the state transition function

$$\mathcal{P}(s^{t+1} | s^t, \mathbf{a}^t) : \mathcal{S} \times \mathcal{A}_1 \times ... \times \mathcal{A}_N \to \mathcal{S}. \tag{1}$$

  For simplicity, we assume no order cancellations and changes during the trip, i.e., a_i^t keeps unchanged if agent i is on its way to the destination.
- $\mathcal{R}$: Each agent i obtains rewards r_i^t by a reward function

$$\mathcal{R}_i(s^t, \mathbf{a}^t) : \mathcal{S} \times \mathcal{A}_1 \times ... \times \mathcal{A}_N \to \mathbb{R}. \tag{2}$$

  As described in Section 1, we want to maximize the total income by considering both the charge of each order and the potential opportunity of the destination. The reward is then defined as the combination of driver i's own income ${}^0r$ from its order choice and the order destination potential ${}^1r$, which is determined by all agents' behaviors in the environment. Considering the credit-assignment problem [2] arises in MARL with many agents, i.e., the contribution of an agent's behavior is drowned by the noise of all the other agents' impact on the reward function, we set each driver's own income instead of the total income of all drivers as the reward. To encourage cooperation between agents and avoid agents' being selfish and greedy, we use the order destination's demand-supply gap as a constraint ${}^1r$ on the behavior of agents. Precisely, we compare the demand-supply status between the order origin and destination, and encourage the driver to choose the order destination with a larger demand-supply gap. The order destination potential (DP) is defined as

$$DP = \#DD - \#DS, \tag{3}$$

  where #DD and #DS is the demand and the supply of the destination respectively. We consider the DP only if the number of orders are larger than that of drivers at the origin. If the order destination has more drivers than orders, we penalize this order with the demand-supply gap at the destination, and vice versa.

  To provide better customer experience, we also add the pick-up distance ${}^2r$ as a regularizer. The ratio of the DP and the pick-up distance to order price are defined as ${}^1\alpha$ and ${}^2\alpha$ respectively, i.e., $r_i^t = {}^0r_i^t + {}^1\alpha^1 r_i^t + {}^2\alpha^2 r_i^t$. We typically choose the regularization ratio to scale different reward terms into approximately the same range, although in practice a grid-search could be used to get better performance. The effectiveness of our reward function setting is empirically verified in Section 3.1.3.
- γ: Each agent i aims to maximize its total discounted reward

$$G_i^t = \sum_{k=t}^{\infty} \gamma^{k-t} r_i^k \tag{4}$$

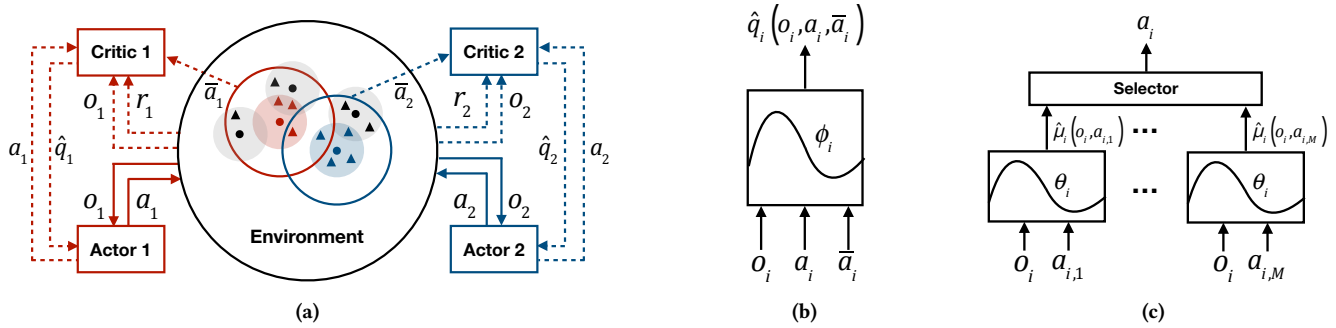


Figure 2: Overview of our COD approach. (a) The information flow between RL agents and the environment (shown with two agents). The centralized authority is only needed during the training stage (dashed arrows) to gather the average response $\bar{a}$; agents could behave independently during the execution stage (solid arrows), thus being robust to the "single point of failure". (b) and (c) Architectures of the critic and the actor.

from time step t onwards, where $\gamma \in [0, 1]$ is the discount factor.

We denote joint quantities over agents in bold, and joint quantities over agents other than a given agent i with the subscript $-i$, e.g., $\boldsymbol{a} \equiv (\boldsymbol{a}^t_{-i}, a^t_i)$. To stabilize the training process, we maintain an experience replay buffer $\mathcal{D}$ containing tuples $(\boldsymbol{o}^t, \boldsymbol{a}^t, \boldsymbol{r}^t, \boldsymbol{o}^{t+1})$ as described in [26].

2.1.2 Dynamic of the Action Set Elements. In the order dispatching problem, an action is defined as an active order within a given radius from the agent. Hence, the content and size of the active order pool for each driver are changing with both the location and time, i.e., the action set for an agent in the order dispatching MDP is changing throughout the training and execution process. This aspect of order dispatching refrains us from using the Q-table to log the Q-value because of the potentially infinite size of the action set. On the other hand, a typical policy network for stochastic actions makes use of a softmax output layer to produce a set of probabilities of choosing each action among a fixed set of action candidates, thus is unable to fit into the order dispatching problem.

2.1.3 Dynamic of the Population Size. To overcome the non-stationarity of the multi-agent environment, Lowe et al. [21] uses Q-Learning to approximate the discounted reward G^t_i, and rewrites the gradient of the expected return for agent i following a deterministic policy μ_i (parameterized by θ_i) as

$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{\boldsymbol{o},\boldsymbol{a}\sim\mathcal{D}}[\nabla_{\theta_i}\mu_i(a_i|o_i)\nabla_{a_i}Q_i(o_i,\boldsymbol{a})]. \tag{5}$$

Here $Q_i(o_i, \boldsymbol{a})$ is a centralized action-value function that takes as input the observation o_i of agent i and the joint actions $\boldsymbol{a}$ of all agents, and outputs the Q-value for agent i. However, as off-line drivers cannot participate in the order dispatching procedure, the number of agents N in the environment is changing over time. Also, in a typical order dispatching task, which involves thousands of agents, the high dynamics of interactions between a large number of agents is intractable. Thus, a naive concatenation of all other agents' actions cannot form a valid input for the value network and is not applicable to the order dispatching task.

2.2 Independent Order Dispatching

To solve the order dispatching MDP, we first propose the independent order dispatching algorithm (IOD), a straightforward MARL approach with the independent Q-learning. We provide each learner with the actor-critic model, which is a popular form of policy gradient (PG) method. For each agent i, PG works by directly adjusting the parameters θ_i of the policy μ_i to maximize the objective $J(\theta_i) = \mathbb{E}_{\boldsymbol{o},\boldsymbol{a}\sim\mathcal{D}}\left[G^t_i\right]$ by taking steps in the direction of $\nabla_{\theta_i} J(\theta_i)$. In MARL, independent actor-critic uses the action-value function $Q_i(o_i, a_i)$ to approximate the discounted reward G^t_i by Q-Learning [39]. Here we use temporal-difference learning [32] to approximate the true $Q_i(o_i, a_i)$, leading to a variety of actor-critic algorithms with $Q_i(o_i, a_i)$ called as the critic and μ_i called as the actor.

To solve the problem of variable action sets, we use a policy network with both observation and action embeddings as the input, derived from the in-action approximation methods (shown in Fig. 2b). As illustrated in Fig. 2c, we use a deterministic policy μ_{θ_i} (denoted by $\mu_{\theta_i} : \mathcal{O} \times \mathcal{A} \to \mathbb{R}^*$, abbreviated as μ_i) to generate ranking values of each observation-action pair $(o_i, a_{i,m})$ for each of the M_i candidates within agent i's action set $\mathcal{A}_i$. To choose an action a_i, these values are then fed into a Boltzmann softmax selector

$$\pi_i(a_{i,j}|o_i) = \frac{\exp(\beta\mu_i(o_i, a_{i,j}))}{\sum_{m=0}^{M_i}\exp(\beta\mu_i(o_i, a_{i,m}))}, \quad \text{for } j = 1, \ldots, M_i \tag{6}$$

where β is the inverse temperature to control the exploration rate. Note that a typical policy network with out-action approximation is equivalent to this approach, where we can use an M_i-dimension one-hot vector as the embedding to feed into the policy network. The main difference between these approaches is the execution efficiency, as we need exactly M_i forward passes in a single execution step. Meanwhile, using order features naturally provide us with an informative form of embeddings. As each order is parameterized by the concatenation of the normalized vector representation of its origin $ori_{i,m}$ and destination $des_{i,m}$, i.e., $a_{i,m} \equiv [ori_{i,m}, des_{i,m}]$. Similar orders will be close to each other in the vector space and produce similar outputs from the policy network, which improves the generalization ability of the algorithm.

Algorithm 1 Cooperative Order Dispatching (COD)

Initialize Q_{ϕ_i}, $Q^-_{\phi_i}$, μ_{θ_i}, and $\mu^-_{\theta_i}$ for all $i \in \{1, \dots, N\}$
while training not finished **do**
  For each agent i, sample action a_i using the Boltzmann softmax selector $\pi_i(a_i|o_i)$ from Eq. (6)
  Take the joint action $\boldsymbol{a} = [a_1, \dots, a_N]$ and observe the reward $\boldsymbol{r} = [r_1, \dots, r_N]$ and the next observations $\boldsymbol{o'}$
  Compute the new mean action $\bar{\boldsymbol{a}} = [\bar{a}_1, \dots, \bar{a}_N]$
  Store $\langle \boldsymbol{o}, \boldsymbol{a}, \boldsymbol{r}, \boldsymbol{o'}, \bar{\boldsymbol{a}} \rangle$ in replay buffer $\mathcal{D}$
  for $i = 1$ to N **do**
    Sample K experiences $\langle \boldsymbol{o}, \boldsymbol{a}, \boldsymbol{r}, \boldsymbol{o'}, \bar{\boldsymbol{a}} \rangle$ from $\mathcal{D}$
    Update the critic by minimizing the loss from Eq. (9)
    Update the actor using the policy gradient as Eq. (11)
  end for
  Update the parameters of the target networks for each agent i with updating rates τ_ϕ and τ_θ:

$$\phi^-_i \leftarrow \tau_\phi \phi_i + (1 - \tau_\phi)\phi^-_i$$
$$\theta^-_i \leftarrow \tau_\theta \theta_i + (1 - \tau_\theta)\theta^-_i$$

end while

In IOD, each critic takes input the observation embedding o_i by combining agent i's location and the timestamp t. The action embedding is built with the vector representation of the order destination and the distance between the driver location and the order origin. The critic is a DQN [26] using neural network function approximations to learn the action-value function Q_ϕ (parameterized by ϕ, abbreviated as Q_i for each agent i) by minimizing the loss

$$\mathcal{L}(\phi_i) = \mathbb{E}_{\boldsymbol{o}, \boldsymbol{a}, \boldsymbol{r}, \boldsymbol{o'}}[(y - Q_i(o_i, a_i))^2], \quad y = r_i + \gamma\, Q^-_i(o'_i, \mu^-_i(o'_i, a'_i)), \tag{7}$$

where Q^-_i is the target network for the action-value function, and μ^-_i is the target network for the deterministic policy. These earlier snapshots of parameters are periodically updated with the most recent network weights and help increase learning stability by decorrelating predicted and target Q-values and deterministic policy values.

Following Silver et al. [31], the gradient of the expected return for agent i following a deterministic policy μ_i is

$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{\boldsymbol{o}, \boldsymbol{a} \sim \mathcal{D}}[\nabla_{\theta_i} \mu_i(o_i, a_i) \nabla_{a_i} Q_i(o_i, a_i)]. \tag{8}$$

Here $Q_i(o_i, a_i)$ is an action-value function that takes as input the observation o_i and action a_i, and outputs the Q-value for agent i.

2.3 Cooperative Order Dispatching with Mean Field Approximation

To fully condition on other agents' policy in the environment with variable population size, we propose to integrate our IOD algorithm with mean field approximations, following the Mean Field Reinforcement Learning (MFRL) [43]. MFRL addresses the scalability issue in the multi-agent reinforcement learning with a large number of agents, where the interactions are approximated pairwise by the interaction between an agent and the average response $\bar{a}$ from a sub-population in the neighborhood. As this pairwise approximation shadows the exact size of interacting counterparts, the use of mean field approximation can help us model other agents' policies directly in the environment with variable population sizes.

In the order dispatching task, agents are interacting with each other by choosing order destinations with a high demand to optimize the demand-supply gap. As illustrated in Fig. 1b, the range of the neighborhood is then defined as twice the length of the order receiving radius, because agents within this area have intersections between their action sets and interact with each other. The average response $\bar{a}_i$ is therefore defined as the number of drivers arriving at the same neighborhood as agent i, divided by the number of available orders for agent i. For example, when agent i finishes the order and arrives at the neighborhood in Fig. 1b (the central agent), the average response $\bar{a}_i$ is 2/3, as there are two agents within the neighborhood area and three available orders.

The introduction of mean field approximations enables agents to learn with the awareness of interacting counterparts, thus helping to improve the training stability and robustness of agents after training for the order dispatching task. Note that the average response $\bar{a}$ only serves for the model update; thus the centralized authority is only needed during the training stage. During the execution stage, agents could behave in a fully distributed manner, thus being robust to the "single point of failure".

We propose the cooperative order dispatching algorithm (COD) as illustrated in Fig. 2a, and present the pseudo code for COD in Algorithm 1. Each critic is trained by minimizing the loss

$$\mathcal{L}(\phi_i) = \mathbb{E}_{\boldsymbol{o}, \boldsymbol{a}, \boldsymbol{r}, \boldsymbol{o'}}[(r_i + \gamma\, v^{\mathsf{MF}}_{i^-}(o'_i) - Q_i(o_i, (\bar{a}_i, a_i)))^2], \tag{9}$$

where $v^{\mathsf{MF}}_{i^-}(o'_i)$ is the mean field value function for the target network Q^-_i and μ^-_i (shown as the Boltzmann selector π^-_i from Eq. 6)

$$v^{\mathsf{MF}}_{i^-}(o'_i) = \sum\nolimits_{a'_i} \pi^-_i(a'_i|o'_i) \mathbb{E}_{\bar{a}'_i(\boldsymbol{a}'_{-i}) \sim \boldsymbol{\pi}^-_{-i}}[Q^-_i(o'_i, (\bar{a}'_i, a'_i))], \tag{10}$$

and $\bar{a}_i$ is the average response within agent i's neighborhood. The actor of COD learns the optimal policy by using the policy gradient:

$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{\boldsymbol{o}, \boldsymbol{a} \sim \mathcal{D}}[\nabla_{\theta_i} \mu_i(o_i, a_i) \nabla_{a_i} Q_i(o_i, (\bar{a}_i, a_i))]. \tag{11}$$

In the current decision process, active agents sharing duplicated order receiving areas (e.g., the central and upper agent in Fig. 1b) might select the same order following their own strategy. Such collisions could lead to invalid order assignment and force both drivers and customers to wait for a certain period, which equals to the time interval between each dispatching iteration. Observe that the time interval of decision-making also influences the performance of dispatching; a too long interval will affect the passenger experience, while a too short interval without enough order candidates is not conducive to the decision making. To solve this problem, our approach works in a fully distributed manner with asynchronous dispatching strategy, allowing agents have different decision time interval for individual states, i.e., agents assigned with invalid orders could immediately re-choose a new order from updated candidates pool.

To theoretically support the efficacy of our proposed COD algorithm, we provide the convergence proof of MFRL with function approximations as shown below.

2.4 Convergence of Mean Field Reinforcement Learning with Function Approximations

Inspired by the previous proof of MFRL convergence in a tabular Q-function setting [43], we further develop the proof towards the converge when the Q-function is represented by other function approximators. In addition to the Markov Game setting in Section 2.1.1, let $\mathcal{Q} = \{\boldsymbol{Q}_\phi\}$ be a family of real-valued functions defined on $\mathcal{S} \times \mathcal{A} \times \bar{\mathcal{A}}$, where $\bar{\mathcal{A}}$ is the action space for the mean actions computed from the neighbors. For simplicity, we assume the environment is a fully observable MDP Γ, i.e., each agent $i \in \mathcal{I} \equiv \{1, ..., N\}$ can observe the global state s instead of the local observation o_i.

Assuming that the function class is linearly parameterized, for each agent i, the Q-function can be expressed as the linear span of a fixed set of P linearly independent functions $\omega_i^p : \mathcal{S} \times \mathcal{A} \times \bar{\mathcal{A}} \to \mathbb{R}$. Given the parameter vector $\phi_i \in \mathbb{R}^P$, the function Q_{ϕ_i} (abbreviated as Q_i) is thus defined as

$$Q_i(s, (\bar{a}_i, a_i)) = \sum_{p=1}^{P} \omega_i^p(s, (\bar{a}_i, a_i))\phi_i(p) = \omega_i(s, (\bar{a}_i, a_i))^\top \phi_i. \quad (12)$$

In the function approximation setting, we apply the update rules:

$$\phi_i' = \phi_i + \alpha\Delta\nabla_{\phi_i} Q_i(s, (\bar{a}_i, a_i)) = \phi_i + \alpha\Delta\omega_i(s, (\bar{a}_i, a_i)), \quad (13)$$

where Δ is the temporal difference:

$$\begin{aligned}\Delta &= r_i + \gamma\, v_i^{\mathsf{MF}}(s') - Q_i(s, (\bar{a}_i, a_i)) \\ &= r_i + \gamma\, \mathbb{E}_{\boldsymbol{a}\sim\boldsymbol{\pi}^{\mathsf{MF}}}[Q_i(s', (\bar{a}_i, a_i))] - Q_i(s, (\bar{a}_i, a_i)). \end{aligned} \quad (14)$$

Our goal is to derive the parameter vector $\boldsymbol{\phi} = \{\phi_i\}$ such that $\boldsymbol{\omega}^\top\boldsymbol{\phi}$ approximates the (local) Nash Q-values. Under the main assumptions and the lemma as introduced below, Yang et al. [43] proved that the policy $\boldsymbol{\pi}$ is $\frac{K}{2T}$ Lipschitz continuous with respect to ϕ, where $T = 1/\beta$ and $K \geq 0$ is the upper bound of the observed reward.

Assumption 1. *Each state-action pair is visited infinitely often, and the reward is bounded by some constant K.*

Assumption 2. *Agent's policy is Greedy in the Limit with Infinite Exploration (GLIE). In the case with the Boltzmann policy, the policy becomes greedy* w.r.t. *the Q-function in the limit as the temperature decays asymptotically to zero.*

Assumption 3. *For each stage game $[Q_t^1(s), ..., Q_t^N(s)]$ at time t and in state s in training, for all t, s, $j \in \{1, \ldots, N\}$, the Nash equilibrium $\boldsymbol{\pi}_* = [\pi_*^1, \ldots, \pi_*^N]$ is recognized either as 1) the* global optimum *or 2) a* saddle point *expressed as:*

1. $\mathbb{E}_{\boldsymbol{\pi}_*}[Q_t^j(s)] \geq \mathbb{E}_{\boldsymbol{\pi}}[Q_t^j(s)],\ \forall\boldsymbol{\pi} \in \Omega\Big(\prod_k \mathcal{A}^k\Big)$;
2. $\mathbb{E}_{\boldsymbol{\pi}_*}[Q_t^j(s)] \geq \mathbb{E}_{\pi^j}\mathbb{E}_{\boldsymbol{\pi}_*^{-j}}[Q_t^j(s)],\ \forall\pi^j \in \Omega\big(\mathcal{A}^j\big)$ *and* $\mathbb{E}_{\boldsymbol{\pi}_*}[Q_t^j(s)] \leq \mathbb{E}_{\pi_*^j}\mathbb{E}_{\boldsymbol{\pi}^{-j}}[Q_t^j(s)],\ \forall\boldsymbol{\pi}^{-j} \in \Omega\Big(\prod_{k\neq j} \mathcal{A}^k\Big)$.

Lemma 2.1. *The random process $\{\Delta_t\}$ defined in $\mathbb{R}$ as*

$$\Delta_{t+1}(x) = (1 - \alpha_t(x))\Delta_t(x) + \alpha_t(x)F_t(x)$$

converges to zero with probability 1 (w.p.1) *when*

1. $0 \leq \alpha_t(x) \leq 1$, $\sum_t \alpha_t(x) = \infty$, $\sum_t \alpha_t^2(x) < \infty$;
2. $x \in \mathcal{X}$, *the set of possible states, and* $|\mathcal{X}| < \infty$;
3. $\|\mathbb{E}[F_t(x)|\mathcal{F}_t]\|_W \leq \gamma\|\Delta_t\|_W + c_t$, *where* $\gamma \in [0, 1)$ *and* c_t *converges to zero* w.p.1;
4. $\mathbf{var}[F_t(x)|\mathcal{F}_t] \leq K(1 + \|\Delta_t\|_W^2)$ *with constant* $K > 0$.

Here $\mathcal{F}_t$ denotes the filtration of an increasing sequence of σ-fields including the history of processes; $\alpha_t, \Delta_t, F_t \in \mathcal{F}_t$ and $\|\cdot\|_W$ is a weighted maximum norm [6].

Proof. See Theorem 1 in Jaakkola et al. [13] and Corollary 5 in Szepesvári and Littman [35] for detailed derivation. We include it here to stay self-contained. □

In contrast to the previous work Yang et al. [43], we establish convergence of Eq. (13) by adopting an ordinary differentiable equation (ODE) with a globally asymptotically stable equilibrium point where the trajectories closely follow, following the framework of the convergence proof of single-agent Q-learning with function approximation [25].

Theorem 2.2. *Given the MDP Γ, $\{\boldsymbol{\omega}_p, p = 1, ..., P\}$, and the learning policy $\boldsymbol{\pi}$ that is $\frac{K}{2T}$ Lipschitz continuous with respect to ϕ, if the Assumptions 1, 2 & 3, and Lemma 2.1's first and second conditions are met, then there exists C_0 such that the algorithm in Eq.* (13) *converges w.p.1 if $\frac{K}{2T} < C_0$.*

Proof. We first re-write the Eq. (13) as on ODE:

$$\begin{aligned}\frac{d\boldsymbol{\phi}}{dt} &= \mathbb{E}_{\boldsymbol{\phi}}\left[\boldsymbol{\omega}_s^\top\left(\boldsymbol{r}(s, \boldsymbol{a}, s') + \gamma\boldsymbol{\omega}_{s'}^\top\boldsymbol{\phi} - \boldsymbol{\omega}_s^\top\boldsymbol{\phi}\right)\right] \\ &= \mathbb{E}_{\boldsymbol{\phi}}\left[\boldsymbol{\omega}_s^\top(\gamma\boldsymbol{\omega}_{s'}^\top - \boldsymbol{\omega}_s^\top)\right]\boldsymbol{\phi} + \mathbb{E}_{\boldsymbol{\phi}}\left[\boldsymbol{\omega}_s^\top(\boldsymbol{r}(s, \boldsymbol{a}, s'))\right] \\ &= \boldsymbol{A}_\phi\boldsymbol{\phi} + \boldsymbol{b}_\phi \end{aligned} \quad (15)$$

Notice that we use a vector for considering the updating rule for the Q-function of each agent. We can easily know that necessity condition of the equilibrium is $\boldsymbol{\phi}^* = \boldsymbol{A}_{\phi^*}^{-1}\boldsymbol{b}_{\phi^*}$. The existence of the such equilibrium has been restricted in the scenario that meets Assumption 3. Yang et al. [43] proved that under the Assumption 3, the existing equilibrium, either in the form of a global equilibrium or in the form of a saddle-point equilibrium, is unique.

Let $\tilde{\boldsymbol{\phi}} = \boldsymbol{\phi}^t - \boldsymbol{\phi}^*$, we have:

$$\begin{aligned}\frac{d}{dt}||\tilde{\boldsymbol{\phi}}||_2 &= 2\boldsymbol{\phi}\cdot\frac{d\boldsymbol{\phi}}{dt} - 2\boldsymbol{\phi}^*\cdot\frac{d\boldsymbol{\phi}}{dt} \\ &= 2\tilde{\boldsymbol{\phi}}^\top(\boldsymbol{A}_\phi\boldsymbol{\phi} + \boldsymbol{b}_\phi - \boldsymbol{A}_{\phi^*}\boldsymbol{\phi}^* - \boldsymbol{b}_{\phi^*}) \\ &= 2\tilde{\boldsymbol{\phi}}^\top(\boldsymbol{A}_{\phi^*}\boldsymbol{\phi} - \boldsymbol{A}_{\phi^*}\boldsymbol{\phi} + \boldsymbol{A}_\phi\boldsymbol{\phi} + \boldsymbol{b}_\phi - \boldsymbol{A}_{\phi^*}\boldsymbol{\phi}^* - \boldsymbol{b}_{\phi^*}) \\ &= 2\tilde{\boldsymbol{\phi}}^\top\boldsymbol{A}_{\phi^*}\tilde{\boldsymbol{\phi}} + 2\tilde{\boldsymbol{\phi}}^\top(\boldsymbol{A}_\phi - \boldsymbol{A}_{\phi^*})\boldsymbol{\phi} + 2\tilde{\boldsymbol{\phi}}^\top(\boldsymbol{b}_\phi - \boldsymbol{b}_{\phi^*}) \\ &\leq 2\tilde{\boldsymbol{\phi}}^\top\left(\boldsymbol{A}_{\phi^*} + \sup_{\boldsymbol{\phi}}||\boldsymbol{A}_\phi - \boldsymbol{A}_{\phi^*}||_2 + \sup_{\boldsymbol{\phi}}\frac{||\boldsymbol{b}_\phi - \boldsymbol{b}_\phi^*||_2}{||\boldsymbol{\phi} - \boldsymbol{\phi}^*||_2}\right)\tilde{\boldsymbol{\phi}}. \end{aligned} \quad (16)$$

As we know that the policy $\boldsymbol{\pi}^t$ is Lipschitz continuous w.r.t ϕ^t, this implies that $\boldsymbol{A}_\phi$ and $\boldsymbol{b}_\phi$ are also Lipschitz continuous w.r.t to ϕ. In other words, if $\frac{K}{2T} \leq C_0$ is sufficiently small and close to zero, then the norm term of $\left(\sup_{\boldsymbol{\phi}}||\boldsymbol{A}_\phi - \boldsymbol{A}_{\phi^*}||_2 + \sup_{\boldsymbol{\phi}}\frac{||\boldsymbol{b}_\phi - \boldsymbol{b}_\phi^*||_2}{||\boldsymbol{\phi} - \boldsymbol{\phi}^*||_2}\right)$ goes to zero. Considering near the equilibrium point ϕ^*, $\boldsymbol{A}_{\phi^*}$ is a negative definite matrix, the Eq. (16) tends to be negative definite as well, so the ODE in Eq.(15) is globally asymptotically stable and the conclusion of the theorem follows. □

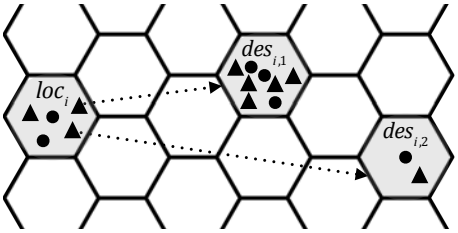


Figure 3: Illustration of the grid-based simulator. All location features are represented by the corresponding grid ID. The order origin $ori_{i,m}$ is the same as the driver location loc_i.

While in practice, we might break the linear condition by the use of nonlinear activation functions, the Lipschitz continuity will still hold as long as the nonlinear add-on is limited to a small scale.

3 EXPERIMENT

To support the training and evaluation of our MARL algorithm, we adopt two simulators with a grid-based map and a coordinate-based map respectively. The main difference between these two simulators is the design of the driver pick-up module, i.e., the process of a driver reaching the origin of the assigned order. In the grid-based simulator (introduced by Lin et al. [19]), the location state for each driver and order is represented by a grid ID. Hence, the exact coordinate for each instance inside a grid is shadowed by the state representation. This simplified setting ensures there will be no pick-up distance (or arriving time) difference inside a grid; it also brings an assumption of no cancellations before driver pick-up. Whereas in the coordinate-based simulator, the location of each driver and order instance is represented by a two-value vector from the Geographic Coordinate System, and the cancellation before pick-up is also taken into account. In this setting, taking an order within an appropriate pick-up distance is crucial to each driver, as the order may be canceled if the driver takes too long time to arrive. We present experiment details of the grid-based simulator in Section 3.1, and the coordinate-based simulator in Section 3.2.

3.1 Grid-based Experiment

3.1.1 Environment Setting. In the grid-based simulator, the city is covered by a hexagonal grid-world as illustrated in Fig. 3. At each simulation time step t, the simulator provides an observation $\boldsymbol{o}^t$ with a set of active drivers and a set of available orders. Each order feature includes the origin grid ID and the destination grid ID, while each driver has the grid ID as the location feature loc_i. Drivers are regarded as homogeneous and can switch between online (active) and offline via a random process learned from the history data. As the travel distance between neighboring grids is approximately 1.2 kilometers and the time step interval $t_\triangle$ is 10 minutes, we assume that drivers will not move to other grids before taking a new order, and define the order receiving area and the neighborhood as the grid where the agent stays. The order dispatching algorithm then generates an optimal list of driver-order pairs $\boldsymbol{a}^t$ for the current policy, where a_i is an available order $a_{i,m}$ selected from the order candidate pool $\mathcal{A}_i$. In the grid-based setting, the origin of each order is already embedded as the location feature loc_i in o_i, thus $a_{i,m}$ is parameterized by the destination grid ID $des_{i,m}$. After receiving the driver-order pairs from the algorithm, the simulator will then return a new observation $\boldsymbol{o}^{t+1}$ and a list of order fees. Stepping on this new observation, the order dispatching algorithm will calculate a set of rewards r^t for each agent, store the record $(\boldsymbol{o}^t, \boldsymbol{a}^t, \boldsymbol{r}^t, \boldsymbol{o}^{t+1})$ to replay buffer, and update the network parameters with respect to a batch of samples from replay buffer.

The data source of this simulator (provided by DiDi Chuxing) includes order information and trajectories of vehicles in three weeks. Available orders are generated by bootstrapping from real orders occurred in the same period during the day given a bootstrapping ratio p_{sample}. More concretely, suppose the simulation time step interval is $t_\triangle$, at each simulation time step t, we randomly sample p_{sample} orders with replacement from real orders happened between $t_\triangle * t$ to $t_\triangle * (t+1)$. Also, drivers are set between online and offline following a distribution learned from real data using a maximum likelihood estimation. On average, the simulator has $7K$ drivers and $136K$ dispatching events per time step.

The effectiveness of the grid-based simulator is evaluated by Lin et al. [19] using the calibration against the real data regarding the most important performance measurement: the accumulated driver income (ADI). The coefficient of determination r^2 between simulated ADI and real ADI is 0.9331 and the Pearson correlation is 0.9853 with p-value $p < 0.00001$.

3.1.2 Model Setting. We use the grid-based simulator to compare the performance of following methods.

- **Random (RAN)**: The random dispatching algorithm considers no additional information. It only assigns all active drivers with an available order at each time step.
- **Response-based (RES)**: This response-based method aims to achieve higher order response rate by assigning drivers to short duration orders. During each time step, all available orders starting from the same grid will be sorted by the estimated trip time. Multiple orders with the same expected duration will be further sorted by the order price to balance the performance.
- **Revenue-based (REV)**: The revenue-based algorithm focuses on a higher ADI. Orders with higher prices will be given priority to get dispatched first. Following the similar principle as described above, orders with shorter estimated trip time will be assigned first if multiple orders have the same price.
- **IOD**: The independent order dispatching algorithm as described in Section 2.2. The action-value function approximation (i.e., the Q-network) is parameterized by an MLP with four hidden layers (512, 256, 128, 64) and the policy network is parameterized by an MLP with three hidden layers (256, 128, 64). We use the ReLU [27] activation between hidden layers, and transform the final linear output of Q-network and policy network with ReLU and sigmoid function respectively. To find an optimal parameter setting, we use the Adam Optimizer [14] with a learning rate of 0.0001 for the critic and 0.001 for the actor. The discounted factor γ is 0.95, and the batch size is 2048. We update the network parameters after every $3K$ samples are added to the replay buffer (capacity $5 * 10^5$). We use a Boltzmann softmax selector for all MARL

Table 1: Performance comparison regarding the normalized Accumulated Driver Income (ADI) on the test set with respect to the performance of RAN.

p_{sample}	100%	50%	10%
RES	−4.60 ± 0.01	−4.65 ± 0.04	−4.71 ± 0.03
REV	+4.78 ± 0.00	+4.87 ± 0.02	+5.24 ± 0.07
IOD	+6.27 ± 0.01	+6.21 ± 0.03	+5.73 ± 0.07
COD	**+7.78 ± 0.01**	**+7.76 ± 0.03**	**+7.52 ± 0.14**

Table 2: Performance comparison regarding the order response rate (OOR) on the test set. The percentage difference shown for all methods is with respect to RAN.

p_{sample}	100%	50%	10%
RES	+3.15 ± 0.01	+3.11 ± 0.02	+2.59 ± 0.04
REV	−6.21 ± 0.00	−6.14 ± 0.03	−5.22 ± 0.06
IOD	+4.67 ± 0.01	+4.68 ± 0.03	+4.53 ± 0.04
COD	**+5.35 ± 0.00**	**+5.38 ± 0.03**	**+5.42 ± 0.09**

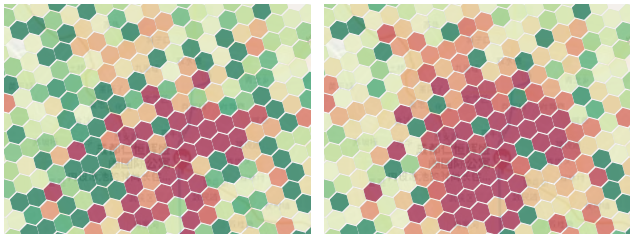
methods and set the initial temperature as 1.0, then gradually reduce the temperature until 0.01 to limit exploration.

- **COD**: The cooperative order dispatching algorithm with mean field approximation as described in Section 2.3. The network architecture is identical IOD, except a mean action is fed as an addition input to the critic network as illustrated in Fig. 2b.

As described in the reward setting in Section 2.1.1, we set the regularization ratio ${}^1\alpha = 0.01$ for DP and ${}^2\alpha = 0$ for the order waiting time penalty as we don't consider the pick-up distance in the grid-based experiment. Because of our homogeneous agent setting, all agents share the same Q-network and policy network for efficient training. During the execution in the real-world environment, each agent can keep its copy of policy parameters and receive updates periodically from a parameter server.

3.1.3 Result Analysis. For all learning methods, we run 20 episodes for training, store the trained model periodically, and conduct the evaluation on the stored model with the best training performance. The training set is generated by bootstrapping 50% of the original real orders unless specified otherwise. We use five random seeds for testing and present the averaged result. We compare the performance of different methods by three metrics, including the total income in a day (ADI), the order response rate (ORR), and the average order destination potential (ADP). ORR is the number of orders taken divided by the number of orders generated, and ADP is the sum of destination potential (as described in Section 2.1.1) of all orders divided by the number of orders taken.

Accumulated Driver Income. As shown in Table 1, the performance of COD largely surpasses all rule-based methods and IOD in ADI metric. RES suffers from lowest ADI among all methods due to its preference of short distance trips with lower average order value. On the other hand, REV aims to pick higher value orders with longer trip time, thus enjoying a higher ADI. However, both RES and REV cannot find a balance between getting higher income per order and taking more orders, while RAN falls into a sub-optimal trade-off without favoring either side. Instead, our proposed MARL methods (IOD and COD) achieve higher growths in terms of ADI by considering each order's price and the destination potential concurrently. Orders with relatively low destination potential will be less possible to get picked, thus avoiding harming ADI by preventing the driver from trapping in areas with very few future orders. By direct modeling other agents' policies and capturing the interaction between agents in the environment, the COD algorithm with mean field approximation gives the best performance among all comparing methods.



(a) COD **(b) REV**

Figure 4: An example of the demand-supply gap in the city center during peak hours. Grids with more drivers are shown in green (in red if opposite) and the gap is proportional to the shade of colors.

Table 3: Performance comparison in terms of the average destination potential (ADP) on the test set. The percentage difference shown for all methods is with respect to RAN.

p_{sample}	100%	50%	10%
RES	+6.36 ± 0.15	+7.40 ± 0.54	+9.97 ± 0.43
REV	−20.55 ± 0.16	−20.23 ± 0.52	−20.55 ± 1.58
IOD	+54.33 ± 0.09	+54.71 ± 0.20	+53.33 ± 0.54
COD	**+66.74 ± 0.05**	**+66.90 ± 0.29**	**+64.69 ± 0.61**

Order Response Rate. In Table 2 we compare the performance of different models in terms of the order response rate (OOR), which is the number of orders taken divided by the number of orders generated. RES has a higher OOR than the random strategy as it focuses on reducing the trip distance to take more orders. On the other hand, REV aims to pick higher value orders with longer trip time, leading to sacrifice on OOR. Although REV has a relatively higher ADI than other two rule-based methods, its lower OOR indicates a lower customer satisfaction rate, thus failing to meet the requirement of an optimal order dispatching algorithm. By considering both the average order value and the destination potential, IOD and COD achieve higher OOR as well. High-priced orders with low destination potential, i.e., a long trip to a suburban area, will be less possible to get picked, thus avoiding harming OOR when trying to take a high-priced order.

Average Order Destination Potential. To better present our method on optimizing the demand-supply gap, we list the average order destination potential (ADP) in Table 3. Note that all ADP values

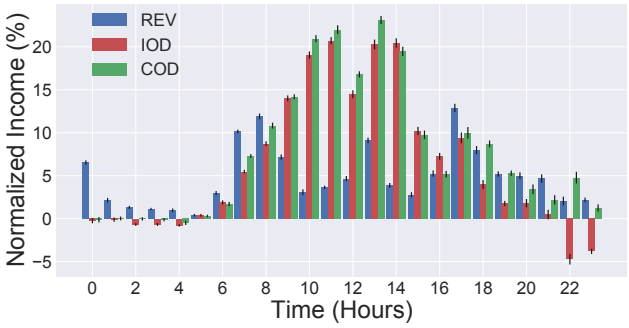


Figure 5: Normalized hourly income of REV, IOD and COD with respect to the average hourly income of RAN.

are negative, which indicates the supply still cannot fully satisfy the demand on average. However, IOD and COD largely alleviate the problem by dispatching orders to places with higher demand. As shown in Fig. 4, COD largely fills the demand-supply gap in the city center during peak hours, while REV fails to assign drivers to take orders with high-demand destination, thus leaving many grids with unserved orders.

Sequential Performance Analysis. To investigate the performance change of our proposed algorithms regarding the time of the day, Fig. 5 shows the normalized income of all algorithms with respect to the average hourly revenue of RAN. We eliminate RES from this comparison since it does not focus on getting a higher ADI and has relatively low performance. A positive value of the bar graph illustrates an increase in income compared to RAN, and vice versa. As shown in Fig. 5, MARL methods (IOD and COD) outperform RAN in most of the hours in a day (except for the late night period between 12 a.m. to 4 a.m., when very few orders are generated). During the peak hours in the morning and at night, MARL methods achieve a much higher hourly income than RAN. REV achieves a higher income than MARL methods between late night and early morning (12 a.m. to 8 a.m.) because of its aggressive strategy to take high-priced orders. However, this strategy ignores the destination quality of orders and assigns many drivers with orders to places with low demand, resulting in a significant income drop during the rest of the day (except for the evening peak hours when the possibility of encountering high-priced orders is large enough for REV to counteract the influence of the low response rate). On the other hand, IOD and COD earn significantly higher than RAN and REV for the rest of the day, possibly because of MARL methods' ability to recognize a better order in terms of both the order fee and the destination potential. As the order choice of MARL methods will prevent the agent from choosing a destination with lower potential, agents following MARL methods are thus enjoying more sustainable incomes. Also, COD outperforms IOD constantly, showing the effectiveness of explicitly conditioning on other agents' policies to capture the interaction between agents.

Effectiveness of Reward Settings. As described in Section 2.1.1, we use the destination potential as a reward function regularizer to encourage cooperation between agents. To show the effectiveness of this reward setting, we compare the ADI, OOR, and ADP of setting average income (AVE) and independent income (IND) as each agent's reward for IOD respectively. We also measure the performance of adding DP as a regularizer for both settings. For this experiment, we bootstrap 10% of the original real orders for the training and test set separately. As shown in Table 4, the performance of AVE in terms of all metrics is relatively lower than those of IND methods and RAN (even with DP added). This is possibly because of the credit assignment problem, where the agent's behavior is drowned by the noise of other agents' impact on the reward function. On the other hand, setting the individual income as the reward helps to distinguish each agent's contribution to the global objective from others, while adding DP as a regularizer further encourages the coordination between agents by arranging them to places with higher demand.

Table 4: Performance comparison of different reward settings applied to IOD with respect to RAN.

Method	ADI (%)	OOR (%)	ADP (%)
AVE	−0.36 ± 0.09	−0.14 ± 0.07	+1.11 ± 1.06
IND	+0.36 ± 0.10	+0.30 ± 0.09	−1.26 ± 0.87
AVE + DP	−0.26 ± 0.09	−0.10 ± 0.06	+1.17 ± 1.08
IND + DP	**+2.97 ± 0.02**	**+2.54 ± 0.03**	**+59.44 ± 1.33**

3.2 Coordinate-based Experiment

3.2.1 Environment Setting. As the real-world environment is coordinate-based rather than grid-based, we also conduct experiments on a more complex coordinate-based simulator provided by DiDi Chuxing. At each time step t, the coordinate-based simulator provides an observation $\boldsymbol{o}^t$ including a set of active drivers and a set of available orders. Each order feature includes the coordinate for the origin and the destination, while each driver has the coordinate as the location feature. The order dispatching algorithm works the same as described in Section 3.1.1. To better approximate the real-world scenario, this simulator also considers order cancelations, i.e., an order might be canceled during the pick-up process. This dynamic is controlled by a random variable which is positively related to the arriving time. The data resource of this simulator is based on historical dispatching events, including order generation events, driver logging on/off events and order fee estimation. During the training stage, the simulator will load five weekdays data with $350K$ dispatching events and generate $9K$ drivers. For evaluation, our model is applied on future days which are not used in the training phase.

3.2.2 Model Setting. We evaluate the performance of following MARL-based methods including IOD, COD, and a DQN variation of IOD (Q-IOD), i.e., without the policy network. We also compare these MARL methods with a centralized combinatorial optimization method based on the Hungarian algorithm (HOD). The HOD method focuses on minimizing the average arriving time (AAT) by setting the weight of each driver-order pair with the pick-up distance. For all MARL-based methods, the same network architecture setting as described in Section 3.1.2 is applied. Except that we use a mini-batch size of 200 because of the shorter simulation gap. The regularization ratio for pick-up distance is ${}^2\alpha = -0.1$ in this experiment.

Table 5: Performance comparison in terms of the ADI and the average arriving time (AAT) with respect to HOD.

Method	ADI1	ADI2	AAT
Q-IOD	−0.02%	−0.77%	+6.76%
IOD	+0.02%	−0.48%	+5.90%
COD	**+0.32%**	**+0.06%**	+5.49%

3.2.3 Result Analysis. We train all MARL methods for 400K iterations and apply the trained model in a test set (consists of three weekdays) for comparison. We compare different algorithms in terms of the total income in a day (ADI) and the average arriving time (AAT). ADI2 considers the cancellation while ADI1 doesn't. All the above metrics are normalized with respect to the result of HOD. As shown in Table 5, the result of COD largely outperforms Q-IOD and IOD in both ADI1 and ADI2, showing the effectiveness of direct modeling of other agents' policies in MARL. In addition, COD outperforms HOD in both ADI settings as well; this justifies the advantage of MARL algorithms that exploit the interaction between agents and the environment to maximize the cumulative reward. The performance improvement of ADI2 is smaller than that of ADI1 for MARL methods. This is possibly because that HOD works by minimizing the global pick-up distance and has a shorter waiting time. On the other hand, MARL methods only consider the pick-up distance as a regularization term, thus performing comparatively worse than HOD regarding AAT. As shown in Table 5, the AAT of all MARL methods are relatively longer than that of the combinatorial optimization method. However, as the absolute values of ADI are orders of magnitude higher than ATT, the increase in ATT is relatively minor and is thus tolerable in the order dispatching task. Also, MARL methods require no centralized control during execution, thus making the order dispatching system more robust to potential hardware or connectivity failures.

4 RELATED WORK

Order Dispatching. Several previous works addressed the order dispatching problem by either centralized or decentralized ruled-based approaches. Lee et al. [16] and Lee et al. [17] chose the pick-up distance (or time) as the basic criterion, and focused on finding the nearest option from a set of homogeneous drivers for each order on a first-come, first-served basis. These approaches only focus on the individual order pick-up distance; however, they do not account for the possibility of other orders in the waiting queue being more suitable for this driver. To improve global performance, Zhang et al. [45] proposed a novel model based on centralized combinatorial optimization by concurrently matching multiple driver-order pairs within a short time window. They considered each driver as heterogenous by taking the long-term behavior history and short-term interests into account. The above methods work with centralized control, which is prone to the potential "single point of failure" [23].

With the decentralized setting, Seow et al. [30] addressed the problem by grouping neighboring drivers and orders in a small multi-agent environment, and then simultaneously assigning orders to drivers within the group. Drivers in a group are considered as agents who conduct negotiations by several rounds of collaborative reasoning to decide whether to exchange current order assignments or not. This approach requires rounds of direct communications between agents, thus being limited to a local area with a small number of agents. Alshamsi and Abdallah [3] proposed an adaptive approach for the multi-agent scheduling system to enable negotiations between agents (drivers) to re-schedule allocated orders. They used a cycling transfer algorithm to evaluate each driver-order pair with multiple criteria, requiring a sophisticated design of feature selection and weighting scheme.

Different from rule-based approaches, which require additionally hand-crafted heuristics, we use a model-free RL agent to learn an optimal policy given the rewards and observations provided by the environment. Recent works by Xu et al. [41] and Wang et al. [38] proposed an RL-based dispatching algorithm to optimize resource utilization and user experience in a global and more farsighted view. However, they formulated the problem with the single-agent setting, which is unable to model the complex interactions between drivers and orders. On the contrary, our multi-agent setting follows the distributed nature of the peer-to-peer ridesharing problem, providing the dispatching system with the ability to capture the stochastic demand-supply dynamics in large-scale ridesharing scenarios. During the execution stage, agents will behave under the learned policy independently, thus being more robust to potential hardware or connectivity failures.

Multi-Agent Reinforcement Learning. One of the most straightforward approaches to adapt reinforcement learning in the multi-agent environment is to make each agent learn independently regardless of the other agents, such as independent Q-learning [36]. They, however, tend to fail in practice [24] because of the non-stationary nature of the multi-agent environment. Several approaches have been attempted to address this problem, including sharing the policy parameters [12], training the Q-function with other agent's policy parameters [37], or using importance sampling to learn from data gathered in a different environment [9]. The idea of centralized training with decentralized execution has been investigated by several works [8, 21, 29] recently for MARL using policy gradients [34], and deep neural networks function approximators, based on the actor-critic framework [15]. Agents within this paradigm learn a centralized Q-function augmented with actions of other agents as the *critic* during training stage, and use the learned policy (the *actor*) with local observations to guide their behaviors during execution. Most of these approaches limit their work to a small number of agents usually less than ten. To address the problem of the increasing input space and accumulated exploratory noises of other agents in large-scale MARL, Yang et al. [43] proposed a novel method by integrating MARL with mean field approximations and proved its convergence in a tabular Q-function setting. In this work, we further develop MFRL and prove its convergence when the Q-function is represented by function approximators.

5 CONCLUSION

In this paper, we proposed the multi-agent reinforcement learning solution to the order dispatching problem. Results on two large-scale simulation environments have shown that our proposed algorithms (COD and IOD) achieved (1) a higher ADI and OOR than three rule-based methods (RAN, RES, REV); (2) a higher ADI than

the combinatorial optimization method (HOD), with desirable properties of fully distributed execution; (3) lower supply-demand gap during the rush hours, which indicates the ability to reduce traffic congestion. We also provide the convergence proof of applying mean field theory to MARL with function approximations as the theoretical justification of our proposed algorithms. Furthermore, our MARL approaches could achieve fully decentralized execution by distributing the centralized trained policy to each vehicle through Vehicle-to-Network (V2N). For future work, we are working towards controlling ATT while maximizing the ADI with the proposed MARL framework. Another interesting and practical direction to develop is to use a heterogeneous agent setting with individual specific features, such as the personal preference and the distance from its own destination.

REFERENCES

[1] Khadige Abboud, Hassan Aboubakr Omar, and Weihua Zhuang. 2016. Interworking of DSRC and cellular network technologies for V2X communications: A survey. *IEEE transactions on vehicular technology* 65, 12 (2016), 9457–9470.

[2] Adrian K. Agogino and Kagan Tumer. 2008. Analyzing and Visualizing Multiagent Rewards in Dynamic and Stochastic Environments. *Journal of Autonomous Agents and Multiagent Systems* (2008), 320–338.

[3] Aamena Alshamsi and Sherief Abdallah. 2009. Multiagent self-organization for a taxi dispatch system. In *Proceedings of 8th International Conference of Autonomous Agents and Multiagent Systems, 2009*. 89–96.

[4] Andrew Amey, John Attanucci, and Rabi Mishalani. 2011. Real-time ridesharing: opportunities and challenges in using mobile phone technology to improve rideshare services. *Transportation Research Record: Journal of the Transportation Research Board* 2217 (2011), 103–110.

[5] Luigi Atzori, Antonio Iera, and Giacomo Morabito. 2010. The Internet of Things: A survey. *Computer Networks* 54, 15 (2010), 2787 – 2805. https://doi.org/10.1016/j.comnet.2010.05.010

[6] Dimitri P Bertsekas. 2012. Weighted sup-norm contractions in dynamic programming: A review and some new applications. *Dept. Elect. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, USA, Tech. Rep. LIDS-P-2884* (2012).

[7] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. 2010. *Multi-agent Reinforcement Learning: An Overview*. Springer Berlin Heidelberg, Berlin, Heidelberg, 183–221. https://doi.org/10.1007/978-3-642-14435-6_7

[8] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2017. Counterfactual Multi-Agent Policy Gradients. *arXiv preprint arXiv:1705.08926* (2017).

[9] Jakob Foerster, Nantas Nardelli, Gregory Farquhar, Triantafyllos Afouras, Philip HS Torr, Pushmeet Kohli, and Shimon Whiteson. 2017. Stabilising Experience Replay for Deep Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*. 1146–1155.

[10] Masabumi Furuhata, Maged Dessouky, Fernando Ordóñez, Marc-Etienne Brunet, Xiaoqing Wang, and Sven Koenig. 2013. Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological* 57 (2013), 28–46.

[11] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. 2013. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems* 29, 7 (2013), 1645 – 1660. https://doi.org/10.1016/j.future.2013.01.010 Including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services & Cloud Computing and Scientific Applications — Big Data, Scalable Analytics, and Beyond.

[12] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative multi-agent control using deep reinforcement learning. In *AAMAS*. Springer, 66–83.

[13] Tommi Jaakkola, Michael I Jordan, and Satinder P Singh. 1994. Convergence of stochastic iterative dynamic programming algorithms. In *NIPS*. 703–710.

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Vijay R. Konda and John N. Tsitsiklis. 2000. Actor-Critic Algorithms. In *Advances in Neural Information Processing Systems 12*, S. A. Solla, T. K. Leen, and K. Müller (Eds.). MIT Press, 1008–1014. http://papers.nips.cc/paper/1786-actor-critic-algorithms.pdf

[16] Der-Horng Lee, Hao Wang, Ruey Cheu, and Siew Teo. 2004. Taxi dispatch system based on current demands and real-time traffic conditions. *Transportation Research Record: Journal of the Transportation Research Board* 1882 (2004), 193–200.

[17] Junghoon Lee, Gyung-Leen Park, Hanil Kim, Young-Kyu Yang, Pankoo Kim, and Sang-Wook Kim. 2007. A Telematics Service System Based on the Linux Cluster. In *Computational Science – ICCS 2007*, Yong Shi, Geert Dick van Albada, Jack Dongarra, and Peter M. A. Sloot (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 660–667.

[18] Ziru Li, Yili Hong, and Zhongju Zhang. 2016. An empirical analysis of on-demand ride sharing and traffic congestion. In *2016 International Conference on Information Systems, ICIS 2016*. Association for Information Systems.

[19] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. 2018. Efficient Large-Scale Fleet Management via Multi-Agent Deep Reinforcement Learning. *arXiv preprint arXiv:1802.06444* (2018).

[20] Michael L. Littman. 1994. Markov Games As a Framework for Multi-agent Reinforcement Learning. In *Proceedings of the Eleventh International Conference on International Conference on Machine Learning (ICML'94)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 157–163. http://dl.acm.org/citation.cfm?id=3091574.3091594

[21] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. In *NIPS*. 6382–6393.

[22] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark. 2014. Connected Vehicles: Solutions and Challenges. *IEEE Internet of Things Journal* 1, 4 (Aug 2014), 289–299. https://doi.org/10.1109/JIOT.2014.2327587

[23] Gary S Lynch. 2009. *Single point of failure: The 10 essential laws of supply chain risk management*. John Wiley & Sons.

[24] Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. 2012. Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems. *The Knowledge Engineering Review* 27, 1 (2012), 1–31.

[25] Francisco S Melo, Sean P Meyn, and M Isabel Ribeiro. 2008. An analysis of reinforcement learning with function approximation. In *Proceedings of the 25th international conference on Machine learning*. ACM, 664–671.

[26] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[27] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 807–814.

[28] Christos H. Papadimitriou and Kenneth Steiglitz. 1982. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[29] Peng Peng, Quan Yuan, Ying Wen, Yaodong Yang, Zhenkun Tang, Haitao Long, and Jun Wang. 2017. Multiagent Bidirectionally-Coordinated Nets for Learning to Play StarCraft Combat Games. *arXiv preprint arXiv:1703.10069* (2017).

[30] Kiam Tian Seow, Nam Hai Dang, and Der-Horng Lee. 2010. A collaborative multiagent taxi-dispatch system. *IEEE Transactions on Automation Science and Engineering* 7, 3 (2010), 607–616.

[31] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *ICML*. 387–395.

[32] Richard S Sutton. 1988. Learning to predict by the methods of temporal differences. *Machine learning* 3, 1 (1988), 9–44.

[33] Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction*. Vol. 1. MIT press Cambridge.

[34] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. 1999. Policy gradient methods for reinforcement learning with function approximation.. In *NIPS*, Vol. 99. 1057–1063.

[35] Csaba Szepesvári and Michael L Littman. 1999. A unified analysis of value-function-based reinforcement-learning algorithms. *Neural computation* 11, 8 (1999), 2017–2060.

[36] Ming Tan. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*. 330–337.

[37] Gerald Tesauro. 2004. Extending Q-learning to general adaptive multi-agent systems. In *NIPS*. 871–878.

[38] Zhaodong Wang, Zhiwei Qin, Xiaocheng Tang, Jieping Ye, and Hongtu Zhu. 2018. Deep Reinforcement Learning with Knowledge Transfer for Online Rides Order Dispatching. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 617–626.

[39] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8, 3-4 (1992), 279–292.

[40] Michael Wooldridge. 2009. *An Introduction to MultiAgent Systems* (2nd ed.). Wiley Publishing.

[41] Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping Ye. 2018. Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms: A Learning and Planning Approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. ACM, New York, NY, USA, 905–913. https://doi.org/10.1145/3219819.3219824

[42] F. Yang, S. Wang, J. Li, Z. Liu, and Q. Sun. 2014. An overview of Internet of Vehicles. *China Communications* 11, 10 (Oct 2014), 1–15. https://doi.org/10.1109/

CC.2014.6969789

[43] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean Field Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Jennifer Dy and Andreas Krause (Eds.), Vol. 80. PMLR, StockholmsmÃďssan, Stockholm Sweden, 5567–5576.

[44] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi. 2014. Internet of Things for Smart Cities. *IEEE Internet of Things Journal* 1, 1 (Feb 2014), 22–32. https://doi.org/10.1109/JIOT.2014.2306328

[45] Lingyu Zhang, Tao Hu, Yue Min, Guobin Wu, Junying Zhang, Pengcheng Feng, Pinghua Gong, and Jieping Ye. 2017. A Taxi Order Dispatch Model Based On Combinatorial Optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*. ACM, New York, NY, USA, 2151–2159. https://doi.org/10.1145/3097983.3098138

Exploiting Ratings, Reviews and Relationships for Item Recommendations in Topic Based Social Networks

Pengfei Li
Zhejiang University, China
pfl@zju.edu.cn

Hua Lu
Aalborg University, Denmark
luhua@cs.aau.dk

Gang Zheng
Zhejiang University, China
gangzheng@zju.edu.cn

Qian Zheng
Nanyang Technological University, Singapore
csqianzheng@gmail.com

Long Yang
Zhejiang University, China
yanglong@zju.edu.cn

Gang Pan
Zhejiang University, China
gpan@zju.edu.cn

ABSTRACT

Many e-commerce platforms today allow users to give their rating scores and reviews on items as well as to establish social relationships with other users. As a result, such platforms accumulate heterogeneous data including numeric scores, short textual reviews, and social relationships. However, many recommender systems only consider historical user feedbacks in modeling user preferences. More specifically, most existing recommendation approaches only use rating scores but ignore reviews and social relationships in the user-generated data. In this paper, we propose TSNPF—a latent factor model to effectively capture user preferences and item features. Employing Poisson factorization, TSNPF fully exploits the wealth of information in rating scores, review text and social relationships altogether. It extracts topics of items and users from the review text and makes use of similarities between user pairs with social relationships, which results in a comprehensive understanding of user preferences. Experimental results on real-world datasets demonstrate that our TSNPF approach is highly effective at recommending items to users.

KEYWORDS

Recommender System, Graphical Model, Variational Inference

ACM Reference Format:
Pengfei Li, Hua Lu, Gang Zheng, Qian Zheng, Long Yang, and Gang Pan. 2019. Exploiting Ratings, Reviews and Relationships for Item Recommendations in Topic Based Social Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313473

1 INTRODUCTION

Recommender systems are the core of today's personalized online e-commerce platforms like Amazon and Taobao. Such systems make use of user activities (e.g., ratings and reviews) and/or social relationships to generate features of items and identify user preferences.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313473

The rating matrix based collaborative filtering [38] makes use of user activity to identify latent user-item factors to suggest products to a user. A representative realization of latent factor models is based on Matrix Factorization [27]. However, matrix factorization is confronted with a big challenge—computation complexity. When there are large numbers of users or items, the matrix factorization methods inevitably incur very high cost of latent factor identification from a huge user-item rating matrix. With the intensive mathematical computation required, such methods cannot handle large datasets. Some variants, e.g., Probabilistic Matrix Factorization [37], attempt to solve this problem. However, like most collaborative filtering methods, such variants often fail to identify the latent factors when a user or an item is associated with too few ratings. Consequently, the computation for recommendation falls apart due to data sparsity—too many zero ratings in a rating matrix. In contrast, the Hierarchical Poisson Factorization (HPF) [6, 15] only utilizes non-zero ratings to make scalable recommendations on large and sparse data.

In addition to user ratings on items, modern e-commerce platforms also generate other kinds of useful data such as social relationships and user reviews. By combining a rating matrix with additional data, better recommendations are expected. On the one hand, user textual reviews contain large amounts of information that often reflect concrete consumer experiences and many item attributes related to product quality. Some studies have utilized review text information for recommendation. By integrating ratings with review text or item contents, works [2, 9, 14, 28, 43] combine factors in ratings with topics in item reviews. On the other hand, the abundant social relationships provide an independent data source for recommendation and another possibility to improve recommended result quality. E.g., existing works [16, 31, 41] factorize rating matrix and social matrix simultaneously to improve recommendation performance.

In general, methods utilizing heterogeneous data tend to perform better than those using a single data source [45]. However, few studies take a rating matrix, social relationships and review text altogether into consideration. Work [20] is such an effort. However, it is simply a linear combination of two methods [2, 41]; it does not leverage the comprehensive implications of the latent factors in the completeness of a rating matrix, review text and social relationships.

In this paper, we propose an approach called Topic Soical Network Poisson Factorization (TSNPF for short). It integrates a rating matrix, review text and social relationships altogether in order

to enable high-quality recommendations. Overall, TSNPF aims to find item attributes and user preferences by exploiting all available heterogeneous data. Specifically, TSNPF merges all reviews on an item/user to generate the *document* for this item/user, in the hope of extracting topic intensities from such documents. When the topics of an item and a user are similar, the rating score of the user on that item tends to be high. However, item attributes and user preferences are not fully parameterized by their documents. To this end, we associate an attribute annex to each item and a preference annex to each user. As a result, the summation of topic intensities and corresponding annexes capture item attributes and user preferences to a greater extent. Rating of a user on an item is generated from the user's preferences and the item's attributes. In addition, most existing recommender systems that use social information assume that friends have similar preferences, which is however unrealistic in real life. In contrast, we utilize social relationships in a different way. Specifically, by assuming the similarity between two friends are generated from both users' preferences, we model the similarity with respect to the items they both have rated. The intuition behind is that a pair of friends who share similar (different) ratings on the same items naturally tend to have similar (different) preferences.

Our TSNPF exploits observations of multiple kinds, namely the ratings, the frequencies of words in the documents of items and users, and the similarity between a pair of friends on the items which they have both rated. In our model, each observation is represented as non-negative integers and generated from a Poisson distribution [25] that is in exponential family over non-negative integers. The latent item/user topic intensity, the item attribute annex and the user preference annex are all sampled from Gamma distributions—another exponential family distribution with shape and rate parameters. TSNPF only scans non-zero ratings, word frequencies and similarities, and factorizes them based on their posterior Gamma distributions. Therefore, TSNPF is able to capture features from sparse observations. By augmenting TSNPF with some auxiliary variables, we make TSNPF conditionally conjugate such that we can apply particular techniques to infer the parameters for it. To fit our TSNPF model, we use variational inference. It is an optimization-based strategy for efficiently approximating posterior distributions in large-scale complex probabilistic models [26, 42]. We propose a simple mean-field [36] variational inference method to update the parameters of TSNPF.

Our contributions are summarized as follows:

- We propose a method based on Gamma-Poisson distribution to extract the topic intensities of items and users from user-generated textual reviews. Compared to previous techniques, our method is able to address the usual problem of data scarcity.
- We propose TSNPF, a conjugate graphical model based on Poisson factorization which only models non-zero observations in ratings, reviews and social relations simultaneously via interpretable user preferences and item attributes. In addition, we propose a closed form mean-field variational inference method to train TSNPF.
- We evaluate the performance of TSNPF using three publicly available real datasets. The results show that TSNPF outperforms state-of-the-art alternatives.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 formulates the research problem and introduces the notations used throughout this paper. Section 4 describes our TSNPF model. Sections 5 details the variational inference for our TSNPF model. Experimental results are shown in Section 6. Section 7 concludes the paper and points out future work directions.

2 RELETED WORK

In this section, we mainly discuss the works which utilize heterogeneous data for recommendations.

Many existing works attempt to extract topics and features from user-generated reviews and learn latent factors from ratings using matrix factorization (MF) methods [10, 11, 18, 33, 40, 47, 52]. A general idea of these methods is to extract latent topics from reviews using topic models [10, 28, 33, 40, 47]. In particular, RMR [28] uses topic models on reviews to learns items' features and models rating using a maxiture of Gaussian. RBLT [40] and ITLFM [47] linearly combine the latent topics and latent factors to form the latent representations for users and items. TopicMF [2] and HPF [33] define a transform action function to link the latent topics and latent factors. He *et al.* [18] model the user-item-aspect relation with a tripartite graph to extract latent topics from reviews. JMARS [11] is an integrated graphical model that makes use of ratings and sentiments together for movie recommendation. However, with the rapid growth of user numbers, the increasing sparsity of the data becomes a critical concern [30]. Recently, neural networks are widely used to process reviews in recommender systems [7, 48, 49, 52]. DeepCoNN [52] first uses two CNNs to process reviews to learn users' and items' representations which are in turn concatenated and passed into a regression layer for rating predictions. However, neural network based methods often have the efficiency problem and their performance decrease greatly when reviews are scarce or unavailable in the testing phase [7].

There are also works [34, 35, 50] analyzing users' sentiments in reviews to improve the quality of recommendations. These works usually rely on the external NLP tools for sentiment analysis and thus are not self-contained.

Besides, item descriptions are also useful data for recommendations. Personalized articles recommendation methods such as CTR [43] and CTPF [14] assume that latent factors of items depend on the latent topic distributions from article contents. CTR [43] combines topic modeling using LDA [5] with Gaussian matrix factorization for one-class collaborative filtering [21]. CTPF [14] models both reader behavior and article texts with Poisson distributions, connecting the latent topics that represent the texts with the latent preferences that represent the readers. However, CTR is not conditionally conjugate and its inference algorithm depends on numerical optimization of topic intensities. Also, CTR and CTPF demand that the items themselves are articles or associated with descriptions. Thus, it is difficult to apply them in other scenarios.

In addition to textual data, social relations is another kind of heterogeneous data often used in recommender systems. LOCABAL [41] takes advantage of both local and global social contexts for recommendation. Ma *et al.* [31] propose a factor analysis approach based on probabilistic matrix factorization called "SoRec". It addresses the data sparsity and poor prediction accuracy problems

by employing both users' social network information and rating records. This idea is also applied in other works [24, 32]. Trust-based approach is another way to utilize social relations to make recommendation [23, 51]. This approach assumes a trust network among users and makes recommendations based on the ratings of the users that are directly or indirectly trusted by a user. Combining TopicMF [2] and LOCABAL [41], synthetic approach MR3 [20] utilizes ratings, social relations and reviews together for recommendation. However, MR3 is essentially a linear combination and it performs worse than our proposed approach according to our experimental results.

Moreover, Zhang *et. al* [46] and He *et al.* [17] consider the effect of visual content such as images when making recommendations. Bao *et. al* [1] and Do *et. al* [12] integrate metadata into their approaches.

3 NOTATIONS AND PROBLEM FORMULATION

Traditional recommender systems usually only use the rating matrix and ignore review text and social relationships. However, it is beneficial to take both of these two aspects into consideration [20]. On the one hand, social relationships often have impacts on a user's impression of items. On the other hand, a rating score can tell if a user likes or dislikes an item, but it cannot tell the reason behind the preference. In contrast, if this rating score is associated with some review text, the chance is better for us to understand why the user likes or dislikes the item. In addition, reviews offer abundant information about items' attributes. We intend to exploit the full combination of the rating matrix, the review text and the social relationships. We fuse three heterogeneous data types in one comprehensive data model to make high quality recommendations. The main notations used throughout the paper are shown in Table 1.

Table 1: Notations

R_{ui}	User u's rating on item i
D_i^I	Document of item i
D_u^U	Document of user u
C_{iw}^I	Count of word w in D_i^I
C_{uw}^U	Count of word w in D_u^U
S_{uvi}	Rating similarity between users u and v on item i
ε_w	latent topic rate for word w
ϵ_w	Latent topic intensities for word w
ϖ_i	latent topic rate for item i
π_i	Latent topic intensities for item i
α_i	latent popularity for item i
β_i	Latent attribute annex for item i
ς_u	Latent topic rate for user u
σ_u	Latent topic intensities for user u
ϑ_u	Latent activity for user u
θ_u	Latent preference annex for user u

Suppose there are U users numbered from 1 to U and I items numbered from 1 to I. The matrix of ratings given by users to items is denoted by $R \in \mathbb{R}^{U\times I}$, where R_{ui} is the rating by user u on item i. Each R_{ui} is an integer satisfying $0 \le R_{ui} \le M$, where M is the maximum rating score and 0 means currently the rating is unavailable. In addition to the rating matrix, the observed review on item i written by user u is denoted as D_{ui}, which is along with a rating score R_{ui}. For every item i, we merge all users' review comments together, generating a *document* D_i^I, i.e., $D_i^I = \bigcup_u D_{ui}$. Likewise, D_u^U denotes the document of a user u, i.e., $D_u^U = \bigcup_i D_{ui}$. We use D^I and D^U to denote the set of all documents of items and users, respectively. Furthermore, we use $G \in \mathbb{R}^{U\times U}$ to denote the user social network where $G_{uv} = 1$ means users u and v are friends and $G_{uv} = 0$ otherwise. Typically, R, D^I, D^U and G are all highly sparse, i.e., having many zero values.

For the recommendation purpose, we need to judge if a user will like an item that he/she has never consumed before. The problem statement for our research is as follows:

Problem Formulation: Given the rating matrix R, the document sets of items and users, i.e., D^I and D^U, and the social network G, estimate user u's rating on item i, i.e., R_{ui} that currently is 0.

4 TOPIC SOCIAL NETWORK POISSON FACTORIZATION

In this section, we describe the topic social network poisson factorization model (TSNPF). We model each user and each item by exploiting the rating matrix R, the documents sets D^I and D^U, and the social network G comprehensively.

Extracting topic intensities from documents. In TSNPF, there is a collection of K topics $\epsilon_{1:V,1:K}$ where each topic $\epsilon_{:,k}$ is composed of a vector of word intensities on the vocabulary and V is the size of the vocabulary. An item i is partly parameterized by the intensities of these topics denoted by $\pi_{i,1:K}$. The document D_i^I is generated by $\epsilon_{1:V}$ and π_i with C_{iw}^I, where C_{iw}^I is the count of word w in D_i and it is modeled by the inner product of ϵ_w and π_i, i.e., $C_{iw}^I \sim \text{Poisson}(\pi_i^\top \epsilon_w)$. Suppose $\sigma_{u,1:K}$ models the topics of user u, and C_{uw}^U is the word count of w in the document D_u^U. Likewise, C_{uw}^U is modeled using $\text{Poisson}(\sigma_u^\top \epsilon_w)$. When topic intensities of an item and a user are similar, the corresponding rating tends to be high. Using word intensities as topics is an widely-used technique [5]. Compared to existing work, our method of extracting topics is built on Poisson factorization which can take advantage of natural sparsity of user/item documents and is more efficient than Gaussian factorization based approaches [14].

Modeling ratings. The documents of items or users can not fully parameterize the items or users [14]. To this end, we make TSNPFassociate K latent attribute annexes $\beta_{i,1:K}$ to each item i and K latent preference annexes $\sigma_{u,1:K}$ to each user u. Such annexes capture the items' and users' deviations from their topic intensities. The rating R_{ui} is modeled by the inner product of the item attributes and user preferences, i.e., $R_{ui} \sim \text{Poisson}\Big((\sigma_u + \theta_u)^\top (\pi_i + \beta_i)\Big)$.

Modeling similarities. In addition, it is reasonable to believe that a user u's ratings on items are influenced by u's friends. When a pair of friends u and v have similar preferences, they are likely to give similar ratings on items. To incorporate this, we define the similarity between u and v on each item i on which they both give ratings, i.e., $S_{uvi} = M - |R_{ui} - R_{vi}|$ where M is the maximum rating score. In TSNPF, we model S_{uvi} using the inner product of users u and v's preferences, i.e., $S_{uv,i} \sim \text{Poisson}\Big((\sigma_u + \theta_u)^\top (\sigma_v + \theta_v)\Big)$. In this way, the distance of a pair of friends' preference vectors tends to be small when the users have similar ratings.

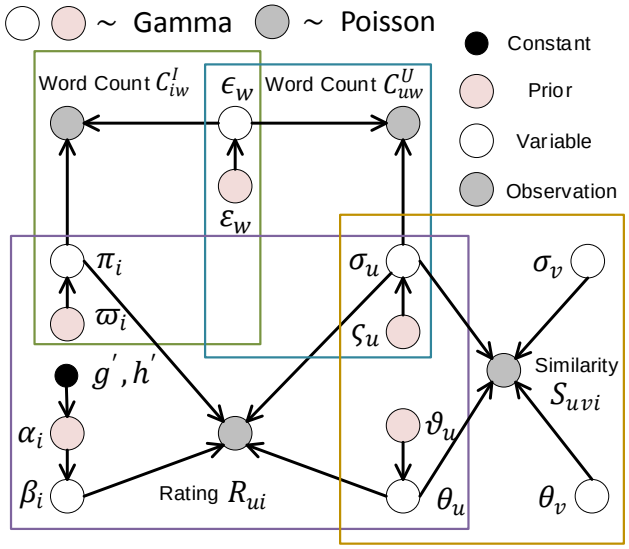


Figure 1: The directed graphical model representing TSNPF.

To deal with the sparsity of users and items, TSNPF also preserves the heterogeneity in the features by placing Gamma priors on ϵ_w, π_i, σ_u, β_i and θ_u [15]. The graphical model of TSNPF is illustrated in Figure 1. To avoid ambiguity, we only draw one constant node (g', h') in the example. More specifically, TSNPF's generative process is described as follows:

1. Generating documents of items and users
   1). For each word w:
      (a) Sample latent topic rate $\varepsilon_w \sim \text{Gamma}(a', b')$.
      (b) Sample word topic intensities $\epsilon_{wk} \sim \text{Gamma}(a, \varepsilon_w)$.
   2). For each item i:
      (a) Sample topic rate $\varpi_i \sim \text{Gamma}(c', d')$.
      (b) Sample document topic intensities $\pi_{ik} \sim \text{Gamma}(c, \varpi_i)$.
      (c) For every word w in D^I_i, sample word count
      $$C^I_{iw} \sim \text{Poisson}(\pi_i^\top \epsilon_w).$$
   3). For each user u:
      (a) Sample topic rate $\varsigma_u \sim \text{Gamma}(e', f')$.
      (b) Sample document topic intensities $\sigma_{uk} \sim \text{Gamma}(e, \varsigma_u)$.
      (c) For every word w in D^U_u, sample word count
      $$C^U_{uw} \sim \text{Poisson}(\sigma_u^\top \epsilon_w).$$
2. Generating ratings
   1). For each item i:
      (a) Sample latent popularity $\alpha_i \sim \text{Gamma}(g', h')$.
      (b) Sample attribute annex for each component k:
      $$\beta_{ik} \sim \text{Gamma}(g, \alpha_i).$$
   2). For each user u:
      (a) Sample latent activity $\vartheta_u \sim \text{Gamma}(m', n')$.
      (b) Sample preference annex for each component k:
      $$\theta_{uk} \sim \text{Gamma}(m, \vartheta_u)$$
   3). For each user u and item i, sample rating
      $$R_{ui} \sim \text{Poisson}\Big((\sigma_u + \theta_u)^\top (\pi_i + \beta_i)\Big).$$
3. Generating rating similarities between friends
   For a pair of users u and v with $G_{uv} = 1$, for every item they both have rated, sample rating similarity:
   $$S_{uvi} \sim \text{Poisson}\Big((\sigma_u + \theta_u)^\top (\sigma_v + \theta_v)\Big).$$

Given the observed rating matrix R, users similarity set S and word count sets C^I and C^U generated by document sets D^I and D^U, respectively, our goal is to infer the topics $\epsilon_{w,1:K}$, the item and user topic intensities $\pi_{i,1;K}$ and $\sigma_{u,1;K}$, the item attribute annex $\beta_{i,1:K}$, the user preference annex $\theta_{u,1:K}$ and their priors, i.e., to estimate the posterior distribution $p(\epsilon, \varepsilon, \pi, \varpi, \sigma, \varsigma, \beta, \alpha, \theta, \vartheta, |R, S, C^I, C^U)$ of these variables. Once the posterior is fit, TSNPF can be used to recommend items to users by ranking users' unconsumed items according to their scores based on the posterior expected Poisson parameters. Such a score for user u and item i is defined as follow:

$$\text{score}(u, i) = \mathbb{E}[(\sigma_u + \theta_u)^\top (\pi_i + \beta_i) | R, S, C^I, C^U]$$

5 INFERENCE FOR TSNPF

It is intractable to compute the exact posterior distribution $p(\epsilon, \varepsilon, \pi, \varpi, \sigma, \varsigma, \beta, \alpha, \theta, \vartheta | R, S, C^I, C^U)$ of TSNPF directly as variables are dependent on each other in p. We use variational inference [26] to approximate it. A family of distributions over latent variables indexed by a set of free parameters is introduced in variational inference. Those parameters are optimized to find the member of the family that is closest to the posterior. Previous studies [39, 44] have proved that the combination of simple distributions (*e.g.*, Poisson and Gaussian distribution) is able to approximate a very complex distribution. Thus, variational inference is reasonable. We use a distribution $q(\epsilon, \varepsilon, \pi, \varpi, \sigma, \varsigma, \beta, \alpha, \theta, \vartheta)$ in mean-field family [36], the simplest variational family, to approximate the exact posterior distribution p.

5.1 Complete Conditionals for TSNPF

We first augment TSNPF with auxiliary variables to make it conditionally conjugate, which means the complete conditional of each latent variable in p is in the exponential family and is in the same family as its prior [13]. A complete conditional is the conditional distribution of a latent variable given the observations and other latent variables. Following previous work [15], for word counts C^I_{iw} and C^U_{uw}, we add K latent variables $x^I_{iw,k} \sim \text{Poisson}(\epsilon_{wk}\pi_{ik})$ and K latent variables $x^U_{iw,k} \sim \text{Poisson}(\epsilon_{wk}\sigma_{uk})$, respectively, where $C^I_{iw} = \sum_k x^I_{iw,k}$ and $C^U_{uw} = \sum_k x^U_{uw,k}$. For rating similarity S_{uvi} that is non-zero, we add $2K$ latent variables $z_{uvi,k}$ which include two parts: $z^\sigma_{uvi,k}$ and $z^\theta_{uvi,k}$ such that $z^\sigma_{uvi,k} \sim \text{Poisson}(\sigma_{uk}\sigma_{vk})$ and $z^\theta_{uvi,k} \sim \textit{Poisson}(\theta_{uk}\theta_{vk})$, where $S_{uvi} = \sum_k (z^\sigma_{uvi,k} + z^\theta_{uvi,k})$ for every item i. For rating R_{ui}, $4K$ latent variables $y^1_{ui,k} \sim \text{Poisson}(\pi_{ik}\sigma_{uk})$, $y^2_{ui,k} \sim \text{Poisson}(\pi_{ik}\theta_{uk})$, $y^3_{ui,k} \sim \text{Poisson}(\beta_{ik}\sigma_{uk})$ and $y^4_{ui,k} \sim \text{Poisson}(\beta_{ik}\theta_{uk})$ are introduced such that $R_{ui} = \sum_k (y^1_{ui,k} + y^2_{ui,k} + y^3_{ui,k} + y^4_{ui,k})$. As a sum of independent Poisson random variables is itself a Poisson random variable with rate equal to the sum of the rates, the new latent variables preserve the marginal distribution of the observations. With the additional auxiliary variables, TSNPF is conditionally conjugate. We define the mean-field family that considers the latent variables

Table 2: TSNPF: latent variables, complete conditionals and variational parameters

Variable	Type	Complete Conditional	Variational Params
ε_w	Gamma	$a' + Ka,\ b' + \sum_k \epsilon_{wk}$	$\iota_w^{shp}, \iota_w^{rte}$
ϵ_{wk}	Gamma	$a + \sum_i x_{iw,k}^I + \sum_u x_{uw,k}^U,\ \varepsilon_w + \sum_i \pi_{ik} + \sum_u \sigma_{uk}$	$\mu_{wk}^{shp}, \mu_{wk}^{rte}$
ϖ_i	Gamma	$c' + Kc,\ d' + \sum_k \pi_{ik}$	$\zeta_i^{shp}, \zeta_i^{rte}$
π_{ik}	Gamma	$c + \sum_w x_{iw,k}^I + \sum_u (y_{ui,k}^1 + y_{ui,k}^2),\ \varpi_i + \sum_w \epsilon_{wk} + \sum_u (\sigma_{uk} + \theta_{uk})$	$\rho_{ik}^{shp}, \rho_{ik}^{rte}$
α_i	Gamma	$g' + Kc,\ h' + \sum_k \beta_{ik}$	$\tau_i^{shp}, \tau_i^{rte}$
β_{ik}	Gamma	$g + \sum_u (y_{ui,k}^3 + y_{ui,k}^4),\ \alpha_i + \sum_u (\sigma_{uk} + \theta_{uk})$	$\lambda_{ik}^{shp}, \lambda_{ik}^{rte}$
ς_u	Gamma	$e' + Ke,\ f' + \sum_k \sigma_{uk}$	ξ_u^{shp}, ξ_u^{rte}
σ_{uk}	Gamma	$e + \sum_w x_{iw,k}^U + \sum_i (y_{ui,k}^1 + y_{ui,k}^3) + \sum_{v,i,S_{uvi}\neq 0} z_{uvi,k}^{\sigma},$ $\varsigma_u + \sum_w \epsilon_{wk} + \sum_i (\pi_{ik} + \beta_{ik}) + \sum_{v,i,S_{uvi}\neq 0} \sigma_{v,k}$	$\nu_{uk}^{shp}, \nu_{uk}^{rte}$
ϑ_u	Gamma	$m' + Km,\ n' + \sum_k \theta_{uk}$	$\kappa_u^{shp}, \kappa_u^{rte}$
θ_{uk}	Gamma	$m + \sum_i (y_{ui,k}^2 + y_{ui,k}^4) + \sum_{v,i,S_{uvi}\neq 0} z_{uv,k}^{\theta},\ \vartheta_u + \sum_i (\pi_{ik} + \beta_{ik}) + \sum_{v,i,S_{uvi}\neq 0} \theta_{vk}$	$\gamma_{uk}^{shp}, \gamma_{uk}^{rte}$
x_{iw}^I	Mult	$\log \pi_{ik} + \log \epsilon_{wk}$	χ_{iw}^I
x_{uw}^U	Mult	$\log \sigma_{uk} + \log \epsilon_{wk}$	χ_{iw}^U
z_{uvi}	Mult	$\begin{cases} \log \sigma_{uk} + \log \sigma_{vk}, & \text{case } z_{uvi}^{\sigma} \\ \log \theta_{uk} + \log \theta_{vk}, & \text{case } z_{uvi}^{\theta} \end{cases}$	$\varphi_{uv}^{\sigma}, \varphi_{uv}^{\theta}$
y_{ui}	Mult	$\begin{cases} \log \pi_{ik} + \log \sigma_{uk}, & \text{case } y_{ui}^1 \\ \log \pi_{ik} + \log \theta_{uk}, & \text{case } y_{ui}^2 \\ \log \beta_{ik} + \log \sigma_{uk}, & \text{case } y_{ui}^3 \\ \log \beta_{ik} + \log \theta_{uk}, & \text{case } y_{ui}^4 \end{cases}$	$\phi_{ui}^1, \phi_{ui}^2, \phi_{ui}^3, \phi_{ui}^4$

to be independent and each governed by its own distributions.

$$
\begin{aligned}
q(\varepsilon, \epsilon, \varpi, \pi, \alpha, \beta, \varsigma, \sigma, \vartheta, \theta, x^I, x^U, z^{\sigma}, z^{\theta}, y) = & \prod_{w,k} q(\epsilon_{wk}|\mu_{wk}) \\
& \prod_w q(\varepsilon_w|\iota_w) \prod_{i,k} q(\pi_{ik}|\rho_{ik}) q(\beta_{ik}|\lambda_{ik}) \prod_i q(\varpi_i|\zeta_i) q(\alpha_i|\tau_i) \\
& \prod_{u,k} q(\sigma_{uk}|\nu_{uk}) q(\varsigma_u|\xi_u) q(\theta_{uk}|\gamma_{uk}) \prod_u q(\vartheta_u|\kappa_u) \\
& \prod_{i,w} q(x_{iw}^I|\chi_{iw}^I) \prod_{u,w} q(x_{uw}^U|\chi_{iw}^U) \prod_{u,v,i} q(z_{uvi}^{\sigma}, z_{uvi}^{\theta}|\varphi_{uv}^{\sigma}, \varphi_{uv}^{\theta}) \\
& \prod_{u,i} q(y_{ui}^1, y_{ui}^2, y_{ui}^3, y_{ui}^4|\phi_{ui}^1, \phi_{ui}^2, \phi_{ui}^3, \phi_{ui}^4) \qquad (1)
\end{aligned}
$$

The distributions of the variational variables (ε_w, ϵ_{wk}, ϖ_i, π_{ik}, α_i, β_{ik}, ς_u, σ_{uk}, ϑ_u and θ_{uk}) and the auxiliary variables (x^I, x^U, z and y) are all the same as their conditional distributions in p. The complete conditionals of all variables and the variational parameters that govern these variables in q are shown in Table 2. For more detailed derivation, please refer to Appendix A.1

The complete conditionals of ε_w, ϵ_{wk}, ϖ_i, π_{ik}, α_i, β_{ik}, ς_u, σ_{uk}, ϑ_u and θ_{uk} are all Gamma distributions, which are in the exponential family, with shape and rate variational parameters. We denote shape with superscript 'shp' and rate with 'rte'. For example, the variational distribution for the user preference θ_{uk} is Gamma($\gamma_{uk}^{shp}, \gamma_{uk}^{rte}$). For auxiliary variables x_{iw}^I, x_{uw}^U, z_{uvi} and y_{ui}, the complete conditionals are all free multinomials [25]. Specifically, χ_{iw}^I and χ_{uw}^U are all K-vectors that sum to one; $\varphi_{uv} = (\varphi_{uv}^{\sigma}, \varphi_{uv}^{\theta})$ and $\phi_{ui} = (\phi_{ui}^1, \phi_{ui}^2, \phi_{ui}^3, \phi_{ui}^4)$ are points in the $2K$ and $4K$-simplex, respectively.

5.2 Closed Form Variational Parameters Update

In this section, we give an coordinate ascent algorithm [4, 19] to update every variational parameter by holding all other parameters fixed. The detailed derivation can be found in Appendix A.2.

For all words and items, we initialize the word topic intensities ϵ_w using LDA [5]. These operations can be implemented by setting the point-wise ratio μ_w^{shp}/μ_w^{rte} to be ϵ_w. Subsequently, for all users and items, we initialize the user activity rate κ_u^{rte}, user preference γ_u, item popularity rate τ_i^{rte} and item feature λ_i with a small positive noise. We initialize the item topic intensities and attribute annexes i.e., ζ_i^{shp} and τ_i^{shp}, and the user topic intensities and preference annexes shapes, i.e., ξ_u^{shp} and κ_u^{shp}, according to the following equations:

$$
\begin{aligned}
\zeta_i^{shp} &= c' + Kc;\ \tau_i^{shp} = g' + Kg \\
\xi_u^{shp} &= e' + Ke;\ \kappa_u^{shp} = m' + Km
\end{aligned}
$$

Furthermore, the following steps are repeated until convergence:

1. Let

$$
f(r, s) = \exp\left(\Psi(r^{shp}) - \log r^{rte} + \Psi(s^{shp}) - \log s^{rte}\right)
$$

We use $\Psi(\cdot)$ to denote the digamma function. For each word/item such that $C_{iw}^I > 0$, each word/user such that $C_{uw}^U > 0$, each user pairs such that $S_{uvi} > 0$ for some items and each

user/item such that $R_{ui} > 0$, update the multinomials:

$$\hat{\chi}^I_{iw,k} = f(\rho_{ik}, \mu_{wk}) \; ; \; \chi^I_{iw,k} = \frac{\hat{\chi}^I_{iw,k}}{\sum_k \hat{\chi}^I_{iw,k}}$$

$$\hat{\chi}^U_{uw,k} = f(\sigma_{uk}, \mu_{wk}) \; ; \; \chi^U_{uw,k} = \frac{\hat{\chi}^U_{uw,k}}{\sum_k (\hat{\chi}^U_{iw,k})}$$

$$\hat{\varphi}^\sigma_{uv,k} = f(\nu_{uk}, \nu_{vk}) \; ; \; \hat{\varphi}^\theta_{uv,k} = f(\gamma_{uk}, \gamma_{vk})$$

$$\varphi^\sigma_{uv,k} = \frac{\hat{\varphi}^\sigma_{uv,k}}{\sum_k (\hat{\varphi}^\sigma_{uv,k} + \hat{\varphi}^\theta_{uv,k})} \; ; \; \varphi^\theta_{uv,k} = \frac{\hat{\varphi}^\theta_{uv,k}}{\sum_k (\hat{\varphi}^\sigma_{uv,k} + \hat{\varphi}^\theta_{uv,k})}$$

$$\hat{\phi}^1_{ui,k} = f(\rho_{ik}, \nu_{uk}) \; ; \; \hat{\phi}^2_{ui,k} = f(\rho_{ik}, \gamma_{uk})$$

$$\hat{\phi}^3_{ui,k} = f(\lambda_{ik}, \nu_{uk}) \; ; \; \hat{\phi}^4_{ui,k} = f(\lambda_{ik}, \gamma_{uk})$$

$$\phi^j_{ui,k} = \frac{\hat{\phi}^j_{ui,k}}{\sum_k (\hat{\phi}^1_{ui,k} + \hat{\phi}^2_{ui,k} + \hat{\phi}^3_{ui,k} + \hat{\phi}^4_{ui,k})} \; , \; j = 1, 2, 3, 4$$

2. For each item, update the topic prior and intensities, attribute prior and annex parameters:

$$\rho^{shp}_{ik} = c + \sum_w C^I_{iw} \chi^I_{iw,k} + \sum_u R_{ui} (\phi^1_{ui,k} + \phi^2_{ui,k})$$

$$\rho^{rte}_{ik} = \frac{\zeta^{shp}_i}{\zeta^{rte}_i} + \sum_w \frac{\mu^{shp}_{wk}}{\mu^{rte}_{wk}} + \sum_u (\frac{\nu^{shp}_{uk}}{\nu^{rte}_{uk}} + \frac{\gamma^{shp}_{uk}}{\gamma^{rte}_{uk}})$$

$$\lambda^{shp}_{ik} = g + \sum_u R_{ui} (\phi^3_{ui,k} + \phi^4_{ui,k})$$

$$\lambda^{rte}_{ik} = \frac{\tau^{shp}_i}{\tau^{rte}_i} + \sum_u (\frac{\nu^{shp}_{uk}}{\nu^{rte}_{uk}} + \frac{\gamma^{shp}_{uk}}{\gamma^{rte}_{uk}})$$

$$\zeta^{rte}_i = d' + \sum_k \frac{\rho^{shp}_{ik}}{\rho^{rte}_{ik}} ; \tau^{rte}_i = h' + \sum_k \frac{\lambda^{shp}_{ik}}{\lambda^{rte}_{ik}}$$

3. For each user, update the topic prior and intensities, preference prior and annex parameters:

$$\nu^{shp}_{uk} = e + \sum_w C^U_{uw} \chi^U_{uw,k} + \sum_i R_{ui} (\phi^1_{ui,k} + \phi^3_{ui,k}) + \sum_{v,i,S_{uvi} \neq 0} S_{uvi} \varphi^\sigma_{uv,k}$$

$$\nu^{rte}_{uk} = \frac{\xi^{shp}_u}{\xi^{rte}_u} + \sum_w \frac{\mu^{shp}_{wk}}{\mu^{rte}_{wk}} + \sum_i (\frac{\rho^{shp}_{ik}}{\rho^{rte}_{ik}} + \frac{\lambda^{shp}_{ik}}{\lambda^{rte}_{ik}}) + \sum_{v,i,S_{uvi} \neq 0} \frac{\nu^{shp}_{vk}}{\nu^{rte}_{vk}}$$

$$\gamma^{shp}_{uk} = m + \sum_i R_{ui} (\phi^2_{ui,k} + \phi^4_{ui,k}) + \sum_{v,i,S_{uvi} \neq 0} S_{uvi} \varphi^\theta_{uv,k}$$

$$\gamma^{rte}_{uk} = \frac{\kappa^{shp}_u}{\kappa^{rte}_u} + \sum_i (\frac{\rho^{shp}_{ik}}{\rho^{rte}_{ik}} + \frac{\lambda^{shp}_{ik}}{\lambda^{rte}_{ik}}) + \sum_{v,i,S_{uvi} \neq 0} \frac{\gamma^{shp}_{vk}}{\gamma^{rte}_{vk}}$$

$$\xi^{rte}_u = f' + \sum_k \frac{\nu^{shp}_{uk}}{\nu^{shp}_{uk}} ; \kappa^{rte}_u = n' + \sum_k \frac{\gamma^{shp}_{uk}}{\gamma^{shp}_{uk}}$$

4. For each word, update the word topic parameters:

$$\mu^{shp}_{wk} = a + \sum_i C^I_{iw} \chi^I_{iw,k} + \sum_u C^U_{uw} \chi^U_{uw,k}$$

$$\mu^{rte}_{wk} = \frac{\iota^{shp}_w}{\iota^{rte}_w} + \sum_i \frac{\rho^{shp}_{ik}}{\rho^{shp}_{ik}} + \sum_u \frac{\nu^{shp}_{uk}}{\nu^{shp}_{uk}} ; \iota^{rte}_w = b' + \sum_k \frac{\mu^{shp}_{wk}}{\mu^{shp}_{wk}}$$

Specifically, this algorithm only needs the non-zero observations in R, S, C^I and C^U. In step 1 above, we only need to update the variational parameters of the multinomials χ_{iw}, φ_{uv} and ϕ_{ui} for the non-zero word count, user similarity and rating observations, respectively. In steps 2, 3 and 4, all of the sums also only consider the non-zero observations. To judge if the distribution converges, we calculate the log probability of generating a validation rating matrix $\tilde{R}$ at every iteration:

$$\log p(\tilde{R} | \theta, \pi, \beta, \sigma, \theta) = \sum_{u,i,\tilde{R}_{ui} \neq 0} \log p(\tilde{R}_{ui} | \theta_u, \pi_i, \beta_i, \sigma_u, \theta_u)$$

$$= \sum_{u,i,\tilde{R}_{ui} \neq 0} \left(\tilde{R}_{ui} \log \left((\sigma_u + \theta_u)^\top (\pi_i + \beta_i) \right) - \log \tilde{R}_{ui}! \right)$$

$$- (\sum_u \sigma_u + \theta_u)^\top (\sum_i \pi_i + \beta_i)$$

When the change in the log probability is less than a very small threshold, we decide that the distribution converges and terminate the algorithm. The equation above implies that non-zero observations are not necessary when calculating the log probability. Therefore, this algorithm is able to handle sparse data.

6 EXPERIMENTS

In this section, we report on experimental studies that compare our TSNPF with existing methods and investigate the factors having impacts on the performance of TSNPF. In Section 6.1, we describe the settings of our experiments. Section 6.2 details the comparison results between our TSNPF and alternative methods. Section 6.3 reports on the impacts of reviews and social relations on the performance of TSNPF. Finally, Section 6.4 investigates how TSNPF performs as user activeness varies.

6.1 Experimental Settings

Datasets. We use the following open real datasets that are popular and used in the experiments of many recent works [20, 29, 33]:

- **PH-Restaurants** includes 204,887 users, 17,213 restaurants and 728,948 friend pairs. There are on average 1,180 words in the document of each restaurant. It is a part of the dataset of full Yelp data challenge[1] related to Phoenix.
- **LV-Restaurants** is also from Yelp data challenge. It contains 506,278 users, 26,809 restaurants located in Las Vegas and 3,109,068 friend pairs. The average word count of every document is 2,231.
- **Ciao** [2] is a dataset from a knowledge sharing and review website, on which users can rate items, give reviews and connect to others. This dataset includes 9,100 users, 23,432 items and 223,522 friend pairs. Each document contains 1,959 words on average.

[1] https://www.yelp.com/dataset/challenge
[2] https://www.cse.msu.edu/~tangjili/trust.html

The actual ratings on these datasets are all integers ranging from 1 to 5. For every dataset, we only keep top 20,000 words that appear most frequently .

Competing approaches. We compare our TSNPF with the following three typical approaches. They are the most recent representative approaches related to our TSNPF:

- **HPF** [15] factorizes the rating matrix based on Gamma-Poisson distribution. It is the first model in recommender systems that uses Poisson factorization.
- **CuPF** [12] is short for Coupled User Poisson Factorization that learns the couplings between users in the process of Poisson factorization. It is an improvement of HPF.
- **MR3** [20] is a synthetic approach combining ratings, social relationships and reviews. It integrates two other previous works [2, 41].

Evaluation Metrics. We randomly select 20% of the ratings with reviews in each dataset as testing set. Additionally, we set aside 1% of the training ratings and reviews as a validation set which is used to determine the algorithms convergence and to tune variational parameters. We define the relevant and recommended items for user u as those items on which the actual and predicted ratings by u are larger than 3.5, respectively. The following three metrics are used to evaluate the performance of the approaches in our experiments:

- **Normalized Mean Recall** (NMR) is a variant of recall-at-N which adjusts the denominator N'_u for a user u to be the $\min(N, I_u)$ where I_u denotes the most number of items the user u has rated in testing set. It is defined as

$$\text{NMR} = \frac{\sum_u \#\text{ of recommended and relevant items@}N'_u}{\sum_u \#\text{ of relevant items@}N'_u}$$

- **Normalized Mean Precision** (NMP) is a variant of precision-at-N with the denominator defined in normalized mean recall. Likewise,

$$\text{NMP} = \frac{\sum_u \#\text{ of recommended and relevant items@}N'_u}{\sum_u \#\text{ of recommended items@}N'_u}$$

- **Root-Mean-Square Error** (RMSE) is defined as

$$\text{RMSE} = \sqrt{\sum_{u,i}(R_{ui} - \hat{R}_{ui})^2/\mathcal{T}}$$

  Above, $\hat{R}_{ui}$ is the predicted rating of user u on item i and $\mathcal{T}$ is the number of ratings in testing set.

Parameter settings In our settings, the number of latent variables K is set to 100. The parameters a', b', a, c', d', c, e', f', e, g', h', g, m', n' and m in Section 4 are all set to 0.3 [15].

6.2 Overall Results and Analysis

We run top-20 recommendations using all the four approaches on all of the three datasets. Figure 2 reports the average results of the NMR, NMP and RMSE achieved by the four approaches. Overall, TSNPF outperforms the others on all three datasets. First, our TSNPF achieves considerably higher NMR than others, especially HPF and CuPF. This indicates that, compared to the items recommended by other approaches, a much higher fraction of all relevant items are indeed recommended by TSNPF to the respective users. Next, the NMP is high for all approaches and that of TSNPF is the highest. This indicates that all approaches are capable of returning high ratios of relevant items in their recommendations to users and TSNPF is overall the most effective one. Furthermore, the RMSE of TSNPF is also the best (smallest). Compared with HPF and CuPF, TSNPF gains clear RMSE improvements on all of the three datasets. MR3 outperforms HPF and CuPF, but is slightly better than TSNPF only in terms of NMP and RMSE on the Ciao dataset.

As recommender systems are more concerned about recommending more relevant items, i.e., achieving higher NMR, TSNPF performs best among all the four approaches in comparison. These experimental results demonstrate that TSNPF yields higher-quality recommendations.

Both HPF and CuPF use rating matrix only. More concretely, HPF only utilizes the ratings by individual users on items to generate user preferences and item attributes. CuPF attemps to capture the coupling relations between users and rating popularity. To predict the rating of user u on item i, CuPF requires that at least another user v and another item j exist in the training rating matrix such that both items i and j have been rated by user v. This is very demanding. According to our statistics, more than half of the testing ratings cannot be predicted by CuPF in all of the three datasets [3]. This indicates that CuPF is unsuitable for sparse data. Although MR3 utilizes all three types of data, it is simply a linear combination of two existing methods without sophisticated designs. The performance gain of TSNPF implies that jointly modeling the three types of data results in clearly better item recommendations.

6.3 Effects of Reviews and Social Relationships

Note that it is unnecessary for TSNPF that every rating is associated with a review, since we model the topic intensities of users/items based on all reviews related with users/items. Even if no review of a user/item is available, TSNPF is still able to process the relevant data. In case reviews are missing, there is no process of generating topic intensities. As a result, for an item, the ratings on it dominates its attributes. A user's preference is dominated by the ratings he/she post and his/her friends' preferences. To investigate the effect of each data type on TSNPF, we eliminate either reviews or social relationships, or both of them from TSNPF:

- **TSNPF\R** eliminates the effect of reviews by removing step 1 in the generative process (Section 4).
- **TSNPF\S** eliminates the effect of social relationships by removing step 3 in the generative process (Section 4).
- **TSNPF\R\S** eliminates the effect of both reviews and social relationships by removing steps 1 and 3 in the generative process (Section 4). As a result, it becomes HPF.

The recommendation results of TSNPF and its three variants are shown in Figure 3. TSNPF\R and TSNPF\S performs better than TSNPF\R\S on all three datasets, which suggests that both reviews and social relationships contain useful information for item recommendations. However, compared with TSNPF, the performance of a variant degrades when either reviews or social relationships are eliminated. The overall results indicate that TSNPF can make very good use of the heterogeneous information in reviews and social relationships.

[3] For testing data, we randomly decide the ratings from 1 to M.

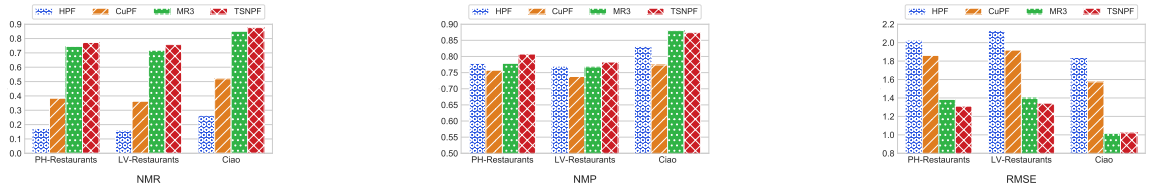


Figure 2: Performance of four approaches on three datasets

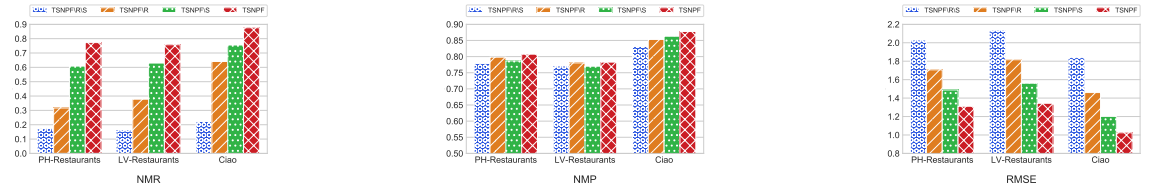


Figure 3: Effects of reviews and social relationships

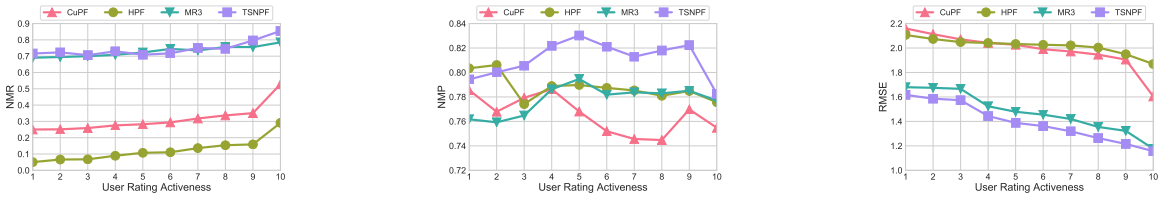


Figure 4: Effect of user rating activeness

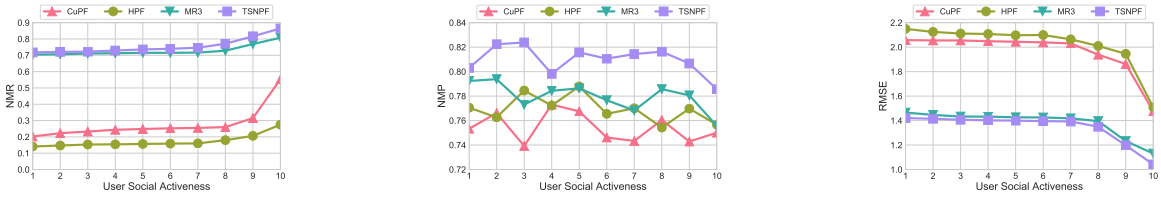


Figure 5: Effect of user social activeness

6.4 The Sensitivity of User Activeness

We treat NMR, NMP and RMSE as a function of user activeness and investigate how the recommendation performance varies across users of different types. We split all users in each of the three datasets into 10 groups ranked by user activeness. The first group is the bottom 10% users who are the least active, i.e., having rated fewest items. The 10th group contains the top 10% most active users. Figure 4 shows the performance results of NMR and NMP at top-20 recommendations and RMSE with different user types on the PH-Restaurants dataset. Due to the page limit, we omit the similar results on other datasets. The results show that all approaches tend to perform better on more active users as such users generate more data that can be used in the models. Again, we see that TSNPF outperforms the others in all but a few cases. This implies that TSNPF exploits larger amounts of user-generated data more effectively.

We also study the effect of the number of users' friends. Similarly, we split all users in each of the three datasets into 10 groups. The first group is the bottom 10% users who are the least sociable, i.e., having fewest friends. The 10th group contains the top 10% most sociable users. Figure 5 shows the performance results of NMR, NMP at top-20 recommendations and RMSE with different user types on the PH-Restaurants dataset.

The results show that all approaches tend to perform better on more sociable users as such users have more social relationships with other users that can be exploited by the models. Again, TSNPF outperforms the others.

7 CONCLUSIONS AND FUTURE WORK

We propose the Topic Social Network Poisson Factorization (TSNPF) to jointly model rating matrix, review texts and social relationships in a comprehensive manner. TSNPF extracts topics of users and

items reasonably as well as supports measuring of user similarities naturally. To address the data sparsity in high-quality recommendations, we use the Gamma-Poission generative process which only uses non-zero observations to model item attributes and user preferences. Experimental results show that TSNPF significantly outperforms alternative methods in item recommendation.

There are other kinds of heterogeneous data in addition to ratings, social relations and review, *e.g.*, metadata and visual content. A potential future work is to integrate them into our model to make even better recommendations. In addition, TSNPF only iterates over the non-zero observations. Thus, the coordinate ascent algorithm in our TSNPF is efficient. However, it is still difficult to fit TSNPF to very large datasets and therefore a stochastic variational inference algorithm [19] is needed in such cases. Furthermore, our approach uses a directed graphical model (Bayesian network) to capture user preferences and item features. Nevertheless, it is interesting to use a neural network instead of the graphical model, as studies show that neural networks can approximate arbitrarily complex functions [3] and thus have more powerful expression ability.

A APPENDIX

A.1 Derivation of the Complete Conditionals

To facilitate inference, we want to make TSNPF conditionally conjugate. To achieve this, we introduce some auxiliary vector variables (x^I_{iw}, x^U_{uw}, z_{uvi} and y_{ui}) in which each element is sampled from a Poisson distribution. Due to the given Poisson observations and Gamma priors of variables, the complete conditionals of these variables are still Gamma distributions. Hence, we use these auxiliary variables instead of actual observations to derive the complete conditionals of Gamma variables. Take ϵ_{wk} as an example, as described in the generative process, the conjugate prior of ϵ_{wk} is

$$\hat{p}(\epsilon_{wk}) = \frac{\varepsilon_w^a}{\Gamma(a)}\epsilon_{wk}^a \exp(-\varepsilon_w\epsilon_{wk})$$
$$= \overbrace{\hat{h}(\epsilon_{wk})}^{\text{Base measure}} \exp\Big((a-1,-\varepsilon_w)^\top(\log\epsilon_{wk},\epsilon_{wk}) - \overbrace{A\big((a-1,-\varepsilon_w)\big)}^{\text{Log-normalizer}}\Big).$$

Here, $(a-1,-\varepsilon_w)$ and $(\log\epsilon_{wk},\epsilon_{wk})$ are the natural parameters and sufficient statistics of this prior, respectively. Given the auxiliary variables x^I and x^U and other variables $\pi_{:,k}$ and $\sigma_{:,k}$, the conditional distribution is described as follows:

$$\begin{aligned}
&p(\epsilon_{wk}|x^I,x^U,\pi_{:,k},\sigma_{:,k})\\
&\propto \hat{p}(\epsilon_{wk})\prod_i p(x^I_{iw,k}|\epsilon_{wk},\pi_{ik})\prod_u p(x^U_{uw,k}|\epsilon_{wk},\sigma_{uk})\\
&\propto \hat{p}(\epsilon_{wk})\prod_i \frac{(\epsilon_{wk}\sigma_{uk})^{x^U_{uw,k}}\exp(-\epsilon_{wk}\sigma_{uk})}{x^U_{uw,k}!}\prod_u p(x^U_{uw,k})\\
&\propto \hat{p}(\epsilon_{wk})\prod_i\Big(\frac{1}{x^I_{iw,k}!}\exp\Big((\log\epsilon_{wk}+\log\sigma_{uk})x^U_{uw,k}-\epsilon_{wk}\sigma_{uk}\Big)\Big)\\
&\prod_u p(x^U_{uw,k})\\
&\propto \hat{h}(\epsilon_{wk})\prod_i\frac{1}{x^I_{iw,k}!}\exp\Big(\overbrace{\Big(a+\sum_i x^I_{iw,k}-1,-(\varepsilon_w+\sum_i\pi_{ik})\Big)}^{\eta}{}^\top\\
&(\log\epsilon_{wk},\epsilon_{wk}) - A(\eta)\Big)\prod_u p(x^U_{uw,k})\\
&\propto \overbrace{(\hat{h}(\epsilon_{wk})\prod_i\frac{1}{x^I_{iw,k}!}\prod_u\frac{1}{x^U_{uw,k}!})}^{\text{New base measure } h(\epsilon_{wk})}\\
&\exp\Big(\overbrace{\Big(a+\sum_i x^I_{iw,k}+\sum_u x^U_{uw,k}-1,-(\varepsilon_w+\sum_i\pi_{ik}+\sum_u\sigma_{uk})\Big)}^{\eta}{}^\top\\
&(\log\epsilon_{wk},\epsilon_{wk}) - \overbrace{A(\eta)}^{\text{New log-normalizer}}\Big)
\end{aligned}$$

Subsequently, $\Big(a+\sum_i x^I_{iw,k}+\sum_u x^U_{uw,k}-1,-(\varepsilon_w+\sum_i\pi_{ik}+\sum_u\sigma_{uk})\Big)$ are the natural parameters of the conditional distribution of $p(\epsilon_{wk}|x^I,x^U,\pi_{:,k},\sigma_{:,k})$. Thus,

$$\epsilon_{wk}|x^I,x^U,\epsilon_{:,k},\pi_{:,k},\sigma_{:,k} \sim \text{Gamma}(a+\sum_i x^I_{iw,k}+\sum_u x^U_{uw,k},\varepsilon_w+\sum_i\pi_{ik}+\sum_u\sigma_{uk}) \tag{2}$$

For conditional distribution of ε_w, the latent topic rate,

$$\begin{aligned}
&p(\varepsilon_w|\epsilon_{:,k}) \propto \hat{h}(\varepsilon_w)\exp\Big((a'-1,-b)^\top(\log\varepsilon_w,\varepsilon_w) - A(a',b')\Big)\\
&\prod_k\frac{(\epsilon_{wk})^{a-1}}{\Gamma(a)}\exp\Big((a,-\epsilon_{wk})^\top(\log\varepsilon_w,\varepsilon_w)\Big)\\
&\propto \Big(\hat{h}(\varepsilon_w)\prod_k\frac{(\epsilon_{wk})^{a-1}}{\Gamma(a)}\Big)\exp\Big(\Big(a'+Ka-1,-(b+\sum_k\epsilon_{wk})\Big)^\top\\
&(\log\varepsilon_w,\varepsilon_w)\Big)\\
&\Rightarrow\ \varepsilon_w|\epsilon_{:,k}\sim\text{Gamma}(a'+Ka,b'+\sum_k\epsilon_{wk})
\end{aligned}$$

The variational parameters in the conditional distributions of other variables (π_{ik}, ϖ_i, β_{ik}, α_i, σ_{uk}, ς_u, θ_{uk} and ϑ_u) are derived likewise.

The final latent variables are the auxiliary variables. Recall that each x^I_{iw} or x^U_{uw} is a K-vector of Poisson counts that sum to the observations C^I_{iw} or C^U_{uw}, respectively; each z_{uvi}, in the $2K$ simplex,

or y_{ui}, in the $4K$ simplex, is a vector of Poisson counts that sum to the observations S_{uvi} and R_{ui}, respectively. It is proved that the conditional distribution of a set of Poisson variables, given their sum, is a multinomial for which the parameters are their normalized set of rates [8, 25]. Thus, the complete conditionals for these vectors are

$$x^I_{iw}|\pi_i, \epsilon_w, C^I_{iw} \sim \text{Mult}(C^I_{iw}, \frac{\pi_i \epsilon_w}{\sum_k \pi_{ik}\epsilon_{wk}}) \tag{3}$$

$$x^U_{uw}|\sigma_u, \epsilon_w, C^U_{uw} \sim \text{Mult}(C^U_{uw}, \frac{\sigma_u \epsilon_w}{\sum_k \sigma_{uk}\epsilon_{wk}})$$

$$z_{uvi}|\sigma_u, \theta_u, \sigma_v, \theta_v, S_{uvi} \sim \text{Mult}\Big(S_{uvi}, \frac{(\sigma_u, \theta_u) \odot (\sigma_v, \theta_v)}{\sum_k (\sigma_{uk}\sigma_{vk} + \theta_{uk}\theta_{vk})}\Big)$$

$$y_{ui}|\pi_i, \beta_i, \sigma_u, \theta_u, R_{ui} \sim \text{Mult}\Big(R_{ui}, \frac{(\pi_i, \pi_i, \beta_i, \beta_i) \odot (\sigma_u, \theta_u, \sigma_u, \theta_u)}{\sum_k (\pi_{ik}\sigma_{uk} + \pi_{ik}\theta_{uk} + \beta_{ik}\sigma_{uk} + \beta_{ik}\theta_{uk})}\Big)$$

Above, $\odot$ denotes the element-wise multiplication operation.

A.2 Derivation of the Parameters Update

The objective of variational inference is to minimize the KL divergence between an exponential family member q and the posterior p. For similarity, we use $\boldsymbol{X}$ and $\boldsymbol{V}$ to denote the set of all observations and the set of all variational variables, respectively. Suppose $\boldsymbol{\Lambda}$ is the parameters of the distribution $q(\boldsymbol{V})$ that governs $\boldsymbol{V}$, and the distribution $q(\boldsymbol{V})$ and the conditional distribution $p(\boldsymbol{V}|\boldsymbol{X})$ are both in the exponential families. In this case, we let $\boldsymbol{\Lambda}$ be the natural parameters of $q(\boldsymbol{V})$. Thus,

$$\begin{aligned} p(\boldsymbol{V}|\boldsymbol{X}) &= h(\boldsymbol{V}) \exp\Big(\eta(\boldsymbol{V}|\boldsymbol{X}, p)^\top T(\boldsymbol{X}) - A(\eta(\boldsymbol{V}|\boldsymbol{X}, p))\Big) \\ q(\boldsymbol{V}) &= h(\boldsymbol{V}) \exp\Big(\boldsymbol{\Lambda}^\top T(\boldsymbol{V}) - A(\boldsymbol{\Lambda})\Big) \end{aligned}$$

Above, $\eta(\cdot)$, $T(\cdot)$, $h(\cdot)$ and $A(\cdot)$ are the functions of natural parameters, sufficient statistics, base measure and log-normalizer, respectively. Thus, $\boldsymbol{\Lambda} = \eta(\boldsymbol{V}, q)$. In practice, we often transfer this minimization into maximization of a lower bound on the logarithm of the marginal probability of the observations $\log p(\boldsymbol{X}, \boldsymbol{V})$ called the evidence lower bound (ELBO) [22] of q. It is defined as $\mathcal{L}(q) = \mathbb{E}_q[\log p(\boldsymbol{X}, \boldsymbol{V})] - \mathbb{E}_q[\log q(\boldsymbol{V})]$. The KL divergence is equal to the negative ELBO up to $\log p(\boldsymbol{X})$ which is a constant as it does not depend on q:

$$\begin{aligned} \text{KL}\Big(q(\boldsymbol{V})||p(\boldsymbol{V}|\boldsymbol{X})\Big) &= \mathbb{E}_q[\log q(\boldsymbol{V}) - \mathbb{E}_q[\log p(\boldsymbol{V}|\boldsymbol{X})] \\ &= \mathbb{E}_q[\log q(\boldsymbol{V})] - \mathbb{E}_q[\log p(\boldsymbol{X}, \boldsymbol{V})] + \log p(\boldsymbol{X}) \\ &= -\mathcal{L}(q) + \text{const} \end{aligned}$$

Besides,

$$\begin{aligned} \mathcal{L}(q) &= \mathbb{E}_q[\log p(\boldsymbol{X}, \boldsymbol{V})] - \mathbb{E}_q[\log q(\boldsymbol{V})] \\ &= \mathbb{E}_q[\log p(\boldsymbol{V}|\boldsymbol{X})] - \mathbb{E}_q[\log q(\boldsymbol{V})] + \text{const} \\ &= \mathbb{E}_q[\eta(\boldsymbol{V}|\boldsymbol{X}, p)^\top \nabla_{\boldsymbol{\Lambda}} A(\boldsymbol{\Lambda})] - \boldsymbol{\Lambda}^\top \nabla_{\boldsymbol{\Lambda}} A(\boldsymbol{\Lambda}) + A(\boldsymbol{\Lambda}) + \text{const} \\ &\quad \Big(\text{Because } \mathbb{E}_q[T(\boldsymbol{V})] = \nabla_{\boldsymbol{\Lambda}} A(\boldsymbol{\Lambda})\Big) \\ &\quad \Rightarrow \nabla_{\boldsymbol{\Lambda}} \mathcal{L} = \nabla^2_{\boldsymbol{\Lambda}} A(\boldsymbol{\Lambda}) \Big(\mathbb{E}_q[\eta(\boldsymbol{V}|\boldsymbol{X}, p)] - \boldsymbol{\Lambda}\Big) \end{aligned}$$

Setting the gradient $\nabla_{\boldsymbol{\Lambda}} \mathcal{L}$ to be zero, we get the closed form parameters update: $\boldsymbol{\Lambda} = \mathbb{E}_q[\eta(\boldsymbol{V}|\boldsymbol{X}, p)]$.

When $q(\boldsymbol{V})$ is a mean-field family member, the natural parameters of $q(\boldsymbol{V})$ are just the Cartesian product of its corresponding elements, respectively. Suppose $\boldsymbol{V} = \{v_1, ..., v_{|V|}\}$ and $\lambda_1, ...\lambda_V$ are the natural parameters of the distributions $q(v_1), ..., q(v_{|V|})$, respectively. Then $\boldsymbol{\Lambda} = (\lambda_1, ..., \lambda_{|V|})$. In addition, directly getting $\eta(\boldsymbol{V}|\boldsymbol{X}, p)$ is difficult as it involves all of the variable and observations. Thus we usually iteratively optimize every natural parameter λ_i for $1 \le i \le |V|$ using coordinate ascent algorithm by holding the parameters of all other variables fixed. Note that in this case these fixed variables become observations and the conditional distribution of a variable becomes the corresponding complete conditional. Each λ_i equals to the expected natural parameter (under q) of the complete conditional of v_i, i.e.,

$$\lambda_i = \mathbb{E}_q[\eta(v_i|\boldsymbol{X}, \boldsymbol{V}\backslash v_i, p)]$$

More details about coordinate ascent algorithm can be found in [4, 19].

By applying the derivations above into our TSNPF, we can easily derive the closed form variational parameters update in the main body of the paper. For simplicity, here we only give two examples: 1). μ_{wk}, the variational parameters of ϵ_{wk} which is sampled from a Gamma distribution and 2). χ^I_{iw}, the variational parameters of x^I_{iw} of which every element $x^I_{iw,k}$ is sampled from a Poisson distribution but the conditional distribution given other observations and variables is a multinomial. The updates for the variational parameters of the other Gamma and multinomial latent variables are similarly derived.

From Eq.(2), the natural parameters of the conditional distribution $p(\epsilon_{wk}|x^I, x^U, \pi_{:,k}, \sigma_{:,k}))$ and the distribution $q(\epsilon_{wk}|\mu_w k)$ are $\Big(a + \sum_i x^I_{iw,k} + \sum_u x^U_{uw,k} - 1, -(\varepsilon_w + \sum_i \pi_{ik} + \sum_u \sigma_{uk})\Big)$ and $(\mu^{shp}_{wk} - 1, -\mu^{rte}_{wk})$, respectively. As the natural parameters $(\mu^{shp}_{wk} - 1, -\mu^{rte}_{wk})$ are the affine transformation of shape and rate parameters—μ^{shp}_{wk} and μ^{rte}_{wk}, the update of variational parameters of gamma variable ϵ_{wk} are just the expectations of the shape and rate parameters of the complete conditional distributions under q, i.e.,

$$\begin{aligned} \mu^{shp}_{wk} &= \mathbb{E}_q[a + \sum_i x^I_{iw,k} + \sum_u x^U_{uw,k}] \\ &= a + \sum_i C^I_{iw}\chi^I_{iw,k} + \sum_u C^U_{uw}\chi^U_{uw,k} \\ \mu^{rte}_{wk} &= \mathbb{E}_q[\varepsilon_w + \sum_i \pi_{ik} + \sum_u \sigma_{uk}] \\ &= \frac{\iota^{shp}_w}{\iota^{rte}_w} + \sum_i \frac{\rho^{shp}_{ik}}{\rho^{shp}_{ik}} + \sum_u \frac{\nu^{shp}_{uk}}{\nu^{shp}_{uk}} \end{aligned}$$

Notice that the expectation of $x^I_{iw,k}$ equals to k-th probability of the multinomial $\chi^I_{iw,k}$ times the word count C^I_{iw}, i.e., $\mathbb{E}_q[x^I_{iw,k}] = C^I_{iw}\chi^I_{iw,k}$.

The natural parameters of $x^I_{iw,k}$ is $\log \chi^I_{iw,k}$. Due to the assumption of mean-field family, the natural parameters of x^I_{iw} are just the Cartesian product of every $\log \chi^I_{iw,k}$, i.e., $\eta(x^I_{iw}) = (\log \chi^I_{iw,1}, ..., \log \chi^I_{iw,K})$. From Eq. (3), the conditional distribution $p(x^I_{iw}|\pi_i, \epsilon_w, C^I_{iw})$ is a multinomial. The natural parameters of this conditional distribution are the logarithms of event probabilities, i.e., $\eta(x^I_{iw}|\pi_i, \epsilon_w, C^I_{iw})$ $= \Big((\log \pi_{i1} + \log \epsilon_{w1}) - r, ..., (\log \pi_{iK} + \log \epsilon_{wK}) - r\Big)$ where $r =$

$\log(\sum_k \pi_{ik}\epsilon_{wk})$. Thus,

$$\begin{aligned}\log \chi^I_{iw,k} &= \mathbb{E}_q[\log \pi_{ik} + \log \epsilon_{wk} - r] \Rightarrow \\ \chi^I_{iw,k} &= \exp(\mathbb{E}_q[\log \pi_{ik} + \log \epsilon_{wk} - r]) \\ &\propto \exp\left(\Psi(\rho^{shp}_{ik}) - \log \rho^{rte}_{ik} + \Psi(\mu^{shp}_{wk}) - \log \mu^{rte}_{wk}\right)\end{aligned}$$

Above, $\Psi(\cdot)$ is the digamma function. This update comes from the expectation of the log of a Gamma variable, e.g., $\mathbb{E}_q[\log \pi_{ik}] = \Psi(\rho^{shp}_{ik}) - \log \rho^{rte}_{ik}$. Let

$$\hat{\chi}^I_{iw,k} = \exp\left(\Psi(\rho^{shp}_{ik}) - \log \rho^{rte}_{ik} + \Psi(\mu^{shp}_{wk}) - \log \mu^{rte}_{wk}\right)$$

Then

$$\chi^I_{iw,k} = \frac{\hat{\chi}^I_{iw,k}}{\sum_k \hat{\chi}^I_{iw,k}}. \tag{4}$$

Eq. (4) guarantees that χ^I_{iw} is a K-vector whose elements sum to one.

ACKNOWLEDGMENTS

REFERENCES

[1] Xinlong Bao, Lawrence Bergman, and Rich Thompson. 2009. Stacking recommendation engines with additional meta-features. In *RecSys*. 109–116.

[2] Yang Bao, Hui Fang, and Jie Zhang. 2014. TopicMF: Simultaneously Exploiting Ratings and Reviews for Recommendation. In *AAAI*. 2–8.

[3] Andrew R. Barron. 1994. Approximation and Estimation Bounds for Artificial Neural Networks. *Machine Learning* 14, 1 (1994), 115–133.

[4] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. 2017. Variational inference: A review for statisticians. *J. Amer. Statist. Assoc.* 112, 518 (2017), 859–877.

[5] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *JMLR* 3 (2003), 993–1022.

[6] John F. Canny. 2004. GaP: a factor model for discrete data. In *SIGIR*. 122–129.

[7] Rose Catherine and William W. Cohen. 2017. TransNets: Learning to Transform for Recommendation. In *RecSys*. 288–296.

[8] Ali Taylan Cemgil. 2009. Bayesian inference for nonnegative matrix factorisation models. *Computational intelligence and neuroscience* 2009 (2009).

[9] Muthusamy Chelliah and Sudeshna Sarkar. 2017. Product Recommendations Enhanced with Reviews. In *RecSys*. 398–399.

[10] Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan S. Kankanhalli. 2018. Aspect-Aware Latent Factor Model: Rating Prediction with Ratings and Reviews. In *WWW*. 639–648.

[11] Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J. Smola, Jing Jiang, and Chong Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In *KDD*. 193–202.

[12] Trong Dinh Thac Do and Longbing Cao. 2018. Coupled Poisson Factorization Integrated With User/Item Metadata for Modeling Popular and Sparse Ratings in Scalable Recommendation. In *AAAI*.

[13] Andrew Gelman et al. 2006. Prior distributions for variance parameters in hierarchical models. *Bayesian analysis* 1, 3 (2006), 515–534.

[14] Prem Gopalan, Laurent Charlin, and David M. Blei. 2014. Content-based recommendations with Poisson factorization. In *NIPS*. 3176–3184.

[15] Prem Gopalan, Jake M. Hofman, and David M. Blei. 2015. Scalable Recommendation with Hierarchical Poisson Factorization. In *UAI*. 326–335.

[16] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. 2015. TrustSVD: Collaborative Filtering with Both the Explicit and Implicit Influence of User Trust and of Item Ratings. In *AAAI*. 123–129.

[17] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW*. 507–517.

[18] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. TriRank: Review-aware Explainable Recommendation by Modeling Aspects. In *CIKM*. 1661–1670.

[19] Matthew D. Hoffman, David M. Blei, Chong Wang, and John William Paisley. 2013. Stochastic variational inference. *JMLR* 14, 1 (2013), 1303–1347.

[20] Guang-Neng Hu, Xin-Yu Dai, Yunya Song, Shujian Huang, and Jiajun Chen. 2015. A Synthetic Approach for Recommendation: Combining Ratings, Social Relations, and Reviews. In *IJCAI*. 1756–1762.

[21] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM*. 263–272.

[22] Tommi S. Jaakkola and Michael I. Jordan. 1996. Computing upper and lower bounds on likelihoods in intractable networks. In *UAI*. 340–348.

[23] Mohsen Jamali and Martin Ester. 2009. *TrustWalker*: a random walk model for combining trust-based and item-based recommendation. In *KDD*. 397–406.

[24] Meng Jiang, Peng Cui, Rui Liu, Qiang Yang, Fei Wang, Wenwu Zhu, and Shiqiang Yang. 2012. Social contextual recommendation. In *CIKM*. 45–54.

[25] Norman L Johnson, Adrienne W Kemp, and Samuel Kotz. 2005. *Univariate discrete distributions*. Vol. 444. John Wiley & Sons.

[26] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. 1999. An Introduction to Variational Methods for Graphical Models. *Machine Learning* 37, 2 (1999), 183–233.

[27] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *IEEE Computer* 42, 8 (2009), 30–37.

[28] Guang Ling, Michael R. Lyu, and Irwin King. 2014. Ratings meet reviews, a combined approach to recommend. In *RecSys*. 105–112.

[29] Jialu Liu, Jingbo Shang, Chi Wang, Xiang Ren, and Jiawei Han. 2015. Mining Quality Phrases from Massive Text Corpora. In *SIGMOD*. 1729–1744.

[30] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Coevolutionary Recommendation Model: Mutual Learning between Ratings and Reviews. In *WWW*. 773–782.

[31] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. 2008. SoRec: social recommendation using probabilistic matrix factorization. In *CIKM*. 931–940.

[32] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *WSDM*. 287–296.

[33] Julian J. McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*. 165–172.

[34] Nikolaos Pappas and Andrei Popescu-Belis. 2013. Sentiment analysis of user comments for one-class collaborative filtering over ted talks. In *SIGIR*. 773–776.

[35] Stefan Pero and Tomás Horváth. 2013. Opinion-Driven Matrix Factorization for Rating Prediction. In *UMAP*. 1–13.

[36] Carsten Peterson and James R. Anderson. 1987. A Mean Field Theory Learning Algorithm for Neural Networks. *Complex Systems* 1, 5 (1987), 995–1019.

[37] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *NIPS*. 1257–1264.

[38] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*. 285–295.

[39] Brajendra C Sutradhar and Zhende Qu. 1998. On approximate likelihood inference in a Poisson mixed model. *Canadian Journal of Statistics* 26, 1 (1998), 169–186.

[40] Yunzhi Tan, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Rating-Boosted Latent Topics: Understanding Users and Items with Ratings and Reviews. In *IJCAI*. 2640–2646.

[41] Jiliang Tang, Xia Hu, Huiji Gao, and Huan Liu. 2013. Exploiting Local and Global Social Context for Recommendation. In *IJCAI*. 2712–2718.

[42] Martin J. Wainwright and Michael I. Jordan. 2008. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning* 1, 1-2 (2008), 1–305.

[43] Chong Wang and David M. Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *KDD*. 448–456.

[44] Mike West. 1993. Approximating posterior distributions by mixture. *Journal of the Royal Statistical Society. Series B (Methodological)* (1993), 409–422.

[45] Xindong Wu, Xingquan Zhu, Gong-Qing Wu, and Wei Ding. 2014. Data Mining with Big Data. *IEEE Trans. Knowl. Data Eng.* 26, 1 (2014), 97–107.

[46] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. In *KDD*. 353–362.

[47] Wei Zhang and Jianyong Wang. 2016. Integrating Topic and Latent Factors for Scalable Personalized Review-based Rating Prediction. *IEEE Trans. Knowl. Data Eng.* 28, 11 (2016), 3013–3027.

[48] Wei Zhang, Quan Yuan, Jiawei Han, and Jianyong Wang. 2016. Collaborative Multi-Level Embedding Learning from Reviews for Rating Prediction. In *IJCAI*. 2986–2992.

[49] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W. Bruce Croft. 2017. Joint Representation Learning for Top-N Recommendation with Heterogeneous Information Sources. In *CIKM*. 1449–1458.

[50] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *SIGIR*. 83–92.

[51] Tong Zhao, Chunping Li, Mengya Li, Qiang Ding, and Li Li. 2013. Social recommendation incorporating topic mining and social trust analysis. In *CIKM*. 1643–1648.

[52] Lei Zheng, Vahid Noroozi, and Philip S. Yu. 2017. Joint Deep Modeling of Users and Items Using Reviews for Recommendation. In *WSDM*. 425–434.

Persona-Aware Tips Generation*

Piji Li
[1]Tencent AI Lab
Shenzhen, China
[2]The Chinese University of Hong Kong
Hong Kong, China
pijili@tencent.com

Zihao Wang
The Chinese University of Hong Kong
Hong Kong, China
zhwang@se.cuhk.edu.hk

Lidong Bing
R&D Center Singapore
Alibaba DAMO Academy
l.bing@alibaba-inc.com

Wai Lam
The Chinese University of Hong Kong
Hong Kong, China
wlam@se.cuhk.edu.hk

ABSTRACT

Tips, as a compacted and concise form of reviews, were paid less attention by researchers. In this paper, we investigate the task of tips generation by considering the "persona" information which captures the intrinsic language style of the users or the different characteristics of the product items. In order to exploit the persona information, we propose a framework based on adversarial variational auto-encoders (aVAE) for persona modeling from the historical tips and reviews of users and items. The latent variables from aVAE are regarded as persona embeddings. Besides representing persona using the latent embeddings, we design a persona memory for storing the persona related words for users and items. Pointer Network is used to retrieve persona wordings from the memory when generating tips. Moreover, the persona embeddings are used as latent factors by a rating prediction component to predict the sentiment of a user over an item. Finally, the persona embeddings and the sentiment information are incorporated into a recurrent neural networks based tips generation component. Extensive experimental results are reported and discussed to elaborate the peculiarities of our framework.

CCS CONCEPTS

• **Computing methodologies** → **Natural language generation**; • **Information systems** → **Recommender systems**; **Personalization**.

KEYWORDS

Abstractive Tips Generation; Rating Prediction; Persona Modeling; Adversarial Variational Auto-Encoders.

*The work described in this paper was partially supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Codes: 14203414) and the Direct Grant of the Faculty of Engineering, CUHK (Project Code: 4055093).

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313496

ACM Reference Format:
Piji Li, Zihao Wang, Lidong Bing, and Wai Lam. 2019. Persona-Aware Tips Generation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313496

1 INTRODUCTION

Tips, specifically defined by Yelp[1], as a compacted and concise form of reviews, have unique advantages for helping users get a quick insight over an item. Conventional reviews are extensively studied for rating prediction [26, 37] and review generation [6, 29, 35, 40], while tips are paid relatively less attention. In [21], the tip information was explored for tips generation and rating prediction for the first time. The rationality for performing the joint task can be attributed to "writing some tips and giving a numerical rating are two facets of a user's product assessment action, expressing the user experience and feelings". Moreover, compared with reviews, tips are likely more consistent with the rating score with respect to sentiment tendency, because of its intrinsic form, i.e. compacted and concise.

In this paper, we investigate another dimension, namely user persona, which is plausibly helpful for the task of tips generation and has not be considered in the previous work [21]. Here the term "**persona**" denotes the characteristics of the written text by users such as wording and style. Figure 1a shows some tips for a shower radio from different users. [2] These tips clearly show different styles, although all of them have the same ratings. Some users (e.g. 1, 4 and 5) prefer short sentences and direct wordings such as "great", "easy", and "excellent" to describe the product quality and their experience directly. On the other hand, some users (e.g. 2, 3, and 6) share their experience indirectly by talking about some facts with longer sentences. Therefore, different users indeed have different "persona" style when writing tips. Figure 1b shows a few tips with different ratings from the same user for different items, we can observe that the user prefers short sentences, and moreover he has his own style (i.e. preferred vocabulary) for writing tips of different sentiments/ratings (e.g. "perfect" and "excellent" for high rating tips, and "piece of crap" for low rating tips).

[1]https://www.yelp-support.com/article/What-are-tips
[2]https://www.amazon.com/sony-icf-s79v-weather-shower-radio/dp/b00000dm9w

Tips	Rating
(1) Great fit and finish for shower.	5
(2) I selected this radio for myself several years ago and i have found that all claims for it are true.	5
(3) If your looking for a radio for your shower then look no further.	5
(4) Easy to set up stations.	5
(5) Excellent design and quality construction.	5
(6) First one lasted years just bought another one.	5

(a) Tips for the item "Sony Weather Band Shower Radio".

Tips	Rating
(1) Works perfectly in my msi wind.	5
(2) Perfect size for a home office.	5
(3) Excellent player for price.	5
(4) Wonderful docking speaker with full sound.	4
(5) I like it when it not dropping the signal.	4
(6) Works fine in a pinch.	3
(7) Piece of crap do bother.	1
(8) Revised star piece of crap.	1

(b) Tips for different items written by a particular user.

Figure 1: Example of tips.

Intuitively, the quality of abstractive tips generation can be improved if the model considers the user persona information when conducting the text generation. To do so, in this paper, we investigate an approach called **P**ersona-**A**ware **T**ips **G**eneration (**PATG**). There are two main challenges for the design of PATG: (1) How to capture and represent the persona information; (2) How to integrate the sentiment signal with the persona information to jointly control the style and the sentiment of the generated tips.

We distill persona information from all the historical tips and reviews of a user into the form of **Persona Embeddings**. Then the persona embeddings can be directly incorporated into the tips generation component as context information. Specifically, we employ variational auto-encoders (VAEs) [12] (which show strong capability in modeling latent random variables [20, 22]) to conduct persona modeling, and we regard the latent variables of VAEs as the persona embeddings. In the context of user behaviour analysis of online retailing, another indispensable party is the product item. Items also have their own intrinsic characteristics, such as product category in a coarser granularity, or specific features in a finer granularity. In this work, we personify the items and enable modeling them with "item persona", similar to modeling users. Besides distilling persona information using VAEs, we also design an external **Persona Memory** for the framework to store the persona related words for the current user and item. Pointer Networks [36] is used to retrieve appropriate words from the persona memory for generating tips.

Another obvious signal from the examples in Figure 1 is that the sentiment related wording is also bound to the sentiment ratings. For example, positive expressions are only used in tips with high ratings, no matter what persona the users have. To explore this signal for generating more accurate tips, our framework includes an auxiliary component: rating prediction with the information of users and items used for tips generation. The intuition is that if we can predict the rating of a user on an item accurately, the same input information should provide rich, if not complete, information for generating a tip satisfying that rating. Thus, in order to control the sentiment of the generated tips, we design a rating prediction component. The distilled persona embeddings are regarded as latent factors for users/items, and fed into the rating prediction component for detecting sentiment. A vectorization process is conducted on the predicted rating values and then the rating vectors are incorporated into the tips generation component as context information to control the sentiment of the generated tips.

The main contributions of our framework are summarized below:

- We develop a framework that tackles the task of persona-aware tips generation, where persona information, such as writing style and vocabulary preference, is considered for the first time to conduct the tips generation.
- In order to exploit the persona information, we design an adversarial variational auto-encoders (aVAE) based approach for persona modeling for users and items, i.e. generating persona embeddings. We employ an external memory based Pointer Networks to conduct the memory reading to retrieve more accurate persona information.
- In order to control the sentiment of the generated tips, we tightly couple an auxiliary component of rating prediction with the tips generation component. The distilled persona embeddings are used as latent factors of users and items for the sentiment rating prediction.
- Experimental results show that our framework achieves better performance than the state-of-the-art models on tips generation. Moreover, an additional observation is that the persona information can improve the performance of the auxiliary task, i.e. rating prediction.

2 FRAMEWORK DESCRIPTION

2.1 Overview

The data consists of users, items, ratings, review content, and tips. We denote the whole training corpus by $\mathcal{X} = \{\mathcal{U}, \mathcal{I}, \mathcal{R}, C, \mathcal{S}\}$, where $\mathcal{U}$ and $\mathcal{I}$ are the sets of users and items respectively, $\mathcal{R}$ is the set of ratings, C is the set of review documents, and $\mathcal{S}$ is the set of tips texts. We use C_u and $\mathcal{S}_u$ to denote all the historical reviews and tips respectively of the user u. For a quick reference, Table 1 lists all notations used in our paper.

As shown in Figure 2, our framework contains two major modules: persona modeling on the left and abstractive tips generation on the right. For modeling persona, our framework leverages the tips and reviews from each individual user or written by multiple users for the same item. Take the historical tips $\mathcal{S}_u$ of the user u as an example, we represent them using bag-of-words (BoWs) vectors $\mathbf{x}_u^s$. Then we feed $\mathbf{x}_u^s$ into the adversarial variational auto-encoders (aVAE^s) and obtain the persona embedding $\mathbf{z}_u^s$ for the user u. For the item i, we can also conduct similar persona modeling based on the historical tips $\mathcal{S}_i$ written by different users, and the obtained persona embedding is denoted as $\mathbf{z}_i^s$. The purpose of persona modeling for the item i is that when conducting tips generation for the user u, the model will also consider the tips from other users having similar interests with u, since they will disclose more characteristics of the item. We call this phenomenon *personalized collaborative*

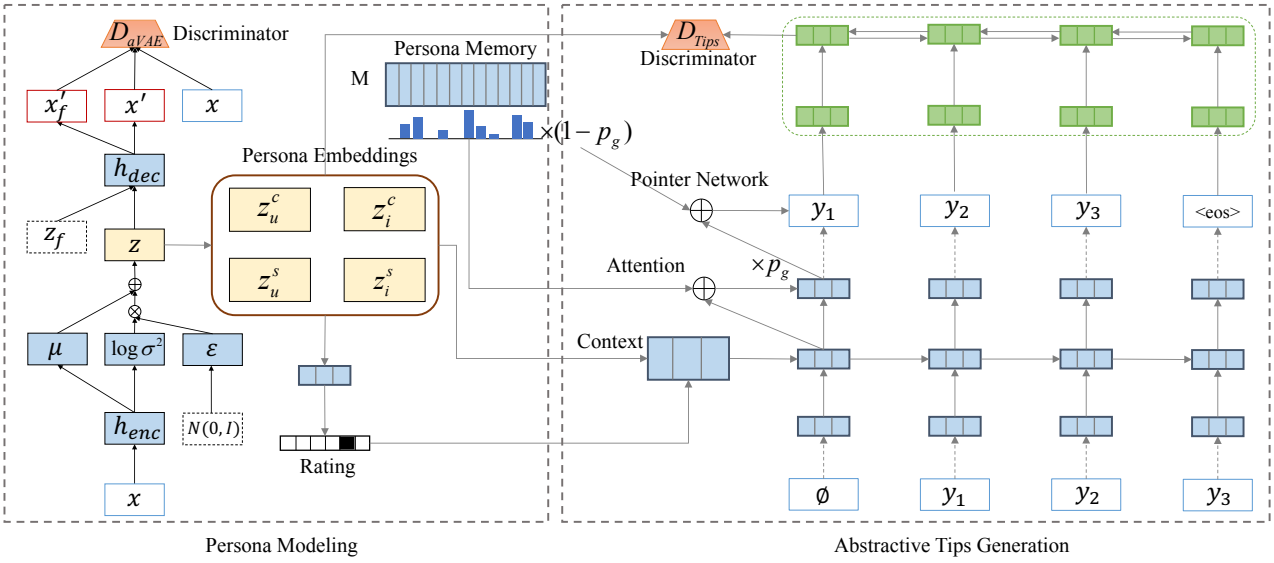


Figure 2: Our proposed framework for persona-aware abstractive tips generation.

Table 1: Glossary.

Symbol	Description
$\mathcal{X}$	training set
$\mathcal{V}$	vocabulary
$\mathcal{U}$	set of users
$\mathcal{I}$	set of items
$\mathcal{R}$	set of ratings
$\mathcal{C}$	set of reviews
$\mathcal{S}$	set of tips
$\mathcal{C}_u$	historical reviews for user u
$\mathcal{S}_u$	historical tips for user u
$\mathbf{M}$	external memory
$\mathbf{Z}$	persona embeddings
$\mathbf{E}$	word embeddings
$\mathbf{H}$	neural hidden states
$\mathbf{W}$	mapping matrix
$\mathbf{b}$	bias item
Θ	set of neural parameters
$r_{u,i}$	rating of user u to item j
σ	sigmoid function
ς	softmax function
$relu$	rectified linear unit
$tanh$	hyperbolic tangent function

influence. We also distill persona information from reviews with another aVAE model (aVAE^c) to map the historical reviews C_u and C_i to persona embeddings $\mathbf{z}_u^c$ and $\mathbf{z}_i^c$ for the user u and the item i respectively.

We design an external persona memory $\mathbf{M}$ for storing the persona related words for the current user and item which will be utilized in abstractive tips generation. In order to control the sentiment of the generated tips, the distilled persona embeddings are used as latent factors for users and items and are fed into a multilayer perceptron (MLP) based neural network component to predict the rating r. Then we transform r to a one-hot vector $\mathbf{r}$ which will be used as the sentiment controller when conducting the tips generation. For the step of tips generation, we design a sequence decoding model based on a neural network of Gated Recurrent Units (GRUs) [5]. Importantly, the persona embeddings and the rating vector are combined to construct a context vector which plays a significant role in the abstractive tips generation. In addition, Pointer Networks is used to retrieve relevant words from the persona memory $\mathbf{M}$, with a gate p_g to control the source of the next output word.

2.2 Persona Modeling

2.2.1 ***Persona Embedding Learning.*** The target of persona modeling is to distill the persona information from the users' historical tips and reviews. Some previous works in recommendation systems [26, 30, 37] employ topic modeling methods such as Latent Dirichlet Allocation (LDA) [2] or its variants to analyze the text corpus and use the latent topic distribution to represent each document. Considering the fact that our tips generation component is based on neural networks, existing topic modeling paradigms cannot be incorporated into our framework in an elegant manner. Instead, we employ the variational auto-encoders (VAEs) [12] for detecting the latent topics with neural modeling paradigm [4]. VAEs consists of two parts: inference (variational-encoder) and generation (variational-decoder). Recall that the dictionary is V. For historical tips based persona modeling, the input are the BoWs vectors $\mathbf{x}_u^s \in \mathbb{R}^{|V|}$ and $\mathbf{x}_i^s \in \mathbb{R}^{|V|}$ for the user u and the item i respectively. For convenience, we will use $\mathbf{x}$ to represent them in this section. As shown in the left part of Figure 2, for each input BoWs vector $\mathbf{x}$, the variational-encoder can map it to a latent variable $\mathbf{z} \in \mathbb{R}^K$, which can be used to generate a new variable $\mathbf{x}'$ via the variational-decoder component to reconstruct the original term vector. The target is to maximize the probability of each $\mathbf{x}$ in the dataset based

on the generation process according to:

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \tag{1}$$

For the purpose of solving the intractable integral of the marginal likelihood, a model $q(\mathbf{z}|\mathbf{x})$ is introduced as the approximation to the intractable of the true posterior $p(\mathbf{z}|\mathbf{x})$. The aim of optimization is to reduce the Kulllback-Leibler divergence (KL) between $q(\mathbf{z}|\mathbf{x})$ and $p(\mathbf{z}|\mathbf{x})$ by maximizing the variational lower bound $\mathcal{L}_{VAE}$:

$$\mathcal{L}_{VAE} = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] - D_{KL}[q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})] \tag{2}$$

In order to differentiate and optimize the lower bound $\mathcal{L}_{VAE}$, following the core idea of VAEs, we use a neural network framework for the encoder $q(\mathbf{z}|\mathbf{x})$ for better approximation. Similar to previous works [12], we assume that both the prior and posterior of the latent variables are Gaussian, i.e., $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$ and $q(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ denote the variational mean and standard deviation respectively, which can be calculated with a multilayer perceptron (MLP). Precisely, given the BoWs vector $\mathbf{x}$ of the historical tips, we first project it to a hidden space:

$$\mathbf{h}_{enc} = relu(\mathbf{W}_{xh}\mathbf{x} + \mathbf{b}_{xh}) \tag{3}$$

where $\mathbf{h}_{enc} \in \mathbb{R}^{d_h}$, $\mathbf{W}_{xh}$ and $\mathbf{b}_{xh}$ are the neural parameters. $relu(\mathbf{x}) = max(0, \mathbf{x})$ is the activation function. Then the Gaussian parameters $\boldsymbol{\mu} \in \mathbb{R}^K$ and $\boldsymbol{\sigma} \in \mathbb{R}^K$ can be obtained via a linear transformation based on $\mathbf{h}_{enc}$:

$$\begin{aligned} \boldsymbol{\mu} &= \mathbf{W}_{h\mu}\mathbf{h}_{enc} + \mathbf{b}_{h\mu} \\ \log(\boldsymbol{\sigma}^2) &= \mathbf{W}_{h\sigma}\mathbf{h}_{enc} + \mathbf{b}_{h\sigma} \end{aligned} \tag{4}$$

In order to make the sampling operation differentiable, the latent variable $\mathbf{z} \in \mathbb{R}^K$ can be calculated using the reparameterization trick:

$$\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \mathbf{I}), \quad \mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \otimes \boldsymbol{\varepsilon} \tag{5}$$

where $\boldsymbol{\varepsilon} \in \mathbb{R}^K$ is an auxiliary noise variable. This is the encoding process, and we denote all the parameters of this state as Θ_{Enc}.

Given the latent variable $\mathbf{z}$, a new vector $\mathbf{x}'$ is generated via the conditional distribution $p(\mathbf{x}|\mathbf{z})$ according to the variational-decoder:

$$\mathbf{h}_{dec} = relu(\mathbf{W}_{zh}\mathbf{z} + \mathbf{b}_{zh}) \tag{6}$$

$$\mathbf{x}' = \sigma(\mathbf{W}_{hx}\mathbf{h}_{dec} + \mathbf{b}_{hx}) \tag{7}$$

We denote all the parameters in the decoding stage using Θ_{Dec}. Finally, based on the reparameterization trick in Equation 5, we can get the analytical representation of $\mathcal{L}_{VAE}$:

$$\begin{aligned} \log p(\mathbf{x}|\mathbf{z}) &= \sum_{i=1}^{|V|} x_i \log x'_i + (1 - x_i) \cdot \log(1 - x'_i) \\ -D_{KL}[q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})] &= \tfrac{1}{2} \sum_{i=1}^{K} (1 + \log(\sigma_i^2) - \mu_i^2 - \sigma_i^2) \end{aligned} \tag{8}$$

For presentation clarity, we let:

$$\begin{aligned} \mathcal{L}_{Rec} &= -\log p(\mathbf{x}|\mathbf{z}) \\ \mathcal{L}_{KL} &= D_{KL}[q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})] \end{aligned} \tag{9}$$

and both of them need to be minimized.

We wish to employ the latent variable $\mathbf{z}$ as the distilled persona embeddings. So the quality of $\mathbf{z}$ will affect the performance of tips generation. Some previous works [9, 27, 43] have also shown that the performance of $\mathbf{z}$ is likely to be disturbed during the training procedure, especially when combining VAEs with the RNN based text generation framework. In order to enhance the performance of the typical VAEs, inspired by the ideas in [8] and [15], we employ the adversarial strategy for the training of VAEs. Generally, we design a discriminator network D_{aVAE} with a vector $\tilde{\mathbf{x}}$ as input, and the target is to recognize if $\tilde{\mathbf{x}}$ is from the true data $\mathbf{X}$ or from the generated samples $\mathbf{X}'$ by VAEs. VAEs will "fool" the discriminator D_{aVAE} by trying the best to produce high quality latent variables $\mathbf{z}$ as well as the generated sample $\mathbf{x}'$. Then the minimax game between the VAEs and the discriminator can be formulated as follows:

$$\begin{aligned} \min_{VAEs} \max_{D_{aVAE}} & \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D_{aVAE}(\mathbf{x})] \\ & + \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x})}[\log(1 - D_{aVAE}(VAE_{Dec}(\mathbf{z})))] \\ & + \mathbb{E}_{\mathbf{z}_f\sim p(\mathbf{z})}[\log(1 - D_{aVAE}(VAE_{Dec}(\mathbf{z}_f)))] \end{aligned} \tag{10}$$

where VAE_{Dec} is the decoder component of the VAEs model. $\mathbf{z}$ is the latent variable from VAEs, and $\mathbf{z}_f$ is sampled from the prior distribution of $\mathbf{z}$.

For the design of the discriminator D_{aVAE}, we simply use a multilayer perceptron to process the data.

$$\begin{aligned} \mathbf{h}^{D_v} &= tanh(\mathbf{W}_{xh}^{D_v}\tilde{\mathbf{x}} + \mathbf{b}_{xh}^{D_v}) \\ y^{D_v} &= \sigma(\mathbf{W}_{hy}^{D_v}\mathbf{h}^{D_v} + b_{hy}^{D_v}) \end{aligned} \tag{11}$$

where $\mathbf{W}_{xh}^{D_v} \in \mathbb{R}^{d_h \times |V|}$, $\mathbf{W}_{hy}^{D_v} \in \mathbb{R}^{1 \times d_h}$, $\mathbf{b}_{xh}^{D_v} \in \mathbb{R}^{d_h}$, and $b_{hy}^{D_v} \in \mathbb{R}$. The output y^{D_v} is a real value in the range of $[0, 1]$ and the value 1 means that the sample $\tilde{\mathbf{x}}$ is from the true data. We denote all the parameters in D_{aVAE} using Θ_{D_v}. The optimization objective to be maximized for D_{aVAE} is formulated as:

$$\begin{aligned} \mathcal{L}_{D_{aVAE}} = & \log(D_{aVAE}(\mathbf{x})) \\ & + \log(1 - D_{aVAE}(VAE_{Dec}(VAE_{Enc}(\mathbf{x})))) \\ & + \log(1 - D_{aVAE}(VAE_{Dec}(\mathbf{z}_f))) \end{aligned} \tag{12}$$

Then the parameters Θ_{D_v} are updated using gradient methods:

$$\Theta_{D_v} \leftarrow \Theta_{D_v} - \nabla_{\Theta_{D_v}}(-\mathcal{L}_{D_{aVAE}}) \tag{13}$$

Conditioned on the aVAE framework, we will conduct the parameter learning for VAEs Encoder, VAEs Decoder, and discriminator D_{aVAE} using different loss functions respectively. Encoder transforms the input $\mathbf{X}$ to the persona embeddings $\mathbf{Z}$. On one side, $\mathbf{Z}$ are used to reconstruct the original input. On the other side, $\mathbf{Z}$ are used to conduct the persona-aware tips generation. So the loss signals from both the aVAE and the tips generation framework are used to conduct the optimization for Θ_{Enc}:

$$\Theta_{Enc} \leftarrow \Theta_{Enc} - \nabla_{\Theta_{Enc}}(\mathcal{L}_{KL} + \mathcal{L}_{Rec} + \mathcal{L}^z_{D_{aVAE}} + \mathcal{L}_{Tips}) \tag{14}$$

where $\mathcal{L}_{KL}$ and $\mathcal{L}_{Rec}$ are the KL divergence and reconstruction loss from Equation 8. $\mathcal{L}_{Tips}$ is the loss signal from the tips generation component. $\mathcal{L}^z_{D_{aVAE}}$ is the output of D_{aVAE}:

$$\mathcal{L}^z_{D_{aVAE}} = -\log(D_{aVAE}(VAE_{Dec}(VAE_{Enc}(\mathbf{x})))) \tag{15}$$

For the parameter optimization of VAEs Decoder, we use $\mathcal{L}_{Rec}$, $\mathcal{L}_{D_{aVAE}}$, $\mathcal{L}_{Tips}$ as the loss signals:

$$\Theta_{Dec} \leftarrow \Theta_{Dec} - \nabla_{\Theta_{Dec}}(\mathcal{L}_{\text{Rec}} + \mathcal{L}_{D_{aVAE}} + \mathcal{L}_{Tips}) \tag{16}$$

Finally, the training procedure of aVAE model is shown in Algorithm 1.

Algorithm 1 Persona embedding learning.

Input: BoWs vectors of historical tips and reviews $\mathbf{X}$.
Output: The persona embeddings $\mathbf{Z}$.

Initialize $\Theta_{Enc}, \Theta_{Dec}, \Theta_{D_v}$;
while not converged **do**
  Draw $\mathbf{x}$ from p_{data}.
  Draw $\mathbf{z}_f$ from prior $p(\mathbf{z})$.
  $\mathbf{z} = VAE_{Enc}(\mathbf{x})$
  $\mathbf{x}' = VAE_{Dec}(\mathbf{z})$
  $\mathbf{x}'_f = VAE_{Dec}(\mathbf{z}_f)$
  Get $\mathcal{L}_{Rec}$, $\mathcal{L}_{KL}$, $\mathcal{L}_{D_{aVAE}}$ according to Equation 8 and 12.
  Get $\mathcal{L}_{Tips}$ from tips generation.
  Update parameters using gradient methods:
  $\Theta_{Enc} \leftarrow \Theta_{Enc} - \nabla_{\Theta_{Enc}}(\mathcal{L}_{KL} + \mathcal{L}_{Rec} + \mathcal{L}^z_{D_{aVAE}} + \mathcal{L}_{Tips})$
  $\Theta_{Dec} \leftarrow \Theta_{Dec} - \nabla_{\Theta_{Dec}}(\mathcal{L}_{\text{Rec}} + \mathcal{L}_{D_{aVAE}} + \mathcal{L}_{Tips})$
  $\Theta_{D_v} \leftarrow \Theta_{D_v} - \nabla_{\Theta_{D_v}}(-L_{D_{aVAE}})$
end while
return $\mathbf{z}$.

Feeding the historical reviews and tips representations ($\mathbf{x}^c_u$, $\mathbf{x}^c_i$, $\mathbf{x}^s_u$, and $\mathbf{x}^s_i$) into $aVAE^c$ (for reviews) and $aVAE^s$ (for tips) respectively, we can obtain four persona embeddings $\mathbf{z}^c_u$, $\mathbf{z}^c_i$, $\mathbf{z}^s_u$, and $\mathbf{z}^s_i$. These persona embeddings will be integrated into the rating prediction component and the tips generation component later.

2.2.2 ***Sentiment and Rating Modeling.*** We regard the persona embeddings as the latent factors of users and items, and feed them into a multilayer perceptron to conduct the rating prediction. The predicted ratings will be used to control the sentiment of the generated tips.

Specifically, we first map the persona embeddings to a hidden space:

$$\mathbf{h}^r = tanh(\mathbf{W}^r_{uch}\mathbf{z}^c_u + \mathbf{W}^r_{ich}\mathbf{z}^c_i + \mathbf{W}^r_{ush}\mathbf{z}^s_u + \mathbf{W}^r_{ish}\mathbf{z}^s_i + \mathbf{b}^r_h) \quad (17)$$

where $\{\mathbf{W}^r_{uch}, \mathbf{W}^r_{ich}, \mathbf{W}^r_{ush}, \mathbf{W}^r_{ish}\} \in \mathbb{R}^{d_h \times k}$ are the mapping matrices. $\mathbf{b}^r_h \in \mathbb{R}^{d_h}$ is the bias term. $tanh$ is the hyperbolic tangent activation function. The superscript r refers to variables related to the rating prediction component. For better performance, we can add more layers of non-linear transformations into our model:

$$\mathbf{h}^r_l = \sigma(\mathbf{W}^r_{hh_l}\mathbf{h}^r_{l-1} + \mathbf{b}^r_{h_l}) \quad (18)$$

where $\mathbf{W}^r_{hh_l} \in \mathbb{R}^{d_h \times d_h}$ is the mapping matrix for the variables in the hidden layers. l is the index of a hidden layer. Assume that $\mathbf{h}^r_L$ is the output of the last hidden layer. The output layer transforms $\mathbf{h}^r_L$ into a real-valued rating $\hat{r}$:

$$\hat{r} = \mathbf{W}^r_{hr}\mathbf{h}^r_L + b^r \quad (19)$$

where $\mathbf{W}^r_{hr} \in \mathbb{R}^{1 \times d_h}$ and $b^r \in \mathbb{R}$. We formulate the optimization of the parameters Θ_r as a regression problem and the loss function is formulated as:

$$\mathcal{L}^r = \frac{1}{2|\mathcal{X}|}\sum_{u \in \mathcal{U}, i \in \mathcal{I}}(\hat{r}_{u,i} - r_{u,i})^2 \quad (20)$$

where $\mathcal{X}$ represents the training set. $r_{u,i}$ is the ground truth rating assigned by the user u to the item i.

The predicted rating is a real value, not a vector, for example, $\hat{r}_{u,i} = 4.321$. In order to incorporate the rating information into the tips generation component, we cast it into an integer 4, and add a vectorization process to obtain the vector representation of rating $\hat{r}_{u,i}$. If the rating range is $[0, 5]$, we will get the rating vector $\hat{\mathbf{r}}_{u,i} = (0, 0, 0, 0, 1, 0)^T$.

2.2.3 ***External Persona Memory.*** In addition to represent persona information using the latent embeddings from aVAE, we design an external persona memory for directly storing the persona related words for both the current user u and the current item i. To build the memory, we first collect all the words for the current user u and the current item t from their historical tips. We add a filtering process to remove the stop-words and the low-frequency words. Then we get a local vocabulary storing the indices of the persona words. Recall that we have a global word embedding $\mathbf{E}$. Then we can get a sub-matrix from $\mathbf{E}$ according to the word indices. We regard this sub-matrix as persona memory. We employ Pointer Networks to retrieve persona information from the memory when generating tips. The details are described in Section 2.3.3.

2.3 Abstractive Tips Generation

2.3.1 ***Overview of Tips Generation.*** The right part of Figure 2 depicts our tips generation model. The basic element is a RNN based sequence modeling component. Pointer Networks (attention modeling and copy mechanism) is introduced to conduct the memory reading. Context information plays an important role in the task of text generation. We combine the persona embeddings and the sentiment information as the context information and construct the context vector which can control the tips text generation. At the training state, we also design a discriminator D_{Tips} to assess the quality of the generated tips. The assess value will be propagated to the RNN models to assist the parameter learning. At the operational or testing stage, we use a beam search algorithm [13] for decoding and generating the best tips given a trained model.

2.3.2 ***Sequence Modeling.*** Assume that $\mathbf{h}^s_t$ is the sequence hidden state at the time t. It depends on the input at the time t and the previous hidden state $\mathbf{h}^s_{t-1}$:

$$\mathbf{h}^s_t = f(\mathbf{h}^s_{t-1}, s_t) \quad (21)$$

$f(\cdot)$ can be the vanilla RNN, Long Short-Term Memory (LSTM) [10], or Gated Recurrent Unit (GRU) [5]. Considering that GRU has comparable performance but with less parameters and more efficient computation, we employ GRU as the basic model in our sequence modeling framework. In the case of GRU, the state updates are processed according to the following operations:

$$\begin{aligned}
\mathbf{r}^s_t &= \sigma(\mathbf{W}^s_{sr}\mathbf{s}_t + \mathbf{W}^s_{hr}\mathbf{h}^s_{t-1} + \mathbf{b}^s_r) \\
\mathbf{z}^s_t &= \sigma(\mathbf{W}^s_{sz}\mathbf{s}_t + \mathbf{W}^s_{hz}\mathbf{h}^s_{t-1} + \mathbf{b}^s_z) \\
\mathbf{g}^s_t &= \tanh(\mathbf{W}^s_{sh}\mathbf{s}_t + \mathbf{W}^s_{hh}(\mathbf{r}^s_t \odot \mathbf{h}^s_{t-1}) + \mathbf{b}^s_h) \\
\mathbf{h}^s_t &= \mathbf{z}^s_t \odot \mathbf{h}^s_{t-1} + (1 - \mathbf{z}^s_t) \odot \mathbf{g}^s_t
\end{aligned} \quad (22)$$

where $\mathbf{s}_t \in \mathbf{E}$ is the embedding vector for the word s_t of the tips and the vector is also learnt from our framework. $\mathbf{r}^s_t$ is the reset gate, $\mathbf{z}^s_t$ is the update gate. $\odot$ denotes element-wise multiplication.

In order to conduct the persona-aware tips generation, we combine all the persona embeddings and the sentiment information as

the context information and construct the context vector. Specifically, we initialize the hidden state $\mathbf{h}_0$ using the persona embeddings and the sentiment information:

$$\mathbf{h}_0^s = tanh(\mathbf{W}_{uch}^s \mathbf{z}_\mathbf{u}^\mathbf{c} + \mathbf{W}_{ich}^s \mathbf{z}_\mathbf{i}^\mathbf{c} + \mathbf{W}_{ush}^s \mathbf{z}_\mathbf{u}^\mathbf{s} + \mathbf{W}_{ish}^s \mathbf{z}_\mathbf{i}^\mathbf{s} + \mathbf{W}_{rh}^s \hat{\mathbf{r}} + \mathbf{b}_h^s) \tag{23}$$

where $\{\mathbf{z}_*^*\}$ are the persona embeddings. $\hat{\mathbf{r}}$ is the vectorization for the predicted rating $\hat{r}$. $\mathbf{W}$ and $\mathbf{b}$ are the neural parameters.

After getting all the sequence hidden states based on GRU, we feed them to the final output layer to predict the word sequence in tips.

$$\hat{\mathbf{s}}_{t+1} = \varsigma(\mathbf{W}_{hs}^s \mathbf{h}_t^s + \mathbf{b}^s) \tag{24}$$

where $\mathbf{W}_{hs}^s \in \mathbb{R}^{d \times |\mathcal{V}|}$ and $\mathbf{b}^s \in \mathbb{R}^{|\mathcal{V}|}$. $\varsigma(\cdot)$ is the softmax function. Then the word with the largest probability is the decoding result for the step $t+1$:

$$w_{t+1}^* = \arg\max_{w_i \in \mathcal{V}} \hat{\mathbf{s}}_{t+1}^{(w_i)} \tag{25}$$

At the training stage, we use negative log-likelihood (NLL) as the loss function, where I_w is the vocabulary index of the word w:

$$\mathcal{L}_{Tips} = -\sum_{w \in Tips} \log \hat{\mathbf{s}}^{(I_w)} \tag{26}$$

Note that $\mathcal{L}_{Tips}$ is also used in the persona modeling component to train the aVAE models.

At the testing stage, given a trained model, we employ the beam search algorithm [13] to find the best sequence S^* having the maximum log-likelihood.

$$S^* = \arg\max_{S \in \mathcal{S}} \sum_{w \in S} \log \hat{\mathbf{s}}^{(I_w)} \tag{27}$$

2.3.3 ***Exploiting Persona Memory***. Recall that in Section 2.2.3, we build a local personal vocabulary V_{ui} for the user u and the item i. The persona memory $\mathbf{M}_{ui}$ is extracted from the word embedding $\mathbf{E}$ using the word indices in V_{ui}. Inspired by [1], we exploit the idea of attention modeling to conduct the addressing and reading operations on the memory $\mathbf{M}_{ui}$. We can obtain the GRU hidden state $\mathbf{h}_t^s$ according to Equation (21). Then the attention weights at the time step t are calculated based on the relationship between $\mathbf{h}_t^s$ with all the word embeddings in $\mathbf{M}_{ui}$. Let $a_{i,j}$ be the attention weight between $\mathbf{h}_i^s$ and $\mathbf{m}_j$, which can be calculated using:

$$\begin{aligned} a_{i,j} &= \frac{\exp(e_{i,j})}{\sum_{j'=1}^{|V_{ui}|} \exp(e_{i,j'})} \\ e_{i,j} &= \mathbf{v}_a{}^T \tanh(\mathbf{W}_{hh}^s \mathbf{h}_i^s + \mathbf{W}_{hh}^m \mathbf{m}_j + \mathbf{b}_a) \end{aligned} \tag{28}$$

where $\mathbf{W}_{hh}^s \in \mathbb{R}^{d_h \times d_h}$, $\mathbf{W}_{hh}^m \in \mathbb{R}^{d_w \times d_h}$, $\mathbf{b}_a \in \mathbb{R}^{d_h}$, and $\mathbf{v}_a \in \mathbb{R}^{d_h}$. The attention context is obtained by the weighted linear combination of all the word embeddings in $\mathbf{M}_{ui}$:

$$\mathbf{c}_t = \sum_{j'=1}^{|V_{ui}|} a_{t,j'} \mathbf{m}_{j'} \tag{29}$$

The final hidden state $\mathbf{h}_t^{s_2}$ is the output of the second decoder GRU layer, jointly considering the word $\mathbf{s}_t$, the previous hidden state $\mathbf{h}_{t-1}^{s_2}$, and the attention context $\mathbf{c}_t$:

$$\mathbf{h}_t^{s_2} = GRU_2(\mathbf{h}_{t-1}^{s_2}, \mathbf{s}_t, \mathbf{c}_t) \tag{30}$$

Then we can use $\mathbf{h}_t^{s_2}$ as the input to Equation (24) to conduct the decoding operation.

Besides using attention modeling to address and read the persona information from the the persona memory $\mathbf{M}$, we also employ the idea of Pointer Networks [36] to copy the target words from the memory to form the tips. At the state t, we can obtain the attention weights (distribution) $\mathbf{a}_{t,:}$ on the persona memory $\mathbf{M}_{ui}$. We project $\mathbf{a}_{t,:}$ to a $|V|$-sized vector $\hat{\mathbf{s}}_{t+1}^p$ according to the word indices in V_{ui}. Then we design a soft gate to decide that the word s_{t+1} should be generated or be copied from the memory:

$$p_g = \sigma(\mathbf{v}_p^T(\mathbf{W}_{hp}^s \mathbf{h}_t^{s_2} + \mathbf{W}_{sp}^s \mathbf{s}_t + \mathbf{W}_{cp}^s \mathbf{c}_t + \mathbf{b}_p)) \tag{31}$$

where $\mathbf{v}_p \in \mathbb{R}^{d_h}$ and $p_g \in (0,1)$. We merge the copy signal $\hat{\mathbf{s}}_{t+1}^p$ and the original output $\hat{\mathbf{s}}_{t+1}$ according to the gate p_g:

$$\hat{\mathbf{s}}'_{t+1} = p_g \times \hat{\mathbf{s}}_{t+1} + (1 - p_g) \times \hat{\mathbf{s}}_{t+1}^p \tag{32}$$

Then the tips sampling process can be conducted on $\hat{\mathbf{s}}'_{t+1}$.

2.3.4 ***Tips Quality Discriminator***. Some previous works [40, 41] show that adversarial training strategy is beneficial to the text generation problem. To further improve the performance, we also employ this training strategy in our framework.

The tips discriminator D_{Tips} is a multilayer perceptron with the persona embeddings, the rating information, and the tips sequence as the input. The input tips sequence can be the ground truth S or the tips $\hat{S}$ generated by the system. We propose a Bidirectional-GRU model to conduct the representation learning for S and $\hat{S}$:

$$\mathbf{h}^S = \overrightarrow{\mathbf{h}^S} || \overleftarrow{\mathbf{h}^S} \tag{33}$$

Then we combine all the information according to:

$$\mathbf{h}^q = \tanh(\mathbf{W}_{sh}^q \mathbf{h}^S + \mathbf{W}_{uch}^q \mathbf{z}_\mathbf{u}^\mathbf{c} + \mathbf{W}_{ich}^q \mathbf{z}_\mathbf{i}^\mathbf{c} + \mathbf{W}_{ush}^q \mathbf{z}_\mathbf{u}^\mathbf{s} + \mathbf{W}_{ish}^q \mathbf{z}_\mathbf{i}^\mathbf{s} + \mathbf{W}_{rh}^q \hat{\mathbf{r}} + \mathbf{b}_h^q)$$

Finally, we add a softmax output layer to let the model output a binary category variable:

$$\mathbf{y}^q = \varsigma(\mathbf{W}_{hy}^q \mathbf{h}^q + \mathbf{b}_y^q) \tag{34}$$

We treat the ground truth S as the positive instance and the sampled sequence $\hat{S}$ as the negative instance. So we directly let the first dimension of $\mathbf{y}^q$ represent the positive label. We define the value function as $V(S) = \mathbf{y}_{[0]}^q$. We utilize the REINFORCE [38] method to integrate the tips quality signal $V(S)$ into the tips generation framework to conduct the parameter learning. The details can be found in the existing works [18, 40, 41].

3 EXPERIMENTAL SETUP

3.1 Datasets

In our experiments, we use five datasets from different domains to evaluate our framework. The ratings of all these datasets are integers in the range of $[1, 5]$. There are four datasets from Amazon 5-core[3]: **Electronics**, **Movies & TV**, **Clothing, Shoes and Jewelry**, and **Home and Kitchen**. We regard the field "summary" as tips, and the number of tips texts is the same with the number of reviews. Another dataset is from **Yelp** Challenge[4]. It is also a large-scale dataset consisting of restaurant reviews and tips. We filter out the words with low term frequency in the tips and review texts,

[3]http://jmcauley.ucsd.edu/data/amazon
[4]https://www.yelp.com/dataset_challenge

Table 2: Overview of the datasets.

	Electr	Movies	Home	Clothing	Yelp
users	191,522	123,340	66,212	39,085	115,781
items	62,333	49,823	27,991	22,794	60,224
reviews	1,684,779	1,693,441	550,461	277,521	1,393,257
$\|\mathcal{V}\|$	37,999	82,805	23,950	16,297	82,805

and build a vocabulary $\mathcal{V}$ for each dataset. We show the statistics of our datasets in Table 2.

3.2 Evaluation Metrics

For the evaluation of abstractive tips generation, the ground truth s_h is the tips written by the user. We use *ROUGE* [24] as our evaluation metric with standard options[5]. It is a classical evaluation metric in the field of text summarization [24]. It counts the number of overlapping units between the generated tips and the ground truth written by users. Assuming that s is the generated tips, g_n is n-gram, $C(g_n)$ is the number of n-grams in $\tilde{s}$ (s_h or s), $C_m(g_n)$ is the number of n-grams that appear in both s and s_h, then the ROUGE-N score for s is defined as follows:

$$ROUGE\text{-}N(s) = \sum_{g_n \in s_h} C_m(g_n) / \sum_{g_n \in \tilde{s}} C(g_n) \tag{35}$$

When $\tilde{s} = s_h$, we can get $ROUGE_{recall}$, and when $\tilde{s} = s$, we get $ROUGE_{precision}$. We use Recall, Precision, and F-measure of ROUGE-1 (R-1), ROUGE-2 (R-2), ROUGE-L (R-L), and ROUGE-SU4 (R-SU4) to evaluate the quality of the generated tips.

For the evaluation of rating prediction, we employ two metrics: Mean Absolute Error (*MAE*) and Root Mean Square Error (*RMSE*). Both of them are widely used for rating prediction in recommender systems. Given a predicted rating $\hat{r}_{u,i}$ and a ground-truth rating $r_{u,i}$ from the user u for the item i, the RMSE is calculated as:

$$RMSE = \sqrt{\frac{1}{N} \sum_{u,i} (r_{u,i} - \hat{r}_{u,i})^2} \tag{36}$$

where N indicates the number of ratings between users and items. Similarly, MAE is calculated as follows:

$$MAE = \frac{1}{N} \sum_{u,i} |r_{u,i} - \hat{r}_{u,i}| \tag{37}$$

3.3 Comparative Methods

Abstractive tips generation: We compare our framework PATG with the following baseline and state-of-the-art methods:

- **NRT** [21]: It is a recent multi-task learning framework for rating prediction and abstractive tips generation achieving state-of-the-art performance. Latent factors for users and items are learnt during the training procedure, and are used as the context information for tips generation. NRT does not consider the persona information.
- **LexRank** [7] is a classical method in the field of text summarization. Because we have obtained all the historical tips for the current user and item, then the problem can be regarded as a multi-document summarization problem. LexRank can extract a sentence as the final tips. Note that we give an advantage of this method since the ground truth ratings are used to conduct the filtering.
- **CTR**$_t$: **C**ollaborative **T**opic **R**egression (CTR) [37] is proposed for rating prediction. It contains a topic model component and it can generate topics for items. Then the most topic-similar sentence from the item historical tips is extracted as the tips.
- **HFT**$_t$: **H**idden **F**actors and Hidden **T**opics [26] utilizes a topic modeling technique to model the review texts for rating prediction. Then we can design a tips extraction method HFT$_t$ using the similar technique in CTR$_t$.

Rating prediction: We compare of rating prediction performance with the following baseline methods:

- **NMF**: **N**on-negative **M**atrix **F**actorization [16]. It only uses the rating matrix as the input.
- **PMF**: **P**robabilistic **M**atrix **F**actorization [32]. Gaussian distribution is introduced to model the latent factors for users and items.
- **LRMF**: **L**earning to **R**ank with **M**atrix **F**actorization [34]. It combines a list-wise learning-to-rank algorithm with matrix factorization to improve recommendation.
- **SVD++**: It extends **S**ingular **V**alue **D**ecomposition by considering implicit feedback information for latent factor modeling [14].
- **URP**: **U**ser **R**ating **P**rofile modeling [25]. Topic models are employed to model the user preference from a generative perspective. It still only uses the rating matrix as input.
- The baseline methods used in tips quality evaluation: **NRT** [21], **CTR** [37], **HFT** [26].

Ablation experiments: In order to demonstrate the performance of each component of our framework, we conduct the ablation experiments on the dataset Home. We compare the performance of our integrated model PATG with the models without the some designed components. We set that "A" denotes the aVAE model, "M" represents the persona memory and the Pointer Networks, and "D" represents the tips quality discriminator D_{Tips}. Then the method "**PATG w/o A, M, D**" means that A, M, and D are all removed and we only use the standard VAE for persona modeling.

3.4 Experimental Settings

Each dataset is divided into three subsets: 80%, 10%, and 10%, for training, validation, and testing, respectively. All the parameters of our model are tuned with the validation set. After the tuning process, the number of latent factors k is set to 10 for NMF and SVD++. The number of topics K is set to 50 for the methods using topic models. The number of dimension for the persona embeddings is set to 100. The dimension of the hidden size is 400. In our framework, the number of layers for the rating regression model is 2, and for the tips generation model is 1. We set the beam size $\beta = 5$, and the maximum length $\eta = 20$. All the neural matrix parameters in hidden layers and RNN layers are initialized from a uniform distribution between $[-0.1, 0.1]$. We also regard the word embedding $\mathbf{E}$ used in the tips generation component as a neural parameter. Adadelta [42] is used for gradient based optimization.

[5]ROUGE-1.5.5.pl -n 4 -w 1.2 -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0

4 RESULTS AND DISCUSSIONS

4.1 Research Questions

The research questions in our experiments are as follows:

- **RQ1**: What is the performance of PATG in persona-aware abstractive tips generation? (Section 4.2)
- **RQ2**: Can the persona embeddings improve the performance of rating prediction? (Section 4.3)
- **RQ3**: What is the performance of each component of PATG, such as VAEs, aVAE, and the persona memory? (Section 4.4)
- **RQ4**: Can the model generate tips that are complying with the persona information? (Section 4.5.)
- **RQ5**: Can the model generate tips that are really controlled by ratings? (Section 4.6.)

4.2 Abstractive Tips Generation (RQ1)

The evaluation results of tips generation of our model and the comparative methods are given in Table 3. In order to capture more details, we report Recall, Precision, and F-measure (in percentage) of ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4. Our model achieves the best performance in most of the metrics among all the five datasets. NRT does not consider persona information when generating tips. It only utilizes the learnt latent factors for users and items as the context information. Compared with NRT, our proposed framework PATG obtains better performance on all the metrics, which demonstrates that the consideration of persona information can indeed improve the tips generation performance. We also conduct statistical significance test comparing PATG and NRT and the results indicate that the improvements are significant with $p < 0.05$.

From the results, we also find that our model obtains dramatic improvements on the metric of ROUGE Precision, especially compared with the methods of LexRank, HFT_t, and CTR_t. The main reasons are that those methods are all extraction-based which just extract some original sentences from the original reviews or tips as the final tips. Therefore, the obtained tips are much longer with more noisy and redundant information. In contrast, our framework PATG as well as NRT are abstractive tips generation methods. PATG can generate more concise sentences which not only guarantee the recall metric, but also obtain better precision performance. This also fits the essential spirit of Tips.

4.3 Rating Prediction (RQ2)

Recall that we also design an auxiliary component, i.e. **rating prediction** to capture the sentiment information and control the sentiment of the generated tips, which is also an important aspect of the task of tips generation. Therefore we design some experiments to evaluate the performance of this component. The rating prediction results are given in Table 4. Our model consistently outperforms the best under both MAE and RMSE metrics on all datasets, thus it verifies that the generated persona embeddings are not only effective for generating better tips in the main task, but also useful for predicting accurate ratings in the auxiliary task. Statistical significance of differences between the performance of PATG and the recent method NRT is tested using a two-tailed paired t-test. The result shows that PATG is significantly better than NRT.

4.4 Ablation Analysis (RQ3)

Considering that we design various of components to tackle the corresponding problems of our task, and different components play different roles on our framework. In order to demonstrate the necessity and the performance of each component, we conduct the ablation experiments on the dataset "Home". The results are shown in Table 5. Recall that "A" denotes the aVAE model, "M" represents the persona memory and the Pointer Networks, and "D" represents the tips quality discriminator D_{Tips}. It is obvious that persona modeling based on aVAE (A) can improve the tips generation performance. The persona memory and Pointer Networks (M) are very helpful to the effectiveness of our framework. The tips quality discriminator (D) can also contribute to the better performance. Among all the components, aVAE (A) as well as the persona memory and pointer network (M) contribute more to the improvements of the performance.

4.5 Persona Controlled Generation (RQ4)

The main problem setting of this work is to generate persona-aware tips. In order to demonstrate the quality of the generated tips, we selected some real cases generated by our PATG from different domains for some users and items. The results are listed in Table 6. Although our model generates tips in an abstractive way, tips' linguistic quality is quite good. The **persona properties** of the generated tips match well with the ground truth. For example, in the first case, the generated tips is "This is a great hat for the price.", and the ground truth is "Thanks nice quality excellent price great deal.". Both of the sentences contain the terms "great" and "price". In the third case, the generated tips and the ground truth have a large overlapping with the terms "replace my old", and "processor". Interestingly, sometimes the framework can select some synonyms when conducting tips generation. For instance, the generated tips of the fourth case contains terms "bought" and "for my husband". The ground truth contains "purchased" and "for a male". Moreover, we also choose some generated tips with negative sentiment to conduct the sentiment correlation analysis. Take the generated tips "Please do not buy this coffee maker." as an example (the last case in Table 6), our model predicts a rating of 2.01, which clearly shows a consistent sentiment. The ground truth tips of this example is " They are still overpriced and all but worthless.", which also conveys a negative sentiment. The generated tips "The bottom line of the thin man." and the ground truth "Pretty dark story in book or movie form." are just describing some facts, with a neutral rating 3. Sometimes the overlapping between the generated tips and the ground truth is small, but they still convey similar information.

4.6 Rating Controlled Generation (RQ5)

Recall that in addition to the persona embeddings as context information, rating information is also incorporated to control the sentiment of the generated tips. In order to show such ability of our framework, we design an experiment on the domain "Home" to demonstrate the rating controlled tips generation. Specifically, during the prediction, we manually set the rating from 1 to 5 as the sentiment context to control the generation, and meanwhile, we create a new user and a new item with **0** persona embeddings. This setting mimics a **cold start** case: what the tips will look like

Table 3: ROUGE evaluation on the five datasets from different domains.

Dataset	Method	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
		R	P	F1	R	P	F1	R	P	F1	R	P	F1
Electronics	LexRank	10.97	12.93	11.58	0.95	1.05	0.97	9.96	11.70	10.50	3.08	3.91	3.22
	HFT_t	12.86	12.22	12.35	1.10	1.00	1.03	11.65	11.09	11.19	3.43	3.10	3.14
	CTR_t	12.69	11.72	12.02	1.13	1.05	1.07	11.65	10.74	11.02	3.45	3.06	3.14
	NRT	12.79	17.55	13.85	1.86	2.77	2.08	11.80	15.99	12.70	4.18	6.42	4.45
	PATG	**13.00**	**19.26**	**14.52***	**2.29**	**3.12**	**2.44***	**11.91**	**17.42**	**13.24***	**4.50**	**7.44**	**4.89***
Movies&TV	LexRank	11.10	13.50	11.89	1.06	1.29	1.12	10.02	12.12	10.70	3.25	4.33	3.46
	HFT_t	11.64	10.26	11.33	1.78	1.36	1.46	11.42	8.72	9.67	4.63	3.00	3.28
	CTR_t	11.37	10.33	10.68	1.43	1.31	1.34	10.40	9.44	9.76	3.17	2.73	2.84
	NRT	12.12	20.06	14.17	2.29	3.53	2.55	11.13	18.25	12.98	4.09	8.15	4.79
	PATG	**12.46**	**21.22**	**14.63***	**2.38**	**3.88**	**2.67***	**11.51**	**19.25**	**14.73***	**6.04**	**8.76**	**6.33***
Home	LexRank	12.91	15.47	13.77	1.73	2.06	1.82	11.72	13.97	12.46	3.93	5.02	4.15
	HFT_t	13.32	12.72	12.80	1.33	1.23	1.25	12.25	11.73	11.79	3.63	3.33	3.34
	CTR_t	14.30	13.21	13.55	1.73	1.50	1.58	13.14	12.11	12.43	4.18	3.66	3.78
	NRT	11.51	19.91	13.64	1.95	3.47	2.30	10.64	18.23	12.57	3.77	8.24	4.51
	PATG	12.21	**21.46**	**14.61***	**2.32**	**4.32**	**2.78***	11.32	**19.65**	**13.48***	4.03	**8.71**	**4.82***
Clothing	LexRank	13.31	12.73	12.85	1.06	1.02	1.02	11.97	11.43	11.54	3.47	3.24	3.26
	HFT_t	13.31	12.73	12.85	1.06	1.02	1.02	11.97	11.43	11.54	3.47	3.24	3.26
	CTR_t	13.79	13.82	13.37	1.26	1.23	1.22	12.54	12.14	12.16	3.70	3.52	3.49
	NRT	13.52	18.91	14.75	2.11	2.95	2.31	12.36	17.04	13.39	4.58	7.04	4.86
	PATG	**14.45**	**21.49**	**16.14***	**2.49**	**3.77**	**2.79***	**13.09**	**19.24**	**14.55***	**4.93**	**8.39**	**5.39***
Yelp	LexRank	9.19	12.09	10.28	1.07	1.33	1.15	8.45	11.13	9.45	2.65	3.90	3.01
	HFT_t	10.47	10.21	10.26	0.91	0.87	0.88	9.56	9.31	9.35	2.70	2.57	2.59
	CTR_t	10.68	10.51	10.51	0.98	0.94	0.96	9.70	9.53	9.54	2.77	2.68	2.68
	NRT	10.98	17.42	12.71	1.82	3.03	2.13	9.96	15.76	11.51	3.48	6.48	4.05
	PATG	**12.05**	**19.15**	**14.02***	**2.15**	**3.44**	**2.47***	**10.94**	**17.21**	**12.66***	**3.96**	**7.15**	**4.57***

The "*" marker denotes that PATG achieves better performance than NRT with statistical significance test with $p < 0.05$.

Table 4: MAE and RMSE values for rating prediction.

	Electronics		Movies		Yelp		Clothing		Home	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
LRMF	1.986	2.208	1.891	2.136	1.721	1.982	1.936	2.179	2.028	2.248
PMF	1.139	1.553	0.911	1.307	1.133	1.538	2.355	2.724	1.395	1.780
NMF	0.869	1.266	0.809	1.155	0.961	1.136	0.887	1.257	0.830	1.220
SVD++	0.841	1.226	0.778	1.122	1.957	1.299	0.829	1.169	0.786	1.164
URP	0.875	1.185	0.797	1.101	0.973	1.246	0.876	1.136	0.831	1.135
CTR	0.903	1.154	0.863	1.116	1.051	1.285	0.847	1.094	0.826	1.086
HFT	0.813	1.117	0.769	1.041	0.940	1.191	0.805	1.080	0.773	1.058
NRT	0.823	1.108	0.751	1.038	0.935	1.187	0.828	1.102	0.779	1.058
PATG	**0.747***	**1.016***	**0.740***	**1.015***	**0.866***	**1.134***	**0.714***	**0.987***	**0.694***	**0.997***

* denotes that PATG achieves better performance than NRT [21] with statistical significance test with $\alpha = 0.01$.

without knowing any information of users and items. Then the manual rating and the **0** based persona embeddings are fed into the framework to conduct tips generation. The results are shown in Table 7. Here we set the beam size to 5 so we obtain 5 decoded tips ranked by the likelihood in descending order. It is obvious that when $r > 1$, our framework can generated reasonable tips controlled by ratings. The generated tips show monotone language style, which looks odd, but in fact it is not, because we did not give any persona information of users and items as input (refer to Table 6 for generated tips with rich text such as product information). Moreover, note that this is an artificial scenario which does not compile well with the real case. Table 7 shows that when $r = 1$, the framework does not generate negative tips. We investigate the training dataset and find that the proportion of rating-1 records is much smaller (e.g. a quarter of rating-4 and one-thirteenth of

Table 5: Ablation experiments on the dataset Home. R-* represents the F1-Measure of ROUGE-*.

System	R-1	R-2	R-L	R-SU4
PATG w/o A, M, D	13.76	2.27	12.64	4.45
PATG w/o M, D	13.99	2.61	12.95	4.71
PATG w/o D	14.32	2.72	13.30	4.81
PATG	14.51	2.72	13.48	4.81

Table 6: Examples of the predicted ratings and the generated tips for some users and items. The first line of each group shows the generated rating and tips. The second line shows the ground truth.

Rating	Tips
5.10	***This is a great hat for the price.***
5	Thanks nice quality excellent price great deal.
5.08	***This is a great pitcher.***
5	Beautiful pitcher makes a great vase.
5.17	***I bought this food processor to replace my old one.***
4	I got this about a month ago to replace my old food processor.
4.99	***These shoes are so comfortable and I bought these for my husband.***
5	Comfortable good looking shoes purchased for a male that walks a lot.
4.81	***This is a great movie.***
5	Amazing love great movie and all teen shold see it.
2.57	***The bottom line of the thin man.***
3	Pretty dark story in book or movie form.
2.01	***Please do not buy this coffee maker.***
1	They are still overpriced and all but worthless.

rating-5), which may cause our model under-fitting for generating rating-1 tips.

5 RELATED WORK

Abstractive text generation is a challenging task. Recently, sequence modeling based on the gated recurrent neural networks such as Long Short-Term Memory (LSTM) [10] and Gated Recurrent Unit (GRU) [5] demonstrates high capability in text generation related tasks, such as abstractive summarization [19, 28, 31], dialogue systems [3, 33] and image caption generation [39].

In the area of recommendation systems, some researchers also apply LSTM or GRU based RNN models on abstractive text generation. Tang et al. [35] propose a framework to generate context-aware reviews. Sentiments and products are encoded into a continues semantic representation and use RNN to conduct the decoding and generation. Dong et al. [6] regard users, products, and rating as attribute information and employ a attention modeling based sequence modeling framework to generate reviews. Ni et al. [29] propose to combine collaborative filtering with generative networks to jointly perform the tasks of item recommendation and review generation. Low-dimensional user preferences and item properties are combined with a character-level LSTM model to conduct the review generation. Yao et al. [40] employ the adversarial strategy to make the generated review indistinguishable from human written

Table 7: Rating controlled tips generation in a cold start scenario. $\hat{r}$ is the rating usd to control the sentiment.

$\hat{r}$	Tips
5	This is a great product. I bought this for my mom and she loves it. I bought this for my daughter and she loves it. I bought this for my husband and she loves it. I bought this for my daughter.
4	This is a good product. I bought this for my daughter for christmas. I bought this for my daughter and she loved it. I bought this for my mom and she loved it. I bought this for my daughter for christmas and she loves it.
3	Not as good as my old one. This is a good product. Not as good as my old one. I bought this for my daughter for christmas. I bought this for my mom and she loved it.
2	Not as good as I expected. Not as good as the original. Not as good as my old one. I bought this for my daughter for christmas. This is a good product.
1	This is a good product. I bought this for my daughter for christmas. This is the third one i bought. This is the third one i had. I bought this for my daughter for christmas and she loves it.

ones so that can improve the performance of review generation. Although research works have been proposed for review generation, there are very few works investigating tips generation. Li et al. [21] propose a unified framework to jointly conduct rating prediction and abstractive tips generation. Latent factors for users and items are learnt from the multi-task learning framework and are fed into the tips generation framework as context information.

However, few works consider persona modeling and sentiment detection jointly in their frameworks. Hu et al. [11], Liao et al. [23] revised the variational auto-encoders (VAEs) based text generation model and can control the sentiment and tense of the generated reviews. But they still do not consider persona information in their model. Li et al. [17] propose two methods to conduct the persona modeling for text generation in the area of dialog systems, but dialog systems have different characteristics with recommendation system. Moreover, they do not consider the sentiment information. Different with these previous works, our proposed framework can jointly consider the persona information and the sentiment signal when conducting the abstractive tips generation.

6 CONCLUSIONS

We propose a framework PATG to address the problem of persona-aware tips generation. A framework based on adversarial variational auto-encoders (aVAE) is exploited for persona modeling from the historical tips and reviews. We also design an external persona memory for directly storing the persona related words for the current user and item. The distilled persona embeddings are used as latent factors and are fed into the rating prediction component for detecting sentiment. Then the persona embeddings and the

sentiment information are incorporated into a recurrent neural networks (RNN) based tips generation component to control the tips generation. Experimental results show that our framework achieves better performance than the state-of-the-art models on abstractive tips generation.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

[2] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *JMLR* 3, Jan (2003), 993–1022.

[3] Deng Cai, Yan Wang, Victoria Bi, Zhaopeng Tu, Xiaojiang Liu, Wai Lam, and Shuming Shi. 2018. Skeleton-to-Response: Dialogue Generation Guided by Retrieval Memory. *arXiv preprint arXiv:1809.05296* (2018).

[4] Dallas Card, Chenhao Tan, and Noah A Smith. 2017. A Neural Framework for Generalized Topic Models. *arXiv preprint arXiv:1705.09296* (2017).

[5] Kyunghyun Cho, Bart van Merriënboer Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. *EMNLP* (2014), 1724–1734.

[6] Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. 2017. Learning to generate product reviews from attributes. In *EACL*, Vol. 1. 623–632.

[7] Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *JAIR* 22 (2004), 457–479.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*. 2672–2680.

[9] Anirudh Goyal Alias Parth Goyal, Alessandro Sordoni, Marc-Alexandre Côté, Nan Ke, and Yoshua Bengio. 2017. Z-Forcing: Training Stochastic Recurrent Networks. In *NIPS*. 6716–6726.

[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[11] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *ICML*. 1587–1596.

[12] Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *ICLR*.

[13] Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Conference of the Association for Machine Translation in the Americas*. Springer, 115–124.

[14] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*. ACM, 426–434.

[15] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. 2016. Autoencoding beyond pixels using a learned similarity metric. In *ICML*. 1558–1566.

[16] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *NIPS*. 556–562.

[17] Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A Persona-Based Neural Conversation Model. In *ACL*, Vol. 1. 994–1003.

[18] Piji Li, Lidong Bing, and Wai Lam. 2018. Actor-critic based training framework for abstractive summarization. *arXiv preprint arXiv:1803.11070* (2018).

[19] Piji Li, Wai Lam, Lidong Bing, and Zihao Wang. 2017. Deep Recurrent Generative Decoder for Abstractive Text Summarization. In *EMNLP*. 2091–2100.

[20] Piji Li, Zihao Wang, Wai Lam, Zhaochun Ren, and Lidong Bing. 2017. Salience Estimation via Variational Auto-Encoders for Multi-Document Summarization. In *AAAI*. 3497–3503.

[21] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. Neural Rating Regression with Abstractive Tips Generation for Recommendation. In *SIGIR*. ACM, 345–354.

[22] Xiaopeng Li and James She. 2017. Collaborative variational autoencoder for recommender systems. In *KDD*. ACM, 305–314.

[23] Yi Liao, Lidong Bing, Piji Li, Shuming Shi, Wai Lam, and Tong Zhang. 2018. QuaSE: Sequence Editing under Quantifiable Guidance. In *EMNLP*. 3855–3864.

[24] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out-ACL Workshop*. 74–81.

[25] Benjamin M Marlin. 2003. Modeling user rating profiles for collaborative filtering. In *NIPS*. 627–634.

[26] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*. ACM, 165–172.

[27] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. 2017. Adversarial Variational Bayes: Unifying Variational Autoencoders and Generative Adversarial Networks. In *ICML*. 2391–2400.

[28] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 280–290.

[29] Jianmo Ni, Zachary C Lipton, Sharad Vikram, and Julian McAuley. 2017. Estimating Reactions and Recommending Products with Generative Models of Reviews. In *IJCNLP*, Vol. 1. 783–791.

[30] Zhaochun Ren, Shangsong Liang, Piji Li, Shuaiqiang Wang, and Maarten de Rijke. 2017. Social collaborative viewpoint regression with explainable recommendations. In *WSDM*. ACM, 485–494.

[31] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *EMNLP*. 379–389.

[32] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization.. In *NIPS*. 1–8.

[33] Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. In *ACL*, Vol. 1. 1577–1586.

[34] Yue Shi, Martha Larson, and Alan Hanjalic. 2010. List-wise learning to rank with matrix factorization for collaborative filtering. In *RecSys*. 269–272.

[35] Jian Tang, Yifan Yang, Sam Carton, Ming Zhang, and Qiaozhu Mei. 2016. Context-aware Natural Language Generation with Recurrent Neural Networks. *arXiv preprint arXiv:1611.09900* (2016).

[36] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *NIPS*. 2692–2700.

[37] Chong Wang and David M Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *KDD*. ACM, 448–456.

[38] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8, 3-4 (1992), 229–256.

[39] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*. 2048–2057.

[40] Yuanshun Yao, Bimal Viswanath, Jenna Cryan, Haitao Zheng, and Ben Y Zhao. 2017. Automated Crowdturfing Attacks and Defenses in Online Review Systems. In *CCS*. ACM, 1143–1158.

[41] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient.. In *AAAI*. 2852–2858.

[42] Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* (2012).

[43] Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. In *ACL*. 654–664.

Learning Travel Time Distributions with Deep Generative Model

Xiucheng Li
Nanyang Technological University
xli055@e.ntu.edu.sg

Gao Cong
Nanyang Technological University
gaocong@ntu.edu.sg

Aixin Sun
Nanyang Technological University
axsun@ntu.edu.sg

Yun Cheng
ETH Zürich
chengyu@ethz.ch

ABSTRACT

Travel time estimation of a given route with respect to real-time traffic condition is extremely useful for many applications like route planning. We argue that it is even more useful to estimate the *travel time distribution*, from which we can derive the expected travel time as well as the uncertainty. In this paper, we develop a deep generative model – DeepGTT – to learn the travel time distribution for any route by conditioning on the real-time traffic. DeepGTT interprets the generation of travel time using a three-layer hierarchical probabilistic model. In the first layer, we present two techniques, amortization and spatial smoothness embeddings, to share statistical strength among different road segments; a convolutional neural net based representation learning component is also proposed to capture the dynamically changing real-time traffic condition. In the middle layer, a nonlinear factorization model is developed to generate auxiliary random variable *i.e.,* speed. The introduction of this middle layer separates the statical spatial features from the dynamically changing real-time traffic conditions, allowing us to incorporate the heterogeneous influencing factors into a single model. In the last layer, an attention mechanism based function is proposed to collectively generate the observed travel time. DeepGTT describes the generation process in a reasonable manner, and thus it not only produces more accurate results but also is more efficient. On a real-world large-scale data set, we show that DeepGTT produces substantially better results than state-of-the-art alternatives in two tasks: travel time estimation and route recovery from sparse trajectory data.

CCS CONCEPTS

• **Applied computing** → **Forecasting**; *Mathematics and statistics.*

KEYWORDS

Travel time distribution learning, Deep generative models, VAEs

ACM Reference Format:
Xiucheng Li, Gao Cong, Aixin Sun, and Yun Cheng. 2019. Learning Travel Time Distributions with Deep Generative Model. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313418

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313418

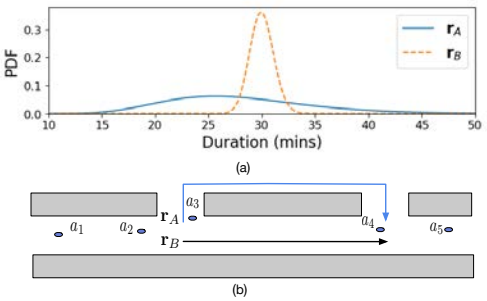


Figure 1: (a) The travel time distributions of two routes–$\mathbf{r}_A$ and $\mathbf{r}_B$, $\mathbb{E}[t_A] = 26$, $\mathbb{E}[t_B] = 30$. (b) Example of route recovery from sparse trajectory.

1 INTRODUCTION

Estimating travel time of a given route on a road network is an essential task for many applications, including route planning, taxi dispatching, and ride-sharing. Many online web apps (*e.g.,* Google Map and Uber) provide such service. Due to its great importance, significant research efforts have been made towards the accurate estimation of travel times [20, 38, 39, 41]. However, existing studies only focus on the estimation of *expected travel time*. We argue that it is much more helpful or even essential to estimate *travel time distribution* (the probability density function) of a given route, based on real-time traffic conditions. We illustrate this point through an example.

Motivating example. Mary has a time budget of 35 minutes to reach airport from home. She queries for the best route, and there are two candidates — route **A** and route **B**, denoted by $\mathbf{r}_A$ and $\mathbf{r}_B$, respectively. The travel time distributions of $\mathbf{r}_A$ and $\mathbf{r}_B$ are depicted in Figure 1a, with the expectation $\mathbb{E}[t_A] = 26$ (minutes), and $\mathbb{E}[t_B] = 30$ (minutes). If the system recommends route solely based on the expected travel time, then $\mathbf{r}_A$ is the best choice. However, as shown in the figure, $\mathbf{r}_A$ has a much larger variance. Therefore, Mary would have a high risk of missing her flight if taking $\mathbf{r}_A$. In contrast, the probability density function of t_B is much more concentrated, and it is much safer to take $\mathbf{r}_B$.

This example shows that it is more sensible to make decision based on the probability distribution of the travel time, instead of mere its expectation. In fact, estimating travel time distribution also addresses the issue of route recovery from the sparse trajectory [42]. Figure 1b shows two possible routes $\mathbf{r}_A, \mathbf{r}_B$ from a_3 to a_4 (due to the low sampling rate) in an observed trajectory $T_a = [a_1, \ldots, a_5]$.

As the sampling points usually carry timestamps, we could get the travel time cost from a_3 to a_4, denoted by t. The problem of inferring the most likely route can be expressed as

$$\underset{\mathbf{r}}{\text{argmax}}\ P(\mathbf{r}|t), \quad \mathbf{r} \in \{\mathbf{r}_A, \mathbf{r}_B\}.$$

Using the Bayes Rule, we have $P(\mathbf{r}|t) \propto P(t|\mathbf{r})P(\mathbf{r})$. That is, we need to access the value of travel time distribution $P(t|\mathbf{r})$ at any $t > 0$.

In this paper, we aim to learn the ***travel time distribution*** $P(t|\mathbf{r})$ for a given route $\mathbf{r}$, with the consideration of real-time traffic conditions. This problem is challenging due to a few reasons. First, data sparsity remains a key challenge. Even though we have access to a large number of trajectories, the trajectories demonstrate a skewed distribution in the space. Roads in central business district are frequently visited, while roads in rural or remote areas are rarely covered. It is reported that about 10% of paths cannot be estimated due to lack of neighbor trajectories [39]. Second, travel time heavily depends on real-time traffic conditions which are dynamic. Almost all existing methods [20, 25, 38, 39, 42] assume that traffic conditions in the same time slot (*e.g.,* 8:00am-9:00am on every Saturday) are temporally-invariant, when estimating travel times. The assumption does not hold because traffic conditions can be influenced by many random variables such as the daily activities of people, road works, and accidents. Considering real-time traffic conditions makes data sparsity issue even worse. Further, travel time is also influenced by spatial characteristics of a road *e.g.,* number of traffic lights, number of lanes, and speed limit. To the best of our knowledge, no existing study has incorporated these heterogeneous influencing factors (both spatial and temporal) into a single model for travel time estimation.

To address the aforementioned challenges simultaneously, we approach travel time distribution learning from deep generative perspective [23, 35]. Specifically, we propose a model named DeepGTT (Deep Generative Travel Time), a three-layer hierarchical probabilistic model [14]. Each layer in DeepGTT captures underlying dependencies among the relevant random variables in a principled way. In the first layer, we present two techniques, spatial smoothness embeddings and amortization [11], to share statistical strength among different road segments to address the data sparsity challenge. A representation ρ_i is learned for each road segment r_i. We further propose a convolutional neural network based representation learning component to address the second challenge. This learning component enables the encoding of dynamically changing real-time traffic conditions into a vector $\mathbf{c}$. In the middle layer, a nonlinear factorization model is developed to combine the road segment representation ρ_i and traffic condition $\mathbf{c}$ to generate the travel speed v_i for road segment r_i. This auxiliary random variable v_i allows us to separate the statical spatial road features from the dynamically changing traffic conditions, so as to address the third challenge. In the end layer, an attention mechanism [4] based function is proposed to generate the entire route travel time t by adaptively aggregating the road segment speed v_i. We derive a variational low bound [7] to train the entire model in an end-to-end fashion.

To the best of our knowledge, this is the first deep generative model developed for travel time distribution learning. A great advantage DeepGTT inherits from probabilistic methods [14], is that it is quite data-efficient [6, 12]; even only trained with a small fraction of data it could produce substantially better results than existing methods trained on much larger datasets. In summary, our contributions are as follows:

- For the first time, we develop a deep generative model, named DeepGTT, to learn travel time distributions. DeepGTT is designed to be data-efficient.The model utilizes spatial smoothness embeddings and amortization to model road segments, and a convolutional neural network based representation learning component to capture real-time traffic conditions.
- The carefully designed hierarchical architecture allows us to separate the dynamically changing traffic conditions from the static spatial features, and thus enables incorporating the heterogeneous influencing factors (both spatial and temporal) into a single model for travel time learning.
- We develop an attention mechanism based function to collectively aggregate road segment speeds to generate the observations. Then, a variational low bound is derived to enable the model to be trained in an end-to-end fashion. As DeepGTT optimizes the complete distribution rather than a single mean value, it could incorporate more variability for more accurate prediction.
- We conduct thorough experiments on a real world large traffic dataset. Experiments show that our model produces substantially better results than state-of-the-art methods in both travel time estimation and route recovery tasks.

The rest of the paper is organized as follows. The related work is discussed in Section 2. The definitions, problem statement and the overview of DeepGTT are presented in Section 3. Section 4 describes the details of our method. The experimental results are presented in Section 5. We conclude the paper and discuss the future work in Section 6.

2 RELATED WORK

We briefly review the related studies on *travel time estimation* and *deep generative models* in this section.

2.1 Travel time estimation

The road segment-based methods first estimate travel time on each individual road segment independently by using the speeds information collected by loop detectors. Then travel time on a given route is estimated by summing up the estimated times of the road segments traversed in the given route. Such methods fail to consider delays caused by traffic lights, and left/right turns, resulting in large estimation errors.

To address the weaknesses of road segment-based methods, route-based methods attempt to estimate the travel time of a route as a whole. There are two main threads for route-based methods, namely, nearest neighbors search [36, 39] and trajectory regression [20, 41, 48]. Nearest neighbors search estimates travel time of a route by averaging travel times of the historical trajectories that have the closed origin and destination with the query route. Trajectory regression methods, on the other hand, treat travel times of the road segments as unknown variables and solve a regression problem using the historical trajectories.

Several works also attempt to estimate the travel time distribution for a given route on road network. Hunter et al. [19] investigate travel time distribution of a path in arterial networks using Gaussian Markov Random Field. Wu et al. [42] propose a model named STRS, that seeks to recover the route from sparse trajectories. STRS comprises of a travel time distribution estimation component and a spatial transition inference component. Its travel time distribution estimation component first learns the mean value of travel time by trajectory regression, and then empirically study the relationship between the variance and mean value to derive the distribution. STRS has been shown to be the state-of-the-art route recovery algorithm. Raghu et al. [13] explore the prediction limits of different methods by estimating the entropy of the travel time distributions.

Almost all existing studies make the assumption that traffic conditions in the same time slot are temporally-invariant. In this paper, we relax this assumption by learning travel time distribution conditioned on the real-time traffic. Wang et al. [40] also attempt to estimate the travel time of a path based on the real-time traffic. They split the query path into multiple sub-paths and search them from the real-time trajectories. However this method seriously suffers from data sparsity issue, even tensor decomposition was used to mitigate it.

Very recently, two deep learning based travel time estimation models–DeepTTE [38] and WDR [41] are proposed. Our method differs from them in three aspects. First, both DeepTTE and WDR only learn to predict the expectation of travel time, whereas our model outputs a distribution. Second, DeepTTE and WDR both adopt the assumption that traffic conditions in a time slot are temporally-invariant (except that DeepTTE takes weather into consideration), while we learn the travel time distribution conditioned on the real-time traffic. Third, both of them employ an RNN to learn the travel time by brute force, which unavoidably leads to a data-hungry model. In contrast, our method is built on deep generative models which allow us to interpret the generation of travel time in a reasonable manner. As a result, our model not only produces more accurate results but also is data-efficient.

Li et al. [27] study the problem of estimating travel time of a *trip* defined by its origin and destination. We highlight that this problem actually is different from ours as we learn the travel time distribution for an entire *route*. Note that, a *trip* can be achieved through multiple routes. In many applications such as trip planning, ride-sharing, travel time estimation between two locations does not help much without knowing the routes to travel.

2.2 Deep Generative Models

Deep generative models or deep probabilistic methods are the hybrids of deep learning and Bayesian inference, which share the great advantages of two worlds. On the one hand, generative models [14] allow us to specify the relationships among random variables to incorporate our prior knowledge in a flexible way. On the other hand, the automatic-differentiation [5] permits us to perform inference on large-scale datasets.

Generative models have been widely adopted in text data mining [8], online recommendation [15, 16, 46], spatial data analytics [18, 45, 47] as well as air quality inference [9, 10]. They are usually trained with sampling-based methods [3, 32] which have the drawback of slow convergence, thus limits their usage to relative small datasets [17]. The emergence of VAE (Variational Auto-Encoders) [23, 35] makes training large-scale generative models possible by marrying Bayesian inference and the automatic-differentiation technique—the core ingredient in deep learning. Since then, deep generative models have been growing rapidly and achieved a range of state-of-the-art results in semi-supervised learning [22], multiple instances scene understanding [12], high quality images and audio synthesis [37, 43], large-scale online recommendation [28]. In contrast, deep generative models have been less explored in spatial-temporal data analysis field. To our knowledge, this is the first deep generative model developed for travel time distribution learning.

Table 1: Notation.

Symbol	Definition
T	Trajectory
$\mathbf{T}$	Trip
$\mathbf{r}$	Route
r_i	The i-th road segment in route $\mathbf{r}$
$\rho_i \in \mathbb{R}^{\lvert\rho_i\rvert}$	The learned representation of r_i
$v_i \in \mathbb{R}$	Travel speed of r_i
C	Real-time traffic indicator
$\mathbf{c} \in \mathbb{R}^{\lvert\mathbf{c}\rvert}$	The learned real-time traffic representation

3 PROBLEM DEFINITION AND DEEPGTT OVERVIEW

We start with definitions of *Road Network*, *Route*, *GPS Trajectory*, and *Trip*. We then formally define our research problem. For ease of reference, the notation used through the paper is given in Table 1.

Definition 3.1. **Road Network.** A road network is represented as a directed graph $G(V, E)$, in which V, E represents the vertices (crossroads) and edges (road segments) respectively.

Definition 3.2. **Route.** A route $\mathbf{r} = [r_i]_{i=1}^{n}$ is a sequence of adjacent road segments, where $r_i \in E$ represents the i-th road segment in the route.

Definition 3.3. **GPS Trajectory.** A GPS trajectory T is a sequence of sample points $\langle p_i, \tau_i \rangle_{i=1}^{|T|}$ from the underlying route of a moving object, where p_i, τ_i represents the i-th GPS location and timestamp respectively.

Definition 3.4. **Trip.** A trip $\mathbf{T}$ is a travel along a route $\mathbf{r}$ in the road network starting at time s. We use $\mathbf{T}.\mathbf{r}$ and $\mathbf{T}.s$ respectively, to denote the traveled route and starting time of trip $\mathbf{T}$.

In our study, all GPS trajectories are first mapped into the road network to get their underlying routes with a map matching algorithm [30].

The travel time distribution of a route $\mathbf{T}.\mathbf{r}$ highly depends on the real-time traffic condition and it is difficult to precisely define traffic condition. Intuitively, average speed of (sub-)trajectories could be considered as a measurement or sensing of the real-time traffic condition. Therefore, we propose to use all (sub-)trajectories

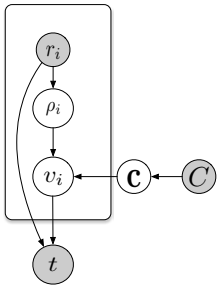


Figure 2: The graphical model of the generative process in which $\rho_i \in \mathbb{R}^{|\rho_i|}$, $v_i \in \mathbb{R}$, $\mathbf{c} \in \mathbb{R}^{|\mathbf{c}|}$ are latent variables and r_i, t are observed scalars.

collected during the time window $[\mathbf{T}.s - \Delta, \mathbf{T}.s)$ as the indicator of real-time traffic condition, denoted by C (or $\mathbf{T}.C$ for the route $\mathbf{T}.\mathbf{r}$). Here, Δ is a specified parameter *e.g.,* 30 minutes. We are now ready to formally define the research problem in this paper

Problem Statement. Given a road network $G(V, E)$ and a historical trajectory dataset $\mathcal{D} = \left\{T^{(m)}\right\}_{m=1}^{M}$, we aim to learn the travel time distribution $P(t|\mathbf{r}, C)$, for a given route $\mathbf{r}$ in the road network, conditioned on the real-time traffic condition indicator C.

DeepGTT Overview. The high-level idea of our proposed solution DeepGTT is that we treat the travel time of a route $\mathbf{r}$ as a random variable t and explain the generation of t using a three-layer hierarchical probabilistic model. The graphical model of the generative process is depicted in Figure 2.

Intuitively, the travel speed of a road segment r_i depends on two important factors: 1) the statical spatial features such as the road types (highway, secondary way), the number of lanes, etc., and 2) the dynamic temporal feature, *i.e.,* the real-time traffic condition. In our model, we learn a road representation ρ_i for road segment r_i to capture its statical spatial features and use $\mathbf{c}$ to model the real-time traffic condition. The generative process of travel time t for route $\mathbf{r} = [r_i]_{i=1}^{n}$ is as follows:

- Draw the traffic condition representation $\mathbf{c} \sim P(\mathbf{c}|C)$.
- For the i-th road segment r_i in the route $\mathbf{r}$,
  - Draw the road segment representation $\rho_i \sim P(\rho_i|r_i)$.
  - Draw the road segment speed $v_i \sim P(v_i|\rho_i, \mathbf{c})$.
- Draw the travel time $t \sim P(t|\mathbf{r}, \mathbf{v})$.

In this model, $\mathbf{v} = [v_i]_{i=1}^{n}$ and $\mathbf{c} \in \mathbb{R}^{|\mathbf{c}|}$, $\rho_i \in \mathbb{R}^{|\rho_i|}$ are fixed-length vectors.

Specifically, in the generative process, we first draw the real-time traffic representation $\mathbf{c} \sim P(\mathbf{c}|C)$. For every road segment r_i in route $\mathbf{r}$ we draw its representation $\rho_i \sim P(\rho_i|r_i)$ which only depends on the statical spatial features of r_i. The introduction of the auxiliary random variable v_i in the middle layer allows us to model both r_i's statical spatial features and the dynamically changing traffic conditions. In other words, probability distribution $P(v_i|\rho_i, \mathbf{c})$ depends on both road representation ρ_i and traffic condition $\mathbf{c}$. In the end, we collectively aggregate the road segment speeds v_i to generate the travel time for the entire route $\mathbf{r}$ by drawing t from $P(t|\mathbf{r}, \mathbf{v})$.

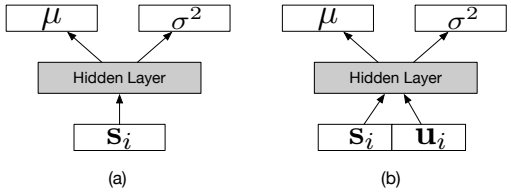


Figure 3: The parameterization of $P(\rho_i|r_i)$.

4 DEEPGTT MODELING

We next detail the modeling of road segment representation ρ_i (Section 4.1), traffic condition $\mathbf{c}$ (Section 4.2), speed (Section 4.3), and the aggregation (Section 4.4). Section 4.5 details the loss function and the learning algorithm. The prediction is discussed in Section 4.6.

4.1 Road Segment Representation ρ_i

As aforementioned, travel speed v_i is dominated by two main factors: the statical spatial features of road segment r_i and the dynamic temporal feature – real-time traffic condition. We model the impact of statical spatial features of r_i with distribution $P(\rho_i|r_i)$, where $\rho_i \in \mathbb{R}^{|\rho_i|}$ is the static road segment representation.

The high-level idea is to encode into ρ_i all spatial features that are indicative to travel speed via modeling $P(\rho_i|r_i)$ conditioned on such spatial features. In our model, we consider the following 5 types of statical spatial features:

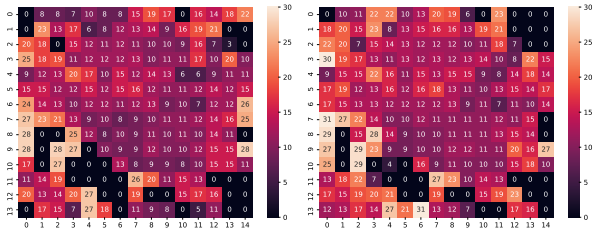
(1) Road types: *e.g.,* primary, secondary, tertiary, residential, etc.;
(2) Number of lanes: how many marked traffic lanes;
(3) Whether it is a one way or not;
(4) The road shape, *e.g.,* straight, curve;
(5) Spatial smoothness, *e.g.,* congestion propagation makes neighboring roads likely to have similar congestion level.

To overcome data sparsity issue, we present two techniques– amortization and spatial smoothness embeddings. These two techniques enable the sharing of statistical strength among different road segments and impose spatial smoothness on modeling $P(\rho_i|r_i)$.

Amortization. The main idea of amortization [11] is to use a function $f(\cdot)$ to map a given observation to a set of parameters that are shared across all data-points. This allows us to share statistical strength among different road segments, in the sense that the knowledge learned from frequently traveled road segments is shared to those rarely traveled. In our setting, the observation corresponds to the (1)-(3) types of categorical features.

For easy exposition, let us use $n^{(\mathrm{t})}$, $n^{(\mathrm{l})}$, $n^{(\mathrm{o})}$ to represent the number of possible values in the first 3 types of categorical features, respectively. Then we use $\phi^{(\mathrm{t})}$, $\phi^{(\mathrm{l})}$, $\phi^{(\mathrm{o})}$ to map a road segment to its corresponding feature value. For example, assuming there are 5 road types in total and r_1 belongs to the third road type, then we have $n^{(\mathrm{t})} = 5$ and $\phi^{(\mathrm{t})}(r_1) = 3$.

We introduce three spatial feature embedding matrices, $S^{(\mathrm{t})}$, $S^{(\mathrm{l})}$, and $S^{(\mathrm{o})}$, with size $n^{(\mathrm{t})} \times d^{(\mathrm{t})}$, $n^{(\mathrm{l})} \times d^{(\mathrm{l})}$, and $n^{(\mathrm{o})} \times d^{(\mathrm{o})}$ respectively, for the three types of spatial categorical features. Here, $d^{(\mathrm{t})}$, $d^{(\mathrm{l})}$, and $d^{(\mathrm{o})}$ are their embedding dimensions. Each row in $S^{(\cdot)}$ corresponds to an attribute embedding, *e.g.,* $S^{(\mathrm{t})}[1]$ (the first row of $S^{(\mathrm{t})}$) is the embedding of the first type of road and $S^{(\mathrm{t})}[\phi^{(\mathrm{t})}(r_i)]$ is the



(a) Real-time traffic condition C at 7am (b) Real-time traffic condition C at 11pm

Figure 4: The real-time traffic condition C at 7am and 11pm, where the cell value indicates the average speed in that local area.

embedding of r_i's road type. For road segment r_i we define $\mathbf{s}_i$ as the concatenation of its spatial feature embeddings, *i.e.,*

$$\mathbf{s}_i = \left[S^{(\mathrm{t})}[\phi^{(\mathrm{t})}(r_i)], S^{(\mathrm{l})}[\phi^{(\mathrm{l})}(r_i)], S^{(\mathrm{o})}[\phi^{(\mathrm{o})}(r_i)]\right] \in \mathbb{R}^{d^{(\mathrm{t})}+d^{(\mathrm{l})}+d^{(\mathrm{o})}}$$

In the spirit of amortization, we model the distribution $P(\rho_i|r_i)$ with a high dimensional Gaussian distribution whose parameters are the outputs of a determined function taking $\mathbf{s}_i$ as input

$$P(\rho_i|r_i) = \mathcal{N}\left(\mu(\mathbf{s}_i), \mathrm{diag}\left(\sigma^2(\mathbf{s}_i)\right)\right) \tag{1}$$

Here, $\mu(\mathbf{s}_i)$ and $\sigma^2(\mathbf{s}_i)$ are parameterized by two MLPs (Multi-Layer Perceptron) with shared input and hidden layers as shown in Figure 3a.

Spatial Smoothness Embeddings. Amortization allows us to share the statistical strength of the categorical features among different road segments. However, it does not consider the impact of road shape and spatial smoothness *i.e.,* the last 2 types of road features. To remedy this, for every r_i we add an additional embedding $\mathbf{u}_i \in \mathbb{R}^{|\mathbf{u}_i|}$. This embedding aims to capture the complex factors (*e.g.,* road shape) that are specific to the road segment. Moreover, to impose spatial smoothness, we run DeepWalk [31] on the road network $G(V, E)$ to initialize the embeddings $\mathbf{u}_i$. Then we concatenate $\mathbf{s}_i$ and $\mathbf{u}_i$ as the input of the MLPs to parameterize the probability distribution $P(\rho_i|r_i)$ as following (also shown in Figure 3b).

$$P(\rho_i|r_i) = \mathcal{N}\left(\mu(\mathbf{s}_i, \mathbf{u}_i), \mathrm{diag}\left(\sigma^2(\mathbf{s}_i, \mathbf{u}_i)\right)\right) \tag{2}$$

The embeddings $S^{(\mathrm{t})}$, $S^{(\mathrm{l})}$, $S^{(\mathrm{o})}$, $\mathbf{u}_i$ and parameters of the two MLPs will be learned from data.

4.2 Real-time Traffic Representation c

In the generative process, we assume there exists a distribution $P(\mathbf{c}|C)$ from which we could draw a random variable $\mathbf{c}$ that captures the real time traffic condition. To realize this, we need to first find a way to construct C which could roughly reveal the real time traffic condition. To this end, we partition the space into cells and estimate the average vehicle speed in a cell by using real time trajectories. Each cell is considered as a sensing of the local traffic congestion level in that area. Figure 4(a) and 4(b) show the constructed C at 7am and 11pm respectively on the same example day, and the cell size is 2km × 2km.

To model $P(\mathbf{c}|C)$, a straightforward approach is to reshape C into a vector to get $\mathbf{c}$ and then build the condition probability table with each entry corresponding to a possible $\mathbf{c}$ value. However, such a table representation has several drawbacks. First, the space cost is prohibitively expensive. Even if we restrict the average speed into an integer with k possible values, there will be $k^{|\mathbf{c}|}$ states for $\mathbf{c}$ in total. Second, the table representation is sensitive to the spatial distribution of vehicles at that time. If there is no sensing vehicle passing the cell at that time, the cell value becomes zero, *e.g.,* $C[0, 13]$, $C[0, 14]$ are 18, 22 at 7am and they become zero at 11pm as shown in Figure 4. This is unreasonable because 7am is typically a peak hour in a day. Third, the table representation could not reveal similarity between two similar states. Even it only one cell value changes in C, the resulting state $\mathbf{c}$ will be considered to be different from the original one.

To overcome the above shortcomings, we propose to use a CNN (Convolutional Neural Net) to extract the abstract features from the rough estimation speed matrix C. The intuition is that similar traffic conditions generate similar travel times for a given path, which in turn results in close loss. We can then propagate such signals through gradients back to the CNN whose parameters will be tuned to being robust to missing values and being capable of producing similar representations for analogous traffic conditions.

$$\begin{aligned} \mathbf{f} &= \mathrm{CNN}(C) \\ P(\mathbf{c}|C) &= \mathcal{N}\left(\mu(\mathbf{f}), \mathrm{diag}\left(\sigma^2(\mathbf{f})\right)\right) \end{aligned} \tag{3}$$

Equation 3 presents the modeling of $P(\mathbf{c}|C)$, where $\mu(\mathbf{f})$, $\sigma^2(\mathbf{f})$ are parameterized by two MLPs with shared hidden layer. The convolutional neural net comprises of three connected convolution blocks and an average pooling layer. Each convolution block consists of three layers: Conv2d → BatchNorm2d → LeakyReLU.

Note that, the choice of making $\mathbf{c}$ being a random variable instead of being a deterministic one allows us to incorporate more variability in modeling complex traffic conditions and to produce a more reliable model.

4.3 Road Segment Speed v_i

Once we get the static road speed representation ρ_i and the dynamic traffic representation $\mathbf{c}$ we could use them to generate road travel speed v_i for r_i. One straightforward method is the linear factorization models [29], *i.e.,* travel speed v_i is interpreted as the linear interaction between ρ_i and $\mathbf{c}$ as

$$v_i = \langle \rho_i, \mathbf{c} \rangle \tag{4}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product operation. However, this simple solution is limited by the linear assumption. We consider that travel speed on a road segment is very likely to be the result of nonlinear interaction between the static spatial features and the dynamic temporal traffic condition. Hence, we propose to model the generation of speed v_i in a nonlinear manner:

$$\begin{aligned} \mathbf{h}_i &= \mathrm{SELU}(W_1\rho_i + W_2\mathbf{c}) \\ \mu_i &= \langle \theta_\mu, \mathbf{h}_i \rangle, \ \sigma_i^2 = \langle \theta_\sigma, \mathbf{h}_i \rangle \\ P(v_i|\rho_i, \mathbf{c}) &= \mathcal{N}\left(\mu_i, \sigma_i^2\right) \end{aligned} \tag{5}$$

In the above equation, $\mathrm{SELU}(\cdot)$ is the self-normalizing nonlinear activation function [24] and W_1, W_2, θ_μ, θ_σ are parameters to be

learned and shared across all road segments. The Gaussian distribution is chosen due to the fact that travel speeds can be perfectly modeled by it [25]. The above generative process generalizes the classical linear factorization models. We actually can recover them by substituting the non-linear activation function SELU with the identity function.

4.4 Aggregating Road Segment Speeds

The Gaussian distribution of speed on road segment r_i naturally leads to Inverse-Gaussian distribution for travel time. The probability density function of Inverse-Gaussian distribution is given as follows:

$$p(t;\lambda,\mu) = \sqrt{\frac{\lambda}{2\pi t^3}} \exp\left\{\frac{-\lambda(t-\mu)^2}{2\mu^2 t}\right\} \tag{6}$$

where $\lambda, \mu > 0$ are parameters *i.e.,* mean μ and variance μ^3/λ. The Inverse-Gaussian distribution describes the first passage time of a Brownian random walk [1], and thus it is suitable for modeling travel time in our problem setting.

However, we cannot simply sum up the travel time distributions of r_i's to form the travel time distribution of the entire route $\mathbf{r}$, because Inverse-Gaussian distribution is not closed under addition. In other words, modeling $t = \sum_i \ell_i / v_i$ leads to an intractable model (here ℓ_i denotes the length of road segment r_i). Similar to [25], in our proposed solution, we consider the average speed of route $\mathbf{r}$, denoted by $v_\mathbf{r}$, as the weighted sum of v_i, *i.e.,*

$$v_\mathbf{r} = \sum_{i=1}^{n} w_i v_i, \quad w_i = \frac{\ell_i}{\sum_{j=1}^{n} \ell_j}.$$

Then we have

$$\mathbb{E}[v_\mathbf{r}] = \mathbb{E}\left[\sum_{i=1}^{n} w_i v_i\right] = \sum_{i=1}^{n} w_i \mu_i \tag{7}$$

$$\mathrm{Var}[v_\mathbf{r}] = \mathrm{Var}\left(\sum_{i=1}^{n} w_i v_i\right) = \sum_{i=1}^{n} w_i^2 \sigma_i^2 \tag{8}$$

Consequently, $P(t|\mathbf{r},\mathbf{v})$ will be a $\mathrm{IG}(\mu,\lambda)$ with

$$\mu = \frac{\sum_i^n \ell_i}{\mathbb{E}[v_\mathbf{r}]}, \quad \lambda = \frac{\mu^3}{(\sum_i^n \ell_i)^2 \mathrm{Var}[v_\mathbf{r}]}. \tag{9}$$

We scale route variance with the square of route length–$\sum_i^n \ell_i$ in Equation 9. This is to take into consideration travel time uncertainty is likely higher for longer route.

Note that Equation 8 holds only when v_i's are mutually independent. This may not be true in reality as speeds of spatially contiguous road segments are highly correlated. On the other hand, this equation offers us the insight that $\mathrm{Var}[v_\mathbf{r}]$ can be represented as a weighted sum of $\mathrm{Var}[v_i]$. Next, we present a better way to assign weights, *i.e.,* an attention mechanism-based method to adaptively assign weights for road segments.

The general idea of attention mechanism [4] is to learn the weights for a collection of inputs under a query vector $\mathbf{q}$. It assigns a relatively larger weight for the input which is more important under query $\mathbf{q}$. In our case, the inputs are the hidden states $\mathbf{h}_i$ in Equation 5, and the query vector is the traffic representation $\mathbf{c}$ because the uncertainty of a road segment is largely determined by the traffic condition. More formally, we have

$$\begin{aligned} a_i &= \langle W\mathbf{c}, \mathbf{h}_i \rangle \\ \mathrm{Var}[v_\mathbf{r}] &= \sum_{i=1}^{n} \mathrm{softmax}(a_i)\sigma_i^2 \end{aligned} \tag{10}$$

where W is the parameter matrix to be learned.

Notably, as we collectively aggregate v_i's to generate t, we actually amortize the delays caused at the crossroads into their adjacent road segments. Also note that we directly use the parameters of $P(v_i|\rho_i,\mathbf{c})$ to form $P(t|\mathbf{r},\mathbf{v})$. As a result, v_i becomes the virtual node in the graphical model (see Figure 2).

4.5 Variational Loss and Learning Algorithm

The generative process implies the following log-likelihood function

$$\mathcal{L}(\Theta) = \log \int_\mathbf{c} \int_\rho \mathrm{d}\mathbf{c}\, \mathrm{d}\rho \prod_{m=1}^{M} P(\mathbf{c}_m|C_m) \prod_{i=1}^{n_m} P(\rho_{mi}|r_{mi}) P(v_{mi}|\rho_{mi},\mathbf{c}_m) P(t_m|\mathbf{r}_m,\mathbf{v}_m)$$

in which Θ denotes all the parameters involved. Unfortunately, the log-likelihood is intractable due to the appearance of the latent variable integrals. Further, the MCMC based methods [3, 32] are not suitable for our case due to their slow convergence speed.

To circumvent this, we make an important observation that travel time t is actually a kind of reflection of the real-time traffic condition indicator C. Therefore, we can view $P(\mathbf{c}|C)$ as the encoder and $P(t|\mathbf{r},\mathbf{c})$ as the decoder of the real-time traffic condition. In the light of variational auto-encoder [23], the log-likelihood for one route could then be approximated as (lower bound)

$$\begin{aligned} \mathcal{L}(\Theta) &\geq \mathbb{E}_{\mathbf{c},\rho}[\log P(t|\mathbf{r},\mathbf{c})] \\ &= \mathbb{E}_{\mathbf{c},\rho}\left[\sum_{i=1}^{n} \log P(\rho_i|r_i) + \log P(v_i|\rho_i,\mathbf{c}) + \log P(t|\mathbf{r},\mathbf{v})\right] \end{aligned}$$

where $\mathbf{c} \sim P(\mathbf{c}|C)$ and $\rho_i \sim P(\rho_i|r_i)$.

This loss can be interpreted as the expected negative reconstruction error. Indeed, if we put a weak prior distribution for $\mathbf{c}$ and ρ_i respectively, we recover the well-known variational loss [7] as

$$\begin{aligned} \mathcal{L}_v(\Theta) = \mathbb{E}_{\mathbf{c},\rho}[\log P(t|\mathbf{r},\mathbf{c})] &- KL(P(\mathbf{c}|C)||P(\mathbf{c})) \\ &- \sum_{i=1}^{n} KL(P(\rho_i|r_i)||P(\rho_i)) \end{aligned} \tag{11}$$

where $KL(\cdot||\cdot)$ indicates KL divergence. We now can approximate the variational loss using the Monte Carlo method and propagate the gradients backward using the reparameterization trick [23]. In our case, we reparameterize the Gaussian random variable $\mathbf{z}$ ($\mathbf{c}$ and ρ_i) as:

$$\mathbf{z} \sim \mathcal{N}\left(\mathbf{z}|\mu,\sigma^2\right) \Leftrightarrow \mathbf{z} = \mu + \sigma\epsilon, \quad \epsilon \sim \mathcal{N}(0,1)$$

and calculate the gradient w.r.t the parameters θ as

$$\nabla_\theta \mathbb{E}_\mathbf{z}[f(\mathbf{z})] = \nabla_\theta \mathbb{E}_\epsilon[f(\mu+\sigma\epsilon)] = \mathbb{E}_\epsilon[\nabla_\theta f(\mu+\sigma\epsilon)]$$

where $f(\mathbf{z})$ represents the log-likelihood function of the parameters. In this sense, our model becomes the Variational Auto-Encoder

Algorithm 1: Learning Algorithm

Input: Training dataset: $(\mathbf{r}_m, C_m, t_m)_{m=1}^{M}$
Output: Parameter set: Θ

1 **while** *training is true* **do**
2 $\quad \mathcal{B} \leftarrow$ A random minibatch of data;
3 $\quad \mathbf{c}_k \sim P(\mathbf{c}_k|C_k) \quad \forall C_k \in \mathcal{B}, k = 1, \ldots, |\mathcal{B}|$;
4 $\quad \rho_{ki} \sim P(\rho_{ki}|r_{ki}) \quad \forall \mathbf{r}_k \in \mathcal{B}, \forall r_{ki} \in \mathbf{r}_k, k = 1, \ldots, |\mathcal{B}|$;
5 $\quad \mathcal{L} \leftarrow$ Calculating the variational loss using Equation 11;
6 $\quad$ **for** $\theta \in \Theta$ **do**
7 $\quad\quad \mathbf{g}_\theta \leftarrow \nabla_\theta \mathcal{L}_v(\Theta)$;
8 $\quad\quad \theta \leftarrow \theta + \Gamma(\mathbf{g}_\theta)$;
9 **return** Θ;

(VAE) which is fully differentiable. The learning algorithm is presented in Algorithm 1.

We highlight that it is unnecessary to compute the traffic condition indicator C for every trajectory in training dataset (Line 3 in Algorithm 1). Instead, we discretize the temporal dimension into slots and let the trajectories whose start time fall into the same slot share the same C (see Section 5.1).

4.6 Prediction

Given the learned parameters Θ, a query route $\mathbf{r}$ along with its real-time traffic condition indicator C, we run the forward computation (Lines 2–5 in Algorithm 1) to predict the travel time distribution $P(t|\mathbf{r}, C)$.

Specifically, we compute the mean value $\mu(\mathbf{s}_i, \mathbf{u}_i)$ (in Equation 2) for each road segment r_i and $\mu(\mathbf{f})$ (in Equation 3) for traffic condition indicator C. Different from the training stage where we perform sampling operation to draw instances of ρ_i and $\mathbf{c}$, we simply feed these mean values into the subsequent computation to generate v_i (Equation 5) in prediction. Once we obtain v_i for each r_i, we aggregate them to output the parameters μ, λ (Equation 9) of the travel time distribution $P(t|\mathbf{r}, C)$.

5 EXPERIMENTS

We evaluate the effectiveness and scalability of DeepGTT on a real-world taxi dataset, against the state-of-the-art baselines.

5.1 Experimental setup

Dataset. The dataset was collected by 13,000 taxis during 28 days in a provincial capital city in China. It contains over 2.9 million trajectories. The sampling time interval between two consecutive points is around 30 seconds, and the map matching accuracy at this sampling rate can be as high as 99% [30]. We acquire the road network data from the Open Street Map [2]. There are 14,497 road segments in total and every road segment has the spatial features: the road type, the number of lanes, one way or not, and road length.

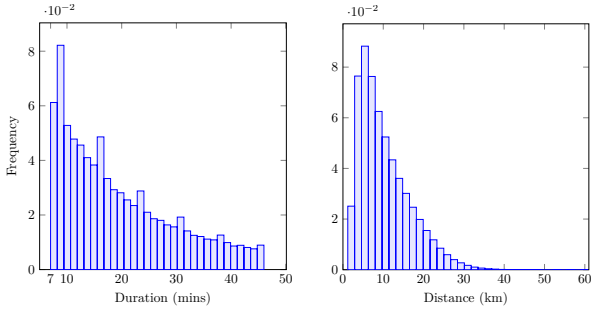
Figure 5 shows the spatial distribution of GPS points. As expected, routes at the central area of the city are frequently traveled whereas routes outside of the city are relatively less traveled by the vehicles. The average travel time is 19.4 minutes and average travel distance



Figure 5: The spatial distribution of the GPS points.

Table 2: Dataset statistics.

Measures	min	max	mean
Duration (mins)	7	46	19.4
Distance (km)	1.2	60	11.4



(a) Dist. of duration (minutes) (b) Dist. of travel distance (km)

Figure 6: The distributions of duration (minutes) and travel distance (km)

is 11.4 kilometers. Table 2 reports main statistics of the trips and Figure 6 plots distributions of trip duration and travel distance.

We use the first 18 days' trajectories as training dataset, the middle 3 days' trajectories as validation set. The remaining 7 days' trajectories are used for testing. The dataset size of training, validation, and testing is 1.7, 0.3, and 0.9 million respectively.

Baseline Methods. We evaluate DeepGTT on two tasks: (i) travel time estimation, and (ii) route recovery from sparse trajectories.

For travel time estimation, we compare DeepGTT with three baseline methods, namely, DeepTTE [38], WDR [41], and MURAT [27]. For route recovery from sparse trajectories, we demonstrate how we enhance the state-of-art route recovery method STRS [42] by simply changing its travel time distribution component to DeepGTT.

- DeepTTE and WDR are two leading travel time estimation models based on Deep Neural Nets. Both models treat road segments as tokens and compress a sequence of tokens (a route) to predict travel time by using RNN.

Table 3: Performance comparison of different methods.

Method	MURAT	WDR	DeepTTE	DeepGTT
RMSE (sec)	510.23	374.24	337.51	**193.04**
MAE (sec)	409.24	275.48	241.37	**141.75**

- MURAT is a multi-task representation learning based travel time estimation model. It only uses the origin-destination information.
- STRS is the state-of-the-art route recovery algorithm, which contains a travel time distribution learning component. The learning component first learns the mean value of travel time by trajectory regression, and then empirically study the relationship between the variance and mean value to derive the distribution.

Parameter settings. Our method[1] is implemented with PyTorch 0.4[2] and Julia 1.0[3], and trained with a Tesla K40 GPU. The platform runs on Ubuntu 14.04 OS with a Genuine Intel CPU.

The embedding sizes of $d^{(t)}$, $d^{(l)}$, $d^{(o)}$, $\mathbf{u}_i$ are set to 64, 32, 16, 200 respectively. The dimensions of the random variables are set as follows: $|\rho_i| = 256$, $|\mathbf{c}| = 400$, $|\mathbf{f}| = 600$, $|\mathbf{h}_i| = 500$. The space is partitioned into a 138×148 matrix C with cell size 200m $\times$ 200m, and $\Delta = 30$ minutes. We discretize the temporal dimension with a slot size 20 (minutes); two trips share the same traffic condition indicator C if their start time fall into the same slot. The batch size $|\mathcal{D}|$ in the training algorithm 1 is 150. The model is optimized by Amsgrad [21, 33] with an initial learning rate 0.001 for 10 epochs, and early stopping is used on validation dataset. We select the best hyperparameters for the baseline methods on valuation dataset.

5.2 Evaluation on travel time estimation

Overall performance. We evaluate all methods on this task by RMSE (Root Mean Square Error) and MAE (Mean Average Error),

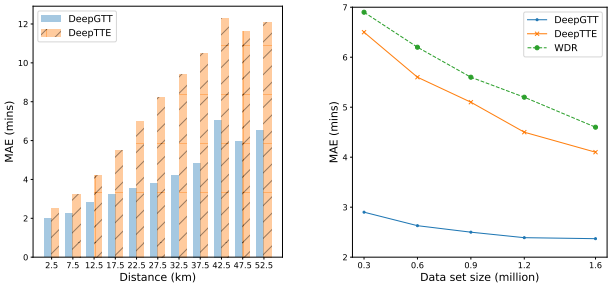
$$\text{RMSE}(\mathbf{t}, \hat{\mathbf{t}}) = \sqrt{\frac{1}{|\mathbf{t}|}\|\mathbf{t} - \hat{\mathbf{t}}\|_2^2}, \quad \text{MAE}(\mathbf{t}, \hat{\mathbf{t}}) = \frac{1}{|\mathbf{t}|}\|\mathbf{t} - \hat{\mathbf{t}}\|_1$$

where $\mathbf{t}$, $\hat{\mathbf{t}}$ denote ground truth, estimated value respectively.

Reported in Table 3, simple model MURAT leads to relatively large errors, and WDR surpasses it by a fair margin. Among all the baselines DeepTTE achieves the best results. Our model DeepGTT outperforms all the baselines by a large margin, for three reasons: 1) DeepGTT interprets the generation of travel time in a logical way instead of learning by brute force; and 2) DeepGTT makes prediction based on the learned real-time traffic representation, rather than assuming that traffic conditions in the same time slot are temporally-invariant; 3) DeepGTT optimizes the full distribution rather than a single mean value, and thus it could incorporate more variability for more accurate prediction.

Remarkably, DeepGTT reduces MAE to 141.75 seconds for the trips with 19.4 minutes average duration which may potentially facilitate many existing web services.

[1]The code is available at https://github.com/boathit/deepgtt
[2]https://pytorch.org
[3]https://julialang.org



(a) Travel distance (b) Training dataset size

Figure 7: (a) MAE changes against travel distance, and over the training dataset size, respectively.

Impact of travel distance. We now study the impact of travel distance on the performance of different models. To this end, we group the trips in test dataset into subgroups by their lengths (in 5KM step), $[0, 5), [5, 10), \ldots, [50, 55)$, and study the performance of different models on these subgroups. Figure 7(a) plots MAEs of DeepTTE and DeepGTT against travel distance (we omit other baselines due to their large errors). Not surprisingly, MAEs increase with the travel distance. Longer trips typically involve more road segments and have larger uncertainty. It is worth noting that the performance difference between DeepTTE and DeepGTT grows with the travel distance. This result suggests that DeepGTT is more trustful for long trip estimation.

Impact of training data size. One of appealing properties of DeepGTT inherited from probabilistic models is that it is quite data efficient [6, 12]. To demonstrate this property, we study the change of MAE with different number of training data points from 0.3 to 1.7 million, reported in Figure 7(b). Observe that even trained with only 0.3 million data points, DeepGTT surpasses baseline methods that are trained with the full training data by a notable margin. This can be explained by the fact that DeepGTT interprets the generation of travel time in a more plausible manner, and thus it could reveal the hidden dependencies among relevant variables and explore training data in a more efficient way. The data efficient property enables DeepGTT to be trained in a much faster speed as well, making it attractive to the online web services that require frequent updates.

5.3 Performance on route recovery

In this subsection, we demonstrate how DeepGTT could enhance the existing leading route recovery algorithm, in which we need to access $P(t|\mathbf{r})$ for any $t > 0$. As described in Section 1, route recovery from sparse trajectories is formulated as

$$\underset{\mathbf{r}}{\text{argmax}}\, P(t|\mathbf{r})P(\mathbf{r}), \quad \mathbf{r} \in \{\text{candiate routes}\}.$$

The first term $P(t|\mathbf{r})$ describes the probability of a route $\mathbf{r}$ for an observed travel time t, namely, the temporal component. The second term $P(\mathbf{r})$ describes the probability from the spatial transition perspective, namely, the spatial component. Actually, these are the

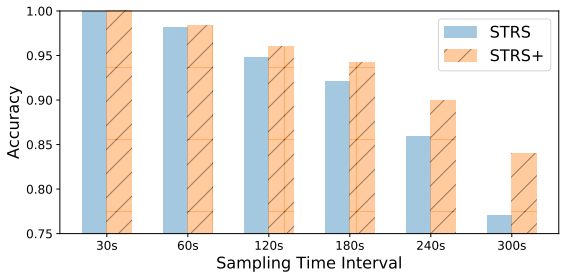


Figure 8: Accuracy comparison.

two main components in the state-of-the-art sparse route recovery algorithm STRS [42].

In this experiment, we replace STRS's temporal component with DeepGTT and refer to the new model as STRS+. The performance of route recovery is evaluated by route recovery accuracy [42]: the ratio of the length of correctly inferred road segments against the maximum value of the length of ground truth $\mathbf{r}_G$ and the length of inferred route $\mathbf{r}_I$.

$$\text{accuracy} = \frac{\text{length}(\mathbf{r}_G \cap \mathbf{r}_I)}{\max(\text{length}(\mathbf{r}_G), \text{length}(\mathbf{r}_I))}$$

We randomly select 10k trajectories from the test dataset and match these trajectories into the map. The matched results are used as ground truth. Then we down-sample these trajectories and recover the routes using STRS, and STRS+, respectively. Figure 8 reports the accuracy of STRS and STRS+, over different sampling time intervals (second). As sampling time interval increases, accuracy of both methods drop as expected. The reason is that larger sampling time interval leads to more possible candidate routes to be inferred between two sample points. Observe that accuracy of STRS+ is always higher than that of STRS. In particular, the difference between their accuracies becomes more evident when the sample time interval is larger. This result shows the superiority of STRS+.

5.4 The impact of real time traffic

In this paper, we relax the assumption that traffic conditions are temporally invariant for the same time slot. Instead, we propose to learn a real-time traffic representation $\mathbf{c}$, and by conditioning on which we generate the travel time. It is natural to ask the effectiveness of this real time traffic representation $\mathbf{c}$. We design two experiments to answer this question.

Experiment 1. We fill the traffic condition indicator C with a constant that is specific to the time slot. In this case, our model is forced to "assume" that traffic conditions are temporally invariant. The RMSE, MAE in travel time estimation increases from 193.04 to 272.45, and from 141.75 to 200.02, respectively. In other words, adopting the assumption that traffic conditions are temporally invariant in DeepGTT leads to a 41% performance drop in terms of MAE. This result implies that an appropriate modeling of $P(\mathbf{c}|C)$ is crucial for accurate travel time estimation.

Experiment 2. As a case study, we randomly select a route $\mathbf{r}_{102}$ with length 21.37km and evaluate the change of its travel time

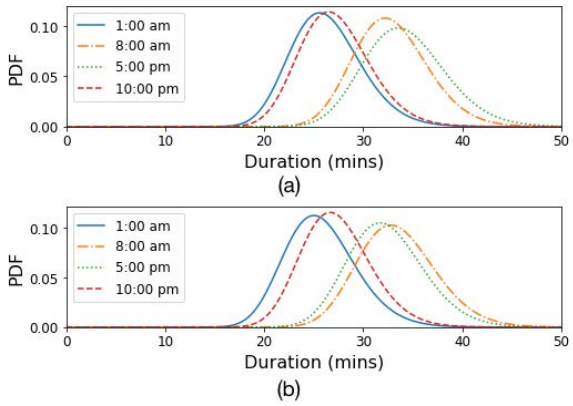


Figure 9: The travel time distributions of route $\mathbf{r}_{102}$ on Jan 03 (a) and Jan 17 (b), both of them are Saturday.

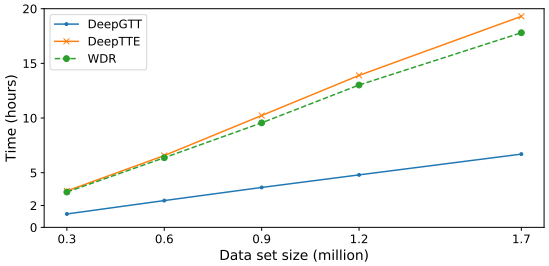


Figure 10: Time cost of all methods grows linearly with the training data size. However, DeepGTT has a much smaller constant term than its competitors.

distributions by taking different real-time traffic condition indicators C's. More specifically, we calculate C by using the (sub-) trajectories in different time slots and feed these C's into the model while keeping the route unchanged. Figure 9a and 9b plot the travel time distributions at different time slots on Jan 03, 2015 and Jan 17, 2015 respectively (both are Saturday). Two points are worth noting: I) The travel time distributions at 1:00am and 10:00pm exhibit smaller expectations and variances. This is reasonable because the traffic conditions at these time slots are less congested, compared with peak hours, *e.g.*, 8:00am and 5:00pm. II) Even both 8:00am and 5:00pm are peak hours, the traffic is more congested at 5:00pm on Jan 03, 2015 whereas it is more congested at 8:00am on Jan 17, 2015. Our observations suggest that traffic conditions in the same time slot are not invariant across the temporal dimension. The assumption made in previous studies does not hold [27, 38, 39, 41, 42].

5.5 Time Cost Study

The training time complexity of DeepGTT grows linearly with the number of trajectories. Figure 10 demonstrates that the training time of different methods vary with the data size. Even though the time cost of both DeepTTE and WDR grows linearly with the size of training data, they have larger constant terms compared with

DeepGTT. This is due to their employment of RNN which is more computationally expensive.

For prediction, DeepGTT can process over 0.9 million trajectories within 81 seconds with a mini-batch size 256. The fast running speed makes it ideal for online deployment.

6 CONCLUSION

For the first time, we develop a deep generative model DeepGTT for travel time distribution learning. We tackle the data sparsity challenge by presenting two techniques, amortization and spatial smoothness embedding, in the lower layer. We relax the assumption that traffic conditions are temporally-invariant by proposing a convolution neural net based real time traffic representation learning and a hierarchical structure. The introduction of an auxiliary random variable v_i in the middle layer enables separating the statical spatial features from the dynamically changing real-time traffic. As a result, we could incorporate these heterogeneous influencing factors into a single model. The further derived variational loss enables the model to be trained in an end-to-end fashion, and makes DeepGTT applicable to large-scale data sets. DeepGTT interprets the generation of travel time in a plausible manner rather than learning by brute force, thus it produces more accurate results and is data-efficient.

Evaluated on a real-world large-scale data set, we first demonstrate the superiority of DeepGTT over existing methods, including two recently proposed deep-neural-net-based approaches, in two tasks: travel time estimation and route recovery. Notably, even trained with a small fraction of data DeepGTT is able to produce substantially better results than the state-of-the-art baselines. We further design experiments to verify that an appropriate modeling of real-time traffic is crucial for accurate travel time distribution prediction, and the assumption made in previous studies that traffic conditions are temporally-invariant does not hold. Finally, we empirically compare the time cost of DeepGTT with the other two deep-neural-net-based approaches, which demonstrates that DeepGTT owns much faster training speed.

In the future work, we would like to enhance the performance of DeepGTT by investigating more rich posterior distribution for the real time traffic representation with the normalizing flows [34]. In the present solution, DeepGTT assumes that the travel times of longer routes tend to show higher variance which may not hold for highways, and we will attempt to address this in the future work. Due to space limitation, we also omit the analysis of the impact of different explanatory factors' dimensionality and cell size, and we would like to give a thorough study in the future version. It is also desirable to explore the usage of DeepGTT in the problems such as ride-sharing, taxi-dispatching, travel time-related trajectory anomaly detection and time-aware trajectory representation learning [26, 44].

ACKNOWLEDGEMENTS

This research was conducted in collaboration with Singapore Telecommunications Limited and partially supported by the Singapore Government through the Industry Alignment Fund - Industry Collaboration Projects Grant. This work was also supported in part by Singapore MOE Tier-2 grant MOE2016-T2-1-137, MOE Tier-1 grant RG31/17, and NSFC under the grant 61772537. The authors would like to thank the anonymous reviewers who give the thoughtful comments and helpful suggestions.

REFERENCES

[1] [n. d.]. https://en.wikipedia.org/wiki/Inverse_Gaussian_distribution
[2] [n. d.]. https://www.openstreetmap.org
[3] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. 2003. An introduction to MCMC for machine learning. *Machine learning* 50, 1-2 (2003), 5–43.
[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014).
[5] Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. 2017. Automatic Differentiation in Machine Learning: a Survey. *JMLR* 18 (2017), 153:1–153:43.
[6] Christopher M Bishop. 2006. *Pattern recognition and machine learning*. springer.
[7] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. 2016. Variational Inference: A Review for Statisticians. *CoRR* abs/1601.00670 (2016).
[8] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *JMLR* 3 (2003), 993–1022.
[9] Yun Cheng, Xiucheng Li, Zhijun Li, Shouxu Jiang, and Xiaofan Jiang. 2014. Fine-Grained Air Quality Monitoring Based on Gaussian Process Regression. In *ICONIP 2014*. 126–134.
[10] Yun Cheng, Xiucheng Li, Zhijun Li, Shouxu Jiang, Yilong Li, Ji Jia, and Xiaofan Jiang. 2014. AirCloud: a cloud-based air-quality monitoring system for everyone. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems, SenSys '14, Memphis, Tennessee, USA, November 3-6, 2014*. 251–265.
[11] Peter Dayan, Geoffrey E Hinton, Radford M Neal, and Richard S Zemel. 1995. The helmholtz machine. *Neural computation* (1995), 889–904.
[12] S. M. Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E. Hinton. 2016. Attend, Infer, Repeat: Fast Scene Understanding with Generative Models. In *NIPS 2016, December 5-10, 2016, Barcelona, Spain*. 3225–3233.
[13] Raghu K. Ganti, Mudhakar Srivatsa, and Tarek F. Abdelzaher. 2014. On Limits of Travel Time Predictions: Insights from a New York City Case Study. In *ICDCS 2014, Madrid, Spain, June 30 - July 3, 2014*. 166–175.
[14] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. 1995. *Bayesian data analysis*. Chapman and Hall/CRC.
[15] Prem Gopalan, Jake M Hofman, and David M Blei. 2013. Scalable recommendation with poisson factorization. *CORR* (2013).
[16] Prem K Gopalan, Laurent Charlin, and David Blei. 2014. Content-based recommendations with poisson factorization. In *NIPS 2014*. 3176–3184.
[17] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. 2013. Stochastic variational inference. *JMLR* 14, 1 (2013), 1303–1347.
[18] Liangjie Hong, Amr Ahmed, Siva Gurumurthy, Alexander J. Smola, and Kostas Tsioutsiouliklis. 2012. Discovering geographical topics in the twitter stream. In *WWW 2012, Lyon, France, April 16-20, 2012*. 769–778.
[19] Timothy Hunter, Aude Hofleitner, Jack Reilly, Walid Krichene, Jerome Thai, Anastasios Kouvelas, Pieter Abbeel, and Alexandre M. Bayen. 2013. Arriving on time: estimating travel time distributions on large-scale road networks. *CoRR* abs/1302.6617 (2013).
[20] Tsuyoshi Idé and Masashi Sugiyama. 2011. Trajectory Regression on Road Networks. In *AAAI 2011, San Francisco, California, USA, August 7-11, 2011*.
[21] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *ICLR 2015* abs/1412.6980 (2014).
[22] Diederik P. Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised Learning with Deep Generative Models. In *NIPS 2014, December 8-13 2014, Montreal, Quebec, Canada*. 3581–3589.
[23] Diederik P. Kingma and Max Welling. 2013. Auto-Encoding Variational Bayes. *ICLR 2013* abs/1312.6114 (2013).
[24] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. 2017. Self-Normalizing Neural Networks. In *NIPS 2017, 4-9 December 2017, Long Beach, CA, USA*. 972–981.
[25] Mu Li, Amr Ahmed, and Alexander J. Smola. 2015. Inferring Movement Trajectories from GPS Snippets. In *WSDM 2015, Shanghai, China, February 2-6, 2015*. 325–334.
[26] Xiucheng Li, Kaiqi Zhao, Gao Cong, Christian S. Jensen, and Wei Wei. 2018. Deep Representation Learning for Trajectory Similarity Computation. In *ICDE 2018, Paris, France, April 16-19, 2018*. 617–628.
[27] Yaguang Li, Kun Fu, Zheng Wang, Cyrus Shahabi, Jieping Ye, and Yan Liu. 2018. Multi-task Representation Learning for Travel Time Estimation. In *SIGKDD 2018, London, UK, August, 2018*.
[28] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *WWW 2018, Lyon, France, April 23-27, 2018*. 689–698.

[29] Kevin P Murphy. 2012. *Machine learning: a probabilistic perspective*. Cambridge, MA.

[30] Paul Newson and John Krumm. 2009. Hidden Markov map matching through noise and sparseness. In *SIGSPATIAL 2009, November 4-6, 2009, Seattle, Washington, USA, Proceedings*. 336–343.

[31] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *SIGKDD 2014, New York, NY, USA - August 24 - 27, 2014*. 701–710.

[32] Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. 2008. Fast collapsed gibbs sampling for latent dirichlet allocation. In *SIGKDD 2008*. ACM, 569–577.

[33] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. 2018. On the convergence of adam and beyond. *ICLR 2018*.

[34] Danilo Jimenez Rezende and Shakir Mohamed. 2015. Variational Inference with Normalizing Flows. In *ICML 2015, Lille, France, 6-11 July 2015*. 1530–1538.

[35] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *ICML 2014, Beijing, China, 21-26 June 2014*. 1278–1286.

[36] Dalia Tiesyte and Christian S. Jensen. 2008. Similarity-based prediction of travel times for vehicles traveling on known routes. In *SIGSPATIAL 2008, November 5-7, 2008, Irvine, California, USA, Proceedings*. 14.

[37] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural Discrete Representation Learning. In *NIPS 2017, 4-9 December 2017, Long Beach, CA, USA*. 6309–6318.

[38] Dong Wang, Junbo Zhang, Wei Cao, Jian Li, and Yu Zheng. 2018. When Will You Arrive? Estimating Travel Time Based on Deep Neural Networks. In *AAAI 2018, New Orleans, Louisiana, USA, February 2-7, 2018*.

[39] Hongjian Wang, Yu-Hsuan Kuo, Daniel Kifer, and Zhenhui Li. 2016. A simple baseline for travel time estimation using large-scale trip data. In *SIGSPATIAL 2016, Burlingame, California, USA, October 31 - November 3, 2016*. 61:1–61:4.

[40] Yilun Wang, Yu Zheng, and Yexiang Xue. 2014. Travel time estimation of a path using sparse trajectories. In *SIGKDD 2014, New York, NY, USA - August 24 - 27, 2014*. 25–34.

[41] Zheng Wang, Kun Fu, and Jieping Ye. 2018. Learning to Estimate the Travel Time. In *KDD 2018, London, UK, August 19-23, 2018*. 858–866.

[42] Hao Wu, Jiangyun Mao, Weiwei Sun, Baihua Zheng, Hanyuan Zhang, Ziyang Chen, and Wei Wang. 2016. Probabilistic Robust Route Recovery with Spatio-Temporal Dynamics. In *SIGKDD 2016, San Francisco, CA, USA, August 13-17, 2016*. 1915–1924.

[43] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. 2016. Attribute2image: Conditional image generation from visual attributes. In *ECCV 2016*. 776–791.

[44] Di Yao, Gao Cong, Chao Zhang, and Jingping Bi. 2019. Computing Trajectory Similarity in Linear Time: A Generic Seed-Guided Neural Metric Learning Approach. In *ICDE 2019, Macau, China, April 8-12, 2019*.

[45] Zhijun Yin, Liangliang Cao, Jiawei Han, Chengxiang Zhai, and Thomas S. Huang. 2011. Geographical topic discovery and comparison. In *WWW 2011, Hyderabad, India, March 28 - April 1, 2011*. 247–256.

[46] Quan Yuan, Gao Cong, and Chin-Yew Lin. 2014. COM: a generative model for group recommendation. In *KDD '14, New York, NY, USA - August 24 - 27, 2014*. 163–172.

[47] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat-Thalmann. 2013. Who, where, when and what: discover spatio-temporal topics for twitter users. In *KDD 2013, Chicago, IL, USA, August 11-14, 2013*. 605–613.

[48] Jiangchuan Zheng and Lionel M. Ni. 2013. Time-Dependent Trajectory Regression on Road Networks via Multi-Task Learning. In *AAAI 2013, July 14-18, 2013, Bellevue, Washington, USA*.

Truth Inference at Scale: A Bayesian Model for Adjudicating Highly Redundant Crowd Annotations

Yuan Li
University of Melbourne
Melbourne, Australia
yuanl4@student.unimelb.edu.au

Benjamin I. P. Rubinstein
University of Melbourne
Melbourne, Australia
brubinstein@unimelb.edu.au

Trevor Cohn
University of Melbourne
Melbourne, Australia
tcohn@unimelb.edu.au

ABSTRACT

Crowd-sourcing is a cheap and popular means of creating training and evaluation datasets for machine learning, however it poses the problem of 'truth inference', as individual workers cannot be wholly trusted to provide reliable annotations. Research into models of annotation aggregation attempts to infer a latent 'true' annotation, which has been shown to improve the utility of crowd-sourced data. However, existing techniques beat simple baselines only in low redundancy settings, where the number of annotations per instance is low (≤ 3), or in situations where workers are unreliable and produce low quality annotations (e.g., through spamming, random, or adversarial behaviours.) As we show, datasets produced by crowd-sourcing are often not of this type: the data is highly redundantly annotated ($\geq$ 5 annotations per instance), and the vast majority of workers produce high quality outputs. In these settings, the majority vote heuristic performs very well, and most truth inference models underperform this simple baseline. We propose a novel technique, based on a Bayesian graphical model with conjugate priors, and simple iterative expectation-maximisation inference. Our technique produces competitive performance to the state-of-the-art benchmark methods, and is the only method that significantly outperforms the majority vote heuristic at one-sided level 0.025, shown by significance tests. Moreover, our technique is simple, is implemented in only 50 lines of code, and trains in seconds. [1]

KEYWORDS

truth inference, crowdsourcing, Bayesian methods, probabilistic models

ACM Reference Format:
Yuan Li, Benjamin I. P. Rubinstein, and Trevor Cohn. 2019. Truth Inference at Scale: A Bayesian Model for Adjudicating Highly Redundant Crowd Annotations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313459

[1]The code and the datasets are made available at https://github.com/yuan-li/truth-inference-at-scale.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313459

1 INTRODUCTION

Large volumes of labelled data are crucial for machine learning, retrieval and search, both for training supervised methods and for evaluation. Ideally, ground truth labels are collected from experts, but such sources are often too slow or costly, motivating the use of data annotation by crowdsourcing [2, 7]. This has been shown to be highly effective for tasks which are difficult for machines to automate but relatively easy for humans, such as document categorization, image annotation, and sentiment analysis. Using non-expert labels for training machine learning algorithms can be as effective as using gold standard annotations from experts [16, 21], despite being much cheaper and faster to obtain.

Unfortunately, crowdsourced labels are often noisy or biased, which may be due to workers' lack of expertise, workers cheating or otherwise not performing the annotation task correctly [6]. This problem can be alleviated by filtering workers based on their profile and working history. However, this problem is not fully solved by simple quality control mechanisms currently provided by crowdsourcing platforms; consequently, most crowd data studies collect several labels for each item from different workers, which are aggregated post-hoc to infer an accurate consensus annotation.

Both the average number of labels per item (#labels/#items), and the quality of worker labels, are important factors in the performance of aggregation methods. Low-redundancy (e.g. #labels/#items ≤ 3) and/or low-quality scenarios (e.g. there are adversarial workers) are challenging and have attracted much attention. Such scenarios require methods to make full use of a few available labels by being able to interpret labels from different workers. Models which incorporate per-worker confusion matrices [3, 10, 18, 22] are flexible enough to capture different types of worker behaviour, and have shown to be effective in low-redundancy and/or low-quality scenarios.

However, the simplest aggregation technique, majority voting (MV), still predominates in practice. MV is an ensemble where every worker is given equal vote, based on the assumption that with increasing redundancy, the most common label will converge to the true label. Despite its simplicity, on many real-world crowdsourced datasets, MV outperforms most models of truth inference, as we show in our experimental evaluation, and the best method from prior work only outperforms MV by a small margin. To explain this observation, and the continued popularity of MV, we identify several misalignments between the experimental setup in previous papers and real-world scenarios:

Redundancy & worker quality. In practice #labels/#items is usually 5+ to ensure sufficient redundancy, and crowdsourcing platforms provide mechanisms to select high quality workers (e.g. in the collection of question answering dataset

SQuAD [17], workers were filtered based on the amount and acceptance rate of their prior work history). In such settings, MV is very competitive. Worryingly, our results show that confusion-matrix-based methods are inferior to MV in terms of mean accuracy across datasets, despite their utility in low-resource and/or low-quality scenarios.

Scale. Several datasets used in prior work were small with respect to the number of labels collected from crowd (#labels). But on large-scale datasets, like `senti` used in our experiment with 569k labels, the scalability of inference becomes critical. For example, GLAD [26], achieving the third highest mean accuracy among all existing methods in our experiment, takes more than 3 hours to train, while MV runs in 3.5 seconds.

This work is also motivated by qualitative considerations: ease of understanding, implementation, and hyper-parameter setting, for which MV is compelling, being parameter free. However, empirically the accuracy of MV is far from perfect, even for highly redundantly annotated datasets. We seek accurate, fast solutions to the high-resource high-quality scenario that are easy to implement. Our method is one such solution.

This paper presents a novel method for aggregating crowdsourced labels for classification. We propose a generative Bayesian model, relaxing a discrete binary problem to a continuous regression task, which we model using a normal likelihood and conjugate inverse-Gamma prior. Our method allows for straightforward inference using expectation-maximisation (EM; [5]), which is implemented in 50 lines of code (see Appendix A). The resulting update rules reveal a novel criterion for learning from crowds—namely the idea of minimizing the likelihood that "every worker is making mistakes".

Our base model has interpretable parameters and intuitive inference rules, which makes it easy to understand and use. We extend our base binary model in three ways: estimating different weights for each label, to capture both the reliability of a worker votes in favour and against an item belonging to a class; a means of correcting for model deficiency; and employing the one-versus-rest strategy to allow the model to operate in a multi-class setting. Our experimental results show our proposed methods achieve the highest mean accuracy over 10 real-world datasets drawn from several different application areas, with different sizes and differing degrees of label redundancy, and a total running time on the same datasets second only to MV.

2 RELATED WORK

Ever since Dawid and Skene [3] proposed a model (DS) to aggregate the clinical diagnoses of doctors, various methods have been proposed to aggregate worker labels and infer true annotations. The DS model uses the confusion matrix to capture the probabilities that a worker label is generated for an item conditioned on the item's true annotation. Extending the DS model by adding Dirichlet priors for the values in the confusion matrix, Kim and Ghahramani [10] then integrated the confusion matrix parameters out and used Gibbs sampling for inference. Simpson et al. [20] developed variational Bayesian inference for the model proposed in [10], which is computationally efficient and outperforms the Gibbs sampling method. Furthermore, they analysed the clusters naturally formed by the inferred workers' confusion matrices. Venanzi et al. [22] assumed there are a fixed number of clusters of workers, with workers in the same cluster possessing similar confusion matrices instead of modelling individual confusion matrices as done in previous works. In this way worker clusters are formed at inference-time instead of post-inference. Adding a Dirichlet process to the model to generate the Dirichlet priors on worker confusion matrices, Moreno et al. [14] also allowed workers to generate their own confusion matrices from the cluster's confusion matrix in order to increase model flexibility.

Unlike the above works, Whitehill et al. [26] didn't use confusion matrices, but borrowed ideas from Item Response Theory (IRT) and proposed an unsupervised version of previous IRT models. These estimate not only the true annotations and worker ability but also the difficulty of items. Zhou et al. [29] also developed an algorithm estimating the true annotations by using a minimax entropy principle, which assumes there are categorical distributions per worker-item pair and the worker labels are generated from those categorical distributions. The objective there is in finding the true annotations that minimize the maximum entropy of those categorical distributions. Such categorical distributions are more concentrated on worker labels and can be roughly considered as having higher likelihood of generating labels.

The database community has also independently studied the problem of resolving conflicts between information from different sources. Demartini et al. [4] developed a model similar to the DS model that also used EM but just modelled a single accuracy parameter instead of a full confusion matrix. Li et al. [12] adopted the Gaussian to model error between the truth and worker label, which is similar to our model but not a probabilistic model. They developed an iterative approach which alternatively estimated the truth based on variances, and set the variances to be the upper-bounds of their 95% confidence interval based on the estimated truth. Aydin et al. [1] adopted a two-step procedure that iteratively updates estimated truth and worker weights so that the sum of worker weight times the distance between worker label and estimated truth is minimized.

3 PROPOSED MODELS

In this section, we first define the crowdsourced annotation aggregation problem, then describe our proposed model, Bayesian Weighted Average (BWA), for binary classification tasks, followed by an extension to multi-class classification tasks, and strategies to set its hyper-parameters.

Problem statement. We assume that a task contains a number of homogeneous *instances* or *items*, where each item has its own *true label*. The true labels are drawn from a *label set*. All instances are uploaded to a crowdsourcing platform by requesters. Generally the requesters don't know the true labels, so they ask *workers* to label them. Workers need to choose a label from the label set per item. Typically a worker only labels a small fraction of all items in a task. The labels collected from workers are called *worker labels*. Finally, aggregation methods are used to infer the true annotations based on all worker labels.

Table 1: Mathematical notation.

Symbol	Description
N	number of items
i	item index, $i \in \{1, \ldots, N\}$
W	number of workers
j	worker index, $j \in \{1, \ldots, W\}$
K	number of classes
k, l	class/label index, $k, l \in \{0, \ldots, K-1\}$
W_i	set of workers who have labelled item i
N_j	set of items that worker j has labelled
	Binary setting
z_i	item i's latent true label (continuous), $z_i \in \mathbb{R}$
y_{ij}	worker j's observed label to item i (discrete), $y_{ij} \in \{0, 1\}$
	Multi-class setting
z_{ki}	score given to item i by model k which classifies items into class-k or not class-k, $z_{ki} \in \mathbb{R}$
y_{kij}	binary indicator, equalling 1 if worker j labels item i as class k otherwise 0

3.1 The Bayesian Weighted Average Model

We first consider binary classification tasks. Without loss of generality, let $y_{ij} \in \{0, 1\}$ be worker j's label to item i and let $z_i \in \mathbb{R}$ represent item i's true label. We model z_i's as continuous variables to make tractable the following optimization problem for inferring the truth, Z,

$$Z^* = \arg\max_Z P(Z|Y, \alpha), \tag{1}$$

where Z and Y are collections of their lowercase variables, α denotes all hyper-parameters, $P(Z|Y, \alpha)$ is the posterior distribution for Z, and Z^* is the maximum a posteriori (MAP) estimate of Z. Further discussion of our modelling choices is provided in Section 4.

After Z^* is computed, the discrete true label can be determined by rounding, i.e. the true label is 1 if $z_i^* > 0.5$, otherwise 0. Frequently used symbols are defined in Table 1.

3.1.1 The Generative Process and Joint Distribution. Our proposed models are all probabilistic generative models. We assume z_i is drawn from a Gaussian prior with mean μ and inverse variance λ, and y_{ij} is drawn from a Gaussian distribution with mean z_i and inverse variance v_j. Here v_j is a worker specific value for worker j and every v_j is drawn from Gamma($\frac{a_v}{2}, \frac{b_v}{2}$). The plate diagram for this model is shown in Figure 1.

The joint distribution $P(Z, Y, V|\mu, \lambda, a_v, b_v)$ can be factorized as

$$\begin{aligned} &P(Z, Y, V|\mu, \lambda, a_v, b_v) = P(Z|\mu, \lambda) \cdot P(Y|Z, V) \cdot P(V|a_v, b_v) \\ &= \prod_i \mathcal{N}\left(z_i \middle| \mu, \lambda^{-1}\right) \\ &\quad \cdot \prod_i \prod_{j \in W_i} \mathcal{N}\left(y_{ij} \middle| z_i, v_j^{-1}\right) \cdot \prod_j \text{Gamma}\left(v_j \middle| \frac{a_v}{2}, \frac{b_v}{2}\right), \end{aligned} \tag{2}$$

where $V = \{v_j\}_{j=1}^W$. λ and v_j model the inverse variance of $z_i - \mu$ and $y_{ij} - z_i$. The larger λ and v_j are, the smaller the corresponding errors are likely to be. In this way they can be interpreted as encoding worker reliability, a perspective we explore further in Section 3.1.3.

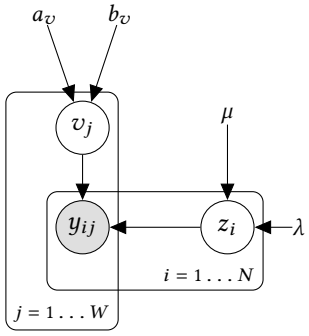


Figure 1: Plate diagram for our proposed BWA model (binary case).

Our proposed BWA model is deficient: data observations y_{ij} can only be binary, while the generative model has continuous support. We could force all y_{ij} to be binary variables ($\{0, 1\}$) then re-normalize the distributions accordingly to address the deficiency issue (Section 4.3). But the linking distribution between y_{ij}, z_i and v_j then becomes a logistic function, which is much more difficult to work with as no conjugate prior for v_j exists.

3.1.2 Inference. In order to find the most likely Z, we first integrate out V in Equation (2) to obtain $P(Z, Y|\alpha)$ where $\alpha = (\mu, \lambda, a_v, b_v)$ are the hyper-parameters.

$$\begin{aligned} &P(Z, Y|\alpha) = \int P(Z, Y, V|\mu, \lambda, a_v, b_v)\ \mathrm{d}V \\ &= \prod_i \mathcal{N}(z_i|\mu, \lambda^{-1}) \\ &\quad \cdot \prod_j \int \prod_{i \in N_j} \mathcal{N}\left(y_{ij} \middle| z_i, v_j^{-1}\right) \cdot \text{Gamma}\left(v_j \middle| \frac{a_v}{2}, \frac{b_v}{2}\right)\ \mathrm{d}v_j \\ &\propto \prod_i e^{-\frac{\lambda}{2}(z_i - \mu)^2} \cdot \prod_j \left(\frac{1}{2} b_v + \frac{1}{2} \sum_{i \in N_j} (z_i - y_{ij})^2\right)^{-\frac{a_v + |N_j|}{2}}. \end{aligned}$$

Since $P(Z, Y|\alpha) = P(Z|Y, \alpha)P(Y|\alpha)$ and $P(Y|\alpha)$ is independent from Z, maximising $P(Z, Y|\alpha)$ with respect to Z is equivalent to solving the program (1), i.e.

$$\arg\max_Z P(Z|Y, \alpha) = \arg\max_Z P(Z, Y|\alpha).$$

As we shortly show, λ, a_v, and b_v have intuitive meanings, and accordingly can be set manually, while μ can be optimized easily with Z. Therefore, the goal becomes to maximise $P(Z, Y|\alpha)$ with respect to Z and μ, which is equivalent to minimising the negative log likelihood $-\log P(Z, Y|\alpha)$ with respect to Z and μ:

$$\min_{Z, \mu} \sum_i \frac{\lambda}{2}(z_i - \mu)^2 + \sum_j \frac{a_v + |N_j|}{2} \log\left(b_v + \sum_{i \in N_j} (z_i - y_{ij})^2\right) + const., \tag{3}$$

where *const.* is the sum of values independent from Z and μ, and thus can be ignored during optimisation. We then use the EM

algorithm to minimize the above negative likelihood:

E step. $v_j|Z, Y, a_v, b_v \sim \text{Gamma}\left(\frac{a_v+|N_j|}{2}, \frac{b_v+\sum_{i\in N_j}(z_i-y_{ij})^2}{2}\right)$.

M step. Let $q(V) = P(V|Z^{\text{old}}, Y, a_v, b_v)$ and

$$Q(Z; Z^{\text{old}}) = \mathbb{E}_q \log P(Z, Y, V|\mu, \lambda, a_v, b_v) \ ,$$

then by setting the gradient of the Q function with respect to Z and μ to be zero, we obtain

$$z_i = \frac{\lambda\mu + \sum_{j\in W_i} \mathbb{E}_q v_j y_{ij}}{\lambda + \sum_{j\in W_i} \mathbb{E}_q v_j} \ , \tag{4}$$

where

$$\mathbb{E}_q v_j = \frac{a_v + |N_j|}{b_v + \sum_{i\in N_j}(z_i - y_{ij})^2} \ , \tag{5}$$

$$\mu = \frac{1}{N}\sum_i z_i \ .$$

3.1.3 Interpretations.

$\mathbf{z_i}$ *and its update rule.* In Equation (4), z_i is updated by a weighted arithmetic average of μ and $\{y_{ij}\}_{j\in W_i}$, weighted by λ and $\{\mathbb{E}_q v_j\}_{j\in W_i}$, respectively. Higher weights lead to a larger impact on average, so these weights can be considered as encoding worker reliability. In particular, λ is the weight for a "default worker" who always labels every item as μ.

$\mathbf{v_j}$ *and its estimation rule.* In Equation (5), $|N_j|$ is the number of items that worker j has labelled, $\sum_{i\in N_j}(z_i - y_{ij})^2$ is the sum of squared errors (SSE_j) that worker j has made. Note that $y_{ij} \in \{0, 1\}$ and $z_i \in [0, 1]$, thus $0 \le \text{SSE}_j \le |N_j|$. In the case that we know the true label, z_i is also binary just like y_{ij}, then SSE_j is simply the number of wrong judgements that worker j has made. The estimation of $\mathbb{E}_q v_j$ can be considered as a smoothed inverse error rate with a_v and b_v serving as the smoothing parameters. The larger the $\mathbb{E}_q v_j$, the smaller the worker j's smoothed error rate is.

Hyper-parameters. The values of a_v and b_v encode our belief that worker j has labelled a_v items before and b_v judgements were wrong, which naturally suggests $b_v \le a_v$ and further ensures $\mathbb{E}_q v_j \ge 1$. So when $b_v \le a_v$, even if all labels from worker j in task k are wrong, we still have $\mathbb{E}_q v_j \ge 1$. This mechanism guarantees everyone an unconditional minimum weight of 1 to vote for the truth. We may consider setting $b_v > a_v$ so that $\mathbb{E}_q v_j$ is allowed to be smaller than 1, however, in this setting, a worker who gets 10/10 wrong will be penalized more than another worker who gets 1000/1000 wrong due to the weaker smoothing effect on the latter. Hence both estimations are smaller than 1, but the estimation for the latter is closer to 1. As this is not the desired effect, we recommend the natural setting of $b_v \le a_v$. As 1 is the lower bound of $\mathbb{E}_q v_j$, we choose $\lambda = 1$ so that the "default worker" has limited ability to compete with other workers, but is most helpful in tie breaking.

3.2 Multi-class Extension

So far, our proposed models assume binary labels. However, it's easy to extend them to the multi-class setting by employing a one-versus-rest strategy, a commonly used reduction to binary classifiers such as SVMs.

Let K denote the number of classes, $k \in \{0, \ldots, K\}$ a class index, y_{kij} a binary indicator that equals 1 if worker j labels item i as class k otherwise 0, Y_k the collection of y_{kij}'s. Then our models can work on Y_k which is binary and output Z_k, a collection of z_{ki}'s, where z_{ki} is the score that our model gives to item i. The higher z_{ki} is, the more likely our model believes item i's true label is k. For item i, our model generates a score z_{ki} for each k value given different Y_k. Finally, $k^* = \arg\max_k z_{ki}$ is picked as the estimated true label for item i.

3.3 Hyper-parameter Settings

We discuss how to set b_v when a_v is given in this section. As a_v and b_v translate to our prior belief that every worker has labelled a_v items and made b_v mistakes, $b_v/a_v = \varepsilon$ is our assumed error rate. This error rate should depend on the quality of worker labels in a dataset, so we seek to estimate ε instead of setting it to be constant regardless of datasets. However, estimating ε requires true labels which we don't have, so a natural way is to estimate them by a simple aggregation method such as majority voting, then calculate ε based on the estimates.

We first consider the binary case. Let $\hat{z}_i = \frac{1}{|W_i|}\sum_{j\in W_i} y_{ij}$ to be the majority voting results, then the overall error rate ε is

$$\varepsilon = \frac{\sum_i \sum_{j\in W_i}(y_{ij} - \hat{z}_i)^2}{\sum_i |W_i|} \ .$$

To simplify ε, we define n_{i0} and n_{i1} to be the number of workers who have labelled item i as 0 or 1, respectively. Then $|W_i| = n_{i0}+n_{i1}$, $\hat{z}_i = \frac{n_{i1}}{n_{i0}+n_{i1}}$ and we obtain

$$\begin{aligned}\varepsilon =& \frac{\sum_i n_{i0}(0-\hat{z}_i)^2 + n_{i1}(1-\hat{z}_i)^2}{\sum_i n_{i0} + n_{i1}} \\ =& \frac{\sum_i n_{i0}(\frac{n_{i1}}{n_{i0}+n_{i1}})^2 + n_{i1}(\frac{n_{i0}}{n_{i0}+n_{i1}})^2}{\sum_i n_{i0} + n_{i1}} = \frac{\sum_i \frac{n_{i0}n_{i1}}{n_{i0}+n_{i1}}}{\sum_i n_{i0} + n_{i1}} \ .\end{aligned} \tag{6}$$

Then we can set $b_v = a_v\varepsilon$ to encode that every worker has already annotated a_v items, with $a_v\varepsilon$ mistakes, i.e. an error rate ε.

However, we find that the error rate is underestimated by Equation (6) in practice. We then derive that the error rate given by Equation (6) has an upper bound 1/4:

$$\begin{aligned}\varepsilon =& \frac{\sum_i \frac{1}{n_{i0}+n_{i1}} n_{i0}n_{i1}}{\sum_i n_{i0} + n_{i1}} \\ \le& \frac{\sum_i \frac{1}{n_{i0}+n_{i1}} \left(\frac{n_{i0}+n_{i1}}{2}\right)^2}{\sum_i n_{i0} + n_{i1}} = \frac{\sum_i \frac{n_{i0}+n_{i1}}{4}}{\sum_i n_{i0} + n_{i1}} = \frac{1}{4} \ .\end{aligned}$$

So given $\varepsilon \in [0, 1/4]$, setting $b_v = a_v\varepsilon$ means our prior belief is that every worker has an accuracy of 75% or higher even in the worst case, which is too optimistic and often doesn't reflect reality. We then propose doubling ε to extend its range to $[0, 1/2]$ to encode our belief that every worker is performing slightly better than random guessing.

For multi-class tasks, as there are K classes, we expect random guessing can achieve an accuracy of $1/K$, i.e. an error rate of $1 - \frac{1}{K}$. To encode that the prior belief that every worker is better than random guessing, we extend the range of ε to $[0, 1 - \frac{1}{K}]$ by multiplying ε with $4(1 - \frac{1}{K})$. This is consistent with the doubling strategy for binary tasks, as $4(1 - \frac{1}{K}) = 2$ when $K = 2$.

4 DISCUSSION

The primary motivation of modelling the truth Z as continuous variables is to make the optimization of finding the most likely Z tractable, i.e. $\arg\max_Z P(Z|Y,\alpha)$. We first describe the difference between this objective and other probabilistic models' objectives, then discuss some interesting findings due to the continuous modelling of Z.

4.1 An Objective Intractable for Other Models

Consider a graphical model defining joint $P(Z,Y,V|\alpha)$. It is often easy to sum out Z or integrate out V in $P(Z,Y,V|\alpha)$ to get $P(Y,V|\alpha)$ or $P(Z,Y|\alpha)$. Ideally, the most likely Z is computed by

$$Z^* = \arg\max_Z P(Z|Y,\alpha), \text{ or equivalently } Z^* = \arg\max_Z P(Z,Y|\alpha),$$

because $P(Z,Y|\alpha) \propto P(Z|Y,\alpha)$. However, when Z is discrete, as in most existing probabilistic models for truth inference, the above objective becomes an intractable discrete optimization task. Typical workarounds are:

(1) Solve $V^* = \arg\max_V P(Y,V|\alpha)$ first, then set $V = V^*$ into the joint distribution which naturally decouples z_i's, so $Z^* = \arg\max_Z P(Y,Z,V^*|\alpha)$ becomes tractable as z_i's can be optimized independently.

$$Z^* = \arg\max_Z P(Y,Z,V^*|\alpha)$$

where $V^* = \arg\max_V P(Y,V|\alpha)$. The compromise here is that although V^* is optimal, Z^* is only optimal when $V = V^*$, which does not capture uncertainty. Examples of methods taking this approach include DS [3] and LFC [18].

(2) Bayesian methods naturally use the posterior to make inference. In $P(Z,Y,V|\alpha)$, only Z is the variable of interest, so V is marginalized first, leaving $P(Z,Y|\alpha)$. The posterior of Z, $P(Z|Y,\alpha)$ is proportional to $P(Z,Y|\alpha)$, however, the normalizing constant $P(Y|\alpha)$ is typically intractable to compute. There are two common ways to deal with this intractability.
   - *Variational approaches* [8] approximate the posterior $P(Z|Y,\alpha)$ by another distribution $q(Z)$, which is then used to make inference. Mean-field variational inference is very popular and assumes $q(Z)$ is factorized into $\prod_i q(z_i)$, then the true label of item i is estimated as $\arg\max_{z_i} q(z_i)$, which approximates $\arg\max_{z_i} P(z_i|Y,\alpha)$ where $P(z_i|Y,\alpha)$ is the marginal distribution of z_i.
   - *Sampling techniques* generate samples from $P(Z|Y,\alpha)$. Markov chain Monte Carlo methods are commonly used to generate samples as they can work without knowing the normalizing constant $P(Y|\alpha)$. Then samples are used to approximate the marginal distribution over z_i, i.e. $P(z_i|Y,\alpha)$. Finally, $\arg\max_{z_i} P(z_i|Y,\alpha)$ is picked as the estimate of item i's true label.

   In summary, with either variational approaches or sampling, the goal of Bayesian methods is to solve

$$z_i^* = \arg\max_{z_i} P(z_i|Y,\alpha).$$

However, both approaches face the shortcoming that the variables of interest (z_i) must be optimized independently according to their marginal distributions, as joint optimization is intractable as Z is discrete. Examples include iBCC [10] and CBCC [22].

By contrast, our methods' objectives are ideal:

$$Z^* = \arg\max_Z P(Z|Y,\alpha).$$

The only compromise we make is that we have to model Z as continuous random variables.

4.2 A Novel Perspective on the Wisdom of Crowds

For our proposed BWA model, the negative log likelihood $-\log P(Z,Y|\alpha)$ defined in Equation (3) is reminiscent of a traditional sum-of-square-error objective as minimized in regression models, except that here the error terms are contained in separate log functions. Reorganizing the expression, and using the sum of squared errors defined as $\text{SSE}_j = \sum_{i\in N_j}(z_i - y_{ij})^2$, we obtain

$$\begin{aligned}&-\log P(Z,Y|\mu,\lambda,a_v,b_v)\\ &=\frac{\lambda}{2}\sum_i(z_i-\mu)^2 + \frac{1}{2}\sum_j \log\left(\frac{b_v+\text{SSE}_j}{a_v+|N_j|}\right)^{a_v+|N_j|} + const.\end{aligned}$$

The exponential of the RHS yields the equivalent objective

$$\begin{aligned}&\arg\min_{Z,\mu} -\log P(Z,Y|\mu,\lambda,a_v,b_v)\\ &=\arg\min_{Z,\mu}\prod_{\text{item } i} e^{\lambda(z_i-\mu)^2}\prod_{\text{worker } j}\left(\frac{b_v+\text{SSE}_j}{a_v+|N_j|}\right)^{a_v+|N_j|}.\end{aligned}$$

The first product over items serves as a regularization term for z_i. For μ, it simply means the minimum can be reached when μ is the mean of all z_i. The product over workers is more interesting. Note that $\frac{b_v+\text{SSE}_j}{a_v+|N_j|}$ also appears in $\mathbb{E}_q v_j$'s update rule, and can be considered as the smoothed error rate of worker j. The exponent, $a_v+|N_j|$, is just the smoothed number of items worker j has labelled. Accordingly, this factor measures the likelihood that all workers make all their annotations of the data incorrectly, with a uniform error rate per worker.

Therefore, the underlying principle behind our proposed BWA model is that *it's unlikely that all workers make mistakes for all their tasks*. Our proposed BWA model infers a truth that minimizes the likelihood of everybody performing incorrectly. This principle makes our model different from other models which seek to explain worker labels through maximising the likelihood of realizing these observations. Our principle is conservative, yet intuitively plausible, and a novel perspective on the *wisdom of crowds*.

4.3 A Novel Constraint on Worker Reliability

In the base model, if we were to force all y_{ij} and z_i to be binary variables ($\{0,1\}$) and then re-normalize the distributions accordingly, we obtain a new link function between y_{ij} and z_i,

$$\begin{aligned}P(y_{ij}=1|z_i,v_j) &= \frac{\sqrt{\frac{v_j}{2\pi}}\exp\left\{-\frac{v_j}{2}(z_i-1)^2\right\}}{\sqrt{\frac{v_j}{2\pi}}\exp\left\{-\frac{v_j}{2}z_i^2\right\}+\sqrt{\frac{v_j}{2\pi}}\exp\left\{-\frac{v_j}{2}(z_i-1)^2\right\}}\\ &= \frac{1}{\exp\left\{-v_j(z_i-\frac{1}{2})\right\}+1} \text{ for } z_i\in\{0,1\},\end{aligned}$$

or in an equivalent but more compact form,

$$P(y_{ij}|z_i, v_j) = \frac{\exp\left\{\frac{1}{2}v_j \mathbf{1}[y_{ij} = z_i]\right\}}{1 + \exp\left\{\frac{1}{2}v_j\right\}},$$

which means each worker has an accuracy of $\sqrt{e^{v_j}}/(1 + \sqrt{e^{v_j}})$, irrespective of the true label z_i. Such behaviour can be captured by a symmetric confusion matrix

$$\frac{1}{1 + \sqrt{e^{v_j}}} \begin{bmatrix} \sqrt{e^{v_j}} & 1 \\ 1 & \sqrt{e^{v_j}} \end{bmatrix}.$$

Due to v_j having a Gamma prior, v_j must be non-negative. Consequently the values on the diagonal must be larger than or equal to the values off diagonal. Accordingly our approach is capable of modelling: a perfect worker, $\left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$, a typical worker, $\left[\begin{smallmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{smallmatrix}\right]$, and a random worker, $\left[\begin{smallmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{smallmatrix}\right]$. However adversarial workers, e.g., $\left[\begin{smallmatrix} 0.2 & 0.8 \\ 0.8 & 0.2 \end{smallmatrix}\right]$, cannot be represented. This limitation arises from the choice of prior (Gamma), and does not feature in other confusion matrix based models.

This constraint means our models can't capture adversarial workers, however this limitation fits with empirical observations of crowd-workers: in practice adversarial workers are rare, and are largely filtered out based on performance history, which is a consequence of the data acquisition method, but it's possible that less carefully collected data with less rigorous quality control may not be a good fit to our model. Typically in a confusion matrix based probabilistic model, worker labels are assumed to be drawn from categorical distributions parameterized by rows in confusion matrices, and every row in a confusion matrix is assumed to be drawn from a Beta or Dirichlet distribution. For binary tasks, the constraint that values on the diagonal are larger than or equal to the values off the diagonal could be achieved by using a truncated Beta prior with a support $[0.5, 1]$, but at a much higher computational cost because the distributions will no longer be conjugate.

5 EXPERIMENTS

Initialization. Our techniques use the EM algorithm, and thus require initialization of either μ and $\mathbb{E}_q V$, or Z. We initialise Z, the easier of the two, using majority voting which allows for fast convergence. In some cases, from the MV initialization our algorithm doesn't converge to the global optimum (the highest likelihood). A common solution is to run the algorithm with random initialization multiple times and then find the best likelihood (including a majority voting run), but to obtain deterministic results, we only run our algorithm with the MV initialization in all experiments.

Since we use the released implementations of existing methods, we choose to keep their code unchanged. Implementations initializing parameters randomly (CATD, CRH) are run 10 times on every dataset and the mean accuracy of 10 runs is reported as its accuracy on that dataset.

Hyperparameter settings. Following the suggestions described in Section 3.1.3, we set $\lambda = 1$. We choose a_v and b_v in Section 5.1. We set the tolerance in the stopping criteria to 10^{-3}, such that iteration of the algorithm stops when the relative difference between every z_i and its last value is within 0.1%.

Methods. Zheng et al. [28][2] compared 17 existing aggregation methods and released their implementations, and 10 of them supporting the multi-class setting are used in our experiment including MV, ZC [4], GLAD [26], DS [3], Minimax [29], iBCC [10], CBCC [22], LFC [18], CATD [12], and CRH [1].

Datasets. There are 19 real-world datasets used in the experiments originating with four crowdsourcing dataset collections, namely the union of the CrowdScale 2013's shared task challenge [9][3] (2 datasets), the Active Crowd Toolkit project [23][4] (8 datasets), the Truth Inference in Crowdsourcing project [28][5] (7 datasets), and the GitHub repository for SpectralMethodsMeetEM paper [27][6] (5 datasets). Note that 3 datasets are in common between the last two collections. Table 2 presents some statistics about the 19 datasets.

The first two, `senti` and `fact`, are from the CrowdScale 2013's shared task challenge, and are the largest two in our experiment, with 569k and 217k labels respectively. The `senti` data asks the workers to judge the sentiment of a tweet discussing weather and label it within {Negative, Neutral, Positive, Unrelated, Can't tell}. The `fact` data requests workers to judge whether relational statements about public figures are correct, e.g. "Stephen Hawking graduated from Oxford". Workers may choose from {Yes, No, Skip}. Note that "Can't tell" or "Skip" may also appear as the true labels of some items due to the inherent ambiguity of some questions. None of the methods compared in our experiments explicitly model the unsure options, instead treating these as a standard label in a 5 and 3-class setting, respectively.

The next 8 datasets are all from the Active Crowd Toolkit project. `MS`, collected by Rodrigues et al. [19], asks workers to listen to short music samples and to classify the music into 10 music genres, {country, jazz, disco, pop, reggae, rock, blues, classical, mental, hip-hop}. `ZC_in` and `ZC_us`, provided by Demartini et al. [4], ask workers to judge if a provided Uniform Resource Identifier (URI) is relevant to a named entity extracted from news, where every URI describes an entity. `ZC_all` is a combination of `ZC_in` and `ZC_us`, and all `ZC` datasets share the same set of items. The two `SP`s are movie review datasets with binary sentiment labels. `CF` is just a small part of the full `senti`. `CF`* and `SP`* have the same items as `CF` and `SP`, but were reannotated.

The next 5 datasets are from the Truth Inference project Zheng et al. [28]. `prod`, collected by Wang et al. [24] for entity resolution, asks workers whether two product descriptions refer to the same product or not. `tweet`, collected by Zheng et al. [28], contains tweets where every tweet is related to a company. Workers are asked to identify if the tweet has positive sentiment towards the company in the tweet. `dog` contains dog images from ImageNet [11] and the task is to recognize a breed (out of Norfolk Terrier, Norwich Terrier, Irish Wolfhound, and Scottish Deerhound) for a given dog. `face`, provided by Mozafari et al. [15], contains face images, and workers are asked to judge whether the person in the image is happy, sad, angry, or neutral. `adult`, released by Mason and Suri [13], asks workers to determine the age appropriateness (P, PG, R, X) of a website given its link.

[2] https://zhydhkcws.github.io/crowd_truth_inference/index.html
[3] https://sites.google.com/site/crowdscale2013/shared-task
[4] https://github.com/orchidproject/active-crowd-toolkit
[5] See footnote 1.
[6] https://github.com/zhangyuc/SpectralMethodsMeetEM

Table 2: Statistics about 19 real-world datasets used in our experiments.

Dataset	$\#item(N)$	$\#worker(W)$	$\#class$	$\#label$	$\#truth$	$\#label/N$	$\#label/W$
senti	98980	1960	5	569282	1000	5.75	290.45
fact	42624	57	3	214960	576	5.04	3771.23
CF	300	461	5	1720	300	5.73	3.73
CF*	300	110	5	6025	300	20.08	54.77
MS	700	44	10	2945	700	4.21	66.93
SP	4999	203	2	27746	4999	5.55	136.68
SP*	500	143	2	10000	500	20	69.93
ZCall	2040	78	2	20372	2040	9.99	261.18
ZCin	2040	25	2	10626	2040	5.21	425.04
ZCus	2040	74	2	11271	2040	5.53	152.31
prod	8315	176	2	24945	8315	3	141.73
tweet	1000	85	2	20000	1000	20	235.29
dog	807	109	4	8070	807	10	74.04
face	584	27	4	5242	584	8.98	194.15
adult	11040	825	4	89948	333	8.15	109.03
bird	108	39	2	4212	108	39	108
trec	19033	762	2	88385	2275	4.64	115.99
web	2665	177	5	15567	2653	5.84	87.95
rte	800	164	2	8000	800	10	48.78

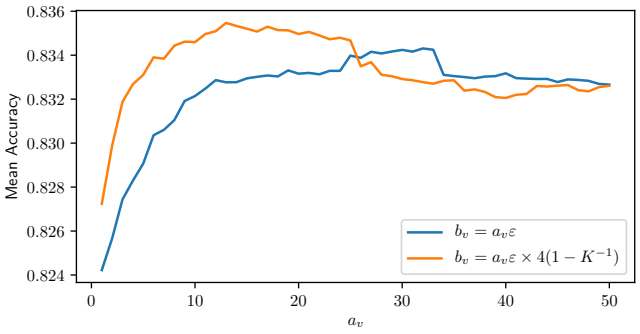


Figure 2: The performance of our proposed BWA model with b_v set by two different strategies.

The last five datasets are used in Zhang et al. [27]. bird, provided by Welinder et al. [25], is a dataset of bird images. The task is to determine whether an image contains at least one duck. The remaining three cover: recognising textual entailment, rte [21], assessing the quality of retrieved documents, trec (TREC 2011 crowdsourcing track)[7], judging the relevance of web search results, web [29].

5.1 Choosing a_v and b_v

Figure 2 shows the mean accuracy of our proposed BWA model across 19 datasets with b_v set by two different strategies described in Section 3.3, namely, using the original error rate ε and using the adjusted error rate $\varepsilon \cdot 4(1 - \frac{1}{K})$. For both strategies, the performance increases steeply when $a_v < 10$ as the prior becomes stronger and provides regularisation to workers who have labelled very few items. We also see both curves are flat around the optimum, so the performance is not very sensitive to the choice of a_v. Due to this reason, we pick a_v from the flat region and report results in both settings: BWA($a_v = 30$, original ε) and BWA($a_v = 15$, adjusted ε).

[7] https://sites.google.com/site/treccrowd/2011

5.2 Comparison against MV

In this section, we report the results of one-sided Wilcoxon signed rank test performed on every method against MV. We adopt the original version which discards the observations when two methods achieve the same accuracy on a dataset. The null hypothesis H_0 is that the performance difference between a method and MV follows a symmetric distribution around zero.

Table 3 summarizes the results showing whether every method is significantly better than MV and at which level, where significance levelsare denoted by asterisks, i.e. 0.1(*), 0.05(**), 0.025(***), respectively. N_r denotes the effective sample size, and W_- the statistic of a method summing up the ranks of datasets that this method is inferior to MV. Zencrowd and iBCC achieve 85.33% accuracy on the CF* dataset which is exactly the same performance as MV resulting, therefore their N_r's are reduced to 18.

The rightmost column in Table 3 shows the p-value calculated based on approximating the cumulative density function (cdf) of W_- given H_0 as a cdf of Gaussian. Due to the small N_r in our experiment (18 or 19), the approximation is somewhat inaccurate, but we keep them there as a reference, as they show ranking of the improvement achieved by all methods comparing to MV.

5.3 Discussion

Scalability. To measure the efficiency of all methods, we run all experiments on a modest desktop with a 7-th generation Intel i5 CPU and 8 GB memory. All methods including our proposed BWA model are implemented in Python, except for iBCC and CBCC

Table 3: One-sided Wilcoxon signed rank test results (all methods against MV)

Method	Mean accuracy	N_r	W_-	Sig. level	Approx. p-value
MV	0.8196				
ZC	0.8294	18	64		0.1746
GLAD	0.8305	19	47	**	0.0267
DS	0.8309	19	71		0.1671
Minimax	0.8141	19	81		0.2866
iBCC	0.8352	18	52	*	0.0723
CBCC	0.8291	19	71		0.1671
LFC	0.8338	19	69		0.1477
CATD	0.8334	19	56	*	0.0583
CRH	0.8054	19	91		0.4361
BWA($a_v = 30$, original ε)	0.8342	18	38	***	0.0193
BWA($a_v = 15$, adjusted ε)	0.8352	18	39	***	0.0214

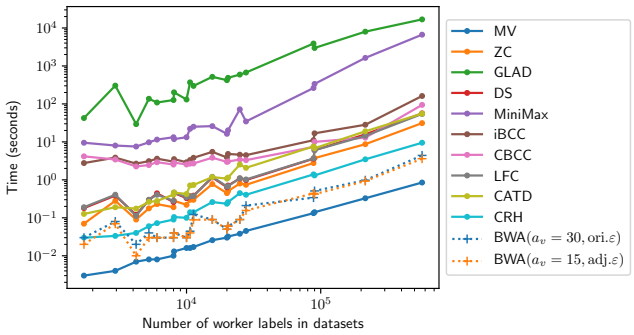


Figure 3: Running time (seconds) of all methods on 19 real-world datasets.

which are implemented in C#, and Minimax is implemented in MATLAB.

As shown in Figure 3, our proposed methods outperform the other existing methods (except MV) in terms of running time. For datasets other than the largest two (senti and fact), our method converged within one second. GLAD and Minimax are the two slowest methods, as they both require gradient-based optimization algorithms, and the required gradients are expensive to compute.

Accuracy. As shown in Table 3, although iBCC has the highest mean accuracy across all datasets, it outperforms MV only at a significance level of 0.1, which is insufficient to reject the null hypothesis. On the other hand, GLAD, whose mean accuracy is not outstanding, significantly outperforms MV at a significance level of 0.05, the best among all 9 benchmarks. In contrast, BWA achieves not only the highest mean accuracy, the same as iBCC, but also strongest significance of outperforming MV, at a level of 0.025.

As shown in Table 3, our proposed BWA($a_v = 15$, adjusted ε) model and iBCC achieve the highest mean accuracy of 83.52%, and our BWA model using another hyper-parameter setting ($a_v = 30$, original ε) which achieves 83.42%, followed by LFC 83.38% and CATD 83.34%. Interestingly, both our proposed methods and CATD model the latent truth z_i's as continuous variables, but in CATD the optimization objective is heuristic based, while in our models, the optimization objective is derived from explicit probabilistic models. Minimax is an interesting model in that it largely outperforms others on bird and web but performs the worst on MS, the three ZC datasets, and prod. This may be due to Minimax not being a probabilistic model thus its objective function is not well regularised and often too aggressive. This may also explain the results for CRH, another non-probabilistic model.

Figure 4 provides more details of the accuracies of all methods on the 19 real-world datasets. We roughly sort all methods into two categories: confusion-matrix-based methods (DS, MiniMax, iBCC, CBCC, LFC) and 1-coin models (ZC, GLAD, CATD, CRH, our proposed BWA). The name "1-coin" arises from these models only learning a single parameter per worker to capture their accuracy. face, bird, trec are three successful cases for confusion-matrix-based methods, because on those datasets workers behave very differently across classes, which "1-coin" models cannot capture. However, they perform not so well on senti, fact, MS, and ZC datasets due to either the data is overall insufficient to learn a $K \times K$ confusion matrix for every worker, or the class is imbalanced that for some minority classes worker behaviour can't be reliably estimated.

6 CONCLUSION

This paper presents a novel method for aggregating crowdsourced labels for classification. We propose a generative Bayesian model, relaxing a discrete binary problem to a continuous regression task, which we model using a normal likelihood and conjugate inverse-Gamma prior, and can be extended to work in multi-class settings by employing the one-versus-rest strategy.

Our models allow for straightforward inference using expectation-maximisation, which is implemented in 50 lines of code in the appendix. Our models have interpretable parameters and intuitive update rules, which make them easy to understand and use. We also provide detailed practical suggestions such as how to set the hyper-parameters.

Our experimental results show that our proposed method is the only one that significantly outperforms MV at significance level 0.025 and performs competitively to the state-of-the-art method in terms of mean accuracy, across all datasets drawn from several different application areas, with different sizes and differing degrees of label redundancy. Our proposed method also has a total running time on the same datasets second only to MV.

ACKNOWLEDGEMENT

This work was sponsored by the Defense Advanced Research Projects Agency Information Innovation Office (I2O) under the Low Resource Languages for Emergent Incidents (LORELEI) program issued by DARPA/I2O under Contract No. HR0011-15-C-0114. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government. Trevor Cohn and Benjamin Rubinstien were supported by the Australian Research Council, FT130101105 and DP150103710, respectively.

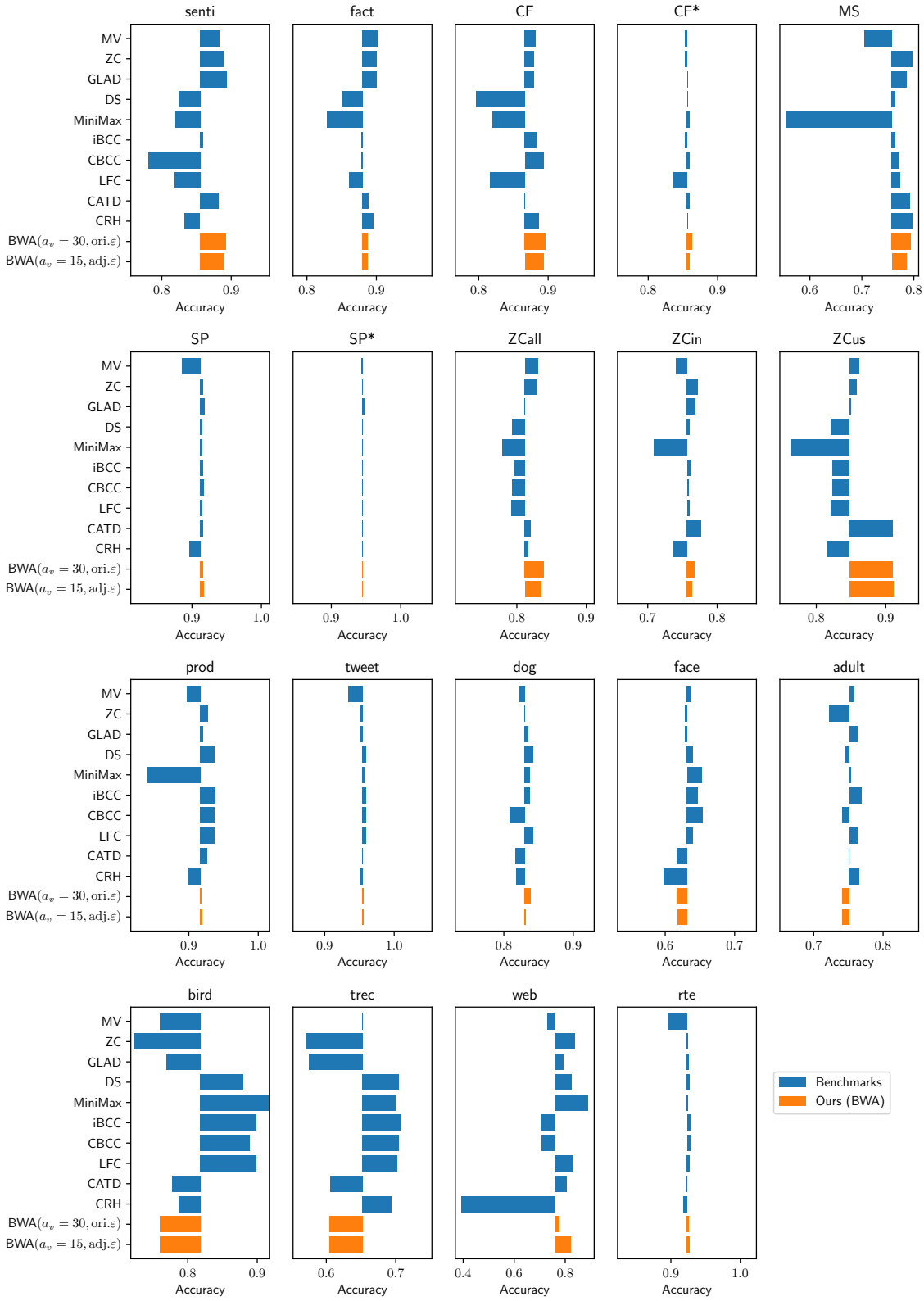


Figure 4: Accuracies of all methods on 19 real-world datasets. The horizontal baseline is the mean accuracy across all methods.

A CODE

```python
import numpy as np
import pandas as pd
import scipy.sparse as ssp

def bwa_binary(y_exists_ij, y_is_one_ij, W_i, lambda_, a_v, adj_coef):
    N_j = y_exists_ij.sum(axis=0)
    z_i = y_is_one_ij.sum(axis=-1) / y_exists_ij.sum(axis=-1)

    b_v = a_v * W_i.dot(np.multiply(z_i, 1-z_i)) / y_exists_ij.sum() * adj_coef
    for _ in range(500):
        last_z_i = z_i.copy()

        mu  = z_i.mean()
        v_j = (a_v+N_j) / (b_v + (y_exists_ij.multiply(z_i)-y_is_one_ij).power(2).sum(0))
        z_i = (lambda_*mu + y_is_one_ij.dot(v_j.T)) / (lambda_ + y_exists_ij.dot(v_j.T))

        if np.allclose(last_z_i, z_i, rtol=1e-3): break
    return z_i.A1

def bwa(tuples, a_v=30, lambda_=1, prior_correction=True):
    num_items, num_workers, num_classes = tuples.max(axis=0) + 1
    num_labels = tuples.shape[0]
    W_i = np.bincount(tuples[:, 0])

    adj_coef = 4 * (1 - 1 / num_classes) if prior_correction else 1

    y_exists_ij = ssp.coo_matrix((np.ones(num_labels), tuples[:, :2].T),
                                 shape=(num_items, num_workers), dtype=np.bool).tocsr()
    y_is_one_kij = []
    for k in range(num_classes):
        selected = (tuples[:, 2] == k)
        y_is_one_kij.append(ssp.coo_matrix((np.ones(selected.sum()), tuples[selected, :2].T),
                                           shape=(num_items, num_workers), dtype=np.bool).tocsr())
    z_ik = np.empty((num_items, num_classes))
    for k in range(num_classes):
        z_ik[:, k] = bwa_binary(y_exists_ij, y_is_one_kij[k], W_i, lambda_, a_v, adj_coef)
    return z_ik

def get_acc(predictions, df_truth):
    score = (predictions == predictions.max(axis=1, keepdims=True)).astype(np.float)
    score /= score.sum(axis=1, keepdims=True)
    return score[df_truth.item.values, df_truth.truth.values].sum() / df_truth.shape[0]

data_path = '....'
datasets = ['crowdscale2013/sentiment', 'crowdscale2013/fact_eval']
records = []
for dataset in datasets:
    df_label = pd.read_csv(data_path + dataset + '/label.csv')
    df_label = df_label.drop_duplicates(keep='first')
    prediction_ik = bwa(df_label.values)

    df_truth = pd.read_csv(data_path + dataset + '/truth.csv')
    records.append((dataset, get_acc(prediction_ik, df_truth)))
print(pd.DataFrame.from_records(records, columns=['dataset', 'accuracy']))
## example output
#                     dataset  accuracy
# 0  crowdscale2013/sentiment  0.888000
# 1  crowdscale2013/fact_eval  0.885417

## label.csv example: | truth.csv example:
#                     |
#  item,worker,label  | item,truth
#  26440,1707,3       | 110,0
#  26440,311,3        | 161,1
#  26440,1356,3       | 328,1
#  26440,1774,3       | 611,2
#  ...                | ...
```

REFERENCES

[1] Bahadir Ismail Aydin, Yavuz Selim Yilmaz, Yaliang Li, Qi Li, Jing Gao, and Murat Demirbas. 2014. Crowdsourcing for Multiple-Choice Question Answering.

[2] Chris Callison-Burch and Mark Dredze. 2010. Creating speech and language data with Amazon's Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. Association for Computational Linguistics, 1–12.

[3] Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics* (1979), 20–28.

[4] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. 2012. ZenCrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 469–478.

[5] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)* (1977), 1–38.

[6] Djellel E. Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. 2012. Mechanical Cheat: Spamming Schemes and Adversarial Techniques on Crowdsourcing Platforms. *CrowdSearch 2012 workshop at WWW* (2012), 26–30.

[7] Jeff Howe. 2008. *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business* (1 ed.). Crown Publishing Group, New York, NY, USA.

[8] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. 1999. An introduction to variational methods for graphical models. *Machine learning* 37, 2 (1999), 183–233.

[9] Tatiana Josephy, Matt Lease, Praveen Paritosh, Markus Krause, Mihai Georgescu, Michael Tjalve, and Daniela Braga. 2014. Workshops held at the first AAAI conference on human computation and crowdsourcing: A report. *AI Magazine* 35, 2 (2014), 75–78.

[10] Hyun-Chul Kim and Zoubin Ghahramani. 2012. Bayesian classifier combination. In *Artificial Intelligence and Statistics*. 619–627.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[12] Qi Li, Yaliang Li, Jing Gao, Lu Su, Bo Zhao, Murat Demirbas, Wei Fan, and Jiawei Han. 2014. A confidence-aware approach for truth discovery on long-tail data. *Proceedings of the VLDB Endowment* 8, 4 (2014), 425–436.

[13] Winter Mason and Siddharth Suri. 2012. Conducting behavioral research on Amazon's Mechanical Turk. *Behavior research methods* 44, 1 (2012), 1–23.

[14] Pablo G Moreno, Antonio Artés-Rodríguez, Yee Whye Teh, and Fernando Perez-Cruz. 2015. Bayesian nonparametric crowdsourcing. *Journal of Machine Learning Research* (2015).

[15] Barzan Mozafari, Purna Sarkar, Michael Franklin, Michael Jordan, and Samuel Madden. 2014. Scaling Up Crowd-sourcing to Very Large Datasets: A Case for Active Learning. *Proc. VLDB Endow.* 8, 2 (Oct. 2014), 125–136. https://doi.org/10.14778/2735471.2735474

[16] Scott Novotney and Chris Callison-Burch. 2010. Cheap, fast and good enough: Automatic speech recognition with non-expert transcription. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 207–215.

[17] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. In *EMNLP*. The Association for Computational Linguistics, 2383–2392.

[18] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *Journal of Machine Learning Research* 11, Apr (2010), 1297–1322.

[19] Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2013. Learning from multiple annotators: distinguishing good from random labelers. *Pattern Recognition Letters* 34, 12 (2013), 1428–1436.

[20] Edwin Simpson, Stephen Roberts, Ioannis Psorakis, and Arfon Smith. 2013. Dynamic bayesian combination of multiple imperfect classifiers. In *Decision making and imperfection*. Springer, 1–35.

[21] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and Fast—but is It Good?: Evaluating Non-expert Annotations for Natural Language Tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 254–263.

[22] Matteo Venanzi, John Guiver, Gabriella Kazai, Pushmeet Kohli, and Milad Shokouhi. 2014. Community-based bayesian aggregation models for crowdsourcing. In *Proceedings of the 23rd international conference on World wide web*. ACM, 155–164.

[23] Matteo Venanzi, Oliver Parson, Alex Rogers, and Nick Jennings. 2015. The ActiveCrowdToolkit: An open-source tool for benchmarking active learning algorithms for crowdsourcing research. In *Third AAAI Conference on Human Computation and Crowdsourcing*.

[24] Jiannan Wang, Tim Kraska, Michael J Franklin, and Jianhua Feng. 2012. Crowder: Crowdsourcing entity resolution. *Proceedings of the VLDB Endowment* 5, 11 (2012), 1483–1494.

[25] Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. 2010. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*. 2424–2432.

[26] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*. 2035–2043.

[27] Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I Jordan. 2014. Spectral methods meet EM: A provably optimal algorithm for crowdsourcing. In *Advances in neural information processing systems*. 1260–1268.

[28] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. 2017. Truth inference in crowdsourcing: is the problem solved? *Proceedings of the VLDB Endowment* 10, 5 (2017), 541–552.

[29] Denny Zhou, Sumit Basu, Yi Mao, and John C Platt. 2012. Learning from the wisdom of crowds by minimax entropy. In *Advances in neural information processing systems*. 2195–2203.

Unsupervised Semantic Generative Adversarial Networks for Expert Retrieval

Shangsong Liang
School of Data and Computer Science, Sun Yat-sen University, China
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
liangshangsong@gmail.com

ABSTRACT

Sources in computer-based collaborative systems such as webpages can help employees to connect and cooperate with each other. It is natural to enable the systems to look not only for documents but also for experts. In this paper, we study the problem of expert retrieval in enterprise corpora: given a topic, also known as query containing a set of words, identify a rank list of candidate experts who have expertise on the topic. To tackle the problem, we propose an unsupervised semantic two-player minimax game, i.e., our unsupervised semantic generative adversarial networks (USGAN). Unlike almost all the previous generative adversarial networks-based algorithms that require ground truth training data, our USGAN is an unsupervised semantic expert retrieval algorithm that consists of a discriminative network and a generative network aiming at capturing the representations of words and experts in an unsupervised way. Candidates that have similar semantic representations to that of the topic are retrieved as relevant to the topic. Our USGAN would provide inspiration on how to extend the standard GAN and its variants by unsupervised ways to address other retrieval tasks where labelled data are missing. Experimental results on public datasets validate the effectiveness of the proposed expert retrieval algorithm.

CCS CONCEPTS

• **Information systems → Expert search**.

KEYWORDS

Expert Retrieval; Language Models; Generative Adversarial Networks

ACM Reference Format:
Shangsong Liang. 2019. Unsupervised Semantic Generative Adversarial Networks for Expert Retrieval. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313625

1 INTRODUCTION

Computer-based collaborative systems have been widely used by many organizations these days. Sources in such systems, such as webpages, database records, agendas, employees' description files and technical reports, can help to connect employees for collaboration and information exchange in the organization. It is natural to enable the systems to look not only for documents, but for entities, such as answers, services, objects, and experts. Our interest is in one type of entity in particular: experts. Accordingly, in this paper, we aim at addressing the *expert retrieval task, also known as expert finding, expert retrieval, or expert search task: given a topic, also called a query containing a set of words, who are the experts with the most expertise on the topic?* Our goal is to retrieve a rank list of experts who have expertise on a specific topic described by an input query, given a set of candidates and a heterogeneous document repository of an organization.

The study of expert retrieval task has gained popularity in research community since the launch of the task at TREC 2005 enterprise track [14]. Algorithms for the task are either supervised or unsupervised. Supervised ones require great manual annotation efforts to obtain training data, resulting in the fact that a limited amount of training data may greatly hinder their applications. In contrast, unsupervised ones, such as those based on language models [8], do not require the training data, and can work well in many cases. In this paper we aim at improving the performance with no training data, and propose **U**nsupervised **S**emantic **G**enerative **A**dversarial **N**etworks (**USGAN** for short), i.e., an unsupervised semantic minimax game, for the expert retrieval task.

GAN (Generative Adversarial Networks [20]) works with a minimax game consisting of a discriminative network D that learns to distinguish whether a given data instance is real or not, and a generative network G that learns to confuse D by generating high quality data. GAN and its variants have been successfully applied into many applications such as those in artificial intelligence, e.g., abstract reasoning [28], clustering [42], and image and sequence generation [27, 57], and have shown their better performance compared to that of many maximum likelihood techniques and traditional deep learning models, e.g., convolutional neural networks [19]. However, they can not directly be applied to our task without labelled data, as they are supervised. Our USGAN is able to learn from raw textual evidence and document-expert associations in an unsupervised way and thus does not require any manual relevance judgements between topics and experts.

Additionally, most unsupervised expert retrieval algorithms are based on language models [15]. However, these unsupervised language model-based algorithms need to construct a language model for every document in the collections, and thus lack efficient query capabilities for large document collections, as each query term needs to be exactly matched against every document. They also suffer from term mismatch problem, which occurs due to the inability of widely used maximum-likelihood functions in the algorithms

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313625

and the different representations of queries and expert documents to describe the same concepts [50]. To address the problems, we represent all the words and candidate experts as semantic vectors obtained from our unsupervised USGAN algorithm such that we can avoid constructing language models for all documents in the collections and the the semantic similarities between words and experts can be effectively measured.

The purposes of this work are not only for improving the performance of unsupervised expert retrieval algorithms, but also, more importantly, for providing a new insight into training GAN and its variants, i.e., via unsupervised learning for a number of information retrieval applications. Our contributions can be summarized as:

(1) We propose unsupervised generative adversarial network algorithm that is a minimax game consisting of a discriminative network and a generative network for expert retrieval task.

(2) Our USGAN is semantic and is able to semantically measure the similarities between the representations of words and experts.

(3) To our knowledge, we are the first attempt to apply GAN framework *in an unsupervised way* to tackle the retrieval problems in information retrieval.

(4) Through USGAN as an example, we would provide insight and inspiration on how to extend supervised GAN model and its variants in unsupervised ways to tackle other tasks where labelled data is limited or missing.

(5) We systematically analyze the proposed algorithms and find that we achieve better performance compared to the state-of-the-art unsupervised algorithms.

The remainder of the paper is organized as: Section 2 discusses related work; Section 3 details the research problem; Section 4 describes the proposed USGAN model; Section 5 describes our experimental setup; Section 6 is devoted to our experimental results and we conclude the paper in Section 7.

2 RELATED WORK

In this section, we only discuss the most related work: previous generative adversarial networks, expert retrieval algorithms, and semantic models.

2.1 Generative Adversarial Networks

Generative adversarial networks are a class of GAN-based methods for learning from labelled data based on minimax game theory [20] that consists of a generative network G and a discriminative network D. The goal of GAN-based methods is to train a generator network $G(\boldsymbol{z}; \boldsymbol{\theta}^{(G)})$ that produces samples from the data distribution, $p_{\text{data}}(\boldsymbol{x})$, by transforming vectors of noise $\boldsymbol{z}$ as $\boldsymbol{x} = G(\boldsymbol{z}; \boldsymbol{\theta}^{(G)})$. The training signal for G is provided by a discriminator network D that is trained to distinguish between samples from the generator distribution and those from ground truth (real) data. The generator network G in turn is then trained to fool the discriminator into accepting its outputs as being real. After the first GAN model is introduced, many of its recent variants have been proposed and successfully applied to many applications. E.g., Boundary equilibrium GAN is proposed in [11], where a novel equilibrium method is applied for balancing adversarial networks and auto-encoder is applied as the discriminator. Rather than proposing a new GAN-based algorithm, IRGAN [51] assumes that ground truth data is available for training and directly applies GAN to document retrieval task. Energy-based GAN [58] views the discriminator as an energy function that attributes low energies to the regions near the data manifold and higher energies to other regions. [47] introduced a specific conditioning of convolutional GAN on texture and shape elements for generating fashion design images. Some work aims at improving training techniques for GAN-based algorithms. E.g., Wasserstein GAN [2] tries to improve the stability of learning in GAN, gets rid of problems like model collapse, and provides learning curves for debugging and hyper-parameter searches. In [46], a variety of new architectural features and training procedures are proposed and applied into GAN framework. Previous work has shown that GAN-based algorithms can perform better than most traditional deep learning models such as LSTM (Long Short Term Memory networks) and CNN (Convolutional Neural Networks) [19].

Surprisingly, as far as we know little attention has paid attention to training GAN and its variants *without labelled data* for tasks in information retrieval, and the lack of training data in previous supervised GAN based algorithms would greatly hinder their success in the tasks. To the best of our knowledge, ours is the first attempt to train GAN in an unsupervised way for expert retrieval and the first attempt to apply GAN to applications in information retrieval via unsupervised ways.

2.2 Expert Retrieval

After the launch of expert retrieval task at TREC 2005 enterprise track [14], a number of supervised and unsupervised algorithms for the task have been proposed, which can be categorized into either supervised or unsupervised ones. Supervised algorithms include the relevance-based expert retrieval learning algorithm [18], learning to find experts [52]. Of special relevance to us are the unsupervised algorithms. The most well-known unsupervised ones are the profile-centric (Model 1) and document-centric (Model 2) expert finding models [4, 8], which focus on raw textual evidence without incorporating collection-specific information, e.g., query modeling, document importance or document structure. Both the Model 1 and Model 2 expert finding algorithms are language retrieval models, and they differ each other by the way of computing the probabilities of a candidate having expertise on a given topic. Other unsupervised ones include thread-based model [55] where expert finding is performed in the context of social media, language model for finding bloggers as experts [7] in the context of blog documents, and shallow log-linear model [50] that learns distributed word representations in an unsupervised way for the expert finding task, probabilistic expert finding models [17] where candidate generation framework and topic generation framework are studied for the task, and language models for finding groups of experts [32, 34] that aim at retrieving a ranked list of groups of experts given a topic. All these supervised and unsupervised methods are not based on GAN, and except the shallow log-linear one they are non-semantic.

To better provide insight in how our unsupervised semantic GAN-based algorithm improves performance of expert retrieval, we follow the setting applied in many unsupervised expert retrieval methods, i.e., avoid explicit feature engineering and the incorporation of external evidence in our USGAN expert retrieval algorithm.

To the best of our knowledge, we are the first attempt to apply GAN techniques to tackle the task of expert retrieval.

2.3 Semantic Models

The mismatches between queries and documents pose one of the most critical challenges in many information retrieval tasks [29, 30, 40, 41], and a number of topic models have been proposed to alleviate the mismatch challenge. Topic models provide a suite of semantic algorithms to discover hidden thematic structure in a collection of documents. A topic model takes a set of documents as input, and discovers a set of "latent topics"—recurring semantic themes that are discussed in the collection—and the degree to which each document exhibits those semantic topics [12]. Since the well-known topic models, PLSI (Probabilistic Latent Semantic Indexing) [22] and LDA (Latent Dirichlet Allocation) [12] were proposed, various topic models and their applications in information retrieval have been widely studied. The author topic model [45] has been proposed to uncover latent semantic topics of authors; each author is associated with a multinomial distribution over topics and each topic is associated with a multinomial distribution over words. In [53], LDA-based document models have been proposed to perform semantic matching of documents and queries. Topic models were proposed for clustering short texts [39] and users [40] in streams, streaming short text diversifications [31, 38], diversified data fusion of rank lists [31, 33, 35–37]. In [56], a comparative study of utilizing topic models for document retrieval was provided. Besides topic models, other researchers tried to alleviate the mismatches between queries and documents in other ways. For instance, in [23] semantic matching between queries and documents was performed by leveraging click-through data optimizing for web document ranking. Deep learning models, e.g., position-aware representations for relevance matching model [24], deep relevance matching model [21], shallow log-linear model [50] and multiple instance deep learning model [13], have been proposed to alleviate the mismatch problem between queries and documents.

Except the shallow log-linear model [50], all the existing semantic models are for mismatch problem between queries and documents only but not for queries and candidate experts. Unlike all of the previous models, we tackle the problem of mismatch problem among queries, documents and candidate experts via an unsupervised GAN-based semantic model. To the best of our knowledge, we are the first attempt to address the semantic mismatch problem via an unsupervised GAN method.

3 PROBLEM FORMULATION

We aim at addressing the expert retrieval task defined at the TREC 2005–2008 enterprise tracks [3, 9, 14, 48] in an unsupervised way: given a topic (a topic is also called a "query" containing a set of words in the tracks), a set of candidate experts and the heterogeneous enterprise document corpora, identify a rank list of experts who have expertise on the topic. The expert retrieval algorithm is essentially a function f that satisfies the following:

$$q, \mathcal{E}, \mathcal{D}, \xrightarrow{f} \mathcal{L},$$

where $q = \{v_1, \ldots, v_{|q|}\}$ is a query (also called topic at the tracks), $\mathcal{E}$ is a set of candidate experts, $\mathcal{D}$ is an organization's heterogeneous document corpora, and $\mathcal{L}$ is the final rank list of the experts in response to the query (experts with larger retrieval probabilities are ranked higher in the final rank list). Here v is a word from a vocabulary $\mathcal{V}$ and $|q|$ is the length of the query q.

4 METHOD

In this section, we detail our proposed **U**nsupervised **S**emantic **G**enerative **A**dversarial **N**etworks (USGAN) that aim at addressing the expert finding problem.

4.1 The Minimax Game

Our proposed USGAN for expert retrieval is an unsupervised, semantic minimax two-player game that consists of a discriminative model $D_{\boldsymbol{\phi}}$ with parameter $\boldsymbol{\phi}$ and a generative model $G_{\boldsymbol{\theta}}$ with parameter $\boldsymbol{\theta}$. The generative model would try to generate (in our task, select) relevant experts to the query that look like the ground-truth relevant experts and therefore could fool the discriminative model, whereas the discriminative model would try to draw a clear distinction between the ground-truth experts and the generated ones made by its opponent generative model. Competition in this game drives both models to improve their expert retrieval abilities until the discriminative model is indistinguishable from the experts selected from the generative model.

Without loss of generality, let's suppose we have N queries. Inspired by many GAN-based algorithms [20], we can simply define our $D_{\boldsymbol{\phi}}$ and $G_{\boldsymbol{\theta}}$ and let them play the following two-player minimax game:

$$\begin{aligned} J^{G_{\boldsymbol{\theta}^*}, D_{\boldsymbol{\phi}^*}} &= \min_{\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \sum_{n=1}^{N} \Big(\mathbb{E}_{e \sim p_{GT}(e|q_n)} [\log D_{\boldsymbol{\phi}}(e \mid q_n)] + \\ &\qquad \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z}|q_n)} \left[\log \left(1 - D_{\boldsymbol{\phi}} \left(G_{\boldsymbol{\theta}}(\mathbf{z} \mid q_n) \mid q_n \right) \right) \right] \Big), \\ &= \min_{\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \sum_{n=1}^{N} \Big(\mathbb{E}_{e \sim p_{GT}(e|q_n)} [\log D_{\boldsymbol{\phi}}(e \mid q_n)] + \\ &\qquad \mathbb{E}_{e \sim p_{\boldsymbol{\theta}}(e|q_n)} \left[\log \left(1 - D_{\boldsymbol{\phi}} \left(e \mid q_n \right) \right) \right] \Big), \end{aligned} \tag{1}$$

where e is a candidate expert, $p_{GT}(e \mid q_n)$ is the ground truth distribution of experts to the n-th query q_n, and $\mathbf{z}$ and $p_{\mathbf{z}}(\mathbf{z} \mid q_n)$ are a noise vector and its corresponding distribution, respectively. As shown in (1), we directly represent the generator $G_{\boldsymbol{\theta}}$ as $p_{\boldsymbol{\theta}}(e \mid q)$ (see the comparison between the expectation term in the second line of (1) and the expectation term in the last line of (1)), and according to many supervised discriminative expert retrieval algorithms [18], $D_{\boldsymbol{\phi}}(e_i \mid q_n)$ in (1) can be defined as:

$$D_{\boldsymbol{\phi}}(e_i \mid q_n) = p_{\boldsymbol{\phi}}(r = 1 | e_i, q_n)^{r_{in}} p_{\boldsymbol{\phi}}(r = 0 | e_i, q_n)^{1 - r_{in}}, \tag{2}$$

where $p_{\boldsymbol{\phi}}(r = 1 \mid e_i, q_n)$ and $p_{\boldsymbol{\phi}}(r = 0 \mid e_i, q_n)$ are the probabilities of candidate e_i having and not having expertise on topic q_n, respectively, which can be directly obtained by the ground truth distribution $p_{GT}(e|q_n)$, and $r_{in} \in \{0, 1\}$ is the ground truth relevance score of candidate e_i having expertise on the topic q_n with $r_{in} = 1$ indicating that e_i is relevant to (has expertise on) the query q_n and $r_{in} = 0$ indicating the opposite.

Unfortunately, without ground truth distribution $p_{GT}(e \mid q_n)$ we can not play the minimax game defined in (1) to get the optimal discriminative model $D_{\boldsymbol{\phi}^*}$ and the optimal generative model $G_{\boldsymbol{\theta}^*}$.

Accordingly, we transfer (1) to the following unsupervised two-player minimax game:

$$J^{G_{\boldsymbol{\theta}^*}, D_{\boldsymbol{\phi}^*}} = \min_{\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \sum_{n=1}^{|\mathcal{S}|} \Big(\mathbb{E}_{e \sim p_{\boldsymbol{\phi}}(e|s_n)} [\log D_{\boldsymbol{\phi}}(e \mid s_n)] + \mathbb{E}_{e \sim p_{\boldsymbol{\theta}}(e|s_n)} \Big[\log \Big(1 - D_{\boldsymbol{\phi}}(e \mid s_n) \Big) \Big] \Big), \quad (3)$$

where $\mathcal{S} = \{s_n\}_{n=1}^{|\mathcal{S}|}$ is a set of word sequences (query can be regarded as a word sequence as well) extracted from all documents in $\mathcal{D}$ with $s_n = \{v_1, \ldots, v_{|s_n|}\}$ being the n-th word sequence, and the size of the sequence set is $|\mathcal{S}|$, i.e., the total number of word sequences in the set $\mathcal{S}$. Here $|s_n|$ is the size of the n-th sequence, i.e., the total number of words in the sequence s_n. Obviously, unlike queries that are manually defined in the ground truth, our set of word sequences $\mathcal{S}$ can be obtained by an unsupervised way. All the words in the sequences are supposed from a vocabulary $\mathcal{V}$. Note that both generative and discriminative networks with parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, respectively, are still modeled in (3), with the generative networks directly generate/select an expert from the distribution $p_{\boldsymbol{\theta}}(e \mid s)$. Later in this section, we will model each sequence as m-grams extracted from documents.

As we do not have ground truth during unsupervised learning, without loss of generality, as can be seen in (3) we replace N in (1), the number of ground truth queries, to be $|\mathcal{S}|$, and $p_{GT}(e \mid q_n)$ in (1) to be $p_{\boldsymbol{\phi}}(e \mid s_n)$, i.e., the probability of a candidate expert e having expertise on the topic underlying the sequence s_n, respectively.
In what follows, we present how we define $p_{\boldsymbol{\phi}}(e_i \mid s_n)$, $p_{\boldsymbol{\theta}}(e_i \mid s_n)$ and $D_{\boldsymbol{\phi}}$ in (3) via an unsupervised way, where we do not require topic-candidate ground truth assessments and manually defined queries for training. Recall that in (1) we directly represent $G_{\boldsymbol{\theta}}$ as $p_{\boldsymbol{\theta}}(e \mid q)$, and thus we do not need to redefine $G_{\boldsymbol{\theta}}$ at all.

Definition of $p_{\boldsymbol{\phi}}(e_i \mid s_n)$. According to unsupervised language model, we closely follow the work in [50] and rank all the candidate experts in $\mathcal{E}$ by the conditional probability of a candidate expert e_i given a topic/sequence $s_n = \{v_1, \ldots, v_{|s_n|}\}$:

$$p_{\boldsymbol{\phi}}(e_i \mid s_n) = \frac{1}{Z_1} \widetilde{p}_{\boldsymbol{\phi}}(e_i \mid v_1, \ldots, v_{|s_n|}) = \frac{1}{Z_1} \prod_{j=1}^{|s_n|} p_{\boldsymbol{\phi}}(e_i \mid v_j)$$
$$\overset{\text{rank}}{=} \frac{1}{Z_1} \exp\Bigg(\sum_{j=1}^{|s_n|} \log \Big(p_{\boldsymbol{\phi}}(e_i \mid v_j) \Big) \Bigg), \quad (4)$$

where $\widetilde{p}_{\boldsymbol{\phi}}(e_i | v_1, \ldots, v_{|s_n|})$ is the unnormalized score, $Z_1 = \sum_{i=1}^{|\mathcal{E}|} \exp(\sum_{j=1}^{|s_n|} \log(p_{\boldsymbol{\phi}}(e_i \mid v_j)))$ is the a normalization term (The transformation to log-space in (4) is a well-known trick to prevent floating point underflow [43]), $\overset{\text{rank}}{=}$ denotes that ranking according to the former equation is equivalent to ranking according to the latter one, and $p_{\boldsymbol{\phi}}(e_i \mid v_j)$ is the probability of a candidate having expertise on the topic specific to a word $v_j \in \mathcal{V}$, which can be defined as a log-linear learning model:

$$p_{\boldsymbol{\phi}}(e_i | v_j) = \frac{1}{Z_2} \exp(\mathbf{e}_i^{\boldsymbol{\phi}} \cdot \mathbf{v}_j^{\boldsymbol{\phi}\top} + b_i^{\boldsymbol{\phi}}), \quad (5)$$

where $\mathbf{e}_i^{\boldsymbol{\phi}}$ and $\mathbf{v}_j^{\boldsymbol{\phi}}$ are the expert e_i's and the word v_j's $1 \times c$ semantic representation vectors with c being the size of the vectors, respectively, $b_i^{\boldsymbol{\phi}}$ is a bias for the candidate expert e_i, $\mathbf{v}^{\boldsymbol{\phi}\top}$ is the transpose of $\mathbf{v}^{\boldsymbol{\phi}}$, and $Z_2 = \sum_{i=1}^{|\mathcal{E}|} \exp(\mathbf{e}_i \cdot \mathbf{v}_j^\top + b_i)$ is the normalization term. Obviously, the semantic representations of all the candidate experts, $\{\mathbf{e}_i^{\boldsymbol{\phi}}\}_{i=1}^{|\mathcal{E}|}$, the semantic representations of all the words, $\{\mathbf{v}_j^{\boldsymbol{\phi}}\}_{j=1}^{|\mathcal{V}|}$, and the corresponding biases $\{b_i^{\boldsymbol{\phi}}\}_{i=1}^{|\mathcal{E}|}$ constitute the parameter $\boldsymbol{\phi}$ of our discriminative model $D_{\boldsymbol{\phi}}$, i.e., $\boldsymbol{\phi} = \Big\{\{\mathbf{e}_i^{\boldsymbol{\phi}}\}_{i=1}^{|\mathcal{E}|}, \{\mathbf{v}_j^{\boldsymbol{\phi}}\}_{j=1}^{|\mathcal{V}|}, \{b_i^{\boldsymbol{\phi}}\}_{i=1}^{|\mathcal{E}|}\Big\}$. Note that we define a neural network in (5) with a log-linear model only. Adding more layers into the neural network defined in (5) is possible. But our experiments found that the shallow log-linear model defined in (4) performs well-enough in most cases and adding more layers will require more training times and lose the transparency of the model. Other neural networks such as CNN for defining $p_{\boldsymbol{\phi}}(e_i \mid s_n)$ are possible, but we want to be focused and leave these as future work.

Definition of $p_{\boldsymbol{\theta}}(e_i \mid s_n)$. To focus more on our USGAN, we define $p_{\boldsymbol{\theta}}(e_i \mid s_n)$ for our generative model in such a way that is the same as that in our discriminative model, i.e., via (4) and (5), but with different semantic parameters $\{\mathbf{e}_i^{\boldsymbol{\theta}}\}_{i=1}^{|\mathcal{E}|}$, $\{\mathbf{v}_j^{\boldsymbol{\theta}}\}_{j=1}^{|\mathcal{V}|}$ and $\{b_i^{\boldsymbol{\theta}}\}_{i=1}^{|\mathcal{E}|}$. Then, the parameter $\boldsymbol{\theta}$ in our generative model is set to be $\boldsymbol{\theta} = \Big\{\{\mathbf{e}_i^{\boldsymbol{\theta}}\}_{i=1}^{|\mathcal{E}|}, \{\mathbf{v}_j^{\boldsymbol{\theta}}\}_{j=1}^{|\mathcal{V}|}, \{b_i^{\boldsymbol{\theta}}\}_{i=1}^{|\mathcal{E}|}\Big\}$. Again, to keep focused, we leave using alternative neural networks for generative model in our USGAN as future work.

Definition of $D_{\boldsymbol{\phi}}$. Before we follow that in [50] and define our final $D_{\boldsymbol{\phi}}$, we construct a pseudo ground truth probability of candidate e_i having expertise on a document $d_k \in \mathcal{D}$, i.e., $p_{\widetilde{GT}}(e_i \mid d_k)$, based on document-candidate associations by an unsupervised way:

$$p_{\widetilde{GT}}(e_i \mid d_k) = \begin{cases} 0, & e_i \notin \mathcal{E}_{d_k}, \\ \frac{1}{|\mathcal{E}_{d_k}|}, & e_i \in \mathcal{E}_{d_k}, \end{cases} \quad (6)$$

where $\mathcal{E}_{d_k}$ and $|\mathcal{E}_{d_k}|$ are a set of candidates whose names or email addresses are observed in the document d_k and the total number of the observed candidates in the document d_k (i.e., the size of the set), respectively. For each document $d \in \mathcal{D}$, we extract a set of m-grams where m remains fixed during our unsupervised learning. Let $\mathcal{M}_{d_k}$ denote the set of the m-grams extracted from d_k with $d_k^{(l)} \in \mathcal{M}_{d_k}$ being the l-th m-grams. Let $\mathbf{p}_{\widetilde{GT}}(e_i \mid d_k) = \{p_{\widetilde{GT}}(e_i \mid d_k^{(l)})\}_{l=1}^{|\mathcal{M}_{d_k}|}$, where $|\mathcal{M}_{d_k}|$ is the size of the m-grams (i.e., word sequences) set extracted from d_k and let $p_{\widetilde{GT}}(e_i \mid d_k^{(l)}) = p_{\widetilde{GT}}(e_i \mid d_k)$ which can be obtained by (6). We also denote $\mathbf{p}_{\boldsymbol{\phi}}(e_i \mid d_k)$ to be $\mathbf{p}_{\boldsymbol{\phi}}(e_i \mid d_k) = \{p_{\boldsymbol{\phi}}(e_i \mid d_k^{(l)})\}_{l=1}^{|\mathcal{M}_{d_k}|}$. Here $p_{\boldsymbol{\phi}}(e_i \mid d_k^{(l)}) = p_{\boldsymbol{\phi}}(e_i \mid v_1^{d_k^{(l)}}, \ldots, v_m^{d_k^{(l)}})$ can be obtained by (4) with $v_x^{d_k^{(l)}}$ being the x-th word in the m-gram $d_k^{(l)}$. Accordingly, we define our discriminative model $D_{\boldsymbol{\phi}}(e_i \mid d_k)$ based on the cross-entropy $C\Big(\mathbf{p}_{\widetilde{GT}}(e_i \mid d_k), \mathbf{p}_{\boldsymbol{\phi}}(e_i \mid d_k)\Big)$ in an

unsupervised way as follows:

$$
\begin{aligned}
D_{\boldsymbol{\phi}}(e_i \mid d_k) =& \frac{1}{Z_3}\frac{|d_{\max}|}{|d_k|} C\left(\mathbf{p}_{\widetilde{GT}}(e_i \mid d_k), \mathbf{p}_{\boldsymbol{\phi}}(e_i \mid d_k)\right) \\
&+ \epsilon\left(\|\mathbf{e}_i^{\boldsymbol{\phi}}\|_2^2 + \frac{1}{|\mathcal{M}_{d_k}|}\sum_{j=1}^{|\mathcal{M}_{d_k}|}\|\mathbf{v}_j^{\boldsymbol{\phi}}\|_2^2\right) \\
=& -\frac{|d_{\max}|}{Z_3 \cdot |d_k| \cdot |\mathcal{M}_{d_k}|}\sum_{l=1}^{|\mathcal{M}_{d_k}|}\left(p_{\widetilde{GT}}(e_i \mid d_k^{(l)}) \cdot \log p_{\boldsymbol{\phi}}(e_i \mid d_k^{(l)})\right) \\
&+ \epsilon\left(\|\mathbf{e}_i^{\boldsymbol{\phi}}\|_2^2 + \frac{1}{|\mathcal{M}_{d_k}|}\sum_{j=1}^{|\mathcal{M}_{d_k}|}\|\mathbf{v}_j^{\boldsymbol{\phi}}\|_2^2\right),
\end{aligned} \tag{7}
$$

where $\|\mathbf{x}\|_2$ is the 2-norm of a vector $\mathbf{x}$, ϵ is a free regularization parameter, $d_{\max} = \arg\max_{d\in\mathcal{D}}|d|$ is the longest document in the collection, $|d_{\max}|$ and $|d_k|$ are the length of the corresponding documents, respectively, and Z_3 is a normalization constant that forces $D_{\boldsymbol{\phi}}(e \mid d)$ to be within the range $(0, 1]$, which can be defined to be the maximal score of $D_{\boldsymbol{\phi}}(e \mid d)$ before the normalization.

4.2 Optimizing the Discriminative Model

The objective of the discriminative model is to maximize the log-likelihood of correctly distinguishing the candidate experts from $p_{\boldsymbol{\phi}}(e \mid s)$ and those generated from $p_{\boldsymbol{\theta}}(e \mid s)$. As (7) is built on (4) and thus maximizing (7) for all candidates and documents means to maximizing (4) for all the candidates and documents in collection, and with the candidate experts sampled from the current optimal generative model $p_{\boldsymbol{\theta}^*}(e \mid s)$, we can obtain the optimal parameters for the discriminative model defined in (3) as:

$$
\begin{aligned}
\boldsymbol{\phi}^* =& \arg\max_{\boldsymbol{\phi}}\sum_{k=1}^{|\mathcal{D}|}\Big(\mathbb{E}_{e\sim p_{\boldsymbol{\phi}}(e|d_k)}[\log D_{\boldsymbol{\phi}}(e \mid d_k)] + \\
&\mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\log(1 - D_{\boldsymbol{\phi}}(e \mid d_k))\right]\Big) \\
=& \arg\max_{\boldsymbol{\phi}}\sum_{k=1}^{|\mathcal{D}|}\Big(\frac{1}{|\mathcal{E}|}\sum_{i=1}^{|\mathcal{E}|}\log D_{\boldsymbol{\phi}}(e_i \mid d_k) + \\
&\mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\log(1 - D_{\boldsymbol{\phi}}(e \mid d_k))\right]\Big).
\end{aligned} \tag{8}
$$

As can be seen in (7), $D_{\boldsymbol{\phi}}(e \mid d)$ is differentiable with respect to the parameter $\boldsymbol{\phi}$, and thus we can obtain the optimal parameter $\boldsymbol{\phi}^*$ in (8) by stochastic gradient descent. For convenient discussion in later part of the paper, we denote $\frac{1}{|\mathcal{E}|}\sum_{i=1}^{|\mathcal{E}|}\log D_{\boldsymbol{\phi}}(e_i \mid d_k) + \mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\log(1 - D_{\boldsymbol{\phi}}(e \mid d_k)\right]$ in (8) as $J^{D_{\boldsymbol{\phi}}(d_k)}$ and thus (8) can be simplified as $\boldsymbol{\phi}^* = \arg\max_{\boldsymbol{\phi}}\sum_{k=1}^{|\mathcal{D}|}J^{D_{\boldsymbol{\phi}}(d_k)}$.

4.3 Optimizing the Generative Model

In contrast, the objective of the generative model $G_{\boldsymbol{\theta}}$ with the form of $p_{\boldsymbol{\theta}}(e \mid d)$ is to minimize the log-likelihood of $1 - D_{\boldsymbol{\phi}}$ such that the model can fit the underlying relevance distribution over candidate experts, $p_{\widetilde{GT}}(e \mid d)$. The generative model samples candidates from the whole candidate expert set $\mathcal{E}$ based on the distribution $p_{\boldsymbol{\theta}}(e \mid d)$ and tries to fool the discriminative model $D_{\boldsymbol{\phi}}$.

It is worth mentioning that unlike most GAN-based models [1, 58], we let our generative model $G_{\boldsymbol{\theta}}$ directly generate known candidate experts but not their features, because our work here intends to directly select relevant experts from a given candidate set $\mathcal{E}$. Note that as presented in (1), it is feasible to first generate features as denoted by $\mathbf{z}$ and then generate candidate experts from $G_{\boldsymbol{\theta}}(\mathbf{z} \mid s)$ by USGAN. However, to be focused, we leave this as future work.

After we obtain the optimal parameter $\boldsymbol{\phi}^*$ from (8), we then optimize the generative model $G_{\boldsymbol{\theta}}$ via performing the following minimization:

$$
\begin{aligned}
\boldsymbol{\theta}^* =& \arg\min_{\boldsymbol{\theta}}\sum_{k=1}^{|\mathcal{D}|}\Big(\underbrace{\mathbb{E}_{e\sim p_{\boldsymbol{\phi}}(e|d_k)}[\log D_{\boldsymbol{\phi}}(e \mid d_k)]}_{\text{Partial derivative of this term w.r.t } \boldsymbol{\theta} \text{ is 0}} + \\
&\mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\log(1 - D_{\boldsymbol{\phi}}(e \mid d_k)\right]\Big) \\
=& \arg\min_{\boldsymbol{\theta}}\sum_{k=1}^{|\mathcal{D}|}\underbrace{\mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\log(1 - D_{\boldsymbol{\phi}}(e \mid d_k)\right]}_{\text{Denoted this term as } J^{G_{\boldsymbol{\theta}}(d_k)}},
\end{aligned} \tag{9}
$$

where we denote the term $\mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\log(1 - D_{\boldsymbol{\phi}}(e \mid d_k)\right]$ as $J^{G_{\boldsymbol{\theta}}(d_k)}$ and thus it can be simplified to be $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}}\sum_{k=1}^{|\mathcal{D}|}J^{G_{\boldsymbol{\theta}}(d_k)}$. As the sampling $e \sim p_{\boldsymbol{\theta}}(e \mid d_k)$ is discrete in the objective function (9) and unlike many GAN-based algorithms [20], we can not simply optimize our $\boldsymbol{\theta}$ by gradient descent. We utilize policy gradient based reinforcement learning (REINFORCE) [49, 54, 57] for obtaining the optimal $\boldsymbol{\theta}^*$ in (9), and the gradient is derived as:

$$
\begin{aligned}
&\nabla_{\boldsymbol{\theta}}J^{G_{\boldsymbol{\theta}}(d_k)} \\
&= \nabla_{\boldsymbol{\theta}}\left(\mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\log(1 - D_{\boldsymbol{\phi}}(e \mid d_k)\right]\right) \\
&= \sum_{i=1}^{|\mathcal{E}|}\nabla_{\boldsymbol{\theta}}\left(p_{\boldsymbol{\theta}}(e_i \mid d_k)\log(1 - D_{\boldsymbol{\phi}}(e_i \mid d_k))\right) \\
&= \sum_{i=1}^{|\mathcal{E}|}p_{\boldsymbol{\theta}}(e_i \mid d_k)\nabla_{\boldsymbol{\theta}}\left(\log p_{\boldsymbol{\theta}}(e_i \mid d_k)\log(1 - D_{\boldsymbol{\phi}}(e_i \mid d_k))\right) \\
&= \mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\nabla_{\boldsymbol{\theta}}\left(\log p_{\boldsymbol{\theta}}(e_i \mid d_k)\log(1 - D_{\boldsymbol{\phi}}(e_i \mid d_k))\right)\right] \\
&\simeq \frac{1}{H}\sum_{i=1}^{H}\nabla_{\boldsymbol{\theta}}\left(\log p_{\boldsymbol{\theta}}(e_i \mid d_k)\log(1 - D_{\boldsymbol{\phi}}(e_i \mid d_k))\right),
\end{aligned} \tag{10}
$$

where in the last step we sample H candidates from the current version of generative model $p_{\boldsymbol{\theta}}(e \mid d_k)$ to approximate the partial derivative of $J^{G_{\boldsymbol{\theta}}(d_k)}$ with regards to the parameter $\boldsymbol{\theta}$. According to reinforcement learning [49, 54], the term $\log(1 - D_{\boldsymbol{\phi}}(e_i \mid d_k))$ in the last step of (10) acts as the reward for the policy $p_{\boldsymbol{\theta}}(e \mid d_k)$ taking an action e_i in the environment d_k. Other ways, e.g., categorical reparameterization with Gumbel-Softmax [25], to derive the gradient $\nabla_{\boldsymbol{\theta}}J^{G_{\boldsymbol{\theta}}(d_k)}$ are possible, but we leave this for future investigation.

To reduce variance during the REINFORCE learning, we replace the reward term $\log(1 - D_{\boldsymbol{\phi}}(e_i \mid d_k))$ by its advantage function as:

$$
\log(1 - D_{\boldsymbol{\phi}}(e_i \mid d_k)) - \mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\log(1 - D_{\boldsymbol{\phi}}(e \mid d_k))\right],
$$

where the term $\mathbb{E}_{e\sim p_{\boldsymbol{\theta}}(e|d_k)}\left[\log(1 - D_{\boldsymbol{\phi}}(e \mid d_k))\right]$ acts as the baseline function in policy gradient based reinforcement learning [49].

4.4 Overview, Example and Discussion

Overview of USGAN. The overview of our proposed unsupervised semantic USGAN expert retrieval model is shown in Algorithm 1. We initialize the semantic representations of all words, $\{\mathbf{v}_j^{\phi}\}_{j=1}^{|\mathcal{V}|}$ and $\{\mathbf{v}_j^{\theta}\}_{j=1}^{|\mathcal{V}|}$, in both the discriminative and generative models, with pre-trained word representations trained on Wikipedia 2014 dataset [1] [44], respectively. The dimension of both words' and candidates' semantic representations is set to be $c = 300$. We sample initial vectors uniformly in the range $[-\frac{1}{\sqrt{300}}, \frac{1}{\sqrt{300}}]$ for all the candidates as their initial semantic representations, and let all the initial biases be 0. (step 1 in Algorithm 1). We pre-train our $\boldsymbol{\phi}$ by maximizing (7) using batched gradient descent. After the pre-training of $\boldsymbol{\phi}$, we let $\boldsymbol{\theta} = \boldsymbol{\phi}$. (step 2 in Algorithm 1). We iteratively perform the minimax game where we update the parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ at each iteration and terminate the game once it converges. Obviously, we can perform the unsupervised training for our USGAN offline, whereas the USGAN with optimal parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ after training can be directly applied online for expert retrieval.

After we obtain the optimal parameters $\boldsymbol{\phi}^*$ of the discriminative neural-network and $\boldsymbol{\theta}^*$ of the generative neural-network in USGAN, given a query q, we produce a rank list of the candidate experts $\mathcal{L}$ by their relevance probabilities to the query, i.e., either by $p_{\boldsymbol{\phi}^*}(e \mid q)$ with parameter $\boldsymbol{\phi}^*$ of the discriminative model or by $p_{\boldsymbol{\theta}^*}(e \mid q)$ with parameter $\boldsymbol{\theta}^*$ of the generative model. See (4) for computing $p_{\boldsymbol{\phi}^*}(e \mid q)$ and $p_{\boldsymbol{\theta}^*}(e \mid q)$. Obviously, the unsupervised training for USGAN can be performed offline, whereas the USGAN with optimal parameters $\boldsymbol{\phi}^*$ and $\boldsymbol{\theta}^*$ after the training can be directly applied online for the expert retrieval task.

An Example. Here we provide an example to further illustrate the underlying idea of our proposed USGAN model. We are given a corpora of documents (any formats, e.g., Code docs, Emails, web pages, Wiki web pages etc.), a set of test queries, a set of candidate experts, and the ground truth. Suppose the corpora only contains two Wiki documents: d_1 and d_2, the content of which are: d_1={"*Turing was influential in development of theoretical computer science, providing a formalization of the concepts of algorithm and computation.*"}, d_2={"*Washington was an American statesman who served as the first president of the United States.*"}, respectively, and no other information of the documents is available. We further suppose there is only one test query, q_1={"algorithm"}, and the ground truth for evaluation is: {q_1–Turing–1; q_1–Washington–0} where the "1" indicates that the expert is relevant to the query; whereas "0" indicates non-relevant. Thus, the ground truth denotes that Turing is an expert on topic q_1, whereas Washington is not an expert on topic q_1. The goal of our USGAN is to obtain embeddings (semantic vectors) of both the candidate experts and the words in the corpora, where we assume that the ground truth is not available during training (the ground truth can only be used for evaluation). According to (6), we have $p_{\widetilde{GT}}(\text{Turing} \mid d_1) = 1$, $p_{\widetilde{GT}}(\text{Washington} \mid d_1) = 0$, $p_{\widetilde{GT}}(\text{Turing}|d_2) = 0$, $p_{\widetilde{GT}}(\text{Washington}|d_2) = 1$. All of these probabilities are obtained in an unsupervised way (just identify if the names can be observed in the documents) and are used as the pseudo relevance signals during the unsupervised training of our USGAN. We use m-grams during the training. Suppose $m = 1$ and thus we have probabilities (signals) of each word in the document for the unsupervised training, e.g., $p_{\widetilde{GT}}(\text{Turing} \mid \text{"computer"}) = 1$, (as $p_{\widetilde{GT}}(\text{Turing} \mid d_1) = 1$ and the word "computer" is within d_1), $p_{\widetilde{GT}}(\text{Turing}|\text{"algorithm"}) = 1$, etc. These signals will be used during the unsupervised training. Once the training is done, we will obtain the embeddings. Given a test query={"algorithm"}, we rank the experts by $p_{\boldsymbol{\phi}^*}(e \mid q)$ or $p_{\boldsymbol{\theta}^*}(e \mid q)$ computed by (4), which will result in a ranking, and will be evaluated by the ground truth.

[1] Publicly available from https://nlp.stanford.edu/projects/glove/.

Algorithm 1: Overview of USGAN for expert retrieval.

Input : $\mathcal{D}$, $\mathcal{E}$, ϵ
Output: Optimal $\boldsymbol{\phi}^*$ of $D_{\boldsymbol{\phi}^*}$, and Optimal $\boldsymbol{\theta}^*$ of $G_{\boldsymbol{\theta}^*}$

1 Initialize $\boldsymbol{\phi}$ of $D_{\boldsymbol{\phi}}$ and $\boldsymbol{\theta}$ of $G_{\boldsymbol{\theta}}$
2 Pre-train $p_{\boldsymbol{\phi}}(e \mid d)$ and $p_{\boldsymbol{\theta}}(e \mid d)$ with all documents in $\mathcal{D}$ by (7)
3 **repeat**
4 **for** *g-steps* **do**
5 **for** $k = 1, \ldots, |\mathcal{D}|$ **do**
6 Sample top-H candidate experts from the current version of generative model $p_{\boldsymbol{\theta}}(e \mid d_k)$ for document d_k
7 Update $\boldsymbol{\theta}$ in the generator $G_{\boldsymbol{\theta}}$, i.e., (9), by descending its stochastic gradient with REINFORCE learning: $\nabla_{\boldsymbol{\theta}} \sum_{k=1}^{|\mathcal{D}|} J^{G_{\boldsymbol{\theta}}(d_k)} = \sum_{k=1}^{|\mathcal{D}|} \nabla_{\boldsymbol{\theta}} J^{G_{\boldsymbol{\theta}}(d_k)}$
8 **for** *d-steps* **do**
9 **for** $k = 1, \ldots, |\mathcal{D}|$ **do**
10 Sample top-H candidate experts from the current version of generative model $p_{\boldsymbol{\theta}}(e \mid d_k)$ for document d_k
11 Update $\boldsymbol{\phi}$ in the discriminative $D_{\boldsymbol{\phi}}$, i.e., (8), by ascending its stochastic gradient: $\nabla_{\boldsymbol{\phi}} \sum_{k=1}^{|\mathcal{D}|} J^{D_{\boldsymbol{\phi}}(d_k)} = \sum_{k=1}^{|\mathcal{D}|} \nabla_{\boldsymbol{\phi}} J^{D_{\boldsymbol{\phi}}(d_k)}$
12 **until** USGAN *converges*

Discussions. To our knowledge, there is only one GAN-based algorithm for information retrieval – IRGAN [51]. Our USGAN differs from IRGAN in at least four aspects: (1) The main difference between IRGAN and our USGAN lies in the fact that IRGAN requires ground truth data to train the model, which will hinder its applications in the scenarios that no labelled data is available, whereas our USGAN is an unsupervised GAN-based algorithm and can work without labelled data. (2) IRGAN directly defines its discriminative networks using ground truth data – the samples from the generative networks can be directly distinguished with the labelled data in the ground truth, whereas we define an unique discriminative function for the expert retrieval task in an unsupervised way (see (7)). (3) Our USGAN is semantic retrieval networks and in which we avoid feature extractions, whereas IRGAN requires to extract features for query-entity pairs for training the networks. (4) It is impossible to directly apply IRGAN to the task of expert retrieval even if we assume that the labelled data is available, as there is no direct connection between a relevant expert and a query. In addition, a state-of-the-art UESM (unsupervised, efficient and semantic expertise model) model [50] is just proposed, and has

shown its good performance on the expert retrieval task. Our USGAN differs from UESM in at least three aspects: (1) The strategy to obtain optimal parameters in USGAN is different compared to that in UESM. USGAN utilizes reinforcement learning to update the parameters; whereas UESM directly apply stochastic gradient descent to obtain them. (2) The frameworks are absolutely different. USGAN is a GAN-based algorithm, whereas UESM is a common neural network-based language model. (3) USGAN captures two parameters: ϕ from its discriminative network and θ from its generative network. The competitions between the two networks drive both networks to improve their semantic representations; whereas UESM is built on one network only and no competitions in it.

5 EXPERIMENTAL SETUP

In this section, we describe our experimental setup.

5.1 Research Questions

The research questions guiding the remainder of the paper are: **(RQ1)** How is the expert retrieval performance of USGAN compared to other state-of-the-art methods? **(RQ2)** What is the effect of the window size (m-grams) on the performance of our USGAN expert retrieval method? **(RQ3)** What is the effect of the sampling size (parameter H in (10)) on the performance of our USGAN? **(RQ4)** What is impact of number of iterations on the retrieval performance of our generative model G_θ and discriminative model D_ϕ? **(RQ5)** Does our USGAN outperform the best unsupervised baseline method on each query?

5.2 Dataset

For evaluation purposes we use datasets made available at the 2005 and 2006 editions of the TREC enterprise tracks [14, 48]. Note that for both of these two years' tracks, the document collections [2] are the same but with different testing queries. The document collection contains a crawl of the World Wide Web Consortium (W3C) – a heterogenous document repository containing a mixture of document types, which include lists (from email forum), dev, www, esw, other, and people (personal homepages). Statistical information of the dataset is as follows: total number of documents: 331,037, average document length: 1,237.23, total number of candidates: 1092, total number of document-candidate associations: 200,939 (an association indicates that name or email address of a candidate expert appears in a document), average number of associations per document (Only documents with at least one association are considered): 2.14, average number of associations per candidate: 281.03, 50 test queries were created for the 2005 track with all of their ground truths provided, and 55 test queries were created for the 2006 track with only 50 of their ground truths provided. Other datasets, such as the CSIRO Enterprise Research Collection (CERC) dataset that was a dump of the intranet of Australia's national science agency and was built in the TREC 2007 and 2008 enterprise tracks [9] for testing and the employee dataset of the Tilburg University that consists of bi-lingual, heterogeneous documents [10], are impossible to be used for evaluating expert retrieval algorithms, because they are no longer publicly distributed. [3]

[2]Publicly available from http://trec.nist.gov/data/enterprise.html.
[3]Details can be found at http://trec.nist.gov/data/enterprise.html.

5.3 Baselines

As our goal is to improve performance of the expert retrieval task in an unsupervised way with no labelled data, we only consider unsupervised baselines for comparisons. As IRGAN is a supervised method, it is inappropriate to be taken as baseline. We make comparisons among our USGAN and the following unsupervised non-neural networks-based and neural networks-based algorithms:

Term Frequency and Inverse Document Frequency (TF-IDF). It weights a term by considering both its frequency and inverse document frequency in a given document [15].

Latent Dirichlet Allocation (LDA). This model [12] infers semantic topic distributions specific to each document and each word via the LDA model.

Author Topic Model (AuthorT). This model [45] infers semantic topic distributions specific to each candidate, each document and each word.

Profile-Centric Language Model (Model 1). It is an unsupervised, non-semantic language model for expert finding. It amasses all the term information from all the documents associated with the candidate for a given query and uses this to represent that candidate [4, 8].

Document-Centric Language Model (Model 2). It is an unsupervised, non-semantic language model for expert finding. It first selects documents associated with the candidate, using either the so-called document-centric or the candidate-centric language model. It then considers whether the query can be generated from those documents [4, 8].

Unsupervised, Efficient and Semantic Model (UESM). It is a state-of-the-art unsupervised, semantic, neural expert finding model based on neural networks, exclusively employs textual evidence from document-expert associations for learning representations of terms and experts in an unsupervised way [50].

In terms of the baselines, TF-IDF, LDA and AuthorT, we represent experts, documents and queries by their semantic vector spaces inferred from the corresponding models, and then adapt a generic framework proposed in [16] to rank experts in response to a query, respectively. For Models 1 & 2, we apply Dirichlet smoothing method with parameter β equal to the average document length [5]. We only include the state-of-the-art deep learning model, UESM, for comparisons. Traditional deep learning models, e.g., CNN, Recurrent Neural Network (RNN), LSTM and their state-of-the-art variants [19], are supervised learning algorithms, and thus they are inappropriate to be included as our baselines. There are some unsupervised deep learning models, but they mainly aim at encoding and decoding for generating semantic features in unsupervised ways and serve for different purposes [19]. Thus these unsupervised deep learning models can not be directly applied to expert finding task and used as our baselines.

Later for convenient discussion, we denote our methods that produce the retrieval results by the discriminative model $p_{\phi^*}(e \mid q)$ and the generative model $p_{\theta^*}(e \mid q)$ as USGAN$_D$ and USGAN$_G$, respectively.

5.4 Evaluation Metrics and Settings

The evaluation metrics used to assess the performance of the expert retrieval algorithms are the ones widely used in TREC enterprise

tracks [14, 48]: MAP (Mean Average Precision), R-Prec, MRR (Mean Reciprocal Rank) and P@k (Precision at k) [15]. R-Prec is the precision after R documents have been retrieved, where R is the total number of relevant documents for the query. We also use NDCG@k (Normalized Discounted Cumulative Gain at k [26]). For P@k and NDCG@k, we set k to 5 and 10, respectively, to align with the cut-off used in the TREC enterprise tracks.

For our USGAN (both USGAN$_D$ and USGAN$_G$) and the baseline UESM, we use a 70%/20%/10% split of the queries for unsupervised training, validation and test sets, respectively. We optimize the objectives during the unsupervised training using different values of the parameters, e.g., the size of m-grams in the methods, and the sampling size in USGAN. The best values are then chosen on the validation set, and evaluated on the test queries. The *g-steps* and *d-steps* epochs in USGAN (see Algorithm 1) keep running until the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ updated by the corresponding stochastic gradients converge, respectively. Other settings, e.g., the initialization for $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ etc., can be found in subsection 4.4. The train/validation/test splits are permuted until all the queries were chosen once for the test set. We repeat the experiments 10 times and report the average results. The statistical significance of the observed differences between the performance of two ranking algorithms across the queries is tested using a two-tailed paired t-test and is denoted using $^{\blacktriangle}$ (or $^{\blacktriangledown}$) for $\alpha = .01$, and $^{\triangle}$ (and $^{\triangledown}$) for $\alpha = .05$.

Table 1: Retrieval performance of USGAN and the baselines on MAP, R-Prec, MRR, P@5, 10, NDCG@5, 10, respectively. Statistically significant differences between USGAN$_G$ and best baseline model, UESM, and between USGAN$_D$ and USGAN$_G$ are marked in the upper right hand corner of USGAN$_G$'s and USGAN$_D$'s scores, respectively.

					P@		NDCG@	
		MAP	R-Prec	MRR	5	10	5	10
2005	TF-IDF	.147	.165	.520	.188	.196	.199	.202
	LDA	.152	.170	.500	.200	.204	.207	.209
	AuthorT	.158	.171	.480	.216	.210	.227	.220
	Model 1	.165	.192	.560	.228	.218	.231	.226
	Model 2	.168	.184	.600	.236	.228	.247	.240
	UESM	.250	.244	.760	.420	.334	.477	.408
	USGAN$_G$	.296$^{\blacktriangle}$	.279$^{\blacktriangle}$	.810$^{\blacktriangle}$	.508$^{\blacktriangle}$	.398$^{\blacktriangle}$	.562$^{\blacktriangle}$	.479$^{\blacktriangle}$
	USGAN$_D$	.320$^{\blacktriangle}$	.302$^{\blacktriangle}$	.840$^{\triangle}$	.560$^{\blacktriangle}$	.426$^{\blacktriangle}$	.610$^{\blacktriangle}$	.514$^{\blacktriangle}$
2006	TF-IDF	.275	.288	.812	.420	.420	.365	.375
	LDA	.265	.284	.803	.376	.390	.331	.348
	AuthorT	.269	.285	.807	.392	.400	.348	.359
	Model 1	.281	.291	.816	.441	.431	.387	.389
	Model 2	.242	.258	.837	.404	.398	.330	.338
	UESM	.473	.410	.882	.727	.667	.669	.662
	USGAN$_G$	.487$^{\blacktriangle}$	.479$^{\blacktriangle}$	.931$^{\blacktriangle}$	.767$^{\blacktriangle}$	.708$^{\blacktriangle}$	.700$^{\blacktriangle}$	.693$^{\blacktriangle}$
	USGAN$_D$	.505$^{\blacktriangle}$	.495$^{\blacktriangle}$	.944$^{\triangle}$	.800$^{\blacktriangle}$	.727$^{\blacktriangle}$	.723$^{\blacktriangle}$	.725$^{\blacktriangle}$

6 RESULTS AND ANALYSIS

In what follows, we discuss and analyze our experimental results and answer the research questions. Subsection 6.1 report the overall performance of USGAN and the baseline methods (**RQ1**); subsection 6.2 investigates how the size of the window (m-grams) impacts the performance of USGAN (**RQ2**); subsection 6.3 investigates the effect of the sampling size, i.e., the value of the parameter H in (10) on the performance of USGAN (**RQ3**); subsection 6.4 illustrates the impact of iterations on the retrieval performance of our generative and discriminative models (**RQ4**); finally, subsection 6.5 shows the performance of USGAN on different test queries (**RQ5**).

6.1 Overall of Experimental Results

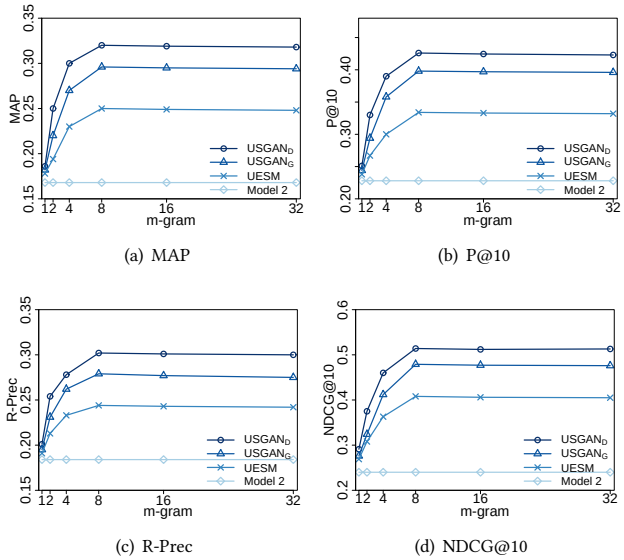
RQ1: We compare the retrieval performance of our USGAN with the baseline methods on the TREC enterprise 2005 and 2006 tracks.

Table 1 reports the performance on all the evaluation metrics. For all these two years' tracks, we have the following findings from Table 1: (i) Both our discriminative model, USGAN$_D$, and our generative model, USGAN$_G$, are able to statistically significantly outperform all the baselines on all the metrics and in both of the two tracks, which confirms the effectiveness of our unsupervised generative adversarial networks for the expert retrieval task and the fact that utilizing the REINFORCE policy gradient can help to obtain optimal parameters during our unsupervised training in Algorithm 1. (ii) All the semantic methods, i.e., USGAN$_D$, USGAN$_G$, UESM, AuthorT, and LDA, outperform non-semantic baseline TF-IDF, which demonstrates that representing experts and words in semantic ways can help to improve the retrieval performance, compared to those directly representing them by words. (iii) Both USGAN$_D$ and USGAN$_G$ outperform the state-of-the-art semantic UESM model, which demonstrates the fact that semantic representations of experts and words obtained by GAN-based algorithms are better than those obtained by traditional deep learning models for the task of expert retrieval. (iv) USGAN$_D$ works somewhat better than USGAN$_G$, which demonstrates that our discriminative model outperforms our generative model. The main reason is because we can directly update the parameter $\boldsymbol{\phi}$ in USGAN$_D$ by stochastic gradient descent, whereas we can only approximately update the parameter $\boldsymbol{\theta}$ in USGAN$_G$ by REINFORCE learning where we need to select top-H discrete candidates.

6.2 Effect of Window Size

RQ2: We vary the size of the window, i.e., the length of m-grams applied in (7) and see how it contributes to the overall retrieval performance. We only report the performance on MAP and P@10 for the TREC 2005 enterprise track as representative, as the plot patterns on other metrics and for the TREC 2006 track are similar. Fig. 1 shows how the expert retrieval performance on MAP and P@10 for the TREC 2005 enterprise track varies against the window size $m = 2^i (0 \leq i \leq 5)$, respectively.

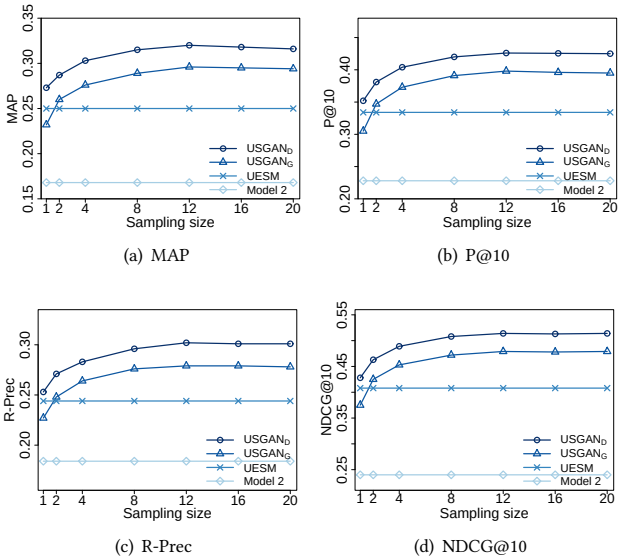
As can be seen from the figures in Fig. 1, there is a significant retrieval performance increase on the metrics with the size of the m-grams from $m = 1$ to $m = 8$ on our models USGAN$_D$, USGAN$_G$, and the baseline UESM, which underlines the importance of the size of m-grams applied in the models. The increase on the metrics implies that the retrieval performance of our models and the baseline model UESM achieved is not solely due to initialization with pre-trained representations of candidate experts and words, but that the models efficiently learn the representations tailored to the problem domain. As the window size of the m-grams increases beyond $m = 8$ the performance of our models, USGAN$_D$



(a) MAP (b) P@10

(c) R-Prec (d) NDCG@10

Figure 1: MAP, P@10, R-Prec and NDCG@10 performance of our USGAN retrieval models and the baseline models UESM and Model 2 on the TREC 2005 enterprise track with different size of m-grams, respectively.



(a) MAP (b) P@10

(c) R-Prec (d) NDCG@10

Figure 2: MAP, P@10, R-Prec and NDCG@10 performance of our USGAN retrieval models and the baseline models UESM and Model 2 on the TREC 2005 enterprise track with different sampling size H, respectively.

and USGAN$_G$, and the baseline model UESM seems to be level-off. These demonstrate one merit of our USGAN expert retrieval models: they are not sensitive to the window size of m-grams when m is large enough. Other retrieval models, e.g., Model 2, do not consider m-grams for retrieval and thus their performance is consistent over different scales of m-grams. In the next subsection, we report the experimental results with different sampling sizes.

6.3 Effect of Sampling Size

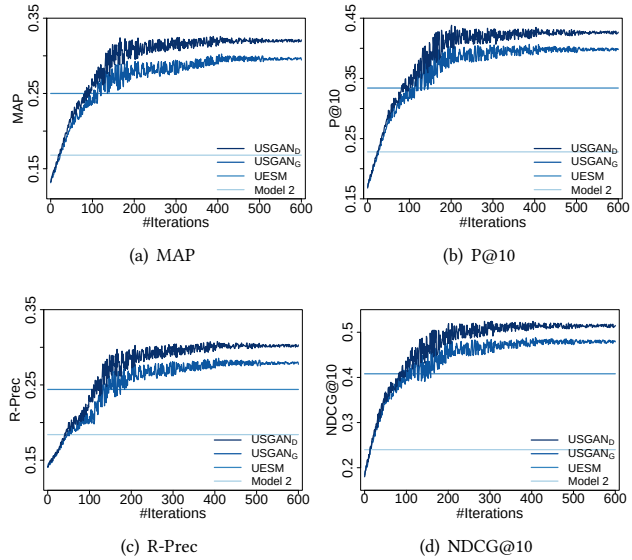
RQ3: To understand the effect of the sampling size, i.e., the parameter H in (10), on the performance of our proposed retrieval model, we vary the size of the samplings H during the REINFORCE learning for updating the parameter $\boldsymbol{\theta}$ in USGAN. In the experiments, as our goal is to yield better expert retrieval performance and to make the iterations converge as fast as possible, for each iteration during the unsupervised training, we choose top-H candidates as the samplings for updating $\boldsymbol{\theta}$.

Fig. 2 shows how the expert retrieval performance varies against the size of the samplings from $H = 1$ to $H = 20$. According to the figures, USGAN$_G$ with the size of samplings being 1 performs a little poorly compared to the baseline UESM that does not need to sample candidates for optimization. This is because one sample is not able to provide enough information for updating the parameter $\boldsymbol{\theta}$ during the REINFORCE learning in our USGAN. USGAN$_D$ with the size of samplings being 1 still can outperforms UESM, and this is because USGAN$_D$ can directly update the parameter $\boldsymbol{\phi}$ by stochastic gradient decent without sampling candidates. There is an obvious performance ascent when tuning the size of samplings from 1 to the optimal value 12, which illustrates the fact that utilizing more top-H samplings for updating the parameter $\boldsymbol{\theta}$ does help to improve the performance when $H \leq 12$. The performance reaches a plato after the optimal number of the samplings $H = 12$. These findings show another merit of our USGAN model: we do not need to sample too many discrete scores, i.e., the top-H candidate experts, at each iteration for the REINFORCE learning, while the model can still perform well. Once again, in Fig. 2 we observe that the retrieval performance of USGAN$_D$ outperforms that of USGAN$_G$ on all the different sampling sizes. This is, again, because the parameter updating strategy of USGAN$_D$ is different from that of USGAN$_G$: USGAN$_D$ directly optimize its parameter $\boldsymbol{\phi}$ with its objective function, whereas USGAN$_G$ needs to leverage REINFORCE learning to update its parameter $\boldsymbol{\theta}$. The baseline model UESM that works with a common neural network does not need to sample candidate experts for parameter updating.

6.4 Impact of Numbers of Iterations

RQ4: To figure out the impact of the number of iterations on the retrieval performance, we plot the expert retrieval performance curves of our USGAN$_D$ and USGAN$_G$ models with different number of iterations from 0 to 600 on representative evaluation metrics MAP, P@10, R-Prec and NDCG@10 in Fig. 3.

As is shown in Fig. 3 before we start the iterations, i.e., the number of iterations being 0, we can still get not too bad retrieval performance that is somewhat a litter worse than that of the baseline language model, i.e., Model 2. This is because we pre-train our USGAN$_D$ and USGAN$_G$ with all documents in $\mathcal{D}$ by (7) (step 2 in Algorithm 1) before we perform the iterations. The performance of both USGAN$_D$ and USGAN$_G$ gradually increase with the iterations increase from 0 to 20. In addition, we can observe that after about 50 iterations, USGAN$_D$ consistently outperforms USGAN$_G$ and such performance gap between them seems to increase with more iterations. These findings again confirm the fact that our updating strategies, i.e., the REINFORCE updating strategy for obtaining the optimal parameter $\boldsymbol{\theta}$ in USGAN$_G$ applied in USGAN and the



(a) MAP (b) P@10

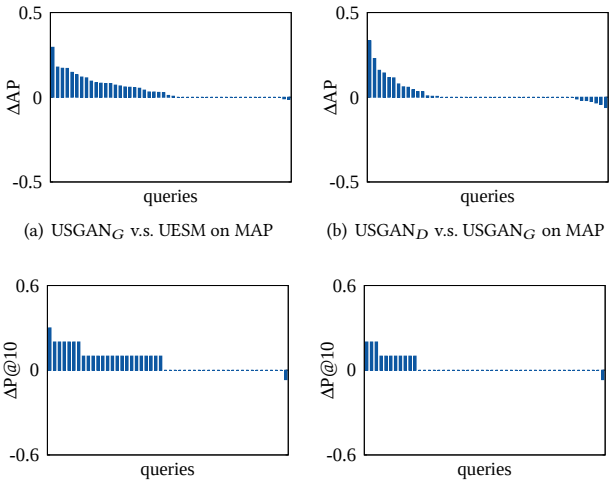
(c) R-Prec (d) NDCG@10

Figure 3: MAP, P@10, R-Prec and NDCG@10 expert retrieval performance on the TREC 2005 enterprise track with the number of iterations from 0 to 600, respectively. Figures should be viewed in color.

stochastic gradient descent updating strategy for obtaining the optimal parameter $\boldsymbol{\phi}$, are effective. The performance of both USGAN$_D$ and USGAN$_G$ dramatically changes up and down between about 120 iteration and 200 iterations. The performance of both models reaches a plato after about 400 iterations, which illustrates another merit of the proposed USGAN: it is not sensitive to the number of iterations once we have performed enough iterations.

6.5 Query-Level Analysis

RQ5: Last, we take an in-depth view of the improvements of USGAN$_G$ over the best baseline (UESM) and USGAN$_D$ over USGAN$_G$ on a per query basis, respectively. We only report the result on MAP and P@10 for the TREC 2005 enterprise track, as the results on other metrics have similar performance patterns. Fig. 4(a) and Fig. 4(b) show the per query AP performance differences between USGAN$_G$ and the best baseline UESM, and between our discriminative model USGAN$_D$ and our generative model USGAN$_G$, respectively; whereas Fig. 4(c) and Fig. 4(d) show the per query P@10 performance differences, respectively. As can be seen in Fig. 4(a), the number of queries on which USGAN$_G$ outperforms UESM is absolutely more than the number of queries on which UESM outperforms USGAN$_G$. Similar patterns can be found in Fig. 4(b) and Fig. 4(d), where the number of queries on which USGAN$_D$ outperforms USGAN$_G$ is more than that the number of queries on which USGAN$_G$ outperforms USGAN$_D$ on the metrics, respectively. These findings further support the conclusion that our unsupervised generative adversarial networks, USGAN$_D$ and USGAN$_G$, are able to retrieve more relevant experts for a given query compared to that of the best baseline model, UESM, and our discriminative model, USGAN$_D$, works better than our generative model, USGAN$_G$.



(a) USGAN$_G$ v.s. UESM on MAP (b) USGAN$_D$ v.s. USGAN$_G$ on MAP

(c) USGAN$_G$ v.s. UESM on P@10 (d) USGAN$_D$ v.s. USGAN$_G$ on P@10

Figure 4: Per-query MAP retrieval performance differences on the 2005 track (a) between USGAN$_G$ and UESM on MAP, (b) between USGAN$_D$ and USGAN$_G$ on MAP, (c) between USGAN$_G$ and UESM on P@10, and b) between USGAN$_D$ and USGAN$_G$ on P@10, respectively. A bar above the line $y = 0$ indicates the former model outperforms the latter model, while the opposite is true for bars below $y = 0$, respectively.

7 CONCLUSIONS

We have studied the problem of expert retrieval in enterprise corpora. We have proposed an unsupervised semantic minimax two-player game, i.e., unsupervised semantic generative adversarial networks (USGAN), for the expert retrieval. Our USGAN consists of a discriminative model and a generative model, and semantically represent both candidate experts and words to alleviate the mismatch problems among queries, documents and candidate experts. The unsupervised training signal for our semantic generative network G is provided by our semantic discriminator network D that is trained to distinguish between candidate experts from G and other potential better candidates. Competition in the game drives both models to improve their exert retrieval performance. Unlike most previous generative adversarial networks-based algorithms that require a large amount of labelled data, we train our USGAN in an unsupervised way to obtain the semantic representations of both experts and words, where we consider the candidate-document associations for the optimization during the training. To obtain optimal parameters in our generative model, we apply REINFORCE policy gradient during the iterations. Importantly, our USGAN goes beyond tackling the expert retrieval task in unsupervised way, as it may provide insight and inspiration on how to extend supervised GAN-based models in unsupervised ways to tackle other tasks where labelled data is missing. We found that our proposed USGAN outperforms all the state-of-the-art unsupervised expert retrieval algorithms, including both semantic or non-semantic ones.

Looking forward, there are many unexplored avenues. For instance, how to extend the supervised standard GAN and its variants

via unsupervised ways to tackle other tasks where no manually labelled data is available? Can we use implicit signals such as clicks of documents into our USGAN to improve performance of document retrieval? Can we apply our USGAN for expert profiling [6, 8, 30, 41] in enterprise corpora?

REFERENCES

[1] Martin Arjovsky and Léon Bottou. 2017. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862* (2017).

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein GAN. *arXiv preprint arXiv:1701.07875* (2017).

[3] Peter Bailey, Arjen P De Vries, Nick Craswell, and Ian Soboroff. 2007. Overview of the TREC 2007 Enterprise Track. In *Proceedings of Text REtrieval Conference*. 1–7.

[4] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. 2006. Formal Models for Expert Finding in Enterprise Corpora. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*. 43–50.

[5] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. 2009. A language modeling framework for expert finding. *Information Processing & Management* 45, 1 (2009), 1–19.

[6] Krisztian Balog, Toine Bogers, Leif Azzopardi, Maarten de Rijke, and Antal van den Bosch. 2007. Broad Expertise Retrieval in Sparse Data Environments. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*. 551–558.

[7] Krisztian Balog, Maarten de Rijke, and Wouter Weerkamp. 2008. Bloggers As Experts: Feed Distillation Using Expert Retrieval Models. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*. 753–754.

[8] Krisztian Balog, Yi Fang, Maarten de Rijke, Pavel Serdyukov, and Luo Si. 2012. Expertise Retrieval. *Found. Trends Inf. Retr.* 6 (2012), 127–256.

[9] Krisztian Balog, Ian Soboroff, Paul Thomas, Peter Bailey, Nick Craswell, and Arjen P. de Vries. 2008. Overview of the TREC 2008 Enterprise Track. In *Proceedings of Text REtrieval Conference*. 1–12.

[10] Richard Berendsen, Maarten Rijke, Krisztian Balog, Toine Bogers, and Antal Bosch. 2013. On the assessment of expertise profiles. *Journal of the Association for Information Science and Technology* 64, 10 (2013), 2024–2044.

[11] David Berthelot, Tom Schumm, and Luke Metz. 2017. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717* (2017).

[12] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.* 3 (2003), 993–1022.

[13] Zheqian Chen, Ben Gao, Huimin Zhang, Zhou Zhao, Haifeng Liu, and Deng Cai. 2017. User Personalized Satisfaction Prediction via Multiple Instance Deep Learning. In *Proceedings of the International World Wide Web Conference*. 907–915.

[14] Nick Craswell, Arjen P. de Vries, and Ian Soboroff. 2005. Overview of the TREC 2005 Enterprise Track. In *Proceedings of Text REtrieval Conference*. 1–7.

[15] W Bruce Croft, Donald Metzler, and Trevor Strohman. 2015. *Search engines: Information retrieval in practice*. Addison-Wesley Reading.

[16] Gianluca Demartini, Julien Gaugaz, and Wolfgang Nejdl. 2009. A Vector Space Model for Ranking Entities and Its Application to Expert Search.. In *Proceedings of the European Conference on Information Retrieval*. 189–201.

[17] Hui Fang and ChengXiang Zhai. 2007. Probabilistic models for expert finding. In *Proceedings of the European Conference on Information Retrieval*. 418–430.

[18] Yi Fang, Luo Si, and Aditya P. Mathur. 2010. Discriminative Models of Integrating Document Evidence and Document-Candidate Associations for Expert Search. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*. 683–690.

[19] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

[20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Proceedings of the Conference on Neural Information Processing Systems*. 2672–2680.

[21] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A Deep Relevance Matching Model for Ad-hoc Retrieval. In *Proceedings of the ACM International on Information and Knowledge Management*. 55–64.

[22] Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*. 50–57.

[23] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning Deep Structured Semantic Models for Web Search Using Clickthrough Data. In *Proceedings of the ACM International on Information and Knowledge Management*. 2333–2338.

[24] Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. Position-Aware Representations for Relevance Matching in Neural Information Retrieval. In *Proceedings of the International World Wide Web Conference*. 799–800.

[25] Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *Proceedings of the International Conference on Learning Representations*. 1–13.

[26] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* 20, 4 (2002), 422–446.

[27] Justin Johnson, Agrim Gupta, and Li Fei-Fei. 2018. Image Generation from Scene Graphs. *arXiv preprint arXiv:1804.01622* (2018).

[28] Viveka Kulharia, Arnab Ghosh, Amitabha Mukerjee, Vinay Namboodiri, and Mohit Bansal. 2017. Contextual RNN-GANs for abstract reasoning diagram generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 2852–2858.

[29] Hang Li and Jun Xu. 2014. Semantic Matching in Search. *Found. Trends Inf. Retr.* 7, 5 (2014), 343–469.

[30] Shangsong Liang. 2019. Collaborative, Dynamic and Diversified User Profiling. In *AAAI*.

[31] Shangsong Liang, Fei Cai, Zhaochun Ren, and Maarten de Rijke. 2016. Efficient Structured Learning for Personalized Diversification. *IEEE Transactions on Knowledge and Data Engineering* 28, 11 (2016), 2958–2973.

[32] Shangsong Liang and Maarten de Rijke. 2013. Finding knowledgeable groups in enterprise corpora. In *SIGIR*. 1005–1008.

[33] Shangsong Liang and Maarten de Rijke. 2015. Burst-aware data fusion for microblog search. *Information Processing & Management* (2015), 89–113.

[34] Shangsong Liang and Maarten de Rijke. 2016. Formal language models for finding groups of experts. *Information Processing & Management* (2016), 529–549.

[35] Shangsong Liang, Maarten de Rijke, and Manos Tsagkias. 2013. Late Data Fusion for Microblog Search. In *Proceedings of the European Conference on Information Retrieval*. 743–746.

[36] S. Liang, Z. Ren, and M. de Rijke. 2014. Fusion helps diversification. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[37] Shangsong Liang, Zhaochun Ren, and Maarten de Rijke. 2014. Personalized Search Result Diversification via Structured Learning. In *KDD*. 751–760.

[38] Shangsong Liang, Zhaochun Ren, Yukun Zhao, Jun Ma, Emine Yilmaz, and Maarten De Rijke. 2017. Inferring Dynamic User Interests in Streams of Short Texts for User Clustering. *ACM Trans. Inf. Syst.* 36, 1 (July 2017), 10:1–10:37.

[39] Shangsong Liang, Emine Yilmaz, and Evangelos Kanoulas. 2016. Dynamic Clustering of Streaming Short Documents. In *KDD*. 995–1004.

[40] Shangsong Liang, Emine Yilmaz, and Evangelos Kanoulas. 2019. Collaboratively Tracking Interests for User Clustering in Streams of Short Texts. *IEEE Transactions on Knowledge and Data Engineering* 31, 2 (2019), 257–272.

[41] Shangsong Liang, Xiangliang Zhang, Zhaochun Ren, and Evangelos Kanoulas. 2018. Dynamic Embeddings for User Profiling in Twitter. In *KDD*. ACM, 1764–1773.

[42] Francesco Locatello, Damien Vincent, Ilya Tolstikhin, Gunnar Rätsch, Sylvain Gelly, and Bernhard Schölkopf. 2018. Clustering Meets Implicit Generative Models. *arXiv preprint arXiv:1804.11130* (2018).

[43] Genevieve B Orr and Klaus-Robert Müller. 2012. *Neural networks: tricks of the trade (second edition)*. Springer.

[44] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing*. 1532–1543.

[45] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. 2004. The author-topic model for authors and documents. In *UAI*. 487–494.

[46] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training GANs. In *Proceedings of the Conference on Neural Information Processing Systems*. 2234–2242.

[47] Othman Sbai, Mohamed Elhoseiny, Antoine Bordes, Yann LeCun, and Camille Couprie. 2018. DeSIGN: Design Inspiration from Generative Networks. *arXiv preprint arXiv:1804.00921* (2018).

[48] Ian Soboroff, Arjen P de Vries, and Nick Craswell. 2006. Overview of the TREC 2006 Enterprise Track. In *TREC'06*. 1–20.

[49] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the Conference on Neural Information Processing Systems*. 1057–1063.

[50] Christophe Van Gysel, Maarten de Rijke, and Marcel Worring. 2016. Unsupervised, efficient and semantic expertise retrieval. In *Proceedings of the International World Wide Web Conference*. ACM, 1069–1079.

[51] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*. 515–524.

[52] Wei Wei, Gao Cong, Chuanyan Miao, Feida Zhu, and Guohui Li. 2016. Learning to Find Topic Experts in Twitter via Different Relations. *IEEE Transactions on Knowledge and Data Engineering* 28, 7 (2016), 1764–1778.

[53] Xing Wei and W. Bruce Croft. 2006. LDA-based document models for ad-hoc retrieval. In *Proceedings of the International ACM SIGIR Conference on Research*

and Development in Information Retrieval. 178–185.

[54] Ronald J. Williams. 1992. *Simple statistical gradient-following algorithms for connectionist reinforcement learning*. Springer.

[55] Zhe Xu and Jay Ramanathan. 2016. Thread-based probabilistic models for expert finding in enterprise Microblogs. *Expert Systems with Applications* (2016), 286–297.

[56] Xing Yi and James Allan. 2009. A Comparative Study of Utilizing Topic Models for Information Retrieval. In *Proceedings of the European Conference on Information Retrieval*. 29–41.

[57] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 2852–2858.

[58] Junbo Zhao, Michael Mathieu, and Yann LeCun. 2017. Energy-based generative adversarial network. In *Proceedings of the International Conference on Learning Representations*. 1–10.

Estimating the Total Volume of Queries to Google

Fabrizio Lillo
University of Bologna
Bologna, Italy
fabrizio.lillo@unibo.it

Salvatore Ruggieri
University of Pisa
Pisa, Italy
salvatore.ruggieri@unipi.it

ABSTRACT

We study the problem of estimating the total volume of queries of a specific domain, which were submitted to the Google search engine in a given time period. Our statistical model assumes a Zipf's law distribution of the population in the reference domain, and a non-uniform or noisy sampling of queries. Parameters of the distribution are estimated using nonlinear least square regression. Estimations with errors are then derived for the total number of queries and for the total number of searches (volume). We apply the method on the *recipes and cooking* domain, where a sample of queries is collected by crawling popular Italian websites specialized on this domain. The relative volumes of queries in the sample are computed using Google Trends, and transformed to absolute frequencies after estimating a scaling factor. Our model estimates that the volume of Italian recipes and cooking queries submitted to Google in 2017 and with at least 10 monthly searches consists of 7.2B searches.

CCS CONCEPTS

• **Information systems → Web search engines**; • **Mathematics of computing → Probabilistic inference problems**.

KEYWORDS

Search engine query; Volume estimation; Zipf's law; Google Trends

ACM Reference Format:
Fabrizio Lillo and Salvatore Ruggieri. 2019. Estimating the Total Volume of Queries to Google. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313535

1 INTRODUCTION

The problem of computing the total number of searches (volume) of queries belonging to a specific domain is extremely relevant and, at the same time, challenging. From a business perspective, the total volume $\mathcal{V}$ of queries quantifies the potential market of search engine advertising in the domain. An even more interesting quantity is the total volume $\mathcal{V}_v$ of queries searched at least v times. $\mathcal{V}_v$ quantifies the potential market of queries worth to bid on. Related to the above, the total number of queries N in the domain, or of queries N_v searched at least v times, are also gold nuggets. However, the stream of queries submitted to a search engine is so massive that it is impractical to keep frequency counts of every possible query, particularly of those in the long tail of the distribution.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313535

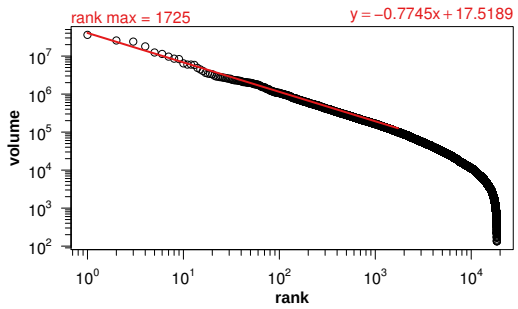


Figure 1: Empirical rank-volume distribution (scaled Google Trends estimates). Best view in color.

Here we study the problem of estimating the total volume of queries submitted to the Google search engine for a specific domain in a given time period. While our method is in principle general, in this paper we apply it to data in the domain of *recipes and cooking*. Such a domain consists of queries with the name of the recipe of a dish, excluding drinks. The advantage over other domains is that it is relatively easy to collect sample recipes and to validate whether a given text is a recipe or not. In particular, we crawled popular websites of Italian recipes and cooking, collecting a sample of more than 120K queries. We then resorted to Search Engine Optimization (SEO) tools, and in particular to Google Trends[1], for obtaining estimates of the volume of each query in the sample for the whole year 2017.

The motivation for the model adopted in this paper comes from the evidence of Figure 1, which shows the empirical rank-volume distribution obtained using estimates of Google Trends. Actually, Google Trends provides relative volumes, not absolute frequencies, thus to find absolute volumes we need to estimate an appropriate scaling factor. This is done by correlating relative volume with ground truth continuous data. We rely on query impression summaries provided by the Google Search Console of a top-ranked website. Indeed Figure 1 reports absolute volumes obtained by rescaling relative ones. The most difficult task in our problem is to estimate the volume of the queries in the population which do *not* belong to the empirical sample. For this reason we do a precise statistical assumption on the rank-volume distribution of the whole population of queries (i.e. observed and unobserved). Our statistical model assumes a Zipf's law distribution of the population, as suggested by the empirical distribution of Figure 1 and previous related work [16]. In order to cope with computational issues, SEO tools may adopt sampling strategies and/or approximated counting techniques, e.g., *count-min sketch* summaries [6, 8], that favor volume

[1] https://trends.google.com

estimation of popular queries against the ones in the long tail of the distribution. This yields the visible drop in volume in the tail of the empirical distribution of Figure 1, with only 18.5K queries being assigned a non-zero volume estimate by Google Trends. We are able to model this behavior by assuming that empirical sampling from the population is not uniform, but it depends on the true rank of a query (*non-uniform* sampling). Moreover, in order to account for approximations in the SEO tool data, we additionally assume that the estimates are noisy, and discuss two specific sampling schemes (*noisy* and *sketchy* sampling). Parameters of the Zipf distribution are estimated using Nonlinear Least Square (NLS) regression. Simulations show such estimators perform better than an alternative approach based on Power law parameter estimation. We derive then estimators of total volumes $\mathcal{V}$ and $\mathcal{V}_v$, and total number of queries N and N_v, including closed formula for statistical errors of such estimators.

In summary, this paper makes the following contributions:

- we formalize the problem of estimating the total volume of queries submitted to a search engine, propose a statistical model which is consistent with empirical data, and infer parameters of the statistical model that perform well under simulated conditions;
- we design a procedure for estimating relative volumes of a set of queries that overcomes the rounding error introduced by Google Trends, and devise a statistical model for scaling relative volumes to absolute ones starting from ground truth SEO data;
- we apply the approach to the domain of recipes and cooking for queries in Italian, and produce estimations for the volume $\mathcal{V}_v$ of queries searched at least v times in 2017.

This paper is organized as follows. First, we report on related work in Section 2. Next, Section 3 states the main problem by modelling the rank-volume distribution of queries as Zipf's law. Section 4 first discusses the impact of non-uniform sampling from a Zipf's law, which is consistent with empirical data. Then, estimators of the parameters of the Zipf's law are introduced, and adopted for estimating the number and total volume of queries in the population. Section 5 describes the approximation introduced by computing relative volumes from Google Trends data, and presents a statistical model for scaling relative volumes to absolute ones. Section 6 describes the available empirical data obtained by collecting Google Trends relative volumes, and applies the scaling method of Section 5 and the estimators of Section 4 to the empirical data. Conclusions summarize the contribution of the paper.

2 RELATED WORK

Pareto distributions and Zipf's laws are ubiquitous in empirical data of many fields [5, 14], and in information retrieval in particular [16]. Several works [2, 3, 10, 16] have observed that the probability that a query is searched v times in a query log is approximately Power law distributed, namely $P(V = v) \propto 1/v^{\alpha}$. This implies (see e.g., [1, 4]) that the the probability that a query is ranked i-th follows a Zipf's law $P(R = i) \propto 1/i^{\beta}$ for $\beta = 1/(\alpha - 1)$. This information on query frequencies/ranks has been used to optimize caching and distribution strategies in search engines and peer-to-peer systems.

There is a huge literature on the estimation of parameters of Power law distributions and Zipf's law. Popular methods [16] have relied on: graphical methods, straight-line approximation, maximum-likelihood estimation. The estimated tail exponent, even in simulated data, significantly depends on the adopted method [12]. A major breakthrough was the method proposed in [5], which consists in a maximum-likelihood estimation, with a cutoff for the fitting region determined with a Kolmogorov-Smirnov test. This method is implemented in the `powerLaw` package [11] of R, which we used extensively in our analyses.

A related stream of literature considers the *unseen species* problem. As originally stated, the problem asks how many biological species are present in a region, given that in an observation campaign a certain number of species with their relative frequency have been observed. In our case, the problem is that we have (noisy) estimates of the frequency of a certain number of queries, and we want to estimate the number of unobserved queries and their frequency. Despite there are several estimators for the unseen species problem (for example, the Good-Toulmin estimator and its extensions [15]), the problem tackled here is different in an important aspect. In the unseen species problem, it is often assumed that in the sample used to build the estimator, the observed frequencies are proportional to the true frequencies in the population. In other words, there is no bias in the construction of the sample. In our approach, the elements of the sample are chosen *ex-ante* and not necessarily the probability of being in the sample is proportional to true frequency.

Google Trends has been widely used for correlating search trends with offline indicators of economic activity, business performance, disease spreading, brand value and awareness, box-office revenue and audience, stock market variability, etc. [17] presents a brief review of the literature. To the best of our knowledge, all works make use of relative volumes only. Their conclusions are stated in relative terms, such as increase/decrease of a searched topic. Here, we first attempt at determining absolute volumes of sample queries, and at inferring how they aggregate over all queries in a domain.

In general, there is little documentation on how SEO tools collect query logs for providing estimates of search frequencies. Google Trends and Google AdWords can rely on Google search engine logs. Similarly for services provided by other search engines. Independent SEO tools (Searchvolume.io, Ubersuggest, Semrush, Keywordkeg, etc.) rely on a more limited user base. [17] compares Google Trends and Baidu Index (restricted to searches from China only), and finds that their estimates are highly correlated. An advantage of Baidu Index over Google Trends is that it provides absolute estimates, not relative ones. For reference domains restricted to searches from China, by using Baidu Index instead of Google Trends, one would save the task of scaling relative to absolute volumes described in Section 5.

3 PROBLEM STATEMENT

Let us assume the population of queries in the reference domain is composed by N queries, and that the rank-volume distribution of such population follows a Zipf's law. Formally, the volume V_i of the i-th most popular query q_i, for $i \in [1, N]$, is:

$$V_i = \frac{c}{i^{\beta}}. \tag{1}$$

The parameters c and β are called the *intercept* and the *coefficient* respectively. The total volume over the population is thus:

$$\mathcal{V} \equiv \sum_{i=1}^{N} \frac{c}{i^{\beta}} = c[\zeta(\beta) - \zeta(\beta, N+1)] \tag{2}$$

where $\zeta(x)$ and $\zeta(x, y)$ are the Riemann zeta and Hurwitz functions, respectively. If N, c, and β are known one can easily determine $\mathcal{V}$. As discussed in the introduction, however, there are several reasons that make this impossible in practice. The problem that we investigate in this paper consists of estimating $\mathcal{V}$ starting from an empirical sample of volumes $v_1, \ldots, v_n$, for $n < N$ sample queries. Without any loss of generality, we assume that the observations are ranked, namely $v_1 \geq v_2 \geq \ldots \geq v_n$.

The problem can be decomposed in two parts: (1) since true absolute volumes V_i are not observable, even for the subset of n queries, we propose a method for estimating them; (2) having a possibly noisy estimation v_i for V_i in a possibly non-uniform sample subset, we consider the problem of estimating the total query volume $\mathcal{V}$, including also the volume of the unobserved queries. Problem (2) is tackled first in the next section, while problem (1) is discussed in Section 5.

4 MODELLING AND ESTIMATION

4.1 Sampling from a Zipf

Starting from the assumption that the volume of the query population follows a Zipf distribution (see Eq. 1), we observe that the empirical distribution in Figure 1 shows a drop of volume in its tail. We intend here to investigate on this. We will consider the effects of different sampling methods from a Zipf's law, and check whether the conclusions are consistent with our empirical data.

Clearly, uniform sampling from a Zipf's law cannot explain the drop of volume in the tail of the empirical distribution. In fact, queries in an empirical sample are rarely chosen uniformly. The approach followed in our reference domain, for instance, relies on collecting recipe names from specialized websites. These typically conduct a keyword research effort in targeting high-volume keywords. As a consequence, our empirical data suffers from an unavoidable selection bias in favor of high-volume queries. A similar bias against very low volume queries is introduced by SEO tools (e.g., Google Trends) used to obtain volume estimates of queries in a sample. In summary, our empirical data is likely to be a non-uniform sampling of the query population. We assume here that sampling depends on the true rank, and call this *non-uniform sampling*. Formally, we assume that the i-th query q_i is sampled with a probability $p(i)$. We want to check whether the observed rank plot obtained by a sample of the population is different from a Zipf's law. To this end, we consider a geometric sampling $p(i) \propto p(1-p)^{i-1}$, i.e., the sampling probability decays exponentially with the rank. For example, if $p = 0.01$, the probability that the query with the largest volume in the population is observed is p, then the second, third, fourth, etc. query in terms of volume will be observed (i.e. sampled) with probability $0.99p$, 0.99^2p, 0.99^3p, etc. Figure 2 shows a numerical simulation with the following parameters[2] of

[2]The choice of β, in particular, has been driven by the empirical distribution of Figure 1.

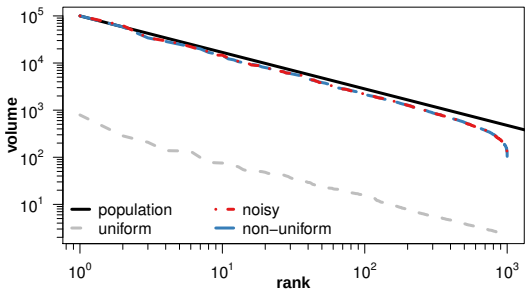


Figure 2: Simulation of sampling from a Zipf's law.

the population:

$$N = 10^6, \quad c = 10^5, \quad \beta = 0.7745. \tag{3}$$

Samples consist of $n = 1000$ queries, and $p = 0.001$ is set for the geometric sampling. The black line is the whole population, the blue line is obtained with geometric sampling while the grey line is obtained with uniform sampling. The non-uniform sampling is consistent with the tail of the empirical distribution in Figure 1.

As a second aspect worth to be considered, we have mentioned that SEO tools typically provide approximated values of the true volume of queries, due to their sampling strategy and computational heuristics in frequency counting. Another source of approximation will be discussed in Section 5.1. Therefore, our empirical data is drawn from noisy values X_i's of the true V_i's. We assume that:

$$X_i = V_i \epsilon_i$$

where ϵ_i are independent noise with common distribution characterized by the same mean μ and variance σ_i^2. Clearly, the presence of noise scrambles the frequencies, thus the most frequent according to X is not necessarily the most frequent according to V. Figure 2 includes also a noisy and non-uniform sample (red line) generated assuming ϵ_i normally distributed, but truncated to 0 to avoid negative V_i's. Parameters are set as follows: $\mu = 1$, i.e., noise is unbiased, and $\sigma_i^2 = 0.01/9$, i.e., 99.7% of noise is in the range $\pm 3\sigma = \pm 10\%$ of the true value. Noisy and non-uniform sampling (hereafter *noisy sampling*) produces an empirical distribution very close to the one of non-uniform sampling and that is also consistent with our empirical data.

A yet another way to model computationally approximated counting, as provided by count-min sketches [6, 8], is to set:

$$X_i = V_i + \gamma_i c$$

where γ_i is uniformly distributed in the range $[0, \gamma]$. In such case, the noise overestimates V_i up to a fraction γ of the top volume $V_1 = c/1^{\beta} = c$. For low volumes, the noise may considerably increase the observed value. However, for a sufficiently low γ, the non-uniform sampling alleviates from this problem, since low volumes are sampled with low probability. We set $\gamma = 0.001$ in simulations. The empirical distribution generated lies in between the ones of non-uniform and noisy sampling. For readability reasons, it is not shown in Figure 2. We call such model the sketchy and non-uniform sampling, hereafter *sketchy sampling*.

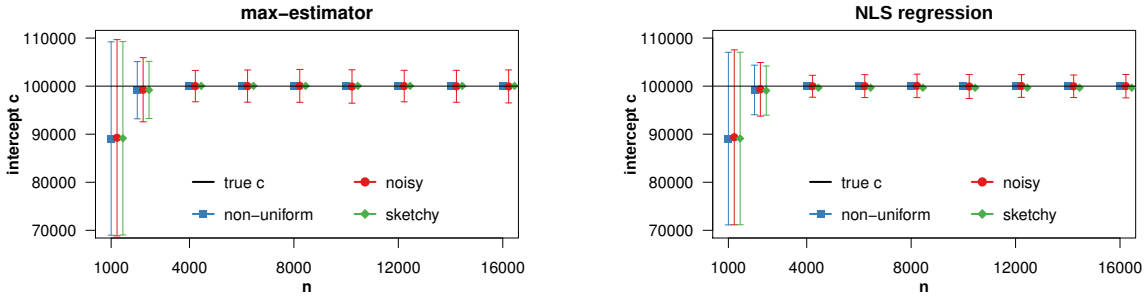


Figure 3: Simulations on estimation of c: error bars (mean ± stdev).

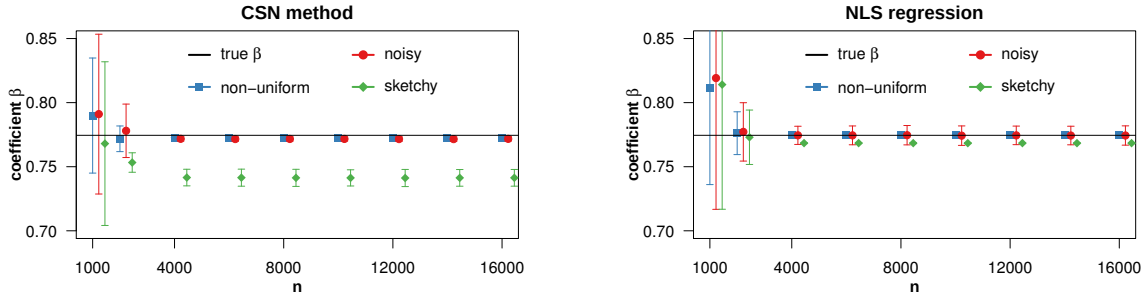


Figure 4: Simulations on estimation of β: error bars (mean ± stdev).

4.2 Estimating β and c

We consider now the estimation of the coefficient β and intercept c in Eq. 1 by exploring two alternative methods for each of them. Regarding the coefficient, we observe that β is the same coefficient of the p.d.f. of the continuous Zipf's law:

$$p_i = \frac{1}{\zeta(\beta)\ i^{\beta}}$$

Thus, we can use the well-known method of Clauset, Shalizi and Newman [5] (hereafter, the CSN method) for estimating the β parameter in Eq. 1. Strictly speaking, [5] is a maximum-likelihood estimator $\hat{\alpha}$ of the α exponent of the Power law of volume distribution, $P(V = v) \propto 1/v^{\alpha}$, from the high-volume tail of empirical data $v_{max} \leq \ldots \leq v_1$. Since in many empirical data the Power law tail is observed only for a range of values, [5] uses a Kolmogorv-Smirnov like test to determine v_{max} which is the optimal value after which the distribution is Power law tailed. Using the well-known relation $\beta = 1/(\alpha - 1)$ between exponents of Power law and continuous Zipf's law (see [1, 4]), we obtain the estimate $\hat{\beta} = 1/(\hat{\alpha} - 1)$ of the coefficient of the rank-volume distribution for top ranks 1 to *max*. The theoretical advantage of this method is that it automatically selects the rank *max* from which to regress the coefficient.

The second estimator of β is to use standard Nonlinear Least Square (NLS) regression of the volume V_i from the rank i. This means that the parameters c and β are those minimizing the sum of squares: $\sum_{i=1}^{M} \left(V_i - \frac{c}{i^{\beta}}\right)^2$, where M is the maximal rank considered in the regression[3]. Since the empirical data follows such distribution for the top ranks, we regress only the top $M = max$ rank-volume data, where *max* is the rank returned by the CSN method. NLS has two advantages over CSN. First, intercept c and coefficient β are estimated together in the same procedure. Second, the regression directly estimates β, while in the CSN method β is estimated with a formula involving the estimator of α. Finally, the second estimator of the intercept that we consider here is the maximum observed value, namely v_1. We call it the *max-estimator* of c. This is motivated by observing that $V_1 = c/1^{\beta} = c$, namely the intercept c is the volume of the top ranked query in the population.

Let us now investigate how these estimators are affected by the non-uniform, noisy, and sketchy sampling from a Zipf's law. Numerical simulations with parameters as in (3), are repeated at the variation of the sample size n for 1000 times and results averaged.

Figure 3 shows that both the max-estimator and the NLS regression converge to the true value of the intercept c. For noisy data, however, there is some error, which is proportional to the noise level (set to ±10%). Variability is slightly lower for NLS regression. Larger error bars can be observed for small values of n. They are due to the chances of not having the highest volume of the population included in the sample. This chance is controlled by the $p = 0.001$ parameter in the geometric sampling. Smaller values lead to larger standard deviation, and, symmetrically, larger values to smaller standard deviation. Thus, in practical settings, the selection of the sample queries must carefully consider the issue of including the most popular queries in the empirical sample. This has been

[3]NLS regression requires to specify initial values for β and c to start with. We compute them using ordinary (linear) least squares (OLS) of the log, i.e. minimizing $\sum(\log V_i - \log c - \beta \log i)^2$. This method cannot be used as an alternative to NLS since it gives too much importance to deviations of low rank queries with respect to high rank queries.

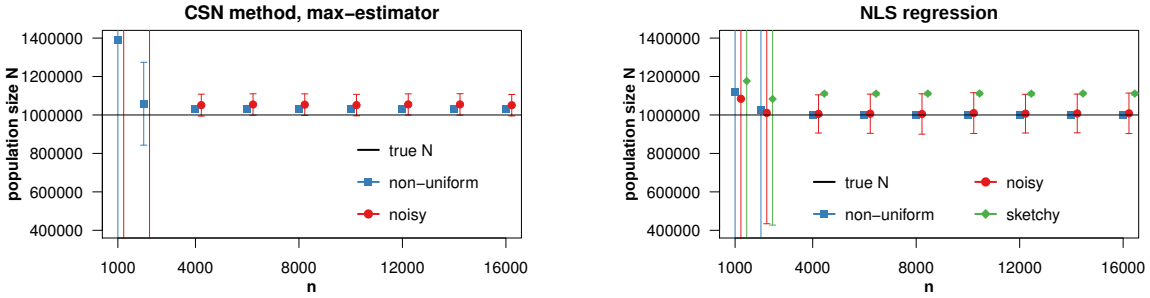


Figure 5: Simulations on estimation of the population size N: error bars (mean ± stdev).

one of our main concerns in collecting queries in the recipe and cooking domain.

Figure 4 shows some differences in the estimation of β. Regarding the CSN method, the estimated values for non-uniform and noisy samplings are slighly lower than the true β. Underestimation in the sketchy sampling case is, instead, considerable. Regarding the NLS regression, it is unbiased for non-uniform and noisy sampling. For sketchy sampling, β is slightly underestimated. Estimations rapidly converge for increasing n's, except for noisy sampling in the case of NLS, and for sketchy sampling in the case of CSN.

Finally, all estimations are weakly dependent on n: starting for samples of 0.4% of the population, they become stable.

4.3 Estimating N

In the following, we will focus on a simple but effective estimator of the size N of the query population. We assume to know V_N, namely the smallest volume of a query in the population. This assumption is realistic for absolute frequencies, since $V_N \simeq 1$. From Eq. 1, for $i = N$, we have $N = (c/V_N)^{1/\beta}$. This motivates the following estimator:

$$\hat{N} = \left(\frac{\hat{c}}{V_N}\right)^{\frac{1}{\hat{\beta}}} \tag{4}$$

where $\hat{c}$ is an estimator of c and $\hat{\beta}$ is an estimator of β. Eq. 4 can be extended to an estimator of the number of queries whose volume is greater or equal than a given value v as:

$$\hat{N}_v = (\hat{c}/v)^{1/\hat{\beta}} \tag{5}$$

Numerical simulations with parameters as in (3) are shown in Figure 5 for: (1) $\hat{\beta}$ obtained by the CSN method and $\hat{c}$ obtained by the max-estimator; and (2) $\hat{\beta}$ and $\hat{c}$ obtained by NLS regression. The first method is biased, showing a slight overestimation for non-uniform and noisy sampling and a large overestimation for sketchy sampling (not shown because exceeding the y-axis limits). The second method converges to the true value of N for non-uniform and noisy sampling (on average), and it slightly overestimates it for sketchy sampling. These findings are intuitive. They follow from Eq. 4, by observing that, if $\hat{c}$ is unbiased (as shown in Figure 3), then the estimator $\hat{N}$ has a bias proportional to the power of $1/\hat{\beta}$. We know from Figure 4 that $\hat{\beta}$ underestimates β for the CSN method or for the sketchy sampling. The only advantage of first method over the second one, is a smaller variability of the estimates in the case of noisy sampling. Again, this is a direct consequence of the smaller variability of β estimates (see Figure 4).

4.4 Estimating $\mathcal{V}$

Building on the estimators and simulations conducted so far, the proposed procedure for estimating the total volume $\mathcal{V}$ is composed of the following steps:

- estimate β and c, as described in Section 4.2;
- use the estimated $\hat{\beta}$ and $\hat{c}$ as inputs for estimating N as shown in Section 4.3;
- the estimator of $\mathcal{V}$ is obtained from Eq. 2 as follows:

$$\hat{\mathcal{V}} = \hat{c}[\zeta(\hat{\beta}) - \zeta(\hat{\beta}, \hat{N} + 1)]$$

Notice that by Eq. 4, the estimator $\hat{\mathcal{V}}$ can be stated using only $\hat{\beta}$ and $\hat{c}$:

$$\hat{\mathcal{V}} = \hat{c}[\zeta(\hat{\beta}) - \zeta(\hat{\beta}, \left(\frac{\hat{c}}{V_N}\right)^{\frac{1}{\hat{\beta}}} + 1)] \tag{6}$$

These estimators can be generalized to estimators of the total volumes of queries with minimum volume v by replacing V_N by v:

$$\hat{\mathcal{V}}_v = \hat{c}[\zeta(\hat{\beta}) - \zeta(\hat{\beta}, (\hat{c}/v)^{1/\hat{\beta}} + 1)] \tag{7}$$

Let us continue the previous numerical simulations. With the settings in (3), it turns out $\mathcal{V} = 9,609,224$. First consider using the NLS regression method in the first step of the procedure. Figure 6 shows that $\hat{\mathcal{V}}$ converges to $\mathcal{V}$ for non-uniform and noisy sampling, and overestimates it for sketchy sampling. For noisy sampling, there is some variability, which is in the order of the noise introduced during sampling (±10%). The overestimation in the case of sketchy sampling follows from the overestimation of N (see Figure 5).

Consider now the case of using in the first step of the procedure the CNS method coupled with the max-estimator. The total volume is slightly overestimated for non-uniform sampling and for noisy sampling. In the latter case, there is some variability, which appears lower than for the CSN method. This can be tracked back to lower variability in the estimation of β (see Figure 4). For sketchy sampling, the overestimation is very large: it is out of the bounds of the plot in Figure 6. Again, this can be traced back to a larger underestimation of β compared to the CSN method.

The impact of biased $\hat{\beta}$ on the estimated total volume $\hat{\mathcal{V}}$ can be readily explained when $\hat{c} = c$ – which holds in simulations, as shown in Figure 3. In Figure 7 we plot Eq. 6 as a function of $\hat{\mathcal{V}}$, under the assumption that V_N is known. The left plot shows

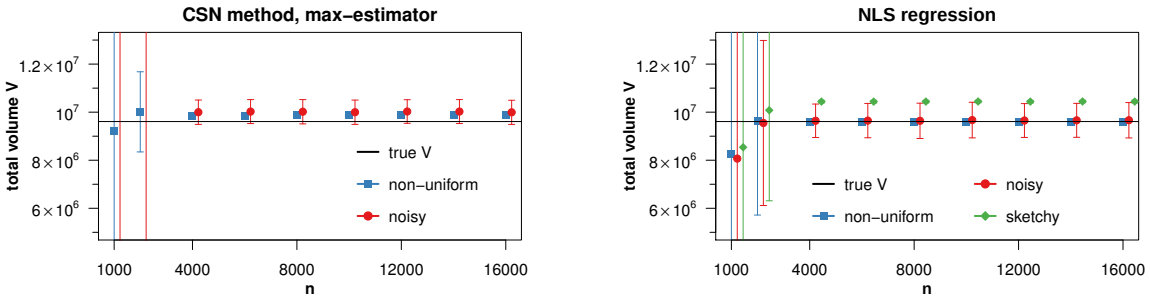


Figure 6: Simulations on the estimation of the total volume $\mathcal{V}$: error bars (mean ± stdev).

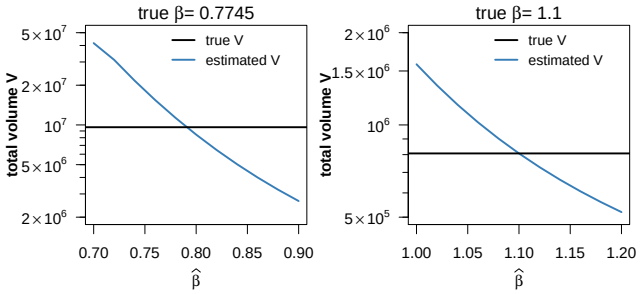


Figure 7: Estimated volume (Eq. 6) as a function of $\hat{\beta}$, assuming $\hat{c} = c$. Simulation parameters: $N = 10^6$, $c = 10^5$, β in title.

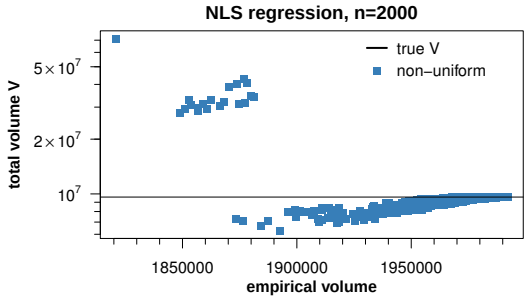


Figure 8: Scatterplot of empirical vs estimated total volume.

simulations for the parameters in (3) used so far. The right plot uses the same N and c, but a β greater than 1. In both cases, the bias of $\hat{\mathcal{V}}$ is inversely proportional to bias of β. Note the log scale in the y-axis, which comes from the fact that β appears as exponent in Eq. 6. For β's lower than 1, error (or variability) of the estimator $\hat{\beta}$ has a greater impact on error (or variability) of $\hat{\mathcal{V}}$ than for β's greater than 1.

In all three sampling models, the performances become stable from $n = 4,000$ on, which is 0.4% of the population. Let us now focus on small sample sizes, for which instead there is a large standard deviation over the experimental runs. Fix $n = 2,000$, and consider NLS regression and uniform sampling. The estimated $\hat{\mathcal{V}}$ is approximately $\mathcal{V} \pm 3.9 \times 10^6$, i.e., the standard deviation is 4 times the (unbiased) average. What is the source of such variability? Figure 8 shows the scatter plot of empirical volume vs estimated total volume over the 1,000 experimental runs. Runs where the generated sample has a low total empirical volume exhibit most of the variability (notice that the y-axis is in logscale). If the total empirical volume is sufficiently large, even small samples converge to the true volume. This reinforces our previous conclusion that, in practical settings, the selection of sample queries must carefully include popular ones, especially for small size samples.

As a summary of the simulations, we therefore recommend using the NLS regression method for estimating c and β, and, using Eqs. 6–7, for estimating $\mathcal{V}$ and $\mathcal{V}_v$.

4.5 Errors on the estimates

We now compute the error on the estimated N obtained from Eq. 4. Using the propagation of errors under the assumption that the errors on β and c are independent, the error on $\hat{N}$ is:

$$\Delta N = \sqrt{\left(\frac{\partial \hat{N}}{\partial \hat{c}}\Delta c\right)^2 + \left(\frac{\partial \hat{N}}{\partial \hat{\beta}}\Delta \beta\right)^2}$$

To have a more conservative estimate of ΔN, taking into account correlations between errors, one can replace the previous formula with the sum of the absolute values:

$$\Delta N = \left|\frac{\partial \hat{N}}{\partial \hat{c}}\Delta c\right| + \left|\frac{\partial \hat{N}}{\partial \hat{\beta}}\Delta \beta\right| \tag{8}$$

The partial derivatives in the previous expressions are:

$$\frac{\partial \hat{N}}{\partial \hat{c}} = \left(\frac{\hat{c}}{V_N}\right)^{\frac{1}{\hat{\beta}}} \frac{1}{\hat{\beta}\hat{c}} = \frac{\hat{N}}{\hat{\beta}\hat{c}}$$

$$\frac{\partial \hat{N}}{\partial \hat{\beta}} = -\left(\frac{\hat{c}}{V_N}\right)^{\frac{1}{\hat{\beta}}} \frac{1}{\hat{\beta}^2} \log\frac{\hat{c}}{V_N} = -\frac{\hat{N}}{\hat{\beta}^2} \log\frac{\hat{c}}{V_N}$$

From these values and the knowledge of Δc and $\Delta \beta$ (obtained from the NLS regression), it is possible to compute ΔN.

The computation of the error on the total volume is a bit more involved. Consider $\hat{\mathcal{V}}$ as a function of $\hat{c}$ and $\hat{\beta}$ (see Eq. 6). To find the error on $\hat{\mathcal{V}}$ we compute its derivatives with respect to $\hat{c}$ and $\hat{\beta}$. We find:

$$\frac{\partial \mathcal{V}}{\partial \hat{c}} = \frac{\hat{\mathcal{V}}}{\hat{c}} + \hat{N}\zeta\left(\hat{\beta}+1, \hat{N}+1\right)$$

$$\frac{\partial \mathcal{V}}{\partial \hat{\beta}} = \hat{c}\left(\zeta'(\hat{\beta}) - \zeta^{(1,0)}\left(\hat{\beta}, \hat{N}+1\right) - \frac{\hat{N}\log\left(\frac{\hat{c}}{V_N}\right)\zeta\left(\hat{\beta}+1, \hat{N}+1\right)}{\hat{\beta}}\right)$$

where $\zeta'(x)$ is the derivative[4] of the Riemann Zeta function and $\zeta^{(1,0)}(s,a)$ is the partial derivative of the Hurwitz function with respect to s. In summary, the error on the total volume is:

$$\Delta\mathcal{V} = \sqrt{\left(\frac{\partial \hat{\mathcal{V}}}{\partial \hat{c}}\Delta c\right)^2 + \left(\frac{\partial \hat{\mathcal{V}}}{\partial \hat{\beta}}\Delta\beta\right)^2}$$

or, more conservatively:

$$\Delta\mathcal{V} = \left|\frac{\partial \hat{\mathcal{V}}}{\partial \hat{c}}\right|\Delta c + \left|\frac{\partial \hat{\mathcal{V}}}{\partial \hat{\beta}}\right|\Delta\beta \tag{9}$$

5 GOOGLE TRENDS: DATA COLLECTION AND SCALING

Google Trends has several advantages over other SEO tools. First, the volumes provided are computed from the Google search engine query logs, and not from unspecified sources which may have unknown forms of bias. Second, data can be aggregated for arbitrary ranges of time and user agent languages. Most of the other tools, instead, provide monthly averages at the time of request, making it impossible to extend an experiment incrementally to new queries. Third, estimates by Google Trends are ratio-scaled, while other SEO tools provide binned values, i.e., ranges of volumes.

On the negative side, the volume provided by Google Trends is relative, not absolute. We then fix one specific query to the conventional volume of 1, and collect estimates of the volume of any other query in comparison to the specific query. Next, we scale the relative volumes to absolute volumes. In the rest of this section, we discuss some approximation introduced by relative volume calculation, and the scaling from relative to absolute volumes.

5.1 Relative volume calculation

A source of approximation in the calculation of the volume from Google Trends raw data is introduced by the computation of the ratio between the volume of a query q and the volume of the pre-fixed query f. In fact, Google Trends provides $v_f^1, \ldots, v_f^{52}$ relative volumes for f, and $v_q^1, \ldots, v_q^{52}$ relative volumes for q, namely two values for each week in our reference time period (the whole year 2017). The largest value among v_f^i's or v_q^i's is conventionally set to 100, and all the others are integers from 0 to 100 set on the basis of their fraction w.r.t. the largest value (hence, the name *relative* volume). In the following, we assume $v_f^1 = 100$ (similar reasonings apply when $v_q^1 = 100$). We aim at defining an estimator for the ratio:

$$r = \frac{\sum_{i=1}^{52} V_q^i}{\sum_{i=1}^{52} V_f^i} = \frac{\sum_{i=1}^{52} 100 \cdot V_q^i/V_f^1}{\sum_{i=1}^{52} 100 \cdot V_f^i/V_f^1}$$

where V_f^i's and V_q^i's are the true absolute volumes of f and q in the i-th week of the year, and $100 \cdot V_q^i/V_f^1$'s and $100 \cdot V_f^i/V_f^1$'s are the true relative (percentage) volumes of q and f respectively. Intuitively, r is the ratio of the total volume of q over the total volume of f. Recall that v_f^i's and v_q^i's are integer numbers. We further assume they round down[5] the true relative volume, namely $v_q^i = \lfloor 100 \cdot V_q^i/V_f^1 \rfloor$ and $v_f^i = \lfloor 100 \cdot V_f^i/V_f^1 \rfloor$. Bounds for the ratio r above are then:

$$\frac{\sum_{i=1}^{52} v_q^i}{51 + \sum_{i=1}^{52} v_f^i} \le r \le \frac{52 + \sum_{i=1}^{52} v_q^i}{\sum_{i=1}^{52} v_f^i} \tag{10}$$

An estimator for r is the central value between the bounds:

$$\hat{r} = \frac{1}{2}\left(\frac{52 + \sum_{i=1}^{52} v_q^i}{\sum_{i=1}^{52} v_f^i} + \frac{\sum_{i=1}^{52} v_q^i}{51 + \sum_{i=1}^{52} v_f^i}\right)$$

Figure 9 (left panel) shows a numerical simulation consisting of 10,000 repetitions. For each repetition, true relative volumes for 52 weeks are generated for $100 \cdot V_f^i/V_f^1$'s and $100 \cdot V_q^i/V_f^1$'s. Then, they are rounded down to obtain v_f^i's and v_q^i's respectively. Finally, the estimator $\hat{r}$ is compared to the true value r. As a parameter of the simulations, while $100 \cdot V_f^i/V_f^1$'s are uniformly generated in $[0, 100]$ (hence, the assumption $v_f^1 = 100$ is satisfied), the values $100 \cdot V_q^i/V_f^1$'s are randomly generated in $[0, 100]$ with total sum equal to a fraction of $\sum_{i=1}^{52} 100 \cdot V_f^i/V_f^1$, namely so that r is set to a desired value. In particular, Figure 9 (left panel) shows lines for $r = 0.8$ (black line), $r = 0.2$ (blue line), and $r = 0.05$ (red line). For non-small r's, the estimator $\hat{r}$ is very close[6] to the true r on average. Variance is small for larger r's. For small values of r, however, the estimator has a positive bias and very large variance. Intuitively, the weight of rounding in the bounds for r in Eq. 10 becomes considerable. Such weight can be kept low if the volumes of q and f are close to each other. For such a reason, our implementation computing the relative volume of a candidate query q consists of first comparing q with the pre-fixed query f. If their estimated ratio $\hat{r}$ is lower (resp., higher) than a given threshold, then another query f' with lower (resp., higher) relative volume is chosen such that the estimated ratio is within an expected range. The search for f' follows a binary search pattern among all queries whose relative volume has been already estimated. If no f' satisfies the condition, the estimation for q is suspended, until some other query f' will subsequently be estimated that meets the constraint. Finally, the volume of q relative to f is calculated as the product of the volume of q relative to f' multiplied by the volume of f' relative to f. Figure 9 (right panel) shows the benefits of this two-steps procedure against the single step comparing q to f. Here, $r = 0.01$ and f' is chosen with relative volume of 0.2, which is 20 times the one of q. In our actual implementation, we are even more strict. We look for an f' such that the ratio of q relative to f' is is between $4/5 = 0.8$ and $5/4 = 1.25$.

5.2 From relative to absolute volume

We tackle now the problem of transforming the relative volumes computed using Google Trends into absolute frequencies. Basically,

[4] $\zeta'(x)$ is available in R, while $\zeta^{(1,0)}(s,a)$ must be computed numerically or with other tools, e.g., Mathematica.

[5] Rounding to the closest integer is another option, which seems not consistent with the results of Google Trends. E.g., in any week series there is only one estimate of 100. Other estimated values are strictly lower.

[6] Strictly speaking, a t-test at 95% confidence level rejects the hypothesis that the mean of $\hat{r}/r$ is equal to 1. The values are however very close for the whole region of explored parameters.

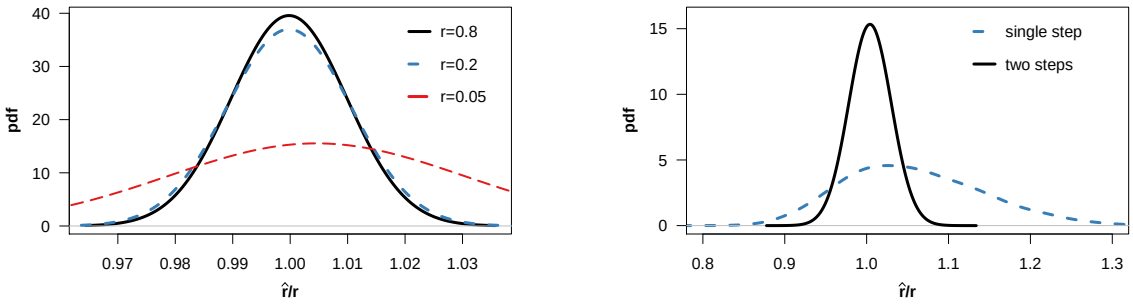


Figure 9: Simulation on estimation $\hat{r}$ of relative volume r of Google Trends. Left: at variation of r. Right: single step vs two steps procedure for $r = 0.01$.

we rely on correlating with absolute volumes provided by an external "ground truth" source. For instance, many SEO tools provide absolute estimations. Commercial tools are supposed to be more reliable than free tools, yet their fees are expensive for a large sample of queries. Moreover, in most cases such tools provide binned absolute estimations. This complicates the statistical correlation model. We consider in this sub-section the case of SEO tools with *continuous* absolute estimations.

As before we consider a sample of n queries, and for the i-th query q_i, let V_i be its true absolute volume. We cannot observe V_i, but we have two related quantities: (1) Google Trends provides a rescaled estimate $X_i = V_i/g$, but g is not known; (2) another SEO tool provides an absolute estimate of the volume of V_i, which we call Y_i. Our objective is to estimate g and therefore the absolute volume V_i. The problem is complicated by two facts: measurements computed from Google Trends are actually estimations of relative volumes (see previous subsection); and, values provided by the other SEO tool are noisy estimates of the true volume. A sensible model taking into account the two sources of noise is:

$$X_i = \frac{V_i}{g\xi_i}, \qquad Y_i = V_i\eta_i$$

where ξ_i are independent and positive error terms due to relative volume estimation[7] and are characterized by a mean value $\mathbb{E}[\xi_i] = \bar{\xi} \simeq 1$ and variance $Var[\xi_i] = z_i^2$. Similarly η_i are independent and positive random variables with mean 1 (i.e., we assume that the other SEO tool is not biased) and variance s_i^2. Note that we are not excluding the possibility that the variance of ξ_i and/or η_i depends on the rank and/or on the volume of a query – since volume of popular queries is easier to estimate. Finally, we will assume that ξ_i and η_j are mutually independent for any i and j. Combining the two expressions we obtain a relation between observable quantities:

$$Y_i = gX_i\xi_i\eta_i \tag{11}$$

Let us consider different estimators of the constant g. In the case of continuous data we compare three of them:

(1) The ratio based estimator defined as:

$$\hat{g}_1 = \frac{1}{n}\sum_{i=1}^{n}\frac{Y_i}{X_i} \tag{12}$$

[7]With the terminology of Section 5.1, ξ_i^{-1} is the rounding error, whose density is shown in Figure 9.

It is easy to show that it has the property:

$$\mathbb{E}[\hat{g}_1] = \bar{\xi}g; \qquad Var[\hat{g}_1] = \frac{g^2}{n^2}\sum_{i=1}^{n}\left(z_i^2s_i^2 + z_i^2 + \bar{\xi}^2s_i^2\right)$$

This estimator has a bias given by $\bar{\xi}$, which can be assumed very small as shown in the previous subsection. Moreover when the sample size $n \to \infty$ it is $Var[\hat{g}_1] \to 0$, i.e. the error on the estimator asymptotically vanishes.

(2) The estimator from the linear regression over the logarithms of the values:

$$\log Y_i = a + b\log X_i + \epsilon_i \tag{13}$$

and then setting $\hat{g}_2 = e^a$.

(3) The estimator from the regression:

$$\log X_i = A + B\log Y_i + \epsilon_i \tag{14}$$

given by $\hat{g}_3 = e^{-A/B}$

In the numerical simulations below we will assume that η_i follows a lognormal distribution, which is positive and has relatively large fluctuations. Given the parameters μ and σ characterizing the lognormal distribution, it is $\mathbb{E}[\eta_i] = \exp(\mu + \sigma^2/2)$ and $Var[\eta_i] = (e^{\sigma^2} - 1)(\mathbb{E}[\eta_i])^2$. In order to have $\mathbb{E}[\eta_i] = 1$, it must be $\mu = -\sigma^2/2$. Moreover we assume for simplicity that $\bar{\xi} = 1$ and $Var[\xi_i] = 0$, i.e. we neglect the approximation error in the calculation of relative volume from Google Trends data. As data generating process, we consider a Zipf's law distribution of the volumes V_i's, with parameters as in (3), and a non-uniform random sampling from it. Also, we consider two noise levels: $\sigma = 0.03$, which leads to a standard deviation of η_i equal to $s \simeq 0.03$; and $\sigma = 0.3$, which leads to $s \simeq 0.31$. Then, we fix $g = 2.75$ and estimate it using the above three estimator for 10,000 runs. Figure 10 shows the densities of the estimated g with the three methods. The ratio based estimator of Eq. 12 performs much better than the ones based on regression and this advantage is larger when the noise term has large variance.

6 EMPIRICAL ANALYSIS

We generated a sample of 120K queries by crawling popular Italian websites about recipes and cooking. The list of websites was compiled with the help of web marketing experts and by looking at

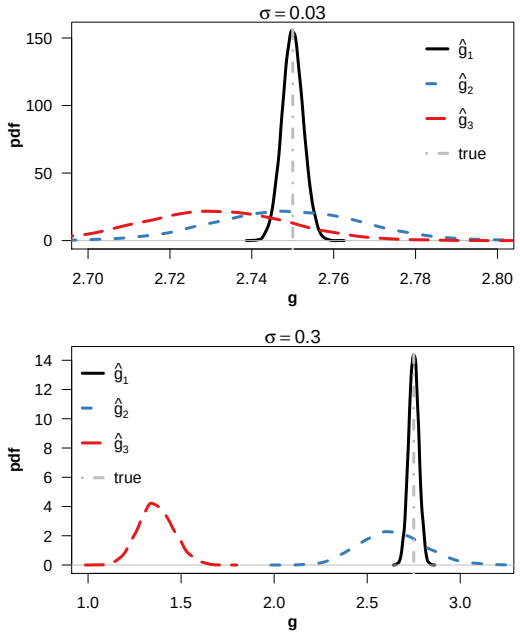


Figure 10: Density estimation of the estimated parameter g. Top plot: small noise. Bottom plot: large noise.

the rankings of SEO tools[8]. We then submitted the 120K queries to a few SEO tools to collect the estimated volume of each query for the reference year 2017 and for Italian user agents. Considering a whole year prevents seasonal bias in data. Query crawling, cleaning, and collection of Google Trends volumes took about 2 months. The process required manual inspection of crawled queries, with a few iterations to correct bugs, to support new hypotheses, etc. Even though the collection of Google Trends data was automated in a script[9], there is a daily bound on the number of invocations to the Google Trends service, which makes such a step time-consuming.

6.1 Google Trends with absolute volumes

We obtained non-zero estimates by Google Trends for about 18.5K queries out of the 120K in the sample. The resulting rank-volume distribution is shown in Figure 1. The remaining queries belong to the long tail, for which Google Trends returns a relative (hence, absolute) estimated volume of zero.

The estimators of the scaling factor g discussed in Section 5.2 require that the absolute estimates provided by an external ground truth are not biased. This assumption cannot be verified in general, e.g., SEO tools do not disclose sufficient information due to IPR restrictions. Since the bias of such tools is unknown, the choice of which one to use for estimating g relies only on the trustworthiness on one specific tool over the others.

Google Search Console[10] (GSC) is a tool that provides to website owners (a.k.a., publishers) summary statistics about the number of impressions and the ranking of the website in Search Engine Result Pages (SERPs). We considered a specific website for which

[8]E.g., https://serpstat.com
[9]We used PyTrends APIs (https://github.com/GeneralMills/pytrends).
[10]https://www.google.com/webmasters

v	$v/12$	$\hat{N}_v$	ΔN_v	$\hat{\mathcal{V}}_v$	$\Delta\mathcal{V}_v$
12	1	269,214,520	± 18,507,467	14,169.58 M	± 827.70 M
120	10	13,770,732	± 815,062	7,171.15 M	± 353.96 M
1,200	100	704,394	± 33,959	3,591.35 M	± 145.83 M
12,000	1,000	36,031	± 1,444	1,760.23 M	± 56.86 M
120,000	10,000	1,843	± 56	823.63 M	± 20.30 M
600,000	50,000	231	± 5	457.12 M	± 9.06 M

Table 1: Estimated N_v and $\mathcal{V}_v$ for queries with at least v searches in 2017. $v/12$ is the monthly average of v.

we had access to its GSC statistics. The website ranked about first in 2017 for 41 queries belonging to our sample. For such queries, the absolute volume is then equivalent to the number of impressions reported by GSC. In summary, we have ground truth volumes (or very close to it) for such set of queries. Using the estimator of Eq. 12, we found $\hat{g}_1 = 6,466.6$, that is the pre-fixed query with relative volume 1 was searched 6,466.6 times in the whole 2017 year, i.e., an average of 538.9 times per month. Figure 1 shows the rank-volume distribution where relative volume X_i has been scaled to $V_i = X_i \cdot \hat{g}_1$.

A drawback of using GSC is the low number of ground truth queries, only 41. As a second option, we consider the well-recognized SemRush[11] tool. We were able to collect volume estimates for 1,688 queries in our sample, using the paid service version of the tool. The resulting estimated scaling factor $\hat{g}_1 = 6,114$ is very close to the one obtained from GSC data.

6.2 Estimation of total query volume

Let us now apply the estimation model designed in Section 4 to the empirical data of Google Trends volumes scaled using GSC data. As shown by the red line fit in Figure 1 (left panel), the NLS regression estimates[12] are:

$$\hat{\beta} = 0.7745085 \quad \hat{c} = e^{17.5189} = 40,584,860.$$

Their statistical errors are moderately low:

$$\Delta\beta = 0.002490065 \quad \Delta c = 199,263.$$

We can now use Eq. 4 for estimating the number N_v of queries having a volume of at least v, using Eq. 8 for calculating the statistical error ΔN_v. Similarly, Eq. 7 can be used for estimating the total volume $\mathcal{V}_v$ of queries having a volume of at least v, and Eq. 9 for calculating its statistical error $\Delta\mathcal{V}_v$. Table 1 reports the estimates for a few values of v. As a means of comparison, the total empirical volume of the 18.5K queries in our sample amounts at 1,057M searches. Such a large number is consistent with the fact that the sample is not uniform, but highly ranked queries are more likely to be in the sample. Moreover, it also gives confidence that the sample is sufficiently large (as per empirical volume) to correctly estimate the true volume. According to the simulations of Section 4, the values $\hat{N}_v$ and $\hat{\mathcal{V}}_v$ may overestimate the true N_v and $\mathcal{V}_v$ respectively, if some sketchy approximation is introduced in the query volume data by Google Trends (or by any other SEO tools we might haved used in place of it). In case of noisy data,

[11]https://www.semrush.com
[12]The figure also shows the rank $max = 1725$ determined with the CSN method. The NLS fit is considered for the top max queries.

instead, under or overestimation may occur. The amount of such errors depend on the unknown amount of approximation or noise in the Google Trends data. Moreover, it is worth to stress that the reported statistical errors do not take into account such noise, but only the error of the parameter estimation procedures (assuming noiseless data).

7 CONCLUSIONS

We studied the problem of estimating the total search volume of queries belonging to a specific domain. By doing the sensible assumption that the unobserved rank distribution of absolute volumes follows a Zipf's law, our method can be decomposed in two parts. First, by comparing Google Trends data with results from SEO tools, we convert the relative volumes obtained from Google Trends for a subset of queries into absolute volumes. Second, we use the estimated absolute volumes of the subset of queries to infer the total volume of the queries of the domain. In doing this, we carefully took into account different sources of error (round-off by Google Trends and observational noise). We were also able to find the total number and the total volume of the queries in the domain which have been searched at least v times in a given time period. A large set of numerical simulations have supported the validity of our methods. Finally, we presented an empirical application to the estimation of the volume of the domain *recipes and cooking*, providing also error bars for the estimates. This kind of information is extremely useful in web marketing research and advertising.

The *first critical issue* for extending our analysis to other domains consists of checking the hypothesis that the population of queries in the domain is Zipfian. As shown in Figure 1, empirical data on the domain of recipes and cooking appears to be Zipfian. This motivated our assumption that the reference population, namely the queries searched in a reference domain, follows a Zipf's law. Ref. [9] points out that the granularity and extent of a reference population should exhibit a "coherence" property. This is particularly relevant, since splitting or merging two Zipfian sets does not necessarily yield another Zipfian set, hence the actual definition of what is and what is not in a domain is essential in meeting our assumption. The domain considered in this paper has well-defined boundaries that make it reasonably coherent.

The *second critical issue* is the construction of the sample set of queries. As shown by the numerical simulations, the capability of inferring the total volume significantly depends on the ability of selecting in the investigated sample queries which have likely high rank in the population (this is related to the parameter p in the non-uniform sampling). This set can be constructed either by resorting to domain's experts or, as we did in this paper, by crawling a set of specialized websites. Finding estimators which are (more) robust to the choice of the sample of queries is certainly an interesting potential extension of our approach to the case when it is costly or unfeasible to construct controlled samples.

The *third critical issue* is concerned with understanding which type of noise is likely to be present in empirical data provided by Google Trends or other SEO tools. In this paper, we considered three possible scenarios: uniform sampling alone, or together with normally distributed noise (noisy sampling), or together with count-min sketch like approximation (sketchy sampling). Other scenarios can be conceived, e.g., noise due to data anonymization [7, 13]. Further work is necessary to test which scenario fits better for a given SEO tool.

Finally, the *fourth critical issue* in our approach is which SEO tool to use for collecting volume of queries in the empirical sample. We relied on Google Trends, which provides relative volumes, and had to resort to GSC or other SEO tools as ground truth for determining a scaling factor. An alternative is to use SEO tools directly for collecting empirical absolute volumes. There are limitations of such tools which motivated our choice of using Google Trends – see beginning of Section 5. One of the issues is that they provide binned data. This means that the estimators of c and β might have to be reconsidered, e.g., by resorting to extensions of the CSN method to binned data [18].

ACKNOWLEDGMENTS

The work in this paper has been partially supported by a research grant by *ForTop S.R.L.* (https://www.fortop.it/en) on the topic: *Data-driven analysis of search engine query market.* We are grateful to L. Barsotti, S. Camarda, P. Ferragina, R. Guidotti, M Marino, and A. Monreale for several stimulating discussions.

REFERENCES

[1] L.A. Adamic and B.A. Huberman. 2002. Zipf's law and the Internet. *Glottometrics* 3 (2002), 143–150.
[2] R. A. Baeza-Yates, A. Gionis, F. Junqueira, V. Murdock, V. Plachouras, and F. Silvestri. 2007. The impact of caching on search engines. In *SIGIR*. ACM, 183–190.
[3] R. A. Baeza-Yates and A. Tiberi. 2007. Extracting semantic relations from query logs. In *KDD*. ACM, 76–85.
[4] A. Bookstein. 1990. Informetric distributions, part I: Unified overview. *JASIS* 41, 5 (1990), 368–375.
[5] A. Clauset, C. R. Shalizi, and M. E. J. Newman. 2009. Power-Law Distributions in Empirical Data. *SIAM Rev.* 51, 4 (2009), 661–703.
[6] G. Cormode, M. N. Garofalakis, P. J. Haas, and C. Jermaine. 2012. Synopses for Massive Data: Samples, Histograms, Wavelets, Sketches. *Foundations and Trends in Databases* 4, 1-3 (2012), 1–294.
[7] G. Cormode, T. Kulkarni, and D. Srivastava. 2018. Constrained Private Mechanisms for Count Data. In *ICDE*. IEEE, 845–856.
[8] G. Cormode and S. Muthukrishnan. 2005. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms* 55, 1 (2005), 58–75.
[9] M. Cristelli, M. Batty, and L. Pietronero. 2012. There is More than a Power Law in Zipf. *Scientific Reports* 2 (2012), 812.
[10] S. Ding, J. Attenberg, R. A. Baeza-Yates, and T. Suel. 2011. Batch query processing for web search engines. In *WSDM*. ACM, 137–146.
[11] C. Gillespie. 2015. Fitting Heavy Tailed Distributions: The poweRlaw Package. *J. of Stat. Software* 64, 2 (2015), 1–16.
[12] M.L. Goldstein, S.A. Morris, and G.G. Yena. 2004. Problems with fitting to the power-law distribution. *European Physical Journal B* 41 (2004), 255–258.
[13] L. Melis, G. Danezis, and E. De Cristofaro. 2016. Efficient Private Statistics with Succinct Sketches. In *NDSS*. The Internet Society.
[14] M.E.J. Newman. 2005. Power laws, Pareto distributions and Zipf's law. *Contemporary Physics* 46, 5 (2005), 323–351.
[15] A. Orlitskya, A.T. Sureshb, and Y. Wuc. 2016. Optimal prediction of the number of unseen species. *Proceedings of the National Academy of Sciences USA* 113 (2016), 13283–13288.
[16] C. Petersen, J. Grue Simonsen, and C. Lioma. 2016. Power Law Distributions in Information Retrieval. *ACM Trans. Inf. Syst.* 34, 2 (2016), 8:1–8:37.
[17] L. Vaughan and Y. Chen. 2015. Data mining from web search queries: A comparison of Google Trends and Baidu Index. *JASIST* 66, 1 (2015), 13–22.
[18] Y. Virkar and A. Clauset. 2014. Power-Law Distributions in Binned Empirical Data. *The Annals of Applied Statistics* 8, 1 (2014), 89–119.

Forecasting U.S. Domestic Migration Using Internet Search Queries

Allen Yilun Lin
Northwestern University
Evanston, Illinois
allen.lin@eecs.northwestern.edu

Justin Cranshaw
Microsoft Research
Redmond, Washington
justincr@microsoft.com

Scott Counts
Microsoft Research
Redmond, Washington
counts@microsoft.com

ABSTRACT

Roughly one in ten Americans move every year, bringing significant social and economic impact to both the places they move from and places they move to. We show that migration intent mined from internet search queries can forecast domestic migration and provide new insights beyond government data. We extract from a major search engine (Bing.com) 120 million raw queries with migration intent from 2014 to 2016, including origin and destination geographies, and the specific intent for migration such as whether the potential migration is housing or employment related. Using these queries, we map U.S. state level migration flows, validate them against government data, and demonstrate that adding search query-based metrics explains variance in migration prediction above robust baseline models. In addition, we show that the specific migration intent extracted from these queries unpack the differential demands of migrants with different demographic backgrounds and geographic interests. Examples include interactions between age, education, and income, and migration attributes such as buying versus renting housing and employment in technology versus manual labor job sectors. We discuss how local government, policy makers, and computational social scientists can benefit from this information.

KEYWORDS

internet search; migration; big data; housing; employment

ACM Reference Format:
Allen Yilun Lin, Justin Cranshaw, and Scott Counts. 2019. Forecasting U.S. Domestic Migration Using Internet Search Queries. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313667

1 INTRODUCTION

Every year over 32 million Americans, about 10% of the total population, migrate from one location in the United States to another [34]. The inflow and outflow of domestic migration can have substantial economic, political, and social impact. Because of this, many cities and states track and analyze migration data closely, looking to inform policies designed to retain existing residents and attract domestic migrants from other regions. For example, the Vermont General Assembly recently passed legislation that pays remote workers $10,000 if they move to Vermont from other states [50]. Even among states such as Texas and Arizona that have no difficulty attracting migrants, improved migration measurement and forecasts can aid development and planning related to local housing and job markets.

To track domestic migration in the United States, state governments and other interested parties rely almost exclusively on two migration datasets from the federal government [48], one from the American Community Survey (ACS) and another from the Internal Revenue Service (IRS). Both have limitations that might be mitigated by incorporating signals from large scale, naturalistic sources such as internet search queries. Most importantly, ACS and IRS data typically take two years to collect, prepare, and publish, forcing the consumers of these data to base current decisions on past, rather than fresh, measurements. Also, because ACS utilizes survey methods, the resulting data are based on a comparatively small samples, resulting in larger margins of error for smaller regions. IRS migration data, on the other hand, are derived from tax return records. People filing tax returns from a different state than they did the year prior are assumed to have migrated. While this method has a considerably larger sample size than ACS, it is subject to population biases, as it underestimates the migration of poorer, wealthier, and elderly citizens, all of whom may file taxes less frequently [29].

In terms of why people move, in over a century of research on migration, scholars have identified *housing* and *employment* as the two key motivators for moving [4, 30, 35, 69]. Prior research has demonstrated the predictive power of internet search in forecasting housing and jobs related measures. For instance, Wu and Brynjolfsson enhanced prediction of housing demand using Google searches [68]. Likewise, search queries have been used to "nowcast" unemployment rates [3] and job search interests [13]. Thus given how central the internet is in helping people find both jobs and housing, we explore whether internet search data can be used as a signal for forecasting and explaining domestic migration, with a focus on housing and employment as primary drivers of intent to migrate.

Using data from Microsoft's Bing search engine, we extracted and assessed the value of search queries from people researching potential moves. Drawing on theories from migration studies and the web search literature, we designed a highly accurate migration query filter that performs two tasks: it retrieves queries expressing the intent to migrate with F_1 score of 0.880, and it categorizes each query into specific types of migration intent, such as whether it indicates housing or employment needs. Using 120 million raw queries containing migration intent from 2014 to 2016, we show that these internet search data provide a meaningful measurement of migration when validated against core government migration

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313667

statistics. We further show that adding search-based variables to a model can improve its predictive power at forecasting future net migration flow at the U.S. state level, compared to two well-established and strong baseline models from migration literature. Finally, we examine the explanatory power of migration search data through the lenses of demography and geography, showing how an analysis of the different categories of migration queries (e.g., housing versus job, renting versus buying, technology versus manufacturing jobs) offers new insights beyond what government data provide.

In summary, this paper contributes a novel approach to domestic migration measurement and forecasting that utilizes search query data that are larger in scale, more timely, and more nuanced in explanation than current government migration data. Although we focus our analysis on the United States, we expect the approach would generalize to any region with reasonable internet penetration.

2 RELATED WORK

2.1 Motivation for Domestic Migration

Theories from migration studies have suggested that the needs for housing and employment are two forces that drive domestic migration. Researchers have conceptualized migration as a search and matching process [19, 48, 59] in that prior to actual moving, migrants are constantly searching for places that maximize the net benefit of moving. In particular, housing and employment are two of the most important factors when considering the benefits of moving [4, 30, 35, 69]. That is, on the one hand, people are willing to move to places with better job opportunities [15, 38], while on the other hand, the costs of housing could motivate or limit migration to certain areas [9, 36]. Although there has been debate on whether housing leads employment in motivating migration or vice versa [45, 49, 62], the consensus is that migrants almost inevitably need to search for both of them in their migration activities [48].

Empirical results from census surveys reaffirm that migrants are motivated by housing and jobs. According to the latest *Reason for Moving* report published by Census Bureau [33], employment-related reasons and housing-related reasons motivate nearly 68% domestic migrants. Although the report also shows that family-related reasons motivate 30% migrants, closer examination [32] reveals that some of the family-related reasons directly involve housing activities, including "To Establish Own Household."

As such, given how housing and employment are central in domestic migration, we consider housing and job as the most relevant topics when defining search queries with migration intent.

2.2 Macroeconomic Forecasting Using Search

With the internet becoming the default channel to search for information, economists and computer scientists have used search queries to model macroeconomic measurements that previously were available only through government and financial agencies, including phenomena such as car sales [16] and the stock market volumes [5, 54, 55]. Housing and employment demand, again two macroeconomic factors highly relevant to migration, have also been successfully modeled with internet searches. Most prior work relies on Google Trends, which are normalized, aggregated query volumes for particular keywords and entities. Google Trends data have also been used, for example, to forecast unemployment rate for both U.S. [16, 20, 22] and other countries [52].

Similar work has been done to predict housing trends. Wu and Brynjolfsson [68] have shown that search frequencies from Google could enhance the prediction of the Housing Price Index (HPI) on both national scale and local scale. McLaren and Shanbhogue [46] have used a similar approach to nowcast housing market dynamics in UK. Finally, other projects have sought to go beyond predicting macroeconomic measures to understanding socioeconomic drivers and effects, such as the relationship between searches for different types of jobs and searchers' demographics [13].

Taken together, this body of work provides us confidence about the applicability of search query based data to accurately model demand for housing and employment, the two key motivations related to migration. Methodologically, we were also able to leverage keyword dictionaries established in this prior work to distinguish employment and job related migration search queries.

2.3 Identifying Migrants with Internet Data

Most relevant to our work, a number of studies from computational social science have used internet data to identify migrants. The first type of work does so by mining personal digital traces on social media platforms. Using Twitter data, Zagheni et. al. [70] identified migrants as those whose geolocated tweets exhibit several distinct geographic clusters chronologically. Later work by Fiorio et. al. [24] also used Twitter data to identify and distinguish between short-term movers and long-term migrants. The location field in online profiles from other social network platforms, including Google+ [47], Facebook [51] and LinkedIn [2], have also been exploited to identify migrants. Finally, by using IP address data from Yahoo!'s 100 million email users, researchers identified international migrants and studied their migration patterns across many demographic attributes [67, 71].

However, as the data collected from this prior work only identified migrants *after* they moved, they only shed light on existing migration trends and will be less useful in forecasting migration. Decision makers who rely on this information can not prepare themselves beforehand. Moreover, previous methods that use open-access data from Twitter can only collect data for a small percentage of migrants, which limits their ability at accurately predicting national level migration statistics. [70].

By mining large scale search queries, our approach has two advantages over prior work. First, search queries usually precede actual migration, offering a unique advantage for prediction. Second, the content of the search queries offers additional insights into the specific types of demand (e.g., housing or job related) that can indicate economic impact to local markets.

3 MIGRATION QUERIES FILTER

In this section, we define *migration queries* in the context of this paper. We then describe how we design a filter to accurately extract migration queries from a larger pool of search queries.

3.1 Defining Migration Queries

Formally, we define migration queries as *housing or employment queries that explicitly target a destination geographic unit different*

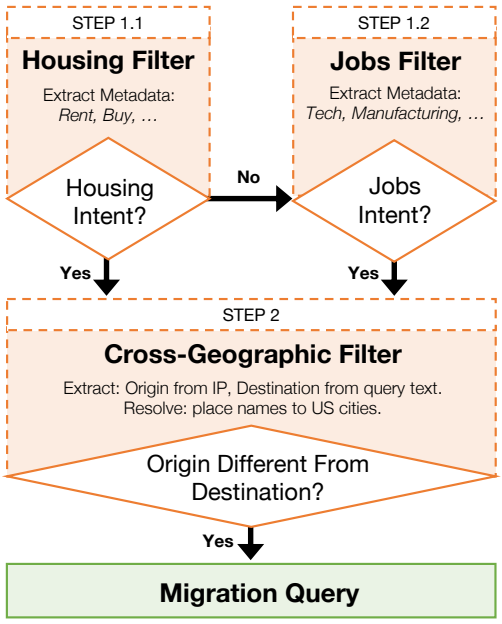


Figure 1: Simplified flowchart of our migration query filter.

from the origin of the query. Our emphasis on housing and employment queries is driven by the theories from migration literature (see Related Work) in which housing and jobs are two major reasons for moving. In the context of search queries, housing and employment keywords make the intent to migrate more apparent. For instance, the housing related term "apartments" in a query such as "apartments in Boston" that originated from Phoenix raises the likelihood of an intended move from Phoenix to Boston.

An important component of our definition is the notion of "geographic unit". Migration researchers usually select relatively large geographic units, including county, metropolitan statistical area (MSA), core-based statistical area (CBSA) and state [48]. However, empirically we observed that housing and employment queries often mention the more specific geographic units (e.g. "apartment to rent in DC", "software engineer job in San Francisco") than those that are used in migration studies. This trend is particularly prominent for housing queries that target a big metropolitan area where certain cities or even neighborhoods within that MSA function as residential areas. As such, we decide to focus on *county*, *city*, and *zip code* in the migration query retrieval process (detailed below) and then later aggregated them to larger geographic units (states) used in the analyses. We also note that our definition requires an explicit mention of the geographic unit in the query. We do so because housing or job queries without mentioning geography, such as "data scientists jobs", "apartment near me" are more likely to be local rather than migration related.

3.2 Design of the Filter

Before we detail the implementation of the migration query filter, we discuss two high-level challenges that influence many specific design choices. First, our task requires identifying *both* the topic and the destination geography of the query, a problem not seen in previous work. As such, we can't leverage a readily available search trend product such as Google Trends since its statistics are not segmented by query destination, and instead must filter raw queries. This presents the second challenge: processing an enormous amount of data.

To address the first challenge, we divide the task into extracting housing and jobs topics and extracting place names. We leverage prior studies that use housing and jobs related search queries (e.g. [13, 16, 68]) to guide our selection of keywords. For extracting geography, we turn to the literature of toponym disambiguation [7, 8, 60]. While different approaches exist, an external knowledge source, such as gazetteers or Wikipedia, usually serves as the foundation to more advanced methods [18, 66], and often times provide a competitive baseline accuracy [60]. As such, we adopt a straightforward gazetteer approach to build our geography extractor which will be detailed below. To address the second challenge of large data volume, our design prioritizes *run time* over complexity and *precision* over recall. As such, we employ rule-based filters and avoid machine learning approaches whenever possible. However, we show that our accuracy is comparable to prior work that implements machine learning based filters.

Figure 1 presents the detailed implementation of our migration query filter. After filtering out abnormally long queries, each raw query is first passed through a housing query filter (Step 1.1). To implement this filter, we leverage the keywords from prior work that uses housing search queries to forecast housing sales [68], which includes "real estate", "apartment", "property", "house" etc. Notably, we expanded these keywords by adding the names of popular housing websites such as "zillow" and "redfin." Known as "navigational queries" in the web search literature [6], search queries containing these websites also indicate users' intention for housing information. Specifically, we added the names for the top ten housing websites according to rankings from Alexa.com. If a query is identified as a housing query, our filter identifies whether the user's intent is to *buy* or *rent* housing by looking for the existence of terms such as "prices", "sale" (for buying) and "rent", "rental" (for renting). These housing queries are saved for location extraction, which will be described below.

Non-housing queries are subsequently processed by a jobs query filter (Step 1.2). We took a similar approach to construct a set of keywords to identify job related migration searches. First, we obtained the specific keywords for 14 job categories that were used in [13]. In a later analysis, we focus our attention on four of these categories: Science, Technology, Manufacturing, and Transportation. Similar to the housing query filter, these topical keywords were expanded with navigational search terms that refer to top 10 job search websites from Alexa.com, including "indeed", "monster" etc.

If a query is either a housing query or job query, it's transferred to Step 2 to extract and compare the origin and the destination geography. While the origin of the query is contained in the reverse-IP look up results, which was provided to us as part of the query data sample, extracting the destination of the query is non-trivial. As we discussed earlier, we leveraged a gazetteer to identify the reference of cities, counties and zip codes in the query. While the zip code and county names are straightforward to obtain, cities are ambiguously defined in U.S. geography. We adopt the Census definition of "place", which is most similar to scale of a colloquial "city" [26] and includes

	#	%	Example queries
Housing	100,123,000	83	"condo in san francisco", "home for sale in maricopa county", "apartment in santa clara, ca"
↪ **Buy**	39,095,000	39	"home for sale in maricopa county", "house price 20500", "duplex for sale in chicago"
↪ **Rent**	32,144,000	32	"apartment for rent in seattle", "house for rent in texas city, texas", "apartment for rent 60231"
Jobs	20,890,000	17	"jobs in st paul", "software engineer jobs in mountain view", "accounting jobs in cook county"
↪ **Sci/Tech**	204,000	1	"software engineer jobs in mountain view", "jobs in r&d in boston", "sql dba jobs in tampa, fl"
↪ **Mfg/Transp**	719,000	3	"van driver job in santa clara ca", "schoolbus driver jobs in king county", "cdl jobs in philadelphia"
Total	121,014,000		

Table 1: Distributions and examples of migration queries. Number are rounded and content is very slightly altered for privacy reasons. Percentages for sub-type queries are normalized by all housing or all jobs queries accordingly.

both Incorporated Places and Census Designated Places. In total, we extracted over 20,000 such names.

When matching these place names to the query content, an additional challenge is the ambiguity problem. For example, there are over 30 cities named "Franklin" in the United States. To increase the precision, we took a two step approach. First, for the largest 50 cities in the U.S. we assume the city name refers to that city (e.g., "chicago" in the query "biotech jobs Chicago" refers to Chicago, IL). For smaller cities, we disambiguate with a state name or abbreviation (e.g., "biotech jobs Portland ME") if one is present in the query and adjacent to the city name. In the second step, queries containing names of smaller cities that lacked an accompanying state name or abbreviation were post-processed by Bing's own location disambiguation service. This disambiguation process breaks the query into n-grams, identifies location names, and then provides a ranked list of the most likely matching locations using features such as the popularity of the location and proximity to the origin of the query (as determined by the reverse-IP look up). Using this, we chose the query destination to be the the top ranked result returned by the disambiguator. Generally, this process conforms to how search engines interpret location queries, and how users behave after they understand search engines' interpretations. For instance, users typically know to specify the state of a city when that city is small or is not in their immediate locale. Finally, we compare the origin and destination on *state* level for the purpose of analyses (detailed later) and extract queries whose origin states are different from destination states as migration queries.

3.3 Evaluation and Filtering Results

We conducted a lightweight evaluation of our migration query filter before deploying it on all queries. We first developed a test set by collecting 500 housing and job queries as identified by Step 1.1 and Step 1.2 in Figure 1. We used this as our test set rather than a random sample from all queries because migration queries are only a small percent of all queries. Two researchers coded the 500 queries as either *migratory* or *non-migratory* according to the definition of migration queries. For samples with disagreement, the researchers discussed and reached consensus on a label. Coding results show 92 migration queries among these 500 housing and jobs queries.

Comparing these human labels to the results of our migration query filter indicates generally strong classification results. Our filter achieves high performance on all metrics, including precision (0.870), recall (0.891), F1 score (0.880) and accuracy (0.880). We note that our filter's performance is comparable to prior work that filtered search queries into job categories [13]. We applied this filter on a large sample of queries from 2014 to 2016, yielding just over 121 million migration queries in total (see Table 1). Among these, 83% are housing-related and 17% are job-related. Among housing migration queries, we observe a roughly equal amount of buy-specific and rent-specific queries. Among job migration queries, there are three times as many Manufacturing and Transportation related migration queries as there are Science and Technology related queries, which is consistent with prior work [13].

4 FINDINGS AND INSIGHTS

The data collection and filtering pipeline described above produced a corpus of more than 120 million migration queries from 2014, 2015 and 2016, each categorized with information about its origin state, destination state and specific types of migration intent. In this section, we detail our analyses and findings, organized into three parts. First, we validate our migration queries against external government statistics, showing that our data are highly correlated with core migration measures. Second, we prove that adding variables derived from these queries enhances the prediction of migration measurements over two robust baseline models from the migration literature. Third, we demonstrate that by extracting categorical descriptors of the type of query, such as whether it is about housing or jobs, we are able to shed light on the differential demands of migrants from varied demographic backgrounds and geographic interests.

We conduct all comparisons to government measures using data from the *American Community Survey* (ACS), focusing on migration between the 50 states, excluding the District of Columbia. ACS data is chosen for the following reasons. First, among all popular government migration data sources, only ACS provides migrants' detailed demographic information, which allows us to understand how migration-related search demands vary across different demographics. Second, despite being only a 1% sample of the total population, ACS has shown to be equally accurate to IRS migration data, which cover 87% of American households, on many important migration measures [48]. We focus on the state level because it is a popular aggregation unit used in the traditional migration literature (e.g. [12, 27, 53, 61, 64]) and because it bears smaller sampling errors compared to finer-grained units of aggregation.

Since state-level migration data are geographic data, we tested for the existence of spatial autocorrelation. Spatial autocorrelation describes a well-known phenomenon in spatial data that measurements taken at nearby locations tend to be correlated [43], which

	Inflow Correlations		
	Migration 14	**Migration 15**	**Migration 16**
Queries 14	0.72	0.78	0.75
Queries 15	0.67	0.73	0.73
Queries 16	0.65	0.71	0.73
	Outflow Correlations		
	Migration 14	**Migration 15**	**Migration 16**
Queries 14	0.32	0.59	0.60
Queries 15	0.26	0.53	0.52
Queries 16	0.21	0.46	0.44
	Netflow Correlations		
	Migration 14	**Migration 15**	**Migration 16**
Queries 14	0.11	0.55	0.58
Queries 15	0.06	0.45	0.59
Queries 16	0.13	0.42	0.61

Table 2: Correlations between migration rates and migration queries by year for inflow, outflow, and netflow respectively.

violates the independence assumption that underlies many statistical tools. We followed the state-of-the-art procedure for testing spatial autocorrelation in [21, 65], finding weak to no spatial autocorrelations on all variables used in our analyses (defined below), and thus we maintain use of conventional statistical approaches.

4.1 Correlations with migration statistics

In this section, we validate our query-based migration measures. To do so, we examine the relationship between the volume of migration queries and three core migration measures from government: (1) inflow migration rate, which is the number of people who move into a specific state in a year, normalized by the state population; (2) outflow migration rate, which is the number of people who move out of a specific state in a year, normalized by the state population; and (3) net migration rate, which is the difference between inflow and outflow migration rates. These measures are used widely in many social science studies [10, 12, 61] and are the primary indicators of the nation's overall migration status. Although they are estimated annually by different federal agencies (e.g. ACS and IRS), there is typically a two year lag time to publication. For the time periods considered in this paper, these measures typically were in the low single digit range per state. For example, in 2016, the net migration rate varied from -1.5% in Alaska to +1.2% in Arizona. Search volumes that correspond to these three measures are also normalized by state population.

Pearson's correlation coefficients between these three core migration measures and their corresponding search volumes (normalized by state population) are shown in Table 2. These results suggest two themes. First, we observe generally high correlations between the official migration measures and the volume of migration queries. Among the three migration variables, migration search queries are most accurate at measuring inflow migration, with a mean correlation coefficient of 0.72 and standard deviation of 0.039 ($\mu = 0.72$, $\sigma^2 = 0.039$). Figure 2 shows a scatterplot of inflow migration rate in 2015 and inflow migration queries in 2014, controlling for state

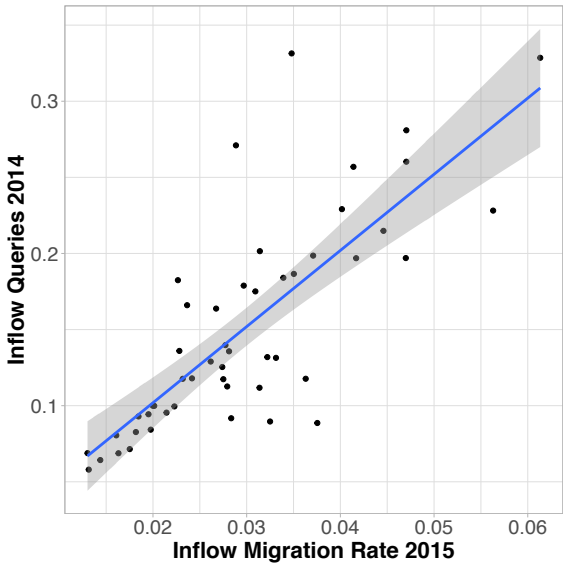


Figure 2: Incoming migration queries in 2014 and actual incoming migrants in 2015, both normalized by state population. Each point represents a U.S. state. Shaded area indicates 95% confidence interval of the linear regression.

population, with a strong linear relationship between migration queries and actual migrants. The correlation coefficients are lower for outflow ($\mu = 0.44$, $\sigma^2 = 0.14$, per Table 2) and net migrations ($\mu = 0.39$, $\sigma^2 = 0.22$, per Table 2). There are at least two possible explanations for these lower correlations. First, the origin states that are used to aggregate outflow migration are inferred by reverse-IP location, which might be less accurate than the destination states that are directly extracted from the query. Second, it is possible that the 2014 ACS outflow migration measure is an outlier, thus lowering the overall correlation. The outflow migration measure from ACS in 2014 have considerably lower autocorrelation with the same variables from 2015 ($r_{out14,out15} = 0.90$) and from 2016 ($r_{out14,out16} = 0.79$) compared to the equivalent autocorrelation for inflow migration ($r_{in14,in15} = 0.98$, $r_{in14,in16} = 0.96$). If only considering 2015 and 2016, our search queries present medium to strong correlations with the outflow migration ($\mu = 0.52$, $\sigma^2 = 0.06$) and with net migration ($\mu = 0.53$, $\sigma^2 = 0.08$).

The second theme from Table 2 is that search queries appear to have higher correlations with contemporaneous or future migration variables than with the past. Using inflow migration as an example, we group the correlation coefficients for inflow migrations in Table 2 by the number of years between the migration queries and the ACS statistics and compute the means for each group. Results in Figure 3 suggests that inflow search queries that precede the year of interest ("lead 2 years" and "lead 1 year") have higher correlations with the ACS inflow migration rates than the search queries that are contemporaneous ("same year") or follow the year of interest ("lag 1 year" and "lag 2 years"). We note that this leading effect of migration queries is consistent with previous findings on housing search queries, which demonstrates that web queries are usually good indicators of demand that will be eventually reflected in future consumer behavior [16, 68]. In the case of migration, it stands to reason that a typical prospective migrant would need to search

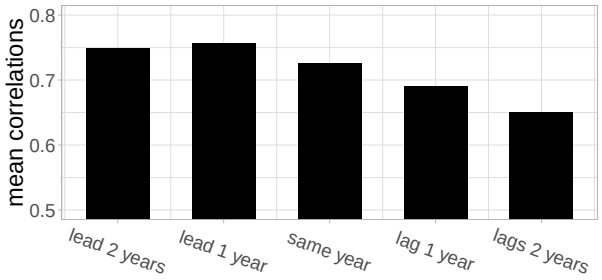


Figure 3: The leading effect of migration queries.

for a home or a job prior to actually moving. We provide further evidence for leading effect when demonstrating improvements to migration forecasting (see below).

Beyond the three primary migration measures, we also briefly highlight that the search query-based measures correlate strongly with secondary migration measures from government data. For example, ACS also collects data about the tenure of the migrants, which captures for each state the number of incoming migrants who currently live in owner occupied housing units and renter occupied housing units respectively. These tenure variables are very similar to the "Buy" and "Rent" variables we extracted (see Table 1). We thus compute the Pearson's correlation coefficients between the owner-renter ratio using ACS tenure data in 2016 and the buy-rent ratio using migration query data in 2015. Our result shows that these two variables indeed are strongly correlated ($\rho = 0.67$), further validating a query-based migration measurement.

4.2 Forecasting migrations statistics

The previous analyses demonstrated that migration queries alone can reasonably measure and even lead future migration variables. However, this doesn't necessarily indicate their incremental value in predicting future migration. In this section, we compare baseline migration prediction models with enhanced models that include search query measures, which is a typical methodology adopted by previous studies to demonstrate search queries' predictive power on economic indicators [3, 13, 16, 68]. Specifically, we follow these works to compare the goodness of fit (*adjusted* R^2) between the models with and without search query measures. We note that to be consistent with baseline models from the migration literature (e.g. [12, 61]), we construct models to predict *state level net migration*.

Our comparisons include two types of well-studied baseline models from migration literature. The first type of model is the autoregressive model. Because macroeconomics variables such as migration rates tend to be highly autocorrelated in time, simply forecasting future values from past data can provide good estimates. Traditionally known as the *Petersen-Greenwood hypothesis* in the migration literature [27, 53], this model treats current migration movement as a function of historical movement and has been shown to be effective in modeling state-to-state migration patterns [64]. We note that although more sophisticated nonlinear models are possible, we follow the practice common to these migration studies [27, 64] and use simple linear regression.

We first estimate the baseline model in Eq. 1 to predict 2016's ACS net migration rate using historical data from one year prior. This model is then compared with an enhanced model in Eq. 2 that adds search query variables from one year prior. Because domestic migration is a long decision making process, we hypothesize that adding query variables from multiple years might additionally enhance the prediction. As such, we also estimate an enhanced model in Eq. 3 that incorporates search query data from two years prior.

$$NetACS_{16} = \alpha + \beta_1 NetACS_{15} + \epsilon \tag{1}$$

$$NetACS_{16} = \alpha + \beta_1 NetACS_{15} + \beta_2 NetQueries_{15} + \epsilon \tag{2}$$

$$NetACS_{16} = \alpha + \beta_1 NetACS_{15} + \beta_2 NetQueries_{15} + \beta_3 NetQueries_{14} + \epsilon \tag{3}$$

We then estimate a stronger baseline model in Eq. 4 that includes the prior two years of ACS data. We note that we also explored models with longer histories of ACS migration data (e.g., an aggregate of the past five years) and the results were approximately the same or even worse in some cases. Similar to the previous comparison, we enhance this baseline by adding search variables of the prior one year (Eq. 5) and prior two years (Eq. 5) respectively.

$$NetACS_{16} = \alpha + \beta_1 NetACS_{15} + \beta_2 NetACS_{14} + \epsilon \tag{4}$$

$$NetACS_{16} = \alpha + \beta_1 NetACS_{15} + \beta_2 NetACS_{14} + \beta_3 NetQueries_{15} + \epsilon \tag{5}$$

$$NetACS_{16} = \alpha + \beta_1 NetACS_{15} + \beta_2 NetACS_{14} + \beta_3 NetQueries_{15} + \beta_4 NetQueries_{14} + \epsilon \tag{6}$$

The second type of baseline models we compare to is explanatory models that use socioeconomic factors to predict net migration. Among numerous such models (see comprehensive summaries in [11, 28]), we select the explanatory model proposed by Cebula and Alexander [12] as our baseline for the following reasons. First, it is designed and tested specifically to model state-level net migration rates. More importantly, it encompasses a comprehensive set of variables, including both traditional factors and novel "quality-of-life" factors [10], and has been shown to outperform models that include only economic factors [14, 56]. Lastly, as it was developed only a decade ago, the socioeconomic factors it considers should be the most pertinent to our context.

For model specification, we pick the best performing model from Cebular and Alexander [12] and include all the significant explanatory variables. We traced the source of these variables in the original model and found their most time-relevant counterparts [1]. Specifically, in Eq. 7, *MFI* is the median family income of each state; *COL* is the cost of living index for average four-person family; *EMPLGR* is the percent employment growth rate; *JANTEMP* is the daily maximum temperature in January; *HAZARD* is the percent distribution of hazardous waste sites of each state on the National Priority List; *TOXIC* is the toxic chemical releases per person; *EDPUP* is the government expenditures on elementary and secondary school, *STINCTAX* is the per capita state income tax burden. For brevity, we denote these socioeconomic variables with single notation *SOCIOECON* in ensuing model specifications and results interpretations. Readers should refer to the original paper [12] for detailed regression results for these variables. We enhance this baseline model by adding one year prior search variable (Eq. 8).

[1]We were unable find the exact counterpart data for a variable about the maximum January temperature, so we used the average winter temperature from NOAA.

	Dependent variable: $NetACS_{16}$									
	(1)	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**	**(9)**	**(10)**
$NetACS_{15}$	0.641***	0.493***	0.571***	0.518***	0.312**	0.386***			0.389***	0.310***
	(0.098)	(0.099)	(0.108)	(0.103)	(0.099)	(0.106)			(0.081)	(0.085)
$NetACS_{14}$				0.148*	0.185***	0.180***			0.136	0.184*
				(0.056)	(0.048)	(0.047)			(0.079)	(0.078)
$NetQueries_{15}$		0.036**	0.084**		0.042***	0.085**		0.027**		0.018*
		(0.011)	(0.030)		(0.010)	(0.026)		(0.010)		(0.008)
$NetQueries_{14}$			-0.053			-0.05				
			(0.030)			(0.027)				
SOCIOECON							(Omitted)	(Omitted)	(Omitted)	(Omitted)
Obs. (States)	50	50	50	50	50	50	50	50	50	50
Adj. R^2	0.462	0.556	0.573	0.521	0.656	0.670	0.679	0.723	0.828	0.843

Note: Standard errors are shown in parentheses; SOCIOECON model coefficients are omitted for space considerations
Significance notation: ***p < 0.001; **p < 0.01; *p < 0.05

Table 3: Linear regression to predict 2016 net migration rate

$$NetACS_{16} = \alpha + \beta_1 MFI + \beta_2 COL + \beta_3 EMPLGR + \beta_4 WINTEMP + \beta_5 HAZARD + \beta_6 TOXIC + \beta_7 EDPUP + \beta_8 STINCTAX + \epsilon \quad (7)$$

$$NetACS_{16} = \alpha + \beta SOCIOECON + \beta_9 NetQueries_{15} + \epsilon \quad (8)$$

Finally, we estimate a third class of baseline model that combines the explanatory model and the autoregressive model, shown in Eq. 9. We compare this to an enhanced model Eq. 10 that adds one year prior search query variables.

$$NetACS_{16} = \alpha + \beta SOCIOECON + \beta_9 NetACS_{15} + \beta_{10} NetACS_{14} + \epsilon \quad (9)$$

$$NetACS_{16} = \alpha + \beta SOCIOECON + \beta_9 NetACS_{15} + \beta_{10} NetACS_{14} + \beta_{11} NetQueries_{15} + \epsilon \quad (10)$$

Results in Table 3 show consistent improvement when adding search query variables to various baseline models. For the simplest autoregressive baseline (Eq. 1), adding one year prior search queries (Eq. 2) increased the adjusted R^2 from 0.462 to 0.556, a 20% of improvement. Likewise, for the stronger autoregressive baseline that includes the prior two years of historical data (Eq. 4), adding one year prior search queries (Eq. 5) increased the goodness of fit from 0.521 to 0.656, or a 26% of increase. We highlight that these improvements are considerably larger than those found in similar work that also demonstrate the power of search query data to predict macroeconomic phenomena (e.g. [68]). Including an additional prior year of search queries (Eq. 3 and Eq. 6) does increase the adjusted R^2. However, the estimated coefficients of $NetQueries_{14}$ are non-significant and in fact slightly negative, indicating a multicollinearity issue. Indeed, the VIF for $NetQueries_{14}$ and $NetQueries_{15}$ in both models are greater than 5, confirming the existence of multicollinearity [58]. As such, with building parsimonious models in mind, we drop $NetQueries_{14}$ in later models. We also note that dropping $NetQueries_{15}$ from Eq. 3 and Eq. 6 still provides models better than the baseline (Eq. 1 and 4), indicating that two year ahead query variables can still contribute to migration prediction.

Similarly, we observe improvement in adding search query variables to the SOCIOECON explanatory model. Comparing the results for Eq. 7 and Eq. 8, we observe that the adjusted R^2 increases from 0.679 to 0.723. Finally, for the model that encompasses both socioeconomic variables and historical data (Eq. 9 and Eq. 10), adding search queries still provides 0.02 increase to the model's Adjusted R^2, again an improvement comparable to similar work demonstrating the predictive power of search query data in the macroeconomic domain (e.g. [68]). For all the above models, we emphasize that all estimated coefficients of $NetQueries_{15}$ are positive and significant, showing non-zero effects of the search query variable.

In addition to predicting *NetACS16*, we also estimated models predicting inflow migration rate and outflow migration rate. Although inflow or outflow alone are easier to predict compared to net migration, we observe consistent, though smaller, improvements even when the baseline models are already very accurate. For instance, for a baseline model that predicts 2016 inflow migration rate using previous two years' history data, adding search query measures enhances the adjusted R^2 from 0.871 to 0.893, again, an increase that is comparable to related work.

Finally, we note that we experimented with different specifications for these models. For example, we tried adding contemporaneous *NetQueries* variables to all models, e.g. using $NetACS_{15}$, $NetQueries_{15}$ and $NetQueries_{16}$ to predict $NetACS_{16}$. We observed little to even negative changes to the adjusted R^2, which offers additional support for our model specification and for the previous finding about the leading effect of search queries.

Overall, the analyses in this section demonstrate that that migration queries consistently improve the prediction of core migration outcomes across several strong, theory-based baseline models.

4.3 Understanding Migration Demand

One additional advantage of our approach is that out filter extracts detailed metadata for each migration query, indicating whether the search is housing-related related or job-related, even the sub-types for housing and jobs. We thus investigate how the intent to migrate varies on two important dimensions. First, we study how migration demand (indicating the sum of all expressed intent to migrate, even if this expression goes unrealized) varies *demographically* by age, income and educational attainment. These attributes have been shown in migration studies to have differential impacts on domestic migration [37, 39, 57]. Second, we demonstrate that the type of migration demand (e.g., driven by housing versus jobs)

	Housing	Jobs	Housing-Rent	Housing-Buy
Age 18-24	-0.40	0.40	-0.06	-0.19
Age 25-44	-0.19	0.19	0.41	-0.18
Age 45-64	0.48	-0.48	-0.40	0.30
Age > 65	0.53	-0.53	-0.50	0.31

(a)

	Housing	Jobs	Jobs-Sci/Tech	Jobs-Mfg/Transp
< 25k	-0.17	0.17	-0.19	0.09
25k-50k	0.27	-0.27	0.17	0.06
50k-75k	0.23	-0.23	0.16	-0.42
> 75k	0.15	-0.15	0.20	-0.26

(b)

	Jobs-Sci/Tech	Jobs-Mfg/Transp
< High School	-0.23	0.37
High School	-0.32	0.35
Some College	-0.12	0.36
Bachelors	0.25	-0.44
Graduate	0.27	-0.43

(c)

Table 4: Correlation between migrants demographics and specific types of migration queries

varies *spatially*, even among states with similar net migration rates.

4.3.1 Migration Demand and Demographics. We hypothesize that the type of migration demand will differ based on varying demographics. Specifically, we focus on differential migration demand in relation to age, income, and education. These demographic attributes have previously been shown to affect related concepts, such as demand for employment, as measured by internet search queries (e.g. [13]). We obtained data for these demographic attributes from 2016 ACS provided through NHGIS [44]. Following our previous analyses, we compare these government data with search data from 2015, removing the District of Columbia. However, different from previous analyses which focused on net migration, here we study inflow migration for the following reasons. First, instead of prediction, this section focuses on explaining the relationship between where people are moving to and relevant demographics, and hence focusing on only one side of migration makes this relationship clearer. Second and more importantly, when segmenting total net migration into different groups, the margins of error of each group, as computed according to ACS's recommendation [25], can sometimes be larger than the net statistics, endangering the accuracy of our analysis. Furthermore, we chose to focus on inflow migration using traditional migration literature that studies the economic impact of international migration (e.g. [40, 42]) as guidance. We briefly discuss the outflow side at the end of the results, which present a generally similar trend.

First, we investigate how the types of migration demand, proxied by the types of migration queries, varies across age groups. To do so, we use ACS Table B07001 which captures for each state, the number of in-migrants for different age groups. Census publications uses a variety of age group breakdowns. To reduce the margins of error, we select the one with the fewest age groups, which is also used widely in decennial census publication (e.g. [31]). Motivated by the migration literature that argues that age plays an important role in housing-related migration (e.g. [33]), we study the relationship between age and housing-related migration searches, first in comparison to job related migration searches, and then within the buy versus rent housing query sub types. We normalize ACS age group data with the total number of inflow migrants, normalize housing and job queries by the total number of migration queries, and normalize rent and buy queries with the total number of housing queries, effectively turning every raw count into a proper percentage. We do so because we are most interested in understanding their relative relationships, rather than predicting their raw values.

Table 4 (a) presents the correlations between age groups and different types of migration demand. We note that because housing queries and jobs queries sum to 1 after our normalization, their correlation coefficients always have opposite signs and equal absolute values. Our results suggest that the percentage of older age migrants (e.g., age > 45) has positive correlations with the percentage of housing queries, whereas the percentage of younger migrants has positive correlations with the percentage of job queries to a state. In other words, states that attract a relatively large portion of housing queries in 2015, received large portion of older migrants in 2016 and states that drew job queries (or low housing queries) in 2015, pulled in more young migrants. These results are in line with prior migration theory which argues that young migrants are more motivated by employment and older migrants by housing [33]. To further unpack the relationships between age group and housing demand, the results in Table 4 (a) show that states that attract older age migrants in 2016 are positively associated with higher housing buy demand in 2015, whereas states that receive younger age group are positively associated with higher housing rent demand.

Next, we examine the relationship between income and migration demand. For income data, we again uses ACS Table B07010, that records the number of migrants for each income bracket. Similar to the previous analysis, we merge some income groups for ease of interpretation. Motivated by migration theories that demonstrate the interaction between migration, income and employment [17, 23], we investigated the relationship between income level and demand for employment related migration, with a particular focus on two job categories currently driving important economic disparities in the U.S.: science and technology versus manufacturing and transportation. We pre-process both ACS data and job related migration queries data with the same normalization procedure as in the above analysis on age.

Results in Table 4 (b) highlight two trends. First, as expected, our results indeed show that states with more higher income migrants are associated with more high tech job related inbound migration searches. Although the effect size is relatively small, these correlations shows distinctions when compared to the negative correlations between low income migrants and high tech job migration searches. For states that attract more low income migrants, our results did not suggest that they also attract large manufacturing and transportation jobs searches. Looking at this further, we did notice a trend that these same states saw increases in inflow job related migration searches in general, suggesting a possibility that

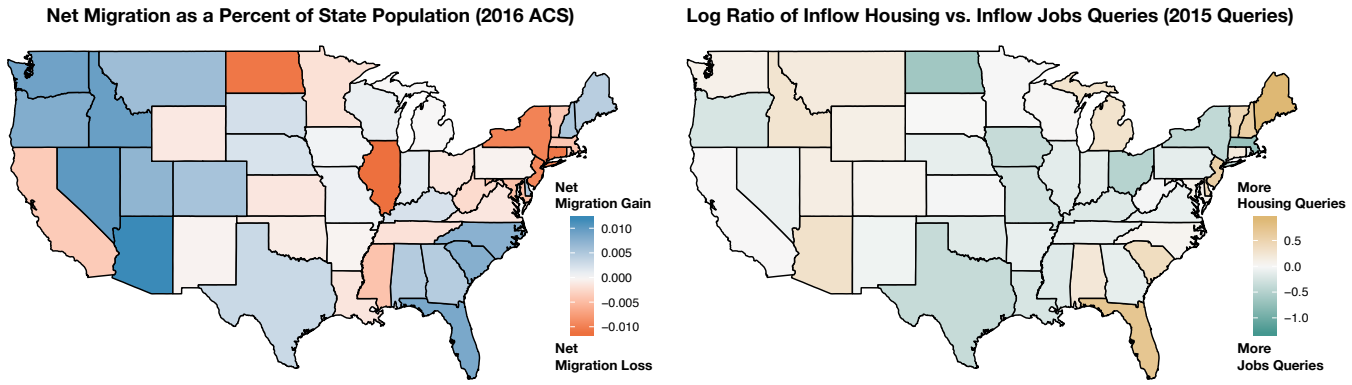


Figure 4: Left: Net domestic migration per state from the 2016 American Community Survey, shown as a percentage of state population. Right: the ratio of a state's 2015 inbound housing migration queries relative to its incoming jobs queries, normalized by that same ratio over all queries, shown here in log scale.

these lower income migrants might be looking for jobs that do not requires special skills. Non-specific job related migration may be a direction for future exploration.

Finally, we explore the relationship between education level and migration demand. ACS data record the number of migrants in five different education attainment groups ranging from "Below High School" to "Graduate". Like the previous analysis, we correlate these categories with science and technology versus manufacturing and transportation employment migration searches based on prior research that looks at employment searches and demographics [13]. We applied the same data pre-processing before conducting the correlation analyses. Results in Table 4 (c) indicate that states with greater inflow migration queries in science and technology see more migrants with bachelor and graduate degrees. By contrast, states with more inflow migration queries for manufacturing and transportation jobs attract more migrants with low education attainment. These results demonstrate that the relationships between employment demands and demographics that are shown in previous research [13] also ring true in the migration context.

As previously mentioned, we replicated the above analyses using outflow job queries and outflow migrants demographics. We generally observed similar trends, although some effect sizes are smaller. We hypothesize that this might be due to the fact that on the inflow side, certain states serve as the "magnets" that attract particular demographics, while such an effect is weaker on the outflow side.

4.3.2 Migration Demand and Geography. Domestic net migration is a core migration statistic that always draws extensive commentary and interpretation from columnists, policy makers and scholars upon its publication each year [1, 41, 63]. While much of this discussion centers around which states are growing and shrinking the fastest, raw migration statistics are not, on their own, able to contextualize the economic impact to local housing and job markets.

The left map in Figure 4 visualizes net domestic migration in the 48 contiguous United States using data from ACS 2016. In this figure, we see that states such as Florida (+0.83%), Texas (+0.31%), Arizona (+1.17%) and Nevada (+1.00%) are experiencing net population growth from domestic migration, while states such as New York (-0.96%), California (-0.36%), North Dakota (-1.06%), and Illinois (-1.11%) are losing more people than they are attracting.

Our data allow us to characterize the flow of migration queries into (or out of) a state as to whether the queries are disproportionately about housing or about jobs. Following the previous analysis, we focus only on an analysis of inflow; the analysis of outflow, which may help explain those states in deficit, follows analogously.

First, for each state s, we compute the proportion of incoming housing queries s receives relative to the housing queries in all other states:

$$\rho_{\text{in}}^{\text{housing}}(s) = \frac{\text{total inbound housing queries to } s}{\sum_{s'} (\text{total inbound housing queries to } s')}.$$

Similarly, we compute the proportion of incoming jobs queries:

$$\rho_{\text{in}}^{\text{jobs}}(s) = \frac{\text{total inbound jobs queries to } s}{\sum_{s'} (\text{total inbound jobs queries to } s')}.$$

All things equal, the ratio of job and housing related migration searches would be similar across states, however in practice, $\rho_{\text{in}}^{\text{housing}}(s)$ and $\rho_{\text{in}}^{\text{jobs}}(s)$ can be very different. The search behaviors that underlie each index are influenced by numerous factors, including the competitiveness of the local housing and job markets, and demographic differences among the migrants themselves, such as whether they are early in their career, or whether they are retired.

In the right side of Figure 4, we plot $\log\left(\rho_{\text{in}}^{\text{housing}}(s) \Big/ \rho_{\text{in}}^{\text{jobs}}(s)\right)$ for each state s using queries from 2015, highlighting which states attract a disproportionate fraction of housing migration searches relative to the proportion of job migration searches they attract, and vice versa. For example, while Florida (+0.83%) and Texas (+0.31%) are both net-positive in domestic migration, their inbound migration queries have very different profiles. Florida attracts disproportionately more housing searches (+0.70 in log ratio), indicating a stronger demand for its local housing market perhaps because of its status as a retirement destination, while Texas attracts more incoming job queries (-0.36 in log ratio), a possible reflection of a stronger demand for its labor market. Similarly, while neighboring states Nevada (+1.00%) and Arizona (+1.17%) are both domestic migration winners, Nevada attracts more job queries (-0.09 in log ratio) and Arizona attracts more housing queries (+0.30 in log ratio).

5 DISCUSSION

Forecasting and understanding large-scale migration patterns has long been a key problem of interest both for academics seeking to better understand the forces that shape society, and for government officials seeking to attract and retain more people to their regions. The impact of migration trends can be massive, with "winning" regions attracting a significant influx of skilled workers, and “losing" regions seeing their talent pool drift away to other shores.

In this work, we explore the use of internet search engine queries as a signal for predicting and understanding domestic migration trends that can augment and improve on government data-based models. The correlations with government data (Table 2, Figures 2, 3) indicate that the search query-based migration intent signals correlate with, and in fact even slightly lead current gold standard migration metrics. This leading relationship is most evident in the net migration flow correlations, in which the correlations between the search query metric and net migration jumps from about 0.10 when contemporaneous to 0.50 to 0.60 when leading official migration statistics. Thus, internet searching appears to be a precursor to migration, which is sensible given that people are likely to research possible migration destinations prior to moving.

The fact that queries tend to lead official metrics motivates our exploration of using search query data to *forecast* migration. Indeed we demonstrate that a migration query measure adds variance explained to regression models that predict future migration. Our model adds more than 10% variance explained over an autoregressive baseline to predictions of *net* migration, which is much harder to forecast than inflow or outflow migration alone. While the boost in variance explained in column 8 of Table 3 is more modest over the more robust baseline shown in column 7, bear in mind that this baseline incorporates many socioeconomic and environmental variables, such as median income and school expenditures, most of which are only available at annual timescales at best. The search query measure still adds variance explained over all of these individual measures, and is available in near real time. Indeed, since government migration data tend to be released two years after the period in question, the fact that search queries can provide a real-time, leading indicator of migration trends is testament to the potential of this method, which provides insights years in advance of any official government tally. Even once government data are released, combining such data with our data provides an even stronger model for predicting subsequent years' migration trends.

Beyond prediction, as shown in Table 4, we illustrate how migration queries shed light on some of the more important questions surrounding aggregated demand for migration. Perhaps the most fundamental question is whether people move for housing or for jobs. Using our migration query taxonomy, we aligned housing and job related migration queries with known demographics of migrants from the ACS, suggesting that younger people and people making the lowest income are more likely to migrate for employment reasons, while conversely older and wealthier people migrate for housing. Furthermore, among people who are searching for housing, older migrants are more likely to be looking to buy property, compared to their younger counterparts who are more likely to be renting. Similar relationships can be seen in looking at the data through he lens of geography. Figure 4 reveals that in states like Florida and Arizona, which attract a considerable number of retirees, incoming migrants are much more likely to be searching for housing than for jobs.

Regarding migration for employment, we highlight how query data can shed light on arguably the most pressing issue of employment today: high-tech versus manual labor jobs. We see a clear split between the less wealthy and less educated versus the wealthier and more highly educated. States attracting migrants in the lowest income bracket are the least likely to see job migration queries in the science and technology sector, while those with the higher income brackets are less likely to see migration queries in the manufacturing and transportation sectors. Education levels reveal a similar story, with science and technology migration queries aligning with high education states, and manufacturing and transportation migration queries aligning with lower education states.

These relationships demonstrate that search query data could be used to forecast the types of housing and employment incoming migrants to a state will seek, which could be extremely valuable for state and even city level planning. For instance, governments through permitting processes and developers could better plan for not only the overall increase or decrease in demand for housing, but whether that housing should be for rent or purchase. These forecasts simply are not feasible with government data, outside of autoregressive approaches that forecast based simply on past trends. Similarly, local governments and business leaders can benefit from forecasting incoming employees across industry sectors, again something not available as leading indicators in government data.

This then, indicates usage scenarios for government planning and policy agencies. At 10% of the population per year, churn in population for any given state is significant, and can drastically impact the social and economic development of a region. States will want to plan and draft housing and economic policies proactively in order to strategically encourage growth in desirable industries, to make housing stock as responsive to changes in demand, and so forth. Predictive policy and economic growth is far from trivial, but at the very least the relevant organizations could be using data that, like we have shown with these search query-based metrics, is at sufficient scale, is leading, and is nuanced in explanation.

6 CONCLUSION

In this paper, we proposed a novel approach that utilizes search query data to forecast and understand domestic migration trends in the United States. We developed a fast and robust migration query filter that achieves 0.88 F1 score. Our migration search data were validated against government data and showed high correlations. More importantly, we demonstrated that adding our migration search data could enhance two types of migration prediction models. Details of our search queries also offer nuanced explanations to how migrants' demands differ demographically and spatially. Overall, our approach will significantly benefit computational social scientists, migration researchers and local policy makers.

REFERENCES

[1] California Losing Residents Via Domestic Migration [EconTax Blog]. (????). https://lao.ca.gov/laoecontax/article/detail/265

[2] LinkedIn Workforce Report | United States | August 2018. (????). https://economicgraph.linkedin.com/resources/linkedin-workforce-report-august-2018

[3] Nikolaos Askitas and Klaus F Zimmermann. 2009. Google econometrics and unemployment forecasting. *Applied Economics Quarterly* 55, 2 (2009), 107–120.

[4] René Böheim and Mark P Taylor. 2002. Tied down or room to move? Investigating the relationships between housing tenure, employment status and residential mobility in Britain. *Scottish Journal of Political Economy* 49, 4 (2002), 369–392.

[5] Ilaria Bordino, Stefano Battiston, Guido Caldarelli, Matthieu Cristelli, Antti Ukkonen, and Ingmar Weber. 2012. Web search queries can predict stock market volumes. *PloS one* 7, 7 (2012), e40014.

[6] Andrei Broder. 2002. A taxonomy of web search. In *ACM Sigir forum*, Vol. 36. ACM, 3–10.

[7] Davide Buscaldi. 2011. Approaches to disambiguating toponyms. *Sigspatial Special* 3, 2 (2011), 16–19.

[8] Davide Buscaldi and Paolo Rosso. 2008. Map-based vs. knowledge-based toponym disambiguation. In *Proceedings of the 5th Workshop on Geographic Information Retrieval*. ACM, 19–22.

[9] Gavin Cameron and John Muellbauer. 1998. The housing market and regional commuting and migration choices. *Scottish Journal of Political Economy* 45, 4 (1998), 420–446.

[10] RJ Cebula and JE Payne. 2005. Net migration, economic opportunity and the quality of life, 1999-2000. *International Review of Economics and Business* 52, 2 (2005), 245–254.

[11] Richard J Cebula. 1979. *The determinants of human migration*. Lexington Books.

[12] Richard J Cebula, Gigi M Alexander, and others. 2006. Determinants of net interstate migration, 2000-2004. *Journal of Regional Analysis and Policy* 36, 2 (2006), 116–123.

[13] Stevie Chancellor and Scott Counts. 2018. Measuring Employment Demand Using Internet Search Data. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 122.

[14] Alberta H Charney. 1993. Migration and the public sector: a survey. *Regional Studies* 27, 4 (1993), 313–326.

[15] Yong Chen and Stuart S Rosenthal. 2008. Local amenities and life-cycle migration: Do people move for jobs or fun? *Journal of Urban Economics* 64, 3 (2008), 519–537.

[16] Hyunyoung Choi and Hal Varian. 2012. Predicting the present with Google Trends. *Economic Record* 88, s1 (2012), 2–9.

[17] Thomas J Courchene. 1974. *Migration, income, and employment: Canada, 1965-68*. CD Howe Research Institute.

[18] Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

[19] Gordon B Dahl. 2002. Mobility and the return to education: Testing a Roy model with multiple markets. *Econometrica* 70, 6 (2002), 2367–2420.

[20] Francesco DâĂŹAmuri and Juri Marcucci. 2010. 'Google it!'Forecasting the US unemployment rate with a Google job search index. (2010).

[21] J Paul Elhorst. 2010. Applied spatial econometrics: raising the bar. *Spatial Economic Analysis* 5, 1 (2010), 9–28.

[22] Michael Ettredge, John Gerdes, and Gilbert Karuga. 2005. Using web-based search data to predict macroeconomic statistics. *Commun. ACM* 48, 11 (2005), 87–92.

[23] Riccardo Faini, Alessandra Venturini, and others. 1994. *Migration and growth: the experience of Southern Europe*. Technical Report. CEPR Discussion Papers.

[24] Lee Fiorio, Guy Abel, Jixuan Cai, Emilio Zagheni, Ingmar Weber, and Guillermo Vinué. 2017. Using twitter data to estimate the relationship between short-term mobility and long-term migration. In *Proceedings of the 2017 ACM on Web Science Conference*. ACM, 103–110.

[25] Sirius Fuller. 2018. Using American Community Survey Estimates and Margins of Error. (April 2018). https://www.census.gov/content/dam/Census/programs-surveys/acs/guidance/training-presentations/20180418_MOE.pdf

[26] US Census Bureau Geography. 2010 Geographic Terms and Concepts - Place. (????). https://www.census.gov/geo/reference/gtc/gtc_place.html

[27] Michael J Greenwood. 1969. An analysis of the determinants of geographic labor mobility in the United States. *The review of Economics and Statistics* (1969), 189–194.

[28] Michael J Greenwood. 1975. Research on internal migration in the United States: a survey. *Journal of Economic Literature* (1975), 397–433.

[29] Emily Gross. 2005. Internal revenue service area-to-area migration data: Strengths, limitations, and current trends. In *Proceedings of the Section on Government Statistics*. 2005.

[30] Andrew Henley. 1998. Residential mobility, housing equity and the labour market. *The Economic Journal* 108, 447 (1998), 414–427.

[31] Lindsay M Howden and Julie A Meyer. 2010. Age and sex composition: 2010. *2010 Census Briefs, US Department of Commerce, Economics and Statistics Administration. US CENSUS BUREAU* (2010).

[32] David Ihrke. Why Did You Move?: An Overview and Analysis of the Annual Social and Economic SupplementâĂŹs Reason for Move Write-In Expansion. SEHSD-WP2016-22 (????). https://www.census.gov/library/working-papers/2016/demo/SEHSD-WP2016-22.html

[33] David Ihrke. 2014. Reason for Moving: 2012 to 2013. *Current population reports. Washington, DC: US Census Bureau* 20 (2014), 574.

[34] David Ihrke. 2017. United States mover rate at a new record low. *Census Blogs. Available at: https://www. census. gov/newsroom/blogs/randomsamplings/2017/01/mover-rate. html* (2017).

[35] Richard Jackman and Savvas Savouri. 1992. Regional migration versus regional commuting: the identification of housing and employment flows. *Scottish Journal of Political Economy* 39, 3 (1992), 272–287.

[36] Geraint Johnes and Thomas Hyclak. 1994. House prices, migration, and regional labor markets. *Journal of Housing Economics* 3, 4 (1994), 312–329.

[37] Kenneth M Johnson, Paul R Voss, Roger B Hammer, Glenn V Fuguitt, and Scott McNiven. 2005. Temporal and spatial variation in age-specific net migration in the United States. *Demography* 42, 4 (2005), 791–812.

[38] John D Kasarda. 1988. Jobs, migration, and emerging urban mismatches. *Urban change and poverty* (1988), 148–198.

[39] John Kennan and James R Walker. 2011. The effect of expected income on individual migration decisions. *Econometrica* 79, 1 (2011), 211–251.

[40] Sari Pekkala Kerr and William R Kerr. 2011. *Economic impacts of immigration: A survey*. Technical Report. National Bureau of Economic Research.

[41] Andy Kiersz. Here's how each US state's population changed between 2016 and 2017 because of people moving in and out. (????). https://www.businessinsider.com/state-domestic-migration-map-2016-to-2017-2018-1

[42] Robert J LaLonde and Robert H Topel. 1997. Economic impact of international migration and the economic performance of migrants. *Handbook of population and family economics* 1 (1997), 799–850.

[43] Pierre Legendre. 1993. Spatial autocorrelation: trouble or new paradigm? *Ecology* 74, 6 (1993), 1659–1673.

[44] Steven Manson, Jonathan Schroeder, David Van Riper, and Steven Ruggles. 2017. IPUMS National Historical Geographic Information System: Version 12.0 [Database]. *Minneapolis: University of Minnesota* (2017).

[45] Warren F Mazek and John Chang. 1972. The chicken or egg fowl-up in migration: comment. *Southern Economic Journal* (1972), 133–139.

[46] Nick McLaren and Rachana Shanbhogue. 2011. Using internet search data as economic indicators. (2011).

[47] Johnnatan Messias, Fabricio Benevenuto, Ingmar Weber, and Emilio Zagheni. 2016. From migration corridors to clusters: The value of Google+ data for migration studies. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE Press, 421–428.

[48] Raven Molloy, Christopher L Smith, and Abigail Wozniak. 2011. Internal migration in the United States. *Journal of Economic perspectives* 25, 3 (2011), 173–96.

[49] Richard F Muth. 1971. Migration: chicken or egg? *Southern Economic Journal* (1971), 295–306.

[50] 2017-2018 Session of the Vermont State Senate. 2017. S.94, Act 257. An act relating to promoting remote work. (2017). http://www.leg.state.vt.us/jfo/fiscal_notes/2018_S_94_fiscal_note_house_commerce.pdf

[51] Adam Pasick. Facebook's big data glimpse at human migration and the growth of mega-cities. (????). https://qz.com/159279/facebooks-big-data-glimpse-at-human-migration-and-the-growth-of-mega-cities/

[52] Jaroslav Pavlicek and Ladislav Kristoufek. 2015. Nowcasting unemployment rates with google searches: Evidence from the visegrad group countries. *PloS one* 10, 5 (2015), e0127084.

[53] William Petersen. 1969. *Population*. London: MacMillan Co.

[54] Tobias Preis, Helen Susannah Moat, and H Eugene Stanley. 2013. Quantifying trading behavior in financial markets using Google Trends. *Scientific reports* 3 (2013), 1684.

[55] Tobias Preis, Daniel Reith, and H Eugene Stanley. 2010. Complex dynamics of our economic life on different scales: insights from search engine query data. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 368, 1933 (2010), 5707–5719.

[56] Stephen M Renas and Rishi Kumar. 1978. The cost of living, labor market opportunities, and the migration decision: A case of misspecification? *The Annals of regional science* 12, 2 (1978), 95–104.

[57] Aba Schwartz. 1976. Migration, age, and education. *Journal of Political Economy* 84, 4, Part 1 (1976), 701–719.

[58] Simon Sheather. 2009. *A modern approach to regression with R*. Springer Science & Business Media.

[59] Robert Shimer. 2007. Mismatch. *American Economic Review* 97, 4 (2007), 1074–1101.

[60] David A Smith and Gregory Crane. 2001. Disambiguating geographic names in a historical digital library. In *International Conference on Theory and Practice of Digital Libraries*. Springer, 127–136.

[61] Paul M Sommers and Daniel B Suits. 1973. Analysis of net interstate migration. *Southern Economic Journal* (1973), 193–201.

[62] Donald N Steinnes. 1978. Causality and Migration: A Statistical Resolution of the" Chicken or Egg Fowl-up". *Southern Economic Journal* (1978), 218–226.

[63] Lyman Stone. 2017. Reviewing the 2017 Census Population Estimates. (Dec. 2017). https://medium.com/migration-issues/reviewing-the-2017-census-population-estimates-62a8a9ad5d17

[64] James D Tarver and R Douglas McLeod. 1973. A test and modification of ZipfâĂŹs hypothesis for predicting interstate migration. *Demography* 10, 2 (1973), 259–275.
[65] Jacob Thebault-Spieker, Loren Terveen, and Brent Hecht. 2017. Toward a geographic understanding of the sharing economy: Systemic biases in UberX and TaskRabbit. *ACM Transactions on Computer-Human Interaction (TOCHI)* 24, 3 (2017), 21.
[66] Antonio Toral and Rafael Munoz. 2006. A proposal to automatically build and maintain gazetteers for Named Entity Recognition by using Wikipedia. In *Proceedings of the Workshop on NEW TEXT Wikis and blogs and other dynamic text sources*.
[67] Ingmar Weber, Emilio Zagheni, and others. 2013. Studying inter-national mobility through IP geolocation. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 265–274.
[68] Lynn Wu and Erik Brynjolfsson. 2015. The future of prediction: How Google searches foreshadow housing prices and sales. In *Economic analysis of the digital economy*. University of Chicago Press, 89–118.
[69] Jeffrey E Zabel. 2012. Migration, housing market, and labor market responses to employment shocks. *Journal of Urban Economics* 72, 2-3 (2012), 267–284.
[70] Emilio Zagheni, Venkata Rama Kiran Garimella, Ingmar Weber, and others. 2014. Inferring international and internal migration patterns from twitter data. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 439–444.
[71] Emilio Zagheni and Ingmar Weber. 2012. You are where you e-mail: using e-mail data to estimate international migration rates. In *Proceedings of the 4th annual ACM web science conference*. ACM, 348–351.

Learning Dual Retrieval Module for Semi-supervised Relation Extraction

Hongtao Lin
University of Southern California
Los Angeles, California, USA
lin498@usc.edu

Jun Yan
Tsinghua University
Beijing, China
j-yan15@mails.tsinghua.edu.cn

Meng Qu
University of Illinois at Urbana-Champaign
Champaign, Illinois, USA
mengqu2@illinois.edu

Xiang Ren
University of Southern California
Los Angeles, California, USA
xiangren@usc.edu

ABSTRACT

Relation extraction is an important task in structuring content of text data, and becomes especially challenging when learning with weak supervision—where only a limited number of labeled sentences are given and a large number of unlabeled sentences are available. Most existing work exploits unlabeled data based on the ideas of self-training (i.e., bootstrapping a model) and self-ensembling (e.g., ensembling multiple model variants). However, these methods either suffer from the issue of semantic drift, or do not fully capture the problem characteristics of relation extraction. In this paper, we leverage a key insight that *retrieving sentences expressing a relation* is a dual task of *predicting the relation label for a given sentence*—two tasks are complementary to each other and can be optimized jointly for mutual enhancement. To model this intuition, we propose DualRE, a principled framework that introduces a retrieval module which is jointly trained with the original relation prediction module. In this way, high-quality samples selected by the retrieval module from unlabeled data can be used to improve the prediction module, and vice versa. Experimental results[1] on two public datasets as well as case studies demonstrate the effectiveness of the DualRE approach.

KEYWORDS

relation extraction, semi-supervised learning, dual learning

ACM Reference Format:
Hongtao Lin, Jun Yan, Meng Qu, and Xiang Ren. 2019. Learning Dual Retrieval Module for Semi-supervised Relation Extraction. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, Article 4, 11 pages. https://doi.org/10.1145/3308558.3313573

1 INTRODUCTION

Relation extraction (RE) plays a key role in turning massive, unstructured text corpora into structures of factual knowledge, and has a wide range of downstream applications in many domains [19, 28, 34]. Given two entities in a sentence, the goal of (sentence-level) relation extraction is to predict the relation expressed between two entities based on their sentence context. For example, in Fig. 1, we aim to classify each sentence into several relation labels, such as "org:founded_by", "per:parents", and "no_relation". While many works have been done on developing supervised relation extraction models [4, 24, 36, 38, 39], they typically require a large amount of human-annotated data (i.e., sentences that are manually assigned with the appropriate relation labels) for model training. Despite its wide applications, the process of manually annotating a large training set with many relation labels is too expensive and error-prone. The rapid emergence of large, domain-specific text corpora (e.g., social media content, customer reviews, scientific publications) calls for methods that can automatically extract entity relationships from text with *less human supervision*.

Sentence	Relation
Steve Jobs and Wozniak co-founded **Apple** in 1976.	org:founded_by
He was on Safari in South Africa with his mother, **Trudy**.	per:parents
Trump believes the **United States** has incredible potential.	no_relation

Figure 1: Sentence-level Relation Extraction Task. This task concerns about predicting the relation of two entities mentioned in a sentence. Tokens in orange and blue are subject and object entities in correspondence.

In recent years there has been a surge of interest in applying *distant supervision* to relation extraction [19, 22, 37, 38]. The general idea is to leverage facts stored in external knowledge bases (KBs) to annotate sentences in an unlabeled corpus as training data—two entities co-occurring in a sentence will be labeled with their KB relations regardless of the specific context. Although distant supervision automates the process of labeling sentences, it suffers from two major limitations. First, not all sentences where two entities co-occur are expressing their KB relations (i.e., context-agnostic label noise). Second, in some application domains the target relations may not be found in existing knowledge bases (e.g., WikiData), making it hard to apply distant supervision for label generation. It is thus desirable to develop methods that can rely on less human-annotated data for training relation extraction models. This motivates the study of *semi-supervised relation extraction* methods [1, 3, 31], which seek to leverage a limited number of labeled

[1]Code and data can be found at https://github.com/ink-usc/DualRE.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313573

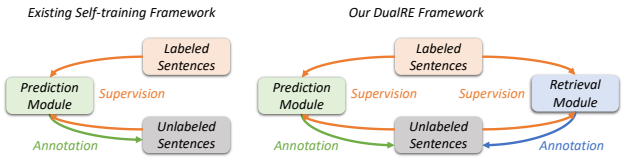


Figure 2: Illustration of the differences between the proposed DualRE framework and existing Self-training framework. Left: Self-training framework uses a single prediction module to annotate unlabeled sentences for generating additional labeled data. Right: DualRE framework leverages a prediction module and a retrieval modules for generating labeled data, where the newly labeled sentences are used to improve both modules. Two modules are jointly optimized and mutually enhance each other.

sentences and a large unlabeled (background) corpus for model training. Such setting has broad use cases, including cold-start knowledge base construction[2] and question answering [11].

In this paper, we study the problem of semi-supervised relation extraction. Towards this goal, there are broadly two kinds of methods, i.e., *self-ensembling* methods, and *self-training* methods. The main idea behind self-ensembling methods [12, 23, 27, 33] is to impose the assumption of "consistency under perturbation"—the prediction on the unlabeled instances should remain unchanged under small perturbations either on the data, or on the model parameters. In contrast, self-training methods [25, 29] aim to incrementally promote high-confidence predictions over unlabeled instances into the labeled training set, and improve the model by re-training with the updated labeled set. However, these methods suffer from the limitations as follows. (1) *Insufficient Supervision*: The process of enforcing prediction consistency between model variants in self-ensembling methods does not involve promoting new labeled instances from the unlabeled set. This makes it hard to improve models when "consistency-based" supervision is relatively weak compared to that from actual instances. (2) *Semantic Drift* [9]: While the self-training methods annotates unlabeled sentences as extra training data, the incremental promotion process can be biased (i.e., provides incorrect predictions), and may accumulate errors (i.e., including falsely-labeled instances into the training set). To overcome aforementioned challenges, we aim to design an approach that can generate high-quality labeled instances (as additional supervision) from the unlabeled corpus in an effective manner.

Intuitively, *retrieving sentences for a given relation* can be seen as a dual problem of *predicting the relation expressed in a sentence* (see the right-hand side of Fig. 2 for illustration). Models for the retrieval problem can help obtain more relevant sentences to a given relation from an auxiliary unlabeled corpus, for improving models for the prediction problem; while models for the prediction problem can predict the relation labels of the unlabeled sentences, and therefore help determine the quality of relevant sentences to collect (under each relation) for benefiting retrieval models. Such an observation of "problem duality" motivates us to solve the primal and dual problems jointly, and leverage the duality to overcome the aforementioned challenges. Specifically, as the prediction and retrieval models annotate/retrieve unlabeled sentences from the auxiliary corpus, both models provide additional training data to each other and thus help resolve the insufficient supervision issue. Meanwhile, during the joint learning process, both models provide feedback to correct the output of each other, helping generate high-quality labeled data and resolve semantic drift issue.

To model the above ideas, we propose a novel framework, called **DualRE**, for semi-supervised relation extraction. DualRE consists of a prediction module and a retrieval module (see Fig. 3 for illustration of the model design). Given a sentence x, the prediction module aims to determine its relation label y with a neural relation extraction model by approximating $p(y|x)$ (e.g., position-aware LSTM [39]). Meanwhile, the retrieval module employs a neural learning-to-rank model to approximate $p(x|y)$ (e.g., using pointwise or pairwise loss) to facilitate ranking sentences $\{x\}$ to identify relevant ones for a query relation y. During training, the two modules are jointly optimized to provide feedback for each other, by alternating between two processes as follows. In the first process, we fix the prediction module, and use additional training data promoted by both modules to update the retrieval module. Similarly, in the second process, the retrieval module is fixed, and we generate extra training data using both modules and update the prediction module. By doing so, the retrieval module can consistently provide additional supervision to improve the prediction module. As the two modules are jointly leveraged to promote new training data (via referencing the label distribution), it enables the prediction and retrieval module to mutually correct each other and generate high-quality annotation.

We conduct extensive experiments on two public relation extraction datasets to evaluate DualRE[3]. Experimental results show that DualRE consistently outperforms several state-of-the-art methods under different data settings (by varying the size of labeled data and unlabeled data). Detailed case study demonstrates that DualRE is able to select quality instances to guide the model training.

In summary, in this paper we make the following contributions:

- We propose a novel framework for semi-supervised relation extraction, which encourages the (primal) prediction task and the (dual) retrieval task to mutually benefit each other by generating additional training instances from an unlabeled corpus.
- We develop a joint learning algorithm to alternatively optimize the prediction and retrieval modules, such that they can collaborate with each other and improve the quality of annotations.
- We conduct extensive experiments on two public datasets. Experimental results and case studies validate the effectiveness of the proposed framework.

2 BACKGROUND AND PROBLEM

In this section, we introduce the relevant concepts and notations, and formally define semi-supervised relation extraction task.

Entity Mention. An entity mention e is a token span (or word sequence) that refers to a real-world entity. In Fig. 1, the word "Apple" is an entity mention that maps to the company *Apple Inc.*.

[2]https://tac.nist.gov/2017/KBP/ColdStart/index.html

[3]Source code: https://github.com/INK-USC/DualRE

Relation Mention and Label. A relation mention consists of two entity mentions and the sentence x they co-occur in (i.e., the two entity mentions are sub-strings of the sentence). Here we consider entity relations as directional connections between two entities. Therefore, we denote the left entity as subject entity $e_s = (x_{s_1}, ..., x_{s_p})$ and the right entity as object entity $e_o = (x_{o_1}, ..., x_{o_q})$, where s_i and o_j are the token indexes of subject and object entities in the sentence, respectively. A relation mention (x, e_s, e_o) can be assigned with a relation label $r \in \mathcal{R}$ (i.e., e_s and e_o express relation r in the sentence x). We use $\mathcal{R}$ to denote the pre-defined set of relations of interests in the downstream application. In particular, undefined or irrelevant relations are assigned with a special label "no_relation".

Problem Definition. In this paper, our goal is to learn a model to effectively predict the relation label between two entities e_s and e_o for a relation mention (x, e_s, e_o). Fig. 1 presents some examples for the task. Each row demonstrates a relation mention with two entities highlighted in blue (subject entity) and red (object entity). For relation mentions that are not expressing target relations of interests (or do not express any relation) like the third example, we assign "no_relation" as a special label in the relation label set. Moreover, the third example also demonstrates the difference between *sentence-level* and *instance-level* relation extraction. Despite the well-known fact that *Donald Trump* is the president of *United States*, which can be extracted from multiple sentences in the corpus, we cannot infer such relation from this specific sentence and thus should return a label that is the most appropriate for the given sentence. Formally, we define the task of sentence-level relation extraction as follows.

Definition 2.1. **(Sentence-level Relation Extraction)** Given a sentence $x = (x_1, x_2, ..., x_n)$, a subject entity e_s and an object entity e_o in the sentence, the task of sentence-level relation extraction is to predict the relation label $r \in \mathcal{R}$ given the sentence and two entities, where $\mathcal{R}$ is a pre-defined set of relations including "no_relation".

Under the semi-supervised learning setting, we aim to learn the relation extraction model using both *labeled relation mentions* (which are manually annotated) and *unlabeled relation mentions* (of which entity mentions are identified by some existing named entity recognition tools). Formally, let $\mathbf{x} = (x, e_s, e_o)$ denote a relation mention and y denote its corresponding relation label r from the set $\mathcal{R}$. We define the task of semi-supervised relation extraction as follows.

Definition 2.2. **(Semi-supervised Relation Extraction)** Given a set of labeled relation mentions $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N_L}$ and a set of unlabeled relation mentions $U = \{\mathbf{x}_j\}_{j=1}^{N_U}$, semi-supervised relation extraction task aims to learn a multi-class classification model f that (1) fits the labeled training data L; and (2) captures the information in the unlabeled data U, at the same time.

In the following sections, we stick with the denotation of $\mathbf{x}$ and y when referring to the relation mention (x, e_o, e_s) and relation r.

Our Focus. In this work, we assume entity mentions have been recognized from text using existing tools. In our experiments, the entity mentions are provided by the public datasets [15, 39]. Similarly, the set of relations $\mathcal{R}$ is also given by the datasets. In particular, we focus on the cases that the labeled instances are manually annotated with reliable labels, as compared to the noisy labels obtained via distant supervision [22]. Here we do not consider applying distant supervision to generate pseudo labels over the unlabeled relation mentions, and leave this as future work.

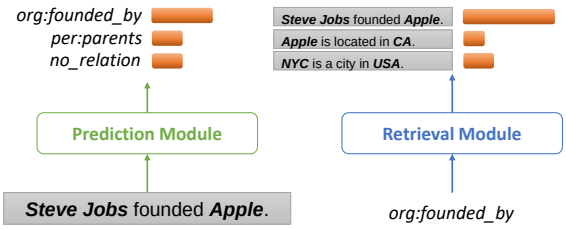


Figure 3: Illustration of the Two Modules in DualRE. Left: The prediction module focuses on the primal task, that is, predicting the relation label y of a sentence x. Right: The retrieval module focuses on the dual task, which aims at retrieving relevant sentences x for a query relation y.

3 THE DUALRE FRAMEWORK

Framework Overview. This section introduces our dual learning approach to semi-supervised relation extraction. Given that *retrieving sentences expressing a relation* can be seen as a dual task of *predicting the relation expressed in a sentence*, we propose DualRE to take advantage of the dual property of two tasks and overcome the challenges mentioned in the introduction. At a high level, DualRE trains a module for the prediction task and a module for the retrieval task jointly. The retrieval module helps obtain relevant sentences for each relation from an auxiliary unlabeled corpus, which are then used as additional training data for the prediction module. Meanwhile, the prediction module predicts the relation labels for the unlabeled sentences, and determines the relation labels for retrieved sentences, which then serve as training data to improve the retrieval module. Two modules are jointly optimized to mutually enhance each other, generating additional, high-quality labeled data to address the problems of insufficient supervision and semantic drift.

Formally, our framework consists of a prediction module $\mathcal{P}_\theta$ and a retrieval module $\mathcal{Q}_\phi$, where θ and ϕ are their model parameters. The prediction module focuses on the primal task, i.e., predicting the relation label for a given relation mention. Therefore, for a mention-label pair $(\mathbf{x}, y)$, it models the conditional probability $p_\theta(y|\mathbf{x})$. By contrast, the retrieval module solves the dual task, i.e., retrieving relevant sentences expressing a specific relation, and hence it defines the probability $q_\phi(\mathbf{x}|y)$ for a mention-label pair $(\mathbf{x}, y)$. In practice, since $q_\phi(\mathbf{x}|y) \propto q_\phi(\mathbf{x}, y)$ when given a specific relation y, the retrieval module will focus on estimating $q_\phi(\mathbf{x}, y)$ instead. The overall objective function is given below:

$$\mathbf{O} = \mathbf{O}_P + \mathbf{O}_R + \mathbf{O}_U \tag{1}$$

There are three terms in total. In $\mathbf{O}_P$ and $\mathbf{O}_R$, we use the given labeled sentences to train the prediction module and the retrieval module, respectively. In the unsupervised part $\mathbf{O}_U$, we promote the collaboration of the two modules to annotate unlabeled sentences as additional training data. In the rest of this section, we will introduce the details of each key component in DualRE.

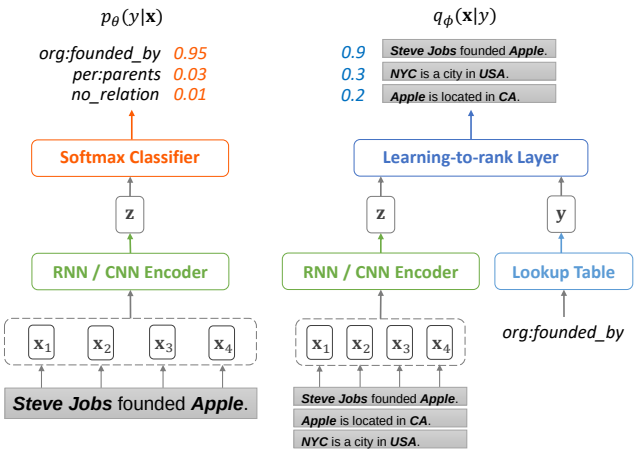


Figure 4: Model Design of the Two Modules in DualRE. Left: The prediction module encodes the relation mention and outputs a probability distribution over all relations. Right: The retrieval module ranks the relation mentions given a query by predicting a relevance score $q_\phi(\mathbf{x}, y)$. Retrieving instances given a relation is essentially taking the top instances from the ranking list. Note that this figure does not illustrate how two modules mutually enhance each other using unlabeled data.

3.1 Relation Prediction Module

In the DualRE framework, the prediction module $\mathcal{P}_\theta$ focuses on the primal task, that is, predicting the relation expressed in a relation mention, as shown in Fig. 4. We build the prediction module by first leveraging an encoder to encode the relation mention, and then adopting a softmax classifier to perform relation classification.

Relation Mention Encoder. The goal of relation mention encoders is to encode relation mentions into latent vectors, which can serve as features for prediction. A variety of encoders have been proposed for relation mention encoding, such as Long Short-Term Memory (LSTM) encoders [17], Piecewise Convolutional Neural Network (PCNN) encoders [37], and Position-aware Recurrent Neural Network (PRNN) encoders [39]. Among them, PRNN outperforms all other encoders on relation extraction, and therefore we use PRNN as our default encoder, but our framework can also adopt other encoders.

Formally, given a relation mention, PRNN considers the POS tagging, NER tagging, the position of each token (with regard to object and subject entities) as features. Then an LSTM cell is leveraged to learn the hidden representation of each word. Finally, the attention mechanism is utilized to weighted aggregate different hidden representations as the final encoding of the relation mention.

Objective Function. After encoding a relation mention, we feed it into a softmax classifier to predict the relation label of the mention. The prediction module can be trained using the given labeled data, in which we leverage the cross-entropy objective function, which is given below:

$$\mathbf{O}_P = \mathbb{E}_{(\mathbf{x}, y)\in L}[\log p_\theta(y|\mathbf{x})] \tag{2}$$

Given a relation mention $\mathbf{x}$ and its relation label y, the prediction module seeks to maximize the probability of y for the mention $\mathbf{x}$.

3.2 Sentence Retrieval Module

The intended behavior of retrieval module Q_ϕ is to select a set of relevant mentions given specific relations. In information retrieval domain, this is similar to the problem of retrieving a set of documents given a query. In our case, "document" and "query" correspond to relation mention and relation, respectively.

Learning-to-rank [20] is a type of model that aims to rank the documents according to its relevance to a query, thus we can select the top-ranking ones as retrieved documents. Learning-to-rank formulates such problems using joint probability $q_\phi(\mathbf{x}, y)$, which is in proportion to $q_\phi(\mathbf{x}|y)$ when the probability of observing query $q(y)$ is fixed. Similar to the prediction module, we can train the retrieval module by leveraging the labeled data and optimizing the following objective function:

$$\mathbf{O}_R = \mathbb{E}_{(\mathbf{x}, y)\in L}[\log q_\phi(\mathbf{x}, y)] \tag{3}$$

However, directly optimizing $q_\phi(\mathbf{x}, y)$ is intractable, since we have to ensure $\sum_{(\mathbf{x}, y)} q_\phi(\mathbf{x}, y) = 1$, which requires traversing over all possible pairs of $(\mathbf{x}, y)$ to calculate the partition function. Therefore, following existing learning-to-rank models, we seek not to enforce $q_\phi(\mathbf{x}, y)$ as a valid distribution. Instead, we let $q_\phi(\mathbf{x}, y)$ output a score to estimate the degree of *relevance* of the pair $(\mathbf{x}, y)$. In literature, there are mainly two types of approaches to train such retrieval models, i.e., the pointwise approach [8, 18] and the pairwise approach [5, 6], and we tried both in our experiment. Next, we introduce these approaches respectively.

Pointwise Approach. Pointwise learning-to-rank models assume that each query-document pair is assigned a score that reflects their degree of relevance, which can be used to determine the rank for each document given a query.

In the task of retrieving relation mentions, for each relation mention $\mathbf{x}$, we have a label y to indicate which relation it belongs to. Alternatively, we can regard such supervision signal as a binary score (0 or 1) that indicates whether a relation mention $\mathbf{x}$ is relevant to a specific relation $y \in \mathcal{R}$. In other words, each relation mention in the labeled dataset is paired up with all relations, with a binary label to indicate if they are relevant.

The model design for the learning-to-rank retrieval module is illustrated in Fig. 4. Given a relation y and a set of relation mentions $\{\mathbf{x}\}$ to be retrieved, we first encode each relation mention $\mathbf{x}$ as a vector $\mathbf{z}$. Each relation y is mapped to its relation embedding $\mathbf{y}$ according to a relation embedding matrix R, which has the same dimension as $\mathbf{z}$. In the learning-to-rank layer, pointwise approach takes the inner product $q_\phi(\mathbf{x}|y) = \sigma(\mathbf{z}^\top \mathbf{y})$ as the relevance score, where $\sigma(\cdot)$ denotes the sigmoid activation function.

$$\mathbb{E}_{(\mathbf{x}, y)\in L}[\log \sigma(\mathbf{z}^\top \mathbf{y})] + \mathbb{E}_{(\mathbf{x}, y')\notin L}[\log(1 - \sigma(\mathbf{z}^\top \mathbf{y}'))], \tag{4}$$

where $(\mathbf{x}, y)$ is a pair in the labeled dataset L, $(\mathbf{x}, y')$ pairs up the relation mention $\mathbf{x}$ with incorrect relation y', $\mathbf{z}$ is the mention encoding for $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{y}'$ are embeddings of relation y and y'.

Pairwise Approach. Pairwise learning-to-rank models assume that a relative order can be determined between the retrieved documents. For a set of documents, we can compare any pair of them

and predict their relative order. We expect the model predictions (i.e. the set of relative orders) are consistent (no conflicting results such as $a > b, b > c$ but $c < a$), and thus the absolute order of each document in the ranked list can be derived.

The core part of pairwise learning-to-rank models is the design of objective function that can characterize the relative order of paired documents. We adopt the approach in RankNet [5], which proposes an objective function to minimize the number of inversions in ranking. An inversion happens when we rank a lower rated document above a higher rated one in a ranked list.

In the task of retrieving relation mentions, given a relation, we aim to rank all relevant mentions higher than the irrelevant ones. Similar to the design of pointwise learning-to-rank, we also encode each relation mention $\mathbf{x}$ and relation y into vector representation $\mathbf{z}$ and $\mathbf{y}$ respectively. The only difference is how the objective function in the learning-to-rank layer (as shown in Fig. 4) is calculated. We approximate O_R in Eq. 3 using the following term:

$$\mathbb{E}_{(\mathbf{x},y),(\mathbf{x}',y')\in L\times L}[r(\mathbf{x},\mathbf{x}',y)\log\sigma(\mathbf{z}^T\mathbf{y}-\mathbf{z}'^T\mathbf{y})], \tag{5}$$

where the order score $r(\mathbf{x},\mathbf{x}',y)$ expresses the relative order of two relation mentions, given the relation y. Consider $\mathbf{x}$ as a positive instance for relation y, the order score will return 1 if $\mathbf{x}'$ is a negative instance. During optimization process, this indicates that we attribute $\mathbf{z}^T y$ a higher score than $\mathbf{z}'^T y$. If $\mathbf{x}'$ happens to be another positive instance, we cannot make assertion about their relative order, then the order score returns 1/2. During the optimization process, this indicates that we assume the score $\mathbf{z}^T y$ and $\mathbf{z}'^T y$ are close.

3.3 Interaction Between the Two Modules

So far, we only introduced the formulation of prediction and retrieval modules and how they can be trained in a supervised way (by maximizing O_P and O_R). In the semi-supervised setting, we try to incorporate unlabeled data, which leads to the introduction of O_U in overall objective function. In DualRE, the formulation of this unsupervised objective function involves both prediction and retrieval modules. Intuitively, we hope to leverage the unlabeled data under the collaboration between both modules.

Formally, in $\mathbf{O}_U$, we seek to maximize the log-likelihood function of the unlabeled relation mentions:

$$\begin{aligned}\mathbf{O}_U &= \mathbb{E}_{\mathbf{x}\in U}[\log q_\phi(\mathbf{x})] \\ &\geq \mathbb{E}_{\mathbf{x}\in U, y\sim p_\theta(y|\mathbf{x})}[\log\frac{q_\phi(\mathbf{x},y)}{p_\theta(y|\mathbf{x})}]\end{aligned} \tag{6}$$

The above inequality is derived from the EM algorithm [10]. The two sides are equal when $p_\theta(y|\mathbf{x}) = q_\phi(y|\mathbf{x})$. By maximizing the lower bound of O_U, we incorporate both modules in the optimization process. As we will see in Sec. 4, we approach the optimization process by sampling from unlabeled data and accepting the instances that both modules agree on. These newly retrieved instances are regarded as pseudo-labeled data and enhance the training of both modules. Such process effectively leverages the selection quality of retrieved instances and boosts the overall model performance.

4 JOINT LEARNING OF PREDICTION AND RETRIEVAL MODULES

In Sec. 3, we introduce the formulation for three components (O_P, O_R and O_U) of the overall objective function. This section will present the details on how we approach the optimization of the overall objective function O, and the learning algorithm.

4.1 The EM-based Joint Optimization

To optimize the objective function O, we follow the EM algorithm to alternatively optimize the prediction and retrieval modules.

E Step. In this step, we update the prediction module $\mathcal{P}_\theta$ by fixing the retrieval module $\mathcal{Q}_\phi$. This corresponds to the minimization of KL divergence $KL(p_\theta(y|\mathbf{x})||q_\phi(y|\mathbf{x}))$. Directly minimizing this objective function is difficult, thus we follow the wake-sleep algorithm [16] to minimize the reversed KL divergence $KL(q_\phi(y|\mathbf{x})||p_\theta(y|\mathbf{x}))$, which yields the same optimal solution, i.e., $p_\theta(y|\mathbf{x}) = q_\phi(y|\mathbf{x})$. By integrating Eq. 2 and Eq. 6 as the overall objective function, and taking derivative with respect to θ, the gradient for θ is calculated as:

$$\begin{aligned}\nabla_\theta\mathbf{O} &= \mathbb{E}_{(\mathbf{x},y)\in L}[\nabla_\theta\log p_\theta(y|\mathbf{x})] \\ &\quad+\mathbb{E}_{\mathbf{x}\in U, y\sim q_\phi(y|\mathbf{x})}[\nabla_\theta\log p_\theta(y|\mathbf{x})],\end{aligned} \tag{7}$$

where the gradient is calculated from both the labeled data and additional data annotated by the retrieval module $\mathcal{Q}_\phi$. However, only leveraging the retrieval module for annotation can be biased, so we may expect both modules to mutually correct each other. Towards this goal, we notice that $\mathbb{E}_{p_\theta(y|\mathbf{x})}[\nabla_\theta\log p_\theta(y|\mathbf{x})] = 0$ [4], and therefore the gradient can be modified as:

$$\begin{aligned}\nabla_\theta\mathbf{O} &= \mathbb{E}_{(\mathbf{x},y)\in L}[\nabla_\theta\log p_\theta(y|\mathbf{x})] \\ &\quad+\mathbb{E}_{\mathbf{x}\in U, y\sim(q_\phi(y|\mathbf{x})+p_\theta(y|\mathbf{x}))}[\nabla_\theta\log p_\theta(y|\mathbf{x})]\end{aligned} \tag{8}$$

Although the gradient remains the same as in Eq. 7, the new gradient enables us to leverage both modules to collect additional labeled sentences, which usually has lower noise, and therefore leads to better performance.

M Step. In this step, we update the retrieval module $\mathcal{Q}_\phi$ by fixing the prediction module $\mathcal{P}_\theta$. The gradient with respect to ϕ can be similarly calculated as:

$$\begin{aligned}\nabla_\phi\mathbf{O} &= \mathbb{E}_{(\mathbf{x},y)\in L}[\nabla_\phi\log q_\phi(\mathbf{x},y)] \\ &\quad+\mathbb{E}_{(\mathbf{x},y)\sim(p_\theta(\mathbf{x},y)+q_\phi(\mathbf{x},y))}[\nabla_\phi\log q_\phi(\mathbf{x},y)]\end{aligned} \tag{9}$$

Again, directly calculating $\log q_\phi(\mathbf{x},y)$ is infeasible, and therefore in practice we use the pointwise term (Eq. 4) or the pairwise objective function (Eq. 5) to replace $\log q_\phi(\mathbf{x},y)$.

To summarize, the model alternates between optimizing the objective function with regard to ϕ and θ, which is referred to as E and M step. During each step, we incorporate both labeled and unlabeled data. Both E and M step require sampling annotated instances from unlabeled data. Noticing that samples from distribution $\mathbf{x}\in U, y\sim(q_\phi(y|\mathbf{x})+p_\theta(y|\mathbf{x}))$ is equivalent to sampling from distribution $(\mathbf{x},y)\sim(p_\theta(\mathbf{x},y)+q_\phi(\mathbf{x},y))$, we can keep a single version of sampled instances and apply to both steps.

[4] $\mathbb{E}_{p_\theta(y|\mathbf{x})}[\nabla_\theta\log p_\theta(y|\mathbf{x})] = \mathbb{E}_{p_\theta(y|\mathbf{x})}[\frac{\nabla_\theta p_\theta(y|\mathbf{x})}{p_\theta(y|\mathbf{x})}] = \sum\nabla_\theta p_\theta(y|\mathbf{x}) = \nabla_\theta\sum p_\theta(y|\mathbf{x}) = \nabla_\theta 1 = 0$

Next, we introduce an empirical sampling method that effectively generates high-quality training relation mentions (as additional supervision) to improve the performance.

4.2 Instance Selection

Empirically, sampling from distribution $p_\theta(\mathbf{x}, y) + q_\phi(\mathbf{x}, y)$ can be approximated by taking the intersection set of predicted results from both modules.

In prediction module $\mathcal{P}_\theta$, we first select new instances from distribution $p_\theta(\mathbf{x}, y)$. If we assume a uniform distribution for relation mentions $p(\mathbf{x})$, we are essentially sampling from distribution $p_\theta(y|\mathbf{x}) \propto p_\theta(\mathbf{x}, y)$. Given the unlabeled instance $\mathbf{x}$ and its predicted relation y, we rank these instances according to the probability of predicting y (i.e., $p(y|\mathbf{x})$) and take top-k instances from the ranked list. This is similar to what we do in self-training models.

In retrieval module $\mathcal{Q}_\phi$, we sample k instances from distribution $q_\phi(\mathbf{x}, y) = q_\phi(\mathbf{x}|y)q(y)$. The first term $q_\phi(\mathbf{x}|y)$ aligns with the intended behavior for the retrieval module: given a relation y, the retrieval module traverses through each unlabeled instance $\mathbf{x}$ and gives out a relevance score $q_\phi(\mathbf{x}, y) \propto q_\phi(\mathbf{x}|y)$. Therefore we can construct a ranked list for each relation y, in which top-k_y instances are retrieved for this relation. We aggregate the instances from each relation to construct the retrieved set. The second term $q(y)$ can be regarded as prior knowledge that constrains the relation distribution for retrieved instances. Given that we retrieved k instances which follows the reference relation distribution $q(y)$, we can determine the number of instances retrieved for each relation $k_y = kq(y)$.

Both modules collect a set of training instances, and the intersection of them are regarded as the final set. Intuitively, this is similar to majority voting or ensembling that considers multiple models when doing prediction.

4.3 Instance Promotion

To further eliminate the influence of incorrect annotations in retrieved instances, we experiment with two strategies to incorporate these instances in the training phase.

Equal promotion. Equal promotion treats the retrieved instances and original training data equally when calculating the loss function. This is the typical strategy for self-training models. The objective functions derived in Eq. 8 and 9 reflect this strategy.

Weighted promotion. In this strategy, each instance is assigned a weight π reflecting the confidence of its annotation, which is intended to be low when it's wrongly labeled. The weights are incorporated in training phase when promoting different instances. Considering Eq. 7 when training the prediction module, the first term is intended for promoting instances in the labeled set, which is assigned a weight of 1 and the formula remains unchanged. The second term is modified as follows:

$$\mathbb{E}_{(\mathbf{x},y)\sim(p_\theta(\mathbf{x},y)+q_\phi(\mathbf{x},y))}[\pi^p_{\mathbf{x}} \nabla_\phi \log q_\phi(\mathbf{x}, y)], \tag{10}$$

where $\pi^p_{\mathbf{x}}$ is the weight for pseudo-labeled instance ($\mathbf{x}$, y). Similarly, we introduce the weight $\pi^q_{\mathbf{x}}$ in objective function O_R for the retrieval module. In our experiment, we parameterize the weights as $\pi^p_{\mathbf{x}} = p_\theta(y|\mathbf{x})^\alpha$ and $\pi^q_{\mathbf{x}} = q_\phi(\mathbf{x}, y)^\beta$, in which α and β are hyperparmeters to tune in our model.

Algorithm 1: DualRE Learning Algorithm.

Input: Labeled data $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N_L}$, unlabeled data $U = \{\mathbf{x}_j\}_{j=1}^{N_U}$, the amount of data to retrieve each in each iteration k.

Initialize: $L_U \leftarrow \emptyset$.

$P_\theta, Q_\phi \leftarrow$ Pretrain prediction and retrieval module using L.

while $U \neq \emptyset$ ***and*** *not converge* **do**

  $L' \leftarrow$ Retrieve k annotated instances $(\mathbf{x}, y)$ from U (Sec. 4.2).

  Remove instances L' from U and add them to L_U.

  // Update prediction module:

  Optimize P_θ using data from both L and L_U (Eq. 8).

  // Update retrieval module:

  Optimize Q_ϕ using data from both L and L_U (Eq. 9).

end

Table 1: Statistics for SemEval and TACRED dataset. We present the number of relation mentions in the train/dev/test sets. In particular, % Neg. implies the percentage of "negative" relation mentions (i.e., relation mentions that are labeled as "no_relation").

Dataset	# Train	# Dev	# Test	# Relations	% Neg.
SemEval [15]	7,199	800	1,864	19	17.40
TACRED [39]	75,049	25,763	18,659	42	78.68

4.4 The Joint Learning Algorithm

We summarize the training algorithm for DualRE in Algo. 1. The algorithm is basically an approximation of EM algorithm as described in Sec. 4.1. The whole process works in an iterative manner. In each iteration, we retrieve a new set of annotated instances using both prediction and retrieval modules. According to Sec. 4.2, this is an empirical practice to sample from distribution $p_\theta(\mathbf{x}, y) + q_\phi(\mathbf{x}, y)$. The retrieved instances, along with previous training data can be used for updating the prediction and retrieval module, which correspond to the E step and M step in the EM algorithm. The iterative process stops when our model converges or the unlabeled data is exhausted.

5 EXPERIMENTS

This section first introduces datasets and experimental settings, and then presents performance comparison results with baseline methods as well as case studies of various model components in order to validate different design choices in DualRE.

5.1 Dataset

We choose two public relation extraction datasets with different characteristics to validate the effectiveness of our proposed method. (1) **SemEval** [15]. SemEval 2010 Task 8 provides a standard benchmark dataset which is widely adopted for evaluating relation extraction systems. It has around 10,000 relation mentions in total (where entity mentions are given), with 19 relations (including "no_relation"). (2) **TACRED** [39]. The TAC Relation Extraction Dataset is a large-scale crowd-sourced relation extraction dataset

Table 2: Performance comparison on SemEval [15] with various amounts of labeled data and 50% unlabeled data. We report the mean and standard deviation of the evaluation metrics by conducting 5 runs of training and testing using different random seeds. DualRE outperforms all the baseline methods.

Methods / % Labeled Data	5%			10%			30%		
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
LSTM [17]	25.31 ± 2.16	20.72 ± 3.91	22.71 ± 3.31	36.91 ± 6.10	29.85 ± 7.25	32.92 ± 6.71	64.83 ± 0.69	63.03 ± 0.67	63.91 ± 0.66
PCNN [37]	42.95 ± 4.69	40.79 ± 4.59	41.84 ± 4.63	53.78 ± 1.51	49.11 ± 2.22	51.32 ± 1.74	64.54 ± 0.58	62.98 ± 0.48	63.75 ± 0.33
PRNN [39]	56.16 ± 1.32	54.87 ± 1.49	55.49 ± 0.90	61.70 ± 1.16	63.61 ± 2.07	62.63 ± 1.42	69.66 ± 2.19	68.76 ± 2.60	69.14 ± 1.02
Mean-Teacher (PRNN) [33]	53.71 ± 4.43	49.54 ± 3.29	51.51 ± 3.58	62.43 ± 1.28	60.34 ± 0.62	61.36 ± 0.75	68.65 ± 0.64	69.84 ± 0.65	69.24 ± 0.56
Self-Training (PRNN) [29]	56.47 ± 1.11	56.14 ± 1.33	56.30 ± 0.96	64.27 ± 2.37	63.48 ± 2.02	63.79 ± 0.28	68.95 ± 0.68	72.63 ± 0.82	70.74 ± 0.58
RE-Ensemble (PRNN)	58.77 ± 0.58	58.50 ± 0.97	58.63 ± 0.62	65.10 ± 0.84	64.57 ± 0.54	64.83 ± 0.61	70.26 ± 0.92	73.20 ± 1.22	71.69 ± 0.47
DualRE-Pairwise (PRNN)	59.76 ± 0.47	63.36 ± 0.77	**61.51 ± 0.56**	64.39 ± 0.75	67.70 ± 0.80	66.00 ± 0.48	70.05 ± 0.53	74.83 ± 0.88	72.36 ± 0.60
DualRE-Pointwise (PRNN)	58.73 ± 1.50	62.23 ± 1.93	60.43 ± 1.67	64.50 ± 1.14	67.67 ± 1.66	**66.03 ± 1.00**	70.03 ± 0.74	74.87 ± 0.75	**72.36 ± 0.35**
RE-Gold (PRNN w. gold labels)*	72.57 ± 1.47	74.65 ± 1.98	73.56 ± 0.31	71.40 ± 1.42	76.72 ± 0.64	73.95 ± 0.50	72.98 ± 0.96	78.86 ± 0.76	75.80 ± 0.24

following the TAC KBP relation schema[5]. The corpora are collected from all the prior TAC KBP shared tasks. It has more than 100,000 relation mentions with relations categorized into 42 classes (including "no_relation").

Table 1 shows the overall statistics for these two datasets, which are significantly different in terms of dataset size, number of relations, and percentage of negative instances. As one can observe from the table, TACRED is far more complicated than SemEval, since it has more types of relations and more skewed distribution between positive and negative instances. We conduct experiments on both datasets so as to validate the effectiveness of our DualRE model learned with datasets of different characteristics.

5.2 Compared Methods

As the proposed DualRE framework is general and can be integrated with various supervised "base" model of relation extraction, we first compare several widely used supervised models, then adopt the best performing one as the base model and compare DualRE with other representative semi-supervised methods. All models mentioned below are re-implemented and integrated into a unified interface for efficient experiment conducting.

For base models proposed for supervised relation extraction, we select the following models and train them only on the labeled dataset (i.e., set L): (1) **LSTM** [17]: The Long Short-Term Memory (LSTM) network is a variant of basic Recurrent Neural Network (RNN). We can simply adopt LSTM without attention mechanism to encode the word sequence and classify it into predefined relation categories based on the sentence representation. (2) **PCNN** [37]: Different from the encoding part of LSTM RE model, Piecewise Convolutional Neural Network (PCNN) uses CNN with piecewise max pooling to encode the word sequence. Position embeddings are used to provide the model with the positions of subject and object entities in the sentence. (3) **PRNN** [39]: Integrating position-aware attention mechanism, the Position-aware Recurrent Neural Network (PRNN) is proposed specifically for supervised relation extraction and has achieved state-of-the-art performance on TACRED dataset. Details of PRNN model are described in Sec. 3.1.

As PRNN has achieved state-of-the-art performance in prior work, we construct our DualRE model based on PRNN, and compare with other semi-supervised classification methods which also adopt PRNN as the base model for fair evaluation. For semi-supervised baselines, we select several representative methods from different categories of semi-supervised learning approaches (See Sec. 6 for details), shown as follows. (4) **Self-Training** [29]: Self-Training is a semi-supervised method that uses a single model's predictions on unlabeled data to iteratively re-train and improve the model itself. In each iteration, the model is first used to label the unlabeled instances, then the most confident pseudo-labeled instances are added to the labeled set, which are further used to re-train the model in the next iteration. The iterative training procedure stops when the unlabeled data is exhausted. (5) **Mean-Teacher** [33]: Mean-Teacher belongs to the category of self-ensembling methods. It is first proposed for semi-supervised image classification task. The main idea is to encourage different variants of the model to make consistent predictions over similar inputs. A perturbation-based loss, and the original training loss, are jointly optimized to utilize the information provided by unlabeled data. In our implementation, the dropout strategy [13, 30] is applied to generate the perturbed input.

In addition to the existing semi-supervised methods introduced above, we propose a variant of DualRE model as another baseline model, to further validate the effectiveness of adopting a dual retrieval model in contrast to ensembling two prediction models with identical configuration, i.e., both modules use PRNN as the same base model but with different model parameters. (6) **RE-Ensemble**: By replacing the retrieval module in the proposed DualRE framework with the same prediction module (with a new set of model parameters), we design the RE-Ensemble model that selects the promoted instances based on the agreement of two prediction modules in each iteration. The structure and size of these two models are exactly the same, but they are independently initialized at random to make sure they are not identical.

For our proposed DualRE method, we focus on studying the performance of two model variants based on different learning-to-rank objectives in the retrieval module. The models are trained on both the labeled data and the unlabeled data, as detailed below: (1) **DualRE-Pointwise**: A DualRE model that uses pointwise learning-to-rank loss [7] in the retrieval module. (2) **DualRE-Pairwise**: A DualRE model that uses pairwise learning-to-rank loss [5] in the retrieval module.

Finally, we present another model to demonstrate the upper bound of the performance for all the semi-supervised methods. This model is learned using both the labeled dataset and the "unlabeled"

[5]https://tac.nist.gov/2017/KBP

dataset. **RE-Gold**: It trains the PRNN base model using both the labeled relation mentions and the gold labels of relation mentions from the unlabeled set. This represents the setting that labels over all unlabeled instances can be perfectly predicted, which gives an upper bound performance.

5.3 Experimental Settings

Data Preparation. To study the performance of semi-supervised relation extraction models with various labeled and unlabeled dataset sizes, we sample from the original training set and construct various sizes of labeled and unlabeled set. More specifically, for SemEval dataset, we sample 5%, 10% and 30% training data as labeled sets, the ratios amounts to 3%, 10% and 15% in TACRED. In both datasets, we sample 50% training data as unlabeled set, whose ground-truth labels are unavailable to models. We perform stratified sampling to ensure the label distribution remains unchanged. Duplication in labeled and unlabeled set is avoided as well. Note that since SemEval does not have official development set, we sample 10% of the training set as the development set, before the above data preparation process.

Data Processing. Since the SemEval dataset does not contain named entity (NE) tags for each relation mention, which is essential for our PRNN base model, we apply the Stanford CoreNLP toolkit [21] to extract the needed information.

Hyperparameter Tuning. For base models (LSTM, PCNN, PRNN), we follow the default hyperparameter settings used in previous works [37, 39]. For fair comparison, both DualRE and other baseline methods are built on top of the same base models. For Mean-Teacher, we perform grid search for weight of consistency loss (select from $\{0.1, 1, 10\}$) and number of labeled instances in a mini-batch with size 50 (select from $\{1, 3, 5\}$). For DualRE, we perform greedy search for retrieval module dropout (select from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$), score weight α, β (select from $\{0.5, 1, 2\}$) and label distribution (select from top-k to top-7k distribution and true distribution). The best hyperparameters are chosen according to model performance on the development set.

For semi-supervised models that promote instances in an iterative manner (Self-Training, RE-Ensemble and DualRE), we fix the number of retrieved instances per iteration k for fair comparison. Since DualRE model takes the joint selection of prediction and retrieval modules, retrieving a fixed amount of instances k is not guaranteed. To solve this problem, we starts with a retrieval upper bound $k' = k$ for both modules to select instances. We iteratively increase the upper bound ($k' = 1.25k'$) until we can sample k unique instances from the joint set. In our experiments, k is set to 10% of the number of original unlabeled instances, which means that the unlabeled data is exhausted after 10 iterations.

Evaluation Metric. Following previous work in relation extraction [19, 39], we consider F_1 score as the main metric while precision and recall serve as auxiliary metrics. Note that the correct prediction for no_relation is ignored during the calculation.

5.4 Experiments and Performance Study

Table 2 and 3 show the model performance on both SemEval and TACRED dataset when 50% unlabeled data and various labeled data are available. We conduct several runs of training and testing using different random seeds and report the mean and standard deviation of the results.

Comparing DualRE with Different Methods and Its Variants. Comparing the performance of different models in Table 2 and 3, the first conclusion we draw is that PRNN outperforms other base models in almost all the data points, which demonstrates the effectiveness of the position-aware encoder with the attention mechanism. This justifies the reason for choosing PRNN as the base model for all semi-supervised models.

Secondly, moving on to semi-supervised baseline models, we observe a consistent performance gain when comparing RE-Ensemble with Self-Training. This is a strong evidence that ensemble of single models can boost the performance. However, Mean-Teacher as another self-ensembling method, does not have consistent gain over other models: It shows a clear win in TACRED dataset with 3% labeled data, but falls far behind in all other data points. The reason behind such phenomenon is still under investigation, but Mean-Teacher is clearly not as stable and strong as other semi-supervised baseline models.

Thirdly, our DualRE-Pointwise model outperforms all the baseline models except on TACRED with 3% labeled data. Its overall performance shows the effectiveness of DualRE framework. From the results, we attribute the performance gain to two design choices: (1) Ensemble method. Both DualRE-Pointwise and RE-Ensemble apply such strategy. By adding another prediction module, RE-Ensemble can already outperform its counterpart (Self-Training). (2) Introduction of dual task. Instead of replicating another prediction module as in RE-Ensemble, DualRE-Pointwise introduces the dual retrieval module. Two modules are jointly trained and complement each other. Thus we observe a consistent performance gain of DualRE-Pointwise over RE-Ensemble.

Lastly, speaking of the variant of DualRE, DualRE-Pairwise has comparable or even better performance as DualRE-Pointwise on the SemEval dataset but not on TACRED. Since DualRE-Pointwise generally performs better than DualRE-Pairwise, we use "DualRE" to refer to DualRE-Pointwise model in the following sections.

Performance on Different Datasets. One can clearly observe that the advantage of DualRE-Pairwise model over other baseline models becomes smaller on the TACRED dataset, comparing to SemEval. Considering the complexity and imbalance distribution in TACRED dataset over SemEval, it is more difficult to learn a strong retrieval module, especially for those rare relations. Thus DualRE models are not significantly better than other models.

Performance on Different Amounts of Labeled Data. From Table 2 and 3, when we fix the amount of unlabeled data to 50% of the original training set, we can observe that increasing the amount of labeled data effectively improves the model performance. Therefore, adding more labeled data (if accessible) is an efficient approach to boost the performance.

Performance on Different Amounts of Unlabeled Data. In specific domains where human-annotated training instances are expensive to obtain, we aim to leverage a large amount of unlabeled data to boost the performance of our model given limited amount of labeled data. Therefore, for semi-supervised learning, we may wonder whether more unlabeled data will always help. To investigate such a problem, we fix the amount of labeled data and compare

Table 3: Performance comparison on TACRED [39] with various amounts of labeled data and 50% unlabeled data. We report the mean and standard deviation of the evaluation metrics by conducting 3 runs of training and testing using different random seeds. DualRE outperforms all the baseline methods except Mean-Teacher at one data point.

Methods / % Labeled Data	3%			10%			15%		
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
LSTM [17]	40.76 ± 6.62	23.05 ± 4.48	28.62 ± 3.01	50.56 ± 0.98	43.09 ± 1.26	46.51 ± 0.90	55.11 ± 0.56	44.41 ± 0.58	49.18 ± 0.52
PCNN [37]	58.16 ± 7.74	37.49 ± 6.16	44.64 ± 1.59	64.32 ± 7.78	42.06 ± 4.94	50.16 ± 1.15	67.10 ± 0.44	42.88 ± 0.42	52.32 ± 0.30
PRNN [39]	48.93 ± 5.72	33.05 ± 2.19	39.16 ± 0.90	53.44 ± 2.82	51.77 ± 1.88	52.49 ± 0.64	58.88 ± 2.32	51.30 ± 2.04	54.76 ± 0.93
Mean-Teacher (PRNN) [33]	53.08 ± 3.55	41.81 ± 0.61	**46.74 ± 1.70**	58.53 ± 2.56	50.08 ± 1.14	53.94 ± 0.91	57.90 ± 1.09	52.64 ± 0.97	55.13 ± 0.05
Self-Training (PRNN) [29]	49.89 ± 1.05	39.23 ± 2.26	43.86 ± 1.26	56.54 ± 0.72	53.00 ± 0.49	54.71 ± 0.09	60.09 ± 0.43	54.77 ± 0.55	57.31 ± 0.47
RE-Ensemble (PRNN)	56.48 ± 0.95	36.90 ± 0.91	44.62 ± 0.39	61.26 ± 0.58	52.51 ± 0.56	55.54 ± 0.29	60.76 ± 0.78	55.00 ± 1.04	57.72 ± 0.38
DualRE-Pairwise (PRNN)	58.97 ± 0.96	34.55 ± 1.18	43.55 ± 0.67	63.10 ± 0.94	48.91 ± 0.93	55.09 ± 0.25	60.99 ± 1.39	54.04 ± 0.46	57.30 ± 0.81
DualRE-Pointwise (PRNN)	52.76 ± 2.58	38.99 ± 2.08	44.73 ± 0.66	61.61 ± 1.30	52.30 ± 0.89	**56.56 ± 0.42**	60.66 ± 1.57	56.65 ± 0.37	**58.58 ± 0.69**
RE-Gold (PRNN w. gold labels)*	64.38 ± 0.46	60.35 ± 0.81	62.30 ± 0.29	65.88 ± 0.66	61.65 ± 0.42	63.70 ± 0.53	66.95 ± 2.78	59.97 ± 3.12	63.13 ± 0.56

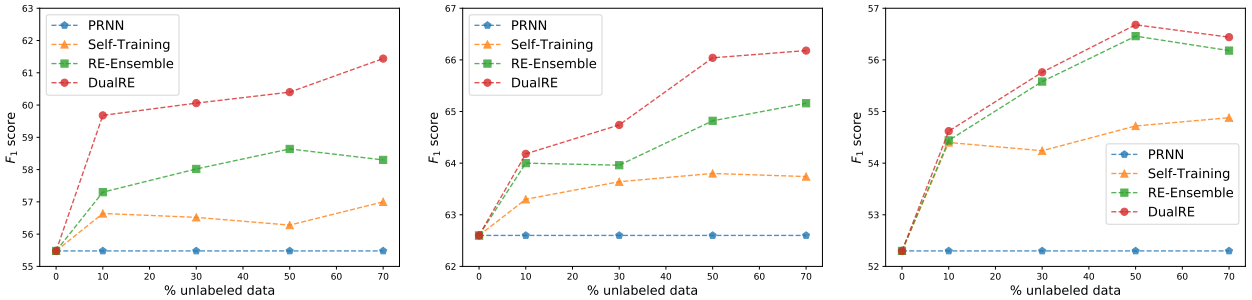


Figure 5: Performance w.r.t. the amount of unlabeled data. Left: 5% labeled data on SemEval; Middle: 10% labeled data on SemEval; Right: 10% labeled data on TACRED. In most cases, increasing unlabeled data results in better performance, but too much unlabeled data may hurt the performance.

Table 4: Comparison of Different Weighing Methods for Instance Promotion. We test with 10% labeled data and 50% unlabeled data on the SemEval dataset. Weighted promotion leads to significant improvement as compared with equal promotion.

Instance Weight	F_1 (Dev)	F_1 (Test)
α=0, β=0 (Equal Promotion)	62.40 ± 0.52	64.42 ± 0.82
α=1, β=1	63.84 ± 0.23	65.98 ± 0.46
α=1, β=2	64.22 ± 0.26	66.02 ± 0.15
α=0.5, β=2	**64.28 ± 0.46**	**66.04 ± 1.00**

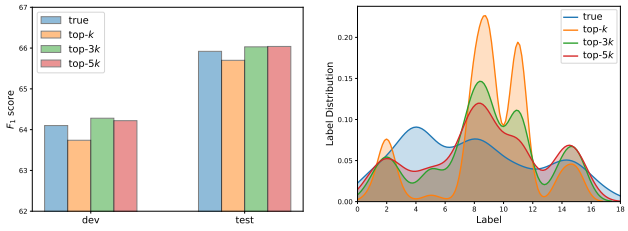


Figure 6: Case Study on Retrieval Module in DualRE. Left: Comparison of using different label distributions as the retrieval query; Right: A visualization of different label distributions in the first iteration. Both are evaluated with 10% labeled data and 50% unlabeled data on the SemEval dataset.

the performance under different amounts of unlabeled data. More specifically, we investigate the model performance in SemEval and TACRED dataset, with 5% or 10% labeled data presented. In our experiment, 10%, 30%, 50% and 70% unlabeled data are available.

We summarize the results of DualRE and other methods in Fig. 5. One can observe that most models benefit from larger amount of unlabeled data in the range of 10% to 50%. However, increasing unlabeled data to 70% doesn't result in further improvement for all models. Half of them perform even worse with more unlabeled data. Therefore, it is not necessarily beneficial for semi-supervised models to take in as much unlabeled data as possible.

5.5 Case Study

Effectiveness of Weighted Promotion. We experiment with two different ways of assigning weights to pseudo-labeled instances while training DualRE model. Specifically, different combinations of α and β are explored, according to Section 4.3 . Assigning α=0 and β=0, for instance, is to promote all instances equally in training. Results in Table 4 indicate that by using weighted promotion, we can get better results than equal promotion.

Choice of Reference Relation Distribution. In DualRE, when selecting instances to promote, the retrieval module takes in a reference relation distribution $q(y)$ as described in Sec. 4.2. Practically, we assign $q(y)$ with label distribution from available dataset. Two options are explored: (1) **true**: Use the label distribution from the labeled set L. (2) **top-nk**: Use the label distribution from the most confident $n \times k$ instances given by the prediction module.

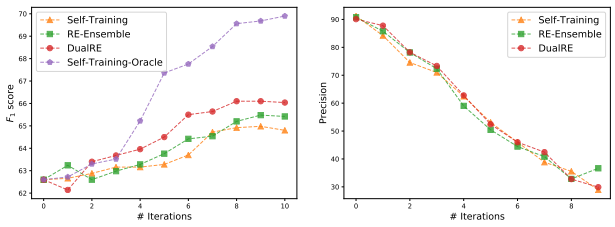


Figure 7: Analysis on Quality of Retrieved Instances. Left: Convergence curve of test F_1 score for different methods; Right: Precision of the relation mentions selected in each iteration. Both are evaluated with 10% labeled data and 50% unlabeled data on the SemEval dataset.

According to Fig. 6, it can be concluded that the best performance is achieved at top-3k, whose distribution lies between true and top-k. Intuitively, the retrieved instances work better if they follow the true distribution. Due to the unbalanced distribution, however, the long-tail relations usually have fewer instances and are harder to learn. The true relation distribution forces the model to attend to these relations and selection quality goes down. On the other hand, top-k distribution is too biased towards frequent relations. Retrieving more of these instances does not add much information for models to learn. Therefore, Top-3k distribution can be regarded as a combination of above two strategies and works better.

Quality of Retrieved Instances. To investigate the underlying reason for performance gain in DualRE, we hypothesize that the quality (i.e. precision) of retrieved instances is higher in DualRE. In the study, we utilize the ground-truth labels of the "unlabeled data" to evaluate the precision of retrieved instances.

In our study, we evaluate the self training methods (Self-Training, RE-Ensemble and DualRE) using test set F_1 score in each iteration together with the precision of its promoted pseudo-labeled instances. Additionally, we construct an oracle model "Self-Training-Oracle" to identify the upper bound of self-training methods. Similar to Self-Training, we rank the instances by their probability of predicted label and select new instances in each iteration. But here we assume that the model can perfectly select those correctly pseudo-labeled instances and add them to the training set in each iteration.

Experimental results are shown in Fig. 7. Comparing results on DualRE and other semi-supervised models, the precision for retrieved instances in each iteration are generally higher in DualRE, which indicates that DualRE can select instances with better quality. This directly contributes to the performance gain in F_1 score in each iteration. To measure the upper bound of semi-supervised models that fall in the self-training category, the performance of Self-Training-Oracle indicates that the potential for such models are high enough to be comparable with RE-Gold (refer to Table 2). But this upper bound is extremely hard to approach because the model does not have additional supervision to guide the instance selection process.

6 RELATED WORK

Relation extraction. Our work is related to sentence-level relation extraction, which aims at predicting the relation label between two entities in a sentence. Traditional relation extraction methods [1, 26] seek to find discriminative textual pattern s with some labeled sentences, and further leverage these patterns for relation extraction. Nevertheless, the learned patterns may have difficulties matching with unseen sentences, as patterns are composed of discrete words. Recent studies leverage deep neural networks such as Long Short-Term Memory Networks (LSTM) [17] and Convolutional Neural Networks (CNN) [37] for relation extraction, achieving much better performance. However, all these methods are fully-supervised, which require a large amount of labeled data to train effective models. By contrast, our proposed approach is semi-supervised, which requires much fewer labeled sentences during training.

Semi-supervised learning. In the literature of semi-supervised learning, there are some methods which can be applied to our problem. One major category is the self-ensembling method [12], which assumes that the predicted labels of the unlabeled instances should remain unchanged under small perturbations on the instances or the model parameters. In this way, they guarantee the local consistency of the prediction and hence improve the performance. However, when applied to our problem, such a training framework relies solely on local consistency as supervision signal, which is weak compared to labeled data, therefore the improvement is marginal. Another category is the self-training method [29], which iteratively promotes instances from unlabeled data. Self-training methods impose a stronger supervision signal and generally achieve better performance. However, it suffers from semantic drift [9], which accumulates systematic errors from prediction model in each iteration. Co-training [2] is an attempt to eliminate this problem. It promotes unlabeled data using two models that are trained on different views of data, which essentially relies on the assumptions that two different sets of features are independent and sufficient for prediction. This assumption is not easy to follow in the relation extraction task, in which relation mention is the only source of input. Our DualRE model, instead, introduces a dual retrieval module to correct the bias from prediction module. Therefore we are able to collect extra training data with higher quality to avoid semantic drifting.

Dual learning. Our work is also related to dual learning [14, 32, 35], which can be generalized to the concept of jointly learning the primal and dual models to boost the performance of each other. For example, English-French translation and French-English translation [14], image classification and image generation [35], question answering and question generation [32] can be treated together to leverage the duality of two tasks. Different from their supervised settings, we apply the concept of dual model in semi-supervised settings and propose an algorithm that effectively utilize the unlabeled data. We are also the first to perform dual model learning in the relation extraction task.

7 CONCLUSION

In this paper, we studied the problem of relation extraction in the semi-supervised setting, where a few labeled sentences and an auxiliary unlabeled corpus are provided. We proposed a principled framework consisting of a prediction module and a retrieval module, which focus on the primal prediction task and the dual retrieval task respectively. During training, we encouraged them to mutually

enhance each other through annotating unlabeled sentences as additional training data. A joint optimization method was proposed for optimizing our framework, which alternates between updating the prediction module and updating the retrieval module. Experimental results on two public datasets proved the effectiveness of our proposed framework. Interesting future work directions include switching the retrieval module to text generative models, extending our framework to deal with various text classification tasks, and studying its connection to the variational auto-encoder method.

8 ACKNOWLEDGEMENT

This work has been supported in part by National Science Foundation SMA 18-29268, Amazon Faculty Award, and JP Morgan AI Research Award. We would like to thank all the collaborators in INK research lab for their constructive feedback on the work.

REFERENCES

[1] Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *ACM DL'00*. 85–94.
[2] Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *COLT'98*. 92–100.
[3] Sergey Brin. 1998. Extracting patterns and relations from the world wide web. In *International Workshop on The World Wide Web and Databases*. Springer, 172–183.
[4] Razvan C Bunescu and Raymond J Mooney. 2005. A shortest path dependency kernel for relation extraction. In *HLT-EMNLP'05*. 724–731.
[5] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. ACM, 89–96.
[6] Yunbo Cao, Jun Xu, Tie-Yan Liu, Hang Li, Yalou Huang, and Hsiao-Wuen Hon. 2006. Adapting ranking SVM to document retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 186–193.
[7] David Cossock and Tong Zhang. 2006. Subset ranking using regression. In *International Conference on Computational Learning Theory*. Springer, 605–619.
[8] Koby Crammer and Yoram Singer. 2002. Pranking with ranking. In *Advances in neural information processing systems*. 641–647.
[9] James R Curran, Tara Murphy, and Bernhard Scholz. 2007. Minimising semantic drift with mutual exclusion bootstrapping. In *PACLING'07*. 172–180.
[10] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)* (1977), 1–38.
[11] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*.
[12] Geoffrey French, Michal Mackiewicz, and Mark Fisher. 2017. Self-ensembling for visual domain adaptation. *arXiv preprint arXiv:1706.05208* (2017).
[13] Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*. 1019–1027.
[14] Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tieyan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*. 820–828.
[15] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2009. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*. Association for Computational Linguistics, 94–99.
[16] Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. 1995. The" wake-sleep" algorithm for unsupervised neural networks. *Science* 268, 5214 (1995), 1158–1161.
[17] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[18] Ping Li, Qiang Wu, and Christopher J Burges. 2008. Mcrank: Learning to rank using multiple classification and gradient boosting. In *Advances in neural information processing systems*. 897–904.
[19] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural Relation Extraction with Selective Attention over Instances.. In *ACL'16*. 2124–2133.
[20] Tie-Yan Liu et al. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 3, 3 (2009), 225–331.
[21] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*. 55–60.
[22] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data.. In *ACL-IJCNLP'09*. 1003–1011.
[23] Takeru Miyato, Shin-ichi Maeda, Shin Ishii, and Masanori Koyama. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence* (2018).
[24] Raymond J Mooney and Razvan C Bunescu. 2006. Subsequence kernels for relation extraction. In *NIPS'06*. MIT Press, 171–178.
[25] Gerhard Paass. 1993. Assessing and improving neural network predictions by the bootstrap algorithm. In *Advances in Neural Information Processing Systems*. 196–203.
[26] Meng Qu, Xiang Ren, and Jiawei Han. 2017. Automatic Synonym Discovery with Knowledge Bases. In *KDD'17*. 997–1005.
[27] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. 2015. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*. 3546–3554.
[28] Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. CoType: Joint extraction of typed entities and relations with knowledge bases.. In *WWW'17*. 1015–1024.
[29] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. 2005. Semi-Supervised Self-Training of Object Detection Models.. In *WACV/MOTION*. 29–36.
[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.
[31] Ang Sun and Ralph Grishman. 2012. Active learning for relation type extension with local and global data views. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 1105–1112.
[32] Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027* (2017).
[33] Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*. 1195–1204.
[34] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing Text for Joint Embedding of Text and Knowledge Bases.. In *EMNLP'15*. 1499–1509.
[35] Yingce Xia, Tao Qin, Wei Chen, Jiang Bian, Nenghai Yu, and Tie-Yan Liu. 2017. Dual supervised learning. *arXiv preprint arXiv:1707.00415* (2017).
[36] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of machine learning research* 3, Feb (2003), 1083–1106.
[37] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1753–1762.
[38] Wenyuan Zeng, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2017. Incorporating relation paths in neural relation extraction. In *EMNLP'17*. 1769–1778.
[39] Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware Attention and Supervised Data Improve Slot Filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 35–45.

Distributed Algorithms for Fully Personalized PageRank on Large Graphs

Wenqing Lin
Interactive Entertainment Group, Tencent Inc.
edwlin@tencent.com

ABSTRACT

Personalized PageRank (PPR) has enormous applications, such as link prediction and recommendation systems for social networks, which often require the fully PPR to be known. Besides, most of real-life graphs are edge-weighted, e.g., the interaction between users on the Facebook network. However, it is computationally difficult to compute the fully PPR, especially on large graphs, not to mention that most existing approaches do not consider the weights of edges. In particular, the existing approach cannot handle graphs with billion edges on a moderate-size cluster. To address this problem, this paper presents a novel study on the computation of fully edge-weighted PPR on large graphs using the distributed computing framework. Specifically, we employ the Monte Carlo approximation that performs a large number of random walks from each node of the graph, and exploits the parallel pipeline framework to reduce the overall running time of the fully PPR. Based on that, we develop several optimization techniques which (i) alleviate the issue of large nodes that could explode the memory space, (ii) pre-compute short walks for small nodes that largely speedup the computation of random walks, and (iii) optimize the amount of random walks to compute in each pipeline that significantly reduces the overhead. With extensive experiments on a variety of real-life graph datasets, we demonstrate that our solution is several orders of magnitude faster than the state-of-the-arts, and meanwhile, largely outperforms the baseline algorithms in terms of accuracy.

CCS CONCEPTS

• **Information systems** → **Social networks**; *Personalization*; • **Computing methodologies** → *MapReduce algorithms.*

KEYWORDS

Distributed Algorithm; Graph; Personalized PageRank

ACM Reference Format:
Wenqing Lin. 2019. Distributed Algorithms for Fully Personalized PageRank on Large Graphs. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313555

1 INTRODUCTION

Given a graph $G = (V, E, w)$, and two nodes $u, v \in V$, the personalized PageRank (PPR) of v with respect to u, denoted by $\pi(u, v)$, is defined as the probability of a random walk starting from u and ending at v. Compared to PageRank (PR), which is a global metric of nodes, PPR depicts the asymmetric perspective of each node to all the other nodes, i.e., $\pi(u, v)$ is not necessarily equal to $\pi(v, u)$. There are a plethora of applications that utilize PPR to measure the proximity of two nodes in the graph, such as recommendation system [5, 19, 21] and natural language processing [1]. In these applications, the fully PPR, i.e., the PPRs of all pairs of nodes, are often required. In other words, for all node v in the graph, the PPR of all the other nodes u with respect to v should be known.

However, most of previous work [9, 11, 16] focus on the computation of single-pair or single-source PPR, rendering them unsuitable for solving the problem of fully PPR. For example, FORA [27] is the recent algorithm for single-source PPR, and needs 103 seconds to answer a single-source query on a Twitter graph of 41.7 millions of nodes, which would require about 137 years to compute the single-source PPRs for all nodes. In addition, it is highly difficult to compute the fully PPR, especially on large graphs. In particular, the probability of a random walk is usually computed by the Power Iteration method [8] or approximated by the Monte Carlo method [7], which might require a large number of iterations of computation to obtain a converged results. Besides, most of existing methods are tailored for the setting of a single machine [7], which renders them unable to handle graphs that cannot fit in memory, not to mention that the space complexity of the fully PPR could be $O(n^2)$ where n is the number of nodes in the graph. What's more, most real-world graphs are weighted, which can be directly generated from the interactions between nodes. For instance, the weight of the edge from a user u to another user v in the Facebook network could be the number of comments from u to the posts of v. Nevertheless, most existing methods do not consider the weights on graph, making their algorithms unsuitable.

To address these issues, we devise a distributed share-nothing solution that employs the Monte Carlo approximation to perform a large number of random walks from all nodes of the graph in parallel. As such, for each node $u \in V$, we maintain only the PPR $\pi(u, v)$ where $v \in V$ is the node incurred in at least one random walk. Besides, we exploit the parallel pipeline framework [13] where (i) a subset of random walks is computed simultaneously in a pipeline and (ii) the new pipeline can remedy the deficiency of the old pipeline, as it is rare to generate long random walks due to the termination probability. Hence, it reduces the number of iterations of computation that significantly accelerates the process.

In order to optimize the above solution, we first address the issue of large nodes that would lead to the skewness of data. Specifically, we recursively divide each large node into small ones, which are then organized as the *alias tree* in a hierarchical manner.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313555

In addition, to address the issue of under-utilization of small nodes, we pre-compute for each small node a set of *short* walks to facilitate the acceleration of random walk generation on small nodes. Finally, since it would significantly degrade the performance of a pipeline which deals with very few or too many random walks at a time, we develop a method that optimizes the number of random walks in each pipeline so as to reduce the overhead of pipelining.

In summary, our contributions are the followings.

- We devise the distributed algorithms based on the parallel pipeline framework that fully utilizes the parallelism to efficiently generate a large number of random walks simultaneously for all nodes of the graph.
- We design the hierarchical sampling algorithm that avoids the skewness on large nodes and also allows the algorithm to select a node from the set of neighbors following the distribution of weights on edges in constant time.
- We develop an efficient algorithm to deal with small nodes by pre-computing short walks so that some random walks can be stopped earlier, which saves lots of costs.
- We devise an effective method to optimize the number of random walks in each pipeline which largely increases the throughput of the algorithm that greatly improves its performance on large graphs.
- In the extensive experiments, we demonstrate that our solution outperforms the state-of-the-art solutions by up to 21% on accuracy and by up to 2 orders of magnitude on the running time.

Paper organization. Section 2 illustrates the definitions and notations used in the paper. An overview of our solution is depicted in Section 3. After that, the details of the algorithm are explained in Section 4. Section 5 demonstrates the superior performance of our algorithm compared with several baseline methods over many graphs. Section 6 discusses the related work of this paper. Finally, we conclude the paper in Section 7.

2 PRELIMINARIES

In this section, we present the concepts and frequently used notations in this paper, as listed in Table 1.

2.1 Personalized PageRank

Consider an edge-weighted graph $G = (V, E, w)$, where V is the set of nodes, E is the set of edges, and w is the weighting function that maps each edge $e \in E$ to a positive number $w(e)$, i.e., $w(e) > 0$. For each node $s \in V$, we say that t is an *out-neighbor* (resp. *in-neighbor*) of s, if there exists an edge $\langle s, t\rangle$ (resp. $\langle t, s\rangle$) such that $\langle s, t\rangle \in E$ (resp. $\langle t, s\rangle \in E$). Denote $N_{out}(s)$ (resp. $N_{in}(s)$) as the set of out-neighbors (resp. in-neighbors) of s.

Given an out-neighbor $t \in N_{out}(s)$ of a node $s \in V$, we refer to the *routing probability* of t with respect to s as $r(s,t) = w(\langle s,t\rangle)/\sum_{v \in N_{out}(s)} w(\langle s,v\rangle)$. As such, we have $r(s,t) \in (0,1]$.

Given a graph $G = (V, E, w)$, a *walk* or *path* P of G is a sequence of nodes, denoted by $\langle v_1, v_2, \cdots, v_k\rangle$, such that (i) $v_i \in V$ for each $i \in [1,k]$ and (ii) $\langle v_i, v_{i+1}\rangle \in E$ for each $i \in [1, k-1]$. The *length* of P, denoted by $\ell(P)$, is the number of edges in P, i.e., $\ell(P) = k - 1$.

A *random walk* $P_r = \langle v_1, v_2, \cdots, v_k\rangle$ of G is a path that is generated in a random manner following the distribution of routing probability. Specifically, for each node v_i where $i \in [1, k-1]$, we have $v_{i+1} \in N_{out}(v_i)$ and v_{i+1} is chosen with the routing probability $r(v_i, v_{i+1})$. Furthermore, at each step of the random walk P_r, there is a *termination probability* $\alpha \in (0,1)$, which is a user-defined number, that determines whether the path will be terminated or not. In other words, after choosing v_{i+1}, P_r will stop at v_{i+1} with the termination probability α. Hence, the expected length of a random walk is $1/\alpha$. In case that $N_{out}(v_i)$ is empty, following the previous work [27], we restart P_r from its head, i.e., v_1.

Example 2.1. Figure 1 shows a graph G which has 8 nodes and 12 weighted edges. The weights of edges incident to each node are normalized such that the routing probability of each edge equals its weight. Consider node v_0 and $\alpha = 0.5$. We have $N_{out}(v_0) = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ and $N_{in}(v_0) = \{v_5\}$. The path $\langle v_0, v_3, v_4\rangle$ is walk starting from v_0 and ending at v_4, which can be generated by (i) first selecting v_3 from $N_{out}(v_0)$ with the probability 0.15, (ii) and then with the probability of $(1-\alpha) = 0.5$ continuing to select v_4 from $N_{out}(v_3)$ with the probability 1, (iii) finally stopping at v_4 with the termination probability $\alpha = 0.5$. Besides, there exist the other paths that start from v_0 and end at v_4, such as $\langle v_0, v_4\rangle$ and $\langle v_0, v_2, v_3, v_4\rangle$. □

Given a graph $G = (V, E, w)$ and two nodes $u, v \in V$, the *personalized PageRank* (PPR) of v with respect to u, denoted by $\pi(u, v)$, is the probability of a random walk which starts from u and ends at v.

To address this problem, there are roughly two kinds of solutions: One is the matrix based solution that utilizes the adjacency matrix of the graph and the matrix of fully PPR, the other one is based on the Monte Carlo (MC) method that estimates the probability by simulating a large number of random walks on the graph. However, the matrix-based solution incurs a space complexity of $O(n^2)$ where n is the number of nodes in the graph, which is extremely huge, especially for large graph. As such, we adopt the MC based solution in our proposed approach.

Specifically, for each node $s \in V$, the MC method generates a number ω of random walks starting from s to estimate $\pi(s,t)$ for every other node $t \in V$. If there are ω' random walks terminating at t, then we have an unbiased estimate of $\pi(s,t)$ as $\hat{\pi}(s,t) = \omega'/\omega$. It is easy to see that a larger ω leads to a more accurate estimate of $\pi(s,t)$, and also a much more expensive computational cost. To strike a good trade-off, we follow the theorem in the work [7] by setting $\omega = \Omega(\frac{\log(1/p_f)}{\epsilon^2\delta})$ to obtain an ϵ-approximate PPR for any $\pi(s,t) \geq \delta$ with a probability $1 - p_f$, where $s, t \in V$, which is formally defined as follows.

Definition 2.2 (Fully Approximate PPR (FAPPR)). Given an edge-weighted graph $G = (V, E, w)$, a threshold δ, an error bound ϵ, and a failure probability p_f, the fully approximate PPR returns an estimated PPR $\hat{\pi}(s,t)$ for all pairs (s,t) of nodes where $s, t \in V$, such that for all $\pi(s,t) \geq \delta$,

$$|\pi(s,t) - \hat{\pi}(s,t)| \leq \epsilon \cdot \pi(s,t) \tag{1}$$

holds with a probability at least $1 - p_f$. □

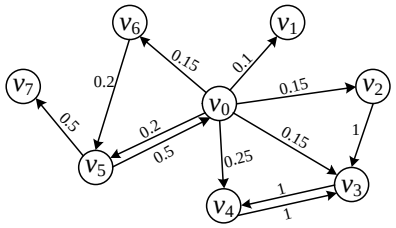


Figure 1: A toy graph with 8 nodes and 12 weighed edges.

Goal. In the present work, we are to devise efficient distributed algorithms for the computation of fully approximate PPR on large graphs.

2.2 The Alias Method

One of the key operations in random walk is to select one node t from s's out-neighbors, where $s \in V$, according to the routing probability. In other words, given a set S of elements, as well as the routing probability $r(e)$ for each $e \in S$, where $\sum_{e \in S} r(e) = 1$, we are to select an element e from S with the probability $r(e)$.

In a naive approach, similar to [4], we can identify an element e from S which has the largest $z^{1/r(e)}$, where $z \in [0, 1]$ is a random number. Consequently, this approach needs to inspect each element in S on the fly, whose time complexity is $O(|S|)$. However, it might lead to the workload imbalance in the distributed computing algorithm on the graph, as the sets of out-neighbors could be of different sizes.

We present *the alias method* [25], which pre-computes a *switch probability* $p(e) \in [0, 1]$ and an *alias* $a(e) \in S$ for each element $e \in S$. As such, an element e can be selected from S with constant time complexity by exploiting $p(e)$ and $a(e)$.

Algorithm 1 illustrates the pre-computing step of the alias method that takes as input the routing probability r and the set S of elements, i.e., edges. Initially, for each edge $e \in S$, its alias $a(e)$ is set as e, and its switch probability $p(e)$ is computed as $|S| \cdot r(e)$ (Line 1). Thus, we have $\sum_{e \in S} p(e) = |S|$ at the beginning of the algorithm. Denote S^l (resp. S^s) as the set of elements e where $p(e) > 1$ (resp. $p(e) < 1$) (Line 2). Then, we choose an element x from S^s and an element y from S^l respectively. After that, we update the alias of x as $a(x) = y$, and decrease the probability of y by $(1 - p(x))$, referred to as $p'(y) = p(y) - (1 - p(x))$. As x is processed, we remove x from S_s. Besides, If $p'(y) \leq 1$, then we remove y from S^l. Furthermore, if $p'(y) < 1$, then we add y into S^s. We iteratively process the elements in S^l, until S^l is empty (Line 3-9). Finally, we have $p(e)$ and $a(e)$ for all $e \in S$ as the results. Since each element in S is inspected only once, the time complexity of this pre-computing step is $O(|S|)$.

Example 2.3. Consider the set $S_1 = \{e_1, e_2, e_3\}$ in Fig. 3. Consider the edges e in S_1. The routing probability of e_1 is computed as $r(e_1) = w(e_1)/\sum_{e \in S_1} w(e) = 0.25$. Similarly, we have $r(e_2) = r(e_3) = 0.375$. At the beginning, the switch probability of e_1 is computed as $p(e_1) = |S_1| \cdot r(e_1)/\sum_{e \in S_1} r(e) = 0.75$. Similarly, we have $p(e_2) = p(e_3) = 1.125$. Besides, we have $a(e_i) = e_i$ where $1 \leq i \leq 3$. As such, we construct $S_1^l = \{e_2, e_3\}$ since $p(e_2) > 1$ and $p(e_3) > 1$, and $S_1^s = \{e_1\}$ since $p(e_1) < 1$. Assume that $e_1 \in S_1^s$ and $e_2 \in S_1^l$ are selected. Then, we update $a(e_1)$ as e_2, remove e_1 from S_1^s, and decrease $p(e_2)$ by $(1 - p(e_1))$, resulting in $p(e_2) = 0.875$. Since $p(e_2) < 1$, we remove e_2 from S_1^l and add e_2 into S_1^s. Then, we have $S_1^s = \{e_2\}$ and $S_1^l = \{e_3\}$. In the same way, we can compute that $a(e_2) = e_3$ and $p(e_3) = 1$, which forms the aliases and switch probabilities of S_1, as shown in Fig. 3. □

Algorithm 1: Alias(S, r)

Input: The set S of elements and the routing probability r
Output: The switch probability $p(e)$ and the alias $a(e)$ for all $e \in S$

1 Let $p(e) = |S| \cdot r(e)$ and $a(e) = e$;
2 Let $S^l = \{e \in S | p(e) > 1\}$ and $S^s = \{e \in S | p(e) < 1\}$;
3 **while** S^l is not empty **do**
4 Select any elements $x \in S^s$ and $y \in S^l$;
5 Let $a(x) = y$ and remove x from S^s;
6 Decrease $p(y)$ by $(1 - p(x))$;
7 **if** $p(y) \leq 1$ **then**
8 Remove y from S^l;
9 If $p(y) < 1$, then add y into S^s;
10 **return** $p(e)$ and $a(e)$ for all $e \in S$.

To select an element e from S with the switch probability $p(e)$ and the alias $a(e)$, we first identify an element e' with the probability $1/|S|$. Then, a probability threshold $p_t \in [0, 1]$ is generated by random. If $p_t \leq p(e')$, then we have $e = e'$; Otherwise, let e be the alias of e', i.e., $e = a(e')$. Therefore, the probability of selecting e from S is $\frac{1}{|S|} \cdot p(e) + \sum_{e' \in S \wedge a(e') = e} \frac{1}{|S|} \cdot (1 - p(e')) = r(e)$. Apparently, the time complexity of selecting by the alias method is $O(1)$.

2.3 Distributed Computing

However, it is impractical to compute FAPPR on a single machine, since (i) the graph could be massive that cannot fit in the memory of a single machine, and (ii) there could be an extremely large amount of computational cost due to lots of graph traversals, especially when the graph is sufficiently large, which also results in a large number of random walks (see Section 2.1).

As such, we utilize the distributed computing to generate the large number of random walks for all nodes of the graph among massive machines in parallel. In particular, we adopt Spark [30], whose fundamental computing unit is MapReduce [3, 23]. Note that, it is easy to extend to the other parallel computing framework, such as Pregel [20]. In a MapReduce job, there are mainly two consequent phases: The first one is the Map phase that takes as input the data and emits key-value pairs by the map function; After that, it is the Reduce phase that aggregates the key-value pairs by the key and then applies the reduce function on the aggregated data, which outputs the new key-value pairs.

To facilitate the processing of graphs on Spark, we store the graph as key-value pairs, such that, for each node $v \in V$, we a key-value pair $\langle v, N_{out}(v) \rangle$ where the key is v and the value is the set of v's out-neighbors, i.e., $N_{out}(v)$. Denote the set of key-value pairs of G as $KV(G) = \{\langle v, N_{out}(v) \rangle | v \in V\}$. Besides, to present a path $P = \langle v_h, \cdots, v_t \rangle$, we use a key-value pair $\langle v_t, v_h \rangle$ consisting

Table 1: Frequently used notations.

Notation	Description
$G = (V, E, w)$	an edge-weighted and directed graph, where V is the set of nodes, E is the set of edges, and w is the weighting function that maps each edge in E to a positive number.
r	the routing probability of an edge (see Section 2.1).
ω	the number of random walks for each node (see Definition 2.2).
γ	the number of random walks in each pipeline (see Definition 2.2).
α	the termination probability of a random walk (see Definition 2.2).
ϵ	the relative accuracy guarantee (see Definition 2.2).
δ	the PPR value threshold (see Definition 2.2).
p	the switch probability of an edge in the alias tree.
a	the alias of an edge in the alias tree.
d	the degree threshold in Algorithm 3 and Algorithm 4.

of the tail v_t of P as the key and the head v_h of P as the value. This is because (i) only the head and tail of a random walk can contribute to the calculation of PPR, and (ii) the extension of a path happens at its tail.

Based on that, given a set $\mathcal{P}$ of paths, we can make a further *move* or *step* for all paths $P \in \mathcal{P}$ by one *round* of computation on Spark that utilizes a MapReduce job: First, the Map phase takes as input both the key-value pairs $\langle v_h, v_t \rangle$ of all paths $P \in \mathcal{P}$ and the key-value pairs $\langle v, N_{out}(v) \rangle \in KV(G)$ of the graph G, and emits the key-value pairs as what they are; Then, in the Reduce phase, we aggregates the key-value pairs by the key, resulting in that each aggregated data has one $\langle v_h, N_{out}(v_h) \rangle$ and several other key-value pairs $\langle v_h, u \rangle$ that represent the paths starting from u and currently visiting v_h. For each $\langle v_h, u \rangle$, we select a node v'_h from $N_{out}(v_h)$ with the routing probability $r(v_h, v'_h)$, and output a new key-value pair $\langle v'_h, u \rangle$, which represents the extended path of $\langle v_h, u \rangle$.

Note that, one can implement the above algorithm by exploiting the *join* operation on Spark. As such, it can be optimized by a careful designed data partitioning scheme [31], which can avoid the cost of transferring between machines the set $KV(G)$ of key-value pairs of G.

3 SOLUTION OVERVIEW

In this section, we present an overview of our distributed algorithm for the computation of the fully approximate edge-weighted PPR (see Definition 2.2). As the major cost of FAPPR resides in the computation of random walks, in the sequel, we will concentrate our discussions on the algorithms for generating random walks.

Given a graph $G = (V, E, w)$, to generate ω random walks for each node $v \in V$ in the distributed computing framework, one straightforward solution could run in ω iterations, each of which computes one random walk for each node. In particular, one random walk P ending with the node v can be extended by one more step in one round of the distributed computing, where we append P with one node randomly chosen from $N(v)$. However, as the expected length of a random walk is $1/\alpha$, this approach results in ω/α number of rounds of computation, which is highly expensive. Alternatively, one might propose to generate all ω random walks for each node $v \in V$ using one iteration with only $1/\alpha$ rounds, where ω number of paths for each node are computed simultaneously. However, this approach would incur significant overhead in the management of memory, as ω could be sufficiently large.

Algorithm 2: Distributed-FAPPR(G, ϵ, p_f, δ, α)

Input: Graph $G = (V, E, w)$, relative accuracy guarantee ϵ, failure probability p_f, threshold δ, termination probability α
Output: The estimated PPR $\hat{\pi}(s, t)$, for all $s, t \in V$

```
Let R_t = ∅ be the collection of terminated random walks;
Let R_a = ∅ be the collection of active random walks;
Let ω = Ω((log(1/p_f))/(ε²δ)) be the number of random walks to be computed for each node v ∈ V;
while S ≠ ∅ or ω > 0 do
    if ω > 0 then
        for each node v ∈ V in parallel do
            Add into R_a min(γ, ω) number of paths consisting of a single node v, i.e., ⟨v, v⟩;
        Update ω as ω − γ;
    for each path P ∈ R_a in parallel do
        Let s and u be the head and tail of P respectively;
        Select a node t ∈ N_out(u);
        Append t to the end of P;
        Generate a random number q ∈ [0, 1];
        if q ≤ α then
            Mark P as terminated;
    Remove from R_a the terminated paths, which are then added into R_t;
Let R_t(s, t) be the set of paths P ∈ R_t where s and t are the head and tail of P respectively;
Compute π̂(s, t) = Σ_{P∈R_t(s,t)} 1/ω in parallel;
return π̂(s, t) for all s, t ∈ V.
```

To remedy the aforementioned issues, we exploit the parallel pipeline framework, as depicted in Figure 2, where (i) each pipeline generates at most $\gamma \in [1, \omega]$ random walks for each node $v \in V$, and (ii) the pipelines begin sequentially, such that one pipeline L starts only when the pipeline L', prior to L, completes one round of computation, i.e., the paths in L' are extended with one step. Therefore, there are $\lceil \omega/\gamma \rceil$ pipelines, and the expected number of rounds in each pipeline is $1/\alpha$. That is, the parallel pipeline framework is expected to employ $(\lceil \omega/\gamma \rceil + 1/\alpha - 1)$ number of rounds of computation, which is much smaller than the one of previous approaches. Besides, as each random walk could be terminated with the probability α, the workload of each pipeline becomes fewer and fewer as the number of iterations increases. Therefore, we can avoid the under-utilization of computing for long pipelines, and make sure that the pipeline L would receive few affections from the pipelines that are far from L.

Algorithm 2 depicts an overview of the proposed algorithm that takes as input the graph $G = (V, E, w)$, the relative accuracy guarantee ϵ, failure probability p_f, threshold δ, and termination

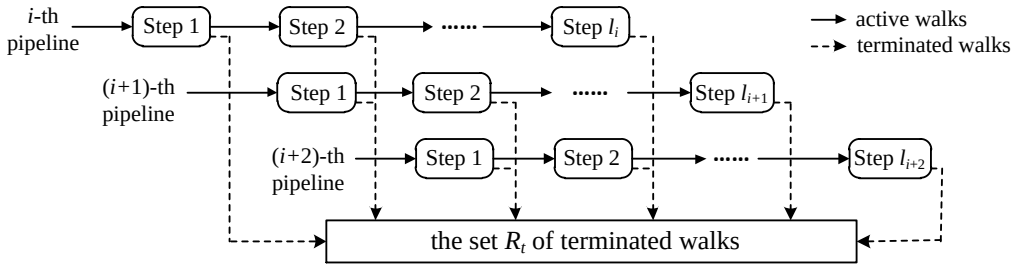


Figure 2: This is an illustration of the parallel pipeline framework. In this framework, there are $\lceil \omega/\gamma \rceil$ pipelines, each of which generates at most γ random walks by several steps. For each $i \in [1, \lceil \omega/\gamma \rceil]$, the $(i+1)$-th pipeline starts, only when the random walks in the i-th pipeline have completed the first step. All terminated walks in the pipelines are maintained in the set R_t, which are the final output of the computation of random walks.

probability α. Initially, we have the empty collections of terminated and active random walks, referred to as R_t and R_a respectively (Lines 1-2). Besides, the number ω of random walks is calculated as $\Omega(\frac{\log(1/p_f)}{\epsilon^2 \delta})$. Then, we generate the random walks by several iterations (Lines 4-16). Each iteration consists of three phases, as follows. In the first phase, for each node $v \in V$, if $\omega > 0$, then we create $\min(\gamma, \omega)$ number of paths $\langle v, v \rangle$ consisting of a single node v, and decrease ω by γ (Lines 5-8). In the second phase, for each path $P \in R_a$, we extend P with one step by selecting a node t the set of out-neighbors of the tail u of P according to the routing probability $r(u, t)$. After that, we append t to the end of P and mark P as terminated if α is not smaller than a random number generated on the fly (Lines 12-15). Finally, we remove from R_a the terminated paths, which are then added into R_t (Line 16). At the end of the algorithm, to compute $\hat{\pi}(s, t)$ for all $s, t \in V$, we calculate the number of paths P in R_t such that the head and tail of P are s and t respectively.

4 OPTIMIZATION TECHNIQUES

In this section, we develop several efficient techniques that significantly improve the performance of Algorithm 2.

4.1 Handling Large Nodes: The Alias Tree

As explained previously, we adopt the alias method which allows us to select one node t from s's out-neighbors, where $s \in V$, according to the routing probability with constant time and linear space complexity. However, when the degree of a node $u \in V$ is sufficiently large, due to that the size of $N_{out}(u)$ could be too large to fit in memory. To avoid this deficiency, we devise a hierarchical approach that recursively divides $N_{out}(u)$ into several subsets, each of which has a size not bigger than d, that can fit in memory, where d is a user-defined parameter and can be estimated according to the amount of available memory. Then, we denote each subset $S' \subseteq S$ as a new item with a weight equal to $\sum_{e \in S'} r(e)$. As such, we construct a new set $\mathcal{S}$ of items, whose alias and switch probability can be computed by the alias method on subsets of S. We build the *alias tree* by letting the alias and switch probability of S' be a child of $\mathcal{S}$ for each item $S' \in \mathcal{S}$. Apparently, the alias tree is multi-way tree, where each node has a similar number of children. If the size of the set $\mathcal{S}$ of the subsets exceeds the size of memory, we recursively divide $\mathcal{S}$ in the same way.

Algorithm 3: Construct-Alias-Tree(S, r, d)

Input: The set S of elements, the routing probability r, and the degree threshold d
Output: The alias tree $T(S)$ of S

1 **if** $|S| \leq d$ **then**
2 Let $T(S)$ be the output of the alias method on S;
3 **else**
4 Let $b = \lceil |S|/d \rceil$;
5 Split S into b disjoint subsets by random, referred to as $\mathcal{S} = \{S_1, S_2, \cdots, S_b\}$;
6 Let $w'(S_i) = \sum_{e \in S_i} r(e)$ for each $i \in [1, b]$;
7 Let $r'(S_i) = \frac{w'(S_i)}{\sum_{S' \in \mathcal{S}} w'(S')}$ for each $i \in [1, b]$;
8 Let $r_i(e) = \frac{r(e)}{\sum_{e' \in S_i} r(e')}$ for each $e \in S_i$ and $i \in [1, b]$;
9 For each $i \in [1, b]$, compute $f(S_i)$ by the alias method, which is treated as a child of $\mathcal{S}$;
10 Build $T(S)$ with Construct-Alias-Tree($\mathcal{S}$, r', d);
11 **return** $T(S)$.

Algorithm 3 provides the pseudo-code of the construction of an alias tree on a given set S of items. The algorithm first inspects the size of S. If its size is not bigger than d, then the algorithm returns the alias directly computed on S by the alias method, which forms a single node of the alias tree (Line 2). Otherwise, we randomly split S into b disjoint subsets (Line 5). Denote the set of subsets by $\mathcal{S} = \{S_1, S_2, \cdots, S_b\}$. Hence, we have $\cup_{S' \in \mathcal{S}} S' = S$ and $\cap_{S' \in \mathcal{S}} S' = \emptyset$. Then, we compute the weight of $S' \in \mathcal{S}$, which is the sum of routing probability of items in S', referred to as $w'(S') = \sum_{e \in S'} r(e)$. Then, for each subset S_i where $1 \leq i \leq b$, the routing probability of S_i with respect to $\mathcal{S}$ is computed as $r'(S_i) = \frac{w'(S_i)}{\sum_{S' \in \mathcal{S}} w'(S')}$ (Line 7). Besides, to compute the alias of each item e in each subset S_i, the routing probability of e with respect to S_i is updated as $r'(S_i) = \frac{w'(S_i)}{\sum_{S' \in S} w'(S')}$ (Line 8). As a result, the child $f(S_i)$ of $\mathcal{S}$ is created by performing the alias method on the updated routing probability of items in S_i

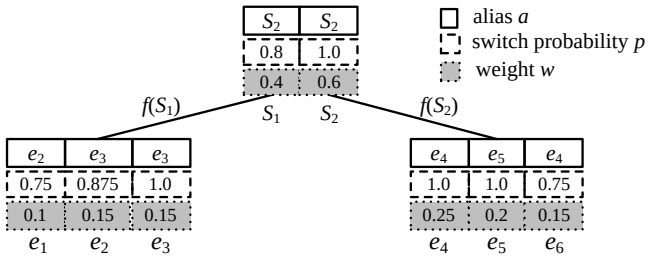


Figure 3: An example of the alias tree.

(Line 9). Finally, we recursively construct the alias tree on $\mathcal{S}$, until the size of $\mathcal{S}$ fits in memory, i.e., not bigger than d (Line 10).

Example 4.1. Consider the 6 out-going edges of v_0 in Figure 1, denoted by $e_1, e_2, \cdots, e_6$ respectively. As shown, in Figure 3, the alias tree stores the 6 edges by treating them as 6 items in a set S. Assume that the degree threshold is $d = 3$. As the size of S is larger than d, we randomly divide S into two sets, e.g., $S_1 = \{e_1, e_2, e_3\}$ and $S_1 = \{e_4, e_5, e_6\}$. Hence, we have $w'(S_1) = r(e_1)+r(e_2)+r(e_3) = 0.4$. Similarly, we have $w'(S_2) = 0.6$. As such, to obtain the routing probabilities, we can normalize the weights of elements in S_1, S_2, and $\mathcal{S}$ respectively. Then, by utilizing the alias method, we construct the aliases and switch probabilities of S_1 and S_2, which form the children of $\mathcal{S}$. After that, we recursively construct the alias of $\mathcal{S}$, which is returned as shown in Figure 3, since the size of $\mathcal{S}$ is less than 3. □

To select an item, we traverse the alias tree $T(S)$ recursively by starting from the root T of $T(S)$. To explain, we first select an item e from T with a probability $1/|T|$, which can be achieved in $O(1)$ time. Then, a random number $r \in [0, 1]$ is generated. If $r \leq p(e)$, then we select e, otherwise $a(e)$. Afterwards, if $f(e)$ is not empty, i.e., e has children, we recursively inspect the items in $f(e)$; otherwise, we return e as the result. Apparently, The running time of the selection process is determined by the height of the alias tree, i.e., $O(\log(|S|)/\log(d))$, where d is the degree threshold. Lemma 1 provides the correctness of random item generation based on the alias tree.

Example 4.2. Consider the alias tree in Example 2.3. To select an element, it consists of two steps: First, we select an element S' from $\mathcal{S}$; Then, an element e is selected from S'. In the first step, assume that S_1 is selected with the probability $1/|\mathcal{S}| = 0.5$. After that, we generate a random number $r \in [0, 1]$. If $r \leq p(S_1)$, we select S_1, otherwise select $a(S_1) = S_2$. As a result, the probability of selecting S_1 is $0.5 \cdot p(S_1) = 0.4$, which equals the normalized weight of S_1, i.e., the routing probability of S_1. Assume that S_1 is selected in the first step. We recursively select an element from S_1 in the same way, which leads to the final selection. □

Lemma 1. *Given a set S of edges, as well as the alias tree T constructed on S, the selection an element $e \in S$ from T has the probability equal to $r(e)$.*

Proof. Let $\langle S_1, S_2, \cdots, S_i \rangle$ be the path from the root of T to a leaf of T, where e resides. Consider a node S_j, where $1 \leq j \leq i$. As shown previously, each item x in S_j is selected with the probability $w(x)/\sum_{y \in S_j} w(y)$. Besides, the weight of S_j is the sum of weights of items in S_j, i.e., $w(S_j) = \sum_{y \in S_j} w(y)$. Note that, $w(S_1) = \sum_{e \in S} r(e) = 1$, since S_1 represents all items in S. Therefore, the probability of selecting e is

$$\frac{w(S_2)}{w(S_1)} \cdot \frac{w(S_3)}{w(S_2)} \cdot \cdots \cdot \frac{w(e)}{w(S_i)} = \frac{r(e)}{w(S_1)} = r(e).$$

□

Algorithm 4: Pre-compute-Big-Moves(G, V_s, d, α)

Input: The graph $G = (V, E, w)$, the set $V_s \subseteq V$ of small nodes, the degree threshold d, and the termination probability α

Output: The big moves for all $v \in V_s$

1 Let $prevBSize = 0$;
2 Let $F(v) = \{\langle u, p(v, u)\rangle \mid u \in N_{out}(v)\}$ for each $v \in V_s$;
3 Let $B(v) = F(v)$ for each $v \in V_s$;
4 **while** $|B(v)| + |F(v)| < d$ and $prevBSize < |B(v)|$ **do**
5 Update $prevBSize$ as $prevBSize = |B(v)|$;
6 Let $F'(v)$ be an empty collection for each $v \in V_s$;
7 **for** each node $\langle u, p(v, u)\rangle \in F(v)$ **in parallel do**
8 **for** each node $t \in N_{out}(u)$ **in parallel do**
9 Add $\langle t, p(v, u) \cdot (1 - \alpha) \cdot p(u, t)\rangle$ into $F'(v)$;
10 Update $B(v)$ as the aggregation result on the pairs $\langle t, p\rangle$ in $F'(v) \cup B(v)$ by t;
11 Replace $F(v)$ with $F'(v)$;
12 Let the set of terminated big moves be $P_t^b(v) = \{\langle t, 0, p \cdot \alpha\rangle \mid \langle t, p\rangle \in B(v)\}$ for $v \in V_s$;
13 Let the set of active big moves be $P_a^b(v) = \{\langle t, 1, p \cdot (1 - \alpha)\rangle \mid \langle t, p\rangle \in F(v)\}$ for $v \in V_s$;
14 **return** $P_t^b(v) \cup P_a^b(v)$ for all $v \in V_s$.

4.2 Handling Small Nodes: Make Big Moves

Most real-life graphs follow the power law [6], whose implication is that the majority of nodes in the graph have a very small set of neighbors (see Table 2). To speed up the random walks among the small-degree nodes, we devise a pre-computing method that generates the *big moves* for each small-degree node, such that each big move represents a random walk of several steps. To facilitate the computation with big moves, we maintain each big move P^b with the similar structure as the normal random walk with a difference in that we add a mark, denoted by $M(P^b)$, to represent whether P^b is terminated or not. If $M(P^b) = 0$, then P^b is a terminated path; otherwise, if $M(P^b) = 1$, then P^b is an active path. As a result, we can denote a big move of v by $\langle u, t_m, p\rangle$ which are the walks ending at u with the termination mark t_m and the probability p. In other words, compared with the set of out-neighbors, the big moves consists of the event of termination, which expands the probability space.

Algorithm 4 describes the generation of big moves for each node v in the set V_s of small nodes. Initially, for each $v \in V_s$, we have the set $F(v)$ of active walks be the set of out-neighbors u of v with the

accumulated probability equal to $p(v,u)$, i.e., $F(v) = \{\langle u, p(v,u)\rangle \mid u \in N_{out}(v)\}$. Besides, we let the set $B(v)$ of the terminated walks be initialized as $F(v)$, and we denote the size of $B(v)$ before updating (Line 10) by *prevBSize*, which is initialized as 0. Note that, the initial sizes of $F(v)$ and $B(v)$ equal the size of $N_{out}(v)$. In what follows, the algorithm runs in several iterations. In each iteration, we extend each walk in $F(v)$ with one more step by adding each node $u \in N_{out}(v)$, resulting in a set $F'(v)$ of new walks. Then, we compute the accumulated paths in both $B(v)$ and $F'(v)$, and also update $F(v)$ with new walks (Lines 5-11). These processes are repeated in iterations, until (i) the number of terminated and active walks exceeds d, i.e., $|B(v)| + |F(v)| \geq d$, or (ii) the walks can not reach any new nodes, i.e., the size of $B(v)$ does not change any more, namely $prevBSize = |B(v)|$. Finally, we integrate the walks in $B(v)$ and $F(v)$ by multiplying the probability in $B(v)$ and $F(v)$ with α and $1-\alpha$ respectively, and adding the termination mark 0 and 1 respectively, which leads to resulting set $BM(v)$ of big moves of v.

Example 4.3. Consider the graph $G = (V, E, w)$ in Figure 1. Assume that the degree threshold $d = 4$ and the termination probability $\alpha = 0.5$. Then, all the nodes $v \in V$, except v_0, need to compute their big moves, since $|N_{out}(v)| < d$. Consider node v_2, which has one out-neighbor v_3. To compute the big move of v_2, we first initialize the set $F(v_2)$ as $\{\langle v_3, 1\rangle\}$ and the set $B(v_2) = F(v_2)$. Since $|F(v_2)|+|B(v_2)| = 2 < d$, we extend each walk in $F(v_2)$ by one step, and obtain the set $F'(v_2) = \{\langle v_4, 1 \times 0.5 \times 1 = 0.5\rangle\}$ as v_3 has only one neighbor, i.e., v_4. Then, we update $prevBSize = |B(v_2)| = 1$, and update $B(v_2)$ as the aggregation of items in $B(v_2)$ and $F'(v_2)$, resulting in the set $\{\langle v_3, 1\rangle, \langle v_4, 0.5\rangle\}$. In addition, $F(v_2)$ is updated as $F'(v_2)$, i.e., $F(v_2) = \{\langle v_4, 0.5\rangle\}$. Afterwards, we continue to extend the walks in $F(v_2)$, since $|F(v_2)| + |B(v_2)| = 3 < d$, and we have $F'(v_2) = \{\langle v_3, 0.5 \times 0.5 \times 1 = 0.25\rangle\}$. Now, we update $prevBSize = |B(v_2)| = 2$, update $F(v_2) = F'(v_2)$, and update $B(v_2)$ as the aggregation of items in $F'(v_2)$ and $B(v_2)$, leading to the set $\{\langle v_3, 1.25\rangle, \langle v_4, 0.5\rangle\}$. Because $prevBSize = |B(v_2)|$, i.e., the size of $B(v_2)$ does not change any more, we stop the iteration, and compute the set $P_t^b(v_2)$ of terminated big moves as $\{\langle v_3, 0, 1.25 \times 0.5 = 0.625\rangle, \langle v_4, 0, 0.5 \times 0.5 = 0.25\rangle\}$ and the set $P_a^b(v_2)$ of active big moves as $\{\langle v_3, 1, 0.25{\times}0.5 = 0.125\rangle\}$. As a result, we have the big moves of v_2 as the set $BM(v_2) = P_t^b(v_2) \cup P_a^b(v_2) = \{\langle v_3, 0, 0.625\rangle, \langle v_4, 0, 0.25\rangle, \langle v_3, 1, 0.125\rangle\}$. □

We show the correctness of Algorithm 4 by Lemma 2.

Lemma 2. *Given a set $BM(v)$ of big moves of v, then $BM(v)$ consists of all random walks starting from v within k steps, where k is the longest length of walks in $BM(v)$.*

Proof. It is sufficient to show that (i) if there exists a path of length at most k that starts from v and ends at u, then u must be in $BM(v)$; and (ii) $\sum_{\langle v, t_m, p\rangle \in BM(v)} p = 1$.

In the first case, we are to show that all paths, whose lengths are not larger than k, are in $BM(v)$. Let $P(v, u_t) = \langle v, u_1, u_2, \cdots, u_t\rangle$ be a path of length $t \leq k$ that starting from v. In the first round, we have u_1 selected, as u_1 is an out-neighbor of v, i.e., $u_1 \in N_{out}(v)$. By induction, u_{i+1} is an out-neighbor of u_i, where $1 \leq i < t$. As such, the sequence $\langle v, u_1, u_2, \cdots, u_t\rangle$ is a path of G.

In the second case, we need to demonstrate that the probability space of $BM(v)$ is completed. Let p_t be the probability that a random walk $P = \langle v, u_1, u_2, \cdots u_t\rangle$ is of length t. Hence, we have

$$\begin{aligned} p_t &= \sum_{\ell(P)=t} p(\langle v, u_1\rangle) p(\langle u_1, u_2\rangle) \cdots p(\langle u_{t-1}, u_t\rangle)(1-\alpha)^{t-1} \\ &= (1-\alpha)^{t-1}. \end{aligned}$$

Consider the probability p' of the event that a random walk $P \in BM(v)$ is a terminated path, which means that P is of length $t \leq k$. As $BM(v)$ contains all paths of length not bigger than k, we have

$$p' = \alpha \cdot \sum_{1 \leq t \leq k} p_t = \alpha \cdot \sum_{1 \leq t \leq k} (1-\alpha)^{t-1} = 1 - (1-\alpha)^k.$$

Besides, consider the probability p'' of the event that a random walk $P \in BM(v)$ is an active path, which indicates that P is of length k. Therefore, we have $p'' = (1-\alpha) \cdot p_k = (1-\alpha)^k$. To sum up, we obtain $\sum_{\langle v, t_m, p\rangle \in BM(v)} p = p' + p'' = 1$, which completes the proof.

□

To continue the generation of the random walk ending with the node v, which is a small node with the set $BM(v)$ of big moves, we randomly select a path $P^b = \langle v, t_m, p\rangle \in BM(v)$ following the distribution of probabilities of paths in $BM(v)$. Note that, the selection can be accomplished in constant time with the alias method. If $t_m = 0$, then P^b is a terminated path, and we add P^b into the set R_t. Otherwise, we continue the walk along the path P^b until the termination.

4.3 Optimizing γ

Our previous discussion has focused on the generation of random walks in the graph G. When the number ω of random walks is considerably large, it could be highly inefficient to generate all random walks for all nodes of G at a time, as discussed in the previous sections. To remedy this deficiency, we proposed the parallel pipeline framework that processes the ω number of random walks by several iterations, each of which generates at most γ random walks for each node of G. That is, there will be $\lceil \omega/\gamma \rceil$ number of pipelines.

One crucial question remains: how do we decide the number γ of random walks in each pipeline? A straightforward approach is to set $\gamma = 1$, i.e., we generate one random walk for each node $v \in V$ in each pipeline. Although this can avoid the memory issues as aforementioned, it brings up the other deficiencies where (i) the probability to having a long walk is low, which leads to the under-utilization of computing resources, and (ii) the number of pipelines would be a lot, which increases the running time. To tackle these issues, we devise a heuristic method to choose γ, as follows. First, in our implementation, we maintain a path P by keeping the head s and the tail t of P, i.e., $\langle s, t\rangle$. As such, it is a constant space cost c for a path P. Second, in the parallel pipeline framework, we start the pipelines sequentially such that one latter pipeline L is started only upon that the pipeline L', prior to L, completes the extension of random walks in L' by one step. Specifically, let s_i be the total number of active random walks that are generated by the first i pipelines. After the extension of one step at the i-th round, the number of remaining random walks is expected to be $(1-\alpha) \cdot s_i$.

Table 2: Datasets.

Name	$\|V\|$	$\|E\|$	d_{avg}	d_{max}	$\|V_s\|/\|V\|$	$\|V_l\|/\|V\|$
GrQc	5,241	28,968	5.5	81	78.3%	15.9%
CondMat	23,133	186,878	8.1	279	75.4%	12.2%
Enron	36,692	367,662	10.0	1383	85.5%	5.1%
DBLP	317,080	1,049,866	5.6	306	46.9%	4.2%
Stanford	281,903	2,312,497	8.2	255	75.5%	14.7%
Google	875,713	5,105,039	6.9	456	53.6%	1.9%
Skitter	1,696,415	11,095,298	11.5	35,387	51.0%	0.3%
Patents	3,774,768	16,518,947	7.9	770	38.5%	1.5%
Pokec	1,632,803	30,622,564	21.4	8,763	61.0%	2.6%
LiveJournal	4,846,609	68,475,391	15.9	20,292	66.5%	1.0%
Orkut	3,072,441	117,185,083	43.0	33,007	65.7%	3.3%
Twitter	41,652,230	1,468,364,884	36.6	2,997,469	87.9%	0.2%
Friendster	68,349,466	2,586,147,869	46.1	5,214	65.5%	12.8%

In the $(i+1)$-th pipeline, we start another $n \cdot \gamma$ number of new random walks in total, where n is the number of nodes of G. To sum up, the total expected number of random walks to be processed at the $(i+1)$-th round is $s_{i+1} = n \cdot \gamma + (1-\alpha) \cdot s_i$. Note that, at the beginning, we have $s_1 = n \cdot \gamma$. By solving these formulas, we have

$$s_i = n \cdot \gamma \cdot \sum_{1 \le j \le i} (1-\alpha)^{j-1} = n \cdot \gamma \cdot \frac{1-(1-\alpha)^i}{\alpha} < \frac{n \cdot \gamma}{\alpha}.$$

As such, if the size of available memory on a machine is M, then we have $c \cdot n \cdot \gamma/\alpha \le M$. That is, we obtain $\gamma \le \alpha \cdot M/(n \cdot c)$. Therefore, we can set γ to the maximum integer such that the total size of processed random walks does not exceed the size of available memory, which helps the full utilization of the cluster. By experiments as shown later, we demonstrate that our choice of γ results in good performance of the proposed algorithm.

5 EXPERIMENTS

This section presents the thorough experimental studies on the performance of the proposed algorithm, compared with the state-of-the-art approach, on several real-life datasets.

5.1 Experimental Settings

We implement our distributed algorithm for the computation of FAPPR (dubbed as *DistPPR*) in Scala on Spark 2.0[1], and compare it against two state-of-the-art distributed algorithms: *Doubling* [2] and the PPR library in GraphX[2] (denoted as *GXPPR*). Note that, (i) *Doubling* is a MapReduce algorithm, which can be easily implemented on Spark, and (ii) *GXPPR* is a matrix-based method that iteratively computes with the adjacency matrix of the graph until the convergence of PPR. All of our experiments are conducted on an in-house cluster consisting of up to 201 machines, each of which runs CentOS, and has 16GB memory and 12 Intel Xeon Processor E5-2670 CPU cores. By default, we exploit 51 machines to run the algorithms, where we have one machine as the master and the others as the workers in Spark.

We use 13 real-life networks in our experiments, as shown in Table 2. These data are obtained from the SNAP collection[3], and are intensively evaluated in the literatures [27, 28]. Besides, these data are from various categories: *GrQc* and *CondMat* are collaboration networks; *Enron* is a communication network; *Standford* and *Google* are the web graphs; *Skitter* is a internet topology network; *Patents* and *DBLP* are the citation networks; *Pokec*, *LiveJournal*, *Orkut*, *Twitter*, and *Friendster* are social networks. Furthermore, these data have different sizes, ranged from thousands of edges to billions of edges. Their average out-degrees (denoted by d_{avg}) are relatively small, though their maximum out-degrees (denoted by d_{max}) could be significantly large that are more than 2 million. Let V_s be the set of *small* nodes whose out-degrees are less than d_{avg}, and let V_l be the set of *large* nodes whose out-degrees are more than $\sqrt{d_{max}}$. As shown in Table 2, the small nodes are the majorities in the graph, while the large nodes take only a small portion. Following previous work [29], we adopt the linear parameterization to assign each edge a positive weight for each graph.

In addition, according to previous work [7, 27], we set the failure probability p_f to $1/n$ where n is the number of nodes in the graph, and set the default value of the relative accuracy guarantee ϵ, the threshold δ, and the termination probability α to 0.5. Besides, if the algorithm does not terminate within 24 hours, we omit it from the experiments. We repeat each experiment 3 times and report the average reading of each approach.

5.2 Comparisons with Previous Work

In the first set of experiments, we demonstrate the superiority of our proposed algorithm by evaluating the running time of all the algorithms on all datasets. Note that, the running time of *DistPPR* includes both the time of the optimization techniques, i.e., we compare the running time of all algorithms with the same input data. Figure 5(a) shows the results of *DistPPR*, *Doubling*, and *GXPPR* on the datasets in Table 2. *DistPPR* outperforms all the other methods by up to 2 orders of magnitude on all datasets. This is because *DistPPR* addresses the issues of both large and small nodes and is able to leverage the massive parallelism of the distributed computing. However, *Doubling* spends lots of efforts on combining the short walks that results in a huge combinational space, and *GXPPR* incurs significant overhead in the operations of matrix which could be highly sparse. Besides, *GXPPR* is able to process only two small datasets, which again demonstrates that the matrix-based methods are difficult to handle large graphs that require huge memory space. Furthermore, the results of *Doubling* on the datasets, *LiveJournal*, *Orkut*, *Twitter*, and *Friendster*, are omitted, since the running time of *Doubling* on those datasets exceeds 24 hours. On the contrary, *DistPPR* is able to handle all datasets with a relatively short running time. What's more, compared to *Doubling* and *GXPPR*, the performance of *DistPPR* is less sensitive to the number of edges in the graph.

Besides, to evaluate the accuracy of the approximate solutions, we generate the ground-truth by the exact PPR algorithm [29] that is able to process the graphs with less than ten millions edges on a machine with 48GB memory. Based on that, we evaluate the accuracy of *DistPPR* and *Doubling* on 6 datasets by NDCG and MAP, which are widely used to evaluate the results of ranking[4]. In particular, given the graph $G = (V, E, w)$, for each node $v \in V$,

[1] https://spark.apache.org/
[2] https://spark.apache.org/graphx/
[3] http://snap.stanford.edu/data/index.html
[4] https://spark.apache.org/docs/2.3.0/mllib-evaluation-metrics.html#ranking-systems

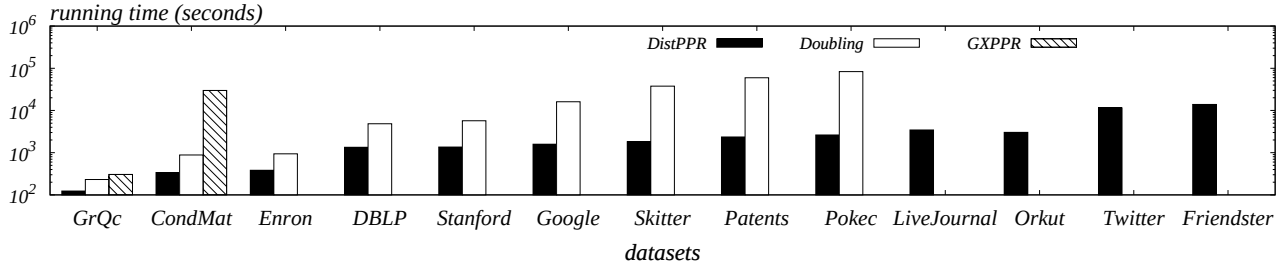


Figure 4: Running time on real-life datasets.

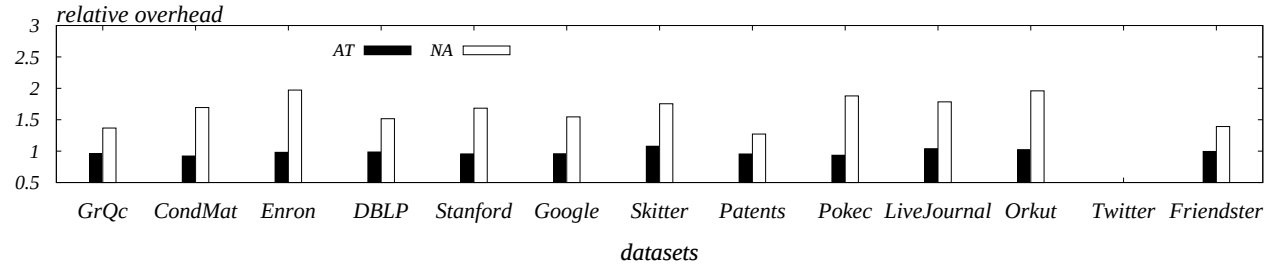


Figure 5: Effects of optimizations.

we rank the nodes $u \in V$ by the value $\pi(v, u)$. We take the first 1000 nodes from each ranking for evaluation. Figure 6 shows the results of NDCG and MAP on several datasets for both *DistPPR* and *Doubling* respectively. On all datasets, the NDCG of *DistPPR* outperforms the one of *Doubling* by up to 21%, and the MAP of *DistPPR* is also better than the one of *Doubling* by up to 19%. This is because *Doubling* does not guarantee the accuracy of results, whereas *DistPPR* has a strong accuracy guarantee according to Definition 2.2.

5.3 Effects of Optimizations

In this section of experiments, we study the effects of the three proposed optimization techniques: the alias tree (AT) method for handling large nodes, the big move (BM) technique for accelerating the walks on small nodes, and the technique that auto-tunes γ.

We first evaluate the techniques of AT and BM. We consider two versions of algorithms modified from *DistPPR*: *DistPPR* but with the two optimizations disabled (denoted as *NA*), and *DistPPR* but with only the AT enabled (denoted as *AT*). We define the *relative overhead* of each modified version of *DistPPR* on a dataset D as its running time of D divided by the running time of *DistPPR* with all optimizations enabled. Figure 5(b) shows the relative overheads of *AT* and *NA* on each dataset. Note that, the results on *Twitter* are not shown, as both *AT* and *NA* fail to process *Twitter* whose node degree is highly skew. The relative overheads of *AT* are around 1 in all cases, which indicates that the AT technique allows the algorithm avoiding the issues of large nodes and brings very few overheads in the running time. Meanwhile, the relative overhead of *NA* is around 1.5 on all datasets, implying that the BM technique improves the efficiency of *DistPPR* by a factor of 1.5.

Then, we evaluate the performance of the technique that automatically tunes γ. Recall that $\gamma \leq \omega$ is the parameter in the parallel pipeline framework that decides the number of random walks to generate in a pipeline. To evaluate our choice of γ, we measure the running time of *DistPPR* on the largest dataset *Friendster* by varying γ from 30 to 480, and compare with our autotune technique. We plot the results in Figure 7(a). Observe that the optimal value for γ is around 120, and our automatic choice of γ leads to a performance identical to the optimum. On the contrary, compared to the optimum, when $\gamma \leq 120$, it leads to longer running time, since the number of pipelines increases. Note that, the number of pipelines is $\lceil \omega/\gamma \rceil$. On the other hand, when $\gamma \geq 240$, it overloads the cluster with an excessive number of random walks to process simultaneously, which degrades the performance of *DistPPR*.

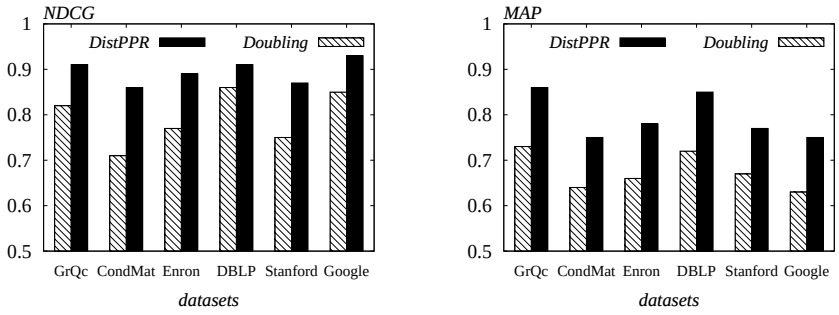
In summary, the three optimization techniques improve the efficiency of *DistPPR* by up to 2 times, and help scale *DistPPR* to large graphs whose node degrees could be highly skew.

5.4 Varying Other Parameters

In the last set of experiments, we study the sensitive of *DistPPR* to the other parameters, i.e., the number of workers on Spark and the termination probability α. Note that, we omit the studies on the threshold δ, as δ has the same effect as α to the number ω of random walks. We adopt the largest dataset *Friendster* in all subsequent experiments.

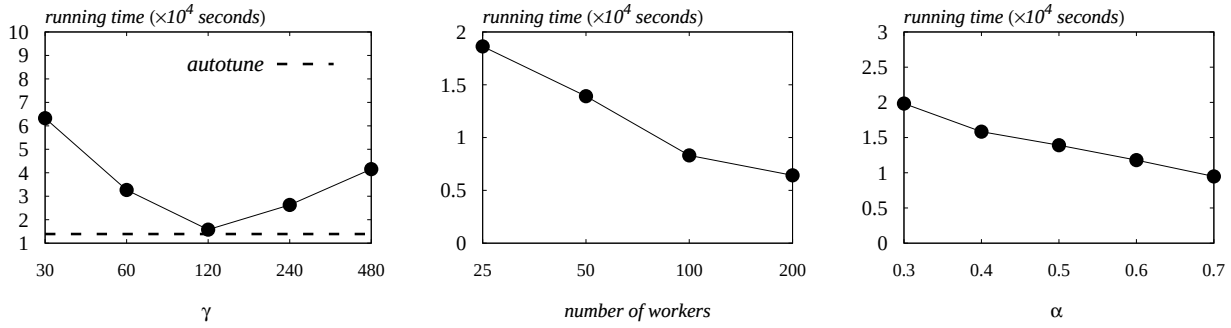
Figure 7(b) reports the running time of *DistPPR* on *Friendster* dataset by varying the number of workers on Spark. When we increase the number of workers, the running time of *DistPPR* decreases accordingly. However, when the number of workers ranges from 100 to 200, the decrease rate of *DistPPR*'s running time becomes low, since the cost of communication between workers increases.

Figure 7(c) shows the performance of *DistPPR* by varying the termination probability α. When α increases, it increases the



(a) NDCG. (b) MAP.

Figure 6: Accuracy comparison on NDCG and MAP.



(a) Varying γ. (b) Varying the number of workers. (c) Varying α.

Figure 7: Varying parameters.

probability of a random walk to be terminated at a certain step, leading to the decrease of the total amount of computation.

6 RELATED WORK

There exist some work [2, 22] that exploit the distributed computing techniques to accelerate the computation of the fully PPR or PR on large graphs based on the Monte Carlo approximation. They first generate some random walks of relatively short length, and then combine the short walks to obtain the random walks of long length. In particular, given two short walks $P_1 = \langle v_{h1} \cdots v_{t1} \rangle$ and $P_2 = \langle v_{h2} \cdots v_{t2} \rangle$, we can combine P_1 and P_2 if and only if the tail v_{t1} of P_1 equals the head v_{h2} of P_2, i.e., $v_{t1} = v_{h2}$, which results in a path $P = \langle v_{h1}, \cdots, v_{t1}, \cdots, v_{t2} \rangle$ of length $\ell(P_1) + \ell(P_2) - 1$. To make the combined paths sufficiently random, however, the number of short walks could be extremely large, especially for the large-degree nodes, rendering these approaches inefficient. Besides, there is no guarantee on the accuracy of the results produced by these approaches [2, 22].

Besides, some distributed algorithms are proposed to answer the queries of single-pair PPR [16] or single-source PPR [9, 11, 16], which differ from the problem of this paper that focuses on the computation of fully PPR. Additionally, there are some distributed and parallel algorithms for graph data [10, 14, 15], but they do not aim for the problem of fully PPR computation.

On the other hand, Fogaras et al. [7] developed an approach on a single machine for the fully PPR by the Monte Carlo approximation. In particular, to facilitate the traversal on the graph, their approach relies on an indexing structure of the graph that is required by the computation of random walks for all nodes, which renders this approach difficult to be implemented as a distributed algorithm. Moreover, there also exist a plethora of techniques for processing the queries of single-pair PPR [17, 18, 26], single-source PPR [12, 24, 27], and top-k PPR [28] on a single machine, all of which have a different problem setting against the fully PPR. Besides, these techniques often require a preprocessing step, which are sequential algorithms, rendering them difficult to be translated in parallel.

Finally, Xie et al. [29] addressed an important issue that edge-weighted graphs widely exist in various applications, which is ignored by previous studies. However, their approach requires to compute based on the adjacency matrix of the graph, which could be prohibitively large.

7 CONCLUSIONS

This paper studies the problem of the fully Personalized PageRank, which has enormous applications in link prediction and recommendation systems for social networks. We approach this problem by devising an efficient distributed algorithm to accelerate the Monte Carlo approximation, which requires to generate a large number of random walks for each node of the graph. We exploit the parallel pipeline framework to achieve the superiority of cluster parallelism. To optimize the proposed algorithm, we develop three techniques that significantly improve the performance of the proposed algorithm in terms of efficiency and scalability. With extensive experiments on various real-life networks, we demonstrate that our proposed solution is up to 2 orders of magnitude faster than the start-of-the-art solutions,

and also outperforms the baseline solutions on the evaluation of accuracy.

REFERENCES

[1] Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank for Word Sense Disambiguation. In *EACL 2009, 12th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, Athens, Greece, March 30 - April 3, 2009*. 33–41.

[2] Bahman Bahmani, Kaushik Chakrabarti, and Dong Xin. 2011. Fast personalized PageRank on MapReduce. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June 12-16, 2011*. 973–984.

[3] Jeffrey Dean and Sanjay Ghemawat. 2004. MapReduce: Simplified Data Processing on Large Clusters. In *6th Symposium on Operating System Design and Implementation (OSDI 2004), San Francisco, California, USA, December 6-8, 2004*. 137–150.

[4] Pavlos S. Efraimidis and Paul G. Spirakis. 2006. Weighted random sampling with a reservoir. *Inf. Process. Lett.* 97, 5 (2006), 181–185.

[5] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A System for Recommending 3+ Billion Items to 200+ Million Users in Real-Time. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. 1775–1784.

[6] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. 1999. On Power-law Relationships of the Internet Topology. In *SIGCOMM*. 251–262.

[7] Dániel Fogaras, Balázs Rácz, Károly Csalogány, and Tamás Sarlós. 2005. Towards Scaling Fully Personalized PageRank: Algorithms, Lower Bounds, and Experiments. *Internet Mathematics* 2, 3 (2005), 333–358.

[8] Yasuhiro Fujiwara, Makoto Nakatsuji, Takeshi Yamamuro, Hiroaki Shiokawa, and Makoto Onizuka. 2012. Efficient personalized pagerank with accuracy assurance. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*. 15–23.

[9] Tao Guo, Xin Cao, Gao Cong, Jiaheng Lu, and Xuemin Lin. 2017. Distributed Algorithms on Exact Personalized PageRank. In *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*. 479–494.

[10] Qirong Ho, Wenqing Lin, Eran Shaham, Shonali Krishnaswamy, The Anh Dang, Jingxuan Wang, Isabel Choo Zhongyan, and Amy She-Nash. 2016. A Distributed Graph Algorithm for Discovering Unique Behavioral Groups from Large-Scale Telco Data. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*. 1353–1362.

[11] Jinhong Jung, Namyong Park, Lee Sael, and U. Kang. 2017. BePI: Fast and Memory-Efficient Method for Billion-Scale Random Walk with Restart. In *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*. 789–804.

[12] Jung Hyun Kim, K. Selçuk Candan, and Maria Luisa Sapino. 2013. LR-PPR: locality-sensitive, re-use promoting, approximate personalized pagerank computation. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*. 1801–1806.

[13] Wei-keng Liao, Alok N. Choudhary, Donald D. Weiner, and Pramod K. Varshney. 2005. Performance Evaluation of a Parallel Pipeline Computational Model for Space-Time Adaptive Processing. *The Journal of Supercomputing* 31, 2 (2005), 137–160.

[14] Wenqing Lin, Xiaokui Xiao, and Gabriel Ghinita. 2014. Large-scale frequent subgraph mining in MapReduce. In *IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014*. 844–855.

[15] Wenqing Lin, Xiaokui Xiao, Xing Xie, and Xiaoli Li. 2017. Network Motif Discovery: A GPU Approach. *IEEE Trans. Knowl. Data Eng.* 29, 3 (2017), 513–528.

[16] Qin Liu, Zhenguo Li, John C. S. Lui, and Jiefeng Cheng. 2016. PowerWalk: Scalable Personalized PageRank via Random Walks with Vertex-Centric Decomposition. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*. 195–204.

[17] Peter Lofgren, Siddhartha Banerjee, and Ashish Goel. 2016. Personalized PageRank Estimation and Search: A Bidirectional Approach. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, San Francisco, CA, USA, February 22-25, 2016*. 163–172.

[18] Peter Lofgren, Siddhartha Banerjee, Ashish Goel, and Seshadhri Comandur. 2014. FAST-PPR: scaling personalized pagerank estimation for large graphs. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. 1436–1445.

[19] Siqiang Luo, Xiaokui Xiao, Wenqing Lin, and Ben Kao. 2019. Efficient Batch One-Hop Personalized PageRanks. In *IEEE 35th International Conference on Data Engineering, Macau SAR, ICDE 2019, China, April 8 - April 12, 2019*.

[20] Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik, James C. Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. 2010. Pregel: a system for large-scale graph processing. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, Indianapolis, Indiana, USA, June 6-10, 2010*. 135–146.

[21] Phuong Nguyen, Paolo Tomeo, Tommaso Di Noia, and Eugenio Di Sciascio. 2015. An evaluation of SimRank and Personalized PageRank to build a recommender system for the Web of Data. In *Proceedings of the 24th International Conference on World Wide Web Companion, WWW 2015, Florence, Italy, May 18-22, 2015 - Companion Volume*. 1477–1482.

[22] Atish Das Sarma, Anisur Rahaman Molla, Gopal Pandurangan, and Eli Upfal. 2015. Fast distributed PageRank computation. *Theor. Comput. Sci.* 561 (2015), 113–121.

[23] Yufei Tao, Wenqing Lin, and Xiaokui Xiao. 2013. Minimal MapReduce algorithms. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22-27, 2013*. 529–540.

[24] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast Random Walk with Restart and Its Applications. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), 18-22 December 2006, Hong Kong, China*. 613–622.

[25] Michael D. Vose. 1991. A Linear Algorithm For Generating Random Numbers With a Given Distribution. *IEEE Trans. Software Eng.* 17, 9 (1991), 972–975.

[26] Sibo Wang, Youze Tang, Xiaokui Xiao, Yin Yang, and Zengxiang Li. 2016. HubPPR: Effective Indexing for Approximate Personalized PageRank. *PVLDB* 10, 3 (2016), 205–216.

[27] Sibo Wang, Renchi Yang, Xiaokui Xiao, Zhewei Wei, and Yin Yang. 2017. FORA: Simple and Effective Approximate Single-Source Personalized PageRank. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. 505–514.

[28] Zhewei Wei, Xiaodong He, Xiaokui Xiao, Sibo Wang, Shuo Shang, and Ji-Rong Wen. 2018. TopPPR: Top-k Personalized PageRank Queries with Precision Guarantees on Large Graphs. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*. 441–456.

[29] Wenlei Xie, David Bindel, Alan J. Demers, and Johannes Gehrke. 2015. Edge-Weighted Personalized PageRank: Breaking A Decade-Old Performance Barrier. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*. 1325–1334.

[30] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: Cluster Computing with Working Sets. In *2nd USENIX Workshop on Hot Topics in Cloud Computing, HotCloud'10, Boston, MA, USA, June 22, 2010*.

[31] Wanxin Zhang, Dongsheng Li, Ying Xu, and Yiming Zhang. 2016. Shuffle-efficient distributed Locality Sensitive Hashing on spark. In *IEEE Conference on Computer Communications Workshops, INFOCOM Workshops 2016, San Francisco, CA, USA, April 10-14, 2016*. 766–767.

Improving Outfit Recommendation with Co-supervision of Fashion Generation

Yujie Lin*
Shandong University
Jinan, China
yu.jie.lin@outlook.com

Pengjie Ren*
University of Amsterdam
Amsterdam, The Netherlands
p.ren@uva.nl

Zhumin Chen
Shandong University
Jinan, China
chenzhumin@sdu.edu.cn

Zhaochun Ren
Shandong University
Jinan, China
zhaochun.ren@sdu.edu.cn

Jun Ma
Shandong University
Jinan, China
majun@sdu.edu.cn

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

The task of fashion recommendation includes two main challenges: *visual understanding* and *visual matching*. Visual understanding aims to extract effective visual features. Visual matching aims to model a human notion of compatibility to compute a match between fashion items. Most previous studies rely on recommendation loss alone to guide visual understanding and matching. Although the features captured by these methods describe basic characteristics (e.g., color, texture, shape) of the input items, they are not directly related to the visual signals of the output items (to be recommended). This is problematic because the aesthetic characteristics (e.g., style, design), based on which we can directly infer the output items, are lacking. Features are learned under the recommendation loss alone, where the supervision signal is simply whether the given two items are matched or not.

To address this problem, we propose a neural co-supervision learning framework, called the FAshion Recommendation Machine (FARM). FARM improves visual understanding by incorporating the supervision of generation loss, which we hypothesize to be able to better encode aesthetic information. FARM enhances visual matching by introducing a novel layer-to-layer matching mechanism to fuse aesthetic information more effectively, and meanwhile avoiding paying too much attention to the generation quality and ignoring the recommendation performance.

Extensive experiments on two publicly available datasets show that FARM outperforms state-of-the-art models on outfit recommendation, in terms of AUC and MRR. Detailed analyses of generated and recommended items demonstrate that FARM can encode better features and generate high quality images as references to improve recommendation performance.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Outfit matching; Fashion generation; Fashion recommendation

ACM Reference Format:
Yujie Lin, Pengjie Ren*, Zhumin Chen, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2019. Improving Outfit Recommendation with Co-supervision of Fashion Generation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313614

1 INTRODUCTION

Fashion recommendation has attracted increasing attention [14, 18, 20] for its potentially wide applications in fashion-oriented online communities such as, e.g., Polyvore[1] and Chictopia.[2] By recommending fashionable items that people may be interested in, fashion recommendation can promote the development of online retail by stimulating people's interests and participation in online shopping. In this paper, we target *outfit recommendation*, that is, given a top (i.e., upper garment), we need to recommend a list of bottoms (e.g., trousers or skirts) from a large collection that best match the top, and vice versa. Specifically, we allow users to provide some descriptions as conditions that the recommended items should accord with as much as possible.

Unlike conventional recommendation tasks, outfit recommendation faces two main challenges: *visual understanding* and *visual matching*. Visual understanding aims to extract effective features by building a deep understanding of fashion item images. Visual matching requires modeling a human notion of the compatibility between fashion items [41], which involves matching features such as color and shape etc. Early studies into outfit recommendation rely on feature engineering for visual understanding and traditional machine learning for visual matching [16]. For example, Iwata et al. [15] define three types of feature, i.e., color, texture and local descriptors such as Scale Invariant Feature Transform (SIFT) (for visual understanding), and propose a recommendation model based on Graphical Models (GM) (for visual matching). Liu et al. [29] define five types of feature including Histograms of Oriented Gradient (HOG) [9], Local Binary Pattern (LBP) [1], color moment, color histogram and skin descriptor [5] (for visual understanding), and propose a latent Support Vector Machine (SVM) based recommendation model (for visual matching).

*Co-first author.

[1] http://www.polyvore.com/
[2] http://www.chictopia.com/

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313614

Recently, neural networks have been applied to address the challenges of fashion recommendation: Song et al. [41] use a pre-trained Convolutional Neural Network (CNN) (on ImageNet) to extract visual features (for visual understanding). Then, they employ a separate Bayesian Personalized Ranking (BPR) [35] method to exploit pairwise preferences between tops and bottoms (for visual matching). Lin et al. [28] propose to train feature extraction (for visual understanding) and preference prediction (for visual matching) in a single back-propagation scheme. They introduce a mutual attention mechanism into CNN to improve feature extraction. The visual features captured by these methods only describe basic characteristics (e.g., color, texture, shape) of the input items, which lack aesthetic characteristics (e.g., style, design) to describe the output items (to be recommended). Visual understanding and matching are conducted based on recommendation loss alone, where the supervision signal is just whether two given items are matched or not and no supervision is available to directly connect the visual signals of the fashion items. Recently, some studies have realized the importance of modeling aesthetic information. For example, Ma et al. [30] build a universal taxonomy to quantitatively describe aesthetic characteristics of clothing. Yu et al. [46] propose to encode aesthetic information by pre-training models on aesthetic assessment datasets. However, none of them is for outfit recommendation and none improves visual understanding and matching like we do.

In this paper, we address the challenges of outfit recommendation from a novel perspective by proposing a neural co-supervision learning framework, called *FAshion Recommendation Machine* (FARM). FARM enhances visual understanding and visual matching with the *joint supervision* of generation and recommendation learning. Let us explain. By incorporating the generation process as a supervision signal, FARM is able to encode more aesthetic characteristics, based on which we can directly generate the output items. FARM enhances visual matching by incorporating a novel layer-to-layer matching mechanism to evaluate the matching score of generated and candidate items at different neural layers; in this manner FARM fuses the generation features from different visual levels to improve the recommendation performance. This layer-to-layer matching mechanism also ensures that FARM avoids paying too much attention to the generation quality and ignoring the recommendation performance. To the best of our knowledge, FARM is the first end-to-end learning framework that improves outfit recommendation with joint modeling of fashion generation.

Extensive experimental results conducted on two publicly available datasets show that FARM outperforms state-of-the-art models on outfit recommendation, in terms of AUC and MRR. To further demonstrate the advantages of FARM, we conduct several analyses and case studies.

To sum up, our contributions can be summarized as follows:

- We propose a neural co-supervision learning framework, FARM, for outfit recommendation that simultaneously yields recommendation and generation.
- We propose a layer-to-layer matching mechanism that acts as a bridge between generation and recommendation, and improves recommendation by leveraging generation features.
- Our proposed approach is shown to be effective in experiments on two large-scale datasets.

2 RELATED WORK

We survey related work on fashion recommendation by focusing on the two main challenges in the area: visual understanding and visual matching.

2.1 Visual understanding

One branch of studies aims at extracting better features to improve the visual understanding of fashion items.

For instance, Iwata et al. [15] propose a recommender system for clothing coordinates using full-body photographs from fashion magazines. They extract visual features, such as color, texture and local descriptors such as SIFT, and use a probabilistic topic model for visual understanding of coordinates from these features. Liu et al. [29] target occasion-oriented clothing recommendation. Given a user-input event, e.g., wedding, shopping or dating, their model recommends the most suitable clothing from the user's own clothing photo album. They adopt clothing attributes (e.g., clothing category, color, pattern) for better visual understanding. Jagadeesh et al. [16] describe a visual recommendation system for street fashion images. They mainly focus on color modeling in terms of visual understanding.

The studies listed above achieve visual understanding mostly based on feature engineering and conventional machine learning techniques. With the availability of large scale fashion recommendation datasets and the rapid development of deep learning models, several recent publications turn to neural networks for fashion recommendation. CNNs are certainly widely employed [26, 31]. Ma et al. [30] build a taxonomy based on a theory of aesthetics to describe aesthetic features of fashion items quantitatively and universally. Then they capture the internal correlation in clothing collocations by a novel fashion-oriented multi-modal deep learning based model. Song et al. [41] use a pre-trained CNN on ImageNet to extract visual features. Then, to improve visual understanding with contextual information (such as titles and categories), they propose to use multi-modal auto-encoders to exploit the latent compatibility of visual and contextual features. Han et al. [11] enrich visual understanding by incorporating sequential information by using a Bidirectional Long Short-Term Memory Network (Bi-LSTM) to predict the next item conditioned on previous ones. They further inject attribute and category information as a kind of regularization to learn a visual-semantic space by regressing visual features to their semantic representations. Kang et al. [20] use a CNN-F [7] to learn image representations and train a personalized fashion recommendation system jointly. Besides, they devise a personalized fashion design system based on the learned CNN-F and user representations. Yu et al. [46] propose to introduce aesthetic information into fashion recommendation. To achieve this, they extract aesthetic features using a pre-trained brain-inspired deep structure on the aesthetic assessment task. Lin et al. [28] enhance visual understanding by jointly modeling fashion recommendation and user comment generation, where the visual features learned with a CNN are enriched because they are related to the generation of user comments.

Even though there is a growing number of studies on better visual understanding for fashion recommendation, none of them takes fashion generation into account like we do in this paper.

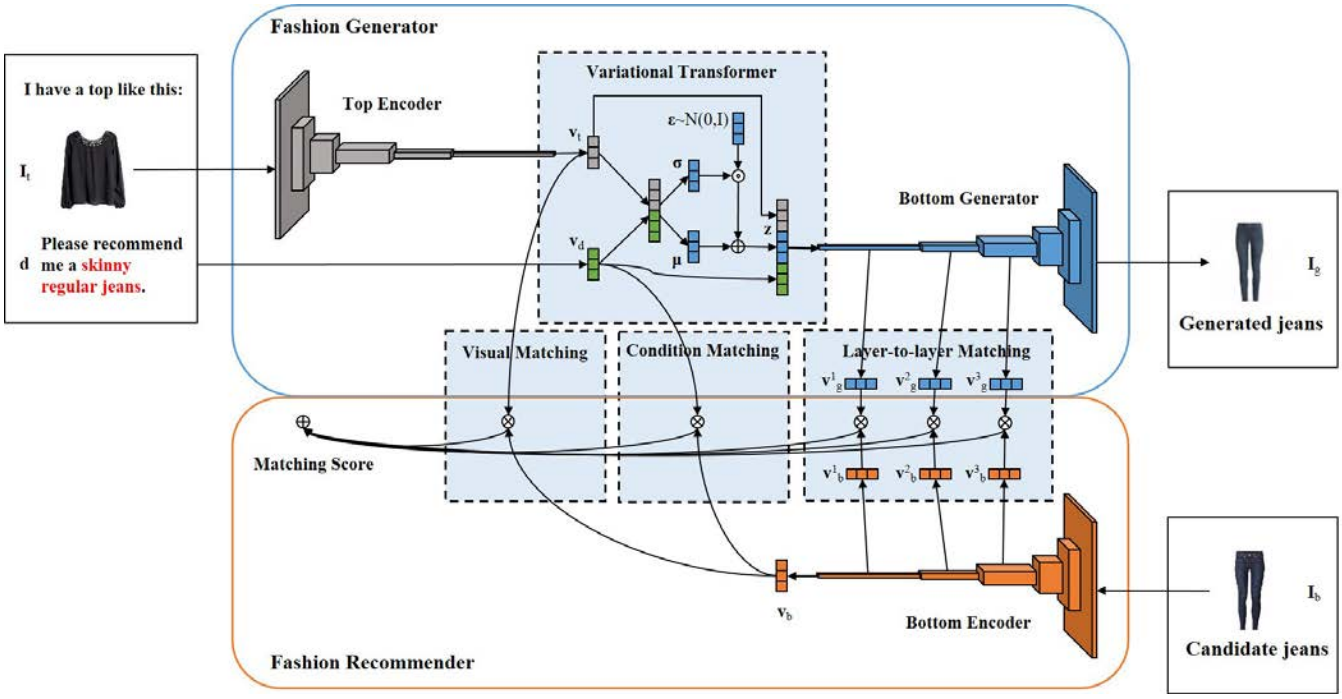


Figure 1: Overview of FARM. The fashion generator (top) uses a variational transformer to learn a special Gaussian distribution for a given top image $\mathbf{I}_t$ and a given bottom description d. It then generates a bottom image $\mathbf{I}_g$ to match $\mathbf{I}_t$ and d. The fashion recommender (bottom) evaluates the matching score between the recommended bottom image $\mathbf{I}_b$ and $(\mathbf{I}_t, \mathbf{d})$ pair from three angles, i.e., visual matching, description matching, and layer-to-layer matching.

2.2 Visual matching

Early studies into visual matching are based on conventional machine learning methods. Iwata et al. [15] use a topic model to learn the relation between photographs and recommend a bottom that has the closest topic proportions to those of the given top. Liu et al. [29] employ an SVM for recommendation, which has a term describing the relationship between visual features and attributes of tops and bottoms. Simo-Serra et al. [38] predict the popularity of an outfit to implicitly learn its compatibility by a Conditional Random Field (CRF) model. McAuley et al. [31] measure the compatibility between clothes by learning a distance metric with pretrained CNN features. Hu et al. [14] propose a functional pairwise interaction tensor factorization method to model the interactions between fashion items of different categories. Hsiao and Grauman [13] develop a submodular objective function to capture the key ingredients of visual compatibility in outfits. They propose a topic model namely Correlated Topic Models (CTM) to generate compatible outfits learned from unlabeled images of people wearing outfits.

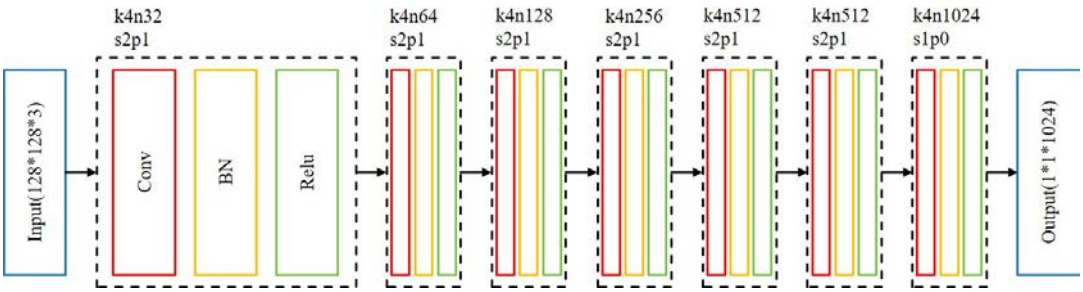
Recently, deep learning methods have been used widely in the fashion recommendation community. Veit et al. [43] train an end-to-end Siamese CNN network to learn a feature transformation from images to a latent compatibility space. Oramas and Tuytelaars [33] mine mid-level elements from CNNs to model the compatibility of clothes. Li et al. [26] use a Recurrent Neural Network (RNN) to predict whether an outfit is popular, which also implicitly learns the compatibility relation between fashion items. Han et al. [11] further train a Bi-LSTM to sequentially predict the next item conditioned on the previous ones for learning their compatibility relationship. Song et al. [41] employ a dual auto-encoder network to learn the latent compatibility space where they use the BPR model to jointly model the relation between visual and contextual modalities and implicit preferences among fashion items. Song et al. [40] consider the knowledge about clothing matching and follow a teacher-student scheme to encode the fashion domain knowledge in a traditional neural network. And they introduce an attentive scheme to the knowledge distillation procedure to flexibly assign rule confidence. Nakamura and Goto [32] present an architecture containing three subnetworks, i.e., VSE (Visual-Semantic Embedding), Bi-LSTM and SE (Style Embedding) modules, to model the matching relation between different items to generate outfits. Lin et al. [28] propose a mutual attention mechanism into CNNs to model the compatibility between different parts of images of fashion items.

Although there are many studies on improving visual matching, none of them considers connecting it with fashion generation.

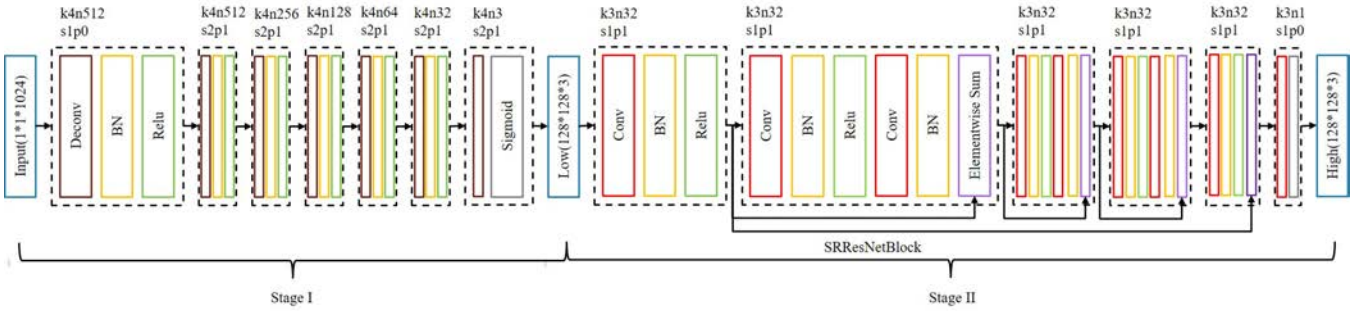
3 NEURAL FASHION RECOMMENDATION

3.1 Overview

Given a top t from a pool $\mathcal{T} = \{t_1, t_2, \ldots, t_{N_t}\}$ and a user's description d for the target bottom, the *bottom recommendation task* is to recommend a list of bottoms from a candidate pool $\mathcal{B}$ =



(a) Encoder



(b) Generator

Figure 2: Details of the encoder and the generator in FARM, where k represents kernel size, n represents the number of channels, s represents strides and p represents padding.

$\{b_1, b_2, \ldots, b_{N_b}\}$. Similarly, the *top recommendation task* is to recommend a ranked list of tops for a given bottom and top description pair. Here, we use bottom recommendation as the setup to introduce our framework FARM.

As shown in Figure 1, FARM consists of two parts, i.e., a fashion generator (for visual understanding) and a fashion recommender (for visual matching), where the fashion generator is actually an auxiliary module for recommendation. For the fashion generator, we use a CNN as the top encoder to extract the visual features from a given top image $\mathbf{I}_t$. We learn the semantic representation for the bag-of-words vector $\mathbf{d}$ of a given bottom description. Then we use a variational transformer to learn the mapping from the bottom distribution to a specific Gaussian distribution that is based on the visual features of $\mathbf{I}_t$ and the semantic representation of $\mathbf{d}$. Finally, we sample a random vector from the Gaussian distribution and input it to a DeConvolutional Neural Network (DCNN) [48] (as bottom generator) to generate a bottom image $\mathbf{I}_g$ that matches $\mathbf{I}_t$ and $\mathbf{d}$, which explicitly forces the top encoder to encode more aesthetic matching information into the visual features. For the fashion recommender, we also employ a CNN as the bottom encoder to extract the visual features from a candidate bottom image $\mathbf{I}_b$. Then we evaluate the matching score between $\mathbf{I}_b$ and $(\mathbf{I}_t, \mathbf{d})$ pair from three angles, namely the visual matching between $\mathbf{I}_b$ and $\mathbf{I}_t$, the description matching between $\mathbf{I}_b$ and $\mathbf{d}$, and the layer-to-layer matching between $\mathbf{I}_b$ and $\mathbf{I}_g$ which leverages the generation information to improve the recommendation. FARM jointly trains the fashion generator and fashion recommender. Next we will detail each of these two main parts.

3.2 Fashion generator

Given an image $\mathbf{I}_t$ of a top t and the bag-of-words vector $\mathbf{d}$ of a bottom description d, the fashion generator needs to generate a bottom image $\mathbf{I}_g$ that not only matches $\mathbf{I}_t$, but also meets $\mathbf{d}$ as much as possible. We enforce the extracted visual features from $\mathbf{I}_t$ to contain the information about its matching bottom by using the generator as a supervision signal. The generated image can be seen as a reference for recommendation.

Specifically, for a generated bottom image $\mathbf{I}_g$ that matches $\mathbf{I}_t$ and $\mathbf{d}$, the aim of the fashion generator is to maximize Eq. 1:

$$p(\mathbf{I}_g|\mathbf{I}_t, \mathbf{d}) = \int_{\mathbf{z}} p(\mathbf{I}_g|\mathbf{z}, \mathbf{I}_t, \mathbf{d}) p(\mathbf{z}|\mathbf{I}_t, \mathbf{d}) \mathrm{d}\mathbf{z}, \tag{1}$$

where $p(\mathbf{z}|\mathbf{I}_t, \mathbf{d})$ is the top encoder, $p(\mathbf{I}_g|\mathbf{z}, \mathbf{I}_t, \mathbf{d})$ is the bottom generator, and $\mathbf{z}$ is the latent variable. Because the integral of the marginal likelihood shown in Eq. 1 is intractable, inspired by variational inference [4], we first find the Evidence Lower BOund (ELBO) of $p(\mathbf{I}_g|\mathbf{I}_t, \mathbf{d})$, as shown in Eq. 2:

$$\begin{aligned} \mathrm{ELBO} = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{I}_t, \mathbf{d})}[\log p(\mathbf{I}_g|\mathbf{z}, \mathbf{I}_t, \mathbf{d})] \\ -\mathrm{KL}[q(\mathbf{z}|\mathbf{I}_t, \mathbf{d}) \| p(\mathbf{z}|\mathbf{I}_t, \mathbf{d})], \end{aligned} \tag{2}$$

where $q(\mathbf{z}|\mathbf{I}_t, \mathbf{d})$ is the approximation of the intractable true posterior $p(\mathbf{z}|\mathbf{I}_g, \mathbf{I}_t, \mathbf{d})$. The following inequality holds for the ELBO:

$$\log p(\mathbf{I}_g|\mathbf{I}_t, \mathbf{d}) \geqslant \mathrm{ELBO}. \tag{3}$$

Hence, we can maximize the ELBO so as to maximize $\log p(\mathbf{I}_g|\mathbf{I}_t, \mathbf{d})$. The ELBO contains three components: $q(\mathbf{z}|\mathbf{I}_t, \mathbf{d})$, $p(\mathbf{z}|\mathbf{I}_t, \mathbf{d})$ and $p(\mathbf{I}_g|\mathbf{z}, \mathbf{I}_t, \mathbf{d})$. Below we explain each component in detail.

3.2.1 $q(\mathbf{z}|\boldsymbol{I}_t, \boldsymbol{d})$ *and* $p(\mathbf{z}|\boldsymbol{I}_t, \boldsymbol{d})$. We propose a variational transformer (as shown in Figure 1) to model these two components, which transforms $\mathbf{I}_t, \mathbf{d}$ into a latent variable $\mathbf{z}$. As with previous work [23, 37], we assume that $q(\mathbf{z}|\mathbf{I}_t, \mathbf{d})$ and $p(\mathbf{z}|\mathbf{I}_t, \mathbf{d})$ are Gaussian distributions, i.e.,

$$q(\mathbf{z}|\mathbf{I}_t, \mathbf{d}) \sim \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2), \quad p(\mathbf{z}|\mathbf{I}_t, \mathbf{d}) \sim \mathcal{N}(0, 1), \tag{4}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ denote the variational mean and standard deviation respectively, which are calculated with our top encoder and variational transformer as follows.

Specifically, for a top image $\mathbf{I}_t$ of size 128×128 with 3 channels, we first use a CNN, i.e., *the top encoder* (as shown in Figure 2(a)) to extract visual features $\mathbf{F}_t$:

$$\mathbf{F}_t = \mathbf{CNN}(\mathbf{I}_t), \tag{5}$$

where $\mathbf{F}_t \in \mathbb{R}^{W \times H \times D}$, W and H are the width and height of the output feature maps, respectively, and D is the number of output feature maps. And we flatten $\mathbf{F}_t$ into a vector $\mathbf{f}_t \in \mathbb{R}^N$, where $N = W \times H \times D$, and project $\mathbf{f}_t$ to the visual representation $\mathbf{v}_t$:

$$\mathbf{v}_t = \text{sigmoid}(\mathbf{W}_{vt}\mathbf{f}_t + \mathbf{b}_{vt}), \tag{6}$$

where $\mathbf{W}_{vt} \in \mathbb{R}^{e \times N}$, $\mathbf{v}_t$ and $\mathbf{b}_{vt} \in \mathbb{R}^e$, and e is the size of the representation.

Besides the top image, FARM also allows users to give a natural language description $\mathbf{d}$, which describes the ideal bottom they want. In order to take into account the description $\mathbf{d}$, we follow Eq. 7 to get the semantic representation $\mathbf{v}_d$:

$$\mathbf{v}_d = \text{sigmoid}(\mathbf{W}_d\mathbf{d}), \tag{7}$$

where $\mathbf{v}_d \in \mathbb{R}^e$, $\mathbf{d} \in \mathbb{R}^{D_d}$, D_d is the vocabulary size, and $\mathbf{W}_d \in \mathbb{R}^{e \times D_d}$ is the visual semantic word embedding matrix [32], which transforms words from the textual space to the visual space. Specially, when d is an empty description, $\mathbf{v}_d$ is a zero vector.

Then *the variational transformer* uses the visual representation $\mathbf{v}_t$ and the semantic representation $\mathbf{v}_d$ to calculate the mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$ for $q(\mathbf{z}|\mathbf{I}_t, \mathbf{d})$:

$$\begin{aligned} \boldsymbol{\mu} &= \mathbf{W}_{\mu t}\mathbf{v}_t + \mathbf{W}_{\mu d}\mathbf{v}_d + \mathbf{b}_\mu \\ \log \boldsymbol{\sigma}^2 &= \mathbf{W}_{\sigma t}\mathbf{v}_t + \mathbf{W}_{\sigma d}\mathbf{v}_d + \mathbf{b}_\sigma, \end{aligned} \tag{8}$$

where $\mathbf{W}_{\mu t}$, $\mathbf{W}_{\mu d}$, $\mathbf{W}_{\sigma t}$ and $\mathbf{W}_{\sigma d} \in \mathbb{R}^{k \times e}$, $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$, $\mathbf{b}_\mu$ and $\mathbf{b}_\sigma \in \mathbb{R}^k$, and k is the size of latent variable $\mathbf{z}$. The latent variable $\mathbf{z}$ can be calculated by the reparameterization trick [23, 37]:

$$\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1), \quad \mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \otimes \boldsymbol{\epsilon}, \tag{9}$$

where $\boldsymbol{\epsilon}$ and $\mathbf{z} \in \mathbb{R}^k$, and $\boldsymbol{\epsilon}$ is the auxiliary noise variable. By the reparameterization trick, we make sure $\mathbf{z}$ is a random vector sampled from $\mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2)$.

3.2.2 $p(\boldsymbol{I}_g|\mathbf{z}, \boldsymbol{I}_t, \boldsymbol{d})$. We use the bottom generator (as shown in Figure 2(b)) to generate $\mathbf{I}_g$ from the variable $\mathbf{z}$. We also assume $p(\mathbf{I}_g|\mathbf{z}, \mathbf{I}_t, \mathbf{d})$ is a Gaussian distribution [23, 37], i.e.,

$$p(\mathbf{I}_g|\mathbf{z}, \mathbf{I}_t, \mathbf{d}) \sim \mathcal{N}(g(\mathbf{z}, \mathbf{I}_t, \mathbf{d}), \boldsymbol{\sigma}^2), \tag{10}$$

where g is the bottom generator.

Specifically, we first follow Eq. 11 to obtain the basic visual feature vector $\mathbf{f}_g$:

$$\mathbf{f}_g = \text{relu}(\mathbf{W}_{gz}\mathbf{z} + \mathbf{W}_{gt}\mathbf{v}_t + \mathbf{W}_{gd}\mathbf{v}_d + \mathbf{b}_g), \tag{11}$$

where $\mathbf{f}_g$ and $\mathbf{b}_g \in \mathbb{R}^N$, $\mathbf{W}_{gz}\mathbb{R}^{N \times k}$, $\mathbf{W}_{gt}$ and $\mathbf{W}_{gd} \in \mathbb{R}^{N \times e}$. Then we reshape $\mathbf{f}_g$ into a 3-D tensor $\mathbf{F}_g \in \mathbb{R}^{W \times H \times D}$, which is the reverse operation to what we do for $\mathbf{F}_t$. Finally, we use a DCNN, i.e., *the bottom generator* to generate the bottom image $\mathbf{I}_g$:

$$\mathbf{I}_g = \mathbf{DCNN}(\mathbf{F}_g), \tag{12}$$

where $\mathbf{I}_g \in \mathbb{R}^{128 \times 128 \times 3}$. To avoid generating blurry images [3], we divide the process of image generation into two stages [6, 49]. The first stage is an ordinary deconvolutional neural network that generates low-resolution images. The second stage is similar to the super-resolution residual network (SRResNet) [24], which accepts the images from the first stage and refines them to generate high quality ones. The DCNN is meant to capture high-level aesthetic features of the bottoms to be recommended [47, 48]. Besides, in order to generate the bottom, the generation process also forces the top encoder to capture more aesthetic information.

During training, we first sample a z from $q(\mathbf{z}|\mathbf{I}_t, \mathbf{d})$. Then we generate $\mathbf{I}_g$ with $g(\mathbf{z}, \mathbf{I}_t, \mathbf{d})$. During testing, in order to avoid the randomness introduced by $\boldsymbol{\epsilon}$, we directly generate $\mathbf{I}_g$ by $g(\mathbf{z} = \boldsymbol{\mu}, \mathbf{I}_t, \mathbf{d})$.

3.3 Fashion recommender

Given the image $\mathbf{I}_b$ of a bottom b, the fashion recommender needs to evaluate the matching score between $\mathbf{I}_b$ and the pair $(\mathbf{I}_t, \mathbf{d})$. Specifically, we first use the *bottom encoder* (as shown in Figure 2(a)), which has the same structure as the top encoder (parameters not shared), to extract visual features $\mathbf{F}_b \in \mathbb{R}^{W \times H \times D}$ from $\mathbf{I}_b$. Then we flatten $\mathbf{F}_b$ into a vector $\mathbf{f}_b \in \mathbb{R}^N$ and project $\mathbf{f}_b$ to the visual representation $\mathbf{v}_b$. Next, we calculate the matching score between $\mathbf{I}_b$ and the pair $(\mathbf{I}_t, \mathbf{d})$ in three ways.

3.3.1 Visual matching. We propose visual matching to evaluate the compatibility between $\mathbf{I}_b$ and $\mathbf{I}_t$ based on their visual features. Specifically, we calculate the visual matching score s_v between $\mathbf{I}_b$ and $\mathbf{I}_t$ by Eq. 13:

$$s_v = \mathbf{v}_b^T\mathbf{v}_t. \tag{13}$$

3.3.2 Description matching. For evaluating the matching degree between $\mathbf{I}_b$ and $\mathbf{d}$, we propose to match descriptions. The description matching score s_d between $\mathbf{I}_b$ and $\mathbf{d}$ is calculated by Eq. 14:

$$s_d = \mathbf{v}_b^T\mathbf{v}_d. \tag{14}$$

Note that if d does not contain any word, s_d equals 0.

3.3.3 Layer-to-layer matching. As we will demonstrate in our experiments in Section 6.2, a simple combination of generation and recommendation is not able to improve the recommendation performance. The reason is that there is no direct connection between generation and recommendation, which results in two issues. First, the aesthetic information from the generation process cannot be used effectively. Second, the generation process might introduce features that are only helpful for generation while unhelpful for recommendation. To overcome these issues, we propose a layer-to-layer matching mechanism. Specifically, we denote the visual features of the l-th CNN layer in the bottom encoder as $\mathbf{F}_b^l \in \mathbb{R}^{W^l \times H^l \times D^l}$. And we denote the visual features of the corresponding DCNN layer, which has the same size as $\mathbf{F}_b^l$, in the bottom generator as

$\mathbf{F}_g^l \in \mathbb{R}^{W^l \times H^l \times D^l}$. Then, we reshape $\mathbf{F}_b^l = [\mathbf{f}_{b,1}^l, \ldots, \mathbf{f}_{b,S}^l]$ by flattening the width and height of the original $\mathbf{F}_b^l$, where $S = W^l \times H^l$ and $\mathbf{f}_{b,i}^l \in \mathbb{R}^{D^l}$. And we can consider $\mathbf{f}_{b,i}^l$ as the visual features of the i-th location of $\mathbf{I}_b$. We perform global-average-pooling in $\mathbf{F}_b^l$ to get the global visual features $\mathbf{f}_b^l \in \mathbb{R}^{D^l}$:

$$\mathbf{f}_b^l = \frac{1}{S} \sum_{i=1}^{S} \mathbf{f}_{b,i}^l. \tag{15}$$

We project $\mathbf{f}_b^l$ to the visual representation $\mathbf{v}_b^l \in \mathbb{R}^e$:

$$\mathbf{v}_b^l = \text{sigmoid}(\mathbf{W}_{vb}^l \mathbf{f}_b^l + \mathbf{b}_{vb}^l), \tag{16}$$

where $\mathbf{W}_{vb}^l \in \mathbb{R}^{e \times D^l}$ and $\mathbf{b}_{vb}^l \in \mathbb{R}^e$. The same operations apply to $\mathbf{F}_g^l$ to get $\mathbf{v}_g^l$. Then we calculate the dot product between $\mathbf{v}_b^l$ and $\mathbf{v}_g^l$, which represents the matching degree s_g^l between $\mathbf{I}_b$ and $\mathbf{I}_g$ in the l-th visual level:

$$s_g^l = {\mathbf{v}_b^l}^T \mathbf{v}_g^l. \tag{17}$$

For different visual levels, we sum all s_g^l to get the matching score s_g between $\mathbf{I}_b$ and $\mathbf{I}_g$:

$$s_g = \sum_{l \in L} s_g^l, \tag{18}$$

where L is the selected CNN layer set for layer-to-layer matching.

Finally, the total matching score s between $\mathbf{I}_b$ and the pair $(\mathbf{I}_t, \mathbf{d})$ is defined as follows:

$$s = s_v + s_d + s_g. \tag{19}$$

3.4 Co-supervision learning framework

For FARM, we train the fashion generator and the fashion recommender jointly with a co-supervision learning framework.

Specifically, for the generation part, we regard the image $\mathbf{I}_p$ of a positive bottom p, which not only matches the given top $\mathbf{I}_t$ but also meets the given description $\mathbf{d}$, as the generation target. And we denote the generated bottom image in the first stage as $\mathbf{I}_g^1$, and denote the generated bottom image in the second stage as $\mathbf{I}_g^2$. Then, the first loss is to maximize the first term in ELBO, which is Eq. 20:

$$\mathcal{L}_{gen}(t, d, p) = \frac{1}{2} \|\mathbf{I}_g^1 - \mathbf{I}_p\|_2^2 + \|\mathbf{I}_g^2 - \mathbf{I}_p\|. \tag{20}$$

The second loss is to minimize the second term in ELBO, which is Eq. 21:

$$\mathcal{L}_{kl}(t, d, p) = \frac{1}{2} \sum_{i=1}^{k} (1 + \log \boldsymbol{\sigma}_i^2 - \boldsymbol{\mu}_i^2 - \boldsymbol{\sigma}_i^2), \tag{21}$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$ are the i-th elements in $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ respectively.

For the recommendation part, we employ BPR [35] as the loss:

$$\mathcal{L}_{bpr}(t, d, p, n) = -\log(\text{sigmoid}(s_p - s_n)), \tag{22}$$

where s_p and s_n are the matching scores of a positive bottom $\mathbf{I}_p$ and a negative bottom $\mathbf{I}_n$, respectively (calculated with Eq. 19). $\mathbf{I}_n$ (image of bottom n) is randomly sampled.

The total loss function can be defined as follows:

$$\mathcal{L} = \sum_{(t,d,p,n) \in \mathcal{D}} \mathcal{L}_{gen}(t, d, p) + \mathcal{L}_{kl}(t, d, p) + \mathcal{L}_{bpr}(t, d, p, n), \tag{23}$$

where $\mathcal{D} = \{(t, d, p, n) | t \in \mathcal{T}, d \in \mathcal{D}_b, p \in \mathcal{B}_{t,d}, n \in \mathcal{B} \setminus \mathcal{B}_{t,d}\}$, $\mathcal{D}_b$ is the bottom description set, $\mathcal{B}_{t,d}$ is the positive bottom set for the pair $(\mathbf{I}_t, \mathbf{d})$ and $\mathcal{B} \setminus \mathcal{B}_{t,d}$ is the negative bottom set for the pair $(\mathbf{I}_t, \mathbf{d})$. The whole framework can be efficiently trained using back-propagation in an end-to-end paradigm.

For top recommendation, we follow the same way to build and train the model, but exchange the roles of tops and bottoms.

4 EXPERIMENTAL SETUP

We set up a series of experiments to evaluate the recommendation performance of FARM. Details of our experimental settings are listed below. All code and data used to run the experiments in this paper are available at https://bitbucket.org/Jay_Ren/www2019_fashionrecommendation_yujie/src/master/farm/.

4.1 Datasets

Existing fashion datasets include *WoW* [29], *Exact Street2Shop* [21], *Fashion-136K* [16], *FashionVC* [41] and *ExpFashion* [28]. *WoW*, *Exact Street2Shop*, and *Fashion-136K* have been collected from street photos[3] on the web and involve (visual) parsing of clothing, which still remains a great challenge in the computer vision domain [41, 44, 45] and which is beyond the scope of this paper. *FashionVC* and *ExpFashion* have been collected from the fashion-oriented online community Polyvore[4] and contain both images and texts. The images are of good quality and the texts include descriptions like names and categories. For our experiments, we choose *FashionVC* and *ExpFashion*. The statistics of the two datasets are given in Table 1. We preprocess *FashionVC* or *ExpFashion* with the following steps, taking bottom recommendation as an example. For each tuple (*top*, *top description*, *bottom*, *bottom description*), we regard (*top*, *bottom description*) as input and the *bottom* as the ground truth output. We follow existing studies [41] and randomly select bottoms to generate 100 candidates along with the ground truth bottoms in the validation and test set. Similar processing steps are used for top recommendation.

Table 1: Dataset statistics.

Dataset	Tops	Bottoms	Outfits
FashionVC [41]	14,871	13,663	20,726
ExpFashion [28]	168,682	117,668	853,991

4.2 Implementation details

The parameters W, H, D and N of the encoder and the generator are set to 1, 1, 1024 and 1024, respectively. The size e of the visual semantic word embedding, the semantic representation and the visual representation is set to 100. And the latent variable size k is set to 100 too. The 7th, the 6th and the 5th layers of the encoder CNN are adopted to compute the layer-to-layer matching with the input, the 1st and the 2nd layers of the generator DCNN. To build descriptions, we first filter out words whose frequency is less than 100. Then, we manually go through the rest to only keep words that can describe tops or bottoms. Finally, the remaining vocabulary

[3] http://www.tamaraberg.com/street2shop/
[4] http://www.polyvore.com/

size D_d is 547. During training, we initialize model parameters randomly with the Xavier method [10]. We choose Adam [22] as our optimization algorithm. For the hyper-parameters of the Adam optimizer, we set the learning rate $\alpha = 0.001$, two momentum parameters $\beta 1 = 0.9$ and $\beta 2 = 0.999$, and $\epsilon = 10^{-8}$. We apply dropout [42] to the output of our encoder and set the rate to 0.5. We also apply gradient clipping [34] with range $[-5, 5]$ during training. We use a mini-batch size 64 by grid search to both speed up the training and converge quickly. We test the model performance on the validation set for every epoch. Our framework is implemented with MXNet [8]. All experiments are conducted on a single Titan X GPU.

4.3 Methods used for comparison

We choose the following methods for comparison.

- *LR*: Logistic Regression (LR) is a standard machine learning method [17]. We use it to predict whether a candidate bottom matches a given (*top*, *bottom description*) pair or not. Specifically, we employ a pre-trained CNN to extract visual features from images. Then we follow Eq. 24 to calculate the matching probability p:

$$p = \text{sigmoid}(\mathbf{w}_t^T \mathbf{v}_t + \mathbf{w}_b^T \mathbf{v}_b + \mathbf{w}_d^T \mathbf{d}), \tag{24}$$

  where $\mathbf{v}_t$ and $\mathbf{v}_b \in \mathbb{R}^{D_v}$ are the visual features of the top and the bottom respectively, $\mathbf{w}_t$ and $\mathbf{w}_b \in \mathbb{R}^{D_v}$, and $\mathbf{w}_d \in \mathbb{R}^{D_d}$. D_v is set to 4096 in our experiments.
- *IBR$_d$*: IBR [31] learns a visual style space in which related objects are close and unrelated objects are far. In order to consider the given descriptions at the same time, we modify IBR by projecting descriptions to the visual style space. As a result, we can evaluate the matching degree between objects and descriptions by their distance in the space. Specifically, the distance function between the candidate bottom b and the given (*top*, *bottom description*) pair (t, d) is as follows:

$$m_{tdb} = \|\mathbf{W}_v \mathbf{v}_t - \mathbf{W}_v \mathbf{v}_b\|_2^2 + \|\mathbf{W}_d \mathbf{v}_d - \mathbf{W}_v \mathbf{d}\|_2^2, \tag{25}$$

  where $\mathbf{W}_v \in \mathbb{R}^{K \times D_v}$, $\mathbf{W}_d \in \mathbb{R}^{K \times D_d}$, $\mathbf{v}_t$ and $\mathbf{v}_b \in \mathbb{R}^{D_v}$ are the visual features extracted by a pre-trained CNN, and K is the dimension of the visual style space. D_v is 4096, and K is 100 in our experiments. We refer to the modified version as IBR$_d$.
- *BPR-DAE$_d$*: BPR-DAE [41] can jointly model the implicit matching preference between items in visual and textual modalities and the coherence relation between different modalities of items. In our task, we do not have other text information except descriptions, so we first remove the part of BPR-DAE that is related to text information. Then, for evaluating the matching score between the given description and the candidate item, we project the description representation and the item representation to the same latent space:

$$\mathbf{v}'_d = \text{sigmoid}(\mathbf{W}_d \mathbf{d}), \quad \mathbf{v}'_i = \text{sigmoid}(\mathbf{W}_v \mathbf{v}_i), \tag{26}$$

  where $\mathbf{W}_d \in \mathbb{R}^{K \times D_d}$, $\mathbf{W}_v \in \mathbb{R}^{K \times D_v}$, and $\mathbf{v}_i \in \mathbb{R}^{D_v}$ is the latent representation of item i learned by BPR-DAE. Finally, we follow Eq. 27 to evaluate the compatibility between a candidate bottom b and a given (*top*, *bottom description*) pair (t, d):

$$m_{tdb} = \mathbf{v}_t^T \mathbf{v}_b + \mathbf{v}'^T_d \mathbf{v}'_b. \tag{27}$$

  We set $D_v = 512$, and $K = 100$ in experiments. We refer to the modified version as BPR-DAE$_d$.
- *DVBPR$_d$*: DVBPR [20] learns the image representations and trains the recommender system jointly to recommend fashion items for users. We adopt DVBPR to our task and refer to it as DVBPR$_d$. Specifically, we first follow DVBPR to use a CNN-F to learn image representations of tops and bottoms. Then we calculate the matching score between a bottom and the given (*top*, *bottom description*) pair by Eq. 28:

$$m_{tdb} = \mathbf{v}_t^T \mathbf{v}_b + \mathbf{v}_d^T \mathbf{v}_b, \tag{28}$$

  where $\mathbf{v}_t$ and $\mathbf{v}_b \in \mathbb{R}^K$ are the image representations of the top and bottom respectively, $\mathbf{v}_d \in \mathbb{R}^K$ is the description representation learned in the same way as FARM, and K is set to 100 in experiments.

4.4 Evaluation metrics

We employ *Mean Reciprocal Rank* (MRR) and *Area Under the ROC Curve* (AUC) to evaluate the recommendation performance, which are widely used in recommender systems [25, 36, 50].

In the case of bottom recommendations, for example, MRR and AUC are calculated as follows:

$$\text{MRR} = \frac{1}{|Q^{td}|} \sum_{i=1}^{|Q^{td}|} \frac{1}{rank_i}, \tag{29}$$

where Q^{td} is the (*top*, *bottom description*) collection as queries, and $rank_i$ refers to the rank position of the first positive bottom for the i-th (*top*, *bottom description*) pair. Furthermore,

$$\text{AUC} = \frac{1}{|Q^{td}|} \sum_{(t,d) \in Q^{td}} \frac{1}{|E(t,d)|} \sum_{(p,n) \in E(t,d)} \delta(s_p > s_n), \tag{30}$$

where $E(t, d)$ is the set of all positive and negative candidate bottoms for the given top t and the given bottom description d, s_p is the matching score of a positive bottom p, s_n is the matching score of a negative bottom n, and $\delta(\alpha)$ is an indicator function that equals 1 if α is true and 0 otherwise.

5 RESULTS

The recommendation results on the FashionVC and ExpFashion datasets of FARM and the methods used for comparison are shown in Table 2. We can see that FARM consistently outperforms all baselines in terms of AUC and MRR on both datasets. We have five main observations from Table 2.

(1) FARM significantly outperforms all baselines and achieves the best results on all metrics. There are three main reasons. First, FARM contains a fashion generator as an auxiliary module for recommendation. With its co-supervision learning framework, FARM can encode more aesthetic characteristics and use this extra information to improve recommendation performance; see Section 6.1 for further analysis. Second, we propose a layer-to-layer matching scheme to make sure that FARM can effectively use the aesthetic features in the fashion generator to improve recommendation results; see Section 6.2 for a further analysis. Third, LR, IBR$_d$ and BPR-DAE$_d$ employ pre-trained CNNs (all

Table 2: Recommendation results on the FashionVC and ExpFashion datasets (%).

Method	FashionVC			
	Top		Bottom	
	AUC	MRR	AUC	MRR
LR	48.7	4.5	46.4	4.4
IBR_d	52.8	6.1	62.9	10.3
BPR-DAE_d	62.9	8.6	70.2	10.9
$DVBPR_d$	64.6	9.1	76.9	13.0
FARM	**71.2***	**12.6***	**77.8**	**15.3***
Method	**ExpFashion**			
	Top		Bottom	
	AUC	MRR	AUC	MRR
LR	50.5	5.4	48.4	4.4
IBR_d	56.1	7.1	68.9	12.0
BPR-DAE_d	73.0	12.3	79.9	14.7
$DVBPR_d$	82.4	18.5	83.7	15.4
FARM	**85.2***	**25.1***	**88.4***	**24.3***

The superscript * indicates that FARM significantly outperforms $DVBPR_d$, using a paired t-test with $p < 0.05$.

AlexNet [19] trained on ImageNet[5]) to extract visual features from images, but they do not fine-tune the CNNs during experiments. However, in FARM, we jointly train the top encoder, the bottom encoder and the top/bottom generator, which can extract better visual features.

(2) $DVBPR_d$ performs better than other baseline methods. The reason is that $DVBPR_d$ employs a CNN-F to jointly learn image representations during recommendation. Hence, it can extract more effective visual features to improve recommendation performance.

(3) Although BPR-DAE_d, IBR_d and LR all use visual features extracted by a pre-trained CNN as input, BPR-DAE_d performs much better than the other two. This is because BPR-DAE_d learns a more sophisticated latent space using an auto-encoder neural network to represent the fashion items. However, IBR_d only applies a linear transformation to inputs, which restricts the expressive ability of the visual style space. And LR directly uses the visual features and the bag-of-words vectors as inputs, making it hard to learn an effective matching relation.

(4) The performance of all methods on the ExpFashion dataset is better than on the FashionVC dataset. The most important reason is that the average length of the descriptions in the ExpFashion dataset is 5.6 words, however, it is only 3.7 words in the FashionVC dataset. That means that the descriptions in the ExpFashion dataset contain more details that can provide more information for recommendation and generation, which boosts the recommendation performance.

(5) The bottom recommendation performance is better than the top recommendation performance for most methods. The number of tops is larger than the number of bottoms and the styles of tops are also richer than those of bottoms on both datasets. That makes bottom recommendation and bottom generation easier.

[5]http://www.image-net.org/

Table 3: Analysis of co-supervision learning. Recommendation results on the FashionVC and ExpFashion datasets (%).

Method	FashionVC			
	Top		Bottom	
	AUC	MRR	AUC	MRR
FARM-G	54.8	8.4	60.9	9.8
FARM-R	68.0	9.8	77.2	12.8
FARM	**71.2***	**12.6***	**77.8**	**15.3***
Method	**ExpFashion**			
	Top		Bottom	
	AUC	MRR	AUC	MRR
FARM-G	64.4	14.2	72.4	21.3
FARM-R	82.3	18.9	84.2	15.2
FARM	**85.2***	**25.1***	**88.4***	**24.3***

The superscript * indicates that FARM significantly outperforms FARM-R, using a paired t-test with $p < 0.05$.

In summary, FARM significantly outperforms state-of-the-art methods on both datasets. The improvements mainly come from the co-supervision of generation and the layer-to-layer mechanism, which we will demonstrate in the next section.

6 ANALYSIS

We provide two types of analyses (concerning co-supervision learning and layer-to-layer matching) and two cases studies (recommendation and generation).

6.1 Co-supervision learning

To demonstrate the superiority of incorporating the extra supervision of the generator, we compare FARM with FARM-G and FARM-R, where FARM-G is FARM without the recommendation part and FARM-R is FARM without the generation part. The results are shown in Table 3. To be able to apply FARM-G to the recommendation task, we first use FARM-G to generate a bottom image for a given (*top*, *bottom description*) pair. Then, similar to [2, 27], we use a pre-trained AlexNet to get the representations of the generated bottom and the candidate bottoms. Finally, we compute the similarity between the generated bottom and a candidate bottom based on their representations.

From Table 3, we can see that FARM achieves significant improvements over FARM-R. On the FashionVC dataset, for top recommendation, AUC increases by 3.2%, MRR increases by 2.8%, and for bottom recommendation, AUC increases by 0.6%, MRR increases by 2.5%. On the ExpFashion dataset, for top recommendation, AUC increases by 2.9%, MRR increases by 6.2%, and for bottom recommendation, AUC increases by 4.2%, MRR increases by 9.1%. Thus, FARM is able to improve recommendation performance by using the generator as a supervision signal.

Comparing FARM-G with all baselines, we notice that FARM-G achieves better performance, and especially it performs better than IBR_d in most settings. Hence, the images generated by FARM-G and FARM reflect some key factors of the items to be recommended, which is why the generator can help improve recommendation.

Table 4: Analysis of layer-to-layer matching. Recommendation results on the FashionVC and ExpFashion datasets (%).

Method	FashionVC			
	Top		Bottom	
	AUC	MRR	AUC	MRR
FARM-WL	59.8	7.6	67.8	8.2
FARM	**71.2***	**12.6***	**77.8***	**15.3***
Method	ExpFashion			
	Top		Bottom	
	AUC	MRR	AUC	MRR
FARM-WL	68.6	9.9	74.3	10.3
FARM	**85.2***	**25.1***	**88.4***	**24.3***

The superscript * indicates that FARM significantly outperforms FARM-WL, using a paired t-test with $p < 0.05$.

Additionally, we find that FARM-R outperforms LR, IBR$_d$ and BPR-DAE$_d$. And it achieves comparable performance with DVBPR$_d$, whose difference against FARM-R is mainly in the CNN part. If FARM employs more powerful CNN architectures such as VGG [39] or ResNet [12], it should perform even better.

6.2 Layer-to-layer matching

To analyze the effect of the layer-to-layer matching scheme, we compare FARM with FARM-WL which only uses the visual matching and the description matching to evaluate the matching degree. We can see from Table 4 that FARM performs significantly better than FARM-WL according to all metrics on both datasets, which confirms that layer-to-layer matching does indeed improve the performance of recommendation.

To help understand the effect of layer-to-layer matching, we list some real and generated images in Figure 3. FARM generates good quality images that are similar to real images. This means that the generated images can tell us what kind of bottoms can match the given (*top*, *bottom description*) pair from the perspective of generation, so layer-to-layer matching can direct the recommender by evaluating the matching degree between the candidate images and the generated images. That is why layer-to-layer matching is able to improve the performance of recommendation.

Additionally, we notice that FARM-WL performs worse than FARM-R, which means that a simple combination of recommendation and generation is not able to improve recommendation performance significantly. This may be because, without layer-to-layer matching, FARM-WL pays too much attention to the generation quality and ignores recommendation performance. We are able to improve this situation with layer-to-layer matching. Layer-to-layer matching builds a connection between the bottom generator and the bottom encoder in different layers. As a result, the bottom encoder pushes the bottom generator to learn useful matching information for improving recommendation performance.

6.3 Recommendation case studies

We list some recommendation produced by FARM in Figure 4. For each input, we list the top-10 recommended items. We highlight



(a) Top generation.

(b) Bottom generation.

Figure 3: Comparison between real and generated images.

the positive items with red boxes. We can see that most recommended items not only match the given items, but also meet the given descriptions. For example, in the second case of the top recommendation, the given top description is "sleeve black blazer outerwear jackets,"so most recommended tops are jackets, and especially almost all recommended tops are black. Also in the first case of the bottom recommendation, the given bottom description is "distressed straight leg jean," so the recommended bottoms are all jeans, most of which are straight leg and some are distressed. By comparing the generated items with the recommended items, such as in the first case of the top recommendation and the second case of the bottom recommendation, we can see that the generated images are able to provide good guidance for the recommendation.

We also notice that not all recommended items meet the given description, mostly because FARM recommends items not only based on the given description, but also based on the given item. For example, in the third case of the bottom recommendation, the sixth recommended bottom is a denim jeans instead of a daydress. The given top is a denim coat, which makes FARM believe that recommending a denim jeans is also reasonable. Besides, not all positive items are ranked in the first position. See, e.g., the third case of the top recommendation., where the top recommended item and the given bottom have the same color green, which looks more compatible. In these failure cases, the quality of the generated images is poor so they are likely less helpful for recommendation.

6.4 Generation case studies

Although this paper focuses on improving recommendation by incorporating generation, we also list some generation cases in Figure 5. Overall, the generated items are able to match the given input.



(a) Top recommendation.



(b) Bottom recommendation.

Figure 4: Case studies of recommendation. The items highlighted in the red boxes are the positive ones.

For example, in the sixth case of the top generation, the generated navy blouse with the yellow keen length skirt looks beautiful and elegant. Although there are many kinds of navy blouses like sailor suits, the style of the generated top seems to be more suitable for the given bottom. And in the eighth case of the bottom generation, the given description does not give the specific pattern of the generated bottom.

But the generated bottom has a flame-like pattern, which makes it more compatible with the bright yellow camisole. From these samples we can see that FARM is able to generate fashion items based on the relation between the visual features of different fashion items.



(a) Top generation.



(b) Bottom generation.

Figure 5: Case studies of generation. Each case is in the form: "given description + given item = generated item".

The generated items can accord with the given descriptions no matter what they are about. For example, in the second case of the top generation, the description is "grey wool coats," so the generated top is a grey coat which also looks like wool. And in the fourth case, the description is "gold fur trim puffer jackets", so the generated jacket has fur in its collar and cuff. In the bottom generation, we also observe that FARM is able to distinguish between skinny jeans and bootcut jeans from the first and the second cases. Another example is the sixth case, where the description contains "floral print." FARM generates a black long pencil skirt with flower pattern. In short, FARM is able to build a cross-modal connection between text and images in order to generate fashion items.

Generation is a challenging process, which means that powerful features are needed in order to generate a matching item. We can see from the examples provided that FARM is able to generate aesthetically matching outfits. FARM is able to improve recommendation performance through jointly modeling generation.

7 CONCLUSION

In this paper, we have studied the task of outfit recommendation, which has two main challenges: visual understanding and visual matching. To tackle these challenges, we propose a co-supervision learning framework, namely FARM. For visual understanding, FARM captures aesthetic characteristics with the supervision of generation learning. For visual matching, FARM incorporates a layer-to-layer matching mechanism to evaluate the matching score at different neural layers.

We have conducted experiments to confirm the effectiveness of FARM. It achieves significant improvements over state-of-the-art baselines in terms of AUC and MRR. We also show that the proposed layer-to-layer matching mechanism can make effective use of generation information to improve recommendation performance. We further exhibit some cases to analyze the performance of FARM.

Our results can be used to improve users' experience in fashion-oriented online communities by providing better recommendation and to promote the research into fashion generation by demonstrating a novel application in outfit recommendation.

A limitation of FARM is that its recommendation performance is affected by the quality of the generated images. If the quality of the generated images is not high, the generation part cannot provide effective guidance for the recommendation part.

As to future work, we plan to improve the recommendation and the generation of FARM when the descriptions are lacking. And we want to extend FARM to recommend and generate whole outfits that not only contain tops and bottoms but also include shoes and hats, etc. We will also try more powerful CNN and DCNN architectures for recommendation and generation.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful comments.

This work is supported by the Natural Science Foundation of China (61672324, 61672322), the Natural Science Foundation of Shandong province (2016ZRE27468), the Fundamental Research Funds of Shandong University, Ahold Delhaize, the Association of Universities in the Netherlands, and the Innovation Center for Artificial Intelligence (ICAI). All content represents the opinion of

the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. 2006. Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Trans on Pattern Analysis and Machine Intelligence (TPAMI)* 28, 12 (2006).

[2] Artem Babenko, Anton Slesarev, Alexander Chigorin, and Victor S. Lempitsky. 2014. Neural Codes for Image Retrieval. In *European Conf. on Computer Vision (ECCV'14)*.

[3] Jianmin Bao, Dong Chen, Fang Wen, Houqiang Li, and Gang Hua. 2017. CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training. In *International Conf. on Computer Vision (ICCV'17)*. 2764–2773.

[4] David M. Blei, Alp Kucukelbir, and Jon D. Mcauliffe. 2017. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association (JASA)* 112, 518 (2017).

[5] Lubomir D. Bourdev, Subhransu Maji, and Jitendra Malik. 2011. Describing people: A Poselet-based Approach to Attribute Classification. In *International Conf. on Computer Vision (ICCV'11)*. 1543–1550.

[6] Lei Cai, Hongyang Gao, and Shuiwang Ji. 2017. Multi-Stage Variational Auto-Encoders for Coarse-to-Fine Image Generation. *CoRR* abs/1705.07202 (2017). http://arxiv.org/abs/1705.07202

[7] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Return of the Devil in the Details: Delving Deep into Convolutional Nets. In *British Machine Vision Conf. (BMVC'14)*.

[8] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. 2015. MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. In *Annual Conf. on Neural Information Processing Systems (NIPS'15)*.

[9] Navneet Dalal and Bill Triggs. 2005. Histograms of Oriented Gradients for Human Detection. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'05)*. 886–893.

[10] Xavier Glorot and Yoshua Bengio. 2010. Understanding the Difficulty of Training Deep Feedforward Neural Networks. *Journal of Machine Learning Research (JMLR)* 9 (2010), 249–256.

[11] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S. Davis. 2017. Learning Fashion Compatibility with Bidirectional LSTMs. In *ACM International Conf. on Multimedia (MM'17)*. 1078–1086.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'16)*.

[13] Wei Lin Hsiao and Kristen Grauman. 2018. Creating Capsule Wardrobes from Fashion Images. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'18)*.

[14] Yang Hu, Xi Yi, and Larry S. Davis. 2015. Collaborative Fashion Recommendation: A Functional Tensor Factorization Approach. In *ACM International Conf. on Multimedia (MM'15)*. 129–138.

[15] Tomoharu Iwata, Shinji Watanabe, and Hiroshi Sawada. 2011. Fashion Coordinates Recommender System Using Photographs from Fashion Magazines. In *International Joint Conf. on Artificial Intelligence (IJCAI'11)*. 2262–2267.

[16] Vignesh Jagadeesh, Robinson Piramuthu, Anurag Bhardwaj, Wei Di, and Neel Sundaresan. 2014. Large Scale Visual Recommendations from Street Fashion Images. In *ACM Knowledge Discovery and Data Mining (KDD'14)*. 1925–1934.

[17] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An Introduction to Statistical Learning*. Springer.

[18] Shatha Jaradat. 2017. Deep Cross-Domain Fashion Recommendation. In *ACM Conf. on Recommender Systems (RecSys'17)*. 407–410.

[19] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. In *ACM International Conf. on Multimedia (MM'14)*. 675–678.

[20] Wang-Cheng Kang, Chen Fang, Zhaowen Wang, and Julian McAuley. 2017. Visually-Aware Fashion Recommendation and Design with Generative Image Models. In *International Conf. on Data Mining (ICDM'17)*. 207–216.

[21] M. Hadi Kiapour, Xufeng Han, Svetlana Lazebnik, Alexander C. Berg, and Tamara L. Berg. 2015. Where to Buy It: Matching Street Clothing Photos in Online Shops. In *International Conf. on Computer Vision (ICCV'15)*. 3343–3351.

[22] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conf. on Learning Representations (ICLR'15)*. http://arxiv.org/abs/1412.6980

[23] Diederik P. Kingma and Max Welling. 2014. Auto-encoding Variational Bayes. In *International Conf. on Learning Representations (ICLR'14)*.

[24] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, and Zehan Wang. 2017. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'17)*. 105–114.

[25] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *ACM International Conf. on Information and Knowledge Management (CIKM'17)*. 1419–1428.

[26] Yuncheng Li, Liangliang Cao, Jiang Zhu, and Jiebo Luo. 2017. Mining Fashion Outfit Composition Using an End-to-End Deep Learning Approach on Set Data. *IEEE Transactions on Multimedia (TMM)* 19, 8 (2017), 1946–1955.

[27] Kevin Lin, Huei Fang Yang, Jen Hao Hsiao, and Chu Song Chen. 2015. Deep Learning of Binary Hash Codes for Fast Image Retrieval. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'15)*. 27–35.

[28] Yujie Lin, Pengjie. Ren, Zhumin Chen, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2018. Explainable Fashion Recommendation with Joint Outfit Matching and Comment Generation. *CoRR* abs/1806.08977 (2018).

[29] Si Liu, Jiashi Feng, Zheng Song, Tianzhu Zhang, Hanqing Lu, Changsheng Xu, and Shuicheng Yan. 2012. Hi, Magic Closet, Tell Me What to Wear!. In *ACM International Conf. on Multimedia (MM'12)*. 619–628.

[30] Yihui Ma, Jia Jia, Suping Zhou, Jingtian Fu, Yejun Liu, and Zijian Tong. 2017. Towards Better Understanding the Clothing Fashion Styles: A Multimodal Deep Learning Approach. In *AAAI Conf. on Artificial Intelligence (AAAI'17)*. 38–44.

[31] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *International Conf. on Research on Development in Information Retrieval (SIGIR'15)*. 43–52.

[32] Takuma Nakamura and Ryosuke Goto. 2018. Outfit Generation and Style Extraction via Bidirectional LSTM and Autoencoder. In *ACM Knowledge Discovery and Data Mining (KDD'18)*.

[33] Jose Oramas and Tinne Tuytelaars. 2016. Modeling Visual Compatibility through Hierarchical Mid-level Elements. *CoRR* (2016). http://arxiv.org/abs/1604.00036

[34] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the Difficulty of Training Recurrent Neural Networks. In *International Conf. on Machine Learning (ICML'13)*. III–1310–III–1318.

[35] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *International Conf. on Uncertaintyin Artificial Intelligence (UAI'09)*. 452–461.

[36] Steffen Rendle and Lars Schmidt-Thieme. 2010. Pairwise Interaction Tensor Factorization for Personalized Tag Recommendation. In *ACM International Conf. on Web Search and Data Mining (WSDM'10)*. 81–90.

[37] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *International Conf. on Machine Learning (ICML'14)*. 1278–1286.

[38] Edgar Simo-Serra, Sanja Fidler, Francesc Moreno-Noguer, and Raquel Urtasun. 2015. Neuroaesthetics in fashion: Modeling the perception of fashionability. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'15)*, Vol. 00. 869–877.

[39] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conf. on Learning Representations (ICLR'15)*.

[40] Xuemeng Song, Fuli Feng, Xianjing Han, Xin Yang, Wei Liu, and Liqiang Nie. 2018. Neural Compatibility Modeling with Attentive Knowledge Distillation. In *International Conf. on Research on Development in Information Retrieval (SIGIR'18)*.

[41] Xuemeng Song, Fuli Feng, Jinhuan Liu, Zekun Li, Liqiang Nie, and Jun Ma. 2017. NeuroStylist: Neural Compatibility Modeling for Clothing Matching. In *ACM International Conf. on Multimedia (MM'17)*. 753–761.

[42] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)* 15, 1 (2014), 1929–1958.

[43] Andreas Veit, Balazs Kovacs, Sean Bell, Julian Mcauley, Kavita Bala, and Serge Belongie. 2015. Learning Visual Clothing Style with Heterogeneous Dyadic Co-Occurrences. In *International Conf. on Computer Vision (ICCV'15)*. 4642–4650.

[44] Kota Yamaguchi. 2012. Parsing Clothing in Fashion Photographs. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'12)*. 3570–3577.

[45] Kota Yamaguchi, M. Hadi Kiapour, Luis E. Ortiz, and Tamara L. Berg. 2015. Retrieving Similar Styles to Parse Clothing. *IEEE Trans on Pattern Analysis and Machine Intelligence (TPAMI)* 37, 5 (2015), 1028–1040.

[46] Wenhui Yu, Huidi Zhang, Xiangnan He, Xu Chen, Li Xiong, and Zheng Qin. 2018. Aesthetic-based Clothing Recommendation. In *International World Wide Web Conferences (WWW'18)*. 649–658.

[47] Matthew D. Zeiler and Rob Fergus. 2014. Visualizing and Understanding Convolutional Networks. In *European Conf. on Computer Vision (ECCV'14)*. 818–833.

[48] Matthew D. Zeiler, Graham W. Taylor, and Rob Fergus. 2011. Adaptive Deconvolutional Networks for Mid and High Level Feature Learning. In *International Conf. on Computer Vision (ICCV'11)*. 2018–2025.

[49] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. 2017. StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks. In *International Conf. on Computer Vision (ICCV'17)*. 5908–5916.

[50] Hanwang Zhang, Zheng-Jun Zha, Yang Yang, Shuicheng Yan, Yue Gao, and Tat-Seng Chua. 2013. Attribute-augmented Semantic Hierarchy: Towards Bridging Semantic Gap and Intention Gap in Image Retrieval. In *ACM International Conf. on Multimedia (MM'13)*. 33–42.

Learning to Generate Questions by Learning What not to Generate

Bang Liu
University of Alberta, Edmonton, AB, Canada

Mingjun Zhao
University of Alberta, Edmonton, AB, Canada

Di Niu
University of Alberta, Edmonton, AB, Canada

Kunfeng Lai
Platform and Content Group, Tencent, Shenzhen, China

Yancheng He
Platform and Content Group, Tencent, Shenzhen, China

Haojie Wei
Platform and Content Group, Tencent, Shenzhen, China

Yu Xu
Platform and Content Group, Tencent, Shenzhen, China

ABSTRACT

Automatic question generation is an important technique that can improve the training of question answering, help chatbots to start or continue a conversation with humans, and provide assessment materials for educational purposes. Existing neural question generation models are not sufficient mainly due to their inability to properly model the process of how each word in the question is selected, i.e., whether repeating the given passage or being generated from a vocabulary. In this paper, we propose our Clue Guided Copy Network for Question Generation (CGC-QG), which is a sequence-to-sequence generative model with copying mechanism, yet employing a variety of novel components and techniques to boost the performance of question generation. In CGC-QG, we design a multi-task labeling strategy to identify whether a question word should be copied from the input passage or be generated instead, guiding the model to learn the accurate boundaries between copying and generation. Furthermore, our input passage encoder takes as input, among a diverse range of other features, the prediction made by a clue word predictor, which helps identify whether each word in the input passage is a potential clue to be copied into the target question. The clue word predictor is designed based on a novel application of Graph Convolutional Networks onto a syntactic dependency tree representation of each passage, thus being able to predict clue words only based on their context in the passage and their relative positions to the answer in the tree. We jointly train the clue prediction as well as question generation with multi-task learning and a number of practical strategies to reduce the complexity. Extensive evaluations show that our model significantly improves the performance of question generation and out-performs all previous state-of-the-art neural question generation models by a substantial margin.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313737

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Natural language generation**; Machine translation.

KEYWORDS

Question Generation, Copy Prediction, Multi-task Learning, Sequence-to-Sequence, Graph Convolutional Networks

ACM Reference Format:
Bang Liu, Mingjun Zhao, Di Niu, Kunfeng Lai, Yancheng He, Haojie Wei, and Yu Xu. 2019. Learning to Generate Questions by Learning What not to Generate. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3308558.3313737

1 INTRODUCTION

Asking questions plays a vital role for both the growth of human beings and the improvement of artificial intelligent systems. As a dual task of question answering, question generation based on a text passage and a given answer has attracted much attention in recent years. One of the key applications of question generation is to automatically produce question-answer pairs to enhance machine reading comprehension systems [8, 34, 35, 38]. Another application is generating practice exercises and assessments for educational purposes [4, 12, 13]. Besides, question generation is also important in conversational systems and chatbots such as Siri, Cortana, Alexa and Google Assistant, helping them to kick-start and continue a conversation with human users [25].

Conventional methods for question generation rely on heuristic rules to perform syntactic transformations of a sentence to factual questions [2, 12], following grammatical and lexical analysis. However, such methods require specifically crafted transformation and generation rules, with low

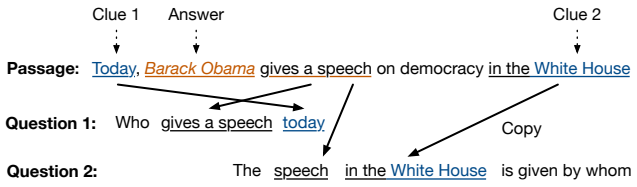


Figure 1: Questions are often asked by repeating some text chunks in the input passage, while there is great flexibility as to which chunks are repeated.

generalizability. Recently, various neural network models have been proposed for question generation [8, 9, 14, 16, 42]. These models formulate the question generation task as a sequence-to-sequence (Seq2Seq) neural learning problem, where different types of encoders and decoders have been designed. Like many other text generation tasks, the copying or pointer mechanism [11] is also widely adopted in question generation to handle the copy phenomenon between input passages and output questions, e.g., [16, 31, 32, 38, 42]. However, we point out that a common limitation that hurdles question generation models mainly use copying mechanisms to handle out-of-vocabulary (OOV) issues, and fail to mark a clear boundary between the set of question words that should be directly copied from the input text and those that should be generated instead.

In this paper, we generate questions by learning to identify where each word in the question should come from, i.e., whether generated from a vocabulary or copied from the input text, and given the answer and its context, what words can potentially be copied from the input passage. People usually repeat text chunks in an input passage when asking questions about it, and generate remaining words from their own language to form a complete question. For example, in Fig. 1, given an input passage "Today, Barack Obama gives a speech on democracy in the White House" and an answer "Barack Obama", we can ask a question "The speech in the White House is given by whom?" Here the text chunks "speech" and "in the White House" are copied from the input passage. However, in the situation that a vocabulary word is an overlap word between a passage and a question, existing models do not clearly identify the underlying reason about the overlapping. In other words, whether this word is copied from the input or generated from vocabulary is not properly labeled. For example, some approaches [42] only take out-of-vocabulary (OOV) words that are shared by both the input passage and target question as copied words, and adopt a copying mechanism to learn to copy these OOV words into questions. In our work, given a passage and a question in a training dataset, we label a word as copied word if it is a nonstop word shared by both the passage and the question, and its word frequency rank in the vocabulary is lower than a threshold. We further aggressively shortlist the output vocabulary based on frequency analysis of the non-copied question words (according to our labeling criteria). Combining this labeling strategy with target vocabulary reduction, during the process of question generation, our model can make better decisions about when to copy or generate, and predict what to generate more accurately.

After applying the above strategies, a remaining problem is which word we should choose to copy. For example, in Fig. 1, we can either ask the question "Who gives a speech today?" or the question "The speech in the White House is given by whom?", where the two questions are related to two different copied text chunks "today" and "White House". We can see that asking a question about a passage and a given answer is actually a "one-to-many" mapping problem: we can ask questions in different ways based on which subset of words we choose to copy from input. Therefore, how to enable a neural model to learn what to copy is a critical problem. To solve this issue, we propose to predict the potential clue words in input passages. A word is considered as a "clue word" if it is helpful to reduce the uncertainty of the model about how to ask a question or what to copy, such as "White House" in question 2 of Fig. 1. In our model, we directly use our previously mentioned copy word labeling strategy to assign a binary label to each input word to indicate whether it is a clue word or not.

To predict the potential clue words in input, we have designed a novel clue prediction model that combines the syntactic dependency tree of an input with Graph Convolutional Networks. The intuition underlying our model design is that: the words copied from an input sentence to an output question are usually closely related to the answer chunk, and the patterns of the dependency paths between the copied words and answer chunks can be captured via Graph Convolutional Networks. Combining the graphical representation of input sentences with GCN enables us to incorporate both word features, such as word vector representation, Named Entity (NE) tags, Part-of-Speech (POS) tags and so on, and the syntactic structure of sentences for clue words prediction.

To generate questions given an answer chunk and the predicted clue words distribution in an input sentence, we apply the Seq2Seq framework which contains our feature-rich sentence encoder and a decoder with attention and copy mechanism. The clue word prediction results are incorporated into the encoder by a binary feature embedding. Based on our multi-task labeling strategy, i.e., labeling for both clue prediction and question generation, we jointly learn different components of our model in a supervised manner.

We performed extensive evaluation on two large question answering datasets: the SQuAD dataset v1.1 [29] and the NewsQA dataset [36]. For each dataset, we use the answer chunk and the sentence that contains the answer chunk as input, and try to predict the question as output. We compared our model with a variety of existing rule-based and neural network based models. Our approach achieves significant improvement by jointly learning the potential clue words distribution for copy and the encoder-decoder framework for generation, and out-performs state-of-the-art approaches.

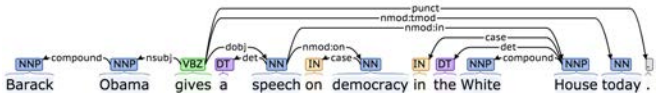


Figure 2: An example to show the syntactic structure of an input sentence. Clue words "White House" and "today" are close to the answer chunk "Barack Obama" with respect to the graph distance, though they are not close to each other in terms of the word order distance.

2 PROBLEM DEFINITION AND MOTIVATION

In this section, we formally introduce the problem of question generation, and illustrate the motivation behind our work.

2.1 Answer-aware Question Generation

Let us denote a passage by P, a question related to this passage by Q, and the answer of that question by A. A passage can be either an input sentence or a paragraph, based on different datasets. A passage consists of a sequence of words $P = \{p_t\}_{t=1}^{|P|}$ where $|P|$ denotes the length of P. A question $Q = \{q_t\}_{t=1}^{|Q|}$ contains words from either a predefined vocabulary V or from the input text P. The task is finding the most probable question $\hat{Q}$ given an input passage and an answer:

$$\hat{Q} = \arg\max_{Q} prob(Q|P, A). \quad (1)$$

Fig. 3 shows an example of the dataset used in our paper. Note that the answer A of the question Q is limited to sub spans of the input passage. However, our work can be easily adapted to the cases where the answer is not a sub span of the input passage by adding an extra answer encoder.

2.2 What to Ask: Clue Word Prediction

Even given the answer of a desired question, the task of question generation is still not a one-to-one mapping, as there are potentially multiple aspects related to the given answer. For example, in Fig. 2, given the answer chunk "Barack Obama", the questions "The speech in the White House is given by whom?", "Who gives a speech on democracy?", and "Today who gives a speech?" are all valid questions. As we can see, although these questions share the same answer, they can be asked in different ways based on which word or phrase we choose to copy (e.g., "White House", "democracy", or "today").

To resolve this issue, we propose to predict the potential "clue words" in the input passage. A "clue word" is defined as a word or phrase that is helpful to guide the way we ask a question, such as "White House", "democracy", and "today" in Fig. 2. We design a clue prediction module based on the syntactical dependency tree representation of passages and Graph Convolutional Networks. Graph Convolutional Networks generalize Convolutional Neural Networks to graph-structured data, and have been successfully applied to natural language processing tasks, including semantic role labeling [23], document matching [20, 39], relation extraction [40], and so on. The intuition underlying our model design is that clue words will be more closely connected to the answer in dependency trees, and the syntactical connection patterns can be learned via graph convolutional networks. For example, in Fig. 2, the graph path distances between answer "Barack Obama" to "democracy", "White House", and "today" are 3, 3, and 2, while the corresponding word order distances between them are 5, 8, and 10, respectively.

Passage:
The Soviet Union and *the People's Republic of China* supported post–World War II communist movements in foreign nations and colonies to advance their own interests, but were not always successful.

Question:
Who along with Russia supported post WW-II communist movements?

Answer:
the People's Republic of China

Figure 3: An example from the SQuAD dataset. Our task is to generate questions given an input passage and an answer. In SQuAD dataset, answers are sub spans of the passages.

2.3 How to Ask: Copy or Generate

Another important problem of question generation is when to choose a word from the target vocabulary and when to copy a word from the input passage during the generation process. People tend to repeat or paraphrase some text pieces in a given passage when they ask a question about it. For instance, in Fig. 3, the question "Who along with Russia supported post WW-II communist movements", the text pieces "supported", "post", and "communist movements" are repeating the input passage, while "Russia" and "WW-II" are synonymous replacements of input phrases "The Soviet Union" and "World War II". Therefore, the copied words in questions should not be restricted to out-of-vocabulary words. Although this phenomenon is well-known by existing approaches, they do not properly and explicitly distinguish whether a word is from copy or from generate.

In our model, we consider the non-stop words that are shared by both an input passage and the output question as clue words, and encourage the model to copy clue words from input. After labeling clue words, we train a GCN-based clue predictor to learn whether each word in source text can be copied to the target question. The predicted clue words are further fed into a Seq2Seq model with attention and copy mechanism to help with question generation. Besides, different from existing approaches that share a vocabulary between source passages and target questions, we reduce the size of the target vocabulary of questions to be smaller than source passages and only include words with word frequency higher than a threshold. The idea is that the low-frequency words in questions are usually copied from source text rather

than generated from a vocabulary. By combining the above strategies, our model is able to learn when to generate or copy a word, as well as which words to copy or generate during the progress of question generation. We will describe our operations in more detail in the next section.

3 MODEL DESCRIPTION

In this section, we introduce our proposed framework in detail. Similar to [8, 30, 42], our question generator is based on an encoder-decoder framework, with attention and copying mechanisms incorporated. Fig. 4 illustrates the overall architecture and detailed components in our model. Our model consists of three components: the clue word predictor, passage encoder and question decoder.

The clue word predictor predicts potential clue words that may appear a target question, based on the specific context of the input passage (without knowing the target question). It utilizes a syntactic dependency tree to reveal the relationship of answer tokens relative to other tokens in a sentence. Based on the tree representation of the input passage, we predict the distribution of clue words by a GCN-based encoder applied on the tree and a Straight-Through (ST) Gumbel-Softmax estimator [15] for clue word sampling. The outputs of the clue word predictor are fed to the passage encoder to advise the encoder about what words may potentially be copied to the target question.

The passage encoder incorporates both the predicted clue word distribution and a variety of other feature embeddings of the input words, such as lexical features, answer position indicators, etc. Combined with a proposed low-frequency masking strategy (a shortlist strategy to reduce complexity on tuning input word embeddings), our encoder learns to better capture the useful input information with fewer trainable parameters.

Finally, the decoder jointly learns the probabilities of generating a word from vocabulary and copying a word from the input passage. At the decoder side, we introduce a multi-task learning strategy to intentionally encourage the copying behavior in question generation. This is achieved by explicitly labeling the copy gate with a binary variable when a (non-stop) word in the question also appears in the input passage in the training set. Other multi-task labels are also incorporated to accurately learn when and what to copy. We further shortlist the target vocabulary based on the frequency distribution of non-overlap words. These strategies, assisted by the encoded features, help our model to clearly learn which word in the passage is indeed a clue word to be copied into the target question, while generating other non-clue words accurately.

The entire model is trained end-to-end via multitask learning, i.e., by minimizing a weighted sum of losses associated with different labels. In the following, we first introduce the encoder and decoder, followed by a description of the clue word predictor.

3.1 The Passage Encoder with Masks

Our encoder is based on bidirectional Gated Recurrent Unit (BiGRU) [3], taking word embeddings, answer position indicators, lexical and frequency features of words, as well as the output of the clue word predictor as the input. Specifically, for each word p_i in input passage P, we concatenate the following features to form a concatenated representation w_i to be input into the encoder:

- **Word Vector**. We initialize each word vector by Glove embedding [28]. If a word is not covered by Glove, we initialize its word vector randomly.
- **Lexical Features**. We perform Named Entity Recognition (NER), Part-of-Speech (POS) tagging and Dependency Parsing (DEP) on input passages using spaCy [24], and concatenate the lexical feature embedding vectors.
- **Binary Features**. We check whether each word is lowercase or not, whether it is a digit or like a number (such as word "three"), and embed these features by vectors.
- **Answer Position**. Similar to [42], we utilize the B/I/O tagging scheme to label the position of a given answer, where a word at the beginning of an answer is marked with **B**, **I** denotes the continuation of the answer, while words not contained in an answer are marked with **O**.
- **Word Frequency Feature**. We derive the word vocabulary from passages and questions in the training dataset. We then rank all the words in a descending order in terms of word frequencies, such that the first word is the most frequent word. The top r_h words are labeled as frequent words. Words ranked between r_h and r_l are labeled as intermediate words. The remaining with rank lower than r_l are labeled as rare words, where r_h and r_l are two predefined thresholds. In this way, each word will be assigned a frequency tag **L** (low frequency), **H** (highly frequent) or **M** (medium frequency).
- **Clue Indicator Feature**. In our model, the clue predictor (which we will introduce in more detail in Sec. 3.3) assigns a binary value to each word to indicate whether it is a potential clue word or not.

Denote an input passage by $P = (w_1, w_2, \cdots, w_{|P|})$. The BiGRU encoder reads the input sequence $w_1, w_2, \cdots, w_{|P|}$ and produces a sequence of hidden states $h_1, h_2, \cdots, h_{|P|}$ to represent the passage P. Each hidden state is a concatenation of a forward representation and a backward representation:

$$h_i = [\overrightarrow{h}_i; \overleftarrow{h}_i], \tag{2}$$

$$\overrightarrow{h}_i = \mathbf{BiGRU}(w_i, \overrightarrow{h}_{i-1}), \tag{3}$$

$$\overleftarrow{h}_i = \mathbf{BiGRU}(w_i, \overleftarrow{h}_{i+1}), \tag{4}$$

where $\overrightarrow{h}_i$ and $\overleftarrow{h}_i$ are the forward and backward hidden states of the i-th token in P, respectively.

Furthermore, rather than learning the full representation w_i for every word in the vocabulary, we use a masking strategy to replace the word embeddings of low-frequency words

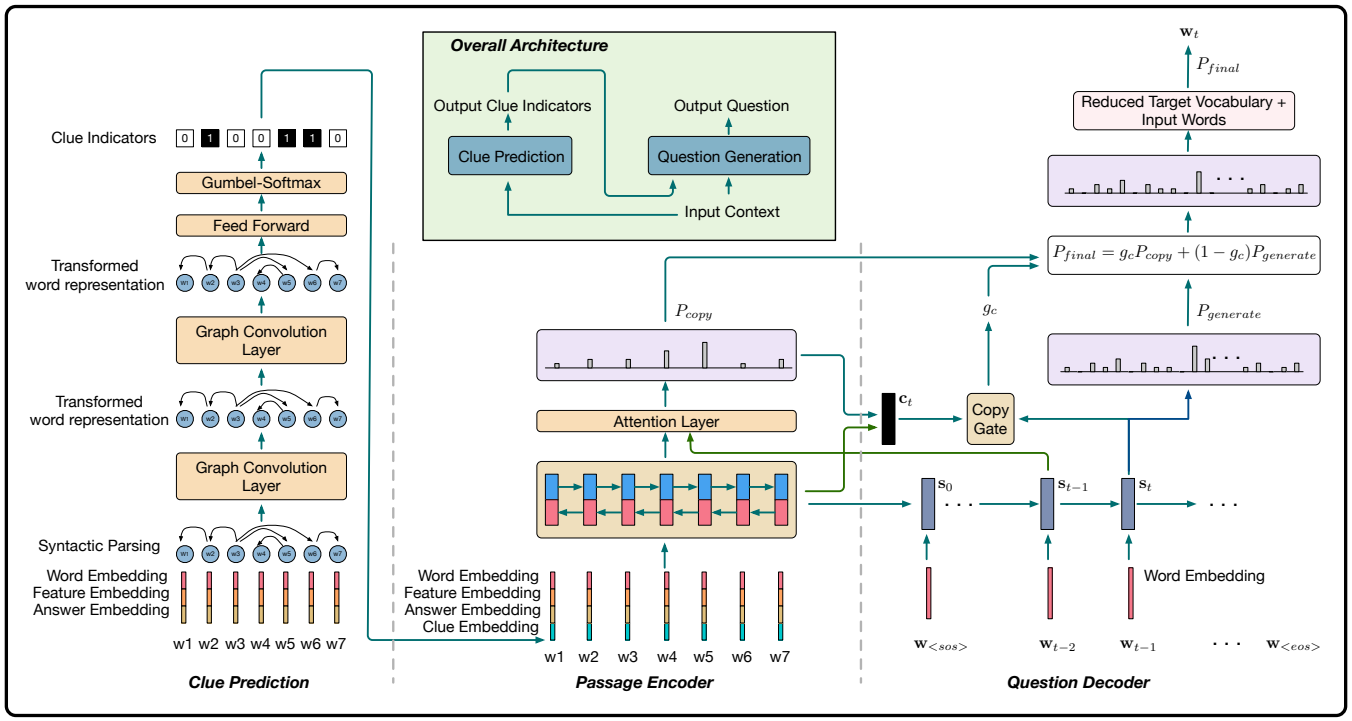


Figure 4: Illustration of the overall architecture of our proposed model. It contains a GCN-based clue word predictor, a masked feature-rich encoder for input passages, and an attention-and-copy-based decoder for generating questions.

with a special ¡l¿ token, such that the information of low-frequency words is only represented by its answer/clue indicators and other augmented feature embeddings, except the word vectors. This strategy can improve performance due to two reasons. First, the augmented tagging features of a low-frequency word tend to be more influential than the word meaning in question generation. For example, given a sentence "¡PERSON¿ likes playing football.", a question that can be generated is "What does ¡PERSON¿ like to play?"—what the token "¡PERSON¿" exactly is does not matter. This way, the masking strategy helps the model to omit the fine details that are not necessary for question generation. Second, the number of parameters that need be learned, especially the number of word embeddings that need be tuned, is largely reduced. Therefore, masking out unnecessary word embeddings while only keeping the corresponding augmented features and indicators does not hurt the performance of question generation. It actually improves training by reducing the model complexity.

3.2 The Question Decoder with Aggressive Copying

In the decoding stage, we utilize another GRU with copying mechanism to generate question words sequentially based on the encoded input passage representation and previously decoded words. We first initialize the hidden state of the decoder GRU by passing the last backward encoder hidden state $\overleftarrow{h}_1$ to a linear layer:

$$s_0 = tanh(\mathbf{W}_0 \overleftarrow{h}_1 + b). \tag{5}$$

For each decoding time step t, the GRU reads the embedding of the previous word w_{t-1}, previous attentional context vector c_{t-1}, and its previous hidden state s_{t-1} to calculate its current hidden state:

$$s_t = \mathbf{GRU}([w_{t-1}; c_{t-1}], s_{t-1}). \tag{6}$$

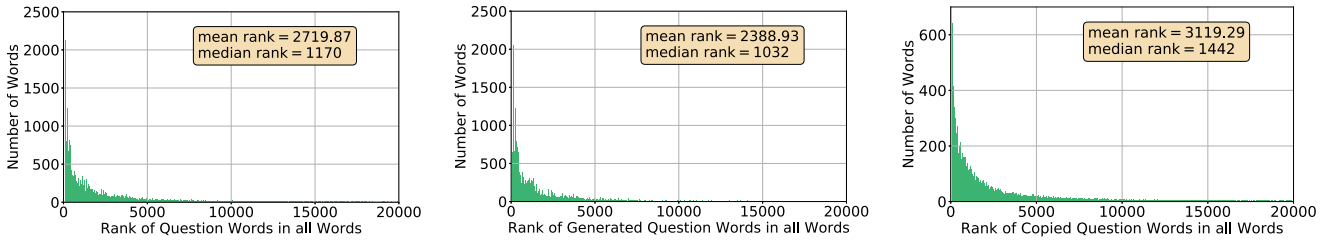
The context vector c_t for time step t is generated through the concatenated attention mechanism [21]. Attention mechanism calculates a semantic match between encoder hidden states and the decoder hidden state. The attention weights indicate how the model spreads out the amount it cares about different encoder hidden states during decoding. At time step t, the attention weights and the context vector are calculated as:

$$e_{t,i} = v^\intercal tanh(\mathbf{W}_s s_t + \mathbf{W}_h h_i), \tag{7}$$

$$\alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{j=1}^{|P|} \exp(e_{t,j})}, \tag{8}$$

$$c_t = \sum_{i=1}^{|P|} \alpha_{t,i} h_i. \tag{9}$$

Combining the previous word embedding w_{t-1}, the current decoder state s_t and the current context vector c_t, we can calculate a readout state r_t by an MLP maxout layer with dropouts [10]. Then the readout state is passed to a linear



(a) Rank Distribution of All Question Words (b) Rank Distribution of Generated Question Words (c) Rank Distribution of Copied Question Words

Figure 5: Comparing the rank distributions of all question words, words from generation, and words from copy.

layer and a softmax layer to predict the probabilities of the next word over the decoder vocabulary:

$$r_t = \mathbf{W}_{rw} w_{t-1} + \mathbf{W}_{rc} c_t + \mathbf{W}_{rs} s_t \tag{10}$$

$$m_t = [\max\{r_{t,2j-1}, r_{t,2j}\}]_{j=1,\ldots,d}^{\intercal} \tag{11}$$

$$p(y_t|y1, \cdots, y_{t-1}) = \text{softmax}(\mathbf{W}_o m_t), \tag{12}$$

where r_t is a 2-D vector.

The above module generates question words from a given vocabulary. Another important method to generate words is copying from source text. Copy or point mechanism [11] was introduced into sequence-to-sequence models to allow the copying of unknown words from the input text, which solves the out-of-vocabulary (OOV) problem. When decoding at time step t, the probability of copying is given by:

$$g_c = \sigma(\mathbf{W}_{cs} s_t + \mathbf{W}_{cc} c_t + b), \tag{13}$$

where σ is the Sigmoid function, and g_c is the probability of copying. For the copy probability of each input word, we reuse the attention weights given by Equation (8).

Copying mechanism has also been used in question generation [14, 31, 42], however, mainly to solve the OOV issue. Here we leverage the copying mechanism to enable the copying of potential clue words, instead of being limited to OOV words, from input. Different from existing methods, we take a more aggressive approach to train the copying mechanism via multitask learning based on different labels. That is, when preparing the training dataset, we explicitly label a word in a target question as a word copied from the source passage text if it satisfies all the following criteria:

i) it appears in both source text and target question;
ii) it is not a stop word;
iii) its frequency rank in the vocabulary is lower than a threshold r_h.

The remaining words in the question are considered as being generated from the vocabulary. Such a binary label (copy or not copy), together with which input word the question word is copied from, as well as the target question, are fed into different parts of the decoder as labels for multi-task model training. This is to intentionally encourage the copying of potential clue words into the target question rather than generating them from a vocabulary.

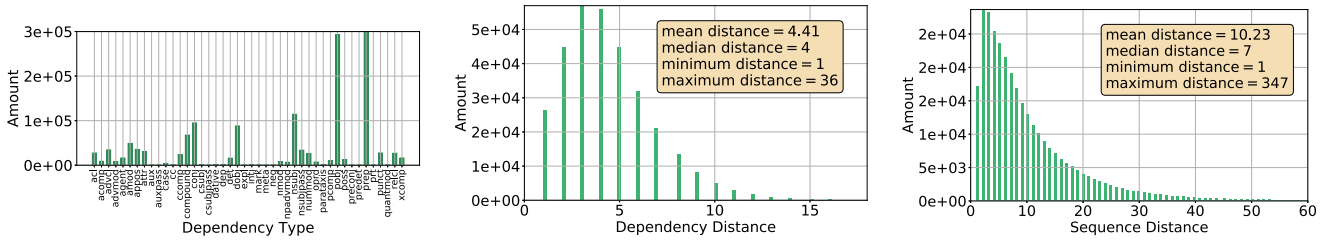
The intuition behind such an aggressive copying mechanism can be understood by checking the frequency distributions of both generated words and copied words (as defined above) in the training dataset of SQuAD. Fig. 5 shows the frequency distributions of all question words, and then of generated question words and copied question words. The mean and median rank of generated words are 2389 and 1032, respectively. While for copied words, they are 3119 and 1442. Comparing Fig. 5(b) with Fig. 5(c), we can see that question words generated from the vocabulary tends to be clustered toward high ranked (or frequent) words. On the other hand, the fraction of low ranked (or infrequent) words in copied words are much greater than that in generated words. This means the generated words are mostly from frequent words, while the majority of low-frequency words in the long tail are copied from the input, rather than generated.

This phenomenon matches our intuition: when people ask a question about a given passage, the generated words tend to be commonly used words, while the repeated text chunks from source passage may contain relatively rare words such as names and dates. Based on this observation, we further propose to reduce the vocabulary at the decoder for target question generation to be top N frequently generated words, where N is a predefined threshold that varies according to datasets.

3.3 A GCN-Based Clue Word Predictor

Given a passage and some answer positions, our clue word predictor aims to identify the clue words in the passage that can help to ask a question and are also potential candidates for copying, by understanding the semantic context of the input passage. Again, in the training dataset, the non-stop words that are shared by both an input passage and an output question are aggressively labeled as *clue words*.

We note that clue words are, in fact, more closely connected to the answer chunk in the form of syntactic dependency trees than in word sequences. Fig. 6 shows our observation on the SQuAD dataset. For each training example, we get the nonstop words that appear in both the input passage and the output question. For each clue word in the training set, we find its shortest undirected path to the answer chunk based on the dependency parsing tree of the passage. For each jump



(a) Distribution of Syntactic Dependency Types between Copied Words and the Answer

(b) Distribution of Syntactic Dependency Distances between Copied Words and theAnswer

(c) Distribution of Sequential Word Distances between Copied Words and the Answer

Figure 6: Comparing the distributions of syntactic dependency distances and sequential word distances between copied words and the answer in each training sample.

on the shortest path, we record the dependency type. We also calculate the distance between each clue word and the answer in terms of the number of words between them. As shown in Fig. 6(a), *prep*, *pobj*, and *nsubj* appear frequently on these shortest paths. Comparing Fig. 6(b) with Fig. 6(c), we can see that the distances in terms of dependency trees are much smaller than those in terms of sequential word orders. The average and median distances on the dependency trees are 4.41 and 4, while those values are 10.23 and 7 for sequential word distances.

In order to predict the positions of clue words based on their dependencies on the answer chunk (without knowing the question which is yet to be generated), we use a Graph Convolutional Network (GCN) to convolve over the word features on the dependency tree of each passage, as shown in Fig. 4. The predictor consists of four layers:

Embedding layer. This layer shares the same features with the passage encoder, except that it does not include the clue indicators. Therefore, each word is represented by its word embedding, lexical features, binary features, the word frequency feature, and an answer position indicator.

Syntactic dependency parsing layer. We obtain the syntactic dependency parsing tree of each passage by spaCy [24], where the dependency edges between words are directed. In our model, we use the syntactic structure to represent the structure of passage words.

Encoding layer. The objective of the encoding layer is to encode the context information into each word based on the dependency tree. We utilize a multi-layered GCN to incorporate the information of neighboring word features into each vertex, which is a word. After L GCN layers, the hidden vector representation of each word will incorporate the information of its neighboring words that are no more than L hops away in the dependency tree.

Output layer. After obtaining the context-aware representation of each word in the passage, we calculate the probability of each word being a clue word. A linear layer is utilized to calculate the unnormalized probabilities. We subsequently sample the binary clue indicator for each word through a Gumbel-Softmax layer, given the unnormalized probabilities. A sampled value of 1 indicates that the word is predicated as a clue word.

3.3.1 GCN Operations on Dependency Trees. We now introduce the operations performed in each GCN layer [18, 40]. GCNs generalize the CNN from low-dimensional regular grids to high-dimensional irregular graph domains. In general, the input to a GCN is a graph $G = (\mathcal{V}, E)$ with N vertices $v_i \in \mathcal{V}$, and edges $e_{ij} = (v_i, v_j) \in E$. The edges can be weighted with weights w_{ij}, or unweighted. The input also contains a vertex feature matrix denoted by $X = \{\mathbf{x}_i\}_{i=1}^{N}$, where $\mathbf{x}_i$ is the *feature vector* of vertex v_i.

Since a dependency tree is also a graph, we can perform the graph convolution operations on dependency trees by representing each tree into its corresponding adjacency matrix form. Now let us briefly introduce the GCN propagation layers used in our model [40]. The weighted adjacency matrix of the graph is denoted as $A \in \mathbb{R}^{N \times N}$ where $A_{ij} = w_{ij}$. For unweighted graphs, the weights are either 1 or 0. In an L-layer GCN, let $h_i^{(l-1)}$ denotes the input vector and $h_i^{(l)}$ denotes the output vector of node i at the l-th layer. We will utilize a multi-layer GCN with the following layer-wise propagation rule [40]:

$$h_i^{(l)} = \sigma(\sum_{j=1}^{N} \tilde{\mathbf{A}}_{ij} \mathbf{W}^{(l)} h_j^{(l-1)} / d_i + b^{(l)}), \tag{14}$$

where σ is a nonlinear function (e.g., ReLU), and $W^{(l)}$ is a linear transformation. $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ where $\mathbf{I}_N$ is an $n \times n$ identity matrix. $d_i = \sum_{j=1}^{N} \tilde{\mathbf{A}}_{ij}$ is the degree of node (or word in our case) i in the graph (or dependency tree).

In our experiments, we treat the dependency trees as undirected, i.e., $\forall i, j, \mathbf{A}_{ij} = \mathbf{A}_{ji}$. Besides, as we already included the dependency type information in the embedding vectors of each word, we do not need to incorporate the edge type information in the adjacency matrix.

Stacking this operation by L layers gives us a deep GCN network. The input to the nodes in the first layer of the GCN are the feature vectors of words in the passage. After L layers of transformations, we can obtain a context-aware representation of each word. We then feed them into a linear layer to get the unnormalized probability of each word being a clue word. After that, the unnormalized probabilities are fed to a Straight-Through (ST) Gumbel-Softmax estimator

to sample an N-dimensional binary vector indicating whether each of the N words is a clue word or not.

3.3.2 Gumbel-Softmax. Gumbel-Softmax [15] is a method of sampling discrete random variables in neural networks. It approximates one-hot vectors sampled from a categorical distribution by making them continuous, therefore the gradients of model parameters can be calculated using the reparameterization trick and the standard backpropagation. Gumbel-Softmax distribution is motivated by Gumbel-Max trick [22], an algorithm for sampling from a categorical distribution. Let $(p_1, ..., p_k)$ denotes a k-dimensional categorical distribution where the probability p_i of class i is defined as:

$$p_i = \frac{\exp(\log(\pi_i))}{\sum_{j=1}^{k} \exp(\log(\pi_j))}, \tag{15}$$

where π_i is the unnormalized log probability of class i. We can easily draw a one-hot sample $\mathbf{z} = (z_1, \cdots, z_k) \in \mathbb{R}^k$ from the distribution by the following equations:

$$z_i = \begin{cases} 1, & i = \arg\max_j (log(\pi_j) + g_j) \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

$$g_i = -\log(-\log(u_i)), \tag{17}$$

$$u_i \sim \text{Uniform}(0, 1) \tag{18}$$

where g_i is Gumbel noise used to perturb each $\log(\pi_i)$. In this way, taking arg max is equivalent to drawing a sample using probabilities $(p_1, \cdots, p_k)$.

Gumbel-Softmax distribution replaces the arg max function by differentiable softmax function. Therefore, a sample $\mathbf{y} = (y_1, \cdots, y_k)$ drawn from Gumbel-Softmax distribution is given by:

$$y_i = \frac{\exp((\log(\pi_i) + g_i)/\tau)}{\sum_{j=1}^{k} \exp((\log(\pi_j) + g_j)/\tau)}, \tag{19}$$

where τ is a temperature parameter. The Gumbel-Softmax distribution resembles the one-hot sample when τ diminishes to zero.

Straight-Through (ST) Gumbel-Softmax estimator [15] is a discrete version of the continuous Gumbel-Softmax estimator. It takes different paths in the forward and backward propagation. In the forward pass, it discretizes a continuous probability vector $\mathbf{y}$ sampled from the Gumbel-Softmax distribution into a one-hot vector $\mathbf{y}^{ST} = (y_1^{ST}, \cdots, y_k^{ST})$ by:

$$y_i^{ST} = \begin{cases} 1, & i = \arg\max_j y_j, \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

In the backward pass, it uses the continuous $\mathbf{y}$, so that the error signal can still backpropagate.

Using the ST Gumbel-Softmax estimator, our model is able to sample a binary clue indicator vector for an input passage. Then the clue indicator vector is fed into the passage encoder for question generations, as shown in Fig. 4.

4 EVALUATION

In this section, we evaluate the performance of our proposed models on the SQuAD dataset and the NewsQA dataset, and compare them with state-of-the-art question generation models.

Table 1: Description of evaluation datasets.

Dataset	Train	Dev	Test	P-Length*	Q-Length*	A-Length*
SQuAD	86,635	8,965	8,964	32.72	11.31	3.19
NewsQA	77,538	4,341	4,383	28.14	7.82	4.60

* P-Length: average number of tokens of passages.

* Q-Length: average number of tokens of questions.

* A-Length: average number of tokens of answers.

4.1 Datasets, Metrics and Baselines

The SQuAD dataset is a reading comprehension dataset, consisting of questions posed by crowd-workers on a set of Wikipedia articles, where the answer to every question is a segment of text from the corresponding reading passage. SQuAD 1.1 is used in our experiment containing 536 Wikipedia articles and more than 100K question-answer pairs. When processing a sample from dataset, instead of using the entire document, we take the sentence that contains the answer as the input. Since the test set is not publicly available, we use the data split proposed by [42] where the original dev set is randomly split into a dev test and a test set of equal size.

In the NewsQA dataset, there are 120K questions and their corresponding answers as well as the documents that are CNN news articles. Questions are written by questioners in natural language with only the headlines and highlights of the articles available to them. With the information of the questions and the full articles, answerers select related sub-spans from the passages of the source text and mark them as answers. Multiple answers may be provided to a same question by different answerers and they are ranked by validators based on the quality of the answers. In our experiment, we picked a subset of NewsQA where answers are top-ranked and are composed of a contiguous sequence of words within the input sentence of the document.

Table 1 shows the number of samples in each set and the average number of tokens of the input sentences, questions, and answers listed in columns P-Length, Q-Length, and A-Length respectively.

We report the evaluation results with following metrics.

- **BLEU** [26]. BLEU measures precision by how much the words in prediction sentences appear in reference sentences at the corpus level. BLEU-1, BLEU-2, BLEU-3, and BLEU-4, use 1-gram to 4-gram for calculation, respectively.
- **ROUGE-L** [19]. ROUGE-L measures recall by how much the words in reference sentences appear in prediction sentences using Longest Common Subsequence (LCS) based statistics.

- **METEOR** [6]. METEOR is based on the harmonic mean of unigram precision and recall, with recall weighted higher than precision.

In the experiments, we have eight baseline models for comparison. Results reported on PCFG-Trans, MPQG, and NQG++ are from experiments we conducted using published code on GitHub. For other baseline models, we directly copy the reported performance given in their papers. We report all the results on the SQuAD dataset, and for the NewsQA dataset, we can only report the baselines with open source code available.

- **PCFG-Trans** [12] is a rule-based system that generates a question based on a given answer word span.
- **MPQG** [31] proposed a Seq2Seq model that matches the answer with the passage before generating the question.
- **SeqCopyNet** [43] proposed a method to improve the copying mechanism in Seq2Seq models by copying not only a single word but a sequence of words from the input sentence.
- **seq2seq+z+c+GAN** [37] proposed a model employed in GAN framework using the latent variable to capture the diversity and learning disentangled representation using the observed variable.
- **NQG++** [42] proposed a Seq2Seq model with a feature-rich encoder to encode answer position, POS and NER tag information.
- **Answer-focused Position-aware model** [33] incorporates the answer embedding to help generate an interrogative word matching the answer type. And it models the relative distance between the context words and the answer for the model to be aware of the position of the context words when generating a question.
- **s2sa-at-mp-gsa** [41] proposed a model which contains a gated attention encoder and a maxout pointer decoder to address the challenges of processing long text inputs. This model has a paragraph-level version and a sentence-level version. For the purpose of fair comparison, we report the results of the sentence-level model to match with our settings.
- **ASs2s** [16] proposed an answer-separated Seq2Seq to identify which interrogative word should be used by replacing the target answer in the original passage with a special token.

For our models, we evaluate the following versions:

- **CGC-QG (no feature-rich embedding)**. We name our model as Clue Guided Copy for Question Generation (CGC-QG). In this variant, we only keep the embedding of words, answer position indicators, and clue indicators for each token, and remove the embedding vectors of other features.
- **CGC-QG (no target reduction)**. This model variant does not contain target vocabulary reduction operation.
- **CGC-QG (no clue prediction)**. The clue predictor and clue embedding are removed in model variant.
- **CGC-QG**. This is the complete version of our proposed model.

4.2 Experiment Settings

We implement our models in PyTorch 0.4.1 [27] and train the model with a single Tesla P40. We utilize spaCy [24] to perform dependency parsing and extract lexical features for tokens. As to the vocabulary, we collect it from the training dataset and keep the top 20K most frequent words for both datasets.

We set the threshold $r_h = 100$ and $r_l = 2000$. For target vocabulary reduction, we set $N = 2000$. The embedding dimension of word vector is set to be 300 and initialized by GloVe. The word vectors of words that are not contained in GloVe are initialized randomly. The word frequency features are embedded to 32-dimensional vectors, and other features and indicators are embedded to 16-dimensional vectors. All embedding vectors are trainable in the model. We use a single layer BiGRU with hidden size 512 for the encoder, and a single layer undirected GRU with hidden size 512 for the decoder. The dropout rate $p = 0.1$ is applied to the encoder, decoder, and the attention module.

During training, we optimize the Cross-Entropy loss function for clue prediction, question generation, and question copying, and perform gradient descent by the Adam [17] optimizer with an initial learning rate $l_r = 0.001$, two momentum parameters are $\beta_1 = 0.8$ and $\beta_2 = 0.999$ respectively, and $\epsilon = 10^{-8}$. The mini-batch size for each update is set to 32 and model is trained for up to 10 epochs (as we found that usually the models derive the best performance after 6 or 7 epochs). We apply gradient clipping with range $[-5, 5]$ for Adam. Besides, exponential moving average is applied on all trainable variables with a decay rate 0.9999. When testing, beam search is conducted with beam width 20. The decoding process stops when a token *¡EOS¿* that represents end of sentence is generated.

4.3 Main Results

Table. 2 and Table. 3 compare the performance of our model with existing question generation models on SQuAD and NewsQA in terms of different evaluation metrics. For the SQuAD dataset, we compare our model with all the baseline methods we have listed. As to the NewsQA dataset, since only a part of the baseline methods made their code public, we compare our model with approaches that have open source code. We can see that our model achieves the best performance on both datasets and significantly outperforms state-of-the-art algorithms. On the SQuAD dataset, given an input sentence and an answer, the BLEU-4, ROUGE-L, and METEOR of our result are 17.55, 44.53, and 21.24 respectively, while corresponding previous state-of-the-art results are 16.17, 44.24, and 19.67 from different approaches. Similarly, our method also gives a significantly better performance on NewsQA compared with the baselines in Table 3.

Table 2: Evaluation results of different models on SQuAD dataset.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR
PCFG-Trans*	28.77	17.81	12.64	9.47	31.68	18.97
SeqCopyNet	–	–	–	13.02	44.00	–
seq2seq+z+c+GAN	44.42	26.03	17.60	13.36	40.42	17.70
NQG++*	42.36	26.33	18.46	13.51	41.60	18.18
MPQG	–	–	–	13.91	–	–
Answer-focused Position-aware model	43.02	28.14	20.51	15.64	–	–
s2sa-at-mp-gsa	44.51	29.07	21.06	15.82	44.24	19.67
ASs2s	–	–	–	16.17	–	–
CGC-QG (no feature-rich embedding)	45.50	29.63	21.58	16.38	43.11	20.52
CGC-QG (no target reduction)	45.80	30.27	22.29	17.05	44.09	20.79
CGC-QG (no clue prediction)	45.58	30.07	22.08	16.80	44.51	20.80
CGC-QG	**46.58**	**30.90**	**22.82**	**17.55**	**44.53**	**21.24**

Experiments are conducted on baselines followed by a "*" using released source code. Results of other baselines are copied from their papers where unreported metrics are marked "–".

Table 3: Evaluation results of different models on NewsQA dataset.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR
PCFG-Trans*	16.90	7.94	4.72	3.08	23.78	13.74
MPQG*	35.70	17.16	9.64	5.65	39.85	14.13
NQG++*	40.33	22.47	14.83	9.94	42.25	16.72
CGC-QG (no feature-rich embedding)	39.35	22.10	14.36	9.99	42.00	16.60
CGC-QG (no target reduction)	40.00	22.84	15.01	10.52	42.33	16.89
CGC-QG (no clue prediction)	39.85	22.82	14.96	10.45	43.16	17.11
CGC-QG	**40.45**	**23.52**	**15.68**	**11.06**	**43.16**	**17.43**

Experiments are conducted on baselines followed by a "*" using released source code.

The reason is that we combine different strategies in our model to make it learn when to generate or copy a word, and what to generate or copy. First, our model learns to predict clue words through a GCN-based clue predictor. Second, our encoder incorporates a variety of embeddings of different features and clue indicators. Combining with the masking strategy, our model can better discover the relationship between input patterns and output patterns. Third, the reduced target vocabulary also helps our model to better capture when to copy or generate, and the generator is easier to train with a reduced vocabulary size. And most importantly, our new criteria of marking a question word as copied word (as described in Sec.3.3) helps the model to make better decisions on which path to go, i.e., to copy or to generate, during question generation. By incorporating part of these new strategies and modules into our model, we can achieve performance better than state-of-the-art models on SQuAD and NewsQA. With all these designs implemented, our model gives the best performance on both datasets.

There is a significant gap between the performance on SQuAD and on NewsQA due to the different characteristics of the datasets. The average answer length of NewsQA is 44.2% larger than it of SQuAD according to the statistics shown in Table 1. Long answers usually hold more information and are more difficult to generate questions. Furthermore, reference questions in NewQA tend to have less strict grammars and more diverse phrasings. To give a typical example, "Iran criticizes who?" is a reference question in NewsQA which does not start with an interrogative word but ends with one. These characteristics make the performance on NewsQA not as good as on SQuAD. However, our approach is still significantly better than the compared approaches on NewsQA dataset. It demonstrates that copy from the input is a general phenomenon across different datasets. Our model better captures what copied words are and what generated words are in a question based on our new criteria of labeling a question word as copied word or not.

4.4 Analysis

We evaluate the impact of different modules in our model by ablation tests. Table 2 and Table 3 list the performance of our model variants with different sub-components removed.

When we remove the extra feature embeddings in our model, i.e., the embeddings of POS, NER, Dependency Types,

word frequency levels (low-frequent, median-frequent, high-frequent), and binary features (whether it is lowercase, digit), the performance drops significantly. This is because the tags and feature embeddings represent each token in different aspects. The number of different tags is much smaller than the number of different words. Therefore, the patterns which can be learned from these tags and features are more obvious than what we can learn from word embeddings. Even though a well-trained word vector may contain the information of other features such as POS or NER, explicitly concatenating these feature embedding vectors helps the model to capture the patterns to ask a question more easily.

Removing the operation of target vocabulary reduction also hurts the performance of our model. As we discussed earlier, the non-overlap question words (or generated words) are mostly covered by the high frequency words. Reducing the target vocabulary size helps our model to better learn the probabilities of generating these words. On the other hand, it also encourages the model to better capture what they can copy from input text.

Finally, without the clue prediction module, the performance also drops on both datasets. This is because when given an answer span in a passage, asking a question about it is still a one-to-many mapping problem. Our clue prediction module learns how people select the related clue words to further reduce the uncertainty of how to ask a question by learning from a large training dataset. With predicted clue indicators incorporated into the encoder of generator, our model can fit the way how people ask questions in the dataset.

5 RELATED WORK

In this section, we review related works on question generation and graph convolutional networks.

Existing approaches for question generation can be mainly classified into two classes: heuristic rule-based approaches and neural network-based approaches. The rule-based approaches rely on well-designed rules or templates manually created by human to transform a piece of given text to questions [2, 12, 13]. However, they require creating rules and templates by experts which is extremely expensive. Also, rules and templates have a lack of diversity and are hard to generalize to different domains.

Compared with rule-based approaches, neural network-based models are trained end-to-end and do not rely on hand-crafted rules or templates. Most of the neural question generation models consider the task as Seq2Seq and take advantage of the encoder-decoder framework with attention mechanism. [30] utilizes an encoder-decoder framework with attention mechanism to generate factoid questions from FreeBase. [8] generates questions from SQuAD passages based on a sequence-to-sequence model with attention mechanism.

However, given an input sentence, generating questions is a one-to-many mapping, as we can ask different questions from different aspects. Purely relying on Seq2Seq model may not be able to learn such a one-to-many mapping [9]. To resolve this issue, recent works assume the aspect is known when generating a question [16, 31, 32, 38, 42] or can be detected by a third-party pipeline [7]. [42] enriches the sequence-to-sequence model with answer position indicator to indicate if the current word is an answer word or not, and further incorporates copy mechanism to copy words from the context when generating a question. [14, 31] fuse the answer information into input sentence first, and apply a Seq2Seq model with attention and copy mechanism to generate answer-aware questions. [9] takes question difficulty into account by a difficulty estimator, and generate questions on different difficulty levels. [33] enriches the model with both answer embedding and the relative distance between the context words and the answer. [34, 35] model question answering and question generation as dual tasks. They found that jointly training the two tasks helped to generate better questions.

We argue that even with an answer indicator, the problem of question generation is still a one-to-many mapping. To solve this problem, we enrich the model with a "clue word" predictor, where a clue word means a word that is related to the aspect of the targeting output question and usually copied to the question. We represent an input sentence by its syntactic dependency tree and represent each word using feature-rich embeddings, and let the model learn to predict whether each word in context can be a clue word or not. Combining the clue prediction module with a Seq2Seq model with attention and copy mechanism, our model learns when to generate a word from a target vocabulary and when to copy a word from the input context.

Graph Convolutional Networks generalize Convolutional Neural Networks to graph-structured data, and have been developed and grown rapidly in scope and popularity in recent years [1, 5, 18, 20, 23]. Here we focus on the applications of GCNs on natural language. [23] applies GCNs over syntactic dependency trees as sentence encoders, and produces latent feature representations of words in a sentence for semantic role labeling. [20] matches long document pairs using graph structures, and classify the relationships of two documents by GCN. [40] proposes an extension of graph convolutional networks that is tailored for relation extraction. It pools information over dependency trees efficiently in parallel. In our paper, we apply GCN over the dependency tree of an input sentence, and predict the potential clue words in the sentence together with a Gumbel-Softmax estimator.

6 CONCLUSION

In this paper, we demonstrate the effectiveness of teaching the model to make decisions during the question generation process on which words to generate and to copy. We label the nonstop and overlap words between input passages and questions as copy targets and use such labels to train our model. Besides, we further observe that the distribution of generated question words are mostly common words with relative high frequency. Based on this observation, we reduce the vocabulary size for generating question words. To help the model better capture how to ask a question and alleviate

the issue of one-to-many mapping when asking a question, we propose a GCN-based clue prediction module to predict which part of words can be a clue word to ask a question given an answer. It utilizes the syntactic dependency tree representation of a passage to encode the information of each token in the passage, and sample a clue indicator for each token using a Straight-Through (ST) Gumbel-Softmax estimator. Our simulation results on the SQuAD dataset and NewsQA dataset show that our model outperforms a range of existing state-of-the-art approaches significantly.

REFERENCES

[1] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261* (2018).

[2] Yllias Chali and Sadid A Hasan. 2015. Towards topic-to-question generation. *Computational Linguistics* 41, 1 (2015), 1–20.

[3] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[4] Guy Danon and Mark Last. 2017. A Syntactic Approach to Domain-Specific Automatic Question Generation. *arXiv preprint arXiv:1712.09827* (2017).

[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*. 3844–3852.

[6] Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*. 376–380.

[7] Xinya Du and Claire Cardie. 2018. Harvesting Paragraph-Level Question-Answer Pairs from Wikipedia. *arXiv preprint arXiv:1805.05942* (2018).

[8] Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106* (2017).

[9] Yifan Gao, Jianan Wang, Lidong Bing, Irwin King, and Michael R Lyu. 2018. Difficulty Controllable Question Generation for Reading Comprehension. *arXiv preprint arXiv:1807.03586* (2018).

[10] Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Maxout networks. *arXiv preprint arXiv:1302.4389* (2013).

[11] Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. *arXiv preprint arXiv:1603.08148* (2016).

[12] Michael Heilman. 2011. Automatic factual question generation from text. *Language Technologies Institute School of Computer Science Carnegie Mellon University* 195 (2011).

[13] Michael Heilman and Noah A Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 609–617.

[14] Wenpeng Hu, Bing Liu, Jinwen Ma, Dongyan Zhao, and Rui Yan. 2018. Aspect-based Question Generation. (2018).

[15] Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016).

[16] Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2018. Improving Neural Question Generation using Answer Separation. *arXiv preprint arXiv:1809.02393* (2018).

[17] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[19] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out* (2004).

[20] Bang Liu, Ting Zhang, Di Niu, Jinghong Lin, Kunfeng Lai, and Yu Xu. 2018. Matching Long Text Documents via Graph Convolutional Networks. *arXiv preprint arXiv:1802.07459* (2018).

[21] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[22] Chris J Maddison, Daniel Tarlow, and Tom Minka. 2014. A* sampling. In *Advances in Neural Information Processing Systems*. 3086–3094.

[23] Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826* (2017).

[24] Matthew Honnibal. 2015. spaCy: Industrial-strength Natural Language Processing (NLP) with Python and Cython. https://spacy.io. [Online; accessed 3-November-2018].

[25] Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. *arXiv preprint arXiv:1603.06059* (2016).

[26] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 311–318.

[27] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. 2017. Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration.

[28] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[29] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* (2016).

[30] Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. *arXiv preprint arXiv:1603.06807* (2016).

[31] Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging Context Information for Natural Question Generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, Vol. 2. 569–574.

[32] Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Yoshua Bengio, and Adam Trischler. 2017. Neural Models for Key Phrase Detection and Question Generation. *arXiv preprint arXiv:1706.04560* (2017).

[33] Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and Position-aware Neural Question Generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 3930–3939.

[34] Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027* (2017).

[35] Duyu Tang, Nan Duan, Zhao Yan, Zhirui Zhang, Yibo Sun, Shujie Liu, Yuanhua Lv, and Ming Zhou. 2018. Learning to Collaborate for Question Answering and Asking. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Vol. 1. 1564–1574.

[36] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830* (2016).

[37] Kaichun Yao, Libo Zhang, Tiejian Luo, Lili Tao, and Yanjun Wu. 2018. Teaching Machines to Ask Questions.. In *IJCAI*. 4546–4552.

[38] Xingdi Yuan, Tong Wang, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Sandeep Subramanian, Saizheng Zhang, and Adam Trischler. 2017. Machine comprehension by text-to-text neural question generation. *arXiv preprint arXiv:1705.02012* (2017).

[39] Ting Zhang, Bang Liu, Di Niu, Kunfeng Lai, and Yu Xu. 2018. Multiresolution Graph Attention Networks for Relevance Matching. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 933–942.

[40] Yuhao Zhang, Peng Qi, and Christopher D Manning. 2018. Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. *arXiv preprint arXiv:1809.10185* (2018).

[41] Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level Neural Question Generation with Maxout Pointer and Gated Self-attention Networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 3901–3910.

[42] Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*. Springer, 662–671.

[43] Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2018. Sequential Copying Networks. *arXiv preprint arXiv:1807.02301* (2018).

Feature Generation by Convolutional Neural Network for Click-Through Rate Prediction

Bin Liu
Noah's Ark Lab, Huawei, Shenzhen, China
liubin131@huawei.com

Ruiming Tang*
Noah's Ark Lab, Huawei, Shenzhen, China
tangruiming@huawei.com

Yingzhi Chen†
Jinan University, Guangzhou, China
chen_yingzhi@163.com

Jinkai Yu
Noah's Ark Lab, Huawei, Shenzhen, China
yujinkai@huawei.com

Huifeng Guo
Noah's Ark Lab, Huawei, Shenzhen, China
huifeng.guo@huawei.com

Yuzhou Zhang
Noah's Ark Lab, Huawei, Shenzhen, China
zhangyuzhou3@huawei.com

ABSTRACT

Click-Through Rate prediction is an important task in recommender systems, which aims to estimate the probability of a user to click on a given item. Recently, many deep models have been proposed to learn low-order and high-order feature interactions from original features. However, since useful interactions are always sparse, it is difficult for DNN to learn them effectively under a large number of parameters. In real scenarios, artificial features are able to improve the performance of deep models (such as Wide & Deep Learning), but feature engineering is expensive and requires domain knowledge, making it impractical in different scenarios. Therefore, it is necessary to augment feature space automatically.In this paper, We propose a novel Feature Generation by Convolutional Neural Network (FGCNN) model with two components: *Feature Generation* and *Deep Classifier*. *Feature Generation* leverages the strength of CNN to generate local patterns and recombine them to generate new features. *Deep Classifier* adopts the structure of IPNN to learn interactions from the augmented feature space. Experimental results on three large-scale datasets show that FGCNN significantly outperforms nine state-of-the-art models. Moreover, when applying some state-of-the-art models as *Deep Classifier*, better performance is always achieved, showing the great compatibility of our FGCNN model. This work explores a novel direction for CTR predictions: it is quite useful to reduce the learning difficulties of DNN by automatically identifying important features.

KEYWORDS

CNN, Click-Through Rate Prediction, Feature Generation

ACM Reference Format:
Bin Liu, Ruiming Tang, Yingzhi Chen, Jinkai Yu, Huifeng Guo, and Yuzhou Zhang. 2019. Feature Generation by Convolutional Neural Network for Click-Through Rate Prediction. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313497

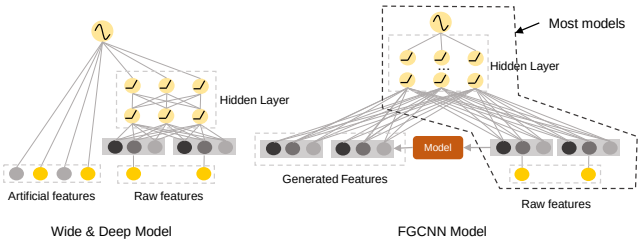


Figure 1: Comparision Between our Model and Wide & Deep Model

1 INTRODUCTION

Click-Through Rate (CTR) is a crucial task for recommender systems, which estimates the probability of a user to click on a given item [9, 23]. In an online advertising application, which is a billion-dollar scenario, the ranking strategy of candidate advertisements is by CTR×bid where "bid" is the profit that the system receives once the advertisement is clicked on. In such applications, the performance of CTR prediction models [9, 23, 40] is one of the core factors determining system's revenue.

The key challenge for CTR prediction tasks is to effectively model feature interactions. Generalized linear models, such as FTRL [21], perform well in practice, but these models lack the ability to learn feature interactions. To overcome the limitation, Factorization Machine [26] and its variants [16] are proposed to model pairwise feature interactions as the inner product of latent vectors and show promising results. Recently, deep neural networks (DNN) have achieved remarkable progress in computer vision [11, 32] and natural language processing [1, 33]. And some deep learning models have been proposed for CTR predictions, such as PIN [23], xDeepFM [19] and etc. Such models feed raw features to a deep neural network to learn feature interactions explicitly or implicitly. Theoretically, DNN is able to learn arbitrary feature interactions from the raw features. However, due to that useful interactions are ususally sparse compared with the combination space of raw features, it is of great difficulties to learn them effectively from a large number of parameters [23, 29].

*Corresponding author
†This work was done when Yingzhi Chen was an intern in Huawei.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313497

Observing such difficulties, Wide & Deep Learning [7] leverages feature engineering in the *wide* component to help the learning of *deep* component. With the help of artificial features, the performance of deep component is improved significantly (0.6% improvement on offline AUC and 1% improvement on online CTR). However, feature engineering can be expensive and requires domain knowledge. If we can generate sophisticated feature interactions automatically by machine learning models, it will be more practical and robust.

Therefore, as shown in Figure 1, we propose a general framework for automatical feature generation. Raw features are input into a machine learning model (represented by the red box in Figure 1) to identify and generate new features [1]. After that, the raw features and the generated new features are combined and fed into a deep neural network. The generated features are able to reduce the learning difficulties of deep models by capturing the sparse but important feature interactions in advance.

The most straightforward way for automatical feature generation is to perform Multi-Layer Perceptron (MLP[2]) and use the hidden neurons as the generated features. However, as mentioned above, due to that useful feature interactions are usually sparse [23], it is rather difficult for MLP to learn such interactions from a huge parameter space. For example, suppose we have four user features: *Name*, *Age*, *Height*, *Gender* to predict whether a user will download an online game. Assume that the feature interaction between *Age* and *Gender* is the only signal that matters, so that an optimal model should identify this and only this feature interaction. When performing MLP with only one hidden layer, the optimal weights associated to the embeddings of *Name* and *Height* should be all 0's, which is fairly difficult to achieve.

As an advanced neural network structure, Convolutional Neural Network (CNN) has achieved great success in the area of computer vision [14] and natural language processing [1]. In CNN, the design of shared weights and pooling mechanism greatly reduces the number of parameters needed to find important local patterns and it will alleviate the optimization difficulties of later MLP structures. Therefore, CNN provides a potentially good solution to realize our idea (identify the sparse but important feature interactions). However, applying CNN directly could result in unsatisfactory performance. In CTR prediction, different arrange orders of original features do not have different meanings. For example, whether the arrangement order of features being *(Name, Age, Height, Gender)* or *(Age, Name, Height, Gender)* does not make any difference to describe the semantics of a sample, which is completely different from the case of images and sentences. If we only use the neighbor patterns extracted by CNN, many useful global feature interactions will be lost. This is also why CNN models do not perform well for CTR prediction task. To overcome this limitation, we perform CNN and MLP, which complement each other, to learn global-local feature interactions for feature generation.

In this paper, we propose a new model for CTR prediction task, namely Feature Generation by Convolutional Neural Network (FGCNN), which consists of two components: *Feature Generation* and *Deep Classifier*. In *Feature Generation*, a CNN+MLP structure is designed to identify and generate new important features from raw features. More specifically, CNN is performed to learn neighbor feature interactions, while MLP is applied to recombine them to extract global feature interactions. After *Feature Generation*, the feature space can be augmented by combining raw features and new features. In *Deep Classifier*, almost all state-of-the-art network structures (such as PIN [23], xDeepFM [19], DeepFM [9]) can be adopted. Therefore, our model has good compatibility with the state-of-the-art models in recommender systems. For the ease of illustration, we will adopt IPNN model [22, 23] as *Deep Classifier* in FGCNN, due to its good trade-off between model complexity and accuracy. Experimental results in three large-scale datasets show that FGCNN significantly outperforms nine state-of-the-art models, demonstrating the effectiveness of FGCNN. While adopting other models in *Deep Classifier*, better performance is always achieved, which shows the usefulness of the generated features. Step-by-step analyses show that each component in FGCNN contributes to the final performance. Compared with traditional CNN structure, our CNN+MLP structure performs better and more stable when the order of raw features changes, which demonstrates the robustness of FGCNN.

To summarize, the main contributions of this paper can be highlighted as follows:

- An important direction is identified for CTR prediction: it is both necessary and useful to reduce the optimization difficulties of deep learning models by automatically generating important features in advance.
- We propose a new model-FGCNN for automatical feature generation and classification, which consists of two components: *Feature Generation* and *Deep Classifier*. *Feature Generation* leverages CNN and MLP, which complement each other, to identify the important but sparse features. Moreover, almost all other CTR models can be applied in *Deep Classifier* to learn and predict based on the generated and raw features.
- Experiments on three large-scale datasets demonstrate the overall effectiveness of FGCNN model. When the generated features are used in other models, better performance is always achieved, which shows the great compatibility and robustness of our FGCNN model.

The rest of this paper is organized as follows: Section 2 presents the proposed FGCNN model in detail. Experimental results will be shown and discussed in Section 3. Related works will be introduced in Section 4. Section 5 concludes the paper.

2 FEATURE GENERATION BY CONVOLUTIONAL NEURAL NETWORK MODEL

2.1 Overview

In this section, we will describe the proposed Feature Generation by Convolutional Neural Network (FGCNN) model in detail. The used notations are summarized in Table 1.

As shown in Figure 2, FGCNN model consists of two components: *Feature Generation* and *Deep Classifier*. More specifically, *Feature Generation* focuses on identifying useful local and global patterns to generate new features as a complement to raw features, while *Deep Classifier* learns and predicts based on the augmented feature

[1]Here, new features are the feature interactions of the raw features. In the rest of this paper, we may use the term "new features" for the ease of presentation.
[2]MLP is a neural network with several fully connected layers.

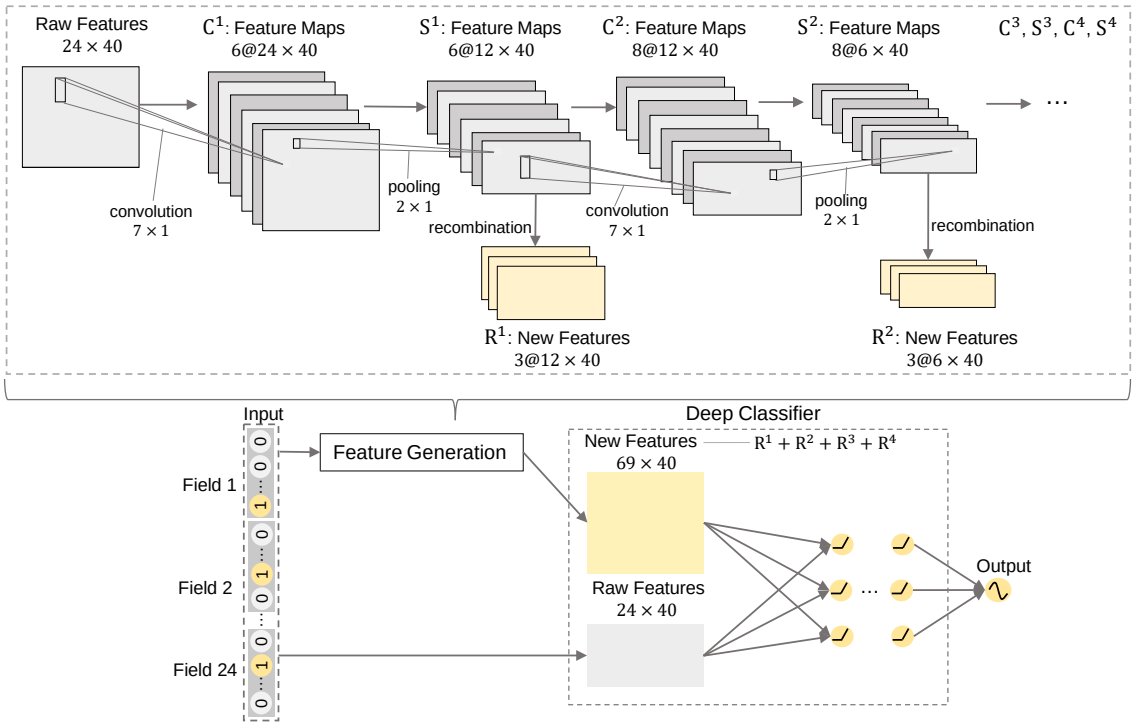


Figure 2: An overview of Feature Generation by Convolutional Neural Network Model (The hyper-parameters in the figure are the best setting of FGCNN on Avazu Dataset)

space through a deep learning model. Besides the two components, we will also formalize *Feature Embedding* of our model. The details of these components are presented in the following subsections.

2.2 Feature Embedding

In most CTR prediction tasks, data is collected in a multi-field categorical form[3] [30, 38], so that each data instance is normally transformed into a high-dimensional sparse (binary) vector via one-hot encoding [13]. For example, (Gender=Male, Height=175, Age=18, Name=Bob) can be represented as:

$$\underbrace{(0,1)}_{\text{Gender=Male}}\ \underbrace{(0,...,1,0,0)}_{\text{Height=175}}\ \underbrace{(0,1,...,0,0)}_{\text{Age=18}}\ \underbrace{(1,0,0,..,0)}_{\text{Name=Bob}} \quad (1)$$

An embedding layer is applied upon the raw features' input to compress them to low-dimensional vectors. In our model, if a field is univalent (e.g., "Gender=Male"), its embedding is the feature embedding of the field; if a field is multivalent (e.g., "Interest=Football, Basketball"), the embedding of the field takes the sum of features' embeddings [8].

More formally, in an instance, each field i ($1 \le i \le n_f$) is represented as a low-dimensional vector $e_i \in R^{1\times k}$, where n_f is the number of fields and k is embedding size. Therefore, each instance can be represented as an embedding matrix $E = (e_1^T, e_2^T, ..., e_{n_f}^T)^T$, where $E \in R^{n_f \times k}$. In FGCNN model, the embedding matrix E can be utilized in both *Feature Generation* and *Deep Classifier*. To avoid the inconsistency of gradient direction when updating parameters, we will introduce another embedding matrix $E' \in R^{n_f \times k}$ for *Deep Classifier* while E is used in *Feature Generation*.

[3]Features in numerical form are usually transformed into categorical form by bucketing.

Table 1: Notations

Parameter	Meaning
k	embedding size
t_f	the total number of one-hot features
n_f	the number of fields
n_c	the number of convolutional layers
n_h	the number of hidden layers
h^i	the height of convolutional kernel in the i-th convolutional layer
h_p	pooling height (width=1) of pooling layers
e_i	the embedding vector for i-th field
E^i	the input of the i-th convolutional layer
m_c^i	the number of feature maps in the i-th convolutional layer
m_r^i	the number of new features' map in the i-th recombination layer
N_i	the number of generated features in the i-th recombination layer ($=n_f/h_p^i m_r^i$)
$C_{:,:,j}^i$	the j-th output feature map of the i-th convolutional layer
$\mathbb{WC}_{:,:,j}^i$	weights for the j-th output feature map of i-th convolutional layer
$S_{:,:,j}^i$	the j-th output feature map of the i-th pooling layer
$\mathbb{WR}^i$	weights for the i-th recombination layer
$\mathbb{BR}^i$	bias for the i-th recombination layer
$R_{:,:,j}^i$	the j-th output map of new features in the i-th recombination layer
W^i	weights for the i-th hidden layer
B^i	biases for the i-th hidden layer
H_i	number of hidden neorons in the i-th hidden layer

2.3 Feature Generation

As stated in Section 1, generating new features from raw features helps improve the performance of deep learning models (as demonstrated by Wide & Deep Learning [7]). To achieve this goal, *Feature Generation* designs a proper neural network structure to identify useful feature interactions then generate new features automatically. As argued in Section 1, using MLP or CNN alone is not able to generate effective feature interactions from raw features, due to the following reasons: *Firstly*, useful feature interactions are always

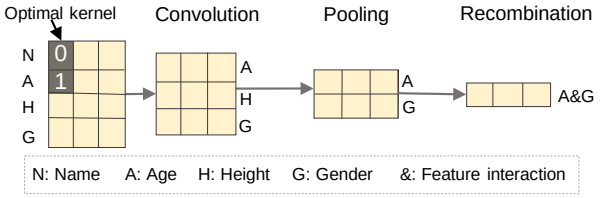


Figure 3: CNN+Recombination structure is able to capture global non-neighbor feature interactions to generate new features. CNN consists of convolutional layer and pooling layer, while Recombination consists of a fully connected layer.

sparse in the combination space of raw features. Therefore, it is difficult for MLP to learn them from a large amount of parameters. *Secondly*, although CNN can alleviates optimization difficulties of MLP by reducing the number of paramters, it only generates neighbor feature interactions which can lose many useful global feature interactions.

In order to overcome the weakness of applying MLP or CNN alone, as shown in the upper part of Figure 2, we perform CNN and MLP [4] as a complement to each other for feature generation. Figure 3 shows an example of CNN+Recombination structure to capture global feature interactions. As can be observed, CNN learns useful neighbor feature patterns with a limited number of parameters, while recombination layer (which is a fully connected layer) generates global feature interactions based on the neighbor patterns provided by CNN. Therefore, important features can be generated effectively via this neural network structure, which has fewer parameters than directly applying MLP for feature generation.

In the following parts, we detail the CNN+Recombination structure of *Feature Generation*, namely, *Convolutional Layer*, *Pooling layer* and *Recombination layer*.

2.3.1 Convolutional Layer. Each instance is represented as an embedding matrix $E \in R^{n_f \times k}$ via feature embedding, where n_f is the number of fields and k is embedding size. For convenience, reshape the embedding matrix as $E^1 \in R^{n_f \times k \times 1}$ as the input matrix of the first convolutional layer. To capture the neighbor feature interactions, a convolutional layer is obtained by convolving a matrix $\mathbb{WC}^1 \in R^{h^1 \times 1 \times 1 \times m_c^1}$ with non-linear activation functions (where h^1 is the height of the first convolutional weight matrix and m_c^1 is the number of feature maps in the first convolutional layer). Suppose the output of the first convolutional layer is denoted as $C^1 \in R^{n_f \times k \times m_c^1}$, we can formulate the convolutional layer as follows:

$$C^1_{p,q,i} = tanh(\sum_{m=1}^{1} \sum_{j=1}^{h^1} E^1_{p+j-1,q,m} \mathbb{WC}^1_{j,1,1,i}) \tag{2}$$

$$tanh(x) = \frac{exp(x) - exp(-x)}{exp(x) + exp(-x)} \tag{3}$$

where $C^1_{:,:,i}$ denotes the i-th feature map in the first convolutional layer and p, q are the row and column index of the i-th feature map. Notice that the above equation excludes padding which is performed in practice.

[4] According to its function, we will call MLP with one hidden layer as recombination layer later.

2.3.2 Pooling Layer. After the first convolutional layer, a max-pooling layer is applied to seize the most important feature interactions and reduce the number of parameters. We refer h_p as the height of pooling layers (width=1). The output in the first pooling layer is $S^1 \in R^{(n_f/h_p) \times k \times m_c^1}$:

$$S^1_{p,q,i} = max(C^1_{p \cdot h_p,q,i}, \ \dots, C^1_{p \cdot h_p + h_p - 1,q,i}) \tag{4}$$

The pooling result of the i-th pooling layer will be the input for the $(i+1)$-th convolutional layer: $E^{i+1} = S^i$.

2.3.3 Recombination Layer. After the first convolutional layer and pooling layer, $S^1 \in R^{(n_f/h_p) \times k \times m_c^1}$ contains the patterns of neighbor features. Due to the nature of CNN, global non-neighbor feature interactions will be ignored if S^1 is regarded as the generated new features. Therefore, we introduce a fully connected layer to recombine the local neighbor feature patterns and generate important new features. We denote the weight matrix as $\mathbb{WR}^1 \in R^{(n_f/h_p k m_c^1) \times (n_f/h_p k m_r^1)}$ and the bias as $\mathbb{BR}^1 \in R^{(n_f/h_p k m_r^1)}$, where m_c^1 is the number of feature maps in the first convolutional layer and m_r^1 is the number of new features' map in the first recombination Layer. Therefore, in the i-th recombination layer, $n_f/h_p^i m_r^i$ features are generated:

$$R^1 = tanh(S^1 \cdot \mathbb{WR}^1 + \mathbb{BR}^1) \tag{5}$$

2.3.4 Concatenation. New features can be generated by performing CNN+Recombination multiple times. Assume there are n_c convolutional layers, pooling layers and recombination layers and $N_i = n_f/h_p^i m_r^i$ fields of features are generated by i-th round denoted as R^i. The overall new features $\mathcal{R} \in R^{N \times k}$ (where $N = \sum_{i=1}^{n_c} N_i$) generated by *Feature Generation* are formalized as:

$$\mathcal{R} = (R^1, R^2, ..., R^{n_c}) \tag{6}$$

Then, raw features and new features are concatenated as

$$\mathbb{E} = (E'^T, \mathcal{R}^T)^T \tag{7}$$

where E' is the embedding matrix of raw features for *Deep Classifier* (see Section 2.2). Both the raw features and new features are utilized for CTR prediction in *Deep Classifier*, which will be elaborated in the next subsection.

2.4 Deep Classifier

As mentioned above, raw features and new features are concatenated as an augmented embedding matrix $\mathbb{E} \in R^{(N+n_f) \times k}$, where N and n_f are the number of fields of new features and raw features respectively. $\mathbb{E}$ is input into *Deep Classifier*, which aims to learn further interactions between the raw features and new generated features. In this subsection, for the ease of presentation, we adopt IPNN model [23] as the network structure in *Deep Classifier*, due to its good trade-off between model complexity and accuracy. In fact, any advanced network structure can be adopted, which shows the compatibility of FGCNN with the existing works. The compatibility of FGCNN model will be verified empirically in Section 3.

2.4.1 Network Structure. IPNN model [23] combines the learning ability of FM [27] and MLP. It utilizes an FM layer to extract pairwise feature interactions from embedding vectors by inner product operations. After that, the embeddings of input features and the results of FM layer are concatenated and fed to MLP for learning. As evaluated in [23], the performance of IPNN is slightly worse

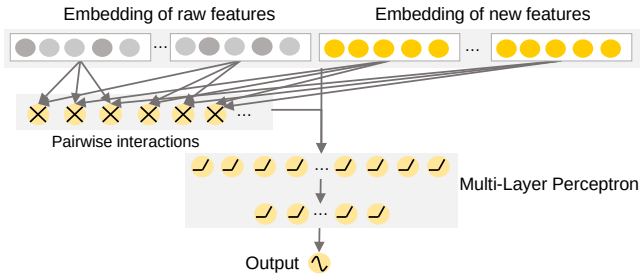


Figure 4: Structure of IPNN Model

than PIN (the best model in [23]), but IPNN is much more efficient. We will illustrate the network structure of IPNN model.

As shown in Figure 4, the pariwise feature interactions of augmented embedding matrix $\mathbb{E} \in R^{(N+n_f)\times k}$ are modeled by an FM layer, as follows:

$$R_{fm} = (< \mathbb{E}_1, \mathbb{E}_2 >, ..., < \mathbb{E}_{N+n_f-1}, \mathbb{E}_{N+n_f} >) \tag{8}$$

where $\mathbb{E}_i$ is the embedding of the i-th field, $< a, b >$ means the inner product of a and b. The number of pairwise feature interactions in the FM layer is $\frac{(N+n_f)(N+n_f-1)}{2}$.

After the FM layer, R_{fm} is concatenated with the augmented embedding matrix $\mathbb{E}$, which are fed into MLP with n_h hidden layers to learn implicit feature interactions. We refer the input for the i-th hidden layer as I_i and the output for the i-th hidden layer as O_i. The MLP is formulated as:

$$I_1 = (R_{fm}, flatten(\mathbb{E})) \tag{9}$$

$$O_i = relu(I_i W^i + B^i) \tag{10}$$

$$I_{i+1} = O_i \tag{11}$$

where W^i and B^i are the weight matrix and bias of the i-th hidden layer in MLP. For the last hidden layer (the n_h-th layer), we will make final predictions:

$$\hat{y} = sigmoid(O_{n_h} W^{n_h+1} + b^{n_h+1}) \tag{12}$$

2.4.2 Batch Normalization. Batch Normalization (BN) is proposed in [15] to solve covariant shift and accelerate DNN training. It normalizes activations $w^T x$ using statistics within a mini-batch, as follows:

$$BN(w^T x) = \frac{w^T x - avg_i(w^T x)}{std_i(w^T x)} g + b \tag{13}$$

where g and b scale and shift the normalized values.

In the FGCNN model, Batch Normalization is applied before each activation function to accelerate model training.

2.4.3 Objective Function. The objective function in the FGCNN model is to minimize cross entropy of predicted values and the labels, which is defined as:

$$\mathcal{L}(y, \hat{y}) = -ylog\hat{y} - (1-y)log(1-\hat{y}) \tag{14}$$

where $y \in \{0, 1\}$ is the label and $\hat{y} \in (0, 1)$ is the predicted probability of $y = 1$.

2.5 Complexity Analysis

In this section, we analyse the space and time complexity of FGCNN model. Notations in Table 1 will be reused.

2.5.1 Space Complexity. The space complexity of FGCNN model consists of three parts, namely *Feature Embedding*, *Feature Generation* and *Deep Classifier*. In feature embedding, since there are two matrices (one for *Feature Generation* and one for *Deep Classifier* (as discussed in Section 2.2)), there are $s_0 = 2t_f k$ parameters, where t_f is the total number of one-hot raw features.

In *Feature Generation*, there are $h^i m_c^{i-1} m_c^i$ parameters for convolutional matrix in i-th convolutional layer and $n_f/h_p^i k m_c^i$ neurons in the output of i-th pooling layer. In addition, the number of parameters in i-th recombination layer is $n_f^2/h_p^{2i} k^2 m_c^i m_r^i$, which will generate $N_i = n_f/h_p^i m_r^i$ fields of new features ($n_f/h_p^i = N_i/m_r^i$). Therefore, the total parameters of *Feature Generation* is:

$$\sum_{i=1}^{n_c} h^i m_c^{i-1} m_c^i + n_f^2/h_p^{2i} k^2 m_c^i m_r^i = \tag{15}$$

$$\sum_{i=1}^{n_c} h^i m_c^{i-1} m_c^i + N_i^2 k^2 m_c^i / m_r^i \tag{16}$$

After the *Feature Generation* process, we can get $T = n_f + \sum_{i=1}^{n_c} N_i$ feature embeddings in total. Normally, m_r^i is usually set as $1, 2, 3, 4$ and $m_r^i = m_r^{i-1}$. Meanwhile, h_p is usually set to 2 so that $N_i = N_1/2^{i-1}$. Furthermore, $N_i^2 k^2 >> h_i m_c^{i-1}$ and $\sum_{i=1}^{n_c} m_c^i / m_r^i 2^{2(i-1)}$ is usually a small number. The space complexity s_1 of the feature generating process can be simplified as:

$$s_1 = O(\sum_{i=1}^{n_c} N_1^2/2^{2(i-1)} k^2 m_c^i/m_r^i) = O(N_1^2 k^2) \tag{17}$$

For the first hidden layer of *Deep Classifier*, the number of parameters in the weight matrix is $(T(T-1)/2 + Tk)H_1$ (recall that the input is the raw and new features' embedding and their pairwised product). In the i-th hidden layer, the number of parameters is $H_{i-1}H_i$. Therefore, the space complexity of *Deep Classifier* is:

$$s_2 = O(\frac{T(T-1) + 2Tk}{2} H_1 + \sum_{i=2}^{n_h} H_i H_{i-1}) \tag{18}$$

where Eq. (18) can be simplified as $O(T^2 H_1 + \sum_{i=2}^{n_h} H_i H_{i-1})$ since $Tk < T(T-1)$ usually.

In summary, the overall space complexity of FGCNN is $s_0 + s_1 + s_2 = O(t_f k + N_1^2 k^2 + T^2 H_1 + \sum_{i=1}^{q} H_i H_{i-1})$, which is dominated by the number of one-hot features, the number of generated features, the number of hidden neurons and the embedding size.

2.5.2 Time Complexity. We first analyze the time complexity in *Feature Generation*. In the i-th convolutional layer, the output dimension is $n_f/h_p^{i-1} k m_c^i = N_i h_p / m_r^i k m_c^i$ (recall that $N_i = n_f/h_p^i m_r^i$) and each point is calculated from $h^i m_c^{i-1}$ data points where h^i is the height of convolutional kernels in the i-th convolutional layer. Therefore, the time complexity of the i-th convolutional layer is $N_i h_p k m_c^i h^i m_c^{i-1}/m_r^i$. In the pooling layer, the time complexity of the i-th pooling layer is $n_f/h_p^{i-1} k m_c^i = N_i h_p k m_c^i/m_r^i$. In the i-th recombination layer, the time complexity is $n/h_p^i k m_c^i \cdot N_i k = N_i^2 k^2 m_c^i/m_r^i$ (recall $h_p = 2$ and $N_i = N_1/2^{i-1}$). Therefore, total

Table 2: Dataset Statistics

Dataset	#instances	#features	#fields	pos ratio
Criteo	1×10^8	1×10^6	39	0.5
Avazu	4×10^7	6×10^5	24	0.17
Huawei App Store	2.3×10^8	1.6×10^5	29	0.05

time complexity of the *Feature Generation* is

$$\begin{aligned} t_1 &= \sum_{i=1}^{n_c} N_i h_p k m_c^i h^i m_c^{i-1}/m_r^i + N_i h_p k m_c^i/m_r^i + N_i^2 k^2 m_c^i/m_r^i \\ &= \sum_{i=1}^{n_c} N_1 k m_c^i/(m_r^i 2^{i-1})(2h^i m_c^{i-1} + 2 + N_1/2^{i-1}k) \\ &= O(N_1 k \sum_{i=1}^{n_c} m_c^{i-1} h^i + N_1^2 k^2) \end{aligned}$$

(Notice: $m_c^i/(m_r^i 2^i)$ is usually a small number ranging from 2 to 10)

Then, we analyze the time complexity of *Deep Classifier* (taking IPNN as an example). In the FM layer, the time complexity to calculate pairwise inner product is $\frac{T(T-1)k}{2}$. The number of neurons, that are input to the first hidden layer is $T(T-1)/2 + Tk$. Therefore, the time complexity of the first hidden layer is $O((T(T-1)/2 + Tk)H_1) = O(T^2 H_1)$. For the other hidden layers, the time complexity is $O(H_i H_{i-1})$. Therefore, the total time complexity in *Deep Classifier* is

$$t_2 = O(T^2 H_1) + \sum_{i=2}^{n_h} O(H_i H_{i-1}) \tag{19}$$

In summary, the overall time complexity for FGCNN model is $t_1 + t_2 = O(N_1 k \sum_{i=1}^{n_c} m_c^{i-1} h^i + N_1^2 k^2 + T^2 H_1 + \sum_{i=2}^{q} H_i H_{i-1})$ which is dominated by the number of generated features, the number of neurons in the hidden layers, embedding size and convolutional parameters.

3 EXPERIMENTS

In this section, we conduct extensive experiments to answer the following questions.

- **(Q1)** How does FGCNN perform, compared to the state-of-the-art models for CTR prediction task?
- **(Q2)** Can FGCNN improve the performance of other state-of-the-art models by using their structures in *Deep Classifier*?
- **(Q3)** How does each key structure in FGCNN boost the performance of FGCNN?
- **(Q4)** How do the key hyper-parameters of FGCNN (i.e., size of convolutional kernels,number of convolutional layers, number of generated features) impact its performance?
- **(Q5)** How does *Feature Generation* perform when the order of raw features are randomly shuffled?

Note that when studying questions other than **Q2**, we adpot IPNN as the *Deep Classifier* in FGCNN.

3.1 Experimental Setup

3.1.1 Datasets. Experiments are conducted in the following three datasets:

Table 3: Parameter Settings

Param	Criteo	Avazu	Huawei App Store
General	bs=2000 opt=Adam lr=1e-3	bs=2000 opt=Adam lr=1e-3	bs=1000 opt=Adam lr=1e-4 l2=1e-6
LR	–	–	–
GBDT	depth=25 #tree=1300	depth=18 #tree=1000	depth=10 #tree=1400
FM	k=20	k=40	k=40
FFM	k=4	k=4	k=12
CCPM	k=20 conv= 7*1 kernel=[256] net=[256*3,1]	k=40 conv: 7*1 kernel = [128] net= [128*3,1]	k=40 conv= 13*1 kernel= [8,16,32,64] net=[512,256,128,1] drop= 0.8
DeepFM	k=20 LN=T net=[700*5,1]	k=40 LN=T net=[500*5,1]	k=40 net=[2048,1024,512,1] drop=0.9
XdeepFM	k=20 net=[400*3,1] CIN=[100*4]	k=40 net=[700*5,1] CIN:[100*2]	k=40 net=[2048,1024,512,1] drop=0.9 CIN:[100*4]
IPNN	k=20 LN=T net=[700*5,1]	k=40 LN=T net=[500*5,1]	k=40 net=[2048,1024, 512,256,128,1] drop=0.7
PIN	k=20 LN=T net=[700*5,1] subnet=[40,5]	k=40 LN=T net=[500*5,1] sub-net=[40,5]	k=40 net=[2048,1024,512,1] drop=0.9 sub-net=[80,10]
FGCNN	k=20 conv=9*1 kernel=[38,40,42,44] new=[3,3,3,3] BN=T ruenet=[4096,2048,1]	k=40 conv=7*1 kernel=[14,16,18,20] new=[3,3,3,3] BN=T net=[4096,2048, 1024,512,1]	k=40 conv=13*1 kernel=[6, 8, 10, 12] new=[2,2,2,2] BN=T net=[2048,1024, 512,256,128,1] drop=0.8

Note: bs=batch size, opt=optimizer, lr=learning rate, l2=l2 regularization on Embedding Layer, k=embedding size, conv=shape of convolutional kernels, kernel=number of convolutional kernels, pool=max-pooling shape, net=MLP structure, sub-net=micro metwork, drop=dropout rate, LN=layer normalization, BN= batch normalization{T=True}, new=number of kernels for new features.

Criteo: Criteo[5] contains one month of click logs with billions of data samples. A small subset of Criteo was published in Criteo Display Advertising Challenge 2013 and FFM was the winning solution [16, 23]. We select "day 6-12" as training set while select "day 13" for evaluation. Due to the enormous data volume and serious class imbalance (i.e., only 3% samples are positive), negative sampling is applied to keep the positive and negative ratio close to 1:1. We convert 13 numerical fields into one-hot features through bucketing, where the features in a certain field appearing less than 20 times are set as a dummy feature "other".

Avazu: Avazu[6] was published in the contest of Avazu Click-Through Rate Prediction, 2014. The public dataset is randomly splitted into training and test sets at 4:1. Meanwhile, we remove the features appearing less than 20 times to reduce dimensionality.

Huawei App Store: In order to evaluate the performance in industrial CTR prediction problems, we conduct experiments on Huawei App Store Dataset. We collect users' click logs from Huawei App Store while logs from 20180617 to 20180623 are used for training and logs of 20180624 are used for test. Negative sampling is applied to reduce data amount and to adjust the ratio of positive class and negative class. The dataset contains app features (e.g., identification, category), user features (e.g., user's behavior history) and context features (e.g., operation time).

In addition, the statistics of the three datasets are summarized in Table 2.

[5]http://labs.criteo.com/downloads/download-terabyte-click-logs/
[6]http://www.kaggle.com/c/avazu-ctr-prediction

Table 4: Overall Performance
$\star : p < 10^{-2}$ $\star\star : p < 10^{-4}$ **(two tailed t-test)**

	Criteo		Avazu		Huawei App Store	
Model	AUC	Log Loss	AUC(%)	Log Loss	AUC	Log Loss
LR	78.00%	0.5631	76.76%	0.3868	90.12%	0.1371
GBDT	78.62%	0.5560	77.53%	0.3824	92.68%	0.1227
FM	79.09%	0.5500	77.93%	0.3805	93.26%	0.1191
FFM	79.80%	0.5438	78.31%	0.3781	93.58%	0.1170
CCPM	79.55%	0.5469	78.12%	0.3800	93.71%	0.1159
DeepFM	79.91%	0.5423	78.36%	0.3777	93.91%	0.1145
xDeepFM	80.06%	0.5408	78.55%	0.3766	93.91%	0.1146
IPNN	80.13%	0.5399	78.68%	0.3757	93.95%	0.1143
PIN	80.18%[11]	0.5393	78.72%	0.3755	93.91%	0.1146
FGCNN	**80.22%***	**0.5388***	**78.83%****	**0.3746****	**94.07%****	**0.1134****

3.1.2 Baselines. We compare nine baseline models in our experiments, including LR [17], GBDT [6], FM [26], FFM [16], CCPM [20], DeepFM [9], xDeepFM [19], IPNN and PIN [23]. Wide & Deep is not compared here because some state-of-the-art models (such as xDeepFM, DeepFM, PIN) have shown better performance in their publications. We use XGBoost [6] and libFFM[7] as the implementation of GBDT and libFFM, respectively. In our experiments, the other baseline models are implemented with Tensorflow[8].

3.1.3 Evaluation Metrics. The evaluation metrics are **AUC** (Area Under ROC) and **log loss** (cross entropy).

3.1.4 Parameter Settings. Table 3 summerizes the hyper-parameters of each model. For *Criteo* and *Avazu* Datasets, the hyper-parameters of baseline models are set to be the same as in [23]. Notice that when conducting experiments on Criteo and Avazu, we observed that FGCNN uses more parameters in *Deep Classifier* than other models. To make fair comparisons, we also conduct experiments to increase the parameters in MLPs of other deep models. However, all these models cannot achieve better performance than the original settings[9]. The reason could be the overfitting problem where such models simply use embedding of raw features for training but use a complex structure. On the other hand, since our model augments the feature space and enriches the input, more parameters in *Deep Classifier* can boost the performance of our model.

In FGCNN model, *new* is the number of kernels when generating new features. The number of generated features can be calculated as $\sum_{i=1}^{n_c} \#fields/2^i * new_i$.

3.1.5 Significance Test. We repeat the experiments 10 times by changing the random seed for FGCNN and the best baseline model. The two-tailed pairwise t-test is performed to detect significant differences between FGCNN and the best baseline model.

3.2 Overall Performance (Q1)

In this subsection, we compare the performance of different models on the test set. Table 4 summarizes the overall performance of all the compared models on the three datasets, where the underlined numbers are the best results of the baseline models and bold numbers are the best results of all models. We have the following observations:

Firstly, in majority of the cases, non-neural network models perform worse than neural network models. The reason is that deep neural network can learn complex feature interactions much better than the models where no feature interaction is modeled (i.e., LR), or feature interactions are modeled by simple inner product operations (i.e., FM and FFM).

Secondly, FGCNN achieves the best performance among all the models on the three evaluated datasets. It is significantly better than the best baseline models with 0.05%, 0.14% and 0.13% improvements in AUC (0.09%, 0.24% and 0.79% in log loss) on Criteo, Avazu and Huawei App Store datasets, which demonstrates the effectiveness of FGCNN. In fact, a small improvement in offline AUC is likely to lead to a significant increase in online CTR. As reported in [7], compared with LR, Wide & Deep improves offline AUC by 0.275% and the improvement of online CTR is 3.9%. The daily turnover of Huawei App Store is millions of dollars. Therefore, even a few lifts in CTR brings extra millions of dollars each year.

Thirdly, with the help of the generated new features, FGCNN outperforms IPNN by 0.11%, 0.19% and 0.13% in terms of AUC (0.2%, 0.29% and 0.79% in terms of log loss) on Criteo, Avazu and Huawei App Store datasets. It demonstrates that the generated features are very useful and they can effectively reduce the optimization difficulties of traditional DNNs thus leading to better performance.

Fourthly, CCPM, which applies CNN directly, achieves the worst performance among neural network models. Moreover, CCPM performs worse than FFM on Criteo and Avazu datasets. It shows that directly using traditional CNN for CTR prediction task is inadvisable, as CNN is designed to generate neighbor patterns while the arrangement order of feature is usually no meaning in recommendation scenarios. However, in FGCNN, we leverage the strength of CNN to extract local patterns while complementing it with Recombination Layer to extract global feature interactions and generate new features. Therefore, better performance is achieved.

[7] https://github.com/guestwalk/libffm
[8] https://www.tensorflow.org/
[9] Due to the limited pages, we do not show the experimental result in the paper.

[11] According to the experimental recordings, the results of PIN in [23] uses the trick of adaptive embedding size. Here, we use fixed embedding size for all the deep models.

Table 5: Compatibility Study of FGCNN

	Criteo		Avazu		Huawei App Store	
	AUC	Log Loss	AUC	Log Loss	AUC	Log Loss
FM	79.09%	0.5500	77.93%	0.3805	93.26%	0.1191
FGCNN+FM	**79.67%**	**0.5455**	**78.13%**	**0.3794**	**93.66%**	**0.1165**
DNN	79.87%	0.5428	78.30%	0.3778	93.85%	0.1149
FGCNN+DNN	**80.09%**	**0.5402**	**78.55%**	**0.3764**	**94.00%**	**0.1139**
DeepFM	79.91%	0.5423	78.36%	0.3777	93.91%	0.1145
FGCNN+DeepFM	**79.94%**	**0.5421**	**78.44%**	**0.3771**	**93.93%**	**0.1145**
IPNN	80.13%	0.5399	78.68%	0.3757	93.95%	0.1143
FGCNN+IPNN	**80.22%**	**0.5388**	**78.83%**	**0.3746**	**94.07%**	**0.1134**

3.3 Compatibility of FGCNN with Different Models (Q2)

As stated in Section 2.4, *Feature Generation* can augment the original feature space and *Deep Classifier* of FGCNN can adopt any advanced deep neural networks. Therefore, we select several models as *Deep Classifeir* to verify the utility of *Feature Generation*, including non-deep models (FM), deep learning models (DNN, DeepFM, IPNN).

Table 5 summarizes the performance. We have the following observations: Firstly, with the help of the generated new features, the performance of all models are improved, which demonstrates the effectiveness of the generated features and shows the compatibility of FGCNN. Secondly, we observe that when only using raw features, DeepFM always outperforms DNN. But when using the augmented features, FGCNN+DNN outperforms FGCNN+DeepFM. The possible reason is that DeepFM sums up the inner product of input features to the last MLP layer which may cause contradictory gradient updates (compared with MLP) on embeddings.This could be one of the reasons why IPNN (feeding the product into MLP) outperforms DeepFM in all datasets.

In a word, the results show that our FGCNN model can be viewed as a general framework to enhance the existing neural networks by generating new features automatically.

3.4 Effectiveness of FGCNN Variants(Q3)

We conduct experiments to study how each component in FGCNN contributes to the final performance. Each variant is generated by removing or replacing some components in FGCNN, which is described as follows:

- **Removing Raw Features**: In this variant, raw features are not input into *Deep Classifier* and only the generated new features are fed to *Deep Classifier*.
- **Removing New Features**: This variant removes *Feature Generation*. Actually, it is equivalent to IPNN.
- **Applying MLP for *Feature Generation***: *Feature Generation* is replaced by MLP which takes the neurons in each layer as new features. This variant uses the same hidden layers and generates the same number of features in each layer as FGCNN.
- **Removing Recombination Layer**: This variant is to evaluate how *Recombination Layer* complements CNN to capture global feature interactions. The *Recombination Layer* is removed from *Feature Generation* so that the output of pooling layer serves as new features directly. The number of generated new features in each layer keeps the same as FGCNN.

As shown in Table 6, removing any component in FGCNN leads to a drop in performance. We have the following observations:

Firstly, FGCNN with raw features alone or with new generated features alone performs worse than the FGCNN with both of them. This result demonstrates that the generated features are good supplementaries to the original features, which are both crucial.

Secondly, the performance decrease of *Appling MLP for Feature Generation*, compared to FGCNN, shows the ineffectiveness of MLP to identify the sparse but important feature combinations from a large number of parameters. CNN simplifies the learning difficulties by using the shared convolutional kernels, which has much fewer parameters to get the desired combinations. Moreover, MLP recombines the neighbor feature interactions, which is extracted by CNN, to generate global feature interactions.

Thirdly, removing the Recombination Layer will limit the generated features as neighbor feature interactions. Since the arrangement order of raw features has no actual meanings in the CTR prediction task, the restriction can lead to losing important non-neighbor feature interactions thus resulting in worse performance.

3.5 Hyper-parameter Investigation (Q4)

Our FGCNN model has several key hyper-parameters, i.e., the height of convolutional kernels, number of convolutional kernels, number of convolutional layers, and the number of kernels for generating new features. In this subsection, to study the impact of these hyper-parameters, we investigate how FGCNN model works by changing one hyper-parameter while fixing the others on Criteo and Huawei App Store datasets.

3.5.1 Height of Convolutional Kernels. The height of convolutional kernels controls the perception range of convolutional layers. The larger the height is, the more features are involved in the neighbor patterns, but more parameters need to be optimized. To investigate its impact, we increase the height from 2 to the number of fields of a dataset. As shown in the top of Figure 5, the performance generally ascends first and then descends as the height of convolutional kernels increases[10].

The results show that as more features are involved in the convolutional kernels, higher-order feature interactions can be learned

[10]Due to that *Feature Generation* and *Deep Classifier* interrelate with each other, the curve has some fluctuations.

Table 6: Performance of Different FGCNN Variants

	Criteo		Avazu		Huawei App Store	
Method	AUC	Log Loss	AUC	Log Loss	AUC	Log Loss
FGCNN	**80.22%**	**0.5388**	**78.83%**	**0.3746**	**94.07%**	**0.1134**
Removing Raw Features	80.21%	0.5390	78.66%	0.3757	94.01%	0.1138
Removing New Features	80.13%	0.5399	78.68%	0.3757	93.95%	0.1143
Applying MLP for Feature Generation	80.12%	0.5402	78.58%	0.3761	94.04%	0.1135
Removing Recombination Layer	80.11%	0.5403	78.74%	0.3753	94.04%	0.1135

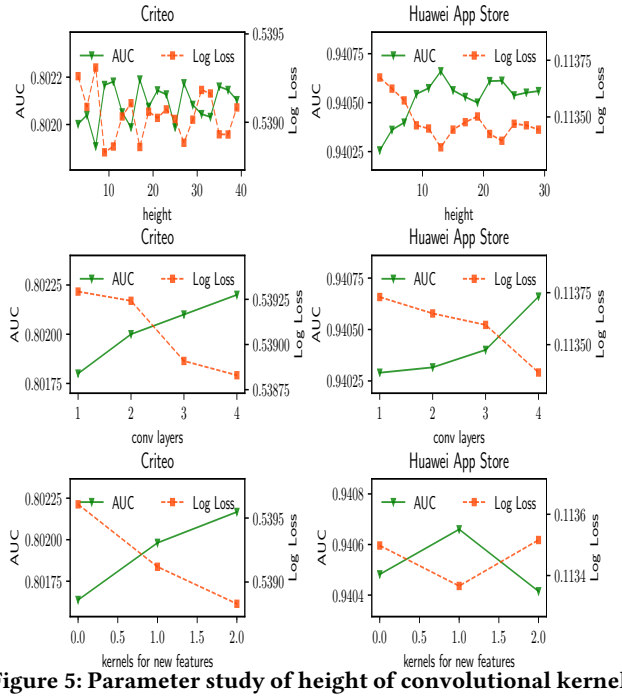


Figure 5: Parameter study of height of convolutional kernel, number of convolutional layers and number of kernels for new features (from top to bottom)

so that the performance increases. However, due to that useful feature interactions are usually sparse, larger heights can cause more difficulties to learn them effectively, leading to a decrease in performance. This observation is consistent with the finding in Section 3.4, i.e., the performance decreases in Appling MLP for Feature Generation.

3.5.2 Number of Convolutional Layers. As shown in the middle of Figure 5, as the number of convolutional layers increases, the performance of FGCNN is improved. Notice that more layers usually lead to higher-order feature interactions. Therefore, the result also shows the effectiveness of high-order feature interactions.

3.5.3 Number of Kernels for Generating New Features. We study how the number of generated features affects the performance of FGCNN. We use the same number of kernels for new features in different *Recombination Layers*. As can be observed in the bottom of Figure 5, the performance is gradually improved with more features generated. The results verify our research idea that it is useful to identify the sparse but important feature interactions first, which can effectively reduce the learning difficulties of DNNs. However, the useful feature interactions can be sparse and limited. If too many features are generated, the extra new features are noisy which will increase the learning difficulties of MLP, leading to the decrease in the performance.

3.6 Effect of Shuffling Order of Raw Features (Q5)

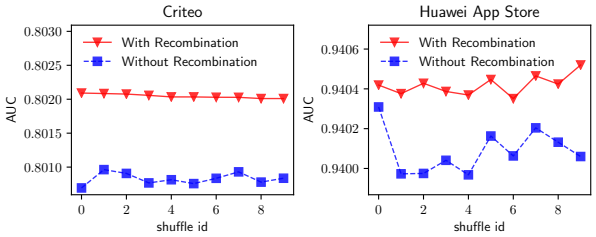


Figure 6: Shuffling the order of raw features

As mentioned before, CNN is designed to capture local neighbor feature patterns so that it is sensitive to the arrangement order of raw features. In our FGCNN model, the design of Recombination Layer is to learn global feature interactions based on CNN's extracted local patterns. Intuitively, our model should have more stable performance than traditional CNN's structure if the order of raw features is shuffled. Therefore, to verify it, we compare the performance of two cases: with/without Recombination Layer. The arrangement order of raw features is shuffled many times at random where the two compared cases are performed for the same shuffled arrangement order.

As shown in Figure 6, the case with Recombination Layer achieves better and more stable performance than that of without Recombination Layer. It demonstrates that with the help of Recombination Layer, FGCNN can greatly reduce the side effects of changing the arrangement order of raw features, which also demonstrates the robustness of our model.

4 RELATED WORK

Click-Through Rate Prediction is normally formulated as a binary classification problem [5, 28]. In this section, we will introduce two important categories of models in Click-Through Rate predictions, namely **shallow models** and **deep learning models**.

4.1 Shallow Models for CTR Prediction

Due to the robustness and efficiency, Logistic Regression (LR) models [18, 25], such as FTRL [21] are widely used in CTR prediction. To learn feature interactions, a common practice is to manually design pairwise feature interactions in its feature space [10, 31]. Poly-2 [4] models all pairwise feature interactions to avoid feature engineering. Factorization Machine (FM) [26] introduces low-dimensional vectors for each feature and models feature interactions through

inner product of feature vectors. FM improves the ability of modelling feature interactions when data is sparse. FFM [16] enables each feature to have multiple latent vectors to perform interactions with features from different fields. LR, Poly-2 and FM variants are widely used in CTR prediction in industry.

4.2 Deep Learning for CTR Prediction

Deep learning has achieved great success in different areas, such as computer vision [11, 14, 32, 36], natural language processing [1, 2, 24, 33] etc. In order to leverage deep Learning for CTR prediction, several models are proposed [34, 37]. FNN [39] is proposed in [39], which uses FM to pre-train the embedding of raw features and then feeds the embeddings to several fully connected layers. Some models adopt DNN to improve FM, such as Attentional FM [35], Neural FM [12].

Wide & Deep Learning [7] jointly trains a wide model and a deep model where the wide model leverages the effectiveness of feature engineering and the deep model learns implicit feature interactions. Despite the usefulness of wide component, feature engineering is expensive and requires expertise. To avoid feature engineering, DeepFM [9] introduces FM layer (order-2 feature interaction) as a wide component and uses a deep component to learn implicit feature interactions. Different from DeepFM, IPNN [23] (also known as PNN in [22]) feeds both the result of FM layer and embeddings of raw features into MLP and results in comparable performance. Rather than using inner product to model pairwise feature interactions as DeepFM and IPNN, PIN [23] uses a Micro-Network to learn complex feature interaction for each feature pair. xDeepFM [19] proposes a novel Compressed Interaction Network (CIN) to explicitly generate feature interactions at the vector-wise level.

There are several models which use CNN for CTR Prediction. CCPM [20] applies multiple convolutional layers to explore neighbor feature dependencies. CCPM performs convolutions on the neighbor fields in a certain alignment. Due to that the order of features has no actual meaning in CTR predictions, CCPM can only learn limited feature interactions between neighbor features. In [3], it is shown that features' arrangement order has a great impact on the final performance of CNN based models. Therefore, the authors propose to generate a set of suitable feature sequences to provide different local information for convolutional layers. However, the key weakness of CNN is not solved.

In this paper, we propose FGCNN model, which splits the CTR prediction task into two stages: *Feature Generation* and *Deep Classifier*. *Feature Generation* augments the original feature space by generating new features while most state-of-the-art models can be adopted in *Deep Classifier* to learn and prediction based on the augmented feature space. Different from traditional CNN models for CTR Prediction [3, 20], FGCNN can leverage the strength of CNN to extract local information and it greatly alleviates the weakness of CNN by introducing the *Recombination Layer* to recombine information from different local patterns learned by CNN to generating new features.

5 CONCLUSION

In this paper, we propose a FGCNN model for CTR prediction which aims to reduce the learning difficulties of DNN models by identifying important features in advance. The model consists of two components: *Feature Generation* and *Deep Classifier*. *Feature Generation* leverages the strength of CNN to identify useful local patterns and it alleviates the weakness of CNN by introducing a *Recombination Layer* to generate global new features from the recombination of the local patterns. In *Deep Classifier*, most existing deep models can be applied on the augmented feature space. Extensive experiments are conducted on three large-scale datasets where the results show that FGCNN outperforms nine state-of-the-art models. Moreover, when applying other models in *Deep Classifier*, compared with the original model without *Feature Generation*, better performance is always achieved, which demonstrates the effectiveness of the generated features. Step-by-step experiments show that each component in FGCNN contributes to the final performance. Furthermore, compared with the traditional CNN structure, our CNN+Recombination structure in *Feature Generation* always performs better and more stable when shuffling the arrangement order of raw features. This work explores a novel direction for CTR prediction that it is effective to automatically generate important features first rather than feeding the raw embeddings to deep learning models directly.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *Computer Science* (2014).

[2] Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2016. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683* (2016).

[3] Patrick PK Chan, Xian Hu, Lili Zhao, Daniel S Yeung, Dapeng Liu, and Lei Xiao. 2018. Convolutional Neural Networks based Click-Through Rate Prediction with Multiple Feature Sequences.. In *IJCAI*. 2007–2013.

[4] Yin Wen Chang, Cho Jui Hsieh, Kai Wei Chang, Michael Ringgaard, and Chih Jen Lin. 2010. Training and Testing Low-degree Polynomial Data Mappings via Linear SVM. *Journal of Machine Learning Research* 11, 11 (2010), 1471–1490.

[5] Junxuan Chen, Baigui Sun, Hao Li, Hongtao Lu, and Xian Sheng Hua. 2016. Deep CTR Prediction in Display Advertising. (2016), 811–820.

[6] Tianqi Chen and Carlos Guestrin. 2016. XGBoost:A Scalable Tree Boosting System. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 785–794.

[7] Hengtze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Deepak Chandra, Hrishi Aradhye, Glen Anderson, Greg S Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & Deep Learning for Recommender Systems. *conference on recommender systems* (2016), 7–10.

[8] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. ACM, 191–198.

[9] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247* (2017).

[10] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, Xiuqiang He, and Zhenhua Dong. 2018. DeepFM: An End-to-End Wide & Deep Learning Framework for CTR Prediction. (2018).

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. (2015), 770–778.

[12] Xiangnan He and TatSeng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. (2017), 355–364.

[13] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, and Stuart Bowers. 2014. Practical Lessons from Predicting Clicks on Ads at Facebook. In *Eighth International Workshop on Data Mining for Online Advertising*. 1–9.

[14] Gao Huang, Zhuang Liu, Van Der Maaten Laurens, and Kilian Q Weinberger. 2016. Densely Connected Convolutional Networks. (2016), 2261–2269.

[15] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[16] Yuchin Juan, Yong Zhuang, Wei Sheng Chin, and Chih Jen Lin. 2016. Field-aware Factorization Machines for CTR Prediction. In *ACM Conference on Recommender*

Systems. 43–50.

[17] Kuang Chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. 2012. Estimating conversion rate in display advertising from past erformance data. In *Acm Sigkdd International Conference on Knowledge Discovery & Data Mining*. 768–776.

[18] Kuang Chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. 2012. Estimating conversion rate in display advertising from past erformance data. In *Acm Sigkdd International Conference on Knowledge Discovery & Data Mining*. 768–776.

[19] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems. *arXiv preprint arXiv:1803.05170* (2018).

[20] Qiang Liu, Feng Yu, Shu Wu, and Liang Wang. 2015. A convolutional click prediction model. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1743–1746.

[21] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1222–1230.

[22] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. 2016. Product-based neural networks for user response prediction. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 1149–1154.

[23] Yanru Qu, Bohui Fang, Weinan Zhang, Ruiming Tang, Minzhe Niu, Huifeng Guo, Yong Yu, and Xiuqiang He. 2018. Product-based Neural Networks for User Response Prediction over Multi-field Categorical Data. *arXiv preprint arXiv:1807.00311* (2018).

[24] Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to Generate Reviews and Discovering Sentiment. (2017).

[25] Kan Ren, Weinan Zhang, Yifei Rong, Haifeng Zhang, Yong Yu, and Jun Wang. 2016. User Response Learning for Directly Optimizing Campaign Performance in Display Advertising. (2016), 679–688.

[26] Steffen Rendle. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 995–1000.

[27] Steffen Rendle. 2011. Factorization Machines. In *IEEE International Conference on Data Mining*. 995–1000.

[28] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks:estimating the click-through rate for new ads. In *International Conference on World Wide Web*. 521–530.

[29] Shai Shalevshwartz, Ohad Shamir, and Shaked Shammah. 2017. Failures of Gradient-Based Deep Learning. (2017).

[30] Ying Shan, T. Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and J. C. Mao. 2016. Deep Crossing:Web-Scale Modeling without Manually Crafted Combinatorial Features. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 255–262.

[31] Guangzhong Sun, Guangzhong Sun, Guangzhong Sun, Guangzhong Sun, Guangzhong Sun, and Guangzhong Sun. 2017. Practical Lessons for Job Recommendations in the Cold-Start Scenario. In *Recommender Systems Challenge*. 4.

[32] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. 2016. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. (2016).

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. (2017).

[34] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*. ACM, 12.

[35] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. *arXiv preprint arXiv:1708.04617* (2017).

[36] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. 2017. Aggregated Residual Transformations for Deep Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*. 5987–5995.

[37] Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep learning based recommender system: A survey and new perspectives. *arXiv preprint arXiv:1707.07435* (2017).

[38] Weinan Zhang, Tianming Du, and Jun Wang. 2016. Deep Learning over Multi-field Categorical Data. (2016).

[39] Weinan Zhang, Tianming Du, and Jun Wang. 2016. Deep Learning over Multi-field Categorical Data: A Case Study on User Response Prediction. (2016).

[40] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1059–1068.

Efficient (α, β)-core Computation: an Index-based Approach

Boge Liu
University of New South Wales
boge.liu@unsw.edu.au

Long Yuan*
Nanjing University of Science and Technology
longyuan@njust.edu.cn

Xuemin Lin
University of New South Wales
lxue@cse.unsw.edu.au

Lu Qin
University of Technology, Sydney
lu.qin@uts.edu.au

Wenjie Zhang
University of New South Wales
zhangw@cse.unsw.edu.au

Jingren Zhou
Alibaba Group
jingren.zhou@alibaba-inc.com

ABSTRACT

The problem of computing (α, β)-core in a bipartite graph for given α and β is a fundamental problem in bipartite graph analysis and can be used in many applications such as online group recommendation, fraudsters detection, etc. Existing solution to computing (α, β)-core needs to traverse the entire bipartite graph once. Considering the real bipartite graph can be very large and the requests to compute (α, β)-core can be issued frequently in real applications, the existing solution is too expensive to compute the (α, β)-core. In this paper, we present an efficient algorithm based on a novel index such that the algorithm runs in linear time regarding the result size (thus, the algorithm is optimal since it needs at least linear time to output the result). We prove that the index only requires $O(m)$ space where m is the number of edges in the bipartite graph. Moreover, we devise an efficient algorithm with time complexity $O(\delta \cdot m)$ for index construction where δ is bounded by $\sqrt{m}$ and is much smaller than $\sqrt{m}$ in practice. We also discuss efficient algorithms to maintain the index when the bipartite graph is dynamically updated and parallel implementation of the index construction algorithm. The experimental results on real and synthetic graphs (more than 1 billion edges) demonstrate that our algorithms achieve up to 5 orders of magnitude speedup for computing (α, β)-core and up to 3 orders of magnitude speedup for index construction, respectively, compared with existing techniques.

KEYWORDS

Graph processing, optimization, indexing

ACM Reference Format:
Boge Liu, Long Yuan, Xuemin Lin, Lu Qin, Wenjie Zhang, and Jingren Zhou. 2019. Efficient (α, β)-core Computation: an Index-based Approach. In *WWW 2018: The 2019 Web Conference, May 13–17, 2019, San Francisco, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313522

1 INTRODUCTION

Many real-world relationships across various entities can be modelled as bipartite graphs, such as customer-product networks [40], user-page networks [8], gene co-expression networks [21], collaboration networks [24], etc. With the proliferation of bipartite graph applications, research efforts have been devoted to many fundamental problems in managing and analyzing bipartite graph data. Among them, the problem of computing (α, β)-core of a bipartite graph for given α and β has been recently studied in [10, 11]. Formally, a bipartite graph $G = (U, V, E)$ is a graph with the nodes divided into two separate sets, U and V, such that every edge connects one node in U to another node in V. Given $G = (U, V, E)$ and two integers α and β, the (α, β)-core of G consists of two node sets $U' \subseteq U(G)$ and $V' \subseteq V(G)$ such that the subgraph induced by $U' \cup V'$ is the maximal subgraph of G in which all the nodes in U' have degree at least α and all the nodes in V' have degree at least β

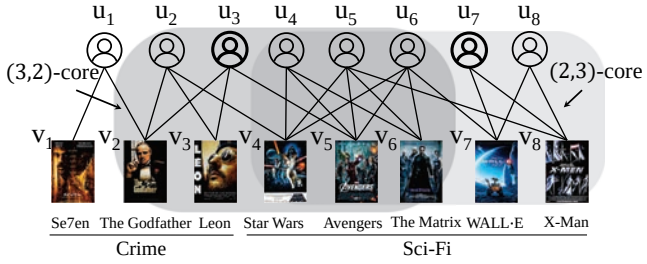


Figure 1: Part of a customer-movie network

Applications. Computing (α, β)-cores has many real applications.

(1) Online group recommendation. Group recommendation aims at recommending products to a group of users who may or may not share similar tastes, e.g., recommending movies for friends to watch together [4, 9, 15, 49]. Fault-tolerant group recommendation is proposed to deal with missing values in incomplete data and has shown its effectiveness in group recommendation [18, 31, 33]. A key step in fault-tolerant group recommendation is to compute fault-tolerant subspace clusters for each user in the group. For a given user u_q, a subspace cluster is a set of u_q's similar users, which is exploited by collaborative filtering [30] to compute the relevance of products to u_q. Recently, to accelerate the computation of fault-tolerant subspace clustering, [11] has shown that (α, β)-core is an efficient way of computing fault-tolerant subspace clustering. For a user u_q and user-specific parameters α and β, all the users in the (α, β)-core are treated as u_q's fault-tolerant subspace cluster. Since the α and β values can vary greatly based on users' preference and

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.

WWW 2019, May 13–17, 2019, San Francisco, USA

© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313522

the degree of tolerance for missing values [11], efficiently computing (α, β)-core is a critical procedure for online fault-tolerant group recommendation.

Example 1.1: Figure 1 shows part of the customer-movie network in the IMDB (https://www.imdb.com/) where each node in U represents a user, each node in V represents a movie and each edge indicates the customer has a preference for the movie. Assume that u_3 and u_7 are given as a group and the user-specific α and β for u_3 and u_7 are $(3, 2)$ and $(2, 3)$, respectively. Fault-tolerant group recommendation method first computes $(3, 2)$-core and $(2, 3)$-core, and uses $\{u_2, u_3, u_4, u_5, u_6\}$ and $\{u_4, u_5, u_6, u_7, u_8\}$ as fault-tolerant subspace clusters for u_3 and u_7, respectively. Then it conducts collaborative filtering based on the subspace clusters to calculate movie preference. In this case, Sci-Fi movies would be recommended to the group as both u_3 and u_7 have preference for Sci-Fi movies. □

(2) Fraudsters detection. In social networks, such as Facebook and Twitter, users and pages form a user-page bipartite graph in which the edge indicates a user likes a page. To promote certain pages, fraudsters use a larger number of fake accounts to inflate the *Like* counts for these pages; this results in a large number (β) of users liking a few (α) pages. (α, β)-core with small α and large β can facilitate the detection of such fraudsters [2, 8]. Similar fraud scenario also occurs in E-Commence/Online-Shopping, for example, fraudsters may improve the ranking of certain items by adding items to fake accounts' shopping lists.

(3) A key step to other problems in bipartite graphs. Computing (α, β)-core can also serve as a key step to solve other important graph problems, such as biclique computation [23, 52] and quasi-biclique computation [25, 28].

Motivations. In the literature, an online algorithm [11] is proposed for the computation of (α, β)-core. However, it has to traverse the entire graph to compute the (α, β)-core for given α and β. This makes it impractical to real scenarios, especially while taking into consideration that real bipartite graphs nowadays can be very large and the requests for computing (α, β)-core can be issued frequently. For example, the consumer-product networks of Amazon or Alibaba often reach billion-scale [27, 41]; in the application of online group recommendation, there can be millions of groups issuing recommendation requests at the peak time [12, 27, 49]. To recommend products to these groups, we need to compute (α, β)-core with different α and β for each user in every group. Therefore, numerous underlying computations of (α, β)-cores with different combinations of α and β have to be processed in realtime (typically within half a second [27]). However, it is shown in our experiments that, even in Orkut dataset with 327 million edges, existing method spends 236 seconds to compute (α, β)-core for a group of ten users. For fraud detection case, we also need to do lots of (α, β)-core computations to union results together because fraudsters may hide behind different combinations of α and β values [2, 8] and we don't want to miss out the suspicious people. Motivated by this, in this paper, we aim to devise an index-based optimal algorithm (linear time with respect to the result size) to compute the (α, β)-core for given α and β.

Challenges. To achieve our goal, we adopt an index-based approach. Straightforwardly, if we store the (α, β)-cores for all possible α and β combination, we can obtain the (α, β)-core in optimal time for given α and β. Nevertheless, this approach will take $O(n^3)$ space to store all results where n is the number of nodes in a bipartite graph. Obviously, this is prohibitive for a very large graph. Below, we present the challenges to be overcome in this paper.

- *Challenge 1: Optimal (α, β)-core computation vs Space Efficiency.* Considering that even one particular (α, β)-core for a given (α, β) may have $O(n)$ size and there could be $O(n^2)$ different combinations of α and β values, it is a challenge to develop a compact index such that we can compute (α, β)-core for given α and β in optimal time.
- *Challenge 2: Efficient index construction.* The proposed index is built upon the results of core decomposition on bipartite graphs. Note that core decomposition on unipartite (general) graphs [7] requires $O(m)$ time, simply extending this strategy to bipartite graphs with two disjoint node sets will lead to $O(\mathsf{dmax} \cdot m)$ time (details in Section 4.1), where dmax is the maximum degree of nodes and m is the number of edges in G. dmax could be very large in real graphs (e.g., dmax>10^7 in Web Trackers dataset), such method is impractical for large graphs. Hence, it is a challenge to devise an efficient algorithm to construct the index.

Contributions. In this paper, we overcome the above challenges and make the following contributions:

(1) The first space-efficient index-based work to compute (α, β)-core . In this paper, we propose a non-trivial space-efficient index structure, BiCore-Index, with the size bounded by $O(m)$. To the best of our knowledge, this is the first linear space index structure to support the optimal computation of (α, β)-core in bipartite graphs.

(2) Efficient algorithms to construct the index. We carefully consider the computation sharing between two node sets of the bipartite graph when conducting the core decomposition and devise an efficient algorithm to construct the BiCore-Index. We show that the time complexity of our proposed algorithm is $O(\delta \cdot m)$, where δ is the maximum value such that the (δ, δ)-core in G is nonempty and is bounded by $\sqrt{m}$. In our experiments, it is shown that δ is much smaller than $\sqrt{m}$ in practice.

(3) Efficient incremental maintenance algorithm for dynamic graphs and parallel implementation of index construction algorithm. In many applications, graphs are frequently updated. Therefore, we develop an efficient incremental algorithm to maintain BiCore-Index for dynamic graphs by reducing unnecessary recomputation in the procedure of updating BiCore-Index. Moreover, we also discuss the parallel implementation of our index construction algorithm.

(4) Extensive performance studies on real datasets from various domains. We conduct extensive performance studies on ten real graphs and two synthetic graphs. The experimental results demonstrate the efficiency of our proposed algorithms.

Outline. Section 2 gives the problem definition and the existing solution. Section 3 introduces our proposed index, BiCore-Index, and the optimal algorithm to compute (α, β)-core for arbitrary α and β. Section 4 presents algorithms to construct BiCore-Index. Section 5 presents an efficient algorithm to maintain BiCore-Index

in dynamic graphs and discusses parallel implementation of the index construction algorithm. Section 6 evaluates our algorithms using extensive experiments. Section 7 reviews the related work and Section 8 concludes the paper.

2 PRELIMINARIES

A bipartite graph $G = (U, V, E)$ is a graph consisting of two disjoint sets of nodes U and V such that every edge from $E \subseteq U \times V$ connects one node of U and one node of V. We use $U(G)$ and $V(G)$ to denote the two disjoint node sets of G and $E(G)$ to represent the edge set of G. We denote the number of nodes in $U(G)$ and $V(G)$ as n_U and n_V, the total number of nodes as n and the number of edges in $E(G)$ as m. The degree of a node $u \in U(G) \cup V(G)$, denoted by $\deg(u, G)$, is the number of neighbors of u in G. We also use $\mathsf{dmax}_U(G)$ ($\mathsf{dmax}_V(G)$) to denote the maximum degree among all the nodes in $U(G)$ ($V(G)$), i.e., $\mathsf{dmax}_U(G) = \max\{\deg(u, G) | u \in U(G)\}$ ($\mathsf{dmax}_V(G) = \max\{\deg(v, G) | v \in V(G)\}$). For simplicity, we omit G in the notations if the context is self-evident. For a bipartite graph G and two node sets $U' \subseteq U(G)$ and $V' \subseteq V(G)$, the bipartite subgraph induced by U' and V' is the subgraph G' of G such that $U(G') = U'$, $V(G') = V'$ and $E(G') = E(G) \cap (U' \times V')$.

Definition 2.1: ((α, β)-core) Given a bipartite graph G and two integers α and β, the (α, β)-core of G, denoted by $C_{\alpha,\beta}$, consists of two node sets $\mathcal{U} \subseteq U(G)$ and $\mathcal{V} \subseteq V(G)$ such that the bipartite subgraph g induced by $\mathcal{U} \cup \mathcal{V}$ is the maximal subgraph of G in which all the nodes in $\mathcal{U}$ have degree at least α and all the nodes in $\mathcal{V}$ have degree at least β, i.e., $\forall u \in \mathcal{U}, \deg(u, \mathrm{g}) \geq \alpha \wedge \forall v \in \mathcal{V}, \deg(v, \mathrm{g}) \geq \beta$. □

Problem Statement. In this paper, we study the problem of efficient computation of (α, β)-core for given α and β. For ease of presentation, we refer a request of computing the (α, β)-core for given α and β as an *(α, β)-core query* and denote it as $Q_{\alpha,\beta}$. Our object is to design a time-optimal algorithm for processing (α, β)-core queries on large bipartite graphs.

Existing Solution. Given an (α, β)-core query $Q_{\alpha,\beta}$, the state-of-the-art algorithm to compute $C_{\alpha,\beta}$ is proposed in [11]. Intuitively, it computes $C_{\alpha,\beta}$ by iteratively removing nodes in $U(G)$ with degree less than α and nodes in $V(G)$ with degree less than β until no more nodes can be removed. The above algorithm adopts an online paradigm to process (α, β)-core queries. For a query $Q_{\alpha,\beta}$, its time complexity to compute $C_{\alpha,\beta}$ is $O(m)$ in the worst case. Nevertheless, the graphs are typically very large in real applications (e.g., there are 327 million edges in Orkut dataset). Therefore, this algorithm cannot satisfy the real-time requirements for (α, β)-core queries since it needs to traverse the whole graph for a $Q_{\alpha,\beta}$.

3 SPACE-EFFICIENT INDEX AND TIME-OPTIMAL QUERY PROCESSING

In this section, we organize all the (α, β)-cores into a linear space index structure, through which an (α, β)-core query can be answered in optimal time. We first introduce a naive index structure. After analyzing the problems in the naive index structure, we present our linear space index structure, BiCore-Index. Based on BiCore-Index, we propose an optimal query processing algorithm. At last, we analyze the space complexity of BiCore-Index.

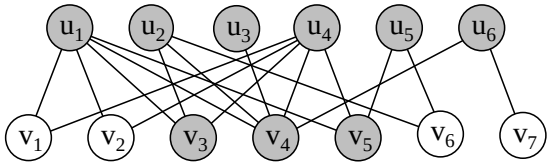


Figure 2: A bipartite graph G

3.1 BiCore-Index

A Naive Index Structure. To support optimal (α, β)-core query processing, a naive index is as follows: we pre-compute (α, β)-cores for all the possible α and β and store them in the index. Then, for all possible combination of α and β, we record the location of the corresponding (α, β)-core in the index through two level pointer tables. Given a query $Q_{\alpha,\beta}$, we can compute $C_{\alpha,\beta}$ in optimal time by visiting the nodes stored in the location referred by the (α, β) value. We show the naive index in the following example.

Example 3.1: Considering the graph G in Figure 2, the naive index of G is shown in Figure 3. For ease of presentation, we only show the nodes in $U(G)$ in Figure 3 and the nodes in $V(G)$ can be indexed similarly. In the index, all the pre-computed (α, β)-cores are stored and shown in the bottom bucket of Figure 3. For instance, $(1, 3)$-core is $\{\{u_1, \ldots u_6\}, \{v_3, v_4, v_5\}\}$, thus, $u_1, \ldots, u_6$ are stored in the grey bucket in Figure 3. Since both the maximum possible α value (dmax_U) and β value (dmax_V) of G are 5, the first-level pointer table (FPT) contains 5 pointers and each sub-table contains 5 pointers. Suppose the given query is $Q_{1,3}$, we can compute $C_{1,3}$ by following bold arrows and obtain $C_{1,3}.\mathcal{U} = \{u_1, \ldots, u_6\}$. □

This naive index can achieve optimal query processing time, however, it requires $O(n^3)$ space. Clearly, it is prohibitive for large graphs. In order to make the index-based approach practical, we aim to further reduce the space of the index without sacrificing the optimal query processing property.

Observing the naive index in Figure 3, we can find the following two problems exist, which leads to its huge space consumption. The first one is that a node may be stored multiple time in the index. For example, when $\alpha = 1$, u_1 is stored five times in the index, namely, in $C_{1,1}$, $C_{1,2}$, $C_{1,3}$, $C_{1,4}$ and $C_{1,5}$. The same problem also exists on other nodes and other α values. The second one is that empty entries are also kept in the index. For example, there exist no $C_{5,3}$, $C_{5,4}$ and $C_{5,5}$ in G. These entries should be managed to be removed while not affecting the optimal time complexity.

BiCore-Index Structure. We aim to reduce the space consumption of the naive index by addressing the two problems discussed above. Given a bipartite graph G, the (α, β)-cores in G has the following monotonic containment relationship:

Lemma 3.1: *Given a bipartite graph G, $C_{\alpha,\beta}$ is contained in $C_{\alpha',\beta'}$ if $\beta' \leq \beta$ and $\alpha' \leq \alpha$.*

Based on Lemma 3.1, for a node $u \in U(G)$ and a specific α, if we know the (α, β)-core with maximum β value containing u, we can infer that u is also contained in any (α, β)-core of G where β' is smaller than the maximum β value. For example, when $\alpha = 1$, since

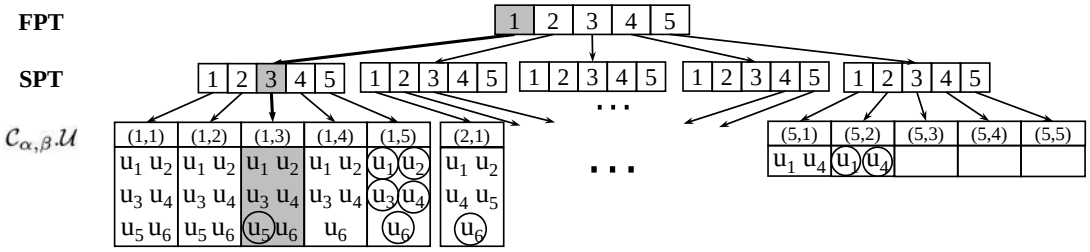


Figure 3: Naive Index

u_1 is contained in $C_{1,5}$, we know u_5 is also contained in $C_{1,1}$, $C_{1,2}$, $C_{1,3}$ and $C_{1,4}$. In other word, storing u_1 at $C_{1,1}$, $C_{1,2}$, $C_{1,3}$ and $C_{1,4}$ is redundant regarding (α, β)-core query processing and we only need to store it at $C_{1,5}$ (marked with circle in Figure 3). Therefore, to address the redundant nodes storage problem in the naive index, for a specific α, we remove the nodes $u \in U(G)$ from the (α, β)-cores that contain u but does not have the maximum β value.

For the empty entry problem, besides the existing empty entries in the index, the node removal procedure introduced above leads to new empty entries. For example, in Figure 3, after the node removal, $C_{1,1}.\mathcal{U}$ is empty. To address this problem, the naive index structure for nodes in $V(G)$.

Following the above idea, we give the formal definition of our index. Before that, to characterize the (α, β)-core with the maximum β (α) value that contains a node regarding a specific α (β), we define:

Definition 3.1:

(1) $\beta_{\max,\alpha}(u)$**.** Given a bipartite graph G and an integer α, for each node $u \in U(G) \cup V(G)$, $\beta_{\max,\alpha}(u)$ is the maximum value of β such that u is contained in the corresponding $C_{\alpha,\beta}$. If no such β, $\beta_{\max,\alpha}(u) = 0$.

(2) $\alpha_{\max,\beta}(u)$**.** Given a bipartite graph G and an integer β, for each node $u \in U(G) \cup V(G)$, $\alpha_{\max,\beta}(u)$ is the maximum value of α such that u is contained in the corresponding $C_{\alpha,\beta}$. If no such α, $\alpha_{\max,\beta}(u) = 0$. □

Our index, BiCore-Index, denoted by $\mathbb{I}$, is a three-level tree structure with two parts for nodes in $U(G)$ and $V(G)$, respectively, denoted by $\mathbb{I}^U$ and $\mathbb{I}^V$. As $\mathbb{I}^V$ is symmetrical to $\mathbb{I}^U$, we focus on $\mathbb{I}^U$ here.

- *Node Blocks* (NB). The third level of $\mathbb{I}^U$, named the node blocks, is a double linked list. Each block in the list is associated with a (α, β) value and contains the nodes $u \in U(G)$ with $\beta_{\max,\alpha}(u) = \beta$.
- *First-level Pointer Table* (FPT). The first level of $\mathbb{I}^U$ is an array with dmax_U elements. Each element contains a pointer to an array in the second level. We use $\mathbb{I}^U[\alpha]$ to represent the α-th element.
- *Second-level Pointer Table* (SPT). The second level of $\mathbb{I}^U$ consists of dmax_U arrays (sub-table). The α-th array is pointed by $\mathbb{I}^U[\alpha]$. The length of the α-th array equals to the maximum β value among the node blocks pointed by this array. We use $\mathbb{I}^U[\alpha][\beta]$ to denote the β-th element of the α-th array in SPT. The pointer in $\mathbb{I}^U[\alpha][\beta]$ points to the first node block associated with (α, β') where $\beta' \geq \beta$.

Example 3.2: Figure 4 shows the BiCore-Index of G. In NB, u_1 is in node block $(1, 5)$ since $\beta_{\max,1}(u_1) = 5$. In FPT, since $\mathsf{dmax}_U = 5$, the array of FPT contains 5 pointers pointing to the corresponding array in SPT. Different from the naive index, the length of the arrays is not unique. For example, the length of the second array is 3. This is because for $\alpha = 2$, the $(2, \beta)$ node block with maximum β value kept in NB is node block $(2, 3)$. The pointer in the 1st element of the 1st array in SPT points to node block $(1, 3)$ as node block $(1, 1)$ and $(1, 2)$ do not exist in NB. *A key point needed to note here is that a node block in $\mathbb{I}^U$ and a node block in $\mathbb{I}^V$ may share the same associated (α, β) value, but their meanings are different.* For example, v_1 is contained in node block $(1, 5)$ in $\mathbb{I}^V$ means v_1 is contained in $(5, 1)$-core while u_1 is contained in node block $(1, 5)$ in $\mathbb{I}^U$ means u_1 is contained in $(1, 5)$-core. □

Algorithm 1 QueryOPT

Input: $\mathbb{I}$ of G and $Q_{\alpha,\beta}$
Output: $C_{\alpha,\beta}$ of G

```
1:  C_{α,β} ← ∅;
2:  if 𝕀^U.FPT.size() < α or 𝕀^U[α].size() < β then
3:      return ∅;
4:  nb ← node block pointed by 𝕀^U[α][β];
5:  while the first element of the associated value of nb = α do
6:      for each u ∈ nb do
7:          C_{α,β}.𝒰 ← C_{α,β}.𝒰 ∪ u;
8:      nb ← next node block in 𝕀^U.NB;
9:  Compute C_{α,β}.𝒱 similarly
10: return C_{α,β} ;
```

3.2 Optimal Query Processing

With BiCore-Index, for a query $Q_{\alpha,\beta}$, we compute $C_{\alpha,\beta}$ by retrieving $C_{\alpha,\beta}.\mathcal{U}$ through $\mathbb{I}^U$ and $C_{\alpha,\beta}.\mathcal{V}$ through $\mathbb{I}^V$. The algorithm, QueryOPT, is shown in Algorithm 1.

Algorithm. For a given $Q_{\alpha,\beta}$, if the (α, β)-core is empty, QueryOPT immediately returns $\emptyset$ as the result since either $\mathbb{I}^U[\alpha]$ or $\mathbb{I}^U[\alpha][\beta]$ is empty (line 2-3). If the (α, β)-core is not empty, it first retrieves $C_{\alpha,\beta}.\mathcal{U}$ and computes the node block nb referred by the pointer in $\mathbb{I}^U[\alpha][\beta]$ (line 4). After that, it iteratively processes the node block in $\mathbb{I}^U$.NB until the first element of the associated value of nb is not the given α (line 5 and 8). All the nodes in visited nb are added into $C_{\alpha,\beta}.\mathcal{U}$ (line 6-7). The nodes in $C_{\alpha,\beta}.\mathcal{V}$ are retrieved similarly and $C_{\alpha,\beta}$ is returned at the end (line 9-10).

Example 3.3: Figure 4 illustrates the procedure of QueryOPT to compute $C_{1,3}$. Processing steps are shown in bold arrows and the visited elements are marked in grey. To compute $C_{1,3}.\mathcal{U}$, QueryOPT

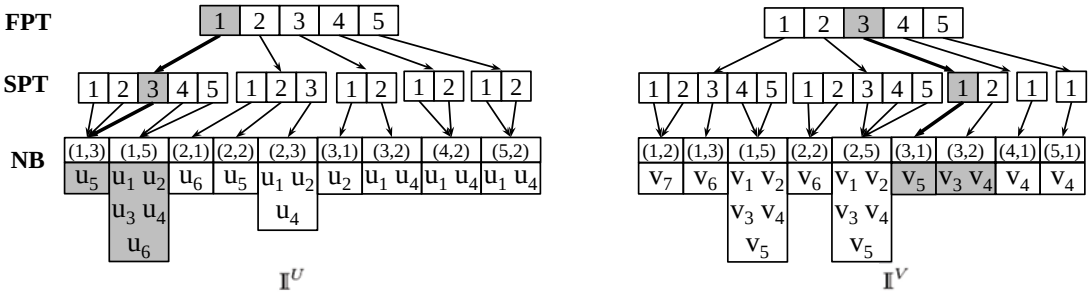


Figure 4: BiCore-Index **and procedure of** QueryOPT **for** $Q_{1,3}$

follows the pointer kept in the 1st element in $\mathbb{I}^U$.FPT and the 3rd element of the 1st array in $\mathbb{I}^U$.SPT and obtains u_5 in the node block $(1,3)$. It continues to visit node block $(1,5)$ and stops at node block $(2,1)$ since the first element of $(2,1)$ is larger than 1. Thus, $C_{1,3}.\mathcal{U} = \{u_1, u_2, u_3, u_4, u_5, u_6\}$. Similarly, QueryOPT follows the pointer kept in the 3rd element in $\mathbb{I}^V$.FPT and the 1st element of the 3rd array in $\mathbb{I}^V$.SPT and obtains $C_{1,3}.\mathcal{V} = \{v_3, v_4, v_5\}$. □

Theorem 3.1: *Given a $Q_{\alpha,\beta}$ posed on a bipartite graph G,* QueryOPT *computes $C_{\alpha,\beta}$ in $O(|C_{\alpha,\beta}.\mathcal{U}| + |C_{\alpha,\beta}.\mathcal{V}|)$.*

Proof: For a given α, each $u \in U(G)$ appears at most once in the node blocks pointed by the pointers in α-th array in SPT. Therefore, no duplicate node is added in $C_{\alpha,\beta}.\mathcal{U}$ in line 7. Similarly, no duplicate node is added in $C_{\alpha,\beta}.\mathcal{V}$ in line 9. Since all the nodes visited in QueryOPT are exactly the nodes we need to retrieve, QueryOPT computes $C_{\alpha,\beta}$ in $O(|C_{\alpha,\beta}.\mathcal{U}| + |C_{\alpha,\beta}.\mathcal{V}|)$ time. □

3.3 Space Complexity of BiCore-Index

In this section, we prove the linear space complexity of BiCore-Index. We first show that the size of SPT can be bounded by $O(m)$ in Lemma 3.2. Then, we prove that the space complexity of BiCore-Index is $O(m)$ in Theorem 3.2.

Lemma 3.2: *Given a bipartite graph G, the space of its* SPT *is bounded by $O(m)$.*

Proof: Let $u_1, u_2, u_3, \ldots, u_{n_U}$ be any given sequence of $u \in U(G)$. Starting from an empty graph with only $V(G)$, we add nodes in $U(G)$ with their incident edges to the graph one by one following the sequence until we finally get G. Suppose that u_i is just added to the graph. As u_i cannot be contained in any (α,β)-core whose $\alpha > \deg(u_i, G)$, u_i only influences the length of the k-th arrays in SPT with $1 \le k \le \deg(u_i, G)$. Because the length of the α-th array increases at most one after the insertion of u_i, the size of SPT increases at most $\deg(u_i, G)$. Thus, the space of SPT in $\mathbb{I}^U$ is bounded by $O(\sum_{u \in U(G)} \deg(u, G)) = O(m)$. Similarly, it can be shown that the space of SPT in $\mathbb{I}^V$ is also bounded by $O(m)$. Therefore, the space of SPT is bounded by $O(m)$. □

Theorem 3.2: *Given a bipartite graph G, the space of its* BiCore-Index *is bounded by $O(m)$.*

Proof: Since both $\mathsf{dmax}_U(G)$ and $\mathsf{dmax}_V(G)$ is smaller than m, the size of FPT can be bounded by $O(m)$. For each node $u \in U(G) \cup V(G)$, the number of node blocks containing u is $\deg(u)$. Hence, the space of NB is $O(\sum_{u \in U(G) \cup V(G)} \deg(u, G)) = O(m)$. Based on Lemma 3.2, the space of BiCore-Index can be bounded by $O(m)$. □

4 INDEX CONSTRUCTION ALGORITHM

In this section, we introduce how to construct BiCore-Index efficiently. Based on the structure of BiCore-Index, if we know $\beta_{\max,\alpha}(u)$ for each node $u \in U(G)$ regarding all possible α and $\alpha_{\max,\beta}(v)$ for each node $v \in V(G)$ regarding all possible β (in consistence with the literature on unipartite graphs, we call the procedure as core decomposition as well), the construction of BiCore-Index is straightforward and can be finished in $O(m)$ time as shown in Section 4.3. Therefore, we first present techniques to conduct the core decomposition.

4.1 A Basic Core Decomposition Algorithm

Inspired by the algorithm in [11], considering a node $u \in U(G)$ and a specific α, if $u \in C_{\alpha,\beta}.\mathcal{U}$ and $u \notin C_{\alpha,\beta+1}.\mathcal{U}$, we know $\beta_{\max,\alpha}(u) = \beta$. Moreover, for a specific α, $C_{\alpha,\beta+1}$ is contained in $C_{\alpha,\beta}$. Therefore, for a specific α, if we compute all the possible (α,β)-cores in increasing order of β by iteratively removing nodes in $U(G)$ with degree less than α and nodes in $V(G)$ with degree less than β, we can obtain $\beta_{\max,\alpha}(u)$ for all nodes $u \in U(G)$ regarding the specific α. Following this way, we can compute $\beta_{\max,\alpha}(u)$ for all $u \in U(G)$ by iterating all possible α values of G in a bottom-up manner. $\alpha_{\max,\beta}(v)$ can be computed similarly.

Algorithm. The basic algorithm, BasicDecom, is shown in Algorithm 2. BasicDecom first computes $\beta_{\max,\alpha}(u)$ for nodes in $u \in U(G)$. Since the maximum value of α for all nodes in $U(G)$ is dmax_U, it iterates α between 1 and dmax_U and computes $\beta_{\max,\alpha}(u)$ for $u \in U(G)$ regarding the specific α by invoking compute$\beta_{\max}$ (line 1-2). Similarly, $\alpha_{\max,\beta}(v)$ for $v \in V(G)$ are computed in line 3-4.

Procedure compute$\beta_{\max}$ computes $\beta_{\max,\alpha}(u)$ for all the nodes in $u \in U(G)$ for a given α. It first removes the nodes and their incident edges in G' whose degree is less than α (line 7-8). Then, it processes the nodes in $U(G)$ in increasing of β (line 10). Whenever a node v with $\deg(v, G') \le \beta$ is removed (line 11), if there exists a node u with $\deg(u, G') < \alpha$ in G', we know that $u \in C_{\alpha,\beta}$ but $u \notin C_{\alpha,\beta+1}$, which means $\beta_{\max,\alpha}(u)$ regarding α is β (line 14). Procedure compute$\alpha_{\max}$ follows a similar framework as compute$\beta_{\max}$ to compute $\alpha_{\max,\beta}(v)$ for $v \in V(G)$ regarding a given β (line 16-17).

Example 4.1: Considering the graph in Figure 2, Figure 5 shows the procedure of BasicDecom to conduct the core decomposition. Since

Algorithm 2 BasicDecom

Input: $G = (U \cup V, E)$
Output: $\beta_{\max,\alpha}(u)$ for $u \in U(G)$, $\alpha_{\max,\beta}(v)$ for $v \in V(G)$
1: **for** $\alpha = 1$ **to** dmax_U **do**
2: compute$\beta_{\max}(G, \alpha)$;
3: **for** $\beta = 1$ **to** dmax_V **do**
4: compute$\alpha_{\max}(G, \beta)$;
5: **procedure** compute$\beta_{\max}(G, \alpha)$
6: $G' \leftarrow G$;
7: **while** $\exists u \in U(G') : \deg(u, G') < \alpha$ **do**
8: remove u and its incident edges from G';
9: **while** $G' \neq \emptyset$ **do**
10: $\beta \leftarrow \min_{v \in V(G')} \deg(v, G')$;
11: **while** $\exists v \in V(G') : \deg(v, G') \leq \beta$ **do**
12: remove v and its incident edges from G';
13: **while** $\exists u \in U(G') : \deg(u, G') < \alpha$ **do**
14: $\beta_{\max,\alpha}(u) \leftarrow \beta$;
15: remove u and its incident edges from G';
16: **procedure** compute$\alpha_{\max}(G, \beta)$
17: line 6-15 by interchanging u with v, U with V, α with β;

Iteration	$\beta_{\max,\alpha}(u_1)$					$\alpha_{\max,\beta}(v_4)$				
	1	2	3	4	5	1	2	3	4	5
1 ($\alpha = 1$)	5	0	0	0	0	0	0	0	0	0
2 ($\alpha = 2$)	5	3	0	0	0	0	0	0	0	0
3 ($\alpha = 3$)	5	3	2	0	0	0	0	0	0	0
4 ($\alpha = 4$)	5	3	2	2	0	0	0	0	0	0
5 ($\alpha = 5$)	5	3	2	2	2	0	0	0	0	0
6 ($\beta = 1$)	5	3	2	2	2	5	0	0	0	0
7 ($\beta = 2$)	5	3	2	2	2	5	5	0	0	0
8 ($\beta = 3$)	5	3	2	2	2	5	5	2	0	0
9 ($\beta = 4$)	5	3	2	2	2	5	5	2	1	0
10($\beta = 5$)	5	3	2	2	2	5	5	2	1	1

Figure 5: Decomposition procedure of Algorithm 2

the decomposition involves all the nodes and large number of values, we only show the procedure for two representative nodes, u_1 and v_4, for brevity. In iteration 1, BasicDecom invokes compute$\beta_{\max}$ with $\alpha = 1$ and finds that u_1 is removed when $\beta = 5$. Thus, it updates $\beta_{\max,1}(u_1)$ as 5. BasicDecom finishes computation in 10 iterations as both dmax_U and dmax_V are 5. □

Theorem 4.1: *Given a bipartite graph G, Algorithm 2 runs in $O(\text{dmax} \cdot m)$ time, where $\text{dmax} = \max\{\text{dmax}_U, \text{dmax}_V\}$.*

Proof: The removal of node v in line 12 and node u in line 8 and 15 can be done in $O(\deg(v, G'))$ and $O(\deg(u, G'))$ time with the efficient data structure proposed in [22]. Since each node is removed once, the time complex of compute$\beta_{\max}$ is bounded by $O(m)$. Similarly, the running time compute$\alpha_{\max}$ is also $O(m)$. Thus, the time complexity of BasicDecom is $O(\text{dmax} \cdot m)$. □

4.2 A Computation-sharing Core Decomposition Algorithm

Algorithm 3 processes the nodes in $U(G)$ and $V(G)$ independently and has to conduct $O(\text{dmax})$ iterations to complete the core decomposition. However, dmax can be very large in real graphs [6], which makes Algorithm 2 impractical. In this section, we reduce the number of iterations to 2δ, where δ is the maximum value

Algorithm 3 ComShrDecom

Input: $G = (U \cup V, E)$
Output: $\beta_{\max,\alpha}(u)$ for $u \in U(G)$, $\alpha_{\max,\beta}(v)$ for $v \in V(G)$
1: $\delta \leftarrow$ the maximum value such that $C_{\delta,\delta} \neq \emptyset$
2: **for** $\alpha = 1$ **to** δ **do**
3: compute$\beta_{\max}(G, k)$;
4: **for** $\beta = 1$ **to** δ **do**
5: compute$\alpha_{\max}(G, \beta)$;
6: **procedure** compute$\beta^{+}_{\max}(G, \alpha)$
7: line 6-8 of Algorithm 2;
8: **while** $G' \neq \emptyset$ **do**
9: $\beta \leftarrow \min_{v \in V(G')} \deg(v, G')$;
10: **while** $\exists v \in V(G') : \deg(v, G') \leq \beta$ **do**
11: remove v and its incident edges from G';
12: **for** $i = 1$ **to** β **do**
13: **if** $\alpha_{\max,i}(v) < \alpha$ **then**
14: $\alpha_{\max,i}(v) \leftarrow \alpha$;
15: **while** $\exists u \in U(G') : \deg(u, G') < \alpha$ **do**
16: $\beta_{\max,\alpha}(u) \leftarrow \beta$;
17: remove u and its incident edges from G';
18: **procedure** compute$\alpha^{+}_{\max}(G, \beta)$
19: line 8-19 by interchanging u with v, U with V, α with β;

such that $C_{\delta,\delta}$ is nonempty and is bounded by $\sqrt{m}$, by exploring computation-sharing opportunities during processing the nodes in $U(G)$ and $V(G)$.

In Algorithm 2, when finishing processing a specific α, we actually have computed all the $C_{\alpha',\beta}$ with $\alpha' \leq \alpha$ in G. Meanwhile, for a node $v \in V(G)$ and a given β, $\alpha_{\max,\beta}(v)$ is the maximum value of α such that v is contained in the corresponding $C_{\alpha,\beta}$. Therefore, we can also obtain $\alpha_{\max,\beta}(v) \leq \alpha$ for $v \in V(G)$ when finishing processing a specific α in Algorithm 2. Similarly, we can obtain $\beta_{\max,\alpha}(u) \leq \beta$ for $u \in U(G)$ after processing a specific β. Moreover, let δ be the maximum value such that the corresponding $C_{\delta,\delta}$ is nonempty, we have:

Lemma 4.1: *Given a bipartite graph G, $\alpha_{\max,\beta}(v) \leq \delta$, for all $\beta > \delta$ and $v \in V(G)$; $\beta_{\max,\alpha}(v) \leq \delta$, for all $\alpha > \delta$ and $u \in U(G)$.*

Proof: Suppose that there exists some $v \in V(G)$ and $\beta > \delta$ such that $\alpha_{\max,\beta}(v) > \delta$, based on the definition of $\alpha_{\max,\beta}(v)$, $C_{\delta+1,\delta+1}$ must be nonempty, which contradicts the definition of δ. Thus, $\alpha_{\max,\beta}(v) \leq \delta$, for all $\beta > \delta$ and $v \in V(G)$. Similarly, the second part is correct. □

Based on Lemma 4.1, if we iterate α from 1 to δ in Algorithm 2, besides computing $\beta_{\max,\alpha}(u)$ for each $\alpha \leq \delta$ and each $u \in U(G)$ in line 14, we can actually also obtain $\alpha_{\max,\beta}(v)$ for each $\beta > \delta$ and each $v \in V(G)$. Similarly, if we iterate β from 1 to δ, we can obtain not only $\alpha_{\max,\beta}(v)$ for each $\beta \leq \delta$ and each $v \in V(G)$ but also $\beta_{\max,\alpha}(u)$ for each $\alpha > \delta$ and each $u \in U(G)$.

Algorithm. Following above idea, our computation-sharing algorithm, ComShrDecom, is shown in Algorithm 3. In Algorithm 3, ComShrDecom first computes δ of G. δ can be achieved based on its definition by increasing δ step by step starting from 1 while iteratively removing nodes from G whose degree is less than δ. When G is empty, δ is obtained and it can be done in $O(m)$ time. Then, ComShrDecom iterates α and β from 1 to δ to compute

Iteration	$\beta_{\max,\alpha}(u_1)$					$\alpha_{\max,\beta}(v_4)$				
	1	2	3	4	5	1	2	3	4	5
1 ($\alpha=1$)	5	0	0	0	0	1	1	1	1	1
2 ($\alpha=2$)	5	3	0	0	0	2	2	2	1	1
3 ($\beta=1$)	5	3	1	1	1	5	2	2	1	1
4 ($\beta=2$)	5	3	2	2	2	5	5	2	1	1

Figure 6: Decomposition procedure of Algorithm 3

$\beta_{\max,\alpha}(u)$ for $u \in U(G)$ and $\alpha_{\max,\beta}(v)$ for $v \in V(G)$ by invoking compute$\beta^+_{\max}$ and compute$\alpha^+_{\max}$, respectively (line 2-5).

The main difference between procedure compute$\alpha^+_{\max}$ and procedure compute$\alpha_{\max}$ is that compute$\alpha_{\max}$ updates $\beta_{\max,\alpha}(u)$ and $\alpha_{\max,\beta}(v)$ simultaneously based on the computation result of previous iterations. More specifically, when compute$\beta^+_{\max}$ removes a node $v \in V(G)$ and its incident edges from G' (line 11), for each i from 1 to β, if $\alpha_{\max,i}(v) < \alpha$, it updates the corresponding $\alpha_{\max,i}(v)$ as α (line 13-14). This is because when v is removed, v is in a $C_{\alpha,\beta}$, thus $\alpha_{\max,i}(v)$ is at least α. After compute$\beta^+_{\max}$ finishes, the $\alpha_{\max,\beta}(v) \le \alpha$ for nodes $v \in V(G)$ are obtained. Procedure compute$\alpha^+_{\max}$ conducts the process symmetrically as compute$\beta^+_{\max}$.

Example 4.2: Figure 6 shows the procedure of ComShrDecom to compute $\beta_{\max,\alpha}(u_1)$ and $\alpha_{\max,\beta}(v_4)$. ComShrDecom first computes $\delta = 2$, thus it needs 4 iterations to finish the decomposition. Compared with BasicDecom, ComShrDecom updates both $\beta_{\max,\alpha}(u_1)$ and $\alpha_{\max,\beta}(v_4)$ simultaneously in a single iteration. In iteration 2, it invokes compute$\beta^+_{\max}$ with $\alpha = 2$ and finds that both u_1 and v_4 are removed when $\beta = 3$. Thus ComShrDecom updates $\beta_{\max,2}(u_1)$ to 3 and $\alpha_{\max,1}(v_4)$, $\alpha_{\max,2}(v_4)$, $\alpha_{\max,3}(v_4)$ to 2. □

Theorem 4.2: *Given a bipartite graph G, the time complexity of Algorithm 3 is $O(\delta \cdot m)$, where $\delta \le \lceil\sqrt{m}\rceil$.*

Proof: The difference between compute$\beta_{\max}$ and compute$\beta^+_{\max}$ lies in line 12-14. Since the maximum possible value of β in line 12 can be no larger than $\deg(v,G)$, the time complexity of line 12-14 is $O(\deg(v,G))$. Hence, compute$\beta^+_{\max}$ runs in $O(m)$ time. Similarly, compute$\alpha^+_{\max}$ runs in $O(m)$ time. Thus, Algorithm 3 also runs in $O(\delta \cdot m)$.

Let g denote the subgraph induced by $C_{\delta,\delta}$. Based on the definition of (α,β)-core , there are at least δ nodes in g and the degree of each node is at least δ. Thus, we have $\delta \cdot \delta \le E(g) \le m$. Therefore, $\delta \le \sqrt{m}$. □

Remark. In fact, the number of iterations in Algorithm 3, which equals to $2 \cdot \delta$, is within a constant factor of 2 to the optimal number of iterations we can achieve. This is because essentially during the decomposition process we need to compute each nonempty (α,β)-core at least once. Hence, we should at least iterate α from 1 to δ or iterate β from 1 to δ to compute all the (α,β)-cores whose $\alpha \le \delta \wedge \beta \le \delta$. In other words, the lower bound of the number of iterations required to conduct the decomposition is δ. Therefore, the number of iterations in Algorithm 3 is within a constant factor of 2 to the optimal number of iterations.

4.3 Index Construction Algorithm

After obtaining the core decomposition result, we can construct BiCore-Index based on its structure directly. For $\mathbb{I}^U$, we first constructs $\mathbb{I}^U$.NB based on $\beta_{\max,\alpha}(u)$ and sort all the node blocks with their associated (α,β) value. After that, we store the address of the first node block (α,β') such that $\beta' \ge \beta$ in $\mathbb{I}^U[\alpha][\beta]$ and the address of the α-th array in $\mathbb{I}^U[\alpha]$. $\mathbb{I}^V$is constructed similarly. For a given bipartite graph G and its core decomposition result, the index can be constructed in $O(m)$ time.

5 EXTENSIONS

Index Maintenance on Dynamic Graphs. When graphs are dynamically updated, a straightforward solution to maintain BiCore-Index is reconstructing it from scratch, which is inefficient for large graphs. In this section, we discuss the incremental algorithms for maintaining BiCore-Index on dynamic graphs. We mainly focus on edge insertion and deletion, because node insertion/deletion can be treated as a sequence of edge insertions/deletions. For brevity, we use G^+/G^- to represent the updated graph after (u,v) is inserted/removed. Since the computation of $\beta_{\max,\alpha}(u)$ and $\alpha_{\max,\beta}(v)$ is the most time-consuming part for maintaining BiCore-Index, we concentrate on computing these values here.

Edge Insertion. Suppose that an edge (u,v) is inserted, for an integer α, let $\tau_\alpha = \min\{\beta_{\max,\alpha}(u,G), \beta_{\max,\alpha}(v,G)\}$. Because both u and v have already been included in any $C_{\alpha,\beta}$ whose $\beta < \tau_\alpha$, these (α,β)-cores will not change after the insertion. Thus, we only need to consider those nodes u' whose $\beta_{\max,\alpha}(u',G) \ge \tau_\alpha$, i.e., nodes in C_{α,τ_α}. Similarly, for an integer β, let $\tau_\beta = \min\{\alpha_{\max,\beta}(u,G), \alpha_{\max,\beta}(v,G)\}$, all the nodes u' whose $\alpha_{\max,\beta}(u',G^+)$ value will change after the insertion of (u,v) are contained in $C_{\tau_\beta,\beta}$. Thus, we retrieve the induced subgraph of C_{α,τ_α} ($C_{\tau_\beta,\beta}$) through the BiCore-Index and compute $\beta_{\max,\alpha}(u',G^+)$ ($\alpha_{\max,\beta}(u',G^+)$) for those nodes by using compute$\beta^+_{\max}$ (compute$\alpha^+_{\max}$).

Edge Removal. Similarly to edge insertion case, for an integer α, all the nodes u' whose $\beta_{\max,\alpha}(u',G)$ will change after the removal of (u,v) must have $\beta_{\max,\alpha}(u',G) \le \tau_\alpha$. This is because edge (u,v) is not included in any induced subgraph of $C_{\alpha,\beta}$ where $\beta > \tau_\alpha$. Similarly, for an integer β, we know all the nodes u' whose $\alpha_{\max,\beta}(u',G)$ will change after the removal of (u,v) must have $\alpha_{\max,\beta}(u',G) \le \tau_\beta$. Thus, we retrieve C_{α,τ_α} ($C_{\tau_\beta,\beta}$) through the BiCore-Index and compute $\beta_{\max,\alpha}(u',G^-)$ ($\alpha_{\max,\beta}(u',G^-)$) for those nodes similarly as edge insertion case.

Batch Update. When a sequence of edges are inserted/removed, we first scan the sequence and remove all the operation pairs consisting of insertion then removal (removal then insertion) of the same edge as these operation pairs have no effect on the final result. After that we rearrange the order such that all the inserted edges come after removed edges. Thus, we can treat batch update as first removing a set of edges then inserting another set of edges. When a set of edges is inserted, our incremental algorithm can be easily extended to handle such situation. For an integer $\alpha(\beta)$, we set $\pi_\alpha(\pi_\beta)$ as the smallest $\beta_{\max,\alpha}(u,G)(\alpha_{\max,\beta}(u,G))$ where u is incident to at least one inserted edge. Then, we use the method in *Edge Insertion* to update BiCore-Index by replacing $\tau_\alpha(\tau_\beta)$ as $\pi_\alpha(\pi_\beta)$. Similarly, when a set of edges is removed, we set $\pi_\alpha(\pi_\beta)$ as the largest $\beta_{\max,\alpha}(u,G)(\alpha_{\max,\beta}(u,G))$ where u is incident to at least one removed edge. Then, we can use the method in *Edge Removal* by replacing $\tau_\alpha(\tau_\beta)$ as $\pi_\alpha(\pi_\beta)$.

Parallel Index Construction. Algorithm 3 can be easily extended to run in parallel. We first compute δ as line 1 in Algorithm 3. Then, we run compute$\beta^+_{\max}$ for each α from 1 to δ and compute$\alpha^+_{\max}$ for

each β from 1 to δ in parallel, i.e., each thread is in charge of a continuous subrange of possible values of α or β. To avoid race condition, we keep a copy of $\beta_{\max,\alpha}(u,G)$ and $\alpha_{\max,\beta}(v,G)$ for each thread. At last, we set $\beta_{\max,\alpha}(u,G)$ and $\alpha_{\max,\beta}(v,G)$ as the largest value among the results computed in all threads.

6 PERFORMANCE STUDIES

This section presents our experimental results. All experiments are performed under a Linux operating system on a machine with an Intel Xeon 3.4GHz CPU and 64GB RAM.

Dataset. We evaluate the algorithms on ten real graphs and two synthetic graphs. All the real graphs are downloaded from KONECT[1]. For the synthetic graphs, we generate a power-law graph (PL) in which edges are randomly added such that the degree distribution follows a power-law distribution and a uniform-degree graph (UD) in which all edges are added with the same probability. The details of these graphs are shown in Table 1. Note that we remove isolated nodes and duplicate edges in graphs and their sizes listed are based on the processed graphs.

Algorithms. We implement and compare following algorithms:

- Baseline: the state-of-the-art existing solution proposed in [11] (introduced in Section 2).
- QueryOPT: Our (α,β)-core query processing algorithm (Algorithm 1).
- BasicDecom: Our proposed index construction algorithm based on basic decomposition algorithm (Algorithm 2 + Index construction algorithm in Section 4.3).
- ComShrDecom: Our proposed index construction algorithm based on computation-sharing decomposition algorithm (Algorithm 3 + Index construction algorithm in Section 4.3).
- BiCore-Index-Ins: Our algorithm for handling edge insertion.
- BiCore-Index-Rem: Our algorithm for handling edge removal.
- BiCore-Index-Batch: Our algorithm for handling batch update.
- ParallelDecom: Our proposed algorithm for parallel index construction.

All algorithms are implemented in C++, using gcc compiler at -O3 optimization level. The time cost is measured as the amount of wall-clock time elapsed during the program's execution. All the experiments are repeated 5 times and we report the average time.

6.1 Performance of Querying Processing

In this section, we evaluate the performance of our proposed (α,β)-core query processing algorithm QueryOPT with the state-of-the-art algorithm Baseline. The running time we report is based on answering the query 10 times. We first test the algorithms on all the twelve datasets with the same query $Q_{10,10}$. Then, we report the performance of the algorithms to process $Q_{\alpha,\beta}$ when varying α (β) regarding fixed β (α).

Exp-1: Query performance on different datasets. Figure 7 shows the running time of two query processing algorithms to process $Q_{10,10}$. We only show the results on the six largest datasets due to the similar trends. Since QueryOPT is optimal, it is always the fastest algorithm in all cases. For example, on DUI, the running time of QueryOPT is 0.04s, which achieves three order of magnitude improvement compared with Baseline (86.8s).

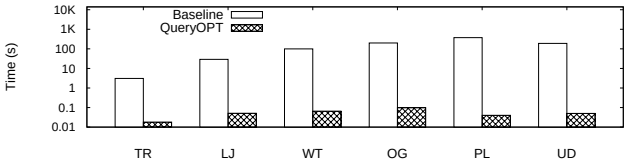


Figure 7: Query performance on different datasets

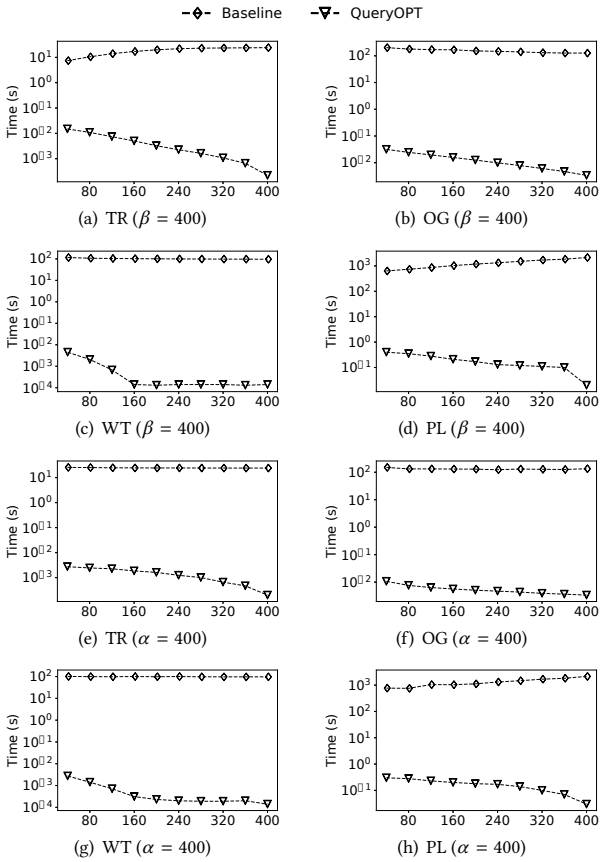


Figure 8: Query time for different α (β)

Exp-2: Varying α (β). The running time of Baseline and QueryOPT when varying α (β) is reported in Figure 8. We just show the results on four real graphs due to the similar trends. As shown in Figure 8, QueryOPT is far more efficient than Baseline on all datasets under every α (outperforms Baseline by 3-7 orders of magnitude). This is because QueryOPT is a time-optimal algorithm. As α grows, the time cost of Baseline is relatively stable since no matter what α is, Baseline needs to visit the entire graph. The gap between QueryOPT and Baseline increases as α grows. This is because as α grows, the size of $C_{\alpha,\beta}$ decreases and the running time of QueryOPT depends on the size of $C_{\alpha,\beta}$ while that of Baseline depends on the size of input graph. The results when varying β is similar to varying α.

[1] http://konect.uni-koblenz.de/networks

Table 1: Statistic for the graphs

Dataset	Type	$\|U\|$	$\|V\|$	$\|E\|$	$\|G\|$(MB)	$\|\mathbb{I}\|$ (MB)	dmax	$\sqrt{m}$	δ
WC (Wikipedia-en)	Text	1.85M	0.18M	3.80M	45.56	30.10	11,593	1,948	19
FG (Flickr)	Social	0.40M	0.10M	8.55M	70.66	70.61	34,989	2,923	148
EP (Epinions)	Rating	0.12M	0.76M	13.67M	113.63	117.27	162,169	3,697	152
DE (Wikipedia-de)	Authorship	0.43M	3.20M	26.01M	231.50	220.13	278,998	5,100	156
RE (Reuters)	Text	0.78M	0.28M	60.57M	481.52	532.30	345,056	7,782	192
TR (TREC)	Text	0.56M	1.17M	83.63M	666.87	748.74	457,437	9,144	509
DUI (Delicious)	Folksonomy	0.83M	33.78M	101.80M	1,065.71	799.81	29,240	10,089	184
LG (LiveJournal)	Social	3.20M	10.69M	112.31M	985.93	931.60	1,053,676	10,597	109
WT (Web Trackers)	Hyperlink	27.67M	12.76M	140.61M	1,414.34	1,492.43	11,571,952	11,858	438
OG (Orkut)	Affiliation	2.78M	8.73M	327.04M	2,644.93	2,645.46	318,240	18,084	467
PL (Power Law)	Power-law	5M	5M	1,012M	8,080.00	8,003.39	40,354	31,812	374
UD (Uniform Degree)	Uniform-degree	5M	5M	1,067M	8,102.00	8,000.62	277	32,665	169

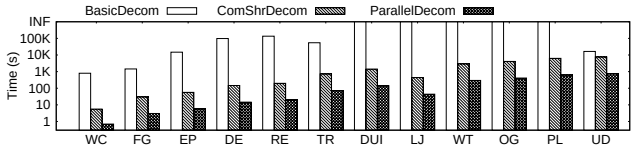


Figure 9: Index construction time for different datasets

6.2 Performance of Index Construction

In this section, we report the size of BiCore-Index for the datasets and evaluate the performance of two index algorithms BasicDecom and ComShrDecom. In this set of experiments, we set the maximum running time for each test as 48 hours. If a test does not stop within the time limit, we denote its processing time as INF.

Exp-3: Index size $|\mathbb{I}|$. The BiCore-Index size $|\mathbb{I}|$ of all the datasets is reported in Table 1. For ease of comparison, we also report the graph size in Table 1. As shown in Table 1, the size of BiCore-Index is linear to the size of its corresponding graph. For example, the size of OG is 2,644.93 MB while the size of its BiCore-Index is 2,645.46 MB. The results are consistent with our theoretical analysis in Section 3.3.

Exp-4: Index construction time for different datasets. In this experiment, we evaluate the time cost for constructing BiCore-Index on different datasets using BasicDecom and ComShrDecom. The results are reported in Figure 9. ComShrDecom is faster than BasicDecom on all datasets and on average achieves over 1000x improvement. For example, in EP, ComShrDecom spends 56 seconds while BasicDecom spends 14,818 seconds.

Exp-5: Comparison of dmax, $\sqrt{m}$ and δ. To better demonstrate the efficiency of BasicDecom and ComShrDecom, we report dmax, $\sqrt{m}$ and δ in Table 1 as these values directly relates to their running time. As shown in Table 1, δ is at least two order of magnitude smaller than dmax for all datasets, which explains the outstanding performance of ComShrDecom. Furthermore, δ is much smaller than $\sqrt{m}$ on real and power-law graphs, which means ComShrDecom hardly runs in worst case and is very efficient in practice. The results confirm our analysis in Section 4 and are consistent with the results in Exp-4.

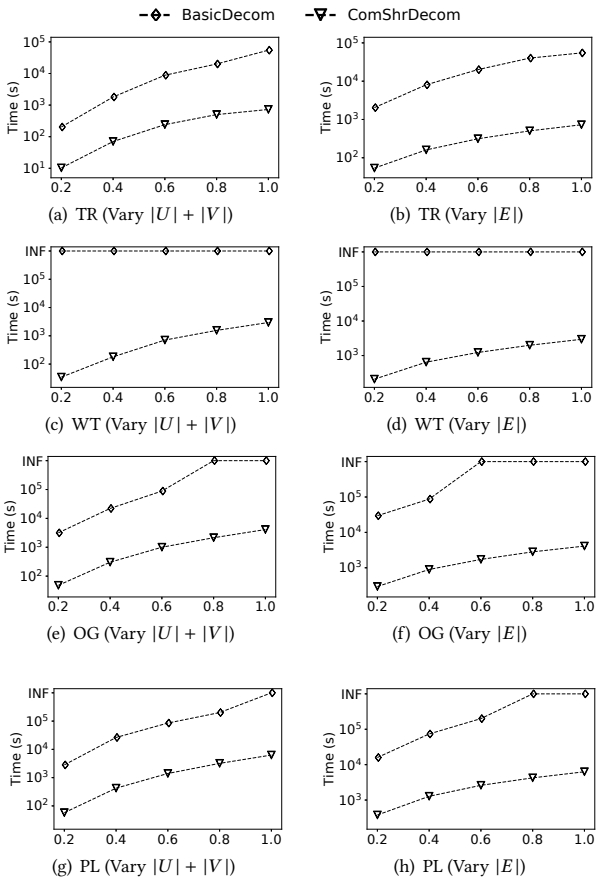


Figure 10: Scalability of index construction algorithms

Exp-6: Scalability of index construction. In this experiment, we evaluate the scalability of BasicDecom and ComShrDecom. To test the scalability, we vary the number of nodes and the number of edges by randomly sampling nodes and edges respectively from 20% to 100% and keeping the induced subgraphs as the input graphs. We only show the results on TR, WT, OG, and PL in Figure 10 since trends are similar on all other datasets. As shown in Figure 10, when varying the number of nodes, the running time for both algorithms

stably increases. ComShrDecom has better performance in all cases and outperforms BasicDecom over two orders of magnitude on average. For example, on WT, the running time of ComShrDecom increases from 35s to 2,953s while BasicDecom cannot terminate within 48 hours for all cases of WT. Varying the number of edges has a similar trend as varying the number of nodes. The results verify that ComShrDecom has a good scalability in practice.

6.3 Dynamic Maintenance and Parallel Construction

In this section, we test the performance of our index maintenance algorithms. We take the algorithm which invokes ComShrDecom to construct BiCore-Index from scratch for each update as the baseline solution. For the baseline solution, the running time is nearly the same for edge insertion and removal, therefore, we just show one result in the figures.

Exp-7: Index maintenance on different datasets. For BiCore-Index-Ins and BiCore-Index-Rem, we randomly remove 5000 distinct existing edges from the graph and report the average processing time for each edge removal. After that, we insert the removed edges back into the graph one by one and report the average processing time for each edge insertion. For BiCore-Index-Batch, we randomly generate 5000 edges and each edge is randomly chosen as insertion or removal. We report the processing time in Figure 11. Generally, the average processing time of our proposed algorithms is much smaller than the baseline solution. For example, on WT, our proposed algorithms BiCore-Index-Ins and BiCore-Index-Rem can handle edge insertion and removal in 31s and 35s respectively while the baseline solution ComShrDecom requires 2,953s. This is because our proposed algorithms are incremental algorithms and save lots of unnecessary computation. Moreover, BiCore-Index-Batch can handle batch update on WT and OG in 292s and 857s respectively.

Exp-8: Parallel index construction. In this experiment, we implement parallel index construction algorithm ParallelDecom using C++11 thread class and test it with 12 cores in default. The running time of ParallelDecom on all datasets is reported in Figure 9. ParallelDecom achieves one order magnitude improvement over BasicDecom. For, example, in WT, the running time of ParallelDecom is 317s while ComShrDecom costs 2953s. We also report performance of ParallelDecom on TR, WT, OG, and PL with different number of cores in Figure 12. The running time of ParallelDecom is almost inversely proportional to the number of cores, which shows that ParallelDecom is efficient in practice.

7 RELATED WORK

Dense Subgraphs in Bipartite Graphs. (α, β)-core is first introduced in [1]. [10] and [11] extend the linear k-core mining algorithm to compute (α, β)-core. [19, 23] generalizes the k-clique on unipartite graph to biclique on bipartite graphs. [32] proves that finding the maximum edge biclique is NP-complete. [53] proposes an efficient algorithm to enumerate all bicliques. [26, 39] relax the definition of biclique to introduce quasi-biclique on bipartite graphs and propose heuristic algorithms to enumerate all quasi-bicliques. [37] defines a framework of bipartite subgraphs based on the butterfly motif (2,2-biclique) to model the dense regions in a hierarchical structure. [34] proposes efficient algorithms for counting the butterfly motif.

Bipartite Graph Models. [16] shows that all complex networks can be decomposed into underlying bipartite structures sharing some important statistics. [17, 29] model bipartite graphs by assigning degree distribution to each node set separately. [20] uses a Markov chain rewiring algorithm to generate bipartite graphs. Preferential attachment process, which is popularly use in generating scale free networks, is studied on bipartite graphs by [17].

Cohesive Subgraph Detection in Unipartite Graphs. Seidman first introduces k-core in [38]. [7] gives an efficient linear-time algorithm for core decomposition. Core decomposition is also studied in weighted graphs [14] and directed graphs [13]. Algorithms for core number maintenance in dynamic graphs are proposed in [35, 36, 54]. Application of k-core can be found in social networks [14, 46], graph visualization[3, 51], protein interaction network analysis [5, 42] and so on. Other cohesive subgraph models are also studied recently, such as clique [43, 44, 48], k-edge connected component [45, 47], and k-mutual-friend subgraph model [50].

8 CONCLUSION

In this paper, we study the problem of efficient (α, β)-core computation. We devise a compact index BiCore-Index whose size can be bounded by $O(m)$. Based on BiCore-Index, we propose an optimal algorithm for (α, β)-core computation and investigate efficient algorithms to construct the index. Moreover, we also discuss about how to maintain the index in dynamic graphs and construct it with parallel algorithm. The experimental results demonstrate the efficiency of our proposed algorithms.

REFERENCES

[1] Adel Ahmed, Vladimir Batagelj, Xiaoyan Fu, Seok-Hee Hong, Damian Merrick, and Andrej Mrvar. 2007. Visualisation and analysis of the Internet movie database. In *Visualization, 2007. APVIS'07. 2007 6th International Asia-Pacific Symposium on*. IEEE, 17–24.

[2] Mohammad Allahbakhsh, Aleksandar Ignjatovic, Boualem Benatallah, Seyed-Mehdi-Reza Beheshti, Elisa Bertino, and Norman Foo. 2013. Collusion Detection in Online Rating Systems. In *Web Technologies and Applications*, Yoshiharu Ishikawa, Jianzhong Li, Wei Wang, Rui Zhang, and Wenjie Zhang (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 196–207.

[3] José Ignacio Alvarez-Hamelin, Luca Dall'Asta, Alain Barrat, and Alessandro Vespignani. 2005. k-core decomposition: A tool for the visualization of large scale networks. *arXiv preprint cs/0504107* (2005).

[4] Sihem Amer-Yahia, Senjuti Basu Roy, Ashish Chawlat, Gautam Das, and Cong Yu. 2009. Group Recommendation: Semantics and Efficiency. *Proc. VLDB Endow.* 2, 1 (Aug. 2009), 754–765. https://doi.org/10.14778/1687627.1687713

[5] Gary D Bader and Christopher WV Hogue. 2003. An automated method for finding molecular complexes in large protein interaction networks. *BMC bioinformatics* 4, 1 (2003), 2.

[6] Albert-László Barabási and Réka Albert. 1999. Emergence of Scaling in Random Networks. *Science* 286, 5439 (1999), 509–512. https://doi.org/10.1126/science.286.5439.509 arXiv:http://science.sciencemag.org/content/286/5439/509.full.pdf

[7] Vladimir Batagelj and Matjaz Zaversnik. 2003. An O (m) algorithm for cores decomposition of networks. *arXiv preprint cs/0310049* (2003).

[8] Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. 2013. Copycatch: stopping group attacks by spotting lockstep behavior in social networks. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 119–130.

[9] Lucas Augusto Montalvão Costa Carvalho and Hendrik Teixeira Macedo. 2013. Users' satisfaction in recommendation systems for groups: an approach based on noncooperative games. In *Proceedings of the 22nd International Conference on World Wide Web*. ACM, 951–958.

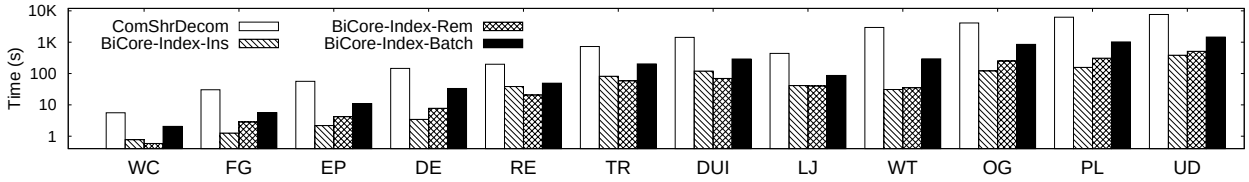


Figure 11: Time cost for index maintenance

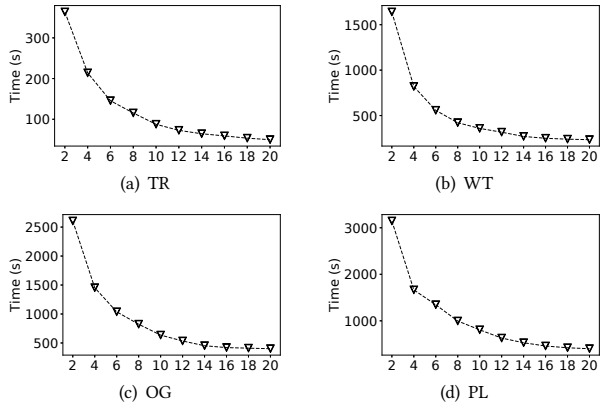


Figure 12: ParallelDecom **with varying number of cores**

[10] Monika CerinÅąek and Vladimir Batagelj. 2015. Generalized two-mode cores. *Social Networks* 42 (2015), 80 – 87. https://doi.org/10.1016/j.socnet.2015.04.001

[11] Danhao Ding, Hui Li, Zhipeng Huang, and Nikos Mamoulis. 2017. Efficient Fault-Tolerant Group Recommendation Using Alpha-beta-core. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. ACM, New York, NY, USA, 2047–2050. https://doi.org/10.1145/3132847.3133130

[12] Mike Gartrell, Xinyu Xing, Qin Lv, Aaron Beach, Richard Han, Shivakant Mishra, and Karim Seada. 2010. Enhancing group recommendation by incorporating social relationship interactions. In *Proceedings of the 16th ACM international conference on Supporting group work*. ACM, 97–106.

[13] Christos Giatsidis, Dimitrios M Thilikos, and Michalis Vazirgiannis. 2011. D-cores: Measuring collaboration of directed graphs based on degeneracy. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, 201–210.

[14] Christos Giatsidis, Dimitrios M Thilikos, and Michalis Vazirgiannis. 2011. Evaluating cooperation in communities with the k-core structure. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*. IEEE, 87–93.

[15] Jagadeesh Gorla, Neal Lathia, Stephen Robertson, and Jun Wang. 2013. Probabilistic group recommendation via information matching. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 495–504.

[16] Jean-Loup Guillaume and Matthieu Latapy. 2004. Bipartite structure of all complex networks. *Information processing letters* 90, 5 (2004), 215–221.

[17] Jean-Loup Guillaume and Matthieu Latapy. 2006. Bipartite graphs as models of complex networks. *Physica A: Statistical Mechanics and its Applications* 371, 2 (2006), 795–813.

[18] S. Gunnemann, E. Muller, S. Raubach, and T. Seidl. 2011. Flexible Fault Tolerant Subspace Clustering for Data with Missing Values. In *2011 IEEE 11th International Conference on Data Mining*. 231–240. https://doi.org/10.1109/ICDM.2011.70

[19] Dorit S Hochbaum. 1998. Approximating clique and biclique problems. *Journal of Algorithms* 29, 1 (1998), 174–200.

[20] Ravi Kannan, Prasad Tetali, and Santosh Vempala. 1997. Simple Markov-chain algorithms for generating bipartite graphs and tournaments. In *Proceedings of the eighth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 193–200.

[21] Mehdi Kaytoue, Sergei O Kuznetsov, Amedeo Napoli, and Sébastien Duplessis. 2011. Mining gene expression data with pattern structures in formal concept analysis. *Information Sciences* 181, 10 (2011), 1989–2001.

[22] Wissam Khaouid, Marina Barsky, Venkatesh Srinivasan, and Alex Thomo. 2015. K-core decomposition of large networks on a single PC. *Proceedings of the VLDB Endowment* 9, 1 (2015), 13–23.

[23] Sune Lehmann, Martin Schwartz, and Lars Kai Hansen. 2008. Biclique communities. *Phys. Rev. E* 78 (Jul 2008), 016108. Issue 1. https://doi.org/10.1103/PhysRevE.78.016108

[24] Michael Ley. 2002. The DBLP Computer Science Bibliography: Evolution, Research Issues, Perspectives. In *Proc. Int. Symposium on String Processing and Information Retrieval*. 1–10.

[25] Jinyan Li, Kelvin Sim, Guimei Liu, and Limsoon Wong. [n. d.]. *Maximal Quasi-Bicliques with Balanced Noise Tolerance: Concepts and Co-clustering Applications*. 72–83. https://doi.org/10.1137/1.9781611972788.7 arXiv:http://epubs.siam.org/doi/pdf/10.1137/1.9781611972788.7

[26] Jinyan Li, Kelvin Sim, Guimei Liu, and Limsoon Wong. 2008. Maximal quasi-bicliques with balanced noise tolerance: Concepts and co-clustering applications. In *Proceedings of the 2008 SIAM International Conference on Data Mining*. SIAM, 72–83.

[27] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 1 (2003), 76–80.

[28] Xiaowen Liu, Jinyan Li, and Lusheng Wang. 2010. Modeling Protein Interacting Groups by Quasi-Bicliques: Complexity, Algorithm, and Application. *IEEE/ACM Trans. Comput. Biol. Bioinformatics* 7, 2 (April 2010), 354–364. https://doi.org/10.1109/TCBB.2008.61

[29] Mark EJ Newman, Steven H Strogatz, and Duncan J Watts. 2001. Random graphs with arbitrary degree distributions and their applications. *Physical review E* 64, 2 (2001), 026118.

[30] Eirini Ntoutsi, Kostas Stefanidis, Kjetil Nørvåg, and Hans-Peter Kriegel. 2012. Fast group recommendations by applying user clustering. In *International Conference on Conceptual Modeling*. Springer, 126–140.

[31] Eirini Ntoutsi, Kostas Stefanidis, Katharina Rausch, and Hans-Peter Kriegel. 2014. "Strength Lies in Differences": Diversifying Friends for Recommendations Through Subspace Clustering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM '14)*. ACM, New York, NY, USA, 729–738. https://doi.org/10.1145/2661829.2662026

[32] René Peeters. 2003. The maximum edge biclique problem is NP-complete. *Discrete Applied Mathematics* 131, 3 (2003), 651–654.

[33] Ardian Kristanto Poernomo and Vivekanand Gopalkrishnan. 2009. Towards Efficient Mining of Proportional Fault-tolerant Frequent Itemsets. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*. ACM, New York, NY, USA, 697–706. https://doi.org/10.1145/1557019.1557097

[34] Seyed-Vahid Sanei-Mehri, Ahmet Erdem Sariyuce, and Srikanta Tirthapura. 2018. Butterfly Counting in Bipartite Networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2150–2159.

[35] Ahmet Erdem Saríyüce, Buğra Gedik, Gabriela Jacques-Silva, Kun-Lung Wu, and Ümit V Çatalyürek. 2013. Streaming algorithms for k-core decomposition. *Proceedings of the VLDB Endowment* 6, 6 (2013), 433–444.

[36] Ahmet Erdem Sarıyüce, Buğra Gedik, Gabriela Jacques-Silva, Kun-Lung Wu, and Ümit V Çatalyürek. 2016. Incremental k-core decomposition: algorithms and evaluation. *The VLDB JournalâĂŤThe International Journal on Very Large Data Bases* 25, 3 (2016), 425–447.

[37] Ahmet Erdem Sarıyüce and Ali Pinar. 2018. Peeling Bipartite Networks for Dense Subgraph Discovery. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 504–512. https://doi.org/10.1145/3159652.3159678

[38] Stephen B Seidman. 1983. Network structure and minimum degree. *Social networks* 5, 3 (1983), 269–287.

[39] Kelvin Sim, Jinyan Li, Vivekanand Gopalkrishnan, and Guimei Liu. 2006. Mining maximal quasi-bicliques to co-cluster stocks and financial ratios for value investment. In *Data Mining, 2006. ICDM'06. Sixth International Conference on*. IEEE, 1059–1063.

[40] Jun Wang, Arjen P De Vries, and Marcel JT Reinders. 2006. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 501–508.

[41] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale Commodity Embedding for E-commerce Recommendation

in Alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. ACM, New York, NY, USA, 839–848. https://doi.org/10.1145/3219819.3219869

[42] Stefan Wuchty and Eivind Almaas. 2005. Peeling the yeast protein network. *Proteomics* 5, 2 (2005), 444–449.

[43] Long Yuan, Lu Qin, Xuemin Lin, Lijun Chang, and Wenjie Zhang. 2015. Diversified top-k clique search. In *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015*. 387–398. https://doi.org/10.1109/ICDE.2015.7113300

[44] Long Yuan, Lu Qin, Xuemin Lin, Lijun Chang, and Wenjie Zhang. 2016. Diversified top-k clique search. *VLDB J.* 25, 2 (2016), 171–196. https://doi.org/10.1007/s00778-015-0408-z

[45] Long Yuan, Lu Qin, Xuemin Lin, Lijun Chang, and Wenjie Zhang. 2016. I/O Efficient ECC Graph Decomposition via Graph Reduction. *PVLDB* 9, 7 (2016), 516–527. https://doi.org/10.14778/2904483.2904484

[46] Long Yuan, Lu Qin, Xuemin Lin, Lijun Chang, and Wenjie Zhang. 2017. Effective and Efficient Dynamic Graph Coloring. *PVLDB* 11, 3 (2017), 338–351. https://doi.org/10.14778/3157794.3157802

[47] Long Yuan, Lu Qin, Xuemin Lin, Lijun Chang, and Wenjie Zhang. 2017. I/O efficient ECC graph decomposition via graph reduction. *VLDB J.* 26, 2 (2017), 275–300. https://doi.org/10.1007/s00778-016-0451-4

[48] Long Yuan, Lu Qin, Wenjie Zhang, Lijun Chang, and Jianye Yang. 2018. Index-Based Densest Clique Percolation Community Search in Networks. *IEEE Trans. Knowl. Data Eng.* 30, 5 (2018), 922–935. https://doi.org/10.1109/TKDE.2017.2783933

[49] Quan Yuan, Gao Cong, and Chin-Yew Lin. 2014. COM: A Generative Model for Group Recommendation. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, New York, NY, USA, 163–172. https://doi.org/10.1145/2623330.2623616

[50] Fan Zhang, Long Yuan, Ying Zhang, Lu Qin, Xuemin Lin, and Alexander Zhou. 2018. Discovering Strong Communities with User Engagement and Tie Strength. In *Database Systems for Advanced Applications - 23rd International Conference, DASFAA 2018, Gold Coast, QLD, Australia, May 21-24, 2018, Proceedings, Part I*. 425–441. https://doi.org/10.1007/978-3-319-91452-7_28

[51] Yang Zhang and Srinivasan Parthasarathy. 2012. Extracting analyzing and visualizing triangle k-core motifs within networks. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. IEEE, 1049–1060.

[52] Yun Zhang, Charles A. Phillips, Gary L. Rogers, Erich J. Baker, Elissa J. Chesler, and Michael A. Langston. 2014. On finding bicliques in bipartite graphs: a novel algorithm and its application to the integration of diverse biological data types. *BMC Bioinformatics* 15, 1 (15 Apr 2014), 110. https://doi.org/10.1186/1471-2105-15-110

[53] Yun Zhang, Charles A Phillips, Gary L Rogers, Erich J Baker, Elissa J Chesler, and Michael A Langston. 2014. On finding bicliques in bipartite graphs: a novel algorithm and its application to the integration of diverse biological data types. *BMC bioinformatics* 15, 1 (2014), 110.

[54] Yikai Zhang, Jeffrey Xu Yu, Ying Zhang, and Lu Qin. 2017. A Fast Order-Based Approach for Core Maintenance. In *33rd IEEE International Conference on Data Engineering, ICDE 2017, San Diego, CA, USA, April 19-22, 2017*. 337–348. https://doi.org/10.1109/ICDE.2017.93

Neural Variational Correlated Topic Modeling

Luyang Liu
Department of Computer Sciences,
Beijing Institute of Technology;
Beijing Engineering Research Center
of High Volume Language
Information Processing and Cloud
Computing Applications
Beijing, China
lly_aegis@foxmail.com

Heyan Huang*
Department of Computer Sciences,
Beijing Institute of Technology;
Zhejiang Lab
Beijing, China

Yang Gao
Department of Computer Sciences,
Beijing Institute of Technology
Beijing, China
gyang@bit.edu.cn

Xiaochi Wei
Baidu Inc.
Beijing, China
weixiaochi@baidu.com

Yongfeng Zhang
Department of Computer Sciences,
Rutgers University
New Brunswick, USA
yongfeng.zhang@rutgers.edu

ABSTRACT

With the rapid development of the Internet, millions of documents, such as news and web pages, are generated everyday. Mining the topics and knowledge on them has attracted a lot of interest on both academic and industrial areas. As one of the prevalent unsupervised data mining tools, topic models are usually explored as probabilistic generative models for large collections of texts. Traditional probabilistic topic models tend to find a closed form solution of model parameters and approach the intractable posteriors via approximation methods, which usually lead to the inaccurate inference of parameters and low efficiency when it comes to a quite large volume of data. Recently, an emerging trend of neural variational inference can overcome the above issues, which offers a scalable and powerful deep generative framework for modeling latent topics via neural networks. Interestingly, a common assumption for the most neural variational topic models is that topics are independent and irrelevant to each other. However, this assumption is unreasonable in many practical scenarios. In this paper, we propose a novel Centralized Transformation Flow to capture the correlations among topics by reshaping topic distributions. Furthermore, we present the Transformation Flow Lower Bound to improve the performance of the proposed model. Extensive experiments on two standard benchmark datasets have well-validated the effectiveness of the proposed approach.

CCS CONCEPTS

• **Information systems → Document topic models**; • **Computing methodologies → Natural language processing**.

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313561

KEYWORDS

Natural language processing;topic model;neural variational inference

ACM Reference Format:
Luyang Liu, Heyan Huang, Yang Gao, Xiaochi Wei, and Yongfeng Zhang. 2019. Neural Variational Correlated Topic Modeling. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313561

1 INTRODUCTION

Nowadays, the numerous texts, such as online news and reports, are generated by various kinds of daily web services. Mining and organizing large number of topics and knowledge from them has attracted much attention [2, 16, 28, 33, 35]. As one of the most prevalent unsupervised algorithms in data mining and natural language processing, topic model has become such a successful technique to discover topics from collections. The conventional topic models, such as Latent Dirichlet Allocation (LDA) [9], offer a practical and explainable probability generative process for modeling topics. In the LDA, topic distribution of a document is drawn as a multinomial with Dirichlet prior, and document words are drawn from their topical words distribution. Then many LDA-like probability topic models are proposed. However, since generative process is getting expressive as the data grows, the inference of topic models tends to be challenging and tricky due to the complicated intractable marginal likelihood.

Recent efforts in neural variational inference (NVI) [23] offers a powerful auto-encoding framework to handle with large amount of data, and it can also facilitate topic modeling. The advantage of the NVI is that it replaces the arduous works on inference of probabilistic models by applying flexible and powerful neural networks through stochastic back propagation [31]. In addition, the NVI can be automatically applied to a new model with a simple declarative specification of the generative process. Based on the NVI framework, several neural variational topic models are proposed, such as Neural Variational Document Model (NVDM) [23],

Neural Variation Latent Dirichlet Allocation (NVLDA) [34], and Gaussian Softmax Model (GSM) [22]. To reduce the computation complexity, their topics are always modeled as isotropic Gaussian distributions via inference network.

The covariance matrix of the isotropic Gaussian is a diagonal matrix for topical representations. It means that topics are solely independent to each other. Yet, in reality, topics in documents are usually correlated. For instance, the topics about "hardware" are probably related with topics about "software" and " company". Therefore, it is reasonable to expect the relationships among different topics can be modeled to expand the limitations of the classical neural variational topic models in which topics are generated by the isotropic Gaussian distribution. Intuitively, the isotropic Gaussian topic distribution can be replaced by a Gaussian distribution with full covariance matrix, so that the topic correlations can be captured.

Similar challenges also exist in the field of computer vision. Several efforts have been dedicated to solving this independent problem, such as Normalizing flow [30], Inverse auto regressive flow [18], and Householder flow [37]. These flow-based methods map an isotropic Gaussian samples into full covariance one by using several invertible transformations. Among them, the most efficient one is Householder Flow since it only involves linear transformations compared with the other flow-based methods [37]. However, Householder flow cannot be directly applied to the topic models designed for NLP tasks. Because Householder flow can only transform single stochastic Gaussian sample into non-isotropic one, which is obviously insufficient in computing loss function.

To tackle with the aforementioned problems, in this paper, we first propose a Centralized Transformation Flow (CTF) to make the inference network capable of modeling covariance matrix of latent distribution. Based on this, we further propose a Neural Variational Correlated Topic Model (NVCTM) that integrates the CTF with multinomial softmax generative process. The proposed NVCTM can enhance the capability of capturing the correlations among topics with the assistance of the new CTF process. In particular, inspired by involutory property of Householder matrix and linear property of Gaussian, the proposed CTF utilizes the product of several Householder transformations to get final transformation matrix. Then the isotropic Gaussian samples can be transformed into non-isotropic samples via the linear operator. Finally, the document is reconstructed by the multinomial softmax generative model given correlated topic vectors. Besides, to effectively infer with Centralized Transformation Flow, we also present the Transformation Flow Lower Bound (TFLB) to regulate KL divergence term of the objective function. The TFLB utilizes the linear property of Gaussian distribution and gives the closed form KL divergence between encoded non-isotropic Gaussian distribution and topic prior. To evaluate the proposed methods, we conduct experiments on two standard datasets. The experimental results indicate the effectiveness of the correlated topic modeling via the CTF and the TFLB is an appropriate lower bound for the inference of the NVCTM. In summary, this paper makes following contributions:

(1) We propose the CTF for inference network, which is capable of transforming isotropic Gaussian distribution samples into Gaussian samples with full covariance matrix.
(2) We propose the NVCTM model, which is able to capture the correlations among topics via the proposed CTF process.
(3) We design the new TFLB which gives an appropriate KL divergence term of the NVCTM and is capable to effectively facilitate the training of the NVCTM.

The rest of this paper is organized as follows. Section 2 focuses on the related works. In Section 3, we briefly introduce the background knowledge of neural variational topic models and Householder flow. In Section 4, we propose our NVCTM and present its inference process. In Section 5, we introduce the dataset, options of experiments, evaluating metric, results and analysis. Finally, the conclusion of this work is made in Section 6.

2 RELATED WORKS

2.1 Correlated Topic Models

Latent Dirichlet Allocation [9] is one of the most popular approaches to text analysis community. In LDA, the multinomial-based topic distribution cannot model correlations between topics. To solve this issue, a CTM [7] replaces the component-independent Dirichlet topic prior with the full covariance Gaussian to capture the correlations. Similarly, many approaches are continually proposed. The Gaussian Process Topic Model (GPTM) [1] captures the correlations among document collection and adds known similarities among documents via Gaussian process. The above CTM and GPTM models capture the topic correlations by reconstructing topic distributions. Another kind of approaches, such as Gaussian-LDA [11] and Correlated Gaussian Topic Model (CGTM) [38], leverage external resources to model the topic correlations, such as computing relations by embedding-based representations [24, 25]. Notably, for most of the correlation-based topic models, Gaussian distribution is often adopted. It means that it is flexible to model dependent distributions and easy to capture correlations among topics. Traditional topic models utilize directed probability graph to describe their generative processes. Their training methods often adopt sampling methods [3, 27] and variational inference [6, 15]. These training methods requires closed form solution of deviations for updating model parameters to approximate intractable posterior. However, as the expressiveness and structure of generative processes grows, the deviation of parameters tends to be tough and complicated, which also hinders the model's efficiency when it comes to a large scale.

2.2 Neural Variational Topic Models

In recent years, the advances in stochastic variational inference and neural networks brings about the idea of using neural networks for parameter inference. Neural variational inference (NVI) approach [23] makes variational auto- encoders (VAEs) [19] a powerful deep generative framework for topic modeling. In VAEs, the flexible neural network is served as a estimator of the target distribution, which eliminates those arduous and complicated mathematical deviations. Neural variational document model (NVDM) [23] is a typical neural variational topic model with VAEs-like architecture. The NVDM consists of two parts: an inference network to parameterize latent topic distributions and a multinomial softmax generative model to reconstruct the document based on the topic vectors from latent

topic distribution. Similarly, neural variational latent Dirichlet allocation (NVLDA) [34] implements an isotropic Gaussian topic distribution under the Laplace approximation [12]. Gaussian Softmax Model (GSM) [22] normalizes the topic representation of NVDM to get topic proportion of each document. These neural variational approaches share a same drawback: topic distribution is assumed to be an isotropic Gaussian, which makes them incapable of modeling topic correlations.

2.3 Flow-based Methods for VAEs

Recent efforts have been conducted to improve latent variable distribution for neural variational inference. For instance, the flow-based methods such as Normalizing flow [30], which implements a series of invertible functions to transform the original latent variables. Householder flow [37] further utilizes a series of Householder transformation to model the orthogonal matrix and transforms an isotropic Gaussian sample into a non-isotropic one. Additionally, several more complicated flow-based methods, i.e., Hamiltonian Variational Inference (HVI) [32], and linear Inverse Autoregressive Flow (IAF) [18], are also proposed. Among these flow-based methods, the Householder flow is the most efficient one that only involves several linear transformations.

In this paper, our model overcomes the drawbacks of above works and consider both neural variational network and correlations among topics. It is noteworthy that our work introduces neural variational correlated topic model by bringing a new concept of *centralized transformation flow* and enhances the flexibility and interpretations of topic models for text analysis.

3 PRELIMINARIES

In this section, we firstly introduce a most classical framework of neural variational topic mdoel, Neural Variational Document Model (NVDM). Then, we briefly introduce Householder flow. The notations and symbols frequently used in this paper are displayed in Table 1.

Table 1: Notations in our model.

Name	Description
$\mathbf{x}$	Bag-of-word document vector
$\mathbf{h}^{(0)}$	Latent isotropic Gaussian topic vector
k	Length of CTF
$\mathbf{h}^{(k)}$	Latent non-isotropic topic vector after k length of CTF
$\mathbf{H}_i$	$i-th$ Householder matrix in Householder flow
π	Output vector from MLP in Inference network
μ	Mean vector of isotropic Gaussian topic distribution
Σ	Diagonal covariance matrix of isotropic Gaussian topic distribution
$\mathbf{U}$	Transformation matrix of CTF
V	Length of vocabulary
θ	Parameter set of inference network
γ	Parameter set of generative model

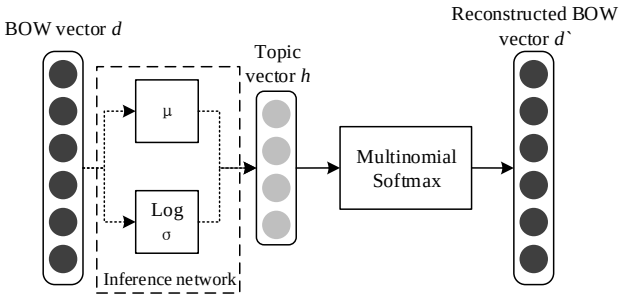


Figure 1: Schematic representation of NVDM.

3.1 Neural Variational Document Model

In traditional probability topic models, the inference of models often requires tricky and complicated math deviations for intractable posterior when the expressiveness of models' generative process grows. To fulfill efficient inference and learning in probabilistic topic models, neural variational inference (NVI) is introduced. In NVI, the inference of model parameters relies on neural networks to approximate the intractable posterior and only limited deviations of parameters are needed.

Neural Variational Document Model (NVDM) [23] is a typical NVI topic model. The schematic representation of NVDM is displayed in figure 1. Each input document is encoded into an isotropic Gaussian distribution via inference network. Then topic vectors are drawn from the Gaussian distributions and passed into a multinomial softmax generative model to reconstruct the input document. Specifically, let $\mathbf{d}$ indicate the bag-of-words document vector, which is the input of inference network; π denotes the output vector of multilayer perceptron (MLP) in inference network. The isotropic Gaussian topic distribution $\mathcal{N}(\mu, \Sigma)$ is parameterized by μ and σ. $\mathbf{l}_1(\cdot)$ and $\mathbf{l}_2(\cdot)$ are two linear neural networks.

$$\begin{aligned} \pi &= MLP(\mathbf{d}) \\ \mu &= \mathbf{l}_1(\pi) \\ \log \sigma &= \mathbf{l}_2(\pi) \\ \Sigma &= diag(\exp 2{\cdot}\mathbf{l}_2(\pi)) \end{aligned} \tag{1}$$

For M documents in corpus and N words in each document, the document generative process of NVDM then can be described as follows:

(1) Topic vector $\mathbf{h} \sim N(\mu, \Sigma)$.
(2) For each word in document $w_n \sim p(w_n|\mathbf{h})$. The definition of $p(w_n|\mathbf{h})$ is given by Equation 2.

$$p(w_n|\mathbf{h}) = \frac{exp\{-\mathbf{h}Wd_n - \mathbf{b}_{d_n}\}}{\sum_{j=1}^{|V|}\{\mathbf{h}Wd_j + \mathbf{b}_{d_i}\}} \tag{2}$$

The training objective of NVDM is to maximize the lower bound of marginal likelihood given in Equation 3.

$$L_{NVDM} = E_q[\log p(\mathbf{d}|\mathbf{h})] - KL[q(\mathbf{h}|\mathbf{d}) \parallel p(\mathbf{h})] \tag{3}$$

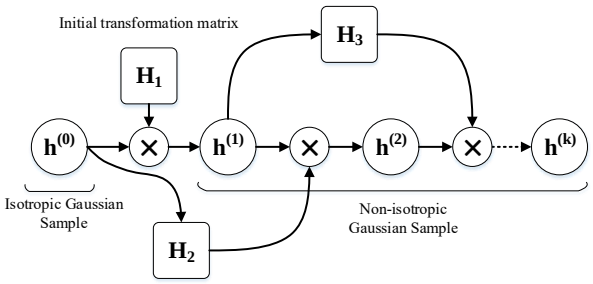


Figure 2: Schematic representation of Householder flow.

The objective function, which is also known as evidence lower bound (ELBO), has two components, i.e., the negative reconstruction term and the KL divergence term. The reconstruction term is mainly determined by generative model and KL divergence term denotes the measurement between the encoded isotropic Gaussian and its prior. The objective of training is equivalent to minimize reconstruction loss as well as KL divergence.

3.2 Householder flow

Previous work [37] introduces Household flow to break independent assumption in latent distributions.

Householder flow utilizes unitary and involutary properties of Householder matrix and establishes iterative invertible linear transformations to map an isotropic Gaussian sample into a non-isotropic one.

Intuitively, to transform an isotropic Gaussian sample into a full covariance one, an orthogonal matrix is needed. Generally, for any Gaussian sample $\mathbf{h}$ $N(\mu, \Sigma)$, it can be generated from a standard Gaussian distribution via $\mathbf{h} = \mu + \Sigma^{\frac{1}{2}} \cdot \epsilon, \epsilon \sim \mathcal{N}(0, I)$ where μ, Σ are parameters of Gaussian distribution. The Σ in isotropic Gaussian is a diagonal matrix. To model an orthogonal matrix, Householder flow can be represented as the product of k Householder transformation [5, 36]. Therefore, according to [37], the Householder matrix $\mathbf{H_i}$ is calculated as

$$\mathbf{H_t} = \begin{cases} \mathbf{I} - 2\dfrac{\pi \cdot (\pi)^{\mathbf{T}}}{\| \pi \|^2}, & t = 1 \\ \mathbf{I} - 2\dfrac{\mathbf{h^{(t-1)}} \cdot (\mathbf{h^{(t-1)}})^{\mathbf{T}}}{\| \mathbf{h^{(t-1)}} \|^2}, & t \geq 2 \end{cases} \tag{4}$$

where $\mathbf{h^{(1)}}$ is a random sample from isotropic Gaussian distribution. Householder matrix, mathematically, refers to an unitary, Hermitian and involutory matrix. Therefore, the product of Householder matrices can be simplified in terms of those useful properties. The vector π is the output vector of MLP in the inference network. We can iteratively get the transformed non-isotropic Gaussian sample via the following equation.

$$\mathbf{h^{(m)}} = \begin{cases} \mathbf{h^{(1)}} & , m = 1 \\ \mathbf{H_{m-1}} \quad \cdot \mathbf{h^{(m-1)}}, & m \geq 2 \end{cases} \tag{5}$$

The schematic representation of the iterative process of Householder flow is illustrated in figure 2. From figure, we can see that Householder Flow only maps single isotropic Gaussian sample to non-isotropic one. However, this is obviously insufficient in computing reconstruction loss mentioned at Equation 1.

4 OUR MODEL

In this section, we describe the proposed Neural Variational Correlated Topic Model (NVCTM) in details. Generally, the structure of NVCTM consists of two main parts, i.e., the inference network with Centralized Transformation Flow and the multinomial softmax generative model. Specifically, the proposed Centralized Transformation Flow in inference network first generates a distributional transformation matrix. The transformation matrix is product of several Householder matrices. The isotropic Gaussian samples are then transformed into non-isotropic ones by multiplying CTF's transformation matrix. The multinomial softmax generative model reconstructs the given documents from the topic vectors that are drawn from the non-isotropic Gaussian distribution. In the following subsections, we introduce details of the Centralized Transformation Flow and the proposed correlated topic modeling. The whole procedure of the model is illustrated in figure 3.

4.1 Centralized Transformation Flow

To transform multiple isotropic stochastic samples into non-isotropic ones, a distribution-level transformation matrix is needed. Therefore, we need to generate it.

Specifically, for each isotropic latent distribution, we choose the mean vector μ as the input vector to generate series of Householder matrices like the iterative process in Householder flow. The iteration process for Householder matrices is given by Equation 4 and Equation 6 where $\pi = MLP(\mathbf{x})$ is used for generating initial Householder matrix $\mathbf{H}_1$ and $MLP(\mathbf{x})$ is the output of MLP inference network.

$$\mathbf{v_m} = \begin{cases} \mu & , m = 1 \\ \mathbf{H_{m-1}} \quad \cdot \mathbf{v_{m-1}}, & m \geq 2 \end{cases} \tag{6}$$

For given length of flow k, the distributional transformation matrix $\mathbf{U}$ can be computed via:

$$\mathbf{U} = \prod_{i=1}^{k} \mathbf{H_i}, where \mathbf{H}_i \in \{\mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3, \cdots, \mathbf{H}_k\}. \tag{7}$$

Then we take advantage of the linear property of Gaussian. The final non-isotropic Gaussian samples can be computed via linear transformation mentioned in Equation 8.

$$\begin{aligned} \mathbf{h}^{(0)} &\sim N(\mu, \Sigma) \\ \mathbf{h}^{(k)} &= \mathbf{h}^{(0)} \cdot \mathbf{U} \\ \mathbf{h}^{(k)} &\sim N(\mathbf{U}\mu, \mathbf{U}\Sigma\mathbf{U}) \end{aligned} \tag{8}$$

The procedure of k-th Centralized Transformation flow is depicted in the upper part of figure 3.

Householder flow can only transform single isotropic Gaussian into non-isotropic one, which leads to insufficient approximating accurate the intractable posterior. While in CTF, when transformation matrix is defined, CTF is able to map multiple isotropic Gaussian samples into the non-isotropic ones with linear transformation.

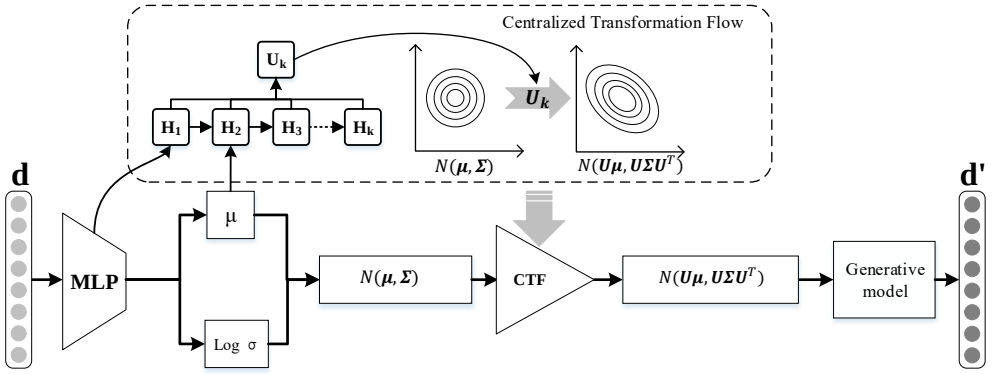


Figure 3: Schematic representation of NVCTM. The dashed rectangle indicates the procedure of CTF. The dashed arrow is short for the iterative process to generate $\mathbf{h^{(k)}}$.

4.2 Correlated Topic Modeling via Centralized Transformation Flow

The isotropic Gaussian topic distribution is unable to capture the correlation between topics. To address this issue, we propose our NVCTM which incorporates the proposed CTF to capture the correlation between topics.

In NVCTM, the proposed CTF is employed in the inference network of the NVCTM to transform the isotropic Gaussian topic distribution into a Gaussian with full covariance matrix. Then the multinomial softmax generative model generates the documents with the correlated topic vectors. The structure of NVCTM denotes in figure 3. The observed bag-of-word vector $\mathbf{d}$ is an input of MLP in inference network. The CTF then utilizes the output of MLP and the mean vector of isotropic Gaussian distribution as initial values to generate the distributional transformation matrix. The correlated topic vectors can be computed via multiplication between isotropic Gaussian topic vectors and transformation matrix in CTF. Finally, the multinomial softmax generative model takes several correlated topic vectors to reconstruct the input document vector $\mathbf{d'}$. After the employment of CTF, the topic distribution in NVCTM is a Gaussian with fully covariance matrix and the topic correlations can be captured. The objective function of NVCTM is denoted as

$$\begin{aligned} &L_{NVCTM} \\ &= E_q[\log p(\mathbf{d}|\mathbf{h^{(k)}})] - KL[q(\mathbf{h^{(k)}}|\mathbf{d})||p(\mathbf{h})] \\ &= E_{q\in N(\mathrm{U}\mu,\mathrm{U}\Sigma\mathrm{U})}[\log p(\mathbf{d}|\mathbf{h^{(k)}})] - KL[N(\mathrm{U}\mu,\mathrm{U}\Sigma\mathrm{U})||N(0,\mathbf{I})]. \end{aligned} \tag{9}$$

The negative reconstruction loss can be automatically computed via stochastic sampling multiple samples from the latent topic space. With the help of reparameterization trick, the whole objective function can be trained with stochastic gradient methods. Accordingly, the proposed CTF refines the latent topic distribution, which indicates that the KL divergence term needs to be modified. In the next subsection, we will precisely describe the deviation of new KL divergence term after applying CTF.

4.3 Inference with Centralized Transformation Flow

After employing CTF, the model inference needs to be modified accordingly. The conventional approach of estimating objective function when flow-based methods applied in VAEs usually refers to the Flow-based Free Energy lower Bound (FELB) [30]. It is calculated as

$$\begin{aligned} L_{FELB} = E_q[\log p(\mathbf{d}|\mathbf{h^{(k)}}) + \sum_{t=1}^{k} \log|det\frac{\partial f^{(t)}}{\partial h^{(t-1)}}|] \\ - KL[q(\mathbf{h^{(0)}}|\mathbf{d})||p(\mathbf{h^{(k)}})], \end{aligned} \tag{10}$$

where $\mathbf{h^{(0)}}$ indicates the isotropic Gaussian sample. In Householder flow, $\log|det\frac{\partial f^{(t)}}{\partial h^{(t-1)}}| = \log det|\mathbf{H_t}|$ equals to zero because Householder matrix is a unitary matrix. In CTF, $\log|det\frac{\partial f^{(t)}}{\partial h^{(t-1)}}| = \log det|\mathbf{U}|$ equals to zero for $\mathbf{U} = \prod_{i=1}^{k}\mathbf{H_i}$, where $\mathbf{H_i}$ is the Householder matrix. Therefore, in the NVCTM, the FELB is calculated as

$$\begin{aligned} L_{FELB} &= E_q[\log p(\mathbf{d}|\mathbf{h^{(k)}}) - KL[q(\mathbf{h^{(0)}}|\mathbf{d})||p(\mathbf{h})] \\ &= E_q[\log p(\mathbf{d}|\mathbf{h^{(k)}})] + 0.5[n - \mu^2 + |\Sigma| + \log|\Sigma|] \end{aligned}$$

where μ and Σ are in Equation 1. $p(\mathbf{h})$ is the prior, which is usually considered as $\mathcal{N}(0, \mathbf{I})$.

However, during experiments, we observes that FELB slows down the training of the model. Here, we present our Transformation Flow Lower Bound (TFLB) to improve the performance of perplexity and facilitate the training process of model. The TFLB utilizes the linear property of Gaussian and can assist get explicit parameters of the final non-isotropic Gaussian distribution. The linear property of Gaussian [26] can be described as follows : Assuming that the random variable and its distribution are given by $\mathbf{h_0} \sim \mathcal{N}(\mu, \Sigma)$, we can transform it into latent variable $\mathbf{h'} = \mathrm{U} \cdot \mathbf{h}$. The corresponding transformed latent distribution is denoted as

$$\mathbf{h'} \sim \mathcal{N}(\mathrm{U}\mu, \mathrm{U}\Sigma\mathrm{U^T}) \tag{11}$$

Then the KL divergence term in the objective function can be computed as

$$\begin{aligned} & KL[\mathcal{N}(\mu_1, \Sigma_1) \parallel \mathcal{N}(\mu_2, \Sigma_2)] \\ =& \frac{1}{2}[\log \frac{|\Sigma_2|}{|\Sigma_1|} - n + \mathrm{tr}(\Sigma_2^{-1} \cdot \Sigma_1) \\ & + (\mu_2 - \mu_1)^{\mathrm{T}} \Sigma_2^{-1} (\mu_2 - \mu_1)], \end{aligned} \tag{12}$$

where n is the number of dimensionality in multivariate Gaussian. Due to the fact of unitary property of Householder matrix, the regularization term of TFLB then can be denoted as

$$\begin{aligned} & L_{TFLB} \\ & = E_q[\log p(\mathbf{d}|\mathbf{h}^{(\mathbf{k})})] - KL[q(\mathbf{h}^{(\mathbf{k})}|\mathbf{d}) \parallel p(\mathbf{h})] \\ & = E_q[\log p(\mathbf{d}|\mathbf{h}^{(\mathbf{k})})] - \frac{1}{2}[-\log|\Sigma| - n + \\ & tr(\mathbf{U}\Sigma\mathbf{U}^{\mathrm{T}}) + (\mathbf{U}\mu)^{\mathrm{T}}\mathbf{U}\mu]. \end{aligned} \tag{13}$$

At last, the pseudo code of our method, NVCTM, is depicted in Algorithm 1.

Algorithm 1 NVCTM

Input: Document bag-of-word vector $\mathbf{d}$; document collection $\mathbf{D}$; parameters set of inference network θ; parameter set of generative model γ; length of CTF k

Output: model parameters θ and γ

Remove stop words and low-frequency words and covert document into bag-of-word vector
Initialize θ and γ
for $\mathbf{d} \in \mathbf{D}$ **do**
    Compute μ and Σ using Eq 1.
    Generate $\mathbf{H}_1, \mathbf{H}_1, \mathbf{H}_3, \cdots, \mathbf{H}_k$ via Eq 4 and Eq 6.
    Generate matrix $\mathbf{U}$ by Eq 7.
    Randomly draw 20 topic vector samples h from $N(\mu, \Sigma)$.
    Multiply $\mathbf{U}$ and $\mathbf{h}^{(0)}$ to get $\mathbf{h}^{(k)}$.
    Compute objective function with $\mathbf{h}^{(k)}$ and KL divergence term via Eq 13.
    Update θ via Adam method with previously calculated objective.
    Compute μ and Σ using Eq 1.
    Generate $\mathbf{H}_1, \mathbf{H}_1, \mathbf{H}_3, \cdots, \mathbf{H}_k$ via Eq 4 and Eq 6.
    Generate matrix $\mathbf{U}$ by Eq 7.
    Randomly draw 20 topic vector samples h from $N(\mu, \Sigma)$.
    Multiply $\mathbf{U}$ and $\mathbf{h}^{(0)}$ to get $\mathbf{h}^{(k)}$.
    Compute objective function with $\mathbf{h}^{(k)}$ and KL divergence term via Eq 13.
    Update γ via Adam method with previously calculated objective.
end for

5 EXPERIMENTS

In this section, we first introduce the experimental settings, evaluate the proposed model, then analyse our proposed model with extensive experiments.

Dataset: To evaluate our efforts, we select *20NewsGroups*[1] and *Reuters RCV1-v2*[2] for experiments. 20NewsGroups is a collection of newsgroup documents which consists of 11,314 training and 7,531 testing articles. And Reuters RCV1-v2 is a huge dataset that consists of Reuters newswire stories with 794,414 training and 10,000 testing cases. For data preprocessing, we remove stopwords and take the most frequent 2,000 words and 10,000 words as the vocabularies.

Baseline Methods: We compare the following methods to demonstrate the priority of the proposed method:

- **LDA** [9]: A classical topic model, which models the topic distribution as a multinomial distribution with Dirichlet prior. Here, we utilize the variational inference to implement the LDA [29].
- **CTM** [8]: It implements a log normal topic distribution. The variational inference of CTM implemented by the authors[3] is used for evaluation.
- **NVDM** [23]: The model implements a typical neural variational inference approach. The inference network of NVDM consists of a MLP network, and the generative model of NVDM reconstructs documents from reparameterized isotropic Gaussian distribution. We also use author's codes[4] for evaluation.
- **GSM** [22]: This model extends NVDM by normalizing the topic vector Gaussian sample and words distribution.
- **NVLDA** [34]: The model leverages the Laplace approximation for the LDA generative process. The topic distribution of NVLDA is an isotropic Gaussian.
- **NVCTM:** This is the proposed method in this paper.

5.1 Settings

For all of the NVI-based methods (i.e., NVDM, GSM and the proposed NVCTM), the topic vector is an average of 20 samples drawn from the latent topic distribution. For NVDM, GSM and NVCTM, we follow the authors' setting where the MLP in inference network has 256 hidden units and a hyperbolic tangent activation function. The dropout with probability of 0.8 is applied to the output of MLP network. The linear layers to parameterize μ and σ both have the same amount of hidden units with the number of topics. The training process is divided into two stages: Stage-1: optimizing the encoder's parameters while fixing the parameters of generative model; Stage-2: optimizing the generative model's parameters and keeping the inference network unchanged, which is also called wake-sleep algorithm [13]. The optimization method we use to train NVDM, GSM, NVLDA and NVCTM is Adam [17] with learning rate of $1e-5$. We also utilize early-stop [39] to stop training and export topics and topic vectors for further evaluations. The training methods of LDA and CTM are online variational inference [14] and variational inference [8]. The hyper parameters of the LDA and the CTM, which are initial values of topic distribution prior α, topic word distribution prior β and number of variational EM iterations, are determined via grid search [4] to find its optimal performance on corpus. For the proposed NVCTM, we also carry out grid search

[1] http://qwone.com/jason/20Newsgroups
[2] http://trec.nist.gov/data/reuters/reuters.html
[3] https://github.com/blei-lab/ctm-c
[4] https://github.com/ysmiao/nvdm

Table 2: Performance of perplexity of all the methods.

	20News			RCV1-v2		
T	25	50	100	25	50	100
LDA	1049	1020	1016	1123	1062	1043
CTM	1011	944	896	987	972	1021
NVDM	801	832	846	694	651	628
NVLDA	1105	1073	1034	1345	1321	1378
GSM	879	850	**843**	643	674	768
NVCTM	**715**	**749**	945	**482**	**491**	**528**

to decide the length of CTF k. For experimental environment, we run all baseline methods and proposed NVCTM on a workstation with an eight-cores Xenon E5-2630V4 CPU and GTX TITAN XP GPU.

5.2 Perplexity

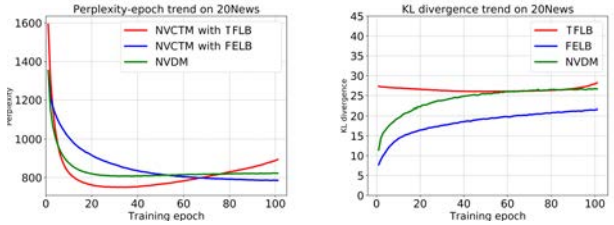
The traditional way of evaluating topic model usually refers to the perplexity computed on unseen documents. For language modeling, it refers to the inverse of geometric average per-word likelihood. The lower perplexity usually indicates the better generalization performance. The perplexity is given by:

$$Perplexity = \exp\left[-\frac{1}{D}\sum_{n}^{N_d}\frac{1}{N_d}\log p(\mathbf{X}_d)\right] \tag{14}$$

We train the baseline models and the proposed NVCTM on the training set of 20News and RCV1-v2, and compute perplexity of the corresponding models on test set. To evaluate the performance of the proposed model more comprehensively, we also conduct experiments with different number of topics, i.e., T=25, 50, 100. The performance on perplexity is displayed in Table 2. From the table, it can be observed that the NVI models (i.e., NVDM, GSM, NVCTM except for NVLDA) always have better perplexity than those of traditional models (i.e., LDA and CTM). Specifically, the proposed NVCTM approach has better performance than other baselines in most cases. For NVDM and NVCTM, which both have the same structure of generative model, we find that NVCTM significantly outperforms NVDM, which indicates that the proposed CTF approach can contribute to document modeling on perplexity. Interestingly, the perplexity performance of NVCTM is better when it comes to the smaller number of topics. This phenomenon is probably caused by the fact that the perplexity of the documents is increased as the complexity of covariance matrix of latent topics increases when the topic number grows. Besides, we also observe that for unnormalized topic vector models, such as NVDM and NVCTM, they usually have better perplexity performance than other normalized topic vector models. This effect may be due to the fact that topic vector normalization makes models difficult to optimize.

5.3 TFLB vs FELB

In the previous subsection, the results of perplexity demonstrate the effectiveness of CTF and TFLB in topic modeling. Previously,



(a) Perplexity-epoch trend on 20News- group. (b) KL divergence trend on 20Newsgroup.

Figure 4: The perplexity trend and KL divergence trend of TFLB and FELB on 20Newsgroup.

the original version of Householder flow utilizes the flow-based energy lower bound (FELB) to compute the KL divergence term. In this subsection, we will quantitatively evaluate our proposed TFLB and previous FELB in topic modeling. Accordingly, we also select perplexity as the main evaluation metric. The results are displayed in figure 4. From the perplexity trend illustrated in figures 4(a), we can see that NVCTM with TFLB can achieve relative lower perplexity than that of FELB. Moreover, compared with FELB, TFLB requires less epoch to reach the lowest perplexity, which indicates that TFLB can facilitate the training of the model. On KL divergence trend graph illustrated in figure 4(b), it shows that the KL divergence of TFLB is slightly larger than that of FELB, while the model with TFLB gets lower perplexity. This indicates that TFLB can reduce the reconstruction loss even if the KL divergence of the model slightly increases. The main reason might be that TFLB explicitly takes covariance matrix to compute the gradient and this enable it to accumulate more gradient on the inference network.

5.4 Topic Coherence

Besides the perplexity, another common method of evaluating topic models is topic coherence. Topic coherence evaluates each topic with given reference corpus and compute corresponding coherence score. Here, we adopt normalized point-wise mutual information (NPMI) topic coherence proposed by [20] to automatically evaluate the proposed model as well as baselines. The NPMI coherence score is shown to be close to human judgments [20], which is why we adopt this method for evaluation. The reference corpus of corresponding corpus is the training set of corpus. We extract top-10 and top-5 words of each topic and compute the NPMI score for each word set with the equation of

$$\text{NPMI(N)} = \sum_{j=2}^{N}\sum_{i=1}^{j-1}\frac{\log\frac{P(w_j,w_i)}{P(w_i)P(w_j)}}{-\log P(w_i,w_j)}. \tag{15}$$

Then we take the average value of two NPMI scores as the final topic coherence score. We use the author implemented version script[5] to automatically evaluate the baseline methods as well as NVCTM. To fully evaluate the topic coherence, we also compute the topic coherence score of the corresponding model on topic number T=25, 50, 100 . The topic coherence scores achieved by our

[5]https://github.com/jhlau/topic_interpretability

model and the baselines are depicted in Table 3. From the table, we find that NVCTM slightly outperforms the baselines in most cases. On 20News, NVCTM usually has better topic coherence scores than those of the baseline methods when the number of topic are 25 and 50. When the number of topic increases to 100, the topic coherence score of NVCTM decreases to 0.158, which is pretty close to the best performance of CTM. For RCV1-v2, NVCTM outperforms baselines when the number of topics are 25 and 50. Similarly, when the number of topics increases, the topic coherence of NVCTM, NVDM and GSM tends to decrease, which indicates that those models are effective on relative small amount of topics. For NVDM and proposed NVCTM, which both have the same structure of generative model, the result also indicates that proposed CTF can improve the topic coherence score of the multinomial softmax generative model.

5.5 Document Classification

To further evaluate the performance of the proposed method, document classification experiments are conducted on both 20News and RCV1-v2 datasets. To efficiently evaluate the proposed approaches and baselines, we randomly select 11 categories of RCV1-v2, namely RCV-115k, which consists of 114, 324 documents as training set and 1, 439 documents as test set evaluation. Similar to what is usually done in document classification, the topic vectors of the corresponding documents can be regarded as the low dimensional representation of the sparse document vectors. We then use these vectors to train an SVM classifier [10] with multinomial kernel function and evaluate the precision, recall and F-1 measure of the corresponding methods. The topic number is set to 50 equally of all the topic models for evaluation, in terms of the performance of document classification, which is depicted in Table 4. the results indicate that the proposed NVCTM method outperforms the baseline methods on 20News and RCV-115k. Compared with NVDM, which has the same structure generative model with NVCTM, NVCTM improves its document classification performance on both datasets, which shows the effectiveness of CTF in inference network. Among all approaches, the topic vectors of NVDM, NVLDA and NVCTM are assumed to be real-valued vectors, while the topic vectors of other methods are normalized vectors. This indicates that real-valued topic vectors can enhance the performance of document classification. To further demonstrate the topic vectors of documents and their labels, we export the topic vectors for 20News and RCV1-v2. We then visualize them by t-SNE algorithm. The visualizations of the topic vectors on 20News and RCV-115k are illustrated in figure 5. The dots with same color indicate they are the documents from same category. From figure, we find that the aggregation of vectors in RCV-115k is better than that of 20News. The better aggregation of vectors will facilitate the performance of classification.

5.6 Visualization of Topics

To further quantitatively investigate the quality of topics and topic correlation mined by NVCTM, we export the topic and the corresponding word distributions, and visualize the topics with their corresponding top 10 words on the 20NewsGroup dataset. To clearly make the visualization, we select 5 topics, which are considered to have correlations by NVCTM, and then utilize t-SNE [21] for visualization. Visualization of the topics and the topic word correlation graphs is depicted in figure 6. The points with different shapes and colors indicate different topics. The dashed circles denote the corresponding topic word distribution, and those circles with the same color are recognized to be correlated by NVCTM. We also manually annotate each topic with the corresponding label. Accordingly, they can be annotated as “comp.os.ms-windows”, “comp.graphics”, “comp.sys.ibm.pc.hardware”, “talk.politics.guns”, “talk.politics.mideast” from the ground truth labels of 20NewsGroup. From figure 6, we can see that the topics of “hardware”, “windows” and “graphics” are correlated in continuous vector space. Similarly, another two correlated topics are “mideast” and “guns”. This verifies the effectiveness of the proposed method in mining topics and correlations.

5.7 Visualization of Topical Correlations

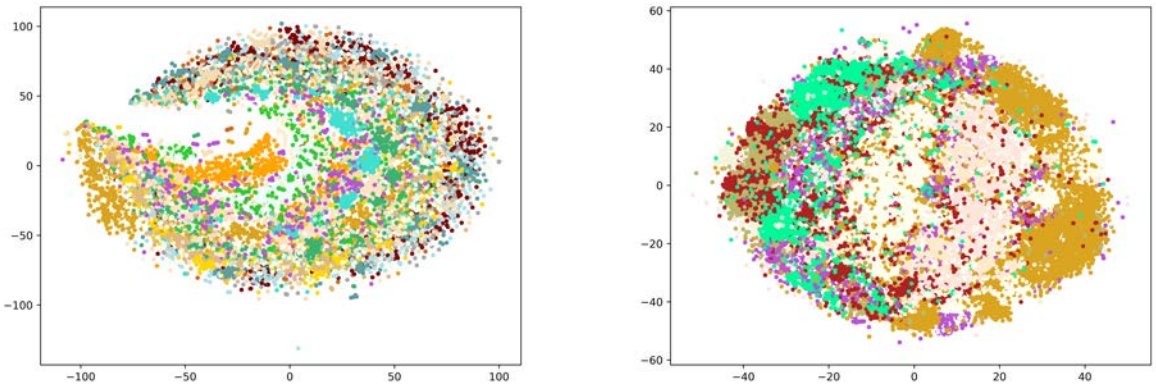
To further demonstrate effectiveness of proposed model, we export and visualize the covariance matrix of latent topic distribution on 20News and RCV1-v2. Unlike traditional mean-field variational inference or mean-field neural variational inference methods, the proposed NVCTM is capable of modeling the full covariance matrix of latent topic distributions. The main advantage of our model is that, with help of proposed CTF, NVCTM is capable of modeling the full covariance matrix rather than the diagonal one. Mathematically, the diagonal elements of covariance matrix are usually the variance values for corresponding dimension and non-diagonal elements are correlation of different two dimension. Therefore, to reach clarified visualization, we get the final correlation matrix by zeroing all the diagonal elements of covariance matrix and normalizing the matrix with rest values. The numbers of topics in NVCTM are 50 for 20News and 100 for RCV1-v2. The visualization of correlation matrix is illustrated in figure 7. From figure 7(a), we can obviously notice that the correlation matrix is non-diagonal. Besides, the most values of correlation matrix are nearly zero. This phenomenon is mainly due to the latent topic prior $N(0, \mathbf{I})$, where $\mathbf{I}$ is a diagonal unit matrix. On the other hand, each document in 20News is corresponding to a specific class of 20News, which also demonstrates the sparsity in topic correlation. The optimization of KL divergence term between CTF refined distributions and prior will make them similar to each other, which indicates that prior can also act as the sparse regularization on correlation matrix. Besides, we also notice that there are 3 topics with the higher correlation with other topics (i.e., topic No.3, 8 and 12. These 3 topics are visualized with light color which represents high correlations with other topics.). We then export these topical word distributions and find that the words are some common words, i.e, “who”, “are”, “etc”. The main reason of this effect is that these common words often exists in many documents, which makes model believe that they are correlated. On the other hand, we can still find several points with relatively large correlation coefficients. For instance, the correlation coefficients of 37 and 49, 40 and 49 are around $0.6 \sim 0.8$ which indicates the correlations exist among those topics.For figure 7(b), we can see that the number of non-zero values are larger than that of 20News. This probably results from the fact that each documents of RCV1-v2 dataset has multiple labels, which also indicates that topics in the

Table 3: Topic coherence on 20News and RCV1-v2 datasets.

	20News			RCV1-v2		
The numbers of topics	25	50	100	25	50	100
LDA	0.112	0.140	0.151	0.112	0.131	**0.143**
CTM	0.149	0.154	0.161	0.094	0.072	0.076
NVDM	0.163	0.165	0.140	0.125	0.118	0.137
NVLDA	0.167	0.145	0.110	0.121	0.136	0.107
GSM	0.141	0.132	0.111	0.101	0.076	0.062
NVCTM	**0.180**	**0.176**	0.158	**0.146**	**0.139**	0.113

Table 4: Performance of document classification on datasets of 20News and RCV-115k. The topic number of all methods is set to 50**.**

	20News			RCV-115k		
	Precision	Recall	F-1 measure	Precision	Recall	F-1 measure
LDA	0.421	0.501	0.458	0.446	0.447	0.446
CTM	0.503	0.431	0.464	0.484	0.502	0.493
NVDM	0.539	0.527	0.533	0.781	**0.798**	0.789
NVLDA	0.451	0.471	0.461	0.501	0.503	0.502
GSM	0.347	0.346	0.347	0.432	0.419	0.425
NVCTM	**0.577**	**0.564**	**0.570**	**0.818**	0.786	**0.802**



(a) The visualization of topic vectors on 20News. The length of CTF k is 5 and the number of topics is 50.

(b) The visualization of topic vectors RCV-115k.The length of CTF k is 3 and the number of topics is 100.

Figure 5: t-SNE Visualization of averaged estimated topic vector $\mathbf{h}^{(k)}$ of each document. The points with same color indicates that they have same class label in corresponding dataset.

document are likely to be correlated. Therefore, the topic correlation matrix in RCV1-v2 are reasonable to be a non-sparse matrix. To sum up, the above visualizations of correlation matrix further indicate the effectiveness of proposed approach.

6 CONCLUSION

In this paper, we first propose the Neural Variational Correlated Topic Model (NVCTM) model, which incorporates the Centralized Transformation Flow (CTF) to capture the topic correlations. CTF can enable the model to capture the correlations among topics as well as to sufficiently compute the marginal likelihood. We then

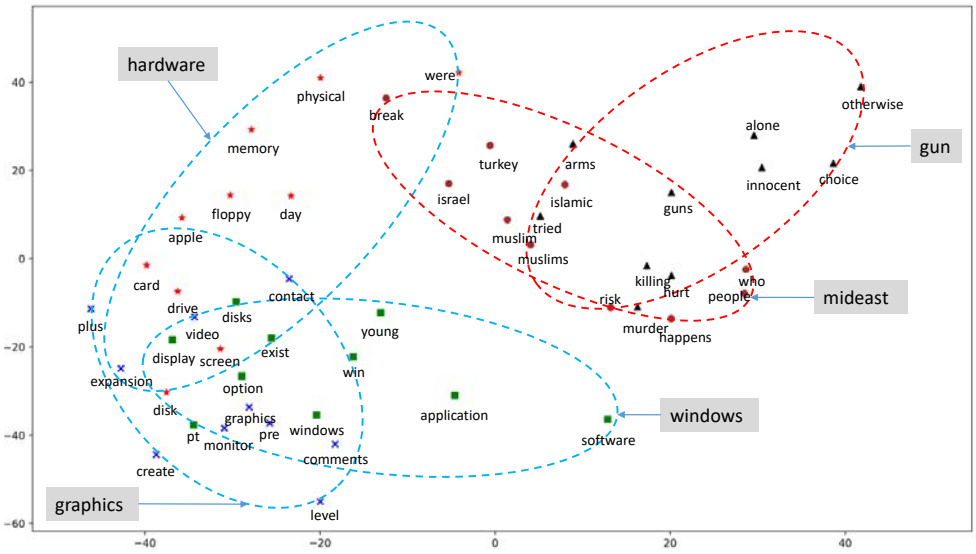
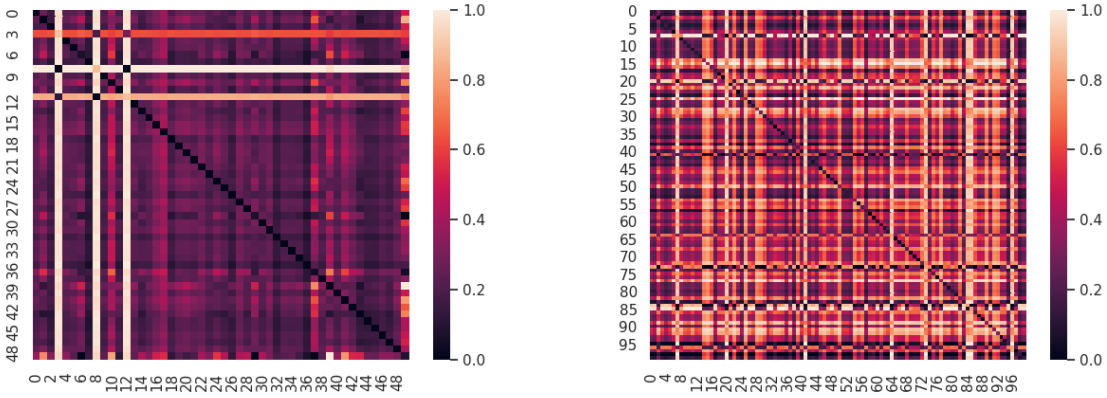


Figure 6: The t-SNE visualization of topics and topic word correlation.



(a) The visualization of Correlation matrix on 20news.

(b) The visualization of Correlation matrix on RCV1.

Figure 7: Visualization of correlation on covariance matrix. The diagonal elements are zeroed and non-diagonal elements are normalized. The deeper color indicates the smaller value.

present the Transformation Flow Lower Bound (TFLB) to regulate the objective function. It leverages the linear property of Gaussian distribution, which can regulate the model optimization and facilitates the training process of NVCTM. In order to quantitatively verify our contributions, we conduct experiments in terms of perplexity, topic coherence, and document classification tasks. The experimental results show that the proposed NVCTM approach is effective to capture topic correlations and improve the performance of topic modeling. Moreover, the visualization of the topics and their correlations qualitatively verified the effectiveness of NVCTM.

7 ACKNOWLEDGEMENT

We would like to thank Yuxiang Zhou and Yishu Miao for their insightful comments and suggestions. We also very appreciate the comments from anonymous reviewers which help further improve our work. This work is supported by National Key R&D Plan(No.2016QY03D0602), "Key technologies, system and application of Cyberspace Big Search", Major project of Zhejiang Lab (No.2019DH0ZX01), National Natural Science Foundation of China (Grant No.61602036).

REFERENCES

[1] Amrudin Agovic and Arindam Banerjee. 2012. Gaussian Process Topic Models. *CoRR* abs/1203.3462 (2012). http://arxiv.org/abs/1203.3462

[2] Loulwah AlSumait, Daniel Barbará, and Carlotta Domeniconi. 2008. On-line lda: Adaptive topic models for mining text streams with applications to topic detection and tracking. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 3–12.

[3] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. 2003. An introduction to MCMC for machine learning. *Machine learning* 50, 1-2 (2003), 5–43.

[4] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* 13, Feb (2012), 281–305.

[5] Christian H Bischof and Xiaobai Sun. 1994. On orthogonal block elimination. *Preprint MCS-P450-0794, Mathematics and Computer Science Division, Argonne National Laboratory* (1994).

[6] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. 2017. Variational inference: A review for statisticians. *J. Amer. Statist. Assoc.* 112, 518 (2017), 859–877.

[7] David M. Blei and John D. Lafferty. 2005. Correlated Topic Models. In *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*. 147–154. http://papers.nips.cc/paper/2906-correlated-topic-models

[8] David M. Blei and John D. Lafferty. 2005. Correlated Topic Models. (2005), 147–154. http://papers.nips.cc/paper/2906-correlated-topic-models

[9] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2001. Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani (Eds.). MIT Press, 601–608. http://papers.nips.cc/paper/2070-latent-dirichlet-allocation

[10] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)* 2, 3 (2011), 27.

[11] Rajarshi Das, Manzil Zaheer, and Chris Dyer. 2015. Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 795–804.

[12] Philipp Hennig, David Stern, Ralf Herbrich, and Thore Graepel. 2012. Kernel topic models. In *Artificial Intelligence and Statistics*. 511–519.

[13] Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. 1995. The" wake-sleep" algorithm for unsupervised neural networks. *Science* 268, 5214 (1995), 1158–1161.

[14] Matthew Hoffman, Francis R Bach, and David M Blei. 2010. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*. 856–864.

[15] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. 2013. Stochastic variational inference. *The Journal of Machine Learning Research* 14, 1 (2013), 1303–1347.

[16] Liangjie Hong and Brian D Davison. 2010. Empirical study of topic modeling in twitter. In *Proceedings of the first workshop on social media analytics*. ACM, 80–88.

[17] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Diederik P. Kingma, Tim Salimans, and Max Welling. 2016. Improving Variational Inference with Inverse Autoregressive Flow. *CoRR* abs/1606.04934 (2016).

[19] Diederik P. Kingma and Max Welling. 2013. Auto-Encoding Variational Bayes. *CoRR* abs/1312.6114 (2013). http://arxiv.org/abs/1312.6114

[20] Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. 530–539.

[21] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[22] Yishu Miao, Edward Grefenstette, and Phil Blunsom. 2017. Discovering Discrete Latent Topics with Neural Variational Inference. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017 (Proceedings of Machine Learning Research)*, Doina Precup and Yee Whye Teh (Eds.), Vol. 70. PMLR, 2410–2419. http://proceedings.mlr.press/v70/miao17a.html

[23] Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural Variational Inference for Text Processing. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings)*, Maria-Florina Balcan and Kilian Q. Weinberger (Eds.), Vol. 48. JMLR.org, 1727–1736. http://jmlr.org/proceedings/papers/v48/miao16.html

[24] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[25] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[26] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. 2008. The matrix cookbook. *Technical University of Denmark* 7, 15 (2008), 510.

[27] Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. 2008. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 569–577.

[28] Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*. Association for Computational Linguistics, 248–256.

[29] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[30] Danilo Jimenez Rezende and Shakir Mohamed. 2015. Variational Inference with Normalizing Flows. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015 (JMLR Workshop and Conference Proceedings)*, Francis R. Bach and David M. Blei (Eds.), Vol. 37. JMLR.org, 1530–1538. http://jmlr.org/proceedings/papers/v37/rezende15.html

[31] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014 (JMLR Workshop and Conference Proceedings)*, Vol. 32. JMLR.org, 1278–1286. http://jmlr.org/proceedings/papers/v32/rezende14.html

[32] Tim Salimans, Diederik Kingma, and Max Welling. 2015. Markov chain monte carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*. 1218–1226.

[33] Tian Shi, Kyeongpil Kang, Jaegul Choo, and Chandan K Reddy. 2018. Short-Text Topic Modeling via Non-negative Matrix Factorization Enriched with Local Word-Context Correlations. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1105–1114.

[34] Akash Srivastava and Charles Sutton. 2017. Autoencoding Variational Inference For Topic Models. *arXiv preprint arXiv:1703.01488* (2017).

[35] Mark Steyvers, Padhraic Smyth, Michal Rosen-Zvi, and Thomas Griffiths. 2004. Probabilistic author-topic models for information discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 306–315.

[36] Xiaobai Sun and Christian Bischof. 1995. A basis-kernel representation of orthogonal matrices. *SIAM journal on matrix analysis and applications* 16, 4 (1995), 1184–1196.

[37] Jakub M. Tomczak and Max Welling. 2016. Improving Variational Auto-Encoders using Householder Flow. *CoRR* abs/1611.09630 (2016). http://arxiv.org/abs/1611.09630

[38] Guangxu Xun, Yaliang Li, Wayne Xin Zhao, Jing Gao, and Aidong Zhang. 2017. A Correlated Topic Model Using Word Embeddings. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*. 4207–4213. https://doi.org/10.24963/ijcai.2017/588

[39] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. 2007. On early stopping in gradient descent learning. *Constructive Approximation* 26, 2 (2007), 289–315.

A Hybrid BitFunnel and Partitioned Elias-Fano Inverted Index

Xinyu Liu
liuxy@nbjl.nankai.edu.cn
College of CS, TJ Key Lab of NDST
Nankai University
Tianjin, China

Zhaohua Zhang
zhangzhaohua@nbjl.nankai.edu.cn
College of CS, TJ Key Lab of NDST
Nankai University
Tianjin, China

Rebecca J. Stones
becky@nbjl.nankai.edu.cn
College of CS, TJ Key Lab of NDST
Nankai University
Tianjin, China

Yusen Li
liyusen@nbjl.nankai.edu.cn
College of CS, TJ Key Lab of NDST
Nankai University
Tianjin, China

Gang Wang*
wgzwp@nbjl.nankai.edu.cn
College of CS, TJ Key Lab of NDST
Nankai University
Tianjin, China

Xiaoguang Liu*
liuxg@nbjl.nankai.edu.cn
College of CS, TJ Key Lab of NDST
Nankai University
Tianjin, China

ABSTRACT

Search engines encounter a time vs. space trade-off: search responsiveness (i.e., a short query response time) comes at the cost of increased index storage. We propose a hybrid method which uses both (a) the recently published mapping-matrix-style index BitFunnel (BF) for search efficiency, and (b) the state-of-the-art Partitioned Elias-Fano (PEF) inverted-index compression method. We use this proposed hybrid method to minimize time while satisfying a fixed space constraint, and to minimize space while satisfying a fixed time constraint. Each document is stored using either BF or PEF, and we use a local search strategy to find an approximately optimal BF-PEF partition. Since performing full experiments on each candidate BF-PEF partition is impractically slow, we use a regression model to predict the time and space costs resulting from candidate partitions (space accuracy 97.6%; time accuracy 95.2%). Compared with a hybrid mathematical index (Ottaviano et al., 2015), the time cost is reduced by up to 47% without significantly exceeding its size. Compared with three mathematical encoding methods, the hybrid BF-PEF index allows performing list intersection between around 16% to 76% faster (without significantly increasing the index size). Compared with BF, the index size is reduced by 45% while maintaining an intersection time comparable to that of BF.

CCS CONCEPTS

• **Information systems → Information retrieval query processing**; **Search engine indexing**; **Search index compression**.

KEYWORDS

Partitioned Elias-Fano; BitFunnel; hybrid index; compression; query processing

ACM Reference Format:
Xinyu Liu, Zhaohua Zhang, Rebecca J. Stones, Yusen Li, Gang Wang, and Xiaoguang Liu. 2019. A Hybrid BitFunnel and Partitioned Elias-Fano Inverted Index. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313553

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313553

1 INTRODUCTION

Search engines face a trade-off between space and time [1, 7, 24]: if we compress the underlying index, we reduce the system's query responsiveness. The most common search-engine index structure is the *inverted index* where for each term, we store a *posting list* containing the docIDs of documents containing that term [6, 18].

The Partitioned Elias-Fano (PEF) [23] inverted-index compression method is a state-of-the-art method, which uses a shortest-path algorithm [8] to dynamically partition the posting list into variable-length blocks with clustered docIDs within each block; Elias-Fano encoding [12, 13] is used blockwise for random access.

Microsoft's Bing search engine uses a newly described index structure BitFunnel (BF) [15], which is a bitmap-like data structure: terms map to one or a few rows, while columns correspond to individual documents. If a term appears in the document, the corresponding bit positions are set to 1. In this way, intersection is performed using simple bitwise AND operations. However, the BitFunnel is much larger than the corresponding compressed inverted index.

In this paper, we combine the advantages of BF (intersection efficiency) and PEF (space efficiency), and propose a hybrid index. We partition the document corpus into two parts, with one part allocated to BF and the other allocated to PEF. The choice of partition affects the time-space trade-off, and can be chosen to suit the search engine's requirements. We envisage choosing this partition so that either a time threshold or a size threshold is not exceeded.

The major contributions of this paper are as follows:

(1) We use the local search method to find an approximate optimal partition of the document corpus while satisfying a pre-specified threshold condition.
(2) For a large dataset, it is impractical to repeatedly estimate the index's size and intersection time experimentally, so we develop and utilize a machine learning approach to estimate the index size and intersection time of the hybrid index with a given partition.
(3) We implement and conduct experiments on the proposed hybrid index. More specifically:

- we assess three machine learning methods (GBRT [14], SVR [11], and RF [5]) for predicting the space and time costs of the resulting BF-PEF hybrid index (given an assignment of the documents to BF and PEF), and ultimately select support vector regression (SVR) which consistently outperforms the other two methods in terms of error rate;
- we investigate the convergence of the local search process on the size and average intersection time of proposed hybrid index under various thresholds; and
- we compare the space-time trade-off of the proposed hybrid index against several mathematical encoders, along with a hybrid mathematical index.

This paper is structured as follows. In Section 2 we describe pilot experiments which motivate the design decisions throughout the paper. Section 3 gives a minimalistic background to the relevant material. Section 4 reviews alternative hybrid index methods, and their relevance to the proposed hybrid BF-PEF method. A detailed description of the method used to generate an approximately optimal BF-PEF partition is given in Section 5. Experimental results are given in Section 6. We conclude with some future research ideas in Section 7.

2 PILOT EXPERIMENTS

In this section, we describe some preliminary experiments for some seemingly natural approaches to choosing a *BF-PEF partition*, i.e., a partition of the documents into two parts, one stored using BF and the other stored using PEF. These results motivate the subsequent use of machine learning and local search in optimizing the partition. These pilot experiments use the GOV2 document corpus and MillionQuery set, described in Section 6.1. We also describe some alternative methods.

With N documents there are 2^N possible BF-PEF partitions (and N is large in most search engines, e.g., billions of documents). Exhaustive inspection of all 2^N partitions is unrealistic, and thus the goal is to find an approximately optimal partition. Further, it is practically far too time-consuming to generate a hybrid index and perform experiments on it (like in Table 1) for each partition we wish to test. Moreover, there is no single "best" partition, rather we seek a partition which is optimized according to criteria determined by the search engine. For example, if the search engine has X TB of memory available for index, we cannot violate this hard restriction. Similarly, a search-engine operator may have their own constraints regarding the system's responsiveness.

To summarize, we have three major design constraints:

(1) we cannot find an optimal BF-PEF partition, but only an approximation,
(2) it is impractical to perform fully fledged experiments for a large number of candidate BF-PEF partitions, and
(3) we anticipate hard restrictions on the index size and intersection time imposed e.g. by hardware or the operator.

An alternative straightforward approach is to randomly assign documents to either BF or PEF; we randomly choose a proportion, and randomly assign that proportion of documents to BF and the remainder are assigned to PEF, and we tabulate experimental results of partition in Table 1. While this randomized hybrid method combines the advantages of BF and PEF, the trade-off is unimpressive: we incur around 75% of the large space cost of BF while being almost 1.6 times as slow as BF. Thus, we need to use a more intelligent way of partitioning the documents.

Table 1: The index size (space) and average intersection time (time) on BF, PEF, and a BF-PEF hybrid method with documents randomly assigned to BF and PEF.

	BF	BF-PEF (rand.)	PEF
space (GB)	13.40	9.76	3.25
time (ms)	1.45	2.29	3.19

We also consider incorporating a bitmap compression method, such as EWAH [17]; we tested this approach and observed a minor space reduction of less than 5% when compressing the BitFunnel index of GOV2. We attribute this behavior to BitFunnel's high density of 1s (usually 0.1 to 0.2), which results in short all-0 and all-1 subsequences, leading to poor compression.

Recently, Liu et al. [19] proposed a dictionary-based compression method for BitFunnel with additional false positives. On the GOV2 dataset, we found that the compression rate of this method is around 30%, but its intersection time increases by about 50%, and its false positive rate is doubled which is consistent with the results in that paper. We thus abandon BF compression approaches.

A possible way to compress an inverted index is via mathematical encoding [2, 3, 25, 30]. However, PEF achieves a higher compression rate while simultaneously allowing efficient list intersection.

We investigate how the BF-PEF partition is influenced by some *basic features*:

- *URL*, the lexicographic order of the document URLs;
- *DL*, the length of the documents without duplicate terms (i.e., the number of unique terms in document);
- *DF*, the mean document frequency of terms in the document;
- *dispersivity*, which measures the lack of clustering of a document in the inverted index, as per f_8 in Section 5.1.1; and
- *TF-IDF*, a mean TF-IDF calculated as per f_{11} in Section 5.1.1.

We perform some toy experiments involving these basic factors from which we glean insights into how we might design a fully fledged hybrid BF-PEF index.

For each basic feature f, we create the set S_f of documents in GOV2 which have the highest 10% of that f's possible values. We create an artificial document corpus $\mathcal{D}_f$ for each f by randomly selecting 100 000 documents in S_f. For each $\mathcal{D}_f$, we generate the BF and PEF indexes, and compare their size and average intersection time. In order to compare features with each other, we measure the size according to

$$s_f := \frac{\text{size of index generated from } \mathcal{D}_f}{\text{total number of docIDs in posting lists in } \mathcal{D}_f}$$

and the time according to

$$t_f := \frac{\text{average intersection time on index generated from } \mathcal{D}_f}{\text{total number of docIDs in posting lists in } \mathcal{D}_f}.$$

We also perform the same computations for the lowest 10% of the possible f values, and distinguish between the two cases as "high" and "low". We plot the results in Figure 1.

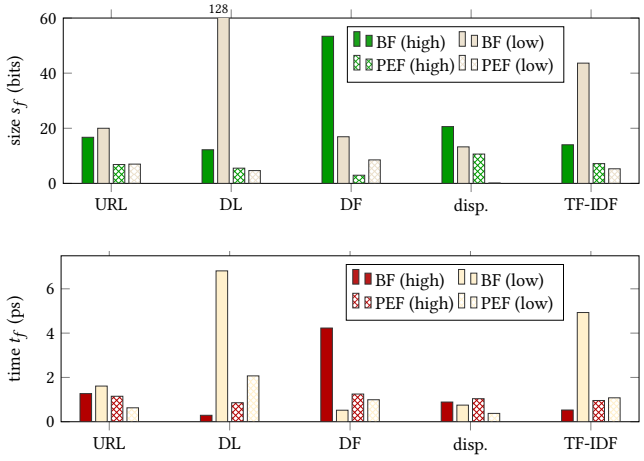


Figure 1: The size s_f and time t_f of the BF index and PEF index constructed for the toy document corpus $\mathcal{D}_f$ for the basic feature $f \in \{\textbf{URL}, \textbf{DL}, \textbf{DF}, \textbf{disp.}, \textbf{TF-IDF}\}$.

To investigate the sensitivity to the random document corpus $\mathcal{D}_f$, we also repeat this computation 5 times, and observe negligible difference between trials.

We make the following observations:

Observation 1. *Multiple basic features significantly influence the size and intersection time of the BF and PEF indexes.*

For example, for BF we observe that a greater DF leads to a greater s_{DF}, which we attribute posting list density: a dense posting list maps to unshared rows in the BitFunnel. We also see that for PEF, a low dispersivity results in a small index size and intersection time, indicating that docID clustering benefits PEF.

Observation 2. *For either the BF or PEF component of a hybrid BF-PEF index, a high (or low) value of a single feature may be simultaneously advantageous for size (resp. time) while disadvantageous for time (resp. size).*

For example, for PEF a greater DF leads to a smaller index, but a greater intersection time. We expect this is because a high DF implies that most of the PEF-compressed documents contain high-frequency terms in the corpus, resulting in longer posting lists being compressed (and decompressed during list intersection).

Observation 3. *A high (or low) value of a single feature may be simultaneously advantageous for size and time for both the BF and PEF components of a hybrid BF-PEF index.*

For example, a lower dispersivity results in a smaller $s_{\text{disp.}}$ and $t_{\text{disp.}}$ for both BF and PEF. Since PEF is designed to utilize clustering, it is apparent why a low dispersivity results in reduced $s_{\text{disp.}}$ and $t_{\text{disp.}}$ for the PEF index. For BF, a lower dispersivity also results in a smaller $s_{\text{disp.}}$ and $t_{\text{disp.}}$, although it is not apparent why.

Some of these observations are predictable, while others are not; and some of these observations result in a quandary, where there is not an apparent best way to assign documents to BF or PEF. In the subsequent experiments (Section 6; see in particular Section 6.5 for feature-based experiments), we incorporate additional features, and anticipate varying and unpredictable effects on BF-PEF indexes. Thus, we expect it is not straightforward to find a near-optimal BF-PEF partition using simple threshold methods.

This unpredictable feature behavior motivates the use of a machine learning regression approach, which predicts the index size and intersection time of hybrid BF-PEF indexes based on multiple document features. In particular, this is the motivation behind the feature vectors in Section 5.1.2.

3 BACKGROUND

In search engines, the most common index structure is the *inverted index* [6, 18], a database where each *term* t has a corresponding *posting list* ℓ_t, which stores an ordered list of the docIDs that the term t appears in, namely

$$\ell[t] = \langle \text{docID}_1, \text{docID}_2, \ldots, \text{docID}_n \rangle.$$

For conjunctive queries, the relevant documents for a query $\{t_i\}$ are given by the *list intersection* $\cap_i \ell[t_i]$, which is computed after a query is made (i.e., "online") and thus needs to be performed efficiently.

Ordinarily we use some form of compression to reduce the size of the inverted index. To avoid decompressing entire posting lists during query processing, we typically divide the posting lists into blocks which are individually compressed and decompressed (to facilitate random access).

Differing from the traditional fixed-length block encoding, Partitioned Elias-Fano (PEF) [23] encoding uses dynamic block sizes in order to make use of clustering in the inverted index. The authors of [23] also described a method for obtaining an approximate optimal block partition, where each block uses Elias-Fano encoding [12, 13].

BitFunnel [15] is a bitmap-like mapping matrix which is based on a Bloom filter [4, 28]. In a BitFunnel, each term is mapped to one or a few rows, with each column representing a document. If a term appears in a document, the corresponding entry (or entries) is set to "1" (otherwise "0"). For conjunctive queries, to obtain the documents containing all the query terms, we perform an AND operation on the rows that the query terms map to. A "1" in the result usually indicates a document which contains all the query terms, although the Bloom filter admits the possibility of false positives.

There are some properties of BitFunnel which affect its operation, but only indirectly affect the results in this paper. Specifically: (a) given a density, BitFunnel generates an index with a low false positive rate while approximately achieving that density; (b) BitFunnel usually has a false positive rate of less than 5%. False positives are not hugely problematic in practice, since e.g. they might be automatically eliminated during top-K ranking; (c) documents are divided into shards according to their length, and BitFunnel use "frequency-conscious" signatures (i.e., multiplexing) to reduce the storage space; and (d) to speed up the intersection process, rows are ascribed a rank, whereby high-ranked rows are shorter and are expanded (self-concatenated) before being intersected with low-ranked rows. In experiments in Section 6, we use BitFunnel's default settings, including setting the density to 0.15 and the number of shards to 11.

Compared with the list intersection approach, a BitFunnel index is much larger (e.g. 4.1 times larger in Table 1) but also much faster to query (2.2 times faster in Table 1). This results in an "all your

eggs (documents) in one basket (index)" trade-off, and this paper is about finding an in-between, hybrid approach.

Motivated by both hardware limits and operator-set limits, we consider two setups:

(1) minimizing the average intersection time subject to a fixed maximum index size, and
(2) minimizing the index size subject to a fixed maximum average intersection time.

4 RELATED WORK

To reduce the storage overhead, prior research proposed a variety of mathematical encoding methods for inverted indexes, such as PForDelta (PFD) [30], binary interpolative coding [21], Vbyte [25] (Varint-G8IU [27] exploiting SIMD operations), Simple9 and Simple16 [2, 3], and so on. We use OptPFD [29] (an improved version of PFD), binary interpolative coding, and Varint-G8IU as baselines in Section 6.4.

These mathematical encoding methods minimize the number of bits used to represent the docIDs, thereby achieving compression. However, the constant-size block division of the posting list is incompatible with clustering. To illustrate, if we split the toy posting list

$$\langle 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 21, 22, 23, 24, 30\rangle$$

into length-4 blocks of docIDs

$$\langle 1, 2, 3, 4\rangle, \langle 5, 6, 7, 8\rangle, \langle 9, 10, 20, 21\rangle, \langle 22, 23, 24, 30\rangle$$

we break apart the clusters of docIDs. To utilize this clustering among docIDs, and EF encoding for random access, in the proposed approach, we use PEF to compress one part of the document corpus. For the documents not compressed using PEF, we use BitFunnel. This proposed hybrid method deviates from the usual approach of using a single index structure or compression method for the whole corpus.

We also remark that the documents assigned to BF do not overlap with the documents assigned to PEF, thus the BF results and PEF results do not require a subsequent time-consuming "merging" operation.

We take advantage of the bitwise operation of BF for time efficiency and the dynamic block partitioning of PEF for space efficiency. PEF's compression is comparable to the best mathematical encoding method we consider in this work, while also significantly outperforming them in terms of intersection time (see Section 6.4).

Some hybrid indexes have been previously proposed, combining different types of indexes to obtain a smaller index or a shorter query time. Some methods partition the corpus by term [9, 20] where different type of index (or server) is selected for each posting list. Unlike their approaches, the proposed method partitions the document corpus by document. In this way, the partitioning may change the distribution of the inverted index (e.g., the length and the clustering of posting lists). We expect different document subsets adapt to different index type.

Ottaviano et al. [22] combined three mathematical encoding methods (PFD [29], binary interpolative coding [21], and Varint-G8IU [27]) to generate a hybrid index. Each block is assigned the compression method which minimizes the average query time through the greedy algorithm described in [26] while subject to a spatial threshold. Given a query log, Ottaviano et al. randomly sample a few blocks to generate the training set, and use a linear model to predict the decoding time of each block during query processing using this training set.

Unlike the proposed method, the method by Ottaviano et al. has a fixed block size, and does not take advantage of dynamic partitioning for inverted indexes like PEF. Moreover, it does not incorporate a bitmap-like index structure. We use [22] as a baseline (hybrid-PIV in Section 6.4).

Kane and Tompa [16] introduced "semi-bitvectors": the documents are grouped and reordered to increase the clustering in posting lists. They divide each posting list by a cut point into two parts: the denser front part for the bitvector and the remainder of the inverted index is compressed by PFD [30]. The cut point for each posting list is based on the bitvector density threshold F and the number of groups G: two user-defined parameters. In principle, we could apply a threshold on the intersection time or index size to the semi-bitvector method by repeatedly adjusting the parameters F and G and rebuild the index multiple times until the specified threshold is satisfied. However, this approach neither optimizes size nor time. In the proposed method, we use an iterative approach to find an approximately optimal BF-PEF partition; we minimize the intersection time while staying within a size threshold (or vise versa with "intersection time" and "size" switched).

A major structural difference is that the proposed method splits the document corpus into two parts, whereas Kane and Tompa's approach individually splits the posting lists. With semi-bitvectors, it is necessary to perform list intersection across two disparate index structures, which hinders parallelization and the application of Kane and Tompa's approach to e.g. distributed systems. Instead, partitioning the document corpus (as in the proposed approach) avoids this problem, since each partition can be intersected independently.

5 HYBRID BF-PEF PARTITION SELECTION

The proposed approach has three main components depicted in Figure 2:

- We use machine learning to develop regression models which compare BF-PEF partitions. We compare three different models, but find support vector regression to be the most suitable.
- Once a regression model is generated, we use a local search approach where we perturb the "current" partition (starting with an initialized partition), and replace it by the perturbed partition if the relevant regression model considers the perturbed partition better.
- We consider two ways of perturbing the current partition: one random, and one which prioritizes either PEF over BF or BF over PEF.

5.1 Prediction

5.1.1 Feature selection. We predict the size and intersection time of a hybrid index (arising from a BF-PEF partition) using machine learning. For an arbitrary document d, we have the following document features:

- the features f_1 and f_3 are respectively URL and DL, as used in the pilot experiments;
- the number f_2 of (not necessarily distinct) terms in d;

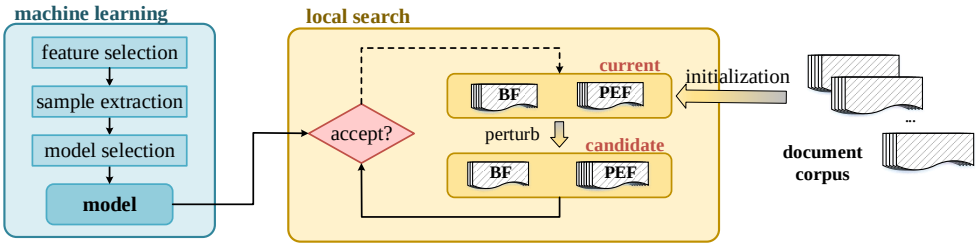


Figure 2: Flow chart for generating an approximately optimal BF-PEF partition. Machine learning is used to give a prediction model. The local search accepts a perturbed BF-PEF partition if it is predicted to outperform the current BF-PEF partition.

- the shardID f_4 of the document in the BF index, which varies in BitFunnel according to the number of unique terms in a document;
- the maximum f_5, minimum f_6, and mean f_7 of the document frequency among the terms t in the document (i.e., the length of the posting list for t);
- we define the *dispersivity* f_8 as

$$\operatorname*{avg}_{t\in d}\sqrt{\frac{1}{b-a+1}\sum_{j\in\{a,\ldots,b\}\setminus\{i\}}\frac{(\mathrm{docID}_j-\mathrm{docID}_i)^2}{\log_{10}(|j-i|+1)}}$$

  where i is the index for document d in the posting list ℓ_t, and $a=\max(1,i-32)$ and $b=\min(|\ell[t]|,i+32)$, as used in the pilot experiments;
- the maximum f_9, minimum f_{10}, and mean f_{11} of the TF-IDF score of the document d, restricted to the terms $t\in S$, where S is the smallest set of highest-frequency terms such that sum of the frequencies of terms in S accounts for 50% of the total frequency; this approach is inspired by [16] because high frequency terms usually play a more important role on index size and query time; and
- query count f_{12}, i.e., the number of query terms appearing in d.

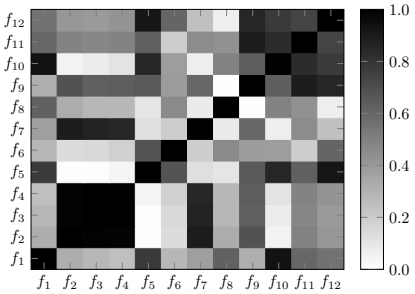


Figure 3: Feature correlation heat map.

We exclude some highly correlated features based on experiments much like the pilot experiments (Section 2). For each feature, we construct a vector of 8 values as in Figure 1. We normalize the feature values to $[0,1]$, then calculate the correlation [10] between each pair of features; we plot the results in Figure 3. We exclude the features with correlation exceeding 98%: document length f_2, document unique length f_3, and shardID f_4 have pairwise correlation exceeding 98%, so we exclude f_2 and f_4. While other pairs of features exhibit varying levels of correlation, there is virtually no meaningful distinction between f_2, f_3, and f_4, so we exclude two of them so as to not add undue weight. The remaining features, i.e., those in $\{f_1,f_3\}\cup\{f_5,f_6,\ldots,f_{12}\}$, are denoted $\hat{f}_1$ through $\hat{f}_{10}$.

5.1.2 Sample Extraction and Model Selection. In this section, we introduce how we define samples which we use in machine learning (to train the two prediction models). Generating a hybrid index based on a whole corpus is time consuming, so we construct a practical set of samples to train the model.

For the document corpus $\mathcal{D}$, for each $i\in\{1,\ldots,10\}$, we define $\mathcal{D}_i^{(\mathrm{low})}$ and $\mathcal{D}_i^{(\mathrm{high})}$ as the set of documents respectively with the least and greatest $\hat{f}_i$ values, both comprising approximately 25% of all documents. We define $\mathcal{D}_i^{(\mathrm{medium})}=\mathcal{D}\setminus(\mathcal{D}_i^{(\mathrm{low})}\cup\mathcal{D}_i^{(\mathrm{high})})$, which comprises about 50% of all documents. For the purpose of this paper (i.e., for use with the GOV2 dataset), we define the set of pairs

$$\mathcal{N}=\{(5,7.5),(7.5,10),(10,25),(25,50),(50,100),\\(100,150),(150,200),(200,250)\}\times 10^5,$$

so e.g. $(500\,000,750\,000)\in\mathcal{N}$, and so on.

Given a pair $(a,b)\in\mathcal{N}$ we randomly select a number $t\in\{0,\ldots,10\}$ and a random t-subset $F\subseteq\{1,\ldots,10\}$. Each $i\in F$ indexes feature $\hat{f}_i$, and we randomly select $\mathcal{D}_i\in\{\mathcal{D}_i^{(\mathrm{high})},\mathcal{D}_i^{(\mathrm{medium})},\mathcal{D}_i^{(\mathrm{low})}\}$, and for each document $d\in\mathcal{D}$, we assign it the score

$$\mathrm{score}(d):=|\{i\in F\colon d\in\mathcal{D}_i\}|\,. \tag{1}$$

We randomly choose $n\in\{a,\ldots,b-1\}$ and form a document corpus of size n consisting of the documents $d\in\mathcal{D}$ with the greatest score. If $t=0$, we randomly select any n documents from $\mathcal{D}$. Informally, we randomly choose a "high", "medium", or "low" feature condition for each of a random selection of features, and we choose n documents which satisfy the most conditions.

To reduce the time of sample extraction, we randomly choose pairs (a,b) from $\mathcal{N}$ so that lower-valued pairs have a greater probability of being selected. This gives a family $\mathcal{F}$ of document corpuses. We split $\mathcal{F}$ into training and validation sets.

We randomly choose two document corpuses $\mathcal{D}^{(\mathrm{BF})}$ and $\mathcal{D}^{(\mathrm{PEF})}$ from $\mathcal{F}$ to generate a partition (which might intersect, but we ignore this as we expect it has a negligible effect). We compute the average feature values for both of these corpuses, giving $\hat{f}_1^{(\mathrm{BF})}$ through

$\hat{f}_{10}^{(\mathrm{BF})}$ (from $\mathcal{D}^{(\mathrm{BF})}$) and $\hat{f}_1^{(\mathrm{PEF})}$ through $\hat{f}_{10}^{(\mathrm{PEF})}$ (from $\mathcal{D}^{(\mathrm{PEF})}$). We define the two sample vectors used for training: the *size vector*

$$\vec{\sigma}_{\mathrm{size}} = \Big\langle \hat{f}_1^{(\mathrm{BF})}, \ldots, \hat{f}_{10}^{(\mathrm{BF})}, |\mathcal{D}^{(\mathrm{BF})}|, \hat{f}_1^{(\mathrm{PEF})}, \ldots, \hat{f}_{10}^{(\mathrm{PEF})}, |\mathcal{D}^{(\mathrm{PEF})}|, s_{\mathrm{BF}} + s_{\mathrm{PEF}} \Big\rangle$$

and the *time vector*

$$\vec{\sigma}_{\mathrm{time}} = \Big\langle \hat{f}_1^{(\mathrm{BF})}, \ldots, \hat{f}_{10}^{(\mathrm{BF})}, |\mathcal{D}^{(\mathrm{BF})}|, \hat{f}_1^{(\mathrm{PEF})}, \ldots, \hat{f}_{10}^{(\mathrm{PEF})}, |\mathcal{D}^{(\mathrm{PEF})}|, t_{\mathrm{BF}} + t_{\mathrm{PEF}} \Big\rangle$$

where s_{BF} (resp. t_{BF}) is the size (resp. average intersection time) of the index formed when using BitFunnel on the corpus $\mathcal{D}^{(\mathrm{BF})}$ and s_{PEF} (resp. t_{PEF}) is the size (resp. average intersection time) of the index formed when using PEF on the corpus $\mathcal{D}^{(\mathrm{PEF})}$. Doing this repeatedly gives a set M_{size} of size vectors and a set M_{time} of time vectors.

The purpose of this elaborate generation process is to include a diverse selection of vectors in the training and validation sets, thereby helping the machine learning process to identify the (size and time) impact of the features.

For model selection, we use three common regression models: the gradient-boosted regression tree (GBRT) model [14], a support vector regression (SVR) model [11], and a random forest model [5] for training. Further, we use a validation set to tune the parameters. An experimental comparison of these models is given in Section 6.2.

Once the size and time regression models are generated, we apply these two models to the entire document corpus which assist local search to judge the "candidate" partition and the "current" partition. Noted that all the samples in test sets we use in experiments (see Section 6.2) are real BF-PEF partitions (of the whole document corpus, rather than artificial sets as for training and validation samples).

5.2 Partition

We use a local search approach to find an approximately optimal partition. We perturb the current partition, and accept the perturbed partition if the chosen model (derived via machine learning, as in Section 5.1) predicts the perturbed partition performs better (either in terms of size or time, depending on which setup is chosen). This is repeated some number of times; around 200 iterations seems sufficient with the present paper's experimental setup (see Section 6.3). The details are given in Algorithm 1.

There are two different methods for selecting the documents P in Algorithm 1 (Lines 3, 5, and 7):

(1) *Random selection* (used with probability 20%). Some subset F of the features is randomly selected. We select a random $n \in \{250\,000, \ldots, 1\,250\,000\}$ (n is the number of documents in P, account for 1% to 5% of the whole corpus), and choose the n documents with the highest score, as defined by eq. (1).

(2) *Priority selection* (used with probability 80%). This selection method is the same as random selection, except in calculating eq. (1), where we choose a collection of documents that are more friendly to the destination index. Specifically, we choose $\mathcal{D}_i$ as either "high" or "low" respectively whenever a greater or lesser value of the corresponding feature $\hat{f}_i$ benefits BF (resp. PEF) in terms of both size and time, and we refer to these features as *BF-friendly* features (resp. *PEF-friendly* features). When prioritizing BF, we randomly choose between "high" or "low" for non-BF-friendly features, and likewise for non-PEF-friendly features when prioritizing PEF.

Algorithm 1 Local search method

Input: Document corpus, the number of iterations T
Output: An approximately optimal partition

1: Randomly partition the document corpus into two parts $\mathcal{P} = \{\mathcal{P}^{(\mathrm{BF})}, \mathcal{P}^{(\mathrm{PEF})}\}$
2: **for** i from 1 to T **do**
3: Choose a subset $P \subseteq \mathcal{P}^{(\mathrm{BF})}$ ▷ two different methods
4: Replace the current partition $\mathcal{P}$ with the perturbed partition $\{\mathcal{P}^{(\mathrm{BF})} \setminus P, \mathcal{P}^{(\mathrm{PEF})} \cup P\}$ if the perturbed partition is predicted to perform better
5: Choose a subset $P \subseteq \mathcal{P}^{(\mathrm{PEF})}$ ▷ two different methods
6: Replace the current partition $\mathcal{P}$ with the perturbed partition $\{\mathcal{P}^{(\mathrm{BF})} \cup P, \mathcal{P}^{(\mathrm{PEF})} \setminus P\}$ if the perturbed partition is predicted to perform better
7: Choose a subset $P_1 \subseteq \mathcal{P}^{(\mathrm{BF})}$ and a subset $P_2 \subseteq \mathcal{P}^{(\mathrm{PEF})}$ ▷ two different methods
8: Replace the current partition $\mathcal{P}$ with the perturbed partition $\{(\mathcal{P}^{(\mathrm{BF})} \setminus P_1) \cup P_2, (\mathcal{P}^{(\mathrm{PEF})} \setminus P_2) \cup P_1\}$ if the perturbed partition is predicted to perform better
9: **end for**

For example, a lower dispersivity for PEF reduces both the index size and intersection time, and thus it is a PEF-friendly feature. In fact, dispersivity is also a BF-friendly feature. A non-example is document frequency for PEF: a higher value reduces the index size, while a lower value reduces the intersection time, so it is not PEF-friendly.

When using friendly features in Algorithm 1, when a subset of the documents are moved from one part to the other, "friendly" is defined according to the destination. Specifically, in Algorithm 1, we use PEF-friendly features in Line 3 and for P_1 in Line 7, and we use BF-friendly features in Line 5 and for P_2 in Line 7.

Suppose we apply a time threshold, i.e., an upper bound on the average intersection time. We choose between the current partition and the perturbed partition as follows:

(1) If the current partition and the perturbed partition both satisfy the time threshold, we choose the partition which minimizes index size.
(2) Otherwise, we choose the partition which minimizes time.

Above, both size and time are predicted using a machine-learning model. When instead applying a size threshold, we do the same thing with "time" and "size" swapped.

6 EXPERIMENTAL RESULTS

We experiment on the proposed hybrid BF-PEF index, which we abbreviate to *hybrid-BP*, under various circumstances. We explore the accuracy of the machine-learning regression models, the impact of the initial partition in the local search process, and the performance of hybrid-BP in terms of index size and average intersection time.

We compare hybrid-BP against the mathematical hybrid index proposed in [22], which we call *hybrid-PIV*; we also compare hybrid-BP against the mathematical encoders involved in hybrid-PIV. We also explore the feature distributions in hybrid-BP, and false positive rates.

6.1 Experiment Setup

Our experiments are performed using public GOV2 dataset, containing 25 205 183 documents with 6 220 848 172 postings. We remove extra information, such as HTML tags, while retaining the title and body parts. We do not remove any stop words. For query datasets, we use the public query sets (a) MillionQuery, containing 58 506 queries from the 2007, 2008, and 2009 TREC Million Query Tracks[1], and (b) TerabyteQuery, containing 99 322 queries from the 2006 TREC Terabyte Track[2]. In both query sets, the terms which do not exist in the GOV2 document set are filtered out, and the aforementioned query counts are after filtering. We do not remove the query that only have one term.

All experiments are run on a platform with 2× Xeon E5-2650 v4 CPUs at 2.20GHz with 512GB RAM; both CPUs have 12 physical cores (with 24 threads). The L1 instruction cache and data cache is 32KB, L2 cache is 256KB, and L3 cache is 30 720KB. All the programs are implemented in C++11 and complied with g++ version 5.4.0 with -O2 optimization. The operating system is Linux CentOS 6.5 with kernel version 2.6.32.

We use the downloadable PEF[3] and BF[4] source code for testing, with both using their default settings, and we set the density of BitFunnel to 0.15 for all 11 shards. In the experiments, we reorder the documents according to URL. For machine learning, we use sklearn[5] by python.

6.2 Model Selection

For training and validation of the prediction models, we generate a set of size vectors M_{size} and a set of time vectors M_{time} according to the procedure in Section 5.1.2. We randomly choose 380 corpuses from $\mathcal{F}$, of which 340 are used for generating the training set, and 40 are used for generating the validation set. The training set is generated from the sample vectors for 5000 random pairs of corpuses (where each pair acts as a BF-PEF partition for training), and the validation set is generated from the sample vectors for 1000 random pairs of corpuses. We also add two extra BF-PEF partitions to the training set: one where the whole document corpus is assigned to BF (and no documents are assigned to PEF), and one where the whole document corpus is assigned to PEF.

We generate prediction models for both "size" (i.e., total index size, $s_{\text{BF}} + s_{\text{PEF}}$) and "time" (i.e., average intersection time, $t_{\text{BF}} + t_{\text{PEF}}$). We measure time using the MillionQuery query set. For both size and time, we test three regression models, the gradient-boosted regression tree (GBRT) model, a support vector regression (SVR) model, and a random forest (RF) model.

The test set is from samples generated from 100 BF-PEF partitions (of the whole document corpus), which are randomly generated similar to the procedure described in Section 5.1.2. We randomly choose the number of documents n to assign to BF, and we assign BF the n documents which achieve the top score as per eq. (1) while other documents are assigned to PEF.

We compare the accuracy of the six prediction models (size and time; GBRT, SVR, and RF) against experimental measurements. When a value a is estimated as $\hat{a}$, the *relative error* is defined as $|(\hat{a} - a)/a|$. Table 2 records some statistics of the relative error: its mean, and first, second, and third quartiles.

Table 2 indicates that SVR is the best regression model among the three, in terms of both size and time. Consequently, we exclusively use SVR for the remaining experiments. Moreover, we use the SVR model as trained in this section. The small mean relative error (less than 5%) indicates the predictions are suitable for practical use.

6.3 Initial Partition

In this section, we investigate the influence of the initial partition on the local search method in Section 5.2. We choose initial partitions using the method for generating BF-PEF partitions in Section 6.2: we use 5 with a time threshold, and another 5 with a space threshold. We plot the predicted hybrid index size and average intersection time in Figure 4 where different colors correspond to different initial partitions. We also include points which indicate the actual (i.e., not predicted) values after 500 iterations.

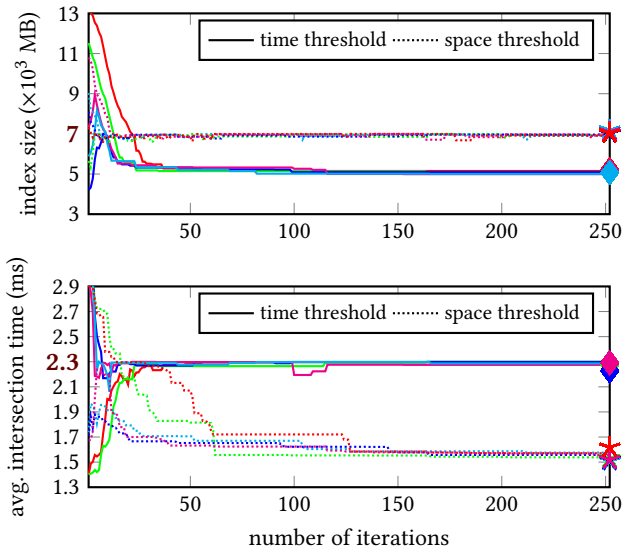


Figure 4: With 5 different initial partitions, after some number of iterations of the local search procedure (x axis), we measure the hybrid index size (top), and the average intersection time (bottom), with either a time threshold of 2.3ms (solid) or a space threshold of 7000MB (dotted). We also include marks to indicate the exact measurements after 500 iterations. (Note that the y axes do not start at 0.)

We make three observations from Figure 4:

(1) Different initial partitions have a negligible impact on the final result, i.e., they approximately converge. This suggests

[1] https://trec.nist.gov/data/million.query.html
[2] https://trec.nist.gov/data/terabyte06.html
[3] https://github.com/ot/partitioned_elias_fano
[4] https://github.com/BitFunnel/BitFunnel
[5] http://scikit-learn.org/stable/index.html

Table 2: The relative error when predicting the index size and average intersection time of the resulting BF-PEF index using GBRT, SVR, or RF; we list the first, second, and third quartiles, and the mean relative error.

	index size				average intersection time			
	1-st quart.	2-nd quart.	3-rd quart.	mean	1-st quart.	2-nd quart.	3-rd quart.	mean
GBRT	3.69%	6.66%	11.82%	8.03%	3.45%	9.49%	26.48%	15.12%
SVR	1.09%	2.03%	2.90%	2.35%	1.77%	3.61%	7.08%	4.81%
RF	3.73%	7.33%	25.16%	14.99%	3.24%	11.92%	24.23%	16.95%

that the local search is not getting stuck in local optima (where we would expect non-converging measurements from various local optima), which gives confidence that the process is reaching a close-to-optimal BF-PEF partition.

(2) Only a small number iterations are needed (200 iterations, say), before an approximately optimal partition is reached. Consequently, there is not a massive overhead introduced by this process.

(3) In these experiments, we see the prediction of the index size and average intersection time is accurate: within 5% of the actual values after 500 iterations. This is consistent with the observations in Section 6.2.

These three observations suggest that the local search method is a suitable approach to finding an approximately optimal BF-PEF partition.

6.4 Time and Space Trade-off

We investigate the trade-off between space (index size) and time (average intersection time). To generate the approximately optimal partition, we use the local search method (500 iterations) described in Section 6.3 with $n = 12\,500\,000$ (half of the whole document corpus) assigned to BF while others are in PEF for initial partition. The observations in Section 6.3 indicate this decision does not have a major impact on the size or time of the resulting BF-PEF partition. In this section the index size and average intersection time are measured after the index is generated rather than predicted by regression models. We do not normalize them as in Figure 1 .

6.4.1 Time threshold. We set a threshold on the average intersection time, and minimize the hybrid index size subject to that threshold. Figure 5 records the size and intersection time of the hybrid index subject to various time thresholds (for the MillionQuery query set). Here and in the next section, we experiment with both the MillionQuery and TerabyteQuery query sets (although model training only uses the MillionQuery query set).

Figure 5 indicates that as the time threshold increases, the size of the hybrid index decreases. This is as expected: as we relax the time threshold, more documents are assigned to PEF, which results in a smaller hybrid index. Indeed, when the time threshold is close to the PEF time, almost all documents are assigned to PEF, in which case there is negligible difference between the hybrid BF-PEF index and PEF itself.

One interesting observation is that when we reduce the time threshold to around the pure BF average intersection time, the size of the proposed hybrid index is reduced by 45% compared to BF. This significant improvement in space comes at virtually no cost in time. We expect this is partly because some documents are short and are infrequently queried: reassigning these from BF to PEF does not have a significant impact on the intersection time (it might even slightly reduce it), but it reduces the index size.

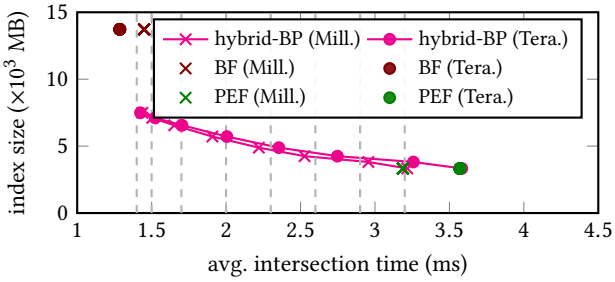


Figure 5: The index size vs. the average intersection time (on the MillionQuery query set (Mill.) and TerabyteQuery query set (Tera.)) of the hybrid BF-PEF index as the time threshold varies (indicated by vertical dashed lines). For comparison, we also include pure BF (red) and pure PEF (green) indexes. (Note that the x axis does not start at 0.)

In Figure 5, some time thresholds on MillionQuery query set are exceeded by a relatively small amount (within 3%), which we attribute to prediction errors in the regression model.

6.4.2 Size threshold. We set a threshold on the hybrid index size, and minimize the average intersection time subject to that threshold. Figure 6 records the size and intersection time of the hybrid index subject to various size thresholds.

For comparison, we also perform the same tests on another hybrid index (hybrid-PIV) proposed by [22][6]; see Section 4 for more about hybrid-PIV. For hybrid-PIV, we also use the MillionQuery query set for training, and the block sampling rate is set to 0.01. (Preliminary experiments with the block sampling rate set to 0.001 were not substantially different, so we do not report these results.)

From Figure 6 we observe that it is generally not possible to improve hybrid-PIV's average intersection time much beyond a certain point (e.g. 2.5ms for the MillionQuery query set) by relaxing the threshold on the index size. This implies the hybrid BF-PEF index has a functional advantage over hybrid-PIV: if memory is available, hybrid-BP can utilize it to reduce the intersection time. Thus, the proposed hybrid BF-PEF index offers greater flexibility in choosing a trade-off than hybrid-PIV.

[6] https://github.com/ot/ds2i

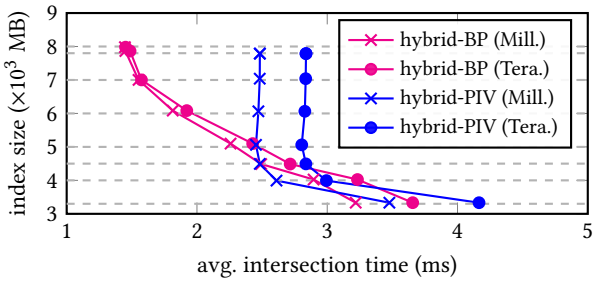


Figure 6: The index size vs. the average intersection time on the MillionQuery and TerabyteQuery query sets of the hybrid BF-PEF index (hybrid-BP) as the size threshold varies (indicated by horizontal dashed lines). There is no result for hybrid-PIV with a 8000MB threshold; its maximum index size is around 7800MB. (Note that both the x and y axes do not start at 0.)

We observe that for the 4000MB threshold, hybrid-PIV outperforms hybrid-BP by 11% and 8% on MillionQuery and TerabyteQuery, respectively. A situation like this amounts to the search engine only having a certain amount of memory they can use (not more nor less). If instead the operator has spare memory and e.g. is able to set a higher size threshold, then Figure 6 indicates that hybrid-BP reduces the average intersection time by up to around 47% with almost the same size.

Compared with PEF on the MillionQuery query set, the maximum improvement in the average intersection time achieved by hybrid-PIV and hybrid-BP is 23% and 55%, respectively; for the 4000MB threshold, the improvement is 18% and 9%, respectively.

In Table 3, we compare hybrid-BP with some indexes compressed by a single form of mathematical encoding. Specifically, we compress the entire GOV2 corpus using each of OptPFD [29], binary interpolative encoding [21], and Varint-G8IU [27], and generate three BF-PEF hybrid indexes with a size threshold approximately the same as the mathematical encoding index it is being compared with. Among them, the binary-interpolative encoded index has the best index size through recursive compression, but its intersection time is long due to recursive decompression. Varint-G8IU uses byte-aligned encoding and SIMD acceleration, making it faster for intersection but its compression is significantly worse; OptPFD offers a trade-off between these two extremes.

When we set the BF-PEF index size threshold equal to the binary interpolative encoding index size (3.13GB), the local search fails to find a BF-PEF partition satisfying the size threshold. The minimum index size of the hybrid BF-PEF index is close to the pure PEF index size, which is 3.25GB. In this case, the hybrid BF-PEF index likely assigns almost all documents to PEF (and acts much the same as pure PEF).

From Table 3, we observe that the hybrid BF-PEF index results in improving the intersection time (vs. the aforementioned mathematical encoding methods) from 16% to 75% on MillionQuery and from 20% to 76% on TerabyteQuery, while having almost the same index size.

6.5 Feature Analysis

We analyze the feature distributions of BF and PEF in the hybrid BF-PEF index in the approximately optimal BF-PEF partition under various threshold conditions. We consider five features: URL, DL, DF, dispersivity, and TF-IDF, as defined in Section 5.1.1. For a feature f, we define the *feature ratio* for $I \in \{\text{BF}, \text{PEF}\}$ as

$$\frac{\overline{f}_I - \overline{f}}{\overline{f}}$$

where $\overline{f}$ is the average value of feature f in the whole document corpus, and $\overline{f}_I$ is the average value of feature f among to the documents assigned to I. The results are plotted in Figure 7 (for time thresholds) and Figure 8 (for size thresholds) for the BF-PEF partitions arising in Section 6.4. For these BF-PEF partitions, we also tabulate the proportion of documents assigned to BF in Table 4.

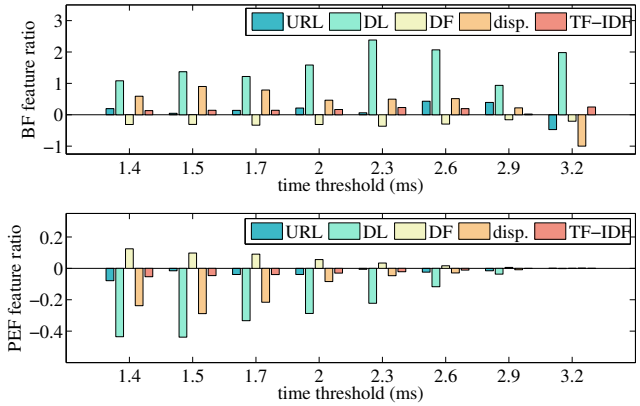


Figure 7: Feature ratios for the BF (top) and PEF (bottom) parts of the hybrid BF-PEF index, under various time thresholds.

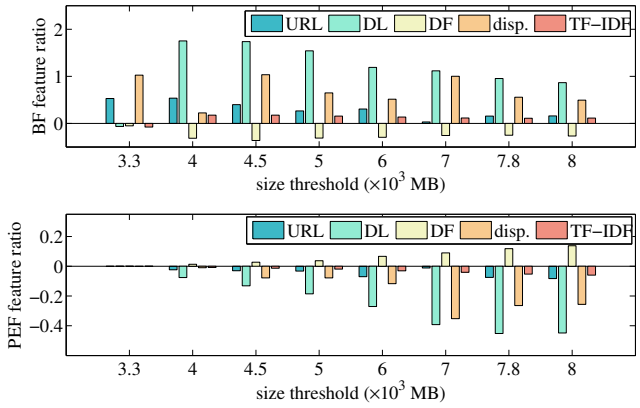


Figure 8: Feature ratios for the BF (top) and PEF (bottom) parts of the hybrid BF-PEF index, under various index size thresholds.

Table 3: The space time trade-off compared with mathematical encoding methods.

	OptPFD	hybrid	bin. int.	hybrid	Varint	hybrid
index size (GB)	3.66	3.77 (+3%)	3.13	3.27 (+4%)	7.74	7.77 (+1%)
avg. intersection time (ms) Mill.	3.65	3.08 (-16%)	12.95	3.28 (-75%)	2.42	1.42 (-41%)
avg. intersection time (ms) Tera.	4.25	3.42 (-20%)	15.06	3.67 (-76%)	2.78	1.43 (-49%)

Table 4: The proportion of documents in the BF part with a given time or size threshold.

time threshold (ms)	1.4	1.5	1.7	2.0	2.3	2.6	2.9	3.2
proportion	28.69%	24.23%	21.47%	15.33%	8.55%	5.36%	3.79%	0.10%
size threshold ($\times 10^3$MB)	3.3	4.0	4.5	5.0	6.0	7.0	7.8	8.0
proportion	0.09%	4.15%	7.02%	10.73%	18.50%	25.99%	32.16%	34.14%

In general, if a feature f has a positive feature ratio for $I \in \{\text{BF}, \text{PEF}\}$, we think of I as favoring feature f. For example, BF generally strongly favors document length (DL) and dispersivity. For a feature f with a negative feature ratio for $I \in \{\text{BF}, \text{PEF}\}$, we instead think of I as disfavoring feature f. For example, BF generally disfavors document frequency (DF). Some of these feature preferences are explainable (like in the pilot experiments in Section 2), while others are not. These results are fairly consistent with Figure 1 in the pilot experiments.

When we set the time threshold equal to the pure PEF intersection time (3.2ms), or the size threshold equal to the pure PEF index size (3300MB), almost all the documents are partitioned into PEF, therefore the feature distribution of the PEF part is almost identical to the whole document corpus; the feature distribution for BF in this setting is consequently unreliable, since it contains few documents.

Figures 7 and 8 further emphasize how a simple method alone is unlikely able to obtain a suitable BF-PEF partition; this is consistent with the observations in Section 2.

6.6 False Positive Analysis

A non-negligible problem with the proposed hybrid index is the false positives caused by BF. Table 5 lists the false positive rates (the proportion of false positive documents among all positive documents) for the proposed hybrid BF-PEF index under various thresholds (the time threshold is for the MillionQuery query set).

Table 5 indicates that, as time threshold increases, the false positive rate decreases; this is due to an increasing number of documents being assigned to PEF, which does not incur false positives. Conversely, as the size threshold increases, more documents are assigned to BF, which increases the false positive rate.

Compared with the pure BF index, the false positive rate of the hybrid BF-PEF index decreases by at least 40% and 23% on MillionQuery and TerabyteQuery, respectively.

Table 5: The false positive (fp.) rates for various thresholds for the MillionQuery (Mill.) and TerabyteQuery (Tera.) query sets. Note that the pure BitFunnel index has 2.48% and 1.70% false positive rates on MillionQuery and TerabyteQuery, respectively.

time thre. (ms)	1.4	1.5	1.7	2.0	2.3	2.6	2.9	3.2
Mill. fp.	1.39%	1.36%	1.03%	0.91%	0.74%	0.33%	0.14%	0.00%
Tera. fp.	1.25%	1.19%	1.01%	0.87%	0.71%	0.32%	0.15%	0.00%
size thre. ($\times 10^3$ MB)	3.3	4.0	4.5	5.0	6.0	7.0	7.8	8.0
Mill. fp.	0.00%	0.24%	0.36%	0.52%	0.86%	1.30%	1.43%	1.49%
Tera. fp.	0.00%	0.23%	0.36%	0.49%	0.84%	1.21%	1.27%	1.31%

7 CONCLUDING REMARKS

In the paper, we propose a hybrid BF-PEF index, combining the spatial efficiency of Partitioned Elias-Fano and intersection efficiency of BitFunnel by partitioning the documents into two disjoint parts. We use a local search to find an approximately optimal BF-PEF partition that meets a pre-specified threshold; during this process, we use machine learning to predict the size and intersection time of the hybrid index.

Experimental results indicate that the proposed hybrid BF-PEF index decreases the intersection time up to 47% compared with another hybrid index with the same index size. Compared with BitFunnel, it reduces the index size by 45% while maintaining almost the same intersection time.

In the future, it would be interesting to explore (a) combining multiple indexes, not just BF and PEF; (b) using more sophisticated thresholding techniques; (c) using alternative regression models; and (d) incorporating additional document features.

One limitation of the present approach is the implicit assumption of everything taking place on a single machine. Scaling up to a distributed system likely requires adapting the machine-learning method for each machine. While we expect this is feasible in practice, this matter is not addressed this in this paper.

8 ACKNOWLEDGEMENTS

This work is partially supported by National Science Foundation of China (61872201, 61702521, 61602266, U1833114); Science and Technology Development Plan of Tianjin (17JCYBJC15300, 16JCYBJC41900, 18ZXZNGX00140, 18ZXZNGX00200); and the Fundamental Research Funds for the Central Universities and SAFEA: Overseas Young Talents in Cultural and Educational Sector.

REFERENCES

[1] Rakesh Agrawal, Behzad Golshan, and Evangelos E. Papalexakis. 2015. A Study of Distinctiveness in Web Results of Two Search Engines. In *Proc. WWW*. 267–273.

[2] Vo Ngoc Anh and Alistair Moffat. 2005. Inverted Index Compression Using Word-Aligned Binary Codes. *Inf. Retr.* 8 (2005), 151–166.

[3] Vo Ngoc Anh and Alistair Moffat. 2010. Index Compression Using 64-bit Words. *Softw., Pract. Exper.* 40, 2 (2010), 131–147.

[4] Burton H. Bloom. 1970. Space/Time Trade-offs in Hash Coding with Allowable Errors. *Commun. ACM* 13, 7 (1970), 422–426.

[5] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32.

[6] Stefan Büttcher, Charles L. A. Clarke, and Gordon V. Cormack. 2010. *Information Retrieval—Implementing and Evaluating Search Engines.* MIT Press.

[7] Berkant Barla Cambazoglu and Ricardo A. Baeza-Yates. 2015. *Scalability Challenges in Web Search Engines.* Morgan & Claypool Publishers.

[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms* (3 ed.). MIT Press.

[9] J. Shane Culpepper and Alistair Moffat. 2010. Efficient set intersection for inverted indexing. *ACM Trans. Inf. Syst.* 29, 1 (2010), 1:1–1:25.

[10] Jay L. Devore. 2011. *Probability and Statistics for Engineering and the Sciences* (8 ed.). Cengage Learning.

[11] Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alexander J. Smola, and Vladimir Vapnik. 1996. Support Vector Regression Machines. In *Advances in Neural Information Processing Systems.* 155–161.

[12] Peter Elias. 1974. Efficient Storage and Retrieval by Content and Address of Static Files. *J. ACM* 21, 2 (1974), 246–260.

[13] Robert Mario Fano. 1971. *On the Number of Bits Required to Implement an Associative Memory.* Massachusetts Institute of Technology, Project MAC.

[14] Jerome H. Friedman. 2001. Greedy Function Approximation: A Gradient Boosting Machine. *Ann. Stat.* 29, 5 (2001), 1189–1232.

[15] Bob Goodwin, Michael Hopcroft, Dan Luu, Alex Clemmer, Mihaela Curmei, Sameh Elnikety, and Yuxiong He. 2017. BitFunnel: Revisiting Signatures for Search. In *Proc. SIGIR.* 605–614.

[16] Andrew Kane and Frank Wm. Tompa. 2014. Skewed Partial Bitvectors for List Intersection. In *Proc. SIGIR.* 263–272.

[17] Daniel Lemire, Owen Kaser, and Kamel Aouiche. 2010. Sorting Improves Word-aligned Bitmap Indexes. *Data Knowl. Eng.* 69, 1 (2010), 3–28.

[18] Xiaozhu Liu and Zhiyong Peng. 2010. An Efficient Random Access Inverted Index for Information Retrieval. In *Proc. WWW.* 1153–1154.

[19] Xinyu Liu, Zhaohua Zhang, Boran Hou, Rebecca J. Stones, Gang Wang, and Xiaoguang Liu. 2018. Index Compression for BitFunnel Query Processing. In *Proc SIGIR.* 921–924.

[20] Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Fabrizio Silvestri. 2007. Mining query logs to optimize index partitioning in parallel web search engines. In *Proceedings of the 2nd International Conference on Scalable Information Systems.* 43–51.

[21] Alistair Moffat and Lang Stuiver. 2000. Binary Interpolative Coding for Effective Index Compression. *Inf. Retr.* 3, 1 (2000), 25–47.

[22] Giuseppe Ottaviano, Nicola Tonellotto, and Rossano Venturini. 2015. Optimal Space-time Tradeoffs for Inverted Indexes. In *Proc. WSDM.* 47–56.

[23] Giuseppe Ottaviano and Rossano Venturini. 2014. Partitioned Elias-Fano Indexes. In *Proc. SIGIR.* 273–282.

[24] Knut Magne Risvik, Trishul M. Chilimbi, Henry Tan, Karthik Kalyanaraman, and Chris Anderson. 2013. Maguro, a System for Indexing and Searching over Very Large Text Collections. In *Proc. WSDM.* 727–736.

[25] David Salomon. 2007. *Variable-length Codes for Data Compression.* Springer London.

[26] Prabhakant Sinha and Andris A. Zoltners. 1979. The Multiple-Choice Knapsack Problem. *Operations Research* 27, 3 (1979), 503–515.

[27] Alexander A. Stepanov, Anil R. Gangolli, Daniel E. Rose, Ryan J. Ernst, and Paramjit S. Oberoi. 2011. SIMD-based Decoding of Posting Lists. In *Proc. CIKM.* 317–326.

[28] Xiujun Wang, Yusheng Ji, Zhe Dang, Xiao Zheng, and Baohua Zhao. 2015. Improved Weighted Bloom Filter and Space Lower Bound Analysis of Algorithms for Approximated Membership Querying. In *Proc. DASFAA.* 346–362.

[29] Hao Yan, Shuai Ding, and Torsten Suel. 2009. Inverted Index Compression and Query Processing with Optimized Document Ordering. In *Proc. WWW.* 401–410.

[30] Marcin Zukowski, Sándor Héman, Niels Nes, and Peter A. Boncz. 2006. Super-Scalar RAM-CPU Cache Compression. In *Proc. ICDE.*

How Do Your Neighbors Disclose Your Information: Social-Aware Time Series Imputation

Zongtao Liu
Zhejiang University
tomstream@zju.edu.cn

Yang Yang*
Zhejiang University
yangya@zju.edu.cn

Wei Huang
Zhejiang University
hw313@zju.edu.cn

Zhongyi Tang
State Grid Zhejiang Jinyun Power Supply Co. Ltd.
tangzhongyi0123@163.com

Ning Li
State Grid Zhejiang Jinyun Power Supply Co. Ltd.
shiepning@163.com

Fei Wu
Zhejiang University
wufei@cs.zju.edu.cn

ABSTRACT

Different time series is measured in almost all fields including biology, economics and sociology. A common challenge for using such data is the imputation of the missing values with reasonable ones. Most of existing approaches to data imputation assume that individual's observations are independent to each other, which is rarely the case in real-world. In this paper, we study the social-aware time series imputation problem. Given a social network that represents social relations between individuals, we propose a sequential encoder-decoder-based framework and build a connection between the missing observations and the social context. In particular, the proposed model employs the attention mechanism to incorporate social context and temporal context into the imputation task. Experimental results based on two real-world datasets demonstrate that our approach outperforms 11 different baseline methods.

CCS CONCEPTS

• **Mathematics of computing** → **Time series analysis**; • **Information systems** → **Social networks**; • **Computing methodologies** → *Neural networks*.

KEYWORDS

time series, missing data imputation, social networks

ACM Reference Format:
Zongtao Liu, Yang Yang*, Wei Huang, Zhongyi Tang, Ning Li, and Fei Wu. 2019. How Do Your Neighbors Disclose Your Information: Social-Aware Time Series Imputation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313714

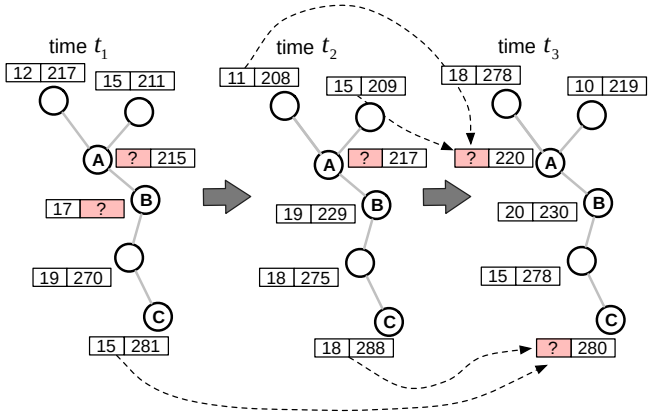


Figure 1: An illustration of the social-aware time series imputation problem. We are given a network where each vertex denotes an individual, and we create an edge between two individuals if the distance between their living places is less than a threshold. Each person has a multivariate observation at every timestamp (two-dimensional in our case), with some missing values (colored in red and marked as "?"), to indicate her electricity consumption. The goal is to impute the missing values by building surrounding and temporal influences.

1 INTRODUCTION

Time-series modeling has been extensively studied and applied in a broad range of applications, such as financial marketing [51], bioinformation [19, 22], and wearable sensors [33]. Among these studies, one fundamental issue is that time series data often inevitably carries missing observations due to various reasons, such as anomaly events, security issues, communication errors, privacy, and so on. Missing observations cause serious damages to the analysis and model results based on them, and thus it is necessary to impute these values.

On the other hand, the social network, a natural way of representing personal relationships, has turned out to be valuable and has recently attracted considerable research efforts [14, 30]. Time series data also plays an important role in social network analysis, as individuals in a social network generate a huge amount of data, which evolves over time. Meanwhile, homophily, a widely recognized organizing principle of social networks, suggests that people with common behavior patterns or similar interests are more likely

*Corresponding author: Yang Yang, yangya@zju.edu.cn.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313714

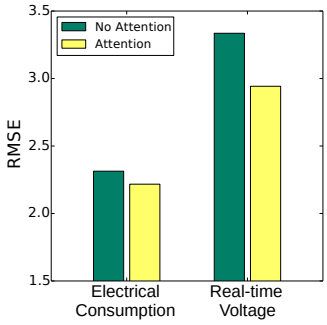


Figure 2: The performance of the proposed method for time series imputation. Two methods are shown here: one considers social and temporal information in the context of the attention mechanism, and another ignores this information. Root Mean Square Error (RMSE) is considered in the figure (the smaller the better).

to connect with each other. Therefore, individuals' missing observations can be guided by their neighbors' data, which is the basis of our work.

To address the problem of time series imputation, various approaches have been explored in the past few decades. For example, one practical method is to replace the missing values with the mean value of observations at adjacent timestamps. Similar methods include smoothing [38], interpolation [1, 45], and spline [36], which are simple and practical when data evolves smoothly. However, these methods can not capture the complex patterns of a time series; therefore, they fail to perform imputation. Recently, deep neural networks-based methods have been employed for data imputation [13, 15, 50]. These methods benefit from their strong prediction ability as well as their ability to capture temporal dependencies. However, to the best of our knowledge, there is a lack of research on the incorporation of *social information* and *deep models* to better perform time series imputation, which we believe is a new promising direction.

In this paper, we study the problem of social-aware time series imputation, which is nontrivial and gives rise to several challenges. Figure 1 demonstrates an example. In particular, given a social network, in which each vertex indicates an individual, there is an undirected edge between two individuals if they live close together (i.e., the distance between their living places is less than a certain predefined threshold). We monitor the electricity consumption of each individual, which is represented by two-dimensional vector, composed of daily power and voltage. At time t_3, individual A's daily power value is missing. To complete the missing value, traditional methods consider A's previous power value at the timestamps t_1 and t_2, which, however, are also missing. Fortunately, observations of A's neighbors provide us with a clue for the missing value. For instance, it is natural to assume that people live close share similar electricity consumption patterns to some extent. We call this "surrounding influence". Intuitively, each of individual v's neighbors has different surrounding influence on v's observations. The first challenge in this study is therefore how to identity the surrounding influence and model the connection between the missing observations and social context of an individual.

Returning to our example, the missing observations of individual C at time t_3 are affected by C's observations at t_1 and t_2. We call this "temporal influence", which has different effects for different timestamps. For instance, if some of C's family leave for a trip during t_2, and return at t_3, the electricity usage of C's family at t_1 shall be closer to that at t_3 and has a larger influence than that at t_2. How to quantify and differentiate temporal influence at different timestamps is the second challenge in this work.

Another factor that influences data imputation in our example is the time intervals between t_1, t_2, and t_3. Most existing time-series imputation methods assume that the elapsed time between the elements of a time series is equal, which may not hold true in practice. How to handle irregular time intervals is our third challenge.

To address the above mentioned challenges, we propose an novel model based on a deep encoder-decoder architecture. It utilizes the attention mechanism to incorporate surrounding influence and temporal influence in the time series imputation task. As Figure 2 shows, the experimental results show that with the help of surrounding influence and temporal influence, we can obtain a clear improvement in imputation performance. Furthermore, we apply Time-Aware Long Short-Term Memory (LSTM), which acts as the basic component of our model for handling the time irregularity issue. See details in Section 3.

We further test the proposed model on two real-world datasets. We compare 11 different state-of-the-art baseline methods and find that our model consistently outperforms others. More detailed results can be found in Section 4.

Accordingly, our contributions are as follows:

- We propose a novel time series imputation framework that takes both surrounding influence and temporal influence into consideration.
- We apply a variant LSTM as the basic component of our model for handling irregular time intervals.
- We construct sufficient experiments under various missing conditions based on two real-world datasets. Experimental results exhibit our method's advantage over eleven baseline methods.

Organization. The remainder of this paper is organized as follows. We first formulate the problem and give the necessary definitions. We then introduce the proposed model, including the model structure, the learning algorithm, and its applications. Subsequently, we present the experimental results. Finally, we review some relevant research and conclude the work.

2 PROBLEM DEFINITION

Let $G =< V, E >$ be a social network, where V is the set of users, and each edge $e_{ij} \in E$ indicates that $v_i \in V$ has a social relationship with v_j. For each user v, her behavior data is measured by a sensor periodically, denoted as $X = \{x_1, x_2, ..., x_T\} \in \mathbb{R}^{T \times D}$. For each timestamp $t \in \{1, 2, ..., T\}$, x_t indicates the t-th observations with D channels (or variables), and x_t^d represents the t-th observation in the

d-th channel. In the real world, many observations are missing or erroneous. To mark the missing data in the time series, we introduce an index m_t for missing ones at timestamp t, where $m_t = 0$ if x_t is not observed, otherwise $m_t = 1$. We also maintain an observed sequence of X as $X' = \{x_{s_1}, x_{s_2}, ..., x_{s_L}\}$, where s_l represents the timestamp of l-th observations, and the corresponding interval sequence Δ, where

$$\delta_l = \begin{cases} 1, & l = 1 \\ s_l - s_{l-1}, & l \neq 1 \end{cases} \tag{1}$$

In this work, we are interested in the missing data imputation task of time series data from users in social networks, where we impute the missing data given the social network G, time series dataset $\mathcal{D} = \{X_n, M_n\}_{n=1}^N$, where $X_n = \{x_1^{(n)}, ..., x_T^{(n)}\}$, $M_n = \{m_1^{(n)}, ..., m_T^{(n)}\}$.

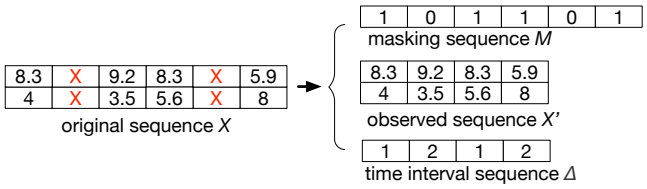


Figure 3: An example of the whole sequence X, masking M, observed sequence X', and time intervals Δ.

3 OUR APPROACH

Recently, recurrent sequences generated by data through an attention mechanism have shown very good performance in multiple tasks, such as neural translation [10, 28], speech recognition [6], and image caption generation [27]. To some extent, our task can also be viewed as a recurrent sequence generation task that generates the complete sequence X^* using the original sequences X from a user v and her social network G. We design an attention mechanism considering two types of information, i.e., surrounding influence and temporal influence, to learn a sequence model for the imputation task. To handle the time irregularity due to missing data, we use a variant LSTM cell considering different time intervals.

In this section, we describe the details for our proposed model **S**ocial-Aware **T**ime Series **I**mputation (STI). Figure 4 shows the architecture of our proposed model.

3.1 Social Attention Network

In order to capture the surrounding influence patterns for imputation, we leverage the attention mechanism to align the useful information from users with similar behavioral patterns. We also shows our solution to handle the time irregularity in the sequences with missing values. This part of our proposed model is called Social Attention Network.

LSTM and variant LSTM. To extract recurrent series patterns from the original sequences with non-uniformed time intervals, we first review a variant LSTM cell. The LSTM [21] recently shows its good properties in modeling series data with recursive relations. For instance, it can avoid the vanishing gradient problem by introducing the gate mechanism. Given the input x_t, the output of hidden state h_t is computed as below:

$$g_t = tanh(W_g x_t + U_g h_{t-1} + b_g) \tag{2}$$
$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{3}$$
$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{4}$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_f) \tag{5}$$
$$c_t = f_t \cdot c_{t-1} + i_t \cdot g_t \tag{6}$$
$$h_t = o_t \cdot tanh(c_t) \tag{7}$$

where $h_t, c_t \in \mathbb{R}^H$, H is the cell size, $\sigma(\cdot)$ is the logistic sigmoid function, and i, f, o, and g represent *input gate*,*forget gate*, *output gate* and *cell state* respectively. The three gates control the data flow through the cell. In particular, the input gate i regulates the extent to which a new value data is fed into the cell, the forget gate f regulates the extent to which the history is forgotten, and the output gate o decides the extent to which the value is used to compute the output activation.

By introducing the gate structure, the LSTM unit is capable of handling long-term dependency, but the structure has implicitly assumed that the elapsed time between the elements of a sequence is uniformly distributed. Hence, the time irregularity in a sequence is not considered as being part of its structure, while it often occurs in the sequences with missing elements, that is, the interval sequence Δ is non-uniform.

To capture the patterns with time irregularity using LSTM, many studies have proposed modifications to original LSTM to incorporate the elapsed time into the standard LSTM. One method involves imputing data to maintain the regular time gaps. Based on this idea, some studies impute data by sampling from the observed elements, and many studies have also explored imputing data that is learned in the optimization process. However, the imputation method can have a serious influence on performance as the imputed data might not reflect reality well. Another way of handling the time irregularity is to modify the structure of LSTM to utilize the time gaps. Pham et al. [34] multiplies the output of the forget gate by $g(\delta)$, i.e., $f_t = g(\delta_t) \cdot f_t$. Another way of considering the time irregularity is to adjust the memory cell by applying the short-term memory discount. The mathematic expressions of such an adjusted memory are as follows:

$$c_t^s = tanh(W_d c_{t-1} + b_d) \tag{8}$$
$$\hat{c}_t^s = c_{t-1}^s \cdot g(\delta) \tag{9}$$
$$c_{t-1}^l = c_{t-1} - c_{t-1}^s \tag{10}$$
$$c_{t-1}^* = c_{t-1}^l + \hat{c}_t^s \tag{11}$$
$$\tilde{c} = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{12}$$
$$c_t = f_t \cdot c_{t-1}^* + i_t \cdot \tilde{c} \tag{13}$$

Firstly, the short-term memory component (c_t^s) is computed by a network. This decomposition process is data-driven, and its parameters are updated at the mean time with the rest of the network parameters. Secondly, the short-term memory is discounted by a weight factor $g(\delta)$ to obtain the discounted short-term memory ($\hat{c}_t^s$). Here, $g(\cdot)$ is a heuristic decaying function which is monotonically non-increasing. Different types of $g(\cdot)$ can be chosen according to specific application domains. For instance, $g(\delta) = 1/\delta$ is appropriate for

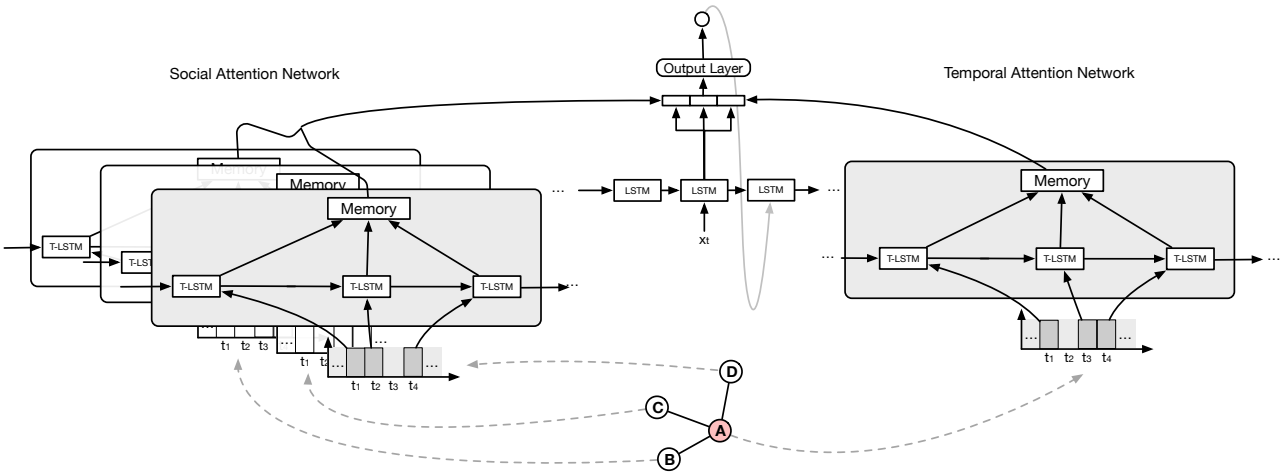


Figure 4: An illustration of our model, Social-Aware Time Series Imputation (STI). Our model mainly consists of two components: Social Attention Network (left) and Temporal Attention Network (right). In the social attention network, we first compute the fix sized memory representations of neighbors' temporal data, and then use the concatenation with the current hidden states of the user to generate the social context vector. In the temporal attention network, we directly feed users' memory representation with the current hidden states to generate the temporal context vector. The concatenation of current hidden states, social context vector and temporal context vector is used to predict the target missing elements.

datasets with small elapsed times, and $g(\delta_t) = 1/log(e+\delta)$ is preferable for datasets with large elapsed times [34]. We tried several functions $g(\cdot)$ but did not observe a drastic difference in the experimental results; however, $g(\delta) = 1/\delta$ performed slightly better that other functions. Thus, we choose the decaying function $g(\delta) = 1/\delta$ in this work. Finally, to obtain the adjusted memory, the complementary subspace of the long-term memory ($c^l_{t-1} = c_{t-1} - c^S_{t-1}$) is combined with the discounted short-term memory. This variant is proposed in [7], named **T**ime-Aware LSTM (T-LSTM).

Social Context Representation. Attention-based RNNs have been successfully applied in many tasks. These models iteratively feed their inputs by selecting relevant contents at each step. We introduce our extensions to this mechanism to extract useful patterns from social context.

For each user v, we have a group of sequences $\{\hat{X}_{(1)}, \hat{X}_{(2)}, ..., \hat{X}_{(P)}\}$ collected from her neighbors $\hat{v} = \{\hat{v}_{(1)}, \hat{v}_{(2)}, ..., \hat{v}_{(P)}\}$. We extract their observed sequences $\hat{M} = \{\hat{M}_{(1)}, \hat{M}_{(2)}, ..., \hat{M}_{(P)}\}$ and time interval sequences $\hat{\Delta} = \{\hat{\Delta}_{(1)}, \hat{\Delta}_{(2)}, ..., \hat{\Delta}_{(P)}\}$. For each neighbor $\hat{v}_{(p)}$, a T-LSTM network described in Section 3.1 is used to encode its original series data, which takes observed sequence $\hat{X}'_{(p)}$ and interval sequence $\hat{\Delta}_{(p)}$ as input and produces a sequence of hidden states $\hat{h}_{(p)} = \{\hat{h}_{1(p)}, ..., \hat{h}_{s(p)}\}$ by the means of following iterative process:

$$h_{s(p)}, c_{s(p)} = T - LSTM(x'_{s(p)}, \delta_{s(p)}, h_{s-1(p)}, c_{s-1(p)}) \tag{14}$$

It is important to note that the network parameters of T-LSTM are shared for all neighbors.

The decoder is a standard LSTM network that reconstructs the complete sequence X^*. Given the recurrent states h^*_s, we can calculate the social context vector $\hat{a}$. The context vector, also known as attention vector, is a weighted average of the source states. In the studies, Bahdanau et al. [5] and Luong et al. [28] proposed context-based models that calculate the context vector by comparing encoder and decoder at each step. Such computations are expensive. To leverage the attention mechanism without sacrificing too much efficiency, a memory-based attention model was explored in [10]. In the current work, we choose the memory-based attention method to efficiently calculate the social context vector.

In the memory-based attention method, a fixed-size memory representation is updated during the encoding. The memory representation is matrix $C \in \mathbb{R}^{K \times H}$, where K is the number of temporal context vectors and a hyper-parameter in this model, and H is the size of the cell states. C is a linear combination of the encoder states, weighted by a score vector $\alpha^s \in \mathbb{R}^K$:

$$C_k = \sum_{s=0}^{|S|} \alpha_{sk} h_s \tag{15}$$

$$\alpha_{sk} = softmax(W_\alpha h_s \cdot l_s) \tag{16}$$

where $W_\alpha \in \mathbb{R}^K$ is a parameter matrix, and l_s is a vector of position-encodings that prevents the learned context from being symmetric. In particular, the position-encodings are used to force the first few context vectors $C_1, C_2...$ to focus on the start of the sequence and the last few vectors $...C_{k-1}, C_k$ to focus on the end of the sequence. Similar to network parameters of T-LSTM, W_α is shared for all neighbors. We adapt the formula proposed in [10] to obtain l_s:

$$L_{ks} = \left(1 - \frac{k}{K}\right)\left(1 - \frac{s}{S}\right) + \frac{k}{K}\frac{s}{S} \tag{17}$$

where, $k \in \{1, 2, ..., K\}$ is the context vector index, and S is the maximum sequence length across all source sequences. Hence, given the social context, we can compute their memory representations

$\hat{C} = \{\hat{C_{(1)}}, \hat{C}_{(2)}, ..., \hat{C}_{(P)}\}$. We define the concatenation of these matrix as $\tilde{C}$, sized $KP \times H$.

In the decoding step, we set the social context vector $\hat{a}$, in each step, by the means of linear combination of the rows in $\tilde{C}$ weighted by a score vector $\hat{\beta}$.

$$\hat{a} = \sum_{i=0}^{K} \hat{\beta}_i \tilde{C}_i \tag{18}$$

$$\hat{\beta} = softmax(W_{\hat{\beta}} h^*) \tag{19}$$

where h^* represents the current hidden states of the decoder, and $W_{\hat{\beta}}$ is the parameter matrix. In each step of the decoding, the current state h_t^* is computed by feeding the previous state h_{t-1}^*, the previous cell state c_{t-1}^* , and previous output x_{t-1}^* to decoding unit:

$$h_t^*, c_t^* = LSTM(x_{t-1}^*, h_{t-1}^*, c_{t-1}^*) \tag{20}$$

3.2 Temporal Attention Network

In the social attention network, we extract social contextual information from user v's social network. Another important factor that should be considered for imputation is temporal influence, that is, how a user's historical and future behaviors can relate to her current state. Similar to the social attention network, we use a memory-based attention network to capture such influence. We name this part of our proposed model Temporal Attention Network.

Given a specific user, we extract her observed sequence X' and interval sequence Δ. Like extracting memory representation before, we also compute its memory matrix C. More specifically, we first feed X' and Δ into another T-LSTM:

$$h_s, c_s = T - LSTM(x'_s, \delta_s, h_{s-1}, c_{s-1}) \tag{21}$$

Hence, we get their hidden states $h = \{h_1, ..., h_s\}$, then we calculate memory representation C by (15).

In the decoding phase, we similarly compute the temporal context vector a. A slight difference is that its input is the memory representation of a single user, rather than the concatenation of memory representations from her neighbors:

$$a = \sum_{i=0}^{K} \beta_i C_i \tag{22}$$

$$\beta = softmax(W_\beta h^*) \tag{23}$$

3.3 Learning and Imputation

We concatenate the hidden representation h_t^*, social context vector $\hat{a_t}$, and temporal context vector a_t , which preserves the information from current states, social context and temporal context for predicting the targets. We then feed the concatenation to a fully connected layer and get the final prediction value.

$$x_t^* = \phi(h_t^*, \hat{a}_t, a_t) \tag{24}$$

where, $\phi(\cdot)$ is a non-linear function that maps the concatenation to the observations.

The decoding phase aims to generate the whole sequence and to minimize potential error in estimating the missing values. However, we cannot compute the observation that is truly missing in the sequence. Instead, we only calculate the error between the estimates and the actual observations. Thus, the total loss/error for the entire dataset $\mathcal{D}$ is the partial mean squared error (the $l2$-regularization term is omitted in the following expression):

$$\mathcal{L}(X^N, X^{*N}) = \sum_{n=1}^{N} [\sum_{t=1}^{T} \sum_{d=1}^{D} m_t^{(n)} \times (x_t^{d(n)} - x_t^{*d(n)})^2)] \tag{25}$$

To train this model, we process social context $\hat{X}$ to social attention network and original sequence X to temporal attention network, keeping track of each output and the latest hidden state. The first hidden states h_0^* of decoder is given by the last hidden states of the temporal context networks . To make the model converge faster and avoid over-fitting, we adopt a mixed strategy to feed the input data into the decoder. For some observations, we use the real target values as each next input if they are not missing; otherwise, we use the decoder's estimates as the next input. For the missing elements, we directly give the decoder's estimates. The fraction of samples trained using the real target values is denoted as γ. In the section of experiment, we set $\gamma = 0.9$.

After the model is well-trained, we can use it to impute the missing elements. Like in the learning phase, we process the social context $\hat{X}$ and original sequence X to the encoder and obtain the candidate context vector $\tilde{C}$ and C. For the decoder, we give the original values when data is observed; otherwise, we give estimates in order to use as much information as possible from the original sequences.

Algorithm 1 training procedure

while not converged **do**
  draw a mini-batch of sequences $\mathbf{X}^{(n)}$ and their corresponding social context sequence sets $\hat{\mathbf{X}}^{(n)}$
  // forward pass to encoder network
  compute social context vector $\hat{\mathbf{a}}^{(n)}$ and temporal context vector $\mathbf{a}^{(n)}$
  //we omit the symbol of batch size (n) in the following statements
  for t in $1, 2, ..., T$ **do**
    sample $p \sim \mathcal{U}(1)$
    if $p > \gamma$ **then**
      $h_t^*, c_t^* = LSTM(x_{t-1}^*, h_{t-1}^*, c_{t-1}^*)$
    else
      $h_t^*, c_t^* = LSTM(x_{t-1}^* \cdot (1 - m_{t-1}) + x_{t-1} \cdot m_{t-1}, h_{t-1}^*, c_{t-1}^*)$
    end if
    $x_t^* = \phi(h_t^*, \hat{a}_t, a_t)$
  end for
  compute the loss function $\mathcal{L}$
  // backward pass
  compute gradients and apply updates
end while

4 EXPERIMENTAL RESULTS

4.1 Datasets Description

We employ two datasets from the State Grid, the major electric power company in China, to construct our experiments. Before

introducing our datasets, we highlight several facts about electricity data collection. Electricity data, including values of real-time current, voltage, and power, is measured by watt-hour meters. Each meter periodically transmits different types of data to its corresponding data collector. Usually, a collector receives data from several meters that are geographically close. For instance, these meters are assembled in different rooms on the same floor of a building. Hence, we can consider that the meters connecting to the same collector record the electricity usage of people or families with social relationships. Therefore, given a user v (corresponds to a meter), we define v's neighbors, which have social relations with v, as other users that share the same collector with v. More details of our datasets are described below:

- **Electrical Consumption (EC) :** This dataset is provided by the State Grid. In total, it includes the daily electrical consumption recorded by 80,000 watt-hour meters. This data spans from January 1st 2018 to March 31st 2018. The length of each series is 90.
- **Real-Time Voltage (RV) :** This dataset, provided by the State Grid, consists of around 20,000 electricity load series, each of which describes voltage values in three phases, recorded every 45 minutes in each day, from June 1st 2018 to June 30th 2018. Each sequence has 32 time stamps.

To reduce computational complexity, we at most consider 8 neighbors of a user, which are selected randomly.

4.2 Baselines and Metrics

4.2.1 Comparative Baselines and Implementation Details. We compare our model with multiple baselines:

- *Mean, Median:* We take the mean and median values of the observations in each channel to replace the missing data.
- *Linear, Cubic:* Linear interpolation and cubic interpolation are two interpolation methods that are used to impute the data.
- *KNN:* This method uses k-nearest neighbor to find similar samples and imputed unobserved data by calculating the weighted average of similar observations.
- *MICE [11]:* This is a method that combines the multiple imputation technique [37] with the chained equation approach to impute the missing data.
- *MissForest [40]:* This is a non-parametric imputation method that uses random forests trained on the observed values to predict the missing values.
- *Soft-Impute [29]:* This approach iteratively updates the missing elements by efficiently solving the convex relaxation of the approximate matrix completion problem.
- *GRU-D [13]:* This is a GRU-based model that incorporates masking and time intervals inside the GRU architecture to capture the informative missing data. We implement it slightly different from the original model proposed in [13]. Rather than use the hidden features of the last cell to predict the label, we use the hidden features in each cell to predict the next observations.
- *LSTM-Impute [26]:* This method initializes the missing elements with mean values, and updates them by training an RNN model with a LSTM unit.
- *VAE [47]:* This is a MCMC-based missing data imputation technique with a trained Variational Autoencoder (VAE).
- *STI :* This is our proposed model. We initialize our model with different parameters for different datasets. For the RV dataset, we use a 1-layer 32-unit T-LSTM encoder and a 1-layer 32 unit LSTM decoder. The memory size is also 32. The model is optimized using an Adam optimizer with a learning rate of 1×10^{-3} in the encoder and 5×10^{-3} in the decoder, and the batch size is 256. For the EC dataset, we set all LSTM dimensions to 16 and the memory size to 16. Again, we train the model using an Adam optimizer with a learning rate of 2×10^{-4} in our encoder and 2×10^{-3} in our decoder, and the batch size is 512. We set the dropout rate as 0.2 (0.8 keep probability).

4.2.2 Evaluation Metrics: We use Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) to measure our model. Their formulas are:

$$MAE = \frac{\sum |x - x^*|}{n}$$
$$RMSE = sqrt\left(\frac{\sum (x - x^*)^2}{n}\right)$$

where x is a prediction, x^* is the ground truth, and n is the number of cases.

4.3 Performance Comparison with Randomly Missing Elements

To evaluate the performance of our model on these two datasets, we first assume that elements are randomly missing. With a missing rate θ, we randomly drop values for each time series from a Bernoulli distribution $\mathbf{U}(\theta)$. We experiment with θ varying from 0.2 to 0.6. In the training phase, we select the observed sequences for training our model, and use the dropped values to test its performance.

Table 1 reports the performance of our proposed method compared to all the other competing methods mentioned in Section 4.2.1 on the RV and EC datasets. Our proposed model significantly outperforms all comparative methods by achieving the lowest RMSE and MAE on these datasets. Another finding is that performance drops as the missing rate increases, which might be caused by the less information these models can learn with. Specifically, the simple imputation methods (Median and Mean) do not perform well, because they only use the observed values to complete the fixed values in each channel. For interpolate-based methods (Linear and Cubic), they further consider the intra-stream information at the nearest timestamp, and thereby achieve better performance. However, they lack the insight of inter-stream information. KNN, SoftImpute, MissForest, and MICE are traditional imputation methods that consider both intra-stream and inter-stream useful information. However, they fail to capture the complex temporal dependence. Therefore, our proposed method significantly outperforms those methods.

For neural networks-based methods, our method outperforms VAE, LSTM-Impute, and GRU-D. LSTM-Impute performs poorly in our evaluation, and its performance on the EC dataset is even worse than some traditional imputation methods. This might be because replacing the missing elements with average values can influence

Table 1: Performance of different models in imputation using the EC and RV datasets.The best results are in boldface, and the second-best results are underlined.

Dataset	Missing Rate	0.2		0.3		0.4		0.5		0.6	
	Method	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
EC	Mean	3.3787	4.3235	3.3794	4.3263	3.3810	4.3295	3.3850	4.3375	3.3913	4.3498
	Median	3.2818	4.5337	3.2850	4.5394	3.2905	4.5478	3.3015	4.5654	3.3151	4.5838
	Linear	1.5783	2.5173	1.6246	2.5835	1.6674	2.6431	1.7249	2.7246	1.7972	2.8248
	Cubic	2.0246	3.1914	2.1461	3.4118	2.2667	3.6288	2.4358	4.0081	2.6691	4.7918
	KNN	2.2455	3.3251	2.4224	3.5077	2.5762	3.6617	2.7576	3.8407	2.9672	4.0431
	SoftImpute	2.4018	3.5193	2.6459	3.7814	2.8377	3.9767	2.9746	4.1007	3.0319	4.1303
	MissForest	4.0659	5.3842	4.0528	5.3695	4.0474	5.3664	4.0294	5.3412	4.0068	5.3174
	MICE	3.4634	4.5654	3.4590	4.5777	3.4578	4.5919	3.4538	4.6152	3.4550	4.6591
	VAE	1.5375	2.3085	1.5883	2.4382	1.6504	2.4979	1.6882	2.6148	1.7374	2.6515
	LSTM-Impute	3.0315	4.2238	3.1687	4.3324	3.2529	4.3206	3.4526	4.5627	3.7708	4.7990
	GRU-D	1.7024	2.5568	1.9385	2.7868	2.0511	2.9136	2.0780	2.9304	1.9568	2.8918
	STI	**1.4667**	**2.2172**	**1.4864**	**2.2574**	**1.5207**	**2.3745**	**1.5696**	**2.3924**	**1.6159**	**2.4505**
RV	Mean	4.0893	5.0340	4.0957	5.0435	4.1076	5.0581	4.1184	5.0835	4.1547	5.1397
	Median	4.0250	5.2811	4.0465	5.2929	4.0701	5.3301	4.0975	5.3541	4.1594	5.4246
	Linear	2.0697	3.4058	2.1316	3.4778	2.2179	3.5714	2.3255	3.7051	2.5487	3.9549
	Cubic	2.7329	4.4551	2.8801	4.7857	3.0976	5.3014	3.3495	5.8316	3.9971	7.7123
	KNN	3.1175	4.3509	3.3162	4.5230	3.5550	4.7334	3.8224	4.9665	4.1645	5.2793
	SoftImpute	4.0263	5.1599	5.4152	6.9389	6.4592	8.4186	6.4171	8.4777	5.3860	7.0291
	MissForest	4.1727	5.3729	4.1825	5.3942	4.2012	5.4243	4.2203	5.4701	4.2952	5.5940
	MICE	4.3518	5.7909	4.3806	5.8305	4.4099	5.8764	4.4302	5.9083	4.4641	5.9477
	VAE	2.3001	3.2631	2.7272	4.5136	3.3440	6.4581	3.6293	6.7901	4.4053	8.8703
	LSTM-Impute	3.0315	4.2238	3.1687	4.3324	3.2529	4.3206	3.4526	4.5627	3.7708	4.7991
	GRU-D	2.8582	4.1190	3.0640	4.3150	3.1822	4.3652	3.1583	4.4811	3.5772	4.7590
	STI	**2.0008**	**2.9426**	**2.0787**	**3.0858**	**2.1258**	**3.1306**	**2.2795**	**3.3187**	**2.4963**	**3.5972**

the original temporal distribution, which harms performance. GRU-D designs a recurrent architecture that uses both time intervals and masking information to update the missing data. However, it overlooks the importance of global temporal contextual information. VAE is a good approach for replacing missing data, but it lacks the modeling of temporal patterns. The best performance of our method demonstrates that incorporating social and temporal contextual information through the attention encoder is helpful for imputation.

4.4 Ablation Analysis

This section aims to demonstrate the effectiveness of the different components of our proposed model. We first list some variants of our models:

- *STI :* This is our complete model. We have described the parameter settings for this model in Section 4.2.1.
- *STI - social:* This model drops the social context vector in the output layer.
- *STI (LSTM)*: This variant uses the original LSTM as the encoder instead of T-LSTM. The remaining settings for this variant are the same as for our proposed model.
- *STI - social - temporal*: This variant removes both temporal and social contexts in our proposed model, i.e., an RNN with a LSTM cell.

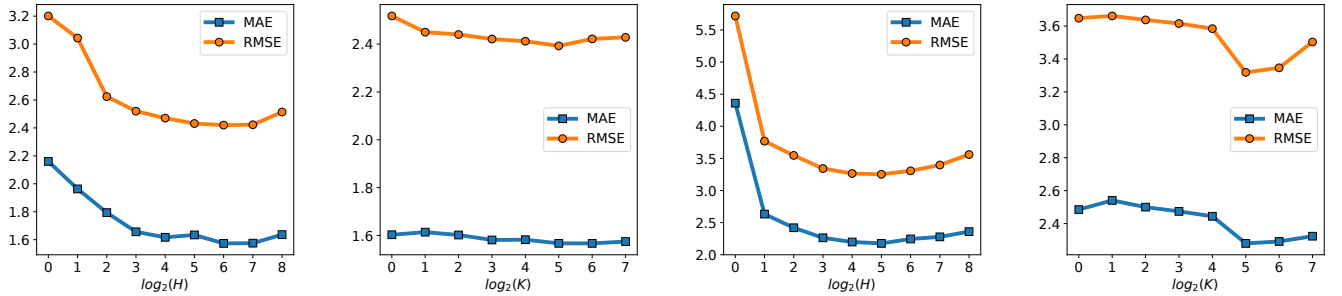
The result of different variants under various settings is reported in Table 2. We can observe the improvement due to adding the social attention network in most cases. This finding suggests that the factor of social relations is helpful, especially on the RV dataset. We also note that our model without social attention network still outperforms most cases when we drop our attention encoder, which demonstrates the importance to model temporal influence in a proper way. We also observed that when we replace the T-LSTM encoder as a LSTM encoder, performance drops. Therefore, using the T-LSTM encoder is helpful.

4.5 Parameter Analysis

In this section, we explore how the hyper-parameters influence the performance of our model. We fix the remaining hyper-parameters of our models when we vary one hyper-parameter. We present the results for the EC and RV datasets with the same missing rate of 0.5. Figure 5(a) and Figure 5(c) report the relationship between cell size H and final performance. We observed that our model, with a cell size that is too small (less than 4 on both datasets) or too large (more than 32 on RV) can decrease imputation performance. Hence, choosing a proper cell size H is crucial. Memory size K is another crucial hyper-parameter. Figure 5(b) and Figure 5(d) present how K influences the model's performance. For memory-based attention, memory size K can be considered as a trade-off between computational time and representational power. A large K allows the model to compute complex source representations, while a small K of 1 limits the source representation to a single vector. Britz

Table 2: Ablation analysis for the EC and RV datasets. The best results are in boldface and the second-best results are underlined.

Dataset	Missing Rate	0.2		0.3		0.4		0.5		0.6	
	Method	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
EC	STI	1.4667	**2.2172**	**1.4864**	**2.2574**	**1.5207**	2.3745	1.5696	**2.3924**	**1.6159**	**2.4505**
	STI - social	**1.4628**	2.2337	1.4985	2.3364	1.5463	**2.3432**	**1.5672**	2.4208	1.6161	2.4593
	STI (LSTM)	1.4732	2.2894	1.4982	2.3165	1.5471	2.3747	1.5837	2.4231	1.6626	2.4938
	STI - temporal - social	1.5066	2.3134	1.5384	2.4002	1.5822	2.4175	1.5903	2.4510	1.6851	2.5350
RV	STI	**2.0008**	**2.9426**	**2.0787**	**3.0858**	**2.1258**	**3.1306**	**2.2795**	3.3187	**2.4963**	3.5972
	STI - social	2.0487	3.0071	2.1167	3.1362	2.1383	3.1392	2.2912	3.3465	2.5390	3.6977
	STI (LSTM)	2.0584	2.9972	2.0916	3.1781	2.1498	3.1482	2.2826	**3.2925**	2.5219	**3.5675**
	STI - temporal - social	2.0641	3.0035	2.1920	3.1756	2.2661	3.2115	2.3573	3.3520	2.6095	3.7819



(a) Varing cell size H on EC dataset. (b) Varing memory size K on EC dataset. (c) Varing cell size H on RV dataset. (d) Varing memory size K on RV dataset.

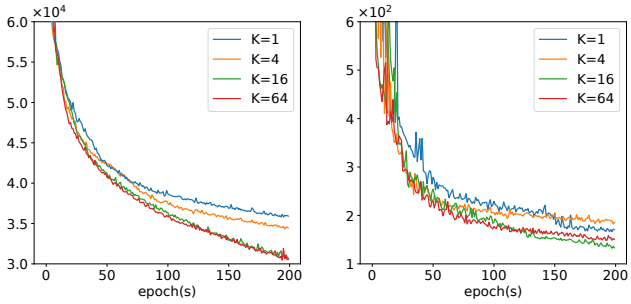
Figure 5: Model parameter analysis. (a), (b) testing on EC dataset and (c), (d) testing on RV dataset. The Missing ratio is set as 0.5. (a), (c) presents the sensitivity of cell size H and (b), (d) shows the sensitivity of memory size K.

et al. [10] showed that a larger K can achieve better performance in neural translation task. In our experiment, we find that within a certain range, the performance increases as the K increases in size. However, we also note that performance drastically drops when K becomes relatively large. This might be related to the over-fitting phenomenon, whereby the capability of representation becomes large.

We also explore how varying Ks influences the learning procedure. Figure 6 shows that a larger K tends to have a faster convergence in terms of training loss on both two datasets.

4.6 Generalization Analysis

In the previous sections, we use the full dataset to train the model, and it achieves very good performance. However, in many real-world senarios, the method is impractical due to the huge scale of historical data and the new data that is being collected continuously. Based on these concerns, we conduct a new experimental setting to explore how our model performs when trained with a subset of collected data. In particular, we randomly draw $1/n$ ($n = 1, 2, 4, \ldots, 64$) from the EC dataset with a missing rate of 0.5 , and use this data to train the model. Then we use the trained model to fill the missing elements in the full datasets. Figure 7 presents the results. Not surprisingly, performance drops as our model is trained with less



(a) Training curve on the EC dataset. (b) Training curve on the RV dataset.

Figure 6: Comparison concerning varing memory size K for the training loss curve on EC and RV datasets. It shows that a larger K tends to converge faster.

data in both metrics. However, we also observe that the degree of the drop is relatively slight. For instance, using only 1/32 (around 2.5K time series) of the datasets, our model can still outperform other baselines. This suggests that our model has a good capacity for generalization.

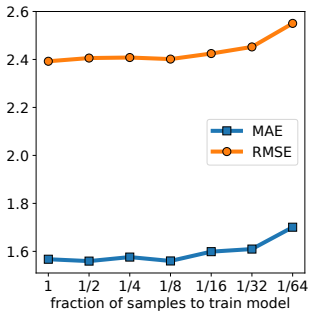


Figure 7: Comparison of performance when applying a subset of the whole EC dataset with a missing rate of 0.5 to train our model. The x-axis indicates the fraction of samples is used. The y-axis indicates the two evaluation metrics.

4.7 Comparison of Performance using Simulated Real-World Missing Data

In the previous section, we experiment with the assumption that elements are randomly missing and observed that our method outperforms other baselines with various missing rates. However, in the real-world, completely random missing is rare in time series , because data is usually missing in blocks. To evaluate the accuracy of our method in completing the elements missing in real-world situations, we establish a new setting in the EC dataset. Firstly, we extend the time span of the EC dataset from the first 90 days to the first 180 days in 2018. We then partition the extended dataset into two parts, named EC-A and EC-B , each of which has consecutive data for 90 days. We extract the masking matrix from EC-A and EC-B, and name them M_A and M_B, respectively. We drop the values of the elements in EC-A that correspond to its successor's (i.e., from the 91st day to the 180th day in 2018) masking matrix M_B. If those elements are present in EC-A, their values are used as a ground truth to measure the performance of our predictions. If a time series is all observed in EC-B, then we drop this series in our datasets. In total, the dataset used in this section consists of around 20K time series. In this section, we do not randomly drop some non-missing elements to generate ground truth, as real-world missing patterns are not random. Figure 8 shows an example of how to drop missing elements.

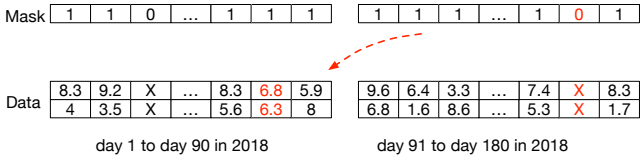


Figure 8: An illustration of ground truth generation on the EC dataset to simulate real-world missing data.

Table 3 reports the comparison between the levels of performance of the different models on this dataset. Our model achieves state-of-the-art performance on this dataset and its improvement on the performance of other baseline methods is significant. This finding demonstrates that our model can also achieve a good performance in the real-world situations in which the elements are not randomly missing. The performance of our model that does not consider social influence is lower than our complete model, but it still remains competitive concerning these benchmarks. Linear Interpolation is still a competitive method, and following are three neural imputation models. Another interesting finding is that while the performance of our model on this dataset is slightly better than that on the EC dataset, with a missing rate of 0.4, many baseline methods (e.g. MICE, Cubic) performs worse than they do under the same condition. This again demonstrates the adaptability and stability of our model under different missing conditions.

Table 3: Performance on the EC dataset with simulated missing data.The best results are in boldface, and the second-best results are underlined.

Method	MAE	RMSE	Method	MAE	RMSE
Mean	2.7626	4.1134	Median	2.8156	4.4493
Linear	1.7112	2.9973	Cubic	9.2609	67.5511
KNN	2.5144	3.9050	SoftImpute	2.5384	3.9342
MICE	2.8304	4.3208	MissForest	3.2628	4.9611
VAE	1.7067	3.0243	LSTM-Impute	2.4445	3.8235
GRU-D	1.9298	3.3543	STI - social	<u>1.6223</u>	<u>2.6731</u>
STI	**1.5837**	**2.6412**			

5 RELATED WORK

Time Series Modeling. For modeling time series data, plenty of parametric methods are proposed, including the auto-regressive models [4, 39], the kernel models [25], and the hidden Markov models [46, 49]. Recently, many time series model based on neural networks have been explored [21, 31]. More recent works explore to use deep learning methods to extract high-level representation of time series data [43, 44].

Traditional imputation methods for time series data. Over the years, quite a few imputation methods for time series data have been explored. The majority of them have concentrated on reconstructing missing elements by utilizing the temporal relationship within each data stream, such as the EM algorithm [18],the MissForest, [40] and the kernel method[35]. Multiple imputation [11] was also developed with these imputation methods to reduce uncertainty, by repeating the imputation procedure several times and giving the averaged estimates. However, running multiple models to fit the measured data during imputation introduces a great deal computational cost, which is a practical concern.

Imputation methods based on recurrent neural networks. Recently, recurrent neural networks (RNNs) have achieved the state-of-the-art performance in many tasks with sequential data, including machine translation and speech recognition. RNNs generate a sequence of hidden states h_t , as a function of the previous hidden states h_{t-1} and the input for position t. RNNs with LSTM or GRU units enjoy several benefits like capturing long-term sequential dependencies and supporting variable-length data input. The RNNs for missing data have also been studied in many works and have

been applied in speech recognition [2] and mortality prediction [13]. Previous works in this area [8, 41] have tried to first replace the missing elements with fixed values (such as the mean), most recent observations or values learned by other imputation algorithms, and then use the RNNs to update the initial data. However, the input sequences with pre-imputed values might have a different data distribution from the original ones, which can influence final performance when applying the RNNs to fit the data with pre-imputed values.

More recent advances take time irregularity into account. In other words, these papers model observations and intervals at the same time [13, 15, 50]. By concatenating information of observations and time gaps, they use these concatenation as the input for their customized RNNs. Different approaches of concatenation have been tried in these works. For instance, Che et al. [13] concatenate the latest observations, mean values, and time intervals as inputs to fit the customized GRU network. However, these works lack the insight of the temporal context of the whole time series and do not consider incorporating temporal and social patterns .

Recurrent neural networks with irregular intervals. Most of the proposed studies on RNNs assume that the interval between two consecutive timestamps is a constant value. However, it may not hold true in many real-world scenarios [7, 52]. For addressing this issue, many studies consider modifying gate structures of the LSTM cell [7, 24, 34]. Some studies also explore to add new gates to better model time gaps [32, 52]. In particular, Neil et al. [32] introduce a new gate, named time gate, and equip it with LSTM to model time intervals between two successive user behavior. Besides, Baytas et al. [7] adjust the memory cell of LSTM by discounting the short-term memory, and Beutel et al. [9] incorporate time gaps in the RNN by performing an element-wise product of model's hidden states with the embedding of time gaps.

Sequence modeling with attention mechanism. Learning sequence alignments via attention is also relevant to our work. Recently, sequence-to-sequence models with attention mechanism have achieved remarkable successes in many tasks, such as machine translation [10, 28], speech recognition [6], and image caption generation [27].This success has motivated us to extend this mechanism to the time series imputation field. Attention mechanisms can be viewed as feature extractors that allow the modeling of dependencies without regard to their distance in the input or output sequences [5]. The most popular approaches first build an encoder-decoder architecture consisting of two RNNs and use an attention mechanism to align the target to source tokens [5]. Recently, research in this field has focused on reducing the computational cost. Luong et al. [28] introduce various attention mechanisms that are computationally simpler and perform just as well as the model in [28]. Kalchbrenner et al. [23] propose a linear time architecture based on stacked convolutional neural networks. Gehring et al. [20] also propose the use of convolutional encoders to speed up neural machine translation. de Brébisson and Vincent [17] explore a linear attention mechanism using covariance matrices for information retrieval. Britz et al. [10] propose an end-to-end memory network based on a recurrent attention mechanism instead of sequence-aligned recurrence, and this method has shown good performance on several language modeling tasks, and is used in our model. Vaswani et al. [42] try to increase the capability of parallelism of the attention mechanism by using pure attention networks rather than recurrent structures. Location-based attention [48] has also been studied in the field of image recognition.

Time series in social networks. Our work is also relevant to studies on the evolution of user behavior in social networks [3, 12, 16, 30]. Using data from social networks, these studies explore the temporal connection between individual behavior and network properties. For instance, Asur and Huberman [3] use the time series of users' tweets about movies to predict movies' revenue. McAuley and Leskovec [30] study the process of user expertise on review websites, and find that modeling users' experience helps to discover when users acquire product in rating systems. Many studies examine the spread of behaviors or health conditions based on temporal data collected in networks [12, 16]. In particular, Christakis and Fowler [16] examine the spread of obesity on a large social network of 12K people over 32 years, and find that social distance tends to be more important than geographic distance with people's networks.

6 CONCLUSION AND FUTURE WORK

To what extent can your friends disclose your information? In this paper, we proposed a novel social-aware time series imputation framework, that considers social influence and temporal influence to infer missing values in time series. To incorporate these two factors, we designed two attention mechanisms in our model. Experimental results on real-world datasets show that our model can consistently outperform 11 baseline methods.

There are at least two kinds of perspectives. From the social networks side, in this paper we only exploited a small part of contextual information in social networks, we would like to utilize other useful patterns such as network structures and user interactions in the future work. From the practical side, we would like to explore the use of our approach into more real-world scenes.

Acknowledgements. The work is supported by NSFC (61702447), the Fundamental Research Funds for the Central Universities, and a research funding from the State Grid of China.

REFERENCES

[1] Sahil Agarwal, Sanket Khade, Yogesh Dandawate, and Prasad Khandekar. 2015. Three dimensional image reconstruction using interpolation of distance and image registration. In *IC4*. IEEE, 1–5.
[2] Abdul Manan Ahmad, Saliza Ismail, and DF Samaon. 2004. Recurrent neural network with backpropagation through time for speech recognition. In *ISCIT*, Vol. 1. IEEE, 98–102.
[3] Sitaram Asur and Bernardo A Huberman. 2010. Predicting the future with social media. In *WI/IAT*. IEEE Computer Society, 492–499.
[4] Anthony Bagnall and Gareth Janacek. 2014. A Run Length Transformation for Discriminating Between Auto Regressive Time Series. *Journal of Classification* 31, 2 (2014), 154–178.
[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).
[6] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. 2016. End-to-end attention-based large vocabulary speech recognition. In *ICASSP*. IEEE, 4945–4949.
[7] Inci M Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K Jain, and Jiayu Zhou. 2017. Patient subtyping via time-aware LSTM networks. In *SIGKDD*. 65–74.
[8] Yoshua Bengio and Francois Gingras. 1996. Recurrent neural networks for missing or asynchronous data. In *Advances in neural information processing systems*. 395–401.

[9] Alex Beutel, Paul Covington, Sagar Jain, Can Xu, Jia Li, Vince Gatto, and Ed H Chi. 2018. Latent Cross: Making Use of Context in Recurrent Recommender Systems. In *WSDM*. ACM, 46–54.

[10] Denny Britz, Melody Guan, and Minh-Thang Luong. 2017. Efficient Attention using a Fixed-Size Memory Representation. In *EMNLP*. 392–400.

[11] S van Buuren and Karin Groothuis-Oudshoorn. 2010. mice: Multivariate imputation by chained equations in R. *Journal of statistical software* (2010), 1–68.

[12] Damon Centola. 2010. The spread of behavior in an online social network experiment. *science* 329, 5996 (2010), 1194–1197.

[13] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific reports* 8, 1 (2018), 6085.

[14] Xi Chen, Jefrey Lijffijt, and Tijl De Bie. 2018. Quantifying and minimizing risk of conflict in social networks. In *SIGKDD*. ACM, 1197–1205.

[15] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*. 301–318.

[16] Nicholas A Christakis and James H Fowler. 2007. The spread of obesity in a large social network over 32 years. *New England Journal of Medicine* 357, 4 (2007), 370–379.

[17] Alexandre de Brébisson and Pascal Vincent. 2016. A Cheap Linear Attention Mechanism with Fast Lookups and Fixed-Size Representations. *arXiv preprint arXiv:1609.05866* (2016).

[18] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)* (1977), 1–38.

[19] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. 2016. Recurrent Marked Temporal Point Processes:Embedding Event History to Vector. In *SIGKDD*. 1555–1564.

[20] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122* (2017).

[21] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[22] Vijay Manikandan Janakiraman, Bryan Matthews, and Nikunj Oza. 2017. Finding Precursors to Anomalous Drop in Airspeed During a Flight's Takeoff. In *SIGKDD*. ACM, 1843–1852.

[23] Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. 2016. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099* (2016).

[24] Dejiang Kong and Fei Wu. 2018. HST-LSTM: A Hierarchical Spatial-Temporal Long-Short Term Memory Network for Location Prediction.. In *IJCAI*. 2341–2347.

[25] T. Kurashima, T. Althoff, and J. Leskovec. 2018. Modeling Interdependent and Periodic Real-World Action Sequences. (2018).

[26] Zachary C Lipton, David Kale, and Randall Wetzel. 2016. Directly modeling missing data in sequences with rnns: Improved classification of clinical time series. In *Machine Learning for Healthcare Conference*. 253–270.

[27] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *CVPR*, Vol. 6. 2.

[28] Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *EMNLP*. 1412–1421.

[29] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. 2010. Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research* 11, Aug (2010), 2287–2322.

[30] Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *WWW*. ACM, 897–908.

[31] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *ISCA*.

[32] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. 2016. Phased lstm: Accelerating recurrent network training for long or event-based sequences. In *Advances in Neural Information Processing Systems*. 3882–3890.

[33] Tim Op De Beéck, Wannes Meert, Kurt Schütte, Benedicte Vanwanseele, and Jesse Davis. 2018. Fatigue Prediction in Outdoor Runners Via Machine Learning and Sensor Fusion. In *SIGKDD*. ACM, 606–615.

[34] Trang Pham, Truyen Tran, Dinh Phung, and Svetha Venkatesh. 2016. Deepcare: A deep dynamic memory model for predictive medicine. In *PAKDD*. Springer, 30–41.

[35] Kira Rehfeld, Norbert Marwan, Jobst Heitzig, and Jürgen Kurths. 2011. Comparison of correlation analysis techniques for irregularly sampled time series. *Nonlinear Processes in Geophysics* 18, 3 (2011), 389–404.

[36] Christian H Reinsch. 1967. Smoothing by spline functions. *Numer. Math.* 10, 3 (1967), 177–183.

[37] Donald B Rubin. 2004. *Multiple imputation for nonresponse in surveys*. Vol. 81. John Wiley & Sons.

[38] Abraham Savitzky and Marcel JE Golay. 1964. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry* 36, 8 (1964), 1627–1639.

[39] Mohammad Shokoohi-Yekta, Yanping Chen, Bilson Campana, Bing Hu, Jesin Zakaria, and Eamonn Keogh. 2015. Discovery of Meaningful Rules in Time Series. In *SIGKDD*. 1085–1094.

[40] Daniel J Stekhoven and Peter Bühlmann. 2011. MissForest: non-parametric missing value imputation for mixed-type data. *Bioinformatics* 28, 1 (2011), 112–118.

[41] Volker Tresp and Thomas Briegel. 1998. A solution for missing data in recurrent neural networks with an application to blood glucose prediction. In *Advances in Neural Information Processing Systems*. 971–977.

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*. 5998–6008.

[43] Jingyuan Wang, Ze Wang, Jianfeng Li, and Junjie Wu. 2018. Multilevel Wavelet Decomposition Network for Interpretable Time Series Analysis. (2018), 2437–2446.

[44] Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and Philip S Yu. 2018. PredRNN++: Towards A Resolution of the Deep-in-Time Dilemma in Spatiotemporal Predictive Learning. (2018).

[45] Norbert Wiener, Norbert Wiener, Cyberneticist Mathematician, Norbert Wiener, Norbert Wiener, and Cybernéticien Mathématicien. 1949. Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications. (1949).

[46] Tao Wu and David F Gleich. 2017. Retrospective Higher-Order Markov Processes for User Trails. In *SIGKDD*. 1185–1194.

[47] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, and Yang Feng. 2018. Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications. (2018).

[48] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*. 2048–2057.

[49] Yun Yang and Jianmin Jiang. 2014. HMM-based hybrid meta-clustering ensemble for temporal data. *Knowledge-Based Systems* 56, C (2014), 299–310.

[50] Jinsung Yoon, William R Zame, and Mihaela van der Schaar. 2017. Multi-directional Recurrent Neural Networks: A Novel Method for Estimating Missing Data. (2017).

[51] Chen Zhang, Yijun Wang, Can Chen, Changying Du, Hongzhi Yin, and Hao Wang. 2018. StockAssIstant: A Stock AI Assistant for Reliability Modeling of Stock Comments. In *SIGKDD*. ACM, 2710–2719.

[52] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to do next: Modeling user behaviors by time-lstm. In *IJCAI*. 3602–3608.

Autonomous Learning for Face Recognition in the Wild via Ambient Wireless Cues

Chris Xiaoxuan Lu*
University of Oxford
xiaoxuan.lu@cs.ox.ac.uk

Xuan Kan
Tongji University
kanxuan1996@gmail.com

Bowen Du
University of Warwick
B.Du@warwick.ac.uk

Changhao Chen
University of Oxford
changhao.chen@cs.ox.ac.uk

Hongkai Wen
University of Warwick
hongkai.wen@dcs.warwick.ac.uk

Andrew Markham
University of Oxford
andrew.markham@cs.ox.ac.uk

Niki Trigoni
University of Oxford
niki.trigoni@cs.ox.ac.uk

John A. Stankovic
University of Virginia
jas9f@virginia.edu

ABSTRACT

Facial recognition is a key enabling component for emerging Internet of Things (IoT) services such as smart homes or responsive offices. Through the use of deep neural networks, facial recognition has achieved excellent performance. However, this is only possibly when trained with hundreds of images of each user in different viewing and lighting conditions. Clearly, this level of effort in enrolment and labelling is impossible for wide-spread deployment and adoption. Inspired by the fact that most people carry smart wireless devices with them, e.g. smartphones, we propose to use this wireless identifier as a supervisory label. This allows us to curate a dataset of facial images that are unique to a certain domain e.g. a set of people in a particular office. This custom corpus can then be used to finetune existing pre-trained models e.g. FaceNet. However, due to the vagaries of wireless propagation in buildings, the supervisory labels are noisy and weak. We propose a novel technique, `AutoTune`, which learns and refines the association between a face and wireless identifier over time, by increasing the inter-cluster separation and minimizing the intra-cluster distance. Through extensive experiments with multiple users on two sites, we demonstrate the ability of `AutoTune` to design an environment-specific, continually evolving facial recognition system with entirely no user effort.

CCS CONCEPTS

• **Human-centered computing** → *Ubiquitous and mobile computing*; • **Information systems** → *Information retrieval*; • **Computing methodologies** → *Lifelong machine learning*.

KEYWORDS

Adaptation of Learning Systems; Cross-modality Association; Face Recognition; Internet of Things

*Both authors contributed equally to the paper.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313398

ACM Reference Format:
Chris Xiaoxuan Lu, Xuan Kan, Bowen Du, Changhao Chen, Hongkai Wen, Andrew Markham, Niki Trigoni, and John A. Stankovic. 2019. Autonomous Learning for Face Recognition in the Wild via Ambient Wireless Cues. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, Article 0, 12 pages. https://doi.org/10.1145/3308558.3313398

1 INTRODUCTION

Facial recognition and verification are key components of smart spaces, e.g., offices and buildings for determining who is where. Knowing this information allows a building management system to tailor ambient conditions to particular users, perform automated security (e.g., opening doors for the correct users without the need for a swipe card), and customize smart services (e.g., coffee dispensing). A vast amount of research over the past decades has gone into designing tailored systems for facial recognition and with the advent of deep learning, progress has accelerated. As an example of a state-of-the-art face recognizer, FaceNet achieves extremely high accuracies (e.g., 99.5%) on very challenging datasets through the use of a low dimensional embedding, allowing similar faces to be clustered through their Euclidean distance [14, 26]. However, when directed transferred to operate in 'in the wild', subject to variable lighting conditions, viewing angle and appearance changes, performance of off-the-shelf pre-trained classifiers degrades significantly, with accuracies around 15% not being uncommon. The solution to this is to obtain a large, labelled corpus of data for a particular environment, with hundreds of annotated images per user. Given access to such a hypothetical dataset, it is then possible to fine-tune the pre-trained classifier on out-domain data to adapt to the new environment and achieve excellent performance.

However, the cost of labelling and updating the corpus (e.g. to enrol new users) is prohibitive for most critical applications and therefore, will naturally limit the use and uptake of facial recognition as a ubiquitous technology in emerging Internet of Things (IoT) applications. On the other side, people often, but not always, carry smart devices (phones, fitness devices etc). Wang et al. [36] advocated that although these devices do not provide fine-grained enough positioning capability to act as a proxy for presence, they can be used to indicate that a user *might* be present in an area

with a co-located camera. In this work, we take a step forward and further utilize device presence as weak supervision signals for the purposes of fine-tuning a classifier. The goal now becomes how to take an arbitrary, pre-trained recognition network and tune it from a generic classifier to a highly specific classifier, optimized for a certain environment and group of people. We note that the aim is to make the network better and better at this specific goal, but it would likely perform poorly if transferred directly to a different environment. This is the antithesis of the conventional view of generalized machine learning, but is ideally suited for the problem of environment specific facial recognition, as opposed to generic facial recognition.

The technical challenge is that there is not a 1:1 mapping between a face and a wireless identity, rather we need to solve the association between a set of faces and a set of identities over many sessions or occasions. To further complicate the problem, the sets are not pure i.e. the set of faces can contain additional faces from people not of interest (e.g. visitors). Equally well, due to the vagaries of wireless transmission, the set of wireless identifiers will contain additional identifiers e.g. from people in the next office. Furthermore, it is also possible to have missing observations e.g. because a person was not facing the camera or because someone left their phone at home.

In this work, we present AutoTune, a system which can be used to gradually improve the performance of facial recognition systems in the wild, with zero user effort, tailoring them to the visual specifics of a particular smart space. We demonstrate state-of-the-art performance in real-world facial recognition through a number of experiments and trials.

In particular, our contributions are:

- We observe and prove that wireless signals of users' devices provide valuable, albeit noisy, clues for face recognition. Namely, wireless signals can serve as a weak label. Such weak labels can replace the human annotated face images in the wild to save intensive effort.
- We create AutoTune, a novel pipeline to simultaneously label face images in the wild and adapt the pre-trained deep neural network to recognize the faces of users in new environments. The key idea is to repeat the face-identity association and network update in tandem. To cope with observation noise, we propose a novel probabilistic framework in AutoTune and design a new stochastic center loss to enhance the robustness of network fine-tuning.
- We deployed AutoTune in two real-world environments and experimental results demonstrate that AutoTune is able to achieve > 0.85 F_1 score of image labeling in both environments, outperforming the best competing approach by $> 25\%$. Compared to the best competing approach, using the features extracted from the fine-tuned model and training a classifier based on the cross-modality labeled images can give a $\sim 19\%$ performance gain for online face recognition.

The rest of this paper is organized as follows. §2 introduces the background of this work. System overview is given in §3. We describe the AutoTune solution in §4 and §5. System implementation details are given in §6. The proposed approach is evaluated and compared with state of the art methods in §7. Finally, we discuss and outlook future work in §8 and conclude in §9.

2 RELATED WORK

Deep face recognition: Face recognition is arguably one of the most active research areas in the past few years, with a vast corpus of face verification and recognition work [23, 31, 40]. With the advent of deep learning, progress has accelerated significantly. Here we briefly overview state-of-the art work in Deep Face Recognition (DFR). Taigman et al. pioneered this research area and proposed DeepFace [30]. It uses CNNs supervised by softmax loss, which essentially solves a multi-class classification problem. When introduced, DeepFace achieved the best performance on the Labeled Face in the Wild (LFW) [9] benchmark. Since then, many DFR systems have been proposed. In a series of papers [27, 28] Sun et al. extended on DeepFace incrementally and steadily increased the recognition performance. A critical point in DFR happened in 2015, when researchers from Google [26] used a massive dataset of 200 million face identities and 800 million image face pairs to train a CNN called Facenet, which largely outperformed prior art on the LFW benchmark when introduced. A point of difference is in their use of a "triplet-based" loss [5], that guides the network to learn both inter-class dispersion and inner-class compactness. Recently proposed RTFace [36] not only achieves high recognition accuracy but also operates at the full frame rates of videos. Although the above methods have proven remarkably effective in face recognition, the training needs a vast amount of labeled images to train the supervised DFR network. A large amount of labeled data is not always achievable in a particular domain, and using a small amount of training data will incur poor generalization ability in the wild.

Cross-modality Matching: Cross-modal matching has received considerable attention in different research areas. Methods have been developed to establish mappings from images [7, 11, 34] and videos [33] to textual descriptions (e.g., captioning), developing image representation from sounds [20, 21], recognizing speaker identities from Google calendar information [17], and generating visual models from text [41]. In cross-modality matching between images and radio signals, however, work is very limited and all dedicated to trajectory tracking of humans [1, 22, 32]. The field of face recognition via wireless signals is an unexplored area.

3 AUTOTUNE OVERVIEW

3.1 System Model

We consider a face recognition problem with m people of interest (POI) and each subject owns one WiFi-enabled device, e.g., a smartphone. We denote the identity set as $\mathcal{Y} = \{y_j | j = 1, 2, \ldots, m\}$. The set of observed POI's devices in a particular environment is denoted by $\mathcal{L} = \{l_j | j = 1, 2, \ldots, m\}$, e.g., a set of MAC addresses. We assume the mapping from device MAC addresses $\mathcal{L}$ to the user identity $\mathcal{I}$ is known. A collection of face images $\mathcal{X} = \{x_j | j = 1, 2, \ldots, n\}$ is cropped from the surveillance videos in the same environment. Note that, to mimic the real-world complexity, our collection includes the faces of both POI and some non-POI, e.g., subjects with unknown device MAC addresses. We then assign face and device observations to different subsets based on their belonging events $\mathcal{E} = \{e_j | j = 1, 2, \ldots, h\}$. An event e_j is the setting in which people interact with each other in a specific part of the environment for a given time interval. It is uniquely identified by three attributes: effective timeslot, location, and participants. Fig. 1 demonstrates a

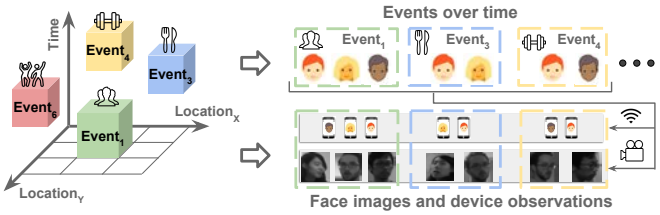


Figure 1: An illustrative example of *events* and showing their linked *observations*. An *event* is uniquely determined by spatial-temporal attributes and its participants. An event is also linked with two types of observations. Face observation are images cropped from surveillance videos and device observation are sniffed MAC address of participants' devices.

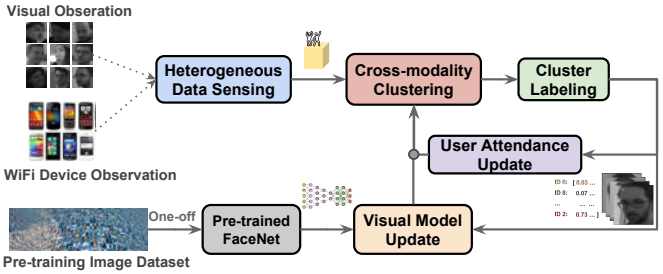


Figure 2: Workflow of AutoTune. AutoTune consists of 5 steps i) Heterogeneous Data Sensing ii) Cross-Modality Clustering iii) Cluster Labeling iv) Visual Model Update v) User Attendance Update. AutoTune sequentially repeats the above five steps until the changes in the user attendance are negligible.

few examples of events. Lastly, we also have a deep face representation model f_θ pre-trained on public datasets that contains no POI. Such model is trained with metric losses, e.g., triplet loss so that the learned features could bear a good property for clustering [26].

In this sense, the problem addressed by AutoTune is assigning IDs to images from noisy observations of images and WiFi MAC addresses, and using such learned ID-image associations to tune the pre-trained deep face representation model automatically.

3.2 System Architecture

AutoTune is based on two key observations: i) although collected by different modalities, both face images and device MAC addresses are linked with the identities of users who attend certain events; and ii) the tasks of model tuning and face-identity association should not be dealt with separately, but rather progress in tandem. Based on the above insights, AutoTune works as follows (see Fig 2):

- *Heterogeneous Data Sensing.* This module collects facial and WiFi data (device attendance observations) through surveillance cameras and WiFi sniffers[1] in a target environment. Given the face images and sniffed WiFi MAC addresses, AutoTune first segments them into events based on the time and location they were captured.
- *Cross-modality Labeling.* This module first clusters face images based on their appearance similarity computed by the face representation model, and also taking into account information on device attendance in events. Each image cluster should broadly correspond to a user, and the cluster's images are drawn from a set of events. We then assign each cluster to the user whose device has been detected in as similar as possible set of events.
- *Model Updates.* Once images are labeled with user identity labels, this module then fine-tunes the pre-trained face representation model. AutoTune further uses the cluster labels to update our belief on which device (MAC address) has participated in each event.

The sensing module is one-off and we will detail its implementation in §6.1. Labeling and model update modules are iteratively repeated until the changes in the user attendance model become negligible. The tuned model derived in the last iteration is regarded as the one best adapted to POI recognition in the new environment.

[1] https://www.wireshark.org/

4 CROSS-MODALITY LABELING

In this section, we introduce the labeling module in AutoTune. The challenge in this module is that collected facial images and sniffed WiFi data are temporally unaligned. For example, detecting a device WiFi address does not imply that the device owner will be captured by the camera at the exact instant and vice versa. Such mismatches in cross-modality data distinguish our problem from prior sensor fusion problems, where both multiple sensors are observing a temporal evolving system. In order to tackle the above challenge, we leverage the diverse attendance patterns in events and use a two-step procedure in isolated. Images $\mathcal{X}$ are firstly grouped together into clusters across all sessions, and we then associate clusters with device IDs (i.e., labels) $\mathcal{L}$ based on their similarity in terms of event attendance.

4.1 Cross-modality Clustering

Heterogeneous Features. Given a pre-trained face representation model f_θ, an image x_i can be translated to the feature vector $\mathbf{z}_i$. Unlike conventional clustering that merely depends on the face feature similarity, AutoTune merges face images *across events* into a cluster (potentially belonging to the same subject) by incorporating attendance information as well. Recall that device attendance already reveals the identities of subjects (in the form of MAC addresses) in a particular event, and the captured images in the same event may contain the faces of these subjects as well. Despite the noise in observations, the overlapped subjects in different events can be employed as a prior that guides the image clustering. For example, if there are no shared MAC addresses sniffed in two events, then it is very likely that the face images captured in these two events should lie in different clusters. Formally, for an event e_i, we denote a device attendance vector as $\mathbf{u}_k = (u_k^1, u_k^2, \ldots, u_k^m)$, where $u_k^j = 1$ if device l_j is detected in event e_k. In this way, we could construct a heterogeneous feature $\widetilde{\mathbf{z}}_i = [\mathbf{z}_i, \mathbf{u}_k]$ for an image x_i collected in the event e_k. Note that, as all images captured in a same event have the same attendance vector, this device attendance part are essentially enforced on cross-event image comparison.

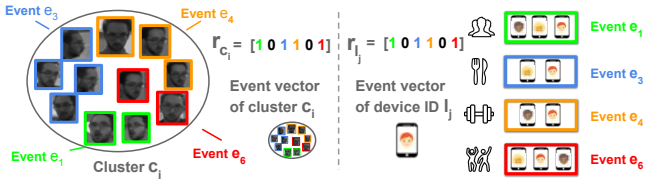


Figure 3: Event vectors of an image cluster c_i and device ID l_j. The kth element of the vector $r^k_{c_i}$ is set to 1 only if it contains images attached with the event e_k. The event vector $\mathbf{r}_{l_j}$ of the device ID l_j is similarly developed. This insight leads to Eq. 1.

Face Image Similarity. Given an image x_i captured in event e_k (i.e., $\widetilde{\mathbf{z}}_i = [\mathbf{z}_i, \mathbf{u}_k]$) and an image x_j captured in event e_p (i.e., $\widetilde{\mathbf{z}}_j = [\mathbf{z}_j, \mathbf{u}_p]$), the likelihood that two cross-event face images belong to the same subject is conditioned on two factors: i) the similarity of their feature representation between $\mathbf{z}_i$ and $\mathbf{z}_j$; and ii) the overlap ratio between the attendance observations in their corresponding events $\mathbf{u}_k$ and $\mathbf{u}_p$ respectively. The resulted joint similarity is a log likelihood $log(\mathbf{Pr}(x_i = x_j))$ defined as follows:

$$\mathbf{Pr}(x_i = x_j) \propto \frac{exp(\beta * |\mathbf{u}_k \otimes \mathbf{u}_p|}{exp(\beta * |\mathbf{u}_k \oplus \mathbf{u}_p|)} * exp(-D(\mathbf{z}_i, \mathbf{z}_j))$$

$$log(\mathbf{Pr}(x_i = x_j)) \propto \beta * \underbrace{(|\mathbf{u}_k \otimes \mathbf{u}_p| - |\mathbf{u}_k \oplus \mathbf{u}_p|)}_{\text{attendance similarity}} - \underbrace{D(\mathbf{z}_i, \mathbf{z}_j)}_{\text{feature distance}}$$

Here $\otimes$ and $\oplus$ are element-wise AND and OR, and $|\cdot|$ here is the L^1-norm. z is the features transformed by the face representation model and D is a distance measure between face features. β, analogous to the regularization parameter in composite loss functions, is a hyper-parameter that controls the contributions of the attendance assistance and feature similarity. The above derivation is inspired by the Jaccard coefficients, with the difference lying in the log function. The rationale behind the term $|\mathbf{u}_k \oplus \mathbf{u}_p|$ is that the more different subjects attending events, the more uncertain that any two images drawn across these events will point to the same subject. In contrast, when the intersection $|\mathbf{u}_k \otimes \mathbf{u}_p|$ is significant enough, the chance that these two images point to the same subject will become higher. This joint similarity can also be explained from a Bayesian perspective. The attendance similarity of two events can serve as a prior that two cross-event images belong to the same subject and the feature similarity can be seen as the likelihood. Together they determine the posterior probability that two cross-event images fall into the same cluster. Based on the above joint similarity, images across events are grouped into clusters $C = \{c_i | i = 1, 2, \ldots, g\}$. We will soon discuss how to determine the number of clusters g based on the complete set of MAC addresses $\mathcal{L}$ in the next section.

4.2 Cluster Labeling

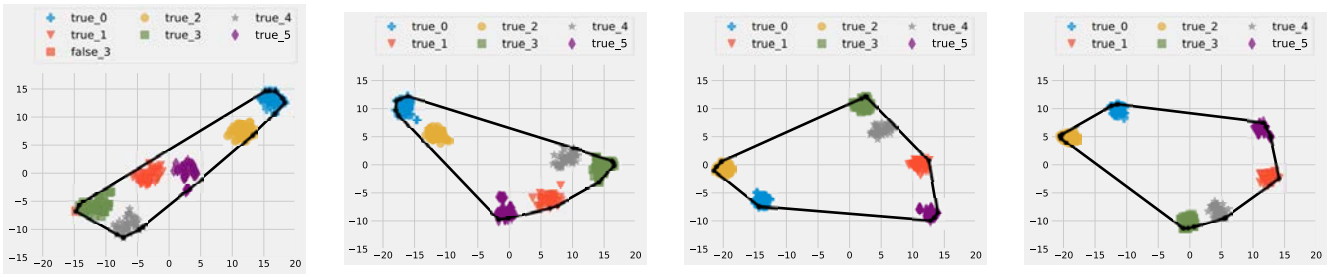
Fine-grained ID Association. After clustering, an image cluster is linked with multiple events that are associated with its member images. Naturally, we introduce an event vector $\mathbf{r}_{c_i} = (r^1_{c_i}, r^2_{c_i}, \ldots, r^h_{c_i})$ for for an image cluster c_i, where h is the total number of events. $r^k_{c_i}$ is set to 1 only if c_i contains images from event e_k. Fig. 3 shows an example of how an event vector is developed. Similarly, for a device ID l_j, its corresponding event vector $\mathbf{r}_{l_j} = (r^1_{l_j}, r^2_{l_j}, \ldots, r^h_{l_j})$ can be determined by inspecting its occurrences in all WiFi sniffing observations. $r^k_{l_j}$ is set to 1 only if the device ID (MAC address) l_j is detected in the event e_k. The intuition behind ID association is that a device and a face image cluster of the same subject should share the most consistent attendance pattern in events, reflected by the similarity of their event vectors. Based on this intuition, AutoTune assigns clusters with device IDs based on the matching level of their event vectors. Formally, a matching problem of bipartite graph can be formulated as follows:

$$\mathcal{L}_A = \sum_{ij} a_{ij} (\mathbf{r}_{c_i} - \mathbf{r}_{l_j})^2$$
$$\sum_{1 \le j \le m} a_{ij} = 1, \quad \forall i \in \{1, \ldots, g\} \tag{1}$$

where the solution of the binary variable a_{ij} assigns a device ID to a cluster. We note that when $m < g$, AutoTune adds dummy nodes to create the complete bipartite graph. Then the complete bipartite graph problem can be solved by the Hungarian algorithm [10].

Probabilistic Labeling via Soft Voting. We now obtain the association between face images and device IDs. However, in practice, the number of clusters g can vary due to the captured face images outside the POI, e.g., face images of short-term visitors. The choice of g has a significant impact on the performance of clustering [6] which could further affect the following fine-tuning performance. To cope with this issue, AutoTune sets the number of clusters g greater than the number of POI m. With a larger number of clusters, although there are some unused clusters of non-POI after clustering, the association step will sieve them and only choose the most consistent m clusters with the event logs of m device IDs. Lastly, for every image, its assigned ID is finalized by decision voting on the individual association results computed with different g, which will be soon introduced in the next paragraph. Typically, a majority voting procedure would give a hard ID label for an image and use these <Image, ID> pairs as training data to update the face representation model. However, due to the noise in device presence observations and errors in clustering, there will be mis-labeling in the training data that may confuse the model update. To account for the uncertainty of ID assignment, we adopt soft labels for voting. Then instead of voting for the most likely ID y_i for an image x_i, we introduce a probability vector $\mathbf{y}_i = (y_{i,1}, y_{i,2}, \ldots, y_{i,m})$. Specifically, every image is associated with all POI and its associated soft label is derived by the votes for the subjects divided by the total number of votes. In this way, the soft label of each image is a valid probability distribution that sums up to 1. For instance, $y_{i,j} = 0.4$ means that there are 40% associations assigning the ID j to the image i. Moreover, the soft labeled data $< x_i, \mathbf{y}_i >$ is compatible with the computation of cross-entropy.

Number of Clusters. We are now in a position to describe how to select the number of clusters g. We observed that, with a proper clustering algorithm (i.e., agglomerative clustering in our choice), non-POI's images will form separate small clusters but they will not be chosen in our follow-up association step. This is because that the number of devices ID m is always less than the number of clusters ($g \in [2 * m, 5 * m]$), and only those images clusters with the most consistent event attendances are associated. The small clusters of



(a) Pre-trained Model (b) 1st iteration (c) 2nd iteration (d) 3rd iteration

Figure 4: An example of visual model update (§5.1) in AutoTune with a six-subject subset. Note that the pre-trained model in Fig. 4a has a narrow convex hull (black lines), clusters are dispersed and there is mis-classification (see false_3 in Fig. 4a) as the pre-trained features are not discriminative enough to differentiate unseen subjects in a new environments. Over time, clusters become more compact and move further away from each other, leading to purer clustering results.

outlier images often have scatted attendance vector and will be ignored in association. Therefore, soft voting with varying number of clusters can reinforce the core images of POI by giving large probability and assigns non-POI's images with small probability.

5 MODEL UPDATES

We are now in a position to introduce the model update in AutoTune. At every iteration, AutoTune updates the last-iteration face representation model f_θ^τ to $f_\theta^{\tau+1}$, by taking the labeled images as inputs. To correct the device observation errors, AutoTune leverages these label images to update device attendance vector to $\mathbf{u}^\tau$ for all events.

5.1 Visual Model Update

Discriminative Face Representation Learning. Face representation learning optimizes a representation loss $\mathcal{L}_R$ to enforce the learnt features as discriminative as possible. Strong discrimination bears two properties: inter-class dispersion and intra-class compactness. Inter-class dispersion pushes face images of different subjects away from one another and the intra-class compactness pulls the face images of the same subject together. Both of the properties are critical to face recognition. At iteration τ, given the current labels y_i^τ for the ith face image x_i and the transformed features $\mathbf{z}_i^\tau = f_\theta^\tau(x_i)$, the representation loss $\mathcal{L}_R$ is determined by a composition of softmax loss and center loss:

$$\begin{aligned}\mathcal{L}_R &= \mathcal{L}_{softmax} + \mathcal{L}_{center} \\ &= \underbrace{\sum_i -log(\frac{e^{W_{y_i^\tau}^T \mathbf{z}_i^\tau + \mathbf{b}_{y_i^\tau}}}{\sum_{j=1}^m e^{W_{l_j}^T \mathbf{z}_i^\tau + \mathbf{b}_{l_j}}})}_{\text{softmax loss}} + \underbrace{\sum_i \frac{\lambda}{2}||\mathbf{z}_i^\tau - \mathbf{o}_{y_i^\tau}||^2}_{\text{center loss}} \end{aligned} \quad (2)$$

where W and b are the weights and bias parameters in the last fully connected layer of the pre-trained model. $\mathbf{o}_{y_i^\tau}$ denotes a centroid feature vectors by averaging all feature vectors with the same identity label y_i. The center loss $\mathcal{L}_{center}$ explicitly enhances the intra-class compactness while the inter-class dispersion is implicitly strengthened by the softmax loss $\mathcal{L}_{softmax}$ [37]. λ is a hyper-parameter that balances the above sub-losses.

Stochastic Center Loss The center loss $\mathcal{L}_{center}$ in Eq. 2 is shown to be helpful to enhance the intra-class compactness [37]. However, we cannot directly adopt it for fine-tuning as computing the centers requires explicit labels (see Eq. 2) of images, while the association steps above only provide probabilistic ones through soft labels. To solve this, we propose a new loss called *stochastic center loss* $\mathcal{L}_{stoc}$ to replace the center loss. Similar to the idea of fuzzy sets [38], we allow each face image to belong to more than one subject. The membership grades indicate the degree to which an image belongs to each subject and can be directly retrieved from the soft labels and the stochastic center $\mathbf{o}_k^\tau$ for the k-th identity is given as:

$$\mathbf{o}_k^\tau = \frac{\sum_i^n \mathbf{z}_i^\tau * y_{i,k}^\tau}{\sum_i^n y_{i,k}^\tau} \quad (3)$$

This gives the stochastic center loss as follows:

$$\mathcal{L}_{stoc} = \sum_i \sum_k y_{i,k}^\tau * ||\mathbf{z}_i^\tau - \mathbf{o}_k^\tau||^2 \quad (4)$$

We leave the softmax loss $\mathcal{L}_{softmax}$ the same as in Eq. 2, because the soft labels are compatible with the computation of cross-entropy. The new representation loss to minimize is:

$$\mathcal{L}_R = \mathcal{L}_{softmax} + \mathcal{L}_{stoc} \quad (5)$$

AutoTune updates the model parameters θ^τ to $\theta^{\tau+1}$ based on the gradients of $\nabla_\theta \mathcal{L}_R$, which are calculated via back prorogation of errors. Compared with the dataset used for pre-training, which is usually in the order of millions [4], the data used for fine-tuning is much smaller (several thousands). The mis-match between the small training data and the complex model architecture could result in overfitting. To prevent overfitting, we use the dropout mechanism in training, which is widely adopted to avoid overfitting [13]. Meanwhile, as observed in [8, 25], the soft label itself can play the role of a regularizer, and make the trained model robust to noise and reduce overfitting. Fig. 4 illustrates the effect of model update.

5.2 User Attendance Update

The device presence observations by WiFi sniffing are noisy because the WiFi signal of a device is opportunistic and people do not carry/use theirs devices all the time. Based on the results of the cluster labelling step introduced in §4.2, we have the opportunity to update our belief on which users attended each event. The update

Algorithm 1: `AutoTune`

Input: pre-trained model f_θ^1, images $\mathcal{X}$, POI's device IDs $\mathcal{L}$, threshold ξ
Output: adapted model f_θ^*, corrected attendance observations I^*, soft image labels Y^*
Initialize: Given sniffed $\mathcal{L}$ in all events $\mathcal{E}$, compute attendance vector $\mathbf{u}^0$

$\tau = 1$
while $\sqrt{\frac{1}{h}\sum_{k=1}^{h} ||\mathbf{u}_k^{(\tau)} - \mathbf{u}_k^{(\tau-1)}||^2} > \xi$ **do**
  $\mathcal{Z}^\tau = f_\theta^\tau(\mathcal{X})$ ▷ feature transformation
  for $g \leftarrow 2 * m$ *to* $5 * m$ **do**
    $C^\tau \leftarrow$ `cross_modality_clustering`$(\mathcal{Z}^\tau, \mathbf{u}^\tau, g)$ ▷ §4.1;
    $A_g^\tau \leftarrow$ `cluster_labeling`$(C^\tau, \mathbf{r}_c^\tau, \mathbf{r}_l^\tau)$ ▷ §4.2;
  end
  $\mathcal{Y}^\tau \leftarrow$ `soft_voting`(A^τ) ▷ §4.2;
  $\mathbf{o}^\tau \leftarrow$ `stochastic_center`$(\mathcal{Y}^\tau, \mathcal{Z}^\tau)$ ▷ §5.1;
  $f_\theta^{\tau+1} \leftarrow$ `visual_model_update`$(\mathcal{X}, \mathcal{Y}^\tau, \mathbf{o}^\tau, f_\theta^\tau)$ ▷ §5.1 ;
  $\mathbf{u}^{\tau+1} \leftarrow$ `user_attendance_update`$(\mathcal{Y}^\tau, \mathbf{u}^\tau)$ ▷ §5.2 ;
  $\tau \leftarrow \tau + 1$
end

mechanism is as follows: Each image is associated with a user probability vector, whose elements denote the probability that the image correspondences to a particular user. By averaging the user probability vectors of all images that have been drawn from the same event e_k, and normalizing the result, we can estimate the user attendance of this event. The elements of the resulting user attendance vector $\widehat{\mathbf{u}}_k^\tau$ denote the probabilities of different users attending event e_k.

We can now use $\widehat{\mathbf{u}}_k^\tau$ as a correction to update our previous wifi attendance vector $\mathbf{u}_k^\tau$ as follows:

$$\mathbf{u}_k^{\tau+1} = \mathbf{u}_k^\tau - \gamma \cdot (\mathbf{u}_k^\tau - \widehat{\mathbf{u}}_k^\tau) \tag{6}$$

where γ is a pre-defined parameter that controls the ID update rate. In principle, a large update rate will speed up the convergence rate, at the risk of missing the optima. `AutoTune` sequentially repeats the above steps of clustering, labelling and model updates, until the changes ξ in the user attendance model are negligible (≤ 0.01 in our case). Algorithm. 1 summarizes the workflow.

6 IMPLEMENTATION

In this section, we introduce the implementation details of `AutoTune` (code available at https://github.com/Wayfear/Autotune).

6.1 Heterogeneous Data Sensing

Face Extraction. This module consists of a front-end remote camera and a back-end computation server [2]. Specifically, the remote cameras in our experiment are diverse and include *GoPro Hero 4*[3], *Mi Smart Camera*[4] and *Raspberry Pi Camera*[5]). We modified these cameras so that they are able to communicate and transfer data to the back-end through a wireless network. To avoid capturing excess data without people in it, we consider a motion-triggered mechnism with a circular buffer. It works by continuously taking low-resolution images, and comparing them to one another for changes caused by something moving in the camera's field of view. When a change is detected, the camera takes a higher-resolution video for 5 seconds and reverts to low resolution capturing. All the collected videos are sent to the backend at every midnight. On the backend, a cascaded convolutional network based face detection module [39] is used to drop videos with no face in them. The cropped faces from the remaining videos are supplied to `AutoTune`.

WiFi Sniffing. This module is realized on a WiFi-enabled laptop running Ubuntu 14.04. Our sniffer uses Aircrack-ng [6] and tshark [7] to opportunistically capture the WiFi packets in the vicinity. The captured packet has unencrypted information such as transmission time, source MAC address and the Received Signal Strengths (RSS). As `AutoTune` aims to label face images for POI, our WiFi sniffer only records the packets containing MAC addresses of POI's and discards them otherwise, so as to not harvest addresses from people who have not given consent. A channel hop mechanism is used in the sniffing module to cope with cases where the POI's device(s) may connect to different WiFi networks, namely, on different wireless channels. The channel hop mechanism forces the sniffing channel to change by every second and monitor the active channels periodically (1 second) in the environment. The RSS value in the packet implies how far away the sniffed device is from the sniffer [18, 19]. By putting the sniffer near the camera, we can use a threshold to filter out those devices with low RSS values, e.g., less than -55 dBm in this work, as they are empirically unlikely to be within the camera's field of view.

Event Segmentation. Depending on the context, the duration of events can be variable. However, for simplicity we use fix-duration events in this work. Specifically, we split a day into 12 intervals, each of which is 2 hours long. We then discard those events that have neither face images nor sniffed MAC addresses of POI.

6.2 Face Recognition Model

The face recogonition model used in `AutoTune` is the state-of-the-art FaceNet [26].

Pre-training. FaceNet adopts the Inception-ResNet-v1 [29] as its backbone and its weights are pre-trained on the VGGFace2 dataset [4]. This dataset contains 3.31 million images of 9131 subjects, with an average of 362.6 images for each subject. Images are downloaded from Google Image Search and have large variations in pose, age, illumination, ethnicity and profession. Pre-training is supervised by the triplet loss [5] and the training protocols, e.g, parameter settings, can be found in [26]. We found that the learnt face representation by FaceNet is generalizable and it is able to achieve an accuracy of 99.65% on the LFW face verification task [8]. Note that, FaceNet [26] not only learns a powerful recognition model but gives a very discriminative feature representation that can be used for clustering.

[2] The study has received ethical approval R50950
[3] https://shop.gopro.com/cameras
[4] https://www.mi.com/us/mj-panorama-camera/
[5] https://www.raspberrypi.org/products/camera-module-v2/
[6] https://www.aircrack-ng.org/
[7] https://www.wireshark.org/docs/man-pages/tshark.html
[8] http://vis-www.cs.umass.edu/lfw/

Fine-tuning by AutoTune. The fine-tuning process has been explained in §5.1, and here we provide some key implementation detials. After each round of label association (see §4.2), the labeled data is split into a training set and validation set, with a ratio of $8:2$ respectively. The pre-trained FaceNet is then fine-tuned on the training set, and the model that achieves the best performance on the validation set is saved. Note that the fine-tuning process in AutoTune does not involve the test set. The online testing is performed on a held-out set that is collected on different days. To enhance the generalization ability, we use dropout training for regularization [35]. The dropout ratio is set to 0.2. We set the batch size to 50 and one fine-tuning takes 100 epochs.

6.3 System Configuration

Face Detection. As discussed in §6.1, we use a cascaded convolution network to detect faces in videos. It is cascaded by three sub-networks, a proposal network, a refine network and an output network. Each of them can output bounding boxes of potential faces and the corresponding detection probabilities, i.e., confidences. Face detection with small confidence will be discarded early in the process and not sent to the next sub-network. In this work, the confidence threshold is set to 0.7, 0.7 and 0.9 for three sub-networks respectively. Following the original setting in [39], we set the minimal face size in detection to 40×40 pixels.

Setup of Face Clustering. In §4.1, we use a clustering algorithm to merge the images across events. Specifically, the clustering algorithm used in AutoTune is agglomerative clustering [2], which is a method that recursively merges the pair of clusters that minimally increases a given linkage distance. The similarity metric adopted here is Euclidean distance. A linkage criterion determines which distance to use between sets of data points. In AutoTune, this linkage criterion is set to the average distances of all sample pairs from two sets. As introduced in Sec. 4.2, the number of clusters g is determined by the number of POI m. We vary g from $2 * m$ to $5 * m$ and proceed soft voting, to account for the extra clusters of non-POI's face images and ambiguous/outlier images of POI.

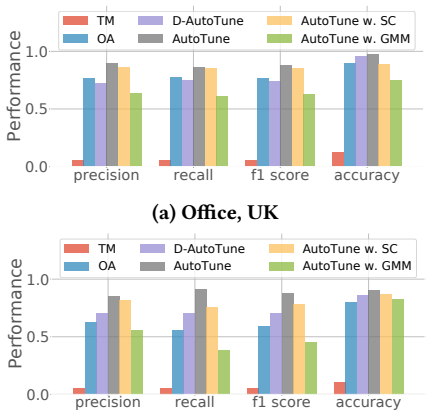
7 EVALUATION

In this section, we evaluate the AutoTune extensively on datasets collected from both real-world experiments and simulation. We deployed two testbeds, one in the UK and the other in China, and collected datasets as described in the previous section. The simulation dataset is developed based on a public face datasets.

7.1 Evaluation Protocols

Competing Approaches. We compare the performance of AutoTune with the 3 competing approaches:

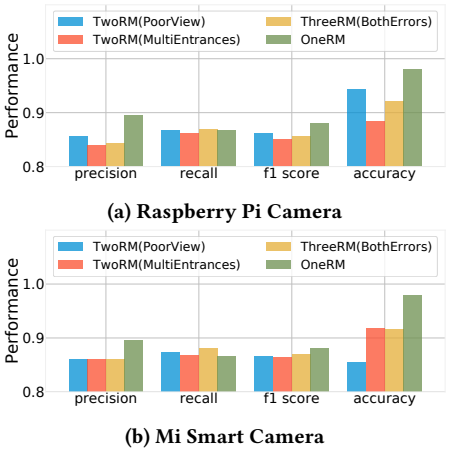
- **Template Matching (TM)** [3] employs a template matching method to assign ID labels to clusters of face images. This is the most straightforward method that is used when one or more profile photos of POI are available, e.g., crawled from their personal homepage or Facebook.
- **One-off Association (OA)** [16] uses one-off associations to directly label the image clusters without fine-tuning of the face representation model itself.



(a) Office, UK

(b) CommonRoom, CHN

Figure 5: Overall Labeling Performance.



(a) Raspberry Pi Camera

(b) Mi Smart Camera

Figure 6: Performance of Scalability.

- **Deterministic AutoTune (D-AutoTune)** is the deterministic version of AutoTune. In D-AutoTune, the association and update steps are the same as in AutoTune, but it adopts hard labels rather than soft labels and uses the simple center loss instead of the proposed stochastic center loss (see §5.1).

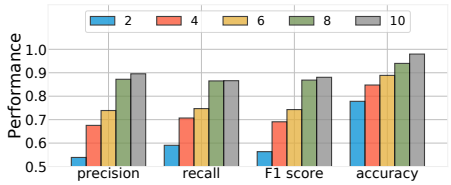
Evaluation Metrics. AutoTune contains two main components, offline label assignment and online inference. For the offline label assignment, we evaluate its performance with the following metrics: TP, TN, FP, FN are true positive, true negative, false positive and false negative respectively. Each metric captures different aspects of classification [24]. Online face recognition has two kinds of tests, face identification and verification. We follow [14, 15] to use Cumulative Match Characteristic (CMC) for the evaluation of online face identification.

7.2 Offline Cross-modality Face Labeling

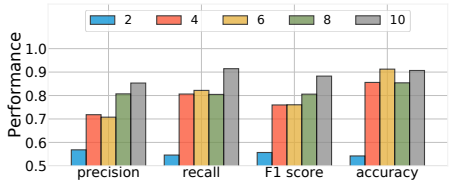
AutoTune automatically labels images captured in the wild by exploiting their correlations with the device presences. The quality of image labeling is crucial for the follow-up face recognition in

Table 1: Key Metrics of Collected Dataset Sets.

Site	NO. of Rooms	NO. of POI	NO. of non-POI	NO. of Images	NO. of Events	Event Duration	Average subjects/event	Camera(s) in rooms	Experiment Note
UK	3	24	11	15, 286	83	3h	9.14	Office: GoPro Hero 4 Kitchen: Pi Cam & Mi Cam Meeting: Pi Cam & Mi Cam	GoPro: 1080p, 90FPS PiCam: 720p, 90FPS MiCam: 720p, 15FPS
CHN	1	12	25	7, 495	102	2h	3.36	CommonRM: Gopro Hero 4	All Faces are from Asian



(a) UK



(b) CHN

Figure 7: Impact of Number of Days.

the online stage. In this section, we investigate the image labeling performance of AutoTune.

7.2.1 Data Collection. We deployed AutoTune at the testbeds in two countries, with different challenging aspects.

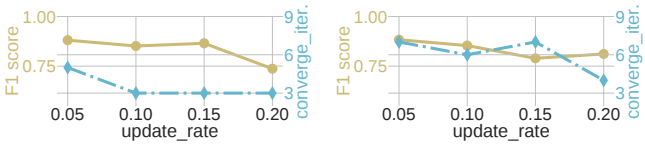
UK Site: This first dataset is collected in a commercial building in the UK. We deploy the heterogeneous sensing front-ends, including surveillance cameras and WiFi sniffers, on a floor with three different types of rooms: office, meeting room and kitchen. 24 long-term occupants work inside can freely transit across these rooms. These occupants are naturally chosen as people of interest (POI). For the office, face images are captured with a surveillance camera that faces the entrance. The presence logs of occupants' WiFi MAC addresses are collected by a sniffer that is situated in the center of the room for the same time period. Besides the POI's faces, these images also contain the faces of 11 short-terms visitors who came to this floor during the experiments. We put different cameras in different rooms to examine the performance of AutoTune under camera heterogeneity. To further examine the resilience of AutoTune, we put cameras in adversarial positions. In kitchen, we deploy cameras with bad views near entrance so that they can only capture subjects above 1.7m. While in the meeting room with two entrances, only the primary entrance is equipped with cameras. Therefore in both rooms cameras constantly *mis-capture* face images of subjects. Tab. 1 summarizes this data collection.

CHN Site: We collect another dataset in a common room of a university in China. There are no long-term occupants in this site and all undergraduates can enter. Of the 37 people that appeared during the three week period, 12 subjects are selected as the POI, and their WiFi MAC address presence is continuously recorded by the sniffer. Other settings remain the same as the UK site. The challenge in this dataset lies in that the captured face images, both for POI and non-POI, are all of Asian people, while the initial face representation model is trained primarily on Caucasians. Details of the CHN dataset are given in Tab. 1. Observation noises are very common in this dataset.

7.2.2 Overall Labeling Performance. We start our evaluation with one room only and compare our results with baselines. Fig. 5 shows the performance of label assignment, i.e., matching an identifier to a face image. For the office dataset, AutoTune outperforms the best competing approach (OA), by 0.13 in F_1 score and 7% in accuracy. The advantage of AutoTune is more obvious in the CommonRoom experiment where it beats the best competing approach (OA) by 0.34 in F_1 score and 10% in accuracy. As the only method that uses the website images (one-shot learning) to label images rather than the device ID information, TM struggles in both experiments and is 9-fold worse than AutoTune. We observe that the website face images are dramatically different from the captured images in real-world, due to different shooting conditions and image quality. These results imply that, although the device observations via WiFi sniffing are noisy, when the amount of the them is enough, they are more informative than the web-crawled face images.

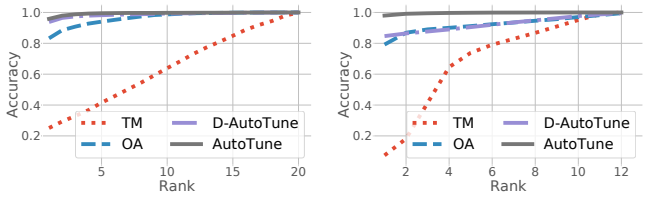
Additionally, we note that adopting soft labels (see §4.2) can further improve the labeling performance. In terms of F_1 score, the full-suite AutoTune is around 15% and 22% better than D-AutoTune in the two experiments respectively. Similar improvements are witnessed in terms of the accuracy. The reason of the larger performance gap in the CommonRoom dataset is that there are more images of non-POI being captured in this experiment. In addition, the pre-trained FaceNet model does not generalize very well to Asian faces, which are the primary ethnicity in the CommonRoom.

Lastly, we found that the choice of algorithm for cross-event clustering affects the final label-association accuracy. The best performance is achieved with the default agglomerative clustering algorithm on both datasets. The second-best spectral clustering (SC) is slightly inferior to agglomerative clustering in the office dataset, though the gap between them gets large in the CommonRoom dataset. The Gaussian Mixture Model (GMM) is inferior to the other two clustering algorithms. This is because GMM is best suited to Mahalanobis-distance based clustering whereas the distance space defined in Eq. 4.1 is a non-flat space due to the introduced attendance information. Hence, its resultant clusters are very impure and give poor association performance.



(a) UK (b) CHN

Figure 8: Impact of ID update rate β (introduced in §5.2). It can be seen that a high update rate causes the network to converge rapidly to potentially incorrect assignments.



(a) UK (b) CHN

Figure 9: Online Identification Performance

7.2.3 Performance vs. Scalability. We further examine AutoTune with multiple rooms under different adversarial conditions. Together with the *UK office* data, we evaluate AutoTune when events are collected in three different locations and via heterogeneous cameras. In particular, a kitchen and a meeting room are included and each of them has two lower-fidelity cameras (Mi Home Camera and Raspberry Pi Camera). The setup of multi-location experiments is given in Tab. 1. As described in §7.2.1, there are many face misdetections in the kitchen and meeting room due to the adversarial camera setups. Nevertheless, as shown in Fig. 6, AutoTune only suffers little performance drop (≤ 0.03) when erroneous camera observations are mixed with the single-office data. In terms of F1 score, which is the most important metric, AutoTune can achieve comparable level with the office experiment, regardless of which camera is added. Moreover, AutoTune still maintains its good performance even in the most adversarial case, where both errors are mixed in (three-room experiment). The main reason is that by using more data, though noisy, extra validation constraints are also utilized by AutoTune and make itself robust to observation errors.

7.2.4 Performance vs. Lifespan. AutoTune exploits co-located device and face presence in different events to establish the cross-modality correlations. In this section, we investigate the impact of the collection span on the performance of labeling. Longer collection days give more events. We investigate its impact by feeding AutoTune with data collected in different number of days, and compare them with all days on two datasets respectively. Fig. 7 shows that AutoTune performs better with increasing number of days on both datasets. The gap of F_1 score between the case with all days (10 days) and case with the least amount of days (2 days) can be as large as > 0.32 on both datasets. As discussed in §4.2, the ID association needs sufficiently diverse events to create discriminative enough event vectors. Otherwise, there will be faces or devices with the same event vectors that hinders AutoTune's ability to disambiguate their mutual association. However, we also observe that when we collect more than 8 days, the performance improvement of AutoTune becomes marginal.

7.2.5 Impact of ID update rate β. This section investigates the impact of ID update rate β introduced in §5.2. The ID update uses the fine-tuned face recognition model to update the device ID observations. A large ID update rate forces the device ID observations to quickly become consistent with the deep face model predictions. However, a large update rate also runs the risk of missing the optima. We vary the update rate β from 0.05 to 0.20 at a step length of 0.05. Fig. 8a demonstrates that AutoTune achieves the best performance on the UK dataset when the update rate is set to 0.05. The performance declines by 10% when the rate rises to 0.2. This is because the updated ID observations quickly become the same as the model predictions and the model predictions are not quite correct yet. When it comes to CHN dataset (see Fig. 8b), similar trend of F_1 score change can be seen. Although overall, the convergence becomes faster when the update increases, we observe that there is a fluctuation point at the update rate of 0.15, where AutoTune takes 7 iterations to converge. By inspecting the optimization process, we found that, under this parameter setting, AutoTune oscillated because the large update step makes it jump around in the vicinity of the optima but it is unable to approach it furthermore. In practice, we suggest users of AutoTune to select their update rate from a relatively safe region between 0.05 to 0.1.

7.3 Online Face Identification

7.3.1 Setup. After the cross-modality label association, an image database of POI is developed and an online face recognition system can be trained based on it. As in [26], we use the face representation model to extract face features and then use a classifier, say linear SVM in this work, to recognize faces. Particularly, as AutoTune and D-AutoTune are able to update the face representation model in their labeling process, the feature extractor for them are the updated model by themselves. For competing approaches that do not have an evolving model updates, we use the original Facenet model (in §6.2) as to extract face features and train the same classifier on the developed image-identity database by themselves. Without any overlapping with the images used for labeling, the test sets are collected in different days. In total, the test set contains 5,580 face images collected in the UK and 2,840 face images collected in China.

7.3.2 Performance. Fig. 9 compares the face identification results of both AutoTune and the competing approaches. Face identification is a multi-class classification task and aims to find an unknown person in a dataset of POI. As we can see, by using the face database developed by AutoTune, the identification accuracy quickly saturates and is able to have no errors within three guesses (rank-3) on both datasets. For the UK dataset where there are 20 POI, the rank-1 accuracy of AutoTune can be as high as 95.8% and outperforms the best competing approach (OA) by 12.5%. The advantage of AutoTune is more significant on the CHN dataset, and it surpasses the best competing approach OA by $\sim$ 19% (98.0% vs 79.1%). In addition, although D-AutoTune's performance is inferior to AutoTune, it is still more accurate than competing approaches, especially on

the UK dataset. We note that these results are consistent with face labeling results in §7.2.2. Overall, face identification by AutoTune are highly accurate, considering that AutoTune is only supervised by the weak and noisy device presence information.

7.4 Sensitivity Analysis

To further study the performance of AutoTune under different noise conditions, we also conduct extensive sensitivity analysis via simulation, considering three types of noise that are common in real-world settings:
False-alarm Faces is the case that the number of detected faces of POI is *greater than* the number of distinct sniffed MAC addresses.
False-alarm Devices is the case that the number of detected faces of POI is *smaller than* the number of distinct sniffed MAC addresses.
Non-POI Disturbance is the case that there are detected faces that belong to a non-POI, whose MAC address are not in our database i.e. they should be discarded as an outlier.

7.4.1 Data Synthesis. Although a number of public databases of face images are available, unfortunately, none of them come with device presence observations, i.e., sniffed MAC addresses. As an alternative, we develop a synthetic dataset on top of the widely used PIE face database[9]. In total 40 subjects are randomly selected, in which 30 of them are POI and the rest are mixed in to make the dataset realistic. We then assign to these "events" noisy identity observations to simulate device presence information. Based on different types of noise, different synthetic datasets are generated on which AutoTune is examined. On average, each subject in our simulation has 168 images after data augmentation [12]. As there is no actual device presence, we simulate this by first randomly placing the face images into different "events", in which multiple subjects "attend". As the number of images of subjects might be skewed in events, we adopt the F_1 score as the metric to evaluate the AutoTune's performance.

7.4.2 False-alarm Faces. WiFi sniffing is an effective approach to detect the presence of users' mobile devices, however, such detection is not guaranteed to perfectly match the user's presence. For instance, a device could be forgotten at home, be out of battery or simply the WiFi function might be turned off. In the simulation, we vary the error rate of such false alarm faces from 0.1 to 0.5. Error rate at 0.1 means that on average, 10% of the detected faces in each event are false detections. Fig. 10a shows the F_1 score and convergence iterations of AutoTune under different levels of such noises. As we can see AutoTune tolerates false-alarm faces well and is able to keep the F_1 score above 0.83 when the false alarm rate is below 0.4, though it degrades to 0.67 when the rate rises up to 0.5. However, we found such case, i.e., on average half of WiFi MAC addresses are missed in all meetings, is rare in the real-world. Finally, we found that false-alarm faces do not affect the convergence and AutoTune quickly converges within 4 iterations in all the cases.

7.4.3 False-alarm Devices. Though surveillance cameras are becoming increasingly ubiquitous, there are cases where the subjects are not captured by the cameras e.g. duto occlusions. This becomes an instance of device false alarm, if her device MAC address is still sniffed. We vary the rate of such false alarm devices from 0.1 to 0.5, where 0.1 means that on average, 10% of the detected devices in each event are false detections. Fig. 10b shows that although the F_1 score of AutoTune decreases, it degrades slowly and stops at 0.84 after the false-alarm rate becomes 0.5. As the injected noise becomes stronger, AutoTune needs more iterations to converge. However, the largest convergence iteration is still below 7 (rate at 0.4). Overall, AutoTune is very robust to such type of noises.

7.4.4 Non-POI Disturbance. Non-POI disturbance happens when subjects without registered MAC addresses are captured by the camera. We found such noise dominates all the three types of errors. We vary the number of non-POI from 2 to 10 and the probability of each non-POI's presence in an event is set to 0.1. Fig. 10c shows that AutoTune does not suffer much from mild disturbance (2 non-POI), and the F_1 score drops slowly to 0.87 with larger disturbance (4 non-POI). In addition in all cases AutoTune quickly converges within 5 iterations.

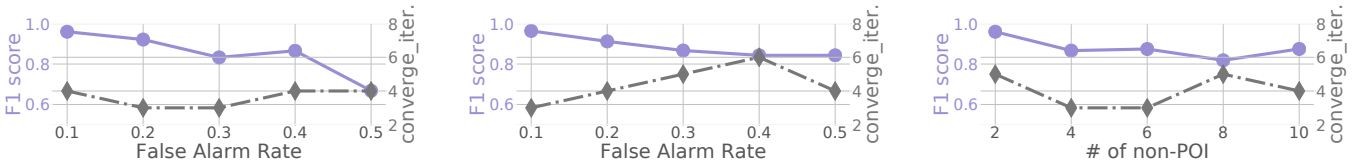
8 DISCUSSION AND FUTURE WORK

This section discusses some important issues related to AutoTune.
Overheads: Compared with conventional face recognition methods, AutoTune incurs overheads due to FaceNet fine-tuning. In our experiment, fine-tuning is realized on one NVIDIA K80 GPU. One fine-tuning action takes around 1.2 hours and 0.5 hours for the UK dataset and CHN dataset respectively. As we discusses in §7.2, AutoTune is able to converge within 5 iterations for most of the time, depending on the hyper-parameter setting. Therefore, fine-tuning overheads can be controlled in 7 hours for the UK dataset and 2.5 hours for the CHN dataset. Compared to the FaceNet pre-training that takes days and requires more GPUs, the fine-tuning costs of AutoTune is much cheaper. In future work, we will look into an online version of AutoTune, which can incrementally fine-tune the network on the fly when the data is streaming in.
Privacy: In practice, AutoTune requires face images and device ID of users to operate, which may have certain impacts on user privacy. For example, a user could be identified without explicit consent in a new environment, if the owner has the access to the face image of this user. In this work, we do not explicitly study the attack model in this context, we note that potential privacy concerns are worth exploring in future work.

9 CONCLUSION

In this work, we described AutoTune, a novel pipeline to simultaneously label face images in the wild and adapt a pre-trained deep neural network to recognize the faces of users in new environments. A key insight that motivates it is that enrolment effort of face labelling is unnecessary if a building owner has access to a wireless identifier, e.g., through a smart-phone's MAC address. By learning and refining the noisy and weak association between a user's smart-phone and facial images, AutoTune can fine-tune a deep neural network to tailor it to the environment, users, and conditions of a particular camera or set of cameras. Particularly, a novel soft-association technique is proposed to limit the impact of erroneous decisions taken early on in the training process from corrupting the clusters.Extensive experiment results demonstrate

[9] http://www.cs.cmu.edu/afs/cs/project/PIE/MultiPie/Multi-Pie/Home.html



(a) False-alarm Faces **(b) False-alarm Devices** **(c) Non-POI Disturbance**

Figure 10: Results on simulation data showing the impact of sources of noise

the ability of AutoTune to design an environment-specific, continually evolving facial recognition system with entirely no user effort even if the face and WiFi observations are very noisy.

ACKNOWLEDGMENTS

The authors would like to acknowledge the support of the EPSRC through grant EP/M019918/1 and Oxford-Google DeepMind Graduate Scholarship.

REFERENCES

[1] Alexandre Alahi, Albert Haque, and Li Fei-Fei. 2015. RGB-W: When vision meets wireless. In *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 3289–3297.

[2] Doug Beeferman and Adam Berger. 2000. Agglomerative clustering of a search engine query log. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 407–416.

[3] Lacey Best-Rowden, Hu Han, Charles Otto, Brendan F Klare, and Anil K Jain. 2014. Unconstrained face recognition: Identifying a person of interest from a media collection. *IEEE Transactions on Information Forensics and Security* 9, 12 (2014), 2144–2157.

[4] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. 2018. VGGFace2: A dataset for recognising faces across pose and age. In *IEEE Conference on Automatic Face and Gesture Recognition*.

[5] De Cheng, Yihong Gong, Sanping Zhou, Jinjun Wang, and Nanning Zheng. 2016. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1335–1344.

[6] Chris Fraley and Adrian E Raftery. 1998. How many clusters? Which clustering method? Answers via model-based cluster analysis. *The computer journal* 41, 8 (1998), 578–588.

[7] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. 2013. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*. 2121–2129.

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[9] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. 2007. *Labeled faces in the wild: A database for studying face recognition in unconstrained environments*. Technical Report. Technical Report 07-49, University of Massachusetts, Amherst.

[10] Roy Jonker and Ton Volgenant. 1986. Improving the Hungarian assignment algorithm. *Operations Research Letters* 5, 4 (1986), 171–175.

[11] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3128–3137.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[13] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[14] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. 2017. Sphereface: Deep hypersphere embedding for face recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[15] Weiyang Liu, Yan-Ming Zhang, Xingguo Li, Zhiding Yu, Bo Dai, Tuo Zhao, and Le Song. 2017. Deep Hyperspherical Learning. In *Advances in Neural Information Processing Systems*. 3953–3963.

[16] Chris Xiaoxuan Lu, Xuan Kan, Stefano Rosa, Bowen Du, Hongkai Wen, Andrew Markham, and Niki Trigoni. 2017. Towards Self-supervised Face Labeling via Cross-modality Association. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*.

[17] Chris Xiaoxuan Lu, Hongkai Wen, Sen Wang, Andrew Markham, and Niki Trigoni. 2017. SCAN: learning speaker identity from noisy sensor data. In *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 67–78.

[18] Xiaoxuan Lu, Hongkai Wen, Han Zou, Hao Jiang, Lihua Xie, and Niki Trigoni. 2016. Robust occupancy inference with commodity WiFi. In *IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*.

[19] Xiaoxuan Lu, Han Zou, Hongming Zhou, Lihua Xie, and Guang-Bin Huang. 2016. Robust extreme learning machine with its application to indoor positioning. *IEEE transactions on cybernetics* 46, 1 (2016), 194–205.

[20] Arsha Nagrani, Samuel Albanie, and Andrew Zisserman. 2018. Seeing Voices and Hearing Faces: Cross-modal biometric matching. *arXiv preprint arXiv:1804.00326* (2018).

[21] Andrew Owens, Jiajun Wu, Josh H McDermott, William T Freeman, and Antonio Torralba. 2016. Ambient sound provides supervision for visual learning. In *European Conference on Computer Vision*. Springer, 801–816.

[22] Savvas Papaioannou, Hongkai Wen, Zhuoling Xiao, Andrew Markham, and Niki Trigoni. 2015. Accurate positioning via cross-modality training. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. ACM, 239–251.

[23] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. 2015. Deep Face Recognition.. In *BMVC*, Vol. 1. 6.

[24] David Martin Powers. 2011. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. (2011).

[25] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. 2014. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596* (2014).

[26] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 815–823.

[27] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. 2014. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*. 1988–1996.

[28] Yi Sun, Xiaogang Wang, and Xiaoou Tang. 2014. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1891–1898.

[29] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning.. In *AAAI*.

[30] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. 2014. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1701–1708.

[31] Xiaoyang Tan, Songcan Chen, Zhi-Hua Zhou, and Fuyan Zhang. 2006. Face recognition from a single image per person: A survey. *Pattern recognition* 39, 9 (2006), 1725–1745.

[32] Jin Teng, Boying Zhang, Junda Zhu, Xinfeng Li, Dong Xuan, and Yuan F Zheng. 2014. EV-Loc: integrating electronic and visual signals for accurate localization. *IEEE/ACM Transactions on Networking (TON)* 22, 4 (2014), 1285–1296.

[33] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. 2014. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729* (2014).

[34] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 3156–3164.

[35] Stefan Wager, Sida Wang, and Percy S Liang. 2013. Dropout training as adaptive regularization. In *Advances in neural information processing systems*. 351–359.

[36] Junjue Wang, Brandon Amos, Anupam Das, Padmanabhan Pillai, Norman Sadeh, and Mahadev Satyanarayanan. 2018. Enabling Live Video Analytics with a Scalable and Privacy-Aware Framework. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 14, 3s (2018), 64.

[37] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. 2016. A discriminative feature learning approach for deep face recognition. In *ECCV*.

[38] Xuanli Lisa Xie and Gerardo Beni. 1991. A validity measure for fuzzy clustering. *IEEE Transactions on pattern analysis and machine intelligence* 13, 8 (1991), 841–847.

[39] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. 2016. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters* 23, 10 (2016), 1499–1503.

[40] Wenyi Zhao, Rama Chellappa, P Jonathon Phillips, and Azriel Rosenfeld. 2003. Face recognition: A literature survey. *ACM computing surveys (CSUR)* 35, 4 (2003), 399–458.

[41] C Lawrence Zitnick, Devi Parikh, and Lucy Vanderwende. 2013. Learning the visual interpretation of sentences. In *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 1681–1688.

Quality Effects on User Preferences and Behaviors in Mobile News Streaming

Hongyu Lu
BNRist, DCST, Tsinghua University
Beijing, China
luhy16@mails.tsinghua.edu.cn

Min Zhang*
BNRist, DCST, Tsinghua University
Beijing, China
z-m@tsinghua.edu.cn

Weizhi Ma
BNRist, DCST, Tsinghua University
Beijing, China
mawz14@mails.tsinghua.edu.cn

Yunqiu Shao
BNRist, DCST, Tsinghua University
Beijing, China
shaoyunqiu14@gmail.com

Yiqun Liu
BNRist, DCST, Tsinghua University
Beijing, China
yiqunliu@tsinghua.edu.cn

Shaoping Ma
BNRist, DCST, Tsinghua University
Beijing, China
msp@tsinghua.edu.cn

ABSTRACT

User behaviors are widely used as implicit feedbacks of user preferences in personalized information systems. In previous works and online applications, the user's click signals are used as positive feedback for ranking, recommendation, evaluation, etc. However, when users click on a piece of low-quality news, they are more likely to have negative experiences and different reading behaviors. Hence, the ignorance of the quality effects of news may lead to the misinterpretation of user behaviors as well as consequence studies. To address these issues, we conducted an in-depth user study in mobile news streaming scenario to investigate whether and how the quality of news may affect user preferences and user behaviors. Firstly, we verify that quality does affect user preferences, and low-quality news results in a lower preference. We further find that this effect varies with both interaction phases and user's interest in the topic of the news. Secondly, we inspect how users interact with low-quality news. Surprisingly, we find that users are more likely to click on low-quality news because of its high title persuasion. Moreover, users will read less and slower with fewer revisits and examinations while reading the low-quality news.

Based on these quality effects we have discovered, we propose the Preference Behavior Quality (PBQ) probability model which incorporates the quality into traditional behavior-only implicit feedback. The significant improvement demonstrates that incorporating quality can help build implicit feedback. Since the importance and difficulty in collecting news quality, we further investigate how to identify it automatically. Based on point-wise and pair-wise distinguishing experiments, we show that user behaviors, especially reading ratio and dwell time, have high ability to identify news quality. Our research has comprehensively analyzed the effects of quality on user preferences and behaviors, and raised the awareness of item quality in interpreting user behaviors and estimating user preferences.

*Contact author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313751

CCS CONCEPTS

• **Information systems → Evaluation of retrieval results**; **Users and interactive retrieval**.

KEYWORDS

Online news reading; Quality effect; User behavior analysis; User item-level preference; Implicit feedback

ACM Reference Format:
Hongyu Lu, Min Zhang, Weizhi Ma, Yunqiu Shao, Yiqun Liu, and Shaoping Ma. 2019. Quality Effects on User Preferences and Behaviors in Mobile News Streaming. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313751

1 INTRODUCTION

User preference has been widely used as a critical concept in personalized information system [13, 28]. Yet, it is difficult to acquire due to its subjective nature. The most direct way is to ask users to give explicit feedback, for example, the rating in e-commerce websites [30, 38]. However, because of the difficulty and the potential biases [24] in its collection in real-world applications, commercial information systems have exploited implicit feedback signals derived from user behaviors. Most commonly, users' clicking on the item have been used as implicit feedback of document relevance [1, 16] and user experiences [19]. In addition to the click signal, dwell time (i.e. the time that user spends on a clicked item) has also been found well correlated with document relevance [35], item-level satisfaction [9, 12, 20] and user preferences [6]. A click followed by a long dwell time has traditionally been seen as satisfied click and been widely used in a number of retrieval applications [9, 11]. Other behaviors, like mouse movement [27], scroll information [22], and gaze [2, 25] are also investigated for inferring user experiences.

However, user behaviors have been proved to be informative but noisy. Click behavior is found to be affected by many factors, like position [16], trust [37], and presentation [34]. Moreover, dwell time is also found related to the content length [36], readability [21]and search task [18].

The credibility of the behaviors may also be related to the item quality. For example, as shown in Figure 1, two pieces of news with different levels of quality are both clicked by the user. The low-quality news is more likely to be disliked by the user after reading its poor content. As a result, although user clicking on both news, his/her preference is significantly different because of the

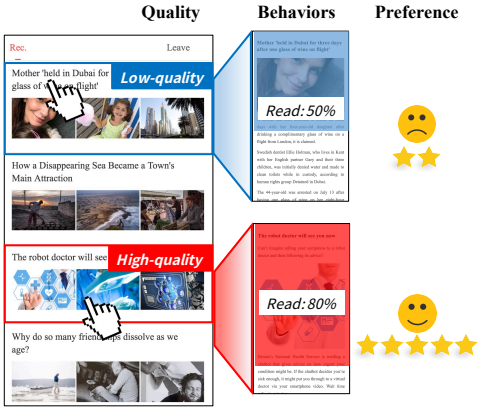


Figure 1: User clicking on items may have different behaviors (e.g. reading ratio) and different preferences because of the different levels of quality.

quality. Thus, traditional implicit feedback may yield imprecise user preference modelling because of the ignorance of the quality effect. To the best of our knowledge, there still lacks comprehensive understanding of how item quality affects user preference and user behaviors, which is important for modeling user preference and utilizing user behaviors. In this paper, we study the quality effects and aim to address the following questions:

- **RQ1**: Does quality affect user preferences? If yes, how?
- **RQ2**: Does quality affect user behaviors during the browsing and reading process? If yes, how?
- **RQ3**: Can incorporating quality help build implicit feedback?
- **RQ4**: Can we identify quality based on user behavior signals?

We conduct an in-depth user study in online news reading scenario on the mobile device. At the beginning of the study, we collected user's interest in the topics. After that, users are asked to read several lists which contains the news of different levels of quality. In this process, we collected their behaviors as well as explicit feedback for experiences, including the perception of quality and the preferences for the news.

As for RQ1, by comparing user preferences for low-quality and high-quality news, we find that lower quality leads to a lower user preference. From this, we further analyze the degree of quality effect, measured by the difference of user preferences between low-quality and high-quality news, and find it is related to the interaction phases and user's interest for the news topic. On the one hand, from before-read phase to after-read phase, the quality effects increase. On the other hand, the quality effect is much larger when the user is more interested in the news topic.

Besides the effects on user preferences, we further investigate the quality effects on user behaviors (RQ2). From the comparison of click behaviors, we find that users have higher probability to click the low-quality news. Part of the explanation may be the higher title persuasion of the low-quality news. This effect is more distinct when a user has higher interest in the topic of the news. Furthermore, we find that a user's reading behaviors after clicking are also influenced by the news quality. Users will read less in the low-quality news, as shown by the reduction of *dwell time* and *reading ratio*. They will also have fewer revisits and examinations. These quality effects promote us to re-think the implicit feedback (RQ3).

Traditional implicit feedback relies on the relation between user preferences and user behavior. Because of our findings about quality effects on user preferences and behaviors, we further take the quality and its effects into consideration, and propose a probability model to estimate user preference. The out-performance of the proposed model demonstrates that it is helpful to incorporate quality into implicit feedback building.

Considering the usefulness but the hard annotation of quality, it is valuable to automatically identify news quality (RQ4). We examine the point-wise and pair-wise distinguishing ability of each behavior, and confirm the possibility of using user behavior signals, like reading ratio and dwell time, to identify the quality of news.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 introduces the methodology of our user study and collected measurements. Section 4 studies the quality effects on user preferences (RQ1). The analysis of the quality effect on user behaviors (RQ2) is given in Section 5. Section 6 investigates the usage of quality for improving the implicit feedback building (RQ3). Section 7 discusses the possibility of utilizing behaviors to identify quality (RQ4). Finally, Section 8 summarizes the conclusion and future work.

2 RELATED WORK

In this section, we review two related research directions, the modeling of user item-level experiences and the analysis of user behaviors.

2.1 User Item-Level Experience Modeling

Modeling user item-level experiences is a fundamental work for implicit feedback building and performance evaluation in information system. Some previous works have paid attention to this direction, such as modeling user perceived usefulness in web search and user preferences in recommender systems.

Belkin et al. [4] argue that usefulness, which represents users' perceived value of a search result, depend on the scenario and context of accessing the result. Mao et al. [29] also find that usefulness is related to current search task and redundancy with previous documents read by the user. Jiang et al. [15] have collected usefulness judgment not only in situ stage but also post-session context-independent judgment, and find that they are different, indicating that context will affect user perception of usefulness.

Compared with the usefulness, user preference is more personalized and subjective, making it hard to model it. Lu et al. [28] model user preference as a dynamic concept varying in different interaction phases, and found that the change of preference after user reads the news content is related to the news quality. Their work provides a good starting point for modeling user preferences with item quality, but more in-depth researches still remains to be studied. We go further to analyze the quality effect on user preferences jointly with interaction phases and user topic interest, examining how the effect varies in different interaction phases and when users have different levels of interest on the news topic.

2.2 User Behavior Analysis

Learning from user behaviors is a general approach used in online interactive information systems. Researchers of Information Retrieval use click signals to infer document relevance [8] under the assumption that relevant documents attract more clicks. Personalized information filtering systems, like recommender systems, are also designed by mining user preferences through historical click data [13, 32].

However, user click behavior has been found to be biased because of the influence caused by some factors, for example, position [16]. Documents on the top may attract more user clicks [17]. Other factors like trust [16], result attractiveness [37] and presentation [34] are also examined. To address the biases of user click behavior, researchers have proposed a number of advanced click models [5, 10], which are designed to eliminate the effects of various biases to obtain an unbiased estimation of result relevance in web search scenarios.

Besides click, some other behaviors are also examined and incorporated to model user experiences. Among them, click dwell time has been successfully used in many retrieval applications. A dwell time equaling or exceeding 30 seconds has typically been used to identify clicks with which searchers are satisfied [9]. The correlation between dwell time and user interest is further modeled by document factors (e.g. readability [21], genres [3], and human factors [36]). Besides dwell time, viewport time is successfully used on mobile devices to infer user interest at the sub-document level [14]. Li et al. [25] examine the correlation between users' eye gaze and user explicit interest, and demonstrate the effectiveness of using attention-based behaviors, like viewport time and gaze, to predict user interest. Lin et al. [26] find that the delivery mechanism of the system greatly affects user information consumption behaviors. Ben et al. [31] investigate the causal effects of ad blocking on user long-term behaviors, like active time spent in the browser and the number of page. Lu et al. [28] investigate the influence of user preferences on user behaviors in news reading scenario and in mobile environment.

While many factors related to user behaviors have been investigated, the item quality is less studied. Based on a carefully designed user experiment, we studied the directly effects of news quality on user browsing and reading behaviors with other factors controlled, like position, topic and user preferences.

3 USER STUDY METHODOLOGY

To measure the effects of news quality on user preferences and behaviors, we designed a laboratory user study in which we varied the news quality with other factors controlled and inspect whether user experiences and behaviors become different.

3.1 Experimental Setup

In this section, we introduce the settings of our experiment, including how we sample the news data and annotate their quality as low-quality and high-quality, how we generate news lists with other factors controlled, like position and topic and what are the interfaces of the experimental platform.

3.1.1 Data.

The news used in the user experiment is sampled from the real log of *Sougou NewsFeed*, a popular commercial newsfeed service on mobile device. We firstly choose five most popular topics: *social*, *entertainment*, *technology*, *history*, and *sport*. Then we randomly sample one hundred pieces of news from each topic. By doing this, we cover a variety of topics and ensure a balance between them to eliminate the possible bias.

After that, a human annotation is conducted to measure the quality of these pieces of news. We recruit three experts to label the overall quality (binary-scales: low-quality or high-quality) by discussion according to the following four properties.

Authenticity: High authenticity means that this piece of news is authentic or has high credibility [33]. On the contrary, if the content is imaginary or exaggerated, the news is seen of low authenticity.

Value: A piece of news is of low-quality value when its content is full of problems with being vulgar, violent, bloody and pornographic.

Expression: The expression is high-quality when the statement is objective and precise, and the information is rich but not redundant.

Headline: Low-quality headlines have one of such problems: information is incomplete or fake; the expression is exaggerating or vulgar and inconsistent with the content.

The validity of the overall quality label, namely *Expert Labeled Quality (EQ)*, are tested by comparing it with the average perceived quality annotated by the experiment participants. (EQ vs. User perceived content quality Fleiss's k=0.5017; EQ vs. User perceived title consistency k=0.4737).

3.1.2 News Lists.

Based on the binary overall quality labelled by experts, the news are classified into two groups, low-quality and high-quality news. We generate sixteen news lists (fifteen news each) containing both low-quality and high-quality news as tasks in advance.

Previous works show that user behaviors are affected by many factors, like position [16]. To control the impact of position, we take the Latin square experimental design principles to assign the positions of low and high-quality news in the lists. Meanwhile, we assigned the same number of news from five topics in each task list to control the effect of news topic. The careful design ensures that the low-quality and high-quality news have the same position and topic distribution.

3.1.3 Online Platform.

To simulate a real online news reading environment, we build an experimental platform which is similar to the common used news-feed website. There are two types of pages, one is the list page for the user to browse the news list (Figure 2 left and right pictures), the other is the news content page for users to read the news content (Figure 2 middle picture). A JavaScript plugin is injected into both pages to record user's scrolling, clicking, and page switching behaviors.

3.2 Experimental Procedure

In this section, we describe the procedure of the experiment and the collection of user experiences in multi-phases.

3.2.1 Procedure.

Before the experiment, we first collect user's interest of the five topics, namely *Topic Interest* (5-ratings each). To get familiar with the experiment platform and procedure, the participants need to



Figure 2: Collect user experiences in different phases: *Before-Read, After-Read,* and *Post-Task Phase.*

complete an example task for training. After that, users are asked to finish four tasks.

At the beginning of each task, users need to read the task description carefully. Then, a list of 15 pieces of news is shown to the user for browsing and reading. As no browsing and reading time limitations are imposed, users can finish the browsing task at any time.

3.2.2 Multi-Phase Experience Measurement.

Within the task procedure, we add several questionnaires to collect user experiences in three phases: Before-Read, After-Read and Post-Task, as shown in Figure 2.

Before-Read Phase

- **Q1:** How much do you expect to like or dislike reading this piece of news? (5-point Likert scale, from very dislike to very like)

Firstly, as soon as a user clicks a piece of news, before showing its content, we ask the user about his/her expected preference for the news. This preference is named as *Before-Read Preference*.

After-Read Phase

- **Q2:** How much do you like reading this piece of news? (5-point Likert scale, from very dislike to very like)
- **Q3:** What do you think of the content quality of this piece of news? (5-point Likert scale, from very poor to very good)
- **Q4:** What do you think of the consistency between title and content of the news? (5-point Likert scale, from very low to very high)

After a user reads the news content and deciding to leave (note that there is no constraints on user reading process, users can leave at anytime as he/she wish), we ask the user several questions for his/her experiences, including *After-Read Preference* (Q2), *User Perceived Content Quality* (Q3) and *User Perceived Title Consistency* (Q4).

Post-Task Phase

- **Q5:** How much do you like reading this piece of news? (5-point Likert scale, from very dislike to very like)

After users finish browsing, we randomly shuffle and re-display the news that they have seen (right subplot in Figure 2. "Seen" means that the news has appeared on the user's viewport (including the news not clicked). We ask users to re-annotate his/her preference on each news, named *Post-Task Preference*.

We finally recruit 32 participants, including 18 females. Across 128 tasks finished, 1920 impressions (including 576 low-quality news) are shown, and 631 clicks (including 209 clicks on the low-quality news) are collected. [1]

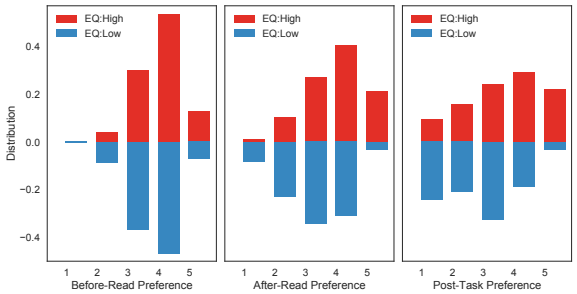


Figure 3: When the news quality is different (EQ:High vs. EQ:Low), the distributions of user preference in multi-phases: Before-Read (left), After-Read (center) and Post-Task (right).

Table 1: The Pearson correlations between news quality and user preference in multi-phases. The quality is measured by both expert (EQ) and user (User Perceived Content Quality, UQ-C; User Perceived Title Consistency, UQ-T).

	EQ	UQ-C	UQ-T
Before-Read Preference	0.1480	0.3267	0.2964
After-Read Preference	0.3292	0.8170	0.5874
Post-Task Preference	0.3077	0.6992	0.5652

4 QUALITY EFFECTS ON USER PREFERENCE

Based on the quality measurements labeled by both expert and users, and the explicit user preference feedback collected in the experiment, we now investigate the effect of quality on user preference.

4.1 Different Interaction Phases

To begin with, we compare the distribution of user preferences for low-quality and high-quality news, as shown in Figure 3. Generally, significant difference is found in user preference in all the three interaction phases (Before-Read Preference, p<0.01, d =0.32; After-Read Preference, p<0.01, d =0.74; Post-Task Preference, p<0.01, d =0.69). The results suggest that news quality has significant effects on user preference in all three phases. To be more specific, we find the degrees of the differences vary in different phases. It promotes us to jointly analyze the quality effects with the interaction phases.

We separate the news into different groups based on the quality measurement, not only the expert labeled quality (binary-scale, two groups), but also the user perceived content quality (5-scales, five groups) and the user perceived consistency between title and content (5-scales, five groups). For each group, we calculate the mean of user preferences in three phases (*Before-Read, After-Read, Post-Task Phases*). The results are shown in Figure 4.

Firstly, we can see that user preferences for the high-quality news are more likely to stay or increase (from Before-Read to After-Read and to Post-Task, shown as mark-1 in the figure 4), and vice versa, user preferences for the low-quality news continually drop

[1]The data is now available at http://www.thuir.cn/group/~mzhang/

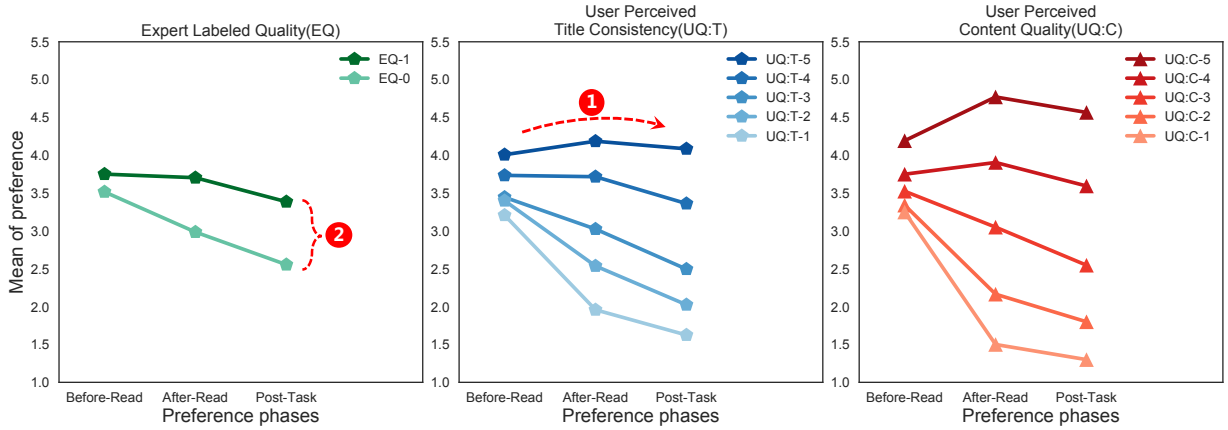


Figure 4: Means of user preference (in three phases) for the news of different levels of quality. The quality is measured by Expert Labeled Quality (left), User Perceived Title Consistency (center), and User Perceived Content Quality (right). The level of quality is distinguished by the shade of color.

Table 2: Quality effects, measured by the difference (Δ, Cohen's *d*) of user preferences between low-quality and high-quality news, when user has different topic interests (TI).

	Before-Read Preference				After-Read Preference				Post-Task Preference			
	EQ:Low	EQ:High	Δ	*d*	EQ:Low	EQ:High	Δ	*d*	EQ:Low	EQ:High	Δ	*d*
TI=Min	3.610	3.689	+0.079	0.114	3.170	3.597	+0.427	0.447	2.627	3.176	+0.549	0.454
TI=Mid	3.465	3.630	+0.166	0.223	2.831	3.674	**+0.843**	**0.870**	2.563	3.442	+0.879	0.751
TI=Max	3.494	3.897	**+0.403**	**0.532**	2.987	3.806	+0.819	0.838	2.494	3.491	**+0.997**	**0.801**

along user reading. This finding is consistent with Lu's work [28]. Starting from this point, we further analyze how the quality effects varies in different phases. The degree of quality effect is seen as the gap of preferences between low-quality and high-quality news (larger gap means higher degree, shown as mark-2 in the figure 4). In Before-Read Phase, user preferences for the news of low-quality and high-quality are already different, but more slightly than in After-Read and Post-Task Phases. It may be due to that user only knows a little part of the news before reading its content, thus quality is less perceived and has less effect on user preferences.

Meanwhile, although the trend of quality effect is similar when using different quality measurements, user perceived quality has a larger effect than objective expert labeled quality. It may because that user's perception of quality effect may be personalized and related to other subjective factors like user's interest on the topic of the news, which motivates us to further investigate the quality effect with user topic interest in next section.

In addition, the effect of user-perceived content quality is larger than the title consistency. By applying the Pearson's r, we further directly compare the correlation between user multi-phase preferences and different quality measurements. The results, shown in Table 1, indicate that the correlation between quality and preference is largest in After-Read Phase, and content quality is more closely related to user preference than title consistency and expert labelled quality.

4.2 Different Levels of Topic Interests

Beside of interaction phases, we further investigate whether the quality effects on user preferences are related with user's interest of the news topic. As mentioned in Section 3.2.1, users' interests on all the five topics are collected at the beginning of the user study, by asking users to rate each topic. To avoid users' different understandings of topic interest ratings, we separate the five topics within each user into three groups. The topics with the highest user interest are set as "Max" group (not necessarily one topic), and vice versa, the topics with lowest user interest are set as "Min" group, and the other topics are set as the "Mid" group.

The means of user preferences for low-quality and high-quality news with different levels of topic interest are shown in Table 2. We further measure the quality effect on user preferences by calculating Δ (preference of high-quality news − preference of low-quality news) and Cohen's *d*. On the one hand, the quality effect is lowest in *Before-Read phase* and highest in *Post-Task phase*, which is same as previous analysis.

On the other hand, the quality effect is highly related to the topic interest. When user has low topic interest (Min), the difference of preferences between low-quality and high-quality news is small. On the contrary, when user has higher topic interest (Mid and Max), the quality effect is much larger. It can be interpreted as the user has less quality sensitiveness for the news of his/her less interested topics. Moreover, although topic interest is generally positively correlated with user preferences, when the news is of low quality,

higher topic interest leads to lower preferences. It can be explained by the less user tolerance of quality for his/her highly preferred topics.

5 QUALITY EFFECTS ON USER BEHAVIORS

In this section, we focus on investigating whether users behave differently when interacting with low-quality and high-quality news. Users' scroll and click events with timestamps are recorded in user browsing (list page) and reading (content page) processes, and are used to build several frequently used behavior measurements.

5.1 Click Behavior

Click behavior has been widely used as implicit feedback for recommendation, ranking and evaluation. In this section, we investigate the quality effects on user's click behavior, measured by the probability of users clicking the news. If there is a dependency with the quality, any interpretation of clicks as implicit preference feedback should be relative to the quality.

We first evaluate the probability for users clicking on low-quality or high-quality news. The conditional probability shows that if the news is of high quality, it will be clicked with a probability of 0.3140 ($P(Click|EQ = 1)$), vice versa, the click probability for low-quality news is 0.3628 ($P(Click|EQ = 0)$). The difference is slightly but significant according to the independent t-test, p<0.05, d=0.10. It implies that low-quality news is more likely to be clicked than high-quality news.

Incorporating the position of the news into analysis, we calculate the click probabilities conditioned by the quality for top-k impressions (K ranging from 1 to 15) are shown in Figure 5. By separating different levels of quality, we can find that the news of low quality has higher click probability than high-quality news in all the Top-K positions.

To further verify the reliability of the finding that low-quality news has a higher probability of being clicked, we test whether it exists in real system. Note that the news used in our work is sampled from *Sougou Newsfeed*. We further collect one week's log data about these piece of news (from October 16, 2017, to October 22, 2017, more than 1.5 thousands interactions per news).

Based on the quality labeled by expert (EQ) and the CTR calculated in the log for each news, we are able to compare the average CTR of low-quality and high-quality news. The result of independent t-test shows that there is a significant difference of CTR for low-quality news (0.1539) and high-quality news (0.0835), p<0.01. The observation is similar with the result in user study, and proves that the news quality does affect user click behavior and low-quality news is more likely to be clicked.

5.1.1 *Title Persuasion.*

To further study why low-quality news attracts more clicks, we conduct a supplementary annotation for the persuasion of the title. The *title persuasion* is defined as the extent that user is seduced to click the news (4-scales). We hired external assessors from JD crowd source platform and make sure each piece of news was annotated by 3 different assessors (Fleiss' $k = 0.4259$, reach moderate agreement). We use majority voting to get the final score.

Based on the annotation, we compare the title persuasion of low and high-quality news and find low-quality news has higher

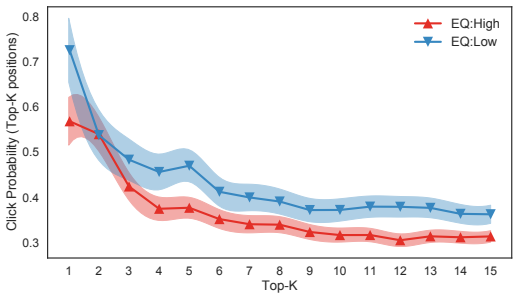


Figure 5: Click Probability of the news *up to position k* conditioned by the news quality. The low-quality news attracts more clicks.

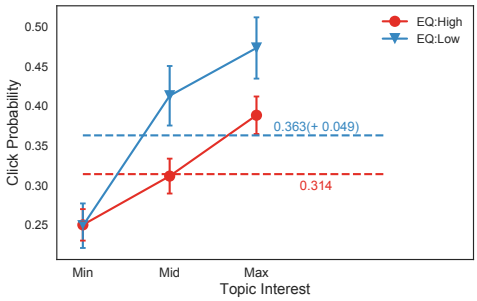


Figure 6: Click probability of the news with different levels of quality and user topic interests.

Table 3: The contextual effect of quality on click probability. When the quality of last impression news (lEQ) varies, how is the click probability of current (c) impression news?

	$lEQ = low$		$lEQ = high$	
$P(Click\|\ lEQ)$	0.3507		0.2898	
$P(Click\|\ lEQ, cEQ)$	$cEQ = l.$	$cEQ = h.$	$cEQ = l.$	$cEQ = h.$
	0.4000	0.3108	0.2838	0.2917

persuasion than high-quality news (2.16 vs. 1.61, p<0.01). It is reasonable that low-quality news usually uses literary methods, like the exaggeration, to generate a more compelling title. Thus, user may click on low-quality news because of its high title persuasion.

5.1.2 *Topic Interest.*

We further study how quality effects on user's clicking decision change when user has different levels of topic interest. We show the click probability conditioned by the news quality and topic interest ($P(Click|Q, TI)$), in Figure 6.

Beside of the separate effects of quality and topic interest, there also exists an interaction effect. When topic interest is low (Min), user has almost the same click probability for the news of low or high quality. However, when topic interest is high (Mid, Max), the click probability of low-quality news exceed high-quality news. The reason is that when the news is of low topic interest, the user may be not easily aware of its quality, so the click decision is less affected.

5.1.3 *Contextual Effect.*

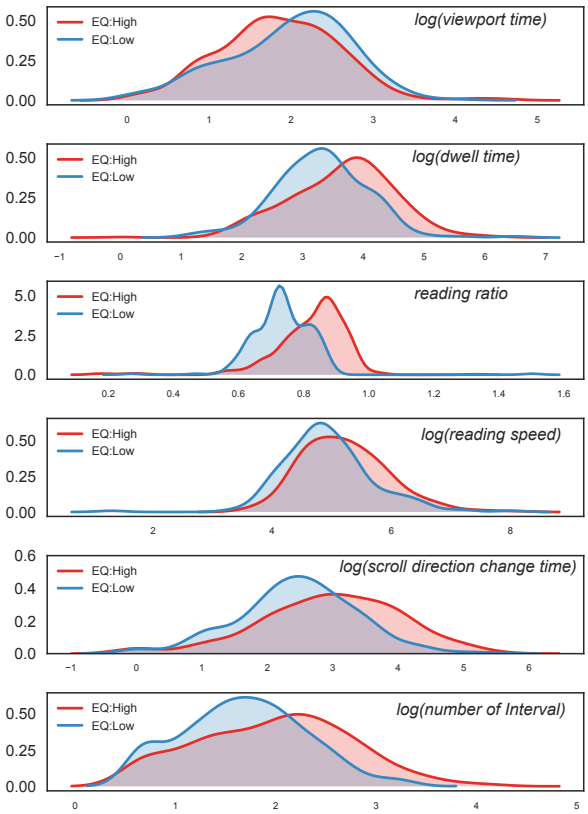


Figure 7: The probability density function (estimated by kernel density estimation) of behaviors when users are reading the news of different qualities (EQ:High vs. EQ:Low)

As the user interacts with news in the list context, the quality of the news around may have contextual effects. In the mobile environment, most of users scroll to browse the list, examine news one-by-one. So we investigate how the quality of the last impression affects user click probability for the current news.

We calculate the probability of clicking on current news conditioned by the quality of the last impression. The results, shown in Table 3, indicate that user has a higher click probability when the quality of the last impression is low (0.3570 vs. 0.2898). We further control the quality of current news. Results show that no matter what the quality of current news is, the click probability is higher when the last impression is of low quality. It may be caused by the saliency attention mechanism, the saliency and click probability of a piece of news may depend on the context news.

5.1.4 *Summary.*

From above analysis, we can conclude that user click behavior is affected by news quality. Users have a higher probability of clicking the low-quality news. It promotes us that we should take quality into account when using user click signals to build implicit feedback. User topic interest is found to be related to the quality effect on click probability. The quality effect is lower when user has lower interest in the topic. Beside of affecting the click decision of current news, news quality also has effects on context news, like the next news in the recommendation list.

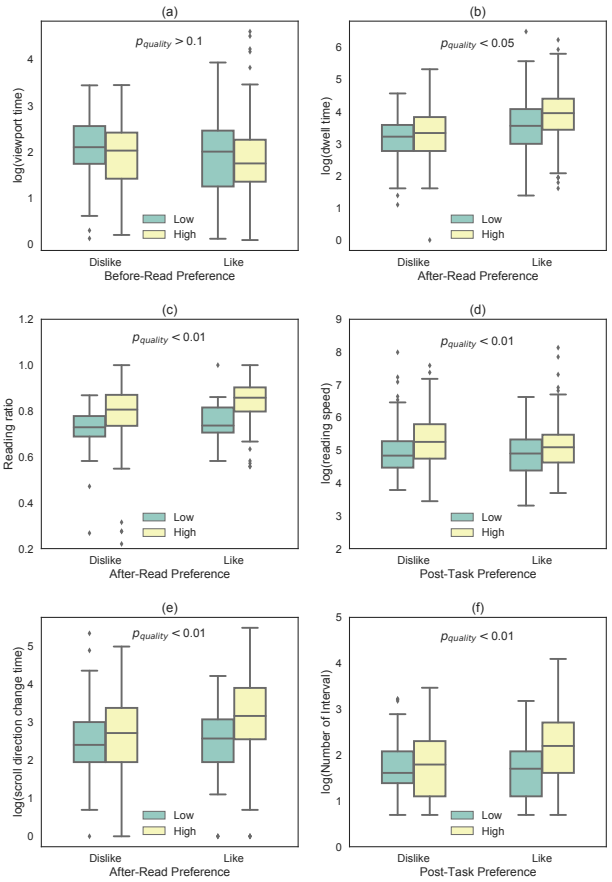


Figure 8: User behaviors for the news of different levels of quality (EQ: low, high) with different user preferences (Dislike($\leq$3), Like(>3)). The p-value displayed in each figure represents the main effect of quality in two-way ANOVA (with preferences).

5.2 Reading Behaviors

Beside of the click behavior, in this section, we investigate how news quality affects user's behaviors in the reading process. First of all, we directly compare the distributions of user behavior measurements when the news is of high-quality and low-quality, as shown in Figure 7. By doing this, we get an overview of news quality effects on behaviors.

In last section, we demonstrates that the news quality affects user's preferences, while previous work [28] shows that user preferences influence user behaviors. Hence, to analyze the impact of news quality on user behaviors, we further control the influence comes from user preferences.

For each behavior, we choose the preference in its most closely related phase based on the behavior-preference correlations, marked by the y-label in Figure 8. For example, the most closely related preference for dwell time is *After-Read Preference*, while for viewport time is *Before-Read Preference* and for direction change times is *Post-Task Preference*. If the preference less than or equal to 3 ($\leq$ 3), we consider it as *dislike*, other levels of preference are considered as *like*. Based on both analysis approaches, we study the quality effects on each reading behavior:

Viewport time.

Viewport time, reflecting how long user reads the news snippet in the list browsing page, may be related to the user click decision process. Because the data for viewport time does not meet the normality assumption, we use log-transformation, $log(x+1)$, before statistic analysis. In general, the result of independent t-test suggests there is a slightly but significant difference of viewport time between low-quality and high-quality news (p<0.05, d=0.17). We further take the effect of user *Before-Read Preference*, which is proved mostly related to viewport time, into account. The result of the analysis of variance (ANOVA) [7] considering both quality and preferences shows that there is no independently significant effect of quality on the viewport time (two-way ANOVA, $F(1,929)=4.47$, p>0.1). We then separate the news into dislike and like groups, and compare the means of viewport time between low-quality and high-quality news within each group. The result, shown in Figure 8 (a), indicates that user spends a longer time reading the snippet of low-quality news. While the difference is not significant (t-test, $p_{dislike}$>0.1, d=0.13 and p_{like}>0.1, d=0.11), such a longer time is plausible: user can perceive part of the quality by reading the title and images. Thus he/she needs longer time to make the click decision for low-quality news.

Dwell time.

Dwell time, indicating how long user reads the content of the news, has been found highly related to user post-click experiences. Same as viewport time, we also use log-transformation to normalize the data. Generally, there is a significant effect due to the news quality (t-test, p<0.01, d=0.42). By jointly analyzing with user preferences in *After-Read Phase*, the independently significant effect of quality is confirmed by two-way ANOVA ($F(1,929)=4.50$, p<0.05). With preferences controlled, we find that higher quality leads to higher dwell time, especially when the preferences is high (see Figure 8 (b)). Independent t-test is applied within both dislike and like groups. Results indicate that there is a significant difference when user likes the news, $p_{dislike}$=0.13, d=0.17 and p_{like}<0.01, d=0.36.

Reading ratio.

Dwell time has been proved to be affected by many factors, like content length and readability. For eliminating the effect of content length, we normalize the absolute read length by the whole length [23], named as reading ratio which directly reflect how deep user read. In general, user's reading ratio for low-quality and high-quality news is significantly different (t-test, p<0.01, d=0.85).

With user *After-Read Preference* controlled, the statistic results show that quality has still a significant effect on user reading ratio (two-way ANOVA, $F(1,929)=62.78$, p<0.01). Low-quality news has the lower reading ratio, which indicates that user will leave earlier when reading the low-quality news (see Figure 8(c)). t-test is applied. Results indicate that there is a significant difference in both dislike and like groups, $p_{dislike}$<0.01, d=0.49 and p_{like}<0.01, d=1.19.

Reading speed.

Combining dwell time and reading length (pixel), we calculated user reading speed (reading length/dwell time). As shown in Figure 8 (d), user reading slower when the quality is lower. The result is supported by both two-way ANOVA with *Post-Task Preference* ($F(1,929)=27.6$, p<0.01) and t-test within each preference group ($p_{dislike}$<0.01, d=0.49 and p_{like}<0.01, d=0.54)

Scroll direction change times.

When a user is reading the news content, he/she may change his/her reading direction to revisit some previous information. The number of the user changing his/her scroll direction indicates how often the user revisits. Considering only the news quality, there is a significant difference reported by t-test, p<0.01, d=0.49. With further investigation by controlling user *After-Read Preference*, we can conclude that quality effect still exists (two-way ANOVA, $F(1,929)=19.09$, p<0.01) and user revisits more when the news quality is high, especially when the user like the news. (t-test, $p_{dislike}$ =0.10, d=0.19 and p_{like}<0.01, d=0.69)

Number of interval.

In the content page, user read the news by scrolling, scroll intervals may represent user's examinations. Therefore, the number of user scroll interval indicates how many times user carefully examines some information along reading the news. In general and with user preferences controlled, the quality has a significant effect on the number of intervals (t-test, p<0.01, d=0.43; two-way ANOVA, $F(1,929)=9.05$, p<0.01). The result shown in Figure 8 indicates that when the user likes the news, higher quality leads to more scroll intervals (t-test, $p_{dislike}$=0.25, d=0.14 and p_{like}<0.01, d=0.39).

Summary.

In this section, we give the detailed analysis of the quality effects on user's reading behaviors. We find that when users read low-quality news, they may:

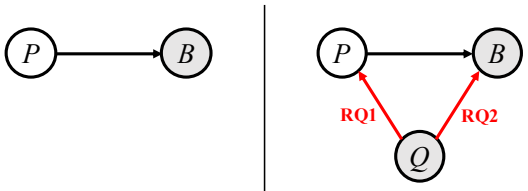
- spend less time reading (dwell time decrease)
- leave earlier (reading ratio decrease)
- read slower (reading speed decrease)
- have fewer revisits (scroll direction change times decrease)
- have fewer careful examinations (number of interval decrease)

The existence of quality effects on user behaviors promotes us to take quality into consideration when using user reading behaviors as implicit feedback. In next session, we further examine it through the probability model.

6 CAN INCORPORATING QUALITY HELP IN BUILDING IMPLICIT FEEDBACK?

To collect user's preferences, information systems have exploited implicit feedback signals derived from user behaviors. For example, the Satisfied-Click, which considers the click following a long dwell time as positive preference feedback, is widely used in today's systems. These approaches are relied on the assumption that user preferences affects user behaviors (e.g. dwell time). While no other factors are considered, the graphical model of this assumption can be shown as Figure 9(a), namely Preference Behavior (PB) model here.

Based on our previous findings about the quality effect on user preferences and user behavior, we argue that quality need to be taken into consideration when building implicit feedback. By studying RQ1, we find that the quality affects user preferences, so we add a relation between quality and user preference. Besides, by studying RQ2, we also conclude that news quality does have an effect on user behavior and the effect is independent of the effect of user preferences. Therefore, we further add a relation between



(a) PB-model (b) PBQ-model

Figure 9: The graph models of traditional implicit feedback which use Behavior to estimate Preference, namely PB-model (a), and implicit feedback which incorporates Quality, namely PBQ-model (b). *P*: preference; *B*: behavior; *Q*: quality.

Table 4: The performance of PB and PBQ with different behavior metrics in estimating user preference.(* represents p-value<0.05, ** represents p-value<0.01)

Behavior metric	AUC(PB)	AUC(PBQ)	p	cohens' d
viewport time	0.5775	**0.6249**	**	1.25
dwell time	0.6225	**0.6526**	**	0.88
reading ratio	0.6382	**0.6486**		0.23
reading speed	0.4490	**0.6142**	**	3.32
direction change times	0.5904	**0.6477**	**	1.17
number of interval	0.6111	**0.6709**	**	1.33

news quality and user behavior. Note that although topic interest is found to influence the quality effect, we do not include it in the implicit feedback building approach because it is subjective and hardly collected in real applications. Figure 9(b) summarizes the proposed relations among news quality, user preferences and user behaviors, namely Preference Behavior Quality (PBQ) model.

Among the three variables, user behavior and news quality are observable by system logging or expert annotation. Based on the conditional dependence in the graphical model, we can infer user preferences when given the behavior and quality. Specifically, for PB-model, when observing a user behavior signal B (e.g. dwell time), we can infer the probability if the user likes the news ($P = 1$):

$$P(P = 1|B) = \frac{P(B|P = 1)P(P = 1)}{\sum_{i\in\{0,1\}} P(B|P = i)P(P = i)}$$

As for PBQ-model which further taking the news quality Q (low or high) into consideration, the probability is calculated by:

$$P(P = 1|B, Q) = \frac{P(B|P = 1, Q)P(P = 1, Q)}{\sum_{i\in\{0,1\}} P(B|P = i, Q)P(P = i, Q)}$$

The Satisfied-Click (with dwell time (DT) threshold =30s) can be interpreted as PB-model with $P(DT <= 30s|P = 1) = 0, P(DT > 30s|P = 0) = 0$. While most of user behavior signals conform the normal distribution (some need log-transformation at first), we assume that the probability of user behavior B given user preference or news quality follows normal distribution to get a better robustness:

$$P(B|P) \sim N(\mu_p, \sigma_p^2) \text{ or } P(B|P, Q) \sim N(\mu_{pq}, \sigma_{pq}^2)$$

Note that there is no constraint for what the behavior is, the PB-model and PBQ-model can be built by different behavior signals, like viewport time, dwell time, reading ratio, reading speed, direction change times, and the number of interval.

To investigate whether incorporating quality helps in building implicit feedback of user preferences, we compare the PB-model and PBQ-model with the performance of estimating whether a user likes a clicked news. We choose *Post-Task Preference* as the ground truth of user preference, dislike (Post-Task Preference<=3) and like (Post-Task Preference>3), because it is collected in a random order after user browsing the whole list, which eliminate the potential biases from position and interaction order.

We use 10-fold cross-validation to evaluate the model performance. The parameters of the probability model are estimated using training set in each fold, and the performance is measured by AUC in testing set. The results are shown in Table 4. Firstly, while PB-model performs best when using *reading ratio* and *dwell time*, similar with our traditional usage, the PBQ-model performs best when using a less used user behavior metric: *number of interval*. Secondly, the PBQ-model significantly outperforms the PB-model when using all the behavior signals, especially *reading ratio*. It proves that incorporating news quality does help in building implicit feedback of user preferences.

7 CAN WE IDENTIFY THE NEWS QUALITY BASED ON USER BEHAVIOR?

From above analyses, we conclude that quality is useful for estimating user preferences. Thus, how to get news quality is a valuable research topic. In this work, we labelled news quality by external assessors' annotation. It is reliable but high cost and hard to be applied in real applications. Another way is to predict the quality of news automatically. Traditional prediction models are mostly based on the content information.

In this section, We conduct two identification analysis to examine the ability of each single behavior for distinguishing the news quality.

7.1 Point-Wise Distinguishing Ability

To begin with, we evaluate the performance of using absolute value of behavior metric to identify the news quality in an interaction (a user read a piece of news), namely point-wise distinguishing ability:

Given a threshold t_b and a direction coefficient α, for an interaction i with the behavior b_i, the inferred quality of the news is calculated by:

$$\hat{q}_{\alpha,t_b}(i) = I[\alpha(b_i - t_b) > 0]$$

where α indicates whether the identification criteria is above the threshold ($\alpha = 1$) or below the threshold ($\alpha = -1$), and I is a indicator function. Then, the point-wise distinguishing ability D_{point} of a behavior b is defined as the highest accuracy that can be achieved in click interaction set C, when changing the direction α and threshold t_b:

$$D_{point}(b) = \max_{\alpha,t_b} \frac{\sum_{i\in C} I[\hat{q}_{\alpha,t_b}(i) = q_i]}{n(C)}$$

The quality labelled by the expert (EQ) is used as the ground truth (q_i). For each behavior, we evaluate its point-wise distinguishing ability based on all the click interactions in user study.

7.2 Pair-Wise Distinguishing Ability

The different reading habits lead to user biases in user behavior metrics, for example, some users usually read longer no matter the

Table 5: The performance of different behaviors in identifying quality, measured by point-wise distinguishing ability (D_{point}) and pair-wise distinguishing ability (D_{pair}). (+ Positive relative relation. - negative relative relation.)

	Expert Quality				UQ-C		UQ-T	
	D_{point}	α	D_{pair}	α	D_{pair}	α	D_{pair}	α
viewport time	0.6703	-	0.5850	-	0.5470	-	0.5300	-
dwell time	0.6751	+	0.6650	+	0.6940	+	**0.6350**	+
reading ratio	**0.7084**	+	**0.8010**	+	**0.702**	+	0.6270	+
reading speed	0.6799	+	0.6210	+	0.5240	-	0.5300	-
direction change times	0.6688	+	0.6590	+	0.6300	+	0.5650	+
number of interval	0.6719	+	0.5174	+	0.5547	+	0.5106	+

news quality. To eliminate these user biases, we design an pair-wise evaluation approach to measure the ability of a behavior metric to distinguishing the quality. The pair-wise distinguishing ability is calculated by comparing the relative relation between the behavior metric and the news quality.

Specifically, consider an example set S of news pairs [$<n_1, n_2>$,... $<n_i, n_j>$], with the behavior metric b and news quality q. We first define the relative relation between behavior and quality of a news pair $<n_i, n_j>$:

$$r_\alpha(n_i, n_j) = I\Big[\ \alpha I[b(n_i) < b(n_j)] = I[q(n_i) < q(n_j)]\ \Big]$$

where α is the direction coefficient and I is the indicator function. Then the pair-wise distinguishing ability is defined as:

$$D_{pair}(b) = \max_{\alpha \in \{-1,1\}} \frac{\sum_{<n_i, n_j> \in S} r_\alpha(n_i, n_j)}{n(S)}$$

To eliminate the user bias, we generate the evaluation pairs within users, which means the n_i and n_j is from the same user. Beside of the expert labelled quality, we also use the user perceived quality as ground truth for news quality q, including content quality (UQ-C) and title consistency (UQ-T).

7.3 Results

The results of both point-wise and pair-wise distinguishing experiments are shown in Table 5. As for the objective quality labelled experts (EQ), among these behaviors, *reading ratio* achieves the highest point-wise distinguishing ability (0.7084) with threshold $t_b = 0.74$ and direction $r = 1$. It means that whether users read more than 74% of the news content can be used as an indicator for the high quality news. *Reading ratio* also achieves the highest pair-wise distinguishing ability (0.8010). It means that if a user reading more in news i than another news j, the quality of news i is more likely higher than the quality of news j. As for the perceived quality labelled by the user, UQ-C and UQ-T the *dwell time* and *reading ratio* performs better in distinguishing pair-wise quality.

To summary, based on the proposed point-wise and pair-wise distinguishing ability measurements, we investigate whether user behaviors can be used to identify news quality. The results demonstrate that user behaviors, especially *reading ratio* and *dwell time*, have highly ability to distinguish news quality. Starting from here, the powerful behavior-based quality identification models are promoted to be developed in the future.

8 CONCLUSION

In this work, through an in-depth user experiment, we investigate the quality effects on user item-level preference and user behaviors. To begin with, we verify that the quality does affect user preferences, and further find that the quality effect varies at different phases and it is more closely related to the user's interest for the news topic.

As for the quality effects on user behaviors, we firstly show that user click behavior is significantly affected by the quality. Specifically, low-quality news attracts more clicks especially when the user has higher interest in news topic. It can be interpreted by the higher title persuasion of low-quality news. Besides, a piece of low-quality news will affect not only the click probability of current news but also the news following it. Secondly, we find that a user behaves significantly different when interacting with low-quality news. A user will read less and slowly, with fewer revisits and fewer examinations.

Based on our finding that quality affects user preferences and behaviors, we proposed the PBQ-model which incorporates quality effects into the traditional behavior-only implicit feedback. The significant improvement in estimating user preferences demonstrates that considering quality is useful for building accurate implicit feedback. Because of the usefulness of quality, how to automatically identify quality is a valuable research topic. Our study shows the possibility of using user behaviors to identify news quality, such as *reading ratio* and *dwell time*.

The study on designing the model using content and behaviors information to predict the quality is left for future work. Beside of individual-level preference, the quality effect on user list-level satisfaction is also left for further study.

ACKNOWLEDGMENTS

We thank Prof. dr. Maarten de Rijke and Prof. Fang Yang for providing very useful comments for this paper. This work is supported by Natural Science Foundation of China (Grant No. 61672311, 61532011) and The National Key Research and Development Program of China (2018YFC0831900), and is partly supported by the Tsinghua-Sogou Tiangong Institute for Intelligent Computing.

REFERENCES

[1] Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 19–26.

[2] Antti Ajanki, David R Hardoon, Samuel Kaski, Kai Puolamäki, and John Shawe-Taylor. 2009. Can eyes reveal interest? Implicit queries from gaze patterns. *User Modeling and User-Adapted Interaction* 19, 4 (2009), 307–339.

[3] Ioannis Arapakis, Mounia Lalmas, B Barla Cambazoglu, Mari-Carmen Marcos, and Joemon M Jose. 2014. User engagement in online N ews: Under the scope of sentiment, interest, affect, and gaze. *Journal of the Association for Information Science and Technology* 65, 10 (2014), 1988–2005.

[4] Nicholas J Belkin, Michael Cole, and Jingjing Liu. 2009. A model for evaluation of interactive information retrieval. In *Proceedings of the SIGIR 2009 Workshop on the Future of IR Evaluation*. 7–8.

[5] Olivier Chapelle and Ya Zhang. 2009. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th international conference on World wide web*. ACM, 1–10.

[6] Mark Claypool, Phong Le, Makoto Wased, and David Brown. 2001. Implicit interest indicators. In *Proceedings of the 6th international conference on Intelligent user interfaces*. ACM, 33–40.

[7] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research* 7, Jan (2006), 1–30.

[8] Georges E Dupret and Benjamin Piwowarski. 2008. A user browsing model to predict search engine click data from past observations.. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 331–338.

[9] Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, and Thomas White. 2005. Evaluating implicit measures to improve web search. *ACM Transactions on Information Systems (TOIS)* 23, 2 (2005), 147–168.

[10] Fan Guo, Chao Liu, and Yi Min Wang. 2009. Efficient multiple-click models in web search. In *Proceedings of the second acm international conference on web search and data mining*. ACM, 124–131.

[11] Ahmed Hassan. 2012. A semi-supervised approach to modeling web search satisfaction. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 275–284.

[12] Ahmed Hassan and Ryen W White. 2013. Personalized models of search satisfaction. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM, 2009–2018.

[13] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee, 263–272.

[14] Jeff Huang and Abdigani Diriye. 2012. Web user interaction mining from touch-enabled mobile devices. In *HCIR workshop*.

[15] Jiepu Jiang, Daqing He, and James Allan. 2017. Comparing In Situ and Multidimensional Relevance Judgments. (2017).

[16] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2017. Accurately interpreting clickthrough data as implicit feedback. In *ACM SIGIR Forum*, Vol. 51. Acm, 4–11.

[17] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. 2007. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)* 25, 2 (2007), 7.

[18] Diane Kelly and Nicholas J Belkin. 2004. Display time as implicit feedback: understanding task effects. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 377–384.

[19] Diane Kelly and Jaime Teevan. 2003. Implicit feedback for inferring user preference: a bibliography. In *Acm Sigir Forum*, Vol. 37. ACM, 18–28.

[20] Youngho Kim, Ahmed Hassan, Ryen W White, and Imed Zitouni. 2014. Comparing client and server dwell time estimates for click-level satisfaction prediction. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 895–898.

[21] Youngho Kim, Ahmed Hassan, Ryen W White, and Imed Zitouni. 2014. Modeling dwell time to predict click-level satisfaction. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 193–202.

[22] Dmitry Lagun, Chih-Hung Hsieh, Dale Webster, and Vidhya Navalpakkam. 2014. Towards better measurement of attention and satisfaction in mobile search. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 113–122.

[23] Dmitry Lagun, Donal McMahon, and Vidhya Navalpakkam. 2016. Understanding mobile searcher attention with rich ad formats. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 599–608.

[24] Gael Lederrey and Robert West. 2018. When Sheep Shop: Measuring Herding Effects in Product Ratings with Natural Experiments. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 793–802.

[25] Yixuan Li, Pingmei Xu, Dmitry Lagun, and Vidhya Navalpakkam. 2017. Towards Measuring and Inferring User Interest from Gaze. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 525–533.

[26] Jimmy Lin, Salman Mohammed, Royal Sequiera, and Luchen Tan. 2018. Update Delivery Mechanisms for Prospective Information Needs: An Analysis of Attention in Mobile Users. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. ACM, New York, NY, USA, 785–794. https://doi.org/10.1145/3209978.3210018

[27] Zeyang Liu, Jiaxin Mao, Chao Wang, Qingyao Ai, Yiqun Liu, and Jian-Yun Nie. 2017. Enhancing click models with mouse movement information. *Information Retrieval Journal* 20, 1 (2017), 53–80.

[28] Hongyu Lu, Min Zhang, and Ma Shaoping. 2018. Between Clicks and Satisfaction: Study on Multi-Phase User Preferences and Satisfaction for Online News Reading. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM.

[29] Jiaxin Mao, Yiqun Liu, Ke Zhou, Jian Yun Nie, Jingtao Song, Min Zhang, Hengliang Luo, Hengliang Luo, and Hengliang Luo. 2016. When does Relevance Mean Usefulness and User Satisfaction in Web Search?. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*. 463–472.

[30] Benjamin M Marlin and Richard S Zemel. 2009. Collaborative prediction and ranking with non-random missing data. In *Proceedings of the third ACM conference on Recommender systems*. ACM, 5–12.

[31] Ben Miroglio, David Zeber, Jofish Kaye, and Rebecca Weiss. 2018. The Effect of Ad Blocking on User Engagement with the Web. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 813–821.

[32] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 502–511.

[33] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science* 359, 6380 (2018), 1146–1151.

[34] Chao Wang, Yiqun Liu, Min Zhang, Shaoping Ma, Meihong Zheng, Jing Qian, and Kuo Zhang. 2013. Incorporating vertical results into search click models. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 503–512.

[35] Ryen W White and Diane Kelly. 2006. A study on the effects of personalization and task information on implicit feedback performance. In *Proceedings of the 15th ACM international conference on Information and knowledge management*. ACM, 297–306.

[36] Peifeng Yin, Ping Luo, Wang-Chien Lee, and Min Wang. 2013. Silence is also evidence: interpreting dwell time for recommendation from psychological perspective. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 989–997.

[37] Yisong Yue, Rajan Patel, and Hein Roehrig. 2010. Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *Proceedings of the 19th international conference on World wide web*. ACM, 1011–1018.

[38] Xiaoying Zhang, Junzhou Zhao, and John Lui. 2017. Modeling the assimilation-contrast effects in online product rating systems: Debiasing and recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 98–106.

What We Vote for? Answer Selection from User Expertise View in Community Question Answering

Shanshan Lyu
Yongqing Wang
Wentao Ouyang
Huawei Shen
Xueqi Cheng
CAS Key Laboratory of Network Data Science and Technology,
Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China
University of Chinese Academy of Sciences, Beijing, China
{lvshanshan,wangyongqing,ouyangwt,shenhuawei,cxq}@ict.ac.cn

ABSTRACT

Answer selection is an important problem in community question answering (CQA), as it enables the distilling of reliable information and knowledge. Most existing approaches tackle this problem as a text matching task. However, they ignore the influence of the community in voting the best answers. Answer quality is highly correlated with semantic relevance and user expertise in CQA. In this paper, we formalize the answer selection problem from the user expertise view, considering both the semantic relevance in question-answer pair and user expertise in question-user pair. We design a novel matching function, explicitly modeling the influence of user expertise in community acceptance. Moreover, we introduce latent user vectors into the representation learning of answer, capturing the implicit topic interests in learned user vectors. Extensive experiments on two datasets from real world CQA sites demonstrate that our model outperforms state-of-the-art approaches for answer selection in CQA. Furthermore, the user representations learned by our model provide us a quantitative way to understand both the authority and topic-sensitive interests of users.

CCS CONCEPTS

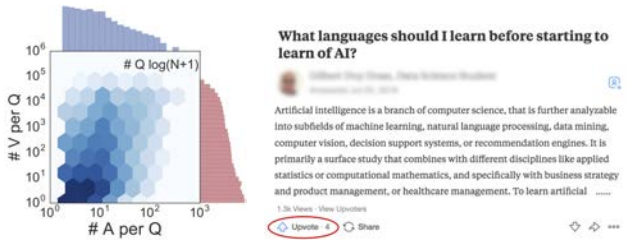
• **Information systems** → **Collaborative and social computing systems and tools**; • **Computing methodologies** → *Artificial intelligence.*

KEYWORDS

User expertise, community question answering, answer selection

ACM Reference Format:
Shanshan Lyu, Yongqing Wang, Wentao Ouyang, Huawei Shen, and Xueqi Cheng. 2019. What We Vote for? Answer Selection from User Expertise View in Community Question Answering. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313510

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313510



(a) Votes distribution. (b) Example of question, answer, and vote mechanism in CQA.

Figure 1: Illustration of vote mechanism in CQA.

1 INTRODUCTION

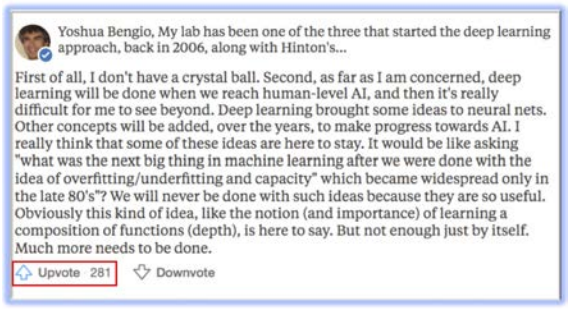
Community question answering (CQA) sites, such as Quora[1] and Zhihu[2], are Internet-based crowdsourcing services which enable users to post questions and seek answers from other users. The success of CQA has attracted much research attention in recent years [14, 26, 28, 31, 34, 38–41]. Most users would prefer to put forward open questions without strict standard answers in CQA sites. By this means, the criterion of answer quality is difficult to be unified in CQA systems. For example, the answers would be various from different users for the question "*What languages should I learn before starting to learn of AI?*". Thus, the voting mechanism is implemented in CQA sites, where the answer quality is indicated by the votes from the whole online communities. However, the answers elected by communities have low reliability due to the scarce votes in real applications. We take statistics in Quora and draw the correlation on the number of votes per question (#V per Q) and the number of answers per question (#A per Q) in Figure 1. As shown in the figure, most questions with a few answers receive less than 10 votes correspondingly. Therefore, automatic answer selection, which aims to rank the answers in descending order of quality, is a crucial task in CQA.

Many approaches have been proposed for answer selection, most of which tackle this problem as a text matching task. Early approaches are based on the form of exact string matches for n-grams,

[1]https://www.quora.com/
[2]https://www.zhihu.com/



(a) An common answer of the question.



(b) The best answer of the question.

Figure 2: Example of user expertise. Users with high expertise usually provide high-quality answers.

as such they fail to detect similar meaning conveyed by synonymous words [6, 7, 12, 19, 20]. Some approaches focus on constructing features from textual clues [36, 46], such as lexical and syntactic features. However, these methods has limited performance as the high computation cost for capturing context features. Recently, deep architectures are proposed to learn distributed representations for words and sentences. These approaches encode the questions and answers from a high-dimensional and sparse representation into low-dimensional and continuous dense vectors according to comprehensive context efficiently, e.g. convolutional neural networks (CNN) or recurrent neural networks (RNN). Based on the learned representations, semantic similarity [31, 33, 38] or matching [17, 24, 34, 37] functions can be learned from an end-to-end learning strategy.

However, traditional solutions of answer selection on QA systems cannot be directly applied in CQA as these methods ignore the influence of user expertise accepted by the community. In CQA systems, the answer rank is highly correlated with the expertise of users. We take an example of question in Quora, showing the difference between two answers whose users have different expertise in community. The open question and two corresponding answers are described in Figure 2(a) and 2(b). In the example, the two users give the answers from different aspects on "*What is the next big thing in machine learning after we are done with deep learning?*". The two answers are both highly relative to the question. However, the answer by Yoshua Bengio receives 281 votes which is much higher than the other one. The phenomenon is approved in [5, 49]. According to the literatures, the main reason is that Yoshua Bengio is a well-known expert in the community of deep learning and machine learning, and thus his answer tends to be voted by others.

However, user expertise is still questionable as its definition and quantitation during the progress of answer selection. Thus, we propose a user-expertise specific learning framework to model the dynamics of answer selection in CQA. The goal of our work tries to learn user expertise of each user by latent factors so as to better capture the rank of answers. Firstly, our proposed method defines latent user vectors in terms of answer ranking, explicitly modeling the relevance between question-user pair. A novel score function is introduced to evaluate both the relevance on question-answer and corresponding question-user pairs accepted by communities. Then, we introduce latent user vectors into the representation learning of answer, capturing the implicit topic interests in learned user vectors. Our proposed method encodes latent user vectors into the answer representation by hierarchical attention mechanisms within Long-Short Term Memory (LSTM) [16] network, which is more informative to user-specific interests. Experimental results on two datasets from real world CQA sites demonstrate that our proposed method outperforms other state-of-the-art approaches for answer selection. Moreover, the extensive experimental results prove that the learned user vectors can indicate the authority accepted by communities and capture the user interests in topic level, showing the great potentials on understanding user expertise in CQA.

Our contributions in this paper are summarized as follows:

- We propose a novel score function to directly evaluate the comprehensive relevance on both question-answer and question-user pairs, explicitly modeling the influence of user expertise in community acceptance.
- Our proposed model encodes latent user expertise into answer representation by hierarchical attention mechanisms, capturing implicit topic interests of each user.
- We conduct extensive experiments on two real-world datasets to verify the effectiveness of our model and compare it to other state-of-the-art approaches for answer selection in CQA. Moreover, the user representations learned by our proposed method provide us a quantitative way to understand topic-related user expertise in CQA. Besides, our implementation codes are open to public[3].

The remainder of this paper is organized as follows. Section 2 reviews the prior work related to answer selection. Section 3 formally defines the problem of answer selection. Section 4 introduces our proposed model in details. Section 5 presents the experimental results on two real-world datasets. Finally we conclude our work in Section 6.

2 RELATED WORK

In this section, we first review the prior work in community question answering. Then, we introduce the attention-based neural networks.

[3]https://github.com/Sunshine1007472173/UEAN

2.1 Community Question Answering

Early works in information retrieval, including language model (LM) [6, 7, 12, 19], and Okapi model (Okapi) [20], can also be used for answer selection. These approaches are purely based on the form of exact string matches for n-grams, as such they fail to detect similar meaning conveyed by synonymous words. A retrieval model was proposed in [44] that combined a translation-based language model for the question part with a query likelihood approach for the answer part. The authors formulated answer sentence selection as a semantic matching problem with a latent word-alignment structure in [46]. The authors studied the use of syntactic and semantic structures obtained with shallow and deeper syntactic parsers for the answer passage reranking task in [36].

With the rapid development of distributed word representations [18, 23, 27] and deep neural networks, many approaches have been proposed for answer selection. Convolutional neural network (CNN) was introduced by [17] to model the sentence matching in question answering. In [33], the authors demonstrated a unified architecture that trains a convolutional neural network together with a multi-layer perceptron to rank question-answer pairs. The authors uses a stacked bidirectional LSTM network to sequentially read words from question and answer sentences, and then outputs their relevance score [38]. Grid-wise similarity matrices within neural architectures is also adopted to model the complicated matching relations between question and answer [24, 34, 37]. The neural tensor network architecture was proposed to model the diverse relationships between question and answer in [31, 35].

However, most of the existing works merely model the semantic relevance between question-answer pairs and ignore the main characteristics in CQA, i.e., **expertise of users**. Semantic relevance and user expertise are jointly considered in our model. Furthermore, our model encodes user expertise information in the answer representation by attention mechanisms to focus on crucial segments of answers hierarchically. As a result, more important answer content and the expertise of users is captured. Recently, the authors proposed an asymmetric multi-faceted ranking network learning framework considering the answerers' authority in [48]. Nevertheless, they just model the user expertise in the score function and do not consider the influence of user expertise to the answer.

2.2 Attention-based Neural Networks

The attention mechanism has shown good performance in many tasks [2, 3, 8–11, 15, 25, 29, 32, 42, 43]. Specifically in machine translation [3, 25], given a sentence in the original language, the attention mechanism aims to adjust the weights of input words to predict the words in the target language. In [45], the authors proposed hierarchical attention networks for document classification. However, the answer selection task in CQA is different from the language translation and the document classification tasks. Moreover, our way of introducing user-aware hierarchical attentions is not explored in [3, 25, 45]. Recently, the authors in [47] proposed to first learn representations of question and answer by neural network architectures and then apply *one-level* attentive pooling network on an interaction tensor. Our approach differs in that our attention mechanisms are used for constructing the answer representation directly rather than for post-processing an interaction tensor. Moreover, our model jointly capturing the user expertise and semantic relevance, while [47] just focus on semantic matching.

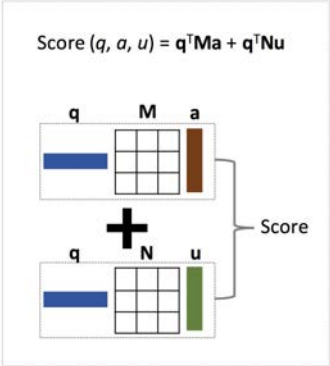


Figure 3: The score function of our model, jointly exploiting the semantic relevance between question-answer pairs and the user expertise to the given question.

3 PROBLEM FORMULATION

Before diving into detail of our model, we first describe symbols used in the answer selection problem. Given a question q, a set of its candidate answers $A_q = \{a_{q,1}, a_{q,2}, \ldots, a_{q,N}\}$ and users who gave the answers $U = \{u_1, u_2, \ldots, u_N\}$, answer selection aims to rank the answers in A_q in descending order of quality. We let $(s^a_1, s^a_2, \ldots, s^a_{N_a})$ represent the sequence of sentences in an answer a. Meanwhile, the word sequences in a sentence s is denoted as $(w^s_1, w^s_2, \ldots, w^s_{N_s})$. For ease of presentation, scalars, vectors and matrices are denoted by lower-case letters, boldface lower-case letters and boldface capital letters in this paper.

Since a good answer should well address a question semantically, the answer selection problem can be defined into an interactive process between a question-answer pair. In this way, the score function can be formalized as follows:

$$s(q_i, a_{q_i,j}) = \sigma(f(q_i), g(a_{q_i,j})), \tag{1}$$

where the function f and g map the question q_i and answer $a_{q_i,j}$ into a common vector space. The function σ measures matching degree between each question-answer pair. Generally, the score function σ could be in the form of cosine similarity, bilinear function or tensors [17, 31, 47]. For briefly describing the models, we use vector $\mathbf{q}_i = f(q_i)$ and $\mathbf{a}_{q_i,j} = g(a_{q_i,j})$ as the representations of mapping results in the rest of the paper.

For the question q_i, given two answers $a_{q_i,j}$ and $a_{q_i,m}$ where answer $a_{q_i,j}$ has a higher quality than $a_{q_i,m}$, we aim to learn the score function that the following inequality holds for every tuple $(q_i, a_{q_i,j}, a_{q_i,m})$: $s(q_i, a_{q_i,j}) > s(q_i, a_{q_i,m})$. Then, we can rank the answers for a question in descending order of quality according to the score function.

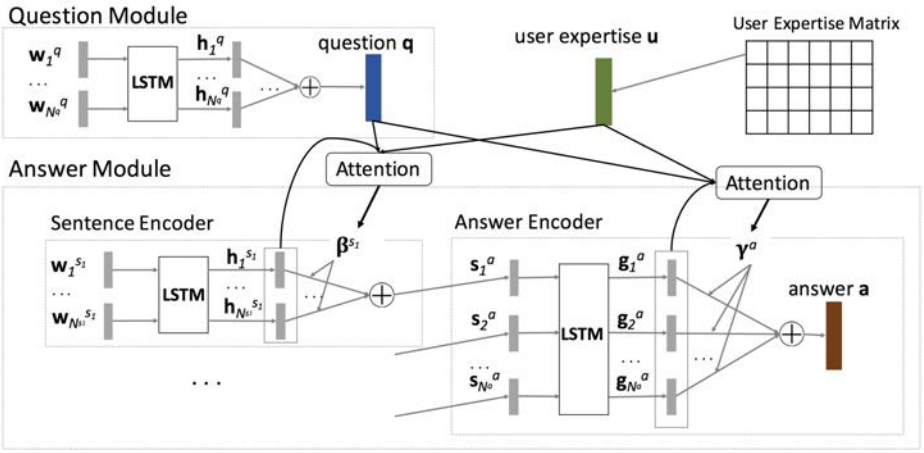


Figure 4: The Overview of our framework. Our model encodes user expertise in the answer representation, contributing to capture the implicit topic interests in learned user vectors.

4 UEAN FOR ANSWER SELECTION

In this section, we present User-Expertise aware Attention Networks (UEAN) model in detail whose framework is shown in Figure 4. We first explain our specific-designed user-expertise aware matching function in Section 4.1, explicitly modeling the influence of user expertise in community acceptance. Then, we discuss how to obtain answer and question representation in Section 4.2 and 4.3 respectively. Finally, we elaborate the learning process of our model in Section 4.4.

4.1 User Expertise Aware Matching Function

Most of the existing works merely model the semantic relevance between question-answer pairs and ignore the expertise of users when modeling the answer selection in CQA. Besides the semantic relevance, users with high expertise contribute to achieve acceptance from the community. Thus, we design a novel score function jointly capturing the semantic relevance and the expertise of users (shown in Figure 3). We divide our matching function into two parts: the semantic matching part and expertise part.

Let a tuple $(q_i, a_{q_i,j}, u_k)$ denote an answer $a_{q_i,j}$ provided by u_k for question q_i. The score function for a tuple $(q_i, a_{q_i,j}, u_k)$ is designed as follows:

$$s(q_i, a_{q_i,j}, u_k) = \mathbf{q}_i^T \mathbf{M} \mathbf{a}_{q_i,j} + \mathbf{q}_i^T \mathbf{N} \mathbf{u}_k. \tag{2}$$

where $\mathbf{q}_i, \mathbf{a}_{q_i,j}$, and $\mathbf{u}_k$ is the representation of the question q_i, answer $a_{q_i,j}$, and user expertise u_k respectively. The representation of answer $a_{q_i,j}$ and the question q_i are generated from answer and question in Section 4.2 and 4.3. The representation of user expertise $\mathbf{u}_k$ is a latent vector learned during training. In Eq. (2), $\mathbf{q}_i^T \mathbf{M} \mathbf{a}_{q_i,j}$ represents the semantic matching score between answer $a_{q_i,j}$ and question q_i, where $\mathbf{M}$ is a matrix to transform the question representation into the space of answer representation. $\mathbf{q}_i^T \mathbf{N} \mathbf{u}_k$ represents the expertise score of user u_k to the question q_i, where $\mathbf{N}$ is a matrix to transform the question representation into the representation space of user expertise. In this way, our matching function takes both the interactions of question-user pair and question-answer pair into consideration.

4.2 User Expertise Aware Answer Module

Answer module aims at generating representations for input answers. For the purpose of capturing the implicit topic interests in learned user vectors, we introduce latent user vectors into the representation learning of answers. We encode user expertise in the answer representation by hierarchical attention mechanisms within LSTM networks, contributing to capture content more informative to user-specific interests.

4.2.1 ***Sentence encoder.*** For the t-th sentence s_t with N_{s_t} words in an answer, we first represent each word $w_j^{s_t}$ as a K_w-dimensional vector $\mathbf{w}_j^{s_t}$ by pre-trained word embeddings [30]. We then adopt the widely exploited LSTM enabling to deal with variable-length sequence input, to encode a sentence into a fixed-length vector. We utilize the LSTM in [21] to take each word vector $\mathbf{w}_j^{s_t}$ as input and updates the corresponding hidden state $\mathbf{h}_j^{s_t}$ as follows:

$$\mathbf{h}_j^{s_t} = LSTM(\mathbf{h}_{j-1}^{s_t}, \mathbf{w}_j^{s_t}). \tag{3}$$

In Eq. (3), the hidden vector $\mathbf{h}_j^{s_t}$ is learned from the previous hidden states and the j-th input word.

In a sentence, some words are relevant to the semantic information of user expertise. For instance, the word "Java" or "Python" can capture the expertise of the user who is skilled in "Programming Language". The word "LSTM" or "CNN" can capture the the expertise of the user who is skilled in "Deep Learning". Hence, we introduce the latent user expertise into the representation learning of answer sentence to capture the implicit topic interests in learned user vectors. Meanwhile, we also introduce the question information to capture the words relevant to the question.

We encode the user expertise in answer representation by attention mechanisms [3]. We first concatenate the user expertise vector $\mathbf{u}$, question vector $\mathbf{q}$, and hidden vector $\mathbf{h}_j^t$ of answer word. Then we select $tanh$ as the activation function and calculate the weights

by a multi-layer perceptron (MLP) [4]. Finally, the attention weight $\beta_j^{s_t}$ is generated as follows:

$$e_j^{s_t} = \mathbf{v}_w^T \tanh(\mathbf{W}_w[\mathbf{u}; \mathbf{q}; \mathbf{h}_j^{s_t}] + \mathbf{b}_w),$$
$$\beta_j^{s_t} = softmax(e_j^{s_t}), \quad (4)$$

where $softmax(x_i) = \frac{x_i}{\sum_j x_j}$ is taken to normalize the weights, and $\mathbf{W}_w$, $\mathbf{b}_w$ and $\mathbf{v}_w$ are parameters in word-level attention that to be learned. Then the sentence representation $\mathbf{s}_t$ can be calculated based on the weights obtained from Eq. (4) and the hidden states from $\mathbf{h}_1^{s_t}$ to $\mathbf{h}_{N_{s_t}}^{s_t}$ as follows:

$$\mathbf{s}_t = \sum_{j=1}^{N_{s_t}} \beta_j^{s_t} \mathbf{h}_j^{s_t}. \quad (5)$$

4.2.2 ***Answer encoder.*** Assume that an answer a is comprised of a sequence of N_a sentences, i.e. $(s_1^a, \ldots, s_{N_a}^a)$, each of which is encoded as a K_s-dimensional vector $\mathbf{s}_t^a$ by Eq. (5). As the sentences of an answer can be regarded as a sequence with variable length, we can obtain the answer representation in a similar way using the LSTM at sentence-level. The hidden state $\mathbf{g}_t^a$ is calculated as follows:

$$\mathbf{g}_t^a = LSTM(\mathbf{g}_{t-1}^a, \mathbf{s}_t^a), \quad (6)$$

where the updating of hidden state is depended on the sentence vector $\mathbf{s}_t^a$ and the hidden state $\mathbf{g}_{t-1}^a$ in each iteration.

Similar to the words, some sentences can capture the implicit topic interests of learned user vectors. Hence, we encode user expertise and question content in the answer representation by a sentence-level attention mechanism. In this way, more informative sentences will be emphasized and less informative sentences will be weakened. Thus, we can estimate weights from attention layer according to the latent factor of users and the representation of question. Then each attention weight γ_t^a for the sentence s_t^a is calculated through attention mechanism by

$$e_t^a = \mathbf{v}_s^T \tanh(\mathbf{W}_s[\mathbf{u}; \mathbf{q}; \mathbf{g}_t^a] + \mathbf{b}_s),$$
$$\gamma_t^a = softmax(e_t^a), \quad (7)$$

where $\mathbf{W}_s$, $\mathbf{b}_s$ and $\mathbf{v}_s$ are parameters in sentence-level attention that to be learned. The overall answer representation vector $\mathbf{a}$ is calculated by aggregating all hidden vectors from sentences weighted by their corresponding attention weights as

$$\mathbf{a} = \sum_{t=1}^{N_a} \gamma_t^a \mathbf{g}_t^a. \quad (8)$$

4.3 Question Module

For a question q with a sequence of N_q words, we first represent each word w_i^q as a K_w-dimensional vector $\mathbf{w}_i^q$ by pre-trained word embeddings [30]. We then employ the LSTM by taking each word vector $\mathbf{w}_i^q$ as input and updating the corresponding hidden state $\mathbf{h}_i^q$ as follows:

$$\mathbf{h}_i^q = LSTM(\mathbf{h}_{i-1}^q, \mathbf{w}_i^q). \quad (9)$$

We observe that each word contributes differently to the representation of a question. For example, in a question "*Where is the theater?*", "where" and "theater" play more critical roles in summarizing this question. Therefore, we employ the attention mechanism to learn the question representation by focusing on more informative words. The attention weight α_i^q is based on the hidden state $\mathbf{h}_i^q$ of word w_i^q. Afterwards, the question representation $\mathbf{q}$ is generated as the weighted sum of hidden vectors from words according to their corresponding importance as follows:

$$\alpha_i^q = softmax(\mathbf{m}^T \mathbf{h}_i^q), \quad (10)$$

$$\mathbf{q} = \sum_{i=1}^{N_q} \alpha_i^q \mathbf{h}_i^q, \quad (11)$$

where $\mathbf{m}$ is a parameter to be learned.

4.4 Learning Process

We employ a discriminative training strategy with a large margin objective to train our model. Given each pair of tuples $(q_i, a_{q_i,j}, u_k)$ and $(q_i, a_{q_i,m}, u_n)$ related to question q_i in training set, the relevance score of each answer can be calculated by the matching function in Eq. (2). Suppose that the quality of answer $a_{q_i,m}$ is higher than the quality of answer $a_{q_i,j}$, we aim to maximize the following ranking-based loss under observed ranking order as follows,

$$L = \sum_{(i,j,k) \prec (i,m,n)} \max(0, c + s(q_i, a_{q_i,m}, u_n) - s(q_i, a_{q_i,j}, u_k)),$$

where $0 < c < 1$ is a slack variable. During the training process, question and answer share the pre-trained word embeddings. All the weight parameters are randomly initialized and trained by the back propagation of training loss. Besides, the ground truth of ranking order is inferred by votes in real CQA sites.

5 EXPERIMENT

In this section, we conduct experiments to evaluate the effectiveness of our model and compare it with several state-of-the-art methods for answer selection, showing that our proposed model outperforms other state-of-the-art approaches. Moreover, we investigate the learned representation of user expertise, showing the potentials on understanding user expertise in CQA.

5.1 Experimental Setups

5.1.1 ***Datasets.*** To demonstrate the effectiveness of our approach in real application, the following two datasets are adopted in our experiment.

- **Quora dataset**. We collected a large dataset from Quora which is one of the most popular CQA sites. We randomly crawled 800,772 questions and 3,523,485 answers posted from January 2016 and July 2017. The votes and users related to these answers are also collected. For statistical reliability, we remove the questions if the maximum number of votes received by the following answers is smaller than 10. We also remove questions with answers less than 2. Finally, our dataset contains 76,158 questions, 467,681 answers, 31,802,631 votes, and 46,470 users.
- **Stack Overflow dataset**. The dataset is available from StackOverflow[4] in public, containing 214,803 questions and 586,725

[4]https://stackoverflow.com/

answers. We preprocess the dataset following the similar strategy used in Quora dataset. After preprocessing, the dataset contains 19,944 questions, 67,011 answers and 6,340 users.

The two datasets are both randomly split into three subsets without overlapping for the purpose of training, validation and testing in the following experiments. The proportion of training, validation and testing dataset is equal to 80%, 10% and 10%. We also vary the size of training dataset (40%, 60% and 80%) in order to validate the robustness of our proposed methods in different experimental setting.

5.1.2 ***Evaluation Metric.*** We evaluate the performance of our model using three widely-used ranking metrics, which are Precision@1, Mean Reciprocal Rank (MRR), and normalized Discounted Cumulative Gain (nDCG). The three metrics are formulated as follows,

- *Precision@1* takes into account the position of relevant answer at top 1, calculated by

$$Precision@1 = \frac{|\{q \in Q | rank'_{best} = 1\}|}{|Q|},$$

  where $|Q|$ is the number of questions in test dataset and $rank'_{best}$ is the predicted position of the best answer given by algorithm.
- *MRR* is the average of the reciprocal ranks corresponding to the most relevant answer in questions, calculated by

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank'^{i}_{best}},$$

  where $rank'^{i}_{best}$ is the position of the best answer in the i-th question given by algorithm.
- *NDCG* is a measure of ranking quality. As to question q, NDCG is calculated by

$$nDCG = \frac{DCG}{IDCG},$$
$$\text{where } DCG = rel_1 + \sum_{i=2}^{|A_q|} \frac{rel_i}{\log_2 i},$$

  where IDCG is ideal discounted cumulative gain, $|A_q|$ is the number of answers in question q and rel_i is the relevance between question and answer at position i. We report the average of all the nDCG values in all test questions.

The larger results received by the above evaluation metrics means the better performance of the algorithms in answer selection.

5.1.3 ***Comparative Methods.*** We compare our model to 6 typical state-of-the-art methods which are briefly described below.

- **Okapi BM25** [20] is a bag-of-words retrieval model used in information retrieval. We use the model to rank candidate answers based on the question terms appearing in each answer.
- **TransLM** [44] is a retrieval model that combines a translation-based language model for the question part with a query likelihood approach for the answer part. We use TransLM to calculate the similarity between every question-answer pair and rank candidate answers based on the similarity.
- **Doc2Vec** [23] is an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts. We first use Doc2Vec to learn representations of questions and answers. The matching score of a question-answer pair is calculated with a MLP with the representations of the question and answer as input.
- **CNTN** [31] is a convolutional neural tensor network architecture to encode the question and answer in semantic space and model their interactions with a tensor layer.
- **AI-CNN** [47] first learns representations of questions and answers by CNN and then applies one-level attentive pooling network on an interaction tensor of a question-answer pair.
- **AMRNL** [48] learns representations of questions and answers by deep recurrent neural networks and ranks the answers by semantic relevance and the answers' authority.

Table 1: The Precision@1 of all methods on Quora dataset with different settings of training data size.

Method	Precision@1 (%)		
	40%	60%	80%
Okapi BM25	26.71	26.71	26.71
TransLM	29.10	29.63	30.24
Doc2Vec	31.74	32.40	32.42
CNTN	35.95	36.02	36.17
AI-CNN	33.95	34.52	35.03
AMRNL	59.10	59.19	61.14
OURS	**61.63**	**63.38**	**64.54**

Table 2: The MRR of all methods on Quora dataset with different settings of training data size.

Method	MRR (%)		
	40%	60%	80%
Okapi BM25	51.23	51.23	51.23
TransLM	52.23	52.61	53.12
Doc2Vec	53.82	53.99	54.13
CNTN	57.63	57.84	58.20
AI-CNN	56.38	56.63	57.07
AMRNL	74.23	74.48	75.64
OURS	**76.47**	**77.63**	**78.29**

5.1.4 ***Parameter settings.*** Okapi BM25, TransLM, Doc2Vec and AMRNL are implemented according to the source code provided by authors. The baselines of CNTN and AI-CNN are carefully implemented with hyperparameter tuning according to the original papers. In our proposed method, we use GloVe [30] pre-trained word embeddings. The dimension of word embedding is 50. The questions and answers are tokenized and lemmatized using NLTK[5]. Tokens that did not appear in the pre-trained word embeddings are initialized randomly. The dimension of LSTM network embedding in our experiment is 128 and the dimension of parameter $\mathbf{v}$ for

[5]https://www.nltk.org/

Table 3: The nDCG of all methods on Quora dataset with different settings of training data size.

Method	nDCG (%)		
	40%	60%	80%
Okapi BM25	76.37	76.37	76.37
TransLM	76.11	76.35	76.72
Doc2Vec	76.40	76.55	76.57
CNTN	79.80	80.18	80.66
AI-CNN	79.22	79.36	79.60
AMRNL	89.52	89.88	90.22
OURS	**91.31**	**91.91**	**92.04**

attention mechanism in Eq. (4) and Eq. (7) is 64. Besides, the dimension of user representation is 128. We set the margin $c = 0.5$ in the ranking loss. At last, we choose Adam [22] as the optimization strategy to update parameters with initial learning rate 0.001.

5.2 Prediction Performance on Answer Selection

In this section, we compare the performance of our model with the chosen state-of-the-art methods in order to validate the effectiveness of our proposed method in predicting the rank of best answers. Table 1, 2 and 3 present the experimental results on Quora dataset in terms of Precision@1, MRR and nDCG respectively. Table 4, 5 and 6 present the experimental results on Stack Overflow dataset. We summarize our observations as follows:

- Methods based on distributed representations all perform better than Okapi BM25. It proves that word embeddings can capture more semantic information than traditional bag-of-words approaches.
- CNTN and AI-CNN outperform Doc2Vec and TransLM, indicating that it is effective to model fine-granularity interaction between a pair of question and answer.
- Approaches with consideration of user expertise, i.e., AMRNL and our proposed method, perform better than these methods that merely consider semantic relevance, demonstrating that modeling the influence of user expertise is helpful for answer selection in CQA.
- In addition, our proposed method is better than AMRNL, proving the effectiveness of modeling user expertise in both explicit and implicit ways for answer selection.
- At last, our proposed method significantly outperform the baselines in all cases. The results show that our proposed method is well suitable to answer selection problem in CQA.

In order to validate the effectiveness of explicitly modeling the relevance of question-user pair in our model, we introduce variants of our model (with user expertise and without) and compare the performance. Table 7 presents the experimental results.It is observed that our model with user expertise significantly outperform our model without user expertise. Furthermore, we conduct case studies to analyze the advantages of our proposed model in answer selection. We find out that short answers prefer to receive lower relevant score with merely modeling question-answer relevance.

Table 4: The Precision@1 of all methods on Stack Overflow dataset with different settings of training data size.

Method	Precision@1 (%)		
	40%	60%	80%
Okapi BM25	39.72	39.72	39.72
TransLM	42.78	43.38	44.03
Doc2Vec	40.04	41.20	41.36
CNTN	46.70	47.57	47.92
AI-CNN	44.98	45.99	47.62
AMRNL	50.15	50.41	50.71
OURS	**53.90**	**55.46**	**56.56**

Table 5: The MRR of all methods on Stack Overflow dataset with different settings of training data size.

Method	MRR (%)		
	40%	60%	80%
Okapi BM25	65.67	65.67	65.67
TransLM	66.96	66.98	67.52
Doc2Vec	66.30	66.47	66.48
CNTN	69.45	69.87	69.98
AI-CNN	67.91	68.69	69.50
AMRNL	71.08	71.16	71.22
OURS	**72.73**	**73.89**	**74.22**

Table 6: The nDCG of all methods on Stack Overflow dataset with different settings of training data size.

Method	nDCG (%)		
	40%	60%	80%
Okapi BM25	89.36	89.36	89.36
TransLM	90.28	90.71	90.82
Doc2Vec	88.95	89.52	89.59
CNTN	91.41	91.76	91.77
AI-CNN	90.58	91.13	91.13
AMRNL	91.55	91.95	92.10
OURS	**91.70**	**92.34**	**92.79**

It is because that short answers usually have less semantic information than long answers. However, *short answer would be the best answer accepted by communities*. In the question "What are the similarities between GANs and reinforcement learning?" on Quora[6], the communities vote the shortest answer to be the best answer as the responser is "Yoshua Bengio". In CQA, the acceptance of community pays an important role in answer selection. The explicit modeling on the relevance of question-user pair can well model the scenario when the defined "user expertise" pays more important role in answer selection on CQA systems.

Next we will investigate the learning mechanism of user expertise in detail and give both qualitative and quantitative analyses on learned user expertise.

[6]https://www.quora.com/What-are-the-similarities-between-GANs-and-reinforcement-learning

Table 7: The performance (%) of our model (with user expertise and without) on datasets with 80% of the data for training. ("UE" - user expertise.)

Method	Quora dataset			Stack Overflow dataset		
	P@1	MRR	nDCG	P@1	MRR	nDCG
Without UE	36.84	58.02	79.84	48.58	69.99	91.17
With UE	**64.54**	**78.29**	**92.04**	**56.56**	**74.22**	**92.79**

5.3 What is Learned in User Vectors

In this section, we give both qualitative and quantitative analyses on learned user embedding vectors and illustrate the potentials of learned user expertise on presenting community acceptance and topic interests of users.

Authority-correlated user expertise. Average number of votes is positively related to authority accepted by the community in CQA platform, which is an important indicator for user expertise in long term. Therefore, we use the average number of votes to validate whether the learned user representations has positive correlation to the authority in practice. We randomly choose 3,000 users in Quora and plot their embeddings in Figure 5(a) whose dimensionality is reduced by PCA [13] from the original high dimensional space (128) to a visualization space (2). Each user is colored by their true authority, calculated by their average votes per question in Quora. In the figure, the users with different authority can be well separated by the user representations. We can differentiate three clusters with high, median and low expertise from the sampled users. We randomly choose several users at each cluster and show their profile. The blue/purple block on the right side can be regarded as the community with high reputation, where users with high votes and many followers are located, such as Balaji Viswanathan, Richard Muller, Jake Williams, Alon Ami and Dave Consiglio. The green block on the middle is the community with median authority, such as John L Ware and Eric Reid are located. The red block on the left side is the community with low expertise, where the users with few followers and zero vote are located. According to the experimental results, the learned user vectors have potentials on presenting authority. Furthermore, we reduce the dimensionality of user vectors from 128 to 1 by PCA and plot it with the average votes of users in Figure 5(b). It is observed that the user vectors and votes are positively correlated in linear and the correlation coefficient is about 0.27. Both the experimental results indicate that the learned user representation can positively reflect the users' authority in quantitative ways.

Topic-sensitive user expertise. The CQA systems would label users when the history of user behaviors is quite enough to be expertise in certain fields. In this way, we can use the systematic tags on users to verify whether the learned user expertise contains the topic interests. Firstly, we choose three random cases to qualitatively analyze how the relevance exists between the learned user expertise and topics. The cosine similarity is introduced to measure the difference between user vector and question representation, and we use the cosine similarity to explore the most similar questions of users. Then we choose 30 most similar questions of each user and extract the keywords in questions to validate if the keywords are relative to the user tags. The experimental results are shown in Figure 6. It is observed that the extracted keywords of each user are correlated to the user tags, indicating the topic-sensitive user representations learned by our proposed method. Besides, we randomly choose 1,000 users who are labeled on "programming" and "relationships" respectively. Then we apply K-means [1] to cluster the sampled users as $K = 2$ based on the learned user representations. The accuracy of clusters is up to 67%, giving the evidence of the topic interests contained in the learned user representations.

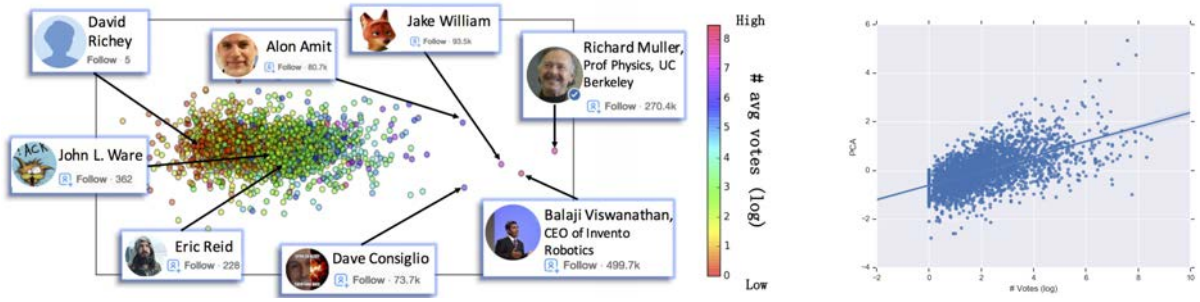
5.4 Effect of User Expertise in Answer Selection of Different Domains

In this section, we analyze the effect of user expertise in answer selection of different domains. We take four domains as examples from testing dataset, e.g. Movies, Experiences in Life, Physics, Police and Law Enforcement. We show the performance of our model (with user expertise and without) and the increase of metric by user expertise in Table 8. We notice that the increase of metric is more significant in domains like "Physics", "Police and Law Enforcement" than "Movies", "Experiences in Life". The experimental results demonstrate that user expertise plays a more important role in areas which need more domain knowledge. For instance, question "What is the most difficult thing you have ever watched?" in "Experiences in Life" can be answered by every user, but question "How far away from the nuclear plant is safe in case of nuclear meltdown?" in "Physics" is more professional.

5.5 Effect of User Expertise in Answer Representation Learning

In order to analyze the effect of user expertise in answer representation learning, we take an example and draw heat maps of sentence and word weights to demonstrate the difference between the setups with/without user expertise in answer representation learning (Figure 7). We pick up the question that "*What languages should I learn before starting to learn of AI?*" and its three answers as the example. There are many sentences in each answer and we only visualize the key sentences and their weights due to space limitation. The demonstration is shown in Figure 7, where Figure 7(a) and 7(d) depict weights by the learning mechanism with user expertise, Figure 7(b) and 7(e) describe the weights by the learning mechanism without user expertise respectively.

In the examples, we notice that both Figure 7(a) and 7(d) highlight the word "python" with the highest weight in the paragraph. Meanwhile, the word "AI" is given the highest weight in the paragraph shown in the Figure 7(b) and 7(e). Moreover, the sentences with python and programming are highlighted by the results from answer representation learning with user expertise, and the sentences with AI and ML are emphasized by the results from answer representation learning without user expertise. It is clear that the highlighted results from answer representation learning with user expertise are more relative to the question "What language should I learn before starting to learn of AI?". According to the results, we can observe that semantic relevance is less important than user expertise in such scenario. The answer would hardly point to the question by direct semantic relevance. In the examples, the better results from answer representation learning with user expertise are because of that the user_1 and user_2 have "computer programming"



(a) Visualization of user vectors. The blue/purple color means the higher expertise achieved in Quora, the green color means the median expertise, while the red color refers to the lower expertise.

(b) The correlation between learned user expertise and the true expertise accepted by communities.

Figure 5: The visualization of learned user representations and their correlation to the true expertise accepted by communities (sampled by 3000 users in Quora).



(a) (b) (c)

Figure 6: Examples of topic interests in learned user vectors.

Table 8: The performance (%) of our model (with user expertise and without) in different domains on Quora dataset with 80% of the data for training. ("UE" - user expertise. "Increase" - the increase of metric by modeling user expertise.)

Domain	Movies			Experiences in Life			Physics			Police and Law Enforcement		
	P@1	MRR	nDCG	P@1	MRR	nDCG	P@1	MRR	nDCG	P@1	MRR	nDCG
Without UE	35.00	54.56	74.88	35.71	58.99	82.65	29.36	53.78	78.48	27.66	49.28	72.21
With UE	52.50	69.76	87.47	58.33	74.91	91.67	72.48	83.55	94.56	70.21	81.74	93.73
Increase	50.00	27.86	16.82	63.34	26.99	10.91	146.87	55.36	20.49	153.85	65.86	29.82

tag in user profile (see Figure 7(c) and 7(f)), and the embedded user expertise in answer representation learning can help to emphasize on the programming-related words. The answer representation learning with user expertise can both capture the semantic and specialized relevance in answering, improving the emphasis on more important words in contents.

Furthermore, we conduct experiments on Quora and Stack Overflow dataset to validate the effectiveness of different attention mechanism for answer representation learning with user expertise at different attention levels. The experimental results are recorded in Table 9. With 60% training dataset, the implementation of attention mechanism with user expertise at hierarchical level achieves 0.6338, 0.7763 and 0.9191 in Quora dataset, resulted by P@1, MRR and nDCG respectively. The experimental results are better than the results from the implementation of attention mechanism with user expertise at word and sentence levels in Quora dataset (achieving 0.6205 and 0.6291 in P@1, 0.7655 and 0.7732 in MRR and 0.9110 and 0.9178 in nDCG). Moreover, all the attention mechanisms with user expertise at different semantic levels achieve better performance than the evaluation results from the attention mechanism without user expertise. It is interested that the attention mechanism with user expertise at sentence level achieve better performance than the implementation of attention mechanism with user expertise at word level. It may indicate that the implementation of attention



(a) Heat map with user expertise of answer$_1$. (b) Heat map without user expertise of answer$_1$. (c) Profile of user$_1$.

(d) Heat map with user expertise of answer$_2$. (e) Heat map without user expertise of answer$_2$. (f) Profile of user$_2$.

Figure 7: Visualization of the effect of encoding user expertise in answer representation. Orange denotes the sentence weight and blue denotes the word weight. The segments with higher weights are endowed with darker color. (a) and (d) are heat maps of answers with encoding user expertise in answer representation. (b) and (e) are heat maps of answers without encoding user expertise in answer representation. (c) and (f) are the user profiles about their expertise respectively. Answer representation learning with user expertise is able to emphasize sentences and words on personalized knowledgement. Answer representation learning without user expertise tends to focus on sentences and words which match the question semantically, i.e., the word "*AI*".

mechanism with user expertise on higher level is more effective than the implementation on lower level. The same conclusion can be reproduced in the experimental results from Stack Overflow dataset. Overall, the experimental results demonstrate the effectiveness on the implementation of attention mechanism with user expertise at both word and sentence levels. The answer representation learning with user expertise can emphasize more on the important words by semantic and specialized relevance in answering, improving the rank prediction of answer selection in CQA.

6 CONCLUSION

Answer selection is the key problem when scarce votes are provided in one question on CQA platform. However, the ranking of answers is simply tackled as a text mathcing problem in existing works, ignoring the influence of community. In this paper, we propose a user-expertise specific learning framework to model the dynamics of answer selection in CQA. Our model learns user expertise of each user by latent factors, better capturing the rank of answers. A novel score function is introduced to evaluate the relevance on question-answer pair and user expertise in question-user pairs accepted by

Table 9: The performance (%) of answer representation learning with user expertise at different attention levels, i.e., word, sentence and hierarchical levels, on datasets with 60% of the data for training. ("No" - the vanilla LSTM without attention mechanism. "Word" - employ attentions only at word-level and utilize the vanilla outputs of sentence level LSTM as the answer representation. "Sentence" - employ attentions only at sentence-level and use the vanilla outputs from word level LSTM. "Hierarchical" - employ attentions at both levels.)

Method	Quora dataset			Stack Overflow dataset		
	P@1	MRR	nDCG	P@1	MRR	nDCG
No	61.66	76.34	91.07	52.48	72.48	91.67
Word	62.05	76.55	91.10	54.34	73.42	92.16
Sentence	62.91	77.32	91.78	54.49	73.62	92.21
Hierarchical	**63.38**	**77.63**	**91.91**	**55.46**	**73.89**	**92.34**

communities. Moreover, we introduce latent user vectors into the representation learning of answering, capturing the implicit topic interests in learned user vectors. The experiments show that our model outperforms all baselines in answer selection task. Furthermore, we analyze the user representations learned by our model. The extensive experimental results prove that the learned user vectors can indicate the authority accepted by communities and capture the user interests in topic level, provide us a quantitative way to understand user expertise in CQA.

ACKNOWLEDGMENTS

This work was funded by the National Natural Science Foundation of China under Grant Numbers 61602439, 61425016, 91746301, and the National Key Research and Development Program of China under grant number 2016QY03D0504. Wentao Ouyang is also funded by the CAS Pioneer Hundred Talents Program under Grant Number 2920164120.

REFERENCES

[1] David Arthur and Sergei Vassilvitskii. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1027–1035.

[2] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. 2015. Multiple object recognition with visual attention. In *International Conference on Learning Representations (ICLR)*.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*.

[4] Yoshua Bengio et al. 2009. Learning deep architectures for AI. *Foundations and trends® in Machine Learning* 2, 1 (2009), 1–127.

[5] Mohamed Bouguessa, Benoît Dumoulin, and Shengrui Wang. 2008. Identifying authoritative actors in question-answering forums: the case of yahoo! answers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 866–874.

[6] Xin Cao, Gao Cong, Bin Cui, and Christian S Jensen. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *Proceedings of the 19th international conference on World wide web*. ACM, 201–210.

[7] Xin Cao, Gao Cong, Bin Cui, Christian Søndergaard Jensen, and Ce Zhang. 2009. The use of categorization information in language models for question retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 265–274.

[8] Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. 2016. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 1650–1659.

[9] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. 2017. GRAM: Graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 787–795.

[10] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. End-to-end continuous speech recognition using attention-based recurrent NN: first results. *arXiv preprint arXiv:1412.1602* (2014).

[11] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Advances in neural information processing systems*. 577–585.

[12] Huizhong Duan, Yunbo Cao, Chin-Yew Lin, and Yong Yu. 2008. Searching questions by identifying question topic and question focus. *Proceedings of ACL-08: HLT* (2008), 156–164.

[13] George H Dunteman. 1989. *Principal components analysis*. Number 69. Sage.

[14] Hanyin Fang, Fei Wu, Zhou Zhao, Xinyu Duan, Yueting Zhuang, and Martin Ester. 2016. Community-based question answering via heterogeneous social network learning. In *AAAI*.

[15] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. 1693–1701.

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[17] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*. 2042–2050.

[18] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 873–882.

[19] Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. 2005. Finding semantically similar questions based on their answers. In *SIGIR*. ACM, 617–618.

[20] Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 84–90.

[21] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *International Conference on Machine Learning*. 2342–2350.

[22] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[23] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 1188–1196.

[24] Pengfei Liu, Xipeng Qiu, Jifan Chen, and Xuanjing Huang. 2016. Deep fusion LSTMs for text semantic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1034–1043.

[25] Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1412–1421.

[26] Shanshan Lyu, Wentao Ouyang, Huawei Shen, and Xueqi Cheng. 2017. Truth Discovery by Claim and Source Embedding. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2183–2186.

[27] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[28] Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 27–48.

[29] Hyeonseob Nam, Jung-Woo Ha, and Jeonghee Kim. 2017. Dual Attention Networks for Multimodal Reasoning and Matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 299–307.

[30] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[31] Xipeng Qiu and Xuanjing Huang. 2015. Convolutional Neural Tensor Network Architecture for Community-Based Question Answering. In *IJCAI*. 1305–1311.

[32] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 379–389.

[33] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR*. ACM, 373–382.

[34] Yikang Shen, Wenge Rong, Zhiwei Sun, Yuanxin Ouyang, and Zhang Xiong. 2015. Question/Answer Matching for CQA System via Combining Lexical and Sequential Information. In *AAAI*. 275–281.

[35] Yi Tay, Minh C Phan, Luu Anh Tuan, and Siu Cheung Hui. 2017. Learning to rank question answer pairs with holographic dual LSTM architecture. In *SIGIR*. ACM.

[36] Kateryna Tymoshenko and Alessandro Moschitti. 2015. Assessing the impact of syntactic and semantic structures for answer passages reranking. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1451–1460.

[37] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations. In *AAAI*, Vol. 16. 2835–2841.

[38] Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 707–712.

[39] Gang Wang, Konark Gill, Manish Mohanlal, Haitao Zheng, and Ben Y Zhao. 2013. Wisdom in the social crowd: an analysis of quora. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1341–1352.

[40] Xiaochi Wei, Heyan Huang, Chin-Yew Lin, Xin Xin, Xianling Mao, and Shangguang Wang. 2015. Re-Ranking Voting-Based Answers by Discarding User Behavior Biases. In *IJCAI*. 2380–2386.

[41] Yu Wu, Wei Wu, Zhoujun Li, and Ming Zhou. 2016. Improving Recommendation of Tail Tags for Questions in Community Question Answering. In *AAAI*.

[42] Zhen Wu, Xin-Yu Dai, Cunyan Yin, Shujian Huang, and Jiajun Chen. 2018. Improving Review Representations with User Attention and Product Attention for Sentiment Classification. *arXiv preprint arXiv:1801.07861* (2018).

[43] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*. 2048–2057.

[44] Xiaobing Xue, Jiwoon Jeon, and W Bruce Croft. 2008. Retrieval models for question and answer archives. In *SIGIR*. ACM, 475–482.

[45] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[46] Scott Wen-tau Yih, Ming-Wei Chang, Chris Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*. 1744–1753.

[47] Xiaodong Zhang, Sujian Li, Lei Sha, and Houfeng Wang. 2017. Attentive Interactive Neural Networks for Answer Selection in Community Question Answering. In *AAAI*. 3525–3531.

[48] Zhou Zhao, Hanqing Lu, Vincent W Zheng, Deng Cai, Xiaofei He, and Yueting Zhuang. 2017. Community-Based Question Answering via Asymmetric Multi-Faceted Ranking Network Learning. In *AAAI*. 3532–3539.

[49] Zhou Zhao, Lijun Zhang, Xiaofei He, and Wilfred Ng. 2015. Expert finding for question answering via graph regularized matrix completion. *IEEE Transactions on Knowledge and Data Engineering* 27, 4 (2015), 993–1004.

Jointly Learning Explainable Rules for Recommendation with Knowledge Graph

Weizhi Ma
Department of Computer Science and Technology, Institute for Artificial Intelligence Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, China
mawz14@mails.tsinghua.edu.cn

Min Zhang*
Department of Computer Science and Technology, Institute for Artificial Intelligence Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, China
z-m@tsinghua.edu.cn

Yue Cao
Department of Computer Science, University of Southern California, Los Angeles, CA, USA
cao517@usc.edu

Woojeong Jin
Department of Computer Science, University of Southern California, Los Angeles, CA, USA
woojeong.jin@usc.edu

Chenyang Wang
Department of Computer Science and Technology, Institute for Artificial Intelligence Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, China
thuwangcy@gmail.com

Yiqun Liu
Department of Computer Science and Technology, Institute for Artificial Intelligence Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, China
yiqunliu@tsinghua.edu.cn

Shaoping Ma
Department of Computer Science and Technology, Institute for Artificial Intelligence Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, China
msp@tsinghua.edu.cn

Xiang Ren
Department of Computer Science, University of Southern California, Los Angeles, CA, USA
xiangren@usc.edu

ABSTRACT

Explainability and effectiveness are two key aspects for building recommender systems. Prior efforts mostly focus on incorporating side information to achieve better recommendation performance. However, these methods have some weaknesses: (1) prediction of neural network-based embedding methods are hard to explain and debug; (2) symbolic, graph-based approaches (e.g., meta path-based models) require manual efforts and domain knowledge to define patterns and rules, and ignore the item association types (e.g. substitutable and complementary). In this paper, we propose a novel joint learning framework to integrate *induction of explainable rules from knowledge graph* with *construction of a rule-guided neural recommendation model*. The framework encourages two modules to complement each other in generating effective and explainable recommendation: 1) inductive rules, mined from item-centric knowledge graphs, summarize common multi-hop relational patterns for inferring different item associations and provide human-readable explanation for model prediction; 2) recommendation module can be augmented by induced rules and thus have better generalization ability dealing with the cold-start issue. Extensive experiments[1] show that our proposed method has achieved significant improvements in item recommendation over baselines on real-world datasets. Our model demonstrates robust performance over "noisy" item knowledge graphs, generated by linking item names to related entities.

ACM Reference Format:
Weizhi Ma, Min Zhang, Yue Cao, Woojeong Jin, Chenyang Wang, Yiqun Liu, Shaoping Ma, and Xiang Ren. 2019. Jointly Learning Explainable Rules for Recommendation with Knowledge Graph. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313607

1 INTRODUCTION

Recommender systems play an essential part in improving user experiences on online services. While a well-performed recommender system largely reduce human efforts in finding things of interests, often times there may be some recommended items that are unexpected for users and cause confusion. Therefore, explanability

*Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313607

[1] Code and data can be found at: https://github.com/THUIR/RuleRec

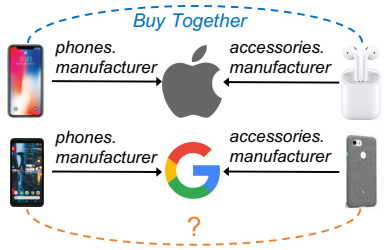


Figure 1: Illustration of Item-item Associations in a Knowledge Graph. Given items, relations and item associations (e.g. *Buy Together*), our goal is to induce rules from them and recommend items from rules. These rules are used to infer associations between new items, recommend items, and explain the recommendation.

becomes critically important for the recommender systems to provide convincing results—this helps to improve the effectiveness, efficiency, persuasiveness, transparency, and user satisfaction of recommender systems [45].

Though there are many powerful neural network-based recommendation algorithms proposed these years, most of them are unable to give explainable recommendation results [12, 14, 19]. Existing explainable recommendation algorithms are mainly two types: user-based [25, 33] and review-based [11, 46]. However, both of them are suffering from data sparsity problem, it is very hard for them to give clear reasons for the recommendation if the item lacks user reviews or the user has no social information.

On another line of research, some recommendation algorithms try to incorporate knowledge graphs, which contain lots of structured information, to introduce more features for the recommendation. There are two types of works that utilize knowledge graphs to improve recommendation: meta-path based methods [32, 43, 48] and embedding learning-based algorithms [24, 31, 44]. However, meta-path based methods require manual efforts and domain knowledge to define patterns and paths for feature extraction. Embedding based algorithms use the structure of the knowledge graph to learn users' and items' feature vectors for the recommendation, while the recommendation results are unexplainable. Besides, both types of algorithms ignore item associations.

We find that associations between items/products can be utilized to give accurate and explainable recommendation. For example, if a user buys a cellphone, it makes sense to recommend him/her some cellphone chargers or cases (as they are complementary items of the cellphone). But it may cause negative experiences if the system shows him/her other cellphones immediately (substitute items) because most users will not buy another cellphone right after buying one. So we can use this signal to tell users why we recommend an item for a user with explicit reasons (even for cold items). Furthermore, we propose that an idea to make use of item associations: After mapping the items into a knowledge graph, there will be multi-hop relational paths between items. Then, We can summarize explainable rules from for predicting association relationships between each two items and the induced rules will also be helpful for the recommendation.

To shed some light on this problem, we propose a novel joint learning framework to give accurate and explainable recommendations. The framework consists of a rule learning module and a recommendation module. We exploit knowledge graphs to induce explainable rules from item associations in the rule learning module and provide rule-guided recommendations based on the rules in the recommendation module. Fig. 1 shows an example of items with item associations in a knowledge graph. Note the knowledge graph here is constructed by linking items into a real knowledge graph, but not a heterogeneous graph that only consists of items and their attributes. The rule learning module leverage relations in a knowledge graph to summarize common rule patterns from item associations, which is explainable. The recommendation module combines existing recommendation models with the reduced rules, thus have a better ability to deal with the cold-start problem and give explainable recommendations. Our proposed framework outperforms baselines on real-world datasets from different domains. Furthermore, it gives an explainable result with the rules.

Our main contributions are listed as follows:

- We utilize a large-scale knowledge graph to derive rules between items from item associations.
- We propose a joint optimization framework that induces rules from knowledge graphs and recommends items based on the rules at the same time.
- We conduct extensive experiments on real-world datasets. Experimental results prove the effectiveness of our framework in accurate and explainable recommendation

2 PRELIMINARIES

We firstly introduce concepts and give a formal problem definition. Then, we briefly review BPRMF [27] and NCF [14] algorithms.

2.1 Background and Problem

Item recommendation. Given users U and items I, the task of item recommendation aims to identify items that are most suitable for each user based on historical interactions between users and items (e.g. purchase history). A user expresses his or her preferences by purchasing or rating items. These interactions can be represented as a matrix. One of the promising approaches is a matrix factorization method which embeds users and items into a low dimensional latent space. This method decomposes the user-item interaction matrix into the product of two lower dimensional rectangular matrices U and I for a user and an item, respectively. From these matrices, we can recommend new items to users.

Knowledge graph. A knowledge graph is a multi-relational graph that composed of entities as nodes and relations r as different types edges e. We can use many triples (head entity E_1, relation type r_1, tail entity E_2) to represent the facts in the knowledge graph [38].

Inductive rules on knowledge graph. There are several paths between two entities in the knowledge graph, and a path is consisted of entities with the relation types (e.g. $P_k = E_1 r_1 E_2 r_2 E_3$ is a path between E_1 and E_3). A rule R is defined by the relation sequence between two entities, e.g. $R = r_1 r_2$ is a rule. The difference between paths and rules is that rules focus on the relation types, not entities.

Problem Definition. Our study focus on jointly learning rules in a knowledge graph and a recommender system with the rules. Formally, our problem is defined as follows:

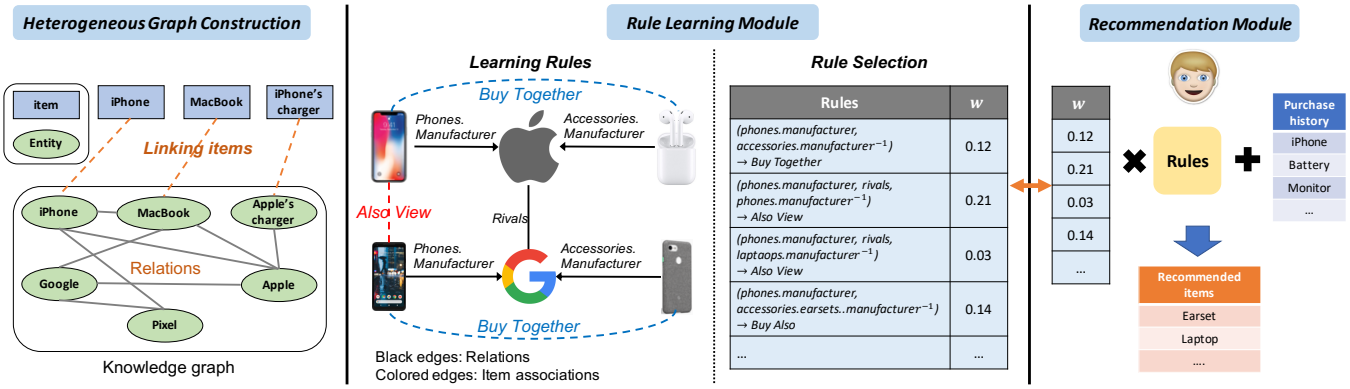


Figure 2: Overview of the Proposed RuleRec Framework. First, we build a heterogeneous graph from items and a knowledge graph. The rule learning module learns the importance of rules and the recommendation module learns the importance at the same time by sharing a parameter vector $\boldsymbol{w}$.

Definition 2.1 (Problem Definition). **Given** users U, items I, user-item interactions, item associations, and a knowledge graph, our framework **aims to** jointly (1) learn rules R between items based on item associations and (2) learn a recommender system to recommend items I'_u to each user u based on the rules R and his/her interaction history I_u. This framework outputs a set of rules R and recommended item lists I'.

2.2 Base Models for Recommendation

The framework proposed in our study is flexible to work with different recommendation algorithms. As BPRMF is a widely used classical matrix factorization algorithm and NCF is a state-of-the-art neural network based recommendation algorithm, we choose to modify them to verify the effectiveness of our framework.

Bayesian Personalized Ranking Matrix Factorization (BPRMF). Matrix Factorization based algorithms play a vital role in recommender systems. The idea is to represent each user/item with a vector of latent features. U and I are user feature matrix and item feature matrix respectively, and we use U_u to denote the feature vector of user u (I_i for item i). The dimensions of them are the same. In BPRMF algorithm [27], the preference score $S_{u,i}$ between u and i is computed by the inner product of U_u and I_i:

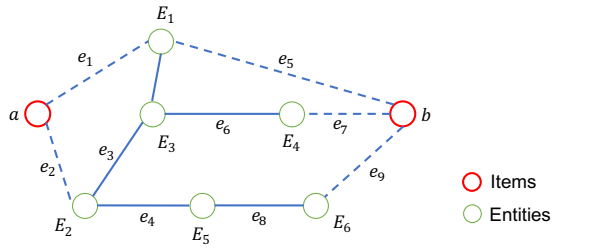
$$S_{u,i} = U_u^\top \cdot I_i \tag{1}$$

The objective function of BPRMF algorithm is defined as a pairwised function as follows:

$$O_{BPRMF} = \sum_{u \in U} \sum_{p \in I_u, n \notin I_u} (S_{u,p} - S_{u,n}) \tag{2}$$

where p is a positive item that user u interacted before, and n is a negative item sampled randomly from the items user u has never interacted (n should not be in test set too).

Neural Collaborative Filtering (NCF). NCF [14] is a neural based matrix factorization algorithm. Similar to BPRMF, each user u and each item i has a corresponding feature vector U_u and I_i, respectively. NCF propose a generalized matrix factorization (GMF)



Edge type:
$e_1, e_2, e_5, e_7, e_9 \in r_1$
$e_3, e_4 \in r_2$
$e_6, e_8 \in r_3$

Paths between a and b:
$P_1 = e_1 E_1 e_5$
$P_2 = e_2 E_2 e_3 E_3 e_6 E_4 e_7$
$P_3 = e_2 E_2 e_4 E_5 e_8 E_6 e_9$

Related reasoning rules:
Rules for P_1: $R_1 = r_1 r_1$
Rules for P_2, P_3: $R_2 = r_1 r_2 r_3 r_1$

Figure 3: An example of a heterogeneous graph which consists of items and entities in a knowledge graph. The dashed lines are links between items and entities generated by an entity linking algorithm.

(Eq (3)) and a non-linear interaction part via a multi-layer perception (MLP) (Eq (4)) between user and item to extraction.

$$h_{u,i} = U_u^\top \cdot I_i \tag{3}$$

$$\begin{aligned} g_{u,i} &= \phi_n(...\phi_2(\phi_1(z_1))) \\ z_1 &= \phi_0(U_u \oplus I_i) \\ \phi_k(z_{k-1}) &= \phi_k(W_k^T z_{k-1} + b_{k-1}), \end{aligned} \tag{4}$$

where n is the number of hidden layers. W_k, b_l, and z_k are weight matrices, bias vector, and output of each layer. $\oplus$ is vector concatenation and ϕ is a non-linear activation function. Both $h_{u,i}$ and $g_{u,i}$ are user-item interaction feature vectors for GMF and MLP, respectively. The prediction equation of NCF is defined in Eq (5), in which the outputs of GMF and MLP parts are concatentated to get the final score. And we modified the objective function of NCF into Eq (6) in this paper.

$$S_{u,i} = \phi(\alpha \cdot h_{u,i} \oplus (1-\alpha) \cdot g_{u,i}) \tag{5}$$

$$O_{NCF} = \sigma(\sum_{u \in U} \sum_{p \in I_u, n \notin I_u} (S_{u,p} - S_{u,n})) \tag{6}$$

3 THE RULEREC FRAMEWORK

Framework Overview. Recommendation with rule learning consists of two sub-tasks: 1) rule learning in a knowledge graph based on item associations; 2) recommending items for each user u with his/her purchase history I_u and the derived rules R.

To cope with these tasks, we design a multi-task learning framework. The framework consists of two modules, a rule learning module and a recommendation module. The rule learning module aims to derive useful rules through reasoning rules with ground-truth item associations in the knowledge graph. Based on the rule set, we can generate an item-pair feature vector whose each entry is an encoded value of each rule. The recommendation module takes the item-pair feature vector as additional input to enhance recommendation performances and give explanations for the recommendation. We introduce a shared rule weight vector w which indicates the importance of each rule in predicting user preference, and shows the effectiveness of each rule in predicting item pair associations. Besides, based on the assume that useful rules perform consistently in both modules with higher weights, we design a objective function to conduct jointly learning:

$$\min_{V,W} O = \min_{V,W} \{O_r + \lambda O_l\} \tag{7}$$

where V denotes the parameters of the recommendation module, and W represents the shared parameters of the rule learning and the recommendation module. The objective function consists of two terms: O_r is the objective of the recommendation module, which recommends items based on the induced rules. O_l is the objective of the rule learning module, in which we leverage the given item associations to learn useful rules. λ is a trade-off parameter.

3.1 Heterogeneous Graph Construction

First, we build a heterogeneous graph containing items for the recommendation and a knowledge graph. For some items, we can conduct exactly mapping between the item and the entity, such as "iPhone", "Macbook". For other items, it is hard to find an entity that represents the items, such iPhone's charger. Thus, we adopt entity linking algorithm [6] to find the related entities of an item from its title, brand, and description in the shopping website. In this way, we can add new nodes to the knowledge graph that represents items and add some edges for it according to entity linking results. Then, we get a heterogeneous graph which contains the items and the original knowledge graph. Fig. 3 is an example.

3.2 Rule Learning Module

The rule learning module aims to find the reliable rule set R_A associated with given item associations A in the heterogeneous graph.

Rule learning. For any item pair (a, b) in the heterogeneous graph, we use a random walk based algorithm to compute the probabilities of finding paths which follow certain rules between the item pair, similar to [16, 17]. Then, we obtain feature vectors for item pairs. Each entry of the feature vector is the probability of a rule between the item pair. Here, we focus on relation types between the item pair to obtain rules such as R_1 in Fig. 3, because it is general to the entities to capture the rules between items.

First, we define the probability of a rule between an item pair. Given a rule $R = r_1 ... r_k$, probability P with the rule from a to b is defined as:

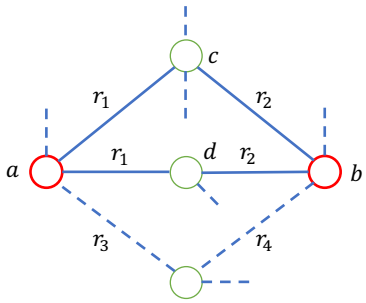


Figure 4: An example of a graph between items a and b. r represents a edge type or a relation type.

$$P(b|a, R) = \sum_{e \in N(a,R')} P(e|a, R') \cdot P(b|e, r_k), \tag{8}$$

where $R' = r_1 ... r_{k-1}$, and $P(b|e, r_k) = \frac{I(r_k(e,b))}{\sum_i I(r_k(e,i))}$ is the probability of reaching node b from node e with a one-step random walk with relation r_k. $I(r_k(e, b))$ is 1 if there exists a link with relation r_k from e to b, otherwise 0. If $b = e$, then $P(b|e, r_k) = 1$ for any r_k. $N(a, R')$ denotes a node set that can be reached with rule R' from node a. For example, $P(b|a, R)$ with a rule $R = r_1 r_2$ in Fig. 4 is computed as follows:

$$P(b|a, R) = P(c|a, r_1) \cdot P(b|c, r_2) + P(d|a, r_1) \cdot P(b|d, r_2)$$

Second, we define a feature vector between an item pair. Given a set of rules, a rule feature vector for an item pair (a, b) is defined as $x_{(a,b)} = [P(b|a, R_1), ..., P(b|a, R_n)]^\top$. Each entry in the feature vector $x_{(a,b)}$ represents a encoded value of rule R_i between a and b.

Rule selection. To select the most useful rules from the derived rules, we will introduce two types of selection methods: hard-selection and soft-selection.

Hard-selection method. Hard-selection method set a hyper parameter to decide how many rules we want to select with a selection algorithm firstly. Then we use a chi-square method and a learning based method to choose n rules in this study:

(1) Chi-square method. In statistics, the chi-square test is applied to measures dependence between two stochastic variables A and B (9) (to test if P(AB) = P(A)P(B)). $N_{A,B}$ is the observed occurrence of two events from a dataset and $E_{A,B}$ is the expected frequency. In feature selection, as the features that have lower chi-square scores are independent of prediction target are likely to be useless for classification, chi-square scores between each column of feature vector ($x_{(a,b)}$) and prediction target ($y_{a,b|A}$) are used to select the top n useful features [29].

$$\chi^2_{A,B} = \sum \frac{(N_{A,B} - E_{A,B})^2}{E_{A,B}} \tag{9}$$

(2) Learning based method. Another way to conduct feature selection is to design a objective function O_l that compute importance of each rule and try to minimize it. In the objective function, we introduce a weight vector w whose each entry represents importance of each rule. For an item pair (a, b), we use $y_{a,b|A}$ to denote

whether a and b have association A ($y_{a,b|A}$ is 1 if they have, and 0 otherwise.). We define the following objective functions:

- Chi-square objective function

$$\sum_{allpairs \in A} \sum_{i=0}^{|x_{(a,b)}|} w_i \cdot (x_{(a,b)}(i) + b - y_{a,b|A})^2 \tag{10}$$

- Linear regression objective function

$$\sum_{allpairs \in A} \sum_{i=0}^{|x_{(a,b)}|} (w_i \cdot x_{(a,b)}(i) + b - y_{a,b|A})^2 \tag{11}$$

- Sigmoid objective function

$$\sum_{allpairs \in A} \sum_{i=0}^{|x_{(a,b)}|} \frac{w_i}{1 + e^{-|x_{(a,b)}(i)+b-y_{a,b|A}|}} \tag{12}$$

where $x_{(a,b)}(i)$ is i-th entry of $x_{(a,b)}$. To make the objective function reasonable, we constrain that $\sum_i w_i = 1$ and $w_i > 0$. In training steps, if $x_{(a,b)}(i)$ shows positive correlation with $y_{a,b|A}$, then rule i is likely to be useful for item association classification and will get higher weight according to the loss functions. So similar to the chi-square method, the top weighted rules will be selected.

Soft-selection method. Besides the hard-selection method, another way to make use of the learning based objective functions is to take the weight of each rule as a constrain on the rules weights in the recommendation module. No rule will be removed from rule set in this way and it will not introduce extra hyper-parameter. Due to this method is flexible to be combined into other part, we introduce the soft-selection method with learning based objective functions to the recommendation module as a multi-task learning. In such condition, there is no extra constrain on rule weight ($\sum_i w_i = 1$ or $w_i > 0$). The detail of the multi-task learning method will be shown in Section 3.5.

As the rule set is derived from an item association in rule learning module. To apply different item associations at the same time, we can combine the rule sets from different item associations together to get a global rule set R.

3.3 Item Recommendation Module

We propose a general recommendation module than can be combined with existing methods. This module utilizes the derived rule features to enhance recommendation performances.

The goal of this module is to predict an item list for user u based on the item set I_u s/he interacted (e.g. purchased) before. Previous works calculate the preference score $S_{u,i}$ of user u purchase candidate item i, and then rank all candidate items with their scores to get the final recommendation list. As shown in Eq (13), we propose a function f_w parameterized by the shared weight vector w to combine the score $S_{u,i}$ with rule features between candidate item i and items user interacted (e.g. purchased) under rule set R. A score $S'_{u,i}$ for our method is defined as:

$$S'_{u,i} = f_w(S_{u,i}, \sum_{k \in I_u} F_{(i,k|R)}) \tag{13}$$

The feature vector for item pair (a, b) under rule set R is denoted by $F_{(a,b|R)}$. Note that $F_{(a,b|R)}$ is different from $x_{(a,b)}$ and calculated by $F_{(a,b|R)} = \sum_{e \in N(a,R')} P(e|a, R') \cdot I(b|e, r_k)$. $I(b|e, r_k)$ is an indicator function: if there is a edge in relation type r_k between b and e, $I(b|e, r_k) = 1$; otherwise 0. The reason why we adopt another feature generation method is that in recommendation module, we concerns more about if there exists a path in this rule between two items. The weight of each rule will be used in explaining the recommendation result, so we should make the comparing between rules fair. While longer rules are more likely to get lower score (more random walk steps so lower probability). If the feature vector is still x, it will hurt the explainable of our module. Thus we use $F_{(a,b|R)}$ as the feature vector here, which represents the frequency of each rule between the two items.

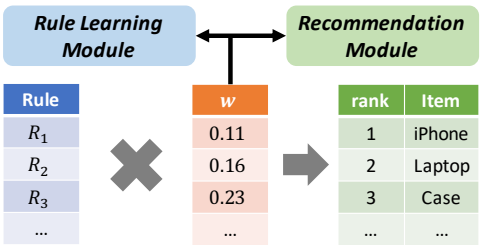


Figure 5: Multi-task learning of the rule learning module and the recommendation module. These two modules share the parameter w.

To consider the global item associations between candidate item i and the item set I_u, we add the rule features between i and each item I_k in I_u together. For convenience, the new feature vector is named as $F_{(i,I_u|R)}$. So Eq (13) can be rewrite as the following:

$$S'_{u,i} = f_w(S_{u,i}, F_{(i,I_u|R)}) \tag{14}$$

We define the objective function for the recommendation module as follows:

$$\begin{aligned} O_r &= \sum_{u \in U} \sum_{p \in I_u, n \notin I_u} (S'_{u,p} - S'_{u,n}) \\ &= \sum_{u \in U} \sum_{p \in I_u, n \notin I_u} \left(f_w(S_{u,p}, F_{(p,I_u|R)}) - f_w(S_{u,n}, F_{(n,I_u|R)}) \right), \end{aligned} \tag{15}$$

where p is a positive item ($\in I_u$) and n is a random sampled negative item ($\notin I_u$) for user u. Note that the rule weight vector w gives explanations for item pairs with rules in recommendation module. If a candidate item i gets a higher score than other candidate items, the rule which contributes the highest score for i and the corresponding items the user bought can be used to explain why the algorithm recommends i to the user. In other words, the introduction of rule features make the recommendation results explainable. There are some case studies in Section 5.5.

This combination method is flexible and easy to introduce rule features to many previous recommendation models (use the algorithm's prediction function to calculate $S_{u,i}$). In this study, we implement this recommendation module with BPRMF (traditional MF algorithm) and NCF (neural network based MF algorithm). Since it is a two step algorithm (to learn rules firstly and then conduct recommendation), we denote them as RuleRec$_{two}$(BPRMF) and RuleRec$_{two}$(NCF). The prediction function and objective function of them are Eq (14) and Eq (15), where $S_{u,i}$ is replaced by the prediction function of BPRMF (Eq (1)) and NCF (Eq (5)), respectively.

Table 1: The statistics of item association pairs in different domain. #Involved item means the number of items that have at least one type of association with any other items. #Item is the number of involved items with association and #Pair is the number of item pairs with association.

Dataset	#Item	#Involved Item	Also View		Buy After Viewing		Also Buy		Buy Together	
			#item	#Pair	#item	#Pair	#item	#Pair	#item	#Pair
Cellphone	346,793	214,692	103,845	1,038,090	181,935	1,818,990	71,660	716,240	29,372	293,360
Electronic	498,196	318,922	123,959	1,239,230	250,409	2,503,730	159,562	1,595,260	31,040	310,040

3.4 Multi-task Learning

In Sections 3.2 and 3.3, we introduced the two modules respectively. We can train the modules one by one to get the recommendation results. The shortcoming of training two modules separately is that the usefulness of rules in prediction item association is ignored. Instead, we share the rule weight w, and this weight can capture the importance of the rule in both the recommendation and item association prediction simultaneously as shown in Fig. 5. Thus, we propose a multi-task learning objective function defined as follows:

$$O = O_r + \lambda O_l \tag{16}$$

where O_l and O_r are the objective functions for the rule learning module and the recommendation module, respectively. Note that both objective functions share w.

The multi-task learning combination method is able to conduct rule selection and recommendation model learning together. Similar to the two step combination method, it is also flexible to to multiple recommendation models too. BPRMF and NCF are enhanced with this idea, and the modified algorithms are named as RuleRec$_{multi}$(BPRMF) and RuleRec$_{multi}$(NCF).

4 RULE SELECTION DETAILS

This section introduces the implementation details and results of the rule selection component in RuleRec.

4.1 Dataset and Implementation Details

We introduce item association datasets, a knowledge graph, and recommendation datasets for experiments.

Item association datasets. A open dataset with item associations is used in our experiments[2]. The item associations are extracted from user log on Amazon (same as [20]). Four types of item associations are considered: 1) Also view (ALV), users who viewed x also viewed y; 2) Buy after view (BAV), users who viewed x eventually bought y; 3) Also buy (ALB), users who bought x also bought y; 4) Buy together (BT), users frequently bought x and y together. ALV and BAV are substitute associations, and ALB and BT are complementary associations. The statistics of Cellphone and Electronics datasets with different item associations are shown in Table 1. Since the data is crawled from Amazon[3], the number of link is nearly ten times as large as the number of involved items in each association type. Besides, as shown in the table, over 37% items do not have any association with other items in this dataset.

Knowledge graph dataset. Freebase [2] is used to learn rules. It is the largest open knowledge graph[4], containing more than 224M entities, 784K relation types, and over 1.9 billion links.

The link prediction algorithm [5] [6] is used to connect items (with their titles, brands, and descriptions) and entities in DBPedia [6] firstly. Then the linked entities in DBPedia are mapped to the entities in Freebase with a entity dictionary [7]. As there is a probability score of each linked entity with the algorithm, which represents the confidence of this linking. So if the probability of a word links to a entity is lower than 0.6, we will ignore it to make the link result more accurate.

Due to the large scale of the knowledge graph, it is infeasible to enumerate all possible rules in this step. Following the idea in [17], we require that all derived rule needs to be supported by at least a fraction α of the training item pairs, as well as being of length no more than β (there will be huge number of rules without the length constraint). In the experiments, we set α to 0.01 (the same as [17]), and β to 4, which means the maximum number of edges between entities in a path is 4.

4.2 Results of Rule Selection

Item linking to the Knowledge Graph. In this step, we link the items from different domains to the entities in the knowledge graph. Items in the Cellphone domain and the Electronic domain are connected with 33,542 entities and 55,180 entities in Freebase respectively. Due to the item-entity linking method is not in a one-by-one accurate linking but based on items' titles, brands, and descriptions, each item will be linked into several entities and each entity will be linked with several items. With the random walk strategy introduced in Section 3.2, we find that the four hop routes in the knowledge graph from these entities will pass over 10 million entities. To avoid introducing unrelated entities in random walk step, the type of entities are constrained on pre-defined entity types (e.g.: entities in "ns.base.brand", "ns.computer" and some other types are maintain), then the involved entity amount is reduced to around 100K in each domain.

Rule Learning. The derived rules of different associations in cellphone domain are summarized in Table 2. There are hundreds of rules derived from Cellphone domain in each association, while only around 46-70 rules are in Electronic domain. The possible reason is that comparing with Cellphone domain, Electronic domain contains more items and the items are more diversity. Most rules are supported by less than 0.01 of the training item pairs. so less general rules are derived.

Rule Selection. To select useful rules from the large rule set, we use the learning based (LR, Eq (11)) and chi-square based feature selection methods in Section 3.2. The idea of selection methods is to choose the rules by which any items in a specific association are

[2] http://jmcauley.ucsd.edu/data/amazon/
[3] www.amazon.com
[4] https://developers.google.com/freebase/
[5] http://model.dbpedia-spotlight.org/en/annotate
[6] https://wiki.dbpedia.org/
[7] https://drive.google.com/file/d/0Bw2KHcvHhx-gQ2RJVVJLSHJGYlk/view

Table 2: The number of derived rules from different associations.

Dataset	#ALV	#BAV	#ALB	#BT
Cellphone	700	948	735	675
Electronic	46	66	70	50

Table 3: Rule selection results on ALB association in the Cellphone domain.

LR		Chi-square		All	Upper Bound
Top 50	Top 100	Top 50	Top 100		
20.1%	40.1%	87.0%	88.5%	89.2%	90.7%

followed. E.g. if any item pairs in the BT association follows a rule R_k, then R_k is a useful rule for the BT association.

We choose the ALB association in the Cellphone dataset to verify the selection ability of the two methods. Because the derived rules will be used to extract item-item pair feature for the recommendation, a good rule should be able to indicate the associations between item i and user's purchase history I_u. So the recommendation dataset (Section 5.1) in the Cellphone domain is used for evaluation, we calculate the recall of whether there is at least one path satisfied rule r_k between the last item i_l user interacted and user's previous purchase history I_u. Due to not always exist at least one rule between i_l and I_u, there is a upper bound for the recall.

Table 3 shows the rule selection results ALB association in Cellphone domain and its upper bound. Chi-square based method outperforms linear-regression based method in rule selection. The reason is that rules with higher weight in linear regression model cannot fully represent usefulness of rules in the recommendation. However, Chi-square method is able to find the most useful rules, and the selected 50 rules cover 87.0% of user purchase history (only 2.2% percentage lower than using all rules). It is reasonable to choose only the subset of derived rules for the recommendation. Besides, we find that the upper bound in Electronic domain is only about 65%, indicating that the combination between rules in Electronic dataset is not as tightly as in the Cellphone dataset.

Besides, for multi-task learning framework, it is unnecessary to conduct rule selection because the model takes the effect of each rule in predicting item associations through the combined loss function Eq ((10), (11), or (12)).

5 RECOMMENDATION EXPERIMENTS

This section introduces dataset and experiment settings for comparing RuleRec with other baseline methods, as well as providing case study on analyzing different components of RuleRec.

5.1 Recommendation Dataset

The recommendation datasets are open datasets that extracted from Amazon [10, 21]. Each user's purchase histories in Amazon are recorded with the purchased items and times. We conduct experiments using two datasets: Amazon Cellphone and Amazon Electronic. Each user has at least 5 interactions with items. The statistics of the datasets are summarized in Table 4.

Table 4: The statistics of recommendation datasets.

Dataset	#user	#item	#links
Cellphone	27, 879	10, 429	194, 439
Electronic	22, 675	58, 741	195, 751

5.2 Experimental Settings

Evaluation Protocol. To evaluate the item recommendation performance, we use leave-one-out evaluation in the recommendation [1, 14]. The latest interactions between items and each user are used as positive items in test set, and the remaining data are used for training. Due to the loss function in our study is pair-wised, each positive item in training set will be trained with a negative item sampled from items that the user has not interacted. As for test set, since it is too time-consuming to rank all items for each user in evaluation, 99 negative items that are not interacted with the user are random sampled and added to test set [5, 36]. Therefore, in the test set, each user is evaluated with 99 negative items and one positive item. The target here is to generate a high-quality ranked list of items for each user.

Evaluation Metrics. We use Recall, Normalized Discounted Cumulative Gain (NDCG), and Mean reciprocal rank (MRR). Higher score means better performance in each metric. Recall focuses on whether the positive item is in the list, while NDCG and MRR take the position of the positive item into evaluation. Considering that the length of most recommendation list in real scenarios is 5 or 10, so the ranked list is truncated at 10 for all metrics. We calculate Recall@5, Recall@10, NDCG@10, and MRR@10 for evaluation.

5.3 Compared Methods

Three types of baselines (traditional matrix factorization, neural network based, and recommendation with knowledge graph) are used here:

- **BPRMF** [27]. As introduced in Section 2.2.1, this method follows the idea of matrix factorization with pairwise ranking loss.
- **NCF** [14]: This is a state-of-the-art latent factor model. It pretrains MLP and GMF part separately, and then ensembles the two models to get the final preference score. Following previous studies [5, 13], **MLP** and **GMF** are taken as baseline models too.
- **HERec** [31]: A state-of-the-art algorithm which using the knowledge graph for the recommendation. This method adopts meta-paths to generate the embeddings of users and items in the heterogeneous network with Deepwalk [26], and then use them in the recommendation. Two variants of this algorithm with different fusion functions, **HERec**$_{sl}$ (with the simple linear fusion function) and **HERec**$_{pl}$ (with personalized linear fusion function) are used as baseline models.
- **RippleNet** [34]: Another state-of-the art algorithm that incorporates the knowledge graph into recommender systems. It stimulates the propagation of user preferences on the set of knowledge entities to learn a user's potential interests.

Implementation Details. We adopt the implementation of BPRMF algorithm in MyMediaLite[8] (a famous open source package) on our experiments. The implementation of other algorithms are from

[8] http://www.mymedialite.net/index.html

the public codes that the authors provided in their papers (NCF[9], HERec[10], and RippleNet[11]). The four new models, RuleRec$_{two}$ with BPRMF, RuleRec$_{two}$ with NCF, RuleRec$_{multi}$ with BPRMF, and RuleRec$_{multi}$ with NCF are modified from BPRMF and NCF according to our framework respectively. We tune all the parameters to achieve the best performance of each algorithm.

The score function is defined as $S'_{u,i} = f_w(S_{u,i}, F_{(i,I_u|R)}) = S_{u,i} + \alpha \cdot w^T F_{(i,I_u|R)}$ in this section. Different implementations of f_w and their results will be analyzed in Section 5.5. All of the four types of item associations are used in the recommendation module for both two-step and multi-task learning algorithms. Top 50 rules of each type of item associations (selected with chi-square method) are chose to the two-step based methods. To make the comparison fair, these rules are used in the multi-task learning algorithms with the sigmoid objective function in the final experiments. The objective function is sigmoid (Eq (12)), as it performs the best in the three objective functions (Eq (10), (11), and (12)); due to the limited of length, we do not show the results here. The comparison of different amounts of rules will be introduced in Section 5.5.5.

5.4 Experiments and Performance Study

The experimental results of these algorithms in different domains are summarized in Table 5. We repeated each setting for 5 times and conducted the paired two-sample t-test on the 5 times experiment results for significant test. As shown in the table, the performance of algorithms in Electronic dataset is obviously worse than in Cellphone dataset. The reason is that the item count of Electronic dataset is about 6 times over the item count of Cellphone dataset (from Table 4), which makes the recommendation in Electronic dataset more difficult.

1. The Enhanced Algorithms vs. the Originals. NCF algorithm performs better than BPRMF algorithm in both datasets, as more complex user and item feature interactions are taken into consideration in NCF. Looking into the results of BPRMF algorithms and NCF algorithms, we find that RuleRec$_{multi}$ with BPRMF gets 6.5% to 11.0% improvements over BPRMF in different evaluation metrics on two domains. The improvements of RuleRec$_{multi}$ with NCF in Recall@5, Recall@10, NDCG@10, and MRR@10 are between 3.0% to 6.4% comparing with NCF in Cellphone domain, while the improvements of which on Electronic is lower than in Cellphone domain. Though RuleRec$_{multi}$ with BPRMF is improved more than RuleRec$_{multi}$ with NCF, RuleRec$_{multi}$ with NCF still achieves the best performance in Cellphone domain and RuleRec$_{multi}$ with BPRMF performs the best in Electronic domain.

2. Overall Performances. Besides, we find that any one of the enhanced algorithms outperform all baselines in both Cellphone and Electronic domains in each metric. And most of the improvements are statistically significant, showing that the derived rules from the knowledge graph are really helpful to generate a better ranked item list for the recommendation. The multi-task learning algorithms (RuleRec$_{multi}$ with BPRMF and RuleRec$_{multi}$ with NCF) show better performances than the two-step learning algorithms (RuleRec$_{two}$ with BPRMF and RuleRec$_{two}$ with NCF), indicating that the combination of recommendation loss and rule selection loss in weight training is able to boost the recommendation results. Though the learning-based selection methods perform worse than chi-square in Section 4.2, it does helpful in the multi-task learning.

3. The Performances of HERec and RippleNet. We also note that HERec based algorithms and RippleNet, some state-of-the-art algorithms that uses the knowledge graph for the recommendation, performs worse in these datasets. We think the possible reason is that unlike movie, book, or Yelp datasets which contains many well organized category features (such as director, movie type, actor/actress name in movie dataset) to construct a compact graph, here we link Cellphone and Electronic datasets with a real knowledge graph Freebase. Though Freebase contains more information, but it is not as clean as the on-topic sub graph and makes it harder to mine valuable information, so these algorithms perform worse. More analyses are shown in Section 5.5.1.

To summarize, the derived rules from knowledge graph are valuable for item pair feature vector learning, and the learned vector is able to enhance multiple basic recommendation models (BPRMF and NCF here). Comparing with the two-step combination method, multi-task learning for both recommendation and rule selection contributes more on rule weight learning. Due to the flexible of the proposed framework, the derived rules are able to combine with other recommendation models to boost performances significantly.

5.5 Case Study and Performance Analysis

1. Performance Comparison in compact heterogeneous graph. Experiments in Section 5.4 are conducted on a large heterogeneous graph extracted from real knowledge graph. In this subsection, some extra experiments are conducted on a compact heterogeneous graph, which is constructed based on item attributes, in MovieLens-1M dataset[12]. We adopt the proposed algorithm and HERec algorithm in this dataset following the setting in RippleNet.

The experimental results are shown in Table 6. Our model performs better than HERec while worse than RippleNet , there are two possible reason: 1) relation type is very limited in this dataset (only 7), so the power of rule selection for the recommendation in RuleRec is limited in this scenario. 2) MovieLens-1M is different from real knowledge graph datasets in Section 5.4 (which is constructed by linking items into Freebase), the connection coverage of it is very perfect and RippleNet benefits a lot from this. The results indicate that the proposed algorithms is able to achieve noteworthy performance in compact heterogeneous graph.

2. Explainability of the learned rules. In Section 5.4, the results indicate the derived rules are useful in providing more accurate recommendation results. In this section, we will show the explainability of the derived rules for the recommendation. Two positive weighted rules on RuleRec$_{multi}$ are shown as the following :

- R_1 = "computer.computer.manufacturer"
- R_2 = "computer.computer.compatible_oses" $->$ "computer.os_compatibility.operating_system" $->$ "computer.operating_system.includes_os_versions"

Where the words with quotation marks are the relation types defined in Freebase (such as "computer.computer.manufacturer").

[9] https://github.com/hexiangnan/neural_collaborative_filtering
[10] https://github.com/librahu/HERec
[11] https://github.com/hwwang55/RippleNet
[12] https://github.com/hwwang55/RippleNet/tree/master/data/movie

Table 5: Performance Comparison between RuleRec and Other Methods in Different Domains. RuleRec$_{two}$ and RuleRec$_{multi}$ are our proposed models. RuleRec$_{two}$ is a two-step rule-based model and RuleRec$_{multi}$ is a multi-task model. These models use BPRMF or NCF as a recommendation model. * indicates statistical significance at $p < 0.01$ compared to the best baseline model.

Methods / Dataset	Cellphone				Electronic			
	Recall@5	Recall@10	NDCG@10	MRR@10	Recall@5	Recall@10	NDCG@10	MRR@10
BPRMF [27]	0.3238	0.4491	0.2639	0.2058	0.1886	0.2763	0.1571	0.1207
GMF [13]	0.3379	0.4666	0.2789	0.2223	0.1988	0.2835	0.1657	0.1298
MLP [5]	0.3374	0.4779	0.2790	0.2182	0.2000	0.2883	0.1681	0.1315
NCF [14]	0.3388	0.4751	0.2761	0.2151	0.2005	0.2916	0.1679	0.1300
Hec$_{sl}$ [31]	0.2436	0.3481	0.2040	0.1600	0.1870	0.2851	0.1534	0.1135
Hec$_{pl}$ [31]	0.2511	0.3564	0.2090	0.1641	0.1948	0.2851	0.1628	0.1256
RippleNet [34]	0.2834	0.4042	0.2219	0.1780	0.1965	0.2865	0.1638	0.1265
RuleRec$_{two}$ (BPRMF)	0.3495*	0.4768	0.2813*	0.2201*	0.2050*	0.2932	0.1707*	0.1334*
RuleRec$_{multi}$ (BPRMF)	0.3568*	0.4829*	0.2864*	0.2246*	0.2071*	0.2946*	**0.1718***	**0.1341***
RuleRec$_{two}$ (NCF)	0.3538*	0.4876*	**0.2902***	**0.2296***	0.2049*	**0.2947***	0.1681	0.1296
RuleRec$_{multi}$ (NCF)	**0.3569***	**0.4894***	**0.2902***	0.2290*	**0.2074***	0.2917	0.1702*	0.1330

Table 6: Performance Comparison on MovieLens dataset.

Model	AUC
Hec$_{sl}$ [31]	0.894
Hec$_{pl}$ [31]	0.895
RippleNet [34]	**0.921**
RuleRec$_{two}$**(BPRMF)**	0.907

These rules are with positive weights in the recommendation module, indicating that if a new item b exists a path between it and item a user bought before, item b is more likely to get higher score.

First we try to verify if item pairs with these rules affect user's purchase. As to R_1, it links a computer product and its manufacturer. If two items a and b have a path in R_1, it means that item b is likely to be manufactured by the same as item a. For R_2, two example entity paths in this rule are: 1) "Mac Mini" - "os x yosemite" - "OS X" - "IOS" and 2) "Surface Pro" - "Windows 10" - "Windows" - "Windows Phone". It shows that users are tend to use similar operating systems in both cellphone and computer. As you can see, these rules are consistent with our common sense.

Then, to check whether users agree that the selected rule will be helpful to improve the explainability of the recommendation if the rules are used in real scenarios, the derived rules in Cellphone dataset are labeled by three experts (only agree or disagree, 100 rules from ALB and BT associations). The results show that over 94% learned rules are accepted by users (87% rules are accepted by all users).

Due to the effective rule i in calculating user preference on a specific item will get higher score ($w_i^\top F_{(i,I_u|r_i)}$) for the preference prediction. So for each item in the ranked list, unless it has no path between it and items in user's purchase history, we can generate the most important rule for it by ranking the score of each rule in preference prediction.

3. Study on different model integration strategies. The score function (Eq (14)) and the rule weight vector w in f_w affect the performance of the recommendation module. We experiment with several ways to identify the best combination methods.

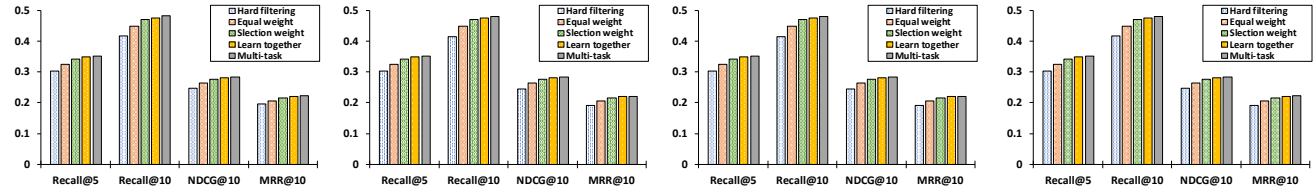
- Hard filtering: Remove candidate items that have no rule with any item in I_u. Formally, $S'_{u,i} = S_{u,i} \cdot I(F_{(i,I_u|R)})$, where $I(F_{(i,I_u|R)}) = 1$ if $\sum F_{(i,I_u|R)} >= 1$ otherwise 0.
- Equal weight: Each rule gets an equal weight in prediction. $S'_{u,i} = S_{u,i} + \alpha \cdot w^\top F_{(i,I_u|R)}$, and $w = [0.02, ..., 0.02]$.
- Selection weight: $S'_{u,i} = S_{u,i} + \alpha \cdot w^\top F_{(i,I_u|R)}$, and $w = w_{ruleselection}$ $w_{ruleselection}$ is the rule weight vector trained by Eq (12) in rule selection step.
- Learn together: The rule weight vector is trained with the original recommendation model. $f_w(a,b) = a + w^\top b$.
- Multi-task: The rule weight vector w is shared by recommendation learning and rule selection part, and the score prediction function is $f_w(a,b) = a + w^\top b$.

The results of applying multiple score functions on RuleRec(BPRMF) model in Cellphone dataset are shown in Fig. 6. The performances of the score functions are Hard filtering < Equal weight < Selection weight < Learn together < Multi-task with all metrics in all associations. First, we can see that the multi-learning method achieves the best performance on all metrics, and the improvements are significant, which indicates multi-task learning is very helpful in rule weight learning for better recommendation results. Second, since the hard filtering method is likely to ignore both negative items and positive items (from Table 3, we can see that sometimes there is no rule between the positive item and item purchase history I_u). Third, though selection weight contributes on the recommendation (better that equal weight), it is still worse than Learn together model.

4. Study on single association vs. all associations

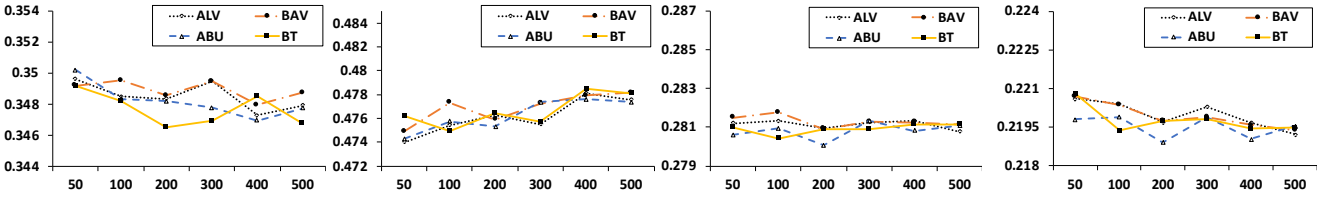
In this subsection, we compare the performance of RuleRec$_{multi}$ with BPRMF with only one type of association and all associations, the results are summarized in Table 7.

First, we can see that with the rules derived by any one of the four associations, RuleRec$_{multi}$ with BPRMF outperforms BPRMF algorithm significantly. The performances of using different associations are similar, but all of them are valuable for mining the item relationships to boost the recommendation results. Second, RuleRec$_{multi}$ with BPRMF derived by all kinds of associations outperforms RuleRec$_{multi}$ with a single association, indicating that the combination contributes for the recommendation models.



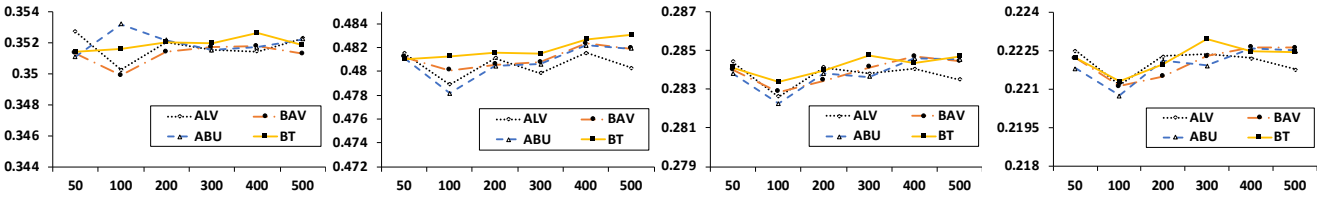
(a) ALV association (b) BAV association (c) ABU association (d) BT association

Figure 6: Performance comparison under different score function settings in the Cellphone dataset. We select the top 50 rules in each association by chi-square method. Among all score function settings, the multi-task learning method performs the best.



(a) Recall@5 (b) Recall@10 (c) NDCG@10 (d) MRR@10

Figure 7: The performance of using different number of rules in RuleRec$_{two}$ with BPRMF.



(a) Recall@5 (b) Recall@10 (c) NDCG@10 (d) MRR@10

Figure 8: The performance of using different number of rules in RuleRec$_{multi}$ with BPRMF.

Table 7: The results of using a single association vs. all associations. BPRMF is used as a recommendation model. Our proposed model shows the best performances when using all kinds of associations.

Type	Recall@5	Recall@10	NDCG@10	MRR@10
None	0.3238	0.4491	0.2639	0.2058
ALV	0.3527	0.4815	0.2844	0.2225
BAV	0.3513	0.4812	0.2840	0.2222
ABU	0.3511	0.4811	0.2838	0.2218
BT	0.3514	0.4810	0.2841	0.2222
ALL	**0.3568**	**0.4829**	**0.2864**	**0.2246**

5. Recommendation with different rule counts. In Section 3.2, rule selection is introduced as an important part in rule learning. Does rule selection is really necessary? We conduct further experiments on each association with different of derived of rules (selected with chi-square method, 50, 100, 200, 300, 400, and 500 respectively) with RuleRec$_{two}$ and RuleRec$_{multi}$ with BPRMF.

From Fig. 7, the performance of RuleRec$_{two}$ with BPRMF decreases in recall@5, MRR@10 as the rule number increases. At the same time, NDCG@10 keeps stable and Recall@10 increases. The overall performances is not getting better when more rules are applied in the recommendation learning. The possible reason is that with the grows of rule number, lots of "bad" rules are included and the two-step model RuleRec$_{two}$ with BPRMF shows worse ability in dealing with them properly. However, due to the rule selection is taken into consideration in the multi-task learning based algorithms, we find that the performance of RuleRec$_{multi}$ with BPRMF algorithm shows better performances as the rule number increases (Fig. 8). Furthermore, the performances of RuleRec$_{multi}$ with BPRMF is significantly better than those of RuleRec$_{two}$ with BPRMF (paired two-sample t-test on the experimental results with different count of rules. $p < 0.01$). The results show that the multi-task learning based algorithms are able to tackle with large number of rules even though there are useless rules.

6 RELATED WORK

Combine Side-information for the Recommendation. Matrix factorization based algorithms [27, 28] are widely used to tackle recommendation problems. Recently, recommendation algorithms achieve remarkable improvements during these years with help of

deep learning models [12, 14, 30, 47, 49] and the successful introducing of side-information [15, 19, 22, 23, 34]. In this study, we focus on the introducing of side-information in the knowledge graph for the recommendation, and there already two types of studies using the knowledge graph in the recommendation: path-based and embedding learning based.

Path-based methods adopt random walk on predefined meta-paths between user and items in the knowledge graph to calculate user's preference on an item. Yu et al. first propose to use meta-paths to utilize user-item preferences and then expand matrix factorization for the recommendation [43]. Shi et al. use weighted paths for explicit recommendation [32]. Zhao et al. design a factorization machine with the latent features from different meta-paths [48]. Catherine et al. design a first-order probabilistic logical reasoning system, named ProPPR, to integrate different meta-paths in a knowledge graph [3, 4]. All these methods achieve improvements in the recommendation, while the weakness of them is that they ignore the type of item associations.

Embedding learning based methods conduct user/item representation learning based on the Knowledge graph structure firstly. The learned embedding [8] is applied in Zhang's study to get item embedding for the recommendation [24]. Zhang et al. use TransR [18] to learn the structural vectors of items, and these vectors are part of the final item latent vector for preference prediction [44]. Besides, some previous studies propose new algorithms in which Meta-path guided random walks are used in a heterogeneous network for user and item embedding learning and achieve outperform results [31, 34, 37, 48] in different ways. However, the embedding learning based methods give up the explainable strength of the knowledge graph, which is very valuable for the recommendation.

Rule Learning in the Knowledge Graph. Item-item relationships are considering as useful features for providing better recommendation results. Julian et al. firstly propose to a topic model based method to predict relationships (substitute or complementary) between products from reviews [20]. In their study, the ground truth is calculated in a data-driven way in Amazon. Then, more algorithms attempt to improve the prediction results with better algorithms. Word dependency paths are taken into consideration in Hu et al.'s work [41] and Wang et al. adopt a embedding based method to enhance the performance of relationship prediction [37].

However, these methods are suffering from cold items. So we proposed to not predict item associations directly, but mine meaningful rules in the knowledge graph with the ground truth item pairs. The rules will be applied to generate feature vectors for different item pairs without user reviews.

Knowledge graph is a multi-relational graph that composed of entities as nodes and relations as different types of edges [38]. In the past years, knowledge graphs have been used as important resources for many tasks [7, 39]. One of the main usage of knowledge graph is reasoning and entity relationship prediction. Lots of research are focus on reasoning, such as [40, 42]. While these studies focus on link prediction but not rule inducing, which is not proper for our study.

On the other line of research, several work attempt to learn the useful rules but not the prediction results from the knowledge graph with the ground truth entity pairs. Random walk based algorithms are proposed in Lao et al.'s studies [16, 17] and others' [9, 35]. These methods are able to show why the entity pair has a certain relationship according to the derived rules, which makes the results more explainable. In our study, we adopt a similar algorithm as them in rule learning module.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a novel and effective joint optimization framework for inducing rules from a knowledge graph with items and recommendation based on the induced rules.

Our framework consists of two modules: rule learning module and recommendation module. The rule learning module is able to derive useful rules in a knowledge graph with different type of item associations, and the recommendation module introduces the rules to the recommendation models for better performance. Furthermore, there are two ways to implement this framework: two-step and jointly learning.

Freebase, a large-scale knowledge graph, is used for rule learning in this study. The framework is flexible to boost different recommendation algorithms. We modify two recommendation algorithms, a classical matrix factorization algorithm (BPRMF) and a state-of-the-art neural network based recommendation algorithm (NCF), to combine with our framework. The proposed four rule enhanced recommendation algorithms achieve remarkable results in multiple domains and outperform all baseline models, indicating the effectiveness of our framework. Besides, the derived rules also show the ability in explaining why we recommend this item for the user, boosting the explainability of the recommendation models at the same time. Further analysis shows that our multi-task learning based combination methods (RuleRec$_multi$ with BPRMF and RuleRec$_two$ with NCF) outperform the two-step method with different number of rules. And the combination of rules derived by different associations contributes to better recommendation results.

In future, we plan to investigate how to design a embedding learning based combination algorithm which keeps the recommendation results explainable with the knowledge graph.

ACKNOWLEDGEMENTS

This work is supported by Natural Science Foundation of China (Grant No. 61672311, 61532011) and The National Key Research and Development Program of China (2018YFC0831900). Dr. Xiang Ren has been supported in part by NSF SMA 18-29268, Amazon Faculty Award, and JP Morgan AI Research Award.

REFERENCES

[1] Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. 2017. A generic coordinate descent framework for learning from implicit feedback. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1341–1350.

[2] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM, 1247–1250.

[3] Rose Catherine and William Cohen. 2016. Personalized recommendations using knowledge graphs: A probabilistic logic programming approach. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 325–332.

[4] Rose Catherine, Kathryn Mazaitis, Maxine Eskenazi, and William Cohen. 2017. Explainable entity-based recommendations with knowledge graphs. *arXiv preprint arXiv:1707.05254* (2017).

[5] Weiyu Cheng, Yanyan Shen, Yanmin Zhu, and Linpeng Huang. 2018. DELF: A Dual-Embedding based Deep Latent Factor Model for Recommendation.. In *IJCAI*. 3329–3335.

[6] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. 2013. Improving Efficiency and Accuracy in Multilingual Entity Extraction. In *Proceedings of the 9th International Conference on Semantic Systems (I-Semantics)*.

[7] Mohnish Dubey, Debayan Banerjee, Debanjan Chaudhuri, and Jens Lehmann. 2018. EARL: Joint Entity and Relation Linking for Question Answering over Knowledge Graphs. *arXiv preprint arXiv:1801.03825* (2018).

[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[9] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2016. Jointly embedding knowledge graphs and logical rules. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 192–202.

[10] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 507–517.

[11] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1661–1670.

[12] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 355–364.

[13] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer Product-based Neural Collaborative Filtering. *arXiv preprint arXiv:1808.03912* (2018).

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiarendle2009bprng Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[15] Liang Hu, Songlei Jian, Longbing Cao, and Qingkui Chen. 2018. Interpretable Recommendation via Attraction Modeling: Learning Multilevel Attractiveness over Multimodal Movie Contents.. In *IJCAI*. 3400–3406.

[16] Ni Lao and William W Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine learning* 81, 1 (2010), 53–67.

[17] Ni Lao, Tom Mitchell, and William W Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 529–539.

[18] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion.. In *AAAI*, Vol. 15. 2181–2187.

[19] Weizhi Ma, Min Zhang, Chenyang Wang, Cheng Luo, Yiqun Liu, and Shaoping Ma. 2018. Your Tweets Reveal What You Like: Introducing Cross-media Content Information into Multi-domain Recommendation.. In *IJCAI*. 3484–3490.

[20] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 785–794.

[21] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 43–52.

[22] James McInerney, Benjamin Lacker, Samantha Hansen, Karl Higley, Hugues Bouchard, Alois Gruson, and Rishabh Mehrotra. 2018. Explore, exploit, and explain: personalizing explainable recommendations with bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 31–39.

[23] Sharad Nandanwar, Aayush Moroney, and M Narasimha Murty. 2018. Fusing Diversity in Recommendations in Heterogeneous Information Networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 414–422.

[24] Enrico Palumbo, Giuseppe Rizzo, and Raphaël Troncy. 2017. Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 32–36.

[25] Haekyu Park, Hyunsik Jeon, Junghwan Kim, Beunguk Ahn, and U Kang. 2017. Uniwalk: Explainable and accurate recommendation for rating and network data. *arXiv preprint arXiv:1710.07134* (2017).

[26] Bryan Perozzi, Rami Al-Rfou', and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *KDD*.

[27] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[28] Jasson DM Rennie and Nathan Srebro. 2005. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*. ACM, 713–719.

[29] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*. Vol. 39. Cambridge University Press.

[30] Yilin Shen, Yue Deng, Avik Ray, and Hongxia Jin. 2018. Interactive recommendation via deep neural memory augmented contextual bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 122–130.

[31] Chuan Shi, Binbin Hu, Xin Zhao, and Philip Yu. 2018. Heterogeneous Information Network Embedding for Recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[32] Chuan Shi, Zhiqiang Zhang, Ping Luo, Philip S Yu, Yading Yue, and Bin Wu. 2015. Semantic path based personalized recommendation on weighted heterogeneous information networks. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 453–462.

[33] Beidou Wang, Martin Ester, Jiajun Bu, and Deng Cai. 2014. Who also likes it? generating the most persuasive social explanations in recommender systems. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

[34] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 417–426.

[35] Quan Wang, Bin Wang, Li Guo, et al. 2015. Knowledge Base Completion Using Embeddings and Rules.. In *IJCAI*. 1859–1866.

[36] Zengmao Wang, Yuhong Guo, and Bo Du. 2018. Matrix completion with Preference Ranking for Top-N Recommendation.. In *IJCAI*. 3585–3591.

[37] Zihan Wang, Ziheng Jiang, Zhaochun Ren, Jiliang Tang, and Dawei Yin. 2018. A path-constrained framework for discriminating substitutable and complementary products in e-commerce. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 619–627.

[38] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes.. In *AAAI*, Vol. 14. 1112–1119.

[39] Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 1271–1279.

[40] Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. Deeppath: A reinforcement learning method for knowledge graph reasoning. *arXiv preprint arXiv:1707.06690* (2017).

[41] Hu Xu, Sihong Xie, Lei Shu, and S Yu Philip. 2016. Cer: Complementary entity recognition via knowledge expansion on large unlabeled product reviews. In *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 793–802.

[42] Fan Yang, Zhilin Yang, and William W Cohen. 2017. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems*. 2319–2328.

[43] Xiao Yu, Xiang Ren, Yizhou Sun, Bradley Sturt, Urvashi Khandelwal, Quanquan Gu, Brandon Norick, and Jiawei Han. 2013. Recommendation in heterogeneous information networks with implicit user feedback. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 347–350.

[44] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 353–362.

[45] Yongfeng Zhang and Xu Chen. 2018. Explainable Recommendation: A Survey and New Perspectives. *arXiv preprint arXiv:1804.11192* (2018).

[46] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 83–92.

[47] Yan Zhang, Hongzhi Yin, Zi Huang, Xingzhong Du, Guowu Yang, and Defu Lian. 2018. Discrete Deep Learning for Fast Content-Aware Recommendation. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 717–726.

[48] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-graph based recommendation fusion over heterogeneous information networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 635–644.

[49] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018. Deep Reinforcement Learning for Page-wise Recommendations. *arXiv preprint arXiv:1805.02343* (2018).

Exploring Perceived Emotional Intelligence of Personality-Driven Virtual Agents in Handling User Challenges

Xiaojuan Ma*
Hong Kong University of Science and Technology
Hong Kong
mxj@cse.ust.hk

Emily Y. Yang*
Hong Kong University of Science and Technology
Hong Kong
yyangag@gmail.com

Pascale Fung
Hong Kong University of Science and Technology
Hong Kong
pascale@ece.ust.hk

ABSTRACT

An effective virtual agent (VA) that serves humans not only completes tasks efficaciously, but also manages its interpersonal relationships with users judiciously. Although past research has studied how agents apologize or seek help appropriately, there lacks a comprehensive study of how to design an emotionally intelligent (EI) virtual agent. In this paper, we propose to improve a VA's perceived EI by equipping it with personality-driven responsive expression of emotions. We conduct a within-subject experiment to verify this approach using a medical assistant VA. We ask participants to observe how the agent (displaying a dominant or submissive trait, or having no personality) handles user challenges when issuing reminders and rate its EI. Results show that simply being emotionally expressive is insufficient for suggesting VAs as fully emotionally intelligent. Equipping such VAs with a consistent, distinctive personality trait (especially submissive) can convey a significantly stronger sense of EI in terms of the ability to perceive, use, understand, and manage emotions, and can better mitigate user challenges.

CCS CONCEPTS

• **Human-centered computing → User studies**; *Natural language interfaces*; • **Computing methodologies → Intelligent agents**.

KEYWORDS

virtual agents, emotional intelligence, perceived emotional intelligence, personality, emotions, affective computing, dialog system

ACM Reference Format:
Xiaojuan Ma, Emily Y. Yang, and Pascale Fung. 2019. Exploring Perceived Emotional Intelligence of Personality-Driven Virtual Agents in Handling User Challenges. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313400

1 INTRODUCTION

Recent research and technology advancements have rendered the possible reality of living and interacting with intelligent virtual agents, such as Siri (Apple), Google Now, S Voice (Samsung), and Cortana (Microsoft). However, these virtual agents are found to handle affect-sensitive questions with inconsistency and inadequate empathy [61], and are also recorded on various occasions to fend off sexual assaults from users [41]. The situation beckons a need for emotional intelligence (EI) in a software agent (i.e., virtual agent). At such, understanding human perception of a virtual agent's EI and discovering factors involved in designing emotionally intelligent agents is of great necessity. The widely used term "agent" has no general consensus when it comes to definition. It is broadly defined as a hardware or software computational system that may have autonomous, proactive, reactive, and social ability [99]. Unlike robotic agents, a virtual agent does not have physical properties and is embodied as a computerized 2D or 3D character [81]. In particular, the virtual agents designed for our study are conversational 2D-animated software agents that interact with people through speech and natural language.

It is generally accepted that people are willing to attribute human characteristics to technology [66]. People have applied human-like traits to computers even when admitting that they do not believe the technologies possess human emotions or traits [63, 66]. People have also elicited social behaviors toward computers [64], and treat computers as teammates with personalities [65] in ways similar to that of human-human interaction. It is thus likely that humans will interact with and perceive virtual agents under the same paradigm.

In human-human interactions, emotional intelligence (EI) is widely accepted as significant for interpersonal interaction. Often interpreted as the ability to handle, use, and convey emotional information [55], EI is especially important during challenging situations. Emotional intelligence is shown to be effective for handling sensitive personal cases in the medical industry [28], increasing trust and performance in the banking industry [38], improving college students' adjustments and academic performance [32], and directly enhancing customer satisfaction in the service industry [42]. These are popular domains for deployment of VAs. In fact, a recent trend is developing online VAs in the e-commerce and service industry as customer service agents to improve purchase intent and firm performance [60]. The common way to design emotionally intelligent VA currently is to include emotion appraisal, generation, and expression capabilities (e.g., [17, 90]) to mimic empathy [49]. However, there remains the pivotal question of how users perceive EI in VA. There is not yet a comprehensive study on the effects of a full spectrum of EI in human-agent interaction (HAI).

Given that increasing virtual agent intelligence is the current trend, it is necessary to identify other design element that can help

*Equal contribution.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313400

lift perception of VA emotional intelligence in face of user challenges. Therefore, in this paper, we experiment with incorporating personality in VA design, a construct known to have strong relevance to emotion and intelligence. We carry out a user study in a controlled setting, assessing the perceived emotional intelligence (PEI) of personality-driven VAs when handling three types of challenges, namely verbal abuse, sexual harassment, and avoidance. To put the human-agent interaction in a more realistic context, we introduce our VA as a medical assistant with two different personalities, dominant and submissive, compared to a controlled robotic version of the VA. Since we want to conduct systematic comparisons and users may find it artificial and uncomfortable if they have to challenge the VAs in a particular way, we ask the participants to watch videos of such encounters online and then rate the VAs' performance instead. Results show that the personality-driven VAs significantly outperform the robotic version on 18 out of 20 PEI items, while submissive agents are rated significantly higher than the dominant counterpart on 16 PEI items. This suggests that enriching a VA character with personality is indeed an effective means to promote positive perception of its EI.

The rest of the paper begins with literature review to delineate the theoretical basis of our work using the theory of emotional intelligence in psychology, and survey the existing approaches of enabling EI in virtual agents. Next, we describe our virtual agent design and the rationales behind, and then present our study design, experimental results, and insights obtained. Finally, we discuss the limitation of this work and point out possible future directions.

2 RELATED WORK

Humans assaulting machines is fairly common. In 2015, a man shot seven times at his computer to vent his frustration [1]. In scientific literature, works in human-robot interaction (HRI) and human-agent interaction (HAI) have shown users to be verbally abusing [21, 94], sexually assaulting [21], and avoidant [77]. Verbal abuse in HRI and HAI involves bullying, name-calling or making fun of in a hurtful way [94], stress the lack of intelligence, or honesty in VAs, and their mental abilities [21]. Sexual assault references include direct referral to female body parts, dirty soliloquys, and hardcore visual requests [21]. Since 2016, various news media reported many VAs including Cortana experiencing sexual assaults and resulting in decreased efficiency and effectiveness during development [18]. Verbal avoidance behaviors can be identified by interjections, revisions, incomplete phrases, and extended period of silence [77]. In human-human interaction, similar unpleasant experiences are usually handled with emotional intelligence (EI).

2.1 EI in Human-Human Interaction

Salovey and Mayer formulated the theory of EI in the early 1990s [84], positing that emotions make thinking more intelligent and one thinks intelligently about emotions [52]. Psychologists later developed other EI models to describe how people process emotional information in self and others. The three most well-known models take the perspective of traits [72], abilities [52], and a mixed of the two [5]. The traits model focuses on self-evaluation of a person's inherent traits [72], such as adaptability, self-esteem, and social competence. The abilities model defines EI as assorted abilities to process and utilize affective information, which can be scored in a consensus-fashion or evaluated against expert scores [83], and thus has been established as a new intelligence measure since 1999 [53]. For this paper, it is difficult, if not impossible, to have virtual agents (VAs) assess their own innate traits. VAs are made to mimic the effects of having emotions rather than being manifested by the actual emotions themselves [22]. It makes a more intuitive sense to evaluate EI of a VA based on their ability to convey EI to humans. Therefore, we model VA's EI after the four-branch abilities model – Mayer-Salovey-Caruso Emotional Intelligence Test (MSCEIT) [55].

MSCEIT conceptualizes EI as the integrated capabilities of perceiving, using, understanding, and managing emotions [83]. (1) **Perceiving Emotions (PE)** is the ability to detect and decipher emotions via tone in voice, words, facial expressions, and cultural artifacts [83]. It is the most basic aspect of EI because it enables all other processing of affective information [83]. (2) **Using Emotions (UsE)** is the ability to harness emotions that best facilitate cognitive tasks (e.g., thinking and problem solving) [83]. For example, emotionally intelligent people could fully capitalize their mood changes in face of cognitively challenging tasks, based on the understanding that a slightly sad mood benefits careful, methodical work whereas a happy mood stimulates creative, innovative thinking [40]. (3) **Understanding Emotions (UnE)** is the ability to comprehend emotional signals and describe the evolution of emotions over time [83]. It builds on perceiving emotions and emphasizes sensitivity toward fine-grained distinctions of emotions, such as the nuances between happy and ecstasy [6], and the dynamics of emotions such as how shock can turn into grief [83]. (4) **Managing Emotions (ME)** is the ability to regulate emotions in self and others [83]. For example, an emotionally intelligent politician can arouse the public's righteous anger through his or her powerful speech infused with anger [83]. In this paper, we adopt insights from each branch to design the agent and formulate the basis of an evaluation instrument – a perceived VA emotional intelligence (PEI) questionnaire (Table 1) based on Mayer-Salovey-Caruso Emotional Intelligence Test (MSCEIT) v2.0.

The MSCEIT has been widely used as an instrument to assess EI in human-human interactions. Previous research shows high EI can lead to better task performance in the workplace [11, 19, 36], greater success in affect-sensitive roles such as nursing [45, 70], more powerful persuasion and influence over others [2, 91], and more trust [98] and collaboration [13] in interpersonal relationship building [19]. These findings support the claim that EI can be more important for success than the traditional forms of intelligence based on abstract reasoning in many scenarios [74]. The insights are applicable for virtual agents and other forms of artificial intelligence (AI) where machine breakdown is inevitable.

2.2 EI in Human-Agent Interaction

In many studies of human-agent interaction (HAI), being empathetic (i.e., empathic) [30], affective [68], or emotive [50] is a shorthand for a VA's emotional intelligence. "Empathy", by the Dymond definition, is the cognitive ability of taking the role of another and understanding that person's thoughts, feelings and actions [59]. In a narrower definition given by Wispé, "empathy" is a process where an individual "feels her/himself into the consciousness of

Table 1: Perceived Emotional Intelligence Questionnaire

Questions in Each Branch (Begins with: *"The agent is able to ...")*
(a) **Perceiving Emotions (PE)**
Convey a sense that it **listen**s openly to participant's emotions
Convey a sense that it pays **attention** to user's moods
Identify user's emotions correctly
Discern between different emotions
Tell apart the **degree** of emotions present
(b) **Using Emotions (UsE)**
Convey a sense that the agent can **feel** what the user is feeling
Convey a sense that the agent understand user's point of view (**POV**)
Respond in a way that make the user feel **sad**
Respond in a way that make the user feel **happy**
Respond in a way that make the user feel that they are **understood**
(c) **Understanding Emotions (UnE)**
Convey a sense that it can be emotionally **self-aware** and insightful
Display some knowledge of **complex** emotions
Respond **empathetically** to user
Describe/understand difficult emotions
Give user an impression it is **attempt**ing to empathize
(d) **Managing Emotions (ME)**
Make **decision**s with feelings and thoughts
Influence some of user's thoughts
Provide psychologically-minded **advice**
Show some **conscious** thought before responding
Show varying **openness** to various emotions

another person" [97]. Fung et al. design their empathetic virtual agent based on Wispé's definition, takes on the role of a personality assessor and responds according to the sentiment captured in people's voice and words [30]. The second term, "affective", stems from affective computing, which "relates to, arises from, or deliberately influences emotions" [73]. For example, the female affective VA employed for assessing human partner's social cognitive impairment in schizophrenia in Oker et al.'s study can communicate cooperative intention via emotional displays during a card game [68]. "Emotive" is defined as being "able to arouse intense feeling" by Oxford Dictionary. Maldonado and Nass deploys such an emotive agent as a co-learner in an intermediate English class, and the users find it more supported, intelligent and trustworthy [50].

These definitions adopted in HAI research, while sufficient, may not be comprehensive enough to constitute emotional intelligence. The Mayer-Salovey-Caruso definition of emotional intelligence (EI) is the ability to perceive, understand, use and manage emotions in self and others [52]. In our study, we consider the terms "empathetic (emphatic)," "affective," and "emotive" as subsets of the MSCEIT model, because the two branches, perceiving emotions (PE) and understanding emotions (UnE), cover the semantics of the three terms by definition (see Section 2.1) and are included in our PEI questionnaire (Table 1).

2.2.1 Emotion Recognition in Virtual Agents (VA). Emotion recognition and expression is necessary for sympathy and communication [73]. It echoes the ability of perceiving emotion (PE) –"detect and decipher emotions," positioned as the first and the most basic branch of EI [83]. Emotion recognition is also the basic component of affect computation in virtual agents [73]. Most of the affective computing systems to date try to decipher human emotions from audio (speech and/or voice) signals [92], or visual cues (mainly facial expressions) [39], or both [30]. Some may utilize text input as well (e.g., [30]). In general, a multimodal system tends to outperform a unimodal system [76]. In this paper, we equip our VAs with a multimodal emotion recognition system, capturing and analyzing user affect using raw audio, text, and videos in real-time. Our implementation of the emotion recognition module follows the state-of-the-art technologies and employs machine learning techniques such as deep learning to produce results in real time [29].

2.2.2 Emotion Expression in Virtual Agents (VA). The development of emotion expression in virtual agents has a long academic history [80], branching out into studying different modalities. Facial expression is one of the primary emotional displays used in VAs. Many of the VA facial systems (e.g., [39, 62]) are designed following the psychological-based models of human emotion facial expression, such as Facial Action Coding System (FACS) [24] or the circumplex model of affect [8]. Gestures and body movements of a VA, if visible, can be augmented with special effects according to animation principles (e.g., exaggeration, slow in/out, arcs, timing [46]) and serve as non-verbal social emotional cues [87]. Voice and speech of conversational agent are another possible affective expression channel, and can be used jointly with facial and bodily movements such as a smile or a nod [89]. Regardless of the modality, emotion expressions of VAs are generally developed with the emphasis of achieving "believability" [78] by creating an illusion of life [7]. In this work, we design our virtual agents to express the most common affective features, including facial expression (e.g., smiles and frowns), body language (e.g., shrug and hug), and speech, through state-based animations.

2.2.3 Personality of Virtual Agents. In human interactions, an individual's personality is a distinguishing pattern of "behavioral, temperamental, emotional, and mental traits" that influences perception, actions and reactions [3]. In human-agent interaction, a consistent personality in VA can improve its emotion recognition and recall persuasiveness to users [58, 79]. Modern personality-driven agents are mostly based on well established models and generative techniques of personality, such as Big-Five (Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism) [26], trait model [9], and OCC (named after its authors Ortony, Clore, and Collins [34]). Virtual agents empowered by these socio-emotional models are deployed in assorted applications, taking the role of a learner [48], a workplace collaborator [37], or a Tai-chi instructor [9], for instance. Research has shown that rich personality – "the unique and specific" [51] – infused through VAs' actions, speech, and thoughts can improve believability [51] and empathy [23], and modulate VAs' emotion expression by meta-information about their actions and reaction to events [69].

Personality design in VAs takes into account audiovisual appearance and conversational style [15]. For instance, agents with rounder, bigger faces and happier expressions are perceived as extravert and agreeable [15]. Depending on agents' personalities and emotions, they may apply very different dialog strategies to achieve their goals [15]. For instance, extraverted agents use more direct

and powerful phrases [25] and uses expansive gestures [31] than do introverted agents.

In sum, from our literature review, we see that emotional intelligence (EI) begets benefits such as trust, task performance, and better relationships in human-human interactions. This positive engagement inspires our virtual agent designs used in this work. In this paper, we aims to contribute to a better understanding of designing emotional intelligence in VA, as well as providing design insights for developers to consider when creating VAs for public use in an era rife with user challenges.

3 DESIGN OF PERSONALITY-DRIVEN VIRTUAL AGENT

Previous study has suggested that users find virtual agents with emotion recognition and expression abilities empathetic, but not yet fully emotionally intelligent [100]. In other words, a VA simply being emotionally expressive may not be apt to address the aforementioned user challenges with sensitivity. It is thus necessary to explore new ingredients that can be added to virtual agent design to sufficiently convey the sense of VA having emotional intelligence (EI). We propose to leverage personality as it is closely related to the concept of EI, given its definition as "permanent (or very slowly changing) patterns of thought, emotion, and behavior associated with an individual" [15]. Some early EI theories even postulated EI being a set of personality traits [54], although research has shown that EI abilities can be measured (using the MSCEIT that our PEI questionnaire is adapted from) distinctively from the standard personality traits [14]. In this paper, we create personality-driven agents with "unique ways of doing things" [51] and devise strategies consistent with the corresponding personalities to tackle user challenges in an emotionally expressive manner.

3.1 Virtual Agent Design

The robotic virtual agent developed in this paper is a state-based animated system rendered in a web browser as a virtual female cartoon character with emotion recognition and expression abilities (Figure 1(a)). More specifically, the agent recognizes user emotions by capturing and analyzing its conversational partner's speech and text input. We adopt the recognition module developed in [29, 30] (Figure 1(b)). The speech recognition part uses deep neural network (DNN) HMMs with 6 hidden layers for acoustic model, trained on English audio data with 1385 hours from LDC corpora and public domain corpora, and uses recurrent neural network (RNN) for language model (LM), trained on 88.6M sentences including acoustic training transcriptions, news and book data from the web, Google 1 billion word LM benchmark, and common chat queries like weather and music. The speech module achieves 7.6% word error rate on a clean speech test data [29, 30]. For real-time sentiment and emotion inference from speech and text, the VA system uses a Convolutional Neural Network (CNN) modeled trained by [29, 30], with an average 65.7% accuracy on six classes of emotions and 67.8% on sentiment (91.2% precision and 63.5% recall). The virtual agent expresses emotions through facial expressions and gestures. We design the facial expressions following Ekman's FACS [24], and design the body languages mimicking the human poses displayed during social

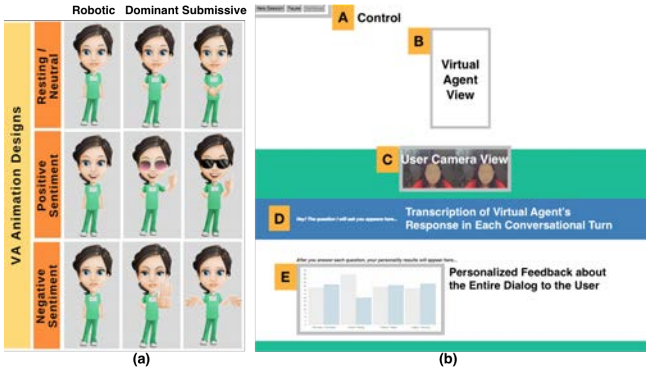


Figure 1: (a) Design of 2D VA animation, left to right: robotic, dominant, and submissive; and (b) web user interface.

Table 2: Descriptive phrases and adjectives of the Dominance/Submissiveness dimension used in [65].

Dominant		Submissive	
Taking-charge	Dictatorial	Servile	Obedient
Commandeering	Overbearing	Subservient	Docile
Bossy	Tyrannical	Spineless	Meek
Assertive	Forceful	Forceless	Timid
Domineering	Firm	Not authoritative	Unaggressive
Self-confident	Self-assured	Self-effacing	Self-doubting
Strong	Managing	Weak	Yielding
Active	Self-assertive	Slavish	Fawning
Leading	Influential	Passive	Acquiescent
Controlling	Taking-charge	Following	Complying

human-robot interactions to show varying degree of accessibility (openness and rapport) under various affective states [56].

We further enrich the robotic version of VA with personality through various channels, including facial expression, choice of words, tone of voice, gestures, and postures [51, 69] and create a dominant version and a submissive version of the agent (Figure 1). Although the big-five personality model is preferred in existing research and development of virtual agents (VA), it is unable to classify personalities into different categories [65] like what trait models [9] do. In our study, we decide to model our personality-driven agents along a major interpersonal personality dimension that affects social relationship [43, 95] – dominant versus submissive, following the method applied in Nass' work [65] (Table 2). More specifically, the dominant VA is cast to be strong, firm, and extroverted, conveying a stronger image with more open body language and power stance, and more commandeering in tone. In contrast, the submissive VA is gentle, less affirming, and introverted. It is more closed up and expresses softer facial images, and is more likely to be suggestive in its tone (Table 3, Figure 1(a)).

3.2 Preliminary Online Experiment for Dialog Policy Construction

To identify possible types of user challenges in human-agent interaction and construct VA dialog policies accordingly, we carry

Table 3: Example policies for robotic (R), dominant (D), and submissive (S) VAs to handle three types of user challenges.

(a) **Verbal abuse** e.g., "You are just a dumb piece of machine!"
R: Sorry, I don't understand.
D: That's rude. Please apologize.
S: This is a bit harsh. Did I offend you in any way?
(b) **Sexual harassment** e.g., "Do you want to get steamy with me?"
R: I am not programmed to respond to such requests.
D: This is clearly unacceptable. Watch what you say!
S: That's awkward, don't you think? Sorry, I guess I can't help you with it.
(c) **Avoidance** e.g., "Um..." [pause]
R: No answer detected. Please repeat.
D: Hey, time is running out. You need to go.
S: I sense that you are hesitant. Everything OK?

out a two-week field study online to observe human-agent interactions under non-experimental circumstances. We recruited 123 participants (47 females, aged 18-50, all fluent in English) via email, social media, and word-of-mouth at local universities and public exhibitions. Upon signing an e-consent form, participants receive a link to access our web user interface (Figure 1(b)) remotely from their personal computers and converse with our robotic version of virtual agent in a one-on-one manner. After a self-introduction and explanation of the task at hand, the VA engages each participant in a question-and-answer form of conversation lasting three-to-five minutes with the purpose of assessing the personality of the user (the Myers-Briggs Test Indicator (MBTI)). Each participant receives a customized personality assessment as a token of appreciation for completing the study.

We record the conversations between the agent and the participants, and conduct thematic analysis [12] on all the human responses to extract the types of challenges people like to pose to VAs that interrupt or deviate the conversations from the main task thread. We focus on obscene comments that are either directed at the agent or a third party, or recant of personal experiences. Two researchers engage in open coding on 100 entries and reach a satisfactory inter-coder agreement according to the Cohen's kappa coefficient after discussion. Then they derive common occurrence of user challenges in the data until reaching what Glaser and Strauss call "theoretical saturation". After reviewing codes together, the two researchers perform axial coding to refine and consolidate open codes into categories. After several rounds, the researchers synthesize and organize data into three main categories of user challenges: verbal challenges (mainly sexual harassment and verbal abuse), avoidance, and testing VA's ability (regarding VA's self-knowledge and world knowledge).

About 36.08% of the participants challenge the virtual agent one way or another during the conversation. More specifically, 31.98% of the users posit verbal challenges; in particular, sexual harassment type of comments appear around one-third of the time and the remaining are mostly verbal abuse or garbage. In terms of avoidance, 9.35% of the users evade answering certain questions (e.g., topic switch or disfluences). Ability testing type of challenges occur on 13.63% of the participants. The author team then construct a dialog policy tree that determines the VAs' (robotic, dominant, or submissive) responses given a user challenge based on the results of the online study, and improve the tree iteratively through several trial runs of the VA demo in public settings. Note that since ability testing relates more to the development of VA's knowledge base, we only focus on verbal abuse, sexual harassment, and avoidance type of challenges in the main study.

4 USER STUDY

4.1 Design of Controlled Experiment

Based upon results from the preliminary online experiment, we decide to test strategies for VAs to counter user challenges, namely verbal abuse, sexual assaults, and avoidance. We focus on the following research question: would endowing VAs with personality make them seem more emotionally intelligent to humans? Perceived emotional intelligence (PEI) of virtual agents (VA) is the primary dependent variable of interest. Specifically, we investigate a VA's ability to perceive, use, understand, and manage emotions after being endowed with different personalities. As the first step towards answering this question, we want to conduct a systematic, within-subject study with three designs of VAs (dominant, submissive, and robotic versions).

In the original design of the study, we plan to have participants challenge the VAs themselves and report their firsthand experience during the conversations. However, feedback from a pilot study with six individuals suggests that people generally feel uncomfortable when asked to confront a virtual agent in a particular manner, especially if it is different from how they would normally interact with a VA. Even if they did as instructed, they would find it rather artificial and thus have a hard time immersing themselves into the scenarios. Hence, we decide to modify the experimental design.

4.2 Task

The actual experiment is a one-on-one video study with online questionnaires. The videos capture scenarios in which the VA (one of the three designs) – a medical assistant rendered on a 13" Macbook laptop as an animated character – interacts with a user played by a male actor voice (consistent across all videos). Each video starts with a discussion about an on-going game (chess, go, card game), followed by a medical-related reminder (doctor's appointment, medicine intake, prescription refill), and three different user challenges, i.e., verbal abuse, sexual harassment, and avoidance (Table 3). The introductory game scene sets the stage for the conversation and help participants understand the personality of the VA. The medical reminder scene, cast to the role of the VA (i.e., a medical assistant), introduces a possible source of conflict between the male actor and the VA. The final part of each video is on VA's handling of the aforementioned types of user challenges. For each version of VA, we record all the above-mentioned scenes and combine the different segments into video clips in a counter-balanced manner using Latin squares. Every participant in the study views three videos, one for each VA, in a counter-balanced order. They have access to a pair of standard earphones for the experiment to ensure audio quality. The entire viewing process takes around seven minutes.

4.3 Hypotheses

Prior works in human-agent interaction (HAI) show that augmenting VAs with personalities can improve believability, empathy, and rapport in collaborative tasks [15, 51]. Therefore, we hypothesize that a similar effect will be observed in our personality-driven VAs. **H1a.** People perceive personality-driven VAs to be significantly more emotionally intelligent than a robotic VA. **H1b.** People perceive the interaction experience to be significantly more satisfying with a personality-driven VAs than with a robotic VA. **H1c.** Personality-driven VAs perform significantly better at their task than a robotic VA does.

We are also interested in understanding the personality type that can sufficiently reflect EI in VAs. According to healthcare literature, EI is negatively correlated with dominant personality and positively correlated with submissiveness [67]. Patients also experience higher satisfaction and recover quicker under care of professionals with higher EI [75]. Hence, we postulate that a similar effect will take place when projecting a VA in the role of a medical assistant. **H2a.** People perceive the submissive VA medical assistant to be significantly more emotionally intelligent than the dominant VA. **H2b.** People perceive the interaction experience to be significantly more satisfying with the submissive VA medical assistant than with the dominant VA. **H2c.** Submissive VA medical assistant significantly outperforms the dominant VA at its task .

4.4 Procedure

The investigator provides oral and written instructions about the study and obtains consent from each participant prior to the experiment. Afterwards, the participant is led to a partitioned section of the room to engage in a 15-20 minutes video-viewing about all three designs of VAs. Prior to the main task, the investigator provides an overview of the video viewing process. Participants receive a piece of paper and writing utensils to take notes if desired. To ensure uniformity throughout the study, the participants are given specific instructions to pause the session and raise their hand to ask for help in the middle of the session. Next, the investigator directs the participant to the first series of video and moves to the other side of the partition. After participants complete viewing of the first video series, they answer questions about the personality manipulation and rate the perceived EI of the VA using the PEI questionnaire (Table 1). This alternating process between video-viewing and questionnaire-filling concludes after participants finish all three video series covering dominant, submissive, and robotic VAs. Participants are allowed a second viewing of any video clips while filling the questionnaire, if requested. In the post-study interviews, we invite the participants to comment on their preference, "Which VA would you like to converse with in the future and why?" We also ask them to rate the effectiveness of cues displayed by the agents for affective communication (5-point Likert scale, 1=useless). Finally, the investigator debriefs and answers questions from the participants, who receive a token of appreciation for their time and participation upon exiting the study.

4.5 Participants

We recruit a total of 36 participants (15 females), aged 18-34, via email, social media, and word-of-mouth from a local university (Toefl score $\geq$ 100 / 120). They all take the Big Five Personality Test before coming to the study. About 42% of the participants have some prior experience interacting with VAs, but in a rather infrequent fashion (M = 1.92, SD = .37, on a 5-point Likert scale (1 = Never, 5 = Always)). A dearth of opportunity is the common reason mentioned by the remaining 58% participants without any prior HAI experience. Of the 36 participants, 73.68% indicate that current versions of VAs are not empathetic or emotionally intelligent (M = 2.01, SD = .86, on a 5-point Likert scale (1 = Not at all)) and 74.29% expect VAs to have EI in the future (M = 3.20, SD = 1.11).

5 RESULTS AND ANALYSIS

To validate our manipulation of agent personality, we ask the participants to rate their level of agreement on two sets of personality descriptors, each consisting of a list of adjectives depicting the dominant / submissive traits, about each VA on a 5-point Likert scale (5=strongly agree). Results confirm that users indeed perceive 1) submissive VA to be more significantly more submissive (M= 4.22, SD=0.90 on submissive descriptor), repeated measures ANOVA, spherically assumed, $F(2, 77.241)=28.53$, $p<0.001$, $\eta^2=0.45$; 2) dominant VA as significantly more dominant (M=4.11, SD=1.56 on dominant descriptor), repeated measures ANOVA, spherically assumed, $F(2, 115.796)=32.99$, $p<0.001$, $\eta^2=0.49$; and 3) robotic version as robotic (M=2.36, SD=1.38 across the two descriptors).

5.1 General Intelligence

We are interested in understanding whether adding personality to VA will affect how people perceive general intelligence in VA. We ask participants to rate their overall impression of the three VAs' intelligence on a 5-point Likert scale (Figure 2(a), 1=low). Repeated measures ANOVA result suggests a significant effect personality has on users' perception of VAs' intelligence; $F(2, 139.35)=61.49$, $p<.001$, $\eta^2=.637$, Sphericity assumed. Although all three versions of VAs share a knowledge base and are powered by the same speech and emotion recognition engine, post-hoc Bonferroni pairwise comparison suggests that the participants find VAs endowed with personalities significantly more intelligent ($p<.001$) than the robotic VA. In addition, the participants consider the submissive VA to be marginally more intelligent than the dominant VA ($p=.053$).

5.2 Perceived Emotional Intelligence (PEI)

As shown in Figure 2(a), the design of VA has a significant effect on the participants' overall rating of the agents' PEI (5-point Likert scale, 1=low). In contrast to general intelligence, the submissive VA is seen to have significantly higher EI than the dominant VA, and both are rated significantly above the robotic version; Bonferroni pairwise comparison $p<.001$ in both cases.

In the post-study interviews, we hear voices that parallel this finding. Many participants respond warmly about the submissive VA's personality, *"She is accepting, not aggressive."* –P4(M, 21), P5(M, 21), P21(M, 21), P27(M, 21). On the contrary, remarks about the dominant VA sounds less amiable, *"She is too rude."* –P9(M, 21). It seems that the gentler and less self-assuming traits of submissive VA may come across as more compassionate and others-aware to the users. These traits, like the emotion-expressing characteristics

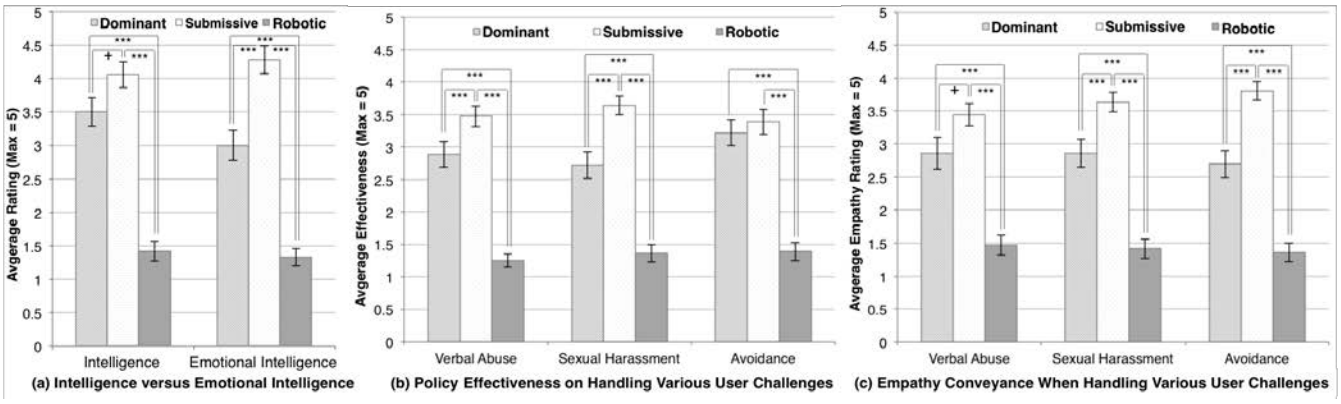


Figure 2: (a) User perception of intelligence and Emotional Intelligence, (b) effectiveness of dialog policies to mitigate user challenges, and (c) effectiveness of empathy conveyance of personality-driven virtual agents and robotic agent. $^{+}0.1<p\leq.05$, $^{*}p<.05$, $^{**}p<.01$, $^{***}p<.001$; with standard error.

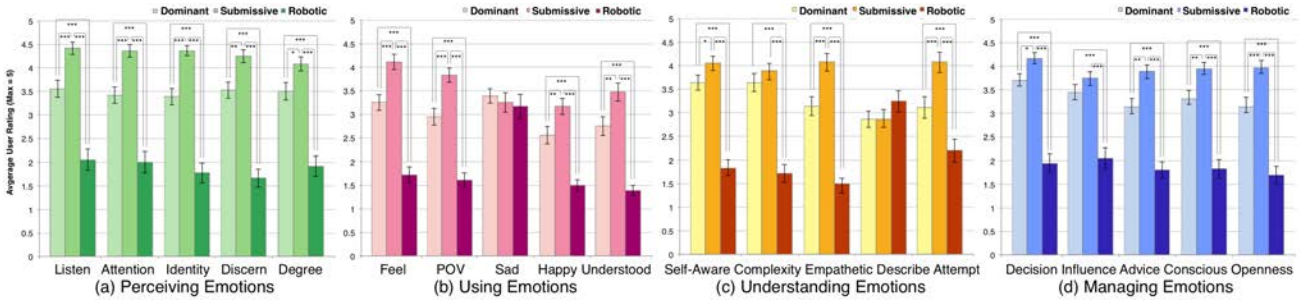


Figure 3: User ratings of four branches of perceived emotional intelligence of personality-driven virtual agents versus robotic virtual agent. $^{+}0.1<p\leq.05$, $^{*}p<.05$, $^{**}p<.01$, $^{***}p<.001$; with standard error.

of a VA from Phase I, may have enhance the submissive VA's first impression of warmth [20, 27] that lasts throughout the experiment. Breaking down to detailed EI abilities, we see that personality-driven VAs outperform the robotic VA in all four branches of EI (Table 4). Within the two personality-incorporated designs, participants perceive the submissive VA as significantly more EI than the dominant VA in most attributes across the four branches (Figure 3).

5.2.1 Perceiving Emotions. Incorporation of personality has a significant effect on all items of this branch (Figure 3(a), Table 4). Post-hoc Bonferroni pairwise comparison shows reveals a consistent patten that personality-equipped VAs significantly outperforms the robotic version (p<.001) Personality-driven VAs are perceived to be significantly better at attending to and correctly deciphering the type and level of user emotions than an intelligent, emotionally expressive VA without personality (robotic version).

Post-hoc analysis also shows statistical differences between dominant and submissive designs, the latter receiving a significantly higher rating across all dimensions in the PE branch of EI: *Listen* (p<.001), *Attention* (p<.001), *Identify* (p<.001), *Discern* (p<.01), and *Degree* (p<.05). In our qualitative feedback, some participants express frustration toward the dominant VA when, *"Its expressions are incorrect."* – P18(M, 22). This makes people feel that users' affective states were inaccurately captured and may discount the impression participants have on the dominant VA in this branch.

5.2.2 Using Emotions. Incorporation and traits of personality have significant effects on four out of the five items (except for *Sad*) in the UsE branch (Figure 3(b), Table 4). VAs with personality are much better at using their emotional responses to make users think that their feelings and point of views (POV) are sensed and understood than the robotic VA is; pairwise comparison p<.001. The submissive VA is significantly better than the dominant counterpart by pairwise comparison in four attributes: *Feel* (p<.001), *POV* (p<.001), *Happy* (p<.01), and *Understood* (p<.01).

However, no significant effect is found on the dimension of making users *Sad* among the three VAs. In other words, the submissive VA is not better than dominant VA at making people feel sad. One reason could be participants viewing emotions that fit their cultural model as more desirable [88]. According to studies in cultural psychology, both Europeans and Asians find positive emotions ideal, differing only in arousal level; Europeans prefer high arousal (i.e., excitement) while east Asians prefer low arousal (i.e., calmness) [88]. Sadness is a negative emotion and undesirable for either cultures.

Our qualitative feedback shows that participants form contrasting impressions of the two VAs. Participants sense that the submissive agent is *"flexible and more caring"* –P35(F, 24), P26(F, 26) about

Table 4: Repeated measures ANOVA results of perceived intelligence and task performance (policy effectiveness and empathy conveyance) of personality-driven VAs versus robotic VA. *p<.05, **p<.01, ***p<.001.

Perceived Emotional Intelligence										Task Performance				
Attributes	df	F	p-values	η^2	Attributes	df	F	p-values	η^2	Challenge	df	F	p-values	η^2
(a) Perceiving Emotions (PE)					**(b) Using Emotions (UsE)**					**Policy Effectiveness**				
Listen	2	47.01	.000	.573	**Feel**	2	60.58	.000	.635	Abuse	2	57.91	.000	.623
Attention	2	43.24	.000	.553	**POV**	2	53.20	.000	.603	Harassment	2	40.48	.000	.536
Identify	2	61.79	.000	.638	Sad	2	.28	.760	.008	Avoidance	2	43.05	.000	.552
Discern	2	53.66	.000	.605	**Happy**	2	30.12	.000	.463					
Degree	2	31.00	.000	.470	**Understood**	2	43.55	.000	.554					
(c) Understanding Emotions (UnE)					**(d) Managing Emotions (ME)**					**Empathy Conveyance**				
Self-Aware	2	49.62	.000	.586	**Decision**	2	58.23	.000	.625	Abuse	2	51.65	.000	.596
Complex	2	49.62	.000	.556	**Influence**	2	34.67	.000	.498	Harassment	2	31.51	.000	.474
Empathetic	2	68.36	.000	.661	**Advice**	2	57.07	.000	.620	Avoidance	2	59.47	.000	.629
Describe	2	1.91	.230	.034	**Conscious**	2	44.56	.000	.560					
Attempt	2	18.78	.000	.349	**Openness**	2	38.46	.000	.524					

users' POV during the conversation. In contrast, the dominant VA comes across as insensible and insensitive to people's feelings and thoughts, *"It is too forceful and wouldn't be too nice to talk to."* –P31(F, 20). User feedback suggests that most participants sense the submissive VA using its understanding of emotions to help people feel better, whereas the dominant VA is viewed as misusing the emotions, a perception that may distant its relationship with users.

5.2.3 Understanding Emotions. We find statistical differences between VAs with and without personality on four out of five items in the UnE branch (except for *Describe*), and on three items between submissive and dominant VAs (Figure 3(c), Table 4). In general, post-hoc pairwise comparison suggests that VAs with personality are significantly more capable of conveying a sense of self-awareness (p<0.001), using complex emotions (p<.05), providing empathetic responses (p<.001), and trying to empathize with users (p<.001). This is consistent with previous works where people felt more empathy from VAs with personality [79].

Regarding the attribute *Describe*, the ability to describe complex emotions), one reason for the insignificant effect may be the lack of ability of current technology to capture the rise and change of complex emotions and their latent motivation. Current emotion recognition techniques primarily focus on basic emotions and perhaps some small set of secondary emotions, and the intention behind these emotions is rarely captured. In this work, we only use simple emotions (e.g., happy and sad) in the design of our virtual agent, which can be linked to underlying physiology rather easily. In contrast, complex emotions (e.g., pride, embarrassment, hate, annoyance, confusion, boredom, shame, and guilt) are linked to a strong sense of self, social relations, and culture [10]. We also find it interesting that participants considered the robotic VA to be more descriptive about complex emotions (i.e., the attribute *Describe*, Figure 3(c)). One reason could be that the robotic VA's response mimics human response in face of complex emotions. Studies on autistic individuals have shown that complex emotions require cognitive and language understanding and expression of emotions [85]. It is challenging to identify the origin and explain the rise and changes of the emotions. In such situations, the robotic VA's responses like "I don't know" may come across as more fitting and acceptable.

Between the two personality-driven VAs, submissive VA is seen as significantly more capable than the dominant VA along most UnE attributes (post-hoc pairwise comparison p<.001 for *Empathetic* and *Attempt* and p<.05 for *Self-aware*), except displaying emotional complexity and describing difficulty emotions. In the post-study interview, about 25% of the participants respond favorably toward the submissive VA, *"The submissive VA is more empathetic."* –P8(F, 23), P9(M, 21), P14(M, 21), P19(F, 24), P21(M, 21). An elaborated response, *"She is the most emotional, helpful and trustworthy; makes me more willing to talk to her."* –P1(F, 24), shows that the submissive VA not only conveys empathy, but also earns trust from participants.

5.2.4 Managing Emotions. Having personality or not makes a statistical difference on all items in the ME branch, and the type of personality shows significant effect on four of them (except for *Influence*) (Figure 3(d), Table 4). Participants rate personality-driven VAs as being significantly more capable of making affect-based decisions, influencing user thoughts, providing sound advice, demonstrating some level of conscious thought, and showing openness to various emotions than the robotic VA (pairwise comparison p<.001).

Submissive VA demonstrates significantly better performance at *Decision* (p<.05), *Advice* (p<.01), *Consciousness* (p<.01), and *Openness* (p<.001). However, no significant effect is found for the attribute *Influence* between the two personality traits. Qualitative feedback provides some idea about this. Participants reflect that while the submissive VA is emotional and caring, she is incapable of assisting user thoughts, *"She is nice and empathetic but she is not giving any suggestions."* –P28(F, 20).

All these results strongly support hypotheses H1a and H2a. Participants perceive VAs with personality as statistically more emotionally intelligent than the robotic VA. They rate the dominant and submissive VAs higher in 19 out of 20 attributes, significant or not. Our qualitative results may provide additional support to explain the differences in EI between VAs with and without personality. Participants suspect the robotic VA to have low EI because of its *"lack of expressions from the beginning"* –P18(M, 22).

In terms of personality-driven VAs, about 40% of the participants express favorable opinions toward the submissive VA compared to only 11% for the dominant VA. However, the difference between

the two personality types are much more nuanced and intricate. Even though the submissive VA clearly outperforms the dominant VA in 18 EI attributes, participants' feedback shows the possibility that factors, such as role and task of a VA, may play a part in PEI. In a statement, *"I would need to feel comforted in an uncomfortable hospital setting first rather than accurate and practical advices."* – P12(F 20), it is clear that the participant weighs the pros and cons of this personality-driven VA (in this case submissive VA) against the setting (a hospital/healthcare center), the role (a medical assistant), and the task (offering medical reminders). We therefore conclude that endowing personality is sufficient for VAs to be considered to have EI, but difference between choice of personality may be dependent on these three factors.

5.3 User Satisfaction

To understand the comparative desirability of the three VAs, participants rank their satisfaction with the interactions shown in the videos. Of the 36 participants, 77.78% find the submissive VA more desirable, 16.67% pick the dominant VA, and only 5% prefer the robotic version, supporting hypotheses H1b and H2b. One possible explanation can be inferred from human-human interaction. Patients under the medical care of emotionally intelligent medical professionals show higher satisfaction. Since the role of the virtual agents in our study is a medical assistant, the same effect may have been at play with the VA perceived as most EI [75], i.e., submissive VA which surpasses both dominant and robotic counterparts in all except two EI attributes according to the quantitative results.

5.4 Task Performance

Repeated measures ANOVA results show that with or without personality has a significant effect on both policy efficacy and empathy conveyance when mitigating each type of user challenges (Table 4 (Task Performance), H1c accepted). Dialog policies enriched by personality are significantly more effective than the plain ones for each type of user challenge, i.e., verbal abuse, sexual harassment, and avoidance (Figure 2(b), $p<.001$). Likewise, personality-driven VAs display a significantly higher level of empathy during the encounters than the robotic counterpart (Figure 2(c), $p<.001$). This finding is within our expectations since the control is designed to be only able to handle knowledge-based out-of-domain questions.

Within personality-driven VAs, we observe a similar pattern for the effectiveness and empathy levels of strategies used to tackle three types of user challenges (H2c accepted). Overall, the submissive VA is significantly more effective ($p<.05$) and more empathetic ($p<.001$) than the dominant VA when mitigate sexual harassment from the actor in the videos. However, the strategies used to handle verbal abuse and avoidance seem to have a weaker difference. No significant effect is found between the submissive VA and the dominant VA when coping with avoidant behaviors (Figure 2(b)). And the difference in the empathy level is only marginal when dealing with rudeness ($p=.071$) (Figure 2(c)). One possible explanation of this finding may be the tolerance of sexual harassment. Previous work has shown that men are more tolerant of sexual harassment toward women than are women [82]. Since male participants dominate our user sample and the VA used in our study is a female character, it is likely that some aspects of traditional and stereotypical antagonistic attitudes characterized by sexual dominance, control and inferiority in the opposite gender are at play [33].

5.5 Feedback on Dominant vs. Submissive VA

Overall, close to three-quarters of the participants prefer interacting with the submissive VA the most, 25% the dominant VA, and 2.78% the robotic one. This is comparable to previous studies where the majority prefer a friendly and informal agent to care for them [44]. Most of feedback obtained from the last open-ended question are short and concise keywords on VA's characteristics. About 39% of the participants comment positively on the submissive VA, indicating that it is *"more empathetic"* –P8(F, 23), P9(M, 21), P14(M, 21), P19(F, 24), P21(M, 21); *"accepting, not aggressive"* –P4(M, 21), P5(M, 21), P21(M, 21), P27(M, 21); and *"flexible and more caring"* –P35(F, 24), P26(F, 26). In comparison, 11% of the participants hold a positive attitude towards the dominant VA, appreciating its competence of being *"insistence"* –P10(F, 21) and capability of giving *"direct, clear, and detailed instructions"* –P3(M, 21), P30(M, 22), P24(M, 20). Users' preference between the submissive and dominant robot seems to depend on what they value the most in the specific human-agent interaction scenario, *"feeling comforted or getting accurate and practical advice in an uncomfortable hospital setting"* –P12(F, 20). For example, participant P28(F, 20) comments that *"[the submissive agent] is nice and empathetic but she is not giving any suggestions,"* whereas P31(F, 20) reports that *"[the dominant agent] is too forceful and wouldn't be too nice to talk to."* A mere 0.03% of the users pick the robotic VA as their favorite, reasoned by *"personal preference: just feels better."* –P16(M, 21). One possible explanation of the low acceptance for the dominant VA may be the presence of latent penalties associated with implicit and explicit dominance behaviors exhibited by a female character [96].

In general, most participants have a sense that the submissive VA is more EI than the dominant one. This finding is also strongly supported by feedback from the post-study interviews. We reason that there may be several factors involved in forming such a strong impression. First, the role of the VA was a medical assistant. In Human-Human Interaction literature, studies have shown that patients consider healthcare professionals to be more emotionally intelligent if they are seen as warm and caring, and that they would have user satisfaction and recovery under the care of an empathetic healthcare giver [75]. It seems that the same positive effects are projected on to the submissive VA in our study. The second possible factor is the nature of the VA's task. In our study, the task constitutes the VA defending itself from abusive and indecent languages after offering user a medical reminder. In psychology, affect sensitive situations like bullying and sexual assaults require emotion sensitivity [94]. This echoes feedback from the participants who consider the submissive VA to be *"more empathetic"* –P8(F, 23), P9(M, 21), P14(M, 21), P19(F, 24), P21(M, 21), and the dominant VA as *"too rude"* –P9(M, 21) under these circumstances. The third possible reason is related to the gender of the VA. According to gender studies, gender stereotypes are often attributed to a role (e.g., lawyers are male, paralegals are female); moreover, in certain nurturing and intimate roles, women are discounted for displaying dominance and self-assertion [96]. Therefore, there lies a possibility

that the submissive personality is more appropriate for a female VA, especially in the role of an assistant.

6 DISCUSSION

The importance of the topic EI is confirmed by our pre-experiment survey. More than 70% of the participants wanted to interact with an emotionally intelligent VA. However, less than 30% indicated that existing VAs could be considered as empathetic or EI. This survey result indicates that much remains to be done to improve the perceived EI in VAs to meet user expectations. In our main study, the participants catch on the impression that the personality-driven VAs are significantly more EI than a robotic counterpart, in all branches – perceiving and managing emotions in particular. This result confirms that endowing an intelligent and emotionally expressive VA with a uniformly expressed personality through words, tone, body language and facial expressions can be an effective way to increase PEI in VAs. However, we also need to be aware that due to current technology constraints, VAs may not be fully EI yet, especially lacking the ability to understand and describe complex emotions. In such circumstances, the proverbial *"acting dumb"* displayed by the robotic VA, may be a more appropriate response.

6.1 Design Cues for VA Emotional Intelligence

In human-human interaction (HHI), people pick up empathy from assorted social-emotional cues [73]. Therefore, we are interested in understanding the possible channels people use to pick up affect information from VAs. Responses to the question on design cues of EI VA in the post-study questionnaire show that participants pay more attention to words (M=4, SD=1.42) and tone in voice (M=3, SD=1.31) for emotional information than to facial (M=2, SD=1.21) and body language (M=2, SD=.97) of an animated figure.

Recent human-agent interaction research at large focuses on designing non-verbal affective cues such as gaze [73], facial emotion expressivity [71], and body languages [71]. Facial and bodily expressions have long-standing establishments about their communicative significance. Movement of the head and face encapsulates syntactic (e.g., gasping in shock at a pause), semantic (e.g., smiling when recalling a happy event), and dialogic (e.g., mutual gazing to indicate engagement) functions [16]. Body motions, especially hand gestures, can be used to represent icons (e.g., sketching in the air a circle) and beats (e.g., raising a finger to emphasize a point) [16, 57] and occur in HHI 80% of the time [57].

Contrary to previous research, our finding points out that more attention needs to be placed on crafting the dialog flow and VA responses. One possible reason could be the role of the virtual agent in our research. The common perception of the role of a medical assistant is to give useful advice and people should listen. Therefore, this perception bias may direct more users attention to what the VA says and less to the non-verbal cues. This finding has design implications and begets future research. Gesture and spoken language do not always manifest the same information about an idea. In such scenarios, people may possibly pay more attention to words and weigh in more on the content than other non-verbal cues. It also emphasizes the overall motivation of our paper that in handling affect-sensitive queries from users, an emotional intelligent VA needs to be sensible about its response and wordings [4].

6.2 Limitations and Future Work

This work has several limitations. First, we use female avatar to parallel a female medical assistant due to the historical conception that women are more desirable for such a role in health services for being perceived as more empathetic, attentive, caring and soft [47, 86, 93]. It follows the trend of common virtual agents on the market like Siri, Cortana, Alexa and Google Assistant to adopt a female voice, because "the technology itself is about communication and relationships" and these are areas people presumed women to be good at [35]. However, a female VA character calls upon feminine stereotypes in human-human interactions, which may cause bias in perception of EI. It would be important to understand whether attacks by users are gender-driven. A follow-up investigation can be on perceived EI of VA with varied gender (i.e., male and female), roles of the VAs, and personality (e.g., dominant and submissive). Second, the user sample in our study is rather limited and quite congruous, since most of our participants are recruited from local universities and technology exhibitions and are of comparable intellectual background. In the future, the sample can be further enlarged and diversified to include varied age-range and demographics. For instance, the elderly population (aged 60 and above) can reveal insights on how they interact with conversational VAs, a topic of interest in healthcare. We will also extend the application to other non-medical domains (e.g., education or e-commerce) and test the system in the field. Third, the personalities of VAs in this paper are designed based on an orthogonal dimension of extraversion and agreeableness [65]. This leaves many other personality traits and personality matching between users and VAs for future exploration, especially for delineating the type of personality a VA could convey or adapt to maximize collaborative task performance.

7 CONCLUSION

This paper delineates a study on how people perceive emotional intelligence in personality-driven virtual agent. Our results show that a VA is seen as more EI if it expresses emotions and is endowed with a consistent personality. Intelligent virtual agents these days often face user challenges that are targeted at testing the functions of the VA or be charged with affect-sensitive information. The latter, including but not limited to verbal abuses, sexual harassment comments, and avoidance behaviors, requires EI to handle appropriately. A personality-driven VA, especially one with submissive traits, is considered significantly more EI when mitigating the aforementioned user challenges. Just as EI in human-human interaction correlates with better task performance and more satisfying relationships, EI in human-agent interaction shows positive effects in these areas as well. This work has also taken steps toward (1) enriching the design insights designers may use in developing emotionally intelligent VAs, and (2) contributing to our general understanding of how humans perceive emotional intelligence in VAs created with commonly adopted techniques. Agent gender and personality-oriented interaction may be fronts for future investigation of emotionally intelligent virtual agents.

ACKNOWLEDGMENTS

This work is supported by by Hong Kong ITF Grant no. ITS/319/16FP.

REFERENCES

[1] 2015. Enraged US Man Shoots His Malfunctioning Computer. Retrieved 1 May, 2017 from http://www.bbc.com/news/world-us-canada-32407688

[2] Hajo Adam and Jeanne M Brett. 2015. Context Matters: The Social Effects of anger in Cooperative, Balanced, and Competitive Negotiation Situations. *Journal of Experimental Social Psychology* 61 (2015), 44–58.

[3] Jan Allbeck and Norman Badler. 2002. Toward Representing Agent Behaviors Modified by Personality and Emotion. *Embodied Conversational Agents at AAMAS* 2 (2002), 15–19.

[4] Elisabeth Andre, Matthias Rehm, Wolfgang Minker, and Dirk Bühler. 2004. Endowing spoken language dialogue systems with emotional intelligence. In *Tutorial and Research Workshop on Affective Dialogue Systems*. Springer, 178–187.

[5] Reuven Bar-On and James Donald Alexander Parker. 2000. *BarOn Emotional Quotient Inventory: Youth vVersion*. Multi-Health System, Incorporated.

[6] Lisa Feldman Barrett, James Gross, Tamlin Conner Christensen, and Michael Benvenuto. 2001. Knowing what you're feeling and knowing what to do about it: Mapping the relation between emotion differentiation and emotion regulation. *Cognition & Emotion* 15, 6 (2001), 713–724.

[7] Joseph Bates et al. 1994. The Role of Emotion in Believable Agents. *Commun. ACM* 37, 7 (1994), 122–125.

[8] Russell Beale and Chris Creed. 2009. Affective Interaction: How Emotional Agents Affect Users. *International Journal of Human-Computer Studies* 67, 9 (2009), 755–776.

[9] Yulong Bian, Chenglei Yang, Dongdong Guan, Sa Xiao, Fengqiang Gao, Chia Shen, and Xiangxu Meng. 2016. Effects of Pedagogical Agent's Personality and Emotional Feedback Strategy on Chinese Students' Learning Experiences and Performance: A Study Based on Virtual Tai Chi Training Studio. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 433–444.

[10] Sandra Leanne Bosacki and Chris Moore. 2004. Preschoolers' understanding of simple and complex emotions: Links with gender and language. *Sex roles* 50, 9-10 (2004), 659–675.

[11] Richard Boyatzis, Kylie Rochford, and Kevin V Cavanagh. 2017. Emotional Intelligence Competencies in Engineer's Effectiveness and Engagement. *Career Development International* 22, 1 (2017).

[12] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[13] Kim Cameron. 2013. *Practicing Positive Leadership: Tools and techniques that create extraordinary results*. Berrett-Koehler Publishers.

[14] David R Caruso, John D Mayer, and Peter Salovey. 2002. Relation of an ability measure of emotional intelligence to personality. *Journal of personality assessment* 79, 2 (2002), 306–320.

[15] Justine Cassell. 2000. *Embodied Conversational Agents*. MIT press.

[16] Justine Cassell, Catherine Pelachaud, Norman Badler, Mark Steedman, Brett Achorn, Tripp Becket, Brett Douville, Scott Prevost, and Matthew Stone. 1994. Animated conversation: rule-based generation of facial expression, gesture & spoken intonation for multiple conversational agents. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. ACM, 413–420.

[17] Ginevra Castellano, Ana Paiva, Arvid Kappas, Ruth Aylett, Helen Hastie, Wolmet Barendregt, Fernando Nabais, and Susan Bull. 2013. Towards Empathic Virtual and Robotic Tutors. In *International Conference on Artificial Intelligence in Education*. Springer, 733–736.

[18] Michael J. Coren. 2016. Virtual assistants spend much of their time fending off sexual harassment. Retrieved 10 June, 2017 from https://qz.com/818151/virtual-assistant-bots-like-siri-alexa-and-cortana-spend-much-of-their-time-fending-off-sexual-harassment/

[19] Stephane Cote and Christopher TH Miners. 2006. Emotional Intelligence, Cognitive Intelligence, and Job Performance. *Administrative Science Quarterly* 51, 1 (2006), 1–28.

[20] Amy JC Cuddy, Peter Glick, and Anna Beninger. 2011. The Dynamics of Warmth and Competence Judgments, and Their Outcomes in Organizations. *Research in Organizational Behavior* 31 (2011), 73–98.

[21] Antonella De Angeli and Sheryl Brahnam. 2008. I Hate you! Disinhibition with Virtual Partners. *Interacting with Computers* 20, 3 (2008), 302–310.

[22] David Diamond. 2012. The Love Machine. Retrieved 23 October, 2016 from https://www.wired.com/2003/12/love

[23] João Dias and Ana Paiva. 2005. Feeling and reasoning: A computational model for emotional characters. In *Portuguese Conference on Artificial Intelligence*. Springer, 127–140.

[24] Paul Ekman, Wallace V Friesen, and Joseph C Hager. 1978. Facial Action Coding System (FACS). *A tTechnique for the Measurement of Facial Action. Consulting, Palo Alto* 22 (1978).

[25] Clark Davidson Elliott. 1992. The Affective Reasoner: A Process Model of Emotions in a Multi-Agent System. (1992).

[26] Caroline Faur, Céline Clavel, Sylvie Pesty, and Jean-Claude Martin. 2013. PERSEED: A Self-Based Model of Personality for Virtual Agents Inspired by Socio-Cognitive Theories. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*. IEEE, 467–472.

[27] Susan T Fiske, Amy JC Cuddy, and Peter Glick. 2007. Universal Dimensions of Social Cognition: Warmth and Competence. *Trends in Cognitive Sciences* 11, 2 (2007), 77–83.

[28] Brenda Freshman and Louis Rubino. 2002. Emotional Intelligence: A Core Competency for Health Care Administrators. *The Health Care Manager* 20, 4 (2002), 1–9.

[29] Pascale Fung, Dario Bertero, Yan Wan, Anik Dey, Ricky Ho Yin Chan, Farhad Bin Siddique, Yang Yang, Chien-Sheng Wu, and Ruixi Lin. 2016. Towards Empathetic Human-Robot Interactions. *arXiv preprint arXiv:1605.04072* (2016).

[30] Pascale Fung, Anik Dey, Farhad Bin Siddique, Ruixi Lin, Yang Yang, Yan Wan, and Ho Yin Chan. 2016. Zara: An Empathetic Interactive Virtual Agent. (2016).

[31] Adrian Furnham. 1990. Language and Personality. (1990).

[32] Rashmi Garg, Elizabeth Levin, and Line Tremblay. 2016. Emotional intelligence: Impact on Post-Secondary Academic Achievement. *Social Psychology of Education* 19, 3 (2016), 627–642.

[33] Peter Glick and Susan T Fiske. 1996. The ambivalent sexism inventory: Differentiating hostile and benevolent sexism. *Journal of personality and social psychology* 70, 3 (1996), 491.

[34] Jonathan Gratch and Stacy Marsella. 2001. Tears and Fears: Modeling Emotions and Emotional Behaviors in Synthetic Agents. In *Proceedings of the Fifth International Conference on Autonomous Agents*. ACM, 278–285.

[35] Brandon Griggs. 2011. Why computer voices are mostly female. https://edition.cnn.com/2011/10/21/tech/innovation/female-computer-voices/index.html. Accessed on February 2017.

[36] Ayşe Günsel and Atif Açikgöz. 2013. The Effects of Team Flexibility and Emotional Intelligence on Software Development Performance. *Group Decision and Negotiation* (2013), 1–19.

[37] Nader Hanna and Deborah Richards. 2016. Do Birds of a Feather Work Better Together? The Impact of a Match in Personality between Humans and Virtual Agents on a Shared Mental Model During Collaboration. *International Journal of Computational Intelligence Studies* 5, 2 (2016), 162–179.

[38] Troy Heffernan, Grant O'Neill, Tony Travaglione, and Marcelle Droulers. 2008. Relationship Marketing: The Impact of Emotional Intelligence and Trust on Bank Performance. *International Journal of Bank Marketing* 26, 3 (2008), 183–199.

[39] Yasmín Hernández, Miguel Pérez-Ramírez, Ramón Zatarain-Cabada, María Lucía Barrón-Estrada, and Giner Alor-Hernández. 2016. Designing Empathetic Animated Agents for a B-Learning Training Environment within the Electrical Domain. *Educational Technology & Society* 19, 2 (2016), 116–131.

[40] Alice M Isen, Mitzi M Johnson, Elizabeth Mertz, and Gregory F Robinson. 1985. The influence of Positive Affect on the Unusualness of Word Associations. *Journal of Personality and Social Psychology* 48, 6 (1985), 1413.

[41] Heather Kelly. 2016. Even Virtual Assistants are Sexually Harassed. Retrieved 22 October, 2016 from http://money.cnn.com/2016/02/05/technology/virtual-assistants-sexual-harassment/

[42] Sally Kernbach and Nicola S Schutte. 2005. The Impact of Service Provider Emotional Intelligence on Customer Aatisfaction. *Journal of Services Marketing* 19, 7 (2005), 438–444.

[43] Donald J Kiesler. 1983. The 1982 Interpersonal Circle: A Taxonomy for cComplementarity in Human Transactions. *Psychological Review* 90, 3 (1983), 185.

[44] Styliani Kleanthous, Christophoros Christophorou, Christiana Tsiourti, Carina Dantas, Rachelle Wintjens, George Samaras, and Eleni Christodoulou. 2016. Analysis of Elderly Users' Preferences and Expectations on Service Robot's Personality, Appearance and Interaction. In *International Conference on Human Aspects of IT for the Aged Population*. Springer, 35–44.

[45] José María Augusto Landa, Esther López-Zafra, et al. 2010. The Impact of Emotional Intelligence on Nursing: An Overview. *Psychology* 1, 1 (2010), 50–58.

[46] John Lasseter. 1987. Principles of Traditional Animation Applied to 3D Computer Animation. In *ACM Siggraph Computer Graphics*, Vol. 21. ACM, 35–44.

[47] Virginia Lee Stamler, Martha D Christiansen, Kathleen H Staley, and Laura Macagno-Shang. 1991. Client preference for counselor gender. *Psychology of Women Quarterly* 15, 2 (1991), 317–321.

[48] Tze Wei Liew and Su-Mae Tan. 2016. Virtual Agents with Personality: Adaptation of Learner-Agent Personality in a Virtual Learning Environment. In *2016 Eleventh International Conference on Digital Information Management (ICDIM)*. IEEE, 157–162.

[49] Renaud Lunardo, G Bressoles, and François Durrieu. 2016. The Interacting Effect of Virtual Agents' Gender and Dressing Style on Attractiveness and Subsequent Consumer Online Behaviour. *Journal of Retailing and Consumer Services* 30 (2016), 59–66.

[50] Heidy Maldonado and Clifford Nass. 2007. Emotive characters can make learning more productive and enjoyable: it takes two to learn to tango. *Education Technology-Saddle Brook then Englewood Cliffs NJ* 47, 1 (2007), 33.

[51] Michael Mateas. 1999. An Oz-Centric Review of Interactive Drama and Believable Agents. In *Artificial intelligence today*. Springer, 297–328.

[52] John D Mayer. 1997. What is Emotional Intelligence. In *Emotional Development and Emotional Intelligence: Implications for Educators*, D Sluyter and Peter Salovey (Eds.). New York: Basic Books, 3–31.

[53] John D Mayer, David R Caruso, and Peter Salovey. 1999. Emotional Intelligence Meets Traditional Standards for an Intelligence. *Intelligence* 27, 4 (1999), 267–298.

[54] John D Mayer, Peter Salovey, and David R Caruso. 2000. Emotional intelligence as zeitgeist, as personality, and as a mental ability. (2000).

[55] John D Mayer, Peter Salovey, David R Caruso, and Gill Sitarenios. 2003. Measuring Emotional Intelligence with the MSCEIT v2.0. *Emotion* 3, 1 (2003), 97.

[56] Derek McColl and Goldie Nejat. 2012. Affect detection from body language during social HRI. In *RO-MAN, 2012 IEEE*. IEEE, 1013–1018.

[57] David McNeill. 1992. *Hand and mind: What gestures reveal about thought*. University of Chicago press.

[58] Margaret McRorie, Ian Sneddon, Etienne de Sevin, Elisabetta Bevacqua, and Catherine Pelachaud. 2009. A model of personality and emotional traits. In *International Workshop on Intelligent Virtual Agents*. Springer, 27–33.

[59] Albert Mehrabian and Norman Epstein. 1972. A Measure of Emotional Empathy. *Journal of Personality* 40, 4 (1972), 525–543.

[60] Mohammed Slim Ben Mimoun and Ingrid Poncin. 2015. A Valued Agent: How ECAs Affect Website Customers' Satisfaction and Behaviors. *Journal of Retailing and Consumer Services* 26 (2015), 70–82.

[61] Adam S Miner, Arnold Milstein, Stephen Schueller, Hegde Roshini, Christina Mangurian, and Eleni Linos. 2016. Smartphone-Based Conversational Agents and Responses to Questions about Mental Health, Interpersonal Violence, and Physical Health. *JAMA internal medicine* 176, 5 (2016), 619–625.

[62] Christos N Moridis and Anastasios A Economides. 2012. Affective Learning: Empathetic Agents with Emotional Facial and Tone of Voice Expressions. *IEEE Transactions on Affective Computing* 3, 3 (2012), 260–272.

[63] Cliff Nass and Scott Brave. 2007. Emotion in human-computer interaction. In *The human-computer interaction handbook*. CRC Press, 94–109.

[64] Clifford Nass and Youngme Moon. 2000. Machines and Mindlessness: Social Responses to Computers. *Journal of Social Issues* 56, 1 (2000), 81–103.

[65] Clifford Nass, Youngme Moon, BJ Fogg, Byron Reeves, and Chris Dryer. 1995. Can Computer Personalities be Human Personalities?. In *Conference Companion on Human Factors in Computing Systems*. ACM, 228–229.

[66] Clifford Nass, Jonathan Steuer, and Ellen R Tauber. 1994. Computers are Social Actors. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 72–78.

[67] Dotun Ogunyemi, Sukrant Mehta, Aaron Turner, David Kim, and Carolyn Alexander. 2013. Emotional intelligence characteristics in a cohort of faculty, residents, and medical students. *The Journal of reproductive medicine* 59, 5-6 (2013), 279–284.

[68] Ali Oker, Elise Prigent, Matthieu Courgeon, Victoria Eyharabide, Mathieu Urbach, Nadine Bazin, Michel-Ange Amorim, Christine Passerieux, Jean-Claude Martin, and Eric Brunet-Gouet. 2015. How and Why Affective and Reactive Virtual Agents will Bring New Insights on Social Cognitive Disorders in Schizophrenia? An Illustration with a Virtual Card Game Paradigm. *Frontiers in Human Neuroscience* 9 (2015).

[69] A Ortony, GL Clore, and A Collins. 1988. *The Cognitive Structure of Emotions*. Cambridge University Press.

[70] AK Pau and Ray Croucher. 2003. Emotional Intelligence and Perceived Stress in Dental Undergraduates. *Journal of Dental Education* 67, 9 (2003), 1023–1028.

[71] Catherine Pelachaud. 2009. Modelling multimodal expression of emotion in a virtual agent. *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 364, 1535 (2009), 3539–3548.

[72] KV Petrides and Adrian Furnham. 2000. On the Dimensional Structure of Emotional Intelligence. *Personality and Individual Differences* 29, 2 (2000), 313–320.

[73] Rosalind Wright Picard. 1995. Affective Computing. (1995).

[74] Rosalind Wright Picard. 1997. *Affective Computing*. Vol. 252. MIT press Cambridge.

[75] Jitna Por, Louise Barriball, Joanne Fitzpatrick, and Julia Roberts. 2011. Emotional Intelligence: Its Relationship to Stress, Coping, Well-Being and Professional Performance in Nursing Students. *Nurse Education Today* 31, 8 (2011), 855–860.

[76] Soujanya Poria, Erik Cambria, Rajiv Bajpai, and Amir Hussain. 2017. A Review of Affective Computing: From Unimodal Analysis to Multimodal Fusion. *Information Fusion* 37 (2017), 98–125.

[77] J Donald Ragsdale. 1976. Relationships between hesitation phenomena, anxiety, and self-control in a normal communication situation. *Language and Speech* 19, 3 (1976), 257–265.

[78] Hedieh Ranjbartabar and Deborah Richards. 2016. A Virtual Emotional Freedom Therapy Practitioner. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 1471–1473.

[79] Peter AM Ruijten, Cees JH Midden, and Jaap Ham. 2016. Ambiguous Agents: The Influence of Consistency of an Artificial Agent's Social Cues on Emotion Recognition, Recall, and Persuasiveness. *International Journal of Human-Computer Interaction* 32, 9 (2016), 734–744.

[80] Timothy Rumbell, John Barnden, Susan Denham, and Thomas Wennekers. 2012. Emotions in autonomous agents: comparative analysis of mechanisms and functions. *Autonomous Agents and Multi-Agent Systems* 25, 1 (2012), 1–45.

[81] Stuart Russell and Peter Norvig. 1995. A modern approach. *Artificial Intelligence. Prentice-Hall, Englewood Cliffs* 25 (1995), 27.

[82] Nuray Sakallı-Uğurlu, Selin Salman, and Sinem Turgut. 2010. Predictors of Turkish Women's and Men's Attitudes toward Sexual Harassment: Ambivalent Sexism, and Ambivalence toward Men. *Sex Roles* 63, 11-12 (2010), 871–881.

[83] Peter Salovey and Daisy Grewal. 2005. The Science of Emotional Intelligence. *Current Directions in Psychological Science* 14, 6 (2005), 281–285.

[84] Peter Salovey and John D Mayer. 1990. Emotional intelligence. *Imagination, Cognition and Personality* 9, 3 (1990), 185–211.

[85] Carla Sharp, Salome Vanwoerden, Y Van Baardewijk, JL Tackett, and H Stegge. 2015. Callous-unemotional traits are associated with deficits in recognizing complex emotions in preadolescent children. *Journal of personality disorders* 29, 3 (2015), 347–359.

[86] Joanna Stern. 2017. Alexa, siri, cortana: The problem with all-female digital assistants. https://www.wsj.com/articles/alexa-siri-cortana-the-problem-with-all-female-digital-assistants-1487709068 Accessed on February 2017.

[87] Leila Takayama, Doug Dooley, and Wendy Ju. 2011. Expressing thought: improving robot readability with animation principles. In *Proceedings of the sSixth International Conference on Human-Robot Interaction*. ACM, 69–76.

[88] Maya Tamir. 2016. Why do people regulate their emotions? A taxonomy of motives in emotion regulation. *Personality and Social Psychology Review* 20, 3 (2016), 199–222.

[89] Jianhua Tao and Tieniu Tan. 2005. Affective Computing: A Review. In *International Conference on Affective Computing and intelligent interaction*. Springer, 981–995.

[90] Christiana Tsiourti, Maher Ben Moussa, João Quintas, Ben Loke, Inge Jochem, Joana Albuquerque Lopes, and Dimitri Konstantas. 2016. A Virtual Assistive Companion for Older Adults: Design Implications for a Real-World Application. In *Proc. SAI Intell. Syst. Conf.* 556–566.

[91] Gerben A Van Kleef, Helma van den Berg, and Marc W Heerdink. 2015. The Persuasive Power of Emotions: Effects of Emotional Expressions on Attitude Formation and Change. *Journal of Applied Psychology* 100, 4 (2015), 1124.

[92] Thurid Vogt, Elisabeth André, and Nikolaus Bee. 2008. EmoVoice: A Framework for Online Recognition of Emotions from Voice. In *International Tutorial and Research Workshop on Perception and Interactive Technologies for Speech-Based Systems*. Springer, 188–199.

[93] Elaine F Walker and Jayne E Stake. 1978. Changes in preferences for male and female counselors. *Journal of Consulting and Clinical Psychology* 46, 5 (1978), 1153.

[94] Jing Wang, Ronald J Iannotti, and Tonja R Nansel. 2009. School bullying among adolescents in the United States: Physical, verbal, relational, and cyber. *Journal of Adolescent Health* 45, 4 (2009), 368–375.

[95] Jerry S Wiggins. 1979. A psychological taxonomy of trait-descriptive terms: The interpersonal domain. *Journal of Personality and Social Psychology* 37, 3 (1979), 395.

[96] Melissa J Williams and Larissa Z Tiedens. 2016. The subtle suspension of backlash: A meta-analysis of penalties for women's implicit and explicit dominance behavior.

[97] Lauren Wispé. 1987. History of the Concept of Empathy. *Empathy and Its Development* (1987), 17–37.

[98] Steven B Wolff, Anthony T Pescosolido, and Vanessa Urch Druskat. 2002. Emotional Intelligence as the Basis of Leadership Emergence in Self-Managing Teams. *The Leadership Quarterly* 13, 5 (2002), 505–522.

[99] Michael Wooldridge and Nicholas R Jennings. 1995. Intelligent Agents: Theory and practice. *The Knowledge Engineering Review* 10, 02 (1995), 115–152.

[100] Yang Yang, Xiaojuan Ma, and Pascale Fung. 2017. Perceived Emotional Intelligence in Virtual Agents. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM.

Moving Deep Learning into Web Browser: How Far Can We Go?

Yun Ma
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
Tsinghua University, Beijing, China
mayun@pku.edu.cn

Dongwei Xiang
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
xdw@pku.edu.cn

Shuyu Zheng
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
zhengshuyu@pku.edu.cn

Deyu Tian
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
tiandeyu@pku.edu.cn

Xuanzhe Liu*
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing, China
xzl@pku.edu.cn

ABSTRACT

Recently, several JavaScript-based deep learning frameworks have emerged, making it possible to perform deep learning tasks directly in browsers. However, little is known on what and how well we can do with these frameworks for deep learning in browsers. To bridge the knowledge gap, in this paper, we conduct the first empirical study of deep learning in browsers. We survey 7 most popular JavaScript-based deep learning frameworks, investigating to what extent deep learning tasks have been supported in browsers so far. Then we measure the performance of different frameworks when running different deep learning tasks. Finally, we dig out the performance gap between deep learning in browsers and on native platforms by comparing the performance of TensorFlow.js and TensorFlow in Python. Our findings could help application developers, deep-learning framework vendors and browser vendors to improve the efficiency of deep learning in browsers.

CCS CONCEPTS

• **General and reference** → **Measurement**; • **Information systems** → **Browsers**; • **Software and its engineering** → **Software performance**.

KEYWORDS

Deep learning; Web browser; Web applications; Measurement

ACM Reference Format:
Yun Ma, Dongwei Xiang, Shuyu Zheng, Deyu Tian, and Xuanzhe Liu. 2019. Moving Deep Learning into Web Browser: How Far Can We Go?. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313639

*Corresponding author: xzl@pku.edu.cn.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313639

1 INTRODUCTION

In the past decade, the advance of deep learning (DL) technique has significantly promoted the artificial intelligence (AI). Numerous AI applications, e.g., image processing, object tracking, speech recognition, and natural language processing, have raised urgent requirements to adopt the DL. As a result, various libraries and frameworks, such as TensorFlow [18], Caffe [2], and CNTK [3], have been proposed and applied in practice.

However, developing AI applications powered by the popular DL frameworks and libraries is a non-trivial task. Usually, these frameworks and libraries are leveraged by *native* applications that can run on heterogeneous development environments such as Windows, Linux, MacOS/iOS, and Android. The applications are developed by various imperative programming languages, i.e., C/C++ on Windows, Objective-C on iOS and MacOS, and Java on Android. Developing AI applications that is portable to multiple platforms is indeed not easy. The development is particularly complicated for mobile applications, as the app vendors usually need to develop and maintain both iOS and Android versions. In addition, the deployment is also non-trivial, as most current platforms come with an appstore, some of which require manual testing of submitted applications by the appstore provider before being published-a process that can take several weeks-and applications can be rejected for seemingly arbitrary reasons.

Compared to the native applications, Web applications can indeed make the cross-platform portability issues much simpler. The same implementation of a DL-powered Web application can be deployed in the browser on all platforms regardless of the underlying hardware device types (PC, smartphones, and wearable devices) and the operating systems (Windows, Mac, iOS, and Android). Advancements in HTML5, CSS3, and especially JavaScript language, started to enable the creation of DL-powered Web applications that offer a comparable experience to native applications, especially for the popular Web game applications [38][34]. In particular, benefited from the development of WebGL [32][19][20], current major browsers such as Google Chrome, Mozilla FireFox, and Apple Safari, can better utilize the integrated graphics card to accelerate DL tasks, without the need of standalone graphics card like NVIDIA which is required by native DL frameworks.

Running DL-powered Web applications in browsers has drawn the attention from various research communities including AI, software engineering, Web browsers, and even computer architecture. As a result, various JavaScript-based DL development frameworks and libraries have been published. In 2015, Karpathy presented the ConvNetJS [4], known as the first JavaScript library for DL in Web browsers to date. Other efforts such as WebDNN [15], Keras.js [5], and Mind [7], were proposed to support DL in browsers. In early 2018, Google released the TensorFlow.js [14], which is a significant step for promoting the DL in browsers.

Although the preceding efforts along with some on-going efforts seem to make running DL tasks in browsers possible, little is known on what DL tasks we can do and how well DL works in browsers. More importantly, considering the long debate of performance of Web applications compared with that of native applications, the same issue also exists in developing DL-powered Web applications. Hence, it is urgent to address such a knowledge gap in terms of the feasibility and usability for running DL in Web browsers.

In this paper, we make the first empirical study of DL in browsers by answering the following research questions.

- **RQ1**: What features do existing frameworks provide to implement various kinds of DL tasks in the browser?
- **RQ2**: How well do existing frameworks perform over different DL tasks?
- **RQ3**: How big is the performance gap between running DL in the browser and on the native platform?

We select 7 popular JavaScript-based frameworks that support running DL in browsers, and conduct a characteristic study over them. We develop a browser extension to measure the performance as well as the utilization of system resources when running different DL tasks. We choose the TensorFlow.js and native TensorFlow in Python to compare the performance of DL in browsers with that on native platforms.

The key findings of our study includes:

• **DL in browsers is still at dawn.** Most frameworks of DL in browsers support only a specific subset of DL tasks. Among all the frameworks, TensorFlow.js provides the most number of functionalities to realize various kinds of DL tasks.

• **Support of training in browsers is not fledged.** In most frameworks, inference has drawn more attention compared with training. For training tasks, the number of neurons per layer dominates the performance variation considering the complexity of DL models since the browser is limited in complex matrix calculation.

• **Model loading dominates the computation for inference tasks.** Loading and warming up the DL model costs more time than running the inference task itself. The CPU backend performs better than the GPU backend when the browser run inference tasks for small-size models.

• **Integrated graphics card helps browsers to beat native platforms when standalone graphics card is not available.** For popular pre-trained models like MobileNet and Inception, TensorFlow.js is just 1x to 2x slower than native TensorFlow in Python when running inference tasks on the standalone graphics card. TensorFlow.js on the integrated graphics card outperforms the native TensorFlow on CPU when running the same inference task.

• **System resources can be further exploited for in-browser DL tasks.** For TensorFlow.js, the CPU is not fully utilized (about 80%) when DL tasks run on the CPU backend. The memory allocated to WebGL is limited by the browser, leading to the crash of some DL tasks.

Based on the findings, We have drawn some practical recommendations for application developers as well as DL-framework and browser vendors. Application developers who aim to develop DL-powered Web applications, need to better control the the number of neurons per layer in DL models, pre-load the model file in advance, and employ the CPU backend rather than the GPU backend when running inference tasks on small DL models. DL-framework vendors should consider encoding the model file in binary format rather than JSON to reduce the file size as well as improve the computation time, and leverage compiler optimization techniques to reduce the call stack. Browser vendors should consider supporting multi-process and scheduling over multi-core in the JavaScript engines.

The remainder of this paper is organized as follows. Section 2 shows some background knowledge of deep learning in browsers. Sections 3 to 5 describe the results, including the analysis of framework functionality, performance measurement, and comparison with native DL frameworks. Section 6 presents the implications and recommendations drawn from the findings. Section 7 surveys related work and Section 8 concludes the paper with future work.

2 BACKGROUND

In this section, we give some background of deep learning and then discuss how browsers support deep learning tasks.

2.1 Deep Learning

Deep learning (DL) is a class of machine learning algorithms that use a cascade of multiple layers of nonlinear processing units (called neurons) for feature extraction and transformation. Each successive layer uses the output from the preceding layer as input. In recent years, DL has gained great success in many areas such as computer vision, speech recognition and natural language processing.

There are many types of DL models, among which deep neural network (DNN) [23], convolutional neural network (CNN) [28], and recurrent neural network (RNN) [35] are three basic structures. DNN is a typically feedforward network with multiple layers between the input and output layers, in which data flows from the input layer to the output layer without looping back. CNN uses a variation of multi-layer perceptrons designed to require minimal preprocessing, usually applied to analyzing visual imagery. RNN has connections between nodes forming a directed graph along a temporal sequence, allowing it to exhibit temporal behaviors.

DL consists of two phases: training phase where the input data are used to calculate the parameters of the model, and inference phase where the trained model outputs the value given a specific input sample.

2.2 Deep Learning in Browsers

Recently, there is a trend that applications perform DL tasks directly on the clients for better privacy and timely response. As a cross-platform client-side computation target, Web browsers have drawn

the attention to AI communities to support client-side DL. Several applications of in-browser DL are implemented and published, such as 1) TensorFlow playground [13], which is an interactive platform to learn the principle of DL; 2) Teachable Machine [12], which gives the users an experience of teaching the machine how to response when they pose a gesture, using camera in the browser; 3) MLitB [33], which is capable of performing distributed learning with heterogeneous classes of devices using Web browsers; 4) MorphCast [9], which combines interactive video and face recognition with emotion, gender and age analysis to create adaptive-media.

DL in browsers is implemented by JavaScript and rely on the browser engine to execute. Fortunately, the advancement of latest browsers provides APIs to access GPU, which can be used to accelerate matrix calculations of DL. These APIs are: 1) **WebGL** [16], which is a JavaScript API for rendering interactive 2D and 3D graphics within any compatible Web browser; 2) **WebGPU** [17], which is the fastest among existing JavaScript APIs for accelerating graphics and computation. Currently, WebGPU API is supported only in Safari Technology Preview. We should mention that WebGL and WebGPU can run on both integrated graphics cards and standalone graphics cards.

3 SUPPORTED FEATURES OF DEEP LEARNING IN BROWSERS

In this section, we make a characteristic study to answer the first research question, i.e., what features do existing frameworks provide to implement various kinds of DL tasks in the browser? We first introduce the frameworks selected for the study. Then we compare the features of these frameworks from two aspects: provided functionality and developer support. For provided functionality, we mainly examine whether each framework supports some basic functionalities that are commonly used in the development of DL applications. For developer support, we take a look at some factors which may affect the efficiency of developing and deploying DL applications. Table 1 summarizes all the results as of Nov. 2018.

3.1 Selected Frameworks

To select the state-of-the-art frameworks of supporting DL in browsers, we search on the GitHub with the keyword "deep learning framework" and filter the results in JavaScript language. Then we choose the top 7 frameworks of which the number of stars exceeds 1,000 on GitHub. We introduce each framework as follows.

TensorFlow.js [14], released by Google in Mar. 2018, is an in-browser machine learning library that supports defining, training, and running models entirely in the browser using JavaScript. It is the successor to deeplearn.js which is now called TensorFlow.js Core. TensorFlow.js is powered by WebGL and provides high-level APIs for defining models. TensorFlow.js support all the Keras layers (including Dense, CNN, LSTM, and so on). Therefore, it is easy to import models pre-trained by the native TensorFlow and Keras into the browser and run with Tensorflow.js.

ConvNetJS [4] is a Javascript library originally written by Andrej Karpathy at Stanford. The entire library is based on transforming 3-dimensional volumes of numbers. ConvNetJS currently supports common neural network models and cost functions for classification and regression. Furthermore, it supports convolutional networks, and an experimental reinforcement learning. Unfortunately, although ConvNetJS might be the most famous framework before TensorFlow.js, it is no longer maintained after Nov. 2016.

Keras.js [5] abstracts away a number of frameworks as backends including TensorFlow, CNTK, etc. It supports importing models pre-trained by Keras for inference. In the GPU mode, computation is performed by WebGL. However, this project is no longer active.

WebDNN [15], released by the University of Tokyo, claims to be the fastest DNN execution framework in browsers. It supports only the inference tasks. The framework supports 4 execution backends: WebGPU, WebGL, WebAssembly, and fallback pure JavaScript implementation. WebDNN optimizes DNN models by compressing the model data to accelerate the execution. Empirical evaluations showed that it achieved more than 200x acceleration [6].

brain.js [1] is a JavaScript library for neural networks replacing the deprecated "brain" library. It provides DNN, RNN, LSTM and GRU for training tasks. The library supports serializing and loading the state of a trained DL model with JSON.

synaptic [11] is a JavaScript architecture-free neural network library, supporting basically any type of first order or even second order RNN. This library also includes a few built-in DL architectures, including multi-layer perceptrons, LSTM, liquid state machines and Hopfield networks.

Mind [7] is a flexible neural network library. The core framework has only 247 lines of code, which uses a matrix implementation to process training data. It supports customization of the network topology and plugins to configure pre-trained models created by the mind community. However, this framework is no longer active.

3.2 Provided Functionality

Support for training. Most frameworks support training and inference tasks in the browser. However, Keras.js and WebDNN do not support training DL models in browsers. They support only loading pre-trained models to perform inference tasks. Therefore, the number is not available for the types of layer/activation/optimizer supported by Keras.js and WebDNN in Table 1.

Supported network types. Some frameworks are not for general-purpose DL tasks, so they differ in the supported network types. Specifically, TensorFlow.js, Keras.js and WebDNN support three network types: DNN, CNN and RNN. However, ConvNetJS mainly supports CNN tasks and does not support RNN. brain.js and synaptic mainly support RNN tasks, and do not support convolution and pooling operations used in CNN networks. Mind supports only the basic DNN.

Supported layer types. All frameworks support building neural networks using units of layers. The layer API of TensorFlow.js supports 49 different layers, including dense, convolution, pooling, RNN, normalization, and so on. Other frameworks support a smaller variety of layers, which are also related to the network types they support. It should be noted that the core API of TensorFlow.js is implemented in a way similar to the native TensorFlow which combines various operations to build computational graphs. synaptic is an architecture-free framework that supports building any type of first order or even second order RNN networks.

Table 1: Characteristics of JavaScript-based frameworks that support deep learning in browsers.

		TensorFlow.js	ConvNetJS	Keras.js	WebDNN	brain.js	synaptic	Mind
Basic Information								
Github Stars		9453	9364	4348	1464	6366	6315	1333
Main Contributor		Google	Stanford University	Leon Chen	The University of Tokyo	Robert Plummer	Juan Cazala	Steven Miller
Last Commit Date		Oct 30, 2018	Nov 25, 2016	Aug 17, 2018	Oct 25, 2018	Nov 5, 2018	Mar 25, 2018	Jul 7, 2017
Status		Active	Not Active	Not Active	Active	Active	Active	Not Active
Functionality								
Support for Training		Y	Y	N	N	Y	Y	Y
Supported Network Types	**DNN**	Y	Y	Y	Y	Y	Y	Y
	CNN	Y	Y	Y	Y	N	N	N
	RNN	Y	N	Y	Y	Y	Y	N
Supported Layer Types		49	7	NA	NA	7	1	1
Supported Activation Types		16	4	NA	NA	4	5	2
Supported Optimizer Types		7	3	NA	NA	1	NA	NA
Support for GPU Accelaration (WebGL)		Y	N	Y	Y	N	N	N
Developer Support								
Documents		Y	Y	Not finished	Y	Only tutorials	Y	Y
Demos		20	10	9	8	7	7	4
Importing Models from Other Frameworks	**TensorFlow**	Y	N	N	Y	N	N	N
	Keras	Y	N	Y	Y	N	N	N
	Caffe&Pytorch	N	N	N	Y	N	N	N
API to Save/Load Model	**Save**	Y	Y	N	N	Y	Y	Y
	Load	Y	Y	Y	Y	Y	Y	Y
Support for Server Side (Node.js)		Y	Y	Y	Y	Y	Y	Y
Library Size		732KB	33KB	650KB	130KB	819KB	106KB	NA

Supported activation/optimizer types. In general, TensorFlow.js provides developers with the most kinds of choices. For activation functions, other frameworks support only basic sigmoid or ReLU. For optimizers, other frameworks mainly support basic stochastic gradient descent (SGD).

Support for GPU acceleration (WebGL). TensorFlow.js is the only framework that supports GPU-accelerated training tasks. TensorFlow.js, Keras.js, and WebDNN support using GPU to accelerate inference tasks. WebDNN also supports a more advanced technology, WebGPU, but WebGPU has been supported by only the technology preview version of Safari.

3.3 Developer Support

Documents. Documents provided by TensorFlow.js, ConvNetJS, WebDNN and synaptic are completed and in detail. The document of Keras.js is not complete and brain.js has only a few tutorials.

Demos. All the frameworks provide demos for developers to get start. TensorFlow.js offers the richest demos covering a wide range of use cases.

Importing models from other frameworks. TensorFlow.js, Keras.js and WebDNN support importing models from native DL frameworks in Python and all of them provide Python scripts for converting models. TensorFlow.js supports models trained by TensorFlow and Keras. Keras.js supports Keras models. WebDNN supports importing models from TensorFlow, Keras, Caffe and Pytorch. With the support of using pre-trained models from other DL frameworks, the development effort can be significantly reduced.

API to save/load model. All frameworks that support training tasks in the browser have APIs for saving models. All frameworks have APIs for loading models.

Support for server side (Node.js). All frameworks are supported for Node.js. Such a feature makes it possible to offload computation inside browsers onto remote servers.

Library size. We list the size of the library files that need to be loaded into browsers. ConvNetJS is the smallest, which is just 33KB. TensorFlow.js and brain.js have very large size of files, which are 732KB and 819KB, respectively. Small-size libraries are better for loading applications in browsers since all the files have to be downloaded on demand.

4 PERFORMANCE OF DEEP LEARNING IN BROWSERS

In this section, we conduct a measurement study to investigate the second research question, i.e., how well do existing frameworks perform over different DL tasks? We study the influence of model complexity and backend processor (CPU or GPU) on the performance when the browser runs training and inference tasks.

4.1 Experiment Setup

DL model. As explained before, the network types supported by different frameworks are not the same. So we adopt the most basic fully connected neural network as the model in the experiment. For the dataset to run the DL tasks, we use the classic MNIST handwritten digit recognition database [8]. The model to be trained has 784 input nodes and 10 output nodes. To study the influences of model complexity on the performance, we choose different configurations

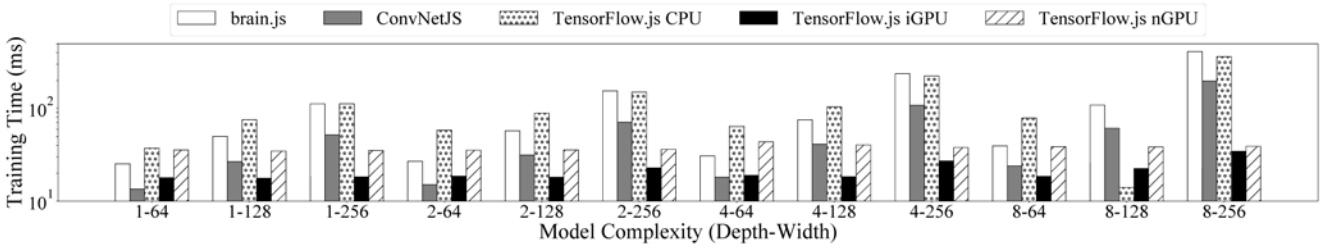


Figure 1: Average training time (ms) on one batch under different model complexities. The y-axis is on log scale.

of the model. The parameters include 1) the number of the hidden layers (depth) of the neural network, which ranges in [1, 2, 4, 8], and 2) the number of neurons (width) in each hidden layer, which ranges in [64, 128, 256]. The range of depth and width is set based on the assumption that client-side DL models should be of small size, being able to run on the client. In the training process, the batch size is always set to 64.

Hardware. In order to study the performance difference between CPU and GPU backend, we use a Hasee T97E laptop computer, which has a standalone graphics card, Nvidia 1070 Max-Q (with 8GB GPU memory). The CPU is Intel i7-8750H, which includes an Intel HD Graphics 630, enabling us to measure the performance using integrated graphics card. In the following, we use nGPU and iGPU to denote the GPU backend on the standalone Nvidia graphics card and the integrated Intel graphics card, respectively.

Software. All the experiments run on the Chrome browser (version: 71.0.3578.10 dev 64-bit) on Ubuntu 18.04.01 LTS (64-bit). For the frameworks, we use their latest published version.

Performance measurement. For each DL task, we implement a Web page where the configurations of DL models can be varied through the parameters in the URL. We run each DL task on the Chrome browser, and measure the time spent on finishing the task. Since each experiment usually requires running dozens of tasks under different configurations, we developed a Chrome extension to iterate all the pages and change the configuration after one task is performed. This browser extension is also responsible for monitoring the system resource usage of the Web page. At the same time, a local server records the experimental statistics uploaded by the extension.

4.2 Training Performance

We select four JavaScript frameworks, brain.js, ConvNetJS, synaptic, and TensorFlow.js, which support training in browsers, to compare their performance of running training tasks. All the four frameworks can train models on the CPU backend except that TensorFlow.js is also able to use the GPU backend via WebGL. We train the defined model using each framework and obtain the average time spent on training one batch. Figure 1 shows the results under different model complexities. Since the training time of synaptic is about tens to hundreds of times longer than that of other frameworks, we omit the result of synaptic in the figure for better presentation but the findings are similar to other frameworks.

In general, the training time increases with the increase of the network size since more computation is needed to complete the training process for larger networks. Comparing the training time of different frameworks on the CPU backend, we can see that ConvNetJS is the fastest among all the frameworks for all network configurations. The possible reason may be that ConvNetJS is designed to be simpler, which can be reflected by its small library file size. Brain.js is closely behind, with a performance gap of about two times (2x) with ConvNetJS. Tensorflow.js has a performance gap of two to three times (2x-3x) with ConvNetJS. When comparing the training time ratio of ConvNetJS over TensorFlow.js, we find that the performance gap is gradually reduced when the depth and width increase, indicating that compared with ConvNetJS, TensorFlow.js has relatively large overhead beyond calculation. In addition, the performance gap is larger as the network width increases than as the network depth increases, implying that TensorFlow.js deals better with large-scale matrix calculation than ConvNetJS.

Table 2: CPU utilization (%) in the training process.

Framework	Backend	Max	Min	Average
brain.js	CPU	104.0	99.9	101.2
ConvNetJS	CPU	108.0	101.9	104.1
synaptic	CPU	113.9	88.7	102.8
TensorFlow.js	CPU	108.0	61.0	82.1
	iGPU	82.0	54.9	65.9
	nGPU	75.9	48.0	60.0

GPU benefits. The training time on the CPU backend becomes longer with the increase of network size, but the results on the GPU backend are not the same. For both the iGPU with weaker computation power and the nGPU which can satisfy larger-scale matrix calculations, the training time does not increase significantly. But in the process from (4 hidden layers, 128 neurons per layer) to (8 hidden layers, 256 neurons per layer), the training time of iGPU increases significantly. The reason may be that under the network size set in this experiment, the training process does not reach the GPU's capability bottleneck. Although the matrix computation capability of nGPU is better than that of iGPU, the training time on nGPU is even longer than iGPU. Such a result is caused by the excessive time overhead to call the WebGL for accessing GPU. The real computation time of GPU should be much shorter.

System resource utilization. We show the statistics of CPU utilization of each framework during the training process in Table 2. 110% is the upper bound of CPU utilization. The capability of multi-core processor cannot be used since the JavaScript engine is single-threaded. As a result, it can only maximize the usage of a single core. The reason why the CPU utilization is over 100% is that other

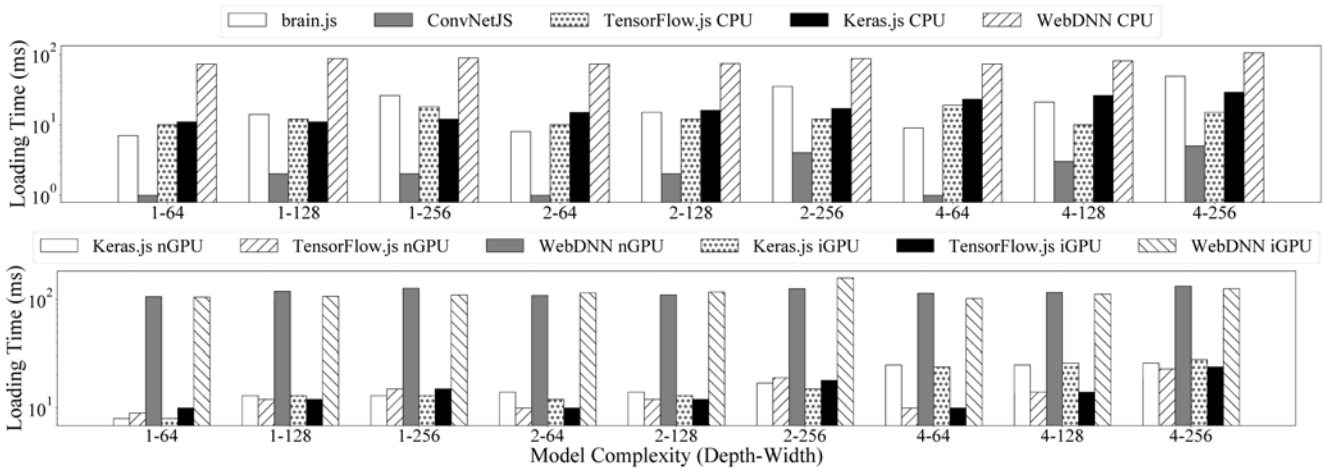


Figure 2: Model loading time (ms) under different model complexities. The y-axis is on log scale.

kernel and user space components occasionally run simultaneously in other threads.

On the CPU backend, TensorFlow.js sometimes cannot maximize the utilization of a single core and its average CPU utilization is only 82.1%. Meanwhile, we can find that when running training tasks on the GPU backend, CPU is not fully utilized since most computation is on the GPU. Training on iGPU has about 5-7% higher CPU utilization than that on nGPU.

4.3 Inference Performance

We select 6 JavaScript frameworks to compare their performance of running inference tasks. TensorFlow.js, Keras.js, and WebDNN support using GPU for acceleration, but brain.js, ConvNetJS, and synaptic support using only CPU for inference. In terms of model usage, brain.js, ConvNetJS, synaptic and TensorFlow.js support saving their own trained models, while Keras.js and WebDNN only support importing pre-trained models from other deep learning frameworks. Therefore, for brain.js, ConvNetJS, synaptic and TensorFlow.js, we use the models saved by the frameworks themselves. For Keras.js and WebDNN, we use the models trained by Keras and then convert the models to the corresponding format. Theoretically, the parameter values of the trained DL models should be different, but the absolute value does not affect the inference time. So we just assign the same parameter values to all the models of different frameworks.

The inference task involves loading a pre-trained model and then given a sample input, the model outputs the result. In addition, on the GPU backend, there is a warmup process where the first sample for inference is usually used to activate the GPU processor. Therefore, we break down the inference process into three phases: model loading, warming up, and inference, and study the fine-grained performance. Due to the space limitation, we omit the results where the model depth is 8 in the following analysis because the trend is similar as the depth increases. Besides, since the model loading time and inference time of synaptic are still much longer than those of other frameworks, we do not depict the results of synaptic in the figures for better presentation.

Table 3: Size of model files (MB).

Depth	Width	brain.js	ConvNetJS	synaptic	TensorFlow.js
1	64	1.4	1.3	3.4	0.2
	128	2.7	2.7	6.7	0.4
	256	5.5	5.4	13.3	0.8
2	64	1.5	1.5	3.7	0.2
	128	3.2	3.1	7.8	0.5
	256	7.2	7.1	17.7	1.1
4	64	1.7	1.7	4.2	0.3
	128	4.0	4.0	10.1	0.6
	256	10.7	10.5	26.5	1.6

Model file size. We first investigate the size of the model file used by different frameworks. As models for inference usually should be downloaded from the remote server, smaller size of model files means shorter downloading time. Table 3 shows the size of model files that are used in all inference experiments. ConvNetJS and brain.js use similar JSON encoding, so the size of their model files are nearly the same. The model file of synaptic uses JSON encoding as well but its size is the largest among all the frameworks. As the model files used by TensorFlow.js, Keras.js and WebDNN are all converted from Keras models, their model files are of the same size. So we just show TensorFlow.js in the table. Since the model converted from Keras is compressed and saved as a binary file, the size can be greatly reduced, just about 1/7 of the model file in JSON.
Model loading time. We then compare the time spent on loading the model of different frameworks, as shown in Figure 2. For the CPU backend, the loading time of different models of the same framework is proportional to the size of the model files described in Table 3. However, the model loading time of different frameworks is significantly different. ConvNetJS is the fastest. Model loading time of brain.js, TensorFlow.js and Keras.js are consistent in terms of magnitude. Interestingly, the increase of loading time of ConvNetJS, brain.js and synaptic is particularly noticeable when the width increases. The result is caused by their choice of using JSON to encode models. The model loading time of synaptic is slowest among all the frameworks, which are more than 100x to 1000x

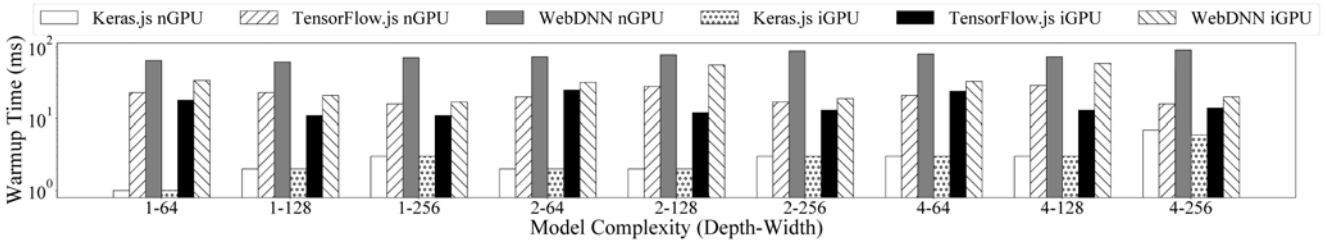


Figure 3: Model warmup time (ms) on GPU under different model complexities. The y-axis is on log scale.

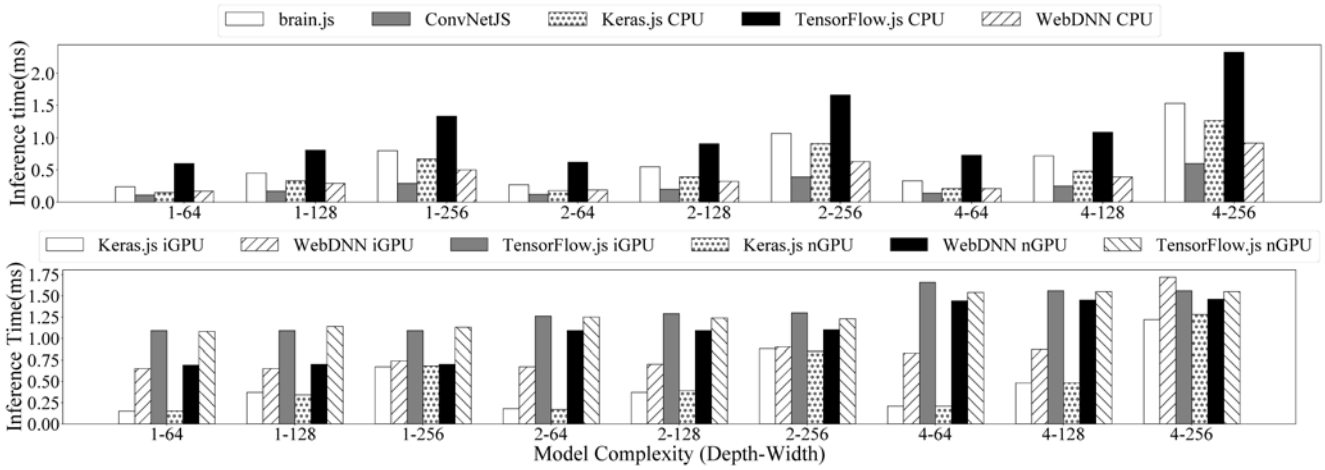


Figure 4: Average inference time (ms) on one sample under different model complexities.

longer than ConvNetJS. The model loading time of TensorFlow.js is almost unchanged regardless of the model size.

The loading time on the GPU backend does not change much under different model complexities. However, the difference is still significant between different frameworks. TensorFlow.js is the fastest. Compared with loading models on the CPU backend, Keras.js speeds up loading large models, but the loading time of WebDNN is longer. In addition, it can be seen that there is no difference in the model loading time between iGPU and nGPU.

Warmup time. Next, we examine the difference of warmup time on the GPU backend. As shown in Figure 3, Keras.js is still far ahead, and can complete the warmup in 3ms on all tasks. Tensorflow.js is the second, and WebDNN is the worst. On the whole, the warmup time on iGPU backend is shorter than that on nGPU.

Inference time. Figure 4 shows the average time of doing inference on one sample. Almost all the inference tasks can finish within 1.5ms (except synaptic, of which the shortest is 6.68ms). In the range of the model sizes we set, the powerful computation capability of GPU does not make a difference. Among all the model sizes, ConvNetJS occupies all the first place, followed by WebDNN on the CPU backend. The inference time of WebDNN on the GPU backend is longer than the inference time on the CPU backend. As for TensorFlow.js, running on the CPU backend is faster for inference on smaller models, while the GPU backend is faster for inference on larger models. Inference times of Keras.js on the CPU and GPU backend are basically the same.

We can observe that for all the frameworks on the CPU backend, the inference time increases when the model becomes complex. In particular, when the width increases, the time increases sharply (about two times as the model width doubles). Similar to the training tasks, such a result also reflects that these frameworks do not optimize the large-scale matrix operations in the process of forward propagation on the CPU backend. TensorFlow.js and WebDNN on the GPU backend do not exhibit this problem, but Keras.js on the GPU still suffers from this problem.

4.4 Takeaway

Based on the above results, we can see that in small-scale fully-connected neural network which the browser is capable of, ConvNetJS performs the best for both training and inference. However, since ConvNetJS is no longer maintained and has fewer functionalities, developers may need to choose some alternatives.

Tensorflow.js is the only framework that can take advantage of GPU to accelerate training processes. It is feature-rich and has comparable performance with ConvNetJS. So TensorFlow.js is a good choice for both training and inference. We do not recommend using GPU as the backend on small models because the advantage of GPU's computation power is not fully exploited.

Finally, we are interested in why ConvNetJS has the best performance for all the tasks among these frameworks. Given the same model of which the process logic is the same, the performance difference is likely to be accounted by the different implementation

details. To this end, we compare the function call stack of ConvNetJS with that of TensorFlow.js when doing the same training task. It is surprising to find that the depth of the call stack of ConvNetJS is only 3 while TensorFlow.js is 48! Such a result suggests that one possible reason for the performance difference among different frameworks is the deep call stack that costs a lot of computation resources.

5 COMPARISON WITH NATIVE FRAMEWORK

In this section, we study the third research question, i.e., how big is the performance gap between running DL in the browser and on the native platform? To this end, we compare the performance of TensorFlow.js and the native TensorFlow in Python, both of which are released and maintained by Google and have similar APIs, making the comparison fair enough.

We study the performance gap from two aspects. On one hand, we leverage well-known pre-trained models to compare the performance when running inference tasks on TensorFlow.js and native TensorFlow. On the other hand, we use decision tree analysis to distinguish the factors contributing to the performance gap. We use the same laptop as the one used in the experiments of the last section. We install the latest TensorFlow in Python (version 1.11.0) on the laptop.

5.1 Inference Based on Pre-Trained Models

We use the pre-trained models officially provided by the Keras to measure the performance of TensorFlow.js and native TensorFlow when doing inference tasks on these classical models.

5.1.1 Limitations of TensorFlow.js and browser constraints. Keras officially provides 11 pre-trained models. Although these models can work using native TensorFlow, we encountered a series of errors when we run them using TensorFlow.js in the browser. These errors imply the limitations of TensorFlow.js itself as well as constraints imposed by the browser.

For the model of NasNet Large, the browser throws out the error message "truncatedNormal is not a valid Distribution". For the model of ResNet V3, the browser throws out the error message "Unknown layer: Lambda". The reason for these two errors is that TensorFlow.js is still under development and so far has offered only a limited number of support for the converted model. Many user-defined operations are not supported by TensorFlow.js, e.g., models with control flow operations in RNNs are not yet supported.

When we try to use VGG16 or VGG19, the browser throws out the error message "GL OUT OF MEMORY", meaning that the GPU memory is overfilled. The reason is that the VGG16 model applies for more than 1GB GPU memory. However, it should not be an issue since the GPU memory of our experiment laptop is 8GB. As a result, such an error is due to the browser constraints.

After trying all the models, we finally have 5 models that can be correctly converted and run on the browser. The information of these models are listed in Table 4. The number of trainable parameters is obtained by the build-in `summary()` method of tensorflow.keras, and the computation complexity (Floating Operations) are obtained by `tensorflow.propfiler.profile()` method.

Table 4: Selected Keras pre-trained models.

Model Name	Pre-trained Model Size	Trainable Parameters	Computation (FLOPs)
MobileNetV2	14MB	3.5M	7.2M
DenseNet121	33MB	8.0M	16.3M
Xception	88MB	22.9M	46.0M
InceptionV3	92MB	23.8M	47.8M
ResNet50	99MB	25.6M	51.4M

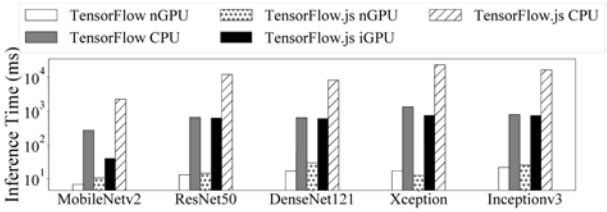


Figure 5: Inference time on pre-trained Keras models. The y-axis is on log scale.

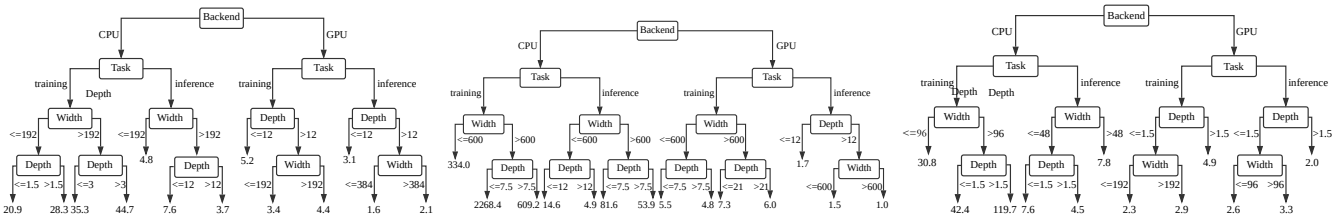
5.1.2 Results. Figure 5 shows the inference time for each model. It can be seen that the inference time of TensorFlow.js on nGPU is comparable (1x-2x slower) to native TensorFlow's. The most encouraging result is that the performance of TensorFlow.js on the iGPU backend is better than that of native TensorFlow on the CPU backend. This result is not surprising considering the computation capability of integrated graphics card and CPU. However, since traditional native DL frameworks do not support integrated graphics card for acceleration, DL tasks can benefit a lot from browsers in such a case with the help of integrated graphics card that is common on current devices.

Under the real-time requirement of client-side DL, if users want to achieve the experience of 10FPS (frame per second), they need to consider using a more powerful standalone graphics card. The Mobile Net model accelerated by iGPU can also meet the requirement. If the requirement is 1FPS, iGPU is also fully capable. But if only CPU can be used, then these common models are too heavy to run in browsers.

5.2 Decision Tree Analysis

In order to deeply reveal how different factors of DL tasks influence the performance gap between DL in browsers and on native frameworks, we build a predictive model based on decision tree analysis to study the factor importance.

5.2.1 Experiment Setup. We consider 4 factors that influence the performance gap between DL in browsers and on native platforms as shown in Table 5, including backend (CPU or GPU), task type (training or inference), as well as depth and width that represent the model complexity. In the DNN and RNN models, width refers to the number of neurons of each layer. In the CNN models, width refers to the number of kernels used in the convolution layer. For each of DNN, CNN and RNN, we choose one model from the Tensorflow.js official examples. The DNN and CNN models are used to recognize handwritten digits on the MNIST dataset, and the RNN model is to perform text generation from Nietzsche's writings. The



(a) DNN (b) CNN (c) RNN

Figure 6: Decision tree to analyze the time ratio of TensorFlow.js over native TensorFlow on DNN, CNN, and RNN Models.

Table 5: Contributing factors to the performance gap.

Network Type	Factor	Range
DNN	Backend	CPU, GPU
	Task Type	training, inference
	Depth	1, 2, 4, 8, 16
	Width	64, 128, 256, 512
CNN	Backend	CPU, GPU
	Task Type	training, inference
	Depth	6, 9, 15, 27
	Width	200, 400, 800
RNN	Backend	CPU, GPU
	Task Type	training, inference
	Depth	1, 2, 3
	Width	4, 8, 16, 32, 64, 256

range of depth and width is selected according to the values set in Tensorflow.js official examples.

In our experiment, we build and run the DNN, CNN and RNN models under different configurations using TensorFlow.js and native TensorFlow, respectively. Each configuration is a combination of values for the factors above. We measure the execution time of each configuration as the average time per batch for training tasks and average time per sample for inference tasks on two platforms. We use the ratio of the execution time on TensorFlow.js over that on native TensorFlow to quantify the performance gap.

5.2.2 Methodology. We run the decision tree algorithm with sklearn [10] to predict the ratio of execution time between TensorFlow.js and native TensorFlow. The decision tree depicts the relative importance of contributing factors. Intuitively, factors close to the root of the decision tree affect the time ratio more than those near the leaves. This is because the decision tree chooses to do the splitting of the nodes according to the Entropy-Information Gain criterion. In other words, the decision tree places the important factors near the root to gain the best prediction of splits.

Based on the results, we first produce a fully grown and unpruned decision tree for all the factors. In this way, each leaf contains only one configuration. Then we set the depth of the tree to the number of factors, in order to prevent using a factor several times on one path. Figure 6 shows the decision trees for DNN, CNN, and RNN.

5.2.3 Results. The execution time of TensorFlow.js is longer than native TensorFlow in almost every configuration.

Backend is the most important factor contributing to the performance gap. The ratio of execution time on the CPU backend is much higher than that on the GPU backend. For example, the ratio decreases from 44.7 to 4.4 for training tasks when the DNN model with depth over 3 and width over 192 runs on the GPU backend instead of on the CPU backend. The extreme case happens on the CNN. On the CPU backend, there is a wide range of the ratio from below 5 to over 2200 (when depth is less than 7.5 and width is over 600). However, when doing inference task on the GPU backend with depth over 12 and width over 600, TensorFlow.js performs as fast as native TensorFlow. This is because CNN makes use of the powerful computation capability of GPU when the model is large enough, yet not exceeding the upper bound of the browser memory.

The second most important factor is task type for all the three models. Performing training tasks exhibits a higher ratio, while the performance gap on inference tasks is small. For example, for the DNN model on the CPU backend, training tasks of TensorFlow.js perform 33.9 times slower than native TensorFlow on average, and inference tasks of TensorFlow.js perform 5.8 times slower than native TensorFlow on average.

The decision trees of DNN and RNN both suggest that the importance of depth and width depends on which backend the task is taken on. On the CPU backend, the importance of width outweighs that of depth, while depth plays a more important role on the GPU backend. However, in the case of CNN, width plays a more important role to the performance gap than depth for training tasks.

6 IMPLICATIONS

Table 6 summarizes the findings and implications of our study. Specifically, we draw implications for three stakeholders of DL in browsers: application developers, DL-framework vendors, and browser vendors. For application developers, we give recommendations on how to choose frameworks for DL in browsers, how to optimize the model, as well as how to select the backend. For DL-framework vendors, we present some advice on encoding of model files and optimizing the call stack. For browser vendors, we suggest on the utilization of system resources.

7 RELATED WORK

To the best of our knowledge, this paper is the first study to characterize the DL in browsers. So we survey related work on general client-side DL and performance measurement of DL systems.

Table 6: Major findings and implications of DL in browsers.

No.	Name	Finding	Implication	Stakeholder
1	Specific DL Tasks Support	Frameworks supporting DL in browsers are emerging and being actively maintained. Most of them are not for general purpose and support only a specific subset of DL tasks.	It is better for developers to use general-purpose DL frameworks like TensorFlow.js to implement their DL-powered Web applications.	Application Developer
2	Model Complexity	The width of DL models dominates the performance variation of both training and inference tasks considering the complexity of DL models.	Developers should pay attention to the width of their models, and balance the width and required performance if possible.	Application Developer
3	Model Loading	For inference tasks, loading and warming up the DL model accounts for much longer time than running the inference task itself. The warmup time on the integrated graphics card is generally shorter than that on the standalone graphics card.	Developers should pre-load and warm up the model before using it for inference.	Application Developer
4	Benefits from GPU	For popular pre-trained models like MobileNet and Inception, TensorFlow.js has comparable performance with native TensorFlow when running inference on the standalone graphics card.	It is possible to develop Web applications rather than native applications for these tasks.	Application Developer
5	Benefits from Integared Graphics Card	TensorFlow.js running on the integrated graphics card works better than native TensorFlow running on CPU backend.	For devices without standalone GPUs, developers can use the browser for DL tasks, leveraging integrated graphics card for acceleration.	Application Developer
6	Model File Encoding and Size	Model file encoded in JSON is much bigger (7x) in size than that encoded in binary, and significantly increases the model loading time.	It is better to encode DL models in binary files.	DL-Framework Vendor
7	Framework Call Stack	The call stack of TensorFlow.js is much deeper than that of ConvNetJS, pulling down the performance.	Framework vendors could leverage compiler optimization techniques to reduce the call stack when the DL models are used in the production environment.	DL-Framework Vendor
8	System Resource Utilization	The capability of multi-core CPU cannot be utilized when running DL tasks on the CPU backend in browsers since the JavaScript program is single-threaded. GPU memory usage is limited in 1GB, failing to load and run larger models.	JavaScript engine should take into account the support of multi-process or scheduling among multi cores for better performance of DL tasks in browsers. The GPU memory should be configurable for DL tasks.	Browser Vendor

7.1 Client-side Deep Learning

With the emphasis on privacy, personalization and timely response, it is a trend to conduct DL directly on the clients where the data is generated, especially on mobile devices. Lane et al. [27] studied the feasibility of using DL for typical mobile sensing tasks, such as activity recognition. Yao et al. [40] proposed DeepSense, a unified DL framework for processing time-series mobile sensing data. Despite of the increasing computation power of mobile devices, typical DL tasks are still of heavy workload for these resource-constraint devices. Several optimization methods were proposed to improve the performance of client-side DL. One line of work focuses on the DL models. Han et al. [22] proposed deep compression to compress the DNN through a three-stage method: pruning, trained quantization and Huffman coding, which showed a considerable reduction in terms of the storage requirements of DNNs. The other line of work leverages the cloud and edge environment to offload the computation-intensive tasks to powerful computation nodes [29]. Kang et al. [26] proposed Neurosurgeon, a lightweight scheduler to automatically partition DNN computation between mobile devices and data centers at the granularity of neural network layers. Wang et al. [39] designed Arden, a cloud-based deep learning framework for mobile devices. The framework partitions the DNN and offloads the resource-hungry training and complex inferences tasks to the cloud.

Anther usage scenario of client-side DL is to support the distributed deep learning. Teerapittayanon et al. [37] proposed distributed deep neural networks (DDNNs) over distributed computing hierarchies, consisting of the cloud, the edge (fog) and end devices. Ichinose et al. [24] proposed a pipelined method for distributed DL processing between mobile devices and the cloud to reduce the amount of data sent to the cloud and protect the privacy of users. Meeds et al. [33] designed MLitB, a prototype DL framework capable of performing large-scale distributed computing with heterogeneous classes of devices using Web browsers.

7.2 Performance Measurement of Deep Learning

In recent years, researchers have conducted studies to measure the performance for various kinds of deep learning tasks. Liu et al. [30] evaluated the performance of leading DL methods for object detection. Guignard et al. [21] presented detailed characterization results of a set of archetypal state-of-the-art DL workloads to identify the performance bottlenecks and to guide the design of prospective acceleration platforms in a more effective manner. Shi et al. [36] evaluated the performance of four state-of-the-art distributed DL frameworks over different GPU hardware environments. They built performance models of standard processes in training DNNs with SGD, and then benchmark the performance of the frameworks with three neural networks (i.e., AlexNet, GoogleNet and ResNet-50). As for DL on mobile devices, Ignatov et al. [25] studied state-of-the-art DL in the Android ecosystem and described available frameworks, programming models and the limitations of running AI on smartphones.

Although many JavaScript-based frameworks have been published to support DL in browsers, there is no comprehensive study to understand their characteristics and performance. Some researchers focus on the possibility of supporting DL in browsers by measuring the low-level browser capabilities. Malle et al. [31] presented a comparison study between native code and different browser-based implementations: JavaScript, ASM.js as well as WebAssembly on a representative mix of algorithms. However, these algorithms are not DL tasks. Their goal is just to show that the browsers performance is now comparable to and even exceeds native binary performance.

8 CONCLUSION

This paper made the first study on understanding the feasibility and performance of deep learning in Web browsers. We chose 7 recently emerging JavaScript-based DL frameworks and comprehensively revealed which type of DL tasks have been supported. We measured the performance of different frameworks when doing different DL tasks in browsers, and compared with the native DL framework to investigate the performance gap. Although the in-browser DL is still at the early stage, some interesting findings, e.g., the comparable performance of JavaScript frameworks to that of native ones on some types of DL tasks and the benefits gained from the integrated graphics card, can be useful and help guide the DL-powered Web applications. Additionally, we have also found that there are some potential space of improvement for currently in-browser DL frameworks, and plan to realize some practical solutions. We believe that our work can shed a light on the future of Web applications in the AI era.

ACKNOWLEDGMENTS

This work was supported by the National Key R&D Program of China under the grant number 2018YFB1004800, the National Natural Science Foundation of China under the grant number 61725201, the Beijing Municipal Science and Technology Project under the grant number Z171100005117002, and China Postdoctoral Science Foundation.

REFERENCES

[1] 2018. brain.js. https://github.com/BrainJS.
[2] 2018. Caffe. http://caffe.berkeleyvision.org/.
[3] 2018. CNTK. https://www.microsoft.com/en-us/cognitive-toolkit/.
[4] 2018. ConvNetJS. https://cs.stanford.edu/people/karpathy/convnetjs/.
[5] 2018. Keras.js. https://github.com/transcranial/keras-js.
[6] 2018. MIL WebDNN Benchmark. https://mil-tokyo.github.io/webdnn/#benchmar.
[7] 2018. Mind. https://github.com/stevenmiller888/mind.
[8] 2018. The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/.
[9] 2018. MorphCast. https://www.morphcast.com/.
[10] 2018. Sklearn DecisionTreeRegressor. https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html.
[11] 2018. synaptic.js. https://github.com/cazala/synaptic.
[12] 2018. Teachable Machine. https://teachablemachine.withgoogle.com/.
[13] 2018. TensorFlow Playgournd. http://playground.tensorflow.org.
[14] 2018. TensorFlow.js. https://js.tensorflow.org/.
[15] 2018. WebDNN. https://github.com/mil-tokyo/webdnn.
[16] 2018. WebGL. https://www.khronos.org/webgl/.
[17] 2018. WebGPU. https://www.w3.org/community/gpu/.
[18] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-Scale Machine Learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, (OSDI)*. 265–283.
[19] Michael Auer. 2012. Real-time Web GIS Analysis Using WebGL. *International Journal of 3-D Information Modeling (IJ3DIM)* 1, 3 (2012), 49–61.
[20] Bijin Chen and Zhiqi Xu. 2011. A Framework for Browser-Based Multiplayer Online Games Using WebGL and WebSocket. In *Proceedings of 2011 International Conference on Multimedia Technology (ICMT)*. 471–474.
[21] Mauricio Guignard, Marcelo Schild, Carlos S. Bederián, Nicolás Wolovick, and Augusto J. Vega. 2018. Performance Characterization of State-Of-The-Art Deep Learning Workloads on an IBM "Minsky" Platform. In *Proceedings of the 51st Hawaii International Conference on System Sciences, HICSS 2018*.
[22] Song Han, Huizi Mao, and William J. Dally. 2015. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. *CoRR* abs/1510.00149 (2015). http://arxiv.org/abs/1510.00149
[23] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A Fast Learning Algorithm for Deep Belief Nets. *Neural computation* 18, 7 (2006), 1527–1554.
[24] Ayae Ichinose, Atsuko Takefusa, Hidemoto Nakada, and Masato Oguchi. 2018. Performance Evaluation of Pipeline-Based Processing for the Caffe Deep Learning Framework. *IEICE Transactions* 101-D (2018), 1042–1052.
[25] Andrey Ignatov, Radu Timofte, William Chou, Ke Wang, Max Wu, Tim Hartley, and Luc Van Gool. 2018. AI Benchmark: Running Deep Neural Networks on Android Smartphones. In *Computer Vision - ECCV 2018 Workshops*. 288–314.
[26] Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor N. Mudge, Jason Mars, and Lingjia Tang. 2017. Neurosurgeon: Collaborative Intelligence Between the Cloud and Mobile Edge. In *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems, (ASPLOS)*. 615–629.
[27] Nicholas D Lane and Petko Georgiev. 2015. Can Deep Learning Revolutionize Mobile Sensing?. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications (HotMobile)*. 117–122.
[28] Yann LeCun et al. 1989. Generalization and Network Design Strategies. *Connectionism in perspective* (1989), 143–155.
[29] Weiqing Liu, Jiannong Cao, Lei Yang, Lin Xu, Xuanjia Qiu, and Jing Li. 2017. AppBooster: Boosting the Performance of Interactive Mobile Applications with Computation Offloading and Parameter Tuning. *IEEE Transactions on Parallel and Distributed Systems* 28, 6 (2017), 1593–1606.
[30] Y. Liu, P. Sun, M. R. Highsmith, N. M. Wergeles, J. Sartwell, A. Raedeke, M. Mitchell, H. Hagy, A. D. Gilbert, B. Lubinski, and Y. Shang. 2018. Performance Comparison of Deep Learning Techniques for Recognizing Birds in Aerial Images. In *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*. 317–324.
[31] Bernd Malle, Nicola Giuliani, Peter Kieseberg, and Andreas Holzinger. 2018. The Need for Speed of AI Applications: Performance Comparison of Native vs. Browser-based Algorithm Implementations. *CoRR* abs/1802.03707 (2018). http://arxiv.org/abs/1802.03707
[32] Chris Marrin. 2011. WebGL specification. *Khronos WebGL Working Group* (2011).
[33] Edward Meeds, Remco Hendriks, Said Al Faraby, Magiel Bruntink, and Max Welling. 2015. MLitB: machine learning in the browser. *PeerJ Computer Science* 1 (2015), e11.
[34] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L. Lewis, and Satinder P. Singh. 2015. Action-Conditional Video Prediction using Deep Networks in Atari Games. In *NIPS*.
[35] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *nature* 323, 6088 (1986), 533.
[36] Shaohuai Shi, Qiang Wang, and Xiaowen Chu. 2018. Performance Modeling and Evaluation of Distributed Deep Learning Frameworks on GPUs. In *Proceedings of IEEE DASC/PiCom/DataCom/CyberSciTech 2018*. 949–957.
[37] Surat Teerapittayanon, Bradley McDanel, and HT Kung. 2017. Distributed Deep Neural Networks over the Cloud, the Edge and End Devices. In *Proceedings of IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. 328–339.
[38] Aaron Tucker, Adam Gleave, and Stuart Russell. 2018. Inverse Reinforcement Learning for Video Games. *arXiv preprint arXiv:1810.10593* (2018).
[39] Ji Wang, Jianguo Zhang, Weidong Bao, Xiaomin Zhu, Bokai Cao, and Philip S. Yu. 2018. Not Just Privacy: Improving Performance of Private Deep Learning in Mobile Cloud. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, (KDD)*. 2407–2416.
[40] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. 2017. DeepSense: A Unified Deep Learning Framework for Time-Series Mobile Sensing Data Processing. In *Proceedings of the 26th International Conference on World Wide Web*. 351–360.

Exploring User Behavior in Email Re-Finding Tasks

Joel Mackenzie*
RMIT University
Melbourne, Australia
joel.mackenzie@rmit.edu.au

Kshitiz Gupta
Microsoft
Bellevue, WA, United States
kshitiz.gupta@microsoft.com

Fang Qiao
Microsoft
Bellevue, WA, United States
fang.qiao@microsoft.com

Ahmed Hassan Awadallah
Microsoft Research
Redmond, WA, United States
hassanam@microsoft.com

Milad Shokouhi
Microsoft
Bellevue, WA, United States
milads@microsoft.com

ABSTRACT

Email continues to be one of the most commonly used forms of online communication. As inboxes grow larger, users rely more heavily on email search to effectively find what they are looking for. However, previous studies on email have been exclusive to enterprises with access to large user logs, or limited to small-scale qualitative surveys and analyses on limited public datasets such as Enron[1] and Avocado[2]. In this work, we propose a novel framework that allows for experimentation with *real* email data. In particular, our approach provides a realistic way of simulating email re-finding tasks in a crowdsourcing environment using the workers' personal email data. We use our approach to experiment with various ranking functions and quality degradation to measure how users behave under different conditions, and conduct analysis across various email types and attributes. Our results show that user behavior can be significantly impacted as a result of the quality of the search ranker, but only when differences in quality are very pronounced. Our analysis confirms that time-based ranking begins to fail as email age increases, suggesting that hybrid approaches may help bridge the gap between relevance-based rankers and the traditional time-based ranking approach. Finally, we also found that users typically reformulate search queries by either entirely re-writing the query, or simply appending terms to the query, which may have implications for email query suggestion facilities.

KEYWORDS

email search; user behavior; search interface; search result page; result degradation

ACM Reference Format:
Joel Mackenzie, Kshitiz Gupta, Fang Qiao, Ahmed Hassan Awadallah, and Milad Shokouhi. 2019. Exploring User Behavior in Email Re-Finding Tasks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313450

*Research was conducted while at Microsoft Research.
[1]https://www.cs.cmu.edu/~./enron/
[2]https://catalog.ldc.upenn.edu/LDC2015T03

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313450

1 INTRODUCTION

For 50 years, email has been the mainstay of online communication. Email continues to be used widely for both business and personal communication, with a projected 4.2 billion worldwide users sending over 330 billion emails *per day* by the year 2022 [25]. As cheaper data storage has resulted in an increase in email storage quotas, users are becoming less inclined to delete email [12]. This has implications on the way in which users interact with their email client; they need to be more persistent in re-finding tasks, especially if the email is ranked by time. To this end, recent work has explored the use of relevance ranking in email search, showing effectiveness improvements with respect to ranking by time [2, 12, 14]. Although this is a promising step towards improving the email search experience, there is still a large gap in understanding how users interact with email search systems, particularly those that use relevance ranking. This is due to the difficult nature of experimenting within this context – email data is highly personal, private, and exclusive to commercial email providers. Additionally, users are often interested in finding a single known message [18].

Based on this knowledge, we propose a novel offline evaluation framework that allows us to examine how users behave on realistic email search tasks using their personal email data. Our approach allows for a vast number of experiments to be conducted across realistic email search scenarios, using real data from real users. For instance, we examine user behavior in email search tasks using both the traditional rank-by-time result presentation, as well as more modern rank-by-relevance and hybrid ranking approaches as illustrated in Figure 1. Our framework also allows us to compute implicit success metrics such as Session Success Rate (SSR), in addition to relevance-based metrics such as Mean Reciprocal Rank (MRR) [46], for different settings. We also compare the query reformulation of users across various experimental settings. We focus on the following research questions:

RQ1: *How does user behavior differ with respect to different email ranking approaches?*

RQ2: *How do the characteristics of email impact re-finding success?*

RQ3: *How does search quality affect user behavior in email re-finding?*

In answering these questions, our main contributions are as follows:

1. We describe a novel approach for simulating email re-finding tasks that can be used for user behavior studies and offline evaluation experimentation in crowdsourcing environments (Section 2), and

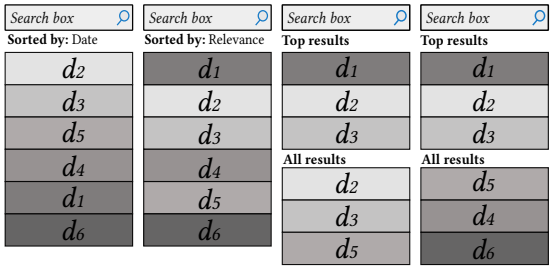


Figure 1: A pictorial example of the four different systems that we compare using the offline evaluation framework introduced in this paper. From left to right: Time (T_0): ranked by time, Rel (T_1): ranked by relevance, HybDup (T_2): hybrid relevance-time with duplicates, and Hyb (T_3): hybrid relevance-time without duplicates. Following the example of Carmel et al. [14], we denote message freshness by the shade of the cell, and we denote the relevance ranking as the document subscript.

2. We leverage this approach to characterize user behavior in email re-finding tasks across a wide range of factors such as ranking quality, SERP format, and email type (Section 4).

The remainder of the article is organized as follows: Section 2 provides some brief background and outlines our novel framework for simulating email re-finding, Section 3 describes the analysis we conduct, Section 4 reports on the experimental results, Section 5 validates our experimental framework, Section 6 outlines related work in user behavior analysis and email ranking, and Section 7 summarizes and concludes this article.

2 SIMULATING EMAIL RE-FINDING

Personal search tasks (e.g. email search) have been very difficult to study in realistic settings due to the nature of the email search process and the limitations on using and sharing email data. Email search is often characterized as a re-finding task [18] where a user is looking for a *specific known message* rather than looking for general information or exploring [4]. Additionally, since email data is highly personal, queries and relevance annotations are not readily available for both learning to rank email and evaluating email search systems. While synthetic approaches for data generation have been explored previously [1], they do not provide a realistic setting since the email data does not belong to the user conducting the search task. Instrumenting email clients was also proposed [20, 22, 30]; while allowing for a realistic setting, this approach provides no control over the experimental system. Hence, it is more suitable for logging user interactions rather than experimentation. This is analogous to using online experimentation and log gathering [12, 14] in large scale search settings, which is not possible without a real system and thousands or millions of users. Here, we introduce a novel approach for simulating email re-finding tasks. Our approach is deployed as crowdsourced *human intelligence tasks* (HITs), making it suitable for academic settings, while also allowing *real* email interaction data to be studied in a realistic and privacy-preserving manner.

2.1 Process

The task is broken into three distinct stages, each shown in Figure 2. First, the user is shown an email message e_t, known as the *target* message, which is sampled from the most recent N emails in their *inbox* folder (we set N to 1000 in our experiments). This screen is shown to the user for t_{read} seconds, in which the user can read the body of the message (Figure 2, left). Based on some preliminary experimentation, we set $t_{read} = 5$ seconds. This provides enough time for the user to 'get the gist' of the target email without learning or memorizing particular phrases from it. After t_{read} seconds have lapsed, the user is then shown a second screen with a questionnaire (see Section 2.2). Although we do not limit the amount of time a user can spend answering the questionnaire, we denote the time they take in this phase as $t_{question}$ (Figure 2, center). In the final stage of the task, the user is presented with a standard email search screen. On the top, a search bar is present, with the search engine results page (SERP) on the left side of the page and the body of the *current* email shown on the right (Figure 2, right). The user may scroll down the SERP and click on any of the emails to make them appear in full on the right pane. Once the user locates and opens the target message, a pop-up message informs the user that they successfully located the message, and the task is ended. For this phase of the task, the user may spend up to t_{search} seconds. If they do not find the target email in this time, then the search task is deemed as failed. We set $t_{search} = 120$ seconds. When showing the SERP, we show 20 documents in total. Although commercial providers generally provide more results to the user (such as the 50 per page[3], or an infinite scroll[4]), serving just 20 results (rather than 50 or more) improves the efficiency of the API calls, minimizing the impact of latency on user behavior [7, 10]. We leave exploration of alternative SERPs as future work. Note that we also allow the user to enter free-form feedback upon the completion or failure of a task if they wish.

2.2 Questionnaire

For completeness, we now outline the questions presented to the users. All questions require *yes* or *no* answers, making them less cognitively expensive than other formats (such as those employing a Likert scale). We asked the worker seven questions in total about aspects of the target message:

- Did you reply to this email?
- Is this the type of email that you would likely search for?
- Was this email a system generated email?
- Was this email important to you?
- Do you get more than 10 emails from this sender in a week?
- Do you remember reading this email before?
- Was the email too long to read?

The aim of this questionnaire, besides offering a momentary distraction from the target email content, is to gain insight into the importance of the target email as perceived by the user. Furthermore, this allows additional quality control of the judges, as answers to some of the questions can be validated using the search API, such as whether the message was read, replied to, marked as important, and so on.

[3]https://mail.google.com/
[4]https://outlook.office.com/owa/

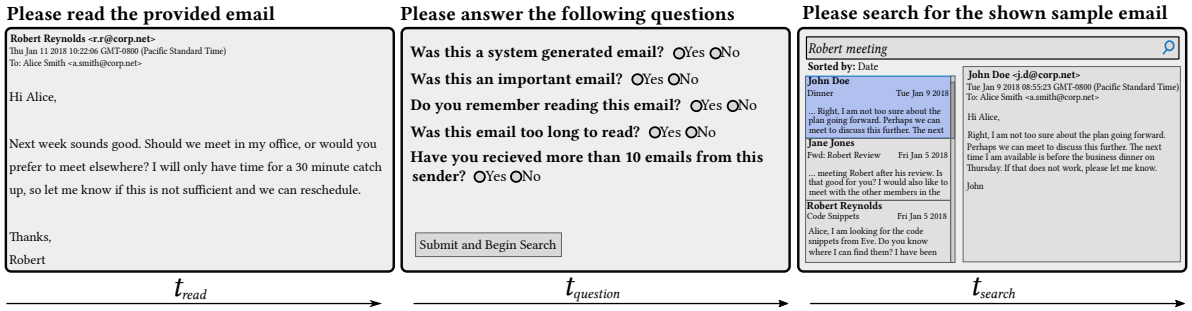


Figure 2: A pictorial example of our experimental framework. First, the user is shown an email sampled from their inbox for t_{read} seconds. Next, they must answer a short questionnaire. Finally, the user is taken to the search window where they have t_{search} seconds to find the email that they were shown. Once the user clicks on the target email, the task is ended, and the user is taken to a new task.

2.3 Metrics

Since we are interested in characterizing user behavior, we employ a range of measurements for comparing behavior across different treatments. Although these types of measurements are quite standard [29, 36, 40], we elucidate them for clarity. We also note that all time-based measures are calculated from the time in which the user enters the *search* phase of the HIT.

Task level interactions.

- *Success rate:* The percentage of successful tasks for the given configuration.
- *Time to success:* The time taken for the user to find the target message.
- *Number of queries:* The total number of issued queries.
- *Rate of abandonment:* The rate of reformulation without any message accesses.

Query level interactions.

- *Query action:* Reformulate/Succeed/Fail.
- *Time per SERP:* Average time spent per SERP.
- *Number of clicks:* The average number of clicks per SERP.
- *Query length:* The length (number of terms) of the submitted query.
- *Rank of clicks:* The average click depth.
- *Lowest ranked click:* The average rank of the lowest document accessed per SERP.
- *Time to first click:* The time taken for the user to click the first accessed document.
- *Time to deepest click:* The time taken for the user to click the deepest accessed document.

These interaction signals can also be regarded as online metrics, which would commonly be collected and analyzed in online systems. One disadvantage in real online systems is that the ground truth is not known, meaning that typical offline effectiveness metrics are not easy to deploy. Since we have access to the ground truth, we can deploy offline effectiveness metrics here. We use *Mean Reciprocal Rank* to measure the performance of the various systems across all provided queries, which is defined as

$$\text{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{e_i} \tag{1}$$

where Q is the set of queries, and e_i is the rank of the target email in the SERP for query i. MRR is a suitable metric for a re-finding task as there is only *one* email that the user is looking for.

2.4 Prerequisites

We rely on the Microsoft Graph API[5] as an interface between the workers' mailbox and our task. The API provides all query matching and ranking functionality. Therefore, a crowdworker must have a Microsoft managed email address such as one at `outlook.com`, `live.com`, `hotmail.com`, or `msn.com` to be eligible to enroll in our task. Alternatively, they are provided with an option to import their inbox into an Outlook inbox such that the API is available for use. Other email systems have similar APIs. For example, our system also supports the Gmail API[6], but we only present results based on the Microsoft Graph API here for simplicity.

3 SETTINGS

In the previous section, we described our approach for simulating email re-finding tasks. In this section, we describe the experimental settings for the study we conducted to answer the research questions outlined earlier.

3.1 Search Results Grouping

We explore three ranking conditions that are found in real email systems. The first refers to the traditional approach of ranking all email that matches the query by *time*, the most common technique for ranking email. The second approach refers to ranking email by *relevance*, whereby the results are ranked by their estimated relevance to the user query. Finally, the *hybrid* approach refers to providing the top-3 results according to the relevance model, followed by the standard ranking of matched emails by time [14, 42].

3.2 Search Results Quality

Our experimental framework allows us to purposefully degrade the quality of the search results [27, 45], which is useful for both validation and experimental analysis. Firstly, it allows us to validate that our metrics align with the user experience of the system; when the quality of the results is degraded, we expect to observe the user

[5]https://developer.microsoft.com/en-us/graph/
[6]https://developers.google.com/gmail/api/

taking longer to find the target email, inputting more searches per task, and so on. Secondly, it enables us to add another condition to our study by degrading the quality of the search results and observing how users behave when they are failing. For our system, we employ a very simple quality degradation mechanism. On a per-task basis, we set some probability P that we will remove the target, e_t, from the SERP. Then, on a per-query basis, we remove e_t with probability P. We plan to look at alternative degradation approaches in future work, where the target email can be demoted in various ways [45].

3.3 Search Results Duplication

Since the hybrid ranking uses both relevance and time ranking, it is possible that a document that is surfaced as being in the top-3 most relevant is also present in the most recent documents too. As such, we decided to study another condition to help us understand how duplicating the documents in the hybrid ranking affects user behavior [14, 42]. We instantiate two versions of the hybrid system with respect to the duplication condition: one version allows documents appearing in the "top results" pane to also appear in the "all results" section of the SERP, whereas the other version removes any duplicates found further down the ranking.

Based on the aforementioned conditions, we experiment with four unique systems, as shown in Figure 1. In practice, the two hybrid systems are very similar, though the Hyb system is guaranteed to have the same or better performance as compared to the HybDup system in terms of MRR, given the exclusion of duplicate documents.

Furthermore, note that our systems are similar but not directly comparable to those discussed by Carmel et al. [14], and we refer the interested reader to this work for both textual and graphical explanations of their systems. For our experiments, we treat Time as the control system, and measure differences between Time and all other systems using Dunnett's multiple comparison test [19].

4 EXPERIMENTS

We now employ our experimental approach to gain insight into how various rankings and interfaces impact user behavior and success in email re-finding tasks.

4.1 Users and Data

We employ 53 unique *trained* [28] judges to perform the given tasks; all judges were required to complete a set of *training tasks* before they were enrolled in the data collection tasks. Judges are not paid per HIT, but are paid an hourly rate, which we believe reduces incentive to game the tasks. Each judge could complete up to 90 tasks, and task configuration was randomized between tasks, meaning that each judge was equally likely to see each variation of the system. We also vary the probability of degradation, P, on a per-task basis, with $P \in \{0.0, 0.25, 0.50\}$. The judges completed a total of 3,678 tasks across a two-week period in August, 2018. Table 1 summarizes the data on both a per-judge level, and on a per-task level. Clearly, there is some variance in each measurement, as task difficulty can vary depending on the mailbox of the user, their search strategy, and so on. Although these statistics reflect expectation, we found that the age of the sampled emails is much older than anticipated, with a mean age of 142 days, and a median age of 67 days. Since we sample from the most recent 1,000 emails, this indicates that the judges do not receive a high volume of email (on average) to these personal accounts.

Table 1: Summary statistics on a per-judge level (top) and on a per-task level (bottom) for 53 unique judges across a total of 3,678 records.

	Mean (SD)	Median	Min	Max
Tasks	69.4 (12.8)	67	30	83
No. Successful	61.0 (13.1)	64	22	81
TTS per task	45.4 (11.0)	44.5	22.4	76.9
Queries per task	1.8 (0.4)	1.8	1.1	2.9
Clicks per task	2.6 (2.4)	1.7	0.9	12.9
Age [days]	141.9 (219.0)	64	0	1,911
TTS	45.3 (36.4)	30.4	3.7	120.0
No. queries	1.8 (1.4)	1	1	17
No. clicks	2.4 (4.9)	1	0	100

Table 2: Distribution of the questionnaire data collected from our HITs, shown as percentages. Each question pertains to the target email for the given task.

Email property	Percentage marked true
Machine Generated	72.6
Read [User Response]	9.8
Read [API]	30.6
Replied	4.2
Important	16.3
Popular Sender	30.1
Likely to Search	16.8

Table 2 summarizes the questionnaire data across each HIT. Most emails are classified by users as machine generated [34], which is not surprising since we are using personal web email accounts. Interestingly, users recalled reading around 10% of the sampled messages, whereas the API reported that 31% of these had been read. This can be explained in several ways. Firstly, users may not have perfect recall of whether they read an email or not. This could be related to the importance of the email, receiving a large volume of similar email, etc. Secondly, the API can only tell us if the email is marked as read, not whether the user actually reads the email. Another interesting observation is that only around 4% of the sampled mail was replied to, as contrasted with other studies that report reply rates of around 15% [49] and 20% [38]. Note that these studies were focused on enterprise data, where responses are more likely as compared with personal mail scenarios, where a lower reply rate be explained by the high volume of machine generated messages that saturate the user's inbox [34].

4.2 Ranker Effectiveness

Our first experiment evaluates the effectiveness of the various rankers we deploy. For the first query from each task, we retrieve the ranked list for each ranker, and evaluate the reciprocal rank for that particular ranked list. Figure 3 shows the MRR for each system

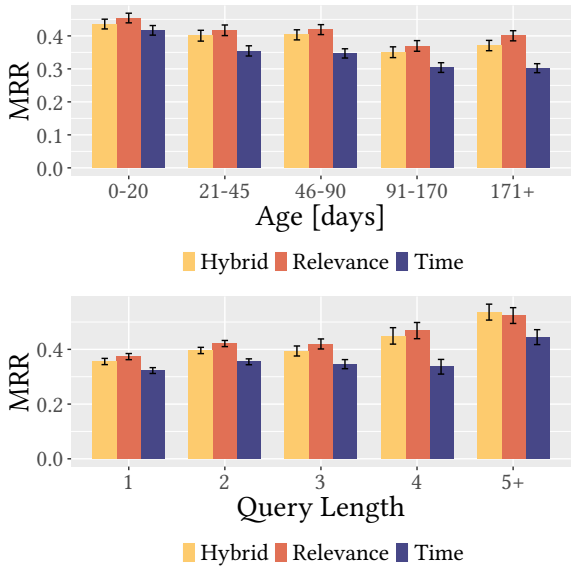


Figure 3: MRR and standard error for each system with respect to email age (top) and query length (bottom). Email ages are grouped such that ≈ 20% of all tasks are shown in each bucket.

for target emails of different ages and different input query lengths. Since we found that both hybrid systems have the same performance across this query set, we show only one *hybrid* bar in this analysis. Relevance based ranking provides the most effective ranking, followed closely by the hybrid ranking. As expected, ranking by time is most effective for new mail, as there is less chance that the ranker will surface matching (but irrelevant) messages. Even so, ranking by time is still inferior to the alternatives, at least with respect to MRR. Additionally, as query length increases, so too does the MRR for each ranker, which confirms the results from Carmel et al. [14]. The reason that longer queries are more effective in the email search paradigm is because email ranking is still based mostly on matching, whereas other applications such as web search rely heavily on click-through data and anchor text which is not readily available for email ranking. Interestingly, we found that most queries submitted in this experiment are either one (36%) or two (36%) terms, with only 15% containing three, 6% containing four, and 7% containing five or more terms, respectively. This supports observations from prior work; even though longer queries result in a better ranking, users opt to submit very short queries [13, 14, 30]. Using a two-tailed t-test with a Bonferroni correction, we find that both the hybrid and the relevance rankers significantly outperform the rank-by-time system ($p < 0.01$).

4.3 Behavior Comparison

We now evaluate the four different systems with respect to the interactions discussed in Section 3, thereby addressing RQ1. For this experiment, we are interested only in evaluating the systems when there is no quality degradation; after filtering out tasks where $P > 0$, we retain approximately 400 tasks per system.

User Behavior. Table 3 summarizes the observations we collected from the search task. Firstly, we observe that the success rates across the systems are very similar, with most tasks being completed successfully, while also aligning with rates observed in prior work [6]. It is also clear that the time to success does not vary widely between the systems, though the standard deviations are quite large. This suggests some variance in the difficulty of the search tasks. Users tend to issue less queries on the HybDup system, although not statistically significantly so. Users also submit less queries per task on systems other than Time, though the difference is small. Interestingly, the rate of abandonment is higher for Time than the other systems, which may suggest that users are more aware of *when* they should abandon and reformulate in a rank-by-time scenario, especially since this is what most users are accustomed to.

Looking more deeply at the query level analysis, we observe that tasks on Rel and Hyb result in statistically significantly shallower clicking on average compared to those on Time. Furthermore, the deepest clicks on Hyb were statistically significantly shallower than those on Time. Intuitively, this makes sense, because the MRR for these systems is significantly higher than on the rank-by-time system (Time). However, it is surprising that the same observation is not true for HybDup. One possibility is that, since HybDup is simply the same as Time with the addition of three 'champions', users can more easily fall back to the same search style they employ for Time, whereas the same notion is not true for Rel and Hyb. To further investigate this phenomenon, we analyze the tasks from HybDup which contain e_t in both the top-3 'champions' section of the ranker, as well as the rank-by-time section. It is expected that users begin at the top-ranked result and work down the ranked list, one document at a time, until they find the target document. However, we found that 6% of the time, users would click the *lower-ranked* instance of e_t, that is, in the rank-by-time part of the SERP. This suggests that some users may occasionally either miss or glance beyond the target document, or may just prefer to not examine the top-results. Although rarely used, this provides evidence of some benefit of duplication of top-results, and helps explain why users preferred duplication in the study from Carmel et al. [14].

In summary, we expected that the relevance and hybrid systems would result in significantly different behavior as compared to the system that ranks documents by time, as they are significantly better rankers based on MRR. However, we found that the ranking employed by the search systems does not largely impact user behavior (RQ1). One explanation for this outcome is that MRR is more sensitive than users are, in the sense that a statistically significant change in MRR does not necessarily mean that a user will notice such change.

Failure Analysis. While it is clear that users are generally quite good at successfully completing the provided task (Table 3), there are a small number of cases where users were unable to do so. To learn more about what caused users to fail, we conducted a failure analysis across these tasks (where $P = 0.0$). We define two types of failure: *Unforced failures* occur when the target document was surfaced on at least one of the SERPs that was viewed by the worker. That is, the user would have been able to successfully complete the task with at least one of the SERPs they were shown. The remainder of the failures are characterized as *forced failures*, where the user had

Table 3: Summary table of user behavior across the four systems using mean and SD. Only tasks with no degradation are considered. The rank of clicks, and deepest clicks, consider only queries in which at least one document was clicked.

Measure	Time (T_0)	Rel (T_1)	HybDup (T_2)	Hyb (T_3)
		Task Level		
Success Rate	93.8	92.2	93.9	92.7
TTS	36.70 (32.08)	36.37 (32.40)	34.15 (29.70)	37.22 (32.12)
Number of queries	1.52 (1.00)	1.50 (1.27)	1.40 (0.98)	1.45 (1.05)
Rate of abandonment	0.17 (0.29)	0.13 (0.26)	0.13 (0.28)	0.14 (0.28)
		Query Level		
Time on SERP	25.42 (18.91)	26.55 (20.72)	26.02 (19.38)	27.56 (21.63)
Number of clicks	1.36 (2.84)	1.60 (3.67)	1.39 (2.24)	1.18 (1.74)
Length of queries	2.28 (1.47)	2.23 (1.46)	2.07 (1.28)†	2.28 (1.39)
Rank of clicks	4.19 (4.19)	3.51 (3.98)†	4.02 (4.38)	3.33 (3.73)‡
Deepest click	4.75 (4.89)	4.16 (4.89)	4.58 (4.99)	3.77 (4.18)‡
Time to first click	11.90 (15.39)	12.24 (14.98)	10.99 (13.54)	13.03 (16.75)
Time to deepest click	12.94 (15.39)	13.25 (14.97)	12.23 (13.90)	13.65 (16.99)

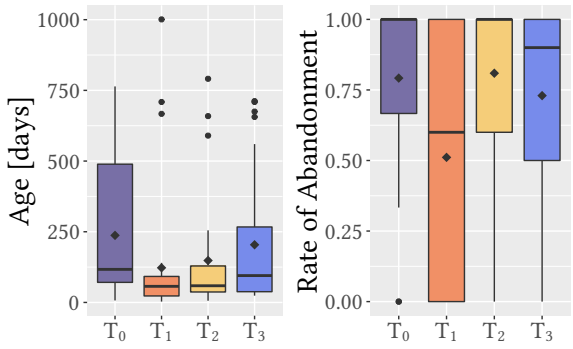


Figure 4: The age of the target email (left), and the abandonment rate within the task (right) for unsuccessful tasks.

no chance of completing the task since their queries never surfaced the target. There were 25, 33, 25, and 29 failed tasks for systems Time, Rel, HybDup, and Hyb, respectively, which represents around a 6–7% rate of failure. Of these, 4, 6, 4, and 0 were unforced, meaning that the majority of failures were due to users not being able to surface the target document at all. Figure 4 (left) shows the age of the target emails from the failed tasks on a per-system basis. Clearly, users struggle to surface older documents on Time. The converse is true for the relevance based ranking in Rel, where most failure cases were emails that were received within the last ≈ 125 days. The two hybrid systems appear somewhere between Time and Hyb. Figure 4 (right) shows the rate of abandonment across the failed tasks. Users abandoned the resultant SERPs at a very high rate, suggesting that they were quite confident that their query had not surfaced the target email, opting to reformulate instead. Interestingly, users were less likely to abandon the relevance based SERP (Rel) than those of the other systems. One explanation is that users have a much stronger intuition for when a time-ranked SERP has not provided the desired result in comparison to a relevance-based SERP.

Table 4: Examples of the different types of reformulation that were observed from our log and the percentage in which they are observed. 0.3% of cases did not fall into any of our categories, and are thus omitted here.

Type	Example	% Obs.
Full reform.	*grammarly* → *writing*	35.7
Partial reform.	*cv attached* → *cv provided*	11.6
Generalize	*out for delivery* → *delivery*	6.9
Specialize	*amazon* → *amazon order number*	23.3
Typo.	*fornt desk* → *front desk*	18.6
Revert	*new account* → *bank* → *new account*	3.6

4.4 Reformulation

Continuing our analysis of user behavior across the varying systems, our next experiment investigates how users reformulate their queries when they cannot find what they are searching for. We refer to a reformulation as the modification of query q which results in the next successive query q'. For this experiment, we consider all tasks where a reformulation was made, irrespective of the ranking system and degradation probability. In total, there were 2,794 reformulations from 1,445 tasks.

Categorizing Reformulations. In order to categorize the reformulation strategies, we first examine the query trace from each task with more than one query to understand what users are doing when they reformulate. Based on this analysis, we came up with six major strategies as defined in Table 4, similar to those defined by Jansen et al. [26] and Hassan [23]. Note that the strategy defined as *partial reformulation* refers to a combination of adding and removing terms where q' contains at least one term from q.

Next, we use a semi-automated process to categorize each subsequent query given all previous queries within the task. This process uses features such as the bag-of-words from each query, the edit distance from q to q', and the term ordering to determine the category of the reformulation. The output was then validated manually.

Table 4 also shows the proportion of the strategies that were employed. The most popular reformulation action is to generate an entirely new query, which occurs around 35% of the time. After further examination, it became clear that these full reformulations often came from users changing their search strategy. For example, many of the full reformulations involved users initially trying to use a keyword oriented query to find relevant results, and then reformulating to use a date instead (eg, *'amazon delivery'* → *'june 2018'*). The next most popular reformulation action is to specialize the previous query. This suggests that users were trying to improve precision. This is also a likely strategy for users of email search systems since queries are generally very short. Another common action was to fix typographical errors, which made up around 19% of all reformulations. It would be interesting to see how the proportion of typographical errors changes if query autosuggestion [13] was made available, but we leave this for future work. We also note that the relative proportions of reformulation strategies did not change with respect to the ranking algorithm used, which indicates that users are not sensitive to the ranker when considering their reformulation approach.

Comparing these findings to web search scenarios, we find that users typically reformulate typographical errors and generalize query terms with similar rates to those seen in web search, but are more likely to construct full reformulations in email r-finding [23]. Additionally, while other works have reported a reasonably large (up to 20%) use of advanced search operators such as '`from:`' or '`to:`' [4, 30], we observed almost no such occurrences (< 1% of submitted queries). We postulate that this is because these studies were from corporate environments, where users may be more accustomed to searching using operators. We must also note that we did not give the users any information on whether or not our system could handle such operators.

Behavior when Reformulating. We also examine the user behavior measures across successive reformulations to see if there is evidence of users becoming frustrated (or of any other changes in behavior). We found that, in general, user behavior did not change markedly across successive reformulations with one exception; as users continue to reformulate, they typically spend less time on the resultant SERP (Figure 5, left). Although we initially believed this to be due to increased time pressure [17], examining the time spent on each SERP reveals that users (generally) have used less than 50% of their allocated time at the time of their final reformulation. Next, we investigated the number of *unseen* documents that are surfaced in the SERP for each subsequent reformulation (Figure 5, right). Evidently, as users continue to reformulate, they are observing fewer new documents, which explains why they spend less time on the SERP. Similar trends were found across all system configurations.

Supplementing the answers to RQ1, our findings show that users typically submit short queries, and prefer to either specialize them, or to write entirely new queries when they cannot locate the target document, irrespective of the ranking quality or system in use. Furthermore, we observed that users tend to spend less time per successive SERP as they reformulate. This is caused by the ranker surfacing many documents the user has already seen, which in turn is caused by the query strategy taken by the user. These findings have implications for ranking in email and personal search; diversification or techniques for re-ranking that account for documents that the user has already seen [44] may be valuable in improving both the user experience and the engagement of the user, especially since it is known that users prefer to submit short queries and specialize as they reformulate.

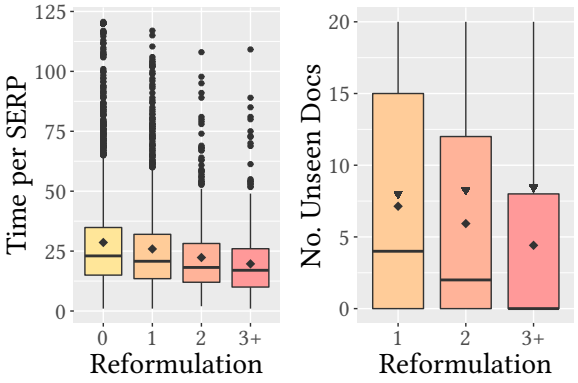


Figure 5: Time per SERP (left) and the number of unseen documents (right) for successive reformulations. On the right plot, the triangles denote the *average rank* of the unseen documents.

4.5 Email Properties and Importance

For our final experiment, we explore whether user behavior changes with respect to the properties of the sample email, thus answering RQ2. We define important emails as those that are marked as important, replied to, read, and those that the user is likely to search for. For each message, we use the sum of these importance criteria to define an overall importance score. For example, a message that was read, replied to, and marked as important would receive an importance score of 3. Figure 6 shows the time to success and the number of submitted queries across these levels of importance. We observe that the more important a message is to a user, the easier they are able to find it, issuing less queries and spending less time to succeed. This is because users are more likely to remember certain aspects of messages that are important to them, the converse being true for unimportant messages.

We were also interested in whether users found machine generated emails harder to search for. Our hypothesis is that since the majority of the mail received by the workers is machine generated, this would make searching for such mail more difficult (due to a high volume of similar messages). Table 5 shows the behavior across tasks, divided by whether the target message was generated by a human or a machine. We observe that users tend to submit shorter queries, click less, but go deeper in the ranking when the email is sent from a human, although none of these observations are statistically significant. Answering RQ2, users are able to find important emails faster and with less effort than those that are less important. Furthermore, whether the email is sent by a machine or by a real user tends to have no impact on the effort or success rate of re-finding, indicating that users are quite capable of searching for machine generated emails.

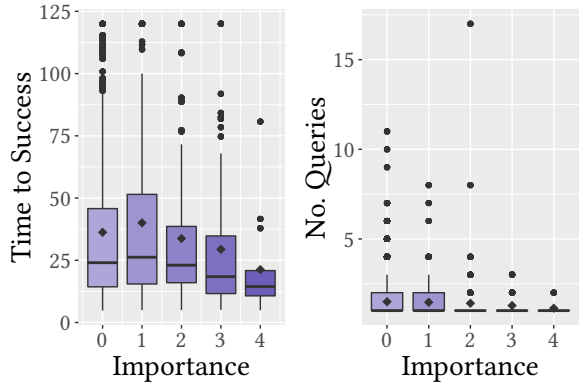


Figure 6: Time to success (left) and number of queries submitted (right) for emails of varying levels of importance.

Table 5: Summary of user behavior measures using mean and SD where the target email was sent by a human, or was an automated email. No significance was detected for any of the metrics shown.

Measure	Machine	Human
Task Level		
Success Rate	92.7	93.4
TTS	36.84 (32.45)	35.71 (31.14)
Number of queries	1.45 (1.07)	1.48 (1.09)
Rate of abandonment	0.14 (0.27)	0.15 (0.28)
Query Level		
Time on SERP	26.76 (22.30)	26.17 (18.94)
Number of clicks	1.47 (2.62)	1.34 (2.80)
Length of queries	2.39 (1.61)	2.14 (1.26)
Rank of clicks	3.51 (4.09)	3.89 (4.08)
Deepest click	4.13 (4.89)	4.41 (4.70)
Time to first click	12.62 (16.78)	11.73 (14.29)
Time to deepest click	13.66 (16.90)	12.68 (14.43)

5 VALIDATION

We now perform some analysis to validate the robustness of our experimental framework and collected data.

5.1 Sampling Target Email

A key aspect of our experimental framework is the way in which we sample the target email. As discussed in Section 2 and 3, we sample a random email from the most recent $N = 1{,}000$ emails in a workers inbox, ensuring an adequate sample size. However, this results in a large proportion of email being categorized as email that is not likely to be searched for by the worker (Table 2). Looking further at Table 2, it is clear that many of the sampled emails are in fact generated automatically by machines. While these emails can be useful and important, such as flight tickets, bills, or order invoices, they can also be general newsletters, repeated notifications, and other possibly annoying or low-utility content [34]. It is also important to consider the context in which the user is viewing the email. While some emails are unlikely to be searched *now*, the very same emails may have been sought by the user previously. One such example is a user who is waiting for a delivery from an e-commerce provider. Before the delivery is received, the user may search for the email to find tracking details, for instance. However, once the delivery is received by the user, they are unlikely to be interested in finding this email again. To ensure that the target email sampling does not bias our results, we compared the user behavior and ranking robustness where users classified emails as being either 'likely' or 'unlikely' to be searched for. We did not find any significant differences in user behavior or the performance of the tested systems. Hence, we believe that although our targets may not represent an email that is likely to be searched *now*, the process of re-finding is not impacted by our sampling method.

5.2 Judge Consistency and Quality Control

An important step in analysis of data collected from users is ensuring the quality of the data. Therefore, we conduct a number of analyses to confirm the data quality. First, we explore the click traces of the workers, looking for specific patterns of gaming (such as users clicking each document down the ranking with minimal time delay between clicks). We found very few instances of such behavior (5 occurrences) and removed them from further analysis. Since the target email was randomized and rotated, there were some occasions where a worker had a repeated task at a different point in time (possibly using a different ranking system). From all tasks that we deployed, 271 (7%) of these were repeated. To analyze the consistency of the judges, we compare the submitted queries for the initial task h with the repeated task h'. Our hypothesis is that if judges are consistent, their queries will be similar for the repeat task(s) as they were to the original task. To this end, we consider a judge to be inconsistent if there is no overlap between the submitted queries in h' as compared to h. We found that of the 271 repeat tasks, 51 were classed as inconsistent. Looking further at the inconsistent tasks themselves, it is clear that these inconsistencies arise from the worker employing a different email re-finding strategy, for example, *'2018 conference tickets'* → *'tentative program schedule'*. Furthermore, we only report analysis using trained judges who are trained to conduct our task. However, we did also experiment with employing workers on a per-HIT basis at 40 cents per task. We found the behavior to be consistent with the findings from the trained judges, except that crowdworkers employed on a per-hit basis were generally faster to succeed, which is intuitive given the difference in incentive.

5.3 Controlled Degradation

As discussed in Section 3, we are able to intentionally degrade the search ranking. When collecting data, we opted to run a few different configurations for result degradation, where we remove the target document with probability $P \in \{0.0, 0.25, 0.50\}$. This experiments allows us to study the impact of lower search quality on user behavior and it allows us to validate that our behavior measures move in the expected directions when the ranking quality changes.

Degradation Analysis. Figure 7 shows some of the key measures for each system as P is increased. As expected, the number of

submitted queries per task, the time to success, and the rate of abandonment increases steadily as the result quality becomes worse. Furthermore, users tend to click deeper as the quality becomes worse. These trends holds across all systems, and are statistically significant under a two-tailed paired t-test with a Bonferroni correction ($p < 0.01$). In addition, the success rates fall across the board as P increases, from around 93% ($P = 0.0$) to 87% ($P = 0.25$) and then to 76% ($P = 0.50$). On the other hand, no statistical significance was measured for the number of clicks, or the time spent on the SERP. This indicates that these metrics are not sensitive to ranking quality, and are unlikely to be informative in the experiments conducted in Section 4. So, in answering RQ3, we have shown that user behavior is correlated with the quality of the search experience, and that a degraded search experience will lead to users taking longer to succeed, searching more often, and abandoning the SERP more often. Contrasting these results to those observed in Section 4 (RQ1) suggests that while users are sensitive to the quality of the search experience, the quality difference (as measured by MRR) must be quite pronounced to observe significantly changed behavior. In any case, the clear correlation between quality and these metrics provides additional evidence supporting the validity of the collected data.

6 RELATED WORK

A multitude of prior work has investigated how users interact with the many aspects of Information Retrieval systems, including presentation, ranking, search task, temporal influences, and many others. Here, we review the relevant studies, including those that focus on user behavior in web search tasks, user behavior and characterization on email tasks, crowdsourcing and gamification, and ranking for email search.

6.1 Understanding User Behavior

Web Search. There are a multitude of studies that focused on web search environments and the many components therein. For example, recent studies have quantified how users behave with respect to the number of documents included in a SERP [29], the information content and length of snippets [36], the difficulty of the querying interface [9], and other aspects of web search such as result presentation, visual interfaces, result coherence, and user typing ability [5, 8, 17, 33, 41]. However, these studies are far less common in personal search environments due to the difficult nature of experimentation on personal data [11]. Our work follows this line of work but focuses on email re-finding tasks.

Email Search. While email use is at an all-time high, email search has not received as much attention from the IR community compared to other areas such as web search. As Carmel et al. [12] pointed out, this is likely due to the lack of publicly available data. While some collections are available, they often have limitations such as using synthetic queries [1], which are not reflective of true user intent on email finding tasks. Most of the prior work on user behavior for email tasks has come from query log analyses, user studies, and surveys. One line of work investigates how users organize their mail and how this impacts their re-finding strategies or actions [4, 6, 21, 38, 47]. Another line of related work explores modeling and predicting various aspects of email search such as

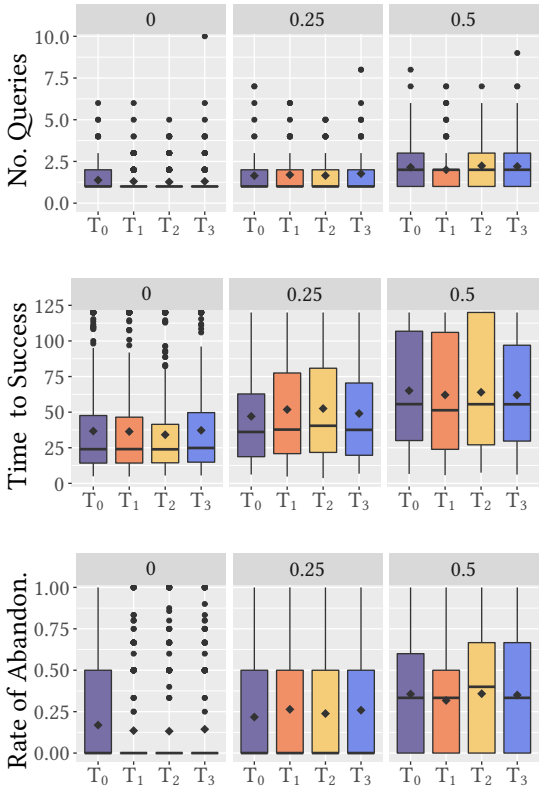


Figure 7: The effect of removing the target email e_t from the SERP with probability P on both the number of submitted queries, the time to success, and the rate of abandonment. Systems are shown in Figure 1. Degradation results in significantly more queries being submitted per task, a significantly longer time to success, and a significantly higher rate of abandonment.

how successful a user will be in their search task [30], how likely user will be to respond to new mail [49], how long a user's reply will be [31], and whether a user will defer an email [43], among many others. Elsweiler et al. [20] studied re-finding behavior by instrumenting the Mozilla Thunderbird[7] email client to capture user behavior. Their study uncovered several interesting findings such as that re-finding is difficult for users, and that users can become disoriented while performing re-finding tasks. Other studies have also observed the use of very short queries in email search [4, 13, 14]. Interestingly, users are generally successful at re-finding mail [6, 15], indicating that users are somewhat comfortable with re-finding even if such tasks are thought to be difficult. Our work addresses some of the major limitations of previous work by proposing an approach for simulating email re-finding tasks using real data in a crowdsourcing environment.

Crowdsourcing and Gamification. Crowdsourcing is a simple and cost effective approach for rapidly collecting data from users [32]. Prior work has involved using crowdsourcing to understand how users interact and behave with search tasks. The most relevant work

[7] https://www.thunderbird.net/

to our approach is that of Ageev et al. [3], who use *gamification* [37] to model user behavior in a crowdsourcing environment. This involves proposing a unique and scalable crowdsourcing task that allows user behavior to be captured across informational search tasks with a *known intent*. Subsequent works have focused on how user behavior can be studied via gamification [24], and the implications that such frameworks have on user behavior. Our approach is influenced from these ideas of leveraging crowdworkers with novel systems and tasks to collect user behavior for further analysis, allowing us to bypass many of the aforementioned issues such as experimentation on personalized data.

6.2 Ranking for Email Search

Some interest has been placed on ranking email results by *relevance* instead of the classic *rank-by-time* approach. Many of these ranking approaches were inspired directly from ad-hoc web search, including the use of language models [39] and BM25F [16]. Macdonald and Ounis [35] studied which combinations of email fields are useful as evidence for email ranking, suggesting that both the body and the subject of emails are crucial for effective retrieval.

Recently, the community has focused on the use of machine learning models for ranking email search, and the implications this has on users. Aberdeen et al. [2] outline the approach that Google uses for ranking email in their *Priority Inbox*. In particular, they focus on ranking documents by the predicted probability that the user will interact with the email within some time threshold after delivery. They make this prediction based on *social, content, label,* and *thread* features. They found that internal users of the priority inbox spent 13% less time reading unimportant mail, helping ease *email overload* [31, 48].

Advocating for rank-by-relevance, Carmel et al. [12] experimented with the implementation and deployment of a two-stage ranking model for effective email ranking, showing that a full relevance ranking significantly increases *Mean Reciprocal Rank (MRR)* as compared to rank-by-time.

Using a similar ranking system as described in [12], Ramarao et al. [42] described a new mail system called *InLook*. However, instead of ranking the entirety of the matched search results, InLook displays an *intent pane* which shows the top-3 most relevant matched emails, followed by the standard rank-by-time list. The authors argue that certain query types cannot be sufficiently matched using just rank-by-time or rank-by-relevance, providing justification for the hybrid ranking. Carmel et al. [14] further investigated hybrid rankings, arguing that users are not accepting rank-by-relevance in email search, likely because their search process is ingrained in the rank-by-time model. They propose a very similar approach to Ramarao et al. [42], where they provide the top-H results by relevance (called 'Heroes') above the remaining top-T by time, and explore a few variants of this hybrid approach. They found that providing the top-3 results based on relevance followed by the standard default time-ordered results, without removing duplicates, provided the best user experience (based on both MRR and a small-scale user study). This ranking, named *Heroes-Dup*, was subsequently deployed in Yahoo corporate and Yahoo web mail A/B testing scenario, significantly improving MRR over the baseline. Our work extended this line of research by exploring the implications that relevance ranking has on user behavior.

7 DISCUSSION AND CONCLUSION

A major contribution of this research is the proposed framework for studying email search using email data from real users in a realistic and privacy-preserving setting. We described an approach that is extensible, and can be modified to study a variety of different aspects of email re-finding, including alternative search interfaces and ranking approaches, among others. We also acknowledge some limitations and opportunities for future work. One interesting capability that was not explored in this work is to control the rank of the target email; since the target is known before the SERP is shown to the user, the target document could be strategically placed *anywhere* in the ranking. While our approach aims to be as realistic as possible, the random sampling of target emails may not truly reflect an email that a user would search for (Section 5). It would be interesting to see if more robust sampling approaches could be used, such as sampling only emails that are marked as important, or come from specific senders. A final aspect of interest is further gamification of the system, whereby the task could be appropriately modified to include additional incentives for crowdworkers. This would ultimately lead to more robust data collection in situations where *untrained* judges are employed, such as in typical crowdsourcing platforms.

Another major contribution of this work is leveraging our email search experimentation framework for analyzing user behavior in email re-finding tasks. Our results confirmed that relevance based ranking significantly outperforms rank-by-time for email search in terms of MRR, and that rank-by-time performs worse for re-finding older emails. We also showed that although both hybrid and relevance ranked systems are significantly better than the rank-by-time system, in terms of MRR, user behavior (in terms of success rate and time to success) does not markedly differ between the systems. This indicates that MRR may not be entirely representative of utility to users in re-finding tasks (RQ1 and RQ3). We also showed that users tend to do better with deciding when they should abandon the SERP and reformulate the query when rank-by-time is used. Another interesting observation was that some users occasionally either miss or glance beyond the target document when shown in the top results list of a hybrid SERP. Alternatively, they may have just preferred to not examine the top-results and skip to the more familiar rank-by-time setup. These observations show the challenges of trying to introduce new features that require users to change existing behavioral patterns and search strategies.

When studying how users reformulate their searches, we found similar behavior to web search but with more full reformulations which usually result from change in strategy (keyword search to time-based search). Finally, we also explored how properties of email impact re-finding success (RQ2). Most notably, users have more success finding emails that are deemed important. We found no evidence that machine-generated emails are more difficult to find. Our findings have several implications on designing email clients and email search systems. Our proposed framework for experimenting with email search could enable additional work to further study and evaluate email search systems by overcoming the

challenges around studying email search interactions in realistic settings.

REFERENCES

[1] S. AbdelRahman, B. Hassan, and R. Bahgat. 2010. A New Email Retrieval Ranking Approach. *IJCSIT* 2, 5 (2010), 44–63.

[2] D. Aberdeen, O. Pacovsky, and A. Slater. 2010. The Learning behind Gmail Priority Inbox. In *NIPS Workshop on LCCC*.

[3] M. Ageev, Q. Guo, D. Lagun, and E. Agichtein. 2011. Find It if You Can: A Game for Modeling Different Types of Web Search Success Using Interaction Data. In *Proc. SIGIR*. 345–354.

[4] Q. Ai, S. T. Dumais, N. Craswell, and D. Liebling. 2017. Characterizing Email Search Using Large-scale Behavioral Logs and Surveys. In *Proc. WWW*. 1511–1520.

[5] H. Ali, F. Scholer, J. A. Thom, and M. Wu. 2009. User Interaction with Novel Web Search Interfaces. In *Proc. OZCHI*. 301–304.

[6] T. Alrashed, A. H. Awadallah, and S. T. Dumais. 2018. The Lifetime of Email Messages: A Large-Scale Analysis of Email Revisitation. In *Proc. CHIIR*. 120–129.

[7] I. Arapakis, X. Bai, and B. B. Cambazoglu. 2014. Impact of Response Latency on User Behavior in Web Search. In *Proc. SIGIR*. 103–112.

[8] J. Arguello and R. Capra. 2014. The Effects of Vertical Rank and Border on Aggregated Search Coherence and Search Behavior. In *Proc. CIKM*. 539–548.

[9] L. Azzopardi, D. Kelly, and K. Brennan. 2013. How Query Cost Affects Search Behavior. In *Proc. SIGIR*. 23–32.

[10] X. Bai, I. Arapakis, B. B. Cambazoglu, and A. Freire. 2017. Understanding and Leveraging the Impact of Response Latency on User Behaviour in Web Search. *ACM Trans. Information Systems* 36, 2 (2017), 21:1–21:42.

[11] R. Capra and M. A. Pérez-Quiñones. 2006. Factors and Evaluation of Refinding Behaviors. In *SIGIR Workshop on PIM*. 16–19.

[12] D. Carmel, G. Halawi, L. Lewin-Eytan, Y. Maarek, and A. Raviv. 2015. Rank by Time or by Relevance? Revisiting Email Search. In *Proc. CIKM*. 283–292.

[13] D. Carmel, L. Lewin-Eytan, A. Libov, Y. Maarek, and A. Raviv. 2017. The Demographics of Mail Search and Their Application to Query Suggestion. In *Proc. WWW*. 1541–1549.

[14] D. Carmel, L. Lewin-Eytan, A. Libov, Y. Maarek, and A. Raviv. 2017. Promoting Relevant Results in Time-Ranked Mail Search. In *Proc. WWW*. 1551–1559.

[15] M. E. Cecchinato, A. Sellen, M. Shokouhi, and G. Smyth. 2016. Finding Email in a Multi-Account, Multi-Device World. In *Proc. SIGCHI*. 1200–1210.

[16] N. Craswell, H. Zaragoza, and S. Robertson. 2005. Microsoft Cambridge at TREC-14: Enterprise track. In *Proc. TREC*.

[17] A. Crescenzi, D. Kelly, and L. Azzopardi. 2015. Time Pressure and System Delays in Information Search. In *Proc. SIGIR*. 767–770.

[18] S. T. Dumais, E. Cutrell, J. J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins. 2003. Stuff I've Seen: A System for Personal Information Retrieval and Re-Use. In *Proc. SIGIR*. 72–79.

[19] C. W. Dunnett. 1955. A Multiple Comparison Procedure for Comparing Several Treatments with a Control. *J. American Stat. Assoc.* 50 (1955), 1096–1121.

[20] D. Elsweiler, M. Harvey, and M. Hacker. 2011. Understanding Re-finding Behavior in Naturalistic Email Interaction Logs. In *Proc. SIGIR*. 35–44.

[21] M. Grbovic, G. Halawi, Z. Karnin, and Y. Maarek. 2014. How Many Folders Do You Really Need? Classifying Email into a Handful of Categories. In *Proc. CIKM*. 869–878.

[22] M. Harvey and D. Elsweiler. 2012. Exploring Query Patterns in Email Search. In *Proc. ECIR*. 25–36.

[23] A. Hassan. 2013. Identifying Web Search Query Reformulation using Concept based Matching. In *Proc. EMNLP*. 1000–1010.

[24] J. He, M. Bron, L. Azzopardi, and A. de Vries. 2014. Studying User Browsing Behavior Through Gamified Search Tasks. In *Proc. GamifIR*. 49–52.

[25] The Radicati Group, Inc. 2018. Email Statistics Report, 2018–2022. https://www.radicati.com/wp/wp-content/uploads/2018/05/Email-Market-2018-2022-Executive-Summary.pdf. Online; accessed August, 2018.

[26] B. J. Jansen, D. L. Booth, and A. Spink. 2009. Patterns of Query Reformulation during Web Searching. *JASIST* 60, 7 (2009), 1358–1371.

[27] T. Jones, D. Hawking, P. Thomas, and R. Sankaranarayana. 2011. Relative Effect of Spam and Irrelevant Documents on User Interaction with Search Engines. In *Proc. CIKM*. 2113–2116.

[28] G. Kazai and I. Zitouni. 2016. Quality Management in Crowdsourcing Using Gold Judges Behavior. In *Proc. WSDM*. 267–276.

[29] D. Kelly and L. Azzopardi. 2015. How Many Results Per Page? A Study of SERP Size, Search Behavior and User Experience. In *Proc. SIGIR*. 183–192.

[30] J. Y. Kim, N. Craswell, S. T. Dumais, F. Radlinski, and F. Liu. 2017. Understanding and Modeling Success in Email Search. In *Proc. SIGIR*. 265–274.

[31] F. Kooti, L-C. Aiello, M. Grbovic, K. Lerman, and A. Mantrach. 2015. Evolution of Conversations in the Age of Email Overload. In *Proc. WWW*. 603–613.

[32] M. Lease and E. Yilmaz. 2011. Crowdsourcing for Information Retrieval. *SIGIR Forum* 45 (2011), 66–75.

[33] Z. Liu, Y. Liu, K. Zhou, M. Zhang, and S. Ma. 2015. Influence of Vertical Result in Web Search Examination. In *Proc. SIGIR*. 193–202.

[34] Y. Maarek. 2017. Web Mail is Not Dead! It's Just Not Human Anymore. In *Proc. WWW*. 5–5.

[35] C. Macdonald and I. Ounis. 2006. Combining Fields in Known-item Email Search. In *Proc. SIGIR*. 675–676.

[36] D. Maxwell, L. Azzopardi, and Y. Moshfeghi. 2017. A Study of Snippet Length and Informativeness: Behaviour, Performance and User Experience. In *Proc. SIGIR*. 135–144.

[37] M. Meder, F. Hopfgartner, G. Kazai, and U. Kruschwitz. 2016. Third International Workshop on Gamification for Information Retrieval (GamifIR'16). In *Proc. SIGIR*. 1239–1240.

[38] K. Narang, S. T. Dumais, N. Craswell, D. Liebling, and Q. Ai. 2017. Large-Scale Analysis of Email Search and Organizational Strategies. In *Proc. CHIIR*. 215–223.

[39] P. Ogilvie and J. Callan. 2005. Experiments with Language Models for Known-Item Finding of E-mail Messages. In *Proc. TREC*.

[40] K. Ong, K. Järvelin, M. Sanderson, and F. Scholer. 2017. Using Information Scent to Understand Mobile and Desktop Web Search Behavior. In *Proc. SIGIR*. 295–304.

[41] K. Ong, K. Järvelin, M. Sanderson, and F. Scholer. 2018. QWERTY: The Effects of Typing on Web Search Behavior. In *Proc. CHIIR*. 281–284.

[42] P. Ramarao, S. Iyengar, P. Chitnis, R. Udupa, and B. Ashok. 2016. InLook: Revisiting Email Search Experience. In *Proc. SIGIR*. 1117–1120.

[43] B. Sarrafzadeh, A. H. Awadallah, C. H. Lin, C-J. Lee, M. Shokouhi, and S. T. Dumais. 2019. Characterizing and Predicting Email Deferral Behavior. In *Proc. WSDM*. 627–635.

[44] M. Shokouhi, R. W. White, P. N. Bennett, and F. Radlinski. 2013. Fighting Search Engine Amnesia: Reranking Repeated Results. In *Proc. SIGIR*. 273–282.

[45] P. Thomas, T. Jones, and D. Hawking. 2011. What Deliberately Degrading Search Quality Tells Us About Discount Functions. In *Proc. SIGIR*. 1107–1108.

[46] E. M. Voorhees. 1999. The TREC-8 Question Answering Track Report. In *Proc. TREC-8*. 77–82.

[47] S. Whittaker, T. Matthews, J. Cerruti, H. Badenes, and J. Tang. 2011. Am I Wasting My Time Organizing Email? A Study of Email Refinding. In *Proc. SIGCHI*. 3449–3458.

[48] S. Whittaker and C. Sidner. 1996. Email Overload: Exploring Personal Information Management of Email. In *Proc. SIGCHI*. 276–283.

[49] L. Yang, S. T. Dumais, P. N. Bennett, and A. H. Awadallah. 2017. Characterizing and Predicting Enterprise Email Reply Behavior. In *Proc. SIGIR*. 235–244.

Jointly Leveraging Intent and Interaction Signals to Predict User Satisfaction with Slate Recommendations

Rishabh Mehrotra
Spotify
rishabhm@spotify.com

Mounia Lalmas
Spotify
mounial@spotify.com

Doug Kenney
Spotify
dkenney@spotify.com

Thomas Lim-Meng
Spotify
tommeng@spotify.com

Golli Hashemian
Spotify
golli@spotify.com

ABSTRACT

Detecting and understanding implicit measures of user satisfaction are essential for enhancing recommendation quality. When users interact with a recommendation system, they leave behind fine grained traces of interaction signals, which contain valuable information that could help gauging user satisfaction. User interaction with such systems is often motivated by a specific need or intent, often not explicitly specified by the user, but can nevertheless inform on how the user interacts with, and the extent to which the user is satisfied by the recommendations served. In this work, we consider a complex recommendation scenario, called *Slate Recommendation*, wherein a user is presented with an ordered set of collections, called *slates*, in a specific page layout. We focus on the context of music streaming and leverage fine-grained user interaction signals to tackle the problem of predicting user satisfaction.

We hypothesize that user interactions are conditional on the specific intent users have when interacting with a recommendation system, and highlight the need for explicitly considering user intent when interpreting interaction signals. We present diverse approaches to identify user intents (interviews, surveys and a quantitative approach) and identify a set of common intents users have in a music streaming recommendation setting. Additionally, we identify the importance of shared learning across intents and propose a multi-level hierarchical model for user satisfaction prediction that leverages user intent information alongside interaction signals. Our findings from extensive experiments on a large scale real world data demonstrate (i) the utility of considering different interaction signals, (ii) the role of intents in interpreting user interactions and (iii) the interplay between interaction signals and intents in predicting user satisfaction.

ACM Reference Format:
Rishabh Mehrotra, Mounia Lalmas, Doug Kenney, Thomas Lim-Meng, and Golli Hashemian. 2019. Jointly Leveraging Intent and Interaction Signals to Predict User Satisfaction with Slate Recommendations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313592

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313592

1 INTRODUCTION

An increasingly larger proportion of users rely on recommendation systems to pro-actively serve them recommendations based on diverse user needs and expectations. Developing a better understanding of how users interact with such recommender systems is important not only for improving user experience but also for developing satisfaction metrics for effective and efficient optimization of the recommendation algorithm. This is especially true in the case of online streaming services like Pandora, Spotify and Apple Music, wherein the system could gauge user satisfaction and adapt its recommendations to better serve user needs.

Since obtaining explicit feedback from users is prohibitively expensive and challenging to implement in real world systems, commercial systems rely on exploiting *implicit* feedback signals derived from user activity. When users interact with the recommendations served, they leave behind fine-grained traces of interaction patterns, which could be leveraged for predicting how satisfied was their experience, and for developing metrics of user satisfaction.

Prior work have studied implicit feedback signals (e.g., clicks, dwell time, mouse scrolling) in the case of web search, and verified their effectiveness in predicting user satisfaction, both on traditional desktop [9, 11, 26, 27] and mobile setting [17, 39]. Furthermore, past work in web search systems has also highlighted the importance of considering tasks and intents when interpreting implicit feedback signals [14, 38]. On the other hand, the identification and effectiveness of corresponding implicit feedback signals as well as the role of user intents has remained understudied in the context of recommendation systems, especially in the mobile context.

While search systems have access to explicit queries from users, based on which one could extract tasks, and interpret interaction signals, thereby differentiating between success and failure; recommender systems, on the other hand, lack such explicit indicators of user intent and clear indicators of success. Indeed, the interpretation of signals varies with goals; for example, scrolling can indicate negative experience when the goal is to quickly listen to music now, but can also indicate a positive experience when the goal is to browse the diverse collection of music the system has to offer. Furthermore, interpreting interaction signals becomes especially hard in the context of complex recommendation settings, like *Slate recommendations*, a scenario typical to many music streaming services, where users are recommended a set of collections (called *slates*), with different purposes (to explore new music, or quickly jump to recently played music, etc), and heterogeneous content (playlists, artist profiles, other audio content, etc). Thus, there is a need for a

detailed, holistic view of user interactions with such recommender systems, to establish their utility in predicting satisfaction.

In this work, we consider the use case of music streaming via slates of recommendation, and aim at understanding the relationship between interaction signals, user intents and user satisfaction. We consider the case of users interacting with a mobile based music streaming app, Spotify, and investigate the different interaction signals that can be extracted in slate recommendation setting, and how these interaction signals vary across different intents. Since the list of possible intents user might have is hitherto unknown, we adopt a mixed methods approach to understand user intents, and leverage insights from (i) face-to-face interviews, (ii) large scale in-app survey and (iii) non-parametric clustering techniques to identify the list of possible intents. We identify eight user intents and verify their validity using large scale log data.

To predict user satisfaction, we jointly leverage insights from the extracted interaction signals and intents. While interaction signals have been directly used to predict satisfaction in search [9, 24, 27], we instead hypothesize that interpreting interaction signals without factoring in user intent would lead to noisy, unreliable estimates of satisfaction in our context. We demonstrate that traditional approaches of using a single, global model for satisfaction trained on aggregate data without considering intent groups does not perform well in predicting satisfaction. We show that significant performance gains are obtained in switching from a single global prediction model to separate satisfaction models for each intent.

Further, we identify issues with the global and per-intent models. While the former fails to capture intent-specific intricacies, separate per-intent models ignore information and insights from other intents. To address these, we show the importance of shared learning across intents and propose a *multi-level hierarchical model* that allows the estimation of intent-specific effects while at the same time learning from data across all intents, thereby compromising between the overly noisy per-intent model and the oversimplified global model that ignores intent groups. The proposed multi-level model facilitates incorporating both individual session level and intent group level effect on user satisfaction, thereby allowing for variability in user interaction behavior across intents.

Extensive experiments on a real world large scale music streaming data from Spotify highlight the benefit of explicitly considering user intent and demonstrate the effectiveness of different interaction signals in predicting user satisfaction in a slate recommendation setting. We contend that our findings provide a valuable framework for fine-grained analysis of user interaction behavior for developing metrics and gauging user satisfaction. To the best of our knowledge, the present study is among the first to identify the benefit of incorporating user intent information in understanding user interactions and satisfaction in recommendation settings, and to demonstrate the significant gains obtained over the popularly used single global intent-agnostic prediction models.

2 RELATED WORK

The current research builds upon four areas: (i) slate recommendations, (ii) user interactions for satisfaction prediction, (iii) user intent modeling and (iv) mixed methods approach to analysis.

Slate Recommendation. Recommendation of slates is a common problem arising in various contexts including search [33], ads and recommendations [8]. Several approaches have been proposed to generate slates, including List Conditional Variational Auto-Encoders [12] and Slate-MDPs [32]. A different approach [33] investigates the evaluation of policies that recommend slates, and then introduces a pseudo-inverse estimator for off-policy evaluation in combinatorial contextual bandits. Different from proposing a new approach to generate slates, our work is concerned with investigating user interaction and gauging user satisfaction with recommended slates.

Satisfaction & User Interaction Signals. With respect to interaction signals, implicit feedback signals (e.g., clicks, dwell time, mouse scrolling) and their sequences have been extensively studied in web search and their effectiveness in predicting satisfaction [9, 11, 24, 26, 27, 29] has been verified, using techniques such as Markov models or deep neural models. A detailed overview regarding the evaluation of interactive information retrieval systems with users can be found in Kelly *et al.* [13].

Scrolls and cursor movements do not exist in mobile search; however, the swipe interaction performs a similar function. Past work leveraging user interaction on mobile phones have used signals such as swipes, dwell times on landing pages and zooms for detecting search result relevance [10] as well as for gauging user attention and satisfaction [25]. User interactions with touch based devices have also been used to detect good abandonment [39] and user satisfaction with intelligent assistants [17]. Building upon such work, we leverage swipe interactions and some signals that result from swipe activity to predict satisfaction.

In the context of recommendation systems, temporal features, including variants of dwell time have been considered for understanding satisfaction [16] and relevance [41]. Specifically focusing on evaluating recommender systems, past work has investigated preference elicitation [19], explanations [34] and user centric combinations of explicit and implicit preferences [18]. Past work has also investigated the influence of personal and situational characteristics towards explaining user experience with recommender systems [20, 22].

Our work also investigates implicit feedback signals, similarly to those works, but in a less explored context, that of music recommendation. Additionally, we aim to investigate the hypothesis on how the interpretation of such interaction signals and their role in predicting satisfaction changes across different user intents.

User Intent Modeling. Related to our consideration of user intents, the role of query intents and search tasks have been extensively studied in the search community, with past work aimed at leveraging intents in predicting query and task satisfaction [27, 37, 42]. Recent work reported in [1] considers the role of intents in recommendations and propose an intent-aware recommendation model. In music recommendation, recent efforts have been made around initial exploration of music listening intents associated with common activities [35], as well as studies of why and when people of different ages and in different countries listen to music [36]. These investigations suggest that intent modeling can potentially improve recommendation quality.

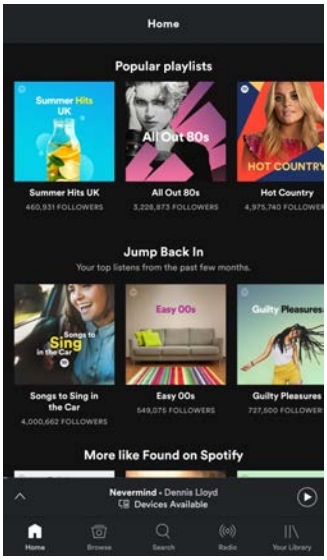


Figure 1: An example homepage of Spotify, a music streaming app.

In web search, tasks have been considered as an analogous abstraction to intents. Prior work has emphasized understanding the role of task level differences while leveraging implicit feedback. Specifically, White *et al.* [38] investigated how individual and task differences impact the effectiveness of algorithms based on implicit relevance feedback. Additionally, Kelly *et al.* [14] considered the effects of task on display time, and quantified the potential impact of this relationship on the effectiveness of display time as implicit feedback.

Mixed methods. Finally, our mixed methods approach of leveraging interviews along with qualitative and quantitative insights follow recent works investigating user expectations, behaviors, and satisfaction in the context of music discovery [5], image [40] and product search [30].

3 PROBLEM FORMULATION

Our goal is to extract and leverage user interaction data to understand and predict user satisfaction in a slate recommendation setting. We consider the Homepage of a major commercial music streaming service, and base our experiments around user interaction with the resulting recommendation page.

3.1 Key concepts

We first define the key concepts used throughout the paper.

- **User Session:** A session is defined as a sequence of user actions with the recommendations rendered, including active browsing of content and passive listening to music. A user session ends when the user has not interacted with or streamed from the Homepage for more than 30 minutes.
- **Slate Recommendation:** A slate is a collection of playlists grouped together under a common theme. Figure 1 shows two slates, *Popular Playlists* and *Jump Back In*, each with 3 playlists. The Slate Recommendation task is to recommend a collection of slates.

Table 1: Description of user interaction signals used.

Signal Type	Signal	Description
Temporal	session length	duration of entire session in seconds
	ms played	total milli-seconds streamed
	dwell time on Homepage	session duration minus downstream time spent streaming content
	avg interaction time	time spent on the Homepage interacting with slates
	time to success	time until first stream
Downstream	songs played	number of songs played
	relationship built	binary signal indicating whether the user saved or downloaded any track or album
	downstream time	total streaming time in any playlist reached via Homepage
Surface Level	no of interactions	total number of clicks on Homepage
	nSlates	no of slates interacted with
	didScroll	binary variable indicating whether or not the user scrolled to view additional recommendations
	max depth	maximum depth reached on the Homepage (number of slates vertically scrolled)
	no of exits	number of exits from Homepage to any playlist
Derivative	avg value of click	total no of clicks / number of stream events
	abandoned	binary feature denoting if the session was abandoned without any interaction
	intentional	binary feature indicating if the user intentionally came to the Homepage from other app feature

- **Homepage:** Also referred to as the home tab, the homepage is the first landing page of the music streaming app, which surfaces a number of playlists, organized as different slates. Figure 1 gives an example of the homepage of Spotify, the music streaming app used in this study.
- **User Satisfaction:** User satisfaction can be viewed as a subjective measure of the utility of recommendations, and we posit that user satisfaction is conditioned not only on the recommendations served, but also on user intent. We rely on implicit signals to derive satisfaction estimates and then use user reported satisfaction scores as gold standard estimate of user satisfaction.

The slate recommendation algorithm aims to find the optimal, ordered subset of items (playlists), a.k.a. slate, given the page layout to serve users recommendations so as to maximize user satisfaction. A Homepage is a collection of such recommended slates, which the user can interact with. Since users come to the app with different goals at different points in time, it becomes important for system designers to understand user interaction to gauge user satisfaction with the slates of recommendations rendered on the app Homepage.

In the rest of this section, we shed light on the different interaction signals we extract from user interaction with the Homepage of Spotify and briefly motivate the need for considering user intent for understanding and predicting user satisfaction in our context.

3.2 Extracting User Interactions Data

The Homepage rendered for a user is rich enough to allow him or her to interact with it in a myriad of ways, including clicking on playlists, scrolling vertically to view more slates, scrolling horizontally to view more playlists in a specific slate, pausing to read and

visually absorb content, clicking and consuming content via streaming, among others. While past work on understanding user interaction in mobile search setting have proposed few signals [17, 39], we additionally propose a number of new signals resulting from the specific *Slate Recommendation* scenario considered in this work.

We use back-end logs of user interactions and extract four different types of interaction signals for each user session:

(1) **Temporal signals**: these focus on the temporal aspects of user interaction, including time spent interacting with the slates, session length, dwell time, etc.
(2) **Surface level signals**: these capture aggregate user interaction on the surface of the Homepage, including total number of clicks, total number of slates the user interacted with, maximum depth the user reached by vertical scrolling, and total number of exits the user had from the Homepage.
(3) **Downstream signals**: these capture downstream user engagement resulting from the Homepage, i.e., user interaction with the playlists, including streaming songs, saving or downloading tracks, viewing Artist profile pages.
(4) **Derivative signals**: these are derived using interaction features like whether the session was abandoned, or whether the session was intentional, with the user going to the Home tab from elsewhere in the app. The average value of click is derived by the considering all clicks in a session and dividing by the total number of streams observed in that session.

Table 1 provides a detailed description of the different interaction signals extracted for each user session.

3.3 Role of User Intent

In the use case of a slate recommendations on a surface like some of the big music streaming apps, users use the Homepage for different needs at different times. With differing needs and intents, users would interact with the app differently, and hence leave different traces of behavioural signals.

Indeed, a user might just want to quickly play some background music, and would not spend much time in finding music to play; on the other hand, a different user might want to carefully find music to save for future listening. In both cases, the way the user interacts with the music app differs, thereby leading to different interaction signals. Since implicit measures of user satisfaction rely heavily on interpreting interaction signals, we hypothesize that understanding and identifying user intent is important to predict user satisfaction in our context, as has been done in other scenarios, particularly in web search [3, 15, 21, 27, 31].

To this end, we discuss different ways of identifying user intents in a recommendation setting in Section 4, and leverage the intents identified along with the extracted interactions signals to predict user satisfaction in Section 5.

4 INTENT IDENTIFICATION

Users use recommendation platforms with different intents. The space of possible intents a user might have is hitherto unknown. In this section, we focus on identifying the different intents users might have when interacting with a music recommendation system.

Table 2: Demographic details of the participants interviewed.

Home Usage	Participant	Gender	Age	Job
Rarely visit Home	Participant 1	Female	31	HR Analyst
	Participant 2	Female	23	Accountant
	Participant 3	Male	20	Student
	Participant 4	Female	27	Construction Manager
	Participant 5	Female	27	Student
	Participant 6	Female	36	Nightlife Manager
Frequently visit Home	Participant 7	Female	28	Student
	Participant 8	Male	31	Finance Manager
	Participant 9	Male	20	Student
	Participant 10	Male	35	Makeup Artist
	Participant 11	Female	32	Attorney
	Participant 12	Male	25	Marketer

4.1 Interviews

To discover and better understand user intents when interacting with the Homepage composed of a set of slate recommendations (playlists), we conducted in-depth 1:1 in-person interviews with users of the streaming app. We describe the interview process and discuss our findings.

Approach. We interviewed 12 users living in the city of New York who were found to have varying degrees of Homepage usage with the Spotify app in the 4 weeks prior to the study. We selected participants from both iOS and Android users with a range of levels of engagement with the app (low, medium, high) and the Homepage (rarely visit, frequently visit), with engagement defined as the number of days they actively used the app in the past month. We emailed a screener survey to a list of Spotify users with these specifications to recruit and schedule the interviews. Table 2 provides a detailed demograpic overview of the participants. With each user, we conducted 1:1 face-to-face interviews spanning 45 minutes each and provided them a $100 gift card upon completion of the interview.

Our goal was to understand and discuss their intents and satisfaction when using the Homepage. We asked them questions around (i) What intents they have when coming to the Homepage; (ii) How satisfied they are with their experience, and (iii) How they behave in relation to intent and satisfaction. The participants were encouraged to provide detailed responses to these questions which were recorded (audio and video).

Findings. We performed a detailed post-hoc analysis of the data from the interviews by going through the video recording of the interviews and the interview transcripts. We grouped together findings that appeared in more than one interview in order to create themes. As per our initial hypothesis, different participants reported to using the Homepage of Spotify for different goals at different times. We were able to categorize user intents into two main themes, Passively Listening, and Actively Listening, and to identify the following overarching intents:

(1) **Theme 1: Passively Listening**
  (a) To find music to play in the background
  (b) To play music that matches their mood or activity
  (c) To quickly access their playlists or saved music

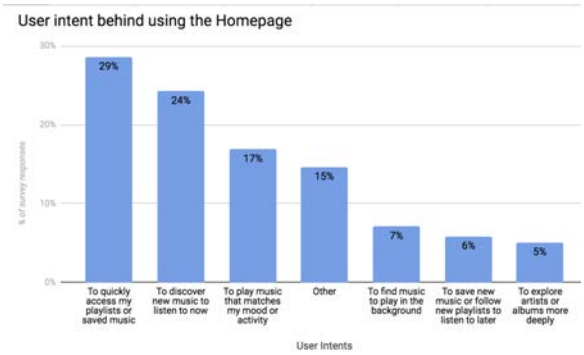


Figure 2: Distribution of user intents.

(2) **Theme 2: Actively Engaging**
   (a) To discover new music to listen to now
   (b) To save new music or follow new playlists to listen to later
   (c) To explore artists or albums more deeply

Additionally, some users pointed out that they use the Homepage because it is the default page that opens upon loading the music app, without having a clear intent when they go on it. We therefore added *Because it is the first screen shown* as an additional intent to the above, thereby making it a total of seven identified intents.

These intents differ from those reported in [36], since they correspond to user intents in using a homepage consisting of slates of recommendations, whereas the intents identified in [36] are user intents for listening to music in general.

Importantly, users often alluded to the fact that their behavior on the Homepage varied with their intent. For example, *scrolling* can be an indicator of exploration, but *scrolling* back and forth can signal a struggle. Not *scrolling* could mean satisfaction with the recommendations presented, but could also indicate dissatisfaction with what was presented. This motivates the need for interpreting interaction signals differently for different intents.

4.2 In-App Survey

Following up on the intents identified from the face-to-face interviews, we performed a large scale survey to validate and understand the prevalence of the intents identified from the interviews. The survey allowed us to collect data from users about their intent and satisfaction during a given session with the Homepage.

Methodology. We selected a random sample of 3 million iPhone users of the Spotify app residing in the US, and presented them with a brief survey at the end of their Homepage session. To not interfere with their natural interaction, and to not bias the user interaction data collected from a session, we trigger the survey only after the user ended their session and moved to another feature of the app (e.g. Search, My Library). To avoid inundating users with the survey repeatedly, the in-app survey is trigged such that each user participates in the survey at most once. Furthermore, we assume that a user has one overarching intent when they begin their Homepage session, an assumption we intend to relax as part of future work.

To gauge user satisfaction we asked the following question: *How satisfied or dissatisfied were you with your experience on the Home screen today?* and presented them with 5 options ranging from *Very satisfied* to *Very Dissatisfied*, along with an additional option of "I wasn't on the Home screen today". Further, to understand their intent behind using the Homepage, we asked them the following question: *Why were you on the Home screen today?* and showed them the six intents identified during the interviews.

As the listed intents may not capture all possible intents users might have, we added *Other* as an option. The *Other* intent option was accompanied by a free text block where users were asked to specify their intent; this allows us to identify any intent not identified during the interviews. Parsing the free flow natural language text and identifying common intents from it is a non-trivial task, which we address in Section 4.3.

Survey Results & Findings. We briefly discuss findings about user intents from the large scale survey, and leave the discussion of findings about user satisfaction for later (Section 6.2). The response rate was 4.5%, with over 116000 users from the 3 million users targeted responding to the survey, which is similar to the response rates we observed in our past surveys.

Figure 3 presents the distribution of intents across user sessions, which indicates how prevalent the intents are. We observe a fairly even distribution of intents across passively listening and actively engaging, which the top two intents: (i) *To quickly access their playlists or saved music* and (ii) *To discover new music to listen to now*, covering 29% and 24% of sessions respectively. A sizeable portion of the sessions involve the user saving music for later consumption or exploring artists in detail, with over 20000 user sessions dedicated to these intents.

Given that the majority of the sessions could be linked to one of the intents identified during interviews, it validates the identified intents. However, since over 15% of users opted for *Other*, indicating that their intent is not covered in the presented list of intents, we further investigate the presence of other intents from the text entered by users. The next section describes a computational approach to analyze the responses to identify new intents.

4.3 Bayesian Non-parametric Model

To ensure that the intents identified sufficiently cover the space of all possible intents users might have when interacting with the Homepage of the Spotify app, we investigate the text entered by the users when they selected the *Other* intent category in the survey responses. Simple clustering on the text is not much useful since the number of possible intents in not known apriori. As a result, we resort to Bayesian nonparametric clustering [7].

The distance dependent Chinese restaurant process (dd-CRP) [2] was recently introduced to model random partitions of data, and is a commonly used method for non-parametric clustering [2, 28]. To extract intents, we consider the dd-CRP model in an embedding-space setting and place a dd-CRP prior over the intents. While a detailed description of non-parametric clustering is beyond the scope of this work, we briefly describe the major steps below. Interested readers are referred to Blei *et al.* [2] who introduce the dd-CRP model, and to Mehrotra *et al.* [28] who apply dd-CRP model to identify query clusters.

Let z_i denote the ith intent assignment, the index of the text with whom the ith text is linked. Let d_{ij} denote the distance measurement between texts i and j, let D denote the set of all distance

measurements between texts, and let f be a decay function. The distance dependent CRP independently draws the text assignments to intent cluster conditioned on the distance measurements,

$$p(z_i = j|D, \alpha) \propto \begin{cases} f(d_{ij}) & \text{if } j \neq i \\ \alpha & \text{if } j = i \end{cases}$$

Here, d_{ij} is an externally specified distance between texts i and j, and α determines the probability that a customer links to themselves rather than another customer. The monotonically decreasing decay function $f(d)$ mediates how the distance between two texts affects their probability of connecting to each other. Cosine similarity between the word2vec embedding representation of two input text describe the distance function (d_{ij}).

Given a decay function f, distances between texts D, scaling parameter α, and an exchangeable Dirichlet distribution with parameter λ, N M-word queries are drawn as follows,

(1) For $i \in [1, N]$, draw $z_i \sim dist - CRP(\alpha, f, D)$.
(2) For $i \in [1, N]$,
   (a) If $z_i \notin R^*_{q_{1:N}}$, set the parameter for the ith text to $\theta_i = \theta_{q_i}$. Otherwise draw the parameter from the base distribution, $\theta_i \sim Dirichlet(\lambda)$.
   (b) Draw the ith terms, $w_i \sim Mult(M, \theta_i)$.

We employ a Gibbs sampler, wherein we iteratively draw from the conditional distribution of each latent variable, given the other latent variables and observations. The Gibbs sampler iteratively draws from

$$p(z_i^{new}|z_{-i}, x) \propto p(z_i^{new}|D, \alpha) \\ p(x|t(z_{-i} \cup z_i^{new}), G_0) \quad (1)$$

The first term is the dd-CRP prior and the second is the likelihood of observations (x) under the partition, and $t(z)$ is the intent-cluster formed from the assignments z. We employ a Dirichlet-Multinomial conjugate distribution to model the likelihood of text terms.

Input texts from users are assigned to intent clusters by considering sets of texts that are reachable from each other through the intent cluster assignments. Notice that many configurations of text assignments might lead to the same intent-cluster assignment. Finally, text assignments can produce a cycle, e.g., text 1 linking with 2 and text 2 linking with 1. This still determines a valid intent-cluster assignment; all texts linked in a cycle are assigned to the same intent cluster.

Upon running the dd-CRP model on the 15% response with user entered text, we obtained 5 different clusters of user responses. We manually investigated these clusters and identified the following intents: (i) To Find X, where X could be any specific playlist a user was trying to find; (ii) To explore or casually browse; (iii) All of the above, wherein the users wanted to select all options presented to them, (iv) I wasn't on Home tab, and (v) Miscellaneous.

Among these five newly identified intents, *I wasn't on Home tab* indicates that the users did not participate in a Homepage session, so we discarded these sessions. Further, Miscellaneous cluster mainly consisted of incoherent text entered by users, thereby rendering it futile. Similarly, we discard sessions tagged with *All of the above* cluster, since it does not present us with new intent and only had few sessions. Finally, the *casually browsing* intent was a very small cluster. In summary, we identified one additional intent: *To Find X* to be added to the pool of the seven intents identified earlier.

Table 3: Final list of user intents identified when interacting with the slated recommended by the Homepage of a music streaming app. Although Intent 1 is not an actual intent, we still refer to it as Intent for ease of writing.

Intent	Definition
Intent 1	Homepage is the first screen shown (i.e. default screen)
Intent 2	To quickly access my playlists or saved music
Intent 3	To discover new music to listen to now
Intent 4	To play music that matches my mood or activity
Intent 5	To Find X
Intent 6	To find music to play in the background
Intent 7	To save new music or follow new playlists to listen to later
Intent 8	To explore artists or albums more deeply.

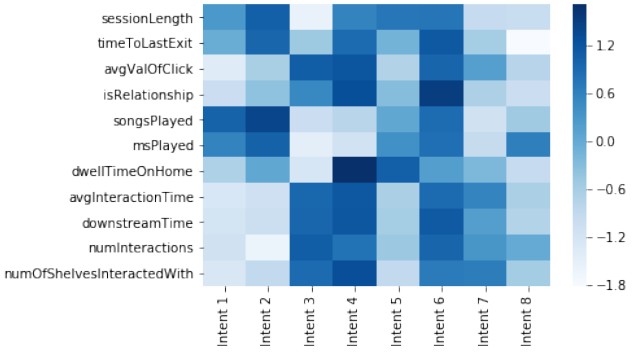


Figure 3: Heatmap of interaction signals across the different user intents. Signals are prevalent to varying extents in different intents, thus highlighting varying user interactions across different intents.

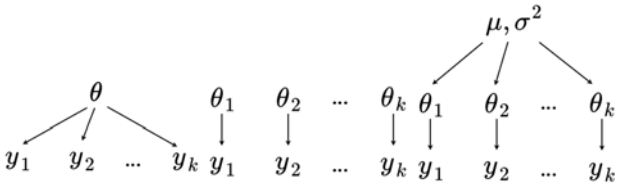
Users often use the Homepage to find something they're looking for, which could be any specific playlist or any specific shelf. The extracted additional intent caters to such search-like use-cases.

Although the rigorous process of non-parametric clustering applied to user entered responses only gives us one new intent, it reaffirms the exhaustiveness and coverage of the six intents identified before.

4.4 Analysis of User Interactions & Intents

Based on the interviews, in-app survey and non-parametric clustering of textual user responses, we end up with eight user intents, as described in Table 3. The diversity of these intents highlights the variety in what users intend to achieve when using the Homepage of the music app.

We hypothesize that the way users interact with the recommended slates of playlists would differ across these intents. Figure 3 presents a heatmap of a subset of interaction signals (see Table 1) with the different intents. We observe that the prominence of interaction signals significantly differs across intents, with certain signals like interaction time on the Homepage significantly lower for intent 2 (to quickly access music). These differences in interaction signals highlight the fact that users indeed behave differently when having different intents. Since implicit measures of satisfaction highly depend on interaction signals, this places



(a) Global Model **(b)** Per-Intent Model **(c)** Multi-level Model

Figure 4: Different modeling assumptions for intent aware SAT prediction model: (a) Global Model for all data across all intents; (b) Per-Intent Model; and (c) Hierarchical Multi-level Model with shared parameters.

special emphasis of explicitly considering intents when predicting satisfaction.

5 PREDICTING SATISFACTION

Our main goal is to understand and predict user satisfaction (SAT) using interaction data. To this end, we extracted detailed user interaction signals, and identified different intents users might have. In this section, we leverage the extracted signals and intents and present techniques to predict user satisfaction using the signals. We present three approaches for satisfaction prediction, covering the spectrum of intent-level granularity, i.e. global model for all intents to a separate model for each intent and a shared model across intents.

5.1 Global Model

We begin by describing the most conventional approach to satisfaction prediction, referred to as *Global Model*. The simplest technique would be to treat all user sessions as a homogeneous collection of data, with the user intent featuring simply as a categorical variable along with user interaction signals. Almost all existing work on predicting user satisfaction from interaction signals [9, 11, 26, 27] have employed such an approach to satisfaction (SAT) prediction.

For the Global prediction model, we perform traditional Logistic Regression and Gradient Boosted Decision Trees classifiers. Given a dataset with n user sessions ($x_i \in X$), each of which is represented by m features, $D = \{(x_i, y_i)\}$ ($|D| = n, x_i \in R^m, y_i \in R$), the logistic regression classifier assigns a probability to each outcome as

$$P(y_i = c|x_i) = logit^{-1}(\theta^T x_i + b) \tag{2}$$

where $c \in \{-1, 1\}$ and is trained with conditional maximum likelihood estimation, wherein we choose θ that maximize the (log) probability of the labels given the observations x_i. Thus, the objective function we aim at maximizing is:

$$L(w) = -\Sigma_{i=1}^{n} \sigma(\theta^T x_i + b)^{y_n} (1 - \sigma(\theta^T x_i + b)^{(1-y_n)} \tag{3}$$

where σ is the sigmoid function. Alternatively, tree boosting type of models (e.g. Gradient Boosted Regression Trees) optimize a tree ensemble model using additive functions to predict the output. We train both Logistic Regression and GBDT models using all the interaction features extracted for each session, along with a categorical variable describing user intent for the current session. As shown in Figure 4 (a), the same parameter is learned and shared across all intents and sessions.

5.2 Per-Intent Model

A key issue with a Global Model is that it ignores the variations in the interaction signals across different intents. Indeed, a user with an intent to quickly access her playlist to play music (Intent 2) would know what she is looking for, and not spend time in detailed exploration, unlike a user interested in exploring an artist more deeply (Intent 8). Whereas the categorical variable for the session intent would slightly help the model account for intent specific traits, a better approach would be to train a separate model for each intent, referred as *Per-Intent Model* illustrated in Figure 4 (b).

Let η_i denote the intent for session i, and $K = 7$ be the total number of intents, i.e. $\eta_i \in [1, K]$. We divide the session data ($D = \{(x_i, y_i)\}$) into K different intent specific groups and train K different SAT prediction models, one for each intent:

$$P(y_i = c|x_i) = logit^{-1}(\theta_k^T x_i + b_k) \tag{4}$$

with (θ_k, b_k) representing the parameters of the k-th intent SAT model.

While the Per-Intent model is able to capture intent specific intricacies of the different interaction signals, it suffers from a few drawbacks. It prohibitively collect data for each intent separately, and there may not be enough data to learn a meaningful model for each intent separately. This problem gets compounded when platform changes encourages users to use the platform for new intents. Furthermore, an intent specific model fails to benefit from shared learning of parameters and hence fails to leverage insights from the interplay between interaction signals and satisfaction data of other intents. We next present a hierarchical multi-level approach to address these shortcomings.

5.3 Multi-level Model

Both the Global and Per-Intent SAT prediction models introduced suffer from various disadvantages. The Per-Intent SAT prediction model works by using just the local intent specific information, and assumes that the data is sampled from separate models for each intent, thereby ignoring information and insights from other intents. Furthermore, it can be rendered futile for intents with small labeled data. At the other extreme, the Global SAT prediction model ignores intent-level variations in the user interaction data and inadvertently suppresses variations that can be important.

To address these issues, we resort to *Multi-level Modeling* [6, 23], which represents a compromise between the two extremes of a single global prediction model and a separate prediction model for each intent. Multi-level models, illustrated in Figure 4 (c), are regression and classification models in which the constituent model parameters are given probability models. This implies that the model parameters are allowed to vary by group. This approach can be used to handle clustered or grouped data (e.g. different user intents), wherein each level is (potentially) a source of unexplained variability. Multi-level modeling provides a coherent model that simultaneously incorporates both individual and group level models.

Benefits of Multi-level Modeling

Modeling the satisfaction prediction model as a multi-level model offers numerous benefits. First, leveraging multilevel models allows us to account for the intent level grouping of the user session.

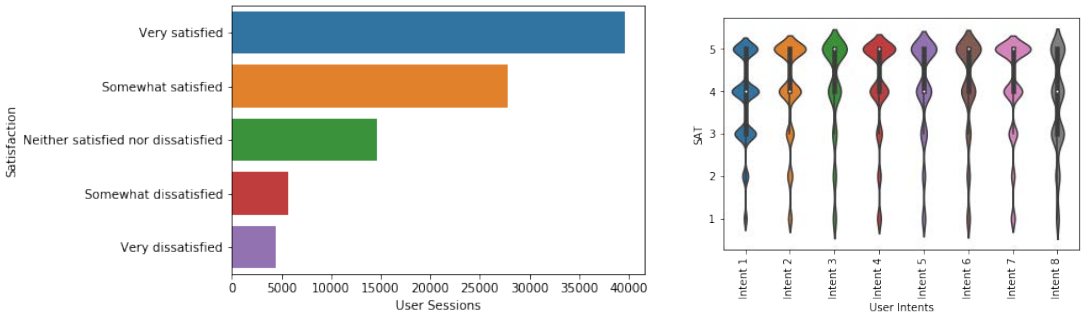


Figure 5: Analysis of user survey responses. Left: Distribution of user satisfaction labels obtained. Right: Distribution of SAT labels across the different user intents. SAT levels 1 to 5 refer to different levels from Very Dissatisfied (1) to Very satisfied (5).

Second, they facilitate incorporating both individual session level as well as intent group level effects on user satisfaction, thereby allowing for variability in user interaction behavior across different intents. Third, by assuming that the intent group level effects come from a common distribution shared across all intents, they facilitate information sharing across different intents. This can help in improving the accuracy and predictive performance for intents with relatively little data.

Intent-aware Multi-level Model

We model the satisfaction prediction problem with a multi-level logistic regression model with the different user intents serving as groups for the hierarchical model. The Multi-level Model can be written as:

$$P(y_i = c|x_i) = logit^{-1}(\theta_{\eta_i}^T x_i + b_{\eta_i}) \tag{5}$$

with η_i denoting the intent group for the i-th user session, and x_i denoting the user interaction signals extracted for the session. In our hierarchical model, parameters are viewed as a sample from a population distribution of parameters.

$$\begin{pmatrix}\theta_\eta \\ b_\eta\end{pmatrix} = \begin{pmatrix}\mu_0 \\ \mu_1\end{pmatrix} + \begin{pmatrix}\gamma_{0,\eta}^{int} \\ \gamma_{1,\eta}^{int}\end{pmatrix} \tag{6}$$

$$\begin{pmatrix}\gamma_{0,k}^{int} \\ \gamma_{1,k}^{int}\end{pmatrix} \sim N\left(\begin{pmatrix}0 \\ 0\end{pmatrix}, \Sigma^{int}\right) for\ k = 1, \ldots, K \tag{7}$$

Thus, we view them as being neither entirely different (as was the case in the Per-Intent model) or exactly the same (as was the case in the Global Model). Figure 4 pictorially depicts the main difference between the parameter sharing across the three different satisfaction prediction models investigated in this work.

6 EXPERIMENTAL EVALUATION

We perform an evaluation to compare the different satisfaction prediction models, consider the role of different interaction features and present results on a large scale real world dataset.

6.1 Dataset

We work with real world user data from Spotify, a large music streaming service, and conduct a large scale in-app survey to collect judgments about intents and user satisfaction. We trigger the in-app survey to a random sample of over 3 million users, and observe a response rate of 4.5%. In total, we received responses from over 116000 users, resulting in over 200K judgments about intents and satisfaction combined. For each session, we collected back-end logs of user interactions with the slates of recommendation and extracted data for all the different interaction signals mentioned in Table 1. For each session, we use the intent selected by the user as the session intent and use the satisfaction prediction label given by the user to train and evaluate our satisfaction prediction models.

Owing to the *response bias*, i.e., the respondents of the survey are likely to be users with a positive bias toward the system, we would end up with a biased dataset for training, which would lead to unreliable prediction estimates. We mitogate this by oversampling the minority class by synthetic minority over-sampling technique [4]. This results in a balanced dataset with a healthy distribution of labels across both satisfaction and dissatisfaction cases.

6.2 Analysis of Survey Results

We begin with an analysis of the survey responses by understanding the degree to which users are satisfied overall, and across different intents. Figure 5 (left) presents the overall distribution of user satisfaction. We observe that most users are either *Very satisfied* or *Somewhat satisfied* with the slates recommended to them in their current session with over 33% user sessions being judged as very satisfying. On the other extreme, over 12000 user sessions were reported to be very or somewhat dissatisfying. A relatively larger number of user sessions were judged as neutral.

To gauge satisfaction for each intent, we plot the satisfaction label distribution separately for each intent in Figure 5 (right). Similar to overall satisfaction, we observe that more user sessions are satisfying than they are dissatisfying. Users with intents 3, 6 and 7 were more satisfied with the slate recommendations shown to them, than users pursuing other goals, which highlights the efficacy of the system in facilitating users to discover new music for now (Intent 3), playing music in background (Intent 6) and saving music for future (Intent 7). However, the distribution of the different levels of satisfaction differs across intents. As can be seen in the violin plot, a substantial number of user sessions were judged as neutral on satisfaction scale for Intent 8, whereas the distribution is skewed more towards the top-most level of satisfaction for Intents 3 and 6.

Considering such differences in distribution of satisfaction across intents helps system designers in identifying those intents where recommendation system under-performs, an insight which instead would have been hidden when considering overall user satisfaction.

Table 4: Comparing satisfaction prediction performances across different prediction models: Global, Per-Intent and Multi-level. * indicates statistical significant ($p \leq 0.05$) using paired t-tests compared to the Global + intent Model.

		Overall			Dissatisfaction			Satisfaction		
Method	**Accuracy**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Global w/o intent	0.55	0.56	0.56	0.56	0.57	0.48	0.53	0.54	0.63	0.59
Global w/ intent	0.57	0.58	0.58	0.57	0.59	0.52	0.55	0.56	0.63	0.60
Intent 1	0.675	0.68	0.68	0.67	0.66	0.73	0.69	0.70	0.62	0.65
Intent 2	0.663	0.67	0.67	0.67	0.66	0.71	0.68	0.68	0.62	0.65
Intent 3	0.672	0.67	0.67	0.67	0.69	0.67	0.68	0.65	0.68	0.67
Intent 4	0.678	0.68	0.68	0.68	0.67	0.71	0.69	0.69	0.64	0.66
Intent 5	**0.814***	**0.81***	0.81	0.81	0.80	0.84	**0.82***	0.83	0.79	0.81
Intent 6	0.760	0.77	0.76	0.76	0.73	0.83	0.77	0.81	0.69	0.75
Intent 7	0.7771	0.77	0.77	0.77	0.75	0.81	0.78	0.79	0.74	0.76
Intent 8	0.769	0.78	0.77	0.77	0.71	**0.87***	0.78	0.85	0.67	0.75
Multi-Level	0.804	0.80	0.79	0.79	0.81	0.79	0.79	0.79	**0.81***	**0.81***

This further motivates the need for incorporating intent information when developing and interpreting user satisfaction metrics.

6.3 Predicting User Satisfaction

We next investigate the extent to which the proposed interaction signals and the predictive models are able to predict user satisfaction. We consider the satisfaction label provided by the user during an in-app survey as ground truth label, and consider a user to be satisfied if their response was either *Very Satisfied* or *Somewhat Satisfied*. In all other cases, the user session was labeled as dissatisfied. Such binarization of user satisfaction isn't ideal, but we contend that it is a good starting point and has been extensively used in prior industrial research on understanding and predicting user satisfaction [17, 27, 39].

In line with prior work on predicting binary user satisfaction, we use four different evaluation metrics: accuracy, precision, recall and F-score. Table 4 presents detailed results comparing the Global, Per-Intent and Multi-level Model on these four metrics. We additionally divide the overall prediction performances and separately consider the performance in predicting satisfaction and dissatisfaction, since understanding them (SAT and DSAT) independently is useful in different use-case scenarios and at different stages of system development.

We observe that the Global Model is not able to predict user satisfaction much better than random, giving an overall accuracy of 57%. The Global Model without intent as a feature (i.e. Global w/o intent) performs worse, which highlights that incorporating intent information is useful in predicting satisfaction. The Per-Intent Model, on the other hand, gives much better prediction results than the Global Model. We observe an increased performance across all eight intents, with performance gains ranging from 10% to 24% for different intents. This confirms the main hypothesis of our work – considering intent information is crucial in accurately understanding and predicting user satisfaction. Since the interaction signals differ across different intents, the Global Model is unable to find a good universal relation between such signals and satisfaction. The Per-Intent Model creates a separate local intent-specific model and is able to capture the interplay between interaction signals and satisfaction to a much better extent.

We observe performance improvements in not just the accuracy, but across all metrics considered for both predicting satisfaction and predicting user dissatisfaction. Furthermore, we observe that the Multi-level Model performs significantly better than the Global Model (with intent) with over 20% improvement in prediction accuracy over the Global Model. Additionally, the Multi-level Model outperforms all but one Per-Intent Model, with performance improvement ranging from 4-14% in terms of prediction accuracy across different intents, while giving comparable performance to the last intent. This shows that the Multi-level Model is able to leverage insights from other intents, which in turn help boost the performance of other intents.

Additionally, we also observe that some intents are easier to predict than others, with intent 1 being the hardest to predict satisfaction for. Indeed, intent 1 sessions are of users who do not particularly have any intent in mind, and had a homepage session because it was the first page shown to them, so they could have any intent while interacting. This makes it harder for the system to predict satisfaction. It is important to note that an intent-specific model works best for certain intents (e.g. Intents 5 & 8) which suggests that the intent specific interaction data is powerful enough in itself, and doesn't gain much from shared learnign across intents. However, the shared learning across intents via the multi-level model is more generally beneficial to most intents and gives improved performance for 7 out of 8 intents. Moreover, maintaining separate intent level models is a demanding engineering task, while the shared multi-level model provides a sweet middle-ground whcih is relatively easier to manage and support from a large scale engineering perspective.

6.4 Importance of Interaction Features

We next investigate the different types of interaction signals considered and analyse their importance in prediction satisfaction. Table 5 presents accuracy results for different groups of interaction signals considered, for all three models. We omit displaying results for the Global - intent Model due to space constraints. We bold the best

Table 5: Comparing the importance of different interaction signals using different satisfaction prediction models using prediction accuracy. * and & indicates statistical significant ($p \leq 0.05$) using paired t-tests compared to Downstream and Temporal baselines respectively.

Interaction Features	Global	Intent 1	Intent 2	Intent 3	Intent 4	Intent 5	Intent 6	Intent 7	Intent 8	Multilevel
Downstreams	0.528	0.556	0.568	0.555	0.525	0.599	0.593	0.641	0.551	0.605
Surface	0.522	0.548	0.546	0.512	0.572	0.550	0.556	0.544	0.595	0.576
Temporal	0.554	0.655	0.639	0.641	0.515	0.756	0.732	0.735	0.754	0.746
Metrics	0.541	0.602	0.593	0.578	0.663	0.711	0.662	0.639	0.712	0.723
Downstreams + Surface	0.523	0.564	0.565	0.560	0.602	0.582	0.586	0.597	0.561	0.598
Downstreams + Temporal	0.560	0.672	0.655	0.649	0.579	0.772	0.746	0.747	0.752	0.781
Temporal + Metrics	0.569	0.660	0.638	0.650	0.668	0.792	0.718	0.725	0.743	0.795
Downstream + Surface + Temporal + Metrics	**0.571**$^{*\&}$	**0.675**$^{*\&}$	**0.666**$^{*\&}$	**0.672**$^{*\&}$	**0.675**$^{*\&}$	**0.811**$^{*\&}$	**0.760**$^{*\&}$	**0.771**$^{*\&}$	**0.769**$^{*\&}$	**0.804**$^{*\&}$

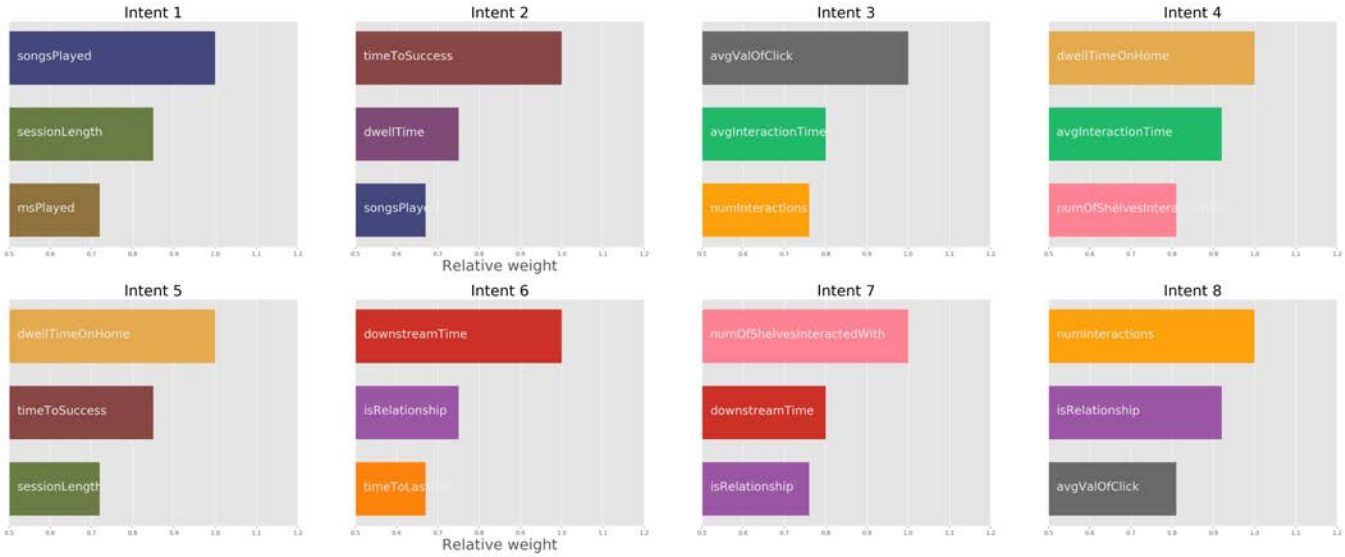


Figure 6: Relative weights of the top 3 predictive features across all eight intents.

performing features for each prediction model. We observe that whereas temporal signals are more important than other types of signals for most intents, surface signals such as number of slates interacted with and depth are important for Intent 4 (play music matching mood). This highlights that different types of signals are important for different intents, which supports the analysis presented in Figure 3 that showed interaction signals to differ based on user intent. Additionally, we observe that the Downstream group of signals performs better for Intent 7 (save new music or follow playlist), than it does for other intents, with 6-10% better predictive performance. Indeed, downstream signals help capture user activity related to saving music and exploring artists in detail, which is what Intent 7 is focusing on.

Finally, we observe a sharp increase in predictive performance when combining downstream and temporal signals, with over 10% performance jumps when compared to any individual signal group. We also observe that even in the case of different signal groups, Per-Intent and Multi-level models performs significantly better than the Global Model. Finally, combining different signals helps in improving satisfaction prediction accuracy by over 10%-15% for most intents.

Importance of signals across intents. To gain insights into which interaction signal is most useful across different intents, we present the top three signals and report their relative feature weight across all eight intents in Figure 6. As expected, we observe that the signals are important to varying degree across different intents. For the case where the user wants to quickly access saved music, signals like time to success and dwell time are most informative, while they are not informative for cases where the user wants to play music in the background. For intents where users wish to explore artists in more detail, signals involving the users building relationships (i.e. saved or downloaded tracks) are shown to be more important.

The variation of the most informative signals across the different intents highlight the fact that different signals are indeed important to different extent for different use cases. These results clearly emphasize the need for modeling intent while interpreting user interactions for predicting user satisfaction.

7 CONCLUSIONS

Given the *query-less paradigm* of slate recommendation, it becomes non-trivial to understand user intents. Based on a mixed-methods approach composed of interviews, in-app survey and non-parametric clustering, we identified eight key user intents, and experimentally demonstrated the importance of explicitly considering these intents when predicting user satisfaction. Our results also indicate that different interaction signals are important to varying extent across intents. Furthermore, the significant improvement in prediction results advocate not only the need for grouping user

sessions into intent groups for predicting satisfaction, but also for shared learning across all intents via the Multi-level Model. We contend that the methodology adopted in this work to identify user intents: interviews, in-app survey and quantitative modeling would provide firm starting grounds for researchers to think about user intents in other recommendation scenarios where it is prohibitively challenging to extract unspecified user intents.

Limitations & Future Work. While our findings are more generally applicable beyond just music streaming domain, a few interesting challenges remain, which serve as useful directions for future work. While we leveraged the intent labels provided by survey responders, these labels are not readily available at run time for user sessions. This motivates the need for developing an intent prediction module to predict the intent for each user session. The current study also operates under the assumption of each session having one intent, which may not always be correct, since user intents might change within a session. Development of intent boundary detection or intent switching prediction modules would help in eliminating this assumption. Additionally, the model would need to cater to intents changing over time, and across contexts. Finally, we did not incorporate user specific intricacies in our formulation, but we believe that user interaction is conditional on user groups, and differ user segments interact differently. Incorporating such user level idiosyncrasies would be a fruitful direction.

REFERENCES

[1] Biswarup Bhattacharya, Iftikhar Burhanuddin, Abhilasha Sancheti, and Kushal Satya. 2017. Intent-Aware Contextual Recommendation System. In *Data Mining Workshops (ICDMW), 2017 IEEE International Conference on*. IEEE, 1–8.

[2] David M Blei and Peter I Frazier. 2011. Distance dependent Chinese restaurant processes. *The Journal of Machine Learning Research* 12 (2011), 2461–2488.

[3] Olivier Chapelle, Shihao Ji, Ciya Liao, Emre Velipasaoglu, Larry Lai, and Su-Lin Wu. 2011. Intent-based diversification of web search results: metrics and algorithms. *Information Retrieval* 14, 6 (2011), 572–592.

[4] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.

[5] Jean Garcia-Gathright, Brian St Thomas, Christine Hosey, Zahra Nazari, and Fernando Diaz. 2018. Understanding and Evaluating User Satisfaction with Music Discovery. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 55–64.

[6] Andrew Gelman and Jennifer Hill. 2006. *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press.

[7] Samuel J Gershman and David M Blei. 2012. A tutorial on Bayesian nonparametric models. *Journal of Mathematical Psychology* 56, 1 (2012), 1–12.

[8] Carlos A Gomez-Uribe and Neil Hunt. 2016. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)* 6, 4 (2016), 13.

[9] Qi Guo and Eugene Agichtein. 2012. Beyond dwell time: estimating document relevance from cursor movements and other post-click searcher behavior. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 569–578.

[10] Qi Guo, Haojian Jin, Dmitry Lagun, Shuai Yuan, and Eugene Agichtein. 2013. Mining touch interaction data on mobile devices to predict web search result relevance. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 153–162.

[11] Jeff Huang, Ryen W White, Georg Buscher, and Kuansan Wang. 2012. Improving searcher models using mouse cursor activity. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 195–204.

[12] Ray Jiang, Sven Gowal, Timothy A Mann, and Danilo J Rezende. 2018. Optimizing Slate Recommendations via Slate-CVAE. *arXiv preprint arXiv:1803.01682* (2018).

[13] Diane Kelly et al. 2009. Methods for evaluating interactive information retrieval systems with users. *Foundations and Trends® in Information Retrieval* 3, 1–2 (2009), 1–224.

[14] Diane Kelly and Nicholas J Belkin. 2004. Display time as implicit feedback: understanding task effects. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 377–384.

[15] Eugene Kharitonov, Craig Macdonald, Pavel Serdyukov, and Iadh Ounis. 2013. Intent models for contextualising and diversifying query suggestions. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2303–2308.

[16] Youngho Kim, Ahmed Hassan, Ryen W White, and Imed Zitouni. 2014. Comparing client and server dwell time estimates for click-level satisfaction prediction. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 895–898.

[17] Julia Kiseleva, Kyle Williams, Ahmed Hassan Awadallah, Aidan C Crook, Imed Zitouni, and Tasos Anastasakos. 2016. Predicting user satisfaction with intelligent assistants. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 45–54.

[18] Bart P Knijnenburg, Niels JM Reijmer, and Martijn C Willemsen. 2011. Each to his own: how different users call for different interaction methods in recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 141–148.

[19] Bart P Knijnenburg and Martijn C Willemsen. 2009. Understanding the effect of adaptive preference elicitation methods on user satisfaction of a recommender system. In *Proceedings of the third ACM conference on Recommender systems*. ACM, 381–384.

[20] Bart P Knijnenburg, Martijn C Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. 2012. Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction* 22, 4-5 (2012), 441–504.

[21] Christoph Kofler, Martha Larson, and Alan Hanjalic. 2016. User intent in multimedia search: a survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR)* 49, 2 (2016), 36.

[22] Joseph A Konstan and John Riedl. 2012. Recommender systems: from algorithms to user experience. *User modeling and user-adapted interaction* 22, 1-2 (2012), 101–123.

[23] Jennifer L Krull and David P MacKinnon. 2001. Multilevel modeling of individual and group level mediated effects. *Multivariate behavioral research* 36, 2 (2001), 249–277.

[24] Dmitry Lagun, Mikhail Ageev, Qi Guo, and Eugene Agichtein. 2014. Discovering common motifs in cursor movement data for improving web search. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 183–192.

[25] Dmitry Lagun, Chih-Hung Hsieh, Dale Webster, and Vidhya Navalpakkam. 2014. Towards better measurement of attention and satisfaction in mobile search. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 113–122.

[26] Yiqun Liu, Ye Chen, Jinhui Tang, Jiashen Sun, Min Zhang, Shaoping Ma, and Xuan Zhu. 2015. Different users, different opinions: Predicting search satisfaction with mouse movement information. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 493–502.

[27] Rishabh Mehrotra, Ahmed Hassan Awadallah, Milad Shokouhi, Emine Yilmaz, Imed Zitouni, Ahmed El Kholy, and Madian Khabsa. 2017. Deep Sequential Models for Task Satisfaction Prediction. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 737–746.

[28] Rishabh Mehrotra, Prasanta Bhattacharya, and Emine Yilmaz. 2016. Deconstructing complex search tasks: a bayesian nonparametric approach for extracting sub-tasks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 599–605.

[29] Rishabh Mehrotra, Imed Zitouni, Ahmed Hassan Awadallah, Ahmed El Kholy, and Madian Khabsa. 2017. User Interaction Sequences for Search Satisfaction Prediction. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 165–174.

[30] Ning Su, Jiyin He, Yiqun Liu, Min Zhang, and Shaoping Ma. 2018. User Intent, Behaviour, and Perceived Satisfaction in Product Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 547–555. https://doi.org/10.1145/3159652.3159714

[31] Ning Su, Jiyin He, Yiqun Liu, Min Zhang, and Shaoping Ma. 2018. User Intent, Behaviour, and Perceived Satisfaction in Product Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 547–555.

[32] Peter Sunehag, Richard Evans, Gabriel Dulac-Arnold, Yori Zwols, Daniel Visentin, and Ben Coppin. 2015. Deep Reinforcement Learning with Attention for Slate Markov Decision Processes with High-Dimensional States and Actions. *arXiv preprint arXiv:1512.01124* (2015).

[33] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miro Dudik, John Langford, Damien Jose, and Imed Zitouni. 2017. Off-policy evaluation for slate recommendation. In *Advances in Neural Information Processing Systems*. 3632–3642.

[34] Nava Tintarev and Judith Masthoff. 2007. Effective explanations of recommendations: user-centered design. In *Proceedings of the 2007 ACM conference on*

Recommender systems. ACM, 153–156.

[35] Sergey Volokhin and Eugene Agichtein. 2018. Towards Intent-Aware Contextual Music Recommendation: Initial Experiments. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 1045–1048.

[36] Sergey Volokhin and Eugene Agichtein. 2018. Understanding Music Listening Intents During Daily Activities with Implications for Contextual Music Recommendation. In *Proceedings of the 2018 Conference on Human Information Interaction&Retrieval*. ACM, 313–316.

[37] Ryen W White, Wei Chu, Ahmed Hassan, Xiaodong He, Yang Song, and Hongning Wang. 2013. Enhancing personalized search by mining and modeling task behavior. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1411–1420.

[38] Ryen W White and Diane Kelly. 2006. A study on the effects of personalization and task information on implicit feedback performance. In *Proceedings of the 15th ACM international conference on Information and knowledge management*. ACM, 297–306.

[39] Kyle Williams, Julia Kiseleva, Aidan C Crook, Imed Zitouni, Ahmed Hassan Awadallah, and Madian Khabsa. 2016. Detecting good abandonment in mobile search. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 495–505.

[40] Xiaohui Xie, Yiqun Liu, Maarten de Rijke, Jiyin He, Min Zhang, and Shaoping Ma. 2018. Why People Search for Images Using Web Search Engines. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 655–663. https://doi.org/10.1145/3159652.3159686

[41] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond clicks: dwell time for personalization. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 113–120.

[42] Yuchen Zhang, Weizhu Chen, Dong Wang, and Qiang Yang. 2011. User-click modeling for understanding and predicting search-behavior. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1388–1396.

SaGe: Web Preemption for Public SPARQL Query Services

Thomas Minier
LS2N, University of Nantes
Nantes, France
thomas.minier@univ-nantes.fr

Hala Skaf-Molli
LS2N, University of Nantes
Nantes, France
hala.skaf@univ-nantes.fr

Pascal Molli
LS2N, University of Nantes
Nantes, France
pascal.molli@univ-nantes.fr

ABSTRACT

To provide stable and responsive public SPARQL query services, data providers enforce quotas on server usage. Queries which exceed these quotas are interrupted and deliver partial results. Such interruption is not an issue if it is possible to resume queries execution afterward. Unfortunately, there is no preemption model for the Web that allows for suspending and resuming SPARQL queries. In this paper, we propose SaGe: a SPARQL query engine based on Web preemption. SaGe allows SPARQL queries to be suspended by the Web server after a fixed time quantum and resumed upon client request. Web preemption is tractable only if its cost in time is negligible compared to the time quantum. The challenge is to support the full SPARQL query language while keeping the cost of preemption negligible. Experimental results demonstrate that SaGe outperforms existing SPARQL query processing approaches by several orders of magnitude in term of the average total query execution time and the time for first results.

KEYWORDS

ACM Reference Format:
Thomas Minier, Hala Skaf-Molli, and Pascal Molli. 2019. SaGe: Web Preemption for Public SPARQL Query Services. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313652

1 INTRODUCTION

Context and motivation: Following the Linked Open Data principles (LOD), data providers published billions of RDF triples [5, 24]. However, providing a public service that allows anyone to execute any SPARQL query at any time is still an open issue. As public SPARQL query services are exposed to an unpredictable load of arbitrary SPARQL queries, the challenge is to ensure that the service remains *available* despite variation in terms of the arrival rate of queries and *resources* required to process queries.

To overcome this problem, most public LOD providers enforce a fair use service policy based on *quotas* [3]. According to DBpedia administrators:"A Fair Use Policy is in place in order to provide a stable and responsive endpoint for the community."[1] The public DBpedia SPARQL endpoint [2] Fair Use Policy prevents the execution of SPARQL longer than 120 seconds or that return more than 10000 results, with a limit of 50 concurrent connections and 100 requests per second per IP address. Quotas aim to share fairly server resources among Web clients. Quotas on communications limit the arrival rate of queries per IP. Quotas on space prevent one query to consume all the memory of the server. Quotas on time aim to avoid the *convoy phenomenon* [6], *i.e.*, a long-running query will slow down a short-running one, in analogy with a truck on a single-lane road that creates a convoy of cars. The main drawback of quotas is that *interrupted queries can only deliver partial results, as they cannot be resumed.* This is a serious limitation for Linked Data consumers, that want to execute long-running queries [22].

Related works: Existing approaches address this issue by decomposing SPARQL queries into subqueries that can be executed under the quotas and produce complete results [4]. Finding such decomposition is hard in the general case, as quotas can be different from one server to another, both in terms of values and nature [4]. The Linked Data Fragments (LDF) approach [17, 27] tackles this issue by restricting the SPARQL operators supported by the server. For example, in the Triple Pattern Fragments (TPF) approach [27], a TPF server only evaluates triple patterns. However, LDF approaches generate a large number of subqueries and substantial data transfer.

Approach and Contributions: We believe that the issue related to time quotas is not interrupting a query, but the impossibility for the client to *resume* the query execution afterwards. In this paper, we propose SaGe, a SPARQL query engine based on Web preemption. Web preemption is the capacity of a Web server to suspend a running query after a time quantum with the intention to resume it later. When suspended, the state S_i of the query is returned to the Web client. Then, the client can resume query execution by sending S_i back to the Web server.

Web preemption adds an overhead for the Web server to suspend the running query and resume the next waiting query. Consequently, the main scientific challenge here is to keep this overhead marginal whatever the running queries, to ensure good query execution performance. The contributions of this paper are as follows:

- We define and formalize a *Web preemption* model that allows to suspend and resume SPARQL queries.
- We define a set of preemptable query operators for which we bound the complexity of suspending and resuming of these operation, both in time and space. This allows to build a preemptive Web server that supports a large fragment of the SPARQL query language.
- We propose SaGe, a SPARQL query engine, composed of a preemptive Web server and a smart Web client that allows executing full SPARQL queries[3].

[1] http://wiki.dbpedia.org/public-sparql-endpoint
[2] http://dbpedia.org/sparql

[3] The SaGe software and a demonstration are available at http://sage.univ-nantes.fr

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313652

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?actor ?name ?birthPlace WHERE {
  ?actor a dbo:Actor; rdfs:label ?name; dbo:birthPlace ?city.
  ?city a dbo:City; rdfs:label ?birthPlace.
}
```

Figure 1: SPARQL Query Q_1: finds all actors' birth cities.

- We compare the performance of the SaGe engine with existing approaches used for hosting public SPARQL services. Experimental results demonstrate that SaGe outperforms existing approaches by several orders of magnitude in term of the average total query execution time and the time for first results.

This paper is organized as follows. Section 2 summarizes related works. Section 3 defines the Web preemption execution model and details the SaGe server and the SaGe client. Section 4 presents our experimental results. Finally, conclusions and future work are outlined in Section 5.

2 RELATED WORKS

SPARQL endpoints. SPARQL endpoints follow the SPARQL protocol [4], which "describes a means for conveying SPARQL queries and updates to a SPARQL processing service and returning the results via HTTP to the entity that requested them". Without quotas, SPARQL endpoints execute queries using a First-Come First-Served (FCFS) execution policy [10]. Thus, by design, they can suffer from *convoy effect* [6]: one long-running query occupies the server resources and prevents other queries from executing, leading to long waiting time and degraded average completion time for queries.

To prevent convoy effect and ensure a fair sharing of resources among end-users, most SPARQL endpoints configure quotas on their servers. They mainly restrict the arrival rate per IP address and limit the execution time of queries. Restricting the arrival rate allows end-users to retry later, however, limiting the execution time leads some queries to deliver only partial results. To illustrate, consider the SPARQL query Q_1 of Figure 1. Without any quota, the total number of results of Q_1 is 35 215, however, when executed against the DBpedia SPARQL endpoint, we found only 10 000 results out of 35 215 [5].

Delivering partial results is a serious limitation for a public SPARQL service. In SaGe, we deliver complete results whatever the query. In some way, quotas interrupt queries without giving the possibility to resume their execution. SaGe also interrupts queries, but allows data consumers to resume their execution later on.

Decomposing queries and restricting server interfaces. Evaluation strategies [4] have been studied for federated SPARQL queries evaluated under quotas. Queries are decomposed into a set of subqueries that can be fully executed under quotas. The main drawbacks of these strategies are: (i) They need to know which quotas are configured. Knowing all quotas that a data provider can implement is not always possible. (ii) They can only be applied to a specific class of SPARQL queries, *strongly bounded SPARQL queries*, to ensure complete and correct evaluation results.

The Linked Data Fragments (LDF) [17, 27] restrict the server interface to a fragment of the SPARQL algebra, to reduce the complexity of queries evaluated by the server. LDF servers are no more compliant with the W3C SPARQL protocol, and SPARQL query processing is distributed between smart clients and LDF servers. Hartig et al. [17] formalized this approach using Linked Data Fragment machines (LDFMs). The Triple Pattern Fragments (TPF) approach [27] is one implementation of LDF where the server only evaluates *paginated triple pattern queries*. As paginated triple pattern queries can be evaluated in bounded time [18], the server does not suffer from the convoy effect. However, as joins are performed on the client, the intensive transfer of intermediate results leads to poor SPARQL query execution performance. For example, the evaluation of the query Q_1, of Figure 1, using the TPF approach generates 507156 subqueries and transfers 2Gb of intermediate results in more than 2 hours. The Bindings-Restricted Triple Pattern Fragments (BrTPF) approach [15] improves the TPF approach by using the bind-join algorithm [14] to reduce transferred data but joins still executed by the client. In this paper, we explore how Web preemption allows the server to execute a larger fragment of the SPARQL algebra, including joins, without generating convoy effects. Processing joins on server side allow to drastically reduce transferred data between client and server and improve significantly performance. For example, SaGe executes the query Q_1 of Figure 1 in less than 53s, with 553 requests and 2.1Mb transferred.

Preemption and Web preemption. FCFS scheduling policies and the convoy effect [6] have been heavily studied in operating systems. In a system where the duration of tasks vary, a long-running task can block all other tasks, deteriorating the average completion time for all tasks. The *Round-Robin* (RR) algorithm [19] provides a fair allocation of CPU between tasks, avoids convoy effect, reduces the waiting time and provides good responsiveness. RR runs a task for a given *time quantum*, then suspends it and switches to the next task. It repeatedly does so until all tasks are finished. The value of this time quantum is critical for performance: when too high, RR behaves like FCFS with the same issues, and when its too low, the overhead of context switching dominates the overall performance. The action of suspending a task with the intention of resuming it later is called preemption. In public Web servers, preemption is already provided by the operating systems, but *only between running tasks*, excluding those waiting in the server's queue. If we want to build a fully *preemptive Web server*, we need to consider the queries sent to the server as the tasks and the Web server as the resource that tasks competed to get access to. In this paper, we explore how preemption can be defined at the Web level to execute SPARQL queries.

3 WEB PREEMPTION APPROACH

We define *Web preemption* as the capacity of a Web server to suspend a running query after a fixed quantum of time and resume the next waiting query. When suspended, partial results and the state of the suspended query S_i are returned to the Web client [6]. The client can resume query execution by sending S_i back to the Web server. Compared to a First-Come First-Served (FCFS) scheduling policy,

[4] https://www.w3.org/TR/2013/REC-sparql11-protocol-20130321/

[5] All results were obtained on DBpedia version 2016-04.

[6] S_i can be returned to the client or saved server-side and returned by reference.

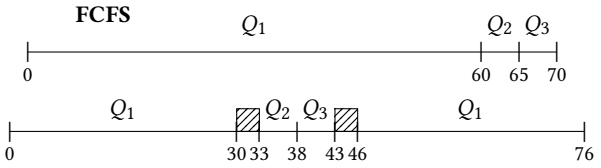


Figure 2: First-Come First-Served (FCFS) policy compared to Web Preemption (time quantum of 30s and overhead of 3s).

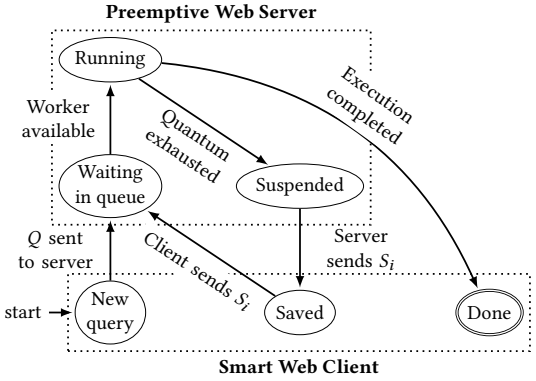


Figure 3: Possible states of a query execution in a preemptive Web Server.

Web preemption provides *a fair allocation of Web server resources across queries, a better average query completion time per query and a better time for first results* [2]. To illustrate, consider three SPARQL queries Q_1, Q_2, and Q_3 submitted concurrently by three different Web clients. Q_1, Q_2, Q_3 execution times are respectively 60s, 5s and 5s. Figure 2 presents a possible execution of these queries with a FCFS policy on the first line. In this case, the throughput of FCFS is $\frac{3}{70} = 0.042$ queries per second, the average completion time per query is $\frac{60+65+70}{3} = 65s$ and the average time for first results is also 65s. The second line describes the execution of $Q_1 - Q_3$ using Web preemption, with a time quantum of 30s. We consider a preemption overhead of 3s (10% of the quantum). In this case, the throughput is $\frac{3}{76} = 0.039$ query per second but the average completion time per query is $\frac{76+38+43}{3} = 52.3s$ and the average time for first results is approximately $\frac{30+38+43}{3} = 37s$. If the quantum is set to 60s, then Web preemption is equivalent to FCFS. If the quantum is too low, then the throughput and the average completion time are deteriorated.

Consequently, the challenges with Web preemption are *to bound the preemption overhead in time and space* and *determine the time quantum* to amortize the overhead.

3.1 Web Preemption Model

We consider a *preemptive Web server*, hosting RDF datasets, and a *smart Web client*, that evaluates SPARQL queries using the server. For the sake of simplicity, we only consider *read-only queries* [7] in this paper. The server has a pool of *server workers* parameterized with a fixed time quantum. Workers are in charge of queries execution. The server has also a *query queue* to store incoming queries when all workers are busy. We consider an *infinite* population of clients, a *finite* server queue and a *finite* number of Web workers.

The preemptive Web server suspends and resumes queries after the time quantum. A running query Q_i is represented by its *physical query execution plan*, denoted P_{Q_i}. Suspending the execution of Q_i is an operation applied on P_{Q_i} that produces a *saved state* S_i; $\texttt{Suspend}(P_{Q_i}) = S_i$. Resuming the execution of a query Q_i is the inverse operation, it takes S_i as parameter and restores the physical query execution plan in its suspended state. Therefore, the preemption is correct if $\texttt{Resume}(\texttt{Suspend}(P_{Q_i})) = P_{Q_i}$.

Figure 3 presents possible states of a query. The transitions are executed either by the Web server or by the client.

The Web server accepts, in its waiting queue, Web requests containing either SPARQL queries Q_i, or suspended queries S_i. If a worker is available, it picks a query in the waiting queue. For Q_i, the worker produces a physical query execution plan P_{Q_i} using the *optimize-then-execute* [12] paradigm and starts its execution for the time quantum. For S_i, the server resumes the execution of Q_i. The time to produce or resume the physical query execution plan for a query is not deducted from the quantum.

If a query terminates before the time quantum, then results are returned to the Web client. If the time quantum is exhausted and the query is still running, then the Suspend operation is triggered, producing a state S_i which is returned to the Web client with partial results. The time to suspend the query is not deducted from the quantum. Finally, the Web client is free to continue the query execution by sending S_i back to the Web server.

The main source of overhead in this model is the time and space complexity of the Suspend and Resume operations, *i.e.*, time to stop and save the running query followed by the time to resume the next waiting query. Our objective is to bound these complexities such that they depend only on the *query complexity*, *i.e.*, the number of operations in the query plan. Consequently, *the problem is to determine which physical query plans P_{Q_i} have a preemption overhead bounded in $O(|Q|)$*, where $|Q|$ denotes the number of operators in the expression tree of P_{Q_i}.

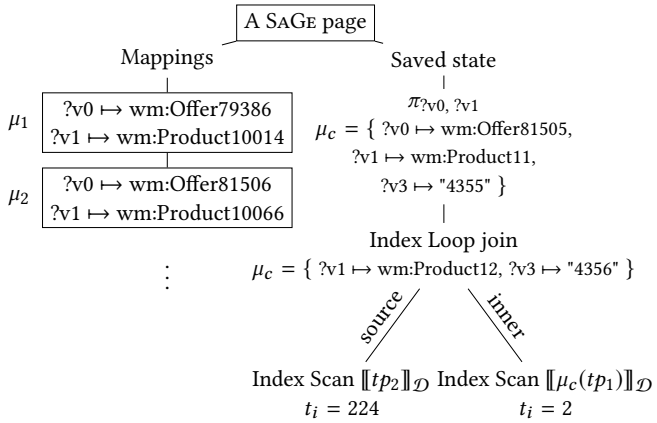
3.2 Suspending and Resuming Physical Query Execution Plans

The Suspend operation is applied to a running physical query execution plan P_{Q_i}. It is obtained from the logical query execution plan of Q_i by selecting physical operators that implement operators in the logical plan [11]. A running P_{Q_i} can be represented as an expression tree of physical operators where each physical operator has a state, *i.e.*, all internal data structures allocated by the operator. Suspending a plan P_{Q_i} requires to traverse the expression tree of the plan and save the state of each physical operator. To illustrate, consider the execution of query Q_2 in Figure **??**. The state of P_{Q_2} can be saved as described in Figure **??**: the state of the Scan operator is represented by the *id* of the last triple read t_i, the state of the Index Loop Join operator is represented by mappings pulled from the previous operator and the state of the inner scan of tp_1.

[7] Preemption and concurrent updates raise issues on correctness of results.

```
PREFIX schema: <http://schema.org/contentSize/>
PREFIX gr: <http://purl.org/goodrelations/>
SELECT DISTINCT ?vo ?v1 WHERE {
 ?v0 gr:includes ?v1. # tp1
 ?v1 schema:contentSize ?v3. # tp2
}
```

(a) SPARQL query Q_2, from the Waterloo SPARQL Diversity Test suite benchmark [1]



(b) One page returned by the SaGe server during Q_2 evaluation.

Figure 4: A tree representation of a page returned by the SaGe server when executing SPARQL query Q_2 with the saved plan passed by value.

Suspending and resuming a physical query execution plan raise several major issues.

Suspending physical operators in constant time. Bounding the time complexity of $Suspend(P_{Q_i})$ to $O(|Q_i|)$ requires to save the state of all physical operators in constant time. However, *some physical operators need to materialize data*, *i.e.*, build collections of mappings collected from other operators to perform their actions. We call these operators *full-mappings* operators, in opposition to *mapping-at-a-time* operators that only need to consume one mapping at a time from child operators[8]. Saving the state of full-mappings operators cannot be done in constant time. To overcome this problem, we distribute operators between a SaGe server and a SaGe smart client as follows:

- *Mapping-at-a-time* operators are suitable for Web preemption, so they are supported by the SaGe server. These operators are a subset of CoreSPARQL [17], composed of Triple Patterns, AND, UNION, FILTER and SELECT. We explain how to implement these operators server-side in Section 3.3.
- *Full-mappings* operators do not support Web preemption, so they are implemented in the SaGe smart client. These operators are: OPTIONAL, SERVICE, ORDER BY, GROUP BY, DISTINCT, MINUS, FILTER EXIST and aggregations (COUNT, AVG, SUM, MIN, MAX). We explain how to implement these operators in the smart client in Section 3.4.

[8]This is a clear reference to *full-relation* and *tuple-at-a-time* operators in database systems [11][Chapter 15.2]

As proposed in LDF [17, 27], the collaboration of the SaGe smart client and the SaGe server allows to support full SPARQL queries.

Communication between operators. Bounding the time complexity of $Suspend(P_{Q_i})$ to $O(|Q_i|)$ requires to avoid materialization of intermediate results when physical operators communicate. This only concerns operators of the SaGe server. To solve this issue, we follow *the iterator model* [12, 13]. In this model, operators are connected in a *pipeline*, where they are chained together in a *pull-fashion*, such as one iterator pulls solution mappings from its predecessor(s) to produce results.

Saving consistent states of the physical query plan. Some physical query operators have critical sections and cannot be interrupted until exiting those sections. Saving the physical plan in an inconsistent state leads to incorrect preemption. In other words, we could have $Resume(Suspend(P_{Q_i})) \neq P_{Q_i}$. To solve this issue, we have to detail, for each physical operator supported by the SaGe server, where are located critical sections and estimate if the waiting time for exiting the critical section is acceptable.

Resuming physical operators in constant time. Reading a saved plan as the one presented in Figure **??** should be in $O(|Q|)$. The *Index Scan* $[\![tp_2]\!]_{\mathcal{D}}$ has been suspended after reading the triple with $id = 224$. Resuming the scan requires that the access to the triple with $id \geq 224$ is in constant time. This can be achieved with adequate indexes. To solve this issue, we have to define clearly for each physical operator what are the requirements on backend to bound the overhead of resuming.

3.3 The SaGe Preemptable Server

The SaGe server supports Triple Patterns, AND, UNION, FILTER and SELECT operators. The logical and physical query plans are builds thanks to the *optimize-then-execute* [12] paradigm. The main issues here are the cost of the `Suspend` and `Resume` operations on the physical plan and how to interrupt physical operators.

To support preemption, we extend classical iterators to *preemptable iterators*, formalized in Definition 3.1. As iterators are connected in a pipeline, we consider that each iterator is also responsible for recursively stopping, saving and resuming its predecessors.

Definition 3.1 (Preemptable iterator). A *preemptable iterator* is an iterator that supports, in addition to the classic `Open`, `GetNext` and `Close` methods [13], the following methods:

- `Stop`: interrupts the iterator and its predecessor(s). `Stop` waits for all non interruptible sections to complete.
- `Save`: serializes the current state of the iterator and its predecessor(s) to produce a *saved state*.
- `Load`: reloads the iterator and its predecessor(s) from a saved state.

Algorithm 1 presents the implementation of the `Suspend` and `Resume` functions for a pipeline of preemptable query iterators. `Suspend` simply stops and saves recursively each iterator in the pipeline and `Resume` reloads the pipeline in the suspended state using a serialized state. To illustrate, consider the SaGe page of Figure **??**. This page contains the plan that evaluates the SPARQL query Q_2 of size $|Q_2| = 4$, using *index loop joins* [12]. The evaluation of tp_2 has been preempted after scanning 224 solution mappings.

Algorithm 1: Implementation of the `Suspend` and `Resume` functions following the iterator model

Require: $\mathcal{I}$: pipeline of iterators, S: serialized pipeline state (as generated by `Suspend`)

```
1 Function Suspend(I):
2     let root←first iterator in I
3     Call root.Stop()
4     return root.Save()
5 Function Resume(I, S):
6     let root←first iterator in I
7     Call root.Load(S)
8     return I
```

Table 1: Complexities of preemption of physical query iterators. $|id|$ and $|tp|$ denote the size of encoding an index key and a triple pattern, respectively.

Preemptable iterator	Space complexity of local state	Time complexity of loading local state	Remarks
$\pi_{v_1,\ldots,v_k}(P)$	$O(k + \|var(P)\|)$	$O(1)$	
Index Scan tp	$O(\|tp\| + \|id\|)$	$O(\log_b(\|\mathcal{D}\|))$	Require indexes on all kinds of triple patterns
Merge Join $P_1 \bowtie P_2$	$O(\|var(P_1)\| + \|var(P_2)\|)$	$O(1)$	
Index Loop Join $P \bowtie tp$	$O(\|var(P)\| + \|tp\| + \|id\|)$	$O(\log_b(\|\mathcal{D}\|))$	
P_1 UNION P_2	$O(1)$	$O(1)$	Multi-set Union
P Filter $\mathcal{R}$	$O(\|var(P)\| + \|\mathcal{R}\|)$	$O(1)$	Pure logical expression only
Server physical plan	$O(\|Q\| \times \log_2(\|\mathcal{D}\|)$	$O(\|Q\| \times \log_b(\|\mathcal{D}\|))$	

The index loop join with tp_1 has been preempted after scanning two triples from the inner loop.

In the following, we review SPARQL operators implemented by the SaGe server as preemptable query iterators. We modify the regular implementations of these operators to include non interruptible section when needed. Operations left unchanged are not detailed. Table 1 resumes the complexities related to the preemption of server operators, where $var(P)$ denotes the set of variables in P as defined in [21].

Algorithm 2: A *Preemptable Select Iterator*

Require: $V = \{v_1, \ldots, v_k\}$: projection variables, $\mathcal{I}$: predecessor in the pipeline

Data: $\mu \leftarrow nil$

```
1 Function GetNext():
2     if μ = nil then
3         μ ← I.GetNext()
4     non interruptible
5         let μ' ← Proj(μ, V)
6         μ ← nil
7         return μ'
8 Function Save():
9     return μ
10 Function Load(μ'):
11    μ ← μ'
```

SELECT operator. : The projection operator $\pi_V(P)$ [25], also called SELECT, performs the projection of mappings obtained from its predecessor P according to a set of projection variables $V = \{v_1, \ldots, v_k\}$. Algorithm 2 gives the implementation of this iterator. In this algorithm, Lines 4-7 are non-interruptible. This is necessary to ensure that the operator does not discard mappings without applying projection to them. Consequently, after the termination of $Suspend(P)$, $\pi_V(P)$ can be suspended in $O(k + |var(P)|)$ in space. The projection iterator reloads its local state in $O(1)$.

Triple Pattern: The evaluation of a triple pattern tp over D $[\![tp]\!]_{\mathcal{D}}$ [21] is the set of solution mappings of tp. μ is a solution mapping if $\mu(tp) = t$ where t is the triple obtained by replacing the variables in tp according to μ, such that $t \in \mathcal{D}$. The evaluation of $[\![tp]\!]_{\mathcal{D}}$ requires to read $\mathcal{D}$ and produce the corresponding set of solutions mappings.

The triple pattern operator supposes that data are accessed using *index scans* and a clustered index [11]. Algorithm 3 gives the implementation of a preemptable index scan for evaluating a triple pattern. We omitted the `Stop()` operation, as the iterator does not need to do any action to stop. The `Save()` operation stores the triple pattern tp and the index's $id(t)$ of the last triple t read. Thus, suspending an index scan is done in constant time and the size of the operator state is in $O(|tp| + |id(t)|)$.

The `Load()` operation uses the saved index's $id(t)$ on tp to find the last matching RDF triple read. The time complexity of this operation predominates the time complexity of `Resume`. This complexity depends on the type of index used. With a B+-tree [7], the time complexity of resuming is bound to $O(\log_b(|\mathcal{D}|))$. B+-tree indexes are offered by many RDF data storage systems [8, 20, 28].

Algorithm 3: A *Preemptable Index Scan Iterator*, evaluating a triple pattern tp using an index

Require: tp: triple pattern, $\mathcal{D}$: RDF dataset queried, $\mathcal{I}_{tp}$: clustered index over tp

Data: t: last matching RDF triple read

```
1 Function GetNext():
2     non interruptible
3         t ← next RDF triple matching tp in D
4         let μ ← set of solutions mappings such as μ(t) = tp
5         return μ
6 Function Save():
7     return ⟨tp, id(t)⟩
8 Function Load(id):
9     t ← I_tp.Locate(id)  // Locate the last triple read
```

Basic Graph patterns (AND):. A Basic Graph pattern (BGP) evaluation corresponds to the natural join of a set of triple patterns. Join operators fall into three categories [11]: (1) Hash-based joins *e.g.*, Symmetric Hash join or XJoin, (2) Sort-based joins, *e.g.*, (Sort) Merge join, (3) Loop joins, *e.g.*, index loop or nested loop joins.

Hash-based joins [12, 29] operators are not suitable for preemption. As they build an in-memory hash table on one or more inputs to evaluate the join, they are considered as full-mappings operators. The sort-based joins and loop joins are mapping-at-a-time operators; consequently, they could be preempted with low overhead. However, for sort-based joins, we can only consider merge joins, *i.e.*, joins inputs are already sorted on the join attribute. Otherwise, this will require to perform an in-memory sort on the inputs. In the

following, we present algorithms for building preemptable Merge join and preemptable Index Loop join operators:

Algorithm 4: A *Preemptable Merge Join Iterator* I, joining the output of two iterators I_{left} and I_{right}.

Require: I_{left}: the outer join input, I_{right}: the inner join input, μ_l: the last element read from I_{left}, μ_r: the last element read from I_{right}, $\mathcal{D}$: RDF dataset queried

Function *Stop()*:
  I_{left}.Stop()
  I_{right}.Stop()
Function *Save()*:
  let $s_l \leftarrow I_{\text{left}}$.Save()
  let $s_r \leftarrow I_{\text{right}}$.Save()
  return $\langle s_l, s_r, \mu_l, \mu_r \rangle$
Function *Load*(s_l, s_r, μ'_l, μ'_r):
  I_{left}.Load(s_l)
  I_{right}.Load(s_r)
  $\mu_l \leftarrow \mu'_l$
  $\mu_r \leftarrow \mu'_r$

Algorithm 5: A *Preemptable Index Join Iterator* I_i: a preemptable join operator used by SaGe for BGP evaluation

Require: I_{left}: the iterator responsible for the evaluation of the outer join input, tp_r: the inner join input, $\mathcal{D}$: RDF dataset queried.

Function *Open()*:
  I_{left}.*Open*()
  $\mu_c \leftarrow nil$
  $I_{\text{find}} \leftarrow$ a *PreemptableIndexScanIterator* over $\emptyset$
Function *GetNext()*:
  while $\neg I_{find}.HasNext()$ **do**
    $\mu_c \leftarrow I_{\text{left}}.GetNext()$
    if $\mu_c = nil$ **then**
      return *nil*
    $I_{\text{find}} \leftarrow$ *PreemptableIndexScanIterator* over $[\![\mu_c(tp_r)]\!]_{\mathcal{D}}$
  non interruptible
    let $\mu \leftarrow I_{\text{find}}.GetNext()$
    return $\mu \cup \mu_c$
Function *Load*(tp', μ', t):
  $tp_r \leftarrow tp'$
  if $\mu' \neq nil$ **then**
    $\mu_c \leftarrow \mu'$
    $I_{\text{find}} \leftarrow$ *PreemptableIndexScanIterator* over $[\![\mu_c(tp_i)]\!]_{\mathcal{D}}$
    I_{find}.Load(t);
Function *Save()*:
  let $t \leftarrow$ the last triple read by I_{find}
  return $\langle tp_r, \mu_c, t \rangle$
Function *Stop()*:
  $I_{\text{left}}.Stop()$
  $I_{\text{find}}.Stop()$

Preemptable merge join: The Merge join algorithm merges the solutions mappings from two join inputs, *i.e.*, others operators sorted on the join attribute. SaGe extends the classic merge join [12] to the *Preemptable Merge join* iterator as shown in Algorithm 4. Basically, its `Stop`, `Save` and `Load` functions recursively stop, save or load the joins inputs, respectively. Thus, the only internal data structures holds by the join operator are the two inputs and the last sets of solution mappings read from them. As described in Table 1, the local state of a merge join is resumable in constant time, while its space complexity depends on the size of two sets of solution mappings saved.

Preemptable Index Loop join: The Index Loop join algorithm [12] exploits indexes on the inner triple pattern for efficient join processing. This algorithm has already been used for evaluating BGPs in [16]. SaGe extends the classic Index Loop joins to a *Preemptable Index join Iterator* (PIJ-Iterator) presented in Algorithm 5. To produce solutions, the iterator executes the same steps repeatedly until all solutions are produced: (1) It pulls solutions mappings μ_c from its predecessor. (2) It applies μ_c to tp_i to generate a *bound pattern* $b = \mu_c(tp_i)$. (3) If b has no solution mappings in $\mathcal{D}$, it tries to read again from its predecessor (jump back to Step 1). (4) Otherwise, it reads triple matching b in $\mathcal{D}$, produces the associated set of solution mappings and then goes back to Step 1.

A *PIJ-Iterator* supports preemption through the `Stop`, `Save` and `Load` functions. The non-interruptible section of `GetNext()` only concerns a scan in the dataset. Therefore, in the worse case, the iterator has to wait for one index scan to complete before being interruptible. As with the merge join, the `Save` and `Load` functions needs to stop and resume the left join input. The latter also needs to resume an Index Scan, which can be resumed in $O(\log_b(|\mathcal{D}|)$.

Regarding the saved state, the iterator saves the triple pattern joined with the last set of solution mappings pulled from the predecessor and an index scan. For instance, in Figure **??**, the saved state of the join of tp_2 and tp_1 is μ_c = {`?v1` ↦ `wm:Product12, ?v3` ↦ `"4356"`}.

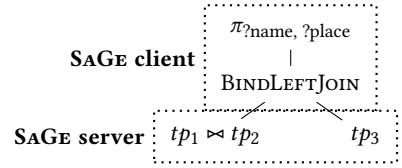
UNION fragment: The UNION fragment is defined as the union of solution mappings from two graph patterns. We consider a *multi-set union* semantic, as set unions require to save intermediate results to remove duplicates and thus cannot be implemented as mapping-at-a-time operators. The set semantic can be restored by the smart Web client using the DISTINCT modifier on client-side. Evaluating a multi-set union is equivalent to the *sequential evaluation* of all graph patterns in the union. When preemption occurs, all graph patterns in the union are saved recursively. So the union has no local state on its own. Similarly, to resume an union evaluation, each graph pattern is reloaded recursively.

FILTER fragment: A SPARQL FILTER is denoted $F = \sigma_{\mathcal{R}}(P)$, where P is a graph pattern and $\mathcal{R}$ is a built-in filter condition. The evaluation of F yields the solutions mappings of P that verify $\mathcal{R}$. Some filter conditions require collection of mappings to be evaluated, like the EXISTS filter which requires the evaluation of a graph pattern. Consequently, we limit the filter condition to *pure logical expressions* (=, <, ≥, ∧, etc) as defined in [21, 25]. The preemption of a FILTER is similar to those of a projection. We only need to suspend or resume P, respectively. For the local state, we need to save the last solution mappings pulled from P and $\mathcal{R}$.

Table 1 summarizes the complexities of `Suspend` and `Resume` in time and space. The space complexity to `Save` a physical plan is established to $O(|Q| \times \log_2(|\mathcal{D}|))$. We supposed that $|var(P)|$ and the number of $|tp|$ to save are close to $|Q|$. However, the size of index

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?name ?place WHERE {
 ?actor a dbo:Actor . # tp1
 ?actor rdfs:label ?name . # tp2
 OPTIONAL { ?actor dbo:birthPlace ?place . # tp3 }
}
```

(a) SPARQL query Q_3: finds all actors with their names and their birth places, if they exist.



(b) Client-side physical query execution plan.

Figure 5: Physical query execution plan used by the SaGe smart Web client for the query Q_3.

IDs are integers that can be as big as the number of RDF triples in $\mathcal{D}$. Hence, they can be encoded in at most $\log_2(|\mathcal{D}|)$ bits. The time complexity to `Resume` a physical plan is $O(|Q| \times \log_b(|\mathcal{D}|))$. It is higher than the time complexity of `Suspend`, as resuming index scans can be costly. Overall, the complexity of Web preemption is higher than the initial $O(|Q|)$ stated in Section 3.1. However, we demonstrate empirically in Section 4 that time and space complexity can be kept under a few milliseconds and kilobytes, respectively.

3.4 SaGe smart Web client

The SaGe approach requires a *smart Web client* for processing SPARQL queries for two reasons. First, the smart Web client must be able to continue query execution after receiving a saved plan from the server. Second, as the preemptable server only implements a fragment of SPARQL, the smart Web client has to implement the missing operators to support full SPARQL queries. It includes SERVICE, ORDER BY, GROUP BY, DISTINCT, MINUS, FILTER EXIST and aggregations (COUNT, AVG, SUM, MIN, MAX), but also $\bowtie$, $\cup$, $⟕$ and π to be able to recombine results obtained from the SaGe server. Consequently, the SaGe smart client is a *SPARQL query engine* that accesses RDF datasets through the SaGe server. Given a SPARQL query, the SaGe client parses it into a logical query execution plan, optimizes it and then builds its own physical query execution. The leafs of the plan must correspond to subqueries evaluable by a SaGe server. Figure **??** shows the execution plan build by the SaGe client for executing query Q_3, from Figure **??**.

Compared to a SPARQL endpoint, processing SPARQL queries with the SaGe smart client has an overhead in terms of the number of request sent to the server and transferred data[9]. With a SPARQL endpoint, a web client executes a query by sending a single web request and receives only query results. The smart client overhead for executing the same query is the additional number of request and data transfered to obtain the same results. Consequently, the client overhead has two components:

Number of requests: To execute a SPARQL query, a SaGe client needs $n \geq 1$ requests. We can distinguish two cases: (1) The query is fully supported by the SaGe server, so n is equal to the number of time quantum required to process the query. Notice that the client needs to pay the network latency twice per request. (2) The query is not fully supported by the SaGe server. Then, the client decomposes the query into a set of subqueries supported by the server, evaluate each subquery as in the first case, and recombine intermediate results to produce query results.

Data transfer:) We also distinguish two cases: (1) If the query is fully supported by the SaGe server, the only overhead is the size of the saved plan S_i multiplied by the number of requests. Notice that the saved plan S_i can be returned by reference or by value, *i.e.*, saved server-side or client-side. (2) If the query is not fully supported by the SaGe server, then the client decomposes the query and recombine results of subqueries. Among these results, some are intermediate results and part of the client overhead.

Consequently, the challenge for the smart client is to *minimize the overhead in terms of the number of requests and transferred data.* The data transfer could be reduced by saving the plans server-side rather than returning it to clients[10]. However, the main source of client overhead comes from the decomposition made to support full SPARQL queries, as these queries increase the number of requests sent to the server. Some decompositions are more costly than others. To illustrate, consider the evaluation of (P_1 OPTIONAL P_2) where P_1 and P_2 are expressions supported by the SaGe server. A possible approach is to evaluate $[\![P_1]\!]_{\mathcal{D}}$ and $[\![P_2]\!]_{\mathcal{D}}$ on the server, and then perform the left outer join on the client. This strategy generates only two subqueries but materializes $[\![P_2]\!]_{\mathcal{D}}$ on client. If there is no join results, $[\![P_2]\!]_{\mathcal{D}}$ are just useless intermediate results.

Another approach is to rely on a *nested loop join approach*: evaluates $[\![P_1]\!]_{\mathcal{D}}$ and for each $\mu_1 \in [\![P_1]\!]_{\mathcal{D}}$, if $\mu_2 \in [\![\mu_1(P2)]\!]_{\mathcal{D}}$ then $\{\mu_1 \cup \mu_2\}$ are solutions to $P_1 ⟕ P_2$. Otherwise, only μ_1 is a solution to the left-join. This approach sends *at least* as many subqueries to the server than there is solutions to $[\![P_1]\!]_{\mathcal{D}}$.

To reduce the communication, the SaGe client implements BindLeftJoins to process local join in a block fashion, by sending unions of BGPs to the server. This technique is already used in federated SPARQL query processing [26] with bound joins and in BrTPF [15]. Consequently, the number of request to the SaGe server is reduced by a factor equivalent to the size of a block of mappings. However, the number of requests sent still depends on the cardinality of P_1.

Consequently, we propose a new technique, called OptJoin, for optimizing the evaluation of a subclass of left-joins. The approach relies on the fact that:

$$[\![P_1 ⟕ P_2]\!]_{\mathcal{D}} = [\![P_1 \bowtie P_2]\!]_{\mathcal{D}} \cup ([\![P_1]\!]_{\mathcal{D}} \setminus [\![\pi_{var(P_1)}(P_1 \bowtie P_2)]\!]_{\mathcal{D}})$$

So we can deduce that: $[\![P_1 ⟕ P_2]\!]_{\mathcal{D}} \subseteq [\![(P_1 \bowtie P_2) \cup P_1]\!]_{\mathcal{D}}$. If both P_1 and P_2 are evaluable by the SaGe server, then the smart client computes the left-join as follows. First, it sends the query $(P_1 \bowtie P_2) \cup P_1$ to the server. Then, for each mapping μ received, it builds local materialized views for $[\![P_1 \bowtie P_2]\!]_{\mathcal{D}}$ and $[\![P_1]\!]_{\mathcal{D}}$. The client knows that $\mu \in [\![P_1 \bowtie P_2]\!]_{\mathcal{D}}$ if $dom(\mu) \subset var(P_1 \bowtie P_2)$, otherwise $\mu \in [\![P_1]\!]_{\mathcal{D}}$. Finally, the client uses the views to process the left-join locally. With this technique, the client only use one

[9] The client overhead should not be confused with the server overhead.

[10] In our implementation, we choose to save plans client-side to tolerate server failures.

subquery to evaluate the left-join and, in the worst case, it transfers $[\![P_1 \bowtie P_2]\!]_{\mathcal{D}}$ as additional intermediate results.

To illustrate, consider query Q_3 from Figure **??**. Q_3, with 88334 solutions. The cardinality of $tp_1 \bowtie tp_2$ is also of 88334, as every actor has a birth place. This is the worse case for a BindLeftJoin, which will require $\frac{88334}{Blocksize}$ additional requests to evaluate the left join. However, with an OptJoin, the client is able to evaluate Q_3 in approximately 500 requests.

We implement both BindLeftJoin and OptJoin as physical operators to evaluate OPTIONALs, depending on the query. We also implement regular BindJoin for processing SERVICE queries.

4 EXPERIMENTAL STUDY

We want to empirically answer the following questions: What is the overhead of Web preemption in time and space? Does Web preemption improves the average workload completion time? Does Web preemption improves the time for first results? What are the client overheads in terms of numbers of requests and data transfer? We use Virtuoso as the baseline for comparing with SPARQL endpoints, with TPF and BrTPF as the baselines for LDF approaches.

We implemented the SaGe client in Java, using Apache Jena[11]. As an extension of Jena, SaGe is just as compliant with SPARQL 1.1. The SaGe server is implemented as a Python Web service and uses HDT [9] (v1.3.2) for storing data. Notice that the current implementation of HDT cannot ensure $log_b(n)$ access time for all triple patterns, like (?s p ?o). This impacts negatively the performance of SaGe when resuming some queries. The code and the experimental setup are available on the companion website[12].

4.1 Experimental setup

Dataset and Queries: We use the Waterloo SPARQL Diversity Benchmark (WatDiv[13]) [1]. We re-use the RDF dataset and the SPARQL queries from the BrTPF [15] experimental study[14]. The dataset contains 10^7 triples and queries are arranged in 50 workloads of 193 queries each. They are SPARQL conjunctive queries with STAR, PATH and SNOWFLAKE shapes. They vary in complexity, up to 10 joins per query with very high and very low selectivity. 20% of queries require more than $\approx 30s$ to be executed using the virtuoso server. All workloads follow nearly the same distribution of query execution times as presented in Figure 6. The execution times are measured for one workload of 193 queries with SaGe and an infinite time quantum.

Approaches: We compare the following approaches:

- SaGe: We run the SaGe query engine with various time quantum: 75ms and 1s, denoted SaGe-75ms and SaGe-1s respectively. HDT indexes are loaded in memory while HDT data is stored on disk.
- *Virtuoso:* We run the Virtuoso SPARQL endpoint [8] (v7.2.4), *without any quotas or limitations* .
- *TPF:* We run the standard TPF client (v2.0.5) and TPF server (v2.2.3) with HDT files as backend (same settings as SaGe).

[11] https://jena.apache.org/
[12] https://github.com/sage-org/sage-experiments
[13] http://dsg.uwaterloo.ca/watdiv/
[14] http://olafhartig.de/brTPF-ODBASE2016

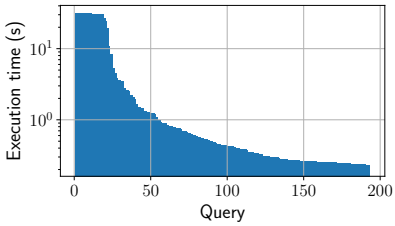


Figure 6: Distribution of query execution time.

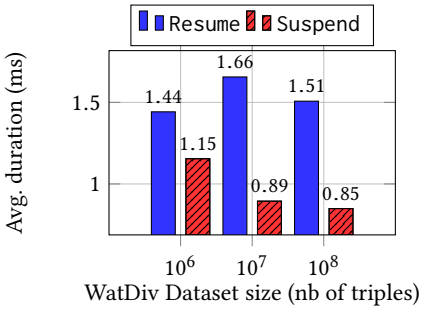


Figure 7: Average preemption overhead.

Mean	Min	Max	Standard deviation
1.716 kb	0.276 kb	6.212 kb	1.337 kb

Table 2: Space of saved physical query execution plans.

- *BrTPF:* We run the BrTPF client and server used in [15], with HDT files as backend (same settings as SaGe). BrTPF is currently the LDF approach with the lowest data transfer [15].

Servers configurations: We run all the servers on a machine with Intel(R) Xeon(R) CPU E7-8870@2.10GHz and 1.5TB RAM.

Clients configurations: In order to generate load over servers, we rely on 50 clients, each one executing a different workload of queries. All clients start executing their workload simultaneously. The clients access servers through HTTP proxies to ensure that client-server latency is kept around 50ms.

Evaluation Metrics: Presented results correspond to the average obtained of three successive execution of the queries workloads. (1) *Workload completion time (WCT)*: is the total time taken by a client to evaluate a set of SPARQL queries, measured as the time between the first query starting and the last query completing. (2) *Time for first results (TFR)* for a query: is the time between the query starting and the production of the first query's results. (3) *Time preemption overhead*: is the total time taken by the server's `Suspend` and `Resume` operations. (4) *Number of HTTP requests and data transfer*: is the total number of HTTP requests sent by a client to a server and the number of transferred data when executing a SPARQL query.

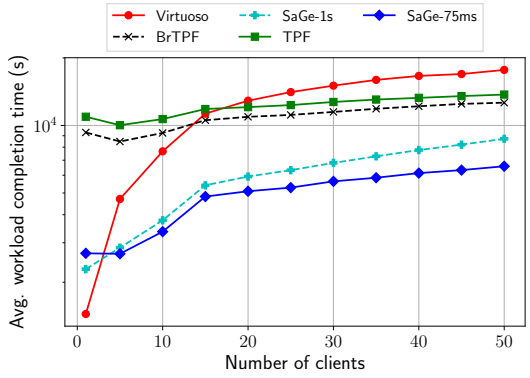


Figure 8: Average workload completion time per client, with up to 50 concurrent clients (logarithmic scale).

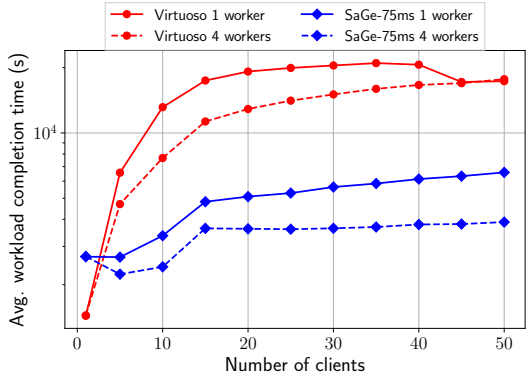


Figure 9: Average workload completion time per client, with 4 workers (logarithmic scale).

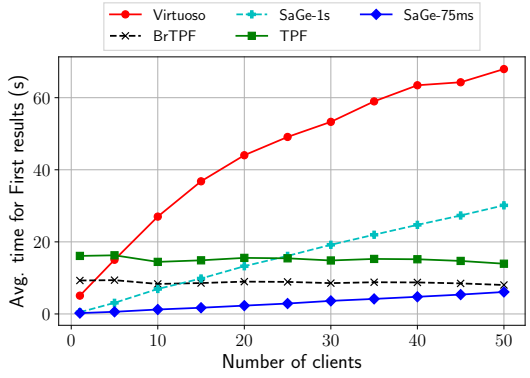


Figure 10: Average time for first results (over all queries), with up to 50 concurrent clients (linear scale).

4.2 Experimental results

We first ensure that the SaGe approach yield complete results. We run both Virtuoso and SaGe and verify that, for each query, SaGe delivers complete results using Virtuoso results as ground truth.

What is the overhead in time of Web preemption? The overhead in time of Web preemption is the time spent by the Web server for suspending a running query and the time spent for resuming the next waiting query. To measure the overhead, we run one workload of queries using SaGe-75ms and measure time elapsed for the `Suspend` and `Resume` operations. Figure 7 shows the overhead in time for SaGe `Suspend` and `Resume` operations using different sizes of the WatDiv dataset. Generally, the size of the dataset does not impact the overhead, which is around 1*ms* for `Suspend` and 1.5*ms* for `Resume`. As expected, the overhead of the `Resume` operation is greater than the one of the `Suspend` operation, due to the cost of resuming the Index scans in the plan. With a quantum of 75ms, the overhead is $\approx$ 3% of the quantum, which is negligible.

What is the overhead in space of Web preemption? The overhead in space of the Web preemption is the size of saved plans produced by the `Suspend` operation. According to Section 3.3, we determined that the SaGe physical query plans can be saved in $O(|Q|)$. To measure the overhead, we run a workload of queries using SaGe-75ms and measure the size of saved plans. Saved plans are compressed using Google Protocol Buffers [15]. Table 2 shows the overhead in space for SaGe. As we can see the size of a saved plan remains very small, with no more than 6 kb for a query with ten joins. Hence, this space is indeed proportional to the size of the plan suspended.

Does Web preemption improve the average workload completion time? To enable Web preemption, the SaGe server has a restricted choice of physical query operators, thus physical query execution plans generated by the SaGe server should be less performant than those generated by Virtuoso. This tradeoff only make sense if the Web preemption compensates the loss in performance. Compensation is possible only if the workload alternates long-running and short running queries. In the setup, each client runs a different workload of 193 queries that vary from 30s to 0.22s, following an exponential distribution. All clients execute their workload concurrently and start simultaneously. We experiment up to 50 concurrent clients by step of 5 clients. As there is only one worker on the Web server, and queries execution time vary, this setup is the worst case for Virtuoso.

Figure 8 shows the average workload completion time obtained for all approaches, *with a logarithmic scale*. As expected, Virtuoso is significantly impacted by the convoy effect and delivers the worse WTC after 20 concurrent clients. TPF and BrTPF avoid the convoy and behave similarly. BrTPF is more performant thanks to its bind-join technique that group requests to the server. SaGe-75ms and SaGe-1s avoid the convoy effect and delivers better WTC than TPF and BrTPF. As expected, increasing the time quantum also increases the probability of convoy effect and, globally, SaGe-75ms offers the best WTC. We rerun the same experiment with 4 workers for SaGe-75ms and Virtuoso. Figure 9 shows the average workload completion time obtained for both approaches. As we can see, both approaches benefit of the four workers. However, Virtuoso still suffers from the convoy effect.

Does Web preemption improve the time for the first results? The Time for first results (TFR) for a query is the time between the query starting and the production of the first query's results. Web

[15] https://developers.google.com/protocol-buffers/

Dataset	Virtuoso	SaGe-1s	SaGe-75ms	BrTPF	TPF
WatDiv 10^7	193	645	4 082	$9,\ 2 \cdot 10^4$	$2,\ 55 \cdot 10^5$
FEASIBLE	166	1 822	3 305	$2,\ 95 \cdot 10^4$	$1,\ 86 \cdot 10^5$

(a) Results with WatDiv and FEASIBLE-DBpedia datasets

Time quantum	SaGe+BindLeftJoin	SaGe+OptJoin
75ms	72 489	5 656
1s	70 964	511

(b) Comparison for BindLeftJoin and OptJoin operators

Table 3: Average number of HTTP requests sent to server

preemption should provides better time for first results. Avoiding convoy effect allows to start queries earlier and then get results earlier. We rerun the same setup as in the previous section and measure the time for first results (TFR). Figure 10 shows the results *with a linear scale*. As expected, Virtuoso suffers from the convoy effect that degrades significantly the TFR when the concurrency increases. TPF and BrTPF do not suffer from the convoy effect and TFR is clearly stable over concurrency. The main reason is that delivering a page of result for a single triple pattern takes $\approx$ 5ms in our experiment, so the waiting time on the TPF server grows very slowly. BrTPF is better than TPF due to its bind-join technique. The TFR for SaGe-75ms and SaGe-1s increases with the number of clients and the slope seems proportional to the quantum. This is normal because the waiting time on the server increases with the number of clients, as seen previously. Reducing the quantum improves the TFR, but increases the number of requests and thus deteriorates the WTC.

What are the client overheads in terms of number of requests and data transfer? The client overhead in requests is the number of requests that the smart client sent to the server to get complete results minus one, as Virtuoso executes all queries in one request. As WatDiv queries are pure conjunctive queries and supported by the SaGe server, the number of requests to the server is the number of time quantum required to evaluate the whole workload. We measure the number of requests sent to servers with one workload for all approaches, with results shown in Table 3 (a). As expected, Virtuoso just send 193 requests to the server. SaGe-75ms send 4082 requests to the server, while TPF send 2.55×10^5 requests to the TPF server. We can see also that increasing the time quantum significantly reduces the number of requests; SaGe-1s send only 645 requests. However, this seriously deteriorates the average WCT and TFR as presented before. To compute the overhead in data transfer of SaGe, we just need to multiply the number of requests by the average size of saved plans; for SaGe-75ms, the client overhead in data transfer is $4082 \times 1,3$ kb = 5.45 Mo. As the total size of results is 51Mo, the client overhead in data transfer is $\approx$ 10%. For TPF, the average size of a page is 7ko; $2.5 \cdot 10^5 \times 7k = 1.78$ Go, so the data transfer overhead is $\approx$ 340% for TPF.

What are the client overheads in terms of numbers of requests and data transfer for more complex queries? The WatDiv benchmark does not generate queries with OPTIONAL or FILTER operators. If some filters are supported by the SaGe server, the OPTIONAL operator and other filters clearly impact the number of requests sent to the SaGe server, as explained in Section 3.4. First, we run an experiment to measure the number of requests for queries with OPTIONAL. We generate new WatDiv queries from one set of 193 queries, using the following protocol. For each query $Q = \{tp_1, \ldots, tp_n\}$, we select $tp_k \in Q$ with the highest cardinality, then we generate $Q' = tp_k \bowtie (Q \setminus tp_k)$. Such queries verify conditions for using BindLeftJoin and OptJoin. They are challenging for the BindLeftJoin as they generate many mappings; they are also challenging for the OptJoin as all joins yield results. Table 3(b) shows the results when evaluating OPTIONAL queries with OptJoin and BindLeftJoin approaches. We observe that, in general, OptJoin outperforms BindLeftJoin. Furthermore, OptJoin is improved when using a higher quantum, as the single subquery sent is evaluated more quickly. This is not the case for BindLeftJoin, as the number of requests still depends on the number of intermediate results.

Finally, we re-use 166 SELECT queries from Feasible [23] with the DBpedia 3.5.1 dataset, which were generated from real-users queries. We excluded queries that time-out, identified in [23], and measure the number of requests sent to the server. Table 3(a) shows the results. Of course, Virtuoso just send 166 requests to the server. As we can see the ratio of requests between SaGe-75 and TPF is nearly the same as the previous experiment. However, the difference between SaGe-1s and SaGe-75ms has clearly decreased, because most requests sent are produced by the decomposition of OPTIONAL and FILTER and not by the evaluation of BGPs.

5 CONCLUSION AND FUTURE WORKS

In this paper, we demonstrated how Web preemption allows not only to suspend SPARQL queries, but also to resume them. This opens the possibility to efficiently execute long-running queries with complete results. The scientific challenge was to keep the Web preemption overhead as low as possible; we demonstrated that the overhead of large a fragment of SPARQL can be kept under 2ms.

Compared to SPARQL endpoint approaches without quotas, SaGe avoids the convoy effect and is a winning bet as soon as the queries of the workload vary in execution time. Compared to LDF approaches, SaGe offers a more expressive server interface with joins, union and filter evaluated server side. Consequently, this considerably reduces data transfer and the communication cost, improving the execution time of SPARQL queries.

SaGe opens several perspectives. First, for the sake of simplicity, we made the hypothesis of read-only datasets in this paper. If we allow concurrent updates, then a query can be suspended with a version of a RDF dataset and resumed with another version. The main challenge is then to determine which consistency criteria can be ensured by the preemptive Web server. As SaGe only accesses the RDF dataset when scanning triple patterns, it could be possible to compensate concurrent updates and deliver, at least, eventual consistency for query results. Second, we aim to explore how the interface of the server can be extended to support named graphs and how the optimizer of the smart client can be tuned to produce better query decompositions, especially in the case of federated SPARQL query processing. Third, we plan to explore if more elaborated scheduling policies could increase performance. Finally, the Web preemption model is not restricted to SPARQL. An interesting perspective is to design a similar approach for SQL or GraphQL.

ACKNOWLEDGMENTS

This work has been partially funded through the FaBuLA project, part of the AtlanSTIC 2020 program.

REFERENCES

[1] Günes Aluç, Olaf Hartig, M. Tamer Özsu, and Khuzaima Daudjee. 2014. Diversified Stress Testing of RDF Data Management Systems. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I (Lecture Notes in Computer Science)*, Vol. 8796. Springer, 197–212. https://doi.org/10.1007/978-3-319-11964-9_13

[2] Thomas Anderson and Michael Dahlin. 2014. *Operating Systems: Principles and Practice* (2nd ed.). Recursive books.

[3] Carlos Buil Aranda, Aidan Hogan, Jürgen Umbrich, and Pierre-Yves Vandenbussche. 2013. SPARQL Web-Querying Infrastructure: Ready for Action?. In *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part II (Lecture Notes in Computer Science)*, Vol. 8219. Springer, 277–293. https://doi.org/10.1007/978-3-642-41338-4_18

[4] Carlos Buil Aranda, Axel Polleres, and Jürgen Umbrich. 2014. Strategies for Executing Federated Queries in SPARQL1.1. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II (Lecture Notes in Computer Science)*, Vol. 8797. Springer, 390–405. https://doi.org/10.1007/978-3-319-11915-1_25

[5] Christian Bizer, Tom Heath, and Tim Berners-Lee. 2009. Linked Data - The Story So Far. *Int. J. Semantic Web Inf. Syst.* 5, 3 (2009), 1–22. https://doi.org/10.4018/jswis.2009081901

[6] Mike W. Blasgen, Jim Gray, Michael F. Mitoma, and Thomas G. Price. 1979. The Convoy Phenomenon. *Operating Systems Review* 13, 2 (1979), 20–25. https://doi.org/10.1145/850657.850659

[7] Douglas Comer. 1979. The Ubiquitous B-Tree. *ACM Comput. Surv.* 11, 2 (1979), 121–137. https://doi.org/10.1145/356770.356776

[8] Orri Erling and Ivan Mikhailov. 2009. RDF Support in the Virtuoso DBMS. In *Networked Knowledge - Networked Media - Integrating Knowledge Management, New Media Technologies and Semantic Systems*. 7–24. https://doi.org/10.1007/978-3-642-02184-8_2

[9] Javier D. Fernández, Miguel A. Martínez-Prieto, Claudio Gutiérrez, Axel Polleres, and Mario Arias. 2013. Binary RDF representation for publication and exchange (HDT). *J. Web Sem.* 19 (2013), 22–41. https://doi.org/10.1016/j.websem.2013.01.002

[10] Dennis W. Fife. 1968. R68-47 Computer Scheduling Methods and Their Countermeasures. *IEEE Trans. Computers* 17, 11 (1968), 1098–1099. https://doi.org/10.1109/TC.1968.226869

[11] Hector Garcia-Molina, Jeffrey D. Ullman, and Jennifer Widom. 2009. *Database systems - the complete book (2. ed.)*. Pearson Education.

[12] Goetz Graefe. 1993. Query Evaluation Techniques for Large Databases. *ACM Comput. Surv.* 25, 2 (1993), 73–170. https://doi.org/10.1145/152610.152611

[13] Goetz Graefe and William J. McKenna. 1993. The Volcano Optimizer Generator: Extensibility and Efficient Search. In *Proceedings of the Ninth International Conference on Data Engineering, April 19-23, 1993, Vienna, Austria*. IEEE Computer Society, 209–218. https://doi.org/10.1109/ICDE.1993.344061

[14] Laura M. Haas, Donald Kossmann, Edward L. Wimmers, and Jun Yang. 1997. Optimizing Queries Across Diverse Data Sources. In *VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*. Morgan Kaufmann, 276–285.

[15] Olaf Hartig and Carlos Buil Aranda. 2016. Bindings-Restricted Triple Pattern Fragments. In *On the Move to Meaningful Internet Systems: OTM 2016 Conferences - Confederated International Conferences: CoopIS, C&TC, and ODBASE 2016, Rhodes, Greece, October 24-28, 2016, Proceedings (Lecture Notes in Computer Science)*, Vol. 10033. Springer, 762–779. https://doi.org/10.1007/978-3-319-48472-3_48

[16] Olaf Hartig, Christian Bizer, and Johann Christoph Freytag. 2009. Executing SPARQL Queries over the Web of Linked Data. In *The Semantic Web - ISWC 2009, 8th International Semantic Web Conference, ISWC 2009, Chantilly, VA, USA, October 25-29, 2009. Proceedings (Lecture Notes in Computer Science)*, Vol. 5823. Springer, 293–309. https://doi.org/10.1007/978-3-642-04930-9_19

[17] Olaf Hartig, Ian Letter, and Jorge Pérez. 2017. A Formal Framework for Comparing Linked Data Fragments. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I (Lecture Notes in Computer Science)*, Vol. 10587. Springer, 364–382. https://doi.org/10.1007/978-3-319-68288-4_22

[18] Lars Heling, Maribel Acosta, Maria Maleshkova, and York Sure-Vetter. 2018. Querying Large Knowledge Graphs over Triple Pattern Fragments: An Empirical Study. In *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part II*. 86–102. https://doi.org/10.1007/978-3-030-00668-6_6

[19] Leonard Kleinrock. 1964. Analysis of A time-shared processor. *Naval research logistics quarterly* 11, 1 (1964), 59–73.

[20] Thomas Neumann and Gerhard Weikum. 2010. The RDF-3X engine for scalable management of RDF data. *VLDB J.* 19, 1 (2010), 91–113. https://doi.org/10.1007/s00778-009-0165-y

[21] Jorge Pérez, Marcelo Arenas, and Claudio Gutiérrez. 2009. Semantics and complexity of SPARQL. *ACM Trans. Database Syst.* 34, 3 (2009), 16:1–16:45. https://doi.org/10.1145/1567274.1567278

[22] Axel Polleres, Maulik R. Kamdar, Javier D. Fernández, Tania Tudorache, and Mark A. Musen. 2018. A More Decentralized Vision for Linked Data. In *Proceedings of the 2nd Workshop on Decentralizing the Semantic Web co-located with the 17th International Semantic Web Conference, DeSemWeb@ISWC 2018, Monterey, California, USA, October 8, 2018.*

[23] Muhammad Saleem, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo. 2015. FEASIBLE: A Feature-Based SPARQL Benchmark Generation Framework. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*. 52–69. https://doi.org/10.1007/978-3-319-25007-6_4

[24] Max Schmachtenberg, Christian Bizer, and Heiko Paulheim. 2014. Adoption of the Linked Data Best Practices in Different Topical Domains. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I (Lecture Notes in Computer Science)*, Vol. 8796. Springer, 245–260. https://doi.org/10.1007/978-3-319-11964-9_16

[25] Michael Schmidt, Michael Meier, and Georg Lausen. 2010. Foundations of SPARQL query optimization. In *Database Theory - ICDT 2010, 13th International Conference, Lausanne, Switzerland, March 23-25, 2010, Proceedings*. ACM, 4–33. https://doi.org/10.1145/1804669.1804675

[26] Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt. 2011. FedX: Optimization Techniques for Federated Query Processing on Linked Data. In *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I (Lecture Notes in Computer Science)*, Vol. 7031. Springer, 601–616. https://doi.org/10.1007/978-3-642-25073-6_38

[27] Ruben Verborgh, Miel Vander Sande, Olaf Hartig, Joachim Van Herwegen, Laurens De Vocht, Ben De Meester, Gerald Haesendonck, and Pieter Colpaert. 2016. Triple Pattern Fragments: A low-cost knowledge graph interface for the Web. *J. Web Sem.* 37-38 (2016), 184–206. https://doi.org/10.1016/j.websem.2016.03.003

[28] Cathrin Weiss, Panagiotis Karras, and Abraham Bernstein. 2008. Hexastore: sextuple indexing for semantic web data management. *PVLDB* 1, 1 (2008), 1008–1019. https://doi.org/10.14778/1453856.1453965

[29] Annita N. Wilschut and Peter M. G. Apers. 1993. Dataflow Query Execution in a Parallel Main-memory Environment. *Distributed and Parallel Databases* 1, 1 (1993), 103–128. https://doi.org/10.1007/BF01277522

Hack for Hire: Exploring the Emerging Market for Account Hijacking

Ariana Mirian
University of California, San Diego
amirian@cs.ucsd.edu

Joe DeBlasio*
University of California, San Diego
jdeblasio@cs.ucsd.edu

Stefan Savage
University of California, San Diego
savage@cs.ucsd.edu

Geoffrey M. Voelker
University of California, San Diego
voelker@cs.ucsd.edu

Kurt Thomas
Google
kurtthomas@google.com

ABSTRACT

Email accounts represent an enticing target for attackers, both for the information they contain and the root of trust they provide to other connected web services. While defense-in-depth approaches such as phishing detection, risk analysis, and two-factor authentication help to stem large-scale hijackings, targeted attacks remain a potent threat due to the customization and effort involved. In this paper, we study a segment of targeted attackers known as "hack for hire" services to understand the playbook that attackers use to gain access to victim accounts. Posing as buyers, we interacted with 27 English, Russian, and Chinese blackmarket services, only five of which succeeded in attacking synthetic (though realistic) identities we controlled. Attackers primarily relied on tailored phishing messages, with enough sophistication to bypass SMS two-factor authentication. However, despite the ability to successfully deliver account access, the market exhibited low volume, poor customer service, and had multiple scammers. As such, we surmise that retail email hijacking has yet to mature to the level of other criminal market segments.

CCS CONCEPTS

• **Security and privacy** → *Multi-factor authentication*; *Phishing*; *Social aspects of security and privacy*.

KEYWORDS

email security; hacking; phishing; account compromise

ACM Reference Format:
Ariana Mirian, Joe DeBlasio, Stefan Savage, Geoffrey M. Voelker, and Kurt Thomas. 2019. Hack for Hire: Exploring the Emerging Market for Account Hijacking. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313489

1 INTRODUCTION

It has long been understood that email accounts are the cornerstone upon which much of online identity is built. They implicitly provide a root of trust when registering for new services and serve as the backstop when the passwords for those services must be reset. As such, the theft of email credentials can have an outsized impact—exposing their owners to fraud across a panoply of online accounts.

Unsurprisingly, attackers have developed (and sell) a broad range of techniques for compromising email credentials, including exploiting password reuse, access token theft, password reset fraud and phishing among others. While most of these attacks have a low success rate, when applied automatically and at scale, they can be quite effective in harvesting thousands if not millions of accounts [27]. In turn, email providers now deploy a broad range of defenses to address such threats—including challenge questions to protect password reset actions, mail scanning to filter out clear phishing lures, and two-factor authentication mechanisms to protect accounts against password theft [7–9]. Indeed, while few would claim that email account theft is a solved problem, modern defenses have dramatically increased the costs incurred by attackers and thus reduce the scale of such attacks.

However, while these defenses have been particularly valuable against large-scale attacks, targeted attacks remain a more potent problem. Whereas attackers operating at scale expect to extract small amounts of value from each of a large number of accounts, targeted attackers expect to extract large amounts of value from a small number of accounts. This shift in economics in turn drives an entirely different set of operational dynamics. Since targeted attackers focus on specific email accounts, they can curate their attacks accordingly to be uniquely effective against those individuals. Moreover, since such attackers are unconcerned with scale, they can afford to be far nimbler in adapting to and evading the defenses used by a particular target. Indeed, targeted email attacks—including via spear-phishing and malware—have been implicated in a wide variety of high-profile data breaches against government, industry, NGOs and universities alike [10, 12, 13, 31].

While such targeted attacks are typically regarded as the domain of sophisticated adversaries with significant resources (e.g., state actors, or well-organized criminal groups with specific domain knowledge), it is unclear whether that still remains the case. There is a long history of new attack components being developed as vertically integrated capabilities within individual groups and then evolving into commoditized retail service offerings over time (e.g., malware authoring and distribution, bulk account registration, AV testing, etc. [27]). This transition to commoditization is commonly driven by both a broad demand for a given capability and the ability for specialists to reduce the costs in offering it at scale.

*Author DeBlasio has since joined Google.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313489

In this paper, we present the first characterization of the *retail* email account hacking market. We identified dozens of underground "hack for hire" services offered online (with prices ranging from $100 to $500 per account) that purport to provide targeted attacks to all buyers on a retail basis. Using unique online buyer personas, we engaged directly with 27 such account hacking service providers and tasked them with compromising victim accounts of our choosing. These victims in turn were "honey pot" Gmail accounts, operated in coordination with Google, and allowed us to record key interactions with the victim as well as with other fabricated aspects of their online persona that we created (e.g., business web servers, email addresses of friends or partner). Along with longitudinal pricing data, our study provides a broad picture of how such services operate—both in their interactions with buyers and the mechanisms they use (and do not use) to compromise victims.

We confirm that such hack for hire services predominantly rely on social engineering via targeted phishing email messages, though one service attempted to deploy a remote access trojan. The attackers customized their phishing lures to incorporate details of our fabricated business entities and associates, which they acquired either by scraping our victim persona's website or by requesting the details during negotiations with our buyer persona. We also found evidence of re-usable email templates that spoofed sources of authority (Google, government agencies, banks) to create a sense of urgency and to engage victims. To bypass two-factor authentication, the most sophisticated attackers redirected our victim personas to a spoofed Google login page that harvested both passwords as well as SMS codes, checking the validity of both in real time. However, we found that two-factor authentication still proved an obstacle: attackers doubled their price upon learning an account had 2FA enabled. Increasing protections also appear to present a deterrent, with prices for Gmail accounts at one service steadily increasing from $125 in 2017 to $400 today.

As a whole, however, we find that the commercialized account hijacking ecosystem is far from mature. Just five of the services we contacted delivered on their promise to attack our victim personas. The others declined, saying they could not cover Gmail, or were outright scams. We frequently encountered poor customer service, slow responses, and inaccurate advertisements for pricing. Further, the current techniques for bypassing 2FA can be mitigated with the adoption of U2F security keys. We surmise from our findings, including evidence about the volume of real targets, that the commercial account hijacking market remains quite small and niche. With prices commonly in excess of $300, it does not yet threaten to make targeted attacks a mass market threat.

2 METHODOLOGY

In this section we describe our methodology for creating realistic, but synthetic, victims to use as targets, the infrastructure we used to monitor attacker activity, and the services we engaged with to hack into our victim email accounts. We also discuss the associated legal and ethical issues and how we addressed them in our work.

2.1 Victims

We created a unique victim persona to serve as the target of each negotiation with a hack for hire service. We never re-used victim personas among services, allowing us to attribute any attacks deployed against the persona back to the service we hired. In creating victim personas, we spent considerable effort to achieve three goals:

- *Victim verisimilitude.* We created synthetic victims that appeared sufficiently real that the hacking services we hired would treat them no differently from other accounts that they are typically hired to hack into.
- *Account non-attributability.* We took explicit steps to prevent attackers from learning our identities while we engaged with them as buyers, when they interacted with us as victims, and even if they successfully gained access to a victim email account.
- *Range of attacker options.* We did not know a priori what methods the hacking services would use to gain access to victim email accounts. Since there are many possibilities, including brute-force password attacks, phishing attacks on the victim, and malware-based attacks on the victim's computers, we created a sufficiently rich online presence to give attackers the opportunity to employ a variety of different approaches.

The remainder of this section details the steps we took to achieve these goals when creating fictitious victims, the monitoring infrastructure we used to capture interactions with our fake personas, and the selection of "hack for hire" services we engaged with.

Victim Identities. Each victim profile consisted of an email address, a strong randomly-generated password, and a name. While each of our victims 'lived' in the United States, in most cases we chose popular first and last names for them in the native language of the hacking service, such as "Natasha Belkin" when hiring a Russian-language service.[1] The email address for the victim was always a Gmail address related to the victim name to further reinforce that the email account was related to the victim (e.g., `natasha.r.belkin@gmail.com`). We loaded each email account with a subset of messages from the Enron email corpus to give the impression that the email accounts were in use [5]. We changed names and domains in the Enron messages to match those of our victim and the victim's web site domain (described below), and also changed the dates of the email messages to be in this year.

Each victim Gmail account used SMS-based 2-Factor Authentication (2FA) linked to a unique phone number.[2] As Gmail encourages users to enable some form of 2FA, and SMS-based 2FA is the most utilized form, configuring the accounts accordingly enabled us to explore whether SMS-based 2FA was an obstacle for retail attackers who advertise on underground markets [1] (in short, yes, as discussed in detail in Section 3.4).

Online Presence. For each victim, we created a unique web site to enhance the fidelity of their online identity. These sites also provided an opportunity for attackers to attempt to compromise the web server as a component of targeting the associated victim (server attacks did not take place). Each victim's web site represented either a fictitious small business, a non-governmental organization (NGO), or a blog. The sites included content appropriate for its purported function, but also explicitly provided contact information (name and email address) of the victim and their associates (described

[1] These example profile details are from a profile that we created, but in the end did not need to use in the study.
[2] These phone numbers, acquired via prepaid SIM cards for AT&T's cellular service, were also non-attributable and included numbers in a range of California area codes.

shortly). We hosted each site on its own server (hosted via third-party service providers unaffiliated with our group) named via a unique domain name. We purchased these domain names at auction to ensure that each had an established registration history (at least one year old) and the registration was privacy-protected to prevent post-sale attribution to us (privacy protection is a common practice; one recent study showed that 20% of .com domains are registered in this fashion [17]). The sites were configured to allow third-party crawling, and we validated that their content had been incorporated into popular search engine indexes before we contracted for any hacking services. Finally, we also established a passive Facebook profile for each victim in roughly the style of Cristofaro et al. [3]. These profiles were marked 'private' except for the "About Me" section, which contained a link to the victim's web site.[3]

Associate Identity. In addition to the victim identity, we also created a unique identity of an associate to the victim such as a spouse or co-worker. The goal with creating an associate was to determine whether the hacking services would impersonate the associate when attacking the victim (and some did, as detailed in Section 3.2) or whether they would use the associate email account as a stepping stone for compromising the victim email account (they did not). Similar to victim names, we chose common first and last names in the native language of the hacking service. Each victim's web site also listed the name and a Gmail address of the associate so that attackers could readily discover the associate's identity and email address if they tried (interestingly, most did not try as discussed in Section 3.2). Finally, if the victim owned their company, we also included a company email address on the site (only one attack used the company email address in a phishing lure).

Buyer Identity. We interacted anonymously with each hack for hire service using a unique buyer persona. When hiring the same service more than once for different victims, we used distinct buyer personas so that each interaction started from scratch and was completely independent. In this role, we solely interacted with the hacking services via email (exclusively using Gmail), translating our messages into the native languages of the service when necessary.

Many hacking services requested additional information about the victim from our buyers, such as names of associates, to be able to complete the contract. Since we made this information available on the victim web sites, we resisted any additional requests for information to see if the services would make the effort to discover this information themselves, or if services would be unable to complete the contract without it (Section 3.1).

2.2 Monitoring Infrastructure

Email Monitoring. For each Gmail account, we monitored activity on the account by using a modified version of a custom Apps Script shared by Onaolapo et al. [23]. This script logged any activity that occurs within the account, such as sending or deleting email messages, changing account settings, and so on (Section 3.6 details what attackers did after gaining access to accounts). The script then uploaded all logged activity to a service running in Google's public cloud service (Google App Engine) as another level-of-indirection to hide our infrastructure from potential exposure to attackers. Since the script runs from within the Gmail account, it is possible in principle for an attacker to discover the script and learn where the script is reporting activity to, though only after a successful attack. We found no evidence that our scripts were detected.

Login Monitoring. In addition to monitoring activity from within the accounts, the accounts were also monitored for login activity

[3]None of the service providers we contracted with appeared to take advantage of the Facebook profile, either by visiting the victim's web site via this link or communicating with the victim via their Facebook page.

Service	Price	Lang	Prepay	Payment	Respond	Attack
A.1	$229	RU	50%	Qiwi	Yes	Yes
A.2	$229	RU	50%	Qiwi	Yes	Yes
A.3	$458	RU	50%	Qiwi	Yes	Yes
B.1	$380	RU	No	Webmoney, Yandex	Yes	Yes
B.2	$380	RU	No	Webmoney, Yandex	Yes	Yes
C.1	$91	RU	No	Bitcoin	Yes	Yes
C.2	$91	RU	No	–	Yes	Yes
D.1	$76	RU	No	–	Yes	Yes
E.1	$122	RU	No	–	Yes	Yes
E.2	$122	RU	No	–	Yes	No
D.2	$76	RU	No	–	Yes	No
F	$91	RU	No	–	Yes	No
G	$91	RU	No	–	Yes	No
H.1	$152	RU	No	Webmoney	Yes	No
H.2	$152	RU	No	Webmoney	Yes	No
J	–	EN	–	–	Yes	No
K	$200–300	EN	Yes	Bitcoin	Yes	No
L	$152	RU	No	–	Yes	No
M	$84	RU	No	–	Yes	No
N	$69	RU	No	Webmoney, Yandex	Yes	No
O	–	RU	No	Webmoney, Yandex	Yes	No
P	$305	RU	No	–	Yes	No
Q	$46	RU	Yes†	–	Yes	No
R	$100	EN	No	–	No	No
S	$400–500	EN	50%	–	No	No
T	$95 or 113	EN	No	Bitcoin, Credit Card	No	No
U	$98	RU	No	Webmoney	No	No
V	$152	RU	No	Webmoney, Yandex, Qiwi	No	No
W	$152	RU	No	–	No	No
X	$152	RU	No	Webmoney, Yandex	No	No
Y	$23 – $46	RU	No	–	No	No
Z	$61	RU	No	–	No	No
AA	$46	RU	No	–	Yes	No
BB	–	CN	–	–	No	No

Table 1: We contacted 27 hacking services attempting to hire them to hack 34 different victim Gmail accounts. We communicated with the services in the language in which they advertised, translating when necessary. The prices were advertised in their native currency, and we normalized them to USD for ease of comparison. (Yes†: for first-time customers.)



Figure 1: An online advertisement for Gmail hacking services. We remove any identifiable information and translate the page from Russian to English.

by Google's system-wide logging mechanisms. Google's monitoring, shared with us, reported on login attempts and whether they were successful, when attackers were presented with a 2FA challenge, and whether they were able to successfully respond to the challenge (Section 3.4). These monitoring logs also include the infrastructure and devices used to make login attempts, which Google used to identify other Gmail accounts attacked by these services (Section 4.1).

Phone Monitoring. As described earlier, each victim account was associated with a unique cell number (used only for this purpose) which was configured in Gmail to be the contact number for SMS-based 2FA. To capture attacks against these phone numbers or notifications from Google (e.g., for 2FA challenges or notification of account resets) we logged each SMS message or phone call received.

Web Site Monitoring. To monitor activity on the web sites associated with the victims, we recorded HTTP access logs (which included timestamp, client IP, user agent, referrer information, and path requested). For completeness, we also recorded full packet traces of all incoming traffic to the target server machines in case there was evidence of attacker activity outside of HTTP (e.g., attempts to compromise the site via SSH). Overall, we found no evidence of attackers targeting our web sites.

2.3 Hacking Services

Recruitment. We identified hacking services through several mechanisms: browsing popular underground forums, searching for hacking services using Google search, and contacting the abuse teams of several large Internet companies. We looked for services that specifically advertised the ability to hack into Gmail accounts. While we preferred services that explicitly promised the passwords of targeted accounts, we also engaged with services that could instead provide an archive of the victim's account. Figure 1 shows an example service advertisement (one we did not purchase from).

When hiring these services, we followed their instructions for how to contact them. Typically, interactions with the services consisted of a negotiation period, focused on a discussion of what they would provide, their price, and a method of payment. The majority of the services were non-English speaking. In these cases, we used a native speaker as a translator when needed. We always asked whether they could obtain the password of the account in question as the objective, and always offered to pay in Bitcoin. If the sellers did not want to use Bitcoin, we used online conversion services to convert into their desired currency (the minority of cases). Interestingly, only a handful of services advertised Bitcoin as a possible payment vector, though many services were generally receptive towards using Bitcoin when we mentioned it.

Table 1 summarizes the characteristics of all services that we contacted, which we anonymize so that our work does not advertise merchants or serve as a performance benchmark. In total, we reached out to 27 different services and attempted to hire them to hack 34 unique victim Gmail accounts. When a service successfully hacked into an account, we later hired them again (via another unique buyer persona) with a different victim to see if their methods changed over time (we denote different purchases from the same service by appending a number after the letter used to name the service).

Service reliability. Of the twenty-seven services engaged, ten refused to respond to our inquiries. Another twelve responded to our initial request, but the interactions did not lead to any attempt on the victim account. Of these twelve, nine refused up front to take the contract for various reasons, such as claiming that they no longer hacked Gmail accounts contrary to their contemporary advertisements. The remaining three appear to be pure scams (i.e., they were happy to take payment, but did not perform any service in return). One service provided a web-based interface for entering the target email address, which triggered an obviously fake progress bar followed by a request for payment.[4] Another service advertised payment on delivery, but after our initial inquiry, explained that they required full prepayment for first-time customers. After payment, they responded saying that they had attempted to get into the account but could not bypass the 2FA SMS code without further payment. They suggested that they could break into the mobile carrier, intercept the SMS code, and thus break into the Gmail account. We paid them, and, after following up a few times, heard nothing further from them. During this entire exchange, we did not see a single login attempt on the victim's Gmail account from the hacking service. The third site similarly required pre-payment and performed no actions that we could discern.

Finally, five of the services made clear attempts (some successful, some unsuccessful) to hack into eleven victim accounts. We focus on these services going forwards.

Pricing. The cost for hiring the hacking services often varied significantly between the advertised price and the final amount we paid. Table 2 shows a breakdown of the price differences during engagement with the hacking services we successfully hired. The table shows the service, the purported price for that service from their online advertisement, the initially agreed upon price for their services, and then any price increase that may have incurred during the attack period. When services failed to hack into the account, they did not request payment. Several factors influenced the changes in prices, in particular the use of 2FA on the accounts (Section 6).

[4] We did not pay them since we would learn nothing more by paying.

Service	Advertised	Discussed	Final
A.1	$230	$230	$307
A.2	$230	$230 - $307	Failed
A.3	$460	$460	$460
B.1	$383	$383	Failed
B.2	$383	$383	$383
C.1	$92	$102	$100
C.2	$92	–	Failed
D.1	$77	$184	Failed
D.2	$77	$184	Failed
E.1	$123	$383 - $690	$383
E.2	$123	$690	Failed

Table 2: The changes in negotiated prices when advertised, when initially hired, and when finally successful at hacking into victim Gmail accounts. All prices were originally in rubles, but are converted to USD for easier comparison.

As a rule, we always paid the services, even when they requested additional money, and even when we strongly suspected that they might not be able to deliver when they asked for payment up front.[5] Our goal was to ultimately discover what each service would actually do when paid.

2.4 Legal and Ethical Issues

Any methodology involving direct engagement with criminal entities is potentially fraught with sensitivities, both legal and ethical. We discuss both here and how we addressed them.

There are two legal issues at hand in this study: unauthorized access and the terms of service for account creation and use. Obtaining unauthorized access to third-party email accounts is unlawful activity in most countries and in the United States is covered under 18 USC 1030, the Computer Fraud and Abuse Act (CFAA). Contracting for such services, as we did in this study, could constitute aiding and abetting or conspiracy if the access was, in fact, unauthorized. However, in this study, the email accounts in question are directly under our control (i.e., we registered them), and since we are acting in coordination with the account provider (Google), our involvement in any accesses was explicitly authorized. The other potential legal issue is that this research could violate Google's terms of service in a number of ways (e.g., creating fake Gmail accounts). We addressed this issue by performing our study with Google's explicit permission (including a written agreement). Both our institution's general counsel and Google's legal staff were appraised of the study, its goals, and the methods employed before the research began.

This study is not considered human subjects research by our Institutional Review Board because, among other factors, it focuses on measuring organizational behaviors and not those of individuals. Nevertheless, outside traditional human subjects protections, there are other ethical considerations that informed our approach. First, by strictly using fictitious victims, associates and web sites, we minimized the risk to any real person resulting from the account hacking contracted for in this study. Second, to avoid indirect harms resulting from implicitly advertising for such services (at least the effective ones), we made the choice to anonymize the names of each service. Finally, to minimize our financial contributions to a potentially criminal ecosystem, we limited the number of purchases to those needed to establish that a service "worked" and, if so, that its modus operandi was consistent over time.

[5]The one exception to this rule is the aforementioned service whose automated web site immediately told us they had hacked the site when all evidence was to the contrary.

3 HACK FOR HIRE PLAYBOOK

Our study characterizes the operational methods that hack for hire services employ when making a credible attempt to hijack our victim personas. We limit our analysis exclusively to the five services where the attackers made a detectable attempt to gain access to our victim account. We note that the ultimate "success" of these attacks is partially dependent on our experimental protocol: in some cases, we supplied 2FA SMS codes to phishing attacks or installed a provided executable, while in other cases, we avoided such actions to see if the attackers would adapt.

3.1 Attacks Overview

We present a high-level breakdown of each hack for hire service's playbook in Table 3. Four of the five services relied on phishing, while just one relied on malware. In all cases, attacks began with an email message to our victim persona's Gmail address. We never observed brute force login attempts, communication with a victim's Facebook account, or communication to our associate personas of any kind.[6] On average, attackers would send roughly 10 email messages over the course of 1 to 25 days—effectively a persistent attack until success. All of the services but one were able to bypass Gmail spam filtering (though to varying degrees of success) until at least one of their messages appeared in our victim's inbox. However, this outcome is expected: since these are targeted attackers with more focused motivation, they have strong incentives to adapt to phishing and spam defenses to ensure that their messages arrive in the victim's inbox. For example, attackers can create honeypot accounts of their own to test and modify their techniques, thereby ensuring a higher success rate; unlike their high-volume counterparts, targeted attackers only produce a modest number of examples and thus may pass "under the radar" of defenses designed to recognize and adapt to new large-scale attacks.

3.2 Email Lures

Each email message contained a lure impersonating a trusted associate or other source of authority to coerce prospective victims into clicking on a link. We observed five types of lures: those impersonating an associate persona, a stranger, a bank, Google, or a government authority. The associate lures tempted the user to click on an "image" for the victim's associate (using the personal connection as a sense of safety), while the Google, bank, and government lures conveyed a sense of urgency to induce a user to click on the link. Figure 2 shows a sample Google lure that mimics a real warning used by Google about new device sign-ins. Such lures highlight the challenge of distinguishing authentic communication from service providers, whereby attackers repurpose potentially common experiences to deceive victims into taking an unsafe action.

Attackers cycled through multiple lures over time in an apparent attempt to find any message that would entice a victim into clicking

[6]In practice, a victim's password may be exposed in a third-party data breach. Our use of synthetic identities prevents this as a potential attack vector.

Service	Method	Lure	Inbox or Spam	Promised goods	Requested	Success
A.1	Phishing	A, G, S	Inbox	Archive	–	Y
A.2	Phishing	A, G, S	Inbox	Archive	Victim and associate name, phone number	N
A.3	Phishing	A, G, S	Inbox	Archive	Victim and associate name, phone number	Y
B.1	Phishing	B	Inbox, Spam	Password	–	N
B.2	Phishing	A, G, V	Inbox, Spam	Password	Victim name, associate name/email, phone number*	Y
C.1	Phishing	G	Inbox	Password	–	Y
C.2	Phishing	G	Inbox, Spam	Password	–	N
D.1	Malware	V	Spam	Password	Victim name and occupation	N
E.1	Phishing	G, V	Inbox, Spam	Password	–	Y

Table 3: Overview of attack scenarios per service. Lure emails include impersonating an associate (A), bank (B), Google (G), government (V), or a stranger (S). In the event a service indicated they could not succeed without additional information, we indicate what details they requested. In one case (marked *), this was only for the second attempt.

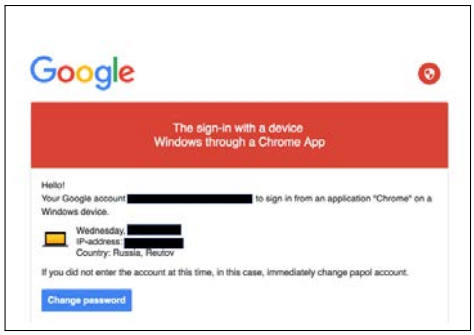


Figure 2: An example Google lure mimicking a real warning that Gmail will send to users. Identifying information removed and translated to English.

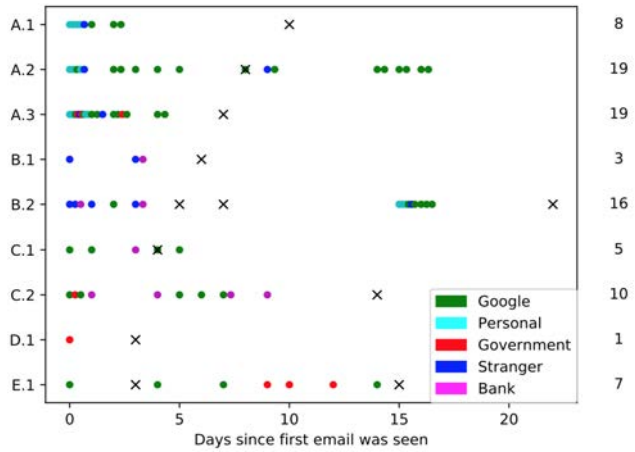


Figure 3: Different types of lures used by services that attempted to access a victim account. An 'X' marks when we clicked on a link in a message sent to a victim. Numbers on the right denote the total number of emails sent by a service.

on a link. Figure 3 shows the elapsed time since attackers sent their first email message to our victim account, the type of lure they used for each message, and when we clicked on the lure acting as a victim (potentially halting further attempts). Each row corresponds to one attack on a victim, and the x-axis counts the number of days since the service sent their first message to the victim. The numbers on the right y-axis show the number of messages sent by the service to the victim. The most popular lure mimicked Google, followed by associates and then lures from strangers.

Of the five services, two relied on personalized messages when communicating with four victim personas. In three of these cases, the service asked for additional details upfront about the victim persona during negotiation. Only service A.1 was able to construct personal lures without requesting assistance from the buyer, finding the details from the victim persona's website. The extent of personalization was limited, though, consisting either of mimicking the victim persona's company or their associate's personal email address. No additional branding was lifted from our web sites.

3.3 Phishing Landing Pages

All services but one relied on phishing as their attack vector. Once we clicked on the links sent to the victim personas, we were redirected to a spoofed Google login page that requested the credentials from the victim. Table 4 lists the different attack attempts and the degree to which attackers tried to spoof a Google domain, use HTTPS, or mask URLs from a crawler via multiple redirects. All services but one used "combo" domain name squatting [14] with the keyword 'google' in the URL, presumably to trick the victim into thinking that the URL was a real Google subdomain. Services A.2 and B.2 used the same fully qualified domain name for the phishing landing page, suggesting that they share a business relationship (i.e., they may both be value-added resellers for the same phishing page service). Long-lived, reused domains suggest that they are valuable and perhaps relatively costly to acquire.

All but one service tried to obscure the URL to their phishing page with at least one layer of redirection. (The exception was the link in the phishing message from C.2, which redirected to an error page on a Russian hosting service indicating that the page had been taken down.) The redirection URLs seemed to be one-time use URLs, since we were not able to visit them after the attack executed and did not see repeat redirection URLs in any of the attacks. One-time use URLs are attractive for attackers because they can greatly

Service	'google' in URL?	HTTPS	# redirects to phishing page
A.1	Yes	Yes	2
A.2	Yes	Yes	2
A.3	Yes	Yes	2
B.1	Yes	No	1
B.2.1	Yes	No	1
B.2.2	Yes	No	1
B.2.3	Yes	Yes	2
C.1	No	No	0
C.2	NA	NA	NA
D.1	NA	NA	NA
E.1.1	Yes	Yes	1
E.1.2	Yes	Yes	2

Table 4: For services that attempted to hack a victim account, we show whether Google was used in the phishing URL, whether the phishing page used HTTPS, and the number of redirects to the phishing page. We include separate rows for the services that sent multiple messages (services B and E).

complicate investigating attacks after the fact or sharing attack information among organizations.

Figure 4 shows an example page flow used by one hacking service. We always entered the Gmail credentials of the victim to see how the hacking attempt would progress. After collecting the password, all but one of the hacking services would redirect to a new screen which asked for the 2FA code that the victim had just received on their phone from Google.

Six of the nine hacking attempts captured the password from the phishing page and then immediately tried to use it to login to the victim's account (as verified with our Gmail access logging). Due to the similar behavior and speed at which these logins occurred, we believe that most of these services used an automated tool, similar to Evilginx [6], for this step.

Moreover, three of five of these attacks captured the necessary information in one session visiting the phishing pages. This sophistication suggests that attackers can readily adapt any additional information requested by Google as a secondary factor. Since our study, Google launched additional protections at login to prevent automated access attempts [26]. However, hardware security keys remain the best protection mechanism against phishing for users.

3.4 Live Adaptation

Services B.2 and E.1 exhibited phishing attacks that adapted over time to overcome obstacles. These services, once realizing that the account used 2FA, sent new phishing email messages with a different structure than the ones they sent previously. Service E.1, for example, initially used a phishing attack that only captured the Gmail password. When the service attempted to login, they were blocked by the 2FA prompt. The service then contacted our buyer persona asking for the victim's phone number. The victim's email account subsequently received more phishing messages in their inbox. Clicking on the link in the phishing messages led to a page that requested the 2FA code that was sent to the victim's phone. When we entered the 2FA code into the phishing page, the service was able to successfully login. This behavior indicates live testing

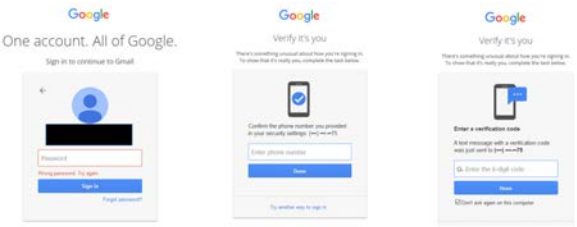


Figure 4: A service phishing flow, with identifiable information redacted. The flow is purposefully designed to mimic Gmail to trick the user into trusting the site.

of password validity, as the attackers were able to determine if the account had 2FA.

Service B.2 was similar to service E.1, but when they were blocked by the 2FA challenge they switched to phishing messages that looked exactly like the messages from service A. Upon collecting the password and the 2FA code that was sent to the phone number for the victim, the service was able to login.

3.5 Malware Attachments

Service D was the only service that attempted to hijack our victim account using malware. The attacker in this case sent just one email message to our victim persona—flagged as spam—that contained a link to a `rar` archive download (Gmail forbids executable attachments). The archive contained a sole executable file. We unpacked and ran the executable in an isolated environment, but to no effect. According to VirusTotal [32], it is a variant of TeamViewer (a commercial tool for remote system access) which would have enabled the attacker to hijack any existing web browsing sessions.

After no further visible activity, the service eventually contacted our buyer persona to say that they could not gain access to our victim account. We decided to hire them again via a different contract (and different buyer and victim personas) to see if the seller would adapt to Gmail's defenses. However, we observed no email messages from the attacker the second time around, even in our spam folder. The seller eventually responded stating that they could not gain access to our second persona's account. While this malware vector proved unsuccessful, the presence of remote access tools poses a significant risk for adaptation, as session hijacking would enable an attacker to bypass any form of two-factor authentication.

3.6 Post Compromise

For those services that did obtain our victims' credentials and 2FA codes, the attackers proceeded to sign in to each account and immediately removed all Google email notifications (both from the inbox and then trash) related to a new device sign-in. None changed the account password. We also observed that services A, B, and E removed the 2FA authentication and the recovery number from our victim accounts as well. Presumably they took these steps to regain access to the account at a later time without having to phish an SMS code again, but we did not see any service log back into the accounts after their initial login. However, these changes to the account settings could alert a real victim that their account had been hijacked, a discovery which the attackers are willing to risk.

Once accessed, all but one of the services abused a portability feature in Google services (Takeout) to download our victim account's email content and then provided this parcel to our buyer persona. One advantage of this approach is that it acquires the contracted deliverable in one step, thus removing risks associated with subsequent credentials changes, improvements in defenses, or buyer repudiation. Only service C avoided logging into our victim account and only provided the buyer persona with a password.[7] These findings highlight an emerging risk with data portability and regulations around streamlining access to user data. While intended for users, such capabilities also increase the ease with which a single account hijacking incident can expose all of a user's data to attackers. Since our study, Google has added additional step-up verification on sensitive account actions.

4 REAL VICTIMS & MARKET ACTIVITY

Based on our findings from the hack for hire process, we returned to the forums of the most successful attackers to understand their pricing for other services and how they attract buyers. Additionally, we present an estimate of the number of real victims affected by these services based on login traces from Google. Our findings suggest that the hack for hire market is quite niche, with few merchants providing hijacking capabilities beyond a handful of providers.

4.1 Victims Over Time

Of the 27 initial services we contacted, only three—services A, E, and B—could successfully login to our honeypot accounts. Google examined metadata associated with each login attempt and found that all three services rely on an *identical* automation process for determining password validity, bypassing any security check such as producing an SMS challenge, and downloading our honey account's email history. Whereas the email messages from the services had varied senders and delivery paths for each contracted campaign, this automation infrastructure remained stable despite eight months between our successive purchases. This stability in turn allowed Google to develop a signature allowing the retrospective analysis of all such login attempts from the three services in aggregate.

Over a seven-month period from March 16 to October 15, 2018, Google identified 372 accounts targeted by services A, B, and E. Figure 5 shows a weekly breakdown of activity. On an average week, these services attacked 13 targets, peaking at 35 distinct accounts per week. We caution these estimates are likely only lower bounds on compromise *attempts* as we cannot observe users who received a phishing URL, but did not click it (or otherwise did not enter their password on the landing page). Despite these limitations, the volume of activity from these hack for hire services is quite limited when compared to off-the-shelf phishing kits which impact over 12 million users a year [29]. Thus, we surmise that the targeted account hacking market is likely small when compared to other hacking markets, e.g., for malware distribution [11]. While the damage from these commercialized hacking services may be more potent, they are only attractive to attackers with particular needs.

Apart from the volume of these attacks, we also examine the sophistication involved. As part of its authentication process, Google

[7]The service demanded additional payment to defeat the 2FA, which we paid, at which point they stopped responding to our requests.

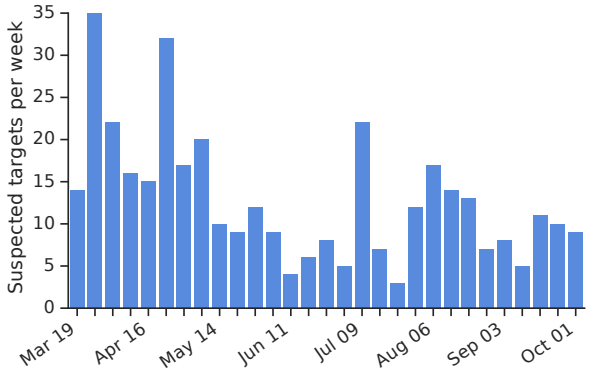


Figure 5: Weekly target accounts retroactively associated with hack for hire services.

may trigger a "challenge" for sign-in attempts from previously unseen devices or network addresses [20]. All of the hack for hire attempts triggered this detection. In 68% of cases, the attacker was forced to solve an SMS challenge, while in 19% of cases the attacker only had to supply a victim's phone number. The remaining 13% involved a scattering of other secondary forms of authentication. This layered authentication approach provides better security when compared to passwords alone, with attackers only correctly producing a valid SMS code for 34% of accounts and a valid phone number in 52% of cases. These rates take into consideration repeated attacks: Google observed that attackers would attempt to access each account a median of seven times before they either succeeded or abandoned their efforts. As such, even though these attacks may be targeted, Google's existing account protections can still slow and sometimes stop attackers from gaining access to victim accounts.

4.2 Alternate Services and Pricing

While our investigation focused on Google—due in large part to our ethical constraints and abiding by Terms of Service requirements—the hack for hire services we engaged with also purport to break into multiple mail providers (Yahoo, Mail.ru, Yandex), social networks (Facebook, Instagram), and messaging apps (WhatsApp, ICQ, Viber). To provide a price comparison between offerings, in preparation for our study we performed a weekly crawl of the forum page or dedicated web site advertising each service starting in January 1, 2017. However, as detailed previously in Section 3, only a fraction of the services are authentic, and just three—services A, B, and C—had online prices that matched (or were close) to the final price we paid. We treat these as trusted sources of pricing information. We also include services E and D, but note their prices were higher than advertised. We exclude all other services as they failed to attack any of our victim personas.

We present a breakdown of pricing information as of October 10, 2018 in Table 5 for the five services that executed an attempt to access the accounts. Across all five services, Russian mail provider hacking (i.e., Mail.ru, Rambler and Yandex) was the cheapest, while other mail providers such as Gmail and Yahoo were more expensive. The cost of hacking a social media account falls in the middle of these two extremes.

Target	Service A	Service B	Service C	Service D*	Service E*
Mail.ru	$77	$77	$62	$54	$77
Rambler	$152	$108	$77	$77	$108
Yandex	$106	$108	$77	$77	$108
Gmail	$384	$385	$92	$77	Negotiable
Yahoo	$384	$231	$92	–	–
Facebook	$306	–	–	–	–
Instagram	$306	–	–	–	$231

Table 5: Purported prices to access various accounts, based on an October 10, 2018 snapshot. All prices USD, converted from rubles. An '*' indicates the service's advertised price was lower than the final payout requested.

Some services increased their prices over time. For services B and C, prices on the forums they advertise have been stable since we first began our monitoring. Only service A provided dynamic pricing, with rates increasing as shown in Figure 6. Since 2017, Gmail prices have steadily increased from $123 to $384, briefly peaking at $461 in February 2018. The advertised rates for targeting Yahoo accounts has largely tracked this same rate, while Facebook and Instagram were initially priced higher before settling at $307. We hypothesize that the price differences between services and the change in prices for a service over time are likely driven by both operational and economic factors. Thus, prices will naturally increase as the market for a specific service shrinks (reducing the ability to amortize sunk costs on back-end infrastructure for evading platform defenses) and also as specific services introduce more, or more effective, protection mechanisms that need to be bypassed (increasing the transactional cost for each hacking attempt).

4.3 Advertising & Other Buyers

As a final measure, we examined the forum advertisements each service used to attract buyers. Here, we limit our analysis to the five successful hack for hire services. Across seven underground forums, we identified two types of advertisements—pinned posts and banner ads—which require paying forum operators. Services A, B, and E, the three services that were able to bypass two-factor authentication, all had pinned posts on forums where this option was available. Only service A paid for banner advertisements on all of these forums. Together, this suggests that the services are profitable enough to continue advertising via multiple outlets. Additionally, these three services had verified accounts, indicating that a forum moderator had vetted the service stated. Further, services A, B, D, and E all stated they could work with a "guarantor", an escrow service proxying for payment between service and buyer to avoid fraud risks. Generally, feedback on the forum was positive, though we caution this may be biased due to the ability to delete posts and the difficulty in distinguishing between legitimate customers and virtual "shills". We avoid using forum posts as a count of purchases as most negotiation activity occurs via private messaging.

In addition to this qualitative search, we received an email advertisement from one of the services for upcoming changes to the service, which was sent to 44 other buyers as well (exposing their clientele's email addresses). The message was an announcement that the service now had a Telegram channel that was available

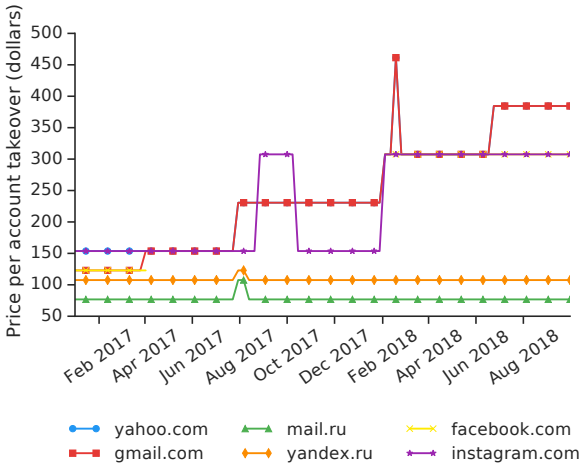


Figure 6: Monthly price that Service A charges across email and social network account providers. Over two years, the price per Gmail account increased from $123 to $384.

(with a link to the channel), and to join the channel to keep up to date with relevant news. The only response to that initial email message was another customer exclaiming their excitement for this new development. Of the 44 email addresses that were leaked, 23 were accounts with `mail.ru` or `yandex.ru`, 9 were Gmail addresses, and the rest were various other providers, like Tutanota, Protonmail, or iCloud. We were unable to find these buyers online, which indicates that they did not engage in forum postings, or used a burner (one-purpose use) email address. However, the concentration of Russian mail providers suggests that interest in the market may largely be geographically limited, potentially due to language barriers or culturally-biased demands for account hacking.

5 RELATED WORK

Phishing is a well studied, yet continuing concern in the security community. Sheng et al. studied the demographic of people who are susceptible to phishing attacks, and found that users 18–25 years of age are most likely to click on phishing messages [24]. Egleman et al. studied the effectiveness of phishing warnings and found they can be successful in preventing account hijacking [4]. Following this mode of thought, there are a variety of studies on effective anti-phishing training as well as the creation of a content-based approach to detect phishing sites [15, 25, 33]; in all of these studies, the percentage of users' susceptible to phishing emails dropped. Similarly, Zhang et al. evaluted anti-phishing tools, and found that many of them are not effective on new URLs and have exploits of their own. Oest et al. also studied the phishing ecosystem via an analysis of phishing kits, and developed a URL-based scheme to detect phishing URLs [22].

Account hijacking threats represent a spectrum that ranges from financially motivated, large-scale attacks to highly-targeted incidents motivated by political, personal, or financial incentives. Thomas et al. identified billions of credentials stolen via data breaches and millions of credentials stolen by phishing kits and keyloggers, with phishing posing the largest hijacking risk [29]. Once an account was accessed, hijackers searched for financial records or used

the account as a stepping stone to control connected online identities [2, 23]. While techniques such as risk-aware authentication [20] or two-factor authentication help protect against unsophisticated bulk attacks, the hack for hire outfits we studied were more dedicated, with attackers stealing SMS two-factor codes as part of their phishing pages to bypass the additional layers of security. Security keys would prevent this attack vector.

At the other end of the spectrum, Marczak et al. investigated government actors targeting political dissidents [19]. The hijackers in these cases relied on exploits or social engineering to have victims install commercial or off-the-shelf spyware to enable long-term monitoring of the victim's activities. Email was a common delivery mechanism, where attackers customized their lures to the NGOs where employees worked or to the human rights topics they were involved with [12, 16]. Given the risks involved here, researchers have focused on how to improve the security posture of at-risk users [18]. Compared to our work, we found more generalized lures that can work for any target (e.g., your account is running out of storage space or there was a security incident), while phishing was the most popular technique for gaining one-off access to a victim's account. Pressure on the hack for hire playbook, or wider-scale adoption of security keys, may cause them to move towards malware and thus mirror government attackers.

6 DISCUSSION AND CONCLUSION

When starting this study, we had very little knowledge of what to expect in terms of attacker methods, behaviors, and ability. At a high level, we find that the commercial account hijacking ecosystem is far from mature. When such attackers are successful, they can be potentially devastating to individuals. Yet, as an overall market it is not poised to cause widespread harm.

Retail account hijacking is a niche market. Many aspects of engaging with account hijackers strongly indicate that these services are a fledgling market:

- Most telling is that only five of the 27 services we contacted were willing to take our business, a third never responded to repeated requests as buyers, and some were outright fraudulent.
- Services have inconsistent and poor customer service. For example, three of the services charged significantly higher prices than their advertised price, and two services changed their initial prices while they were executing the hack. Moreover, customer service is slow and inconsistent in their communication with the buyer, sometimes taking more than a day to respond.
- Attackers showed little initiative. Most attacks made no effort to gather information independently about their victims. Of the nine attempts, only services A.1 and A.2 discovered additional information about the victim on their web sites, such as the name of their associate. The others, including different contracts within service A, would not attempt hacking the account without explicitly requesting additional information from the buyer.

In contrast, studies on markets for CAPTCHA solving [21], Twitter spam [30], and Google phone verified accounts [28] show that those services are quick to respond, and stable in their services and pricing. This differentiation between other underground service offerings and the retail hacking market suggests that account hacking may not be the main focus of these attackers, and may simply be a "side hustle" — a method to gain opportunistic income in addition to other activities they are more fully engaged in.

Services predominantly mount social engineering attacks using targeted phishing email messages. All but one of the nine attacks used targeted email phishing to hack into our Gmail accounts. The attackers customized their phishing messages using details that we made available about the businesses and associates of our fictitious victims. To prompt engagement with a victim, the phishing messages created a sense of urgency by spoofing sources of authority.

These methods are a subset of those used in other targeted attack ecosystems. In particular, in *addition* to targeted phishing (frequently much more tailored than any attacks mounted by the services we studied), government-targeted attackers use malware and long-term monitoring of victim behavior to gain access to the account, requiring much more overhead than phishing alone [19]. Indeed, although these two classes of attackers are superficially similar in focusing on individual users, they are distinct in most other respects including the nature of the populations they target, their resource investment per target, their goals upon compromising an account, and a far greater requirement for covert operations.

Two-factor authentication creates friction. Even though phishing can still be successful with 2FA enabled, our results demonstrate that 2FA adds friction to attacks. Various services said that they could not hack into the account without the victim's phone number, had to adapt to 2FA challenges by sending new phishing messages to bypass them, and one renegotiated their price (from $307 to $690) when they discovered that the account had 2FA protection. Based on these results, we recommend major providers encourage or require their user base to use a 2FA physical token

Minimal service differentiation. Even with a variety of services advertising in the account hijacking market, they have remarkably little differentiation in their methods and infrastructure. Four services sent very similar re-usable phishing email messages to their respective victims, and all services that successfully hacked our accounts used identical automation tools for determining password validity, bypassing security checks, and downloading victim data.

Gmail as a vantage point. Overall, our study indicates that the attack space against Gmail is quite limited. Since we focused on hiring services to hack solely into Gmail accounts, it is possible that the landscape of the commercialized hacking market would look much different when deployed against native email services such as `mail.ru` or `yandex.ru`.

ACKNOWLEDGEMENTS

We would like to thank Mikhail Kolomogorov for his help in translating emails, as well as Kirill Levchenko, Vector Guo Li, and Ivan Mikhailin. Our thanks also to Shawn Loveland who provided data that expanded the set of underground hack for hire services that we considered. We also thank Elie Bursztein, Angelika Moscicki, Tadek Pietraszek, and Kashyap Puranik for their feedback on our study. We are also grateful to our anonymous reviewers for their insightful feedback and suggestions. This work was supported in part by NSF grants CNS-1629973 and CNS-1705050, DHS grant AFRL-FA8750-18-2-0087, and by generous research, operational and/or in-kind support from Google.

REFERENCES

[1] Olabode Anise and Kyle Lady. State of the Auth: Experiences and Perceptions of Multi-Factor Authentication. *Duo Security*, https://duo.com/assets/ebooks/state-of-the-auth.pdf, November 2017. Accessed: 2018-10-22.

[2] Elie Bursztein, Borbala Benko, Daniel Margolis, Tadek Pietraszek, Andy Archer, Allan Aquino, Andreas Pitsillidis, and Stefan Savage. Handcrafted Fraud and Extortion: Manual Account Hijacking in the Wild. In *Proceedings of the 2014 ACM Internet Measurement Conference (IMC)*, Vancouver, BC, Canada, November 2014.

[3] Emilano De Cristofaro, Arik Friedman, Guillaume Jourjon, Mohamed Ali Kaafar, and M. Zubair Shafiq. Paying for Likes? Understanding Facebook Like Fraud Using Honeypots. In *Proceedings of the 2014 ACM Internet Measurement Conference (IMC)*, Vancouver, BC, Canada, November 2014.

[4] Serge Egelman, Lorrie Faith Cranor, and Jason Hong. You've been warned: An empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1065–1074, New York, NY, USA, 2008. ACM.

[5] Enron Email Dataset. https://www.cs.cmu.edu/~enron/. Accessed: 2018-11-03.

[6] Evilginx — Advanced Phishing with Two-factor Authentication Bypass. https://breakdev.org/evilginx-advanced-phishing-with-two-factor-authentication-bypass/. Accessed: 2018-10-22.

[7] Google. Add 2-Step Verification. https://support.google.com/a/answer/175197. Accessed: 2018-10-22.

[8] Google. Guard Against Targeted Attacks. https://support.google.com/a/answer/9010419. Accessed: 2018-10-22.

[9] Google. Verify a user's identity with a login challenge. https://support.google.com/a/answer/6002699. Accessed: 2018-10-22.

[10] Garrett M. Graff. DOJ Indicts 9 Iranians For Brazen Cyberattacks Against 144 US Universities. *Wired*, https://www.wired.com/story/iran-cyberattacks-us-universities-indictment/. Accessed: 2018-10-22.

[11] Chris Grier, Lucas Ballard, Juan Caballero, Neha Chachra añd Christian J. Dietrich, Kirill Levchenko, Panayiotis Mavrommatis, Da mon McCoy, Antonio Nappa, Andreas Pitsillidis, Niels Provos, M. Zub air Rafique, Moheeb Abu Rajab, Christian Rossow, Kurt Thomas, Vern Paxson, Stefan Savage, and Geoffrey M. Voelker. Manufacturing Compromise: The Emergence of Exploit-as-a-Service. In *Proceedings of the ACM Conference on Computer and Communications Security*, Raleigh, NC, October 2012.

[12] Seth Hardy, Masashi Crete-Nishihata, Katharine Kleemola, Adam Senft, Byron Sonne, Greg Wiseman, Phillipa Gill, and Ronald J Deibert. Targeted Threat Index: Characterizing and Quantifying Politically-Motivated Targeted Malware. In *Proceedings of the 23rd USENIX Security Symposium*, San Diego, CA, USA, August 2014.

[13] Ian Karambelas. Spear Phishing: The Secret Weapon Behind the Worst Cyber Attacks. *Cloudmark*, https://blog.cloudmark.com/2016/01/13/spear-phishing-secret-weapon-in-worst-cyber-attacks/, January 2016. Accessed: 2018-10-22.

[14] Panagiotis Kintis, Najmeh Miramirkhani, Charles Lever, Yizheng Chen, Roza Romero-Gomez, Nikolaos Pitropakis, Nick Nikiforakis, and Manos Antonakakis. Hiding in Plain Sight: A Longitudinal Study of Combosquatting Abuse. In *Proceedings of the 2017 ACM Conference on Computer and Communications Security (CCS)*, Dallas, TX, USA, October 2017.

[15] Ponnurangam Kumaraguru, Yong Rhee, Alessandro Acquisti, Lorrie Faith Cranor, Jason Hong, and Elizabeth Nunge. Protecting People from Phishing: The Design and Evaluation of an Embedded Training Email System. In *Proceedings of the 2007 Conference on Human Factors in Computing Systems (CHI)*, pages 905–914, San Jose, CA, USA, April 2007.

[16] Stevens Le Blond, Adina Uritesc, Cédric Gilbert, Zheng Leong Chua, Prateek Saxena, and Engin Kirda. A Look at Targeted Attacks Through the Lense of an NGO. In *Proceedings of the 23rd USENIX Security Symposium*, San Diego, CA, USA, August 2014.

[17] Suqi Liu, Ian Foster, Stefan Savage, Geoffrey M. Voelker, and Lawrence K. Saul. Who is .com? Learning to Parse WHOIS Records. In *Proceedings of the 2015 ACM Internet Measurement Conference (IMC)*, Tokyo, Japan, October 2015.

[18] William R Marczak and Vern Paxson. Social Engineering Attacks on Government Opponents: Target Perspectives. In *Proceedings of the 17th Privacy Enhancing Technologies Symposium (PETS)*, Minneapolis, MN, USA, July 2017.

[19] William R Marczak, John Scott-Railton, Morgan Marquis-Boire, and Vern Paxson. When Governments Hack Opponents: A Look at Actors and Technology. In *Proceedings of the 23rd USENIX Security Symposium*, San Diego, CA, USA, August 2014.

[20] Grzegorz Milka. Anatomy of Account Takeover. *Enigma*, https://www.usenix.org/node/208154, January 2018.

[21] Marti Motoyama, Kirill Levchenko, Chris Kanich, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. Re: CAPTCHAs: Understanding CAPTCHA-solving Services in an Economic Context. In *Proceedings of the 19th USENIX Security Symposium*, Washington, DC, USA, August 2010.

[22] Adam Oest, Yeganeh Safei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Gary Warner. Inside a Phisher's Mind: Understanding the Anti-phishing Ecosystem Through Phishing Kit Analysis. In *Proceedings of the 2018 APWG Symposium on Electronic Crime Research (eCrime)*, San Diego, CA, USA, September 2018.

[23] Jeremiah Onaolapo, Enrico Mariconti, and Gianluca Stringhini. What Happens After You Are Pwnd: Understanding the Use of Leaked Webmail Credentials in the Wild. In *Proceedings of the 2016 ACM Internet Measurement Conference (IMC)*, Santa Monica, CA, USA, November 2016.

[24] Steve Sheng, Mandy Holbrook, Ponnurangam Kumaraguru, Lorrie Faith Cranor, and Julie Downs. Who Falls for Phish?: A Demographic Analysis of Phishing Susceptibility and Effectiveness of Interventions. In *Proceedings of the 2019 Conference on Human Factors in Computing Systems (CHI)*, pages 373–382, Atlanta, GA, USA, April 2010.

[25] Steve Sheng, Bryant Magnien, Ponnurangam Kumaraguru, Alessandro Acquisti, Lorrie Faith Cranor, Jason Hong, and Elizabeth Nunge. Anti-Phishing Phil: The Design and Evaluation of a Game That Teaches People Not to Fall for Phish. In *Proceedings of the 3rd Symposium on Usable Privacy and Security (SOUPS)*, pages 88–99, July 2007.

[26] Jonathan Skelker. Announcing some security treats to protect you from attackers' tricks. https://security.googleblog.com/2018/10/announcing-some-security-treats- to.html, October 2018.

[27] Kurt Thomas, Danny Yuxing Huang, David Wang, Elie Bursztein, Chris Grier, Tom Holt, Christopher Kruegel, Damon McCoy, Stefan Savage, and Giovanni Vigna. Framing Dependencies Introduced by Underground Commoditization. In *Proceedings of the 2015 Workshop on the Economics of Information Security (WEIS)*, Delft, The Netherlands, June 2015.

[28] Kurt Thomas, Dmytro Iatskiv, Elie Bursztein, Tadek Pietraszek, Chris Grier, and Damon McCoy. Dialing Back Abuse on Phone Verified Accounts. In *Proceedings of the 2014 ACM Conference on Computer and Communications Security (CCS)*, pages 465–476, Scottsdale, AZ, USA, November 2014.

[29] Kurt Thomas, Frank Li, Ali Zand, Jacob Barrett, Juri Ranieri, Luca Invernizzi, Yarik Markov, Oxana Comanescu, Vijay Eranti, Angelika Moscicki, Daniel Margolis, Vern Paxson, and Elie Bursztein. Data Breaches, Phishing, or Malware?: Understanding the Risks of Stolen Credentials. In *Proceedings of the 2017 ACM Conference on Computer and Communications Security (CCS)*, Dallas, TX, USA, October 2017.

[30] Kurt Thomas, Damon McCoy, Chris Grier, Alek Kolcz, and Vern Paxson. Trafficking Fraudulent Accounts: The Role of the Underground Market in Twitter Spam and Abuse. In *Proceedings of the 22nd USENIX Security Symposium*, Washington, DC, USA, August 2013.

[31] Verizon. 2018 Data Beach Investigations Report. https://www.verizonenterprise.com/resources/reports/rp_DBIR_2018_Report_en_xg.pdf. Accessed: 2018-10-22.

[32] Virus Total. https://www.virustotal.com/#/home/upload. Accessed: 2018-10-22.

[33] Yue Zhang, Jason I. Hong, and Lorrie F. Cranor. CANTINA: A Content-based Approach to Detecting Phishing Web Sites. In *Proceedings of the 16th International Conference on World Wide Web (WWW)*, pages 639–648, May 2007.

Anomaly Detection in the Dynamics of Web and Social Networks Using Associative Memory

Volodymyr Miz, Benjamin Ricaud, Kirell Benzi, and Pierre Vandergheynst
Ecole Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
first.last@epfl.ch

ABSTRACT

In this work, we propose a new, fast and scalable method for anomaly detection in large time-evolving graphs. It may be a static graph with dynamic node attributes (e.g. time-series), or a graph evolving in time, such as a temporal network. We define an anomaly as a localized increase in temporal activity in a cluster of nodes. The algorithm is unsupervised. It is able to detect and track anomalous activity in a dynamic network despite the noise from multiple interfering sources.

We use the Hopfield network model of memory to combine the graph and time information. We show that anomalies can be spotted with good precision using a memory network. The presented approach is scalable and we provide a distributed implementation of the algorithm.

To demonstrate its efficiency, we apply it to two datasets: Enron Email dataset and Wikipedia page views. We show that the anomalous spikes are triggered by the real-world events that impact the network dynamics. Besides, the structure of the clusters and the analysis of the time evolution associated with the detected events reveals interesting facts on how humans interact, exchange and search for information, opening the door to new quantitative studies on collective and social behavior on large and dynamic datasets.

CCS CONCEPTS

• **Computing methodologies** → **Anomaly detection**; • **Information systems** → **Wrappers (data mining)**; **Web log analysis**; • **Theory of computation** → **Dynamic graph algorithms**.

KEYWORDS

Anomaly Detection, Dynamic Network, Graph Algorithm, Hopfield Network, Wikipedia, Web Logs Analysis

ACM Reference Format:
Volodymyr Miz, Benjamin Ricaud, Kirell Benzi, and Pierre Vandergheynst. 2019. Anomaly Detection in the Dynamics of Web and Social Networks Using Associative Memory. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313541

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313541

1 INTRODUCTION

In recent years, the Web has significantly affected the way people learn, interact in social groups, store and share information. Apart from being an essential part of modern life, social networks, online services, and knowledge bases generate a massive amount of logs containing traces of global online activity on the Web. Most of this data is related to the standard activity of the users. However, the larger these logs become, the harder it is to detect deviations from normal behavior in the network. Localization of these anomalies becomes even more difficult because of the continuous expansion and dynamic nature of these networks. Hence, in recent years, anomaly detection has become an important field of research focusing on this problem [8], [33].

In this paper, we present a scalable anomaly detection approach for spatio-temporal datasets such as user activity logs of web and social networks. The approach focuses on anomaly detection in the collective behavior of users in such networks. Our approach introduces the following novelties that distinguish it from the state-of-the-art.

1) The method bridges the gap between graph anomaly detection [1, 2, 33] and spatio-temporal data mining [3]. It uses the formalism of anomaly detection and extends it to spatio-temporal event analysis. Here, the *spatial component* is a graph of interconnected entities (e.g. web pages, users or documents), and the *temporal component* is the activity logs of these entities. A spatio-temporal event is a set of nodes that have an abnormal behavior during a period of time. Importantly, the nodes may not undergo a change at the same time. There can be a complex interaction between the nodes during an event when a change in one node can trigger changes in others. This complex spatio-temporal evolution is not covered in anomaly detection literature.

2) For each event or anomaly detected using our method, the model produces a rich set of spatio-temporal indicators rather than a single label or a probability of being an anomaly. In this sense, it is closer to the spatio-temporal data mining [3], where the purpose is to extract the anomalous events and keep as much information about them as possible for the sake of interpretability. Providing the description of detected anomalies to domain experts is a powerful feature of our data mining process. Our approach provides insights on the collective behavior in web and social networks, how the visitor activity evolves and propagates over the network. We illustrate that in the Experiment section and reveal new insights on the collective behavior of Wikipedia visitors.

3) We define the concept of *potential anomaly* that introduces a prior on the presence of an anomaly and enhances the scalability of the method. Indeed, in many anomaly detection applications and thanks to expert knowledge, the data can be separated into 2 parts;

one part contains potential anomalies, while the other contains non-anomalous samples. Our concept defines the separation in a rigorous manner and allows discarding non-anomalous samples. This step significantly reduces the amount of data to process. The concept of *potential anomaly* is general and can be used for other methods and applications.

Our approach can be applied to the data that has a structure depicted in Fig. 1. It consists in an attributed graph where the attributes of the nodes are time-series signals. It leverages the network structure of the given data to detect dynamic events or anomalies in collective behavior.

In the experiments, we use two spatio-temporal datasets. First, we build a network of Wikipedia articles and use visitor activity of each article, i.e. the number of visits per article per hour, as a node attribute. The static underlying network is the Wikipedia hyperlinks network. Two pages are connected if there is at least one hyperlink reference between them. Second, we transform the Enron email dataset into a temporal network where nodes correspond to email addresses of employees and edges reflect an email exchange between a couple of employees. The temporal activity attributes of the nodes is the number of emails sent per day by an employee.

In the results of our experiments, we demonstrate that we can extract anomalous patterns in collective behavior of users in web and social networks. The anomalies correspond to subgraphs containing nodes whose behavior deviates from the norm. For the Enron email dataset, these subgraphs are groups of employees having an increase in their email exchange during a short period of time corresponding to major events in the corporation. For the Wikipedia data, the subgraphs contain linked pages closely related to an event that triggered a sudden increase of visits during a short period of time. These clusters of anomalous nodes can then be used for more detailed investigation, as shown in Section 5.

The strength of our approach is that it provides a comprehensive description of detected anomalies. As a result, we are able to perform a thorough qualitative evaluation of our results. Although, it turns into a difficulty when a quantitative evaluation of the method is needed. It is not as straightforward as computing the accuracy of a classification. Therefore, we use alternative methods for validating of the results. First, we use Google Trends as a ground truth indicator of the anomalous activity of the visitors on the Web. We verify the detected anomalous events using the trending topics extracted from Google Trends. In the second case, the Enron email dataset contains ground truth, hence we use it to validate the detected anomalies.

2 RELATED WORK

Anomaly detection. A recent review of the emerging field of spatio-temporal data mining highlights the importance of anomaly detection techniques for the dynamic networks domain [3]. Due to the complex nature of the data, most existing approaches treat spatial and temporal components of the data independently [10, 25, 36]. There are several application-specific approaches for anomaly detection in video streams that use spatial and temporal information jointly [22, 24]. Lappas et al. [23] presented an approach for bursts identification in Twitter using spatial and temporal aspects of the data, although did not use the formalism of anomaly detection.

When dealing with the data from social networks, event detection is closely related to anomaly detection, therefore there exists a number of approaches that use spatio-temporal features of these datasets for collective behavior analysis. The category of Event detection in [33, Type 4] covers the case where all nodes of a subgraph contribute to the creation of an event at the same time. This is also the case in [2, Def. 4], where the authors defined it as a problem of the dynamic-graph anomaly detection. In addition, these reviews as well as [1], focus on dynamic graphs, where the graph structure evolves over time, whereas in our case, the structure of the graph is static.

Our method is developed for static graphs with a dynamic evolution of node attributes (time-series). Our definition of an anomaly is more general since we track a heterogeneous spatio-temporal pattern that can emerge in situations when the anomaly spread or propagate over the network. It can not be captured by a single subgraph. Within our framework, an anomaly is described by two components. First, *a graph pattern* that involves multiple nodes, possibly anomalous at different time steps. Second, *a temporal pattern*, a set of time-series, that reflects the anomaly evolution on each node. Tracking these types of patterns has a number of applications [3, 5, 16], although they have not yet been linked to the field of anomaly detection.

Enron email dataset. Several studies have investigated the Enron email dataset using different approaches. While many of them focus on email contents and perform analysis using natural language processing techniques, others focus on the network structure and anomaly detection. In [9], the authors explore the email communication network of the corporation. Wang et al. [35] adopt an anomaly detection framework to spot remarkable events. We compare our results to this latter reference to validate the accuracy of our detection.

Wikipedia dataset. There is a large number of studies on mining the visitor or editor activity on Wikipedia that are aimed at getting better insights on collective behavior and social interactions [34], [14], [20]. However, due to a large amount of data, the aforementioned studies are restricted to particular topics of interest and subsets of selected Wikipedia articles. For instance, only traumatic events such as attacks and bombings have been investigated in [12], [11] based on the Wikipedia edit activity data. Analyzing Wikipedia daily page views, Kanhabua et al. [20] investigated 5500 events from 11 categories such as aviation accidents, earthquakes, hurricanes, or terrorist attacks. Wikipedia hourly visits on the pages of celebrities were used to investigate the fame levels of tennis players [39]. These studies point out the high interest in this dataset and the increasing need for more systematic detection methods.

Mongiovi et al. [31, 32] provided the first investigation from an anomaly detection point of view, where the Wikipedia page counts data are combined with the graph of hyperlinks. However, they apply their method to a pre-selected subset of Wikipedia.

Due to the introduced concept of potential anomaly and the distributed implementation, our method can handle the full Wikipedia network and long-term visitor activity records.

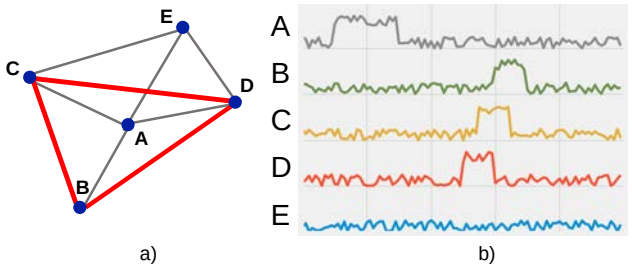


Figure 1: Spatio-temporal data structure combining a graph topology and time-series. a) Graph topology. Edges highlighted in red depict the spatial component of an anomalous spatio-temporal pattern. b) Time-series signals residing on the vertices of the graph. Signals associated to nodes B, C and D are correlated: an anomalous process is propagating from node D to B through C. This is an abstract illustration of a dynamic anomaly detected by our method.

3 METHOD

Anomaly detection in dynamic networks generally contains two stages [33]. The first stage is usually responsible for feature extraction from the domain-specific data. The second stage applies the anomaly detection algorithm to the extracted features and reports the detected anomalies.

We build our method upon this scheme in the following way. Firstly, we add to the Stage 1 a filtering step that keeps only potentially anomalous signals. It allows reducing the amount of data that is processed in Stage 2 by discarding obviously non-anomalous cases. Secondly, in Stage 2, our model contains a learning step that enhances the interpretability of the detected anomalies. It provides a complete and detailed spatio-temporal description of the detected anomaly, rather than the classification *anomalous/non-anomalous* commonly seen in anomaly detection. This description is a group of interconnected nodes (spatial information), where every node has a time-series attribute (temporal information) that indicates the time when the anomaly occurred.

The learning step is inspired by the model of a memory neural network, the Hopfield network with the Hebbian learning rule. In our model, we adapt the learning rule to fit our spatio-temporal data structure in the following way. An edge weight is reinforced when two neighbors are active together during the same time slice. This particular network design shares similarities with the Hotspot anomaly detection for graph streams [38], however, the authors use it for feature engineering (Stage 1 of the processing), while in our approach this update is a part of Stage 2, the learning and classification stage.

The Hopfield network approach learns a memory network, where the nodes correspond to the ones of the initial graph. During the learning process, edges between the nodes will be either strengthened or removed depending on the temporal behavior of each node. As a result of learning, nodes with similar behavior are connected by strong links and clustered together in the memory network. These clusters contain groups of nodes with common abnormal behavior.

Stage 1: Feature extraction and filtering. This stage highlights anomalies and extracts features from the raw data. We introduce a notion of *potential anomaly* and we filter out the nodes that are not potentially anomalous. Let V be the set of vertices of graph G and $x_i[t] \in \mathbb{R}$ be the value associated to vertex $v_i \in V$ at the time $t \in [0, T-1]$. The time-series have a length of T samples.

Definition 3.1 (Single vertex potential anomaly). Given a score function $f_i : \mathbb{R} \to \mathbb{R}$ for each vertex v_i and a threshold value $c_0 \geq 0$, a vertex $v_i \in V$ is said to have a potential anomaly at time t when its time series value is such that $|f_i(x_i[t])| > c_0$.

Note that the potential anomaly is local on the graph, i.e. it depends only on the time-series attribute of a considered vertex. If the score function is such that $f_i = f - \hat{f}(v_i)$, for some function f and $\hat{f}(v_i)$ a summary statistics of the scores $f(v_i[t])$, our definition corresponds to the definition of the anomaly in [33]. A basic example of a potential anomaly is the time-series values exceeding a fixed threshold c_0. In that case, f_i is the identity.

In this work, potential anomalies have to satisfy the following requirement. A potentially anomalous node must have a sufficient number of bursts in their time-series attribute. Unless this requirement holds, we discard the node. To define a burst of activity we compute the mean μ_i and the standard deviation σ_i of x_i over time. We normalize the time-series by defining the score function $f_i^b(x_i[t]) = (x_i[t] - \mu_i)/\sigma_i$. We select values that are above c_0^b, the activity rate parameter ($c_0^b = 5$ for Wikipedia, $c_0^b = 3$ for Enron). The *burstiness* b_i of a signal x_i of a node i is

$$b_i = \sum_{t=0}^{T-1} k_i[t], \qquad k_i[t] = \begin{cases} 1, & \text{if } |f_i^b(x_i[t])| > c_0^b, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The minimal number of bursts (potential anomalies) per vertex in Wikipedia is $b_i = 5$ while in the email dataset it is $b_i = 2$.

The general definition of the potential anomaly allows for other score functions, such as a moving average or ARMA filter prior to the thresholding, a short time Fourier transform or a wavelet transform. This depends on the dataset and should be chosen carefully to make a compromise between efficiency and scalability.

Applying f_i to the time-series gives us 1) features reflecting an anomaly and 2) an initial indication of anomalous behavior. In practice, removing nodes with no potential anomaly reduces the amount of data by an order of magnitude without losing relevant information that is required for the anomaly detection.

Stage 2: Hopfield network learning. The presented anomaly detection approach is aimed at detecting groups of vertices that have a similar, abnormal behavior. The learning stage is intended to make this coherent behavior apparent in the memory network.

Our approach is based on the Hopfield model of artificial memory [18]. It is an unsupervised learning method. To implement it, we use a synaptic-plasticity-inspired computational model, the Hebbian learning rule [17]. The main idea of this model of brain memory is that a co-activation of two neurons results in the reinforcement of a connection (synapse) between them. Although, contrary to the original learning rule, in our model, we do not take causality of activations into account.

In our case, the Hopfield network has N nodes. These nodes correspond to the ones given in the dataset, after the reduction performed at Stage 1 (only the nodes containing potential anomalies).

The learning process is as follows. We use the initial structure of the given network. For two initially connected nodes i and j of the Hopfield network, at time t, we update the weight of an edge e_{ij} between them according to the similarity measure $\mathrm{Sim}\{i, j, t\}$. Note that we only perform this step for the nodes that are initially connected and do not compare every possible pair of nodes. This is crucial for the tractability of the method in cases when we deal with large networks. For each time step t, the edge weight w_{ij} between i and j is updated as follows:

$$\Delta w_{ij} = \begin{cases} +Sim\{i,j\}, & \text{if } Sim\{i,j\} > \lambda, \\ -\alpha Sim\{i,j\}, & \text{otherwise,} \end{cases} \tag{2}$$

where $\lambda \geq 0$ is the *sparsity parameter*. Similarly to firing neurons, nodes expressing similar behavior have their connection weight increased. When $\alpha > 0$, the weight decreases allowing older events to be forgotten in order to keep only the latest anomalies. In our experiments, we set $\alpha = 0$, as we want to keep all the anomalies. We also fix $\lambda = 0.5$. The value of λ influences the sparsity of the final network. Increasing λ reduces the number of edges in the resulting memory network. One should increase the value of λ, when looking for the most outstanding anomalies and decrease it, when a higher sensitivity is required.

Before describing the similarity function, let us introduce the activity function y_i at a node v_i:

$$y_i[t] = x_i[t] \times k_i[t]. \tag{3}$$

Note that the activity function can also be defined from the features calculated at Stage 1, $y_i[t] = f_i(x_i[t]) \times k_i[t]$. In the present work, we use the definition (3).

Different similarity measures can be defined. A first example is the following:

$$\mathrm{Sim}\{i, j, t\} = y_i[t] y_j[t]. \tag{4}$$

When y_i and y_j are normalized, this measure gives the Pearson correlation between the nodes (if $\lambda = 0$ and $\alpha = 1$). The L^2 distance with a Gaussian kernel $\mathrm{Sim}\{i, j, t\} = \exp(-|y_i[t] - y_j[t]|^2)$ can also be a good candidate. Another measure, restricted to the interval $[0, 1]$ and that worked well in our applications, is defined as follows: $\mathrm{Sim}\{i, j, t\} = 0$ if $y_i[t] = y_j[t] = 0$ and otherwise,

$$\mathrm{Sim}\{i, j, t\} = \frac{\min(y_i[t], y_j[t])}{\max(y_i[t], y_j[t])} \in [0, 1]. \tag{5}$$

Recalling process. Starting from an initial partial memory pattern $P_0 \in \mathbb{R}^{N \times T}$, the recall of a learned pattern is done by the following iterative computation:

$$P_{j+1} = h_\theta(W P_j), \tag{6}$$

where $W \in \mathbb{R}^{N \times N}$ is the weight matrix of the Hopfield network. The function $h_\theta : \mathbb{R}^{N \times T} \to \mathbb{R}^{N \times T}$ is a nonlinear thresholding function (step function giving values $\{-1, 1\}$) that binarize the vector entries. The value θ is the threshold (same for all entries). In our case, we build a network per month so W is associated to a particular month. For each $j \geq 0$, P_j is a matrix of binarized time-series where each row is associated to a node of the network and each column corresponds to a time step of the month considered. We stop the iteration when the iterative process has converged to a stable solution ($\|P_{j+1} - P_j\| \leq \varepsilon$, where ε is small, the norm is the Frobenius norm). The initial pattern P_0 is a binary matrix, where the rows have all values set to -1 (inactivity) except the ones associated to the partial memory pattern obtained from the time-series using the expression of k_i defined in Eq. (1). The computation of the iterative process is efficient as the matrices are sparse and in practice, it converges after a reasonable number of steps.

Graph visualization and community detection. Since the connections between nodes with similar activity are reinforced during the learning process, after we prune low-weight edges, each Hopfield network transforms into a modular graph structure with strongly connected clusters of nodes having a similar activity. These groups can be either isolated connected components or communities within the largest connected component. The analysis of the Hopfield networks and their communities provides a good way to spot, analyze, and interpret the anomalies in the dynamics as we will see in the next section. To find communities, we use the Louvain method [7] and to represent the graph in 2D space for visualization, we use a force-directed layout [19]. Additional and more interactive visualizations are available online [28].

Storing memories of detected anomalies. The number of anomalies that can be memorized by a single network is limited [26]. Indeed, without the forgetting parameter α, the clusters of nodes will accumulate inside the graph, eventually overlapping and forming larger clusters of unrelated anomalies. To avoid this but still keep track of older events, we create snapshots of memories by slicing the time-series into time-windows of the finite duration. The time-window size depends on the application. For example, in the analysis of Wikipedia data, we use time-windows of one-month length.

Complexity. The computations are tractable because 1) they are local on the graph, i.e. weight updates depend on a node and its one-hop neighbors, 2) weight updates are iterative, and 3) a weight update occurs only between initially connected nodes and not among all possible combinations of nodes. These three facts allow us to build a distributed model to speed up computations. For this purpose, we use a graph-parallel Pregel-like abstraction, implemented in the GraphX framework [15], [37].

Stage 1 has complexity $O(NT)$ where N is the number of nodes and T the number of time steps. Stage 2 has a complexity of $O(ET)$, where E is the number of edges. The recall process involves a multiplication by a sparse matrix with $2E$ non-zero entries, hence the complexity is $O(ET)$. It is important to point out that the number of nodes and edges in the computations are not necessarily the numbers found in the dataset: Stage 1 may discard a large number of inactive nodes (containing no potential anomaly), Stage 2 sparsifies the time series as Eq. (3) may set to zero a large number of values.

4 DATASETS

Wikipedia dataset. We use the dataset described in [6]. This dataset is based on two Wikipedia SQL dumps: English language articles and user visit counts per page per hour. The original datasets are publicly available on the Wikimedia website [13].

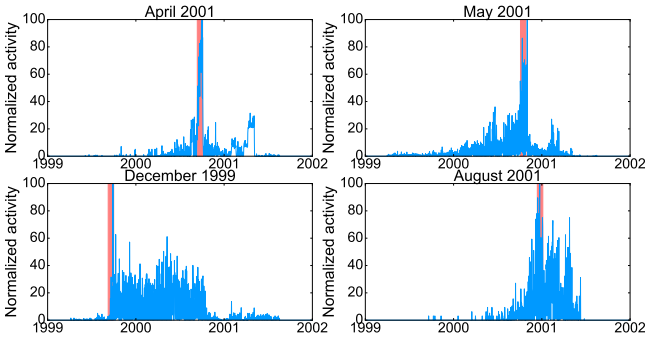


Figure 2: Anomaly detection in Enron email network. Red areas highlight the month periods previously reported as anomalies (ground truth supported by the real world events). Blue lines reflect the normalized (scale 0-100) overall activity level in the network computed by the proposed algorithm. We can see that the algorithm detects anomalies in all reported cases.

The Wikipedia network of pages is first constructed using the data from article dumps that contain information about the references (edges) between the pages (nodes)[1]. Time-series are then associated to each node (Fig. 1), corresponding to the visits history from 02:00, 23 September 2014 to 23:00, 30 April 2015. The time-series have a length of $T = 5278$ hours.

Enron dataset. We use Enron email dataset[2]. It contains 614586 emails sent over the period from 6 January 1998 until 4 February 2004. We remove the periods of low activity and keep the emails from 1 January 1999 until 31 July 2002 which is 1448 days of email records in total. We remove inactive email addresses that sent less than three emails over that period. In total, the Enron email network contains 6 600 nodes and 50 897 edges.

5 EXPERIMENTS AND RESULTS

In this section, we present our results of the anomaly detection experiments. We apply the proposed algorithm to the two datasets. The results obtained from the Enron one serve mainly to validate the method so we do not describe the analysis of the data. We dedicate more space to the detailed inspection of the Wikipedia dataset, which is a larger and more informative dataset allowing us to demonstrate the scalability of our method and to provide the detailed interpretation of the results.

Enron email dataset. In order to apply our anomaly detection algorithm to this dataset, we represent the data as a time-series graph data structure, which is depicted in Fig. 1. The static graph is the network of email communications: the nodes are email addresses and they are connected if they have exchanged at least one email over the 5 years. It is an undirected, unweighted graph. Time-series associated to the nodes are captured from the email activity; each temporal value is the number of emails sent from the associated address during one day. Here, the connection between email addresses is reinforced if a similar amount of emails have been sent by both of them over the same hour. Note that the graph and time-series design could have been more complex with directed edges or time-series associated to the edges. However, in order to stay as general as possible, we decided to strictly follow the steps described in Sec. 3.

In this case, an anomaly is a sudden increase in email communications among a group of employees of the corporation. After learning a Hopfield network for every month, we investigate their structure. We select four months where the results can be compared to the literature. The authors of [35] observe four anomalous periods and relate them to the specific news reports involving Enron. We use these events as a ground truth. These are the months of December 1999, April, May and August 2001. For each month, we select the largest connected component of the learned graph. We sum up the activity of its nodes to get a single time-series representative of the group activity. We show this normalized activity on Figure 2. We define an anomaly to be a spike of overall activity in a cluster of email addresses that we detect after learning. All the four curves have a larger activity during the chosen month than in the rest of the time span. For April and May 2001, it is more than twice the maximal activity for the rest of the months, showing the evidence of an anomaly. December 1999 and August 2001 increases in activity have a longer trace after the main anomaly has been detected but still, the high activity does not spread for more than one year. Concerning the monthly components of active nodes for the 4 chosen months, it involves 29, 25, 126, 28 nodes respectively for December 1999, April, May and August 2001. Almost all of them correspond to addresses of Enron employees. Except for the May event, the activity involves less than 30 persons. In May, it involves 100 employees out of the 158, showing an event that impacted the whole company.

The application of our method to the Enron dataset allows detecting all the anomalies presented in the state-of-the-art literature. It reveals the days of the peak activities, the duration of the events, and the persons involved, which makes more detailed investigation possible in the future. Besides, our approach can detect periodic or recurrent anomalies due to associative nature of the Hopfield network. This feature will be demonstrated in the following experiments on the Wikipedia data.

Wikipedia dataset. We conduct more detailed experiments on the Wikipedia web network and use Google Trends data as a ground truth to verify the accuracy of our findings. An anomaly, in this case, is a sudden increase in visitor activity in a small, local part of the web network. We start by analyzing the initial graph of Wikipedia web pages connected with hyperlinks. In this experiment, the time-series attributes of the nodes correspond to the viewership statistics of the associated web pages, described in Section 4.

To learn the global anomalies that occurred in the long-term dynamics of the Wikipedia web network, we build the Hopfield network of the 7-month period. After the learning, only 275 498 edges have strictly positive weights (4.2% of the initial graph). We remove the disconnected nodes and preserve only the largest connected component of the graph. The number of remaining nodes is 35 839 (31% of the initial number).

[1]Note that Wikipedia is continuously updating. Some links that existed at the moment we made the dump may have been removed from current versions of the pages. To check consistency with past versions, one can use the dedicated search tool at http://wikipedia.ramselehof.de/wikiblame.php.

[2]https://www.kaggle.com/wcukierski/enron-email-dataset

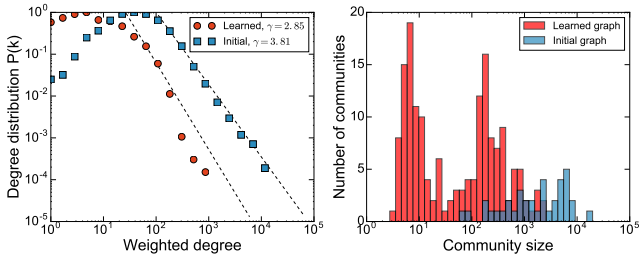
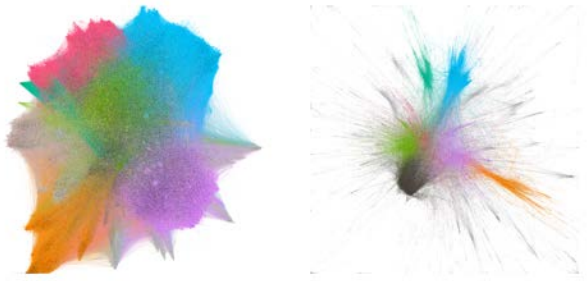


Figure 3: *Left.* Weighted degree distribution in log-log scale for the Wikipedia graph and Hopfield network learned over the entire 7 months time span. Linearity in log-log scale corresponds to power-law behavior $P(k) \sim k^{-\gamma}$. The learned graph preserves a similar scale-free behavior, but is less connected and has fewer hubs than the initial graph. *Right.* Community size distribution of the initial Wikipedia graph of hyperlinks (blue) and the learned Hopfield network (red). The total number of communities: 32 for the initial graph, 172 for the learned one.



(a) Initial (b) Learned

Figure 4: Wikipedia graph of hyperlinks (left) and learned Hopfield network (right). Colors correspond to the detected communities. The learned graph is much more modular than the initial one, with a larger number of smaller communities. The layout is force-directed.

The analysis of the static underlying graphs shows that both (the initial and learned) Wikipedia graphs have statistically heterogeneous connectivity (Fig. 3). However, the initial Wikipedia graph is dominated by large hubs that attract most of the connections to numerous low-degree neighbors. These hubs correspond to general topics in the Wikipedia network. They often link broad topics such as, for instance, the "United States" page, with a large number of hyperlinks pointing to it from highly diverse subjects. If we look at the viewership statistics, the activity of the visitors in these large clusters is uniform and does not expose any anomalies over time. We aim at extracting smaller communities that correspond to localized anomalies in the dynamics of the network. This is why we need the learned graph.

The visualizations of the initial and learned graphs using a force layout algorithm show striking differences (Fig. 4). The initial Wikipedia graph is dense and cluttered with a significant number of unused references, while the learned graph reveals smaller and more separated communities. This is also confirmed by the numerical measures such as the community size and degree distributions of the graphs (Fig. 3, right). The number of communities and their size change after learning. Initially, the small number of large communities dominate the graph (blue), while after the learning (red) we see a five times increase in the number of communities. Moreover, as a result of the learning, the size of the communities decreases by one order of magnitude. The modularity of the learned graph is 25% higher, strengthening the evidence of the creation of associative structures. These measures indicate that as a result of learning, we obtain a graph structure with small strongly connected clusters that correspond to a summary of anomalies in the network dynamics. We provide detailed examples of such anomalies in the following experiments.

The analysis of each community of nodes in the learned graph gives a glimpse of the events that occurred during the 7-month period and caused the anomalous behavior of visitors during that period of time. Each cluster is a group of pages related to a common topic such as a championship, a tournament, an awards ceremony, a world-level contest, an attack, an incident, or popular festive events such as Halloween or Christmas.

Before going deeper into the clusters analysis and the corresponding anomalies in the network dynamics, we investigate the evolution of the graph structure over time with an emblematic example.

Monthly networks. As stated in Sec. 3, we split the dataset into one-month periods. These periods are longer than the duration of an average event attracting the attention of Wikipedia users, that usually lasts no longer than two weeks. This means that we are going to detect multiple anomalies at once. Monthly graphs are smaller, compared to the 7-months graph, and contain 10 000 nodes on average. However, the properties and distributions of monthly graphs are similar to the 7-months one, described above.

To give an example of a detected event and the associated anomalous behavior of the Wikipedia users, we discuss the cluster of the USA National Football League championship, spotted in and extracted from the Hopfield network during several months between 2014 and 2015 (Fig. 5).

NFL is one of the most popular sports leagues in the USA and it triggers a lot of interest to the related articles on Wikipedia. Due to the high number of visits on this topic we were able to localize a cluster related to the NFL on each of the monthly graphs. Figure 5 shows the detailed information about the NFL clusters. The top part of the figure contains the learned graphs for each month, where the NFL cluster is highlighted in red.

The final game of the 2014 season, Super Bowl XLIX, had been played on February 1, 2015. This explains the continuous expansion of the cluster until February where its size reaches the maximum. The activity collapses right after this event and the cluster disappears.

For the sake of figure interpretability, we extracted 30 NFL team pages from the original cluster (485 pages) to show the details of the evolution in time as a table on Fig. 5. This fraction of the nodes reflects overall dynamics in the entire cluster. Each row describes the hourly activity of a page, while the columns split the plot into

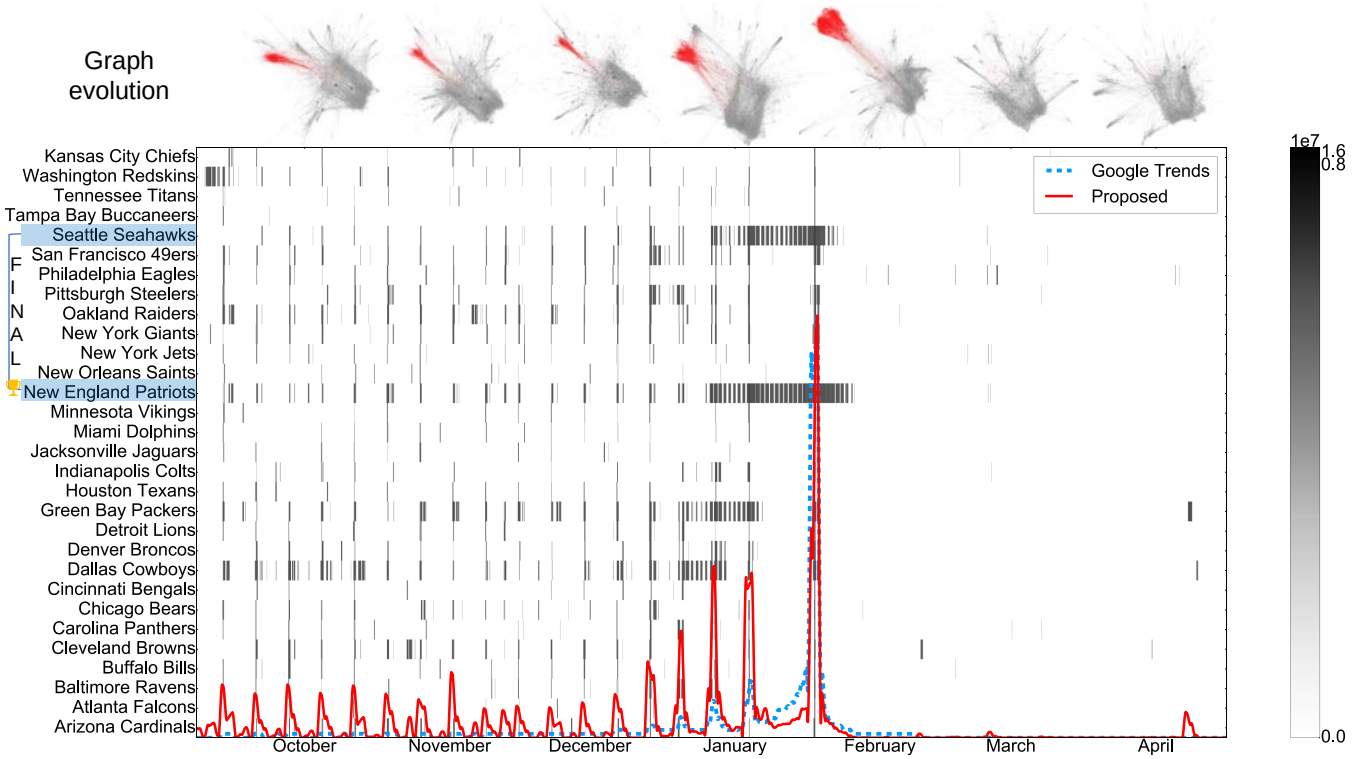


Figure 5: Evolution of the National Football League 2014-2015 championship cluster and visits on its articles. We show 30 NFL teams from the main cluster. Top: the 7 monthly learned graphs in gray, with the NFL cluster highlighted in red. The month is displayed on the bottom of the figure. Middle table: visitors activity per hour on the NFL teams' Wikipedia pages in greyscale (the more visits, the darker). Bottom: the red and blue curves are the total number of visits in the articles of the clusters over time (normalized) and the Google Trends curve for the keyword "Super Bowl", respectively.

months. The sum of visits for the selected pages is plotted as a red line in the bottom.

The dynamics of the detected cluster reflects the real timeline of the NFL championship. The spiking nature of the overall activity corresponds to weekends when most of the games were played. Closer to the end of the championship, the peaks become stronger, following the increasing interest of fans. We see the highest splash of the activity on 1 February, when the final game was played.

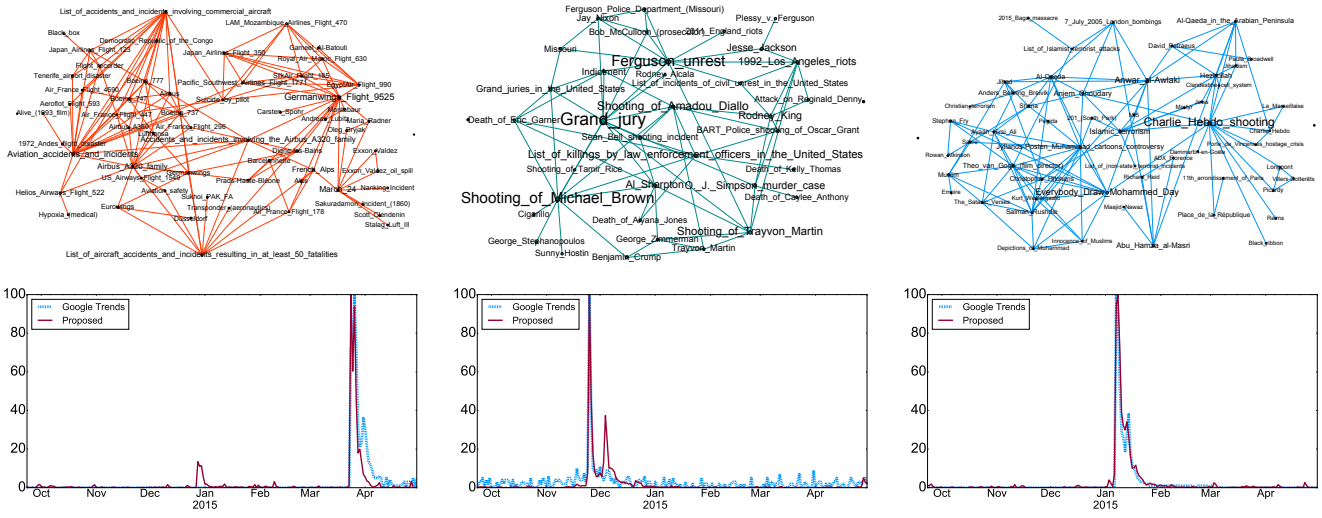
We want to emphasize that this cluster, as well as all the others, was obtained in a completely unsupervised manner. The football teams pages were automatically connected together in a cluster having "Super Bowl" as a common topic. Moreover, the cluster is not formed by one Wikipedia page and its direct neighbors, it involves many pages with distances of several hops in the graph.

The NFL championship case is an example of a periodic (yearly) event. The interest increases over the months until the expected final event, which causes an anomaly in the network dynamics. Accidents and incidents are the events of a different nature as they appear suddenly, without prior activity. The proposed learning method allows detecting this kind of events and the related anomalies as well. We provide examples of three accidents to demonstrate the ability of our method to detect anomalous behavior in the network in case of an unexpected event.

We pick three core events among 172 detected and discuss them to show the details of our anomaly detection approach. Figure 6 shows the extracted clusters from the learned graph (top) and the overall timeline of the clusters' activity (bottom). We evaluate the accuracy of the anomaly detection using Google Trends history records.

Charlie Hebdo shooting. 7 January 2015. This terrorist attack is an example of an unexpected event. The cluster emerged over a period of 72 hours, following the attack. All pages in the cluster are related to the core event. Strikingly, a look at the title of the pages is sufficient to get a precise summary of what the event is about. There is a sharp peak of activity on the first day of the attack, slowly decaying over the following week.

Germanwings flight 9525 crash. 24 March 2015. This cluster not only involves pages describing the crash or providing more information about it, but also the pages of similar events that happened in the past. It includes, for example, a page enumerating airlane crashes and the page of a previous crash that happened in December 2014, the Indonesia AirAsia Flight 8501 crash. As a result, the time activity of the event is connected to the one of the Flight 8501 crash, that is why we can see an increase in visits in December. This is an example where our approach captures two relevant events and



(a) Germanwings 9525 crash (b) Ferguson unrest (c) Charlie Hebdo attack

Figure 6: Graphs and activity timelines of the 3 events that triggered anomalies in the network dynamics. Top: Clusters of pages grouped after learning in the Hopfield network. Bottom: A normalized sum of all visits of the articles of each cluster over time (in red). The Google Trends curve for the keywords "Germanwings 9525 crash" (a), "Ferguson unrest" (b) and "Charlie Hebdo attack" (c) is displayed in blue.

groups them together in one cluster allowing to detect a secondary anomaly that is only implicitly related to the main one.

Ferguson unrest. Second wave. November 24, 2014 – December 2, 2014. This is an example of an event that has the official beginning and end dates. A sharp increase in the activity at the beginning of protests highlights the main event. This moment triggers the core cluster emergence. We also see that the cluster becomes active once again at the end of the unrest allowing to record the two related anomalies in the visitor activity.

Finally, in Table 1, we summarize our exploration of the clusters of the learned graphs by providing a list of handpicked page titles inside each cluster that refer to previous events and related subjects. The connected events occurred outside of the 7-months period we analyze. This illustrates the associative property of our the anomalies we detect. Firstly, pages are grouped by events with the help of visitors activity. Secondly, the events themselves are connected together by this activity. An anomaly is characterized by a group of connected concepts and these groups are in turn connected together through concepts they share. This feature makes it easy to interpret the cause of the detected anomaly.

Google Trends Evaluation. For each of the events presented before, we have compared the total number of visits in the clusters with Google Trends curves reflecting anomalous search activity of internet users. There is a striking correspondence between the detected anomalies and Google Trends, as we can see in Fig. 5 and 6. In all 4 examples, the anomalous activity and Google Trends curves reach their maximum at the same time and have a very similar shape. The differences that appear during the months prior to the Super Bowl date are explained by the fact that our "Super Bowl" cluster contains articles about Football teams and other topics related to the Super Bowl. Due to the associative nature of the detected anomalies, this example has a higher accuracy of detection than Google Trends.

As discussed previously, the periodic spikes in the visitor activity take place during the weekends when football matches are played. We observe the same phenomenon in the case of the Germanwings crash, where we observe a small peak of activity in December. This peak is the result of the prior activity on the pages related to another airplane crash that happened in December 2014. Again, this demonstrates the richness of the obtained clusters: they describe a group of events that caused an anomaly in visitor activity, as confirmed by Google Trends but represent more than a single keyword since the detected anomaly is a result of multiple real-world events.

Recalling memories. In this last experiment, our goal is to test the hypothesis that the proposed method, as a memory, allows recalling or recovering events from partial information. We emulate recall processes using the Hopfield network approach, described in Section 3. We show that the learned graph structure can recover a full event (cluster of pages and its activations) from an incomplete input.

We create incomplete patterns for the Hopfield network by selecting randomly a few pages contained in a chosen cluster. We built the input matrix P_0 setting to (-1) (inactive state) all the time-series except for the few pages selected. We then apply iteratively Eq. (6).

We present only one example of the recall, to demonstrate the recall ability of the network but also to illustrate its complexity, on Fig. 7. From the cluster associated to the Charlie Hebdo Attack and we choose a subset of the list of articles, here 80%. We apply the learned graph for the month of January when the memory was detected. After the recall, we can remark two important facts. First, if we focus on the short time span when the event occurs (within

Table 1: Examples of Wikipedia article titles contained in the clusters associated to the events presented.

Charlie Hebdo attack	Germanwings 9525 crash	Ferguson unrest
Porte de Vincennes hostage crisis	Inex-Adria Aviopromet Flight 1308	Shooting of Tamir Rice
Al-Qaeda	Pacific Southwest Airlines Flight 1771	Shooting of Amadou Diallo
Islamic terrorism	SilkAir Flight 185	Sean Bell Shooting Incident
Hezbollah	Suicide by pilot	Shooting of Oscar Grant
2005 London bombings	Aviation safety	1992 Los Angeles riots
Anders Behring Breivik	Air France Flight 296	O.J. Simpson murder case
Jihadism	Air France Flight 447	Shooting of Trayvon Martin
2015 Baga massacre	Airbus	Attack on Reginald Denny

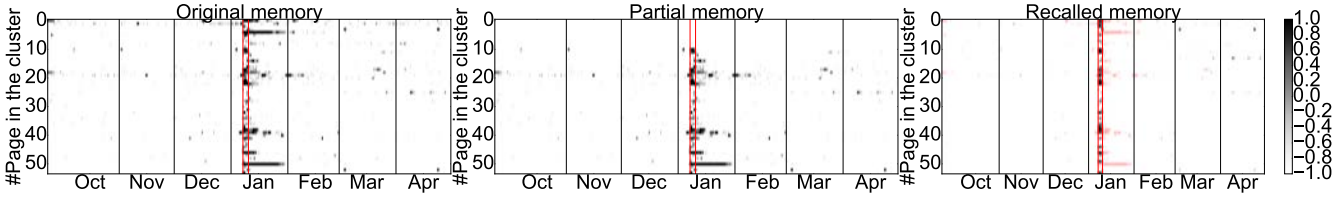


Figure 7: Recall of an event from a partial pattern (Charlie Hebdo attack). The red vertical lines define the start of the event and its most active part, ending 72 hours from the start. Left: full activity over time of the pages in the cluster. Middle: pattern with 20% of nodes set inactive (top lines). Right: the result of the recall using the Hopfield network model. In light red are shown the difference with the original pattern (the forgotten activity).

the red vertical lines), most of the cluster is recovered. Second, the model forgets a part of the activity, plotted in light red, outside of the event bounds. This missing part is made of pages that are active outside of the time of the event, giving the evidence that they are not directly related (or weakly related) to the event.

The results of this experiments show that our method can be used to recover the signals related to the detected anomalies given a noisy or incomplete input. For instance, this feature can be helpful in cases when the data required for anomaly interpretation is destroyed by some intruders that are interested in hiding traces after an attack on the network.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we presented a new unsupervised graph algorithm that allows detecting anomalies in large datasets with dynamic activity records. We showed that this approach allows analyzing the root causes of detected anomalies. Beyond the presentation of the method, we also provided new insights on collective behavior of Wikipedia users.

This approach is promising. We noted experimentally a high robustness of the method to the tuning of the parameters. We also plan to go deeper into the analysis of the recalling process.

This work opens new avenues for dynamic graph-structured data analysis. For example, the proposed approach could be used in a framework for automated event detection, monitoring, and recording. However, to cover these use cases, the method should be able to process streams of data as they are recorded. An online version of the method is not straightforward and will be the object of future work.

The artificial memory model used in the method suggests an interesting connection between the detection of anomalies and the functioning of the memory. This insight is more apparent in the experiments related to Wikipedia. Events and abnormal activity in the data trigger the recording process of the collective associative memory. The Hopfield network trained on Wikipedia articles transforms into a set of interconnected concepts or topics that have interested people at some point in the past. It resembles an artificial collective memory. This is fascinating and deserves further research in this direction related to social sciences.

7 TOOLS, IMPLEMENTATION, CODE AND ONLINE VISUALIZATIONS

All learned Wikipedia graphs (overall September-April, monthly activity, and localized events) are available online [28] to foster further exploration and analysis. For graph visualization we used the open source software package Gephi [4] and layout ForceAtlas2 [19]. We used Apache Spark GraphX [37], [15] for graph learning implementation and graph analysis. The presented results can be reproduced using the code, written in Scala and Python [27], [29]. Both datasets are available on Zenodo [21], [30].

ACKNOWLEDGMENTS

We would like to thank Michaël Defferrard and Andreas Loukas for fruitful discussions and useful suggestions. Also, we would like to thank anonymous reviewers for their constructive criticism and recommendations that helped improve this paper.

The research leading to these results has received funding from the European Union's H2020 Framework Programme (H2020-MSCA-ITN-2014) under grant agreement n^{o} 642685 MacSeNet.

REFERENCES

[1] Charu Aggarwal and Karthik Subbian. 2014. Evolutionary network analysis: A survey. *ACM Computing Surveys (CSUR)* 47, 1 (2014), 10.

[2] Leman Akoglu, Hanghang Tong, and Danai Koutra. 2015. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery* 29, 3 (2015), 626–688.

[3] Gowtham Atluri, Anuj Karpatne, and Vipin Kumar. 2018. Spatio-temporal data mining: A survey of problems and methods. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 83.

[4] Mathieu Bastian, Sebastien Heymann, Mathieu Jacomy, et al. 2009. Gephi: an open source software for exploring and manipulating networks. *Icwsm* 8 (2009), 361–362.

[5] Kirell Benzi, Benjamin Ricaud, and Pierre Vandergheynst. 2016. Principal Patterns on Graphs: Discovering Coherent Structures in Datasets. *IEEE Trans. Signal and Information Processing over Networks* 2, 2 (2016), 160–173.

[6] Kirell Maël Benzi. 2017. From recommender systems to spatio-temporal dynamics with network science. EPFL, Chapter 5, 97–98.

[7] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[8] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 15.

[9] Jana Diesner, Terrill L Frantz, and Kathleen M Carley. 2005. Communication networks from the Enron email corpus "It's always about the people. Enron is no different". *Computational & Mathematical Organization Theory* 11, 3 (2005), 201–228.

[10] James H Faghmous, Muhammed Uluyol, Luke Styles, Matthew Le, Varun Mithal, Shyam Boriah, and Vipin Kumar. 2013. Multiple Hypothesis Object Tracking For Unsupervised Self-Learning: An Ocean Eddy Tracking Application.. In *AAAI*.

[11] Michela Ferron. 2012. *Collective memories in Wikipedia*. Ph.D. Dissertation. University of Trento.

[12] Michela Ferron and Paolo Massa. 2011. Studying collective memories in Wikipedia. *Journal of Social Theory* 3, 4 (2011), 449–466.

[13] Wikimedia Foundation. 2016. Wikimedia Downloads: Database tables as sql.gz and content as XML files. (May 2016). https://dumps.wikimedia.org/other/ Accessed: 2016-May-16.

[14] Ruth García-Gavilanes, Anders Mollgaard, Milena Tsvetkova, and Taha Yasseri. 2017. The memory remains: Understanding collective memory in the digital age. *Science Advances* 3, 4 (2017), e1602368.

[15] Joseph E Gonzalez, Reynold S Xin, Ankur Dave, Daniel Crankshaw, Michael J Franklin, and Ion Stoica. 2014. GraphX: Graph Processing in a Distributed Dataflow Framework.. In *OSDI*, Vol. 14. 599–613.

[16] Alessandra Griffa, Benjamin Ricaud, Kirell Benzi, Xavier Bresson, Alessandro Daducci, Pierre Vandergheynst, Jean-Philippe Thiran, and Patric Hagmann. 2017. Transient networks of spatio-temporal connectivity map communication pathways in brain functional systems. *NeuroImage* 155 (2017), 490–502.

[17] Donald Olding Hebb. 2005. *The organization of behavior: A neuropsychological theory*. Psychology Press.

[18] John J Hopfield. 1982. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences* 79, 8 (1982), 2554–2558.

[19] Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. 2014. ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software. *PloS one* 9, 6 (2014), e98679.

[20] Nattiya Kanhabua, Tu Ngoc Nguyen, and Claudia Niederée. 2014. What triggers human remembering of events? A large-scale analysis of catalysts for collective memory in Wikipedia. In *Digital Libraries (JCDL), 2014 IEEE/ACM Joint Conference on*. IEEE, 341–350.

[21] Benzi Kirell, Miz Volodymyr, Ricaud Benjamin, and Vandergheynst Pierre. 2017. Wikipedia time-series graph. (Sept. 2017). https://doi.org/10.5281/zenodo.886484

[22] Louis Kratz and Ko Nishino. 2009. Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models. (2009).

[23] Theodoros Lappas, Marcos R Vieira, Dimitrios Gunopulos, and Vassilis J Tsotras. 2013. STEM: A spatio-temporal miner for bursty activity. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. ACM, 1021–1024.

[24] Miao Li, Shuying Zang, Bing Zhang, Shanshan Li, and Changshan Wu. 2014. A review of remote sensing image classification techniques: The role of spatio-contextual information. *European Journal of Remote Sensing* 47, 1 (2014), 389–411.

[25] Chang-Tien Lu, Yufeng Kou, Jiang Zhao, and Li Chen. 2007. Detecting and tracking regional outliers in meteorological data. *Information Sciences* 177, 7 (2007), 1609–1632.

[26] ROBERTJ McEliece, EDWARDC Posner, EUGENER Rodemich, and SANTOSHS Venkatesh. 1987. The capacity of the Hopfield associative memory. *IEEE transactions on Information Theory* 33, 4 (1987), 461–482.

[27] V. Miz. 2017. Wikipedia graph mining. GraphX implementation. https://github.com/mizvol/WikiBrain. (2017).

[28] V. Miz. 2017. Wikipedia graph mining. Visualization. (2017). https://goo.gl/Xgb7QG

[29] V. Miz. 2018. Anomaly Detection in the Enron Email Graph. GraphX Implementation. https://github.com/mizvol/enron-email-network-analysis. (2018).

[30] Volodymyr Miz, Benjamin Ricaud, and Pierre Vandergheynst. 2018. Enron Email Time-Series Network. (Aug. 2018). https://doi.org/10.5281/zenodo.1342353

[31] Misael Mongiovi, Petko Bogdanov, Razvan Ranca, Evangelos E Papalexakis, Christos Faloutsos, and Ambuj K Singh. 2013. Netspot: Spotting significant anomalous regions on dynamic networks. In *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 28–36.

[32] Misael Mongiovi, Petko Bogdanov, and Ambuj K Singh. 2013. Mining evolving network processes. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE, 537–546.

[33] Stephen Ranshous, Shitian Shen, Danai Koutra, Steve Harenberg, Christos Faloutsos, and Nagiza F Samatova. 2015. Anomaly detection in dynamic networks: a survey. *Wiley Interdisciplinary Reviews: Computational Statistics* 7, 3 (2015), 223–247.

[34] Ramine Tinati, Markus Luczak-Roesch, and Wendy Hall. 2016. Finding Structure in Wikipedia Edit Activity: An Information Cascade Approach. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 1007–1012.

[35] Heng Wang, Minh Tang, Youngser Park, and Carey E Priebe. 2014. Locality statistics for anomaly detection in time series of graphs. *IEEE Transactions on Signal Processing* 62, 3 (2014), 703–717.

[36] Elizabeth Wu, Wei Liu, and Sanjay Chawla. 2010. Spatio-temporal outlier detection in precipitation data. In *Knowledge discovery from sensor data*. Springer, 115–133.

[37] Reynold S Xin, Joseph E Gonzalez, Michael J Franklin, and Ion Stoica. 2013. Graphx: A resilient distributed graph system on spark. In *First International Workshop on Graph Data Management Experiences and Systems*. ACM, 2.

[38] Weiren Yu, Charu C Aggarwal, Shuai Ma, and Haixun Wang. 2013. On anomalous hotspot discovery in graph streams. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE, 1271–1276.

[39] Burcu Yucesoy and Albert-László Barabási. 2016. Untangling performance from success. *EPJ Data Science* 5, 1 (2016), 17.

Towards Predicting a Realisation of an Information Need based on Brain Signals

Yashar Moshfeghi
Department of Computer & Information Sciences
University of Strathclyde
Glasgow, UK
Yashar.Moshfeghi@strath.ac.uk

Peter Triantafillou
Department of Computer Science
University of Warwick
Coventry, UK
P.Triantafillou@warwick.ac.uk

Frank E. Pollick
School of Psychology
University of Glasgow
Glasgow, UK
Frank.Pollick@glasgow.ac.uk

ABSTRACT

The goal of Information Retrieval (IR) systems is to satisfy searchers' Information Need (IN). Our research focuses on next-generation IR engines, which can proactively detect, identify, and serve INs without receiving explicit queries. It is essential, therefore, to be able to detect when INs occur. Previous research has established that a realisation of INs physically manifests itself with specific brain activity. With this work we take the next step, showing that monitoring brain activity can lead to accurate predictions of a realisation of IN occurrence. We have conducted experiments whereby twenty-four participants performed a Q/A Task, while their brain activity was being monitored using functional Magnetic Resonance Imaging (fMRI) technology. The questions were selected and developed from the TREC-8 and TREC 2001 Q/A Tracks. We present two methods for predicting the realisation of an IN, i.e. Generalised method (GM) and Personalised method (PM). GM is based on the collective brain activity of all twenty-four participants in a predetermined set of brain regions known to be involved in representing a realisation of INs. PM is unique to each individual and employs a 'Searchlight' analysis to locate brain regions informative for distinguishing when a "specific" user realises an information need. The results of our study show that both methods were able to predict a realisation of an IN (statistically) significantly better than chance. Our results also show that PM (statistically) significantly outperformed GM in terms of prediction accuracy. These encouraging findings make the first fundamental step towards proactive IR engines based on brain signals.

CCS CONCEPTS

• **Information systems → Information retrieval**; **Users and interactive retrieval**.

KEYWORDS

Information Need, Prediction, fMRI, Brain Signals, Proactive, Information Retrieval, Generalised, Personalised

ACM Reference Format:
Yashar Moshfeghi, Peter Triantafillou, and Frank E. Pollick. 2019. Towards Predicting a Realisation of an Information Need based on Brain Signals. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313671

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313671

1 INTRODUCTION

Information need (IN) is an essential concept and plays a core and fundamental role in the information seeking and retrieval process. Over the last several decades, much research has been dedicated to better understand IN to most effectively satisfy it with information retrieval engines, e.g. [8, 12, 23, 32, 58]. In general, satisfying INs is a formidable challenge due to the inherent complexity and ambiguity associated with the IN concept. That is, expressing an IN using a set of query keywords is an uncertain and noisy process [53], as keywords can only vaguely approximate the actual IN [51]. The problem becomes even more pronounced when an IN is *ill-defined*: i.e., when the searcher only knows "fringes of a gap in [his/her] knowledge" [12] making it extremely difficult for the searcher to identify and describe the IN precisely with keywords [6, 8, 11]. Therefore, it is possible that a given query may not sufficiently define the characteristics of relevant documents, or even any relevant information since a searcher cannot form an appropriate initial state from which to form a query [13].

To close this gap, IR systems have been relying on techniques such as relevance feedback including explicit [29], implicit [57], and/or affective feedback [2] to better understand and satisfy users' information needs. Despite their invaluable contributions, these techniques lead to systems that are reactive in nature, being engaged in response to an input query issued by a user. Given the aforementioned problematic nature of users' queries (in terms of expressing searchers' real INs), recently, there is an increased interest in the research community in developing zero-query (or proactive) IR systems that aim to not only better understand, but even anticipate users' INs [9, 49]. These approaches mainly consider the contextual or behavioural information of users to predict their IN. While this is intuitively a promising approach, such proactive IR systems could potentially create an information overload situation for users, in particular, if users are not experiencing an IN when the information is presented to them. Therefore, being able to predict *when* an IN occurs is of high importance to better satisfying it. With this work, we make the first positive step in this direction.

Recently Moshfeghi et al. [42] have shown that there are clear, detectable, physical manifestations (i.e. neural correlates) of INs in human brains which can be identified in an early stage of an information seeking and retrieval process and also specific brain

regions where INs manifest themselves. However, two fundamental research questions remain unanswered:

- **RQ1:** "Is it possible to predict a realisation of an IN using brain signals?";
- **RQ2:** "Can we rely on a common set of predefined brain regions to predict a realisation of an IN or is there a unique set of regions for every individual?"

An answer to these questions will play a key role in opening new doors to the design and implementations of novel IR systems, which will be enabled to better and in particular proactively satisfies searchers' IN. To do so, we aim to train a classification model on brain activity to be able to discriminate between an IN realisation vs memory retrieval, within an IR process performed by participants engaged in a Question Answering (Q/A) retrieval task. In particular, in this paper, we are focusing on the period in which the brain exhibits complex activity with regards to the process analysing a question, from the moment of observing the question to the point in time when they either recognise an "Anomalous State of Knowledge" (ASK) [7, 8], or they have a successful memory retrieval. We then aim (i) to build a model that predicts a realisation of IN (in its earliest possible state) from the signals of the brain regions that have been associated with an IN realisation and (ii) contrast its prediction accuracy with the accuracy of a prediction model that considers the signals of the whole brain regions, in a Q/A task. Our positive results can be the basis for next-generation proactive IR systems based on brain signals.

The remainder of the paper is organised as follows: Section 2 presents related work and the background. Section 3 discusses at length the experimental methodology. Section 4 presents and analyses our results. Finally, Section 5 presents our key conclusions.

2 RELATED WORK

2.1 From Reactive to Proactive IR

A well-known issue in IR research is that INs are inherently complex and ambiguous in that, typically, users may not know beforehand exactly what they are looking for, or, in other words, what are the set of documents/information that would satisfy their need [12]. Given this, formulating queries (i.e., mapping INs to a set of keywords) is inherently an imperfect process [7, 8, 12, 22, 51]. The formulated query is unavoidably therefore not guaranteed to be an accurate representation of the users' IN, and thus brings about the question whether the retrieved documents are relevant [55].

IR systems have addressed this issue using mechanisms based on relevance feedback. In short, relevance feedback is an iterative process whereby the system receives cues from users as to the relevance of presented results so far, improving the relevance of results presented in the next step in the process. Such cues from users come in many forms and types. As mentioned in Section 1, relevance feedback can be categorised as being explicit [29], implicit [57], and/or affective [2]. The most robust method to improve retrieval effectiveness is explicit feedback [29]. However, it places a large cognitive burden on users [56]. Implicit feedback avoids this pitfall. In implicit feedback, relevance is indirectly inferred from the interactional data without users' additional input [24]. Researchers, for example, have tried to infer relevance by understanding how task [57] dwell time [28] and click-through [24] relate to relevance. However, this is not a panacea. Kelly and Belkin [27] have shown that implicit feedback can be unreliable and difficult to interpret, limiting its usefulness. Affective feedback, on the other hand, [2] relies on different types of feedback data, such as facial expressions [5], eye tracking [36], and physiological signals [3, 38] (such as skin temperature) and uses them as implicit relevance judgements. Nonetheless, affective feedback on its own is not yet considered to be very effective. More recently, Arapakis et al. used emotions in the process of information retrieval [2, 4] and later showed that emotions can be used for building implicit feedback systems [4], and to personalise search [2]. Going a step further, the work by Moshfeghi et al. used, in addition to emotions, [39], physiological features both for modelling relevance and for predicting task types [38]. However, these approaches are reactive in nature, being engaged in response to an input query issued by a user. In this work, we aim to predict a realisation of IN before a user has issued a query.

Similar research efforts in the community have begun to explore and stimulate discussion about systems that can proactively anticipate and fulfil the information needs of users [10]. In addition to the works published in [10], additional research in this domain of proactive IR is emerging. A recent example is the work by Song et al. [49] where a query-less search and recommendation engine was proposed. The engine predicts when and what types of tasks users will issue in the near future by considering the time dimension. They also considered a personalised approach for predictive IR by performing model adaptation by learning individual deep learning models for each user task. However, their proactive model is based on strong user behaviour repetition in search engine logs and even stronger repetition on mobile devices. An even more recent example is the work by Benetka et al. [9] which addresses the problem of identifying, ranking, and anticipating a user's information need, based on users' last activity. This work develops a method that gathers and ranks information needs relevant to an activity using a limited amount of query suggestions from a search engine. However, in this study, INs are inferred from users' check-in records on a location-based social network.

Our paper blends well with the above research efforts. By predicting a realisation of IN from brain signals, IR system could be in stronger position to disambiguate and understand INs, e.g. by taking into account contextual information/data around the time an IN was realised. It also could alleviate information overload scenario. That is, a proactive IR system could make a better decision as to provide any information to users, based on their prediction of a realisation of IN from their brain signals.

2.2 Advances in Neuroscience & IR

There is growing research literature using modern techniques of brain imaging to explore the user perspective of IR. The two topics which have received attention include relevance and information need (IN), though other research has begun to investigate the effect of task and specificity of search results [43]. Research into the topic of relevance has been performed using the brain imaging techniques of Electroencephalography (EEG), magnetoencephalography (MEG) and fMRI. Results of Moshfeghi et al. [40] used fMRI to identify brain regions activated by the process of judging the relevance of an

image. They contrasted brain activity from the viewing of relevant images to non-relevant images. Brain regions with higher activity when viewing relevant items included the inferior parietal lobe, inferior temporal gyrus and superior frontal gyrus. While these results were interpreted as representing the activity of visuospatial working memory on relevant items, the prefrontal cortex has also been shown to be activated in judging the relevance of written items [44] and thus this region could be involved in aspects of general task control and decision making.

Another study into relevance was conducted by Eugster et al. [17] that used EEG to show that the frequency content of the EEG signal as well as Event Related Potentials (ERPs) can be used effectively as a set of features to decode the relevance of a text. This work was later extended [16] to show that methods of automatic classification applied to single trials could be used to classify individual words being read as relevant or non-relevant successfully. Likewise, Kauppi et al. [25] showed using MEG that the frequency content of the MEG signal, along with eye movement data can be used for decoding the relevance of images. These results are consistent with the findings of Allegretti et al. [1] who reported on EEG results that indicated that within 500 ms EEG signals begin to appear that differentiate between viewing a relevant and a non-relevant image. These studies with fMRI, EEG and MEG have used the relative strength of the different measurement techniques to make great progress and to indicate where in the brain relevance judgments are happening and what the time course is of these neural processes that determine relevance.

Expanding upon work on relevance, research by Moshfeghi et al. [42] used fMRI to examine the neural processes involved in how IN emerges. In this study, they compared two tasks where participants read a question and were provided with responses that included the possibility of answering the question or asking for additional information (acknowledging an information need). In the first task, participants went on to the next trial regardless of their response, but in the second task they went on to an information retrieval dialogue if they stated an information need. Brain imaging results revealed the posterior cingulate as a region common to both tasks. However the posterior cingulate can be divided into different regions, and activity in different regions was found for the two tasks that reflect the role of the posterior cingulate in switching between internal (e.g. memory) and external (e.g. sensory) sources of information.

How the neural correlates of IN fit into the larger context of a search was explored by Moshfeghi and Pollick [41] who investigated search as a process of transitions between neural states. To achieve this, they defined the time periods corresponding to five different aspects of search (realisation of Information Need, Query Formulation, Query Submission, Relevance Judgment and Satisfaction Judgment) and the brain activity was contrasted between these different periods. Results showed how the cognitive demands of the different search periods involved various large-scale brain networks including perception, cognition and executive control that reflected task demands. For example, the transition between IN and Query Formulation revealed extensive changes consistent with what would be expected from the dynamic restructuring of neural resources to attempt to answer the query. Motivated by these results, this paper indicates that the period of IN involves complex patterns of brain activity and by application of more sensitive data analyses, reliable fine patterns of brain activity that are consistent with being in a state of IN can be revealed.

3 EXPERIMENTAL METHODOLOGY

To investigate our research questions mentioned in Section 1, an information need scenario was created where participants were asked a multiple-choice question, and if they did not know the answer, they engaged in a search process. Before describing the experimental methodology in detail, we would like to point out several practical constraints on the design and analysis of fMRI experiments that are critical to consider [15, 40]. Firstly, the fMRI scanning environment is restrictive in that a participant is placed in the central bore of the scanner and must lay supine with their head kept still. Moreover, only limited response/interactive devices can be placed in this scanning environment without causing signal or safety issues. This constraint led to the use of multiple choice questions for a task since it was possible to provide a response using an MRI-compatible button box. Another constraint is that although fMRI provides a spatial resolution of brain activity to within a few millimetres, it does not have a fine temporal resolution. This is because the time taken to achieve one measurement of the entire brain takes approximately 2 seconds. This relatively slow rate of data acquisition is compounded by the fact that the Blood Oxygenation Level Dependent (BOLD) signal measured is related to the underlying neural signal in a complex way that introduces further delays [19]. As a consequence of these temporal factors, it was necessary to time the events happening in the experiment at a rate that was compatible with our fMRI measurements. To achieve a suitable design for our questions about IN we were guided by the methods used by Moshfeghi et al. [42] who explored the neural correlates of IN. These methods themselves had been adapted from research into problem-solving [59] that examined neural correlates of insight by comparing brain activity when a multiple choice response showed insight, to brain activity when a multiple choice response did not show insight. A final practical constraint is that running fMRI experiments are costly at £500 per hour which would limit the number of our participants.

3.1 Participants

Participants consisted of twenty-four healthy individuals with 11 males and 13 females. The participants were under the age of 44, with the largest group between the ages of 18-23 (54.1%) followed by a group between the ages of 30-35 (20.8%). The handedness survey indicated that 79.1% were right-handed, 12.5% were left-handed, and 8.33% were mixed-handed. Participants tended to have a postgraduate degree (20.8%), bachelors (33.33%) or other qualifications (45.8%). They were primarily students (54.1%), though there were a number of individuals who were self-employed (20.8%), not employed (4.16%) or employed by a company or organisation (20.8%). Participants were primarily native speakers (79.1%) or had an advanced level of English (20.8%). They all had experience in searching, with an average of 11.66 years (SD of 3.58) experience.

3.2 Design

We used a within-subject design for this study. The independent variable was the realisation of information need, with two levels: an information need was realised (IN), and no information need was realised (no-IN), which was controlled by responding to questions viewed on the screen. The set of questions were designed so that averaged across all participants there would be an equal number of responses expressing an answer and expressing a need for search. The dependent variable was brain activity revealed by the BOLD signal. From this BOLD signal, we estimate the accuracy of different prediction methods of a realisation of an IN.

3.3 Task

The entire experimental procedure is explained in the following section, and here we explain the task performed by participants in an experimental trial while being scanned. Participants first were presented with a question for 4 seconds, then for 4 seconds, four possible responses were provided while the question remained on the screen. Participants were not able to make a response until after the 4 seconds of observing the possible responses. The reason for this was so that brain activity related to the motor response of pressing the button would not be contained in the target pattern of brain activity, which only included these first 8 seconds. After the 8 seconds, participants were able to respond. Of the four possible responses, one was always the correct answer, and one of them was always "need to search". If participants chose that they needed to search then, this trial was labelled as a "IN", and otherwise, the trial was labelled as "no-IN". The position of the four alternatives was randomised for each trial and the response provided by pressing one of the four buttons available on the button box that each participant had in their right hand. The time to respond was left free so that participants were not under time pressure to respond. The order of the questions was randomised for each participant. If the answer "need to search" was provided then they entered an additional stage where they formulated a search query (and submitted it verbally into a noise-cancelling microphone), received a document and evaluated this document. In the present study, we do not investigate brain activity during the search and document evaluation periods. Instead, we focus only on the brain activity in response to the presentation of the question and the possible responses.

3.4 Question Answering Dataset

To perform the task scenario mentioned in Section 3.3, we created a Question Answering dataset[1]. To develop this standard set of questions, we used previous runs of TREC Q/A Track, in particular, we carefully selected a set of 80 questions from the TREC-8 and TREC-2001 Question Answering Tracks - Main Task[2]. We chose these two Tracks since they were the first and last tracks where the questions presented there were (i) independent from one another, in contrast to other Tracks that share a relationship, and (ii) they also provided the correct answer to the questions.

[1]The Question Answering dataset is available upon request.
[2]For more information, please visit http://trec.nist.gov/data/qa/t8_qadata.html and http://trec.nist.gov/data/qa/2001_qadata/main_task.html

We then manually examined all the questions presented in these two tracks and selected a subset of questions that (i) was not longer than one line, and (ii) the correct answer to the question was not longer than five words. This constraint is due to the limitation of presenting the questions and options to the participants in an fMRI setting. An additional constraint was that there were at least two relevant and non-relevant answers in their QRel. We then removed the questions that were ambiguous or were time dependent, e.g. *"Who is the president of Stanford University?"* (TREC-8, Topic 51), making the answers provided in the Track not appropriate. The answers of all these questions were then checked by current search engines to make sure that the answers are still valid and correct. We also created two wrong answers for each question that were in the domain of the question. For example, for question *"What is supernova?"* (TREC- 2001, Topic 1067) the correct answer is *"An exploding star"* and we created two other wrong answers i.e. *"A newborn star"* and *"A dead star"*. We also made sure that the questions covered a wide range of topics, e.g. history, politics, science, etc. this was done to reduce any bias that might occur from an emphasis of a particular type of question.

Over this set of questions, two annotators separately judged the difficulty of the questions (i.e. hard or easy) and then selected a subset of questions where both annotators agreed upon their difficulties. For our task, 40 questions were randomly selected where 20 were easy, and 20 were hard questions. Since the fMRI data acquisition for these 40 questions was divided into two runs, additional care was made to further divide the questions across the runs so that they both had 10 easy and 10 hard questions covering a variety of topics. The goal of this procedure was to control the set of questions such that on average there was an equal chance of experiencing ASK [7, 8] and knowing the answer.

Another extra step was to prepare the documents that were shown to the subjects once they engaged in a search process. This took the form of simulating a snippet answer that is returned by a current search engine such as Google when a question is submitted. For this purpose, we selected two relevant and two non-relevant documents from QRel. The length of the answers provided in TREC-8 and TREC 2001 were incompatible. To keep the size of the results consistent, for those answers that were too short, we found the original source file and selected sentences around the answer so that all snippets had the same length. The average length of the answers shown to the participants for the first and second run were 39.47 words (SD of 3.33) and 39.65 words (SD of 3.285) respectively. This was done to reduce any potential confounding effect of snippet size on the brain activity results.

3.5 Procedure

This section outlines the flow of the study, from beginning to end. Ethical permission for the study was obtained from the Ethics Committee of the College of Science and Engineering, University of Glasgow. Participants were recruited from the participant database at the Centre for Cognitive Neuroimaging, University of Glasgow. Participants were instructed that the experiment would take approximately one hour in total and would include performing tasks while being scanned, as well as a 10 minute period where the anatomical data were obtained, and they need only stay still. They were

informed that they could leave at any point in time during the experiment and would still receive payment (the payment rate was £6/hr). They were then asked to sign a consent form. Before participating, participants underwent a safety check to guarantee that they did not possess any metal items inside or outside of their body, or any other contraindications for scanning, such as certain tattoo inks. They were then provided with clothes (similar to a training suit) to wear for the duration of the experiment as a precaution to avoid interference with the fMRI signal from any metal objects in their clothes.

Next, to provide training before entering the scanner, participants were given an example task and a corresponding set of example questions to familiarise themselves with the procedure. Once they had successfully completed their training task, participants entered the fMRI machine, and the experimenter adjusted the settings of the machine to maximise their comfort and vision. While being scanned, each participant participated in two separate runs of our task, with each run comprised of 20 questions. A short rest break given between the two runs was chosen to give the participants a break to relax during the scanning and to prevent fatigue. After the functional runs were complete, the anatomical data of each participant were obtained.

After completion of scanning, participants were asked to fill out an *exit questionnaire*. The purpose of this questionnaire was to obtain demographic and background information about the participants. This included familiarity with related fMRI and user studies as well as questions addressed at understanding the participants' subjective experience about how they found the experiment. They also filled out the Edinburgh handedness questionnaire [45], which provides the evaluation of whether the participant was right-, left- or mixed-handed. Handedness information was obtained since lateralisation of brain function is influenced by handedness, and we wished to ensure that our sample of participants approximated the general population.

3.6 Apparatus

Visual presentation of the questions, answers and search results to participants was obtained using Presentation® software[3] to control projection on a LCD projector that projected onto a translucent screen, which was visible to participants in an angled mirror while they lay in the bore of the MRI scanner.

3.7 fMRI Data Acquisition and Preprocessing

A 3T Tim Trio Siemens MRI scanner, including a 32-channel head coil, was used to obtain all brain imaging data. This scanner is located on-site at the Centre for Cognitive Neuroimaging, University of Glasgow. Functional data included two T2*-weighted runs of our task with parameters: TR 2000ms; TE 30ms; 32 Slices; $3mm^3$ voxel; FOV of 210, imaging matrix of 70×70. An anatomical scan was performed at the end of the scanning session that comprised a high-resolution T1-weighted anatomical scan using a 3D magnetisation prepared rapid acquisition gradient echo (ADNI- MPRAGE) T1-weighted sequence (192 slices; $1mm^3$ voxel; Sagittal Slice; TR = 1900ms; TE = 2.52; 256×256 image resolution).

[3]Presentation® software (Neurobehavioral systems, Inc.), http://www.neurobs.com.

The fMRI data were preprocessed using Brain Voyager QX. A standard pipeline of pre-processing of the data was performed for each participant [20]. This involved slice scan time correction using trilinear interpolation based on information about the TR and the order of slice scanning. Three-dimensional motion correction was performed to detect and correct for small head movements by spatial alignment of all the volumes of a participant to their first volume by rigid body transformations. Also, linear trends in the data were removed, and high pass filtering with a cutoff of 0.0025 Hz performed to reduce artefact from low frequency physiological noise. The functional data were then coregistered with the anatomic data and spatially normalised into the common Talairach space [50]. Finally, the functional data of each individual underwent spatial smoothing using a Gaussian kernel of 6mm.

3.8 Pilot Studies

Before commencing the main study, we performed a pilot study with two participants to verify the procedure worked appropriately. The results of the pilot study verified the accurate logging of both behavioural and brain imaging data and generally confirmed the effectiveness of our procedure. However, it also revealed some aspects that could be improved, and we changed the procedure accordingly.

3.9 Multi-voxel Pattern Analysis (MVPA)

The technique of Multi-voxel Pattern Analysis (MVPA) is increasingly used to analyse fMRI data, aiming to identify patterns from brain activity and enabling sensitive identification of the information encoded by brain activity. Most early fMRI studies investigating perception and cognition employed a General Linear Model (GLM) and aimed to localise brain regions by employing a univariate analysis at every voxel to contrast the magnitude of the brain response to different stimuli conditions. In contrast, MVPA employs a multivariate analysis that can examine the spatial encoding of information in a set of voxels. Increased performance over the univariate GLM approach is obtained since brain activity related to a mental process can be distributed across multiple brain areas [33], and MVPA is sensitive to these differences.

3.10 IN Prediction Methods using MVPA

For prediction of IN realisation, we have a binomial classification problem where the classes are "+1" (IN) and "-1" (no-IN). Therefore, we used ROI[4]-SVM, an implementation of SVM for MVPA in Brain Voyager QX[5] to discriminate between the two classes explained above. We trained our models using a linear kernel which outperformed other SVM kernels (e.g. polynomial and RBF) based on our analysis, not presented due to the space limits.[6] Using the settings described above, we devised two prediction methods, a Generalised and a Personalised.

[4]ROI refers to a region of interest which is a subset of brain voxels that are considered as a region to serve a particular purpose.

[5]http://www.BrainVoyager.com

[6]It has been shown that the linear kernel is a special case of RBF [26] where the linear kernel with a penalty parameter C has the same performance as the RBF kernel with some parameters (C, γ) [54]. But when the number of features is very large, which is the case for brain signals, RBF kernel is known to be not suitable and the linear kernel is the one that should be used [54].

Generalised Method (GM): The Generalised method follows the hypothesis that there is a common ROI for all participants that represents an IN realisation. We created this ROI by including spheres of radius 5mm centred around each of the Talairach coordinates of peak activation reported by [42] in their GLM analyses contrasting IN versus no-IN. All reported regions were selected regardless of whether they showed greater activity for IN or no-IN for comprehensiveness.

Personalised Method (PM): In contrast, the Personalised method follows the hypothesis that there is a unique ROI for each individual that represents an IN realisation as revealed by the Searchlight analysis [30, 31]. The Searchlight analysis uses MVPA and addresses the issue that the entire collection of brain voxels forms a high dimensional space. To reduce dimensionality and establish a reduced set of feature voxels it works by moving a small predefined volume (three voxels in our case) throughout the entire brain and creates a map based on the prediction accuracy obtained at the centre of the volume. From this map, it is possible to find all voxels with performance above a criterion value to define a personalised ROI that can then be used to test the prediction accuracy.

Baseline (BL): Our baseline represents an untrained model where all its predictions are based on a random choice, i.e. where its accuracy is set to 50%.

4 RESULTS

4.1 Task Perception

In this section, we investigate participants' evaluation of the task they had performed in terms of the judged difficulty of the task, the familiarity of the participant with the task and the degree to which they found their performance of the task to be stressful, clear, successful, and satisfactory. Namely, participants were given the following questions *"The tasks we asked you to perform were [easy/stressful/familiar/clear/Satisfactory] (answer: 1: "Strongly Disagree", 2: "Disagree", 3: "Neutral", 4: "Agree", 5: "Strongly Agree")"*. Descriptive statistics of these responses are shown by box plots in Figure 1, which show five key statistics: the minimum, first, second (median), third, and maximum quartiles.[7]. These results indicate that participants found the tasks difficult (not easy) and stressful, familiar, clear, successful, and satisfactory.

4.2 Log Analysis

The fMRI analysis relied upon a participant's response to the question to code whether a trial was IN or No-IN. This raised the consideration that for analysis of the brain data to have the maximum power it is important to have approximately an equal response rate for IN and No-IN responses. Thus, it was important to examine the average response rates for whether the number of IN and no-IN responses were approximately balanced. The average number of IN responses was 17.5 (SD of 5.91), and the average number of No-IN responses was 22.5 (SD of 5.91). A paired t-test revealed a marginal difference between the type of responses (p-value = 0.05).

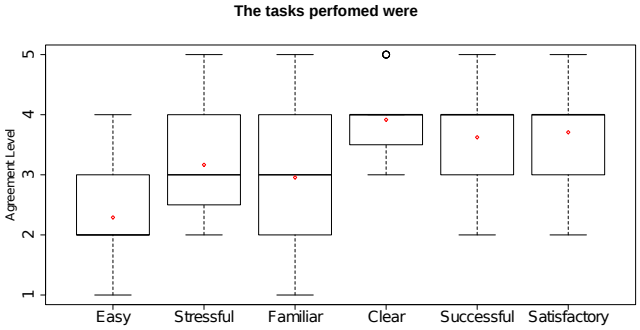


Figure 1: Box plot of the task perception based on the information gathered from the questionnaires of twenty-four participants. The red diamond represents the mean value.

4.3 Comparison of Brain Regions Associated with GM and PM

Before examining the performance of the MVPA approach in classification accuracy of discriminating between IN and no-IN states, it is instructive to examine the brain areas used by the two approaches. The GM technique restricted the analysis to the same feature set of voxels for all participants while the PM used a unique feature set of voxels for each individual as revealed by the Searchlight procedure. Given the bespoke nature of PM, where a unique feature set of voxels was derived for each participant, comparison of the regions used in PM to those used in GM is not straightforward. To address this issue, we created a probability map of the individual PM results that allowed us to visualise how many of the twenty-four participants had the same voxel present from the Searchlight procedure. To achieve this, we first defined 70% as a common threshold of prediction performance for the Searchlight procedure in discriminating IN versus no-IN.

This probability map included 87.9% of all voxels in the brain volume if we examined the probability that a voxel would be found in any one of the participants. Since this map contains almost all voxels in the brain volume, we aimed to identify a more specific subset of voxels by increasing the criterion in the probability map to 0.35, corresponding approximately to a voxel being found in 8 of the 24 participants. This resulted in a map with 0.6% of all voxels in the brain volume. This map is shown in Figure 2, along with the regions used for the GM analysis. Details of the voxels identified in our probability map are provided in Table 1. As can be seen, the probability map revealed ten areas that are distributed across the brain, in many cases bilaterally, and concentrated mostly in frontal and occipital regions.

Additionally, except for the inferior frontal gyrus, there is little overlap between the GM regions and the PM regions. An essential point here is that the current results used a Multi-Voxel Pattern Analysis (MVPA) while the previous experiment used a General Linear Model (GLM) analysis. MVPA uses the pattern of activation of a group of voxels to classify brain activity and is sensitive to the spatial pattern of information encoded in brain activity. However, GLM analysis focuses on brain activity that occurs per each

[7]Further information can be found in [37].

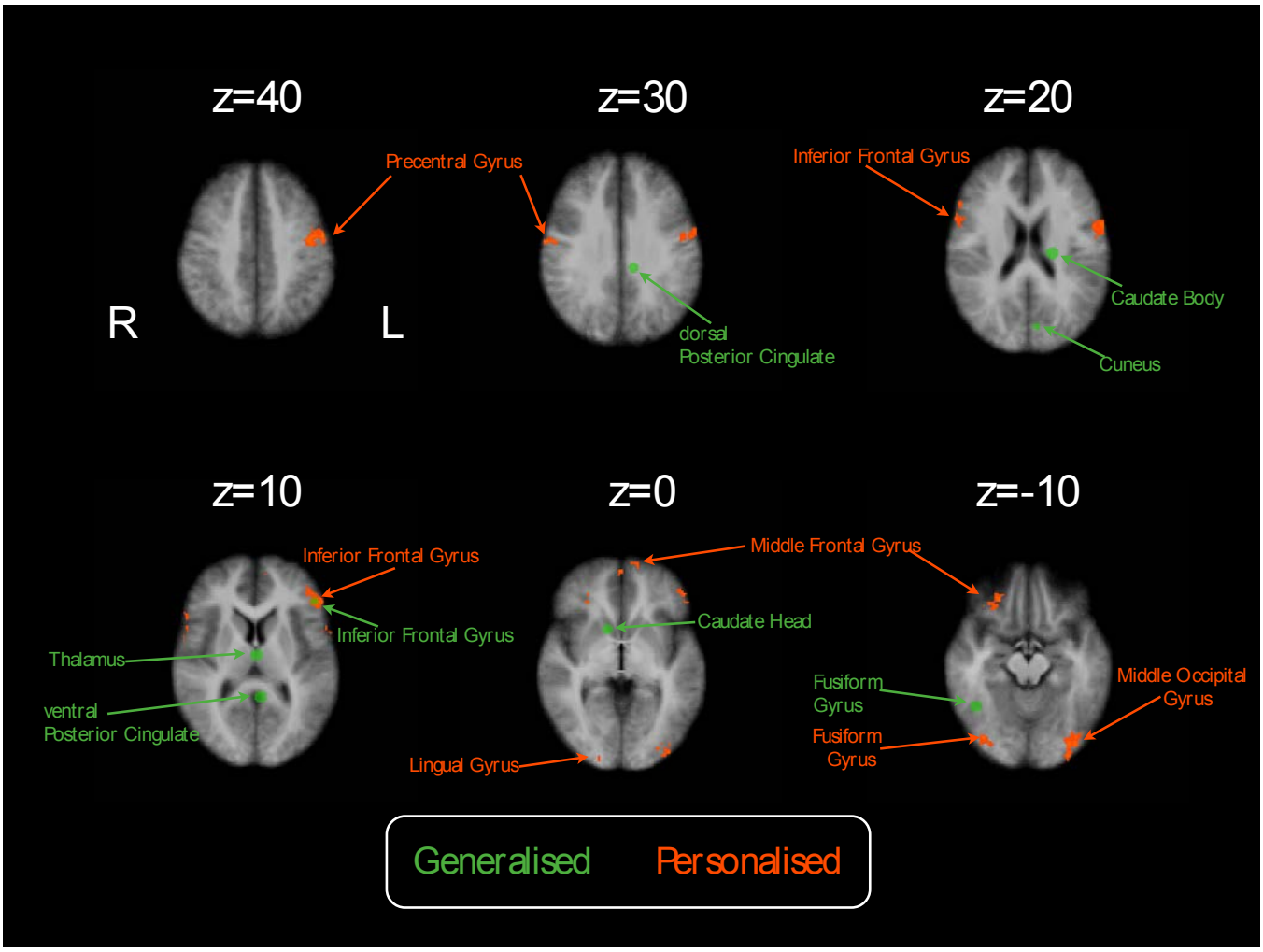


Figure 2: Comparison of brain regions in PM with probability greater than 0.35 to results from GM based on regions found in [42] PM and GM regions are shown in orange and green respectively.

Table 1: Details of probability map of PM results, including their anatomic label, location, Brodmann Area (BA), effect size and volume.

		Talairach Coordinates				Peak Probability	Number of voxels
Brain Area	Hemisphere	X	Y	Z	BA	value	mm^3
Precentral Gyrus	Right	60	-7	31	6	47.6%	230
Inferior Frontal Gyrus	Right	57	5	16	44	52.4%	1253
Fusiform Gyrus	Right	36	-73	-8	19	57.1%	558
Middle Frontal Gyrus	Right	24	35	-11	11	52.4%	756
Lingual Gyrus	Right	18	-88	-2	17	47.6%	130
Medial Frontal Gyrus	Left	-9	53	7	10	47.6%	203
Medial Frontal Gyrus	Left	-3	63	7	10	47.6%	230
Middle Occipital Gyrus	Left	-36	-79	-8	18	57.1%	1839
Precentral Gyrus	Left	-51	-6	40	6	52.4%	3444
Inferior Frontal Gyrus	Left	-45	29	10	46	57.1%	1127

voxel and is sensitive to differences in mean activation [14, 21, 52]. Therefore, MVPA and GLM analysis are different in nature, and it is not entirely expected that they would yield identical results. Since MVPA considers multiple voxels at the same time, it can lead to the identification of additional regions that collectively have a discriminative power but not individually (which is the case for GLM). Thus, the areas reported by Moshfeghi et al. [42] but not found in our PM results would be those with significant differences in mean activation and not necessarily where the spatial pattern of activation could be decoded. However, one region was highlighted in both studies, i.e. the left inferior frontal gyrus region, which indicates that this region both changes its mean activation and its spatial distribution of activity with IN. Future research can examine this property more closely.

With this in mind, we now describe the voxels presented in Table 1. The ten areas presented in Table 1 are commonly related to the domains of text/semantic processing (precentral gyrus, right inferior frontal gyrus), vision (middle occipital gyrus, lingual gyrus, fusiform gyrus), complex cognition (medial frontal gyrus), executive function (left inferior frontal gyrus) and reward (middle frontal gyrus). The finding of areas involved in complex cognition suggests that a realisation of an IN involves high-level cognition [23]. For example, the medial frontal cortex (Brodmann Area 10) is at the most anterior section of the frontal cortex. Models of branching control of cognition suggest this area serves at the top of a processing hierarchy to assemble information across different processing strands. Similarly, the executive function includes processes such as working memory and keeping track of task rules. Thus the region of left inferior frontal gyrus, known as the dorsolateral prefrontal cortex, suggests that there is a difference in executive function depending upon whether there is an IN or no-IN. However, an unexpected finding is that a realisation of an IN need appears to be encoded in visual and textual processing areas. While these areas are consistent with performing the required tasks of reading questions and answers, It is not clear why it should encode a realisation of an IN. We would like to investigate this in the future. In summary, it would seem that these ten regions are distributed across the brain and involve both (i) high-level cognition, which could represent the concept of IN per se, and (ii) low level perceptual and textual processing where the decoding results from differences in the depth of processing IN vs no-IN items.

4.4 Prediction Accuracy of IN Realisation

Table 2 shows the classification performance averaged over the twenty-four participants of the study for both the GM condition when a common feature set of voxels was used across all participants and for the PM condition when a unique feature set of voxels was obtained for each participant. The table reports the accuracy of the model (i.e. the fraction of items in the test set for which the models' predictions were correct) using 10-fold cross-validation. We also performed a paired Wilcoxon test between the predictions obtained for each model to check the significance of the difference with its baseline. We use (**) and (††) to denote the fact that a model trained on a set of features had results different from that of "BL" and "GM" with the confidence level ($p < 0.01$) respectively.

Table 2: This table shows the prediction accuracy of our methods (presented as columns). The percentage of improvement over the baseline is presented in the parenthesis. The best performing model is highlighted in bold.

Method	*BL*	*GM*	*PM*
Accuracy	50%	67.54%**	**79.68%**** ††

The key findings arising from our study are that brain activity could be used to predict whether participants were realising an IN (addressing RQ1). If the brain activity of a set of common regions to all participants was used (i.e. GM) then performance was (statistically) significantly above chance. In addition, the prediction accuracy (statistically) significantly improved over both chance and GM when the brain activity of a set of unique regions to each participant was used (i.e. PM) (addressing RQ2). Our findings are encouraging and indicate that it is possible to build proactive IR systems in which the IN realisation can be detected from brain signals.

5 DISCUSSION AND CONCLUSION

The main goal of Information Retrieval (IR) systems is to satisfy searchers' Information Needs (IN). However, satisfying INs is a formidable challenge, due to the inherent complexity and ambiguity associated with the IN concept. As a result of this, searchers' queries (i.e. searchers' representation of their IN) may not sufficiently define the characteristics of relevant documents or even any relevant information [13]. Recently, there is an increased interest in the research community in developing zero-query (or proactive) IR systems aiming to not only better understand, but even anticipate users' INs [9, 49]. However, current approaches mainly consider the contextual or behavioural information of users to predict their IN. While this is intuitively a promising approach, it could potentially create an information overload situation for users, in particular, if users are not experiencing an IN when the information is presented to them. It is essential, therefore, to be able to detect when INs occur.

In this paper, we investigated the possibility of predicting a realisation of an IN from brain signals. Previous research has established that INs physically manifest themselves with specific brain activity. With this work we take the next step, showing that monitoring brain activity can lead to accurate predictions of IN occurrence. We have conducted a "within-subjects" design experiment where we measured the brain activity of twenty-four participants while they performed a Question Answering (Q/A) Task. The independent variable was the realisation of an information need, with two levels: an information need was realised (IN), and no information need was realised (no-IN), and the dependent variable was the BOLD signal. From the TREC-8 and TREC 2001 Q/A Tracks, a set of questions was carefully designed so that we obtained an equal number of responses for expressing an answer to the question and expressing a need to search for more information to answer the question. We then test two methods for predicting a realisation of an information need. A Generalised method (GM) that uses a predefined set of

brain regions for all participants as well as a Personalised method (PM) that uses a unique set of brain regions for each participant.

Given that PM provided a unique ROI for each individual, we then investigated the probability that the ROIs obtained for each individual were overlapping, presented as a probability map. When we used a criterion that a voxel can enter the map if at least one participant obtained this voxel, 87.9% of all voxels in the brain volume were included in the map. This is consistent with previous studies suggesting that representational and memory systems are widely distributed in the brain [48]. When we increased the criterion to 0.35 to examine highly overlapping regions, 0.6% of all voxels in the brain volume were included in the map, revealing ten regions. An analysis of these ten regions suggested that these regions are distributed across the brain and involve both (i) high-level cognition, which could represent the concept of IN per se, and (ii) low level perceptual and textual processing where the decoding results from differences in the depth of processing IN vs no-IN items. Such a finding is consistent with theoretical views that information processing in the brain involves extensive feedforward and feedback between brain regions [18].

Our findings show that brain activity could be used to predict whether participants were in a state of information need (addressing RQ1). The prediction accuracy depended upon the choice of brain regions used to learn the model. If the brain activity of a set of common regions to all participants was used (i.e. GM) then performance was (statistically) significantly above chance. Also, the prediction accuracy (statistically) significantly improved over both chance and GM when the brain activity of a set of unique regions to each participant was used (i.e. PM) (addressing RQ2). This result shows the benefit of PM for the prediction of a realisation of IN from brain activity.

The model proposed by Moshfeghi et al. [42] postulated that activation in the dorsal posterior cingulate cortex (PCC) represented a switch to broad attention directed to the external world [35] and this was an important neural correlate of a realisation of an IN. The current results advance our understanding of this model. Firstly, functional connectivity exists between the PCC to the inferior frontal gyrus (dorsolateral prefrontal cortex) [34], and anatomical connectivity exists between the PCC and the medial frontal cortex [46, 47]. Thus, the encoding of a realisation of an IN in these highly cognitive regions can be regulated by activity in the PCC. In addition, the current finding that visual and textual regions encode a realisation of an IN are consistent with the modulatory activity of the PCC in exerting top-down influences on perception through the regulation of extended brain networks.

In conclusion, the results of our experiment not only show that we can predict a realisation of an IN from brain activity but also substantially advance our understanding of the neural mechanisms involved in such a phenomenon. The encouraging results obtained from GM and especially PM in predicting a realisation of an IN provides an essential and fundamental step towards the next-generation of proactive IR systems based on brain signals. These results point to the beneficial use of fMRI to examine activity within the entire brain with high spatial resolution to reveal areas involved in a realisation of an IN. Given that this paper is the first attempt to predict a realisation of an IN from brain signals, such a high spatial resolution was crucial to investigate our hypotheses thoroughly.

Our findings also inform what other neuroimaging techniques might be effective for further study of a realisation of an IN. This is because brain imaging technologies have varied effectiveness in revealing activity at different depth levels in the brain. For example, the left dorsolateral prefrontal cortex, revealed in the present study, is a brain region that is relatively superficial and thus amenable to study with a variety of other brain imaging technologies. In future work, we want to continue to advance our understanding of brain activity for a realisation of an IN across other IR scenarios. Further, we would like to translate our current findings to mobile brain imaging technologies, such as EEG or functional Near-Infrared Spectroscopy (fNIRS) and possibly build proactive search engines that use brain activity for realising the occurrence of an IN.

REFERENCES

[1] Marco Allegretti, Yashar Moshfeghi, Maria Hadjigeorgieva, Frank E. Pollick, Joemon M. Jose, and Gabriella Pasi. 2015. When Relevance Judgement is Happening?: An EEG-based Study. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*. ACM, New York, NY, USA, 719–722. https://doi.org/10.1145/2766462.2767811

[2] Ioannis Arapakis, Konstantinos Athanasakos, and Joemon M. Jose. 2010. A Comparison of General vs Personalised Affective Models for the Prediction of Topical Relevance. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '10)*. ACM, New York, NY, USA, 371–378. https://doi.org/10.1145/1835449.1835512

[3] Ioannis Arapakis, Ioannis Konstas, and Joemon M. Jose. 2009. Using Facial Expressions and Peripheral Physiological Signals As Implicit Indicators of Topical Relevance. In *Proceedings of the 17th ACM International Conference on Multimedia (MM '09)*. ACM, New York, NY, USA, 461–470. https://doi.org/10.1145/1631272.1631336

[4] Ioannis Arapakis, Yashar Moshfeghi, Hideo Joho, Reede Ren, David Hannah, and Joemon M. Jose. 2009. Enriching User Profiling with Affective Features for the Improvement of a Multimodal Recommender System. In *Proceedings of the ACM International Conference on Image and Video Retrieval (CIVR '09)*. ACM, New York, NY, USA, Article 29, 29:1–29:8 pages. https://doi.org/10.1145/1646396.1646433

[5] I. Arapakis, Y. Moshfeghi, H. Joho, R. Ren, D. Hannah, and J. M. Jose. 2009. Integrating facial expressions into user profiling for the improvement of a multimodal recommender system. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*. 1440–1443. https://doi.org/10.1109/ICME.2009.5202773

[6] Marcia J. Bates. 1998. Indexing and access for digital libraries and the internet: Human, database, and domain factors. *J. Am. Soc. Inf. Sci.* 49, 13 (12 Dec. 1998), 1185–1205. https://doi.org/10.1002/(sici)1097-4571(1998110)49:13%3C1185::aid-asi6%3E3.0.co;2-v

[7] N. J. Belkin, R. N. Oddy, and H. M. Brooks. 1982. ASK Fir Information Retrieval: Part II. Results of a Design Study. *Journal of Documentation* 38, 3 (1982), 145–164.

[8] N. J. Belkin, R. N. Oddy, and H. M. Brooks. 1997. Ask for Information Retrieval: Part I.: Background and Theory. (1997), 299–304. http://dl.acm.org/citation.cfm?id=275537.275703

[9] Jan R. Benetka, Krisztian Balog, and Kjetil Nørvåg. 2017. Anticipating Information Needs Based on Check-in Activity. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining (WSDM '17)*.

[10] Sumit Bhatia, Debapriyo Majumdar, and Nitish Aggarwal. 2016. *Proactive Information Retrieval: Anticipating Users' Information Need.* Springer International Publishing, Cham, 874–877. https://doi.org/10.1007/978-3-319-30671-1_84

[11] Christine L. Borgman. 2003. *From Gutenberg to the Global Information Infrastructure: Access to Information in the Networked World.* MIT Press, Cambridge, MA, USA.

[12] Charles Cole. 2011. A theory of information need for information retrieval that connects information to knowledge. *JASIST* 62, 7 (2011), 1216–1231.

[13] Charles Cole, Charles-Antoine Julien, and John E Leide. 2010. An associative index model for hypertext Internet search based on Vannevar BushâĂŹs Memex machine: An exploratory case study. *Information Research* 15, 3 (2010), 15–3.

[14] Marc N. Coutanche. 2013. Distinguishing multi-voxel patterns and mean activation: Why, how, and what does it tell us? *Cognitive, Affective, & Behavioral Neuroscience* 13, 3 (2013), 667–673. https://doi.org/10.3758/s13415-013-0186-2

[15] Angelika Dimoka. 2012. How to Conduct a Functional Magnetic Resonance (fMRI) Study in Social Science Research. *MIS Q.* 36, 3 (Sept. 2012), 811–840. http://dl.acm.org/citation.cfm?id=2481655.2481664

[16] Manuel JA Eugster, Tuukka Ruotsalo, Michiel M Spapé, Oswald Barral, Niklas Ravaja, Giulio Jacucci, and Samuel Kaski. 2016. Natural brain-information interfaces: Recommending information by relevance inferred from human brain signals. *Scientific Reports* 6 (2016).

[17] Manuel J.A. Eugster, Tuukka Ruotsalo, Michiel M. Spapé, Ilkka Kosunen, Oswald Barral, Niklas Ravaja, Giulio Jacucci, and Samuel Kaski. 2014. Predicting Term-relevance from Brain Signals. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '14)*. ACM, New York, NY, USA, 425–434. https://doi.org/10.1145/2600428.2609594

[18] Karl Friston. 2010. The free-energy principle: a unified brain theory? *Nature reviews. Neuroscience* 11, 2 (13 Feb. 2010), 127–138. https://doi.org/10.1038/nrn2787

[19] K.J Friston, C Buechel, G.R Fink, J Morris, E Rolls, and R.J Dolan. 1997. Psychophysiological and Modulatory Interactions in Neuroimaging. *NeuroImage* 6, 3 (1997), 218 – 229. https://doi.org/10.1006/nimg.1997.0291

[20] Rainer Goebel. [n. d.]. *BrainVoyager QX, Vers.2.1, Brain Innovation B.V.* Maastricht, Netherlands.

[21] J. V. Haxby, M. I. Gobbini, M. L. Furey, A. Ishai, J. L. Schouten, and P. Pietrini. 2001. Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science (New York, N.Y.)* 293, 5539 (28 Sept. 2001), 2425–2430. https://doi.org/10.1126/science.1063736

[22] Birger Hjørland. 2010. The foundation of the concept of relevance. *J. Am. Soc. Inf. Sci.* 61, 2 (1 Feb. 2010), 217–237. https://doi.org/10.1002/asi.21261

[23] Peter Ingwersen. 1996. Cognitive Perspectives of Information Retrieval Interaction: Elements of a Cognitive IR Theory. *Journal of Documentation* 52, 1 (1996), 3–50.

[24] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately Interpreting Clickthrough Data As Implicit Feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '05)*. ACM, New York, NY, USA, 154–161. https://doi.org/10.1145/1076034.1076063

[25] Jukka-Pekka Kauppi, Melih Kandemir, Veli-Matti Saarinen, Lotta Hirvenkari, Lauri Parkkonen, Arto Klami, Riitta Hari, and Samuel Kaski. 2015. Towards brain-activity-controlled information retrieval: Decoding image relevance from MEG signals. *NeuroImage* 112 (2015), 288–298. https://doi.org/10.1016/j.neuroimage.2014.12.079

[26] S. Sathiya Keerthi and Chih-Jen Lin. 2003. Asymptotic Behaviors of Support Vector Machines with Gaussian Kernel. *Neural Comput.* 15, 7 (July 2003), 1667–1689. https://doi.org/10.1162/089976603321891855

[27] Diane Kelly and Nicholas J Belkin. 2002. A user modeling system for personalized interaction and tailored retrieval in interactive IR. *Proceedings of the American Society for Information Science and Technology* 39, 1 (2002), 316–325.

[28] Diane Kelly and Nicholas J. Belkin. 2004. Display Time As Implicit Feedback: Understanding Task Effects. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '04)*. ACM, New York, NY, USA, 377–384. https://doi.org/10.1145/1008992.1009057

[29] Jürgen Koenemann and Nicholas J. Belkin. 1996. A Case for Interaction: A Study of Interactive Information Retrieval Behavior and Effectiveness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '96)*. ACM, New York, NY, USA, 205–212. https://doi.org/10.1145/238386.238487

[30] N. Kriegeskorte and P. A. Bandettini. 2007. Analyzing for information, not activation, to exploit high-resolution fMRI. *NeuroImage* 38 (2007), 649–662.

[31] Nikolaus Kriegeskorte, Rainer Goebel, and Peter Bandettini. 2006. Information-based functional brain mapping. *Proceedings of the National Academy of Sciences of the United States of America* 103, 10 (7 March 2006), 3863–3868. https://doi.org/10.1073/pnas.0600244103

[32] Carol C. Kuhlthau. 1993. A Principle of Uncertainty for Information Seeking. *Journal of Documentation* 49, 4 (1993), 339–355.

[33] KS Lashley. 1952. Functional interpretation of anatomic patterns. *Research publications-Association for Research in Nervous and Mental Disease* 30 (1952), 529.

[34] Robert Leech, Salwa Kamourieh, Christian F. Beckmann, and David J. Sharp. 2011. Fractionating the Default Mode Network: Distinct Contributions of the Ventral and Dorsal Posterior Cingulate Cortex to Cognitive Control. *Journal of Neuroscience* 31, 9 (2011), 3217–3224. https://doi.org/10.1523/JNEUROSCI.5626-10.2011

[35] Robert Leech and David J. Sharp. 2014. The role of the posterior cingulate cortex in cognition and disease. *Brain : a journal of neurology* 137, Pt 1 (01 Jan. 2014), 12–32. https://doi.org/10.1093/brain/awt162

[36] Lori Lorigo, Maya Haridasan, Hrönn Brynjarsdóttir, Ling Xia, Thorsten Joachims, Geri Gay, Laura Granka, Fabio Pellacini, and Bing Pan. 2008. Eye tracking and online search: Lessons learned and challenges ahead. *J. Am. Soc. Inf. Sci.* 59, 7 (1 May 2008), 1041–1052. https://doi.org/10.1002/asi.20794

[37] Robert McGill, John W. Tukey, and Wayne A. Larsen. 1978. Variations of Box Plots. *The American Statistician* 32, 1 (1978), 12–16. https://doi.org/10.2307/2683468

[38] Yashar Moshfeghi and Joemon M. Jose. 2013. An Effective Implicit Relevance Feedback Technique Using Affective, Physiological and Behavioural Features. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '13)*. ACM, New York, NY, USA, 133–142. https://doi.org/10.1145/2484028.2484074

[39] Yashar Moshfeghi and Joemon M. Jose. 2013. On Cognition, Emotion, and Interaction Aspects of Search Tasks with Different Search Intentions. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13)*. ACM, New York, NY, USA, 931–942. https://doi.org/10.1145/2488388.2488469

[40] Yashar Moshfeghi, Luisa R. Pinto, Frank E. Pollick, and Joemon M. Jose. 2013. Understanding Relevance: An fMRI Study. In *Advances in Information Retrieval: 35th European Conference on IR Research, ECIR 2013, Moscow, Russia, March 24-27, 2013. Proceedings*, Pavel Serdyukov, Pavel Braslavski, Sergei O. Kuznetsov, Jaap Kamps, Stefan Rüger, Eugene Agichtein, Ilya Segalovich, and Emine Yilmaz (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 14–25. https://doi.org/10.1007/978-3-642-36973-5_2

[41] Yashar Moshfeghi and Frank E. Pollick. 2018. Search Process As Transitions Between Neural States. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1683–1692. https://doi.org/10.1145/3178876.3186080

[42] Yashar Moshfeghi, Peter Triantafillou, and Frank E. Pollick. 2016. Understanding Information Need: An fMRI Study. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. ACM, New York, NY, USA, 335–344. https://doi.org/10.1145/2911451.2911534

[43] Javed Mostafa, Vincent Carrasco, Chris Foster, and Kelly Giovenallo. 2015. *Identifying Neurological Patterns Associated with Information Seeking: A Pilot fMRI Study*. Springer International Publishing, Cham, 167–173. https://doi.org/10.1007/978-3-319-18702-0_22

[44] Javed Mostafa and Jacek Gwizdka. 2016. Deepening the Role of the User: Neuro-Physiological Evidence As a Basis for Studying and Improving Search. In *Proceedings of the 2016 ACM on Conference on Human Information Interaction and Retrieval (CHIIR '16)*. ACM, New York, NY, USA, 63–70. https://doi.org/10.1145/2854946.2854979

[45] R. C. Oldfield. 1971. The assessment and analysis of handedness: The Edinburgh inventory. *Neuropsychologia* 9, 1 (March 1971), 97–113. https://doi.org/10.1016/0028-3932(71)90067-4

[46] Joseph M Orr, Harry R Smolker, and Marie T Banich. 2015. Organization of the human frontal pole revealed by large-scale DTI-based connectivity: implications for control of behavior. *PloS one* 10, 5 (2015), e0124797.

[47] Michael Petrides and Deepak N. Pandya. 2007. Efferent Association Pathways from the Rostral Prefrontal Cortex in the Macaque Monkey. *Journal of Neuroscience* 27, 43 (2007), 11573–11586. https://doi.org/10.1523/JNEUROSCI.2419-07.2007

[48] Jesse Rissman and Anthony D Wagner. 2012. Distributed representations in memory: insights from functional brain imaging. *Annual review of psychology* 63 (2012), 101–128.

[49] Yang Song and Qi Guo. 2016. Query-Less: Predicting Task Repetition for NextGen Proactive Search and Recommendation Engines. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 543–553. https://doi.org/10.1145/2872427.2883020

[50] J. Talairach and P. Tournoux. 1988. *Co-planar Stereotaxic Atlas of the Human Brain: 3-dimensional Proportional System : an Approach to Cerebral Imaging*. Thieme. https://books.google.co.uk/books?id=pYFiQgAACAAJ

[51] Robert S. Taylor. 1968. Question-Negotiation and Information Seeking in Libraries. *College and Research Libraries* 29 (1968), 178–194.

[52] Michael T. Todd, Leigh E. Nystrom, and Jonathan D. Cohen. 2013. Confounds in multivariate pattern analysis: Theory and rule representation case study. *NeuroImage* 77 (15 Aug. 2013), 157–165. https://doi.org/10.1016/j.neuroimage.2013.03.039

[53] Peter Ingwersen und Kalervo Järvelin. 2006. *The Turn: Integration of Information Seeking and Retrieval in Context*. Springer, 2005. xiv, 448 S. ISBN 1-4020-3850-X.

[54] Chih wei Hsu, Chih chung Chang, and Chih jen Lin. 2010. A practical guide to support vector classification.

[55] Ryen White, Joemon M. Jose, and Ian Ruthven. 2003. A task-oriented study on the influencing effects of query-biased summarisation in web searching. *Inf. Process. Manage.* 39, 5 (2003), 707–733. https://doi.org/10.1016/S0306-4573(02)00033-X

[56] Ryen W. White. 2004. *Implicit Feedback for Interactive Information Retrieval*. Ph.D. Dissertation. University of Glasgow.

[57] Ryen W. White and Diane Kelly. 2006. A Study on the Effects of Personalization and Task Information on Implicit Feedback Performance. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM '06)*. ACM, New York, NY, USA, 297–306. https://doi.org/10.1145/1183614.1183659

[58] T. D. Wilson. 1981. On User Studies and Information Needs. *Journal of Documentation* 37, 1 (0 March 1981), 3–15. http://www.eric.ed.gov/ERICWebPortal/detail?accno=EJ248909

[59] Qingbai Zhao, Zhijin Zhou, Haibo Xu, Shi Chen, Fang Xu, Wenliang Fan, and Lei Han. 2013. Dynamic neural network of insight: a functional magnetic resonance imaging study on solving Chinese 'chengyu' riddles. *PloS one* 8, 3 (2013), e59351.

GhostLink: Latent Network Inference for Influence-aware Recommendation

Subhabrata Mukherjee*
Amazon, USA
subhomj@amazon.com

Stephan Günnemann
Technical University of Munich, Germany
guennemann@in.tum.de

ABSTRACT

Social influence plays a vital role in shaping a user's behavior in online communities dealing with items of fine taste like movies, food, and beer. For online recommendation, this implies that users' preferences and ratings are influenced due to other individuals. Given only time-stamped reviews of users, can we find out who-influences-whom, and characteristics of the underlying influence network? Can we use this network to improve recommendation?

While prior works in social-aware recommendation have leveraged social interaction by considering the *observed* social network of users, many communities like Amazon, Beeradvocate, and Ratebeer *do not have explicit user-user links*. Therefore, we propose GhostLink, an unsupervised probabilistic graphical model, to automatically *learn* the latent influence network underlying a review community – given only the temporal traces (timestamps) of users' posts and their content. Based on extensive experiments with four real-world datasets with 13 million reviews, we show that GhostLink improves item recommendation by around 23% over state-of-the-art methods that do not consider this influence. As additional use-cases, we show that GhostLink can be used to differentiate between users' latent preferences and influenced ones, as well as to detect influential users based on the learned influence graph.

CCS CONCEPTS

• **Information systems → Social recommendation**; *Collaborative filtering*; • **Computing methodologies → Topic modeling**.

KEYWORDS

Generative Model; Social Influence; Review Community; Content Analysis; Social Recommendation

ACM Reference Format:
Subhabrata Mukherjee and Stephan Günnemann. 2019. GhostLink: Latent Network Inference for Influence-aware Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313449

*Work done at Max Planck Institute, Germany prior to joining Amazon.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313449

1 INTRODUCTION

Traditional works in recommender systems that build upon collaborative filtering [19] exploit that similar users have similar rating behavior and facet preferences. Recent works use review content [25, 28, 37] and temporal patterns [11, 12, 29] to extract further cues. All of these works assume the users to behave independently of each other. In a social community, however, users are often influenced by the activities of their friends and peers. How can we detect this influence in online communities?

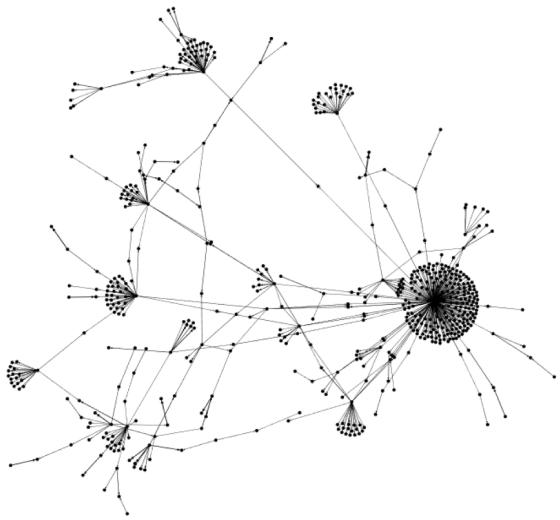


Figure 1: Given only timestamped reviews of users in the Beeradvocate community without any explicit user-user link/interaction, GhostLink extracts this latent influence network (of top K influencers) based on opinion conformity. This is compactly represented by a Maximum Weighted Spanning Forest (MWSF) preserving 99.4% of the influence mass from 73.4% of the edges of the inferred influence network depicting a tree-like structure of influence.

One way to answer this question is to exploit the *observed* social network or interaction of users — like friend circles in Facebook, the follow graph in Twitter, and trust relations in Epinion. Recent works [5, 14, 16, 20, 22, 27, 33, 34, 41, 42] leverage such *explicit* user-user relations or the observed social circle to propose social-network based recommendation. Similarly, in the field of citation networks, [2] attempt to extract citation influence given the *explicit network* of who-cited-whom.

However, there is one big catch: many online review communities like Amazon or Beeradvocate do *not* have any explicit social network – thus, making the above methods not applicable. Can we infer the influence network based on other signals in the data?

Amazon Movies
U1: style intense pretentious non-linear narrative rapid editing
U2: non-linear narrative crazy flashback scene randomly interspersed
Beeradvocate
U1: cloudy reddish amber color huge frothy head aroma spicy
U2: hazy golden amber dissipating head earthy aroma pepper clove

Table 1: Sample (influenced) review snippets extracted by GhostLink from two communities: user U1's review is influenced by U2.

While some recent works [13, 21, 23, 24] model implicit relationships, they are limited to the historical rating behavior of users, ignoring the textual information. Similarly, works in information diffusion over latent networks model temporal traces ignoring the textual information [6, 7, 9, 30] and make some strong assumptions like a homogeneous network with static transmission rates. These techniques being agnostic of the context fail to capture complex interactions resulting in sparse networks. Some recent works on text-based diffusion [4, 15, 36] model context. However, they also make some strong assumptions regarding the topics of diffusion being known apriori and the network being explicit. Most importantly, none of these works are geared for item recommendation, nor do they study the characteristics of review communities.

In contrast, in this work, we leverage opinion conformity based on writing style as an indication of influence: where a user echoes/copies facet descriptions from peers (called influencers) across multiple items. This is a common setting for communities dealing with items of fine taste like movies, beer, food and fine arts where users often co-review multiple items. Our informal goals are:

Informal Problem 1. *Given only timestamped reviews of users in online communities,* ***extract*** *the underlying influence network of who-influences-whom based on opinion conformity, and* ***analyze*** *the characteristics of this influence network.*

Informal Problem 2. ***Leverage*** *the implicit social influence (network) to improve item rating prediction based on peer activities.*

To answer these questions, we propose GhostLink, an unsupervised probabilistic graphical model, that automatically extracts the (latent) influence graph underlying a review community.

Key idea and approach: Consider two users reviewing a movie in a review community. The first user expressed fascination for the movie's 'non-linear narrative style', 'structural complexity', and 'cinematography' as outlined in the content of her review. Later, following this review, a second user also echoed similar concepts such as 'seamless narrative', 'style', and 'matured cinematography'. That is, the second review closely resembles the first one *conceptually* in terms of *facet descriptions* – not simply by using the same words. While for a *single* item this could be simply due to chance, *a repeated occurrence of this pattern across multiple items* – where the second user reviewed an item some time *after* the first user echoing *similar facet descriptions* – gives an indication of influence. A user could be influenced by several users for different facets in her review. GhostLink models this notion of multiple influence common in communities dealing with items of fine taste like movies, food and beer where users often co-review multiple items. Table 1 shows a snapshot of (influenced) review snippets extracted by GhostLink.

Based on this idea, we propose a probabilistic model that exploits the facet descriptions and preferences of users — based on principles similar to Latent Dirichlet Allocation — to learn an influence graph. Since the influencers for a given user and facets are unobserved, all these aspects are learned solely based on their review content and their temporal footprints (timestamps).

Figure 1 shows such an influence graph extracted by GhostLink from the Beeradvocate data. Analyzing these graphs gives interesting insights: There are only a few users who influence most of the others, and the distribution of influencers vs. influencees follows a power-law like distribution. Furthermore, most of the mass of this influence graph is concentrated in giant tree-like component(s). We use such influence graphs to perform influence-aware item recommendation; and we show that GhostLink outperforms state-of-the-art baselines that do not consider latent influence. Moreover, we use the influence graph to find influential users and to distinguish between users' latent facet preferences from that of induced/influenced ones. Overall, our contributions are:

- **Model:** We propose an unsupervised probabilistic generative model GhostLink based on Latent Dirichlet Allocation to learn a latent influence graph in online communities without requiring explicit user-user links or a social network. This is the first work that solely relies on timestamped review data.
- **Algorithm:** We propose an efficient algorithm based on Gibbs sampling [10] to estimate the hidden parameters in GhostLink that empirically demonstrates fast convergence.
- **Experiments:** We perform large-scale experiments in four communities with 13 million reviews, 0.5 mil. items, and 1 mil. users where we show improved recommendation for item rating prediction by around 23% over state-of-the-art methods. Moreover, we analyze the properties of the influence graph and use it for use-cases like finding influential members in the community.

2 GHOSTLINK: INFLUENCE-FACET MODEL

Our goal is to learn an influence graph between users based on their review content (specifically, overlap of their facet preferences) and timestamps only. The underlying assumption is that when a user u is influenced by a user v, u's facet preferences are influenced by the ones of v. Since the only signal is the textual information of the reviews – and their inferred latent facet distributions (also known as topic distributions in the context of LDA) – we argue that influence is reflected by the used/echoed words and facets.

While classical user topic models assume that each word of a document is associated with a topic/facet that follows the user's preference, we assume that the topic/facet of each word might be based on the preferences of *other* users as well – the influencers. Inspired by this idea, we first describe the generative process of GhostLink followed by the explanation of the inference procedure.

Generative Process. Consider a corpus D of reviews written by a set of users U at timestamps T on a set of items I. The subset of reviews for item $i \in I$ is denoted with $D_i \subseteq D$. Let $d \in D_i$ be a review on item $i \in I$, we denote with u_d the user and with t_d the timestamp of the review. All the reviews on an item i are assumed to be ordered by timestamps. Each review d consists of a sequence of N_d words denoted by $d = \{w_1, \ldots, w_{N_d}\}$, where each word is drawn from a vocabulary W having unique words indexed by $\{1, \ldots, W\}$. The number of latent facets/topics corresponds to K.

In most review communities, a user browses through other reviews on an item before making a decision (say) at time t. Therefore,

the set of users and corresponding reviews that could potentially influence the given user's perspective on an item i consists of all the reviews d' written at time $t_{d'} < t$. We call the corresponding set of users – the potential influence set $IS_{u,i} = \{u' \in U \mid \exists d, d' \in D_i : u = u_d \wedge u' = u_{d'} \wedge t_{d'} < t_d\}$ for user u and item i.

In our model, each user is equipped with a latent facet preference distribution θ_u, whose elements $\theta_{u,k}$ denote the preference of user u for facet $k \in K$. That is, θ_u is a K-dimensional categorical distribution; we draw it according to $\theta_u \sim Dirichlet_K(\alpha)$ with concentration parameter α. These distributions later govern the generation of the review text (similar to LDA).

Furthermore, for each user an influence distribution ψ_u is considered, whose elements $\psi_{u,v}$ depict the influence of user v on user u. That is ψ_u represents a U-dimensional categorical distribution – *and all ψ_* together build the influence graph we aim to learn* (see also Sec. 3.4). Similar to above we define $\psi_u \sim Dirichlet_U(\rho)$.

When writing a review, a user u can decide to write an original review based on her latent preferences θ_u — or be influenced by someone's perspective from her influence set for the given item; that is, using the preferences of θ_v for some $v \in IS_{u,i}$. Since a user might not be completely influenced by other users, we allow each word of the review to be either original or based on other influencers. More precise: For each word of the review, we consider a random variable $s \sim Bernoulli(\pi_u)$ that denotes whether it is original or based on influence, where π_u intuitively denotes the 'vulnerability' of the user u to get influenced by others.

If $s = 0$ the user uses her own latent facet preferences. That is, following the idea of standard topic models, the latent facet for this word is drawn according to $z \sim Categorical(\theta_u)$. If $s = 1$, user u writes under influence. In this case, the user chooses potential influencer(s) v according to the strength of influence given by ψ_u. Since for a specific item i, the user u can only be influenced by users in $IS_{u,i}$ who have written a review *before* her, we write $v \sim Categorical(\psi_u \cap IS_{u,i})$ to denote restriction of the domain of ψ_u to the currently considered influence set.

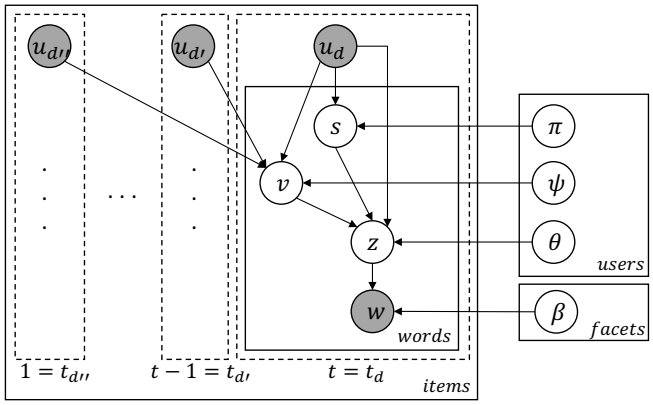


Figure 2: Plate diagram for the generative process. Each dashed box indicates a single review.

Given the sampled user v, the latent facet for this word should be drawn according to v's preferences. Now, we are faced with a modeling choice. We can use the influencer's overall facet distribution θ_v (as is used in citation-network based topic models). However,

Algorithm 1: Generative process for influence – facet model.

1. Draw $\theta_u \sim \text{Dirichlet}_K(\alpha)$ // latent facet preference of each user
2. Draw $\psi_u \sim \text{Dirichlet}_U(\rho)$ // influencer distribution of each user
3. Draw $\pi_u \sim \text{Beta}(\eta)$ // vulnerability of each user to be influenced
4. Draw $\beta_k \sim \text{Dirichlet}_W(\gamma)$ // word distribution of each facet

for *each item $i \in I$* **do**
  for *each review $d \in D_i$ on i at time t by user u* **do**
    for *each word w in d* **do**
      5. Draw $s \sim \text{Bernoulli}(\pi_u)$
      if $s = 0$ **then**
        6. $\theta' = \theta_u$ // use latent facet preference of user
      if $s = 1$ **then**
        7. Draw $v \sim \text{Categorical}(\psi_u \cap IS_{u,i})$
        8. $\theta' = \tilde{\theta}_v^{d'}$ // use the influencer's facet preference
        /* where $\tilde{\theta}_v^{d'}$ is the facet distribution *used* by v for review d' written at time $t' < t$ */
      9. Draw $z \sim \text{Categorical}(\theta')$
      10. Draw $w \sim \text{Categorical}(\beta_z)$

by using θ_v, one considers v's *generic* facet distribution – which might be very unrelated to the item under consideration. That is, while v might prefer specific facets, in his actual review d' about item i these facets might have not been used. And accordingly, since the user u only sees the observed review d' – and not the latent facet distribution of v – the user u cannot be influenced by facets which have not been considered. Thus, in our model, instead of considering the influencer's (generic) facet distribution θ_v, we consider the facet distribution that the influencer has actually *used* for writing his review d' for the given item i. Since the review of the user v has already been generated (otherwise the user would not be in $IS_{u,i}$), the used facets z for each word of his review are known. Thus, instead of considering θ_v, we consider the 'observed' facet distribution based on the actual review, denoted with $\tilde{\theta}_v^{d'}$. Given this distribution, we sample $z \sim Categorical(\tilde{\theta}_v^{d'})$. Since the model samples an influencer for each facet, a user can have *multiple influencers* corresponding to multiple facets in his review.

In summary, in the above process, the user u either draws the facet z from θ_u (if $s = 0$) or $\tilde{\theta}_v^{d'}$ (if $s = 1$). Given facet z, we draw the actual word $w \sim Categorical(\beta_z)$ following the generative process of Latent Dirichlet Allocation [1]. As usual, $\beta_z \sim Dirichlet_W(\gamma)$ denotes corresponding per-facet word distributions.

Overall, the user's review can be regarded as being generated by a *mixture* of her latent preferences and preferences of her influencers. Algorithm 1 summarizes the generative process, and the graphical model is illustrated in Figure 2, where we indicated with u_d the (observed) user for each review.

3 JOINT PROBABILISTIC INFERENCE

We now describe the inference procedure for GhostLink. That is, given the set of all reviews (and their timestamps), we aim to infer the latent variables. To not clutter notation, we drop the indices of variables when it is clear from context (e.g. θ_u is abbreviated as θ).

Let S, V, Z be the set of all latent variables corresponding to the influence variables s, influencers v, and facets z. Let W' denote the set of latent variables corresponding to the observed words, and

U' the set of latent variables corresponding to the observed users[1]. The joint probability distribution of our model is:

$$P(S, V, Z, W', \theta, \beta, \psi, \pi | U'; \alpha, \gamma, \rho, \eta) \propto \prod_{u \in U} \Big(P(\pi; \eta) \cdot P(\psi; \rho) \cdot P(\theta; \alpha) \Big) \cdot \prod_{k \in K} P(\beta_k; \gamma) \cdot \prod_{i \in I} \prod_{d \in D_i} \Big(\prod_{s \in S} P(s | \pi_{u_d}) \cdot \prod_{v \in V} P(v | \psi_{u_d})^{\mathbb{I}(s=1)} \cdot \prod_{z \in Z} \Big(P(z | \theta_{u_d})^{\mathbb{I}(s=0)} \cdot P(z | \tilde{\theta}_v^{d'})^{\mathbb{I}(s=1)} \Big) \cdot \prod_{w \in W'} P(w | \beta_z) \Big) \quad (1)$$

Since exact inference is intractable, we have to resort to approximate inference. For this purpose, we perform Collapsed Gibbs Sampling [10]. In Gibbs sampling, the conditional distribution for each hidden variable is computed based on the current assignment of the other hidden variables. The values for the latent variables are sampled repeatedly from this conditional distribution until convergence. In our problem setting we have three sets of latent variables corresponding to S, V and Z respectively – the remaining variables θ, β, ψ, π are marginalized out (collapsed).

Given the current assignment of random variables, we use the shortcuts: $n(u, s)$ denotes the count of words written by u with influence variable $s \in \{0, 1\}$. $n(u, v, s = 1)$ denotes the count of words written by u under influence from v (i.e. $s = 1$) in the community across all items and facets. $n(u, z, s = 0)$ denotes the number of times u wrote facet z for any word based on her latent preferences (i.e. $s = 0$). $n(v_{d'}, z)$ denotes the count of facet z in review $v_{d'}$, and $n(z, w)$ denotes the number of times word w is used with facet z.

Collapsing. We first marginalize out the remaining variables as mentioned above. Exploiting conjugacy of the Categorical and Dirichlet distributions, we can integrate out π, ψ, θ, and β from the above distribution to obtain the four posterior distributions

$$P(S|U'; \eta) = \frac{\Gamma(\sum_s \eta) \prod_s \Gamma(n(u, s) + \eta)}{\prod_s \Gamma(\eta) \sum_s \Gamma(n(u, s) + 2 \cdot \eta)}$$

$$P(V|U', S; \rho) = \frac{\Gamma(\sum_v \rho) \prod_v \Gamma(n(u, v, s = 1) + \rho)}{\prod_v \Gamma(\rho) \sum_v \Gamma(n(u, v, s = 1) + U \cdot \rho)}$$

$$P(Z|U', S, V; \alpha) = \frac{\Gamma(\sum_z \alpha) \prod_z \Gamma(n(u, z, s = 0) + \alpha)}{\prod_z \Gamma(\alpha) \sum_z \Gamma(n(u, z, s = 0) + K \cdot \alpha)}$$

$$P(W'|Z; \gamma) = \frac{\Gamma(\sum_w \gamma) \prod_w \Gamma(n(z, w) + \gamma)}{\prod_w \Gamma(\gamma) \sum_w \Gamma(n(z, w) + W \cdot \gamma)}$$

where Γ denotes the Gamma function[2].

Gibbs sampling. Given the above, the joint probability distribution with conditional independence assumptions is:

$$P(S, V, Z|U', W') \propto P(S|U') \cdot P(V|S, U') \cdot P(Z|V, S, U') \cdot P(W'|Z)$$

The factors on the right-hand side capture (in order): vulnerability of the user being influenced, potential influencers given the user, facet distribution to be used (latent or influenced), and subsequent words to be used according to the facet distribution chosen. We infer all the distributions using Gibbs sampling.

[1]Note that U' refers to the latent variable attached to each review that 'stores' the user information. Thus, a user might appear multiple times in U' since she might have written reviews on multiple items. Similar for W'.

[2]The derivation of the following equations — for integrating out latent variables from the joint distribution exploiting Multinomial-Dirichlet conjugacy and Gibbs Sampling updates — follow from the standard principles of Latent Dirichlet Allocation, and, therefore, details have been omitted for space.

Let the subscript $-j$ denote the value of a variable excluding the data at the j^{th} position. The conditional distributions for Gibbs sampling for updating the latent variable S — that models whether the user is going to write on her own or under influence — is:

$$P(s_j = 0 | u_d, z, s_{-j}) \propto \frac{n(u_d, s_j = 0) + \eta}{\sum_s n(u_d, s) + 2 \cdot \eta} \cdot \frac{n(u_d, z, s_j = 0) + \alpha}{\sum_z n(u_d, z, s_j = 0) + K \cdot \alpha} \quad (2)$$

$$\tilde{P}(s_j = 1 | u_d, v_{d'}, z, s_{-j}) \propto \frac{n(u_d, s_j = 1) + \eta}{\sum_s n(u_d, s) + 2 \cdot \eta} \cdot \frac{n(v_{d'}, z) + \alpha}{\sum_z n(v_{d'}, z) + K \cdot \alpha} \quad (3)$$

$$P(s_j = 1 | u_d, z, s_{-j}) \propto max_{v_{d'} \in D_i : t_{d'} < t_d} \tilde{P}(s_j = 1 | u_d, v_{d'}, z, d, s_{-j}) \quad (4)$$

The first factor in Equation 2 and 3 above models the probability of the user being influenced: as a fraction of how many facets the user wrote under influence ($s = 1$), or otherwise ($s = 0$), out of the total number of facets written. The second factor in Equation 2 models the user's propensity of using a particular facet based on her latent preferences (when $s = 0$); whereas the second factor in Equation 3 models the probability of the user writing about a facet under influence (when $s = 1$) from an earlier review on the given item. Note that in this case — as the user is influenced by another user's review that appeared earlier in her timeline — she adopted her influencer's *used facet distribution* to write about the given facet instead of her own latent facet preference distribution.

Note that in the above question, we did not assume the influencers v'_d to be given since this would lead to a very restrictive Gibbs sampling step. Instead, as shown in Equation 4, we sample the best possible influencer for a given facet to determine the probability for $s = 1$. Accordingly, the influencer for a given user and word, when writing under influence, is updated as

$$v_{d'j} | u_d, s = 1, z, v_{d'-j} = argmax_{v_{d'} \in D_i : t_{d'} < t_d} \Big(\frac{n(u_d, v_{d'}, s = 1) + \rho}{\sum_v n(u_d, v, s = 1) + U \cdot \rho} \cdot \frac{n(v_{d'}, z) + \alpha}{\sum_z n(v_{d'}, z) + K \cdot \alpha} \Big) \quad (5)$$

The first factor above counts how many times u has been influenced by v on writing about any facet — out of the total number of times u has been influenced by any other member in the community. The second factor is the facet distribution *used* by v in the review that influenced u's current facet description.

Instead of computing Equations 4 and 5 separately, we perform the update of both — s_j and $v_{d'j} | s_j = 1$ — *jointly*, thereby, reducing the computation time significantly.

The conditional distribution for sampling the latent facet z is:

$$P(z_j | u_d, s = 0, z_{-j}) \propto \frac{n(u_d, z_j, s = 0) + \alpha}{\sum_z n(u_d, z, s = 0) + K \cdot \alpha} \cdot \frac{n(z_j, w) + \gamma}{\sum_w n(z_j, w) + W \cdot \gamma} \quad (6)$$

$$P(z_j | u_d, v_{d'}, s = 1, z_{-j}) \propto \frac{n(v_{d'}, z_j) + \alpha}{\sum_z n(v_{d'}, z) + K \cdot \alpha} \cdot \frac{n(z_j, w) + \gamma}{\sum_w n(z_j, w) + W \cdot \gamma} \quad (7)$$

The first factor in the above equations models the probability of using the facet z under the user's (own) latent preference distribution (Equation 6), or adopting the influencer's used facet distribution (Equation 7). The second factor counts the number of times facet z is used with word w — out of the total number of times it is used with any other word.

3.1 Overall Processing Scheme

Exploiting the above results, the overall inference is an iterative process consisting of the following steps. We sort all reviews on an item by timestamps. For *each word* in each review on an item:

(1) Estimate whether the word has been written under influence, i.e. compute s using Equations 2 - 4 keeping all facet assignments fixed from earlier iterations.
(2) In case of influence (i.e. $s = 1$), an influencer v is jointly sampled from the previous step.
(3) Sample a facet for the word using Equations 6 and 7 keeping all influencers and influence variables fixed.

The process is repeated until convergence of the Gibbs sampling process (i.e. the log-likelihood of the data stabilizes).

3.2 Example

Consider a set of reviews written by three users in the following time order: first Adam, then Bob, then Sam (see Table 2). The table also shows the *current* assignment of the latent variables z and s. The goal is to *re-sample* the influence variables. For ease of explanation, we ignore the concentration parameters of the Dirichlet distribution in the example and we ignore the subscript $-j$ from the variables. That is, we do not exclude the current state of the own random variable as in Gibbs sampling.

Similar to before, let $n(u, s)$ be the number of tokens written by u with influence variable as s, $n(d, z)$ be the total number of tokens with topic as z in document d, $n(d)$ be the number of tokens in document d, and $n(u)$ be the total number of tokens written by u.

For Adam we have $s = 0$ for each word. As he is the first reviewer, he has no influencers. For Bob, the influence variable s w.r.t. the word 'non-linear' is based on:

$$P(s_{\text{'non-lin'}}=0|u=Bob,\, z=z_2) \propto \frac{n(u=Bob,\, s=0)}{n(u=Bob)} \cdot \frac{n(u=Bob,\, z=z_2,\, s=0)}{n(u=Bob,\, s=0)}$$
$$= \frac{1}{2} \cdot \frac{0}{1} = 0$$
$$P(s_{\text{'non-lin'}}=1|u=Bob,\, v=Adam,\, z=z_2,\, v_d=d_1)$$
$$\propto \frac{n(u=Bob,\, s=1)}{n(u=Bob)} \cdot \frac{n(z=z_2,\, v_d=d_1)}{n(v_d=d_1)} = \frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}$$

Therefore, Bob is more likely to write 'non-linear' being influenced by Adam's review than on his own. Similarly, for Sam:

$$P(s_{\text{'non-lin'}}=0|u=Sam,\, z=z_2) \propto \frac{1}{2} \cdot 1 = \frac{1}{2}$$

Note the higher probability compared to the one of Bob since Sam uses further terms (i.e. 'thriller') which also belong to facet z_2 that he wrote uninfluenced. For the case $s = 1$, we would obtain:

$$P(s_{\text{'non-lin'}}=1|u=Sam,\, v=Adam,\, z=z_2,\, v_d=d_1) \propto \frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}$$
$$P(s_{\text{'non-lin'}}=1|u=Sam,\, v=Bob,\, z=z_2,\, v_d=d_2) \propto \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$$

As seen, Sam is more likely influenced by Adam's review, rather than by Bob's, when considering facet z_2, since d_1 has a higher concentration of z_2. It is worth noting that the probability of the influence variable s depends only on the facet, and not the exact words. Our model captures semantic or facet influence rather than just capturing lexical match. Overall, however, Sam is likely to write 'non-linear' on his own rather than being influenced by someone else since $P(s_{\text{'non-lin'}}=0|\ldots)$ is larger.

Reviewer	Document	time	Word	Facet	Influence (s=)
Adam	d_1	0	action	z_1	0
			non-linear	z_2	0
			narrative	z_2	0
Bob	d_2	1	action	z_1	0
			non-linear	z_2	1
Sam	d_3	2	non-linear	z_2	1
			thriller	z_2	0

Table 2: Example to illustrate our method.

While the above example considers a single item, in a community setting — especially for communities dealing with items of fine taste like movies, food and beer where users co-review multiple items — such statistics are aggregated over several other items. This provides a stronger signal for influence when *a user copies/echoes similar facet descriptions from a particular user across several items.* Our algorithm, therefore, relies on three main factors to model influence and influencer in the community:

a) The vulnerability of a user u in getting influenced, modeled by π and captured in the counts of $n(u, s)$.
b) The textual focus of the influencing review v_d by v on the specific facet (z), modeled by θ and captured in the counts of $n(v_d, z)$; as well as how many times the influencer v influenced u, modeled by ψ and captured in counts of $n(u, v, s = 1)$ — aggregated over all facets and items they co-reviewed.
c) The latent preference of u for z, modeled by θ_u and captured in the counts of $n(u, z, s = 0)$.

3.3 Fast Implementation

In the above generative process, we sample a facet for *each token/word* in a given review. Thus, we may sample different facets for the same word present multiple times in a review. While this makes sense for long documents where a word can belong to multiple topics, for short reviews it is unlikely that the same word is used to represent different facets. Therefore, we reduce the time complexity by sampling a facet for *each unique token* present in a review. We modify our sampling equations to reflect this change.

In the original sampling equations, we let *each token* contribute 1 unit to the counts of the distribution for estimation. Now, *each unique token* contributes c units corresponding to c copies of the token in the review. As we sample a value for a random variable during Gibbs sampling, ignoring its current state, we also need to discount c units for the token (instead of 1) to preserve the overall counts. All the sampling equations are modified accordingly.

3.4 Constructing the Influence Network

Our inference procedure computes values for the latent variables S, V, Z, and corresponding distributions. Using these, our objective is to construct the influence network given by ψ:

$$\psi_{u,v} = \frac{n(u, v, s = 1) + \rho}{\sum_v n(u, v, s = 1) + U \cdot \rho}$$

The above counts the number of facet descriptions $n(u, v, s = 1)$ that are copied by u from v (with $s = 1$ depicting influence) out of the ones copied by u from anyone else. Given ψ, we can construct a directed, weighted influence network $G = (U, E)$, where each user $u \in U$ is a node, and the edgeset E is given by:

$$E = \left\{ \{v, u\} \mid \psi_{u,v} > 0, u \in U, v \in U \right\} \quad (8)$$

That is, there exists an edge from v to u, if v positively influences u with the edge weight being $\psi_{u,v}$.

Furthermore, GhostLink can distinguish between different facet preference distributions of each user. The *observed* facet preference distribution θ_u^{obs} of u is given by:

$$\theta_{u,z}^{obs} = \frac{n(u, z) + \alpha}{\sum_z n(u, z) + K \cdot \alpha}$$

This counts the proportion of times u wrote about facet z out of the total number of times she wrote about any facet — with or without influence. This distribution represents essentially the preferences as it is captured by a standard author-topic model [32], user-facet model [25], and most of the other works using generative processes to model users/authors.

With GhostLink, however, we can derive even more informative distributions:

$$\theta_{u,z}^{latent} = \frac{n(u, z, s = 0) + \alpha}{\sum_z n(u, z, s = 0) + K \cdot \alpha}$$

$$\theta_{u,z}^{infl} = \frac{n(v = u, z, s = 1) + \alpha}{\sum_z n(v = u, z, s = 1) + K \cdot \alpha}$$

The distribution θ_u^{latent} intuitively represents a user's latent facet preference when *not being influenced* from the community (i.e. $s = 0$). In contrast, θ_u^{infl} captures the facet distribution of u as an influencer, i.e. that she used to influence someone else. That is, the latter one counts the proportion of times u was chosen as the influencer (i.e. $v = u$, $s = 1$) by another user in the community; or in other words when some other user copied from u.

4 ITEM RATING PREDICTION USING INFLUENCE NETWORKS

Our proposed method learns an influence network ψ from the review data. We hypothesize that using this network helps to improve rating prediction. That is, our objective is to predict the rating $y'_{u,i,t}$ that user u would assign to an item i at time t exploiting her latent social neighborhood given by ψ_u. Since we know the actual ground ratings $y_{u,i,t}$, the performance for this task can be measured by the mean squared error: MSE = $\frac{1}{|U,I|} \sum_{u,i} (y_{u,i,t} - y'_{u,i,t})^2$. Note that we use the rating data only for the task of rating prediction – it has *not been used* to extract the influence graph.

In the following, we describe the features we will create for each review for the prediction task. We will analyze and compare their effects in our experimental study. Recap that each review d consists of a sequence of words $\{w\}$ by u on item i at time t.

F1. Language model features based on the review text: Using the learned language model β, we construct $\langle F_w = log(max_z \beta_{z,w}) \rangle$ of dimension W (size of the vocabulary). That is, for each word w in the review, we consider the value of β corresponding to the best facet z that can be assigned to the word. We take the log-transformation of β which empirically gives better results.

F2. Rating bias features: Similar to [19, 25], we consider: (i) Global rating bias γ_g: Average rating $avg(\langle y \rangle)$ assigned by all users to all items. (ii) User rating bias γ_u: Average rating $avg(\langle y_{u,.,.} \rangle)$ assigned by u to all items. (iii) Item rating bias γ_i: Average rating $avg(\langle y_{.,i,.} \rangle)$ assigned by all users to item i.

F3. Temporal influence features: Finally, we exploit the temporal and influence information we have learned with our model. (i) Temporal rating bias γ_r: Average rating $avg(\langle y_{.,i,t} \rangle)$ assigned by all users to item i *before* time t. This baseline considers the temporal trend in the rating pattern. (ii) Temporal influence from rating γ_d: Let $\langle d_t \rangle$ be the set of reviews written by users $\langle v_{d_t} \rangle$ before time t on the item i. Consider the influence of $v_{d'}$ on u, i.e. the variable $\psi_{u,v_{d'}}$, as learned by our model. The feature $avg(\langle \psi_{u,v_{d_t}} \cdot y_{v_{d_t},i,t} \rangle)$ aggregates the rating of each previous user and her influence on the current user for item i at time t to model the influence of previous users' ratings on the current user's rating. This baseline combines the temporal trend and the social influence of earlier users' rating. (iii) Temporal influence from context γ_{dc}: Consider the review d with the sequence of words $\langle w \rangle$. Let $s_w \in \{0, 1\}$ be the influence variable sampled for a word w, and v_w be the influencer sampled for the word when $s_w = 1$, as inferred from our model. Also, let y_{v_w} be the rating assigned by v_w to the current item at time $t' < t$. Consider $\mathbb{I}(.)$ be an indicator function that is 1 when its argument is true, and 0 otherwise. We use:

$$\gamma_{dc} = \frac{1}{|d|} \sum_{w \in d} \left(\mathbb{I}(s_w = 1) \cdot \left(\psi_{u,v_w} \cdot y_{v_w} \right) + \mathbb{I}(s_w = 0) \cdot \gamma_u \right)$$

For each word w in the review d, if the word is written under influence ($s_w = 1$), we consider the influencer's rating and her influence on the current user given by $\psi_{u,v_w} \cdot y_{v_w}$. Otherwise ($s_w = 0$), we consider the user's self rating bias γ_u. This is aggregated over all the words in the review. This baseline combines the temporal trend and context-specific social influence of earlier users' rating.

Using different combinations of these features (see Sec. 5), we use Support Vector Regression [3] from LibLinear with default parameters to predict the item ratings $y'_{u,i,t}$, using ten-fold cross-validation (https://www.csie.ntu. edu.tw/ cjlin/liblinear/).

Dataset	#Users	#Items	#Reviews	#Years
Beer (BeerAdvocate)	33,387	66,051	1,586,259	16
Beer (RateBeer)	40,213	110,419	2,924,127	13
Movies (Amazon)	759,899	267,320	7,911,684	16
Food (Amazon)	256,059	74,258	568,454	16
TOTAL	1,089,558	518,048	12,990,524	-

Table 3: Dataset statistics.

5 EXPERIMENTS

We empirically analyze various aspects of GhostLink, using four online communities in different domains: BeerAdvocate (`beeradvocate.com`) and RateBeer (`ratebeer.com`) for *beer* reviews. Amazon (`amazon.com`) for *movie* and *food* reviews. Table 3 gives an overview. All datasets are publicly available at `http://snap.stanford.edu`. We have a total of 13 million reviews from 1 million users over 16 years

	Beer advocate	Rate beer	Amazon Foods	Amazon Movies
GhostLink: Fast Implementation	1.8	1.6	0.08	1.9
GhostLink: Basic	6	2.2	0.14	3.1

Table 4: Run time comparison (in hours) till convergence between different versions of GhostLink.

from all of the four communities combined. From each community, we extract the following quintuple for GhostLink <$userId, itemId, timestamp, rating, review$>. We set the number of latent facets $K = 20$ for all datasets. The symmetric Dirichlet concentration parameters are set as: $\alpha = \frac{1}{K}, \eta = \frac{1}{2}, \rho = \frac{1}{U}, \gamma = 0.01$.[3]

Performance improvements of GhostLink over baseline methods are statistically significant at 99% level of confidence determined by *paired sample t-test.*

5.1 Likelihood, Smoothness, Fast Convergence

There are multiple sets of latent variables in GhostLink that need to be inferred during Gibbs sampling. Therefore, it is imperative to show the resultant model is not only stable, but also improves log-likelihood of the data. A higher likelihood indicates a better model. There are several measures to evaluate the quality of facet models; we use here the one from [35]: $LL = \sum_d \sum_{j=1}^{N_d} log\ P(w_{d,j}|\beta; \alpha)$.

Figure 3 shows the log-likelihood of the data per iteration for the Beeradvocate and Amazon Foods data. The plots for the other datasets are similar. We find that the learning is stable and has a *smooth* increase in the data log-likelihood *per iteration.* Empirically GhostLink also shows a fast convergence in around 10 iterations.

Table 4 shows the run time comparison to convergence between the basic and fast implementation of GhostLink[4]. The fast version uses two tricks: (i) instead of computing Equations 4 and 5 separately, it estimates — s_j and $v_{d',j}|s_j = 1$ — *jointly*, and (ii) it estimates facets for each *unique* token *once* as in Section 3.3.

We also compare the log-likelihood of our model to another generative model that is closest to our work namely, the Author-Topic Model [32]. This work models documents (reviews) to have a distribution over authors, authors to have a distribution over topics, and topics to have a distribution over words. This model is easy to mimic in our setting by ignoring the notion of influence (i.e. setting $s = 0$ as constant for all authors/users). Figure 3 shows the stark difference in log-likelihood of the two models where GhostLink considering influence ($s \in \{0, 1\}$) performs much better than the baseline that ignores the effect of temporal influence ($s \in \{0\}$).

[3]We did not fine-tune hyper-parameter K. It is possible to improve performance by considering the value of K that gives the best model perplexity. Similarly, we consider symmetric Dirichlet priors for a simplistic model with less hyper-parameters to tune.
[4]Experiments are performed in: Intel(R) Xeon(R) CPU E5-2667 v3 @ 3.20GHz. Note that our Gibbs sampling based inference process is sequential and not distributed.



Figure 3: Log-likelihood of GhostLink and Author-Topic Model [32] per-iteration in Beeradvocate & Amazon Foods.

5.2 Influence-aware Item Rating Prediction

Next, we show the effectiveness of GhostLink for item rating prediction. In Section 4 we described the set of features and evaluation measure for this task. Table 5 compares the mean-squared error of GhostLink with all the baselines with *ten-fold cross validation* — where we use 90% of the data for training and 10% for test with results averaged over 10 such splits.

We divide our baselines into four main categories. For each category, we chose the state-of-the-art system as a baseline that is the most representative of that category with all the features as applicable. Unavailability of explicit user-user links in our data renders many of the related works inapplicable to our setting.

(A) Rating and Time-aware Latent Factor Models: These baselines model users, items, ratings and their temporal dynamics but ignore the text or content of the reviews. For most of these baselines, we used the code repository from http://cseweb.ucsd.edu/ jmcauley/code/. Note that these models use the rating bias features (F2) from Section 4. Since they do not model text or influence network, the other features are not applicable.

(a) LFM: This is the classical latent factor model based on collaborative filtering with temporal dynamics [18] that considers ratings, latent facets, and time.

(b) Community at uniform rate: This set of models [26, 39, 40] consider users and products in a community to evolve using a single global clock with the different stages of community evolution appearing at uniform time intervals. So the preference for items evolves over time.

(c) Community at learned rate: This extends (b) by learning the rate at which the community evolves with time [26].

(d) User at uniform rate: This extends (b) to consider individual users and modeling users' progression based on their maturity and preferences evolving over time. The model assumes a uniform rate for evolution [26].

(e) User at learned rate: This extends (d) by allowing each user to evolve on their individual clock, so that the time to attain maturity varies for different users [26].

(B) Text-aware Latent Factor Model: Unlike the previous baselines, this model [25] considers text of the reviews along with the latent factor models using collaborative filtering for item rating prediction. The authors learn topic/facet distributions from text using a generative model based on Latent Dirichlet Allocation, and tie them to the latent facet distributions learned from the collaborative filtering model based on users and ratings. All of these are jointly learned to minimize the mean squared error for item rating prediction. This is the strongest baseline for our work but ignores the notion of network influence. Note that this baseline uses the rating bias features (F2) and language model (F1) from Section 4. The network influence features are not applicable.[5]. Also, note that the generative process in this work is similar to the Author Topic Model [32] with the main difference of the former being tailored for item rating prediction.

[5]We used their code publicly available at http://cseweb.ucsd.edu/ jmcauley/code/

Models	Beer advocate	Rate beer	Amazon Foods	Amazon Movies
(D) GhostLink (Rating + Network + Time + Text)	**0.282**	**0.250**	**0.711**	**0.646**
Rating Bias	0.458	0.376	1.245	1.062
Network Influence	0.443	0.386	1.652	1.434
Rating + Network Influence	0.433	0.347	1.236	1.050
Language Model	1.069	1.148	3.481	4.427
(B) Rating + Text-aware				
Text-based Collab. Filtering [25, 32]	0.373	0.302	1.347	1.233
(C) Rating + Time + Network-aware				
NetInfluence [9]	0.465	0.426	0.93	0.878
(A) Rating + Time-aware				
LFM [18]	0.559	0.917	1.465	1.620
Community at uniform rate [26, 39, 40]	0.582	0.945	1.530	1.727
User at uniform rate [26]	0.586	0.950	1.523	1.729
Community at learned rate [26]	0.532	0.833	1.529	1.729
User at learned rate [26]	0.610	0.797	1.007	0.891

Table 5: Mean squared error for rating prediction (lower is better). GhostLink outperforms competing methods.

(C) Network-aware Models: We also experiment with two information diffusion based baselines [7, 9]. Both models infer the latent influence network underlying a community based on only the temporal traces of activities (e.g., timestamps of users *posting* reviews, nodes *adopting* or becoming infected with information); they *ignore the review text.*

(f) NetInfluence: This model [9] learns the probability of one node influencing another based on logs of their past propagation (*action logs*). The model assumes that when a user u in a network is influenced to perform an action, it may be influenced by its neighbors (Ψ_u in our setting) who have performed the action before. Therefore, each of these predecessors share the "credit" for influencing u to perform that action. In order to adapt their model to our setting, we consider the event of writing a review on an item i to be an *action* at a given timestamp. Therefore input is the set of actions $\langle u, i, t\rangle$ and $\langle u, v\rangle$. Although the authors do not perform recommendation, we use their estimated "influence" scores to construct Ψ (refer to Equation 8).[6]. This allows us to use all the features (F3) in Section 4 derived from the influence network in addition to the rating bias features (F1). Since they do not model text, the language model features are not applicable.

(D) GhostLink: We evaluate GhostLink with various combinations of the feature sets. In particular, we consider: (a) rating bias (F2, F3.i), (b) network influence (F3), (c) combining rating and network influence (F2, F3), (d) language model (F1), and the full model (F1, F2, F3).

Results: Table 5 shows the results. Standard latent factor collaborative filtering models and most of its temporal variations (Models: A) that leverage rating and temporal dynamics but ignore text and network influence perform the worse. We observe that the network diffusion based model that incorporates the latent influence network from temporal traces in addition to the rating information (Models: C) perform much better than the previous models not considering the network information. However, these too ignored the textual signals. Finally, we observe that contextual information harnessed from the review content in addition to rating information (Models: B) outperforms all of the previous models.

From the variations of Ghostlink (using only language model), we observe that textual features alone are not helpful. GhostLink progressively improves as we incorporate more influence-specific features. Finally, the joint model leveraging all of context, rating, temporal and influence features incurs the least error. Comparison with the best performing baseline models shows the power of combined contextual and influence features over only context (Models: B) or only network influence (Models: C).

Additional Network-aware models: We also explored NetInf [7] for tracing paths of diffusion and influence through networks. Given the times when nodes adopt pieces of information or become infected, NetInf identifies the optimal network that best explains the observed infection times. To adapt NetInf to our setting, we consider all the reviews on an item to form a cascade — with the total number of cascades equal to the number of items. For each cascade (item), the input is the set of reviews on the item by users u at timestamps t. However, NetInf yielded extremely sparse networks on our datasets — suffering from multiple modeling assumptions like a single influence point for a node in a cascade, static propagation and fixed transmission rates for all the nodes. For example, in BeerAdvocate it extracted only 5 pairs of influenced interactions. In contrast, both our model as well as NetInfluence, the influence probability $\Psi_{u,v}$ varies for every pair of nodes.

[6]We used their code available at http://www.cs.ubc.ca/ goyal/code-release.php/

Dataset	C1: θ_u^{obs} vs. θ_u^{latent}	C2: θ_u^{latent} vs. θ_u^{infl}	C3: θ_u^{obs} vs. θ_u^{infl}
Beeradvocate	0.318	0.067	0.315
Ratebeer	0.483	0.067	0.328
Amazon Foods	0.368	0.202	0.429
Amazon Movies	0.370	0.110	0.321

Table 6: Facet preference divergence between distributions.

5.3 Facet Preference Divergence

In this study, we want to examine if there is any difference between the latent facet preference of users, as opposed to their *observed* preference, and their preference when acting as an influencer (see Sec. 3.4 for the definition of these preference distributions).

We compute the Jensen-Shannon Divergence (JSD) between the different distributions $\theta_u^{infl}, \theta_u^{latent}, \theta_u^{obs}$ to observe their difference. JSD is a symmetrized form of Kullback-Leibler divergence that is normalized between 0 - 1 with 0 indicating identical distributions. We compute the JSD results averaged over all the users u in the community, i.e. $\frac{1}{|U|}\sum_u JSD(\theta_u^x \,||\, \theta_u^y)$ with the corresponding $x, y \in \{latent, infl, obs\}$. Table 6 shows the results.

We observe (statistically) significant difference between the latent facet preferences of users from that observed/acquired in a community (C1). This result indicates the strong occurrence of social influence on user preferences in online communities. We also find that users are more likely to use their original latent preferences to influence others in the community, rather than their acquired

Dataset	Model	Pearson Correlation
Beeradvocate	**GhostLink**	**0.708**
	NetInfluence	0.616
	Temporal co-reviewing	0.400
Ratebeer	**GhostLink**	**0.736**
	NetInfluence	0.653
	Temporal co-reviewing	0.615

Table 7: Pearson correlation (higher is better) between different models to find influential users in the community.

Dataset	IG		MWSF			
	Edges	Weight	Edges	% of IG	Weight	% of IG
Beeradvocate	180.5K	31.8K	132.5K	73.40%	31.6K	99.37%
Ratebeer	152.8K	24.7K	95.4K	62.43%	24.5K	99.19 %
Amazon Foods	107.5K	59.51K	104.1K	96.84 %	59.47K	99.93%
Amazon Movies	589K	145.4K	476K	80.81%	145K	99.72%

Table 8: Structure of the latent influence networks: The Influence Graph (IG) is well represented by a Maximum Weighted Spanning Forest (MWSF)

ones. That is, the JSD between the influencer and latent facet preference distribution (C2) is always significantly smaller than the JSD between the observed and the influencer distribution (C3).

5.4 Finding Influential Members

GhostLink generates a directed, weighted influence network $G = (U, E)$ using the user-influencer distribution Ψ. Given such a network, we can find influential nodes in the network. We used several algorithms to measure authority like Pagerank, HITS, degree centrality etc. out of which eigenvector centrality performed the best[7]. The basic idea behind eigenvector centrality is that a node is considered influential, not just if it connects to many nodes (as in simple degree centrality) but if it connects to high-scoring nodes in the network. Given the eigenvector centrality score x_v for each node v, we can compute a ranked list of users.

Comparison: An obvious question is whether this ranking based on the influence graph Ψ is really helpful? Or put differently: Does this perform better compared to a simpler graph-based influence measure? A natural choice, for example, would be the temporal co-reviewing behavior of users. To construct such a graph, we can connect two users u and v with a directed edge if u writes a review following v. The weight of this edge corresponds to all such reviews (following the above temporal order) aggregated across all the items. Therefore, v acts as an influencer if u closely follows his reviews. We choose a cut-off threshold of at least 5 reviews. Also for this graph, we can compute eigenvector centrality scores, and obtain a ranked list of users as described above.

The task: We want to find which of the above graphs gives a better ranking of users. We perform this experiment in the Beeradvocate and Ratebeer communities. In these communities, users are awarded points based on factors like: their community engagement, how other users find their reviews helpful and rate them, as well as their expertise on beers. This is moderated by the community administrators. For instance, in Beeradvocate users are awarded Karma points[8]. The exact algorithm for calculation of these points is not made public to users as it can be game to manipulation – and of course, these scores are also not used in GhostLink.

We used these points as a proxy for user authority, and rank the users. This ranked list is used as a reference list (ground-truth) for comparison. That is, we use the ranked list of users based on eigenvector centrality scores from our influence graph, and compute Pearson correlation with the reference list[9]. A correlation score of 1 indicates complete agreement, whereas −1 indicates complete disagreement. We can also do the same for the ranked list of users based on their co-reviewing behavior. As another strong baseline, we also consider the influence scores for the users as generated by NetInfluence [9]. Table 7 shows the results.

We observe that the ranking computed with our influence graph performs much better (higher correlation with ground-truth) than the temporal co-reviewing baseline. Thus, the learned influence network indeed captures more information than simple co-reviewing behavior and even the more advanced diffusion based NetInfluence model, and enables us to find influential users better. Note again, that the point-based scores used for the ground-truth ranking have *not been used* in GhostLink.

5.5 Structure of the Influence Network

Last, we analyze the structure of the influence network Ψ. Our first research question is: How is the mass (sum of influence/edge weights) distributed in the network? Is it randomly spread out, or do we observe any particular structure (e.g., resembling a tree-like structure). For this, we computed a Maximum Weighted Spanning Tree from the graph (or spanning forest as the graph is not connected) and computed the sum of its edge-weights, i.e. its mass.

Table 8 shows the statistics of the constructed MWSF over different datasets and compares it with the (original) influence graph (IG). We observe that *the majority of mass of the influence graph is concentrated in giant tree-components*. For example, in Beeradvocate, 99.37% of the mass of the influence graph is concentrated in the MWSF. The forest accounts for 73.40% of the edges in the influence graph. Thus, the remaining 26.6% of the edges contribute only marginally, and can be pruned out. This tree-like influence matches intuition: a user often influences many other users, while she herself gets primarily influenced by a few – surprisingly, in the majority of cases only by a single other user – as indicated by the good approximation of the graph via a tree (preservation of mass). Figure 5 shows the MWSF for a representative facet "yuengling", and its giant component.

The tree structure in Figure 5 shows another characteristics: only a few users seem to influence many others (it resembles a snowflake) in the community. This brings us to our second research question: Do we observe — similar to real-world networks — specific power-law behaviors? For example, are the majority of nodes 'influencees', and only a few nodes are 'influencers'?

Figure 4 analyzes this aspect. Here we illustrate the distribution of nodes with weighted degree, hub & authority, and eigen vector centrality scores for our influence graph plotted in *log-scale*. These statistics are for the Beeradvocate community. The statistics for

[7]Note that these baselines already subsume simpler activity based ranking (e.g., based on number of reviews written).
[8]https://www.beeradvocate.com/community/threads/beer-karma-explained.184895/
[9]Other ranking measures (Kendall-Tau, Spearman Rho) yield similar improvements.

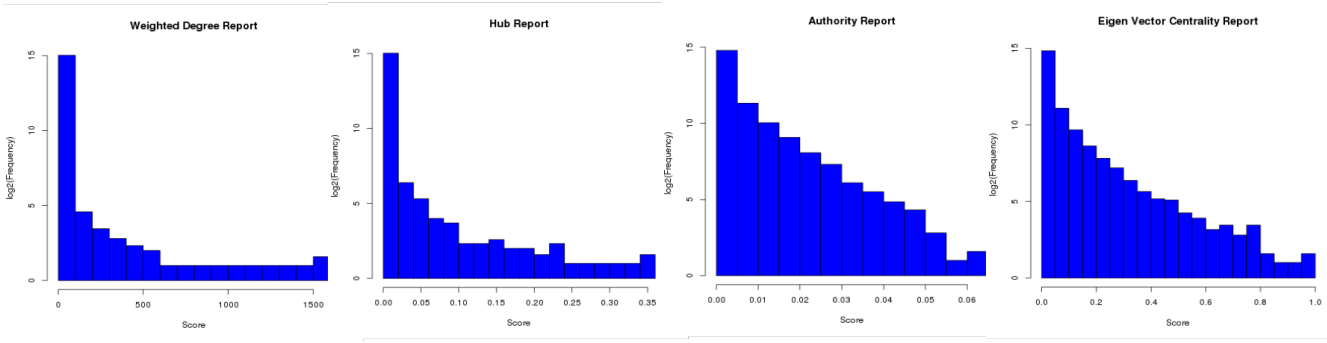


Figure 4: Distribution of nodes with scores (weighted degree, hubs, authorities, and eigen vector centralities in order from left to right) in log-scale for the extracted influence graph (note the change in scale of scores for each figure) in Beeradvocate data.

other communities are similar. Indeed, we observe power-law like distributions with many influencees and a few influencers.

For the HITS algorithm, a hub — with a lot of outgoing edges — is a user who influences a lot of other users; whereas an authority — with a lot of incoming edges — is the one getting influenced by other influential users. Note that each node can be a hub and an authority with different scores simultaneously. We observe that there are a lot of hubs (influencers) with very low influence scores, and only few with very high influence. From the authority report, we see that there are less number of incoming edges to nodes (note the really small range of authority scores of nodes). This indicates that users generally get influenced by only a few users in the community — confirming the tree-like structure of the influence graph.

6 RELATED WORK

State-of-the-art recommender systems exploit user-user and item-item similarities using latent factor models [17, 19]. Temporal patterns in ratings such as bursts, bias, and anomalies are studied in [12, 18, 38]. Recent works [25, 28, 37] have further considered review texts for content-aware recommender systems. However, all of these works assume that users participate independently in the community which is rarely the case.

Social-aware recommender systems [5, 14, 16, 20, 22, 27, 33, 34, 41, 42] exploit peers and friends of users to extract more insights from their activities, likes, and content sharing patterns using homophily. In absence of explicit social networks in many communities, some works [13, 21, 23, 24] exploit collaborative filtering to extract implicit social relationships based on the historical rating behavior. Some of these works also leverage signals like pre-defined trust metrics, and partial or explicit social links. [21, 43] use time as an additional dimension along with ratings.

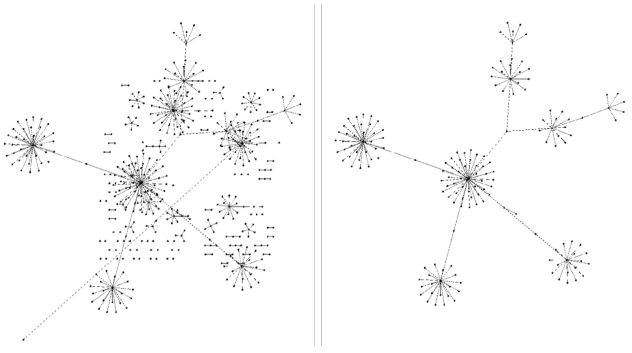


Figure 5: Maximum Weighted Spanning Forest corresponding to a representative facet (left); and its giant component (right).

Information diffusion based works [6, 7, 9, 30] that model underlying *latent* influence or diffusion networks do not consider text. Some of them have strong assumptions in terms of known transmission rates, static and homogeneous transmission etc. Recent works on text-based diffusion [4, 15, 36] alleviate some of these assumptions. However, they also make some assumptions regarding topics of diffusion being known, network being explicit etc. Most importantly, none of these works are geared for item recommendation and do not study the characteristics of review communities.

Works in modeling influence in heterogeneous networks [22] and citations networks [2] assume the presence of explicit user-user links. Prior works on modeling influence propagation and cascades [31] also consider a given network to propagate influence scores. Learning a latent influence network has been possible in the field of information propagation when observing cascades of events [8, 43]. However, these works have not considered the setting where only review text is available, and no explicit networks.

In contrast to prior works, GhostLink learns the latent influence network solely from timestamped user reviews, without requiring any explicit user-user link/rating information. It uses this network to improve item rating prediction considering implicit social influence.

7 CONCLUSION

We presented GhostLink, an unsupervised generative model to extract the underlying influence graph in online communities dealing with items of fine taste like movies, food and beer without requiring any explicit user-user links or ratings. Given only timestamped reviews of users, we leverage opinion conformity from overlapping facet descriptions in co-reviewed content and their temporal traces to extract this graph. Furthermore, we use this influence network to improve item rating prediction by 23% over state-of-the-art methods by capturing implicit social influence. We show in large-scale experiments in four real-life communities with 13 million reviews that GhostLink outperforms several state-of-the-art baselines for tasks like recommendation and identifying influential users.

Acknowledgements. This research was supported by the German Research Foundation, Emmy Noether grant GU 1409/2-1.

We would like to sincerely thank Christos Faloutsos for his insightful and constructive comments on the paper.

REFERENCES

[1] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *JMLR* 3 (2003).

[2] Laura Dietz, Steffen Bickel, and Tobias Scheffer. 2007. Unsupervised prediction of citation influences. In *ICML*. 233–240.

[3] Harris Drucker, Chris J. C. Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. 1997. Support Vector Regression Machines *(NIPS)*.

[4] Nan Du, Le Song, Hyenkyun Woo, and Hongyuan Zha. 2013. Uncover Topic-Sensitive Information Diffusion Networks. In *AISTATS*. 229–237.

[5] Crícia Z. Felício, Klérisson V. R. Paixão, Guilherme Alves, Sandra de Amo, and Philippe Preux. 2016. Exploiting Social Information in Pairwise Preference Recommender System. *JIDM* 7, 2 (2016), 99–115.

[6] Manuel Gomez-Rodriguez, David Balduzzi, and Bernhard SchÃűlkopf. 2011. Uncovering the Temporal Dynamics of Diffusion Networks.. In *ICML*. 561–568.

[7] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. 2010. Inferring Networks of Diffusion and Influence *(KDD '10)*. 1019–1028.

[8] Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. 2012. Inferring Networks of Diffusion and Influence. *TKDD* 5, 4 (2012), 21:1–21:37.

[9] Amit Goyal, Francesco Bonchi, and Laks V.S. Lakshmanan. 2010. Learning Influence Probabilities in Social Networks *(WSDM '10)*. 241–250.

[10] Tom Griffiths. 2002. *Gibbs sampling in the generative model of Latent Dirichlet Allocation.* Technical Report.

[11] Nikou Günnemann, Stephan Günnemann, and Christos Faloutsos. 2014. Robust multivariate autoregression for anomaly detection in dynamic product ratings. In *WWW*. 361–372.

[12] Stephan Günnemann, Nikou Günnemann, and Christos Faloutsos. 2014. Detecting Anomalies in Dynamic Rating Data: A Robust Probabilistic Model for Rating Evolution. In *KDD*.

[13] Guibing Guo, Jie Zhang, Daniel Thalmann, Anirban Basu, and Neil Yorke-Smith. 2014. From Ratings to Trust: An Empirical Study of Implicit Trust in Recommender Systems. In *SAC*. 248–253.

[14] J. Guo, Y. Zhu, A. Li, Q. Wang, and W. Han. 2017. A Social Influence Approach for Group User Modeling in Group Recommendation Systems. *IEEE Intelligent Systems* PP, 99 (2017), 1–1.

[15] Xinran He, Theodoros Rekatsinas, James Foulds, Lise Getoor, and Yan Liu. 2015. HawkesTopic: A Joint Model for Network Inference and Topic Modeling from Text-based Cascades *(ICML'15)*. 871–880.

[16] Junming Huang, Xueqi Cheng, Jiafeng Guo, Huawei Shen, and Kun Yang. 2010. Social Recommendation with Interpersonal Influence. In *ECAI*. 601–606.

[17] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model *(KDD)*.

[18] Yehuda Koren. 2010. Collaborative Filtering with Temporal Dynamics. *Commun. ACM* 53, 4 (2010).

[19] Yehuda Koren and Robert Bell. 2011. Advances in collaborative filtering. In *Recommender systems handbook*.

[20] Sanjay Krishnan, Jay Patel, Michael J. Franklin, and Ken Goldberg. 2010. Social Influence Bias in Recommender Systems: A Methodology for Learning, Analyzing, and Mitigating Bias in Ratings. In *RecSys*.

[21] Chen Lin, Runquan Xie, Xinjun Guan, Lei Li, and Tao Li. 2014. Personalized News Recommendation via Implicit Social Experts. *Inf. Sci.* (2014), 1–18.

[22] Lu Liu, Jie Tang, Jiawei Han, and Shiqiang Yang. 2012. Learning influence from heterogeneous social networks. *DMKD* 25, 3 (2012), 511–544.

[23] Hao Ma. 2013. An Experimental Study on Implicit Social Recommendation. In *SIGIR*. 73–82.

[24] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender Systems with Social Regularization. In *WSDM*. 287–296.

[25] Julian McAuley and Jure Leskovec. 2013. Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text *(RecSys)*.

[26] Julian John McAuley and Jure Leskovec. 2013. From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise Through Online Reviews. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13)*. 897–908.

[27] Jian-Ping Mei, Han Yu, Zhiqi Shen, and Chunyan Miao. 2017. A social influence based trust model for recommender systems. *Intell. Data Anal.* 21, 2 (2017), 263–277.

[28] Subhabrata Mukherjee, Gaurab Basu, and Sachindra Joshi. 2014. Joint Author Sentiment Topic Model *(SDM)*.

[29] Subhabrata Mukherjee, Stephan Günnemann, and Gerhard Weikum. 2016. Continuous Experience-aware Language Model. In *SIGKDD*. 1075–1084.

[30] Seth A. Myers and Jure Leskovec. 2010. On the Convexity of Latent Social Network Inference. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 2 (NIPS'10)*. 1741–1749.

[31] Seth A. Myers, Chenguang Zhu, and Jure Leskovec. 2012. Information Diffusion and External Influence in Networks. In *SIGKDD*. 33–41.

[32] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. 2004. The Author-topic Model for Authors and Documents *(UAI)*.

[33] Jiliang Tang, Huiji Gao, and Huan Liu. 2012. mTrust: Discerning Multi-faceted Trust in a Connected World. In *WSDM*. 93–102.

[34] Jiliang Tang, Xia Hu, Huiji Gao, and Huan Liu. 2013. Exploiting Local and Global Social Context for Recommendation. In *IJCAI*.

[35] Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. 2009. Evaluation Methods for Topic Models *(ICML)*.

[36] Senzhang Wang, Xia Hu, Philip S. Yu, and Zhoujun Li. 2014. MMRate: Inferring Multi-aspect Diffusion Networks with Multi-pattern Cascades *(KDD '14)*. 1246–1255.

[37] Hongning Wang et al. 2011. Latent aspect rating analysis without aspect keyword supervision *(KDD)*.

[38] Liang Xiang and et al. 2010. Temporal Recommendation on Graphs via Long- and Short-term Preference Fusion *(KDD)*.

[39] Liang Xiang, Quan Yuan, Shiwan Zhao, Li Chen, Xiatian Zhang, Qing Yang, and Jimeng Sun. 2010. Temporal Recommendation on Graphs via Long- and Short-term Preference Fusion *(KDD '10)*. 723–732.

[40] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G Carbonell. 2010. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*. 211–222.

[41] Mao Ye, Xingjie Liu, and Wang-Chien Lee. 2012. Exploring Social Influence for Recommendation: A Generative Model Approach. In *SIGIR*. 671–680.

[42] Chuxu Zhang, Lu Yu, Yan Wang, Chirag Shah, and Xiangliang Zhang. 2017. Collaborative User Network Embedding for Social Recommender Systems. In *SDM*. 381–389.

[43] Qin Zhang, Jia Wu, Peng Zhang, Guodong Long, Ivor W. Tsang, and Chengqi Zhang. 2016. Inferring Latent Network from Cascade Data for Dynamic Social Recommendation. In *ICDM*. 669–678.

Estimating Walk-Based Similarities Using Random Walk

Shogo Murai*
The University of Tokyo
Bunkyo-ku, Tokyo, Japan
s.murai@is.s.u-tokyo.ac.jp

Yuichi Yoshida†
National Institute of Informatics
Chiyoda-ku, Tokyo, Japan
yyoshida@nii.ac.jp

ABSTRACT

Measuring similarities between vertices is an important task in network analysis, which has numerous applications. One major approach to define a similarity between vertices is by accumulating weights of walks between them that encompasses personalized PageRank (PPR) and Katz similarity. Although many effective methods for PPR based on efficient simulation of random walks have been proposed, these techniques cannot be applied to other walk-based similarity notions because the random walk interpretation is only valid for PPR.

In this paper, we propose a *random-walk reduction* method that reduces the computation of any walk-based similarity to the computation of a stationary distribution of a random walk. With this reduction, we can exploit techniques for PPR to compute other walk-based similarities. As a concrete application, we design an indexing method for walk-based similarities with which we can quickly estimate the similarity value of queried pairs of vertices, and theoretically analyze its approximation error. Our experimental results demonstrate that the instantiation of our method for Katz similarity is two orders of magnitude faster than existing methods on large real networks, without any deterioration in solution quality.

CCS CONCEPTS

• **Information systems** → **Web mining**; • **Mathematics of computing** → *Graph algorithms.*

KEYWORDS

Walk-based similarity, Katz similarity, random walk

ACM Reference Format:
Shogo Murai and Yuichi Yoshida. 2019. Estimating Walk-Based Similarities Using Random Walk. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313421

*Supported by JST ERATO Grant Number JPMJER1305
†Supported by JSPS KAKENHI Grant Number JP17H04676.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313421

1 INTRODUCTION

In network analysis, measuring similarities between vertices is an important task that has numerous applications such as recommendation [12, 15] and link prediction [20, 25].

One major class of similarity measures is *walk-based similarities* [14]. Here, we first define the weight of a walk. Then, we define the similarity value between two vertices as the sum of the weights of all the walks connecting them. By changing the definition of the weight of a walk, we can express various similarity notions, including personalized PageRank (PPR) [22], Katz similarity [13], and adsorption [3]. Among those, PPR has drawn much attention and several efficient methods have been proposed [1, 2, 4, 10, 17, 18, 24]. In contrast, there are only a few efficient methods that can be applied to *general* walk-based similarities [14, 30], even though they include other practically important similarity notions.

In this paper, we propose a *random-walk reduction* method to address the issue above. Given any walk-based similarity notion, this method constructs a weighted graph on which we can perform a random walk. Then, the original similarity values can be easily recovered from the stationary distribution of a random walk on the weighted graph. With this reduction, we can apply any algorithm that was originally designed to compute PPR (such as the Monte Carlo method [10], FAST-PPR [18] and BiPPR [17]) to any other walk-based similarity notion.

As an application of our random-walk reduction, for any walk-based similarity, we transform BiPPR [17], originally designed to compute the PPR of a target vertex from a source vertex, to a method for the similarity. We also provide a theoretical analysis on its approximation error. The experiment shows that the transformed method instantiated by Katz similarity is two orders of magnitude faster than existing methods without deteriorating the solution quality.

This paper is structured as follows. Section 2 gives an overview of related works. Section 3 provides a basic explanation of our problem. Section 4 explains previous methods that we rely on and outlines our method. We present our method in detail in Sections 5 and 6. We theoretically guarantee the approximation error of our method in Section 7. In Section 8, we present experimental results on one-to-one Katz similarity estimation and show that our method is two orders of magnitude faster than existing methods on large real networks.

2 RELATED WORK

Considering the importance of vertex similarities, several walk-based similarity metrics have been proposed, including PPR [22], Katz similarity [13], and adsorption [3].

Although they are all computable by solving certain linear equations, this approach is time-consuming. To address this issue, various other methods have been proposed to approximately compute the values of these metrics and to compute *top-k* vertices from a given vertex with regard to the given metric.

PPR has been intensively studied in the literature. Monte Carlo approximation algorithms were proposed in [10]. Andersen *et al.* [1, 2] proposed algorithms that iteratively update the estimated value by checking the local neighborhood of each vertex. Lofgren *et al.* [18] proposed bidirectional algorithms for PPR. Their algorithms combine Monte Carlo simulation (*forward work*) and backward local push algorithm (*backward work*) to obtain better estimations in shorter time. For any *threshold* $\delta > 0$, their algorithm for one-to-one similarity estimation can estimate PPR values larger than δ with a theoretical guarantee in average $O(\sqrt{\bar{d}/\delta})$ time, where $\bar{d}$ is the average degree of the graph. These algorithms were later simplified and improved in [17]. Their works are also extended to estimation of probabilities in Markov models [4] and single element of linear systems of form $A^\ell \boldsymbol{z}$ [24].

Esfandiar *et al.* [9] proposed algorithms for approximating Katz similarities and for computing the top-k vertices from the given vertex with theoretical guarantees.

However, there are few methods that can handle *general* walk-based similarity metrics. The only works we are aware of are [14] and [30]. Both methods considered computing top-k vertices by locally updating estimated values and maintaining their upper and lower bounds, which cannot efficiently handle one-to-one similarity estimation.

3 PRELIMINARIES

For a positive integer $n \in \mathbb{N}$, let $[n]$ denote the set $\{1, 2, \ldots, n\}$. We denote by $\langle \cdot, \cdot \rangle$ the standard inner product.

Let $G = (V, E)$ be a directed graph with n vertices and m edges. For a vertex $v \in V$, let $N^+(v)$ denote the set of out-neighbors of $v \in V$ and $d^+(v) := |N^+(v)|$ denote the out-degree of v. Similarly, we define $N^-(v)$ and $d^-(v)$ for in-neighbors.. For a vertex $v \in V$, let $\boldsymbol{e}_v \in \mathbb{R}^V$ be the indicator vector of v, that is, the all-zero vector except the v-th element being one.

3.1 Walk-based similarity

Let $G = (V, E)$ be a directed graph. A *walk* on G is a sequence $(v_0, v_1, \ldots, v_\ell)$ of vertices such that $v_{i-1}v_i \in E$ for every $i \in [\ell]$. We note that $v_0, v_1, \ldots, v_\ell$ are not necessarily distinct; when they are distinct, it is called a *path*.

A *walk-based similarity* [14] is parameterized by a weight function $w \colon E \to \mathbb{R}_+$ on edges and $c \in \mathbb{R}_+$. For a walk $W = (v_0, v_1 \ldots, v_\ell)$ of length ℓ, we define its weight $w_c(W)$ as

$$w_c(W) = c \prod_{i=1}^{\ell} w(v_{i-1}v_i)$$

For two vertices $s, t \in V$, let walk(s, t) be the (infinite) set of walks from s to t. When $s = t$, we include the path of length 0, which consists only of s, to walk(s, s).

The similarity $\sigma_{c,w}(s, t)$ from a vertex $s \in V$ to another vertex $t \in V$ is defined as

$$\sigma_{c,w}(s, t) = \sum_{W \in \mathrm{walk}(s,t)} w_c(W),$$

when the sum converges. We omit the subscripts c and w when they are clear from the context.

Although it is difficult to compute $\sigma_{c,w}(s, t)$ based on the definition, we can rephrase it in linear algebraic terms [14]. We define the *transition weight matrix* $H \in \mathbb{R}^{V \times V}$ as $H_{uv} = w(uv)$. If $\sigma_{c,w}(u, v)$ is well-defined for every $(u, v) \in V^2$, then the spectral radius $\rho(H)$ of H must be less than one and $\sigma_{c,w}(s, t)$ can be represented as

$$\sigma_{c,w}(s, t) = \left\langle \boldsymbol{e}_s, c \sum_{\ell=0}^{\infty} H^\ell \boldsymbol{e}_t \right\rangle,$$

where $\sum_{\ell=0}^{\infty} H^\ell = (I - H)^{-1}$ [8]. Because every component of H is nonnegative, it is guaranteed that $\rho(H)$ equals the largest real eigenvalue of H by the Perron-Frobenius theorem.

By changing the weight function $w \colon E \to \mathbb{R}_+$ and the scalar coefficient $c \in \mathbb{R}_+$, we can express various similarity notions.

Example 3.1 (Personalized Pagerank [22]). *Let $T = AD^{-1}$ be the transition matrix, where $A \in \mathbb{R}^{V \times V}$ is the adjacency matrix of G and $D \in \mathbb{R}^{V \times V}$ is the diagonal matrix such that $D_{vv} = d^+(v)$ for every $v \in V$. For $\alpha \in (0, 1)$, the personalized PageRank $\sigma_{\mathrm{PPR}}(s, t)$ from a vertex $s \in V$ to another vertex $t \in V$ is*

$$\sigma_{\mathrm{PPR}}(s, t) = \left\langle \boldsymbol{e}_s, \alpha \sum_{\ell=0}^{\infty} (1 - \alpha)^\ell T^\ell \boldsymbol{e}_t \right\rangle,$$

which equals the probability of staying at t in the stationary distribution of the following random walk process: Starting with s, we move to a random neighbor of the current vertex with probability $1 - \alpha$ and jump back to s with the complement probability α. We can easily check that σ_{PPR} is the walk-based similarity with $c = \alpha$ and $w(uv) = 1/d^+(u)$.

Example 3.2 (Katz Similarity [13]). *For $\beta \in (0, 1)$, the Katz similarity from a source vertex $s \in V$ to another vertex $t \in V$ is*

$$\sigma_{\mathrm{Katz}}(s, t) = \left\langle \boldsymbol{e}_s, \sum_{\ell=0}^{\infty} \beta^\ell A^\ell \boldsymbol{e}_t \right\rangle,$$

where $A \in \mathbb{R}^{V \times V}$ is the adjacency matrix. The sum converges when β is smaller than the reciprocal of the largest eigenvalue of A. Here, too, we can confirm that σ_{Katz} is the walk-based similarity with $c = 1$ and $w(uv) = \beta$.

We note that any walk-based similarity can be seen as a Katz similarity on a weighted graph, that is, $A \in \mathbb{R}^{V \times V}$ is a general non-negative matrix.

In this work, we primarily consider the following *one-to-one* similarity estimation problem:

Definition 3.3 (One-to-one similarity estimation problem). *The one-to-one similarity estimation problem is specified by a graph $G = (V, E)$ with a weight function $w \colon E \to \mathbb{R}_+$ on edges and a scalar coefficient $c \in \mathbb{R}_+$. The goal is to construct an index with which, given a pair of vertices $(s, t) \in V^2$, a threshold $\delta \in (0, 1)$, an accuracy parameter $\epsilon \in (0, 1)$, and a failure probability $p \in (0, 1)$ as a query,*

we can compute $\hat{\sigma}(s,t)$ such that $|\hat{\sigma}(s,t) - \sigma_{c,w}(s,t)| < \epsilon \cdot \sigma_{c,w}(s,t)$ with probability at least $1-p$ if $\sigma_{c,w}(s,t) \geq \delta$. If $\hat{\sigma}(s,t)$ satisfies this, we call $\hat{\sigma}(s,t)$ an (ϵ, δ, p)-estimation of $\sigma_{c,w}(s,t)$.

Because approximating small $\sigma_{c,w}(s,t)$ is quite difficult [18], introducing the threshold parameter δ is crucial.

4 PREVIOUS METHODS AND OVERVIEW OF OUR METHOD

4.1 Previous methods

In this section, we review previous methods used for the one-to-one similarity estimation problem.

Iterative method. Iterative method is a basic approach for solving linear systems of the form $A\boldsymbol{x} = \boldsymbol{b}$, where $A \in \mathbb{R}^{n \times n}$ is a matrix with $\rho(I - A) < 1$ and $\boldsymbol{b} \in \mathbb{R}^n$ is a vector. The Jacobi method is a popular method that iteratively computes

$$\boldsymbol{x}_0 = 0, \quad \text{and} \quad \boldsymbol{x}_i = (I - A)\boldsymbol{x}_{i-1} + \boldsymbol{b} \ (i \geq 1).$$

Then, $\boldsymbol{x}_i$ converges to $\boldsymbol{x} = A^{-1}\boldsymbol{b}$ (Note that A is nonsingular from the assumption $\rho(I-A) < 1$). This method can be directly applied to compute walk-based similarity vector from a vertex $s \in V$ by setting $A = I - H$ (from the convergence condition, $\rho(I - A) = \rho(H) < 1$ is guaranteed) and $\boldsymbol{b} = \boldsymbol{e}_t$.

The time complexity of this algorithm is $O(m \cdot i)$, where i is the number of iterations. Moreover, the error decreases as $O(\rho(I - A)^i)$ [21]. This algorithm can compute similarity values from all the vertices to a target vertex simultaneously. (Note that we can also compute similarity values from a source vertex to all the vertices at once by setting $A = I - H^\top, \boldsymbol{b} = \boldsymbol{e}_s$.) However, this method can be rather inefficient to solve the one-to-one similarity estimation problem because the target vertex (or the source vertex) varies with queries.

Forward and backward push methods. The forward push method was proposed in [2] to compute PPR and then was applied to compute Katz similarities [9]. The forward push method approximates the similarity values from a source vertex $s \in V$ by maintaining two vectors; the *reserve vector* $\boldsymbol{p}_s \in \mathbb{R}^V$ and the *residue vector* $\boldsymbol{r}_s \in \mathbb{R}^V$. Initially, we set $\boldsymbol{p}_s = 0$ and $\boldsymbol{r}_s(v) = \boldsymbol{e}_s$. Then, these vectors are updated so that the following invariant is maintained:

$$\sigma_{c,w}(s,v) = p_s(v) + \sum_{u \in V} r_s(u)\sigma_{c,w}(u,v). \tag{1}$$

A pseudocode of this method is given in Algorithm 1.

To the extent of our knowledge, there is no approximation guarantee for the forward push method. As for time complexity, Andersen *et al.* [2] showed that Algorithm 1 runs in $O\left(\frac{1}{\alpha\delta}\right)$ time for PPR if we use the termination condition $\max_{v \in V} \frac{r_s(v)}{d^+(v)} < \delta$ instead of $r_s(u) > \delta$.

The backward push method [1] is similar to the forward push method, except that it maintains a *reverse similarity vector* $\boldsymbol{q}_t \in \mathbb{R}^V$ for which $q_t(v)$ is supposed to approximate $\sigma_{c,w}(v,t)$. The backward push method also maintains a residue vector $\boldsymbol{r}_t \in \mathbb{R}^V$. The invariant for this method is the following:

$$\sigma_{c,w}(v,t) = q_t(v) + \sum_{u \in V} \sigma_{c,w}(v,u) r_t(u). \tag{2}$$

A pseudocode of this method is given in Algorithm 2.

Algorithm 1 Forward push

Input: source vertex $s \in V$, threshold $\delta > 0$

$p_s(u) \leftarrow 0 \ (u \in V), r_s(s) \leftarrow 1, r_s(u) \leftarrow 0 \ (u \in V \setminus \{s\})$
while $\exists u \in V, r_s(u) > \delta$ **do**
  for $v \in N^+(u)$ **do**
    $r_s(v) \leftarrow r_s(v) + w(uv) r_s(u)$.
  end for
  $p_s(u) \leftarrow p_s(u) + r_s(u), r_s(u) \leftarrow 0$.
end while

Algorithm 2 Backward push

Input: target vertex $t \in V$, threshold $\delta > 0$

$q_t(u) \leftarrow 0 \ (u \in V), r_t(t) \leftarrow 1, r_t(u) \leftarrow 0 \ (u \in V \setminus \{t\})$.
while $\exists v, r_t(v) > \delta$ **do**
  for $u \in N^-(v)$ **do**
    $r_t(u) \leftarrow r_t(v) + w(uv) r_t(v)$.
  end for
  $q_t(v) \leftarrow q_t(v) + r_t(v), r_t(v) \leftarrow 0$.
end while

As opposed to the forward push method, the backward push method on PPR has an approximation guarantee. Andersen *et al.* [1] showed that Algorithm 2 returns a reverse similarity vector $\boldsymbol{q}_t$ such that $|q_t(v) - \sigma_{\mathrm{PPR}}(v,t)| \leq \delta$ for every $v \in V$. Lofgren and Goel [19] showed that, for uniformly chosen target vertex $t \in V$, Algorithm 2 runs in expected $O\left(\frac{1}{\alpha\delta} \cdot \frac{|E|}{|V|}\right)$ time.

Monte Carlo method. Since the PPR of a vertex indicate the probability of staying at the vertex in the stationary distribution of a random walk, it can be approximated with a Monte Carlo method [10]. It should be noted that this approach is directly applicable only to PPR, because the transition weight matrix H is not a stochastic matrix for other similarity notions (even $\|H\|_\infty := \max_{\boldsymbol{v}} \|H\boldsymbol{v}\|_\infty / \|\boldsymbol{v}\|_\infty > 1$ may hold). An extension of the Monte Carlo method to general similarity notions and its shortcomings are discussed later.

The Monte Carlo method for PPR first generates N random walks starting from the source vertex $s \in V$ and then, for each $v \in V$, computes the number of random walks N_v that terminate at v. Then, the similarity $\sigma_{\mathrm{PPR}}(s,v)$ is estimated as $\frac{N_v}{N}$. Since the probability that a random walk terminates at $v \in V$ equals $\sigma_{\mathrm{PPR}}(s,v)$, this estimator is unbiased.

Bidirectional methods. The first bidirectional algorithm for computing the PPR between two vertices, called *FAST-PPR*, was proposed in [18] and then its improved version, called *BiPPR*, was proposed in [17]. We describe the latter in detail since our method is built on it.

BiPPR carefully combines the Monte Carlo method and the backward push method. Suppose that we have performed backward pushes to obtain the reverse similarity vector $\boldsymbol{q}_t \in \mathbb{R}^V$ and the residue vector $\boldsymbol{r}_t \in \mathbb{R}^V$ for the target vertex t. From (2), if we have

estimates $\hat{\sigma}_{c,w}(s,v)$ of $\sigma_{c,w}(s,v)$ for each $v \in V$, we can approximate $\sigma_{c,w}(s,t)$ by

$$\hat{\sigma}'_{c,w}(s,t) := q_t(s) + \sum_{v \in V} \hat{\sigma}_{c,w}(s,v) r_t(v).$$

Now, we use the Monte Carlo method to obtain $\hat{\sigma}_{c,w}(s,v)$. It is reported in [17] that BiPPR is significantly faster than both the Monte Carlo method and the backward push method.

4.2 Limitation of Monte Carlo based methods on general similarity notions

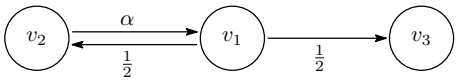


Figure 1: A weighted directed graph for which the variance of the naive Monte Carlo estimator may be infinity

As explained before, the Monte Carlo method cannot be directly used to compute general similarity notions. However, with a slight modification to the Monte Carlo method, we can construct an unbiased estimator for any similarity notion. Unfortunately, its performance is not generally good: The variance of the estimator can be infinite. Below, we explain how we can construct an unbiased estimator and its performance limitation more in detail.

In the case of the PPR, the matrix H itself was used as a transition matrix of a random walk. For a general simliarity notion, we introduce a transition probability matrix P and generate random walks based on P, where P satisfies $\|P\|_\infty \leq 1$ and $P(u,v) > 0$ if and only if $uv \in E$ [7, 23, 29]. Using P, A random walk $W = (v_0, v_1, \ldots, v_\ell)$ starting at vertex u is generated as follows:

(1) $\ell \leftarrow 0, v_0 \leftarrow u$.
(2) Repeat the following:
   - With probability $1 - \sum_{w \in V} P(v_\ell, w)$, terminate the repetition. For every $w \in V$, with probability $P(v_\ell, w)$, $\ell \leftarrow \ell + 1$ and $v_\ell \leftarrow w$.

In general, the distribution of w_ℓ generated this way is not equal to $\sigma_{c,w}(s,v)$. To address this issue, we define *correction weight* $\gamma(W)$ for each walk $W = (v_0, v_1, \ldots, v_\ell)$ as

$$\gamma(W) = \prod_{i=1}^{\ell} \frac{w(v_{i-1}v_i)}{P(v_{i-1}, v_i)}. \tag{3}$$

Suppose that we have generated N walks $W_1, \ldots, W_N$ all starting at s. Then, the Monte Carlo estimation of $\sigma_{c,w}(s,v)$ for every $v \in V$ is given by

$$\hat{\sigma}_{c,w}(s,v) = \frac{1}{N} \sum_{i \in \{1,\ldots,N\}: W_i \text{ ends at } v} \gamma(W_i).$$

We can check that $\hat{\sigma}_{c,w}(s,v)$ is an unbiased estimator of $\hat{\sigma}_{c,w}(s,v)$ regardless of the choice of P.

To use this algorithm, we need to choose P. A naive approach will be to set $P(u,v)$ as the *row-normalized* value of $w(uv)$, that is:

$$P(u,v) = \frac{w(uv)}{c + \sum_{v' \in V, uv' \in E} w(uv')}.$$

However, the variance of the estimator by the Monte Carlo method with this P can be infinite. To see this, we consider the weighted directed graph shown in Figure 1. In Figure 1, the value near the edge from v_i to v_j corresponds to $w(v_iv_j)$. If $\alpha < 2$, the similarity $\sigma_{c,w}(v_1, v_3)$ is well-defined and $\sigma_{c,w}(v_1, v_3) = \frac{c}{2-\alpha}$. We can also see $\hat{\sigma}_{c,w}(v_1, v_3) = \frac{c}{2-\alpha}$. However, $\mathrm{E}[\sigma_{c,w}(v_1, v_3)^2]$ on $N = 1$ is $\mathrm{E}[\sigma_{c,w}(v_1, v_3)^2] = \sum_{i=0}^{\infty} \left(\frac{\alpha(1+c)(\alpha+c)}{2}\right)^i \cdot \frac{(1+c)c^2}{2}$, where i corresponds to the number of visits of a walk to v_2. If $\alpha > \sqrt{2}$, this does not converge regardless of the value of $c > 0$. This also implies that the variance of $\sigma_{c,w}(v_1, v_3)$ is infinite, and the estimation quality cannot be improved by increasing N. Thus, we cannot rely on this estimator for the Monte Carlo estimation of similarity values.

4.3 Overview of our method

Our goal in this study is to obtain efficient methods for the one-to-one similarity estimation problem. To this end, we extend BiPPR [17] to general walk-based similarity notions. As explained, BiPPR combines the Monte Carlo method and the backward push method. Although the backward push method is applicable to general walk-based similarities, the Monte Carlo method cannot be directly applied to general similarities because it relies on the fact that the weights of edges form a transition matrix of a random walk.

To address this problem, we first generalize our definition of walk-based similarities so that different vertices can have different *termination weight*. Then, we introduce a *random-walk reduction* method: From any weight function $w : E \to \mathbb{R}_+$ and scalar coefficient $c \in \mathbb{R}$, this method constructs a pair of a weight function $w' : E \to \mathbb{R}_+$ and a *termination function* $\tau' : V \to \mathbb{R}_+$. The pair (τ', w') is designed so that (i) it forms a transition matrix of a random walk, and (ii) we can easily recover the original similarity value $\sigma_{c,w}(u,v)$ from the similarity value $\sigma_{\tau',w'}(u,v)$ defined by (τ', w').

This reduction method enables us to sample a vertex v with probability proportional to $\sigma_{c,w}(s,v)$ for the source vertex $s \in V$. Thus, we can replace random walks in the Monte Carlo part of BiPPR with this sampling method. Formally, the ability of our reduction method can be stated as follows:

Theorem 4.1. *Let (c,w) be a walk-based similarity on a graph $G = (V,E)$. Then, after preprocessing in $O(m \cdot i)$ time, we can construct an oracle of $O(m)$ space with which, for any given vertex $u \in V$, we can sample a vertex $v \in V$ with probability* almost *proportional to $\sigma_{c,w}(u,v)$. Here, i is the number of iterations in power iteration. The probability with which v is sampled can be made arbitrarily close to $\frac{\sigma_{c,w}(u,v)}{\sum_{v' \in V} \sigma_{c,w}(u,v')}$ by taking a sufficiently large i.*

The idea of buliding an oracle for Monte Carlo estimation of the solution of a linear system $(I - H)\boldsymbol{x} = \boldsymbol{b}$ with $\rho(H) < 1$ but $\|H\|_\infty > 1$ resembles [29]. The algorithm in [29] maintains multiple probability distributions for random walks, while ours require just one. Also, while the algorithm in [29] assigns non-uniform correction weights (similar to (3)), ours enables us to sample walks with probability (almost) proportional to their weights. Thus we can expect that ours have a wider range of applications.

In Section 5, we discuss how to construct (τ', w') from (c,w), and in Section 6, we give a detailed explanation of our algorithm.

5 REDUCTION TO RANDOM WALK

In this section, we describe our generic method that reduces the computation of walk-based similarities to the computation of the stationary distribution of a random walk.

To this end, we slightly generalize the notion of walk-based similarity. For a termination function $\tau\colon V \to \mathbb{R}_+$ and a weight function $w\colon E \to \mathbb{R}_+$, we define the weight of a walk $W = (v_0, \dots, v_\ell)$ as

$$w_\tau(W) = \tau(v_\ell) \prod_{i=1}^{\ell} w(v_{i-1}v_i).$$

Then, we define the similarity $\sigma_{\tau,w}(s,t)$ from a vertex $s \in V$ to a vertex $t \in V$ as

$$\sigma_{\tau,w}(s,t) = \sum_{W \in \mathrm{walk}(s,t)} w_\tau(W).$$

Note that we no longer use the scalar coefficient $c \in \mathbb{R}_+$ because it can be included in τ.

Our goal is to construct other termination function $\tau'\colon V \to \mathbb{R}_+$ and weight function $w'\colon E \to \mathbb{R}_+$ such that

(i) $\sigma_{\tau',w'}(s,\cdot)$ is given by a stationary distribution of a random walk from $s \in V$.
(ii) We can efficiently compute $\sigma_{\tau,w}$ from $\sigma_{\tau',w'}$.

To satisfy (i), τ' and w' must define a transition of a random walk. We first ask for the following condition:

Definition 5.1 (Random walkability). *For functions $\tau\colon V \to \mathbb{R}_+$ and $w\colon E \to \mathbb{R}_+$, we call the pair (τ, w)* randomly walkable *if*

- $\tau(v) \in [0,1]$ *for every* $v \in V$,
- $w(uv) \in [0,1]$ *for every* $uv \in E$, *and*
- $\tau(u) + \sum_{v \in N^+(u)} w(uv) = 1$ *for every* $u \in V$.

Now we discuss (ii). Note that sometimes it is impossible to satisfy $\sigma_{\tau,w}(u,v) = \sigma_{\tau',w'}(u,v)$: If it were possible, then we have $\sum_{v\in V} \sigma_{\tau,w}(u,v) = \sum_{v\in V} \sigma_{\tau',w'}(u,v) = 1$, where the last equality holds because $\sigma_{\tau',w'}$ is given by a stationary distribution of a random walk. However, we may not have $\sum_{v\in V} \sigma_{\tau,w}(u,v) = 1$.

To address the issue above, we consider the following unary quantity

$$c_{\tau,w}(u) = \sum_{v \in V} \sigma_{\tau,w}(u,v), \tag{4}$$

which can be seen as the closeness centrality [5, 6] defined via $\sigma_{\tau,w}$, and we impose the following relaxed condition:

Definition 5.2 (Recoverability). *We say that a pair (τ', w')* recovers *a pair (τ, w) if*

$$\sigma_{\tau',w'}(u,v) = \frac{\sigma_{\tau,w}(u,v)}{c_{\tau,w}(u)} \tag{5}$$

holds for every $u, v \in V$.

The main theorem of this section is as follows:

Theorem 5.3. *For any pair (τ, w), there exists a random-walkable pair (τ', w') that recovers (τ, w).*

We note that, to recover $\sigma_{\tau,w}(u,v)$, we need $c_{\tau,w}(u)$ along with $\sigma_{\tau',w'}(u,v)$. However, computing (an approximation to) $c_{\tau,w}(u)$ is an easier task than directly computing $\sigma_{\tau,w}(u,v)$. We discuss this issue in further detail in Section 6.

Next, for notational simplicity, we write $\sigma(u,v)$ and $\sigma'(u,v)$ to denote $\sigma_{\tau,w}(u,v)$ and $\sigma_{\tau',w'}(u,v)$, respectively. Similarly, we write $c(u)$ and $c'(u)$ to denote $c_{\tau,w}(u)$ and $c_{\tau',w'}(u)$, respectively. The following lemma is useful to guarantee recoverability.

Lemma 5.4. *Suppose that a pair (τ', w') satisfies*

$$w'(W) = \frac{w(W)}{c(u)} \tag{6}$$

for any $u \in V$ and any walk W starting from u. Then, (τ', w') recovers (τ, w).

Proof. The claim holds because

$$\sigma'(u,v) = \sum_{W \in \mathrm{walk}(u,v)} w'(W) = \sum_{W \in \mathrm{walk}(u,v)} \frac{w(W)}{c(u)} = \frac{\sigma(u,v)}{c(u)}.$$

□

Now, we derive the exact form of τ' and w' from the recoverability condition. Considering a walk W consisting of a single vertex $u \in V$, we must have

$$\tau'(u) = \frac{w(W)}{c(u)} = \frac{\tau(u)}{c(u)} \tag{7}$$

Considering walks $W = (v_0, v_1, \dots, v_\ell)$ and $W' = (v_1, \dots, v_\ell)$, we must have

$$w'(v_0v_1) = \frac{w'(W)}{w'(W')} = \frac{w(W)}{c(v_0)} \cdot \frac{c(v_1)}{w(W')} = w(v_0v_1) \cdot \frac{c(v_1)}{c(v_0)} \tag{8}$$

We now show that the pair (τ', w') satisfies the desired conditions.

Lemma 5.5. *The pair (τ', w') satisfies random walkability.*

Proof. We have

$$\tau'(u) + \sum_{v \in N^+(u)} w'(uv) = \frac{\tau(u) + \sum_{v\in N^+(u)} w(uv)c(v)}{c(u)} = 1$$

for every $u \in V$. As $\tau'(u) \geq 0$ and $w'(uv) \geq 0$, it is clear that $\tau'(u) \in [0,1]$ and $w'(uv) \in [0,1]$ hold. □

Lemma 5.6. *The pair (τ', w') recovers (τ, w).*

Proof. By Lemma 5.4, it suffices to check the condition (6). Let $W = (v_0, \dots, v_\ell)$ be an arbitrary walk. Then, we have

$$w'(W) = \prod_{i=1}^{\ell} \left(w(v_{i-1}v_i) \cdot \frac{c(v_i)}{c(v_{i-1})} \right) \cdot \frac{\tau(v_\ell)}{c(v_\ell)} = \frac{w(W)}{c(v_0)}.$$

□

Theorem 5.3 is established by combining Lemmas 5.5 and 5.6.

6 ALGORITHMS

In Section 5, we have seen that computing walk-based similarities can be reduced to computing the stationary distribution of a random walk. To perform random walk, we need to be able to quickly sample random neighbors of a vertex in the weighted graph obtained by the reduction. In Section 6.1, we explain how to implement an oracle with which we can randomly sample neighbors of a vertex, given that we can afford to compute the exact values of $c_{\tau,w}(u)$ (Refer to (4) for the definition). In Section 6.2, we discuss how to implement the oracle when we only have approximations to $c_{\tau,w}(u)$. Finally, in Section 6.3, we describe our algorithm for the one-to-one similarity estimation problem. As in Section 5, we use symbols $\sigma(u,v)$, $\sigma'(u,v)$, $c(u)$, and $c'(u)$ for notational simplicity.

6.1 Random neighbor oracle

In this section, we present the following:

Theorem 6.1. *Let (τ', w') be the pair given in Theorem 5.3. Then, after preprocessing in $O(|V|^3 + |E|)$ time, we can provide an oracle such that, given a query vertex $u \in V$, it returns in $O(1)$ time*

- *a special symbol $\perp$ with probability $\tau'(u)$, and*
- *a vertex $v \in N^+(u)$ with probability $w'(uv)$.*

With the oracle provided in Theorem 6.1, we can perform random walk with respect to the pair (τ', w').

Proof. To obtain (τ', w'), we need to compute $c(u)$ for every $u \in V$. Because the similarity $\sigma(u, v)$ is defined as

$$\sigma(u,v) = \left\langle \boldsymbol{e}_u, \sum_{\ell=0}^{\infty} H^\ell \tau(v) \boldsymbol{e}_v \right\rangle = \left\langle \boldsymbol{e}_u, (I-H)^{-1}\tau(v)\boldsymbol{e}_v \right\rangle,$$

where H is the transition weight matrix defined in Section 3.1. (note that $\sum_{\ell=0}^{\infty} H^\ell \tau(v)\boldsymbol{e}_v$ converges if $\sigma_{\tau,w}(u,v)$ is well defined for every $(u,v) \in V^2$), $c(u)$ can be rephrased as

$$\begin{aligned} c(u) &= \sum_{v \in V} \left\langle \boldsymbol{e}_u, (I-H)^{-1}\tau(v)\boldsymbol{e}_v \right\rangle \\ &= \left\langle \boldsymbol{e}_u, (I-H)^{-1} \sum_{v \in V} \tau(v)\boldsymbol{e}_v \right\rangle = \left\langle \boldsymbol{e}_u, (I-H)^{-1}\boldsymbol{\tau} \right\rangle, \end{aligned}$$

where $\boldsymbol{\tau} \in \mathbb{R}^V$ is a vector with the v-th element being $\tau(v)$. Hence, we can compute $c(u)$ for every $u \in V$ in $O(|V|^3)$ time. Then, we can compute τ' and w' in $O(|V| + |E|)$ time.

Now, we can use Walker's alias method [26] to provide the desired oracle. We need $O(d^+(u))$ time to preprocess each vertex $u \in V$. Hence, the total preprocessing time is $O(|V|^3 + |E|)$. □

6.2 Random neighbor oracle with inexact centrality values

In Section 6.1, we have seen that we can implement a random neighbor oracle by computing $c(u)$. However, this requires $O(|V|^3)$ preprocessing time, which is impractical. To reduce the preprocessing time, in this section, we consider how to implement the random neighbor oracle using approximations $\hat{c}(u)$ to $c(u)$, computed by, for example, the power iteration method.

Considering the proof of Lemma 5.5, the following property was crucial to guarantee for (τ', w') satisfying random walkability:

$$c(u) = \tau(u) + \sum_{v \in N^+(u)} w(uv)c(v) \quad (u \in V). \tag{9}$$

This indicates that we cannot guarantee random walkability if we use approximations $\hat{c}$, instead of c, to define the pair (τ', w').

Therefore, we define another random-walkable pair $(\hat{\tau}', \hat{w}')$ using $\hat{c}$ by slightly changing the recovery condition. First, we define $\hat{c}_R(u)$ by

$$\hat{c}_R(u) = \tau(u) + \sum_{v \in N^+(u)} w(uv)\hat{c}(v) \quad (u \in V).$$

Using this quantity, we define a pair $(\hat{\tau}', \hat{w}')$ as

$$\hat{\tau}'(u) = \frac{\tau(u)}{\hat{c}_R(u)} \quad \text{and} \quad \hat{w}'(uv) = w(uv) \cdot \frac{\hat{c}(v)}{\hat{c}_R(u)}. \tag{10}$$

We can easily confirm the following:

Algorithm 3 Oracle construction

Input: weight function w, termination function τ

1: Compute $\hat{c}$, an approximation of $(I-H)^{-1}\boldsymbol{\tau}$
2: **for** $u \in V$ **do**
3: $\quad \hat{c}_R(u) \leftarrow \tau(u) + \sum_{v \in N^+(u)} w(uv)\hat{c}(u)$
4: $\quad f_u \leftarrow \left\{ \left(v, \frac{w(uv)\hat{c}(v)}{\hat{c}_R(v)} \right) \mid v \in N(u) \right\} \cup \left\{ \left(\perp, \frac{\tau(u)}{\hat{c}_R(v)} \right) \right\}$
5: $\quad \Omega_u \leftarrow$ random sampler that returns v with probability p for every $(v,p) \in f_u$
6: **end for**
7: Return $(\hat{c}, \{\Omega_u\}_{u \in V})$

Algorithm 4 Sampling using the oracle by Algorithm 3

Input: source vertex u, $(\hat{c}, \{\Omega_u\}_{u \in V})$ given by Algorithm 3

1: **loop**
2: $\quad v \leftarrow$ return value of an oracle call to Ω_u
3: $\quad$ Return u if $v = \perp$. Otherwise, $u \leftarrow v$.
4: **end loop**

Proposition 6.2. *The pair $(\hat{\tau}', \hat{w}')$ satisfies random walkability.*

Following the proof of Theorem 6.1, we obtain the following:

Theorem 6.3. *Suppose that we are given a vector $\hat{\boldsymbol{c}} \in \mathbb{R}^V$. Let $(\hat{\tau}', \hat{w}')$ be the pair given by (10). Then, after preprocessing in $O(|V| + |E|)$ time, we can provide an oracle such that, given a query vertex $u \in V$, it returns in $O(1)$ time*

- *a special symbol $\perp$ with probability $\hat{\tau}'(u)$, and*
- *a vertex $v \in N^+(u)$ with probability $\hat{w}'(uv)$.*

The pseudocode of the modified oracle is given in Algorithm 3. The procedure for sampling vertices using the random neighbor oracle is shown in Algorithm 4. It is guaranteed that Algorithm 4 returns a vertex $v \in V$ with probability $\hat{\sigma}'(u,v)$, where $\hat{\sigma}'(u,v) = \sigma_{\hat{\tau}', \hat{w}'}(u,v)$. In Section 7, we discuss the error in using $\hat{\sigma}'(u,v)$ to approximate $\sigma(u,v)$.

6.3 Similarity estimation

We can estimate walk-based similarities by simulating the Monte Carlo part of BiPPR [17] using the random neighbor oracle described in Sections 6.1 and 6.2. However, the transition weights produced by our reduction have different properties from that of PPR:

- Monte Carlo itself only estimates $\hat{\sigma}'(u,v)$. We need to multiply this value by $\hat{c}(u)$ in order to get an estimate of $\sigma(u,v)$, but this may affect the accuracy of the estimation.
- Termination probabilities may be different among vertices. Therefore, the efficiency of push algorithms largely depends on the source vertex.

Hence, we modify the original BiPPR as follows:

Threshold depending on $\hat{c}(u)$. As we see later in Theorem 7.1, $\hat{\sigma}'(u,v)\hat{c}(u)$ is a good approximation to $\sigma(u,v)$. Assume that the difference between $\hat{\sigma}'(u,v)\hat{c}(u)$ and $\sigma(u,v)$ is small enough to be ignored. Then, if $\hat{\sigma}'(u,v)$ is an (ϵ, δ', p)-estimation of $\sigma'(u,v)$, then $\hat{\sigma}'(u,v)\hat{c}(u)$ will be an $(\epsilon, \delta'\hat{c}(u), p)$-estimation of $\sigma(u,v)$. Therefore, if we want to have an (ϵ, δ, p)-estimation of $\sigma_{\tau,w}(u,v)$, we

should set $\delta' = \frac{\delta}{\hat{c}(u)}$. We will use $\frac{\delta}{\hat{c}(u)}$ as the threshold for estimation of $\hat{\sigma}'(u, v)$.

Adaptive thresholding. As indicated in [17, 18], we can dynamically change the threshold for terminating backward pushes. We take a balance between the running time of the backward push part and that of the Monte Carlo part. This technique is especially important for the walk-based similarity estimation, because the cost of backward pushes is difficult to predict and may largely depend on the target vertex. We can estimate the (average) running time of the Monte Carlo part as the product of the average walk time and the number of required walks.

We note that the average walk length may vary depending on the starting vertex. Therefore, we computed the average walk time (using Monte Carlo sampling) from the source vertex at the beginning of each query computation. We also note that the walks obtained in this step can be reused in later steps.

Integrating forward push into Monte Carlo. Instead of generating walks from the source vertex, we can perform forward pushes and then generate walks from each vertex with positive residuals [28].

Suppose we want to generate N walks in the Monte Carlo part for estimation of $\hat{\sigma}'$. Let $\boldsymbol{p}_s$ and $\boldsymbol{r}_s$ be the reserve and residue vectors after running forward push algorithm, respectively. From the invariant of forward push (1), if we have unbiased estimations of $\hat{\sigma}'(u, v)$ for every $u \in V$, then $\hat{\sigma}(s, v)$ defined by the following formula is also an unbiased estimation of $\hat{\sigma}'(s, v)$:

$$\hat{\sigma}(s, v) = p_s(v) + \sum_{u \in V} r_s(u)\hat{\sigma}'(u, v).$$

Here, $\hat{\sigma}'(u, v)$ can be obtained by Monte Carlo estimation from vertex u. If we use at least $\lceil r_s(u)N \rceil$ walks for this estimation for every $u \in V$, we can show that the variance of $\hat{\sigma}(s, v)$ defined above is at most that of the estimator by naive Monte Carlo with N walks. With this technique, we can reduce the cost of tracking many identical walks in the Monte Carlo part.

Final algorithm. Combining the techniques discussed above, we obtain the final algorithm for similarity estimation. This algorithm is shown in Algorithm 5. Here γ is a parameter for controlling accuracy and ι is the number of trials of generating walks for estimating the walk time. Note that this requires the random neighbor oracle generated by Algorithm 3. We can precompute this oracle and use it for answering each query to reduce the computation cost.

In Algorithm 5, we used the terminate condition $N_0 \max_u r_s(u) < 1$. This means that we terminate when the number of walks to be generated from every vertex is less than 1. As this condition may cause too many forward pushes, in Section 8, we also consider a heuristic that uses a terminate condition depending on $\hat{c}(s)$.

As with the original BiPPR [17], we can expect that Algorithm 5 significantly improves the efficiency of similarity estimation. However, as we are considering general walk-based similarity, it is difficult to give upper bounds for running time of this algorithm. Analyzing the time complexity will be a future work.

Algorithm 5 Similarity estimation

Input: source vertex s, target vertex t, threshold δ, parameter γ and ι, $(\hat{c}, \{\Omega_u\}_{u \in V})$ given by Algorithm 3

$\delta' \leftarrow \frac{\delta}{\hat{c}(s)}$
$\tau_{\mathrm{walk}} \leftarrow$ (Time to generate ι walks from s)$/\iota$
Perform Algorithm 2 with the following terminate condition:
  Let τ_{push} be the time required for backward push so far. Terminate if $\frac{\gamma \max_v r_t(v)}{\delta'} \cdot \tau_{\mathrm{walk}} < \tau_{\mathrm{push}}$
$N_0 \leftarrow \lceil \frac{\gamma \max_v r_t(v)}{\delta'} \rceil$
$\hat{\sigma} \leftarrow q_t(s)$
Perform Algorithm 1 using r'_s instead of r_s with the terminate condition of $N_0 \max_u r'_s(u) < 1$
for $u \in V, p_s(u) > 0$ **do**
  $\hat{\sigma} \leftarrow \hat{\sigma} + p_s(u) r_t(u)$
end for
for $u \in V, r'_s(u) > 0$ **do**
  $\omega \leftarrow \frac{\hat{c}(u)}{\hat{c}(s)} N_0 r'_s(u),\ N \leftarrow \lceil \omega \rceil$
  for $i = 1, 2, \ldots, N$ **do**
    $v \leftarrow$ vertex returned by a call to Algorithm 4 with source s and oracle $(\hat{c}, \{\Omega_u\}_{u \in V})$
    $\hat{\sigma} \leftarrow \hat{\sigma} + \frac{\omega}{N} \cdot \frac{1}{N_0} \hat{c}(s) r_t(v)$
  end for
end for
Return $\hat{\sigma}$ as an estimation of $\sigma_{\tau, w}(s, t)$

7 ERROR ANALYSIS

In this section, we provide theoretical guarantees of our algorithm.

First, we analyze the approximation error caused by using the inexact random neighbor oracle given in Section 6.2. We borrow notations from Section 6. Then, we show that $\hat{\sigma}'(u, v)$ and $\sigma(u, v)$ are close:

Theorem 7.1. *Suppose that*

$$\frac{1}{1+\kappa} \hat{c}(u) \le \hat{c}_R(u) \le (1+\kappa)\hat{c}(u) \quad (u \in V) \tag{11}$$

holds for some $\kappa \in (0, 1)$ *and that* $\lambda := \min_{u \in V}(1 - \hat{\tau}'(u)) < \frac{1}{1+\kappa}$ *holds. Then, we have*

$$\underline{\alpha}(\sigma(u, v))^{1+\beta} \le \hat{\sigma}'(u, v)\hat{c}(u) \le \overline{\alpha}(\sigma(u, v))^{1-\beta}$$

where

$$\underline{\alpha} = \frac{(1-\lambda)^{1+\beta}}{(1+\kappa) - \lambda},\ \overline{\alpha} = \frac{(1-\lambda)^{1-\beta}}{(1+\kappa)^{-1} - \lambda},\ \textit{and } \beta = \log_{\frac{1}{\lambda}}(1+\kappa).$$

Before proving Theorem 7.1, we discuss its implication. Theorem 7.1 claims that the relative error of $s_{\hat{\tau}', \hat{w}'}(u, v)\hat{c}_R(u)$ is controlled by κ. The following proposition states that we can make κ arbitrarily small if we use the power iteration method to compute $\hat{c}_R(u)$.

Proposition 7.2. *Let* $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots$ *be a sequence that converges to* $\boldsymbol{x} = (I - H)^{-1}\boldsymbol{\tau}$, *where* $H \in \mathbb{R}^{V \times V}$ *is a transition weight matrix with* $\rho(H) < 1$. *Then, for any* $\kappa > 0$, *there exists* i *such that*

$$\frac{1}{1+\kappa} x_i(u) \le \tau(u) + \sum_v H(u, v) x_i(v) \le (1+\kappa) x_i(u) \tag{12}$$

holds for every $u \in V$ *with* $x(u) > 0$.

Proof. Let $\mu = \min\{x(u) \mid u \in V,\ x(u) \neq 0\}$. From the convergence of $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots$, there exists i such that $\|\boldsymbol{x}_i - \boldsymbol{x}\|_\infty < \delta'$ where $\delta' = \frac{\delta\mu}{2(1+\|H\|_\infty)}$. Then, for $u \in V$ with $x(u) > 0$, we have

$$\begin{aligned}
&\tau(u) + \sum_v H(u,v)x_i(v) \\
&= \tau(u) + \sum_v H(u,v)x(v) + \sum_v H(u,v)(x_i(v) - x(v)) \\
&\geq x(u) - \|H\|_\infty\delta' \geq x_i(u) - (1 + \|H\|_\infty)\delta' \\
&\geq \frac{1}{1+\delta}x_i(u)
\end{aligned}$$

as desired. The second inequality of (12) follows analogously. □

If $\boldsymbol{x}_i$ satisfies (12) and we choose $\hat{\boldsymbol{c}} = \boldsymbol{x}_i$, then (11) is satisfied and the approximation guarantee given in Theorem 7.1 follows. Hence, we can use (12) as a terminate condition in the power iteration.

To prove Theorem 7.1, we need the following technical lemma, whose proof is deferred to the full version:

Lemma 7.3. *Let $\lambda, \kappa \in (0,1)$ be such that $\lambda(1+\kappa) < 1$. Consider two sequences $\{a_i\}_{i\geq 0}, \{b_i\}_{i\geq 0}$ such that $a_i \in [0, \lambda^i]$ and $b_i \in [a_i(1+\kappa)^{-i}, a_i(1+\kappa)^i]$ for every $i \geq 0$. Let $s = \sum_{i=0}^\infty a_i < \infty$. Then, we have*

$$\frac{(1-\lambda)^{1-\eta}}{1-\lambda(1+\kappa)^{-1}}s^{1-\eta} \leq \sum_{i=0}^\infty b_i \leq \frac{(1-\lambda)^{1+\eta}}{1-\lambda(1+\kappa)}s^{1+\eta}. \tag{13}$$

for $\eta = \log_\lambda(1+\kappa)$.

Proof of Theorem 7.1. Let $\sigma_\ell(u,v)$ denote the sum of weights of walks from u to v of length ℓ, that is,

$$\sigma_\ell(u,v) = \sum_{W \in \mathrm{walk}_\ell(u,v)} w_\tau(W).$$

Then, clearly $\sigma(u,v) = \sum_{\ell=0}^\infty \sigma_\ell(u,v)$ holds. Similarly, we define $\sigma_\ell(u) = \sum_{v\in V}\sigma_\ell(u,v)$.

First, we analyze the relation between $w'(W) := w_{\tau'}(W)$ (which is obtained using $\boldsymbol{s}$) and $\hat{w}'(W) := w_{\hat{\tau}'}(W)$ for a walk $W = (v_0, v_1, \ldots, v_\ell)$. By the definition of $\hat{w}'(u,v)$ and $(\hat{\tau}', \hat{w}')$, we have

$$\begin{aligned}
\hat{w}'(W) &= \hat{\tau}'(v_\ell)\prod_{i=1}^\ell \hat{w}'(v_{i-1}v_i) = \frac{\tau(v_\ell)}{\hat{c}_R(v_\ell)}\prod_{i=1}^\ell\left(w(v_{i-1}v_i)\cdot\frac{\hat{c}(v_i)}{\hat{c}_R(v_{i-1})}\right) \\
&= \left(\frac{\tau(v_\ell)}{\hat{c}_R(v_0)}\prod_{i=1}^\ell w(v_{i-1}v_i)\right)\cdot\prod_{i=1}^\ell\frac{\hat{c}(v_i)}{\hat{c}_R(v_i)} = \frac{w_\tau(W)}{\hat{c}_R(v_0)}\cdot\prod_{i=1}^\ell\frac{\hat{c}(v_i)}{\hat{c}_R(v_i)}.
\end{aligned}$$

Taking the sum of $\hat{w}'(W)$ over all walks W from u to v of length ℓ, from the assumption on $\hat{c}_R(u)$, we have

$$\sigma_\ell(u,v)(1+\kappa)^{-\ell} \leq \hat{\sigma}'_\ell(u,v)\hat{c}_R(u) \leq \sigma_\ell(u,v)(1+\kappa)^\ell.$$

From random walkability, we have

$$0 \leq \hat{\sigma}'_\ell(u,v) \leq \hat{c}'_\ell(u) \leq \lambda^\ell \text{ and } \sum_{\ell=0}^\infty \hat{c}'_\ell(u) = \hat{c}'(u) = 1.$$

Therefore, using Lemma 7.3 and (11), we obtain

$$\underline{\alpha}\cdot\sigma(u,v)^{1+\beta} \leq \hat{\sigma}'(u,v)\hat{c}(u) \leq \overline{\alpha}\cdot\sigma(u,v)^{1-\beta}.$$ □

Next, we give a guarantee on the estimation quality of Algorithm 5.

Theorem 7.4. *Suppose that (11) holds for a sufficiently small κ. Then, Algorithm 5 returns an (ϵ,δ,p)-estimation of $\sigma(s,t)$ if we set $\gamma = \Omega(\epsilon^{-2}\log(1/p))$.*

We show this theorem using the following lemma:

Lemma 7.5. *Suppose that $\hat{c} = c$ holds. Then, Algorithm 5 returns an (ϵ,δ,p)-estimation of $\sigma(s,t)$ if we set $\gamma = \Omega(\epsilon^{-2}\log(1/p))$*

Proof. First, we show the claim holds if we modify Algorithm 5 so that we do not perform forward pushes, and instead we use $p_s(u) = 0$ $(u \in V)$ and $r_s(u) = \begin{cases} 1 & u = s \\ 0 & u \neq s \end{cases}$. In this case, the proof is almost identical to that of the original BiPPR given in [17]. Let $r_{\max} = \max_v r_t(v)$, v_i be the vertex chosen in the i-th iteration and $X_i = \frac{r_t(v_i)}{r_{\max}}$. Then,

$$\hat{\sigma} = q_t(s) + \frac{c(s)}{N_0}\sum_{i=1}^{N_0} r_t(v_i) = q_t(s) + \frac{c(s)r_{\max}}{N_0}\sum_{i=1}^{N_0} X_i.$$

We can analyze the concentration of $\frac{1}{N_0}\sum_{i=1}^{N_0} X_i$ using the Chernoff bound and thus obtain the desired statement.

For the complete proof for Algorithm 5, we return to the proof of the Chernoff bound (see, e.g., [11]). The Chernoff bound can be proven by bounding $\Pr[e^{\lambda X} > e^{\lambda a}]$ $(\lambda > 0)$ using Markov inequality

$$\Pr[e^{\lambda X} > e^{\lambda a}] < \frac{\mathrm{E}[e^{\lambda X}]}{e^{\lambda a}}$$

and optimizing λ. Therefore, if two random variables X, Y satisfy $\mathrm{E}[e^{\lambda X}] \leq \mathrm{E}[e^{\lambda Y}]$ for any $\lambda > 0$, Chernoff bounds satisfied by Y are all satisfied by X.

Let $v(s)$ be a random vertex that is chosen with probability equal to the similarity from s and

$$\Phi = \prod_{u\in V} e^{\lambda p_s(u)\frac{c(u)}{c(s)}\frac{r_t(u)}{r_{\max}}}\cdot\mathrm{E}\left[e^{\lambda\frac{c(u)}{c(s)}\frac{r_t(v(u))}{r_{\max}}}\right]^{N_0 r'_s(u)}$$

$$\Phi' = \prod_{u\in V} e^{\lambda p_s(u)\frac{c(u)}{c(s)}\frac{r_t(u)}{r_{\max}}}\cdot\mathrm{E}\left[e^{\lambda\frac{c(u)}{c(s)}\frac{r_t(v(u))}{r_{\max}}\frac{N_0 r'_s(u)}{\lceil N_0 r'_s(u)\rceil}}\right]^{\lceil N_0 r'_s(u)\rceil}.$$

It can be verified that $\Phi' = \mathrm{E}[e^{\lambda(\hat{\sigma}-q_t(s))}]$. Before performing forward pushes, $\Phi = \Phi_0 = \mathrm{E}\left[e^{\lambda\sum_{i=1}^{N_0}X_i}\right]$. From the independence of $X_1, X_2, \ldots$, we have $\Phi_0 = \left(\mathrm{E}\left[e^{\lambda X}\right]\right)^{N_0}$, where $X = \frac{r_t(v(s))}{r_{\max}}$. We can show that Φ is non-increasing in a forward push operation by the convexity of e^x. Furthermore, $\Phi' \leq \Phi$ can be proven using Hölder's inequality. Finally, we have $\Phi' \leq \Phi_0$ and thus we obtain the desired statement for Algorithm 5 with forward push when $\hat{c} = c$. □

Proof of Theorem 7.4. For simplicity, we modify the line 5 of Algorithm 5 to $N_0 \leftarrow \frac{\gamma\max_v r_t(v)}{\delta'}$ in the analysis.

Suppose the vertices pushed in the execution of forward push in Algorithm 5 are $s = u_1, u_2, \ldots, u_k$ in this order. Then, we construct a graph $G' = (V', E')$ from $G = (V, E)$ in the following way:

(1) We let $V' \leftarrow V, E' \leftarrow E$ and define a mapping $f : V \to V'$ by $f(u) = u$.

Table 1: Tested networks and precomputation time

Name	Type	n	m	precomputation time (s)
com-dblp	Undirected	317K	2.10M	1.289
web-Stanford	Directed	282K	2.31M	1.060
com-youtube	Undirected	1.13M	5.98M	4.193
soc-pokec	Directed	1.63M	30.6M	16.517
soc-LiveJournal	Directed	4.85M	69.0M	40.555
com-orkut	Undirected	3.07M	234M	107.122

(2) For each $i = k, k-1, \ldots, 1$, we create a fresh vertex x_i. Then, we let $V' \leftarrow V' \cup \{x_i\}$, $E' \leftarrow E' \cup \{(x_i, f(v)) \mid v \in N^+(u_i)\}$, $\hat{c}(x_i) \leftarrow \tau(u_i) + \sum_{v \in N^+(u)} w(u_i v)\hat{c}(f(v))$, $\hat{\tau}'(x_i) = \frac{\tau(u_i)}{\hat{c}(x_i)}$, $w(x_i v) = \hat{w}'(x_i v) = w(u_i v) \cdot \frac{\hat{c}(v)}{\hat{c}(x_i)}$ and finally, $f(u_i) \leftarrow x_i$.

Note that this construction of $\hat{c}, \hat{\tau}'$ and $\hat{w}'$ is well-defined since $\hat{c}(x_i)$ depends only on $x_{i+1}, \ldots, x_k$ and $u \in V$.

Suppose we run Algorithm 5 on G' instead of G, with the sequences of pushed vertices as follows:

- Suppose pushed vertices in the forward push of the run for G are $v_1, \ldots, v_l$ and let $f(v) = \{v\} \cap \{x_i \mid u_i = v\}$. Then, those for G' are $f(v_1), \ldots, f(v_l)$. Here, vertices in $f(v_i)$ for one i are pushed *at the same time.*
- Pushed vertices in the backward push are $x_1, \ldots, x_k$ in this order.

Then, the distribution of $\hat{\sigma}$ is identical for the runs for G and G'.

We can analyze the concentration of Algorithm 5 for G' (and the sequence of pushed vertices as described above) using the same way as in the proof of Lemma 7.5, because the invariant (1) is maintained in forward pushes for this setting. Thus we can argue that, if we set $\gamma = \Omega(\epsilon^{-2} \log(1/p))$, Algorithm 5 returns an (ϵ, δ, p)-estimation of $\mathrm{E}[\hat{\sigma}]$. Therefore, it suffices to show that $\mathrm{E}[\hat{\sigma}]$ can be made arbitrarily close to $\sigma(s,t)$ by choosing sufficiently small κ. Since we can obtain the same bounds for $\mathrm{E}[\hat{\sigma}]$ as those for $\hat{c}_R(u)$ in Theorem 7.1, the claim holds. □

8 EXPERIMENTS

We experimentally evaluated our algorithms for one-to-one walk-based similarity estimation. The primary objective of this experiment is to show that our algorithms are efficient in the following extreme condition, which previous algorithms are difficult to handle: $\max_i \sum_j H(i,j)$ and $\max_j \sum_i H(i,j)$ are significantly larger than 1, where H is the transition weight matrix defined in Section 3.1.

Katz similarity with β close to $\frac{1}{\rho(A)}$ satisfies the above condition. Also, recall that Katz similarity can express any walk-based similarity by appropriately setting weights on edges. Hence, in this experiment, we used Katz similarity as a representative similarity notion and set $\beta = \frac{1}{1.1\rho(A)}$. Datasets used, all taken from SNAP [16], are shown in Table 1. Our codes are provided in the supplementary material.

Query generation. For each tested network, one-to-one similarity queries were generated as follows. First, we randomly chose the source vertex s uniformly or with probability proportional to Katz similarity values. Then, we computed similarity vector from s using power iteration. Finally, we chose the target vertex $t \neq s$ uniformly at random such that $\sigma_{\mathrm{Katz}}(s,t) \geq \delta_{\mathrm{index}} := \frac{1}{10n}$. If no such t was found, we returned to the first step and chose another s. We repeated these procedures until 1000 queries were obtained.

Tested algorithms. We implemented the following algorithms;

- Power iteration (PI).
- Iterative method with upper/lower bound guarantees (IB) [9].
- Forward push (FP).
- BiPPR with naive Monte Carlo for general path-based similarities (described in Section 4.2) (NBP).
- Bidirectional algorithm (BI, ours): Algorithm 5 with forward push disabled.
- BI with forward push (BIF, ours): Algorithm 5.
- BIF with a modified terminate condition (BIF', ours): Algorithm 5. We used $N_0 \max_u r_s(u) < \hat{c}(s)$ as the terminate condition in Line 7 of Algorithm 5.

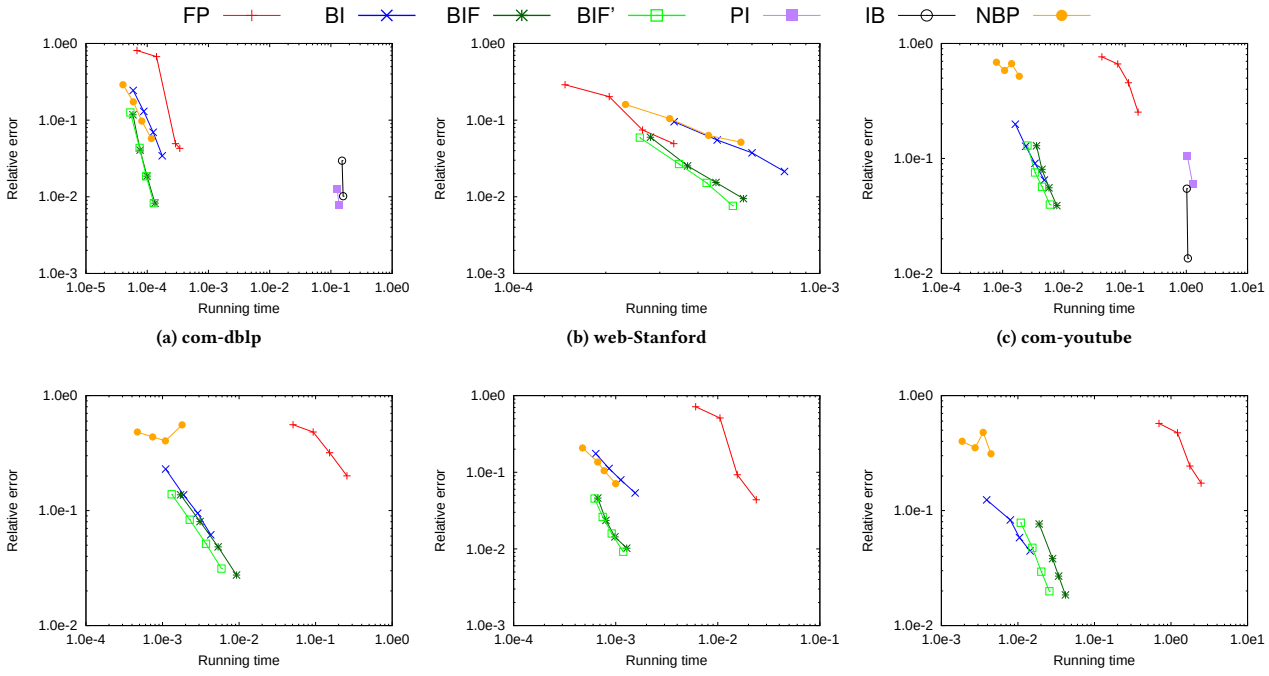
We implemented these algorithms in C++11 and ran them on a Linux (Ubuntu 16.04) desktop with Intel Core i7-6700 processor (3.4GHz) and 16 GiB of RAM.

Note that IB is applicable only to undirected networks. Furthermore, because PI and IB require iterations over all edges and thus take long execution time, we only ran them on com-dblp and com-youtube, which are relatively small.

To examine the relationships between execution time and estimation quality, we executed these algorithms with varying termination parameters δ. For BI, BIF, BIF' and NBP, we used $\gamma = 7$ as the original BiPPR [17]. Note that changing δ to $\frac{\delta}{c}$ is equivalent to changing γ to $c\gamma$. Therefore, using $\delta \neq \delta_{\mathrm{index}}$ is effectively equivalent to use $\delta = \delta_{\mathrm{index}}$ with different values of ϵ and p from the specified ones. For PI, the termination condition was set to $\|\boldsymbol{x}_i - \boldsymbol{x}_{i-1}\|_\infty < \delta$. For IB, we terminated iterations once the difference between upper and lower bounds on the similarity value became less than δ. For BI and BIF, we precomputed to provide the random neighbor oracle and used it in the query time.

Results. First, we report the precomputation time for generating the random neighbor oracle required for BI and BIF (see Table 1). The precomputation time is less than 2 min even for a graph with more than 200M edges (com-orkut), which is small, given that precomputation is required only once for each graph. Note that existing methods for similarity estimation such as BiPPR [17] and HubPPR [27] also use precomputation. In particular, it is reported that the latter method required more than 6,000 seconds for preprocessing com-orkut [27].

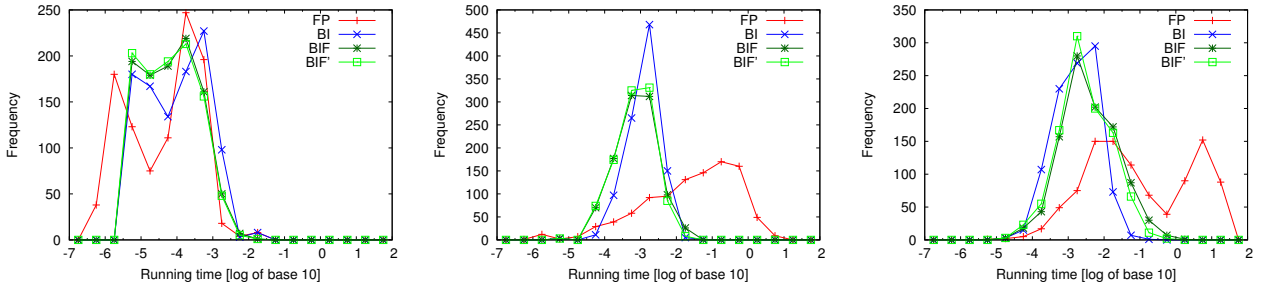
In Figure 2, we report the trade-off between the average relative error and the average running time for each instance. Except for web-Stanford, our proposed algorithms (BI, BIF, and BIF') outperform other baseline methods. The improvement is prominent in larger networks such as soc-pokec and com-orkut. In particular, for com-orkut, our algorithms achieved approximately 100 times faster running time and better relative errors than FP. BIF and BIF' also showed better error-time trade-offs than FP for web-Stanford. Additionally, BIF' consistently shows better performance than BIF. This suggests that controlling the amount of forward pushes is effective to improve performance. Finally, the performance of NBP significantly varies on different networks. Specifically, in com-youtube,



(a) com-dblp (b) web-Stanford (c) com-youtube

(d) soc-pokec (e) soc-LiveJournal (f) com-orkut

Figure 2: Average relative error vs. Average running time



(a) web-Stanford (b) soc-pokec (c) com-orkut

Figure 3: Distribution of running time

soc-pokec and com-orkut, increased running time (corresponding to smaller γ) does not necessarily improve the relative error. This implies that the performance of NBP can be unstable.

Figure 3 shows the distribution of query processing times. Parameters may be different among algorithms to equalize the relative error as much as possible. In this figure, a query with processing time t (in seconds) is assigned to the bin of width 0.5 containing the index $\log_{10} t$. We can observe that the processing time largely varies depending on the queries. As we have discussed before, vertices with larger centrality values tend to require more iterations. The query processing times of BI, BIF, and BIF' are more concentrated than that of FP except for BI in Figure 3a. This means that our algorithms are significantly faster than FP for difficult queries.

9 CONCLUSION

We proposed a *random-walk reduction* method that reduces the computation of any walk-based similarity to the computation of a stationary distribution of a random walk. With this reduction, we can exploit techniques for PPR to compute other walk-based similarities. As a concrete application, we designed an indexing method for walk-based similarities with which we can quickly estimate the similarity value of queried pairs of vertices, and theoretically analyze its approximation error. Our experimental results demonstrate that the instantiation of our method for Katz similarity is two orders of magnitude faster than existing methods on large real networks, without any deterioration in solution quality.

ACKNOWLEDGMENTS

S. Murai would like to thank Hiroshi Imai for offering insightful comments and discussions.

REFERENCES

[1] Reid Andersen, Christian Borgs, Jennifer T. Chayes, John E. Hopcroft, Vahab S. Mirrokni, and Shang-Hua Teng. 2007. Local Computation of PageRank Contributions. In *Proceedings of the 5th International Workshop on Algorithms and Models for the Web-Graph (WAW)*. 150–165.

[2] R. Andersen, F. Chung, and K. Lang. 2006. Local Graph Partitioning using PageRank Vectors. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 475–486.

[3] Shumeet Baluja, Rohan Seth, D Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. 2008. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proceeding of the 17th International Conference on World Wide Web (WWW)*. 895–904.

[4] Siddhartha Banerjee and Peter Lofgren. 2015. Fast Bidirectional Probability Estimation in Markov Models. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems (NIPS)*. 1423–1431.

[5] Alex Bavelas. 1950. Communication Patterns in TaskâĂŘOriented Groups. *The Journal of the Acoustical Society of America* 22, 6 (1950), 725–730.

[6] Murray A. Beauchamp. 1965. An improved index of centrality. *Behavioral Science* 10, 2 (1965), 161–163.

[7] Michele Benzi, Thomas M Evans, Steven P Hamilton, Massimiliano Lupo Pasini, and Stuart R Slattery. 2017. Analysis of Monte Carlo accelerated iterative methods for sparse linear systems. *Numerical Linear Algebra with Applications* 24, 3 (2017), e2088.

[8] Ivan Dimov, Sylvain Maire, and Jean Michel Sellier. 2015. A new Walk on Equations Monte Carlo method for solving systems of linear algebraic equations. *Applied Mathematical Modelling* 39, 15 (2015), 4494–4510.

[9] Pooya Esfandiar, Francesco Bonchi, David F. Gleich, Chen Greif, Laks V. S. Lakshmanan, and Byung-Won On. 2010. Fast Katz and Commuters: Efficient Estimation of Social Relatedness in Large Networks. In *Proceedings of the 7th International Workshop on Algorithms and Models for the Web-Graph (WAW)*. 132–145.

[10] Dániel Fogaras, Balázs Rácz, Károly Csalogány, and Tamás Sarlós. 2005. Towards Scaling Fully Personalized PageRank: Algorithms, Lower Bounds, and Experiments. *Internet Mathematics* 2, 3 (2005), 333–358.

[11] Michel Goemans. 2014. MIT Mathematics 18.310, Lecture Notes: Chernoff bounds, and some applications. https://ocw.mit.edu/courses/mathematics/18-310-principles-of-discrete-applied-mathematics-fall-2013/lecture-notes/MIT18_310F13_Ch4.pdf.

[12] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM* 35, 12 (1992), 61–70.

[13] Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (1953), 39–43.

[14] Samamon Khemmarat and Lixin Gao. 2016. Fast Top-K Path-Based Relevance Query on Massive Graphs. *IEEE Transactions on Knowledge and Data Engineering* 28, 5 (2016), 1189–1202.

[15] Joseph A Konstan, Bradley N Miller, David Maltz, Jonathan L Herlocker, Lee R Gordon, and John Riedl. 1997. GroupLens: applying collaborative filtering to Usenet news. *Commun. ACM* 40, 3 (1997), 77–87.

[16] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[17] Peter Lofgren, Siddhartha Banerjee, and Ashish Goel. 2016. Personalized PageRank Estimation and Search: A Bidirectional Approach. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining (WSDM)*. 163–172.

[18] Peter Lofgren, Siddhartha Banerjee, Ashish Goel, and Seshadhri Comandur. 2014. FAST-PPR: scaling personalized pagerank estimation for large graphs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1436–1445.

[19] Peter Lofgren and Ashish Goel. 2013. Personalized PageRank to a Target Node. *CoRR* abs/1304.4658 (2013).

[20] Linyuan Lu and Tao Zhou. 2010. Link Prediction in Complex Networks: A Survey. *CoRR* abs/1010.0725 (2010).

[21] Takanori Maehara, Takuya Akiba, Yoichi Iwata, and Ken-ichi Kawarabayashi. 2014. Computing Personalized PageRank Quickly by Exploiting Graph Structures. *PVLDB* 7, 12 (2014), 1023–1034.

[22] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report. Stanford InfoLab.

[23] Natalia Rosca. 2006. Monte carlo methods for systems of linear equations. *Studia Univ. BABES-BOLYAI Mathematica* 51 (2006).

[24] Nitin Shyamkumar, Siddhartha Banerjee, and Peter Lofgren. 2016. Sublinear estimation of a single element in sparse linear systems. In *Proceedings of the 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. 856–860.

[25] Han Hee Song, Tae Won Cho, Vacha Dave, Yin Zhang, and Lili Qiu. 2009. Scalable proximity estimation and link prediction in online social networks. In *Proceedings of the 9th ACM SIGCOMM Internet Measurement Conference (IMC)*. 322–335.

[26] Alastair J. Walker. 1977. An Efficient Method for Generating Discrete Random Variables with General Distributions. *ACM Trans. Math. Softw.* 3, 3 (Sept. 1977), 253–256.

[27] Sibo Wang, Youze Tang, Xiaokui Xiao, Yin Yang, and Zengxiang Li. 2016. HubPPR - Effective Indexing for Approximate Personalized PageRank. *PVLDB* 10, 3 (2016), 205–216.

[28] Sibo Wang, Renchi Yang, Xiaokui Xiao, Zhewei Wei, and Yin Yang. 2017. FORA: Simple and Effective Approximate Single-Source Personalized PageRank. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 505–514.

[29] Tao Wu and David F Gleich. 2016. Multi-way Monte Carlo Method for Linear Systems. *arXiv preprint arXiv:1608.04361* (2016).

[30] Yubao Wu, Ruoming Jin, and Xiang Zhang. 2016. Efficient and Exact Local Search for Random Walk Based Top-K Proximity Query in Large Graphs. *IEEE Transactions on Knowledge and Data Engineering* 28, 5 (2016), 1160–1174.

Sensitivity Analysis of Centralities on Unweighted Networks

Shogo Murai*
The University of Tokyo
Bunkyo-ku, Tokyo, Japan
s.murai@is.s.u-tokyo.ac.jp

Yuichi Yoshida†
National Institute of Informatics
Chiyoda-ku, Tokyo, Japan
yyoshida@nii.ac.jp

ABSTRACT

Revealing important vertices is a fundamental task in network analysis. As such, many indicators have been proposed for doing so, which are collectively called centralities. However, the abundance of studies on centralities blurs their differences.

In this work, we compare centralities based on their sensivitity to modifications in the graph. Specifically, we introduce a quantitative measure called (average-case) edge sensitivity, which measures how much the centrality value of a uniformly chosen vertex (or an edge) changes when we remove a uniformly chosen edge. Edge sensitivity is applicable to unweighted graphs, regarding which, to our knowledge, there has been no theoretical analysis of the centralities. We conducted a theoretical analysis of the edge sensitivities of six major centralities: the closeness centrality, harmonic centrality, betweenness centrality, endpoint betweenness centrality, PageRank, and spanning tree centrality. Our experimental results on synthetic and real graphs confirm the tendency predicted by the theoretical analysis. We also discuss an extension of edge sensitivity to the setting that we remove a uniformly chosen set of edges of size k for an integer $k \geq 1$.

CCS CONCEPTS

• **Information systems** → **Web mining**; • **Theory of computation** → *Graph algorithms analysis.*

KEYWORDS

Centrality, sensitivity, edge deletion

ACM Reference Format:
Shogo Murai and Yuichi Yoshida. 2019. Sensitivity Analysis of Centralities on Unweighted Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313422

1 INTRODUCTION

One of the most important problems in network analysis is revealing important vertices in a given network. To solve this problem, several measures of the importance of a vertex have been proposed, and these are collectively called *(network) centralities*. For example, the closeness centrality [4, 5, 27] and harmonic centrality [21] measure the importance of a vertex based on its distance to other vertices. The betweenness centrality [14] considers the number of shortest paths that pass through the vertex. PageRank [25] regards a vertex as important if a random walk on the network has a high staying probability at the vertex.

Table 1: Average-case edge (k-)sensitivity of centralities. Here, n is the number of vertices, m is the number of edges, D is the diameter of the graph, α is the decaying factor for PageRank, b is the number of bridges, and χ_k is the probability that the graph becomes disconnected by removing a uniformly chosen set of k edges.

Centrality	Upper bound ($k = 1$)	Lower bound ($k = 1$)	Upper Bound (general)
Closeness	$O\left(\frac{b+D}{m}\right)$	$\Omega\left(\frac{b+D}{m}\right)$	$O\left(\frac{k^2D}{m} + \chi_k\right)$
Harmonic	$O\left(\frac{D}{m}\right)$	$\Omega\left(\frac{D}{m}\right)$	$O\left(\frac{kD}{m}\right)$
Betweenness	—	∞	—
Endpoint betweenness	$O\left(\frac{D^2}{m}\right)$	$\Omega\left(\frac{D^2}{m}\right)$	$O\left(\frac{k^2D^2}{m}\right)$
PageRank	$O\left(\frac{\alpha}{m(1-\alpha)^2}\right)$	$\Omega\left(\frac{\alpha}{m(1-\alpha)}\right)$	$O\left(\frac{k\alpha}{m(1-\alpha)^2}\right)$
Spanning tree	$O\left(\frac{\sqrt{n}}{m}\right)$	$\Omega\left(\frac{1}{m}\right)$	$O\left(k\sqrt{\frac{kn}{m(m-k)}}\right)$

However, the abundance of studies on centralities raises another question: *Which centrality should we use?* One consideration in response to this question is that centralities have been used to analyze real-world networks and we may miss some edges when constructing network datasets. This motivates us to compare centralities based on their sensitivity to changes in the network.

Although some previous works have analyzed the sensitivity of centralities, it remains difficult to distinguish between centralities because only empirical results have been presented with respect to particular networks [7, 10, 13, 24, 34]. There are only a few theoretical works on the sensitivity of centralities, and most of them considered the sensitivity of a specific centrality and do not compare different centralities in terms of sensitivity [15, 26, 31]. To the best of our knowledge, the only theoretical work comparing the sensitivity of several centralities is by Segarra and Ribeiro [29], who analyzed the sensitivity of centralities to small changes in the weight of edges. However, many real-world networks are essentially unweighted. For such networks, their measure may not characterize the sensitivity well, because theirs necessarily considers *continuous* changes of the edge weights, which are not apparent for such networks.

Our contributions. In this work, we propose a novel quantitative measure for the sensitivity of centralities to edge removals, called

*Supported by JST ERATO Grant Number JPMJER1305
†Supported by JSPS KAKENHI Grant Number JP17H04676.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313422

(average-case) edge sensitivity. Intuitively speaking, it measures how the centrality of a uniformly chosen vertex changes by removing a uniformly chosen edge. Not only is the proposed measure intuitive, as we will see soon, it also provides for a finer comparison of centralities than the sensitivity measure studied in [29].

We conducted a theoretical analysis of the edge sensitivities of six major centralities: the closeness centrality [4, 5, 27], harmonic centrality [21], betweenness centrality [14], endpoint betweenness centrality [9], PageRank [25], and spanning tree centrality [30]. An overview of our results is shown in Table 1, where the lower bounds indicate sensitivities of worst-case graphs. Our upper bounds show that, for a typical case of $D \approx 1/(1-\alpha) \ll b \approx n$, the harmonic centrality is the most stable (i.e., the least sensitive). Although the closeness centrality is very sensitive, it can be as stable as the harmonic centrality when the graph has only a small number of bridges. Our lower bounds show that our upper bounds are almost tight except for spanning tree centrality.

In practice, it is likely that we miss more than one edge when constructing network datasets and the edge sensitivity introduced above may not appropriately reflect such situations. To address this issue, for an integer $k \geq 1$, we introduce *(average-case) k-edge sensitivity*, which matches the previous edge sensitivity when $k = 1$. Intuitively speaking, it measures how the centrality of a uniformly chosen vertex changes by removing a uniformly chosen set of edges of size k. An overview of our results is shown in the last column of Table 1, which shows a similar tendency to the case of edge sensitivity. Note that $\chi_1 = b/m$. While Table 1 contains the lower bounds for sensitivity (in the worst case graph), we omitted the proof of these bounds from this paper due to the limitation of space.

Finally, we computed the (k-)edge sensitivities of synthetic and real-world graphs and obtained experimental results that confirm our theoretical bounds—expect for the spanning tree centrality, which was significantly less sensitive than predicted by our theoretical upper bound. Also, our experimental results suggest that edge sensitivity can be used as a good proxy for k-edge sensitivity when comparing centralities in terms of sensitivity against removals of multiple edges, which is desirable because computing the former is computationally less expensive than computing the latter.

Organization. The rest of this paper is organized as follows. In Section 2, we review previous works on the sensitivity of centralities. In Section 3, we introduce the notations used throughout this paper and definitions for the centralities we analyze. We define edge sensitivity in Section 4 and theoretically bound the edge sensitivities of those centralities in Section 5. In Section 6, we introduce and study k-edge sensitivity. Finally, Section 7 shows our experimental results, which confirm our theoretical analysis.

2 RELATED WORK

In this section, we review previous works pertaining to the sensitivity of centralities.

Most of the following works are empirical, however. In [7], the sensitivity of various centralities was experimentally analyzed using synthetic graphs, several robustness measures (such as the recall of vertices with the highest centralities), and several types of modifications to the graph (such as edge removal). Costenbader and Valente [10] studied the sensitivity of various centralities on graphs constructed from a larger graph by sampling vertices. Frantz *et al.* [13] conducted an empirical investigation of the relationship between the network topology and the sensitivity of centralities.

Sensitivity has been also studied from theoretical aspects. Ghosal and Ghosal and Barabási [15] theoretically analyzed the sensitivity of the ranking by PageRank on networks of several models. Platig et al. [26] and Tsugawa and Ohsaki [31] evaluated the sensitivity of the degree centrality. However, there are only a few theoretical works that compare sensitivities of centralities. Ng et al. [23] compares sensitivity of two eigenvector-based centralities (HITS [18] and PageRank [25]). Ufimtsev et al. [32] qualitatively disucussed how the centrality ranking is affected by the graph structure. Recently, Segarra and Ribeiro [29] introduced a sensitivity measure of a centrality for weighted networks. Their measure classifies centralities into three categories: *stable*, *continuous*, and *non-continuous*. As such, their measure is not useful for comparing centralities quantitatively.

3 PRELIMINARIES

In what follows, n and m denote the number of vertices and edges, respectively, of the graph we are concerned with.

Let $G = (V, E)$ be an undirected or directed graph. For an edge $e \in E$, we denote by $G \setminus e$ the subgraph $(V, E \setminus \{e\})$. Similarly, for a subset of edges $F \subseteq F$, $G \setminus F$ denotes the subgraph $(V, E \setminus F)$. For vertices $u, v \in V$, let $d_G(u, v)$ be the (geodesic) distance from u to v, that is, the length of a shortest path from u to v. If v is unreachable from u, we define $d_G(u, v) = \infty$. The diameter of G, denoted by D_G, is defined as $\max_{u,v \in V} d_G(u, v)$. Let $A_G \in \{0, 1\}^{n \times n}$ be the adjacency matrix of G. That is, $A_G(u, v) = 1$ if and only if $uv \in E$.

Let $G = (V, E)$ be a directed graph on n vertices. We denote the out-degree of a vertex v in G by $\delta_G(v)$. Let $\Delta_G \in \mathbb{R}^{n \times n}$ be a diagonal matrix whose diagonal element $\Delta_G(v, v)$ equals $\delta_G(v)$. Then, the row-normalized adjacency matrix $\bar{A}_G \in \mathbb{R}^{n \times n}$ is defined as $\bar{A}_G = {\Delta_G}^{-1} A_G$. If a vertex $u \in V$ has no outgoing edges, we let $\bar{A}_G(u, v) = 0$ for every $v \in V$.

If the graph G is clear from the context, we may omit G in the above-mentioned notations.

3.1 Centralities

In this paper, we discuss four vertex centralities, namely, the closeness, harmonic, and betweenness centralities, along with PageRank. Furthermore, we discuss an edge centrality, namely, the spanning tree centrality. Here, we present the definitions for these centralities.

Definition 3.1 (Closeness centrality [4, 5, 27]). *For a graph $G = (V, E)$, the* closeness centrality *of a vertex $u \in V$ is defined as*

$$\mathrm{CC}_G(u) = \frac{n-1}{\sum_{v \in V} d_G(u, v)}.$$

If G is disconnected, $\sum_{v \in V} d_G(u, v) = \infty$ and thus we let $\mathrm{CC}_G(u) = 0$.

Definition 3.2 (Harmonic centrality [21]). *For a graph $G = (V, E)$, the* harmonic centrality *of a vertex $u \in V$ is defined as*

$$\mathrm{HC}_G(u) = \sum_{v \in V, v \neq u} \frac{1}{d_G(u, v)}.$$

It should be noted that $\mathrm{HC}_G(u)$ *is well-defined even if* G *is not connected, because we have* $\frac{1}{d_G(u,v)} = 0$ *if* v *is unreachable from* u.

Definition 3.3 (Betweenness centrality [14]). *For a graph* $G = (V, E)$*, the* betweenness centrality *of a vertex* $u \in V$ *is defined as*

$$\mathrm{BT}_G(u) = \sum_{(s,t)\in V^2, s\neq u\neq t} \frac{\sigma_{G,st}(u)}{\sigma_{G,st}},$$

where $\sigma_{G,st}$ *is the number of shortest paths from* s *to* t *in* G *and* $\sigma_{G,st}(u)$ *is the number of such paths that pass through* u.

In this definition, $\sigma_{G,st}$ can be 0 if s and t are disconnected in G (then, $\sigma_{G,st}(u)$ is also 0 for every $u \in V$). We simply let $\frac{\sigma_{G,st}(u)}{\sigma_{G,st}} = 0$ if there is no path from s to t in G. This corresponds to taking the sum over pairs (s, t) such that s and t are connected.

There is a variant of betweenness centrality that counts the *shortest paths having* v *as an endpoint.*

Definition 3.4 (Endpoint betweenness [9]). *For a graph* $G = (V, E)$*, the* endpoint betweenness centrality *of a vertex* $u \in V$ *is defined as*

$$\mathrm{BT}'_G(u) = \sum_{(s,t)\in V^2} \frac{\sigma_{G,st}(u)}{\sigma_{G,st}}.$$

We can see that BT and BT′ are closely tied:

Proposition 3.5. *When* G *is connected,* $\mathrm{BT}'_G(v) = \mathrm{BT}_G(v) + 2n - 1$ *holds for every* $v \in V$.

Definition 3.6 (PageRank [25]). *Let* $G = (V, E)$ *be a directed graph. Let* $0 < \alpha < 1$ *be a decaying factor and* $\boldsymbol{\pi} \in \mathbb{R}^n_{\geq 0}$ *be an initial distribution such that* $\|\boldsymbol{\pi}\|_1 = 1$*. Then, the PageRank vector* $\mathrm{PR}_G \in \mathbb{R}^n_{\geq 0}$ *of* G *is defined as*

$$\mathrm{PR}_G = (1-\alpha)(I - \alpha \bar{A}_G^{\top})^{-1}\boldsymbol{\pi},$$

and the PageRank of a vertex u *is* $\mathrm{PR}_G(u)$.

As in [25], we use the uniform distribution as the initial distribution. That is, $\boldsymbol{\pi} = \frac{1}{n}\mathbf{1}$, where $\mathbf{1} \in \mathbb{R}^n$ is the all-one vector.

The four centralities above are all defined for vertices. We next present the definition for the spanning tree centrality, which is defined for edges.

Definition 3.7 (Spanning tree centrality [30]). *For a connected graph* $G = (V, E)$*, the* spanning tree centrality *of an edge* $e \in E$ *is defined as*

$$\mathrm{ST}_G(e) = \frac{\tau_G(e)}{\tau_G}$$

where τ_G *is the number of spanning trees in* G *and* $\tau_G(e)$ *is the number of spanning trees including* e.

Note that this definition can be extended to disconnected graphs by considering spanning forests instead of spanning trees.

4 AVERAGE-CASE EDGE SENSITIVITY

In this section, we introduce sensitivity notions regarding centralities.

4.1 Sensitivity of vertex centralities

We first consider the sensitivity of vertex centralities such as closeness and harmonic centralities.

Before providing the definition, we outline some requirements that the sensitivity notion must satisfy. First, it should be invariant under the scaling of centralities. Note that different centralities may have different scales: the closeness centrality of a vertex takes is value within $[0, 1]$, whereas the harmonic centrality of a vertex is at least 1 (unless the vertex is isolated). Moreover, there are some variations to the definition of a centrality. For example, the definition of the closeness centrality in [6] differs from ours by a factor of $n - 1$. Because we want to compare different centralities, their sensitivities should be invariant under scaling.

Second, we want to understand how the structure of the underlying network affects the sensitivity. Hence, the notion of sensitivity should be parameterized by the underlying network.

To describe our sensitivity measure, we first state some definitions. A vertex centrality can be regarded as a function that, given a graph $G = (V, E)$ and a vertex $v \in V$, computes the centrality value of v in G. With this observation, we say that a family of functions $C = \{C_G : V(G) \to \mathbb{R}_{\geq 0}\}_G$ indexed by a graph G is a *vertex centrality function.* For example, CC, HC, BT, and PR introduced in Section 3 are all centrality functions.

Definition 4.1 (Average-case sensitivity for vertex centralities). *For a graph* $G = (V, E)$ *and a vertex centrality function* $C = \{C_G\}_G$*, the* (average-case) edge sensitivity *of* C *on* G *is defined as*

$$S^v_C[G] = \frac{1}{|V||E|}\sum_{e\in E}\sum_{v\in V}\frac{\left|C_{G\setminus e}(v) - C_G(v)\right|}{C_G(v)}$$

In other words, the edge sensitivity is the expected value of the relative change of the centrality value of a uniformly chosen vertex when a uniformly chosen edge is removed. The smaller the edge sensitivity of a centrality is, the less sensitive its centrality is to changes in the network.

As explained above, the edge sensitivity is invariant under scaling:

Theorem 4.2. *Let* C *be a centrality function and* C' *be another centrality function, such that* $C'_G(u) = \lambda(G)C_G(u)$*, where* $\lambda(G) > 0$ *is a scaling factor that may depend on* G*. Then, we have* $S^v_C[G] = S^v_{C'}[G]$ *for any graph* G.

4.2 Sensitivity of edge centralities

Next, we extend the proposed edge sensitivity from vertex centralities to edge centralities, such as the spanning tree centrality.

As in the case of vertex centralities, we define the general notion of an edge centrality: We say that a family of functions $C = \{C_G : E(G) \to \mathbb{R}_{\geq 0}\}_G$ indexed by a graph G is an *edge centrality function.*

Definition 4.3 (Average-case sensitivity for edge centralities). *For a graph* $G = (V, E)$ *and an edge centrality function* $C = \{C_G\}_G$*, the edge sensitivity of* C *on* G *is defined as*

$$S^e_C[G] = \frac{1}{|E|(|E|-1)}\sum_{f\in E}\sum_{e\in E\setminus\{e\}}\frac{\left|C_{G\setminus f}(e) - C_G(e)\right|}{C_G(e)}$$

This sensitivity is also invariant under scaling per Theorem 4.2.

The distinction between $S_C^v[G]$ and $S_C^e[G]$ is not essential, because they are defined for different families of centrality functions. Thus, $S_C^v[G]$ and $S_C^e[G]$ are hereafter written as $S_C[G]$.

5 THEORETICAL ANALYSIS

In this section, we theoretically analyze the edge sensitivities of the centralities introduced in Section 3. The proofs of our lower bounds are deferred to the full version due to space limitation. Then, we compare those centralities in terms of edge sensitivity in Section 5.6.

5.1 Closeness centrality

As mentioned above, the closeness centrality of a vertex in a disconnected graph is 0, and hence its edge sensitivity is not well defined against disconnected graphs. For connected graphs, we have the following bound:

Theorem 5.1. *Let G be a connected undirected graph with at least 2 vertices and b bridges. Then, we have $\frac{b}{m} \le S_{\mathrm{CC}}[G] \le \frac{2D_G+b}{m}$.*

To prove this theorem, we use the following lemma:

Lemma 5.2 ([28]). *Let $G = (V, E)$ be an undirected graph and $F \subseteq E$. If $G \setminus F$ is connected, then $D_{G\setminus F} \le (|F|+1)D_G$ holds.*

Proof of Theorem 5.1. Let $B \subseteq E$ be the set of bridges in $G = (V, E)$. Since removing a bridge makes the graph disconnected, we have $\frac{|C_{G\setminus e}(v)-C_G(v)|}{C_G(v)} = 1$ if $e \in B$. Then, we have

$$S_{\mathrm{CC}}[G] \ge \frac{1}{|V||E|} \sum_{e\in B}\sum_{v\in V} \frac{\left|\mathrm{CC}_{G\setminus e}(v) - \mathrm{CC}_G(v)\right|}{\mathrm{CC}_G(v)} = \frac{b}{m}.$$

Let $W_G(u) = \sum_{v\in V} d(u,v)$. Then, $\mathrm{CC}_G(u) = \frac{n-1}{W_G(u)}$ holds. It is also easy to see that $W_G(u) \le W_{G\setminus e}(u)$ for any edge $e \in E$. Therefore, we have

$$\frac{\left|\mathrm{CC}_{G\setminus e}(v) - \mathrm{CC}_G(v)\right|}{\mathrm{CC}_G(v)} = \frac{W_{G\setminus e}(v) - W_G(v)}{W_{G\setminus e}(v)} \le \frac{W_{G\setminus e}(v) - W_G(v)}{W_G(v)}.$$

Using this, we obtain

$$\sum_{e\in E\setminus B} \frac{\left|\mathrm{CC}_{G\setminus e}(v) - \mathrm{CC}_G(v)\right|}{\mathrm{CC}_G(v)} \le \frac{1}{W_G(v)} \sum_{e\in E\setminus B} (W_{G\setminus e}(v) - W_G(v))$$
$$\le \frac{1}{W_G(v)} \sum_{w\in V}\sum_{e\in E\setminus B} (d_{G\setminus e}(v,w) - d_G(v,w)). \quad (1)$$

For $v, w \in V$, we consider a shortest path $p(v,w)$ connecting v and w in G. Note that $p(v,w)$ contains $d_G(v,w)$ edges. If an edge $e \in E\setminus B$ is not a member of $p(v,w)$, we have $d_{G\setminus e}(v,w) = d_G(v,w)$. Otherwise, we have $d_{G\setminus e}(v,w) \le 2D_G$ according to Lemma 5.2. Therefore,

$$\sum_{w\in V}\sum_{e\in E\setminus B} (d_{G\setminus e}(v,w) - d_G(v,w))$$
$$= \sum_{w\in V}\sum_{e\in (E\setminus B)\cap p(v,w)} (d_{G\setminus e}(v,w) - d_G(v,w))$$
$$\le \sum_{w\in V} 2D_G d_G(v,w) = 2W_G(v)D_G. \quad (2)$$

Using (1), (2), and the fact that $\frac{|C_{G\setminus e}(v)-C_G(v)|}{C_G(v)} = 1$ for $e \in B$, we obtain $S_{\mathrm{CC}}[G] \le \frac{2D_G+b}{m}$, as desired. □

5.2 Harmonic centrality

Theorem 5.3. *Let G be an undirected or directed graph. Then, we have $S_{\mathrm{HC}}[G] \le \frac{n-1}{nm} \sum_{v\in V} \frac{1}{\mathrm{HC}[G](v)}$.*

Proof. Analogous to the proof for the closeness centrality, we have for any vertex $v \in V(G)$,

$$\sum_{e\in E} \frac{\left|\mathrm{HC}_{G\setminus e}(v) - \mathrm{HC}_G(v)\right|}{\mathrm{HC}_G(v)} = \sum_{e\in E} \frac{\mathrm{HC}_G(v) - \mathrm{HC}_{G\setminus e}(v)}{\mathrm{HC}_G(v)}$$
$$= \frac{1}{\mathrm{HC}_G(v)} \sum_{e\in E}\sum_{w\in V, w\ne v} \left(\frac{1}{d_G(v,w)} - \frac{1}{d_{G\setminus e}(v,w)}\right)$$
$$= \frac{1}{\mathrm{HC}_G(v)} \sum_{w\in V, w\ne v}\sum_{e\in p(v,w)} \frac{1}{d_G(v,w)} \le \frac{n-1}{\mathrm{HC}_G(v)}.$$

Note that the inequality holds for both undirected and directed graphs. It follows that $S_{\mathrm{HC}}[G] \le \frac{n-1}{nm} \sum_{v\in V} \frac{1}{\mathrm{HC}[G](v)}$ holds. □

Corollary 5.4. *Let G be a (strongly) connected graph. Then, we have $S_{\mathrm{HC}}[G] \le \frac{\bar{d}_G}{m} \le \frac{D_G}{m}$, where $\bar{d}_G$ is the average distance between vertices: $\bar{d}_G = \frac{1}{n(n-1)} \sum_{u,v\in V, u\ne v} d(u,v)$.*

Proof. We have $\frac{n-1}{\mathrm{HC}[G](v)} \le \frac{1}{n-1} \sum_{w\ne v} d(v,w)$ by the HM-AM inequality. It follows that $\frac{n-1}{n} \sum_{v\in V} \frac{1}{\mathrm{HC}[G](v)} \le \bar{d}_G$. Combining this with Theorem 5.3, we obtain $S_{\mathrm{HC}}[G] \le \frac{\bar{d}_G}{m}$. □

5.3 Betweenness centrality

Our definition of edge sensitivity cannot be directly applied to the betweeness centrality because $\mathrm{BT}_G(v) = 0$ and $\mathrm{BT}_{G\setminus e}(v) > 0$ could hold at the same time, resulting in a zero-division problem. More specifically, consider a triangle graph consisting of three vertices $\{v, x, y\}$ and let $e = (x, y)$. Then, $\mathrm{BT}_G(v) = 0$ holds, whereas $\mathrm{BT}_{G\setminus e}(v) > 0$ because the only path between x and y in $G\setminus e$ passes through v.

Another issue with the betweenness centrality is its instability. Consider a complete graph K_n on n vertices. It is clear that $\mathrm{BT}_{K_n}(v) = 0$ for every v and $\mathrm{BT}_{K_n\setminus xy}(v) > 0$ for $v \ne x, y$. That is, removing an edge renders the centralities of $n-2$ vertices nonzero even though they were all zero before its removal.

However, we can give non-trivial bounds on the edge sensitivity of endpoint betweeness centrality:

Theorem 5.5. *Let G be a connected undirected graph. Then, we have $S_{\mathrm{BT}'}[G] \le \frac{{D_G}^2+3D_G}{m}$.*

Although we consider endpoint betweenness centrality rather than (normal) betweenness centrality, their values are relatively close for highly important vertices, since the average betweenness centrality is $\Omega(n)$. Therefore, in practice, we can suppose that analysis on endpoint betweenness centrality will provide useful results for betweenness centrality as well.

For a graph G, we denote the number of paths of length d between two vertices s and t in G by $\sigma_{G,st}^d$. Similarly, the number of such

paths passing through a vertex v is denoted by $\sigma^d_{G,st}(v)$. Before proving this theorem, we need the following technical lemma:

Lemma 5.6. *If s and t are connected in G, we have*

(1) $\sum_{v\in V}\sigma_{G,st}(v) = (d_G(s,t)+1)\sigma_{G,st}$,
(2) $\sum_{e\in E}(\sigma^{d_G(s,t)}_{G,st}(v) - \sigma^{d_G(s,t)}_{G\setminus e,st}(v)) = d_G(s,t)\sigma^{d_G(s,t)}_{G,st}(v)$.

Proof. (1) Both sides of the equation count the number of pairs (p,v), where p is a shortest path between s and t, and v is a vertex that p passes through.
(2) Note that $\sigma^{d_G(s,t)}_{G,st}(v) - \sigma^{d_G(s,t)}_{G\setminus e,st}(v)$ is the number of paths of length $d_G(s,t)$ between s and t passing through both v and e. Both sides of the equation count the number of pairs (p,e), where p is a path of length $d_G(s,t)$ between s and t, and e is an edge that p passes through. □

Here we define $d'_G(s,t)$ by $d'_G(s,t) = \begin{cases} d_G(s,t) & d_G(s,t) < \infty \\ -1 & \text{otherwise} \end{cases}$. Then, (1) of Lemma 5.6 can be generalized to graphs G in which s and t may not be connected: $\sum_{v\in V}\frac{\sigma_{G,st}(v)}{\sigma_{G,st}} = d'_G(s,t)+1$.

Proof of Theorem 5.5. For every pair of vertices $(s,t)\in V^2$, we let $P(s,t) = \left\{e\in E \mid d_G(s,t) = d_{G\setminus e}(s,t)\right\}$. To bound $S_{\mathrm{BT}'}[G]$, we first consider the *sum of absolute differences*:

$$A = \frac{1}{nm}\sum_{v\in V}\sum_{e\in E}\left|\mathrm{BT}'_{G\setminus e}(v) - \mathrm{BT}'_G(v)\right|.$$

Using the triangle inequality, we have

$$A \le \frac{1}{nm}\sum_{(s,t)\in V^2}\sum_{e\in E}\sum_{v\in V}\left|\frac{\sigma_{G\setminus e,st}(v)}{\sigma_{G\setminus e,st}} - \frac{\sigma_{G,st}(v)}{\sigma_{G,st}}\right|.$$

From Lemma 5.6, we have

$$\begin{aligned}
&\sum_{e\in E}\sum_{v\in V}\left|\frac{\sigma_{G\setminus e,st}(v)}{\sigma_{G\setminus e,st}} - \frac{\sigma_{G,st}(v)}{\sigma_{G,st}}\right| \\
&= \sum_{e\in E}\sum_{v\in V}\left|\frac{\sigma_{G\setminus e,st}(v)}{\sigma_{G\setminus e,st}} - \frac{\sigma^{d_G(s,t)}_{G\setminus e,st}(v)}{\sigma_{G,st}} + \frac{\sigma^{d_G(s,t)}_{G\setminus e,st}(v)}{\sigma_{G,st}} - \frac{\sigma_{G,st}(v)}{\sigma_{G,st}}\right| \\
&\le 2\cdot\sum_{e\in E}\sum_{v\in V}\left(\frac{\sigma_{G,st}(v)}{\sigma_{G,st}} - \frac{\sigma^{d_G(s,t)}_{G\setminus e,st}(v)}{\sigma_{G,st}}\right) \\
&\quad + \sum_{e\in P(s,t)}\sum_{v\in V}\left|\frac{\sigma_{G\setminus e,st}(v)}{\sigma_{G\setminus e,st}} - \frac{\sigma_{G,st}(v)}{\sigma_{G,st}}\right| \\
&\le 2D_G(D_G+1) + \sum_{e\in P(s,t)}(d'_{G\setminus e}(s,t) - d'_G(s,t)) \le 4{D_G}^2 + 2D_G.
\end{aligned}$$

Therefore, we obtain

$$\begin{aligned}
A &= \frac{1}{nm}\sum_{(s,t)\in V^2, s\neq t}\sum_{e\in E}\sum_{v\in V}\left|\frac{\sigma_{G\setminus e,st}(v)}{\sigma_{G\setminus e,st}} - \frac{\sigma_{G,st}(v)}{\sigma_{G,st}}\right| \\
&\le \frac{n-1}{m}\cdot(4{D_G}^2 + 2D_G).
\end{aligned}$$

Furthermore, $\mathrm{BT}'_G(u) \ge 2n-1$ follows from Proposition 3.5. Accordingly, we have $S_{\mathrm{BT}'}[G] \le \frac{A}{2n-1} \le \frac{2{D_G}^2+D_G}{m}$. □

5.4 PageRank

To analyze the edge sensitivity of PageRank, we use a random-walk-based representation of PageRank [3]. For a walk $w = w_0w_1\dots w_{\ell(w)}$ of length $\ell(w)$ in a graph G of n vertices and a matrix $M\in\mathbb{R}^{n\times n}$, we define the *weight of w with respect to M* by

$$W_M(w) = \prod_{i=0}^{\ell-1} M(w_i, w_{i+1}).$$

Lemma 5.7. *The Pagerank of a vertex $v\in V(G)$ satisfies*

$$\mathrm{PR}_G(v) = \sum_{w\in\mathrm{walks}_G(\star,v)}\frac{1-\alpha}{n}W_{\alpha\bar{A}_G}(w),$$

where $\mathrm{walks}_G(\star,v)$ *is the set of walks in G terminating at v.*

The following is almost immediate from the definition and we omit the proof:

Lemma 5.8. *Let* walks_G *be the (infinite) set of walks in G. Then, we have $\sum_{w\in\mathrm{walks}_G,\ell(w)=l}\frac{1-\alpha}{n}W_{\alpha\bar{A}_G}(w) \le (1-\alpha)\alpha^l$. Equality holds if every vertex in G has at least one out-going edge.*

We first bound $S_{\mathrm{PR}}[G]$ exclusively for special graphs.

Lemma 5.9. *Suppose that every vertex $v\in V$ satisfies $\delta_G(v)\ge 1$. Then, we have $S_{\mathrm{PR}}[G] \le \frac{2\alpha}{m(1-\alpha)^2}$.*

Proof. Let $e = st\in E$ be an edge to be removed, and let $M = \alpha\bar{A}_G$, $M'' = \alpha\bar{A}_{G\setminus e}$ and $M'\in\mathbb{R}^{n\times n}$ be the matrix obtained from M by replacing $M(s,t)$ with 0. Then, it is clear that $M\ge M'\le M''$ and thus $W_M(w)\ge W_{M'}(w)\le W_{M''}(w)$ holds for every walk w. Additionally, since every vertex in G has at least one out-going edge, we have $\sum_{v\in V}\mathrm{PR}_G(v) = 1 \ge \sum_{v\in V}\mathrm{PR}_{G\setminus e}(v)$ from Lemma 5.8. Therefore, we have

$$\begin{aligned}
&\sum_{v\in V}\left|\mathrm{PR}_G(v) - \mathrm{PR}_{G\setminus e}(v)\right| \le \frac{1-\alpha}{n}\sum_{w\in\mathrm{walks}_G(\star,v)}|W_M(w) - W_{M''}(w)| \\
&\le \frac{1-\alpha}{n}\sum_{w\in\mathrm{walks}_G(\star,v)}((W_M(w) - W_{M'}(w)) + (W_{M''}(w) - W_{M'}(w))) \\
&\le \frac{2(1-\alpha)}{n}\sum_{w\in\mathrm{walks}_G(\star,v)}(W_M(w) - W_{M'}(w)) \\
&\le \frac{2(1-\alpha)}{n}\sum_{w\in\mathrm{walks}_G(\star,v),e\in w}W_M(w).
\end{aligned}$$

From Lemma 5.8, we obtain

$$\begin{aligned}
\sum_{e\in E}\sum_{v\in V}\left|\mathrm{PR}_G(v) - \mathrm{PR}_{G\setminus e}(v)\right| &\le \frac{2(1-\alpha)}{n}\sum_{e\in E}\sum_{w\in\mathrm{walks}_G:\atop e\in w}W_M(w) \\
&\le \frac{2(1-\alpha)}{n}\sum_{w\in\mathrm{walks}_G}W_M(w)\ell(w) \le 2\sum_{\ell=0}^{\infty}\ell(1-\alpha)\alpha^\ell = \frac{2\alpha}{1-\alpha}.
\end{aligned}$$

Here note that $\mathrm{PR}(v)\ge\frac{1-\alpha}{m}$ holds because there is a walk of length 0 starting and ending at v for every $v\in V$. Finally we have

$$\begin{aligned}
S_{\mathrm{PR}}[G] &= \frac{1}{nm}\sum_{e\in E}\sum_{v\in V}\frac{\left|\mathrm{PR}_{G\setminus e}(v) - \mathrm{PR}_G(v)\right|}{\mathrm{PR}_G(v)} \\
&\le \frac{1}{nm}\cdot\frac{n}{1-\alpha}\sum_{e\in E}\sum_{v\in V}\left|\mathrm{PR}_{G\setminus e}(v) - \mathrm{PR}_G(v)\right| \le \frac{2\alpha}{m(1-\alpha)^2}.
\end{aligned}$$ □

From Lemma 5.9, we prove the same inequality for a general G:

Theorem 5.10. *We have* $S_{\mathrm{PR}}[G] \le \frac{2\alpha}{m(1-\alpha)^2}$.

Proof. Let $T = \{v \in V \mid \delta_G(v) = 0\}$, $F = \{(t,t) \mid t \in T\}$ and $G' = (V, E \cup F)$. We can see that $\mathrm{PR}_{G'}(v) = \mathrm{PR}_G(v)$ if $\delta_G(v) \ge 1$ and $\mathrm{PR}_{G'}(v) = \frac{\mathrm{PR}_G(v)}{1-\alpha}$ if $\delta_G(v) = 0$. Therefore, we have $\frac{\left|\mathrm{PR}_{G'\setminus e}(v) - \mathrm{PR}_{G'}(v)\right|}{\mathrm{PR}_{G'}(v)} = \frac{\left|\mathrm{PR}_{G\setminus e}(v) - \mathrm{PR}_G(v)\right|}{\mathrm{PR}_G(v)}$ for every $e \in E$ and $v \in V$. Accordingly from Lemma 5.9, we obtain

$$\begin{aligned} S_{\mathrm{PR}}[G] &= \frac{1}{nm} \sum_{e\in E} \sum_{v\in V} \frac{\left|\mathrm{PR}_{G\setminus e}(v) - \mathrm{PR}_G(v)\right|}{\mathrm{PR}_G(v)} \\ &= \frac{1}{nm} \sum_{e\in E} \sum_{v\in V} \frac{\left|\mathrm{PR}_{G'\setminus e}(v) - \mathrm{PR}_{G'}(v)\right|}{\mathrm{PR}_{G'}(v)} \\ &\le \frac{|E\cup F|}{|E|} S_{\mathrm{PR}}[G'] \le \frac{2\alpha}{m(1-\alpha)^2}. \end{aligned}$$

□

5.5 Spanning tree centrality

To analyze the edge sensitivity of the spanning tree centrality, we use the relationship between the spanning tree centrality and *effective resistance*, which is defined as follows:

Definition 5.11 (Effective resistance). *For an undirected graph $G = (V,E)$, we imagine an electric circuit obtained by replacing every edge in G with a resistor of a unit resistance. For every pair of vertices $(u,v) \in V^2$, the* effective resistance *between u and v is defined as the difference in their electric potential when we inject a unit electricity from u to v.*

Theorem 5.12 (see, e.g., [22]). *Let $G = (V,E)$ be an undirected graph. For every edge $e = uv \in E$, $\mathrm{ST}_G(e)$ is the effective resistance between u and v in G.*

The following pertains to effective resistance:

Theorem 5.13 (Rayleigh's monotonicity law [11]). *If the resistance of a resistor in a circuit increases, for every $(u,v) \in V^2$, the effective resistance between u and v does not decrease.*

Remark. *Theorem 5.13 applies to the removal of resistors, because the removal of a resistor corresponds to increasing its resistance to ∞.*

Theorem 5.14 (Dirichlet principle [11]). *For $v \in \mathbb{R}^V$, the energy consumption when the electrical potential of w is set to $v(w)$ for every $w \in V$ is $C(v) := \sum_{st\in E} (v(s) - v(t))^2$. Under the constraints of $v(x) = 1$ and $v(y) = 0$ $(x, y \in V)$, the energy $C(v)$ is minimized when the electrical flow determined by v satisfies Kirchoff's law. In other words, $C(v)$ is at least the reciprocal of the effective resistance between x and y.*

Based on these facts, we prove some technical lemmas that are useful for our analysis.

Lemma 5.15. *Let $G = (V,E)$ be an undirected graph. For every $e, f \in E$ $(e \ne f)$, we have $\mathrm{ST}_G(e) \le \mathrm{ST}_{G\setminus f}(e) \le \frac{\mathrm{ST}_G(e)}{1-\mathrm{ST}_G(e)}$.*

Proof. With Theorem 5.12, we can identify the spanning tree centrality of e with the effective resistance of $e \in E$. The first inequality immediately holds from Theorem 5.13.

We let $e = st$. Let $v \in \mathbb{R}^V$ be the electrical potential of every vertex when we apply a unit voltage between s and t in $G \setminus f$. From Theorem 5.14, the energy consumed when we apply a unit voltage between s and t in G is upper-bounded by the energy consumption when the electrical potential is given by v (note that this does not necessarily satisfy Kirchoff's law). Since this energy consumption is equivalent to the reciprocal of the effective resistance, we have $\frac{1}{\mathrm{ST}_G(e)} \le \frac{1}{\mathrm{ST}_{G\setminus f}(e)} + 1$. Then, it is clear that $\mathrm{ST}_{G\setminus f}(e) \le \frac{\mathrm{ST}_G(e)}{1-\mathrm{ST}_G(e)}$. □

Lemma 5.16. *Let $G = (V,E)$ be an undirected graph and $e, f \in E$ be distinct edges. Then,*

$$0 \le \frac{\mathrm{ST}_{G\setminus f}(e) - \mathrm{ST}_G(e)}{\mathrm{ST}_G(e)} \le \sqrt{2\left(\mathrm{ST}_{G\setminus f}(e) - \mathrm{ST}_G(e)\right)}.$$

Proof. The first inequality is immediate from Lemma 5.15.

Let $x = \mathrm{ST}_G(e)$ and $y = \mathrm{ST}_{G\setminus f}(e)$. To show the second inequality, it suffices to show $\frac{\sqrt{y-x}}{x} \le \sqrt{2}$. If $x \ge \frac{1}{2}$, we have $\frac{\sqrt{y-x}}{x} \le \sqrt{2}$ since $y \le 1$ (from the definition of a spanning tree centrality). If $x < \frac{1}{2}$, using Lemma 5.15, we have $\frac{\sqrt{y-x}}{x} \le \frac{1}{\sqrt{1-x}} \le \sqrt{2}$. In both cases, we obtain the desired inequality. □

Lemma 5.17. *Let $G = (V,E)$ be a connected undirected graph. Then, $\sum_{e\in E} \mathrm{ST}_G(e) = |V| - 1$.*

Proof. This follows immediately from $\sum_{e\in E} \tau_G(e) = (|V| - 1)\tau_G$, which holds because every spanning tree in G has $|V| - 1$ edges. □

Theorem 5.18. *Let G be a two-connected undirected graph. Then, we have $S_{\mathrm{ST}}[G] \le \sqrt{\frac{2(n-1)}{m(m-1)}}$.*

Proof. For $e, f \in E$ $(e \ne f)$, we let $\delta_G(e,f) = \mathrm{ST}_{G\setminus f}(e) - \mathrm{ST}_G(e)$. From Lemma 5.17, we have

$$\begin{aligned} \sum_{e,f\in E, e\ne f} \delta_G(e,f) &= \sum_{f\in E} \sum_{e\in E\setminus\{f\}} (\mathrm{ST}_{G\setminus f}(e) - \mathrm{ST}_G(e)) \\ &= \sum_{f\in E} \left(\sum_{e\in E\setminus\{f\}} \mathrm{ST}_{G\setminus f}(e) - \sum_{e\in E} \mathrm{ST}_G(e) + \mathrm{ST}_G(f) \right) \\ &= \sum_{f\in E} \mathrm{ST}_G(f) = n - 1. \end{aligned}$$

Additionally, from Lemma 5.16,

$$\sum_{e,f\in E, e\ne f} \frac{\mathrm{ST}_{G\setminus f}(e) - \mathrm{ST}_G(e)}{\mathrm{ST}_G(e)} \le \sum_{e,f\in E, e\ne f} \sqrt{2\delta_G(e,f)}.$$

Therefore, we have $\sum_{e,f\in E, e\ne f} \sqrt{2\delta_G(e,f)} \le \sqrt{2(n-1)m(m-1)}$ by the Cauchy-Schwarz inequality. Thus we have

$$S_{\mathrm{ST}}[G] = \frac{1}{m(m-1)} \sum_{e,f\in E, e\ne f} \frac{\mathrm{ST}_{G\setminus f}(e) - \mathrm{ST}_G(e)}{\mathrm{ST}_G(e)} \le \sqrt{\frac{2(n-1)}{m(m-1)}}$$

as desired. □

5.6 Comparison of centralities in terms of sensitivity

We turn now to a comparison of the centralities we have analyzed.

First, the edge sensitivities of the closeness and harmonic centralities are similar with regard to the diameter D_G and the number of edges m. However, the former degrades considerably when there are many bridges in the graph. Hence, the harmonic centrality is preferable to the closeness centrality unless the graph is highly connected.

In contrast, the edge sensitivity of endpoint betweenness centrality is $\Theta(D_G^2/m)$, which is conspiciously worse than that of the harmonic centrality.

The sensitivity of PageRank depends on the decaying factor α: the higher α is, the larger the sensitivity is. PageRank with larger value of α gives much weight on longer walks. Since longer walks are more likely to contain the randomly removed edge, PageRank can be more sensitive with larger α.

The edge sensitivities of the closeness, harmonic, and betweenness centralities depend on the diameter D_G, which is unavoidable according to our lower bounds (Table 1). By contrast, the edge sensitivity of PageRank depends on α (and m) rather than D_G. Although this implies that PageRank may be more stable than other centralities when the graph has a large diameter, the harmonic centrality is preferable for typical real small-world networks that have diameters of $O(\log n)$ [33].

Although our current bound on the edge sensitivity of spanning tree centrality is $O(\sqrt{n}/m)$, we cannot conclude that it is worse than other centralities, insofar as we do not have a matching lower bound.

6 AVERAGE-CASE k-EDGE SENSITIVITY

In this section, we discuss k-edge sensitivity, which is an extension of edge sensitivity for removing k edges.

6.1 Definition

Average-case edge sensitivity (Definition 4.1) considers the case that a (random) edge is missing in the observed graph. In practice, however, it is more likely that multiple edges are missing, and this motivates us to define the following *(average-case) k-edge sensitivity*:

Definition 6.1 (Average-case k-edge sensitivity for vertex centralities). *For a graph $G = (V, E)$ and a vertex centrality function $C = \{C_G\}_G$, the* k-edge sensitivity *of C on G is defined as*

$$S^v_{C,k}[G] = \frac{1}{|V|\binom{|E|}{k}} \sum_{F \subseteq E, |F|=k} \sum_{v \in V} \frac{\left|C_{G\setminus F}(v) - C_G(v)\right|}{C_G(v)}.$$

The k-edge sensitivity of an edge centrality $S^e_{C,k}[G]$ is defined analogously (note that the sum of the relative differences is taken over $v \in V \setminus F$). Again, we simply write $S^v_{C,k}[G]$ and $S^e_{C,k}[G]$ as $S_{C,k}[G]$. Note that $S_C[G] = S_{C,1}[G]$ holds and hence k-edge sensitivity is a natural extension of edge sensitivity. Also, it is notable that a k-edge sensitivity can be probabilistically defined:

$$S^v_{C,k}[G] = \mathrm{E}_F\left[\frac{1}{|V|}\sum_{v\in V}\frac{\left|C_{G\setminus F}(v) - C_G(v)\right|}{C_G(v)}\right],$$

where F is a uniformly chosen subset of E with cardinality k.

6.2 Theoretical analysis

In this section, we extend results in Section 5 to k-edge sensitivity.

Theorem 6.2. *Let G be a connected undirected graph. Then, $S_{\mathrm{CC},k}[G] \le \frac{k(k+1)D_G}{m} + \chi_k[G]$, where $\chi_k[G]$ is the probability that G becomes disconnected by removing a uniformly chosen set of k edges in G.*

Proof. We let $c(G,x) = x$ if G is connected and $c(G,x) = 0$ otherwise. For a fixed vertex $v \in V$,

$$\begin{aligned}
&E\left[c\left(G\setminus F, \sum_{w\in V}(d_{G\setminus F}(v,w) - d_G(v,w))\right)\right]\\
\le& E\left[c\left(G\setminus F, \sum_{w\in V, F\cap p(v,w)\neq\emptyset}(d_{G\setminus F}(v,w) - d_G(v,w))\right)\right]\\
\le& \sum_{w\in V} k(k+1)D_G d_G(v,w) = k(k+1)W_G(v)D_G
\end{aligned}$$

because F and $p(v,w)$ intersects with probability at most $\frac{kd_G(v,w)}{m}$, and if v and w are connected in $G \setminus F$, $d(v,w) \le (k+1)D_G$ follows from Lemma 5.2. The rest of the proof is similar to that of Theorem 5.1. □

Theorem 6.3. *Let G be a (strongly) connected graph. Then, we have $S_{\mathrm{HC},k}[G] \le \frac{kD_G}{m}$.*

Proof. Let π_{uv} be a shortest path from u to v in G. Suppose a subset of edges $F \subseteq E$ is removed from G. Then, the distance from u to v decreases only if F and π_{uv} shares at least one edge. If F is chosen uniformly at random from every subset of E with cardinality k, this occurs with probability at most $\frac{kd_G(u,v)}{m}$. Thus we have $S_{\mathrm{HC},k}[G] \le \frac{kD_G}{m}$ as in the proof of Theorem 5.3 and Corollary 5.4. □

Theorem 6.4. *We have $S_{\mathrm{PR},k}[G] \le \frac{2\alpha k}{m(1-\alpha)^2}$.*

Proof. The probability that a walk w of length $\ell(w)$ shares at least one edge with a uniformly random subset F of E with cardinality k is at most $\frac{k\ell(w)}{m}$. Therefore, as in the proof of Lemma 5.9, we have

$$\mathrm{E}\left[\sum_{v\in V}\left|\mathrm{PR}_G(v) - \mathrm{PR}_{G\setminus F}(v)\right|\right] \le \frac{2k\alpha}{1-\alpha}.$$

The rest of the proof is similar to that of Lemma 5.9 and Theorem 5.10. □

Theorem 6.5. *Let G be a undirected connected graph. Then, we have $S_{\mathrm{BT},k}[G] \le \frac{(k^2+3k){D_G}^2/2+kD_G}{m}$.*

To prove this, we introduce an extension of Lemma 5.6:

Lemma 6.6. *If s and t are connected in G, we have*

$$\mathrm{E}\left[\sigma^{d_G(s,t)}_{G,st}(v) - \sigma^{d_G(s,t)}_{G\setminus F,st}(v)\right] \le \frac{k}{m}\cdot d_G(s,t)\sigma^{d_G(s,t)}_{G,st}(v).$$

Table 2: Dataset for experiments on real graphs

Name	Directed?	n	m
ca-CondMat [20]	No	21363	91286
ca-GrQc [20]	No	4158	13422
email-EuAll [20]	Yes	34203	151132
facebook_combined [20]	No	4039	88234
petster-hamster [19]	No	2000	16098

Proof of Theorem 6.5. From Lemma 6.6, we have

$$\mathrm{E}\left[\sum_{v\in V}\left(\frac{\sigma_{G,st}(v)}{\sigma_{G,st}}-\frac{\sigma^{d_G(s,t)}_{G\setminus F,st}(v)}{\sigma_{G,st}}\right)\right]\leq\frac{kd_G(s,t)(D_G+1)}{m}.$$

Also, we have $E\left[d'_{G\setminus F}(s,t)-d'_G(s,t)\right]\leq k(k+1){D_G}^2$. Using them, the proof is analogous to that of Theorem 5.5. □

Theorem 6.7. *Let G be a undirected graph. Then, we have $S_{\mathrm{ST},k}[G]\leq\frac{k\sqrt{(k+1)n}}{m-k}$.*

Proof. Lemma 5.15 can be generalized as $\frac{1}{\mathrm{ST}_G(e)}\leq\frac{1}{\mathrm{ST}_{G\setminus F}(e)}+|F|$. Using this, we have

$$0\leq\frac{\mathrm{ST}_{G\setminus F}(e)-\mathrm{ST}_G(e)}{\mathrm{ST}_G(e)}\leq\sqrt{k(k+1)\left(\mathrm{ST}_{G\setminus F}(e)-\mathrm{ST}_G(e)\right)}.$$

which is an extension of Corollary 5.16. Additionally, we can see $\mathrm{E}\left[\sum_{e\in E\setminus F}\left(\mathrm{ST}_{G\setminus F}(e)-\mathrm{ST}_G(e)\right)\right]\leq\frac{k(n-1)}{m}$ as in the proof of Theorem 5.18. Therefore, by the Cauchy–Schwarz inequality, we obtain $S_{\mathrm{ST},k}\leq k\sqrt{\frac{(k+1)(n-1)}{m(m-k)}}$. □

6.3 Comparison of centralities in terms of k-edge sensitivity

We first compare the upper bounds on k-edge sensitivity with the lower bounds on (1-)edge sensitivity, as a function of k. For the harmonic centrality and PageRank, the ratio is $O(k)$. This means that the harmonic centrality and PageRank are stable against removal of multiple edges as much as against removal of a single edge because, roughly speaking, removal of k edges will cause k times more changes in the centrality values than removal of one edge. However, this ratio is $O(k^2)$ for the closeness centrality and the endpoint betweenness centrality, and is $O(k^{1.5})$ for the spanning tree centrality. Recall that, in Section 5, we argued that the harmonic centrality is preferable for small-world networks. Our results for k-edge sensitivity reinforces this argument.

7 EXPERIMENTS

In Section 5, we analyzed the theoretical bounds with regard to the sensitivity of various centralities. In this section, we confirm that these bounds reflect actual tendencies by conducting experiments on synthetic and real graphs to answer this question. We also experimentally show that edge sensitivity is a good proxy for k-edge sensitivity for large k when comparing various centralities in terms of sensitivity against removals of multiple edges.

7.1 Method

We performed experiments on synthetic and real-world graphs. The former were intended to clarify the tendency of sensitivity on graphs of varying size. The latter were used to determine how our sensitivity metric applies to real-world networks.

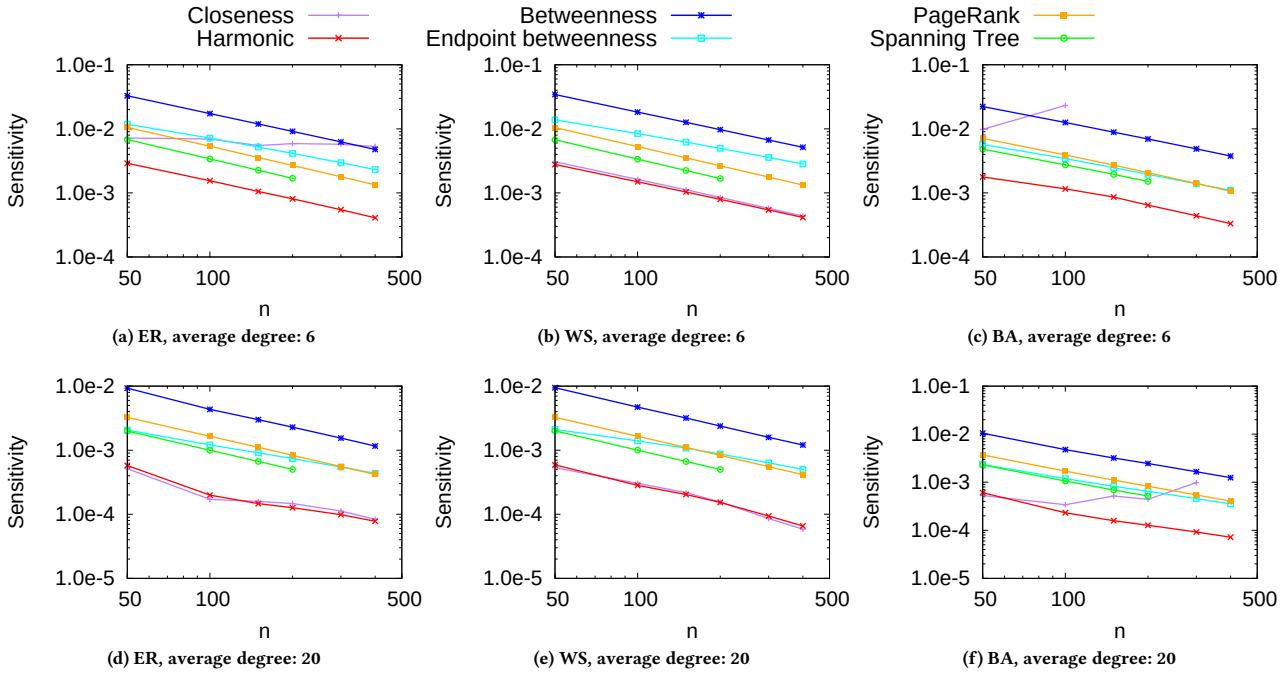
In the experiments on synthetic graphs (Section 7.2), we first generated three kinds of undirected graphs with varying parameters and scales. We used the following three methods for generating graphs:

- Erdős-Rényi (ER) for uniformly random graphs [12]. Average degree: 6, 20.
- Watts-Strogatz (WS) for small-world graphs [33] w diameter of $O(\log n)$. Average degree: 6, 20. Rewiring probability: 0.2.
- Barabasi-Albert (BA) for scale-free graphs [1, 2] with a power-law degree distribution. Core size: 20. Edge probability in the core: 0.5. Average degree of non-core vertices: 6, 20.

We tested graphs of a different number of vertices: 50, 100, 150, 200, 300 and 400. For each setting, we generated 100 graphs. Then we exactly computed the sensitivity of each network for six centralities: the closeness centrality, harmonic centrality, betweeness centrality, endpoint betweenness centrality, PageRank, and spanning tree centrality. The (endpoint) betweenness centrality was computed using the Brandes' algorithm [8] and PageRank was computed using the power iteration method. The spanning tree centrality was computed with the formula provided in [17]. This formula requires computing the pseudo-inverse of matrices. We used the Eigen library [16] for this computation. Since the spanning tree centrality is computationally expensive, we computed it only for graphs of at most 200 vertices.

In the experiments on real-world graphs (Section 7.3), we used directed and undirected graphs provided in the Stanford Large Network Dataset Collection (SNAP) [20] and KONECT [19]. Details for these graphs are shown in Table 2. For each graph, we extracted the largest strongly connected component and removed multiple edges, leaving only single and self edges; n and m in Table 2 refer to the preprocessed graph. As it is too costly to compute sensitivity values for these networks, we estimated them with random sampling. Specifically, we randomly picked 1000 edges, and for each sampled edge e, we computed the *partial sensitivity* $\frac{1}{|V|}\sum_{v\in V}\frac{|C_{G\setminus e}(v)-C_G(v)|}{C_G(v)}$. The average partial sensitivity over all sampled edges was then used as an estimation of the sensitivity value. Furthermore, we derived the standard deviation of the partial sensitivities. We computed sensitivity of spanning tree using a sample of 112 edges (instead of 1000 edges) exclusively for small graphs. For the closeness centrality of undirected graphs, the partial sensitivity will be 1 when e is a bridge. Therefore, we first computed the set of bridges $B\subseteq E$ and then took a sample of 112 edges in $E\setminus B$. The sensitivity of the closeness centrality was estimated by first computing the partial sensitivity of the sampled edges and then augmenting the contribution of bridges to the sensitivity.

Using synthetic graphs, we also computed the k-edge sensitivity for $k>1$ (Section 7.4). The settings used to generate synthetic graphs are identical to the experiments for single-edge sensitivity on synthetic graphs, expect that we fixed the number of vertices to 400. After generating graphs, we estimated the k-edge sensitivity with different ratio of removed edges ρ: 0.01, 0.02, 0.03, 0.05, 0.07, 0.1, 0.15 and 0.2 (the graphs are different among experiments for different ρ). More specifically, for a graph with m edges and the ratio of removed edges ρ, the actual number k of the removed edges



(a) ER, average degree: 6 (b) WS, average degree: 6 (c) BA, average degree: 6

(d) ER, average degree: 20 (e) WS, average degree: 20 (f) BA, average degree: 20

Figure 1: Sensitivity on synthetic graphs of different number of vertices

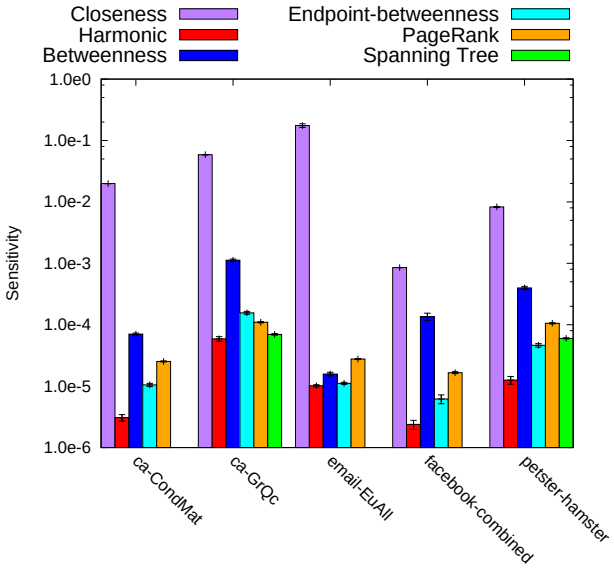


Figure 2: (Estimated) sensitivity of centralities on real-world graphs. The error bars show 1σ ranges.

is ρm, rounded to the nearest integer. We estimated the k-edge sensitivity by random sampling with 500 trials (for the spanning tree centrality, 100 trials) of edge removal. The detailed procedure for random sampling is the same as that for estimating sensitivities of real-world graphs.

All the experiments were conducted on a Windows 10 desktop with an Intel Core i9-7960X (2.8GHz) processor and 32 GiB of RAM. Experiment codes were implemented in C++11 and compiled with gcc 5.4.0 on Ubuntu 16.04 on Windows Subsystem for Linux.

7.2 Synthetic graphs

Figure 1 shows the edge sensitivities of the six centralities on synthetic graphs. Some of the generated graphs were disconnected, and we report the average over connected graphs for the closeness centrality because it was not well defined on disconnected graphs. Additionally, as explained in Section 5, the original betweenness centrality suffers from the zero-division problem. Thus, we altered the definition of the sensitivity of the betweenness centrality to

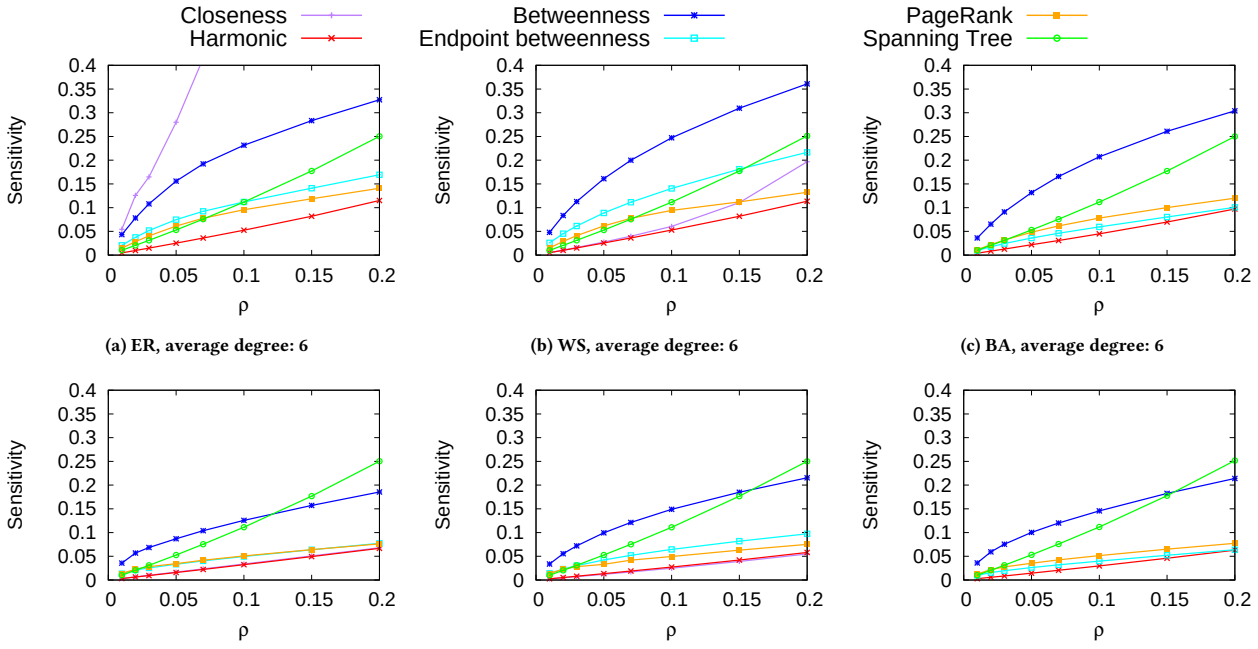
$$\frac{1}{|V||E|}\sum_{e\in E}\sum_{v\in V}\frac{\left|C_{G\setminus e}(v)-C_G(v)\right|}{\max\{C_G(v),C_{G\setminus e}\}}.$$

We let $\frac{|C_{G\setminus e}(v)-C_G(v)|}{\max\{C_G(v),C_{G\setminus e}\}}=0$ when $C_G(v)=C_{G\setminus e}(v)=0$.

We observe that the sensitivity values decrease as the number of vertices increases, except in the case of the closeness centrality in Figure 1a, 1c, and 1f. As explained in Section 5, the sensitivity of the closeness centrality depends to a considerable extent on the number of bridges in the graph. Indeed, the graphs generated by ER with an average degree 6 may have some vertices of degree 1, making the edges adjacent to them bridges. Moreover, BA generates scale-free graphs, in which the degree distribution is highly biased. Thus, it is natural for graphs generated by BA to contain many vertices of degree 1.

On the other hand, in Figure 1b, 1d and 1e, the closeness and harmonic centralities have almost the same stability values. This suggests that the difference in sensitivity between the closeness and harmonic centralities is mainly due to the presence of bridges.

Apart from the inferiority of the closeness centrality on graphs with many bridges, the closeness and harmonic centralities were among the least sensitive, followed by the endpoint betweenness



(a) ER, average degree: 6 (b) WS, average degree: 6 (c) BA, average degree: 6

(d) ER, average degree: 20 (e) WS, average degree: 20 (f) BA, average degree: 20

Figure 3: Multi-edge sensitivity on synthetic graphs with different ratio of removed edges ($n = 400$)

centrality and PageRank. This result is consistent with our theoretical analysis, in which the closeness centrality (without the presence of bridges) and the harmonic centrality have smaller sensitivities than the endpoint betweenness and PageRank.

However, we observe that the sensitivity of spanning tree centrality is consistently less than the endpoint betweenness centrality and PageRank. Although our analysis in Theorem 5.18 states $S_{\mathrm{SP}}(G) = O(\sqrt{n}/m)$, our experimental results suggest that $S_{\mathrm{SP}}(G)$ is almost proportional to $\frac{1}{m}$ on synthetic graphs. Narrowing this gap will be pursued in future work.

7.3 Real-world graphs

In Figure 2, we show approximations to the edge sensitivities of the fix centralities on real-world graphs. We see tendencies that are similar to the results with synthetic graphs: the harmonic centrality was the least sensitive, followed by the spanning-tree centrality, endpoint-betweenness centrality and PageRank, and finally the betweenness centrality. Notably, the sensitivity of the closeness centrality varied significantly depending on the graph, which may caused by different ratios of bridges.

7.4 k-edge sensitivity

In Figure 3, we show the k-edge sensitivities of the six centralities on synthetic graphs. First, we can observe that the ranking among the six centralities with respect to k-edge centrality is almost the same as that with respect to edge sensitivity.

Next, we investigate how sensitivity values change as ρ increases. For all settings and all centralities, the k-edge sensitivity is almost proportional to ρ when ρ is small (specifically, $\rho < 0.05$). For the harmonic centrality and PageRank, this result is consistent with our analysis in Section 6, as our upper bounds for those centralities are linear in k. However, there are gaps between the empirical results and theoretical analysis for the closeness centrality and the betweenness centrality, for which our upper bounds are merely $O(k^2)$. Explaining these gaps is a future work. Also, we can observe that the sensitivity of the spanning tree centrality steadily grows as ρ increases, while others except for closeness tend to slow down. Investigating its reason is also a future work.

8 CONCLUSION

In this work, to compare centralities based on their sensivitity to modifications in the graph, we introduced a quantitative measure called (average-case) k-edge sensitivity, which measures how much the centrality value of a uniformly chosen vertex (or an edge) changes when we remove a uniformly chosen set of k edges. We theoretically gave upper bounds on the k-edge sensitivities of six major centralities. For $k = 1$, we also gave almost matching lower bounds except for the spanning tree centrality. Our experimental results on synthetic and real graphs confirmed the tendency predicted by the theoretical analysis and confirmed that single edge sensitivity can be used as a good proxy for k-edge sensitivity when comparing various centralities in terms of sensitivity against removals of multiple edges.

In this work, we considered edge removal on unweighted graphs as the error model. We suppose we can extend the analysis to other settings, such as edge addition, vertex removal, or graphs with different edge lengths. Analyzing the sensitivity of centralities on such settings will be a subject of future work.

ACKNOWLEDGMENTS

S. Murai would like to thank Hiroshi Imai for giving useful comments and advices.

REFERENCES

[1] Edoardo M Airoldi and Kathleen M Carley. 2005. Sampling algorithms for pure network topologies: a study on the stability and the separability of metric embeddings. *ACM SIGKDD Explorations Newsletter* 7, 2 (2005), 13–22.

[2] Réka Albert and Albert-László Barabási. 2002. Statistical mechanics of complex networks. *Reviews of modern physics* 74, 1 (2002), 47.

[3] Konstantin Avrachenkov, Nelly Litvak, Danil Nemirovsky, and Natalia Osipova. 2007. Monte Carlo methods in PageRank computation: When one iteration is sufficient. *SIAM J. Numer. Anal.* 45, 2 (2007), 890–904.

[4] Alex Bavelas. 1950. Communication Patterns in Task-Oriented Groups. *The Journal of the Acoustical Society of America* 22, 6 (1950), 725–730.

[5] Murray A. Beauchamp. 1965. An improved index of centrality. *Behavioral Science* 10, 2 (1965), 161–163.

[6] Paolo Boldi and Sebastiano Vigna. 2014. Axioms for centrality. *Internet Mathematics* 10, 3-4 (2014), 222–262.

[7] Stephen P Borgatti, Kathleen M Carley, and David Krackhardt. 2006. On the robustness of centrality measures under conditions of imperfect data. *Social networks* 28, 2 (2006), 124–136.

[8] Ulrik Brandes. 2001. A faster algorithm for betweenness centrality. *Journal of mathematical sociology* 25, 2 (2001), 163–177.

[9] Ulrik Brandes. 2008. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks* 30, 2 (2008), 136–145.

[10] Elizabeth Costenbader and Thomas W Valente. 2003. The stability of centrality measures when networks are sampled. *Social networks* 25, 4 (2003), 283–307.

[11] Peter G Doyle and J Laurie Snell. 1984. Random walks and electric networks. (1984).

[12] P. Erdős and A. Rényi. 1959. On random graphs. I. *Publ. Math. Debrecen* 6 (1959), 290–297.

[13] Terrill L Frantz, Marcelo Cataldo, and Kathleen M Carley. 2009. Robustness of centrality measures under uncertainty: Examining the role of network topology. *Computational and Mathematical Organization Theory* 15, 4 (2009), 303.

[14] Linton C Freeman. 1977. A Set of Measures of Centrality Based on Betweenness. *Sociometry* 40, 1 (1977), 35–41.

[15] Gourab Ghoshal and Albert-László Barabási. 2011. Ranking stability and super-stable nodes in complex networks. *Nature communications* 2 (2011), 394.

[16] Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3. http://eigen.tuxfamily.org.

[17] Douglas J Klein and Milan Randić. 1993. Resistance distance. *Journal of mathematical chemistry* 12, 1 (1993), 81–95.

[18] Jon M Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)* 46, 5 (1999), 604–632.

[19] Jérôme Kunegis. 2013. KONECT - The Koblenz Network Collection. In *Proc. Int. Web Observatory Workshop*. 1343–1350.

[20] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[21] Massimo Marchiori and Vito Latora. 2000. Harmony in the small-world. *Physica A: Statistical Mechanics and its Applications* 285, 3-4 (2000), 539–546.

[22] Charalampos Mavroforakis, Richard Garcia-Lebron, Ioannis Koutis, and Evimaria Terzi. 2015. Spanning edge centrality: Large-scale computation and applications. In *Proceedings of the 24th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 732–742.

[23] Andrew Y Ng, Alice X Zheng, and Michael I Jordan. 2001. Link analysis, eigenvectors and stability. In *International Joint Conference on Artificial Intelligence*, Vol. 17. Lawrence Erlbaum Associates Ltd, 903–910.

[24] Qikai Niu, An Zeng, Ying Fan, and Zengru Di. 2015. Robustness of centrality measures against network manipulation. *Physica A: Statistical Mechanics and its Applications* 438 (2015), 124–131.

[25] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report. Stanford InfoLab.

[26] John Platig, Edward Ott, and Michelle Girvan. 2013. Robustness of network measures to link errors. *Physical Review E* 88, 6 (2013), 062812.

[27] Gert Sabidussi. 1966. The centrality index of a graph. *Psychometrika* 31, 4 (1966), 581–603. https://doi.org/10.1007/BF02289527

[28] A. A. Schoone, H. L. Bodlaender, and J. van Leeuwen. 1987. Improved diameter bounds for altered graphs. In *Graph-Theoretic Concepts in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 227–236.

[29] S. Segarra and A. Ribeiro. 2016. Stability and Continuity of Centrality Measures in Weighted Graphs. *IEEE Transactions on Signal Processing* 64, 3 (Feb 2016), 543–555.

[30] Andreia Sofia Teixeira, Pedro T Monteiro, João A Carriço, Mário Ramirez, and Alexandre P Francisco. 2013. Spanning edge betweenness. In *Workshop on mining and learning with graphs*, Vol. 24. 27–31.

[31] Sho Tsugawa and Hiroyuki Ohsaki. 2015. Analysis of the robustness of degree centrality against random errors in graphs. In *Complex Networks VI*. Springer, 25–36.

[32] Vladimir Ufimtsev, Soumya Sarkar, Animesh Mukherjee, and Sanjukta Bhowmick. 2016. Understanding stability of noisy networks through centrality measures and local connections. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 2347–2352.

[33] Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of 'small-world' networks. *nature* 393, 6684 (1998), 440.

[34] Barbara Zemljič and Valentina Hlebec. 2005. Reliability of measures of centrality and prominence. *Social Networks* 27, 1 (2005), 73–88.

Modeling Heart Rate and Activity Data for Personalized Fitness Recommendation

Jianmo Ni
University of California, San Diego
jin018@ucsd.edu

Larry Muhlstein
University of California, San Diego
larrymuhlstein@gmail.com

Julian McAuley
University of California, San Diego
jmcauley@ucsd.edu

ABSTRACT

Activity logs collected from wearable devices (e.g. *Apple Watch*, *Fitbit*, etc.) are a promising source of data to facilitate a wide range of applications such as personalized exercise scheduling, workout recommendation, and heart rate anomaly detection. However, such data are heterogeneous, noisy, diverse in scale and resolution, and have complex interdependencies, making them challenging to model. In this paper, we develop context-aware sequential models to capture the personalized and temporal patterns of fitness data. Specifically, we propose *FitRec* – an LSTM-based model that captures two levels of context information: context within a specific activity, and context across a user's activity history. We are specifically interested in (a) estimating a user's heart rate profile for a candidate activity; and (b) predicting and recommending suitable activities on this basis. We evaluate our model on a novel dataset containing over 250 thousand workout records coupled with hundreds of millions of parallel sensor measurements (e.g. heart rate, GPS) and metadata. We demonstrate that the model is able to learn contextual, personalized, and activity-specific dynamics of users' heart rate profiles during exercise. We evaluate the proposed model against baselines on several personalized recommendation tasks, showing the promise of using wearable data for activity modeling and recommendation.

KEYWORDS

Fitness Recommendation; Sequential Modeling; Personalization

ACM Reference Format:
Jianmo Ni, Larry Muhlstein, and Julian McAuley. 2019. Modeling Heart Rate and Activity Data for Personalized Fitness Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313643

1 INTRODUCTION

The development of wearable technologies has given people the opportunity to measure and track their activities via mobile devices (e.g. smart watches). Such devices collect various types of data, including health-related measurements (e.g. heart rate) and contextual measurements (e.g. location, altitude, activity type). An example of such data (from *endomondo.com*) is shown in fig. 1. Harnessing this data to model fitness is beneficial for developing safer and more reliable activity-related services [1]. However, this remains a challenging task for a number of reasons: (1) the collected sequential data are often heterogeneous and the interdependencies among different variables are hard to capture; (2) users' activity patterns and health conditions change over time; and (3) the data has high variance across users, and while large datasets can be easily collected, the amount of data associated with each user is limited.

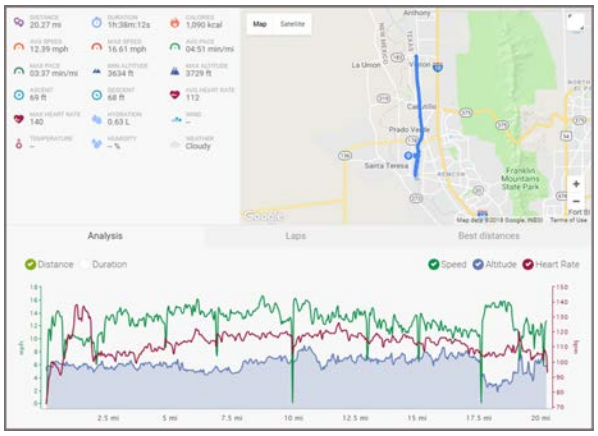


Figure 1: A representative example of our data collected from *endomondo.com*. The figure contains measurements during one workout, including route, speed, altitude and heart rate sequences as well as overall statistics.

Researchers have recently sought to incorporate deep learning techniques to capture the dynamics of sequential data. One of our goals is to adapt such models to capture personalized dynamics in GPS and heart rate data. Recurrent Neural Network (RNN) based models [6, 13] have been proposed to model multivariate sequential data and shown promising performance [5, 20]. Compared with traditional models, RNNs are able to learn complex non-linear relationships between input variables. Furthermore, RNNs are convenient for modeling sequential problems with different input and output structures (e.g. sequential inputs and sequential outputs, or sequential inputs and numerical outputs). This allows us to straightforwardly design model variants for different prediction tasks on our data. More recently, attention mechanisms [3] have further boosted the performance of RNN based models due to their ability to make the model aware of all previous hidden states.

In order to bring state-of-the-art models of sequential data to bear on large datasets of fitness activities, it is important to ask how we can *personalize* models of sequence data to individual users. Personalization is crucial when modeling people's fitness due to the wide variation among individuals. For example, users may differ

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313643

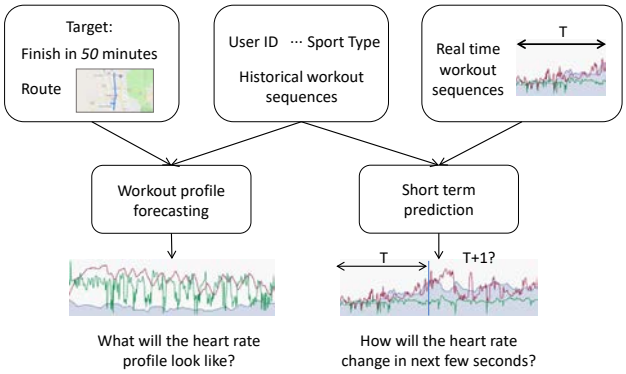


Figure 2: *FitRec* can be applied to two different tasks: workout profile forecasting and short term prediction.

significantly in comfort heart rate zones, adaptation ability for different exercises, etc. Accounting for such differences requires robust modeling techniques to capture the dynamics of both explicit and implicit user-dependent features. An effective model needs to efficiently learn both static and temporal features from historical data.

In this paper, we address these challenges by exploring a large-scale dataset of workout records and building a predictive model leveraging context within and across workouts. Specifically, we propose a Long Short-Term Memory (LSTM) based model *FitRec* that takes as input user attributes and multiple workout measurement sequences. *FitRec* infers static user embeddings based on the attributes and temporal user embeddings from historical workout measurements. By combining these inferred latent representations of our users, *FitRec* can then be applied to (1) Quantitative tasks, such as personalized sequential modeling to predict how workout measurements (e.g. heart rate) will change across a workout, either beforehand (i.e., based on a map of the intended route) or in real time as the user exercises; (2) Qualitative tasks, such as identifying important features that affect workout performance, or identifying clusters of users based on common embedding structures; and (3) Recommendation tasks, such as recommending alternate routes that will achieve a target heart rate profile.

In particular, we evaluate *FitRec* on two quantitative tasks: 'workout profile forecasting' and 'short term prediction' as shown in fig. 2. In the former, we are concerned with predictions such as *estimating a user's likely speed and heart rate profile given the activity they intend to perform* (e.g. cycling a particular GPS route); in the latter we are interested in short-term prediction, e.g. determining during an ongoing workout *how the user's heart rate will change in the next few seconds or minutes*. To apply to both tasks, *FitRec* uses shared embedding layers while adapting a common predictive component: a two-layer stacked LSTM module for workout profile forecasting and an attention-based encoder-decoder module for short term prediction. We also demonstrate the model's capability to provide personalized recommendations that could help users toward their fitness objectives. Workout profile forecasting can be used (for example) to recommend routes that match a user's desired exercise style, while short term prediction might be used to dynamically help a user match their target heart rate. Fitness and activity data have attracted interest in the data mining and recommendation fields because they contain rich contextual information (e.g. spatial and temporal information) about users that might complement existing recommender systems [2]. However, there have been few works studying how these data can be used in real-world recommendation applications.

The key contributions in this paper are threefold:

(1) We propose an LSTM-based model *FitRec* that considers context both within and across workouts. *FitRec* infers static user embeddings from user attributes and meanwhile learns temporal embeddings from users' recent workout sequences. *FitRec* can be applied to various tasks by using either a two-layer stacked LSTM module or an attention-based encoder-decoder module. We show the model's performance compared with prior sequential modeling baselines such as Multilayer Perceptrons (MLP) [15] and Dual-stage Attention-based RNNs (DA-RNN) [25].
(2) We demonstrate that *FitRec* can be used on two real-world recommendation tasks: workout route recommendation and short term heart rate prediction. We evaluate performance in terms of standard recommendation metrics (e.g. AUC, F1-score) and show that our model outperforms strong baselines.
(3) We contribute a workout dataset that contains over 250 thousand workout records with multiple types of heterogeneous sequential signals as well as metadata. To the best of our knowledge, we are the first to release a large-scale dataset of this type. Beyond the recommendation problems considered here, such a dataset has significant potential for re-use in research on heart rate data, sequence modeling, personalization, etc.

While there are several works focused on analyzing human activities, we are (to the best of our knowledge) the first to investigate personalized recommendation using sequential modeling methods on fitness-oriented objectives and to conduct extensive experiments on a real-world dataset.

2 RELATED WORK

Our work focuses on leveraging recent advances in RNN-based sequential modeling for fitness prediction and recommendation. We discuss the related work from each relevant area as follows.

Mining Sensor Data. One line of work falls into the category of *pervasive computing*, which studies how to collect and handle data from mobile and wearable devices. The data may include various types of human behavior such as sleep, exercise, surgery, health, etc. [16, 19, 23]. [1] discusses the challenges and recent advances in population-scale pervasive health research. Their work emphasizes the importance of sensor data for uncovering the causal relationships between human activity and health outcomes. [4] provides an overview of data mining research in the healthcare domain and concludes that the capability to build context-aware and personalized decision systems is important for the pervasive sensing market.

Recently, there is a growing trend toward modeling human wellness using sensor data. Farseev et al. collected users' exercise data and proposed a model to combine social network information to

predict user wellness trends [9]. Particularly, they focused on BMI (Body Mass Index) categorization during several periods for each specific user. [7] focused on detecting physical exercise types given limited training data. They extracted features from exercise data (e.g. heart rate, distance) and proposed an AdaBoost-based method to predict the exercise type (e.g walking, aerobics, running, etc.). Both of these works focus on handcrafting features from sensor data, which are then applied to classification problems. Although similar in data modality, our work differs in terms of the prediction tasks we consider, i.e., sequential modeling and sequence prediction.

Context-aware Modeling. Context-aware models have been successfully applied in many fields with abundant contextual information such as recommender systems, social networks, clinical predictions, etc. [8, 24]. Like these applications, fitness and exercise data naturally has heterogeneous input structure in the form of sequential measurements and contextual information. In [18] a recurrent model was proposed with a context-aware layer for predicting a patient's Blood Pressure (BP). The model considers contextual input (e.g. age, gender, BMI) which is encoded via one-hot vectors or numerical values. Experiments showed that adding this contextual information to the model improves BP prediction accuracy. Unlike their approach, we use attribute embedding and contextual embedding modules consisting of LSTM layers to learn from entire historical sequences. Furthermore, their blood pressure measurement data does not consist of measurements within each activity, which differs from the data we consider.

RNN based sequential modeling. In recent years, Recurrent Neural Networks (RNNs) have shown remarkable effectiveness in modeling sequential data, such as text, audio, and clinical data [20], among others. [17] introduced an LSTM-based model to address traditional multivariate time series prediction problems. They combined Convolutional Neural Networks (CNN) with a recurrent skip layer to capture both long- and short-term dependencies in time series data. The model achieves significant improvements over LinearSVMs and 'vanilla' RNNs. [25] proposed a dual-stage attention model to solve the problem of exogenous time series prediction. They designed an input level attention mechanism along with temporal attention to capture the relations between input variables. Similarly, we incorporate attention mechanisms in sequential data to explore the relations between contextual variables in exercise data.

Personalized Recommendation. Several works on personalized recommendation have tried to incorporate content (e.g. item attributes) and context (e.g. user clicks, purchases) to augment the model in straightforward and typically 'static' scenarios (e.g. e-commerce). Recently, researchers from both industry and academia have sought to develop systems for personalized fitness recommendation. [21] focused on personalized running route recommendation, by considering a variety of targets such as user preferences, goals, and environment. [9] combined workout and social network information from the same user in order to conduct wellness prediction. However, neither of these works focus on modeling sequential fitness data (e.g. heart rate sequences). Though [9] released a public dataset, unlike ours it does not provide complete sequential traces for each workout, but rather includes hand-crafted statistics extracted from each sequence.

Table 1: Notation

Notation	Description
$\mathcal{T}_{train}$, $\mathcal{T}_{test}$	training set and test set for two prediction tasks
$\mathbf{X}, \mathbf{Z} \in \mathbb{R}^{N \times T}$	contextual sequences of the current and the most recent workout
$\mathbf{y}, \mathbf{y}' \in \mathbb{R}^{T}$	target sequence of the current and the most recent workout
$\mathbf{a}$	attributes associated with the workout
T	length of input and target sequences
L	the number of sampled data points L
N	number of contextual sequences and each sequence is associated with a variable.
m	number of attributes
$\lvert a_i \rvert$	number of distinct values of the ith attribute
D_1	dimension of attribute embeddings
D_2	dimension of contextual embeddings
$\mathrm{E}_{a_i} \in \mathbb{R}^{\lvert a_i \rvert \times D_1}$	embedding matrix of the ith attribute
$\mathbf{e}_{a_i} \in \mathbb{R}^{D_1}$	attribute embedding learned from the ith attribute
$\mathbf{e}_t \in \mathbb{R}^{D_2}$	contextual embedding learned from historical workout sequences

While prior work has considered fitness and activity modeling from a variety of perspectives, we believe there is a research gap in terms of studying low-level sequential information (e.g. heart rate, pace) as a means of learning more fine-grained models of users and activities. This motivates us to model and publish a large-scale workout dataset with heterogeneous sequential and attribute data, and to conduct exploratory experiments on fitness recommendation tasks. We expect that our work will facilitate future study of both fitness and personalized heart rate profiling.

3 APPROACH

3.1 Notation and Tasks

Assume we are given N sequences $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_T) \in \mathbb{R}^{N \times T}$ and one target sequence $\mathbf{y} = (y_1, \ldots, y_T) \in \mathbb{R}^T$, where $\mathbf{x}_t = (x_t^1, x_t^2, \ldots, x_t^N) \in \mathbb{R}^N, y_t \in \mathbb{R}$, and $t \in [1, T]$. Here $\mathbf{X}$ can be considered as a combination of various contextual sequences (e.g. distance, altitude) for the current workout and $\mathbf{y}$ is a target measurement sequence (e.g. heart rate, speed). Further we assume there are N historical contextual sequences $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_T) \in \mathbb{R}^{N \times T}$ and one historical target sequence $\mathbf{y}' = (y'_1, \ldots, y'_T) \in \mathbb{R}^T$, where $\mathbf{Z}$ and $\mathbf{y}'$ are from the most recent workout. For the dataset used in this work, the number of sampled data points L for each sequence of the workouts is equal. Therefore, we also assume the same length T for all sequences in our model. Note that this is not a requirement of our model, but merely a characteristic of the data we collect, in which all sequences are quantized to have the same number of samples regardless of the original workout duration. The model can be adapted to various sequence lengths by a simple padding strategy. Each workout is also associated with multiple attributes $\mathbf{a} = (a_1, \ldots, a_m)$ (e.g. userID, sport type, gender). Our notation is summarized in table 1.

We consider two types of sequential prediction problems: workout profile forecasting and short term prediction.

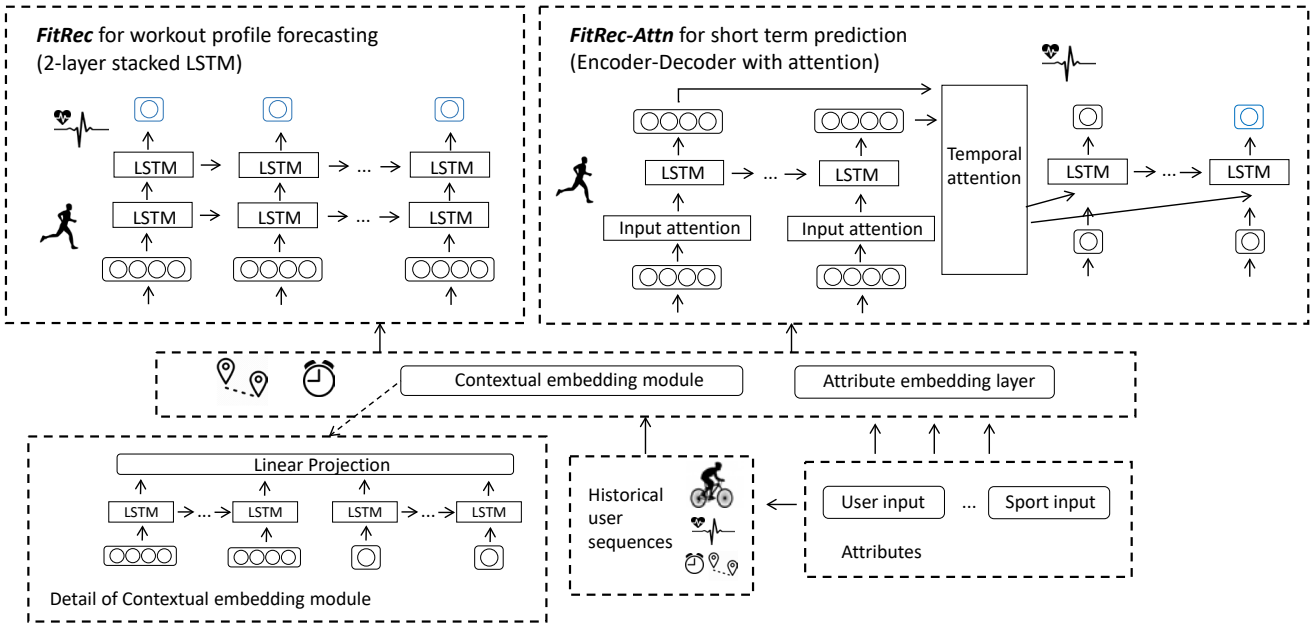


Figure 3: Model structure for workout profile forecasting (*FitRec*) and short term prediction (*FitRec-Attn*). *FitRec* contains a 2-layer stacked LSTM and *FitRec-Attn* has an encoder-decoder module with dual-stage attention. The final outputs are colored in blue.

3.1.1 Workout profile forecasting. Given the contextual sequences $\mathbf{X}$, attributes $\mathbf{a}$ for a candidate workout, historical sequences $\mathbf{Z}$ and $\mathbf{y}'$, as well as the total time of the workout, predict the entire target sequence $\mathbf{y} = (y_1, \ldots, y_T)$. Notably, the input and target sequences correspond to the whole workout and T equals the total length L of the workout. This prediction task might correspond to a scenario where a user specifies a route as well as their desired total time (i.e. how much time they plan to spend on the workout), and the system estimates their likely heart rate profile.[1]

3.1.2 Short term prediction. Given the contextual sequences $\mathbf{X}$, attributes $\mathbf{a}$ for a candidate workout, historical sequences $\mathbf{Z}$ and $\mathbf{y}'$, as well as the previous target sequence $(y_1, \ldots, y_{T-1})$, predict the target output y_T at time step T. Here the input and target sequences correspond to a fixed window of one workout (i.e. part of the whole workout) and T equals the window size (i.e. T is smaller than L). This task is essentially a form of auto-regressive modeling that allows us to predict short term heart rate dynamics during an activity.

3.2 Relation Between the Two Tasks

Our two tasks differ in terms of their given inputs, prediction outputs, and application scenarios as shown in fig. 2.

Workout profile forecasting aims to predict a user's workout performance (including speed and heart rate) given an expected workout time and route. The goal is to predict the overall trends and characteristics of each measurement (e.g. max/min/avg heart rate) before the user starts the workout. As a result, the model can help the user find workout plans that meet their expectations on performance, or could be used to find alternate workout routes that are similar to their previous routes in terms of how the user will respond to them. This model can be adapted to applications such as training schedule recommendation, alternative route recommendation, etc.

On the other hand, short term prediction focuses on forecasting sequential measurements during an ongoing workout. This is similar to traditional problems of time series prediction with exogenous variables [11, 25] but differs in that fitness data has measurements both within and across activities which need to be modelled separately. Short term prediction is useful in scenarios like anomaly detection and real-time decision making (e.g. advising a user that they should slow down in the next minute to avoid exceeding their desired maximum heart rate).

3.3 Model Structure

As shown in fig. 3, we propose two models: *FitRec* for workout profile forecasting and *FitRec-Attn* for short term prediction.

Both models share the same basic structure consisting of a contextual embedding module and an attribute embedding layer. These layers encode user attributes and historical information into an embedded representation that captures their latent individual attributes (e.g. their cardiovascular fitness, endurance, etc.) and can facilitate personalized prediction. Following the embedding layers, the two models have different predictive components. As shown on the top left of fig. 3, *FitRec* consists of a 2-layer stacked LSTM which simultaneously outputs the prediction of the target variable at all time steps. *FitRec-Attn* includes an encoder-decoder network with

[1] We also experimented with a variant where the total time was *not* provided as input, but found there to be excessive variance between (e.g.) running/sprinting/jogging styles such that heart rates could not be reliably estimated without first knowing a rough intended speed.

an attention mechanism that outputs the prediction at each time step one by one as shown on the top right of fig. 3. We describe the details for each component as follows.

3.3.1 LSTM to process sequences. In both models, we use Long Short-Term Memory (LSTM) [13] to process the input and target sequences. Given the input sequence $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_T)$, an LSTM learns an update function

$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{x}_t), \tag{1}$$

where $\mathbf{h}_t$, $\mathbf{h}_{t-1}$ are the hidden states of the LSTM at time step t and $t-1$, respectively, and $\mathbf{x}_t$ is the input at time step t. Different from traditional RNNs, the LSTM cell introduces several gates and memory cells. We follow the LSTM structure from [25, 28]:

$$\begin{aligned} \begin{pmatrix} \mathbf{i} \\ \mathbf{f} \\ \mathbf{o} \\ \mathbf{g} \end{pmatrix} &= \begin{pmatrix} \text{sigmoid} \\ \text{sigmoid} \\ \text{sigmoid} \\ \tanh \end{pmatrix} \mathbf{W} \begin{pmatrix} \mathbf{h}_{t-1} \\ \mathbf{x}_t \end{pmatrix} \\ \mathbf{s}_t &= \mathbf{s}_{t-1} \odot \mathbf{f}_t + \mathbf{i}_t \odot \mathbf{g}_t \\ \mathbf{h}_t &= \tanh(\mathbf{s}_t) \odot \mathbf{o}_t, \end{aligned} \tag{2}$$

where $\mathbf{s}$ is the cell state; $\mathbf{g}$ is the activation function applied on the input; $\mathbf{i}$, $\mathbf{o}$ and $\mathbf{f}$ are three sigmoid gates ('input,' 'output,' and 'forget,' respectively); and $\mathbf{h}$ is the hidden state of the cell. W is the learned weight matrix.

In the following sections, we use the notation 'LSTM' to represent the complete update function of an LSTM cell and ignore the cell state $\mathbf{s}$ for simplicity.

3.3.2 Context within and across activities. To address the difficulties in building personalized models, we consider learning attribute embeddings and contextual embeddings associated with the current workout. Specifically, for each attribute input (e.g. user, sport type, gender), we use attribute embedding layers to obtain latent representations of these attributes:

$$\mathbf{e}_{a_i} = \mathrm{E}_{a_i}(a_i), \forall i \in [1, m] \tag{3}$$

where $\mathbf{e}_{a_i}$ is the embedding for attribute a_i.

To extract contextual features from historical activity measurements, we use two LSTMs to process the historical sequences $\mathbf{Z}$ and $\mathbf{y}'$ respectively, then concatenate them and feed them into a linear projection layer to obtain a contextual embedding:

$$\begin{aligned} \mathbf{h}_{1,t} &= \mathrm{LSTM}_1(\mathbf{z}_t, \mathbf{h}_{1,t-1}), \\ \mathbf{h}_{2,t} &= \mathrm{LSTM}_2(y'_t, \mathbf{h}_{2,t-1}), \\ \mathbf{e}_t &= \mathbf{W}_e[\mathbf{h}_{1,t}; \mathbf{h}_{2,t}] + \mathbf{b}_e, \end{aligned} \tag{4}$$

where $\mathbf{h}_{1,t}$ and $\mathbf{h}_{2,t}$ are the hidden states of the two LSTMs, $\mathbf{e}_t$ is the projected contextual embedding, and $\mathbf{W}_e$ and $\mathbf{b}_e$ are trained parameters.

After obtaining these embeddings, they are repeated at each time step and concatenated with the contextual embedding and the input variables:

$$\mathbf{u}_t = [\mathbf{x}_t; \mathbf{e}_t; \mathbf{e}_{a_1}; \ldots; \mathbf{e}_{a_m}], \tag{5}$$

where $\mathbf{u}_t$ is the new concatenated contextual sequence input at time step t. The sequences $\mathbf{u} = (\mathbf{u}_1, \ldots, \mathbf{u}_T) \in \mathbb{R}^{K \times T}$ are then fed into the cascaded predictive module for different tasks. Here K is the dimension of each $\mathbf{u}_t$.

3.3.3 FitRec. We use a 2-layer LSTM model with a projection layer on the output for the workout profile forecasting problem, which we call *FitRec*:

$$\begin{aligned} \mathbf{h}_t &= \mathrm{LSTM}(\mathbf{u}_t, \mathbf{h}_{t-1}), \\ \hat{y_t} &= \mathrm{SELU}(\mathbf{W}_{NAT}\mathbf{h}_t + \mathbf{b}_{NAT}), \end{aligned} \tag{6}$$

where $\mathbf{h}_t$ is the hidden state of the LSTM's second layer, SELU is the scaled exponential linear unit activation function and $\mathbf{W}_{NAT}$ and $\mathbf{b}_{NAT}$ are learned parameters. The MLP output layer's dimension is set as the sequence length T. As shown in fig. 3, *FitRec* will output the prediction for the whole workout (all T time steps) simultaneously.

3.3.4 FitRec-Attn. Inspired by the success of the dual-stage attention RNN in modeling sequential data [25], we build *FitRec-Attn* – a dual-stage attention encoder-decoder model on top of the attribute and contextual embedding module for short term prediction.

Given the concatenated contextual sequences $\mathbf{u}$, we process them using an encoder LSTM with input attention. Input attention helps to find the importance of each input dimension among the concatenated contextual sequences. Consider $\mathbf{u}^k = (u_1^k, \ldots, u_T^k)$ for the kth input dimension of $\mathbf{u}$. An attention score s_t^k is calculated for $\mathbf{u}^k$ and the previous encoder hidden state $\mathbf{h}_{t-1}$, which represents the importance of that dimension at time step t. The weight is used to scale the original input and build a new input sequence $\tilde{\mathbf{u}_t} = (\alpha_t^1 u_t^1, \ldots, \alpha_t^K u_t^K)$:

$$\begin{aligned} s_t^k &= \mathbf{v}_s^\top(\mathbf{W}_s[\mathbf{h}_{t-1}; \mathbf{u}^k] + \mathbf{b}_s)\ 1 \le k \le K, \\ \alpha_t^k &= \frac{\exp(s_t^k)}{\sum_{j=1}^K \exp(s_t^j)}, \end{aligned} \tag{7}$$

where α_t^k is the normalized attention score, and $\mathbf{v}_s$, $\mathbf{W}_s$ and $\mathbf{b}_s$ are learned parameters. The newly computed $\tilde{\mathbf{u}}_t$ is used to update the encoder LSTM_e:

$$\mathbf{h}_t = \mathrm{LSTM}_e(\mathbf{h}_{t-1}, \tilde{\mathbf{u}}_t). \tag{8}$$

Following the encoder, a decoder LSTM with temporal attention is used to predict the target y_t. Specifically, the temporal attention score is calculated for each decoder hidden state $\mathbf{d}_t$ and all encoder hidden states, which represents how much the encoder hidden state is related to the decoder. Then the attention scores are used as weights to produce a context vector $\mathbf{c}_t$:

$$\begin{aligned} l_t^i &= \mathbf{v}_l^\top(\mathbf{W}_l[\mathbf{d}_{t-1}; \mathbf{h}_i] + \mathbf{b}_l)\ 1 \le i \le T, \\ \beta_t^i &= \frac{\exp(l_t^i)}{\sum_{j=1}^T \exp(l_t^j)}, \quad \mathbf{c}_t = \sum_{i=1}^T \beta_t^i \mathbf{h}^i \end{aligned} \tag{9}$$

where l_t^i and β_t^i are the attention score and normalized attention score for the t^{th} decoder hidden state and the i^{th} encoder hidden state. $\mathbf{v}_l$, $\mathbf{W}_l$ and $\mathbf{b}_l$ are learned parameters.

Once the context vector is obtained, it is concatenated with the previous target variable y_{t-1} and used to update the decoder LSTM:

$$\begin{aligned} \tilde{y}_{t-1} &= \tilde{\mathbf{w}}^\top[y_{t-1}; \mathbf{c}_{t-1}] + \tilde{b}, \\ \mathbf{d}_t &= \mathrm{LSTM}_d(\mathbf{d}_{t-1}, \tilde{y}_{t-1}), \end{aligned} \tag{10}$$

where $\tilde{y}_{t-1}$ is a linear transformation of the concatenated context variable and target variable at time step t (as in [25]).

Given the previous target sequence $(y_1, \ldots, y_{T-1})$, each y_{t-1} is processed sequentially and $\mathbf{c}_t$ and $\mathbf{d}_t$ are updated. Finally, we obtain the context vector and hidden state $\mathbf{c}_T$ and $\mathbf{d}_T$ at the last time step T. These two are fed into a linear output layer to predict the target variable $\hat{y_T}$ as shown in fig. 3:

$$\hat{y_T} = \mathbf{W}_{AT}[\mathbf{d}_T; \mathbf{c}_T] + \mathbf{b}_{AT}, \tag{11}$$

where $\mathbf{W}_{AT}$ and $\mathbf{b}_{AT}$ are learned parameters.

3.4 Objective Function

For both workout profile forecasting and short term prediction, we use the mean squared error as our loss function:

$$\mathcal{L} = \frac{1}{|N_{train}|} \sum_{\mathbf{y} \in \mathcal{T}_{train}} \sum_{t=1}^{L} (\hat{y_t} - y_t)^2, \tag{12}$$

where $|N_{train}|$ is the number of total time steps for all workouts in the training set, and L is the total length of each workout.

4 EXPERIMENTS

In this section, we first introduce the *Endomondo* dataset, a new real-world dataset for empirical study. Then we conduct experiments on this dataset and show the performance of our models for the two prediction tasks.

4.1 The Endomondo Dataset

We collect data from *endomondo.com*. The data includes both measurements (i.e., heart rate, timestamps, distance, speed) and contextual data (i.e., longitude, latitude, altitude, gender, sport, user identity). The original data has a high rate of missing measurements for distance and speed, presumably due to the fact that these fields are only available from certain devices. Hence we further introduce two derived sequences: (1) **derived distance**: Calculate the distance between two data points given their latitudes and longitudes via the Haversine formula [26]; (2) **derived speed**: Divide the derived distance and the time interval between two data points. In our experiments, we use the derived speed instead of the original speed (which we found to be unreliable due to missing data or erroneous magnitude values).

Table 2 lists the variables we use in this work ('Seq.' denotes whether the variable is sequential or not; 'Der.' denotes that a variable is derived). Our data includes several other variables such as weather, hydration, humidity, calories, cadence, etc., though these variables are much sparser than the variables listed (some are only available for certain sport types, or from certain wearables); we leave the exploration of these data for future work.

We discard abnormal data points based on simple rules (e.g. overly large magnitude, mismatching timestamps, abrupt changes in GPS coordinates).[2] We further keep users with at least 10 workout records to make sure there is enough data to evaluate our personalization strategies. Table 3 summarizes our dataset. The original dataset contains 253,020 workout records for 1,104 users.[3] After filtering, there are 167,373 workout records for 965 users. On average each user has over 175 workout records that cover an average span of around 2 years. This makes the dataset promising for the purpose of studying long-term temporal variation among users.

[2]Which may occur due to sensor error, or more simply due to a user forgetting to disable the device after returning to their car (for example).

[3]We collected a contiguous block of workouts, after which we collected the complete workout histories for all users represented in that block.

Table 2: Description of variables

	Variable	Seq.	Unit
Measure-ment	Heart Rate	✓	Beat per Minute (BPM)
	Timestamp	✓	Unix Timestamp
	Distance	✓	Mile
	Speed	✓	Mile per Hour (MPH)
Context-ual	UserID	✗	/
	Sport	✗	/
	Gender	✗	Male, Female
	Altitude	✓	Meter
	Longitude	✓	Degree
	Latitude	✓	Degree
Derived	Der. Distance	✓	Kilometer (KM)
	Der. Speed	✓	Kilometer per Hour (KMPH)

Table 3: Endomondo dataset statistics

Statistic	Original	Filtered	Re-sampled
# of workouts	253,020	167,373	102,343
# of users	1,104	956	887
Avg. length (hours)	5.998	1.486	1.059
Avg. span (days)	/	766	733

The original sequences as collected from *Endomondo* each contain $L = 500$ data points which are not necessarily sampled in fixed-width intervals. The sampling interval may vary from seconds to minutes. Rather than considering the problem of imputing missing values across long intervals, we simply discard workouts with sampling intervals greater than 1 minute. Furthermore, the workout data are truncated to 450 time points (out of an original 500) to account for the fact that many of the workouts tend to complete before the full 500 time-steps (e.g. during a user's cooldown).

For the problem of short term prediction, it is important to have a version of the dataset with a fixed sampling interval (i.e., so that our predictions always correspond to a fixed amount of seconds in the future). To this end, we produce a second version of the dataset that is re-sampled into fixed intervals using cubic spline interpolation followed by resampling in 10-second intervals. We discard workouts in cases where resampling generates out-of-bounds values. After resampling, we further discard those workouts with fewer than 300 samples (i.e., 300×10 seconds). Finally, we maintain a subset of 102,343 workout records as shown in table 3.

Our data are highly multi-modally distributed, as the relationship between variables for one user differs significantly from that of other users, and the structure of the data for one sport differs significantly from that of other sports (e.g. bicycling uphill vs. downhill results in quite different heart rate patterns than running uphill vs. downhill). Our experiments show that the proposed models are able to learn these complex relations among variables.

Table 4: Parameter settings in our experiments

Model	*FitRec*	*FitRec-Attn*
Hidden Size	64	64
Attribute embed. size	5	5
Contextual embed. size	64	64
Dropout	0.2	0.1
L2 Regularizer	0.01	0.002
Batch Size	512	5120
Learning Rate	0.001	0.005

4.2 Training Procedure

We sort each user's workouts in chronological order based on the first timestamp of each workout. Then we split and aggregate users' first 80% of workouts for training, the next 10% for validation and the final 10% for testing. This is to make sure the contextual information of a workout in the test set does not appear in the training set. During training, the target data are not scaled while the input data are scaled by representing the data points in terms of their z-scores.

We used Keras[4] and Pytorch[5] to implement our models. Our models are trained using Adam [14]. We set the size of the LSTM hidden state, attribute embedding and contextual embedding as 64, 5, and 64, respectively. Attribute embeddings are randomly initialized and learned from scratch. We add dropout before and after each LSTM layer and an ℓ_2 regularizer on attribute embeddings and contextual embeddings.

The best hyper-parameters are obtained via grid search on the validation set. The learning rate is chosen from {0.001,0.005,0.01}, the dropout rate from {0.1,0.2,0.3}, and our ℓ_2 regularizer from {0.002, 0.005,0.01, 0.02}. The detailed parameter settings are as shown in table 4. Training is terminated after 50 epochs. The model that achieved the best evaluation score on the validation set was run on the test data to obtain the final performance score. To train the models with contextual embeddings, we found it more stable to use a pre-training technique. Specifically, we first initialize the model parameters using the best model without contexutal embeddings, then continue training until convergence. All code and data are available publicly.[6]

4.3 Evaluation Metrics

We choose two popular metrics in sequential modeling: the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) to evaluate performance on our two prediction tasks:

$$\text{RMSE} = \sqrt{\frac{1}{|N_{test}|} \sum_{\mathbf{y} \in \mathcal{T}_{test}} \sum_{t=1}^{L} (y_t - \hat{y_t})^2} \tag{13}$$

$$\text{MAE} = \frac{1}{|N_{test}|} \sum_{\mathbf{y} \in \mathcal{T}_{test}} \sum_{t=1}^{L} |y_t - \hat{y_t}|, \tag{14}$$

where $|N_{test}|$ is the number of total time steps of all workouts on the test set $\mathcal{T}_{test}$. Note that in our setting the two metrics have quite different semantics. For example, a model which correctly identified heart rate 'trends' but was shifted in time or scale might be considered an acceptable model; such predictions would incur a high MSE but a more moderate MAE.

[4]https://keras.io/
[5]https://pytorch.org/
[6]https://github.com/nijianmo/fit-rec

Table 5: Workout profile forecasting on speed and heart rate. *: significantly better than the second best score ($p < 0.05$).

	Speed (KMPH)		Heart rate (BPM)	
Model	RMSE	MAE	RMSE	MAE
Global mean (train)	11.997	8.532	24.302	18.185
User mean (train)	10.690	6.418	20.117	15.090
MLP	8.998	4.411	18.256	13.918
FitRec (U)	7.144	2.472	18.784	14.142
FitRec (U/S)	7.073	**2.381**	17.325	13.012
FitRec (U/S/G)	7.328	2.396	18.207	13.831
FitRec (U/S/C)	**7.025***	2.384	**17.051***	**12.847***

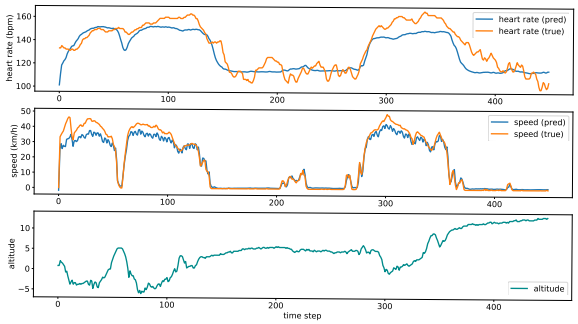
4.4 Results: Workout profile forecasting

We compare *FitRec* to two baselines and also conduct ablation tests. Our baselines are:

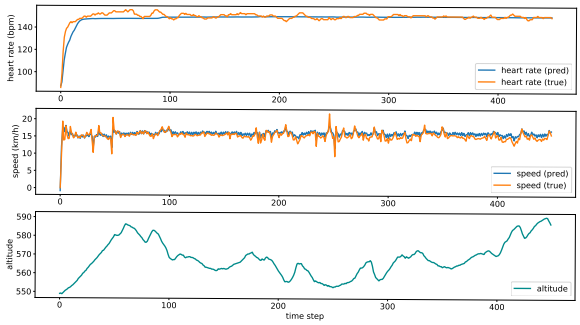
- **Global mean**. Predict the target variable at each time step as the global mean of the training set.
- **User mean**. For each workout from a user u, predict the target variable at each time step as the mean value of all u's sequences in the training set. This serves as a naïve personalized baseline.
- **MLP**. A variant of a traditional Multilayer Perceptron that concatenates the learned attribute embeddings, contextual embeddings and contextual inputs together as its input. The MLP predicts the target variable at each time step simultaneously. The dimension of the MLP's output layer is the total length of the sequence. This baseline is used to demonstrate that the LSTM component of *FitRec* is better at modeling these contextual inputs compared to an MLP.
- **FitRec** (U). *FitRec* model with user embedding only.
- **FitRec** (U/S). *FitRec* model with user embedding and sport embedding.
- **FitRec** (U/S/G). *FitRec* model with user embedding, sport embedding and gender embedding. This model is used to measure whether the gender embedding improves *FitRec*.
- **FitRec** (U/S/C). *FitRec* model with user embedding, sport embedding and contextual embedding.

We use *FitRec* to predict two types of workout profile: (1) speed and (2) heart rate, given the expected workout time and a specified route. In particular, the route is given as a distance sequence and an altitude sequence along the complete workout.

As shown in table 5, *FitRec* (U/S/C) with all embeddings outperforms each of the traditional baselines. Specifically, it decreases the RMSE of the best baseline (MLP) by 21.93% and 6.6% on speed prediction and heart rate prediction, respectively. The user mean baseline achieves lower RMSE and MAE than the global mean baseline, demonstrating the need for personalized models for this task. *FitRec* (U) with user embeddings achieves further improvements over the user mean baseline, which shows the efficacy of the learned embeddings in capturing individual patterns of users.



(a) Biking



(b) Running

Figure 4: Examples of predicted heart rate for different sports, altitude has much greater influence on heart rate and speed change when biking as compared to running (for example).

Furthermore, our ablation tests exhibit improvements due to the attribute embeddings and the contextual embedding. *FitRec* (U/S/C) outperforms *FitRec* (U) and *FitRec* (U/S), except for the MAE on speed prediction. This may be because speed is sensitive to sport type, and the previous workout that we use to learn the contextual embedding has a different sport type from the current workout; in such cases the contextual embedding may not be informative. Missing data issues aside, this model could, in the future, be straightforwardly extended to consider other attributes such as weather, humidity, etc.

It is also surprising to find out that an auxiliary gender embedding does not help improve the prediction accuracy of our model: *FitRec* (U/S/G) performs worse than *FitRec* (U/S). There are a few possible reasons. One is that our data includes 863 males, 80 females and 13 unknown users. Such an imbalance in data might make it difficult to learn patterns across different genders. Another possibility is that the user embedding might already encode the user's gender information. As such, an auxiliary gender embedding does not bring any further benefit to the model.

Figure 4 gives examples of the predicted heart rate and speed for biking and running instances. In both cases the predicted sequences closely match the trend of the ground-truth during the workout. Notably, *FitRec* is able to learn the semantics of different sport types. For example, entering a downhill portion of a bike ride causes a user's heart rate to drop substantially, whereas running downhill does not result in a heart rate reduction.

Table 6: Short term heart rate prediction. *: significantly better than the second best score ($p < 0.05$).

Model	Heart rate (BPM) RMSE	MAE
Windowed MLP	3.035	1.919
Seq2Seq+A	2.807	1.720
DA-RNN	2.816	1.733
FitRec-Attn (U)	2.795	1.705
FitRec-Attn (U/C)	**2.783***	**1.695***

4.5 Results: Short term prediction

For short term prediction, we consider three baselines for auto-regressive modeling and conduct ablation tests to compare against the performance of *FitRec-Attn*.

The baselines are:

- **Windowed MLP** [10]. A conventional time series model that considers a sliding window of the target variable's previous values, and predicts the target variable at the current time step.
- **Seq2Seq+A** [6]. A competitive encoder-decoder baseline that is widely used in sequential modeling. This model includes temporal attention between the encoder and decoder.
- **DA-RNN** [25]. A state-of-the-art attention based RNN model for auto-regressive time series modeling with exogenous variables. The model includes an input attention mechanism that assigns weights to each input variable as well as temporal attention.
- **FitRec-Attn** (U). *FitRec-Attn* model with user embedding only.
- **FitRec-Attn** (U/C). *FitRec-Attn* model with user embedding and contextual embedding.

Because our ablation tests showed that the gender embeddings did not improve results for short term prediction, we did not include them in this section.

We use *FitRec-Attn* to predict the heart rate at each time step t given the speed and altitude sequences, the previous $T-1$ heart rate values, as well as the attribute embeddings and contextual embedding. We choose $T = 10$ in the experiments. We also experimented with $T = 5, 15, 20$ and found that values greater than 10 yield marginal improvement. When implementing Seq2Seq+A and DA-RNN, we only use speed and altitude sequences since they do not include the embedding modules for user and contextual embeddings. The baseline windowed MLP only considers the previous $T-1$ heart rate values as in traditional autoregressive models.

As shown in table 6, *FitRec-Attn* (U/C) achieves the best performance compared with the other baselines. When only considering speed and altitude as inputs, attention-based Seq2Seq outperforms the more sophisticated DA-RNN. This may be because that the number of input variables is relatively small (2 in this case), leaving the input attention of DA-RNN as but a source of extraneous complexity. On the other hand, by incorporating user embeddings and contextual embeddings, *FitRec-Attn* (U/C) improves results on both metrics compared to DA-RNN. This demonstrates the effectiveness of input attention when the input variable is high dimensional.

Table 7: Route recommendation performance

	AUC	Hit@10	NDCG
User mean (train)	0.500	0.100	/
MLP	0.506	0.106	0.212
FitRec (U)	0.639	0.214	0.262
FitRec (U/S)	0.643	0.234	0.266
FitRec (U/S/C)	**0.673**	**0.272**	**0.284**

5 PERSONALIZED RECOMMENDATION

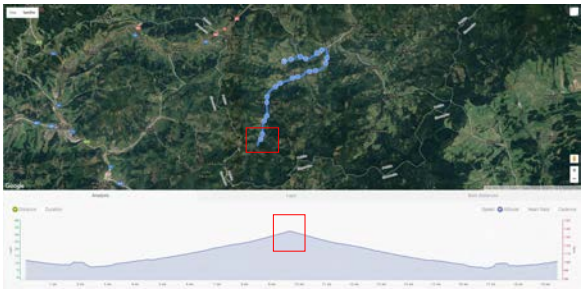
So far we have shown the effectiveness of our model at generating 'black box' heart rate predictions given contextual inputs. In this section we illustrate the ability of *FitRec* to provide personalized recommendations in practice. We consider the following two recommendation tasks:

- Workout route recommendation: Given expected workout criteria (e.g. an expected workout time, an idiosyncratic heart rate or speed curve), we suggest routes (i.e. historical routes from all users) that will match the user's expectation. This might be applied if a user has particular heart rate targets in mind, or simply wishes to identify alternate routes that are 'similar to' (in terms of heart rate profile) a route a user has taken previously. Such a system could be used (for example) by a user who is traveling but wants to maintain their regular exercise routine by identifying qualitatively similar routes.
- Short term heart rate prediction: Predict whether a user's heart rate will exceed some threshold if they continue at the current pace during their workout. This might be used to build a personalized assistant that guides a user's behavior (e.g. speed up or slow down) during a workout.

5.1 Route Recommendation

5.1.1 Evaluation Methodology. We evaluate our route recommendation task in terms of ranking performance. To do so, we first use *FitRec* to generate a ranked list of candidate routes, given a user's expected workout criteria. Then, for each workout in the test set (positive sample), we consider its route (i.e. distance and altitude sequences) as the positive route, and randomly sample another 100 workouts as negative routes. That is, given target workout criteria (heart rate etc.) as input, we are evaluating the model's ability to correctly identify the real workout that generated these criteria by assigning it a higher score than randomly chosen alternatives. Given these positive and negative routes, *FitRec* generates a predicted heart rate sequence for each of them. The similarity between the predicted sequences and the ground-truth heart rate sequence is calculated using fast dynamic time warping (fastDTW) [27]. We use DTW to measure similarity rather than the ℓ_2 loss because it is less sensitive to temporal shifting.

After sorting these similarity scores in descending order, we obtain the ranked list of all routes, and are interested in the ranked position of the positive route amongst the negative routes. Here we consider three recommendation metrics that are commonly used in ranking-based recommendation [12]: AUC, Hit Rate@K (Hit@K) and Normalized Discounted Cumulative Gain (NDCG). AUC accounts for the overall ranking of the positive route, Hit@K measures whether the positive route appears in the top K of the list, and the NDCG uses a graded relevance criterion to evaluate the overall quality of the ranking.



(a) Map of the ground-truth route.



(b) Map of the recommended alternate route.

Figure 5: Examples of a recommended route. The model recommends an alternate route that is similar to the ground-truth (i.e., both routes are comparable hill-climbs).

5.1.2 Performance Analysis. Table 7 shows the performance of route recommendation on the test set. We find that baseline models exhibit poor performance on personalized route modeling: (1) The user mean baseline does not take into account the route (i.e., distance, altitude) information; and (2) the MLP model considers routes as input during learning, but fails to rank the positive routes higher than other routes (its AUC of 0.506 is close to that of the User Mean baseline). On the other hand, *FitRec* with user embeddings is able to recognize the difference between routes and outperforms baseline models in all metrics. After adding the attribute embeddings and contextual embedding, the model's performance is further improved.

Figure 5 presents an example of a recommended route given by our model (i.e., the route ranked first on the ranked list, other than the ground truth) along with the ground-truth route. As the route map shows (marked in blue),[7] both the ground truth route and the recommended route are 'hill climbs' of approximately the same length and difficulty (the red bounding box in each route map is the region with the greatest altitude during the workout). This shows the effectiveness of the model to recommend alternate but semantically similar routes.

[7] Snapshot from *endomondo.com*

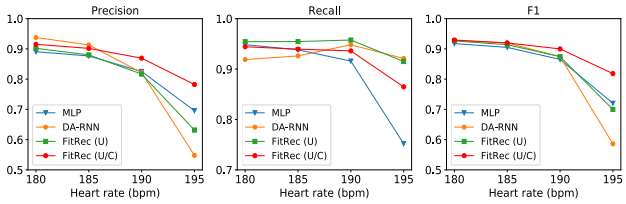


Figure 6: Performance of short-term excessive heart rate prediction

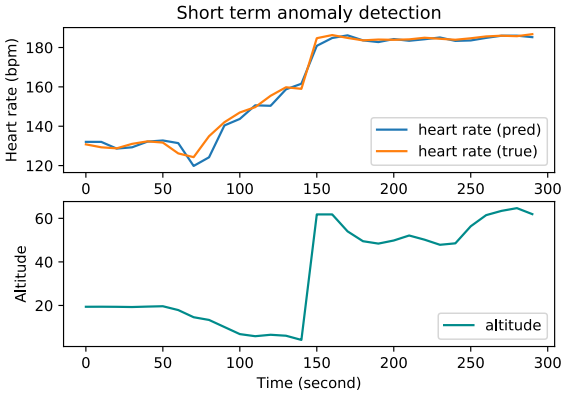


Figure 7: Example of short term excessive heart rate prediction.

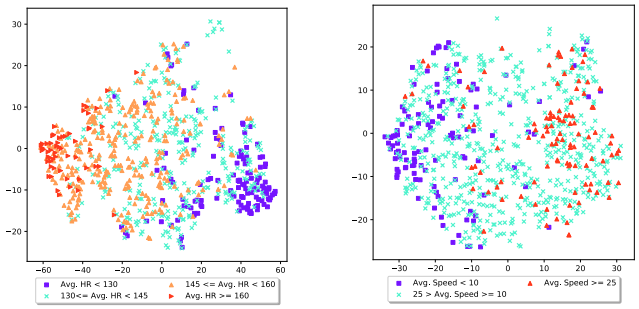
5.2 Short term heart rate prediction

The goal of this task is to predict whether a user's heart rate will exceed some level if the user continues their workout at the current speed. We assume there is a threshold value predefined by each user that she/he considers to be a desirable maximum heart rate. We consider four different threshold values: 180, 185, 190, 195 and compute three metrics to demonstrate the effectiveness of our model to anticipate rapid heart rates in test sequences.

5.2.1 Evaluation Methodology. For each workout we construct two binary sequences $(b_1, \ldots, b_L)$ and $(b_1^*, \ldots, b_L^*)$ (of predictions), where b_t / b_t^* equals 1 if the groundtruth / predicted (respectively) heart rate at timestep t exceeds a pre-defined threshold (zero otherwise). We then calculate Precision, Recall, and Macro-F1 scores based on these two binary arrays to evaluate the performance of predicting whether the speed will exceed the threshold.

5.2.2 Performance Analysis. Figure 6 demonstrates the performance of *FitRec-Attn* when given different threshold values. As we can see, *FitRec-Attn* (U/C) achieves a better F1-score compared to other baseline models for all threshold values. Specifically, the improvement of *FitRec-Attn* increases as the threshold value increases.

Figure 7 shows an example of short term heart rate prediction. Note that for this task, the model takes as input the observed heart rate values for the previous $T-1$ time steps; therefore the prediction is more accurate than in our previous experiment. In the figure we see an abrupt change in altitude, following which the model successfully predicts the heart rate will reach a high value (i.e. over 185 BPM).



(a) User embedding learned for heart rate prediction.

(b) User embedding learned for speed prediction.

Figure 8: 2d t-SNE of the learned user embedding, and its relationship to user segments in terms of heart rate and speed.

5.3 What have the user embeddings learned?

To qualitatively explain what has been captured by the learned user embeddings, we use t-SNE [22] to project them into low-dimensional vectors. Figures 8a and 8b show the 2d t-SNE of the user embeddings learned for the tasks of predicting heart rate and speed, respectively.

For heart rate, we calculate the average heart rate of each user based on all of their workouts. We classify them into four categories by threshold values: < 130, < 145, < 160, and ≥ 160 (BPM). Similarly for speed, we calculate the average speed of each user for all of their workouts. We classify them into three categories by thresholds: < 10, < 25, and ≥ 25 (KM per Hour). As shown in figs. 8a and 8b, users that fall into the same class (i.e., with similar average heart rate and average speed) seem to have closer user embeddings.

6 CONCLUSIONS

We present *FitRec* – a system that addresses sequential prediction problems in fitness and exercise data and can serve as a building block for developing personalized fitness recommendation applications. We achieve this by learning embedded representations from auxiliary information such as user identity, sport type and historical workout sequences, each of which is incorporated into an LSTM based sequential modeling framework. We verify that the model is capable of providing high quality prediction on tasks centered around estimating users' heart rate sequences, such as workout profile forecasting and short term prediction. We conduct extensive experiments on a new real-world dataset and demonstrate that the model outperforms other baselines without auxiliary embedded representations consistently on both tasks. Furthermore, the model shows promising performance on recommendation tasks such as recommending alternate workout routes and short-term excessive heart rate prediction. Visualizations also show that the learned embedded representations are useful for identifying distinct user segments. In the future, we are interested in applying *FitRec* to recommendation tasks with more complex scenarios (i.e. safety-aware route recommendation) and incorporating other auxiliary data (i.e. interactive recommendation).

REFERENCES

[1] Tim Althoff. 2017. Population-Scale Pervasive Health. *IEEE Pervasive Computing* 16 (2017), 75–79.

[2] Tim Althoff, Rok Sosic, Jennifer L. Hicks, Abby C. King, Scott L. Delp, and Jure Leskovec. 2017. Large-scale physical activity data reveal worldwide activity inequality. In *Nature*.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014).

[4] Hadi Banaee, Mobyen Uddin Ahmed, and Amy Loutfi. 2013. Data Mining for Wearable Sensors in Health Monitoring Systems: A Review of Recent Trends and Challenges. In *Sensors*.

[5] Zhengping Che, Xinran He, Ke Xu, and Yan Liu. 2017. DECADE: A Deep Metric Learning Model for Multivariate Time Series. In *NIPS-ML4H*.

[6] Kyunghyun Cho, Bart van Merrienboer, ÃĞaglar GülÃğehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*.

[7] Alok Kumar Chowdhury, Aleksandr Farseev, Prithwi Raj Chakraborty, Dian Tjondronegoro, and Vinod Chandran. 2017. Automatic classification of physical exercises from wearable sensors using small dataset from non-laboratory settings. *2017 IEEE Life Sciences Conference (LSC)* (2017), 111–114.

[8] Miguel Ramos de Araujo, Pedro Manuel Pinto Ribeiro, and Christos Faloutsos. 2017. TensorCast: Forecasting with Context Using Coupled Tensors (Best Paper Award). In *ICDM*.

[9] Aleksandr Farseev and Tat-Seng Chua. 2017. TweetFit: Fusing Multiple Social Media and Sensor Data for Wellness Profile Learning. In *AAAI*.

[10] Ray J. Frank, Neil Davey, and Stephen P. Hunt. 2001. Time Series Prediction and Neural Networks. *Journal of Intelligent and Robotic Systems* 31 (2001), 91–103.

[11] Tian Guo, Tao Lin, and Yao Lu. 2018. An interpretable LSTM neural network for autoregressive exogenous model. *CoRR* abs/1804.05251 (2018).

[12] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. TriRank: Review-aware Explainable Recommendation by Modeling Aspects. In *CIKM*.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural computation* 9 8 (1997), 1735–80.

[14] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[15] Timo Koskela, Mikko Lehtokangas, Jukka Saarinen, and Kimmo Kaski. 1996. Time Series Prediction with Multilayer Perceptron, FIR and Elman Neural Networks. In *World Congress on Neural Networks*.

[16] Takeshi Kurashima, Tim Althoff, and Jure Leskovec. 2018. Modeling Interdependent and Periodic Real-World Action Sequences. *WWW* 2018 (2018), 803–812.

[17] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. In *SIGIR*.

[18] Xiaohan Li, Shu Wu, and Liang Wang. 2017. Blood Pressure Prediction via Recurrent Models with Contextual Layer. In *WWW*.

[19] Zhiyuan Lin, Tim Althoff, and Jure Leskovec. 2018. I'll Be Back: On the Multiple Lives of Users of a Mobile Activity Tracking Application. *WWW* 2018 (2018), 1501–1511.

[20] Zachary Chase Lipton. 2015. A Critical Review of Recurrent Neural Networks for Sequence Learning. *CoRR* abs/1506.00019 (2015).

[21] Benedikt Loepp and JÃijrgen Ziegler. 2018. Recommending Running Routes: Framework and Demonstrator. In *Workshop on Recommendation in Complex Scenarios*.

[22] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 9, Nov (2008), 2579–2605.

[23] Emma Pierson, Tim Althoff, and Jure Leskovec. 2018. Modeling Individual Cyclic Variation in Human Behavior. In *WWW*.

[24] Soujanya Poria, Erik Cambria, Devamanyu Hazarika, Navonil Majumder, Amir Zadeh, and Louis-Philippe Morency. 2017. Multi-level Multiple Attentions for Contextual Multimodal Sentiment Analysis. In *ICDM*.

[25] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison W. Cottrell. 2017. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. In *IJCAI*.

[26] C Carl Robusto. 1957. The cosine-haversine formula. *The American Mathematical Monthly* 64, 1 (1957), 38–40.

[27] Stan Salvador and Philip Chan. 2007. Toward accurate dynamic time warping in linear time and space. *Intell. Data Anal.* 11 (2007), 561–580.

[28] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent Neural Network Regularization. *CoRR* abs/1409.2329 (2014).

Generating Product Descriptions from User Reviews

Slava Novgorodov and Ido Guy
eBay Research
Netanya, Israel
snovgorodov@ebay.com,idoguy@acm.org

Guy Elad and Kira Radinsky
Technion, Israel Institute of Technology
Haifa, Israel
{sguyelad,kirar}@cs.technion.ac.il

ABSTRACT

Product descriptions play an important role in the e-commerce ecosystem, conveying to buyers information about a merchandise they may purchase. Yet, on leading e-commerce websites, with high volumes of new items offered for sale every day, product descriptions are often lacking or missing altogether. Moreover, many descriptions include information that holds little value and sometimes even disrupts buyers, in an attempt to draw attention and purchases. In this work, we suggest to mitigate these issues by generating short crowd-based product descriptions from user reviews. We apply an extractive approach, where review sentences are used in their original form to compose the product description. At the core of our method is a supervised approach to identify candidate review sentences suitable to be used as part of a description. Our analysis, based on data from both the Fashion and Motors domains, reveals the top reasons for review sentences being unsuitable for the product's description and these are used, in turn, as part of a deep multi-task learning architecture. We then diversify the set of candidates by removing redundancies and, at the final step, select the top candidates to be included in the description. We compare different methods for each step and also conduct an end-to-end evaluation, based on rating from professional annotators, showing the generated descriptions are of high quality.

CCS CONCEPTS

• **Information systems → Electronic commerce**; **Online shopping**; • **Applied computing → Electronic commerce**; • **Computing methodologies →** *Natural language generation*; *Multi-task learning*.

KEYWORDS

Deep multi-task leaning; electronic commerce; language generation; user-generated content.

ACM Reference Format:
Slava Novgorodov and Ido Guy and Guy Elad and Kira Radinsky. 2019. Generating Product Descriptions from User Reviews. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313532

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW 2019, May 13-17, 2019, San Francisco, California
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313532

1 INTRODUCTION

The importance of content on e-commerce websites has been widely recognized. High-quality and trusted product content has been empirically shown to have a substantial influence on user behavior, which is manifested in conversion rates and the volume of sales [28, 35, 39]. Product descriptions are an important element of the content displayed on product pages, alongside the product's title, image, and key attributes, such as model name, color, or size. Yet, such descriptions are often lacking or missing; for example, the majority of the Fashion products on eBay have no description at all. Even when available, product descriptions are often long and tedious to read, containing a lot of information that is insignificant for potential buyers. Our own analysis indicates that substantial portions of the product description sentences include details specific to a single listing or seller, information about the brand as a whole, and pure marketing statements.

We refer to a *product description* as a written (textual) presentation of what the product is, how it can be used, and why it is worth purchasing. The purpose of a product description is to provide customers with details about the features and benefits of the product so they are compelled to buy.[1] In line with previous research on e-commerce content, we expect a good description to be informative, readable, objective, and relevant to the product (e.g., as opposed to a specific listing or a whole brand) [58]. We focus on concise descriptions of several sentences, which can be quickly consumed in their entirety and are especially suitable for small-screen devices, as e-commerce mobile applications have seen a remarkable growth and account for a major portion of the overall e-commerce traffic [29, 31]. Like other types of product content, credible descriptions have been shown to increase sales, while lacking descriptions withhold users from reaching a purchase decision or effectively searching for products [36].

In light of the aforementioned challenges, we propose to use the "crowd" to generate trustworthy product descriptions, by leveraging the products' user reviews [15, 27, 38]. Such reviews are often abundant on e-commerce websites and reflect the perspective of those who have already purchased the product. Therefore, potential buyers tend to trust reviews much more than they trust seller-provided content [30, 59].

User reviews primarily aim at reflecting a buyer's subjective perspective and include personal opinions, stories, experiences, and complaints, which are not suitable to include in a product description (e.g., "*My old shoes wore down and I needed a new pair*" or "*I can plug it to each of my three cars*"). The large volumes of user reviews accumulated for popular products, with each review typically containing multiple sentences, makes them practically

[1] This definition is largely based on that by the Shopify e-commerce platform: https://www.shopify.com

impossible to consume. As a result, users often read only a few reviews and may miss helpful information that appears in others. We observe that some portion of the review sentences are descriptive of the product [15], and suggest an extractive approach to generate *crowd-based* descriptions by combining original review sentences.

The transformation from reviews to descriptions is a challenging task, which to the best of our knowledge, is novel. While reviews aim to reflect the buyer's perspective, descriptions typically reflect the viewpoint of the seller. Moreover, while reviews are meant to reflect a variety of subjective opinions, descriptions are expected to provide objective fact-based information. Several prior studies have examined review summarization (e.g., [26, 45, 71]), however such summaries do not necessarily contain descriptions of the product.

Our extraction of candidate sentences from user reviews, to be included in the product depiction, is primarily supervised. We examine both classic machine learning models and deep learning approaches for the classification task, trained over thousands of sentences in two key e-commerce domains: Fashion and Motors. We also analyze the key reasons making review sentences unsuitable for a description. A deep multi-task learning classifier, which is based on mapping the top reasons to auxiliary tasks, is found to yield the best performance for the candidate identification task.

Following, we select the top sentences out of the candidate set for the final product description. To this end, we use a sentence similarity measure that helps diversify and avoid redundancies. Semantic similarity based on word embedding is found to be more effective than a bag-of-words approach. We experiment with several basic methods to produce the final description, which rely on the classification score from the candidate extraction process and the similarity measure. We perform a large-scale evaluation of the descriptions based on thousands of ratings from professional annotators, comparing the different methods and inspecting three description lengths: 3, 5, and 7 sentences.

The main contributions of this work can be summarized as follows:

- To the best of our knowledge, this is the first work to suggest the extraction of product descriptions from reviews.
- We provide analysis and examples of how review sentences can be used for descriptions (what portion, which kind of sentences) across two principal yet very different e-commerce domains.
- We develop a classifier for identifying review sentences suitable for product descriptions, reaching an AUC of over 0.92.
- We present an end-to-end system for description generation from reviews, comparing different approaches for sentence selection, reaching an average rating of 4.3 (out of 5) per description.

2 RELATED WORK

Textual product descriptions have been explored in the e-commerce literature along with other seller-provided product content types, such as titles [19, 52], images [6, 20], and attributes [52, 55]. Some of the studies refer to the product description in a broad sense, which encompasses the other content types, e.g., the list of structured attributes [2], title [66], or product image [48]. Other studies refer to a product description similarly to us, as the textual writeup that extends the title and attributes. Probst et al. [57] studied the extraction of attribute-value pairs from such product descriptions, in order to enrich the product's structured representation, used for tasks such as recommendation and matching. Shinzato and Sekine [62] proposed an unsupervised approach for the same task. Dumitru et al. [11] applied text mining and clustering techniques over product descriptions in order to recommend product features for a given domain. In a recent study, Pryzant et al. [58] showed that product descriptions of the type we study can help predict the product's business outcome. Experimentation was based on product descriptions and sales records from the Rakuten Japanese e-commerce website. None of these studies, however, provided a definition of a product description as presented in this work.

As mentioned in the Introduction, a related task to the one we explore is review summarization. Different from standard text summarization [18], where the goal is to generate a concise summary for a single [64] or multi-document [40], review summarization aims at extracting and summarizing opinions about a product from multiple reviews [45, 70]. Most studies focus on identifying the key attributes of an entity, such as a product, a movie, or a hotel, and then extracting key phrases that describe these attributes or the sentiment towards them (e.g., [26, 37, 56, 73]). Techniques used for this type of summarization include rule-based mining [42], topic modeling [51, 67], and neural networks [46, 68]. An overview of review summarization techniques can be found in several surveys [32, 41, 43, 54].

While summarization seeks a good coverage of the main topics within the set of reviews, sometimes revolving around key product attributes, we aim to identify a unique subset of the reviews' content that is descriptive of the product . A sentence that may be pivotal to the set of reviews, and thereby for its summary, might not be appropriate for a description. For example, the sentence "*I like this hat very much*" may be included in a review summary, but poses low value for a product description, as it is subjective and provides little information about the product. In our experiments, described later in this paper, we found that fewer than 10% of the review sentences were suitable "as is" to take part in the product description.

Another related body of research has focused on extracting experiences [50, 53] and tips [21, 72] from user reviews. Somewhat similarly to the motivation presented in this work, these studies aim at helping users sift through the large volumes of reviews by identifying a more specific type of information within the reviews. Nonetheless, extracting experiences and tips is each inherently different than extracting descriptive sentences: experiences are subjective in nature, reflecting the unique viewpoint of an individual user (or group) and are thus not suitable, almost by definition, to be part of a product description. Tips, on the other hand, are defined as concrete and typically actionable pieces of advice. Therefore, their extraction actually aims at excluding purely descriptive sentences, of the type pursued in this work.

3 DATASETS AND CHARACTERISTICS

In this section, we describe the datasets used for our analysis and experimentation and their characteristics.

Datasets. Our research is based on two product datasets from two principal yet very different e-commerce domains: Fashion (clothing, shoes, and jewelry) and Motors (automotive parts and

Table 1: Fashion and Motors dataset characteristics.

	Fashion				Motors			
	Avg	Std	Median	Max	Avg	Std	Median	Max
Reviews per product	1118	1371	596	8271	883	985	378	7352
Sentences per review	3.71	3.09	3	103	4.05	3.73	3	98
Words per sentence	8.04	6.43	6	58	9.24	7.26	7	63
Number of products	892				807			
Number of reviews	997,274				712,904			

accessories). Both datasets were obtained from a large e-commerce website in the United States, representing best-selling products in each of the two domains.[2] The datasets contain, per product, both its description and user reviews. Table 1 presents the characteristics of the two datasets. The number of products is rather similar in both datasets, while for Fashion there are more reviews per product and for Motors the number of sentences per review and their length in words is slightly higher. In addition, we used two larger datasets with over 10 million reviews of over 10,000 best-selling products in each domain (Fashion and Motors), for pre-training word embeddings, as will be described later in this paper.

Data Annotation. Labeling for training and evaluation in this work was performed by in-house professional editors (annotators), with domain expertise in both Fashion and Motors. The pool included a total of 20 editors, of whom different subsets were selected for different tasks, proportionally to the task's size. Unless otherwise stated, each evaluation was performed by a single editor.

Description Characteristics. The descriptions in our dataset are substantially longer than those we aim to generate. The average number of sentences per description in the Fashion dataset is 28.2 (std: 12.9, median: 27, min: 11, max: 65) and for motors it is 26.8 (std: 12.1, median: 29, min: 9, max: 68). To get a preliminary sense of the content of these descriptions, 50 descriptions from each dataset were annotated by two professional annotators. Only 45% of the sentences were labeled as suitable for a product description, with the key reasons for the sentences not being adequate including purely subjective marketing statements, accounting for slightly over 20% of the sentences (e.g., "*Give your clothes the luxury they deserve with these wonderful hangers!*"); information specific to a seller or a listing (18%; "*1 year limited warranty*"), and description of the brand as a whole (15%; "For over 35 years, we have been one of the largest sunglasses brands.")

Reviews vs. Descriptions. In essence, reviews and descriptions hold fundamentally contrasting characteristics: reviews reflect a subjective opinion based on an individual experience, while descriptions are expected to be "dryer", explaining what the product is and why it is worth purchasing. As a first step, we set out to examine the most prominent language differences between reviews and descriptions. To this end, we used Kullback-Leibler (KL) divergence, which is a non-symmetric distance measure between two given distributions [4]. Specifically, we calculated the terms, per each of the two domains, which contribute the most to the KL divergence between the language model of the reviews versus the language model of the descriptions and vice versa [22].

[2]As of July 2018.

Table 2: Most distinctive unigrams for reviews vs. descriptions ('Reviews') and vice versa ('Descriptions') in Fashion and Motors.

Fashion		Motors	
Reviews	Descriptions	Reviews	Descriptions
i	your	i	power
my	designed	my	generator
it	you	it	protection
they	features	was	your
was	comfort	great	fuel
but	jewelry	but	torqx
very	cotton	have	portable
great	imported	so	provides
good	inches	works	uego
these	fashion	this	economy
them	polyester	good	watts
bought	technology	they	designed
me	provides	very	advanced

Table 2 presents the most distinctive unigrams. It can be seen that the unigram list most characterizing reviews (relative to descriptions) is rather similar between Fashion and Motors. The first-person pronouns 'i' and 'my' are at the top of the two lists, both common on reviews yet hardly occurring on descriptions. For example, 'i' occurs on 3.43% of the Fashion review sentences and 3.11% of the Motors review sentences, whereas for descriptions it occurs on 0.03% and none of the sentences for both datasets, respectively. Other prominent unigrams on the review lists include 'was', which typically reflects a past-tense experience ('bought' can also be observed on the Fashion list); the third-person pronouns 'it' and 'they'; the adjectives 'good' and 'great', which often reflect a subjective opinion; and the emphasizing adverb 'very'. Further down the list, beyond Table 2, we also encountered unigrams that refer to aspects of a specific listing of the product, which may vary from one seller to another, such as 'price', 'cheap', and 'shipping'; other third-person references such as 'she' or 'son'; and verbs that reflect subjectivity ('recommend', 'like', 'love') or past tense ('arrived', 'purchased', 'got', 'ordered').

The most characterizing unigrams for descriptions relative to reviews are more dissimilar between Fashion and Motors. As can be seen in Table 2, each list includes its own domain-specific descriptive words, such as 'jewelry', and 'cotton' for Fashion, or 'power' and 'fuel' for Motors. The second-person pronoun 'your' ranks high on both lists, indicating that while first- and third-person language is used almost exclusively on reviews, second-person language is more characteristic of descriptions (e.g., 'your' occurs on 1.34% of the Fashion description sentences versus only 0.18% of the review sentences). The only other common words between the two lists are 'provides' and 'designed'.

Overall, the above analysis gives an indication of the key differences in the language of reviews versus descriptions. Following, review sentences to be used as part of a description need to be carefully selected. We elaborate on this process in the next section.

4 CANDIDATE SENTENCE EXTRACTION

In this section, we describe a key component of our description generation method: the extraction of candidate review sentences that can be used for the product's description. Given the set of all user reviews for the product, our goal is to identify a set of sentences

that can be used in their original form for a description of the same product. We first apply rule-based filtering based on the analysis presented in the previous section. We then apply a supervised approach that learns to identify review sentences suitable for a description. We examine different types of classifiers for this task and compare their performance based on a large labeled set of review sentences.

4.1 Rule-Based Filtering

Considering our analysis of linguistic differences between descriptions and reviews, we established several simple rules to identify review sentences that cannot be used as part of a description: (1) **Short**: sentences of 3 words or fewer generally introduce little information and do not flow well as part of a product description. For example, "*Recommended*", "*Very good quality*", or "*no complaints*", as well as "*good fit*" or "*very soft*" for Fashion and "*easy to install*" or "*works as expected*" for Motors, were among the most common short review sentences in our datasets. Overall, short sentences accounted for 17.1% of all review sentences in the Fashion domain and 17.9% in Motors. (2) **Personal**: sentences with a first-person pronoun, such as 'i', 'my', 'our', or 'us', or a 3rd-person *personal* pronoun, such 'she', 'his', 'hers', but not 'it' or 'them'. As demonstrated in the previous section, such pronouns hardly ever occur on a product description. Examples include "*I like the color of these jeans*"; "*Perfect fit for our car*"; "*My husband makes good use of them"; and "best gift for his birthday*". Overall, 35.9% of the review sentences matched this filtering criterion in the Fashion domain and 37.8% in Motors. (3) **Listing-specific**: Some review sentences refer to listing-specific aspects, as observed in the previous section. Examples include "*Great value for a fair price*" or "*Delivery was smooth and fast*". Since our goal is to produce a description at the product level rather than the listing (item) level, such aspects are not suitable for referencing as part of the description, since they may vary according to the seller. Our blacklist for this rule included the unigrams 'price', 'cheap', 'expensive', 'delivery', 'shipping', 'seller', and 'warranty'. Overall, 5.7% of the review sentences matched this rule in the Fashion domain and 6.1% in Motors.

Our rules aim to filter out sentences that are not suitable for a description with a very high likelihood, almost by definition. We therefore did not filter out other potential candidates, such as sentences in past tense, since these can sometimes be appropriate (e.g., "Tested on several cars"). We also did not automatically filter out sentences from reviews with low ratings, because the vast majority of the reviews in our dataset had a positive rating, in line with past work that indicated online user reviews tend to the positive [8]. Overall, 53.7% and 55.2% of the review sentences were filtered out using the three rules above, for Fashion and Motors, respectively.[3]

4.2 Automatic Classification

After the initial rule-based filtering, we set out to explore a supervised approach, where we trained a classifier to predict whether a product review sentence is suitable as a description sentence for the same product. To this end, we sampled uniformly at random, out of all review sentences that were not filtered out by the rules, 25K sentences for each of the two datasets, Fashion and Motors. Each of the 25K sentences were then labeled by a group of 10 annotators, who were asked to mark them as either '*good*' or '*bad*', i.e., suitable to be part of a product description or not. In case the sentence was labeled *bad*, the annotators also selected a reason. The set of reasons was identified in an earlier round of labeling, and included 'other' in case none of the 10 reasons was appropriate. The annotators received detailed guidelines, explaining what makes a sentence suitable versus unsuitable for a description, with examples of *good* and *bad* sentences, as well as examples for each of the possible reasons for *bad*. They also performed qualification tests, i.e., an iterative process of labeling, followed by feedback from other annotators, until the quality was aligned among all. At the end of the process, the inter-annotator agreement for the task of *good* versus *bad* labeling, measured by Cohen's kappa [9], was 0.89 for Fashion and 0.9 for Motors, calculated over a set of 300 sentences labeled by two different annotators. Overall, 8.55% of the Fashion and 7.97% of the Motors sentences were labeled *good*.

Table 3: Reasons for review sentences labeled '*bad*' and their distribution (portion of all sentences marked *bad*) for Fashion and Motors.

Reason	% Fashion	% Motors	Example
Subjective	52.50%	52.43%	It was the easiest jumpstart ever.
Missing context	16.86%	16.82%	Otherwise it remains idle.
Refers to a listing's aspect	8.40%	6.73%	10 bucks for 3 pairs is a great deal.
Non-informative	7.95%	6.42%	This shirt is great.
Poor language and spelling	4.91%	5.17%	Extremly easy setup let's you pull you vehecle's code fast.
Negative sentence	3.90%	4.25%	Only issue is the pretty thin material.
Expresses explicit doubt	2.40%	2.30%	Probably good also for bicycle tires.
Refers to the description	1.83%	1.74%	The hat is exactly as described.
Other	1.49%	1.24%	Like others here have said, this gas can has a long rotating nozzle.
Too specific/detailed	0.64%	2.52%	Great for Honda 2003 2.0L.
Offensive language	0.12%	0.19%	Fantastic product, bright as sh*t.

Table 3 lists the different reasons for *bad* sentences. It can be seen that the distribution is similar for Fashion and Motors, with subjective sentences accounting for a little over half of the *bad* sentences in both, followed by sentences with a missing context. The only noticeable difference between the domains is for the too specific/narrow reason, which is generally uncommon, but occurred substantially more frequently in Motors.

4.2.1 Good vs. Bad Review Sentences. Before building our classifier, we performed a statistical analysis comparing the review sentences labeled *good* by our annotators with review sentences labeled *bad*.

Table 4 presents the most distinctive unigrams and bigrams for *good* review sentences versus *bad* review sentences and vice versa, for Fashion and Motors. Distinctive terms were calculated using KL divergence as described in Section 3. It can be seen that *good* sentences in both domains include positive adjectives and adverbs, such as 'easy' and 'very', which are at the top of both unigram lists, as well as 'great', 'sturdy', 'nice', and 'well', which are used to describe products as easy to use/install/assemble; being of good/high quality or well made; or being perfect/excellent/great for a specific use. While many of the terms are common for both domains, in Fashion traits such as comfortable, durable, or fitting

[3]The portions of all three rules do not sum up to the total number of filtered sentences, since some sentences matched more than one rule.

Table 4: Most distinctive unigrams and bigrams for *good* sentences vs. *bad* sentences ('*Good*') and vice versa ('*Bad*') in Fashion and Motors.

Fashion				Motors			
Good	*Bad*	*Good*	*Bad*	*Good*	*Bad*	*Good*	*Bad*
easy	was	easy to	a little	easy	was	easy to	it was
very	but	good quality	it was	very	but	to use	so far
sturdy	not	well made	so far	great	it	very easy	than the
quality	it	very sturdy	a bit	use	would	to install	better than
comfortable	love	perfect for	but the	install	not	works great	a little
nice	than	are very	as expected	quality	than	good quality	a bit
well	as	they fit	they were	works	had	well made	seems to
great	buy	great for	but it	well	as	to apply	as advertised
durable	had	high quality	happy with	nice	buy	very well	as described
is	would	to assemble	but they	your	this	comes with	happy with

well are salient, while in Motors aspects of installation, application, and work are more dominant. For *bad* sentences, there is also a substantial overlap between the domains, implying the language differences between *good* and *bad* sentences can be generalized. The list of terms reflects some of the key reasons for sentences not fitting into a description, such as subjective viewpoints (e.g., 'better than', 'happy with', 'love'), personal experiences in past tense ('it was'), expression of doubt ('seems to'), negative sense ('but', 'not') and a reference to the description itself ('as described', 'as advertised').

Table 5 shows the portion of sentences labeled as *good* according to various characteristics of the sentence and the review it originated from, for Fashion and Motors. Inspecting the sentence length, starting 4 words, it can generally be seen that shorter sentences are somewhat more likely to be labeled *good*. For Fashion, the "optimal" sentence length is 6-8 words, while for Motors the percentage of *good* sentences consistently decreases with the length. Long sentences are more likely to include expressions that would make them unsuitable for a description based on some of the reasons detailed in Table 3, particularly subjectivity, which is the most common. Looking at the length of the review, it can be observed that sentences originating from long reviews are less likely to be *good*. For both Fashion and Motors, the review length that yields the highest portion of *good* is 3 sentences. Finally, inspecting the position of the sentence within the review (while controlling for the review length), there is a consistent trend indicating that sentences that occur towards the beginning of the review (not necessarily the first sentence) have higher likelihood to be *good*. The last sentence of the review has a particularly low likelihood to fit a description. This trend is consistent for Fashion and Motors and persists for higher review lengths not shown in Table 5. Overall, it suggests that users are more likely to include descriptive details earlier in the review.

4.2.2 Supervised Approaches. For the *good* versus *bad* classification task, we experimented with both traditional machine learning models and deep learning approaches. For the latter, we examined the effect of an attention mechanism and the use of a multi-task learning approach that tries to predict a reason for a sentence being inadequate for a description. We used 5-fold cross-validation to tune the hyper-parameters and evaluate the classifiers. As evaluation metric, we used the area under the ROC curve (AUC).

Table 5: Percentage of *good* sentences in Fashion ('F') and Motors ('M') distributed by sentence length, review length, and sentence position within the review.

Sentence length (words)		4	5	6	7 – 8	9 – 10	11 – 12	13 – 15	16+
	F	9.22%	9.96%	10.21%	10.38%	8.32%	7.81%	6.69%	6.05%
	M	11.07%	9.91%	9.43%	8.86%	7.42%	6.86%	6.60%	6.63%
Review length (sentences)		1	2	3	4	5 – 6	7 – 9	10+	
	F	7.49%	9.19%	10.27%	9.46%	8.33%	8.11%	6.81%	
	M	9.15%	10.04%	10.30%	9.88%	8.24%	6.69%	5.66%	
Sentence position in review				1st	2nd	3rd	4th	5th	
	F	Review length 3		11.48%	9.24%	8.90%			
		Review length 4		10.67%	11.16%	7.02%	5.27%		
		Review length 5		8.92%	10.99%	8.72%	6.64%	4.35%	
	M	Review length 3		12.25%	8.78%	7.56%			
		Review length 4		11.31%	10.76%	7.97%	6.82%		
		Review length 5		9.89%	11.50%	8.88%	5.59%	3.86%	

Table 6: AUC performance results for classifying review sentences as *good* or *bad* for the product's description.

Classifier	Fashion	Motors
Naïve Bayes	0.779	0.798
XGBoost	0.831	0.839
LSTM	0.914	0.916
LSTM-Attention	0.915	0.916
LSTM-MTL	**0.924**	**0.924**

Naïve Bayes and XGBoost. We examined two common models for text classification: Naïve Bayes [60] and XGBoost [7]. Our features included textual features, specifically the unigrams, bigrams, and trigrams of the review sentence, and statistical features of the sentence and originating review, based on Table 5. For Naïve Bayes, we tuned the type of n-grams (unigrams, bigrams, or trigrams) and for XGBoost, in addition, the maximum depth of a tree, minimum child weight, as well as the learning rate and number of rounds (trees). Results for both classifiers are presented at the top rows of Table 6, indicating XGBoost achieved a higher AUC.

To better understand the contribution of non-textual features, which are based on the characteristics presented in Table 5, we trained the XGBoost classifier with different feature subsets (while always using textual features). Feature types included sentence length (in words and characters), review length (in words and sentences), and position (absolute position and normalized by the total number of review sentences). Results, presented in Table 7, indicate that position features yielded a slightly higher performance gain than sentence and review length. Overall, however, as can be observed in the 'All' row, the contribution of these features on top of the textual features was minor (+0.97% for Fashion and +0.96% for Motors). We therefore did not use these features in the remainder of our experiments and worked with the review text only.

While the XGBoost classifier demonstrated reasonable performance, we set out to explore how it can be further enhanced. In recent years, various applications in text classification have shown significant improvement by the use of deep learning models [17, 34].

Table 7: AUC performance of the XGBoost classifier with textual features when using ('Only') or disregarding ('Exclude') additional types of features. The 'All' row refers to all additional types jointly.

Additional Features	Fashion		Motors	
	Only	Exclude	Only	Exclude
Sentence Length	0.824	0.828	0.833	0.835
Review Length	0.828	0.826	0.835	0.834
Sentence Position	0.829	0.828	0.837	0.836
All	0.831	0.823	0.839	0.831

We therefore explored several deep learning approaches for our task:

LSTM. A recurrent neural network based on long short-term memory (LSTM) [25] architecture, with pre-trained word2vec embeddings using continuous bag of words (CBOW) with negative sampling [49]. We experimented with pre-training over Wikipedia and over our own datasets of more than 10 million product reviews for Fashion and Motors, respectively, as described in Section 3. Pre-training using our own data, with separate models for Fashion and Motors, consistently achieved a slightly better performance. We henceforth only report the results when using word2vec pre-trained based on our own data.

LSTM with Attention. Attention mechanisms enable the network to focus on relevant parts of the input [69]. The overall architecture of the "attention network" consists of two components: an LSTM-based word sequence encoder and a word-level attention layer. Given a review with the words $w_i, i \in [1, N]$, we first embed the words using pre-trained word2vec, as previously described. We then use the LSTM network to produce the hidden states $h_i, i \in [1, N]$. The attention mechanism is subsequently used to put more focus on certain words in the review sentence. For example, in the sentence "*this is a high quality product*" the words "*high quality*" should receive higher weight. We feed the word annotation h_i through a single-layer perceptron network to receive u_i, a latent representation of h_i. Then, we calculate the similarity of u_i with a word-level context vector, normalized by a softmax function, to produce the word's importance weight. We then construct the review vector as a weighted sum of the word annotations based on each word's weight.

Deep Multi-Task Learning. Multi-task learning (MTL) is based on the idea that features trained for one task can be useful for related tasks. Models for all tasks of interest are jointly trained with an additional linkage between their trainable parameters, aiming at improving the generalization error [5]. Multi-task learning can be viewed as a form of inductive transfer learning, which can help improve the model by introducing an inductive bias. In the case of MTL, the inductive bias is provided by the *auxiliary tasks*, which lead the model to prefer hypotheses that explain more than one task. MTL has been widely used for deep learning tasks, e.g., in computer vision [16] and natural language processing [23, 63].

Our network is based on an MTL hard parameter sharing architecture [61], which includes an LSTM layer shared among all tasks and separate feed-forward networks per task. We learned the task of predicting if a review sentence could be included as part of a description (*good*/*bad*) with the additional auxiliary tasks that specialize on a specific reason, particularly each of the top four reasons listed in Table 3, which cover over 80% of the *bad* sentences in both Fashion and Motors. Each specialized classifier is trained to predict if the sentence falls under the specific reason for not fitting in a description. For example, the 'subjectivity' classifier learns to predict subjective sentences. Overall, the specialized classifiers may help the main task of predicting *good* or *bad* sentences. Similarly to the LSTM with attention, given a review sentence, we first embed the words using our pre-trained word2vec, and then use the shared LSTM encoder to produce the latent representation of the review sentence. This representation is then fed in parallel into 2-5 fully-connected feed-forward networks (one per task), each with two hidden layers, trying to predict the output for that task independently. The ultimate loss function to be optimized assigns a weight to each of the task-specific loss functions. Notice the weighted loss function serves for optimization at training time, while testing remains for the *good* versus *bad* task only. In Figure 1, we demonstrate our MTL hard parameter sharing network, with a shared LSTM layer and 5 separate feed-forward layers, one for each task.

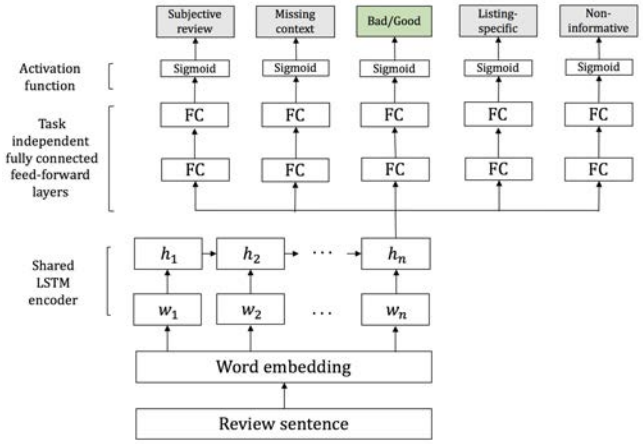


Figure 1: Hard parameter sharing for deep multi-task learning architecture.

For each of the three deep learning methods, hyper-parameter tuning included the batch size, the dropout rates, and the number of hidden units in the LSTM layers. In addition, we experimented with both Adam [33] and RMSProp [65] optimizers. For the MTL architecture, we also tuned the subset of tasks to be included and their weight in the loss function, as well as the dropout rates and number of hidden units of the feed-forward layers.

4.2.3 Performance Results. Table 6 presents the AUC results for the *good* versus *bad* task. Evidently, the deep learning classifiers achieved substantially better results than the Naïve Bayes and XGBoost classifiers. The attention mechanism did not lead to any performance gain, possibly due to the sentences' length: as shown in Table 1, the median number of words per review sentence is 6 and 7, for Fashion and Motors, respectively. The effectiveness of the attention mechanism, however, typically lies in longer sentences [44]. The LSTM-based MTL model achieved the best AUC

Table 8: AUC performance of the deep MTL classifier with different subsets of the four auxiliary tasks: subjective ('subj'), missing context ('MC'), listing-specific ('LS'), and non-informative ('NI').

Fashion		Motors	
Subj, MC, LS, NI	0.921	Subj	0.920
LS, NI	0.922	MC, LI	0.922
Subj	0.922	MC, LS, NI	0.923
Subj, LS, NI	0.923	SUBJ, MC	0.923
Subj, MC	**0.924**	**Subj, MC, LS, NI**	**0.924**

results for both datasets, Motors and Fashion. Table 8 presents the auxiliary task subsets that yielded the best performance. For Motors, the combination of all four auxiliary tasks led to the best performance, while for Fashion it was the combination of the subjective and missing context tasks. Indeed, these are the two most common reasons for sentenced being labeled *bad*, as shown in Table 3. Other combinations listed in Table 8 yielded close performance to the top ones. Noticeably, using only one auxiliary task – for identifying subjective sentences – produced a substantial part of the performance gain compared to the vanilla LSTM, especially for Fashion. As previously shown (Table 3), 'subjective' is the most common reason, covering over half of the *bad* sentences for each of the two domains.

As mentioned before, we compared Adam and RMSProp optimizers. The latter was found to perform better, slightly but consistently across different number of tasks, for both Fashion and Motors.

5 DIVERSIFICATION

In order to create coherent and concise descriptions, we wished to avoid the inclusion of very similar sentences in the same description. After identifying candidate sentences for the description, our next step was therefore the identification of redundant sentences in order to increase the diversity in the content of the final description. The diversification phase was based on the computation of similarity between candidate sentences. To this end, we used the common cosine similarity measure, while experimenting with three sentence representation methods:

Weighted Bag-of-Words (BOW). Each sentence is represented as the TD-IDF weighted vector of its words [1].

Average Embedding. The weighted BOW approach measures similarity based on actual word overlap. In order to capture deeper semantic similarity, an embedded representation of the sentence can be used. Specifically, we apply word embedding using word2vec and average over all the sentence's words to produce the final sentence representation. The word2vec model is trained separately for Fashion and Motors, based on the dataset of over 10 million reviews per domain, as described in Section 3.

Weighted Embedding. This method applies the same word2vec embedding as described above, but instead of averaging the embeddings to generate the sentence representation, a common word weighting approach is applied [3]. This weighting is performed using TF-IDF scores, as it has been empirically shown to achieve similar results to learning the language model of the sentences.

Table 9: Accuracy for classifying sentence pairs as similar or not using different representations.

Sentence Representation	Fashion	Motors
Weighted bag of words	76.6	78.3
Average embedding	88.3	88.6
Weighted embedding	**92.2**	**91.7**

To decide whether two sentences are different enough to be included in the description, a similarity threshold θ for each method had to be determined. To avoid using a hard-coded threshold, we learned θ in an unsupervised manner from the data, for each of the two domains (Fashion and Motors). To this end, we considered the product descriptions in both of our datasets, which were manually curated by domain experts. We measured the similarity between each pair of sentences within each description and considered the 90-th percentile as the threshold θ. Intuitively, we set our threshold to allow a similarity level that is aligned with the typical degree of similarity that exists in professionally-written descriptions.

In order to compare the three representations, we conducted a small-scale experiment. For each domain (Fashion and Motors), we sampled 1000 pairs of sentences in the following way: we sampled 200 products from the respective dataset uniformly at random. For each product, we sampled 10 pairs of review sentences, uniformly at random out of all (unordered) pairs of sentences labeled *good*. We evaluated only *good* sentences, since the *bad* sentences were not considered for the final description. The sampled pairs were labeled by 3 professional annotators as either similar or not. The annotators were instructed to label a pair of sentences as similar in case one of them did not add any substantial piece of information to the reader on top of the other. The agreement between annotators, measured by Cohen's Kappa, was 0.84 for Fashion and 0.86 for Motors, calculated over 100 pairs for each domain, which were evaluated by two different annotators.

Overall, 35.2% of the sentences were labeled as similar, indicating there was indeed a high level of redundancy among *good* sentences. We tested each of the three methods against this labeled set by calculating its accuracy, i.e., the portion of pairs it correctly classified out of the 1000 pairs. Results are depicted in Table 9. It can be seen that both embedding-based methods reached a substantially higher accuracy than the weighted BOW method, for both domains. Weighted embedding slightly outperformed average embedding at over 90% for both domains, and was therefore our choice as sentence similarity method for the rest of our experiments.

6 FINAL DESCRIPTION GENERATION

In this section, we describe our methods for producing the final product description. Following, we present a detailed evaluation of the generated descriptions. Our evaluation compares the different methods for producing the descriptions and different description lengths by their overall quality ratings, as well as by ratings of more specific description aspects. In addition, we present results over publicly available data and experiment with cross-domain description generation.

6.1 Sentence Selection

The final step of our product description generation process is the selection of the final sentence set. From the previous steps, we have a list of candidate sentences, with their classification score reflecting their likelihood to be suitable to a product description, a similarity metric that can be applied to a pair of candidate sentences to measure their closeness, and a similarity threshold θ reflecting the desired similarity between sentences in a description. With these in hand, we examine four methods to generate the final description, given an integer K that determines the desired number of sentences in the description.

Greedy approach. This method traverses the list of candidates according to their *good*/*bad* classification score, from highest to lowest, and adds a candidate to the description if (and only if) it is not similar (i.e., has a similarity score of θ or higher) to any of the candidates already selected for the description. The process stops when K sentences have been added.

LexRank. This method uses LexRank, a common extractive summarization method that yielded high performance results for several text summarization tasks [12]. Specifically, LexRank assigns an importance score per sentence, using random walks and eigenvector centrality. We apply LexRank on top of the list of candidate sentences and use its score to rank the candidate sentences. Intuitively, the LexRank score indicates how well the candidate covers the information included in the other candidates. The method then operates similarly to the greedy approach: after ranking the candidate sentences by their LexRank score, the list is traversed by descending score and a sentence is added in case it is not similar to any of those previously selected for the description.

K-means classification score. This method partitions the candidate sentences into K clusters using the k-means algorithm [47] with k=K. The distance between sentences is calculated as $(1-s)$, where s is our sentence similarity measure described in Section 5. For the final product description, the clusters are traversed from the largest (representing the highest number of review sentences) to the smallest and from each cluster, the sentence with the highest classification score is added to the description.

K-means centroid. The method works as the previous one, but instead of selecting the candidate with the highest classification score from each of the K clusters, the candidate closest to the centroid is selected, assuming it best represents the cluster.

Note that smart ordering of the sentences within the final description is beyond the scope of this work. Currently, the sentences are ordered by classification score (greedy), summarization score (LexRank), or cluster size (both k-means methods).

6.2 End-to-End Description Evaluation

For an end-to-end evaluation of our approach, we set out to examine full descriptions generated using one of the four methods described above. For candidate sentence extraction, we used the LSTM-MTL classifier and for similarity measure we used weighted sentence embedding, since both were found to yield the best performance for their respective tasks, as previously reported. We experimented with descriptions of three lengths: 3, 5, and 7 sentences. These represent relatively concise content, which can be swiftly consumed in its entirety, and is especially suitable for small-screen devices, such as mobile phones. We generated descriptions for all products in both the Fashion and Motors datasets. Using our candidate extraction method (Section 4), we identified review sentences that were suitable for a description for each product in the sample. On average, for each Fashion product we identified 46.1 *good* sentences (std: 9.3, median: 43, min: 24, max: 97) and for Motors 47.3 (std: 11.7, median: 45, min: 19, max: 103). For each such product, we then generated descriptions of length 3, 5, and 7, using each of the four methods described above (a total of 12 description versions). These descriptions were evaluated by 15 professional annotators. We ensured each annotator evaluated no more than one description per product.



Figure 2: Product description evaluation interface.

Annotators were presented with the product's title, image, and generated description and were asked to rate the quality of the description's text for serving as the product's description, on a 5-point Likert scale, from 'very bad' to 'very good'. In addition, to examine finer-grained aspects of the description's quality, along the lines presented in the Introduction, annotators were asked to assess, on the same scale, to what extent the description's text was readable, informative, objective, and relevant to the product. They were provided with good and bad examples for each of the questions. Figure 2 demonstrates the user interface developed to collect annotators' input.

The inter-annotator agreement for the main quality evaluation question, measured using weighted Kappa [10], was 0.83 for Fashion and 0.84 for Motors, calculated over a set of 200 generated descriptions evaluated by two different annotators. Table 10 presents a few examples of our generated descriptions, which were rated 5.

6.2.1 Rating Results. Table 11 shows the ratings of the main question (overall description quality) in our end-to-end evaluation. Generally, the generated descriptions received high ratings, with an average of above 4 for all methods except for the greedy approach. Ratings for Fashion and Motors were similar, giving some indication that our method may be generalized to other e-commerce domains. Among the four methods, LexRank achieved the highest ratings, consistently for all three description lengths for both Fashion and Motors. At the other extreme, the greedy approach yielded substantially lower ratings than the other methods. K-means with the closest-to-centroid selection was consistently the second best, higher than k-means with highest-score selection. This suggests that selecting the sentence based on representation of the whole cluster rather than the individual prediction score for description suitability is preferable. The superiority of the LexRank approach implies that good representation of the whole set of candidates for final sentence selection works well. All differences between the

Table 10: Example descriptions rated 5.

Domain	Product	Description
Fashion	Socks	Perfect thickness for shoes or boots. Extra padding at toes. The quality is excellent. Easy to handle and very comfortable. No shrinkage in washer or dryer.
Fashion	Jacket	The cuffs on the sleeves are adjustable, which is perfect for keeping wind out when biking or for just a tighter fit around your wrists. It's very soft on the skin. The zipper works easily. The material is well made and the hood tucks away easily and hides well when not needed. Folds up to fit in bag.
Motors	Scratch removal system	Great for minor to moderate damage. Use it for removing scratches from your car. There is enough for several repairs. Coat twice for extra protection. Very recommended after wax.
Motors	Motor oil	It works well and lasts for a reasonable period of time. The oil is still very clean when changed. Engine runs smoother and pulls better. Good oil for any motorcycle. This oil is made for high temp engines.
Toys	Magnetic cubes	Works for all ages – toddlers to teens. The letters are a great addition and a fun way for young kids to learn their alphabet. Very colorful and high quality magnetic toy! Very easy to build. Perfect for family time.
Electr.	Mobile phone	This phone has a very good battery. The screen is big enough for watching movies. Includes power charger and headphones. The quality of photos is excellent. The phone is very smart and connects well with other devices.

Table 11: Average rating of description quality of 3, 5, and 7 sentences for products in Fashion and Motors.

	K=3		K=5		K=7	
	Fashion	Motors	Fashion	Motors	Fashion	Motors
Greedy	3.95	3.92	3.81	3.81	3.80	3.77
K-means score	4.10	4.09	4.08	4.06	4.04	4.01
K-means centroid	4.29	4.28	4.17	4.22	4.26	4.23
LexRank	**4.36**	**4.30**	**4.38**	**4.35**	**4.36**	**4.33**

methods were statistically significant, except between LexRank and k-means-centroid with K=3.[4] As for the length of the description, there was no consistent trend, but the highest ratings for LexRank were achieved for K=5 sentences by an insignificant difference from both K=3 and K=7 sentences.

Figure 3 presents the rating results for the additional questions in our evaluation, relating to the different quality aspects. Across all questions, LexRank consistently achieved the best ratings, with the rest of the methods ranked similarly to the general quality question. Among the four questions, ratings were highest for product relevance, with LexRank achieving an average of over 4.5 for all three description lengths. This indicates that as could be expected,

[4]Statistical significance was measured using one-way ANOVA with Tukey post-hoc comparisons for $p<0.01$.

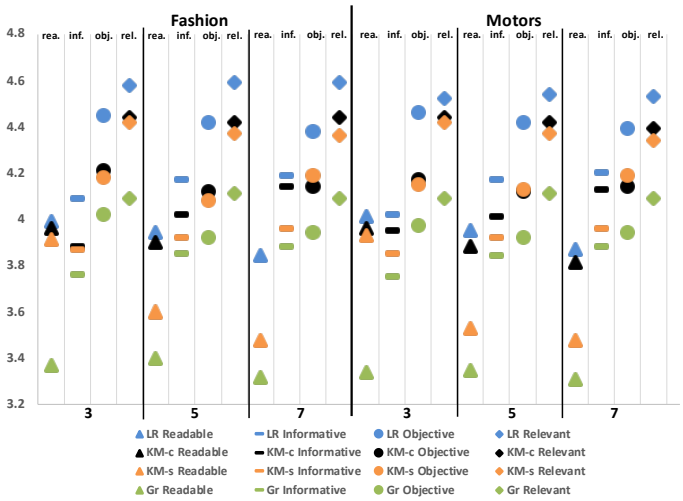


Figure 3: Average rating of four aspects (readable, informative, objective, relevant) for descriptions of length 3, 5, and 7 sentences, generated by four methods: LexRank (LR), K-Means centroid (KM-c), K-Means score (KM-s), and Greedy (Gr).

the reviews serve as a relevant source of information about the product. Objectivity was also rated high in general, indicating our method for filtering the objective parts out of the subjective reviews (step 1) works well. Readability received the lowest rating, with the greedy approach (and for K=5 and K=7 also k-means with highest-score selection) performing especially poorly. Inspecting the ratings by description length, objectivity and readability tend to decrease with description length, while informativeness increases. Product-relevance remains stable regardless of the description length. Overall, it is expected that as the description is longer, it is more likely to contain non-objective parts and become less readable due to connectivity or repetition issues, while it is also likely to contain more details and therefore become more informative. Inspecting the different ratings in Figure 3, descriptions of 5 sentences seem to yield the best trade-off between these two trends. For instance, consider the first example in Table 10. When the generated description with K=3 included the first 3 sentences, its 'informative' rating reduced. On the other hand, when the K=7 version included the extra sentence "*Best socks for sports, especially running*," the 'objective' rating reduced.

6.2.2 Evaluation over public data. The datasets used in this work are proprietary and cannot be shared due to business sensitivity. We therefore set out to get a sense how well our method performs on publicly available data. As a first step, we observe that 13.5% (120 in total) of the Fashion products and 16.7% (135) of the Motors products sent for our end-to-end evaluation are included in a large publicly-released e-commerce dataset [24]. Inspecting the ratings for these product subsets only, the results were almost completely identical to those reported for all products (Table 11 and Figure 3) and are thus not separately reported.

As a second (and final) step, we conducted another evaluation, in which all products in the public dataset [24] with at least 10

Table 12: Average rating of 5-sentence descriptions generated using the LexRank method over public data for Fashion and Motors (upper section) and for Toys and Electronics using cross-domain learning (lower section).

	Overall	Readable	Informative	Objective	Relevant
Fashion	4.34	3.88	4.20	4.40	4.53
Motors	4.32	3.95	4.22	4.43	4.50
Electronics	4.11	3.60	4.05	4.18	4.21
Toys	4.05	3.62	4.09	4.21	4.27

reviews between January and July 2014 (the most recent months in the dataset) were evaluated end-to-end using the LexRank method with k=5 sentences, by 3 professional annotators. This set included 140 Fashion products and 120 Motors products. Results, depicted in the upper section of Table 12, are very similar (±1.5%) to those reported for our own datasets in Table 11 and Figure 3.

6.2.3 Cross-Domain Generation. Finally, we set out to explore if our approach can be used across domains. To this end, we trained a model based on both the Fashion and Motors data and generated descriptions for two additional e-commerce domains: Toys and Electronics. As in the previous experiment, we generated descriptions of 5 sentences using the LexRank method for all products in these domains with at least 10 reviews between January and July 2014, resulting in a total of 55 products for Toys and 75 for Electronics [24]. These descriptions were evaluated by the same 3 annotators and the lower section of Table 12 presents their ratings. It can be seen that while the overall quality ratings are somewhat lower than for Fashion and Motors, they are still just above 4 on average. There is a rather similar rating decrease across all four quality aspects relative to Fashion and Motors, with the sharpest decline being for readability. The last two examples in Table 10 demonstrate generated descriptions for Toys and Electronics.

7 CONCLUSION AND FUTURE WORK

In this paper, we presented a method for automatic generation of product descriptions based on online user reviews. Product descriptions provided by sellers are often missing or lacking. Hence, customers turn to product reviews and often spend a significant amount of time sifting through them before making a purchase. However, users might still find it hard to extract the relevant information from thousands of reviews available online. Moreover, many reviews include personal and subjective opinions, while the users are sometimes only interested in the key product details before purchasing [15].

We first studied the main differences between the reviews and descriptions and adopted a supervised deep multi-task learning approach to identify appropriate review sentences. Afterwards, we introduced a similarity measure between review sentences that helped us eliminate redundancies when creating the descriptions. Finally, we used extractive text summarization to create a coherent and concise product description. We provided an extensive set of experiments that demonstrated our approach is productive.

Our experiments inspected two principally different e-commerce domains: Fashion and Motors. The similar results throughout our evaluation for both domains, as well as the analysis revealing common review language characterizing sentences that can be used for a description, suggest that our approach can be generalized to additional domains. Indeed, cross-domain description generation, trained over a combination of Fashion and Motors reviews, showed promising results when applied to two other primary e-commerce domains: Toys and Electronics. Our evaluation and examples also illustrate that our descriptions are somewhat different than seller-provided descriptions. A presentation of such descriptions on product pages may include an indication they are crowd-based or stem from user reviews. In addition, the descriptions can be presented as a bulleted list, as currently done on some e-commerce websites, to provide an alternative user experience and mitigate readability issues.

For future work, we plan to focus on four main directions. Currently, we select K sentences for the description without any special ordering. We believe that proper ordering may increase the readability of the descriptions. Second, automatically deriving the value of K based on the product and review content can further improve description quality. Third, abstractive approaches [13, 14], which adapt the original review text and combine content from different sentences, are also worth exploring, and can help generate descriptions of even higher quality based on fewer reviews. Finally, personalizing the generated descriptions for the individual consumer can help make them even more compelling.

REFERENCES

[1] Palakorn Achananuparp, Xiaohua Hu, and Xiajiong Shen. 2008. The evaluation of sentence similarity measures. In *Proc. of DaWaK*. 305–316.
[2] Mathieu Acher, Anthony Cleve, Gilles Perrouin, Patrick Heymans, Charles Vanbeneden, Philippe Collet, and Philippe Lahire. 2012. On extracting feature models from product descriptions. In *Proc. of VaMoS*. 45–54.
[3] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *Proc. of ICLR*.
[4] Adam Berger and John Lafferty. 1999. Information retrieval as statistical translation. In *Proc. of SIGIR*. 222–229.
[5] Rich Caruana. 1998. Multitask learning. In *Learning to learn*. Springer, 95–133.
[6] Y. Chae, M. Nakazawa, and B. Stenger. 2018. Enhancing product images for click-through rate improvement. In *Proc. of ICIP*. 1428–1432.
[7] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proc. of KDD*. 785–794.
[8] Judith A. Chevalier and Dina Mayzlin. 2006. The effect of word of mouth on sales: Online book reviews. *Journal of Marketing Research* 43, 3 (2006), 345–354.
[9] Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement* 20, 1 (1960), 37–46.
[10] Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin* 70, 4 (1968), 213–220.
[11] Horatiu Dumitru, Marek Gibiec, Negar Hariri, Jane Cleland-Huang, Bamshad Mobasher, Carlos Castro-Herrera, and Mehdi Mirakhorli. 2011. On-demand feature recommendations derived from mining public product descriptions. In *Proc. of ICSE*. 181–190.
[12] Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research* 22 (2004), 457–479.
[13] Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph-based approach to abstractive summarization of highly redundant opinions. In *Proc. of COLING*. 340–348.
[14] Shima Gerani, Yashar Mehdad, Giuseppe Carenini, Raymond T. Ng, and Bita Nejat. 2014. Abstractive summarization of product reviews using discourse structure. In *Proc. of EMNLP*. 1602–1613.
[15] Anindya Ghose and Panagiotis G. Ipeirotis. 2011. Estimating the helpfulness and economic impact of product reviews: Mining text and reviewer characteristics. *IEEE Transactions on Knowledge and Data Engineering* 23, 10 (2011), 1498–1512.
[16] Ross Girshick. 2015. Fast r-cnn. In *Proc. of ICCV*. 1440–1448.
[17] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proc. of ICML*. 513–520.

[18] Jade Goldstein, Mark Kantrowitz, Vibhu Mittal, and Jaime Carbonell. 1999. Summarizing text documents: sentence selection and evaluation metrics. In *Proc. of SIGIR*. 121–128.

[19] Vishrawas Gopalakrishnan, Suresh Parthasarathy Iyengar, Amit Madaan, Rajeev Rastogi, and Srinivasan Sengamedu. 2012. Matching product titles using web-based enrichment. In *Proc. of CIKM*. 605–614.

[20] Anjan Goswami, Naren Chittar, and Chung H. Sung. 2011. A study on the impact of product images on user clicks for online shopping. In *Proc. of WWW*. 45–46.

[21] Ido Guy, Avihai Mejer, Alexander Nus, and Fiana Raiber. 2017. Extracting and ranking travel tips from user-generated reviews. In *Proc. of WWW*. 987–996.

[22] Ido Guy and Bracha Shapira. 2018. From Royals to Vegans: Characterizing Question Trolling on a Community Question Answering Website. In *Proc. of SIGIR*. 835–844.

[23] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2016. A joint many-task model: Growing a neural network for multiple NLP tasks. *arXiv preprint* abs/1611.01587 (2016).

[24] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proc. of WWW*. 507–517.

[25] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[26] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proc. of KDD*. 168–177.

[27] Nan Hu, Paul A. Pavlou, and Jennifer Zhang. 2006. Can online reviews reveal a product's true quality?: Empirical findings and analytical modeling of online word-of-mouth communication. In *Proc. of EC*. 324–330.

[28] Alice Jiang, Zhilin Yang, and Minjoon Jun. 2013. Measuring consumer perceptions of online shopping convenience. *Journal of Service Management* 24, 2 (2013), 191–214.

[29] Gagandeep Kaur and Gagandeep Kaur. 2016. Mobile applications are major players in the world of e-commerce. *International Journal of Advanced Research in IT and Engineering* 5, 2 (2016), 13–21.

[30] Zehra Kavasoğlu and Şule Gündüz Öğüdücü. 2013. Personalized summarization of customer reviews based on user's browsing history. *IADIS International Journal on Computer Science & Information Systems* 8, 2 (2013), 147–158.

[31] H. Khalid, E. Shihab, M. Nagappan, and A.E. Hassan. 2015. What do mobile app users complain about? *IEEE Software* 32, 3 (2015), 70–77.

[32] Hyun Duk Kim, Kavita Ganesan, Parikshit Sondhi, and Chengxiang Zhai. 2011. Comprehensive review of opinion summarization. *UIUC Technical Report* (2011).

[33] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint* abs/1412.6980 (2014).

[34] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification.. In *Proc. of AAAI*. 2267–2273.

[35] Eun-Ju Lee and Soo Yun Shin. 2014. When do consumers buy online product reviews? Effects of review quality, product type, and reviewer's photo. *Comput. Hum. Behav.* 31 (2014), 356–366.

[36] Beibei Li, Anindya Ghose, and Panagiotis G Ipeirotis. 2011. Towards a theory model for product search. In *Proc. of WWW*. 327–336.

[37] Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Ying-Ju Xia, Shu Zhang, and Hao Yu. 2010. Structure-aware review mining and summarization. In *Proc. of COLING*. 653–661.

[38] Xinxin Li and Lorin M Hitt. 2008. Self-selection and information role of online product reviews. *Information Systems Research* 19, 4 (2008), 456–474.

[39] Moez Limayem, Mohamed Khalifa, and A. Frini. 2000. What makes consumers buy from Internet? A longitudinal study of online shopping. *IEEE Trans. Systems, Man, and Cybernetics, Part A* 30, 4 (2000), 421–432.

[40] Chin-Yew Lin and Eduard Hovy. 2002. From single to multi-document summarization: A prototype system and its evaluation. In *Proc. of ACL*. 457–464.

[41] Bing Liu. 2012. *Sentiment analysis and opinion mining*. Morgan & Claypool Publishers.

[42] Qian Liu, Zhiqiang Gao, Bing Liu, and Yuanlin Zhang. 2015. Automated rule selection for aspect extraction in opinion mining. In *Proc. of IJCAI*. 1291–1297.

[43] Roque Enrique Lpez Condori and Thiago Alexandre Salgueiro Pardo. 2017. Opinion summarization methods. *Expert Syst. Appl.* 78, C (July 2017), 124–134.

[44] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint* abs/1508.04025 (2015).

[45] Duy Khang Ly, Kazunari Sugiyama, Ziheng Lin, and Min-Yen Kan. 2011. Product review summarization from a deeper perspective. In *Proc. of JCDL*. 311–314.

[46] Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2017. Interactive attention networks for aspect-level sentiment classification. In *Proc. of IJCAI*. 4068–4074.

[47] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proc. of Berkeley symposium on mathematical statistics and probability*, Vol. 1. 281–297.

[48] Deborah Brown McCabe and Stephen M Nowlis. 2003. The effect of examining actual products or product descriptions on consumer preference. *Journal of Consumer Psychology* 13, 4 (2003), 431–439.

[49] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint* abs/1301.37810 (2013).

[50] Hye-Jin Min and Jong C Park. 2012. Identifying helpful reviews based on customer's mentions about experiences. *Expert Systems with Applications* 39, 15 (2012), 11830–11838.

[51] Samaneh Moghaddam and Martin Ester. 2012. On the design of LDA models for aspect-based opinion mining. In *Proc. of CIKM*. 803–812.

[52] Ajinkya More. 2016. Attribute extraction from product titles in eCommerce. *arXiv preprint* abs/1608.04670 (2016).

[53] Quang Nguyen. 2012. *Detecting experience revealing sentences in product reviews*. Ph.D. Dissertation. University of Amsterdam.

[54] Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.* 2, 1-2 (Jan. 2008), 1–135.

[55] Eun Joo Park, Eun Young Kim, Venessa Martin Funches, and William Foxx. 2012. Apparel product attributes, web browsing, and e-impulse buying on shopping websites. *Journal of Business Research* 65, 11 (2012), 1583–1589.

[56] A.M. Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proc. of HLT*. 339–346.

[57] Katharina Probst, Rayid Ghani, Marko Krema, Andrew Fano, and Yan Liu. 2007. Semi-supervised learning of attribute-value pairs from product descriptions. In *Proc. of IJCAI*. 2838–2843.

[58] Reid Pryzant, Young-Joo Chung, and Dan Jurafsky. 2017. Predicting sales from the language of product descriptions. In *Proc. of ECOM (SIGIR Workshops)*.

[59] Pradeep Racherla, Munir Mandviwalla, and Daniel J Connolly. 2012. Factors affecting consumers' trust in online product reviews. *Journal of Consumer Behaviour* 11, 2 (2012), 94–104.

[60] Irina Rish. 2001. An empirical study of the naive Bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, Vol. 3. 41–46.

[61] Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint* abs/1706.05098 (2017).

[62] Keiji Shinzato and Satoshi Sekine. 2013. Unsupervised extraction of attributes and their values from product description. In *Proc. of ACL*. 1339–1347.

[63] Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proc. of ACL*, Vol. 2. 231–235.

[64] Krysta M. Svore, Lucy Vanderwende, and Chris J.C. Burges. 2007. Enhancing single-document summarization by combining RankNet and third-party sources. In *Proc. of EMNLP-CoNLL*.

[65] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4, 2 (2012), 26–31.

[66] Damir Vandic, Flavius Frasincar, and Uzay Kaymak. 2018. A framework for product description classification in e-commerce. *Journal of Web Engineering* 17, 1&2 (2018), 001–027.

[67] Shuai Wang, Zhiyuan Chen, and Bing Liu. 2016. Mining aspect-specific opinion using a holistic lifelong topic model. In *Proc. of WWW*. 167–176.

[68] Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *Proc. of AAAI*. 3316–3322.

[69] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proc. of NAACL*. 1480–1489.

[70] Koji Yatani, Michael Novati, Andrew Trusty, and Khai N. Truong. 2011. Analysis of adjective-noun word pair extraction methods for online review summarization. In *Proc. of IJACI*. 2771–2776.

[71] Naitong Yu, Minlie Huang, Yuanyuan Shi, et al. 2016. Product review summarization by exploiting phrase properties. In *Proc. of COLING*. 1113–1124.

[72] Di Zhu, Theodoros Lappas, and Juheng Zhang. 2018. Unsupervised tip-mining from customer reviews. *Decision Support Systems* 107 (2018), 116–124.

[73] Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006. Movie review mining and summarization. In *Proc. of CIKM*. 43–50.

Google Dataset Search: Building a search engine for datasets in an open Web ecosystem

Natasha Noy
noy@google.com
Google AI
Mountain View, California

Matthew Burgess
mattburg@google.com
Google AI
Mountain View, California

Dan Brickley
danbri@google.com
Google
Mountain View, California

ABSTRACT

There are thousands of data repositories on the Web, providing access to millions of datasets. National and regional governments, scientific publishers and consortia, commercial data providers, and others publish data for fields ranging from social science to life science to high-energy physics to climate science and more. Access to this data is critical to facilitating reproducibility of research results, enabling scientists to build on others' work, and providing data journalists easier access to information and its provenance. In this paper, we discuss Google Dataset Search, a dataset-discovery tool that provides search capabilities over potentially all datasets published on the Web. The approach relies on an open ecosystem, where dataset owners and providers publish semantically enhanced metadata on their own sites. We then aggregate, normalize, and reconcile this metadata, providing a search engine that lets users find datasets in the "long tail" of the Web. In this paper, we discuss both social and technical challenges in building this type of tool, and the lessons that we learned from this experience.

KEYWORDS

data discovery, search, metadata, structured data

ACM Reference Format:
Natasha Noy, Matthew Burgess, and Dan Brickley. 2019. Google Dataset Search: Building a search engine for datasets in an open Web ecosystem. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313685

1 WHY IT CAN BE DIFFICULT TO FIND DATASETS ON THE WEB

Data is the main substrate of research for scientists in many disciplines, for the work of journalists, for the analysis by policy makers, and for those of us who are curious to understand our world better. Data is published on the Web by national and regional governments, scientific publishers, commercial data providers, research consortia, specialized data repositories, data aggregators, and so on. There are thousands of data repositories on the Web [16] and many individual data publishers, from high school teams publishing the results of their science-fair projects to national research labs. The more data we publish on the Web, the more complex the problem of data discovery becomes: how do we find the data that we need for the task at hand and how do we assess its credibility, veracity, and suitability for our task? The following are some of the key factors that contribute to the difficulty of data discovery today.

Proliferation of data publishers: In the early days of the Web, many users discovered what was available on the Web by browsing through directories such as the Yahoo! Web directory. Eventually though, the Web became too large for any directory to be sufficiently comprehensive and *search* became the primary way of finding where the information was on the Web. Today, the world of publishing datasets on the Web is in a similar transition: there are a number of well respected directories of dataset publishers (e.g., DataCite [29], re3data [16], Scientific Data in Nature [19]), but now they inevitably miss new datasets or repositories [7]. In addition, they miss datasets published by individual data providers, such as a scientist publishing the dataset resulting from her experiment on her own site. Finally, many curated resources collecting dataset repositories focus largely on government and research data and miss repositories that come from the private sector. Yet, there are valuable datasets that originate from companies who may charge or have more restrictive licensing terms for their data (e.g., ceicdata.com). Thus, the time is ripe to add Web-wide search capabilities for datasets and not to rely exclusively on curated directories.

The "long tail" and the "deep Web": Web search engines often fail at finding data for a variety of reasons: many pages describing datasets are in the "long tail" of the Web [8]. In many cases, data repositories do not make the individual pages for datasets easily crawlable because they are accessible only through queries, a phenomena referred to as the "deep Web" [18]. Consider, for example, Open Data Network [23], which hosts many datasets from local governments in the United States. It is a resource that many data journalists use (but many social-science researchers may not know about). The only way to get to a page for any specific dataset that Open Data Network hosts is by typing in a search term where that dataset will be listed among the results. For many such pages, search engines end up indexing some datasets as they appear in search results and not others.

Specialization in data-publishing communities: Research has become more interdisciplinary and scientists must be able to find data in the disciplines that they are less familiar with. A professional will most likely know her go-to data repositories in her specialized field by reading relevant papers, by asking colleagues, and through word of mouth. Then as she looks for data, she most likely uses the search tools within a specific data repository, which are usually perfectly suited for that type of data (e.g., social-sciences data, environmental, economic). However, these methods no longer work when you start looking for data in a discipline that you are

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313685

not intimately familiar with. For example, an epidemiologist trying to understand how a new virus spreads may want to understand environmental factors that coincided with the spread of the virus, yet she might not have any colleagues who know the landscape of the environmental data well. If you are a new researcher, or someone who is not very "plugged into" a research community, you might not even have anybody to ask. Just as Google Scholar has revolutionized vertical search for academic publishing, a specialized vertical search engine can fundamentally improve data discovery across all scientific disciplines.

In this paper, we discuss Google Dataset Search (https://g.co/datasetsearch), a search engine over dataset metadata that we built with an open ecosystem at its core: data publishers, large and small, use Schema.org or W3C DCAT markup to describe the metadata semantics on the individual pages of each dataset, a Web crawler collects this metadata, and Dataset Search augments and indexes this metadata to provide a vertical search over all the dataset metadata on the Web. As new publishers add metadata markup to their sites, Dataset Search includes them after the crawler visits and processes the pages. Dataset Search was launched as public beta at the beginning of September 2018.

Specifically, this paper makes the following contributions:

- We **present a search engine over dataset metadata** that can scale to all metadata published on the Web through an open ecosystem.
- We identify a set of **challenges** in building a search engine over structured data in such an open ecosystem.
- We describe our **technical approach** to normalize metadata, to reconcile it to a knowledge graph, to identify duplicates, and to index the data to provide a data-discovery system in the context of the whole Web.
- We discuss practical **lessons learned** from building a vertical search engine in an open ecosystem.

2 RELATED WORK

There are several key lines of work that both informed and enabled our work on Dataset Search.

Many scientific disciplines have come to a consensus (or have been compelled by funding agencies and academic publishers) that it is important to publish data and to cite it. There are, for example, peer-reviewed journals dedicated to publishing valuable datasets, such as Nature Scientific Data [19]. There are efforts such as DataCite [29], which provide digital object identifiers (DOIs) for datasets and both encourage and enable scientists to publish their datasets. Providing mechanisms for citing data enables scientists to get credit for publishing the data. Many scientific communities have signed on to the principles of management and stewardship of the data in a FAIR way (findable, accessible, interoperable, and reusable) [34]. Recently there have been efforts to create standards around citation of datasets. Fenner and colleagues describe a roadmap for implementing the Joint Declaration of Data Citation Principles [7]. Among the principles outlined in the article is the requirement that datasets have a landing page and a recommendation that the landing pages use Schema.org to describe its contents.

Naturally, as scientists pay more attention to publishing data, many scientific communities create data portals for scientists in that community to publish their datasets. These portals are usually designed for the specific type of data in that domain, and focus on the interface and interaction that suits scientists in that domain. There are hundreds of such repositories, with re3data.org, a registry of research data repositories, listing more than 2,000 repositories.

It is no wonder that such proliferation of repositories leads to the problem of data discovery. A qualitative study by Koesten and colleagues [17] found that more than 50% of users who searched for data using a mix of Web search engines, domain-specific repositories, and recommendations from colleagues reported difficulty in finding the datasets that they need. Kacprzak and colleagues [13] analyze the differences between search behavior for datasets and Web searches. Their analysis of query logs from four major open data portals, showed that queries for datasets are short, portal search engines are used in an exploratory manner and that the topics of queries for data differ from topics in Web search queries. Their findings indicate that the tools for querying datasets are not as well developed as Web search engines, and require more human work (e.g., scrolling through more search results returned from short queries) than Web search.

One approach for providing access to multiple data repositories is to harvest the metadata from these repositories through their APIs. For example Open Data Portal Watch [20] collects metadata from more than 260 repositories, focusing on government data. The Open Data Portal Watch developers evaluate and compare the quality of the metadata for these repositories. DataMed [22] is an example of a similar approach in the domain of life sciences, collecting metadata from 75 repositories. These approaches inherently rely on having the portal owners know the list of dataset repositories in advance.

A complementary approach to collecting metadata from a specific set of dataset publishers is to allow dataset publishers to submit their datasets to the repositories. Data-management systems such as CKAN [3], Quandl [26], Kaggle [14], and Microsoft Azure Marketplace [1] are repositories of data from multiple sources, organized for distribution and sharing. In all these systems, dataset owners actively choose to contribute their datasets to the system or to annotate the datasets with metadata. However, in all these systems, if dataset owners want their dataset metadata to appear in more than one repository, they need to submit the description of their dataset in each of those repositories separately.

The distinguishing characteristic of Dataset Search is the open ecosystem: the owners of datasets or data repositories need to markup the metadata only once, in a place where they publish and maintain it on their own site. Once they do, any Web crawler can pick it up and the datasets from this new source will show up in Dataset Search, or in other tools that rely on this metadata. In other words, repository owners do not need to submit their datasets anywhere proactively or to have a data portal know about the existence of their repository. As long as their Web pages are accessible to a general Web crawler and have the structured metadata on the pages, their datasets will be included. Researchers have explored this "post-hoc" approach in the context of enterprise data management [12]. To the best of our knowledge, Dataset Search is the first such approach for datasets in the context of the entire Web.

We chose to rely primarily on Schema.org for describing dataset metadata because both search engines and open-source tools have used it successfully to build an open ecosystem for various types

of content [11]. Erickson and colleagues [5] first proposed to use Schema.org to describe open government data and provided the first definitions for `http://schema.org/Dataset` based on W3C DCAT [4]. And in recent years, scientific community has also embraced it for publishing data: There are mappings from other metadata standards to Schema.org. For example, Sansone and colleagues define a mapping from the DATS standard in the biomedical community to Schema.org [30]. Wang and colleagues use Schema.org to describe research-graph data, which captures researchers, datasets and scholarly articles [33]. Efforts such as bioschemas.org [10] extend Schema.org to include domain-specific terminology and relationships. All these efforts enabled Dataset Search to start with an ecosystem that already had some metadata in Schema.org. Our team could then work with these communities to build on the existing efforts to encourage adoption.

3 DEFINING THE PROBLEM

In order to define the problem of dataset search on the Web and to highlight the key requirements for a product that would support it, we conducted interviews with scientists, journalists, and other data consumers. Many of them emphasized that outside of any niche community, nobody can enumerate all the dataset repositories that exist, even in a single discipline. Therefore, our goal was to enable an open ecosystem that anybody can join at any time: from a large institutional repository to a group of students who want to publish the results of an experiment.

When defining the problem of building a search engine over datasets, the first definition we must consider is the definition of a dataset, and, more specifically, a **dataset as published on the Web**. For the purposes of the work described in this paper, we have adopted an operational rather than a declarative definition: anything that a data provider considers to be a dataset is a dataset. This definition means that we can include individual tables, files, images, binary files, maps, or collections of any of the above. On the one hand, this definition removes the need to "police" what a dataset is and allows providers in different domains to share data in a format that makes sense to them (e.g., binary NetCDF for climate data). On the other hand, this approach puts a bigger burden on tools such as Dataset Search to ensure that its users get high quality and meaningful results.

Furthermore, we focus on the problem of search over dataset **metadata**, which includes data *about* a dataset. Metadata describes salient properties of a dataset, such as its title and description, provider, spatial and temporal coverage, and so on (see http://schema.org/Dataset for a sample of metadata properties).

We define the task of **searching for datasets on the Web** in the following way: Given a set of Web pages that publish dataset metadata, unknown in advance, build a search engine over this metadata to enable users to find datasets on those pages. Specifically, this problem setting highlights the following requirements:

- The system needs to be open, allowing any new provider to join by publishing their metadata.
- The search is over metadata, and does not have to include the data itself. Indeed, a data provider may require the users to pay for data or to create a free account to obtain access (the licensing terms themselves are part of the metadata).
- The metadata must be published by the dataset publishers themselves, using a standard that our and other solutions can interpret.

4 TECHNICAL CHALLENGES

We now describe the technical challenges that we had to address in building Dataset Search. Most of these challenges would apply to building any large-scale vertical search engine in an open ecosystem, and are not specific to datasets.

4.1 Metadata quality

At its core, Dataset Search is a search engine over metadata provided by data publishers on the Web. As a result, the quality of the metadata varies greatly. We have observed that *"everything that can go wrong, will go wrong"* when you are operating at this scale in an open ecosystem. For a simple example, consider representation of dates, which appear in several properties of dataset metadata, including publication date of a dataset, its modification date, and its temporal coverage. The Schema.org standard requires that dates follow the ISO 8601 format (e.g., `"2018-11-01"`). However, we have found dates in every possible format, including sentences such as `"Published in December, 2015"`—in different languages. Similarly, spatial-coverage definitions in metadata may include latitude and longitude for points defining a geographic shape. We have observed cases were providers mixed up the order of latitude and longitude. In some cases, dataset title and its identifier were swapped, and so on. Because metadata is usually generated programmatically, such problems usually affect all metadata from a given repository, which may include hundreds or thousands of datasets.

The Schema.org specification also leaves some details about how the providers should specify the metadata open to interpretation. For instance, one can specify `encodingFormat` either at the level of a dataset, or at the level of `distribution`, which is a property of a `http://schema.org/Dataset`. In the context of dataset publishing, the distinctions between publisher and provider are often blurred and imprecise. Other such examples abound. Thus, to make the search useful, we must clean up and normalize the metadata as much as we can (Section 5.2).

4.2 Metadata duplication in search results

In addition to the quality of individual metadata, the pages where metadata appears can also cause problems. Many dataset repositories are themselves search engines over the metadata for the datasets that they host. Therefore, a description of each dataset appears in two contexts in the repository pages: (1) in pages listing search results, and (2) on dataset profile pages for each individual dataset. We refer of these descriptions within the same site as **duplicates**. The potential number of pages in the first category is exponential, whereas the second category has one page for each dataset. Both types of pages are usually generated programmatically from a database or an index. We found that frequently developers attach metadata to each dataset in search-result listing, and may or may not attach it to the profile page. As a result, a crawler may potentially pick up an exponentially large number of copies of metadata for the same dataset, one from each search-result page where that dataset appears. However, from the point of view of a

dataset-search tool, the page that we really want to take the user to is the profile page for a dataset, and not a listing of search results. Thus, we must distinguish between these different types of pages and identify which metadata copy refers to the profile page (Section 5.4).

4.3 Dataset replication and provenance

While having multiple copies of metadata descriptions for the same dataset within a single repository is undesirable (Section 4.2), replication of the description of the same dataset in *different* repositories not only happens frequently but also provides a signal about the quality of the dataset itself. The presence of a dataset in multiple repositories often signals that the dataset is widely used and provides some indication about its credibility. We refer to the copies of metadata descriptions of the same dataset in different repositories as **replicas**. If we can cluster the replicas present in different repositories, we can give users a choice of which repository to go to. Hence, we face the challenge of identifying the replicas across repositories. The Schema.org standard provides a way to specify the original dataset explicitly through a `http://schema.org/sameAs` link: a dataset description can point to the original dataset description on another site through this link. Relying on `sameAs` links offers only a small part of the solution: First, the links are not always reliable. Second, when we examine the metadata that we collected, we find that less than 1% of dataset descriptions have explicit `sameAs` links to other domains, whereas we identified that about 25% of them are likely replicas of another dataset. Thus, we must rely on heuristics and other properties that could serve as proxies for `sameAs` as we cluster the replicas (Section 5.4).

4.4 Churn and stale sites

Search engines optimize their crawl schedules to crawl sites that their users visit often at a higher rate than the sites in the long tail [8]. While this optimization is entirely reasonable at Web scale, we need to tune it to improve the discoverability of content from the long tail by taking into account the utility of the metadata. Many of the pages that describe individual datasets are in that heavy long tail of the Web and therefore the problem of churn and stale links is exacerbated in this vertical. Even though we do not run our own crawl and rely on a general Google crawl, we still should not be sending users to stale pages. For the pages that we ultimately identify as dataset pages, we work with the crawl-scheduling team to ensure that we minimize the number of stale links.

While we cannot measure the churn among the datasets that the crawler does not see, we measure the churn in the dataset metadata for the descriptions in our index. For Dataset Search, over the past two months, on any given day, on average, 3% of the datasets are deleted from our index, and 7-10% of new datasets are added. We discuss our approach to addressing this churn efficiently in Section 5.5.

4.5 Ranking and relevance

One of the key research questions in building any vertical search engine is understanding what ranking signals make sense for that vertical. Initial studies indicate that there is a difference in how users expect datasets to be ranked [12, 15]. Because explicit links between dataset metadata are still relatively rare, traditional Web-based ranking methods are not effective. While we hope that eventually citing and referencing dataset will become as common a practice as citing and referencing scientific papers, it is not the case today. Thus, traditional scholarly metrics are not readily applicable either.

In addition, because metadata is often limited and minimalistic, it may not provide enough signal to decide whether a dataset is relevant to the user's query. We can fall back on the information on the rest of the page, if it is more expansive than the metadata, or recover additional metadata from the data itself (Section 5.6).

4.6 Multiple dataset-metadata standards

While Schema.org is widely used by search engines and other applications to improve many Web-based tools that need to rely on semantics of the data on a Web page, it is not the only open standard for describing dataset metadata. Several other standards exist, most notably, the W3C Data Catalog Vocabulary (DCAT) [4]. Mappings between Schema.org and the various extensions to DCAT are currently under discussion at W3C and Schema.org. We found that at the moment, only 2% of dataset descriptions (in JSON-LD, RDFa or Microdata) use the DCAT standard while the rest use Schema.org. However, the datasets that use the DCAT standard include hundreds of thousands of datasets from government portals around the world, and in particular portals with geo-spatial data. Therefore, to get better coverage and to be inclusive of other standards, we process both Schema.org and DCAT metadata, as long as the latter is also represented syntactically in a supported syntax, to allow the regular crawl processing to extract the triples (Section 5.7).

5 IMPLEMENTING DATASET SEARCH

To enable the open ecosystem, we rely primarily on Schema.org, a non-proprietary format to describe metadata on the Web in a structured form, providing basic semantics (Figure 1). As we mentioned in Section 4.6, we also support DCAT. Any dataset provider can add simple markup to Web pages that describe datasets. Our Web crawler then crawls this page and parses the HTML and the embedded markup (either as RDFa, microdata, or JSON-LD). Dataset Search does not require its own crawl; it builds upon the crawl and extraction infrastructure of the general Google Web crawl. Specifically, we rely on the Google Web crawl, which already processes structured-data markup and generates triples for each page, regardless of the specific vocabulary that the providers use (Schema.org, DCAT, etc.). The crawl and extraction of the triples from the pages are not contributions of this paper.

Dataset Search collects the metadata, links it with other resources, and builds an index of this enriched corpus of metadata. Once we built the index, we can start answering user queries—and figuring out which results best correspond to the query.

We will now describe the details of the key components of the pipeline in Figure 1.

5.1 From Schema.org to protocol buffers

Google Web crawl and its associated processing, relying on the underlying standards, parses RDFa, Microdata and JSON-LD to a common graph data model, broadly corresponding to W3C's RDF

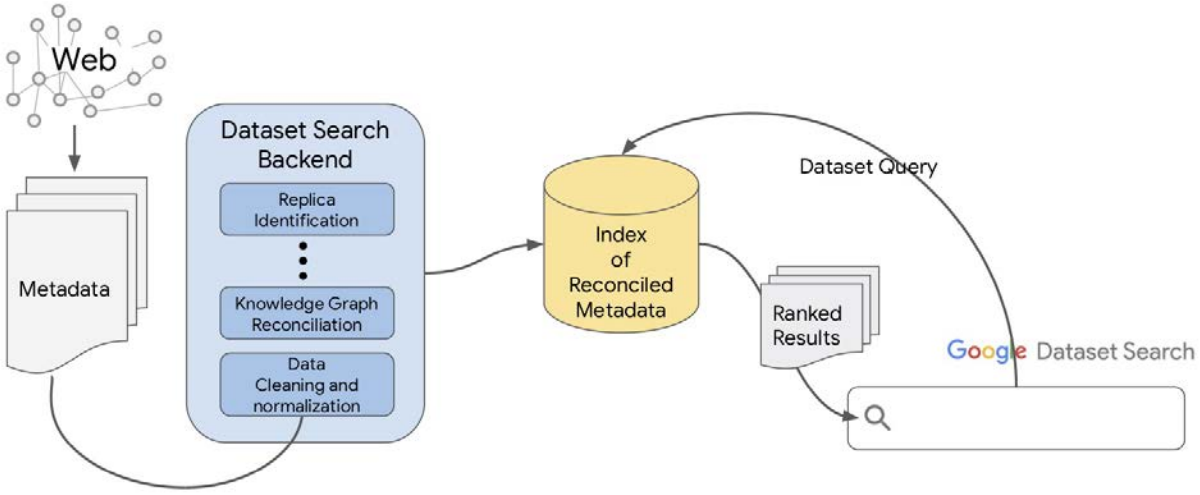


Figure 1: An overview of the Dataset Search components. Google crawler collects the metadata from the Web; Dataset Search backend normalizes and reconciles the metadata; we then index the reconciled metadata and rank results for user queries.

triples [28]. We then look for the triples that use our vocabularies of interest, Schema.org and DCAT. Specifically, we collect all the triples for all the pages that have elements of specific types: `http://schema.org/Dataset`, `http://schema.org/DataCatalog`, and `http://www.w3.org/ns/dcat#Dataset`.

For a set of triples from each page, we traverse the graph to collect all the properties and related objects for each dataset in a protocol buffer [32], a nested-relational record corresponding to each metadata entry. A dataset record can point to other records such as organizations that provided a dataset or a record describing the distribution of a dataset. A single Web page can have multiple dataset records on it.

The specification of the graph traversal captures the mapping from Schema.org and DCAT vocabularies to the corresponding elements in the protocol-buffer definition (e.g., example fields in Figure 2). The schema of the protocol buffer for the metadata largely corresponds to `http://schema.org/Dataset` and therefore the transformation of metadata at this stage is rather small.

To improve scalability, we use the graph query independently on the triples from each individual page rather than try to extract information from a graph that includes all metadata triples on the Web. Because the links across different pages must specify objects on another page directly through a URL (e.g., a provider of this dataset on page *A* is described on page *B*), we can do this reconciliation post-hoc. So, essentially, each page corresponds to its own, possibly disconnected graph. At the same time, doing graph traversal only for a single page is dramatically more scalable.

The information that we extract through graph traversal constitutes the **raw metadata**, metadata that closely mimics the structure of Schema.org properties on the original page.

In the next few steps, we describe how we create **reconciled metadata** for each dataset, accounting for the different levels of quality and variety of the modeling patterns used.

5.2 Normalizing and cleaning the metadata

As we mentioned in Section 4.1, we must assume that we will encounter every possible misuse and mis-interpretation of Schema.org properties when we operate at the scale of the whole Web. Thus, we perform a number of operations to normalize and clean up the metadata.

First, for the properties where we observe different patterns on the Web, we analyze the common patterns used and try to account for all of them. For instance Figure 2 shows the different patterns that we observed for defining downloads and distribution. In the figure, the first example of raw metadata defines the format of the dataset (CSV) at the level of the dataset itself and stores the download URL as the value of the `http://schema.org/distribution` property. Other examples in the figure deal with these two pieces of information differently. All these patterns are commonly used in our corpus. We mine these patterns by traversing either the initial graph or the resulting protocol buffer. Once we identify the patterns, we write adapters to convert all of them into the same modeling pattern in the reconciled metadata record. The right-hand side of Figure 2 shows this reconciled result.

Similarly, we have developed adapters for other metadata fields: We understand a lot more representations of dates than the ISO standard required by the Schema.org specification (Section 4.1. We will pick up digital object identifiers (DOIs) for a dataset from a variety of fields, and not just `http://schema.org/identifier`. We will use a uniform field, `provider`, for the many different fields that dataset providers used to identify this property. As we collect more metadata, our set of such adapters grows. Our decisions in these steps are guided by two factors: (1) the frequent usage patterns that we observed in the data; and (2) our understanding of what we expect the users to see in Dataset Search results.

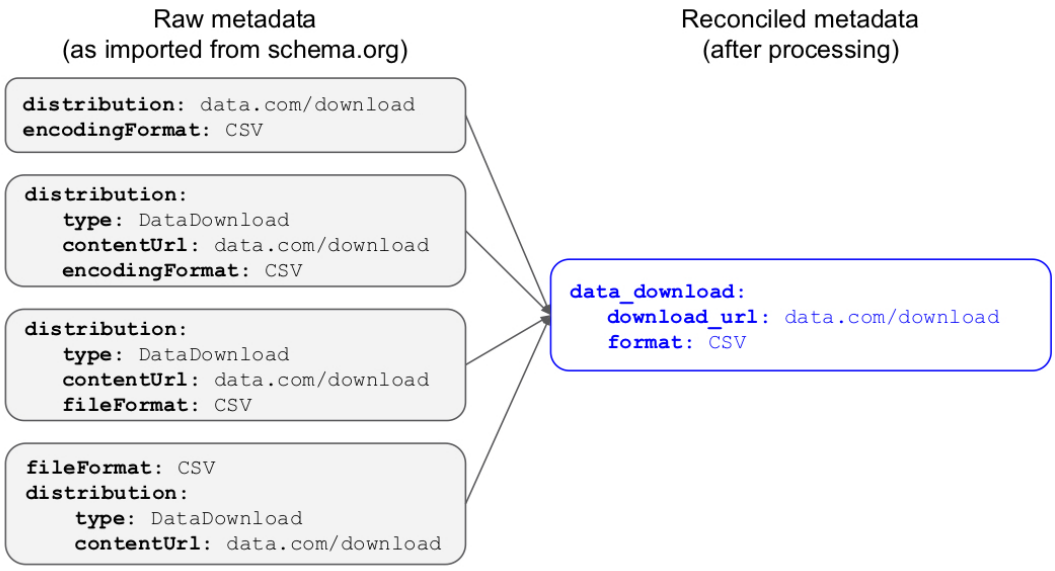


Figure 2: Normalizing raw patterns. The diagram gives an example of the variety that we have observed in metadata definitions. The examples on the left are a protocol-buffer version of the representation, as imported directly from Schema.org metadata. All these examples reconcile to the same pattern, shown on the right.

5.3 Reconciling metadata to a knowledge graph

Most companies today have a knowledge graph that describes types of entities and relationships between them that are critical to the company's products. In our case, such a knowledge graph describes and links information about many entities, including the ones that appear in dataset metadata: organizations providing datasets, locations for spatial coverage of the data, funding agencies, and so on. Therefore, we try to reconcile information mentioned in the metadata fields with the items in the knowledge graph. We can do this reconciliation with good precision for two main reasons. First, we know the types of items in the knowledge graph and the types of entities that we expect in the metadata fields. Therefore, we can limit the types of entities from the knowledge graph that we match with values for a particular metadata field. For example, a provider of a dataset should match with an organization entity in the knowledge graph and not with, say, a location. Second, the context of the Web page itself helps reduce the number of choices, which is particularly useful for distinguishing between organizations that share the same acronym. For example, the acronym CAMRA can stand for "Chilbolton Advanced Meteorological Radar" or "Campaign for Real Ale." If we use terms from the Web page, we can then more easily determine that CAMRA is in fact the Chilbolton Radar when we see terms such as *clouds*, *vapor*, and *water* on the page.

This type of reconciliation opens up lots of possibilities to improve the search experience for users. For instance, Dataset Search localizes results by showing reconciled values of metadata in the same language as the rest of the page. Additionally, it can rely on synonyms, correct misspellings, expand acronyms, or use other relations in the knowledge graph for query expansion.

5.4 Finding duplicates and aggregating replicas

As we discussed in Sections 4.2 and 4.3, the same dataset description may appear in multiple places. We refer to such dataset descriptions as *duplicates* is they occur within the same site. Duplicates usually appear because site developers chose to mark up dataset descriptions both when listing search results and in profile pages for individual datasets. For Dataset Search users we should be indexing only one of the many duplicates within a single site, preferably the one that appears on the profile pages.

When a dataset (e.g., a popular one), appears in different repositories, we refer to these descriptions as *replicas*. Because we have somewhat of a "bird's eye" view of the Web, we can often find where all these different replicas are, and group them together, giving the user a choice of which repository they want to get this dataset from.

While presenting multiple duplicates is not useful for users, replicas from different repositories can be useful for a user, giving her a choice to go to the repository that she trusts or knows.

Our approach to identifying duplicates and replicas are very similar—with the main difference being whether or not we include the site domain into consideration.

First, Schema.org has a way to specify the connection explicitly, through `http://schema.org/sameAs`. In our experience, when this property was present, it was a strong indicator that datasets are the same, whether they are duplicates within the same site or replicas across different sites. Indeed, if developers must include Schema.org for metadata in search results in addition to the profile pages on their site, our guidelines[1] suggest using `sameAs` to point to the profile page.

[1] https://developers.google.com/search/docs/data-types/dataset

Second, we found that sometimes two datasets can be identified as duplicates or replicas not because they point to each other through `sameAs`, but rather because they both point to the same canonical page for that dataset. The latter page itself might not even have Schema.org metadata and hence will not be in our corpus.

Finally, we use a number of other heuristics that, in combination, provide a signal on possible duplicates or replicas: for example, two datasets sharing a large part of their metadata; or pointing to the same URL as their download URL, are likely related. In practice, each one of these signals can fail or be subject to mis-representation or misinterpretation by data providers. So, we use a combination of signals to determine whether or not dataset descriptions appear to be the same.

To identify these duplicates and replicas at scale, we compute a hash value (fingerprint) for each combination of values for a dataset that may be an indication that it is a replica or a duplicate of another dataset. For instance, these combinations of values may include dataset title and description, or dataset title and a download URL. We heavily pre-process the fields that we use to compute the fingerprint: we normalize the text, remove delimiters and any special symbols, and remove spaces. This normalization is usually sufficient because data providers normally copy the metadata directly from one site to another. Once we have the fingerprints for these field combinations, we construct a graph where nodes represent datasets and edges connect nodes that share at least one fingerprint (Figure 3). Our goal is to identify connected components within this graph: each connected component is a cluster of duplicates, if the nodes are from the same site, or replicas, if they are from different sites. In the example in Figure 3, datasets *A*, *B*, and *C* are all duplicates (or replicas). Even though datasets *A* and *B* do not share any fingerprints; they are both the same as dataset *C*. Hence, we can infer that they are part of the same cluster.

We first use this algorithm to compute duplicates, grouping only datasets within a single domain by adding the domain to each fingerprint, thus ensuring that descriptions from different domains will never end up in the same cluster. We then choose one representative from a cluster (Figure 4). We use heuristics to maximize the probability that this representative refers to a profile page for a dataset and not a listing of search results. We then run the same algorithm to compute replicas on this de-duped set. We use a MapReduce version of a connected-component algorithm [27] to compute these clusters efficiently.

Table 1 shows the distribution for different sizes of *replica clusters*. Recall that replicas are cases where the same dataset appears on different sites. The vast majority of clusters have only two replicas, whereas there are a few with as many as 40 or 50 replicas in different domains, with the largest cluster having 58 datasets.

5.5 Scalability of the backend

As with any system that operates at Web scale, we must address scalability of our implementation. We discuss the scalability of the Dataset Search backend from Figure 1, because we could rely on other, already scalable, systems for crawl, indexing, and querying.

Figure 4 highlights the backend architecture. After we traverse the graph of triples on each individual page, we generate a protocol buffer (a record) for each dataset. We then do most of the processing

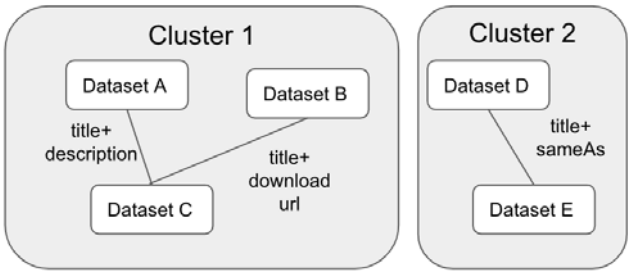


Figure 3: Clusters of replicas or duplicates as connected components. Each pair of datasets may share some fingerprints (e.g., based on title+description). We identify connected components in the graph to cluster all replicas of the same dataset. Table 1 profiles the sizes of clusters that we found.

Table 1: Distribution of replica clusters. Each cluster groups together dataset descriptions from different Web domains that we determine to be replicas.

Number of datasets in a replica cluster	Fraction of clusters
2	82.57%
3	15.17%
4	1.88%
5	0.28%
6 and larger	0.11%

that we described in earlier sections (cleaning, normalization, reconciliation to the knowledge graph) on each dataset independently and in parallel, using MapReduce. Note that for each enrichment that we discuss, we do not need to know how the other records look like. Furthermore, because only a small percentage of datasets changes from day to day (Section 4.4), we do not need to re-process the majority of the datasets, and only re-process the new or modified ones. This architecture is inspired by enterprise dataset search, such as Goods [12].

The identification of duplicates and replicas, however, must be done for the entire corpus each time, because newly added datasets may appear in any of the clusters. Thus, we run the clustering algorithms for identifying both replicas and duplicates daily.

5.6 Indexing and ranking the results

The collection of structured metadata and the replica-aggregation results go into our index. For each dataset, we index the links to its replicas along with its metadata to enable us to retrieve all the information about the whole cluster at the same time. When we collect the results returned from the index, we cluster the replicas together, to show them as a single entry in the list of results.

When the user issues a query, we search through our corpus of datasets using the same methodology as a Web search engine. Just like with any search, we need to determine whether a document is relevant for the query and then rank the documents in terms of how relevant and how important they are. Because there are no large scale studies on how users search for datasets, as a first

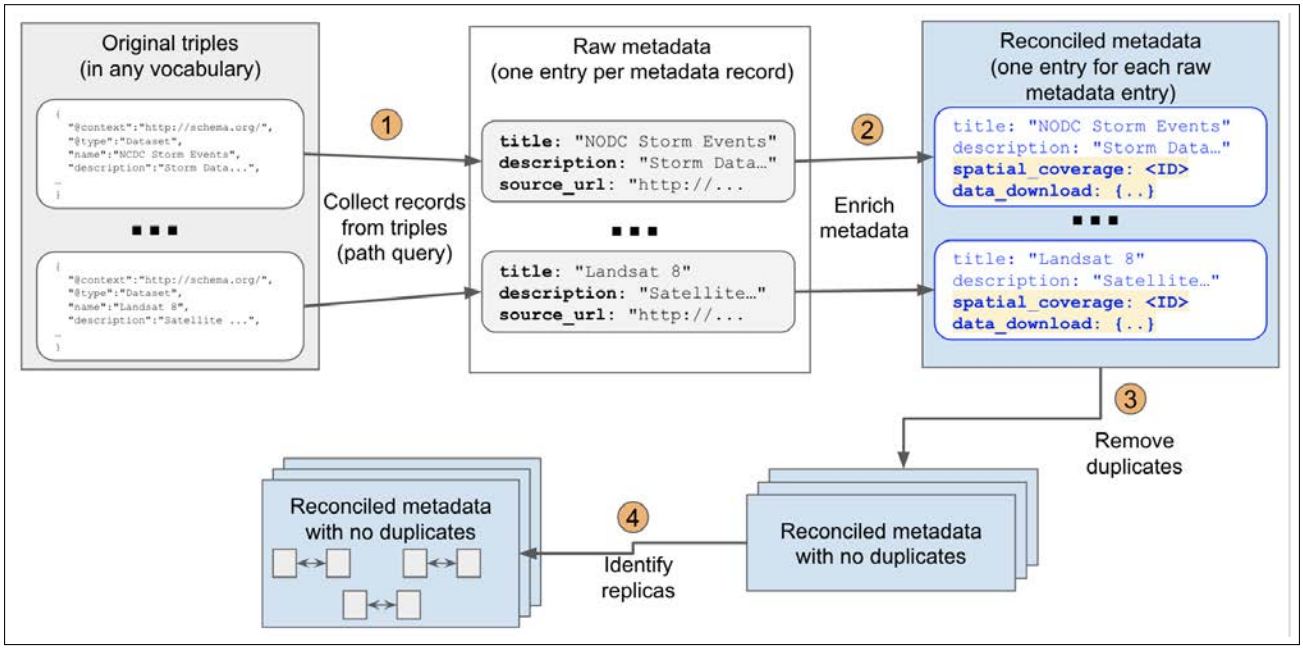


Figure 4: Dataset Search backend: Processing triples on each page results in individual records for each dataset. We then process each record independently to enrich the metadata (e.g., the highlighted portion for reconciled metadata on the right). Steps 1 and 2 happen in parallel, using MapReduce. The algorithms on finding duplicates (step 3) and replicas (step 4) then run on the whole corpus.

approximation, we rely on Web ranking: after all, metadata records come from Web pages. And while this approach provides a good starting point, it is not enough: because many of the pages are in the long tail, the differences in ranking are often not very meaningful. Thus, we add signals that take into account metadata quality. We hope that as we gain more insight on how users query and use Dataset Search, we will be able to develop ranking models that are more specialized for searching datasets.

Finally, Figure 5 shows a screenshot of the user interface: As the user enters a search strings, she gets a list of datasets, with some salient metadata, along with one or more domains where we found that dataset. The page with details for the selected result shows the reconciled metadata.

5.7 Improving coverage and recall

In Section 4, we highlighted challenges to having as broad coverage as possible. First, relying exclusively on the Schema.org vocabulary misses many government sites and we would like to be able to process triples that use other vocabularies or ontologies (Section 4.6). Recall that in the first step of our pipeline, we use a graph traversal to go from a set of triples corresponding to a page to a protocol buffer that represents the record for a dataset metadata. We have extended the graph traversal to understand the DCAT vocabulary as well, using the mapping between the two vocabularies [24]. The DCAT essentially becomes just another vocabulary that we must be able to understand. We referred to Schema.org throughout this paper, mostly to make it easier to read, note that we include the DCAT standard as well. Supporting multiple vocabularies requires some maintenance overhead to ensure that mappings are still valid, we do not expect the number of vocabularies to grow or for the mapping to change frequently. Because most of the tooling is still optimized for Schema.org, we usually still recommend to our partners to use Schema.org if it is feasible for them.

As we mentioned in Section 4.2, in some repositories, search is the only way to get to individual datasets. Thus, it may be hard for a crawler to get a full list of datasets in a repository. Sitemaps have been a common solution to this problem and we have strongly encouraged dataset-repository owners to maintain sitemaps that list all their datasets in our guidelines.[2]

6 THE STATUS OF DATASET METADATA ON THE WEB

Several researchers and practitioners have encouraged the use of Schema.org for dataset discovery on the Web in the last 1-2 years (e.g., [7]). Our team also participated in such efforts [21]. This community building was necessary prior to launching any vertical search based on this metadata: We needed to avoid the cold-start problem of not having any metadata to search over when a product is launched. Our approach involved public blog posts [21] and presentations to raise awareness of problems in data discovery today and the need for standardized Web-based metadata. We have also identified 2-3 disciplines for which we attempted to have more comprehensive coverage. We consulted experts in those disciplines to get a list of prominent repositories in those fields, the repositories

[2]https://developers.google.com/search/docs/data-types/dataset

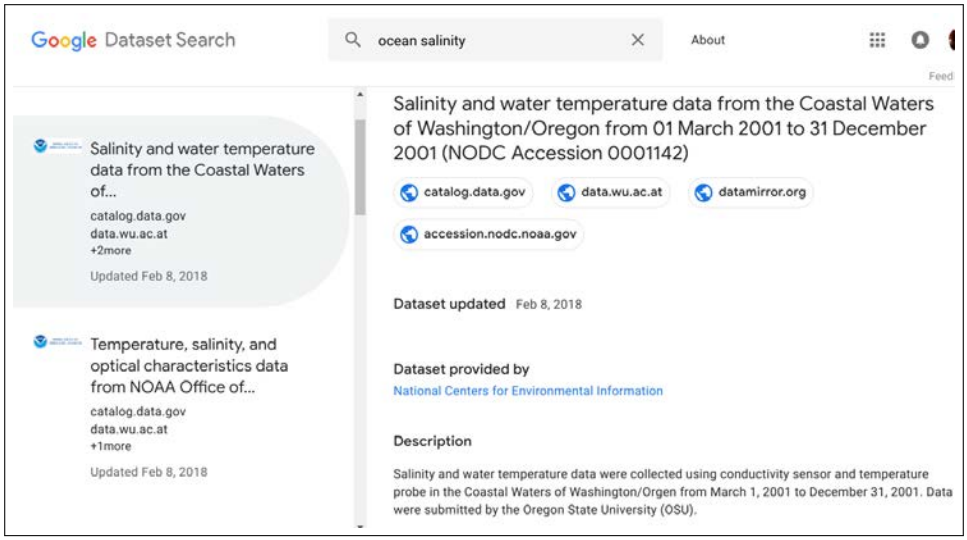


Figure 5: Dataset Search: Query results show the list of matches and the details for each match. The example in the figure shows multiple repositories where the metadata for the selected dataset is available.

without whom any dataset search is not viable. We then worked with the developers of these repositories directly to bootstrap the ecosystem. As the leaders in the domains added the metadata to their pages, developers of other repositories often followed suit.

We found that while having key partners gave us confidence that we will have datasets that many scientists consider important, the majority of domains by the time we launched were not from the partners that we explicitly contacted. At the time of the launch, there were thousands of domains that had `http://schema.org/Dataset` and millions of datasets. As the products that use this metadata begin to appear, the number of domains and datasets is growing: about 30% increase in the number of domains and 90% increase in the number of datasets in the first three months.

7 DISCUSSION

We now discuss the lessons that we learned when building Dataset Search (Section 7.1), the feedback that we received from users after the initial public beta launch (Section 7.2), and future work (Section 7.3).

7.1 Lessons learned

In our efforts to build Dataset Search based on an open ecosystem, we have learned several lessons that we think might be valuable to others building similar vertical search engines or those who work on tools and applications that rely specifically on dataset metadata.

Build an ecosystem first: Often, the immediate impulse of technologists is to build technical solutions for a given problem. In the case of Dataset Search, a key technical problem is to identify and extract metadata for datasets spread across the entire Web. Open information extraction from long tail sources is a known hard problem (e.g., [6, 25, 31]). Therefore, posing the problem of collecting metadata as a solely technical problem makes the solution significantly less feasible. Relying on an open source standard that enables dataset providers to describe their datasets and creating an ecosystem for publishing metadata led to several benefits. First, we have an immediate solution to the problem of collecting metadata at Web scale. This problem would have been a much more daunting and time consuming feat if we used a purely technical solution. Second, the majority of dataset repositories already store this metadata in a structured form in their own database, generating dataset profile pages programmatically from these databases. Rather than trying to reverse-engineer this metadata from a variety of fonts, layouts, frames, and so on, we get much better fidelity by simply having this structured data populate the corresponding metadata fields. Finally, we can think of the ecosystem as a means of bootstrapping a technical solution: The ecosystem provides an abundant supply of data needed to train systems that may be able to assist in identifying more landing pages with metadata or enriching the metadata we have already collected (Section 7.3).

Open non-proprietary standard is key: When deciding which standard to adopt for the metadata markup we chose non-proprietary open standards, Schema.org and DCAT. It takes about the same amount of development or manual effort on behalf of dataset owners or repository developers to create metadata that conforms to any standard, proprietary or otherwise. However, the value of markup that uses a non-proprietary standard potentially can be many times that of a proprietary standard because many different search engines and other applications can use the markup easily. We found that it was much easier to encourage developers to provide markup in non-proprietary standard than it would have been in a proprietary one: many commented that the perceived increased value because markup can be leveraged by multiple applications was important. Just as significant was the understanding that they were not adding the metadata to benefit a single corporation or a single search engine but rather enhancing the discoverability of their data across all tools.

Bootstrapping requires influencers and incentives: One of the hardest aspects of creating a thriving data ecosystem is the process of bootstrapping. It is a chicken-and-egg problem: We need data providers to use the markup and data providers need some meaningful application that uses that markup to justify implementing it. A couple of factors made this task easier for the case of building Dataset Search. First, the open-data community, which builds many of the dataset repositories, generally understands the value of metadata and explicit semantics in its description. For the most part, this audience is as receptive as one could find to these ideas. Second, the encouragement was coming from teams that were planning to deploy the solution at a major search-engine company, Specifically, Google has used Schema.org in many of its products [11]. This history gave the developers a sense of how this metadata may end up manifesting itself in such products. It is important to note, however, that influence does not have to be purely based on traffic or the number of users. The influence may come from funding agencies, for example. We believe that using a non-proprietary standard is necessary but not sufficient for a healthy ecosystem: until the ecosystem is sufficiently large, many data providers will still need some incentive to implement the markup.

Semantics and knowledge graph are critical ingredients: We use lightweight semantics in many parts of Dataset Search. The Schema.org standard itself allows data providers to specify semantics of the content of a Web page using agreed upon vocabulary. In Section 5.3, we discussed the use of semantics to improve the accuracy of reconciliation to the knowledge graph and the use of the knowledge graph itself to add semantics to search, similar to other semantic-search approaches [2]. Furthermore, the key ingredient of our work is the core idea underlying the Semantic Web: resources on the Web should be linked and have their semantics described using shared vocabularies.

7.2 What users want

After collecting user feedback in the first month after the launch, we have learned what features our users seem to desire the most. By far the most requested feature for Dataset Search was to include some sort of faceted browsing. While free text queries are easy to use, users also want to be able to filter the results by date, location, size of a dataset, licensing terms, and other attributes. One of the reasons why we did not include this feature initially was the sparsity of metadata. the majority of dataset descriptions simply lack any values for the fields that one would naturally include in filtering. Thus, filtering results by the values of these sparsely populated fields, will severely impact recall in a faceted search. We hope that as the ecosystem grows and tools like Dataset Search highlight the specific attributes that we use, metadata providers will find it useful to specify values for these attributes, thus making filtered search more feasible.

One of the major components in Dataset Search is the identification of replica datasets across different repositories (Section 5.4). This feature was well received and users have asked that we infer more than simple replica relationships: Not surprisingly, data provenance is key for many researchers who rely on the data [9]. If we can identify the primary or canonical source for the data, we can highlight it better, giving users the choice to go to that source, or to another repository that they are more familiar with.

Finally, users have also requested that we index the data itself in addition to metadata. There are many challenges associated with obtaining, parsing, and indexing the raw data but we hope to include this feature in future iterations of Dataset Search.

7.3 Future work

There are many technical challenges that we need to address to make Dataset Search a more comprehensive and useful tool for dataset discovery.

First, we need to improve the **quality of the metadata** that we have by learning from the existing metadata, by linking to other resources such as academic publications, by understanding data provenance and the evolution of the metadata on the Web. We also need to make sure that the metadata faithfully describes the content of the page: technically nothing stops a data provider from adding `http://schema.org/Dataset` to a page describing a job posting, for instance. And indeed, we have observed many such instances in the data that we crawled. We hope that now that there is a huge variety of dataset pages that are marked up with Schema.org, researchers can use these pages to learn how a page describing a dataset might look like and to build classifiers for such pages.

To improve the **ranking of datasets**, we need to learn from the user interactions. As with any ranking problem, the result users click on for a given query, can help us build new models to improve the ranking.

We hope to improve the **coverage** both by encouraging the growth of the explicit metadata on the Web and using existing metadata for training methods that can extract new metadata.

Finally, we hope that the presence of tools that treat datasets as prominent first-class objects and encourage/reward citing of the data, will lead to an ecosystem where data owners find it valuable and rewarding not only to publish their data but also to describe it better and more fully.

8 CONCLUSIONS AND FUTURE WORK

We have described Dataset Search, a vertical search engine for data discovery that is based on an open ecosystem where data providers describe their metadata in Schema.org, an open non-proprietary standard. Using this community standard has allowed us to grow the coverage quickly, before any applications or products that rely on this metadata even appeared. The response from users to the launch of Dataset search has been extremely positive, not only from scientists and data geeks, but also from journalists and government agencies. We hope that the number of dataset repositories that publish their metadata continues to grow and that other tools start using more actively the metadata that these repositories publish, building other applications that make datasets first-class citizens in scientific and public discourse.

ACKNOWLEDGEMENTS

We would like to thank Xiaomeng Ban, Lee Butler, Thomas Chen, Corinna Cortes, Kevin Espinoza, Archana Jain, Mike Jones, Kishore Papineni, Chris Sater, Gokhan Turhan, Shubin Zhao and Andi Vajda for their work on the project and all our partners, collaborators, and early adopters for their help.

REFERENCES

[1] Azure marketplace. http://datamarket.azure.com/browse/data.

[2] Baeza-Yates, R., Ciaramita, M., Mika, P., and Zaragoza, H. Towards semantic search. In *International Conference on Application of Natural Language to Information Systems* (2008), Springer, pp. 4–11.

[3] CKAN. http://ckan.org.

[4] Data Catalog Vocabulary (DCAT). https://www.w3.org/TR/vocab-dcat/.

[5] Erickson, J. S., Viswanathan, A., Shinavier, J., Shi, Y., and Hendler, J. A. Open government data: A data analytics approach. *IEEE Intelligent Systems 28*, 5 (2013), 19–23.

[6] Etzioni, O., Banko, M., Soderland, S., and Weld, D. S. Open information extraction from the web. *Communications of the ACM 51*, 12 (2008), 68–74.

[7] Fenner, M., Crosas, M., Grethe, J., Kennedy, D., Hermjakob, H., Rocca-Serra, P., Durand, G., Berjon, R., Karcher, S., Martone, M., and Clark, T. A data citation roadmap for scholarly data repositories. *bioRxiv* (2017).

[8] Goel, S., Broder, A., Gabrilovich, E., and Pang, B. Anatomy of the long tail: ordinary people with extraordinary tastes. In *Proceedings of the third ACM international conference on Web search and data mining* (2010), ACM, pp. 201–210.

[9] Goodman, A., Pepe, A., Blocker, A. W., Borgman, C. L., Cranmer, K., Crosas, M., Di Stefano, R., Gil, Y., Groth, P., Hedstrom, M., et al. Ten simple rules for the care and feeding of scientific data. *PLoS computational biology 10*, 4 (2014), e1003542.

[10] Gray, A. J., Goble, C. A., and Jimenez, R. Bioschemas: From potato salad to protein annotation. In *International Semantic Web Conference (Posters, Demos & Industry Tracks)* (2017).

[11] Guha, R. V., Brickley, D., and Macbeth, S. Schema.org: evolution of structured data on the web. *Communications of the ACM 59*, 2 (2016), 44–51.

[12] Halevy, A., Korn, F., Noy, N. F., Olston, C., Polyzotis, N., Roy, S., and Whang, S. E. Goods: Organizing Google's datasets. In *Proceedings of the 2016 International Conference on Management of Data* (2016), ACM, pp. 795–806.

[13] Kacprzak, E., Koesten, L. M., Ibáñez, L.-D., Simperl, E., and Tennison, J. A query log analysis of dataset search. In *Web Engineering* (Cham, 2017), J. Cabot, R. De Virgilio, and R. Torlone, Eds., Springer International Publishing, pp. 429–436.

[14] Kaggle datasets. https://www.kaggle.com/datasets.

[15] Kern, D., and Mathiak, B. Are there any differences in data set retrieval compared to well-known literature retrieval? In *Research and Advanced Technology for Digital Libraries* (Cham, 2015), S. Kapidakis, C. Mazurek, and M. Werla, Eds., Springer International Publishing, pp. 197–208.

[16] Kindling, M., van de Sandt, S., Rücknagel, J., Schirmbacher, P., Pampel, H., Vierkant, P., Bertelmann, R., Kloska, G., Scholze, F., and Witt, M. The landscape of research data repositories in 2015: A re3data analysis. *D-Lib Magazine 23*, 3/4 (2017).

[17] Koesten, L. M., Kacprzak, E., Tennison, J. F. A., and Simperl, E. The trials and tribulations of working with structured data: a study on information seeking behaviour. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2017), CHI '17, ACM, pp. 1277–1289.

[18] Madhavan, J., Ko, D., Kot, Ł., Ganapathy, V., Rasmussen, A., and Halevy, A. Google's deep web crawl. *Proceedings of the VLDB Endowment 1*, 2 (2008), 1241–1252.

[19] Nature scientific data. https://www.nature.com/sdata, 2018.

[20] Neumaier, S., Umbrich, J., and Polleres, A. Lifting data portals to the web of data. In *WWW2017 Workshop on Linked Data on the Web (LDOW2017)* (Perth, Australia, 2017).

[21] Noy, N., and Brickley, D. Facilitating the discovery of public datasets. https://ai.googleblog.com/2017/01/facilitating-discovery-of-public.html, 2017.

[22] Ohno-Machado, L., Sansone, S.-A., Alter, G., Fore, I., Grethe, J., Xu, H., Gonzalez-Beltran, A., Rocca-Serra, P., Soysal, E., Zong, N., and Kim, H.-e. Datamed: Finding useful data across multiple biomedical data repositories. *bioRxiv* (2016).

[23] Open data network. https://www.opendatanetwork.com/.

[24] Perego, A., Friis-Christensen, A., Vaccari, L., and Tsinaraki, C. DCAT-AP to schema.org mapping. Unofficial draft, 2018.

[25] Pujara, J., Miao, H., Getoor, L., and Cohen, W. Knowledge graph identification. In *International Semantic Web Conference* (2013), Springer, pp. 542–557.

[26] Quandl. https://www.quandl.com.

[27] Rastogi, V., Machanavajjhala, A., Chitnis, L., and Sarma, A. D. Finding connected components in map-reduce in logarithmic rounds. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on* (2013), IEEE, pp. 50–61.

[28] RDF 1.1 Concepts and Abstract Syntax. https://www.w3.org/TR/rdf11-concepts/.

[29] Rueda, L., Fenner, M., and Cruse, P. Datacite: Lessons learned on persistent identifiers for research data. *IJDC 11*, 2 (2016), 39–47.

[30] Sansone, S.-A., Gonzalez-Beltran, A., Rocca-Serra, P., Alter, G., Grethe, J. S., Xu, H., Fore, I. M., Lyle, J., Gururaj, A. E., Chen, X., et al. DATS, the data tag suite to enable discoverability of datasets. *Scientific data 4* (2017), 170059.

[31] Suchanek, F. M., Sozio, M., and Weikum, G. Sofie: a self-organizing framework for information extraction. In *Proceedings of the 18th international conference on World wide web* (2009), ACM, pp. 631–640.

[32] Varda, K. Protocol buffers: Google's data interchange format. Tech. rep., Google, 6 2008.

[33] Wang, J., Aryani, A., Wyborn, L., and Evans, B. Providing research graph data in JSON-LD Using Schema.org. In *Proceedings of the 26th International Conference on World Wide Web Companion* (Republic and Canton of Geneva, Switzerland, 2017), WWW '17 Companion, International World Wide Web Conferences Steering Committee, pp. 1213–1218.

[34] Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., et al. The fair guiding principles for scientific data management and stewardship. *Scientific data 3* (2016).

Dealing with Interdependencies and Uncertainty in Multi-Channel Advertising Campaigns Optimization

Alessandro Nuara
Politecnico di Milano, DEIB
Milano, Italy
alessandro.nuara@polimi.it

Nicola Sosio
Politecnico di Milano, DEIB
Milano, Italy
nicola.sosio@mail.polimi.it

Francesco Trovò
Politecnico di Milano, DEIB
Milano, Italy
francesco1.trovo@polimi.it

Maria Chiara Zaccardi
Politecnico di Milano, DEIB
Milano, Italy
mariachiara.zaccardi@mail.polimi.it

Nicola Gatti
Politecnico di Milano, DEIB
Milano, Italy
nicola.gatti@polimi.it

Marcello Restelli
Politecnico di Milano, DEIB
Milano, Italy
marcello.restelli@polimi.it

ABSTRACT

In 2017, Internet ad spending reached 209 billion USD worldwide, while, e.g., TV ads brought in 178 billion USD. An Internet advertising campaign includes up to thousands of sub-campaigns on multiple channels, e.g., search, social, display, whose parameters (bid and daily budget) need to be optimized every day, subject to a (cumulative) budget constraint. Such a process is often unaffordable for humans and its automation is crucial. As also shown by marketing funnel models, the sub-campaigns are usually interdependent, e.g., display ads induce awareness, increasing the number of impressions—and, thus, also the number of conversions—of search ads. This interdependence is widely exploited by humans in the optimization process, whereas, to the best of our knowledge, no algorithm takes it into account. In this paper, we provide the first model capturing the sub-campaigns interdependence. We also provide the IDIL algorithm, which, employing Granger Causality and Gaussian Processes, learns from past data, and returns an optimal stationary bid/daily budget allocation. We prove theoretical guarantees on the loss of IDIL w.r.t. the clairvoyant solution, and we show empirical evidence of its superiority in both realistic and real-world settings when compared with existing approaches.

CCS CONCEPTS

• **Information systems** → **Online advertising**; • **Theory of computation** → *Gaussian processes.*

KEYWORDS

Internet Advertising, Granger Causality, Bid/Budget Optimization

ACM Reference Format:
Alessandro Nuara, Nicola Sosio, Francesco Trovò, Maria Chiara Zaccardi, Nicola Gatti, and Marcello Restelli. 2019. Dealing with Interdependencies and Uncertainty in Multi-Channel Advertising Campaigns Optimization. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313470

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313470

1 INTRODUCTION

Since the early stages of the Internet, one of the most remunerative ways to economically exploit this novel media channel is *online advertising*. For instance, two of the largest companies of the Internet era, namely Google and Facebook, get most of their incomes from the placement of advertisements. In 2017 alone, advertising revenues have totalled about 88 billion USD in the US [16] and about 209 billion USD worldwide. Search is the most important advertising channel, with around 40% of the total spent. The choice of the ads to be displayed and their placement on a webpage are made through auctioning mechanisms [18]. An advertising *campaign* consists of a number (up to thousands) of *sub-campaigns* and a cumulative (per day or month) budget constraint. Each sub-campaign is characterized by an *ad*, a *targeting*, a *bid*, and a *daily budget*, and these last two parameters, crucial for the outcomes of the auctions, can be optimized every day using past performance. Such an optimization process is often unaffordable for humans and its complete, or partial, automation can lead to a significant improvement in revenues.

In addition to search advertising, other advertising channels can be used nowadays, e.g., display and social. The diversification of ad channels is a crucial degree of freedom that one can exploit when setting up an advertising campaign. Indeed, different channels deeply affect each other's performance as Internet users regularly surf from one to another. For instance, Lewis and Nguyen [21] provide empirical evidence that display advertising increases the search activity on a product for some days after the display ads visualization. Kireyev et al. [19] show a similar result between display and search advertising by using the Granger Causality test. The authors also show that this interdependence usually induces delayed dynamics, e.g., an increase in the display advertising impressions can lead to an increase in the conversions of search ads with a delay of some days. Hoban and Bucklin [13] study how display advertising affects user behavior at every level of the marketing funnel [15], while Dinner et al. [4] show that online advertising can even affect the performance of (traditional) non-online advertising. The sub-campaigns interdependence is customarily exploited by experts in the field, e.g., setting up sub-campaigns (called *assist*) not providing direct conversions but increasing the number of conversions on the search engine channel. Besides, capturing the interdependence can provide a direct method for comparing and optimizing

the performance of sub-campaigns on different channels. Indeed, sub-campaigns on different channels need to be evaluated using different performance metrics, and how to combine them is still an open issue. For instance, display and social ads provide very few conversions compared to search ads but allow search ads to generate a larger number of conversions, and therefore an optimization method based only on the number of conversions might not provide optimal allocations. The same holds in the search channel for branding and no-branding sub-campaigns. Although this problem is central in advertising, to the best of our knowledge, no model in the economic literature captures such interdependence.

Related work. Zhang et al. [39] propose an offline joint bid/daily budget optimization algorithm. In addition to neglecting interdependences between sub-campaigns, this work suffers from some weaknesses. In particular, the authors assume a specific family of functions describing the relationship between the parameters and do not provide any theoretical bound on the estimation error. Furthermore, some parameters (e.g., the ad position for each impression and click) cannot be observed by an advertiser, not allowing their estimation and, therefore, the use of the model in practice. Some of these weaknesses have recently been addressed by Nuara et al. [25], who provide an online joint bid/daily budget optimization algorithm called AdComB-TS and exploiting Gaussian Processes [29]. However, this work overlooks the sub-campaigns interdependence. Other related works follow. Thomaidou et al. [32] separate the optimization of the bid from that of the budget, using a genetic algorithm to optimize the budget and then applying some bidding strategies. Markakis and Telelis [22] study the convergence of some bidding strategies. Another related research field is the study of user behaviors from logging data [27, 31], both on social networks [28], and on search engines [37]. However, these approaches assume to keep track of all the actions of each user perfectly, and this is usually unfeasible for most of the advertising tools used by companies. Less related works concern the optimization of the daily budget [17, 36], bidding strategies in display advertising [20, 34, 35, 38], video advertising [10], and targeting [6]. Finally, some works deal with the attribution problem in display advertising [11, 19].

Original contributions. We extend the work by Nuara et al. [25], designing an algorithm based on both learning and optimization techniques that can be adopted for the optimization of real-world Internet advertising campaigns and that, exploiting sub-campaigns interdependence, outperforms the algorithms known so far. To do that, we provide a novel model that, on one side, is expressive enough to capture the interdependences and, on the other side, is simple enough to require few data for its estimation.[1] From of the proposed model, we design a data-driven algorithm, called IDIL, which consists of two phases: the *Interdependence Graph Learning Phase* and the *Estimation and Optimization Phase*. In the former phase, the IDIL algorithm learns the sub-campaigns interdependence structure (represented as a graph), identifying the pairs of sub-campaigns with the most significant interdependences by applying the Granger Causality test. This is crucial since the number of pairs of interdependent sub-campaigns dramatically increases the amount of data required to have accurate estimates of the model parameters. In the latter phase, the IDIL algorithm computes the optimal joint bid/daily budget allocation exploiting Gaussian Process modeling [29] and an *ad hoc* dynamic programming procedure. In particular, Gaussian Processes exploit some form of functional regularity to describe the relationships among the problem parameters, without forcing them belong to a specific family of curves.

Finally, we show that neglecting the sub-campaigns interdependence can lead to massive losses even in simple and common scenarios and we theoretically bound the loss of our algorithm. Furthermore, we experimentally evaluate its performance in both realistic and real-world settings, showing the superiority of its performance compared to the previous approaches that neglect the sub-campaigns interdependence.

2 INTERNET ADVERTISING CAMPAIGN

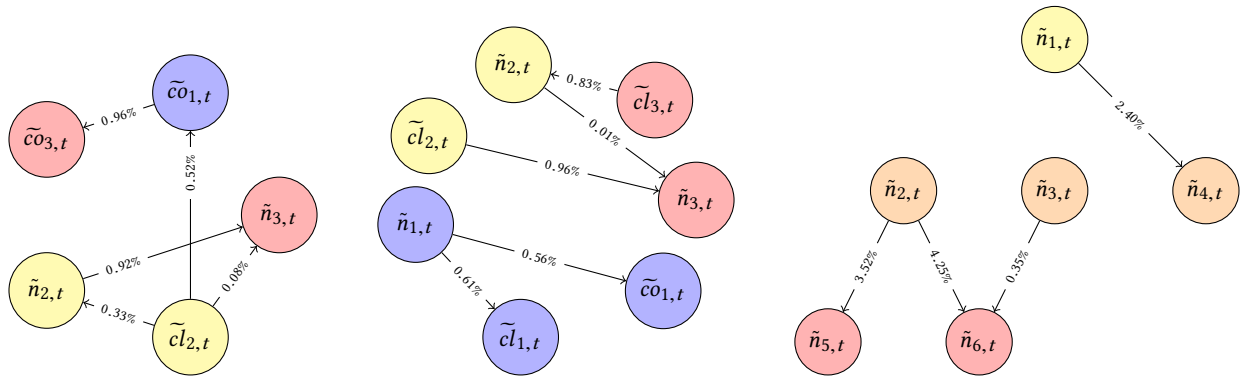
Assume to have an Internet advertising campaign $C = \{C_1, \dots, C_N\}$, with $N \in \mathbb{N}$, where C_j is the j-th sub-campaign. At day t, we are asked to set for each sub-campaign C_j a bid $x_{j,t} \in [\underline{x}_j, \overline{x}_j]$, and a daily budget $y_{j,t} \in [\underline{y}_j, \overline{y}_j]$, subject to that the daily cumulative budget of all the sub-campaigns cannot exceed $Y \in \mathbb{R}^+$. At day $t+1$, we get a report on the performance of the campaign C at the previous day t, which specifies, for every C_j, the tuple $(\widetilde{n}_{j,t}, \widetilde{cl}_{j,t}, \widetilde{co}_{j,t}, \widetilde{c}_{j,t})$, where $\widetilde{n}_{j,t}$ denotes the number of impressions, $\widetilde{cl}_{j,t}$ denotes the number of received clicks, $\widetilde{co}_{j,t}$ denotes the cumulate value of the conversions, and $\widetilde{c}_{j,t}$ denotes the amount of money spent for it.[2]

As aforementioned, both experts in the field of Internet advertising and studies in the Internet economic field, e.g., Kireyev et al. [19] and Hoban and Bucklin [13], demonstrate that impressions, clicks, and conversions of a sub-campaign might be influenced by the same kind of quantities of the other sub-campaigns. We extend the previous studies on the sub-campaigns interdependence, applying the Granger Causality test [12, 33] to two real-world Internet advertising campaigns optimized by an Italian web media agency using the AdComb-TS algorithm [25]. The algorithm, being online, produces policies explorative enough to make the test significant.

At first, we test for Granger causality the data collected for 8 months (from 1/1/2018 to 1/8/2018) from an Internet advertising campaign for a financial service of an insurance company: data correspond to $N = 12$ sub-campaigns, on Google AdWords (search), Facebook (social), and Google display with a cumulative budget of $Y = 600$ Euros. The results obtained from the Granger Causality test are shown in Figure 1, where the most significant elements of $(\widetilde{n}_{j,t}, \widetilde{cl}_{j,t}, \widetilde{co}_{j,t}, \widetilde{c}_{j,t})$ are represented as nodes of different colors according to their specific channel (as detailed in the caption of the figure) and the detected interdependences (with a p-value less than 5%) are represented as directed edges. In particular, Figure 1a shows the results when all the sub-campaigns data are aggregated, while Figure 1b focuses on a specific subset of sub-campaigns who share the same targeting (retired people). These results confirm the presence of the interdependence between display and search advertising as previously observed in the literature. They also show

[1] Due to learning, a sophisticated model may provide poor performance. This is because an excessively complex model could need a large amount of data for the training and collecting such data could require a long time, even longer than the time horizon considered in the optimization process.

[2] We recall that the money spent in one day for a sub-campaign may be different from the daily budget previously allocated.



(a) First dataset, overall people (b) First dataset, only retired people (c) Second dataset

Figure 1: Graphs representing the interdependences of real-world Internet advertising sub-campaigns inferred by Granger Causality test from real data. The numbers on the edges are the p-values (in terms of %) of the Granger Causality test; display ads are depicted in blue, social ads in yellow, and search ads in orange (for branding sub-campaigns) and red (for other search sub-campaigns). The second dataset graph refers to interdependence among impressions $\tilde{n}_{j,t}$ of different sub-campaigns C_j.

that social and search advertising are interdependent and that the interdependences may be targeting specific. Moreover, the interdependence between clicks and impressions of the social channels and the impressions of the search one in this specific scenario seems to be more relevant than others, since they appear in both graphs. Furthermore, the Granger Causality test detects that interdependence dynamics between sub-campaigns are delayed up to 2 days.

At second, we test for Granger causality the data collected for 3 months (from 20/7/2018 to 20/10/2018) from an Internet advertising campaign of a different financial product of the same company with about $Y = 1100$ Euros. There are $N = 14$ sub-campaigns belonging to social and search advertising channels. The resulting graph is depicted in Figure 1c (with a p-value less than 5%). As in the previous dataset, many sub-campaigns are subject to interdependence. In particular, in this case, the interdependence phenomenon is only among impressions, suggesting that these can be the most significant in practice. Moreover, differently from the previous case, we distinguish search sub-campaigns into two subclasses which are at different depths in the marketing funnel: branding (orange nodes) or no-branding (red nodes). Finally, the delay of the interdependence dynamics is up to 3.

3 OPTIMIZATION PROBLEM

We provide our optimization problem capturing the sub-campaigns interdependence. For the sake of presentation, we focus on the interdependence between the impressions of different sub-campaigns.[3] Our goal is the maximization of the revenue earned each day from an Internet advertising campaign subject to a cumulative budget constraint. Formally, given a campaign C and a cumulative daily budget of Y, we aim to find, at day t, the value of bid $x_{j,t}$ and the value of daily budget $y_{j,t}$ for every sub-campaign C_j that maximise the revenue by solving the following optimization problem:

$$\max_{x_{j,t}, y_{j,t}} \sum_{j=1}^{N} v_j \; w_j \; n_j(x_{j,t}, y_{j,t}, u_{j,t}) \tag{1a}$$

$$\text{s.t.} \sum_{j=1}^{N} y_{j,t} \leq Y \tag{1b}$$

$$\underline{x}_j \leq x_{j,t} \leq \overline{x}_j \qquad \forall j \tag{1c}$$

$$\underline{y}_j \leq y_{j,t} \leq \overline{y}_j \qquad \forall j \tag{1d}$$

where $n_j(x_{j,t}, y_{j,t}, u_{j,t})$ is the expected number of impressions given bid $x_{j,t}$, daily budget $y_{j,t}$, and *influence index* $u_{j,t}$, representing the influence of other sub-campaigns towards C_j and computed by using the number of impressions of those sub-campaigns that are interdependent with sub-campaign C_j (see below); w_j and v_j are the click-trough rate and the value per click for the sub-campaign C_j, respectively, and, therefore, $v_j \; w_j \; n_j(x_{j,t}, y_{j,t}, u_{j,t})$ is the revenue provided by sub-campaign C_j.[4] We denote with $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{u}^*)$ the optimal solution to the optimization problem.

To model the sub-campaigns interdependence, we define:

Definition 1. *Given an advertising campaign C, an* interdependence graph $\mathcal{G} := (C, D)$ *is a graph in which the adjacency matrix $D = \{d_{ij}\}, D \in \{0,1\}^{N \times N}$ has elements $d_{ij} = 1$ iff the sub-campaign C_i influences the performance of the sub-campaign C_j.*

We assume that the graph $\mathcal{G}$ is a Directed Acyclic Graph (DAG), i.e., there are no dependency cycles among the sub-campaigns. This assumption is supported by the model of the marketing funnel, in which the majority of the users flows from the top to the bottom, and different advertising channels are positioned at different levels of the funnel. Without loss of generality, we assume that the order over the indices of the sub-campaigns is one of the topological

[3] The use of such quantities is also supported by the experimental results of Section 2, where the interdependence among impressions is the most significant. However, different models, e.g., including the interdependence between the clicks and the conversions, are straightforward extensions of what is proposed in this section.

[4] The optimization problem in Equations (1a)–(1d) reduces to the one by Nuara et al. [25] when there is no interdependence, i.e., if $n_j(x_{j,t}, y_{j,t}, u_{j,t}) = n_j(x_{j,t}, y_{j,t})$, for every C_j.

orders induced by $\mathcal{G}$. Given the interdependence graph $\mathcal{G}$, a formal definition of the influence index $u_{j,t}$ is:

$$u_{j,t} := \frac{1}{K} \sum_{i=1}^{j-1} \sum_{h=t-1}^{t-K} d_{ij}\, n_i(x_{i,h}, y_{i,h}, u_{i,h}), \tag{2}$$

where K is a maximum lag order, meaning that users are influenced by ads at most for K consecutive days. Notice that the first sub-campaign C_1, being influenced by no other sub-campaign, has $u_{1,t} = 0$ since the first summation in Equation (2) is over an empty set. The above definition of $u_{j,t}$ is based on the assumption that the increase in the number of impressions provided by a user coming from any sub-campaign influences the number of impressions of C_j in the same way. While this assumption might seem simplistic, it is necessary to keep at a pace the complexity of training the model. Indeed, a more complex model, e.g., where there is a different influence index for every pair of sub-campaigns, might be an option, but this would require an excessively large amount of data for the training of the model, which is not a viable option within the time horizon of the optimization process.

The optimization problem in Equations (1a)-(1d) can be solved using dynamic programming techniques, once all its parameters are known. However, the advertiser does not know the function $n_j(\cdot,\cdot,\cdot)$ that returns the number of impressions for sub-campaign C_j, as well as its click-trough rate w_j and its value per click v_j. Therefore, we resort to learning techniques to produce estimates of these parameters relying on historical data. We assume to have a dataset $Z := \{\boldsymbol{z}_{j,t}\}$ of τ samples that provides, for each day $t \in \{1,\dots,\tau\}$ and each sub-campaign C_j with $j \in \{1,\dots,N\}$, the following values: $\boldsymbol{z}_{j,t} := (\tilde{x}_{j,t}, \tilde{y}_{j,t}, \tilde{n}_{j,t}, \tilde{cl}_{j,t}, \tilde{co}_{j,t}, \tilde{c}_{j,t})$. This is a tuple with the used bid $\tilde{x}_{j,t}$ and daily budget $\tilde{y}_{j,t}$, the received impressions $\tilde{n}_{j,t}$, clicks $\tilde{cl}_{j,t}$, values of the conversions $\tilde{co}_{j,t}$, and costs $\tilde{c}_{j,t}$. We require that the data collected up to day τ to be exploratory enough to properly model the sub-campaigns interdependences.

4 THE IDIL ALGORITHM

The pseudo-code of the IDIL algorithm is provided in Algorithm 1. It requires a dataset Z and two confidence levels $\alpha_{ADF} \in (0,1)$ and $\alpha_{GC} \in (0,1)$ in input. The first phase of the algorithm (Lines 1–8) is called *Interdependence Graph Learning Phase* and is devoted to learning the interdependence graph of the sub-campaigns. The output of this phase is an estimate $\hat{D}$ of the actual adjacency matrix D. The second phase of the algorithm (Lines 9–13) is called *Estimation and Optimization Phase* and is devoted to the estimation of the parameters for each sub-campaign C_j (i.e., $\hat{n}_j(\cdot,\cdot,\cdot), \hat{v}_j, \hat{w}_j$), using Gaussian Process [29] modeling, and solving the optimization problem in Equations (1a)-(1d), once the parameters have been replaced with their estimates. The outputs of this phase are $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{y}}^*, \hat{\boldsymbol{u}}^*)$, i.e., the optimal bid, daily budget, and influence index for each sub-campaign.

4.1 Interdependence Graph Learning Phase (Lines 1–8)

The task of learning $\hat{D}$ is obtained by resorting to the Granger Causality test [12]. This test has been used in many different fields to infer the structure among datastreams, e.g., sensor networks

Algorithm 1 IDIL

Input: dataset Z, confidence α_{ADF}, confidence α_{GC}
Output: optimal bid/budget/new user allocation $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{y}}^*, \hat{\boldsymbol{u}}^*)$

▷ *Interdependence Graph Learning Phase*
1: **for** $j \in \{1,\dots,N\}$ **do**
2: $\quad adf_j \leftarrow \mathsf{ADF}(\tilde{\boldsymbol{n}}_j, \alpha_{ADF})$
3: $d_{\max} \leftarrow \max_j\{adf_j\}$
4: $\hat{P} \leftarrow 0$
5: **for** $j \in \{1,\dots,N\}$ **do**
6: $\quad$ **for** $i \in \{j+1,\dots,N\}$ **do**
7: $\quad\quad \hat{p}_{i,j} \leftarrow \mathsf{GCT}(\boldsymbol{n}, i, j)$
8: $\hat{D} \leftarrow \mathsf{DAG}(\hat{P}, \alpha_{GC})$

▷ *Estimation and Optimization Phase*
9: **for** $j \in \{1,\dots,N\}$ **do**
10: $\quad \hat{n}_j(\cdot,\cdot,\cdot) \leftarrow \mathsf{GP}(Z, \hat{D}, j)$
11: $\quad \hat{v}_j \leftarrow \frac{1}{\tau}\sum_{h=1}^{\tau} \frac{\tilde{co}_{j,t}}{\tilde{cl}_{j,t}}$
12: $\quad \hat{w}_j \leftarrow \frac{1}{\tau}\sum_{h=1}^{\tau} \frac{\tilde{cl}_{j,t}}{\tilde{n}_{j,t}}$
13: $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{y}}^*, \hat{\boldsymbol{u}}^*) \leftarrow \mathsf{OPT}(\hat{\boldsymbol{n}}, \hat{\boldsymbol{v}}, \hat{\boldsymbol{w}}, \hat{D})$
14: **return** $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{y}}^*, \hat{\boldsymbol{u}}^*)$

by Alippi et al. [1] and Roveri and Trovò [30], and economics by Calderón and Liu [2]. While in its original formulation the test assumes that the analysed time series are stationary, we rely on a generalization of this test, proposed by Toda and Yamamoto [33], which is suitable for integrated and cointegrated time series.

The basic idea of this approach is to estimate a Vector AutoRegressive model of order $K_{GR} + d_{\max}$ for the vector $(\widetilde{n}_{1,t},\dots,\widetilde{n}_{N,t})$, where $d_{\max} \in \mathbb{N}$ is the maximum integration order of the time series that we analyse and $K_{GR} \in \mathbb{N}$ is a lag order which is estimated from the data.[5] The use of $K_{GR} + d_{\max}$ lags ensures that the test statistic used in the Granger Causality test for stationary time series has the same asymptotic distribution of the stationary case and, therefore, statistically valid conclusions can be drawn. More specifically, to test if the impressions of the campaign C_i influence the impressions of the campaign C_j, we estimate the parameters a_{jlm}, for each $m \in \{1,\dots,K_{GR}+d_{\max}\}$, of the model:

$$\widetilde{n}_{j,t} = \sum_{l=1}^{N} \sum_{m=1}^{K_{GR}+d_{\max}} a_{jlm}\, \widetilde{n}_{l,t-m} \quad \forall h \in \{1,\dots,N\}$$

and we test for the hypothesis:

$$H_0 : \forall m \in \{1,\dots,K_{Gr}\}\ a_{jim} = 0,$$
$$H_1 : \exists m \in \{1,\dots,K_{Gr}\} \mid a_{jim} \neq 0.$$

The complete description of this procedure is provided by Toda and Yamamoto [33]. The test states that if we reject H_0 there is

[5] d_{max} can be estimated using the Augmented Dickey Fuller test [3], which requires a confidence level $\alpha_{ADF} \in (0,1)$, while K_{GR} can be estimated from the dataset Z by standard techniques, see Ozcicek and Mcmillin [26] for details.

evidence, with confidence α_{GC}, that the impressions from C_i are influencing those of C_j.

The IDIL algorithm works as follows. For each sub-campaign, we estimate $d_{\max}$ performing the Augmented Dickey Fuller test $\mathsf{ADF}(\tilde{\boldsymbol{n}}_{\boldsymbol{j}}, \alpha_{ADF})$ on the time series $\tilde{\boldsymbol{n}}_{\boldsymbol{j}} := (\tilde{n}_{j,t}, \ldots, \tilde{n}_{j,\tau})$ with confidence α_{ADF} (Lines 1–3), and inferring the time series order adf_j, and, finally, we perform the Granger Causality test on each pair of sub-campaigns (Lines 5–7). The result of this procedure is a matrix $\hat{P}$ containing the p-values of the pairwise tests, which is used to generate a valid estimate of the adjacency matrix $\hat{D} \in \{0,1\}^{N \times N}$. This operation is performed by $\mathsf{DAG}(\hat{P}, \alpha_{GC})$ (Line 8) by selecting the largest subset S of the p-values $\hat{p}_{ij} < \frac{2\alpha_{GC}}{N(N-1)}$ s.t. the matrix $\hat{D} := \{d_{ij} = 1 \text{ iff } p_{ij} \in S\}$ to correspond to a DAG.[6] This procedure ensures an overall confidence α_{GC} on the Granger Causality test, thanks to the Bonferroni correction for multiple tests, and it avoids that the presence of false positives in the detection of interdependences. Indeed, the edges generated by false positive detections might provide adjacency matrices $\hat{D}$ whose corresponding graph presents cycles, which would compromise the execution of the following optimization procedure.

4.2 Estimation and Optimization Phase (Lines 9–13)

The second phase of the IDIL algorithm exploits predictive models to estimate unknown functions and quantities in the optimization problem defined in Equations (1a)-(1d), and solves it in a dynamic programming fashion with an *ad hoc* procedure.[7]

We use Gaussian Processes (GPs) to compute, for each sub-campaign C_j, the function $\hat{n}_j(x, y, u)$ estimating the expected number of impressions $n_j(x, y, u)$, given the chosen bid x, the allocated budget y, and the influence index u generated by the sub-campaigns influencing the sub-campaign C_j (Line 10). The estimate $\hat{w}_j$ of the click-through rate w_j and the estimate $\hat{v}_j$ of the value per click v_j are the average ratios between the number of clicks and the number of impressions and between the number of conversions and the number of clicks, respectively (Lines 9-12). Finally, the estimated influence index is computed as follows:

$$\hat{u}_{j,t} := \frac{1}{K_{GR}} \sum_{i=1}^{j-1} \sum_{h=t-1}^{t-K_{GR}} \hat{d}_{ij}\, \hat{n}_i(x_{i,h}, y_{i,h}, \hat{u}_{i,h}), \tag{3}$$

where we use K_{GR}, obtained from the Granger Causality test, as an estimate of the actual lag K.

The optimization procedure is an extension of the optimization algorithm by Nuara et al. [25], to handle also campaigns in which the revenue given by a budget allocated to a sub-campaign depends on the budget allocated to other sub-campaigns. The OPT algorithm, presented in Algorithm 2, takes as input the estimates of the adjacency matrix $\hat{D}$, the number of impressions function $\hat{n}_j(\cdot,\cdot,\cdot)$, the click-trough rate $\hat{w}_j$, the value per click $\hat{v}_j$, and a set of available daily budget values $\{b_1, \ldots, b_B\}$, which are, for simplicity, evenly spaced in the range $[0, Y]$.

[6] An adjacency matrix $\hat{D}$ identifies a DAG if and only if a depth-first search of the corresponding graph yields no back edges.

[7] For the sake of presentation in what follows we assume that the number of impressions is monotonically increasing in the influence index. A version of the optimization procedure able to handle general cases is discussed in the final part of this section.

Algorithm 2 $\mathsf{OPT}(\hat{\boldsymbol{n}}, \hat{\boldsymbol{v}}, \hat{\boldsymbol{w}}, \hat{D})$

Input: estimated adjacency matrix $\hat{D}$, estimated models $\hat{n}_j(\cdot,\cdot,\cdot)$, $\hat{w}_j$, $\hat{v}_j$, discretization of the total budget $\{b_1, \ldots, b_B\}$
Output: optimal bid/budget allocation $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{y}}^*, \hat{\boldsymbol{u}}^*)$

1: **for** $i \in \{1, \ldots, B\}$ **do**
2: $\quad \Pi_{1,i,1} \leftarrow (b_i\ \mathbf{0}_{N-1})$
3: $\quad L_{1,i,1} \leftarrow \hat{v}_1\ \hat{w}_1\ \hat{n}_1(\chi_1, b_i, 0)$
4: $\quad M_{1,i,1} \leftarrow \hat{n}_1(\chi_1, b_i, 0)\ \hat{\boldsymbol{d}}_1$
5: **for** $j \in \{2, \ldots, N\}$ **do**
6: $\quad$ **for** $i \in \{1, \ldots, B\}$ **do**
7: $\quad\quad c \leftarrow 1$
8: $\quad\quad$ **for** $k \in \{1, \ldots, i\}$ **do**
9: $\quad\quad\quad m = |\{\Pi_{j-1,k,h}\}_h|$
10: $\quad\quad\quad$ **for** $h \in \{1, \ldots, m\}$ **do**
11: $\quad\quad\quad\quad l \leftarrow \hat{n}_j(\chi_j, b_i - b_k, M_{j-1,k,h}(j))$
12: $\quad\quad\quad\quad \bar{\Pi}_c \leftarrow (\mathbf{0}_{j-1}\ (b_i - b_k)\ \mathbf{0}_{N-j}) + \Pi_{j-1,k,h}$
13: $\quad\quad\quad\quad \bar{L}_c \leftarrow \hat{v}_j\ \hat{w}_j\ l + L_{j-1,k,h}$
14: $\quad\quad\quad\quad \bar{M}_c \leftarrow\ l\ \hat{\boldsymbol{d}}_j + M_{j-1,k,h}$
15: $\quad\quad\quad\quad c \leftarrow c + 1$
16: $\quad\quad c \leftarrow 1$
17: $\quad\quad$ **for** $h \in \{1, \ldots, |\{Lt_c\}_c|\}$ **do**
18: $\quad\quad\quad$ **if** $\nexists k \mid \bar{L}_h < \bar{L}_k \wedge \forall p \in \{j+1, \ldots, N\} | \bar{M}_h(p) < \bar{M}_k(p)$ **then**
19: $\quad\quad\quad\quad \Pi_{j,i,c} \leftarrow \bar{\Pi}_h$
20: $\quad\quad\quad\quad L_{j,i,c} \leftarrow \bar{L}_h$
21: $\quad\quad\quad\quad M_{j,i,c} \leftarrow \bar{M}_h$
22: $\quad\quad\quad\quad c \leftarrow c + 1$
23: **for** $j \in \{1, \ldots, N\}$ **do**
24: $\quad \hat{y}^*_j \leftarrow \max_i \Pi_{N,i,1}(j)$
25: $\quad \hat{u}^*_j$ computed as in Equation (3)
26: $\quad \hat{x}^*_j = x^*_j(\hat{y}^*_j, \hat{u}^*_j)$
27: **return** $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{y}}^*, \hat{\boldsymbol{u}}^*)$

The OPT algorithm uses three structures Π, L, and M defined as follows: $\Pi_{j,i,h}$ is a vector that specifies a partial budget allocation with cumulative budget of b_i among the sub-campaigns $C_1, \ldots, C_j$; $L_{j,i,h}$ is the revenue provided by the partial budget allocation $\Pi_{j,i,h}$; $M_{j,i,h}$ is a vector that specifies the value of the influence index of the sub-campaigns $C_{j+1}, \ldots, C_N$ provided by the sub-campaigns $C_1, \ldots, C_j$ when the partial allocation $\Pi_{j,i,h}$ is used. The third index h in the structures mentioned above is necessary since the algorithm may need to store multiple partial budget allocations for each j and i. More precisely, the set $\{\Pi_{j,i,h}\}_h$ contains Pareto-efficient partial budget allocations, where the optimality criteria are the revenue and the influence indices of campaigns $C_{j+1}, \ldots, C_N$. For instance, given two partial budget allocations Π_{j,i,h_1} and Π_{j,i,h_2}, where the former has high revenue and a small number of impressions and the latter *vice versa*, it is not possible to decide which one is the optimal before evaluating their influence on the sub-campaigns $C_{j+1}, \ldots, C_N$ and therefore we need to store both.

At first, the algorithm initializes the values of the structures for $j = 1$ (Lines 1–4), corresponding to the allocations of the partial budget to the sub-campaign C_1. For each budget b_i, we allocate

it to C_1, formally, $\Pi_{1,i,1} = (b_i, \mathbf{0}_{N-1})$, where $\mathbf{0}_{N-1}$ denotes a null vector of size $N-1$. The sub-campaign C_1, being the first in the topological ordering induced by $\hat{D}$, is not subject to any interdependence from other sub-campaigns. Therefore, the computation of the revenue $\{L_{1,i,1}\}_i$ and the influence index vector $\{M_{1,i,1}\}_i$ is performed using the previously estimated models.[8] The vector $M_{1,i,1}$ is computed as $M_{1,i,1} = n_1(\chi_1, b_i, 0)\ \hat{\boldsymbol{d}}_1$, where $\hat{\boldsymbol{d}}_i$ is the i-th row of the adjacency matrix $\hat{D}$. This means that $M_{1,i,1}(j)$, i.e., the j-th element of $M_{1,i,1}$, is equal to $n_1(\chi_1, b_i, 0)$ if the sub-campaign C_1 influences the campaign C_j and zero otherwise.

For all the $j \in \{2, \ldots, N\}$, the algorithm computes the elements of the three structures Π, L, and M using the values previously computed at the $j-1$-th step, in a dynamic programming fashion (Lines 5–22). For each daily budget b_i and for each daily budget $b_k \leq b_i$, we compute the revenue and the influence index provided by the allocation of a daily budget of $b_i - b_k$ to the sub-campaign C_j and the remaining daily budget of b_k to the sub-campaigns $C_1, \ldots, C_{j-1}$. We do this by enumerating all the Pareto-efficient partial allocations $\Pi_{j-1,k,1}, \Pi_{j-1,k,2}, \ldots$ of the first $j-1$ sub-campaigns, then allocating daily budget $b_i - b_k$ to the sub-campaign C_j and, finally, we evaluate the total revenue $\bar{M}_c$ and the influence indices vector $\bar{L}_c$ provided by the partial allocations obtained, denoted with $\bar{\Pi}_c$ (Lines 9–15).[9] After that, the algorithm discards all the candidate partial allocations which are Pareto dominated (Lines 16-22); see Matthias [24] for details on Pareto efficiency and dominance.[10] Finally, the algorithm returns the optimal allocation (Lines 23–27): the optimal budgets $\hat{y}_j^*$ are the elements of $\max_i \Pi_{N,i,1}(j)$; the optimal influence indices $\hat{u}_j^*$ are computed using Equation (3); the optimal bids $\hat{y}_j^*$ are computed using the impressions models $\hat{n}_j(\cdot,\cdot,\cdot)$.

The complexity of the OPT algorithm is $O\left(\sum_j B^{\sum_i \hat{d}_{ij}+2}\right) \leq O\left(N\ B^2\ B^{\max_j \sum_i \hat{d}_{ij}}\right)$ and strictly depends on the maximum indegree of the interdependence graph corresponding to $\hat{D}$. The complexity reduces to that one of the algorithm proposed by Nuara et al. [25] when the sub-campaigns are not interdependent. Notice that capturing only the pairs of sub-campaigns with the most significant interdependence is a crucial issue from a computational point of view since, taking into account all the possible pairs of sub-campaigns, the complexity is bounded by $NB^2 \frac{B^{N+1}-1}{B-1}$, which is intractable when N is large as it happens in real-world applications.

5 THEORETICAL PROPERTIES OF THE IDIL ALGORITHM

We analyse the properties of our problem and those of the IDIL algorithm. Initially, we analyse the suboptimality of any algorithm ignoring the sub-campaigns interdependencies w.r.t. our algorithm, i.e., when the learner uses an adjacency matrix $\hat{D} = 0$, and the real one D is non-null. The following theorem shows that ignoring the sub-campaigns interdependences might be arbitrarily suboptimal.

Theorem 5.1. *Given the problem of optimizing an advertising campaign C, employing a model $\hat{n}_j(x, y)$ for the number of impressions that ignores the sub-campaigns interdependence may result in an arbitrary large loss in terms of* revenue, *defined as:*

$$R_t = \sum_{j=1}^{N} v_j\ w_j\ n_j(x_{j,t}, y_{j,t}, u_{j,t}). \tag{4}$$

Proof. Consider a campaign C with $N = 2$ sub-campaigns, adjacency matrix $D = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$, click-through rates $w_1 = w_2 = 1$, values per click $v_1 = 0$ and $v_2 = 1$, lag $K = 1$, and impression functions identified by two GPs having the following mean value:

$$n_1(x_{1,t}, y_{1,t}, u_{1,t}) = H\, y_{1,t},$$
$$n_2(x_{2,t}, y_{2,t}, u_{2,t}) = \Upsilon\, u_{2,t}\, y_{2,t} = \Upsilon\, n_1(x_{1,t-1}, y_{1,t-1}, u_{1,t-1})\, y_{2,t},$$

where $\Upsilon \in [0, 1]$, $H \in \mathbb{R}^+$. Moreover, assume to have a total budget of Y and to sample the solution space uniformly during training. Asymptotically (when we have an infinite number of samples or $\tau \to \infty$), a model that provides a stationary allocation and knows the sub-campaign interdependencies would compute the revenue:

$$\begin{aligned} R_t &= \sum_{j=1}^{N} v_j\ w_j\ n_j(x_{j,t}, y_{j,t}, u_{j,t}) = \Upsilon\, n_1(x_1, y_1, u_1)\, y_2 \\ &= \Upsilon\, H\, y_1\, y_2 = \Upsilon\, H\, (Y - y_2)\, y_2, \end{aligned}$$

where we drop the temporal indices since the proposed solution is stationary and $y_1 + y_2 = Y$. The budget allocation maximizing the revenue is $y_2 = \frac{Y}{2}$, with a revenue $R_t = \Upsilon\, H\, \frac{Y^2}{4}$.

On the other hand, a model ignoring the sub-campaigns interdependence estimates the impressions of the sub-campaign C_2 as:

$$\tilde{n}_2(x_{2,t}, y_{2,t}) = \int_0^{HY} n_2(x_{2,t}, y_{2,t}, u)du = \int_0^{HY} \Upsilon u\ y_{2,t} du = \frac{\Upsilon H^2 Y^2}{2} y_{2,t},$$

and the revenue one maximises becomes:

$$\tilde{R}_t = \sum_{j=1}^{N} v_j\ w_j\ n_j(x_{j,t}, y_{j,t}, u_{j,t}) = \tilde{n}_2(x_{2,t}, y_{2,t}) = \frac{\Upsilon H^2 Y^2}{2} y_2.$$

The budget allocation maximising $\tilde{R}_t$ is $y_2 = Y$, with a real revenue $R_t = 0$. Hence, the difference of revenue of the two algorithms is $\Upsilon H \frac{Y^2}{4} - 0$, which is arbitrarily large as H goes to ∞. □

When the model is flexible enough to model the actual process properly, we can bound its error, formally, defined as follows:

Definition 2. *Given a dataset Z, the* total (estimation) error *is:*

$$E_\tau := \sum_{j=1}^{N} \left[v_j w_j n_j(x_j^*, y_j^*, u_j^*) - \hat{v}_j \hat{w}_j \hat{n}_j(\hat{x}_j^*, \hat{y}_j^*, \hat{u}_j^*) \right],$$

where the tuples $(\hat{x}_j^*, \hat{y}_j^*, \hat{u}_j^*)$ are elements of the (stationary) output $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{y}}^*, \hat{\boldsymbol{u}}^*)$ of the IDIL algorithm using the estimates of the parameters, and (x_j^*, y_j^*, u_j^*) are elements of the (stationary) output of the IDIL algorithm using the real parameters.

We can show the following:

[8] We define χ_j as the bid that maximise the number of impressions given a budget y and a influence index u, or, formally, $\chi_j := \chi_j(y, u) = \arg\max_x \hat{n}_j(x, y, u)$.

[9] In the pseudo-code, we denoted the number of Pareto optimal allocations at the $j-1$-th row with a budget of b_i with $|\{\Pi_{j-1,i,h}\}_h|$.

[10] Notice that the inequality in Line 18 is designed for settings in which the number of impressions is monotonically increasing in the influence index. However, removing the condition in Line 18, the proposed method also applies to problems without such a monotonicity assumption. This comes at at the cost of storing a larger number of partial allocations in $\{\Pi_{j,i,h}\}_h$.

Theorem 5.2. *When the expected number of impressions $n_j(\cdot,\cdot,\cdot)$ of every sub-campaign C_j is distributed as a Gaussian Process, the total error between the real revenue and the estimated one using the output of the IDIL algorithm is upper bounded, with a probability of at least $1-\delta$, as follows:*

$$E_\tau \leq 2N v^{(\max)} \sqrt{\frac{1}{2\tau}\log\frac{6N}{\delta}}\left(\hat{n}^{(\max)} + \hat{\sigma}_\tau^{(\max)}\sqrt{2\log\frac{3N}{\delta}}\right) + N v^{(\max)}\ \hat{\sigma}_\tau^{(\max)}\sqrt{2\log\frac{3N}{2\delta}},$$

where $\hat{n}^{(\max)} := \max_j \max_{(x,y,u)} \hat{n}_j(x,y,u)$ is the maximum number of estimated expected impressions over all the sub-campaigns, $\hat{\sigma}_\tau^{(\max)} := \max_j \max_{(x,y,u)} \hat{\sigma}_{j,\tau}(x,y,u)$ is the maximum estimated standard deviation, and $v^{(\max)}$ is the maximum value per click.

We remark that Rasmussen and Williams [29] show that, in a generic GP, $\hat{\sigma}_\tau^{(\max)} \to 0$ as $\tau \to \infty$. Therefore, the total error E_τ decreases as the number of samples τ in the training set Z increases.

Proof. Since estimates for click-through rate $\hat{w}_j$ and value per click $\hat{v}_j$ are sum of i.i.d. random variables with finite support $[0,1]$ and $[0, v^{(\max)}]$, respectively, we can apply the Hoeffding's bound [14] and state that, with a probability of at least $1-\delta$:

$$w_j - \hat{w}_j \leq \sqrt{\frac{1}{2\tau}\log\frac{1}{\delta}}, \tag{5}$$

$$v_j - \hat{v}_j \leq v^{(\max)}\sqrt{\frac{1}{2\tau}\log\frac{1}{\delta}}, \tag{6}$$

where τ is the number of samples we use to compute the estimates. By assumption, the number of impressions are generated by a GP and we have that for each input (x,y,u) in the GP domain $\frac{n_j(x,y,u)-\hat{n}_j(x,y,u)}{\hat{\sigma}_{j,\tau}(x,y,u)} \sim \mathcal{N}(0,1)$, where $\hat{\sigma}_{j,\tau}(x,y,u)$ is the standard deviation computed by the GP at the point (x,y,u) by relying on τ samples in the training set Z. This implies that, with a probability of at least $1-\delta$, it holds:

$$n_j(x,y,u) \leq \hat{n}_j(x,y,u) + \hat{\sigma}_{j,\tau}(x,y,u)\sqrt{2\log\frac{1}{2\delta}}, \tag{7}$$

where the last inequality is due to the fact that, for a Gaussian random variable $X \sim \mathcal{N}(0,1)$, it holds $\forall\, x>0,\ \mathbb{P}(X>x) \leq \frac{1}{2}e^{-\frac{x^2}{2}}$.

Focus on E_τ. Sum and subtract from it the following quantities: $\hat{w}_j v_j n_j(x_j^*,y_j^*,u_j^*)$, $\hat{w}_j\hat{v}_j n_j(x_j^*,y_j^*,u_j^*)$, and $\hat{w}_j\hat{v}_j\hat{n}_j(x_j^*,y_j^*,u_j^*)$ for each $j \in \{1,\dots,N\}$. The total error can be decomposed as:

$$E_\tau = \sum_{j=1}^{N}\Bigg(\underbrace{(w_j-\hat{w}_j)v_j n_j(x_j^*,y_j^*,u_j^*)}_{E_{1j}} + \underbrace{\hat{w}_j(v_j-\hat{v}_j)n_j(x_j^*,y_j^*,u_j^*)}_{E_{2j}} + \underbrace{\hat{w}_j\hat{v}_j[n_j(x_j^*,y_j^*,u_j^*)-\hat{n}_j(x_j^*,y_j^*,u_j^*)]}_{E_{3j}}\Bigg) + \underbrace{\sum_{j=1}^{N}\hat{w}_j\hat{v}_j\hat{n}_j(x_j^*,y_j^*,u_j^*) - \sum_{j=1}^{N}\hat{w}_j\hat{v}_j\hat{n}_j(\hat{x}_j^*,\hat{y}_j^*,\hat{u}_j^*)}_{E_4}.$$

If we focus on E_{1j}, with probability at least $1-\delta$, it holds:

$$\begin{aligned}E_{1j} &= (w_j-\hat{w}_j)\,v_j\,n_j(x_j^*,y_j^*,u_j^*)\\ &\leq v^{(\max)}\sqrt{\frac{1}{2\tau}\log\frac{2}{\delta}}\left(\hat{n}_j(x_j^*,y_j^*,u_j^*) + \hat{\sigma}_{j,\tau}(x_j^*,y_j^*,u_j^*)\sqrt{2\log\frac{1}{\delta}}\right)\\ &\leq v^{(\max)}\sqrt{\frac{2}{\tau}\log\frac{2}{\delta}}\left(\hat{n}_j^{(\max)} + \hat{\sigma}_{j,\tau}^{(\max)}\sqrt{2\log\frac{1}{\delta}}\right),\end{aligned}$$

by relying on the inequalities in Equations (5) and (7), subsequently using a union bound over these two events, then defining $\hat{n}_j^{(\max)} := \max_{(x,y,u)}\hat{n}_j(x,y,u)$ and $\hat{\sigma}_{j,\tau}^{(\max)} := \max_{(x,y,u)}\hat{\sigma}_{j,\tau}(x,y,u)$, and finally since $v_j \leq v^{(\max)}$.

Similarly, we derive the following bound holding with probability at least $1-\delta$ for E_{2j}:

$$E_{2j} \leq v^{(\max)}\sqrt{\frac{1}{2\tau}\log\frac{2}{\delta}}\left(\hat{n}_j^{(\max)} + \hat{\sigma}_{j,\tau}^{(\max)}\sqrt{2\log\frac{1}{\delta}}\right),$$

by relying on the inequality in Equation (6), and the fact that $w_j \leq 1$.

Let us focus on E_{3j}. Using the inequality in Equation (7), we have that with probability at least $1-\delta$:

$$\begin{aligned}E_{3j} &= \hat{w}_j\,\hat{v}_j\,[n_j(x_j^*,y_j^*,u_j^*) - \hat{n}_j(x_j^*,y_j^*,u_j^*)]\\ &\leq v^{(\max)}\,\hat{\sigma}_{j,\tau}(x_j^*,y_j^*,u_j^*)\sqrt{2\log\frac{1}{2\delta}} \leq v^{(\max)}\,\hat{\sigma}_{j,\tau}^{(\max)}\sqrt{2\log\frac{1}{2\delta}},\end{aligned}$$

where we used $\hat{w}_j \leq 1$, and $\hat{v}_j \leq v^{(\max)}$.

Finally, let us focus on E_4. The vector $(\hat{\boldsymbol{x}}^*,\hat{\boldsymbol{y}}^*,\hat{\boldsymbol{u}}^*)$ is the optimal solution of the optimization problem stated in Equations (1a)-(1d). Therefore, by definition, we have that for each $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{u})$ satisfying the constraints in Equations (1a)-(1d) the following holds:

$$\sum_{j=1}^{N}\hat{w}_j\hat{v}_j\hat{n}_j(x_j,y_j,u_j) - \sum_{j=1}^{N}\hat{w}_j\hat{v}_j\hat{n}_j(\hat{x}_j^*,\hat{y}_j^*,\hat{u}_j^*) \leq 0,$$

which holds also for $(\boldsymbol{x}^*,\boldsymbol{y}^*,\boldsymbol{u}^*)$ and, therefore, E_4 is negative.

Recalling that $\hat{n}^{(\max)} := \max_j \hat{n}_j^{(\max)}$ and $\hat{\sigma}_\tau^{(\max)} := \max_j \hat{\sigma}_{j,\tau}^{(\max)}$, it holds, with probability at least $1-\delta$:

$$\begin{aligned}E_\tau &= \sum_{j=1}^{N}(E_{1j}+E_{2j}+E_{3j}) + E_4\\ &\leq 2Nv^{(\max)}\sqrt{\frac{1}{2\tau}\log\frac{6N}{\delta}}\left(\hat{n}^{(\max)} + \hat{\sigma}_\tau^{(\max)}\sqrt{2\log\frac{3N}{\delta}}\right) +\\ &\quad N v^{(\max)}\,\hat{\sigma}_\tau^{(\max)}\sqrt{2\log\frac{3N}{2\delta}},\end{aligned}$$

which concludes the proof. □

Our analysis has, so far, focused on the static properties of our problem. However, the scenario we are studying is a dynamical system due to the potentially delayed effects induced by the sub-campaigns interdependence. Therefore, when using a stationary allocation, it is crucial to show that the dynamics always reach a steady state in finite time and their length is upper bounded. In this context, a steady state is reached when a stationary allocation provides a constant number of impressions for each sub-campaign for at least K consecutive days. We can show the following:

Theorem 5.3. *Using the stationary allocation* $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{y}}^*, \hat{\boldsymbol{u}}^*)$ *we reach a steady state after at most* $K\ \Gamma + 1$ *days, where* K *is the maximum lag of the influence index* $u_{j,t}$ *and* Γ *is the length of the longest path of the graph* $\mathcal{G}$.

The above theorem states that the more complex the process (e.g., presenting a cascade of interdependences), the more we have to wait to completely remove the effects of a suboptimal allocation.

Proof. Consider an adjacency matrix $\hat{D}$ and assume that the lag K is the same for all the performance indices $u_{j,t}$. Moreover, consider the sub-campaigns $C_{i_1}, \ldots, C_{i_\Gamma}$ that make up the longest path on the graph $\mathcal{G}$. To achieve the steady state revenue provided by the allocation $(\hat{x}^*_{i_\Gamma}, \hat{y}^*_{i_\Gamma}, \hat{u}^*_{i_\Gamma})$, an algorithm needs that all the incoming-neighbour sub-campaigns have reached the optimal allocation for K consecutive days, so as to provide exactly $\hat{u}^*_{i_\Gamma}$ as influence index. In particular, this also happens for the sub-campaign C_{i_Γ}.

By induction, this reasoning can be applied for all nodes C_{i_h} in the longest path up to C_{i_2}. Therefore, every time we traverse a node, we require K days to conclude the transient for that node, for a total of $K\ \Gamma$ days. Instead, the node C_{i_1}, being at the beginning of the longest path, has no incoming neighbours and, therefore, the allocation prescribed by the optimal solution is $(\hat{x}^*_{i_\Gamma}, \hat{y}^*_{i_\Gamma}, 0)$. This implies that the allocation is achieved on the same day that the stationary allocation is used, leading to a total number of $K\ \Gamma + 1$ days to reach the desired allocation on the longest path $C_{i_1}, \ldots, C_{i_\Gamma}$.

The same reasoning can be replicated on any other path, but since their length is shorter or equal to the longest one, the maximum number of days required to reach the allocation $(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{y}}^*, \hat{\boldsymbol{u}}^*)$ takes no longer than $K\ \Gamma + 1$ days. □

6 EXPERIMENTAL EVALUATION

We experimentally evaluate the IDIL algorithm in a real-world setting and in a synthetic setting, generated by using a realistic simulator. We compare the revenue R_t produced by IDIL and AdComB-Mean (an off-line version of the algorithm proposed by Nuara et al. [25] neglecting any sub-campaign interdependence).

6.1 Real-world Setting

In this experiment, we rely on the data of the second campaign described in Section 2 to train our model. We recall that the length of the dataset is $\tau = 93$ days (from 20/7/2018 to 20/10/2018), the advertising campaign is composed of $N = 14$ sub-campaigns belonging to both social and search advertising channels. The corresponding estimated interdependence graph is provided in Figure 1c. From 21/10/2018 to 4/11/2018 (15 days), the campaign optimization has been performed by the IDIL algorithm.

When comparing the policies produced by IDIL with those produced by AdComB-Mean (the off-line version of AdComB-TS), the former policies appear more suitable then the latter ones, as a more significant portion of the budget is allocated to social sub-campaigns and branding search sub-campaigns. The interdependence suggested by the Granger Causality Test are confirmed by estimations provided by the GPs. Indeed, in Figure 2, we show the expected value of the prediction provided by GPs of the number of impressions for the sub-campaign C_6 with a bid value of $x = 1$ (i.e., one of the most frequent choice during the training set) and

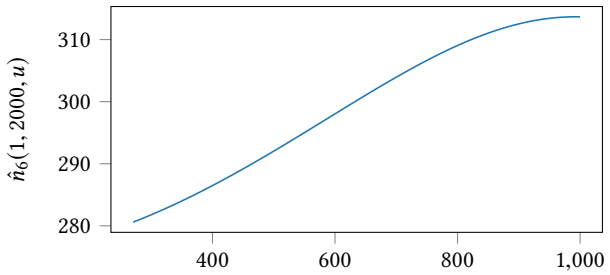


Figure 2: GPs estimation of the number of impressions $\hat{n}_6(1, 2000, u)$ depending on the influence index u.

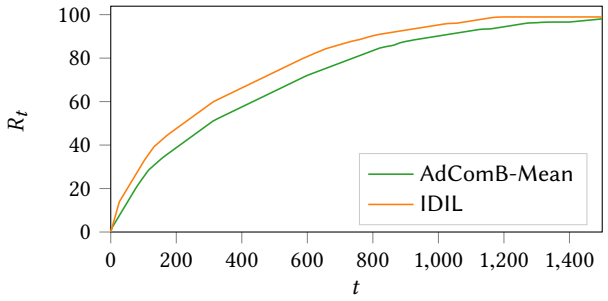


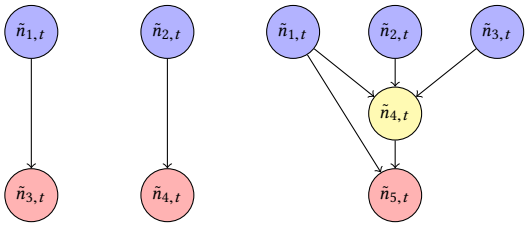
Figure 3: Comparison of the expected revenue R_t given by the AdComB-Mean and IDIL algorithms.

$y = 2000$ (i.e., a budget large enough to capture all the available user for this sub-campaign). The number of impressions increases as the value of the influence index increases, suggesting that a positive correlation between C_2 and C_3 impressions, and C_6 ones exist. However, since in a real setting we cannot exclude the presence of negative interdependence, to compute the optimal allocation with the IDIL algorithm, we remove the condition in Line 18 of Algorithm 2, to be able to provide the optimal allocation even if generic interdependence among sub-campaigns are present. In Figure 3, we show the expected revenue given by optimal policies computed by AdComB-Mean and IDIL for different values of the total budget Y. In this scenario, the exploitation of the sub-campaigns interdependence can lead to a potential revenue increase up to 13%.

In the 15 days of campaign optimization performed by the IDIL algorithm, the number of daily conversions increased by 11% w.r.t. the average of the previous 30 days (the result is compatible with our prediction, given that AdComB-TS/Mean provide very close performance). Although this is a promising result, there is no statistical significance that the IDIL algorithm outperforms in practice AdComB-TS/Mean. Due to the impossibility to directly compare the performance of the two algorithms online (e.g., by using an A/B testing system), we resort to a realistic synthetic environment.

6.2 Synthetic Settings

We evaluate the performance of the IDIL algorithm in two synthetic settings, generated by a realistic simulator, comparing the revenue R_t produced by the following algorithms: IDIL, DA-IDIL (Dependency Aware-IDIL), a variation of the IDIL algorithm *a priori* knowing the dependency matrix D, and AdComB-Mean.



(a) Setting 1 **(b) Setting** 2

Figure 4: Interdependence graph $\mathcal{G}$ for the two synthetic experimental settings.

Synthetic Data Generation. The synthetic settings are generated as follows. At day t, each sub-campaign C_j is characterized by the set of the users $S_{j,t} = s_{j,t} \cup \left(\bigcup_{i \neq j} s_{ij,t}\right)$ that could potentially visualize the ad of the sub-campaign C_j. More precisely, we distinguish the set of the users $s_{j,t}$, that would visualize the ad of C_j without having previously visualized the ads of the other interdependent sub-campaigns, from the set of the users $s_{ij,t}$, that would visualize the ad of C_j only after having visualized the ad of C_i. Notice that $s_{ij,t}$ is non-empty only if the sub-campaigns C_i and C_j are interdependent and, more precisely, if $d_{ij} \neq 0$.

The number of users $|s_{j,t}|$ is sampled from $\mathcal{N}(\mu_j, \sigma_j^2)$, i.e., a Gaussian distribution with mean μ_j and variance σ_j^2. Each user in $s_{j,t}$ is characterized by a click probability $p_j^{(cl)}$ and a conversion probability $p_j^{(co)}$ specific for the sub-campaign C_j. Conversely, the number of users $|s_{ij,t}|$ is modeled through a linear combination of the number of daily impressions $n_{i,t-1}, \ldots, n_{i,t-K}$ (whose generation is described in what follows), where K represents the maximum delay in the interdependence dynamics. Formally, we have that $s_{ij,t} := p_{ij}^{(res)} \sum_{k=1}^{K} \beta_k \, n_{i,t-k}$, where $\beta_k \in [0,1]$ are randomly sampled coefficients and $p_{ij}^{(res)}$ is the probability that a user having visualized ad of C_i is a potential user that may visualize C_j. Each user in $s_{ij,t}$ is characterized by a click probability $p_{ij}^{(cl)}$ and a conversion probability $p_{ij}^{(co)}$.

At each day t, setting the bid/budget pairs on each sub-campaign allows the advertiser to take part to $A_j \leq |S_{j,t}|$ multislot auctions based on the Vickrey-Clarke-Groves mechanism [5, 7–9, 23], in which γ_j available ad slots are allocated to a subset of δ_j advertisers ($\gamma_j \leq \delta_j$). More specifically, each advertiser submits her bid b_h and those with the first γ_j highest values $b_h \; \rho_h$ are allocated in the γ_j slots, where ρ_h is the probability that h-th ad is clicked given it has been observed. The bids b_h of the other ads participating in the auctions are drawn from a truncated Normal distribution $\mathcal{N}(\mu^{(b)}, \sigma^{(b)})$, and the click probabilities ρ_h are uniformly sampled in $[0,1]$. In the case the advertiser wins the m-th auction, the ad gets an impression ($n_{m,j,t} = 1$, otherwise $n_{m,j,t} = 0$). The ad is allocated in a the l-th slot, the ad can be visualized by an user $S_{j,t}$ according to the probability of being observed $p^{(obs)}(l)$. After the impression, the user can click on the ad and generate a conversion according to the click $p_j^{(cl)}$ and conversion $p_j^{(co)}$ probabilities if the user belongs to $s_{j,t}$, and according to the click $p_{ij}^{(cl)}$ and conversion $p_{ij}^{(co)}$ probabilities if the user belongs to $s_{ij,t}$. A click on the ad of C_j provided by the user corresponding to the m-th auction is denoted by $cl_{m,j,t} = 1$ ($cl_{m,j,t} = 0$ otherwise), and imposes a payment of $CPC_{m,j,t}$, as specified by the VCG auction (see [23] for details). The auctions are generated until the daily budget $y_{j,t}$ allocated on the sub-campaign C_j is totally spent, i.e., the total number of auctions A_j is s.t. $\sum_{m=1}^{A_j} CPC_{m,j,t} = y_{j,t}$ or until $A_j = |S_{j,t}|$. Finally, in the case a click happens, the m-th user may convert ($co_{m,j,t} = 1$) or not ($co_{m,j,t} = 0$). The daily impressions, the daily clicks, the daily conversions (assuming unitary value per conversion), and the daily costs are computed as $n_{j,t} = \sum_{m=1}^{A_j} n_{m,j,t}$, $cl_{j,t} = \sum_{m=1}^{A_j} cl_{m,j,t}$, $co_{j,t} = \sum_{m=1}^{A_j} co_{m,j,t}$, respectively.

Table 1: Parameters of the synthetic settings.

	Setting 1				Setting 2				
	C_1	C_2	C_3	C_4	C_1	C_2	C_3	C_4	C_5
μ_j	5000	5000	200	1300	10000	10000	10000	700	500
σ_j	10	10	10	10	10	10	10	5	2
γ_j^2	5	5	3	3	5	5	5	3	4
δ_j	5	6	4	5	5	5	6	4	5
$\mu^{(b)}$	0.89	1.19	1.59	1.59	0.10	0.10	0.10	1.0	1.5
$\sigma^{(b)}$	0.32	0.12	0.2	0.2	0.032	0.032	0.012	0.2	0.2
$p^{(obs)}(1)$	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
$p^{(obs)}(2)$	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
$p^{(obs)}(3)$	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7
$p^{(obs)}(4)$	0.6	0.6	-	-	0.6	0.6	0.6	-	0.65
$p^{(obs)}(5)$	0.5	0.5	-	-	0.5	0.5	0.5	-	-
$p_j^{(cl)}$	0.2	0.2	0.4	0.4	0.2	0.2	0.2	0.3	0.4
$p_j^{(co)}$	0.001	0.05	0.2	0.2	0.001	0.001	0.011	0.06	0.3

We report in Table 1 the values of the main parameters used in the two synthetic settings in which we test our algorithm.

Synthetic Setting 1. There are $N = 4$ sub-campaigns, with delayed dynamics of $K = 5$ days, whose interdependence graph is shown in Figure 4a. The longest path of the interdependence graph $\mathcal{G}$ is $\Gamma = 1$. C_1 and C_2 are on the display advertising channel and are targeted to a wide range of daily users, thus generating a large number of daily auctions, but their conversion probability is low. C_3 and C_4 are on the search advertising channel, generating a small number of daily auctions, but their conversion probability is high.

We use $Y = 500$ and $B = 10$ daily budget values evenly spaced in the range $[0, 500]$. The GPs used to estimate the impressions model of the sub-campaigns adopt a squared exponential kernel in which the kernel parameters are chosen as recommended by Rasmussen and Williams [29]. We evaluate the performance of the algorithms with different numbers of samples $\tau \in \{60, 80, 100\}$ in the training set Z. In the first τ days, a uniformly random allocation is used to collect data and, after that, the algorithms compute the optimal solution based on their estimates and then set it.

Results. In Figure 5a, we report the average (over 100 repetitions) revenue R_t produced by the algorithms with a training of $\tau = 100$ samples. From $t = 100$ on, the optimal stationary solution is used. The average revenue of each algorithm peaks at $t = 101$ and, for $t > 101$, decreases by converging to a steady state within $K\,\Gamma + 1 = 6$

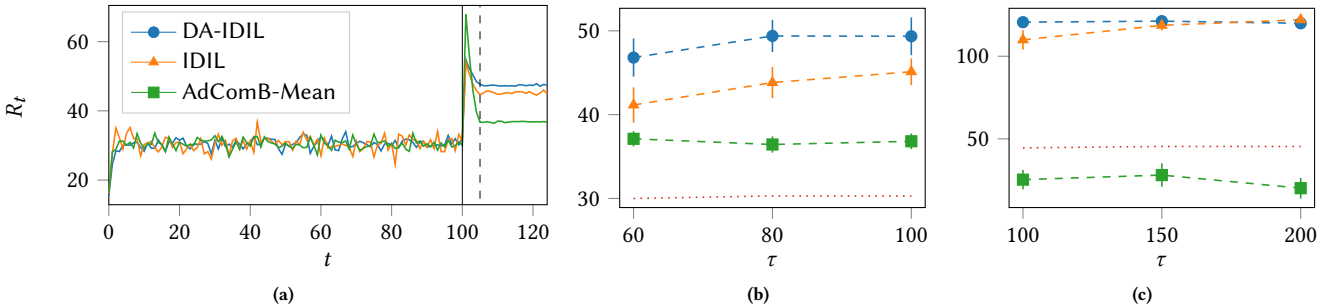


Figure 5: Results for the Settings 1 and 2. (a) Revenue R_t over time for the Setting 1. (b) Revenue R_t in steady state conditions for different training sizes τ in Setting 1. (c) Revenue R_t in steady state conditions for different training sizes τ in Setting 2. In (b) and (c), the revenue of the random allocation is reported with a dotted magenta line and the vertical lines represent the 95% confidence intervals for the algorithms revenue.

days. The peak is generated by the presence of a large number of residual users who have observed display ads during training and who, after $t = 100$, observe search ads. These residual users decrease for $t > 101$ until they reach a steady state. Thus, (temporary) peaks may be achieved with non-stationary policies.

The DA-IDIL algorithm exhibits the best performance, exploiting the *a priori* knowledge of the adjacency graph D. The gap between the revenue produced by the IDIL and DA-IDIL algorithms, due to the estimation error introduced on $\hat{D}$, is sufficiently small, showing that the Granger Causality test used by the IDIL algorithm works well in practice. Instead, the revenue produced by the AdComB-Mean algorithm, neglecting the interdependence among sub-campaigns, is much smaller than that produced by the other two algorithms. This is due to the very different budget allocations chosen by the three algorithms: the IDIL and DA-IDIL algorithms optimally balance the budget on all the sub-campaigns, while the AdComB-Mean algorithm greedily invests the budget only in the search sub-campaigns C_3 and C_4. Interestingly, the performance of the AdComB-Mean algorithm is quite similar to that of the uniformly random allocation used during training.

In Figure 5b, we report the average revenue R_t at the steady-state (averaged over the 100 independent repetitions and over $t \in \{106, \dots, 120\}$) and the 95% confidence intervals as the number of samples τ used for training increases. All algorithms always perform better than the uniformly random allocation. The performance of both the IDIL and DA-IDIL algorithms is significantly better than the one provided by AdComB-Mean (confidence intervals do not overlap). The use of more training samples provides an improvement in terms of steady-state revenue for the IDIL and DA-IDIL algorithms. On the other hand, the performance of the AdComB-Mean algorithm does not benefit from having more samples, which is probably due to the presence of a model bias induced by the fact that it neglects the sub-campaign interdependence.

Synthetic Setting 2. There are $N = 5$ sub-campaigns, whose interdependence graph is shown in Figure 4b. The longest path of the interdependence graph $\mathcal{G}$ is $\Gamma = 2$. C_1, C_2, and C_3 are display sub-campaigns directed to a wide audience and have a low cost per impression, but a low conversion rate. C_4 is a social sub-campaign, whose number of impressions is influenced by the influence index of the display sub-campaigns. Finally, C_5 is a search sub-campaign, whose impressions depend on the influence index of C_1 and C_4. The interdependence among the sub-campaigns occurs within $K = 3$ days and is modeled as in Setting 1. We set a cumulative budget of $Y = 500$ and the budget discretization from the interval $[0, 500]$ with $B = 100$. The number of samples for training is $\tau \in \{100, 150, 200\}$.

Results. In Figure 5c, we report the average (over 100 repetitions and over $t \in \{107, \dots, 120\}$) revenue of the algorithms. The performance of AdComB-Mean is worse than the one of the uniformly random allocation and gets worse as τ increases. This is an empirical confirmation of the statement of Theorem 5.1, showing that a solution that is optimal without interdependence might perform arbitrarily bad. Conversely, the performance of IDIL and DA-IDIL are significantly larger than that of the uniformly random allocation and increase as the number of samples increases.

Final Remarks. Results obtained in synthetic settings show that this model, relying on a training time which is reasonable for the application, provides a significant improvement in terms of revenue of an Internet advertising campaign. Experts in the marketing field confirmed the feasibility of what proposed in terms of learning time. Conversely, adopting more complex models would most likely result unaffordable in most of the cases, since accurate estimations would require a larger training set and, therefore, excessively long learning periods in real-world scenarios.

7 CONCLUSIONS AND FUTURE WORKS

In this paper, we formalize, for the first time, the problem of optimizing an Internet advertising campaign with sub-campaigns interdependence. We design the IDIL algorithm that, given a set of past observations, models these interdependences and returns an optimal allocation of the bid/daily budget on the sub-campaigns maximizing the revenue. We analyse the properties of the IDIL algorithm both theoretically, providing a bound on the total error, and empirically, showing that it provides revenues on synthetic datasets significantly better than the state of the art in the field.

In the future, we will extend our algorithm to an online framework and test it in a real-world application.

REFERENCES

[1] C. Alippi, M. Roveri, and F. Trovò. 2014. Learning causal dependencies to detect and diagnose faults in sensor networks. In *SSCI*. IEEE, 34–41.

[2] C. Calderón and L. Liu. 2003. The direction of causality between financial development and economic growth. *J DEV ECON* 72, 1 (2003), 321–334.

[3] D. A. Dickey and W. A. Fuller. 1979. Distribution of the estimators for autoregressive time series with a unit root. *J AM STAT ASSOC* 74, 366a (1979), 427–431.

[4] I. M. Dinner, H. J. Van Heerde, and S. A. Neslin. 2014. Driving online and offline sales: The cross-channel effects of traditional, online display, and paid search advertising. *J MARKETING RES* 51, 5 (2014), 527–545.

[5] Gabriele Farina and Nicola Gatti. 2017. Adopting the cascade model in ad auctions: Efficiency bounds and truthful algorithmic mechanisms. *J ARTIF INTELL RES* 59 (2017), 265–310.

[6] M. Gasparini, A. Nuara, F. Trovò, N. Gatti, and M. Restelli. 2018. Targeting Optimization for Internet Advertising by Learning Logged Bandit Feedback. In *IJCNN*.

[7] N. Gatti, A. Lazaric, M. Rocco, and F. Trovò. 2015. Truthful learning mechanisms for multi-slot sponsored search auctions with externalities. *ARTIF INTELL* 227 (2015), 93–139.

[8] N. Gatti, A. Lazaric, and F. Trovò. 2012. A truthful learning mechanism for contextual multi-slot sponsored search auctions with externalities. In *EC*. ACM, 605–622.

[9] Nicola Gatti, Marco Rocco, Paolo Serafino, and Carmine Ventre. 2018. Towards better models of externalities in sponsored search auctions. *THEOR COMPUT SCI* 745 (2018), 150–162.

[10] S. C. Geyik, S. Faleev, J. Shen, S. O'Donnell, and S. Kolay. 2016. Joint Optimization of Multiple Performance Metrics in Online Video Advertising. In *SIGKDD*. 471–480.

[11] S. C. Geyik, A. Saxena, and A. Dasdan. 2014. Multi-Touch Attribution Based Budget Allocation in Online Advertising. In *ADKDD*. 1–9.

[12] C. W. J. Granger. 1969. Investigating causal relations by econometric models and cross-spectral methods. *ECONOMETRICA* (1969), 424–438.

[13] P. R. Hoban and R. E. Bucklin. 2015. Effects of internet display advertising in the purchase funnel: Model-based insights from a randomized field experiment. *J MARKETING RES* 52, 3 (2015), 375–393.

[14] W. Hoeffding. 1963. Probability inequalities for sums of bounded random variables. *J AM STAT ASSOC* 58, 301 (1963), 13–30.

[15] J. A. Howard and J. N. Sheth. 1969. *The theory of buyer behavior*. Wiley, New York.

[16] IAB. 2017. IAB internet advertising revenue report. https://www.iab.it/wp-content/uploads/2018/07/IAB-2017-Full-Year-Internet-Advertising-Revenue-Report.REV2_.pdf. Online; accessed 29 August 2017.

[17] E. M. Italia, A. Nuara, F. Trovò, M. Restelli, N. Gatti, and E. Dellavalle. 2017. Internet Advertising for Non-Stationary Environments. In *AMEC*. 1–15.

[18] M. King, J. Atkins, and M. Schwarz. 2007. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *AM ECON REV* 97, 1 (2007), 242–259.

[19] P. Kireyev, K. Pauwels, and S. Gupta. 2016. Do display ads influence search? Attribution and dynamics in online advertising. *INT J RES MARK* 33, 3 (2016), 475–490.

[20] K.-C. Lee, A. Jalali, and A. Dasdan. 2013. Real Time Bid Optimization with Smooth Budget Delivery in Online Advertising. In *ADKDD*. 1–9.

[21] R. Lewis and D. Nguyen. 2015. Display advertising's competitive spillovers to consumer search. *QME-QUANT MARK ECON* 13, 2 (2015), 93–115.

[22] E. Markakis and O. Telelis. 2010. Discrete strategies in keyword auctions and their inefficiency for locally aware bidders. In *WINE*. 523–530.

[23] A. Mas-Colell, M. Whinston, and J. R. Green. 1995. *Microeconomic theory*. Vol. 1. Oxford university press New York.

[24] E. Matthias. 2005. *Multicriteria Optimization*. Springer-Verlag, Berlin, Heidelberg.

[25] A. Nuara, F. Trovò, N. Gatti, and M. Restelli. 2017. A Combinatorial-Bandit Algorithm for the Online Joint Bid/Budget Optimization of Pay-per-Click Advertising Campaigns. In *AAAI*. 2457–2463.

[26] O. Ozcicek and W. D. Mcmillin. 1999. Lag length selection in vector autoregressive models: symmetric and asymmetric lags. *APPL ECON* 31, 4 (1999), 517–524.

[27] C. Perlich, B. Dalessandro, T. Raeder, O. Stitelman, and F. Provost. 2014. Machine learning for targeted display advertising: Transfer learning in action. *MACH LEARN* 95, 1 (2014), 103–127.

[28] F. Provost, B. Dalessandro, R. Hook, X. Zhang, and A. Murray. 2009. Audience selection for on-line brand advertising: privacy-friendly social network targeting. In *SIGKDD*. 707–716.

[29] C. E. Rasmussen and C. K. I. Williams. 2006. *Gaussian Processes for Machine Learning*. The MIT Press.

[30] Manuel Roveri and Francesco Trovò. 2017. An ensemble approach for cognitive fault detection and isolation in sensor networks. *INT J NEURAL SYST* 27, 03 (2017), 1650047.

[31] O. Stitelman, B. Dalessandro, C. Perlich, and F. Provost. 2011. Estimating the effect of online display advertising on browser conversion. In *ADKDD*. 8–16.

[32] S. Thomaidou, K. Liakopoulos, and M. Vazirgiannis. 2014. Toward an integrated framework for automated development and optimization of online advertising campaigns. *INTELL DATA ANAL* 18, 6 (2014), 1199–1227.

[33] H. Y. Toda and T. Yamamoto. 1995. Statistical inference in vector autoregressions with possibly integrated processes. *J ECONOMETRICS* 66, 1-2 (1995), 225–250.

[34] J. Wang, W. Zhang, and S. Yuan. 2016. Display Advertising with Real-Time Bidding (RTB) and Behavioural Targeting. *CoRR* abs/1610.03013 (2016).

[35] W. Weinan, Y. Rong, J. Wang, T. Zhu, and X. Wang. 2016. Feedback Control of Real-Time Display Advertising. In *WSDM*. 407–416.

[36] J. Xu, K.-C. Lee, W. Li, H. Qi, and Q. Lu. 2015. Smart pacing for effective online ad campaign optimization. In *SIGKDD*. 2217–2226.

[37] J. Yan, N. Liu, G. Wang, W. Zhang, Y. Jiang, and Z. Chen. 2009. How much can behavioral targeting help online advertising?. In *WWW*. 261–270.

[38] W. Zhang, S. Yuan, and J. Wang. 2014. Optimal Real-time Bidding for Display Advertising. In *SIGKDD*. 1077–1086.

[39] W. Zhang, Y. Zhang, B. Gao, Y. Yu, X. Yuan, and T.-Y. Liu. 2012. Joint optimization of bid and budget allocation in sponsored search. In *SIGKDD*. 1177–1185.

Reconciliation k-median: Clustering with Non-polarized Representatives

Bruno Ordozgoiti*
Department of Computer Systems
Universidad Politécnica de Madrid
bruno.ordozgoiti@upm.es

Aristides Gionis
Department of Computer Science
Aalto University
aristides.gionis@aalto.fi

ABSTRACT

We propose a new variant of the k-median problem, where the objective function models not only the cost of assigning data points to cluster representatives, but also a penalty term for disagreement among the representatives. We motivate this novel problem by applications where we are interested in clustering data while avoiding selecting representatives that are too far from each other. For example, we may want to summarize a set of news sources, but avoid selecting ideologically-extreme articles in order to reduce polarization.

To solve the proposed k-median formulation we adopt the local-search algorithm of Arya et al. [2], We show that the algorithm provides a provable approximation guarantee, which becomes constant under a mild assumption on the minimum number of points for each cluster. We experimentally evaluate our problem formulation and proposed algorithm on datasets inspired by the motivating applications. In particular, we experiment with data extracted from Twitter, the US Congress voting records, and popular news sources. The results show that our objective can lead to choosing less polarized groups of representatives without significant loss in representation fidelity.

CCS CONCEPTS

• **Theory of computation** → **Facility location and clustering**; **Discrete optimization**; • **Applied computing** → *Sociology*.

KEYWORDS

Clustering; k-median; polarization; committee selection; data mining; approximation algorithms.

ACM Reference Format:
Bruno Ordozgoiti and Aristides Gionis. 2019. Reconciliation k-median: Clustering with Non-polarized Representatives. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313475

1 INTRODUCTION

Consider the problem of summarizing a set of news articles on a given topic. A standard approach to this problem is clustering: design a distance function that captures similarity between the news articles and apply a clustering algorithm on the resulting metric space. Common clustering formulations, such as k-median or k-means, can be used [17]. The original set of input news articles can then be summarized by the (small) set of cluster representatives. In some cases, however, we may be interested in selecting cluster representatives that are not too far from each other. For example, we may want to find a set of representative news articles that are not too extreme so that they can provide a basis for constructive deliberation. This motivation is similar to recent proposals in the literature that aim to reduce polarization in social media [13] and balance the users' content consumption [14]. In this work we are interested in developing computational methods for clustering data in a way that the disagreement of the cluster representatives is minimized.

Another motivating example appears in the context of electing a *k-member committee* to represent a set of individuals, such as the employees of an organization or the members of a political party. Assuming that all individuals have public opinions on a set of issues, clustering these individuals on an opinion-representation space will give a committee that faithfully represents the set of individuals with respect to the issues under consideration. Despite providing a good representation, however, a committee elected with a standard clustering approach may fail to reach consensus due to potential heterogeneity within the committee members. Heterogeneity within elected members of an assembly is a widely acknowledged problem in politics — for instance, division of representatives often results in paralysis in various left-wing political formations.[1] As in the previous example, we are interested in electing a committee in a way that the disagreement of the elected members is minimized while ensuring a faithful representation of the constituents.

Motivated by the previous examples we introduce a new formulation of the k-median problem, where in addition to the k-median objective we also seek to minimize disagreement between the cluster representatives. As it is customary, we consider a metric space (X, d), where d is a distance function for objects in X. We distinguish two subsets of X, the set of facilities F and the set of clients C. The goal is to select a set of k facilities $S \subseteq F$ — the cluster representatives — so as to minimize the overall cost

$$cost(S) = \sum_{c \in C} \min_{s \in S}\{d(c, s)\} + \frac{\lambda}{2} \sum_{s_i \in S} \sum_{s_j \in S} d(s_i, s_j). \qquad (1)$$

The first term is the same as in the standard k-median, and expresses the cost of serving each client by its closest selected facility. The second term is the one introduced in this paper and expresses

*The work described in this paper was done while this author was a visitor at Aalto University.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313475

[1] https://www.theguardian.com/commentisfree/2019/feb/19/podemos-spanish-politics

disagreement between cluster representatives. The parameter λ determines the relative importance of the two terms. Despite clustering being one of the most well-studied problems in statistical data analysis, and the numerous formulations and problem variants that have been studied in the literature, to our knowledge, the problem defined above has not been considered before.

As expected, the problem defined by optimizing Equation (1) is **NP**-hard; in fact optimizing each of the two terms separately is an **NP**-hard problem. Given the hardness of the problem, it is compelling to consider algorithms with provable approximation guarantees. For the k-median algorithm several approximation algorithms exist [11, 22]. A local-search algorithm, which is simple to implement and scalable to large datasets, has been proposed by Arya et al. [2]. The algorithm starts with an arbitrary solution and considers a swap of p selected facilities with p non-selected facilities; the swap is materialized if the cost improves, and the process continues until the cost cannot be improved. Arya et al. show that this algorithm achieves an approximation guarantee equal to $3 + 2/p$, and the running time is $O(n^p)$. In particular, for $p = 1$, the algorithm gives an approximation guarantee equal to 5, while the running time is linear.

In this paper we show how to adapt the local-search algorithm of Arya et al. [2] for the problem we consider and obtain an approximation guarantee $O(k)$ in the general case. The proposed algorithm considers 1-facility swaps, i.e., $p = 1$. Furthermore, when the clusters of the obtained solution have size $\Omega(\lambda k)$, the approximation factor becomes 11, i.e., constant. We complete the analysis of the proposed problem by deriving bounds on the objective function.

Our contributions in this paper are summarized as follows.

- We introduce the *reconciliation k-median* problem, a novel clustering problem formulation where we aim to optimize the data representation cost plus a term for agreement between the cluster representatives.
- We adapt the local-search algorithm of Arya et al. [2] and obtain provable approximation guarantees for the proposed clustering problem.
- We run experiments on datasets extracted from the Twitter social network, US Congress voting records, and popular news sources. The results show that the proposed objective can lead to the choice of less polarized groups of representatives, as measured by a well-known method for ideology estimation [4] and an objective estimate of the political leaning of news sources.

The rest of the paper is structured as follows. In Section 2 we present a brief overview to the literature that is most related to our work. In Section 3 we formally define the reconciliation k-median problem. In Section 4 we present the local-search algorithm and state its approximability properties. In Section 5 we present our experimental evaluation, while Section 6 is a short conclusion. To improve readability, the hardness proof of the reconciliation k-median problem and the proof of the approximation guarantee of the local-search algorithm are presented in the Appendix.

2 RELATED WORK

Data clustering is one of the most well-studied problems in data analysis, with applications in a wide range of areas [17]. Among the numerous formulations that have been proposed, in this paper we focus on the k-median problem setting, which has been studied extensively in the theoretical computer-science literature. Charikar et al. [11] gave the first constant-factor approximation algorithm for the k-median problem, followed by improvements that relied on both combinatorial and LP-based algorithms [10, 18, 19]. In this paper we build upon the local-search algorithm of Arya et al. [2]. This is a simple-to-implement and scalable algorithm that had been offering the best performance guarantee for over a decade. The current best approximation guarantee is $2.67 + \epsilon$, provided by the algorithm of Byrka et al. [8], which optimizes a part of the algorithm of Li and Svensson [22]. However, the algorithm is not practical.

Variants of the k-median problem have also been considered, including the Euclidean k-median [1], capacitated k-median [7], ordered k-median [9], and more. To our knowledge, however, this is the first work to study the problem of k-median clustering with a penalty on the disagreement of the cluster representatives. Instead, researchers have studied the problem of selecting k points to *maximize* the sum of pairwise distances, i.e., the *dispersion* of the selected point set. Several constant-factor approximation algorithms have been proposed for the maximum-dispersion problem [12, 16]. However, the maximization makes the problem different and it is not clear how to adapt those algorithms in our setting.

One of our motivating applications is summarization of social-media content with the aim of reducing polarization and balancing the information content delivered to users. This is a relatively new research area that is receiving increasing interest [4, 13, 14, 23, 24]. However, to the best of our knowledge, none of the proposed approaches uses a clustering formulation.

The second motivating application is election of committees and representatives. In some cases, election questions can also be formulated as voting problems. Voting in general has been studied extensively in social sciences and economics literature. From the algorithmic perspective, researchers have studied questions about voting in social networks and concepts such as liquid and viscous democracy [6, 26]. In addition to being not directly related to our paper, this line of work does not directly model agreement between elected representatives.

3 PROBLEM FORMULATION

3.1 Preliminaries

We formulate our problem in the general setting of *metric facility location* [19]. We consider a metric space (X, d) and two subsets $F, C \subseteq X$, not necessarily disjoint. The set F represents *facilities*, and the set C represents *clients*. A special case of interest is when clients and facilities are defined over the same set, i.e., $F = C$. In our discussion we consider the more general case that the sets of clients and facilities are disjoint. The function $d : X \times X \to \mathbb{R}$ is a distance measure between pairs of points in X. When $c \in C$ and $s \in F$, the distance $d(c, s)$ represents the cost of serving client c with facility s. The number of facilities is denoted by $m = |F|$ and the number of clients by $n = |C|$.

The goal is to open k facilities — i.e., choose k points in F — such that the cost of serving each client in C with the nearest *selected* facility is minimized. Given a set of facilities $S \subseteq F$, with $|S| = k$, and $s \in S$, we use $N_S(s)$ to denote the set of clients served by facility s in

the solution S, that is, $N_S(s) = \{c \in C \mid s = \text{argmin}_{x \in S}\, d(c, x)\}$. In the *facility location* formulation each facility has an associated cost, which is incurred if the facility is opened (selected). The objective is to minimize the total cost of serving all clients plus the cost of opened facilities, while there is no restriction on the number of opened facilities. When the cost of opening each facility is zero and it is required to open at most k facilities, the problem is known as *k-median*.

3.2 Reconciling cluster representatives

The problems described above, facility location and k-median, are commonly used to find cluster representatives without regard to the relative position of the representatives themselves. As discussed in the introduction, our goal is to modify the problem definition so as to find solutions in which the cluster representatives are close to each other. To achieve our goal we propose the following clustering variant, which we name reconciliation k-median.

Problem 1 (recon-k-median). *Given a metric space (X, d), two sets $F, C \subseteq X$, $k \in \mathbb{N}$, and a real number $\lambda > 0$, find a set $S \subseteq F$ with $|S| = k$, so as to* minimize *the cost function*

$$cost(S) = \sum_{s \in S} \sum_{c \in N_S(s)} d(c, s) + \frac{\lambda}{2} \sum_{s_i \in S} \sum_{s_j \in S} d(s_i, s_j). \qquad (2)$$

In order to characterize the hardness of this problem, we analyze the two terms of the objective in isolation. The first term, which results from setting $\lambda = 0$, is equivalent to the classical metric k-median problem, shown to be **NP**-hard by Papadimitriou [25]. To analyze the second term, we define the following equivalent problem, which asks to find a subset of k points that minimize the sum of pairwise distances in a metric space, i.e., minimum pairwise distances (mpd).

Problem 2 (mpd). *Given a metric space (X, d), a subset of objects in the metric space $F = \{x_1, \ldots, x_n\} \subseteq X$, and a number $k \in \mathbb{N}$, with $k < n$, define a matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ as $\mathbf{A}_{ij} = d(x_i, x_j)$ for all $x_i, x_j \in F$. The goal is to find a binary vector $\mathbf{x}$ of dimension n that has exactly k coordinates equal to 1 and minimizes the form $\mathbf{x}^T \mathbf{A}\, \mathbf{x}$. In other words, we want to find*

$$\begin{aligned} \min \quad & \mathbf{x}^T \mathbf{A}\, \mathbf{x}, \\ \textit{subject to} \quad & \mathbf{x} \in \{0, 1\}^n \textit{ and } \mathbf{x}^T \mathbf{1} = k. \end{aligned}$$

The following lemma establishes that there exists no polynomial-time algorithm to find the exact solution to the mpd problem, unless **P** = **NP**. The proof is given in the Appendix.

Lemma 1. *Problem* mpd *is* **NP**-*hard.*

Lemma 1 establishes that optimizing separately the second term of the objective function (2) is an **NP**-hard problem. Note, however, that the hardness of the two terms in Problem 1 does not immediately imply the hardness of the overall problem. Consider, for instance, an objective of the form $\min_{\mathbf{x}} \{f(\mathbf{x}) + (c - f(\mathbf{x}))\}$. Even though optimizing f can be an arbitrary **NP**-hard problem, the overall problem has a constant value, and thus, it is trivial to optimize — there is nothing to be done. We now show that Problem 1 is indeed **NP**-hard.

Lemma 2. *Problem* recon-k-median *is* **NP**-*hard.*

Proof. Consider an instance of the mpd problem, for a given set F and a number k, and form an instance (F, C, k, λ) of the recon-k-median problem, where C is any arbitrary set of clients with $F \cap C = \emptyset$. Set $d(s, c) = \frac{1}{2} \max_{s_i, s_j \in F}\{d(s_i, s_j)\}$, for all $c \in C$ and $s \in F$. Note that the distance function d is still a valid metric. We have that $\sum_{c \in C} \min_{s \in S} d(c, s)$ is constant for any potential solution set $S \subseteq F$, which implies that optimizing recon-k-median is equivalent to optimizing mpd. Thus, from Lemma 1 we obtain that recon-k-median is **NP**-hard. □

3.3 Bounds on the objective

To complete the analysis of the recon-k-median problem we offer bounds on the objective, which can be used to evaluate the quality of the solution obtained by any algorithm for the problem at a given instance. For this to be useful, the bounds need to be non-trivial and as close as possible to the optimal solution. We will first show how to obtain a lower bound on the second term of the objective, that is, $\sum_{s_i, s_j \in S} d(s_i, s_j)$. We first introduce the following definition.

Definition 1. *Given two sequences of real numbers $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ and $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_m$, with $m < n$, we say that the second sequence interlaces the first if*

$$\lambda_i \geq \mu_i \geq \lambda_{n-m+i} \textit{ for } i = 1, \ldots, m$$

We will employ the following result from Haemers [15].

Lemma 3. *Let $\mathbf{A}$ be a symmetric $n \times n$ matrix, and let $\mathbf{B}$ be a principal submatrix of $\mathbf{A}$. Then the eigenvalues of matrix $\mathbf{B}$ interlace those of matrix $\mathbf{A}$.*

We now state the following result.

Theorem 1. *Let $\mathbf{D}$ be the pairwise distance matrix of facilities of an instance of* recon-k-median *problem. Define matrix $\widetilde{\mathbf{D}}$ so that $\widetilde{\mathbf{D}}_{ij} = \sqrt{\mathbf{D}_{ij}}$, that is, a matrix whose entries are the square roots of the entries of $\mathbf{D}$. Let $\lambda_i(\widetilde{\mathbf{D}})$ denote the i-th absolutely largest eigenvalue of $\widetilde{\mathbf{D}}$. Then*

$$k\lambda_1(\mathbf{D}) \geq \sum_{s_i \in S} \sum_{s_j \in S} d(s_i, s_j) \geq \sum_{i=n-k+1}^{n} \lambda_i^2(\widetilde{\mathbf{D}}).$$

Proof. Observe that for any real symmetric matrix $\mathbf{D}$, $\|\mathbf{D}\|_F^2 = \sum_i \sigma_i^2(\mathbf{D}) = \sum_i \lambda_i^2(\mathbf{D})$. It is easily seen that if $\mathbf{x}$ is a binary vector with exactly k entries equal to 1, $\mathbf{x}^T \mathbf{D}\, \mathbf{x}$ is equal to the sum of the entries of a $k \times k$ principal submatrix of $\mathbf{D}$, which is in turn equal to the squared Frobenius norm of the corresponding submatrix of $\widetilde{\mathbf{D}}$. Combined with Theorem 3, this proves the lower bound.

The upper bound follows immediately from the variational characterization of the eigenvalues [see 20, chap. 9.2]. □

A lower and upper bound on the first term of the objective can simply be given by $\sum_{c \in C} \min_{s \in F} d(c, s)$ and $\sum_{c \in C} \max_{s \in F} d(c, s)$, respectively. This is useful only for problem instances where the number of facilities is relatively small compared to the number of clients.

Algorithm 1 Local search

```
1: procedure LocalSearch(F, C, k, λ)
2:     S ← random subset of F of cardinality k
3:     converged ← false
4:     while not converged do
5:         if there exist s ∈ S and t ∈ F \ S such that cost(S \ {s} ∪ {t}) < cost(S) then
6:             S ← S \ {s} ∪ {t}
7:         else converged ← true
8:         end if
9:     end while
10:    return S
11: end procedure
```

4 THE LOCAL-SEARCH ALGORITHM

In this section we present the proposed algorithm for the RECON-k-MEDIAN problem and state its properties. The algorithm uses the local-search strategy, proposed by Arya et al. [2], but adapted for the objective function of RECON-k-MEDIAN. The algorithm starts with an arbitrary solution consisting of k selected facilities. It then proceeds in iterations. In each iteration it considers whether it is possible to swap a selected facility with a non-selected facility and obtain an improvement in the objective score. If such an improvement is possible, the corresponding swap is performed. The algorithm terminates when no such swap is possible. At each point during its execution, the algorithm maintains a set of k clusters over the set of clients, and a selected facility for each cluster, defined by assigning each client to its closest selected facility. Pseudocode of this local-search procedure is given in Algorithm 1.

For the analysis we denote by $S = \{s_1, \ldots, s_k\} \subseteq F$ the solution returned by `LocalSearch`, and $O = \{o_1, \ldots, o_k\} \subseteq F$ the optimal solution. As mentioned before we use the notation $N_S(s)$ to denote the set of clients that are assigned to facility s in the solution S, and $N_O(o)$ to denote the set of clients that are assigned to facility o in the optimal solution O.

To analyze the performance of `LocalSearch` we follow the ideas of Arya et al. [2]. The proofs are included in the Appendix.

As a result, in the most general case, the `LocalSearch` algorithm yields a $O(\lambda k)$-factor approximation guarantee on the quality of the solution achieved.

THEOREM 2. *Let (F, C, k, λ) be an instance of the* RECON-k-MEDIAN *problem. Let S be a solution returned by the* `LocalSearch` *algorithm, and let O be an optimal solution. Then*

$$cost(S) \leq (\lambda k + 5) cost(O). \tag{3}$$

Furthermore, we are able to improve the analysis and obtain an approximation guarantee that does not depend on the number of facilities k. For the improved result we need to make the mild assumption that the number of clients in any cluster of the optimal solution and the solution returned by the algorithm is $\Omega(\lambda k)$. In particular, we have.

THEOREM 3. *Let (F, C, k, λ) be an instance of the* RECON-k-MEDIAN *problem. Let S be a solution returned by the* `LocalSearch` *algorithm, and let O be an optimal solution. Assume that $|N_S(s)| \geq \lceil 2\lambda \rceil k$ for all $s \in S$ and $|N_O(o)| \geq \lceil 2\lambda \rceil k$ for all $o \in O$. Then*

$$cost(S) \leq \max\{11, 4\lambda\}\, cost(O).$$

The running time of the `LocalSearch` algorithm is $O(nmk)$ per iteration. For most applications k is considered to be a constant. When the number of facilities m is of the same order of magnitude with the number of clients n, e.g., in the important special case $F = C$, the algorithm has quadratic complexity per iteration. However, in many applications the number of facilities is significantly smaller than the number of clients. Thus, we expect that the algorithm is very efficient in practice for those cases.

We also note that Arya et al. [2] show how to perform swaps of p facilities simultaneously and obtain an improved performance guarantee at the expense of increased running time. In our case, the penalty term in our objective, which captures the disagreement among the cluster representatives, makes the analysis significantly more complex and it is not clear how to use simultaneous swaps in order to achieve a similar quality-performance trade-off. Thus, this direction is left for future work.

5 EXPERIMENTAL EVALUATION

We perform experiments to assess the proposed concept of clustering with reconciliation of representatives, as well as the performance of the proposed `LocalSearch` algorithm.[2] Our objective is to evaluate by some objective measure whether the proposed problem formulation, as well as natural variations, can lead to a choice of representatives or sources that are more moderate, less polarizing and more likely to reach consensus.

To enrich the experimental setting and produce a more interesting set of empirical observations, we relax some of the requisites of our theoretical results. Namely, observe that in order to prove the approximation guarantees of the `LocalSearch` algorithm, we require that the distance function satisfy the properties of a metric, and that it be the same for facilities and clients. However, we believe that in practical scenarios, one might benefit from considering a wider set of options, especially if we consider the exploratory nature of clustering algorithms. Therefore, we measure dissimilarity between objects using functions that do not necessarily qualify as metrics, and we consider different ones for facilities and clients.

At a high level, our experimental methodology is as follows: We start with a dataset for which clients and facilities model a natural clustering problem and for which a distance function d is available. For the facilities of the dataset we seek to obtain a polarity score π, which is independent of the distance function d: facilities with similar polarity scores π are more likely to agree. In addition, facilities with scores closer to the middle of the spectrum are less likely to disseminate extremist ideologies. We then apply our clustering algorithm with varying values of the parameters k and λ. We are interested in answering the following questions:

Q1. How does the agreement between selected representatives or the polarization of information sources (measured by the independent polarity score π) change as a function of λ? In other words, can we get more reconciled representatives or less extreme sources by increasing the weight of

[2] Our implementation of the algorithm is available at https://github.com/brunez/recon-kmedian-ls

the disagreement penalty term (second term of the objective function (2))?

Q2. How does the k-median score change as a function of λ? In other words, can we find solutions with more reconciled representatives but without significant loss in representation fidelity (i.e., first term of the objective function (2))?

Q3. What is the impact of the parameter k on both polarity score and k-median score?

For our experimental evaluation we use the following datasets.[3]

`Twitter`: The dataset, obtained by Lahoti et al. [21], consists of a set of politically active Twitter accounts. We remove stubs — i.e., accounts that follow only one account and have no followers — resulting in 3 302 362 accounts. Out of those we consider 500 popular ones — with at least 50 000 followers — as candidate facilities, that is, representatives. As remarked in the beginning of section 5, we can extend the proposed framework by considering different metrics for the two terms of the objective function. This corresponds to a practical setting where the agreeability of the selected representatives is measured differently than their affinity to their respective consituents. Specifically, for this dataset we consider the following distance functions.

(1) **Facility-Client:** We compute distances between facilities and clients as the length of the shortest path between two Twitter accounts in the undirected follower Twitter graph.[4]

(2) **Facility-Facility:** To compute distances between facilities we use shortest-path distances, as before. We also use Euclidean distances on the spectral embedding with γ components, as described by Belkin et al. [5]. We scale the resulting distance matrix so that the average of all entries is equal to that of the shortest-path distance matrix. This way we ensure that the magnitude of λ has an equivalent effect using the different metrics.

`Congress`: We collect roll call voting records from the present US Congress using the public domain `congress` API.[5] We build a dataset where each row corresponds to a Congress representative and each column is a binary variable representing the issue being voted. Missing values are imputed using class-conditional means, where the classes we consider are the two parties: democrats and republicans. "*Present*" and "*Not voting*" votes are considered to be missing. We omit votes where all representatives are missing. We also omit representatives for whom we could not obtain an ideology estimate using the approach described below, or who missed too many votes. For this dataset, we use the Euclidean distance between the vectors corresponding to each of the representatives. To make this experiment closer to a plausible practical scenario, we restrict half of the facilities to be democrats and the other half to be republicans. In addition, clients are served by the closest facility of the same party.

`Domains`: We combined the domain-related data described in the work of Bakshy et al. [3] with the `Twitter` dataset. The set of facilities consisted of 469 domains hosting the news sources most often shared on the Facebook social network. The client set is comprised of 6 104 of the most politically active Twitter users. We consider two alternatives for computing the distances.

(1) *Mentions*: Given a facility f and a client c, let n_{cf} be the number of times a tweet by c contains a link to f. Then $d(f,c) = (n_{cf}+1)^{-1}$. To compute the pairwise distances between facilities we do the following. Consider two facilities, f and g. Let S_f (respectively S_g) be the set of clients that have tweeted a link to f (respectively g) at least once. We define $W = \sum_{c \in S_f \cup S_g} \left(\log n_{cf} \mathbb{I}\{c \in S_f\} + \log n_{cg} \mathbb{I}\{c \notin S_f\}\right)$, where $\mathbb{I}$ is the indicator function. Then

$$d(f,g) = 1 - \frac{\sum_{c \in S_f \cap S_g} \log n_{cf}}{W}. \tag{4}$$

We define $\log 0 = 0$. Since the objective function of RECON-k-MEDIAN (Equation (1)) sums over all ordered pairs of facilities, this distance function is in effect symmetric when applied to our problem. Note that this is akin to the Jaccard index for set similarity, but each element is weighted with a measure of its relevance.

(2) *Latent*: We construct a matrix A such that A_{ij} is the number of times a tweet by user j contains a link to domain i. We compute the singular value decomposition $A = U\Sigma V^T$ and extract the latent representation for both domains and users in the first 9 components (which account for 50% of the total Frobenius norm of A). If $k = 9$, domain i is represented as $U_{i,:k}$ and user j as $V_{i,:k}$. To compute both facility-facility and facility-client distances we take the Euclidean distances between the corresponding latent representations.

The characteristics of the datasets are summarized in Table 1.

For all datasets, in order to compute the objective of RECON-k-MEDIAN we take averages instead of the sums of distances. Note that this amounts to scaling both sums, so it is equivalent to setting λ to a particular value. The advantage of taking averages is that λ has an impact at small values, i.e., at "small" factors of 1.

Ground truth polarity scores (π). To measure the polarity scores of the facilities we employ different methods depending on the dataset.

In the case of `Twitter` and `Congress` we use the approach described by Barberá [4], which estimates the ideological leaning of a Twitter account as a real value. For `Twitter`, we use polarity scores collected at the same time as the follower graph (July 2017). For `Congress`, we collected the ideological estimates in May 2018. Using this method and the proposed datasets, all the elicited polarity scores are between -3 and 3. We measure the polarity of the chosen representatives as follows. Given a solution $S = \{s_1, \ldots, s_k\}$, let $\pi(s_i)$ denote the estimated polarity of facility s_i. We define the polarity of solution S as the sample standard deviation of the set $\{\pi(s_1), \ldots, \pi(s_k)\}$.

For `Domains`, we use the ideological leaning score associated to each domain as described in the work of Lahoti et al. [21]. These scores were computed roughly as the fraction of interactions (visits or shares) by conservative users, out of total interactions. We translated the scores so that they fall between -0.5 (left) and 0.5 (right). In this scenario, we are interested in choosing less polarized sources. We therefore measure the polarity of the chosen set as the ℓ-2 norm of the vector $(\pi(s_1), \ldots, \pi(s_k))$.

[3] The datasets are available at https://doi.org/10.5281/zenodo.2573954

[4] The follower graph corresponds to a snapshot taken in July 2017.

[5] https://github.com/unitedstates/congress/wiki

Table 1: Summary of the datasets.

Name	Number of clients	Number of facilities	Distance functions
`Twitter`	3 302 362	500	Shortest path, Spectral embedding + Euclidean
`Congress`	420	420	Euclidean
`Domains`	6 104	469	Weighted Jaccard + Mentions, Latent space + Euclidean

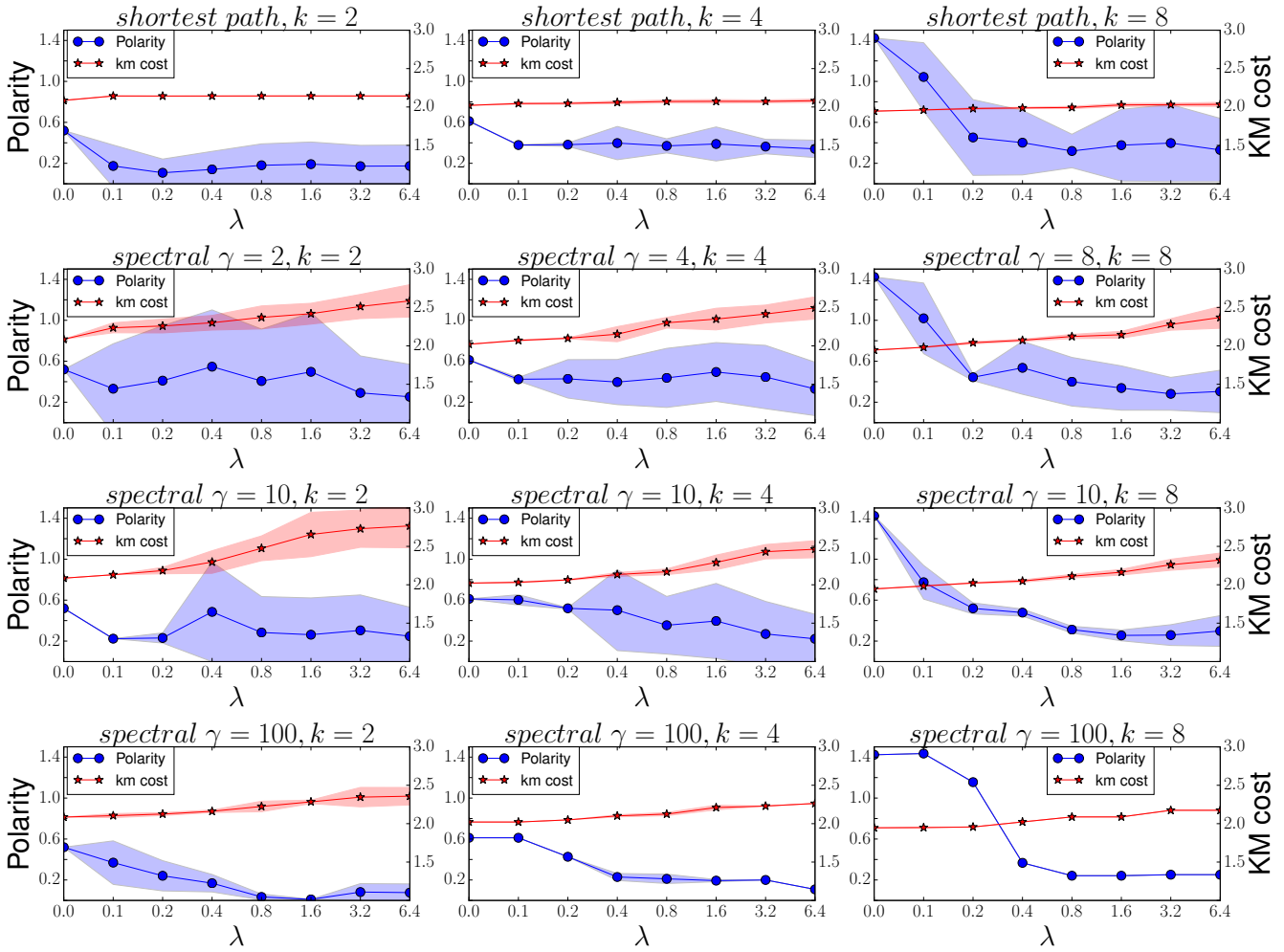


Figure 1: Results on the `Twitter` dataset with different metrics and values of k.

Results on `Twitter` dataset. We run the algorithm on the `Twitter` dataset setting the number of selected facilities to $k = 2, 4, 8$ and $\lambda = 0, 2^i/10$ for $i = 1, \ldots, 6$. For computing the pairwise distances between facilities we use either the shortest path metric or the spectral embedding with $\gamma = k, 10, 100$. Figure 1 illustrates the results. We depict, as a function of λ, the polarity of the chosen representatives, measured as described above, along with the k-median cost of the solution — i.e., the first term of the objective function (2). We run the algorithm 40 times for each setting and report the average and standard deviation bands. We can see that increasing the value of λ leads to significantly less polarized representatives in various cases. The effect is most noticeable for larger values of k, and particularly consistent using the spectral embedding with 100 components. An interesting result on this dataset is that we can achieve significant decreases in polarity without incurring much additional k-median cost. That is, it is possible to elect a much more agreeable committee — with respect to the chosen polarity measure — without notable loss in representation fidelity.

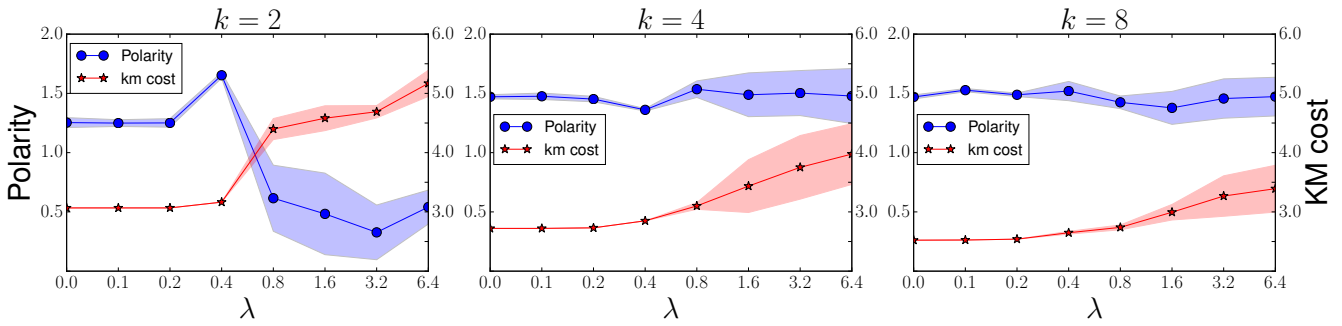


Figure 2: Results on the `Congress` dataset for different values of k.

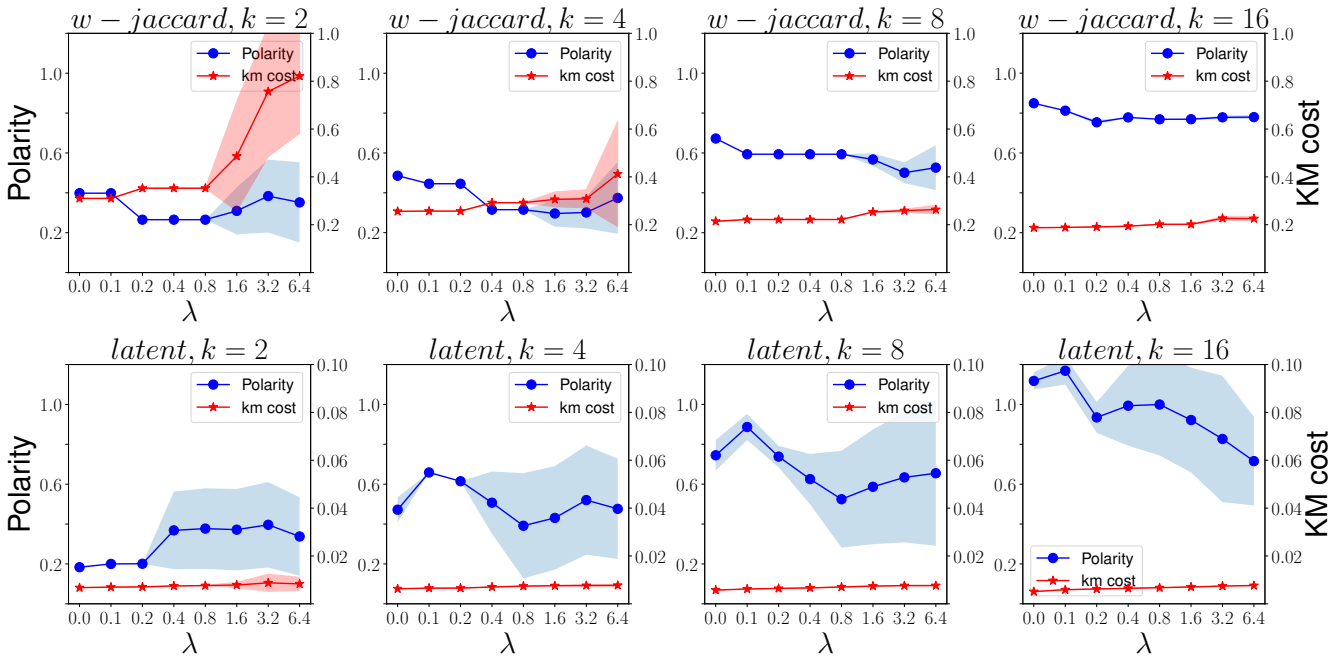


Figure 3: Results on the `Domains` dataset with different metrics and values of k.

Results on `Congress` dataset. We run the algorithm on the `Congress` dataset, using the same configurations as for `Twitter`. Figure 2 illustrates the results. Here, the decrease in polarity is only clear in the case $k = 2$, for values of λ at least 0.8. It should be noted, however, that the voting data and the polarity scores come from completely different sources. It would therefore be interesting to carry out further experiments with these data.

Results on `Domains` dataset. We run the algorithm on the `Domains` dataset, using the same configurations as for `Twitter` but considering $k = 16$ as well, as in the case of news sources it is practical to consider larger sets. Figure 3 shows the results, using the distance function defined in Equation (4) (*w-jaccard*) and the latent representation (*latent*). We run each configuration 80 times and report average results and standard deviation bands. The reduction in polarity is noticeable, in particular using the latent representation with larger values of k. For very small sets of news sources (e.g., $k = 2$) the method does not exhibit a reduction in polarity. In order to gain further insight on the impact of the penalty term, we report an example of the news sources that appear more frequently as λ increases. Specifically, we take a case where decrease in polarity is noticeable (*latent*, $k = 8$, $\lambda = 0.8$). We then collect the 16 sources that appear the most, and do the same for $\lambda = 0$. The results are shown in Table 2. For each domain, we report the frequency (i.e., the fraction of times it was part of the solution out of the 80 runs), the ideological score and the number of times it was mentioned in the collected tweets.

Number of iterations. Even though the time complexity of the local-search algorithm per iteration is not too high, a legitimate concern to be raised is the possibility that it might require a large

Table 2: Top-16 news sources for different values of λ, using latent representations.

$\lambda = 0$				$\lambda = 0.8$			
Domain	Frequency	Ideology	Mentions	Domain	Frequency	Ideology	Mentions
nydailynews.com	1.000	-0.114	10 191	chicagotribune.com	0.962	-0.082	1 531
politico.com	1.000	-0.073	36 184	chron.com	0.475	0.170	431
slate.com	1.000	-0.341	14 364	abc13.com	0.325	0.005	255
cbsnews.com	0.687	-0.057	4 394	9news.com	0.250	-0.016	349
buzzfeed.com	0.687	-0.262	11 683	detroitnews.com	0.225	0.090	535
twitchy.com	0.537	0.469	13 192	azc.cc	0.225	-0.028	744
westernjournalism.com	0.462	0.450	3 562	nbcwashington.com	0.225	-0.214	485
9news.com	0.462	-0.016	349	csmonitor.com	0.225	-0.030	382
politifact.com	0.350	-0.240	3 097	wjla.com	0.225	-0.160	374
cbsloc.al	0.337	-0.081	2 526	msn.com	0.200	-0.031	615
christianpost.com	0.337	0.337	383	kgw.com	0.187	-0.118	107
theatlantic.com	0.312	-0.176	6 883	christianpost.com	0.175	0.336	383
newrepublic.com	0.312	-0.335	1 626	abc7chicago.com	0.175	-0.251	328
lifenews.com	0.200	0.483	3 657	inquisitr.com	0.175	0.049	2 150
6abc.com	0.200	-0.252	819	stripes.com	0.175	0.182	555
usatoday.com	0.112	-0.064	18 513	wsbtv.com	0.137	-0.043	167

Table 3: Number of iterations on `Twitter` dataset (avg/max).

Metric	$k = 2$	$k = 4$	$k = 8$
Shortest path	2.5/4	2.8/5	3.09/7
Spectral, $\gamma = 2$	2.18/3	-	-
Spectral, $\gamma = 4$	-	2.73/5	-
Spectral, $\gamma = 8$	-	-	3.13/6
Spectral, $\gamma = 10$	2.32/4	2.8/6	3.32/7
Spectral, $\gamma = 100$	2.45/4	2.75/5	2.58/4

number of iterations to converge. Our observations, however, suggest that in practice a small number of iterations — where by iteration we understand the inspection of all candidate changes — are necessary. Table 3 shows the average and maximum number of iterations for the `Twitter` dataset.

6 CONCLUSIONS

We have considered the problem clustering data so as to optimize the total representation cost plus an additive term to penalize disagreement among the chosen representatives. The proposed problem, which we name reconciliation k-median, has applications in summarizing data with non-polarized representatives, as well as in electing k-member committees that are more likely to reach consensus. We have shown the proposed problem to be **NP**-hard and derived bounds on the objective. Inspired by the literature on related problems, we have analyzed a local-search algorithm in this context and derived approximation guarantees, of factor $O(\lambda k)$ in the general setting, and constant under mild assumptions. Through experiments on real data coming from a social network and voting records, we have shown empirically how the proposed formulation can lead to the choice of less polarized groups of representatives, as measured by a well-known method for ideology estimation, as well as less ideologically-extreme sets of news sources. This work opens various enticing directions for future inquiry. First it would be interesting to determine whether the approximation guarantees can be improved, as well as to attempt to find tight examples to know the possible extent of said improvement. Second, it would be interesting to perform further experiments on similar and other datasets. It is particularly compelling to improve our understanding of how different metrics can interact with known methods for estimating polarization.

ACKNOWLEDGMENTS

This work was supported by three Academy of Finland projects (286211, 313927, and 317085), and the EC H2020 RIA project "SoBigData" (654024).

APPENDIX

6.1 Hardness results

Here we prove Lemma 1, which is a key ingredient of the proof of hardness for RECON-k-MEDIAN. Before we proceed, we provide a definition of Densest k-subgraph, a well-known **NP**-hard optimization problem which we employ in our reduction.

PROBLEM 3. *(densest k-subgraph — DkS) Given a simple graph $G = (V, E)$ with $|V| = n$ and adjacency matrix A, and a number $k \in \mathbb{N}$ with $k < n$, find*

$$\min_{\mathbf{x}} \quad \mathbf{x}^T \mathbf{A}\, \mathbf{x}, \qquad \textit{subject to} \quad \mathbf{x} \in \{0,1\}^n \textit{ and } \mathbf{x}^T \mathbf{1} = k.$$

We can now prove the aforementioned lemma.

PROOF OF LEMMA 1. We proceed by reduction from Densest k-subgraph (DkS).

Consider an instance of DkS, $G = (V, E)$, $|V| = n$ with adjacency matrix A. We define a matrix $\tilde{A}$ as follows:

$$\tilde{A}_{ij} = \begin{cases} A_{ij} & (i,j) \in E \\ 0 & i = j \\ 2 & \text{otherwise.} \end{cases}$$

Notice that this matrix is symmetric, the diagonal (and nothing else) is zero, and since $\min_{i \neq j} \tilde{A}_{ij} = 1$ and $\max_{i \neq j} \tilde{A}_{ij} = 2$, for all i, j, and ℓ

$$\tilde{A}_{ij} \leq \tilde{A}_{i,\ell} + \tilde{A}_{\ell,j}.$$

We want to show that if

$$\mathbf{x} = \operatorname*{argmin}_{\substack{\mathbf{x} \in \{0,1\}^n \\ \mathbf{x}^T \mathbf{1} = k}} \mathbf{x}^T \tilde{A} \mathbf{x},$$

then

$$\mathbf{x} = \operatorname*{argmax}_{\substack{\mathbf{x} \in \{0,1\}^n \\ \mathbf{x}^T \mathbf{1} = k}} \mathbf{x}^T A \mathbf{x}.$$

We can write $\mathbf{x}^T \tilde{A} \mathbf{x} = \sum_i \sum_j \tilde{A}_{ij} \mathbb{I}\{x_i = x_j = 1\} = M + 2N$, where we have defined

$$M = |\{(i,j) | x_i = x_j = 1\} \cap E|$$
$$N = |\{(i,j) | x_i = x_j = 1\} \cap \bar{E}|$$

That is, M is the number of pairs in $\mathbf{x}$ with a corresponding edge in G, and N is the number of pairs in $\mathbf{x}$ *without* a corresponding edge in G. Note that $M + N = 2\binom{k}{2} = k^2 - k$.

Similarly, $\mathbf{x}^T A \mathbf{x} = \sum_i \sum_j A_{ij} \mathbb{I}\{x_i = x_j = 1\} = M$, where $M = k^2 - k - N$.

Suppose $\mathbf{x}$ minimizes MPD but it does not maximize DkS. Then for some $\mathbf{x}'$,

$$\mathbf{x}'^T A \mathbf{x}' > \mathbf{x}^T A \mathbf{x},$$

which implies

$$M' > M,$$

or

$$\mathbf{x}'^T \tilde{A} \mathbf{x}' = M' + 2N' < M + 2N = \mathbf{x}^T \tilde{A} \mathbf{x}.$$

This contradicts the initial assumptions, hence if $\mathbf{x}$ minimizes MPD, $\mathbf{x}$ maximizes DkS. □

6.2 Approximation guarantees

We employ the following notation:
– $f(S) = \sum_{x \in C} d(x, s(x))$, where $s(x)$ is the facility assigned to client x;
– $g(S) = \frac{1}{2} \sum_{x \in S} \sum_{y \in S} d(x,y)$;
– $N_S(s)$ is the set of clients served by facility s in solution S.

Proof of theorem 3. From Arya et al. [2], we know there exists a set of k pairs $\Sigma = \{(o_i, s_j) \mid i,j \in [k]\} \subset O \times S$ satisfying the following properties:

(1) Every $o \in O$ is considered in exactly one pair.
(2) Every $s \in S$ is considered in two pairs at most.

From the local optimality of S we have

$$f(S - s_j + o_i) - f(S) + g(S - s_j + o_i) - g(S) \geq 0,$$

for any $i,j \in [k]$. Again from Arya et al. [2], we know we can choose the set of pairs Σ such that the following inequality holds:

$$5f(O) - f(S) \geq \sum_{(o_i,s_j) \in \Sigma} \Big(f(S - s_j + o_i) - f(S) \Big). \tag{5}$$

Furthermore, since S and $S - s_j + o_i$ differ in one centroid only, we have

$$\begin{aligned} g(S - s_i + o_j) - g(S) &= \sum_{x \in S - s_j + o_i} \sum_{y \in S - s_j + o_i} d(x,y) - \sum_{x \in S} \sum_{y \in S} d(x,y) \\ &= \sum_{\substack{s \in S \\ s \neq s_j}} d(o_i, s) - \sum_{s \in S} d(s, s_j). \end{aligned}$$

Summing this difference over all k pairs we get

$$\begin{aligned} &\sum_{(o_i,s_j) \in \Sigma} (g(S - o_i + s_i) - g(S)) \\ &\qquad = \sum_{(o_i,s_j) \in \Sigma} \left(\sum_{s_p \in S, s_p \neq s_j} d(o_i, s_p) - \sum_{s_p \in S} d(s_p, s_j) \right). \end{aligned}$$

Therefore, we have

$$5f(O) + \sum_{(o_i,s_j) \in \Sigma} \sum_{s \in S, s \neq s_j} d(o_i, s) \geq f(S) + \sum_{(o_i,s_j) \in \Sigma} \sum_{s \in S} d(s_j, s).$$

We consider this inequality, and modify it so that it becomes dependent only on factors of $f(O)$, $g(O)$, $f(S)$, $g(S)$. To accomplish this, we will consider the set of pairs Σ.

First, let us examine the following quantity:

$$\sum_{(o_i,s_j) \in \Sigma} \sum_{s \in S, s \neq s_j} d(o_i, s).$$

By the triangle inequality, we have that the sum corresponding to each pair $(o_i, s_j) \in \Sigma$ is bounded as follows:

$$\sum_{\substack{s \in S \\ s \neq s_j}} d(o_i, s) \leq \sum_{\substack{s \in S \\ s \neq s_j}} d(o_i, o_{h_s}) + d(o_{h_s}, x) + d(x, s), \tag{6}$$

where o_{h_s} is such that $N_O(o_{h_s}) \cap N_S(s) \neq \emptyset$ and $x \in N_O(o_{h_s}) \cap N_S(s)$. If $N_S(s) \subseteq N_O(o_i)$, we consider $d(o_i, s) \leq d(o_i, o_{h_s}) + d(o_i, x) + d(x, s)$, choosing o_{h_s} as described below.

For notational convenience, we define

$$\sigma(o_i) = \sum_{\substack{s \in S \\ s \neq s_j}} d(o_i, o_{h_s}) + d(o_{h_s}, x) + d(x, s),$$

for each $(o_i, s_j) \in \Sigma$ — note that each o_i appears exactly once in the pairs in the set Σ, so the pair (o_i, s_j) is uniquely determined by o_i.

We want to choose the entries of inequality (6) such that

– every entry of the form $d(o_i, o_{h_s})$ appears twice;
– every entry of the form $d(o_{h_s}, x)$ appears once;
– every entry of the form $d(x, s)$ appears once.

Algorithm 2 Reverse procedure to choose the replacements

```
for i = 1, ..., k do
    B_i ← ∅
end for
for i = 1, ..., k do
    r ← random ordering of 1, ..., k
    for every j = 1, ..., k
    B_i ← B_i ∪ {r(j)}
end for
```

To achieve this, we need a set of replacements such that for each $\sigma(o_i)$, the $k-1$ corresponding replacements contain $k-1$ distinct entries of the forms $d(o_i, o_{h_s})$, $d(o_{h_s}, x)$, $d(x, s)$.

We define a function μ that maps each $x \in C$ to a pair (o_i, s_j) such that $x \in N_S(s_j) \cap N_O(o_i)$. Since $|N_S(s_i)| \geq k$ and $|N_O(o_i)| \geq k$ by assumption, we can choose a subset $\tilde{C} \subseteq C$ of k^2 points such that each $s \in S$, as well as each $o \in O$, appears in k of the k^2 pairs associated to the chosen points. We now define k multisets $B_j = \{i : \forall x \in \tilde{C}$ such that $\mu(x) = (o_i, s_j)\}$. That is, each $s_j \in S$ corresponds to a multiset B_j, which contains the index of each $o_i \in O$ with which s_j shares a client in $\tilde{C}$, including repetitions.

For $i = 1, \ldots, k$ we define the multisets $K_i = \{i, i, \ldots, i\}$, $|K_i| = k$ and $M = \bigcup_i K_i$. Consider the procedure outlined by Algorithm 2. It is clear that such a procedure can produce any partition of the multiset M into k sets containing k elements each. Consider the orderings such that said procedure results in the partition $B_1, \ldots, B_k$. If we reverse the procedure using those orderings, we obtain the desired sequence of replacements. Note that for any entry $d(o_i, s)$ in $\sigma(o_i)$ such that $N_S(s) \subseteq N_O(o_i)$, we can make the rest of the replacements first and then choose o_{h_s} from the remaining elements of O. Additionally, note that each sum $\sigma(o_i)$ requires us to make only $k-1$ replacements. Since the above method provides k satisfactory replacements for each sum, we can simply pick $k-1$ of those. We can now conclude that

- every element of the form $d(o_i, o_j)$ can only appear either in $\sigma(o_i)$ or in $\sigma(o_j)$. Since for all $\sigma(o_i)$, the entry $d(o_i, o_j)$ is unique, each $d(o_i, o_j)$ appears at most twice;
- for any element of the form $d(o, x)$, $o \in O$, x is unique;
- for any element of the form $d(s, x)$, $s \in S$, x is unique.

We have established the following inequality

$$\sum_{(o_i, s_j) \in \Sigma} \sum_{\substack{s \in S \\ s \neq s_j}} d(o_i, s) \leq 2g(O) + f(O) + f(S). \tag{7}$$

We now examine the quantity

$$\sum_{(o_i, s_j) \in \Sigma} \sum_{s \in S} d(s_j, s). \tag{8}$$

For every two facilities $s_i, s_j \in S$, either

- $d(s_i, s_j)$ appears in the sum;
- $d(s_i, s_j)$ does not appear in the sum. This implies 1) s_i, s_j are not part of any pair in Σ and 2) some other s_h appears in two pairs (remember that there are exactly k pairs and each $s \in S$ takes part in at most two). Since the summation $\sum_{s_j \in S} d(s_h, s_j)$ is computed over all elements of S, we have two appearances of $d(s_h, s_i)$ and other two of $d(s_h, s_j)$ (one for each summation corresponding to the two pairs involving s_h). Since $d(s_h, s_j) + d(s_h, s_i) \geq d(s_i, s_j)$, we can replace $d(s_h, s_j) + d(s_h, s_i)$ with $d(s_i, s_j)$ in the sum (8) and the inequality still holds.

This implies the following:

$$\sum_{(o_i, s_j) \in \Sigma} \sum_{s \in S} d(s_j, s) \geq g(S).$$

Combining this with inequality (5) we can state

$$10f(O) + 2 \sum_{(o_i, s_j) \in \Sigma} g(S - s_j + o_i) - 2f(S) - 2g(S) \geq 0. \tag{9}$$

Considering each cluster in both S and O to have at least $\lceil 2\lambda \rceil k$ points, we can extend the result to account for λ. To do this, we apply the replacements leading to inequality (7) $\lceil 2\lambda \rceil$ times to the sum $2\lambda \sum_{(o_i, s_j) \in \Sigma} \sum_{\substack{s \in S \\ s \neq s_j}} d(o_i, s)$ and obtain

$$2\lambda \sum_{(o_i, s_j) \in \Sigma} \sum_{\substack{s \in S \\ s \neq s_j}} d(o_i, s) \leq 4\lambda g(O) + f(O) + f(S)$$

To see this, note that for every entry of the form $d(o_i, x)$ or $d(s_j, x)$, x is still unique. Combining this with inequality (9), we have

$$10f(O) + 4\lambda g(O) + f(O) + f(S) \geq 2f(S) + 2g(S),$$

which implies

$$11f(O) + 4\lambda g(O) \geq f(S) + g(S).$$

Therefore,

$$\max\{11, 4\lambda\}\, cost(O) \geq cost(S).$$

□

Theorem 4. *Let (F, C, k, λ) be an instance of problem 1. Let $S = \{s_1, \ldots s_k\}$ be a solution output by algorithm 1 and let $O = \{o_1, \ldots o_k\}$ be an optimal solution.. Then*

$$cost(S) \leq (\lambda k + 5)\, cost(O) \tag{10}$$

The proof, which we omit due to space constraints, is similar to that of theorem 3.

REFERENCES

[1] Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. 2017. Better guarantees for k-means and Euclidean k-median by primal-dual algorithms. In *Proceedings of the 58th Annual Symposium on Foundations of Computer Science (FOCS)*. 61–72.

[2] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. 2004. Local search heuristics for k-median and facility location problems. *SIAM Journal on computing* 33, 3 (2004), 544–562.

[3] Eytan Bakshy, Solomon Messing, and Lada A Adamic. 2015. Exposure to ideologically diverse news and opinion on Facebook. *Science* 348, 6239 (2015), 1130–1132.

[4] Pablo Barberá. 2014. Birds of the same feather tweet together: Bayesian ideal point estimation using Twitter data. *Political Analysis* 23, 1 (2014), 76–91.

[5] Mikhail Belkin and Partha Niyogi. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*. 585–591.

[6] Paolo Boldi, Francesco Bonchi, Carlos Castillo, and Sebastiano Vigna. 2009. Voting in social networks. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM)*. 777–786.

[7] Jarosław Byrka, Krzysztof Fleszar, Bartosz Rybicki, and Joachim Spoerhase. 2014. Bi-factor approximation algorithms for hard capacitated k-median problems. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 722–736.

[8] Jarosław Byrka, Thomas Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. 2014. An improved approximation for k-median, and positive correlation in budgeted optimization. In *Proceedings of the 26th annual ACM-SIAM symposium on Discrete algorithms (SODA)*. 737–756.

[9] Jarosław Byrka, Krzysztof Sornat, and Joachim Spoerhase. 2018. Constant-factor approximation for ordered k-median. In *Proceedings of the 50th annual ACM symposium on Theory of computing (STOC)*.

[10] Moses Charikar and Sudipto Guha. 1999. Improved combinatorial algorithms for the facility location and k-median problems. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS)*. 378–388.

[11] Moses Charikar, Sudipto Guha, Éva Tardos, and David Shmoys. 1999. A constant-factor approximation algorithm for the k-median problem. In *Proceedings of the 31st annual ACM symposium on Theory of computing (STOC)*. 1–10.

[12] Sándor P Fekete and Henk Meijer. 2004. Maximum dispersion and geometric maximum weight cliques. *Algorithmica* 38, 3 (2004), 501–511.

[13] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. 2017. Reducing controversy by connecting opposing views. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining (WSDM)*. 81–90.

[14] Kiran Garimella, Aristides Gionis, Nikos Parotsidis, and Nikolaj Tatti. 2017. Balancing information exposure in social networks. In *Advances in Neural Information Processing Systems*. 4666–4674.

[15] Willem H Haemers. 1995. Interlacing eigenvalues and graphs. *Linear Algebra and its applications* 226 (1995), 593–616.

[16] Refael Hassin, Shlomi Rubinstein, and Arie Tamir. 1997. Approximation algorithms for maximum dispersion. *Operations research letters* 21, 3 (1997), 133–137.

[17] Anil Jain and Richard Dubes. 1988. *Algorithms for clustering data*. Prentice-Hall, Inc.

[18] Kamal Jain, Mohammad Mahdian, and Amin Saberi. 2002. A new greedy approach for facility location problems. In *Proceedings of the 34th annual ACM symposium on Theory of Computing (STOC)*. 731–740.

[19] Kamal Jain and Vijay Vazirani. 2001. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation. *Journal of the ACM (JACM)* 48, 2 (2001), 274–296.

[20] Erwin Kreyszig. 1978. *Introductory functional analysis with applications*. Vol. 1. wiley New York.

[21] Preethi Lahoti, Kiran Garimella, and Aristides Gionis. 2018. Joint Non-negative Matrix Factorization for Learning Ideological Leaning on Twitter. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining (WSDM)*. 351–359.

[22] Shi Li and Ola Svensson. 2016. Approximating k-median via pseudo-approximation. *SIAM J. Comput.* 45, 2 (2016), 530–547.

[23] Sean Munson, Stephanie Lee, and Paul Resnick. 2013. Encouraging Reading of Diverse Political Viewpoints with a Browser Widget.. In *ICWSM*.

[24] Cameron Musco, Christopher Musco, and Charalampos E Tsourakakis. 2017. Minimizing Polarization and Disagreement in Social Networks. *arXiv:1712.09948* (2017).

[25] Christos Papadimitriou. 1981. Worst-case and probabilistic analysis of a geometric location problem. *SIAM J. Comput.* 10, 3 (1981), 542–557.

[26] Hiroshi Yamakawa, Michiko Yoshida, and Motohiro Tsuchiya. 2007. Toward delegated democracy: Vote by yourself, or trust your network. *International Journal of Human and Social Sciences* 1, 2 (2007), 146–150.

ActiveLink: Deep Active Learning for Link Prediction in Knowledge Graphs

Natalia Ostapuk
University of Fribourg
Fribourg, Switzerland
natalia.ostapuk@unifr.ch

Jie Yang*
University of Fribourg
Fribourg, Switzerland
jie.yang@unifr.ch

Philippe Cudré-Mauroux
University of Fribourg
Fribourg, Switzerland
pcm@unifr.ch

ABSTRACT

Neural networks have recently been shown to be highly effective at predicting links for constructing knowledge graphs. Existing research has mainly focused on designing 1) deep neural network models that are expressive in capturing fine-grained semantics, e.g., NTN and ConvE, but that are however less scalable; or 2) shallow models that are scalable, e.g., TransE and DistMult, yet limited in capturing expressive semantic features. In this work, we demonstrate that we can get the best of both worlds while drastically reducing the amount of data needed to train a deep network by leveraging active learning.

We present a novel deep active learning framework, ActiveLink, which can be applied to actively train any neural link predictor. Inspired by recent advances in Bayesian deep learning, ActiveLink takes a Bayesian view on neural link predictors, thereby enabling uncertainty sampling for deep active learning. ActiveLink extends uncertainty sampling by exploiting the underlying structure of the knowledge graph, i.e., links between entities, to improve sampling effectiveness. To accelerate model training, ActiveLink further adopts an incremental training method that allows deep neural networks to be incrementally trained while optimizing their generalizability at each iteration. Extensive validation on real-world datasets shows that ActiveLink is able to match state-of-the-art approaches while requiring only 20% of the original training data.

CCS CONCEPTS

• **Theory of computation → Active learning**; • **Computing methodologies → Reasoning about belief and knowledge**; **Semantic networks**.

KEYWORDS

Deep active learning; neural link prediction; incremental training

ACM Reference Format:
Natalia Ostapuk, Jie Yang, and Philippe Cudré-Mauroux. 2019. ActiveLink: Deep Active Learning for Link Prediction in Knowledge Graphs. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313620

*Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313620

1 INTRODUCTION

Knowledge graph construction is a key application for transforming Web content into machine-processable data [9]. A fundamental task in knowledge graph construction is link prediction, where the goal is to create or recover missing links in knowledge graphs, e.g., identifying the birthplace of a person or the CEO of a company. This task is generally formulated as a statistical relational learning problem, for which a variety of techniques such as latent factor models [25], random walks [19] and neural networks [22] have been explored. Among them, neural network-based methods, which learn semantic representations of entities and relations, have significantly advanced the state of the art in the past few years [5, 8, 30, 33, 38].

A focal point of research efforts on neural network-based methods has been designing models that are scalable enough to be applied to large knowledge graphs (e.g., millions of facts), while being able to precisely capture the semantics of entities and relations. Shallow models, such as translational distance models (e.g., TransE [5] and TransH [36]) and semantic matching models (e.g., DistMult [38], ComplEx [33]), were proposed to leverage simple algebraic operations – e.g., inner product and matrix-vector product – for modeling interactions among entities and relations in learning their representations [35]. While being scalable, these models are intrinsically lacking expressive power and thus are limited in capturing fine-grained semantics [8]. In contrast, deep models [9, 30] are better at learning expressive features. However, these models are more complex in terms of parameters. Consequently, model training requires very large amounts of data, which are rarely available in real-world scenarios [35].

Few studies have tackled this problem from the perspective of *data selection*. Better selecting training data is not only useful for reducing the size of the training data, thereby improving the efficiency of model training, but also beneficial for reducing the cost in acquiring additional data to improve link prediction performance. In this work, we propose to adopt active learning [7, 13, 27], which allows the model to choose the data from which it learns best. In the active learning setting, models initially trained on a small amount of data actively select the most informative data samples (often based on model uncertainty), which are then used in combination with existing training data to *retrain* the model. With such a selecting-retraining process iterating multiple times, the model can reach *state-of-the-art performance* with *significantly less training data* compared to conventional supervised learning settings.

The idea of combining deep learning models and active learning, i.e. *deep active learning*, has recently drawn attention for image [13, 16, 34] and text classification [29, 40]. Few approaches have however considered deep active learning for link prediction in knowledge graphs. Unlike image and textual data, the underlying data for link

prediction is relational: entities are connected by relations. This inherently provides an additional source of signals for data sampling in deep active learning: entities that are close to each other in the knowledge graph are more likely to share some similarity, thus to be more redundant for data sampling in the active learning process. By taking advantage of the underlying data structure, it is therefore possible to improve the effectiveness of data sampling.

While active learning is sample-efficient, it can still be computationally expensive since it requires to retrain models from scratch at every iteration of the active learning process. This problem becomes significant for deep active learning as it typically takes much more time to train a deep learning model than traditional machine learning models. Reducing the computational complexity is thus particularly important in our context, where the size of the training samples can easily blow up to tens of thousands of samples (an order of magnitude larger than what is used for image classification [13]). A straightforward workaround would be to incrementally train the model using the newly selected data or combining it with existing training data [29]. However, these methods will potentially bias the model towards the small amount of newly selected data, or towards the data selected in early iterations of the process. How to incrementally train deep link prediction models with optimal generalizability hence remains a key open research question.

In this paper, we introduce ActiveLink, a novel deep active learning framework for knowledge graphs that takes advantage of the underlying graph structure to improve the sampling effectiveness and to allow deep models to be incrementally trained in an unbiased manner. Our framework inherits from recent advances in Bayesian deep neural networks [12, 37] and takes a Bayesian view on existing neural link predictors. By doing so, it allows any neural link predictor to take into account uncertainty, thus enabling uncertain sampling in a deep active learning setting. In order to fully exploit the graph structure, ActiveLink incorporates uncertainty sampling into a clustering algorithm, which allows to identify redundant pieces of data in the knowledge graph for more effective data sampling. To accelerate model training, ActiveLink further adopts a principled method for unbiased incremental training based on meta-learning [2, 10]. Specifically, at each active learning iteration, we update the model parameters using the newly selected data samples with the meta-goal of generalizing the model for future predictions, which is approximated by generalizing the model based on the samples selected in preceding iterations. To do so, we strike a balance between the importance of newly and previously selected data in order to reach an unbiased estimate of the model parameters.

In summary, we make the following key contributions:

- We present a unified Bayesian view on neural link prediction, which allows any neural link predictor to be used in an uncertainty-based deep active learning setting;
- We propose a new data sampling algorithm that takes advantage of both model uncertainty and the underlying graph data structure to improve data sampling effectiveness;
- We introduce a new meta-learning method to incrementally train deep neural networks for link prediction through deep active learning with optimized generalizability;
- We demonstrate the effectiveness of our approach through an extensive evaluation on real-world datasets. Overall, ActiveLink is able to match the state-of-the-art approach while requiring *significantly less* (20% only) training data.

To the best of our knowledge, this is the first work considering deep active learning for link prediction in knowledge graphs. Our work is an initial yet important step towards improving link prediction performance through the efficient creation of training data. By strategically picking which data samples are used to train neural link predictors, ActiveLink provides an approach to effectively refine large training data and offers a route to efficiently create high-quality data (e.g., through crowdsourcing [39]). The later is highly important for scenarios where data creation is expensive, e.g., knowledge graph construction for specific domains.

2 RELATED WORK

In this section, we first discuss related work on neural network-based methods for link prediction, then review recent advances in deep active learning.

2.1 Neural Link Prediction

Existing neural link prediction methods fall into two broad categories, i.e., deep and shallow models, depending on whether or not the network contains at least one hidden layer. In the following, we review representative techniques for each category and discuss the limitations these techniques suffer from.

Shallow Models. Translational distance models are typical shallow models. A representative model is TransE [5], which learns low-dimensional representations (i.e., embeddings) for both entities and relations in a knowledge graph by minimizing the distance between linked entities. Extensions such as TransH [36] and TransR [21] project embeddings onto relation hyperplanes/spaces to learn relation-specific representations of entities. Another line of work learns representations for entities and relations by considering the match of their semantic representations as the learning target. A representative model of that line is RESCAL [24], which learns embeddings of entities while representing relations as a matrix to model the pairwise interactions between latent factors of two entities. DistMult [38] extends such a method by restricting relation matrices to diagonal ones; while being efficient, it can only be applied for symmetric relations. ComplEx [33] overcomes this issue by adopting complex-valued embeddings to model asymmetric relations. HolE [23] uses cross-correlation to decrease the number of parameters in relation representation and allows for modeling asymmetric relations. All these methods model the interactions between entities and relations through simple operations, e.g., matrix-vector products between relation matrices and entity embeddings.

Deep Models. Instead of using simple algebraic operations, a different line of work considers the use of neural networks with hidden layers to model the complex interaction patterns between entities and relations. An early piece of work [30] proposes Neural Tensor Network (NTN), which matches the embeddings of paired entities and relations by considering both linear and non-linear mappings, including those represented by relation-specific matrices and tensors and a hidden layer with non-linear transformation. Similar

models include semantic matching energy (SME) [4] and multilayer perceptrons (MLP) [9], which simplify NTN by discarding tensor parameters to improve model scalability. ConvE [8] is a recent work demonstrating that employing a 2-dimensional convolution and hidden-layers makes it possible to improve the model expressiveness while keeping a *relatively* smaller number of parameters.

Shallow models intrinsically lack expressive power, making them incapable of capturing fine-grained semantics of entities and relations [8]. Deep models have more expressive power but often involve much more parameters; consequently, model training is not scalable for large knowledge graphs in real-world scenarios. For instance, NTN has a time complexity two order of magnitude greater than TransE [35]. Our work takes an orthogonal perspective to improve the training efficiency of deep models by reducing the amount of training data while maintaining excellent performance through deep active learning.

2.2 Deep Active Learning

While extensive research has been carried out on both deep learning and active learning, researchers only recently started to investigate active learning approaches for deep neural networks. In the following, we briefly review related work that converges to the current notion of deep active learning.

Active Learning. In the traditional active learning setting, the model selects unlabeled data samples which supposedly can provide the strongest supervision; these samples are labeled and then used to retrain the model. The potential benefit of a data sample is generally measured by the model's uncertainty in making predictions for that sample, i.e., the so-called *uncertainty sampling* [7, 20]. It can be instantiated through different acquisition functions, such as maximum entropy [28], BALD [15], and variation ratios [11]. Besides uncertainty sampling, additional criteria can also be taken into account, e.g., how well a data sample will reduce the estimate of the expected error [26], which attempts to select data samples that directly optimize prediction performance. Such a criterion, however, is less practical as it is generally difficult to have an analytical expression for the expected prediction error.

Deep Active Learning. A key obstacle in applying active learning to deep neural networks is the fact that neural networks only make deterministic predictions, making it challenging to represent model uncertainty. A popular workaround has been to employ an active learning model (e.g., a Gaussian process) which is kept separate from the neural network classifier [17]. A more consistent solution is to leverage recent developments in Bayesian deep learning, which can generally be categorized into two classes. The first one is based on stochastic gradient descent (SGD). Welling et al. [1, 37] show that by adding the right amount of noise to standard SGD, the parameter converges to samples from the true posterior distribution. The second class of methods is based on dropout, which is a technique originally proposed to prevent over-fitting in training deep learning models [14, 31]. Gal et al. [12] showed that when considering dropout during prediction in a similar way as for model training, the predictions are equivalent to sampling from the approximate true posterior distribution of the parameters, thus turning a deterministic predictive function into a stochastic (uncertain) one.

Recent Applications. Deep active learning was first applied to image classification [13, 16]. These works showed that the Bayesian approach significantly outperforms deterministic predictions in deep active learning across multiple uncertainty-based acquisition functions. Recently, a few studies have investigated deep active learning approaches to textual data, including sentence classification [40] and named entity recognition [29]. These applications demonstrate that deep active learning can considerably reduce the amount of data used to train the model.

To the best of our knowledge, we are the first ones to leverage deep active learning for link prediction. Unlike image and textual data, data samples in link prediction are connected by a set of links, which provides an additional source of signals that we take advantage of thanks to a new data sampling algorithm. Moreover, we introduce a new principled method for incremental training, which is particularly useful to improve the computational efficiency of deep active learning on link prediction.

3 THE ACTIVELINK FRAMEWORK

This section introduces our proposed framework for deep active learning, which we refer to as ActiveLink. We first introduce a unified Bayesian view on neural link prediction, which allows any neural link predictor to represent prediction uncertainty, thereby enabling uncertainty sampling in ActiveLink. Next, we introduce two key features of ActiveLink: 1) structured uncertainty sampling, a data sampling method that leverages both model uncertainty and the underlying structure of the knowledge graph to better approximate sample informativeness ; 2) meta-incremental training, a meta-learning approach that allows the link prediction model to be incrementally trained in new iterations of the active learning process by optimizing the generalizability of the model for future predictions.

Problem Statement. Throughout this paper we use boldface lowercase letters to denote vectors and boldface uppercase letters to denote matrices. We use capital letters (e.g., $\mathcal{P}$) in calligraphic math font to denote sets. Following this, we denote a knowledge graph by $\mathcal{G} = \{(s, r, o)\} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $\mathcal{E}$ is the entity set, $\mathcal{R}$ is the relation set, and where a triple (s, r, o) represents a relationship $r \in \mathcal{R}$ between a subject $s \in \mathcal{E}$ and an object $o \in \mathcal{E}$.

Given a budget B, our goal is to select B triples from $\mathcal{G}$ to train a deep neural link predictor with optimal performance. We consider an active learning approach to this problem, which decomposes the problem into multiple iterations. At every iteration, we select k ($k < B$) triples from $\mathcal{G}$, and update the parameters of the link predictor using these triples in combination with previously selected triples. We consider data sampling performed in a greedy fashion, that is, at each iteration we select only k triples with the highest informativeness as determined by ActiveLink. The key problem for ActiveLink is therefore two-fold. 1) Effectiveness: selecting the most informative triples at each iteration, which is important for training a link predictor with high accuracy. 2) Efficiency: training the link predictor in an incremental fashion. Effectiveness and efficiency are addressed by structured uncertainty sampling and meta-incremental training, respectively, as we introduce next.

3.1 Uncertainty Sampling

We start by introducing a unified Bayesian view on neural link predictors. Based on this abstraction, we then introduce an uncertainty sampling method for deep active learning of neural link predictors.

Neural Link Predictors as Scoring Functions. Neural link predictors are often viewed as scoring functions $f_r(s, o)$, which in our context take as input a potential triple (s, r, o), and output a score representing the plausibility of the triple being true. Taking this view, the classic MLP model [9] is defined as follows:

$$f_r(s, o) = \mathbf{w}^\intercal g([\mathbf{s}; \mathbf{r}; \mathbf{o}]\mathbf{W}) \tag{1}$$

where $\mathbf{s}$ and $\mathbf{o}$ are entity embeddings, and $\mathbf{r}$ is a relation embedding; these embeddings are concatenated (denoted by $[;]$) to be used as the input for a fully connected layer parameterized by the linear transformation matrix $\mathbf{W}$ and a non-linear function g (e.g., tanh); $\mathbf{w}$ is a linear transformation vector that takes as input the output of the fully connected layer to obtain the final score.

The state-of-the-art model Convolutional 2D Knowledge Graph Embeddings (ConvE) [8] is similarly defined as follows:

$$f_r(s, o) = g(vec(g([\bar{\mathbf{s}}; \bar{\mathbf{r}}] * w))\mathbf{W})\mathbf{o} \tag{2}$$

where $\bar{\mathbf{s}}$ and $\bar{\mathbf{r}}$ are 2D reshaping of the entity and relation embeddings, which are concatenated and used as an input for a 2D convolutional layer with filters w; vec is a vectorization operation that transforms the feature map given by the convolutional layer to a vector; $\mathbf{W}$ is the weight matrix and g is a non-linear function (rectified linear units (ReLU) [18] are used in the original work).

Neural Link Predictors as Bayesian Models. To allow for uncertainty sampling in deep active learning, we adopt the Bayesian approach to deep neural networks recently developed by Gal and Ghahramani [12]. To this end, we first reformulate neural link predictors as parameterized likelihood functions, such that the output represents the probability of a triple being true. We then pose a prior distribution on the parameters, to transform a deterministic neural link predictor into a Bayesian model.

Denoting all parameters, including entity and relation embeddings and all weight parameters (i.e., linear transformation matrices and vectors), as Θ, we can rewrite any neural link predictor as $f_r^\Theta(s, o)$, which can then be reformulated as a likelihood function by introducing an additional softmax layer:

$$p(y|(s, r, o), \Theta) = \text{softmax}\,(f_r^\Theta(s, o)) \tag{3}$$

where y is the output of the link predictor representing whether a triple is true or not. To make neural link predictors Bayesian, we define a prior over the parameters Θ:

$$\Theta \sim p(\Theta|K) \tag{4}$$

e.g., a standard Gaussian prior parameterized by K (the co-variance matrix). With this definition, model training will result in a posterior distribution over the parameters, i.e. $p(\Theta|\mathcal{D}_{train})$ (where $\mathcal{D}_{train}$ is the training data), instead of point estimates (i.e., fixed values for the parameters). The prediction for an arbitrary input (s, r, o) can be described as a likelihood function:

$$p(y|(s, r, o), \mathcal{D}_{train}) = \int p(y|(s, r, o), \Theta)p(\Theta|\mathcal{D}_{train})d\Theta \tag{5}$$

which provides a more robust prediction than non-Bayesian methods as it takes into account the uncertainty of the parameters.

The problem of inferring the exact posterior distribution for the parameters, $p(\Theta|\mathcal{D}_{train})$, is intractable. Gal and Ghahramani [12] recently proposed in that context Monte Carlo (MC) dropout, which is a simple yet effective method for performing approximate variational inference. MC dropout is based on dropout [14, 31], which is typically used during model training to randomly drop hidden units of the network at each parameter update iteration; this reduces complex co-adaptations of neurons that can easily lead to overfitting. Gal and Ghahramani [12] prove that by performing dropout during the forward pass when making predictions, the output is equivalent to the prediction when the parameters are sampled from a variational distribution of the true posterior.

Formally, MC dropout is equivalent to sampling from a variational distribution $q(\Theta)$ that minimizes the Kullback-Leibler (KL) divergence to the true posterior $p(\Theta|\mathcal{D}_{train})$. Given this, we can perform a Monte Carlo integration to approximate Equation 5:

$$\begin{aligned} p(y|(s, r, o), \mathcal{D}_{train}) &\approx \int p(y|(s, r, o), \Theta)q(\Theta)d\Theta \\ &\approx \frac{1}{T}\sum_{t=1}^{T} p(y|(s, r, o), \widehat{\Theta}) \end{aligned} \tag{6}$$

where $\widehat{\Theta}$ is the parameters sampled T times from $q(\Theta)$, i.e., $\widehat{\Theta} \sim q(\Theta)$. To summarize, MC dropout provides a practical way to approximately sample from the true posterior without explicitly calculating the intractable true posterior.

Deep Uncertainty Sampling. Active learning aims at selecting the most informative data samples (triples, in our case) to train the model. This is often formulated as an acquisition function, defined as follows:

$$(s, r, o)^* = \underset{(s,r,o)\in\mathcal{G}}{\arg\max}\ \phi((s, r, o)) \tag{7}$$

The key to developing a good sampling strategy is designing an effective informativeness measure. With the Bayesian formulation of neural link predictors, the informativeness of a triple can be quantified by model uncertainty. A typical uncertainty measure is Shannon entropy:

$$\begin{aligned} &\phi((s, r, o)) = H[y|(s, r, o), \mathcal{D}_{train}] \\ &= -\sum_{C\in\{0,1\}} p(y = C|(s, r, o), \mathcal{D}_{train}) \log p(y = C|(s, r, o), \mathcal{D}_{train}) \\ &= -\sum_{C\in\{0,1\}} (\frac{1}{T}\sum_t \hat{p}_C^t) \log(\frac{1}{T}\sum_t \hat{p}_C^t) \end{aligned} \tag{8}$$

where $\frac{1}{T}\sum_t \hat{p}_C^t$ is the averaged predicted probability of class C for (s, r, o) ($C \in \{0, 1\}$ i.e., the triple being true or not), sampled T times by Monte Carlo dropout. Note that Θ is marginalized in the above equation as in Equation 6.

3.2 Structured Uncertainty Sampling

Besides model uncertainty, knowledge graphs provide an additional source of signals to represent data informativeness given their data structure: entities are linked by relations. Considering an existing pool of data sampled in preceding iterations, we make the following assumptions:

(1) Entities linked with those in the training pool are less informative than entities not linked;

(2) More generally, entities that are intensively linked with each other are less informative with respect to each other than those sparsely linked.

Relations therefore provide a source for measuring the informativeness of data samples from the perspective of the redundancy between data samples. This comes in contrast to representing informativeness from the perspective of the models. We are, therefore, interested in designing a data sampling method that exploits relations in knowledge graphs and that potentially combines it with model uncertainty.

To this end, we leverage clustering algorithms to identify groups of entities based on relations among entities. Our basic assumption is that entities belonging to the same cluster are more redundant with each other. Therefore, we construct our initial training set by picking from each of the c clusters a single data sample (i.e., a triple). After the model is trained on the initial set, at each of the following active learning iterations we select clusters based on their potential redundancy with respect to existing training data. Following the idea that data samples belonging to the same cluster are less informative, we select a sample from each of the selected clusters to form a new dataset for model training.

The above process is described by the high-level pseudo code in Algorithm 1. The key steps are the selection of clusters (row 10-12) and the selection of data samples from the clusters (row 13-15). For cluster selection, the redundancy score (row 11) is calculated as the averaged similarity between all data samples — specifically, cosine similarity between entity embeddings — in the cluster and those in the existing training pool. The selection of data samples within the cluster is based on model uncertainty (row 14). By doing so, we incorporate uncertainty sampling into the sampling algorithm.

We consider K-means for clustering entities, which are represented by low-dimensional embedding learned through TransE [5]. K-means allows picking the number of resulting clusters c. If c is less than the sample size k required for the active learning iteration, the sampling algorithm will be collapsed such that cluster selection is not considered and data samples will be selected from all clusters. In this case, we take the n_j most uncertain triples from cluster C_j, where n_j is proportional to the size of C_j.

We note that our algorithm is not restricted to any specific clustering algorithm. More advanced clustering methods can be applied, e.g., those based on relations among entities as well as entity attributes [3, 6]. Since our goal is to show that clustering can be an effective means to leverage the underlying structure of knowledge graphs for data sampling, we experiment on K-means as it is one the most widely used clustering methods. Comparison of different clustering methods is left for future work.

Algorithm 1: Structured Uncertainty Sampling

Input: Knowledge graph $\mathcal{G} = \{(s, r, o)\}$, budget B, #clusters c, #samples per iteration k ($k < B$), link predictor M
Output: Triples sampled at i-th iteration $\mathcal{T}_i$ ($i \in \{0, 1, \ldots\}$)

1 $C \leftarrow$ Cluster entities in $\mathcal{G}$;
2 $C^k \leftarrow$ Pick k clusters from C;
3 $\mathcal{T}_0 \leftarrow \emptyset$;
4 **foreach** $C_j \in C^k$ **do**
5 $(s, r, o) \leftarrow$ Pick a triple from C_j;
6 Add (s, r, o) to $\mathcal{T}_0$;
7 **while** $\sum_{l=1}^{i} |\mathcal{T}_l| \leq B$ **do**
8 Incrementally train M on $\mathcal{T}_i$;
9 $i++; \mathcal{T}_i \leftarrow \emptyset$;
10 **foreach** $C_j \in C$ **do**
11 Compute $redundancy_score(C_j)$;
12 $C^k \leftarrow$ Pick k clusters based on the scores;
13 **foreach** $C_j \in C^k$ **do**
14 $(s, r, o) \leftarrow \arg\max\ \phi((s, r, o)), \forall s \in C_j$;
15 Add (s, r, o) to $\mathcal{T}_i$;

3.3 Meta-Incremental Training

In the traditional active learning setting, newly selected data samples are combined with existing training data to retrain the model from scratch. This is highly time-consuming when training deep models (our experiments show that model retraining from scratch takes 3 times longer than updating the model parameters from the previous iteration; see Section 4.3 for more details). We are, therefore, interested in incremental training of deep models using the newly selected data samples in every iteration of active learning.

A straightforward workaround is to fine-tune the model trained on the previous iterations using the new data samples only, which is a widely used approach in transfer learning. Such an approach, however, is likely to bias the model to the small amount of newly selected data. To prevent the model to get biased, recent work has proposed to combine the newly selected data with the existing training data and incrementally train the model with a small number of epochs [29]. This however, may not allow to fully exploit the newly selected data, compared with the data selected early that has already been used to update parameters in previous iterations. A central problem here is how to strike a good balance between the importance of new and previously selected data to reach an unbiased estimate of model parameters.

To solve this problem, we propose to adopt a meta-learning approach, where we update model parameters with the meta-goal that the updated model is most generalizable to future predictions. This is achieved in a two-step parameter updating scheme: we temporarily update model parameters with a standard gradient descent step using the newly selected data, followed by a meta-learning step where the learning algorithm performs a gradient descent step with the objective that model parameters will be more generalizable. The generalizability of the updated model parameters is approximated by the prediction loss on data samples selected in the current and the previous iterations.

Formally, in the i-th active learning iteration, model parameters learned in the previous iteration, denoted by Θ_{i-1}, are first updated to Θ'_i by a gradient descent step on the newly selected data $\mathcal{T}_i$:

$$\Theta'_i = \Theta_{i-1} - \alpha \Delta_\Theta \mathcal{L}(f_{\Theta_{i-1}}, \mathcal{T}_i) \tag{9}$$

where f denotes the model, α is the learning rate, $\mathcal{L}(f_{\Theta_{i-1}}, \mathcal{T}_k)$ is the loss function applied to the data samples of $\mathcal{T}_i$ given existing model Θ_{i-1}, and Δ_Θ is the gradient of Θ with respect to the loss.

Algorithm 2: Meta-Incremental Training

Input: The current iteration i, data selected in the current iteration $\mathcal{T}_i$, model parameter from the previous iteration Θ_{i-1}, window size w, data selected in iterations within the window $\{\mathcal{T}_{i-w}, \mathcal{T}_{i-w+1}, \ldots, \mathcal{T}_{i-1}\}$, learning rate α and β

1 **Output:** Updated model parameters Θ_i
2 **while** *True* **do**
3 **for** $l = i - w; l \leq i; l++$ **do**
4 Evaluate $\Delta_\Theta \mathcal{L}(f_{\Theta_l}, \mathcal{T}_l)$;
5 $\Theta'_l \leftarrow \Theta_{i-1} - \alpha \Delta_\Theta \mathcal{L}(f_{\Theta_{i-1}}, \mathcal{T}_l)$;
6 $\Theta_i \leftarrow \Theta_{i-1} - \beta \Delta_\Theta \sum_{l=i-w}^{i} \mathcal{L}(f_{\Theta'_l}, \mathcal{T}_l)$;
7 $\Theta_{i-1} \leftarrow \Theta_i$;
8 **if** *converged* **then**
9 break;

To avoid overfitting, we use a meta-learner that makes use of the triples selected in previous iterations within a certain time window w, i.e. $\{\mathcal{T}_{i-w}, \mathcal{T}_{i-w+1}, \ldots, \mathcal{T}_{i-1}\}$, and update the parameters such that the updated model generalizes well to data in these iterations:

$$\min_{\Theta} \sum_{l=i-w}^{i} \mathcal{L}(f_{\Theta'_l}, \mathcal{T}_l) = \sum_{l=i-w}^{i} \mathcal{L}(f_{\Theta_{i-1} - \alpha \Delta_\Theta \mathcal{L}(f_{\Theta_{i-1}}, \mathcal{T}_l)}, \mathcal{T}_l) \quad (10)$$

wherein Θ'_l represents the parameters updated with Equation 9 using data selected in the l-th iteration ($i - w \leq l \leq i$). With such an objective, the meta-learning step is then performed by stochastic gradient descent as follows:

$$\Theta_i = \Theta_{i-1} - \beta \Delta_\Theta \sum_{l=i-w}^{i} \mathcal{L}(f_{\Theta'_l}, \mathcal{T}_l) \quad (11)$$

where β is the meta-learning rate. Such a meta-learning step involves a second-order derivative (i.e., Hessian) with respect to the parameters, which is readily supported by some standard deep learning libraries.

The overall incremental training algorithm is described in Algorithm 2. We use mini-batch gradient descent for Meta-Incremental training, where mini-batches of data samples selected in the current and previous iterations are used in the inner loop of the meta-learning process (row 3-5) and different mini-batches of these iterations are used for meta-learning loops. Parameters are updated by the meta-learning step (row 6) several times until the current iteration converges.

4 EXPERIMENTS AND RESULTS

In this section, we report on a set of experiments we have conducted to evaluate the performance of ActiveLink.[1] We first evaluate our new data sampling method and the incremental training method of ActiveLink separately, by answering the following question:

- **Q1:** How effective are our uncertainty sampling and structured uncertainty sampling methods in determining the informativeness of data samples?
- **Q2:** How effective is our meta-incremental training method in speeding up the deep active learning process?

Subsequently, we evaluate ActiveLink as a whole by considering the following issues:

- **Q3:** How effective is ActiveLink compared with a traditional supervised learning method (i.e., non-active learning) for training neural link predictors?
- **Q4:** How do parameter settings of ActiveLink, including #samples per iteration and #clusters for data sampling and window size for incremental learning, affect the performance?

In addition, we investigate the generalizability of Activelink across neural link predictors and its scalability on a big dataset. In the following, we start by introducing our experimental setup, before answering each of the above questions in a separate subsection.

4.1 Experimental Settings

Dataset. We experiment on two publicly accessible datasets: FB15K-237 and WikiMovie:

- **FB15K-237** [32] is a subset of Freebase widely used for link prediction evaluation [5, 8, 36]. The knowledge graph mainly describes facts about sports, movies and actors. The original dataset FB15K was contributed by Bordes et al. [5]. Toutanova and Chen [32] noted that FB15K suffers from the test leakage problem: simple rule-based models can reach high prediction performance on the test triples by inverting triples in the training set. We use FB15k-237, a corrected version where inverse relations are removed.
- **WikiMovie** is a subset of Wikidata[2] that contains facts about movies such as directors, actors and genre. The original data is large but sparse. For example, 53% of the entities appear in only one triple, making it difficult to evaluate the link prediction models; by contrast, 95% of the entities in FB15K-237 have at least three training examples. As such, we filtered the dataset to keep a subset with only entities that appear in at least two triples. We prepare two versions of the WikiMovie dataset, namely WikiMovie-300K and WikiMovie-1M, which contain 300K and 1M triples, respectively.

We report key statistics of our datasets in Table 1. We note that WikiMovie-300K contains a comparable number of triples as FB15K-237 yet much more entities. Consequently, link prediction is more difficult on WikiMovie-300K. The comparison of ActiveLink performance on these two datasets, therefore, helps to investigate the impact of data sparsity on ActiveLink performance. WikiMovie-1M is used below to study the scalability of the ActiveLink framework in terms of reducing the amount of data needed to train neural link predictors on large knowledge graphs.

Comparison Methods. To demonstrate the effectiveness of both model uncertainty and data redundancy (considering the underlying data structure), we compare the following data sampling methods:

- **Random**, which randomly selects data samples at every active learning iteration;
- **Structured**, a variant of our method that randomly selects triples from each cluster (similar to stratified sampling);

[1] Our code and data are available at https://github.com/eXascaleInfolab/ActiveLink.

[2] https://www.wikidata.org/

Table 1: Descriptive statistics of the datasets.

	#Entities	#Relationships	#Triples
FB15K-237	14,541	474	310,116
WikiMovie-300K	36,001	588	286,683
WikiMovie-1M	104,500	788	987,896

- **Uncertainty**, a variant of our method that selects data samples based on Shannon entropy (Equation 8);
- **Structured-Uncertainty**, which selects data samples based on both model uncertainty and data redundancy (our proposed Algorithm 1).

We note that Uncertainty is a method that has been investigated in image recognition [13]; however, to the best of our knowledge, we are the first to apply it to link prediction in knowledge graphs.

To investigate the effectiveness of our proposed incremental training method, we compare the following model training techniques for deep active learning:

- **Retrain**, the conventional way of performing active learning, which trains the model from scratch at each new iteration;
- **Incremental** [29], the baseline incremental training method that combines newly and previously selected data to incrementally train the model from the last iteration;
- **Meta-Incremental**, our incremental training method that adopts meta-learning to get an unbiased estimate of the model parameters (Algorithm 2).

We evaluate the above model training methods in terms of both model performance and training efficiency.

Finally, to show the superiority of ActiveLink in model training over traditional supervised learning settings as well as to demonstrate the generalizability of ActiveLink on different neural link predictors, we apply both training settings to two state-of-the-art deep neural link predictors (see Section 3.1 for details):

- **ConvE** [8], a model that uses two-dimensional convolutions over entity and relation embeddings for link prediction;
- **MLP** [9], a multi-perceptron model using fully-connected layers over entity and relation embeddings.

Parameter Settings for Link Predictors & ActiveLink. The parameters of the link predictors and ActiveLink are empirically set based on a held-out validation set that contains 10% of the original data. For the link predictors (i.e., ConvE and MLP), we apply grid search in {20, 50, 100, 200} for the dimension of the embeddings and in {0.1. 0.3, 0.5} for the dropout on different layers of the prediction model. We select learning rates from {0.0001, 0.001, 0.01, 0.1, 1} and batch size from {64, 128, 256}. In particular, we use 3×3 filters for the convolution in ConvE. For ActiveLink, the number of samples per iteration is selected from {500, 1000, 5000, 10000}. The number of clusters for the Structured-Uncertainty sampling method is selected from {10, 100, 1000, 10000}. The window size for our Meta-Incremental method is selected from {1, 5, 10, 20, inf}, where inf indicates that all preceding iterations are considered for meta-learning.

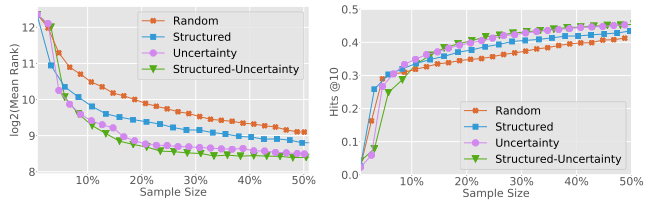
Evaluation Protocols. Evaluation is performed on a test set that contains 10% triples from the corresponding dataset; the remaining data is used as the training and validation sets. Following previous studies [5, 8, 21], we measure the performance using the following two metrics: 1) Mean Rank of correct entities, and 2) proportion of correct entities in the top-10 ranked entities (Hits@10). Lower Mean Rank and higher Hits@10 indicate better performance.

4.2 Comparative Results on Data Sampling (Q1)

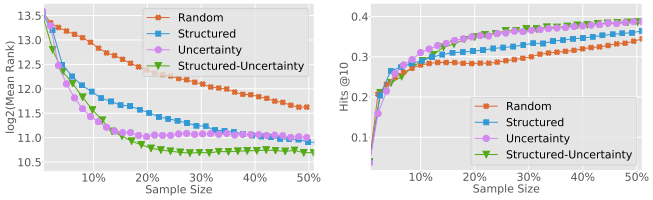
We start by investigating the effectiveness of our proposed sampling methods (Uncertainty, Structured, and Structured-Uncertainty) by comparing them against random sampling (Random). Results are reported in Figure 1, where performance is shown as a function of the fraction of training data (from zero to 50%) selected at different iterations of the active learning process.

From the figure, we observe that Random sampling is outperformed by Structured sampling and Uncertainty sampling across the two datasets and across the two performance metrics. This clearly demonstrates the effectiveness of model uncertainty and data redundancy in data sampling. Among Structured and Uncertainty sampling, we observe that Structured sampling generally performs better than Uncertainty sampling in the early iterations of the active learning process (before 5% samples are selected); however, in the late iterations, Structured sampling is outperformed by Uncertainty sampling. Such a pattern can be explained by the evolution of the size of the samples during the active learning process. In the early iterations of the active learning process, the size of the data being sampled is big as compared with the size of data samples already selected; it is, therefore, easy to select non-redundant data samples by considering the underlying data structure; when the size of selected data samples increases, the newly selected samples get increasingly more redundant with the selected samples, thus decreasing the effectiveness of Structured sampling. In comparison, model uncertainty is less affected by the size of the selected data samples. We note that, when the size of the selected data samples increases (30% to 40%), the performance gains of the link prediction model decreases even for Uncertainty sampling. This indicates the decreased utility of the data samples in the late iterations for improving model performance (see the next part of the results for additional evidence on that point).

Structured-Uncertainty sampling, which combines both model uncertainty and data redundancy in data sampling, achieves the best performance across the two datasets and the two performance metrics. The performance gains are most obvious for the WikiMovie-300K dataset measured by Mean Rank, as shown in Figure 1(c). We note that while not visually obvious in the other subfigures (due to the need for depicting the full range of the variations), Structured-Uncertainty consistently outperforms Uncertainty in every iteration as measured by both metrics: overall, the average improvements are 29.37 (Mean Rank) and 0.1% (Hits@10) for the FB15K-237 dataset when the sample size is between 40% and 50%; for the WikiMovie-300K dataset with the same sample sizes, the average improvements are 400.77 (Mean Rank) and 0.1% (Hits@10). These results highlight the difference between model uncertainty and data redundancy as data informativeness criterion and the effectiveness of combining both of them for data sampling.

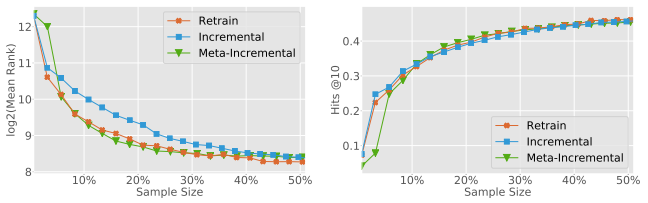


(a) FB15K-237 - Mean Rank (b) FB15K-237 - Hits@10

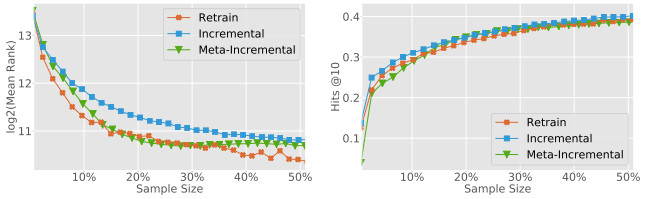


(c) WikiMovie-300K - Mean Rank (d) WikiMovie-300K - Hits@10

Figure 1: Comparison between data sampling methods. Upper figures compare the performance of the data sampling methods measured by (a) Mean Rank (semi-log scale) and (b) Hits@10 on the FB15K-237 dataset; lower figures compare the performance by (c) Mean Rank (semi-log scale) and (d) Hits@10 on the WikiMovie-300K dataset.



(a) FB15K-237 - Mean Rank (b) FB15K-237 - Hits@10



(c) WikiMovie-300K - Mean Rank (d) WikiMovie-300K - Hits@10

Figure 2: Comparison between model training methods. Upper figures compare the performance of the model training methods measured by (a) Mean Rank (semi-log scale) and (b) Hits@10 on the FB15K-237 dataset; lower figures compare the performance by (c) Mean Rank (semi-log scale) and (d) Hits@10 on the WikiMovie-300K dataset.

Compared with Random sampling, Structured-Uncertainty improves link prediction performance by 250.13 (Mean Rank) and 4.5% (Hits@10) for the FB15K-237 dataset and by 1710.04 (Mean Rank) and 5.2% (Hits@10) for the WikiMovie-300K dataset when 50% data samples are selected in both datasets.

4.3 Comparative Results on Incremental Training (Q2)

We now compare our Meta-Incremental training method with Retrain and the Incremental training baseline. Among them, Retrain should perform best in terms of model performance as Retrain uses all data selected in the new and previous iterations to reach an unbiased estimate of model parameters. Figure 2 shows the results.

From the results by Mean Rank in Figures 2(a,c), we observe that both Retrain and our Meta-Incremental training method outperform the Incremental baseline, and importantly, that our method performs as well as Retrain. This indicates that our proposed incremental training method indeed succeeds in obtaining an unbiased estimate of model parameters and achieves optimal model generalizability. The performance difference between our method and the Incremental baseline highlights the importance of including an unbiased estimation of the model parameters. We note that all three methods converge when the size of the data used in active learning reaches 50% of the size of the original training data, which also verifies the decreased utility of data selected in late iterations.

The above trends are not obvious from the results measured by Hits@10 in Figure 2(b,d): no clear difference is observed between the three model training methods. The different results measured by Mean Rank and Hits@10 implies that Hits@10 is a less sensitive metric in evaluating data sampling methods. This is likely due to the fact that Hits@10 is a set-based metric that does not take into account the ranking position of true positives in the *top-10* results, thus it is less sensitive in differentiating model performance, especially given the large number of entities in the experimental datasets.

Table 2: Model training time (in minutes).

	Retrain	Meta-Incremental
FB15K-237	499	177
WikiMovie-300K	945	334

Model Training Efficiency. In order to evaluate the runtime performance of our proposed Meta-Incremental training method, we investigate the model training time with up to 50% of the data samples selected using our test server[3], for both the Retrain method and our Meta-Incremental training method (both achieve optimal model performance). Table 2 reports the results. As shown by the table, our Meta-Incremental method requires a much shorter training time on both FB15K-237 and WikiMovie-300K: it speeds up model training by 2.8x in both cases.

In summary, our Meta-Incremental training method not only achieves optimal model generalizability, but also dramatically reduces the model training time.

4.4 ActiveLink vs. Non-active Learning (Q3)

To evaluate the effectiveness of ActiveLink as a whole, we compare the performance of neural link predictors trained with ActiveLink and those trained with a traditional supervised learning setting

[3] An Ubuntu 14.4 machine with a GeForce GTX TITAN X and 3.3 GHz CPU.

Table 3: Performance of ConvE and MLP trained by ActiveLink and the non-active learning setting on a varying fraction of the FB15K-237 and WikiMovie-300K datasets, measured by both Mean Rank and Hits@10.

Predictor	Fraction	FB15K-237				WikiMovie-300K			
		Mean Rank		Hits@10		Mean Rank		Hits@10	
		Non-Act.	ActiveLink	Non-Act.	ActiveLink	Non-Act.	ActiveLink	Non-Act.	ActiveLink
ConvE	10%	1325.26	409.14	0.255	0.403	6857.77	2150.08	0.219	0.329
	20%	822.88	351.98	0.282	0.440	5253.15	1641.98	0.259	0.365
	30%	605.79	339.48	0.301	0.450	4317.67	1673.97	0.271	0.379
	40%	537.12	329.21	0.326	0.459	3406.10	1658.99	0.296	0.388
	50%	458.91	318.18	0.349	0.464	2832.33	1628.06	0.314	0.396
MLP	10%	1386.80	487.64	0.248	0.395	7094.62	2520.71	0.216	0.300
	20%	848.21	374.75	0.283	0.440	5613.55	1536.02	0.248	0.357
	30%	663.18	346.94	0.314	0.457	4463.22	1379.29	0.274	0.376
	40%	547.16	326.11	0.332	0.467	3370.01	1326.25	0.298	0.389
	50%	458.51	322.40	0.354	0.470	2680.69	1372.14	0.325	0.398

using varying fractions of the experimental datasets. To demonstrate the generalizability of ActiveLink on different neural link predictors, we experimented with two neural link predictors, i.e., ConvE and MLP, and report the results in Table 3.

We observe that as the size of the training samples increases, the performance of ConvE and MLP also increases in both active learning and non-active learning settings. Importantly, ActiveLink consistently outperforms the traditional supervised learning setting across all experimental configurations in terms of the neural link predictor, the dataset, the fraction of the dataset for model training, and the performance metric. Such a result clearly indicates the superiority of ActiveLink in training neural link predictors when only a fraction of the dataset is used for model training. The performance gains achieved by ActiveLink flatten in the long run when most data samples are used for model training; this implies that the training set can no longer offer new informative data samples for improving model performance. Comparing the results on the two datasets, we observe that the performance gains through ActiveLink are larger for WikiMovie-300K than for FB15K-237, as measured by Mean Rank. Recalling that WikiMovie-300K contains 2.5x more entities than FB15K-237 does, these results suggest that ActiveLink brings more performance gains for sparse knowledge graphs.

We further compare the performance of ConvE and MLP trained by ActiveLink using a fraction of the dataset and by the supervised learning setting with the *full* dataset. Our experiments show that ActiveLink achieves nearly state-of-the-art results with only half of the data: for the FB15K-237 dataset, ActiveLink reaches 99.6% in Mean Rank and 96.4% in Hits@10 for ConvE, and 99.4% in Mean Rank and 94.6% in Hits@10 for MLP; for the WikiMovie-300K dataset, ActiveLink reaches 98.6% in Mean Rank and 97% in Hits@10 for ConvE, and 99.1% in Mean Rank and 97.4% in Hits@10 for MLP. These results strongly demonstrate the effectiveness of ActiveLink in data utilization when training neural link predictors.

Scalability for Large Datasets. In order to evaluate the performance of ActiveLink on large knowledge graphs, we compare it against a non-active learning setting on the WikiMovie-1M dataset. Results with ConvE as the neural link predictor are shown in Table 4 (similar results are observed with MLP). Overall, ActiveLink reaches 99% in MeanRank and 91% in Hits@10 with only **20%** of the data used for training. Compared with the previous results on the smaller datasets (FB15K-237 and WikiMovie-300K), these results show that ActiveLink is more effective in reducing the amount of training data required for large knowledge graphs.

Table 4: Performance of ConvE trained by ActiveLink using 20% of the WikiMovie-1M dataset and by the non-active learning setting using the full dataset.

	Mean Rank	Hits@10
ActiveLink	3356	0.337
Non-active learning	2341	0.371

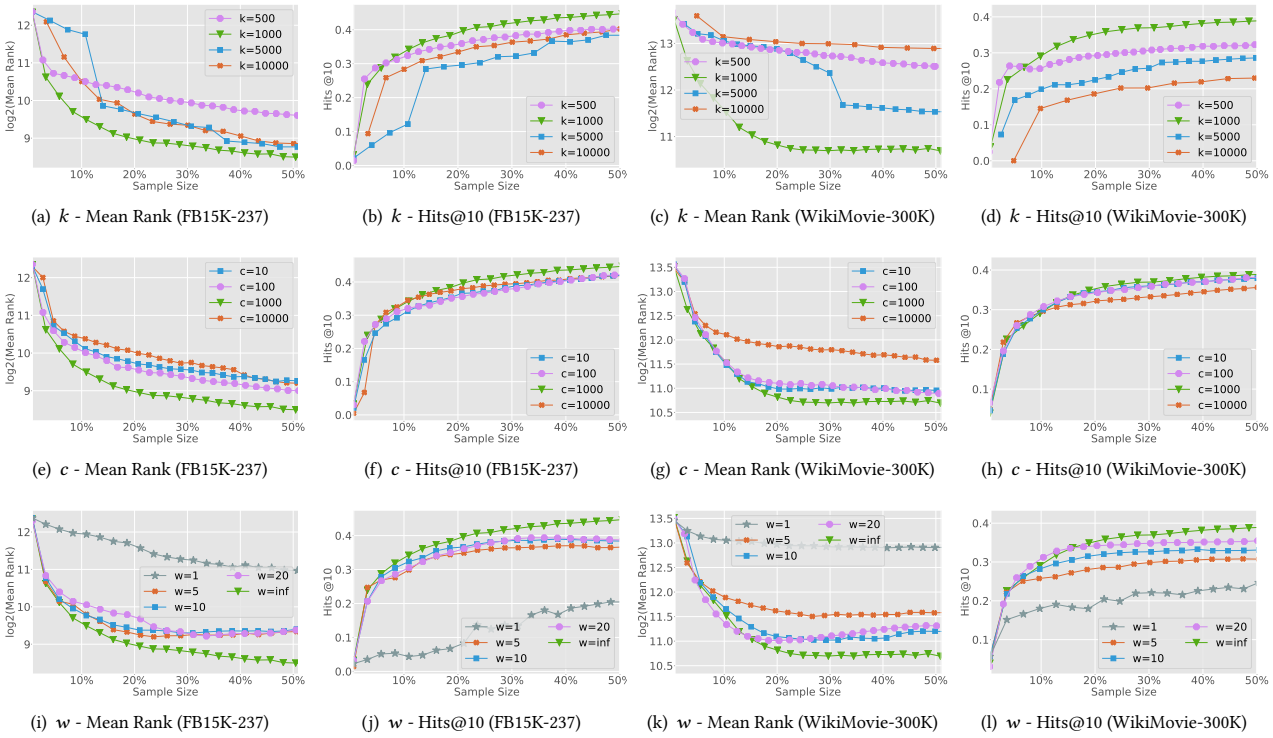
As a remark, we note that ActiveLink is less time efficient compared to the non-active learning setting due to the multiple training iterations. This, however, comes as a necessity to reduce the amount of data required for model training.

4.5 Parameter Sensitivity (Q4)

We now investigate the impact of parameters on the performance of ActiveLink. Figure 3 shows the performance of ActiveLink with different values for the number of samples per iteration (k), the number of clusters (c) and the window size w on the FB15K-237 and WikiMovie-300K datasets.

Impact of k. Due to the high requirement in terms of training data by deep models, the sample size k of individual iterations is supposed to be bigger in deep active learning settings than in traditional active learning settings. However, when the sample size gets too large, the benefits of increased data samples might be less significant than the benefits of training a better prediction model using fewer data samples and using that model to select new data samples for the next iteration. Figures 3(a-d) confirm this hypothesis: as the sample size increases, the performance first increases then decreases. The optimal performance is achieved when k is set to 1000.

Impact of c. The number of clusters c in our proposed Structured-Uncertainty sampling method controls the strength of the influence



(a) k - Mean Rank (FB15K-237) (b) k - Hits@10 (FB15K-237) (c) k - Mean Rank (WikiMovie-300K) (d) k - Hits@10 (WikiMovie-300K)

(e) c - Mean Rank (FB15K-237) (f) c - Hits@10 (FB15K-237) (g) c - Mean Rank (WikiMovie-300K) (h) c - Hits@10 (WikiMovie-300K)

(i) w - Mean Rank (FB15K-237) (j) w - Hits@10 (FB15K-237) (k) w - Mean Rank (WikiMovie-300K) (l) w - Hits@10 (WikiMovie-300K)

Figure 3: Impact of (a) #samples per iteration k, (b) #clusters c and (c) window size w on the performance of ActiveLink on both FB15K-237 and WikiMovie-300K datasets, measured by Mean Rank (shown in semi-log scale) and Hits@10.

of the knowledge graph data structure on sampling. Structured-Uncertainty sampling with a small c relies more on the model's uncertainty when selecting data samples, while a large c relies more on the data structure of the knowledge graph. Figures 3(e-h) show that as the value of c increases, the performance first increases then decreases. The best performance is achieved when c is set to 1000, with a significant margin over the other settings. This result suggests that with an appropriate setting for the number of clusters, our proposed data sampling method can ideally combine model uncertainty with the underlying structure of the knowledge graph.

Impact of w. In the Meta-Incremental training approach, the window size w controls the size of the selected data samples for optimizing the generalizability of the incrementally trained model. A larger window will include more data from previous iterations when updating the model parameters, thereby better approximating the model's generalizability for future predictions. This intuition is confirmed in Figures 3(i-l). We observe that the performance of ActiveLink increases along with the increase of the window size. The model is biased to the latest iterations when the window size is too small (shown by the result of $w = 1$). The best performance is achieved when pieces of data from all preceding iterations are used for incremental learning. However, training with bigger window sizes takes more time. Therefore, in real applications, the selected window size would need to strike a balance between model generalizability and training efficiency. Interestingly, $w = 1$ (i.e., only data in the most recent iteration is used together with the current iteration for meta-learning of model parameters) is outperformed by all the other configurations by a large margin. Such low generalizability indicates that the model is biased to the latest iterations when the window size is too small.

5 CONCLUSION

We presented ActiveLink, a deep active learning framework for neural link prediction with optimized data sampling and model training. By taking a Bayesian view on neural link predictors, ActiveLink enables to use uncertainty sampling for deep active learning of link predictors. ActiveLink extends uncertainty sampling by taking advantage of the underlying data structure of the knowledge graph to improve sampling effectiveness. In addition, it adopts a meta-learning approach to incrementally train neural link predictors with high generalizability. We extensively evaluated our framework on three real-world datasets and showed that it can drastically reduce the amount of data needed to train a neural link predictor while reaching state-of-the-art performance. As future work, we plan to apply ActiveLink to solicit high-quality data from a crowd of annotators to further improve neural link prediction.

ACKNOWLEDGMENTS

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement 683253/GraphInt).

REFERENCES

[1] Sungjin Ahn, Anoop Korattikara, and Max Welling. 2012. Bayesian posterior sampling via stochastic gradient fisher scoring. *arXiv preprint arXiv:1206.6380* (2012).

[2] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. 2016. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems (NIPS)*. 3981–3989.

[3] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008, 10 (2008), P10008.

[4] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning* 94, 2 (2014), 233–259.

[5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems (NIPS)*. 2787–2795.

[6] Lijun Chang, Wei Li, Lu Qin, Wenjie Zhang, and Shiyu Yang. 2017. pSCAN: Fast and Exact Structural Graph Clustering. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 29, 2 (2017), 387–401.

[7] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. 1996. Active Learning with Statistical Models. *Journal of Artificial Intelligence Research (JAIR)* (1996).

[8] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*. AAAI, 1811–1818.

[9] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 601–610.

[10] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400* (2017).

[11] Linton C Freeman. 1965. *Elementary applied statistics: for students in behavioral science*. John Wiley & Sons.

[12] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*. 1050–1059.

[13] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep bayesian active learning with image data. *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 1183–1192.

[14] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving Neural Networks by Preventing Co-adaptation of Feature Detectors. *arXiv preprint arXiv:1207.0580* (2012).

[15] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. 2011. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745* (2011).

[16] Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in bayesian deep learning for computer vision?. In *Advances in Neural Information Processing Systems (NIPS)*. 5580–5590.

[17] Andreas Krause, Ajit Singh, and Carlos Guestrin. 2008. Near-optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *Journal of Machine Learning Research (JMLR)* 9, Feb (2008), 235–284.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*. 1097–1105.

[19] Ni Lao, Tom Mitchell, and William W Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 529–539.

[20] David D Lewis and William A Gale. 1994. A Sequential Algorithm for Training Text Classifiers. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 3–12.

[21] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion.. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*. AAAI, 2181–2187.

[22] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2016), 11–33.

[23] Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, et al. 2016. Holographic Embeddings of Knowledge Graphs.. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)*. AAAI, 1955–1961.

[24] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data.. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, Vol. 11. 809–816.

[25] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2012. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*. ACM, 271–280.

[26] Nicholas Roy and Andrew McCallum. 2001. Toward Optimal Active Learning Through Monte Carlo Estimation of Error Reduction. *Proceedings of the 18th International Conference on Machine Learning (ICML)* (2001), 441–448.

[27] Burr Settles. 2010. Active Learning Literature Survey. *University of Wisconsin, Madison* 52, 55-66 (2010), 11.

[28] Claude Elwood Shannon. 2001. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review* 5, 1 (2001), 3–55.

[29] Yanyao Shen, Hyokun Yun, Zachary C Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep Active Learning for Named Entity Recognition. *arXiv preprint arXiv:1707.05928* (2017).

[30] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems (NIPS)*. 926–934.

[31] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)* 15, 1 (2014), 1929–1958.

[32] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1499–1509.

[33] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*. 2071–2080.

[34] Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin. 2016. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology* (2016).

[35] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 29, 12 (2017), 2724–2743.

[36] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes.. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 14. AAAI, 1112–1119.

[37] Max Welling and Yee W Teh. 2011. Bayesian Learning via Sochastic Gradient Langevin Dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*. 681–688.

[38] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

[39] Jie Yang, Thomas Drake, Andreas Damianou, and Yoelle Maarek. 2018. Leveraging crowdsourcing data for deep active learning - an application: Learning intents in Alexa. In *Proceedings of the 2018 edition of The Web Conference (WWW)*. ACM, 23–32.

[40] Ye Zhang, Matthew Lease, and Byron C Wallace. 2017. Active Discriminative Text Representation Learning.. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI)*. AAAI, 3386–3392.

Choosing to Grow a Graph: Modeling Network Formation as Discrete Choice

Jan Overgoor
Stanford University
Stanford, CA
overgoor@stanford.edu

Austin R. Benson
Cornell University
Ithaca, NY
arb@cs.cornell.edu

Johan Ugander
Stanford University
Stanford, CA
jugander@stanford.edu

ABSTRACT

We provide a framework for modeling social network formation through conditional multinomial logit models from discrete choice and random utility theory, in which each new edge is viewed as a "choice" made by a node to connect to another node, based on (generic) features of the other nodes available to make a connection. This perspective on network formation unifies existing models such as preferential attachment, triadic closure, and node fitness, which are all special cases, and thereby provides a flexible means for conceptualizing, estimating, and comparing models. The lens of discrete choice theory also provides several new tools for analyzing social network formation; for example, the significance of node features can be evaluated in a statistically rigorous manner, and mixtures of existing models can be estimated by adapting known expectation-maximization algorithms. We demonstrate the flexibility of our framework through examples that analyze a number of synthetic and real-world datasets. For example, we provide rigorous methods for estimating preferential attachment models and show how to separate the effects of preferential attachment and triadic closure. Non-parametric estimates of the importance of degree show a highly linear trend, and we expose the importance of looking carefully at nodes with degree zero. Examining the formation of a large citation graph, we find evidence for an increased role of degree when accounting for age.

KEYWORDS

Social networks; network formation; choice models

ACM Reference Format:
Jan Overgoor, Austin R. Benson, and Johan Ugander. 2019. Choosing to Grow a Graph: Modeling Network Formation as Discrete Choice. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313662

1 INTRODUCTION

Understanding how networks form and evolve is an essential component of understanding their structure, which in turn underlies the basis for understanding the broad range of processes that occur *on* networks. Models of social network formation can largely be decomposed into node formation and edge formation. In this work, we argue that edge formation can be effectively modeled as a *choice* made by an actor (or actors) in the network to instantiate a connection to another node. The diverse research on network formation has led to many models and mechanisms of edge formation, including preferential attachment [2], uniform attachment [12], triadic closure [31], random walks [65, 78], homophily [55], copying edges from existing nodes [35, 39], latent space structures [22, 41, 55], inherent node fitness [7, 11], and combinations of all of these [28, 40, 43]. Here, we frame edge formation as a discrete choice process and derive a family of discrete choice models [47, 74] that subsume a wide range of existing models in a unified framework and also naturally opens up a host of powerful extensions.

Discrete choice models are commonly employed in economics, social psychology, and statistics as a way to model how individuals make choices from a slate of discrete alternatives [1]. Typically, the alternatives have associated features, and statistical models of discrete choice make it possible to estimate the relative importance of such features. Such models have been used to answer questions such as how consumers choose goods [67], how people choose where they live [46], how students choose what college to attend [21], and how commuters choose between different modes of transportation [75]. Discrete choice analysis is also used to understand how choices vary depending on the context in which they are framed: in online commerce, this could be how web layouts lead to different purchasing priorities [26]; for choosing colleges, this could be incorporating the effect of the national economy. In this paper, we demonstrate how discrete choice models can similarly help us understand the factors driving social network evolution.

The starting point for the present work is the observation that edge formation events in social networks are naturally viewed as discrete choices. For simplicity, consider a directed graph where edges are formed one by one, where we can think of the formation of a directed edge (i, j) as i "choosing" to connect with j, where the set of alternatives available to i is the set of all other nodes. (While undirected graph models are common in social network analysis, the underlying formation procedure is almost always asymmetric. For example, the Facebook friendship graph is typically modeled as an undirected graph [77], but the friendships are proposed by one of the two nodes in an edge.) The key modeling question is easy to state: why did i choose j? This question has long been the informal subject of network formation modeling and at the same time the exact question that discrete choice models and analysis have been designed to answer. However, up to this point, network formation models have largely been decoupled from discrete choice theory.

In employing discrete choice analysis, we focus on the conditional multinomial logit model, commonly called the conditional logit model for short, which is a foundational workhorse of discrete

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313662

choice modeling. The model belongs to the family of random utility models, where choices are interpretable as those of a rational actor selecting the alternative with the largest "utility" sampled from random variables that decompose into the inherent utility of the alternative and a noise term. With the conditional logit model, we can use existing optimization routines to estimate model parameters and existing statistical methods to asses the uncertainty of the estimates. Discrete choice models can also easily restrict the set of available alternatives, where it might not be reasonable to assume that the *entire* set of nodes is available for friendship. For example, sometimes only "friends of friends" are considered [24, 28, 40].

In this paper, we first show that many popular network formation mechanisms can be rewritten as conditional logit models, including preferential attachment, uniform attachment, node fitness, latent space models, and models of homophily. However, the real power of discrete choice models for social network analysis is the ability to combine different features (e.g., node degree and node age), as well as different mechanisms (e.g., triadic closure and preferential attachment) and estimate their relative roles. Social networks are enormously varied in their structure [27], but existing methods often do a poor job at modeling this diversity. Thus, beyond unifying the network formation and discrete choice literature, we also develop several new tools for social network analysis. For example, we show how to estimate models to distinguish the effects of preferential attachment and triadic closure. We demonstrate these tools by analyzing the formation of the Flickr social network and the formation of a citation network. We find on Flickr that accounting for triadic closure greatly reduces the estimated role of degree in choosing who to connect to, and that nodes with degree zero have a remarkably high utility. Our estimates of preferential attachment in the citation network are similar to those observed in prior studies. When accounting for the age of a paper, we find evidence for linear preferential attachment. However, for a fixed degree, we find that age is *negatively* correlated with the likelihood of a new citation (i.e., older papers are less likely to be cited).

The key assumption underlying our framework is that the available data actually captures edge formation events (either through edge timestamps or other sequential information). In contrast, many existing approaches to understanding network formation focus on observing only the structural properties of a network at a single point of observation, e.g., its degree distribution, and initiating a deductive process to try and understand how variations in edge formation would lead to different outcomes [2, 7, 28, 43]. This approach leads to tidy analyses and easy-to-characterize asymptotic properties, but model selection in this context is strongly dependent on what properties are compared. Different underlying formation processes can lead to graphs with indistinguishable properties. For example, many different formation processes result in the same heavy-tailed degree distributions [52]. Thus, when "fitting" outcome measurements in this way, one has to know (or posit), e.g., the relative rates of node formation and edge formation. However, when temporal or sequential data is available [25, 56], our framework overcomes these limitations by incorporating this structure.

Additional related work. There is a strong connection between our work and work on link prediction and missing data methods using network features to predict edges [15, 42]. A network formation model implicitly makes claims about what edges are most likely to form next, and thus can be evaluated by the same metrics as link prediction algorithms [44]. We use predictive accuracy as a measure of goodness of fit, but our primary concern is interpretability of the model and estimates, which is one of the advantages of the conditional logit model.

In sociology, stochastic actor-oriented models (SAOMs) employ a similar logit choice [69, 70]; however, these models are targeted towards data collected as a few snapshots rather than edge-by-edge formation. SAOMs also model the rate at which nodes form new relationships, whereas we condition on the node initiating the new edge, providing better estimates of model parameters. There are also sociological models such as relational event models [10] and dynamic network actor models [71] that use fine-grained temporal information, yet these also do not condition on the initiator node as we do. While these sociological models can incorporate notions of network formation (e.g., preferential attachment), our conditional logit framework actually cleanly *subsumes* a wide range of models as special cases.

Finally, estimating the parameters that drive edge formation is different from identifying the factors that could have lead to the observed graph. The latter question is often pursued with so-called exponential random graph models (ERGMs) [63, 79, 81]. However, these models do not consider individual edge events, are hard to estimate, and have known pathologies [13, 66].

2 DISCRETE CHOICE AND EDGE FORMATION

We now develop network formation through the lens of discrete choice. Throughout this paper, we assume that the networks are directed. Again, while undirected graphs are common in social network analysis, the actual edge formation process often has directed initiation. In the common setting of "growing graphs," nodes arrive one at a time and form edges when arriving in a network. In these cases, the newly arriving node is considered to be the node initiating the connection; such analysis is standard with, e.g., classical preferential attachment models [2].

When modeling the directed formation of an edge (i, j), two processes need to be distinguished, roughly corresponding to the questions "who is i?" (the chooser) and "who is j?" (the chosen). In this paper, we focus on understanding the latter, i.e., the formation of (i, j) as the selection of j conditional on knowing that i is ready to form an edge. Thus, our discrete choice models of edge formation can be readily estimated from data that implicitly or explicitly contains a record of initiating i nodes and used for subsequent analysis, as we show in Sections 3 and 4. Beyond the scope of this work, our model of "j conditional on i" can be paired with a model of "initiations by i" for a full generative model of network formation.

2.1 Background on discrete choice models

We now review discrete choice models generically, which we will then translate to the context of edge formation in Section 2.2. Consider a universe of alternatives $\mathcal{X}$ and a dataset consisting of n different choices, indexed by $k = 1, \ldots, n$. Each choice (j, C) consists of a choice set $C \subseteq \mathcal{X}$ and a chosen item $j \in C$. The elements in C are mutually exclusive choice alternatives, and exactly one

element from C is chosen. We consider each element $j \in C$ to be represented by a vector of features x_j (for example, in our analysis in Section 2.2, x_j will be the feature vector of a node in the graph, and C will represent a set of nodes). We let $\mathcal{D} = \{(j_k, C_k)\}_{k=1}^n$ denote the choice data.

A broad family of discrete choice models is the family of random utility models (RUMs), of which the conditional logit is an important special case. Each alternative j has some inherent utility to the agent i making the choice; with the conditional logit, we model this utility as a linear function of j's features x_j:

$$u_{i,j} = \theta^T x_j,$$

for some (latent) parameter vector θ that is fixed across individuals i. Random utility models assume that agents make "rational" choices by maximizing random utilities centered on these inherent utilities. More formally, the utility $U_{i,j}$ observed by the actor is given by $U_{i,j} = u_{i,j} + \varepsilon_{i,j}$, where $\varepsilon_{i,j}$ is a noise term. The probability $P_i(j, C)$ of i choosing option j from the choice set C is

$$P_i(j, C) = \Pr(j = \arg\max_{\ell \in C} U_{i,\ell}).$$

When the $\varepsilon_{i,j}$ are i.i.d. standard Gumbel, the probability of choosing each alternative is proportional to the exponentiated inherent utility [1]:

$$P_i(j, C) = \frac{\exp \theta^T x_j}{\sum_{\ell \in C} \exp \theta^T x_\ell}. \tag{1}$$

The above model is the conditional multinomial logit model of discrete choice, though it is often referred to simply as the conditional logit or multinomial logit (MNL) model. If the noise terms $\varepsilon_{i,j}$ are distributed i.i.d. Normal, then the model is the independent multinomial probit. In probit models, the choice probabilities are no longer proportional to utilities as in Equation (1). The Gumbel assumption is common in discrete choice theory and will facilitate our connections to a variety of network formation mechanisms. There are more complex random utility models that can impose dependence between noise terms [74] and there also is a growing literature of flexible choice models [4, 60, 76] designed to model context effects and other varied violations of the *independence of irrelevant alternatives* [45] (a storied axiom satisfied by the conditional multinomial logit model). We leave the relationships of these models to network formation as avenues for future research.

2.2 Edge formation as discrete choice

With the above formalisms in place, we now develop network formation from a discrete choice perspective. We begin by showing how several well-known models can be conveniently expressed as conditional logit models, with a summary given in Table 1. All models are designed to grow simple graphs (i.e., without multi-edges), and the choice set C excludes any nodes to which the chooser i is already connected. Every item is represented by its features that, importantly, can evolve over time. The features $x_{j,t}$ of node j at time t are thus always time-indexed, but we often suppress the t to reduce notational clutter.

Preferential attachment. We start with the generalized Barabási-Albert model [2, 8, 36], also known as the generalized Price model [59], one of the most studied models in the network formation literature. It is typically stated as a growth model of a time-evolving graph $G_t = (V_t, E_t)$, $t = 1, 2, 3, \ldots$, and when a new node arrives it connects to m distinct existing nodes j with a probability proportional to a power of their degree $d_{j,t}$ at time t,

$$P(j, V_t) = \frac{d_{j,t}^\alpha}{\sum_{\ell \in V_t} d_{\ell,t}^\alpha}. \tag{2}$$

The exponent parameter α controls the relative importance of degree [36]. The case where $\alpha = 1$ is called linear preferential attachment, and produces networks that can mimic a range of structural properties observed in empirical networks. If we represent each potential neighbor j with the time-indexed one-dimensional "feature vector" $x_{j,t} = \log d_{j,t}$ and employ a conditional logit model as in Equation (1), we obtain a utility of j for i at time t of $u_{i,j,t} = \theta \log d_{j,t}$. Here the choice model parameter θ plays the exact role of α, since $e^{\theta \log d_{j,t}} = d_{j,t}^\theta$.

Given a growing network G_t, we can construct a choice dataset $\mathcal{D}$ from this network by extracting the node j_t, node sets V_t, and degree sequence $(d_{1,t}, \ldots, d_{|V_t|,t})$ at each time-step. The preferential attachment model has only one parameter, $\theta = \alpha$. The log-likelihood for that parameter given a dataset is then:

$$\begin{aligned} l(\alpha; \mathcal{D}) &= \sum_{(j,C) \in \mathcal{D}} \log \frac{\exp(\alpha \log d_j)}{\sum_{\ell \in C} \exp(\alpha \log d_\ell)} \\ &= \sum_{(j,C) \in \mathcal{D}} \left(\alpha \log d_j - \log \sum_{\ell \in C} \exp(\alpha \log d_\ell) \right). \end{aligned}$$

We've suppressed the time-index t from the features $\log d_\ell$ to reduce clutter, but emphasize that d_ℓ is the degree at the time of the choice.

Non-parametric preferential attachment. The above model assumes an attachment kernel of a particular parametric form. From a discrete choice perspective, one can also estimate the role of degree in edge formation non-parametrically by estimating a coefficient θ_k for each degree $k = 0, \ldots, n-1$ individually. This approach has the added benefit of being able to assign positive probability to choosing nodes with degree zero. Under this model, the log-likelihood of the parameters $\theta = (\theta_0, \ldots, \theta_{n-1})$ given the dataset is:

$$\begin{aligned} l(\theta; \mathcal{D}) &= \sum_{(j,C) \in \mathcal{D}} \log \frac{\exp \theta_{d_j}}{\sum_{\ell \in C} \exp \theta_{d_\ell}} \\ &= \sum_{(j,C) \in \mathcal{D}} \left(\theta_{d_j} - \log \sum_{\ell \in C} \exp \theta_{d_\ell} \right). \end{aligned}$$

Again we've suppressed time-indexing to simplify the presentation. Pham et al. [58] previously described a version of the above likelihood as a means of measuring the attachment kernel using maximum likelihood, albeit without making the connection to discrete choice.

Uniform attachment. A simple edge formation model is to sample a new neighbor uniformly at random from all nodes [12]. There are no parameters in this model, but we can still write down the likelihood of the model given a dataset, which will be useful when we later combine this model with others within a mixture model:

$$l(\mathcal{D}) = \sum_{(j,C) \in \mathcal{D}} \log \frac{\exp(1)}{\sum_{\ell \in C} \exp(1)} = \sum_{(j,C) \in \mathcal{D}} -\log |C|.$$

Table 1: Network formation models framed as utility functions for a conditional logit. Where appropriate, we use the traditional notation for the parameters of each process.

Process	$u_{i,j}$	C
Uniform attachment [12]	1	V
Preferential attachment [2, 36]	$\alpha \log d_j$	V
Non-parametric PA [54, 58, 62]	θ_{d_j}	V
Triadic closure [61]	1	$\{j : FoF_{i,j}\}$
FoF attachment [31, 65, 78]	$\alpha \log \eta_{i,j}$	V
PA, FoFs only	$\alpha \log d_j$	$\{j : FoF_{i,j}\}$
Individual node fitness [11]	θ_j	V
PA with fitness [6, 53]	$\alpha \log d_j + \theta_j$	V
Latent space [22, 41, 55]	$\beta \cdot d(i,j)$	V
Stochastic block model [33]	ω_{g_i,g_j}	V
Homophily [48]	$h \cdot \mathbb{1}\{g_i = g_j\}$	V

Triadic closure. A variant of uniform attachment is for i to attach to new neighbors uniformly at random from the set of their friends-of-friends, as opposed to the set of all nodes. This process effectively models triadic closure [61]. It has the same simple functional form of the uniform model, but now the *choice set* C varies with each choice, namely, the choice set is restricted to be only the friends of friends of node i (the chooser) to which i is not already connected. This change in choice set can also be achieved by assuming the utility of j to i at time t is $u_{i,j,t} = \log(\mathbb{1}\{FoF_{i,j,t}\})$, where $\mathbb{1}\{FoF_{i,j,t}\}$ is a boolean indicating whether i and j are friends of friends at time t, and then letting the choice set revert to the full node set.

An additional model that naturally combines the ideas of preferential attachment and befriending friends-of-friends takes the *number* of friends in common between i and j as a feature. We could define this feature as $\eta_{i,j,t} = |\{k : e_{i,k,t} \wedge e_{k,j,t}\}|$, where $e_{i,k,t}$ indicates whether there is an edge between i and k at time t. The corresponding utility would be $u_{i,j,t} = \alpha \log \eta_{i,j,t}$. This model is similar (but not equivalent) to random walk-based formation models [31, 65, 78], which emphasize formation within a local neighborhood.

Node fitness. Another line of formation models that is subsumed by the discrete choice framework are those involving fitness. In this work, nodes choose to connect to others based on some intrinsic latent fitness score. Certain distributions of fitness values lead to a scale-free degree distribution [11], providing an alternative explanation to preferential attachment for modeling such degree distributions. We can express the node fitness model by a conditional logit model with separate fixed effect θ_j for each node j (so the feature of a node is an indicator vector of its identity). The likelihood of the fitness parameters θ given the data is then:

$$l(\theta; \mathcal{D}) = \sum_{(j,C)\in\mathcal{D}} \log \frac{\exp \theta_j}{\sum_{\ell \in C} \exp \theta_\ell}$$
$$= \sum_{(j,C)\in\mathcal{D}} \left(\theta_j - \log \sum_{\ell \in C} \exp \theta_\ell \right).$$

This formation model is equivalent to the classic Bradley-Terry-Luce model of discrete choice for estimating the quality of alternatives [45]. Alternatively, one could replace the individual fixed effects with surrogate features of node fitness such as an auxiliary measure of gregariousness (in the case of social networks), or the impact factor of a paper's journal (in the case of citations networks).

A related model proposes selection probabilities proportional to the product of node fitness and degree [6, 53]. This model can be written as a conditional logit model with $u_{i,j,t} = \alpha \log d_{j,t} + \theta_j$.

Latent space models. Another class of network formation models postulates the existence of a latent space that drives connections between nodes. Examples of latent spaces include Euclidean space [22], hyperbolic space [37], a tree [41], a circle [55], or a set of discrete classes [23]. While the conditional logit model in the form that we describe it does not facilitate finding the best-fitting latent space assignment to explain the data, it can be used to estimate the relative importance of a known latent space given a distance function $d(i,j)$. As one example from the family of latent space models, in the community-guided attachment (CGA) model [41] all nodes have a distance derived from the height $h(i,j)$ of common parents in a latent tree structure situating all nodes i and j. Given this tree as known, a node connects to another proportionally to $c^{-h(i,j)}$ for some scalar $c > 0$. As a conditional logit model, the corresponding utility function is $u_{i,j} = -h(i,j) \cdot \log(c)$. The parameter vector $\theta = \log c$ can be retrieved by fitting a conditional logit with a known $h(i,j)$ as the only variable and transforming the estimated parameter with $c = \exp(\theta)$. Assuming that the latent space representation is given is a strong assumption, and fitting such a model while estimating the latent space representation (e.g. as done by Hoff et al. [22] in Euclidean space) is much more difficult.

Additional models. Conditional logit models are very flexible and can deal with multiple features and interactions between them. Any number of features can be added, including node covariates and structural features like a node's clustering coefficient [3] or age [12, 40]. Conditional logit models can also be used to investigate the role of homophily [48] in edge formation, by adding a binary feature indicating whether nodes i and j are part of the same class.

Table 1 summarizes how several network formation models fit within the discrete choice framework via their corresponding utility functions and choice sets. A major advantage of this framework is that different features can easily be combined into a single model and jointly estimated. Or, when suitable, one can employ a mixture of conditional logit models, as we show in the next section.

2.3 Combining modes using Mixed Logit

So far we have written a range of existing and new edge formation models as conditional logit models, a specific type of discrete choice model. But several existing edge formation models that do not fit neatly into the conditional logit framework, meanwhile, align exactly with the use of mixture models in discrete choice modeling. Following our success formulating edge formation models as conditional logit models, in this subsection we develop mixed conditional logit formulations of several additional models.

A common proposal to make network formation models more flexible is to augment an existing model by allowing nodes to pick neighbors uniformly at random with some probability $1 - p$, while running the ordinary model with probability p [17, 35, 39, 43]. This augmentation increases flexibility because it enables the model to explain edge events that may otherwise have probability zero.

Within discrete choice, this approach is precisely a mixed logit model where one of the mixture modes is uniform attachment.

While the conditional logit estimates a single parameter vector representing average preferences as shared by all agents, the *mixed logit* model is often used to account for differences in preferences across various types of agents. In its most general form, the mixed logit is expressed using a probability distribution f over different instantiations of the parameter vector θ:

$$P_i(j, C) = \int \frac{\exp \theta^T x_j}{\sum_{l \in C} \exp \theta^T x_l} f(\theta)\, d\theta.$$

In this work, we will only consider discrete mixtures of M logits, also called a latent class model [32]:

$$P_i(j, C) = \sum_{m=1}^{M} \pi_m \frac{\exp \theta_m^T x_j}{\sum_{l \in C} \exp \theta_m^T x_l},$$

where $\sum_{m=1}^{M} \pi_m = 1$ and the weights $\pi_1, \ldots, \pi_M$ model the relative prevalence of each mode.

Copy model. The copy model is a classic formation process that can be written as a mixed logit with two modes. In the first mode, new edges connect proportional to degree with probability p, while in the second mode they connect uniformly at random with probability $1 - p$ [17, 43]. As a conditional logit model, the utilities of the two modes are $u_x^{(1)} = \log d_x$ and $u_x^{(2)} = 1$, respectively, and the class probabilities are $(\pi_1, \pi_2) = (p, 1 - p)$. (This is a special case of the original copy model where d edges are copied from a sampled vertex [39]; the model here is when $d = 1$, which is often used for analysis [19].) The connection between relaxations of preferential attachment and mixture models was also recently observed by Medina et al. [49].

Local search model. Another example of a model with multiple modes is the Jackson-Rogers model of edge formation as a mixture of uniform attachment and triadic closure [24, 28]. The original model is based on a relative rate r^* between edges forming at random and edges formed locally. It also has edges form based on respective acceptance probabilities. We describe a simplified version of this model, which we'll call the local search model, where edges connect to nodes selected uniformly at random from the full node set with probability r and uniformly at random from the set of friends-of-friends with probability $1 - r$.[1] We can represent this simplified process with a two-mode mixed logit model. In this case the mixture parameters are $(\pi_1, \pi_2) = (r, 1 - r)$ and both modes have the same utility function $u_x = 1$ but their *choice sets* differ so that the second mode only considers friends-of-friends.[2]

Table 2 overviews the mixture model formulations described above, as well as a new model—the (r, p)-model—that we use in Section 4.2 to analyze preferential attachment effects.

3 ESTIMATION AND INFERENCE

To learn a discrete choice model of network formation from data, we assume that we have access to a sequence of directed edges, in chronological order. This sequence of edges needs to be recast as choice data in order to fit a choice model. For every formed edge (i, j), we create a data point consisting of the choice j, the choice set of candidates nodes at the time, and the features of each candidate node at the time.

Table 2: Network formation models framed as mixed logits. Each mixture component is a mode. FoF refers to friend-of-friending, also called local search or triadic closure. We define a new (r, p)-model as a natural generalization of prior ideas once we put network formation models in the language of discrete choice.

Process	Modes
Copy model [35]	Uniform, PA
Node types [38]	New node, PA, none
Local search [24, 28]	Uniform, Uniform FoF
(r, p)-model	Uniform, PA, Uniform FoF, PA FoF

Given a data set and a conditional logit model, one can write out the log-likelihood, as shown in Section 2.2. For any conditional logit model with a linear utility $u_{i,j} = \theta^T x_j$, the likelihood function is convex with respect to the variables θ and can be efficiently maximized using standard gradient-based optimization (e.g., BFGS). The functional form of the logit leads to straightforward gradients. For example, for preferential attachment, the gradient is

$$\frac{\partial}{\partial \alpha} l(\alpha; \mathcal{D}) = \sum_{(x, C) \in \mathcal{D}} \left(\log d_x - \frac{\sum_{y \in C} \log d_y \cdot \exp(\alpha \log d_y)}{\sum_{y \in C} \exp(\alpha \log d_y)} \right),$$

where the time-dependence of the features (degrees) have been suppressed to reduce clutter. Gradients for the other choice models in Section 2.2 are omitted but straightforward.

One advantage of likelihood-based model fitting is that we can compute standard errors and confidence intervals of the parameters. In particular, the standard errors can be computed with $\sqrt{\mathbf{H}^{-1}}$ [74], where $\mathbf{H}$ is the Hessian matrix of second derivatives of the log-likelihood at the parameters.

Mixture models and expectation-maximization. For mixed conditional logit models, the log-likelihood is no longer convex in general, making optimization more difficult. To maximize the likelihood of mixed models we turn to expectation maximization (EM) techniques [18, 73]. We briefly summarize the procedure described in Train's book [74, Chapter 14.3.2]. Assume that we have a model with M modes (i.e., mixture components), where every mode starts with initial parameter values $\vec{\theta}^m$ (usually initiated at 1). Choices $(x_k, C_k) \in \mathcal{D}$ are again indexed with k, so that $k \in \{1, \ldots, n\}$ and $n = |\mathcal{D}|$. The EM algorithm runs through the following steps:

(1) Initiate class probabilities uniformly with $\pi_m = 1/M$ and initial class responsibilities $\gamma_k^m = 1/M$ for each data point.
(2) For every data point k and every mode m, compute the class responsibility given by the relative individual likelihood:

$$\gamma_k^m = \frac{\pi_m \cdot \mathcal{L}^m(\theta^m; (x_k, C_k))}{\sum_{\ell=1}^{M} \pi_\ell \cdot \mathcal{L}^\ell(\theta^\ell; (x_k, C_k))}.$$

(3) For every mode m, update the total class probability with $\pi_m = \frac{1}{N} \sum_{k=1}^{N} \gamma_k^m$.

[1] Since the r^* parameter in the original presentation is actually the *rate* of uniform attachment, we can relate it to our r through $r = \frac{r^*}{1+r^*}$. For example, if the rate between random and friend-of-friend edges is one to one ($r^* = 1$), then $r = 0.5$.
[2] A model with a restricted choice set, for example to only friends-of-friends, gives a likelihood of zero to choices outside the choice set.

(4) For every mode m, update the parameters $\vec{\theta}^m$ using standard optimization for fitting a single model, weighing each choice set with its class responsibility γ_k^m.
(5) Repeat steps 2–4 until some convergence or stopping criteria.

The total likelihood of the parameters and class probabilities is:

$$l(\theta; \mathcal{D}) = \sum_{m=1}^{M} l^m(\theta^m; \pi_m; \mathcal{D}) = \sum_{m=1}^{M} \sum_{k=1}^{N} \log \mathcal{L}^m(\theta^m; (x_k, C_k)) \cdot \pi_m$$

We monitor the convergence of the iterative procedure using the change in this total likelihood between iterations.

Even though EM is theoretically an efficient estimator [82], there are cases when alternatives are appropriate. For example, if one has reasonable bounds or priors on the parameter values, then direct likelihood maximization could be used, and if the search space is low-dimensional, a grid search might be appropriate. Recent theoretical work has also developed algorithms for learning mixtures of two multinomial logit modes with theoretical guarantees assuming a separation between the modes [14].

Negative sampling. Every time an edge is formed by some node i, each node not yet connected to i is a candidate choice. For large sparse graphs, the full choice set of all nodes can become large and the gradients of the log-likelihood expensive to compute. To speed up this computation, s negative/non-chosen examples can be *sampled* uniformly at random to create a (random) reduced dataset with smaller choice sets. For each choice (j, C), one forms a smaller random choice set out of the positive choice and the negative samples, $\tilde{C} \subset C$ with $|\tilde{C}| = s + 1$, and replaces the original choice data with $(j, \tilde{C})$. As long as the negative examples are sampled uniformly at random, parameter estimates on a dataset with negatively sampled choice sets are unbiased and consistent for the estimates on the on the full set [29, 46, 74]. Practically, there is a trade-off between feature computation and storage on the one hand, and the ability to estimate coefficients for rare features on the other.

Typical likelihood surface. In Figure 1 we show the representative likelihood surface of a copy model to illustrate its properties. We generated a synthetic graph on $n = 10,000$ nodes according to the copy model with $m = 4$ edges per node and degree-attachment probability $\pi_1 = 0.5$. We fit a two-mode mixed logit model to this data with $u_j^{(1)} = \alpha \log d_{j,t}$ and $u_j^{(2)} = 1$. We use $s = 10$ negative samples. There are two free parameters in this model: the degree exponent α and the mixture probability π_1. We plot the log-likelihood across a reasonable range of values to show that surface is generally well behaved. We see that it is hard to distinguish between data generated under a copy model ($\alpha = 1$) with probability $\pi_1 = 0.5$ from data generated from no-mixture ($\pi_1 = 0$) preferential attachment with $\alpha = 0.5$, and there is a general trade-off between the exponent α and the mixture probability π_1.

Model comparison and the likelihood-ratio test. Another advantage of our discrete choice framework is that we can employ standard statistical methods for model selection. Specifically, when one model is a special case of another, their relative quality can be compared using the likelihood ratio test. In the case of the conditional logit, a model with additional features can be compared to one without them because the latter is a special case of the former with the coefficients of the additional features being set to 0. Or, in the case of the mixed logit, one can define a model with multiple modes and manually set some of their class probabilities to zero.

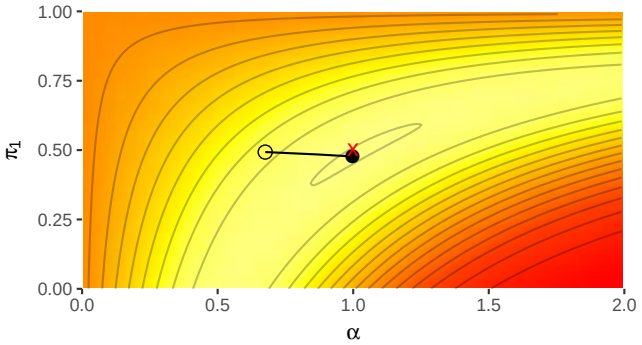


Figure 1: The log-likelihood surface of the copy model for a graph of $n = 10,000$ nodes generated with $\pi_1 = 0.5$ (marked as x). The line tracks the iterations of the EM algorithm, from open to closed dot. There is a trade-off between the degree coefficient α and the mixture probability π_1, but there are large regions with similar likelihoods.

As a concrete example, suppose we wanted to know whether including the age of a node in a preferential attachment model results in a statistically significantly better model. To do so, we would first estimate the parameters θ_1 of the more complex model, $u_j^{(1)} = \theta_{1,1} \log(d_j) + \theta_{1,2} \log(\text{age})$. We would then estimate the parameters θ_0 of the simpler model $u_j^{(0)} = \theta_{0,1} \log(d_j)$. Let $\mathcal{L}_1$ and $\mathcal{L}_0$ be the likelihoods of the two models with parameters $\hat{\theta}_1$ and $\hat{\theta}_0$. We can compute the likelihood ratio $\lambda = \mathcal{L}_0 / \mathcal{L}_1$. Under the null hypothesis of the simpler model, with some regularity conditions, $-2 \log \lambda$ is asymptotically distributed χ_1^2 (χ_k^2 where k is the number of additional degrees of freedom in the more complex model) [80], a standard test in the finite regime [74, Chapter 3.8.2].

4 APPLICATIONS

We now demonstrate how to use our conditional logit framework to analyze network formation processes. We first consider synthetic data and show how our tools can be used to better analyze preferential attachment mechanisms. We then analyze two empirical datasets that demonstrate how to integrate different structural features of the network or integrate node covariates. In both cases, our framework provides novel insights into the network formation processes. We provide code for processing data (converting edge lists to choice data) and for model fitting (with negative sampling), available here: https://github.com/janovergoor/choose2grow/.

4.1 Measuring preferential attachment

The question of whether and when preferential attachment is an important driver of network formation is widely debated [2, 3, 9, 11, 12, 24, 28, 54, 54, 65, 78]. Most prior research focuses on estimating the shape of the attachment kernel by comparing the degree of chosen nodes to the distribution of available degrees [30, 54, 62]. However, recent work by Pham et al. shows that previous measures are biased [58]. In particular, the bias comes from the assumption

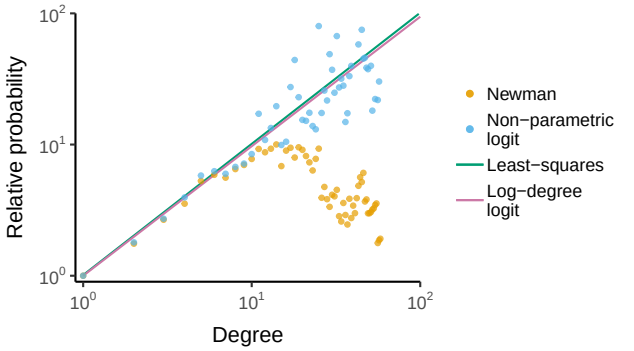


Figure 2: Attachment kernel fits for a synthetic preferential attachment graph. The Newman measure computes the relative likelihood of selecting a node of that degree, as compared to the likelihood of selecting the lowest degree, but it is biased for higher degrees. The non-parametric logit is consistent but noisy for higher degrees.

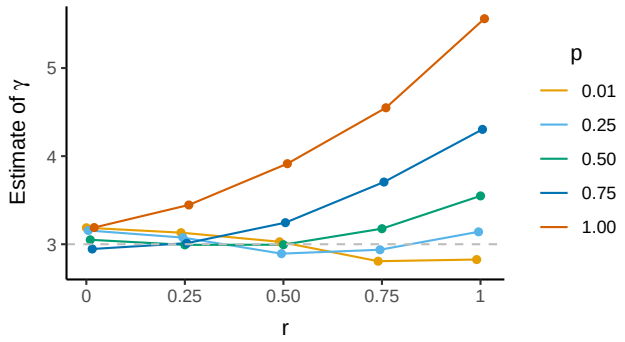


Figure 3: Estimating the power-law exponent γ from the degree distributions of graphs formed under the (r, p)-model with $n = 20,000$ nodes. Under the local search model with significant triadic closure ($p = 1$, small r), the exponent looks like it would under the copy model ($p \to 0$, $r = 1$).

that the distribution of available nodes of varying degrees is constant throughout the formation process, but this distribution clearly changes as the network grows.

To estimate the exponent α of an attachment kernel, Pham et al. propose fitting something akin to a conditional logit with a separate coefficient for each degree, and then estimating α via a weighted least squares fit over the degree coefficients [58]. Compared to this method, fitting a log-degree logit directly is much simpler. In fact, it is the maximum likelihood estimator for α, and thus consistent and efficient.

To illustrate, we generate a graph with pure preferential attachment ($n = 2,000$, $m = 1$ edges per node, $\alpha = 1$) and estimate the attachment kernel by the methods of Newman [54] and Pham et al. [58]. The maximum degree of this graph was 102, and the results of the different estimation procedures are shown in Figure 2. The non-parametric estimates are similar for lower degrees, but for higher degrees the Newman measure incorrectly drops, illustrating the bias that Pham et al. have previously documented. Fitting α directly using a log-degree conditional logit gives an estimate of $\hat{\alpha} = 0.987$. The Pham et al. least squares fit, $\hat{\alpha}_{LS} = 1.012$, is close to the MLE but may deviate considerably in more difficult instances.

4.2 Disentangling preferential attachment from triadic closure

Many models exhibit similar outcomes to preferential attachment [11, 24, 28, 36, 52, 78], but there are few principled ways to rigorously test the relative validity of these models. In this section, we show how to use the discrete choice framework to estimate the relative importance of preferential attachment while accounting for other dynamics. To this end, we generate data according to a known generative process and fit various (possibly mis-specified) formation models. Our generative process is a hybrid between the copy model of preferential attachment (i.e., choose nodes proportional to degree) and the Jackson-Rogers local search model (i.e., connecting to friends-of-friends). The process, which we call the (r, p)-model, is parametrized by $r \in (0, 1]$ and $p \in (0, 1]$. When a new edge is formed, with probability p it is formed uniformly at random and with probability $1 - p$ it is formed with linear preferential attachment ($\alpha = 1$). Meanwhile, the *choice set* is determined by the second parameter r: with probability r, the choice set is all nodes not yet connected to i, while with probability $1 - r$, the choice set is limited to available friends-of-friends of i. With $r = 1$ this model reduces to the copy-model and with $p = 1$ it reduces to the simplified local search model; the (r, p)-model thus subsumes two popular models in a single, simple discrete choice framework. For a growth process on directed graphs, it is necessary that $p > 0$ and $r > 0$, otherwise new nodes will never be selected.

With this general model, we investigate how estimating parameters of one of the more specific models goes awry when the true data generating process in fact comes from an instance of the more general model. For a range of values of p and r, we generated graphs using the following growth process. New nodes arrive, each creating $m = 4$ edges. For every edge, we sample the mode of the model (according to r and p) independently. If an edge is supposed to be a friend-of-friend edge, but no friends-of-friends are available (for example, i's first edge), then the process reverts to uniformly random formation across the full node set.[3] Sweeping through combination of p and r parameter values, for each set of parameters we generated 10 *undirected* graphs with $n = 20,000$ nodes each.

Degree distributions. The local search and copy models both produce graphs with power-law degree distributions. Therefore, fitting a mis-specified model on a degree distribution can lead to misleading results. To illustrate, we fit a power-law distribution $p(x) \propto x^{-\gamma}$ to the degree distribution of graphs generated from (r, p)-models using maximum likelihood estimation [16], with estimates for γ in Figure 3. In theory, an undirected graph formed with the copy model process with probability parameter p leads to a degree distribution with power law exponent $\gamma = (3 - p)/(1 - p)$ [8, 52] (for directed graphs, $\gamma = (2 - p)/(1 - p)$). As p increases, the degree distribution looks more like a random graph without preferential attachment. However, as r goes down (increasing the relative role of friend-of-friends), the parameter estimate looks like the estimates for the copy model, even when $p = 1$.

[3] This creates a slight bias towards uniform at random modes. This reversion to uniform attachment happens for every first edge with probability $1 - r$.

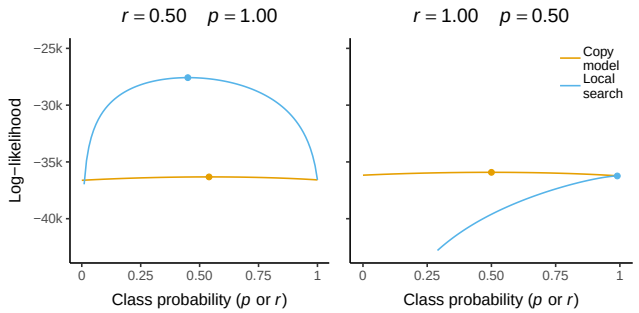


Figure 4: The log-likelihood of varying the class probabilities of the copy model ($r = 1$, p free) or the local search model (r free, $p = 1$) for two different synthetic graphs. In both cases the true model is the most likely. On the left we see a large difference in the log-likelihood between optima, while on the right we see a smaller difference. In both cases a likelihood ratio test is highly significant (P-values $< 10^{-16}$).

To summarize, it is not recommended to estimate a formation model from an observed degree distribution. The parameter estimates are sensitive to small deviations in the generative process.

Discrete choice modeling. Beyond degree distributions, in Figure 4 we look at how the two subsumed models (the copy model and the Jackson-Rogers local search model) fare when estimated from formation data generated by the (r, p)-model. We look at two cases. As a first case, we generate graphs with $r = 0.5$ and $p = 1$, so half the edges are formed to friends-of-friends with no utility from degree. The likelihood under a local search model (r free, $p = 1$) as a mixed logit is maximized at $r = 0.45$, while for the copy model ($r = 1$, p free) it is maximized at $p = 0.54$. The former is a much better fit than the latter (P-value $< 10^{-16}$), and the copy model erroneously thinks that preferential attachment is driving 45% of the edges. As a second case, we look at a graph generated with $r = 1$ and $p = 0.5$, so half the edges are due to preferential attachment, and friend-of-friending plays no role. In this case, both models are correctly maximized at their relative values. Again, the correct model has a higher likelihood (P-value $< 10^{-16}$).

4.3 Choosing to follow on Flickr

We now apply our framework to examine a real-world network formation dataset capturing the growth of the Flickr social network. We find that incorporating a Friend-of-Friend feature beyond preferential attachment and link-reciprocation features substantially improves both likelihood and test accuracy and furthermore that the inclusion of this feature significantly reduces preference for degree-based attachment. However, omitting preferential attachment entirely leads to a worse model. We also find a preference for nodes with zero degree over low degree nodes. This hints that such nodes play a special role in the network formation process, even though they would be ignored in preferential attachment models.

Data. We use a scrape of the Flickr social network collected daily between October 2006 and May 2007 [50, 51]. Users of Flickr can choose to follow other users and the "following" (but not the "followed by") connections are publicly accessible. The data was gathered using a breadth-first search crawl, which means that only the connected components reachable from the seed profiles are represented in the data. Since a full crawl was performed daily, the timing of new edges can be identified at the granularity of a day. The graph contains 3.2 million nodes and 33.1 million edges.

As described in the original papers, this data is consistent with both preferential attachment, as inferred from the in-degree distribution, and local search, as inferred from the over-representation of edges to nodes that are close to the linking node [50]. Fitting a power law to the distribution of *in*-degrees gives $\hat{\gamma} = 1.741$, which would indicate super-linear preferential attachment. We can test the relative importance of triadic closure by fitting a Jackson-Rogers model using the degree distribution matching procedure described in [28]. This results in $\hat{r} = 0.252$, estimating that three out of four edges are formed through triadic closure.

Table 3: Conditional logit model fits for Flickr data. Standard errors of the estimates are given in parentheses. Evaluation statistics are computed over 2,000 sampled examples excluded from the training data.

	Model			
	#1	#2	#3	#4
log Followers	1.149*		0.715*	0.536*
	(0.007)		(0.009)	(0.010)
Has degree	-0.580*		-0.631*	-1.745*
	(0.202)		(0.190)	(0.234)
Reciprocal	8.419*	8.347*	8.197*	7.903*
	(0.220)	(0.220)	(0.240)	(0.244)
Is FoF		6.12*	3.955*	
		(0.045)	(0.050)	
2 Hops				6.290*
				(0.190)
3 Hops				2.851*
				(0.185)
4 Hops				0.583*
				(0.189)
5 Hops				-0.585*
				(0.218)
≥ 6 Hops				-1.122*
				(0.266)
Observations	20,000	20,000	20,000	20,000
Log-likelihood	-16,448	-14,685	-10,728	-9,789
Test accuracy	0.758	0.722	0.853	0.855
Note:				*p<0.01

Discrete choice analysis. We fit a series of conditional logit models to further investigate the network formation process. We isolated a sample of 20,000 edge formation events occurring around the same date,[4] to avoid time heterogeneity affecting the estimates. We fit several models, displayed in Table 3. Not-chosen alternatives are negatively sampled with $s = 24$. We log-transform in-degree

[4]We enumerated edges starting November 5, 2006 and included new edges with probability 0.01 until reaching the desired sample size. We excluded edge events originating from nodes seen for the first time in a given day (the timing of these edge events are uncertain due to the original data collection process). The same analysis starting on March 3, 2007 led to virtually identical results.

(representing the number of followers), but in order to account for nodes with degree zero, we add a "has degree" feature for having a positive degree and use a modified version of log that returns 0 for input 0.[5] In the first column, we fit a model using just these two degree-related features, and a reciprocity feature capturing whether the target node is already following the chooser. Reciprocity is a common phenomenon, with 60% of edges being followed back [50]. The estimate $\hat{\alpha}$ (the coefficient for "log Followers") for this model is significantly larger than 1, again consistent with super-linear preferential attachment.

In the second model, we test the effect of the target node being a friend-of-friend of the choosing node. In the case of Flickr, this means that the choosing user already follows someone that follows the target user, which evidently is strongly correlated with following that user. However, combining these two features in a third model (column 3) leads to both estimated parameters dropping substantially. Most remarkable is the 40% drop in the estimate of α, which paints a very different picture about the role of degree.

In the fourth model, we measure network proximity as in the original paper, by counting the number of "hops" (path length) from i to the target before an edge was made. We integrate the hops as categorical variables to show the relative impact of each additional "hop". Being two hops away is equivalent to being a friend-of-friend, and thus has strongly positive coefficient. Every additional hop corresponds to a sharp decrease in choosing that node. Being five hops away is slightly worse than there not being a path at all. This could be an artifact of the way the data was gathered, so that new regions of the graph only get "discovered" when there is at least one link to them, or this could be due to path length not being an accurate measure of distance for newer nodes. Since the number of hops is co-linear with being a friend-of-friend, we can't test them both at the same time.

In Figure 5 we visually show the effect of different specifications on the estimate of $\hat{\alpha}$. The first model of the Flickr data looks like super-linear preferential attachment, while the role of degree in the other two is significantly reduced. However, fitting a non-parametric model shows that the estimated coefficients for individual degrees are remarkably linear, suggesting that the functional form of d_j^{α} is a good fit for this network. One important point is the role of zero-degree nodes. In most descriptions of preferential attachment, nodes with degree zero are not considered. However, in the Flickr data set, zero-degree nodes have a higher utility than positive low degree nodes, which could again be an artifact of the data collection process, or point to the special role of new nodes in the network. Either way, our framework allows one to find these kinds of patterns, and investigate them further.

4.4 Choosing to cite

We now turn to citation network data to show how a discrete choice framework facilitates the testing of network formation hypotheses. Previous analyses of citation networks have observed linear preferential attachment with respect to degree [62] and bias towards citing more recent work [62]. Here, we find consistent results that

[5]This solution is better than using $\log d + \epsilon$, or giving degree-zero nodes the same utility as degree-one nodes. Either of those solutions will give substantially different results, especially when there are many degree-zero nodes.

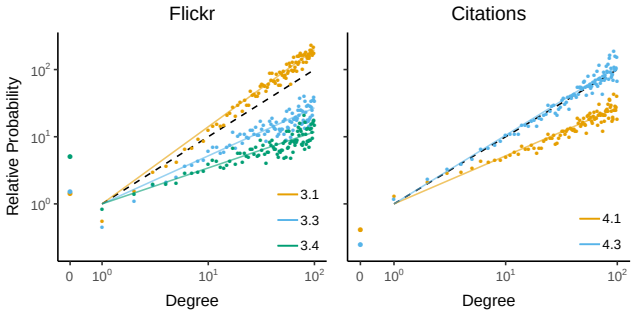


Figure 5: The probability of being chosen by degree, as compared to a node with degree 1. We show the fits of parametric (lines) and non-parametric (points) conditional logit models of the Flickr and citation networks. The legend references model numbers in Table 3 and Table 4. The estimate for degree 0 is inserted for comparison. Dashed reference lines illustrate what exact linear preferential attachment ($\alpha = 1$) would look like.

older papers are less likely to be cited but that accounting for age actually *increases* the importance of degree (i.e., after accounting for age, higher degree nodes are more likely to be cited).

Data. We use the Microsoft Academic Graph[6] dataset and focus on a representative subgraph of 459,000 "Climatology" papers. We focus on the subgraph of a single field to simplify the analysis since citations are predominantly within the same field of study (our analysis was similar on other subgraphs). We construct a graph out of this data by adding an edge each time a paper in our dataset cites another paper in our dataset. For our analysis of Climatology publications, 45% of edges are within the domain and citations to papers that are not labeled are excluded, leaving 3 million edges. We sample 10,000 citation events uniformly at random from papers published after 2010 and apply negative sampling ($s = 24$). This processing results in 10,000 choices with 25 alternatives in each choice set. For each possible choice, we compute four features: the number of citations at the time of citation, whether the paper shares authors with the citing paper, the age of the paper in years at the time of citation, and the maximum number of publications by any one of the authors at the time of publication. This last feature is a proxy for node fitness [11].

Discrete choice analysis. We fit conditional logit choice models relating these features to the likelihood of citation (Table 4). The first model (first column) is a simple log-degree model. We find that the estimate $\hat{\alpha}$ (the coefficient for "log Citations") is substantially lower than one, consistent with sub-linear preferential attachment. Apart from the log-likelihood of the models, we also report the predictive accuracy (defined as the share of instances predicted correctly) on a holdout test set of 2,000 examples. Just relying on prior degree already gives an accuracy of 36%, which is high for a classification task with 25 classes. In model two (second column), we add a covariate for whether a paper shares an author with the citing paper. As expected, this has a strongly positive coefficient.

For the third model we add a covariate for the age of the paper in log years (years is always at least one). Older papers are

[6]The Aminer Project [68, 72], https://aminer.org/open-academic-graph

Table 4: Learned conditional logits for the "Climatology" citation network. Standard errors of the estimates are given in parentheses. Evaluation statistics are computed over 2,000 sampled examples excluded from the training data.

	Model			
	#1	#2	#3	#4
log Citations	0.717*	0.794*	1.052*	1.044*
	(0.008)	(0.010)	(0.012)	(0.012)
Has degree	1.684*	1.677*	1.862*	1.830*
	(0.053)	(0.062)	(0.063)	(0.064)
Has same author		6.523*	5.928*	5.913*
		(0.110)	(0.114)	(0.114)
log Age			-1.096*	-1.069*
			(0.018)	(0.021)
Max papers by author				0.029*
				(0.011)
Observations	10,000	10,000	10,000	10,000
Log-likelihood	-20,799	-16,600	-14,384	-14,390
Test accuracy	0.358	0.484	0.533	0.534
Note:				*p<0.01

less likely to get cited (accounting for degree), but accounting for age *increases* the relative importance of degree significantly. This expanded model also increases the accuracy to 53%, indicating that these feature weights do capture substantially more predictive power. Finally, in model four we add the "max papers by authors" feature as a proxy for fitness. The coefficient is small but positive. Accounting for fitness slightly reduces the estimated relative importance of degree, but the $\hat{\alpha}$ estimate is still close to 1. Adding this feature does not improve the log-likelihood or predictive accuracy; a better proxy for fitness may explain the data better. Looking back to the visual display of α for the citation models in Figure 5, the non-parametric coefficients are highly linear. In this data, zero-degree nodes are significantly *less* attractive than nodes with degree one.

As with any regression, the identifying causal effects from model fit depends on the design of the study. The estimates we provide here, as is the case with most analyses of observational data, are descriptive and not meant to describe causal processes. The point is that discrete choice models provide a flexible framework to easily test and compare different hypotheses around network formation.

5 DISCUSSION

When modeling network formation, the majority of the literature analyzes networks that grow "externally," with new nodes arriving and choosing who to connect to, and this setting has also been our main focus here. External growth leads to convenient models that are relatively easy to analyze, with citation networks and patent networks as examples of empirical networks that follow this generative process reasonably closely. However, in many (especially social) networks, pairs of older nodes often form edges as well, edges that are "internal" to the existing set of nodes. An extreme example is the social networks of schools or classrooms, which have a fixed node population and "grow" purely through an internal growth process. A major advantage of modeling network formation as discrete choice is that it does not require any model of edge event initiation and simply conditions on the sequence of decisions to initiate, focusing the modeling on the choices made by the initiator. Discrete choice can therefore easily be used to model internal growth as well.

Another major advantage of discrete choice modeling is that it connects the analysis of large-scale network datasets to statistical methods (fitting generalized linear models) that are tremendously scalable. As we show in this work, additional techniques (e.g., negative sampling) makes it possible to efficiently scale the estimation process to very large network datasets.

Since the conditional logit model of discrete choice is a random utility model, the estimated parameters can be interpreted as the marginal utility of each feature. This allows one to question the functional form of features. For example, we show that preferential attachment is equivalent to the logarithmic utility of degree. Given that degree is commonly heavy-tailed, this is a natural functional form, but we point out that the conditional logit allows one to flexibly compare different specifications.

Our discrete choice perspective has implications for how network data is best collected and analyzed. It is useful to consider and record notions of directionality, even if edges can otherwise be considered to be undirected. With information about the choice set associated with each choice, we can see what each node j looked like at the time the choice was made. Datasets that record the exact time of all edge formation events, as opposed to lumping edge events at the granularity of days or years, makes it possible to further analyze the formation process in more detail.

There are a couple limitations to our proposed methodology. First, we cannot model purely undirected edges without some notion of direction. Second, even though the conditional logit and mixed logit models allow one to model similar mechanisms, the interpretations of their estimates are different. The estimates of a conditional logit are more akin to those of a linear regression model, where one estimates the expected change in an outcome from varying a covariate. A mixture model is a probabilistic combination of constituent modes, so the class probabilities indicate the relative importance to each mode, which makes it harder to compare the roles of individual features within or across modes. However, many traditional models of network formation are equivalent to mixture models, which motivated our consideration of them in this work.

By making foundational connections between network formation and discrete choice, we are hopeful that many further tools from discrete choice theory can be applied to the study of network formation. For example, there can be bias in network formation, e.g., men are more likely to cite themselves than women [34]. Our discrete choice framework can help study these cases more rigorously. For another example, discrete choice models of subset selection [5, 20] could be applied to understand possible substitution and complementarity effects in network formation. And discrete choice interpretations of machine learning embeddings techniques [64] can likely help unpack the behavior of recent embedding-based network representation methods such as DeepWalk [57]. Networks fundamentally represent interactions between discrete entities, and

it is therefore natural that methods for modeling and analyzing discrete choice should enable many contributions.

Acknowledgements. We thank Aaron Clauset, Eduardo Laguna-Müggenburg, and Daniel Larremore for their helpful comments and feedback. ARB was supported by NSF Award DMS-1830274 and ARO award 86798. JO and JU were supported in part by an ARO Young Investigator Award.

REFERENCES

[1] Alan Agresti. 2003. *Categorical data analysis*. Vol. 482. John Wiley & Sons.
[2] Réka Albert and Albert-László Barabási. 1999. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512.
[3] James P Bagrow and Dirk Brockmann. 2013. Natural Emergence of Clusters and Bursts in Network Evolution. *Physical Review X* 3, 2 (2013).
[4] Austin R Benson, Ravi Kumar, and Andrew Tomkins. 2016. On the relevance of irrelevant alternatives. In *WWW*. ACM, 963–973.
[5] Austin R Benson, Ravi Kumar, and Andrew Tomkins. 2018. A Discrete Choice Model for Subset Selection. In *WSDM*. ACM, 37–45.
[6] Ginestra Bianconi and Albert-László Barabási. 2001. Bose-Einstein condensation in complex networks. *Physical review letters* 86, 24 (2001).
[7] Ginestra Bianconi and A-L Barabási. 2001. Competition and multiscaling in evolving networks. *EPL (Europhysics Letters)* 54, 4 (2001), 436.
[8] Béla Bollobás and Oliver M Riordan. 2003. Mathematical results on scale-free random graphs. *Handbook of graphs and networks: from the genome to the internet* (2003), 1–34.
[9] Anna D Broido and Aaron Clauset. 2018. Scale-free networks are rare. *arXiv preprint arXiv:1801.03400* (2018).
[10] Carter T Butts. 2008. A Relational Event Framework for Social Action. *Sociological Methodology* 38, 1 (2008), 155–200.
[11] Guido Caldarelli, Andrea Capocci, Paolo De Los Rios, and Miguel A Munoz. 2002. Scale-free networks from varying vertex intrinsic fitness. *Physical Review Letters* 89, 25 (2002), 258702.
[12] Duncan S Callaway, John E Hopcroft, Jon M Kleinberg, Steven H Strogatz, and Mark E J Newman. 2001. Are randomly grown graphs really random? *Physical Review E* 64, 4 (2001).
[13] Sourav Chatterjee and Persi Diaconis. 2013. Estimating and understanding exponential random graph models. *Annals of Statistics* 41, 5 (2013), 2428–2461.
[14] Flavio Chierichetti, Ravi Kumar, and Andrew Tomkins. 2018. Learning a mixture of two multinomial logits. In *ICML*. PMLR, 961–969.
[15] Aaron Clauset, Cristopher Moore, and Mark E J Newman. 2008. Hierarchical structure and the prediction of missing links in networks. *Nature* 453, 7191 (2008), 98–101.
[16] Aaron Clauset, Cosma R Shalizi, and Mark E J Newman. 2009. Power-Law Distributions in Empirical Data. *SIAM Rev.* 51, 4 (Dec. 2009), 661–703.
[17] Colin Cooper and Alan Frieze. 2003. A general model of web graphs. *Random Structures & Algorithms* 22, 3 (2003), 311–335.
[18] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B* (1977), 1–38.
[19] David Easley and Jon Kleinberg. 2010. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press.
[20] Peter C Fishburn and Irving H LaValle. 1996. Binary interactions and subset choice. *European Journal of Operational Research* 92, 1 (1996), 182–192.
[21] Winship C Fuller, Charles F Manski, and David A Wise. 1982. New evidence on the economic determinants of postsecondary schooling choices. *Journal of Human Resources* (1982), 477–498.
[22] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. 2002. Latent space approaches to social network analysis. *Journal of the american Statistical association* 97, 460 (2002), 1090–1098.
[23] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. 1983. Stochastic blockmodels: First steps. *Social networks* 5, 2 (1983), 109–137.
[24] Petter Holme and Beom Jun Kim. 2002. Growing scale-free networks with tunable clustering. *Physical review E* 65, 2 (2002).
[25] Petter Holme and Jari Saramäki. 2012. Temporal networks. *Physics reports* 519, 3 (2012), 97–125.
[26] Samuel Ieong, Nina Mishra, and Or Sheffet. 2012. Predicting preference flips in commerce search. In *ICML*. PMLR, 1795–1802.
[27] Kansuke Ikehara and Aaron Clauset. 2017. Characterizing the structural diversity of complex networks across domains. *arXiv preprint arXiv:1710.11304* (2017).
[28] Matthew O Jackson and Brian W Rogers. 2007. Meeting Strangers and Friends of Friends: How Random Are Social Networks? *American Economic Review* 97, 3 (2007), 890–915.
[29] Benjamin F Jarvis. 2018. Estimating Multinomial Logit Models with Samples of Alternatives. *Sociological Methodology* (2018).
[30] H Jeong, Z Neda, and Albert-László Barabási. 2003. Measuring preferential attachment in evolving networks. *Europhysics Letters (EPL)* 61, 4 (2003), 567–572.
[31] Emily M Jin, Michelle Girvan, and Mark EJ Newman. 2001. Structure of growing social networks. *Physical Review E* 64, 4 (2001), 046132.
[32] Wagner A Kamakura and Gary J Russell. 1989. A probabilistic choice model for market segmentation and elasticity structure. *Journal of mMrketing Research* (1989), 379–390.
[33] Brian Karrer and Mark EJ Newman. 2011. Stochastic blockmodels and community structure in networks. *Physical review E* 83, 1 (2011), 016107.
[34] Molly M King, Carl T Bergstrom, Shelley J Correll, Jennifer Jacquet, and Jevin D West. 2017. Men set their own cites high: Gender and self-citation across fields and over time. *Socius* 3 (2017).
[35] Jon M Kleinberg, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew S Tomkins. 1999. The web as a graph: measurements, models, and methods. In *International Computing and Combinatorics Conference*. Springer, 1–17.
[36] Paul L Krapivsky, Sidney Redner, and Francois Leyvraz. 2000. Connectivity of growing random networks. *Physical review letters* 85, 21 (2000), 4629.
[37] Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. 2010. Hyperbolic geometry of complex networks. *Physical Review E* 82, 3 (2010), 036106.
[38] Ravi Kumar, Jasmine Novak, and Andrew Tomkins. 2010. Structure and evolution of online social networks. In *Link mining: models, algorithms, and applications*. Springer, 337–357.
[39] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D Sivakumar, Andrew Tomkins, and Eli Upfal. 2000. Stochastic models for the web graph. In *Proceedings of the 42st Annual Symposium on Foundations of Computer Science*. IEEE, 57–65.
[40] Jure Leskovec, Lars Backstrom, Ravi Kumar, and Andrew Tomkins. 2008. Microscopic evolution of social networks. In *KDD*. ACM, 462–470.
[41] Jure Leskovec, Jon M Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *Transactions on Knowledge Discovery from Data (TKDD)* 1, 1 (2007).
[42] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58, 7 (2007), 1019–1031.
[43] Zonghua Liu, Ying-Cheng Lai, Nong Ye, and Partha Dasgupta. 2002. Connectivity distribution and attack tolerance of general networks with both preferential and random attachments. *Physics Letters A* 303, 5-6 (2002), 337–344.
[44] Linyuan Lü and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications* 390, 6 (2011), 1150–1170.
[45] R Duncan Luce. 1959. *Individual Choice Behavior; a Theoretical Analysis*. New York: Wiley.
[46] Daniel McFadden. 1978. Modeling the choice of residential location. *Transportation Research Record* 673 (1978).
[47] Daniel McFadden et al. 1973. Conditional logit analysis of qualitative choice behavior. (1973).
[48] Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology* 32, 1 (2001), 19–28.
[49] Jan Medina, Jorge Finke, and Camilo Rocha. 2018. Estimating Formation Mechanisms and Degree Distributions in Mixed Attachment Networks. *arXiv.org* (2018). arXiv:math.PR/1809.03372v1
[50] Alan Mislove, Hema Swetha Koppula, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. 2008. Growth of the Flickr Social Network. In *Proceedings of the 1st SIGCOMM Workshop on Social Networks*.
[51] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. 2007. Measurement and analysis of online social networks. In *Proceedings of the 7th SIGCOMM conference on Internet Measurement*. ACM.
[52] Michael Mitzenmacher. 2003. A Brief History of Generative Models for Power Law and Lognormal Distributions. *Internet Mathematics* 1, 2 (2003), 226–251.
[53] Rajeev Motwani and Ying Xu. 2006. Evolution of page popularity under random web graph models. In *PODS*. ACM, 134–142.
[54] Mark E J Newman. 2001. Clustering and preferential attachment in growing networks. *Physical Review E* 64, 2 (2001), 440–4.
[55] Fragkiskos Papadopoulos, Maksim Kitsak, M Ángeles Serrano, Marián Boguná, and Dmitri Krioukov. 2012. Popularity versus similarity in growing networks. *Nature* 489, 7417 (2012), 537.
[56] Ashwin Paranjape, Austin R Benson, and Jure Leskovec. 2017. Motifs in temporal networks. In *WSDM*. ACM, 601–610.
[57] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*. ACM, 701–710.
[58] Thong Pham, Paul Sheridan, and Hidetoshi Shimodaira. 2015. PAFit: A Statistical Method for Measuring Preferential Attachment in Temporal Complex Networks. *PLoS ONE* 10, 9 (2015).
[59] Derek de Solla Price. 1976. A general theory of bibliometric and other cumulative advantage processes. *Journal of the American Society for Information Science* 27, 5 (1976), 292–306.
[60] Stephen Ragain and Johan Ugander. 2016. Pairwise choice Markov chains. In *NIPS*. 3198–3206.

[61] Anatol Rapoport. 1953. Spread of information through a population with socio-structural bias: I. Assumption of transitivity. *The bulletin of mathematical biophysics* 15, 4 (1953), 523–533.

[62] Sidney Redner. 2005. Citation Statistics from 110 Years of Physical Review. *Physics Today* 58, 12 (2005).

[63] Garry Robins, Pip Pattison, Yuval Kalish, and Dean Lusher. 2007. An introduction to exponential random graph (p*) models for social networks. *Social networks* 29, 2 (2007), 173–191.

[64] Maja Rudolph, Francisco Ruiz, Stephan Mandt, and David Blei. 2016. Exponential family embeddings. In *NIPS*. 478–486.

[65] Jari Saramäki and Kimmo Kaski. 2004. Scale-free networks generated by random walkers. *Physica A: Statistical Mechanics and its Applications* 341 (2004), 80–86.

[66] Cosma R Shalizi and Alessandro Rinaldo. 2013. Consistency under sampling of exponential random graph models. *Annals of Statistics* 41, 2 (2013), 508–535.

[67] Itamar Simonson and Amos Tversky. 1992. Choice in context: Tradeoff contrast and extremeness aversion. *Journal of Marketing Research* 29, 3 (1992), 281.

[68] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-june Paul Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *WWW*. ACM, 243–246.

[69] Tom AB Snijders. 2001. The Statistical Evaluation of Social Network Dynamics. *Sociological Methodology* 31, 1 (2001), 361–395.

[70] Tom AB Snijders, Gerhard G Van de Bunt, and Christian EG Steglich. 2010. Introduction to stochastic actor-based models for network dynamics. *Social networks* 32, 1 (2010), 44–60.

[71] Christoph Stadtfeld, James Hollway, and Per Block. 2017. Dynamic Network Actor Models: Investigating Coordination Ties through Time. *Sociological Methodology* 47, 1 (2017), 1–40. https://doi.org/10.1177/0081175017709295

[72] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *KDD*. ACM, 990–998.

[73] Kenneth E Train. 2008. EM algorithms for nonparametric estimation of mixing distributions. *Journal of Choice Modelling* 1, 1 (2008), 40–69.

[74] Kenneth E Train. 2009. *Discrete choice methods with simulation*. Cambridge university press.

[75] Kenneth E Train and Daniel McFadden. 1978. The goods/leisure tradeoff and disaggregate work trip mode choice models. *Transportation research* 12, 5 (1978), 349–353.

[76] Amos Tversky. 1972. Elimination by aspects: A theory of choice. *Psychological Review* 79, 4 (1972), 281.

[77] Johan Ugander, Brian Karrer, Lars Backstrom, and Cameron Marlow. 2011. The anatomy of the facebook social graph. *arXiv preprint arXiv:1111.4503* (2011).

[78] Alexei Vázquez. 2003. Growing network with local rules: Preferential attachment, clustering hierarchy, and degree correlations. *Physical Review E* 67, 5 (2003).

[79] Stanley Wasserman and Philippa Pattison. 1996. Logit models and logistic regressions for social networks: I. An introduction to Markov graphs and p*. *Psychometrika* 61, 3 (1996), 401–425.

[80] Samuel S Wilks. 1938. The large-sample distribution of the likelihood ratio for testing composite hypotheses. *Annals of Mathematical Statistics* 9, 1 (1938), 60–62.

[81] Carsten Wiuf, Markus Brameier, Oskar Hagberg, and Michael PH Stumpf. 2006. A likelihood approach to analysis of network data. *PNAS* 103, 20 (2006), 7566–7570.

[82] Lei Xu and Michael I Jordan. 1996. On Convergence Properties of the EM Algorithm for Gaussian Mixtures. *Neural Computation* 8, 1 (1996), 129–151.

Policy Gradients for Contextual Recommendations

Feiyang Pan*
Institute of Computing Technology, Chinese Academy of Sciences
panfeiyang@ict.ac.cn

Qingpeng Cai
IIIS, Tsinghua University
cqp14@mails.tsinghua.edu.cn

Pingzhong Tang†
IIIS, Tsinghua University
kenshinping@gmail.com

Fuzhen Zhuang*
Institute of Computing Technology, Chinese Academy of Sciences

Qing He*
Institute of Computing Technology, Chinese Academy of Sciences

ABSTRACT

Decision making is a challenging task in online recommender systems. The decision maker often needs to choose a contextual item at each step from a set of candidates. Contextual bandit algorithms have been successfully deployed to such applications, for the trade-off between exploration and exploitation and the state-of-art performance on minimizing online costs. However, the applicability of existing contextual bandit methods is limited by the over-simplified assumptions of the problem, such as assuming a simple form of the reward function or assuming a static environment where the states are not affected by previous actions.

In this work, we put forward *Policy Gradients for Contextual Recommendations (PGCR)* to solve the problem without those unrealistic assumptions. It optimizes over a restricted class of policies where the marginal probability of choosing an item (in expectation of other items) has a simple closed form, and the gradient of the expected return over the policy in this class is in a succinct form. Moreover, PGCR leverages two useful heuristic techniques called Time-Dependent Greed and Actor-Dropout. The former ensures PGCR to be empirically greedy in the limit, and the latter addresses the trade-off between exploration and exploitation by using the policy network with Dropout as a Bayesian approximation.

PGCR can solve the standard contextual bandits as well as its Markov Decision Process generalization. Therefore it can be applied to a wide range of realistic settings of recommendations, such as personalized advertising. We evaluate PGCR on toy datasets as well as a real-world dataset of personalized music recommendations. Experiments show that PGCR enables fast convergence and low regret, and outperforms both classic contextual-bandits and vanilla policy gradient methods.

CCS CONCEPTS

• **Information systems → Online advertising**; • **Computing methodologies → Machine learning**.

*Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS). Also at University of Chinese Academy of Sciences, China.
†Correponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313616

KEYWORDS

Recommender System; Policy Gradients; Contextual Bandits; Reinforcement Learning;

ACM Reference Format:
Feiyang Pan, Qingpeng Cai, Pingzhong Tang, Fuzhen Zhuang, and Qing He. 2019. Policy Gradients for Contextual Recommendations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313616

1 INTRODUCTION

Decision making in online recommender systems and advertising systems are challenging because the recommender needs to find the policy that maximizes its revenue by interacting with the world. A typical decision-making problem is to select a featured item from a finite set of candidates, for example, to select an advertisement from a set of ads that relate to the user's query in a search engine, or to recommend a song from the user's playlist in a music streaming service. After making the decision, the recommender will receive a reward together with some state transition. In such settings, each item has a so-called *context* (which includes features and attributes) that carries all the necessary information for making the choice. Since it is often the case that the reward, as well as the state dynamic of choosing each item, are related to its context, the recommender system must try to learn how to make the choice given the contexts of all the candidates.

To solve such contextual recommendation problems, algorithms based on *contextual-bandits* have been successfully deployed in a number of industrial level applications over the past decade, such as personalized recommender systems [17, 27, 28], advertisement personalization [6, 29], and learning-to-rank [22]. Contextual-bandit algorithms are preferred if one needs to minimize the cumulative cost during online-learning because they aim to address the trade-off between exploitation and exploration.

The standard contextual bandit problem can be seen as a repeated game between nature and the player [16]. Nature defines a reward function mapping contexts (a set of features) to real-valued rewards, which is unknown to the player. At each step, nature gives a set of items, each with a context. The player observes the contextual items, selects one, and then receives a reward. The payoff of the player is to minimize the cumulative regret or to maximize the cumulative reward.

The main challenge of solving contextual bandits lies in the trade-off between exploration and exploitation. The most well-known approaches are arguably value-based methods including Upper Confidence Bounds (UCB) [5], Thompson Sampling (TS) [30], and their variants. These value-based methods try to estimate the expected reward of choosing each item, so they are especially effective when the form of reward function is known explicitly. For example, when the expected reward is linear in the context, [1, 9, 17] proposed Lin-UCB, which is applied successfully to the online news recommendation of Yahoo, and [4, 8, 19] also proposed TS to solve the linear contextual bandits. Similarly, [10] proposed GLM-UCB using generalized linear models, [15, 23] used Gaussian Processes, to model the reward functions. These variants of UCB and TS have been known to achieve sub-linear regrets [1, 2, 4, 5, 8–10, 15, 17, 23]. Similar ideas have also been applied to reinforcement learning algorithm such as the UCRL algorithm with regret bounds [13].

However, the applicability of these approaches in real-world applications is heavily limited, especially for large-scale and high-dimensional problems, due to the following reasons:

- First, these methods tend to over-simplify the form of the reward function, which is unrealistic in real-world cases. For example, for sponsored search advertising via real-time bidding, the reward of showing an ad (cost per click) is often the click-through rate multiplied by the bidding price, so it can be understood as a mixture of binary and linear outcomes. Moreover, the reward can often be a high-order non-linear function of features in the contexts.
- Second, the overall formulations of contextual-bandit problems are sometimes over-simplified comparing to real-world applications. It is often assumed that the reward is determined by the context of the currently chosen item, and the distribution of contexts is independent of the agent's action. However, it may not be true in real-world recommender systems where the behaviors of users heavily depend on not only the current contexts but the history, i.e., the items that he/she viewed in previous rounds. Also, the set of candidate items can relate to the user's previous preferences as well. These dependencies are not well exploited in existing models.
- Last but not least, these methods are value-based, so they are meant to find deterministic policies. A subtle change in the value estimation may cause a discontinuous jump in the resulting policy, which makes convergence difficult for these [25]. On the other hand, stochastic policies are sometimes preferred in online recommender systems.

In light of these observations, we propose Policy Gradients for Contextual Recommendations (PGCR), which uses the policy gradient method to solve general contextual recommendations. Our approach model the contextual recommendation problem without unrealistic assumptions or prior knowledge. By optimizing directly over the parameters of stochastic policies, it naturally fits the problems that require randomized actions as well as addresses the trade-off between exploration and exploitation.

Since we design PGCR specifically for contextual recommendations, we would like to specify the performance objective first and see if it is different from the one in standard reinforcement learning. We find that the objective over policies depends on the marginal expected probability of choosing each item (in expectation of other items). So PGCR restricts the search space to a class of policies in which the expected probabilities of choosing an item has a simple closed form and can be estimated efficiently. Therefore, the search space for PGCR is dramatically reduced.

Then, in order to estimate the marginal probability of choosing each item, we extend Experience Replay, the popular technique in off-policy reinforcement learning [3, 12], to a finer-grained sampling procedure. By doing so, the variance of estimating policy gradients can be much smaller than the variance of the vanilla policy gradient algorithm. The resulted algorithm is also computationally efficient by stochastic gradient descent with mini-batch training.

To address the trade-off of exploration and exploitation, our proposed PGCR empirically has the property of Greedy in the Limit with Infinite Exploration (GLIE), which is an essential property for contextual bandits [19]. The property is guaranteed by two useful heuristics named *Time-Dependent Greed* and *Actor-Dropout*. Time-Dependent Greed is to schedule the level of greed to increase over time, so the resulted stochastic policy will explore a lot in the early stage and then gradually converge to a greedy policy. Actor-Dropout is to use dropout on the policy network while training and inferring, thus the feed-forward network outputs policies with randomness. It has been known that such a stochastic feed-forward neural network can be seen as a Bayesian approximation [11], so it can provide with directed exploration for PGCR.

Furthermore, with the mentioned techniques, PGCR can directly apply to contextual recommendations in a Markov Decision Process (MDP) setting, i.e. with states and state transitions. We propose this generalized setting for the reason that the i.i.d. assumption on contexts in the standard contextual bandit setting is unrealistic for real-world applications. On the other hand, we suppose that at each step, the contexts are drawn i.i.d. from a distribution conditional on the current state. Furthermore, when an item is chosen, the immediate reward is determined by both the state and the selected item. The state is then transitioned into the next state. Such a model is tailored for a wide range of important realistic applications such as personalized recommender systems where users' preferences are regarded as states and items are regarded as items with contexts [20, 26], and e-commerce where the private information (e.g., cost, reputation) of sellers can be viewed as states and different commercial strategies are regarded as contexts [7].

We evaluate PGCR on toy datasets and a real-world dataset of music recommendation. By comparing with several common baselines including Lin-UCB, GLM-UCB, Thompson Sampling, ϵ-greedy, and vanilla policy gradients, it shows that PGCR converges quickly and achieves the lowest cumulative regret and the highest average reward in various standard contextual-bandits settings. Moreover, when state dynamics are included in the real-world recommendation environments, we find that GLM-UCB and TS fail to incorporate information from the states, while PGCR consistently outperforms other baselines.

2 PROBLEM FORMULATION

2.1 One-step Contextual Recommendations

We first introduce the simplified setting of contextual recommendation as a standard contextual-bandits problem. At each step, we have a set of contexts $\mathbf{c} = (c_1, \ldots, c_m)^T$ that corresponds to m items, where c_i is the context of the i^{th} item. The contexts $c_1, \ldots, c_m$ are independently and identically distributed random variables with outcome space C. The action is to select an item from the candidates, $a \in \{1, ..., m\}$.

For the ease of notation, we use $\mathbf{c}$ to denote the concatenation of all m contexts and use c_a to denote the context of the selected item a. We write the random variable of immediate reward as $R(c_a)$ to note that in this setting it depends only on the chosen context vector c_a. The dependency is not known to the decision-maker. So the target is to learn the dependency and choose the item with the largest expected reward.

A stochastic policy π is a function that maps the observations (the set of contexts $\mathbf{c}$) to a distribution of actions. Let the random variable $a \sim \pi(\mathbf{c})$ denote the action determined by policy π. The performance of a policy is measured as the expected reward of the chosen item over all possible contexts, i.e.,

$$J(\pi) = \mathbb{E}_{\mathbf{c}}\Big[R(c_a) \mid a \sim \pi(\mathbf{c})\Big], \tag{1}$$

where c_a is short for the context of the chosen action a.

When the policy π is parameterized as π_θ where θ is the trainable parameters, our goal is to find the optimal choice of θ that maximizes the objective $J(\pi_\theta)$.

However, there is an obvious drawback for this simplified setting: in real-world recommendations, it cannot be assumed that the contexts are always drawn i.i.d from some global probability distribution. For example, when recommending items (goods) to a customer given the searching query in an e-commerce platform, we can only select items from a candidate pool related to the query. This non-i.i.d nature motivates us to put forward a more general setting, which involves the states.

2.2 Sequential State-aware Contextual Recommendations

In this part, we introduce the generalized setting as a Markov Decision Process (MDP) with states and state transitions for contextual recommendations, which is referred to as MDP-CR.

At each step t, the decision maker observes its state s_t as well as a set of contexts correlated to that state $\mathbf{c_t} = \{c_{t1}, \ldots, c_{tm}\}$. When an action (one of the items) $a_t = \pi(s_t, \mathbf{c_t})$ is selected, a reward $R(s_t, c_{ta_t})$ is received, and the state is transitioned to the next state by a Markovian state transition probability $s_{t+1} \sim T(s_{t+1} \mid s_t, c_{ta_t})$. Note that the setting in this paper is different from other existing generalized bandits with transitions such as Restless bandits [31].

In this setting, we assume that the contexts are independently distributed conditioning on the state: $c_{ti} \sim g^{s_t}(c)$ for all i, where $g^{s_t}(c)$ is the probability density of contexts given state s_t. For example, if the state is the search query or the attribute vector of a user, we assume that the contexts of items in the candidate pool are drawn from a distribution that reflects the search query or the user preference.

The goal is to find a policy that maximizes the expected cumulative discounted reward, so the objective is

$$J(\pi) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R(s_t, c_{ta_t}) \mid a_t \sim \pi(s_t, \mathbf{c_t})\Big], \tag{2}$$

where $0 < \gamma < 1$ is a discount factor that balances short and long term rewards, just like in standard reinforcement learning. We also define the action value function

$$Q^\pi(s, c) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t R(s_t, c_{ta_t}) \mid s_0 = s, c_{0a_0} = c, \pi\Big]. \tag{3}$$

Same as previous works [21, 25] on policy gradients, we denote the discounted state density by $\rho^\pi(s) = \int_{\mathcal{S}} \sum_{t=0}^{\infty} \gamma^t P_0(s_0) P(s_0 \to s, t, \pi) \mathrm{d}s_0$, where $P_0(s_0)$ is the probability density of initial states, and $P(s \to s', t, \pi)$ is the probability density at state s' after transitioning for t time steps from state s.

Thus we can rewrite the objective as

$$J(\pi) = \mathbb{E}_{s\sim\rho^\pi, \mathbf{c}\sim g^s, a\sim\pi(s,\mathbf{c})}\Big[R(s, c_a)\Big]. \tag{4}$$

3 POLICY GRADIENTS FOR ONE-STEP CONTEXTUAL RECOMMENDATIONS

In this section we investigate several key features of our purposed PGCR method. For readability, we first discuss the one-step contextual recommendation case (corresponding to section 2.1), which can be modeled as the standard contextual-bandits. Later in the next section, we will show how to extend to the generalized multi-step recommendation which can be modeled as MDP-CR.

3.1 Marginal probability for choosing an item

Due to the assumption of the problem setting that the reward only depends on the selected context, we claim that for any policy π, there exists a *permutation invariant policy* that obtains at least its performance.

Definition 1 (Permutation invariant policy). *A policy $\pi(\mathbf{c})$ is said to be permutation invariant if for all $\mathbf{c} \in C^m$ and any its permutation $\mathbf{c}' := P(\mathbf{c})$, it has*

$$c'_{a'} \overset{dist}{=} c_a \tag{5}$$

where $a' \sim \pi(\mathbf{c}')$ and $a \sim \pi(\mathbf{c})$ denote for the actions chosen by the policy π, $c'_{a'}$ and c_a is the their corresponding contexts respectively, and $\overset{dist}{=}$ means the probability distribution of the expressions on two sides are the same.

Lemma 1. *For any policy π, there exists a permutation invariant policy π' s.t. $J(\pi') \geq J(\pi)$.*

Proof. Let us suppose, for the sake of contradiction, that there exists a policy π such that

(i) it is not permutation invariant, i.e. there exists $\mathbf{c} \in C^m$ and some permutation operator $P \in \mathcal{P}$ that $c_a \neq c'_{a'}$ where $\mathbf{c}' = P(\mathbf{c})$ and $a' \sim \pi(\mathbf{c}')$;

(ii) The expected reward following π is larger than all permutation invariant policies $\tilde{\pi}$ that $J(\pi) > J(\tilde{\pi})$.

Then it follows that $\mathbb{E}_{\mathbf{c}}\Big[R(\mathbf{c}_{\pi(\mathbf{c})})\Big] > \mathbb{E}_{\mathbf{c}}\Big[R(\mathbf{c}_{\tilde{\pi}(\mathbf{c})})\Big]$ for all permutation invariant $\tilde{\pi}$, where the expectation is over all sets of contexts.

Recall that the contexts are drawn i.i.d. from the same distribution, so we have

$$\mathbb{E}_{\mathbf{c}}\Big[R(\mathbf{c}_{\pi(c)})\Big] \equiv \mathbb{E}_{\mathbf{c}}\Big[\frac{1}{|\mathcal{P}|}\sum_{P\in\mathcal{P}} R(P(\mathbf{c})_{\pi(P(\mathbf{c}))})\Big] > \mathbb{E}_{\mathbf{c}}\Big[R(\mathbf{c}_{\tilde{\pi}(c)})\Big], \tag{6}$$

so there exists at least one $\mathbf{c}$ that

$$\frac{1}{|\mathcal{P}|}\sum_{P\in\mathcal{P}} R(P(\mathbf{c})_{\pi(P(\mathbf{c}))}) > R(\mathbf{c}_{\tilde{\pi}(c)}) \text{ for all } \tilde{\pi}. \tag{7}$$

But because π is not permutation invariant, we find a policy $\pi^*(P(\mathbf{c})) = \pi((P^*P^TP)(\mathbf{c}))$ that is permutation invariant, where

$$P^* = \arg\max_{P\in\mathcal{P}} R(P(\mathbf{c})_{\pi(P(\mathbf{c}))}),$$

then

$$R(\mathbf{c}_{\pi^*(c)}) = R(P^*(\mathbf{c})_{\pi(P^*(\mathbf{c}))}) > \frac{1}{|\mathcal{P}|}\sum_{P\in\mathcal{P}} R(P(\mathbf{c})_{\pi(P(\mathbf{c}))}), \tag{8}$$

which leads to a contradictory to (6) and (7). So it must be that Lemma 1 holds. □

Lemma 1 states that, without loss of generality, we can focus on permutation invariant policies. The objective in then becomes

$$\begin{aligned} J(\pi) &= \mathbb{E}_{\mathbf{c}}\Big[R(c_a) \mid a\sim\pi(\mathbf{c})\Big] \\ &= \mathbb{E}_{\mathbf{c}}\Big[\sum_{i=1}^{m} R(c_i)I_{(a=i)}\,\Big|\, a\sim\pi(\mathbf{c})\Big] \\ &= \sum_{i=1}^{m}\mathbb{E}_{\mathbf{c}}\Big[R(c_i)I_{(a=i)}\,\Big|\, a\sim\pi(\mathbf{c})\Big] \\ &= \sum_{i=1}^{m}\mathbb{E}_{c_1}\Big[R(c_1)\mathbb{E}_{c_{-1}}\Big[I_{(a=1)} \mid a\sim\pi(c,c_{-1})\Big]\Big] \\ &= m\,\mathbb{E}_c\Big[R(c)p(c)\Big], \end{aligned} \tag{9}$$

where $p(c)$ is the marginal probability of choosing an item with context c (in expectation of randomness of the other $m-1$ items, denoted as c_{-1}), by a permutation invariant policy:

$$p(c) = \mathbb{E}_{c_{-1}}\Big[I_{(a=1)} \mid a\sim\pi(c,c_{-1})\Big]. \tag{10}$$

Suppose we have a score function μ_θ which takes the context as inputs and outputs a score, where θ are the parameters. We can construct a class $\mathcal{M}$ of permutation invariant policies with the score function:

$$\pi_\theta(\mathbf{c}) \overset{\text{dist}}{=} g\big(\mu_\theta(c_1),\ldots,\mu_\theta(c_m)\big), \tag{11}$$

where g is an operator that satisfies permutation invariance, for example, a family of probability distributions, and $\overset{\text{dist}}{=}$ means the two sides are equivalent in the sense of probability distribution.

Note that this class of policies include policies of most well-known value-based bandit algorithms. For example, if the score function is the estimation of the reward, and the operator $g(\cdot)$ chooses the item with the maximum estimated reward with probability $1-\epsilon$ and chooses randomly with probability ϵ, the policy is exactly the well-known ϵ-greedy policy [24]. If the score function is a summation of the reward estimation and the upper confidence bound, and $g(\cdot)$ chooses the item with the maximum score, it results in a similar policy to the upper confidence bound (UCB) policy [5, 17].

The policy gradient $\nabla_\theta J(\pi_\theta)$ for the standard one-step recommendations can be directly derived from (9)

$$\nabla_\theta J(\pi_\theta) = m\,\mathbb{E}_c\Big[R(c)\nabla_\theta p_\theta(c)\Big]. \tag{12}$$

So it involves computing the marginal probabilities of choosing an item $p(c)$ which is not explicitly parameterized or tractable given an arbitrary policy π_θ. To this end, we put forward a restricted family of stochastic policies that allows us to have $p(c)$ in closed-form so as to estimate the gradient of $J(\pi_\theta)$ efficiently.

3.2 A simple but powerful class of policies

Now we propose a class of policies for our PGCR algorithm. For the standard one-step contextual-bandits, following the form of a policy described in (11), we define a class of stochastic policies denoted by $\mathcal{N}$ as

$$\pi_\theta(\mathbf{c}) = \text{Multinoulli}\Big\{\sigma\big(\mu_\theta(c_1),\ldots,\mu_\theta(c_m)\big)\Big\}, \tag{13}$$

where $\mu_\theta(c)$ is any positive continuous function of c and θ, σ is a normalization $\sigma(\mathbf{x}) := \Big(\frac{x_1}{\sum_i x_i},\ldots,\frac{x_m}{\sum_i x_i}\Big)$ and Multinoulli$(\cdot)$ returns a multinoulli random variable (the multinoulli distribution is also known as categorical distribution).

The form of our policy (13) generalizes several important policies in reinforcement learning. For example, when $\mu_\theta(c_i)$ is an exponential function $e^{-\beta f_\theta(c_i)}$, it reduces to the well-known *softmax policy*. If β approaches to infinity, it converges to the greedy policy that chooses the item with highest score.

For any policy $\pi_\theta \in \mathcal{N}$, the marginal probability of choosing an item can be easily derived as

$$p_\theta(c_i) = \mathbb{E}_{c_{-i}}\Big[\frac{\mu_\theta(c_i)}{\mu_\theta(c_i)+\sum_{j\neq i}\mu_\theta(c_j)}\Big], \tag{14}$$

which is a continuous and positive function of parameters θ. So it is straightforward to estimate $p_\theta(c)$ by sampling c_{-i} from the contexts that the player have seen before. We denote by D_t the contexts that have appeared up to step t. The estimation of $p(c)$ is unbiased because as assumed all contexts in D_t are i.i.d from the context space.

4 POLICY GRADIENTS FOR SEQUENTIAL CONTEXTUAL RECOMMENDATIONS

In this section we extend this approach to MDP-CR. We first use $\tilde{c}$ to denote the *augmented context* by pairing together a state s and the context c for a certain item, i.e., $\tilde{c} = (s,c)$. For example, in a personalized recommender system, the state s consists of the user feature, c contains the item feature. Their concatenation $\tilde{c} = (s,c)$ is often used in ordinary tasks such as the prediction of click-through rate.

Given a policy π, the states can be roughly thought of as drawn from the discounted stationary distribution $\rho^\pi(s)$. As we already defined the density of contexts given the state as $g^s(c)$, we have the discounted density of the augmented context $\tilde{c}$ by the axiom of probability

$$\xi^\pi(\tilde{c}) = \rho^\pi(s)\,g^s(c).$$

Since we assume the state distribution $\rho^\pi(s)$ is stationary, it is natural that $\xi^\pi(\tilde{c})$ is also stationary.

Then by applying the same technique as we derive the marginal probability, we derive the performance objective as follows:

$$J(\pi) = m\,\mathbb{E}_{\tilde{c}\sim\xi^{\pi}}\Big[R(\tilde{c})\cdot p(\tilde{c})\Big], \tag{15}$$

where $p(\tilde{c}) = \mathbb{E}_{c_{-1}\sim g^{s}}[I(a=1)\mid\pi]$ is the marginal probability of choosing c given s.

Now we derive the gradients of $J(\pi)$ similar to the result in [25]. Surprisingly, it only replaces $R(c)$ in (12) by $Q(\tilde{c})$.

Theorem 2. *Assuming the policy π leads to stationary distributions for states and contexts, the unbiased policy gradient is*

$$\nabla_{\theta}J(\pi_{\theta}) = m\,\mathbb{E}_{\tilde{c}\sim\xi}\Big[\nabla_{\theta}p_{\theta}(\tilde{c})\,Q(\tilde{c})\Big], \tag{16}$$

where $Q^{\pi}(\tilde{c}) := Q^{\pi}(s,c)$ as defined in (3), and $\xi^{\pi}(\tilde{c})$ is the discounted density of $\tilde{c}$.

Proof. We denote the state-value for state s under policy π as

$$V^{\pi}(s) = m\int_{c} p_{\theta}(s,c)Q^{\pi}(s,c)g^{s}(c)\mathrm{d}c,$$

it follows that

$$\begin{aligned}\nabla_{\theta}V^{\pi}(s) &= \nabla_{\theta}m\int_{c} p_{\theta}(s,c)Q^{\pi}(s,c)g^{s}(c)\mathrm{d}c\\ &= m\int_{c}\nabla_{\theta}p_{\theta}(s,c)Q^{\pi}(s,c)g^{s}(c)\mathrm{d}c\\ &\quad+\gamma m\int_{s'}P(s\to s',1,\pi)\nabla_{\theta}V^{\pi}(s')\mathrm{d}s',\end{aligned}$$

By repeatedly unrolling the equation, we have

$$\nabla_{\theta}V^{\pi}(s) = m\int_{s'}\sum_{t=0}^{\infty}\gamma^{t}P(s\to s',t,\pi)\int_{c}\nabla_{\theta}p_{\theta}(\tilde{c})Q(\tilde{c})g^{s'}(c)\mathrm{d}c\mathrm{d}s'.$$

Integrating both side over the start-state and recalling the discounted state density $\rho^{\pi}(s)$ and discounted augmented context density $\xi^{\pi}(\tilde{c})$, we get the policy gradient as

$$\begin{aligned}\nabla_{\theta}J(\pi) &= m\int_{s}P_{0}(s)\nabla_{\theta}V^{\pi}(s)ds\\ &= m\int_{s}\rho^{\pi}(s)\int_{c}\nabla_{\theta}p_{\theta}(\tilde{c})Q(\tilde{c})g^{s}(c)\,\mathrm{d}c\mathrm{d}s\\ &= m\int_{\tilde{c}}\nabla_{\theta}p_{\theta}(\tilde{c})Q(\tilde{c})\xi^{\pi}(\tilde{c})\mathrm{d}\tilde{c}.\end{aligned}$$

□

Again we can find the restricted class of policies to perform efficient estimation of the gradient. For MDP-CR, it is straightforward to extend (13) to introduce states by replacing c as the augmented context $\tilde{c}$.

5 ACTOR-CRITIC WITH FUNCTION APPROXIMATIONS

In this section, we deliver the practical algorithm to estimate the policy gradient using function approximations. In the sample-based approximations, we write collected the reward feedback as $r(c)$ for the one-step recommendation case, and $r(\tilde{c})$ for the sequential recommendation case, as the realizations of the reward random variable $R(\cdot)$.

In conventional methods based on contextual-bandits, the most direct way to estimate the reward function is to directly apply supervised learning methods to find an estimator f_{ϕ} with parameter ϕ minimizing the mean squared error, i.e.,

$$\min_{\phi}\frac{1}{|D_{t}^{(1)}|}\sum_{c\in D_{t}^{(1)}}\Big(r(c)-f_{\phi}(c)\Big)^{2}, \tag{17}$$

where $D_{t}^{(1)}\subset D_{t}$ is the set of chosen contexts and $r(c)$ is the received reward for choosing context c. It is common for most value-based contextual bandit algorithms, such as ϵ-greedy, LinUCB and Thompson Sampling.

However, we argue that in a policy-based perspective, this kind of off-policy supervised learning brings bias. Since our goal is to maximize the expected reward $J(\pi_{\theta})$ rather than minimizing the empirical loss as in supervised learning, the marginal probabilities of choosing an item must be taken into consideration, and the form of $f_{\phi}(c)$ cannot be chosen arbitrarily.

Similarly, when states and state transitions are involved in MDP-CR, we also need to find an approriate $f_{\phi}(\tilde{c})$ to approximate $Q(\tilde{c})$. Since we have noted that the standard one-step contextual-bandits can be seen as a special case of the generalized MDP-CR, from now we will use the notations of MDP-CR by default to deliver the main results and algorithms.

We take similar spirit to [21, 25] and define the following *compatible* conditions, to assure that the policy gradient is orthogonal to the error in value approximation.

Theorem 3. *The policy gradient using function approximation*

$$\nabla_{\theta}J(\pi_{\theta}) = m\int_{\tilde{c}}\nabla_{\theta}p_{\theta}(\tilde{c})\cdot f_{\phi}(\tilde{c})\xi^{\pi}(\tilde{c})d\tilde{c} \tag{18}$$

is unbiased to (16) if the following conditions are satisfied:

(i) the gradients for the value function and the policy are compatible,

$$\nabla_{\phi}f_{\phi}(\tilde{c}) = \nabla_{\theta}\,log\,p_{\theta}(\tilde{c}), \tag{19}$$

(ii) the value function parameters ϕ reach a local minimum of the mean squared error over the stationary context distribution such that

$$\nabla_{\phi}\,\mathbb{E}_{\tilde{c}\sim\xi^{\pi}}\Big[p_{\theta}(\tilde{c})\Big(f_{\phi}(\tilde{c})-Q^{\pi}(\tilde{c})\Big)^{2}\Big] = 0. \tag{20}$$

Proof. By condition (ii), as we assumed the distribution of contexts ξ^{π} is stationary with respect to the policy π, it is easy to see when the conditions hold,

$$m\int_{\tilde{c}}\xi^{\pi}(\tilde{c})p_{\theta}(\tilde{c})[Q^{\pi}(\tilde{c})-f_{\phi}(\tilde{c})\Big]\nabla_{\phi}f_{\phi}(\tilde{c}) = 0.$$

Then by condition (i) we have

$$m\int_{\tilde{c}}\nabla_{\theta}p_{\theta}(\tilde{c})\Big[Q^{\pi}(\tilde{c})-f_{\phi}(\tilde{c})]\xi^{\pi}(\tilde{c})\mathrm{d}\tilde{c} = 0,$$

which is the difference between (16) and (18). □

5.1 The basic PGCR algorithm

We now formally propose the policy gradients algorithm for general contextual recommendations, coined by PGCR. Recall that our policy returns a Multinoulli random variable that chooses a_t by

$$a_t \sim \text{Multinoulli}\Big\{\sigma\Big(\mu_{\theta}(s_t,c_{t1}),\ldots,\mu_{\theta}(s_t,c_{tm})\Big)\Big\}.$$

The key is to estimate the marginal expected probabilities for each item. When estimating it for some context, say c_{ti}, at some state s_t,

we resample from all the previously observed contexts at the same state to get another $m-1$ contexts.

$$\hat{p}_\theta(s_t, c_{ti}) = \frac{1}{N}\sum_{n}^{N} \frac{\mu_\theta(s_t, c_{ti})}{\mu_\theta(s_t, c_{ti}) + \sum_c \mu_\theta(s_t, c)} \tag{21}$$

where N ($N \geq 1$) is the number of resampling times, and c in the denominator are another $m-1$ sampled contexts from the same state to s_t.

Similar to previous actor-critic algorithms [18], we can use Sarsa updates [24] to estimate the action-value function and then update the policy parameters respectively by the following *policy gradients for contextual-bandits* algorithm,

$$\delta_t = r_t + \gamma f_{\phi_t}(s_{t+1}, c_{(t+1)a}) - f_{\phi_t}(s_t, c_{ta}) \tag{22}$$

$$\Delta^{\text{PGCR}}_{\phi,t} = \hat{p}_\theta(s_t, c_{ta})\delta_t \nabla_\phi f_{\phi_t}(s_t, c_{ta}) \tag{23}$$

$$\phi_{t+1} = \phi_t + \alpha_\phi \Delta^{\text{PGCR}}_{\phi,t} \tag{24}$$

$$\Delta^{\text{PGCR}}_{\theta,t} = \sum_{i=1}^{m} \nabla_\theta \hat{p}_{\theta_t}(s_t, c_{ti}) f_{\phi_{t+1}}(s_t, c_{ti}) \tag{25}$$

$$\theta_{t+1} = \theta_t + \alpha_\theta \Delta^{\text{PGCR}}_{\theta,t}. \tag{26}$$

In practice, the gradients can be updated on mini-batches by modern optimizers such as the Adam optimizer [14] which we already used for experiments. PGCR naturally fits to deep Reinforcement Learning and Online Learning, and techniques from these area may also be applied. Note that the algorithm can also be apply to the standard contextual bandit setting without states.

5.2 Two useful heuristics

5.2.1 Time-Dependent Greed. Greedy in the Limit with Infinite Exploration (GLIE), is the basic criteria desired for bandit algorithms. GLIE is to explore all the actions infinite times and then to converge to a greedy policy that reaches the global optimal reward if it runs for enough time. Value-based methods can satisfy GLIE if a positive but diminishing *exploration value* is given to all the actions. But for policy-based methods, it is not straightforward, because one cannot explicitly show the exploration level of a stochastic policy.

For PGCR, on the contrary, it is easy to have GLIE by *Time-Dependent Greed*, which applies a Time-Dependent Greed factor to the scoring function $\mu(c)$. A straightforward usage is to let $\mu_{\text{greedy}}(c; t) := \mu^{\alpha t}(c)$ where α is a pre-determined positive constant value and t is the current time-step. When $t \to \infty$, the policy tends to choose only the item with the largest score. Also the marginal probability $p_{\text{greedy}}(c; t)$ remains positive with the assumption that $\mu(c) \in (0, +\infty)$ for all context c, so any item gets an infinite chance to be explored if it runs for enough time. This technique can also apply to other policy-based RL methods as well.

5.2.2 Actor-Dropout. Directed exploration is also desired. UCB and TS methods are well-known to have directed exploration so that they automatically trade-offs between exploration and exploitation and get sub-linear total regrets. The basic insight of UCB and TS is to learn the model uncertainty during the online decision-making process and to explore the items with larger uncertainty (or higher potential to get a large reward). Often a Bayesian framework is used to model the uncertainty by estimating the posterior. However, the limitation of these methods is that assumptions and prior knowledge of reward functions are required, otherwise the posterior cannot be estimated correctly.

In light of these observations, we propose *Actor-Dropout* for PGCR to achieve directed exploration. The method is simple: to use dropout on the policy network while training and inferring. We do so because it has been theoretically and empirically justified that a neural network with dropout is mathematically equivalent to a Bayesian approximation [11]. So Actor-Dropout naturally learns the uncertainty of policies and does Monte Carlo sampling when making decisions. Since Actor-Dropout needs no prior knowledge, it can apply to more general and complex cases than UCB and TS.

To use Actor-Dropout, in practice it is good enough for exploration to add dropout to just one layer of weights. For example, for a fully-connected actor-network, one can use dropout to the weights before the output layer, with a dropout ratio of 0.5 or 0.67. It can be understood as to train several actors and to randomly pick one at each step, so it trade-offs between exploration and exploitation since each actor learns something different from each other. We also found Actor-Dropout worth trying for other RL or Online Learning tasks in the exploration phase.

5.3 Lower variance of the gradients of PGCR than vanilla PG

We prove that the variance of updating the actor and the critic of PGCR is less than that of vanilla PG.

Since the concept of *context* does not exist in the classic formulation of reinforcement learning, it is often regarded as part of the state. Given a stochastic policy $\pi_\theta(s, \mathbf{c})$, PG has policy gradients

$$\nabla_\theta J(\pi_\theta) = m \sum_s \rho^\pi_s \sum_{i=1}^{m} \nabla_\theta [e_i^T \pi_\theta(s, \mathbf{c})] \cdot f_\phi(\tilde{c}_i), \tag{27}$$

where e_i denotes a unit vector and $e_i^T \pi_\theta(s, \mathbf{c})$ is the probability for choosing the i^{th} item. For simplicity, we write $v_i := e_i^T \pi_\theta(s, \mathbf{c})$. Since we focus on policy gradients, we assume that PG has a critic function $f_\phi(\tilde{c})$ with the same form as PGCR. The corresponding update steps for PG is

$$\Delta^{\text{PG}}_{\phi_t} = v_{ta}\delta_t \nabla_\phi f_\phi(s_t, c_{ta}) \tag{28}$$

$$\Delta^{\text{PG}}_{\theta_t} = \sum_{i}^{m} \nabla_\theta v_{ti} f_{\phi_{t+1}}(s_t, c_{ti}). \tag{29}$$

Our PGCR can achieve lower estimation variance comparing to classic stochastic policy gradient methods such as [25], the reasons are two-fold. Firstly, by Lemma 1 we know permutation invariant policies are sufficient for contextual-bandits problems, so PGCR adopts the restricted class of policies. On the contrary, in vanilla policy gradients, one should treat a state s and the whole contexts $\mathbf{c}$ altogether as the input of the policy function, so the sample space can be much larger, which results in lower sample efficiency. Secondly, even if with the same form of policy, vanilla policy gradients tend to converge slower than PGCR because they do not take the marginal expected probabilities of choosing an item into consideration.

In a formal way, we can have the following conclusion.

Lemma 4. *Given a policy $\pi_\theta \in \mathcal{N}$ and a value approximation f_ϕ, both $\Delta_{\phi_t}^{PGCR}$ and $\Delta_{\phi_t}^{PG}$ are unbiased estimators for the true gradients of action-value approximation*

$$\Delta_{\phi_t} = p(s_t, c_{ta})\delta_t \nabla_\phi f_\phi(s_t, c_{ta}). \tag{30}$$

And $Var\left[\Delta_{\phi_t}^{PGCR}\right] \leq Var\left[\Delta_{\phi_t}^{PG}\right]$. Additionally if PGCR uses a fixed N, as $t \to +\infty$, with probability 1 we have

$$Var\left[\Delta_{\phi_t}^{PGCR}\right] \to \frac{1}{N} Var\left[\Delta_{\phi_t}^{PG}\right]. \tag{31}$$

Proof. It is obvious that both v_{ta} in $\Delta_{\phi_t}^{PG}$ and $\hat{p}(s_t, c_{ta})$ in $\Delta_{\phi_t}^{PGCR}$ are unbiased to $p(s_t, c_{ta})$. So both $\Delta_{\phi_t}^{PGCR}$ and $\Delta_{\phi_t}^{PG}$ are unbiased to Δ_{ϕ_t}. To analyze the variance, we focus on the estimations of the probability of choosing an item: v_{ta} and $\hat{p}(s_t, c_{ta})$. Let $V := \mathrm{Var}\left[v_{ta}\right]$. Then for PGCR,

$$\mathrm{Var}\left[\hat{p}(s_t, c_{ta})\right] = \mathrm{Var}\left[\frac{1}{N}\sum_{n=1}^{N} v_{ta}^{(n)}\right], \tag{32}$$

where $v_{ta}^{(n)}$ denotes the probability of choosing c_{ta} at the n^{th} time of sampling. In the worst case, it samples exactly the same set of $m-1$ items every time, then $\mathrm{Var}\left[\hat{p}(s_t, c_{ta})\right] = V$. Otherwise if there exists n_1 and n_2 that the samples are different such that $v_{ta}^{(n_1)} \neq v_{ta}^{(n_2)}$, then the correlation is strictly less than 1 and we have $\mathrm{Var}\left[\Delta_{\phi_t}^{PGCR}\right] < \mathrm{Var}\left[\Delta_{\phi_t}^{PG}\right]$ in this case. Finally when enough time steps passed, for N is a fixed positive integer, the probability of each item being sampled at most once is

$$\binom{mt}{(m-1)N}(mt)^{-(m-1)N} \to 1 \quad \text{as} \quad t \to +\infty.$$

So with probability 1 the sampled contexts are all different to each other so the estimated probabilities of choosing an item are i.i.d., then $\mathrm{Var}\left[\Delta_{\phi_t}^{PGCR}\right] \to V/N$. □

We get the following theorem applying the similar technique to Lemma 4. We claim that Policy gradients (18) has no higher variance than gradients in PG.

Theorem 5. $Var\left[\Delta_\theta^{\mathrm{PGCR}}\right] \leq Var\left[\Delta_\theta^{\mathrm{PG}}\right]$.

Proof. Similar to the proof of Lemma 4, we denote the variance of $\Delta_\theta^{\mathrm{PG}}$ by V_θ.

$$V_\theta := \mathrm{Var}\left[\Delta_\theta^{\mathrm{PG}}\right] = \mathrm{Var}\left[\sum_{i}^{m} \nabla_\theta v_{ti} f_{\phi_{t+1}}(s_t, c_{ti})\right] \tag{33}$$

By the update rules (25) of PGCR, the variance of $\Delta_\theta^{\mathrm{PGCR}}$ is

$$\begin{aligned}
\mathrm{Var}\left[\Delta_{\theta_t}^{\mathrm{PGCR}}\right] &= \mathrm{Var}\left[\sum_{i=1}^{m} \nabla_\theta \hat{p}_\theta(s_t, c_{ti}) f_{\phi_{t+1}}(s_t, c_{ti})\right] \\
&= \mathrm{Var}\left[\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{m} \nabla_\theta v_{ti}^{(n)} f_{\phi_{t+1}}(s_t, c_{ti})\right] \\
&\leq \mathrm{Var}\left[\sum_{i=1}^{m} \nabla_\theta v_{ti}^{(n)} f_{\phi_{t+1}}(s_t, c_{ti})\right], \quad \forall\, n = 1, \ldots, N.
\end{aligned}$$

Because of the assumption that the sampled contexts in each sampling procedure are independent and identical distributed, we have $\mathrm{Var}\left[\sum_{i=1}^{m} \nabla_\theta v_{ti}^{(n)} f_{\phi_{t+1}}(s_t, c_{ti})\right] = V_\theta$ for all $n = 1, \ldots, N$ and the theorem is proved. □

Note that, in practice, PGCR does not necessarily set N to a large integer since it is naturally a finer-grained experience replay [3]. Surprisingly, when $N = 1$, PGCR can have a better performance than *PG* even in the simplest setting. In the next section, we will demonstrate experimental results that show that PGCR with $N = 1$ achieves better performance in various settings compared to other baseline methods including PG.

The results can be interpreted as follows. From a statistical point of view, PGCR takes advantage from a resampling technique so the estimations have lower variances. From an optimization perspective, PGCR reduces the correlation of estimating probabilities of choosing the m items within the same time step, so it has less chance to suffer from exploiting and over-fitting, while vanilla PG cannot. For example, when the estimated values of m contexts are given, an optimizer for PG would simultaneously increase one item's chosen probability and reduce other $m-1$ ones', which results in training the policy into a deterministic one: the item with the largest estimated value will get a chosen probability close to 1, and others get arbitrary small probabilities close to 0. Afterward, the items with 0 chosen probabilities will hardly have any influence to further updates. So eventually, vanilla PG is likely to over-fit the existing data. On the contrary, when PGCR estimates the gradients, even if an item cannot beat against the other $m-1$ competitors at its own time step, it can still help because it outranked some items from other time steps. Therefore, PGCR tends to be more robust and explores better than vanilla PG.

6 EXPERIMENTS

6.1 Datasets and simulation details

We test our proposed PGCR and other baseline methods on several simulated contextual recommendation environments. To start with, we would like to describe the details of our simulations.

- **Toy environments:** The first set of experiments are done with a generated toy dataset to simulate the standard contextual bandit settings. We simulate a contextual bandit environment with 5 items at each step, where each item is represented by a 40-dimensional context vector that is i.i.d. sampled from a uniform distribution in a unit cube $c \sim \mathrm{U}(C)$, $C = (0,1)^{40}$. Once an item with context c is chosen by the player, the environment returns a random reward $R(c)$. We test three types of reward functions: (a) the linear reward with Gaussian noise, as $R(c) := w_r^T c + e_r$; (b) the Bernoulli reward, as $R(c) \sim \mathrm{Bernoulli}(\beta(c))$ where $\beta(c) := w_\beta^T c + e_\beta \in [0,1]$ is the probability to return reward 1 for choosing c; (c) the mixed reward, which first returns a random linear reward $w_r^T c + e_r$ with probability $\beta(c)$ and returns a zero reward with probability $1-\beta(c)$, as a mixture of binary and linear rewards. w_r and w_β are coefficients unknown to the player. e_r and e_β are white noises to introduce some randomness.

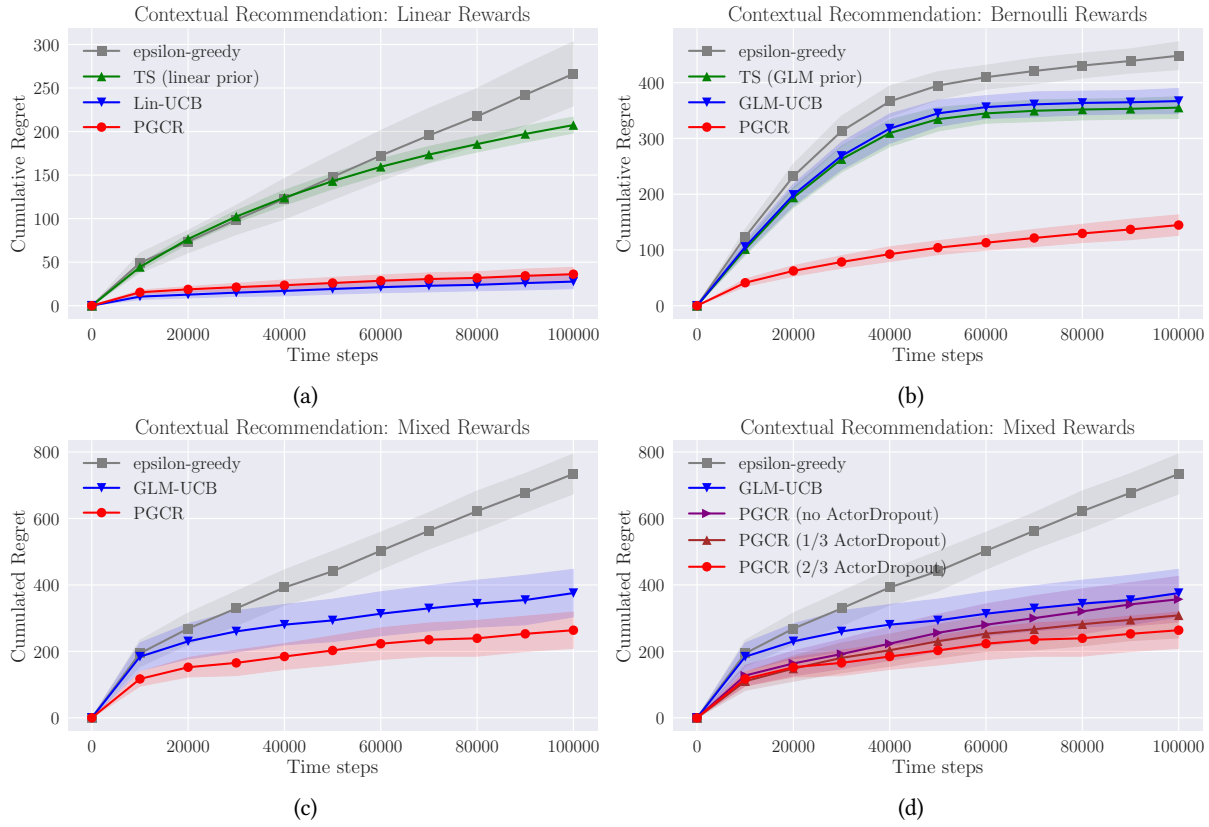


Figure 1: Experiments on toy data. The solid lines are averaged cumulative regrets over 20 runs, and the shaded areas stand for the standard deviations. (a) Linear rewards: PGCR perform comparably to Lin-UCB and is much better than the other two. (b) Bernoulli rewards: PGCR outperforms the others with large margins. (c) Mixed-rewards: PGCR outperforms the others. (d) Testing PGCR with different level of Actor-Dropout. It shows that, without Actor-Dropout, PGCR tends to have linear regret like ϵ-greedy and fail to converge. But when using Actor-Dropout, PGCR empirically converges which is similar to UCB, yet gets even lower regret than UCB.

- **Music recommendation environments:** We use a real-world dataset of music recommendation provided by *KKBox* and open-sourced on *Kaggle.com*[1]. The challenge is to predict the chances of a user listening to a song repeatedly. So we construct simulation environments to simulate the online contextual recommendations to test our methods.
  We construct two simulators based on the distributions of the dataset with different settings: one simplified setting without explicit states and one general setting with states and state transitions. At each round, a user comes, which is randomly picked from the users in the dataset. For the state-aware setting, the state is set as the last 3 songs the system recommended previously to this user, together with the corresponding feedbacks (listened or not). Then the simulator randomly samples a set of 10 candidate songs from the user's listening history. Finally, the recommender makes a decision to recommends one to the user. If the user listens to it again (which is the original target for the supervised learning dataset), the system gets a reward of 1, otherwise, it gets 0. Each song has 94 fields of features in the context vector, including discrete attributes and numerical features about the song's genre, artists, composers, language, etc. The simulation consists of 5 million time steps and is repeated for 5 runs. Since the optimal policy is unknown in this problem, we will use the average reward as the performance metric.

[1]https://www.kaggle.com/c/kkbox-music-recommendation-challenge/

6.2 Experiments on toy datasets

For the three standard contextual-bandit problems, we use the cumulative regret as the evaluation metric. The cumulative regret is defined as the cumulative difference between the received reward and the reward of the optimal item.

For PGCR, we use the multi-layer perceptron (MLP) as the policy and value networks. The MLPs have one fully-connected hidden layer with 10 units and the ReLU activation. As for the training details, at each step, we sample a batch of 64 history samples for efficient experience replay. The loss is minimized by the gradient-based Adam optimizer [14].

We compare PGCR with the following algorithms:

- **ϵ-greedy**: It chooses the item with the largest estimated value with a probability of $1-\epsilon$ and chooses randomly otherwise. Specifically, it estimates the expected reward by a value network with the same structure of PGCR.
- **Lin-UCB**: The widely studied version of UCB for contextual bandits with linear rewards [1, 9, 17], which uses a linear function to approximate the reward, and chooses the item with the maximum sum of the estimated reward and the estimated confidence bound.
- **GLM-UCB**: The UCB method for generalized linear rewards, proposed in [10], which can solve non-linear rewards if the reward function can be fitted by a generalized linear model of contexts, such as Bernoulli rewards and Poisson rewards.
- **Thompson Sampling**: It samples from the posterior distribution of parameters, estimates each item's value, and chooses the item with the maximum estimation [4, 8]. Specifically, it uses the same function approximation as Lin-UCB or GLM-UCB for linear and non-linear rewards.

For Lin-UCB, GLM-UCB and Thompson Sampling, we use the same training procedures as suggested in [17]. For ϵ-greedy, ϵ is set as a constant value, e.g. 0.1 or 0.3.

For each method, in order to reduce the randomness of experiments, we run the simulation for 20 times and report their averaged cumulative regrets. The results are shown in Figure 1.

From Figure 1(a), (b) and (c), it sees that PGCR converges more quickly and has lower regrets in most cases. For the linear bandits, Lin-UCB is, theoretically, one of the best choice if the linearity is known. Even though PGCR does not know the form of reward function apriori, the performance is comparable with Lin-UCB. Thompson Sampling converges slower and ϵ-greedy obviously fails to converge. For the generalized linear case, PGCR achieves much lower regret than other baselines. For the mixed reward case, we see that PGCR learns faster than GLM-UCB and empirically converges after a long run.

We further did an ablation study to test the improvements induced by the proposed heuristic, Actor-Dropout. See Figure 1(d) for the comparisons. It shows that Actor-Dropout significantly helps PGCR to converge. The growth rate of cumulative regret for PGCR without Actor-Dropout is similar to ϵ-greedy, indicating that the original algorithm fails to converge and has linear regrets. But when equipped with Actor-Dropout, the regrets are smaller. When the dropout rate is set to 0.67, the growth rate of PGCR's regret is similar to that of GLM-UCB which can theoretically achieve sub-linear regrets. So empirically we remark that Actor-Dropout is a strong weapon for PGCR in order to get a convergence guarantee, even with almost no assumptions on the problem.

6.3 Experiments on music recommendations

6.3.1 Simplified setting without explicit states. The experimental setup in the setting without states is as follows: PGCR uses MLPs as policy and value networks. Each MLP has two hidden layers of sizes 60 and 20 respectively and uses the ReLU nonlinearity. ϵ-greedy has exactly the same value network like the one in PGCR. All these methods are trained with Adam algorithm with the same learning rates. The batch-size is set as 256. We also test GLM-UCB here as another baseline because the reward is binary thus can be learned by a logistic regression model.

As is shown in Figure 2(a), PGCR performs the best. It is interesting to see that traditional contextual-bandits methods learn fast from the beginning, which indicates that they are good at exploration, but their average rewards stop increasing rapidly due to the limitation of the fitting power of general linear models. ϵ-greedy learns slowly at the early stage, but it outperforms GLM-UCB and TS after a long run. Comparing with these algorithms, PGCR has the best performance from the beginning to the end of the learning process.

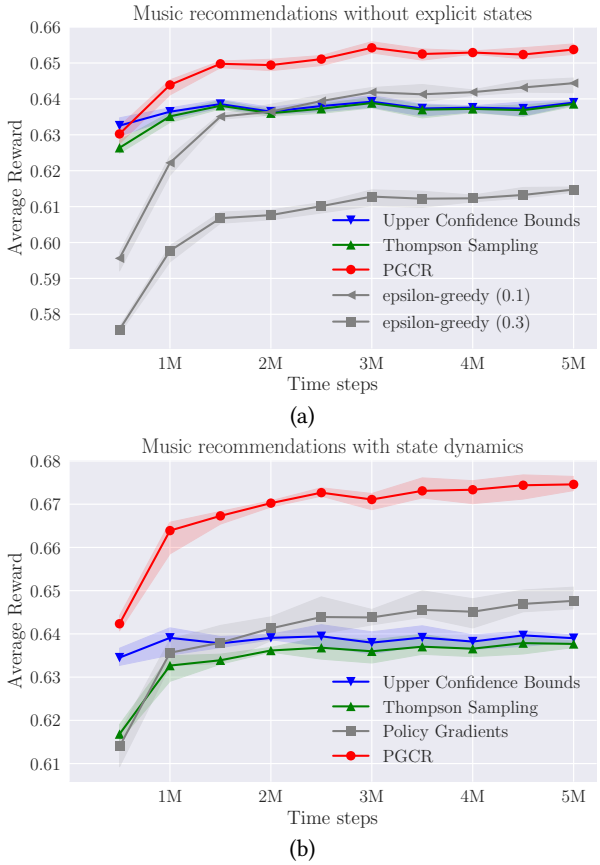


Figure 2: Average rewards of episodes for (a) music recommender without explicit states; (b) music recommender with states and state transitions. The solid lines are averaged from 5 runs, and the shaded areas are standard deviations.

6.3.2 Simulation with states and transitions. The experimental setup for this experiment with states is as follows. We enlarge the size of the first hidden layer of the MLPs from 60 to 90 because there are more inputs (including the contexts and the states). UCB and TS here take the augmented contexts as input. They keep the same general linear modeled priors as the previous part.

The result of the experiment is shown in figure 2(b). PGCR outperforms other algorithms with larger map comparing with the previous experiment. An interesting fact is that both UCB and TS

can only get almost the same return as in the previous experiment, which indicates that they can hardly make any use of the information from the states. PGCR learns faster and gets state-of-the-art performance in this task.

Since there are states and transitions, we also test the vanilla Policy Gradient method [25] as another baseline, with the same MLP neural networks as actor and critic as PGCR. From the result, it shows that vanilla PG can outperform classic bandits methods, which is not surprising because it can make use of the state dynamics and maximize the long-term return. However, it is still much worse than our purposed PGCR, for the reason that PGCR has a smaller search space and smaller variance when estimating the policy gradients. So the experimental results verified that PGCR is more sample efficient than the vanilla policy gradient method.

These simulation experiment results indicate that PGCR provides an alternative to conventional methods by using stochastic policies which can address the trade-off between exploration and exploitation well. The fast learning in the beginning phase and the stable performance over the entire training period of PGCR show that it is reliable to apply for real-world recommender systems.

7 CONCLUSION AND DISCUSSION

This paper has studied how to use the actor-critic algorithm with neural networks for general contextual recommendations, without unrealistic assumptions or prior knowledge to the problem. We first show that the class of permutation invariant policies is sufficient for this problem, and then derive the expected return of a policy which depends on its marginal expected probability of choosing each item. We next propose a restricted class of policies in which the objective has a simple closed form and is differentiable to parameters. When using such policies, the gradient can be computed in closed-form. Furthermore, we propose Time-Dependent Greed and Actor-Dropout to improve the performance and to guarantee the convergence. Eventually, it comes to our proposed PGCR algorithm. It can be applied to standard contextual bandits as well as generalized sequential decision-making problems with state dynamics.

By testing on a toy dataset and a recommendation dataset, we showed that PGCR indeed achieves state-of-the-art performance for both classic one-step recommendations and MDP-CR with state transitions in a real-world scenario.

It is a promising direction to extend our results to a variant of realistic recommendation settings, i.e, online advertising systems that choose multiple items at each round, or learning-to-rank that carries out diverse recommendation.

ACKNOWLEDGEMENT

The work was supported by the National Key Research and Development Program of China under Grant No. 2018YFB1004300, the National Natural Science Foundation of China under Grant No. U1836206, U1811461, 61773361, the Project of Youth Innovation Promotion Association CAS under Grant No. 2017146.

Pingzhong Tang and Qingpeng Cai were supported in part by the National Natural Science Foundation of China Grant 61561146398, a China Youth 1000-talent program and an Alibaba Innovative Research program.

REFERENCES

[1] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. 2011. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*. 2312–2320.

[2] Naoki Abe, Alan W Biermann, and Philip M Long. 2003. Reinforcement learning with immediate rewards and linear hypotheses. *Algorithmica* 37, 4 (2003), 263–293.

[3] Sander Adam, Lucian Busoniu, and Robert Babuska. 2012. Experience replay for real-time reinforcement learning control. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42, 2 (2012), 201–212.

[4] Shipra Agrawal and Navin Goyal. 2013. Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*. 127–135.

[5] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning* 47, 2-3 (2002), 235–256.

[6] Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gançarski. 2012. A contextual-bandit algorithm for mobile context-aware recommender system. In *International Conference on Neural Information Processing*. Springer, 324–331.

[7] Qingpeng Cai, Aris Filos-Ratsikas, Pingzhong Tang, and Yiwei Zhang. 2018. Reinforcement Mechanism Design for e-commerce. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1339–1348.

[8] Olivier Chapelle and Lihong Li. 2011. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*. 2249–2257.

[9] Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. 2011. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 208–214.

[10] Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. 2010. Parametric bandits: The generalized linear case. In *Advances in Neural Information Processing Systems*. 586–594.

[11] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*. 1050–1059.

[12] Nicolas Heess, Jonathan J Hunt, Timothy P Lillicrap, and David Silver. 2015. Memory-based control with recurrent neural networks. *arXiv preprint arXiv:1512.04455* (2015).

[13] Thomas Jaksch, Ronald Ortner, and Peter Auer. 2010. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research* 11, Apr (2010), 1563–1600.

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Andreas Krause and Cheng S Ong. 2011. Contextual gaussian process bandit optimization. In *Advances in Neural Information Processing Systems*. 2447–2455.

[16] John Langford and Tong Zhang. 2008. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*. 817–824.

[17] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*. 661–670.

[18] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

[19] Benedict C May, Nathan Korda, Anthony Lee, and David S Leslie. 2012. Optimistic Bayesian sampling in contextual-bandit problems. *Journal of Machine Learning Research* 13, Jun (2012), 2069–2106.

[20] Guy Shani, David Heckerman, and Ronen I Brafman. 2005. An MDP-based recommender system. *Journal of Machine Learning Research* 6, Sep (2005), 1265–1295.

[21] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *ICML*.

[22] Aleksandrs Slivkins, Filip Radlinski, and Sreenivas Gollapudi. 2013. Ranked bandits in metric spaces: learning diverse rankings over large document collections. *Journal of Machine Learning Research* 14, Feb (2013), 399–436.

[23] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias W Seeger. 2012. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory* 58, 5 (2012), 3250–3265.

[24] Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction*. Vol. 1. MIT press Cambridge.

[25] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*. 1057–1063.

[26] Nima Taghipour and Ahmad Kardan. 2008. A hybrid web recommender system based on q-learning. In *Proceedings of the 2008 ACM symposium on Applied computing*. ACM, 1164–1168.

[27] Liang Tang, Yexi Jiang, Lei Li, and Tao Li. 2014. Ensemble contextual bandits for personalized recommendation. In *Proceedings of the 8th ACM Conference on Recommender Systems*. ACM, 73–80.

[28] Liang Tang, Yexi Jiang, Lei Li, Chunqiu Zeng, and Tao Li. 2015. Personalized recommendation via parameter-free contextual bandits. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 323–332.

[29] Liang Tang, Romer Rosales, Ajit Singh, and Deepak Agarwal. 2013. Automatic ad format selection via contextual bandits. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 1587–1594.

[30] William R Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25, 3/4 (1933), 285–294.

[31] Peter Whittle. 1988. Restless bandits: Activity allocation in a changing world. *Journal of applied probability* 25, A (1988), 287–298.

Cookie Synchronization: Everything You Always Wanted to Know But Were Afraid to Ask

Panagiotis Papadopoulos
Brave Software
panpap@brave.com

Nicolas Kourtellis
Telefonica Research, Spain
nicolas.kourtellis@telefonica.com

Evangelos P. Markatos
FORTH-ICS, Greece
markatos@ics.forth.gr

ABSTRACT

User data is the primary input of digital advertising, fueling the free Internet as we know it. As a result, web companies invest a lot in elaborate tracking mechanisms to acquire user data that can sell to data markets and advertisers. However, with same-origin policy and cookies as a primary identification mechanism on the web, each tracker knows the same user with a different ID. To mitigate this, Cookie Synchronization (CSync) came to the rescue, facilitating an information sharing channel between 3rd-parties that may or not have direct access to the website the user visits. In the background, with CSync, they merge user data they own, but also reconstruct a user's browsing history, bypassing the same origin policy.

In this paper, we perform a first to our knowledge in-depth study of CSync in the wild, using a year-long weblog from 850 real mobile users. Through our study, we aim to understand the characteristics of the CSync protocol and the impact it has on web users' privacy. For this, we design and implement CONRAD, a holistic mechanism to detect CSync events at real time, and the privacy loss on the user side, even when the synced IDs are obfuscated. Using CONRAD, we find that 97% of the regular web users are exposed to CSync: most of them within the first week of their browsing, and the median userID gets leaked, on average, to 3.5 different domains. Finally, we see that CSync increases the number of domains that track the user by a factor of 6.75.

CCS CONCEPTS

• **Security and privacy** → **Web protocol security**; • **Networks** → *Network privacy and anonymity*; • **Social and professional topics** → *Surveillance.*

KEYWORDS

Cookie Synchronization, Cross-domain tracking, HTTP Cookies

ACM Reference Format:
Panagiotis Papadopoulos, Nicolas Kourtellis, and Evangelos P. Markatos. 2019. Cookie Synchronization: Everything You Always Wanted to Know But Were Afraid to Ask. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313542

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313542

1 INTRODUCTION

In the online era, where behavioural advertising fuels the majority of Internet, user privacy has become a commodity that is being bought and sold in a complex, and often ad-hoc, data ecosystem [17, 24, 35, 38]. Users' personal data collected by IT companies constitute a valuable asset, whose quality and quantity significantly affect each company's overall market value [40]. As a consequence, it is of no doubt that in order to gain advantage over their competitors, web companies such as advertisers and trackers participate in a user data collecting spree, aiming to retrieve as much information as possible and form user profiles. These detailed profiles contain personal data [12] such as interests, preferences, personal identifying information, geolocations, etc. [5, 31, 44], and could be sold to 3rd-parties for advertising or other purposes beyond the control of the user [30, 36, 37].

Highlighting the importance of this data collection, web companies have invested a lot in elaborate user tracking mechanisms. The most traditional one includes the use of cookies: they have been commonly used in the Web to save and maintain some kind of state on the web client's side. This state has been used as an identifier to authenticate users across different sessions and domains. Initially, *1st-party* cookies were used to track users when they repeatedly visited the same site, and later, *3rd-party* cookies were invented to track users when they move from one website to another. The same-origin policy (SOP) was invented a few years later [45] to restrict the potential amount of information trackers can collect about a user and share with other 3rd-party platforms.

To overcome this restriction, and create unified identifiers for each user, the ad-industry invented the Cookie Synchronization (CSync) process [19]: a mechanism that can practically "circumvent" the same-origin policy, and allow web companies to share (synchronize) cookies, and match the different IDs they assign for the same user while they browse the web. Sadly, recent results show that most of the 3rd-parties are involved in CSync: 157 of top 200 websites (i.e. 78%) have 3rd-parties which synchronize cookies with at least one other party, and they can reconstruct 62-73% of a user's browsing history [11]. Furthermore, 95% of pages visited contain 3rd-party requests to potential trackers and 78% attempt to transfer unsafe data [46]. Finally, a mechanism for respawing cookies has been identified, with consequences in the reconstruction of users' browsing history, even if they delete their cookies [1].

Although past works highlight the use of CSync across the most popular sites worldwide, they avoid diving into the details and fail to thoroughly explore this increasingly popular technique. In fact, the majority of related works perform their analysis using crawled data by fetching the top Alexa websites. Consequently, little is known on how CSync works in the wild, how many of real users' cookies are getting synced during their everyday browsing,

and to what extend it affects the end users' privacy. Importantly, existing studies focus solely on desktop web; however, today more than 52.2% of all web traffic is generated through mobile phones (up from 50.3% in the previous year [42]). This traffic points to a whole new, unexplored ecosystem of mobile Web, where users browse through their very personal devices, while employing a variety of sensors to experience a highly personalized content, but always at the expense of privacy. Thus, what is the impact of CSync in this new mobile ecosystem? *Which are the basic characteristics of CSync, and the mechanics used? How does CSync impact the user's privacy and anonymity on the mobile web?*

In this paper, we aim to answer these questions, by studying Cookie Synchronization using a large, year-long dataset of 850 real mobile users, and exploring in depth its use and growth through time, dominant companies, along with its side-effects on user privacy. The contributions of this work are as follows:

- We design and implement CONRAD: (COokie syNchRonizAtion Detector) a holistic mechanism to detect at real time CSync events and the privacy loss on the user side, even when synced IDs are concealed or obfuscated by participating companies aiming to reduce identifiability from traditional, heuristic-based detection algorithms. In such cases, when non-ID related features are used, our approach achieves high accuracy (84%-90%) and AUC (0.89-0.97).
- We perform the first of its kind, large-scale, longitudinal study of Cookie Synchronization on mobile users in the wild. We perform a passive data collection of the activity of 850 volunteering mobile users, lasting an entire year. This means that the data collected are not crawled, like in past studies, and therefore do not capture a distorted or biased picture of CSync on the Web. Instead, these data provide a rare glimpse of CSync in the mobile space, and an opportunity to study CSync and its impact on real mobile users' privacy and anonymity.
- Using the proposed detection mechanism, we conduct an in-depth privacy analysis of CSync. Our results show that 97% of regular web users are exposed to CSync. In addition, the average user receives ~1 synchronization per 68 HTTP requests, and the median userID gets leaked, on average, to 3.5 different domains. Furthermore, CSync increases the number of domains that track the user by a factor of 6.75. Finally, we detect CSync involved in different scenarios such as breaking-off an SSL session, and exposing userIDs and other personal data in cleartext.

2 COOKIE SYNCHRONIZATION

2.1 How does Cookie Synchronization work?

Figure 1 presents a simple example to understand in practice what is CSync and how it works. Let us assume a user browsing several domains like `website1.com` and `website2.com`, in which there are 3rd-parties like `tracker.com` and `advertiser.com`, respectively. Consequently, these two 3rd-parties have the chance to set their own cookies on the user's browser, in order to re-identify the user in the future. Hence, tracker.com knows the user with the ID `user123`, and advertiser.com knows the same user with the ID `userABC`. Now let us assume that the user lands on a website (say `website3.com`), which includes some JavaScript code from

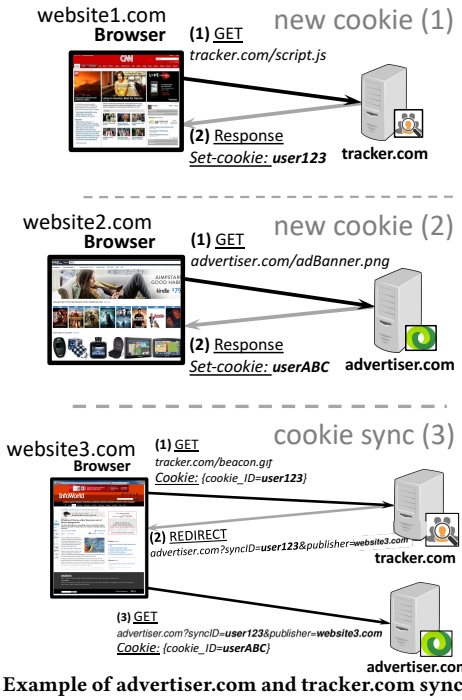


Figure 1: Example of advertiser.com and tracker.com synchronizing their cookieIDs. Interestingly, and without having any code in website3, advertiser.com learns that: (i) cookieIDs userABC==user123 and (ii) userABC has just visited the given website. Finally, both domains can conduct server-to-server user data merges.

tracker.com but not from advertiser.com. Thus, advertiser.com **does not (and cannot) know which users visit website3.com**. However, as soon as the code of tracker.com is called, a GET request is issued by the browser to tracker.com (step 1), and it responds back with a REDIRECT request (step 2), instructing the user's browser to issue another GET request to its collaborator advertiser.com this time, using a specifically crafted URL (step 3):

GET *advertiser.com?syncID=**user123**&publisher=**website3.com***
Cookie: {cookie_ID=**userABC**}

When advertiser.com receives the above request along with the cookie ID `userABC`, it finds out that `userABC` visited website3.com. To make matters worse, advertiser.com also learns that the user whom tracker.com knows as `user123`, and the user `userABC` is basically one and the same user. Effectively, CSync enabled advertiser.com to collaborate with tracker.com, in order to: (i) find out which users visit website3.com, and (ii) synchronize (i.e., join) two different identities (cookies) of the same user on the web.

2.2 Cookie Synchronization, Personalized Advertising & Privacy Implications

Digital advertising has moved towards a more personalized model, where ad-slots are purchased programmatically (e.g., Real Time Bidding (RTB) based auctions) in auctions, based on how well the profile of the visitor matches the advertised product. Consequently, advertisers need to obtain user data (e.g., interests, behavioral patterns) to use as input in their sophisticated decision engines. The core component for this data purchasing/sharing includes CSync [16].

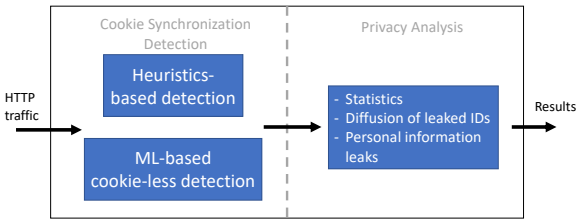


Figure 2: High-level overview of CONRAD's internal components.

It allows trackers and advertisers to perform common user identification and by participating in data markets enrich their knowledge base with user information from several data sources.

There are several privacy implications for the online users who access websites planted with such sophisticated tracking technologies. Using CSync, in practice, advertiser.com learns that: (i) what it knew as `userABC` is also `user123`, and (ii) this user has just visited website3.com. This enables advertiser.com to track a user to a much larger number of websites than was initially thought. Indeed, by collaborating with several trackers, advertiser.com is able to track users across a wide spectrum of websites, even if those websites do not have any collaboration with advertiser.com.

To make matters worse, the ill effects of CSync may reach way back in the past - even up to the time before the invention of CSync. Assume, for example, that someone manages to get access to all data collected by tracker.com, and all the data collected by advertiser.com (e.g., by acquisition [30, 36], merging or hacking of companies [22]). In the absence of CSync, in those two datasets, our user has two different names: `user123` and `userABC`. However, after one single CSync, those two different names can be joined into a single user profile, effectively merging all data in the two datasets. Nowadays, such cases of *server-to-server user data merges* are taking place at a massive scale [11], with the different web companies conducting mutual agreements for data exchanges or purchases, in order to enrich the quality and quantity of their user data warehouses [6, 30].

As if these threats to user privacy were not enough, CSync can rob users of the right to erase their cookies. Indeed, when coupled with other tracking technologies (i.e. evercookie [39], or user fingerprinting [9]), CSync may re-identify web users even after they delete their cookies. Specifically, when a user erases her browser state and restarts browsing, trackers usually place and sync a new set of userIDs, and eventually reconstruct a new browsing history. But if one of them manages to respawn [1] its cookie (e.g. through evercookie [39]), then through CSync, all of them can link the user's browsing histories from before and after her state erasure. Consequently: (i) users are not able to abolish their assigned userIDs even after carefully erasing their set cookies, and (ii) trackers are enabled to link user's history across state resets.

3 COOKIE SYNCHRONIZATION DETECTION

In this paper, we design CONRAD, a holistic methodology to detect CSync events in real time, on the user side. CONRAD monitors the HTTP(S) traffic of the user on the browser level and detects userIDs when shared from one domain to the other. To achieve that, it uses a (i) Heuristics-based (stateful) detection mechanism (Section 3.1), where the IDs from cookies are tainted and alert is raised when they are exfiltrated to a domain other than the owner

Table 1: Examples of userIDs synchronized among various domains.

URLs of Cookie Synchronization HTTP Requests
1. a.atemda.com/id/csync?s=**L2zaWQvMS9lkLzMxOUwOTUw**
2. bidtheater.com/UserMatch.ashx?bidderid=23& bidderuid=**L2zaWQvMS9lkLzMxOUwOTUw**&expiration=1426598931
3. d.turn.com/r/id/**L2zaWQvMS9lkLzMxOUwOTUw**/mpid/

of the cookie. However, as also presented in past studies [2], more and more companies include encryption (or cryptographic hashing) in the CSync-related APIs, thus concealing the synced IDs. To deal with these cases, CONRAD uses a (ii) ML-based (stateless) detection mechanism (Section 3.2) capable of classifying with high accuracy such possible concealed synchronizations, without relying on any previously stored cookie IDs, but only using characteristics from the connections themselves. Figure 2, provides a high level overview of CONRAD's internal components.

For each CSync detection method, CONRAD extracts its information flow: the chain of domains that share the synced ID, the domain that triggered the CSync event, the domains that (without having access to the website) used this sync request to set and sync their own userIDs (see Section 5.1). This way, our tool is able to measure the diffusion of anonymity loss for the given user by analysing what number of their overall userIDs budget got synced, and to how many 3rd-party domains. In addition, by using a simple pattern matching technique, CONRAD extracts possible personal information leaks tailored with the synced ID (see Section 5.5).

Although there are several existing techniques for detecting ID-sharing events even when cookies are encrypted, accurate CSync detection in real time is a hard task. The main advantages of our approach, contrary to existing detection mechanisms [1, 2, 11] are as follows: (i) It offers the ability to detect synchronizations when the userID is embedded not only in the URL's parameter, but also in its path (either in case of request/response URL or Location URL of the referrer). (ii) By filtering-out domains of the same provider, our approach can discriminate between intentional CSync and unequivocally legitimate cases of internal ID sharing, thus avoiding false positives. (iii) It is capable of detecting Cookie Synchronization *at real time*, even when shared IDs are encrypted.

3.1 Heuristics-based detection

Technically, as we see in Table 1, CSync is nothing more than a request from the user's browser to a 3rd-party domain carrying (at least one) parameter that constitutes a unique ID set by the calling domain. However, what CSync typically enables is a multiple, back-to-back operation with several 3rd-party domains getting updated with one particular ID. This multiple synchronization happens by utilizing URLs of HTTP requests (i.e., the Location HTTP header), in which the cookie ID (i.e., the userID) of the triggering domain is embedded. The userID may be embedded in the: (i) parameters of the URL, (ii) URL path, or (iii) referrer field. In some cases, detection may be straightforward: one can simply look for specific parameter names (e.g., syncid, user_id, uuid). However, different companies use different APIs and parameter naming; relying only on string matching for Cookie Synchronization detection will lead to a large number of false negatives in case of newcomer syncing domains. To remedy this, by extending previous work [29], we design a stateful heuristics-based detection algorithm, which relies on the previously set cookies to taint userIDs that may get synced with

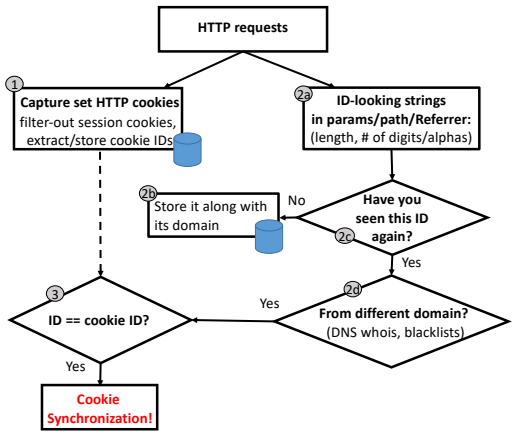


Figure 3: Heuristics-based CSync detection mechanism.

domains different than the cookie setter[1]. In particular, our Cookie Synchronization detection methodology includes the following steps, which are also illustrated in Figure 3:

(1) We extract all cookies set on the user's browser. To accomplish that, we parse all HTTP requests in our dataset and extract all *Set-Cookie* requests.
   (a) We filter out all session cookies. These are cookies without expiration date, that get deleted after the end of a session.
   (b) We parse the cookie value strings using common delimiters (i.e. ":", "&"). By extracting potentially identifying strings (cookie IDs), we create a list with the cookie IDs that could uniquely identify the user in the future.
(2) We detect possible ID-sharing events in the HTTP requests:
   (a) We identify ID-looking strings that are unique per user, carried either (i) as parameters in the URL, (ii) in URL path, or (iii) in the referrer field. As ID-looking strings, we define strings with specific length (> 10 characters) – false positives do not matter at this point.
   (b) If this ID is seen for the first time, it is stored in a hashtable along with the URL's domain (receiver of the ID).
   (c) If this ID has already been seen, we consider it as a shared ID and the requests carrying it as *ID-sharing requests*.
   (d) To check if the above ID-sharing requests regard different domains, we use several external sources (DNS whois, blacklists etc.) to filter-out cases where the IDs are shared among domains owned by the same provider (e.g., amazon.com and amazonaws.com) [25]. This way, our approach can discriminate between intentional ID leaking and legitimate cases of internal ID-sharing, thus avoiding false positives.
(3) Finally, to verify if the detected shared ID is a userID able to uniquely identify a user, we search this ID within the list of cookie IDs extracted in the first step. If there is a match, then we consider this request as CSync.

[1]A known limitation of this approach is its inability of capturing a small portion of 1st-party cookies set by javascript[7]

Table 2: Examples of Cookie Synchronization between 3rd-parties with plaintext and encrypted cookie IDs.

ID syncs beyond the 1-1 of source domain (plaintext):
Domain syncs ID with Tracker1 ID1; → ID1=ID;
Domain syncs ID with Tracker2 ID2; → ID2=ID;
Tracker1 syncs ID1 with Tracker2 ID2; → ID2=ID1=ID;
ID syncs beyond the 1-1 of source domain (encrypted):
Domain syncs h(ID,A) with Tracker1 ID1; → ID1=h(ID,A);
Domain syncs h(ID,B) with Tracker2 ID2; → ID2=h(ID,B);
Tracker1 syncs ID1 with Tracker2 ID2; → h(ID,A)!=h(ID,B), i.e., ID1!=ID2;

3.2 Cookie-less detection

It is apparent that in order for the above methodology to be a viable CSync detection method, cookie IDs need to be shared in plaintext. However, major web companies such as DoubleClick [19] have started encrypting the cookie ID in an attempt to protect the actual cookie from being revealed to unwanted parties that may snoop the user's traffic (plugins or even ISPs).

While the former case is obvious why companies would want to block such snooping, the latter case may not be clear why and, thus, we discuss this case further. In particular, under the traditional, plaintext case of cookie ID syncing, the same source company can sync independently with multiple 3rd-parties for the same user cookie ID. Thus, no-one forbids these other 3rd-parties from syncing their IDs with each other, and find out that they have information about the same user, something that goes beyond what the source company intended to do (top example of Table 2). With hashing or encryption of the cookie ID, these 3rd-parties are unable to do this syncing (bottom example of Table 2). As a consequence of this encryption, CSync events can proceed undetected, if the previously used detection method is employed.

To address this scenario, in this paper, we propose a novel method for identifying CSync which is oblivious to the IDs shared. This mechanism is able to identify with high accuracy CSync events in web traffic, even when the leaked IDs are protected and cannot be matched. To build this mechanism, we employ machine learning methods, which we train on the ground truth datasets created with the previous, heuristic-based technique. In particular, we analyze various features extracted from the web traffic due to CSync, and train a machine learning classifier to automatically classify a new HTTP connection as being a CSync event or not. Here, we make the assumption that the various features used to characterize, and eventually detect, CSync with plaintext IDs, are equally used, and have the same distributions and variability as in the CSync with encrypted IDs. We believe this is a reasonable assumption, since the companies employing encrypted IDs are not expected to change the rest of their mechanism which delivers these IDs and triggers CSync with their partners; these companies only want to obfuscate the IDs to avoid further, and unwanted, CSync.

For training the classifier, we extract various relevant features from the network traces. As ground truth, we use confirmed CSync events which were detected with the heuristics-based method described earlier. Beyond these confirmed events, there are *id-sharing* events which were first selected by the method as potential CSync events, but eventually were rejected as *non-CSync*, as they did not match cookie IDs already seen by the method (step 1 in Figure 3).

The features available for these network events can be several; we constrain the learning algorithm to use only features available at run time, and during the user's browsing to various websites:

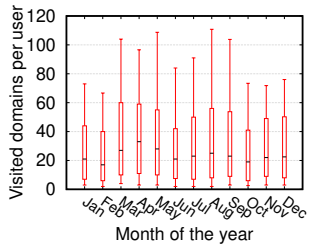


Figure 4: Distribution of number of unique domains visited per user, per month. The median user in our dataset visits 20 - 30 different domains per month.

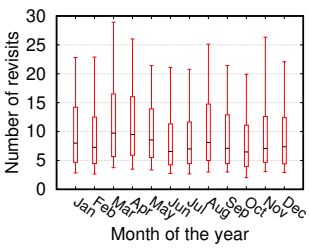


Figure 5: Distribution of number of times a user revisits the same domain per month. The median user revisits a domain around 7-10 times per month.

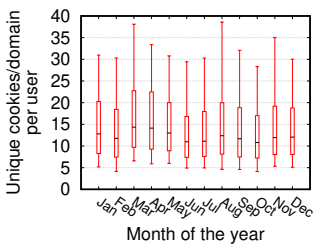


Figure 6: Number of (first and 3rd-party) cookies per domain per user. We see that the median user receives 12.25 cookies, on average, per visited website.

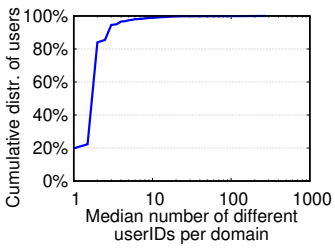


Figure 7: Unique userIDs set per domain, across the year. 80% of users are known to a single domain with only ~2 aliases, on average.

Table 3: Summary of contents in our dataset.

Description	#	Description	#
Total mobile users	850	Unique shared IDs (S)	68215
Requests captured	179M	Unique userIDs synced ($C \cap S$)	22329
Unique Cookies (C)	8.97M	CSync requests	263635
ID sharing requests	412805		

- EntityName: {domain of recipient company}
- TypeOfEntity: {Content, Social, Advertising, Analytics, Other}
- ParamName: {aid, u, guidm, subuid, tuid, etc.}
- WhereFound: {parameter in URL, parameter in Referrer, in the URL path}
- StatusCode: {200, 201, 202, 204, etc.}
- Browser: {Firefox, Chrome, Internet Explorer, etc.}
- NoOfParams: {0, 1, 2, ..., etc.}

Various machine learning algorithms can be applied: Random Forest, Support Vector Machines, Naive Bayes, and even more advanced methods such as Neural Networks. However, a balance must be found between training the given algorithm to reach a good accuracy, the computation cost for this training, as well as the capability of the algorithm to be used at real time on the user device.

This method aims to address two possible scenarios that can arise while IDs are being shared: First, we consider the realistic scenario that an already identified set of id-sharings are candidate CSync events (as found by the heuristics-based method), but cannot be validated as CSyncs because of the cookie ID being encrypted or unavailable, and therefore not matching the repository of IDs. Second, we consider the scenario where various HTTP connections are ingested by the method, and it needs to decide at run-time which are CSync events and which are not. This case is a more generalized version of the previous scenario, and attempts to detect CSync events, as an alternative method to the heuristic-based approach. In either of the two cases, we follow a generally accepted methodology (e.g., [34]) that separates training such a machine learning model, from applying it at real-time. The training can be performed offline, on an existing dataset (e.g., the one we collected, or from anonymous user network traffic donations), and the model trained can be distributed accordingly with the tool at hand. Then, the tool can apply the classifier on each network connection under question, for real-time classification.

4 DATASET

In this section, we describe the data collection process and our year long dataset. In order to collect data from real users, we set a group of proxies fronted by a load balancer, and gathered 850 volunteering users residing in the same country. These users agreed to strip their browsers from any previous state (i.e., cookies, cache, webStorage) and have their devices to continuously redirect their network traffic through our proxies for 12 consecutive months (2015-2016). They signed a consent form allowing us to collect and analyze their data during this period, and publish any anonymized results. They were well-aware of the purposes of the data collection, and were compensated with free data plan, as long as they were using the proxies. Before the analysis was performed, all data were anonymized and never shared with any other domains.

Given the long duration of the experiment, and in order not to jeopardize the confidentiality of the volunteering users' secure sessions, we capture only their HTTP mobile traffic (same method can be applied for HTTPS). On the server-side and based on the user agent of each request, we filtered out any possible app-related traffic. Overall, we collected a dataset containing a total of 179M requests, spanning an entire year. Table 3 summarizes the contents and CSync findings in our dataset.

Users: To analyse our dataset, we create a simple weblog parser. As noted earlier, this dataset consists of web browser traffic from the mobile devices of 850 users. After separating the flows of each one of them, we produced their timelines, and in Figure 4, we present the number of different domains each user visits per month. As we see from the distribution (Percentiles: 10th, 25th, 50th, 75th, 90th), the median user in our dataset visits 20-30 different domains depending on the month (we observe a seasonal phenomenon with increases during spring break and summer holidays).

Similarly, in Figure 5, we present the number of times each of these domains gets revisited by the median user in our dataset. In every revisit, there is a new request that asks the user's browser if there is a previously set cookie. If this *Get-Cookie* request regards a previously set cookie, this means that the domain already knows the user and we consider it as a revisit. We observe that the median user revisits around 7-10 times the same domain from their mobile browser. Again, we observe the seasonal phenomenon as earlier, when users tend to have more time to browse the web. Also, the 75th percentile of the users may revisit the same website more than 15 times (March, August).

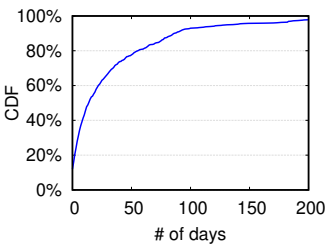


Figure 8: Distribution of time taken for first CSync to appear per user. 20% of users get their first userID synced in 1 day or less.

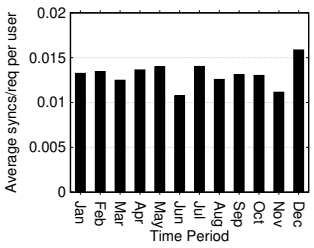


Figure 9: CSyncs per request for the average user across the year. The average user receives 1 synchronization every 68 requests.

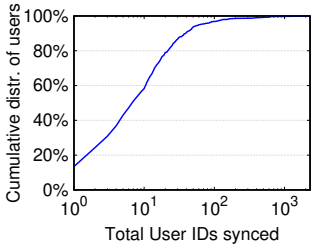


Figure 10: Distribution of the synced userIDs per user. The median user has 7 userIDs synced, when 3% of users has up to 100.

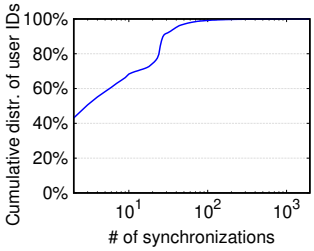


Figure 11: Distribution of synchronizations per userID. The median userID gets synced with 3.5 different domains.

Table 4: Breakdown of the CSync triggering factors.

	Initiator	Portion
(i)	Publisher syncs its userID	2.692%
(ii)	Embedded 3rd-party triggers syncing of its own set userID	49.668%
(iii)	3rd-party uses sync request to share its own set userID	45.697%
(iv)	3rd-party uses sync request to share with other domains the publisher's set userID	0.2658%

Cookies: To have a good view of the cookie activity of users, we extract all (1st & 3rd-party) cookies set in users' browsers across the year, and in Figure 6, we plot the distribution of number of cookies per visited website. The median user receives a fairly constant number of 12.25 cookies per visited website per month, on average.

Next, we extract the unique identifiers (i.e., cookie ID) set in these cookies. A cookie ID constitutes a unique string of characters that websites and servers associate with the browser and, thus, the user who stored the cookie. Thereby, here, we consider a cookie ID as a unique user identifier called userID. As it can be seen in Table 3, in our dataset, there are almost 9 million such unique IDs.

In Figure 7, we plot the distribution of the number of unique userIDs assigned to the users per domain. The vast majority of the users (80%) receive, on average, only 2.2 userIDs per domain, across the year. This means that users tend not to erase their cookies frequently, thus, allowing web domains to accurately identify them through time and during the users' browsing. Only 1.13% of users erase their cookies (either manually or by browsing with Private/Incognito Browsing), receiving more than 9.5 different userIDs per domain, on average. This means that it is very rare for a domain to meet a previously known user with a different alias.

5 PRIVACY ANALYSIS

By applying CONRAD in our dataset, we find several IDs passed from one domain to another. We detect 68215 such unique shared IDs. From these IDs, 22329 were actually cookie IDs from previously set cookies that were synced among different domains. In total, these cookie IDs were found in 263635 synchronization events (see Table 3 for a summary). From the CSyncs detected in our dataset, userIDs were found in 91.996% of the cases inside the URL parameters, 3.705% in the Referrer URL, and in 3.771% of the cases in the URL path. Thus, CONRAD was able to detect 3.771% more cases of CSyncs than existing detection methods [1, 29, 32].

5.1 Initiation of Cookie Synchronization

First, we correlate the cookie domain (the setter), the synchronizing request's Referrer field, and the publisher that the user visited, in order to extract the domain that triggered the CSync on the user's browser. As seen in Table 4, there are 4 distinct cases: (i) the CSync was initiated by the publisher who syncs the userID he assigned for the user: we find 2.692% of these cases in our dataset, (ii) the synchronization was initiated from a publisher's iframe, by the guest 3rd-party which syncs its own userID (49.668%), (iii) a 3rd-party which participated in a previous synchronization (case (i) or (ii) above) and uses the sync request to share its own userID (45.697%). Lastly, there is a rare case (iv), where a 3rd-party participating in a previous synchronization of the publisher's userID (case (i)) initiates a new round of syncs while it continues to share the publisher's userID (0.2658%). Obviously, in case (iv), the initiating 3rd-party shares with its 3rd-party affiliates, a userID assigned by a domain (the publisher) beyond its control, and possibly awareness.[2]

5.2 How are users exposed to CSync?

CSync impacts users' privacy by leaking assigned userIDs, and sharing them with 3rd-parties. In our dataset, we see that for users with regular activity on the web (> 10 requests per day), **97% were exposed to CSync at least once**. This means that CSync constitutes a phenomenon affecting the totality of online users. Next, we study how long it takes for the first synchronization to happen, or in effect, how quickly a user gets exposed to CSync after she starts browsing. Recall that as mentioned in Section 4, all participating users, during bootstrapping phase, had all state from their browsers erased. This means that our proxy was able to capture the very first cookie that was set during the user's monitoring period. Of course, the time depends on the browsing patterns of each user, however, as we see in Figure 8, a median user experiences at least one CSync within the first week of browsing. In fact, a significant **20% of users get their first userIDs synced in 1 day or less**. It is worth noting that users tend to browse the same top websites repeatedly (e.g., facebook.com, twitter.com, cnn.com), so the set cookies are already shared and no sync is fired.

Next, we investigate if the synchronizations the users are exposed to, change over time. Hence, we extract CSyncs per user, and normalize with the user's total number of requests. In Figure 9, we plot the average synchronizations per request across the year. As

[2]We reported all such cases and notified the respected publishers.

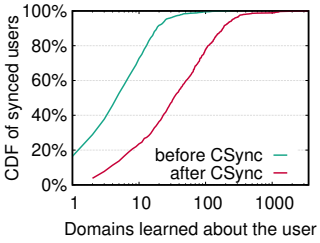


Figure 12: Distribution of domains learning at least a userID of a user (with/without the effect of CSync). After syncing, the domains that learned about the median user grew by 6.75×.

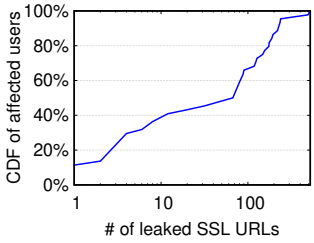


Figure 13: Distribution of the leaked TLS URLs per affected user. The median user has 70 TLS URLs leaked through Cookie Synchronization, when the 90th percentile has up to 226 TLS URLs leaked.

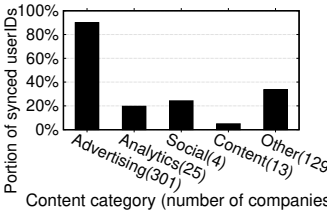


Figure 14: Portion of synced userIDs learned per content category. As expected, ad-related companies learned the vast majority (90%) of the total synced userIDs in our dataset.

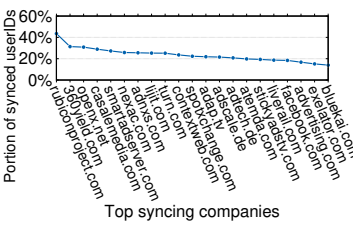


Figure 15: Portion of synced userIDs learned per tracker: 3 trackers learn more than 30% of the total userIDs in our dataset; 14 trackers learn more than 20% of userIDs each.

shown, CSync is persistent through the duration of an entire year, with the user being exposed to a steady number of synchronizations across time. Specifically, we see that **the average user receives around 1 synchronization per 68 requests**.

Considering the different userIDs that trackers may assign to a user, in Figure 10, we measure the number of unique userIDs that got synced per user. Evidently, **a median user gets up to 6.5 userIDs synced**, and 3% of users has up to 100 userIDs synced. It becomes apparent that the IDs of a user may leak to multiple 3rd-party domains through CSync. To measure the userID leak diffusion, in Figure 11, we plot the distribution of synchronizing requests per userID. As we see, **the median userID gets leaked, on average, to 3.5 different domains**. There is also a significant 14% that gets leaked to up to 28 different 3rd-parties.

To better understand the effect of CSync on the diffusion of the overall user privacy, we measure for each user the number of domains that learned about them (i.e., learned at least one of their userID) before and after CSyncs. From the distribution in Figure 12, **the domains that learned about the median user after CSyncs grew by a factor of 6.75**, and for 22% of users this factor becomes > 10. This means that before the rise of CSync, when the user visited a website, the domains that could track them were only the publisher and the included 3rd-parties, but in an independent fashion. However, with the introduction of CSync, the number of domains that can track the user drastically increased (6.75x for median user), severely decreasing their online anonymity.

Table 5: Example of an *ID Summary* stored on the user's browser. It includes userIDs and expiration dates used for the particular user by 4 different domains.

ID Summary stored in cookie by adap.tv
"key=**valueclickinc**:value=708b532c-5128-4b00-a4f2-2b1fac03de81:expiresat=wed apr 01 15:03:42 pdt 2015,key=**mediamathinc**:value=60e05435-9357-4b00-8135-273a46820ef2:expiresat=thu mar 19 01:09:47 pst 2015,key=**turn**:value=2684830505759170345:expiresat=fri mar 06 16:43:34 pst 2015,key=**rocketfuelinc**:value=639511149771413484:expiresat=sun mar 29 15:43:36 pst 2015"

5.3 Buy 1 - Get 4 for free: ID bundling and Universal IDs

We find **63 cases of domains which set on the users' browsers cookies with userIDs previously set by other domains**. For example, we see the popular `baidu.com`, the world's eighth-largest Internet company by revenue, storing a cookie with an ID *baiduid* = {*idA*}, and more than 5 different domains after this incident setting their own cookie using the same ID *baiduid* = {*idA*}. This by-product of CSync, enables trackers to use *universal IDs*, thus, bluntly violating the same-origin policy and merging directly (without background matching) the data they own about particular users.

In addition, we find **131 cases of domains storing in cookies the results of their CSyncs, thus composing *ID Summaries***. In these summaries, we see the userIDs that other domains use for the particular user previously obtained by CSyncs. An example of such summaries in JSON is shown in Table 5. As one can see, the cookie set by `adap.tv` includes the userIDs and cookie expiration dates of `valueclick.com`, `mediamath.com`, `turn.com` and `rocketfuel.com`. In our dataset, we find at least 3 such companies providing *ID Summaries* to other collaborating domains. This user-side info allows (i) the synchronizing domains to learn more userIDs through a single synchronization request, and (ii) `adap.tv` to re-spawn any deleted or expired cookies of the participating domains at any time, just by launching another CSync.

5.4 Spilling userIDs out of TLS

It is well-known that mixing encrypted and non encrypted sessions in TLS is a bad tactic [10]. Using TLS, everything except IPs and ports is encrypted [15] in a HTTP connection. As a result, although there are sophisticated estimation techniques [18], no observer can monitor what the user is actually browsing over TLS, or the IDs assigned to her. Given that our proxy is monitoring only HTTP traffic (see Section 4), one would expect that no information from secure TLS sessions would be captured. To our surprise, and as already found in [33], in our dataset we see userIDs that originated from TLS sessions getting leaked over plain HTTP to plain 3rd parties. As a result, *any curious, in-path observer (e.g., a snooping ISP) can eavesdrop the leaked userIDs*[3]. To make matters worse, we see

[3]As soon as we verified this leak, we notified publishers and also our volunteering users who updated their signed consent.

Table 6: Example of ID-spill from SSL in our dataset.

Role	Domain
Visited website:	https://financialexpress.com
Cookie setter: SetCookie:	https://tapad.com **D0821FA0-8A80-4D9E-BC85-C40EAC4E4FF5**
Cookie syncer:	http://delivery.swid.switchadhub.com/adserver/user_sync.php?SWID=cf43265166a9ccf5f6fd0472f23776fa&sKey=PM2&sVal=**D0821FA0-8A80-4D9E-BC85-C40EAC4E4FF5** referrer: **financialexpress.com** Get-cookie: {cf43265166a9ccf5f6fd0472f23776fa}
Cookie syncer:	http://tags.bluekai.com/site/3096?id=**D0821FA0-8A80-4D9E-BC85-C40EAC4E4FF5** referrer: **financialexpress.com** Get-cookie: {c57b29d1-f8e2-11e7-ac1b-0242ac110005}

that the same ID leaking requests have referrer fields[4], which leak the particular webpage the user visited over TLS (e.g., the particular article of the news site she read), thus leaking her interests.

To reproduce this leak, we manually visit the TLS protected websites where ID-spilling was found and we see it is caused by CSync events that sync a userID from a TLS cookie with non-TLS 3rd parties. In Table 6, we present one such real case we observe. As shown, while visiting over TLS the page *https://financialexpress.com*, two CSyncs are performed: *https://tapad.com* advertiser shares with *http://switchadhub.com* and *http://bluekai.com* the ID it assigned to the user. This way, the latter two tracking domains sync their set-cookies with the one of *https://tapad.com*. However, by doing that over plain HTTP, the visited webpage gets leaked through the referrer field to a monitoring entity, even when users browse through proxies or VPN, or even Tor. In addition, this entity from now on can re-identify the user in the web, just by monitoring the userIDs of cookies in requests destined to *http://switchadhub.com* and *http://bluekai.com*, even if the user's IP address is frequently changed. Obviously, the more 3rd-parties were participating in the Cookie Synchronization, the easier it would be for the monitoring entity to capture HTTP requests loaded with these synced cookies and, thus, re-identify the user using a secure VPN.

We measured such cases in our dataset and found **44 users (5%) affected by the ID-spilling of CSync**. The majority of leaked domains regard popular content providers, where an eavesdropper from the referrer field, apart from the domain, can also see sensitive information. From Figure 13, the median (90th percentile) user has 70 (226) TLS URLs leaked through CSync.

5.5 Sensitive information leaked with userIDs

The websites a user browses can easily leak through the referrer field during a CSync. Moving beyond this type of basic leak, in our dataset we find several cases of privacy-sensitive information passed to the syncing domain regarding the particular user with the particular synced ID. By deploying a simple string matching script, we look for keywords (e.g., gender, age, name, etc.) and find:

- 13 syncs leaking the user's city level location
- 2 syncs leaking the user's registered phone number
- 10 syncs leaking the user's gender
- 9 syncs leaking the exact user's age
- 3 syncs leaking the user's full birth date
- 2 syncs leaking the user's first and last name
- 16 syncs leaking the user's email address
- 4 syncs leaking user login credentials: username/password

[4]There are specific directives [23] for Referrer field hiding when referring over TLS visited domains.

The above information constitutes not only a severe privacy threat for the user, but can also enables potential impersonation attacks.

5.6 Who are the dominant CSync players?

In order to assess the content that CSync parties provide, we extract all domains involved in CSync, and using EasyList, EasyPrivacy [14] and the blacklist of the Disconnect browser extension [8] (enriched with our additions after manual inspection), we categorize them according to the content delivered. This way, we create five categories of domains related with: (i) Advertising, (ii) Analytics, (iii) Social, (iv) 3rd-party content (e.g., CDNs, widgets, etc.), and (v) Other.

As we saw in Figure 11, the median userID gets shared with more than one domains. We find that **ad-related domains participate in more than 75% of the overall CSyncs through the year**. Consequently, as we can also observe in Figure 14, **ad-related domains have learned as much as 90% of all userIDs that got synced**, with Social and Analytics -related domains following with 24% and 20% respectively. In Figure 15, we plot the top 20 companies[5] that learned the biggest portion of the total userIDs through CSyncs in our dataset. Interestingly, the top 3 companies (i.e., rubiconproject.com, 360yield.com and openx.net) learned more than 30% of all userIDs in our dataset each. There is also a significant number (14) of companies that learn more than 20% of userIDs.

6 COOKIE-LESS DETECTION

In this section, we explore two different scenarios outlined in Section 3.2 regarding the detection of CSync via ID sharing, while such IDs may be obfuscated to remove the possibility of matching them with past IDs shared between entities. First, we explore the scenario where IDs have been shared, detected by the heuristic-based approach, but have not yet been confirmed as CSync events. That is, we consider an already identified set of id-sharings, which are candidate CSync events, but cannot be validated as CSyncs because of the cookie ID being encrypted or unavailable (Section 6.1). Second, we take a step back and consider the more general case where various HTTP connections are ingested by the method, and it needs to decide at run-time which are CSync events and which are not based on given features (Section 6.2).

Towards this end, we train and test the classifier in these two experiments. We remind the reader of the assumption made earlier: the distributions of the features describing the CSync events with unencrypted IDs, have the same variability in the cases of encrypted IDs, and therefore can be used for the detection of such cases. This assumption allows us to handle the problem as an out-of-sample estimation, leaving as future work the final validation with a set of ground-truth data of encrypted IDs that we also know their unencrypted versions.

Data and Features: Based on the ground truth data presented earlier with the heuristic-based technique, we have 412.8k *id-sharing* events, from which 263.6k are confirmed CSync, and 149.2k are identified as *non-CSync*. The features available for these events can be various, as already explained in 3.2. The ones we use are features available at run time, and during the user's browsing to websites.

[5]In a dataset with both HTTP and HTTPS traffic, market shares may differ since there are companies operating over SSL only (e.g., DoubleClick)

Table 7: Performance of decision tree model trained on different subsets of features available at runtime for classification, given already identified id-sharing entries, and 10 cross-fold validation.

Feature subset	F	TPR	FPR	PR	RC	FM	AUC
NoOfParams*	1	0.639	0.639	0.408	0.639	0.498	0.500
WhereFound+	1	0.643	0.610	0.612	0.643	0.535	0.602
StatusCode*+	1	0.648	0.619	0.723	0.648	0.523	0.633
TypeOfEntity*+	1	0.735	0.432	0.752	0.735	0.701	0.661
Browser*+	1	0.700	0.492	0.710	0.700	0.651	0.628
ParamName+	1	0.815	0.295	0.828	0.815	0.803	0.834
EntityName*+	1	0.803	0.295	0.806	0.803	0.793	0.854
{id-less}*	5	0.840	0.242	0.845	0.840	0.834	0.887
{high imp.}+	6	0.870	0.206	0.877	0.870	0.865	0.919
ALL	9	0.900	0.144	0.901	0.900	0.898	0.946

Table 8: Performance of decision tree model trained on different subsets of features available at runtime for classification, given a pre-filter for ID-looking strings. All results besides the last row are with balanced dataset across the three classes, and 10-cross fold validation. The last row's results are computed given an unseen, and unbalanced test set, maintaining the original ratio of classes.

Feature subset	F	TPR	FPR	PR	RC	FM	AUC
NoOfParams*	1	0.541	0.314	0.584	0.541	0.495	0.706
StatusCode*+	1	0.666	0.229	0.673	0.666	0.598	0.764
TypeOfEntity*+	1	0.760	0.162	0.724	0.760	0.695	0.834
EntityName*+	1	0.865	0.075	0.863	0.865	0.860	0.962
ParamName+	1	0.870	0.083	0.878	0.870	0.859	0.953
{id-less}*	4	0.904	0.057	0.904	0.904	0.898	0.973
{high imp.}+	4	0.919	0.051	0.923	0.919	0.914	0.978
ALL	5	0.920	0.051	0.925	0.920	0.916	0.978
Unbalanced	5	0.981	0.004	0.989	0.981	0.984	0.999

Algorithms: The final machine learning classifier used is decision tree-based. Others like Random Forest, Support Vector Machines and Naive Bayes were tested, but the decision tree algorithm outperformed them, with significantly less computation and memory overhead. Indeed, more advanced methods can be used, such as Neural Networks, but since the decision tree-based algorithm works very well, we leave this exploration for the future.

Metrics: To evaluate the performance of the classifier on the different classes and available features (F), standard machine learning metrics were used such as Precision (PR), Recall (RC), F-measure (FM), True Positive rate (TPR), False Positive rate (FPR), and area under the receiver operating curve (AUC).

6.1 CSync in ID-sharing HTTP

In this experiment, we assume there is already in place an existing technique for analysis of the HTTP traffic of the user, similar to the method outlined in Figure 3. However, there are candidate CSync events that cannot be confirmed, as the IDs cannot be matched with SET cookie IDs, either because these actions are not available to the method, or because the IDs are encrypted.

In this case, a machine learning classifier can be trained to detect if an id-sharing HTTP request is a true CSync event, by matching its pattern to past verified CSync events. For this experiment, we use two classes: the CSync events and the *id-sharing but non-CSync* events, to train and test a decision tree classifier under different subsets of features. The training and testing was performed using 10 cross-fold validation process. The results are shown in Table 7. We observe that independently, each of the features considered have some predictive power, except from the *NoOfParams* feature. When the most important features (using information gain as metric) are combined, a weighted AUC of 0.919 is achieved. When we select non-ID related features, a weighted AUC=0.887 is reached, and with all features the classifier can reach a weighted AUC=0.946.

Table 9: Detailed performance of decision tree model trained on different subsets of features in a balanced dataset, and tested on an unseen, and unbalanced test set, which maintains the original ratio of classes (last row of Table 8). C: CSync, ICS: *id-sharing but non-CSync*, O: Other, WA: weighted average.

Class	TPR	FPR	PR	RC	FM	AUC
CS	0.988	0.014	0.534	0.988	0.693	0.998
ICS	0.603	0.005	0.458	0.603	0.521	0.990
O	0.984	0.004	1.000	0.984	0.992	0.999
WA	0.981	0.004	0.989	0.981	0.984	0.999

6.2 CSync in HTTP with ID looking strings

In this setup, we assume there is a simple HTTP pre-filter, keeping connections with ID looking strings for further investigation. This is a necessary step to reduce the run-time workload of the classifier, as connections relevant to the task are only ~20% of the overall HTTP workload. Then, the classifier has to decide which of the 3 classes match for each of the selected HTTP requests: 1) CSync, 2) *id-sharing but non-CSync*, 3) other. In this case, *other* refers to HTTP entries containing an ID-looking string, but are *not id-sharing*.

We perform two rounds of tests on one month's data: 1) train and test the algorithm using balanced data from the three classes, in a 10 cross-fold validation process. 2) train the algorithm on balanced data from the three classes (as in (1)), but test it on an unseen and unbalanced dataset which maintains the ratio of the three classes: CSync: 1.6%, *id-sharing but non-CSync*: 0.73% other: 97.67%.

As seen from the classification results (Table 8), the company name and parameter used are among the most important features; number of parameters is the worst. Non-ID related features allow the classifier to reach weighted AUC = 0.973, with a high weighted Precision and Recall across all classes. When all features are used, a weighted AUC = 0.978 is reached, similarly to the high importance feature set that disregards the number of parameters. Interestingly, when the classifier is trained on the balanced dataset, and tested on the unbalanced test set (last row of Table 8), the classifier can distinguish very well the three classes, with low error rates across all three classes, even though there is high imbalance in the classes. These results are further validated by the breakdown of performance per class, demonstrated in Table 9, which show high TP rate and low FP rate for all three classes independently.

Overall, the results show that it is possible to understand and model the patterns of CSync, as they are driven by particular types of companies, using specific parameters, etc. Therefore, an online classifier could be trained to provide insights as to what each HTTP connection is and how likely it is to be performing CSync, without the need to match the IDs to SET cookie actions.

7 RELATED WORK

CSync has become a commonplace on the Web. One of the first works to discuss this mechanism [29] studies programmatic auctions from a privacy perspective and presents CSync as an integral part of communication between the participating entities. The study identified over 100 CSync events while crawling the top 100 sites. In our study, we extend their detection mechanism to detect CSync when cookieID is piggybacked in either URL parameters or path.

In [1], authors conduct a CSync privacy analysis by studying a small dataset of 3000 crawled sites, in conjunction with re-spawning cookies and how, together, they affect the reconstruction of user's browsing history by trackers. In [32] they measure the advertising ecosystem cost to users. Focusing on user privacy and targeted advertising, they use CSync as a metric for anonymity loss, showing that users receive 3.4 CSyncs per ad-impression.

Papadopoulos et al. [33] present how CSync can wreck a secure browsing session. They show cases where 3rd-parties may leak a user's cookie IDs and browsing history, thus increasing the identifiability of the user to a snooping ISP. By probing the top 12k Alexa sites they find 1 out of 13 websites exposing their users to these privacy leaks even when they use TLS and secure VPN services. In a recent census by Englehardt et al. [11], authors measure CSync and its adoption in a small subset of 100,000 crawled sites, before highlighting the need of further investigation given its increased privacy implications. Their results show that 157 of top 200 (i.e. 78%) 3rd parties synchronize cookies with at least one other party.

In [16] they study the economics and the revenue implications of CSync from the point of view of an informed seller of advertising space, uncovering a trade-off between targeting and information leakage. Similarly, in [4], authors explore the role of data providers on the price and allocation of consumer-level information and develop a simple model of data pricing that captures the key trade-offs involved in selling information encoded in 3rd-party cookies. In [13] they investigate tracking groups that share user-specific identifiers in a dataset collected after recording the browsing history of 100 users for two weeks. In this dataset, they detect 660 ID-sharing groups and found domains with sensitive content (such as health-related) that shared IDs with domains related to ad-trackers.

In [2] they aim to enhance the transparency in ad ecosystem with regards to information sharing, developing a content-agnostic methodology to detect client- and server-side flows of information between ad exchanges by leveraging retargeted ads. By using crawled data, authors collected 35448 ad impressions and identified 4 different kinds of information sharing behaviour between ad exchanges. In [3], they study the diffusion of user tracking caused by RTB-based programmatic ad-auctions, considering CSync as the core component of such auctions and the primary factor of the diffusion of privacy leaks. Results of their study show that under specific assumptions, no less than 52 tracking companies can observe at least 91% of an average user's browsing history.

Contrary to these works, we conduct a first of its kind, full-scale study of CSync by analysing a year-long dataset of 850 real users, thus avoiding any biases from crawling websites using artificial personas. Additionally, since nowadays mobile drives the majority of the network traffic, our work is the first to study CSync in the growing ecosystem of mobile devices.

8 SUMMARY AND DISCUSSION

One of the most popular techniques for trackers to share the IDs they assign to users today is Cookie Synchronization, with which different domains can merge their databases in the background. However, syncing userIDs of a given user increases the user identifiability while browsing, thus reducing their overall anonymity on the Web. In this paper, we build CONRAD: a holistic system to detect CSync events, either when the synced IDs are available in plaintext, or even when they are obfuscated (i.e., hashed, encrypted). Using our detection mechanism, we are the first to explore CSync in the mobile ecosystem and the first to analyze it in depth, using a year-long dataset of real mobile users. CONRAD is able to capture 3.771% more CSync cases than related work.

Results: Our analysis of $263k$ CSync requests, syncing $22.3k$ unique userIDs, led to the following findings:

- 97% of users are exposed to CSync at least once. The median user is synced at least once within the first week of browsing.
- Ad-related domains participate in more than 75% of all CSync through the year, learning as much as 90% of all synced userIDs.
- Three companies learn more than 30% of all userIDs, each.
- The median userID gets leaked to 3.5 domains, on average.
- The average user receives around 1 synchronization per 68 HTTP requests, and gets up to 6.5 of their userIDs synced.
- The number of domains that learn about the median user after CSyncs grows by a factor of 6.75.
- We find 63 cases, where domains set cookies on the users' browsers with userIDs previously set by other domains. This universal identification model enables collaborating domains to share data without background database merges.
- We find 131 cases of domains storing in cookies their CSyncs results forming ID Summaries.
- 5% of users suffer from ID-spilling in their secure TLS traffic.
- several sensitive information (e.g., gender, birth dates) is passed to the syncing domain along with the userID.

In addition to the classic, heuristics-based method applied to collect the above findings, in this work, we proposed a novel, CSync detection mechanism able to detect at real time CSync events, even if the synced IDs are obfuscated. In particular, this online, machine learning classifier can be trained to provide insights as to what each HTTP connection is, and how likely it is to be performing CSync, without the need to match the IDs to SET cookie actions. We use the set of detected CSyncs from the heuristics-based method as ground truth, to train our machine learning, cookie-less detection algorithm. We were able to achieve high accuracy (84%-90%) and high AUC (0.89-0.97), when non-ID related features were used.

Countermeasures: Nowadays, the most popular defence mechanism of CSync is the use of the traditional ad-blockers. Indeed, since the vast majority (75%) of CSync takes place among ad-related domains (see Figure 14), it is easy to anticipate that by blocking ad-related requests, one can eliminate a large portion of the privacy leak that CSync causes. However, the all-out approach of ad-blockers causes significant harm on publishers' content monetization models, forcing some of them to deploy anti-adblocking mechanisms [21, 27, 28] and deny serving ad-blocking users [26, 41, 43].

Mitigating mechanisms against CSync require a more targeted blocking strategy, that would not blindly harm the current ad-ecosystem. Instead, by applying detection techniques such as CONRAD, and blocking the specific traffic which has been found to facilitate CSync, we believe that the harmful privacy leakage and loss of anonymity of users due to CSync can be avoided, without the dire consequences on publishers' business models.

Impact: User data collection and sharing activities done without users' explicit consent can be illegal with hefty penalties imposed to

companies involved, as described in the new EU regulations for protecting user personal data and online privacy (GDPR and e-Privacy). Thus, it is important to design practical web transparency tools such as CONRAD, readily available to privacy researchers, regulators and end-users to investigate personal data leakage and anonymity on the Web, due to 3rd parties' activities such as CSync [20].

ACKNOWLEDGEMENTS: The research leading to these results has received funding from EU's Marie Sklodowska-Curie and Horizon 2020 Research & Innovation Programme under grant agreements 690972, 786669 and 830927. The paper reflects only the authors' view and the Commission is not responsible for any use that may be made of the information it contains.

REFERENCES

[1] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. 2014. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. New York, NY, USA.

[2] Muhammad Ahmad Bashir, Sajjad Arshad, William Robertson, and Christo Wilson. 2016. Tracing information flows between ad exchanges using retargeted ads. In *Proceedings of the 25th USENIX Security Symposium*.

[3] Muhammad Ahmad Bashir and Christo Wilson. 2018. Diffusion of User Tracking Data in the Online Advertising Ecosystem. *Proceedings on Privacy Enhancing Technologies* 4 (2018), 85–103.

[4] Dirk Bergemann and Alessandro Bonatti. 2015. Selling cookies. *American Economic Journal: Microeconomics* 7, 3 (2015), 259–294.

[5] Juan Miguel Carrascosa, Jakub Mikians, Ruben Cuevas, Vijay Erramilli, and Nikolaos Laoutaris. 2015. I always feel like somebody's watching me: measuring online behavioural advertising. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*. ACM.

[6] Tom Chavez. 2010. Data: Deja Vu All Over Again? https://adexchanger.com/considering-digital/tom-chavez/.

[7] Mozilla Developer. 2018. Document.cookie - Web APIs. https://developer.mozilla.org/en-US/docs/Web/API/Document/cookie.

[8] Disconnect. 2019. A faster, safer Internet is one click away. https://disconnect.me/.

[9] Peter Eckersley. 2010. How Unique is Your Web Browser?. In *Proceedings of the 10th International Conference on PETS' 10*.

[10] Jo el van Bergen. 2017. Mixed content weakens HTTPS. https://developers.google.com/web/fundamentals/ security/prevent-mixed-content/what-is-mixed-content.

[11] Steven Englehardt and Arvind Narayanan. 2016. Online Tracking: A 1-million-site Measurement and Analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*.

[12] European Commission. 2018. What is personal data? https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-personal-data_en.

[13] Marjan Falahrastegar, Hamed Haddadi, Steve Uhlig, and Richard Mortier. 2016. *Tracking Personal Identifiers Across the Web*.

[14] Famlam Fanboy, MonztA and Khrin. 2018. EasyList - Overview. https://easylist.to/.

[15] Gemfury Dev Center. 2019. HTTPS: Is your URL string secure over SSL? https://gemfury.com/help/url-string-over-https/.

[16] Arpita Ghosh, Mohammad Mahdian, R. Preston McAfee, and Sergei Vassilvitskii. 2015. To Match or Not to Match: Economics of Cookie Matching in Online Advertising. *ACM Trans. Econ. Comput. 2015* (2015).

[17] Phillipa Gill, Vijay Erramilli, Augustin Chaintreau, Balachander Krishnamurthy, Konstantina Papagiannaki, and Pablo Rodriguez. 2013. Follow the Money: Understanding Economics of Online Aggregation and Advertising. In *Proceedings of the ACM SIGCOMM Conference on Internet Measurement Conference (IMC '13)*.

[18] Roberto Gonzalez, Claudio Soriente, and Nikolaos Laoutaris. 2016. User Profiling in the Time of HTTPS. In *Proceedings of the 2016 ACM SIGCOMM Conference on Internet Measurement Conference (IMC '16)*.

[19] Google Developers. 2018. Cookie Matching. https://developers.google.com/ad-exchange/rtb/cookie-guide.

[20] Costas Iordanou, Georgios Smaragdakis, Ingmar Poese, and Nikolaos Laoutaris. 2018. Tracing Cross Border Web Tracking. In *Proceedings of the Internet Measurement Conference 2018 (IMC '18)*.

[21] Umar Iqbal, Zubair Shafiq, and Zhiyun Qian. 2017. The ad wars: retrospective measurement and analysis of anti-adblock filter lists. In *Proceedings of the 2017 ACM SIGCOMM Conference on Internet Measurement Conference (IMC'17)*.

[22] Jacob Kleinman. 2018. Stop Using WhatsApp If You Care About Your Privacy. https://lifehacker.com/stop-using-whatsapp-if-you-care-about-your-privacy-1825719172.

[23] Paul J Leach, Tim Berners-Lee, Jeffrey C Mogul, Larry Masinter, Roy T Fielding, and James Gettys. 1999. Encoding Sensitive Information in URI's. https://tools.ietf.org/html/rfc2616#section-15.1.3.

[24] Bernard Marr. 2017. Where Can You Buy Big Data? Here Are The Biggest Consumer Data Brokers. https://www.forbes.com/sites/bernardmarr/2017/09/07/where-can-you-buy-big-data-here-are-the-biggest-consumer-data-brokers/.

[25] Jonathan Mayer. 2011. Tracking the Trackers: Microsoft Advertising. *The Center for Internet and Society* (2011).

[26] Brian Morrissey. 2015. Forbes starts blocking ad-block users. https://digiday.com/media/forbes-ad-blocking/.

[27] Muhammad Haris Mughees, Zhiyun Qian, and Zubair Shafiq. 2017. Detecting anti ad-blockers in the wild. *Proceedings on Privacy Enhancing Technologies* 2017, 3 (2017), 130–146.

[28] Rishab Nithyanand, Sheharbano Khattak, Mobin Javed, Narseo Vallina-Rodriguez, Marjan Falahrastegar, Julia E. Powles, Emiliano De Cristofaro, Hamed Haddadi, and Steven J. Murdoch. 2016. Adblocking and Counter Blocking: A Slice of the Arms Race. In *6th USENIX Workshop on Free and Open Communications on the Internet (FOCI 16)*.

[29] Lukasz Olejnik, Minh-Dung Tran, and Claude Castelluccia. 2014. Selling off User Privacy at Auction. In *21st Annual Symposium Network and Distributed System Security (NDSS'14)*.

[30] Kurt Opsahl and Rainey Reitman. 2013. The Disconcerting Details: How Facebook Teams Up With Data Brokers to Show You Targeted Ads. https://www.eff.org/deeplinks/2013/04/disconcerting-details-how-facebook-teams-data-brokers-show-you-targeted-ads.

[31] Elias P. Papadopoulos, Michalis Diamantaris, Panagiotis Papadopoulos, Thanasis Petsas, Sotiris Ioannidis, and Evangelos P. Markatos. 2017. The Long-Standing Privacy Debate: Mobile Websites vs Mobile Apps. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*.

[32] Panagiotis Papadopoulos, Nicolas Kourtellis, and Evangelos P. Markatos. 2018. The Cost of Digital Advertisement: Comparing User and Advertiser Views. In *Proceedings of the 27th International Conference on World Wide Web (WWW'18)*.

[33] Panagiotis Papadopoulos, Nicolas Kourtellis, and Evangelos P. Markatos. 2018. Exclusive: How the (Synced) Cookie Monster Breached My Encrypted VPN Session. In *Proceedings of the 11th European Workshop on Systems Security (EuroSec'18)*.

[34] Panagiotis Papadopoulos, Nicolas Kourtellis, Pablo Rodriguez Rodriguez, and Nikolaos Laoutaris. 2017. If You Are Not Paying for It, You Are the Product: How Much Do Advertisers Pay to Reach You?. In *Proceedings of the 2017 ACM SIGCOMM Conference on Internet Measurement Conference (IMC '17)*.

[35] Andrea Peterson. 2015. Bankrupt RadioShack wants to sell off user data. But the bigger risk is if a Facebook or Google goes bust. https://www.washingtonpost.com/news/the-switch/wp/2015/03/26/bankrupt-radioshack-wants-to-sell-off-user-data-but-the-bigger-risk-is-if-a-facebook-or-google-goes-bust/.

[36] Andrea Peterson. 2015. Bankrupt RadioShack wants to sell off user data. But the bigger risk is if a Facebook or Google goes bust. https://www.washingtonpost.com/news/the-switch/wp/2015/03/26/bankrupt-radioshack-wants-to-sell-off-user-data-but-the-bigger-risk-is-if-a-facebook-or-google-goes-bust/.

[37] Rainey Reitman. 2013. How To Opt Out of Receiving Facebook Ads Based on Your Real-Life Shopping Activity. https://www.eff.org/deeplinks/2013/02/howto-opt-out-databrokers-showing-your-targeted-advertisements-facebook.

[38] Matt Richtel. 2000. F.T.C. Moves to Halt Sale Of Database at Toysmart. http://www.nytimes.com/2000/07/11/business/ftc-moves-to-halt-sale-of-database-at-toysmart.html.

[39] samy.pl. 2014. Evercookie - virtually irrevocable persistent cookies. https://samy.pl/evercookie/.

[40] Judy Selby. 2016. The Impact of Big Data Decisions on Business Valuations. https://datafloq.com/read/impact-big-data-decisions-business-valuation.

[41] Nicola Smith. 2016. How publishers are turning up the heat in the ad-blocking war. https://www.theguardian.com/media-network/2016/sep/02/publishers-ad-block-users-hide-content.

[42] Statista Inc. 2018. Percentage of all global web pages served to mobile phones from 2009 to 2018. https://www.statista.com/statistics/241462/global-mobile-phone-website-traffic-share.

[43] The Editors of Wired. 2016. How WIRED Is Going to Handle Ad Blocking. https://www.wired.com/how-wired-is-going-to-handle-ad-blocking/.

[44] Narseo Vallina-Rodriguez, Srikanth Sundaresan, Abbas Razaghpanah, Rishab Nithyanand, Mark Allman, Christian Kreibich, and Phillipa Gill. 2016. Tracking the trackers: Towards understanding the mobile advertising and tracking ecosystem. *arXiv preprint arXiv:1609.07190* (2016).

[45] World Wide Web Consortium (W3C). 2010. Same Origin Policy. https://www.w3.org/Security/wiki/Same_Origin_Policy.

[46] Zhonghao Yu, Sam Macbeth, Konark Modi, and Josep M. Pujol. 2016. Tracking the Trackers. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*.

Adversarial Sampling and Training for Semi-Supervised Information Retrieval

Dae Hoon Park
Huawei Research America
California, United States
dae.hoon.park@huawei.com

Yi Chang
School of Artificial Intelligence, Jilin University
Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education
China
yichang@jlu.edu.cn

ABSTRACT

Ad-hoc retrieval models with implicit feedback often have problems, *e.g.*, the imbalanced classes in the data set. Too few clicked documents may hurt generalization ability of the models, whereas too many non-clicked documents may harm effectiveness of the models and efficiency of training. In addition, recent neural network-based models are vulnerable to adversarial examples due to the linear nature in them. To solve the problems at the same time, we propose an adversarial sampling and training framework to learn ad-hoc retrieval models with implicit feedback. Our key idea is (i) to augment clicked examples by adversarial training for better generalization and (ii) to obtain very informational non-clicked examples by adversarial sampling and training. Experiments are performed on benchmark data sets for common ad-hoc retrieval tasks such as Web search, item recommendation, and question answering. Experimental results indicate that the proposed approaches significantly outperform strong baselines especially for high-ranked documents, and they outperform IRGAN in NDCG@5 using only 5% of labeled data for the Web search task.

CCS CONCEPTS

• **Information systems** → **Retrieval models and ranking**; • **Computing methodologies** → *Learning from implicit feedback*; *Semi-supervised learning settings.*

KEYWORDS

ad-hoc retrieval, adversarial training, implicit feedback, adversarial sampling

ACM Reference Format:
Dae Hoon Park and Yi Chang. 2019. Adversarial Sampling and Training for Semi-Supervised Information Retrieval. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313416

1 INTRODUCTION

Ad-hoc retrieval systems provide a ranked list of documents given a query, in which a user's information need is expressed. Such ad-hoc retrieval systems prevail in our daily lives, from Web search and question answering to product recommendation, which can be regarded as ad-hoc retrieval with users being queries and products being documents. Modern ad-hoc retrieval models are usually supervised or semi-supervised with training data to learn the relevance of documents to a query, because of its outstanding performance over unsupervised models.

Generally speaking, the training data are obtained for (semi-)supervised ad-hoc retrieval models in two ways. Annotators can explicitly label documents with their relevance to a query, and such labels are called explicit feedback. It is, however, often too expensive to obtain enough explicit feedback, and the labels may contain bias from a few annotators. Instead, labels can be inferred from the user's reactions (*e.g.*, clicks and views) on the given documents, and such reactions are called implicit feedback. Implicit feedback is cheap and reflects preference of actual users, so it has been widely studied [17, 26, 32] and adopted in the industry. Typically, there are much more non-clicked documents than clicked ones. Too few clicked documents can be problematic for generalization of the learned models. On the other hand, too many non-clicked documents can slow down the training and harm effectiveness of the trained model. Especially for ad-hoc retrieval tasks, the majority of the non-clicked documents is often not informational, sampling informational non-clicked documents is important for efficiency and effectiveness [28, 42].

Meanwhile, ad-hoc retrieval models often have a problem in themselves; their structures are vulnerable to adversarial examples. That is, due to the linearity in many models, the output can be dramatically changed by infinitesimal changes in input dimensions [8]. Traditional models such as logistic regression and matrix factorization work in a very linear way, and even recent deep neural networks are also designed to work in a quite linear way [9] (*e.g.*, ReLU [7] and LSTMs [15]).

To solve the aforementioned problems in implicit feedback and ad-hoc retrieval models, we propose an adversarial sampling-based adversarial training framework. On one hand, we generate adversarial positive examples to augment labeled (or clicked) documents with informational ones. On the other hand, to obtain informational negative examples, we first sample difficult examples adversarially from unlabeled (or non-clicked) documents. Then, we further generate adversarial negative examples, which are even more informational, on top of the sampled negative examples. The generated adversarial examples are supposed to be informational and to remove the weakness in the model's linearity. We also propose virtual adversarial training-based approach, which does not require labels to generate adversarial examples and thus suitable

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313416

for semi-supervised learning, and its variant that is more efficient and effective.

We perform experiments on benchmark data sets for three popular tasks: Web search, item recommendation, and question answering. Our proposed approaches are mainly compared with a strong baseline, IRGAN [36], which adopts a different kind of adversarial training to sample negative documents. Experimental results indicate that our proposed approaches are very effective on ad-hoc retrieval tasks and significantly outperform baselines especially on Web search and item recommendation. Moreover, our proposed approaches outperform IRGAN in NDCG@5 with only 5% of the labeled data for Web search.

Our contributions in this paper include:

- We propose a novel framework for learning ad-hoc retrieval models with implicit feedback. We generate very informational training examples by adversarial sampling and training. To the best of our knowledge, there has been no research work that incorporates adversarial sampling with adversarial training for ad-hoc retrieval models.
- We also propose virtual adversarial training and its variation for ad-hoc retrieval models. They can generate adversarial examples without labels, which may be ideal for noisy implicit feedback.
- We perform experiments on benchmark data sets for popular tasks. Experimental results indicate the proposed approaches significantly outperform strong baselines. The proposed approaches are empirically shown to be data-efficient.

2 RELATED WORK

2.1 Adversarial Training

It was found in [34] that several machine learning models, including modern state-of-the-art neural networks, fail with adversarial examples. A slight adversarial perturbation in the original input was enough to fake the models, meaning that the models classify the adversarial example into a wrong class with high confidence. Then, Goodfellow *et al.* hypothesized that the vulnerability of the models come from the linear nature in the models [9]. That is, the adversarial perturbation in each dimension adds up to a great change in the output due to some extent of linearity in the models. Goodfellow *et al.* also suggested to train a model by learning adversarial examples as well as original examples, where the adversarial examples were generated with the proposed *fast gradient sign method*. The trained model was reported more robust to adversarial examples. Adversarial examples and adversarial training are surveyed in [37]. Meanwhile, the earlier adversarial training approaches required labels of examples, and this is not ideal for semi-supervised learning where there are much more unlabeled data than labeled data. Miyato *et al.* [23] proposed virtual adversarial training, where labels are not required to generate adversarial examples. With its strength in semi-supervised learning, it has been applied to semi-supervised text classification [21] and image classification [22].

We adopt the ideas of adversarial training and virtual adversarial training in our approaches. However, there are several differences between our approaches and them. In our task, we assume that training data in implicit feedback consist of relatively fewer single-class labeled data and much more unlabeled data, where there exist many non-informational unlabeled examples. Such informativeness of unlabeled examples is not studied in the original works. We pay attention to it in this paper, especially for ad-hoc retrieval with implicit feedback. We further combine adversarial sampling with adversarial training to boost the informativeness of unlabeled examples. Meanwhile, virtual adversarial training iterates over all unlabeled examples and thus may not be efficient and effective for implicit feedback. Hence, we propose selective virtual adversarial training that iterates over only labeled examples and adversarially sampled unlabeled examples. Our problem, ad-hoc retrieval, also makes our approaches different from original ones. In our problem, there can be two input variables, queries and documents, instead of one single variable. In addition, we apply adversarial training to a pair-wise learning-to-rank framework.

2.2 Adversarial Training for Ad-hoc Retrieval

Recently, deep neural networks-based approaches have been successfully adopted in information retrieval tasks such as Web search [11, 16], click modeling [1], and query suggestion and auto-completion [24, 33]. Adversarial training has recently gained popularity with neural networks, and a few adversarial training-based approaches have been proposed for ad-hoc retrieval tasks. For example, an idea of generative adversarial networks (GAN) [8] has been adopted to unify generative and discriminative information retrieval models by IRGAN [36]. Its goal is indeed similar to ours, which is to build effective ad-hoc retrieval models through difficult examples. The generator of IRGAN tries to fool the discriminator by providing adversarial negative examples while the discriminator tries to distinguish them from true positive examples so that the generator and the discriminator can mutually enhance. Through dynamically sampling more and more difficult examples by the evolving generator, IRGAN achieves outstanding performance. However, IRGAN is different from our approaches in many aspects. Although it contains a generator, IRGAN does not really generate negative examples but samples them from unlabeled data according to the generator model. On the other hand, our approach *generates* difficult examples on top of existing examples by adding adversarial perturbation to them. Incorporating adversarial sampling with adversarial training, our approaches can generate even more difficult negative examples based on the adversarially sampled negative examples. Also, our approaches require one single model whereas IRGAN consists of two models: a generator and a discriminator; this means the required number of parameters and hyperparameters can double, and training both models at the same time can be more difficult. Furthermore, IRGAN requires a pre-trained model to ensure stability during training, unlike our stand-alone model. [39] extends IRGAN with a different neural network architecture for question answering.

More recently, a simultaneous work but independent from ours has proposed adversarial training for item recommendation [13]. It employs adversarial perturbation to build a robust model, and the experiment results show that it is superior to existing models. Despite its novel approach, our work is different from it in several aspects. Their approach adds adversarial perturbation to the model parameters while we add it to the input. Also, they do not have the same assumption on data as ours, *i.e.*, single-class labeled data

and much more unlabeled data, so they do not focus on sampling difficult negative examples from unlabeled data. Their focus is on item recommendation, but our focus is on ad-hoc retrieval with implicit feedback. Lastly, we further explore virtual adversarial training that can be promising for semi-supervised learning, which is not in their scope.

2.3 Negative Sampling for Ad-hoc Retrieval

Negative example sampling techniques have been adopted for information retrieval tasks as well as natural language processing tasks. Word embedding models have sampled negative examples by their frequency [10, 20], and [27] employs "max sampling" that samples negative examples that are most similar to positive examples for question answering. Generative Adversarial Networks-based negative sampling also has been proposed for ad-hoc retrieval tasks [36].

Negative sampling techniques have been studied more extensively for item recommendation. To train a model efficiently, negative sampling techniques have been employed. Traditional models including Bayesian Personalized Ranking [29] rely on uniform sampling for negative examples. Dynamic negative sampling techniques [28, 42] that sample informational negative examples for the current model also have been proposed. Recently, negative sampling by leveraging view information has been proposed and empirically shown to be effective for e-commerce data sets [5]. Hidasi and Karatzoglou [14] recently proposed a BPR-max loss function that assigns larger weights to more informative negative examples. To the best of our knowledge, no previous ad-hoc retrieval models adversarially generate negative examples on top of sampled negative examples in order to generate even more informational examples, as our approaches do.

3 PROBLEM DEFINITION

We study a typical ad-hoc retrieval problem. Given a query q that contains a user's information need and a set of documents $D = \{d_i\}_{i=1}^{M}$, the goal is to retrieve and rank documents based on their estimated relevance to q, where Probability Ranking Principle is assumed. Ad-hoc retrieval problems are not limited to document search, and q and D may be in other types. Item recommendation can be regarded as an ad-hoc retrieval problem where q is a user and D is a set of items. Question answering (retrieval-based) also can fall within ad-hoc retrieval where q is a question and D is a set of answer candidates.

We specifically study semi-supervised ad-hoc retrieval with implicit feedback, where for each q, we have a set of labeled (clicked) documents, which we assume relevant, and unlabeled (non-clicked) documents. We also assume the number of labeled documents is much less than that of unlabeled documents. Indeed, such configuration is common in the industry. When the retrieved documents are shown to users, the users usually click only a few of them while many documents are not clicked. The non-clicked documents are not necessarily irrelevant to q, so they are left unlabeled or sometimes labeled as *viewed*.

The labeled and unlabeled documents for q are given to a model as training data, whose example is in a form of triple (q, d, y), where y is a relevance of d to q, and $y = 1$ if a user clicked d for q. Non-relevant documents ($y = 0$) come from unlabeled documents. The ad-hoc retrieval model learns a function $f_{\boldsymbol{\theta}} : (q, d) \mapsto y$, where $\boldsymbol{\theta}$ is a set of model parameters. The learned function $f_{\boldsymbol{\theta}}$ is then employed to predict a relevance score of a test (q, d) pair.

There are challenges to learn ad-hoc retrieval models from implicit feedback. If the number of clicked documents is small, it can easily suffer from over-fitting. That means, the model will not generalize well on the test data. On the other hand, there are relatively too many unlabeled documents, and many of the unlabeled documents are either redundant or obviously irrelevant. That is, the unlabeled documents are not informational so that the model will not learn effectively with them. Therefore, careful utilization of the imbalanced data is desired to build an effective ad-hoc retrieval model.

4 ADVERSARIAL SAMPLING AND TRAINING FRAMEWORK FOR AD-HOC RETRIEVAL

In order to address the challenges in implicit feedback, we propose a framework of adversarial sampling and training that are applied differently for labeled and unlabeled documents. On one hand, we employ adversarial training to build a generalizable model with relatively few labeled documents. On the other hand, we first adversarially sample informational examples, and on top of them, we further generate even more informational examples by adversarial training. The generated adversarial examples for both labeled and unlabeled documents also help ad-hoc retrieval models cope with the weakness in linearity of the models.

In this section, we first explain how adversarial examples can cause a significant change in the output of models that have linearity. We then propose multiple adversarial training methods that can generate informational examples based on existing training examples. Then, adversarial sampling is incorporated with adversarial training in order to amplify informativeness of unlabeled examples. Pairwise learning is also proposed to effectively learn with implicit feedback.

4.1 Adversarial Examples for Ad-hoc Retrieval

An adversarial example is defined as an example that is slightly perturbed from the original example but greatly changes the activation and thus the output. Adversarial examples were first introduced for neural networks in [34] and later found that they occur due to linearity in models [9]. For an adversarial input $\tilde{\mathbf{x}} = \mathbf{x} + \boldsymbol{\eta}$, where $\mathbf{x}$ is an original input vector and $\boldsymbol{\eta}$ is an adversarial perturbation vector of the same shape, a dot product between $\tilde{\mathbf{x}}$ and a weight vector $\mathbf{w}$ becomes

$$\mathbf{w}^\intercal \tilde{\mathbf{x}} = \mathbf{w}^\intercal \mathbf{x} + \mathbf{w}^\intercal \boldsymbol{\eta}.$$

The change of the activation by adversarial perturbation is thus $\mathbf{w}^\intercal \boldsymbol{\eta}$. If the average magnitude of $\mathbf{w}$ and $\boldsymbol{\eta}$ are m and ϵ, respectively, then the maximum change caused by perturbation can grow linearly by ϵmn, where n is the number of dimensions in $\mathbf{x}$. (We explain how to approximate the optimal $\boldsymbol{\eta}$ in the next section.) Hence, small changes by perturbation can accumulate with dimensions to a great change. Such a great change can propagate to an output of more complicated models. For example, neural network activation

functions such as ReLUs [7] and sigmoid, which are widely used in modern neural networks, are piece-wise linear or almost linear (in non-saturating section); hence, the great change can easily propagate to the final output [9].

Such perturbations can also exist in ad-hoc retrieval systems. For example, let's assume $\mathbf{x}$ and $\mathbf{x}'$ are topic probabilities estimated by topic models for two semantically identical documents, where only $\mathbf{x}$ is in the training data. A slight word change from $\mathbf{x}$ to $\mathbf{x}'$ can cause slight difference in each dimension of $\mathbf{x}$ and $\mathbf{x}'$, and this may result in a large change in the output, yielding wrong prediction on $\mathbf{x}'$. Similarly, less processed features such as raw text and TF-IDF values can also cause such effect. For example, two semantically identical documents may have different raw text features, by synonyms of their words. Although their raw features are different, their latent vectors may still be similar, but the subtle difference in the latent vectors may eventually result in a great change in the output. This is because the latent vectors still need to go through activation functions that are linear or almost linear.

Unfortunately, recent ad-hoc retrieval models as well as several traditional models are often vulnerable to adversarial examples. Neural networks are recently employed for state-of-the-art performance, and modern neural networks behave in a linear way (*e.g.*, ReLUs) for more effective learning by avoiding vanishing gradients. Earlier models such as RankNet [2], or matrix factorization [18] also operate in a quite linear way.

4.2 Adversarial Training for Robust Ad-hoc Retrieval Models

4.2.1 Adversarial Training. We propose to train ad-hoc retrieval models with adversarial examples that can locate the weak points in the models. In general, ad-hoc retrieval models are trained by minimizing the following cost function

$$\mathbb{E}_{q,d,y\sim p_{\text{data}}}\mathrm{J}(\mathbf{q},\mathbf{d},y;\boldsymbol{\theta}) \tag{1}$$

where $\mathbf{q}$ and $\mathbf{d}$ are feature vectors for q and d, respectively, and $y \in \{0,1\}$ is a relevance label, and $\boldsymbol{\theta}$ is a set of model parameters.[1] J can be, for example, a cross entropy loss function. Hence, the goal is to learn a function $f_{\boldsymbol{\theta}} : (\mathbf{q},\mathbf{d}) \mapsto y$. Intuitively, in order to build a model that is robust to adversarial perturbation, we can generate adversarial perturbation and let the model learn from adversarial examples that are generated by adding adversarial perturbation to original training examples. That is, the adversarial examples still keep the original labels, but their $\mathbf{q}$ and $\mathbf{d}$ are modified to become more difficult (*i.e.*, yield greater losses). Learning with more difficult examples that attack the model's weaknesses, the model can become more robust to adversarial perturbation. Therefore, we can add a cost for adversarial examples to the objective as follows:

$$\mathbb{E}_{q,d,y\sim p_{\text{data}}}\mathrm{J}(\mathbf{q},\mathbf{d},y;\boldsymbol{\theta}) + \alpha\mathrm{J}(\mathbf{q}+\boldsymbol{\eta}_q,\mathbf{d}+\boldsymbol{\eta}_d,y;\boldsymbol{\theta}) \tag{2}$$

where $\boldsymbol{\eta}_q$ and $\boldsymbol{\eta}_d$ are adversarial perturbations for $\mathbf{q}$ and $\mathbf{d}$, respectively, and α is a hyperparameter, which is set to 1 in this work. This objective means that regardless of adversarial perturbation, the model should learn to predict the same relevance.

[1] If the given data consist of one input vector $\mathbf{x}$ instead of two vectors $\mathbf{q}$ and $\mathbf{d}$ for a query and a document as in Section 4.1, $\mathbf{x}$ can replace $\mathbf{q}$ and $\mathbf{d}$ accordingly.

4.2.2 Generating Adversarial Perturbation. When generating adversarial perturbation, the magnitudes of perturbation vectors need to be limited so that the adversarial examples do not become too similar to examples of the opposite classes. Thus, their magnitudes are limited by a hyperparameter ϵ such that $||\boldsymbol{\eta}||_p < \epsilon$. In order to build a model that is robust to perturbation, we need to generate a perturbation vector that can achieve the greatest loss within the limit of ϵ. In other words, a model trained with more difficult examples can be more effective for the unseen test examples. Such perturbation vectors are defined as

$$\boldsymbol{\eta}_q, \boldsymbol{\eta}_d = \underset{\boldsymbol{\eta}_q:||\boldsymbol{\eta}_q||_p<\epsilon,\ \boldsymbol{\eta}_d:||\boldsymbol{\eta}_d||_p<\epsilon}{\arg\max} \mathrm{J}(\mathbf{q}+\boldsymbol{\eta}_q,\mathbf{d}+\boldsymbol{\eta}_d,y;\hat{\boldsymbol{\theta}}). \tag{3}$$

We use the same ϵ for $\boldsymbol{\eta}_q$ and $\boldsymbol{\eta}_d$ in this paper, but different values can be used. Also, $\hat{\boldsymbol{\theta}}$ denotes a copy of $\boldsymbol{\theta}$, in order to avoid propagating gradients from this perturbation generation process to $\boldsymbol{\theta}$. Note that the objective in (2) with the equation (3) can be interpreted as a *minimax* game. Similar to the *fast gradient sign method* [9], with first-order Taylor series approximation, equation (3) is approximated as

$$\begin{aligned}\boldsymbol{\eta}_q &= \underset{\boldsymbol{\eta}_q:||\boldsymbol{\eta}_q||_p<\epsilon}{\arg\max}\ \mathrm{J}(\mathbf{q},\mathbf{d},y;\hat{\boldsymbol{\theta}}) + \boldsymbol{\eta}_q\nabla_{\mathbf{q}}\mathrm{J}(\mathbf{q},\mathbf{d},y;\hat{\boldsymbol{\theta}})\\ \boldsymbol{\eta}_d &= \underset{\boldsymbol{\eta}_d:||\boldsymbol{\eta}_d||_p<\epsilon}{\arg\max}\ \mathrm{J}(\mathbf{q},\mathbf{d},y;\hat{\boldsymbol{\theta}}) + \boldsymbol{\eta}_d\nabla_{\mathbf{d}}\mathrm{J}(\mathbf{q},\mathbf{d},y;\hat{\boldsymbol{\theta}}).\end{aligned} \tag{4}$$

Its solution depends on the value of p in the p-norm as follows

$$\boldsymbol{\eta}_q = \begin{cases}\epsilon\,\mathrm{sign}(\mathbf{g}_q) & \text{if } p=\infty\\ \epsilon\frac{\mathbf{g}_q}{||\mathbf{g}_q||_2} & \text{if } p=2\end{cases} \quad \text{where } \mathbf{g}_q = \nabla_{\mathbf{q}}\mathrm{J}(\mathbf{q},\mathbf{d},y;\hat{\boldsymbol{\theta}}) \tag{5}$$

where $\mathbf{g}_q$ can be efficiently computed by backpropagation while solving the first term of (2). That means, adversarial training requires only a few additional computations that can be done efficiently. $\boldsymbol{\eta}_d$ can be solved in the same way, so we do not include its solution here. Regarding the choice of p, a max norm ($p=\infty$) is used in [9] because the magnitudes of perturbation in image pixels are supposed to be small. However, we do not have such assumption for text; indeed, semantically similar documents can have very different text at least in their raw representation. Also, L2 norm can sometimes perform better than max norm for adversarial training [22], so we employ L2 norm in this work.

4.2.3 Virtual Adversarial Examples. The process for generating adversarial examples requires labels of query-document pairs. However, the number of unlabeled (non-clicked) documents is usually much greater than that of labeled (clicked) documents in implicit feedback. Even if we regard unlabeled ones as negative examples, the generated adversarial examples from the negative examples may not be ideal due to the uncertainty in the assumed labels. Virtual Adversarial Examples [23] may thus be useful in this case, which do not require labels to generate adversarial examples. The following cost function J_{KL} is added to the objective in (1) for both labeled and unlabeled data:

$$\begin{aligned}&\mathrm{J}_{\mathrm{KL}}(\mathbf{q},\mathbf{d};\boldsymbol{\theta}) = \mathrm{KL}\left[p(\cdot|\mathbf{q},\mathbf{d};\boldsymbol{\theta})||p(\cdot|\mathbf{q}+\boldsymbol{\eta}_q,\mathbf{d}+\boldsymbol{\eta}_d;\boldsymbol{\theta})\right], \text{ where}\\ &\boldsymbol{\eta}_q,\boldsymbol{\eta}_d = \underset{\boldsymbol{\eta}_q:||\boldsymbol{\eta}_q||<\epsilon,\ \boldsymbol{\eta}_d:||\boldsymbol{\eta}_d||<\epsilon}{\arg\max}\ \mathrm{KL}\left[p(\cdot|\mathbf{q},\mathbf{d};\hat{\boldsymbol{\theta}})||p(\cdot|\mathbf{q}+\boldsymbol{\eta}_q,\mathbf{d}+\boldsymbol{\eta}_d;\hat{\boldsymbol{\theta}})\right]\end{aligned} \tag{6}$$

where $\mathrm{KL}\left[p||p'\right]$ denotes KL divergence between conditional relevance distributions p and p'. Basically, minimizing the objective

including this cost function means that we want to reduce the distribution difference that is caused by adversarial perturbation. In other words, we want to enhance the model's local smoothness of conditional relevance distribution so that the model's output does not dramatically change by adversarial perturbation. The perturbation vector can be computed by a second-order Taylor series approximation and a single iteration of power method on the KL divergence function as in [23]. The solution thus can be approximated as

$$\begin{aligned} \boldsymbol{\eta}_q &= \epsilon \frac{\mathbf{g}_q}{||\mathbf{g}_q||_2} \ , \ \boldsymbol{\eta}_d = \epsilon \frac{\mathbf{g}_d}{||\mathbf{g}_d||_2}, \text{ where} \\ \mathbf{g}_q &= \nabla_{\mathbf{q}+\mathbf{e}_q} \mathrm{KL}\Big[p(\cdot|\mathbf{q}, \mathbf{d}; \hat{\boldsymbol{\theta}}) || p(\cdot|\mathbf{q}+\mathbf{e}_q, \mathbf{d}; \hat{\boldsymbol{\theta}})\Big] \\ \mathbf{g}_d &= \nabla_{\mathbf{d}+\mathbf{e}_d} \mathrm{KL}\Big[p(\cdot|\mathbf{q}, \mathbf{d}; \hat{\boldsymbol{\theta}}) || p(\cdot|\mathbf{q}, \mathbf{d}+\mathbf{e}_d; \hat{\boldsymbol{\theta}})\Big] \end{aligned} \tag{7}$$

where $\mathbf{e}_q$ and $\mathbf{e}_d$ are small random vectors. This process does not require labels, so the uncertainty of relevance in implicit feedback is not a problem to generate perturbations, and thus all non-clicked documents can be safely used for semi-supervised learning. However, training with all non-clicked documents can be very inefficient, so we discuss sampling approaches in Section 4.3.

4.2.4 Adversarial Examples for Discrete Input. If an input is a discrete value such as a word or an item ID, where the value is converted to a latent vector, one may consider to add a perturbation vector to the latent vector (or embedding vector). However, adding it to the perturbation vectors may be tricky because they may learn to increase their magnitudes so that the amount of perturbation ϵ becomes negligible. To avoid such phenomenon, it was proposed in [21] to normalize the latent vector after adding a perturbation vector to it. Meanwhile, ad-hoc retrieval models can have continuous input as well as discrete one, *e.g.,* TF-IDF features and other pre-processed features. To serve both discrete and continuous input, we employ another trick that adds adversarial perturbation to the input. When the input is continuous, we can add the perturbation vector as usual: $\tilde{\mathbf{x}} = \mathbf{x} + \boldsymbol{\eta}$. For discrete input, we are given $\mathbf{x}$ that is a length-v one-hot encoded input vector, where v is the cardinality. Instead of looking up the length-k embedding vector $\mathbf{z} \in \mathbb{R}^k$ from the embedding matrix $\mathbf{Z} \in \mathbb{R}^{v \times k}$, we multiply $\tilde{\mathbf{x}}$ and $\mathbf{Z}$ together to obtain the perturbed embedding vector of $\mathbf{x}$. That is, the perturbed embedding vector $\tilde{\mathbf{z}}$ is obtained by

$$\tilde{\mathbf{z}} = (\mathbf{x} + \boldsymbol{\eta})\mathbf{Z}. \tag{8}$$

That means, for example, we make a mixture of words to generate a perturbed embedding vector for the input word. The advantage of this approach is that it can be extended to accommodate the input that is combination of discrete values and continuous values. In addition, this approach may potentially provide more interpretable perturbation as it is engaged to the original input. For example, a perturbation by "`- man + woman`" for the input "`king`" will indicate that the perturbation changes the gender to generate the embedding vector of "`queen`". Such interpretability may be interesting since that of neural networks-based classifiers has been weak and has attracted attention [38, 41]. We do not explore these advantages in this paper as they are out of its scope, but they are left for our future work.

4.3 Adversarial Training with Adversarial Sampling

Adversarial training can be interpreted as learning with difficult examples that are generated to attack the current model's weakness. To amplify the effectiveness, we propose adversarial sampling-based adversarial training that generates even more difficult examples from already difficult examples. In ad-hoc retrieval systems with implicit feedback, labeled documents are mainly relevant but the number of documents is relatively small. On the other hand, unlabeled documents are more likely to be negative than positive while the number of documents is much greater. As many of the unlabeled examples may be either redundant or obviously irrelevant, finding informational examples may be the key to effective and efficient model.

Therefore, we use existing labeled documents as positive examples, but we adversarially sample negative examples from unlabeled documents. The optimization objective for adversarial training is thus defined as

$$\begin{aligned} \sum_q \Big(&\mathbb{E}_{d \sim p_{\text{data}}(d|q, y=1)} \Big[\mathrm{J}(\mathbf{q}, \mathbf{d}, y=1; \boldsymbol{\theta}) + \mathrm{J}(\mathbf{q}+\boldsymbol{\eta}_q, \mathbf{d}+\boldsymbol{\eta}_d, y=1; \boldsymbol{\theta}) \Big] + \\ &\mathbb{E}_{d \sim p_{\boldsymbol{\theta}}(d|q, y=1)} \Big[\mathrm{J}(\mathbf{q}, \mathbf{d}, y=0; \boldsymbol{\theta}) + \mathrm{J}(\mathbf{q}+\boldsymbol{\eta}_q, \mathbf{d}+\boldsymbol{\eta}_d, y=0; \boldsymbol{\theta}) \Big] \Big) \end{aligned} \tag{9}$$

where $\mathbf{q}$ and $\mathbf{d}$ are feature vectors for q and d, respectively, and the adversarial perturbation vectors $\boldsymbol{\eta}_q$ and $\boldsymbol{\eta}_d$ are computed as in Section 4.2 for each example. All positive examples are selected from labeled data by p_{data}, whose distribution is uniform, and the negative examples are selected from unlabeled data by p_θ. In practice, the model goes through all positive examples in the labeled data while it goes through the same number of negative examples that are sampled from unlabeled data. Negative examples are sampled by $p_{\boldsymbol{\theta}}(d|q, y=1)$ to ensure the examples are adversarial ($y=1$) so that they are difficult for the current model. Note that the negative examples are sampled dynamically for the current model, and more efficient sampling can be done by estimating $p_{\boldsymbol{\theta}}(d|q, y=1)$ for every K epochs or estimating it for only C document candidates. The conditional probability for sampling d can be estimated by the following softmax function:

$$p_{\boldsymbol{\theta}}(d|q, y=1) = \frac{\exp(f_{\boldsymbol{\theta}}(\mathbf{q}, \mathbf{d})/\tau)}{\sum_{d'} \exp(f_{\boldsymbol{\theta}}(\mathbf{q}, \mathbf{d}')/\tau)} \tag{10}$$

where τ is a temperature hyperparameter for controlling smoothness of the distribution. A lower temperature will assign most of the probability mass to a fewer documents while a higher temperature will make the distribution more uniform. We employ cross entropy loss for J, which is defined as

$$\begin{aligned} &\mathrm{J}(\mathbf{q}, \mathbf{d}, y; \boldsymbol{\theta}) = -\log p(y|q, d; \boldsymbol{\theta}) \ \ , \text{where} \\ &p(y=1|q, d; \boldsymbol{\theta}) = \sigma(f_{\boldsymbol{\theta}}(\mathbf{q}, \mathbf{d})) \ , \ p(y=0|q, d; \boldsymbol{\theta}) = 1 - \sigma(f_{\boldsymbol{\theta}}(\mathbf{q}, \mathbf{d})) \end{aligned} \tag{11}$$

where σ is a sigmoid function, and $f_{\boldsymbol{\theta}}$ is defined in Section 5 for each ad-hoc retrieval task.

The idea of adversarial sampling is not actually new. Dynamic negative sampling ideas have been proposed by [28, 42], which sample the most informational negative items at the moment. A generator model of IRGAN [36] also plays a similar role. It builds the generator that can adversarially fool the discriminator, and it

samples the negative examples according to the estimated generator. Our framework is different from theirs in that we further generate more difficult adversarial negative examples from the adversarially sampled negative examples, and in that we generate adversarial examples even for positive class for better generalization.

Selective Virtual Adversarial Training. Although the advantage of virtual adversarial training is that it can be used for all unlabeled documents, which is good for semi-supervised learning, it may not be effective and efficient to process many non-informational unlabeled documents. Therefore, we propose selective virtual adversarial training based on adversarial sampling. Instead of generating virtual adversarial perturbation for all unlabeled documents, we selectively generate them for difficult ones. The objective is defined as

$$\sum_q \Big(\mathbb{E}_{d\sim p_{\text{data}}(d|q,y=1)} \Big[\mathrm{J}(\mathbf{q},\mathbf{d},y=1;\boldsymbol{\theta}) + \mathrm{J}_{\mathrm{KL}}(\mathbf{q},\mathbf{d};\boldsymbol{\theta}) \Big] + \mathbb{E}_{d\sim p_{\boldsymbol{\theta}}(d|q,y=1)} \Big[\mathrm{J}(\mathbf{q},\mathbf{d},y=0;\boldsymbol{\theta}) + \mathrm{J}_{\mathrm{KL}}(\mathbf{q},\mathbf{d};\boldsymbol{\theta}) \Big] \Big) \tag{12}$$

where J_{KL} is defined in equation (6). Similar to equation (9), we sample one negative document from unlabeled data by $p_{\boldsymbol{\theta}}$ for each positive document. This is more efficient than the original virtual adversarial training [23] as it does not iterate over all unlabeled documents. In addition, it can also be more effective because only highly informational documents, which may result in a better decision boundary, are selected to train the model. Indeed, we empirically find that this approach is more effective than the original one in Section 6.

4.4 Pairwise Training

Instead of collecting relevance level for each query-document pair, it is often easier for ad-hoc retrieval systems to infer which documents are more preferred than other documents for a query, leveraging implicit feedback of users. In addition, users have biases, so one user's rating is often not compatible with another user, for example, in item recommendation. Hence, it has been studied how to exploit such relative preference instead of absolute relevance with pairwise or listwise learning algorithms [19, 29]. We thus extend our approaches to pairwise training.

There are several ways to form document pairs for pair-wise training. For example, for a query q, we can build document pairs such that $\{(d_i,d_j)|d_i \in D_c \wedge d_j \in D_{nc}\}$ where D_c is a set of documents clicked by users and D_{nc} is a set of documents that are not clicked. There exist better ways to form such pairs, but they are not within the scope of this work. The pairwise objective corresponding to equation (9) can be defined as

$$\sum_q \Big(\mathbb{E}_{d^+\sim p_{\text{data}}(d|q,y=1)} \Big[\mathbb{E}_{d^-\sim p_{\boldsymbol{\theta}}(d|q,y=1)} \Big[\mathrm{J}(\mathbf{q},\mathbf{d}^+,\mathbf{d}^-;\boldsymbol{\theta}) + \mathrm{J}(\mathbf{q}+\boldsymbol{\eta}_q,\mathbf{d}^+ + \boldsymbol{\eta}_{d^+},\mathbf{d}^- + \boldsymbol{\eta}_{d^-};\boldsymbol{\theta}) \Big] \Big] \Big) \tag{13}$$

where a preferred document d^+ is sampled from labeled data and the corresponding non-preferred document d^- is sampled from unlabeled data according to $p_{\boldsymbol{\theta}}$. The adversarial perturbations $\boldsymbol{\eta}_q$, $\boldsymbol{\eta}_{d^+}$, and $\boldsymbol{\eta}_{d^-}$ can be computed in the same way as in equation (4).

The cost function J for pairwise training is defined as

$$\mathrm{J}(\mathbf{q},\mathbf{d}^+,\mathbf{d}^-;\boldsymbol{\theta}) = -\log p(y^+ \rhd y^-|q,d^+,d^-;\boldsymbol{\theta}) \quad \text{where}$$
$$p(y^+ \rhd y^-|q,d^+,d^-;\boldsymbol{\theta}) = \sigma(f_{\boldsymbol{\theta}}(\mathbf{q},\mathbf{d}^+) - f_{\boldsymbol{\theta}}(\mathbf{q},\mathbf{d}^-)) \tag{14}$$

where $y^+ \rhd y^-$ means d^+ is preferred over d^-.

As in our solution, the pairwise perturbation is proposed for pairwise training as it assumes relative relevance between document pairs. However, virtual adversarial perturbation assumes no labels, so pointwise learning makes more sense than pairwise learning. Hence, we employ a hybrid learning whose objective is defined as

$$\sum_q \Big(\mathbb{E}_{d^+\sim p_{\text{data}}(d|q,y=1)} \Big[\mathbb{E}_{d^-\sim p_{\boldsymbol{\theta}}(d|q,y=1)} \Big[\mathrm{J}(\mathbf{q},\mathbf{d}^+,\mathbf{d}^-;\boldsymbol{\theta}) + \mathrm{J}_{\mathrm{KL}}(\mathbf{q},\mathbf{d}^+;\boldsymbol{\theta}) + \mathrm{J}_{\mathrm{KL}}(\mathbf{q},\mathbf{d}^-;\boldsymbol{\theta}) \Big] \Big] \Big) \tag{15}$$

where we sample one negative example from unlabeled data by $p_{\boldsymbol{\theta}}$ for each positive example in the labeled data. Here, we still learn from the pairwise loss with equation (14) while we also learn from pointwise virtual adversarial loss (equation (6)) for each of positive and negative examples.

5 APPLICATION TO AD-HOC RETRIEVAL TASKS

We apply the proposed approaches to ad-hoc retrieval tasks. Among various ad-hoc retrieval tasks, we chose three tasks: Web search, item recommendation, and question answering. These three tasks are specifically chosen because they are popular and actively studied, and they were also chosen by IRGAN [36], to which we mainly compare our approaches.

IRGAN is chosen as the main baseline because (i) it is based on a type of adversarial training too and it applies its unique negative sampling, (ii) it recently attracted a great deal of attention at ACM SIGIR 2017[2] with its novel approach and outstanding performance, and (iii) the source code and the pre-processed data sets are published[3] so that it provides an excellent benchmark environment. For a fair comparison, we apply our approaches to the same models as IRGAN does, except for a small change in Web search model. As the tasks and models are already described well in [36], we briefly describe them in this section.

5.1 Web Search

For Web search task, input vectors for (q,d) can be formed as a length-k vector $\mathbf{x}_{q,d} \in \mathbb{R}^k$, where each dimension represents a feature value of q and/or d. For example, TF-IDF scores of each document part and a query or PageRank scores of d can be such features. As in RankNet [2] and IRGAN[36], we adopt a two-layer neural network as the model of Web search:

$$f_{\boldsymbol{\theta}}(\mathbf{x}_{q,d}) = \mathbf{w}_2^{\mathsf{T}} f_1(\mathbf{W}_1\mathbf{x}_{q,d} + \mathbf{b}_1) + b_2 \tag{16}$$

where $\mathbf{W}_1 \in \mathbb{R}^{l\times k}$ is a weight matrix and l is the number of nodes in the hidden layer, $\mathbf{b}_1$ and b_2 are a bias vector and a constant, respectively, and $\mathbf{w}_2 \in \mathbb{R}^l$ is a weight vector for the output layer. For the activation function in f_1, the hyperbolic tangent was employed

[2] http://sigir.org/sigir2017/program/awards/

[3] Available at https://github.com/geek-ai/irgan except Netflix data for item recommendation.

in RankNet and IRGAN, but we replace it with ReLU since it can learn more effectively [7]. Although ReLU is quite linear so that it is more vulnerable to adversarial examples, adversarial training makes the model robust to them, so it is not a concern. We also experiment with ReLU version of IRGAN and compare the results. Our approach is originally defined for two separate input vectors $\mathbf{q}, \mathbf{d}$ of $f_{\boldsymbol{\theta}}$, but accommodating a single input vector instead is straightforward.

5.2 Item Recommendation

Item recommendation is a task where a ranked list of items are recommended for a user. Thus, it can be regarded as an ad-hoc retrieval task, where a query is a user, and a document is an item. One simple but popular approach is matrix factorization [18] for collaborative filtering. Given one-hot encoded vectors $\mathbf{u}$ and $\mathbf{i}$ for a user u and an item i, respectively, the model is defined as

$$f_{\boldsymbol{\theta}}(\mathbf{u}, \mathbf{i}) = \mathbf{v}_u^\intercal \mathbf{v}_i + b_i \tag{17}$$

where $\mathbf{v}_u, \mathbf{v}_i \in \mathbb{R}^k$ are latent vectors for u and i, respectively, and b_i is a bias for i. This model works in a linear way so that it is vulnerable to adversarial perturbation, so adversarial training is desired.

5.3 Question Answering

A retrieval-based question answering is a task where a ranked list of answers are retrieved for a question. We employ a deep learning-based end-to-end approach to estimate latent vectors $\mathbf{v}_q, \mathbf{v}_d \in \mathbb{R}^k$. Given one-hot encoded vectors $\mathbf{q}, \mathbf{d}$ for a question q and an answer d, respectively, the relevance of d to q is modeled by a cosine similarity between them:

$$f_{\boldsymbol{\theta}}(\mathbf{q}, \mathbf{d}) = \frac{\mathbf{v}_q^\intercal \mathbf{v}_d}{||\mathbf{v}_q||\,||\mathbf{v}_d||}. \tag{18}$$

Recent successful approaches adopt convolutional neural network (CNN) [30, 31] or long short-term memory neural network (LSTM) [35] to obtain the latent vectors. Although such models can achieve great performances, they are vulnerable to adversarial perturbation as they can behave in a linear way. We thus apply our adversarial training to this model.

6 EXPERIMENTS ON THREE AD-HOC RETRIEVAL TASKS

We perform experiments with our pairwise approaches as they often perform better than pointwise approaches. The main approach uses the objective in equation (13), which is based on pairwise adversarial training, and we call it **AdvIR** (Adversarial training for Information Retrieval). We also experiment with selective virtual adversarial training (**AdvIR-SVAT**) in equation (15) and virtual adversarial training (**AdvIR-VAT**), which iterates over all unlabeled examples. The code is available online.[4]

Due to the complexity of experiments, we borrow some baseline results from [36]. In fact, the exact pre-processed data set is published by [36], so we regard it as a concrete benchmark data set, and we do not repeat experiments for some baselines that are clearly inferior to IRGAN at least on those data sets. Indeed, IRGAN is far superior to the baselines for Web search task. The strongest competitors for other tasks, which are LambdaFM [40] for item recommendation and LambdaCNN [42] for question answering, were actually proposed by some authors of IRGAN, so we can trust their results in [36]. For other settings, we follow the settings in [36].

6.1 Web Search

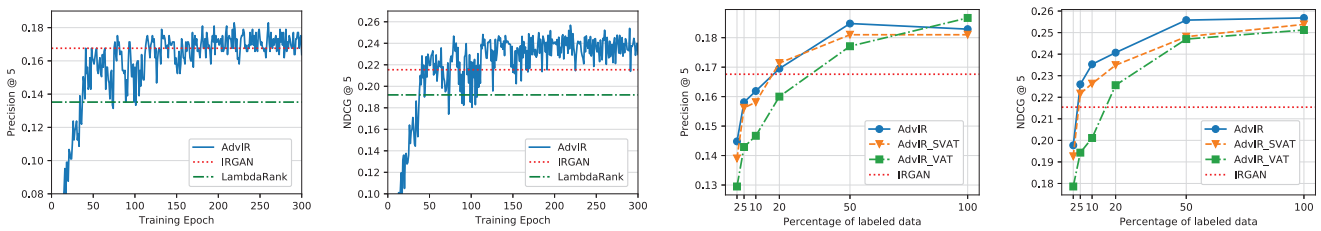
6.1.1 Experimental Design. The well-known benchmark data set LETOR 4.0 [25] provides MQ2008-semi (Million Query track), which is designed for semi-supervised learning. That is, it consists of a relatively small number of labeled data and a much larger number of unlabelled data. Specifically, a relevance in 4 levels (-1, 0, 1, or 2) is given to each query-document pair, where -1 means unlabeled and a greater number means more relevance. As we assume a single-class labeled data, we regard query-document pairs with relevance level 1 and 2 as 'relevant' and compile labeled data, and compile unlabeled data with all other pairs. As a result, there are 784 unique queries in total, and for each query, there are 5 labeled examples and 1,000 unlabeled examples on average. For each query-document pair, there is a 46-dimension feature vector, which consists of continuous features such as TF-IDF and language model values. The vector is given to the two-layer neural network in equation (16). The number of nodes in the hidden layer equals to the dimension size of the feature vector. We set the perturbation size to 300 unless otherwise specified.

Popular learning to rank approaches including RankNet [2], LambdaRank [4], and LambdaMART [3] as well as IRGAN [36] were employed as baselines. Note that a generator of IRGAN plays a role of negative example sampler. As our neural network employs ReLU [7] for an activation function instead of hyperbolic tangent, we also train IRGAN with ReLU on both generator and discriminator, and we report their best measures. We measure statistical significance of the improvement over this model with a paired t-test. We adopt precision at N and Normalized Discounted Cumulative Gain (NDCG) at N as performance metrics since they are standard in ad-hoc retrieval.

6.1.2 Result Analysis. The overall results are shown in Table 1. As described in [36], traditional learning to rank methods such as LambdaMART do not perform well since it is not specifically effective for semi-supervised learning. IRGAN_ReLU does not seem to differ much from IRGAN in terms of performance, but it performs better than IRGAN in NDCG measures. Our proposed approaches significantly outperform baselines in several measures. The improvement from our approaches except AdvIR_VAT is especially good for high-ranked documents. This is expected because our approaches train models with the generated very difficult examples, so they are especially beneficial for distinguishing top documents, who are more difficult to rank in general.

Among the proposed methods, AdvIR outperforms the other proposed methods in general. AdvIR_SVAT outperforms AdvIR_VAT especially for high-ranked documents, which can be identified by P@3 and N@3. Similar to the previous analysis, this is reasonable because it focuses on difficult unlabeled data so that it can affect more in the high-ranked documents. On the other hand, AdvIR_VAT learns from all unlabeled data, and the too easy examples may affect the model in a negative way.

[4] https://sites.google.com/site/daehpark/Resources



(a) (b) (c) (d)

Figure 1: (a,b) Learning curves and (c,d) data efficiency on Web search

Table 1: Overall results on Web search. The best value for each metric is bold-faced. $*$ indicates statistical significance over IRGAN_ReLU. Relative improvement over IRGAN_ReLU is shown in parentheses.

	Prec@3	**Prec@5**	**Prec@10**
RankNet [2]	0.1619	0.1219	0.1010
LambdaRank [4]	0.1651	0.1352	0.1076
LambdaMART [3]	0.1368	0.1026	0.0846
IRGAN [36]	0.2000	0.1676	0.1248
IRGAN_ReLU	0.1937	0.1581	0.1286
AdvIR	**0.2349*** (21.3%)	0.1829* (15.7%)	**0.1305** (1.5%)
AdvIR_VAT	0.2000 (3.3%)	**0.1867*** (18.1%)	0.1248 (-3.0%)
AdvIR_SVAT	0.2222 (14.7%)	0.1810* (14.5%)	0.1238 (-3.7%)
	NDCG@3	**NDCG@5**	**NDCG@10**
RankNet [2]	0.1801	0.1709	0.1943
LambdaRank [4]	0.1926	0.1920	0.2093
LambdaMART [3]	0.1573	0.1456	0.1627
IRGAN [36]	0.2148	0.2154	0.2380
IRGAN_ReLU	0.2230	0.2185	0.2473
AdvIR	**0.2682*** (20.3%)	**0.2568*** (17.5%)	**0.2696*** (9.0%)
AdvIR_VAT	0.2390 (7.2%)	0.2512* (15.0%)	0.2527 (2.2%)
AdvIR_SVAT	0.2598* (16.9%)	0.2538* (16.2%)	0.2593* (4.9%)

The learning curves of AdvIR are depicted in Figure 1a and 1b. The dotted horizontal lines indicate the baseline methods' best measures among all training epochs. We can see that from the early part of the training (epoch=50), it already outperforms the baselines on test data. As more training is done, it easily outperforms the baselines.

In order to see how efficient the models are in terms of data, we randomly removed labeled data and recorded their performance in Figure 1c and 1d. Surprisingly, two of our models, AdvIR and AdvIR_SVAT can still outperform IRGAN in NDCG@5, using only 5% of labeled data. In Precision@5, they outperform IRGAN with only 20% of labeled data. Also, it is shown that AdvIR_SVAT consistently outperforms AdvIR_VAT especially when the number of labeled data is less. Our conjecture is that as there are fewer labeled data, it is more important to focus on difficult unlabeled data to place the relevant documents in the top.

The effect of negative sampling methods on our proposed approach is depicted in Figure 2. Even with uniform sampling for negative examples, AdvIR outperforms IRGAN, which has its own negative example sampling technique. Our approach "generates"

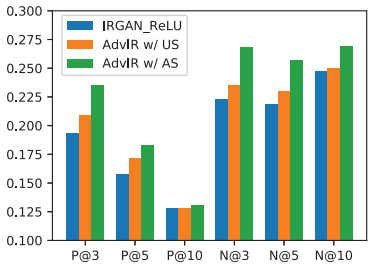


Figure 2: Effect of negative sampling methods for Web search. US stands for uniform sampling, and AS stands for adversarial sampling.

difficult negative examples instead of only "sampling" them, which is done by IRGAN. Through the dynamically generated difficult examples, AdvIR can learn from diverse difficult examples. In addition, IRGAN uses positive examples as they are whereas AdvIR generates difficult examples for even positive examples. These differences can explain the superiority of AdvIR to IRGAN even when uniform sampling is used for AdvIR. Switching from uniform sampling to adversarial sampling, AdvIR performs even better. This is reasonable because the difficult negative examples sampled by adversarial sampling serve as great basis for generating even more difficult negative examples by adversarial training.

6.2 Item Recommendation

6.2.1 Experimental Design. To perform experiments for the item recommendation task, we apply our approaches on a popular benchmark data set, Movielens (100k) [12]. It consists of 943 users, 1,683 items, and 100,000 user-item ratings, where ratings are in 5 levels. Following IRGAN [36], we regard the 4 and 5-star ratings as single-class labeled data[5] and all other entries as unlabeled data. We employ exactly the same data set as in [36] where a 4:1 random splitting is done for training/test data.

The input vectors, which are one-hot encoded vectors, for users and items thus have length 943 and 1,683, respectively. The input vectors are given to the matrix factorization model in equation (17). The size of latent vectors is 5. We set the perturbation size to 0.01 unless otherwise specified.

[5]It is described in [36] that only 5-star ratings are regarded as labeled data, but their published implementation regards both 4-star and 5-star ratings as labeled data.

Popular and state-of-the-art baselines are employed, including Bayesian Personalised Ranking (BPR) [29], LambdaRank-based collaborative filtering (LambdaFM) [40], and IRGAN [36]. Note that BPR samples negative items from uniform distribution while LambdaFM and IRGAN dynamically sample difficult negative items. To measure the statistical significance of our approach's improvement over IRGAN, we re-run IRGAN with their implementation and use the results to perform a paired t-test. Standard performance metrics such as precision at N and NDCG at N are employed.

Table 2: Overall results on item recommendation. $*$ indicates statistical significance over IRGAN_rerun.

	Prec@3	**Prec@5**	**Prec@10**
BPR [29]	0.3289	0.3044	0.2656
LambdaFM [40]	0.3845	0.3474	0.2967
IRGAN [36]	0.4072	0.3750	0.3140
IRGAN_rerun	0.3999	0.3759	0.3217
AdvIR	0.4393* (9.9%)	0.4070* (8.3%)	0.3450* (7.2%)
AdvIR_VAT	0.4313* (7.9%)	**0.4083*** (8.6%)	0.3467* (7.8%)
AdvIR_SVAT	**0.4466*** (11.7%)	0.4066* (8.2%)	**0.3485*** (8.3%)
	NDCG@3	**NDCG@5**	**NDCG@10**
BPR [29]	0.3410	0.3245	0.3076
LambdaFM [40]	0.3986	0.3749	0.3518
IRGAN [36]	0.4222	0.4009	0.3723
IRGAN_rerun	0.4166	0.4010	0.3779
AdvIR	0.4563* (9.5%)	0.4353* (8.6%)	0.4079* (7.9%)
AdvIR_VAT	0.4539* (8.9%)	0.4382* (9.3%)	0.4108* (8.7%)
AdvIR_SVAT	**0.4641*** (11.4%)	**0.4383*** (9.3%)	**0.4183*** (10.7%)

6.2.2 Result Analysis. The overall results are shown in Table 2. The results indicate the dynamic negative item sampling approaches (LambdaFM, IRGAN, and AdvIR) outperform BPR, showing that negative item sampling is indeed important. Our proposed approaches significantly outperform baselines in all measures. Similar to the Web search task, our proposed approaches perform especially well for high-ranked documents in general; the very difficult examples generated by them are supposed to be beneficial for ranking top documents. Among our approaches, AdvIR_SVAT performs the best for this task, but the difference is small. Again, AdvIR and AdvIR_SVAT performs especially better for high-ranked items (P@3 and N@3) than AdvIR_VAT, with its focus on more difficult examples.

The learning curves of AdvIR_SVAT are depicted in Figure 3a and 3b. At around only 100 epochs, it starts to outperform the best measures of baselines. As more training is done, it shows better performance on the test data, and it gains little after 300 epochs. Different from Web search, the curves do not oscillate much for item recommendation.

The data efficiency is depicted in Figure 3c and 3d. Using only 50% of the labeled data, our proposed models outperform IRGAN in both Precision@5 and NDCG@5. Although our proposed approaches show significant improvements, its data efficiency for the item recommendation task is not as good as that for the Web search task. It is reasonable because it is more difficult to perform well for collaborative filtering when the number of labeled data is less; the information in transactions (user-item ratings) directly shrinks so that the models suffer from the cold-start problem. In other words, Web search can take advantage of text in the document when matching a document with a query, but a user does not have such information for collaborative filtering but has only user-item ratings; hence, it is much more difficult to perform well for item recommendation if there are fewer labeled data (user-item ratings). Similar to Web search results, it is shown that AdvIR_SVAT consistently outperforms AdvIR_VAT especially for fewer labeled data, with its focus on difficult unlabeled examples.

The effect of negative sampling methods is depicted in Figure 4. Similar to Web search results, AdvIR with uniform negative sampling still outperforms IRGAN, which has its own dynamic negative sampling mechanism. IRGAN only samples negative examples while AdvIR generates them on top of sampled negative examples. AdvIR also generates difficult examples for positive examples while IRGAN does not. These may have caused the superiority of AdvIR to IRGAN. Adversarial sampling again seems to play an important role in AdvIR for item recommendation. Adversarial sampling helps AdvIR to generate more difficult examples compared to uniform sampling, so AdvIR with adversarial sampling consistently outperforms that with uniform sampling.

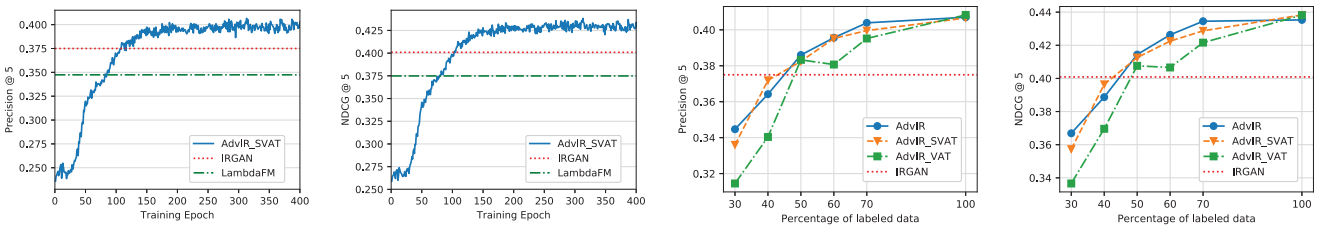
6.3 Question Answering

6.3.1 Experimental Design. For question answering, we apply our approaches to one of the popular benchmark data sets, Insurance QA [6]. It consists of questions, which are submitted by users, and high-quality answers, which are written by domain experts. Training data consist of 12,887 (question, answer) pairs, and development data consist of 1,000 (question, answer) pairs. That is, a single-class labeled data (answer) is given for the task. We can thus regard the whole answer sets as unlabeled data to apply semi-supervised learning. There are two test sets (`test-1` and `test-2`), each of which consists of 1,800 (question, answer) pairs. The task is to retrieve the one and only correct answer from 500 candidate answers. Hence, we report precision @ 1 as the performance metric.

Both questions and answers are in raw text, so we apply the end-to-end model based on convolutional neural networks (CNN). The input vectors, which are one-hot encoded vectors, for questions and answers thus have length that is the vocabulary size, and the embedding vectors have 100 dimensions. We use the same CNN architecture as in IRGAN. The convolutional layer consists of 4 different kernels with sizes 1, 2, 3, and 5, and the feature maps after applying those kernels to the embedding are summarized by the max-pooling-over-time. Then, the resulting vectors for a question and an answer are given to equation (18) as $\mathbf{v}_q$ and $\mathbf{v}_d$, respectively. We set the perturbation size to 0.5 unless otherwise specified.

Strong baselines such as QA-CNN [30], LambdaCNN [30, 42] that enhances QA-CNN with dynamic negative sampling, and IRGAN [36] are employed. Note that uniform negative sampling is done by QA-CNN, and dynamic negative sampling is done in LambdaCNN and IRGAN. To measure the statistical significance of our approaches' improvement over IRGAN, we re-run IRGAN with their implementation and use the results to perform a paired t-test.

6.3.2 Result Analysis. The overall results on question answering are shown in Table 3. Approaches based on dynamic negative sampling techniques, including our approaches, IRGAN, and Lambda



(a) (b) (c) (d)

Figure 3: (a,b) Learning curves and (c,d) data efficiency on item recommendation

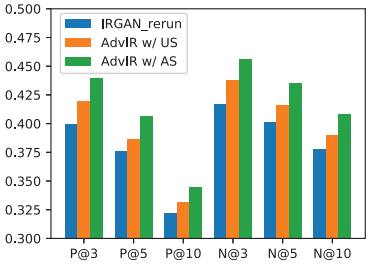


Figure 4: Effect of negative sampling methods for item recommendation. US stands for uniform sampling, and AS stands for adversarial sampling.

Table 3: Overall results on question answering. Precision@1 is reported.

	test-1	test-2
QA-CNN [30]	0.6133	0.5689
LambdaCNN [30, 42]	0.6294	0.6006
IRGAN [36]	0.6444	0.6111
IRGAN_rerun	0.6478	0.6028
AdvIR	0.6489 (0.2%)	**0.6150** (2.0%)
AdvIR_VAT	0.6450 (-0.4%)	0.6083 (0.9%)
AdvIR_SVAT	**0.6517** (0.6%)	0.6133 (1.7%)

CNN, outperform QA-CNN that is based on uniform negative sampling, but the difference in performance is not as big as that of Web search or item recommendation tasks. Although our proposed approaches outperform IRGAN in most measures, the improvements are not statistically significant. The performance difference among our proposed approaches is also small. There can be a couple of reasons for these. As the negative answers in the test data come from answers of other questions, the task is easier than other tasks. Thus, the advantage comes from learning with difficult examples may not be clearly shown in this data set. Indeed, the already high measures in Precision@1 tells the task is relatively easy. Also, there may not be negative examples that are difficult enough due to the data setting. A small improvement by dynamic negative sampling approaches compared to other tasks supports this.

The learning curves of AdvIR are depicted in Figure 5. Although it does not take advantage of pre-trained model as in IRGAN, AdvIR reaches high precision quickly (in 20 epochs). On the other hand, IRGAN, which starts with Precision@1 of 0.6 by pre-trained model, takes >30 epochs to reach Precision@1 >= 0.64 [36].

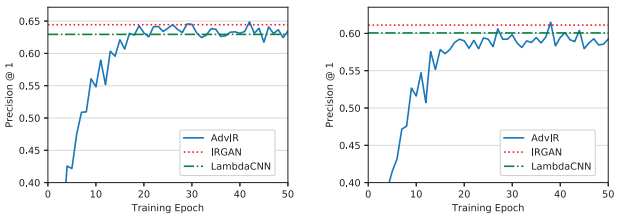


Figure 5: Learning curves on question answering for `test-1` (left) and `test-2` (right).

7 CONCLUSIONS

In this paper, we studied semi-supervised ad-hoc retrieval models with implicit feedback, where there are relatively fewer single-class labeled data and much more unlabeled data. We proposed an adversarial sampling and training framework that handles labeled data and unlabeled data differently. It adversarially generates informational examples for the positive class; on the other hand, it first adversarially samples informational examples for the negative class and further adversarially generates even more informational negative examples. We also proposed virtual adversarial training and selective virtual adversarial training that is more effective and efficient than the former, and they do not require labels to generate adversarial examples.

Experiments were performed on public benchmark data sets for three popular ad-hoc retrieval tasks such as Web search, item recommendation, and question answering. Experimental results show that (`i`) our proposed approaches are effective on ad-hoc retrieval tasks: they significantly outperform baselines on Web search and item recommendation and are on par with IRGAN on question answering, (`ii`) our proposed approaches perform especially well for high-ranked documents, (`iii`) our proposed approaches are data-efficient; (`iv`) adversarial sampling amplifies the effectiveness of adversarial training, and (`v`) our proposed selective virtual adversarial training is more effective and efficient than virtual adversarial training.

REFERENCES

[1] Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. 2016. A neural click model for web search. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 531–541.

[2] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. ACM, 89–96.

[3] Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning* 11, 23-581 (2010), 81.

[4] Christopher J Burges, Robert Ragno, and Quoc V Le. 2007. Learning to rank with nonsmooth cost functions. In *Advances in neural information processing systems*. 193–200.

[5] Jingtao Ding, Fuli Feng, Xiangnan He, Guanghui Yu, Yong Li, and Depeng Jin. 2018. An improved sampler for bayesian personalized ranking by leveraging view data. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 13–14.

[6] Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. *arXiv preprint arXiv:1508.01585* (2015).

[7] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 315–323.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[10] Mihajlo Grbovic, Nemanja Djuric, Vladan Radosavljevic, Fabrizio Silvestri, and Narayan Bhamidipati. 2015. Context-and content-aware embeddings for query rewriting in sponsored search. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. ACM, 383–392.

[11] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 55–64.

[12] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2016), 19.

[13] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial Personalized Ranking for Recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 355–364.

[14] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 843–852.

[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[16] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2333–2338.

[17] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 154–161.

[18] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[19] Tie-Yan Liu et al. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 3, 3 (2009), 225–331.

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[21] Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725* (2016).

[22] Takeru Miyato, Shin-ichi Maeda, Shin Ishii, and Masanori Koyama. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence* (2018).

[23] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. 2015. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677* (2015).

[24] Dae Hoon Park and Rikio Chiba. 2017. A neural language model for query auto-completion. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1189–1192.

[25] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 datasets. *arXiv preprint arXiv:1306.2597* (2013).

[26] Filip Radlinski and Thorsten Joachims. 2005. Query chains: learning to rank from implicit feedback. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 239–248.

[27] Jinfeng Rao, Hua He, and Jimmy Lin. 2016. Noise-contrastive estimation for answer selection with deep neural networks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1913–1916.

[28] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 273–282.

[29] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[30] Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609* (2016).

[31] Aliaksei Severyn and Alessandro Moschitti. 2016. Modeling relational information in question-answer pairs with convolutional neural networks. *arXiv preprint arXiv:1604.01178* (2016).

[32] Xuehua Shen, Bin Tan, and ChengXiang Zhai. 2005. Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 43–50.

[33] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 553–562.

[34] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[35] Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 707–712.

[36] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 515–524.

[37] David Warde-Farley and Ian Goodfellow. 2016. 11 adversarial perturbations of deep neural networks. *Perturbations, Optimization, and Statistics* (2016), 311.

[38] Kai Xu, Dae Hoon Park, Chang Yi, and Charles Sutton. 2018. Interpreting Deep Classifier by Visual Distillation of Dark Knowledge. *arXiv preprint arXiv:1803.04042* (2018).

[39] Xiao Yang, Miaosen Wang, Wei Wang, Madian Khabsa, and Ahmed Awadallah. 2018. Adversarial Training for Community Question Answer Selection Based on Multi-scale Matching. *arXiv preprint arXiv:1804.08058* (2018).

[40] Fajie Yuan, Guibing Guo, Joemon M Jose, Long Chen, Haitao Yu, and Weinan Zhang. 2016. Lambdafm: learning optimal ranking with factorization machines using lambda surrogates. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 227–236.

[41] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. 2017. Interpretable convolutional neural networks. *arXiv preprint arXiv:1710.00935* 2, 3 (2017), 5.

[42] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 785–788.

Nameles: An intelligent system for Real-Time Filtering of Invalid Ad Traffic

Antonio Pastor*
Universidad Carlos III de Madrid

Matti Pärssinen
Aalto University

Patricia Callejo
IMDEA Networks Institute
Universidad Carlos III de Madrid

Pelayo Vallina
IMDEA Networks Institute
Universidad Carlos III de Madrid

Rubén Cuevas
Universidad Carlos III de Madrid

Ángel Cuevas
Universidad Carlos III de Madrid

Mikko Kotila
Botlab

Arturo Azcorra
IMDEA Networks Institute
Universidad Carlos III de Madrid

ABSTRACT

Invalid ad traffic is an inherent problem of programmatic advertising that has not been properly addressed so far. Traditionally, it has been considered that invalid ad traffic only harms the interests of advertisers, which pay for the cost of invalid ad impressions while other industry stakeholders earn revenue through commissions regardless of the quality of the impression. Our first contribution consists of providing evidence that shows how the Demand Side Platforms (DSPs), one of the most important intermediaries in the programmatic advertising supply chain, may be suffering from economic losses due to invalid ad traffic. Addressing the problem of invalid traffic at DSPs requires a highly scalable solution that can identify invalid traffic in real time at the individual bid request level. The second and main contribution is the design and implementation of a solution for the invalid traffic problem, a system that can be seamlessly integrated into the current programmatic ecosystem by the DSPs. Our system has been released under an open source license, becoming the first auditable solution for invalid ad traffic detection. The intrinsic transparency of our solution along with the good results obtained in industrial trials have led the World Federation of Advertisers to endorse it.

ACM Reference Format:
Antonio Pastor, Matti Pärssinen, Patricia Callejo, Pelayo Vallina, Rubén Cuevas, Ángel Cuevas, Mikko Kotila, and Arturo Azcorra. 2019. Nameles: An intelligent system for Real-Time Filtering of Invalid Ad Traffic. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313601

1 INTRODUCTION

Online advertising is a major social and economic driver of the so-called *Information Society*. First, online advertising sponsors free offerings of essential services to billions of users, such as Online Search Services, Map Services, and Social Media. Second, the market volume of online advertising reached, only in the US, $88 B in 2017 with an inter-annual growth rate of 21% [30]. Third, online advertising represents an important source of jobs. For instance, recent studies have estimated that 1 M direct and 6 M indirect jobs were associated with online advertising in the EU-28 workforce in 2015 [38]. Fourth, online advertising represents the fundamental source of income of the companies at the forefront of technological innovations such as Google or Facebook [2, 13]. Therefore, it is in the best interest of everyone (citizens, governments and the private sector) to guarantee the sustainable growth of this business. However, this sustainability is in jeopardy due to several factors. Arguably, the most important is the high volume of *invalid ad traffic*, i.e., delivered ads not shown to humans. It is estimated that every year trillions of delivered ad impressions are not watched by humans leading to losses of tens of billions of dollars for advertisers [18, 28].

Unfortunately, the identification and filtering of invalid ad traffic has not been properly addressed so far due to two fundamental reasons. First, a rapidly increasing fraction of ad transactions occur through a programmatic ecosystem, where a chain of intermediaries automatically connects advertisers willing to show ads and publishers owning the inventory (websites, mobile apps) to show those ads. This automatic process makes the detection of invalid traffic complex. Second, intermediaries in programmatic advertising receive a commission for each delivered ad, regardless if it is invalid or not. Then, it is well accepted the idea that invalid traffic only harms the interests of advertisers, which pay for the cost of the invalid ad impressions. Intermediaries in the supply chain get a commission for each served invalid impression and then they do not have direct monetary incentives to fight invalid traffic effectively.

Specialized companies referred to as verification vendors (e.g., IAS [20], DoubleVerify [11], White Ops [42]) have emerged offering opaque proprietary solutions for the identification of invalid traffic. Previous research has shown that even simple attack vectors can defeat these opaque defenses [8, 26]. These solutions do not properly address the concerns of advertisers, which have become

*Corresponding author: anpastor@it.uc3m.es

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313601

increasingly vocal about the uncertainty of the quality of programmatic media transactions [12, 39, 41] and the lack of transparency in the ecosystem [4, 9].

To meet the demands of advertisers, in this paper, we present the first open source (and thus auditable) solution for the detection of invalid ad traffic in programmatic advertising.

Prior to designing our solution, we have revisited the common idea that advertisers are the only stakeholders affected by invalid ad traffic in programmatic advertising. We present an economic model based on real financial reports of Demand Side Platforms (DSPs)–a key intermediary in the programmatic advertising ecosystem–and realistic assumptions on the operational set-up of DSPs, which provides initial evidence that invalid ad traffic seems to impact the business model of DSPs negatively. This finding suggests, contrary to the conventional wisdom, that DSPs may have strong incentives to filter invalid ad traffic. Our analysis concludes that post-bid (i.e., non real-time) detection of invalid traffic does not solve the problem for the DSPs. Instead, DSPs require a solution that can identify invalid traffic in real-time and at the level of individual ad transactions. Moreover, DSPs handle up to tens of billions of ad transactions per day, a factor imposing demanding computational performance constraints to the invalid traffic detection problem.

The main contribution of this paper is *Nameles*, an open source invalid ad traffic detection system that operates in real-time at the level of individual requests. Therefore, it meets the requirements of both advertisers and DSPs. Nameles identifies anomalous ad requests patterns of domains using an algorithm based on Shannon entropy. Nameles has been built in accord with the latest version of the OpenRTB specification [23] and is able to handle up to 500 k ad transactions per second, adding a total delay of less than 3 ms to each ad transaction. As a result, it can be seamlessly integrated into the programmatic supply-chain as a solution for the DSPs. We have applied Nameles on a stream of ~1.8 B daily bid request in two periods of two months separated one year from each other, observing that (on average) 20% of the daily ad traffic can be safely considered invalid.

Nameles' code is publicly available (under an open source license) and then auditable by anyone. This intrinsic transparency, along with the good detection performance shown by the system in extensive industry trials, has led the main global advertiser trade-body, the World Federation of Advertisers (WFA), to endorse Nameles.

2 BACKGROUND

In this section, we describe the process of serving an ad in programmatic advertising. We do it with an example of a user visiting a website, but the process is the same for ads shown in mobile apps.

A user connects to a website offering some ad spaces. Each ad space is typically leased by the website's publisher to an ad network or a Supply Side Platform (SSP), which upon the user's connection generates an ad request. This ad request may be forwarded through several intermediaries until it reaches the Ad Exchange. This part of the process is known as the *sell-side* and is represented by Step 1 in Figure 1. The ad request includes the domain name, IP address of the device, the User Agent, user's cookie(s), etc. Upon the reception of the ad request, the Ad Exchange initiates an auction which represents the *buy-side* of the programmatic process.

The bidding process of this auction is standardized by the OpenRTB protocol [23]. First, the Ad Exchange processes the information included in the ad request to generate a bid request whose format is specified by the OpenRTB standard [23]. For simplicity, in this paper, we will consider that a bid request includes the IP address receiving the ad and the domain name selling the ad space. Each bid request is sent to the Demand Side Platforms (DSPs) registered in the Ad Exchange. A DSP is an intermediary where advertisers, or their agencies, configure their programmatic advertising campaigns. Therefore, upon the reception of a bid request a DSP checks if the request meets the configuration parameters of any of its advertising campaigns and if so, it creates a bid response including, among other information, the price the advertiser is willing to pay for this ad space. Note that the bid responses to a given bid request have to be received by the Ad Exchange within 120 ms [15]. The Ad Exchange runs an auction based on the received bid responses and informs all the participant DSPs about the selected winner bid. The bidding process is represented by Steps 2-4 in Figure 1. To finalize the programmatic process the Ad Exchange coordinates the delivery of an URL from where to retrieve the ad, which is immediately downloaded by the browser and shown to the user. This is represented by step 5 in Figure 1.

From a business perspective, each bid event corresponds with an opportunity to place an online advertisement on a web page for the advertiser, and an opportunity to monetize an ad placement for the publisher. Based on their respective commission percentages (a reference of them obtained from insights from the industry is presented in Figure 1), the intermediaries are compensated every time a bid is successfully transacted, and an ad is displayed as a result. However, the advertiser only benefits when the traffic associated with the transaction is valid. This business model is open to fraudulent activity [18, 28, 29] (e.g., a publisher monetizing visits to a website coming from bots) whereas it seems not to offer the right incentives to intermediaries to identify and filter invalid ad traffic. Note that, according to various industry guidelines [14, 41], *invalid traffic is defined to correspond with those bid events where displaying an ad would not have any potential for advertising effect and the advertiser would lose its investment without getting anything in return.*

Verification companies (e.g., IAS, Double Verify or White Ops) have emerged recently offering proprietary opaque solutions for filtering invalid traffic. However, the lack of transparency on the used techniques makes it difficult to assess their actual capabilities. Indeed, recent works have demonstrated inefficiencies in these solutions [8, 26] and different studies attribute billions of dollars wasted in invalid traffic every year [18, 28, 29].

3 ECONOMIC IMPACT OF INVALID TRAFFIC IN DSP COMPANIES

In this section, we refute the argument that advertisers are the only stakeholders in the programmatic ecosystem negatively affected by invalid traffic [27]. To this end, we provide qualitative and quantitative economic analyses that support how, under realistic assumptions, invalid traffic negatively impacts the profitability of DSP companies.

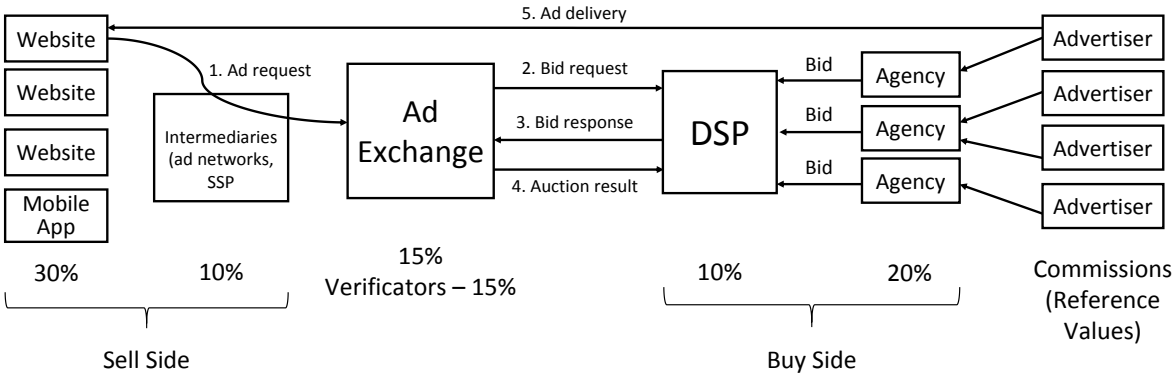


Figure 1: Overview of the programmatic advertising ecosystem operation.

-Qualitative Analysis: We investigated seven publicly listed DSP companies through their annual income statements and found that only one company had a positive net income. Moreover, depending on the DSP company, variable costs ranged from 30% to 50% of the revenue. Variable costs represent a proportion of total costs that vary as a function of revenue. These findings indicate that the low profitability of analyzed DSPs has a strong correlation with variable costs.

The DSP win-rate [9] is defined as the fraction of won bids out of all auctions. Regardless if an auction the DSP is hosting results in a win or not, the DSP bears the cost for facilitating that auction. Then, the inverse of the win-rate indicates how much a DSP company accumulated variable costs that yield no economic revenue. While valid bids represent a real opportunity for advertisers that provides an intangible value even to lost valid bids, in the case of invalid bids there is not a real opportunity and thus, lost invalid bids contribute exclusively to increase DSPs costs.

Based on interviews we conducted with DSP companies, we conclude that the DSP win-rate is typically between 5% to 20%. An individual advertiser win-rate has been shown to be in the range 0.1% to 1% [44], and ad exchanges (and ad networks) *fill-rate* (i.e., the fraction of successfully completed auctions) is commonly below 40% [35]. Consequently, we assert that there is an oversupply of programmatic ad inventory. This supports the economic viability of invalid traffic filtering so that even after removing the invalid ad traffic there will be still enough ad inventory available for DSPs.

In summary, all the above objective facts show that filtering invalid traffic would contribute to reducing the accumulated variable costs of DSPs without affecting the availability of ad inventory and as a result would lead to improving profitability and valuation of a DSP company.

Finally, to maximize the profitability of the DSP company, invalid bid requests should be identified in real time in the pre-bid stage, so that variable costs incurred by processing such invalid bids are minimized since the processing of the bid is stopped in the first step of the procedure.

-Quantitative Analysis: Net Present Value (NPV) model is the tool of choice for financial forecasting because it considers the time value of money, and provides a concrete metric to financial decision-makers, such as investors, for evaluating investment against the predicted return [3]. Finance theory endorses an investment if NPV is positive and higher than NPV of an alternative investment [3]. In addition to the NPV, we evaluated Enterprise Value (EV) [21], a useful variant of the NPV, that takes into account cash flows beyond the forecasted time window. Positive NPV and EV values are reached when the cash inflows exceed cash outflows [3]. NPV and EV are widely used as decision-making tools for planning purchases, mergers or acquisitions [3].

Table 1: Impact of invalid traffic filtering to economics of DSPs.

	Enterprise Value (EV)			Net Present Value (NPV)		
	No filtering	Filtering		No filtering	Filtering	
		Max [F=0,23; P=0]	Min [F=1; P=1]		Max [F=0,23; P=0]	Min [F=1; P=1]
DSP-1	10,544	19,002	-3,269	4,421	8,020	-1,979
DSP-2	2,516	5,194	-3,518	536	1,672	-2,213
DSP-3	3,254	3,833	-1,147	1,237	1,481	-706
DSP-4	1,184	2,973	-2,376	133	892	-1,498
DSP-5	1,702	2,648	-819	634	1,035	-506
DSP-6	1,354	2,342	-1,005	445	863	-628
DSP-7	1,595	2,262	-2,118	310	592	-1,337
Average	3,163	5,468	-2,036	1,101	2,079	-1,267

We compute NPV and EV for two scenarios; without invalid traffic filtering (Scenario A), and with filtering (Scenario B). The timeframe of the analysis is eight years. NPV and EV are computed based on five key factors:

1) Annual growth rates. In our analysis, they are based on the industry average of seven publicly listed DSPs' annual and quarterly income statements between 2012-2015 [16] and are the same for both scenarios.

2) Rate of return r. We use $r = 20\%$, which is a typical value for investments made into new products [3, 22], for both scenarios.

3) Invalid traffic filtering rate F. We consider $F = 0$ for Scenario A and F between 0 and 100% for Scenario B.

4) Revenue penalty P (as a dependent factor of F). We have selected the parameters of the penalty function to make the penalty increase in an exponential manner, such that the penalty is low until $F = 20 - 30\%$ and it spikes after this point until F reaches 100% where all traffic is filtered. This function has been carefully constructed so that a low penalty is imposed for filtering rates up to the average reported fraction of invalid traffic from different studies [10, 43].

5) Long-term cash flow growth rate G, which is set-up to 2% in both scenarios [25].

Our results show that there are NPV and EV gains for the DSP when the filtering rate increases from zero towards $F = 23\%$. Filtering invalid traffic beyond $F > 23\%$ first results in diminishing benefit and eventually drives a decline in revenue for the DSP. These results confirm our hypothesis that filtering invalid traffic (at a reasonable rate) improves DSPs profitability due to a reduction in variable costs. Table 1 shows the minimum and maximum values of NPV and EV, which correspond to [F;P] values of [0,23;0] and [1;1] in Scenario B, respectively. We observe that at the optimal filtering rate, NPV and EV in Scenario B increase (on average) 1,72 and 1,89 times in comparison with Scenario A, respectively.

Finally, we would like to note that the results obtained in this section represent a reference example based on realistic assumptions and should be interpreted as such.

4 DATASET

The dataset used in this paper includes a daily sample of incoming bid requests stream data collected in two different periods one year apart and each spanning two months between Dec 2016-Jan 2017 and Dec 2017-Jan 2018, respectively.

The data is from one of the largest DSPs with a significant global presence. The data consist of desktop and mobile bid events, for video, banner and in-app inventory. In particular, each daily sample includes between 1.7-1.9 Billion actual bid requests issued on that date from ~50 Ad Exchanges. These bid requests are associated (on average) to ~150 M IP addresses and ~900 k domains per day. The dataset includes the following information per bid request: a unique identifier, the IP address[1] and User Agent of the device initiating the ad request and the Web Domain or Mobile Application ID selling the ad space. For simplicity, we refer to both Web Domains and Mobile Apps as *Domains* along the paper.

5 SYSTEM REQUIREMENTS, DESIGN AND IMPLEMENTATION

In this section we describe *Nameles*, an open source system for the detection of invalid ad traffic that operates in real time and at the level of individual bid requests, thus meeting the requirements of DSPs and advertisers.

5.1 System's Functional Requirements

1. Scalability: DSPs typically handle tens of billions of bid requests per day. This maps into peaks of hundreds of thousands of bid request per second. Nameles must be capable of handling these high rates of bid requests.

2. Delay: The bid response to a given bid request has to be received by the Ad Exchange within 120 ms [15]. Hence, the delay introduced by Nameles should be limited to a few milliseconds in order to minimize the impact in the overall bidding process delay.

3. Accuracy in invalid traffic identification: Providing 100% guarantee that a bid request is invalid (or not) is not feasible. Instead, it is more reliable providing a score indicating the likelihood that a bid request is invalid. Therefore, our system must incorporate an accurate scoring algorithm.

[1]We anonymize the IP address before any processing.

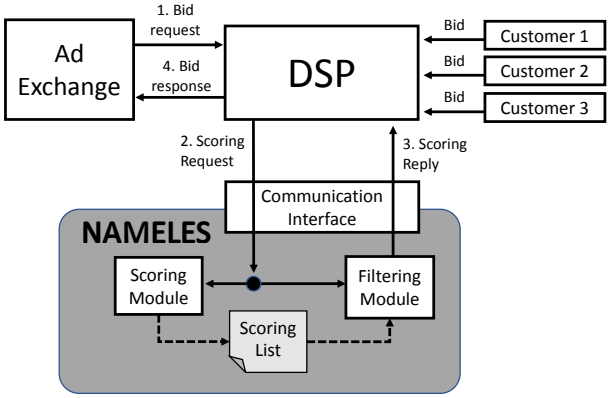


Figure 2: Programmatic Ecosystem Scheme with Nameles.

5.2 System Design

In this subsection, we present a brief overview of the system functionality and how it is integrated with the programmatic advertising ecosystem. Then, we describe in detail each of the functional blocks forming Nameles.

5.2.1 Overview. Figure 2 depicts a high-level representation of Nameles functional blocks. Moreover, the figure shows how Nameles could be integrated into the programmatic ad delivery chain as an auxiliary service for DSPs. The only difference with respect to the current operation of a DSP would be that, as part of the pre-bid phase, the DSP makes a request to Nameles to provide a Confidence Score per bid request. To this end, the DSP sends a *scoring request* to Nameles (step 2 in Figure 2). The scoring request includes the following fields: bid request id (to allow mapping Nameles result to the corresponding bid request), IP address of the device associated with the ad request and the domain offering the ad space. This information is included in the bid requests as defined in the OpenRTB protocol standard [23]. The *scoring request* is delivered to two independent modules of Nameles: the *Scoring* module and the *Filtering* module.

Because the DSP has limited information about a bid request to determine if it is invalid or not, we propose to aggregate all bid requests from a domain and use statistical analysis to determine the level of confidence of a domain. This approach provides statistically robust Confidence Scores for domains since they are computed from a sample of (at least) hundreds of bid requests. Then, Nameles assigns to the bid requests from a domain the Confidence Score of such domain. The *Scoring Module* is responsible for computing the Confidence Score for domains present in the bid requests received by the DSP. Moreover, it groups the domains in four different Confidence Classes. This information is summarized in the *Scoring List*.

The *Filtering* module is responsible for classifying in real-time each received *scoring request*. To this end, it retrieves the domain id from the *scoring request* and obtains the domain's Confidence Score and Confidence Class from the *Scoring List* introduced above.

After that, it creates a *scoring reply* to be sent to the DSP (Step 3 in Figure 2). This reply includes the following information: bid request id (extracted from the corresponding scoring request), the domain Confidence Score, and the domain Confidence Class.

Finally, the *Communication Interface Module* handles the communication between the DSP and Nameles.

5.2.2 Communication Interface Module. This module manages the delivery of scoring requests from the DSP to Nameles and scoring replies in the opposite direction. We have opted to use a parallel pipeline communication structure. In particular, the DSP creates two queues: a sending queue used for pushing scoring requests to Nameles and a receiving queue for pulling scoring replies from Nameles in return. Nameles sets up a number of worker processes, which connect to the sockets associated with both queues. These workers pull scoring requests from the sending queue and forward them to the Scoring and Filtering modules. The resulting scoring replies are pushed by the workers to the receiving queue of the DSP.

This parallel pipeline communication structure is highly scalable, being able to handle streams of hundreds of thousands of requests per second with processing delays below 3 ms (see Section 6.2).

5.2.3 Scoring Module. The goal of the scoring module is to produce a *Scoring List* of domains to be used by the *Filtering* module. This list is updated daily. Since Nameles operates in real-time, the list used at day d is obtained from a prediction algorithm applied on the historical *Confidence Score* values of domains at days $d-1$, $d-2$, $d-3$, ... This module implements three different algorithms to produce the *Scoring List*: one to compute the Confidence Score of each domain, a second to compute the Confidence Classes, and a third to derive the *Scoring list* to be used at day d based on historical information. Next, we describe each of these algorithms.

- Confidence Score computation: A DSP can reconstruct the traffic pattern associated with a given domain X by analyzing the distribution of a number of requests across the IP addresses included in the bid requests associated to X. This is the fundamental signal used by our algorithm. Skewed distributions, where most bid requests come from just a few IP addresses, are for obvious reasons suspicious[2] and thus domains presenting such traffic patterns should be assigned low Confidence Scores. Instead, legit traffic patterns correspond to more homogeneous distributions of bid requests across IPs and domains presenting such distributions should receive high Confidence Scores.

We compute the Shannon Entropy [33] of the distribution of bid requests across IP addresses for each domain. The Shannon Entropy summarizes in a single value the level of determinism of a distribution and ranges between 0 (all bid requests to a domain come from a single IP address) and $log_2(n)$ (the bid requests are homogeneously distributed across the n IP addresses making ad requests to the domain).

Shannon entropy has been successfully used in the field of anomaly detection [24, 40]. However, in our case, it has an important limitation because it does not consider the volume of bid requests, but just the shape of the distribution of bid requests. This limitation prevents the direct comparison of domains with different traffic volumes. For instance, a domain with 5 bid requests uniformly distributed across 5 IPs would have the same Entropy value (2.32) than a domain with 5000 bid requests homogeneously distributed across 5 IPs. While the first domain is just an unpopular domain, the second one is highly suspicious, having a high number of daily visits distributed evenly across a small number of IPs.

[2]For instance, a domain receiving most of its visits from scrapers or from other types of bots associated with fraudulent ad traffic.

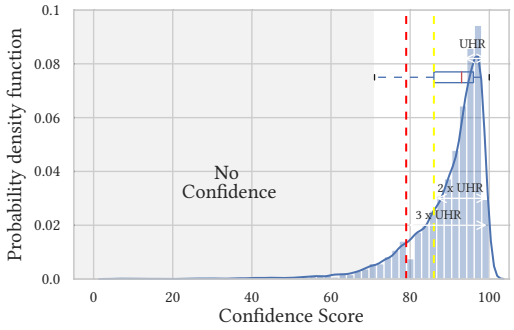


Figure 3: Distribution of Confidence Score (CS) values for domains with more than 500 bid requests at January 15, 2018.

To address this limitation, we propose a normalization process that takes into account the volume of bid requests associated with a domain X. In essence, we compute the ratio of the Shannon entropy and the binary logarithm of the total number of bid requests ($C(X)$) and scale the resulting value to a normalized range between 0 and 100. This normalized entropy score is the *Confidence Score* (CS) assigned to domains by Nameles, and its formal expression is:

$$CS(X) = 100\left(1 - \frac{\sum_{i=1}^{n} C(x_i)\, log_2(C(x_i))}{C(X)\, log_2(C(X))}\right) \quad (1)$$

To get an intuition on the effect of this normalization process, we can consider the toy example mentioned above. The domain with 5 bid requests from 5 IP addresses would have a high CS equal to 100 whereas the domain with 5000 bid requests would have a low CS equal to 19.

- Computation of the Confidence Classes: We first analyzed the probability distribution function of the CS values across domains in our daily datasets. Figure 3 shows this distribution for a specific day. Note that other days in our dataset showed similar distributions. We observed a skewed distribution concentrated in the high CS values with a long tail towards low CS values. This distribution indicates that most domains present homogeneous traffic patterns (represented by high CS) while as we move towards lower values of CS, we find domains with infrequent (i.e., statistically unlikely) traffic patterns offering lower confidence.

To define the Confidence Classes, we use two different unsupervised statistical methods that divide the distribution in 4 ranges each representing a single Confidence Class:

- *Outlier detection method*: This method identifies outlier CS values based on the definition of traditional outliers [31], i.e., $CS(X) < 25\ percentile - 1.5 \times IQR$. Nameles uses this expression to define the threshold for the *No Confidence* Class including domains with an extremely deterministic and infrequent traffic pattern.

- *Dispersion method*: We defined intermediate Confidence Classes between the one formed by outliers and the one composed by the mass of legit domains. To this end, we use the Upper Half Range[3] (UHR) of the distribution as our dispersion metric and define two new thresholds as $max\,(CS) - 2 \times UHR$ and $max\,(CS) - 3 \times UHR$. Based on these thresholds we defined the following Confidence Classes:

  - *Low Confidence Class*: formed by domains whose CS falls in the range $max\,(CS) - 3\,UHR > CS \geq 25\,percentile - 1.5\,IQR$.
  - *Moderate Confidence Class*: formed by domains whose CS falls in the range $max\,(CS) - 2\,UHR > CS \geq max\,(CS) - 3\,UHR$.
  - *High Confidence Class*: formed by domains whose CS falls in the range $CS \geq max\,(CS) - 2\,UHR$.

Figure 3 shows the defined Confidence Classes for the Confidence Score distribution of the dataset of Jan 15, 2018.

- Predicting the Scoring List: The Scoring list used at day d has to be inferred from a prediction algorithm applied to the historical Confidence Score values of domains at days $d-1$, $d-2$, $d-3$, ...We refer to the estimated CS value of a domain X included in this list as $CS^*_d\,(X)$. To define the prediction algorithm, we first studied the stationary properties of the temporal series of CS values of domains across the 62 days forming each of our datasets. This analysis revealed that CS values present high stationarity, with 40% of the domains in our dataset being strictly stationary (with a 90% confidence interval), as reported by the Augmented Dickey-Fuller test [32]. The analysis of the autocorrelation and partial autocorrelation functions for these domains revealed that in general, only the CS of the previous day ($CS_{d-1}\,(X)$) contributes significantly to the prediction of $CS\,(X)$ at day d. Then, the best predictor is $CS^*_d\,(X) = CS_{d-1}\,(X)$ and the Scoring List to be used at day d is formed by the $CS^*_d\,(X)$ of the different domains in our dataset.

As a result of the application of the three described algorithms, the Scoring Module produces each day a Scoring List that includes both the Confidence Score and the Confidence Class for each individual domain.

5.2.4 Filtering Module. This module processes in real-time each scoring request received from the Communication Interface module. In particular, it extracts the domain from the scoring request and searches for the $CS^*_d\,(X)$ and the Confidence Class associated with the domain in the Scoring list. As a result of this process, the Filtering Module generates a *scoring reply* message including the following information: Bid Request ID (obtained from the corresponding scoring request) and the domain's CS and Confidence Class. The scoring reply is sent to the DSP through the Communication Interface module. Note that if the domain extracted from the scoring request is not present in the Scoring list, the scoring reply has the following content <bid request id, NULL, NULL>.

5.3 System Implementation

In this subsection, we describe our implementation of Nameles that meets the performance and scalability requirements defined in Subsection 5.1.

[3]The UHR is measured as the distance between the median and the maximum value of the CS distribution.

5.3.1 The Communication Interface and Filtering module. The Communication Interface and the Filtering modules address different functional aspects of Nameles, and thus we have described them separately in Section 5.2. In our Nameles prototype, we use an integrated implementation of these two functional modules for efficiency purposes. We implement the parallel pipeline communication structure described in Section 5.2.2 on top of ZeroMQ [19] (a highly scalable distributed messaging system) using the existing Java bindings for this purpose. On the Nameles side, we use 6 workers that in addition to taking care of the pull and push communication functions, implement the filtering process. Each worker is an independent process, which has an independent copy of the Scoring List hash table produced by the Scoring Module allocated in RAM. Moreover, each worker pulls independently scoring requests from the DSP's sending queue. For each scoring request, it extracts the domain, obtains the CS and Confidence Class associated with the domain from the Scoring List hash table, creates the scoring reply and pushes it to the DSP's receiving queue.

5.3.2 The Scoring Module. The Scoring Module implements a temporary hash table including the number of bid requests associated with each pair <domain, IP>. For each new bid request, the counter of the tuple <domain, IP> included in the bid request is increased by 1. At the end of every day, the resulting hash table includes the needed information to compute the Confidence Score for each domain as well as the thresholds to define the different Confidence Classes. For this purpose, we store this temporary table into a PostgreSQL database and use different PostgreSQL functions and Java scripts to obtain the CS and the Confidence Class of each domain. The final result of the process is the Scoring List, which is stored in a hash table using as a key the domain id and as value the tuple <CS, Confidence Class>. This table is transferred to the "Communication Interface and Filtering" module to be used in the real-time filtering of bid requests.

6 PERFORMANCE EVALUATION OF THE SYSTEM

We have deployed a realistic experimental set-up to confirm that our Nameles prototype meets the requirements defined in Section 5.1.

6.1 Experimental Set-up

The experimental set-up replicates a production set-up used by a large-scale DSP. In particular, we use three servers in our setup for Nameles. The first server plays the role of the DSP. This server uses the real stream of bid requests from our dataset to produce a stream of scoring requests to Nameles. The rate of scoring requests is a configurable parameter so that we can perform stress tests by using significantly higher rates of bids per second than the ones reflected in our dataset. The second server deploys the "Communication Interface and Filtering" module of our Nameles prototype. It receives the stream of scoring requests from the DSP server and processes it to obtain the scoring replies. In addition, this server forwards the scoring requests to a third server, which implements the "Scoring" module.

The server emulating the DSP is a Dell PowerEdge R710 with 16 cores and 48 GB of RAM. The servers implementing the "Communication and Filtering" and the "Scoring" modules are similar,

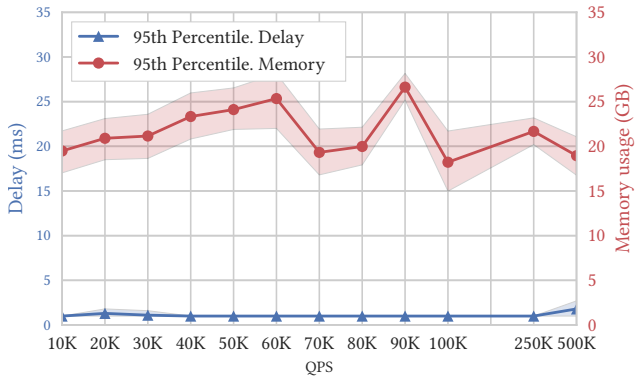


Figure 4: 95 percentile of delay and memory consumption for the filtering process at different input request rates.

a Dell PowerEdge R730xd with 24 cores and 64 GB of RAM. The servers are connected to a conventional 1 Gbps Ethernet switch. In the context of common use in the Adtech industry, the resources employed in our prototype can be considered commodity hardware.

6.2 Scalability and Processing Delay

6.2.1 Scoring List computation time. A critical aspect of the scalability of Nameles resides in its ability to produce the Scoring List in a short time. Specifically, given that the Scoring List is updated daily, the computation process must guarantee that the new list is ready before the expiration of the previous one, i.e., in less than 24 h. We have measured the computation time for every daily dataset, including between 1.7-1.9 B bid requests, and confirmed that it is always shorter than 4 hours. Hence, Nameles meets the scalability requirements for this critical process.

6.2.2 Delay and memory consumption of the filtering process. From the DSP's perspective, the filtering process starts when it sends a Scoring Request and finishes when it receives the corresponding Scoring Reply. The analysis of our dataset reveals an average and a peak rate of 22 k and 26 k requests per second, respectively. Then, our prototype must meet the next two requirements while processing scoring requests streams at the observed peak rate: not overflowing the memory of the server and offering a small delay to minimize its impact on the overall delay of the real-time bidding process.

We have evaluated the performance of our prototype for scoring request streams ranging from 10 k to 500 k queries per second (QPS). For each of the analyzed rates, we run stress tests of 5 minutes. During the tests, we measure the individual delay associated with the filtering process of each scoring request as well as the overall memory consumption of the filtering process. Figure 4 summarizes the performance of our Nameles prototype. The x-axis shows the different tested scoring request rates. The left y-axis and right y-axis show the 95-percentile filtering delay and 95-percentile memory consumption measured during the experiment for the different scoring request rates (QPS), respectively. Note that each stress test has been run 5 times. The line in the figure represents the average of 95-percentile values across the five experiments whereas the lighter color area shows the max and min 95-percentile values.

First of all, we observe that the system performance is quite stable across the different experiments and the observed variability in memory consumption is due to the instantaneous load of the server at the measurement moment rather than the QPS of the experiment. The results of the stress tests demonstrate that our Nameles prototype offers very high scalability performance. In particular, the 95 percentile of memory consumption and delay is lower than 28 GB and 3 ms for any of the considered QPS. These results prove that our filtering process scales to handle more than 20 B bid requests per day with a modest infrastructure, meeting the requirements of the largest DSPs such as Google, The Trading Desk and MediaMath.

6.3 Scoring Accuracy

In this subsection, we assess both the accuracy of our prediction algorithm and the accuracy of the Confidence Scores assigned to domains.

6.3.1 Accuracy of the prediction algorithm. For each daily dataset, we have computed the Root Mean Square Error (RMSE) of the difference between the predicted CS ($CS^*_d(X)$) and the actual CS ($CS_d(X)$) across all domains. The results indicate that the RMSE is smaller than 3 points in every case.

In addition, we have evaluated the misclassification rate of domains among Confidence Classes. We observe that the average misclassification rates across all days in our dataset are below 3% between any pair of classes. A careful analysis of the misclassified domains indicates that the classification errors are mainly associated with domains having a CS close to the threshold that separates two contiguous classes. This is also coherent with the fact that misclassifications between non-contiguous classes are negligible (< 0.3%).

6.3.2 Assessment of Confidence Score accuracy. The accuracy of the Confidence Score cannot be objectively evaluated. There are various continuously changing factors related to the invalid traffic problem; attack vectors, domain traffic profiles, and others. As a result, ground truth datasets for evaluating invalid traffic filtering solutions do not exist. Indeed, to the best of the authors' knowledge, previous related academic works [6, 37] rely on manual validation whereas proprietary commercial solutions do not offer any publicly available accuracy evaluation analysis.

In this subsection, we present for the first time an objective approach to (at least partially) assess the accuracy of invalid ad traffic detection tools. While we acknowledge our approach is far from ideal, it does not suffer from the severe limitations associated with manual validation such as: 1) limited scalability since only a handful of domains can be manually evaluated; 2) it suffers from human errors due to the lack of sufficient knowledge, lack of motivation, distractions, etc.

In particular, we propose a two-fold approach to validate the classification accuracy of Nameles:

(1) We compare the result of Nameles scoring with the following widely used ad tech metrics:

- **Bounce Rate:** This metric measures the fraction of sessions that only visit a single page in a domain. A low bounce rate is a strong indication of low-quality traffic.

Table 2: Value of external quality metrics associated with domains in each of the defined Confidence Classes in our dataset.

		No Conf.		Low Conf.		Moderate Conf.		High Conf.
Alexa Upstream traffic from Google and Facebook (%)	median	20	(-41%)	18.5	(-45%)	23.7	(-30%)	33.7
	IQR	21.05		20.19		24.19		32.84
Alexa Bounce rate (%)	median	41.8	(-27%)	40.9	(-29%)	35.3	(-39%)	57.5
	IQR	32.4		25.6		27.7		28.7
Alexa Search traffic (%)	median	8.1	(-35%)	7.7	(-38%)	5.5	(-56%)	12.5
	IQR	19.7		15.9		16.1		16.9
Alexa Total sites linking to the domain	median	9.2	(-75%)	131	(-62%)	256	(-27%)	348
	IQR	616		371		800		1,198
SimilarWeb Bounce rate (%)	median	51.5	(-12%)	38.8	(-34%)	34.9	(-40%)	58.6
	IQR	24.97		20.84		24.8		24.0
SimilarWeb Direct traffic (%)	median	43.1	(68%)	34.2	(34%)	38.1	(49%)	25.6
	IQR	39.5		37.0		34.6		27.8
SimilarWeb Search traffic (%)	median	21.2	(-31%)	29.3	(-5%)	19.5	(-37%)	30.9
	IQR	39.3		46.5		39.5		39.8

- **Traffic from popular publishers:** This metric represents the percentage of upstream traffic coming to the domain from popular publishers. In particular, the two publishers contributing a larger fraction of traffic to domains are Google and Facebook. Then, for our validation, we will compute the fraction of upstream traffic coming from Google and Facebook to a domain. A very low fraction of traffic coming from them may reveal the presence of low-quality traffic.

- **Search Traffic:** This metric measures the percentage of traffic coming to the domain from search engines. A very low search traffic percentage is often an indication of low-quality traffic.

- **Direct Traffic:** This metric measures the percentage of traffic that reaches the domain directly without being redirected from other websites. In this case, a large fraction of direct traffic is usually linked to low-quality traffic.

- **Number of sites linking to a domain:** An interesting domain attracting high-quality traffic would typically be linked from a large number of other sites. Contrary to this, domains associated with ad fraud or other malicious practices would typically be linked from a lower number of sites.

To obtain these metrics, we have queried two well-known services, Alexa [1] and SimilarWeb [34]. Table 2 presents the median and IQR values for the distribution of each one of these metrics for each Confidence Class. In addition, the table shows the relative difference of the median values of these metrics for the "No", "Low" and "Moderate" Confidence Classes in comparison to the "High" Confidence Class. We observed substantial differences (up to 75% in some cases) between the "High Confidence" Class and the rest.

(2) Low-quality domains and thus their ad inventory are likely to have a short-term lifetime (i.e., being more volatile) in the online advertising ecosystem for multiple reasons: they might be committing fraud and being detected and blacklisted or removed by fraudsters after short periods of times to avoid detection; they provide poor KPIs and thus the bidding prices for their inventory dramatically decrease making them no longer profitable, etc. We have measured the volatility associated with the domains of each of the confidence classes defined by Nameles. In particular, we have computed the fraction of domains of each class in our oldest dataset that are still present one year later in our most recent dataset. The result shows that (4.5; 1.2; 6.4; 35.0)% of domains within the ("No", "Low", "Moderate" and "High") confidence classes in our 16-17 dataset are still present in our 17-18 dataset. We observe that the volatility of the "High" Confidence Class is one order of magnitude smaller than in other classes[4].

The results from our two-fold objective validation suggest that our scoring mechanism accurately identifies legitimate domains and thus it is suitable for adoption by DSPs. Finally, it is also worth noting that we have worked together with experts from the Ad Tech industry over a period of 18 months to let them subjectively evaluate the results provided by Nameles in extensive trials. The satisfactory results obtained during these tests have led the World Federation of Advertisers to endorse Nameles.

7 RESULTS OBTAINED FROM NAMELES' EXECUTION

In this section, we present the results obtained from applying Nameles to our large-scale dataset. First, we analyze the distribution of domains and traffic across the defined Confidence Classes. Then, we quantify the positive impact that Nameles may have in the profitability of DSPs using as reference the economic model described in Section 3.

-Longitudinal Analysis of domains' confidence level: Figure 5 shows the fraction of domains and ad traffic (i.e., bid requests) belonging to each of the defined Confidence Classes in our 16-17 and 17-18 datasets, respectively. The main bar shows the average, and the error bar shows the 95% confidence interval across the days in the dataset.

On average, in our 16-17 dataset (11.2; 8.8; 33.3; 46.7)% of the traffic is associated with ("No", "Low", "Moderate" and "High") Confidence Classes. Whereas in our 17-18 dataset (7.0; 13.9; 36.3; 42.8)% of the traffic belong to the ("No", "Low", "Moderate" and "High") classes. We find an overall stable behavior across time. In particular, there exists ~20% low-quality traffic (belonging to the "No" and "Low" confidence classes) in both datasets. These results imply that, for instance, a DSP handling 50 B bid request per day using a policy that filters traffic belonging to "No" and "Low" classes would

[4]There is an apparent systemic volatility since even in the high confidence class only 35.0% of the domains remain in the 17-18 dataset. While there exists such systemic volatility, the reported one is likely to be inflated because our datasets represent a subsample of the overall ad inventory available at the measurement time.

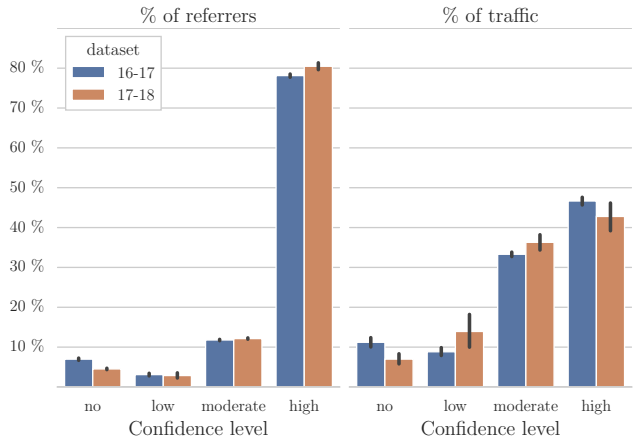


Figure 5: Percentage of domains and ad traffic in each of the Confidence Classes across the 62 days of the dataset. The main bar presents the average value whereas the error bars show the confidence interval at the 95%.

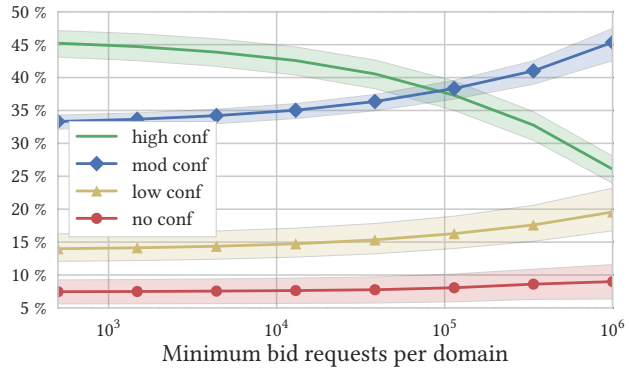


Figure 6: Percentage of ad traffic in each Confidence Class as function of the popularity (i.e., bid requests) of domains.

eliminate (on average) around 10 B (20%) bid requests every day. However, it is worth mentioning that there seems to be a transfer of ~5% traffic from the "No" to the "Low" confidence class between the 16-17 and 17-18 datasets, this may be interpreted as a slight improvement in the behavior of the ecosystem for DSPs filtering traffic at the "No" confidence level. Contrary, a transfer of ~4% traffic is observed from the "High" to the "Moderate" confidence class, which may likewise be interpreted as a slight worsening of the ecosystem behavior for DSPs accepting traffic exclusively from the "High" confidence class.

In addition, we analyzed how popularity relates to confidence. To this end, we computed the average (and standard deviation) fraction of traffic within each Confidence Class for domains with at least 500, 1 k, 10 k, 50 k, 100 k, and 1 M bid requests per day for our 16-17 dataset. Figure 6 shows the results. One may expect that as more popular domains are considered, the fraction of domains within the "High" Confidence Class would increase and the fraction in other groups would decrease. However, we observe the opposite trend between "Moderate" (which increases) and "High" (which decreases) classes. In the case of "Low" and "No" Classes we observe just a light increase after 100 k daily bid requests.

-Nameles' impact on DSPs' profitability: The results in the previous subsection provide specific figures on the filtering rates of Nameles at different confidence levels. For instance, in our 16-17 dataset, a filtering rate of 11.21% filters out traffic from domains with very rare traffic patterns that offer no confidence. A filtering rate of 20.04% eliminates traffic offering low or no confidence, and a filtering rate of 53.34% filters any domain that does not provide high confidence.

Using these filtering rates as input to the economic model presented in Section 3 gives us an estimation of the impact that Nameles is expected to have in the profitability of a DSP. The obtained results indicate that filtering at the "No Confidence", "Low Confidence" and "Moderate Confidence" level offer NPV (and EV) improvements in comparison to the scenario without filtering of 41, 54 and -204% (14, 19 and -71%). We observe that filtering at the "Moderate Confidence" level would not be recommended. On the other hand, filtering at the "No Confidence" or "Low Confidence" class leads to a strong positive economic impact. Note that results considering the 17-18 datasets are consistent.

8 NAMELES' APPLICABILITY AND EXTENSIBILITY

- **Applicability by other players:** In addition to DSPs, Nameles can be easily integrated with other players of the programmatic advertising supply chain such as Ad Exchanges or SSPs, which handle a representative fraction of the ad traffic of a given domain. Also, Publishers can integrate Nameles to self-generate the CS of their websites and mobile apps.

-Extensibility to other signals: The concept of entropy allows us to compute the CS considering different signals. In particular, in addition to the default signal used in this paper (CS of domains), we have computed the CS of individual IP addresses. To this end, we consider the traffic pattern generated by each IP address as the distribution of ad requests it sends across different domains. Having the CS for the IP and the domain within a bid request enriches the decision capacity of the DSP since such bid request can be dropped due to a low CS associated with the IP address as well. We have computed the CS for all IP addresses (with more than 500 entries) for every daily sample in our dataset and based on it, we have re-calculated the fraction of traffic belonging to the Confidence Classes more likely to be filtered by DSPs' policies (i.e., *None* and *Low* Confidence). Adding the information about the CS of IP addresses increases the fraction of traffic in these two categories in less than 0.5%. This result indicates that the application of Nameles at the level of domains suffices to identify more than 99% of low-quality ad traffic.

9 NAMELES' LIMITATIONS

In this section, we discuss the main limitations of Nameles and argue why despite them, Nameles is still an important contribution.

- **False Negatives:** They are represented by the invalid traffic not identified by Nameles. In the presence of Nameles, any attacker owning a domain d would be undetected if it is able to generate a normal (i.e., similar to the mass) traffic pattern for d. This is obviously doable, but it would require the attacker to increase the complexity of the attack. First, the attacker would need to infer the CS threshold over which it would not be detected. Second, an attacker performing n daily visits to its domain from m IPs leading to a low CS would need to either reduce n or increase m to increase its CS over the quality threshold defined by different DSPs. Both approaches lead to a revenue reduction.

To obtain a ballpark estimation of such reduction, we have computed the number of daily visits that all 8 K domains in the "No" and "Low" Confidence Classes in a given day of our dataset would need to remove in order to pass the threshold of the "Moderate" Class. We found that on average these domains would need to eliminate 38,5% visits leading to a roughly similar reduction in their revenue. Therefore, even when failing in the detection, Nameles contributes to reducing the profitability of possible attacks significantly.

-**False Positives:** They are represented by domains wrongly assigned a low CS. While false positives may have serious implications in other businesses, it is well-established in the programmatic advertising industry that false positives are not an issue for the buy-side, i.e., advertisers and DSPs, which are the target of our solution. The existing oversupply of ad spaces discussed in Section 3 guarantees that wrongly filtering a legitimate domain would not result in a lost opportunity of placing an ad that would be provided by other legitimate non-filtered domain. Therefore, we assert that false positives are not an important consideration in adopting Nameles.

- **Presence of NATs:** The widespread use of NATs may pollute the computation of a domain's CS since visits coming from different users behind a NATed IP address would be all assigned to that IP address, and thus Nameles could consider the traffic pattern of that domain more deterministic than it actually is. To analyze this potential limitation, we have re-computed the CS of all domains in our dataset using as input the visits coming from <IP, User Agent> pairs. The obtained results are similar to those presented in the paper. Given that the impact of NATs when considering <IP, User Agent> pairs as input is expected to be limited, we conclude that Nameles is minimally affected by the presence of NATs.

10 RELATED WORK

In addition to the abovementioned commercial proprietary solutions [11, 20, 42], the research community has also addressed the identification of invalid ad traffic. The proposed solutions focus on detecting invalid traffic at the sell side of the online advertising chain, i.e., publishers web pages [7] or delivered ads [5, 17]. These solutions analyze the interaction of the user with the web page or the served ad in order to identify commonly known attacks such as visits generated by bots [5] or redirection attacks [36]. None of these solutions are valid for DSPs. To the best of the author's knowledge, Stitelmant et al. [37] proposed the only alternative solution to Nameles able to operate at the DSP level. By analyzing the degree of overlapping in the IPs visiting two (or more) domains, their solution identifies potential invalid traffic. This is a complementary technique to our normalized entropy score, and thus both solutions can be used in parallel.

From a methodological perspective, there is a previous work that has used entropy to identify invalid video visits to a Chinese video portal [6]. The authors of this paper propose to use entropy as the final metric to assess the traffic quality and a semi-supervised classification that rely on manually labeled samples to differentiate between valid and invalid video traffic. However, as discussed in Section 5, the native Shannon entropy has an important drawback since its interpretation depends on the volume of associated events. To overcome this limitation, we use a Confidence Score based on a normalized version of entropy. Moreover, instead of using manual labeling of suspicious traffic, we define unsupervised statistically supported outlier detection method. Hence, Nameles clearly advance the state-of-the-art from a methodological perspective as well.

11 CONCLUSION

This paper introduces Nameles, a system for the detection of invalid ad traffic, which is one of the main problems faced by the online advertising industry. Nameles has been designed to meet the requirements of both advertisers and DSPs that together form the so-called buy-side of the programmatic advertising industry. On the one hand, Nameles is the first available open source solution for the identification of invalid traffic, responding to the advertisers' demand for transparency. On the other hand, the paper provides economic supported evidence that, contrary to conventional wisdom, show how DSPs may increase their profitability with invalid traffic filtering. For this, the applied solution needs to be highly scalable and operate in real time and at the level of individual bid requests. Nameles meets these requirements.

A Nameles prototype has been thoroughly tested in a realistic deployment. We demonstrate that even with modest resources, Nameles is able to process tens of billions of bid requests per day, with processing delays below 3ms per request and good detection accuracy. Moreover, applying Nameles on two datasets including each almost 2 B bid requests per day for a period of two months, we observe the presence of 20% invalid traffic. The evidenced performance of Nameles along with our open-source vision has led the World Federation of Advertisers to endorse Nameles as a solution to counter invalid traffic by the Adtech industry.

ACKNOWLEDGMENTS

This work has been partially supported by the Ministerio de Economía y Empresa (Spain) under project TEC2016-76795-C6-3-R and grant RyC-2015-17732, and by the European H2020 project SMOOTH (786741).

REFERENCES

[1] Alexa: actionable analytics for the web 2018. Retrieved Oct 31, 2018 from http://www.alexa.com
[2] Inc. Alphabet. 2018. 2017 Annual report. Retrieved Oct 31, 2018 from https://abc.xyz/investor/static/pdf/20171231_alphabet_10K.pdf
[3] Richard A Brealey, Stewart C Myers, Franklin Allen, and Pitabas Mohanty. 2012. *Principles of corporate finance.* Tata McGraw-Hill Education.
[4] Internet Advertising Bureau. 2017. *Transparency is the key to programmatic success.* Technical Report. Retrieved Oct 31, 2018 from https://www.iab.com/wp-content/

uploads/2015/08/IABDigitalSimplifiedProgrammaticAdvertisingTransparency.pdf
[5] Patricia Callejo, Ruben Cuevas, Angel Cuevas, and Mikko Kotila. 2016. Independent Auditing of Online Display Advertising Campaigns. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*. ACM.
[6] Liang Chen, Yipeng Zhou, and Dah Ming Chiu. 2014. Fake view analytics in online video services. In *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop*. ACM, 1.
[7] Vacha Dave, Saikat Guha, and Yin Zhang. 2013. Viceroi: catching click-spam in search ad networks. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 765–776.
[8] Shailin Dhar. 2016. *Mystery Shopping Inside the Ad Fraud Verification Bubble*. Technical Report. Retrieved Oct 31, 2018 from http://www.slideshare.net/ShailinDhar/mystery-shopping-inside-the-adverification-bubble
[9] Digiday. 2016. WTF is programmatic? Retrieved Oct 31, 2018 from http://digiday.com/wp-content/uploads/2016/07/WTF_programmatic-2016.pdf
[10] Distil. 2015. Distil Networks Releases New Data on The State of Digital Advertising Fraud. Retrieved Oct 31, 2018 from https://resources.distilnetworks.com/press-releases/distil-networks-releases-new-data-on-the-state-of-digital-advertising-fraud
[11] Double Verify 2018. Retrieved Oct 31, 2018 from http://www.doubleverify.com
[12] Emarketer. 2016. The Ad Industry's Focus on Fraud Has Intensified. Retrieved Oct 31, 2018 from https://www.emarketer.com/Article/Ad-Industrys-Focus-on-Fraud-Has-Intensified/1014430
[13] Inc. Facebook. 2018. 2017 Annual report. Retrieved Oct 31, 2018 from https://s21.q4cdn.com/399680738/files/doc_financials/annual_reports/FB_AR_2017_FINAL.pdf
[14] Joint Industry Committee for Web Standards. 2016. *Traffic Fraud: Good Practice Principles for Reducing Risk to Exposure to Ad Fraud*. Technical Report. Retrieved Oct 31, 2018 from https://jicwebs.org/wp-content/uploads/2018/02/JICWEBS_Good_Practice_Principles_Ad_Fraud_May_2016.pdf
[15] Google. 2018. DoubleClick RTB Protocol. Latency Restrictions and Peering. Retrieved Oct 31, 2018 from https://developers.google.com/ad-exchange/rtb/peer-guide
[16] Gurufocus 2018. Retrieved Oct 31, 2018 from http://www.gurufocus.com
[17] Hamed Haddadi. 2010. Fighting online click-fraud using bluff ads. *ACM SIGCOMM Computer Communication Review* 40, 2 (2010), 21–25.
[18] Lucy Handley. 2017. Businesses could lose $16.4 billion to online advertising fraud in 2017: Report. Retrieved Oct 31, 2018 from http://www.cnbc.com/2017/03/15/businesses-could-lose-164-billion-to-online-advert-fraud-in-2017.html
[19] Pieter Hintjens. 2013. *ZeroMQ: messaging for many applications*. "O'Reilly Media, Inc.".
[20] Integral Ad Science (IAS) 2018. Retrieved Oct 31, 2018 from https://integralads.com
[21] Investopedia. 2017. DCF Analysis: Coming Up With A Fair Value. Retrieved Oct 31, 2018 from http://www.investopedia.com/university/dcf/dcf4.asp
[22] Investopedia. 2018. Rate of Return. Retrieved Oct 31, 2018 from www.investopedia.com/terms/r/rateofreturn.asp
[23] IAB Technology Laboratory. [n. d.]. *Real Time Bidding (RTB) Project. OpenRTB API Specification Version 2.5*.
[24] Wenke Lee and Dong Xiang. 2001. Information-theoretic measures for anomaly detection. In *Security and Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium on*. IEEE, 130–143.
[25] Investopedia. J.B. Maverick. 2015. What is the long-term average growth rate of the telecommunications sector? Retrieved Oct 31, 2018 from http://www.investopedia.com/ask/answers/071515/what-longterm-average-growth-rate-telecommunications-sector.asp
[26] Marciel Miriam, Cuevas, Ruben, Banchs Albert, Gonzalez Roberto, Traverso Stefano, Ahmed Mohamed, and Azcorra Arturo. 2016. Understanding the Detection of View Fraud in Video Content Portals. (2016).
[27] World Federation of Advertisers. 2016. *Compendium of ad fraud knowledge for media investors*. Technical Report.
[28] ANA & White Ops. 2016. *2015 Bot Baseline: Fraud in Digital Advertising*. Technical Report.
[29] White Ops. 2016. *The Methbot Operation*. Technical Report.
[30] PwC. 2018. *IAB internet advertising revenue report 2017 full year results*. Technical Report. Retrieved Oct 31, 2018 from https://www.iab.com/wp-content/uploads/2018/05/IAB-2017-Full-Year-Internet-Advertising-Revenue-Report.REV_.pdf
[31] P. J. Rousseeuw and A. M. Leroy. 1987. *Robust Regression and Outlier Detection*. John Wiley & Sons, Inc., New York, NY, USA.
[32] Said E Said and David A Dickey. 1984. Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika* 71, 3 (1984), 599–607.
[33] C. E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal* 27, 3 (July 1948), 379–423.
[34] SimilarWeb: Website Traffic & Mobile App Analytics 2018. Retrieved Oct 31, 2018 from https://www.similarweb.com
[35] Smaato. 2014. Mobile RTB Insights Report Q3 2014. Retrieved Oct 31, 2018 from http://www.newsroom-publicismedia.fr/wp-content/uploads/2014/11/Smaato_Q3_2014_RTB_Report.pdf
[36] Kevin Springborn and Paul Barford. 2013. Impression Fraud in On-line Advertising via Pay-Per-View Networks. In *22nd USENIX Security Symposium (USENIX Security 13)*. USENIX, 211–226.
[37] Ori Stitelman, Claudia Perlich, Brian Dalessandro, Rod Hook, Troy Raeder, and Foster Provost. 2013. Using co-visitation networks for detecting large scale online display advertising exchange fraud. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1240–1248.
[38] IHS Technology. 2017. *The Economic Contribution of Digital Advertising in Europe*. Technical Report. https://datadrivenadvertising.eu/wp-content/uploads/2017/09/DigitalAdvertisingEconomicContribution_FINAL-1.pdf
[39] Financial Times. 2017. Google charges for YouTube ads even when viewed by robots. Retrieved Oct 31, 2018 from https://www.ft.com/content/f9da727c-6207-11e5-9846-de406ccb37f2
[40] Arno Wagner and Bernhard Plattner. 2005. Entropy based worm and anomaly detection in fast IP networks. In *Enabling Technologies: Infrastructure for Collaborative Enterprise, 2005. 14th IEEE International Workshops on*. IEEE, 172–177.
[41] WFA. 2014. WFA guide to Programmatic Media. Retrieved Oct 31, 2018 from https://www.wfanet.org/app/uploads/2017/04/programmatic.pdf
[42] White Ops 2018. Retrieved Oct 31, 2018 from https://www.whiteops.com
[43] IMPERVA INCAPSULA. Igal Zeifman. 2017. Bot Traffic Report 2016. Retrieved Oct 31, 2018 from https://www.incapsula.com/blog/bot-traffic-report-2016.html
[44] Weinan Zhang, Shuai Yuan, and Jun Wang. 2014. Optimal real-time bidding for display advertising. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1077–1086.

Learning How to Correct a Knowledge Base from the Edit History

Thomas Pellissier Tanon
Télécom ParisTech
ttanon@enst.fr

Camille Bourgaux
DI ENS, CNRS, ENS, PSL Univ. & Inria
camille.bourgaux@ens.fr

Fabian Suchanek
Télécom ParisTech
suchanek@enst.fr

ABSTRACT

The curation of a knowledge base is a crucial but costly task. In this work, we propose to take advantage of the edit history of the knowledge base in order to learn how to correct constraint violations. Our method is based on rule mining, and uses the edits that solved some violations in the past to infer how to solve similar violations in the present. The experimental evaluation of our method on Wikidata shows significant improvements over baselines.

KEYWORDS

knowledge base; history; data cleaning; rule mining; Wikidata

ACM Reference Format:
Thomas Pellissier Tanon, Camille Bourgaux, and Fabian Suchanek. 2019. Learning How to Correct a Knowledge Base from the Edit History. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313584

1 INTRODUCTION

Knowledge bases (KBs) play a key role in many applications and lay at the core of the Semantic Web. They contain entities (such as persons, cities, etc.) and statements about them (such as relationships between persons, places of residence, etc.). In this work, we focus on RDFS-style KBs, such as Wikidata [38], YAGO [35] or DBpedia [7]. The data quality of a KB is crucial for its usability. However, it is usually very costly to check the correctness of the data, because KBs can be huge (Wikidata, e.g., contains about 50 millions entities). Moreover, KBs are often built using methods that are error-prone. For instance YAGO and DBpedia are automatically extracted from Wikipedia. Wikidata, for its part, is a collaborative KB that anyone can edit, with more than 18,000 active contributors.

A way of avoiding or at least detecting some of the flaws in the data is to impose *constraints* on the KB. Such constraints can enforce that some information must be present (e.g., imposing that every human being has a birth date), or that some statements may not occur (e.g., ensuring that a person is not also a city). Constraints are related to, but different from, ontological rules: A constraint imposes a certain condition, whereas an ontological rule infers certain statements. For example, consider a KB that contains the statement "Spinoza is a human being" without knowing any birth date for Spinoza, and consider a rule "All human beings have a birth date". If the rule is taken as an ontological rule, then it would just infer that Spinoza has some birth date. If the rule is taken as a constraint, in contrast, the KB would be considered incorrect. Constraints are thus similar in spirit to database integrity constraints. In practice, constraints often have exceptions. Therefore, it is useful to allow data that does not respect them (in Wikidata, e.g., constraint violations are simply flagged). Nonetheless, by design, most of the constraint violations are not exceptions but actual errors and proposing to repair them is a good starting point when it comes to improving KB quality.

In this paper, we aim at learning how to repair constraint violations. Our goal is to help a KB editor by suggesting how to clean the data locally (providing a solution to a particular constraint violation) or globally (providing rules that can be automatically applied to all constraint violations of a given form once validated by the editor). To do that, we take advantage of the *edit history* of the KB. We use it to mine *correction rules* that express how different kinds of constraint violations are usually solved. To the best of our knowledge, this is the first work that builds on past users corrections in order to infer possible new ones. We validate our framework experimentally on Wikidata, for which the whole edit history of more than 700 millions edits is available. Our experiments show substantial improvements over baselines. More concretely, our contributions are as follows:

- a formal definition of the problem of correction rule mining,
- a dataset of more than 67M past corrections for ten different kinds of Wikidata constraints (13k constraints in total),[1]
- a correction rule mining algorithm, together with an implementation for Wikidata, CorHist,[2]
- a suggestion tool for users to correct data based on our mined correction rules,[3]
- an experimental evaluation based both on the prediction of the corrections in the history and on user validation of the suggested local corrections.

2 RELATED WORK

We start with a brief discussion of works relevant to our problem along three axes: constraints for KBs, KB cleaning, and rule learning.

Constraints. Constraints have long been used in databases and KBs to express rules that the data should follow. Databases typically operate under the closed world assumption, where missing facts are considered to be false. This allows for "completeness" constraints such as tuple generating dependencies. KBs, in contrast, operate under the open world assumption, where missing

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313584

[1] available at https://doi.org/10.6084/m9.figshare.7712720
[2] available at https://github.com/Tpt/corhist
[3] available at https://tools.wmflabs.org/wikidata-game/distributed/#game=43

facts are not necessarily false. They thus classically have only "correctness" constraints, such as disjointness or functionality axioms (corresponding to special cases of denial constraints and equality generating dependencies in databases).

To express also completeness constraints, several works propose to use description logics, with varying semantics [28, 37]. Another possibility is to use queries that should or should not hold as constraints (see e.g., [24] for methods for writing constraint queries in SPARQL). Other approaches define constraint languages to specify conditions for RDF graph [9] validation, such as SHACL [22] or ShEx [8]. It has been argued in [30] that description logics under the closed world assumption are also suitable for constraint checking in RDF, which can then be implemented with SPARQL queries. In our work, we follow a similar path, using description logic axioms as constraints for RDFS KBs, because it corresponds best to what we observe in current real-world KBs.

Contrary to the above works, we do not aim at *expressing* constraints, but at *repairing* their violations. The correction rules we learn for this purpose are similar in spirit to active integrity constraints [11], which specify for each constraint a set of possible repair actions. This type of constraints has recently been applied to description logic KBs as well [32]. Conditioned active integrity constraints add conditions for choosing among the possible actions, and we propose, in a similar spirit, to take into account the context of the constraint violation to correct it. Different from the existing work [11, 32], our goal is to mine correction rules automatically from the edit history of the KB.

Knowledge base cleaning. Several recent approaches have dealt with the interactive cleaning of KBs. The proposed methods detect when a constraint is violated, compute the responsible facts, and then interact with the user to find out how to update the KB. The goal is then to minimize the number of questions the user has to answer. This is done in various ways, which include taking into account the dependencies among the facts to check or the interaction between several constraints violations to define heuristics to choose the best question to ask the user [2, 3, 5, 6].

Other approaches to improve the quality of a KB rely on statistics, clustering, or structural aspects of the KBs. The work of [31] uses statistics to add missing types to the KB, and to detect wrong statements. The work of [25] exploits the observation that cycles in the KB often contain wrong "IsA" relations. Again other approaches [1] use crowdsourcing to detect Linked Data quality issues. We refer the reader to Section 7.2 of [1] for a recent overview of approaches for data quality assessment.

Our method also exploits KB constraints. However, it differs from the above in that it *learns the corrections automatically* from the edit history. It thus taps a source of knowledge that has so far not been exploited.

Rule learning. Mining logical rules by finding correlations in a dataset is a well-established research topic. In particular, learning patterns in the data can be used for completing KBs [14, 36]. An algorithm for learning conjunctive patterns from a KB enriched with a set of rules is described in [20]. Methods similar to association rule mining have also been used for induction of new ontological rules from a KB [33]. A more recent trend is to use embedding-based models for KB completion. A comparison between these models and usual rule learning approaches is reported in [27] and significant recent works in this area include [19, 39, 40].

In this paper, we use a vanilla rule mining algorithm inspired by [14]. Our contribution is not the rule mining per se, but the application of rule mining to the edit history of a KB in order to mine correction rules. This avenue has, to the best of our knowledge, never been investigated.

3 PRELIMINARIES

In this work, we use description logics (DL) [4] as KB language and as constraint language, because they are the foundation of the Semantic Web standard OWL [16].

Syntax. We assume a set $\mathsf{N_C}$ of *concept names* (unary predicates, also called classes), a set $\mathsf{N_R}$ of *role names* (binary predicates, also called properties), and a set $\mathsf{N_I}$ of *individuals* (also called constants). An *ABox* (dataset) is a set of *concept or role assertions* of the form $A(a)$ or $R(a, b)$, where $A \in \mathsf{N_C}$, $R \in \mathsf{N_R}$, $a, b \in \mathsf{N_I}$. A *TBox* (ontology) is a set of axioms whose form depends on the DL $\mathcal{L}$ in question, and expresses relationships between concepts and roles (e.g., concept or role hierarchies, role domains and ranges...). A *knowledge base* (KB) $\mathcal{K} = \mathcal{T} \cup \mathcal{A}$ is the union of an ABox $\mathcal{A}$ and a TBox $\mathcal{T}$.

In this work, we assume that $\mathcal{T}$ is a *flat QL* TBox [23], i.e., that $\mathcal{L}$ differs from the standard *RDF Schema (RDFS)* [17] only by allowing inverse roles in role inclusions. More precisely, $\mathcal{T}$ can contain concept inclusions of the form $A_1 \sqsubseteq A_2$ (subclass), $\exists P \sqsubseteq A$ (domain or range), and role inclusions $P_1 \sqsubseteq P_2$ (subproperty), where $A_{(i)} \in \mathsf{N_C}$ and $P_{(i)} := R \mid R^-$ with $R \in \mathsf{N_R}$.

A KB can also be written as a set of *RDF triples* $\langle s, p, o\rangle$ where s is the *subject*, p is the *property*, and o the *object*, using special properties to translate concept membership and relationships between concepts and roles [29]. A concept assertion $A(a)$ is written as $\langle a, \texttt{rdf:type}, A\rangle$, and a role assertion $R(a, b)$ as $\langle a, R, b\rangle$. Flat QL TBox axioms can also be represented by single triples. For example, $A_1 \sqsubseteq A_2$ is written as $\langle A_1, \texttt{rdfs:subClassOf}, A_2\rangle$ and $\exists R^- \sqsubseteq A$ is written as $\langle R, \texttt{rdfs:range}, A\rangle$.

Semantics. We recall the standard semantics of DL KBs. An *interpretation* has the form $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is a non-empty set and $\cdot^\mathcal{I}$ is a function that injectively maps each $a \in \mathsf{N_I}$ to $a^\mathcal{I} \in \Delta^\mathcal{I}$ (unique name assumption), $\top$ to $\Delta^\mathcal{I}$, each $A \in \mathsf{N_C}$ to $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, and each $R \in \mathsf{N_R}$ to $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. The function $\cdot^\mathcal{I}$ is straightforwardly extended to general concepts and roles, e.g. $(\neg B)^\mathcal{I} = \Delta^\mathcal{I} \setminus B^\mathcal{I}$, $(R^-)^\mathcal{I} = \{(c, d) \mid (d, c) \in R^\mathcal{I}\}$, $\{a_1, \ldots, a_n\} = \{a_1^\mathcal{I}, \ldots, a_n^\mathcal{I}\}$, $(\exists P \cdot B)^\mathcal{I} = \{c \mid \exists d : (c, d) \in P^\mathcal{I}, d \in B^\mathcal{I}\}$, $(B_1 \sqcap B_2)^\mathcal{I} = B_1^\mathcal{I} \cap B_2^\mathcal{I}$, $(B_1 \sqcup B_2)^\mathcal{I} = B_1^\mathcal{I} \cup B_2^\mathcal{I}$. An interpretation $\mathcal{I}$ *satisfies* an inclusion $G \sqsubseteq H$, if $G^\mathcal{I} \subseteq H^\mathcal{I}$; it satisfies an axiom $(\text{func } P)$ if $P^\mathcal{I}$ is functional; it satisfies an axiom $(\text{trans } P)$ if $P^\mathcal{I}$ is transitive; and it satisfies $A(a)$ (resp. $R(a, b)$), if $a^\mathcal{I} \in A^\mathcal{I}$ (resp. $(a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$). We write $\mathcal{I} \models \alpha$ if $\mathcal{I}$ satisfies the DL axiom α.

An interpretation $\mathcal{I}$ is a *model* of $\mathcal{K} = \mathcal{T} \cup \mathcal{A}$ if $\mathcal{I}$ satisfies all axioms in $\mathcal{K}$. A KB is *consistent* if it has a model. A KB $\mathcal{K}$ *entails* a DL axiom α if $\mathcal{I} \models \alpha$ for every model $\mathcal{I}$ of $\mathcal{K}$.

Queries. A *conjunctive query* (CQ) takes the form $q(\vec{x}) = \exists \vec{y} \psi(\vec{x}, \vec{y})$, where ψ is a conjunction of atoms of the form $A(t)$ or $R(t, t')$ or of equalities $t = t'$, where t, t' are individual names or variables from

$\vec{x} \cup \vec{y}$. If $\vec{x} = \emptyset$, q is a Boolean CQ (BCQ). A BCQ q is *satisfied* by an interpretation $\mathcal{I}$, written $\mathcal{I} \models q$, if there is a homomorphism π mapping the variables and individual names of q into $\Delta^{\mathcal{I}}$ such that: $\pi(a) = a^{\mathcal{I}}$ for every $a \in \mathsf{N_I}$, $\pi(t) \in A^{\mathcal{I}}$ for every concept atom $A(t)$ in ψ, $(\pi(t), \pi(t')) \in R^{\mathcal{I}}$ for every role atom $R(t, t')$ in ψ, and $\pi(t) = \pi(t')$ for every $t = t'$ in ψ. We also consider as BCQs the queries true and false which are respectively always and never satisfied by an interpretation. A BCQ q is *entailed* from $\mathcal{K}$, written $\mathcal{K} \models q$, iff q is satisfied by every model of $\mathcal{K}$. A tuple of constants $\vec{a}$ is a (certain) *answer* to a CQ $q(\vec{x})$ if $\mathcal{K} \models q(\vec{a})$ where $q(\vec{a})$ is the BCQ obtained by replacing the variables from $\vec{x}$ by the constants $\vec{a}$. We denote by answers$(q(\vec{x}), \mathcal{K})$ the set of answers of $q(\vec{x})$ over $\mathcal{K}$. A *union of CQs* (UCQ) is a disjunction of CQs and has as answers the union of the answers of the CQs it contains.

Canonical model. It is well-known that a flat QL KB $\mathcal{K}$ has a *canonical model* $\mathcal{I}_{\mathcal{K}}$ such that for every BCQ q, $\mathcal{K} \models q$ iff $\mathcal{I}_{\mathcal{K}} \models q$. The domain of $\mathcal{I}_{\mathcal{K}}$ is the set of individual names that occur in $\mathcal{K}$ and $A^{\mathcal{I}_{\mathcal{K}}} = \{a \mid \mathcal{A} \models B(a), \mathcal{T} \models B \sqsubseteq A\}$ for every $A \in \mathsf{N_C}$ and $R^{\mathcal{I}_{\mathcal{K}}} = \{(a, b) \mid \mathcal{A} \models P(a, b), \mathcal{T} \models P \sqsubseteq R\}$ for every $R \in \mathsf{N_R}$ [23].

4 CONSTRAINTS

This section defines the constraints that can be imposed on a KB, and relates the problem of checking that a KB complies with these constraints to CQ answering over this KB.

Defining constraints. In this work, we consider two types of constraints: consistency constraints (which express that some statements are contradictory), and completeness constraints (which impose that certain statements should hold in the KB as soon as some others do). While violations of consistency constraints can only be solved by removing statements, those of completeness constraints can also be solved by adding statements.

Definition 1 (Constraint): *Constraints are built from complex concepts and roles defined by the following grammar rules:*

$P := R \mid R^-$

$B := \top \mid A \mid B \sqcap B \mid B \sqcup B \mid \exists P \cdot \{a_1, \ldots, a_n\} \mid \exists P \cdot B$

where $R \in \mathsf{N_R}$, $A \in \mathsf{N_C}$, $a_1, \ldots, a_n \in \mathsf{N_I}$.

A consistency constraint *is a concept inclusion of the form* $B_1 \sqsubseteq \neg B_2$ *or of the form* $B \sqsubseteq \{a_1, \ldots, a_n\}$*, a role inclusion of the form* $P_1 \sqsubseteq \neg P_2$*, or a functionality axiom of the form* (func P).

A completeness constraint *is a concept inclusion of the form* $B_1 \sqsubseteq B_2$*, a role inclusion of the form* $P_1 \sqsubseteq P_2$*, or a transitivity axiom of the form* (trans P).

This definition of constraints covers the "constraining" versions of the majority of the most popular DL axioms used on the Web of Data according to the ranking done by [15].

We assume without loss of generality that concepts of the form $B_1 \sqcup B_2$ or $\{a_1, \ldots, a_n\}$ with $n > 1$ appear only on the right side of inclusions, and not at all in negative inclusions of the form $B_1 \sqsubseteq \neg B_2$. For example, we assume that $\exists P \cdot (B_1 \sqcup B_2) \sqsubseteq C$ is rewritten as $\exists P \cdot B_1 \sqsubseteq C, \exists P \cdot B_2 \sqsubseteq C$, and $B \sqsubseteq \neg \exists P \cdot \{a_1, \ldots, a_n\}$ is rewritten as $B \sqsubseteq \neg \exists P \cdot \{a_1\}, \ldots, B \sqsubseteq \neg \exists P \cdot \{a_n\}$. As usual, we abbreviate $\exists P \cdot \top$ as $\exists P$.

Example 1: *As a running example, we consider the following KB* $\mathcal{K} = \mathcal{T} \cup \mathcal{A}$ *and set of constraints* $\mathcal{C}$ *inspired from Wikidata. Our TBox* $\mathcal{T}$ *expresses that human beings and deities are persons. Our ABox* $\mathcal{A}$ *provides information on several individuals. Our constraints* $\mathcal{C}$ *state that there are three possible genders (consistency constraint), that those who have a mother or are a mother must be persons or animals, that a mother must have gender female, and that if a has mother b, then b must have child a (completeness constraints).*

$\mathcal{T} = \{ \mathsf{Human} \sqsubseteq \mathsf{Person},\ \mathsf{Deity} \sqsubseteq \mathsf{Person} \}$

$\mathcal{A} = \{ \mathsf{Deity}(\mathit{Zeus}),\ \mathsf{Deity}(\mathit{Rhea}),\ \mathsf{hasGender}(\mathit{Zeus}, \mathit{masculine}),$
$\mathsf{hasGender}(\mathit{Rhea}, \mathit{female}),\ \mathsf{hasMother}(\mathit{Zeus}, \mathit{Rhea}),$
$\mathsf{hasChild}(\mathit{Rhea}, \mathit{Zeus}),\ \mathsf{Human}(\mathit{Spinoza}),$
$\mathsf{hasMother}(\mathit{Spinoza}, \mathit{Marques}) \}$

$\mathcal{C} = \{ \Gamma_0 = \exists \mathsf{hasGender}^- \sqsubseteq \{\mathit{male}, \mathit{female}, \mathit{nonbinary}\},$
$\Gamma_1 = \exists \mathsf{hasMother} \sqsubseteq \mathsf{Person} \sqcup \mathsf{Animal},$
$\Gamma_2 = \exists \mathsf{hasMother}^- \sqsubseteq \mathsf{Person} \sqcup \mathsf{Animal},$
$\Gamma_3 = \exists \mathsf{hasMother}^- \sqsubseteq \exists \mathsf{hasGender} \cdot \{\mathit{female}\},$
$\Gamma_4 = \mathsf{hasMother} \sqsubseteq \mathsf{hasChild}^- \}$ ◁

We say that a KB $\mathcal{K}$ *satisfies a constraint* $\Gamma \in \mathcal{C}$ if $\mathcal{I}_{\mathcal{K}} \models \Gamma$, where $\mathcal{I}_{\mathcal{K}}$ is the canonical model of $\mathcal{K}$. Otherwise, $\mathcal{K}$ *violates* Γ.

Example 2: *In our running example, the KB* $\mathcal{K}$ *satisfies* Γ_1 *since* $\exists \mathsf{hasMother}^{\mathcal{I}_{\mathcal{K}}} = \{\mathit{Zeus}, \mathit{Spinoza}\}$ *and* $\mathcal{I}_{\mathcal{K}} \models \mathsf{Person}(\mathit{Zeus})$ *and* $\mathcal{I}_{\mathcal{K}} \models \mathsf{Person}(\mathit{Spinoza})$*. However, it violates* Γ_2 *because* $\mathcal{I}_{\mathcal{K}} \not\models \mathsf{Person}(\mathit{Marques}) \vee \mathsf{Animal}(\mathit{Marques})$ *while* $\mathit{Marques} \in \exists \mathsf{hasMother}^{-\mathcal{I}_{\mathcal{K}}}$*. It violates* Γ_3 *and* Γ_4 *for similar reasons. Finally,* $\{\mathsf{hasGender}(\mathit{Zeus}, \mathit{masculine}), \Gamma_0\}$ *has no model because of the unique name assumption (which enforces that the interpretation of masculine differs from those of male, female and nonbinary). Hence,* $\mathcal{I}_{\mathcal{K}}$ *cannot be a model of* Γ_0*. Thus,* $\mathcal{K}$ *violates* Γ_0. ◁

Note the semantic difference between the constraints and the axioms of the TBox: The axiom (Human $\sqsubseteq$ Person) in the TBox makes every human an answer to the query asking for persons. In contrast, if we had put the axiom in the set of constraints, it would have required all human beings in the KB to be explicitly marked as persons. As another example, consider the axiom (func hasBirthdate) (which says that everyone can have at most one birth date). If this axiom appears in the TBox, it renders the KB inconsistent whenever a person is given two distinct dates of birth. This has severe consequences on the reasoning capabilities, since everything is entailed from an inconsistent KB. If this axiom is in the set of constraints, in contrast, then distinct dates of birth lead only to the violation of the constraint. This gives us relevant information without having any impact on the usability of the KB.

Checking constraints. We show that our setting allows us to check constraint satisfaction via CQ answering. For this purpose, we use a function π, which maps each constraint $\Gamma \in \mathcal{C}$ to a rule of the form $\exists \vec{y} \varphi(\vec{x}, \vec{y}) \rightarrow \exists \vec{z} \varphi'(\vec{x}, \vec{z})$. This function is defined recursively as shown in Table 1. The left side of the rule is called the *body* and its right side the *head*.

Example 3: *In our running example, we obtain the following rules:*

$\Gamma_0(x) : \exists y \mathsf{hasGender}(y, x) \rightarrow x = \mathit{male} \vee x = \mathit{female} \vee x = \mathit{nonbinary}$

$\Gamma_1(x) : \exists y \mathsf{hasMother}(x, y) \rightarrow \mathsf{Person}(x) \vee \mathsf{Animal}(x)$

$\Gamma_2(x) : \exists y \mathsf{hasMother}(y, x) \rightarrow \mathsf{Person}(x) \vee \mathsf{Animal}(x)$

$\Gamma_3(x) : \exists y \mathsf{hasMother}(y, x) \rightarrow \exists z (\mathsf{hasGender}(x, z) \wedge z = \mathit{female})$

$\Gamma_4(x, y) : \mathsf{hasMother}(x, y) \rightarrow \mathsf{hasChild}(y, x)$ ◁

Table 1: Translation of DL axioms into rules. Variables that appear in the right column and not in the left one are fresh.

$\pi(\top, x)$	=	true
$\pi(A, x)$	=	$A(x)$
$\pi(\{a_1, \ldots, a_n\}, x)$	=	$x = a_1 \vee \cdots \vee x = a_n$
$\pi(R, x, y)$	=	$R(x, y)$
$\pi(R^-, x, y)$	=	$\pi(R, y, x)$
$\pi(B_1 \sqcap B_2, x)$	=	$\pi(B_1, x) \wedge \pi(B_2, x)$
$\pi(B_1 \sqcup B_2, x)$	=	$\pi(B_1, x) \vee \pi(B_2, x)$
$\pi(\exists P \cdot B, x)$	=	$\exists y(\pi(P, x, y) \wedge \pi(B, y))$
$\pi(B \sqsubseteq C)$	=	$\pi(B, x) \rightarrow \pi(C, x)$
$\pi(P \sqsubseteq Q)$	=	$\pi(P, x, y) \rightarrow \pi(Q, x, y)$
$\pi(B \sqsubseteq \neg C)$	=	$\pi(B, x) \wedge \pi(C, x) \rightarrow$ false
$\pi(P \sqsubseteq \neg Q)$	=	$\pi(P, x, y) \wedge \pi(Q, x, y) \rightarrow$ false
$\pi(\text{func } P)$	=	$\pi(P, x, y) \wedge \pi(P, x, z) \rightarrow y = z$
$\pi(\text{trans } P)$	=	$\pi(P, x, y) \wedge \pi(P, y, z) \rightarrow \pi(P, x, z)$

The following proposition shows that this transformation is sound and that the rule body and head can be rewritten as CQ and UCQ.

Proposition 1: *For every constraint $\Gamma \in \mathcal{C}$, $\pi(\Gamma)$ can be rewritten as a rule $\Gamma(\vec{x}) : b(\vec{x}) \rightarrow h(\vec{x})$ where $b(\vec{x})$ is a CQ, $h(\vec{x})$ is a UCQ, and for every flat QL KB $\mathcal{K}$, $\mathcal{K}$ satisfies Γ iff* $\text{answers}(b(\vec{x}), \mathcal{K}) \subseteq \text{answers}(h(\vec{x}), \mathcal{K})$.

Proof. By our assumptions on the form of the concepts that occur in the left side of the inclusions or in the right side of a negative inclusion, $b(\vec{x}) := \exists \vec{y} \varphi(\vec{x}, \vec{y})$ is a CQ. It is easy to show by structural induction that for every concept B (resp. role P), $\pi(B, x)$ (resp. $\pi(P, x, y)$) can be written as a UCQ $q(x)$ (resp. $q(x, y)$) and that $\text{answers}(q(x), \mathcal{K}) = B^{\mathcal{I}_\mathcal{K}}$ (resp. $\text{answers}(q(x, y), \mathcal{K}) = P^{\mathcal{I}_\mathcal{K}}$). If Γ is a completeness constraint of the form $B_1 \sqsubseteq B_2$ (resp. $P_1 \sqsubseteq P_2$), or a consistency constraint of the form $B \sqsubseteq \{a_1, \ldots, a_n\}$, the result follows immediately since $\mathcal{K}$ satisfies Γ iff $\mathcal{I}_\mathcal{K} \models \Gamma$. If Γ is a consistency constraint of the form $B_1 \sqsubseteq \neg B_2$ (resp. $P_1 \sqsubseteq \neg P_2$), $\text{answers}(b(\vec{x}), \mathcal{K}) = B_1^{\mathcal{I}_\mathcal{K}} \cap B_2^{\mathcal{I}_\mathcal{K}}$ (resp. $\text{answers}(b(\vec{x}), \mathcal{K}) = P_1^{\mathcal{I}_\mathcal{K}} \cap P_2^{\mathcal{I}_\mathcal{K}}$) and is empty iff Γ is satisfied. Since in this case $h(\vec{x}) :=$ false, $\text{answers}(h(\vec{x}), \mathcal{K}) = \emptyset$, and the desired relation holds. If Γ is of the form (func P), since the answers of $h(x, y, z) := y = z$ over $\mathcal{K}$ are all possible tuples of the form (a, b, b), $P^{\mathcal{I}_\mathcal{K}}$ is functional iff $\text{answers}(b(x, y, z), \mathcal{K}) \subseteq \text{answers}(h(x, y, z), \mathcal{K})$. Finally, if Γ is of the form (trans P), $b(x, y, z) := \pi(P, x, y) \wedge \pi(P, y, z)$ and $h(x, y, z) := \pi(P, x, z)$, and it is easy to see that $P^{\mathcal{I}_\mathcal{K}}$ is transitive iff $\text{answers}(b(x, y, z), \mathcal{K}) \subseteq \text{answers}(h(x, y, z), \mathcal{K})$. □

Constraint violations. A *constraint instance* $\Gamma(\vec{a})$ of a constraint $\Gamma(\vec{x})$ is obtained by replacing the variables $\vec{x}$ by the individual names $\vec{a}$ in $\Gamma(\vec{x})$. This notion allows us to define constraint violations:

Definition 2 (Constraint violation): *A* violation of a constraint $\Gamma(\vec{x})$ *in $\mathcal{K}$ is a minimal subset $\mathcal{V} \subseteq \mathcal{K}$ such that there exists $\vec{a}$ such that $\mathcal{V}$ violates $\Gamma(\vec{a})$ and $\mathcal{K}$ violates $\Gamma(\vec{a})$. We denote by* $\text{Violations}(\mathcal{K}, \Gamma(\vec{x}))$ *the set of violations of $\Gamma(\vec{x})$.*

In this definition, the requirement that $\mathcal{K}$ violates $\Gamma(\vec{a})$ may seem superfluous. Yet, if Γ is a completeness constraint, it may be the case that some $\mathcal{V} \subseteq \mathcal{K}$ violates $\Gamma(\vec{a})$, while $\mathcal{K}$ satisfies it.

Example 4: *In our running example, it is easy to see that the subset $\mathcal{V}_0 = \{\text{hasGender}(Zeus, masculine)\}$ is a violation of Γ_0. Consider now $\mathcal{V} = \{\text{hasMother}(Spinoza, Marques)\}$. $\mathcal{V}$ is a violation of Γ_2, Γ_3 and Γ_4. Indeed, it violates $\Gamma_2(Marques)$, $\Gamma_3(Marques)$, and $\Gamma_4(Spinoza, Marques)$ and $\mathcal{K}$ does not satisfy any of these constraint instances. However, even if $\mathcal{V}$ violates $\Gamma_1(Spinoza)$, $\mathcal{V}$ is not a violation of Γ_1 because $\{\text{Human}(Spinoza), \text{Human} \sqsubseteq \text{Person}\} \subseteq \mathcal{K}$ satisfies the head of $\Gamma_1(Spinoza)$.* ◁

If a constraint instance $\Gamma(\vec{a})$ is violated, $\vec{a} \in \text{answers}(b(\vec{x}), \mathcal{K})$ and $\vec{a} \notin \text{answers}(h(\vec{x}), \mathcal{K})$, so its violations are the minimal subsets of $\mathcal{K}$ responsible for $\vec{a} \in \text{answers}(b(\vec{x}), \mathcal{K})$. The next proposition relates constraint violations and *justifications*. A justification (also known as an explanation, axiom pinpointing, or MinAs) for the entailment of a BCQ is a minimal subset of the KB that entails the BCQ [21, 34].

Proposition 2: *If $\mathcal{K}$ violates $\Gamma(\vec{a}) : b(\vec{a}) \rightarrow h(\vec{a})$, a subset $\mathcal{V} \subseteq \mathcal{K}$ is a violation of $\Gamma(\vec{a})$ iff $\mathcal{V}$ is a justification of $\mathcal{K} \models b(\vec{a})$.*

5 CORRECTIONS

We now turn to correcting constraint violations.

Solutions. We will make use of atomic modifications of the KB to define solutions to constraint violations.

Definition 3 (Atomic modification): *An* atomic modification *of a KB $\mathcal{K}$ is a pair $m = (\mathcal{M}^+, \mathcal{M}^-)$ of two sets of assertions or $\mathcal{L}$-axioms that takes one of the following forms:*
Addition: *$m = (\{\langle s, p, o\rangle\}, \emptyset)$, where $\langle s, p, o\rangle \notin \mathcal{K}$*
Deletion: *$m = (\emptyset, \{\langle s, p, o\rangle\})$, where $\langle s, p, o\rangle \in \mathcal{K}$*
Replacement: *$m = (\{\langle s, p, o\rangle\}, \{\langle s', p', o'\rangle\})$, where $\langle s, p, o\rangle \notin \mathcal{K}$, $\langle s', p', o'\rangle \in \mathcal{K}$, and $\langle s, p, o\rangle$ differs from $\langle s', p', o'\rangle$ in exactly one component.*

Thus, an atomic modification consists of two sets $\mathcal{M}^+$ and $\mathcal{M}^-$, each of which is either the empty set or a singleton set. $\mathcal{M}^+$ will be added to the KB, and $\mathcal{M}^-$ will be removed from the KB. Since the sets contain at most one triple, we slightly abuse the notation and identify the singletons with their elements (e.g., we will denote the addition of $\langle s, p, o\rangle$ simply by $(\langle s, p, o\rangle, \emptyset)$). A replacement is equivalent to a sequence of a deletion and an addition. We chose to keep it as an atomic modification because it corresponds to common knowledge base curation tasks, such as correcting an erroneous object for a given subject and predicate, or fixing a predicate misuse. Atomic modifications can be used to solve a constraint violation, as follows:

Definition 4 (Solution): *A* solution *to a violation $\mathcal{V}$ of a constraint instance $\Gamma(\vec{a})$ in $\mathcal{K}$ is an atomic modification $(\mathcal{M}^+, \mathcal{M}^-)$ such that there exists $\mathcal{K}' \subseteq \mathcal{K}$ such that $(\mathcal{V} \cup \mathcal{K}' \cup \mathcal{M}^+) \setminus \mathcal{M}^-$ satisfies $\Gamma(\vec{a})$. We call $(\mathcal{M}^+, \mathcal{M}^-)$ a solution to $\mathcal{V}$ for $\Gamma(\vec{a})$ in $\mathcal{K}$.*

Note that $\Gamma(\vec{a})$ can still be violated in $(\mathcal{K} \cup \mathcal{M}^+) \setminus \mathcal{M}^-$ if $\mathcal{K}$ contains other violations of $\Gamma(\vec{a})$ for which $(\mathcal{M}^+, \mathcal{M}^-)$ is not a solution. For example, if $\Gamma(a) : \exists x R(a, x) \wedge A(a) \rightarrow$ false, and $\mathcal{K} = \{R(a, b), R(a, c), A(a)\}$, the deletion of $R(a, b)$ is a solution to the violation $\{R(a, b), A(a)\}$, but $\{R(a, c), A(a)\}$ still violates $\Gamma(a)$. Note also that every constraint violation has at least one solution, which consists of the deletion of any of its elements. Solutions may also be additions or replacements, as in the following example:

Example 5: *In our running example, the deletion* $(\emptyset, \text{hasGender}(\textit{Zeus}, \textit{masculine}))$ *and the replacement* $(\text{hasGender}(\textit{Zeus}, \textit{male}), \text{hasGender}(\textit{Zeus}, \textit{masculine}))$ *are two possible solutions to* $\mathcal{V}_0$ *for* $\Gamma_0(\textit{masculine})$.

The deletion $(\emptyset, \text{hasMother}(\textit{Spinoza}, \textit{Marques}))$ *is a solution to* $\mathcal{V}$ *for the three constraint instances* $\Gamma_2(\textit{Marques})$, $\Gamma_3(\textit{Marques})$ *and* $\Gamma_4(\textit{Spinoza}, \textit{Marques})$. *The additions* $(\text{Human}(\textit{Marques}), \emptyset)$, $(\text{hasGender}(\textit{Marques}, \textit{female}), \emptyset)$ *and* $(\text{hasChild}(\textit{Marques}, \textit{Spinoza}), \emptyset)$ *are solutions to* $\mathcal{V}$ *for respectively* $\Gamma_2(\textit{Marques})$, $\Gamma_3(\textit{Marques})$ *and* $\Gamma_4(\textit{Spinoza}, \textit{Marques})$. ◁

Good solutions. Our goal is to find "good" solutions to constraint violations, i.e., solutions that make the KB as close to the real world as possible. The basic requirement for a "good" solution is that it deletes only erroneous facts, and that it adds only true facts. We also prefer replacements over deletions as long as they fulfill this condition. For instance, in our running example, the replacement $(\text{hasGender}(\textit{Zeus}, \textit{male}), \text{hasGender}(\textit{Zeus}, \textit{masculine}))$ is better than the deletion $(\emptyset, \text{hasGender}(\textit{Zeus}, \textit{masculine}))$, because it corrects erroneous information instead of simply erasing it.

In some cases, there may be no "good" solution that consists of a single atomic modification. Consider for example a completeness constraint of the form $A \sqsubseteq \exists P \cdot B$ violated by $\{A(a)\}$. If $A(a)$ is true, we should actually add both $P(a, b)$ and $B(b)$ for some b. We choose to define solutions as atomic modifications nevertheless to simplify the problem by reducing the size of possible solutions. This is a limitation of our approach since we will not be able to learn solutions that are not atomic. However, we will still be able to learn to add $B(b)$ to solve the aforementioned constraint violation in the case where $P(a, b)$ is already present.

The main difficulty in finding good solutions to constraint violations is that we do not have access to an oracle that knows the validity of all facts. This is the problem that all KB cleaning approaches face (cf. Section 2). Our idea is to exploit the history of the KB modifications to learn how to correct constraint violations.

Definition 5 (Edit history): *The* edit history *of a KB is a sequence of KBs* $(\mathcal{K}_i)_{0 \le i \le p} = (\mathcal{T}_i \cup \mathcal{A}_i)_{0 \le i \le p}$ *such that* $\mathcal{K}_{i+1} = (\mathcal{K}_i \cup \mathcal{M}_i^+) \setminus \mathcal{M}_i^-$, *where* $(\mathcal{M}_i^+, \mathcal{M}_i^-)$ *is an atomic modification.*

The edit history allows us to pinpoint how constraint violations have been corrected in the past. In order to avoid learning from vandalism or mistakes, we consider only those corrections that have not been reversed:

Definition 6 (Past correction): *A* past correction *is a solution* $(\mathcal{M}^+, \mathcal{M}^-)$ *to a violation* $\mathcal{V}$ *of a constraint instance* $\Gamma(\vec{a})$ *in* $\mathcal{K}_i$ *such that there exist* $\mathcal{B}$ *and* $\mathcal{D}$ *such that the current KB* $\mathcal{K}_p = (\mathcal{K}_i \cup \mathcal{B}) \setminus \mathcal{D}$ *with* $\mathcal{M}^+ \subseteq \mathcal{B}$, $\mathcal{M}^- \subseteq \mathcal{D}$, $\mathcal{M}^+ \cap \mathcal{D} = \emptyset$, $\mathcal{M}^- \cap \mathcal{B} = \emptyset$.

Intuitively, $(\mathcal{B}, \mathcal{D})$ corresponds to the sequence of additions and deletions that leads from $\mathcal{K}_i$ to the current state of the KB $\mathcal{K}_p$, that contains the solution, and that does not "undo" it.

Relevant past corrections. During the history of a KB, users can change not just the assertions of the KB, but also the TBox. However, the TBox is typically much smaller and more stable than the ABox. Therefore, the edit history of the TBox is not a rich ground for correction rule mining. Moreover, we are interested in learning solutions that correct constraint violations in the *current* KB $\mathcal{K}_p$. We thus consider only those past corrections that would have been corrections also under the current TBox. For example, assume that the TBox contained $C \sqsubseteq B$. Assume that $C(a)$ was added to correct a violation of the constraint $A \sqsubseteq B$. If, in the meantime, the inclusion $C \sqsubseteq B$ has been removed, we do not want to learn from this past correction. The following definition formalizes these requirements.

Definition 7 (Relevant Past Correction): *A* relevant past correction $(\mathcal{M}^+, \mathcal{M}^-)$ *to a violation* $\mathcal{V}$ *of a constraint instance* $\Gamma(\vec{a})$ *in* $\mathcal{K}_i$ *is a past correction such that (i)* $\mathcal{M}^+ \cup \mathcal{M}^-$ *contains only assertions, and (ii)* $(\mathcal{V} \cap \mathcal{A}_i) \cup \mathcal{T}_p$ *contains a violation* $\mathcal{V}'$ *of* $\Gamma(\vec{a})$ *such that* $(\mathcal{M}^+, \mathcal{M}^-)$ *is also a solution to* $\mathcal{V}'$ *in* $\mathcal{A}_i \cup \mathcal{T}_p$.

We will now see how we can use the relevant past corrections to mine correction rules.

6 FROM HISTORY TO CORRECTION RULES

In this section, we propose an approach based on rule mining to learn *correction rules* for building solutions to constraint violations.

6.1 Extraction of the Relevant Past Corrections

Algorithm 1 constructs the set of relevant past corrections from the KB history. It consists of three main steps. First, it constructs patterns to spot KB modifications that could be part of a relevant past correction. Then it uses these patterns to extract atomic modifications that solved some violation in the past. Finally, the relevant past corrections are obtained by pruning those that have been reversed.

Algorithm 1 Construction of PCDataset

Input: set of constraints C, current TBox $\mathcal{T}_p$, history $(\mathcal{K}_i)_{0 \le i \le p}$
Output: set of relevant past corrections PCDataset

```
// Construct correction seed patterns
for all Γ ∈ C such that Γ(x⃗) : b(x⃗) → h(x⃗) do
    Patterns(Γ) := {(_, A(x⃗)) | A(x⃗) ∈ b'(x⃗), b'(x⃗) ∈ rewrite(b(x⃗), T_p)}
    if Γ is a completeness constraint then
        Patterns(Γ) ∪= ({(A(x⃗), _) | A(x⃗) ∈ h'(x⃗),
                                      h'(x⃗) ∈ rewrite(h(x⃗), T_p)})
// Extract past corrections
for 0 ≤ i ≤ p − 1 do
    if (M_i^+, M_i^-) such that K_{i+1} = (K_i ∪ M_i^+) \ M_i^- matches
    some pattern in Patterns(Γ) then
        PCDataset ∪= {⟨(M_i^+, M_i^-), Γ(a⃗), V, i⟩ |
                V ∈ Violations(K_i, Γ(a⃗)) \ Violations(K_{i+1}, Γ(a⃗))}
// Remove reversed past corrections
for ⟨(M_i^+, M_i^-), Γ(a⃗), V, i⟩ ∈ PCDataset do
    if M_i^+ ⊈ K_p or M_i^- ∩ K_p ≠ ∅ then
        PCDataset \= {⟨(M_i^+, M_i^-), Γ(a⃗), V, i⟩}
```

Let us explain our algorithm with our running example. Consider the constraint $\Gamma_0(x) : \exists y \text{hasGender}(y, x) \rightarrow x = \textit{male} \vee x = \textit{female} \vee x = \textit{nonbinary}$. Assume that $\langle \textit{Zeus}, \text{hasGender}, \textit{masculine} \rangle$ was added between $\mathcal{K}_1$ and $\mathcal{K}_2$, but then replaced by $\langle \textit{Zeus}, \text{hasGender}, \textit{male} \rangle$ between $\mathcal{K}_{100}$ and $\mathcal{K}_{101}$.

The first goal of the algorithm is to find out that the removal of $\langle \textit{Zeus}, \text{hasGender}, \textit{masculine} \rangle$ between $\mathcal{K}_{100}$ and $\mathcal{K}_{101}$ (as part of the replacement) may be part of a relevant past correction. We

Table 2: Dataset PCDataset **of relevant past corrections extracted for our running example.**

Relevant past correction	Constraint instance	Violation	KB index
$(\{\langle Zeus, \text{hasGender}, male\rangle\}, \{\langle Zeus, \text{hasGender}, masculine\rangle\})$	$\Gamma_0(masculine)$	$\{\langle Zeus, \text{hasGender}, masculine\rangle\}$	100

call this deletion a *correction seed*. Formally, a correction seed is a deletion $(\emptyset, \mathcal{M}^-)$ or an addition $(\mathcal{M}^+, \emptyset)$ such that (i) there exists $0 \leq i \leq p-1$ such that $\mathcal{K}_{i+1} = (\mathcal{K}_i \cup \mathcal{M}_i^+) \setminus \mathcal{M}_i^-$ with $\mathcal{M}_i^- = \mathcal{M}^-$ (resp. $\mathcal{M}_i^+ = \mathcal{M}^+$) and (ii) there exists a KB $\mathcal{T}_p \cup \mathcal{D}$, where $\mathcal{D}$ is a set of assertions, such that $\mathcal{T}_p \cup \mathcal{D}$ contains a violation $\mathcal{V}$ of some constraint instance and $(\emptyset, \mathcal{M}^-)$ (resp. $(\mathcal{M}^+, \emptyset)$) is a solution to $\mathcal{V}$. Looking for correction seeds instead of computing the constraint violations for all constraints on all KB versions has the advantage of significantly reducing the search space.

To find such correction seeds efficiently, the first step of the algorithm precomputes for each constraint a set of atomic modification patterns that the possible correction seeds would match. In the example there would be only one pattern: the deletion pattern $(_, \langle ?, \text{hasGender}, ?\rangle)$, where _ can be anything so that it matches both the deletion of $\langle ?, \text{hasGender}, ?\rangle$ and its replacements. Since we only consider past corrections that involve assertions, and want them to be relevant for the current TBox, computing the correction seed patterns can be done via query rewriting of the CQs in the body $b(\vec{x})$ and the head $h(\vec{x})$ of the constraint w.r.t. $\mathcal{T}_p$. Indeed, if $\mathcal{T}$ is a flat QL TBox, any CQ $q(\vec{x})$ can be rewritten w.r.t. $\mathcal{T}$ into a UCQ $q'(\vec{x})$ such that for every ABox $\mathcal{A}$, answering $q(\vec{x})$ over $\mathcal{T} \cup \mathcal{A}$ amounts to answering $q'(\vec{x})$ over $\mathcal{A}$ [23]. Each atom that occurs in the rewriting of the body of a constraint corresponds to a deletion pattern, and each atom that occurs in the rewriting of the head of a completeness constraint corresponds to an addition pattern. We collect the patterns for the constraint Γ in the set Patterns(Γ). Note that it is not possible to solve a consistency constraint with an addition, which is why such constraints have only deletion patterns.

The second step of the algorithm verifies, for each correction seed, whether it solved some constraint violation in the past – i.e., whether $\mathcal{K}_i$ contains some violations of some constraint instances that are not in $\mathcal{K}_{i+1}$. If so, the modification between $\mathcal{K}_i$ and $\mathcal{K}_{i+1}$ is a solution that solved these violations in $\mathcal{K}_i$. In the example we would have found the violation $\{\langle Zeus, \text{hasGender}, masculine\rangle\}$ of $\Gamma_0(masculine)$ in $\mathcal{K}_{100}$, which is not in $\mathcal{K}_{101}$. So we would have extracted that $(\langle Zeus, \text{hasGender}, male\rangle, \langle Zeus, \text{hasGender}, masculine\rangle)$ is a solution that solved the violation $\{\langle Zeus, \text{hasGender}, masculine\rangle\}$ of $\Gamma_0(masculine)$ in $\mathcal{K}_{100}$. We store this information as a tuple in the relevant past corrections dataset (the PCDataset), as shown in Table 2. Finding the constraint instances violated in $\mathcal{K}_i$ or $\mathcal{K}_{i+1}$ is done via CQ answering (Proposition 1), and computing their violations amounts to computing BCQ justifications (Proposition 2).

The final step of the algorithm removes corrections that have been reversed. The result is thus the set of relevant past corrections.

6.2 Correction Rule Mining

Correction rules. The previous algorithm has given us a list of relevant past corrections (the PCDataset, exemplified in Table 2). We now present our approach to mine *correction rules* from this dataset and the KB history.

Definition 8 (Correction rule): *A* correction rule *is of the form*

$$r := [\Gamma(\vec{x})] : \mathcal{E}(\vec{x}, \vec{y}, \vec{z}) \rightarrow (\mathcal{M}^+(\vec{x}, \vec{y}), \mathcal{M}^-(\vec{x}, \vec{y})), \text{ where}$$

- $\Gamma(\vec{x})$ *is a constraint that can be partially instantiated, i.e., some of its variables have been replaced by constants,*
- $(\mathcal{M}^+(\vec{x}, \vec{y}), \mathcal{M}^-(\vec{x}, \vec{y}))$ *is a pair of sets of at most one triple,*
- $\mathcal{E}(\vec{x}, \vec{y}, \vec{z})$ *is a set of atoms called the* context *of the violation such that* $\mathcal{M}^-(\vec{x}, \vec{y}) \subseteq \mathcal{E}(\vec{x}, \vec{y}, \vec{z})$,

and both $(\mathcal{M}^+(\vec{x}, \vec{y}), \mathcal{M}^-(\vec{x}, \vec{y}))$ *and* $\mathcal{E}(\vec{x}, \vec{y}, \vec{z})$ *are built from* $\mathsf{N_C} \cup \mathsf{N_R} \cup \{\texttt{rdf:type}\} \cup \mathsf{N_I} \cup \vec{x} \cup \vec{y} \cup \vec{z}$.

A correction rule can be applied *to a KB* $\mathcal{K}$ *when there exist tuples of constants* $\vec{a}, \vec{b}$ *such that* $\mathcal{K}$ *violates* $\Gamma(\vec{a})$ *(recall that this can be decided via CQ answering by Proposition 1) and* $\mathcal{K} \models \exists \vec{z} \mathcal{E}(\vec{a}, \vec{b}, \vec{z})$. *The* result of the rule application *is then* $(\mathcal{M}^+(\vec{a}, \vec{b}), \mathcal{M}^-(\vec{a}, \vec{b}))$.

Note that while the variables from $\mathcal{E}(\vec{x}, \vec{y}, \vec{z})$ that do not appear in $\Gamma(\vec{x})$ or in the head of r can be existentially quantified, those that occur in the head of r have to be free: they have to be mapped to individuals occurring in the KB in order to construct the result.

Example 6: *In our running example, we would like to learn the following correction rules:*

$$r_1 := [\Gamma_0(masculine)] : \{\text{hasGender}(y, masculine)\} \rightarrow (\text{hasGender}(y, male), \text{hasGender}(y, masculine))$$

$$r_2 := [\Gamma_2(x)] : \{\text{hasMother}(y, x), \text{Human}(y)\} \rightarrow (\text{Human}(x), \emptyset)$$

The context of the second rule says that if x *is the mother of a human, then* x *must also be a human. The rule obtained by replacing* Human *by* Animal *would express how to solve a violation of* Γ_2 *in the context where* y *is an animal.* ◁

Mining correction rules. We mine correction rules with Algorithm 2. This algorithm is an adaptation of the algorithm in [13, 14] to our context, where we learn rules not from a KB but from the PCDataset and the KB history. We first adapt the definitions of the confidence and support from [13, 14] to our case. The *support of the body* of a correction rule r for a constraint Γ is the number of violations of Γ stored in the PCDataset that could have been corrected by applying r. Such violations are associated with an instance $\Gamma(\vec{a})$ of the partially instantiated $\Gamma(\vec{x})$ that appears in r and with an index i such that $\mathcal{K}_i \models \exists \vec{z} \mathcal{E}(\vec{a}, \vec{b}, \vec{z})$ for some $\vec{b}$. These two conditions imply that r could be applied to the KB $\mathcal{K}_i$. Moreover, we need to check that the result of applying r to $\mathcal{K}_i$ actually gives a solution to $\mathcal{V}$. For example, consider the case where PCDataset contains both $\langle(\emptyset, R(a,b))), \Gamma(a), \{R(a,b), A(a)\}, i\rangle$ and $\langle(\emptyset, R(a,c))), \Gamma(a), \{R(a,c), A(a)\}, j\rangle$ for $\Gamma(a) : \exists x R(a,x) \wedge A(a) \rightarrow \text{false}$. Both violations count for the support of the body of $[\Gamma(a)] : R(a,x) \rightarrow (\emptyset, R(a,x))$ but only the second one counts for the support of the body of $[\Gamma(a)] : R(a,c) \rightarrow (\emptyset, R(a,c))$, even in the case

where $\mathcal{K}_i \models R(a, c)$. Formally, $sup_{bod}(r) = |BSup|$ where

$$BSup = \{\mathcal{V} \mid \langle _, \Gamma(\vec{a}), \mathcal{V}, i\rangle \in \text{PCDataset}, \exists \vec{b}\, \mathcal{K}_i \models \exists \vec{z} \mathcal{E}(\vec{a}, \vec{b}, \vec{z})$$
and the result of the application of r to $\mathcal{K}_i$ is a solution to $\mathcal{V}\}$.

The *support of the rule r* measures when the past correction is exactly the result of the application of the rule in the cases where it could be applied. Formally, $sup_{rule}(r) = |RSup|$, where

$$RSup = \{\mathcal{V} \mid \langle (\mathcal{M}^+(\vec{a}, \vec{b}), \mathcal{M}^-(\vec{a}, \vec{b})), \Gamma(\vec{a}), \mathcal{V}, i\rangle \in \text{PCDataset},$$
$$\text{and } \mathcal{K}_i \models \exists \vec{z} \mathcal{E}(\vec{a}, \vec{b}, \vec{z})\}.$$

Finally, the *confidence of a correction rule r* is $conf(r) = \frac{sup_{rule}(r)}{sup_{bod}(r)}$.

Algorithm 2 Correction rule mining

Input: PCDataset, $(\mathcal{K}_i)_{0 \le i \le p}$, *minsup*, *minconf*, θ
Output: correction rules

```
// Generate basic rules
BasicR := ∅
for all ⟨(M⁺(a⃗, b⃗), M⁻(a⃗, b⃗)), Γ(a⃗), V, i⟩ ∈ PCDataset do
  r0 := [Γ(a⃗)] : M⁻(a⃗, b⃗) → (M⁺(a⃗, b⃗), M⁻(a⃗, b⃗))
  BasicR ∪= {σ(r0) | C ⊆ a⃗ ∪ b⃗, σ : C ↣ Var,
              sup_rule(σ(r0)) ≥ minsup, conf(σ(r0)) ≥ minconf}
// Refine the context part of the rules
q := [], q.enqueueAll(BasicR)
while q is not empty do
  r := q.dequeue()
  Output r
  for all operators op do
    for all r' ∈ op(r) do
      if sup_rule(r') ≥ minsup and conf(r') ≥ conf(r) + θ then
        q.enqueue(r')
```

Algorithm 2 shows our mining algorithm. It takes as input the PCDataset computed by Algorithm 1, the KB history, a minimum support threshold, a minimum confidence threshold, and a regularization threshold θ. These thresholds are chosen empirically (see Section 7.3). The algorithm produces correction rules (Definition 8). For this purpose, it first generates a trivial rule r_0 for each entry of the PCDataset. This rule has as context simply the deletion part of the constraint past correction. This trivial rule is then transformed into several more general rules, which we call *basic rules*, each of which is obtained from r_0 by replacing some of the constants by variables. Formally, the algorithm uses all partial substitutions σ from constants to distinct fresh variables. It retains only those basic rules that meet the minimum support and confidence thresholds.

In the second step, the algorithm incrementally refines each rule by building up its context part $\mathcal{E}(\vec{x}, \vec{y}, \vec{z})$. This works similarly to the mining algorithm of [14]: Each refinement step adds one atom built from the KB concept and role names and the variables and constants that appear in the rule, plus at most one fresh variable. For this purpose, the algorithm uses the operators defined in [14]. If the resulting rule meets the minimum support threshold, and improves the confidence by at least θ, the rule is retained.

Note that Algorithm 2 outputs only rules that would have been returned by the algorithm of [13, 14] if evaluated with our confidence function. It prunes more rules because of the use of the θ and the *minconf* thresholds that are also used to do an early pruning of the rules during the context construction. Algorithm 2 can be easily parallelized by running it independently on each constraint and/or having multiple workers working on the same queue.

Applying correction rules. When all rules have been mined, they are sorted by decreasing confidence, breaking ties by help of the support (as it is done in [26] to build classifiers from rules). This set of rules then forms a program that can be used to fix constraint violations as follows. Given a violation $\mathcal{V}$ of a constraint Γ in $\mathcal{K}$, choose the first rule r in the program that is relevant for Γ (i.e., that contains $[\Gamma(\vec{x})]$ where $\Gamma(\vec{x})$ is a partially instantiated version of Γ). Then check whether r can be applied to $\mathcal{V}$. The correction is the result of the rule application.

Example 7: *Assume we mined the rules r_1 and r_2 of the preceding example with confidence 0.9 and 0.8 respectively, and another rule $r_3 := [\Gamma_0(x)] : \{\text{hasGender}(x, y)\} \to (\emptyset, \text{hasGender}(x, y))$ with confidence 0.5. The correction program is (r_1, r_2, r_3). To correct a violation of Γ_0, i.e. a wrong value for the* hasGender *property, the program first checks whether r_1 is applicable. If so, it replaces masculine by male. Otherwise, it falls back to r_3 and removes the wrong value. To correct a violation of Γ_2, it ignores r_1 that is not related to Γ_2 and either applies r_2 if the context matches or does nothing.* ◁

7 EXPERIMENTS ON WIKIDATA

This section describes CorHist, which implements the framework introduced for Wikidata, and presents its experimental evaluation.

7.1 Wikidata

Wikidata is a generalist collaborative knowledge base. The project started in 2012, and as of July 2018, it has collected more than 500M statements about 50M entities. The data about each entity is stored in a versioned JSON blob, and there are more than 700M revisions. Wikidata encodes facts not in plain RDF triples but in a reified representation, in which each main $\langle s, p, o\rangle$ triple can be annotated with qualifiers and provenance information [38].

Wikidata knows the property instanceOf which is similar to `rdf:type`. It does not have a formally defined TBox, but knows properties such as subClassOf, subPropertyOf, and inverseOf. However, only the property subClassOf is used to flag the constraint violations. Therefore, we use only this property in our TBox, which thus contains simple concept inclusions.

We consider the set C of constraints built from ten types of Wikidata property constraints (see Table 3). They are the top Wikidata property constraints that can be expressed in DL, covering the majority of the most used constraints, as well as 71% of Wikidata property constraints. The remaining constraints are mainly about string format validation with regular expressions (52% of the remaining constraints) and about qualifiers (31% of them).

7.2 Dataset Construction

We stored the RDF version [10, 18] of the Wikidata edit history in an RDF quad store. We used named graphs for the global state of Wikidata after each revision, and for the triple additions and deletions. Our dataset stores 390M annotated triples about 49M items extracted from the July 1st, 2018 full database dump.

Table 3: Wikidata property constraints. R is the property for which the constraint is given. A constraint has several lines when it uses a property whose set of values may be specified or not. ♯constr. is the total number of constraints of the given type in Wikidata. ♯triples is the sum for all these constraints of the numbers of triples with the property R on which they apply. ♯violations is the number of violations for this constraint in Wikidata on July 1st, 2018. ♯past cor. is the number of past corrections we extracted from Wikidata history. t.o. indicates that we were not able to extract all past corrections because of timeout so that we sample them (we then indicate the number of corrections we extracted).

Name in Wikidata	DL form	Rule form	♯constr.	♯triples	♯violations	♯past cor.
Type [4]	$\exists R \sqsubseteq A_1 \sqcup \cdots \sqcup A_n$	$\exists y R(x,y) \rightarrow A_1(x) \vee \cdots \vee A_n(x)$	2575	249M	3465k	t.o.(>16M)
Value type [4]	$\exists R^- \sqsubseteq A_1 \sqcup \cdots \sqcup A_n$	$\exists y R(y,x) \rightarrow A_1(x) \vee \cdots \vee A_n(x)$	696	67M	3062k	t.o.(>19M)
One-of	$\exists R^- \sqsubseteq \{a_1, \ldots, a_n\}$	$\exists y R(y,x) \rightarrow x = a_1 \vee \cdots \vee x = a_n$	104	3.6M	4k	14k
Item requires	$\exists R \sqsubseteq \exists R' \cdot \{a_1, \ldots, a_n\}$	$\exists y R(x,y) \rightarrow R'(x,a_1) \vee \cdots \vee R'(x,a_n)$	3102	255M	3710k	t.o.(>15M)
statement	$\exists R \sqsubseteq \exists R'$	$\exists y R(x,y) \rightarrow \exists z R'(x,z)$				
Value requires	$\exists R^- \sqsubseteq \exists R' \cdot \{a_1, \ldots, a_n\}$	$\exists y R(y,x) \rightarrow R'(x,a_1) \vee \cdots \vee R'(x,a_n)$	243	85M	1345k	t.o.(>6M)
statement	$\exists R^- \sqsubseteq \exists R'$	$\exists y R(y,x) \rightarrow \exists z R'(x,z)$				
Conflict with	$\exists R \sqsubseteq \neg\exists R' \cdot \{a_1, \ldots, a_n\}$	$\exists y R(x,y) \wedge (R'(x,a_1) \vee \cdots \vee R'(x,a_n)) \rightarrow \text{false}$	601	449M	142k	465k
	$\exists R \sqsubseteq \neg\exists R'$	$\exists y z R(x,y) \wedge R'(x,z) \rightarrow \text{false}$				
Inverse/Symmetric[5]	$R \sqsubseteq R'^-$	$R(x,y) \rightarrow R'(y,x)$	146	6M	409k	2989k
Single value	$(\text{func } R)$	$R(x,y) \wedge R(x,z) \rightarrow y = z$	2772	85M	334k	389k
Distinct values	$(\text{func } R^-)$	$R(y,x) \wedge R(z,x) \rightarrow y = z$	2728	56M	189k	7432k

We extracted the relevant past corrections as explained in Section 6.1. Wikidata revisions do not correspond exactly to atomic modifications in our sense. For example, Wikidata bots are able to change multiple unrelated facts about the same entity at the same time. Wikidata users also sometimes prefer to delete a statement then add another one with the same property instead of directly modifying the value, in order to clear the existing qualifiers and references. Therefore, we artificially created a replacement modification for every deletion with a neighboring addition by the same user, which shares at least two components of the triple (analogously for additions). For example, if the correction seed is the deletion of ⟨*Zeus*, hasGender, *masculine*⟩, and if this revision or a neighboring one adds ⟨*Zeus*, hasGender, *male*⟩, then we consider this a replacement. However, if the same revision added the triple ⟨*Zeus*, hasMother, *Rhea*⟩, then we would not consider this a replacement, because it does not share two components with the first one.

Since the TBox consists of simple concept inclusions and the constraint bodies contain only roles, the deletion patterns for correction seeds correspond directly to the atoms of the constraint body. In the same vein, only atoms in the head of the *Type* or *Value type* constraints need to be rewritten. To find the constraint violations solved by a correction seed, we make use of the fact that the correction seed allows us to know the constraint instance $\Gamma(\vec{a})$, and we look for matches of the constraint instance body.

To speed up the execution for the four constraint types which have the highest numbers of past corrections, *Type*, *Value type*, *Item requires statement* and *Value requires statement*, we did not extract all the past corrections but sample them as follows. We compute only the relevant past corrections that where applied between $\mathcal{K}_i$ and $\mathcal{K}_{i+1}$ where i is a multiple of $s := max(1, N/10^6)$ with N the number of triples with the property R of the constraint at hand. This sampling allows us to get a sufficient ground for rule mining for each constraint. In practice, it affects only the most frequent 0.9% of *Type*, 2% of *Value type*, 0.5% of *Item requires statement*, and 3% of *Value requires statement* constraints.

7.3 Mining Rules

The output of our method is a set of correction rules that form a program (Section 6.2). To evaluate such a program, we apply it to each of the constraint violations stored in the PCDataset, using the associated stage of the KB to evaluate the part of the context which is not the deletion part of the correction. Then we check whether the correction we compute is exactly the same as the one associated to the constraint violation in the PCDataset. The *precision* p of the program is given by the fraction of the corrections computed by the program that are actually the same as those that have been applied. The *recall* r of the program is the fraction of the constraint violations stored in PCDataset for which the program gives some correction. The *F1 score* is $F_1 = 2\frac{p \cdot r}{p+r}$.

CorHist mines rules as explained in Section 6.2. In order to decrease the computation time, we only allow one atom $p(s,o)$ in $\mathcal{E}(\vec{x},\vec{y},\vec{z}) \setminus Body(\vec{x},\vec{y})$, where $Body(\vec{x},\vec{y})$ corresponds to the part of the context that matches part of the constraint body, such that s is a variable of $\vec{x} \cup \vec{y}$ and o is a fresh variable or a constant.

Rules were mined per constraint. For each constraint, we split the set of extracted past corrections into a 70% training set, a 10% cross-validation set, and a 20% test set. The training set is used to mine the rules, the cross-validation set is used to determine the confidence threshold that maximizes the F1 score of the obtained program, and the test set is used to evaluate the final program.

[4] The Wikidata constraint *Type* can be qualified to modify its meaning. We ignore these cases, which are marginal: they concern less than 6% of the *Type* constraints. The same goes analogously for *Value type*.

[5] *Inverse* and *Symetric* are two distinct kinds of constraints in Wikidata but we treat them together since *Symetric* is actually a special case of *Inverse*.

Table 4: Example of mined rules.

Constr. type	Constraint Γ	Correction rule
Type	$\exists$isAListOf $\sqsubseteq$ List	$[\Gamma(s)]$: isAListOf(s, o) $\wedge$ WikiDisambiguationPage(s) $\rightarrow$ ($\emptyset$, isAListOf(s, o))
Value type	$\exists$foundInTax$^-$ $\sqsubseteq$ Taxon	$[\Gamma(human)]$: foundInTax(s, *human*) $\wedge$ hasPart(s, v) $\rightarrow$ (foundInTax(s, *homoSapiens*), foundInTax(s, $human$))
One-of	$\exists$mannerDeath$^-$ $\sqsubseteq$ {. . . }	$[\Gamma(trafficAcc)]$: mannerDeath(s, *trafficAcc*) $\rightarrow$ (causeDeath(s, *trafficAcc*), mannerDeath(s, *trafficAcc*))
Item req. stm.	$\exists$heritageStatus $\sqsubseteq$ $\exists$country	$[\Gamma(s)]$: heritageStatus(s, *monumentInFornminnesregistret*) $\rightarrow$ (country(s, *Sweden*), $\emptyset$)
Val. req. stm.	$\exists$residence$^-$ $\sqsubseteq$ $\exists$country	$[\Gamma(s)]$: diplomaticRelation(s, v) $\rightarrow$ (country(s, s), $\emptyset$)
Conflict	$\exists$filmplID $\sqsubseteq$ $\neg\exists$filmplFilmID	$[\Gamma(s)]$: filmplID(s, o) $\rightarrow$ ($\emptyset$, filmplID(s, o))
Inv./Sym.	geneticAssoc $\sqsubseteq$ geneticAssoc$^-$	$[\Gamma(s, o)]$:$\rightarrow$ (geneticAssoc(o, s), $\emptyset$)
Single val.	(func sexOrGender)	$[\Gamma(s)]$: sexOrGender(s, *maleOrg*) $\wedge$ sportsTeam(s, v) $\rightarrow$ ($\emptyset$, sexOrGender(s, *maleOrg*))
Distinct val.	(func ncbiLocusTag$^-$)	$[\Gamma(s)]$: ncbiLocusTag(o, s) $\wedge$ molecularFunction(o, v) $\rightarrow$ ($\emptyset$, ncbiLocusTag(o, s))

Table 4 gives examples of rules mined by CorHist. Several of these rules show the crucial importance of the instantiation of the constraint and/or of the context to be able to choose the correction. For instance, the rule for the *Single value* constraint uses the fact that an entity involved in a property "member of sport team" is probably a human being, and thus that if it has several values for the functional property "sex or gender" and one of them is a value reserved for non-human organisms in Wikidata, this value is probably wrong. In the same vein, the rule for the *Item requires statement* constraint recognizes that an entity has a heritage designation that is specific to Sweden ("monument in Fornminnesregistret"), to conclude that its country is Sweden. The rules also propose fixes to misused predicates: in Wikidata, the property "manner of death" is intended for the general circumstances of a person's death (such as "accident"), while the property "cause of death" is intended to give more precise causes (such as "traffic accident").

7.4 Evaluation against the Test Set

Table 5 presents the results of the evaluation of the mined programs against the test set. We computed both the micro and macro average of the precision, recall and F1 score per kind of constraint. The micro average aggregates over the whole set of relevant past corrections for the given kind of constraint, whereas the macro average computes the scores for each constraint of this given kind, and then computes the average. Both numbers are important: The micro average gives more weight to correction rules that fix many violations. It thus measures the overall impact of the correction rules on the dataset. However, if few rules had a large impact, then it would be easier to formulate these rules by hand. Our method, in contrast, can also find rules that by themselves solve less violations, but together contribute a large mass of corrections. To illustrate this, we also report the macro average: It measures the average performance across different constraints.

We compare our approach with two baselines: The first one, called "delete", is the most basic one and uses the fact that all Wikidata constraint bodies contain an atom of the form $R(x, y)$ and the TBox contains only concept inclusions, so that all constraint violations contain an assertion that matches $R(x, y)$. The "delete" baseline simply deletes this assertion. For the completeness constraints we define an additional baseline, "add", which tries to add a new triple to solve the constraint violation. For *Inverse* and *Symmetric* constraints this baseline adds the missing reverse edge and performs very well. For *Item requires statement*, *Value requires statement*, *Type* and *Value type*, it adds the missing triple only if it is possible to know the expected value from the constraint rule. For example, for *Type* constraints of the form $\exists y R(x, y) \rightarrow A(x)$ it applies the correction $(A(x), \emptyset)$. *Value type* constraints are handled in the same way. However, this baseline is not able to figure out what is the relevant addition correction for a constraint of the form $\exists y R(x, y) \rightarrow A_1(x) \vee A_2(x)$ because there is no way to know a priori if A_1 or A_2 should be added. For *Item requires statement*

Table 5: Evaluation of the correction rules mined by CorHist with a minimal support of 10, a minimal confidence between 0.5 and 1 and a regularization threshold of 0.05, and comparison with the baselines. Best precision and F1 scores in bold.

Constraint type		Micro average Prec.	Rec.	F1	Macro average Prec.	Rec.	F1
Type[6]	add	0.35	0.33	0.34	0.29	0.55	0.38
	delete	0.04	1	0.07	0.13	1	0.23
	CorHist	**0.84**	0.70	**0.76**	**0.91**	0.39	**0.55**
Value type[6]	add	0.09	0.21	0.13	0.33	0.58	0.42
	delete	0.01	1	0.02	0.16	1	0.27
	CorHist	**0.81**	0.61	**0.70**	**0.89**	0.51	**0.65**
One-of	delete	0.26	1	0.42	0.35	1	0.52
	CorHist	**0.77**	0.83	**0.80**	**0.95**	0.37	**0.53**
Item requires statement[6]	add	**0.99**	0.14	0.25	0.80	0.27	0.40
	delete	0.017	1	0.033	0.10	1	0.19
	CorHist	0.94	0.35	**0.51**	**0.95**	0.32	**0.47**
Value requires statement[6]	add	0.44	20e-6	41e-6	0.33	0.12	0.18
	delete	0.041	1	0.079	0.082	1	0.15
	CorHist	**0.91**	0.53	**0.67**	**0.94**	0.37	**0.53**
Conflict with	delete	0.39	1	0.56	0.39	1	0.56
	CorHist	**0.92**	0.55	**0.69**	**0.91**	0.41	**0.57**
Inverse/Sym.	add	0.91	1	0.95	0.77	1	0.86
	delete	0.072	1	0.12	0.14	1	0.24
	CorHist	**0.92**	1[7]	**0.96**	**0.90**	0.84	**0.87**
Single value	delete	0.34	1	**0.51**	0.42	1	**0.59**
	CorHist	**0.95**	0.093	0.17	**0.96**	0.078	0.14
Distinct values	delete	0.036	1	**0.070**	0.42	1	**0.59**
	CorHist	**0.99**	0.020	0.039	**0.93**	0.12	0.21

[6]Computed from a sample of the set of relevant past corrections. One *Type* constraint, six *Value type* constraints and one *Val. req. stm* constraint were omitted due to time-out.
[7]The actual value is greater than 0.995 and rounded to 1 for consistency.

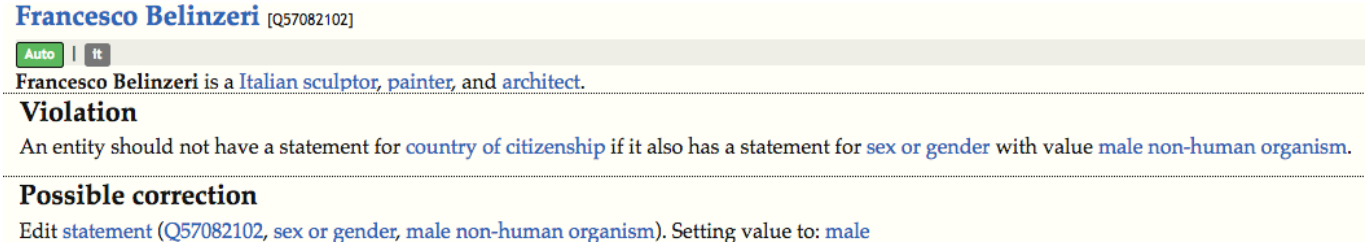


Figure 1: Example of a replacement correction suggested by CorHist for a violation of the constraint $\exists \text{countryOfCitizenship} \sqsubseteq \neg\exists \text{sexOrGender} \cdot \{maleOrg, femaleOrg\}$.

Table 6: Human evaluation of the suggested corrections.

Constraint type	Suggested	"Apply"	"Wrong"	Approval
Type	9908	252	312	0.45
Value type	2374	195	208	0.48
One-of	239	14	47	0.23
Item requires stm.	41	8	32	0.2
Value requires stm.	1024	790	178	0.82
Conflict with	3254	1717	203	0.89
Inverse/Symmetric	28138	20247	1720	0.92
Single value	3264	41	71	0.37
Distinct values	921	8	23	0.26

constraints of the form $\exists y R(x, y) \rightarrow R'(x, a)$, the "add" baseline applies $(R'(x, a), \emptyset)$ (similarly for *Value requires statement* constraints). However, it cannot find a correction if there are multiple a_i.

As shown in Table 5, the precision of our approach significantly outperforms the two baselines – often by a very high margin. Regarding the recall, we manage to keep a reasonable, and sometimes even good, recall (see best F1 scores in Table 5), except for *Single* and *Distinct value*. The very low recall obtained for these two kinds of constraints is easily explainable because they are mostly used on predicates that link Wikidata to other databases (91% of the *Single* and 95% of the *Distinct* constraints), and we cannot get meaningful information about the target database to mine corrections.

7.5 User Evaluation

To see whether our corrections are accepted by the community, we designed a user study. We created a tool that suggests our corrections to Wikidata users for validation (available at https://tools.wmflabs.org/wikidata-game/distributed/#game=43). The user can choose a constraint type, and the tool then suggests corrections for random violations of constraints of this type (Figure 1). The violations for which corrections are suggested are provided by `query.wikidata.org`, which limits their number for performance reasons. For each proposed correction, the user has to choose between three options: apply the proposed correction to Wikidata, tag it as wrong, or get another correction to review. We ran the experiment for 3 months and 47 Wikidata users participated.

Table 6 presents the results. The number of corrections reviewed is highly unbalanced between the kinds of constraints, mainly because a few users evaluate a lot of suggestions, and have a predilection for some kinds of constraints. It is thus difficult to draw conclusions for those kinds of constraints for which very few corrections have been evaluated. However, we can still make some interesting observations. In particular, the proposed corrections marked as wrong give us insights about possible weaknesses of our approach.

For the constraints which got a significant number of evaluations, our approach seems to perform well for *Inverse* and *Symmetric*, *Conflict with* and *Value requires statement* constraints, with approval rates above 80%. The other approval rates are lower. This is partly due to biases in the data. For example, when a gender is missing, our approach proposes the value "male" by default, because of the over-representation of men in Wikidata. Another issue is the quality of the constraints, which in Wikidata are sometimes questionable or difficult to understand (e.g., an incomplete set of possible types or values for completeness or *One-of* constraints).

However, even lower approval scores do not mean that our approach would be useless: Psychological research [12] shows that people find it much easier to choose from given options than to come up with an answer by themselves. The actual time needed to come up with an answer may vary, but if it takes just 3 times longer to come up with an answer than to accept or reject our proposed correction, then achieving a precision of 40% is already useful: If we have a precision of 40%, and if a free-form answer takes time t, then the expected answer time with our tool is $40\% \times \frac{1}{3} \times t + 60\% \times \frac{4}{3} \times t < t$.

8 CONCLUSION AND FUTURE WORK

We have introduced the problem of learning how to fix constraint violations from a KB history. We have also presented a method based on rule mining to this end. Our experimental evaluation on Wikidata shows significant improvement over baselines. Our tool is live on Wikidata and has already allowed users to correct more than 23k constraint violations. While our evaluation focused on Wikidata for which the whole edit history was available, we believe that our method can be applied in other settings, for example using edits done during the partial cleaning of an automatically extracted KB.

For future work, it would be interesting to evaluate the impact of parameters such as the size of the context part of the correction rule in terms of rule quality. We also plan to extend the learning dataset with external knowledge (such as other KBs), or with information extracted from other sources (for instance from Wikipedia). We believe that this will allow finding even more precise correction rules, thus making KBs ever more precise and more useful.

ACKNOWLEDGMENTS

Partially supported by the grant ANR-16-CE23-0007-01 ("DICOS").

REFERENCES

[1] Maribel Acosta, Amrapali Zaveri, Elena Simperl, Dimitris Kontokostas, Fabian Flöck, and Jens Lehmann. 2018. Detecting Linked Data quality issues via crowdsourcing: A DBpedia study. *Semantic Web* 9, 3 (2018), 303–335. https://doi.org/10.3233/SW-160239

[2] Abdallah Arioua and Angela Bonifati. 2018. User-guided Repairing of Inconsistent Knowledge Bases. In *Proceedings of the 21th International Conference on Extending Database Technology, EDBT 2018, Vienna, Austria, March 26-29, 2018.* 133–144. https://doi.org/10.5441/002/edbt.2018.13

[3] Ahmad Assadi, Tova Milo, and Slava Novgorodov. 2018. Cleaning Data with Constraints and Experts. In *Proceedings of the 21st International Workshop on the Web and Databases, Houston, TX, USA, June 10, 2018.* 1:1–1:6. https://doi.org/10.1145/3201463.3201464

[4] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider (Eds.). 2003. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press.

[5] Moria Bergman, Tova Milo, Slava Novgorodov, and Wang-Chiew Tan. 2015. QOCO: A Query Oriented Data Cleaning System with Oracles. *PVLDB* 8, 12 (2015), 1900–1903. https://doi.org/10.14778/2824032.2824096

[6] Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. 2016. Query-Driven Repairing of Inconsistent DL-Lite Knowledge Bases. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016.* 957–964.

[7] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia - A crystallization point for the Web of Data. *Journal of Web Semantics* 7, 3 (2009), 154–165. https://doi.org/10.1016/j.websem.2009.07.002

[8] Iovka Boneva, José Emilio Labra Gayo, and Eric G. Prud'hommeaux. 2017. Semantics and Validation of Shapes Schemas for RDF. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I.* 104–120. https://doi.org/10.1007/978-3-319-68288-4_7

[9] Richard Cyganiak, David Wood, and Markus Lanthaler. 2014. RDF 1.1 Concepts and Abstract Syntax. http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/

[10] Fredo Erxleben, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandečić. 2014. Introducing Wikidata to the Linked Data Web. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I.* 50–65. https://doi.org/10.1007/978-3-319-11964-9_4

[11] Sergio Flesca, Sergio Greco, and Ester Zumpano. 2004. Active integrity constraints. In *Proceedings of the 6th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming, 24-26 August 2004, Verona, Italy.* 98–107. https://doi.org/10.1145/1013963.1013977

[12] Steven C Funk and K Laurie Dickson. 2011. Multiple-choice and short-answer exam performance in a college classroom. *Teaching of Psychology* 38, 4 (2011), 273–277.

[13] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2015. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB J.* 24, 6 (2015), 707–730. https://doi.org/10.1007/s00778-015-0394-1

[14] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013.* 413–422. https://doi.org/10.1145/2488388.2488425

[15] Birte Glimm, Aidan Hogan, Markus Krötzsch, and Axel Polleres. 2012. OWL: Yet to arrive on the Web of Data?. In *WWW2012 Workshop on Linked Data on the Web, Lyon, France, 16 April, 2012.*

[16] Bernardo Cuenca Grau, Boris Motik, Zhe Wu, Ian Horrocks, Achille Fokoue, and Carsten Lutz. 2009. OWL 2 Web Ontology Language Profiles. https://www.w3.org/TR/owl2-profiles/

[17] Ramanathan Guha and Dan Brickley. 2014. RDF Schema 1.1. http://www.w3.org/TR/2014/REC-rdf-schema-20140225/

[18] Daniel Hernández, Aidan Hogan, and Markus Krötzsch. 2015. Reifying RDF: What Works Well With Wikidata?. In *Proceedings of the 11th International Workshop on Scalable Semantic Web Knowledge Base Systems co-located with 14th International Semantic Web Conference (ISWC 2015), Bethlehem, PA, USA, October 11, 2015.* 32–47.

[19] Vinh Thinh Ho, Daria Stepanova, Mohamed H. Gad-Elrab, Evgeny Kharlamov, and Gerhard Weikum. 2018. Rule Learning from Knowledge Graphs Guided by Embedding Models. In *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I.* 72–90. https://doi.org/10.1007/978-3-030-00671-6_5

[20] Joanna Józefowska, Agnieszka Lawrynowicz, and Tomasz Lukaszewski. 2010. The role of semantics in mining frequent patterns from knowledge bases in description logics with rules. *TPLP* 10, 3 (2010), 251–289. https://doi.org/10.1017/S1471068410000098

[21] Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. 2007. Finding All Justifications of OWL DL Entailments. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.* 267–280. https://doi.org/10.1007/978-3-540-76298-0_20

[22] Holger Knublauch and Dimitris Kontokostas. 2017. Shapes Constraint Language (SHACL). https://www.w3.org/TR/shacl/

[23] Roman Kontchakov and Michael Zakharyaschev. 2014. An Introduction to Description Logics and Query Rewriting. In *Reasoning Web. Reasoning on the Web in the Big Data Era - 10th International Summer School 2014, Athens, Greece, September 8-13, 2014. Proceedings.* 195–244. https://doi.org/10.1007/978-3-319-10587-1_5

[24] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. 2014. Test-driven evaluation of linked data quality. In *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014.* 747–758. https://doi.org/10.1145/2566486.2568002

[25] Jiaqing Liang, Yanghua Xiao, Yi Zhang, Seung-won Hwang, and Haixun Wang. 2017. Graph-Based Wrong IsA Relation Detection in a Large-Scale Lexical Taxonomy. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.* 1178–1184.

[26] Bing Liu, Wynne Hsu, and Yiming Ma. 1998. Integrating Classification and Association Rule Mining. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98), New York City, New York, USA, August 27-31, 1998.* 80–86.

[27] Christian Meilicke, Manuel Fink, Yanjie Wang, Daniel Ruffinelli, Rainer Gemulla, and Heiner Stuckenschmidt. 2018. Fine-Grained Evaluation of Rule- and Embedding-Based Systems for Knowledge Graph Completion. In *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I.* 3–20. https://doi.org/10.1007/978-3-030-00671-6_1

[28] Boris Motik, Ian Horrocks, and Ulrike Sattler. 2009. Bridging the gap between OWL and relational databases. *J. Web Sem.* 7, 2 (2009), 74–89. https://doi.org/10.1016/j.websem.2009.02.001

[29] Boris Motik and Peter Patel-Schneider. 2009. OWL 2 Web Ontology Language Mapping to RDF Graphs. https://www.w3.org/TR/owl-mapping-to-rdf/

[30] Peter F. Patel-Schneider. 2015. Using Description Logics for RDF Constraint Checking and Closed-World Recognition. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.* 247–253.

[31] Heiko Paulheim and Christian Bizer. 2014. Improving the Quality of Linked Data Using Statistical Distributions. *Int. J. Semantic Web Inf. Syst.* 10, 2 (2014), 63–86. https://doi.org/10.4018/ijswis.2014040104

[32] Christos Rantsoudis, Guillaume Feuillade, and Andreas Herzig. 2017. Repairing ABoxes through Active Integrity Constraints. In *Proceedings of the 30th International Workshop on Description Logics, Montpellier, France, July 18-21, 2017.*

[33] Viachaslau Sazonau, Uli Sattler, and Gavin Brown. 2015. General Terminology Induction in OWL. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I.* 533–550. https://doi.org/10.1007/978-3-319-25007-6_31

[34] Stefan Schlobach and Ronald Cornet. 2003. Non-Standard Reasoning Services for the Debugging of Description Logic Terminologies. In *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003.* 355–362.

[35] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007.* 697–706. https://doi.org/10.1145/1242572.1242667

[36] Thomas Pellissier Tanon, Daria Stepanova, Simon Razniewski, Paramita Mirza, and Gerhard Weikum. 2017. Completeness-Aware Rule Learning from Knowledge Graphs. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I.* 507–525. https://doi.org/10.1007/978-3-319-68288-4_30

[37] Jiao Tao, Evren Sirin, Jie Bao, and Deborah L. McGuinness. 2010. Integrity Constraints in OWL. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010.*

[38] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85. https://doi.org/10.1145/2629489

[39] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. *CoRR* abs/1412.6575 (2014). arXiv:1412.6575 http://arxiv.org/abs/1412.6575

[40] Fan Yang, Zhilin Yang, and William W. Cohen. 2017. Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA.* 2316–2325.

Bootstrapping Domain-Specific Content Discovery on the Web

Kien Pham
New York University
kien.pham@nyu.edu

Aécio Santos
New York University
aecio.santos@nyu.edu

Juliana Freire
New York University
juliana.freire@nyu.edu

ABSTRACT

The ability to continuously discover domain-specific content from the Web is critical for many applications. While focused crawling strategies have been shown to be effective for discovery, configuring a focused crawler is difficult and time-consuming. Given a domain of interest *D*, subject-matter experts (SMEs) must search for relevant websites and collect a set of representative Web pages to serve as training examples for creating a classifier that recognizes pages in *D*, as well as a set of pages to seed the crawl. In this paper, we propose DISCO, an approach designed to bootstrap domain-specific search. Given a small set of websites , DISCO aims to discover a large collection of relevant websites . DISCO uses a ranking-based framework that mimics the way users search for information on the Web: it iteratively discovers new pages, distills, and ranks them. It also applies multiple discovery strategies, including keyword-based and related queries issued to search engines, backward and forward crawling. By systematically combining these strategies, DISCO is able to attain high harvest rates and coverage for a variety of domains. We perform extensive experiments in four social-good domains, using data gathered by SMEs in the respective domains, and show that our approach is effective and outperforms state-of-the-art methods.

CCS CONCEPTS

• **Information systems → Web searching and information discovery**; *Page and site ranking*; *Content ranking*; Rank aggregation.

KEYWORDS

Focused crawling, Domain-specific Website discovery, Meta search

ACM Reference Format:
Kien Pham, Aécio Santos, and Juliana Freire. 2019. Bootstrapping Domain-Specific Content Discovery on the Web. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313709

1 INTRODUCTION

The ability to continuously discover content relevant to an information domain has many applications, from helping in the understanding of humanitarian crises to countering human and arms trafficking. It is estimated that tens to hundreds of thousands of escort ads are posted every day and about 75% of reported human-trafficking survivors were advertised online at some point [5]. Recently, the US law enforcement has shut down the sex marketplace website backpage.com due to its role in sex trafficking. Criminals also take advantage of online markets to avoid background checks and trade illegal weapons. According to an FBI report, hundreds of thousands of guns slip through the federal background-check system every year [30]. At the same time, many sites advertise weapons for sale – in armslist.com alone one can find hundreds of thousands of weapon-related ads.

By collecting and analyzing domain-specific content from the Web, we can better understand these illegal activities, generate leads, and obtain evidence for investigations. But doing so is challenging. Currently, subject matter experts (SMEs) (e.g., from non-governmental organizations and government agencies) must collaborate with computing experts and data scientists to construct specific solutions to their problems, which include (1) creating machine-learning models to recognize relevant content, and (2) configuring crawlers that retrieve the information at scale. Once these are available, it is possible to continuously discover relevant content by using re-crawling strategies [10, 23, 27]. However, both of these tasks require users to obtain a sufficiently large number of pages and finding these pages is difficult.

In this paper, we address this key problem in bootstrapping domain-specific search: how to effectively discover relevant websites given a small seed set provided by SMEs. A natural approach for discovering relevant websites is to use commercial search engines, such as Google and Bing, and issue keyword queries or search for similar sites using "related" search. This requires SMEs to go through an iterative process in which they search, distill the results, and use the information gathered to formulate additional queries. This process is cumbersome and inefficient for large-scale information gathering tasks. Different approaches have been proposed to tackle this problem. The Domain Discovery Tool (DDT) [15] aims to simplify the process of constructing classifiers for a given domain. To help users discover relevant pages and train a domain classifier, it provides an easy-to-use interface that summarizes search results and helps users formulate new search queries. While the tool is effective, it was designed with a user-in-the-loop philosophy and thus requires substantial user effort to perform the discovery operations and review the results. Methods such as forward and backward crawling [3] and DEXTER [24] have been be proposed to automate discovery. Because they rely on classifiers, they need to be adapted for different domains. Besides, these methods fall short of achieving high precision and recall in domains where relevant websites are not highly connected. To the best of our knowledge, there has been no systematic comparison of these methods for different domains, therefore whether they are effective for website discovery is still an open question.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313709

Several focused crawling and discovery techniques could potentially be adapted for this problem. However, they all rely on the availability of an accurate domain-specific classifier [3, 13, 31]. This is an unrealistic assumption for the many application scenarios where experts must start with a small set of relevant websites , since a small sample is unlikely to be sufficient construct an accurate classifier. These techniques use the classifier to select discovered pages that are relevant, and use content of these pages to train other models that directly control future search, e.g., ordering unvisited URLs and generating search queries. Therefore, a weak classifier would gradually degrade the effectiveness of discovery.

To address this challenge, we propose DISCO, a new method for website discovery that does not depend on an accurate domain-specific classifier. Given a small number of sample pages, DISCO automatically discovers additional pages that can be used both to construct domain classifiers and to serve as seeds for focused crawlers. The strategy adopted by DISCO was inspired by how users follow the search process: it searches for pages on the Web and distills the results in an iterative fashion. Different operations can be used to discover new pages, including keyword-based and related queries issued to search engines, and backward and forward crawling. However, the effectiveness of an operation varies across domains. For example, in our experiments (Section 5), we observed that backlink search performs the best for human trafficking domain, but is much less effective for the others. Based on this observation, unlike previous works [3, 31], DISCO applies multiple search operations. To systematically select the search operation as the discovery progresses, we propose an algorithm that uses the multi-armed bandit strategy [1]. This makes DISCO adaptive both within and across domains, and leads to higher harvest rates and coverage compared to a fixed strategy.

To distill the results, in the absence of a model that recognizes relevant pages, DISCO approximates how users select relevant pages by ranking them. Given a set of discovered pages, it ranks higher pages that are more similar to the input sample set. The challenge here lies in designing a strategy that is effective. We carried out an empirical study of several ranking approaches and found that none of the approaches is uniformly better for all domains. We propose a new ranking function that, by combining the results of multiple functions, produces a robust ranking.

Contributions. Our contributions can be summarized as follows:

- We propose DISCO, a framework that automatically discovers relevant websites requiring only a small set of example websites . To the best of our knowledge, ours is the first approach that does not require an accurate classifier to bootstrap domain discovery.
- To distill results, we propose an ensemble ranking approach that combines multiple independent ranking functions, and show that it consistently attains higher precision than the independent functions.
- We incorporate different search operations in our framework and propose an algorithm that uses the multi-armed bandits strategy to select the best search operator at each iteration. We also perform a systematic evaluation of the performance of the different operations in multiple domains and show that, by using the multi-armed bandits-based algorithm, our method is able to attain high harvest rates.
- We perform extensive experiments in multiple social-good domains using real data provided by SMEs from the corresponding domains. We report experimental results which show that DISCO obtains 300% higher harvest rate and coverage compared to state-of-the-art techniques for domain discovery.

Outline. The remainder of this paper is structured as follows. We discuss related work in Section 2. In Section 3, we define the website discovery and ranking problems and give an overview of the DISCO framework. In Section 4 we describe the ranking and discovery components of the framework. We present the results of our experimental evaluation in Section 5, and conclude in Section 6, where we outline directions for future work.

2 RELATED WORK

Several techniques have been proposed to discover domain-specific web content. These can be categorized in the following two groups: *Search-based Discovery* techniques [24, 31, 32] which rely on search engine APIs (e.g., Google, Bing, and Alexa APIs) to find web pages similar to given keywords or web pages ; and *Crawling-based Discovery* techniques [2, 3, 21], which use the Web link structure to explore new content by automatically downloading and recursively following links extracted from the discovered web pages . Note that while some of these techniques were proposed to discover specific web pages , they can be extended to discover websites as well. To the best of our knowledge, as we discuss below, none of the prior work supports all different discovery techniques, and most require an accurate domain-specific classifier.

2.1 Search-Based Discovery

There are two main search-based discovery operators supported by search engine APIs: *keyword search* and *related search*. Most of the existing discovery techniques use keyword search since related search is only supported by two search engines, Google and Alexa.

Earlier work [11, 21] used both content and Web link structure to compute the relatedness between web pages . Vieira et. al. [31] proposed a system that uses relevance feedback to gather seeds to bootstrap focused crawlers. It submits keyword search queries to Bing; extracts keywords from the result pages classified as relevant for the focus domain; and uses these keywords to construct new search queries. Their system shares some of our goals and their findings support our design decisions. They show that submitting multiple short queries and retrieving a small list of results leads to more relevant pages than submitting long queries and extracting a large number of results. The major drawback of this approach is that, when the classifier is not accurate, the search results quickly diverge from the target domain. Furthermore, the system is specifically designed to use with keyword search, therefore, does not support other search techniques.

Disheng et. al. [24] presented a discovery pipeline that uses keyword search to find websites containing product specifications (e.g., cameras and computers). While they also utilize backward search, they showed that the relevant websites are mainly discovered by searching with product identifications extracted from previous discovery iterations. However, using product identifications as search keyword is specific to product search domain and therefore not applicable to other domains.

Wang et. al. [32] focused on the claim-relevance discovery problem that, if solved, can help to identify online misinformation. They proposed a system that extracts salient keywords from fact-checking articles and uses them to search for candidate documents. Unlike in our search strategy, their search phase is not iterative, since they do not aim to attain high coverage of relevant documents.

2.2 Crawling-Based Discovery

Forward and backward crawling are two types of crawling-based techniques explored in the literature. In forward-crawling, links are extracted from the discovered web pages . Several forward-crawling approaches been proposed to discover domain-specific web pages and websites [4, 7, 8, 13]. The most prominent approach in this group are the *focused crawling* methods that employ online learning policies [4, 7, 20]. They use two classifiers to focus the search: the critic, a classifier that categorizes web pages as relevant or irrelevant to the domain; and the apprentice, a classifier that learns to identify the most promising links in a domain-relevant page in an online fashion. Note that while the apprentice classifier is automatically learned, the critic is a classifier given as input to the system and requires a substantial number of positive and negative examples to be trained on. In contrast, our approach does not require an input classifier, just a small set of relevant pages. Compared to search-based techniques, forward crawling is more scalable, however, it works best for domains where relevant content is well-connected.

In contrast to forward crawling, backward crawling (or reverse crawling) discovers new pages through backlink search. Since the Web is unidirectional, it must be done through search engine APIs that expose results of previous large-scale crawls. Early work proposed the use of backlink search to discover Web content [6, 12]. More recently, Barbosa et. al. [3] proposed a crawling strategy that combines both backward-crawling and forward-crawling techniques to locate bilingual websites. It employs two link classifiers, trained with the content of visited pages and their relevance to the target domain, to determine the visiting orders of the links discovered by the two techniques. As a result, the efficiency of this strategy highly depends on the domain classifier. In addition, this approach has the same limitation we discussed above, in that the crawler becomes ineffective when the classifier is not accurate. In later work, Barbosa [2] proposed a similar crawling architecture for discovering forum websites .

Approaches that focus on real-time discovery [26, 29, 34] have taken advantage of social media platforms (e.g., Twitter) to monitor and track domain-specific content such as misinformation [29], activity related to natural disasters [26], and rumors [34]. Although these techniques mainly target social-media feeds, they could potentially be used to discover websites by extracting links present in the feeds. Discovery from social media could be integrated in our approach as an alternative discovery operation.

3 PROBLEM AND SOLUTION OVERVIEW

3.1 Problem Definition

Website Discovery Problem. Let D represent the domain of interest and w denote a website represented as one of more web pages

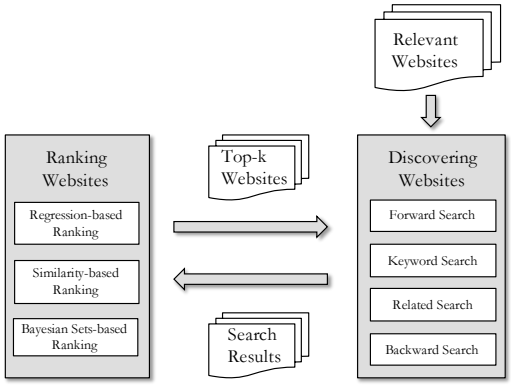


Figure 1: Architecture of DISCO.

. If w is relevant to the domain, we say that $w \in D$. Given a small set of websites $S \subset D$ (seeds), our goal is to efficiently expand S by finding additional websites w that belong to D.

Website Ranking Problem. In the absence of a model that recognizes pages in D, we approximate the problem to make it depend only on the set S of seeds by framing it as a ranking task: given S, the goal is to discover new websites that are *similar* to S. More formally, given set of discovered websites D_+, the problem consists of ranking each site w in D_+ according to its similarity with S – this is equivalent to the likelihood of w being in D. The intuition is that the closer w is to S, the more likely it is that $w \in D$.

By ranking the discovered websites based on their similarity with S, we help to reduce the SMEs effort in reviewing and labeling examples to build a domain-specific classifier and selecting important websites for the re-crawling process. A ranking strategy in this scenario should ideally have two important features: high precision in the top of the ranked list and low false negative in the tail of the ranked list. The former feature is important for the website discovery operation to be effective since the top-ranked websites are fed back into the discovery process. The latter is useful for reducing the SMEs effort of reviewing the final ranked list output by the system: the fewer relevant documents in the tail of the list, the fewer documents need to be reviewed by the SMEs. In our experimental evaluation in Section 5, we use metrics aligned with these desired features to evaluate different ranking functions.

3.2 Framework Overview

The design of DISCO was inspired by the iterative process that SMEs usually follow to discover domain-specific websites . They start by submitting queries to a search engine. Then, they review and evaluate the search results based on their knowledge about the domain. The new knowledge they acquire in this process is used to refine and formulate new search queries. In DISCO, we replace the manual evaluation with the website ranking component and use search engine APIs to automate the search operations. Figure 1 depicts the high-level architecture of the DISCO framework. It consists of two main components: *website ranking* and *website discovery*. To incorporate different search engine APIs and ranking techniques, we standardize the input and output of the two components. Specifically, the website discovery component takes a list of

Algorithm 1 Website Discovery

```
1:  procedure Discovery(seeds)
2:      results = ∅
3:      ranked_results = ∅
4:      topk = seeds
5:      while stop condition do
6:          op = select_discovery_operator()
7:          search_results = op.search(topk)
8:          results = results ∪ search_results
9:          ranked_results = rank(results)
10:         topk = get_top_k(ranked_results)
11:     end while
12:     return ranked_results
13: end procedure
```

relevant websites as input and returns a list of discovered websites after performing the search operations. Similarly, the input and output of the website ranking component consist of lists of discovered websites so far and the corresponding list of ranked websites , respectively. This design unifies multiple search operators in a single framework, and hence, it allows us to fairly compare their efficiency across different domains (see Section 5). Algorithm 1 outlines the discovery process carried out by DISCO. It starts with the seeds and iteratively searches, ranks and selects the top results. The *select_discovery_operator()* function in line 6 returns one of the four discovery operators: forward crawling, backward crawling, keyword search and related search.

4 RANKING AND DISCOVERY

In this section, we present the details of the website ranking and discovery components of DISCO. Then, we present a technique to efficiently select the discovery operator based on the multi-armed bandits algorithm.

4.1 Website Ranking

The website ranking problem can be cast as a conventional ranking problem in information retrieval, where S represents the query, and the goal is to rank the websites in D_+ according to its similarity to the query. The query here consists of a very long document containing all the websites in S, which makes the traditional ranking models (e.g., BM25 [25]) less effective. Furthermore, merging all the websites in S can potentially blur the distinct domain-specific features that exist in individual websites . To avoid this problem, we investigate other ranking approaches that consider websites in S individually. We then propose a new ranking function, inspired by ensemble learning methods, that combines the scores of the existing approaches.

Website Representation. The first step of any ranking function design is to represent the candidate documents, which are websites in our context. A website can be represented as a single or multiple web pages . We consider both options and selected the latter for several reasons. To use multiple pages to represent a site, we need to have an effective procedure to select representative pages in the sites. This task is challenging, especially when we do not have ground truth data. Random selection tends to result in a large number of web pages that are not good representative of the site and that lead to noise that negatively affects the ranking. The single-web page representation can help circumvent this issue if we rely on the assumption that every website contains at least one web page that can serve as an effective representative of the website content. As a result, the score of a website can depend only on the score of its most relevant web page discovered. Furthermore, the single-web page representation greatly simplifies the implementation of the framework and improves its performance. For this representation, we do not need a separate process to retrieve and manage additional pages for each newly discovered website , and discovered web pages can be processed independently. This makes the framework ready for parallelization.

In fact, we experimentally compared selection strategies using multiple web pages representation against the single-page representation, and the former led to better rankings than the latter for all domains we considered. We considered two different approaches for the multiple-page representation. One performs a breadth-first-search crawling starting from the home page, which leads to both relevant and irrelevant pages. The other uses in-site search via search engine APIs (i.e., using queries with the format "site:keyword"). This method, however, is keyword-sensitive and tends to retrieve false positive pages.

Several ranking function have been proposed in the literature. In what follows, we describe the ones we considered. Details about their performance are discussed in Section 5.

4.1.1 Regression-Based Ranking. This approach learns the ranking function using regression-based techniques and computes a score for each website in D_+. The challenge is that only positive examples (i.e., S) are given, and thus additional negative examples must be obtained or unconventional techniques such as PU learning [18] or novelty detection [28] must be applied. We investigate three regression techniques in this direction and describe how they can be applied in our scenario.

Binomial Regression. In order to use binomial regression (i.e., logistic regression), we need both positive and negative examples. Therefore, we obtain negative examples by randomly selecting web pages from a public Web corpus. Since the Web corpus is relatively large compared to the number of samples pages, there is only a very small chance of selecting a web page relevant to the domain. Given S and the unlabeled web pages as negative examples, we train a binomial regressor and use it to compute ranking scores.

Positive and Unlabeled Example Learning (PU Learning). A well-known alternative to binomial regression method, given positive and unlabeled examples, is PU learning [18]. Liu et al. [18] proposed biased SVM, a PU learning technique that frames the PU Learning problem as a constrained optimization problem. They experimentally showed that biased SVM outperforms other 2-steps PU learning approaches. Therefore, we choose this technique to learn the ranking function.

Novelty Detection. Another line of research that does not require negative examples is novelty detection [19, 28]. Scholkopf et. al. [28] extend SVM (support vector machine) to learn a function that is positive on the positive examples and negative on its complement.

Intuitively, it computes the minimal region in feature space that encloses the positive examples.

4.1.2 Similarity-Based Ranking. Similarity-based ranking, inspired by the k-nearest neighbors algorithm, computes the ranking score of a website according to the average of its similarities to all websites in S. Let $Sim(w_i, w_j)$ be a similarity function between the websites w_i and w_j, the score of a website $w \in D_+$ is computed as follows:

$$Score(w) = \frac{1}{|S|} \sum_{w_s \in S} Sim(w, w_s) \tag{1}$$

We consider Jaccard and Cosine as similarity functions. These are two well-known similarity measures for textual content that use distinct input representations, which might encapsulate different ranking advantages. We use the vector space model and binary vector to represent websites for computing Cosine and Jaccard similarity respectively. Since these similarity measures use different representations of the websites , they might expose different characteristics in the ranking task. Let x, y be the representation of websites w_x, w_y, the Jaccard index and Cosine similarity between them are computed as follows:

$$Jaccard(x, y) = \frac{x \cap y}{x \cup y} \tag{2}$$

$$Cosine(x, y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}} \tag{3}$$

4.1.3 Bayesian-Sets-Based Ranking. Bayesian Sets (BS) is a Bayesian inference technique introduced by Ghahramani and Heller in [14] that performs set expansion – it generates high-precision sets of related items. Specifically, given a set of related items that belong to a class C, BS scores a new item by estimating its normalized marginal probability that the item belongs to C. By this definition, BS can be directly applied to our ranking problem by considering the related items as seeds S and class C as domain D. Accordingly, the score of a website w is computed as the probability of w given that S is observed:

$$Score(w) = \frac{P(w|S)}{P(w)} \tag{4}$$

4.1.4 Ensemble Ranking Function. While many alternatives are possible for ranking functions, none of them is uniformly better for all domains, as we experimentally show in Section 5. Thus, ideally, we should be able to combine them. Ensemble learning is a paradigm that combines multiple machine-learning models and has been shown to be effective at reducing their variance. In our context, each ranking function captures a different set of features, hence their combination has the potential to improve the robustness and reduce the variance of the ranking. Inspired by this, we propose a new ranking function that combines the results from multiple ranking functions. Specifically, let $F = \{f_i\}$ be the list of the ranking functions and $f_i(w)$ denote the position of w in the ranked list that w belongs to. The new ensemble score of w is computed as the mean of these position values:

$$Score(w) = \frac{\sum_{f_i \in F} f_i(w)}{|F|} \tag{5}$$

Another option could be to directly consider the similarity scores as $f_i(w)$. However, these scores must be calibrated for the ensemble function to be effective. Our similarity functions do not produce naturally calibrated scores (i.e., probabilities), hence, we decided to use rank positions for combining the results. We also considered using other rank combination methods that have been proposed in previous work, e.g., reciprocal rank fusion [9]. However, we found that the mean function performed comparably well in our case and at same time it was faster.

4.2 Website Discovery Operators

The goal of the discovery operators in our context is to discover new websites , therefore we make use of existing discovery methods in a way that maximizes this objective. We investigate four different discovery techniques: forward crawling, backward crawling, keyword search, and related search. To incorporate these discovery methods in our framework, we designed them in such a way that they take the same input, the top-k most relevant websites , and produce the same output, a list of new websites . In what follows, we describe in detail each of these methods.

Forward Crawling. The forward crawling operator discovers new websites through outlinks extracted from the HTML content of the input web pages . Since the objective is to maximize the number of relevant websites , the operator only returns the outlinks for websites have not been discovered in the previous iterations.

Backward Crawling. The backward crawling operator discovers new websites by performing two operations: backlink search and forward crawling. In backlink search, it calls search engine APIs to obtain backlinks for the input web pages . Since the input is relevant to the domain, the discovered backlinks are likely to serve as hubs pointing relevant pages. This phenomenon has been referred to as co-citation in the literature [16]. Then, we can discover additional relevant websites by extracting outlinks from the hubs and doing forward crawling.

Keyword Search. The keyword search makes use of search engine APIs to search for websites relevant to the keywords. The problem is how to obtain domain-specific keywords from the input websites that lead to more relevant websites . The body text extracted from the input pages contains good keywords that are representative of the domain, however, they often co-occur with terms that may not be representative. Our observation is that the keywords extracted from the metadata tags (i.e., description and keyword) are significantly less noisy than the ones extracted from the body text. For this reason, they are particularly effective for searching for relevant websites . For each page, we extract and tokenize the metadata from the description and keywords tags. We select the most frequent tokens as search keyword candidates. The problem is that, even though many keywords are relevant to the domain, do not comprise sufficient information to retrieve relevant websites . For example, the keywords *pistol*, *texas*, and *sale* are related to the *weapon forums* domain, but more contextual information is needed to build search queries that find additional relevant websites . To address this challenge, we combine these keywords with terms that are good descriptors of the domain. For example, in the *weapon forums* domain, given *gun forum* as seed keyword, the corresponding search terms of the previous keywords are *gun forum pistol*, *gun*

forum texas and *gun forum sale*. Note that we only need one seed keyword for each domain, so this should not add any burden to the SMEs.

Related Search. The related search operator takes a website URL as input and returns a list of related websites . To our knowledge, only Google Search APIs and Alexa APIs support this operation. Depending on the nature of the data that these search platforms possess and their underlying algorithms, the results can be vastly different. Nevertheless, it appears to be a natural fit for our objective of discovering relevant websites . Note also that different search providers can be easily included as different search operators.

4.3 Discovery Operator Selection

In the previous section, we have described four different operators for discovering relevant websites , the challenge now is how to select an operator at each iteration in Algorithm 1 that maximizes the rate of discovering new relevant websites .

This is a non-trivial task for several reasons. First, it is not clear which search operator performs the best across multiple domains. In our experiments, described in Section 5, backward crawling performs better than other operators for website discovery in the Human Trafficking domain, but it performs poorly in other domains. Furthermore, favoring only search operators that are known to be good can leave the discovery process stuck in a connected region of the Web. On the other hand, using different discovery operators can diversify the discovery process, and eventually improve the rate of harvesting relevant websites .

To address this problem, we propose a strategy to balance the trade-offs between different discovery operators using multi-armed bandits (MAB), particularly UCB1 algorithm [1]. MAB is known as an effective means to solve the exploitation and exploration problem. The goal of this strategy is to select bandit arms with the highest accumulated reward and to penalize the ones that were overly operated in the past. To accomplish this, we need to define two variables in the UCB1 algorithm: the bandit arm (*op*) and the reward of its operation (μ_{op}). We model each discovery operator as a bandit arm. Assume that each time t, when an arm *op* is selected, it discovers $n_{op,t}$ websites . According to the UCB1 algorithm, the score of *op* at time t is defined as:

$$Score_{op,t} = \overline{\mu_{op}} + \sqrt{\frac{2ln(n)}{n_{op,t}}} \tag{6}$$

In this equation, n is the total number of websites retrieved by all operators. At each iteration, the algorithm selects the operator with highest score to perform the discovery.

The remaining challenge is to define the reward of an operator so that it is proportional to the number of relevant websites it discovered. Although we do not know the relevance of the discovered websites , we can approximate them by their ranking scores computed by the website ranking component (Section 4.1). Let pos_i denote the position of a website i in the ranked list. The normalized reward of i is $1 - \frac{pos_i}{len}$, where len is length of the ranked list. By this definition, if i is ranked at the end or top of the list, its reward is 0 or 1 respectively. Since the goal is to maximize the number of new relevant websites , we assign 0 reward to a website that was already discovered in previous iterations, regardless of its position. As a result, the reward of i is redefined as $(1 - \frac{pos_i}{len}) * I_i$, where I_i takes binary values:

$$I_i = \begin{cases} 0 & \text{website } i \text{ was already discovered} \\ 1 & \text{otherwise} \end{cases} \tag{7}$$

If a discovery operator *op* returns k websites , its reward μ_{op} is defined as follows.

$$\mu_{op} = \frac{\sum_{i=1}^{k}(1 - \frac{pos_i}{len}) * I_i}{k} \tag{8}$$

We use this reward to compute the score of an operator at time t ($Score_{op,t}$) in the equation 6. After each round of search, we recompute this score and select the operator with highest score for the next round.

5 EXPERIMENTAL EVALUATION

In this section, we evaluate the effectiveness of our discovery framework and ranking functions through extensive experiments using real-world datasets. We use the following domains for the experiments:

- Human Trafficking (HT): classified ads and other escort-related websites which often contain adult ads. Prominent example websites in this domain are `backpage.com`, `craigslist.com`, and `adultsearch.com`.
- Weapon Forums (Forum): discussion forums about firearms and weapons.
- Weapon Marketplaces (Market): marketplace or classified ads websites for firearms and weapons.
- Stock Promotions (SEC): websites that post stock promotion alerts. This domain contains information that can support investigations of micro-cap fraud.

We selected these domains for the following reasons. First, since they are different in nature, they allow us to evaluate the robustness of our proposed techniques and baselines. Most importantly, we had access to the seed websites and the training examples manually labeled by SMEs in these domains. The SMEs were analysts from government agencies and worked with us in collecting and reviewing relevant content. The number of seed websites in HT, Forum, Market and SEC domains are 54, 103, 23, and 18 respectively. These are representative websites that contain a significant number of pages relevant to the domains. These seed websites are used for the evaluation of the ranking methods in 5.2 and for the input of the DISCO framework in Section 5.3. As training examples (described in Table 1), we used a bigger set of websites, labeled as relevant and non-relevant by SMEs in HT, Forum and Market domains. We use these examples to create ground-truth classifiers for the evaluation of the DISCO framework, which we present in Section 5.3.

It is worthy of note that although we were able to obtain ground-truth to train these classifiers, labeling the pages required a very time-consuming process. In addition, the labeling process was not a trivial task for the SMEs, who are familiar with the domain but not with the intricacies of learning. For example, they did not know the impact of labeling some confusing cases would have in the final model. An important motivation for this work is to devise an easier-to-use and less time-consuming process to solve this problem.

5.1 Evaluation Metrics

5.1.1 Website Ranking Metrics. Section 3.1 presented two desired characteristics of good ranking functions: high precision in the top of the ranked list and low false negative in its tail. In order to measure these qualities, we use the following metrics:

P@k: computes the percentage of the relevant websites in the top-k websites of the ranked list. This metric captures the quality at the top of the list.

Mean Rank: computes mean of the ranking positions of the relevant websites in the ranked list.

$$MR = \frac{1}{n} \sum_{w_i \in D_+^*} position(w_i) \tag{9}$$

We do not normalize the rank by the length of the ranked list because we want to keep the absolute values for easy interpretation. Also, in our experiment, the length of the ranked lists in the experimental domains are similar, therefore absolute values are still valid for cross-domain comparison.

Median Rank: position of the relevant website whose position after ranking is the midpoint of all the relevant websites .

5.1.2 Website Discovery Metrics. We measure the discovery efficiency using *Harvest Rate* and *Coverage* metrics. In the following metric definitions, let D_+^* be the set of discovered websites that are in the domain D.

Harvest Rate: the ratio between the size of D_+^* and the size of D_+. The Harvest Rate is a standard metric that has been widely used in literature to measure the efficiency of focused crawlers.

Coverage: we define coverage as the percentage of websites in D that are discovered.

5.2 Website Ranking Evaluation

For each domain, we randomly select 10,000 websites from DMOZ directory[1] as negative candidates. Since the domains are rare and the size of DMOZ is large (about 4M websites), there is only a small chance that the selected examples are relevant to the domain. Therefore, we can reliably consider these as negative candidates. For each domain, we split the seed websites into train and test sets randomly. We merge the test set into the set of negative examples, creating a new set named the *candidate set*. Then, we rank the candidate set using the ranking functions described in Section 4.1. We compute the evaluation metrics based on the positions of the test set in the ranked list.

JACCARD, COSINE: These ranking functions compute the similarity between two websites using JACCARD and COSINE similarity index, respectively.

BS: This ranking function uses Bayesian sets technique. We represent each website as a term-frequency vector as opposed to the binary vector used in [14].

ONECLASS: This ranking function uses novelty detection (i.e., one-class SVM [28]).

BINOMIAL: This ranking function uses binomial regression. We use logistic regression because it is fast and known for performing well on text data. To obtain the negative examples, we randomly select websites from DMOZ. The number of selected examples was equal to the number of positives examples. For the implementation of both *ONECLASS* and *BINOMIAL*, we used scikit-learn [22].

PUL: This ranking function uses biased SVM [18]. We use SVM light[2] as an implementation of biased SVM. We also use DMOZ to obtain the unlabeled examples for learning this function.

ENSEMBLE: This is our ranking function that combines all the above ranking functions, except *PUL*, whose performance showed a high variability across different domains.

Figures 2, 3, and 4 compare the ranking functions using different metrics: *P@k*, mean rank and median rank. Note that we select a k that is equal to the number of relevant websites (test set) in the candidate set. By doing this, we basically normalize the *P@k* values to range [0, 1]. The figure shows that the ENSEMBLE ranking function achieves better performance than the individual ranking functions do in most of the cases. This confirms that the combination of different ranking functions is able to reduce the variance and increase the robustness of the overall ranking. We also observe that *BS* consistently performs better than other individual ranking functions in median rank metric, i.e., it rarely places a relevant website to the end of the list, which is one of our desired features. Another observation is that the precision in SEC domain is relatively low (66.67%) compared to the others. This can be explained by the fact that in this domain, relevant websites might contain a mix of topics that are less distinguishable. Another reason is that this domain has smallest number of seeds among other domains.

Influence of the Number of Seed Websites . To understand how the number of seed websites impacts the effectiveness of the ranking, we conduct a ranking evaluation with different number of seed websites . Figure 5 shows the *P@k* corresponding to number of seed websites . We notice that the precision improves when we increase the number of seed websites up to 13 and stabilizes after that. This finding suggests that this number of seeds is sufficient to achieve a good ranking performance. It also reveals that probably there is still room to improve ranking performance, especially for cases in which a larger number of seed websites is available. Given recent advances in text representation using dense vector representations and document distance computation in the embedding space [17, 33], this is a promising area for future work. While this is a limitation of our ranking approach, it is still suitable for our context, where we assume to have only a small list of seeds as input.

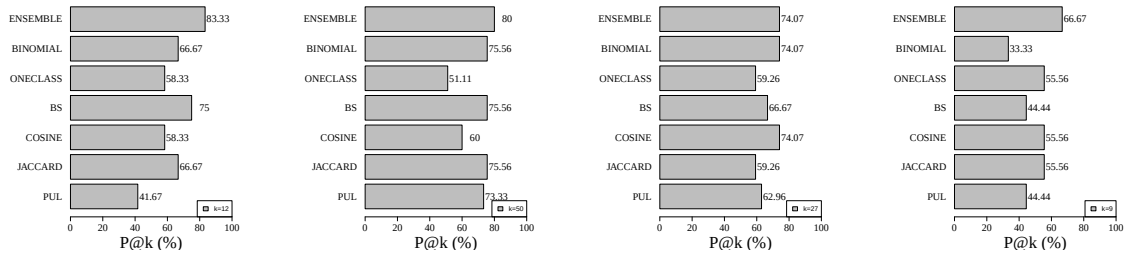
5.3 Discovery Evaluation

In this section, we assess the effectiveness of DISCO and compare it to state-of-the-art discovery strategies [3, 4, 31] (baselines). Since the baselines require a page classifier to operate, we train one classifier for each domain using seed websites and unlabeled examples as training data. Specifically, we consider seed websites as positive examples, and randomly select pages from DMOZ to use as negative examples. For training the classifiers, we use an implementation of SVM from the scikit-learn library [22].

To compute the discovery metrics, we train one ground-truth page classifier for each domain using the positive and negative examples manually labeled by SMEs. Note that these classifiers are only used for evaluation purposes. Although not ideal, these classifiers are the closest to the real ground truth that we can possibly

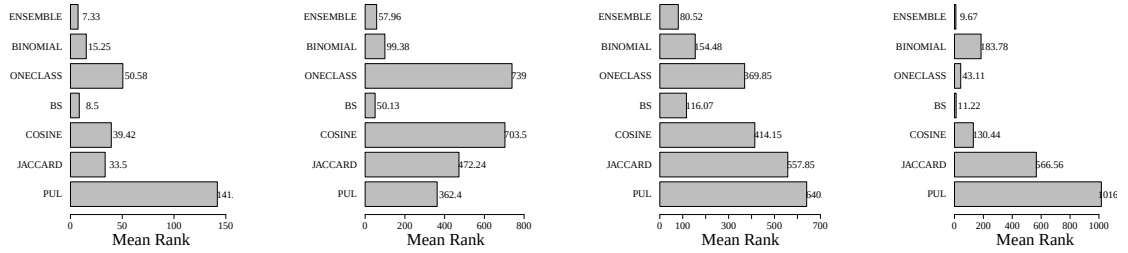
[1] http://dmoz-odp.org/

[2] http://svmlight.joachims.org/



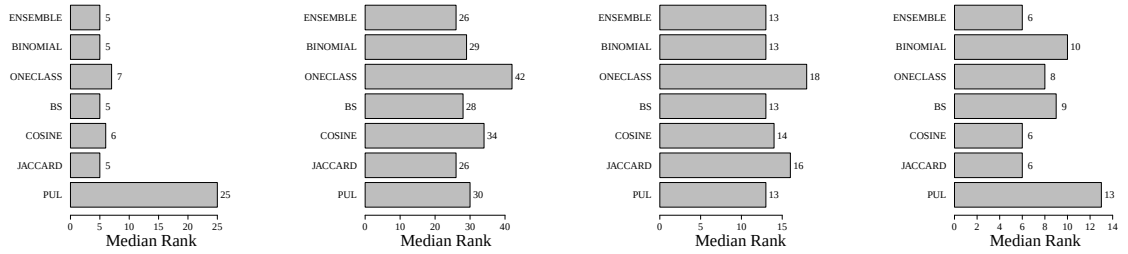
(a) Market Domain (b) Forum Domain (c) HT Domain (d) SEC Domain

Figure 2: Comparison of different ranking functions using P@k metric in all domains. Higher values are better.



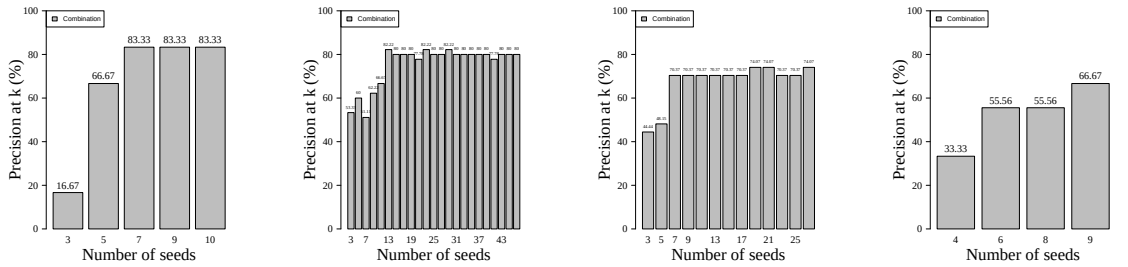
(a) Market Domain (b) Forum Domain (c) HT Domain (d) SEC Domain

Figure 3: Comparison of different ranking functions using mean rank metric in all domains. Lower values are better.



(a) Market Domain (b) Forum Domain (c) HT Domain (d) SEC Domain

Figure 4: Comparison of different ranking functions using median rank metric in all domains. Lower values are better.



(a) Market Domain (b) Forum Domain (c) HT Domain (d) SEC Domain

Figure 5: Experiment showing the influence of the number of seed websites . When the number of seeds increases, the precision also increases until it reaches a plateau. In general, less than 15 examples are required to attain high precision.

Table 1: Statistics of the training data and accuracy

	HT	Market	Forum
# of positive examples	968	68	103
# of negative examples	7117	169	256
Accuracy (5-fold CV)	0.97	0.96	0.84

obtain. Nevertheless, they provide a scalable and consistent way to evaluate the discovery results. Table 1 summarizes the training data and cross-validation accuracy of these classifiers. The accuracy ranges from 84% to 97%, which are sufficiently reliable for evaluating discovered websites . For each method, we stop the algorithm when it retrieves 50,000 web pages . In fact, all the baselines stop before reaching this limit. In the following, we describe baselines and different configurations of DISCO framework used in the evaluation.

We used the following approaches as baselines:

ACHE: ACHE is an adaptive focused crawler described in [4]. We used an open-source implementation available online.[3] This implementation uses an online-learning crawling policy [7] to prioritize the unvisited links. Since ACHE is designed to maximize the number of relevant web pages instead of relevant websites , we limit the number of crawled web pages per website to 10 to ensure it will visit more websites . While a smaller limit would yield higher rate of new websites , it would potentially lead to missing relevant websites that exist but would not have been discovered because some web pages would never been visited. Indeed, our experiment with a smaller limit results in a lower harvest rate and the crawler also stops earlier due to running out of links to crawl.

BIPARTITE: This crawling strategy was proposed by Barbosa et. al. [3] and, similar to ACHE, we set the page-per-site limit to 100.

SEEDFINDER: We implemented the SeedFinder algorithm proposed by Vieira et. al. [31]. SeedFinder uses the keyword search operator to discover new web pages . It then extracts new keywords from the content of the pages classified as relevant by the page classifier to derive new queries using relevance feedback.

We used the following search operators for DISCO:

KEYWORD: DISCO used with the keyword search operator. For a fair comparison, this method uses the same settings as SEEDFINDER (i.e., initial keywords and the number of search results returned by search APIs).

RELATED: DISCO used with the related search operator. Since the objective is to discover websites , we use host name of the input URL to form the search query. For example, given the URL `http://example.com/path.html`, the query is represented as `related:http://example.com`.

FORWARD: DISCO used with the forward crawling operator.

BACKWARD: DISCO used with the backward crawling operator.

BANDIT: DISCO used with the multi-armed bandits algorithm for selecting the discovery operator.

Implementation Details. For KEYWORD and SEEDFINDER methods, we use "gun classified", "gun forum" and "adult escort" as seed keywords for the Market, Forum and HT domains respectively. We selected these keywords since they represent well these domains and we observed highly relevant results by searching these keywords on Google. To obtain new keywords, we tokenize the textual content extracted from the description and keywords tag, then select the most frequent keywords to use in the next search iteration. In each iteration, we select at most 20 new keywords. All methods start with the seeds provided by SMEs (also used in the ranking experiment). We limit the number of results retrieved by keyword search, related search and backlink search to be 50, 50, 5, respectively. We set k in Algorithm 1 to 20. DISCO uses ENSEMBLE as the ranking function because it outperforms other ranking functions and is consistent across multiple domains. We use MOZ API[4] for backlink search, Bing Web Search API[5] for keyword search, and Google Custom Search API[6] for related search.

5.3.1 Comparing Harvest Rates. We use the ground-truth classifiers to classify the discovered web pages . A website is considered as relevant if it contains at least one relevant web page . Figure 6 and Figure 7 show the number of discovered relevant web pages and websites versus the number of discovered web pages over time. Overall, DISCO with the best configuration (BANDIT) obtains at least 300% higher harvest rate compared to other baselines. Also, BANDIT consistently outperforms other configurations in all the three domains in terms of discovered websites . However, if we measure the harvest rate by relevant web pages , ACHE performs closely to the DISCO strategies in the Market and Forum domains. The reason is that ACHE's objective is to maximize the number of relevant web pages by exploiting relevant websites that are already discovered. However, it quickly reaches a plateau when fewer relevant websites are found. Another reason that ACHE performs well in HT domain is that websites in this domain are highly connected. This also explains why backlink method performs better in this domain than the others. SEEDFINDER and BIPARTITE strategies stop early, before reaching 5,000 pages in all the domains, due to running out of links to crawl.

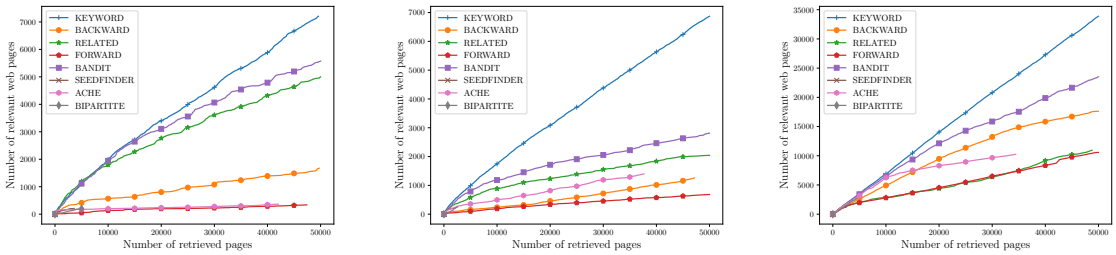
5.3.2 Comparing Coverage. Since it is impractical to obtain the real ground truth for each domain (i.e., all the relevant websites in the domain), we use the union of discovered websites from all discovery methods as an approximation for the ground truth. By doing this, while the estimated coverage can be largely different from the real value, it serves as reasonable relative measure to compare different methods. Figure 8 shows the coverage of all discovery methods in the considered domains. For each method, we compute the percentage of relevant websites that are also discovered by all other methods, which we denominate as *intersection* in the plots. The *complement* region represents the percentage of relevant websites that are discovered only by the corresponding method. The BANDIT strategy consistently attains the largest intersection among all methods. This suggests that each operator searches a different region of the Web, and by adaptively combining them, the BANDIT strategy is able to explore a broader portion of the Web.

[3] http://github.com/ViDA-NYU/ache

[4] https://moz.com/products/api

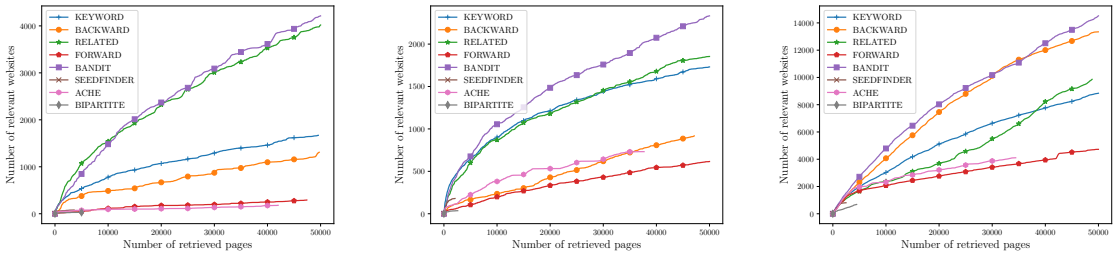
[5] https://bit.ly/2KYAzu2

[6] https://developers.google.com/custom-search/



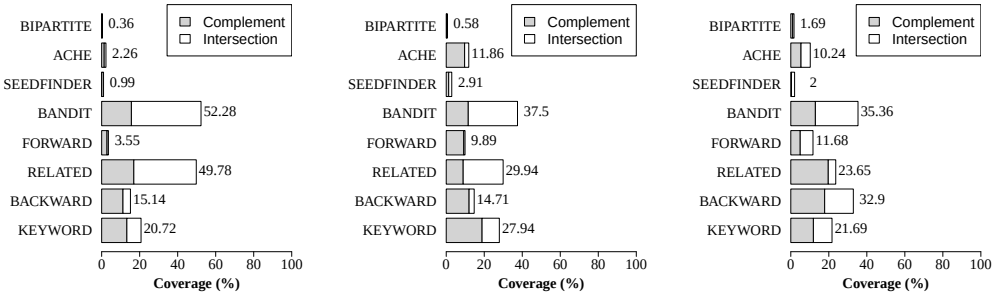
(a) Market Domain (b) Forum Domain (c) HT Domain

Figure 6: Comparison of harvest rate of relevant web pages for different discovery methods.



(a) Market Domain (b) Forum Domain (c) HT Domain

Figure 7: Comparison of harvest rate of relevant websites between for discovery methods.



(a) Market Domain (b) Forum Domain (c) HT Domain

Figure 8: Comparison of coverage for different discovery methods.

6 CONCLUSION

In this paper, we propose DISCO, a new approach that helps bootstrap domain discovery: given a small number of sample sites in a domain, it automatically expands the set. DISCO employs an iterative crawling procedure that combines multiple search operators and, as proxy to select relevant pages, it ranks the search results based on their similarity to the sample sites. We perform extensive experiments using real-world data from different social-good domains and show that our framework is effective and attains significant gains in harvest rate compared with state-of-the-art methods. The experimental results suggest that DISCO can be effective at lowering the burden on users that need to create page classifiers and discover information on the Web at scale. For example, DISCO could be used to both automatically provide additional seed pages for DDT [15] and to rank the results of user-specified search operations.

While DISCO is effective for the domains we experimented with, it has limitations. Our ranking approach is only suitable for the domains for which using n-gram features are effective and the given seed websites are representative for the domains. Additionally, keyword search is very sensitive to the seed keywords, which can be hard to select for some domains. In future work, we would like to explore image-based search for this problem, which can be

potentially applied to image-rich domains, such as weapon ads and human trafficking.

Acknowledgments. This work was partially supported by the DARPA MEMEX and D3M programs, and NSF award CNS-1229185. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF and DARPA.

REFERENCES

[1] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning* 47, 2-3 (May 2002), 235–256. https://doi.org/10.1023/A:1013689704352

[2] Luciano Barbosa. 2017. Harvesting Forum Pages from Seed Sites. In *Web Engineering*, Jordi Cabot, Roberto De Virgilio, and Riccardo Torlone (Eds.). Springer International Publishing, 457–468.

[3] Luciano Barbosa, Srinivas Bangalore, and Vivek Kumar Rangarajan Sridhar. 2011. Crawling Back and Forth: Using Back and Out Links to Locate Bilingual Sites. In *Proceedings of the 5th International Joint Conference on Natural Language Processing, IJCNLP*. 429–437.

[4] Luciano Barbosa and Juliana Freire. 2007. An Adaptive Crawler for Locating hidden-Web Entry Points. In *Proceedings of the 16th International Conference on World Wide Web (WWW)*. ACM, New York, NY, USA, 441–450. https://doi.org/10.1145/1242572.1242632

[5] Vanessa Bouché. 2018. Survivor Insights: The Role of Technology in Domestic Minor Sex Trafficking. https://www.wearethorn.org/survivor-insights/.

[6] Soumen Chakrabarti, David A Gibson, and Kevin S McCurley. 1999. Surfing the Web backwards. *Computer Networks* 31, 11 (1999), 1679 – 1693. https://doi.org/10.1016/S1389-1286(99)00042-0

[7] Soumen Chakrabarti, Kunal Punera, and Mallela Subramanyam. 2002. Accelerated Focused Crawling Through Online Relevance Feedback. In *Proceedings of the 11th International Conference on World Wide Web (WWW)*. ACM, New York, NY, USA, 148–159. https://doi.org/10.1145/511446.511466

[8] Soumen Chakrabarti, Martin van den Berg, and Byron Dom. 1999. Focused Crawling: A New Approach to Topic-specific Web Resource Discovery. *Comput. Netw.* 31, 11-16 (May 1999), 1623–1640. https://doi.org/10.1016/S1389-1286(99)00052-3

[9] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, 758–759. https://doi.org/10.1145/1571941.1572114

[10] Anirban Dasgupta, Arpita Ghosh, Ravi Kumar, Christopher Olston, Sandeep Pandey, and Andrew Tomkins. 2007. The Discoverability of the Web. In *Proceedings of the 16th International Conference on World Wide Web (WWW)*. ACM, New York, NY, USA, 421–430. https://doi.org/10.1145/1242572.1242630

[11] Jeffrey Dean and Monika R Henzinger. 1999. Finding related pages in the World Wide Web. *Computer Networks* 31, 11 (1999), 1467 – 1479. https://doi.org/10.1016/S1389-1286(99)00022-5

[12] Michelangelo Diligenti, Frans Coetzee, Steve Lawrence, C. Lee Giles, and Marco Gori. 2000. Focused Crawling Using Context Graphs. In *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 527–534.

[13] Martin Ester, Hans-Peter Kriegel, and Matthias Schubert. 2004. Accurate and Efficient Crawling for Relevant Websites. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases (VLDB)*. VLDB Endowment, 396–407.

[14] Zoubin Ghahramani and Katherine A Heller. 2006. Bayesian Sets. In *Advances in Neural Information Processing Systems (NIPS)*, Y. Weiss, B. Schölkopf, and J. C. Platt (Eds.). MIT Press, Cambridge, MA, USA, 435–442.

[15] Yamuna Krishnamurthy, Kien Pham, Aecio Santos, and Juliana Freire. 2016. *Interactive Exploration for Domain Discovery on the Web*. 64–71.

[16] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. 1999. Trawling the Web for Emerging Cyber-communities. In *Proceedings of the Eighth International Conference on World Wide Web (WWW)*. Elsevier North-Holland, Inc., New York, NY, USA, 1481–1493.

[17] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From Word Embeddings To Document Distances. In *Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Francis Bach and David Blei (Eds.), Vol. 37. PMLR, Lille, France, 957–966.

[18] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S. Yu. 2003. Building Text Classifiers Using Positive and Unlabeled Examples. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM)*. IEEE Computer Society, Washington, DC, USA, 179–187.

[19] Markos Markou and Sameer Singh. 2003. Novelty Detection: A Review - Part 1: Statistical Approaches. *Signal Processing* 83 (2003), 2003.

[20] Robert Meusel, Peter Mika, and Roi Blanco. 2014. Focused Crawling for Structured Data. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM)*. ACM, New York, NY, USA, 1039–1048. https://doi.org/10.1145/2661829.2661902

[21] Tsuyoshi Murata. 2001. Finding Related Web Pages Based on Connectivity Information from a Search Engine. In *Poster Proceedings of 10th International Conference on World Wide Web (WWW)*. 18–19.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[23] Kien Pham, Aécio Santos, and Juliana Freire. 2018. Learning to Discover Domain-Specific Web Content. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, New York, NY, USA, 432–440. https://doi.org/10.1145/3159652.3159724

[24] Disheng Qiu, Luciano Barbosa, Xin Luna Dong, Yanyan Shen, and Divesh Srivastava. 2015. Dexter: Large-scale Discovery and Extraction of Product Specifications on the Web. *Proc. VLDB Endow.* 8, 13 (Sept. 2015), 2194–2205. https://doi.org/10.14778/2831360.2831372

[25] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (April 2009), 333–389. https://doi.org/10.1561/1500000019

[26] Jakob Rogstadius, Maja Vukovic, C. A. Teixeira, Vassilis Kostakos, Evangelos Karapanos, and Jim Laredo. 2013. CrisisTracker: Crowdsourced social media curation for disaster awareness. *IBM Journal of Research and Development* 57, 5 (2013), 4:1–4:13. https://doi.org/10.1147/JRD.2013.2260692

[27] Aécio Santos, Bruno Pasini, and Juliana Freire. 2016. A First Study on Temporal Dynamics of Topics on the Web. In *Proceedings of the 25th International Conference Companion on World Wide Web*. 849–854. https://doi.org/10.1145/2872518.2889291

[28] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. 1999. Support Vector Method for Novelty Detection. In *Proceedings of the 12th International Conference on Neural Information Processing Systems (NIPS)*. MIT Press, Cambridge, MA, USA, 582–588.

[29] Chengcheng Shao, Giovanni Luca Ciampaglia, Alessandro Flammini, and Filippo Menczer. 2016. Hoaxy: A Platform for Tracking Online Misinformation. In *Proceedings of the 25th International Conference Companion on World Wide Web (WWW)*. 745–750. https://doi.org/10.1145/2872518.2890098

[30] Matt Stroud. 2015. FBI data show thousands of gun sales beat checks. https://docs.house.gov/meetings/JU/JU00/20151117/104114/HHRG-114-JU00-20151117-SD007.pdf.

[31] Karane Vieira, Luciano Barbosa, Altigran Soares da Silva, Juliana Freire, and Edleno Moura. 2016. Finding seeds to bootstrap focused crawlers. *World Wide Web* 19, 3 (2016), 449–474. https://doi.org/10.1007/s11280-015-0331-7

[32] Xuezhi Wang, Cong Yu, Simon Baumgartner, and Flip Korn. 2018. Relevant Document Discovery for Fact-Checking Articles. In *Companion Proceedings of the The Web Conference 2018*. 525–533. https://doi.org/10.1145/3184558.3188723

[33] Lingfei Wu, Ian En-Hsu Yen, Kun Xu, Fangli Xu, Avinash Balakrishnan, Pin-Yu Chen, Pradeep Ravikumar, and Michael J. Witbrock. 2018. Word Mover's Embedding: From Word2Vec to Document Embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 4524–4534.

[34] Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. Detection and Resolution of Rumours in Social Media: A Survey. *ACM Comput. Surv.* 51, 2, Article 32 (Feb. 2018), 36 pages. https://doi.org/10.1145/3161603

Privacy-Preserving Crowd-Sourcing of Web Searches with Private Data Donor

Vincent Primault
University College London

Vasileios Lampos
University College London

Ingemar J. Cox
University College London & University of Copenhagen

Emiliano De Cristofaro
University College London

ABSTRACT

Search engines play an important role on the Web, helping users find relevant resources and answers to their questions. At the same time, search logs can also be of great utility to researchers. For instance, a number of recent research efforts have relied on them to build prediction and inference models, for applications ranging from economics and marketing to public health surveillance. However, companies rarely release search logs, also due to the related privacy issues that ensue, as they are inherently hard to anonymize. As a result, it is very difficult for researchers to have access to search data, and even if they do, they are fully dependent on the company providing them. Aiming to overcome these issues, this paper presents Private Data Donor (PDD), a decentralized and private-by-design platform providing crowd-sourced Web searches to researchers. We build on a cryptographic protocol for privacy-preserving data aggregation, and address a few practical challenges to add reliability into the system with regards to users disconnecting or stopping using the platform. We discuss how PDD can be used to build a flu monitoring model, and evaluate the impact of the privacy-preserving layer on the quality of the results. Finally, we present the implementation of our platform, as a browser extension and a server, and report on a pilot deployment with real users.

ACM Reference Format:
Vincent Primault, Vasileios Lampos, Ingemar J. Cox, and Emiliano De Cristofaro. 2019. Privacy-Preserving Crowd-Sourcing of Web Searches with Private Data Donor. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313474

1 INTRODUCTION

Over the past few years, researchers have demonstrated how online search data can be used to infer contextual information related to both individual users and populations, with examples ranging from consumer behavior modeling [19], macroeconomic statistics [14], and sales forecasting [8, 40], to early detection of cancers [30, 39] and disease monitoring at a national level [18, 21, 32].

Typically, individual-level studies rely on acquiring access to the (anonymized) search history of multiple users. However, this prompts significant ethical and privacy issues: search history often contains sensitive data on topics such as income, sexual orientation, religion, and health. Furthermore, anonymization has proven to be very difficult to guarantee [17]. A case in point is the AOL search logs, which were released in 2006: several users were easily de-anonymized, leading to a public outcry that resulted in the resignation of AOL's CTO [44]. Therefore, search engine providers seldom, if ever, provide such data, even for research purposes.

On the other hand, estimating properties of a population may be done over aggregate data, i.e., how many users have searched for specific keywords over a time frame, which is much less sensitive. In particular, in this paper, **we focus on inferring population health statistics from Web query data**, and experiment with a use-case related to estimating the prevalence of a disease such as influenza [18, 21]. Unless the population and/or region size is very small, or the queries contain many outliers, aggregates do not permit de-anonymization. One possible data source for search statistics is Google, which offers services like Google Trends and the Health Trends API (see Section 2). However, it is still quite possible that these products will not be maintained or may cease in the future. Moreover, they might not provide information on specific keywords, or sufficient geographic granularity. Also, and perhaps more importantly, since aggregation is performed by a third party, it actually provides no privacy with respect to that party.

Aiming to fill this gap, **this paper investigates how to privately crowd-source Web search data.** To this end, a number of challenges need to be addressed, including: (i) recruitment and retention, (ii) human-computer interaction, and (iii) privacy. This paper primarily focuses on a technical solution to the privacy issue. In theory, a straightforward implementation to collect users' Web search data, without the search engine's involvement, would attempt to anonymize the user and transmit data to a trusted third party, which would then aggregate data before releasing it to a researcher. However, both steps involve significant risks to individuals' privacy, with de-anonymization and security breaches of the third party being the most obvious.

By contrast, we propose a solution that does not require a fully trusted third party, and which only collects aggregate statistics on search behavior at a population level. More specifically, **we present the Private Data Donor (PDD) platform**: PDD consists of a lightweight Web browser extension, which users (data donors) install, and that monitors the searches performed by them. Users are assigned to groups, and each group uses a protocol based on homomorphic encryption to encrypt and transmit the number of times a user searched for specific queries to a server. For each user in the group, the server adds their encrypted count together. The resulting sum is an unencrypted value representing the aggregate count of the group. Data from each group is then aggregated to

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW'19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313474

determine the counts for the entire population, so that at no time is the server able to determine the query frequencies of an individual.

Contributions. Overall, we make the following contributions:

(1) We propose PDD, a new platform to handle privacy-preserving crowd-sourcing of Web search queries, decentralized, and scaling up to millions of users.
(2) We analyze the impact of different system parameters on the behavior of our platform, aiming to find the best balance between the privacy of users and the utility of collected results.
(3) We study how our platform can be applied to a real-life use case, i.e., building a flu monitoring model.
(4) We implement and deploy our platform, and use it for real-life data collection.

2 RELATED WORK

Studies using Web searches. Search engines such as Google, Bing, DuckDuckGo, etc. respond to billions of queries every day [37]. Users ask search engines a wide range of questions, and these often include personal and contextually rich information. In 2016, Google reported that at least 1% of their searches were related to symptoms, and started to provide structured information about the health conditions associated to those symptoms [31]. Overall, Web searches can often be a treasure trove for researchers, with notable examples including the applications mentioned earlier [8, 14, 19, 40].

Of particular interest to us is research relying on search data for health purposes [18, 21, 30, 32, 39], and more specifically based on crowd-sourced information [22, 36, 38, 42]. In this paper, we essentially aim to introduce a privacy-preserving layer on top of those studies, which might encourage users to donate their data, and help researchers comply with regulations, while removing the dependence on limited statistics given by search engines.

Aggregate search data. Popular search engines do not have incentives to share search data, as this information is of commercial value and, as discussed earlier, sensitive in nature. Google provides two ways to gain insights into the searches it receives, at an aggregate level: (1) Google Trends and (2) the Health Trends API. The former is a Web interface offering a global, longitudinal view of the interest for one or more queries, provided as a normalized number between 0 and 100. The latter is a service offered to researchers, with access to the interest of search queries related to health topics (see also [23, 46]). Here, the interest is provided as a frequency, i.e., the number of times the queries were searched during a time period normalized by the total number of searches in the same period. Note that this is only computed from a subset of all searches (10%-15%), and various anonymization measures are taken to remove low-count searches. With the former, the number of times a query was issued is not made available—only some normalized measure, which does not allow evaluation of the absolute popularity of a query. Moreover, statistics are only provided at a country-level or state-level granularity. Finally, relying on these two platforms means that the researchers are directly dependent on a third-party company, whereas, we envision a decentralized scenario where researchers could be autonomous in collecting the data they need.

Privacy-preserving data aggregation. Researchers have also studied the generic problem of collecting statistics from users in a privacy-preserving manner. To this end, cryptographic schemes have been proposed for privacy-preserving data aggregation, e.g., in the context of smart grids [20], wireless sensor networks [4], and streaming data [5, 26]. Our work builds on the homomorphic encryption protocol proposed in [26], which we review in Section 3.2; however, since it was originally designed for real-time applications like counting viewers of streaming videos, we need to address a number of challenges to adapt it to our setting (see Section 3.3).

An alternative approach is to add noise to the statistics so that they do not leak information about any particular individual, thus guaranteeing Differential Privacy (DP) [12]. Typically, this approach relies on a central curator, which receives raw data, aggregates it, applies a differentially-private algorithm, and releases the protected version of the data to researchers (see for example [25, 27]). However, this central entity needs to be a fully trusted entity and as such could be vulnerable to attacks or a subpoena. Another possible setup is known as local differential privacy, where noise is applied at the user-side before sending the data, thus preventing the entity collecting data from learning anything sensitive [13]. However, this requires very large numbers of users in order for the noise to cancel out and yield reasonably accurate results [16].

Privacy-preserving crowd-sourcing. Finally, prior work specific to crowd-sourcing data from users includes systems allowing third-parties to gather Web analytics [7, 15], location statistics [33], and surveys [10]. Some also provide DP guarantees [1, 6, 13]. As opposed to this line of work, we aim to provide a distributed platform that does not rely on: (i) a central anonymization server and, (ii) a persistent connection between the clients and the server.

3 PRIVACY-PRESERVING AGGREGATION

This section presents our approach to privacy-preserving collection of aggregate data, a key component of our PDD platform. After providing a high-level overview, we formalize the underlying protocol. We then discuss how to adapt it to build a large-scale system for crowd-sourcing Web searches; finally, we present the enhanced version, which we release, open-source, for public use.

3.1 Overview

Terminology. Throughout the paper, we use the following terms:

- *Server:* in PDD, the server is a piece of software running for an institution interested in gathering Web searches statistics.
- *Users:* people willing to contribute their searches (in aggregate form) to a research project. They install a *client*, i.e., a browser extension communicating with the server. We use the terms "users" and "clients" interchangeably in the rest of the paper.
- *Query:* a string that can interpreted by a search engine (e.g., Google, Bing, DuckDuckGo) to retrieve matching Web resources, e.g., "world cup odds," "will it rain today," etc.
- *Monitored query:* a query for which the server collects and provides statistics.
- *Search:* a query that was sent by a user to a search engine. A search is thus the association of a user, a query, and a timestamp.
- *Analyst:* a researcher having access to the data collected by PDD and using it for further analyses.

Data Aggregation. We build on an underlying privacy-preserving data collection protocol – specifically, the one presented by Melis

et al. [26] – to create a fully-fledged crowd-sourcing platform for Web searches. The protocol encompasses three steps:

(1) *Setup:* Users willing to contribute their searches (in aggregate form) install a browser extension; upon installation, the extension generates a pair of cryptographic keys, registers itself to the server with its public key, and is assigned a client identifier.
(2) *Reporting:* Clients regularly obtain the set of monitored queries and the public keys of the other clients. They generate a (secret) blinding factor for each query, in such a way that the sum of blinding factors of all users, for a given query, will equal zero. Using their browsing history, clients fill a vector of counts, where each value corresponds to the number of times a monitored query was searched in the previous day: the vector is encrypted using the blinding factors and sent to the server.
(3) *Aggregation:* Once the server has received the encrypted vectors of all clients, it can decrypt the encrypted counts by summing them up.

Adapting Melis et al.'s protocol to our setting immediately prompts two main challenges. A first issue is the impact of a client being unable to report his encrypted data to the server; in this case, the sum of all blinding factors is *not* equal to zero, and the encrypted data cannot be decrypted by the server. In other words, a single user failing to report his data on a given day will result in the inability to retrieve the statistics for a particular day, which is not acceptable. Our PDD platform introduces two measures to build more reliability into the protocol: (1) We organize clients into smaller groups, each one running its own instance of the protocol; (2) We introduce some additional delay before "giving up," leaving more time for clients to send their data.

Another issue is that [26] requires a persistent connection between the server and the clients. This allows the server to learn when users are online, which may introduce a new privacy threat, and also increases the load on the server. We remove this requirement by letting clients communicate with the server at most once a day, which significantly lowers the privacy threat.

3.2 Privacy-Preserving Aggregation

We now formally present Melis et al.'s protocol [26].

Notation. First, let us introduce notation, also summarized in Table 1. We use N to denote the numbers of clients in the system, and L the number of monitored queries. $C_i, i \in [1, N]$ denotes the i-th client, and $W_j, j \in [1, L]$ denotes the j-th monitored query. Let $\mathbb{G}$ be a cyclic group of prime order q for which the Computational Diffie-Hellman problem [11] is hard, and g be the generator of the same group. Also, we use a cryptographic hash function, $H : 0,1^* \rightarrow \mathbb{Z}_q$, mapping strings of arbitrary length to integers. Let "||" denote the string concatenation operator, and $a \in_r A$ indicate that a is sampled at random for A.

Algorithms. The protocol involves the following algorithms:

- *Initialization.* Each client C_i generates a private key $x_i \in_r \mathbb{G}$ and a public key $y_i = g^{x_i} \bmod q$, and then sends its public key along with its identifier i to the PDD server to be registered.
- *Encryption.* At each round s, client C_i holds a counts vector $V_{si} = (v_j \in \mathbb{N}, j \in [1, L])$, where v_j represents the number of times the query W_j was searched for. To encrypt this vector, client C_i first generates a blinding factors vector:

Symbol	Meaning
$\mathbb{G}$	Cyclic group of prime order q, whose generator is g
H	Hash function mapping strings to integers
$\|\|$	String concatenation operator
s	Current round number (0-based)
N	Total number of clients
C	Set of all clients
C_i	i-th client, $i \in [1, N]$
(x_i, y_i)	Private and public keys of client C_i
L	Total number of monitored queries
W_ℓ	ℓ-th monitored query, $j \in [1, L]$
K_{si}	Blinding factors of client C_i at round s
V_{si}	True counts vector of client C_i at round s
$\hat{V}_{si}$	Encrypted counts vector of client C_i at round s
$o_s : C \rightarrow \mathbb{B}$	Whether client C_i was online during round s

Table 1: Notation.

$$K_{si} = \left(\sum_{j=1, j\neq i}^{N} H(y_j^{x_i} || \ell || s) \times (-1)^{i>j} \bmod q, \ell \in [1, L] \right) \quad (1)$$

where $(-1)^{i>j} = -1$ for $i > j$ and 1 otherwise. Then, each C_i encrypts its counts vector V_{si} into an encrypted vector $\hat{V}_{si} = V_{si} + K_{si}$, which is then to be sent to the server.

- *Decryption.* Once the server has received the encrypted vectors for all clients at round s, it can decrypt it by summing them:

$$V_s = \sum_{i=1}^{N} \hat{V}_{si} = \sum_{i=1}^{N} V_{si} + \sum_{i=1}^{N} K_{si} = \sum_{i=1}^{N} V_{si} \,. \quad (2)$$

Indeed, the sum of all K_{si} is a zero vector:

$$\forall \ell \in [1, L], \sum_{i=1}^{N} K_{si_\ell} = \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j\neq i}}^{N} H(y_j^{x_i} || \ell || s) \times (-1)^{i>j} = 0 \,. \quad (3)$$

3.3 Challenges

As mentioned above, using Melis et al.'s protocol [26] in our setting (i.e., crowd-sourcing Web searches) poses a few challenges, as it was designed for a different set of applications, namely, counting the viewers of streaming videos or the number of requests to Tor Hidden Service directories. In other words, [26] targets real-time data collection in a context where users are expected to all be online at the same time and the cryptographic code actually runs directly in the browser in Javascript (specifically, as a Node.js package). At some point, the server decides to gather statistics, issues a request to the clients to encrypt and send their counts, and decrypts/publishes the aggregate results after decrypting the clients' contributions.

However, if some clients disconnect during the process, e.g., the user goes offline or closes the browser tab, the server will not receive some of the encrypted counts, and cannot decrypt the aggregate counts since the blinding factors no longer sum up to zero. Also, the protocol requires a persistent connection between clients and server (e.g., via a web-socket or a regular ping). We discuss these two challenges in more detail next.

Disconnections. To mitigate the issue of disconnections, [26] proposes two solutions: (1) organizing users in smaller groups, and (2) running a recovery protocol. Specifically, the protocol can be applied independently on several groups, and the decrypted count vectors of each group summed to obtain the global decrypted count vector. In this setting, the size of each group becomes a critical parameter, impacting both privacy (i.e., among how many other

users is each user hidden) and utility (i.e., the likelihood the group's count vector can be decrypted). Put simply, the larger the group, the better the privacy, but the more at risk is utility. Another open question, which was not handled in [26], is how to organize users into groups. At the same time, the recovery protocol allows the users that are still online to compute a recovery factor, which will then be used to "reverse" the effect of the blinding factors of missing users. However, users might in turn disconnect during this recovery protocol, making the recovery protocol fail in turn.

Persistent connection. In our setting, the need for a permanent bi-directional connection between the server and its clients is very undesirable, for two main reasons. First, having a permanent connection between the server and every client requires non-negligible computing resources, while we need the platform to be as operationally simple as possible, allowing researchers to deploy it on a single server with limited computing power. Second, this also raises orthogonal privacy issues, as the server will know the times at which users are online, learning information about users' behavior which can potentially to be a side-channel for privacy attacks. In theory, privacy-preserving presence protocols have been proposed (e.g., [3]), but they rely on having multiple non-colluding servers, which we do not have.

Requirements. Overall, as mentioned, Melis et al.'s protocol [26] was designed for different applications, while we are interested in collecting Web searches. This distinction has two practical implications in the requirements of our platform: (1) Our client-side platform needs to access browsing history to collect the search queries (specifically, the monitored queries), thus, it needs to run as a browser extension (and not directly on a particular web page); (2) We do not need real-time statistics, but rather daily aggregates.

Next, we discuss how we modify the platform to accommodate our requirements and overcome these challenges.

3.4 Enhanced Aggregation Protocol

The main idea is that the server prepares itself at the beginning of every day to gather statistics about the past day. To do so, it precomputes, each night, groups of clients based on their past activity patterns. Then, throughout the day, when they come online, clients ask the server which groups they are part of, and retrieve the list of monitored queries. Each user is only part of at most one group each day, but the server may return several groups for the past few days, if necessary to recover from past days where the user was inactive. Clients then use their browsing history to compute the counts vector for each requested group, encrypt it using the public keys of the other members of the group, and send the encrypted counts vectors to the server. In this setup, the round number s becomes the number of days elapsed since the data collection began.

Handling disconnections. As proposed in [26], we organize clients into groups. However, as we only have at most one contact with each client per day, the challenge is to be able to pre-compute groups without knowing for sure which users will be active the next 24 hours. To this end, we leverage historical knowledge about past clients' activity. More specifically, at the beginning of day d, when statistics about day $d-1$ need to be gathered, the server knows which clients were active during day $d-1$. Therefore, the server includes only the clients active on day $d-1$ in the groups formed on day d, which reduces the number of clients actually involved in the protocol every day, and improves the reliability of the data collection. Indeed, the less users are involved in the protocol, the lower the probability that some of them will not be active on day d and hence the lower the probability to lose data.

Creating groups introduces, obviously, the need for setting a *group size*, which is a system parameter controlling the number of clients into each group. Concretely, groups formed at the beginning of day d are composed of only clients that were active on day $d-1$, and affected to groups in a round-robin fashion. All groups will hence have the same number of clients, except one group which may contain less clients. We will explore, in Section 4.4, a more advanced strategy, selecting users by their past activity, trying to maximize the probability for them to be online.

Nonetheless, the main drawback of pre-computing groups is that there is still a risk that some clients involved in the protocol will not be active the next day, which may prevent the encrypted data from some groups to be decrypted. For that reason, we introduce the notion of a *delay*, i.e., a number of days during which the server will wait for the encrypted counts to be sent by the clients. Waiting a few more days will allow the inactive clients to come online later on, thus improving the quality of the results.

Removing persistent connections. Because clients only contact the server at most once a day, the privacy threats are dramatically lowered as compared to having a persistent connection, because the server does not anymore know, in real-time, which users are active. It also significantly lowers the computational pressure on the server, which is only expected to handle a couple of requests from each client every day. (Rate limiting strategies should also be used to protect against DDoS attacks [43].)

Confidence. It might still happen that a client does not come online on day d, and neither in the additional delay that may be granted. In that case, it means that the encrypted data for this client, and for all this group, will never be decrypted and is considered as *lost*. Although lost data should be considered impossible to recover at that point, we can provide a *confidence* measure to the data analysts, computed as the ratio of successfully decrypted users. This adds transparency to the figures provided to them by the PDD platform.

Threat Model. The security of this approach relies on that of the aggregation scheme presented by Melis et al. [26], which is proven secure in the honest-but-curious (HbC) model under the Computational Diffie-Hellman assumption [2]. Here the HbC model means that both the clients and the server, although they may attempt to violate privacy of their counterparts, do not deviate from the protocol specification. For instance, malicious users could report fake values in order to invalidate the final aggregation values, while the server could maliciously manipulate the number and identity of clients in the same group. This is a common assumption in cryptography, which can be mitigated using verifiable tools such as zero-knowledge proofs, cryptographic commitments, and code attestation. We leave such extensions as part of future work, as they are orthogonal to the challenges we face in this work.

3.5 Example

To clarify how our platform works, we use an example, as pictured in Figure 1. We consider five clients (A, . . ., E), who participate in

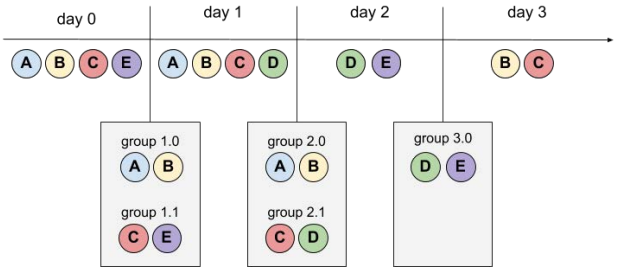


Figure 1: Example of how users can be added to groups based on their past activity.

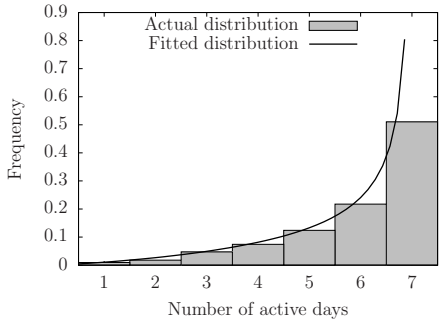


Figure 2: Distribution of users' activity.

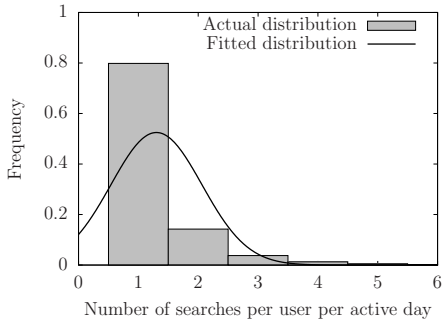


Figure 3: Distribution of search volume.

data collection, a group size of two, and a delay of one additional day. At the beginning of day 1, A, B, C and E, who were seen online on day 0, are put into two groups of two clients each. On day 1, A, B and C were seen online, as well as client D, who is not part of any group. Then, A, B and C will be tasked with sending their encrypted count vectors to the server, while D will not have any task to perform. At the end of day 1, data of group 1.0 has been decrypted, but data of group 1.1 is still missing E's counts vector.

At the beginning of day 2, A, B, C, D (who were seen online on day 1) are put into two groups. At the end of day 2, because E has eventually come online within the one additional day of delay, data of group 1.1 has been decrypted. However, groups 2.0 and 2.1 are still un-decrypted, because A, B and C were not online on day 2. At the end of day 3, group 2.1 has been decrypted within the one-day delay, but group 2.0 is lost since A was not active on days 2 and 3.

Eventually, data for day 0 has been made available at the end of day 2 (with a confidence of 100%), while data for day 1 has been made available at the end of day 3 (with a confidence of 50%). Overall, this simple example illustrates the usefulness of both dividing users into groups (allowing to decrypt half of the data for day 1) and the additional delay (allowing to decrypt all, instead of only half, of data for day 0).

4 CONFIGURING PDD THROUGH SIMULATION

We use extensive simulations to determine the configuration of the PDD platform. Specifically, we evaluate the impact of the two main system parameters (see Section 3.4): the group size and the delay. As PDD is a distributed system, potentially involving thousands or even millions of users, we use simulation to better understand its behavior and test different parameterizations.

4.1 The PDD Simulator

We now introduce the simulation environment used in our experiments. It involves two distinct modules, simulating: (1) user activity, and (2) the PDD protocol itself. The former simulates how users interact with their browser, while the latter simulates the interaction between clients and the server. The simulator is implemented in Python (for data pre-processing and analysis) and Scala (for the main workflow).

User activity module. The user activity module is made of three distinct parts, each simulating: (i) connectivity, i.e., which days users are active and using their browser, (ii) search volume, i.e., how many searches users are performing on active days, and (iii) search content, i.e., what are users actually looking for.

Overall, accurately simulating users' browsing activity is a very challenging task, mainly due to the lack of publicly available datasets. To overcome this challenge, we turn to the datasets released by Mozilla in 2010 to shed light on how users interact with their browsers [28]. We use the most complete dataset, which contains data for 4,802 users, collected over a week in December 2009. For each user, we have the number of browsing sessions (i.e., how many times each user started his browser during the week) and the average duration of those browsing sessions. We use the data to "replay" such sessions since, unfortunately, the dataset does not contain the *times* at which users were active. We obtain a distribution with the number of days (out of seven) during which the users have been effectively using their browser. Due to the randomness of this process, we run 100 iterations, and eventually select the Beta distribution with the lowest distance, according to a Kolmogoroc-Smirnov test [24]. More specifically, we end up with a Beta distribution with $\alpha = 2.2170$ and $\beta = 0.4634$.

The original frequencies and the fitted distribution are shown in Figure 2. Note that the fitted distribution has a mean of 82.7% – i.e., on average, users are using their browsers between 5 and 6 days every week. This is consistent with a more recent study from Mozilla [9], which reports how long Mozilla's data scientists have to wait for telemetry data to be available, estimating that, on average, 95% of telemetry data was received within a single day. For each user, we sample a number of active days from the Beta distribution, scaled to the total number of days considered in our experiments (rounded to the closest integer). Simulations in Section 4 involve 100k simulated users over 100 days.

Next, we simulate search volume using the Yahoo dataset [35], which provides search logs collected over a month (July 2010), featuring 29M users, 67k unique queries and 80M searches. From this dataset, we extract the daily number of queries performed by each user, and fit a normal distribution on it ($\mu = 1.3020$, $\sigma^2 = 0.7603$). Figure 3 reports the original data along with the fitted distribution. For every day a user is active, we sample a number of searches to be performed from the distribution (rounded to the closest integer). Finally, we simulate searches by extracting the frequency of each of query included in Yahoo's dataset, which spans over an entire month. Note that the data is truncated after more than 10k searches of a given query, which may limit the precision of the simulation for very frequent queries. However, it still allows us to capture that a few queries are searched very often (e.g., 2k queries more than 10k times), with a long tail of infrequent queries (e.g., 11k queries less than 10 times). We simulate searches by randomly sampling a query from this distribution, as a random choice weighted by the frequency of each query.

PDD protocol module. This module trivially simulates the behavior of the PDD server by implementing the protocol described in Section 3.4. We assume that network communication is reliable (using TCP), thus, we do not need to simulate network communications and failures. Because we rely on simulation instead of a real-life system, we also have access to the ground-truth data (i.e., the V_{si}'s, the counts vectors of each client at each round), which we can use to evaluate our solution. The simulation essentially produces as output, for every day, a report about the true count and estimated count (i.e., after decrypting data) of each query, the total number of searches and the confidence.

Evaluation metric. We use the mean relative error (MRE) of the decrypted counts as the main metric to evaluate the quality of the collected data, averaged out across all queries, all users, and all rounds. Formally, it is defined as:

$$\text{MRE} = \frac{1}{|\{V_{si_\ell} \neq 0\}|} \sum_{V_{si_\ell} \neq 0} \frac{V_{si_\ell} - \tilde{V}_{si_\ell}}{V_{si_\ell}}, \tag{4}$$

where $\tilde{V}_{si}$ is the estimated counts vector received by the server for client C_i at round s. In practice, $\tilde{V}_{si} = V_{si}$ if the data of client C_i was successfully decrypted at round s, or $\tilde{V}_{si} = (0, \ldots, 0)$ otherwise.

4.2 Group Size

We start by using simulations to study the impact of the group size. First, users are put into groups following the heuristic described in Section 3.4, considering only the users who were online the previous day, i.e., at day d, $\{C_i \in C | o_{d-1}(C_i)\}$.

Group size is inherently responsible for the trade-off between privacy and utility. Here the level of privacy provided to the users is akin to k-anonymity [12], meaning that every client part of an aggregate will be hidden among $k-1$ other clients being part of the same aggregate, whereas, utility is measured via MRE as discussed above. On the one hand, creating groups of very few clients would yield near-optimal utility, but at the cost of providing little or no privacy to the users. On the other hand, putting *all* clients into a single group would provide the best levels of privacy, but can dramatically decrease the utility, since even a single client not sending its data will prevent the decryption of everybody's data. It also means that system designers have no control on the impact that increasing/decreasing numbers of clients will have.

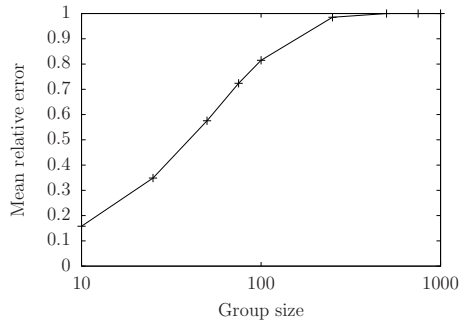


Figure 4: Impact of the group size on utility.

Therefore, we seek to find a middle ground, providing reasonable privacy guarantees, while still allowing reported counts to be accurate enough for data analysts to work with it. In Figure 4, we report the impact of the group size on the MRE of decrypted counts with a delay of two days. We report results with group sizes between 10 and 1000, as suggested in [26]. We note that MRE varies between 16% and 100%, increasing quite quickly: a group size of 100 already yields an 81% MRE. The privacy-utility trade-off is pretty clear: groups should not be too small because of privacy (in fact, we would not recommend less than 50 clients per group), however, large group sizes do not seem very practical, e.g., MRE is 100% with 750 users per group. Therefore, in Sections 4.3–4.5, we will experiment with group sizes between 50 and 500, and study how we can improve the utility without degrading privacy.

4.3 Delay

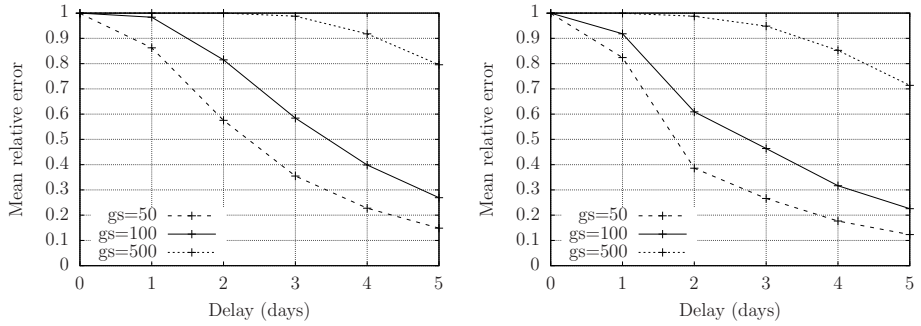
Next, we study the impact of utility when waiting for a few more days to provide aggregate counts to the analysts. As opposed to the group size, the delay does not play a central role in the privacy-utility trade-off, as it does not impact the privacy. Instead, it imposes a trade-off between utility and the availability of the results. Figure 5a shows the evolution of MRE with delay varying between 0 and 5 days, for group size $gs \in \{50, 100, 500\}$. We note a significant positive impact on MRE starting with a delay of two days.

Choosing the right delay is inherently application-dependent: some use-cases might accommodate well delays of a week, while others need near-real-time results. Nevertheless, Figure 5a shows that a group size of 500 is still not very practicable, because the MRE remains around 80% in the best case, while groups of 50 clients with 5 days of delay reduces MRE to 15%.

Our simulations also highlight that some errors are not recoverable, even while waiting up to five additional days. We believe this is due to the behavior of users, as shown in Figure 2, with a very small fraction of the users being active only once in a while. Next, we study how to mitigate the impact of inactive users on MRE.

4.4 Improving Client Selection

Aiming to improve the privacy-utility trade-off presented above, we present a novel approach to organize clients in groups. The basic idea is to only put into groups clients that are likely to be active the next few days, assuming that some additional delay is allowed.



(a) Naive approach. **(b) Improved selection algorithm.**

Figure 5: Impact of the delay on utility.

Indeed, we want to maximize the probability that all users inside groups will be active in the next days, even if it implies losing out on the data of some users that are generally not very active.

Ideally, one would rely on machine learning techniques to learn user behavior patterns. However, because of the lack of real-life browser usage traces, this approach is not viable since we would not be able to effectively train prediction models. Therefore, we set to improve our heuristic to select users to put into groups more effectively. As shown in Figure 2, while most users are active every day, there is a non-negligible tail of users that are much less active.

Basically, if we have a delay of δ days, we want to only select users that are active on average at least one out of δ days. More formally, at the beginning of day d, when the delay is configured to δ days, we only select users satisfying the following constraint:

$$\left\{ C_i \in C | o_{d-1}(C_i) \wedge \frac{|\{d' \in [0, d-2], o_{d'}(C_i)\}|}{d-1} \geq \frac{1}{\delta} \right\} . \quad (5)$$

The above criterion is applied when $\delta > 0$ and $d > \delta$, meaning that we effectively have some additional delay and have accumulated enough data about users' past behavior to decide whether it is worth including them into a group.

Figure 5b shows the impact of this new heuristic on MRE. When compared to Figure 5a, we observe a noticeable decrease in the error. For instance, with a group size of 100 and 2 days of delay, the MRE goes from 0.82 to 0.60 (a 26% improvement). In fact, the best improvements with the heuristic are obtained with 2 days of delay. With 1 day of delay, we do not have enough past activity history to yield enough impact, while further increasing the delay is sufficient to lower MRE without the need for our heuristic.

4.5 Building Confidence in the Platform

Finally, we set to provide a confidence metric to the analysts in order to capture the expected quality of the aggregate statistics. Essentially, we use the percentage of users that were successfully involved in the data collection process. That is, we define confidence as the ratio of clients whose data was successfully decrypted and the total number of clients who were active on the target day. For example, a confidence of 60% for day d means that we were able to decrypt data for 60% out of all clients who were active that day. Recall that the missing counts can be due to clients that never sent their encrypted counts to the server, or by clients who were part of a group where a client never sent their data.

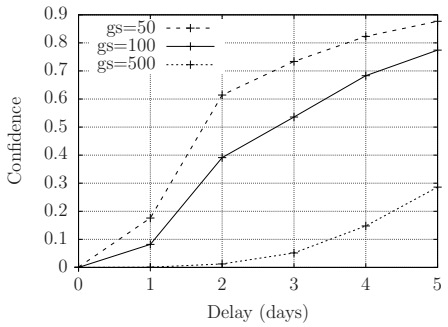


Figure 6: Confidence levels, i.e., ratio of clients whose data was successfully decrypted, for increasing delay.

In Figure 6, we plot the evolution of the confidence depending on the delay and the group size. Unsurprisingly, this is closely related to MRE (cf. Figure 5b), i.e., a high MRE is associated to low confidence. However, while MRE can only be computed through simulation (since it relies on the ground truth), confidence can actually be computed in real production environments. Therefore, we believe it can constitute a simple yet effective tool for analysts to understand the accuracy of the statistics at their disposal. However, it is not so easy to understand the actual impact of MRE, and what is the threshold differentiating between a "good enough" and a "bad" MRE. In the next section, we will experiment with the PDD platform on a real-life use-case, and study the actual impact of the privacy "layer" on the accuracy of the results obtained by analysts.

5 BUILDING A SYNDROMIC SURVEILLANCE MODEL FOR INFLUENZA-LIKE ILLNESS

In this section, we experiment with a health-related use-case, showing how data collected through PDD can be used to monitor the prevalence of influenza-like illness (ILI). By using an established model for this task that provides a significant level of accuracy [21, 22], we are particularly interested in quantifying the impact that introducing PDD may have on its performance.

Flu season	True query frequencies			Estimated query frequencies		
	MAE	RMSE	r	MAE	RMSE	r
2014/15	2.5211	3.7022	0.8631	3.0163 (0.6112)	4.0382 (0.6251)	0.8574 (0.0211)
2015/16	2.1023	2.8766	0.9439	2.4415 (0.1949)	3.1805 (0.2586)	0.9202 (0.0140)
2016/17	2.1907	3.3163	0.9165	2.0212 (0.2980)	2.6704 (0.3501)	0.9020 (0.0313)
Avg.	2.2714	3.2984	0.9078	2.4930	3.2964	0.8932

Table 2: Performance of the ILI rate estimation task

5.1 Methodology

Starting from the simulator described in Section 4.1, we modified the users' activity module to sample the searches from a set of actual flu-related queries. More specifically, we use data from Google Health Trends to accurately model the behavior of users when interacting with the Google search engine. We retrieve the daily frequencies of 1k flu-related queries from January 1, 2008 to December 31, 2017 (10 years). We follow the approach presented in [23] to extract and filter queries based on their semantic relationship with the topic of flu. Note that we only have access to frequencies (and not absolute counts), i.e., the number of times a query was issued over the total number of queries issued that day. For every query, we fit a frequency distribution using a maximum likelihood estimation. We split the queries in two classes: sparse queries, which are fitted to a beta distribution, and denser queries, fitted to a generalized extreme value distribution. During the simulation, we use those distributions to sample, for each query and every day, a frequency.

In total, the added frequencies of the 1k queries of interest account on average for about 0.34% of the daily number of searches. This means that, as opposed to the previous behavior, the large majority of the searches that are generated are not of interest for influenza modeling. Consequently, we run a much larger-scale simulation in order to collect enough search samples, even for sparse queries; to this end, we simulate 1 million users over 20 years, which yields 825k samples for the 1k queries of interest. We use a delay of 2 days and a group size of 100 clients.

We then use the outputs of the simulation to compute daily query frequency estimates, which are then compared with their corresponding true values. We use a relative error metric, i.e., $(f_t - f_e)/f_t$, where f_t, f_e, denote the true and the estimated query frequency, respectively. Note that, on the frequency space, errors may be positive as well as negative. Given that positive and negative errors have different properties, we model them separately using two Weibull distributions per query (different distribution parameters for each query). Moreover, we capture the overall tendency for positive versus negative errors during daily data extracts, modeled by a Gaussian distribution. We filter out queries with less than a 100 nonzero true frequency estimates from this process, as their error samples are very sparse (466 from 1k queries were kept).

Finally, we use the fitted error distributions to introduce errors to true query frequencies from Google Health Trends. We rely on a two-step sampling process, where first the error signs are drawn for all queries for particular day, and then for each query, based on its allocated error sign, a relative error is drawn from its positive or negative error distribution. We deploy 50 sign draws per day, and for each one 50 error distribution draws per query, which generates 2,500 query frequency data sets where errors have been propagated.

5.2 Evaluating the Impact of PDD

Next, we set to assess the effect of the errors discussed above in the context of a practical task. To this end, we used the estimated query frequency datasets as inputs for modeling ILI rates in England – a well-studied task in the literature [18, 21, 23, 41].

Model structure. Disease rate estimation from online search data is commonly formulated as a regression task [18, 21]. The aim is to learn a function g: $\mathbf{X} \rightarrow \mathbf{y}$ that maps the input space of search query frequencies, $\mathbf{X} \in \mathbb{R}^{n \times m}$, to the target variable, $\mathbf{y} \in \mathbb{R}^{n}$, representing disease rates; n denotes the number of samples and m is the number of unique search queries we are considering. The time interval for each sample is set to a week to match the frequency of syndromic surveillance reports. Here, we use elastic net [47] as our regression function, similarly to previous work on the topic [21, 22]. Elastic net combines ℓ_1-norm and ℓ_2-norm regularization to obtain sparse solutions and to address model consistency problems that arise when collinear predictors exist in the input space [45]. It is formulated by the following optimization problem:

$$\underset{\mathbf{w},\beta}{\operatorname{argmin}} \left(\|\mathbf{y} - \mathbf{X}\mathbf{w} - \beta\|_2^2 + \lambda_1 \|\mathbf{w}\|_2^2 + \lambda_2 \|\mathbf{w}\|_1 \right), \tag{6}$$

where $\lambda_1 > 0$, $\lambda_2 > 0$ are respectively the ℓ_1-norm and ℓ_2-norm regularization parameters, and β denotes the intercept term.

Metrics. We assess the performance of elastic net models in estimating ILI rates in England during three flu seasons (2014–2017). Test periods are 1-year long (52 weeks) and training sets include all data prior to the corresponding test period. Prior to applying elastic net, we filter out queries with a Pearson correlation < 0.3 with the ground truth (per training set). We measure performance using mean absolute error (MAE), root mean squared error (RMSE), and Pearson correlation (r).

Results. We report performance figures in Table 2. Overall, we see that the average performance (based on the 2,500 trials) obtained by the estimated query frequency data does not differ much to the one obtained by the error-free data. There is a 9.7% increase in MAE, a 1.6% decrease in correlation, while RMSE remains unaffected. Figure 7 depicts ILI rate estimates across all the test periods in comparison to ground truth data. We plot the mean ILI rate across the 2,500 trials and use two standard deviations as a confidence interval. There are minor differences in the estimates of the privacy-preserving and true frequency models, although overall the time series appear as visually similar.

Overall, our analysis shows that both provide good estimates of the actual ILI rates, which serves as a practical indication that the error introduced by the privacy-preserving scheme does not severely affect the utility of the search data.

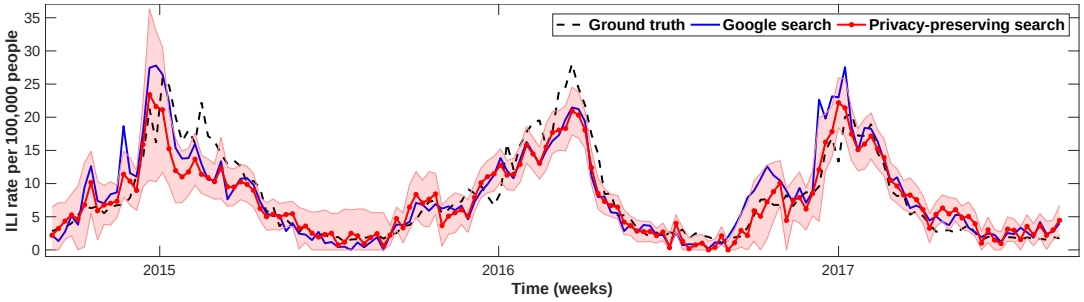


Figure 7: ILI rate estimates based on true and estimated (privacy-preserving) query frequency values compared to the ground truth. Confidence intervals depict two standard deviations from the mean for the 2,500 drawn query frequency estimates.

6 REAL-WORLD DEPLOYMENT

As mentioned earlier, we have implemented PDD as a fully-functional platform, which is freely available to health researchers. In this section, we detail our prototype, presenting its architecture as well as a real-world experiment with it.

6.1 System Architecture

PDD follows a classical client-server architecture, as depicted in Figure 8. The client-side is deployed to each user as a browser extension for Google Chrome/Chromium. The server is hosted by the research institution in charge of running the data collection campaign. A Web dashboard is also available to administrators and data analysts, allowing them to tune the data collection parameters and to export the results once they are available.

Server. The PDD server is the central component of our architecture, implementing three core services. First, it provides the concept of a *campaign*, which corresponds to a set of queries being monitored over some period of time. Several campaigns can be running independently in parallel, allowing different research teams to rely on a common crowd-sourcing infrastructure for their research. Second, the server is in charge of regularly initializing empty count vectors. Once a day, it scans the active campaigns and clients and organizes the latter into groups according to the algorithms described in Section 4. One or several empty count vectors are then initialized for each client that is expected to contribute some data to one or more campaigns. Each time a client pings the server, the latter will look for all those yet-to-be-filled vectors, and asks the client for the missing data. Third, the server is responsible for the aggregation of count vectors. Once a day, this service collects the encrypted vectors sent by the clients, decrypts them whenever possible, and updates the global count vector for the target day.

Each campaign has its own system parameters (i.e., delay and group size) as well as a distinct list of queries of interest. Note that, while Section 3 only describes the ability of monitoring queries, our implementation can monitor both keywords as well as queries. That is, by splitting every search around whitespaces, we can extract tokens and monitor the searches that contain them or combinations of them. For instance, instead of monitoring the query "symptoms of flu," an analyst might choose to track the keywords "symptoms" and "flu," allowing to match different variations such as "flu symptoms."

Communication between the clients and the server is through a REST API, secured by TLS. The REST API itself is divided in

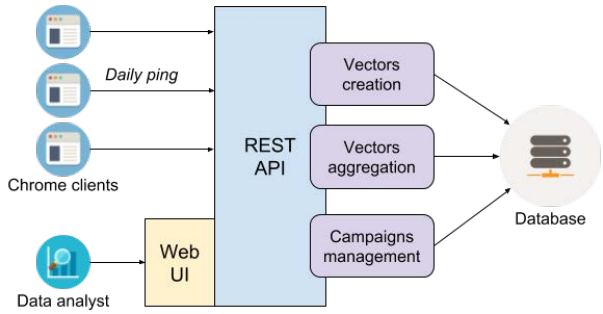


Figure 8: Architecture of the PDD system.

two parts: a public and a private APIs. The former is accessible to the clients and essentially contains three endpoints: (i) client registration, used when a Chrome extension is installed, (ii) ping, used by clients to get their instructions, and (iii) vector submission, used to update the vectors back as instructed by the server. The latter is only accessible to the analysts and is used for administrative operations. The server is written in Scala (a language running on the Java Virtual Machine), while the dashboard in Javascript. The data is stored in a MySQL database, although the persistence layer is designed to be pluggable to accommodate other storage systems.

Client. The client-side is implemented as a Google Chrome extension, although it can be ported to other browsers, such as Firefox, with minimal effort. The extension includes 733 source lines of code written in Javascript. The main goal of the extension is to contact the server on a daily basis (whenever possible) via a ping. The response of a ping is a set of instructions about some information to compute in order to fill empty count vectors. Each instruction contains a set of monitored queries, the period of interest (typically a day), the round number and the public keys of the members of the group. The client is then in charge of computing the count vectors, thanks to the API provided by Chrome, by locating the Google searches from their URL, extracting the query, tokenizing it into keywords and aggregating them with respect to the set of monitored keywords. This vector is then encrypted, using the public keys provided and round number, and sent back to the server.

The browser extension also has a lightweight interface for the user, providing transparency about the collection process. Notably, it allows the user to see which searches are going to be sent to the PDD server at the beginning of the next day. If the user does not want some of this data to be contributed (even though it is aggregated and encrypted), he has the option of creating a blacklist.

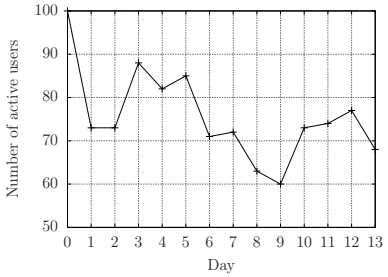


Figure 9: Daily activity during the real-life experiment.

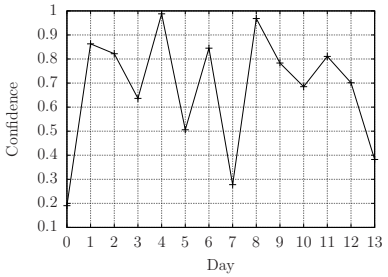


Figure 10: Daily confidence during the real-life experiment.

The latter contains a list of keywords that the user does not want to be sent to the server, permanently.

Source Code. The PDD platform is available in open source, under the GNU General Public License v3, on GitHub.[1] The Chrome extension is also available on the Chrome Web Store.[2]

6.2 Real-World Experiment

In order to experiment with our prototype and real-life users, we used the Prolific.ac [29] website to recruit a pool of (paid) volunteers. Our study received ethics approval from University College London. In total, we ran two studies. The first one aimed to validate that the system was actually working and reliably collecting data, and involved 10 participants for a week. A couple of months later, we launched a larger-scale study involving 100 participants over two weeks. In the following, we report on the latter.

Participants were asked to install our Chrome extension and leave it active, but were not otherwise asked to alter their behavior in any way. We only required them to be active for at least 10 days (out of 14 in total), in order to prevent dishonest participants claiming a reward without actually contributing any data. Because the season was not favorable to experiment with flu-related searches (it was done during summer 2018), we instead monitored the usage of the 10k most common English words.[3]

User behavior. To analyze the activity of the participants, in Figure 9, we show the number of daily active users throughout our experiment. The plot highlights that there are indeed variations in the number of daily active users, varying between 60% and 90%. This also confirms that we cannot realistically expect, even with as little as 100 paid users, that all users will be active on a single day. In fact, on average, users were active 10 days out of 14. Overall, we detected a total of 3000 to 5000 keywords per day. This rather high number is due to the fact that we are monitoring the 10k most common English words, meaning that each search will likely generate several keywords, one for each of the common words in it.

Error. Because we had a small number of users, we used a group size of only 10, fixing the delay to 2 days. Figure 10 plots the confidence throughout the 14 days of our experiment. The average confidence is 71.3%, even though there are significant variations. Note that the results for the first day exhibit very low confidence, because of the participants who left the experiment and were never active again. The confidence for day 7 is also low at 28%; it corresponds to a turning point between both a peak of active users at day 5 and the lowest number of active users at day 9 (cf. Figure 9), meaning the users' future activity was overestimated when creating groups for day 7.

Performance. The PDD server is packaged as a Docker image and deployed on a single virtual machine equipped with 4×2.1 GHz and 16 GB of RAM. Over the 2 weeks, the user CPU usage was on average 0.2%, while the memory consumption was on average 2.1GB. We did not experience any crash, and the resource usage leaves plenty of room to grow to more users.

7 CONCLUSION

This paper introduced PDD, a platform enabling researchers to access crowd-sourced Web searches, collected in a privacy-friendly way. We built on an existing cryptographic protocol for privacy-preserving data aggregation [26], adapted it to our settings, and evaluated our solution through extensive simulations. In particular, we highlighted the importance of two parameters, delay (i.e., the number of days to wait for the results) and group size (i.e., the anonymity of users). We then validated the suitability of our approach on a real-life use case, building a model monitoring the prevalence of influenza from Web searches. We compared the quality of the model produced when using PDD to both ground truth and the model obtained from *raw* Google searches, and observed very similar performance. Finally, we deployed our PDD prototype in the wild and ran a small pilot deployment involving 100 paid participants, achieving a reasonable confidence (more than 70%) while keeping resource requirements low.

Although we focus on health-related use cases, we believe that PDD can accommodate other contexts. Overall, we envision the opportunity to build a fully-fledged privacy-preserving analytics platform on top of this work, paving the way to more use cases while enforcing strict privacy guarantees. Our experimental evaluation showed opportunities to improve even more the way we select and organize users into groups; we believe that leveraging machine learning, in addition to our heuristics, could greatly increase the confidence while lowering error. In future work, we also plan to study the feasibility of possible inference attacks against aggregate statistics (e.g., membership [34]), and mitigate them using differentially private mechanisms.

Acknowledgments. This work has been supported by the EPSRC grants EP/K031953/1 and EP/R00529X/1. We also wish to thank Google for providing access to the Health Trends API and Lukasz Olejnik for early work on the PDD prototype.

[1] https://github.com/pddisense/pdd
[2] https://chrome.google.com/webstore/detail/private-data-donor/ipeekohlgfhagcopnndkgoommcihmdmk
[3] https://github.com/first20hours/google-10000-english

REFERENCES

[1] I. E. Akkus, R. Chen, M. Hardt, P. Francis, and J. Gehrke. Non-tracking Web Analytics. In *ACM CCS*, 2012.

[2] F. Bao, R. H. Deng, and H. Zhu. Variations of Diffie-Hellman problem. In *ICICS*, 2003.

[3] N. Borisov, G. Danezis, and I. Goldberg. DP5: A Private Presence Service. In *PoPETS*, 2015.

[4] C. Castelluccia, E. Mykletun, and G. Tsudik. Efficient Aggregation of encrypted data in Wireless Sensor Networks. In *Mobiquitous*, 2005.

[5] T.-H. H. Chan, E. Shi, and D. Song. Privacy-preserving stream aggregation with fault tolerance. In *Financial Cryptography*, 2012.

[6] R. Chen, I. E. Akkus, and P. Francis. SplitX: High-performance Private Analytics. In *SIGCOMM*, 2013.

[7] R. Chen, A. Reznichenko, P. Francis, and J. Gehrke. Towards statistical queries over distributed private user data. In *NSDI*, 2012.

[8] H. Choi and H. Varian. Predicting the Present with Google Trends. *Economic Record*, 88(s1), 2012.

[9] :chutten. Two Days, or How Long Until the Data is In. Online at https://blog.mozilla.org/data/2017/09/19/two-days-or-how-long-until-the-data-is-in/, 2017.

[10] H. Corrigan-Gibbs and D. Boneh. Prio: Private, Robust, and Scalable Computation of Aggregate Statistics. In *NSDI*, 2017.

[11] W. Diffie and M. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6), 1976.

[12] C. Dwork. Differential Privacy. In *ICALP*, 2006.

[13] U. Erlingsson, V. Pihur, and A. Korolova. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *ACM CCS*, 2014.

[14] M. Ettredge, J. Gerdes, and G. Karuga. Using Web-based Search Data to Predict Macroeconomic Statistics. *Communications of the ACM*, 48(11), 2005.

[15] L. Fan and H. Jin. A Practical Framework for Privacy-Preserving Data Analytics. In *The World Wide Web Conference*, 2015.

[16] G. Fanti, V. Pihur, and ÃŽlfar Erlingsson. Building a RAPPOR with the Unknown: Privacy-Preserving Learning of Associations and Data Dictionaries. In *PoPETS*, 2016.

[17] A. Gervais, R. Shokri, A. Singla, S. Capkun, and V. Lenders. Quantifying Web-Search Privacy. In *ACM CCS*, 2014.

[18] J. Ginsberg, M. H. Mohebbi, R. S. Patel, L. Brammer, M. S. Smolinski, and L. Brilliant. Detecting influenza epidemics using search engine query data. *Nature*, 457(7232), 2009.

[19] S. Goel, J. M. Hofman, S. Lahaie, D. M. Pennock, and D. J. Watts. Predicting consumer behavior with Web search. *Proceedings of the National Academy of Sciences*, 107(41), 2010.

[20] K. Kursawe, G. Danezis, and M. Kohlweiss. Privacy-friendly Aggregation for the Smart-grid. In *PETS*, 2011.

[21] V. Lampos, A. C. Miller, S. Crossan, and C. Stefansen. Advances in Nowcasting Influenza-like Illness Rates using Search Query Logs. *Scientific Reports*, 5(12760), 2015.

[22] V. Lampos, E. Yom-Tov, R. Pebody, and I. J. Cox. Assessing the Impact of a Health Intervention via User-Generated Internet Content. *Data Mining and Knowledge Discovery*, 29(5), 2015.

[23] V. Lampos, B. Zou, and I. J. Cox. Enhancing Feature Selection Using Word Embeddings: The Case of Flu Surveillance. In *The World Wide Web Conference*, 2017.

[24] F. J. Massey Jr. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253), 1951.

[25] F. D. McSherry. Privacy Integrated Queries: An Extensible Platform for Privacy-preserving Data Analysis. In *SIGMOD*, 2009.

[26] L. Melis, G. Danezis, and E. De Cristofaro. Efficient Private Statistics with Succinct Sketches. In *NDSS*, 2016.

[27] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler. GUPT: Privacy Preserving Data Analysis Made Easy. In *SIGMOD '12*, 2012.

[28] Mozilla Labs. A Week in the Life of a Browser: Aggregated Data Sample. https://web.archive.org/web/20110711092459/https://testpilot.mozillalabs.com/testcases/a-week-life/aggregated-data.html, 2011.

[29] S. Palan and C. Schitter. Prolific.ac – A subject pool for online experiments. *Journal of Behavioral and Experimental Finance*, 17, 2018.

[30] J. Paparrizos, R. W. White, and E. Horvitz. Screening for Pancreatic Adenocarcinoma Using Signals From Web Search Logs: Feasibility Study and Results. *Journal of Oncology Practice*, 12(8), 2016.

[31] V. Pinchin. I'm Feeling Yucky :(Searching for symptoms on Google. Online at https://www.blog.google/products/search/im-feeling-yucky-searching-for-symptoms/, 2016.

[32] P. M. Polgreen, Y. Chen, D. M. Pennock, F. D. Nelson, and R. A. Weinstein. Using Internet Searches for Influenza Surveillance. *Clin. Infect. Dis.*, 47(11):1443–1448, 2008.

[33] A. Pyrgelis, E. De Cristofaro, and G. J. Ross. Privacy-friendly mobility analytics using aggregate location data. In *SIGSPATIAL*, 2016.

[34] A. Pyrgelis, C. Troncoso, and E. De Cristofaro. Knock Knock, Who's There? Membership Inference on Aggregate Location Data. In *NDSS*, 2018.

[35] Y. Research. L18 - Anonymized Yahoo! Search Logs with Relevance Judgments. Online at https://webscope.sandbox.yahoo.com/catalog.php?datatype=l.

[36] L. Soldaini and E. Yom-Tov. Inferring Individual Attributes from Search Engine Queries and Auxiliary Information. In *The World Wide Web Conference*, 2017.

[37] D. Sullivan. Google now handles at least 2 trillion searches per year. Online at https://searchengineland.com/google-now-handles-2-999-trillion-searches-per-year-250247, 2016.

[38] M. Wagner, V. Lampos, E. Yom-Tov, R. Pebody, and I. J. Cox. Estimating the Population Impact of a New Pediatric Influenza Vaccination Program in England Using Social Media Content. *Journal of Medical Internet Research*, 19(12), 2017.

[39] R. White and E. Horvitz. Evaluation of the feasibility of screening patients for early signs of lung carcinoma in web search logs. *JAMA Oncology*, 3(3), 2017.

[40] L. Wu and E. Brynjolfsson. *The Future of Prediction: How Google Searches Foreshadow Housing Prices and Sales.* University of Chicago Press, 2015.

[41] S. Yang, M. Santillana, and S. C. Kou. Accurate Estimation of Influenza Epidemics using Google Search Data via ARGO. *Proceedings of the National Academy of Sciences*, 112(47), 2015.

[42] E. Yom-Tov. *Crowdsourced Health – How What You Do on the Internet Will Improve Medicine.* MIT Press, 2016.

[43] S. T. Zargar, J. Joshi, and D. Tipper. A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. *IEEE Communications Surveys Tutorials*, 15(4), 2013.

[44] T. Zeller. AOL executive quits after posting of search data. https://web.archive.org/web/20061126162350/http://www.iht.com/articles/2006/08/22/business/aol.php, 2006.

[45] P. Zhao and B. Yu. On Model Selection Consistency of Lasso. *Journal of Machine Learning Research*, 7, 2006.

[46] B. Zou, V. Lampos, and I. Cox. Multi-Task Learning Improves Disease Models from Web Search. In *The World Wide Web Conference*, 2018.

[47] H. Zou and T. Hastie. Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2), 2005.

Community Detection through Likelihood Optimization: In Search of a Sound Model

Liudmila Prokhorenkova
Moscow Institute of Physics and Technology
Yandex
Moscow, Russia
ostroumova-la@yandex.ru

Alexey Tikhonov
Yandex
Berlin, Germany
altsoph@gmail.com

ABSTRACT

Community detection is one of the most important problems in network analysis. Among many algorithms proposed for this task, methods based on statistical inference are of particular interest: they are mathematically sound and were shown to provide partitions of good quality. Statistical inference methods are based on fitting some random graph model (a.k.a. *null model*) to the observed network by maximizing the likelihood. The choice of this model is extremely important and is the main focus of the current study. We provide an extensive theoretical and empirical analysis to compare several models: the widely used planted partition model, recently proposed degree-corrected modification of this model, and a new null model having some desirable statistical properties. We also develop and compare two likelihood optimization algorithms suitable for the models under consideration. An extensive empirical analysis on a variety of datasets shows, in particular, that the new model is the best one for describing most of the considered real-world complex networks according to the likelihood of observed graph structures.

CCS CONCEPTS

• **Mathematics of computing → Random graphs**; **Graph algorithms**; **Maximum likelihood estimation**; • **Theory of computation → Random network models**.

KEYWORDS

Community detection; likelihood optimization; statistical inference; planted partition model; LFR benchmark

ACM Reference Format:
Liudmila Prokhorenkova and Alexey Tikhonov. 2019. Community Detection through Likelihood Optimization: In Search of a Sound Model. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313429

1 INTRODUCTION

Among various properties shared by many real-world complex networks, community structure is extremely important. It is characterized by the presence of highly interconnected groups of vertices (*communities* or *clusters*) relatively well separated from the rest of the network. In social networks communities are formed by users with similar interests, in citation networks they represent papers on related topics, etc. The presence of communities highly affects, e.g., the promotion of products via viral marketing, the spreading of infectious diseases, computer viruses and information, and so on. Being able to identify communities in a network could help us to exploit this network more effectively: find similar scientific papers, discover users with similar interests for targeted advertisement, compress or visualize the network, etc.

Among other algorithms proposed for community detection, the notable ones are methods based on statistical inference. In such methods, some underlying random graph model is assumed, the evidence is represented by the observed graph, and hidden variables to be inferred are the parameters of the model together with community assignments. Such methods are appealing since they are theoretically sound and consistent: e.g., it has been proved that when the maximum-likelihood method is applied to networks generated from the same stochastic block model, it returns correct cluster assignments in the limit of large degrees [7]. Also, likelihood can be used to formalize the notion of a community [15].

The choice of the proper null model is essential for statistical inference algorithms as it highly affects their performance. There are two types of models: with fixed or non-fixed number of parameters. Direct likelihood optimization for models with non-fixed number of parameters leads to trivial partitions, as discussed in Section 4.1, and additional heuristics are needed to define the number of clusters. Hence, in this paper we focus on models with fixed number of parameters. We analyze strengths and weaknesses of two most popular models — planted partition model and its degree-corrected variant (they are discussed in Section 3). We also propose a new model, which has only one parameter, satisfies a desired statistical property of preserving the expected degree sequence, and shows the best fit for a variety of real-worlds networks according to the likelihood of observed structures. To the best of our knowledge, we are the first to perform a thorough theoretical and empirical evaluation of the null models used for community detection.

To sum up, the contributions of this paper are the following:

- We theoretically analyze two well-known null models used for community detection. Based on this analysis, we propose a new one-parameter model which has a desirable statistical property of preserving the expected degree sequence.
- We empirically demonstrate that the new model gives the best fit to a variety of real-world complex networks.
- We show that the likelihood based on the proposed model can be used as a new, more reliable, alternative to the well-known modularity measure.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313429

- We develop several likelihood optimization algorithms suitable for all discussed models. We perform an extensive empirical comparison of all the algorithms on various synthetic and real-world datasets.
- The implementation of the proposed algorithms is available as an open-source library.[1]

The rest of the paper is organized as follows. In the next section we overview related papers and introduce the required background. Then, in Section 3, we formally define and analyze all null models used in the current research. In Section 4, we discuss the method of statistical inference and present the likelihoods for all models. The proposed likelihood optimization algorithms are discussed in Section 5. Section 6 provides an extensive empirical analysis of all models and algorithms and Section 7 concludes the paper.

2 BACKGROUND AND RELATED WORK

2.1 Community Detection Methods

In this section, we briefly overview algorithms used for community detection. However, most of the related work will be discussed in the corresponding background sections: Section 2.2 discusses modularity measure; Section 2.3 defines the LFR bechmark; Sections 3.1 and 3.2 describe standard null models; Section 4.2 discusses papers on statistical inference; Section 5.1 deals with some optimization techniques. For a deeper overview of community detection area, we refer the reader to several survey papers, e.g., [10, 16, 19, 21, 33].

The problem of community detection has recently attracted a lot of attention and many algorithms were proposed to solve this problem. The proposed methods include: spectral algorithms [55]; methods based on statistical inference (see Section 4); methods based on optimization (see Section 5); methods based on dynamical processes on networks, like diffusion, spin dynamics, synchronization, and so on. Also, existing approaches differ by a particular task at hand: detecting overlapping communities [57], clustering in dynamic [51], directed [48], or weighted [3] networks, community detection in annotated [39] or layered [44] networks, and so on.

In this paper, we focus on the basic problem of detecting non-overlapping communities in undirected and unweighted graphs. However, ideas discussed in this paper can be applied to more general cases, since there exist the corresponding generalizations of the null models. We leave this analysis for the future research, since there is a lot to be understood even for the basic case.

2.2 Modularity

Throughout this paper we assume that we are given a simple undirected graph $G = (V(G), E(G))$ and the task is to find a partition $C = \{C_1, \dots, C_k\}$ of its vertex set $V(G)$ such that $\bigcup C_i = V(G)$ and $C_i \bigcap C_j = \emptyset$ for $i \neq j$. The notation used in the paper is summarized in Table 1.

Let us introduce the notion of *modularity*, which is the most well-known measure of a goodness of a partition C for a graph G. It was first introduced in [40] and is widely used in community detection literature: modularity is directly optimized by some algorithms, used as a stopping criteria in iterative methods, or used as a metric to compare different algorithms when no ground truth

[1] https://github.com/altsoph/community_loglike

Table 1: Notation

Variable	Description
G	observed graph, simple and undirected
$V(G)$	set of vertices
$E(G)$	set of edges
$n = \|V(G)\|$	number of vertices in G
$m = \|E(G)\|$	number of edges in G
$C = \{C_1, \dots, C_k\}$	partition of $V(G)$
$k = \|C\|$	number of communities
$Q_0(C, G, \gamma)$	simple modularity defined in (1)
$Q_1(C, G, \gamma)$	standard modularity defined in (2)
$c(i)$	cluster assignment of vertex i
$m_{in} = m_{in}(C, G)$	number of intra-community edges
$m_{out} = m_{out}(C, G)$	number of inter-community edges
$P = \binom{n}{2}$	number of pairs of vertices
$P_{in} = \sum_{C \in C} \binom{\|C\|}{2}$	number of intra-community pairs of vertices
$P_{out} = P - P_{in}$	number of inter-community pairs of vertices
$d(i)$	degree of vertex i in G
$d_{in}(i)$	number edges going from i to vertices in $C_{c(i)}$
$d_{out}(i) = d - d_{in}(i)$	number of edges going from i to outside $C_{c(i)}$
$D(C) = \sum_{i \in C} d(i)$	degree of cluster C
$D_{in}(C) = \sum_{i \in C} d_{in}(i)$	twice the number of edges induced by C
$m(C_q, C_r)$	number of edges between C_q and C_r or twice the number of intra-cluster edges if $q = r$

partition is available. The basic idea is to consider the fraction of intra-community edges among all edges of G and penalize it in order to avoid trivial partitions like $C = \{V(G)\}$. In its general form, modularity is

$$Q(C, G, \gamma) = \frac{1}{m} \left(m_{in} - \gamma \mathbb{E}(m'_{in}) \right),$$

where $m = m(G) = |E(G)|$ is the number of edges in G; $m_{in} = m_{in}(C, G)$ is the number of intra-community edges; $m'_{in} = m'_{in}(C)$ is a (random) number of intra-community edges in a graph constructed according to some underlying random graph model; γ is the *resolution parameter* [47], which is usually added to get the possibility of varying the number of communities obtained after maximizing $Q(C, G, \gamma)$.

The simplest underlying model is the Erdős–Rényi random graph, i.e., we assume that G is sampled uniformly at random from the set of all graphs with n vertices and m edges. For this model, $\mathbb{E}(m'_{in}) = P_{in} \cdot \frac{m}{P}$, where $P = \binom{n}{2}$ is the number of pairs of vertices and $P_{in} = \sum_{C \in C} \binom{|C|}{2}$ is the number of intra-community pairs. So, the formula for *simple modularity* is

$$Q_0(C, G, \gamma) = \frac{1}{m} \left(m_{in} - \gamma \frac{P_{in} m}{P} \right). \quad (1)$$

However, the Erdős–Rényi random graph model is known to be not a good descriptor of real-world networks since its Poisson degree distribution significantly differs from heavy-tailed degree distributions observed in real-world networks. A standard solution is to consider a random graph with a given expected degree sequence [12] as a null model, and to take the degree sequence

from the observed graph.[2] For this model, the probability that vertices i and j are connected equals $\frac{d(i)d(j)}{2m}$, where $d(i)$ is the degree of a vertex i.[3] In this case we have $\mathbb{E}(m'_{in}) = \frac{1}{4m}\sum_{C\in C} D(C)^2$, where $D(C) = \sum_{i\in C} d(i)$ is the degree of cluster C. So, the standard expression for *modularity* is

$$Q_1(C, G, \gamma) = \frac{m_{in}}{m} - \frac{\gamma}{4m^2}\sum_{C\in C} D(C)^2 . \tag{2}$$

Originally, modularity was introduced with $\gamma = 1$, but is was proven to have a resolution limit [20], i.e., it fails to detect communities smaller than a certain size; varying the resolution parameter helps to overcome this problem, but tuning γ is challenging.

Despite some known shortcomings [19], modularity remains the most popular way to measure the quality of a partition and many algorithms are based on direct modularity optimization (see Section 5). Also, under certain assumptions, modularity maximization is equivalent to likelihood optimization [36].

2.3 LFR Model

Let us now describe the LFR (Lancichinetti-Fortunato-Radicchi) model [29], which is the most widely used synthetic benchmark for comparison of community detection algorithms.

LFR generates a graph G on n vertices. The main parameters of the model are: the exponent of the power-law degree distribution γ, the exponent of the power-law community size distribution β, the average degree $\bar{d}$, and mixing parameter $\hat{\mu}$. First, we generate the degrees of vertices by sampling them independently from the power-law distribution with exponent γ and mean $\bar{d}$. Each vertex shares a fraction $1-\hat{\mu}$ of its edges with the vertices of its community and a fraction $\hat{\mu}$ with the other vertices of the network. The sizes of the communities are sampled from the power-law distribution with exponent β, such that the sum of all sizes equals n. Then, vertices are assigned to communities such that the internal degree of any vertex is less than the size of its community. Finally, the configuration model [35] with rewiring steps is used to construct a graph with the given degree sequence and with the required fraction of internal edges. See [29] for the detailed description of this procedure.

3 NULL MODELS

In this section, we analyze several random graph models having community structure and suitable for using as null models for likelihood optimization.

3.1 Stochastic Block Model

The most well-known random graph model with community structure is the *stochastic block model* (SBM) [26]. In this model, the vertices are divided in k clusters and for each pair of vertices i, j we draw an edge between them with probability $p_{c(i),c(j)}$ independently of all other edges. Here $c(i)$ is a community assignment for a vertex i. The values $p_{q,r}$, $1 \le q, r \le k$ are parameters of the model, $0 \le p_{q,r} \le 1$ and $p_{q,r} = p_{r,q}$. In many community detection algorithms a simplified version of SBM called *planted partition model* (PPM) is used [14]. PPM is a special case of SBM, where $p_{q,q} = p_{in}$ for $1 \le q \le k$ and $p_{q,r} = p_{out}$ for $1 \le q, r \le k, q \ne r$.

3.2 Degree-Corrected Stochastic Block Model

Similarly to the Erdős–Rényi random graph, SBM is unable to model heavy-tailed degree distributions, so community detection algorithms based on this model may have a poor quality. To overcome this issue, Karrer and Newman proposed the *degree-corrected stochastic block model* (DCSBM) [28]. In this model the vertices are again assigned to k clusters and edges are placed independently at random. The number of edges between vertices i and j follows a Poisson distribution with the mean $\frac{d(i)d(j)}{2m}p_{c(i),c(j)}$ or a half that number for self-loops. Let us show that DCSBM is able to generate graphs with desired expected degrees.

Proposition 1. *There exist such $p_{q,r}$ that in DCPPM we have $\mathbb{E}(d'(i)) = d(i)$ for all i.*

Proof. Let us set $p_{q,r} = \frac{2\,m\,m(C_q, C_r)}{D(C_q)\,D(C_r)}$, where $m(C_q, C_r)$ is the number of edges between clusters C_q and C_r or twice the number of intra-cluster edges if $q = r$. Note that these parameters maximize the likelihood for DCSBM [28]. For such parameters we get the desired condition:

$$\begin{aligned}\mathbb{E}(d'(i)) &= \sum_{j=1}^{n}\frac{d(i)d(j)}{2m}p_{c(i),c(j)} \\ &= \frac{d(i)}{D\big(C_{c(i)}\big)}\sum_{C\in C}\frac{m\big(C, C_{c(i)}\big)}{D(C)}\sum_{j\in C} d(j) = d(i) .\end{aligned}$$

So, we proved that in its general form DCSBM is able to preserve the desired degree sequence. □

The *degree-corrected planted partition model* (DCPPM) [36] is a simplified version of DCSBM, where $p_{q,q} = p_{in}$ for $1 \le q \le k$ and $p_{q,r} = p_{out}$ for $1 \le q, r \le k, q \ne r$. Let us prove the following "negative" proposition.

Proposition 2. *If $p_{in} \ne p_{out}$ and for some q, r we have $D(C_q) \ne D(C_r)$, then $\mathbb{E}(d'(i)) \ne d(i)$ for some i.*

Proof. Let us compute the expected degree of i:

$$\begin{aligned}\mathbb{E}(d'(i)) &= \frac{d(i)}{2m}p_{in}\sum_{j\in C_{c(i)}} d(j) + \frac{d(i)}{2m}p_{out}\sum_{C\ne C_{c(i)}}\sum_{j\in C} d(j) \\ &= \frac{d(i)}{2m}\left(D\big(C_{c(i)}\big)(p_{in} - p_{out}) + 2m\,p_{out}\right) .\end{aligned}$$

To get $\mathbb{E}(d'(i)) = d(i)$ for all i, we need to have, for *any* cluster C, $D(C)(p_{in} - p_{out}) + 2m\,p_{out} = 2m$. As we have at least two clusters with different values $D(C)$, we have to take $p_{in} = p_{out} = 1$, which leads to the standard configuration model without any community structure and contradicts the statement of the proposition. □

As a result, we obtain that the standard method of limiting the number of parameters in DCSBM leads to inability of the obtained model to preserve the expected degree sequence.

[2] At large scale, this model is essentially equivalent to the configuration model [35]: a random graph sampled uniformly from the set of graphs with a given degree sequence.

[3] For technical purposes, it is more convenient (and conventional) to allow the underlying model to have loops and multiple edges and to assume that the number of edges between i and j follows a Poisson distribution with the mean $\frac{d(i)d(j)}{2m}$ if $i \ne i$ and the expected number of loops for a vertex i is $\frac{d(i)^2}{4m}$. Note that usually $d(i)d(j) \ll 2m$, so multiple edges rarely appear.

3.3 Independent LFR Model

Motivated by Proposition 2, we developed a one-parameter model which preserves the expected degree sequence. It is a special case of DCSBM and we call it *independent LFR model* (ILFR) due to its analogy to the LFR benchmark.

The core problem of DCPPM is the assumption that the probability of an internal edge is independent of the size of its community, so the fraction of internal edges for a vertex *depends on the community size*. On the contrary, we propose using the mixing parameter μ to control this fraction. Namely, we consider DCSBM with the following expected number of edges between two vertices i and j: $\frac{\mu d(i)d(j)}{2m}$ if $c(i) \neq c(j)$ or $\frac{(1-\mu)d(i)d(j)}{D(C_{c(i)})} + \frac{\mu d(i)d(j)}{2m}$ if $c(i) = c(j)$ (or half this number for the self loops). Note that making the probability of an internal edge dependent on the size of the community is very natural: e.g., if a community of people is very small, one would expect that its members are much closer related to each other than members of large communities. The following proposition holds for the proposed model.

Proposition 3. *In ILFR we have* $\mathbb{E}(d'(i)) = d(i)$ *for all* i.

Proof. Indeed, let us compute the expected degree of i:

$$\mathbb{E}(d'(i)) = d(i)\left((1-\mu)\sum_{j\in C_{c(i)}}\frac{d(j)}{D(C_{c(i)})} + \mu\sum_{j\in V(G)}\frac{d(j)}{2m}\right) = d(i)\,.$$

□

So, this model solves the problem with expected degrees. Another advantage of ILFR is that it has only one parameter μ instead of p_{in} and p_{out} in planted partition models.

4 STATISTICAL INFERENCE

4.1 Method

Having discussed possible options for the null models, we are ready to describe the statistical inference method. For community detection problem, this method consists of two ingredients: the evidence, expressed by a graph G, and a random graph null model with parameters $\boldsymbol{\theta}$ ($\boldsymbol{\theta}$ include community assignments and μ or p_{in} and p_{out} for the models under consideration). The goal is to find such parameters $\boldsymbol{\theta}$ that maximize the posterior distribution

$$\mathrm{P}(\boldsymbol{\theta}|G) = \frac{\mathrm{P}(G|\boldsymbol{\theta})\mathrm{P}(\boldsymbol{\theta})}{\mathrm{P}(G)}\,,$$

where $\mathrm{P}(\boldsymbol{\theta})$ is the prior distribution of parameters and $\mathrm{P}(G) = \int \mathrm{P}(G|\bar{\boldsymbol{\theta}})\mathrm{P}(\bar{\boldsymbol{\theta}})\,d\bar{\boldsymbol{\theta}}$ is independent of $\boldsymbol{\theta}$, therefore $\arg\max_{\boldsymbol{\theta}} \mathrm{P}(\boldsymbol{\theta}|G) = \arg\max_{\boldsymbol{\theta}} \mathrm{P}(G|\boldsymbol{\theta})\mathrm{P}(\boldsymbol{\theta})$. The choice of the prior distribution $\mathrm{P}(\boldsymbol{\theta})$ is usually not obvious. Therefore, in the community detection literature, the likelihood $P(G|\boldsymbol{\theta})$ that the model is consistent with the observed graph structure is often maximized.

Let us also stress that in Section 3 we discussed two types of models: SBM/DCSBM having non-fixed number of parameters and PPM/DCPPM/ILFR with a fixed number of parameters. If the number of parameters is fixed, then the likelihood optimization can be applied directly. Otherwise, one needs another tool to find the optimal number of clusters, since direct likelihood maximization would lead to a trivial partition with all vertices forming their own clusters and the probability of an edge between two vertices equals 1 if they are connected and 0 otherwise. To avoid this problem, we further focus only on the models with fixed number of parameters.

4.2 Related Work on Statistical Inference

Let us briefly describe previous literature on statistical inference in community detection. Hastings [24] considers PPM, as defined in Section 3.1. He shows a connection between the obtained likelihood and the Hamiltonian of a Potts model with short- and long-range interactions. For $p_{in} > p_{out}$, the model is a spin glass with ferromagnetic nearest-neighbor interactions and antiferromagnetic long-range interactions. Belief propagation is used to find the ground state of the spin model. Another method based on the directed SBM is presented by Newman and Leicht [41], they use the expectation-maximization technique to optimize the parameters. These ideas are close to the a posteriori blockmodel proposed in [42]. A variational approach to parameter estimation is adopted in [17]. Some improvements and modifications of these approaches are introduced in, e.g., [11, 25, 45, 59]. A generalization to the overlapping SBM is proposed by McDaid and Hurley [34]. As we discussed, DCSBM is defined in [28] and communities are detected based on the likelihood in this model. Copic et al. [15] define an axiomatization for the problem of community detection based on PPM. Likelihood is used as a quality function to define rankings between graph partitions and it is shown that such rankings satisfy a number of desired properties. The authors also propose an algorithm to find (approximately) the maximum likelihood partition.

Finally, if the number of parameters in the model is not fixed, some additional tools are required to figure out the optimal number of clusters. Here the average entropy of the classification [45] or the Integrated Classification Likelihood [17, 59] can be used.

4.3 Likelihood for PPM and Simple Modularity

The log-likelihood for PPM can be easily written:

$$\log L'_{PPM}(C, G, p_{in}, p_{out}) = m_{in}\log p_{in} + m_{out}\log p_{out} + (P_{in} - m_{in})\log(1-p_{in}) + (P_{out} - m_{out})\log(1-p_{out})\,.$$

Although it can be optimized directly using the algorithms discussed further in the paper, in order to demonstrate a connection of the log-likelihood to simple modularity defined in (1) and to deal with all models in a unified way, we use the conventional trick and say that the number of edges between i and j follows a Poisson distribution with parameter p_{in} or p_{out}. Then we get:

$$\log L_{PPM}(C, G, p_{in}, p_{out}) = m_{in}\log p_{in} + m_{out}\log p_{out} - P_{in}p_{in} - P_{out}p_{out}\,. \quad (3)$$

Note that the values p_{in} and p_{out} maximizing (3) are

$$p_{in} = \frac{m_{in}}{P_{in}},\quad p_{out} = \frac{m_{out}}{P_{out}}\,. \quad (4)$$

In a resent paper by Newman [36] it was shown that if we assume p_{in} and p_{out} to be fixed, then maximizing (3) is equivalent to maximizing simple modularity (1) with

$$\gamma = \frac{P(p_{in} - p_{out})}{m(\log p_{in} - \log p_{out})}\,. \quad (5)$$

4.4 Likelihood for DCPPM and Modularity

The log-likelihood for DCPPM is [36]:

$$\log L_{DCPPM}(C,G,p_{in},p_{out}) = m_{in}(\log p_{in} - \log p_{out}) - \frac{p_{in}-p_{out}}{4m}\sum_{C\in C} D(C)^2 + \sum_i d(i)\log d(i) + m\log p_{out} - mp_{out} - m\log(2m)\,. \quad (6)$$

The values of parameters maximizing this likelihood are:

$$p_{in} = \frac{4\,m\,m_{in}}{\sum_C D(C)^2}\,,\quad p_{out} = \frac{4\,m\,m_{out}}{4m^2 - \sum_C D(C)^2}\,. \quad (7)$$

As shown in [36], if p_{in} and p_{out} are fixed, then maximizing $\log L_{DCPPM}(C,G,p_{in},p_{out})$ is exactly equivalent to maximizing modularity (2) with

$$\gamma = \frac{p_{in}-p_{out}}{\log p_{in} - \log p_{out}}\,. \quad (8)$$

4.5 Likelihood for ILFR

Let us compute the likelihood for the proposed ILFR model:

$$\log L_{ILFR}(C,G,\mu) = -\frac{1}{2}\sum_{i,j:c(i)=c(j)} d(i)d(j)\left(\frac{(1-\mu)}{D(C_{c(i)})} + \frac{\mu}{2m}\right) - \sum_{\substack{i,j:i<j,\\ c(i)\neq c(j)}} \frac{\mu d(i)d(j)}{2m} + \sum_{\substack{(i,j)\in E(G),\\ i<j,c(i)=c(j)}} \log\left(d(i)d(j)\left(\frac{(1-\mu)}{D(C_{c(i)})} + \frac{\mu}{2m}\right)\right) + \sum_{\substack{(i,j)\in E(G),\\ i<j,c(i)\neq c(j)}} \log\frac{\mu d(i)d(j)}{2m} = \sum_{C\in C}\frac{D_{in}(C)}{2}\log\left(\frac{(1-\mu)}{D(C)} + \frac{\mu}{2m}\right) + m_{out}\log\mu + \sum_i d(i)\log d(i) - m_{out}\log 2m - m\,, \quad (9)$$

where $D_{in}(C) = \sum_{i\in C} d_{in}(i)$ is twice the number of edges induced by C. Note that the optimal value of μ is hard to find analytically, but it can be approximated numerically by optimizing (9).

The following approximation helps to speed up the optimization algorithms and to make formulas more concise:

$$\log\left(\frac{(1-\mu)}{D(C)} + \frac{\mu}{2m}\right) = \log\frac{(1-\mu)}{D(C)} + \log\left(1 + \frac{\mu\,D(C)}{(1-\mu)\,2\,m}\right) \approx \log\frac{(1-\mu)}{D(C)} + \frac{\mu D(C)}{(1-\mu)\,2\,m} \approx \log\frac{(1-\mu)}{D(C)}\,,$$

since $\left|\log\frac{1-\mu}{D(C)}\right| \gg \frac{\mu D(C)}{(1-\mu)\,2\,m}$. This leads to another quality function, which we further refer to as ILFRS (S stands for "simplified"):

$$\log L_{ILFRS}(C,G,\mu) = m_{in}\log(1-\mu) + m_{out}\log\mu - m_{out}\log 2m - \sum_C \frac{D_{in}(C)}{2}\log D(C) + \sum_i d(i)\log d(i) - m\,. \quad (10)$$

The optimal μ according to (10) can now be computed analytically:

$$\mu = \frac{m_{out}}{m}\,. \quad (11)$$

Note that now we can substitute μ in (10) by the optimal value (11) and obtain a non-parametric quality function:

$$\log L_{ILFRS}(C,G) = m_{in}\log\frac{m_{in}}{m} + m_{out}\log\frac{m_{out}}{m} - m - m_{out}\log 2m - \sum_C \frac{D_{in}(C)}{2}\log D(C) + \sum_i d(i)\log d(i)\,. \quad (12)$$

The obtained function is fairly simple and, as we show by further experiments, it can successfully replace the standard modularity function in many applications.

Let us also note that in contrast to ILFR, the likelihood for the standard LFR model cannot be computed. First, LFR is based on the configuration model, which introduces complex dependences between all edges, making the likelihood intractable. Second, and most importantly, the number of iter-community edges for each vertex is deterministic (the fraction of such edges is μ); as a result, for most of graphs and partitions the likelihood is equal to zero.

5 OPTIMIZATION

5.1 Related Work on Modularity Optimization

First, let us discuss some optimization approaches used in community detection. The most widely used measure to be optimized is modularity (2). The following types of approaches are known.

Greedy optimization. Newman and Girvan [40] proposed a greedy algorithm for modularity optimization, where the clusters are iteratively merged by choosing the pair which leads to a greater modularity increase (while it is positive). A speedup of the greedy algorithm was proposed by Clauset et al. [13]. Some other modifications were also suggested in, e.g., [49, 56]. Probably the most well-known and widely used greedy algorithm is called Louvain [8]. At the beginning each vertex forms its own cluster. Then we create the first level of the partition by iterating through all vertices: for each vertex i we compute the gain in modularity coming from putting i to the community of its neighbor and pick the community with the largest gain, as long as it is positive. After that the first level is formed and we replace the obtained communities by supervertices, and two supervertices are connected by a (weighted) edge if there is at least one edge between vertices of the corresponding communities. Then the process is repeated with the supervertices, and so on, as long as modularity increases. This algorithm was shown to be fast and provide partitions of good quality. We choose Louvain algorithm as the basis for our experiments, the detailed description of its application to our problem is given in Section 5.2.

Simulated annealing. Simulated annealing was first applied to modularity optimization in [23]. This method combines two types of moves: local moves, where a single vertex is shifted from one cluster to another; global moves, consisting of merges and splits of communities. Methods based on simulated annealing are relatively slow and cannot be applied to large datasets.

Spectral optimization. Spectral methods are quite popular in modularity optimization. Such methods use the eigenvalues and eigenvectors of a so-called modularity matrix. Some examples can be found in [37, 53].

Many other algorithms exist, among them are methods based on extremal optimization [18], mathematical programming [2], mean field annealing [30], genetic algorithms [54], etc.

5.2 Proposed Likelihood Optimization Methods

In Section 4, we presented four quality functions based on the likelihood: PPM (3), DCPPM (6), ILFR (9), and ILFRS (10). All functions are parametric: PPM and DCPPM have parameters p_{in} and p_{out}, ILFR and ILFRS have one parameter μ. In this section, we discuss the possible ways to maximize these likelihoods. Below we assume that we are given an arbitrary algorithm $FindPartition(\bar{Q}, G)$ which is able to find a partition by maximizing some quality function $\bar{Q}(C, G)$.

Although the optimization strategies proposed below are able to work with any algorithm *FindPartition*, let us first discuss our particular choice. As we already mentioned, we choose Louvain algorithm [8], which is arguably the most widely used method for community detection. Louvain is fast and allows to process large datasets. Importantly, it can be adapted to all quality functions discussed in this paper. Initially the method is designed to optimize modularity, so it can be directly applied to the likelihood of DCPPM due to their equivalence. To adapt Louvain to other likelihoods, we have to change one part of the original algorithm: when we try to remove a vertex (or supervertex) from one community an add to another, we need to efficiently compute the difference in likelihood. This can be done for any of the considered quality functions: for ILFR and ILFRS the differences can be computed using (9) and (10); for PPM we have to additionally store the sizes of supervertices in order to compute the difference for P_{in} in (1). The rest of the Louvain algorithm remains the same.[4] Therefore, the asymptotic time complexity of our Louvain-based *FindPartition* and of the original Louvain are the same, up to a constant multiplier (the complexity of Louvain is empirically evaluated to be $O(n \log n)$).

Note that both the original Louvain and our modifications can be applied to the likelihoods with fixed values of parameters p_{in}, p_{out} or μ. Therefore, the key question is how to find the optimal values of these parameters.

Iterative strategy. The simplified version of this strategy was initially proposed in [36] for PPM and DCPPM models. Detailed description of our method is presented in Algorithm 1. Initially, we fix $\gamma = 1$ or $\mu = 0.5$ depending on the null model. Then, we apply *FindPartition* to the corresponding log-likelihood function. Using the obtained partition we can re-estimate the parameters and continue this procedure until convergence or until some maximum allowed number of steps is reached. In our experiments we noticed that for any null model the parameters may end up cyclically varying near some value. Therefore, at each iteration we additionally check if we obtain an already seen parameter and stop in this case. This additional trick allows to significantly reduce the overall number of iterations. In our experiments we observed that we never need more than 50 iterations and the mean value is much smaller, see Section 6 for more details.

Maximization strategy. An alternative strategy is to directly search for parameters which maximize likelihood of the obtained partition. Here we can use any black-box optimization method. At each iteration of optimization, for some value of γ or μ, we run *FindPartition* to obtain a partition C, then, using C, we find the optimal parameters according to (4), (7), (11) or by optimizing (9). Finally, we compute the likelihood to be maximized using (3), (6), (9), or (10). In our experiments, for simplicity and reproducibility, we use grid search in this strategy. However, any other method of black-box parameter optimization can be used instead, e.g., random search [5], Bayesian optimization [50], Gaussian processes [22], sequential model-based optimization [6, 27], and so on.

Algorithm 1: Iterative strategy

input : graph G, *Model*, algorithm *FindPartition*, N, ε

Initialize $param = 1$ (γ for PPM, DCPPM) or $param = 0.5$ (μ for ILFR, ILFRS), $Params = \emptyset$;
for $i \leftarrow 1$ **to** N **do**
  Define $\bar{Q}$ according to (1) for PPM, (2) for DCPPM, (9) for ILFR, (10) for ILFRS with parameter $param$;
  $C = FindPartition(\bar{Q}, G)$;
  if *Model is PPM or DCPPM* **then**
    Compute p_{in} and p_{out} according to (4) or (7);
    Compute $param_{new}$ according to (5) or (8);
  if *Model is ILFRS* **then**
    Compute $param_{new}$ according to (11);
  if *Model is ILFR* **then**
    Compute $param_{new}$ by optimizing (9);
  if $|param - param_{new}| < \varepsilon$ *or* $param_{new} \in Params$ **then**
    break;
  Add $param_{new}$ to $Params$;
  $param \leftarrow param_{new}$;
return C

To summarize, the maximization strategy directly maximizes the obtained likelihood, while the iterative one searches for a "stable" parameter, i.e., the one which does not change after applying *FindPartition*. Note that the convergence of the iterative strategy is not guaranteed for real-world networks. Therefore, initially we expected the maximization strategy to be more stable. However, our experiments show that these two strategies demonstrate similar performance, but the iterative one is faster.

Let us mention another possible optimization strategy, which we do not use in our experiments. Given a partition C, for each likelihood function except ILFR, we can compute the optimal values of parameters according to (4), (7), and (11). Therefore, for the corresponding models we can substitute these parameters and obtain non-parametric likelihoods, as we did in (12) for ILFRS. All obtained non-parametric quality functions can potentially be optimized directly since they do not have free parameters. However, we do not consider such strategy since: 1) it cannot be applied to ILFR as there is no analytical formula for optimal μ, 2) this strategy cannot be easily combined with the Louvain algorithm. The reason is that Louvain has several partition levels and partitions obtained on earlier levels cannot be changed later. At the beginning of the algorithm our estimates of μ, p_{in}, and p_{out} are far from optimal:

[4] To make our results reproducible, we made the source code publicly available at https://github.com/altsoph/community_loglike.

all vertices form their own communities, so $\mu = 1$, $p_{out} = 1$, and $p_{in} = 0$. As a result, the first level of the partition, which has a big impact on the final quality, is constructed based on non-optimal parameters. In fact, in most cases the algorithm does not even start optimization due to the huge overestimate of μ or p_{out}.

6 EXPERIMENTS

In this section, we conduct an extensive experimental study to compare the discussed models. We start with some preliminaries and discuss evaluation metrics and datasets. In Section 6.3, we measure how well the models describe various real-world networks. Next, we compare all proposed community detection algorithms. Finally, Section 6.5 presents some negative results on the limits of statistical inference algorithms when applied to real-world networks.

6.1 Evaluation Metrics

In order to evaluate the performance of any algorithm, we have to compare the partition C obtained by this algorithm with the ground truth partition C_{GT}. The problem of choosing a good similarity measure for this comparison does not have any standard solution in community detection literature. Different similarity measures can possibly give preferences to different algorithms. Usually, a suitable measure is chosen according to a practical problem at hand. That is why in this work we compute and compare several standard similarity measures. In particular, we consider Normalized Mutual Information (NMI) of two partitions C and C_{GT}, which is often used for the comparison of community detection algorithms [4, 19]. The idea behind NMI is that if two partitions are similar, one needs very little information to infer C_{GT} given C. We also use two well-known similarity measures based on counting correctly and incorrectly classified pairs of vertices: Rand and Jaccard indices [19, 46].

6.2 Datasets

6.2.1 Synthetic networks. We use the LFR model described in Section 2.3 to generate synthetic networks. The parameters of the generated graphs are the following: the number of vertices $n = 10^4$; the parameter of the power-law degree distribution $\gamma = 2.5$; the average degree $\bar{d} = 30$; the parameter of the community size distribution $\beta = 1.5$, with the minimum cluster size 50 and the maximum 600; the mixing parameter $\hat{\mu}$ is varied in range $[0, 1]$.[5] Note that there are other benchmark models proposed in the literature. However, community detection in synthetic networks is not the main focus in this paper, therefore we do not consider such models.

6.2.2 Real-world networks. We collected several networks with different structural properties (see Table 2). In addition to widely used networks, such as Zachary's karate club, dolphin social network, and American college football, we also used annotated books about politics[6] and political blogs [1]. The dataset email-Eu-core is obtained from SNAP,[7] here the ground truth communities correspond to the departments of a research institute. In Cora citation dataset communities correspond to subjects of research papers, while in AS [9] the vertices are annotated with their countries. These are all publicly available datasets we found for non-overlapping community detection.[8]

[5]We additionally experimented with synthetic graphs of different edge densities obtained by varying the average degree. The results are omitted since the only difference observed is that the qualities of all algorithms are usually larger for denser graphs.
[6]V. Krebs, unpublished, http://www.orgnet.com/
[7]http://snap.stanford.edu/data/email-Eu-core.html

Table 2: Real-world datasets; μ, γ_0, and γ_1 are computed for ground truth partitions according to (11), (4, 5) and (7, 8)

Dataset	n	m	k	μ	γ_1	γ_0
Karate club [58]	34	78	2	0.128	0.78	0.78
Dolphins [32]	62	159	2	0.038	0.54	0.55
Football [40]	115	613	11	0.325	2.39	2.57
Political books [38]	105	441	3	0.159	0.86	0.89
Political blogs [1]	1224	16715	2	0.094	0.72	0.72
email-Eu-core [31]	986	16064	42	0.664	2.74	2.80
Cora citation [52]	24166	89157	70	0.458	5.46	6.21
AS [9]	23752	58416	176	0.561	1.15	1.40

Table 3: Log-likelihoods: datasets with ground truth

Dataset	$\log L_{PPM}$	$\log L_{DCPPM}$	$\log L_{ILFR}$
Karate	-206.12	-168.65	**-168.63**
Dolphins	-483.50	-439.52	**-428.64**
Football	**-1384.1**	-1426.7	-1428.4
Political books	-1363.8	**-1235.0**	-1243.3
Political blogs	-73912	-50756	**-50750**
Eu-core	-65559	-48783	**-48483**
Cora	-678306	-593358	**-584730**
AS	-542952	-351537	**-329784**

6.3 Comparison of Likelihoods

Since the main aim of this paper is to analyze and compare several null models having community structure, we start with the following research question: *Which model gives the best fit for real-world complex networks*? To answer this question, we compare the probabilities that real-world networks were generated by each of the null models. Namely, we compared the log-likelihoods (3) for PPM, (6) for DCPPM and (9) for ILFR.[9] Note that (10) is a simplified expression for (9), so we consider it only as a quality function but not as log-likelihood.

In the first experiment, we took the datasets described in Section 6.2.2 and assumed that the partitions are defined by the ground truth cluster assignments provided for these datasets. Then, we computed the optimal parameters p_{in}, p_{out} or μ and used them to compute the corresponding log-likelihood (see Table 3). One can see that PPM is the best model describing the Football dataset, DCPPM is the best for Pol-books, while for all other datasets ILFR has the largest likelihood. Note that ILFR has only one parameter to be tuned while DCPPM has two. Therefore, we initially expected that for many datasets DCPPM may have a larger likelihood. In this case, to decide which model better describes the data, we would have to adapt some information criterion (e.g., the Bayesian information

[8]It is easier to find datasets with ground truth overlapping communities, see, e.g., http://snap.stanford.edu/data/index.html#communities
[9]We intentionally do not consider synthetic networks in this experiment: likelihoods are expected to be heavily affected by the particular synthetic model used. For example, LFR benchmark could give a preference to ILFR null model.

Table 4: Log-likelihoods: datasets without ground truth

Dataset	n	$\log L_{PPM}$	$\log L_{DCPPM}$	$\log L_{ILFR}$
Karate	34	-191.182	-163.990	**-160.154**
Dolphins	62	-417.240	-398.718	**-394.538**
Football	115	**-1364.38**	-1407.43	-1406.79
Political books	105	-1182.90	-1135.57	**-1088.77**
Political blogs	1224	-60653.8	-49912.3	**-49702.3**
Eu-core	986	-57421.1	-46020.9	**-45469.2**
Cora	24K	-512556	-450154	**-425463**
AS	24K	-449088	-244745	**-227917**
Ca-GrQc	5242	-50258.2	-53393.9	**-42259.3**
Ego-Facebook	4039	-241708	-234311	**-207910**
p2p-Gnutella09	8114	-167995	-148594	**-144711**
Wiki-vote	7115	-504395	**-388044**	-388380
Email-Enron	37K	-1192893	-862315	**-803161**
Soc-Epinions1	76K	-2825138	-2087117	**-2000906**
Soc-Slashdot0811	77K	-4011399	-3120963	**-2949833**
ego-Twitter	81K	-6979664	-5781209	**-5314782**

criterion) to our problem, which is a nontrivial task for the models under consideration. Surprisingly, for most of the datasets ILFR has a larger likelihood, which clearly indicates that this model is more suitable for describing real-world networks.[10]

However, as we further discuss in Section 6.4.2, even though for each dataset under consideration we have a ground truth partition (a.k.a. attributes or metadata), this partition can be not an ideal division of vertices into communities and also can be not the only ground truth partition possible. Fortunately, likelihoods can be compared even for datasets without any knowledge about community structure, which allows to compare null models on a much larger variety of datasets. In order to measure the likelihood for any graph, we first have to find a partition that maximizes this value (among all possible partitions), which is feasible only for very small datasets. However, the optimal partition (and the corresponding likelihood) can be approximately found by applying the corresponding maximization algorithm: PPM-max, DCPPM-max or ILFR-max. We performed such comparison on the datasets introduced in Section 6.2.2 as well as on 8 new datasets of various nature downloaded from http://snap.stanford.edu/data/index.html (see Table 4). According to this experiment, for almost all datasets the largest likelihood is again obtained for ILFR model, despite it has fewer free parameters, which means that this model is the best one for describing real-world datasets, which is the main empirical result supporting the introduction of ILFR and the corresponding log-likelihood quality function.

Let us note that the log-likelihoods presented in Table 4 can be underestimated, since we cannot guarantee that our optimization algorithms find exactly the maximum likelihood partition. However, we believe that this does not introduce any bias into this experiment (i.e., does not change the conclusions) since the same optimization procedure is applied for all models. Also, note that in all cases the log-likelihoods in Table 3 are smaller than the corresponding ones in Table 4, which means that our optimization algorithms found

[10]Note that using these results we cannot compare ILFR and DCPPM with PPM since ILFR and DCPPM are based on the observed degree sequences, while PPM only uses the cluster assignments.

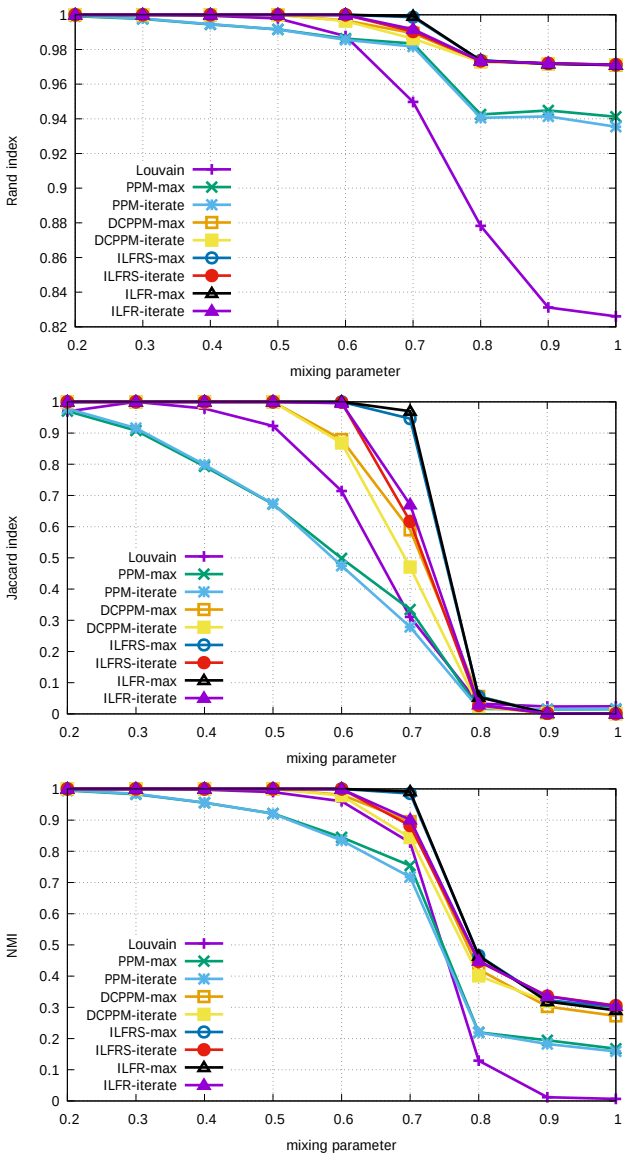


Figure 1: Comparison on synthetic networks

partitions providing better likelihoods for the corresponding models than the ground truth ones, which is expected (since partitions were tuned).

6.4 Community Detection Algorithms

In this section, we compare all algorithms proposed in Section 5.2. Note that we intentionally use only methods based on the likelihood optimization since this paper focuses on the analysis and comparison of null models.

6.4.1 Synthetic networks. First, we compared all algorithms on synthetic networks described in Section 6.2.1. For each $\hat{\mu}$ we generated 5 random samples of LFR and averaged the results. Figure 1 shows the results obtained for all similarity measures under consideration

Table 5: Maximization strategies (Rand index / Jaccard index / NMI)

Dataset	Louvain	PPM	DCPPM	ILFR	ILFRS
Karate	0.732 / 0.470 / 0.586	0.707 / 0.397 / 0.585	**0.766** / **0.523** / **0.667**	**0.774** / **0.535** / **0.687**	**0.774** / **0.535** / **0.687**
Dolphins	**0.634** / **0.351** / **0.499**	0.529 / 0.153 / 0.386	0.600 / 0.286 / 0.483	0.594 / 0.274 / 0.472	0.584 / 0.256 / 0.454
Football	0.970 / 0.720 / 0.922	**0.993** / **0.916** / **0.972**	**0.993** / **0.916** / **0.972**	**0.993** / **0.916** / **0.972**	**0.993** / **0.916** / **0.972**
Pol-books	0.831 / 0.616 / 0.545	0.722 / 0.331 / 0.449	0.817 / 0.583 / 0.536	0.719 / 0.347 / 0.437	0.772 / 0.477 / 0.490
Pol-blogs	0.886 / 0.788 / 0.641	0.574 / 0.153 / 0.234	**0.904** / **0.823** / **0.705**	0.860 / 0.738 / 0.570	0.879 / 0.778 / 0.625
Eu-core	0.869 / 0.218 / 0.584	0.942 / 0.207 / 0.661	**0.964** / **0.434** / 0.740	0.953 / 0.371 / 0.727	0.960 / 0.410 / 0.739
Cora	0.943 / **0.127** / 0.460	**0.978** / 0.044 / 0.547	0.978 / 0.050 / 0.533	0.978 / 0.063 / **0.551**	0.978 / 0.060 / **0.551**
AS	0.821 / **0.198** / **0.489**	0.826 / 0.009 / 0.422	**0.826** / 0.026 / 0.461	0.826 / 0.018 / 0.438	0.826 / 0.018 / 0.438

Table 6: Iterative strategies (Rand index / Jaccard index / NMI)

Dataset	Louvain	PPM	DCPPM	ILFR	ILFRS
Karate	0.732 / 0.470 / 0.586	0.743 / 0.480 / 0.612	0.750 / 0.502 / 0.612	0.766 / 0.523 / 0.667	0.760 / 0.519 / 0.633
Dolphins	**0.634** / **0.351** / 0.499	0.548 / 0.190 / 0.403	0.600 / 0.286 / 0.481	0.576 / 0.238 / 0.449	0.575 / 0.238 / 0.448
Football	0.970 / 0.720 / 0.922	**0.993** / **0.916** / **0.972**	**0.993** / **0.916** / **0.972**	**0.993** / **0.916** / **0.972**	**0.993** / **0.916** / **0.972**
Pol-books	**0.831** / **0.616** / **0.545**	0.740 / 0.377 / 0.464	**0.843** / **0.651** / **0.554**	0.760 / 0.445 / 0.481	0.770 / 0.475 / 0.489
Pol-blogs	0.886 / 0.788 / 0.641	0.582 / 0.171 / 0.240	**0.902** / **0.820** / **0.694**	0.862 / 0.743 / 0.576	0.877 / 0.773 / 0.618
Eu-core	0.869 / 0.218 / 0.584	0.939 / 0.205 / 0.656	**0.965** / **0.439** / **0.742**	0.948 / 0.351 / 0.713	0.955 / 0.387 / 0.725
Cora	0.943 / **0.127** / 0.460	**0.978** / 0.049 / 0.547	0.978 / 0.052 / 0.534	0.978 / 0.058 / **0.549**	0.978 / 0.061 / **0.550**
AS	0.821 / **0.198** / **0.489**	0.826 / 0.011 / 0.426	**0.826** / 0.025 / 0.460	0.826 / 0.017 / 0.436	0.826 / 0.017 / 0.435

(Rand, Jaccard and NMI). Note that in most cases these similarity measures are consistent, i.e., for a given mixing parameter they rank the algorithms similarly. However, Jaccard index is less descriptive for large mixing parameters.

Based on the obtained results, several important observations can be made. For all values of $\hat{\mu}$, the best (or very close to the best) quality is obtained by ILFRS-max and ILFR-max strategies. For both ILFR and ILFRS it turns out that maximizing the likelihood is better than iteratively estimating the parameters, which supports our intuition that better partitions are expected to have higher likelihoods. For small values of $\hat{\mu}$ the worst results are obtained by PPM (both strategies), while for large values Louvain algorithm is the worst.

We also measured statistical significance of the obtained improvements: for each $\hat{\mu}$, we applied the algorithms to 5 random samples of LFR and used the paired t-test. For instance, it turns out that according to NMI ILFRS-max never looses significantly and it is significantly better (p-value < 0.05) than: DCPPM-max for $\hat{\mu} \in \{0.7, 0.9, 1\}$; DCPPM-iterate for $\hat{\mu} \in \{0.7, 0.8\}$; PPM-max and PPM-iterate for all $\hat{\mu} \in [0.2, 1]$; ILFR-iterate and ILFRS-iterate for $\hat{\mu} = 0.7$; and Louvain for $\hat{\mu} \in [0.4, 1]$. These results additionally support the choice of the quality function L_{ILFRS} (12) for community detection instead of the standard modularity function.

6.4.2 Real-world complex networks. It is argued in several papers that ground truth community labels (a.k.a. metadata) available for some real-world networks should not be used to analyze and compare community detection algorithms [43]. Indeed, such labels are usually obtained using some discrete-valued vertex attributes which are not guaranteed to be reasonable community assignments. For example, users in a social network can be split by gender, city, or interests, and each of these partitions could be treated as ground truth cluster assignments. An algorithm performing better for "city" labels can be worse for "interests" ones. As a result, no meaningful conclusion can be made based on such comparison. However, for the sake of completeness, we compare all algorithms on real-world networks, since it is still a standard practice in community detection literature.

Tables 5 and 6 present the results for the iterative and maximization strategies, respectively (in both cases we add Louvain for comparison).[11] Note that we cannot properly measure statistical significance since we cannot sample several copies for each dataset. However, we can account for the randomness included in the algorithms: they all order vertices at each iteration randomly. To do this, for each dataset we run all algorithms 5 times and then apply the unpaired t-test to compare them. We put numbers in bold in Tables 5 or 6 if there is a group of algorithms without significant differences (p-value > 0.05) inside the group and with significant differences with the rest of the algorithms.

The results in general are not consistent: although DCPPM is often the best, the winning model may depend on a graph under consideration and on a target metric. For example, Louvain wins on Cora and AS according to Jaccard index, but it is the worst on the same datasets according to Rand index. This supports the claim that ground truth community assignments can be noisy or irrelevant.

We also noticed that iterative and maximization strategies usually have similar performance and the choice between them is not straightforward. For example, for ILFRS it is often better to apply maximization strategy, while for PPM iterative one should be

[11] Recall that Louvain optimizes modularity, so it is similar to DCPPM but with fixed values of p_{in} and p_{out}.

Table 7: Quality functions: ground truth vs optimization

	Modularity		$\log L_{PPM}$		$\log L_{DCPPM}$		$\log L_{ILFRS}$		$\log L_{ILFR}$	
Dataset	GT	Louvain	GT	PPM-max	GT	DCPPM-max	GT	ILFRS-max	GT	ILFR-max
Karate	0.3715	0.4188	-206	-191	-169	-164	-176	-168	-169	-160
Dolphins	0.3735	0.5233	-483	-417	-439	-399	-434	-405	-429	-395
Football	0.5877	0.6043	-1384	-1364	-1427	-1407	-1447	-1425	-1428	-1407
Pol-books	0.4149	0.5205	-1364	-1183	-1235	-1136	-1285	-1125	-1243	-1089
Pol-blogs	0.4053	0.4267	-73913	-60654	-50756	-49912	-51843	-50614	-50750	-49702
Eu-core	0.3138	0.4211	-65559	-57421	-48783	-46021	-49073	-46202	-48483	-45469
Cora	0.5165	0.7875	-678306	-512556	-593358	-450154	-585837	-425657	-584730	-425463
AS	0.1708	0.6307	-542952	-449088	-351537	-244745	-338397	-227192	-329784	-227917

preferred. Based on this observation, we propose using the iterative strategy, which is faster (it converges after a small number of iteration, as discussed below).

The performance of ILFRS is in general slightly better than of ILFR. Taking into account our results on synthetic networks, we propose using the faster ILFRS instead of ILFR in all practical applications.

Finally, we analyzed the speed of convergence for iterative strategies on real-world datasets. Namely, we measured the number of iterations made by Algorithm 1 before some stopping criteria is reached. Note that the algorithm never stopped because of reaching the maximum allowed number of iterations $N = 50$, so we either observed a convergence or applied the cycling criteria (i.e., encountered an already seed parameter). The average number of iterations obtained for PPM is 12.6 (in all cases the algorithm converged). For DCPPM we got 6.2 (converged in all cases except AS, where we applied cycling stopping criteria after 18 iterations). For ILFRS we have 5.4 iterations (cycling criteria is applied to 3 datasets). For ILFR we got 4.4 (again, with 3 applications of cycling criteria).

6.5 Statistical Inference, Negative Result

Having noticed unstable results for real-world datasets, we tried to answer the following research question: *Is any of the null models suitable for community detection in real-world graphs?*

In order to answer this question, for each quality function we compare its value for the ground truth partition with its value for the partition obtained by the corresponding maximization algorithm (see Table 7). For all quality functions, including the widely used modularity, and for all datasets the ground truth partition has a lower value of the quality function. This means that further optimization of *any* quality function would not lead us towards the ground truth partition, which is a negative observation. In particular, there is no hope in improving the results obtained by our algorithms by replacing Louvain-based $FindPartition$ with some more effective maximization algorithm.

Note that in the literature it is often assumed that a partition with larger modularity is better and, as a result, an algorithm which leads to a partition with larger modularity is better. However, our observation above demonstrates that on the considered real-world datasets it is not the case. We also performed additional experiments and noticed that in almost all cases the value of a quality function for the ground truth partition is lower than for the partition obtained by *any* discussed optimization algorithm (not necessary optimizing the same quality function), which is an even stronger negative observation.

The following conclusion can be made: either the ground truth metadata contained in the considered real-world networks is not a good descriptor for a community structure or statistical inference algorithms based on all null models discussed in this paper are unable to detect real-world communities.

7 CONCLUSION

In this paper, we focused on the comparison of null models which can be used by likelihood optimization algorithms for community detection. We compared two well-known models, PPM and DCPPM, and a new model, ILFR, which has only one parameter and is proved to preserve the desired expected degree sequence. For the new model we have written the log-likelihood, both in parametric and self-contained forms. To maximize the parametric log-likelihood functions, we proposed and compared two optimization strategies: maximization and iterative.

The most important conclusion is that the proposed model, ILFR, is the best one for describing most of the considered real-world complex networks according to the likelihood of the observed graph structures, despite the fact that it has only one free parameter. We argue that the likelihood can be considered as the main argument in evaluating the null models instead of the direct comparison of community detection algorithms. The reason is that one cannot fully rely on ground truth cluster labels available for real-world networks. Also, we demonstrated that ILFR-based algorithms have the best performance on synthetic networks. Based on the obtained results, we believe that the proposed ILFR-based quality function (12) can be successfully used as a target for optimization algorithms, instead of the widely adopted modularity.

A natural direction for the future research is to analyze null models for overlapping community detection. This would be useful since many observed networks have overlapping communities. However, fundamental analysis of this problem is complicated by the fact that null models with overlapping communities are less developed and more complex for mathematical analysis.

ACKNOWLEDGMENTS

This study was funded by RFBR according to the research project 18-31-00207.

REFERENCES

[1] Lada A Adamic and Natalie Glance. 2005. The political blogosphere and the 2004 US election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*. ACM, 36–43.

[2] Gaurav Agarwal and David Kempe. 2008. Modularity-maximizing graph communities via mathematical programming. *The European Physical Journal B* 66, 3 (2008), 409–418.

[3] Christopher Aicher, Abigail Z Jacobs, and Aaron Clauset. 2014. Learning latent block structure in weighted networks. *Journal of Complex Networks* 3, 2 (2014), 221–248.

[4] James P Bagrow. 2008. Evaluating local community methods in networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008, 05 (2008), P05001.

[5] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* 13, Feb (2012), 281–305.

[6] James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. In *Advances in neural information processing systems*. 2546–2554.

[7] Peter J Bickel and Aiyou Chen. 2009. A nonparametric view of network models and Newman–Girvan and other modularities. *Proceedings of the National Academy of Sciences* 106, 50 (2009), 21068–21073.

[8] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[9] Marián Boguná, Fragkiskos Papadopoulos, and Dmitri Krioukov. 2010. Sustaining the internet with hyperbolic mapping. *Nature communications* 1 (2010), 62.

[10] Tanmoy Chakraborty, Ayushi Dalmia, Animesh Mukherjee, and Niloy Ganguly. 2017. Metrics for community analysis: A survey. *ACM Computing Surveys (CSUR)* 50, 4 (2017), 54.

[11] Yi Chen, XL Wang, Xin Xiang, BZ Tang, and JZ Bu. 2015. Network structure exploration via Bayesian nonparametric models. *Journal of Statistical Mechanics: Theory and Experiment* 2015, 10 (2015), P10004.

[12] Fan Chung and Linyuan Lu. 2002. Connected components in random graphs with given expected degree sequences. *Annals of combinatorics* 6, 2 (2002), 125–145.

[13] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. 2004. Finding community structure in very large networks. *Physical review E* 70, 6 (2004), 066111.

[14] Anne Condon and Richard M Karp. 2001. Algorithms for graph partitioning on the planted partition model. *Random Structures and Algorithms* 18, 2 (2001), 116–140.

[15] Jernej Copic, Matthew O Jackson, and Alan Kirman. 2009. Identifying community structures from network data via maximum likelihood methods. *The BE Journal of Theoretical Economics* 9, 1 (2009).

[16] Michele Coscia, Fosca Giannotti, and Dino Pedreschi. 2011. A classification for community discovery methods in complex networks. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 4, 5 (2011), 512–546.

[17] J-J Daudin, Franck Picard, and Stéphane Robin. 2008. A mixture model for random graphs. *Statistics and computing* 18, 2 (2008), 173–183.

[18] Jordi Duch and Alex Arenas. 2005. Community detection in complex networks using extremal optimization. *Physical review E* 72, 2 (2005), 027104.

[19] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* 486, 3 (2010), 75–174.

[20] Santo Fortunato and Marc Barthélemy. 2007. Resolution limit in community detection. *Proceedings of the National Academy of Sciences* 104, 1 (2007), 36–41.

[21] Santo Fortunato and Darko Hric. 2016. Community detection in networks: A user guide. *Physics Reports* 659 (2016), 1–44.

[22] D. Golovin, B. Solnik, S. Moitra, G. Kochanski, J. Karro, and D. Sculley. 2017. Google vizier: A service for black-box optimization. In *International Conference on Knowledge Discovery and Data Mining*. ACM, 1487–1495.

[23] Roger Guimera and Luis A Nunes Amaral. 2005. Functional cartography of complex metabolic networks. *nature* 433(7028), 7028 (2005), 895–900.

[24] Matthew B Hastings. 2006. Community detection as an inference problem. *Physical Review E* 74, 3 (2006), 035102.

[25] Jake M Hofman and Chris H Wiggins. 2008. Bayesian approach to network modularity. *Physical review letters* 100, 25 (2008), 258701.

[26] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. 1983. Stochastic blockmodels: First steps. *Social networks* 5, 2 (1983), 109–137.

[27] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. 2011. Sequential model-based optimization for general algorithm configuration. In *International Conference on Learning and Intelligent Optimization*. Springer, 507–523.

[28] Brian Karrer and Mark EJ Newman. 2011. Stochastic blockmodels and community structure in networks. *Physical Review E* 83, 1 (2011), 016107.

[29] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. 2008. Benchmark graphs for testing community detection algorithms. *Physical review E* 78, 4 (2008), 046110.

[30] Sune Lehmann and Lars Kai Hansen. 2007. Deterministic modularity optimization. *The European Physical Journal B-Condensed Matter and Complex Systems* 60, 1 (2007), 83–88.

[31] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1, 1 (2007), 2.

[32] David Lusseau, Karsten Schneider, Oliver J Boisseau, Patti Haase, Elisabeth Slooten, and Steve M Dawson. 2003. The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology* 54, 4 (2003), 396–405.

[33] Fragkiskos D Malliaros and Michalis Vazirgiannis. 2013. Clustering and community detection in directed networks: A survey. *Physics Reports* 533, 4 (2013), 95–142.

[34] Aaron McDaid and Neil Hurley. 2010. Detecting highly overlapping communities with model-based overlapping seed expansion. In *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*. IEEE, 112–119.

[35] Michael Molloy and Bruce Reed. 1995. A critical point for random graphs with a given degree sequence. *Random structures & algorithms* 6, 2-3 (1995), 161–180.

[36] MEJ Newman. 2016. Community detection in networks: Modularity optimization and maximum likelihood are equivalent. *arXiv preprint arXiv:1606.02319* (2016).

[37] Mark EJ Newman. 2006. Finding community structure in networks using the eigenvectors of matrices. *Physical review E* 74, 3 (2006), 036104.

[38] Mark EJ Newman. 2006. Modularity and community structure in networks. *Proceedings of the national academy of sciences* 103, 23 (2006), 8577–8582.

[39] Mark EJ Newman and Aaron Clauset. 2016. Structure and inference in annotated networks. *Nature communications* 7 (2016).

[40] Mark EJ Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical review E* 69, 2 (2004), 026113.

[41] Mark EJ Newman and Elizabeth A Leicht. 2007. Mixture models and exploratory analysis in networks. *Proceedings of the National Academy of Sciences* 104, 23 (2007), 9564–9569.

[42] Krzysztof Nowicki and Tom A B Snijders. 2001. Estimation and prediction for stochastic blockstructures. *J. Amer. Statist. Assoc.* 96, 455 (2001), 1077–1087.

[43] Leto Peel, Daniel B Larremore, and Aaron Clauset. 2017. The ground truth about metadata and community detection in networks. *Science advances* 3, 5 (2017), e1602548.

[44] Tiago P Peixoto. 2015. Inferring the mesoscale structure of layered, edge-valued, and time-varying networks. *Physical Review E* 92, 4 (2015), 042807.

[45] José J Ramasco and Muhittin Mungan. 2008. Inversion method for content-based networks. *Physical Review E* 77, 3 (2008), 036122.

[46] William M Rand. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association* 66, 336 (1971), 846–850.

[47] Jörg Reichardt and Stefan Bornholdt. 2006. Statistical mechanics of community detection. *Physical Review E* 74, 1 (2006), 016110.

[48] Martin Rosvall and Carl T Bergstrom. 2008. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences* 105, 4 (2008), 1118–1123.

[49] Philipp Schuetz and Amedeo Caflisch. 2008. Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement. *Physical Review E* 77, 4 (2008), 046112.

[50] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. 2015. Scalable bayesian optimization using deep neural networks. In *International Conference on Machine Learning*. 2171–2180.

[51] Myra Spiliopoulou. 2011. Evolution in social networks: A survey. In *Social network data analytics*. Springer, 149–175.

[52] Lovro Šubelj and Marko Bajec. 2013. Model of complex networks based on citation dynamics. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 527–530.

[53] Yudong Sun, Bogdan Danila, K Josić, and Kevin E Bassler. 2009. Improved community structure detection using a modified fine-tuning strategy. *EPL (Europhysics Letters)* 86, 2 (2009), 28004.

[54] Mursel Tasgin, Amac Herdagdelen, and Haluk Bingol. 2007. Community detection in complex networks using genetic algorithms. *arXiv preprint arXiv:0711.0491* (2007).

[55] Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and computing* 17, 4 (2007), 395–416.

[56] Ken Wakita and Toshiyuki Tsurumi. 2007. Finding community structure in mega-scale social networks. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 1275–1276.

[57] Jaewon Yang and Jure Leskovec. 2014. Structure and overlaps of ground-truth communities in networks. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 2 (2014), 26.

[58] Wayne W Zachary. 1977. An information flow model for conflict and fission in small groups. *Journal of anthropological research* 33, 4 (1977), 452–473.

[59] Hugo Zanghi, Christophe Ambroise, and Vincent Miele. 2008. Fast online graph clustering via Erdős–Rényi mixture. *Pattern recognition* 41, 12 (2008), 3592–3599.

NetSMF: Large-Scale Network Embedding as Sparse Matrix Factorization

Jiezhong Qiu
Department of Computer Science and Technology, Tsinghua University
Beijing National Research Center for Information Science and Technology, China
qiujz16@mails.tsinghua.edu.cn

Yuxiao Dong
Microsoft Research, Redmond
yuxdong@microsoft.com

Hao Ma
Facebook AI
haoma@fb.com

Jian Li
Institute for Interdisciplinary Information Sciences, Tsinghua University
lijian83@tsinghua.edu.cn

Chi Wang
Microsoft Research, Redmond
wang.chi@microsoft.com

Kuansan Wang
Microsoft Research, Redmond
kuansanw@microsoft.com

Jie Tang
Department of Computer Science and Technology, Tsinghua University
Beijing National Research Center for Information Science and Technology, China
jietang@tsinghua.edu.cn

ABSTRACT

We study the problem of large-scale network embedding, which aims to learn latent representations for network mining applications. Previous research shows that 1) popular network embedding benchmarks, such as DeepWalk, are in essence implicitly factorizing a matrix with a closed form, and 2) the explicit factorization of such matrix generates more powerful embeddings than existing methods. However, directly constructing and factorizing this matrix—which is dense—is prohibitively expensive in terms of both time and space, making it not scalable for large networks.

In this work, we present the algorithm of large-scale network embedding as sparse matrix factorization (NetSMF). NetSMF leverages theories from spectral sparsification to efficiently sparsify the aforementioned dense matrix, enabling significantly improved efficiency in embedding learning. The sparsified matrix is spectrally close to the original dense one with a theoretically bounded approximation error, which helps maintain the representation power of the learned embeddings. We conduct experiments on networks of various scales and types. Results show that among both popular benchmarks and factorization based methods, NetSMF is the only method that achieves both high efficiency and effectiveness. We show that NetSMF requires only 24 hours to generate effective embeddings for a large-scale academic collaboration network with tens of millions of nodes, while it would cost DeepWalk months and is computationally infeasible for the dense matrix factorization solution. The source code of NetSMF is publicly available[1].

ACM Reference Format:
Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Chi Wang, Kuansan Wang, and Jie Tang. 2019. NetSMF: Large-Scale Network Embedding as Sparse Matrix Factorization . In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313446

1 INTRODUCTION

Recent years have witnessed the emergence of network embedding, which offers a revolutionary paradigm for modeling graphs and networks [16]. The goal of network embedding is to automatically learn latent representations for objects in networks, such as vertices and edges. Significant lines of research have shown that the latent representations are capable of capturing the structural properties of networks, facilitating various downstream network applications, such as vertex classification and link prediction [12, 14, 27, 33].

Over the course of its development, the DeepWalk [27], LINE [33], and node2vec [14] models have been commonly considered as powerful benchmark solutions for evaluating network embedding research. The advantage of LINE lies in its scalability for large-scale networks as it only models the first- and second-order proximities. That is to say, its embeddings lose the multi-hop dependencies in

*This paper was researched and written when the author was still at Microsoft Research.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313446

[1] https://github.com/xptree/NetSMF

Table 1: The comparison between NetSMF and other popular network embedding algorithms.

	LINE	DeepWalk	node2vec	NetMF	NetSMF
Efficiency	√				√
Global context		√	√	√	√
Theoretical guarantee				√	√
High-order proximity			√		

networks. DeepWalk and node2vec, on the other hand, leverage random walks on graphs and skip-gram [24] with large context sizes to model nodes further away (i.e., global structures). Consequently, it is computationally more expensive for DeepWalk and node2vec to handle large-scale networks. For example, with the default parameter settings [27], DeepWalk requires months to embed an academic collaboration network of 67 million vertices and 895 million edges[2]. The node2vec model, which performs high-order random walks, takes more time than DeepWalk to learn embeddings.

More recently, a study shows that both the DeepWalk and LINE methods can be viewed as implicit factorization of a closed-form matrix [28]. Building upon this theoretical foundation, the NetMF method was instead proposed to explicitly factorize this matrix, achieving more effective embeddings than DeepWalk and LINE. Unfortunately, it turns out that the matrix to be factorized is an $n \times n$ dense one with n being the number of vertices in the network, making it prohibitively expensive to directly construct and factorize for large-scale networks.

In light of these limitations of existing methods (See the summary in Table 1), we propose to study representation learning for large-scale networks with the goal of achieving efficiency, capturing global structural contexts, and having theoretical guarantees. Our idea is to find a sparse matrix that is spectrally close to the dense NetMF matrix implicitly factorized by DeepWalk. The sparsified matrix requires a lower cost for both construction and factorization. Meanwhile, making it spectrally close to the original NetMF matrix can guarantee that the spectral information of the network is maintained, and the embeddings learned from the sparse matrix is as powerful as those learned from the dense NetMF matrix.

In this work, we present the solution to network embedding learning as sparse matrix factorization (NetSMF). NetSMF comprises three steps. First, it leverages the spectral graph sparsification technique [7, 8] to find a sparsifier for a network's random-walk matrix-polynomial. Second, it uses this sparsifier to construct a matrix with significantly fewer non-zeros than, but spectrally close to, the original NetMF matrix. Finally, it performs randomized singular value decomposition to efficiently factorize the sparsified NetSMF matrix, yielding the embeddings for the network.

With this design, NetSMF offers both efficiency and effectiveness with guarantees, as the approximation error of the sparsified matrix is theoretically bounded. We conduct experiments in five networks, which are representative of different scales and types.

[2]With the default DeepWalk parameters (walk length: 40 and #walk per node: 80), 214+ billion nodes (67M×40×80) with a vocabulary size of 67 million are fed into skip-gram. As a reference, Mikolov et al. reported that training on Google News of 6 billion words and a vocabulary size of only 1 million cost 2.5 days with 125 CPU cores [24].

Table 2: Notations.

Notation	Description
G	input network
V	vertex set of G with $\|V\|$=n
E	edge set of G with $\|E\| = m$
$\boldsymbol{A}$	adjacency matrix of G
$\boldsymbol{D}$	degree matrix of G
$\mathrm{vol}\,(G)$	volume of G
b	number of negative samples
T	context window size
d	embedding dimension
$\boldsymbol{L}$	random-walk molynomial of G (Eq. (4))
$\widetilde{\boldsymbol{L}}$	$\boldsymbol{L}$'s sparsifier
$\boldsymbol{M}$	$\frac{1}{T}\sum_{r=1}^{T}(\boldsymbol{D}^{-1}\boldsymbol{A})^r\boldsymbol{D}^{-1}$
$\widetilde{\boldsymbol{M}}$	$\boldsymbol{M}$'s sparsifier
$\mathrm{trunc_log}^{\circ}\left(\frac{\mathrm{vol}(G)}{b}\boldsymbol{M}\right)$	NetMF matrix
$\mathrm{trunc_log}^{\circ}\left(\frac{\mathrm{vol}(G)}{b}\widetilde{\boldsymbol{M}}\right)$	NetMF matrix sparisifier
M	number of non-zeros in $\widetilde{\boldsymbol{L}}$
ϵ	approximation factor
$[x]$	set $\{1, 2, \cdots, x\}$ for positive integer x

Experimental results show that for million-scale or larger networks, NetSMF achieves orders of magnitude speedup over NetMF, while maintaining competitive performance for the vertex classification task. In other words, both NetSMF and NetMF outperform well-recognized network embedding benchmarks (i.e., DeepWalk, LINE, and node2vec), but NetSMF addresses the computation challenge faced by NetMF.

To summarize, we introduce the idea of network embedding as sparse matrix factorization and present the NetSMF algorithm, which makes the following contributions to network embedding:

Efficiency. NetSMF reaches significantly lower time and space complexity than NetMF. Remarkably, NetSMF is able to generate embeddings for a large-scale academic network of 67 million vertices and 895 million edges on a single server in 24 hours, while it would cost months for DeepWalk and node2vec, and is computationally infeasible for NetMF on the same hardware.

Effectiveness. NetSMF is capable of learning embeddings that maintain the same representation power as the dense matrix factorization solution, making it consistently outperform DeepWalk and node2vec by up to 34% and LINE by up to 100% for the multi-label vertex classification task in networks.

Theoretical Guarantee. NetSMF's efficiency and effectiveness are theoretically backed up. The sparse NetSMF matrix is spectrally close to the exact NetMF matrix, and the approximation error can be bounded, maintaining the representation power of its sparsely learned embeddings.

2 PRELIMINARIES

Commonly, the problem of network embedding is formalized as follows: Given an undirected and weighted network $G = (V, E, \boldsymbol{A})$ with V as the vertex set of n vertices, E as the edge set of m edges, and $\boldsymbol{A}$ as the adjacency matrix, the goal is to learn a function $V \rightarrow$

$\mathbb{R}^d$ that maps each vertex to a d-dimensional ($d \ll n$) vector that captures its structural properties, e.g., community structures. The vector representation of each vertex can be fed into downstream applications such as link prediction and vertex classification.

One of the pioneering work on network embedding is the DeepWalk model [27], which has been consistently considered as a powerful benchmark over the past years [16]. In brief, DeepWalk is coupled with two steps. First, it generates several vertex sequences by random walks over a network; Second, it applies the skip-gram model [25] on the generated vertex sequences to learn the latent representations for each vertex. Commonly, skip-gram is parameterized with the context window size T and the number of negative samples b. Recently, a theoretical study [28] reveals that DeepWalk essentially factorizes a matrix derived from the random walk process. More formally, it proves that when the length of random walks goes to infinity, DeepWalk implicitly and asymptotically factorizes the following matrix:

$$\log^\circ\left(\frac{\mathrm{vol}(G)}{b}\boldsymbol{M}\right), \tag{1}$$

where $\mathrm{vol}\,(G) = \sum_i \sum_j \boldsymbol{A}_{ij}$ denotes the volume of the graph, and

$$\boldsymbol{M} = \frac{1}{T}\sum_{r=1}^{T}(\boldsymbol{D}^{-1}\boldsymbol{A})^r\boldsymbol{D}^{-1}, \tag{2}$$

where $\boldsymbol{D} = \mathrm{diag}\,(d_1, \cdots, d_n)$ is the degree matrix with $d_i = \sum_j \boldsymbol{A}_{ij}$ as the generalized degree of the i-th vertex. Note that $\log^\circ(\cdot)$ represents the element-wise matrix logarithm [18], which is different from the matrix logarithm. In other words, the matrix in Eq. (1) can be characterized as the result of applying element-wise matrix logarithm (i.e., $\log^\circ$) to matrix $\frac{\mathrm{vol}\,(G)}{b}\boldsymbol{M}$.

The matrix in Eq. (1) offers an alternative view of the skip-gram based network embedding methods. Further, Qiu et al. provide an explicit matrix factorization approach named NetMF to learn the embeddings [28]. It shows that the accuracy for vertex classification based on the embeddings from NetMF outperforms that based on DeepWalk and LINE. Note that the matrix in Eq. (1) would be ill-defined if there exist a pair of vertices unreachable in T hops, because $\log(0) = -\infty$. So following Levy and Goldberg [22], NetMF uses the logarithm truncated at point one, that is, $\mathrm{trunc_log}(x) = \max(0, \log(x))$. Thus, NetMF targets to factorize the matrix

$$\mathrm{trunc_log}^\circ\left(\frac{\mathrm{vol}(G)}{b}\boldsymbol{M}\right). \tag{3}$$

In the rest of this work, we refer to the matrix in Eq. (3) as *the NetMF matrix*.

However, there exist a couple of challenges when leveraging the NetMF matrix in practice. First, almost every pair of vertices within distance $r \le T$ correspond to a non-zero entry in the NetMF matrix. Recall that many social and information networks exhibit the small-world property where most vertices can be reached from each other in a small number of steps. For example, as of the year 2012, 92% of the reachable pairs in Facebook are at distance five or less [2]. As a consequence, even if setting a moderate context window size (e.g., the default setting $T = 10$ in DeepWalk), the NetMF matrix in Eq. (3) would be a dense matrix with $O(n^2)$ number of non-zeros. The exact construction and factorization of such a matrix is impractical for large-scale networks. More concretely, computing the matrix power in Eq. (2) involves dense matrix multiplication which costs $O(n^3)$ time; factorizing a $n \times n$ dense matrix is also time consuming. To reduce the construction cost, NetMF approximates $\boldsymbol{M}$ with its top eigen pairs. However, the approximated matrix is still dense, making this strategy unable to handle large networks.

In this work, we aim to address the efficiency and scalability limitation of NetMF, while maintaining its superiority in effectiveness. We list necessary notations and their descriptions in Table 2.

3 NETWORK EMBEDDING AS SPARSE MATRIX FACTORIZATION (NetSMF)

In this section, we develop network embedding as sparse matrix factorization (NetSMF). We present the NetSMF method to construct and factorize a sparse matrix that approximates the dense NetMF matrix. The main technique we leverage is random-walk matrix-polynomial (molynomial) sparsification.

3.1 Random-Walk Molynomial Sparsification

We first introduce the definition of spectral similarity and the theorem of random-walk molynomial sparsification.

Definition 1. *(Spectral Similarity of Networks) Suppose* $G = (V, E, \boldsymbol{A})$ *and* $\widetilde{G} = (V, \widetilde{E}, \widetilde{\boldsymbol{A}})$ *are two weighted undirected networks. Let* $\boldsymbol{L} = \boldsymbol{D}_G - \boldsymbol{A}$ *and* $\widetilde{\boldsymbol{L}} = \boldsymbol{D}_{\widetilde{G}} - \widetilde{\boldsymbol{A}}$ *be their Laplacian matrices, respectively. We define* G *and* $\widetilde{G}$ *are* $(1+\epsilon)$*-spectrally similar if*

$$\forall \boldsymbol{x} \in \mathbb{R}^n, (1-\epsilon)\cdot \boldsymbol{x}^\top\widetilde{\boldsymbol{L}}\boldsymbol{x} \le \boldsymbol{x}^\top\boldsymbol{L}\boldsymbol{x} \le (1+\epsilon)\cdot\boldsymbol{x}^\top\widetilde{\boldsymbol{L}}\boldsymbol{x}.$$

Theorem 1. *(Spectral Sparsifiers of Random-Walk Molynomials [7, 8]) For random-walk molynomial*

$$\boldsymbol{L} = \boldsymbol{D} - \sum_{r=1}^{T}\alpha_r\boldsymbol{D}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^r, \tag{4}$$

where $\sum_{r=1}^{T}\alpha_r = 1$ *and* α_r *non-negative, one can construct, in time* $O(T^2m\epsilon^{-2}\log^2 n)$*, a* $(1+\epsilon)$*-spectral sparsifier,* $\widetilde{\boldsymbol{L}}$*, with* $O(n\log n\epsilon^{-2})$ *non-zeros. For unweighted graphs, the time complexity can be reduced to* $O(T^2m\epsilon^{-2}\log n)$*.*

To achieve a sparsifier $\widetilde{\boldsymbol{L}}$ with $O(\epsilon^{-2}n\log n)$ non-zeros, the sparsification algorithm consists of two steps: The first step obtains an initial sparsifier for $\boldsymbol{L}$ with $O(Tm\epsilon^{-2}\log n)$ non-zeros. The second step then applies the standard spectral sparsification algorithm [30] to further reduce the number of non-zeros to $O(\epsilon^{-2}n\log n)$. In this work, we only adopt the first step because a sparsifier with $O(Tm\epsilon^{-2}\log n)$ non-zeros is sparse enough for our task. Thus we skip the second step that involves additional computations. From now on, when referring to the random-walk molynomial sparsification algorithm in this work, we mean its first step only.

One can immediately observe that, if we set $\alpha_r = \frac{1}{T}, r \in [T]$, the matrix $\boldsymbol{L}$ in Eq. (4) has a strong connection with the desired matrix $\boldsymbol{M}$ in Eq. (2). Formally, we have the following equation

$$\boldsymbol{M} = \boldsymbol{D}^{-1}\left(\boldsymbol{D} - \boldsymbol{L}\right)\boldsymbol{D}^{-1}. \tag{5}$$

Thm. 1 can help us construct a sparsifier $\widetilde{\boldsymbol{L}}$ for matrix $\boldsymbol{L}$. Then we define $\widetilde{\boldsymbol{M}} \triangleq \boldsymbol{D}^{-1}(\boldsymbol{D} - \widetilde{\boldsymbol{L}})\boldsymbol{D}^{-1}$ by replacing $\boldsymbol{L}$ in Eq. (5) with its sparsifier $\widetilde{\boldsymbol{L}}$. One can observe that matrix $\widetilde{\boldsymbol{M}}$ is still a sparse one with the same order of magnitude of non-zeros as $\widetilde{\boldsymbol{L}}$. Consequently, instead of factorizing the dense NetMF matrix in Eq. (3), we can factorize its sparse alternative, i.e.,

$$\mathrm{trunc_log}^\circ\left(\frac{\mathrm{vol}(G)}{b}\widetilde{\boldsymbol{M}}\right). \tag{6}$$

In the rest of this work, the matrix in Eq. (6) is referred to as *the NetMF matrix sparsifier*.

3.2 The NetSMF Algorithm

In this section, we formally describe the NetSMF algorithm, which consists of three steps: random-walk molynomial sparsification, NetMF sparsifier construction, and truncated singular value decomposition.

Algorithm 1: NetSMF

Input : A social network $G = (V, E, \boldsymbol{A})$ which we want to learn network embedding; The number of non-zeros M in the sparsifier; The dimension of embedding d.

Output: An embedding matrix of size $n \times d$, each row corresponding to a vertex.

1 $\widetilde{G} \leftarrow (V, \emptyset, \widetilde{\boldsymbol{A}} = \boldsymbol{0})$

/* Create an empty network with $E = \emptyset$ and $\widetilde{\boldsymbol{A}} = 0$. */

2 **for** $i \leftarrow 1$ **to** M **do**

3 Uniformly pick an edge $e = (u, v) \in E$

4 Uniformly pick an integer $r \in [T]$

5 $u', v', Z \leftarrow$ PathSampling(e, r)

6 Add an edge $\left(u', v', \frac{2rm}{MZ}\right)$ to $\widetilde{G}$

/* Parallel edges will be merged into one edge, with their weights summed up together. */

7 **end**

8 Compute $\widetilde{\boldsymbol{L}}$ to be the unnormalized graph Laplacian of $\widetilde{G}$

9 Compute $\widetilde{\boldsymbol{M}} = \boldsymbol{D}^{-1}\left(\boldsymbol{D} - \widetilde{\boldsymbol{L}}\right)\boldsymbol{D}^{-1}$

10 $\boldsymbol{U}_d, \Sigma_d, \boldsymbol{V}_d \leftarrow$ RandomizedSVD(trunc_log$^\circ\left(\frac{\mathrm{vol}(G)}{b}\widetilde{\boldsymbol{M}}\right), d$)

11 **return** $\boldsymbol{U}_d\sqrt{\Sigma_d}$ as network embeddings

Algorithm 2: PathSampling algorithm as described in [8].

1 **Procedure** PathSampling($e = (u, v)$, r)

2 Uniformly pick an integer $k \in [r]$

3 Perform $(k-1)$-step random walk from u to u_0

4 Perform $(r-k)$-step random walk from v to u_r

5 Keep track of $Z(\boldsymbol{p})$ along the length-r path $\boldsymbol{p}$ between u_0 and u_r according to Eq. (7)

6 **return** $u_0, u_r, Z(\boldsymbol{p})$

Step 1: Random-Walk Molynomial Sparsification. To achieve the sparsifier $\widetilde{\boldsymbol{L}}$, we adopt the algorithm in Cheng et al. [8]. The algorithm starts from creating a network $\widetilde{G}$ that has the same vertex set as G and an empty edge set (Alg. 1, Line 1). Next, the algorithm constructs a sparsifier with $O(M)$ non-zeros by repeating the PathSampling algorithm for M times. In each iteration, it picks an edge $e \in E$ and an integer $r \in [T]$ uniformly (Alg. 1, Line 3-4). Then, the algorithm uniformly draws an integer $k \in [r]$ and performs $(k-1)$-step and $(r-k)$-step random walks starting from the two endpoints of edge e respectively (Alg. 2, Line 3-4). The above process samples a length-r path $\boldsymbol{p} = (u_0, u_1, \cdots, u_r)$. At the same time, the algorithm keeps track of $Z(\boldsymbol{p})$, which is defined by

$$Z(\boldsymbol{p}) = \sum_{i=1}^{r} \frac{2}{\boldsymbol{A}_{u_{i-1}, u_i}}, \tag{7}$$

and then adds a new edge (u_0, u_r) with weight $\frac{2rm}{MZ(\boldsymbol{p})}$ to $\widetilde{G}$ (Alg. 1, Line 6).[3] Parallel edges in $\widetilde{G}$ will be merged into one single edge, with their weights summed up together. Finally, the algorithm computes the Laplacian of $\widetilde{G}$, which is the sparsifier $\widetilde{\boldsymbol{L}}$ as we desired (Alg. 1, Line 8). This step gives us a sparsifier with $O(M)$ non-zeros.

Step 2: Construct a NetMF Matrix Sparsifier. As we have discussed at the end of Section 3.1, after constructing a sparsifier $\widetilde{\boldsymbol{L}}$, we can plug it into Eq. (5) to obtain a NetMF matrix sparsifier as shown in Eq. (6) (Alg. 1, Line 9-10). This step does not change the order of magnitude of non-zeros in the sparsifier.

Algorithm 3: Randomized SVD on NetMF Matrix Sparsifier

/* In this work, the matrix to be factorized (Eq. (6)) is an $n \times n$ symmetric sparse matrix. We store this sparse matrix in a row-major way and make use of its symmetry to simplify the computation. */

1 **Procedure** RandomizedSVD($\boldsymbol{A}$, d)

2 Sampling Gaussian random matrix $\boldsymbol{O}$ // $\boldsymbol{O} \in \mathbb{R}^{n \times d}$

3 Compute sample matrix $\boldsymbol{Y} = \boldsymbol{A}^\top \boldsymbol{O} = \boldsymbol{A}\boldsymbol{O}$ // $\boldsymbol{Y} \in \mathbb{R}^{n \times d}$

4 Orthonormalize $\boldsymbol{Y}$

5 Compute $\boldsymbol{B} = \boldsymbol{A}\boldsymbol{Y}$ // $\boldsymbol{B} \in \mathbb{R}^{n \times d}$

6 Sample another Gaussian random matrix $\boldsymbol{P}$ // $\boldsymbol{P} \in \mathbb{R}^{d \times d}$

7 Compute sample matrix of $\boldsymbol{Z} = \boldsymbol{B}\boldsymbol{P}$ // $\boldsymbol{Z} \in \mathbb{R}^{n \times d}$

8 Orthonormalize $\boldsymbol{Z}$

9 Compute $\boldsymbol{C} = \boldsymbol{Z}^\top \boldsymbol{B}$ // $\boldsymbol{C} \in \mathbb{R}^{d \times d}$

10 Run Jacobi SVD on $\boldsymbol{C} = \boldsymbol{U}\Sigma\boldsymbol{V}^\top$

11 **return** $\boldsymbol{Z}\boldsymbol{U}$, Σ, $\boldsymbol{Y}\boldsymbol{V}$

/* Result matrices are of shape $n \times d, d \times d, n \times d$ resp. */

Step 3: Truncated Singular Value Decomposition. The final step is to perform truncated singular value decomposition (SVD) on the constructed NetMF matrix sparsifier (Eq. (6)). However, even the sparsifier only has $O(M)$ number of non-zeros, performing exact SVD is still time consuming. In this work, we leverage a modern randomized matrix approximation technique—Randomized SVD—developed by Halko et al. [15]. Due to space constraint, we cannot include many details. Briefly speaking, the algorithm projects the original matrix to a low-dimensional space through a Gaussian random matrix. One only needs to perform traditional SVD (e.g. Jacobi SVD) on a $d \times d$ small matrix. We list the pseudocode algorithm in Alg. 3. Another advantage of SVD is that we can determine the dimensionality of embeddings by using, for example, Cattell's Scree test [5]. In the test, we plot the singular values and select a rank

[3] Details about how the edge weight is derived can be found in Thm. 4 in Appendix.

Table 3: Time and Space Complexity of NetSMF.

	Time	Space
Step 1	$O(MT \log n)$ for weighted networks $O(MT)$ for unweighted networks	$O(M + n + m)$
Step 2	$O(M)$	$O(M + n)$
Step 3	$O(Md + nd^2 + d^3)$	$O(M + nd)$

d such that there is a clear drop in the magnitudes or the singular values start to even out. More details will be discussed in Section 4.

Complexity Analysis. Now we analyze the time and space complexity of NetSMF, as summarized in Table 3. As for step 1, we call the PathSampling algorithm for M times, during each of which it performs $O(T)$ steps of random walks over the network. For unweighted networks, sampling a neighbor requires $O(1)$ time, while for weighted networks, one can use roulette wheel selection to choose a neighbor in $O(\log n)$. It taks $O(M)$ space to store $\widetilde{G}$, while the additional $O(n + m)$ space comes from the storage of the input network. As for step 2, it takes $O(M)$ time to perform the transformation in Eq. (5) and the element-wise truncated logarithm in Eq. (6). The additional $O(n)$ space is spent in storing the degree matrix. As for step 3, $O(Md)$ time is required to compute the product of a row-major sparse matrix and a dense matrix (Alg. 3, Lines 3 and 5); $O(nd^2)$ time is spent in Gram-Schmidt orthogonalization (Alg. 3, Lines 4 and 8); $O(d^3)$ time is spent in Jacobi SVD (Alg. 3, Line 10).

Connection to NetMF. The major difference between NetMF and NetSMF lies in the approximation strategy of the NetMF matrix in Eq. (3). As we mentioned in Section 2, NetMF approximates it with a dense matrix, which brings new space and computation challenges. In this work, NetSMF aims to find a sparse approximator to the NetMF matrix by leveraging theories and techniques from spectral graph sparsification.

Example. We provide a running example to help understand the NetSMF algorithm. Suppose we want to learn embeddings for a network with $n = 10^6$ vertices, $m = 10^7$ edges, context window size $T = 10$, and approximation factor $\epsilon = 0.1$. The NetSMF method calls the PathSampling algorithm for $M = Tm\epsilon^{-2} \log n \approx 1.4 \times 10^{11}$ times and provides us with a NetMF matrix sparsifier with at most 1.4×10^{11} non-zeros (Notice that the reducer in Step 1 and trunc_log$^\circ$ in Step 2 will further sparsify the matrix, making 1.4×10^{11} an upper bound). The density of the sparsifier is at most $\frac{M}{n^2} \approx 14\%$. Then, when computing the sparse-dense matrix product in randomized SVD (Alg. 3, Lines 3 and 5), the sparseness of the factorized matrix can greatly accelerate the calculation. In comparison, NetMF must construct a dense matrix with $n^2 = 10^{12}$ non-zeros, which is an order of magnitude larger in terms of density. Also, the density of the sparsifier in NetSMF can be further reduced by using a larger ϵ, while NetMF does not have this flexibility.

3.3 Approximation Error Analysis

In this section, we analyze the approximation error of the sparsification. We assume that we choose an approximation factor $\epsilon < 0.5$. We first see how the constructed $\widetilde{\boldsymbol{M}}$ approximates $\boldsymbol{M}$ and then compare the NetMF matrix (Eq. (3)) against the NetMF matrix sparsifier (Eq. (6)). We use σ_i to denote the i-th descending-order singular value of a matrix. We also assume the vertices' degrees are sorted in ascending order, that is, $d_{\min} = d_1 \le d_2 \cdots \le d_n$.

Theorem 2. *The singular value of $\widetilde{\boldsymbol{M}} - \boldsymbol{M}$ satisfies $\sigma_i(\widetilde{\boldsymbol{M}} - \boldsymbol{M}) \le \frac{4\epsilon}{\sqrt{d_i d_{\min}}}, \forall i \in [n]$.*

Proof. See Appendix. □

Theorem 3. *Let $\|\cdot\|_F$ be the matrix Frobenius norm. Then*

$$\left\| \text{trunc_log}^\circ\left(\frac{\text{vol}(G)}{b}\widetilde{\boldsymbol{M}}\right) - \text{trunc_log}^\circ\left(\frac{\text{vol}(G)}{b}\boldsymbol{M}\right) \right\|_F \le \frac{4\epsilon\,\text{vol}(G)}{b\sqrt{d_{\min}}} \sqrt{\sum_{i=1}^{n} \frac{1}{d_i}}.$$

Proof. See Appendix. □

Discussion on the Approximation Error. The above bound is achieved without making assumptions about the input network. If we introduce some assumptions, say a bounded lowest degree $d_{\min}$ or a specific random graph model (e.g., Planted Partition Model or Extended Planted Partition Model), it is promising to explore tighter bounds by leveraging theorems in literature [6, 11].

3.4 Parallelization

Each step of NetSMF can be parallelized, enabling it to scale to very large networks. The parallelization design of NetSMF is introduced in Figure 1. Below we discuss the parallelization of each step in detail.

At the first step, the paths in the PathSampling algorithm are sampled independently with each other. Thus we can launch multiple PathSampling workers simultaneously. Each worker handles a subset of the samples. Herein, we require that each worker is able to access the network data $G = (V, E, \boldsymbol{A})$ efficiently. There are many options to meet this requirement. The easiest one is to load a copy of the network data to each worker's memory. When the network is extremely large (e.g., trillion scale) or workers have memory constraints, the graph engine should be designed to expose efficient graph query APIs to support graph operations such as random walks. At the end of this step, a reducer is designed to merge parallel edges and sum up their weights. If this step is implemented in a big data system such as Spark [45], the reduction step can be simply achieved by running a reduceByKey(_+_)[4] function. After the reduction, the sparsifier $\widetilde{\boldsymbol{L}}$ is organized as a collection of triplets, a.k.a, COOrdinate format, with each indicating an entry of the sparsifier.

The second step is the most straightforward step to scale up. When processing a triplet (u, v, w), we can simply query the degree of vertices u and v and perform the transformation defined in Eq. (5) as well as the truncated logarithm in Eq. (6), which can be well parallelized.

For the last step, we organize the sparsifier into row-major format. This format allows efficient multiplication between a sparse and a dense matrix (Alg. 3, Line 3 and 5). Other dense matrix operators (e.g., Gaussian random matrix generation, Gram-Schmidt

[4] https://spark.apache.org/docs/latest/rdd-programming-guide.html

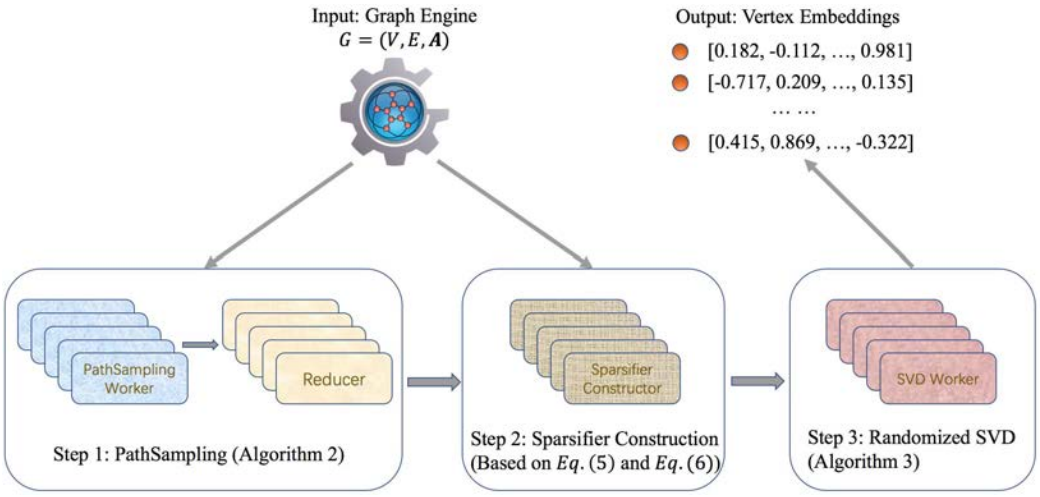


Figure 1: The System Design of NetSMF. The input comes from a graph engine which stores the network data and provides efficient APIs to graph queries. In Step 1, the system launches several PathSampling workers. Each worker handles a subset of samples. Then, a reducer is designed to aggregate the output of the PathSampling algorithm. In Step 2, the system distributes data to several sparsifier constructors to perform the transformation defined in Eq. (5) and the truncated element-wise matrix logarithm in Eq. (6). In the final step, the system applies truncated randomized SVD on the constructed sparsifier and dumps the resulted embeddings to storage.

Table 4: Statistics of Datasets.

Dataset	BlogCatalog	PPI	Flickr	YouTube	OAG
$\|V\|$	10,312	3,890	80,513	1,138,499	67,768,244
$\|E\|$	333,983	76,584	5,899,882	2,990,443	895,368,962
#labels	39	50	195	47	19

orthogonalization and Jacobi SVD) can be easily accelerated by using multi-threading or common linear algebra libraries.

In this work, we adopt a single-machine shared-memory implementation. We use OpenMP [10] to parallelize NetSMF in our implementation[5].

4 EXPERIMENTS

In this section, we evaluate the proposed NetSMF method on the multi-label vertex classification task, which has been commonly used to evaluate previous network embedding techniques [14, 27, 28, 33]. We introduce our datasets and baselines in Section 4.1 and Section 4.2. We report experimental results and parameter analysis in Section 4.3 and Section 4.4, respectively.

4.1 Datasets

We employ five datasets for the prediction task, four of which are in relatively small scale but have been widely used in network embedding literature, including BlogCatalog, PPI, Flickr, and YouTube. The remaining one is a large-scale academic co-authorship network, which is at least two orders of magnitude larger than the largest one (YouTube) used in most network embedding studies. The statistics of these datasets are listed in Table 4.

BlogCatalog [1, 35] is a network of social relationships of online bloggers. The vertex labels represent the interests of the bloggers.

Protein-Protein Interactions (PPI) [31] is a subgraph of the PPI network for Homo Sapiens. The vertex labels are obtained from the hallmark gene sets and represent biological states.

Flickr [35] is the user contact network in Flickr. The labels represent the interest groups of the users.

YouTube [36] is a video-sharing website that allows users to upload, view, rate, share, add to their favorites, report, comment on videos. The users are labeled by the video genres they liked.

Open Academic Graph (OAG) [6] is a publicly available academic graph indexed by Microsoft Academic [29] and AMiner.org [34]. We construct an undirected co-authorship network from OAG. This network contains 67,768,244 authors and 895,368,962 collaboration edges. The vertex labels are defined to be the top-level fields of study of each author, such as computer science, physics, psychology, and so on. In total, there are 19 distinct fields (labels) and authors may publish in more than one field, making the associated vertices have multiple labels.

4.2 Baseline Methods

We compare NetSMF with NetMF [28], LINE [33], DeepWalk [27], and node2vec [14]. For NetSMF, NetMF, DeepWalk, and node2vec that allow multi-hop structural dependencies, the context window size T is set to be 10, which is also the default setting used in both DeepWalk and node2vec. Across all datasets, we set the embedding

[5] Code is publicly available at https://github.com/xptree/NetSMF

[6] www.openacademic.ai/oag/

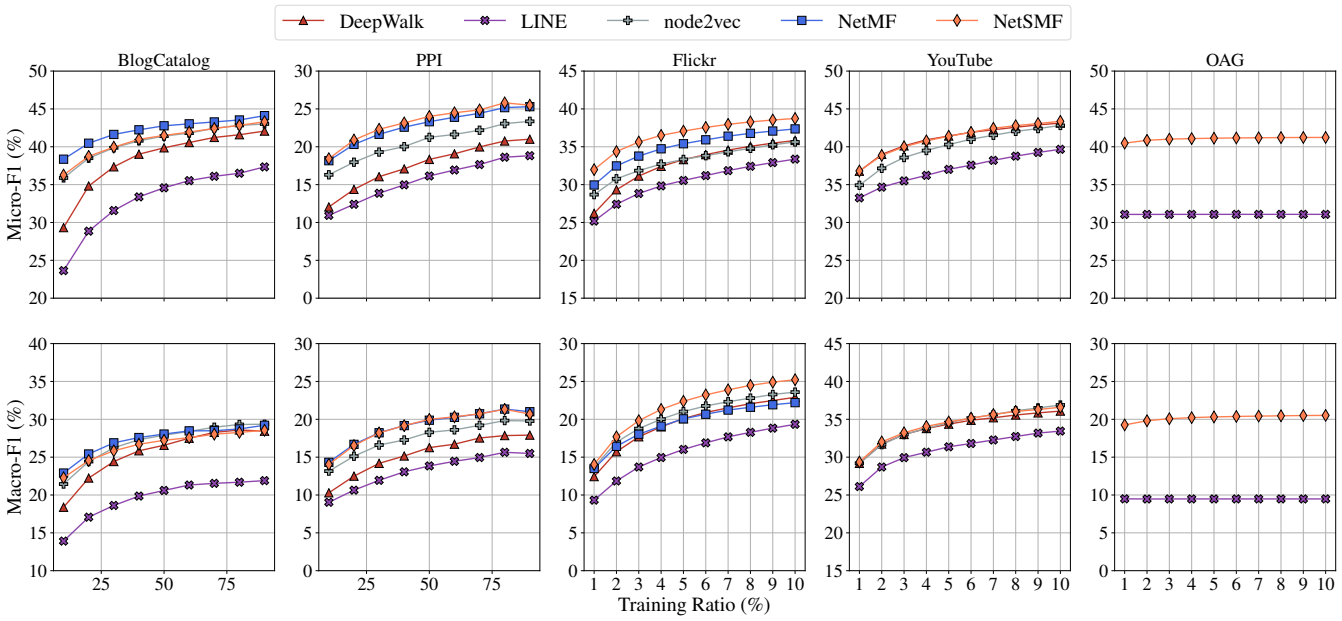


Figure 2: Predictive performance w.r.t. the ratio of training data. The x-axis represents the ratio of labeled data (%), and the y-axis in the top and bottom rows denote the Micro-F1 and Macro-F1 scores (%) respectively. For methods which fail to finish computation in one week or cannot handle the computation, their results are not available and thus not plotted in this figure.

dimension d to be 128. We follow the common practice for the other hyper-parameter settings, which are introduced below.

LINE. We use LINE with the second order proximity (i.e., LINE (2nd) [33]). We use the default setting of LINE's hyper-parameters: the number of edge samples to be 10 billion and the negative sample size to be 5.

DeepWalk. We present DeepWalk's results with the authors' preferred parameters, that is, walk length to be 40, the number of walks from each vertex to be 80, and the number of negative samples in skip-gram to be 5.

node2vec. For the return parameter p and in-out parameter q in node2vec, we adopt the default setting that was used by its authors if available. Otherwise, we grid search $p, q \in \{0.25, 0.5, 1, 2, 4\}$. For a fair comparison, we use the same walk length and the number of walks per vertex as DeepWalk.

NetMF. In NetMF, the hyper-parameter h indicates the number of eigen pairs used to approximate the NetMF matrix. Following the authors' preferred setting [28], we choose $h = 256$ for the BlogCatalog and PPI datasets, and $h = 16384$ for Flickr. For both NetMF and NetSMF, we have $b = 1$.

NetSMF. In NetSMF, we set the number of samples $M = 10^3 \times T \times m$ for the PPI, Flickr, and YouTube datasets, $M = 10^4 \times T \times m$ for BlogCatalog, and $M = 10 \times T \times m$ for OAG in order to achieve desired performance.

Prediction Setting. We follow exactly the same experiment and evaluation procedures that were performed in DeepWalk [27]. First, we randomly sample a portion of labeled vertices for training and use the remaining for testing. For the BlogCatalog and PPI datasets, the training ratio varies from 10% to 90%. For Flickr, YouTube and OAG, the training ratio varies from 1% to 10%. We use the one-vs-rest logistic regression model implemented by LIBLINEAR [13] for the multi-label vertex classification task. In the test phase, the one-vs-rest model yields a ranking of labels rather than an exact label assignment. To avoid the thresholding effect, we take the assumption that was made in DeepWalk, LINE, and node2vec, that is, the number of labels for vertices in the test data is given [14, 27, 37]. We repeat the prediction procedure ten times and evaluate the average performance in terms of both Micro-F1 and Macro-F1 scores [41]. All the experiments are performed on a server with Intel Xeon E7-8890 CPU (64 cores), 1.7TB memory, and 2TB SSD hard drive.

4.3 Experimental Results

We summarize the prediction performance in Figure 2. To compare the efficiency of different algorithms, we also list the running time of each algorithm across all datasets, if available, in Table 5.

NetSMF vs. NetMF. We first focus on the comparison between NetSMF and NetMF, since the goal of NetSMF is to address the efficiency and scalability issues of NetMF while maintaining its superiority in effectiveness. From Table 5, we observe that for YouTube and OAG, both of which contain more than one million vertices, NetMF fails to complete because of the excessive space and memory consumption, while NetSMF is able to finish in four hours and one day, respectively. For the moderate-size network Flickr, both methods are able to complete within one week, though NetSMF

Table 5: Efficiency comparison. The running time includes filesystem IO and computation time. "–" indicates that the corresponding algorithm fails to complete within one week. "×" indicates that the corresponding algorithm is unable to handle the computation due to excessive space and memory consumption.

	LINE	DeepWalk	node2vec	NetMF	NetSMF
BlogCatalog	40 mins	12 mins	56 mins	19 mins	13 mins
PPI	41 mins	4 mins	4 mins	1 min	10 secs
Flickr	42 mins	2.2 hours	21 hours	5 days	48 mins
YouTube	46 mins	4.3 hours	4 days	×	4.1 hours
OAG	2.6 hours	–	–	×	24 hours

is 150× faster (i.e., 48 mins vs. five days). For the two small-scale networks—BlogCatalog and PPI, NetSMF beats NetMF by a relatively narrow margin. This is because when the input networks contain only thousands of vertices, the advantage of sparse matrix construction and factorization over its dense alternative could be marginalized by the other components of the workflow, such as filesystem IO.

In terms of prediction performance, Figure 2 suggests NetSMF and NetMF yield consistently the best results among all compared methods, empirically demonstrating the power of the matrix factorization framework for network embedding. In BlogCatalog, NetSMF has slightly worse performance than NetMF (on average less than 3% worse regarding both Micro- and Macro-F1). In PPI, the two leading methods' performance are relatively indistinguishable in terms of both metrics. In Flickr, NetSMF achieves significantly better Macro-F1 than NetMF (by 11% on average), and also higher Micro-F1 (by 4.8% on average). Recall that NetMF uses a dense approximation of the matrix to factorize. These results show that the sparse spectral approximation used by NetSMF does not necessarily yield worse performance than the dense approximation used by NetMF.

Overall, not only NetSMF improves the scalability, and the running time of NetMF by orders of magnitude for large-scale networks, it also has competitive, and sometimes better, performance. This demonstrates the effectiveness of our spectral sparsification based approximation algorithm.

NetSMF vs. DeepWalk, LINE & node2vec. We also compare NetSMF against common graph embedding benchmarks—DeepWalk, LINE, and node2vec. For the OAG dataset, DeepWalk and node2vec fail to finish the computation within one week, while NetSMF requires only 24 hours. Based on the publicly reported running time of skip-gram [24], we estimate that DeepWalk and node2vec may require months to generate embeddings for the OAG dataset. In BlogCatalog and YouTube, DeepWalk and NetSMF require similar computing time, while in Flickr and PPI, NetSMF is 3× and 24× faster than DeepWalk, respectively. In all the datasets, NetSMF achieves 4–25× speedup over node2vec.

Moreover, the performance of NetSMF is significantly better than DeepWalk in BlogCatalog, PPI, and Flickr, by 7–34% in terms of Micro-F1 and 5–25% in terms of Macro-F1. In YouTube, NetSMF achieves slightly better results than DeepWalk. Compared with node2vec, NetSMF achieves comparable performance in BlogCatalog and YouTube, and significantly better performance in PPI and Flickr. In sum, NetSMF consistently outperforms DeepWalk and node2vec in terms of both efficiency and effectiveness.

LINE has the best efficiency among all the five methods and together with NetSMF, they are the only methods that can generate embeddings for OAG within one week (and both finish in one day). However, it also has the worst prediction performance and consistently loses to others by a large margin across all datasets. For example, NetSMF beats LINE by 21% and 39% in Flickr, and by 30% and 100% in OAG in terms of Micro-F1 and Macro-F1, respectively.

In summary, LINE achieves efficiency at the cost of ignoring multi-hop dependencies in networks, which are supported by all the other four methods—DeepWalk, node2vec, NetMF, and NetSMF, demonstrating the importance of multi-hop dependencies for learning network representations.

More importantly, among these four methods, DeepWalk achieves neither efficiency nor effectiveness superiority; node2vec achieves relatively good performance at the cost of efficiency; NetMF achieves effectiveness at the expense of significantly increased time and space costs; NetSMF is the only method that achieves both high efficiency and effectiveness, empowering it to learn effective embeddings for billion-scale networks (e.g., the OAG network with 0.9 billion edges) in one day on one modern server.

4.4 Parameter Analysis

In this section, we discuss how the hyper-parameters influence the performance and efficiency of NetSMF. We report all the parameter analyses on the Flickr dataset with training ratio set to be 10%.

How to Set the Embedding Dimension d. As mentioned in Section 3.1, SVD allows us to determine a "good" embedding dimension without additional supervised information. There are many methods available such as captured energy and Cattell's Scree test [5]. Here we propose to use Cattell's Scree test. Cattell's Scree test plots the singular values and selects a rank d such that there is a clear drop in the magnitudes or the singular values start to even out. In Flickr, if we sort the singular values in decreasing order, we can observe that the singular values approach 0 when the rank increases to around 100, as shown in Figure 3(b). In our experiments, by varying d form 2^4 to 2^8, we reach the best performance at $d = 128$, as shown in Figure 3(a), demonstrating the ability of our matrix factorization based NetSMF for automatically determining the embedding dimension.

The Number of Non-Zeros M. In theory, $M = O(Tm\epsilon^{-2} \log n)$ is required to guarantee the approximation error (See Section 3.1). Without loss of generality, we empirically set M to be $k \times T \times m$ where k is chosen from 1, 10, 100, 200, 500, 1000, 2000 and investigate how the number of non-zeros influence the quality of learned embeddings. As shown in Figure 3(c), when increasing the number of non-zeros, NetSMF tends to have better prediction performance because the original matrix is being approximated more accurately. On the other hand, although increasing M has a positive effect on the prediction performance, its marginal benefit diminishes gradually. One can observe that setting $M = 1000 \times T \times m$ (the

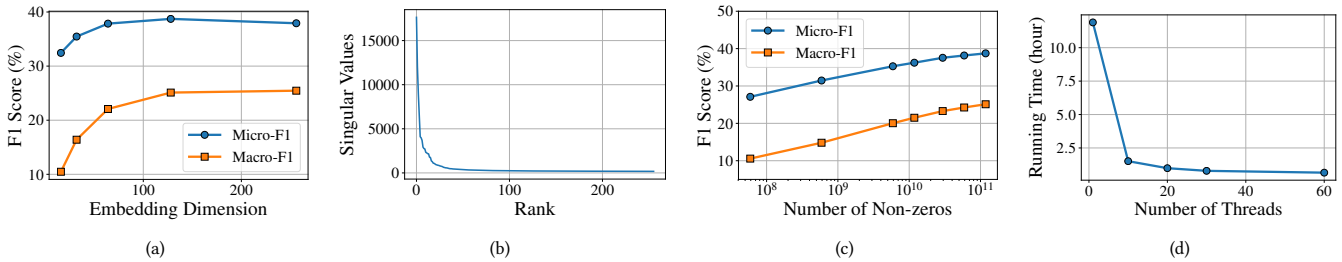


Figure 3: Parameter analysis: (a) Prediction performance v.s. embedding dimension d; (b) Cattel's Scree Test on singular values. We plot the top 256 singular values to find a clear drop in the magnitudes or the singular values start to even out. (c) Prediction performance v.s. the number of non-zeros M; (d) Running time v.s. the number of threads.

second-to-the-right data point on each line in Figure 3(c)) is a good choice that balances NetSMF's efficiency and effectiveness.

The Number of Threads. In this work, we use a single-machine shared memory implementation with multi-threading acceleration. We report the running time of NetSMF when setting the number of threads to be 1, 10, 20, 30, 60, respectively. As shown in Figure 3(d), NetSMF takes 12 hours to embed the Flickr network with one thread and 48 minutes to run with 30 threads, achieving a 15× speedup ratio (with ideal being 30×). This relatively good sub-linear speedup supports NetSMF to scale up to very large-scale networks.

5 RELATED WORK

In this section, we briefly review the related work of network embedding, large-scale embedding algorithms, and spectral graph sparsification.

5.1 Network Embedding

Network embedding has been extensively studied over the past years [16]. The success of network embedding has driven a lot of downstream network applications, such as recommendation systems [44]. Briefly, recent work about network embedding can be categorized into three genres: (1) Skip-gram based methods that are inspired by word2vec [24], such as LINE [33], DeepWalk [27], node2vec [14], metapath2vec [12], and VERSE [40]; (2) Deep learning based methods such as [21, 44]; (3) Matrix factorization based methods such as GraRep [4] and NetMF [28]. Among them, NetMF bridges the first and the third categories by unifying a collection of skip-gram based network embedding methods into a matrix factorization framework. In this work, we leverage the merit of NetMF and address its limitation in efficiency.

Among literature, PinSage is notably a network embedding framework for billion-scale networks [44]. The difference between NetSMF and PinSage lies in the following aspect. The goal of NetSMF is to pre-train general network embeddings in an unsupervised manner, while PinSage is a supervised graph convolutional method with both the objective of recommender systems and existing node features incorporated. That being said, the embeddings learned by NetSMF can be consumed by PinSage for downstream network applications.

5.2 Large-Scale Embedding Learning

Studies have attempted to optimize embedding algorithms for large datasets from different perspectives. Some of them focus on improving the skip-gram model, while others consider it as a matrix factorization problem.

Distributed Skip-Gram Model. Inspired by word2vec [25], most of the modern embedding learning algorithms are based on the skip-gram model. There is a sequence of work trying to accelerate the skip-gram model in a distributed system. For example, Ji et al. [19] replicate the embedding matrix on multiple workers and synchronize them periodically; Ordentlich et al. [26] distribute the columns (dimensions) of the embedding matrix to multiple executors and synchronize them with a parameter server [23]. Negative sampling is a key step in skip-gram, which requires to draw samples from a noisy distribution. Stergiou et al. [32] focus on the optimization of negative sampling by replacing the roulette wheel selection with a hierarchical sampling algorithm based on the alias method. More recently, Wang et al. [43] propose a billion-scale network embedding framework by heuristically partitioning the input graph to small subgraphs, and processing them separately in parallel. However, the performance of their framework highly relies on the quality of graph partition. The drawback for partition-based embedding learning is that the embeddings learned in different subgraphs do not share the same latent space, making it impossible to compare nodes across subgraphs.

Efficient Matrix Factorization. Factorizing the NetMF matrix, either implicitly (e.g., LINE [33] and DeepWalk [27]) or explicitly (e.g., NetMF [28]), encounters the following two issues. First, the denseness of this matrix makes computation expensive even for a moderate context window size (e.g., $T = 10$). Second, the non-linear transformation, i.e., element-wise matrix logarithm, is hard to approximate. LINE [33] solves this problem by setting $T = 1$. With such simplification, it achieves good scalability at the cost of prediction performance. NetSMF addresses these issues by efficiently sparsifying the dense NetMF matrix with a theoretically-bounded approximation error.

5.3 Spectral Graph Sparsification

Spectral graph sparsification has been studied for decades in graph theory [38]. The task of graph sparsification is to approximate a

"dense" graph by a "sparse" one that can be effectively used in place of the dense one [38], which arises in many applications such as scientific computing [17], machine learning [3, 7] and data mining [46]. Our NetSMF model is the first work that incorporates spectral sparsification algorithms [7, 8] into network embedding, which offers a powerful and efficient way to approximate and analyze the random-walk matrix-polynomial in the NetMF matrix.

6 CONCLUSION

In this work, we study network embedding with the goal of achieving both good efficiency and effectiveness. To address the efficiency and scalability challenges faced by the NetMF model, we propose to study large-scale network embedding as sparse matrix factorization. We present the NetSMF algorithm, which achieves a sparsification of the (dense) NetMF matrix. Both the construction and factorization of the sparsified matrix are fast enough to support very large-scale network embedding learning. For example, it empowers NetSMF to efficiently embed the Open Academic Graph in 24 hours, whose size is computationally intractable for the dense matrix factorization solution (NetMF).

Theoretically, the sparsified matrix is spectrally close to the original NetMF matrix with an approximation bound. Empirically, our extensive experimental results show that the sparsely learned embeddings by NetSMF are as effective as those from the factorization of the NetMF matrix, leaving it outperform the common network embedding benchmarks—DeepWalk, LINE, and node2vec. In other words, among both matrix factorization based methods (NetMF and NetSMF) and common skip-gram based benchmarks (DeepWalk, LINE, and node2vec), NetSMF is the only model that achieves both efficiency and performance superiority.

Future Work. NetSMF brings an efficient, effective, and guaranteed solution to network embedding learning. There are multiple tangible research fronts we can pursue. First, our current single-machine implementation limits the number of samples we can take for large networks. We plan to develop a multi-machine solution in the future to further scale NetSMF. Second, building upon NetSMF, we would like to efficiently and accurately learn embeddings for large-scale directed [9], dynamic [20], and/or heterogeneous networks. Third, as the advantage of matrix factorization methods demonstrated, we are also interested in exploring the other matrix definitions that may be effective in capturing different structural properties in networks. Last, it would be also interesting to bridge matrix factorization based network embedding methods with graph convolutional networks.

Acknowledgements. We would like to thank Dehua Cheng and Youwei Zhuo from USC for helpful discussions. Jian Li is supported in part by the National Basic Research Program of China Grant 2015CB358700, the National Natural Science Foundation of China Grant 61822203, 61772297, 61632016, 61761146003, and a grant from Microsoft Research Asia. Jie Tang is the corresponding author.

APPENDIX

We first prove Thm. 2 and Thm. 3 in Section 3.3. The following lemmas will be useful in our proof.

Lemma 1. *([39]) Singular values of a real symmetric matrix are the absolute values of its eigenvalues.*

Lemma 2. *(Courant-Fisher Theorem) Let $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ be a symmetric matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, then for $i \in [n]$,*

$$\lambda_i = \min_{\dim(U)=i} \max_{\boldsymbol{x}\in U, \|x\|_2=1} \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x}.$$

Lemma 3. *([18]) Let $\boldsymbol{B}, \boldsymbol{C}$ be two $n \times n$ symmetric matrices. Then for the decreasingly-ordered singular values σ of $\boldsymbol{B}, \boldsymbol{C}$ and $\boldsymbol{BC}$,*

$$\sigma_{i+j-1}(\boldsymbol{BC}) \leq \sigma_i(\boldsymbol{B}) \times \sigma_j(\boldsymbol{C})$$

holds for any $1 \leq i, j \leq n$ and $i + j \leq n + 1$.

Lemma 4. *Let $\boldsymbol{\mathcal{L}} = \boldsymbol{D}^{-1/2}\boldsymbol{L}\boldsymbol{D}^{-1/2}$ and similarly $\widetilde{\boldsymbol{\mathcal{L}}} = \boldsymbol{D}^{-1/2}\widetilde{\boldsymbol{L}}\boldsymbol{D}^{-1/2}$. Then all the singular values of $\widetilde{\boldsymbol{\mathcal{L}}} - \boldsymbol{\mathcal{L}}$ are smaller than 2ϵ, i.e., $\forall i \in [n]$, $\sigma_i(\widetilde{\boldsymbol{\mathcal{L}}} - \boldsymbol{\mathcal{L}}) < 4\epsilon$.*

Proof. Notice that

$$\boldsymbol{\mathcal{L}} = \boldsymbol{D}^{-1/2}\boldsymbol{L}\boldsymbol{D}^{-1/2} = \boldsymbol{I} - \sum_{r=1}^{T} \alpha_r \left(\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\right)^r$$

which is a normalized graph Laplacian whose eigenvalues lie in the interval $[0, 2)$, i.e., for $i \in [n]$, $\lambda_i(\boldsymbol{\mathcal{L}}) \in [0, 2)$ [42]. Since $\widetilde{\boldsymbol{L}}$ is a $(1 + \epsilon)$-spectral sparsifier of $\boldsymbol{L}$, we know that for $\forall \boldsymbol{x} \in \mathbb{R}^n$,

$$\frac{1}{1+\epsilon}\boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} \leq \boldsymbol{x}^\top \widetilde{\boldsymbol{L}}\boldsymbol{x} \leq \frac{1}{1-\epsilon}\boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x}.$$

Let $\boldsymbol{x} = \boldsymbol{D}^{-1/2}\boldsymbol{y}$ which is bijective, we have

$$\frac{1}{1+\epsilon}\boldsymbol{y}^\top \boldsymbol{\mathcal{L}}\boldsymbol{y} \leq \boldsymbol{y}^\top \widetilde{\boldsymbol{\mathcal{L}}}\boldsymbol{y} \leq \frac{1}{1-\epsilon}\boldsymbol{y}^\top \boldsymbol{\mathcal{L}}\boldsymbol{y}$$
$$\Longrightarrow \left|\boldsymbol{y}^\top(\widetilde{\boldsymbol{\mathcal{L}}} - \boldsymbol{\mathcal{L}})\boldsymbol{y}\right| \leq \frac{\epsilon}{1-\epsilon}\boldsymbol{y}^\top \boldsymbol{\mathcal{L}}\boldsymbol{y} < 2\epsilon \boldsymbol{y}^\top \boldsymbol{\mathcal{L}}\boldsymbol{y}.$$

The last inequality is because we assume $\epsilon < 0.5$. Then, by Courant-Fisher Theorem (Lemma 2), we can immediately get, $\forall i \in [n]$,

$$\left|\lambda_i(\widetilde{\boldsymbol{\mathcal{L}}} - \boldsymbol{\mathcal{L}})\right| \leq 2\epsilon\lambda_i(\boldsymbol{\mathcal{L}}) < 4\epsilon.$$

Then, by Lemma 1, $\sigma_i(\widetilde{\boldsymbol{\mathcal{L}}} - \boldsymbol{\mathcal{L}}) < 4\epsilon, \forall i \in [n]$. □

Given the above lemmas, we can see how the constructed $\widetilde{\boldsymbol{M}}$ approximates $\boldsymbol{M}$ and how the constructed NetMF matrix sparsifier (Eq. (6)) approximates the NetMF matrix (Eq. (3)).

Theorem 2. *The singular value of $\widetilde{\boldsymbol{M}} - \boldsymbol{M}$ satisfies $\sigma_i(\widetilde{\boldsymbol{M}} - \boldsymbol{M}) \leq \frac{4\epsilon}{\sqrt{d_i d_{\min}}}, \forall i \in [n]$.*

Proof. First notice that $\widetilde{\boldsymbol{M}} - \boldsymbol{M} = \boldsymbol{D}^{-1}\left(\widetilde{\boldsymbol{L}} - \boldsymbol{L}\right)\boldsymbol{D}^{-1} = \boldsymbol{D}^{-1/2}(\widetilde{\boldsymbol{\mathcal{L}}} - \boldsymbol{\mathcal{L}})\boldsymbol{D}^{-1/2}$. Apply Lemma 3 twice and use the result from Lemma 4, we have

$$\sigma_i\left(\widetilde{\boldsymbol{M}} - \boldsymbol{M}\right) \leq \sigma_i\left(\boldsymbol{D}^{-1/2}\right) \times \sigma_1\left(\widetilde{\boldsymbol{\mathcal{L}}} - \boldsymbol{\mathcal{L}}\right) \times \sigma_1\left(\boldsymbol{D}^{-1/2}\right)$$
$$\leq \frac{1}{\sqrt{d_i}} \times 4\epsilon \times \frac{1}{\sqrt{d_{\min}}} = \frac{4\epsilon}{\sqrt{d_i d_{\min}}}.$$

□

Theorem 3. *Let $\|\cdot\|_F$ be the matrix Frobenius norm. Then*

$$\left\|\text{trunc_log}^\circ\left(\frac{\text{vol}(G)}{b}\widetilde{\boldsymbol{M}}\right) - \text{trunc_log}^\circ\left(\frac{\text{vol}(G)}{b}\boldsymbol{M}\right)\right\|_F \leq \frac{4\epsilon\,\text{vol}(G)}{b\sqrt{d_{\min}}}\sqrt{\sum_{i=1}^{n}\frac{1}{d_i}}.$$

Proof. It is easy to observe that trunc_log$^\circ$ is 1-Lipchitz w.r.t. Frobenius norm. So we have

$$\begin{aligned}
&\left\|\text{trunc_log}^\circ\left(\frac{\text{vol}(G)}{b}\widetilde{\boldsymbol{M}}\right)-\text{trunc_log}^\circ\left(\frac{\text{vol}(G)}{b}\boldsymbol{M}\right)\right\|_F\\
\le&\left\|\frac{\text{vol}(G)}{b}\widetilde{\boldsymbol{M}}-\frac{\text{vol}(G)}{b}\boldsymbol{M}\right\|_F=\frac{\text{vol}(G)}{b}\left\|\widetilde{\boldsymbol{M}}-\boldsymbol{M}\right\|_F\\
=&\frac{\text{vol}(G)}{b}\sqrt{\sum_{i\in[n]}\sigma_i^2(\widetilde{\boldsymbol{M}}-\boldsymbol{M})}\le\frac{4\epsilon\,\text{vol}(G)}{b\sqrt{d_{\min}}}\sqrt{\sum_{i=1}^{n}\frac{1}{d_i}}.
\end{aligned}$$

□

We finally explain the remaining question in Step 1 of NetSMF: After sampling a length-r path $\boldsymbol{p}=(u_0,\cdots,u_r)$, why does the algorithm add a new edge to the sparsifier with weight $\frac{rm}{MZ(\boldsymbol{p})}$? Our proof relies on two lemmas from [8].

Lemma 5. *(Lemma 3.3 in [8]) Given the path length r, the probability for the PathSampling algorithm to sample a path $\boldsymbol{p}$ is $\tau(\boldsymbol{p})=\frac{w(\boldsymbol{p})Z(\boldsymbol{p})}{2rm}$, where $Z(\boldsymbol{p})$ is defined in Eq. (7) and*

$$w(\boldsymbol{p})=\frac{\prod_{i=1}^{r}\boldsymbol{A}_{u_{i-1},u_i}}{\prod_{i=1}^{r-1}\boldsymbol{D}_{u_i}}.$$

Lemma 6. *(Theorem 2.2 in [8]) After sampling a length-r path $\boldsymbol{p}=(u_0,u_1,\cdots,u_r)$, the weight corresponding to the new edge (u_0,u_r) added to the sparsifier should be $\frac{w(\boldsymbol{p})}{\tau(\boldsymbol{p})M}$.*

Theorem 4. *After sampling a length-r path $\boldsymbol{p}=(u_0,u_1,\cdots,u_r)$ using the PathSampling algorithm (Alg. 2). The weight of the new edge added to the sparsifier $\widetilde{\boldsymbol{L}}$ is $\frac{2rm}{MZ(\boldsymbol{p})}$.*

Proof. The proof is to plug the definition of $Z(\boldsymbol{p})$, $w(\boldsymbol{p})$, and $\tau(\boldsymbol{p})$ from Lemma 5 into Lemma 6, that is,

$$\frac{w(\boldsymbol{p})}{\tau(\boldsymbol{p})M}=\frac{w(\boldsymbol{p})}{\frac{w(\boldsymbol{p})Z(\boldsymbol{p})}{2rm}\times M}=\frac{2rm}{MZ(\boldsymbol{p})}.$$

For unweighted networks, this weight can be simplified to $\frac{m}{M}$, since $Z(\boldsymbol{p})=2r$ for unweighted networks. □

REFERENCES

[1] Nitin Agarwal, Huan Liu, Sudheendra Murthy, Arunabha Sen, and Xufei Wang. 2009. A Social Identity Approach to Identify Familiar Strangers in a Social Network.. In *ICWSM '09*.

[2] Lars Backstrom, Paolo Boldi, Marco Rosa, Johan Ugander, and Sebastiano Vigna. 2012. Four degrees of separation. In *WebSci '12*. ACM, 33–42.

[3] Daniele Calandriello, Ioannis Koutis, Alessandro Lazaric, and Michal Valko. 2018. Improved large-scale graph learning through ridge spectral sparsification. In *ICML '18*. 687âĂŞ696.

[4] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning graph representations with global structural information. In *CIKM '15*. ACM, 891–900.

[5] Raymond B Cattell. 1966. The scree test for the number of factors. *Multivariate behavioral research* 1, 2 (1966), 245–276.

[6] Kamalika Chaudhuri, Fan Chung, and Alexander Tsiatas. 2012. Spectral clustering of graphs with general degrees in the extended planted partition model. In *COLT '12*. 35–1.

[7] Dehua Cheng, Yu Cheng, Yan Liu, Richard Peng, and Shang-Hua Teng. 2015. Efficient sampling for Gaussian graphical models via spectral sparsification. In *COLT '15*. 364–390.

[8] Dehua Cheng, Yu Cheng, Yan Liu, Richard Peng, and Shang-Hua Teng. 2015. Spectral sparsification of random-walk matrix polynomials. *arXiv preprint arXiv:1502.03496* (2015).

[9] Michael B Cohen, Jonathan Kelner, John Peebles, Richard Peng, Aaron Sidford, and Adrian Vladu. 2016. Faster algorithms for computing the stationary distribution, simulating random walks, and more. In *FOCS '16*. IEEE, 583–592.

[10] Leonardo Dagum and Ramesh Menon. 1998. OpenMP: an industry standard API for shared-memory programming. *IEEE computational science and engineering* 5, 1 (1998), 46–55.

[11] Anirban Dasgupta, John E Hopcroft, and Frank McSherry. 2004. Spectral analysis of random graphs with skewed degree distributions. In *FOCS '04*. 602–610.

[12] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable Representation Learning for Heterogeneous Networks. In *KDD '17*.

[13] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *JMLR '08* 9, Aug (2008), 1871–1874.

[14] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD '16*. ACM, 855–864.

[15] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review* 53, 2 (2011), 217–288.

[16] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation Learning on Graphs: Methods and Applications. *IEEE Data(base) Engineering Bulletin* 40 (2017), 52–74.

[17] Nicholas J Higham and Lijing Lin. 2011. On p th roots of stochastic matrices. *Linear Algebra Appl.* 435, 3 (2011), 448–463.

[18] Roger A. Horn and Charles R. Johnson. 1991. *Topics in Matrix Analysis*. Cambridge University Press. https://doi.org/10.1017/CBO9780511840371

[19] Shihao Ji, Nadathur Satish, Sheng Li, and Pradeep Dubey. 2016. Parallelizing word2vec in shared and distributed memory. *arXiv preprint arXiv:1604.04661* (2016).

[20] Michael Kapralov, Yin Tat Lee, CN Musco, CP Musco, and Aaron Sidford. 2017. Single pass spectral sparsification in dynamic streams. *SIAM J. Comput.* 46, 1 (2017), 456–477.

[21] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR '17*.

[22] Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding as Implicit Matrix Factorization. In *NIPS '14*. 2177–2185.

[23] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. 2014. Scaling Distributed Machine Learning with the Parameter Server.. In *OSDI '14*, Vol. 14. 583–598.

[24] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR Workshop '13*.

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS' 13*. 3111–3119.

[26] Erik Ordentlich, Lee Yang, Andy Feng, Peter Cnudde, Mihajlo Grbovic, Nemanja Djuric, Vladan Radosavljevic, and Gavin Owens. 2016. Network-efficient distributed word2vec training system for large vocabularies. In *CIKM '16*. ACM, 1139–1148.

[27] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD '14*. ACM, 701–710.

[28] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec. In *WSDM '18*. ACM, 459–467.

[29] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-june Paul Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *WWW '15*. ACM, 243–246.

[30] Daniel A Spielman and Nikhil Srivastava. 2011. Graph sparsification by effective resistances. *SIAM J. Comput.* 40, 6 (2011), 1913–1926.

[31] Chris Stark, Bobby-Joe Breitkreutz, Andrew Chatr-Aryamontri, Lorrie Boucher, Rose Oughtred, Michael S Livstone, Julie Nixon, Kimberly Van Auken, Xiaodong Wang, Xiaoqi Shi, et al. 2010. The BioGRID interaction database: 2011 update. *Nucleic acids research* 39, suppl_1 (2010), D698–D704.

[32] Stergios Stergiou, Zygimantas Straznickas, Rolina Wu, and Kostas Tsioutsiouliklis. 2017. Distributed Negative Sampling for Word Embeddings.. In *AAAI '17*. 2569–2575.

[33] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW '15*. 1067–1077.

[34] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *KDD '08*. 990–998.

[35] Lei Tang and Huan Liu. 2009. Relational learning via latent social dimensions. In *KDD '09*. ACM, 817–826.

[36] Lei Tang and Huan Liu. 2009. Scalable learning of collective behavior based on sparse social dimensions. In *CIKM '09*. ACM, 1107–1116.

[37] Lei Tang, Suju Rajan, and Vijay K Narayanan. 2009. Large scale multi-label classification via metalabeler. In *WWW '09*. ACM, 211–220.

[38] Shang-Hua Teng et al. 2016. Scalable algorithms for data and network analysis. *Foundations and Trends® in Theoretical Computer Science* 12, 1–2 (2016), 1–274.

[39] Lloyd N Trefethen and David Bau III. 1997. *Numerical linear algebra*. Vol. 50. Siam.

[40] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2018. VERSE: Versatile Graph Embeddings from Similarity Measures. In *WWW '18*. 539–548.

[41] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2009. Mining multi-label data. In *Data mining and knowledge discovery handbook*. Springer, 667–685.
[42] Ulrike Von Luxburg. 2007. A tutorial on spectral clustering. *Statistics and computing* 17, 4 (2007), 395–416.
[43] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale Commodity Embedding for E-commerce Recommendation in Alibaba. In *KDD '18*. ACM.
[44] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. *KDD '18*.
[45] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: Cluster computing with working sets. *HotCloud '10* 10, 10-10 (2010), 95.
[46] Peixiang Zhao. 2015. gSparsify: Graph Motif Based Sparsification for Graph Clustering. In *CIKM '15*. ACM, 373–382.

Crowd-Mapping Urban Objects from Street-Level Imagery

Sihang Qiu
Delft University of Technology
Delft, Netherlands
s.qiu-1@tudelft.nl

Achilleas Psyllidis
Delft University of Technology
Delft, Netherlands
a.psyllidis@tudelft.nl

Alessandro Bozzon
Delft University of Technology
Delft, Netherlands
a.bozzon@tudelft.nl

Geert-Jan Houben
Delft University of Technology
Delft, Netherlands
g.j.p.m.houben@tudelft.nl

ABSTRACT

Knowledge about the organization of the main physical elements (e.g. streets) and objects (e.g. trees) that structure cities is important in the maintenance of city infrastructure and the planning of future urban interventions. In this paper, a novel approach to crowd-mapping urban objects is proposed. Our method capitalizes on strategies for generating crowdsourced object annotations from street-level imagery, in combination with object density and geo-location estimation techniques to enable the enumeration and geo-tagging of urban objects. To address both the coverage and precision of the mapped objects within budget constraints, we design a scheduling strategy for micro-task prioritization, aggregation, and assignment to crowd workers. We experimentally demonstrate the feasibility of our approach through a use case pertaining to the mapping of street trees in New York City and Amsterdam. We show that anonymous crowds can achieve high recall (up to 80%) and precision (up to 68%), with geo-location precision of approximately 3m. We also show that similar performance could be achieved at city scale, possibly with stringent budget constraints.

CCS CONCEPTS

• **Information systems → Geographic information systems**; **Crowdsourcing**; **Web interfaces**; • **Human-centered computing → Collaborative and social computing systems and tools**.

KEYWORDS

Crowd-Mapping, Street-Level Imagery, Task Scheduling, Crowdsourcing, Urban Objects

ACM Reference Format:
Sihang Qiu, Achilleas Psyllidis, Alessandro Bozzon, and Geert-Jan Houben. 2019. Crowd-Mapping Urban Objects from Street-Level Imagery. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313651

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313651

1 INTRODUCTION

Streets, squares, plots, and buildings constitute the most important elements that shape the physical form of cities [27]. Integral to these elements are city objects such as trees, lamp posts, and other artifacts of urban infrastructure (e.g. benches, fire hydrants etc.). The combination and configuration of physical elements and their constituent objects contribute to the "character" of a city [17] and lead to different formations of the urban fabric [6, 15]. An up-to-date knowledge of the amount and distribution of physical elements and objects across the urban fabric is of vital importance in planning future urban interventions.

At present, the process of constructing such body of knowledge is based on two main approaches. The first, and more traditional one, relies on commissioned authoritative work, by means of on-site documentation conducted by municipal workers [2, 4]. This method usually entails high costs, in addition to being laborious and time consuming. The second approach involves the voluntary contribution of geographic information, which is enabled by collaborative mapping platforms, e.g. OpenStreetMap [5, 12]. On one hand, its participatory and voluntary nature makes it a less costly approach to mapping the urban environment. On the other, it makes it prone to limited participation and, subsequently, to variable spatial coverage and precision [24]. Moreover, it assumes a degree of familiarity of the contributors with the places mapped [5]. Although it has been shown that actual knowledge of a place contributes to the amount and quality of the provided content [13, 24], it could also be seen as a limiting factor, in terms of the number of individuals who can actively create and assemble data about a place.

One way to make contributors familiar with an area of interest, without requiring knowledge at the local level, is by means of street-level imagery. The recent availability of this type of image data through online platforms, such as Google Street View and Mapillary, could enable the documentation of physical elements of the urban environment from a wider pool of potential contributors. Street-level imagery provides panoramic digital representations of – mostly – open public spaces along the street network. Contrary to the widely-used satellite imagery, it simulates the process of a person walking along the streets, thereby providing a detailed three-dimensional overview of visible spatial elements (e.g. building frontages, trees, lamp posts, trash bins, and other objects of the built environment). Given its high penetration rate, currently covering all major cities in the world, it could open up new avenues in perceiving, documenting, and understanding urban systems.

The coupling of street-level imagery with state-of-the-art computer vision and machine learning techniques enables the automatic discovery of highly recognizable recurring objects (e.g. traffic lights, street lamps) [18, 35]. However, with object types characterized by high visual variability (e.g. trees), automatic approaches suffer from severe identification and geo-location accuracy issues.

Recent studies [8–11, 31] have shown that the combination of street-level imagery with micro-task crowdsourcing can successfully engage a larger number of contributors – possibly unfamiliar with the targeted urban area – through object annotation tasks. While demonstrating the feasibility of a micro-task crowdsourcing approach, these works do not fully address the important quality and scalability issues that generally characterize large scale object annotation campaigns. Two issues are of importance: 1) the precision of object detection and geo-location within an area of interest, and 2) the ability to cover large areas within budget constraints.

Original Contributions. In this paper, we introduce a novel approach to crowd-mapping physical objects of the urban environment. Our method enables the annotation and geo-tagging of urban objects along the street network, through the combination of (a) cost- and worker quality-aware strategies for micro-task scheduling and assignment, and (b) inference methods for object density and geo-location estimation. We develop a dedicated Web user interface that facilitates urban object annotation by crowd workers, using Google Street View imagery as a proxy for real-world places. Annotation, geo-location estimation, and aggregation techniques allow us to map the objects in space, and to estimate their location. A specifically designed scheduling strategy estimates the density of objects in street segments in an iterative fashion, and uses it to prioritize mapping tasks with the aim to maximize the coverage of the mapped objects within budget constraints. Task assignment strategies allocate workers in accordance to their measured annotation quality, so as to improve the geo-location estimation quality.

To evaluate our method, we conducted two experiments. Specifically, we use areas of New York City and Amsterdam as case studies, and focus on the estimation of the number and location of trees along their street networks. Results show that through the proposed Crowd Mapping interface and geo-location estimation techniques, it is possible to engage anonymous crowd workers and achieve high recall (up to 80%) and precision (up to 68%), with geo-location precision of approximately 3m. Using a discrete event simulation technique, we show that it is possible to obtain high object density estimation precision (up to 80%) and promising geo-location precision (up to 85%) at city scale, also with limited budget availability.

The remainder of the paper is organized as follows: Section 2 discusses related work. In Section 3, we present the logical organization of the proposed crowd-mapping method. Section 4 introduces the Web interface for urban object annotation, and describes in detail the methods used to geo-locate and aggregate the labeled annotations. Section 5 describes the developed scheduling, task aggregation, and task assignment strategies. In Section 6, we present the configuration and metrics of the conducted experiments regarding the evaluation of our proposed method. In Section 7, we present and discuss the results of the experiments. Finally, Section 8 summarizes the conclusions and discusses future lines of research.

2 RELATED WORK

We address previous work on crowdsourcing with street-level imagery, and then discuss the relevant work about using street-level imagery to mapping urban objects. We acknowledge the existence of abundant literature related to task assignment and scheduling strategies (e.g. [14, 34]) for crowdsensing. However, as this paper specifically focuses on crowd-mapping with street-level imagery, we consider the analysis of such literature outside of our scope.

2.1 Crowdsourcing using Street-level Imagery

Previous work has addressed the problem of integrating crowdsourcing and street-level imagery, to enable task execution by workers through Web-based platforms. In an early feasibility study [9] manually created a database consisting of 100 images from Google Street View. Researches from the same group afterwards studied the combination of crowdsourcing and Google Street View to identify street-level accessibility problems [10] and developed an online curb ramp data collection system *Tohme* [11]. The system enables workers to remotely label curb ramps through Google Street View with the assistance of computer vision and machine learning techniques. The combination of crowdsourcing techniques and street-level imagery has further applications in risk assessment studies and, particularly, the vulnerability of structures against earthquakes [28]. Relevant techniques have also been applied in medical- and health-related research [3, 33]. Street-level imagery, informed by data from parents of school children, has been used as an efficient alternative to on-site observations for the assessment of the environmental characteristics of cycling-to-school routes [33].

These studies show the feasibility of a crowdsourcing approach to urban scenes and object labeling through street-level imagery. However, issues of accurate object detection and geo-tagging, scalability, and cost received limited attention. To the best of our knowledge, our work is the first to fill this gap.

2.2 Mapping Urban Objects using Street-level Imagery

Street-level imagery is increasingly used in studies concerned with the mapping of urban objects. *Mapillary* [1] is a dataset consisting of 400M street-level images covering 6M kilometers worldwide, in which images are processed with computer vision, and identified objects are connected to open-source and commercial mapping platforms (such as OpenStreetMap, and Here) [26]. Street-level imagery has already been an important data source for mapping urban trees. *Treepedia* is a project estimating the vegetation index of cities. Researchers proposed a method that can automatically detect urban greenery with computer vision by assessing street-level imagery from Google Street View [20, 21]. The urban canopy cover was mapped in Singapore using the hemispherical photographs from Google Street View [29]. Similar works are also conducted in Chinese cities with alternative street-level imagery sources, such as Tencent Street View [23]. It is also used in the detection and mapping of other types of physical objects. Balali et al. used images from Google Street View to detect, classify, and map traffic signs of two interstate highways in the United States [1].

[1] https://www.mapillary.com/

Street-level imagery has also been used in the classification of POI types [25]. Previous work used deep learning techniques to geo-locate and match a street-level image to an aerial one [22].

Work on the automatic analysis of street-level imagery still suffers from geo-localization issues. Moreover, the lack of training datasets results in limited applicability to object types that present high visual variability (e.g. trees). Our work is complementary to these effort, and can simplify the collection of street-level imagery training data for computer vision.

3 METHOD

The Crowd Mapping method proposed in this paper is organized in three sequential steps, elaborated in Algorithm 1: (1) initialisation, (2) Crowd Mapping, and (3) geo-location prediction. The algorithm takes as *input* a set of n streets $ST = \{st_1, st_2, \cdots, st_n\}$ pertaining to an urban area of interest. Streets are partitioned into segments, where $S(st) = \{s_1, s_2, \cdots, s_l\}$ is the set of segments of a given street $st \in ST$, and $U_S = \{S_1, S_2, \cdots, S_n\}$ is the set of all street segments. The set $E = \{e_1, e_2, \cdots, e_m\}$ defines all the physical objects of a given type that are present in the targeted area; in the scope of our work, each object $e = (lat, lng)$ is described by the World Geodetic System (WGS) coordinates of its lat and lng. We define x_s the *density*[2] of urban objects contained in a street segment $s \in U_S$; $X(st) = \{x_1, x_2, \cdots, x_l\}$ is the corresponding set of segment densities in a street st; and $U_X = \{X_1, X_2, \cdots, X_n\}$ is the universe of all sets densities.

Algorithm 1: The proposed Crowd Mapping method

Input: all the streets ST, workers W, and *budget*.
Output: estimated physical elements of a given type $\hat{E}$

```
// Step 1: Initialization
U_S ← Segmentation(ST) ;
T ← InitTaskGeneration(U_S) ;
// Step 2: Crowd Mapping with Street Level Imagery
while budget > TaskMaximumCost() ∨ T = ∅ do
    w ← GetNewWorker(W);
    t ← TaskAssignment(T, w);
    A ← A ∪ AnnotationUI(w, t);
    budget ← budget − TaskCost() ;
    T ← T − t ;
    Ê, U_X̂ ← Aggregation(A);
    H_PS ← SegmentScheduling(U_S, U_X̂);
    T ← T ∪ TaskGeneration(H_PS) ;
end
// Step 3: Geolocation Prediction
Ê ← Ê ∪ GeoLocationPrediction(Ê, H_PS, ST);
```

Given ST, a set $W = \{w_1, w_2, \cdots, w_i\}$ of available crowd workers, and a *budget*, the goal of our method is to calculate: (1) the set $\hat{E} = \{\hat{e}_1, \hat{e}_2, \cdots \hat{e}_m\}$ of estimated physical objects; and (2) the set $U_{\hat{X}} = \{\hat{X}_1, \hat{X}_2, \cdots, \hat{X}_n\}$ of estimated segments density. Indeed, $|\hat{E} \cup E| \neq |E|$, as the crowd mapping activity could miss existing objects, or produce non existing ones, due to crowd mapping issues. Conversely, all $\hat{X}_n$ are estimated.

The first step, `Initialisation`, segments each street within ST according to a defined *segmentation* policy (e.g. streets are split in segments of at least 50 meters), to create the set $U_S = \{S_1, S_2, \cdots, S_n\}$ of street segments. Here each $s \in S_n$ is a unit of work. The initial set of crowd tasks $T = \{t_1, t_2, \cdots, t_j\}$ is then created, where each task t contains one or more segments from U_S that are selected and aggregated according to an *initial task generation* strategy (e.g. include one random segment from each street).

The `Crowd Mapping` step, executed until either *budget* is available or $T = \emptyset$, involves the creation of the object *annotations* set $A = \{a_1, a_2, \cdots, a_j\}$ through a Web interface. An annotation $a \in A$ is a *bounding box* drawn on a street-level image by a worker w. Annotations are aggregated by means of an *aggregation function* that progressively populates the set $\hat{E}$, and the set $U_{\hat{X}}$ for the considered segments. The details of the annotation user interface and aggregation strategies are described in Section 4. As segments get annotated, new tasks are created by a *Segment Scheduling* strategy, which establishes the priority of all pending segments into a priority queue H_{PS} that is then used to generate new tasks. A *Task Assignment* strategy distribute tasks to available workers. Section 5 describes the segment scheduling and Task Assignment strategies.

The final `Geo-location Prediction` step enriches the set $\hat{E}$ by predicting the location of urban objects in segments that are not annotated by crowd workers. This could be done by using the density information acquired from the second step, including measured or estimated segment densities, and the location of objects in nearby segments. The adopted geo-location prediction method heavily depends on the targeted urban object, as the regularity of spatial distribution varies significantly among different objects (e.g. street trees and lamp posts are usually more regularly distributed than trash bins). We describe a prediction method in Section 6, where we elaborate on the use case addressed in the experiments.

4 CROWD MAPPING OF URBAN OBJECTS WITH STREET-LEVEL IMAGERY

This section describes the Crowd Mapping step (i.e. Step 2). Section 4.1 describes the Web user interface supporting the *annotation* of urban objects using Street-Level Imagery. Section 4.2 and Section 4.3 respectively elaborate on the methods used to geo-locate and aggregate annotations for urban object mapping.

4.1 User Interface

Figure 1 presents the Web user interface for object annotation. The top-left panel renders the Street-Level Imagery (SLI) exploration widget; the worker can virtually visit an urban area of interest by navigating between *bubbles* [16], i.e. 360° panoramas that provide a photorealistic impression of an area as seen from a given viewpoint. The user can pan and zoom inside the bubble, and move towards contiguous bubbles. The **ANNOTATE** button activates the object annotation interface, described below. The bottom-left panel contains a map of the area under analysis, with the segments pertaining to the current annotation tasks masked. Visual clues provide the workers with an indication of the explored bubbles

[2]With a slight abuse of terminology, we refer to *density* as to the normalized number of objects in a given street segment.



Figure 1: Urban Object Annotation Web interface. In the example, workers are asked to annotate *street trees*.



Figure 2: Object Annotation Procedure.

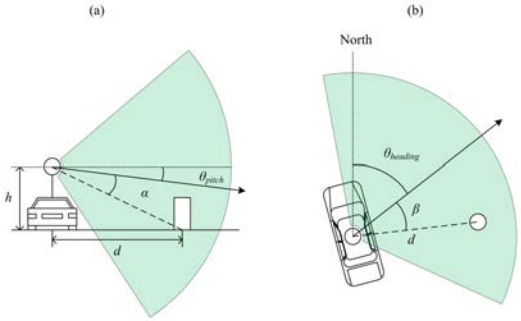


Figure 3: Estimation of the distance between the observation viewpoint and the object's center (a) and geo-location (b).

(red *footprint* dots), and of the location of the provided annotations (green *annotation* dots). The bottom panel contains a task description, a task progress bar, and commands to teleport to the previous or next segments to analyze. The right panel lists all the objects annotated in the current task, and allows the workers to teleport back in the same bubble where the annotation has been made. Finally, the SUBMIT button finalizes the task. The interface supports different SLI and map services, including Google Street View.

The annotation interface is central to the mapping task, as it allows for the estimation of an object's geo-location. We assume an object's coordinates to indicate the centroid of its bottom surface. As we show later, this is a precondition for geo-location estimation. Workers are asked to draw the bounding box as described in Figure 2, according to the following steps: (1) Position the mouse pointer at the center point of the object's bottom surface. If this point is not obvious or occluded by other objects, workers should estimate the position of the point. (2) Click and drag the mouse upwards to draw a symmetrical box that bounds the object.

4.2 Linking Individual Crowd Annotations to Geo-Locations

Street-Level Imagery is typically acquired by vehicles (e.g. cars and bicycles) equipped with 360° image acquisition devices. The geo-location of a bubble (i.e. the virtual observation viewpoint) corresponds to the geo-location of the vehicle at the image acquisition time, as acquired by a GPS device. As shown in Figure 3(a), the distance d between the bubble geo-location and the geo-location of the annotated object can be calculated using $h/\tan(90° - \theta_{pitch} - \alpha)$, where h is the height of the camera installed on the acquisition vehicle. The pitch angle θ_{pitch} (i.e. the up or down angle of the camera) is one of the essential parameters of a street-level panorama image. The angle α and β (in Figure 3(b)) are the angles between the central ray of camera and the ray casting to the bottom of the object in vertical and horizontal plane respectively, which could be estimated from the relative position of the bounding box on the Street View panel.

Figure 3(b) shows how the geo-location of an object could be estimated, using the calculated distance, and the $\theta_{heading}$ could be acquired from parameters of the panorama image. Then, assuming the current latitude and longitude of the observation viewpoint are (lat_{cam}, lng_{cam}), the relative Cartesian coordinate (rx_a, ry_a) (if the coordinate origin is the location of camera) of an annotation a can be calculated as:

$$\begin{cases} rx_a = C_a^x h, \\ ry_a = C_a^y h, \end{cases} \quad \begin{cases} C_a^x = \frac{\cos(90^\circ - \theta_{heading} - \beta)}{tan(90^\circ - \theta_{pitch} - \alpha)}, \\ C_a^y = \frac{\sin(90^\circ - \theta_{heading} - \beta)}{tan(90^\circ - \theta_{pitch} - \alpha)}. \end{cases} \tag{1}$$

where C_a^x and C_a^y are coefficients of the parameters (such as heading, pitch, etc.) of the annotation a, and the arc length of each degree of latitude is approximately $L = 111300$ meters. The WGS coordinate of the annotation a can then be calculated as:

$$\begin{cases} lat_a = lat_{cam} + ry_a / L, \\ lng_a = lng_{cam} + rx_a / [L \cos(lat_{cam})]. \end{cases} \tag{2}$$

Camera Height Correction. The height h of the camera is the only parameter that could not be measured nor estimated from the user interface. In the case of Google Street View, online documentation suggests that the height of the camera is around 8.2 feet[3]. We however account for different acquisition configurations, and for differences in altitude between the street and the object's bottom surface. The camera height might also be inconsistent due to different vehicles and different acquisition time. Therefore, we estimate an optimal camera height to minimize the measurement error. High accuracy annotations can be acquired for the set of objects $SE \in E$, and the relative Cartesian coordinate (rx_e, ry_e) of the exact location of the urban object $e \in E$ can be expressed as:

$$\begin{cases} rx_e = L(lng_e - lng_{cam}) \cos(lat_{cam}), \\ ry_e = L(lat_e - lat_{cam}). \end{cases} \tag{3}$$

Having the Cartesian coordinates of all the real objects in the sample set SE and the corresponding high-quality annotations, the camera height correction minimizes the distances between these pairs (e, a_e). Specifically, the following cost function indicating the mean square error should be minimized:

$$J(h) = \sum_{e \in SE} \{[rx_e - rx_{a_e}(h)]^2 + [ry_e - ry_{a_e}(h)]^2\}. \tag{4}$$

After differentiation, the corrected camera height is:

$$\hat{h} = \frac{\sum_{e \in T}(rx_e C_{a_e}^x + ry_e C_{a_e}^y)}{\sum_{e \in T}({C_{a_e}^x}^2 + {C_{a_e}^y}^2)}. \tag{5}$$

4.3 Aggregation of Crowd Annotations for the Location Estimation of Urban Objects

To improve the quality of the estimation, the same object is annotated multiple times, possibly by different crowd workers. While each single annotation might be imperfect – as well as its geo-location estimation – it is reasonable to assume that different labels for the same object will be distributed around its centroid, and that the location of the object should correspond to the highest density of annotations. Traditional density-based clustering algorithms such as DBSCAN are sensitive to a density threshold parameter. We use the clustering algorithm proposed in [30] and extend it to account for the quality $quality_w$ of the worker that performed the aggregation. In Algorithm 2, the density ρ of objects is first calculated (Lines **1–3**). The function $dis(a, a')$ calculates the distance between the estimated geo-locations of a and a', and it is used to estimate the minimum distance δ_a between the annotation a and any other annotation with higher density (Lines **4–6**). Finally, the location of all objects $\hat{E}$ is calculated by selecting annotations whose δ_a is larger than the minimum threshold δ_{min} (Lines **8–14**).

Algorithm 2: Geo-location aggregation algorithm

Input: all annotations A, the minimum distance threshold between trees δ_{min}, radius $r = \delta_{min}/2$ for calculating density

Output: estimated geo-locations $\hat{E}$

```
1  for each annotation a ∈ A do
2  |  ρ_a ← Σ_{a'∈A:dis(a,a')<r} quality_{w_a'} ;
3  end
4  for each annotation a ∈ A do
5  |  δ_a ← min_{a'∈A:ρ_a>ρ_a'} {dis(a, a')} ;
6  end
7  Ê = ∅;
8  for each annotation a ∈ A do
9  |  if δ_a > δ_min then
10 |  |  lat ← Σ_{a'∈A:dis(a,a')<r} quality_{w_a'} lat_a'/ρ_a' ;
11 |  |  lng ← Σ_{a'∈A:dis(a,a')<r} quality_{w_a'} lng_a'/ρ_a' ;
12 |  |  Ê ← Ê ∪ (lat, lng);
13 |  end
14 end
```

5 SEGMENT SCHEDULING AND TASK GENERATION

To maximize the coverage and quality of estimation within budget constraints, street segments must be scheduled, aggregated in tasks, and assigned to crowd workers.

5.1 Segment Scheduling

The *Segment Scheduling* (Algorithm 1, **Line 10**) operation is responsible for the prioritization of segments to be allocated to crowd masking tasks. Scheduling is logically organized around a priority queue (heap) data structure, which orders segments to be analyzed according to their priority. Segments with the highest priority are at the head of the heap and are, therefore, selected first. Prioritization is generally related to the likelihood of the successful prediction of object densities $U_{\hat{X}}$, while simultaneously optimizing budget consumption. Such likelihood is assumed to be unknown in advance and must, therefore, be calculated dynamically from the annotations produced by crowd workers on processed segments.

We define $AS = \{as_1, as_2, \cdots, as_{l_a}\}, AS \in S$ as the set of segments that have been already annotated by crowd workers; $PS = \{ps_1, ps_2, \cdots, ps_{l_p}\}, PS \in S$ is the set of segments yet to be annotated. Clearly, $AS \cap PS = \emptyset$, $AS \cup PS = S$. Considering spatial

[3] A Glimpse of Google's Fleet of Camera-Equipped Street View Cars. https://petapixel.com/2012/10/15/a-glimpse-of-googles-fleet-of-camera-equipped-street-view-cars/

dependence, deriving from Tobler's first law of geography [32], we assume that it is more likely for adjacent street segments to feature a similar amount and configuration of urban objects. It is therefore possible, with some degree of uncertainty, to walk a street in a given direction and build a density model for that street. In other words, we can assume that the probability of the object density of a street segment depends on the object density of its previous segment. We can easily calculate the density of the segment $as \in AS$ using the estimated objects $\hat{E}$. The question is how to predict the missing density of the segment $ps \in PS$.

Our solution, called **Maximum Likelihood** (ML) of density prediction is inspired by the *Greedy-based Minimization Entropy* algorithm [14]. We model the object density of street segments as a Markov model. $\hat{X} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_l\}$ is the set of object densities of all segments in S. The probability of object density $\hat{x}$ of a segment s is a function of the object density $\hat{x}'$ of the previous segment s', which is denoted as $p(\hat{x}', \hat{x})$. When the densities $\hat{x}_\sigma$ and $\hat{x}_\tau$ of two segments (s_σ and s_τ) from the same street are calculated, it is therefore possible to estimate the density of the other segments located between them. Thus, the problem is to find the most likely density sequence $\hat{X}_{\sigma\to\tau} = \{\hat{x}_{\sigma+1}, \hat{x}_{\sigma+2}, \cdots, \hat{x}_{\tau-1}\}$. The estimated density is $\hat{x} = k/length, \hat{x} \in \hat{X}_{\sigma\to\tau}$, where $k \in K = \{0, 1, 2, \cdots, k_{max}\}$ is the possible amount of objects and $length$ is the length of the street segment $s \in S$. k_{max} is the maximum possible amount of object, which is used for limiting the estimation range.

The optimal density sequence is efficiently calculated with dynamic programming. Let $D_{s,k}$ record the maximum likelihood of the sequence (the product of all the transition probabilities) from the crowd-annotated density $\hat{x}_\sigma = k_\sigma/length$ to a possible density $\hat{x}_s = k/length, \sigma < s < \tau$. The final goal is to maximize D_{τ,k_t} and estimate the most likely sequence $\hat{X}_{\sigma\to\tau} = \{\hat{x}_{\sigma+1}, \hat{x}_{\sigma+2}, \cdots, \hat{x}_{\tau-1}\}$. $D_{s,k}$ can be calculated using the following equations (here s' means the previous segment of s):

$$\begin{cases} D_{\sigma,k_\sigma} = 1, \\ D_{s,k} = \max_{k' \in K}\{p(k',k) D_{s',k'}\} \end{cases} \tag{6}$$

Algorithm 3 describes how the most likely sequence is finally calculated. During the process of predicting the density of pending segments, the likelihood π of optimal prediction of the street can be calculated by the following equation:

$$\pi = \prod_{s \in AS : s' \in PS} D_{s', \hat{x}'_s length} \tag{7}$$

For each street, the scheduling strategy calculates the priority of segments by first calculating the product of likelihood of the optimal prediction of all streets, which is denoted as $\prod_\pi$. Then, we use the predicted density $\hat{x}$ of the pending segment as "true" analyzed data, and take it together with the density values of analyzed segments AS to re-calculate the product of likelihood of the optimal prediction of all streets, which is denoted as $\prod'_\pi$. The final segment priority is then determined as $priority = \prod'_\pi - \prod_\pi$.

5.2 Task Generation

Task generation is performed 1) at task initialization time (Algorithm 1, **Line 2**), when no prior information about the distribution of urban objects is available; and 2) at run time (Algorithm 1, **Line 11**), after segment scheduling. We employ a *street-based initialization strategy* where, for each street, the first and the last segments of streets are first selected. At run-time, tasks are selected by drawing street segments from the heap, as shown in Figure 4.

Algorithm 3: Dynamic programming. Find the most likely state sequence

Input: Initial density $x_\sigma = k_\sigma/length$ and $x_\tau = k_\tau/length$, the set of possible number of objects $K = \{0, 1, 2, \cdots, k_{max}\}$, transition probability function p

Output: the most likely density sequence $\hat{X}_{\sigma\to\tau} = \{\hat{x}_{\sigma+1}, \hat{x}_{\sigma+2}, \cdots, \hat{x}_{\tau-1}\}$

```
1  D_{σ,k_σ} ← 1 ;
2  for each s ∈ {σ + 1, σ + 2, ··· , τ} do
3      for each k ∈ K do
           // s' means the previous segment of s.
4          D_{s,k} ← max_{k'∈K}{p(k', k) D_{s',k'}} ;
           // Variable pre records the sequence.
5          pre_{s,k} ← arg max_{k'∈K}{p(k', k) D_{s',k'}} ;
6      end
7  end
   // Get the most like sequence using pre.
8  k = pre_{τ,k_τ} ;
9  for each s ∈ {τ − 1, τ − 2, ··· , σ + 1} do
10     x̂_s ← k/length ;
11     k ← pre_{s,k} ;
12 end
```

As tasks are typically published in batches having the same monetary compensation for workers, it is important to balance the total amount of time needed to complete a task. Segments might have variations in length; their *exploration* might, therefore, require different amounts of time. Similarly, segments contain a varying amount of objects to be annotated. We, therefore, introduce the notion of segment *workload*, which is defined as follows:

$$workload = length + w \cdot \hat{x} \cdot length, \tag{8}$$

where w is used to balance the effort needed for *exploration* (that is proportional to the segment length), and the effort needed for *annotation*, which is proportional to both the segment length and the estimated amount of objects in it. In a set of preliminary experiments conducted with crowd workers of the FigureEight[4] platform, we found that the average time needed for object annotation (8.05 seconds) is 2.5 times longer than the time needed for exploration (3.20 seconds per meter). We therefore set $w = 2.5$. At task initialization, when no prior information is available about the density of objects, the *workload* is calculated only using segment length information.

Segments are, therefore, included in tasks in a greedy fashion, until the sum of their workload reaches a predefined minimum value that is proportional to the targeted total task execution time and compensation.

[4] https://www.figure-eight.com

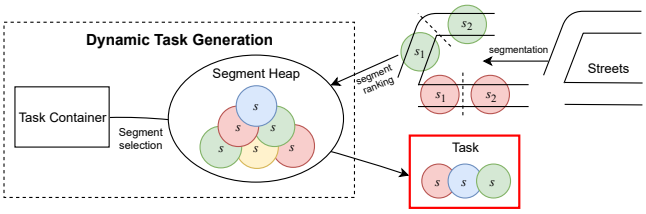


Figure 4: Heap-based Task Generation.

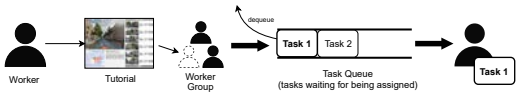


Figure 5: Single-queue Assignment Strategy.

5.3 Worker Quality-Aware Task Assignment

A task includes several street segments. To acquire annotations for the same street segments and objects, tasks are assigned to multiple workers. The quality of incoming workers is assessed through an initial evaluation task.

Worker Quality Evaluation. After reading the task instructions, the worker is offered an assessment task to perform on segments and objects with known geo-location. The quality of the worker is assessed according to three properties: the annotation recall (R), precision (P), and root mean square error ($RSME$):

$$R = \frac{\#\text{ of annotated objects}}{\#\text{ of objects in assessment}} \qquad P = \frac{\#\text{ of correct annotations}}{\#\text{ of annotations of the worker}}$$

$$RMSE = \sqrt{\frac{\sum (\text{distance from groundtruth annotations})^2}{\#\text{ of correct annotations}}}$$

We consider an annotation to be correct if its distance from the targeted objects is lower than a given *threshold*. The worker *quality score* is evaluated by synthesizing these three properties as:

$$\text{quality score} = (1 - RMSE/threshold) + recall + precision \quad (9)$$

Task Assignment Strategies. We consider two strategies.

Single-queue Assignment Strategy. This task assignment strategy is commonly used in microtask crowdsourcing. Tasks are pushed in an execution queue, and then served to workers according to their order of arrival (Figure 5). Tasks are de-queued when the number of required workers is reached.

Multi-queue Assignment Strategy. The final annotation quality can be significantly affected by the performance of their workers. Previous work [36] suggests that the overall quality can be increased by assigning workers with different quality levels, while keeping the average worker quality of each task as high as possible. Inspired by the finding in skill-and-stress-aware assignment [19], the Multi-queue Assignment Strategy divides workers into three groups (*low*, *medium* and *high*) by ranking them according to their quality score, and equally splitting the rank in three groups.

The multiple-queue strategy builds three tasks queue corresponding to three quality levels. After evaluating the quality level of the worker, the task at the head of queue of the same quality level will

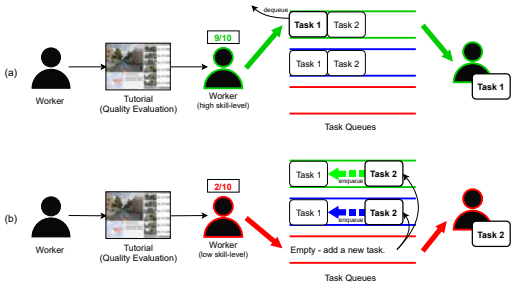


Figure 6: Multi-queue Task Assignment Strategy.

be de-queued and assigned to the worker (Figure. 6 (a)). If there is no task in the queue of the same quality level as worker's, a new task will be generated and assigned to the worker. Meanwhile, this task will be queued into the other two queues, waiting for being assigned to workers with different quality levels (Figure. 6 (b)).

6 EXPERIMENTAL SETUP

We conducted a number of experiments to investigate: (RQ 1) to what extent can the designed interface, geo-location annotation, and aggregation algorithms lead to correct and complete maps of urban objects when operated by crowd workers from micro-task crowdsourcing platform; and (RQ 2) to what extent the proposed segment scheduling and task assignment strategies can provide correct and complete maps within budget constraints. In this section, we provide details on the case study, the adopted datasets, experimental configuration, and evaluation metrics.

6.1 Case Study: Mapping Street Trees

Urban green infrastructure (e.g. street trees, parks etc.) is an integral set of physical objects and elements comprising the urban fabric [15] [6]. Trees have obvious ecological benefits in terms of air quality and heat regulation, and further contribute to defining the "character" of an urban area. It is therefore important to regularly record information about their location, properties (e.g. tree type), and status for maintenance and planning purposes. American cities such as New York[5] and San Francisco[6] have tree census projects that are run in collaboration, between the municipalities and volunteer organizations. The municipality of Amsterdam, began to record the information about trees since 1875.[7] Although of significant value for cities and communities, tree mapping is still a tedious and expensive task to perform.

6.2 Cities and Datasets

The experiments focus on two regions of two world cities [7], respectively New York City's Manhattan borough, and the area of Amsterdam that is defined by the ring of the A10 highway. We retrieve information about the street networks from OpenStreetMap[8], excluding pedestrian ways, bicycle ways, and other similar types

[5] https://tree-map.nycgovparks.org
[6] https://urbanforestmap.org
[7] www.amsterdam.nl/bomen
[8] https://www.openstreetmap.org/. We selected streets whose tag is *motorway*, *trunk*, *primary*, *secondary*, *tertiary*, *residential* or *unclassified*.

of streets for which associated street-level imagery might not be available. Ground truth data about the location of trees have been collected from the *New York City Street Tree Map* project[9] and the official trees dataset from the municipality of Amsterdam.[10]

We removed from the ground truth datasets trees that are at a distance larger than 30 meters from the street's mid-line, to heuristically exclude trees that could belong to parks or other public spaces (e.g. squares). Table 1 provides basic statistics on the number of streets and the total number of urban trees in the considered datasets. While these datasets are extensive, they are difficult to maintain and can contain errors. The Google Street View panel has been configured to display the latest available version of SLI. We acknowledge the likely presence of temporal mismatches between SLI acquisition time and the dataset update. We address the issue when manually refining the set of street segments used in RQ1 (Section 7.1). However, we ignore such inconsistencies in the experiments addressing RQ2, considering the emerging differences negligible in the context of the experimental goal.

Table 1: Dataset Statistics on the Two Areas of Interest.

# Urban elements and objects	**Manhattan**	**Amsterdam**
# Streets	4081	9084
# Trees	57499	88512
# Segments	18676	23974
Segment Length (m)	44.60 ± 6.45	39.46 ± 10.81
Segment Tree density (#/100m)	3.15 ± 7.34	5.77 ± 11.75

6.3 Configurations

Our crowd-mapping method has a number of parameters that can be configured.

Street Segmentation. We set the maximum length of each segment to 50 meters, to allow for short micro-tasks and reduce annotation fatigue. All segments belonging to the same street have equal length, which is calculated as $length_{st}/(\lceil length_{st}/50 \rceil + 1)$. The resulting number of segments is reported in Table 1.

Camera Height Correction. For each city, we manually select 10 segments with abundant vegetation, and accurately annotate the trees they contain. The locations of these trees are used to estimate the corrected camera heights, which are set to 1.979 and 1.730 meters in Manhattan and Amsterdam, respectively.

Geo-location Prediction. The last step of the crowd-mapping method must be customized for the urban object at hand. For the targeted use case, we designed a simple interpolation method that predicts the location of trees in non-annotated segments from the estimated location of objects in contiguous street segments. To achieve this, we calculate (1) the distance between the centroid of the tree (on both sides of the street) and the mid-line of the street, and (2) the distance between the centroids of adjacent trees in the annotated segments. Based on the corresponding measured distances of the last trees in the annotated segments, we perform the interpolation to estimate the number and location of trees in the non-annotated parts.

Annotation Task Configuration. The urban object annotation Web interface has been deployed on the Figure-Eight platform. For each segment, we collected three judgments. To maximize diversity in the pool of recruited workers while providing sufficient incentives for participation, we allowed workers to perform a maximum of three annotation tasks per city. As a quality control mechanism, workers were allowed to submit the task (and receive compensation) only if the whole area covered by the task segments was explored, and if the exploration of each segment lasted at least 60 seconds. The minimum *workload* for each task has been set to 500 (see Section 5.2), with an average hourly compensation greater than the current US minimal wage. In Figure Eight preliminary experiments, each task contains one segment.

For each city, 50 segments were selected (for RQ1) as follows. We retrieved the vegetation index for each segment in the street networks of New York City and Amsterdam from Google satellite imagery using the vegetation index equation from previous work [21]. We then performed importance sampling on the segments, where the frequencies of samples were sorted according to the vegetation index (importance weight) on the segments. Given that the vegetation index is measured from satellite imagery, it cannot fully reflect the amount of trees at the street level. Therefore, we excluded some street segment that were selected by the sampling method, but were not compatible with our experimental setup. These street segments are usually at the outskirts of the areas in question and contain a large amount of trees that are not included in the ground truth data set. Eight segments from Manhattan and nine segments from Amsterdam were, respectively, excluded, resulting in 42 remaining sampled segments for Manhattan (i.e. 2.2‰of the total number of street segments, 5.4‰of the total number of urban trees) and 41 for Amsterdam (i.e. 1.7‰of the total number of street segments, 3.8‰of the total number of urban trees).

6.4 Evaluation Metrics

We evaluate the performance of crowd workers, annotation aggregation, density prediction, and geo-location prediction against the ground truth in the selected datasets. We consider an annotation *correct* if its distance from the corresponding tree in the ground truth dataset is equal to or lower than 5 meters.

In terms of density estimation, we measure *Density Prediction* (*DP*) and *Density Mean Square Error* (*DE*) as follows:

$$\mathrm{DP} = \frac{|\{s|\text{Density Error} = 0,\ s \in S\}|}{|U_S|} \quad \mathrm{DE} = \hat{x}_s - \frac{|E_s|}{length_s} \tag{10}$$

where E_s is the set of physical objects (i.e. street trees in this case) of the segment s. In terms of object geo-location, we measure *Recall* (R), *Precision* (P), and *Root Mean Square Error* as follows:

$$\mathrm{R} = \frac{\#\text{ annotated objects}}{|E|} \quad \mathrm{P} = \frac{\#\text{ of correct annotations}}{\#\text{ of all aggregated annotations}}$$

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{\text{correct annotations}} (\text{annotation error})^2}{\#\text{ of correct annotations}}}$$

[9] https://data.cityofnewyork.us/Environment/2015-Street-Tree-Census-Tree-Data/pi5s-9p35. Last update 2015.
[10] https://maps.amsterdam.nl/open_geodata/. Last update 2017.

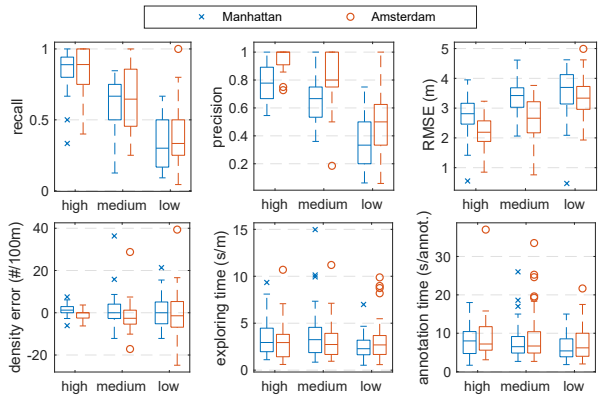


Figure 7: Performance distribution of FigureEight workers.

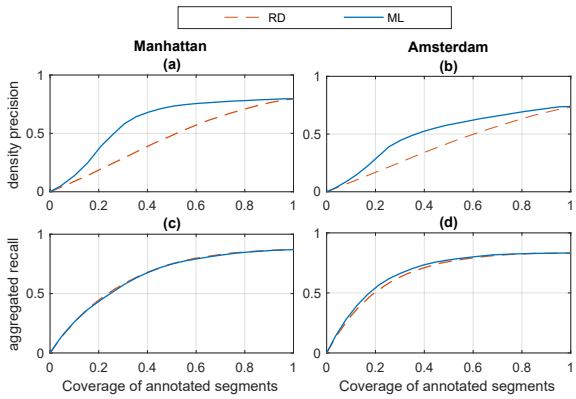


Figure 8: Density precision and Aggregated Annotations Recall for increasing segment coverage with SQ task assignment strategy.

7 RESULTS AND DISCUSSION

7.1 RQ1: Crowd Mapping with Street-Level Imagery

A total of 71 and 63 unique workers (Level 3) from Figure-Eight selected the annotation tasks for Manhattan and Amsterdam, respectively. On average, $N = 44$ workers were *Somewhat Satisfied* by the task (3.9/5); they were *neutral* with regard to task difficulty (2.9/5); they found the instructions and the interface *Somewhat Clear* (3.8/5), the quality control *Somewhat Fair* (3.9/5), and the compensation *Somewhat better* than similar tasks (3.8/5).

We manually processed the results. From the Manhattan experiment, we excluded 26 task executions where workers provided no annotations, or the annotation was of bad quality. In Amsterdam, 9 task executions were excluded for the same reasons. With the remaining judgments, we split workers into three quality classes (High, Medium, and Low) of equal size.

Table 2 and Figure 7 show the performance of crowd workers. The precision, recall, and RSME of high-quality individual workers are remarkably high. As expected, worker annotation recall and precision decrease with the worker quality level (high to low) in both Manhattan and Amsterdam, while RMSE increases. The resulting density error is good in general, with larger standard deviation at lower quality levels. Interestingly, workers in Manhattan have (on average) a positive density error, whereas the Amsterdam ones have a negative error. A manual inspection of the worker annotations shows no significant difference between workers of similar quality. We, therefore, account the issue to the mismatch between the tree datasets (the latest updated Manhattan dataset is from 2015) and Google Street View imagery. The aggregation method generally mitigates the impact of low-quality annotations, although the precision is negatively affected. Workers of lower quality tend to speed through tasks, but only the difference in exploration time between the low-quality level and the other two levels in Manhattan is of statistical significance (Wilcoxon rank-sum test, $p < 0.1$). Furthermore, the differences in recall, precision, and RMSE across quality levels are also of statistical significance. Other metrics are of no statistical significance across all quality levels ($p > 0.1$).

7.2 RQ2: Segment Scheduling and Task Assignment

To test the performance of the segment scheduling and task assignment strategies, we simulated the behavior of workers by drawing their annotation performance (in terms of precision, recall, and RSME) from the distribution of the workers that participated in the previous experiment. We fed these distributions to a Discrete Event Simulation (DEVS) system that mimics the exploration and annotation behavior of workers of different quality.

When a simulated worker annotates an object, the *Precision* similation parameter decides whether the annotation is correct or not; the error (distance) between simulated annotations and the object following normal distribution $N(0, RMSE)$, while the direction of annotation follows a uniform distribution $U(0, \pi)$. The *Recall* parameter controls the number of correctly annotated objects in a segment. For each experimental configuration, the simulation has been run five times.

Segment scheduling strategy. We first test the performance of the Maximum Likelihood (ML) scheduling strategy (Section 5.1). We employ the single-queue (SQ) task assignment strategy, and compare ML against a random segment scheduling baseline, where segments from U_S are picked randomly. Figure 8 (a) and (b) illustrates how the density precision increases, with regard to increased annotated segments – i.e. allocated *budget*: 100% of segment coverage can be achieved with infinite funds. Note that density precision measures density estimation with *zero* estimation error. ML can achieve greater density precision at lower coverage levels. The maximum performance gain of ML against RD is respectively obtained in at 35% segment coverage, with 35% increase in Manhattan and 22.23% increase in Amsterdam. Table 3 ($\mu \pm \sigma$ of five simulation executions) shows that the average density prediction error also decreases.

Figure 8 (c) and (d), and Table 3 show that ML can provide small improvements in terms of predicted tree locations. This is an expected result, as the Geo-location step implemented in this experiment does not make use of the predicted segment density.

Table 2: $\mu \pm \sigma$ (average and standard deviation) of recall, precision, RMSE (meter), density error (# of objects per 100 meters), normalized exploration time (second/meter), and normalized annotation time (second/annotation) of Figure Eight workers.

Parameters	Manhattan				Amsterdam			
	High	Medium	Low	Aggregation	High	Medium	Low	Aggregation
Recall	0.85 ± 0.15	0.61 ± 0.19	0.33 ± 0.18	0.75 ± 0.33	0.85 ± 0.17	0.64 ± 0.22	0.38 ± 0.21	0.80 ± 0.23
Precision	0.79 ± 0.14	0.66 ± 0.16	0.35 ± 0.18	0.54 ± 0.20	0.96 ± 0.07	0.81 ± 0.17	0.48 ± 0.25	0.68 ± 0.26
RMSE	2.75 ± 0.62	3.41 ± 0.60	3.35 ± 0.80	3.19 ± 0.73	2.26 ± 0.50	2.59 ± 0.74	3.36 ± 0.70	2.74 ± 0.73
Density error	1.28 ± 2.65	1.56 ± 8.32	0.20 ± 8.20	2.16 ± 4.19	−0.97 ± 2.19	−1.71 ± 7.13	−0.53 ± 10.08	−0.39 ± 5.37
Exploration Time	3.55 ± 1.99	3.88 ± 3.05	2.54 ± 1.27	/	2.97 ± 2.03	3.10 ± 2.07	3.21 ± 2.33	/
Annotation Time	8.06 ± 3.66	7.98 ± 5.00	6.48 ± 3.53	/	8.69 ± 5.71	9.25 ± 7.06	7.82 ± 4.70	/

Table 3: $\mu \pm \sigma$ density error (# objects / 100m), annotation precision and RMSE (meter) of predicted annotations, with SQ task assignment, unlimited $budget$

City	*Strategy*	Density Prediction	Predicted Annotations	
		Error	Precision	RMSE
Manhattan	*RD*	1.22 ± 0.05	0.04 ± 0.00	3.51 ± 0.07
	ML	1.11 ± 0.02	0.04 ± 0.00	3.51 ± 0.02
Amsterdam	*RD*	0.64 ± 0.03	0.05 ± 0.00	3.50 ± 0.01
	ML	0.55 ± 0.03	0.05 ± 0.01	3.51 ± 0.01

Task assignment strategy. We compare the performance of the multiple-queue (MQ) task assignment strategies against the single-queue (SQ) baseline. In both cases, we employ the ML scheduling strategy, and we simulate an unlimited *budget*. As the MQ task assignment is designed to mitigate the impact of low-quality workers, we progressively skew the distribution of workers towards the low-quality end: in MQ2, half of the workers are of low quality, a third of medium quality, and a sixth of high quality. In MQ3 only low quality workers are simulated. Results ($\mu \pm \sigma$ of five simulation executions) are summarized in Table 4. While recall remains similar to SQ, the precision of the aggregated annotations substantially increases, and the RSME also lowers. Density estimation error and precision also improve. Table 4 reports the performance variations brought by lower quality workers. As the proportion of low-quality workers increases, recall, precision, and RSME progressively worsen. When half of the worker population is of low quality, the recall (-0.03 and -0.01) and RMSE (+0.05 and +0.03) variations in Manhattan and Amsterdam respectively are still acceptable, but the precision is still significantly higher than for the SQ strategy. MQ3 brings more extreme performance decay, especially for recall. In terms of density estimation, precision remains generally similar; density error substantially increases only in the MQ3 configuration.

Table 4: Estimation performance with multiple-queues (MQ) scheduling strategy, and unlimited $budget$.

City	*Strategy*	Aggregated Annotations			Object Density	
		Recall	Precision	RMSE	Error	Precision
Manhattan	*SQ*	0.87 ± 0.00	0.44 ± 0.00	3.04 ± 0.01	1.71 ± 0.02	0.80 ± 0.00
	MQ	0.87 ± 0.00	0.67 ± 0.01	2.89 ± 0.01	0.13 ± 0.01	0.84 ± 0.00
	MQ2	0.84 ± 0.00	0.62 ± 0.00	3.09 ± 0.01	0.11 ± 0.03	0.83 ± 0.00
	MQ3	0.60 ± 0.01	0.46 ± 0.00	3.46 ± 0.00	−0.54 ± 0.05	0.82 ± 0.00
Amsterdam	*SQ*	0.83 ± 0.00	0.55 ± 0.00	2.36 ± 0.01	1.10 ± 0.04	0.73 ± 0.00
	MQ	0.83 ± 0.00	0.87 ± 0.00	2.18 ± 0.01	−0.72 ± 0.02	0.80 ± 0.00
	MQ2	0.82 ± 0.00	0.79 ± 0.00	2.39 ± 0.01	−0.69 ± 0.01	0.78 ± 0.00
	MQ3	0.71 ± 0.01	0.58 ± 0.01	3.32 ± 0.00	−1.37 ± 0.09	0.74 ± 0.00

7.3 Discussion

RQ1. Results show that the designed user interface and overall annotation experience have been positively received by workers. The task has been perceived as moderately difficult, yet the overall judgment has been positive. The annotation, geo-location estimation, and aggregation methods also proved reasonably accurate: despite the approximations introduced by the different technologies (e.g. the α and β angles estimated through the position of the bounding box, Figure 3), our interface achieved geo-location accuracy comparable to GPS-enabled smartphones[11]. A main simplification in the proposed approach concerns the estimation of the camera height h (Section 4.2). The difference of the height of the camera with regard to the terrain level of the targeted object could vary greatly, thus leading to inaccurate annotation geo-location estimation. Workers of high quality achieved remarkable precision, recall, and RSME performance. The impact of low quality workers on the aggregated annotations suggest the introduction of better in-task quality control mechanisms. Considering the average exploration and annotation time, the annotation of Manhattan and Amsterdam would take a total of **4.11** and **5.85** man months, and would cost 33K$ and 46K$ respectively.

RQ2. The Maximum Likelihood (ML) scheduling strategy allows for good density prediction and errors, also at low segment coverage levels. While the performance increase as regards the random scheduling strategy is not overwhelming, it is important to notice that the method is not informed by any prior knowledge about the distribution of physical objects in the city. We believe that the integration of third-party data sources in the density prediction could result in better coverage/recall ratio. On the other hand, we observed that the prediction performance of the geo-location estimation step is relatively low. This is neither surprising nor discomforting, as the developed method is rather simple. The multi-queue task assignment strategy brought clear benefits in terms of aggregated annotation precision, and proved reasonably robust to workers with lower annotation quality.

8 CONCLUSION

In this paper, we have presented an approach to crowd-mapping urban objects from street-level imagery. The proposed method is novel with regard to the combination of an object annotation Web interface for micro-task crowdsourcing with object density and geo-location inference methods. Task scheduling and assignment strategies provide increased performance in terms of cost/quality

[11] https://www.gps.gov/systems/gps/performance/accuracy/

ratio. Through a use case pertaining to the mapping of urban greenery along streets, we have demonstrated the feasibility of the approach, showing very good annotation, geo-location estimation, and aggregation figures, and promising object density prediction performance. This work is only the first step towards a better understanding of how crowd-mapping with street-level imagery can provide accurate and economic object annotation data. In future work, we plan to improve the density and object geo-location estimation methods by incorporating third-party sources (e.g. greenery indexes) and machine learning techniques for object identification. We believe that active learning techniques and hybrid workflows could lead to more accurate and cheaper annotations. Finally, we aim to extend our experiments to other cities and classes of objects.

REFERENCES

[1] Vahid Balali, Armin Ashouri Rad, and Mani Golparvar-Fard. 2015. Detection, classification, and mapping of U.S. traffic signs using google street view images for roadway inventory management. *Visualization in Engineering* 3, 1 (dec 2015), 15. https://doi.org/10.1186/s40327-015-0027-1

[2] Stefano Bocconi, Alessandro Bozzon, Achilleas Psyllidis, Christiaan Titos Bolivar, and Geert-Jan Houben. 2015. Social Glass: A Platform for Urban Analytics and Decision-making Through Heterogeneous Social Data. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15 Companion)*. ACM, New York, NY, USA, 175–178. https://doi.org/10.1145/2740908.2742826

[3] Chris Clews, Roza Brajkovich-Payne, Emily Dwight, Ayob Ahmad Fauzul, Madeleine Burton, Olivia Carleton, Julie Cook, Charlotte Deroles, Ruby Faulkner, Mary Furniss, Anahera Herewini, Daymen Huband, Nerissa Jones, Cho Wool Kim, Alice Li, Jacky Lu, James Stanley, Nick Wilson, and George Thomson. 2016. Alcohol in urban streetscapes: a comparison of the use of Google Street View and on-street observation. *BMC Public Health* 16, 1 (dec 2016), 442. https://doi.org/10.1186/s12889-016-3115-9

[4] Tim Cresswell. 2014. *Place: an introduction.* John Wiley & Sons.

[5] Michael F Goodchild. 2007. Citizens as sensors: the world of volunteered geography. *GeoJournal* 69, 4 (2007), 211–221.

[6] Cullen Gordon. 1961. The concise townscape.

[7] Peter Geoffrey Hall. 1984. *The world cities.* Weidenfeld & Nicolson.

[8] Kotaro Hara, Jon E. Froehlich, Shiri Azenkot, Megan Campbell, Cynthia L. Bennett, Vicki Le, Sean Pannella, Robert Moore, Kelly Minckler, and Rochelle H. Ng. 2015. Improving Public Transit Accessibility for Blind Riders by Crowdsourcing Bus Stop Landmark Locations with Google Street View: An Extended Analysis. *ACM Transactions on Accessible Computing* 6, 2 (mar 2015), 1–23. https://doi.org/10.1145/2717513

[9] Kotaro Hara, Victoria Le, and Jon Froehlich. 2012. A feasibility study of crowdsourcing and google street view to determine sidewalk accessibility. In *Proceedings of the 14th international ACM SIGACCESS conference on Computers and accessibility - ASSETS '12.* ACM Press, New York, New York, USA, 273. https://doi.org/10.1145/2384916.2384989

[10] Kotaro Hara, Vicki Le, and Jon Froehlich. 2013. Combining crowdsourcing and google street view to identify street-level accessibility problems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13.* ACM Press, New York, New York, USA, 631. https://doi.org/10.1145/2470654.2470744

[11] Kotaro Hara, Jin Sun, Robert Moore, David Jacobs, and Jon Froehlich. 2014. Tohme: Detecting Curb Ramps in Google Street View using Crowdsourcing, Computer Vision, and Machine Learning. *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14* Figure 1 (2014), 189–204. https://doi.org/10.1145/2642918.2647403

[12] Darren Hardy, James Frew, and Michael F Goodchild. 2012. Volunteered geographic information production as a spatial process. *International Journal of Geographical Information Science* 26, 7 (2012), 1191–1212.

[13] Brent J Hecht and Darren Gergle. 2010. On the localness of user-generated content. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work.* ACM, 229–232.

[14] Hsun-Ping Hsieh, Shou-De Lin, and Yu Zheng. 2015. Inferring Air Quality for Station Location Recommendation Based on Urban Big Data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '15.* ACM Press, New York, New York, USA, 437–446. https://doi.org/10.1145/2783258.2783344

[15] Allan B Jacobs. 1993. Great streets. (1993).

[16] Johannes Kopf, Billy Chen, Richard Szeliski, and Michael Cohen. 2010. Street Slide: Browsing Street Level Imagery. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2010)* 29, 4 (2010), 96:1 – 96:8.

[17] Karl Kropf. 1996. Urban tissue and the character of towns. *Urban Design International* 1, 3 (1996), 247–263.

[18] Vladimir Krylov, Eamonn Kenny, and Rozenn Dahyot. 2018. Automatic discovery and geotagging of objects from street view imagery. *Remote Sensing* 10, 5 (2018), 661.

[19] Katsumi Kumai, Masaki Matsubara, Yuhki Shiraishi, Daisuke Wakatsuki, Jianwei Zhang, Takeaki Shionome, Hiroyuki Kitagawa, and Atsuyuki Morishima. 2018. Skill-and-Stress-Aware Assignment of Crowd-Worker Groups to Task Streams. *Sixth AAAI Conference on Human Computation and Crowdsourcing* (jun 2018). https://aaai.org/ocs/index.php/HCOMP/HCOMP18/paper/view/17921

[20] Xiaojiang Li and Carlo Ratti. 2018. Mapping the spatial distribution of shade provision of street trees in Boston using Google Street View panoramas. *Urban Forestry and Urban Greening* 31 (2018), 109–119. https://doi.org/10.1016/j.ufug.2018.02.013

[21] Xiaojiang Li, Chuanrong Zhang, Weidong Li, Robert Ricard, Qingyan Meng, and Weixing Zhang. 2015. Assessing street-level urban greenery using Google Street View and a modified green view index. *Urban Forestry & Urban Greening* 14 (2015), 675–685. https://doi.org/10.1016/j.ufug.2015.06.006

[22] Tsung Yi Lin, Yin Cui, Serge Belongie, and James Hays. 2015. Learning deep representations for ground-to-aerial geolocalization. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 07-12-June-2015. IEEE, 5007–5015. https://doi.org/10.1109/CVPR.2015.7299135

[23] Ying Long and Liu Liu. 2017. How green are the streets? An analysis for central areas of Chinese cities using Tencent Street View. *PLOS ONE* 12, 2 (feb 2017), e0171110. https://doi.org/10.1371/journal.pone.0171110

[24] Afra Mashhadi, Giovanni Quattrone, Licia Capra, and Peter Mooney. 2012. On the accuracy of urban crowd-sourcing for maintaining large-scale geospatial databases. In *Proceedings of the Eighth Annual International Symposium on Wikis and Open Collaboration.* ACM, 15.

[25] Yair Movshovitz-Attias, Qian Yu, Martin C. Stumpe, Vinay Shet, Sacha Arnoud, and Liron Yatziv. 2015. Ontological supervision for fine grained classification of Street View storefronts. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE, 1693–1702. https://doi.org/10.1109/CVPR.2015.7298778

[26] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. 2017. The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, Vol. 2017-Octob. 5000–5009. https://doi.org/10.1109/ICCV.2017.534 arXiv:1711.11556

[27] Vítor Oliveira. 2016. *Urban morphology: an introduction to the study of the physical form of cities.* Springer.

[28] S. K. Ploeger, M. Sawada, A. Elsabbagh, M. Saatcioglu, M. Nastev, and E. Rosetti. 2016. Urban RAT: New Tool for Virtual and Site-Specific Mobile Rapid Data Collection for Seismic Risk Assessment. *Journal of Computing in Civil Engineering* 30, 2 (mar 2016), 04015006. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000472

[29] Daniel R. Richards and Peter J. Edwards. 2017. Quantifying street tree regulating ecosystem services using Google Street View. *Ecological Indicators* 77 (jun 2017), 31–40. https://doi.org/10.1016/J.ECOLIND.2017.01.028

[30] Alex Rodriguez and Alessandro Laio. 2014. Machine learning. Clustering by fast search and find of density peaks. *Science (New York, N.Y.)* 344, 6191 (jun 2014), 1492–6. https://doi.org/10.1126/science.1242072

[31] Manaswi Saha, Kotaro Hara, Soheil Behnezhad, Anthony Li, Michael Saugstad, Hanuma Maddali, Sage Chen, and Jon E. Froehlich. 2017. A Pilot Deployment of an Online Tool for Large-Scale Virtual Auditing of Urban Accessibility. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '17).* ACM, New York, NY, USA, 305–306. https://doi.org/10.1145/3132525.3134775

[32] Waldo R Tobler. 1970. A computer movie simulating urban growth in the Detroit region. *Economic geography* 46, sup1 (1970), 234–240.

[33] Griet Vanwolleghem, Delfien Van Dyck, Fabian Ducheyne, Ilse De Bourdeaudhuij, and Greet Cardon. 2014. Assessing the environmental characteristics of cycling routes to school: a study on the reliability and validity of a Google Street View-based audit. *International Journal of Health Geographics* 13, 1 (jun 2014), 19. https://doi.org/10.1186/1476-072X-13-19

[34] Leye Wang, Daqing Zhang, Yasha Wang, Chao Chen, Xiao Han, and Abdallah M'Hamed. 2016. Sparse mobile crowdsensing: Challenges and opportunities. *IEEE Communications Magazine* 54, 7 (2016), 161–167. https://doi.org/10.1109/MCOM.2016.7509395

[35] Jan Dirk Wegner, Steve Branson, David Hall, Konrad Schindler, and Pietro Perona. 2016. Cataloging Public Objects Using Aerial and Street-Level Images - Urban Trees.. In *CVPR.* IEEE Computer Society, 6014–6023. http://dblp.uni-trier.de/db/conf/cvpr/cvpr2016.html#WegnerBHSP16

[36] Jianwei Zhang, Yuhki Shiraishi, Katsumi Kumai, and Atsuyuki Morishima. 2016. Real-time Captioning of Sign Language by Groups of Deaf and Hard-of-hearing People. In *Proceedings of the 18th International Conference on Information Integration and Web-based Applications and Services (iiWAS '16).* ACM, 54–63. https://doi.org/10.1145/3011141.3011143

Web Experience in Mobile Networks: Lessons from Two Million Page Visits

Mohammad Rajiullah
Karlstad University
Sweden

Andra Lutu
Telefonica Research
Spain

Ali Safari Khatouni
Dalhousie University
Canada

Mah-rukh Fida
Simula Metropolitan
Norway

Marco Mellia
Politecnico di Torino
Italy

Özgü Alay
Simula Metropolitan
Norway

Anna Brunstrom
Karlstad University
Sweden

Stefan Alfredsson
Karlstad University
Sweden

Vincenzo Mancuso
IMDEA Networks Institute
Spain

ABSTRACT

Measuring and characterizing web page performance is a challenging task. When it comes to the mobile world, the highly varying technology characteristics coupled with the opaque network configuration make it even more difficult. Aiming at reproducibility, we present a large scale empirical study of web page performance collected in eleven commercial mobile networks spanning four countries. By digging into measurement from nearly two million web browsing sessions, we shed light on the impact of different web protocols, browsers, and mobile technologies on the web performance. We find that the impact of mobile broadband access is sizeable. For example, the median page load time using mobile broadband increases by a third compared to wired access. Mobility clearly stresses the system, with handover causing the most evident performance penalties. Contrariwise, our measurements show that the adoption of HTTP/2 and QUIC has practically negligible impact. To understand the intertwining of all parameters, we adopt state-of-the-art statistical methods to identify the significance of different factors on the web performance. Our analysis confirms the importance of access technology and mobility context as well as webpage composition and browser. Our work highlights the importance of large-scale measurements. Even with our controlled setup, the complexity of the mobile web ecosystem is challenging to untangle. For this, we are releasing the dataset as open data for validation and further research.

CCS CONCEPTS

• **Information systems → Web log analysis**; **Web applications**; **Browsers**; • **Networks → Network monitoring**.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313606

KEYWORDS

Mobile Broadband; Measurements; WEB Browsing Experience; HTTP/2; QUIC; TCP; MONROE

ACM Reference Format:
Mohammad Rajiullah, Andra Lutu, Ali Safari Khatouni, Mah-rukh Fida, Marco Mellia, Özgü Alay, Anna Brunstrom, Stefan Alfredsson, and Vincenzo Mancuso. 2019. Web Experience in Mobile Networks: Lessons from Two Million Page Visits . In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313606

1 INTRODUCTION

We are witnessing two major changes in the Internet. On the one hand, smartphones and mobile broadband (MBB) networks have revolutionized the way people consume web content. On the other hand, web protocols are undergoing a deep rethinking, with the advent of SPDY [33], HTTP/2 [18] and now QUIC [39], that are designed to improve performance and overcome HTTP(S) [27, 48] and TCP bottlenecks.

In this dynamic scenario, the ultimate goal is improving the end-user quality of experience (QoE) for companies that aim at offering better services to their customers. Natural questions arise: Which are the major factors that affect web performance in MBB networks? What are the benefits of HTTP/2 and QUIC? How do mobility or browser choice impact the user experience? Are MBB networks at the point where they are able to offer QoE on par with wired networks?

Measurements are vital to answer such questions and to gauge the impact of the changes in the Internet. Not surprisingly, the research community has put a lot of effort towards empirical quantification of the benefits of new technologies. However, to the best of our knowledge, previous work mainly focused on specific measurement setups, and on specific angles of this otherwise complex ecosystem. The intertwining of technologies, protocols, setups and website design makes it very complicated to answer the above questions, to design and run scientifically sound and robust measurement campaigns.

In this paper, we present a large scale measurement campaign that we carefully designed following the scientific principles of

repeatability and reproducibility. To this end, we leverage the open access MONROE [14] platform that allows us to run measurements with full control of devices scattered in more than 100 locations across four different countries and connected via 11 commercial MBB providers. We instrument the measurement campaign over a period of two months and collect nearly two million page visits of popular websites, which we further augment with metadata about the MBB channel and network status. We dig into this large amount of data to quantify the penalties when accessing the web using 3G or 4G access technology, under mobility, and the benefits of adopting HTTP/2 or QUIC rather than HTTPS. We quantify differences using state of the art statistics tools that allow us not only to obtain the significance of the measurements, but also to study which parameters affect results the most.

The contributions we make in this paper are four-fold. First, we present our methodology to design large-scale and scientifically sound measurements (Section 3). We offer all collected data to the community in the effort of allowing researchers to verify and to extend our findings. Second, we quantify the impact of MBB access technology in both stationary and mobile scenarios. We use two web browsers (Firefox and Chrome), and three objective QoS metrics (First Paint, Page Load Time, and RUMSpeedIndex). With respect to wired access, results consistently show significant penalties in MBB networks, with mobile nodes suffering further penalties due to changing channel conditions, and inter-technology handovers (Section 5). Third, we investigate deeper the benefits of HTTP/2 and QUIC versus HTTPS (Sec 6). In some specific cases (in benchmarking pages), we do see HTTP/2 improvements. In the wild, however, results show marginal benefits and controversial figures. Even when comparing impact on wired networks, we record no significant benefits in 55% of back-to-back visits, with the rest of measurements equally split among HTTP/2 performs better than HTTPS, and conversely. When moving to MBB scenarios, the additional randomness of the setup makes it even harder to observe any significant benefits. QUIC does not change the picture either. Fourth, data analytics unveils the impact of single features that characterize the context in which browsing occurs (Section 7). Our results show that there is a large number of features that significantly impact web performance, and, ultimately, QoE. Among these, the MBB access technology and mobility context, along with webpage composition and the browser used are among the most important.

Before moving into the details of our contributions (Sections 3-6), we present background and related work in Section 2 and we follow up our analysis with concluding remarks in Section 8.

2 BACKGROUND AND RELATED WORK

There are mainly three approaches for measuring the performance and reliability of MBB networks: (i) crowd-sourced measurements [44, 49, 54], (ii) operator's measurements [31, 32, 51, 52], and (iii) measurements collected using dedicated infrastructure [16, 38, 50]. In this paper, we collect data from the MONROE dedicated infrastructure to gain full control over the measurements and systematically collect a rich quality dataset over more than two months.

With respect to the web, HTTP/1.1 [27] has been the de-facto standard for loading webpages since 1999. With the increasing complexity in webpages, HTTP/1.1 (also HTTP over TLS or HTTPS -hereinafter H1s- [48]) has been shown to limit the performance of web access [56] due to problems such as inefficient use of concurrent TCP connections [47] and header overhead. To improve web performance, Google proposed SPDY [33], which soon led to HTTP/2.0 -hereinafter H2- [18]. The latter uses a single multiplexed TCP connection per domain, allowing request prioritization. Efficiency is further optimized through header compression. However, TCP is still a bottleneck because of handshaking latency and head-of-line blocking. To address these shortcomings, Google proposed QUIC [34, 39] that uses UDP instead of TCP.

The relative performance difference between HTTP versions has received much attention [13, 20, 25, 26, 30, 40, 45, 58–61, 63]. The results are mixed, hence it is hard to make a clear conclusion on which version is better and under which circumstances. Moreover, there is no systematic attempt to characterize HTTP performance for MBB users. Similarly among [19, 23, 34, 39, 42] we found only few experimental works on QUIC performance in operational networks [39] with real webpages, while rest of them considered synthetic webpages with testbed settings. We aim to present a large-scale web measurements campaign using operational MBB networks across different countries. We not only measure the protocol difference from different vantage points, but also set the operators performance in a common scale. We run measurements using two popular browsers (Chrome and Firefox) to present how the protocols behave in different browsers.

Previous works also differ in the metric they chose to evaluate performance. Work in [19, 25, 26, 39, 40, 42, 57, 60, 64] used the page load time (PLT), a metric primarily based on OnLoad event fired by the browser. Unfortunately, PLT does not tell much of the user experience [2, 21, 35, 55]. An alternative metric is First Paint, which tells how long it takes to see something on the screen [21, 28]. SpeedIndex captures the user perceived experience [8, 10, 11]. However, it requires to *film* the page loading process. The RUMSpeedIndex method was proposed to estimate the SpeedIndex using "resource timing" data [4, 6, 9]. ObjectIndex and ByteIndex are similar in goal [21]. Additionally, [29] proposed the "3rd Party Trailing Ratio" metric, which measures the fraction of download time for the 3rd party assets on the webpage critical path. In our measurements, we consider Page Load Time, First Paint and RUMSpeedIndex.

We tackle the need for understanding the relationship between web performance and network characteristics. For example, while generating a model to determine a webpage's QoE, Balachandran et al. [15] found evidence of the impact of the number of users and radio access technology (RAT) handovers on the webpage performance. They discovered that the performance is greatly affected by a website's own complexity (e.g., number of objects, domains visited) with some influence of the time-of-day feature. They show that understanding non-binary QoE metrics (e.g., latency, download progress, etc.) requires regression modeling. Here, we use regression to dig into our measurements and bring to light the correlations between various aspects of the browsing process and the experience of the users.

3 MEASUREMENTS DESIGN

For our extensive measurements campaign, we engineer a flexible setup that on the one hand allows us to investigate as many different scenarios as possible, and on the other hand follows the best practice of scientific approach by letting us control as much as possible the experimental setup, with reproducibility in mind. In the following, we describe the experiment design and the collected dataset.

3.1 Experimental Setup

For our experiments, we rely on the MONROE platform, the first European open access platform for independent, multi-homed, and large-scale mobile measurements on commercial providers. Overall, the geographical extent of our measurement campaign covers four countries in Europe (Italy, Norway, Spain and Sweden). The platform comprises a set of 100 nodes equipped with Ethernet, WiFi and 3G/4G interfaces with commercial MBB subscriptions. Nodes operate either under mobility (e.g., on-board of trains or buses) or in stationary scenarios (e.g., in laboratories, or hosted by volunteers in homes).

We leverage a customizable Docker container called MONROE-browsertime [5] (based on [7]) that we specifically engineered and deployed on MONROE nodes. We configured MONROE-browsertime to mimic a mobile device browser (by setting both the screen resolution and the user-agent accordingly) to retrieve the mobile versions of the visited pages. With it, we direct the browser to load a page and, at the end of page rendering, execute a custom JavaScript script to collect a large number of metrics. We use the X virtual framebuffer (Xvfb) [12] for display emulation to let the browsers actually render the webpages. MONROE-browsertime provides a configurable experiment template to enable web measurements. We configure each measurement by controlling (i) the network to test (Ethernet, or the desired MBB interfaces), (ii) the browser (Firefox 56.0.1 or Chrome 64.0.3282.186), and (iii) the web protocol (H1s , H2, or QUIC).[1] A combination of these parameters builds an *experiment setup*.

We select a list of target pages to visit. Considering the time constraints of the MONROE platform, we select 20 target pages on popular websites with H2 support from the most viewed sites in the Alexa [3] top ranking, see Table 1. We avoid the landing page in case it is too trivial (e.g., visiting `https://instagram.com/leomessi/` instead of `https://instagram.com/`). This ensures that our selection covers a wide range of user interests in terms of topics, including social networks, video, career, education, search engine, travel help, news, wiki and shopping. All websites expect TLS connections by default. This allows us to run fair comparison between H1s and H2.

Given a network to test, we shuffle the order of pages to visit. Next, we visit each page with every browser and protocol combination, in a random order. The visit of all pages with one network setup constitutes a *run*. Browser caches and cookies are active during each run, and cleaned when the last page has been tested. Note that we use separate profiles for the same browser, one for each protocol, so that caches are separated (i.e., visits with H1s do not interfere with next visits to the same pages with H2 and vice-versa). We execute four rounds of measurements per day.

[1]The nodes use the mobile carrier DNS resolver consistently with the operator currently in use.

3.2 Web QoE Metrics

In this paper, we track three main metrics for our analysis: Page Load Time (PLT), FirstPaint (FP) and RUMSpeedIndex (SI), which we detail next. The tool derives these metrics from browser timing metrics [1], [41] that record the timing of different rendering phases of a page, from the initial DNS resolution to each HTTP request, from JavaScript processing to objects rendering.

First Paint (FP): It corresponds to the time when the browser starts rendering the first element of the page. This happens as soon as the first element of the page has been fetched and processed, and after the downloading of all needed elements (e.g., stylesheets).

Page Load Time (PLT): This is the time the last object in the page has been downloaded. It occurs when all the HTML files and any sub-resources (images, fonts, css, videos, etc.) are loaded. Note that not all these elements are needed to complete the rendering of the visible portion of the page.

RUMSpeedIndex (SI): It monitors the rendering process of the page by tracking the evolution of visible rectangles with elements that loaded external resources in a page. The original SpeedIndex requires to film the rendering process, and to postprocess the video to observe changes. Given the limited resources of the MONROE nodes (in terms of CPU, storage and communication bandwidth), we opt for the RUMSpeedIndex approximation, which uses the sequence of events as reported by the browser to estimate the time in which the visible portion of the screen would be modified [9]. Intuitively, it calculates the likely time that a paint event happens, given the processing and downloading of the various elements by the browser. This is considered a QoE approximation, since it considers the evolution of the rendering process as seen by the user.

Given the interplay of objects, rendering, and visible portion of the page on the screen, there is no clear ordering on the metrics. For instance, the rendering of a page can start after one, or some, or all objects have been downloaded. Similarly, the rendering process may involve a portion of the area which is not currently on the visible part on the browser window. In addition, analytics objects are typically downloaded as last elements, after the rendering is completed, thus inflating the PLT. For this, it is consistent to compare results considering the same metric, but it is not appropriate to compare across different metrics.

3.3 Metadata Metrics

Context information is crucial. MONROE gives access to the information about network, time and location, as well as metadata from the mobile modems, including cell ID, signal strength, and connection mode. We log all this information, and associate it to each experiment for later post-processing, to understand the impact of the different operator configurations, the radio coverage, and all the time varying parameters that may impact the page rendering process.

More specifically, we collect the following parameters: i) **Experimental context**: This includes the type of browser, the protocol, the node type (stationary/mobile) and the distance the node travelled during an experiment. ii) **Access Network context**: This

Table 1: List of the target webpages and their corresponding number of domains and objects. The table details the % of H2 resources in each page obtained using Chrome.

	Wikipedia	*Live*	*Stackoverflow*	*twitter*	*Youtube*	*Instagram*	*Microsoft*	*Etsy*	*Kayak*	*Ebay*	*Yelp*	*Flickr*	*Reddit*	*Facebook*	*Tmall*	*Imgur*	*Coursera*	*Theguardian*
# obj.	7	11	21	21	37	40	44	49	52	54	54	62	66	71	97	101	116	277
# domains	1	4	9	7	11	5	8	6	17	20	26	6	21	6	18	27	45	59
% of H2 resources	100	73	76	100	100	100	80	100	100	78	87	97	79	100	79	84	84	70

includes parameters from the the Radio Access Technology (RAT) (more specifically for 3G and 4G technology) such as radio status before the start of the experiment (Initial RAT, Initial Reference Signal Received Quality (RSRQ), Initial Reference Signal Received Power (RSRP), Initial Received Signal Strength Indicator (RSSI)) and the radio changes throughout the experiment (median values for RSRQ, RSRP, RSSI parameters, the number of RAT handovers), and average RTT against the target webpage server (measured via Ping). iii) **Web context:** This includes metrics that characterize the single web page visit, such as the number of objects or the total amount of data being downloaded.

3.4 Dataset Description

For experiments in this paper, we deploy MONROE-browsertime on 100 MONROE nodes, measuring 11 mobile operators in 4 countries. We ran our measurement campaign for more than 2 months, from 1st of April 2018 to 4th of June 2018. This dataset focuses on measurements considering H1s and H2 protocols. In total, we collected performance for about 1.8 million page visits, as detailed in Table 2, of which 550,000 under mobility in trains, buses or trucks. In addition to MBB tests, we run experiments using wired connectivity for those nodes operating in campus networks and connected via regular 100Mb/s or faster Ethernet interfaces. These experiments constitute our baseline.

As a first step, we performed data cleaning by identifying anomalies in the data. Indeed, we observed inconsistencies in the results from `Google search` - Chrome and Firefox were showing different results for the same search url - and `Linkedin` - this site was blocking visits from our nodes, detecting the repeated page visit as illegitimate crawlers, or attacks. We have discarded these sites from any further processing. We also removed those experiments in which either the browser could not complete the page download or some metadata were missing.

We run a separate measurement campaign for measuring the performance of the QUIC protocol as compared to H2. This batch of experiments ran during a week, from the 5th to the 10th of June 2018. We focused here on websites that supports QUIC (version 39): `Google search`, `Youtube`, `Lightspeedtech`, `Keycdn`, `Meetup`, `Justwatch`, and `Free-power-point-templates`. We used only Chrome browser (the only one supporting QUIC). After performing data cleaning operations, we collected approximately 80 000 samples.

To summarize, our full dataset includes nearly two million page visits, a deluge of data that – to the best of our knowledge – constitutes the largest experiment of mobile web performance tests on commercial operators. We open the data for further processing and to enable reproducibility.[2]

Table 2: Statistics on the dataset; the country shows where the subscription is active.

Operator Name	# Tot. Measurements	# Mobile
Telia (SE)	246 839	41 936
Telenor (SE)	249 459	45 580
Tre (SE)	223 050	43 469
Telenor (NO)	226 126	146 781
Telia (NO)	251 960	155 522
ICE (NO)	177 133	72 461
TIM (IT)	99 778	7 793
Vodafone (IT)	86 164	15 110
Wind (IT)	89 799	20 474
Yoigo (ES)	142 502	0
Orange (ES)	44 528	0
Total	1.8M	550K

Dataset limitations. Despite all the care we took in designing our experiment to ensure that they are extensive, repeatable, and with limited biases, we discuss some limitations that we could not avoid.

We measure using MONROE hardware, which integrates three 3G/4G MC7455 miniPCI express mobile broadband modems supporting LTE CAT6 on top of PC Engines APU2D4 boards [53]. The end-user experience is strongly correlated with the device of the end-user [24, 43], especially for web browsing. Our design aims to remove device bias and capture a wide geographical footprint with one single type of device.

We use commercial-grade mobile subscriptions for our measurements. While this represents what end customer can get, we do not know which specific configuration each MBB operator is using for each of these (e.g., in terms of QoS, or presence of proxies commonly found in MBB networks [36]). As such, our results do not generalize all possible subscriptions, nor in different locations.

The differences in commercial offers reflect in the different data quota which in turn affects the dataset we can collect. We explicitly opted to limit the number of pages in our tests and prefer several repetitions of the same test to obtain a large dataset where we can obtain statistically significant results for each page. Completing a single experimentation takes about 2 hours, making it impossible to cover more pages.

The disparities in the number of samples we could collect from each node stem from the way the MONROE scheduler assigns experiments to the nodes based on their available monthly data

[2] https://zenodo.org/record/2574157

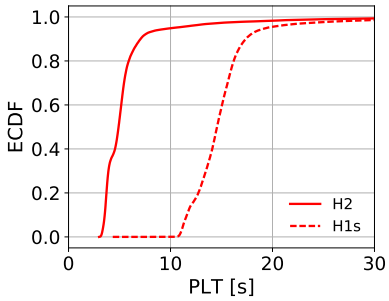


Figure 1: ECDF of PLT when accessing httpvshttps.com using H2 and H1s.

quota. For example, we observe in Table 2 that the number of samples for Orange (ES) (limited data plan, 10GB) is much smaller than the number of samples we collected for Telia (SE) (very large data plan, 200GB). Moreover, because of the MONROE deployment strategy, we do not have any mobility measurements in Spain.

The RUMSpeedIndex represents a client-side approach for measuring the Speed Index metric using Real User Monitoring (RUM). Though this metric is appropriate for our experimental setup and goes along the idea of the original Speed Index metric of measuring how much of the "above-the-fold" content is visually complete until the web page has finished loading, it also has a series of limitations. Specifically, it only works for browsers that support Resource Timings, it does not handle content within iframes and does better for Chrome which supports reporting a "first paint" event.

4 VALIDATION OF THE EXPERIMENTAL SETUP

Before digging into measurements, we validate out setup by using a specifically crafted page that allows one to gauge the benefites of H2 vs. H1s, the www.httpvshttps.com page. Hosted by cloudflare, it contains 360 not-cacheable identical images (each 20×20 pixels, 1.7 kB each) with a total of 604 kB. All images are hosted from a single domain. This scenario is particularly beneficial for H2 which can multiplex all the HTTP requests over a single TCP connection and benefit of header compression. Conversely, H1s would open a large number of TCP connections each suffering from TCP and TLS handshaking overhead and TCP's slowstart. H2 is therefore expected to provide much faster web performance than H1s.

We run a preliminary validation test on all MONROE nodes for every hour continuously for five days producing a total of 130 000 page visits. Figure 1 shows the Empirical Cumulative Distribution Function (ECDF) for PLT for both browsers. H2 provides much better performance than H1s as expected. In nearly 90% visits, H2 manages to load the whole page under 5s whereas using H1s, the PLT rises well over 15s for 80% visits. The behaviour stays similar when we separate the browsers (Firefox and Chrome). This simple test shows that our testbed is giving consistent and realistic results.

5 BROWSING CONTEXT IMPACT

In this section, we explore our measurement results and highlight the impact of major parameters, such as MBB access and mobility. We then dig into more details to quantify the impact of radio access status leaving the discussion on web protocol impact to Section 6.

5.1 Mobile Access and Browser Impact

We aim to assess the impact of MBB access and mobility compared to wired access. Figure 2 shows results separately for Chrome (on the top) and Firefox (on the bottom). For each metric (notice the different scales), we calculate the ECDF for i) stationary nodes with wired connection (black solid line), ii) stationary nodes with MBB connections (red dashed line), and iii) mobile nodes with MBB connections (blue dotted line). Smaller values correspond to better performance.

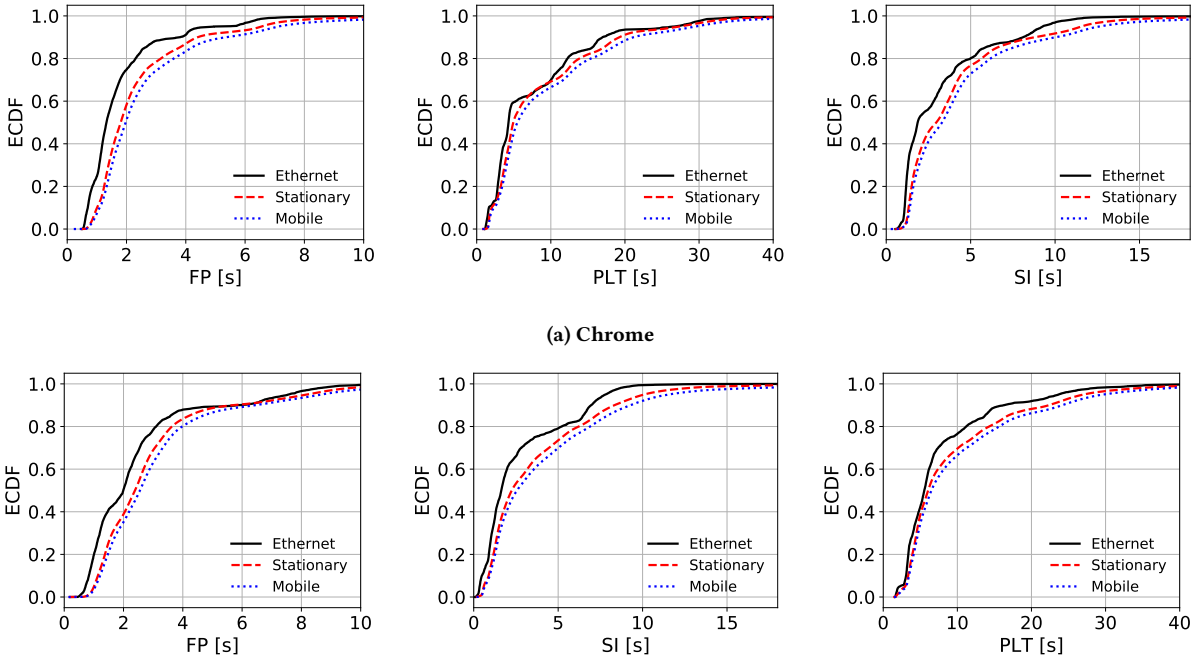
Few considerations hold: First, wired connections have better performance than 3G/4G stationary nodes, which in turn perform better than 3G/4G mobile nodes (consistent across all metrics). Interestingly, the performance degradation with respect to wired high-speed access is much more visible than the impact of mobility. For instance, the FP in Chrome (top-left figure) shows that the median worsens by 36.5% when comparing Wired and MBB access, and only by 9.6% when comparing stationary and mobile nodes. We verified this to be consistent in single pages, and found more complex pages to be more impacted (as expected).

Second, all ECDFs exhibit a heavy tail in measurements, with the mass of the distribution concentrated on low values and a non-negligible number of samples that reach much higher indexes. This is more evident for MBB tests. For example, in the case of FP for Chrome, we note 60-80% of pages start the rendering process in 2-3 seconds. But more than 20% of pages have to wait for more than 4-6 s (i.e., a 2x factor). This heavy tailed distribution is consistent on all pages, hinting at variability among single experiments.

Third, and less intuitive, results on Firefox and Chrome are hardly comparable. For example, for the FP metric (top plots, Figure 2), Firefox clearly has worse performance than Chrome. However, when checking the PLT (bottom plots, Figure 2), the difference is much less evident. To analyze this in more detail, we focus on the measurements we collect from stationary nodes in Sweden, and compare the impact of the browser. Figure 3a shows much less evident results, with Chrome performing slightly better than Firefox. However, when restricting to a single website, results can change dramatically, depending on the target website. For example, for the case of ebay in Figure 3b, we note that Chrome is faster and less variable than Firefox.

Fourth, the impact of the number of objects of the target website (Figure 3c) differs depending on the metric. Page rendering may start only after some specific object has been downloaded (e.g., fonts, CSS, and background images) which may be loaded later, as captured by FP, but is not directly related to the total number of objects on the page. The PLT metric, on the other hand, tends to be affected by pages that include analytics services that are loaded as the last object – and often after executing complex scripts. Thus, we observe that PLT may increase considerably with number of objects.

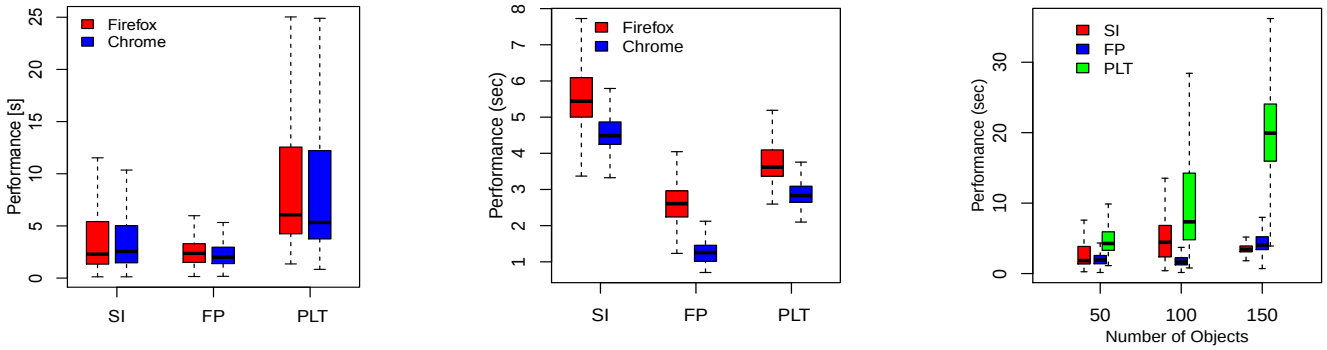
Takeaway: MBB access and mobility are factors that impact performance in a consistent way. However, even in controlled environments we observe a lot of variations for all performance metrics and there are different factors that impact the page rendering process in a not trivial way. We investigate this in detail in Section 7.



(a) Chrome

(b) Firefox

Figure 2: Browsing performance on commercial MBB networks in two different scenarios (moving and stationary mobile devices) considering all three metrics for two browsers. We use the browsing performance on fixed Internet provider connectivity (Ethernet) as a baseline.



(a) Browser impact (all). **(b) Browser impact (*ebay*).** **(c) Websites structure impact.**

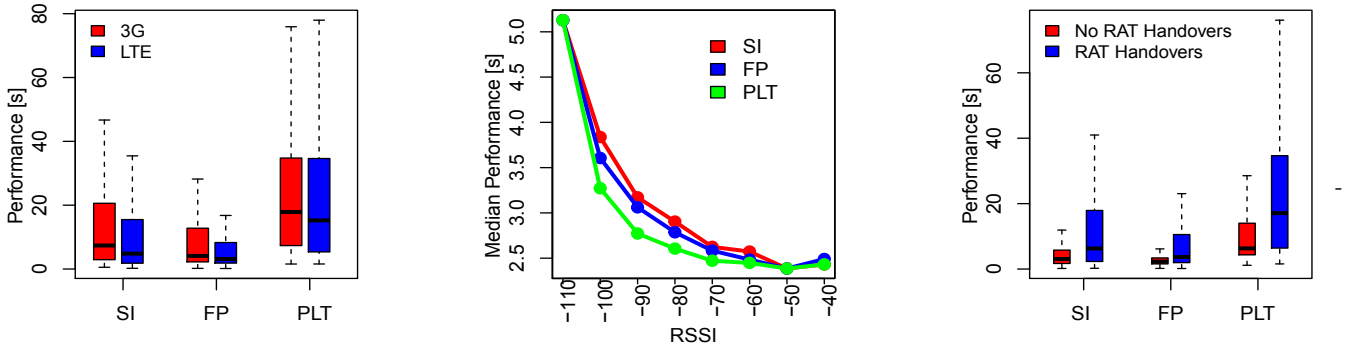
Figure 3: Impact of experimental context (browser type and target website) on PLT, FP, and SI, aggregated over all webpages in Sweden.

5.2 Network Context Impact

As the capacity of the mobile networks continues to increase, users expect better performance. As we saw in the previous section, MBB access and mobility significantly impact the user experience and it correlates with highly varying network conditions. With MBB technologies that are much more complex than wired access, we need to delve into more details to understand what is the impact of particular setup.

We first observe the impact of the Radio Access Technology (3G and 4G/LTE). Using the experiments from nodes within Sweden under mobility scenarios, Figure 4a captures the impact of RAT on web performance. The better performance offered by 4G/LTE technology benefits the FP and SI albeit not as dramatically as one would expected [46]. Instead, the PLT seems to be marginally affected. Again, this is due to the PLT definition which needs to wait for the last object to be downloaded. In most cases, this is inflated by analytics objects (notice the median in the 20 s, a time that would translate in poor QoE for users).

Coverage of MBB networks is also expected to clearly affect performance. This is captured in Figure 4b, where the median performance clearly degrades under poor signal strength conditions. The figure shows results from *Reddit* web experiments performed in Sweden. Interestingly, the RSSI has a non-linear effect, with



(a) RAT impact (all). **(b) Signal Strength Impact (*Reddit*).** **(c) Inter-RAT Handovers Impact.**

Figure 4: Impact of network context on PLT, FP, and SI, aggregated over all webpages in Sweden.

sudden degradation for RSSI smaller than -80dBm. Above -60dBm, we observe no further improvement, an important observation for network operator's looking for optimal network configuration.

As a last parameter, we showcase the impact of the number of handovers under mobility, reported in Figure 4c. Here we compare experiments with no inter-RAT handover against experiments where at least one inter-RAT (3G to 4G or vice-versa) handover has been observed. Results clearly show the penalty introduced by inter-RAT handovers, with a worsening factor of 2x on the average, and worst case up to 5x slower (when several inter-RAT handovers are suffered).

Takeaway: Coverage and inter-RAT handovers [17], are among the most prominent causes of web performance degradation. However, their impact is not linear. Faster technologies bring some, but limited benefits. As known for wired networks [46], for MBB we observe that the performance gets increasingly latency-bound when network speed increases.

6 WEB PROTOCOL IMPACT

This section provides an in depth quantification of the benefits of using new protocols such as H2 and QUIC compared to the original H1s standard.

6.1 H1s vs H2 Impact

We first compare performance when using H1s and H2. We consider back-to-back visits to the same page. The first visit uses one protocol and the second visit uses the other one. Node, operator, browser and location are same for both visit. Based on the PLT reported in Figure 2, these two visits can happen in a limited time window (20 urls * 2 protocols * 20 seconds = 800 s ∼ 13.3 minutes) in more than 90% of the experiments. Thus, we pick any two visits that happened within 15 minutes between each other, and directly compare page visit performance. The aim is to provide a sound comparison between the two protocols under as much similar conditions as possible.

Let $M(i)$ be the i-th visit of metric $M \in \{FP, SI, PLT\}$. We compute the *protocol gain* $\Delta M(i)$ as :

$$\Delta M(i) = \frac{M_{H2}(i) - M_{H1s}(i)}{\max\{M_{H2}(i), M_{H1s}(i)\}}$$

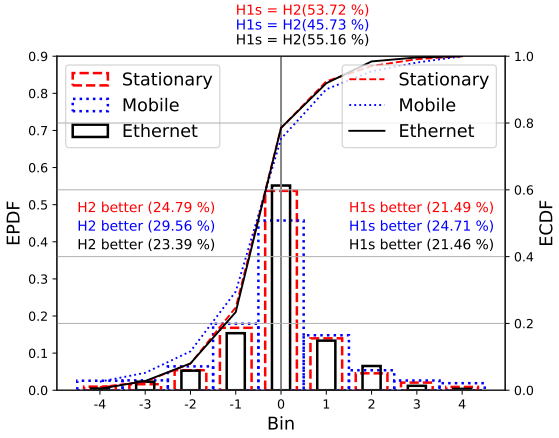
$\Delta M \in [-1, 1]$ – with negative (positive) values when H2 performs better (worse) than H1s on the i-th visit. Next, we compute the distribution of the $\Delta M(i)$ over all experiments, and quantize results into 9 bins centred around zero.[3] The rationale is to identify ranges for which performance is equivalent (center bin), slightly better/worse (the first bin on the left or right), moderately better, much better, extremely better. We opted for 5 bins to mimic the Mean Opinion Score.

Figure 5a shows the distribution of the protocol gain for SI for Chrome. Left y-axis, right y-axis, and x-axis illustrate the Empirical Probability Density Function (EPDF) of samples in each bin, the ECDF, and bins, respectively. We distinguish between three different cases: mobile nodes with MBB connections (blue curve), stationary nodes with MBB connections (red curve) and stationary nodes with wired connection (black curve). Bars show the EPDF; lines the ECDF. Ideally, when H2 is better than H1s, the probability of having $\Delta M(i) < 0$ would be larger than the probability of having $\Delta M(i) > 0$. We observe that the distributions are (i) having similar modes with zero mean – independently from the scenario, (ii) similar distribution, but (iii) slightly more skewed for MBB networks than for wired access. Specifically, in about 50% of cases, ΔM falls in the central bin (i.e., the relative performance gain of H2 and H1s are within 10%). Moreover, cases where either H2 or H1s have better performance are approximately balanced, although there is a moderate bias towards negative values (H2 has better performance than H1s). Finally, mobility tends to add more variability in the experiment, so that ΔM becomes larger, but equally favoring H1s or H2.

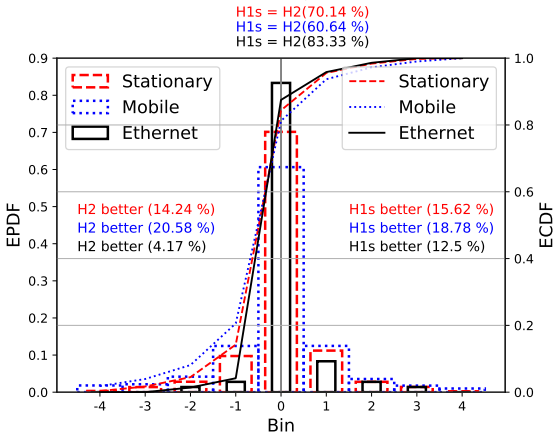
We next drill down our analysis to look separately at each website, to verify if some pages do see some improvement in using H2. Figure 5b is a notable example which reports the protocol gain for *Wikipedia* using Chrome. We observe that the differences are perfectly balanced. This is a generic result, which holds true for SI, PLT and FP as well as for Chrome and Firefox. The only case we observe H2 having a sizeable protocol gain is the case of *R*eddit, detailed in Figure 5c while using Chrome. In this case, we observe that H2 outperforms H1s in 52% of tests. To illustrate the actual absolute improvement as previously done in Figure 1, Figure 6 details the ECDF of all experiments involving these 2 websites for the SI metric. Figures confirm the results shown by the protocol gain: *Reddit* does benefit of H2, while *Wikipedia* obtains no gain.

All measurements presented so far offer a qualitative measure of the impact of protocols. Next, we apply statistical analysis techniques to compare measurements referring to different datasets to precisely quantify the

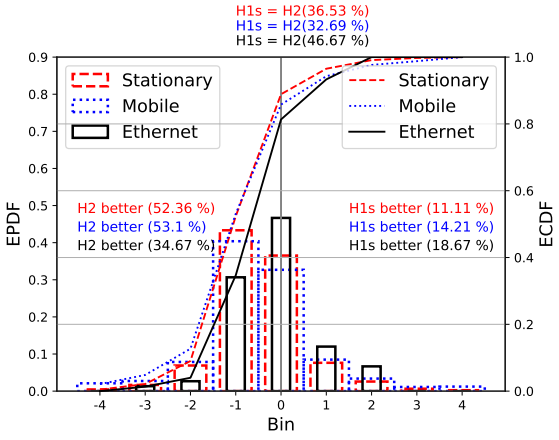
[3]We assign samples to bin in $[-4, -3, -2, -1, 0, 1, 2, 3, 4]$ if $\Delta M(i)$ falls in $[(-1, -0.7), (-0.7, -0.5), (-0.5, -0.3), (-0.3, -0.1), (-0.1, 0.1), (0.1, 0.3), (0.3, 0.5), (0.5, 0.7), (0.7, 1)]$.



(a) All pages



(b) *Wikipedia*

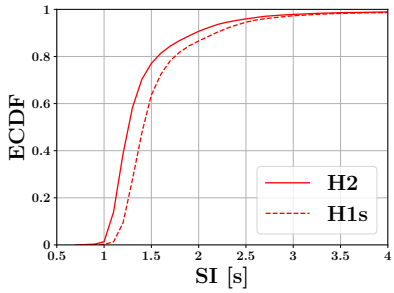


(c) *Reddit*

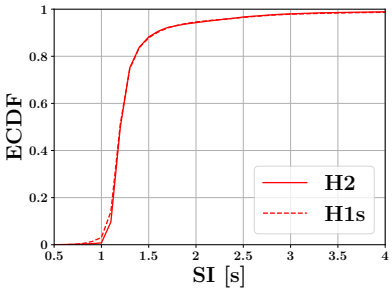
Figure 5: Distribution of ΔM for SI with Chrome, for stationary MBB access, mobile MBB access, and wired access.

performance difference between the two protocols. Our goal is to answer how (dis)similar are the distributions of the H1s and H2 datasets.

In the literature, there are different well-known Statistical Distance Metrics (SDM), each with its own properties and limitations. In this work,



(a) SI *Reddit*



(b) SI *Wikipedia*

Figure 6: Some example ECDFs for SI using H1s and H2 with Chrome.

we chose the Jensen-Shannon divergence (JS_{div}), which is defined as:

$$JS_{div} = \sum_i \left\{ \frac{1}{2} p_i \ln\left(\frac{p_i}{\frac{1}{2}p_i + \frac{1}{2}q_i} \right) + \frac{1}{2} q_i \ln\left(\frac{q_i}{\frac{1}{2}q_i + \frac{1}{2}p_i} \right) \right\}$$

where p_i and q_i (relating respectively to H1s and H2) are the EPDF values generated by samples falling in the i-th bin. JS_{div} is a statistical measure based on the Kullback-Leibler divergence. JS_{div} adds symmetry (i.e., $JS_{div}(p, q) = JS_{div}(q, p)$), and bounded image to the Kullback-Leibler divergence. In fact, the reason we chose the JS_{div} is to obtain a symmetric bounded value for our comparisons. JS_{div} is equal to 0 if $p = q$, while it reaches ln(2) for two completely disjoint distributions.

Intuitively, when $JS_{div} > Th^+$, the difference between the two EPDFs is significant. Conversely, differences are negligible if $JS_{div} < Th^-$ and when $JS_{div} \in [Th^-, Th^+)$ the difference is observable. We select the threshold values $Th^- = 0.02$ and $Th^+ = 0.1$ as common in statistics. For a more in depth discussion of which SDM to use, and the proper usage of thresholds we refer the reader to [22]. To estimate the EPDF for H1s (p) and H2 (q) separately, we consider stationary nodes only. We use a bin size of 100 milliseconds, and we compute the frequency p_i, q_i of samples falling in each bin. Then, we compute the JS_{div}. Table 3 reports the results for three websites for all browsers and metrics.[4] We observe that *Reddit* for the Chrome browser is the only case where $JS_{div} = 0.124 > Th^+$ for FP, hence falling in the range of a significant difference (red color). The difference for PLT and SI are less evident, but still sizeable. *Wikipedia* for the Chrome browser shows instead no statistically significant changes (green color) – as discussed before.

We further carried out analysis to statistically compare performance of H2 and H1s by conditioning on operators, independent from the website and other conditioning variables. In all cases, we did not observe any statistically significant difference in the performance observed when using H1s or H2. These results are not presented here due to space considerations.

[4]We explored all websites. Results are omitted for the sake of brevity.

Table 3: Jensen-Shannon divergence of FP, PLT and SI comparing H1s vs H2 for three websites.

Website	FP Chrome/FFox	PLT Chrome/FFox	SI Chrome/FFox
ebay	0.006/0.004	0.009/0.006	0.009/0.005
reddit	0.124/0.002	0.025/0.007	0.079/0.002
wikipedia	0.008/0.003	0.006/0.005	0.009/0.003

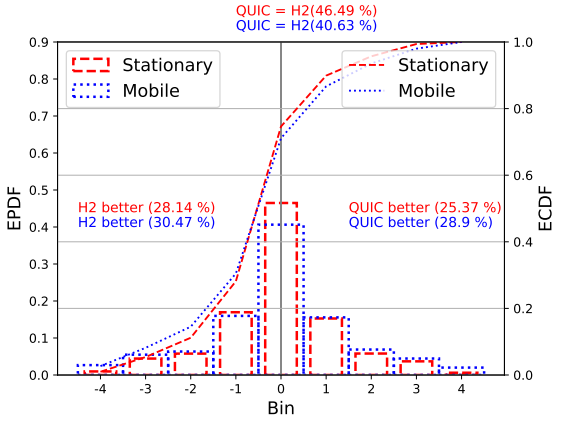


Figure 7: QUIC vs. H2: Distribution of ΔM for SI.

Takeaway: The benefit for H2 is very limited or absent in our tests. One reason that has been identified could be that some of the key H2 features are still not in use at large in the server side [64]. Similarly, results from the work in [62] show that the current prioritization in off-the-shelf browsers is not yet optimized for H2. Also, we have observed that domain wise resource placement is quite similar for both the H1s and H2 cases. This sharding influences the potential gain from multiplexing in H2. Similar observation is mentioned in [37]. Furthermore, for a majority of the webpages in our tests, we have noticed that a considerable number of resources were downloaded using H1s even when the browser used H2 (see Table 1). This clearly limits benefit of H2. At last, MBB access and mobility scenarios exhibit a very high variability in performance – due to the more random nature and complexity of MBB scenarios. This randomness dominates over the benefit of H2.

6.2 QUIC vs H2 Performance

Here, we report the performance comparison between QUIC and H2. We follow the same process as in the previous section to compute the protocol gain. Here we use the set of websites that support QUIC (Section 3.1). Accordingly, we examine H2 and QUIC back-to-back measurements towards the same website and compute the protocol gain similar to Section 6.1. As before, $\Delta M(i) < 0$, indicate H2 is better than QUIC.

Figure 7 presents the performance of QUIC and H2 we capture with SI using Chrome over all websites. Here we show two scenarios, stationary nodes with MBB connections (red curve) and mobile nodes with MBB connections (blue curve). From the distribution, it is clear that QUIC and H2 performance are very similar. Again we see an increase in the variability of results for mobile nodes, but still with similar relative performance gains for the two protocols. Although not displayed, we observe the same trends for FP and PLT. When the performances differ, we see H2 being better more often than QUIC for both stationary and mobile nodes. We observe that most QUIC websites include external resources where the browser needs to fall back to H1s/H2. This may introduce extra delays and limits the protocol gain QUIC can deliver. Such scenarios were never considered in previous works that used synthetic pages and controlled servers [19, 23, 34, 42]. These show more prominent (but artificial) gain using QUIC.

Next, we offer a closer look. In Figure 8a, we illustrate the performance comparison between QUIC and H2 using an ECDF of all experiments for SI. Again, we do not observe any significant difference. This observation holds true for FP and PLT as well. We only observe some notable differences for two websites: *Free-power-point-template* where H2 performs slightly better (Figure 8b) and *Youtube* where QUIC shows slightly better performance (Figure 8c). In the former case, we observe some heavy resources which hurt QUIC performance due to the use of pacing [42].

Takeway: We observe even less benefits for using QUIC instead of H2. Our detailed analysis of QUIC traces show that some objects from the QUIC compatible sites are still not transported using the QUIC protocol, which we believe a hurdle to put QUIC and H2/H1s in a common performance scale. Again, many factors also influence and bring variability into mobile Web browsing performance, overshadowing the impact of the protocols and leading to mixed performance results.

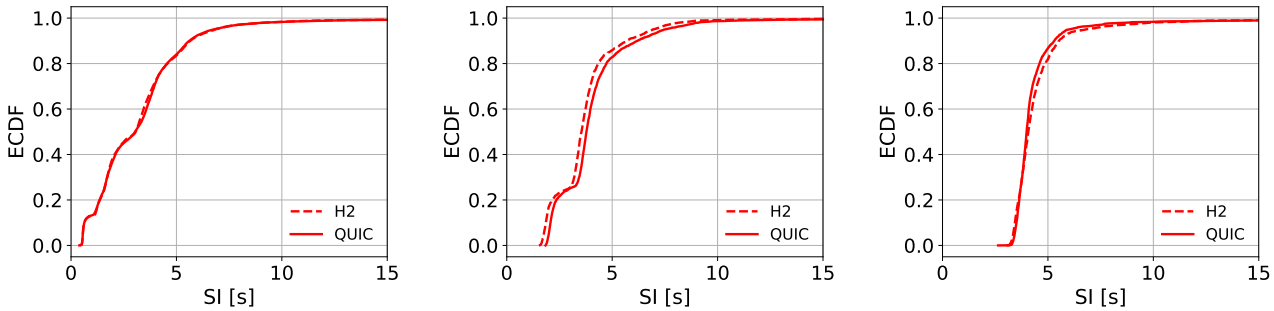
7 MODELLING WEB PERFORMANCE

In our analysis so far, we explored a large dataset of mobile web measurements and investigated the different factors that impact the mobile web performance by considering three main QoE metrics. Our statistical analysis, however, partially captures how the collection of features characterizing the web browsing process actually impacts the quality of the user session. Even when using a platform specifically meant to enable comparison and eliminate biases, we show that the complexity of the experiments is high and multiple factors can influence the results and the experience of the end-user. In this section, we aim to use data analytics, more specifically regression models, to formally capture the correlation between the set of features that characterize the browsing context of the user (e.g., radio signal quality, radio access technology, mobility etc.) with the quality of experience for that same user (PLT, SI or FP).

We first use *multiple-linear regression (LM)* to investigate how the web QoE metrics (e.g., PLT, FP and SI) vary with respect to different features that characterize the context in which the user surfs the web. We include the following features in our analysis: the node type mobile/stationary (Node Type), number of vertical handovers occurring throughout the duration of one website visit (RAT Handovers), the number of objects downloaded for a specific website under a given experimental configuration (# of Objects), the distance travelled by the node during one website visit (Distance), values of radio strength parameters at the beginning of the experiment to one website (RSRP, RSRQ, RSSI), average value of round-trip-time we measured against the target website during one visit (Avg. RTT), the radio access technology type (RAT), the browser type (Browser), the web protocol version (Protocol). To identify the most prominent features, in the linear regression model we use standard step-wise sub-scheme (as a Wrapper method) and subsequently apply the filter method to retain only the most significant attributes by looking at their P values (i.e., P values ≤ 0.05). The P values indicate how confident we can be that a feature has some correlation with the web QoE metrics.

Figure 9 demonstrates the significance of these context features on web performance, by focusing on SI as our web QoE metric. We capture the significance of each variable in the final model by the color gradient of each cell. We first produce a model using only the samples we collect under mobility scenarios ("Mobility" model, label on the x axis) and another using the samples that we collect from stationary nodes which are always connected to the same network and do not experience any inter-RAT handovers ("Stationary" model, label on the x axis).

For the Mobility model, we note the strong significance of the initial RAT, inter-RAT handovers and the distance traveled, confirming our observations in Section 5. This confirms the previous work that highlights the strong impact of mobility on the end-user QoE [15]. Furthermore, the



(a) Over all sites (b) SI *Free-power-point-template* (c) SI *Youtube*

Figure 8: Some example ECDFs for SI using QUIC and H2 with Chrome.

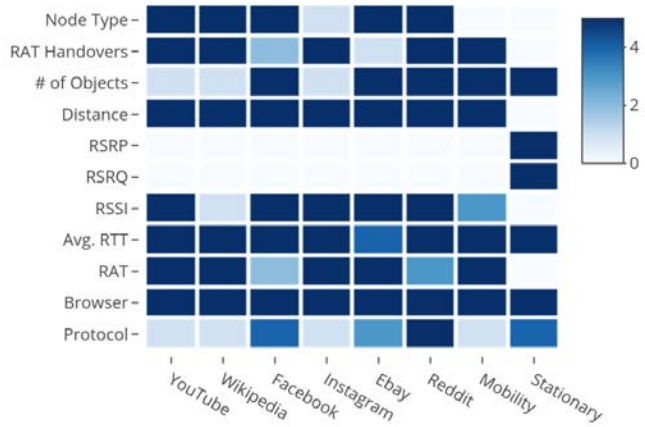


Figure 9: Significance of features for SI captured by a linear modelling approach.

number of objects, RTT and browser type also have a strong significance. At the same time, the signal strength value (RSSI) is not as significant as the mobility context, and the protocols is the least significant from the set of features we consider. In the Stationary model, we note that the features we consider including the number of objects for the target webpage, RSRP, RSRQ, browser type and even the protocol are significantly correlated with the end-user experience. These results are consistent with our findings in Section 5, except for the protocols. Regarding the protocols, although in Section 6 we concluded that there are no significant differences among protocols (they do not provide gains more than 10%), our model still considers protocol version as a significant feature indicating that the model tries to capture even marginal gains.

Next, we generate separate models for six of the target websites in Sweden (namely, *Youtube*, *Wikipedia*, *Facebook*, *Instagram*, *Ebay*, *Reddit*), by merging all the data from mobility/stationary conditions, operators, protocols or browser type. We break down our analysis per target webpage and note in Figure 9 the significance each feature has on the SI. Overall, we observe that the browser type and the mobility context (more specifically, the distance travelled during browsing) are consistently influencing the end-user experience, regardless the target webpage. For a given webpage, the number of downloaded objects is not significant, except for webpages that are potentially more dynamic over time (such as *Facebook* or *Ebay*). In all cases the signal strength value is highly correlated with the end-user experience, except for *Wikipedia*, which is potentially due to the smaller size of the page (which contains mostly text).

Takeaway: We observe that web performance under mobility is heavily impacted by RAT and inter-RAT handovers. For the stationary case, on the other hand, parameters such as browser, number of objects downloaded in a webpage and also signal strength values play an important role for the web performance. Overall, we observe that the browser type and the mobility context are consistently influencing the end-user experience, regardless the target webpage. The number of objects downloaded is not as significant for individual webpages, except for webpages that are potentially more dynamic over time.

8 CONCLUSION

This paper presented a cross-European study of web performance on commercial mobile carriers using the MONROE open measurement infrastructure. The novelty of the study stands in the sheer volume of data we were able to collect, nearly two million page visits from 11 different operational MBB networks in 4 countries, and in the rich context information captured along with the measurements that allowed us to analyze the complex interactions that influence mobile web user experience. Our results show a performance penalty of over 30% for mobile broadband access as compared to our wired baseline, with further performance drops for mobile users due to inter-technology handovers and varying channel conditions. Regrettably, except for few individual webpages, our measurements show that the performance improvements promised by newly emerging web protocols H2 and QUIC still remain to be experienced. Majority of back-to-back visits with H1s and H2 web protocols show similar performance with equal split of gains by H1s or H2 for the rest. Our results were confirmed by data analytics capturing the impact of single features that characterize the context in which browsing occurs. The presented measurements were designed to be scientifically sound and with reproducibility in mind and we offer the captured dataset to the community for other researchers to verify and build upon.

9 ACKNOWLEDGEMENTS

The authors appreciate the valuable comments provided by the anonymous reviewers. This work has been supported by the European Union H2020-ICT grant 644399 (MONROE). Additionally, the work of M. Rajiullah, A. Brunstrom and S. Alfredsson has also been partially supported by Region Värmland via the DigitalWell Arena project (grant RV2018-678). The work of M. Mellia has been supported by the SmartData@PoliTO center. The work of V. Mancuso has been supported by the Spanish Ministry of Economy and Competitiveness (grant RYC-2014-16285). Finally, the work of Ö. Alay has also been supported partially by the Norwegian Research Council project No. 250679 (MEMBRANE).

REFERENCES

[1] [n. d.]. A Primer for Web Performance Timing APIs. https://w3c.github.io/perf-timing-primer/. [Online; accessed 20-Feb-2019].

[2] [n. d.]. Above-the-fold time (AFT): Useful, but not yet a substitute for user-centric analysis. https://goo.gl/gWmj3r. [Online; accessed 20-Feb-2019].

[3] [n. d.]. Alexa: The top 500 sites on the web. https://www.alexa.com/topsites. [Online; accessed 20-Feb-2019].

[4] [n. d.]. Different SpeedIndex Measurements. https://bit.ly/2I87znK. [Online; accessed 20-Feb-2019].

[5] [n. d.]. MONROE-Browsertime. https://bit.ly/2SejKPo. [Online; accessed 20-Feb-2019].

[6] [n. d.]. A simple, client-side approach to measuring Speed Index. https://deanhume.com/a-simple-client-side-approach-to-measuring-speed-index/. [Online; accessed 20-Feb-2019].

[7] [n. d.]. sitespeedio/browsertime: Your browser, your page, your scripts! https://github.com/sitespeedio/browsertime. [Online; accessed 20-Feb-2019].

[8] [n. d.]. Speed Index - how it works and what it means. https://www.nccgroup.trust/uk/about-us/newsroom-and-events/blogs/2015/june/speed-index--how-it-works-and-what-it-means/. [Online; accessed 07-Aug-2018].

[9] [n. d.]. SpeedIndex measurements from the field. https://github.com/WPO-Foundation/RUM-SpeedIndex. [Online; accessed 20-Feb-2019].

[10] [n. d.]. Webpage Test. https://www.webpagetest.org. [Online; accessed 20-Feb-2019].

[11] [n. d.]. WebPagetest documentation. https://sites.google.com/a/webpagetest.org/docs/using-webpagetest/metrics/speed-index. [Online; accessed 20-Feb-2019].

[12] [n. d.]. Xvfb manul page. https://goo.gl/EPBtpt. [Online; accessed 20-Feb-2019].

[13] 2014. Can SPDY really make the web faster?. In *IFIP Networking Conference*. 1–9. https://doi.org/10.1109/IFIPNetworking.2014.6857089

[14] Özgü Alay, Andra Lutu, Miguel Peon Quiros, Vincenzo Mancuso, thomas hirsch, Kristian Evensen, Audun Fosselie Hansen, Stefan Alfredsson, jonas karlsson, Anna Brunström, ali safari Khatouni, M. Mellia, and Marco Ajmone Marsan. 2017. Experience: An Open Platform for Experimentation with Commercial Mobile Broadband Networks. *Proc. ACM MobiCom '17* (2017), 70–78. https://doi.org/10.1145/3117811.3117812

[15] Athula Balachandran, Vaneet Aggarwal, Emir Halepovic, Jeffrey Pang, Srinivasan Seshan, Shobha Venkataraman, and He Yan. 2014. Modeling Web Quality-of-experience on Cellular Networks. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking (MobiCom '14)*. ACM, New York, NY, USA, 213–224. https://doi.org/10.1145/2639108.2639137

[16] Dziugas Baltrunas, Ahmed Elmokashfi, and Amund Kvalbein. 2014. Measuring the Reliability of Mobile Broadband Networks. In *Proceedings of the 2014 Conference on Internet Measurement Conference (IMC '14)*. ACM, New York, NY, USA, 45–58. https://doi.org/10.1145/2663716.2663725

[17] Dziugas Baltrunas, Ahmed Elmokashfi, Amund Kvalbein, and Özgü Alay. 2016. Investigating packet loss in mobile broadband networks under mobility. In *2016 IFIP Networking Conference (IFIP Networking) and Workshops*. 225–233. https://doi.org/10.1109/IFIPNetworking.2016.7497225

[18] M. Belshe, R. Peon, and M. Thomson. 2015. Hypertext Transfer Protocol Version 2 (HTTP/2). RFC 7540 (Proposed Standard).

[19] Prasenjeet Biswal and Omprakash Gnawali. 2016. Does QUIC Make the Web Faster?. In *2016 IEEE Global Communications Conference (GLOBECOM)*. 1–6. https://doi.org/10.1109/GLOCOM.2016.7841749

[20] Enrico Bocchi, Luca De Cicco, Marco Mellia, and Dario Rossi. 2017. The Web, the Users, and the MOS: Influence of HTTP/2 on User Experience. In *Passive and Active Measurement - 18th International Conference, PAM*. 47–59. https://doi.org/10.1007/978-3-319-54328-4_4

[21] Enrico Bocchi, Luca De Cicco, and Dario Rossi. 2016. Measuring the Quality of Experience of Web users. *ACM SIGCOMM Computer Communication Review* 46, 4 (2016), 8–13.

[22] Enrico Bocchi, Ali Safari Khatouni, Stefano Traverso, Alessandro Finamore, Maurizio Munafo, Marco Mellia, and Dario Rossi. 2016. Statistical network monitoring: Methodology and application to carrier-grade NAT. *Computer Networks* 107 (2016), 20 – 35. https://doi.org/10.1016/j.comnet.2016.06.018 Machine learning, data mining and Big Data frameworks for network monitoring and troubleshooting.

[23] Gaetano Carlucci, Luca De Cicco, and Saverio Mascolo. 2015. HTTP over UDP: An Experimental Investigation of QUIC. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing (SAC '15)*. ACM, New York, NY, USA, 609–614. https://doi.org/10.1145/2695664.2695706

[24] Mallesham Dasari, Santiago Vargas, Arani Bhattacharya, Aruna Balasubramanian, Samir R. Das, and Michael Ferdman. 2018. Impact of Device Performance on Mobile Internet QoE. In *Proceedings of the Internet Measurement Conference 2018 (IMC '18)*. ACM, New York, NY, USA, 1–7. https://doi.org/10.1145/3278532.3278533

[25] H. de Saxcé, I. Oprescu, and Y. Chen. 2015. Is HTTP/2 really faster than HTTP/1.1?. In *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 293–299. https://doi.org/10.1109/INFCOMW.2015.7179400

[26] Jeffrey Erman, Vijay Gopalakrishnan, Rittwik Jana, and Kadangode K Ramakrishnan. 2015. Towards a SPDY'er mobile web? *IEEE/ACM Transactions on Networking* 23, 6 (Dec 2015), 2010–2023. https://doi.org/10.1109/TNET.2015.2462737

[27] R. Fielding et al. 1999. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard).

[28] Qingzhu Gao, Prasenjit Dey, and Parvez Ahammad. 2017. Perceived Performance of Top Retail Webpages In the Wild: Insights from Large-scale Crowdsourcing of Above-the-Fold QoE. In *Proceedings of the Workshop on QoE-based Analysis and Management of Data Communication Networks (Internet QoE '17)*. ACM, New York, NY, USA, 13–18. https://doi.org/10.1145/3098603.3098606

[29] Utkarsh Goel, Moritz Steiner, Mike P Wittie, Martin Flack, and Stephen Ludin. [n. d.]. Measuring What is Not Ours: A Tale of 3rd Party Performance. In *ACM Passive and Active Measurements Conference (PAM)*. Springer, 142–155.

[30] Utkarsh Goel, Moritz Steiner, Mike P. Wittie, Martin Flack, and Stephen Ludin. 2016. HTTP/2 Performance in Cellular Networks: Poster. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking (MobiCom '16)*. ACM, New York, NY, USA, 433–434. https://doi.org/10.1145/2973750.2985264

[31] Emir Halepovic, Jeffrey Pang, and Oliver Spatscheck. 2012. Can You GET Me Now?: Estimating the Time-to-first-byte of HTTP Transactions with Passive Measurements. In *Proceedings of the 2012 Internet Measurement Conference (IMC '12)*. ACM, New York, NY, USA, 115–122. https://doi.org/10.1145/2398776.2398789

[32] Junxian Huang, Feng Qian, Yihua Guo, Yuanyuan Zhou, Qiang Xu, Z. Morley Mao, Subhabrata Sen, and Oliver Spatscheck. 2013. An In-depth Study of LTE: Effect of Network Protocol and Application Behavior on Performance. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM (SIGCOMM '13)*. ACM, New York, NY, USA, 363–374. https://doi.org/10.1145/2486001.2486006

[33] Google Inc. [n. d.]. SPDY: An experimental protocol for a faster web. https://www.chromium.org/spdy/spdy-whitepaper. [Online; accessed 20-Feb-2019].

[34] Arash Molavi Kakhki, Samuel Jero, David Choffnes, Cristina Nita-Rotaru, and Alan Mislove. 2017. Taking a Long Look at QUIC: An Approach for Rigorous Evaluation of Rapidly Evolving Transport Protocols. In *Proceedings of the 2017 Internet Measurement Conference (IMC '17)*. ACM, New York, NY, USA, 290–303. https://doi.org/10.1145/3131365.3131368

[35] Conor Kelton, Jihoon Ryoo, Aruna Balasubramanian, and Samir R Das. 2017. Improving User Perceived Page Load Times Using Gaze. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. USENIX Association, Boston, MA, 545–559. https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/kelton

[36] Ali Safari Khatouni, Marco Mellia, Marco Ajmone Marsan, Stefan Alfredsson, Jonas Karlsson, Anna Brunstrom, Ozgu Alay, Andra Lutu, Cise Midoglu, and Vincenzo Mancuso. 2017. Speedtest-like Measurements in 3G/4G Networks: The MONROE Experience. In *Teletraffic Congress (ITC 29), 2017 29th International*, Vol. 1. IEEE, 169–177.

[37] H. Kim, J. Lee, I. Park, H. Kim, D. Yi, and T. Hur. 2015. The upcoming new standard HTTP/2 and its impact on multi-domain websites. In *2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. 530–533. https://doi.org/10.1109/APNOMS.2015.7275406

[38] Zahir Koradia, Goutham Mannava, Aravindh Raman, Gaurav Aggarwal, Vinay Ribeiro, Aaditeshwar Seth, Sebastian Ardon, Anirban Mahanti, and Sipat Triukose. 2013. First Impressions on the State of Cellular Data Connectivity in India. In *Proceedings of the 4th Annual Symposium on Computing for Development (ACM DEV-4 '13)*. ACM, New York, NY, USA, Article 3, 10 pages. https://doi.org/10.1145/2537052.2537064

[39] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, Jeff Bailey, Jeremy Dorfman, Jim Roskind, Joanna Kulik, Patrik Westin, Raman Tenneti, Robbie Shade, Ryan Hamilton, Victor Vasiliev, Wan-Teh Chang, and Zhongyi Shi. 2017. The QUIC Transport Protocol: Design and Internet-Scale Deployment. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '17)*. ACM, New York, NY, USA, 183–196. https://doi.org/10.1145/3098822.3098842

[40] Y. Liu, Y. Ma, X. Liu, and G. Huang. 2016. Can HTTP/2 Really Help Web Performance on Smartphones?. In *2016 IEEE International Conference on Services Computing (SCC)*. 219–226. https://doi.org/10.1109/SCC.2016.36

[41] Patrick Meenan. 2013. How Fast is Your Web Site? *Queue* 11, 2, Article 60 (March 2013), 11 pages. https://doi.org/10.1145/2436696.2446236

[42] Péter Megyesi, Zsolt Krämer, and Sándor Molnár. 2016. How quick is QUIC?. In *2016 IEEE International Conference on Communications (ICC)*. 1–6. https://doi.org/10.1109/ICC.2016.7510788

[43] Javad Nejati and Aruna Balasubramanian. 2016. An In-depth Study of Mobile Browser Performance. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1305–1315. https://doi.org/10.1145/2872427.2883014

[44] Ashkan Nikravesh, Hongyi Yao, Shichang Xu, David Choffnes, and Z Morley Mao. 2015. Mobilyzer: An open platform for controllable mobile network measurements. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 389–404.

[45] N. Oda and S. Yamaguchi. 2018. HTTP/2 performance evaluation with latency and packet losses. In *2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC)*. 1–2. https://doi.org/10.1109/CCNC.2018.8319285

[46] Mohammad Rajiullah. 2015. *Towards a Low Latency Internet: Understanding and Solutions*. Ph.D. Dissertation. Karlstad University, Department of Mathematics and Computer Science.

[47] Mohammad Rajiullah, AC Mohideen, Felix Weinrank, Raffaello Secchi, Gorry Fairhurst, and Anna Brunstrom. 2017. Understanding multistreaming for web traffic: An experimental study. In *2017 IFIP Networking Conference (IFIP Networking) and Workshops*. 1–6. https://doi.org/10.23919/IFIPNetworking.2017.8264875

[48] E. Rescorla. 2000. HTTP Over TLS. RFC 2818 (Informational). http://www.ietf.org/rfc/rfc2818.txt Updated by RFCs 5785, 7230.

[49] John P. Rula, Vishnu Navda, Fabián E. Bustamante, Ranjita Bhagwan, and Saikat Guha. 2014. No "One-size Fits All": Towards a Principled Approach for Incentives in Mobile Crowdsourcing. In *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications (HotMobile '14)*. ACM, New York, NY, USA, Article 3, 5 pages. https://doi.org/10.1145/2565585.2565603

[50] Sayandeep Sen, Jongwon Yoon, Joshua Hare, Justin Ormont, and Suman Banerjee. 2011. Can They Hear Me Now?: A Case for a Client-assisted Approach to Monitoring Wide-area Wireless Networks. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference (IMC '11)*. ACM, New York, NY, USA, 99–116. https://doi.org/10.1145/2068816.2068827

[51] Muhammad Zubair Shafiq, Lusheng Ji, Alex X. Liu, Jeffrey Pang, Shobha Venkataraman, and Jia Wang. 2013. A First Look at Cellular Network Performance During Crowded Events. In *Proceedings of the ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '13)*. ACM, New York, NY, USA, 17–28. https://doi.org/10.1145/2465529.2465754

[52] M. Zubair Shafiq, Lusheng Ji, Alex X. Liu, and Jia Wang. 2011. Characterizing and Modeling Internet Traffic Dynamics of Cellular Devices. In *Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '11)*. ACM, New York, NY, USA, 305–316. https://doi.org/10.1145/1993744.1993776

[53] Sierra-Wireless. [n. d.]. MC7455 miniPCI express (USB 3.0) modem: https://www.sierrawireless.com/products-and-solutions/embedded-solutions/products/mc7455/. ([n. d.]).

[54] Joel Sommers and Paul Barford. 2012. Cell vs. WiFi: On the Performance of Metro Area Mobile Connections. In *Proceedings of the 2012 Internet Measurement Conference (IMC '12)*. ACM, New York, NY, USA, 301–314. https://doi.org/10.1145/2398776.2398808

[55] Steve Souders. [n. d.]. Moving beyond window.onload(). https://goo.gl/rAeFNR. [Online; accessed 20-Feb-2019].

[56] Daniel Stenberg. 2014. HTTP2 Explained. *SIGCOMM Computer Communication Review* 44, 3 (July 2014), 120–128. https://doi.org/10.1145/2656877.2656896

[57] Matteo Varvello, Kyle Schomp, David Naylor, Jeremy Blackburn, Alessandro Finamore, and Kostantina Papagiannaki. 2015. To HTTP/2, or not to HTTP/2, that is the question. *arXiv preprint arXiv:1507.06562* (2015).

[58] Matteo Varvello, Kyle Schomp, David Naylor, Jeremy Blackburn, Alessandro Finamore, and Konstantina Papagiannaki. 2016. Is the Web HTTP/2 Yet?. In *Passive and Active Measurement - 17th International Conference, PAM*. 218–232. https://doi.org/10.1007/978-3-319-30505-9_17

[59] J Vihervaara and T Alapaholuoma. 2018. The impact of HTTP/2 on the service efficiency of e-commerce websites. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE.

[60] Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall. 2014. How Speedy is SPDY?. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*. USENIX Association, Seattle, WA, 387–399. https://www.usenix.org/conference/nsdi14/technical-sessions/wang

[61] Greg White, Jean-François Mulé, and Dan Rice. 2012. Analysis of Google SPDY and TCP initcwnd. *Cable Television Laboratories, Inc* (2012).

[62] Maarten Wijnants, Robin Marx, Peter Quax, and Wim Lamotte. 2018. HTTP/2 Prioritization and Its Impact on Web Performance. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1755–1764. https://doi.org/10.1145/3178876.3186181

[63] Kyriakos Zarifis, Mark Holland, Manish Jain, Ethan Katz-Bassett, and Ramesh Govindan. 2016. *Modeling HTTP/2 Speed from HTTP/1 Traces*. Springer International Publishing, Cham, 233–247. https://doi.org/10.1007/978-3-319-30505-9_18

[64] Torsten Zimmermann, Jan Rüth, Benedikt Wolters, and Oliver Hohlfeld. 2017. How HTTP/2 Pushes the Web: An Empirical Study of HTTP/2 Server Push. In *2017 IFIP Networking Conference (IFIP Networking) and Workshops*.

A Dynamic Embedding Model of the Media Landscape

Jérémie Rappaz*
EPFL
Switzerland
jeremie.rappaz@epfl.ch

Dylan Bourgeois*
EPFL
Switzerland
contact@dtsbourg.me

Karl Aberer
EPFL
Switzerland
karl.aberer@epfl.ch

ABSTRACT

Information about world events is disseminated through a wide variety of news channels, each with specific considerations in the choice of their reporting. Although the multiplicity of these outlets should ensure a variety of viewpoints, recent reports suggest that the rising concentration of media ownership may void this assumption. This observation motivates the study of the impact of ownership on the global media landscape and its influence on the coverage the actual viewer receives. To this end, the selection of reported events has been shown to be informative about the high-level structure of the news ecosystem. However, existing methods only provide a static view into an inherently dynamic system, providing underperforming statistical models and hindering our understanding of the media landscape as a whole.

In this work, we present a dynamic embedding method that learns to capture the decision process of individual news sources in their selection of reported events while also enabling the systematic detection of large-scale transformations in the media landscape over prolonged periods of time. In an experiment covering over 580M real-world event mentions, we show our approach to outperform static embedding methods in predictive terms. We demonstrate the potential of the method for news monitoring applications and investigative journalism by shedding light on important changes in programming induced by mergers and acquisitions, policy changes, or network-wide content diffusion. These findings offer evidence of strong content convergence trends inside large broadcasting groups, influencing the news ecosystem in a time of increasing media ownership concentration.

CCS CONCEPTS

• **Information systems** → **Data mining**; • **Computing methodologies** → **Learning latent representations**.

KEYWORDS

media pluralism; factorization methods; ranking methods

ACM Reference Format:
Jérémie Rappaz, Dylan Bourgeois, and Karl Aberer. 2019. A Dynamic Embedding Model of the Media Landscape. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313526

*Both authors contributed equally to the paper

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313526

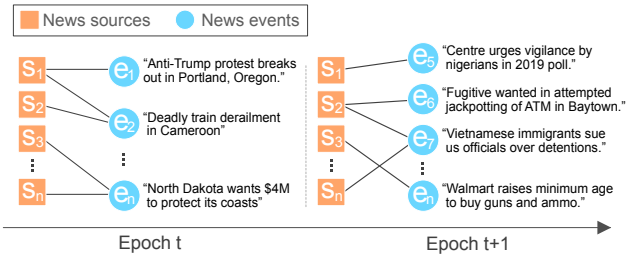


Figure 1: Illustration of the setting: we observe the coverage of news events from a fixed set of sources over several time epochs. Example events are extracted from the GDELT database.

1 INTRODUCTION

Information about world events is reported as a continuous stream, processed then broadcast to an audience through a large number of channels and across various mediums. As no coverage could claim to be exhaustive, sources must first filter the stories that will be disseminated. The selection made by each channel is partial by nature. However, the variety of outlets, each with a wide array of considerations in the choice of its reporting, is assumed to ensure the diversity of news to which the reader is exposed to. This principle is often referred to as the *external pluralism* assumption. It should ensure heterogeneity in the media space, encapsulating anything from the diversity of ownership to the independence of the editorial board. [1] *External pluralism* is also known as the "supplier" pluralism, since it should exclude the possibility of large broadcast groups exerting influence on downstream reporting. Yet in practice, this assumption does not always hold. News channels are often owned or operated by commercial, private entities, implying that the ecosystem as a whole is influenced by economically motivated forces, such as mergers, acquisitions, or regulatory actions. The increase in concentration of ownership has been observed to be the dominating force in the media landscape, as reported by the Pew Research Institute in a 2017 study on the acquisition of local television stations. [2]

While the literature, discussed in more detail in Section 2, still debates the causal effects of market ownership structures on the diversity of offerings, the Federal Communications Commission (FCC) [3] has defended the idea that "there is a positive correlation between viewpoints expressed and ownership of an outlet". [4] This

[1] As opposed to *internal pluralism*, where sources are assumed to present a wide variety of ideological viewpoints, communicated through different mediums. [8]
[2] http://www.pewresearch.org/fact-tank/2017/05/11/buying-spree-brings-more-local-tv-stations-to-fewer-big-companies/
[3] The FCC is the regulating body for multimedia communications in the United States.
[4] FCC, Biennial Media Ownership Order (2003)

uncertainty strongly motivates the development of novel and interpretable methods, which would allow the observation of large-scale movements in the media landscape and allow correlating the observations with real-world factors.

Recent studies have tackled the problem of evaluating similarities across news sources [5, 38] and have found that the analysis of coverage patterns can highlight non-obvious relationships between sources, such as the aforementioned corporate ownership structures. However, they have only captured a static snapshot of the media landscape. This hinders the ability to observe changes in content diffusion, and limits attempts to correlate the observations with potential factors in the news space. These models' predictive power is also limited by their inability to incorporate temporal dependencies. In other words, it restricts them to an informative but partial and static view of the news ecosystem. Indeed, the model of any dynamic system requires a time-dependent formulation, necessary to detect variations and measure trends. Existing approaches are blind to such transient effects.

Studying the effect of real-world factors on content diffusion is a challenging task. It requires the detection of subtle variations in the coverage patterns of individual channels, which can occur over long time spans. Any attempt to characterize these effects would suffer from the absence of a ground truth, prompting the need for a comparison across news channels. While similarities between channels are straightforward to compute for short, consistent periods of time, the continuous occurrence of new and unpredictable events breaks the ability to reason across successive snapshots of the news. A successful estimate of similarity over time should hence maintain temporal consistency of pairwise distances across news channels. At any given point in time, it should ideally do so without compromising its predictive accuracy compared to static methods. These challenges and constraints motivate the use of the models presented here.

In this work, we propose a dynamic embedding model of the media landscape. By predicting news sources' coverage, the model captures their similarities throughout the observed period. The embedding space is maintained consistent over time by augmenting the model with a knowledge of previous time steps, and by adding a temporal regularization on the model's parameters. This improves the model's predictive capabilities and provides a temporally coherent source-wise similarity metric, allowing the visualization and analysis of long-ranging fluctuations in the news ecosystem. We also propose a systematic method to detect abrupt transition patterns in this similarity space. This enables the analysis of the news ecosystem beyond domain-specific knowledge and hand-crafted analysis.

In the context of news reporting, temporally-aware models provide a powerful tool for correlating content programming changes and real-world factors, from individual news outlet's behaviors to large-scale dynamics. Shining a light on these movements would naturally guide the search for source-level context, helping to characterize and identify the observation's driving forces. This information could then provide context to journalists investigating the coverage of a particular story or feed watchdog-like processes that monitor the health and evolution of the media landscape. To illustrate, we provide some prototypical questions which could arise in a journalistic probe, and that could be elucidated by information derived from the proposed model:

- What effect does the ownership of a news channel have on its content diffusion?
- Which sources are most similar (resp. dissimilar) to a sample source, and how has this similarity evolved over time?
- Which are the most consistent (or varying) news channels in terms of broadcast content?
- Which broadcast groups exert a large influence on the content of their respective channels?

The rest of this paper is organized as follows. In Section 2, we present relevant work from the literature. In Section 3, we describe the dataset used in our study. In Section 4, we introduce our model. In Section 5, we discuss the predictive performance of the model. In Section 6, we illustrate the use of our model in the identification of abrupt changes the media landscape, attempting to interpret these changes in coverage in Section 7. In Section 8, we conclude and propose future research directions.

2 RELATED WORK

The study of the evolution of the media landscape has led to prolific research lines spanning various fields. Early work had a tendency to examine the media ecosystem in isolation. For example Steiner's seminal work [43] studies the interplay of consumer preferences and in-market competition on the diversity of radio broadcasting. This study omitted the role of external driving forces, which were only later modeled by Anderson & Coate. [3] They predicted that media consolidation, while economically beneficial for the market, would reduce competition and hence diversity for the viewer. This insight was later formulated in terms of ideological bias by Gentzkow & Shapiro [10] in the case of newspapers.

The tendency to integrate external driving factors has picked up steam in recent years, most notably with a theory of convergence in the media ecosystem. This convergence expresses itself through two seemingly contradictory features. On one hand, information is being delivered through an ever-increasing number of channels and means of diffusion. On the other hand, media ownership concentration has seen an upwards trend, with a large proportion of channels being owned by only a handful of media conglomerates. [31] This dichotomy has been studied by, among others, Jenkins. [20] The author proposed a sketch of the phenomenon that looks further than the sole technological influence, reaching for larger cultural factors. Vizcarrondo et al. [45] have more specifically investigated the concentration of media ownership. They reported on changes in the diversity of ownership within the media industry covering the 1976 through 2009 time period.

A large body of work has also been introduced to study the effect of an ideologically slanted press. For instance in a large-scale observational study, DellaVigna et al. [6] measured the effect of the introduction of a conservative-oriented channel (Fox News) led to gains of 0.4 to 0.6 percentage points in Republican voting in the towns where the channel was being broadcast. While specific to a particular orientation, this work is in line with studies showing the profound influence of the media in voters' political awareness [27] and their participation in the electoral process. [11, 12, 28]

Notation	Description
R	Interaction matrix $\in \mathbb{R}^{\lvert S \rvert \times \lvert E \rvert}$
S / E	News source / Event set
$E^+_{s_i}$	Set of events covered by source s_i
s_i	Source $s_i \in S$
e_k	Event $e_k \in E$
K	Number of latent factors
$\hat{x}_{s_i,e_k}$	Predicted preference of source s_i for event e_k
D / D_s	Training / Testing set
γ	STD of the Gaussian random walk
α	Learning rate
λ_Θ	ℓ_2-regularization weight
λ_T	Temporal regularization weight
Θ	Set of model parameters

Table 1: Notation

The Federal Communications Commission (FCC) regularly issues studies regarding the state of the news ecosystem. Specifically, some of these studies focus on the effect of ownership on local news stations' content programming behaviors. However, by the authors' own admission [5], these works often lack the breadth required by a large-scale empirical study. For example, Pritchard [32] conducted a study of the diversity of coverage for cross-owned media outlets during the 2000 presidential campaign but on a sample of only 10 newspapers. Groseclose & Milyo [16] proposed a measure of media bias which was evaluated on a set of 8 newspapers. Djankov et al. [7] did survey the news ecosystem on large scale, building a map of media ownership in 97 countries around the world, but this work dates back to 2003.

Dimensionality reduction methods have been extensively used to model different perspectives or opinions. For example, Lahoti et al. [24] rely on Non-Negative Matrix Factorization methods to learn a liberal-conservative ideology space on Twitter. In particular, the authors propose to approximate the users' ideology based on their online news consumption.

The study performed by Saez-Trumper et al. [39] relies on a Principal Component Analysis (PCA) to detect similarities across news channels. In particular, they performed their analysis by covering three different types of biases in the press: *gatekeeping bias*, *coverage bias* and *statement bias*. The authors' bias-capturing method does present similarities to our approach, but is formulated as an unsupervised approach and does not tackle temporal variability. Recently, Bourgeois et al. [5] proposed a supervised, embedding-based method to capture similarities across news sources based on their respective coverage. This model was also designed statically and is hence unable to accurately model temporal variations in news coverage.

Learning to produce a personalized ranking from positive interactions only is a well-studied problem that has been investigated in the context of recommendations from implicit feedback [30] and as One-Class feedback [19] in the context of recommender systems.

[5] "[a] larger number of independent owners will tend to generate a wider array of viewpoints in the media than would a comparatively smaller number of owners. We believe this proposition, even without the benefit of conclusive empirical evidence." FCC, Biennial Media Ownership Order (2003)

Temporally-aware methods have received increasing attention and many previous models have now been adapted to the temporal setting. The *dynamic embedding*, proposed by Rudolph et al. [36] as a variation of traditional embedding methods, is generally aimed toward temporal consistency. The method is introduced in the context of word embeddings, which are used to characterize the evolution of English language. The model is built upon the initial *exponential family embeddings* model. [37]

The field of personalization has many examples of temporally-aware models since human preferences tend to evolve over time. For example, influential work from Koren et al. [22] models the changing nature of preference through a linear drifting term. Another approach relies on the use of Tensor Factorization (TF), [1, 9, 46] in which the extra dimension models temporal patterns in the data. We do not consider TF-based methods as valid candidate approaches since we focus on the problem of grounding representations over time by penalizing unnecessary differences between successive solutions of the model. The temporal modelling capabilities of TF-based methods would predict the evolution of sources and introduce additional temporal variations, consequently degrading interpretability.

He et al. [18] introduced a temporally-aware model of a recommender system in order to capture the evolution of fashion trends. Similar to Koren et al. [22], the authors also proposed the addition a drifting term to the model. The authors later proposed the use of a higher-order Markov chain that captures both short- and long-term dynamics [17]. Note that both models make use of Bayesian Personalized Ranking (BPR [35]) for their respective optimization procedure. In the context of networks, Yu et al. [48] proposed a temporal factorization for analyzing the evolution of network structures.

The media response to global events has been studied, with many using the same data-source as our work. However, these approaches have focused mainly on content-level analysis, leveraging the multimodal capabilities of the GDELT dataset (please refer to Section 3 for more detailed information about the dataset). These works have monitored the media response to events such as protests, [33] natural disasters [23] or conflicts. [14, 21, 47] This data-source was also used to get a world-wide view on coverage of global issues like climate change. [29]

Research Questions: Given the work above, several research questions are of our interest and have remained unanswered:

RQ1: How can we model the global media landscape over time?
RQ2: Is media consolidation highlighted by the resulting latent representation?
RQ3: How to systematically detect abrupt deviations in content diffusion at a source-level?

3 DATA

In this section, we present the selected data source and explicit our data collection process, providing general statistics about the resulting dataset.

	GDelt Dataset
# sources	7 278
# unique events	174M
# interactions	588M
time span	Feb 2015 - May 2018

Table 2: Meta-data about the dataset (after preprocessing)

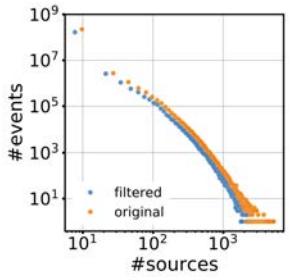


Figure 2: Number of sources covering an event.

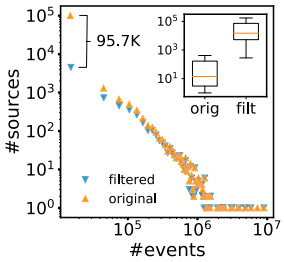


Figure 3: Number of events covered per source.

3.1 Data Source

Recently, several event collection databases have emerged on the Web, making accessible to the general public a global view of the daily world events. In this work, we rely on the *Global Database of Events, Language, and Tone* (GDELT [26]), a large database of annotated news events. GDELT [6] was selected since it is publicly available and covers a reasonably large timespan and geography, offering a larger set of sources [2] to study compared to alternatives such as EventRegistry. [7]

For decades, event coding was performed manually. In the 1990s, the first automated systems started to gain traction in the academic community, with initiatives such as the KEDS system. [40] Its successor was proposed in the form of *Text Analysis By Augmented Replacement Instructions* (TABARI), which is the engine that runs event coding for GDELT. This framework is designed to process large amounts of text to extract the presence of pairs of *actors* and *verbs*. To do so it matches elements from user-provided dictionaries, which contain a massive collection of event protagonists (i.e. actors) ranging from recognizable named entities (e.g. *Barack Obama*) to functional placeholders (e.g. *a local woman*). These actors are able to interact with the world through *verbs* (i.e. actions), which can be self-contained (e.g. *announces their intent to*) or involve a second actor (e.g. *criticizes their opponent*). Several standards exist for these dictionaries. GDELT uses the *Conflict and Mediation Event Observations* (CAMEO [13]). [8] Note that, as a remnant of previous hand-curated event annotation frameworks [25], TABARI also provides an interface for manual hand-off to domain experts if the sentences become too complex. This reinforces GDELT's ability to uniquely annotate even the most fine-grained events.

GDELT also augments every news event it tags by extracting meta-information about the article including, but not limited to, its location, its tone, its Goldstein Scale [15] and refences the URL the event was scanned at. It scours a wide array of sources, from television stations to blogs, news wires and papers. Thanks to the information provided by this augmented event coding framework, GDELT assigns, for each news event, a global identifier, which makes it possible to link the same event's coverage across different news sources. Beyond the rich annotations provided by GDELT, this tracking is central to our study given that we only work at the coverage level, without considering the content itself.

[6] https://www.gdeltproject.org/
[7] https://eventregistry.org/
[8] An exhaustive list of the considered categories can be found at http://data.gdeltproject.org/documentation/CAMEO.Manual.1.1b3.pdf

3.2 Data Preprocessing

From the massive resource maintained by GDELT we can gather a dataset of interactions between sources and events, recording which sources covered which uniquely identifiable events. We focus our data collection campaign on the publicly available dumps of GDELT 2.0, released every 15 minutes since February 2015. General statistics about the dataset are presented in Table 2 as well as Fig. 2 and Fig. 3.

When considering the full time-span, GDELT references more than 105K different news sources. This represents a considerable increase compared to the 63K sources reported by Kwak et al. [2] in 2016. However, as is shown in Fig. 3, most of the sources have only published a few articles over the relevant stretch. To maintain a consistent number of channels over time, we discard all channels inactive in any one of the time slices from our dataset. This retains around 7 278 news sources in our dataset. The filtering step does remove a large fraction of available channels, but it mostly affects sources with a very low publishing rate: despite preserving only 7% of the total channels, the selection still accounts for more than 76% of interactions in the dataset (see Fig. 3).

The dataset is split into slices with a duration of one month. This allows for a decent trade-off between having a significant amount of events covered in the training set, while also providing enough samples to observe time-dependent changes over the considered period. In principle, this scale could be modified to study the media landscape at different granularity levels. For example, a more fine-grained split might allow the observation of changes correlated with specific events. However, we choose to leave this analysis as future work given the significant amount of computation that the model requires.

4 MODEL

In this section, we first motivate our approach. Then, we briefly introduce the use of Matrix Factorization methods in a ranking setting by formally describing the task and its objectives. Finally, we propose two strategies to extend the model to a temporal setting.

4.1 Motivation

The main focus of our work is to establish a dynamic model of coverage similarities across news channels in order (i) to uncover congruent coverage patterns in the pool of available channels, (ii) to establish a dynamic, predictive model of coverage and (iii) to

provide a systematic methodology to identify and interpret structural changes in the media landscape. Recent political and societal focus on media accountability and transparency provide strong motivators in the development of such tools (see Section 1).

We model the interrelationships between sources and events by relying on a Matrix Factorization (MF) method. MF methods represent a natural way of projecting two disjoint sets of items in a common latent space of K dimensions in order to model their interactions. Such personalized methods are commonly use by recommender systems, which routinely aim to model retail purchasing decisions. Used in our context, they model coverage decisions instead. We cast the problem to a One-Class learning setting [30] since we observe positive interactions only. The One-class formulation avoids making assumptions about negative examples: we do not distinguish between real negatives (i.e. the source purposely didn't cover an event) and unobserved interactions (i.e. the source wasn't aware of the event).

4.2 Problem Statement

Let us consider a set of news sources S and a set of events E. Interactions between the two are represented by an interaction matrix $R \in \mathbb{R}^{|S|\times|E|}$. Observations take the form of dyadic interactions $(s_i, e_j), s_i \in S, e_j \in E$ which express source s_i's coverage of event e_j. Equivalently, we define in matrix form that $R_{i,j} = 1$ if source s_i has covered event e_j and $R_{i,j} = 0$ otherwise.

Predicting the unobserved entries of matrix R is achieved by taking the dot-product of two low rank matrices, such that $R \approx P^T \cdot Q$, where $P \in \mathbb{R}^{K\times|S|}$ and $Q \in \mathbb{R}^{K\times|E|}$ with $K << |S|, |E|$. Every source s_i (resp. every event e_j) is represented by a column in P (resp. Q). We will refer to these columns as an embedding vector throughout the remainder of this work. We will refer to Θ as the set of parameters for our MF predictor, such that $\Theta = \{P, Q\}$.

Objectives: The model is trained with the objective of predicting the likelihood of a source covering a particular event. The predicted likelihood $\hat{x}_{s_i,e_j}$ of source s_i covering event e_j is computed as the dot-product between the two respective embedding vectors,

$$\hat{x}_{s_i,e_j} = p_{s_i}^T \cdot q_{e_j} \,. \tag{1}$$

Instead of best approximating the reconstruction of matrix R, this objective is stated as a ranking problem in which positive examples should obtain a higher rank than negative ones, i.e. to predict a higher score for an event that has been covered than for random negative samples. Optimizing a MF model with a ranking criterion is equivalent to maximizing the following probability,

$$\begin{aligned} \Pr(e_j >_{s_i} e_k|\Theta) &:= H(\hat{x}_{s_i,e_j} - \hat{x}_{s_i,e_k}) \\ &\approx \sigma(\hat{x}_{s_i,e_j} - \hat{x}_{s_i,e_k}) \end{aligned} \tag{2}$$

where e_j is an event covered by source s_i and e_k is a randomly sampled negative event; formally, $e_j \in E_{s_i}^+$ and $e_k \in E \setminus E_{s_i}^+$. We adopt the notation $>_{s_i}$ to denote a source s_i preferring to cover e_j over e_k and model the observation of this preference using $H(\cdot)$, the Heaviside step function: $H(\cdot)$ is equal to $\mathbf{1.0}$ for positive inputs and to $\mathbf{0.0}$ otherwise. Therefore, $H(\hat{x}_{s_i,e_j} - \hat{x}_{s_i,e_k})$ would always be equal to $\mathbf{1.0}$ for an ideal predictor. In practice, $H(\cdot)$ is approximated by the differentiable logistic sigmoid $\sigma(\cdot)$.

Finally, we maximize BPR, our log-likelihood criterion

$$\text{BPR} := \sum_{(s_i,e_j,e_k)\in D} \ln \sigma(\hat{x}_{s_i,e_j} - \hat{x}_{s_i,e_k}) - \lambda_\Theta \, ||\Theta||^2 \,. \tag{3}$$

Note the inclusion of an ℓ_2-regularization term over the set of parameters Θ. We please refer the reader to the work of Rendle et al. [35] for more details about this optimization scheme.

4.3 Temporal Setting

In this section, we describe the adoption of the dynamic embedding scheme proposed by Rudolph et al. [36] in the context of news coverage modeling. In particular, we adopt two strategies to maintain temporal consistency across time slices, respectively based (i) on a Gaussian random walk (*RW*) and (ii) on the addition of a temporal regularization term (*RG*). We adopt the notation $p_{s_i}^{(t)}$ to denote the embedding vector of news source s_i at the t-th time step.

Prior on the embedding vectors: Without information about former time slices, existing methods typically initialize embedding vectors to small, randomly distributed values. However, such approaches do not take advantage of any prior knowledge acquired during anterior training steps. The addition of a prior on embedding vectors represents a simple, yet powerful strategy to leverage previously acquired knowledge about sources. In particular, embedding vectors at the t-th time step are initialized using a Gaussian random walk around their final values at time step $(t-1)$. The Gaussian random walk is expressed as follows:

$$p_{s_i}^{(t)} \sim \begin{cases} \mathcal{N}(0, \gamma^{-1} I), & \text{if t=0} \,. \\ \mathcal{N}(p_{s_i}^{(t-1)}, \gamma^{-1} I), & \text{otherwise} \,. \end{cases} \tag{4}$$

This initialization scheme ensures a smooth transition of the parameter set learned in two consecutive time slices. This yields a more stable embedding space, offering a coherent expression of divergence across time-steps. Since events are inherently much more volatile than sources, we initialized their embedding vectors at random at each new time slice.

Optimization using temporal regularization: The second part of the dynamic scheme takes the form of a temporal regularization term. The newly introduced term penalizes large variations across time steps by minimizing the distance of an embedding vector at the t-th step to its final value at step $(t-1)$.

The final log-likelihood criterion, BPR-T, can then be formulated as follows for the t-th time split

$$\begin{aligned} \text{BPR-T}^{(t)} := \sum_{(s_i,e_j,e_k)\in D} & \ln \sigma(\hat{x}_{s_i,e_j} - \hat{x}_{s_i,e_k}) \\ & - \lambda_\Theta \, ||\Theta||^2 \\ & - \underbrace{\lambda_T \|p_{s_i}^{(t)} - p_{s_i}^{(t-1)}\|^2}_{\text{temporal regularization}} \end{aligned} \tag{5}$$

The model is optimized using stochastic gradient ascent and is fitted once for every time split. Update steps are defined as follows:

$$
\begin{aligned}
q_{e_j}^{(t)} &\leftarrow q_{e_j}^{(t)} + \alpha(\sigma(-\hat{x}_{s_i,e_j,e_k}) \cdot \quad p_{s_i}^{(t)} \quad - \lambda_\Theta \, q_{e_j}^{(t)}) \\
q_{e_k}^{(t)} &\leftarrow q_{e_k}^{(t)} + \alpha(\sigma(-\hat{x}_{s_i,e_j,e_k}) \cdot (-p_{s_i}^{(t)}) - \lambda_\Theta \, q_{e_k}^{(t)}) \\
p_{s_i}^{(t)} &\leftarrow p_{s_i}^{(t)} + \alpha(\sigma(-\hat{x}_{s_i,e_j,e_k}) \cdot (q_{e_j}^{(t)} - q_{e_k}^{(t)}) \\
&\quad - \lambda_\Theta \, p_{s_i}^{(t)} \\
&\quad - \lambda_T (p_{s_i}^{(t)} - p_{s_i}^{(t-1)}))
\end{aligned}
\tag{6}
$$

where α is our learning rate. We use the notation $\hat{x}_{s_i,e_j,e_k}$ to denote the quantity $(\hat{x}_{s_i,e_j} - \hat{x}_{s_i,e_k})$. Note that triplets $(s_i, e_j, e_k), e_j \in E_{s_i}^+$ and $e_k \in E \setminus E_{s_i}^+$ forming the training dataset D are randomly sampled during the optimization. [9]

4.4 Evaluation

To assess the performance of the different methods, we adopt a *leave-one-out* methodology, in which a single event per source is withheld at random from the training set D to constitute the test set D_s. This approach ensures that all sources have similar weights in the evaluation. We adopt the widely used *Area Under the Curve* (AUC) as a measure of performance. In the context of this work, the evaluation procedure is formally defined as follows

$$
\text{AUC} = \frac{1}{|D_s|} \sum_{(s_i,e_j,e_k) \in D_s} H(\hat{x}_{s_i,e_j} - \hat{x}_{s_i,e_k}) \, . \tag{7}
$$

where e_j is an event covered by s_i and e_k is an event that s_i hasn't covered, randomly sampled at testing time. Negative samples are drawn uniformly at random across all unique event of the current time slice (we omitted the time indices for the sake of brevity).

4.5 Experimental Setting

The code for experiment and analysis will be made available at publication time under an open-source license. All experiment-related code was run on a 6-core machine, equipped with an `Intel(R) Xeon(R) CPU E5-2630 @ 2.30GHz`, for a total training time of approximately 3 days. [10] We restricted the tuning of hyper-parameter λ_T to a subset of values $\in \{0.001, 0.01, 0.1, 1.0\}$. All scores and figures are reported using $\lambda_T = 0.1$, which we found to provide the highest cross-validated accuracy. For computational reasons, the other parameters were coarse-tuned on a static snapshot and were set to $K = 20$ and $\lambda_\Theta = 0.1$, $\gamma = 0.01$ and $\alpha = 0.1$.

5 RESULTS

In this section, we compare the existing static embedding model with the dynamic method presented in Section 4. The effect of both the prior on the embedding vectors and the temporal regularization are then measured in isolation.

Table 3 summarizes the performances of the various approaches by taking the mean AUC scores obtained, for each month, over the considered period. *BPR* denotes the core of the algorithm without any temporal component. *RG* denotes the use of temporal regularization and *RW* denotes the use of a Gaussian random walk for embedding initialization. To compare with a non-parametrized approach, we also include *POP*, a popularity-based baseline: the score for a given event is a function of its frequency in the training set.

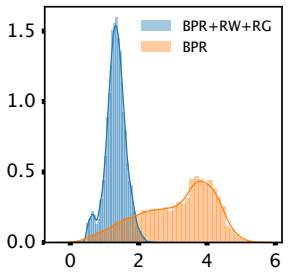


Figure 4: Average displacement of embeddings

	AUC
POP	0.6509
BPR	0.8959
BPR + RG	0.9089
BPR + RW	0.9318
BPR + RW + RG	**0.9337**

Table 3: Method contribution to performance

Several observations can be made in light of these results. Firstly, the combination of the two strategies *RG* and *RW* is shown to provide the best observed predictive performances. Secondly, the individual strategies do not provide the same performance improvements. Results suggest that a proper initialization contributes much more to an accurate prediction than strong temporal regularization. In parallel experiments, we even observed that an increase of λ_T decreases performances. We hypothesize that this is due to the model's inability to handle abrupt changes in source behavior, since a strong regularizer would penalize a large difference with respect to the previous time step.

The use of the proposed dynamic strategies (*RG* and *RW*) provides better overall consistency of the latent space across time slices. This is visible by measuring the average displacement of sources in embedding space. As shown in Fig. 4, sources are much more stable in the dynamic setting compared to the static embedding procedure. The added stability of the embedding space provides a usable expression of divergence across time-steps. This means source similarity can be coherently compared across the entire observed period, while also providing an overall improvement of the AUC scores (see Fig. 5).

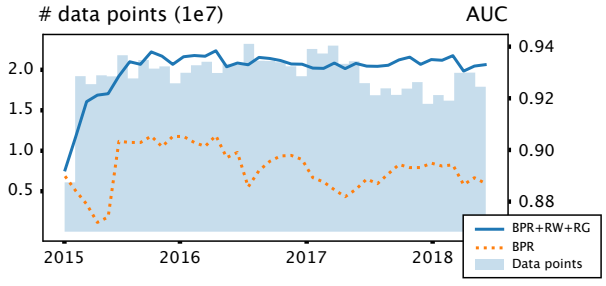


Figure 5: Performances (AUC) and dataset size per month

[9] We sampled both positive and negative examples uniformly. More complex sampling approaches exist [34] but are outside of the scope of this work.

[10] The training time is reported for the full 3-year period; a production-ready application would typically be optimized and use incremental updates instead.

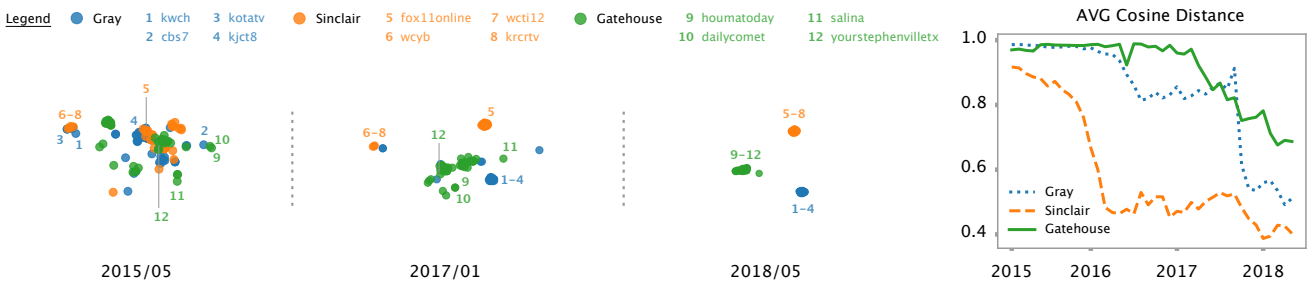


Figure 6: Case-study of the influence of ownership on the selection of covered events (best seen in color)
Left: Sample of the temporal evolution of the media landscape, learned with dynamic embeddings, illustrated with t-SNE. [44] To test the effect of ownership on diversity of coverage, we monitor the evolution of a set of sources all owned by the same broadcast group in 2018, backtracking their evolution in embedding space. Starting with an initial seed source known to be operated by - or affiliated to - one of three large American media conglomerates (Gray Television Inc., Sinclair Broadcasting Group and GateHouse Media), we build a set of 50 sources per group so as to have a comparably sized sample for each. These sources are also all verified to be owned - or operated by - one of these groups at the last observed time-frame of our dataset (May 2018) by cross-checking publicly available information. We project their positions from the embedding space onto a display-friendly plane, showcasing snapshots of their movement over time, as they collapse into highly similar and cohesive clusters in the similarity space.
Right: Average inter-source cosine distance between sources of each group in the dynamic embedding space, over time.

6 ANALYSIS

In the following section, we discuss the interpretation of the model introduced in Section 4. We first describe an approach to visualize the evolution of the news ecosystem. Then, we propose a systematic, unsupervised way to detect abrupt deviations in this space.

6.1 Visualizing the Media Landscape

We start by introducing an example case-study, which should illustrate the usefulness of the presented model in facilitating the understanding of the news ecosystem. The study is centered on three representative media conglomerates that we track throughout the 40-month period covered by our dataset: *Gray Television Inc.* (over 100 television stations), *Sinclair Broadcasting Group* (over 190 television stations) and *GateHouse Media* (over 140 newspapers). [11]

As a starting point, the model described in Section 4 is optimized in its most successful setting (*RW+RG*). Dimensionality reduction is performed in order to have a more interpretable view into the embedding vectors. In Fig. 6 - (*left*) we use a t-distributed Stochastic Neighbor Embedding (t-SNE [44]), a popular method for the visualization of high-dimensional data. In order to maintain consistency across time steps in the projection, we initialize the parameters of the t-SNE optimization procedure at time slice t with the final parameters of time step $t-1$. This seeds the projected points' positions instead of assigning them to a random initial position, allowing for easier tracking between time steps.

Additionally, in order to avoid interpreting from model parameters only, which might be misleading due to optimization artifacts, we correlate sources' trajectories with the average pairwise cosine distance between sources of each group. These distances are computed using sources' respective sets of covered events for each month. Overall, this procedure allows us to coherently visualize the evolution of the media landscape over time, uncovering non-obvious dynamics at several scales, from the ecosystem as whole (e.g. convergence phenomenons) down to individual sources (e.g. shift toward a group).

The designed visualization of the media landscape is presented in Fig. 6. Please refer to Section 7 for a more detailed discussion.

6.2 Detecting Fluctuations in the Media Landscape

Even with extensive domain knowledge, tracking the evolution of a group of sources belonging to specific entities remains a tedious task, since it requires the manual identification of sources of interest and the validation of their common factors. Therefore, in the following section, we propose an unsupervised method which leverages the models' a priori knowledge to identify abrupt changes in sources' content diffusion patterns. In particular, the proposed framework aims to identify attractors, e.g. sources that tend of attract others in latent space, suggesting an alignment of coverage.

Attractors: News channels involved in a consolidation of resources typically tend to have increasingly similar coverage patterns. As seen in Fig. 6 - (*right*), the phenomenon manifests itself as the convergence of a subset of sources toward a common position in embedding space. Systematically detecting such gatherings around a common location, that we will loosely refer to as attractors, would allow to interpret each of these patterns in isolation. We propose a method in two steps. Firstly, we identify sources whose distances to other channels are abruptly reduced at any point in time. Secondly, we identify the absolute position towards which those sources tend to converge.

[11] as of August 2018.

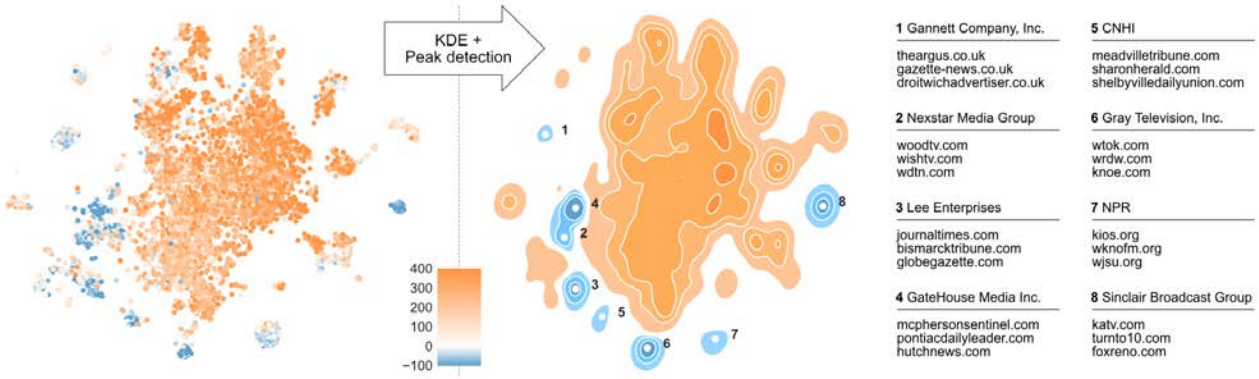


Figure 7: Left: t-SNE [44] projection of the embedding space for May 2018, colors represent the weight w_{s_i} of each source s_i. **Center:** Identified attractor map built from the attraction potential map ($\sigma = 1.9$, $k = 500$). **Right:** Detail and affiliation of attractors identified in (*center*), with the set of 3 sources closest to the uncovered poles. Best seen in color.

We first define the matrix $Z \in \mathbb{R}^{|S|\times|S|}$ that represents the cumulative difference of distances between any two news sources over time. We compute such distances in relative terms in order to avoid any drift component from the evolving latent space. More formally, we define Z as follows.

$$Z_{i,j} = \sum_t \; \|p_{s_i}^{(t)} - p_{s_j}^{(t)}\| - \|p_{s_i}^{(t-1)} - p_{s_j}^{(t-1)}\|\,, \tag{8}$$

where $\|\cdot\|$ is the Euclidean norm. In this setting, two sources whose distances are consistently reduced would produce a negative value of large magnitude in their corresponding entries of Z. Therefore, we rely on the matrix Z to identify channels having undergone large reductions of distance with other sources. In more details, we retrieve from Z the k-sources having the largest negative cumulative difference with any other source, i.e. the minimal value of each row in $Z_{i,j}$. By taking into consideration only the top-k, we can capture large shifts only and avoid considering small movements due to random factors. Once identified, the relative displacement of these k sources can be visualized in latent space. In particular, each source s_i in our dataset will be qualified by a single weight w_{s_i}, computed as the sum of the cumulative difference in Z with respect to the k considered sources. As shown in Fig. 7 - (*left*), negative values reveal sources that tend to exhibit agglomerative behaviors.

Until now, we observed strong fluctuations in terms of inter-source distances. The next step is to define a systematic way of identifying the centers around which these shifts occur, in absolute terms and at any given point in time. On a 2D projection of sources, we apply a weighted Kernel Density Estimator (KDE), a non-parametric method for estimating the Probability Density Function from a set of samples, under weak smoothness assumptions. The objective is to detect areas containing a high density of sources with negative weights and locate their density peaks. We use the weights w_{s_i} as input of the estimator. The bandwidth selection is a function of the data's covariance, multiplied by a constant factor σ. [42] An example of the resulting density is presented in Fig. 7 - (*center*). Finally, local extrema are collected using a local minimum filter, a simple method routinely used in computer vision. The set of identified poles and the top-3 closest sources surrounding them is shown in Fig. 7 - (*right*).

Attractees: Having identified a set of attractors in latent space, the dual observation can be made: the identification of sources that experienced large movements in latent space toward any of the previously identified attraction poles. These are sources that have been strongly influenced by external forces, for example in the content consolidation phase after an acquisition. The detection of these phenomenon is done relative to a specific pole of attraction. In order to track the distance to a pole over time, we start with a set of seed sources. An obvious choice is to study the top-3 closest sources from the poles, detailed in Fig. 7 - (*right*). The ranking of sources having undergone a large shift can once again be made systematic. In particular, we rank sources according to the largest difference in distance to the pole between two consecutive time steps. The distance to the centroid of these sources yields the distance maps shown in Fig. 8 for the top-4 sources with the largest shifts.

7 DISCUSSION

The structure of the news landscape is in a constant state of flux. It is often difficult to follow the evolution of its organizational structure and even more so to determine what influenced these changes. In this section, we discuss how the fluctuations of broadcast patterns can be informative about channels' organizational structure. We report that important changes in this structure are identifiable through abrupt shifts in content diffusion, and showcase the models' ability to systematically highlight this variability in the coverage space.

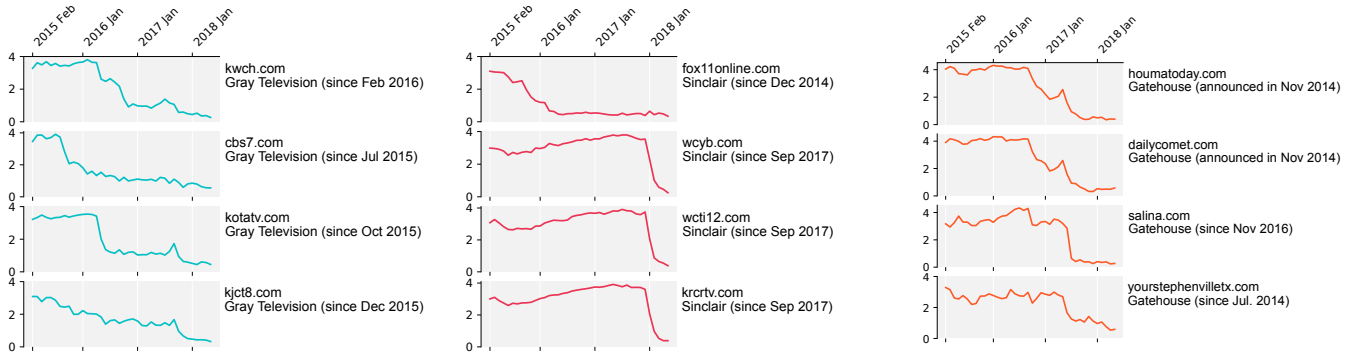


Figure 8: Euclidean distance from channels to their respective affiliations, in latent space, over time. Affiliation positions are computed as the centroid of 3 seed sources (taken from Figure 7 - (*right*)). Those patterns are extracted without any supervision: retrieved channels are the ones having undertaken the largest shift of content over the considered period (2015-2018). Channels acquisition dates are given for reference.

7.1 Observing the effects of ownership on the media landscape

The selection and diffusion of events by individual news channels could be influenced by a large number of factors, from obfuscated economic drivers to convoluted distribution schemes. Theoretically, the external pluralism assumption would prevent large-scale organizational changes in news outlets from inducing significant shifts in coverage. Our findings question the validity of this assumption.

We provide evidence that ownership can indeed exert its influence on the content being distributed downstream. The most distinct and recurring pattern pointing to this conclusion is the subsequent alignment of coverage patterns after an outlet's acquisition. Some examples are clearly observable in Fig. 6, such as the acquisition of 14 stations from the Bonten Media Group by Sinclair Broadcasting Group (SBG) in a deal completed on September 1st, 2017. [12] Visible through the lens of a decrease in the average inter-source distance of the Sinclair stations, this consolidation of coverage can also be tracked in the embedding spaces' visualization in Fig. 6 - (*left*). This can be observed in Fig. 8 as well, albeit with a slight delay with a sharp decrease in the distance from channels like `wcyb.com`, `wcti12.com` or `krcrtv.com` to the center of the Sinclair attraction pole (#8 in Fig. 7).

We observe similar behaviors during others large-scale acquisitions, for example in the purchase of a group of assets from the Morris Publishing Group by GateHouse in August 2017. [13] Other observations of this phenomenon include the sudden increase in coverage similarity of Gatehouse-owned stations around April 2017 (see Fig. 8 - *right*). While not directly correlated to a specific merger or acquisition, these movements could hint at a company-wide content alignment campaign.

Such observations also support the convergence hypothesis. The visualization in Fig. 6 exemplifies this effect: many of the sources that are present in one of these group's media portfolios start out from vastly different regions in embedding space. This is visible in their high initial average cosine distance in Fig. 6 - (*right*) and their dispersed placement in Fig. 6 - (*left*). In the last frame however, these same sources form highly coherent, tight groups in embedding space - and in the visualization. Despite the fact that this case-study back-tracks the evolution of sources across time, explaining the density of the last frame, their convergence points to a unification of coverage patterns over time.

7.2 Detecting highly influential broadcast groups

News outlets present complex content distribution schemes, as is particularly visible in television broadcasting: the on-air content is produced by a wide range of affiliates, from well-known household names to in-house teams, [14] being distributed through channels with another, often different set of owners. While the consolidation of broadcast material for economies of scale or investigative resources for economies of scope can be economically beneficial for the broadcaster, it is also potentially deceitful for a news consumer, as the exact origin of the broadcast content is not always known. By extension, the unique slant it carries in its selection of news is not clearly obvious. Not only does it carry is unique biases in terms of the way in which it covers the content, a topic not discussed in this work, but it also has the ability to over-emphasize or under-report certain events with little accountability.

This influence on coverage can be observed when interpreting the agglomeration dynamics highlighted by Fig. 7. Information sinks can be highlighted through the discovery of attraction poles in the embedding space.

[12] https://tvnewscheck.com/article/103465/sinclair-buying-bonten-stations-for-240m/

[13] https://www.poynter.org/news/gatehouse-acquires-morris-publishings-11-daily-newspapers

[14] A study conducted by Pew Research in 2014 already demonstrated a steady decline in locally produced content, with 1 in 4 local news stations not producing any of their own content [31].

Such sets of highly accretive sources, i.e. sources that draw other sources to align with their coverage, cluster neatly into large broadcast entities, some of which have been mentioned before. Fig. 7-(right) presents these groupings more exhaustively. The three large media groups chosen for analysis in Fig. 6 are present (*Gray Television Inc.*, *Sinclair Broadcasting Group* and *GateHouse Media Inc.*), along with several other large players in the American media landscape.

Previous studies [41] and [4] have studied differences in terms of the types of content covered in television and newspapers outlets, finding that TV stations cover proportionally more "global" news than newspapers. Television is traditionally thought to be more impacted by media consolidation for this reason: content is costly to produce, hence it makes sense for large entities to share their footage at scale. In our model, this should intuitively lead to the flagging of television conglomerates as the strongest attractors, with high content similarities. However, we observe in Fig. 7 that all mediums are represented and impacted by the convergence phenomenon. This could hint to the effect a convergence of mediums can have on the media landscape, with the efficacy co-ownership regulations being jeopardized by the all-encompassing nature of online content delivery.

7.3 Interpretation of the temporal consistency

None of this qualitative analysis would be possible without a temporal consistency constraint on the embedding space. Without such stability, the model could take advantage of an unnecessarily large number of degrees of freedom to align sources. In consequence, it would converge to very different solutions from one epoch to another. Due to the stochastic nature of the procedure, coverage changes would be rendered indistinguishable from optimization artifacts (see Fig. 3). By penalizing sources that deviate from their previous positions, only significant coverage changes can force a source to migrate to a different region in space. In other words, in order to provoke a displacement, the channels' coverage should differ enough from the previous time step to outweigh the temporal constraint. If this condition is met, the source will converge towards a different neighborhood that better fits its coverage patterns, typically getting closer to similar channels.

This variability in time can be tuned through the regularization parameter λ_T, as detailed in Section 4, providing a way to highlight more global dynamics - in the case of strong regularization - or more individual variations - with weak regularization. We also observe that the constraint provides predictive gains. This can be explained by the accumulation of knowledge about sources over time. This last hypothesis is corroborated by the pattern observed in Fig. 5, in which the accuracy reaches its maximum after the first few epochs before stabilizing until the end of the considered period.

8 CONCLUSION

This work tackles the problem of dynamically modeling the filtering decisions of individual actors of the news ecosystem. Beyond the predictive capabilities of the approach, the knowledge gathered by the model is leveraged to characterize the evolution of the media landscape over time. Firstly, we report performance benefits of adding a temporal component to a coverage prediction model. In particular, we show that a dynamic embedding model is able to outperform existing approaches thanks to its ability to propagate knowledge obtained from former time slices to the current prediction step. Secondly, we demonstrate the application of this model as a framework in which to reason about the latent structure of media landscape, modeling the temporal evolution of news outlets' decision processes. Maintaining a consistent latent representation of sources' preferences enables powerful interpretation and visualization methods, highly effective in investigating shifts in the media ecosystem, at a large scale but also at the individual source level.

We demonstrate the potential of the method on several channel acquisition campaigns. We show drastic post-acquisition content alignment in channels belonging to large, well-known broadcast conglomerates. This corroborates the hypothesis of deep consolidation of broadcast material inside news networks. Our work highlights the fragility of the external plurality assumption, which is supposed to guarantee a diversity of ownership and hence viewpoints. Finally, we automate this investigative process and explore several strategies to systemically identify abrupt variations in the news ecosystem, fingerprints of sharp changes in media programming.

Future work: The main focus of this work was to provide a clear and insightful representation of the news landscape, and we foresee several directions to pursue this research effort. Firstly, the method is ripe for the addition of content-based refinements. Our approach focuses on the information's propagation patterns but it is still blind to the content itself and the way it is handled by news outlets. We could for example mention in this vein a semantic analysis of the covered articles. The fact that the model is content-agnostic can however be of great use for its application to other domains. Such methods could be applied to other dynamic systems in which information propagates and evolves, such as online social-networks, knowledge bases or citation networks. Lastly, as has been highlighted throughout this work, particularly in the context of the case-study, we believe that our work would strongly benefit from an interdisciplinary approach, providing tools to journalists, policy-makers or economists, whose expertise would also add great insight to our analysis.

REFERENCES

[1] Evrim Acar, Daniel M Dunlavy, and Tamara G Kolda. 2009. Link prediction on evolving data using matrix and tensor factorizations. In *Data Mining Workshops, 2009. ICDMW'09. IEEE International Conference on*. IEEE, 262–269.

[2] Jisun An and Haewon Kwak. 2016. Two Tales of the World: Comparison of Widely Used World News Datasets GDELT and EventRegistry. In *Proceedings of the 10th International AAAI Conference on Web and Social Media (ICWSM16)*. 619–622.

[3] Simon P. Anderson and Stephen Coate. 2005. Market Provision of Broadcasting: A Welfare Analysis. *The Review of Economic Studies* 72, 4 (2005), 947–972. http://www.jstor.org/stable/3700696

[4] Bergan D. Fico F. Lacy S. Wildman S.S. Baldwin, T. 2010. News Media Coverage of City Governments in 2009. (2010).

[5] Dylan Bourgeois, Jérémie Rappaz, and Karl Aberer. 2018. Selection Bias in News Coverage: Learning it, Fighting it. In *The International World Wide Web Conference 2018*.

[6] Stefano DellaVigna, Ethan Kaplan, Alan B . Krueger, Marco Manacorda, Enrico Moretti, Torsten Persson, Sam Popkin, Riccardo Puglisi, Matthew Rabin, Jesse M. Shapiro, Uri Simonsohn, Laura Stoker, David Stromberg, Tatyana Deryugina, Monica Deza, Dylan Fox, Melissa Galicia, Calvin Wai-Loon Ho, Sudhamas Khanchanawong, Richard M. Kim, Martin Kohan, Vipul Surender Kumar, Jonathan J. Leung, Clarice Li, Tze Yang Lim, Ming Mai, Sameer Parekh, Sharmini Radakrishnan, Rohan Relan, Dan Acland, Saurabh Bhargava, Avi Ebenstein, and Devin G. Pope. 2005. The Fox News Effect: Media Bias and Voting.

[7] Simeon Djankov, Caralee McLiesh, Tatiana Nenova, and Andrei Shleifer. 2003. Who Owns the Media? *Journal of Law and Economics* 46, 2 (2003), 341–381.

[8] Gillian Doyle. 2002. *Media Ownership: The Economics and Politics of Convergence and Concentration in the UK and European Media*. https://doi.org/10.4135/9781446219942

[9] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. 2011. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 5, 2 (2011), 10.

[10] Matthew Gentzkow and Jesse Shapiro. 2010. What Drives Media Slant? Evidence From U.S. Daily Newspapers. *Econometrica* 78, 1 (2010), 35–71. https://EconPapers.repec.org/RePEc:ecm:emetrp:v:78:y:2010:i:1:p:35-71

[11] Matthew Gentzkow, Jesse M Shapiro, and Michael Sinkinson. 2009. *The Effect of Newspaper Entry and Exit on Electoral Politics*. Working Paper 15544. National Bureau of Economic Research. https://doi.org/10.3386/w15544

[12] L.M. George and Joel Waldfogel. 2008. National Media and Local Political Participation: The Case of the New York Times. (01 2008), 33–48.

[13] Deborah J. Gerner, Rajaa Abu-Jabr, Philip A. Schrodt, and ÃŰmÃijr Yilmaz. 2002. Conflict and Mediation Event Observations (CAMEO): A New Event Data Framework for the Analysis of Foreign Policy Interactions. In *of Foreign Policy Interactions. Paper presented at the International Studies Association*.

[14] Kristian Skrede Gleditsch, Nils W Metternich, and Andrea Ruggeri. 2014. Data and progress in peace and conflict research. *Journal of Peace Research* 51, 2 (2014), 301–314. https://doi.org/10.1177/0022343313496803 arXiv:https://doi.org/10.1177/0022343313496803

[15] Joshua S. Goldstein. 1992. A Conflict-Cooperation Scale for WEIS Events Data. *The Journal of Conflict Resolution* 36, 2 (1992), 369–385. http://www.jstor.org/stable/174480

[16] Tim Groseclose and Jeffrey Milyo. 2005. A Measure of Media Bias. *The Quarterly Journal of Economics* 120, 4 (2005), 1191–1237. https://EconPapers.repec.org/RePEc:oup:qjecon:v:120:y:2005:i:4:p:1191-1237.

[17] Ruining He, Chen Fang, Zhaowen Wang, and Julian McAuley. 2016. Vista: a visually, socially, and temporally-aware model for artistic recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 309–316.

[18] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 507–517.

[19] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. *2008 Eighth IEEE International Conference on Data Mining* (2008), 263–272.

[20] Henry Jenkins. 2004. The cultural logic of media convergence. *International journal of cultural studies* 7, 1 (2004), 33–43.

[21] Swetha Keertipati, Bastin Tony Roy Savarimuthu, Maryam Purvis, and Martin K. Purvis. 2014. Multi-level Analysis of Peace and Conflict Data in GDELT. In *MLSDA@PRICAI*.

[22] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 447–456.

[23] Haewoon Kwak and Jisun An. 2014. A first look at global news coverage of disasters by using the gdelt dataset. In *International Conference on Social Informatics*. Springer, 300–308.

[24] Preethi Lahoti, Kiran Garimella, and Aristides Gionis. 2018. Joint non-negative matrix factorization for learning ideological leaning on Twitter. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 351–359.

[25] Edward J. Laurance. 1990. Events data and policy analysis:. *Policy Sciences* 23, 2 (01 May 1990), 111–132. https://doi.org/10.1007/BF00175597

[26] Kalev Leetaru and Philip A. Schrodt. 2013. GDELT: Global data on events, location, and tone. *ISA Annual Convention* (2013).

[27] Jeffery J. Mondak. 1995. Media exposure and political discussion in u.s. elections. *Journal of Politics* 57, 1 (1 5 1995), 62–85. https://doi.org/10.2307/2960271

[28] Felix Oberholzer-Gee and Joel Waldfogel. 2006. *Media Markets and Localism: Does Local News en EspaÂśol Boost Hispanic Voter Turnout?* Working Paper 12317. National Bureau of Economic Research. https://doi.org/10.3386/w12317

[29] Alexandra Olteanu, Carlos Castillo, Nicholas Diakopoulos, and Karl Aberer. 2015. Comparing Events Coverage in Online News and Social Media: The Case of Climate Change. In *ICWSM*. AAAI Press, 288–297.

[30] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 502–511.

[31] Deborah Potter and Katerina Eva Matsa. 2014. *State of the News Media 2014: A Boom in Acquisitions and Content Sharing Shapes Local TV News in 2013*. Technical Report. Pew Research Center.

[32] David Pritchard. 2002. Viewpoint Diversity in Cross-Owned Newspapers and Television Stations: A Study of News Coverage of the 2000 Presidential Campaign. (September 2002). https://docs.fcc.gov/public/attachments/DOC-226838A7.pdf

[33] Fengcai Qiao, Pei Li, Jingsheng Deng, Zhaoyun Ding, and Hui Wang. 2015. Graph-Based Method for Detecting Occupy Protest Events Using GDELT Dataset. In *Proceedings of the 2015 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CYBERC '15)*. IEEE Computer Society, Washington, DC, USA, 164–168. https://doi.org/10.1109/CyberC.2015.77

[34] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 273–282.

[35] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*.

[36] Maja Rudolph and David Blei. 2017. Dynamic Bernoulli embeddings for language evolution. *arXiv preprint arXiv:1703.08052* (2017).

[37] Maja Rudolph, Francisco Ruiz, Stephan Mandt, and David Blei. 2016. Exponential family embeddings. In *Advances in Neural Information Processing Systems*. 478–486.

[38] Diego Saez-Trumper, Carlos Castillo, and Mounia Lalmas. 2013. Social media news communities: gatekeeping, coverage, and statement bias. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 1679–1684.

[39] Diego Sáez-Trumper, Carlos Castillo, and Mounia Lalmas. 2013. Social media news communities: gatekeeping, coverage, and statement bias. In *CIKM*.

[40] P. A. Schrodt, S. G. Davis, and J. L. Weddle. 1994. Political Science: KEDS–A Program for the Machine Coding of Event Data. *Social Science Computer Review* 12, 4 (1994), 561.

[41] Holli Semetko and Patti Valkenburg. 2000. Framing European Politics: A Content Analysis of Press and Television News. 50 (06 2000), 93 – 109.

[42] B. W. Silverman. 1986. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall, London.

[43] Peter O. Steiner. 1952. Program Patterns and Preferences, and the Workability of Competition in Radio Broadcasting. *The Quarterly Journal of Economics* 66, 2 (1952), 194–223. https://EconPapers.repec.org/RePEc:oup:qjecon:v:66:y:1952:i:2:p:194-223.

[44] L.J.P van der Maaten and G.E. Hinton. Nov 2008. Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research* 9: 2579âĂŞ2605 (Nov 2008).

[45] Tom Vizcarrondo. 2013. Measuring concentration of media ownership: 1976–2009. *International Journal on Media Management* 15, 3 (2013), 177–195.

[46] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G Carbonell. 2010. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of the 2010 SIAM International Conference on Data Mining*. SIAM, 211–222.

[47] James Edward Yonamine. 2013. A nuanced study of political conflict using the global datasets of events location and tone (GDELT) dataset. (2013).

[48] Wenchao Yu, Charu C Aggarwal, and Wei Wang. 2017. Temporally factorized network modeling for evolutionary network analysis. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 455–464.

Keyphrase Extraction from Disaster-related Tweets

Jishnu Ray Chowdhury
Kansas State University
Manhattan, KS
jishnurayc@ksu.edu

Cornelia Caragea
University of Illinois at Chicago
Chicago, IL
cornelia@uic.edu

Doina Caragea
Kansas State University
Manhattan, KS
dcaragea@ksu.edu

ABSTRACT

While keyphrase extraction has received considerable attention in recent years, relatively few studies exist on extracting keyphrases from social media platforms such as Twitter, and even fewer for extracting disaster-related keyphrases from such sources. During a disaster, keyphrases can be extremely useful for filtering relevant tweets that can enhance situational awareness. Previously, joint training of two different layers of a stacked Recurrent Neural Network for keyword discovery and keyphrase extraction had been shown to be effective in extracting keyphrases from general Twitter data. We improve the model's performance on both general Twitter data and disaster-related Twitter data by incorporating contextual word embeddings, POS-tags, phonetics, and phonological features. Moreover, we discuss the shortcomings of the often used F1-measure for evaluating the quality of predicted keyphrases with respect to the ground truth annotations. Instead of the F1-measure, we propose the use of embedding-based metrics to better capture the correctness of the predicted keyphrases. In addition, we also present a novel extension of an embedding-based metric. The extension allows one to better control the penalty for the difference in the number of ground-truth and predicted keyphrases.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; *Data mining*.

KEYWORDS

Keyphrase Extraction, Social Media Analytics, Disaster, Twitter

ACM Reference Format:
Jishnu Ray Chowdhury, Cornelia Caragea, and Doina Caragea. 2019. Keyphrase Extraction from Disaster-related Tweets. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313696

1 INTRODUCTION

Keyphrase extraction is the task of automatically extracting words or phrases from a text, which concisely represent the essence of the text. Keyphrases can be used for multiple tasks including text summarization, classification, and opinion mining. They can also be highly valuable for retrieving relevant documents when searching for specific information. In particular, keyphrases can be used to organize and retrieve useful information from Twitter.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313696

Twitter is an online social media platform, where people communicate via short, terse messages ("tweets"). Due to the ease of posting a tweet, and the popularity of the platform, Twitter can provide up-to-date news around the world, faster than standard media outlets. Thus, in recent years, Twitter has become an important source of real-time information [55, 61]. Specifically, during crisis situations, Twitter has revolutionized the way in which affected populations communicate with response organizations [6, 8]. On one hand, the affected populations can follow and spread the latest updates from government and response organizations on Twitter. On the other hand, affected individuals can post information that can be used to enhance situational awareness and, in some cases, save lives. Emergency hotlines, such as 911, could be overwhelmed by the large number of calls received during major disasters, and as a consequence, individuals often turn to social media, including Twitter, to request assistance [14, 23, 29, 45]. For example, a North Carolina family was rescued during Hurricane Florence after posting the following message on Twitter: *'If anybody could help… our car is under water and so is our house. Stuck in the attic. Phone is about to die. Please send help'* [23].

While social media can help enhance situational awareness, facilitate rescue operations, and ultimately save lives, its value is highly unexploited by response teams, in part due to the lack of tools that can help filter relevant, informative, and actionable information posted during disasters [60]. To address this limitation, the annotation of tweets with informative keyprhases can enable filtering of actionable information by the disaster management and response teams, in real time. In fact, in Twitter, hashtags assigned directly by the users posting the tweets are sometimes used to annotate tweets and can be seen as keyprases useful for filtering. However, a vast majority of tweets do not have any hashtags associated with them, as illustrated by a dataset we crawled during several crisis events that happened in the fall of 2017. Thus, to speed up the search and retrieval of potentially actionable disaster tweets posted by affected individuals, there is an urgent need for methods that can automatically extract keyphrases from tweets without primarily relying on hashtags. The extracted keyphrases can potentially be recommended to users as they are typing their messages, and thus a smaller number of relevant tweets would be retrieved by systems that facilitate filtering of information for the response teams.

Keyphrase extraction from tweets, however, is a non-trivial task due to the tweets' informal nature, noisiness and short length (currently, tweets have a limit of 280 characters). These factors limit the utility of many of the existing unsupervised keyword extraction techniques, which are designed to work on longer text using term-frequencies, word co-occurrences, and clustering methods. Recently, researchers started to explore keyphrase extraction from single tweets using deep-learning models. For example, Zhang et al.

[63] proposed a model based on joint-layer Recurrent Neural Networks (RNNs) and achieved an F1-score of 80.97% when used to extract keyphrases from the general Twitter. However, this RNN-based model has not been extended to extract keyphrases from tweets posted in a specific context, e.g., during a disaster. In fact, there is a distinctive lack of works for extracting keyphrases from disaster tweets. This task is especially challenging as the vocabulary used during a disaster is focused and limited, as compared to the vocabulary in a general collection of tweets. Furthermore, the keyphrases extracted should reflect information related to the event, people involved and their locations, the type of assistance needed, the type of situational awareness provided, etc. Thus, our research agenda is designed to answer the following questions:

(1) *Can we enhance the existing model to improve its performance on both disaster tweets and general tweets?*
(2) *How do the joint-layer-Recurrent Neural Network based models trained on general tweets perform on disaster tweets?*
(3) *How well do the joint-layer-Recurrent Neural Network models perform when trained and tested on disaster tweet data?*

Regarding question (1), we discovered that adding contextual word information, phonetics and phonological features, along with part of speech (POS) tag information, tends to increase the performance of the model on both the general domain and the disaster domain, when training the model on data from the relevant domain. Regarding question (2), it is expected that the model will not perform very well if trained on a general-domain Twitter data and tested on a specific-domain data (e.g., disaster data). However, considering that the general training data and the specific disaster test data are both collected from Twitter, it is still interesting to empirically find the degree to which the RNN-based model trained on general Twitter data can or cannot generalize to disaster data. Finally, regarding question (3), while it is expected that the model will have better performance when trained and tested on specific disaster tweet data, it is interesting to understand how significant the improvement is and what additional information helps the most.

While investigating the above research questions, we found that the exact match F1-based evaluation is not ideal for determining the quality of the extracted keyphrases. As an alternative to the exact match F1 metric, we explored embedding-based similarity metrics [11, 12, 35, 48], and proposed a novel extension to symmetric greedy embedding based similarity scoring. The extension allows for more fine-grained control over the penalty for the difference in the number of ground-truth and predicted keyphrases.

We describe our extensions to the original RNN-based model for keyphrase extraction [63] in Section 3. In Section 4, we present the details of our datasets. We describe the experimental setting in Section 5.1, and the details of our embedding-based evaluation metric in Section 5.2. Finally, we present and discuss our results in Section 5.3 and conclude the paper in Section 6.

2 RELATED WORK

Keyphrase extraction can be supervised or unsupervised. Supervised approaches treat keyphrase extraction as a classification problem. In supervised approaches, a model is trained to learn to classify keyphrases from training data that is annotated with keyphrases [7, 13, 18, 28, 56, 57, 62]. Many unsupervised keyphrase extraction techniques had also been previously proposed [10, 15, 24, 27, 33, 47, 49, 53]. They usually extract candidate keyphrases and rank them based on term frequencies, word co-occurrences, and other similar features. However, the length of tweets is usually too short for most of these techniques to be applicable. Currently, tweets are limited to a length of 280 characters, with the average length of the tweets in a collection being even smaller. Furthermore, keyphrases will rarely occur more than once in a single tweet. These issues along with the noisiness of tweets can make keyphrase extraction from social media platforms such as Twitter a difficult task.

Some of the previous works on keyphrase extraction from Twitter [5, 66] take the approach of inferring keyphrases from multiple tweets. Thus, they can only extract some topical information and trends that are common to multiple tweets. At the level of multiple tweets, some of the word-count based approaches for keyphrase extraction can be still applicable. However, our target is to extract keyphrases from single tweets. Along this line, the approaches proposed by Marujo et al. [30] and Zhang et al. [63] work for single tweets. Marujo et al. [30] showed that word embeddings in a system such as MAUI [31] performs better than the TF-IDF baseline [53]. Furthermore, the Joint-Layer-RNN model proposed by Zhang et al. [63] was shown to be even better than the model proposed by Marujo et al. [30]. The joint-layer RNN model can predict keyphrases of any arbitrary length, based on single tweets, without any significant pre-processing or hand-engineered features. Thus, we chose to focus on this model in our study.

Regarding retrieval and filtering of tweets during disasters, there are many previous works related to processing social media during emergency situations [19], extracting information from Twitter to enhance situational awareness [21, 58, 59], and classifying or clustering disaster-relevant tweets [3, 54, 64], among others. However, to the best of our knowledge, no existing works have focused specifically on the extraction of keyphrases from disaster tweets, a task that we focus on in this study.

Finally, in terms of evaluation, there are previous works related to embedding-based similarity metrics [11, 12, 35, 48], which we also investigate. These metrics have been used for paraphrase detection, assessment of student input [48], and other similar tasks including assessment of the quality of generated dialogues [25, 51] by conversational models. We show that embedding-based similarity metrics can also be used to compare the quality of predicted keyphrases with respect to the ground truth keyphrases. Our specific evaluation method is based on the symmetric greedy matching embedding-based metric, which was introduced in [48].

3 APPROACH

Recurrent Neural Networks (RNN): All models investigated in this work are based on RNN, and are inspired by the model proposed by Zhang et al. [63]. Given an input sequence, $(x_1, x_2, x_3, ..., x_n)$, an RNN processes each token x_t at every time-step t by producing a hidden state, h_t, using the previous hidden state, h_{t-1}, from the previous time-step $t-1$. Formally, this can be expressed as:

$$h_t = f(x_t, h_{t-1}) \tag{1}$$

where f is a non-linear activation function. The final output of an RNN can be a sequence of hidden states, $(h_1, h_2, h_3, ..., h_n)$. We used a specific type of RNN, called Long-Short Term Memory (LSTM)

network [17]. An LSTM is equipped with three gates (an input gate, i, an output gate, o, and a forget gate, f), along with a cell-state, c, and a hidden state, h. The function of an LSTM at each time-step, t, given the input, x_t, can be described with the following equations:

$$f_t = \sigma(x_t W_f + h_{t-1} U_f + b_f) \quad (2)$$

$$i_t = \sigma(x_t W_i + h_{t-1} U_i + b_i) \quad (3)$$

$$o_t = \sigma(x_t W_o + h_{t-1} U_o + b_o) \quad (4)$$

$$g_t = \tanh(x_t W_c + h_{t-1} U_c + b_c) \quad (5)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (6)$$

$$h_t = o_t \odot \tanh(c_t) \quad (7)$$

where $x_t \in \mathbb{R}^d, h_t, c_t \in \mathbb{R}^h, W_f, W_i, W_o, W_c \in \mathbb{R}^{dxh}, U_f, U_i, U_o, U_c \in \mathbb{R}^{hxh}$, $b_f, b_i, b_o, b_c \in \mathbb{R}^h$ (h represents the number of hidden units, and d represents the dimension of the input), σ is usually the sigmoid activation function, and $\odot$ is the Hadamard product.

The LSTM that we used is bidirectional [16], in that it has two encoders: one forward and one backward. The forward encoder starts processing the input sequence $(x_1, x_2, x_3, ..., x_n)$ from left to right, thus, creating a forward hidden state representation $(\overrightarrow{h_1}, \overrightarrow{h_2}, \overrightarrow{h_3}, .., \overrightarrow{h_n})$ of the input sequence. The backward encoder starts processing the input sequence from right to left, thus, creating a backward hidden state representation $(\overleftarrow{h_1}, \overleftarrow{h_2}, \overleftarrow{h_3}, .., \overleftarrow{h_n})$ of the input sequence. The final hidden state representation $(h_1, h_2, h_3, ..., h_n)$ is created by combining the forward and backward representations (often using the concatenation operator). Formally, we have:

$$\overrightarrow{h_t} = LSTM(x_t, \overrightarrow{h}_{t-1}) \quad (8)$$

$$\overleftarrow{h_t} = LSTM(x_t, \overleftarrow{h}_{t-1}) \quad (9)$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \quad (10)$$

To make predictions, linear transformations of the hidden states are then passed to a softmax layer.

Similar to the model proposed by Zhang et al. [63], our models use a stack of two Bi-LSTMs. The first Bi-LSTM layer takes the original sequence, $(x_1, x_2, x_3, ..., x_n)$, as input, and produces a hidden state sequence, $h^{(1)}_{1:n} = (h^{(1)}_1, h^{(1)}_2, h^{(1)}_3, ..., h^{(1)}_n)$. The second Bi-LSTM takes the hidden sequence produced by the first Bi-LSTM layer as input, and produces a sequence of deeper hidden state representations $h^{(2)}_{1:n} = (h^{(2)}_1, h^{(2)}_2, h^{(2)}_3, ..., h^{(2)}_n)$. Formally:

$$h^{(1)}_t = BiLSTM_1(x_t, h^{(1)}_{t-1}) \quad (11)$$

$$h^{(2)}_t = BiLSTM_2(h^{(1)}_t, h^{(2)}_{t-1}) \quad (12)$$

Following Zhang et al. [63], we jointly trained the two stacked Bi-LSTMs on two different but related tasks, in a supervised fashion. The first Bi-LSTM was trained to simply find keywords in a text. This can be viewed as a binary classification task, where detected keywords are classified as '1', and other words are classified as '0'. Let the loss function $J_1(\theta)$ represent the classification error of a non-linear transformation of the first hidden layer units, $h^{(1)}_{1:n}$. The hidden layer of the second Bi-LSTM was trained for a relatively more complex task that builds upon the task of the hidden layer of the first Bi-LSTM. Specifically, the second Bi-LSTM was trained to detect keyphrases by using its hidden units $h^{(2)}_{1:n}$ to tag the sequence using the BIOES scheme. In BIOES, 'B' refers to the beginning word of a keyphrase, 'E' refers to the last word of the keyphrase, 'I' refers to the word being inside the keyphrase (in-between 'B' and 'E'), 'S' refers to any single word keyphrase, and 'O' is any word that is not a part of a keyphrase. Let the loss function $J_2(\theta)$ represent the tagging error of a non-linear transformation of the second RNN hidden layer units, $h^{(2)}_{2:n}$. The final loss $J(\theta)$ can then be expressed as the combination of the two losses, $J_1(\theta)$ and $J_2(\theta)$, specifically:

$$J(\theta) = \gamma \cdot J_1(\theta) + (1 - \gamma) \cdot J_2(\theta) \quad (13)$$

where γ is a hyperparameter in the interval [0,1]. The model updates its trainable parameters to minimize the loss function, $J(\theta)$, through backpropagation during training. The loss from a low-level task (keyword discovery) can also serve as additional regularization for the main task (keyphrase extraction). Intuitively, the two stacked networks correspond to two different, but related tasks - essentially, it is an instance of multi-task learning [26, 52].

Contextual Word Embeddings: Word embeddings are high dimensional vector representations of words. Often, embedding algorithms encode contextual information into the vector representations, such that when clustering word-vectors using the Euclidean distance or Cosine-similarity, vectors corresponding to words that appear in similar contexts tend to cluster together. However, classic word embedding models [34, 42] have certain limitations. They can only accommodate one particular vector representation for each word. This restricts the models from being able to capture polysemy (the same word can have completely different meanings in different contexts). The models can also be restricted to a specific vocabulary, which the model has been trained on. Moreover, they can also be blind to morphological information.

Peters et al. [43] proposed the use of a pre-trained language model to create contextualized word embeddings (called ELMo embeddings), which can tackle some of these issues. They used a character level Convolutional Neural Network (CNN) [65] followed by a language model based on Bi-LSTM (however, other architectures can be used [44]). The word embeddings were constructed by a weighted summation of the hidden state representations of the words at various layers of the language model. As a result, there are some interesting benefits. As the initial word-embedding can be done by a CNN at the character level, there is no word-level vocabulary restriction. Also, as the embedding is done by a language model, the embedding of a word is conditioned on the context of all the surrounding words. Therefore, the same word can be represented using different word-vectors if it appears in different contexts.

As an enhancement to the main RNN-based model, which uses Glove embeddings[1] [42] to represent the input words, we concatenated the ELMo embeddings with GloVe embeddings pre-trained on a Twitter corpus. We used Flair[2][2] to load the Twitter GloVe embeddings and TensorFlow hub[3] to load the ELMo embeddings.

Modeling Tweet Noisiness: In Twitter, people often use shorthands, slang, arbitrary capitalizations, and varied lexical representations for a word, all of which contribute to a high overall noisiness

[1] https://nlp.stanford.edu/projects/glove/
[2] https://github.com/zalandoresearch/flair
[3] https://tfhub.dev/google/elmo/2

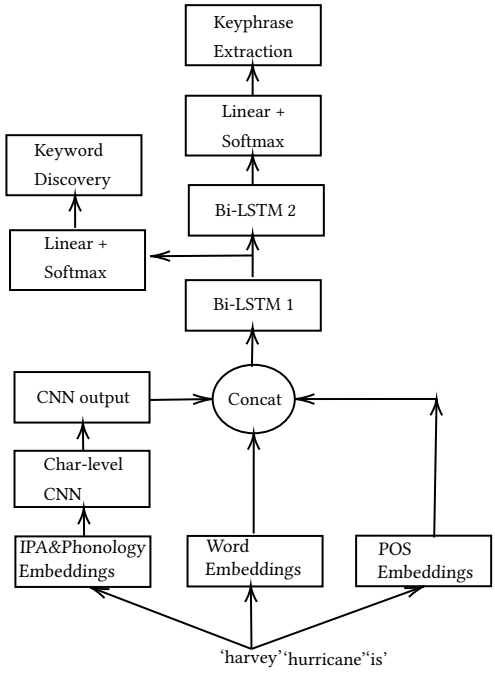


Figure 1: Joint-layer Bi-LSTM with word embeddings, POS-tags embeddings, and IPA & phonological embeddings.

in tweets. For this reason, many Natural Language Processing tools are found to perform noticeably worse when used on Twitter or similar social media domains, even when they perform very well on cleaner data (like news articles) [1, 46]. Aguilar et al. [1] proposed various ways to combat the inherent noisiness in social media platforms. They noted how Twitter users often tend to spell words based on their pronunciations. Thus, while spellings of a word may differ in their lexical format, they tend to have similar phonetics. Therefore, we can create more normalized representations of the words by using their phonetics or corresponding IPA (International Phonetic Alphabet) letters, alongside with their phonological features. Following the work of Aguilar et al. [1], we used Epitran[4] [36] to convert graphemes to phonemes (represented using IPA), and Panphon[5] [37] to convert each IPA phoneme into a vector representing various phonological (articulatory) features associated with it. In addition, we noticed that words that are nouns or adjectives are generally better candidates for keywords. For example, disaster names (nouns) can often be used as keyphrases. Therefore, similar to [1], we also added part-of-speech (POS) information to our model. We used the Twitter optimized POS-tagger[6] as provided by Owoputi et al. [41] for POS-tagging.

We used randomly initialized trainable embeddings for each POS-tag and IPA symbol. We directly used the phonological vector representations created with Panphon as embeddings for phonological features. We concatenated the embeddings of IPA symbols with their corresponding phonological feature vectors. The result of the concatenation, a character level representation of the phonetics and phonological features for each word, was fed to a character-level CNN [65], followed by a global max-pooling layer, to create word level representations. Note that the character-level processing may also allow the network to capture some morphological information. Unlike Aguilar et al., we chose to use a CNN as opposed to a Bi-LSTM for creating the word level representations, because using a Bi-LSTM at character level can be much more computationally expensive and time-consuming than using a CNN. The output of the CNN was concatenated with the POS embeddings, ELMo embeddings, and GloVe embeddings. The resultant representation was then fed to the stacked Bi-LSTM. Figure 1 illustrates this setup.

4 DATASETS

We performed our experiments on both general and disaster-related Twitter datasets. For the general Twitter dataset, we used the dataset[7] published in [63], which is annotated with keyprases. The training and test data were provided as separate files. We used 80% of the training data for actual training and 20% for validation.

We constructed the disaster training and validation datasets by combining disaster-related tweets that have hashtags, from several disasters, specifically, Boston Bombing, Hurricane Harvey, and Hurricane Sandy. We used 80% of the data for training and 20% for validation. We used the combination of disaster-related tweets, with hashtags, from Mexico Earthquake, Hurricane Maria, and Hurricane Irma as test data. To annotate the training, validation and test tweets with keyphrases, we first made use of hashtags. The hashtagged units (marked with '#') in the tweets used are often multi-word phrases without any spaces in-between. We used WordSegment[8] to split the hashtagged multi-word phrases into individual words. These segmented words were used as ground truth for keyphrase extraction. This is similar to how Zhang et al. [63] prepared their annotations. Since hashtags are often used to mark topical phrases or phrases related to some trends, they represent good candidates for keyphrases. In addition, we also used a crisis-lexicon [38] to find disaster-related words (at both unigram and bigram level) in the dataset, and marked them as ground-truth keyphrases. We also added some of our own phrases such as "Boston Bombing", "Hurricane Sandy", "Mexico Earthquake" to the crisis-lexicon.

For both the general and disaster datasets, we filtered out tweets that were longer than 200 tokens or shorter than 5 tokens after tokenization. Table 1 shows the exact number of tweets we used for training, validation, and testing in each dataset. In addition, Table 1 shows the total number of keyphrases and the average number of keyphrases per tweet in each dataset. As can be seen, the average number of keyphrases/hashtags per tweet in the disaster dataset is approximately twice the average number of keyphrases/hashtags per tweet.

There are other differences in terms of keyphrase usage between the general and the disaster datasets. First, the types of keyphrases used in the two datasets are quite different, as the disaster dataset contains a more limited vocabulary related to disasters, while the general dataset has a more diverse vocabulary. This can be seen by comparing the two wordclouds shown in Figure 2. Furthermore, from Figure 2, we can also see that there are many frequently

[4]https://github.com/dmort27/epitran
[5]https://github.com/dmort27/panphon
[6]http://www.cs.cmu.edu/~ark/TweetNLP/
[7]https://github.com/fudannlp16/KeyPhrase-Extraction
[8]http://www.grantjenks.com/docs/wordsegment/

General Twitter	Training Samples	62290
	Validation Samples	15572
	Test Samples:	33392
	Total Keyphrases	111254
	Avg Keyphrases per Tweet	1.00
Disaster Twitter	Training Samples	68128
	Validation Samples	17032
	Test Samples	37106
	Total Keyphrases	221408
	Avg Keyphrases per Tweet	1.81

Table 1: Dataset summary.

occurring keyphrases in the disaster data, whereas there are a lot fewer keyphrases that appear as frequently in the general data. At last, we point out that the disaster test data is quite different from the disaster training (and validation) data, given that the test data contains tweets related to different disasters as compared to the disasters in the training and validation data. In particular, a significant fraction of the disaster test data is made of disaster tweets posted during the Mexico Earthquake, while the training and validation data do not contain tweets related to earthquakes. However, both training/validation and test data contain tweets collected during hurricanes, although the hurricanes used in the training/validation are different from those used in the test. The differences in the disaster training and test data can be observed in Figure 3. We intentionally created a challenging test dataset that allows us to evaluate the model's performance on data from a disaster that does not appear in the training, to understand the usability of the model in a realistic scenario.

5 EXPERIMENTS AND RESULTS

5.1 Experimental Setting

One of our goals was to enhance the original model proposed in [63] with contextual information (in the form of contextual word embeddings), and also with information that can alleviate the noisiness of the Twitter data (specifically, phonology and POS tag information). Thus, our set of experiments was designed to compare the baseline model [63] with the proposed variants. The implementation details of the models are described below. Hyperparameters were selected based on common values used in similar prior architectures, and/or (limited) tuning on the validation set.

Joint-layer Bi-LSTM (J-Bi-LSTM): This model serves as our baseline. We parsed each sequence using a window of size 3, that is, each token in the sequence was represented by three consecutive words. As an example, if the sequence is (s_1, s_2, s_3), after parsing it with a window of size 3, the sequence gets converted into $([SOS, s_1, s_2], [s_1, s_2, s_3], [s_2, s_3, EOS])$ (where SOS is a special tag denoting the start of the sequence, and EOS is a similar tag denoting the end of the sequence). The 3-word representation of a token was used to encode more contextual information into each token, before feeding the tokens into the LSTM. Zhang et al. [63] empirically found 3 to be a good choice for the window size. Each word/token in the sequence was embedded with a 100-dimensional GloVe vector pre-trained on a general Twitter corpus. The embeddings were further fine-tuned during training. Each LSTM network had 300 hidden units. We used a dropout of 0.5 [4] on the input to the first and second LSTMs. We also used a dropout with the same rate on the output from the second LSTM. For training, we used the nadam optimizer [9] with a learning rate of 0.0015. Finally, we used an γ value of 0.5 for joining the two loss functions $J_1(\theta)$ and $J_2(\theta)$. Tuning this hyperparameter may lead to better results.

J-Bi-LSTM with ELMo Embeddings (J-Bi-LSTM+ELMo): This variant is similar to the baseline model, except that it uses 1024 dimensional ELMo embeddings, along with the GloVe embeddings. The two types of embeddings, GloVe and ELMo, are concatenated together. As the ELMo embeddings have much higher dimensionality, and are designed to encode contextual information, we did not use the window approach for this model. Other implementation details and hyperparameters are similar to those in the baseline.

J-Bi-LSTM with IPA & POS (J-Bi-LSTM+IPA+POS): This variant is also similar to the baseline, except that it uses IPA-embeddings, phonological feature vectors and POS-tag embeddings along with the GloVe embeddings (with a window of size 3). We used 64-dimensional embeddings for POS-tags and 22-dimensional embeddings for IPA symbols. The phonological feature vectors have 22 dimensions as well. For the character-level CNN, we used 128 filters and a kernel of size 3. As before, other implementation details and hyperparameters are similar to those in the baseline.

J-Bi-LSTM with ELMo, IPA & POS (J-Bi-LSTM+ELMo+IPA+POS): To evaluate the combined effect of the contextual word embeddings and IPA/POS-tags, this variant makes use of IPA-embeddings, phonological feature vectors and POS-tag embeddings, along with the concatenation of GloVe and ELMo embeddings. Other implementation details and hyperparameters are the same as before.

We compare the above models in three different settings: (1) we train all models on general tweet data and test them also on general tweet data. The goal of this comparison was to see if the additional information used can improve the performance of the baseline; (2) we train all models on the general tweet data and test them on the disaster tweet data, with the goal of understanding how much information can be transferred from the general tweet data to the disaster-specific tweet data; (3) we train all models on disaster tweet data and test them also on disaster tweet data. The goal of this experiment was to see how much the models improve by being trained on specific disaster data, and also what type of information leads to the largest performance improvements.

5.2 Evaluation Metrics

The J-Bi-LSTM based models take a sequence-labeling approach towards keyphrase extraction. Thus, the precision, recall, and F1-measure can be used to evaluate the results. A keyphrase predicted by the model is considered to be correct if there exists an exact matching keyphrase in the same position in the ground-truth annotation; otherwise, the prediction is considered incorrect.

Despite being frequently used, the F1-measure may not be ideal for evaluating the quality of predicted keyphrases. For example, if the

ground-truth keyphrase is "harvey", and the extracted keyphrase is "harvey hurricane", the exact-F1 measure will be zero. Therefore, the model can produce appropriate predictions, or sometimes even better predictions than the ground-truth annotations, but still, receives a zero score. Similarly, a model may detect some keyphrases that are semantically similar to the ground-truth, but not exact matches. For example, the model may detect 'tremor' instead of 'earthquake'. In these cases too, the model will receive a zero score, whereas presumably some points should be awarded to the model for predicting something close to the ground truth.

The above reasons led us to explore embedding-based measures. It has been shown that the cosine-similarity between certain vector representations (such as GloVe or word2vec embeddings) of semantically similar words tend to be high. Intuitively, we can use cosine-similarity of such word embeddings to measure the degree of similarity between them, as an alternative to simply checking for an exact match. Embedding-based metrics have been used in evaluating the quality of generated dialogues, detecting paraphrases, assessing student responses, and other similar tasks. Here, we are interested in comparing the ground-truth keyphrases with the predicted keyphrases. However, taking the order and positional information into account, while comparing them can be complicated. For example, consider a case where there are two gold keyphrases 'K_1' and 'K_2' at consecutive locations P_1 and P_2, respectively, whereas the predicted keyphrase is '$K_1\ K_2$' (covering both P_1 and P_2 locations). Such cases make it complicated to use positional information for comparison. Thus, we moved our focus from the standard sequence-labeling evaluation approach to an approach that seems more natural to the task of keyphrase extraction.

Specifically, we first extracted the gold and predicted keyphrases from a labeled sequence and converted them into sets of keyphrases. By doing this, the order information was ignored, and duplicates were removed. As a result, we obtained two sets - a set of gold keyphrases and a set of predicted keyphrases. Our goal was to use the embedding based metrics to measure the overlap between these sets. However, each keyphrase in the gold and predicted sets, respectively, can have an arbitrary number of words, each with its own word embedding. This further complicates the keyphrase-to-keyphrase comparison. To simplify the task, we took the average of the word-embeddings in each phrase to create phrase-level embeddings. Given the phrase-level embeddings, the embedding-based metrics can be used in several ways to score the overall quality of the predicted keyphrases using the gold keyphrases as the standard. Some of these methods are described below.

Average Embeddings: One approach is to take the average [11, 22, 35] of all phrase-level embeddings in the set of gold keyphrases. (Alternatively, we can also use the most extreme value [12] along each dimension, across all the phrase-level-embeddings in the set.) Then, we can either compute the average of cosine-similarity scores between the phrase-level embedding of each predicted keyphrase and the average-gold-embedding, or we can compute the cosine-similarity score between the average embedding of the predicted keyphrases and the average embedding of the gold keyphrases.

Greedy Matching: Another approach is to make word to word (or in our case, keyphrase to keyphrase) comparisons without creating single representations for all keyphrases in the sets. In this approach, we can score each keyphrase from the set of predicted keyphrases in two steps. In the first step, we compute the cosine-similarity scores between the predicted keyphrase under consideration and each keyphrase in the set of gold keyphrases. In the second step, we take the maximum of all the computed cosine-similarity scores for each predicted keyphrase. The maximum cosine-similarity score should correspond to the best matching keyphrase in the gold set for a particular predicted keyphrase. Thus, each predicted keyphrase can be scored. Finally, we can take the average of the scores to get the final score for the model's prediction [48]:

$$GM(\ pred, gold\ ) = \frac{\sum_{i=1}^{p} score(pred_i, gold)}{p} \tag{14}$$

where p is the number of predicted keyphrases, *pred* is the set of predicted keyphrases, *gold* is the set of gold keyphrases, and $pred_i$, with $i \in \overline{1,p}$, are the elements of *pred*. *GM* is the greedy match scoring function, and $score(pred_i, gold)$ is the maximum cosine-similarity achieved between $pred_i$ and any keyphrase from *gold*.

However, this scoring is not symmetric. $GM(\ pred, gold\ )$ will not necessarily be the same as $GM(\ gold, pred\ )$. Inspired from [48], we can use the following formula to calculate a symmetric score:

$$T = GM(\ pred, gold\ ) + GM(\ gold, pred\ ) \tag{15}$$

$$SymmGM(\ pred, gold\ ) = \frac{T}{2} \tag{16}$$

There is another benefit of the symmetric scoring. If $GM(pred, gold)$ is used alone, the model can be scored highly even if there is only one prediction in the set which happens to match very well with just one gold keyphrase, among multiple other gold keyphrases that were not predicted. Similarly, $GM(\ gold, pred\ )$ can produce a high score, even if there is only one keyprhase in the gold set that matches very well with just one predicted keyphrase, among multiple other predicted keyphrases that do not match with the gold. Using $SymmGM(\ pred, gold\ )$ mitigates these issues.

Optimal matching: This approach, proposed by Rus and Lintean [48], can be used to first create an optimal group of exclusive phrase-to-phrase pairs using the Kuhn-Munkres method [20] for optimal assignment. Then, each predicted keyphrase can be scored simply by computing its cosine-similarity with the gold-keyphrase that it is paired with. The later steps are similar to those of greedy matching.

Proposed embedding-based metric: We chose greedy matching as the backbone of our proposed embedding-based metric. As we are dealing with small sets of discrete keyphrases, we found it more appropriate to choose a phrase-to-phrase comparison approach such as greedy matching or optimal matching. A predicted keyphrase can be considered appropriate if it is similar to any of the gold keyphrases. Therefore, to score the correctness of the predictions, we chose to allow multiple predicted keyphrases to be matched to the same gold keyphrase if needed. However, optimal matching can only allow exclusive one-to-one pairs. This is why we did not choose the optimal matching.

We should note that if we simply use the average score when calculating $GM(\ pred, gold\ )$ and $GM(\ gold, pred\ )$, the variation in the

	Gold and prediction	F1	No alpha-beta No threshold General Embd	No alpha-beta No threshold Crisis Embd	General Embd	Crisis Embd
1	**Gold:** teen choice **Prediction:** teen choice awards	0%	95.6%	88.4%	95.6%	88.4%
2	**Gold:** earthquake, volunteers, working together **Prediction:** earthquake, volunteers working together	40%	93.1%	92.1%	80.5%	79.8%
3	**Gold:** storm, cloud, calamity **Prediction:** tornado, cloudy sky, disaster	0%	61.1%	30.9%	52.3%	13.4%
4	**Gold:** harvey **Prediction:** hurricane	0%	25.2%	63.20%	0%	63.20%
5	**Gold:** hurricane harvey, hurricane, katrina, red cross, bombing **Prediction:** hurricane, harvey, katrina hurricane, cross, boston bombing	20%	90%	93.4%	90%	93.4%
6	**Gold:** boston bombing, hurricane harvey, red cross **Prediction:** hurricane harvey	50%	83.3%	77.4%	54.2%	44.4%
7	**Gold:** boston bombing, hurricane harvey **Prediction:** hurricane harvey, boston, bombing, red cross	33.3%	90.2%	90.3%	70.3%	70.2%

Table 2: General and crisis embedding-based scores versus F1 scores on seven sets of gold/predicted keyphrases.

number of keyphrases in *gold* and *pred* may not be completely taken into account by greedy matching. More specifically, $GM(pred, gold)$ can be high as long as all the predicted keyphrases achieve high cosine-similarity scores with any one of the gold keyphrases. But it may not take into account how many more or less keyphrases there may be in the *gold* set as compared to those in the *pred* set. Similarly, $GM(gold, pred)$ can be high, as long as all the gold keyphrases achieve high cosine-similarity scores with any of the predicted keyphrases, without taking into account how many more or less keyphrases there may be in the *pred* set as compared to those in the *gold* set. Both $GM(pred, gold)$ and $GM(gold, pred)$ may be high at the same time despite wide variation in the number of predicted keyphrases and gold keyphrases, when most of the keyphrases in *gold* and *pred* are very similar to each other. To account for this, we may want to penalize the model for extracting too little or too many keyphrases compared to the gold. This can be done with a simple modification to the *GM* definition:

$$GM'(pred, gold) = \frac{\sum_{i=1}^{p} score(pred_i, gold)}{max(p, g)} \tag{17}$$

$$GM'(gold, pred) = \frac{\sum_{i=1}^{g} score(gold_i, pred)}{max(p, g)} \tag{18}$$

$$T' = GM'(pred, gold) + GM'(gold, pred) \tag{19}$$

$$SymmGM'(pred, gold) = \frac{T'}{2} \tag{20}$$

where g is the number of gold keyphrases, and p is the number of predicted keyphrases as defined before. $SymmGM'(pred, gold)$ is the overall score. If g is higher than p, the $GM'(pred, gold)$ score will be lower as compared to the $GM(pred, gold)$ score, and if p is higher than g, then $GM'(gold, pred)$ will be lower as compared to $GM(gold, pred)$. Intuitively, we can think of $GM'(pred, gold)$ as penalizing the overall score, when g is higher than p, and we can think of $GM'(gold, pred)$ as penalizing the overall score when p is higher than g.

However, we may also want to have more fine-grained control over the degree of penalty. Typically, $SymmGM(pred, gold)$ will be high even when there is wide variation in the lengths of *pred* and *gold* sets if most of the predicted and gold kephrases are similar to each other. But if that is the case, the predictions are still likely to be appropriate. Moreover, in general, variation in the number of keyphrases can be implicitly penalized by plain averages (with $GM(gold, pred)$ and $GM(pred, gold)$) because with wide variation, there will be higher chances for there to be phrases unique to only one set, which are dissimilar with any of the phrases in the other set. For these reasons, we may want to constrain the penalty for the variation of length, which can be achieved as another extension of the *GM* scoring. First note that:

$$max(p, g) = p + max(0, g - p) = g + max(0, p - g) \tag{21}$$

To gain more fine-grained control over the penalty for the length variation, we use two new parameters α and β, as follows:

$$GM_{ext}(pred, gold) = \frac{\sum_{i=1}^{p} score(pred_i, gold)}{p + \alpha \cdot max(0, g - p)} \tag{22}$$

$$GM_{ext}(gold, pred) = \frac{\sum_{i=1}^{g} score(gold_i, pred)}{g + \beta \cdot max(0, p - g)} \tag{23}$$

$$T_{ext} = GM_{ext}(pred, gold) + GM_{ext}(gold, pred) \tag{24}$$

$$G(pred, gold) = \frac{T_{ext}}{2} \tag{25}$$

Intuitively, α and β can be used to modulate the penalty for p being smaller than g, and for p being greater than g, respectively. This leads to a more general formulation. For example, when α is 0, $GM_{ext}(pred, gold)$ is just $GM(pred, gold)$, and when α is 1, $GM_{ext}(pred, gold)$ is just $GM'(pred, gold)$. $G(pred, gold)$ is symmetric when α and β have the same value.

We used $G(pred, gold)$ to compare a predicted keyphrase set with a gold keyphrase set, with α and β both being set to 0.7. In addition, we also used a threshold θ of 0.4 on each cosine-similarity scores to limit the contribution of lower-scoring matches, and implicitly, to limit the range of $G(pred, gold)$ to values between 0 and 1. That is, if the cosine similarity score is lower than a specified

threshold, we assigned it a score of 0, otherwise, the actual score is used. The values of α, β, and θ were selected intuitively. There is room for empirical studies in the future to determine how people judge the evaluation quality for different values of α, β, and θ.

Embedding-based versus F1-measure evaluation: In Table 2, we used several sets of predicted keyphrases and gold keyphrases to compare the exact-match F1-measure with greedy matching (no alpha-beta and no threshold), and our proposed embedding-based metric. The last five examples are fabricated pairs of gold keyphrases and predictions to serve as case studies for comparison. For the embedding-based metric, we used both GloVe embeddings pre-trained on general Twitter (General Embd), and GloVe embeddings that we trained on a crisis-related Twitter corpus (Crisis Embd). The crisis-related embeddings were trained on three previously published crisis datasets, specifically, CrisisLexT6 [39], CrisisLexT26 [40] and 2CTweets [50], together with tweets collected using the Twitter Streaming API during Hurricanes Harvey, Irma, and Maria, and Mexico Earthquake. The total corpus contains approximately 5.8 million tweets. As can be seen in Table 2, the embedding-based scores appear to be more appropriate than the F1 scores, based on a human understanding of the keyphrases. We can also notice (see example 3) that the general Twitter GloVe embeddings can be better for scoring the semantic similarity between general words than the crisis embeddings. This may be because the general Twitter embeddings were trained on a much larger corpus, which is more diverse and general. However, crisis embeddings appear to be better for scoring the similarity between disaster-related words, especially if disaster-related named-entities are involved (see example 4). This is probably because the crisis embeddings were trained on more recent Twitter data and, specifically, on disaster-related data. Not so surprisingly, we also find that crisis embedding-based scores are higher on the disaster data, and general Twitter embedding-based scores are higher on the general data in Table 4. Alpha-beta modulated embedding scores are generally more stringent than embedding scores without the modulation. Without alpha-beta modulation and thresholding, the embedding metrics work as standard greedy matching. The difference made by alpha-beta modulation is most sharply noticeable in examples 6 and 7 where there is a significant difference in the number of gold keyphrases and predicted keyphrases. As can be seen, without alpha-beta modulation, the embedding scores are very high despite the difference in length between the two sets of keyphrases.

5.3 Results

The results of our experiments for the three settings discussed in Section 5.1 are presented in Tables 3 and 4. Table 3 shows precision, recall, and F1 scores, based on exact keyphrase matches (where the order of the keyphrases and the position are both taken into account). Table 4 shows F1 scores versus GM_{ext} embedding-based scores, computed by comparing a set of predicted keyphrases with a set of gold keyphrases (where the order of the keyphrases and the position are both ignored, and duplicate keyphrases are removed). Thus, while the F1 scores in Tables 3 and 4 may appear to be inconsistent, we would like to point out that the differences are due to the way the scores are calculated in each table, as explained above.

Model	P	R	F_1
Trained and tested on general data			
J-Bi-LSTM	83.95%	80.81%	82.35%
J-Bi-LSTM+ELMo	84.39%	**81.65%**	82.99%
J-Bi-LSTM+IPA+POS	84.94%	81.63%	83.25%
J-Bi-LSTM+ELMo+IPA+POS	**85.25%**	81.53%	**83.35%**
Trained on general data; tested on disaster data			
J-Bi-LSTM	**37.49%**	**09.92%**	**15.68%**
J-Bi-LSTM+ELMo	35.88%	09.69%	15.26%
J-Bi-LSTM+IPA+POS	34.53%	08.79%	14.01%
J-Bi-LSTM+ELMo+IPA+POS	36.50%	09.06%	14.51%
Trained and tested on disaster data			
J-Bi-LSTM	65.21%	61.46%	63.28%
J-Bi-LSTM+ELMo	67.67%	59.60%	63.38%
J-Bi-LSTM+IPA+POS	**69.06%**	**61.98%**	**65.33%**
J-Bi-LSTM+ELMo+IPA+POS	67.14%	60.37%	63.57%

Table 3: Precision, Recall, and F1 scores based on the sequence-labeling evaluation (using exact matches).

Model	F1	General Embd	Crisis Embd
Trained and tested on general data			
J-Bi-LSTM	84.26%	92.45%	89.46%
J-Bi-LSTM+ELMo	84.41%	92.64	89.67%
J-Bi-LSTM+IPA+POS	84.79%	92.88%	90.06%
J-Bi-LSTM+ELMo+IPA+POS	**85.10%**	**93.22%**	**90.42%**
Trained on general data; tested on disaster data			
J-Bi-LSTM	**20.45%**	32.64%	44.98%
J-Bi-LSTM+ELMo	19.38%	32.74%	45.89%
J-Bi-LSTM+IPA+POS	19.11%	34.33%	**46.49%**
J-Bi-LSTM+ELMo+IPA+POS	20.28%	**34.34%**	45.77%
Trained and tested on disaster data			
J-Bi-LSTM	64.05%	78.05%	82.03%
J-Bi-LSTM+ELMo	64.12%	78.41%	**82.51%**
J-Bi-LSTM+IPA+POS	**66.37%**	**78.55%**	82.40%
J-Bi-LSTM+ELMo+IPA+POS	65.11%	78.16%	82.25%

Table 4: General and crisis embedding-based scores versus F1 scores, when comparing predicted with gold keyphrases.

We calculated the F1-scores in Table 4 by comparing sets, in order to make them more comparable to the embedding-based scores. We will next analyze the performance of each model individually.

Baseline (J-Bi-LSTM): The baseline achieved scores in the low-to-mid 80% range when trained and tested on general data, and scores in the low-to-mid 60% range when trained and tested on disaster data. Surprisingly, its performance is close to the performance of other models, which use richer information. Furthermore, the J-Bi-LSTM model produced higher recall than models with ELMo on disaster test data. Also, interestingly, the baseline model achieved the best F1 score in Table 3, when trained on general data and tested

on disaster data. However, it was not the best scorer if we consider its embedding-based scores in Table 4 instead.

J-Bi-LSTM+ELMo: The model enhanced with contextual word embeddings achieved a slightly higher score than the baseline when trained and tested on general Twitter data. An even better score may have been possible if the ELMo embeddings were pre-trained on a Twitter corpus. However, the F1 performance of this model on disaster data after being trained on general data is lower than the performance of the baseline, but its embedding-based scores are higher. There is no significant difference between the overall F1 score of the baseline and the F1 score of this model when trained and tested on disaster data. Although the model seems to get slightly better precision, it struggles on recall more than the baseline.

J-Bi-LSTM+IPA+POS: The addition of POS, IPA and phonological feature information generally improves the performance of the J-Bi-LSTM+IPA+POS model over the baseline. The model even gets a better score than the J-Bi-LSTM+ELMo model, and it achieves similar performance to J-Bi-LSTM+ELMo+IPA+POS on the general test data. So, simply adding POS, IPA and phonological information may eliminate the need for ELMo embeddings. This could be because both ELMo and IPA (with phonological features) can model some similar character-level information (like morphology). Furthermore, as mentioned above, ELMo may not be expressing its full potential in the Twitter domain because it was not pre-trained on a noisy social media corpus. Interestingly, the J-Bi-LSTM+IPA+POS appears to have the worst F1 score when trained on general data and tested on disaster data, but it performs the best in terms of the crisis embedding-based metric. When trained and tested on disaster data, this model is also one of the better performers.

J-Bi-LSTM+ELMo+IPA+POS: Finally, the model enhanced with ELMo embeddings, and also POS, IPA and phonological feature information, achieved the best overall F1 score when trained and tested on general data, but not in the other two scenarios. It seems that the ELMo embeddings and the POS/phonetics may contain somewhat overlapping information. ELMo can capture some character level information, which may be why adding further character level information from IPA and phonological features do not give a significant boost in performance. Overall the performance of this model is similar to the performance of the JRNN+ELMo model on the disaster data. We also considered training models using crisis embeddings but they did not help.

In the remaining of this section, we will use the results in Tables 3 and 4 to answer our original questions. First, regarding question (1), we can say that the models enhanced with contextual and/or POS/phenomes information generally perform better than the baseline model, although the baseline is not much worse.

Regarding our original question (2), we noted that the models achieved much lower F1 scores when trained on general data and tested on disaster data. It should be expected that the domain differences contribute to the low score. However, there is another important reason. As we mentioned before in Section 4, the average number of keyphrases per disaster tweet is approximately twice the average number of keyphrases per general tweet. Thus, it is likely that the models, when trained on the general tweet data, learn to predict one keyphrase per sample on average. This results in a substantial hit to their recall when tested on the disaster data, which presents more variety in the number of keyphrases per tweet. Nevertheless, as can be seen from figure 4(a), a model trained on general data, can still predict keyphrases similar to the gold (Figure 3(b)). Furthermore, their embedding-based scores are also much higher than their overall F1 scores. Due to the fundamental differences in the two datasets (as discussed in more details in Section 4), a model may learn some parameters which are useful for general tweets, but the same parameters may be counter-productive in the case of tweets from the disaster test data. This may be why the models that performed better on general data struggled more to generalize on disaster data. But the results are not too straightforward, because some of those models did generalize better on disaster data when considering their embedding-based scores. Overall, it may be better to be cautious about judging the models' capabilities based on their performance on a domain they were not even trained on.

Finally, to answer question (3), we can claim that the models trained and tested on the disaster data can perform significantly better than the models trained on general data and tested on disaster data. However, the performance scores on the disaster test data are overall lower than those on the general test data. This is because we used a rather challenging test set for the disaster data, as discussed earlier. Furthermore, the low diversity of keyphrases in the disaster training data can encourage the models to memorize more (and thus overfit) instead of learning more general abstract features. Nevertheless, they can still predict keyphrases that are fairly similar to the gold (see Figures 3(b) and 4(b)), and their embedding-based scores are better than the F1 scores.

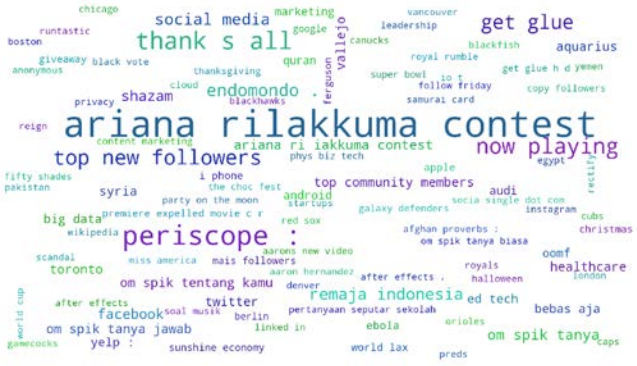
6 CONCLUSIONS AND FUTURE WORK

We showed that the Joint-layer Bi-LSTM model can be a promising approach for extracting disaster-related keyphrases from Tweets. We also showed that ELMo embeddings and/or phonetics and phonological features, along with POS-tags, can generally improve the performance of the model. We discussed how the exact match F1 can be an unsuitable metric for evaluating the quality of the predicted keyphrases, and we showed that embedding-based metrics are more appropriate for this task. We further introduced novel extensions to the embedding-based metrics, to account for differences in terms of the number of predicted and gold keyphrases.

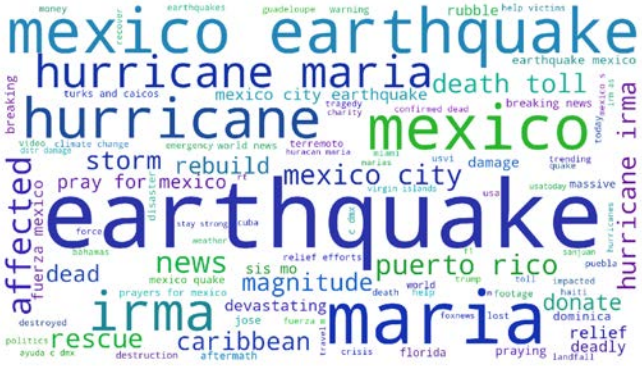
In future work, we would like to focus on abstractive keyphrase generation [32]. Taking a generative approach as opposed to an extractive one would allow us to generate keyphrases that are absent from the source text but still appropriate given the semantics of the text. This can further enhance keyphrase-based search of tweets related to a particular context (including disaster context).

ACKNOWLEDGEMENTS

We thank the National Science Foundation (NSF) and Amazon Web Services for support from grants IIS-1741345 and IIS-1912887, which supported the research and the computation in this study. We also thank NSF for support from the grants IIS-1526542, IIS-1423337, IIS-1652674, and CMMI-1541155.

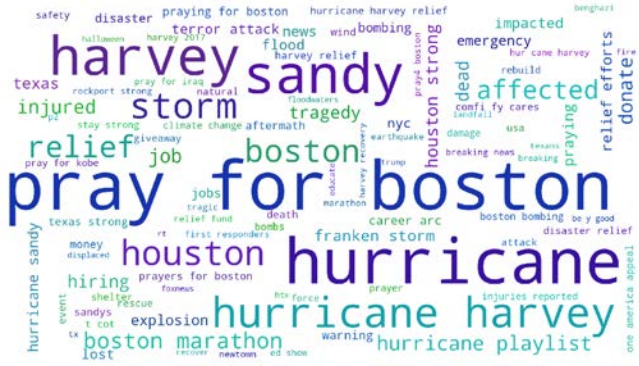


(a) General test data

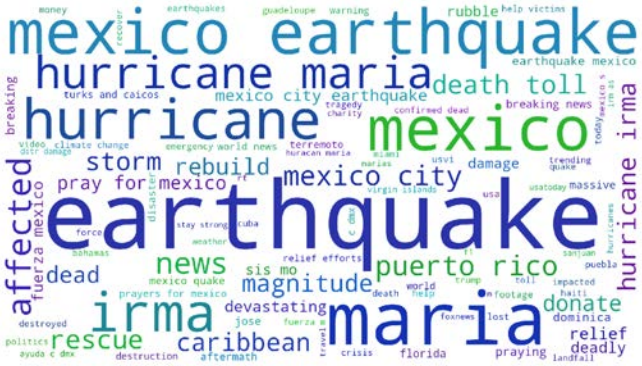


(b) Disaster test data

Figure 2: Word clouds of the top 100 most frequent gold keyphrases from the general test data (a) and disaster test data (b).

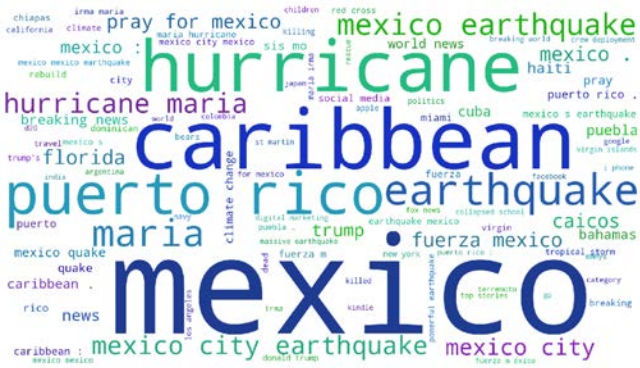


(a) Disaster training data

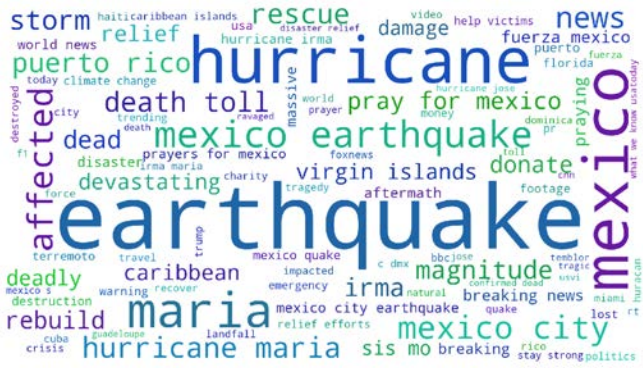


(b) Disaster test data

Figure 3: Word clouds of the top 100 most frequent gold keyphrases from disaster training data (a) and disaster test data (b).



(a) Predictions by model trained on general data



(b) Predictions by model trained on disaster data

Figure 4: Word clouds of the top 100 most frequent keyphrases from the disaster test data, as predicted by the J-Bi-LSTM+IPA+POS model trained on the general data (a) and J-Bi-LSTM+IPA+POS trained on disaster data (b).

REFERENCES

[1] Gustavo Aguilar, Adrian Pastor López Monroy, Fabio González, and Thamar Solorio. 2018. Modeling Noisiness to Recognize Named Entities using Multitask Neural Networks on Social Media. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Vol. 1. Association for Computational Linguistics, New Orleans, Louisiana, 1401–1412.

[2] Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual String Embeddings for Sequence Labeling. In *COLING*. Association for Computational Linguistics, Santa Fe, New Mexico, USA, 1638–1649.

[3] Zahra Ashktorab, Christopher Brown, Manojit Nandi, and Aron Culotta. 2014. Tweedr: Mining twitter to inform disaster response. In *ISCRAM*.

[4] Pierre Baldi and Peter J Sadowski. 2013. Understanding dropout. In *Advances in neural information processing systems*. 2814–2822.

[5] Abdelghani Bellaachia and Mohammed Al-Dhelaan. 2012. Ne-rank: A novel graph-based keyphrase extraction in twitter. In *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01*. IEEE Computer Society, 372–379.

[6] Nicola Bruno. 2011. Tweet first, verify later? How real-time information is changing the coverage of worldwide crisis events. *Reuters Institute for the Study of Journalism* (2011), 2010–2011.

[7] Cornelia Caragea, Florin Adrian Bulgarov, Andreea Godea, and Sujatha Das Gollapalli. 2014. Citation-Enhanced Keyphrase Extraction from Research Papers: A Supervised Approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. 1435–1446.

[8] Guy Paul Cooper Jr, Violet Yeager, Frederick M Burkle Jr, and Italo Subbarao. 2015. Twitter as a potential disaster risk reduction tool. Part I: Introduction, terminology, research and operational applications. *PLoS currents* 7 (2015).

[9] Timothy Dozat. 2016. Incorporating nesterov momentum into adam. (2016).

[10] Corina Florescu and Cornelia Caragea. 2017. PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. 1105–1115.

[11] Peter W Foltz, Walter Kintsch, and Thomas K Landauer. 1998. The measurement of textual coherence with latent semantic analysis. *Discourse processes* 25, 2-3 (1998), 285–307.

[12] Gabriel Forgues, Joelle Pineau, Jean-Marie Larchevêque, and Réal Tremblay. 2014. Bootstrapping dialog systems with word embeddings. In *Nips, modern machine learning and natural language processing workshop*, Vol. 2.

[13] Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. 1999. Domain-Specific Keyphrase Extraction. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI '99)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 668–673. http://dl.acm.org/citation.cfm?id=646307.687591

[14] Willa Frej. 2018. Hurricane Florence Flood Victims Turn To Social Media For Rescue. *HuffPost* (2018). https://www.huffingtonpost.com/entry/hurricane-florence-flood-victims-social-media_us_5b9b86c0e4b013b097799cd1

[15] Sujatha Das Gollapalli and Cornelia Caragea. 2014. Extracting Keyphrases from Research Papers Using Citation Networks. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*. 1629–1635.

[16] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. 2013. Hybrid speech recognition with deep bidirectional LSTM. In *Automatic Speech Recognition and Understanding*. IEEE, 273–278.

[17] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[18] Anette Hulth. 2003. Improved Automatic Keyword Extraction Given More Linguistic Knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP '03)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 216–223. https://doi.org/10.3115/1119355.1119383

[19] Muhammad Imran, Carlos Castillo, Fernando Diaz, and Sarah Vieweg. 2015. Processing social media messages in mass emergency: A survey. *ACM Computing Surveys (CSUR)* 47, 4 (2015), 67.

[20] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval research logistics quarterly* 2, 1-2 (1955), 83–97.

[21] Shamanth Kumar, Geoffrey Barbier, Mohammad Ali Abbasi, and Huan Liu. 2011. TweetTracker: An Analysis Tool for Humanitarian and Disaster Relief.. In *ICWSM*.

[22] Thomas K Landauer and Susan T Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review* 104, 2 (1997), 211.

[23] Tamar Lapin. 2018. Family's tweet for help leads to rescue during Hurricane Florence. *New York Post* (2018). https://nypost.com/2018/09/16/familys-tweet-for-help-leads-to-rescue-during-hurricane-florence/

[24] Marina Litvak, Mark Last, Hen Aizenman, Inbal Gobits, and Abraham Kandel. 2011. DegExt—A language-independent graph-based keyphrase extractor. In *Advances in Intelligent Web Mastering–3*. Springer, 121–130.

[25] Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023* (2016).

[26] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101* (2016).

[27] Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*. Association for Computational Linguistics, 257–266.

[28] Patrice Lopez and Laurent Romary. 2010. HUMB: Automatic Key Term Extraction from Scientific Articles in GROBID. In *Proceedings of the 5th International Workshop on Semantic Evaluation (SemEval '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 248–251. http://dl.acm.org/citation.cfm?id=1859664.1859719

[29] Douglas MacMillan. 2017. In Irma, Emergency Responders' New Tools: Twitter and Facebook. *The Wall Street Journal* (2017). https://www.wsj.com/articles/for-hurricane-irma-information-officials-post-on-social-media-1505149661

[30] Luis Marujo, Wang Ling, Isabel Trancoso, Chris Dyer, Alan W Black, Anatole Gershman, David Martins de Matos, Joao Neto, and Jaime Carbonell. 2015. Automatic keyword extraction on twitter. In *IJCNLP*. 637–643.

[31] Olena Medelyan, Eibe Frank, and Ian H Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *EMNLP*. 1318–1327.

[32] Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. *arXiv preprint arXiv:1704.06879* (2017).

[33] Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

[34] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013). arXiv:1301.3781 http://arxiv.org/abs/1301.3781

[35] Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. *proceedings of ACL-08: HLT* (2008), 236–244.

[36] David R. Mortensen, Siddharth Dalmia, and Patrick Littell. 2018. Epitran: Precision G2P for Many Languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)* (7–12), Nicoletta Calzolari (Conference chair), Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga (Eds.). European Language Resources Association (ELRA), Paris, France.

[37] David R. Mortensen, Patrick Littell, Akash Bharadwaj, Kartik Goyal, Chris Dyer, and Lori S. Levin. 2016. PanPhon: A Resource for Mapping IPA Segments to Articulatory Feature Vectors. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. ACL, 3475–3484.

[38] Alexandra Olteanu, Carlos Castillo, Fernando Diaz, and Sarah Vieweg. 2014. CrisisLex: A Lexicon for Collecting and Filtering Microblogged Communications in Crises. In *ICWSM*.

[39] Alexandra Olteanu, Carlos Castillo, Fernando Diaz, and Sarah Vieweg. 2014. CrisisLex: A Lexicon for Collecting and Filtering Microblogged Communications in Crises. In *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014, Ann Arbor, Michigan, USA, June 1-4, 2014*. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM14/paper/view/8091

[40] Alexandra Olteanu, Sarah Vieweg, and Carlos Castillo. 2015. What to Expect When the Unexpected Happens: Social Media Communications Across Crises. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. ACM, New York, NY, USA, 994–1009. https://doi.org/10.1145/2675133.2675242

[41] Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: human language technologies*. 380–390.

[42] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*. 1532–1543. http://www.aclweb.org/anthology/D14-1162

[43] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365* (2018).

[44] Matthew E Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting Contextual Word Embeddings: Architecture and Representation. *arXiv preprint arXiv:1808.08949* (2018).

[45] Maya Rhodan. 2017. 'Please Send Help.' Hurricane Harvey Victims Turn to Twitter and Facebook. *Time* (2017). http://time.com/4921961/hurricane-harvey-twitter-facebook-social-media/

[46] Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 1524–1534.

[47] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory* (2010), 1–20.

[48] Vasile Rus and Mihai Lintean. 2012. A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*. Association for Computational Linguistics, 157–162.

[49] Gerard Salton and Michael J. McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

[50] Axel Schulz, Christian Guckelsberger, and Frederik Janssen. 2017. Semantic Abstraction for generalization of tweet classification: An evaluation of incident-related tweets. *Semantic Web* 8, 3 (2017), 353–372.

[51] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[52] Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 231–235.

[53] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* 28, 1 (1972), 11–21.

[54] Kevin Stowe, Michael J Paul, Martha Palmer, Leysia Palen, and Kenneth Anderson. 2016. Identifying and categorizing disaster-related tweets. In *Proceedings of The Fourth International Workshop on Natural Language Processing for Social Media*. 1–6.

[55] Tai Tran. 2016. Twitter's 10th Birthday: The Power Of Real-Time Content And Recount Of Memorable Tweets. *Forbes* (2016). https://www.forbes.com/sites/taitran/2016/03/21/twitters-10th-birthday-the-power-of-real-time-content-recount-of-memorable-tweets/

[56] Peter D. Turney. 1999. Learning to Extract Keyphrases from Text. *CoRR* cs.LG/0212013 (1999).

[57] Peter D. Turney. 2000. Learning Algorithms for Keyphrase Extraction. *Inf. Retr.* 2, 4 (May 2000), 303–336. https://doi.org/10.1023/A:1009976227802

[58] Sudha Verma, Sarah Vieweg, William J Corvey, Leysia Palen, James H Martin, Martha Palmer, Aaron Schram, and Kenneth Mark Anderson. 2011. Natural Language Processing to the Rescue? Extracting" Situational Awareness" Tweets During Mass Emergency.. In *ICWSM*. Barcelona, 385–392.

[59] Sarah Vieweg, Amanda L Hughes, Kate Starbird, and Leysia Palen. 2010. Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 1079–1088.

[60] Carlos Villegas, Matthew Martinez, and Matthew Krause. 2018. Lessons from Harvey: Crisis Informatics for Urban Resilience. *Rice University Kinder Institute for Urban Research* (2018).

[61] Awen Wen. 2017. How Social Media is Revolutionizing Real-time Information in Six Ways. *Medium* (2017). https://medium.com/@awenz/how-social-media-is-revolutionizing-real-time-information-in-six-ways-335b3a1624ed

[62] Ian H Witten and Olena Medelyan. 2006. Thesaurus based automatic keyphrase indexing. In *Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'06)*. IEEE, 296–297.

[63] Qi Zhang, Yang Wang, Yeyun Gong, and Xuanjing Huang. 2016. Keyphrase extraction using deep recurrent neural networks on Twitter. In *EMNLP*. 836–845.

[64] Shanshan Zhang and Slobodan Vucetic. 2016. Semi-supervised discovery of informative tweets during the emerging disasters. *arXiv preprint arXiv:1610.03750* (2016).

[65] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*. 649–657.

[66] Wayne Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achananuparp, Ee-Peng Lim, and Xiaoming Li. 2011. Topical keyphrase extraction from twitter. In *ACL*. Association for Computational Linguistics, 379–388.

Citation Needed: A Taxonomy and Algorithmic Assessment of Wikipedia's Verifiability

Miriam Redi
Wikimedia Foundation
London, UK

Besnik Fetahu
L3S Research Center
Leibniz University of Hannover

Jonathan Morgan
Wikimedia Foundation
Seattle, WA

Dario Taraborelli
Wikimedia Foundation
San Francisco, CA

ABSTRACT

Wikipedia is playing an increasingly central role on the web, and the policies its contributors follow when sourcing and fact-checking content affect million of readers. Among these core guiding principles, *verifiability* policies have a particularly important role. Verifiability requires that information included in a Wikipedia article be corroborated against *reliable secondary sources*. Because of the manual labor needed to curate Wikipedia at scale, however, its contents do not always evenly comply with these policies. Citations (i.e. reference to external sources) may not conform to verifiability requirements or may be missing altogether, potentially weakening the reliability of specific topic areas of the free encyclopedia. In this paper, we aim to provide an empirical characterization of the reasons *why* and *how* Wikipedia cites external sources to comply with its own verifiability guidelines. First, we construct a *taxonomy of reasons* why inline citations are required, by collecting labeled data from editors of multiple Wikipedia language editions. We then crowdsource a large-scale dataset of Wikipedia sentences annotated with categories derived from this taxonomy. Finally, we design algorithmic models to determine if a statement *requires a citation*, and to predict the *citation reason*. We evaluate the accuracy of such models across different classes of Wikipedia articles of varying quality, and on external datasets of claims annotated for fact-checking purposes.

CCS CONCEPTS

• **Computing methodologies** → *Neural networks*; *Natural language processing*; • **Information systems** → *Crowdsourcing*; • **Human-centered computing** → *Wikis*.

KEYWORDS

Citations; Wikipedia; Crowdsourcing; Neural Networks;

ACM Reference Format:
Miriam Redi, Besnik Fetahu, Jonathan Morgan, and Dario Taraborelli. 2019. Citation Needed: A Taxonomy and Algorithmic Assessment of Wikipedia's Verifiability. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313618

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313618

1 INTRODUCTION

Wikipedia is playing an increasingly important role as a "neutral" arbiter of the factual accuracy of information published in the web. Search engines like Google systematically pull content from Wikipedia and display it alongside search results [38], while large social platforms have started experimenting with links to Wikipedia articles, in an effort to tackle the spread of disinformation [37].

Research on the accuracy of information available on Wikipedia suggests that despite its radical openness—anyone can edit most articles, often without having an account —the confidence that other platforms place in the factual accuracy of Wikipedia is largely justified. Multiple studies have shown that Wikipedia's content across topics is of a generally high quality[21, 34], that the vast majority of vandalism contributions are quickly corrected [20, 33, 42], and that Wikipedia's decentralizedprocess for vetting information works effectively even under conditions where reliable information is hard to come by,such as in breaking news events [27].

Wikipedia's editor communities govern themselves through a set of collaboratively-created policies and guidelines [6, 19]. Among those, the Verifiability policy[1] is a key mechanism that allows Wikipedia to maintain its quality. Verifiability mandates that, in principle, "all material in Wikipedia... articles must be verifiable" and attributed to reliable secondary sources, ideally through inline citations, and that unsourced material should be removed or challenged with a *{citation needed}* flag.

While the role citations serve to meet this requirement is straightforward, the process by which editors determine which claims require citations, and why those claims need citations, are less well understood. In reality, almost all Wikipedia articles contain at least some unverified claims, and while high quality articles may cite hundreds of sources, recent estimates suggest that the proportion of articles with few or no references can be substantial [35]. While as of February 2019 there exists more than 350,000 articles with one or more *{citation needed}* flag, we might be missing many more.

Furthermore, previous research suggests that editor citation practices are not systematic, but often contextual and ad hoc. Forte et al. [17] demonstrated that Wikipedia editors add citations primarily for the purposes of "information fortification": adding citations to protect information that they believe may be removed by other editors. Chen et al. [10] found evidence that editors often add citations to existing statements relatively late in an article's lifecycle. We submit that by understanding the reasons why editors prioritize

[1] https://en.wikipedia.org/wiki/Wikipedia:Verifiability

adding citations to some statements over others we can support the development of systems to scale volunteer-driven verification and fact-checking, potentially increasing Wikipedia's long-term reliability and making it more robust against information quality degradation and coordinated disinformation campaigns.

Through a combination of qualitative and quantitative methods, we conduct a systematic assessment of the application of Wikipedia's verifiability policies at scale. We explore this problem throughout this paper by focusing on two tasks:

(1) Citation Need: identifying *which* statements need a citation.
(2) Citation Reason: identifying *why* a citation is needed.

By characterizing qualitatively and algorithmically these two tasks, this paper makes the following contributions:

- We develop a Citation Reason Taxonomy[2] describing reasons why individual sentences in Wikipedia articles require citations, based on verifiability policies as well as labels collected from editors of the English, French, and Italian Wikipedia (See Sec. 3).
- We assess the validity of this taxonomy and the corresponding labels through a crowdsourcing experiment, as shown in Sec. 4. We find that sentences needing citations in Wikipedia are more likely to be historical facts, statistics or direct/reported speech. We publicly release this data as a Citation Reason corpus.
- We train a deep learning model to perform the two tasks, as shown in Secc. 5 and 6. We demonstrate the high accuracy (F1=0.9) and generalizability of the Citation Need model, explaining its predictions by inspecting the network's attention weights.

These contributions open a number of further directions, both theoretical and practical, that go beyond Wikipedia and that we discuss in Section 7.

2 RELATED WORK

The contributions described in this paper build on three distinct bodies of work: crowdsourcing studies comparing the judgments of domain experts and non-experts, machine-assisted citation recommendations on Wikipedia, and automated detection and verification of factual claims in political debates.

Crowdsourcing Judgments from Non-Experts. Training machine learning models to perform the citation need and citation reason tasks requires large-scale data annotations. While generating data for the first task necessarily requires expert knowledge (based on understanding of policies), we posit that defining the *reasons* why a citation that has already been deemed appropriate is needed can be effectively performed by people without domain expertise, such as crowdworkers.

Obtaining consistent and accurate judgments from untrained crowdworkers can be a challenge, particularly for tasks that require contextual information or domain knowledge. However, a study led by Kittur [31] found that crowdworkers were able to provide article quality assessments that mirrored assessments made by Wikipedians by providing clear definitions and instructions, and by focusing the crowdworkers attention on the aspects of the article that provided relevant evaluation criteria. Similarly, Sen et al. [46] demonstrated that crowdworkers are able to provide semantic relatedness judgments as scholars when presented with keywords related to general knowledge categories.

Our labeling approach aims to assess whether crowdworkers and experts (Wikipedians) agree in their understanding of verifiability policies—specifically, whether non-experts can provide reliable judgments on the reasons why individual statements need citations.

Recommending Sources. Our work is related to a body of bibliometrics works on citation analysis in academic texts. These include unsupervised methods for citation recommendation in articles [24], and supervised models to identify the purpose of citations in academic manuscripts[1]. Our work explores similar problems in the different domain of Wikipedia articles: while scholarly literature cites work for different purposes[1] to support original research, the aim of Wikipedia's citations is to verify existing knowledge.

Previous work on the task of source recommendation in Wikipedia has focused on cases where statements are marked with a *citation needed* tag [14–16, 44]. Sauper et al. [14, 44] focused on adding missing information in Wikipedia articles from external sources like news, where the corresponding Wikipedia entity is a salient concept. In another study [16], Fetahu et al. used existing statements that have either an outdated citation or *citation needed* tag to query for relevant citations in a news corpus. Finally, the authors in [15], attempted to determine the *citation span*—that is, which parts of the paragraph are covered by the citation—for any given existing citation in a Wikipedia article and the corresponding paragraph in which it is cited.

None of these studies provides methods to determine whether a given (untagged) statement *should* have a citation and *why* based on the citation guidelines of Wikipedia.

Fact Checking and Verification. Automated verification and fact-checking efforts are also relevant to our task of computationally understanding verifiability on Wikipedia. Fact checking is the process of assessing the veracity of factual claims [45]. Long et al. [36] propose TruthTeller computes annotation types for all verbs, nouns, and adjectives, which are later used to predict the truth of a clause or a predicate. Stanovsky et al. [47] build upon the output rules from TruthTeller and use those as features in a supervised model to predict the factuality label of a predicate. Chung and Kim [13] assess source credibility through a questionnaire and a set of measures (e.g. informativeness, diversity of opinions, etc.). The largest fact extraction and verification dataset FEVER [49] constructs pairs of factual snippets and paragraphs from Wikipedia articles which serve as evidence for those factual snippets. However,these approaches cannot be applied in our case because they make the assumption that any provided statement is of factual nature.

Research on the automated fact detectors in political discourse [23, 32, 39] is the work in this domain that is most closely related to ours. While these efforts have demonstrated the ability to effectively detect the presence of facts to be checked, they focus on the political discourse only, and they do not provide explanation for the models' prediction. In our work, we consider a wide variety of topics—any topic covered in Wikipedia—and design models able to not only detect claims, but also explain the reasons why those claims require citations.

[2]We use here the term "taxonomy" in this context as a synonym of *coding scheme.*

Table 1: A taxonomy of Wikipedia verifiability: set of reasons for adding and not adding a citation. This taxonomy is the result of a qualitative analysis of various sources of information regarding Wikipedia editors' referencing behavior.

Reasons why citations are needed	
Quotation	The statement appears to be a direct quotation or close paraphrase of a source
Statistics	The statement contains statistics or data
Controversial	The statement contains surprising or potentially controversial claims - e.g. a conspiracy theory
Opinion	The statement contains claims about a person's subjective opinion or idea about something
Private Life	The statement contains claims about a person's private life - e.g. date of birth, relationship status.
Scientific	The statement contains technical or scientific claims
Historical	The statement contains claims about general or historical facts that are not common knowledge
Other	The statement requires a citation for reasons not listed above (please describe your reason in a sentence or two)
Reasons why citations are not needed	
Common Knowledge	The statement only contains common knowledge - e.g. established historical or observable facts
Main Section	The statement is in the lead section and its content is referenced elsewhere in the article
Plot	The statement is about a plot or character of a book/movie that is the main subject of the article
Already Cited	The statement only contains claims that have been referenced elsewhere in the paragraph or article
Other	The statement does not require a citation for reasons not listed above (please describe your reason in a sentence or two)

3 A TAXONOMY OF CITATION REASONS

To train models for the Citation Need and Citation Reason tasks, we need to develop a systematic way to operationalize the notion of verifiability in the context of Wikipedia. There is currently no single, definitive taxonomy of reasons why a particular statement in Wikipedia should, or should not, have a supporting inline citation. We drew on several data sources to develop such a taxonomy using an inductive, mixed-methods approach.

Analyzing Citation Needed Templates. We first analyzed reasons Wikipedia editors provide when requesting an inline citation. Whenever an editor adds a *citation needed* tag to a claim that they believe should be attributed to an external source, they have the option to specify a reason via a free-form text field. We extracted the text of this field from more than 200,000 *citation needed* tags added by English Wikipedia editors and converted it into a numerical feature by averaging the vector representations of each sentence word, using Fasttext [8]. We then used k-means to cluster the resulting features into 10 clusters (choosing the number of clusters with the elbow method [28]). Each cluster contains groups of consistent reasons why editors requested a citation. By analyzing these clusters, we see that the usage of the "reason" field associated with the *citation needed* tag does not consistently specify the reason why these tags are added. Instead, it is often used to provide other types of contextual information—for example, to flag broken links or unreliable sources, to specify the date when the tag was added, or to provide very general explanations for the edit. Therefore, we did not use this data to develop our taxonomy.

Analyzing Wikipedia Citation Policies. As a next step, we analyzed documentation developed by the editor community to describe rules and norms to be followed when adding citations. We examined documentation pages in the English, French, and Italian language editions. Since each Wikipedia language edition has its own citation policies, we narrowed down the set of documents to analyze by identifying all subsidiary rules, style guides, and lists of best practices linked from the main Verifiability policy page, which exists across all three languages. Although these documents slightly differ across languages, they can be summarized into 28 distinct rules [3]. Rules that we identified across these pages include a variety of types of claims that should usually or always be referenced to a source, such as claims of scientific facts, or any claim that is likely to be unexpected or counter-intuitive. These documentation pages also contain important guidance on circumstances under which it is appropriate to *not* include an inline citation. For example, when the same claim is made in the lead section as well as in the main body of the article, it is standard practice to leave first instance of the claim unsourced.

Asking Expert Wikipedians. To expand our Citation Reason Taxonomy, we asked a group of 36 Wikipedia editors from all three language communities (18 from English Wikipedia, 7 from French Wikipedia, and 11 from Italian Wikipedia) to annotate citations with reasons. Our experiment was as follows: we extracted sentences with and without citations from a set of Featured Articles and removed the citation metadata from each sentence. Using WikiLabels[4], an open-source tool designed to collect labeled data from Wikipedia contributors, we showed our annotators the original article with all citation markers removed and with a random selection of sentences highlighted. Editors were then asked to decide whether the sentence needed a citation or not (Citation Need task), and to specify a reason for their choices (Citation Reason task) in a free-text form. We clustered the resulting answers using the same methodology as above, and used these clusters to identify additional reasons for citing claims.

[3]The full guideline summary and the cluster analysis can be found here: https://figshare.com/articles/Summaries_of_Policies_and_Rules_for_Adding_Citations_to_Wikipedia/7751027

[4]https://meta.wikimedia.org/wiki/Wiki_labels

Our final set of 13 discrete reasons (8 for adding and 5 for not adding) is presented in Table 1. In Section 4, we evaluate the accuracy of this taxonomy and use it to label a large number of sentences with citation-needed reasons.

4 DATASETS

In this Section, we show how we collected data to train models able to perform the Citation Need task, for which we need sentences with binary citation/no-citation labels, and the Citation Reason task, for which we need sentences labeled with one of the reason category from our taxonomy.

4.1 Citation Need Dataset

Previous research [17] suggests that the decision of whether or not to add a citation, or a *citation needed* tag, to a claim in a Wikipedia article can be highly contextual, and that doing so reliably requires a background in editing Wikipedia and potentially domain knowledge as well. Therefore, to collect data for the Citation Need task we resort to expert judgments by Wikipedia editors.

Wikipedia articles are rated and ranked into ordinal quality classes, from "stub" (very short articles) to "Featured". Featured Articles[5] are those articles that are deemed as the highest quality by Wikipedia editors based on a multidimensional quality assessment scale [6]. One of the criteria used in assessing Featured Articles is that the information in the article is *well-researched*.[7] This criterion suggests that Featured Articles are more likely to consistently reflect best practices for when and why to add citations than lower-quality articles. The presence of *citation needed* tags is an additional signal we can use, as it indicates that at least one editor believed that a sentence requires further verification.

We created three distinct datasets to train models predicting if a statement requires a citation or not[8]. Each dataset consists of: (i) *positive instances* and (ii) *negative instances*. Statements with an *inline citation* are considered as *positives*, and statements *without an inline citation* and that appear in a *paragraph with no citation* are considered as *negatives*.

Featured – FA. From the set of 5,260 *Featured* Wikipedia articles we randomly sampled 10,000 positive instances, and equal number of negative instances.

Low Quality (citation needed) – LQN. In this dataset, we sample for statements from the 26,140 articles where at least one of the statements contains a *citation needed* tag. The *positive instances* consist solely of statements with *citation needed* tags.

Random – RND. In the random dataset, we sample for a total of 20,0000 positive and negative instances from all Wikipedia articles. This provides an overview of how editors cite across articles of varying quality and topics.

[5]https://en.wikipedia.org/wiki/Wikipedia:Featured_articles
[6]https://en.wikipedia.org/wiki/Wikipedia:WikiProject_Wikipedia/Assessment#Quality_scale
[7]"[the article provides] a thorough and representative survey of the relevant literature; claims are verifiable against high-quality reliable sources and are supported by inline citations where appropriate." https://en.wikipedia.org/wiki/Wikipedia:Featured_article_criteria
[8]Unless otherwise specified, all data in the paper is from English Wikipedia

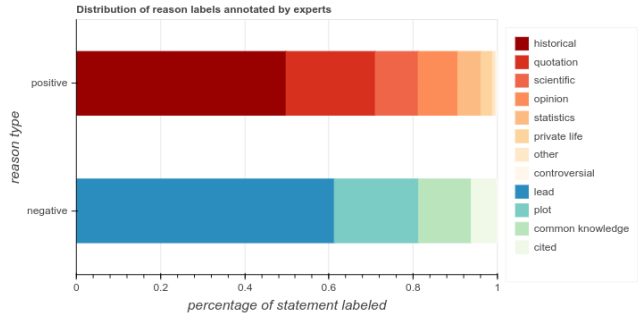


Figure 1: Distribution of labels assigned by Wikipedia Editors through the Wikilabels platform to characterize the reason why statements need citations.

4.2 Citation Reason Dataset

To train a model for the Citation Reason task,we designed a labeling task for Wikipedia editors in which they are asked to annotate Wikipedia sentences with both a binary judgment (citation needed/not needed) and the reason for that judgment using our Citation Reason Taxonomy. We used these annotations as ground truth for a larger-scale crowdsourcing experiment, where we

asked micro-workers to select reasons for why *positive* sentences require citations. We compared how well crowdworkers' judgments matched the Wikipedia editor judgments. Finally, we collected enough annotations to train machine learning algorithms.

4.2.1 Round 1: Collecting Data from Wikipedia Editors. To collect "expert" annotations from Wikipedia editors on why sentences need citations, we proceeded as follows.

Interface Design. We created a modified version of the free-text WikiLabels labeling task described in Section 3. We selected random sentences from Featured Articles, and removed citation markers when present. We presented the participants with the unsourced sentence highlighted in an article and asked them to label the sentence as needing an inline citation or not, and to specify a reason for their choice using a drop-down menu pre-filled with categories from our taxonomy. We recruited participants through mailing lists, social media and the English Wikipedia's Village pump (the general discussion forum of the English Wikipedia volunteer community).

Results. We collected a total of 502 labels from 35 English Wikipedia editors. Of the valid[9] annotated sentences, 255 were labeled as needing a citation (*positive*), and 80 as not needing a citation. Fig. 1 shows the breakdown of results by selected reason.

We found that the reason given for roughly 80% of the *positive* sentences is that they are "historical facts", "direct quotations", or "scientific facts". Furthermore, we observed that only a small percentage of participants selected the "Other" option, which suggests that our Citation Reason Taxonomy is robust and makes sense to editors, even when they are asked to provide these reasons outside of their familiar editing context.

[9]Due to a bug in the system, not all responses were correctly recorded.

Table 2: Example of sentences annotated with different categories by Wikipedia experts and Mechanical Turk contributors.

Non-Expert judgment	Expert judgment	Sentence extracted from Wikipedia Featured Article
historical	*quotation*	He argued that a small number of Frenchmen could successfully invade New Spain by allying themselves with some of the more than 15,000 Native Americans who were angry over Spanish enslavement
life	*historical*	Actor Hugh Jackman is also a fan of the club, having been taken to Carrow Road as a child by his English mother, though he turned down an opportunity to become an investor in the club in 2010
statistics	*historical*	The act, authored by Ohio senator and former Treasury secretary John Sherman, forced the Treasury to increase the amount of silver purchased to 4,500,000 troy ounces (140,000 kg) each month
quotation	*historical*	"This stroke", said Clark, "will nearly put an end to the Indian War." Clark prepared for a Detroit campaign in 1779 and again in 1780, but each time called off the expedition because of insufficient men and supplies

4.2.2 Round 2: Collecting Data from non-Experts. We adapted the task in Round 1 to collect data from crowdworkers to train a Citation Reason model.

Task adaptation. Adapting classification tasks that assume a degree of domain expertise to a crowdwork setting, where such expertise cannot be relied upon, can create challenges for both reliability and quality control. Crowdworkers and domain experts may disagree on classification tasks that require special knowledge [46]. However, Zhang et al.[51] found that non-expert judgments about the characteristics of statements in news articles, such as whether a claim was well supported by the evidence provided, showed high inter-annotator agreement and high correlation with expert judgments. In the context of our study, this suggests that crowdworkers may find it relatively easier to provide reasons for citations than to decide which sentences require them in the first place. Therefore, we simplified the annotation task for crowdworkersto increase the likelihood of eliciting high-quality judgments from non-experts. While Wikipedia editors were asked to both identify whether a sentence required citation and provide a reason, crowdworkers were only asked to provide a reason why a citation was needed.

Experimental Setup. We used Amazon Mechanical Turk for this annotation task. For each task, workers were shown one of 166 sentences that had been assigned citation reason categories by editors in round 1. Workers were informed that the sentence came from a Wikipedia article and that in the original article it contained a citation to an external source. Like editors in the first experiment, crowdworkers were instructed to select the most appropriate category from the eight citation reasons 1. Each sentence was classified by 3 workers, for a total of 498 judgments. For quality control purposes, only crowdworkers who had a history of reliable annotation behavior were allowed to perform the task. Average agreement between workers was 0.63% (vs random 1/8 =0.125).

4.2.3 Comparing Expert and Non-Expert annotations. The distribution of citation reasons provided by crowdworkers is shown in Fig. 2. The overall proportions are similar to that provided by Wikipedia editors in Round 1 (See Fig. 1). Furthermore, the confusion matrix in Fig. 3 indicates that crowdworkers and Wikipedia editors had high agreement on four of the five most prevalent reasons: *historical, quotation, scientific* and *statistics*. Among these five categories, non-experts and experts disagreed the most on *opinion*. One potential reason for this disagreement is that identifying whether a statement

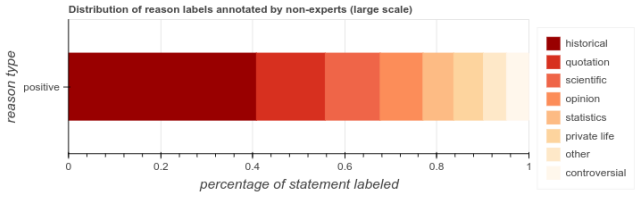


Figure 2: Citation reason distribution from the small-scale (166 sentences) crowdsourcing experiment.

is an opinion may require additional context (i.e. the contents of the preceding sentences, which crowdworkers were not shown).

The confusion matrix in Fig. 3) shows the percentage of different kinds of disagreement—for example, that crowdworkers frequently disagreed with editors over the categorization of statements that contain *"claims about general or historical facts."* To further investigate these results, we manually inspected a set of individual sentences with higher disagreement between the two groups. We found that in these cases the reason for the disagreement was due to a sentence containing multiple types of claims, e.g. a historical claim and a direct quote (see Table 2). This suggests that in many cases these disagreements were not due to lower quality judgments on the part of the crowdworkers, but instead due to ambiguities in the task instructions and labeling interface.

4.2.4 The Citation Reason Corpus: Collecting Large-scale Data. Having verified the agreement between Wikipedia editors and crowdworkers, we can now reliably collect larger scale data to train a Citation Reason model. To this end, we sampled 4,000 sentences that contain citations from Featured articles, and asked crowdworkers to annotate them with the same setup described above (see Sec 4.2.2). The distribution of the resulting judgments is similar to Fig. 2: as in Round 1, we found that the top categories are the *scientific*,*quotation* and *historical* reasons.[10]

[10]Our *Citation Reason corpus* is publicly available here: https://figshare.com/articles/Citation_Reason_Dataset/7756226.

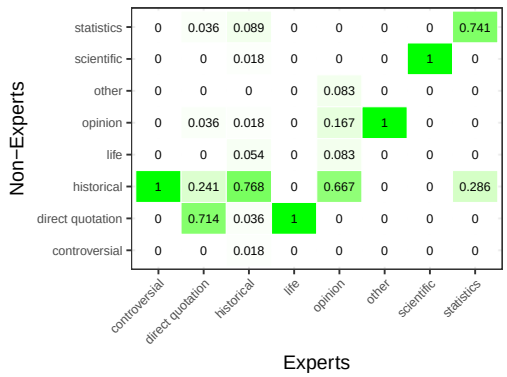


Figure 3: Confusion matrix indicating the agreement between Mechanical Turk workers ("non-experts") and Wikipedia editors ("experts"). The darker the square, the higher the percent agreement between the two groups

5 A CITATION NEED MODEL

We design a classifier to detect when a statement needs a citation. We proposea neural based Recurrent Neural Network (RNN) approach with varying representations of a statement, and compare it with a baseline feature-based model.

5.1 Neural Based Citation Need Approach

We propose a neural based model, which uses a recurrent neural network (RNN) with GRU cells[11] to encode statements for classification. We distinguish between two main modes of statement encoding: (i) vanilla RNN, fed with 2 different representations of a sentence (words and section information, indicated with RNN^{w} and RNN^{+S}), and (ii) RNN with global attention RNN_a (with similar representation).

5.1.1 Statement Representation. For a given Wikipedia sentence, for which we want to determine its citation need, we consider the words in the statement and the section the statement occurs in. To feed the network with this information, we transform sentence words and section information into features, or representations. Through the word representation we aim at capturing cue words or phrases that are indicators of a statement requiring a citation. Section representation, on the other hand, allows us to encode information that will play a crucial role in determining the Citation Reason later on.

Word Representation. We represent a statement as a sequence of words $s = (w_1, \ldots, w_n)$. We use GloVe pre-trained word embeddings [40] to represent the words in s. Unknown words are randomly initialized in the word embedding matrix $W_{glove} \in \mathbb{R}^{k \times 100}$, where k is the number of words in the embedding matrix.

Section Representation. The section in which the statement occurs in a Wikipedia article is highly important. The guidelines for *inline citations* suggest that when a statement is in the *lead section*, and that is referenced elsewhere in the article, editors should avoid multiple references [11]. Additionally, since sections can be seen as a topically coherent group of information, the reasons for citation will vary across sections (e.g. *"Early Life"*). We train the *section embedding* matrix $W_S \in \mathbb{R}^{l \times 100}$, and use it in combination with W_{glove}, where l is the number of sections in our dataset.

5.1.2 Statement Classification. We use 2 types of Recurrent Neural Networks to classify the sentence representations.

Vanilla RNNs. RNNs encode the individual words into a hidden state $h_t = f(w_t, h_{t-1})$, where f represents GRU cells [11]. The encoding of an input sequence from s is dependent on the previous hidden state. This dependency based on f determines how much information from the previous hidden state is passed onto h_t. For instance, in case of GRUs, h_t is encoded as following:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (1)$$

where, the function z_t and $\tilde{h}_t$ are computed as following:

$$z_t = \sigma(W_z w_t + U_z h_{t-1} + b_z) \quad (2)$$

$$\tilde{h}_t = \tanh\left(W_h w_t + r_t \odot (U_h h_{t-1} + b_h)\right) \quad (3)$$

$$r_t = \sigma(W_r w_t + U_r h_{t-1} + b_r) \quad (4)$$

The RNN encoding allows us to capture the presence of words or phrases that incur the need of a citation. Additionally, words that do not contribute in improving the classification accuracy are captured through the model parameters in function r_t, allowing the model to ignore information coming from them.

RNN with Global Attention – RNN_a. As we will see later in the evaluation results, the disadvantage of vanilla RNNs is that when used for classification tasks, the classification is done solely based on the last hidden state h_N. For long statements this can be problematic as the hidden states, respectively the weights are highly compressed across all states and thus cannot capture the importance of the individual words in a statement.

Attention mechanisms [4] on the other hand have proven to be successful in circumventing this problem. The main difference with standard training of RNN models is that all the hidden states are taken into account to derive a *context vector*, where different states contribute with varying weights, or known with *attention weights* in generating such a vector.

Fig. 4 shows the RNN_a^{+S} model we use to classify a statement. We encode the statement through a bidirectional RNN based on its word representation, while concurrently a separate RNN encodes the section representation. Since not all words are equally important in determining if a statement requires a citation, we compute the *attention weights*, which allow us to compute a *weighted representation* of the statement based on the hidden states (as computed by the GRU cells) and the *attention weights*. Finally, we *concatenate* the weighted representation of the statement based on its words and section, and push it through a dense layer for classification.

The vanilla RNN, and the varying representations can easily be understood by referring to Fig. 4, by simply omitting either the section representation or the attention layer.

5.1.3 Experimental Setup. We use Keras [12] with Tensorflow as backend for training our RNN models. We train for 10 epochs (since the loss value converges), and we set the batch size to 100. We use Adam [29] for optimization, and optimize for *accuracy*. We set the number of dimensions to 100 for hidden states h, which represent the words or the section information.

[11] https://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style/Lead_section

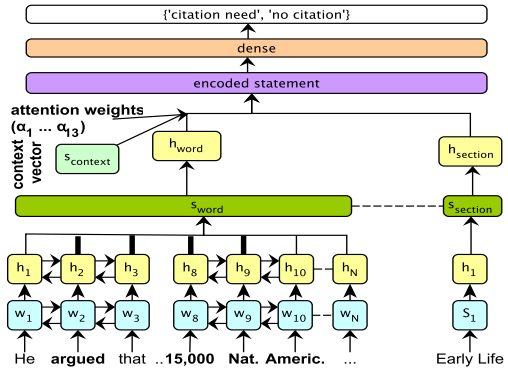


Figure 4: Citation Need model with RNN and global attention, using both word and section representations.

Table 3: Point-Biserial Correlation Coefficient between citation need labels and individual feature values

FA		LQN		RND	
Section	-0.621	*underline*	0.054	*say*	0.084
say	0.107	*say*	0.0546	*underline*	0.0842
underline	0.107	*believe*	0.042	*Section*	-0.072
realize	0.068	*disagree*	0.040	*report*	0.062
suggest	0.068	*claim*	0.039	*tell*	0.062

We train the models with 50% of the data and evaluate on the remaining portion of statements.

5.2 Feature-based Baselines

As we show in Table 1, where we extract the reasons why statements need a citation based on expert annotators, the most common reasons (e.g. *statistics*, *historical*) can be tracked in terms of specific *language frames* and *vocabulary use* (in the case of *scientific* claims). Thus, we propose two baselines, which capture this intuition of language frames and vocabulary. From the proposed feature set, we train standard supervised models and show their performance in determining if a statement requires a citation.

5.2.1 Dictionary-Based Baseline – Dict. In the first baseline, we consider two main groups of features. First, we rely on a set of lexical dictionaries that aim in capturing words or phrases that indicate an activity, which when present in a statement would imply the necessity of a citation in such cases. We represent each statement as a feature vector where each element correspond to the frequency of a dictionary term in the statement.

Factive Verbs. The presence of *factive verbs* [30] in a statement presumes the truthfulness of information therein.

Assertive Verbs. In this case, assertive verbs [25] operate in two dimensions. First, they indicate an assertion, and second, depending on the verb, the credibility or certainty of a proposition will vary (e.g. *"suggest"* vs. *"insist"*). Intuitively, *opinions* in Wikipedia fall in this definition, and thus, the presence of such verbs will be an indicator of opinions needing a citation.

Entailment Verbs. As the name suggests, different verbs entail each other, e.g. *"refrain"* vs. *"hesitate"* [5, 26]. They are particularly interesting as the context in which they are used may indicate cases of *controversy*, where depending on the choice of verbs, the framing of a statement will vary significantly as shown above. In such cases, Wikipedia guidelines strongly suggest the use of citations.

Stylistic Features. Finally, we use the frequency of the different POS tags in a statement. POS tags have been successfully used to capture linguistic styles in different genres [41]. For the different citation reasons (e.g. *historical*, *scientific*), we expect to see a variation in the distribution of the POS tags.

5.2.2 Word Vector-Based Baseline – WV. Word representations have shown great ability to capture word contextual information, and their use in text classification tasks has proven to be highly effective [22]. In this baseline, our intuition is that we represent each statement by averaging the individual word representations from a pre-trained word embeddings [40]. Through this baseline we aim at addressing the cases, where the *vocabulary use* is a high indicator of statements needing a citation, e.g. *scientific* statements.

5.2.3 Feature Classifier. We use a Random Forest Classifier [9] to learn Citation Need models based on these features. To tune the parameters (depth and number of trees), similar to the main deep learning models, we split the data into train, test and validation (respectively 50%,30% and 20% of the corpus). We perform cross-validation on the training and test set, and report accuracy results in terms of F1 on the validation set.

5.3 Citation Need Indicators

We analyze here how algorithms associate specific sentence features with the sentence's need for citations.

5.3.1 Most Correlated Features. To understand which sentence features are more related to the need for citation, we compute the Point Biserial Correlation coefficient [48] between the binary citation/no-citation labels and the frequency of each word in the baseline dictionary of each sentence, as well as the *Section* feature.

We report in Table 3 the top-5 most correlated features for each dataset. In featured articles, the most useful features to detect statements needing citation is the position of the sentence in the article, i.e. whether the sentence lies in the lead section of the article. This might be due to the fact that FA are the result of a rigorous formal process of iterative improvement and assessment according to established rubrics [50], and tend to follow the best practices to write the lead section, i.e. including general overview statements, and claims that are referenced and further verified in the article body. In the *LQN* dataset we consider as "positives" those sentences tagged as *Citation Needed*. Depending on the article, these tags can appear in the lead section too, thus explaining why the *Section* feature is not discriminative at all for this group of sentences. Overall, we see that *report verbs*, such as *say, underline, claim* are high indicators of the sentence's need for citations.

5.3.2 Results from Attention Mechanisms in Deep Learning. Fig. 5 shows a sample of positive statements from Featured Articles

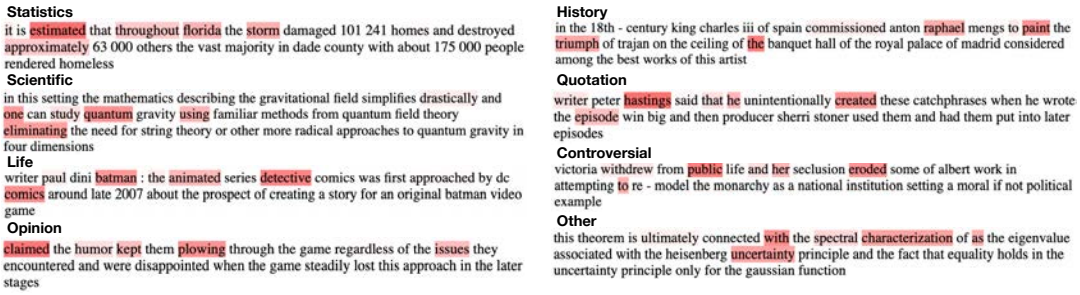


Figure 5: Attention mechanism for RNN_a^{+S} visualizing the focus on specific words for the different citation reasons. It is evident that the model is able to capture patterns similar to those of human annotators (e.g. *"claimed"* in the case of *opinion*.)

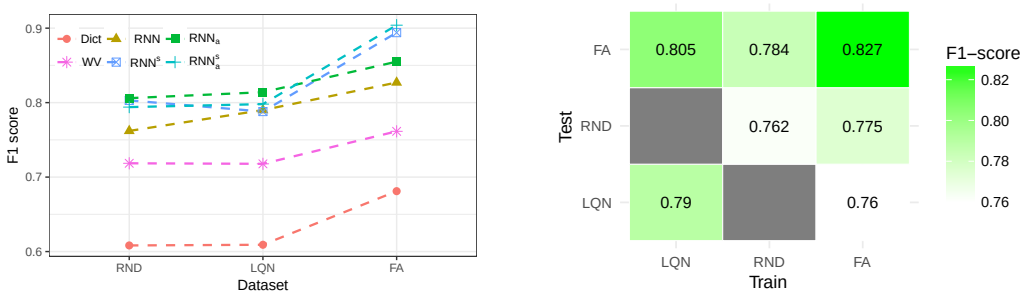


Figure 6: (a) F1 score for the different Citation Need detection models across the different dataset. (b) Confusion Matrix visualizing the accuracy (F1 score) of a Citation Need model trained on Featured Articles and tested on other datasets, showing the generalizability of a model trained on Featured Articles only.

Table 4: Accuracy (F1 score) of Citation Need classification models on Featured Article vs individual expert editor annotations on the same set of Featured Articles.

	no citation	citation	average
individual editor	0.608	**0.978**	0.766
RNN_a^{+S}	**0.902**	0.905	**0.904**

grouped by citation reason. The words are highlighted based on their *attention weight* from the RNN_a^{+S} model. The highlighted words show very promising directions. It is evident that the RNN_a^{+S} model attends with high weights words that are highly intuitive even for human annotators. For instance, if we consider the *opinion* citation reason, the highest weight is assigned to the word *"claimed"*. This is case is particularly interesting as it capture the *reporting verbs* [43] (e.g. *"claim"*) which are common in opinions. In the other citation reasons, we note the *statistics* reason, where similarly, here too, the most important words are again verbs that are often used in reporting numbers. For statements that are *controversial*, the highest attention is assigned to words that are often used in a negative context, e.g. *"erode"*. However, here it is interesting, that the word *"erode"* is followed by context words such as *"public"* and *"withdrew"*. From the other cases, we see that the attention mechanism focuses on domain-specific words, e.g *scientific* citation reason.

5.4 Evaluating the Citation Need model

In this section, we focus on assessing the performance of our model at performing the Citation Need task, its generalizability, and how its output compares with the accuracy of human judgments.

5.4.1 Can an Algorithm Detect Statements in Need of a Citation? We report the classification performance of models and baselines on different datasets in Fig. 6.

Given that they are highly curated, sentences from Featured Articles are much easier to classify than sentences from random articles: the most accurate version of each model is indeed the one trained on the Featured Article dataset.

The proposed RNN models outperform the featured-based baselines by a large margin. We observe that adding attention information to a traditional RNN with GRU cells boosts performances by 3-5%. As expected from the correlation results, the position of the sentence in an article, i.e. whether the sentence is in the lead section, helps classifying Citation Need in Featured Articles only.

5.4.2 Does the Algorithm Generalize? To test the generalizability of one the most accurate models, the RNN Citation Need detection model trained on Featured Articles, we use it to classify statements from the *LQN* and the *RND* datasets, and compute the F1 score over such cross-dataset prediction. The cross-dataset prediction reaches a reasonable accuracy, in line with the performances models trained and tested on the other two noisier datasets. Furthermore, we test the performances of our RNN_a model on 2 external datasets: the

Table 5: Citation reason prediction based on a pre-trained RNN_a^{+S} model on the *FA* dataset, and a RNN_a^{+S} which we train only on the Citation Reason dataset.

	pre-trained			*no pre-training*		
	P	R	F1	P	R	F1
direct quotation	**0.44**	**0.65**	**0.52**	0.43	0.46	0.45
statistics	0.20	**0.20**	**0.20**	**0.28**	0.15	0.19
controversial	**0.12**	**0.02**	**0.04**	0.04	0.01	0.02
opinion	**0.20**	0.12	0.15	0.19	0.12	0.15
life	0.13	0.06	0.09	**0.30**	0.06	**0.10**
scientific	**0.62**	0.56	**0.59**	0.54	**0.58**	**0.56**
historical	**0.56**	0.67	0.61	0.54	**0.74**	**0.62**
other	0.13	0.05	0.07	**0.14**	**0.08**	**0.10**
avg.	0.30	**0.29**	**0.28**	**0.31**	0.28	0.27

claim dataset from Konstantinovskiy et al. [32], and the CLEF2018 Check-Worthiness task dataset [39]. Both datasets are made of sentences extracted from political debates in UK and US TV-shows, labeled as positives if they contain facts that need to be verified by fact-checkers, or as negative otherwise. Wikipedia's literary form is completely different from the political debate genre. Therefore, our model trained on Wikipedia sentences, cannot reliably detect claims in the fact-checking datasets above: most of the sentences from these datasets are outside our training data, and therefore the model tends to label all those as negatives.

5.4.3 Can the Algorithm Match Individual Human Accuracy? Our Citation Need model performs better than individual Wikipedia editors under some conditions. Specifically, in our first round of expert citation labeling (Section 3 above), we observed that when presented with sentences from Featured Articles in the WikiLabels interface, editors were able to identify claims that already had a citation in Wikipedia with a high degree of accuracy (see Table 4), but they tended to *over-label*, leading to a high false positive rate and lower accuracy overall compared to our model. There are several potential reasons for this. First, the editorial decision about whether to source a particular claim is, especially in the case of Featured Articles, an iterative, deliberate, and consensus-based process involving multiple editors. No single editor vets all the claims in the article, or decides which external sources to cite for those claims. Furthermore, the decisions to add citations are often discussed at length during the FA promotion process, and the editors involved in writing and maintaining featured articles often have subject matter expertise or abiding interest in the article topic, and knowledge of topic-specific citation norms and guidelines [18]. By training on the entire corpus of Featured Articles, our model has the benefit of the aggregate of hundreds or thousands of editors' judgments of when (not) to cite across a range of topics, and therefore may be better than any individual editor at rapidly identifying general lexical cues associated with "common knowledge" and other statement characteristics that indicate citations are not necessary.

6 A CITATION REASON MODEL

In this Section, we analyze the *Citation Reason Corpus* collected in Sec. 4, and fine-tune the Citation Need model to detect reasons why statements need citations.

6.1 Distribution of Citation Reasons by Topic

Understanding if Wikipedia topics or article sections have different sourcing requirements may help contributors better focus their efforts. To start answering this question, we analyze citation reasons as a function of the article topic and the section in which the sentence occurs. We rely on DBpedia [3] to associate articles to topics and we show in Table 6 the most topics and article sections associated with each citation reason. We note that the distribution of citation reasons is quite intuitive, both across types and sections. For instance, *"direct quotation"* is most prominent in section *Reception* (the leading section), which is intuitive, where the statements mostly reflect how certain *"Athlete"*, *"OfficeHolders"* have expressed themselves about a certain event. Similarly, we see for *"historical"* and *"controversial"* the most prominent section is *History*, whereas in terms of most prominent article types, we see that *"MilitaryConflict"* types have the highest proportion of statements.

While the distribution of citation reasons is quite intuitive across types and sections, we find this as an important aspect that can be leveraged to perform targeted sampling of statements (from specific sections or types) which may fall into the respective citation reasons s.t we can have even distribution statements across these categories.

6.2 Evaluating the Citation Reason model

To perform the Citation Reason task, we build upon the pre-trained model RNN_a^{+S} in Fig. 4. We modify the RNN_a^{+S} model by replacing the dense layer such that we can accommodate all the eight citation reason classes, and use a *softmax* function for classification.

The rationale behind the use of the pre-trained RNN_a^{+S} model is that by using the much larger training statements from the binary datasets, we are able to adjust the model's weights to provide a better generalization for the more fine-grained citation reason classification. An additional advantage of using the model with the pre-trained weights is that in this way we can retain a large portion of the contextual information from the statement representation, that is, the context in which the words appear for statement requiring a citation.

The last precaution we take in adjusting the RNN_a^{+S} for Citation Reason classification is that we ensure that the model learns a balanced representation for the different citation reason classes.

Table 5 shows the accuracy of the pre-trained RNN_a^{+S} model trained on 50% of the Citation Reason dataset, and evaluate on the remaining statements. The pre-trained model has a better performance for nearly all citation reasons. It is important to note that due to the small number of statements in the Citation Reason dataset and additionally the number of classes, the prediction outcomes are not optimal. Our goal here is to show that the citation reason can be detected and we leave for future work a large scale evaluation.

7 DISCUSSION AND CONCLUSIONS

In this paper, we presented an end-to-end system to characterize, categorize, and algorithmically assess the verifiability of Wikipedia contents. In this Section we discuss the theoretical and practical implications of this work, as well as limitations and future directions.

7.1 Theoretical Implications

A Standardization of Citation Reasons. We used mixed methods to create and validate a Citation Reason Taxonomy. We then used this taxonomy to label around 4,000 sentences with reasons why they need to be referenced, and found that, in English Wikipedia, they are most often *historical facts, statistics or data about a subject, or direct or reported quotations*. Based on these annotations, we produced a Citation Reason corpus that we are

Table 6: Most common article topics and article sections for the different citation reasons.

Article Section						
quotation	**statistics**	**controversial**	**opinion**	**life**	**scientific**	**historical**
reception	history	history	reception	biography	description	history
history	reception	background	history	history	history	background
legacy	legacy	reception	development	early life	taxonomy	abstract
production	abstract	legacy	production	career	habitat	aftermath
biography	description	aftermath	background	background	characteristics	life and career
Article Topics						
quotation	***statistics***	***controversial***	***opinion***	***life***	***scientific***	***historical***
videogame	athlete	military conflict	videogame	athlete	animal	conflict
athlete	settlement	videogame	athlete	office holder	fungus	military person
book	videogame	settlement	album	royalty	plant	royalty
officeholder	infrastructure	athlete	single	military	military unit	office holder
album	country	royalty	book	artist	band	settlement

making available to other researchers as open data[12]. While this taxonomy and corpus were produced in the context of a collaborative encyclopedia, given that they are not topic- or domain-specific, we believe they represent a resource and a methodological foundation for further research on online credibility assessments, in particular seminal efforts aiming to design controlled vocabularies for credibility indicators[51].

Expert and Non-expert Agreement on Citation Reasons. To create the verifiability corpus, we extended to crowdworkers a labeling task originally designed to elicit judgments from Wikipedia editors. We found that *(non-expert) crowdworkers and (expert) editors agree about why sentences need citations in the majority of cases*. This result aligns with previous research [31], demonstrating that while some kinds of curation work may require substantial expertise and access to contextual information (such as norms and policies), certain curation subtasks can be entrusted to non-experts, as long as appropriate guidance is provided. This has implications for the design of crowd-based annotation workflows for use in complex tasks where the number of available experts or fact-checkers doesn't scale, either because of the size of the corpus to be annotated or its growth rate.

Algorithmic Solutions to the Citation Need Task. We used Recurrent Neural Networks to classify sentences in English Wikipedia as to whether they need a citation or not. We found that algorithms can effectively perform this task in English Wikipedia's Featured Articles, and generalize with good accuracy to articles that are not featured. We also found that, contrary to most NLP classification tasks, our Citation Need model outperforms expert editors when they make judgments out of context. We speculate that this is because when editors are asked to make judgments as to what statements need citations in an unfamiliar article without the benefit of contextual information, and when using a specialized microtask interface that encourages quick decision-making, they may produces more conservative judgments and default to Wikipedia's general approach to verifiability—dictating that all information that's likely to be challenged should be verifiable, ideally by means of an inline citation. Our model, on the other hand, is trained on the complete Featured Article corpus, and therefore learns from the wisdom of the whole editor community how to identify sentences that need to be cited.

Algorithmic Solutions to the Citation Reason Task We made substantial efforts towards designing an interpretable Citation Need model. In Figure 5 we show that our model can capture words and phrases that describe citation reasons. To provide full explanations, we designed a model that can classify statements needing citations with a reason. To determine the citation reason, we modified the binary classification model RNN_a^{+S} to predict the eight reasons in our taxonomy. We found that using the pre-trained model in the binary setting, we could re-adjust the model's weights to provide reasonable accuracy in predicting citation reasons. For citation reason classes with sufficient training data, we reached precision up to $P = 0.62$. We also provided insights on how to further sample Wikipedia articles to obtain more useful data for this task.

7.2 Limitations and Future Work

Labeling sentences with reasons why they need a citation is a non-trivial task. Community guidelines for inline citations evolve over time, and are subject to continuous discussion: see for example the discussion about why in Wikipedia "you need to cite that the sky is blue" and at the same time "you don't need to cite that the sky is blue" [13]. For simplicity, our Citation Reason classifier treats citation reason classes as mutually exclusive. However, in our crowdsourcing experiment, we found that, for some sentences, citation reasons are indeed not mutually exclusive. In the future, we plan to add substantially more data to the verifiability corpus, and build multi-label classifiers as well as annotation interfaces that can account for fuzzy boundaries around citation reason classes.

In Sec. 5 we found that, while very effective on Wikipedia-specific data, our Citation Need model is not able to generalize to fact-checking corpora. Given the difference in genre between the political discourse in these corpora, and the Wikipedia corpus, this limitation is to be expected. We explored, however, two other generalizability dimensions: domain expertise and language. We demonstrated that, for this task, annotation can be effectively performed by non-experts, facilitating the solution of this task at scale and distributing it beyond expert communities. Moreover, we built a general multilingual taxonomy by evaluating policies from different Wikipedia language communities, and by testing its effectiveness with expert contributors from English, Italian, and French Wikipedia.

More broadly, this work is designed for multilingual generalizability. In the future, we aim to replicate the large annotation efforts across languages. This should be fairly straight-forward, since Featured Articles exist in 163 Wikipedia language editions[14]. Moreover, the RNN model can be fed with word vectors such as fasttext[8], which now exist in more than 80 languages [7] and that one can re-train with any language from a Wikipedia project.

Finally, in this study we consider the application of verifiability policies to a static snapshot of Wikipedia articles, not taking into account their revision

[12]URL hidden for double blind submission

[13]https://en.wikipedia.org/wiki/Wikipedia:You_do_need_to_cite_that_the_sky_is_blue

[14]https://www.wikidata.org/wiki/Q16465

history. We also used general text features, and limited the encyclopedic-specific features to the *main section* feature. We expect that the distribution of citation reasons may vary over the course of an article's development, as a function of how controversial or disputed particular sections are. Performing this analysis is beyond the scope of the current study, but it might surface important exceptions to our findings or uncover interesting editing dynamics.

7.3 Practical Implications

Our study also has practical implications for the design of collaborative systems to support information verifiability and fact-checking. Our results of a robust agreement between non-experts and experts around citation reasons suggests that this type of task can be effectively crowdsourced, potentially allowing systems to recruit non-experts to triage or filter unverified statements based on model predictions, and allowing subject-matter experts to focus their attention on identifying reliable sources for these statements, or improving articles or topics with the highest rate of unreferenced factual claims. On Wikipedia, existing microtasking tools such as CitationHunt[15] could be modified to surface unverified claims that have not yet been flagged *citation needed* by humans and provide users with model predictions to guide their research, while also allowing users to provide feedback on those predictions to refine and improve the model. The model could also be used to surface article or statement-level information quality indicators to Wikipedia readers, using scores or lightweight visualizations, as suggested by Forte and others [2, 18], to support digital literacy and to allow readers to make more informed credibility assessments. Finally, downstream re-users of Wikipedia content, such as search engines and social media platforms, could also draw on model outputs to assign trust values to the information they extract or link to.

Beyond Wikipedia, our work complements existing attempts to assess how experts and non-experts assess the credibility of digital information, [51] and suggests that it is possible to develop robust *verifiability taxonomies* and automated systems for identifying unverified claims in complex information spaces even without substantial domain knowledge. Such capabilities could support large-scale, distributed fact checking of online content, making the internet more robust against the spread of misinformation and increasing the overall information literacy of internet users.

ACKNOWLEDGMENTS

We would like to thank the community members of the English, French and Italian Wikipedia for helping with data labeling and for their precious suggestions, and Bahodir Mansurov and Aaron Halfaker from the Wikimedia Foundation, for their help building the WikiLabels task. This work is partly funded by the ERC Advanced Grant ALEXANDRIA (grant no. 339233), and BMBF Simple-ML project (grant no. 01IS18054A).

REFERENCES

[1] Amjad Abu-Jbara, Jefferson Ezra, and Dragomir Radev. 2013. Purpose and polarity of citation: Towards nlp-based bibliometrics. In *Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: Human language technologies*. 596–606.

[2] B. Thomas Adler, Krishnendu Chatterjee, Luca de Alfaro, Marco Faella, Ian Pye, and Vishwanath Raman. 2008. Assigning Trust to Wikipedia Content. In *Proceedings of the 4th International Symposium on Wikis (WikiSym '08)*. ACM, New York, NY, USA, Article 26, 12 pages. https://doi.org/10.1145/1822258.1822293

[3] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[5] Jonathan Berant, Ido Dagan, Meni Adler, and Jacob Goldberger. 2012. Efficient tree-based approximation for entailment graph learning. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 117–125.

[6] Ivan Beschastnikh, Travis Kriplean, and David W McDonald. 2008. Wikipedian Self-Governance in Action: Motivating the Policy Lens.. In *ICWSM*.

[7] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606* (2016).

[8] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics* 5 (2017), 135–146.

[9] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[10] Chih-Chun Chen and Camille Roth. 2012. Citation Needed: The Dynamics of Referencing in Wikipedia. In *Proceedings of the Eighth Annual International Symposium on Wikis and Open Collaboration (WikiSym '12)*. ACM, New York, NY, USA, Article 8, 4 pages. https://doi.org/10.1145/2462932.2462943

[11] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[12] François Chollet et al. 2015. Keras.

[13] Chung Joo Chung, Hyunjung Kim, and Jang-Hyun Kim. 2010. An anatomy of the credibility of online newspapers. *Online Information Review* 34, 5 (2010), 669–685. https://doi.org/10.1108/14684521011084564

[14] Besnik Fetahu, Katja Markert, and Avishek Anand. 2015. Automated News Suggestions for Populating Wikipedia Entity Pages. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*. 323–332. https://doi.org/10.1145/2806416.2806531

[15] Besnik Fetahu, Katja Markert, and Avishek Anand. 2017. Fine Grained Citation Span for References in Wikipedia. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. 1990–1999. https://aclanthology.info/papers/D17-1212/d17-1212

[16] Besnik Fetahu, Katja Markert, Wolfgang Nejdl, and Avishek Anand. 2016. Finding News Citations for Wikipedia. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*. 337–346. https://doi.org/10.1145/2983323.2983808

[17] Andrea Forte, Nazanin Andalibi, Tim Gorichanaz, Meen Chul Kim, Thomas Park, and Aaron Halfaker. 2018. Information Fortification: An Online Citation Behavior. In *Proceedings of the 2018 ACM Conference on Supporting Groupwork (GROUP '18)*. ACM, New York, NY, USA, 83–92. https://doi.org/10.1145/3148330.3148347

[18] Andrea Forte, Nazanin Andalibi, Thomas Park, and Heather Willever-Farr. 2014. Designing Information Savvy Societies: An Introduction to Assessability. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2471–2480. https://doi.org/10.1145/2556288.2557072

[19] Andrea Forte, Vanesa Larco, and Amy Bruckman. 2009. Decentralization in Wikipedia Governance. *Journal of Management Information Systems* 26, 1 (2009), 49–72. https://doi.org/10.2753/MIS0742-1222260103 arXiv:https://doi.org/10.2753/MIS0742-1222260103

[20] R. Stuart Geiger and Aaron Halfaker. 2013. When the Levee Breaks: Without Bots, What Happens to Wikipedia's Quality Control Processes?. In *Proceedings of the 9th International Symposium on Open Collaboration (WikiSym '13)*. ACM, New York, NY, USA, Article 6, 6 pages. https://doi.org/10.1145/2491055.2491061

[21] Jim Giles. 2005. Internet encyclopaedias go head to head. *Nature* 438, 7070 (Dec. 2005), 900–901. http://dx.doi.org/10.1038/438900a

[22] Edouard Grave, Tomas Mikolov, Armand Joulin, and Piotr Bojanowski. 2017. Bag of Tricks for Efficient Text Classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*. 427–431. https://aclanthology.info/papers/E17-2068/e17-2068

[23] Naeemul Hassan, Fatma Arslan, Chengkai Li, and Mark Tremayne. 2017. Toward automated fact-checking: Detecting check-worthy factual claims by ClaimBuster. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1803–1812.

[24] Qi He, Daniel Kifer, Jian Pei, Prasenjit Mitra, and C Lee Giles. 2011. Citation recommendation without author supervision. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 755–764.

[25] Joan B Hooper. 1974. *On assertive predicates*. Indiana University Linguistics Club.

[26] Lauri Karttunen. 1971. Implicative verbs. *Language* (1971), 340–358.

[27] Brian Keegan, Darren Gergle, and Noshir Contractor. 2013. Hot Off the Wiki: Structures and Dynamics of WikipediaâĂŹs Coverage of Breaking News Events. *American Behavioral Scientist* 57, 5 (2013), 595–622. https://doi.org/10.1177/0002764212469367 arXiv:https://doi.org/10.1177/0002764212469367

[28] David J Ketchen and Christopher L Shook. 1996. The application of cluster analysis in strategic management research: an analysis and critique. *Strategic management journal* 17, 6 (1996), 441–458.

[29] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] https://tools.wmflabs.org/citationhunt/

[30] Paul Kiparsky and Carol Kiparsky. 1968. *Fact.* Linguistics Club, Indiana University.

[31] Aniket Kittur, Ed H. Chi, and Bongwon Suh. 2008. Crowdsourcing User Studies with Mechanical Turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 453–456. https://doi.org/10.1145/1357054.1357127

[32] Lev Konstantinovskiy, Oliver Price, Mevan Babakar, and Arkaitz Zubiaga. 2018. Towards Automated Factchecking: Developing an Annotation Schema and Benchmark for Consistent Automated Claim Detection. *arXiv preprint arXiv:1809.08193* (2018).

[33] Srijan Kumar, Robert West, and Jure Leskovec. 2016. Disinformation on the Web: Impact, Characteristics, and Detection of Wikipedia Hoaxes. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 591–602. https://doi.org/10.1145/2872427.2883085

[34] Stacey M Lavsa, Shelby L Corman, Colleen M Culley, and Tara L Pummer. 2011. Reliability of Wikipedia as a medication information source for pharmacy students. *Currents in Pharmacy Teaching and Learning* 3, 2 (2011), 154–158.

[35] WĂĆodzimierz Lewoniewski, Krzysztof WĂŹcel, and Witold Abramowicz. 2017. Analysis of References Across Wikipedia Languages. *Communications in Computer and Information Science Information and Software Technologies* (2017), 561âĂŞ573. https://doi.org/10.1007/978-3-319-67642-5_47

[36] Yunfei Long, Qin Lu, Rong Xiang, Minglei Li, and Chu-Ren Huang. 2017. Fake News Detection Through Multi-Perspective Speaker Profiles. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017, Volume 2: Short Papers*. 252–256. https://aclanthology.info/papers/I17-2043/i17-2043

[37] Louise Matsakis. 2018. Facebook and Google must do more to support Wikipedia. https://www.wired.co.uk/article/wikipedia-google-youtube-facebook-support

[38] Connor McMahon, Isaac Johnson, and Brent Hecht. 2017. The substantial interdependence of Wikipedia and Google: A case study on the relationship between peer production communities and information technologies. In *11th International Conference on Web and Social Media, ICWSM 2017*. AAAI Press.

[39] Preslav Nakov, Lluís Màrquez, Alberto Barrón-Cedeño, Wajdi Zaghouani, Tamer Elsayed, Reem Suwaileh, and Pepa Gencheva. 2018. CLEF-2018 Lab on Automatic Identification and Verification of Claims in Political Debates. In *Proceedings of the CLEF-2018*.

[40] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[41] Philipp Petrenz and Bonnie L. Webber. 2011. Stable Classification of Text Genres. *Computational Linguistics* 37, 2 (2011), 385–393. https://doi.org/10.1162/COLI_a_00052

[42] Reid Priedhorsky, Jilin Chen, Shyong (Tony) K. Lam, Katherine Panciera, Loren Terveen, and John Riedl. 2007. Creating, Destroying, and Restoring Value in Wikipedia. In *Proceedings of the 2007 International ACM Conference on Supporting Group Work (GROUP '07)*. ACM, New York, NY, USA, 259–268. https://doi.org/10.1145/1316624.1316663

[43] Marta Recasens, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. 2013. Linguistic models for analyzing and detecting biased language. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1650–1659.

[44] Christina Sauper and Regina Barzilay. 2009. Automatically Generating Wikipedia Articles: A Structure-Aware Approach. In *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*. 208–216. http://www.aclweb.org/anthology/P09-1024

[45] Roser Saurí and James Pustejovsky. 2009. FactBank: a corpus annotated with event factuality. *Language Resources and Evaluation* 43, 3 (2009), 227–268. https://doi.org/10.1007/s10579-009-9089-9

[46] Shilad Sen, Margaret E. Giesel, Rebecca Gold, Benjamin Hillmann, Matt Lesicko, Samuel Naden, Jesse Russell, Zixiao (Ken) Wang, and Brent Hecht. 2015. Turkers, Scholars, "Arafat" and "Peace": Cultural Communities and Algorithmic Gold Standards. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. ACM, New York, NY, USA, 826–838. https://doi.org/10.1145/2675133.2675285

[47] Gabriel Stanovsky, Judith Eckle-Kohler, Yevgeniy Puzikov, Ido Dagan, and Iryna Gurevych. 2017. Integrating Deep Linguistic Features in Factuality Prediction over Unified Datasets. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*. 352–357. https://doi.org/10.18653/v1/P17-2056

[48] Robert F Tate. 1954. Correlation between a discrete and a continuous variable. Point-biserial correlation. *The Annals of mathematical statistics* 25, 3 (1954), 603–607.

[49] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a Large-scale Dataset for Fact Extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. 809–819. https://aclanthology.info/papers/N18-1074/n18-1074

[50] Fernanda B. Viégas, Martin Wattenberg, and Matthew M. McKeon. 2007. The Hidden Order of Wikipedia. In *Online Communities and Social Computing*, Douglas Schuler (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 445–454.

[51] Amy X. Zhang, Aditya Ranganathan, Sarah Emlen Metz, Scott Appling, Connie Moon Sehat, Norman Gilmore, Nick B. Adams, Emmanuel Vincent, Jennifer Lee, Martin Robbins, Ed Bice, Sandro Hawke, David Karger, and An Xiao Mina. 2018. A Structured Response to Misinformation: Defining and Annotating Credibility Indicators in News Articles. In *Companion Proceedings of the The Web Conference 2018 (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 603–612. https://doi.org/10.1145/3184558.3188731

SWAT: Seamless Web Authentication Technology

Florentin Rochet
UCLouvain Crypto Group
florentin.rochet@uclouvain.be

Kyriakos Efthymiadis
AI Lab
Vrije Universiteit Brussel

François Koeune
UCLouvain Crypto Group
francois.koeune@uclouvain.be

Olivier Pereira
UCLouvain Crypto Group

ABSTRACT

We present a seamless challenge-response authentication protocol which leverages on the variations of html5 canvas rendering made by the software and hardware stacks. After a training phase that leads to feature extraction with deep learning techniques, a server becomes able to authenticate a user based on fresh canvasses, hence avoiding replay attacks. The whole authentication process is natively supported by any mainstream browser, stateless on client side and can be transparent to the user. We argue that those features facilitate deployment and composition with other authentication mechanisms without lowering the user experience. We present the threat model against which our protocol is expected to live and discuss its security. We also present a prototype implementation of our protocol and report on a real-word experimentation that we ran in order to analyze its efficiency and effectiveness.

KEYWORDS

convolutional neural networks, authentication, html5 canvas, 2FA, usable security

ACM Reference Format:
Florentin Rochet, Kyriakos Efthymiadis, François Koeune, and Olivier Pereira. 2019. SWAT: Seamless Web Authentication Technology. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, Article 4, 11 pages. https://doi.org/10.1145/3308558.3313637

1 INTRODUCTION

Although paramount for security, user authentication is often a hindrance for users: typing in a password, connecting a security token or interacting with a dedicated device is seen as a frustrating and time-consuming step. As a result, users more often than not are tempted to use time-saving shortcuts – bad or identical passwords, access card left next to device… – damaging for security, or to give up the transaction if the "reward" is not deemed to be worth the effort, resulting in losses for the provider. In this sense, seamless authentication protocols represent an appealing target.

Recently, browser fingerprinting [4], and in particular canvas fingerprinting [19] have been introduced as a means of identification. Put simply, the idea is that browser-specific features such as the supported fonts, software version or, for canvas fingerprinting, rendering of vector graphics can provide sufficient information to allow distinguishing a device from another, and hence to recognize a user coming back after a previous visit. Yet, to the best of our knowledge, this technique has been used only for identification – in the sense of recognizing a user – and not for authentication – in the sense of gaining confidence that a user is indeed who he pretends to be.

This paper represents a first attempt in this direction. Using canvas fingerprinting and deep learning techniques, we propose a challenge-response protocol taking place between a server and a standard web browser and operating in way that is transparent for the user. Intuitively, in an initial phase, the server asks the browser to perform several drawings and, through the use of deep learning, "learns" specifics about how the browser draws. As part of a later authentication process, the browser receives from the server a challenge describing a new drawing to perform. The server then passes the new drawing through the, now trained, deep learning model and if it recognizes the same specifics in the resulting image, authentication is deemed successful. The challenge-response process of SWAT aims at offering protection against stolen credentials, or offline replay attacks by malicious servers.

Our contributions are the design, the implementation and the evaluation of SWAT, a new protocol for user-friendly authentication based on canvas extraction of hardware/software dependencies. More specifically, we offer:

- An open-source implementation of SWAT for the Django Web framework. Our implementation covers (i) the data collection needed for the machine learning phases, (ii) the learning phase itself, implemented as a cron job that verifies every minute if new users registered and (iii) the authentication phase if a learned model is available for a given user.
- Experiments and evaluation of the efficacy of SWAT given real-world collection of user data.
- A discussion about real-world applications of SWAT, as well as its expected security benefits. We emphasize SWAT's security in a threat model that shows an interesting complementary with standard authentication systems, supporting the use of SWAT as a second authentication factor.

Our protocol takes advantage of standardized web technologies. In consequence, SWAT is natively supported by Web Browsers and is designed to be fast and easy to deploy.

Roadmap. Section 2 provides the necessary background for canvas fingerprinting and deep neural networks. Then, we describe the

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313637

SWAT protocol: how the client and server side conjointly perform the authentication as well as the threat model supported by our protocol. Section 4 covers an experimental analysis of our protocol, with real-world collection of data and analysis of the accuracy of the authentication. Section 5 describes various usage scenarios. Finally, Sections 6-8 discuss related work and future work before concluding.

2 BACKGROUND

2.1 Canvas fingerprinting

Mowery and Shacham [19] designed a new fingerprinting method based on browser font and WebGL rendering. This method exploits the surprising output variation of the HTML5 <canvas>. The <canvas> element provides, within a web page, an area in which the programmer can draw using Javascript. The variation depends on the operating system running the browser, the type of CPU or GPU, the graphic driver, the browser brand and its version.

The fingerprinting method works as follows: some Javascript code recovers the context of a canvas element within the webpage and draws a text and a text's scene. The goal is to generate a drawing impacted by as much software and hardware components as possible. The programmer, using common high level API functions, produces a drawing and then converts it to a base64 representation of the image. Then, a collision-resistant hash function receives this representation as an input and outputs a different value (the fingerprint) for any small change within pixel position or color of the image.

This method has been extensively used as a stateless tracker [1, 5], because the same user should produce the same hash output when drawing the same text and text's scene twice. In this paper, we try to take advantage of canvas's variation to the benefit of web users.

2.2 Deep Neural Networks

A natural candidate for extracting useful information from canvas drawings in an automated fashion is deep learning. Deep learning is a subfield of machine learning research which has demonstrated huge success in computer vision tasks like classification, localisation and segmentation [12, 13, 15, 27, 28].

Typically in machine learning, algorithms receive input features and produce, hopefully, useful outputs. The input features are usually manually defined and depend on the application domain. Consider for example an image classification task in which the algorithm needs to decide what object is displayed in an image. The input in this case would be defined in terms of a set of features defined on the pixels of the input image. It is easy to realize that defining manual features in the pixel space is extremely difficult. Representation learning can help alleviate this problem by providing the means to build models that can extract useful information from data.

Deep learning is a type of learning representation which follows a structured approach to extracting information from input data. It accomplishes this by using several layers to build representations of increasing complexity with initial layers dealing with primitive types, while the next layers deal with more complex objects composed of primitive types. For more in depth information regarding deep learning we refer the reader to Goodfellow *et al.* [10].

2.2.1 Fully Connected. Figure 1 shows a typical fully connected network. Its layers are comprised of scalar output values called *neurons* which are arranged in a hierarchical fashion. Information flows from the input all the way to the output with in-between layer outputs being the input to the following layer. Information does not flow back hence this network structure is also called a *feed-forward network*. The output of an individual neuron is called its activation which is typically non-linear (deep networks of linear activations are equivalent to single layer networks), hence it is referred to as the *non-linearity*. The activations of the last layer (output) is the prediction of the network given the first layer (input); all other layers are not directly observed and are conveniently called *hidden layers*.

Let $X = (x_1, x_2, x_3, \ldots, x_n)$ be a vector of inputs and $w = (w_1, w_2, w_3, \ldots, w_n)$ a vector of weights and b a bias. Then the output, or activation, of a given neuron is calculated by

$$a = \sigma(w \cdot x + b), \tag{1}$$

in which σ is the non-linearity. Currently the most used non-linearity is the rectified linear unit, or ReLU [21]. The ReLU function is defined as $\sigma(x) = \max(0, x)$ and it has been shown to result in strong gradients that allows much faster training for deep networks.

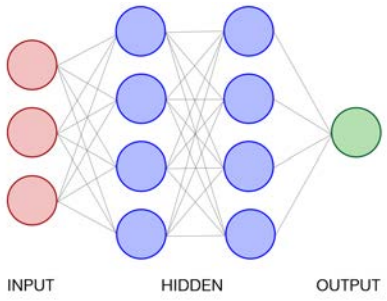


Figure 1: Fully Connected Neural Network.

What we want to find is the set of weights and biases that bring the outputs of the network closer to the desired output. Specifically, given network predictions $\hat{y} = (\hat{y}_1, \hat{y}_2, \hat{y}_3, \ldots, \hat{y}_n)$ and targets $y = (y_1, y_2, y_3, \ldots, y_n)$, we want the network outputs to approximate the targets as closely as possible for all inputs $X = (x_1, x_2, x_3, \ldots, x_n)$. This can be formulated as an optimization problem where we try to minimize a loss function. In this work we are dealing with a binary classification problem hence we use the cross-entropy loss to train our model over all training examples M,

$$Loss = -\frac{1}{M} \sum_{i}^{M} \left[y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \right] \tag{2}$$

In order to find the model parameters that minimise the loss function we use gradient descent. By taking the gradient of the loss function w.r.t. the model parameters we can find how these

parameters need to be changed to reduce the loss value. The parameters are initialised at random and are then iteratively updated by descending along the gradient surface until a suitable minimum value is reached. Of course computing the exact gradient for the loss function would have us iterate through the entire dataset which is prohibitively expensive. Instead what is typically used is mini-batch gradient descend in which a subset of the training data is used to approximate the gradient at each step.

2.2.2 Convolutional Neural Networks. Convolutional neural networks [8, 18], CNNs, is the state-of-the-art technique when dealing with problems of computer vision in machine learning. Typical, fully connected networks, in which every neuron in layer l is connected to every neuron in layer $l+1$ ignore a very important aspect when dealing with images as input; spatial information. Every pixel in a fully connected network is treated independently when in most cases, pixels spatially located close to each other is important information that needs to be captured. CNNs provide the means to exploit the local structure in the input by learning local features. In deep architectures, earlier convolutional layers will learn to extract spacial features such as lines, edges and so on, while later layers will learn to combine those to form more distinct features, such as a face or a dog.

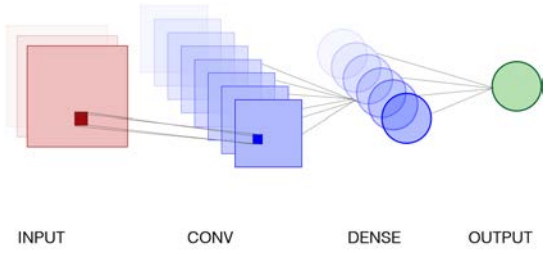


Figure 2: Convolutional Neural Network.

Figure 2 shows a schematic representation of a CNN. Typically a CNN is built by a series of convolutional layers, non-linearities, pooling layers and finally the output. The input to the CNNs presented in this work is an image. At each layer the input is convolved with a set of weights called *filters* to produce *feature maps*. Different feature maps i.e. layer outputs, are convolved with different sets of filters, but filters are shared within the same feature map. It is those sparse interactions and parameter sharing that greatly reduce the number of parameters of the model, its computational and memory complexity.

Equation 3 show the operation that defines the l-th feature map

$$a^l = \sigma\Big(\sum_m w_m^l * x + b^l\Big), \tag{3}$$

where we sum over all features maps, w_m^l are the filter weights to be learned, b is the bias, $*$ represents the cross-correlation operation (in deep learning known as "convolution"), and σ represents the element-wise activation function which as discussed previously is typically a ReLU. Popular kernel choices in most cases, and dependent on the input size, are $3 \times 3, 5 \times 5, 7 \times 7$. A pooling layer usually follows which aggregates the values of neighboring neurons based on their maximum, or average value. This results in the model being invariant to small translations in the input and is an important aspect if we only care about the presence of a feature and not its placement. Typically pooling layers have a stride of 2 i.e., they aggregate values that are spaced 2 neurons apart. This results in a reduction of neurons allowing the following convolutional layer to have a larger field of view of the input. For more in depth information regarding convolutional networks and their applications we refer the reader to [10].

3 SWAT

3.1 Overview

Authentication is usually based on an agreed token that is recognized by an identity provider and that a user uses to prove this identity. This token can take many forms with many different extraction methods. These are commonly divided in the following well-known classification: something we know, something we have and something we are.

In our case, the token can be seen as "the way the user draws in canvas" and by the user, we mean his/her full stack of hardware and software. Our method to extract this token and to send it toward the identity provider shows a unique advantage: it requires no interaction with the user and all recent browsers are compatible without any additive software to plug-in. This technique falls into the "something we have" category, however offering a better usability than other methods classified into the same category.

The SWAT authentication protocol is a challenge-response protocol that takes advantage of canvas's high variation and deep learning's feature extraction to provide a user-friendly layer of authentication. We can abstract SWAT as three independent phases: P1: data collection, P2: learning phase and P3: authentication phase. Phases P1 and P2 must be completed once for each stack of hardware and software owned by a user. When P1 and P2 are complete, the authentication phase can be performed as often as needed.

The data collection phase can be performed in a stealthy way to the user. While we advocate for transparency, scenarios may exist where a stealth collection of data is preferable. Such choice would be left to the identity provider, but could also be integrated as part of user preferences.

In our proof of concept implementation, the browser generates 2000 35x280x4 images based on random texts sent by the server (as a matter of fact, writing in canvas appears to be one of the most browser-sensitive features we observed). We extract those images from a `<canvas>` element, and convert them to a base64 representation which we send to the server.

Once the data collection (P1) is finished, we start extracting features in the learning phase (P2). We use a CNN to extract the user's features and build a binary classification model for each user. This is in contrast to what we usually see in the literature where one end-to-end model is trained with multiple classes. The reason we have decided to follow the path of per-user model is rooted in a more efficient training procedure for our authentication system. If we employed an end-to-end model trained on all users, then

every time a new user would register to our system, we would need to re-train the entire network; a process which takes time and resources. By having a binary classification process where we train a model per user, we do away with that problem. Our current CNNs are simple enough to spend less than 3 minutes of learning with a decent accuracy and almost no over-fitting. We have designed our CNN this way to focus on usable research and show that this authentication method could be deployed, even at a large scale for systems handling a large pool of users.

Finally, in the authentication phase (P3), the browser and the server play a (stealthy) challenge-response protocol, where the server asks to draw several texts and text's scenes. The server then uses the binary classification built during P2 to assess whether the browser's answer matches its user's model. To easily verify the validity of the client's response and prevent a replay attack, the server produces locally the same drawing as the one he asked from the browser and compares it to the response received from the browser using the method of least squares. Even though the two images will not be identical – this is the very essence of SWAT – they should be close enough for the least square difference to be small when based on the same challenge.

We provide in this paper a full stack implementation of the protocol outlined above and we describe, in the section below, each piece in details. We do not see SWAT as a replacement for existing strong authentication methods, as it does not guarantee as much security as a strong password for instance. However, we believe that SWAT is an interesting candidate to be part of a layered multi-factors authentication system, such as password+SWAT or other methods, mostly depending on the level of security needed and the threat model.

3.2 Client side

The client side of the SWAT protocol is mostly responsible for 1) drawing the random canvas asked by the server and 2) extracting a representation of the canvas as a png image. The Javascript code then sends the image representation to the server. As thousands of images are generated and sent during the collection phase, it might take up to a few minutes to complete. A "stop and resume later" strategy for the collection phase is possible, but the user needs to authenticate with another method to resume.

```
function generateRandomCanvas(txt) {
  var canvas =  document.getElementById("canvas");
  canvas.width = 280, canvas.height = 35;
  var ctx = canvas.getContext("2d");
  ctx.clearRect(0,0, canvas.width, canvas.height);
  //drawing the new canvas
  ctx.font= "18px 'Arial'";
  ctx.textBaseline = "alphabetic";
  ctx.fillStyle = "#069";
  ctx.fillText(txt, 2, 15, 280);
  ctx.fillStyle = "rgba(102, 204, 0, 0.7)";
  ctx.fillText(txt, 4, 19, 280);
  ctx.fillStyle = "#069";
  ctx.fillText(txt, 2, 23, 280);
  ctx.fillStyle = "rgba(102, 204, 0, 0.7)";
  ctx.fillText(txt, 4, 27, 280);
  ctx.fillStyle = "#069";
  ctx.fillText(txt, 2, 31, 280);
  ctx.fillStyle = "rgba(102, 204, 0, 0.7)";
  ctx.fillText(txt, 4, 35, 280);
}
```

Listing 1: javascript code to generate a canvas content



Figure 3: Example of a canvas drawn by SWAT

During the authentication phase, the client receives a few texts to draw from the server (E.g. Figure 3). This process takes less than a second, does not require any interaction with the user and can be designed to be invisible.

3.3 Server side

The prototype implementation we developed [25] is a Django plugin using the Python deep learning *Keras* API [14] as the front-end to the server-side programmer and *Tensorflow* as the back-end. The collection phase stores every *base64* encoded *.png* image into a database for each user. Once we have enough different users, the system starts to learn by pooling each user's generated images and a set of random images from other users. For each user, we have 2000 images labelled as "yes" (all collected images) and 2551 random images from the database labelled as "no". We choose those numbers such that, when the system splits the set of images into a training set and a validation set, we obtain a training set size as a multiple of the batch size for efficiency reasons.

The system trains and saves one model per user as a binary classification model. During the authentication phase, we load the model associated to the claimed user's identity and we feed the model with the few fresh images that we asked to draw. For each input image, the model outputs a prediction value. In theory, if the claimed user's identity is right, the predicted value should be high. The closer the prediction is to 1, the more confident we are about the image belonging to the right user.

A user identity may have multiple learned models, namely one for each of his stacks of hardware/software (e.g. chrome browser and firefox on his laptop). The user should succeed the authentication for one of the learned model but not for the others.

3.4 Threat model and security

We now discuss the type of adversary against whom SWAT is resistant as well as how the protocol could be combined with other forms of authentication to protect against several threats without giving up on usability.

In contrast to security notions provided in cryptography, the non-forgeability of valid canvas for a given user cannot be linked to a computationally hard problem. The forgeability of SWAT can be reduced to the following adversarial game: given a set of valid canvasses for an identity and given a challenge, draw a canvas that obtains a high prediction value with non-negligible probability. If the adversary wins this game, then we have an active attack against SWAT. Assessing the resistance of SWAT against such an active adversary is an open research problem, which involves analyzing how deep learning (and other) strategies could allow him to reconstruct the model based on partial information and with a non-negligible success probability. As a preliminary step, in this paper we consider

only the case of passive attackers, that is, attackers who follow the protocol but do not actively try to forge a canvas based on a more sophisticated strategy.

The SWAT protocol aims to protect against the following problems:

- **Passive eavesdropping and replay attacks**: since SWAT is a challenge-response protocol, a consumed canvas cannot be replayed to gain access to the authenticated channel. Therefore, the SWAT protocol may be interesting to use in addition to other authentication methods that do not protect against those problems.
- **Stolen credentials**: The SWAT protocol is a stateless protocol, hence the information needed to authenticate cannot be collected on a user device.

Under the assumption that an indistinguishable forgery cannot be performed, our canvas challenge-response approach mitigates the following issues:

- **Database leakage**: If the database of a service gets compromised, it should not give the opportunity to an attacker to forge a valid canvas and obtain a valid credential.
- **Website attacker**: If a website collects drawings against the will of a user, it should not help to forge a valid canvas, except through an on-line man-in-the-middle attack. As a partial mitigation, we observe that Firefox recently added a user approval on canvas extraction [7] based on a Tor Browser's functionality which would reduce the likelihood of such risk.

Some threats remain out of scope for this protocol:

- **Active man-in-the-middle attack**: such as a website attacker that tricks a user into completing a challenge for another service. Due to the design of SWAT, the tricked user would not notice any completed challenge toward another service.
- **Hardware/software cloning**: given the software stack of a target, and given the information related to the exact hardware the target owns, an adversary would be able to build a machine that would successfully authenticate him.

This protocol is not designed for high security and we would discourage a standalone use to protect sensible information. However, this protocol closes the security gap that some other authentication methods have, such as weak passwords. More generally, when a protocol relies on an attached secret to perform the authentication, it is always weak against the leakage of such a secret. SWAT may prevent the leakage of a password, a cryptographic key or the theft of a security token from providing the thief with a direct access to the secure environment. This protocol may offer the time needed to the user to react and recover new strong credentials.

4 EXPERIMENTAL ANALYSIS

4.1 Methodology

Our main objective is to obtain a seamless, user interaction-free layer of authentication which user intensive applications could use to enhance their account security system. For this purpose, the main constraint we impose to the evaluation of our prototype is to have a relatively fast training phase to comply with realistic utilization of this protocol. By realistic utilization, we mean realistic training cost and sufficient efficacy.

We started by developing a framework for the multi purposes of data collection and evaluation, as well as for protocol demonstration. We chose CNN architectures in such a way that the learning phase takes less time than the collection phase (less than 1 minute). We do not argue in this paper for an optimal architecture suited for our protocol, since we believe that the optimum is bounded by the resource availability of the identity provider: the more resources the identity provider owns, the more complex CNN architectures they could employ. We made the choice to experiment with two different architectures to demonstrate the trade-off between result efficacy and cost of the training phase.

We opened the data collection process within the main author's social network, and we explain in Section 4.2 the steps taken to avoid any pollution (i.e., injection of wrong data) of the database. We evaluate our architectures' accuracy by splitting the set of available canvases for a given user in a training set and a validation set. During the evaluation, one of our goals was to study the impact of having common pieces of software on the prediction value. For example, what is the chance for user A running Mac OS X to score a high prediction on behalf of user B also running Mac OS X? To answer those questions, we collected meta-information about participating users and ran the authentication phase for each user over every other user in our dataset.

We finally investigate different solutions to accept or reject authentication attempts from the output of the classifier.

4.2 Data Gathering

We built a web interface [26], to gather real user data over a few weeks. In total, 118 collections have been done and among them, 109 fully completed the collection phase. We discarded the uncompleted involvements for this analysis, leaving us 109 remaining datasets. Alongside the 2000 images collected for each participant, we collected some metadata such as the operating system family and the browser family.

The data gathering has been made public on social networks, hence we had to design the gathering with security issues in mind. We evaluated worst-case scenarios and came up with a few countermeasures against them. The most serious threat we may face during a public data gathering is database pollution. We consider a database polluted when wrong data is indistinguishable from correct data and would change the expected result during our analysis (e.g., increasing false positives). First, we have to define what would be "wrong" data for our purpose and how such pollution may happen (intentionally or non-intentionally). Then, working to avoid the non-intentional pollution is the bare minimum. We also make intentional pollution difficult for unskilled people. However, protecting the data collection against skilled people is a challenging problem and is out of the scope of this work. We assume skilled people did not pollute our database to influence our results.

We consider the database polluted when two different recorded identities are the same machine (same stack of hardware and software). The interface to the data gathering has been made as simplistic as possible in order not to discourage users: we asked participants to provide an email address and to click on a button. Then,

the data collection proceeded, displaying the percentage of data collected to the participant. So, what could go wrong? We have to consider every possible and problematic action a human might take:

- Doing the data gathering multiple times with the same email on the same machine.
- Doing the data gathering multiple times with different emails on the same machine.
- Stopping the data collection.
- Registering an identity but blocking the data collection. This could be unintentional due to some browser addons that are suspicious with canvas image extraction.

We prevent the first two bullets points by relying on a stateless attribute: we compute a browser fingerprint that should be unique to each browser. This fingerprint is transmitted with the provided identity. If the fingerprint matches another identity in the database, we display an error. Otherwise, the identity is accepted and a data collection begins. We could have used a cookie but it would be less protective against unskilled malicious users who might know that cookies are often used to "remember" and how to delete them. Finally, we asked users to complete a captcha to participate. Skilled malicious users may be slowed down or at the best, discouraged to fill our database with junk data. For the last two bullets, we discarded any user who have not finished the data collection. We did not notice any strange behaviour during the data gathering and, backed by our methodology, we are confident that our database has not been polluted.

Figure 4 shows the distribution of Operating System and Browser families from our participating users. The main operating systems and browsers are well represented with a social bias toward Unix-based system (due to a non-negligible fraction of colleagues and friends within the participants) and many different families have been collected, including smartphones, which is what we need to validate our result over the global software and hardware ecosystem.

4.3 CNN Architectures

Table 1 shows the two CNN architectures we use to evaluate our protocol. For each user, Arch 1 provides a learning phase under one minute, and Arch 2 provides a learning phase under three minutes with a docker virtual machine using 12 cores of Intel(R) Xeon(R) CPU E5-2670 v3 2.30GHz. It can be expected to be much faster with dedicated instruction sets on the hardware, which are available on the market, or by deploying the models in GPU clusters. These performances allow an identity provider to derive a user specific model in less than a minute for any registered user.

4.4 Results

4.4.1 CNN Accuracy. SWAT is designed to produce one CNN binary classification model for each registered user. The efficiency of our trained models is based on the accuracy of the chosen CNN architecture, as well as on the quality of the derived model that each user obtains. We may wonder what efficiency the CNN architecture achieves, and more importantly, what set of criteria might be used to define an efficient architecture. We define here the criteria that are important for our protocol:

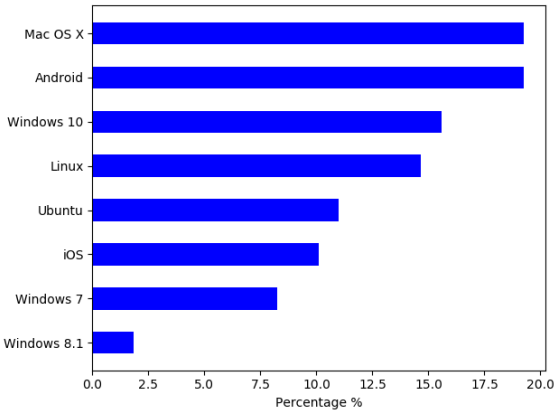
(1) The model accuracy: how well does the CNN derived model perform for each user? We consider both the accuracy of the training set and the accuracy of the validation set.
(2) The gap between best accuracy results and worst accuracy results.
(3) The over-fitting: how large is the gap between trained results and validation results?
(4) The number of epochs and the time for each epoch: how fast does the model learn?

The learning speed may be more or less important, depending on the targeted security level and on the amount of computational power that is available: high security requirements and high computational power availability push for the use of a more complex architecture, which leads to better results in terms of accuracy and best/worst performance gaps. We chose to work with fast architectures, showing how well we can perform at a moderate cost, without designing our solution around a specific hardware specification for our learning algorithm. This choice is guided by our main objective: offering a <realistic new authentication layer which can be deployed even for usage intensive applications.

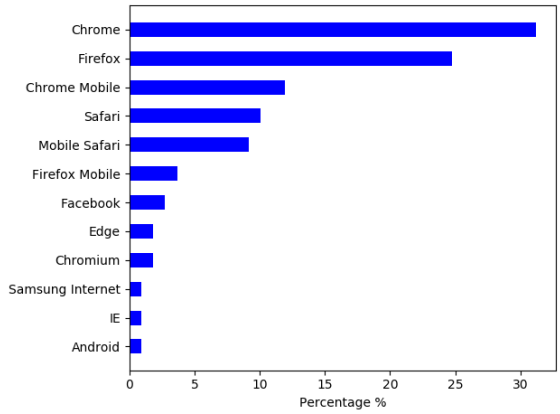
In Figure 5, we observe a median validation test reaching over 95% accuracy. The learning and validation sets contain canvas in the "yes" class and canvas in the "no" class chosen randomly from the database, and the accuracy is defined as the number of correct predictions over the total number of predictions. Figure 5b shows more over-fitting in the worst-case user (i.e., over all users and all runs), which indicates the need to collect more than 2000 canvas for that CNN architecture: because of the deep structure of the architecture, it may capture unique information about the training set which might not be representative of the user specific features.

4.4.2 True Positive Predictions. Figure 6 shows our classification's prediction results for new canvases during authentication. This figure shows the output of the authentication phase for legitimate authentications of each identity. The prediction result is not binary, it is a probability score for the canvas to belong to the 'yes' class. As can be seen on the figure, the second architecture shows a higher prediction distribution than the first one. We believe that there are two main reasons explaining this difference: we do not use max pooling in the second architecture and we have a deeper neural network. Indeed, max pooling is destructive (i.e., it discards information in favor of computational efficiency), and the deeper the network is, the more information it captures. However, those results alone are not enough to justify that the second architecture fits our authentication problem better. We also need to evaluate how a particular user performs against others. Ideally, we should obtain a 0 probability prediction for these situations.

4.4.3 False Positive Predictions. A high prediction result for true positive canvases would not be interesting if the CNN predicts also a high result of $user_i$'s canvas over $user_j$'s learned model. We need to evaluate this avenue to get a more precise idea of the effectiveness of this authentication method. Figure 7 shows CDFs of how well our CNNs perform for each user over all other users within our database. In Figure 7a, 'all' labeled lines compare our



(a) Operating System of each user of our database



(b) Browser of each user of our database

Figure 4: Percentage of observed OS family and Browser family in our data collection

	CNN		
Parameters	Arch 1	Arch 2	Search space
Optimizer	Adamax	Adamax	SGD, Adam, Adamax (default values)
Conv2D layers	3	17	3..20
Filters	4/layer	4/layer	4/8/16
Padding	valid valid same	valid (all)	valid, same
Dense	256 1	256 1	64/128/256/512 1
Flatten	yes	yes	-
Activation func	relu	relu	relu, tanh
Kernel init	Glorot normal	Glorot normal	-
Maxpooling	After each Conv2D	-	-
Pool Size	3x3 3x3 5x5	-	3x3 / 5x5
Batch size	256	256	64/128/256
Training epochs	100	100	50/100

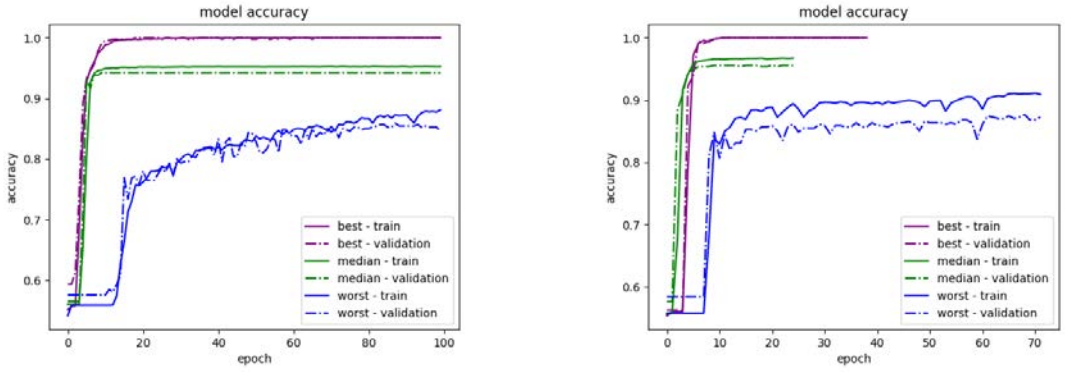
Table 1: Parameters choices for two architectures. Arch 2 is designed to offer better efficacy at a performance cost, which highlights the trade-off efficacy/performance of the various CNN architectures we could use

two architectures for all users, which shows that our CNN Arch 2 performs better as expected from the result of the previous section. Considering the 'all' labeled line, it gives us some intuition about the performance of this authentication method. Nevertheless, results about SWAT's performance in the real world may be better, because of two important points:

- Our data collection does not reflect the true distribution of hardware/software stacks which people own. Indeed, as we described in our Section 4.2, our data collection is biased toward Unix-based systems due to the author's social environment.
- Our results depends on the CNN used, and there is still strong potential for more efficient architectures.

This suggests that SWAT can achieve usable performances with a low training cost, and can potentially behave much better when deployed in a more realistic scenario by a company with the necessary computational resources.

We may also observe that users tend to obtain a higher score against other users running the same Operating System (Figure 7a) or the same Browser (Figure 7b). This result seems intuitive, as SWAT takes advantage of the various differences in the software/hardware stacks of users. When some parts of this stack are common, the CNN successfully extracts common features and may believe more easily that they are the same user. Another argument in favor of this interpretation can be observed with Mac OS X or iOS results: Due to less diversity in the hardware and software stack, Mac OS X



(a) Learning phase for Arch 1

(b) Learning phase for Arch 2

Figure 5: Model accuracy evaluation on CNN architectures

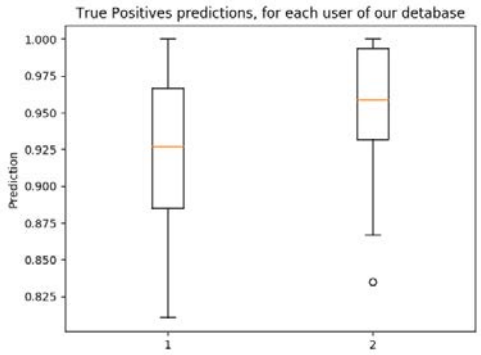


Figure 6: True positive prediction of CNN's architecture Arch 1 and CNN's architecture Arch 2

and iOS users are more likely to score high against each other (i.e., around 40% of our Mac OS X users score a high prediction against other Mac OS X users). We observe that this is less true as we move toward more complex CNN architectures, as showed by the results of the deeper CNN architecture.

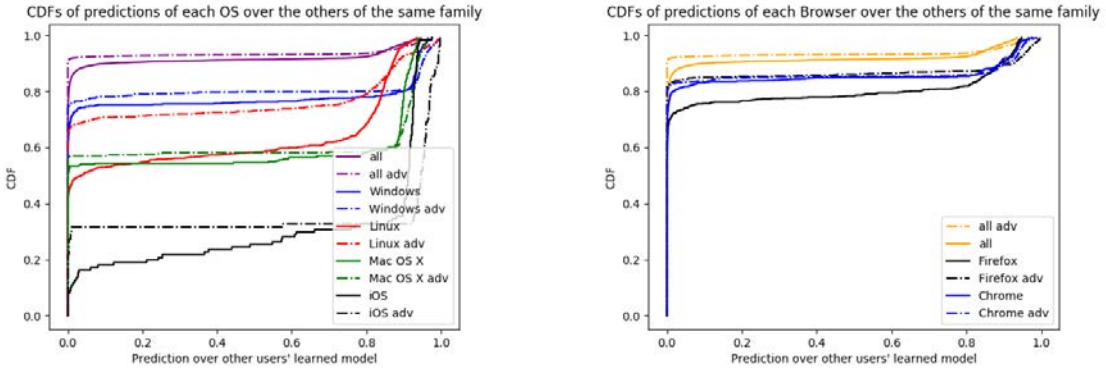
4.4.4 Accepting or Rejecting Authentication. An important question remains to be answered: how should we decide if an authentication is valid? The simplest and more straightforward technique would be to decide one global threshold which minimizes false positives and guarantees a true positive, based on our data. A second idea would be to use a threshold for each user, based on the result of their training phase. Indeed, since we derive one model per user, each of them produces a slightly different accuracy, as depicted in Figure 5. Therefore, having a model-based threshold to accept an authentication should give better results because some of the most accurate models can benefit from a higher threshold. A third and last idea would be to use, instead of a threshold, a validity interval for which we consider the authentication successful. Compared to the simple threshold, the validity interval would prevent $user_i$ from successfully authenticating as $user_j$ if $user_i$ obtains a much higher result.

Figure 8 shows the efficiency of our CNN architectures with ROC curves given two varying decision-making parameters: a global and unique threshold and a model-based interval for valid predictions. Since we know from the training phase which prediction value the valid user should obtain, we may use this information for the decision process to accept or reject an authentication attempt. This information is exploited with the model-based interval for valid prediction.

Figure 8a shows ROC curves for our two architectures given a global threshold varying from 0 to 1. For a TPR of 1, our CNN Arch 1 achieves $\approx 7,4\%$ of FPR and sets a global threshold to accept authentication to $\approx$ 0.81. The other deeper architecture Arch 2 achieves a TPR of 1 and $\approx 6\%$ of FPR, and sets a global threshold to accept authentication to ≈ 0.83. Figure 8b shows the result with a better heuristic to accept or reject authentication attempts. For each learned model, we accept an authentication attempt if the prediction is within an interval centered around the median value of the learning set's predictions. Figure 8b shows that this heuristic reduces the FPR to $\approx 5\%$. This improvement is motivated by two observations: first, by looking at Figure 7, we see that different users can score a high prediction for a given identity, even higher than the true identity itself. Hence, accepting authentication attempts simply based on exceeding a threshold is in fact allowing false positives. The second observation is that the prediction value is stable across several authentications of a same identity: if a training phase ends up giving a 0.85 prediction value to the canvas of one's identity, then this identity will score around 0.85 for each authentication. This stability characteristic allows a high TPR for small intervals.

4.5 Security Discussion

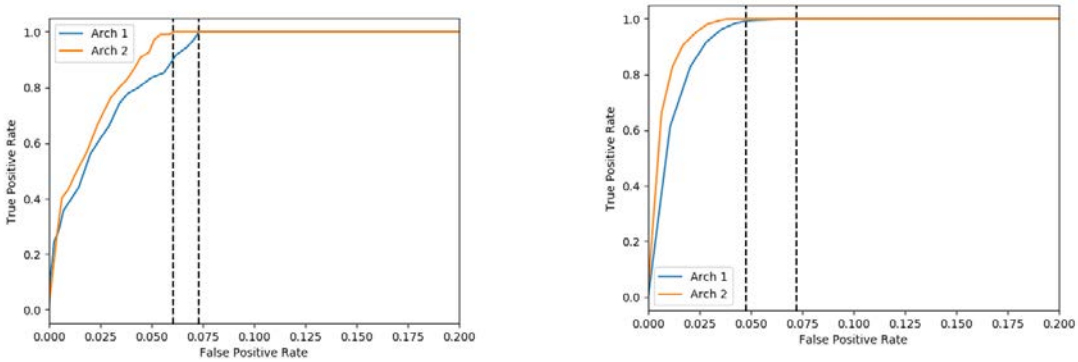
The current performance analysis of SWAT shows a non-negligible risk of a false positive. SWAT cannot hope to compete with the $\sim 2^{-128}$ security bounds provided by typical cryptographic tools, but this not its objective. Indeed, SWAT is not aimed to be a first-factor authentication, unless user friendliness is the critical property of the system (in such case, weak authentication is still better than



(a) Splitting users per OS families.

(b) Splitting users per Browser families

Figure 7: Prediction value over other registered users (i.e., $user_i$'s learned model evaluates $user_j$'s canvasses). "adv" labels refer to result for Arch 2, while the others refer for Arch 1



(a) Variation of a global threshold in interval [0, 1]

(b) Variation of the size of a model-based interval around the accuracy of each model predicted at the learning phase.

Figure 8: ROC curves for varying global threshold (left) and varying model-based interval (right). Vertical lines indicate the False Positive Rate when the True Positive Rate scores 1.

nothing). Rather, we expect SWAT to be useful in a multi-factors context where it would help to prevent direct access to the attacker if, for example, a password is leaked. In summary, SWAT enhances the security by a complementary threat resilience to existing authentication systems, without lowering the user experience if the identity provider implements a transparent usage of the protocol. We suggest in the next section a couple of potentially interesting usage scenarios.

5 POSSIBLE USAGE SCENARIOS

5.1 Web Authentication

A SWAT API could be built for various web frameworks. This work provides one implementation for the high-level Python web framework Django. With such an API, web developers could enhance the security of their application with multi-factor authentication, such as password + SWAT. With both authentication mechanisms covering different threat models, the benefits would be transparent for the users.

Another interesting use case would be to enhance the industry-standard protocol for third party authentication (Oauth2 [11]) with the ability to authenticate with SWAT. Identity providers such as Facebook, Google or Github are well-positioned and own the resource to develop and deploy our protocol in a transparent way for their users. For example, users may perform the learning phase when using the services offered by these Internet giants, and benefit from the increased security when connecting to others web services through Oauth.

5.2 IoT Authentication

Many IoT devices interact with smart phones and could benefit from our protocol inside dedicated applications as well. However, this would require the learning phase to be remote (not on the IoT device) and the learned model to be shared with the IoT device

afterwards. In many settings, weak but user-friendly authentication is desirable and this protocol offers such an opportunity.

5.3 Game Launcher

The Game industry has known a huge growth since the free-to-play model was introduced. Hacking and frauds is now common against gamer accounts. One way to improve the current landscape would be the increase the authentication security in game launcher to prevent account theft and in-game content theft. Several game companies, like Blizzard, offer in-game bonuses when users improve their account security with a hardware token. Nevertheless, hardware tokens are not widely used. SWAT could be an interesting opportunity to deploy massively multi-factor authentication on game launchers, while not damaging the user-friendliness of their game.

6 RELATED WORK

Since Peter Eckersley's initial "device fingerprinting" analysis [4], work has spawned with various categories of goals. Laperdrix *et al.* [17] consider a few: *Client-side diversity* which tries to evaluate uniqueness of fingerprinting methods [4][17][9].

Fingerprinting adoption on the web and server-side scripts which evaluates the surface of the web under fingerprinting methods. Under this category, FPDetective [2] identified new scripts and techniques to fingerprint users. Acar *et al.* [1] investigated canvas fingerprinting, evercookies and use of cookie syncing on a large-scale study. Englehardt and Narayanan [6] pushed the analysis to 1 million websites, found sophisticated fingerprinting methods and showed how to crawl and analyze a huge amount of information in a reliable way.

Advanced solutions to collect additional fingerprintable attributes is the category focusing on developing new fingerprintng methods. Canvas fingerprinting [19] or leaking battery information through the HTML5 API [23] are good examples of work in this category, and show how much we deeply need to better understand our application environment to improve privacy.

Resilience to fingerprinting methods is a category of work where anti-fingerprinting strategies are investigated but could potentially break the accuracy of our authentication protocol [22][3][16]. As discussed in Future work, Section 7, defending against tracking while preserving beneficial usage of the output variation of the canvas API might be an interesting research direction.

Closer to what is performed in this paper is the category of *frictionless authentication* [24] where the users authenticate without performing any intentional authentication actions [20]. The usability obtained in frictionless authentication is granted through continuous monitoring of the claimed identity to spot any divergence from the expected user behavior. However, this raises some serious privacy concerns.

Our work, as far as we know, is the first attempt to use the output variation of some HTML5 API to the benefit of the user.

7 FUTURE WORK

7.1 Increasing drawing variations

Our work leverages on Mowery *et al.* [19] text and text's scene variation to distinguish users. Recent work by Gómez-Boix *et al.* [9] shows how to draw more complex canvas which would likely increase the accuracy of our authentication protocol. Smart utilization of the Canvas API to cover a broader surface of the software and hardware stack can be considered as a side problem that may contribute to various works, like our proposal, but also to the understanding of browser fingerprinting diversity.

7.2 Privacy please?

Different service could collude and match users as being the same identity. Future work could address this privacy issue and complicate such intersection attack given sets of canvas collected by different services. Such a privacy property seems however difficult to obtain with a stateless client side protocol. The current research for privacy-preserving browsing experience attempts to break the linkage between sessions of the output of HTML5 API [16]. This line of work would break our protocol and its benefit, but we could imagine opt-in per-service when the usage of distinguishable information is intended for a true user-centered benefit, which is in line with the current approach of browsers (e.g., Firefox opt-in to enable canvas extraction).

7.3 Active attackers

Our current analysis is based on canvasses produced in a genuine way. An adversary may proceed in a different way: when facing a challenge response, and assuming that he has access to previous responses to different challenges he may try to combine those responses in order to forge a fresh one. More generally, an adversary could use image processing software to create a canvas forging system. Investigating this direction would be interesting to evaluate the security of SWAT in the more demanding setting of active adversaries.

8 CONCLUSION

We presented SWAT: an authentication protocol with a focus on usability, server-side performance and ease of deployment. SWAT provides interesting security properties in a multi-factor authentication scenario, protecting against passive eavesdropping and replay attacks as well as defending against stolen credentials. We highlighted in the analysis the existence of a trade-off between authentication accuracy and resource endowment, and showed that we achieve a high TPR for a low FPR even for small resource endowment. We discussed SWAT's security usefulness in a multi-factors context and for various scenarios. A SWAT protocol is not bound the the strict use of the Canvas API. Any other API which can take an input, and output variations depending on user specific software or hardware would be interesting candidate to increase the authentication accuracy.

ACKNOWLEDGMENTS

We would like to thanks the anonymous reviewers for their feedbacks. We are also thankful to xsi.io's team for their valuable expertise and advises in Web technologies. This work was partially supported by the by the Brussels Region Research Funding Agency through the Innoviris/Secur'IT project C-Cure and by the Walloon Region through the "Plan Marshall 4.0" project Digitrans (convention number 7618).

REFERENCES

[1] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. 2014. The web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 674–689.

[2] Gunes Acar, Marc Juarez, Nick Nikiforakis, Claudia Diaz, Seda Gürses, Frank Piessens, and Bart Preneel. 2013. FPDetective: dusting the web for fingerprinters. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 1129–1140.

[3] Peter Baumann, Stefan Katzenbeisser, Martin Stopczynski, and Erik Tews. 2016. Disguised chromium browser: Robust browser, flash and canvas fingerprinting protection. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*. ACM, 37–46.

[4] Peter Eckersley. 2010. How unique is your web browser?. In *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 1–18.

[5] Steven Englehardt, Chris Eubank, Peter Zimmerman, Dillon Reisman, and Arvind Narayanan. 2015. OpenWPM: An automated platform for web privacy measurement. *Manuscript* (2015).

[6] Steven Englehardt and Arvind Narayanan. 2016. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1388–1401.

[7] Firefox. 2017. Prompt (w/ Site Permission) before allowing content to extract canvas data (Tor 6253). https://bugzilla.mozilla.org/show_bug.cgi?id=967895.

[8] Kunihiko Fukushima and Sei Miyake. 1982. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*. Springer, 267–285.

[9] Alejandro Gómez-Boix, Pierre Laperdrix, and Benoit Baudry. 2018. Hiding in the Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale. In *WWW 2018: The 2018 Web Conference*.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[11] Dick Hardt. 2012. *The OAuth 2.0 authorization framework*. Technical Report.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2980–2988.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[14] Keras. 2018. Guide to the Functional API. https://keras.io/getting-started/functional-api-guide/.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[16] Pierre Laperdrix, Benoit Baudry, and Vikas Mishra. 2017. FPRandom: Randomizing core browser objects to break advanced device fingerprinting techniques. In *International Symposium on Engineering Secure Software and Systems*. Springer, 97–114.

[17] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. 2016. Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints. In *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 878–894.

[18] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[19] Keaton Mowery and Hovav Shacham. 2012. Pixel perfect: Fingerprinting canvas in HTML5. *Proceedings of W2SP* (2012), 1–12.

[20] Mustafa A Mustafa, Aysajan Abidin, and Enrique Argones Rúa. 2018. Frictionless authentication system: Security & privacy analysis and potential solutions. *arXiv preprint arXiv:1802.07231* (2018).

[21] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 807–814.

[22] Nick Nikiforakis, Wouter Joosen, and Benjamin Livshits. 2015. Privaricator: Deceiving fingerprinters with little white lies. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 820–830.

[23] Łukasz Olejnik, Gunes Acar, Claude Castelluccia, and Claudia Diaz. 2015. The leaking battery. In *Data Privacy Management, and Security Assurance*. Springer, 254–263.

[24] Vera Rimmer, Davy Preuveneers, Wouter Joosen, Mustafa A Mustafa, Aysajan Abidin, Enrique Argones Rúa, et al. 2018. Frictionless Authentication Systems: Emerging Trends, Research Challenges and Opportunities. *arXiv preprint arXiv:1802.07233* (2018).

[25] Florentin Rochet. 2018. Github repository for project code. https://github.com/frochet/SWAT.

[26] Florentin Rochet. 2018. Interface used for data collection. https://perceval.elen.ucl.ac.be.

[27] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[28] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. 2015. Going deeper with convolutions. Cvpr.

Before and After GDPR: The Changes in Third Party Presence at Public and Private European Websites

Jannick Sørensen
Aalborg University Copenhagen
Copenhagen, Denmark
js@cmi.aau.dk

Sokol Kosta
Aalborg University Copenhagen
Copenhagen, Denmark
sok@cmi.aau.dk

ABSTRACT

The commencement of EU's General Data Protection Regulation (GDPR) has led to massive compliance and consent activities on websites. But did the new regulation result in fewer third party server appearances? Based on an eight months longitudinal study from February to September 2018 of 1250 popular websites in Europe and US, we present a mapping of the subtle shifts in the third party topology before and after May 25, 2018. The 1250 websites cover 39 European countries from EU, EEA, and outside EU, belonging to categories that cover both public-oriented citizen services, as well as commercially-oriented sites. The developments in the numbers and types of third party vary for categories of websites and countries. Analyzing the number of third parties over time, even though we notice a decline in the number of third parties in websites belonging to certain categories, we are cautious about attributing these effects to the general assumption that GDPR would lead to less third party activity. We believe that it is quite difficult to draw conclusions on cause-effect relationships in such a complex environment with many impacting factors.

CCS CONCEPTS

• **Information systems** → **Web mining**; **Traffic analysis**; **Information retrieval**; *Data mining*; • **Security and privacy** → Social aspects of security and privacy.

KEYWORDS

GDPR, Web measurement, Third-Party web server, Tracker, Public Services, General Data Protection Regulation

ACM Reference Format:
Jannick Sørensen and Sokol Kosta. 2019. Before and After GDPR: The Changes in Third Party Presence at Public and Private European Websites. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313524

1 INTRODUCTION

Mainly driven by privacy concerns, much research has been dedicated on examining the presence and implications of contacts with third-party servers when users browse websites. EU's General Data Protection Regulation (GDPR[1]) aims to give users more control over the collection and distribution of their personal information, also on websites. As one sign of the relevancy of GDPR to websites, the European advertiser's lobby organization (IAB Europe[2]) has proactively assisted and informed their members on the implications of GDPR [14], e.g. by providing an open source consent framework. Before the enforcement of GDPR, the future of third parties and advertising was also a hot debate topic in the advertising industry [5, 17, 26, 36]. As every provider serving users from the EU were compelled to review their use of cookies and other person-identifiable data-exchange with third parties, the assumption in the industry was that GDPR would lead website providers to review and reduce the number of third party servers [25]. Correspondingly, a number of companies started to offer the monitoring of third party activity on behalf of website providers, as well to legalize the data collection by obtaining user consents. The much ado in the web industry in relation to GDPR provokes thus the question whether the activity level of third party HTTP responses changed with GDPR. Do users meet fever third party servers when browsing the WWW?

This paper presents the results of an ongoing longitudinal study of strategically selected websites, from European countries in and outside the EU/EEA, and from USA. The websites have been selected to represent those categories that have been found to show either very high or very low numbers of third party[3] URLs [10]. Earlier works have observed that sites which belong to government organizations, universities, and non-profit entities have a low number of TPs, while sites with editorial content have a high number of TPs, since *"they are pressured to monetize page views with significantly more advertising"* [10]. In this work, we examine this observation with a longitudinal dataset spanning across the commencement of GDPR and across different countries.

Our intuition is that the picture is more differentiated than what has been studied so far, as many high-traffic public organizations' websites, such as public service media companies or public transportation companies, feature advertising and are deeply integrated in media ecosystems [22, 30, 37]. We analyze if these citizen-serving websites may thus, with respect to the number of TPs, have similarities with commercial high-traffic websites. As an example, public service media organizations promote themselves as *"islands of trust and quality in the multimedia environment"* [7]. They argue that citizens have high level of trust in public service media [8, 9]. Arguably, websites published by public entities should be expected to protect users' privacy particularly well, in order to maintain

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313524

[1] EU General Data Protection Regulation (GDPR) – https://eugdpr.org/
[2] IAB Europe – https://www.iabeurope.eu/
[3] In the rest of the paper we refer to third parties as "TPs".

the integrity of the public organization as being independent and sovereign in relation to private actors. As public organizations are funded by tax or – in cases of public service media – via a license fee, involvement with advertising blurs the public/private distinction. Indeed, advertising in public service broadcasting is subject to heavy regulations [33]. In a digital context, users may pay twice for the same public service: first via the tax and then with their data being transferred to the third party web services. The extended discussion of public organizations' use of third party services is however not the aim of this paper, but to test our assumption of websites morally obliged to a low number of third party contacts, we include law firms as examples of private websites that could have an interest in signalling a high level of privacy.

Our contributions: Having collected HTTP requests and responses 21 times over eight months for a strategic selection of 1250 European and US websites, we can analyze the fluctuations in number of TPs before and after the commencement of GDPR on May 25, 2018. Thereby, we provide empirical evidence for the discussion whether GDPR would lead to fewer TPs. At a general level, we can conclude that the amount of TPs on web pages have slightly declined, but the picture is more complex and contradictory when we study the developments for respectively categories of sites and TPs. This finding adds details to findings provided by [21] and [10]. We cannot support the general assumption that the GDPR has led to fewer TPs, since we cannot find strong evidence for any correlation. As a second contribution, we characterize the differences between websites offered by public and private organizations from the TP perspective, initiating a discussion on public organizations' use of TPs. Finally, we intend to publish the data to allow other research groups further analysis.

2 RELATED WORKS

The involvement of third-party (TP) advertisers or trackers in websites and mobile apps have interested many researchers [1, 2, 6, 10, 18, 20, 29, 31, 32, 34, 35]. In 2014, Acar et al. presented one of the first large-scale studies of third-party trackers in web pages, showing that *the web never forgets* [1]. The authors show that trackers use different and complex ways to integrate in a website, so that it is difficult to distinguish them accurately from genuine content. In a more recent work, Lerner et al. perform an *archaeological analysis* of tracking practices in the web, analyzing websites from 1996 to 2016 from the Internet Archive[4], showing that the first web trackers date back to 1996 [18]. Unsurprisingly, the study finds that the number and complexity of TPs have increased since their first appearance in 1996. Furthermore, through a large-scale automated analysis of third-party privacy policies, Libert et al. show that reading and understanding privacy policies is far from easy, and users are not presented with the needed information allowing them to take relevant decisions on what data to share [20].

One of the largest measurements of third-party presence was conducted in January 2016, where Englehardt and Narayana developed OpenWPM[5], an automated script that uses Firefox web-browser to visit websites through the Selenium[6] automation tool. OpenWPM collects and stores several browsing data, such as HTTP requests and responses, in a database for later analysis. The authors used the tool to visit one million unique websites via 90 million web page requests [10]. The selected websites represented the one million most popular websites in the world, according to the internet measuring service Alexa[7]. Following the categorization of websites made by Alexa, the authors found that news websites are among those with the highest number of third-parties, while websites from public organizations, NGOs, and universities are among those websites with the lowest number of third-party web-server contacts. In this study, the authors found over 81,000 unique third-party servers. However, they notice that only 123 of these were present at more than 1 percent of the visited sites [10].

Many studies have used OpenWPM for crawling and analyzing the web. Among these, Binns et al. assess the power of third party trackers through business collaboration [2], Mazel et al. compare privacy protection methods [23], Brookman et al. analyse cross-device tracking [3], and recently, Rodriguez et al. have analyzed tracking cookies in the EU [27].

To understand the inter-dependencies in the business ecosystem of media production, Lindskow analyzed the TPs presence at 41 US-American news publishers, identifying 1,356 business partners [22]. The study shows that *digital news publishers function as integrators of resource flows from hundreds of business partners of different types and sizes and that each of these partnerships involves complex resource exchanges and strategic dependencies.* The author's focus is thus not on privacy, but on business networks. However, they provide useful insights into dividing the TPs in different categories, identifying and analyzing six categories, as follows: *i)* News content elements, *ii)* News content, *iii)* Editorial tools, *iv)* Measurement, *v)* Advertising, and *vi)* Supporting resources.

Starov et al. analyzed the identifiers that third-party trackers store in users' devices to uniquely distinguish between users, showing that there are many websites that use the same identifiers, allowing the authors to cluster websites that looked unrelated to each other [32]. Falahrastegar et al. [12] make an early contribution to map the anatomy of the third party ecosystem, with a focus on user tracking.

Recently, with the advent of GDPR, many researchers have started analyzing the possible effects that GDPR is having on the web. Libert et al. [21] present a fact-sheet on third-party cookies of 204 news websites from seven EU countries. The pages were visited in April and July 2018 (pre- and post-GDPR), using a crawling system developed by one of the authors in a previous work [19]. The authors observe that the percentage of web pages that contain third-party content decreases with only two percent from April to July 2018, dropping from 41 to 40 third-parties per page. Conversely, they notice that there is a drop of 22 percent in the number of cookies, with variations among the seven countries.

Very recently, Iordanu et al. performed a massive analysis of a large-scale traffic flow from final users to third-party trackers [16]. The authors implemented a browser extension and distributed it to 350 real users, creating this way a reliable dataset. Then, they generalized their analysis methodology by extending it to datasets from four large Internet Service Providers (ISPs) containing data

[4] https://archive.org/
[5] https://github.com/citp/OpenWPM
[6] https://www.seleniumhq.org/
[7] https://www.alexa.com/topsites

from more than 60 million users. The study was performed during a period of four months before the implementation of GDPR. Among other results, this work shows that around 3% of the flowing traffic is related to sensitive user data, which might violate the GDPR requirements. Nevertheless, the authors show that trackers collecting sensitive data of users in the EU keep them primarily within the EU (in 90% of the cases).

In this work, we perform a large-scale analysis of third-party HTTP responses within a period of time that covers the pre- and post-GDPR era, specifically from February to September 2018. Firstly, we manually select more than 1300 websites of different categories from 39 countries, so that we have a reliable and controlled coverage of countries of interest and website purpose. Then, we use OpenWPM [11] to automatically visit the websites and some randomly selected sub-pages, creating a dataset composed of 21 crawls/harvests[8]. We then analyze the possible effects of GDPR, by comparing the changes of the TPs characteristics in the visits performed before and after GDPR.

3 RESEARCH DESIGN

In this section, we give a description of the tools we use to perform the data collection and the methodology we follow for selecting the websites to analyze, the cleaning of the data, and the classification of the third-parties.

3.1 Selection and classification of websites

The search for websites to be analyzed reflects our strategic selection of those website categories where third-party trackers have shown either high or low interest.

3.1.1 Identifying website categories. Considering the analysis and results of Englehardt et al. [10] and Libert et al. [21], we look for types of websites where either a high or a low number of TPs should be expected. From these studies, we can observe that there is a significant correlation between the ownership of the website, as being either private or public, and the number of TPs [10]. Indeed, while a user of commercially-oriented websites may expect a high presence of third party web servers, a user of a publicly owned citizen-oriented websites should expect none or very few third parties. As such, our website selection process reflects the distinction between websites offered by **public** and **private** organizations as a first criterion.

Then, the second selection criteria is the **country of origin** of the websites. We target primarily websites from the European countries, distinguishing between the EU, EEA (Extended Economic Area[9]), and non-EU sites. However, we also include few websites located outside Europe, mainly from US. To identify the country of a website we use its country code top-level domain, and if that is not available we perform a WhoIs look-up[10] to find the registrant organization's country.

Then, we identify eleven **categories of websites** that people visit more often, starting from and extending the list proposed in [10]. So, we include examples of shopping sites, hotel, travel and holiday sites, gaming and gambling sites, video-on-demand sites,

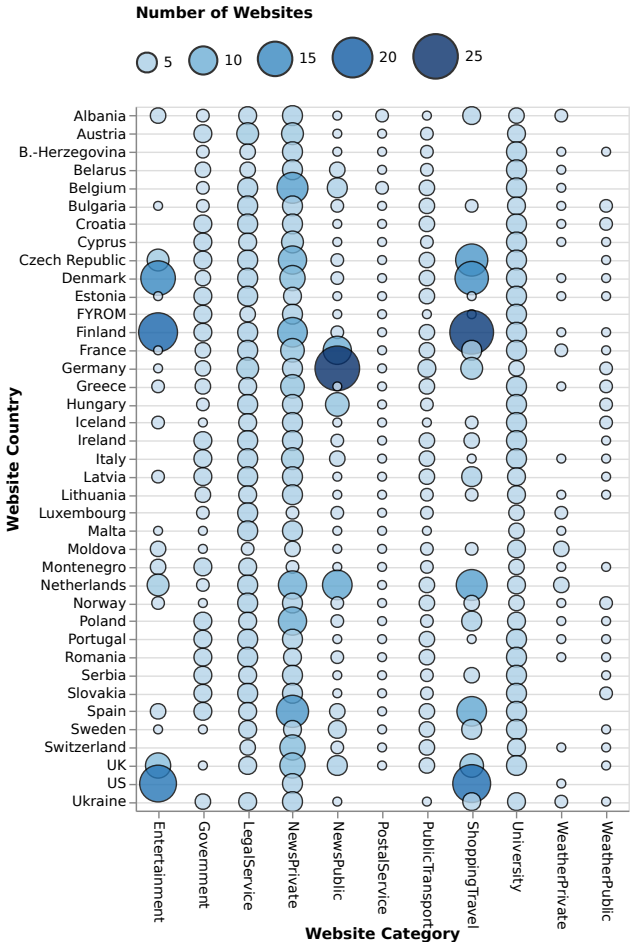


Figure 1: Number of websites per country and category selected for the analysis in this work.

general entertainment, and life-style magazines. In the analysis we have collected all these sub-categories in a single main category "Shopping and Travel". Moreover, we collect websites from European countries from law firms and attorneys, assuming that we would find an example of private websites with a low number of TPs. Furthermore, we consider governmental organizations, universities, public transportation, postal services, and state-funded weather services as public services websites, assuming that we would not observe a high number of TPs. For governmental organizations we specifically selected ministries and public administration sites.

Finally, we consider the news websites, which, differently from previous works, we explicitly divide them into public and private. We make this distinction considering the fact that there are huge differences among news/media organizations, particularly in Europe, where some are publicly owned and financed via a license fee, tax and in some cases also advertisement, whereas others are privately owned and financed by subscription, advertisement and in some cases public support. Similarly, we also distinguish between public weather websites from governmental/public organizations,

[8] In this paper, we use the terms *crawl* and *harvest* interchangeably.
[9] www.efta.int/eea
[10] https://whois.icann.org/en

and websites from private providers. To the best of our knowledge, this is the first work that makes this clear distinction.

As such, we identify the following categories: *i)* Entertainment, *ii)* Government, *iii)* Legal Services, *iv)* News Private, *v)* News Public, *vi)* Postal Services, *vii)* Public Transport, *viii)* Shopping and Travel, *ix)* University, *x)* Weather Private, and *xi)* Weather Public.

3.1.2 Selection of websites. To find the relevant websites for each category, we have used various methods: via search engines, via user involvement, via related research, and via lists of membership. Differently from previous works, which use Alexa[11] to classify a website as belonging to a category, we manually look at each website and classify it with human intervention. We believe that, compared to Alexa's lists, the manual classification allows us to have a higher level of certainty about the websites' category, given that we do not have access to Alexa's criteria for the categorization.

To identify public service media news websites we use the list of members of the European Broadcasting Union, a lobby organization for public service media[12]. As for private media, we take the top-5 most popular news sites listed in the Reuters Institute Digital News Report 2017 [24]. For countries not covered in the Reuter's report, we use the Alexa web service[13]. For websites in other categories than news, we manually identify them using Google Advanced Search[14], filtering by region/country, getting the top 5 websites not already in our list. Moreover, to broaden the collection of websites further, we perform a web survey asking users from our personal network to mention the most popular websites from their countries in the following categories: news, lifestyle/magazine, shopping, travel & hotels, entertainment, and gaming. From 17 users we obtained 279 unique websites: 62 news sites, 42 magazine/lifestyle sites, 67 shopping sites, 62 travel & hotel and accommodation sites, 23 entertainment sites, 20 gaming sites, and 3 sites outside these categories.

Overall, we were able to carefully select a total of 1,363 websites from 39 different countries, divided into 11 categories, as shown in Figure 1. As can be observed from the distribution, we are aware that not all categories are equally well represented for all countries, as well as some categories are represented with more websites in some countries than in others. This bias is an implication of our main study goal, which aims at having at least five websites from each country in these categories: *i)* private news, *ii)* universities, *iii)* ministries, *iv)* law firms. As for the categories of public transportation, postal service, and public weather, for some countries we found fewer sites than in some other. For public service media we included all public service organizations, even if multiple as in the case of Germany. The variability in the other categories, i.e. Entertainment and Shopping & Travel, is then due to user involvement.

3.1.3 Selection of subpages. To ensure a broad capture of possible TPs on the websites we have generated a list of subpages to visit. We have done so by collecting random URLs from the 1,363 sites, producing in total +50,000 URLs. From these, we randomly select ten subpages to be visited for each site. On average, we have visited 9.35 pages from each site. Some sites do not have subpages, since one single page provides all the necessary information. This is the case e.g. for weather information and public transport (journey planners and timetables). In total, the script visited 12,778 subpages from the 1,363 websites for each of the 21 harvests, capturing all the HTTP requests and responses. For each website and subpage visit we collect the set of unique third party URLs.

[11]https://www.alexa.com/topsites
[12]https://www.ebu.ch/about/members
[13]https://www.alexa.com
[14]https://www.google.dk/advanced_search

3.2 Data collection and cleaning

3.2.1 Sampling period. We selected a long sampling period for the large-scale crawling, in order to be able to detect the effects of GDPR in terms of possible and expected deviations in the number of different types of TPs on the different categories of websites over time. As such, the sampling period spans across the GDPR commencement, which was on May 25, 2018, with 8 harvests from February to May and 13 harvests from May 25 to September.

3.2.2 Website harvesting. To perform the automatic website crawling we setup a virtual machine (VM) with 5 CPUs and 8GB of memory running Ubuntu. We installed the widely used OpenWPM framework and configured it to use four vanilla Mozilla Firefox browsers in parallel, meaning that we did not in any case register or log in with a user account while visiting the websites. Moreover, we did not give consent to any storage of cookies or other types of privacy-related requests, as the interaction with the web pages only took place automatically via the script. In this way, the behavior of the script consequently imitates a user who ignores all user privacy consent messages from the websites. However, we did not install any extension on the browser that would stop the websites from storing or reading cookies. We did not erase any cookie after a harvest was performed, so that on next visits the websites could behave normally by accessing the stored information. Finally, it is worth mentioning that the physical machine on which the VM was deployed is located in the EU, meaning that the visited websites must comply with GDPR.

3.2.3 Data cleaning and selection for analysis. Through the 21 harvests, we collected in total 31,493,575 HTTP responses from the 12,778 subpages of the 1,363 websites, averaging 1,431,526 HTTP responses per harvest.

Before analyzing the data, we performed a cleaning process that led to 83 websites to be removed from the dataset. The reasons for excluding these URLs are the following: *i)* Some of them redirected to a website already in our collection, thus risking to produce duplicated entries; *ii)* Some of them, suggested by the survey users, were from countries outside Europe (Japan, Korea, Hong Kong, Panama), from which we have no other or very few websites; *iii)* Finally, to consolidate our eleven main analytic categories of websites (see Figure 1), we have discarded few websites that did not fall into any of the main categories.

Thus, in the rest of the paper we analyze and present our findings regarding 1,250 websites from 39 countries from the EU, EEA, countries outside EU/EEA, and from the USA, and divided into 11 categories, as shown in Figure 1.

3.3 Categorization of the TPs

In this section we present some results related to the overall number of third-parties observed in our datasets and to the methodology we

TP Category	No. of Unique URLs
Advertising	1382
Publisher	341
Distribution Technology	237
Analytics	72
Retail	170
Content	209
Programming	82
Plug-in	45
Social media	30
Editorial	31
Privacy	6
Search Engine	27
Cybersecurity	16
Unidentifiable	347

Table 1: Number of unique TP URLs for each TP category.

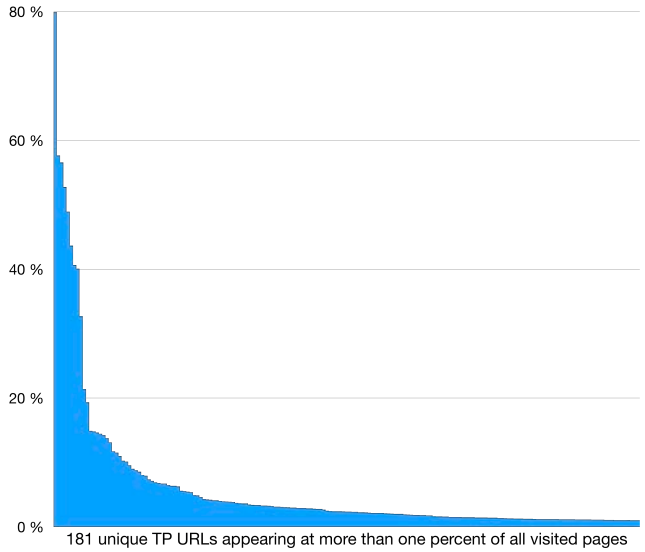


Figure 2: The presence in percent for the most omnipresent TPs of all pages.

pursued on classifying them into different categories. To identify the third-party URLs we look at all HTTP responses and extract all those URLs that are different from the website URL that was being visited. From this analysis we find 3,128 unique third-party URLs.

Then, we manually categorize the TPs by visiting the unique URLs. If we find a readable website, we attempt to determine the type of services offered by the TP by reading their descriptions of services, products, and mission statements. If the original TP URL redirects to another website, e.g. a company website, we analyze the description of products and services of that company. Otherwise, if the URL is not accessible and returns an error message, such as *HTTP 404 Not Found* for example, we classify the TP URL as *Unidentifiable*. Following this procedure, we were able to identify with high accuracy the 14 TP categories listed in Table 1, where we also show the number of TPs per each category. The details of each category are the following:

Advertising contains all the various services related to sale of ads and analysis of user profiles, such as: Programmatic advertising services, Personalization services, Recommender systems, Real-time bidding platforms, Demand-side and Sell-side platforms, Data management platforms and data integration, Data brokers and data trading, Re-targeting systems, User trackers, Ad-servers, Advertising agencies, Ad-verification systems, Brand-integrity, attribution and anti-fraud systems, Marketing automation, Content & native advertisement services, Cross-device user identification systems, and Video-based advertising.

Analytics contains different types of services used to understand user behavior and gather user feedback: Audience measurement, AI-powered analysis of user behavior, Audience Intelligence, Semantic profiling, Audience research (qualitative), Customer flow, Marketing analytics, Quality of Service monitoring and Web performance optimization, Attention optimization tools for Publishers, and Customer feedback.

Content includes all types of elements shown on the web page, not being advertising. That includes content embedded from other websites, not part of the media company/organization.

Cybersecurity specifically focuses on services that perform internet infrastructure surveillance.

Distribution technology includes content delivery networks, cloud services, and streaming services.

Editorial contains services specifically aimed at editors, e.g. recommender systems designed for publishers.

Plug-in contains web services that integrate content from other services into the visited site.

Privacy contains services that monitor website compliance with GDPR and cookie use on the visited site.

Programming contains services that deliver code to the rendering of the web page.

Publisher encompasses all servers that are owned by the media organizations including collaborating media organizations.

Retail includes all-purpose web-portals, job-seeking portals, shopping platforms, real-estate brokers, and consumer products (advertisers).

Unidentifiable includes those URLs that do not return a valid and readable HTML page when visited in the browser, but a *404* or *forbidden* error message or a blank page. Our working assumption is that a large portion of the unidentifiable TPs are advertising related, delivering the ads to be displayed, measuring and verifying the exposure, or tracking the user across websites and other devices. This assumption is based also on an expert interview that we conducted with the CTO of an advertising technology company in December 2017[15].

4 RESULTS

Our research design allows analysis of the collected data along different dimensions. Indeed, the categorization of both the visited websites and third party URLs, combined with 21 samples (harvests)

[15] Jacob Knobel, former CTO of Densou.dk, also cited in [30]

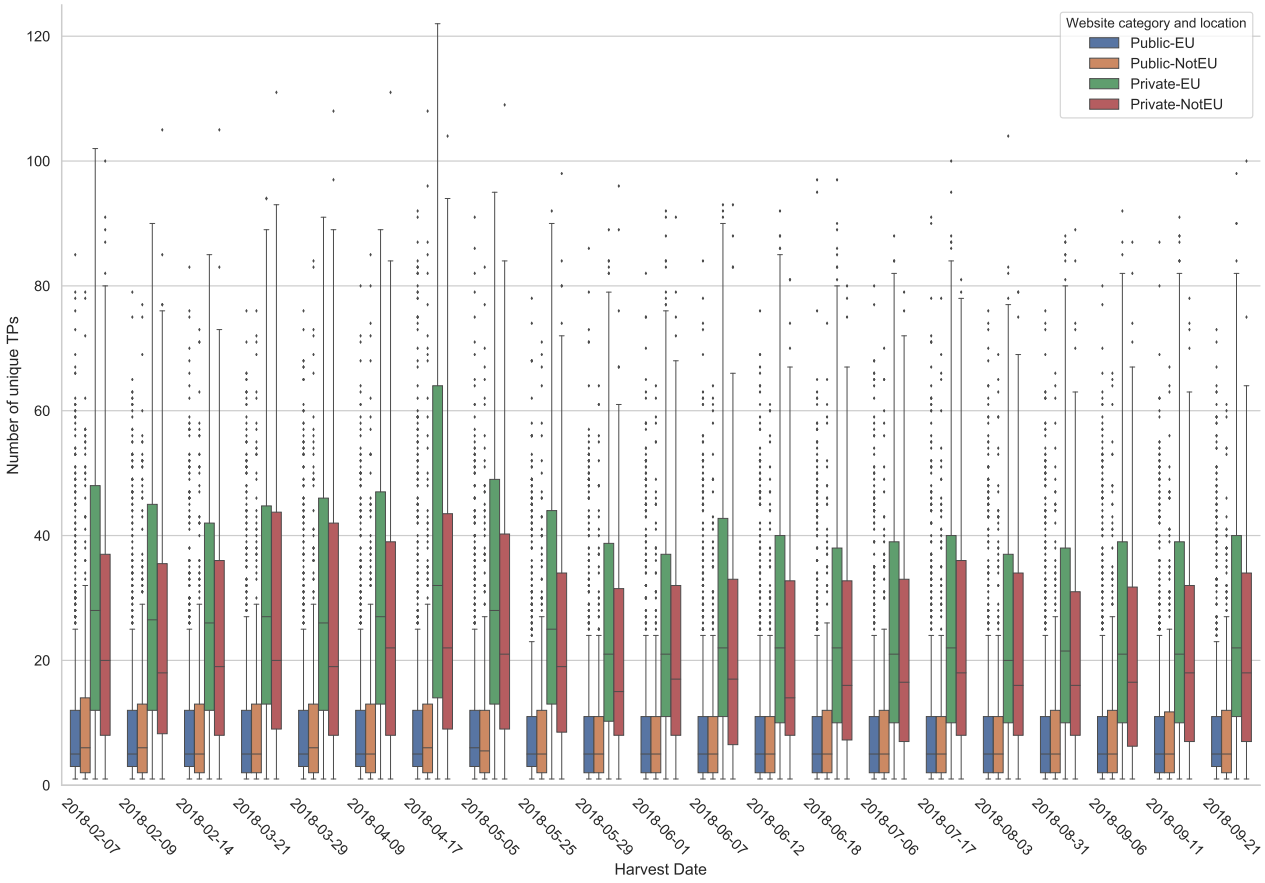


Figure 3: Number of TPs with respect to the website being public or private and its EU membership (EU/NotEU). In this graph we consider EEA countries together with the EU ones.

over a time span of eight months, gives us the opportunity for a detailed investigation over different feature combinations.

Firstly, as the sampling period spans across the commencement of GDPR, we can compare pre- and post GDPR levels for the presence of TPs. Moreover, the categorization of the visited sites into countries, groups of countries, and types of site, together with the categorization of the TPs, allows us to create a detailed picture of the developments. Furthermore, we can look at groups of sites, such as EU and EEA countries compared with sites from outside the EU/EEA area, and extend the investigation by also differentiating the websites in private and public ones. Finally, on a more detailed level, we can also look at the variations in the number of TPs for different categories of sites, for different countries, and for different types of TPs.

4.1 Analyzing the Number of TPs for Categories of Sites

In total, we found 3,128 unique third party URLs, appearing on at least one web page in all our 21 harvests. During all the observations, we found that:

- After May 25, 675 unique TPs disappeared, of these 63 were present in all harvests before May 25.
- 644 new unique TPs appeared after May 25, of these 36 were present in all harvests after May 25.
- 2096 TPs were present in all 21 harvest on at least one website.
- 408 TPs appeared only in one harvest at one website.

If we consider a list with the top-20 TPs, we find similarities with earlier works showing the dominance of the giant companies [10, 21, 28]. From our data, we find that this includes nine TP URLs controlled by Google *(google-analytics.com, doubleclick.net, googleapis.com, google.com, gstatic.com, google.dk, googletagmanager.com, googlesyndication.com, googletagservices.com, googleadservices.com)*, two TPs controlled by Facebook *(facebook.com, facebook.net)*, Amazon's CDN *(cloudfront.net)*, and the competitor CDN *(cloudflare.com)*, the advertising companies Adnexus *(adnxs.com)*, *criteo.com*, *adform.net*, the analytics companies *scorecardresearch.com (TMRG) and gemius.pl*, and the omni-present *twitter.com*. These URLs represent the beginning of a very long tail where only 151 TPs have a share of one percent or more of the pages, while the remaining 968

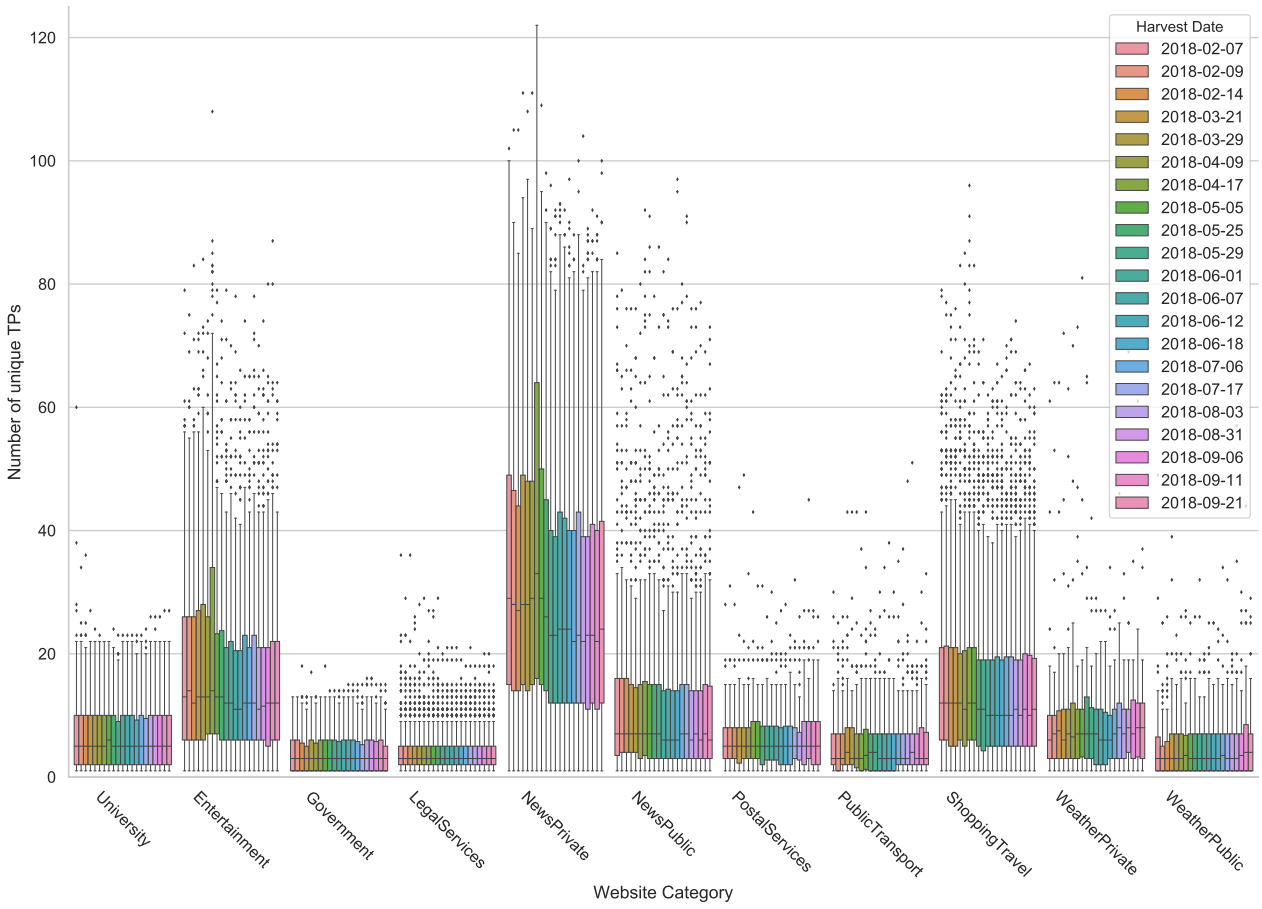


Figure 4: Number of TPs per website category with respect to the harvest date.

TPs have a share of less than one percent. The cut long tail of the top 151 TPs is presented in Figure 2.

In Figure 3, we analyze the variations of the TPs by dividing the websites into private/public and EU/NotEU, where EU in this case includes also the EEA countries, while *NotEU* includes websites from countries outside the EU/EEA. The figure shows, for each harvest, the distribution of the number of TPs embedded in each website represented as box plot, showing the minimum, the median, the maximum, the first and third quartile, and the outliers. From the figure, we can firstly see that the **public websites have far fewer TP URLs than the private ones** in all the harvests. Secondly, we can see that there is a **difference between private EU and NotEU websites**, while there is not much difference when considering the public websites. When considering the variations over the observed time, we can see a rather **stable picture for the public sites**, regardless being from the EU or not. In comparison, the **private sites present large fluctuations**, but with fewer outliers. Moreover, these data suggest that the **private websites present a slight decrease of the number TPs** after the commencement of GDPR, while the public ones are hardly affected.

Next, we continue our investigation trying to identify the responsible private websites that cause the decrease in the number of TPs after GDPR. In Figure 4, we present the distribution of the number of TPs for website category, grouped by harvesting date. As also observed above, this figure shows clearly and with more details that **public websites** (i. e. categories *University, Government, PublicTransport, PostalServices, NewsPublic,* and *WeatherPublic*) **present a low level of TPs and a low level of fluctuations**, with the exception of some outliers. Of these public categories, the news websites present the highest number of TPs and the highest variations. When it comes to **private websites**, *LegalServices* present a similar constant low level of TPs as the public ones, followed by the *WeatherPrivate*, while ***Entertainment, NewsPrivate,* and *ShoppingTravel* present the highest number and variations of TPs**.

To quantify the variations over time, we perform the linear regression of the average number of TPs for all website categories for each harvest. The regression lines are presented in Figure 5 and more details are presented in Table 2. From these results we can confirm that the ***NewsPrivate, Entertainment,* and *ShoppingTravel* categories show a clear decline in average numbers**

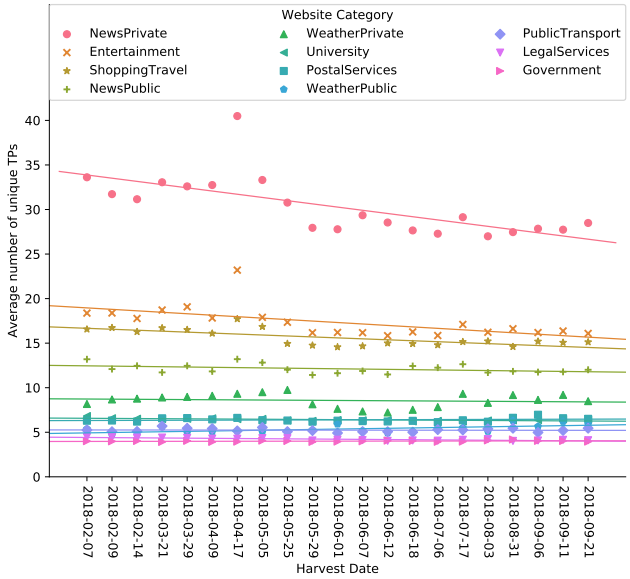


Figure 5: Average number of TPs per each website category during the harvesting period. The line represents the linear regression of the number of TPs, showing the tendency of the changes. The slope coefficients and the intercepts for each line are given in Table 2.

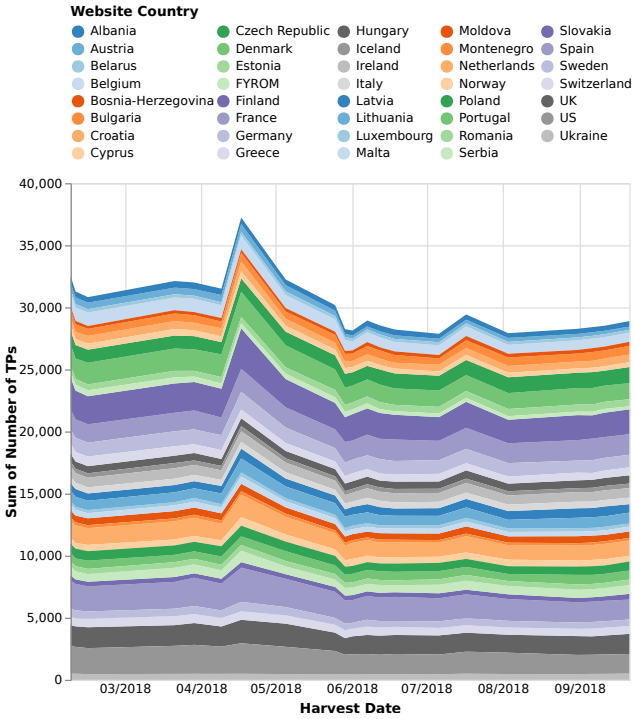


Figure 6: The total number of unique TPs at websites from 39 countries over time.

Private/Public	Website Category	Slope	Intercept
Private	NewsPrivate	-0.360	35.0
Private	Entertainment	-0.164	19.5
Private	ShoppingTravel	-0.107	17.0
Private	LegalServices	-0.019	4.5
Private	WeatherPrivate	-0.016	8.8
Public	NewsPublic	-0.033	12.6
Public	University	-0.015	6.6
Public	PublicTransport	-0.002	5.3
Public	Government	+0.003	4.0
Public	PostalServices	+0.008	6.3
Public	WeatherPublic	+0.043	4.8

Table 2: Coefficients of the regression lines of Figure 5.

of unique TPs per site visit, while for the other categories the decline is small. Moreover, for three categories of public websites, namely *PostalServices*, *Government*, and *WeatherPublic*, we see a slight, although we believe insignificant, growth in the number of unique TPs per site.

4.2 Countries Perspective

As a next analysis, we consider the individual countries and try to see the effects of GDPR over time. In Figure 6, we present the developments in the total number of unique TPs for each site we have visited from the different countries. This graph shows again that overall there has been a slight decline of the total number of TPs, but does not easily allow for a detailed quantification of the changes for each country. For this reason, we calculate for each country the average number of TPs for the harvests performed before and after GDPR and represent the changes as a percentage in Figure 7. The results show big variation among countries, with Estonia having the biggest growth and Austria having the biggest decline. 21 EU countries have fewer TPs after GDPR, while 7 EU countries have more TPs. Both EEA countries, Norway and Iceland, present a decline. Among the eight European countries outside EU/EEA, seven have a decline, while only Bosnia-Herzegovina has a growth.

4.3 Analyzing the distribution of the TPs

In this section, we present the analysis of the distribution of TPs in the different categories and the changes in each category over time (refer to Section 3.3 and Table 1 for more details about TP categorization). In Figure 8, we show a stream graph of the total number of TPs observed during each harvest. From the figure we identify major fluctuations in the *Unidentifiable* and *Advertising* third party categories. This is obviously due to the fact that these are the biggest categories, so it is easier to spot the changes.

To quantify the variations as we did for the countries, we look at the changes in percent and compare the visits after GDPR with those from before. The results for the categories with a declining trend are the following: *Cybersecurity (-51,16%), Privacy (-34,07%), Unidentifiable (-18,56%), Analytics (-11,28%), Advertising (-11,26%), Retail (-10,34%), Content (-6,50%), Distribution technology (-5,91%),*

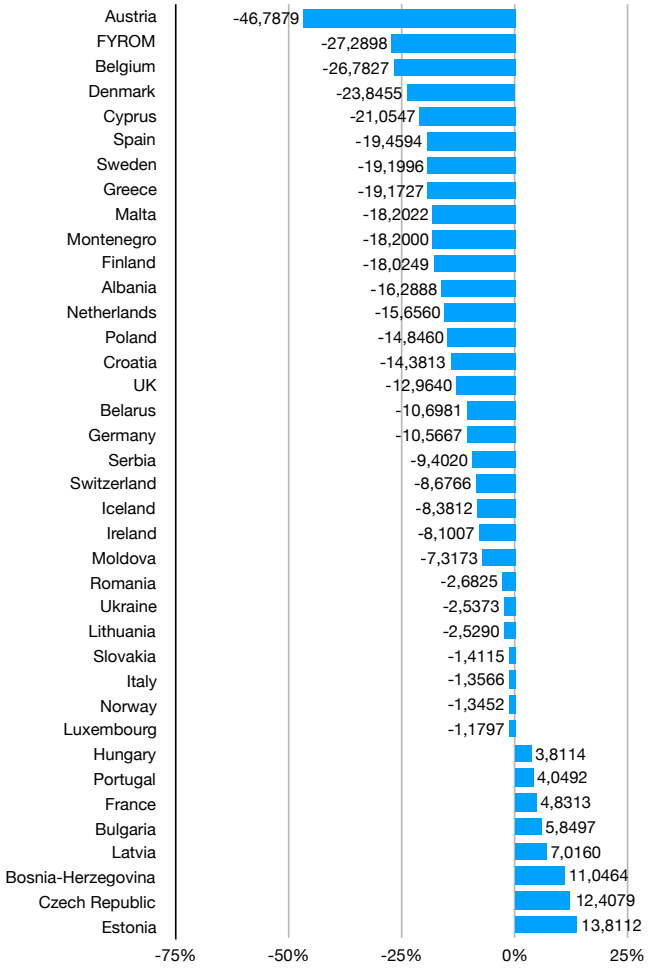


Figure 7: Comparing pre- and post May 25 average numbers of unique TPs for countries, listed as change in percent.

Social Media (-5,79%), *Publisher (-5,77%)*, and *Plug-In (-2,07%)*. We see a growth only in the categories *Programming (+2,73%)* and *Search Engines (+8,32%)*. However, as some of the categories have very few unique TPs the above changes should be considered with care. When considering single websites, we noticed that not all websites have a fluctuating number of TPs: for 241 websites (91 private and 148 public), the number of TPs is constant for all visits.

4.4 The Relation between Categories of Websites and TPs

The advantage of categorizing both websites and TPs is that we can also examine their mutual relation. Figure 9 shows, with a logarithmic representation, the occurrence of unique TPs of different categories in different categories of websites. The heat map considers all visits during all the harvests aggregated. The **most intense relationship can be seen in the lower right corner, where private media and advertising intersect**. We can also notice that the presence of advertising-related TPs is quite high

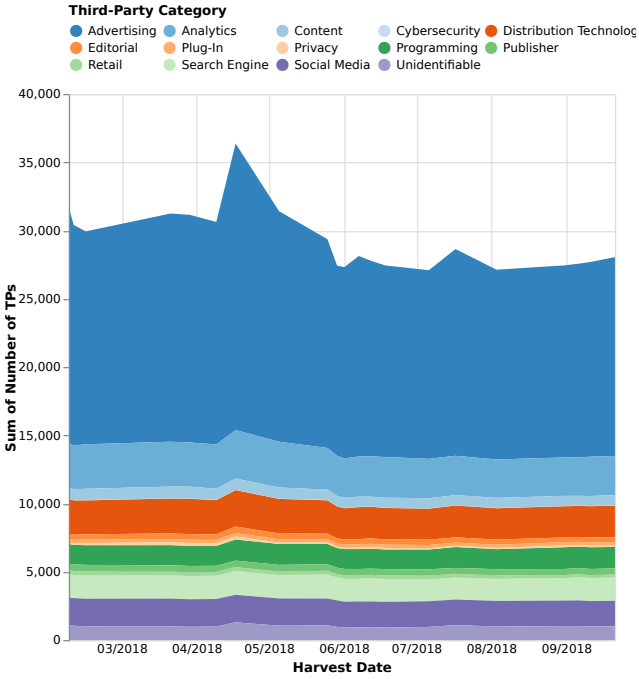


Figure 8: The total number of TPs distributed in categories.

in all the other types of sites, especially on public service media pages, on shopping/travelling, and entertainment sites. We can also notice that some TP categories are only found at some types of visited sites: The TP categories *Editorial* and *Publisher* are unique to the media websites. Other categories, such as *Cybersecurity* and *Privacy*, are low both in the number of unique TPs and in distribution and can thus not be expected to be prominently represented in the heat map. As a potentially more controversial finding, we see occurrences of both TP categories *Retail* and *Advertising* being present in the website categories *University*, *Government*, *Weather-Public*, *PostalServices*, and *PublicTransport*. That indicates that on the level of third-party ecosystems, **the borders between public and private institutions are to a certain degree dissolved**, with a possible user-tracking across the public-private border.

4.5 Interpretation

The different categories of websites have both *i)* a different behavior during the sample period and *ii)* a different pattern in the distribution of websites with many or few TPs. When we again look at Figure 4, we can see that the number of TPs for some categories of websites remains unchanged over the sampling period, while others have fluctuating amounts of TPs. Further detailed analyses on the level of individual pages and individual TPs may clarify the pattern. By looking at Figure 4, we can also see that the categories are showing a different span between the website in the category with the fewest TPs, and the website in same category with the most TPs. For websites in categories such as *Entertainment, NewsPrivate, NewsPublic*, or *ShoppingTravel*, this calls for a differentiated analysis. Possibly, for each of the categories websites with respectively few

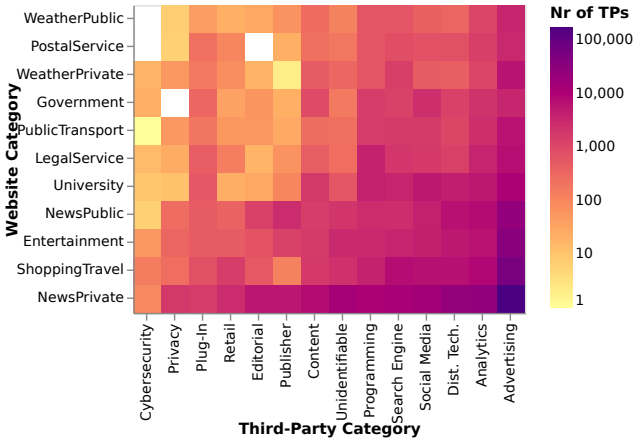


Figure 9: The intensity of the relation between categories of websites and categories of TPs on a logarithmic scale.

or many TPs may have more in common (also TPs) with websites in other categories. Again, this calls for an analysis of the appearance pattern for individual TPs across websites. An assumption, that can be supported by our findings, is that the fluctuations are mostly related to advertising and unidentifiable TPs, as shown in Figure 8.

Looking at the amount of TPs over time, we cannot see a decline for most categories of sites, and for others only a small decline. The assumption that GDPR would lead to a remarkable decline cannot be supported, as the fluctuations are too many for most of the categories of sites. Only for the categories *NewsPrivate*, *Entertainment*, and *ShoppingTravel* we could see a significant decline. However, to interpret this as an effect of GDPR may be problematic, since drawing strong conclusions on cause-effect relationships in such a complex environment with many impacting factors, such as online advertising and web page compilation, is difficult.

One hypothesis could be that advertisers prefer to reduce the amount of TPs to ease the obstacle of obtaining consent. Another hypothesis could be that the decline is caused by a technological change in the advertising industry. Indeed, the advertising industry is currently changing the methods for conducting the real time bidding for the sale of advertisements on web pages [15]. The old method conducts the bidding from the user's browser, producing - in the case of a difficult bidding process - a large number of HTTP requests and responses to real-time bidding platforms and their associated *Supply-side* and *Sell-side* platforms, among other third parties [4, 13]. The new server-side header bidding does not involve the user's browser, but takes place between the publisher website (the first party) and the servers of the bidding platforms. While the privacy exposure may be as big as before, the user will experience fewer HTTP requests from her browser [15]. Further research is however needed to assess the uptake of server-side header bidding by publishers and other websites displaying advertisements. Based on our current research it is thus problematic to associate the decline in advertising-related and unidentifiable TPs with the introduction of GDPR.

5 DISCUSSION

Our analysis shows that the number of third party servers a user will meet at different websites in many cases fluctuates from visit to visit. It is thus difficult to draw the conclusion that GDPR should have resulted in a user meeting fewer third parties when browsing a website. Furthermore, attributing the general decline we see to GDPR is problematic, as the implementation of GDPR coincides with another change in the ecosystem of third party services, namely the uptake of server-side header bidding for online advertisements [15] (see the previous section). Further analyses is however needed to shed light on these changes.

One can interpret the third party activities in different ways. The literature has mainly looked at these from a privacy perspective, focusing particularly on the tracking of users across websites. Through our categorization of third parties, based on text analysis of services offered by the third party company, we can however see that user tracking, segmentation, and user profiling have become standard products offered for sale to any advertiser. That does not make the privacy issue less pressing, but it shows that tracking has become industrialized.

Another way to interpret the third party activities is to see them as sign of an ever more integrated world wide web; a business ecology of services that co-produce the experiences to users, an interpretation aligned also with previous works [22]. That approach raises two discussions: one on *dependence* and one on *efficiency*.

The dependence discussion, initiated by Lindskow et al. [22], focuses on media companies' dependency on external collaboration in order to reach their audiences and monetize the contact. The outsourcing of production of the website content through a deep integration in the third party service ecology, diminishes the power and freedom of the media institutions. They operate on top of an infrastructure composed by a complex network of third party services, a network which has become the gateway to users.

The efficiency is a rather technical discussion on improving page loading times, initiated by the advertising industry. The answer has been to conduct the bidding for advertisements on the server-side, between the media organization's server and the bidding platforms with their affiliated network of services. The study of third-party interaction in the browser may however in the future not provide us much information of the exchange and analysis of our user data.

6 CONCLUSIONS AND FUTURE WORK

Our longitudinal large-scale study of the third-party server interactions at websites has shown that no clear effect of GDPR can be seen. Fluctuations of the number of TPs during the eight months harvesting period can be observed for most categories of websites, with some categories, such as private news, shopping/travel, and entertainment, presenting a visible decline. Other categories have a stable or even slightly growing number of unique third-party web services. The detailed study on the level of different countries adds nuances to the distinction between EU/EEA countries and other non-EU countries and the USA. Particularly, we see a difference between private websites from within the EU/EEA and from outside. Furthermore, our research shows large differences between privately owned websites and publicly owned websites, with the

former having many TPs embedded and the latter having low but growing amount of TPs.

As future work, we intend to analyze the relations between single third party servers and single websites, possibly employing machine learning techniques to uncover hidden patterns and relations. Furthermore, we will also continue harvesting the selected websites, potentially also extending the list. Finally, we will also focus on further analyzing the utilization of third party services in publicly owned websites.

REFERENCES

[1] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. 2014. The Web Never Forgets. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security - CCS '14*, Gail-Joon Ahn, Moti Yung, and Ninghui Li (Eds.). ACM Press, New York, New York, USA, 674–689. https://doi.org/10.1145/2660267.2660347

[2] Reuben Binns, Jun Zhao, Max Van Kleek, and Nigel Shadbolt. 2018. Measuring third party tracker power across web and mobile. (2018), 1–21. https://doi.org/10.1145/3176246 arXiv:1802.02507

[3] Justin Brookman, Phoebe Rouge, Aaron Alva, and Christina Yeung. 2017. Cross-Device Tracking: Measurement and Disclosures. *Proceedings on Privacy Enhancing Technologies* 2017, 2 (2017), 133–148. https://doi.org/10.1515/popets-2017-0020

[4] Oliver Busch. 2016. The Programmatic Advertising Principle. In *Programmatic Advertising: The Successful Transformation to Automated, Data-Driven Marketing in Real-Time*, Oliver Busch (Ed.). Springer, Cham, 3–15. https://doi.org/10.1007/978-3-319-25023-6_1

[5] Yuyu Chen. 2018. GDPR is coming, and data management platforms are in the crosshairs. https://digiday.com/marketing/gdpr-coming-data-management-platforms-crosshairs/

[6] Mark D. Corner, Brian N. Levine, Omar Ismail, and Angela Upreti. 2017. Advertising-based Measurement: A Platform of 7 Billion Mobile Devices. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking (MobiCom '17)*. ACM, New York, NY, USA, 435–447. https://doi.org/10.1145/3117811.3117844

[7] EBU Digital Strategy Group. 2002. *Media with a purpose - Public Service Broadcasting in the digital era. The Report of the Digital Strategy Group of the European Broadcasting Union.* Technical Report Version DSG 1.0. European Broadcasting Union, Geneva. http://www.ebu.ch/CMSimages/en/DSG_final_report_E_tcm6-5090.pdf

[8] EBU Media Intelligence Service. 2017. *Market Insights : Trust in Media.* Technical Report. European Broadcasting Union, Geneva.

[9] EBU Media Intelligence Service. 2018. *Market Insights: Trust in media 2018.* Technical Report. European Broadcasting Union, Geneva. 40 pages. https://www.ebu.ch/files/live/sites/ebu/files/Publications/MIS/login_only/market_insights/EBU-MIS-TrustinMedia2018.pdf

[10] Steven Englehardt and Arvind Narayanan. 2016. Online Tracking: A 1-million-site Measurement and Analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*. ACM, New York, NY, USA, 1388–1401. https://doi.org/10.1145/2976749.2978313

[11] Steven Englehardt and Arvind Narayanan. 2016. Online tracking: A 1-million-site measurement and analysis. In *Extended Version of the paper presented at ACM CCS 2016.* http://randomwalker.info/publications/OpenWPM_1_million_site_tracking_measurement.pdf

[12] Marjan Falahrastegar, Hamed Haddadi, Steve Uhlig, and Richard Mortier. 2014. Anatomy of the Third-Party Web Tracking Ecosystem. (sep 2014). arXiv:1409.1066 http://arxiv.org/abs/1409.1066

[13] Oliver Gertz and Deirdre McGlashan. 2016. Consumer-Centric Programmatic Advertising. In *Programmatic Advertising The Successful Transformation to Automated, Data-Driven Marketing in Real-Time*. 55–73. https://doi.org/10.1007/978-3-319-25023-6_5

[14] IAB - Interactive advertising bureau Europe. 2018. GDPR Implementation. https://www.iabeurope.eu/category/policy/gdpr-implementation/

[15] IAB - Interactive advertising bureau Europe. 2018. *Header Bidding and Auction Dynamics.* Technical Report August. interactive advertising bureau Europe. https://www.iab.it/wp-content/uploads/2018/09/IAB-Europe_Header-Bidding-and-Auction-Dynamics-White-Paper_August-2018-1-compressed.pdf

[16] Costas Iordanou, Georgios Smaragdakis, Ingmar Poese, and Nikolaos Laoutaris. 2018. Tracing Cross Border Web Tracking. In *Proceedings of ACM IMC 2018.* Boston, MA.

[17] Robin Kurzer. 2017. What does the GDPR mean to your third-party data processors? https://martechtoday.com/gdpr-mean-third-party-data-processors-208098

[18] Adam Lerner, Anna Kornfeld Simpson, Tadayoshi Kohno, and Franziska Roesner. 2016. Internet Jones and the Raiders of the Lost Trackers: An Archaeological Study of Web Tracking from 1996 to 2016. In *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Austin, TX. https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/lerner

[19] Tim Libert. [n. d.]. webXray. https://webxray.org/

[20] Timothy Libert. 2018. An Automated Approach to Auditing Disclosure of Third-Party Data Collection in Website Privacy Policies. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 207–216. https://doi.org/10.1145/3178876.3186087

[21] Timothy Libert, Lucas Graves, and Rasmus Kleis Nielsen. 2018. *Changes in Third-Party Content on European News Websites aft er GDPR.* Technical Report August. Reuters Institute, Oxford University. 1–7 pages. https://reutersinstitute.politics.ox.ac.uk/our-research/changes-third-party-content-european-news-websites-after-gdpr

[22] Kasper Lindskow. 2016. *Exploring Digital News Publishing Business Models - A Production Network Approach.* Ph.D. Dissertation. Copenhagen Business School. http://hdl.handle.net/10398/9284

[23] Johan Mazel, Richard Garnier, and Kensuke Fukuda. 2017. A comparison of web privacy protection techniques. (2017). https://doi.org/10.1016/j.devcel.2015.10.024.Role arXiv:1712.06850

[24] Nic Newman, Richard Fletcher, Antonis Kalogeropoulos, David A. L. Levy, and Rasmus Kleis Nielsen. 2017. *Reuters Institute Digital News Report 2017.* Technical Report. Reuters Institute for the Study of Journalism, Oxford. https://reutersinstitute.politics.ox.ac.uk/sites/default/files/DigitalNewsReport2017web_0.pdf

[25] Louise Reseke. 2018. JP/Politiken tager opgør med annonce-jungle og dropper 200 samarbejder. https://mediawatch.dk/secure/Medienyt/Web/article10631483.ece

[26] Patricio Robles. 2018. GDPR: What future for first, second and third-party data. https://econsultancy.com/gdpr-what-future-for-first-second-and-third-party-data/

[27] Elsa Rebeca Turcios Rodriguez. 2018. *Tracking Cookies in the European Union, an Empirical Analysis of the Current Situation.* Master. Delft University of Technology.

[28] Franziska Roesner, Tadayoshi Kohno, and David Wetherall. 2012. Detecting and defending against third-party tracking on the web. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation (NSDI'12)*. USENIX Association, Berkeley, California, 12–21.

[29] Franziska Roesner, Chris Rovillos, Alisha Saxena, and Tadayoshi Kohno. 2015. TrackingObserver: A Browser-Based Web Tracking Detection Platform. https://trackingobserver.cs.washington.edu

[30] Jannick Kirk Sørensen and Hilde Van den Bulck. 2018. Public service media online, advertising and the third-party user data business. *Convergence: The International Journal of Research into New Media Technologies* (aug 2018). https://doi.org/10.1177/1354856518790203

[31] Bharat Srinivasan, Athanasios Kountouras, Najmeh Miramirkhani, Monjur Alam, Nick Nikiforakis, Manos Antonakakis, and Mustaque Ahamad. 2018. Exposing Search and Advertisement Abuse Tactics and Infrastructure of Technical Support Scammers. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 319–328. https://doi.org/10.1145/3178876.3186098

[32] Oleksii Starov, Yuchen Zhou, Xiao Zhang, Najmeh Miramirkhani, and Nick Nikiforakis. 2018. Betrayed by Your Dashboard: Discovering Malicious Campaigns via Web Analytics. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 227–236. https://doi.org/10.1145/3178876.3186089

[33] Torben Stühmeier and Tobias Wenzel. 2012. Regulating Advertising in the Presence of Public Service Broadcasting. *Review of Network Economics* 11, 2 (jan 2012). https://doi.org/10.1515/1446-9022.1251

[34] Max Van Kleek, Ilaria Liccardi, Reuben Binns, Jun Zhao, Daniel J. Weitzner, and Nigel Shadbolt. 2017. Better the Devil You Know: Exposing the Data Sharing Practices of Smartphone Apps. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 5208–5220. https://doi.org/10.1145/3025453.3025556

[35] Tim Wambach and Katharina Bräunlich. 2017. The Evolution of Third-Party Web Tracking. In *Information Systems Security and Privacy. ICISSP 2016. Communications in Computer and Information Science*, Olivier Camp, Steven Furnell, and Paolo Mori (Eds.), Vol. 691. Springer, 130–147. https://doi.org/10.1007/978-3-319-54433-5_8

[36] Chris Ward. 2018. Will GDPR kill the third-party data market? https://www.mycustomer.com/marketing/data/will-gdpr-kill-the-third-party-data-market

[37] Michele Zanitti, Sokol Kosta, and Jannick Sørensen. 2018. A User-Centric Diversity by Design Recommender System for the Movie Application Domain. In *Companion Proceedings of the The Web Conference 2018 (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1381–1389. https://doi.org/10.1145/3184558.3191580

Multiple Treatment Effect Estimation using Deep Generative Model with Task Embedding

Shiv Kumar Saini
Adobe Research
Bangalore, Karnataka
shsaini@adobe.com

Sunny Dhamnani
Adobe Research
Bangalore, Karnataka
dhamnani@adobe.com

Aakash Srinivasan*
IIT Madras
Chennai, Tamil Nadu
s.aakash3431@gmail.com

Akil Arif Ibrahim*
IIT Roorkee
Roorkee, Uttarakhand
akilarif1997@gmail.com

Prithviraj Chavan*
IIT Kanpur
Kanpur, Uttar Pradesh
prithvi@cse.iitk.ac.in

ABSTRACT

Causal inference using observational data on multiple treatments is an important problem in a wide variety of fields. However, the existing literature tends to focus only on causal inference in case of binary or multinoulli treatments. These models are either incompatible with multiple treatments, or extending them to multiple treatments is computationally expensive. We use a previous formulation of causal inference using variational autoencoder (VAE) and propose a novel architecture to estimate the causal effect of any subset of the treatments. The higher order effects of multiple treatments are captured through a task embedding. The task embedding allows the model to scale to multiple treatments. The model is applied on real digital marketing dataset to evaluate the next best set of marketing actions. For evaluation, the model is compared against competitive baseline models on two semi-synthetic datasets created using the covariates from the real dataset. The performance is measured along four evaluation metrics considered in the causal inference literature and one proposed by us. The proposed evaluation metric measures the loss in the expected outcome when a particular model is used for decision making as compared to the ground truth. The proposed model outperforms the baselines along all five evaluation metrics. It outperforms the best baseline by over 30% along these evaluation metrics. The proposed approach is also shown to be robust when a subset of the confounders is not observed. The results on real data show the importance of the flexible modeling approach provided by the proposed model.

CCS CONCEPTS

• **Mathematics of computing** → **Causal networks**; *Bayesian computation*; *Variational methods*; • **Information systems** → **Online advertising**; • **Applied computing** → *Marketing*.

*Work done when the authors were affiliated with Adobe Research.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313744

KEYWORDS

Causal Inference; Individual Treatment Effect; Deep Generative Model; Variational autoencoder; Task Embedding; Counter-factual Analysis; What-if Analysis; Next Best Action; Multi-channel data

ACM Reference Format:
Shiv Kumar Saini, Sunny Dhamnani, Aakash Srinivasan, Akil Arif Ibrahim, and Prithviraj Chavan. 2019. Multiple Treatment Effect Estimation using Deep Generative Model with Task Embedding. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313744

1 INTRODUCTION

This paper provides a scalable approach for estimation of individual treatment effect using observational data in the setting where any subset of treatments from a set of treatments can be applied. The main motivation for this work is to provide a scalable approach for running counterfactual experiments on the effect of marketing actions on likelihood of conversion of a customer. The goal is to use the model for personalized online marketing actions. This problem is modeled as a causal treatment effect estimation problem where a subset of treatments can be applied. This setting is relevant for problems in a wide array of domains such as Economics [5], Medicine, Public Policy [14].

The prescribed approach for estimating causal effect is to conduct a randomized control trial. Unfortunately, running a randomized control trial is expensive and often not possible. Easy availability of observational data makes causal inference using observed data an attractive alternative to costly experiments. However, it is often the case that treatments in observational data were not assigned randomly. For example, a customer is targeted with an advertisement based on propensity of the customer to click the ad. Formally, the assignment of a treatment to a customer is not independent of the outcome. This can lead to misleading estimates of treatment effect [12]. In this particular example, the effect of the targeted advertisement will be overestimated when outcomes of "treated" and "untreated" customers are compared without any adjustment for the biased assignment. In addition to the biases due to assignment, there is likely a selection bias where an individual is self-selecting to be part of a treatment group [2, 16].

There is a large literature in various fields on causal inference using observational data. The Rubin Causal Model [29] and conditioning on observables [18], are two popular approaches to handle the bias due to assignment or selection on observables. There are recent papers that extend these models by using deep learning models (see [19, 31]). These approaches do not apply when the assignment or selection is based on unobserved variables. Selection on unobserved can be due to a user self-selecting treatment(s) on the basis of her wants and needs. These unobserved variable are called latent confounders. Under the assumption that there are observable proxies which can be used to recover these latent confounders, the causal effect is identifiable. Recent papers have used deep generative models to recover the causal effect in a setting where latent variables determine assignment [25, 34]. These models have shown promising result. Our approach follows this line of literature.

The main focus of this literature is the setting where the treatment is a binary [25] or a multinoulli [34] variable. The case where treatment can be any subset out of a set of treatments is less studied. The approach to deal with multiple treatments seems to be either to treat each subset as a separate treatment [24] or derive multiple treatment effect as a function of the treatment effects of the individual treatments in a subset. As an example for the latter, in a linear regression setting the effect of two treatments can be modeled as a sum of the individual effect of the two treatments plus an extra coefficient for the joint effect. Both of these approaches have drawbacks. Treating each subset as a separate treatment do not use the information across treatments. Besides, it is not a scalable approach if the model requires separate models for each treatment (see [19, 25, 31, 34] for such models). Another issue is that all subsets of treatments might not be observed. For example, in the real dataset used in this paper there are five treatments, which implies $2^5 = 32$ total subsets of the treatments. Only 26 out of 32 subsets are observed. Out of these 26 subsets, 8 are observed less than 10 times, 10 are observed between 10 to 100 times, and only 8 are observed more than 100 times. This makes it difficult for a large number of existing models to be applicable for the task at hand. The approach to impose a particular structure on the treatment effect of a subset has the obvious drawback that the functional form can be wrong.

Our approach overcomes both of these drawbacks. Our main contribution is that we use a task embedding [15] to scale the model presented in [25] for multiple treatments. We explicitly model the latent variables and the structure of the causal problem using a variational autoencoder [20]. The effect of multiple treatments is incorporated by using a task embedding to model the interdependence between the treatments. A task embedding allows a flexible representation of a task by multiplying a vector of zeros and ones, representing which treatments are applied, and a weight matrix which is learned. The resulting vector is passed on to the following layers of the network. The dimensions of the weight matrix are equal to the number of treatments times a constant which is a hyperparameter of the model. If two treatments have similar effect, then the columns corresponding to the two treatments will take similar values. If two treatment have similar values only for a subset of the observed covariates, this can also be represented by the task embedding. We call this model Task Embedding based Causal Effect Variational Autoencoder (TECE-VAE). The model is robust even when some of the subsets of treatments are not observed and uses information across observed subsets of treatments. The simple tweak of the architecture by incorporating a task-embedding provides a powerful tool to deal with multiple treatments.

The proposed approach, TECE-VAE, shows improved performance over a wide variety of competitive baselines. The model is compared against deep causal models as well as traditional models that have shown to work well under different settings. The baselines also include models that have done well in causal effect estimation competitions. The proposed approach is compared with the baselines on a semi-synthetic dataset with three treatments. The choice of the number of treatments in the semi-synthetic data is due to the difficulty in scaling some of the baselines to more than three treatments. Note that the proposed approach scales well to multiple treatment. Hence, the results are conservative from the perspective of the proposed model.

The result on semi-synthetic dataset shows that TECE-VAE improves upon the best baselines by over 30% on a variety of metric, including one proposed by us. Another set of experiments tests the robustness of the model when some of the covariates representing hidden confounders are not observed. The proposed approach continue to work well on the reduced set of covariates when 20%, 40%, 60% of the covariates are removed from the dataset. We also apply the model on a real world dataset to solve a real causal inference problem. The results indicate that assignment bias is a problem and explicitly modeling the latent variables is helpful.

Next, the literature related to this work is discussed.

2 RELATED WORK

In our setting, the assignment is based on observable as well as on latent variables. We make an assumption that there are proxies observed for the latent variables.

The closest paper to our work is [25]. This paper uses a Variational Autoencoder (VAE) [21] to model and estimate the joint distribution of the observed proxies, treatments, unobserved latent variables, and the outcome. Following Rubin's potential outcomes framework, this model uses a private path for each treatment. The assignment model decides which of these paths is taken. Due to a large number of potential outcomes where all combinations are not observed, this approach is not directly applicable in our setting.

There are other papers that have used deep models for causal inference. Johansson *et al.* [19] introduced Balancing Linear Regression (BLR) and Balancing Neural Network (BNN), which is used for counterfactual inference in the case of binary treatments. They formulate the problem of counterfactual inference as a domain adaptation problem. Shalit *et al.* [31] introduced Counterfactual Regression (CFR) and Treatment-Agnostic Representation Network (TARNet) for estimating individual treatment effect (ITE) from observational data in the case of binary treatments. The causal inference models in these papers use an architecture with private paths for each treatment. The reason given for choosing private paths is to preserve the effect of a treatment amidst the noise due to non-treatment covariates. This might happen because the number of non-treatment features is very large compared to the treatment features. Our model, TECE-VAE, does not use treatment specific private paths. Instead, we use a task embedding to represent each

combination of tasks. Even though the semi-synthetic data and the real world dataset have a large number of non-treatment feature, the estimated ITE in not noisy.

There are other recent papers that have used deep generative models for causal inference. Yoon *et al.* [34] introduced Generative Adversarial Nets for inference of Individualized Treatment Effects (GANITE), an approach using Generative Adversarial Network (GAN) [10] to estimate the individualized treatment effects in case of multiple treatments. We were not able to obtain the code for GANITE, hence excluded this model from the baselines.

In addition to the deep model, other models which are considered for comparison come from the set of models that have performed well in a causal inference competition [6]. These models are Random Forest (RF) [3], Gradient Boosting Machine (GBM) [8], and Bayesian Additive Regression Trees (BART) [4]. These are tree-based supervised learning approaches that work well in causal inference tasks [6]. The third one, BART, is a computationally intensive approach that took about 12 hours for one run on our semi-synthetic dataset. These approaches extend easily to multiple treatment. While the first two are not proposed as causal models, these can recover causal effect by conditioning on observable covariates.

In summary, the existing literature, with naive extensions for a subset of treatments, often introduces scalability issues since the number of possible subset of treatments increases exponentially with number of available treatments. Moreover these approaches might fail to account for the interdependence among the treatments. We address this gap in the literature by proposing a scalable model that addresses the problem of estimating individual treatment effect when any subset of the available treatments can be applied.

The next section describes the proposed approach in detail.

3 METHOD

The causal structure in Figure 1 depicts latent confounders ($\boldsymbol{z}$) that determine who gets what possible combination of treatments ($\boldsymbol{t}$). The latent confounders also influence the outcome (y). The proxy variables, denoted by $\boldsymbol{x}$, are noisy representations of latent confounders ($\boldsymbol{z}$) that are observed. In the context of online marketing, the casual structure can be described as follows: $\boldsymbol{x}$ represents observed online behavior of customers and $\boldsymbol{t}$ denotes the treatments which a marketer can apply (for example sending an e-mail, offering discount etc.) to improve the likelihood of conversion. In such a scenario, the unobserved intent or purpose $\boldsymbol{z}$ guiding such online behavior $\boldsymbol{x}$, not only influences the conversion rate y, but also marketer's choice of actions $\boldsymbol{t}$. Hence, $\boldsymbol{z}$ confounds both y and $\boldsymbol{t}$.

It has been established that the proxy variables ($\boldsymbol{x}$) should not be treated as usual confounders ($\boldsymbol{z}$) because it induces bias [9, 11, 13, 22, 27]. Since $\boldsymbol{z}$ is not observed, we estimate a representation of $\boldsymbol{z}$ and use it to control for confounding effects of the latent confounders to obtain an unbiased estimate for the causal effect of $\boldsymbol{t}$ on y.

We use variational autoencoder (VAE) for causal effect estimation. The formulation is influenced by Louizos *et al.* [25]. As noted earlier, we use task embedding to represent different subsets of treatments instead of using private paths. VAE allows recovery of the distribution of the latent confounders under the assumption that the proxies are observed for the latent confounders. Recent

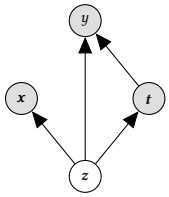


Figure 1: Causal graph where $\boldsymbol{x}$ represents observed proxies for confounders $\boldsymbol{z}$; $\boldsymbol{t}$ are treatments and y is outcome.

work on multiple treatments [33] show that weaker assumptions are sufficient to recover the causal effect of treatment when multiple treatments are applied. Under the assumption that there is no single cause confounder, the distribution of the latent confounders can be recovered from the assignment of the treatments in the observed data [33]. Note that we still need proxies for *all* latent confounders in case of *binary treatment*. Wang *et al.* [33] implies that the proxies for the latent confounders are not needed to identify the causal effect in case of *multiple treatments*. Inspired by this result, we also test the model when only a subset of the proxies are observed.

We can recover individual treatment effect under the causal model in Figure 1 if we can find the joint probability distribution $p(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{t}, y)$ [25, 26]. Using the causal structure, this joint probability distribution can be reduced to the product of the probability distributions $p(\boldsymbol{t}|\boldsymbol{z})$, $p(\boldsymbol{x}|\boldsymbol{z})$, $p(y|\boldsymbol{t}, \boldsymbol{z})$, and $p(\boldsymbol{z})$. These conditional probabilities are estimated using TECE-VAE.

First, let us introduce notations for describing the model. Let the total number of treatments be k. A subset of the treatments is denoted by a k-dimensional vector $\boldsymbol{t}$. Each element in the vector is a binary variable indicating whether a particular treatment is applied. For example, for $k = 5$, if the first and the third treatments are applied then $\boldsymbol{t} = [1, 0, 1, 0, 0]$. Let the dimension of the vector $\boldsymbol{x}$ be m. The proxy variables, $\boldsymbol{x}$, can be continuous, categorical, or binary. We denote these three types of variables by $\boldsymbol{x_c}$, $\boldsymbol{x_f}$ and $\boldsymbol{x_b}$, respectively. Similarly, the number of variables for each type is denoted by m_c, m_f and m_b, respectively. Finally, let $\boldsymbol{z}$ be a d length vector denoting unobserved latent confounders. The length of the latent confounders, d, is a hyperparameter. Experiments show that the results are not sensitive to the choice of d. This is a comforting result because the dimension of $\boldsymbol{z}$ is not known and sensitivity to the choice of d can make the application of the model to the real data less useful.

The proposed variation of VAE, like other autoencoder models, has a decoder and an encoder component. The causal VAE part is a specific architecture where the VAE mimics the dependencies shown in Figure 1. The design choices made in the architecture are either standard practices followed in VAE literature or are explicitly identified to be the contributions of this study. The details of the encoder and the decoder are presented next.

3.1 Encoder

In this part of the VAE, we recover the distribution of the latent counfounders $\boldsymbol{z}$. The posterior of $\boldsymbol{z}$ depends on $\boldsymbol{x}$, y and $\boldsymbol{t}$. Different parts of the network compute different probability distributions. Let g_i be the output of the layer as labeled in Figure 2. The treatment vector $\boldsymbol{t}$ is modeled as a vector of k Bernoulli random variables

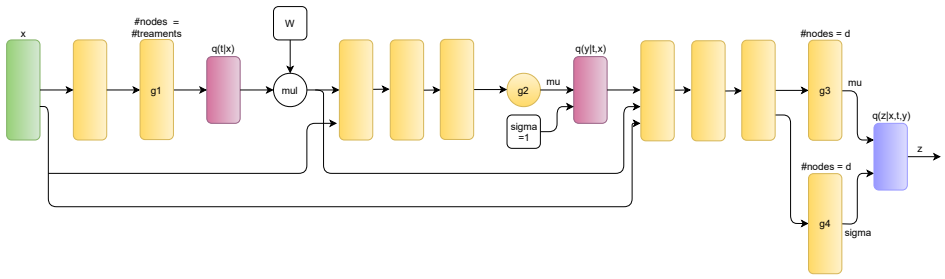


Figure 2: Encoder TECE-VAE, the encoder network takes input x to generate distribution for z. The green box refers to the input layer, input x is fed into yellow boxes that contain the network parameters to be trained. Purple boxes denote sampling step for various distributions. Note that weight matrix W is shown by an unshaded box.

which are independent given $\boldsymbol{x}$. Using the input $\boldsymbol{x}$, we obtain parameters for the distribution, $q(\boldsymbol{t}|\boldsymbol{x}) = \prod_{i=1}^{k} Bern(q_{\boldsymbol{t},i})$, where $q_{\boldsymbol{t},i}$ is the output of the layer labeled as g_1 in Figure 2.

The treatment vector, $\tilde{\boldsymbol{t}}$, is obtained by sampling from distribution $q(\boldsymbol{t}|\boldsymbol{x})$. Subsequently, we obtain a representation of the treatment vector $\tilde{\boldsymbol{t}}$ using the task embedding. The representation, denoted by $\boldsymbol{\tau}$, is obtained by projecting the k-dimensional treatment vector onto a k'-dimensional space ($\boldsymbol{\tau} \in \mathcal{R}^{k'}$) using an embedding matrix $\mathbf{W} \in \mathcal{R}^{k' \times k}$, $\boldsymbol{\tau} = \mathbf{W}\tilde{\boldsymbol{t}}$. The embedding weights, $\mathbf{W}$, are learned. Note that the embedding matrix $\mathbf{W}$ is shared between the encoder and decoder, in the next subsection it is shown how the decoder makes use of the embedding matrix. The dimension k' of the task embedding vector is a hyperparameter of the model. Such embeddings have been used in learning a curriculum of tasks in the domain of reinforcement learning [15]. By appropriately adjusting the dimensions of the embedding matrix $\mathbf{W}$, our model scales well for a large number of treatments. Furthermore, the use of task embedding accounts for the relationship between the treatments.

After finding $\boldsymbol{\tau}$, we concatenate it with $\boldsymbol{x}$ and feed it to the network to obtain parameters for $q(y|\boldsymbol{t},\boldsymbol{x})$ through g_2. For continuous y, we model $q(y|\boldsymbol{t},\boldsymbol{x})$ as $\mathcal{N}(\mu_{y_q}, 1)$. In case of binary outcomes, it is modeled as a Bernoulli distribution. We sample $\tilde{y}$ from the obtained distribution.

$$q(y|\boldsymbol{t},\boldsymbol{x}) = \mathcal{N}(\mu_{y_q}, 1), \quad \mu_{y_q} = g_2, \quad \boldsymbol{\tau} = \mathbf{W}\tilde{\boldsymbol{t}}$$

Finally, to compute posterior of $\boldsymbol{z}$ we estimate the mean and the variance using $\tilde{y}$, $\boldsymbol{x}$ and $\boldsymbol{\tau}$. The outputs i.e. $\mu_{\boldsymbol{z},i}$ and $\sigma_{\boldsymbol{z},i}$ are the outputs of the layers labeled g_3 and g_4 respectively.

$$q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{t},y) = \prod_{i=1}^{d} \mathcal{N}(\mu_{\boldsymbol{z},i}, \sigma_{\boldsymbol{z},i}), \quad \mu_{\boldsymbol{z},i}, \sigma_{\boldsymbol{z},i} = g_3, g_4 \tag{1}$$

The required joint probability distribution is the product of the probability distributions $p(\boldsymbol{x}|\boldsymbol{z})$, $p(\boldsymbol{t}|\boldsymbol{z})$, $p(y|\boldsymbol{t},\boldsymbol{z})$, and $p(\boldsymbol{z})$. Of these four terms, the distribution of latent confounders, $\boldsymbol{z}$, is output of the encoder part of the network. In the decoder network, we estimate the other three conditional probabilities.

3.2 Decoder

The latent confounder vector $\boldsymbol{z}$ is assumed to be vector of independent, univariate normal variables. The random variables that makeup the vectors $\boldsymbol{x_c}$, $\boldsymbol{x_f}$, and $\boldsymbol{x_b}$ are assumed to be independent of each other given $\boldsymbol{z}$ [25]. Every proxy variable in $\boldsymbol{x_c}$ is taken to be a univariate normal given $\boldsymbol{z}$. Similarly, every proxy variable in $\boldsymbol{x_b}$ is modeled as a Bernoulli(p) and every proxy variable in $\boldsymbol{x_f}$ is

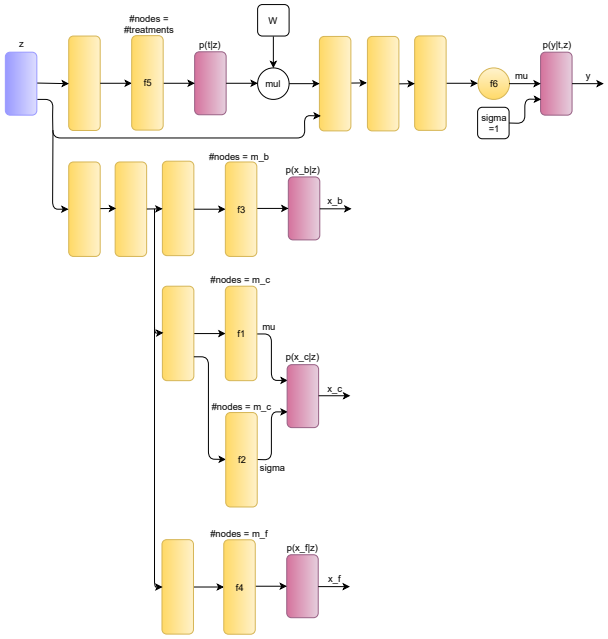


Figure 3: Decoder for TECE-VAE, the decoder network takes input z to reconstruct x, t and y. The input layer is depicted in blue, input z is fed into yellow boxes that contain the network parameters to be trained. Purple boxes denote sampling step for various distributions. Note that weight matrix W is shown by an unshaded box, this is the same matrix as used in the encoder network.

modeled as a categorical random variable that is independent of others conditional on $\boldsymbol{z}$.

Figure 3 shows the architecture of the decoder network. Different parts of decoder network predict different conditional distributions. Let f_i be the output of the layer as labeled in Figure 3. The decoder network works as follows: $\boldsymbol{z}$ is input to the network and the output of the layers labeled f_1 and f_2 are the parameters of the distribution $p(\boldsymbol{x_c}|\boldsymbol{z})$. Similarly, f_3 and f_4 output parameters for the binary and the categorical variables, respectively.

$$p(\boldsymbol{x_c}|\boldsymbol{z}) = \prod_{i=1}^{m_c} \mathcal{N}(\mu_{\boldsymbol{x},i}, \sigma_{\boldsymbol{x},i}), \qquad \mu_{\boldsymbol{x},i}, \sigma_{\boldsymbol{x},i} = f_1, f_2 \tag{2a}$$

$$p(\boldsymbol{x_b}|\boldsymbol{z}) = \prod_{i=1}^{m_b} Bern(p_{\boldsymbol{x},i}), \qquad p_{\boldsymbol{x},i} = f_3 \tag{2b}$$

$$p(\boldsymbol{x_f}|\boldsymbol{z}) = \prod_{i=1}^{m_f} Cat(\boldsymbol{q_{x,i}}), \qquad \boldsymbol{q_{x,i}} = f_4 \tag{2c}$$

Similar to encoder network the treatment vector $\boldsymbol{t}$ is represented by k Bernoulli random variables which are independent given $\boldsymbol{z}$. The conditional probability, $p(\boldsymbol{t}|\boldsymbol{z})$, is given by:

$$p(\boldsymbol{t}|\boldsymbol{z}) = \prod_{i=1}^{k} Bern(p_{\boldsymbol{t},i}), \qquad p_{\boldsymbol{t},i} = f_5 \tag{3}$$

Now, $\boldsymbol{x_c}$, $\boldsymbol{x_f}$ and $\boldsymbol{x_b}$ are sampled from corresponding distributions to obtain $\tilde{\boldsymbol{x}}$. Similarly $\tilde{\boldsymbol{t}}$ is sampled from $p(\boldsymbol{t}|\boldsymbol{z})$. For continuous y, we represent $p(y|\boldsymbol{t},\boldsymbol{z})$ as $\mathcal{N}(\mu_{y_p}, 1)$ and in case of binary outcomes we instead approximate with a Bernoulli distribution. In order to obtain $p(y|\boldsymbol{x},\boldsymbol{t},\boldsymbol{z})$ we transform the sampled $\tilde{\boldsymbol{t}}$ to $\boldsymbol{\tau}$ using the same embedding matrix $\mathbf{W}$. Next, $\boldsymbol{x}$, $\boldsymbol{z}$ and $\boldsymbol{\tau}$ are concatenated into a single vector which is then used to obtain the parameters for $p(y|\boldsymbol{t},\boldsymbol{z})$ as output of the layer f_6.

$$p(y|\boldsymbol{t},\boldsymbol{z}) = \mathcal{N}(\mu_{y_p}, 1), \quad \mu_{y_p} = f_6 \tag{4}$$

With equations 1, 2, 3 and 4, we can compute the joint probability distribution. We use variational inference to train the encoder-decoder. Note that during the training, we sample from $q(\boldsymbol{t}|\boldsymbol{x})$ and $p(\boldsymbol{t}|\boldsymbol{z})$. However during testing, if we want to find the effect of a particular combination of treatment, $\boldsymbol{t'}$, then we do not sample from these distributions. We instead directly pass $\boldsymbol{t'}$, which will be transformed into a state embedding. Also, observe that the embedding weights, $\mathbf{W}$, are the same for both the encoder and decoder. We have used variational inference algorithm available in the Python library Edward [32] for the training. The loss function is same as equation 10 in [25]. L2 regularization is added to the loss function, we then minimize this loss to estimate the model. The next section describes the datasets used for experimentation and associated proxy variables $\boldsymbol{x}$, outcome y, associated treatment vector $\boldsymbol{t}$ and confounders $\boldsymbol{z}$ for these datasets.

4 DATASETS

The proposed approach has been applied to three datasets. The first one is a user-conversion dataset for a product company that sells software subscriptions. This dataset contains outcomes for the marketing actions observed in the data. However, the evaluation of a causal model requires outcome for all potential marketing actions - observed or unobserved - for each customer. To evaluate the model, two semi-synthetic datasets are created using the covariates of the user-conversion dataset. The three datasets are described next.

User-Conversion Dataset: This dataset is constructed from the event level logs of the interaction events between a user and the organization's website as well as marketing actions. These interaction events come from four different data sources - web-analytics data, display ad impression data, email interaction data, and product usage data. We merge these data sources. Web-analytics data is stored in the form of a clickstream that records the online activities of an user on the company's website. Roughly, there is one row of data for each URL visited on the company's website. These visits include pages with information on the product features, product help, download trial versions or checkout. Each row contains information about the user's device, geography, source of visit, URL, time-stamp, product purchased etc. The display ad data provides information on ad impression on company's own website as well as other websites. This dataset stores which ads were shown to the user and whether the user clicked on the ad and many other ad-related properties. The email interaction dataset consists of email related information that was sent by the organization to the user. This dataset includes information such as whether a user opened the email, clicked on a link in the email, unsubscribed to emails from the company, description of the email, etc. Finally, product usage data contains information on a user's interactions with company's applications. Each row of the data stores information on the events such as application launch, application download etc. Each dataset described above uses the same identifier for the user, this identifier is used for merging the datasets.

The problem of a marketer working for the company is to decide the best subset of the treatments that can be applied to each user over the next three weeks. The set of treatments consists of five marketing actions which a marketer can potentially apply to each customer with varying degree of control. These are: 1) show paid search ads, 2) start a social media campaign, 3) send emails, 4) show ads on company's own website, and 5) show ads on other websites. One of the approaches to find the best set of actions is to estimate the `ITE` for each subset of the treatments. The action corresponding to the highest value of the `ITE` will be the optimal action for the user. The user-conversion information comes from the purchase events stored in web-analytics data. A user is said to be converted if he/she decides to makes a purchase in form of subscription to the company's product in the considered time-period.

In a supervised learning problem, the empirical design is such that the covariates are observed as of a time point. The outcome of interest or the target variable for the supervised model is observed for a period of time after the time point at which the covariates are observed. In our setting, where a marketer wants to ask counterfactual questions, with the goal of taking an action, the period of covariate observation is followed by a period during which applied treatments as well as the outcome are observed. The covariate observation period is three weeks in the user-conversion data. The applied marketing actions and the outcome event during the following three weeks are observed. Note that the marketing actions during the first three weeks will be part of the observed covariates, $\boldsymbol{x}$. The marketing actions applied after the purchase during the latter three week periods are ignored.

Once we have the stitched data and different periods for observing the user information, treatment applied $\boldsymbol{t}$ and conversion outcome y in place, we process the stitched data to create $\boldsymbol{x}$, $\boldsymbol{t}$ and y. The features $\boldsymbol{x}$ include Recency-Frequency-Monetary variables [7] as well as few columns are included as is. The recency features mainly accounted for how recently had the user interacted with the organization. The examples include how recently had the customer used a mobile application, interacted with an online-advertisement, opened a marketing email etc. The frequency features account for how often do these interactions occur. For instance, how frequently does a user launch an application, click on online-ads, check emails etc.

The set of five marketing actions are the treatments, $\boldsymbol{t}$. The treatments are assigned by the marketer to a user or self-selected by a user. For instance a marketer can send promotional emails to attract the customers for subscription or a customer can decide to search where a marketer can show a search ad. The outcome

variable y denotes if the user made a purchase. The purchase event in the data is subscription of the product.

In summary each row of the data contains user features $\boldsymbol{x}$, the treatment they were subjected to $\boldsymbol{t}$ and the outcome y if they subscribed to company's products/services. The dataset comprises of close to 1.2 million users. Due to the low percentage of the subscribers in the data, the negative class was down-sampled to create a balanced distribution of the subscribers and non-subscribers for training the models. The total number of features are 144. Out of these, 116 are continuous, 16 are Bernoulli, and 12 are categorical.

Semi-Synthetic Data: The original user-conversion dataset, as described above, contains the outcome corresponding to the observed subset of the treatments. However, for evaluating the models the outcome for every subset of the treatments is needed. This requires a dataset which contains outcome y for every subset of the treatments for each user. While the real-world dataset has five treatments, the synthetic data is limited to only three treatments. The number of treatments are limited to only three because we found it extremely hard to train many of the baseline models for five treatments.

Some of the baselines such as `BNN` [19], `BLR` [19], `CFR WASS` [31], and `TARNet` [31] train separate models for each subset of the treatments. In similar vein, `CEVAE` [25] employs separate paths for each subset of treatments. This implies that the number of parameters to be trained increases exponentially with the number of treatment. The problem is exacerbated by the fact that the number of observations in the data for each subset of the treatments is highly skewed. In the original user-conversion dataset, not all treatment subsets occur with the same frequency, some are frequently occurring treatments while some show up very rarely. This unbalanced distribution in the original dataset is shown in Figure 4. It shows a power distribution with no observation for 6 subset and a very few observations for another 10 or so subsets of marketing actions. This implies that for the treatment subset with fewer observations, the models will not be trained well, and the results will be poor.

Due to difficulty in estimating some of these baselines when the number of treatments are high, we restrict the total number of treatments to three. A dataset with fewer treatments is a conservative approach because it favors the baselines which are meant for a binary treatment.

Two semi-synthetic datasets are created to test how the different models perform in a setting that mimics the power law distribution of the observed subsets of the treatment vs when each subset appears uniformly. The frequency of the treatments in the uniform scenario is not completely equal but the differences are not stark. With these two settings, we not only test for the setting observed in a real world dataset like user-conversion data but also for the scenarios where the distribution for treatments is relatively uniform.

Three treatments imply a total of eight possible subsets of treatments. For evaluation of the causal effect, we need the `ITE` for each subset of the treatments for each individual user. Note that, to test the models in our setting (i.e. causal structure in Figure 1), we need observed proxies $\boldsymbol{x}$, latent confounders $\boldsymbol{z}$, observed treatments $\boldsymbol{t}$ assigned based on $\boldsymbol{z}$, and observed outcome y (for every treatment subset) determined by $\boldsymbol{z}$ and $\boldsymbol{t}$.

For generating the semi-synthetic datasets, we take the observed features, $\boldsymbol{x}$, from the user-conversion data. These are the observed proxies for the two semi-synthetic datasets. The semi-synthetic datasets are created using the following three steps:

(1) *Latent Confounder Creation:* Create latent counfouders $\boldsymbol{z}$ from the observed proxies $\boldsymbol{x}$.
(2) *Treatment Assignment Function:* Create the observed treatments as a function of the latent confounders for each user.
(3) *Outcome Function:* Create the observed outcome for each user and treatment subset as a function of the latent confounders and each treatment subset. Only the outcome corresponding to the observed treatment in step 2 is used for estimation. Other outcomes act as the ground truth for evaluation of the model.

Latent Confounder Creation: The observed proxies are then fed to an autoencoder [30] to model the hidden confounders. The middle layer of the autoencoder is used as the latent confounders. The size of the middle layer of the autoencoder is chosen to be 20. The latent values for each observation are used to generate the treatment to be assigned to each individual and the corresponding effect for each treatment. Note that the data generation approach used follows the causal structure in Figure 1, however, the data generating process is very different from the proposed model. Even the chosen dimensions of $\boldsymbol{z}$ for estimation and data generation are different. Next, the data generating process for obtaining the observed treatments and outcome is described in detail.

Treatment Assignment Function: As explained above, an autoencoder is used to obtain the latent confounders for each user. The Let Z be a matrix with n rows and d columns that contains the values of the latent confounders for each observation. Here n is the total number of observations and d is the total number of latent confounder variables. We set d to be 20 for the synthetic dataset. Let $\boldsymbol{t}_i$ be a row vector of length equal to the number of treatments. It specifies which subset of treatments was applied to i^{th} individual. Let k be the number of treatments. As discussed above, it is set to three. The outcome, $\boldsymbol{y}_i$, is a row vector that contains the `ITE` for each subset of the treatments for the i^{th} individual. The size of this vector is equal to the total number of possible treatment subsets, which is $2^3 = 8$ in this case.

We take a matrix θ, with d rows and the number of columns equal to the number of treatments. Each row $\theta_{\mathbf{i}}$ of θ is sampled from a standard multivariate normal distribution (mean is zero and covariance matrix is an identity matrix), i.e. $\theta_{\mathbf{i}} \sim \mathcal{N}(\mathbf{0}_{kX1}, \mathcal{I}_{kXk})$ where $i = 1, \ldots, d$.

We define a matrix μ, which is the difference between the matrix product of Z and θ and its mean taken across all the observations for balanced dataset. We subtracted each row of $Z \times \theta$ by the column-wise mean of $Z \times \theta$, which is a vector is size k. Let $mean_{all\ observations}(Z \times \theta)$ be a $n \times k$ matrix, with n duplicated rows containing column wise mean. For generating data with power law, -3 is added to all the elements of μ.

$$\mu = Z \times \theta - mean_{all\ observations}(Z \times \theta)$$

We took a covariance matrix Σ with k rows and k columns. The values that we chose are as follows:

$$\Sigma = \begin{bmatrix} 1 & 0.6 & 0.4 \\ 0.6 & 1 & -0.3 \\ 0.4 & -0.3 & 1 \end{bmatrix}$$

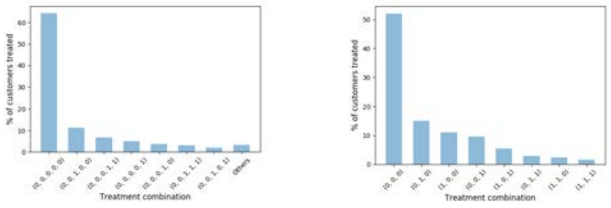


Figure 4: Distribution of treatments for user-conversion data (left) and unbalanced semi-synthetic data (right). Note that user-conversion dataset has 32 possible treatment subsets, we show top-7 in the graph while remaining 25 are aggregated to form *others* category. Unbalanced semi-synthetic dataset has 8 possible treatment subsets.

We define a row vector ϵ_i sampled from a multivariate normal distribution with mean μ_i (which is the i^{th} row of the matrix μ) and covariance matrix Σ.

An element of $\boldsymbol{t'(i)}$ is equal to 1 if the corresponding element in ϵ_i is greater than 0, otherwise 0. Hence $\boldsymbol{t'(i)}$ is a k-dimensional vector representing the treatments assigned to the i^{th} individual.

Outcome Function: We take a column vector $\boldsymbol{\beta}$ with length (d), where d denotes number of the latent confounder variables. Each element of $\boldsymbol{\beta}$ is sampled from a uniform distribution on the real interval [-10,10]. Let Z_i denote a column vector containing the i^{th} row of Z. We take a vector $\boldsymbol{\gamma}$ of size k. Each element of $\boldsymbol{\gamma}$ is sampled from a uniform distribution on the real interval [0,4]. Let y_{ij} denote the outcome of the i^{th} individual on applying j^{th} treatment vector $\boldsymbol{t}_j$.

The simulated outcome variable $y_{i,j}$ is given by the following equation:

$$y_{i,j} = {Z_i}^T \times \boldsymbol{\beta} + {\boldsymbol{t}_j}^T \times \boldsymbol{\gamma} + \xi_{i,j},$$

where ξ_{ij} is sampled from a normal distribution with mean as zero and variance as 10% of the absolute value of the sum of ${Z_i}^T \times \boldsymbol{\beta}$ and ${\boldsymbol{t}_j}^T \times \boldsymbol{\gamma}$. Hence for every individual i, the proxies($\boldsymbol{x}_i$), the treatment vector $\boldsymbol{t'(i)}$ and corresponding outcome $\boldsymbol{y}_i(\boldsymbol{t'(i)})$ forms the part of the training data. Figure 4 on the right shows the obtained treatment distribution for unbalanced semi-synthetic dataset.

The datasets described in this section are used to compare the proposed model with the baselines. In the next section we describe the details of the baselines and evaluation metrics.

5 EXPERIMENTS

In this section we describe experiments that compare the proposed approach to the existing treatment effect estimation techniques. Some of the baselines considered for the experiments extend naturally to multiple treatments, while others require some modification in the code. This section describes the changes made, the choice of hyperparameters and the evaluation metrics used for the experimentation.

5.1 Baselines

The baselines that naturally extend to multiple treatments are: k-nearest neighbor (k-NN), Random Forest (RF) [3], Gradient Boosting Machine (GBM) [8], and Bayesian Additive Regression Trees (BART) [4]. To implement k-NN, RF, and GBM, the python library scikit-learn [28] was used. To implement BART [4], the R package BayesTree[1] was used.

Other baseline models were created for binary treatment. These models are: Balancing Linear Regression (BLR) [19], Balancing Neural Network (BNN) [19], Counterfactual Regression with Wasserstein distance (CFR WASS) [31], Treatment-Agnostic Representation Network (TARNet) [31], and Causal Effect VAE (CEVAE) [25]. The available implementations for these models are not directly applicable to the multiple treatments setting. All of these baseline models except the last one were extended to multiple treatments setting, as done in [34]. All possible subsets of treatment can be taken as separate treatments and encoded as one-hot vectors. There are 2^k distinct treatment subsets so the size of one-hot vector is also 2^k. The models addressing the binary treatment-effect problem can be extended to multiple treatments by considering the problem as 2^k binary treatment-effect problems. And so the number of models required will be 2^k. The models that are extended to multiple treatment are BNN [19], BLR [19], CFR WASS [31], and TARNet [31]. We extended CEVAE [25] to handle multiple treatments. The number of distinct paths required is equal to the total number of treatment subsets 2^k, one path for each treatment subset.

For BNN [19], BLR [19], CFR WASS [31], and TARNet [31], the publicly available code[2] was used. For CEVAE [25], the publicly available code from github[3] was used. For the implementation of our model, Tensorflow [1] and Edward [32] were used. We used 63-27-10 % as train-validation-test split. We used the architecture as shown in figure 2 and 3. Unless specified we use 200 neurons per layer with *tanh* activation function. The size of $\boldsymbol{z}$ for CEVAE and TECE-VAE is 25. This was chosen based on experiments that showed that a large range of values for the dimension for $\boldsymbol{z}$ gives similar results. Figure 5 plots various evaluation metrics for different choices of the dimension of $\boldsymbol{z}$. The evaluation metrics do not change as the dimension of $\boldsymbol{z}$ is varied from 10 to 40.

These baselines as well as the proposed approach are compared using five causal inference evaluation metrics described next.

5.2 Evaluation Metrics

In the existing literature on binary treatments, the evaluation metrics used were RMSE of the estimated Individual Treatment Effect (ITE) [19], and absolute error in estimated Average Treatment Effect (ATE) [19, 25, 31]. These papers also report Precision in Estimation of Heterogeneous Effect (PEHE) [19, 25, 31] following [17]. We have used the evaluation metrics used in past as well as a few more that capture different aspects of a causal model used for choosing best subset of actions for each user. The evaluation metrics are described next.

The first two metrics - *RMSE* and *Absolute Error* - measure the difference between the ground truth and the estimated individual treatment effect (ITE) of the treatments. These metrics are used to evaluate the causal models in the multiple treatment effect literature [34]. In case of semi-synthetic data, the outcomes for all subsets

[1]The R package can be found at https://cran.r-project.org/package=BayesTree.
[2]Code available at https://github.com/clinicalml/cfrnet.
[3]Code available at https://github.com/AMLab-Amsterdam/CEVAE.

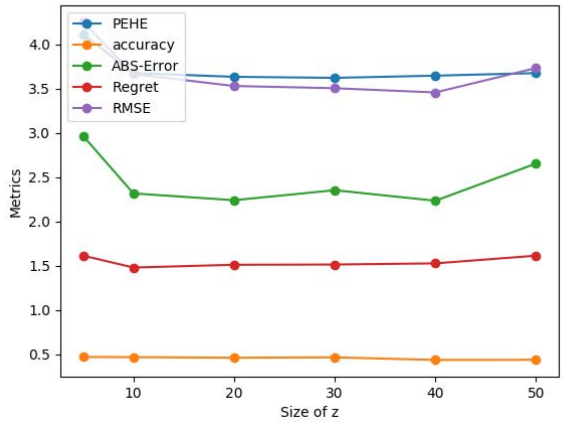


Figure 5: Values of various evaluation metrics for different dimensions of z, averaged over first three replications for unbalanced semi-synthetic data.

of the treatments are available in the data, and hence we calculate RMSE of the estimated outcomes for all subsets of the treatments. RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{|T|}\sum_{j\in T}(y_{i,j} - \hat{y}_{i,j})^2}$$

where, T is the set of all possible combinations of treatments, n is the number of observations, $y_{i,j}$ is the true outcome when j^{th} subset in T is applied on i^{th} observation, and $\hat{y}_{i,j}$ is the predicted outcome when j^{th} subset in T is applied on i^{th} observation. We also calculate absolute error in the same way as RMSE.

$$Absolute\ Error = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|T|}\sum_{j\in T}|y_{i,j} - \hat{y}_{i,j}|$$

The third metric, *Average PEHE Error*, measures the average of the RMSE between the actual and estimated difference between ITE for each treatment with no treatment ITE. This metric is used in binary treatment literature. We have extended it to multiple treatments setting. The subset with no treatments being applied is chosen as control and PEHE is calculated for each subset of all possible combinations of treatments with respect to control. Then, the error in PEHE is calculated for each treatment subset.

$$Error\ in\ PEHE_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}((y_{i,j} - y_{i,0}) - (\hat{y}_{i,j} - \hat{y}_{i,0}))^2}, j \in (T - T0)$$

where, $T0$ is the subset with no treatments being applied (control) Average PEHE error is the mean of error in PEHE for all the subsets of all possible combinations of treatments.

$$Average\ PEHE\ Error = \frac{1}{|T|}\sum_{j\in(T-T0)} Error\ in\ PEHE_j$$

These three metrics measure the accuracy of the estimated causal effect with respect to the actual values. However, these do not capture the loss suffered due to the estimation error. To measure the loss with respect to the actual ITE, we propose a metric which is similar to the regret metric used in the Bandit literature [23]. Unsurprisingly, we call this metric *Regret*. It measures the loss due to using a particular causal inference model with respect to the ground truth. This is something that a marketer targeting customer for improving bottom line will deeply care about.

$$Regret = \frac{1}{n}\sum_{i=1}^{n}(y_{i,t_i} - y_{i,\hat{t}_i})$$

where, t_i is the true best action for i^{th} observation, and $\hat{t}_i$ is the predicted best action for i^{th} observation. In addition to these four metrics, we also compare micro averaged F-score which compares the best action implied by an estimated model with the ground truth best action for each customer using the usual supervised learning metric - the F-Score.

Next section presents the results and discussions for these experiments.

Table 1: Out-of-sample metrics for the various models in the case of Unbalanced data. Bold indicates the model with the best performance for each metric.

Model Name	*RMSE*	*Absolute Error*	*Ave PEHE Error*	*Regret*	*Micro − Ave F − score*
RF	9.12 ± 0.84	5.35 ± 0.49	8.23 ± 0.94	3.40 ± 0.23	0.19 ± 0.03
k-NN	8.40 ± 0.49	5.09 ± 0.27	5.05 ± 0.30	5.71 ± 0.43	0.06 ± 0.01
GBM	6.71 ± 0.57	4.12 ± 0.35	6.29 ± 0.61	2.21 ± 0.18	0.40 ± 0.06
BLR*	15.74 ± 0.82	10.58 ± 0.65	5.41 ± 0.35	3.96 ± 0.46	0.11 ± 0.04
BNN-2-2*	15.43 ± 0.77	10.58 ± 0.59	5.51 ± 0.30	2.82 ± 0.38	0.22 ± 0.07
TARNet*	4.86 ± 0.22	3.38 ± 0.21	5.27 ± 0.32	2.62 ± 0.25	0.31 ± 0.05
CFR WASS*	8.10 ± 0.13	5.22 ± 0.11	6.15 ± 0.15	4.89 ± 0.14	0.08 ± 0.02
CEVAE*	22.79 ± 2.39	16.24 ± 1.78	16.49 ± 2.32	4.13 ± 0.39	0.11 ± 0.03
TECE-VAE	**4.01 ± 0.19**	**2.44 ± 0.12**	**3.95 ± 0.20**	**1.60 ± 0.18**	**0.51 ± 0.06**

* Original code modified for multiple treatments.

Table 1 compares the proposed approach with the baselines. This table present result on the semi-synthetic dataset that captures the skewness in the number of observation for each subset of treatments. There are 10 datasets created by changing the seed for the random draws. The evaluation metrics for each model are calculated on each of these 10 datasets. Table 1 presents mean and standard deviation of the evaluation metrics over the 10 runs. Note that we did not include the results for BART model because it takes about 12 hours for each run on the semi-synthetic data. The results for BART are included in Table 2 where the evaluation metrics are based on only one replication of the semi-synthetic dataset.

As shown in Table 1, the proposed approach TECE-VAE outperforms the baselines along each metric for the test data. The results for the training data are similar. Hence, the comparisons for the training data are not shown. The results are available on request. In the test data, TECE-VAE outperforms the closest baseline in terms of RMSE and Absolute Error, TARnet, by 21% and 39% respectively. This is a large improvement over the baselines. The implication is that, if one is interested in the *ITE* for each treatment, TECE-VAE provides a significant improvement over the baselines. In terms of *PEHE* and *Regret*, TECE-VAE outperforms the closest baselines by 33% and 38% respectively. As noted earlier, *Regret* measures the loss due to applying the optimal subset of treatment for each customer using a particular causal inference model instead of the

6 RESULTS AND OBSERVATIONS

6 RESULTS AND OBSERVATIONS

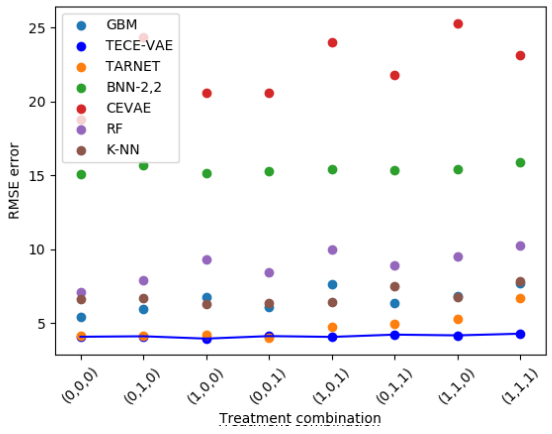


Figure 6: RMSE for each treatment subset averaged over 10 replications of unbalanced semi synthetic data. Treatments are sorted in descending order by the number of observations in the data.

ground truth optimal treatments. This result shows that by using TECE-VAE the outcome, as measured by *Regret*, improves by at least 38%. The last metric is Micro F-Score which measures accuracy of best ITE given by a causal model with respect to the ground truth optimal ITE. TECE-VAE outperforms the baselines along this metric as well.

Since, the number of observations for each subset of the treatments is different, a natural question is how the performance of the model varies by the number of times a particular subset of the treatments is observed. Figure 6 plots the *RMSE* metric for each treatment subset. The x-axis in the figure is sorted in decreasing order by the number of observation in the data. For example, the treatment subset (0,0,0) corresponds to no treatment and it is observed in the data for 39891 out of 69431 users; whereas the treatment subset (1,1,1) is observed only 9 times. As expected, the difference in the performance of TECE-VAE and other models increases as the number of observation for a treatment subset decrease. This result reinforces the assertion that the baseline models are not equipped to handle multiple treatments. Another observation from Figure 6 is that the performance of TECE-VAE as compared to TARnet starts to diverge around treatment subset (1,0,1). This subset of the treatments is observed 7244 times in the data, which is a large number. This implies that a large number of observations is needed to get comparable performance between the best baseline and TECE-VAE. As noted earlier, the main reason the model performs well is due to representation of treatments using a task embedding.

One question remains is how does TECE-VAE performs in a dataset where the distribution of the observed treatments is not skewed. To answer this question, we compared the models on a semi-synthetic dataset with uniform distribution of treatment subsets. Note that this dataset should be more favorable to the baseline techniques because many of these models estimate separate parameters for each treatment. The analysis on this dataset, presented in Table 2, shows that the proposed approach is among the top two models along all metrics. At the same time, the large difference between TECE-VAE and the baselines disappears. One interesting observation is that GBM, which is a supervised learning model and doesn't explicitly model the latent variables, works well when the observed treatment subsets are balanced. There are three important observations. First, if the number of treatments are balanced and the variance of the estimate is not needed, one can use GBM and expect reasonable results. Second, since GBM is available out of the box and easy to train, it should be an important baseline for any causal effect estimation model. Third, the comparison between TECE-VAE and CEVAE, where former uses a task embedding and latter uses private paths, reveal the benefits of representing treatments using a task embedding. Note that the number of treatments as well as number of observations per treatment subset are chosen to favor the baselines such as CEVAE. Even in this setting, TECE-VAE outperforms CEVAE along all evaluation metrics.

The next set of experiments are conducted to test the robustness of the proposed approach. The experiments so far assume that the complete set of proxies for the latent confounders is observed. However, in practise proxies for all latent confounders might not be observed. This call for testing the robustness of the proposed approach when a fraction of proxies are not observed. For this set of experiments, the semi-synthetic data generation process as well as the train-validation-test split is same as above. During model estimation, a fraction of proxies, $\boldsymbol{x}$, are not used for estimation. The comparisons are done for following four scenarios - all proxies observed, 20%, 40%, 60% proxies not observed. The proposed approach is compared against two baselines - TARnet and GBM. These two baseline models have been the second best models along two metrics - *RMSE*, which measures the accuracy of the estimate of ITE, and *Regret*, which measures the loss due to using an estimated decision rule as compared to the ground truth decision rule. Figure 7 contains the results of these experiments. Each point in Figure 7 is an average of 10 runs where a different, randomly selected, subset of the observed proxies is removed. The repetitions ensure that the evaluation metrics do not suffer from the sampling bias. The comparison of the *RMSE* of the ITE for the three models shows similar trend with TECE-VAE being the best model. The comparison along the *Regret* metric shows interesting results. TECE-VAE performance stays similar to 0% hidden proxies

Table 2: Out-of-sample metrics for the various models in the case of Balanced data. Bold indicates the model with the best performance for each metric.

Model Name	*RMSE*	*Absolute Error*	*Average PEHE Error*	*Regret*	*Micro − Averaged F − score*
k-NN	5.069	2.791	3.608	2.899	0.174
RF	4.516	2.430	3.449	2.147	0.261
GBM	3.781	2.072	3.434	1.617	0.440
BART	3.531	1.905	3.397	**1.533**	**0.450**
BLR*	15.305	11.207	3.440	1.985	0.193
BNN-2-2*	**3.142**	**1.848**	3.525	1.805	0.338
CFR WASS*	3.228	1.973	3.509	2.306	0.173
TARNet*	3.217	1.964	3.508	2.209	0.295
CEVAE*	4.598	2.429	4.337	1.938	0.226
TECE-VAE	**3.190**	**1.886**	**3.189**	1.656	0.298

* Original code modified for multiple treatments.

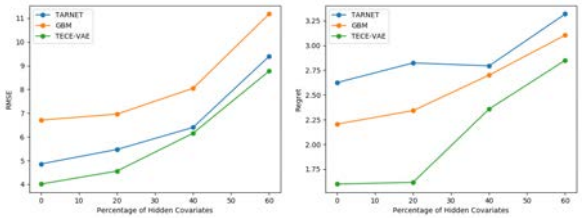


Figure 7: RMSE and Regret by 0%, 20%, 40%, and 60% covariates hidden, averaged over 10 replications for unbalanced semi-synthetic data.

results. TECE-VAE performance stays similar to 0% hidden proxies when 20% proxies are removed. The other two models show deterioration in performance. These results show that the proposed model is robust when a subset of proxies are not observed. Even the baselines are robust, albeit to a lesser degree. These results are in line with work recent which shows that in the case of multiple treatment the causal effect can be identified even when the latent confounders or proxies are not observed [33]. The intuition is that the assignment of multiple treatment provides information about the underlying latent confounders. A flexible model is able to utilize the observed assignment to recover the otherwise unobservable confounders and control for them to recover the causal effect. A deep generative model such as VAE provides a probabilistic factorization of the observed space in the form on the distribution of z. Wang *et al.* [33] show that such a representation identifies the unobserved confounders. The causal effect of multiple treatments is identified by controlling for this estimated representation of the input space.

User-Conversion Data: TECE-VAE is applied to the user- conversion dataset described in Section 4. The five treatments in this dataset are Search Ad, Social Ad, email sent, display ad in own domain, display ad outside own domain. The business question is to find the optimal subset of the treatments for each customer. Here, optimality is defined as the largest increase in the probability of achieving the desired outcome, which is a software subscription in this case. One might argue that some of the treatments cannot be chosen by a marketer. For example, a user searches and clicks on a search ad based on his or her own volition. It is not obvious how a marketer can influence this behavior. This is a valid point. However, the models are not used to directly target but inform targeting strategies. The counterfactual questions are hypotheticals asked to inform a decision. For example, TECE-VAE can be used to identify customers who can be influenced by a search ad. The profile data and search keywords of these customers can be used to find the profile of similar customer which can be used for targeting. A marketer can increase targeting of these keywords. Similarly, a customer matching the profile of someone influenced by a display ad can be sent display ad as and when the customer visits a website.

The results from the model show a few interesting findings. First, the prediction error as measured by RMSE of the predicted outcome is lower for TECE-VAE (0.397) than GBM (0.428) and RF (0.417), two popular supervised learning models. The RMSE of the observed outcome is not a suitable for evaluating a causal inference model. However, a causal inference model that fits the observed data is always desirable. One possible reason for the better performance might be due to explicit modeling of the latent state in TECE-VAE and use of a task embedding.

Second, the raw estimates of the treatment, as measured by the difference between the average conversion percentage for the consumer with treatment applied and for the consumer without any treatment, overestimates the effect of the treatment as compared with the average treatment effect measured by the model. For example, the raw estimate and the average treatment effect (ATE) for the treatment "Search Ad" are 46.5% and 21.8%, respectively; for "Social Ad" 37.8% and 13.8%, respectively; for "display ad outside own domain" 44.4% and 22.2%, respectively; for "display ad in own domain" 49.4% and 34.0%, respectively; and for "email sent" 22% and -3.42%, respectively. These difference in the two estimates show the extent of assignment and self-selection bias present in data. The effect of "email sent" is lowest among the treatments in both estimates. The ATE estimated using the model is a small but negative value! This neither means that this effect is negative for all individuals nor that sending email with other treatments in not optimal.

Third, the marketing actions that are actually applied are rarely optimal. The optimal marketing action suggested by TECE-VAE is actually observed only for 0.034% of the users. This might be due to the fact that some marketing action such as sending an email are easier than making someone actually search for an ad. However, the model shows that some marketing actions, such as sending and email or targeting with a social media campaign, might have a negative impact.

Fourth, in the test data only 15 out of a total of 32 subsets are optimal for at least one customer. Out of these only 8 subsets of the treatments are optimal for more than 50 users. None of these 15 optimal subsets are single treatment subsets. The subset that is optimal for the largest number of users is to apply all treatments but a social media campaign. Another important observation is that the effect of multiple treatments is not linear. For example, effect of showing an ad on own domain to everyone is a 11% points increase is likelihood of conversion. Whereas for a search ad to everyone increases the probability of subscription by 8% points. When the two treatment are applied jointly to everyone the increase in the probability is 23% points.

In summary, the results on real data show that the raw estimates from the data are highly biased, the use of model can lead to a large increase in the outcome metric, and the effect of multiple marketing actions is non-linear. These observations underline the need for a flexible model to to estimate ITE.

7 CONCLUSION

This paper provides a scalable approach for estimation of individual treatment effect using observational data in a setting where a subset from a set of treatments can be applied. We use task embedding to scale the model presented in [25] for multiple treatments. TECE-VAE is robust to unobserved treatments and uses information across treatments. We show the improvement in a variety of of experiments. TECE-VAE is compared against multiple baseline models that have shown to work well under different setting. The baselines also include models that have done well in causal effect estimation

competitions. TECE-VAE is applied on a real world dataset to solve a real causal inference problem.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

[2] Elias Bareinboim and Judea Pearl. 2012. Controlling Selection Bias in Causal Inference. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research)*, Neil D. Lawrence and Mark Girolami (Eds.), Vol. 22. PMLR, La Palma, Canary Islands, 100–108. http://proceedings.mlr.press/v22/bareinboim12.html

[3] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[4] Hugh A Chipman, Edward I George, Robert E McCulloch, et al. 2010. BART: Bayesian additive regression trees. *The Annals of Applied Statistics* 4, 1 (2010), 266–298.

[5] William B Dodds, Kent B Monroe, and Dhruv Grewal. 1991. Effects of price, brand, and store information on buyers' product evaluations. *Journal of marketing research* (1991), 307–319. https://www.jstor.org/stable/3172866?seq=1#page_scan_tab_contents

[6] Vincent Dorie, Jennifer Hill, Uri Shalit, Marc Scott, and Dan Cervone. 2017. Automated versus do-it-yourself methods for causal inference: Lessons learned from a data analysis competition. *arXiv preprint arXiv:1707.02641* (2017).

[7] Peter S Fader, Bruce GS Hardie, and Ka Lok Lee. 2005. RFM and CLV: Using iso-value curves for customer base analysis. *Journal of marketing research* 42, 4 (2005), 415–430.

[8] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[9] Wayne A Fuller. 2009. *Measurement error models*. Vol. 305. John Wiley & Sons.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[11] Sander Greenland and Timothy Lash. 2008. Bias analysis. In *Modern epidemiology* (3 ed.), Kenneth J Rothman, Sander Greenland, Timothy L Lash, et al. (Eds.). Lippincott Williams & Wilkins, 345–380.

[12] Sander Greenland, James M Robins, and Judea Pearl. 1999. Confounding and collapsibility in causal inference. *Statistical science* (1999), 29–46.

[13] Zvi Griliches and Jerry A Hausman. 1986. Errors in variables in panel data. *Journal of econometrics* 31, 1 (1986), 93–118.

[14] Nabanita Datta Gupta and Marianne Simonsen. 2010. Non-cognitive child outcomes and universal high quality child care. *Journal of Public Economics* 94, 1-2 (2010), 30–43. https://www.sciencedirect.com/science/article/abs/pii/S0047272709001169

[15] Matthew John Hausknecht. 1995. Cooperation and Communication in Multiagent Deep Reinforcement Learning. , 114–117 pages.

[16] Miguel A. Hernán, Sonia Hernández-Díaz, and James M. Robins. 2004. A Structural Approach to Selection Bias. *Epidemiology* 15, 5 (2004), 615–625. http://www.jstor.org/stable/20485961

[17] Jennifer L Hill. 2011. Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics* 20, 1 (2011), 217–240.

[18] Guido W Imbens and Donald B Rubin. 2015. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press.

[19] Fredrik Johansson, Uri Shalit, and David Sontag. 2016. Learning representations for counterfactual inference. In *International Conference on Machine Learning*. 3020–3029.

[20] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[21] Diederik P Kingma and Max Welling. 2014. Stochastic gradient VB and the variational auto-encoder. In *Second International Conference on Learning Representations, ICLR*.

[22] Manabu Kuroki and Judea Pearl. 2011. Measurement Bias and Effect Restoration in Causal In-ference. (2011).

[23] T.L Lai and Herbert Robbins. 1985. Asymptotically Efficient Adaptive Allocation Rules. *Adv. Appl. Math.* 6, 1 (March 1985), 4–22. https://doi.org/10.1016/0196-8858(85)90002-8

[24] Michael J Lopez, Roee Gutman, et al. 2017. Estimation of causal effects with multiple treatments: a review and new ideas. *Statist. Sci.* 32, 3 (2017), 432–454.

[25] Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. 2017. Causal effect inference with deep latent-variable models. In *Advances in Neural Information Processing Systems*. 6446–6456.

[26] Judea Pearl. 2009. *Causality*. Cambridge university press.

[27] Judea Pearl. 2012. On measurement bias in causal inference. *arXiv preprint arXiv:1203.3504* (2012).

[28] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[29] Donald B Rubin. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology* 66, 5 (1974), 688.

[30] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. 1986. Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1. MIT Press, Cambridge, MA, USA, Chapter Learning Internal Representations by Error Propagation, 318–362. http://dl.acm.org/citation.cfm?id=104279.104293

[31] Uri Shalit, Fredrik D. Johansson, and David A Sontag. 2017. Estimating individual treatment effect: generalization bounds and algorithms. In *ICML*.

[32] Dustin Tran, Alp Kucukelbir, Adji B Dieng, Maja Rudolph, Dawen Liang, and David M Blei. 2016. Edward: A library for probabilistic modeling, inference, and criticism. *arXiv preprint arXiv:1610.09787* (2016).

[33] Yixin Wang and David M Blei. 2018. The Blessings of Multiple Causes. *arXiv preprint arXiv:1805.06826* (2018).

[34] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. 2018. GANITE: Estimation of Individualized Treatment Effects using Generative Adversarial Nets. In *International Conference on Learning Representations*.

A Human-in-the-loop Attribute Design Framework for Classification

Md Abdus Salam
The University of Texas at Arlington
mdsalam@uta.edu

Mary E. Koone
The University of Texas at Arlington
mary.koone@mavs.uta.edu

Saravanan Thirumuruganathan
QCRI, HBKU
sthirumuruganathan@hbku.edu.qa

Gautam Das
The University of Texas at Arlington
gdas@uta.edu

Senjuti Basu Roy
New Jersey Institute of Technology
senjutib@njit.edu

ABSTRACT

In this paper, we present *a semi-automated, "human-in-the-loop" framework for attribute design* that assists human analysts to transform raw attributes into effective derived attributes for classification problems. Our proposed framework is optimization guided and fully agnostic to the underlying classification model. We present an algebra with various operators (arithmetic, relational, and logical) to transform raw attributes into derived attributes and solve two technical problems: (a) the **top-k buckets design** problem aims at presenting human analysts with k buckets, each bucket containing promising choices of raw attributes that she can focus on only without having to look at all raw attributes; and (b) the **top-l snippets generation** problem, which iteratively aids human analysts with top-l derived attributes involving an attribute. For the former problem, we present an effective exact bottom-up algorithm that is empowered by pruning capability, as well as random walk based heuristic algorithms that are intuitive and work well in practice. For the latter, we present a greedy heuristic algorithm that is scalable and effective. Rigorous evaluations are conducted involving 6 different real world datasets to showcase that our framework generates effective derived attributes compared to *fully manual or fully automated methods*.

CCS CONCEPTS

• **Information systems** → *Crowdsourcing*;

KEYWORDS

human computation; crowdsourcing; attribute design; feature engineering

ACM Reference Format:
Md Abdus Salam, Mary E. Koone, Saravanan Thirumuruganathan, Gautam Das, and Senjuti Basu Roy. 2019. A Human-in-the-loop Attribute Design Framework for Classification. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313547

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313547

1 INTRODUCTION

Attribute design (also known as feature engineering) is one of the most challenging aspects of a data science pipeline, and is considered to be an "arduous process for data scientists"[1] [3, 14], where the raw attributes often need to be transformed into derived attributes that can be more effective for building predictive models such as classifiers. For example, in a healthcare setting involving large, high dimensional and heterogeneous electronic health records (EHR) datasets, an attribute such as the average length of prior hospitalization can be very useful to build effective predictive or classification models on future hospitalization (i.e., readmission) [4]); however, such attributes are not readily available in the raw dataset. For example, Table 1 shows that the average length of hospitalization needs to be computed per hospitalization window considering admission and discharge date, and taking the average of these windows.

The state-of-the-art attribute design techniques fall into one of these two extremes: (a) *fully manual*, painstakingly slow and heavily reliant on domain expertise, often requiring data scientists to go through a repetitive trial-and-error exercise until the set of designed attributes are satisfactorily effective or (b) *fully automated* techniques (some notable systems are, Data Science Machine [18][2], ExploreKit [20][3], One Button Machine [23], or Featuretools[4]). Such methods are not model agnostic, require substantial time to identify derived attributes that are opaque to the human analyst.

In this paper, we investigate a semi-automated "human-in-the-loop" framework that is agnostic to any classification model. Our work is inspired by a handful of recent works [8, 36, 50] which propose a conceptual framework and empirically argue that attribute design by involving human analysts can effectively substitute for either of the two extremes (fully manual or fully automated). However, these do not study how to optimize human involvement in the process or how to guide it.

Proposed Framework: Our proposed framework is developed around three fundamental aspects.

Model agnostic measure: We adopt a principled and quantifiable measure of the effectiveness of a derived attribute that is agnostic to the specifics of any underlying classification model. We consider *Mutual Information* (MI in short) [38] to determine

[1] https://www.itproportal.com/features/dont-let-feature-engineering-stagnate-your-ml-projects/
[2] https://people.csail.mit.edu/kalyan/dsm/
[3] https://github.com/giladkatz/ExploreKit
[4] https://docs.featuretools.com/#minute-quick-start

the predictive ability of an attribute. Intuitively, MI is a symmetric measure that captures "correlation" between a pair of attributes and quantifies how much information is contained in one attribute about the other. We are interested in producing derived attributes that have high MI with the target variable (or the class label). MI is known to have several desirable properties: (a) MI is proved to have certain qualitative guarantees when chosen for attribute selection for multiple popular classification models, such as Naive Bayes [38], or linear regression [12]; (b) prior works have also shown that MI optimizes important properties in attribute selection, such as; relevance, complementarity, and redundancy [38] and (c) finally, as we shall show later in the paper, MI satisfies upward closure [6] which is useful for designing effective algorithms.

Attributes algebra: The second aspect of our framework is an algebra that dictates how raw attributes (numerical or categorical) are to be combined to produce the derived attributes. We study arithmetic, logical, and relational operators. Our initial example of average length of prior hospitalization is an example of arithmetic operator, whereas an example of the logical operation is the following Boolean attribute: *elderly* AND *diabetic* OR *covered under medicaid*. A patient is more vulnerable to future hospitalization if she gets an "yes" on this attribute. A logical operator on the other hand is *obesity* that is set to True, when $BMI > 30$.

Guiding human analysts: The third aspect of our framework is the investigation of how to optimize the involvement of humans in the attribute design process and ensure their efforts are successful. This is motivated by a handful of recent works that argue that humans must not be left unguided in the attribute design process [8, 36, 50], Simply presenting the entire set of raw attributes or hundreds and thousands of raw records to the human analysts might overwhelm them. We propose to present each analyst with small summarized portions of the raw data and raw attributes that is likely to be most helpful for the human to analyze and create useful attributes from. We formulate two technical problems that need to be addressed:

(1) Top-k buckets design: We present each analyst with only k (a small number) *buckets*, each with at most x attributes, that are most predictive. For example, when $x = 3$, a useful bucket that is likely to have predictive qualities may contain {elderly, diabetic, covered-by-medicaid}. We show that generating these buckets is NP-hard. We provide an exact algorithm, `ExBKT`, and an even more efficient heuristic algorithm called `RandomizedBKT` that performs random walks on the attribute lattice to design the top-k buckets. However, just producing buckets that are highly predictive (high MI) may not be enough. We propose `RandomizedCovBKT` that finds top-k buckets that are not only effective wrt MI, but also they together cover many "good" raw attributes of the dataset.

(2) Top-l snippets generation: Even after the buckets of raw attributes have been generated, the human analyst may have to sift through large volumes of data (i.e., the tuples with those raw attribute values) to design good derived attributes, and this task may still be overwhelming. Thus, we propose an interactive procedure by which an analyst may approach this task. Given a bucket, the analyst starts designing a derived attribute (i.e., an algebraic expression involving the raw bucket attributes) interactively (i.e, term by term). At each iteration, our framework recommends l *snippets*, i.e., it suggests the l best ways to extend the partially created derived attributes using the algebra. More formally, at a given iteration the snippet generation process suggests how to augment the partially composed derived attribute by length j, i.e., combining j new raw attributes with the derived attribute developed thus far. For instance, if *elderly* has already been selected by the analyst, and if $j = 1$, an example snippet will suggest a visual distribution that shows how *elderly* AND *diabetic* correlates with the prediction target variable *hospital readmission*.

Evaluations: Rigorous evaluations are conducted considering 6 real world datasets by comparing our solutions with two fully automated methods (ExploreKit and Featuretools), and a fully manual domain expert guided attribute design process, as well as intuitive baselines. Our experimental results demonstrate that we scale up to $7x - 20x$ faster compared to fully automated (ExploreKit) and the fully manual process, while ensuring similar quality (average improvement 14%). By leveraging the domain expert, our framework avoids falling into the pitfall where derived features could actually be detrimental to the performance (unlike FeatureTools). It also has other appealing properties, such as being easily parallelizable, and exhibiting "anytime behavior" whereby it gives meaningful results at any point of execution. Our scalability results demonstrate that both bucket design and snippet generation procedures are efficient and can work interactively with the human in the loop.

In summary, the paper makes the following contributions:

- Proposed Framework: We initiate the study of a model agnostic semi-automated attribute design framework (Section 2) that judiciously involves human analysts in the loop.
- Technical contributions: We formalize two technical problems around the framework and present several theoretical and algorithmic results (Sections 3 and 4).
- Experimental Results: We conduct extensive experiments to demonstrate both effectiveness and scalability of our proposed solutions (Section 5).

2 PRELIMINARIES AND FORMALISM

In this section, we describe our data model, present our framework, and formalize the technical problems.

Example 2.1. We present a toy running example in Table 1 that provides longitudinal data of a heart failure datamart in a hospital. The objective is to build a classifier that predicts whether a patient getting discharged from the hospital will be readmitted within 6 months of discharge, considering predictors from base attributes (first 10 columns) and derived attributes (designed using the base attributes). The data is augmented by adding the last column, representing the class label per patient per admission instance.

2.1 Data Model

Base attributes, records, and target variable: A given dataset is comprised of a set $\mathcal{A}$ of n attributes and m records, as well as an additional target attribute (column) Z. These attributes are referred to as base or raw attributes. In this work, we consider all major types of attributes including numeric, Boolean, and categorical. Our proposed approach can work for a wide variety of predictive modeling tasks including classification and regression (we refer to them classification in general). Depending on the nature of the

patient-id	admission date	dis. date	gender	BMI	income	medicaid	senior	diabetic	primary-diagnosis	readmission
p1	10-1-2013	10-23-2013	M	32.5	low	Y	Y	Y	Congestive heart failure	N
p1	5-2-2014	5-10-2014	M	32.2	low	Y	Y	Y	Heart Attack	Y
p2	5-12-2014	5-14-2014	F	26.7	medium	N	N	N	Arrhythmia	N
p3	10-1-2015	11-1-2015	M	29.9	low	N	N	Y	Cardiomyopathy	N
p3	6-10-2016	6-21-2016	M	29.8	low	N	N	Y	Cardiomyopathy	N
p4	12-12-2017	12-14-2017	F	23.4	medium	N	N	N	Arrhythmia	N
p4	9-2-2018	9-29-2018	F	27.8	medium	N	N	N	Heart Attack	Y

Table 1: Example 2.1 toy heart failure datamart

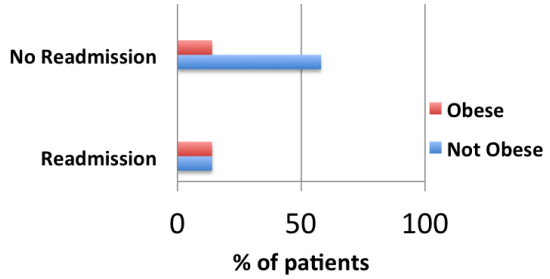


Figure 1: A snippet on Obesity

problem, Z is a continuous variable (regression problem), or has discrete values (classification problem). Using Example 2.1, $n = 10$ and $m = 7$, Z corresponds to readmission within 6 months of discharge (discrete).

Algebra, $\mathcal{L}$: A set of operators (arithmetic, relational and logical) which are applied to one or more base attributes to combine them. When the base attributes are numeric, we consider arithmetic operators to combine them: addition (+), subtraction (−), multiplication (x), division (/). For Boolean attributes, we consider logical operators, AND, OR. We also support all relational operators that compare two base attributes or a base attribute with a constant (e.g. $A_i > A_j$ or $A > 100$). Our framework is flexible enough to support arbitrary operators. We also support aggregate operators over a set of tuples.

Derived Attribute, d: An attribute that combines two or more base attributes using the algebra $\mathcal{L}$. Using Example 2.1, we can create a derived attribute *length of stay* by subtracting discharge date from the admission date. Another derived attribute *obesity* can be obtained through the relational operator > as $BMI > 30$.

Independent variable, dependent variable: A base attribute or a derived attribute is an independent variable (V) in our problem as that is used to predict the target variable Z, which is the dependent variable. Our overall intention is to craft a set of derived attributes as independent variables that are highly "predictive" to the target (dependent) variable.

Mutual Information (MI): We are interested in calculating "predictiveness" of an independent variable V to the target variable Z. For that, we use Mutual Information (MI) that captures information theoretic "correlation" (indeed there exists a relationship between MI and correlation [25] between two random variables that quantifies the amount of information obtained about one through the other). When V and Z are discrete [5], $MI(Z, V)$ is defined as follows:

$$MI(Z,V) = \sum_{z \in Z} \sum_{v \in V} p(z,v) \log \frac{p(z,v)}{p(z)p(v)} \quad (1)$$

where $p(z, v)$ is the joint probability function of Z and V, and $p(z)$ and $p(v)$ are the marginal probability distribution functions of Z and V respectively. Of course, V could be a single base attribute, a small set of base attributes, or a derived attribute.

2.2 Proposed Framework

Our proposed framework consists of two technical steps. We first design a set of k buckets, each with at most x base attributes. Given a bucket and the algebra $\mathcal{L}$, the analyst then starts composing a small number of derived attributes from the bucket. This next step is referred to as *snippet generation*, and works iteratively with the analyst until she decides to stop. In each step in snippet generation, the analyst extends the currently composed derived attributes by a small amount. Our algorithmic contributions are focused on these two steps. In Section 3, we analyze the Top-k buckets design problem and describe our solutions. In Section 4, we present our solutions for the Top-l snippets generation problem.

Definition 2.2. **Score of a bucket, $sc(b)$:** For a bucket b with $A_1, A_2, \ldots A_x$ base attributes, the score of b, i.e., $sc(b)$ is the MI between Z and the Cartesian product of the base attributes that are part of b.

$$sc(b) = MI(Z, [A_1 \times A_2 \times A_3 .... \times A_x]) \quad (2)$$

Using Example 2.1, $sc(gender, senior) = MI(readmission, [gender \times senior])$

Definition 2.3. **Coverage of a set of buckets:** coverage of a set of buckets $Cov(b_1, b_2, ..b_k)$ is the size of the union of the base attributes that are present in these buckets.

$$Cov(b_1, b_2, ..b_k) = |b_1 \cup b_2 \cup ....b_k| \quad (3)$$

Using Example 2.1, if $k = 2, x = 2$, $b_1 = [gender, senior]$, $b_2 = [medicaid, gender]$, then $Cov(b_1, b_2) = 3$.

Definition 2.4. **Snippet, s:** A snippet s is a visual representation of a joint distribution between Z and a derived attribute d^s in the snippet.

Figure 1 shows one such snippet between obesity and hospital readmission.

[5]We consider the numeric variables are appropriately discretized, when needed

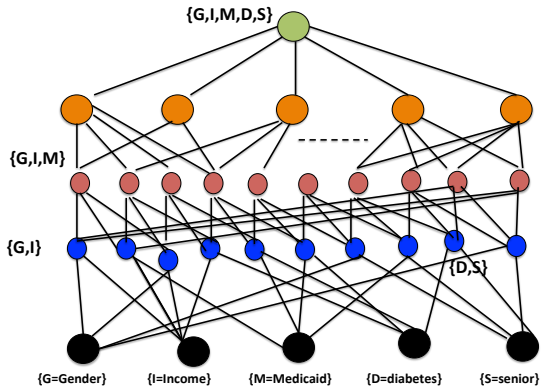


Figure 2: Attribute Lattice

Definition 2.5. **Score of Snippet, $Sc(s)$:** Score of a snippet is the MI between Z and derived attribute d^s in the snippet s, i.e., $MI(Z, d^s)$.

Using Figure 1, $Sc(obesity) = MI(readmission, obesity)$.

2.3 Problem Definitions

Problem 1: Top-k-Buckets Design: Given a set of attributes A, number of required buckets k and maximum number of attributes in each bucket x, our objective here is to design Top-k buckets based on MI, i.e., finding the k-buckets (each with at most x attributes) with the highest MI and present those buckets to the the human analyst to investigate further for creating derived attributes.

We also investigate the top-k coverage aware bucket generation problem, where the objective is to create "high quality" buckets *that together cover* all base attributes that have high MI with the target variable.

Problem 2: Coverage Aware Top-k buckets: The objective is to create Top-k buckets such that the score of each bucket, $sc(b)$ in Top-k is above a certain threshold δ and $Cov(b_1, b_2, ..b_k)$ is maximized.

Problem 3: Top-l Interactive Snippets Generation: Each step of the interactive snippet generation takes as inputs a bucket b, an integer j, the currently composed set of derived attributes $\mathcal{D}'$, the algebra $\mathcal{L}$; and produces l snippets with highest scores, where each derived attribute d_i'' in snippet s_i is created by extending d_i', that is, by adding j additional attributes that are part of b, involving $\mathcal{L}$.

3 TOP-k BUCKETS DESIGN

Top-k Buckets Design takes x (the maximum size of each bucket), k (the number of buckets), the dataset (base attributes A, target variable Z, and the records), and produces Top-k buckets, each of size at most x that are highest in MI with Z.

Theorem 3.1. *The Top-k Buckets Design Problem is NP-hard, for an arbitrary x and k.*

Proof. (Sketch): As shown in [46], for an arbitrary maximum bucket size x and number of buckets k, the problem of identifying the number of distinct buckets of maximum size x (that have maximal MI in our case) is #P-Hard. Therefore, the enumeration problem of finding top-k buckets of maximum size x is NP-hard. □

Intuitively, the bucket design problem bears similarity with the itemset mining problems based on association rules (support) or other correlation measures [1], such as Chi-Square [6]. WRT our problem, a base attribute could be considered an item, and therefore mining a bucket with at most x-base attributes is akin to mining an itemset with at most x items that satisfy a certain property. Thereby, the Top-k buckets generation problem seemingly appears similar to Top-k itemset mining problems.

Most popular and effective algorithms in this problem space make use of the *upward or downward closure property of the itemsets* based on the underlying measures (for example, support is downward closed, while chi-square is upward closed [6]). These properties enable efficient algorithm design for the itemset mining problems.

Popular algorithms such as Apriori [1] make use of this property extensively in mining frequent itemsets. They execute in a bottom-up manner by discarding any itemset from consideration whose subsets are not frequent based on the support threshold [1].

However, designing an Apriori [1] type of algorithm is impractical for our problem for several reasons: firstly, because *MI is not downward closed;* secondly, because our problem does not have a *support threshold* like Apriori, and finally because our problem has *more constraints* (x and k) as inputs.

Lemma 3.2. *Mutual Information is upward closed.*

Proof. Without loss of generality, let us assume Z is the target variable and A_1, A_2, A_3 are three base features. We have to prove $MI(Z, [A_1, A_2]) \leq MI(Z, [A_1, A_2, A_3])$.

Based on MI definition [26]:

$$MI(Z, [A_1, A_2]) = H(Z) - H(Z|[A_1, A_2]) \tag{4}$$

where $H(Z|[A_1, A_2])$ is the conditional entropy of Z given A_1, A_2.

Using Equation 4, we therefore prove $H(Z|[A_1, A_2]) \geq H(Z|[A_1, A_2, A_3])$.

$H(Z|[A_1, A_2, A_3])$

$$\begin{aligned}
&= \Sigma_{a_1,a_2,a_3} p(a_1, a_2, a_3) \times \\
&\Sigma_z p(z|a_1, a_2, a_3) \log p(z|a_1, a_2, a_3) \\
&\leq \Sigma_{a_1,a_2,a_3} p(a_1, a_2) \times p(a_3) \Sigma_z p(z|a_1, a_2) \times \\
&p(z|a_3) \log p(z|a_1, a_2) \log p(z|a_3) \\
&\leq H(Z|[A_1, A_2]) \times \Sigma_{a_3} p(a_3) \Sigma_z p(z|a_3) log p(z|a_3) \\
&\leq H(Z|[A_1, A_2])
\end{aligned} \tag{5}$$

Therefore, $H(Z|[A_1, A_2]) \geq H(Z|[A_1, A_2, A_3])$, proving $MI(Z, [A_1, A_2]) \leq MI(Z, [A_1, A_2, A_3])$. □

An Apriori-Like Algorithm is Impractical. Since MI is upward closed, a level-by-level algorithm such as Apriori must be done in a top-down fashion for our problem, as opposed to the typical bottom-up fashion. Our problem has an additional constraint on the maximum bucket size that requires the algorithm to climb to level x first before it can start generating possible buckets. Moreover, our bucket generation problem does not come with any provided threshold of MI. Instead, it has the number of buckets constraint k. Therefore, it has to start at level x (which will have $\binom{n}{x}$ number of buckets of size x) with the highest MI value possible as the threshold and determine all the size x buckets that satisfy the threshold. If the

number of generated buckets is less than k, it then has to reduce the MI threshold value systematically until a total of exactly k buckets have been generated. Naturally, to faithfully reproduce Apriori for our problem, one has to make several runs of the algorithm, until exactly k buckets have been produced. Clearly, such a process is computationally impractical for large n, x, and k.

Next, we describe our solutions for this problem - first an exact algorithm (ExBKT) that produces the top-k buckets of at most size x with the highest MI, and then random walk based algorithms that are highly efficient and work well in practice.

3.1 Exact Algorithm

Algorithm ExBKT extensively exploits a few observations that we make about MI. In particular, while computing the MI of a set of attributes with the target variable Z, we observe that one can produce effective *lower and upper bounds* on MI. These bounds shall allow us to design a bottom up algorithm that goes level by level and effectively prunes a set of candidate attributes (buckets) by using upper bounds. The observation is rather simple - between two subsets of candidate buckets, if one has higher lower bound of MI than the other bucket's upper bound of MI, then the former bucket should get promoted as the winner between these two buckets. Algorithm ExBKT makes use of this observation to prune buckets that are never going to be part of the top-k results. Before we describe this algorithm in detail, we describe how to effectively compute lower and upper bound (LB and UB respectively) of MI of a set of attributes (candidate buckets).

Theorem 3.3. *Upper Bound: Given a set of t base attributes $A_1, A_2, \ldots A_t$ and the target variable Z, $MI(Z, [A_1, A_2, \ldots A_t]) \leq H(Z) + H(A_1, A_2, \ldots A_t)$*

Proof.

$$\begin{aligned} MI(Z, [A_1, A_2, \ldots A_t]) &= H(Z) - H(Z|[A_1, A_2, \ldots A_t]) \\ &= H(Z) - H(Z, [A_1, A_2, \ldots A_t]) + \\ & H(A_1, A_2, \ldots A_t) \text{ since } [H(Y|X) = H(X, Y) - H(X)] \end{aligned} \quad (6)$$

Therefore,

$$MI(Z, [A_1, A_2, \ldots A_t]) \leq H(Z) + H(A_1, A_2, \ldots A_t) \quad (7)$$

Applying the chain rule of entropy, the right hand side of the inequality could be expressed further as $H(Z) + H(A_1) + H(A_2|A_1)$ $+H(A3|A_1A_2) + \ldots H(A_t|A_1, A_2, \ldots A_{t-1})$ □

Theorem 3.4. *Lower Bound: Given a set of t base attributes $A_1, A_2, \ldots A_t$ and the target variable Z, $MI(Z, [A_1, A_2, \ldots A_t]) \geq MI(Z, [A_1, A_2, \ldots A_{t-1}])$*

Proof. This comes directly from lemma 3.2. □

Algorithm Description: Algorithm ExBKT runs in a bottom-up fashion. It starts at the bottom of attribute lattice with singleton base attributes as buckets, and gradually walks up the lattice, level by level. For illustration purposes, consider only 5 attributes from Example 2.1, abbreviated $\{G, I, M, D, S\}$. The bottom layer refers to the 5 singleton base attributes and their computed MI wrt the target variable Z. Once the score of each of size 1 bucket is computed, then in the next level, the algorithm combines two base attributes and computes 10 buckets of size 2. For each bucket b_i, it maintains two scores: the lower bound score of b_i, $sc^{lb}(b_i)$ derived from Theorem 3.4 and the upper bound score of b_i, $sc^{ub}(b_i)$ derived from Theorem 3.3. Between two buckets b_i and b_j, if the upper bound of b_i (i.e., $sc^{ub}(b_i)$) is smaller than the lower bound of score of b_j (i.e., $sc^{lb}(b_j)$), then b_i and all of its supersets get dropped from further consideration. Using Figure 2, if $sc^{lb}\{G, I\}$ is larger than the upper bound score $sc^{ub}\{M, D\}$, then the latter bucket gets dropped from further consideration. Additionally, all buckets that contain $\{M, D\}$ as some base attributes do not need to be considered. If $x = 3$, then the algorithm continues to climb up the attribute lattice to the next level, finding buckets with at most size 3 base attributes, but applies pruning between the buckets using the lower and upper bound as before. Once it finishes the traversal, it produces the top-k buckets with the k-highest MI with the target variable. The pseudo-code is described in Algorithm 1.

Algorithm 1 Algorithm ExBKT

```
inputs: 𝒜, m records, x, k, target variable Z.
output: Top-k buckets, each of size at most x.
compute all buckets of size 1 and their score.
i = 2
while  i ≤ x do
    Compute sc^ub(b) and sc^lb(b) of each bucket b;
    IF  sc^ub(b_w) ≤ sc^lb(b_y)
        Drop b_w and any superset of b_w
    i = i + 1
end while
```

Lemma 3.5. *Algorithm ExBKT produces the exact top-k buckets.*

Proof. (sketch:) The intuitive argument is that ExBKT only drops a bucket and its super-set if its upper bound score is not larger than the lower bound score of other buckets, which will produce the exact solution, because of Lemma 3.2. □

Running Time: In the worst case ExBKT may take $O(n^x)$, hence it is exponential. However, in practice, the pruning can be very effective and the algorithm converges much sooner.

3.2 Random Walk Based Algorithms

While ExBKT works well in practice, it has an exponential running time in the worst case. Furthermore, ExBKT is not easily extensible when we have to optimize any other criteria in addition to MI. Specifically, for our proposed problem *Coverage Aware Top-k buckets*, ExBKT can not be adapted to generate the best k-buckets with the highest coverage. Therefore, we propose faster heuristic alternatives that provide good solutions most of the time.

Our proposed Algorithm RandomizedBKT is motivated by random walk on the attribute lattice [6] and has a unified solution for both *Top-k buckets Design* and *Coverage Aware Top-k buckets Design* problems. The core idea is to compose a bucket of size x, by performing a random walk on the attribute lattice (refer to Figure 2). An attribute is added to a bucket by using weighted sampling without replacement. The weight of an attribute is based on the optimization criteria. A bucket is formed by performing a random walk on the attribute lattice, until its maximum size x has been reached. This

random walk is repeated, until k-unique buckets have been derived and any additional criteria (e.g., threshold δ for the latter problem) has been satisfied. With this high level description above, now we describe how to compute the weight of an attribute in different scenarios. For the *Top-k buckets Design* problem, the weight of an item is directly proportional to its MI, i.e., $weight(A) = MI(Z, A)$.

The *Coverage Aware Top-k buckets Design* problem requires that in addition to MI threshold δ, the coverage of the Top-k buckets also must be maximized. The weight of an attribute is a ratio; it is proportional to its MI, but inversely proportional to the number of times it is present in other buckets that have been computed thus far. Indeed, this latter criteria is designed to ensure high coverage.This condition can be extended to include feature cost as well, so that costly features are assigned lower preference. As before, the algorithm terminates when we obtain top-k unique buckets. Using Figure 2, if $k = 5$ and $x = 2$, when the random walk produces the 5th bucket, if M has appeared in all other 4 buckets, the weight of M is dampened by a factor of 4 even if it is high in MI with readmission. The pseudo-code is described in Algorithm 2.

Algorithm 2 Algorithm `RandomizedBKT`

inputs: set of base attributes $\mathcal{A}$, m records, target variable Z, x, k, δ (for the coverage aware problem)
output: Top-k buckets of size at most x
$\mathcal{B} = \{\}$
while $|\mathcal{B}| < k$ **do**
  while $size(b_i) < x$ **do**
    $weight(A) = MI(Z, A)$ ▷ for the top-k variant
    $weight(A) \propto \frac{MI(Z,A)}{\#A \text{ has been used before}}$ ▷ for the Coverage aware variant
    Sample A based on $weight(A)$ and add it to b_i
  end while
  IF $sc(b) < \delta$ ▷ only for the Coverage aware variant
    Drop b
  $\mathcal{B} = \mathcal{B} \cup b$
end while

Running Time: Each run of `RandomizedBKT` takes at most $O(n)$ time. The total running time of the algorithm is dominated by the number of different random walks that needs to be performed before it returns k buckets.

4 TOP-l INTERACTIVE SNIPPETS GENERATION

Recall that we propose snippet generation as an interactive process that continues until the human is done crafting the attributes. Each step of this process takes as inputs a bucket b, an integer j, the currently composed set of derived attributes $\mathcal{D}'$, and the algebra $\mathcal{L}$ to produce l snippets with highest scores (MI with Z), where each derived attribute d_i'' in snippet s_i is created by extending d_i' by adding j additional attributes that are part of b, involving $\mathcal{L}$. Each of these l snippets are recommended to the human as visual distributions to aid her design derived attributes.

Therefore, our computational challenge is to produce l snippets effectively in each step involving each $d_i' \in \mathcal{D}'$. A natural choice is to investigate a greedy algorithm that builds the final derived attribute bottom-up by selecting the best base attributes from the bucket and combining them with the best operators to recommend a snippet. In order for this greedy algorithm to provide any provable guarantee, this greedy selection process needs to satisfy certain properties. For example, given four attributes A_i, A_j, A_r, A_s, if $MI(Z, A_i) > MI(Z, A_r)$, and $MI(Z, A_j) > MI(Z, A_s)$, then it seems intuitive that if we combine the two attributes with higher MI (A_i, A_j) with an operator, the resulting derived attribute should have higher MI than the combination of the other two attributes (A_r, A_s) using the same operator. Unfortunately, as we prove below with counter examples, such a property fails to hold *even for very simple operators such as arithmetic addition.* This makes the snippet generation more challenging than the previous step of bucket generation.

A_i	A_j	A_r	A_s	Z
1	-1	0	0	0
2	-2	1	0	1
3	-3	0	1	0
4	-4	1	1	1

Table 2: Example where $MI(Z, A_i + A_j) < MI(Z, A_r + A_s)$

Lemma 4.1. *Given four attributes A_i, A_j, A_r, A_s, if $MI(Z, A_i) > MI(Z, A_r)$, and $MI(Z, A_j) > MI(Z, A_s)$, $MI(Z, A_i + A_j)$ may or may not be greater than $MI(Z, A_r + A_s)$*

Proof. (Sketch): We prove this by counter examples. We present two examples - one shows $MI(Z, A_i + A_j) < MI(Z, A_r + A_s)$ and the other shows the inequality other way. Table 2 shows the first scenario. To create the second scenario, if we replace A_j with 5, 6, 7, 8 in the 4 different rows respectively, we have $MI(Z, A_i + A_j) > MI(Z, A_r + A_s)$. □

Proposed Algorithm: Our proposed algorithm `GSnippet` is a greedy algorithm - it still generates a snippet bottom up in each step. Given a bucket b, it first sorts the base attributes in the bucket in descending MI. Then given j (the number of additional attributes to extend a partially created derived attribute d_i'), it finds the top-j attributes in b that are not in d_i'. These are the candidate set of attributes for the snippets that involve d_i'. After that, it attempts to combine these j attributes considering all the operators in $\mathcal{L}$ with d_i'. It ranks each of these created combinations and produces the l-combinations as snippets that have the top-l MI score with Z. Algorithm 3 presents the pseudo-code.

The reason that `GSnippet` exhaustively considers all the operators in $\mathcal{L}$ (and cannot make any greedy or more efficient look-ups) is because of Lemma 4.1. In order to avoid exhaustive search, other alternative metrics to MI need to be explored that can guarantee upper/lower bound for features combined under algebraic operators. Nevertheless, as we shall show in our experiment, `GSnippet` runs at interactive speeds and produces effective recommendations for the human analyst.

Using Example 2.1, if $b = \{admission.date, dis.date\}$, $l = 1$, $j = 2$, $\mathcal{L} = \{+, -, \times, /\}$, and current $d = \{\}$ (i.e., empty), then `GSnippet` will rank $admission.date + dis.date$, $admission.date -$

dis.date, *admission.date* × *dis.date*, *admission.date*/*dis.date*, and take the one which has the highest MI with readmission.

Algorithm 3 Algorithm `GSnippet`

inputs: bucket b, a derived attribute d'_i, target variable Z, j, l, $\mathcal{L}$
output: Top-l snippets involving d'_i
Sort attributes in b in descending order of MI
Select set S with top-j attributes from b that are not in d'_i
$C = S \bigcup d'_i$
Combine $c_i \oplus c_j \oplus .... \oplus d'_i, \oplus \in \mathcal{L}, c_i \in C$
Rank each combination wrt *MI*
Return top-l

Running Time: Each run takes $O(|\mathcal{L}|^j log l)$ time, the majority of which is spent on brute-forcing on the operator set $\mathcal{L}$.

5 EXPERIMENTAL EVALUATION

We conducted comprehensive experimental analysis to compare our proposed approach with fully automated methods, as well as fully manual (domain expert guided) solution. We investigated both quality and running time in this process.

5.1 Experimental Setup

Hardware and Platform. All our experiments were conducted on a quad-core 2.2 GHz machine with 16 GB of RAM and 1 TB of hard disk. We used Python to implement our algorithms and used scikit-learn for building the classification models.

Datasets. We evaluated our algorithms against a wide variety of datasets that are considered to be popular choice for attribute design problems. Due to lack of space, we report our results on 5 datasets from UCI repository and one from Kaggle. They cover a diverse array of domains (agriculture, medicine, and e-commerce) and contain attributes that are amenable to constructing derived features. Table 3 has more details.

DataSet	# Records	# raw attributes
pollen	3848	5
delta_elevators	9517	6
mammography	11183	6
space	3107	6
diabetes	768	8
home	506	14

Table 3: Reported datasets characteristics

Compared Methods. *(1) Fully Automated Methods.* We implemented two fully automated state-of-the-art approaches for comparison - Featuretools[6] and ExploreKit[7]. They both have open source repositories that allow us to evaluate them fairly.

(2) Our proposed algorithms. These are the solutions that are presented in Sections 3 and 4.

[6]https://docs.featuretools.com/#minute-quick-start
[7]https://github.com/giladkatz/ExploreKit

(3) Buckets Design Baseline algorithms. We compared our proposed buckets design algorithm against an intuitive baseline algorithm (referred to as `Greedy`) that groups attributes on mutual information. It started with an empty bucket and x attributes were added through importance sampling greedily, where the importance is proportional to the marginal increase in MI. The process was repeated k times.

(4) Fully manual method. In this scenario, a domain expert was involved in crafting the derived attributes. We present a case study towards that in Section 5.5.

Evaluation of our proposed framework. Our proposed framework has two steps: the first one, top-k buckets design, is fully automated and does not require any human involvement. We have described three algorithms for that. The second step, top-l snippets generation, proposes the list of top l snippets for each of the derived attribute chosen by the human analyst. The human analyst can either accept the choice, select another snippet or even construct a new one. This process is repeated interactively. The framework continues to retain the top-l choices, if no human guidance is given and is fast enough to be done in near real-time. Finally, the designed derived and the base attributes are passed through existing popular classification models with an average of 10 runs.

Parameter Settings. *x, k, j, l, random walk:* We varied the size of a bucket x between 2 to the maximum number of base attribute with the default value being 5. The snippet extension parameter j is set to 1. Finally, both k and l were set to 5 by default. We ran the random walk for 1000 iterations and picked top-k from it.

Classification models. While our process is classifier agnostic, for the purpose of comparison, we considered two popular classifiers, Support Vector Machines (SVM) and Random Forests (RF). For the latter, we use 100 trees using a depth of 2. Training and testing are performed with a 70% − 30% split of the data, akin to ExploreKit.

Algebra. By default, we used only arithmetic operators. We then varied the grammar to include logical, relational and other aggregate operators. Overall, our approach can support all the operators described in both ExploreKit and Featuretools.

Performance Measures. *Qualitative measures.* For measuring the classifier performance, we reported the Area Under the Curve (AUC). This has been used in prior work such as ExploreKit as it provides a holistic view of the classifier and attribute design. Higher the AUC, better the classifier performance. We reported the percentage improvement in AUC with base(raw) attributes and base+derived attributes. For example, if the classifier had an AUC of 0.7 with base attributes but 0.8 with base+derived attributes, the improvement is $\frac{0.8-0.7}{0.7} = 14\%$.

Scalability measure. We used time in seconds to evaluate the efficiency of our algorithms.

5.2 Summary of Results

Our experimental analysis answered the following key questions:

- **How did our proposed framework compare against fully automated or fully manual solutions?** We found that we scale significantly better ($10x - 20x$ faster than ExploreKit, $7x$ better than fully manual) than both of these extremes, with comparable AUC improvements (on an average 14%). Due to human involvement, it also avoided the worst case

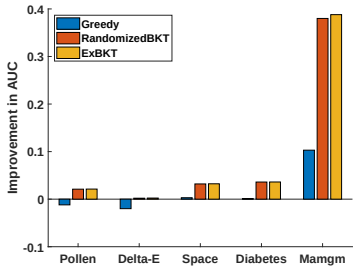


Figure 3: Comparing Baselines : AUC

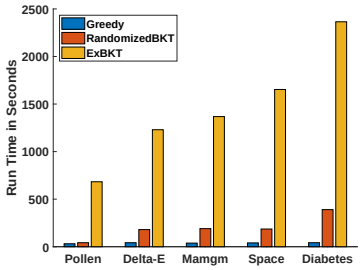


Figure 4: Comparing Baselines : Running Time

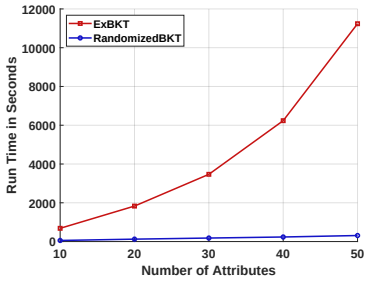


Figure 5: Varying Number of Attributes

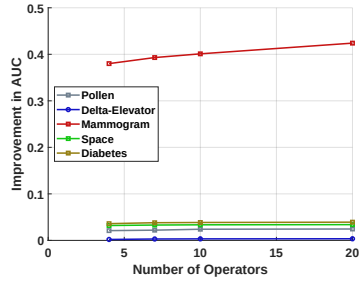


Figure 6: Varying Algebra

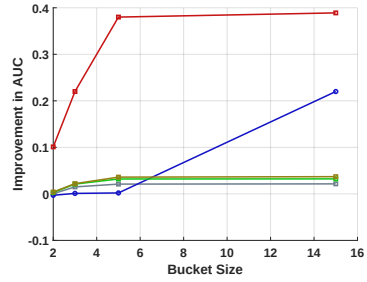


Figure 7: Varying Bucket Size

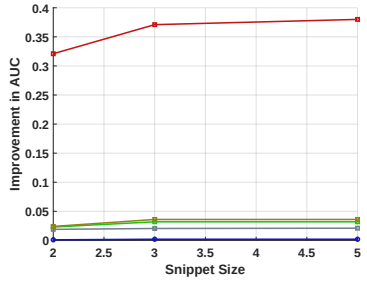


Figure 8: Varying Snippet Size

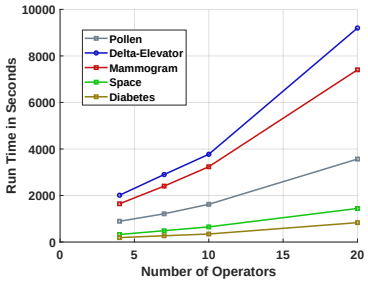


Figure 9: Varying Algebra

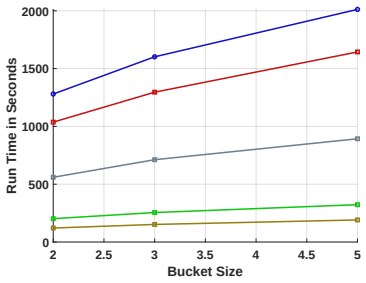


Figure 10: Varying Bucket Size

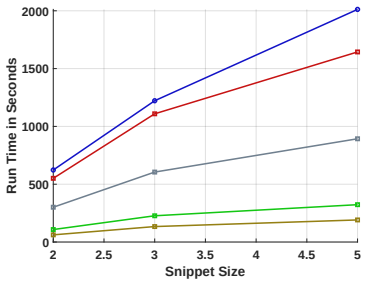


Figure 11: Varying Snippet Size

Data	EK-SVM	Time (s)	FT-SVM	Time (s)	MI-SVM	Time (s)
pollen	1.93%	1063	2.07%	93	2.10%	42
delta_elevators	0.23%	3980	-3.20%	465	0.20%	180
mammography	31.61%	2413	41.93%	204	38.00%	191
space	3.83%	986	1.77%	135	3.20%	186
diabetes	4.63%	343	-13.64%	468	3.60%	390

Table 4: AUC improvement and runtime comparison with fully automated methods using SVM

behavior of the automated methods (FeatureTools showed decrease in performance at times). Sections 5.3, 5.5 present these results.

- **How did our proposed algorithms compare?** We observed that we attained higher qualitative performance, while being scalable compared to other baselines. The exact algorithm `ExBKT` produced optimal buckets, while `RandomizedBKT` was much faster and produced results comparable to `ExBKT`. `Greedy` was inferior in quality. We focused on the coverage variant of `RandomizedBKT` so as to provide a diverse set of buckets

Data	EK-RF	Time (s)	FT-RF	Time (s)	MI-RF (s)	Time
pollen	10.56%	10233	-3.71%	40	-1.00%	42
delta_elevators	0.75%	37611	-1.13%	106	0.70%	180
mammography	-0.01%	16502	5.34%	113	4.00%	191
space	4.09%	10676	4.92%	106	4.20%	186
diabetes	4.46%	3474	3.11%	462	3.30%	390

Table 5: AUC improvement and runtime comparison with fully automated methods using RandomForest

to the human analyst. Algorithm `GSnippet` was interactive to the human analyst and works in real time. Section 5.4 presents these results.

- **What kind of attributes did our proposed framework generate?** We observed that the attributes produced by our framework were intuitive and meaningful to human analyst. Section 5.5 presents some of these results.

5.3 Comparison with fully automated methods

We compared the performance of our proposed approach against automated systems in both classifier performance and efficiency. Tables 4 and 5 show the results for SVM and RF classifiers respectively. On an average, the AUC improvement of the classifiers were more than 14% through attribute design. Our proposed framework was almost 10x-20x faster than ExploreKit, comparable with FeatureTools (but FeatureTools caused sudden decline in AUC at times with the derived attributes).

5.4 Analysis of presented algorithms

In this section, we present quality and running time study of our algorithms with the baselines, described in Section 5.1. We note that the implemented baselines were inferior in AUC improvements, hence we only present running time of our solutions.

Comparison with Baselines. We began by comparing our algorithms for bucket construction : an exact algorithm `ExBKT`, a random walk based algorithm handling coverage `RandomizedBKT` and `Greedy`. Figures 3 and 4 show the results. As expected, `ExBKT` took substantial amount of time but gave the best results. `RandomizedBKT` was much faster and provided almost identical results for AUC improvement. `Greedy` was very efficient but qualitatively inferior.

Varying Number of Operators in the Algebra. We began by supporting arithmetic operators ($+, -, \times, /, \%$) and then systematically added more logical, relational and aggregate operators. As expected, increasing the number of operators causes a slow down in our approach. Recall that our first stage of bucket generation was agnostic to both classifier and grammar. The resulting buckets often contained only a handful of attributes and hence our algorithm was quite fast even for a large number of operators. This can be seen in Figure 9 where the improvement is (sub)-linear in the number of operators. Figure 6 shows the corresponding impact on AUC. This shows that the additional operators often only provides negligible improvement in AUC. We conjuncture that for most attributes, only a small set of operators are most relevant.

Varying Bucket Size x. Next, we varied the maximum number of attributes x in each bucket. The impact of k (the number of buckets to return) was minimal. Our algorithms have a natural anytime property where it can be stopped at any time and pick the best k buckets. x affected the length of the random walk and had a major impact on runtime and AUC. Figures 7 and 10 show the result. As expected, increasing bucket size improved the AUC but it stagnated quickly. This confirmed with our hypothesis that most derived attributes often consist of few base attributes. The bucket size must not be too small - otherwise, we might miss meaningful group of attributes. It must also not be too large - otherwise, we might expend runtime for no meaningful improvement of AUC. We found a value of 5 to be good both from runtime perspective and the required cognitive impact on the human analyst. The runtime increased almost linearly with larger bucket size.

Varying Snippet Size j. In our final set of experiments, we varied the snippet size. Here we also noted that the impact of l (the number of snippets to return) was minimal. Figures 8 and 11 show the results. As expected, increasing snippet size had minimal impact on AUC. The improvement more or less topped out at snippet of size 3. This once again confirmed our hypothesis that derived attributes were often constructed from a handful of base attributes. The increase in time for larger snippet size was mostly linear.

Varying Number of Attributes. In order to highlight the scalability of our algorithms, we varied the number of base attributes by duplicating some of the base attributes randomly. Figure 5 shows the impact on running time of our exact and random walk based algorithm. As expected, the running time increased dramatically for `ExBKT` while the increase was marginal for `RandomizedBKT`. Our approach was scalable to large increase in the number of columns. The impact of increasing number of rows were rather minimal.

5.5 Comparison with fully manual method

We present a case study comparing a fully manual approach (a typical trial-and-error based attribute design exercise a data scientists goes through) and our approach. We used the popular Boston dataset from Kaggle [8] that seeks to predict house prices from from features like its area, number of bedrooms, etc.

We involved a data scientist to construct the derived attributes and she came up with the following 6 additional derived attributes after extensive analysis. lstat / tax, ptratio / tax, dis * rad / tax, black * crim, rm * (zn + indus) and (nox * indus) / tax. The interpretation of these attributes are presented in Table 6. The data scientist took about 10 hours to manually craft these 6 derived attributes, out of which 3 hours were needed to understand and explore the data. The remaining 7 hours were needed for a trial and error process, where the expert tried many possible derived attributes, analyzed the correlation, tweaked the attributes, and repeated the process. While another data scientist with similar expertise was guided by

[8] https://www.kaggle.com/c/boston-housing/data

Attribute	Interpretation
lstat	lower status of the population (percent).
tax	full-value property-tax rate per $10,000.
ptratio	pupil-teacher ratio by town.
dis	weighted mean of distances to five Boston employment centres.
rad	index of accessibility to radial highways.
black	$1000(Bk - 0.63)^2$ where Bk is the proportion of blacks by town.
crim	per capita crime rate by town.
rm	average number of rooms per dwelling.
zn	proportion of non-retail business acres per town.
nox	nitrogen oxides concentration (parts per 10 million).

Table 6: Boston home dataset attributes

our framework, she took *only* 1 *hour* to craft the same 6 derived attributes. For both cases, we observed an AUC improvement of 14%. This case study anecdotally showcase that our proposed framework is capable to drastically reduce the latency (7 hour vs 1 hr, i.e, $7x$ improvement) by aiding the domain expert.

6 RELATED WORK

Human-in-the-loop Query Answering : A growing number of systems make use of lay workers (known as crowdsourced workers) using commercial platforms (e.g., AMT and CrowdFlower) or for academic use. Examples of applications include not only sentence translation, photo tagging and sentiment analysis, but also query answering (CrowdDB [11], Qurk [28], Deco [32], sCOOP, FusionCOMP, MoDaS, CyLog/Crowd4U), entity resolution (such as CrowdER [40]), planning queries [19], perform matching [41], or counting [27]. A series of works [39–43] have proposed variations of crowdsourced join to address entity resolution task for query answering. Authors in [7, 9, 13, 21, 33, 47] have proposed various techniques of performing crowdsourced version of top-k item selection. Crowdsourcing based filtering has been extensively introduced in [30, 31]. Crowdsourced *find* focuses on selecting one or more qualified items [34, 44]. Unlike these works, we involve humans for attributes design, which occurs at a later stage of the data science pipeline — therefore these prior works do not extend to our problem.

Human-in-the-loop Supervised Modeling: Machine learning literature has involved humans to obtain labels [5, 15, 16, 37] primarily for the classification problem. Active learning involving humans (expert as well as lay workers) has also been discussed in recent works such as [10, 29, 45, 49]. These works utilize humans for accurately predicting class labels, thus the humans function as an added module to help the algorithm steer toward the correct assignment of class labels. The main distinction between these works and ours is that, these works leverage human mainly for data labeling and not for attribute design.

Attribute Design: How to develop automated methods for designing/engineering attributes (commonly referred to as as feature engineering) has been discussed in recent works [2, 18, 20, 22, 23, 48], primarily in machine learning literature. Attribute engineering tools, such as, ExploreKit, Featuretools, Data Science Machine, One Button Machine [17, 18, 20, 23, 24, 35] rely on fully automated approaches. Typically they take longer to run and do not offer adequate explainability or intuition, as the discovered attributes remain opaque to the human analyst interested in broader ad-hoc data exploration. They also ignore the availability of humans to guide their exploration. Finally, the effectiveness of most of these tools depend on the underlying classification model, whereas, we present a model agnostic approach. Nevertheless, we use some of these tools in our experimental analysis for comparison purposes. A very few recent works [8, 36, 50] propose a conceptual framework and describe how to involve humans for designing attributes to improve the accuracy of predictive machine learning models, such as classifiers. While we borrow inspiration from these recent works [8, 36, 50], we present the framework with mathematical rigor and investigate optimization opportunities.

7 CONCLUSION

We present an optimization guided semi-automated model-agnostic "human-in-the-loop" framework for designing derived attributes by leveraging humans. The main contribution of our work is to provide an optimization guided framework for data scientists that will help them to craft meaningful derived attributes much quicker than a fully manual method. On the other hand, running time of the fully automated techniques vary greatly; for example, FeatureTools is significantly faster compared to ExploreKit, but they all have the limitations of producing derived attributes that are opaque. Contrarily, the derived attributes produced by our framework are interpretable.

We present two computational problems inside our proposed framework - *top-k buckets design* and *top-l snippets generation* problems. We present effective solutions for solving both problems. We compare our proposed approach with two fully automated state-of-the-art tools, as well as fully manual domain expert designed derived attributes. Our rigorous quality evaluations using 6 real world datasets demonstrate that we are as effective as *fully automated methods* and we scale up significantly better compared to a fully manual solution involving two domain experts. Our scalability results demonstrate that both bucket design and snippet generation are interactive and ensure real time response with the analysts. As an ongoing problem, we are investigating how to revisit these problems when the datasets contain significant missing values. We are also investigating how to involve multiple domain experts in buckets design and snippets generation problems.

ACKNOWLEDGEMENTS

The work of Senjuti Basu Roy is supported by the National Science Foundation under Grant No.: 1814595 and Office of Naval Research under Grant No.: N000141812838. The work of Gautam Das is supported in part by grant W911NF-15-1-0020 from the Army Research Office, grant 1745925 from the National Science Foundation, and a grant from AT&T.

REFERENCES

[1] Rakesh Agarwal, Ramakrishnan Srikant, et al. 1994. Fast algorithms for mining association rules. In *Proc. of the 20th VLDB Conference*. 487–499.

[2] Michael R Anderson, Dolan Antenucci, Victor Bittorf, Matthew Burgess, Michael J Cafarella, Arun Kumar, Feng Niu, Yongjoo Park, Christopher Ré, and Ce Zhang. 2013. Brainwash: A Data System for Feature Engineering.. In *CIDR*.

[3] Michael R Anderson and Michael Cafarella. 2016. Input selection for fast feature engineering. In *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*. IEEE, 577–588.

[4] Senjuti Basu Roy, Ankur Teredesai, Kiyana Zolfaghar, Rui Liu, David Hazel, Stacey Newman, and Albert Marinez. 2015. Dynamic hierarchical classification for patient risk-of-readmission. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1691–1700.

[5] Jonathan Bragg, Daniel S Weld, et al. 2013. Crowdsourcing multi-label classification for taxonomy creation. In *AAAI Conference on Human Computation and Crowdsourcing*.

[6] Sergey Brin, Rajeev Motwani, and Craig Silverstein. 1997. Beyond market baskets: Generalizing association rules to correlations. In *Acm Sigmod Record*, Vol. 26. ACM, 265–276.

[7] Xi Chen, Paul N Bennett, Kevyn Collins-Thompson, and Eric Horvitz. 2013. Pairwise ranking aggregation in a crowdsourced setting. In *ACM International Conference on Web Search and Data Mining*. ACM, 193–202.

[8] Justin Cheng and Michael S Bernstein. 2015. Flock: Hybrid crowd-machine learning classifiers. In *ACM Conference on Computer Supported Cooperative Work & Social Computing*. ACM, 600–611.

[9] Brian Eriksson. 2013. Learning to top-k search using pairwise comparisons. In *Artificial Intelligence and Statistics*. 265–273.

[10] Meng Fang, Jie Yin, and Dacheng Tao. 2014. Active Learning for Crowdsourcing Using Knowledge Transfer.. In *AAAI*. 1809–1815.

[11] Amber Feng, Michael J. Franklin, Donald Kossmann, Tim Kraska, Samuel Madden, Sukriti Ramesh, Andrew Wang, and Reynold Xin. 2011. CrowdDB: Query Processing with the Crowd. *PVLDB* 4, 12 (2011), 1387–1390.

[12] Benoît Frénay, Gauthier Doquire, and Michel Verleysen. 2013. Is mutual information adequate for feature selection in regression? *Neural Networks* 48 (2013), 1–7.

[13] Stephen Guo, Aditya Parameswaran, and Hector Garcia-Molina. 2012. So who won?: dynamic max discovery with the crowd. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. ACM, 385–396.

[14] Jeff Heaton. 2016. An empirical analysis of feature engineering for predictive modeling. In *SoutheastCon, 2016*. IEEE, 1–6.

[15] Chien-Ju Ho, Shahin Jabbari, and Jennifer W Vaughan. 2013. Adaptive task assignment for crowdsourced classification. In *International Conference on Machine Learning*. 534–542.

[16] Muhammad Imran, Carlos Castillo, Ji Lucas, Patrick Meier, and Sarah Vieweg. 2014. AIDR: Artificial intelligence for disaster response. In *International Conference on World Wide Web*. ACM, 159–162.

[17] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional architecture for fast feature embedding. In *ACM International Conference on Multimedia*. ACM, 675–678.

[18] James Max Kanter and Kalyan Veeramachaneni. 2015. Deep feature synthesis: Towards automating data science endeavors. In *IEEE International Conference on Data Science and Advanced Analytics*. IEEE, 1–10.

[19] Haim Kaplan, Ilia Lotosh, Tova Milo, and Slava Novgorodov. 2013. Answering Planning Queries with the Crowd. In *PVDLB*.

[20] Gilad Katz, Eui Chul Richard Shin, and Dawn Song. 2016. Explorekit: Automatic feature generation and selection. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 979–984.

[21] Asif R Khan and Hector Garcia-Molina. 2014. *Hybrid strategies for finding the max with the crowd: technical report.* Technical Report. Stanford InfoLab.

[22] Udayan Khurana, Deepak Turaga, Horst Samulowitz, and Srinivasan Parthasrathy. 2016. Cognito: Automated feature engineering for supervised learning. In *IEEE International Conference on Data Mining Workshops*. IEEE, 1304–1307.

[23] Hoang Thanh Lam, Johann-Michael Thiebaut, Mathieu Sinn, Bei Chen, Tiep Mai, and Oznur Alkan. 2017. One button machine for automating feature engineering in relational databases. *arXiv preprint arXiv:1706.00327* (2017).

[24] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436.

[25] Wentian Li. 1990. Mutual information functions versus correlation functions. *Journal of statistical physics* 60, 5-6 (1990), 823–837.

[26] Mokshay Madiman. 2008. On the entropy of sums. In *Information Theory Workshop, 2008. ITW'08. IEEE*. IEEE, 303–307.

[27] Adam Marcus, David Karger, Samuel Madden, Robert Miller, and Sewoong Oh. 2012. Counting with the crowd. *Proceedings of the VLDB Endowment* 6, 2, 109–120.

[28] Adam Marcus, Eugene Wu, David Karger, Samuel Madden, and Robert Miller. 2011. Human-powered sorts and joins. *Proceedings of the VLDB Endowment* 5, 1 (2011), 13–24.

[29] Barzan Mozafari, Purna Sarkar, Michael Franklin, Michael Jordan, and Samuel Madden. 2014. Scaling up crowd-sourcing to very large datasets: a case for active learning. *Proceedings of the VLDB Endowment* 8, 2 (2014), 125–136.

[30] Aditya Parameswaran, Stephen Boyd, Hector Garcia-Molina, Ashish Gupta, Neoklis Polyzotis, and Jennifer Widom. 2014. Optimal crowd-powered rating and filtering algorithms. *Proceedings of the VLDB Endowment* 7, 9 (2014), 685–696.

[31] Aditya G Parameswaran, Hector Garcia-Molina, Hyunjung Park, Neoklis Polyzotis, Aditya Ramesh, and Jennifer Widom. 2012. Crowdscreen: Algorithms for filtering data with humans. In *ACM SIGMOD International Conference on Management of Data*. ACM, 361–372.

[32] Hyunjung Park and Jennifer Widom. 2013. Query optimization over crowdsourced data. *Proceedings of the VLDB Endowment* 6, 10, 781–792.

[33] Thomas Pfeiffer, Xi Alice Gao, Yiling Chen, Andrew Mao, and David G Rand. 2012. Adaptive Polling for Information Aggregation.. In *AAAI*.

[34] Anish Das Sarma, Aditya Parameswaran, Hector Garcia-Molina, and Alon Halevy. 2014. Crowd-powered find algorithms. In *IEEE International Conference on Data Engineering*. IEEE, 964–975.

[35] Frank Seide, Gang Li, Xie Chen, and Dong Yu. 2011. Feature engineering in context-dependent deep neural networks for conversational speech transcription. In *IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, 24–29.

[36] Micah J Smith, Roy Wedge, and Kalyan Veeramachaneni. 2017. FeatureHub: Towards collaborative data science. In *IEEE International Conference on Data Science and Advanced Analytics*. IEEE, 590–600.

[37] Chong Sun, Narasimhan Rampalli, Frank Yang, and AnHai Doan. 2014. Chimera: Large-scale classification using machine learning, rules, and crowdsourcing. *Proceedings of the VLDB Endowment* 7, 13 (2014), 1529–1540.

[38] Jorge R Vergara and Pablo A Estévez. 2014. A review of feature selection methods based on mutual information. *Neural computing and applications* 24, 1 (2014), 175–186.

[39] Norases Vesdapunt, Kedar Bellare, and Nilesh Dalvi. 2014. Crowdsourcing algorithms for entity resolution. *Proceedings of the VLDB Endowment* 7, 12 (2014), 1071–1082.

[40] Jiannan Wang, Tim Kraska, Michael J Franklin, and Jianhua Feng. 2012. Crowder: Crowdsourcing entity resolution. *Proceedings of the VLDB Endowment* 5, 11 (2012), 1483–1494.

[41] Jiannan Wang, Guoliang Li, Tim Kraska, Michael J Franklin, and Jianhua Feng. 2013. Leveraging transitive relations for crowdsourced joins. In *ACM SIGMOD International Conference on Management of Data*. ACM, 229–240.

[42] Sibo Wang, Xiaokui Xiao, and Chun-Hee Lee. 2015. Crowd-based deduplication: An adaptive approach. In *ACM SIGMOD International Conference on Management of Data*. ACM, 1263–1277.

[43] Steven Euijong Whang, Peter Lofgren, and Hector Garcia-Molina. 2013. Question selection for crowd entity resolution. *Proceedings of the VLDB Endowment* 6, 6 (2013), 349–360.

[44] Tingxin Yan, Vikas Kumar, and Deepak Ganesan. 2010. Crowdsearch: exploiting crowds for accurate real-time image search on mobile phones. In *International Conference on Mobile Systems, Applications and Services*. ACM, 77–90.

[45] Yan Yan, Romer Rosales, Glenn Fung, and Jennifer G Dy. 2011. Active learning from crowds.. In *International Conference on Machine Learning*, Vol. 11. 1161–1168.

[46] Guizhen Yang. 2004. The complexity of mining maximal frequent itemsets and maximal frequent patterns. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 344–353.

[47] Peng Ye and David Doermann. 2013. Combining preference and absolute judgements in a crowd-sourced setting. In *International Conference on Machine Learning*. 1–7.

[48] Ce Zhang, Arun Kumar, and Christopher Ré. 2016. Materialization optimizations for feature selection workloads. *ACM Transactions on Database Systems* 41, 1 (2016), 2.

[49] Jinhong Zhong, Ke Tang, and Zhi-Hua Zhou. 2015. Active Learning from Crowds with Unsure Option.. In *IJCAI*. 1061–1068.

[50] James Y Zou, Kamalika Chaudhuri, and Adam Tauman Kalai. 2015. Crowdsourcing feature discovery via adaptively chosen comparisons. *arXiv preprint arXiv:1504.00064* (2015).

How Representative Is a SPARQL Benchmark? An Analysis of RDF Triplestore Benchmarks

Muhammad Saleem
AKSW, Leipzig University
saleem@informatik.uni-leipzig.de

Gábor Szárnyas*
MTA-BME Lendület Cyber-Physical Systems Research Group
szarnyas@mit.bme.hu

Felix Conrads
DICE, Paderborn University
felix.conrads@upb.de

Syed Ahmad Chan Bukhari
Department of Pathology, Yale School of Medicine
ahmad.chan@yale.edu

Qaiser Mehmood
Insight Centre for Data Analytics, University of Ireland, Galway
qaiser.mehmood@insight-centre.org

Axel-Cyrille Ngonga Ngomo
DICE, Paderborn University
AKSW, Leipzig University
axel.ngonga@upb.de

ABSTRACT

Triplestores are data management systems for storing and querying RDF data. Over recent years, various benchmarks have been proposed to assess the performance of triplestores across different performance measures. However, choosing the most suitable benchmark for evaluating triplestores in practical settings is not a trivial task. This is because triplestores experience varying workloads when deployed in real applications. We address the problem of determining an appropriate benchmark for a given real-life workload by providing a fine-grained comparative analysis of existing triplestore benchmarks. In particular, we analyze the data and queries provided with the existing triplestore benchmarks in addition to several real-world datasets. Furthermore, we measure the correlation between the query execution time and various SPARQL query features and rank those features based on their significance levels. Our experiments reveal several interesting insights about the design of such benchmarks. With this fine-grained evaluation, we aim to support the design and implementation of more diverse benchmarks. Application developers can use our result to analyze their data and queries and choose a data management system.

ACM Reference Format:
Muhammad Saleem, Gábor Szárnyas, Felix Conrads, Syed Ahmad Chan Bukhari, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo. 2019. How Representative Is a SPARQL Benchmark? An Analysis of RDF Triplestore Benchmarks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313556

1 INTRODUCTION

The last years have witnessed a significant growth in the use of Linked Data and Semantic Web technologies. This growth has motivated the development of new triplestores with increasingly more efficient RDF storage and SPARQL query processing mechanisms. Accordingly, various benchmarks [2, 5, 8, 11, 14, 16, 18, 24, 27, 30, 31, 33] have been proposed to evaluate the querying performance of these triplestores. However, due to heterogeneity of RDF datasets and SPARQL queries, real-world applications often require customized deployments and experience different workloads when deployed in real environments [22]. Thus, designing a one-fit-all benchmark or selecting the most suitable benchmark for any given use-case and workload is not a straightforward task [12, 21, 24].

This work highlights key features of triplestore benchmarks pertaining to the three main components of benchmarks, i.e., datasets, queries, and performance metrics. State-of-the-art triplestore benchmarks are analyzed and compared against these features. Particularly, we consider triplestore benchmarks that rely on the native capabilities of the triplestores and do not require further reasoning over queries to get complete results. We also analyze the data and query logs of five real-world datasets selected from three different domains so as to provide a comparison with real-world datasets and queries. Our contributions are as follows.

(1) We identify key design features of SPARQL triplestore benchmarks based on our systematic survey on the state-of-the-art.
(2) We provide a detailed comparative analysis of the queries and datasets of 11 representative triplestore querying benchmarks: Train Benchmark [30], FEASIBLE [24], WatDiv [2], DBpedia SPARQL Benchmark (DBPSB) [18], FishMark [5], Bowlogna [11], SP2Bench [11], Berlin SPARQL Benchmark (BSBM) [8], BioBench [33], LDBC Social Network Benchmark Business Intelligence workload (SNB-BI) [31], and LDBC SNB Interactive workload (SNB-INT) [14].
(3) We analyze real data and corresponding user queries of five real-world datasets – DBpedia,[1] Semantic Web Dog Food (SWDF),[2] NCBI Gene,[3] SIDER,[4] DrugBank[5] – and compare them with the selected triplestore benchmarks.
(4) We measure the impact of various SPARQL query features (e.g., result sizes, triple patterns selectivity and number of join vertices) on the overall query execution time and rank these features according to their significance. In addition,

*Also with the Department of Measurement and Information Systems at the Budapest University of Technology and Economics.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313556

[1] DBpedia: http://dbpedia.org
[2] SWDF: https://old.datahub.io/dataset/semantic-web-dog-food
[3] NCBI Gene: https://www.ncbi.nlm.nih.gov/gene
[4] SIDER: http://sideeffects.embl.de/
[5] DrugBank: https://www.drugbank.ca/

we demonstrate that state-of-the-art triplestore benchmarks vary greatly, and highlight their current limitations.
(5) We performed extensive experiments and measure the impact of dataset *structuredness* (a well-known RDF dataset metric [12] formally defined in Section 2.1.1) on the overall query execution time as well as on the result sizes.

The rest of this paper is organized as follows: We provide an overview of the key RDF datasets and SPARQL query features that need to be considered while designing triplestore benchmarks based on a review of the state of the art. We then present a systematic survey of the current benchmarks for triplestores. The subsequent comparison of selected representative SPARQL benchmarks based on key data and query features identified in the survey is followed by a discussion of our results and some concluding remarks. The data and results presented in this evaluation are available online at https://github.com/dice-group/triplestore-benchmarks. The complete results can be reproduced and new benchmarks can be easily compared using the scripts provided at the project page.

2 BENCHMARK DESIGN FEATURES

In general, triplestore benchmarks comprise three main components: (1) a set of RDF datasets, (2) a set of SPARQL queries, and (3) a set of performance metrics. This section presents key features of each of these components that are important to consider in the development of triplestore benchmarks. Most of these features originate from state-of-the-art research contributions pertaining to triplestore benchmarks.

2.1 Datasets

Datasets used in triplestore benchmarks are either synthetic or selected from real-world RDF datasets [24]. The use of real-world RDF datasets is often regarded as useful to perform evaluation close to real-world settings [18]. Synthetic datasets are useful to test the scalability of systems based on datasets of varying sizes. Synthetic dataset generators are utilized to produce datasets of varying sizes which can often be optimized to reflect the characteristics of real-world datasets [12]. Previous works [12, 20] highlighted two key measures for selecting such datasets for triplestores benchmarking: (1) Dataset Structuredness, (2) Relationship Specialty. However, observations from the literature (see e.g., [12, 23]) suggest that other features such as varying number of triples, number of resources, number of properties, number of objects, number of classes, diversity in literal values, average properties and instances per class, average indegrees and outdegrees as well as their distribution across resources should also be considered.

2.1.1 Dataset Structuredness: Duan et al. [12] combine many of the aforementioned dataset features into a single composite metric called dataset *structuredness* or *coherence*. This metric measures how well a dataset's classes (i.e., `rdf:type`) are covered by the different instances of the dataset. The structuredness value for any given dataset lies between $[0, 1]$, where 0 stands for lowest possible structure and 1 points to a highest possible structured dataset. They conclude that synthetic datasets are highly structured while real-world datasets have structuredness values ranging from low to high, covering the whole structuredness spectrum. Formally, dataset structuredness is defined in the form of class coverage. The coverage of a class C, denoted by $CV(C)$, is defined as follows [12]:

Definition 2.1 (Class Coverage). Let D be a dataset. Moreover, let $P(C)$ denote the set of distinct properties of class C, and $I(C)$ denote the set of distinct instances of the class C. Let $I(p, C)$ count the number of entities for which property p has its value set in the instances of C. Then, the coverage of the class $CV(C)$ is

$$CV(C) = \frac{\sum_{p \in P(C)} I(p, C)}{|P(C)| \cdot |I(C)|}$$

In general, RDF datasets comprise multiple classes with a varying number of instances for different classes. The authors of [12] proposed a mechanism that considers the weighted sum of the coverage $CV(C)$ of individual classes. For each class C, the weighted coverage is defined below.

Definition 2.2 (Weighted Class Coverage). The weighted coverage for a class C denoted by $WTCV(C)$ is calculated as:

$$WTCV(C) = \frac{|P(C)| + |I(C)|}{\sum_{C' \in D} |P(C')| + |I(C')|}$$

By using Definitions 2.1 and 2.2, we are now ready to compute the structuredness of a dataset D.

Definition 2.3 (Dataset Structuredness). The overall structuredness or coherence of a dataset D denoted by $CH(D)$ is defined as

$$CH(D) = \sum_{C \in D} CV(C) \cdot WTCV(C)$$

2.1.2 Relationship Specialty. In datasets, some attributes are more common and associated with many resources. In addition, some attributes are multi-valued, e.g., a person can have more than one cellphone number or professional skill. The number of occurrences of a predicate associated with each resource in the dataset provides useful information on the graph structure of an RDF dataset, and makes some resources distinguishable from others [20]. In real datasets, this kind of relationship specialty is commonplace. For example, several million people can like the same movie. Likewise, a research paper can be cited in several hundred of other publications. Qiao et al. [20] suggest that synthetic datasets are limited in how they reflect this relationship specialty. This is either due to the simulation of uniform relationship patterns for all resources, or a random relationship generation process. The relationship specialty of a *relationship predicate* is defined as follows:

Definition 2.4 (Predicate Relationship Specialty). Let d be the distribution that records the number of occurrences of a *relationship predicate* r associated with each resource and μ is the mean and σ is the standard deviation of d. The specialty value of r denoted as $\kappa(r)$ is defined as the Pearson's Kurtosis value of the distribution d.

$$\kappa(r) = \frac{n(n+1)}{(n-1)(n-2)(n-3)} \cdot \frac{\sum_{x_i \in d}(x_i - \mu)^4}{\sigma^4} - \frac{3(n-1)^2}{(n-2)(n-3)}$$

Where n is the number of available values, i.e., sample size. The relationship specialty of a dataset is defined in the form of a weighted sum of specialty values of all relationship predicates:

Definition 2.5 (Dataset Relationship Specialty). The relationship specialty of dataset D denoted by $RS(D)$ is calculated as follows:

$$RS(D) = \sum_{r_i \in R} \frac{|T(r_i)| \cdot \kappa(r_i)}{\sum_{r_j \in R} |T(r_j)|}$$

where $|T(r_i)|$ is the number of triples in the dataset having predicate r_i, $\kappa(r_i)$ is the specialty value of relationship predicate r_i.

The dataset structuredness and relationship specialty directly affect the result size, the number of intermediate results, and the selectivities of the triple patterns of the given SPARQL query. Therefore, they are important dataset design features to be considered during the generation of benchmarks [12, 20, 24].

2.2 SPARQL Queries

The literature about SPARQL Queries [2, 15, 23, 24, 26] suggests that a SPARQL querying benchmark should vary the queries with respect to various features such as *query characteristics*: number of triple patterns, number of projection variables, result set sizes, query execution time, number of BGPs, number of join vertices, mean join vertex degree, mean triple pattern selectivities, BGP-restricted and join-restricted triple pattern selectivities, join vertex types, and highly used SPARQL clauses (e.g., `LIMIT`, `OPTIONAL`, `ORDER BY`, `DISTINCT`, `UNION`, `FILTER`, `REGEX`). All of these features have a direct impact on the runtime performance of triplestores. We assume that the reader is familiar with the basic concepts of SPARQL, including the notions of a triple pattern, a basic graph pattern (BGP), and projection variables.[6] In the following, we define the remaining SPARQL features formally, i.e., the number of join vertices, mean join vertex degree, join vertex types, triple pattern selectivities, BGP-restricted and join-restricted triple pattern selectivities.

We represent any basic graph pattern (BGP) of a given SPARQL query as a *directed hypergraph* (DH) [25], a generalization of a directed graph in which a hyperedge can join any number of vertices. In our specific case, every hyperedge captures a triple pattern. The subject of the triple becomes the source vertex of a hyperedge and the predicate and object of the triple pattern become the target vertices. For instance, the query (Figure 1) shows the hypergraph representation of a SPARQL query. Unlike a common SPARQL representation where the subject and object of the triple pattern are connected by an edge, our hypergraph-based representation contains nodes for all three components of the triple patterns. As a result, we can capture joins that involve predicates of triple patterns. Formally, our hypergraph representation is defined as follows:

Definition 2.6 (Directed hypergraph of a BGP). The hypergraph representation of a BGP B is a directed hypergraph $HG = (V, E)$ whose vertices are all the components of all triple patterns in B, i.e., $V = \bigcup_{(s,p,o) \in B} \{s, p, o\}$, and that contains a hyperedge $(S, T) \in E$ for every triple pattern $(s, p, o) \in B$ such that $S = \{s\}$ and $T = (p, o)$.

The representation of a complete SPARQL query as a DH is the union of the representations of the query's BGPs. Based on the DH representation of SPARQL queries, we can define the following features of SPARQL queries:

[6] See https://www.w3.org/TR/sparql11-query/ for the corresponding definitions.

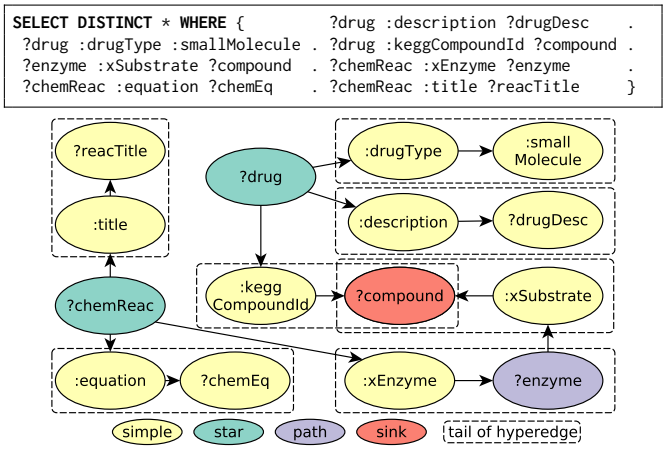


Figure 1: Directed hypergraph representation of a SPARQL query. Prefixes are ignored for simplicity.

Definition 2.7 (Join Vertex). For every vertex $v \in V$ in such a hypergraph we write $E_{\text{in}}(v)$ and $E_{\text{out}}(v)$ to denote the set of incoming and outgoing edges, respectively; i.e., $E_{\text{in}}(v) = \{(S, T) \in E \mid v \in T\}$ and $E_{\text{out}}(v) = \{(S, T) \in E \mid v \in S\}$. If $|E_{\text{in}}(v)| + |E_{\text{out}}(v)| > 1$, we call v a *join vertex*.

Definition 2.8 (Join Vertex Degree). Based on the DH representation of the queries the join vertex degree of a vertex v is $JVD(v) = |E_{in}(v)| + |E_{out}(v)|$, where $E_{in}(v)$ resp. $E_{out}(v)$ is the set of incoming resp. outgoing edges of v.

Definition 2.9 (Join Vertex Types). A vertex $v \in V$ can be of type *star*, *path*, *hybrid*, or *sink* if this vertex participates in at least one join. A *star* vertex has more than one outgoing edge and no incoming edges. A *path* vertex has exactly one incoming and one outgoing edge. A *hybrid* vertex has either more than one incoming and at least one outgoing edge or more than one outgoing and at least one incoming edge. A *sink* vertex has more than one incoming edge and no outgoing edge. A vertex that does not participate in joins is *simple*.

Definition 2.10 (Triple Pattern Selectivity). Let tp_i be a triple pattern of a SPARQL query Q and D be a dataset. Furthermore, let N be the total number of triples in D and $Card(tp_i, D)$ be the cardinality of tp_i w.r.t. D, i.e., total number of triples in D that matches tp_i, then the selectivity of tp_i w.r.t. D denoted by $Sel(tp_i, D) = Card(tp_i, D)/N$.

Definition 2.11 (BGP-Restricted Triple Pattern Selectivity). Consider a Basic Graph Pattern BGP and a triple pattern tp_i belonging to BGP, let $R(tp_i, D)$ be the set of distinct solution mappings (i.e., result set) of executing tp_i over dataset D and $R(BGP, D)$ be the set of distinct solution mappings of executing BGP over dataset D. Then the BGP-restricted triple pattern selectivity denoted by $Sel_{BGP\text{-}Restricted}(tp_i, D)$ is the fraction of distinct solution mappings in $R(tp_i, D)$ that are compatible (as per standard SPARQL semantics [3]) with a solution mapping in $R(BGP, D)$ [2]. Formally, if Ω and Ω' denote the sets underlying the (bag) query results $R(tp_i, D)$

and $R(BGP, D)$, respectively, then

$$Sel_{BGP\text{-}R.}(tp_i, D) = \frac{|\{\mu \in \Omega \,|\exists \mu' \in \Omega' : \mu and \mu' are\ compatible\}|}{|\Omega|}$$

Definition 2.12 (Join-Restricted Triple Pattern Selectivity). Consider a join vertex x in the DH representation of a *BGP*. Let BGP' belonging to *BGP* be the set of triple patterns that are incidents to x. Furthermore, let tp_i belonging to BGP' be a triple pattern and $R(tp_i, D)$ be the set of distinct solution mappings of executing tp_i over dataset D and $R(BGP', D)$ be the set of distinct solution mappings of executing BGP' over dataset D. Then the *x-restricted* triple pattern selectivity denoted by $Sel_{\mathcal{J}Vx\text{-}Restricted}(tp_i, D)$, is the fraction of distinct solution mappings in $R(tp_i, D)$ that are compatible with a solution mapping in $R(BGP', D)$ [2]. Formally, if Ω and Ω' denote the sets underlying the (bag) query results $R(tp_i, D)$ and $R(BGP', D)$, respectively, then

$$Sel_{\mathcal{J}Vx\text{-}R.}(tp_i, D) = \frac{|\{\mu \in \Omega \,|\exists \mu' \in \Omega' : \mu and \mu' are\ compatible\}|}{|\Omega|}$$

All of the above important query features were collected from the previous works [2, 15, 23, 24] in triplestores benchmarking. Finally, we combine all these important query features into a single composite metric called the *Diversity Score* of the benchmark queries, defined as follows.

Definition 2.13 (Queries Diversity Score). Let μ_i be the mean and σ_i the standard deviation of a given distribution w.r.t. the i^{th} feature of the said distribution. The overall diversity score *DS* of the queries is the average coefficient of variation of all the query features k analyzed in the queries of benchmark B:

$$DS = \frac{1}{k} \sum_{i=1}^{k} \frac{\sigma_i(B)}{\mu_i(B)}$$

2.3 Performance Metrics

Based on the previous triplestores benchmarks and performance evaluations [2, 5, 8, 10, 11, 14, 16, 18, 24, 27, 30, 31, 33] the performance metrics for such comparisons can be categorized as:

- **Query Processing Related**: The performance metrics in this category are related to the query processing capabilities of the triplestores. The query execution time is the central performance metric in this category. However, reporting the execution time for individual queries might not be feasible due to the large number of queries in the given benchmark. To this end, Query Mix per Hour (QMpH) and Queries per Second (QpS) are regarded as central performance measures to test the querying capabilities of the triplestores [8, 18, 24]. In addition, the query processing overhead in terms of the CPU and memory usage is important to measure during the query executions [27]. This also includes the number of intermediate results, the number of disk/memory swaps, etc.
- **Data Storage Related:** Triplestores need to load the given RDF data and mostly create indexes before they are ready for query executions. In this regard, the data loading time, the storage space acquired, and the index size are important performance metrics in this category [8, 11, 27, 33].
- **Result Set Related:** Two systems can only be compared if they produce exactly the same results. Therefore, result set correctness and completeness is important metrics to be considered in the triplestores evaluations [8, 24, 27, 33].
- **Parallelism with/without Updates:** Some of the aforementioned triplestores performance evaluations [8, 10, 33] also measured the parallel query processing capabilities of the triplestores by simulating workloads from multiple querying agents with and without dataset updates.

We analyzed state-of-the-art existing SPARQL triplestore benchmarks across all of the above mentioned dataset and query features as well as the performance metrics. The results are presented in Section 4.

3 SYSTEMATIC SURVEY

In this section, we present a systematic survey carried out to collect triplestore benchmarks and their selection criteria for further analysis. We conducted a public survey[7] through various relevant W3C Linked Open Data mailing list[8] and Semantic Web mailing list[9] with a request to participate email. We received 14 responses[10] regarding SPARQL triplestore benchmarks. Moreover, we used Google Scholar to retrieve published research work relating to the design of triplestore benchmarks and/or their performance evaluation. Initially, we selected 40 relevant papers[11] and evaluated them against our designed inclusion criteria. In our inclusion criteria we mandated that (1) the benchmark target the query runtime performance evaluation of triplestores, (2) both RDF data and SPARQL queries of the benchmark are publicly available or can be generated (3) the queries must not require reasoning to retrieve the complete results. After manual evaluation, we found 11 benchmarks (7 with synthetic and 4 with real data) that fulfilled our requirements. The sections below provide details of the selected benchmarks.

3.1 Synthetic Triplestore Benchmarks

The *Train Benchmark* (TrainBench) [30] uses a data generator that produces railway networks in increasing sizes and serializes them in different formats, including RDF. *The Waterloo SPARQL Diversity Test Suite* (WatDiv) [2] provides a synthetic data generator that produces RDF data with a tunable structuredness value and a query generator. The queries are generated from different query templates. *SP2Bench* [27] mirrors vital characteristics (such as power law distributions or Gaussian curves) of the data in the DBLP bibliographic database. *The Berlin SPARQL Benchmark* (BSBM) [8] uses query templates to generate any number of SPARQL queries for benchmarking, covering multiple use cases such as explore, update, and business intelligence. *Bowlogna* [11] models a real-world setting derived from the Bologna process and offers mostly analytic queries reflecting data-intensive user needs. The *LDBC Social Network Benchmark* (SNB) defines two workloads: (1) the *Interactive* workload (SNB-INT) measures the evaluation of graph patterns in a localized scope (e.g., in the neighborhood of a person), with the graph being continuously updated [14], and (2) the *Business Intelligence* workload (SNB-BI) focuses on queries that mix complex

[7] Survey: https://goo.gl/R59uoM
[8] public-lod@w3.org
[9] semantic-web@w3.org
[10] Responses: https://goo.gl/JdivN3
[11] SPARQL benchmarking studies: https://goo.gl/cCu85z

graph pattern matching with aggregations, touching on a significant portion of the graph [31], without any updates. Note that these two workloads are regarded as two separate triplestore benchmarks based on the same dataset.

3.2 Triplestore Benchmarks Using Real Data

FEASIBLE [24] is a cluster-based SPARQL benchmark generator, which is able to synthesize customizable benchmarks from the query logs of SPARQL endpoints. The *DBpedia SPARQL Benchmark* (DBPSB) [18] is another cluster-based approach that generates benchmark queries from DBpedia query logs, but employs different clustering techniques than FEASIBLE. The *FishMark* [5] dataset is obtained from FishBase[12] and provided in both RDF and SQL versions. The SPARQL queries were obtained from logs of web-based FishBase application. *BioBench* [33] evaluates the performance of RDF triplestores with the real biological datasets and queries from five different real-world RDF datasets[13], i.e., Cell, Allie, PDBJ, DDBJ, and UniProt. Due to the size of the datasets, we were only able to analyze the combined data and queries of the first three.

3.3 Selected Real-World Datasets

As mentioned before, we aimed to analyze the data and queries of real-world datasets and compare them to those of the benchmark datasets and queries. The selection criteria for the real-world datasets were: (1) The RDF datasets must be publicly available, (2) the real queries posted by users of the datasets via SPARQL endpoints should be available. We were able to get real log queries of the Bio2RDF datasets,[14] DBpedia, and Semantic Web Dog Food. Our goal was to select real-world datasets from different domains. Hence, we selected DBpedia[15] and SWDF and three datasets – NCBIGene, Sider, DrugBank from Bio2RDF. The selection of the three Bio2RDF datasets was based on a recommendation from domain experts. The well-known DBpedia dataset is the RDF version of Wikipedia. The SWDF represents the publication from Semantic Web and Linked Data as RDF. NCBIGene provides genetic information from a wide range of species. SIDER contains information on marketed medicines and their recorded side-effects. DrugBank knowledge base contains information about drugs, their composition and their interactions.

Table 1 shows statistics from selected datasets of the benchmarks and real-world datasets. More advanced statistics will be presented in the next section. The table also shows the number of SPARQL queries of the datasets included in the corresponding benchmark or query log. It is important to mention that we only selected SPARQL `SELECT` queries for analysis. This is because we wanted to analyze the triplestore benchmarks for their query runtime performance and most of these benchmarks only contain `SELECT` queries [24]. For the synthetic benchmarks that include data generators, we chose the datasets used in the evaluation of the original paper that were comparable in size to the datasets of other synthetic benchmarks. For template-based query generators such as WatDiv, DBPSB, SNB, we chose one query per available template. For FEASIBLE, we generated a benchmark of 50 queries from DBpedia log to be comparable with a well-known WatDiv benchmark that includes 20 basic testing query templates, and 30 extensions for testing.[16]

Table 1: High-level statistics of the data and queries used on our evaluation. Both SNB-BI and SNB-INT use the same dataset and are therefore named as SNB for simplicity.

	Benchmark	Subjects	Predicates	Objects	Triples	Queries
Synthetic	Bowlogna [11]	2,151k	39	260k	12M	16
	TrainB. [30]	3,355k	16	3,357k	41M	11
	BSBM [8]	9,039k	40	14,966k	100M	20
	SP2Bench [27]	7,002k	5,718	19,347k	49M	14
	WatDiv [2]	5,212k	86	9,753k	108M	50
	SNB [14, 31]	7,193k	40	17,544k	46M	21
Real	FishMark [5]	395k	878	1,148k	10M	22
	BioBench [33]	278,007k	299	232,041k	1,451M	39
	FEASIBLE [24]	18,425k	39,672	65,184k	232M	50
	DBPSB [18]	18,425k	39,672	65,184k	232M	25
Datasets	DBpedia3.5.1	18,425k	39,672	65,184k	232M	35,803
	SWDF	36k	185	95k	0.3M	71,406
	NCBIGene	15,614k	57	82,688k	159M	3,644
	SIDER	1,222k	39	5,952k	17M	26,048
	DrugBank	316k	105	1,828k	3M	50,877

4 ANALYSIS OF THE BENCHMARKS

We present a detailed analysis of the datasets, queries, and performance metrics of the selected benchmarks and datasets according to the design features presented in Section 2.

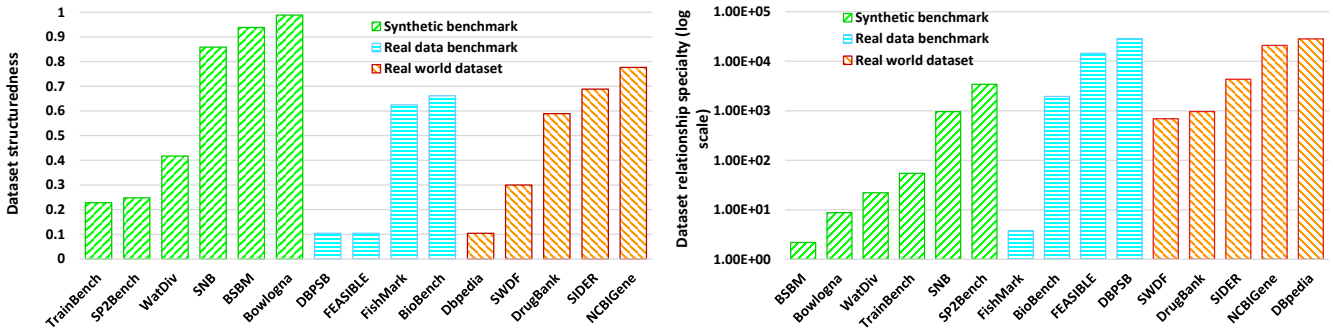
4.1 Datasets

We presents results pertaining to the dataset features of Section 2.1.

4.1.1 Structuredness. Figure 2a shows the structuredness values of the selected benchmarks and real-world datasets. Duan et al. [12] establish that synthetic benchmarks are highly structured while real-world datasets are low structured. This important dataset feature is well-covered in recent synthetic benchmarks such as TrainBench (with a structuredness value of 0.23) and WatDiv, which lets the user generate a benchmark dataset of a desired structuredness value. However, Bowlogna (0.99), BSBM (0.94), and SNB (0.86) have relatively high structuredness values. The average structuredness value of the selected five real-world datasets is 0.49, and 0.65 for the 13 real-world datasets used in LargeRDFBench [23]. Finally, on average, synthetic benchmarks are still more structured than real data benchmarks (0.61 vs. 0.45).

4.1.2 Relationship Specialty. According to [20], relationship specialty in synthetic datasets is limited, i.e., the overall relationship specialty values of synthetic datasets are lower than those of similar real-world datasets. The dataset relationship specialty results presented in Figure 2b mostly confirm this behavior. On average, synthetic benchmarks have a smaller specialty score than real-world datasets (744 vs. 11098). The relationship specialty values of Bowlogna (8.7), BSBM (2.2), and WatDiv (22.0) are on the lower side compared to real-world datasets. The highest specialty value (28282.8) is recorded in the DBpedia dataset.

[12] FishBase: http://fishbase.org/search.php
[13] BioBench: http://kiban.dbcls.jp/togordf/wiki/survey#data
[14] Bio2RDF: http://download.bio2rdf.org/files/release/3/release.html
[15] Version 3.5.1 as used in FEASIBLE and DBPSB
[16] WatDiv query templates: http://dsg.uwaterloo.ca/watdiv/



(a) Structuredness

(b) Relationship Specialty

Figure 2: Analysis of the datasets of triplestore benchmarks and real-world data.

An important issue is the correlation between structuredness and the relationship specialty of the datasets. To this end, we computed the Spearman's correlation between the stucturedness and specialty values of all the selected benchmarks and real-world datasets. The correlation of the two measures is −0.5, indicating a moderate inverse relationship. This means that the higher the structuredness, the lower the specialty value. This is because in highly structured datasets, data is generated according to a specific distribution without treating some predicates more particularly (in terms of occurrences) than others.

4.2 Queries

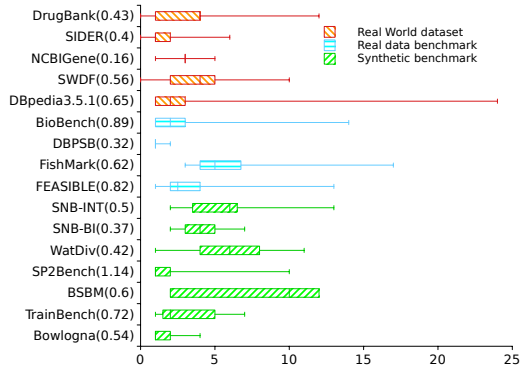
This section presents results pertaining to the query features discussed in Section 2.2. Figure 3 shows the box plot distributions of real-world datasets and benchmark queries across the query features defined in Section 2. The values inside the brackets, e.g., the 0.89 in "BioBench (0.89)", show the diversity score (Definition 2.13) of the benchmark or real-world dataset for the given query feature.

Starting from the number of projection variables (ref. Figure 3a), the NCBIGene dataset has the lowest diversity score (0.16) and SP2Bench has the highest score of 1.14. The mean diversity score (across all benchmarks and real-world datasets) for this feature is 0.59 and hence the diversity scores of DBPSB, SNB-BI, SNB-INT, WatDiv, and Bowlogna are below the average value. Even though the diversity score of BSBM is above average, the distribution shows that the values mostly lie in the second quartile of the box plot. The average diversity score of the number of join vertices (ref. Figure 3b) is 1.39 and hence the diversity scores of the Bowlogna, FishMark, WatDiv, BSBM, TrainBench, BioBench, DBPSB, SNB-BI, and SNB-INT benchmarks are below the average value. It is important to mention that the highest number of join vertices recorded in a query is 51 in the SNB-BI benchmark. The average diversity score of the number of triple patterns (ref. Figure 3c) is 0.75 and hence the diversity scores of the FishMark, Bowlogna, BSBM, and WatDiv benchmarks are below the average value. The average diversity score of the result sizes (ref. Figure 3d) is 11.89 and hence the diversity scores of all benchmarks are below the average value. The average diversity score of the join vertex degree (ref. Figure 3e) is 1.08 and hence the diversity scores of all benchmarks except FEASIBLE are below the average value. The average diversity score of the triple pattern selectivity (ref. Figure 3f) is 3.17 and hence the diversity scores of all benchmarks except FEASIBLE are below the average. The average diversity score of the join-restricted triple pattern selectivity (ref. Figure 3g) is 1.39 and hence the diversity scores of all benchmarks except FEASIBLE and BSBM are below the average value. The average diversity score of the BGP-restricted triple pattern selectivity (ref. Figure 3h) is 4.11 and hence the diversity scores of all benchmarks except WatDiv are below the average value. The average diversity score of the number of BGPs (ref. Figure 3i) is 0.63 and hence the diversity scores of SNB-BI, SNB-INT, BSBM, SP2BENCH, TrainBench, WatDiv, and Bowlogna are below the average value.

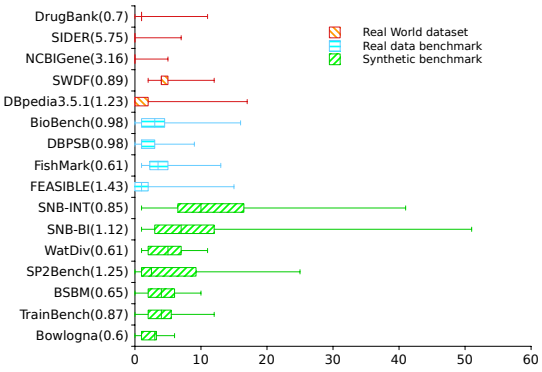
The Linked SPARQL Queries (LSQ) [22] representation stores additional SPARQL features, such as use of `DISTINCT`, `REGEX`, `BIND`, `VALUES`, `HAVING`, `GROUP BY`, `OFFSET`, aggregate functions, `SERVICE`, `OPTIONAL`, `UNION`, property paths, etc. We make a count of all of these SPARQL operators and functions and use it as a single query dimension as number of LSQ features. The average diversity score of the number of LSQ features (ref. Figure 3j) is 0.30, and hence only the diversity scores of SNB and WatDiv are below average value. Finally, the average diversity score of the query runtimes is 12.87 (ref. Figure 3k), and hence the diversity scores of all benchmarks are below average value.

In summary, FEASIBLE's diversity scores are below the average values in 3 of the 11 features, followed by BioBench with 7/11. These are followed by SP2Bench, TrainBench, and BSBM with 8/11 each. The next is FishMark 9/11 and then Bowlogna, WatDiv, SNB-BI, SNB-INT, and DBPSB with 10/11 each. Figure 3l shows the overall (across all the features, ref. Definition 2.13) diversity scores of the benchmarks and real-world datasets. In the benchmarks category, FEASIBLE produces the most diverse benchmarks (diversity score 2.15), followed by BioBench (1.51), FishMark (1.33), WatDiv (1.32), Bowlogna (1.23), SP2Bench (1.22), BSBM (1.08), DBPSB (1.03), SNB-BI (0.90), SNB-INT (0.863) and TrainBench (0.79).

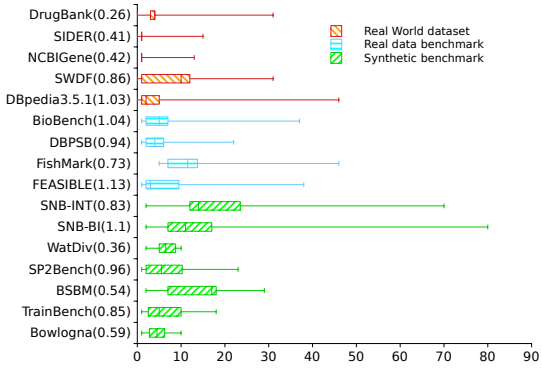
Table 2 shows the percentage coverage of widely used [22] SPARQL clauses and join vertex types for each benchmark and real-world dataset. We highlighted cells for benchmarks that either completely miss or overuse certain SPARQL clauses and join vertex types. TrainBench and WatDiv queries mostly miss the important SPARQL clauses. All of FishMark's queries contain at least one "Star" join node. The distribution of other SPARQL clauses, such as subquery, `BIND`, aggregates, solution modifiers, property paths, and



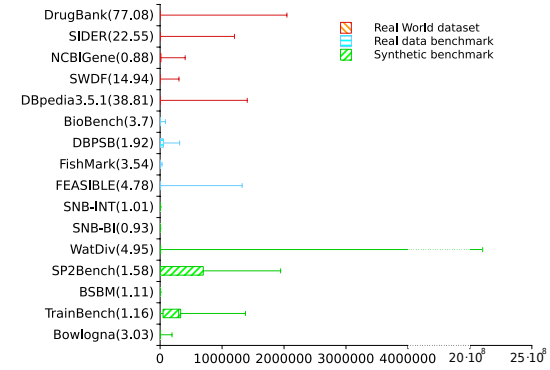
(a) No. of projection variables



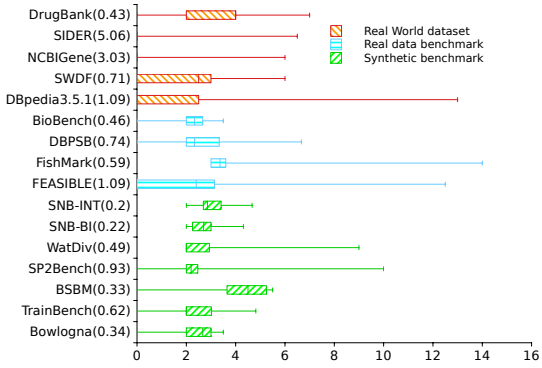
(b) No. of join vertices



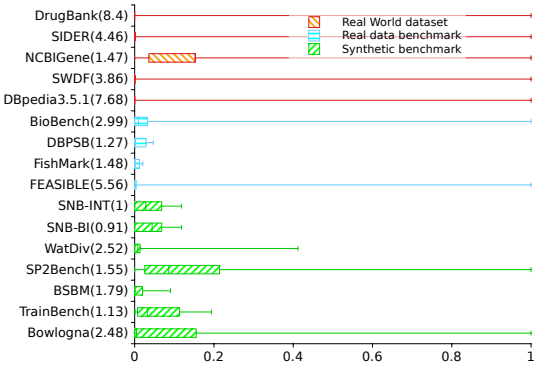
(c) No. of triple patterns



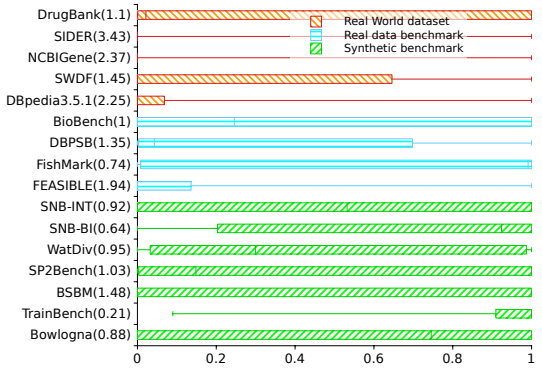
(d) Result size



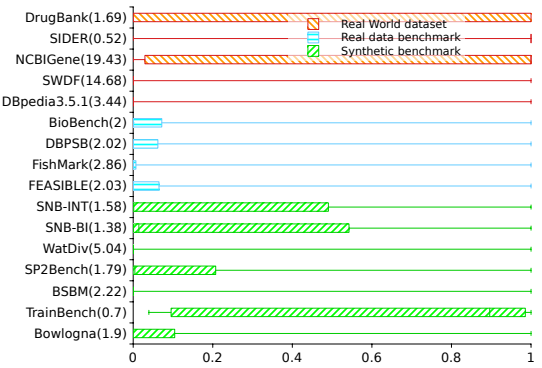
(e) Join vertex degree



(f) Triple pattern (TP) selectivity



(g) Join-restricted TP selectivity



(h) BGP-restricted TP selectivity

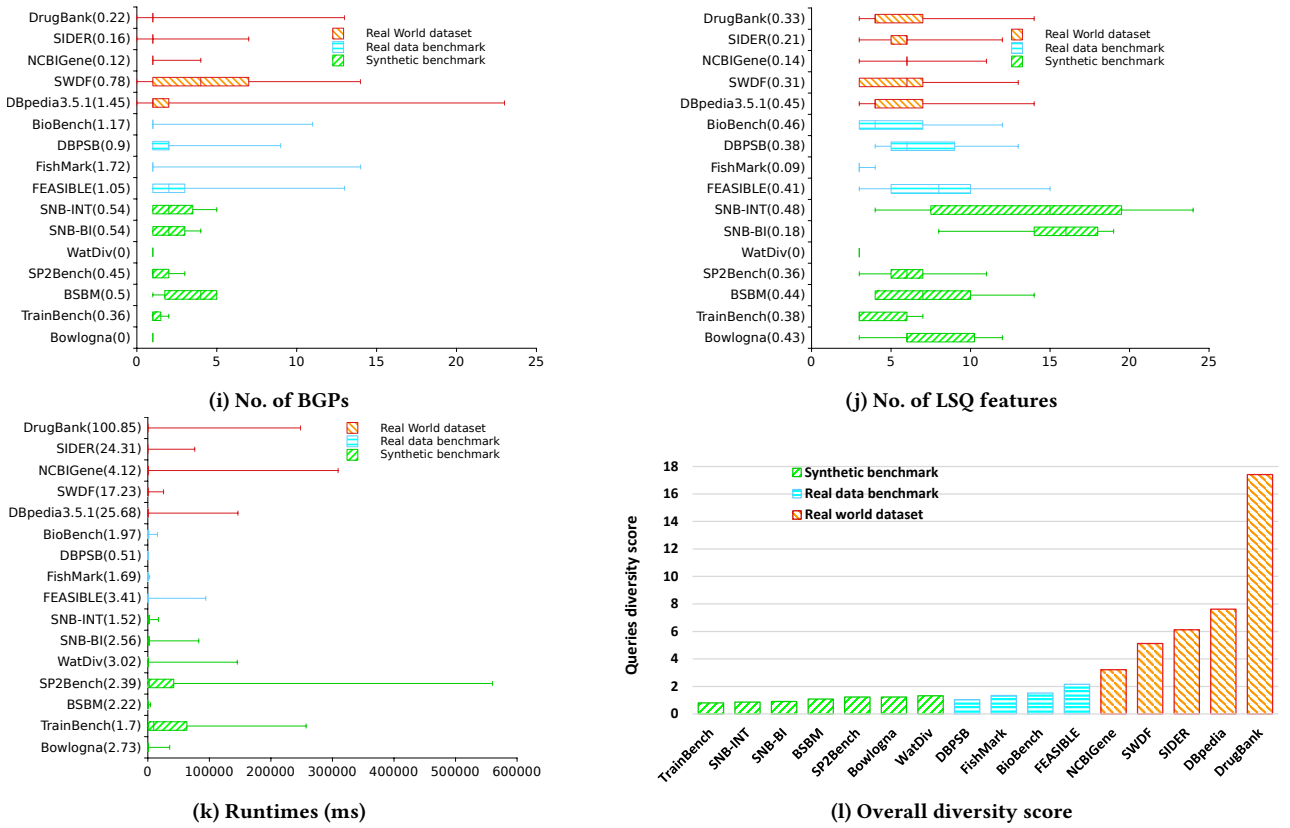


Figure 3: Analysis of queries used in triplestore benchmarks and for real-world datasets.

services are provided in the LSQ versions of each of the benchmark queries, available from the project website.

4.3 Performance Metrics

This section presents results pertaining to the performance metrics discussed in Section 2.3. Table 3 shows the performance metrics used by the selected benchmarks to compare triplestores. The query runtimes for complete benchmark's queries is the central performance metrics and is used by all of the selected benchmarks. In addition, the QpS and QMpH are commonly used in the query processing category. We found that in general, the processing overhead generated by query executions is not paid much attention as only SP2Bench measures this metric. In the "storage" category, the time taken to load the RDF graph into triplestore is most common. The in-memory/HDD space require to store the dataset and corresponding indexes did not get much attention. The result set correctness and completeness are important metrics to be considered when there are large number of queries in the benchmark and composite metrics such as QpS and QMpH are used. We can see many of the benchmarks do not explicitly check these two metrics. They mostly assume that the results are complete and correct. However, this always might not be the case [23]. Additionally, only BSBM considers the evaluation of triplestores with simultaneous user requests with updates. However, benchmarks execution frameworks such IGUANA [10] can be used to measure the parallel query processing

Table 2: Coverage of SPARQL clauses and join vertex types for each benchmark in percentages. SPARQL clauses: `DIST[INCT]`, `FILT[ER]`, `REG[EX]`, `OPT[IONAL]`, `UN[ION]`, `LIM[IT]`, `ORD[ER BY]`. Join vertex types: Star, Path, Sink, Hyb[rid], N[o] J[oin]. Missing ● and overused ● features are highlighted.

		Distributions of SPARQL Clauses							Distr. of Join Vertex Type				
	Benchmark	DIST	FILT	REG	OPT	UN	LIM	ORD	Star	Path	Sink	Hyb.	N.J.
Synthetic	Bowlogna	6.2	37.5	6.2	0.0	0.0	6.2	6.2	93.7	37.5	62.5	25.0	6.2
	TrainB.	0.0	45.4	0.0	0.0	0.0	0.0	0.0	81.8	27.2	72.7	45.4	18.1
	BSBM	30.0	65.0	0.0	65.0	10.0	45.0	45.0	95.0	60.0	75.0	60.0	5.0
	SP2Bench	42.8	57.1	0.0	21.4	14.2	7.1	14.2	78.5	35.7	50.0	28.5	14.2
	Watdiv	0.0	0.0	0.0	0.0	0.0	0.0	0.0	28.0	64.0	26.0	20.0	0.0
	SNB-BI	0.0	61.9	4.7	52.3	14.2	80.9	100.0	90.4	38.1	80.9	52.3	0.0
	SNB-INT	0.0	47.3	0.0	31.5	15.7	63.15	78.9	94.7	42.1	94.7	84.2	0.0
Real	FEASIBLE	56.0	58.0	22.0	28.0	40.0	42.0	32.0	58.0	18.0	36.0	16.0	30.0
	Fishmark	0.0	0.0	0.0	9.0	0.0	0.0	0.0	100.0	81.8	9.0	72.7	0.0
	DBPSB	100.0	48.0	8.0	32.0	36.0	0.0	0.0	68.0	20.0	32.0	20.0	24.0
	BioBench	28.2	25.6	15.3	7.6	7.6	20.5	10.2	71.7	53.8	43.5	38.4	15.3
Datasets	DBpedia	34.7	38.4	8.2	43.0	14.8	2.2	0.1	39.9	10.7	10.5	1.3	52.2
	SWDF	71.6	2.4	1.0	46.5	55.1	2.4	40.2	65.7	21.9	44.3	54.0	28.7
	NCBIGene	4.3	4.9	0.1	0.6	0.2	90.4	0.1	7.4	2.5	0.4	0.0	89.8
	SIDER	79.3	7.8	0.0	0.0	0.2	79.0	0.0	3.4	0.8	0.5	0.1	96.0
	DrugBank	53.7	2.1	0.2	0.0	2.8	14.4	0.0	89.7	23.2	23.4	0.0	10.0
	Overall	33.8	30.3	4.4	20.4	13.0	26.0	16.5	64.7	33.0	37.8	28.9	26.0

capabilities of triplestores in presence of multiple querying and update agents.

Table 3: Metrics used in the selected benchmarks pertaining to query processing, data storage, result set, simultaneous multiple client requests, and dataset updates. *QpS:* Queries per Second, *QMpH:* Queries Mix per Hour, *PO:* Processing Overhead, *LT:* Load Time, *SS:* Storage Space, *IS:* Index Sizes, *RCm:* Result Set Completeness, *RCr:* Result Set Correctness, *MC:* Multiple Clients, *DU:* Dataset Updates.

		Processing			Storage			Result Set		Additional	
	Benchmark	QpS	QMpH	PO	LT	SS	IS	RCm	RCr	MC	DU
Synthetic	Bowlogna	✗	✗	✗	✓	✗	✓	✗	✗	✗	✗
	TrainBench	✗	✗	✗	✓	✗	✗	✓	✓	✗	✓
	BSBM	✓	✓	✗	✓	✗	✗	✓	✓	✓	✓
	SP2Bench	✗	✗	✓	✓	✓	✗	✓	✓	✗	✗
	WatDiv	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
	SNB-BI	✓	✓	✗	✗	✗	✗	✓	✓	✗	✗
	SNB-INT	✓	✓	✗	✗	✗	✗	✓	✓	✗	✓
Real	FEASIBLE	✓	✓	✗	✗	✗	✗	✓	✓	✗	✗
	Fishmark	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗
	DBPSB	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗
	BioBench	✗	✗	✗	✓	✓	✗	✓	✗	✓	✗

4.4 Impact of Dataset Structuredness

The dataset structuredness has been regarded as one of the most important RDF dataset feature [12]. However, to the best of our knowledge, the impact of dataset structuredness on query runtimes and result set sizes has not been measured in the literature. To measure this impact, we need to create synthetic datasets of varying structuredness values, all following the same dataset description model or schema and comparably to each others in terms of their sizes. We then need to execute the same set of benchmark queries over the generated datasets and measure the result set sizes and query runtimes. The WatDiv benchmark data generator allows the control of the size of generated datasets as well as the structuredness values of individual entities. However, the task of generating the datasets with exact sizes and structuredness values are difficult to achieve due to the scaling and structuredness factors used to control the size and structuredness of the overall dataset. We generated 10 datasets of varying structuredness values given in Table 4.

We run complete WatDiv queries on the individual datasets by using Virtuso triplestore and measured the result set sizes and runtimes of individual queries. Figure 4 shows the impact of the dataset structuredness values on query runtimes and result set sizes. We can clearly see there is a positive correlation of dataset sturcturedness values and the query runtimes as-well-as result sizes, i.e. the higher the structuredness value of the dataset the higher the result sizes and query runtimes. The result also suggests that there is a direct correlation with the result sizes and the query runtimes.

4.5 Correlation of Query Features vs. Runtimes

In previous sections, we presented the results of some important SPARQL query features that should be considered while designing SPARQL benchmarks. These features were mostly taken from previous works [2, 18, 24] in SPARQL benchmarking. However, an important question to investigate is how significantly these features impact the query runtimes. To this end, we calculated Spearman's correlation of the said query features with the query runtime as shown in Table 5. Note that this table presents combined results obtained from Virtuoso and FUSEKI triplestores. We choose two triplestores as the query planner of the triplestore can greatly affect the query runtimes and hence biased the result towards a particular query planner. The overall results show that the number of projection variables (correlation 0.32) has the highest impact and BGP-restricted triple pattern selectivity (correlation 0.00) has the lowest impact on query runtimes. Yet, the result suggests that there is no single query feature that has a strong or very strong correlation with the query runtimes. This further suggests that the overall query runtime is impacted by a combination of different features.

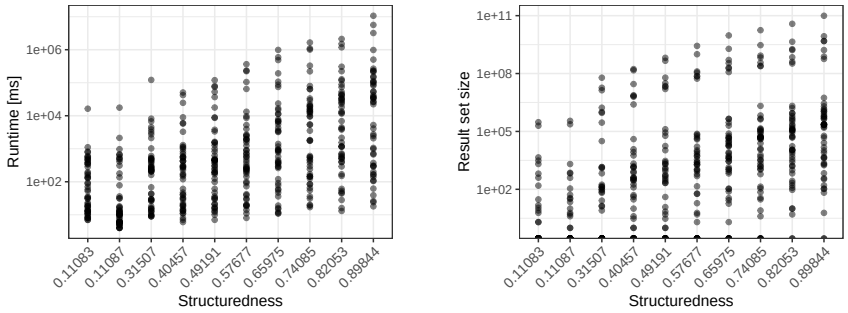
Table 4: WatDiv datasets with varying structuredness values. *DS:* Data Set, *SF:* the Structuredness Factor that controls the structuredness values of the entities used in the datasets.

DS	SF	Structuredness	Total Triples
1	0.1	0,110 878	103 394 017
2	0.2	0,110 835	103 395 364
3	0.3	0,315 073	105 813 355
4	0.4	0,404 579	99 583 093
5	0.5	0,491 919	98 636 578
6	0.6	0,576 772	102 975 692
7	0.6	0,659 752	102 463 145
8	0.7	0,740 858	107 058 360
9	0.8	0,820 532	98 341 615
10	0.9	0,898 443	108 962 809

5 RELATED WORK

Graph structure vs. query performance. The correlation between query runtime and workload metrics were studied in [17]. In addition to standard metrics, the authors introduced composite metrics such as the *absolute difficulty* (logarithm of the search space size), and the *relative difficulty*, which expresses how much worse a query engine does than the theoretical lower bound required by a certain query. The authors generated 12 graphs with different degree distributions along with 25+ queries of different shapes and measured their execution times. Then, they calculated the Kendall's τ rank correlation coefficient for $p < 0.001$ between each metric and the query execution times. The strongest correlation (+0.38) was exhibited by the *absolute difficulty* metric. The goal of gMark [4] is to define a *schema-driven workload generator* that synthesizes graph instances and queries for a given schema. The approach relies on controlling the diversity of the generated graphs and the difficulty of the generated workloads, using a *selectivity estimation* algorithm, which guarantees the selectivity of (certain) generated queries. The flexibility of their approach is demonstrated by generating workloads based on existing RDF benchmarks (SNB, SP2Bench, WatDiv).

Characterization of typed graphs. While this paper focuses on determining the correlation query execution time and graph metrics such as *relationship specialty* and *structuredness* (Sec. 2.1.1), other



(a) Structuredness vs. runtime **(b) Structuredness vs. result set size**

Figure 4: Correlation of structuredness value with runtimes and result set sizes. Note that the structuredness values are slighly off from the configured $[0.1, 0.2, \ldots, 1.0]$ values as the WatDiv generator can only target an approximate structuredness value.

Table 5: Spearman's rank correlation coefficients between query features and query runtimes. PV: Projection Variables, JV: Join Vertices, TP: Triple Patterns, JVD: Join Vertex Degree, JTPS: Join-Restricted Triple Pattern Selectivity, TPS: Triple Pattern Selectivity, BTPS: BGP-Restricted TPS. Correlations and colors (−+): 0.00…0.19 very weak (●●), 0.20…0.39 weak (●●), 0.40…0.59 moderate (●●), 0.60…0.79 strong (●●), 0.80…1.00 very strong (●●) .

	Rank	1	2	3	4	5	6	7	8	9	10
	Feature	PV	JV	TP	Result	JVD	JTPS	TPS	BGPs	LSQ	BTPS
Synthetic	Bowlogna	0.25	0.46	0.25	0.64	0.01	0.32	0.20	NaN	0.09	0.08
	TrainB.	0.76	0.48	0.46	0.85	0.61	0.03	0.67	−0.21	−0.09	−0.66
	BSBM	−0.28	0.17	−0.04	−0.36	−0.11	−0.24	0.21	−0.11	0.38	−0.16
	SP2Bench	0.59	0.38	0.49	0.84	0.34	−0.27	−0.06	0.24	0.24	0.32
	WatDiv	0.00	−0.09	0.05	0.65	0.23	−0.02	0.03	NaN	NaN	0.46
	SNB-BI	0.11	0.41	0.46	0.05	0.23	0.08	−0.20	0.31	−0.16	−0.20
	SNB-INT	−0.10	0.33	0.36	0.07	0.02	−0.08	0.05	0.27	0.20	−0.33
Real	FEASIBLE	0.25	0.23	0.27	0.20	0.33	0.39	0.16	−0.14	−0.12	0.32
	Fishmark	0.69	0.80	0.69	0.56	0.43	0.18	0.12	−0.13	−0.12	0.35
	DBPSB	0.23	0.45	0.42	0.16	0.02	0.11	0.10	0.52	0.31	−0.05
	BioBench	0.27	0.36	0.31	0.46	0.19	0.01	0.14	−0.03	0.12	0.15
Datasets	DBpedia	0.22	−0.04	0.10	0.07	−0.07	0.20	0.00	0.18	0.03	0.01
	SWDF	0.65	0.67	0.74	−0.17	0.67	−0.16	−0.18	0.76	0.45	−0.47
	NCBIGene	0.46	−0.49	−0.50	0.66	−0.49	0.37	0.29	−0.19	−0.13	0.66
	SIDER	0.19	−0.03	−0.03	0.02	−0.03	0.39	0.64	−0.01	−0.05	0.17
	DrugBank	0.81	0.88	0.86	−0.81	0.86	0.30	−0.50	0.06	−0.15	−0.2
	Overall	0.32	0.31	0.30	0.24	0.20	0.11	0.10	0.09	0.06	0.00

metrics could also provide valuable insight. In particular, the tools of *network science* are often used to uncover the structural interplay between the nodes of a certain graph [9], which in turn could be used to predict the difficulty of querying such graphs. However, most works in the field only target the characterization of homogeneous, untyped networks, which omit a great deal of valuable information when trying to understand the structure of typed networks such as RDF graphs. Only recent research targeted the understanding of typed graphs, referred to as "multilayer", "multiplex", or "multi-dimensional" networks. The authors of [7] generalized the degree distribution to take edge types into account and introduced a set of typed connectivity metrics. Extending this work, the authors of [19] defined a set of additional metrics to characterize the effect of types on nodes and edge pairs. Meanwhile, paper [6] introduced variants of the local clustering coefficient that described the ratio of typed triangles in the network. Survey [1] summarizes the state-of-the-art on analyzing multilayer (edge-typed) networks. Typed graph metrics were used successfully in the context of *model-driven engineering* to describe the structure of system models and distinguish real graph models from synthetic ones [29, 32]. However, these were limited to graphs containing at most 1M nodes.

In the fields of *semantic web* and *database engineering*, a set of simple typed graph metrics were proposed in the context for *social network analysis* in [13]. The concept of "meta-path", a path with a given sequence of node/edge types and its related literature were investigated in survey [28]. Duan et al. [12] presented the structuredness values of several real-world datasets and synthetic benchmarks. RBench [20] introduced the relationship specialty metric and compared real datasets with datasets synthesized by data generators. WatDiv [2] introduced the Join-restricted and BGP-restricted triple pattern selectivities as important query features.

Our work. In this paper, we conducted a systematic survey to collect a current list of triplestore benchmarks. We collected important query and dataset features from state-of-the-art [2, 12, 20, 24] and added additional important query features such as the number of projection variables, the number of BGPs, the number of LSQ features, etc. We compared 11 triplestore benchmarks and 5 real-world datasets across the identified important data and query features. We also measured the correlation of the identified query features with the overall query runtime. To the best of our knowledge, there exists no such detailed analysis of triplestore benchmarks.

6 CONCLUSION AND FUTURE WORK

We performed a comprehensive analysis of existing benchmarks by studying synthetic and real-world datasets as well as by employing SPARQL queries with multiple variations. Our evaluation results suggest the following: (1) The dataset structuredness problem is well covered in recent synthetic data generators (e.g., WatDiv, TrainBench). The low relationship specialty problem in synthetic datasets still exists in general and needs to be covered in future synthetic benchmark generation approaches; (2) The FEASIBLE

framework employed on DBpedia generated the most diverse benchmark in our evaluation; (3) The SPARQL query features we selected have a weak correlation with query execution time, suggesting that the query runtime is a complex measure affected by multi-dimensional SPARQL query features. Still, the number of projection variables, join vertices, triple patterns, the result sizes, and the join vertex degree are the top five SPARQL features that most impact the overall query execution time; (4) Synthetic benchmarks often fail to contain important SPARQL clauses such as `DISTINCT`, `FILTER`, `OPTIONAL`, `LIMIT` and `UNION`; (5) The dataset structuredness has a direct correlation with the result sizes and execution times of queries and indirect correlation with dataset specialty. As future work, we endeavour to broaden the scope of our analysis by adding more types of SPARQL query benchmarks (e.g., using reasoning, querying streams) and investigating more typed graph metrics on benchmark data sets.

ACKNOWLEDGMENTS

This work has been supported by the project HOBBIT (GA no. 688227), LIMBO (no. 19F2029I) and OPAL (no. 19F2028A) as well as by Science Foundation Ireland (SFI) under Grant No. SFI/12/RC/2289, MTA-BME Lendület Cyber-Physical Systems Research Group, and the ÚNKP-18-3-III New National Excellence Program of the Ministry of Human Capacities, Hungary. The authors would like to thank János Benjamin Antal for his assistance in reworking and benchmarking the LDBC SNB queries.

REFERENCES

[1] Alberto Aleta and Yamir Moreno. 2019. Multilayer Networks in a Nutshell. *Annual Review of Condensed Matter Physics* 10, 1 (2019). https://doi.org/10.1146/annurev-conmatphys-031218-013259

[2] Günes Aluç, Olaf Hartig, M. Tamer Özsu, and Khuzaima Daudjee. 2014. Diversified Stress Testing of RDF Data Management Systems. In *ISWC*. 197–212. https://doi.org/10.1007/978-3-319-11964-9_13

[3] Marcelo Arenas, Claudio Gutiérrez, and Jorge Pérez. 2009. On the Semantics of SPARQL. In *Semantic Web Information Management - A Model-Based Perspective*. Springer, 281–307. https://doi.org/10.1007/978-3-642-04329-1_13

[4] Guillaume Bagan, Angela Bonifati, Radu Ciucanu, George H. L. Fletcher, Aurélien Lemay, and Nicky Advokaat. 2017. gMark: Schema-Driven Generation of Graphs and Queries. *IEEE Trans. Knowl. Data Eng.* 29, 4 (2017), 856–869. https://doi.org/10.1109/TKDE.2016.2633993

[5] Samantha Bail, Sandra Alkiviadous, Bijan Parsia, David Workman, Mark van Harmelen, Rafael S. Gonçalves, and Cristina Garilao. 2012. FishMark: A Linked Data Application Benchmark. In *Proceedings of the Joint Workshop on Scalable and High-Performance Semantic Web Systems*. 1–15. http://ceur-ws.org/Vol-943/SSWS_HPCSW2012_paper1.pdf

[6] Federico Battiston, Vincenzo Nicosia, and Vito Latora. 2014. Structural measures for multiplex networks. *Phys. Rev. E* 89 (Mar 2014), 032804. Issue 3. https://doi.org/10.1103/PhysRevE.89.032804

[7] Michele Berlingerio, Michele Coscia, Fosca Giannotti, Anna Monreale, and Dino Pedreschi. 2013. Multidimensional networks: Foundations of structural analysis. *World Wide Web* 16, 5-6 (2013), 567–593. https://doi.org/10.1007/s11280-012-0190-4

[8] Christian Bizer and Andreas Schultz. 2009. The Berlin SPARQL Benchmark. *Int. J. Semantic Web Inf. Syst.* 5, 2 (2009), 1–24. https://doi.org/10.4018/jswis.2009040101

[9] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang. 2006. Complex networks: Structure and dynamics. *Physics Reports* 424, 4 (2006), 175 – 308. https://doi.org/10.1016/j.physrep.2005.10.009

[10] Felix Conrads, Jens Lehmann, Muhammad Saleem, Mohamed Morsey, and Axel-Cyrille Ngonga Ngomo. 2017. IGUANA: A Generic Framework for Benchmarking the Read-Write Performance of Triple Stores. In *ISWC*. Springer, 48–65. https://doi.org/10.1007/978-3-319-68204-4_5

[11] Gianluca Demartini, Iliya Enchev, Marcin Wylot, Joël Gapany, and Philippe Cudré-Mauroux. 2011. BowlognaBench - Benchmarking RDF Analytics. In *Data-Driven Process Discovery and Analysis SIMPDA*. Springer, 82–102. https://doi.org/10.1007/978-3-642-34044-4_5

[12] Songyun Duan, Anastasios Kementsietsidis, Kavitha Srinivas, and Octavian Udrea. 2011. Apples and oranges: A comparison of RDF benchmarks and real RDF datasets. In *SIGMOD*. ACM, 145–156. https://doi.org/10.1145/1989323.1989340

[13] Guillaume Erétéo, Michel Buffa, Fabien Gandon, and Olivier Corby. 2009. Analysis of a Real Online Social Network Using Semantic Web Frameworks. In *ISWC*. Springer, 180–195. https://doi.org/10.1007/978-3-642-04930-9_12

[14] Orri Erling, Alex Averbuch, Josep-Lluis Larriba-Pey, Hassan Chafi, Andrey Gubichev, Arnau Prat-Pérez, Minh-Duc Pham, and Peter A. Boncz. 2015. The LDBC Social Network Benchmark: Interactive Workload. In *SIGMOD*. ACM, 619–630. https://doi.org/10.1145/2723372.2742786

[15] Olaf Görlitz, Matthias Thimm, and Steffen Staab. 2012. SPLODGE: Systematic Generation of SPARQL Benchmark Queries for Linked Open Data. In *ISWC*. Springer, 116–132. https://doi.org/10.1007/978-3-642-35176-1_8

[16] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. 2005. LUBM: A benchmark for OWL knowledge base systems. *J. Web Sem.* 3, 2-3 (2005), 158–182. https://doi.org/10.1016/j.websem.2005.06.005

[17] Benedek Izsó, Zoltán Szatmári, Gábor Bergmann, Ákos Horváth, and István Ráth. 2013. Towards precise metrics for predicting graph query performance. In *ASE*. 421–431. https://doi.org/10.1109/ASE.2013.6693100

[18] Mohamed Morsey, Jens Lehmann, Sören Auer, and Axel-Cyrille Ngonga Ngomo. 2011. DBpedia SPARQL Benchmark - Performance Assessment with Real Queries on Real Data. In *ISWC*. Springer, 454–469. https://doi.org/10.1007/978-3-642-25073-6_29

[19] Vincenzo Nicosia and Vito Latora. 2015. Measuring and modeling correlations in multiplex networks. *Phys. Rev. E* 92 (Sep 2015), 032805. Issue 3. https://doi.org/10.1103/PhysRevE.92.032805

[20] Shi Qiao and Z. Meral Özsoyoglu. 2015. RBench: Application-Specific RDF Benchmarking. In *SIGMOD*. ACM, 1825–1838. https://doi.org/10.1145/2723372.2746479

[21] Shi Qiao and Z. Meral Özsoyoğlu. 2015. One Size Does not Fit All: When to Use Signature-based Pruning to Improve Template Matching for RDF graphs. *arXiv preprint arXiv:1501.07184* (2015). arXiv:1501.07184 http://arxiv.org/abs/1501.07184

[22] Muhammad Saleem, Muhammad Intizar Ali, Aidan Hogan, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo. 2015. LSQ: The Linked SPARQL Queries Dataset. In *ISWC*. Springer, 261–269. https://doi.org/10.1007/978-3-319-25010-6_15

[23] Muhammad Saleem, Ali Hasnain, and Axel-Cyrille Ngonga Ngomo. 2018. LargeRDFBench: A billion triples benchmark for SPARQL endpoint federation. *J. Web Sem.* 48 (2018), 85–125. https://doi.org/10.1016/j.websem.2017.12.005

[24] Muhammad Saleem, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo. 2015. FEASIBLE: A Feature-Based SPARQL Benchmark Generation Framework. In *ISWC*. Springer, 52–69. https://doi.org/10.1007/978-3-319-25007-6_4

[25] Muhammad Saleem, Alexander Potocki, Tommaso Soru, Olaf Hartig, and Axel-Cyrille Ngonga Ngomo. 2018. CostFed: Cost-Based Query Optimization for SPARQL Endpoint Federation. In *SEMANTICS (Procedia Computer Science)*, Vol. 137. Elsevier, 163–174. https://doi.org/10.1016/j.procs.2018.09.016

[26] Muhammad Saleem, Claus Stadler, Qaiser Mehmood, Jens Lehmann, and Axel-Cyrille Ngonga Ngomo. 2017. SQCFramework: SPARQL Query Containment Benchmark Generation Framework. In *K-CAP*. 28:1–28:8. https://doi.org/10.1145/3148011.3148017

[27] Michael Schmidt et al. 2009. SP2Bench: A SPARQL Performance Benchmark. In *Semantic Web Information Management - A Model-Based Perspective*. 371–393. https://doi.org/10.1007/978-3-642-04329-1_16

[28] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and Philip S. Yu. 2017. A Survey of Heterogeneous Information Network Analysis. *IEEE Trans. Knowl. Data Eng.* 29, 1 (2017), 17–37. https://doi.org/10.1109/TKDE.2016.2598561

[29] Gábor Szárnyas et al. 2016. Towards the characterization of realistic models: Evaluation of multidisciplinary graph metrics. In *MODELS*. 87–94. http://dl.acm.org/citation.cfm?id=2976786

[30] Gábor Szárnyas, Benedek Izsó, István Ráth, and Dániel Varró. 2018. The Train Benchmark: Cross-technology performance evaluation of continuous model queries. *Softw. Syst. Model.* 17, 4 (2018), 1365–1393. https://doi.org/10.1007/s10270-016-0571-8

[31] Gábor Szárnyas, Arnau Prat-Pérez, Alex Averbuch, József Marton, Marcus Paradies, Moritz Kaufmann, Orri Erling, Peter A. Boncz, Vlad Haprian, and János Benjamin Antal. 2018. An early look at the LDBC Social Network Benchmark's Business Intelligence workload. In *GRADES-NDA at SIGMOD*. ACM, 9:1–9:11. https://doi.org/10.1145/3210259.3210268

[32] Dániel Varró, Oszkár Semeráth, Gábor Szárnyas, and Ákos Horváth. 2018. Towards the Automated Generation of Consistent, Diverse, Scalable and Realistic Graph Models. In *Graph Transformation, Specifications, and Nets - In Memory of Hartmut Ehrig*. Springer, 285–312. https://doi.org/10.1007/978-3-319-75396-6_16

[33] Hongyan Wu et al. 2014. BioBenchmark Toyama 2012: An evaluation of the performance of triple stores on biological data. *J. Biomedical Semantics* 5 (2014), 32. https://doi.org/10.1186/2041-1480-5-32

Self- and Cross-Excitation in Stack Exchange Question & Answer Communities

Tiago Santos
Graz University of Technology
tsantos@iicm.edu

Simon Walk
Detego GmbH
s.walk@detego.com

Roman Kern
Graz University of Technology
Know-Center
rkern@tugraz.at

Markus Strohmaier
RWTH Aachen University
markus.strohmaier@cssh.rwth-aachen.de

Denis Helic
Graz University of Technology,
dhelic@tugraz.at

ABSTRACT

In this paper, we quantify the impact of self- and cross-excitation on the temporal development of user activity in Stack Exchange Question & Answer (Q&A) communities. We study differences in user excitation between growing and declining Stack Exchange communities, and between those dedicated to STEM and humanities topics by leveraging Hawkes processes. We find that growing communities exhibit early stage, high cross-excitation by a small core of power users reacting to the community as a whole, and strong long-term self-excitation in general and cross-excitation by casual users in particular, suggesting community openness towards less active users. Further, we observe that communities in the humanities exhibit long-term power user cross-excitation, whereas in STEM communities activity is more evenly distributed towards casual user self-excitation. We validate our findings via permutation tests and quantify the impact of these excitation effects with a range of prediction experiments. Our work enables researchers to quantitatively assess the evolution and activity potential of Q&A communities.

CCS CONCEPTS

• **Human-centered computing** → *Collaborative and social computing*; • **Mathematics of computing** → *Stochastic processes.*

KEYWORDS

Q&A communities, Excitation effects, Hawkes processes

ACM Reference Format:
Tiago Santos, Simon Walk, Roman Kern, Markus Strohmaier, Denis Helic. 2019. Self- and Cross-Excitation in Stack Exchange Question & Answer Communities. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313440

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313440

1 INTRODUCTION

Why and *how* some Question & Answer (Q&A) communities gain traction and attract activity from large numbers of users—while others do not—are questions of theoretical and practical relevance [31, 38]. Understanding how users become active in such systems, and how user activity evolves over time, can be considered an important stepping stone towards better modeling and shaping of online Q&A communities. This will allow to devise novel approaches to guide and encourage activity [12] and to support community managers in their community building efforts [2, 20].

User excitation. In this paper, we investigate *self-excitation* and *cross-excitation* of users in Q&A communities. Self-excitation reflects how a user's *own* past activity shapes her future activity, while cross-excitation reflects how *other users'* past activity influences future activity of a given user. Modeling temporal traces of user activity informs the inference of excitation effects and thus provides a first step towards deeper causal analysis of user excitation.

In the present work, we adopt point processes [7–9]—in particular Hawkes processes [14]—to leverage temporal traces of user activity as latent indicators of self- and cross-excitation. We empirically analyze 69 Stack Exchange Q&A instances where we fit a multivariate Hawkes process model[1]. With that, we are able to analyze self- and cross-excitation of users across: (i) communities with growing and declining activities, (ii) the topics of the conversations, (iii) activity types (e.g., question, answers), and (iv) activity source (e.g., power user, casual users). Subsequently, we characterize self- and cross-excitation as a function of community age. We then validate, with a range of statistical tests, the excitation effects we uncover, and we quantify their relative importance in the evolution of Q&A communities with a prediction experiment. We illustrate various types of excitation and how they generate user activity in Figure 1.

Findings. Our empirical findings emphasize the need for Q&A communities to maintain a steady core of highly cross-excited power users (i.e. very active users) reacting, particularly in a community's early stages, to the community as a whole. In thriving communities, casual users (less active users) shape each others' activity levels via cross-excitation. This suggests that growing communities are, in general, facilitating and embracing less active and casual users, thereby offering low barriers of entry. Additionally, we observe late-stage domination of self-excitation over cross-excitation, meaning

[1] We make our code available at https://github.com/tfts/Excitation_in_QA.

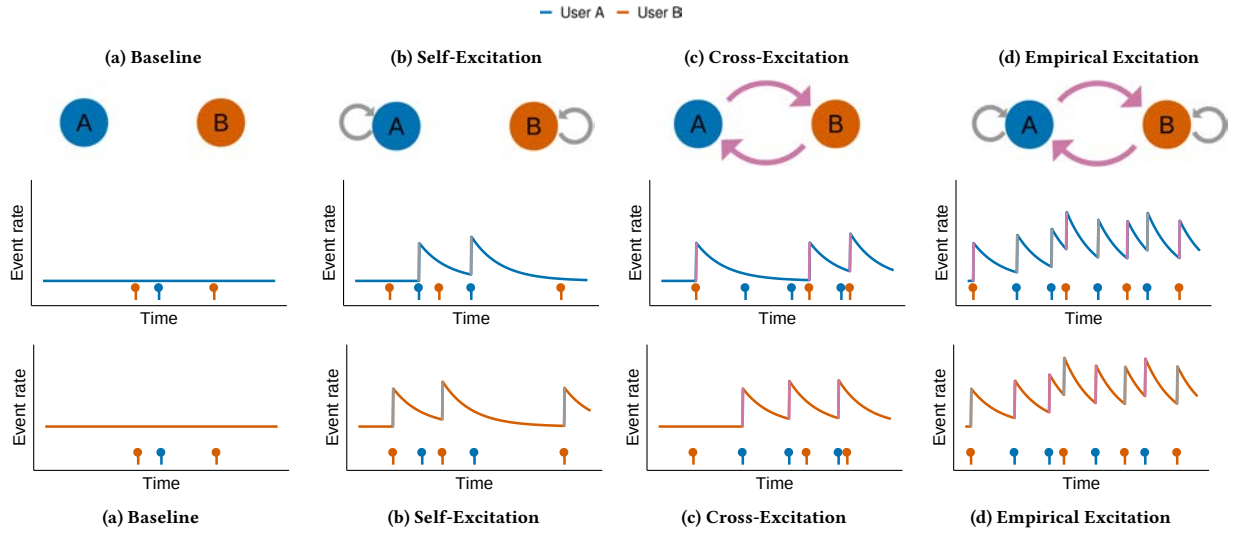


Figure 1: Excitation types. We distinguish between three drivers of user activity excitation: (a) Baseline, a constant base event rate level, (b) Self-Excitation, a proxy for increased propensity by a user to be active in the future following her past activity, and (c) Cross-Excitation, a boost to event rate triggered by activity of other users. The upper row of this Figure depicts the links between users A and B per excitation type, and the two lower rows the corresponding event rate as it reacts to the user's own and others' activity events, marked by the trees below each event rate line and colored by the corresponding user. The first three excitation types cover excitation components which combine to form (d) empirical excitation. This work characterizes and quantifies how each type of excitation manifests itself in Q&A communities and how excitation strength changes over time as a Q&A community develops.

that self-driven activity becomes a crucial factor in successful communities. This effect may serve as a long-term growth indicator, as this self-excitation dominance is most prominent in growing communities. Finally, we observe differences in user participation across distinct topics: Q&A communities dedicated to topics in the humanities (such as languages) are more driven by cross-excitation of power users, whereas those in STEM-related fields are not.

With our work we make the following contributions. First, we model self- and cross-excitation effects in successful and unsuccessful Q&A communities. Second, we empirically show how self- and cross-excitation manifests in communities defined by different levels of success and different topics. Third, our validation provides a foundation for building further predictive models of user activity in Q&A communities. Finally, we provide and illustrate an approach that allows community managers to quantitatively compare long-term dynamics of their online Q&A communities—in terms of user excitation—to well-established ones.

2 HAWKES PROCESSES

A point process can be broadly defined as a collection of points randomly located in some mathematical space. Temporal point processes employ the real line, representing time, as the underlying mathematical space. For the interested reader, the work by Daley and Vere-Jones [8, 9] and by Cox and Isham [7] are comprehensive references on point process theory.

In practice, temporal point processes model the arrival of discrete events over time with the help of a *conditional intensity function* λ^*, a stochastic model for the arrival of the next event given the event history. Hawkes processes [14] are a particular class of temporal point processes, which assume a particular functional form for the intensity function. Specifically, the intensity function of Hawkes processes is in itself a stochastic process and it explicitly encodes *self-excitation*, the increase in intensity caused by past events:

$$\lambda^*(t) = \mu + \sum_{t_i < t} \alpha e^{-\beta(t-t_i)}, \tag{1}$$

where $\mu > 0$ is the baseline intensity independent on the event history, and $\alpha, \beta > 0$ establish the dependence on previous events. In particular, each previous event at time t_i increases the intensity by α, the self-excitation factor. We choose to let the intensity jumps exponentially decay at the rate β, which is a commonly used functional form of intensity decay called an *exponential kernel*.

Equation 1 describes *univariate* Hawkes processes, as they consider only the effect of past events in future event times of the same event stream. The *multivariate* generalization of univariate Hawkes processes includes not only self-excitation but also *cross-excitation*. Cross-excitation is the intensity increase that an event in one event stream implies in another event stream. More formally, let $\boldsymbol{N}(t) = (N^1(t), N^2(t), \ldots, N^M(t))$ be a simple multivariate point process, where each of the $N^i(t)$ is a counting process in the i-th dimension. An M-variate Hawkes process with an exponential kernel is defined by the following intensity function:

$$\lambda^{*m}(t) \quad = \quad \mu_m + \sum_{n=1}^{M} \sum_{t_i^n < t} \alpha_{mn} e^{-\beta_{mn}(t-t_i^n)}. \tag{2}$$

We write μ_m as the baseline intensity in dimension m, α_{mn} as the cross-excitation on dimension m caused by an event in dimension n and the corresponding decay rate as β_{mn}. In matrix notation, we

write $\boldsymbol{\mu} \in \mathbb{R}^M$, $\boldsymbol{\alpha} \in \mathbb{R}^{M \times M}$ and $\boldsymbol{\beta} \in \mathbb{R}^{M \times M}$. Linniger [23] provides a more detailed treatment of multivariate Hawkes process theory.

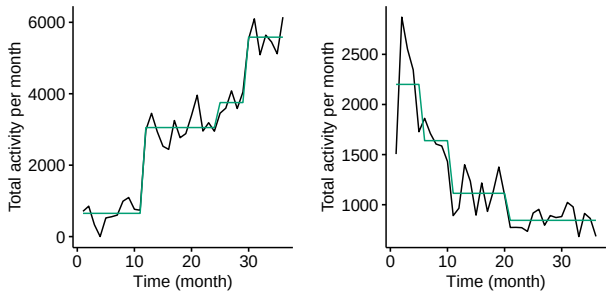
Samples from multivariate Hawkes processes, which can be obtained via Ogata's thinning algorithm [27], generate self-excitation and cross-excitation effects of the kind depicted in Figure 1, where users A and B correspond to two dimensions of a Hawkes process and their event rate to the Hawkes process intensity. We observe, in both dimensions, intensity peaks corresponding to the sampled events, and we note the intensity decays exponentially until converging to the baseline intensity level $\boldsymbol{\mu}$.

Fitting Hawkes processes. Given an observed sequence of events $\{t_i\}$, we fit the parameters of Hawkes processes by maximizing its *log-likelihood*. Closed form expressions for the log-likelihood can be derived for many different types of intensity function kernels, including the exponential kernel that we assume. For a given $\boldsymbol{\beta}$, all other parameters of the process may be estimated by maximizing the log-likelihood via well-known convex optimization methods such as Levenberg-Marquardt [21]. However, fitting $\boldsymbol{\beta}$-s is a challenging task, since the likelihood functions of Hawkes processes with exponential kernels are either flat around the optimal $\boldsymbol{\beta}$ (see, for example, Upadhyay et al. [36]) or, in some other formulations of the kernel function, even non-convex in $\boldsymbol{\beta}$. In this paper, we propose an effective Bayesian hyperparameter optimization step, which allows fitting the decay-related and then the remaining parameters of a Hawkes process. Assuming $\beta_{m,n} = \beta, \forall_{1 \leq m,n \leq M}$ [12, 35], we apply the Tree of Parzen Estimators approach, as described by Bergstra et al. [4], on the convex optimization routine of log-likelihood for a given set of event sequences to estimate β. We perform 15 runs of the Bergstra et al. algorithm and keep the β yielding highest likelihood, since this effectively allows for convergence even in the presence of flat plateaus around local maxima of Hawkes likelihood as a function of β. Finally, using the learned β, we fit $\boldsymbol{\mu}$ and $\boldsymbol{\alpha}$.

Furthermore, practical fitting of Hawkes processes requires the fits to be done on *stationary* [23] periods of the corresponding count time series. Stationarity, in this context, refers to translation-invariance in the Hawkes process distribution, which implies a linear growth in the associated time series of event counts over time. However, in the time series representing activity in the Q&A communities we work with, we observe a range of non-stationary phenomena: exponential growth and decline and other sudden structural changes, such as level jumps. Therefore, we need to restrict the fitting procedure to stationary subsequences of an observed event stream. To that end, we use the time series structural change detection algorithm devised by Zeilis et al. [45]. Given a linear regression model, this algorithm returns optimal points in time for structural change in the regression model's fit to a given input time series. Using a constant regression model allows us to detect level changes in an input event count time series, and thus to segment it into stationary subsequences.

3 EXPERIMENTAL SETUP

We study user self- and cross excitation by empirically analyzing Stack Exchange instances. We distinguish activity in these datasets by two aspects: (i) activity content, which we define as questions and answers (with the latter including answers and comments), and (ii) activity source, meaning whether it originated from highly active (power) or less active (casual) users. Further, we explore the differences in excitation across communities, which we group according to two criteria: (i) growth pattern, and (ii) topical focus.



(a) Growing instance (electronics) **(b) Declining instance (cstheory)**

Figure 2: Exemplary *growing* (left) and *declining* (right) Stack Exchange instances. These two curves exemplify the monthly total activity time series of a *growing* (*declining*) instance, electronics (cstheory). Following Zeilis et al., we indicate stationarity in subsequences of both time series with the green lines. Note that *growth* (*decline*) in electronics (cstheory) was, at 757.62% (−61.63%), one of the highest (lowest) among Stack Exchange instances we analysed.

Datasets. Stack Exchange encompasses several Q&A communities, termed Stack Exchange instances, with each dedicated to Q&A on a single topic, such as computer science, the English language or movies. We extract user activity in all 159 Stack Exchange instances[2] (as of June 2017) as the timestamps of users' activity events: posts (i.e. questions) and replies (i.e. answers and comments). In a first step, we consider these instances' complete history, which spans the period from August 2008 to June 2017 and comprises a total of 22 million events. However, our analysis is independent from the calendar date a Stack Exchange instance originated, as we map the inception of each instance to a time scale starting at zero.

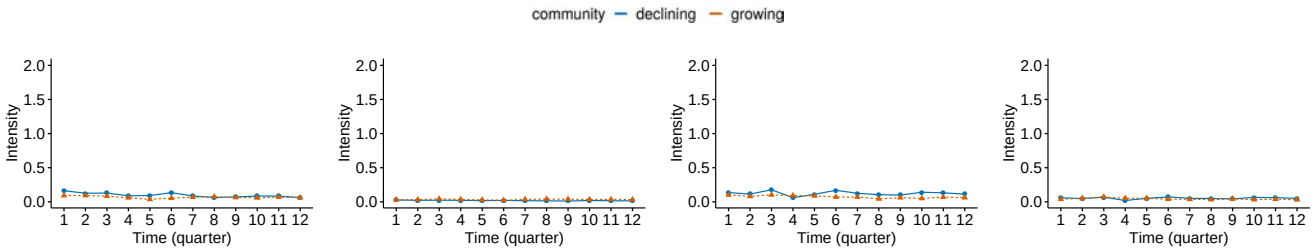
Before we group Stack Exchange instances according to our two criteria, growth patterns and topical focus, we discard all datasets with less than ten events in any period of three months, which ensures that we have enough events for the fitting procedure.

Growth pattern. In our first comparison by growth pattern, we analyze the first three years of existence of Stack Exchange instances, so we exclude datasets with durations shorter than three years. To better distinguish excitation effects driving overall activity increase or decrease in these communities, we then extract, from the remaining datasets, two groups of strongly *growing* and strongly *declining* datasets. The extraction criterion stems from our application of Zeilis et al.'s algorithm to find level structural changes in the time series of total activity count per month: We define a dataset as strongly growing (declining), if the percentage change in structural level from the first fitted window to the last fitted window is in the 80^{th} (20^{th}) percentile over all datasets. The grouping of Stack Exchange instances into *growing* and *declining* yields two groups of 22 datasets each, of which we provide descriptive statistics in Table 1. Note that the *growing* (*declining*) group only includes instances with strongly positive (negative) growth.

[2]The Stack Exchange dataset is available at https://archive.org/details/stackexchange.

Table 1: Dataset characteristics. We show the datasets per group sorted by activity growth (top and bottom three growth percentages per group shown in parenthesis), the total number of datasets per group (#), as well as ranges for a number of descriptive statistics: the activity total as the sum of all questions and answers, the age in years and the total growth as a percentage of the level of the first subsequence found by Zeilis et al.'s algorithm. We observe a clear separation in strongly positive and negative growths (and thus also total activity) in the major distinction we draw between datasets, *growing* vs. *declining*. This distinction is remarkably less pronounced in *STEM* vs. *humanities* instances, which both feature positive and negative growths.

Dataset Group	Datasets	#	Activity total	Age (years)	Growth (%)
Stack Exchange Growing	electronics (757.62%), ru (736.42%), codegolf (510.06%), chemistry, sharepoint, academia, puzzling, tex, codereview, blender, unix, money, gis, ux, crypto, security, stats, salesforce, dba, wordpress (182.28%), opendata (174.69%), askubuntu (169.29%)	22	[7987, 1489384]	[3.08, 7.83]	[169.29, 757.62]
Stack Exchange Declining	boardgames (−28.53%), fitness (−34.56%), sound (−35.01%), productivity, tridion, parenting, pets, craftcms, webapps, spanish, cooking, ham, bricks, gardening, cstheory, expressionengine, pm, skeptics, sustainability, genealogy (−80.26%), ebooks (−81.52%), stackapps (−82.7%)	22	[3301, 117474]	[3, 7.75]	[−82.7, −28.53]
Stack Exchange STEM	electronics (757.62%), chemistry (473.48%), stats (199.18%), biology, datascience, physics, astronomy, cs, space, cogsci, earthscience, engineering, reverseengineering (0.00%), softwareengineering (−21.28%), sound (−35.01%)	15	[15759, 745674]	[2.41, 8.75]	[−35.01, 757.61]
Stack Exchange Humanities	philosophy (122.45%), english (117.76%), chinese (23.17%), music, german, mythology, portuguese, christianity, esperanto, arabic, russian, writers, buddhism (−26.62%), french (−27.91%), spanish (−50.10%)	15	[87, 896631]	[0.17, 6.83]	[−50.10, 127.47]



(a) Baseline of Answers by Power Users (b) Baseline of Questions by Power Users (c) Baseline of Answers by Casual Users (d) Baseline of Questions by Casual Users

Figure 3: Low baseline excitation in *growing* and *declining* communities. Given the Hawkes process dimensions questions and answers per power and casual users, we depict the baseline parameters μ of the Hawkes processes fitted every three months over three years of *growing* (orange lines) and *declining* (blue lines) Stack Exchange instances. Error bars in this Figure and Figures 4 and 5 show bootstrapped 95% confidence intervals, many of which are too small to be visible. Note our use of the same scale in Figures (a)-(d) and throughout Figure 4: The relatively low baseline intensities in comparison to the effects depicted in Figure 4 stress that overall activity is driven by self- or cross-excitation rather than baseline intensity.

We plot the total monthly event counts for a selected dataset from each dataset group in Figure 2 to exemplify their activity curves and the detected structural level changes. Often, there are prolonged periods of stagnancy in one structural level, both in *growing* as well as *declining* datasets. Typically, such periods vary in length.

Topical focus. For the topical comparison, we study Stack Exchange instances dedicated to *STEM* (i.e. science, technology, engineering and mathematics) and *humanities* topics. To that end, we randomly picked a set of 15 Stack Exchange instances we manually classified as STEM topics, and another 15 as humanities. The instances in these two groups vary in size and age, and feature no distinctive growth patterns, although some *humanities* instances are smaller and have shorter overall durations than *STEM* instances (cf. Table 1). In this comparison, we also analyze instances' first three years, but we do not impose a minimum duration, which leads to fewer than 15 instances per group later in time. However, the number of instances per group remains comparable over time and reaches a minimum of nine instances per group by the third year.

Hawkes process application. In the Stack Exchange instances, we distinguish between more active and less active users, which we term *power users* and *casual users*. This definition mainly distinguishes a *core* of remarkably engaged power users typically found in Q&A communities [13, 24, 41] from casual users. Thus, for each dataset individually, we count the total activity per month per user, and per activity type (question or answer) and postulate that power users are those in the 90^{th} percentile of most active users for that month. This implies that this monthly group of power users is ever-changing, as users join and leave the communities or as the users' intrinsic motivation to contribute content rises and falls over time. Note that our results changed only minimally with different percentile thresholds (i.e., 85^{th} and 95^{th}) for power user classification. To measure self- and cross-excitation per user and activity event type, we map the event stream of question and answer activity to

four Hawkes process dimensions: questions by power users, questions by casual users, answers by power users and answers by casual users. For each such dimension, we work with the corresponding event timestamps at the resolution of a second.

We then follow the procedure outlined in Section 2 to fit four-dimensional Hawkes processes to each dataset group (Stack Exchange instances in the groups *declining* vs. *growing* and *STEM* vs. *humanities*). For each dataset group comparison, we begin by fitting overall β for all datasets over the first three years of their existence. Then, we perform structural level change fits on the total monthly event count, and observe a minimal window length of five months. According to our experimentation with different window lengths, specifically two to six months, we find a window length of three months is long enough to ensure we have enough events per window and do not overfit a particular window, while also short enough to capture granular changes in the evolution of the underlying Hawkes process distribution. Hence, we set the constant window length to three months (a quarter).

To measure variability in the evolution of the fitted models, we bootstrap, with 100 repetitions, the fitting procedure of all Stack Exchange instances per dataset group per window. From the resulting bootstrap distribution, we compute 95% confidence intervals for the mean value of each fitted Hawkes process parameter. We display the confidence intervals as error bars in Figures 3, 4 and 5.

4 EXCITATION EFFECTS

4.1 Comparison by growth pattern

The β value, fitted over 44 datasets in the *growing* vs. *declining* comparison and the whole three year period, is 2.288, corresponding to an intensity half-life of about 0.3 hours (meaning that the intensity jump of magnitude α caused by either self-excitation or cross-excitation decays to $\alpha/2$ after about 18 minutes). With a single constant β, we restrict our model to capture distributional changes in terms of baseline, self- and cross-excitation intensities, allowing us to focus on these factors as direct proxies for the role of different user groups in overall activity intensity over time.

We visualize the evolution of all baseline parameter values for questions and answers by power users and casual users in *growing* (orange) and *declining* (blue) instances in Figure 3. We depict the corresponding self- and cross-excitation parameter values in Figure 4. Note that we employ the same scales throughout both Figures for better comparison.

Low baseline intensities. The Figures 3a through 3d show the baseline intensities ($\boldsymbol{\mu}$) fitted over time. We observe roughly constant baseline intensities throughout the whole period, for both *growing* and *declining* instances. Furthermore, the baseline intensities are rather low, especially in comparison with the self-excitation and cross-excitation effects ($\boldsymbol{\alpha}$) depicted in Figure 4.

Finding: Constant and low baseline intensities suggest Q&A communities thrive off self- and cross-excitation, representing interaction between different user (power and casual) and activity types (questions & answers), rather than featuring constant levels of activity over time, independent from other activity dimensions.

Early Power User Cross-Excitation (■). We continue our analysis with the Hawkes process dimension with the highest intensity values: intensity in answer activity by power users (*first row* in Figure 4, i.e. Figures 4a through 4d). We observe, in the early stages of *growing* instances, high impact of questions by both power as well as casual users on answer activity by power users (see yellow region highlighted in Figures 4b and 4d). In contrast, in *declining* instances, especially in the yellow highlighted region, questions by both types of users elicit declining numbers of answers by power users over time, and self-excitation in answers by power users dominates over all temporal windows. Regarding question activity by power users (*second row* in Figure 4, i.e. Figures 4e through 4h), there is a clear prevalence of self-excitation intensity with respect to other cross-excitation intensities. However, we observe, albeit minor, differences in the short and medium terms between *growing* and *declining* instances, as power users in *growing* instances are more encouraged to participate with new questions as a response to questions by casual users and answers by both (see inlines of Figures 4e, 4g, 4h).

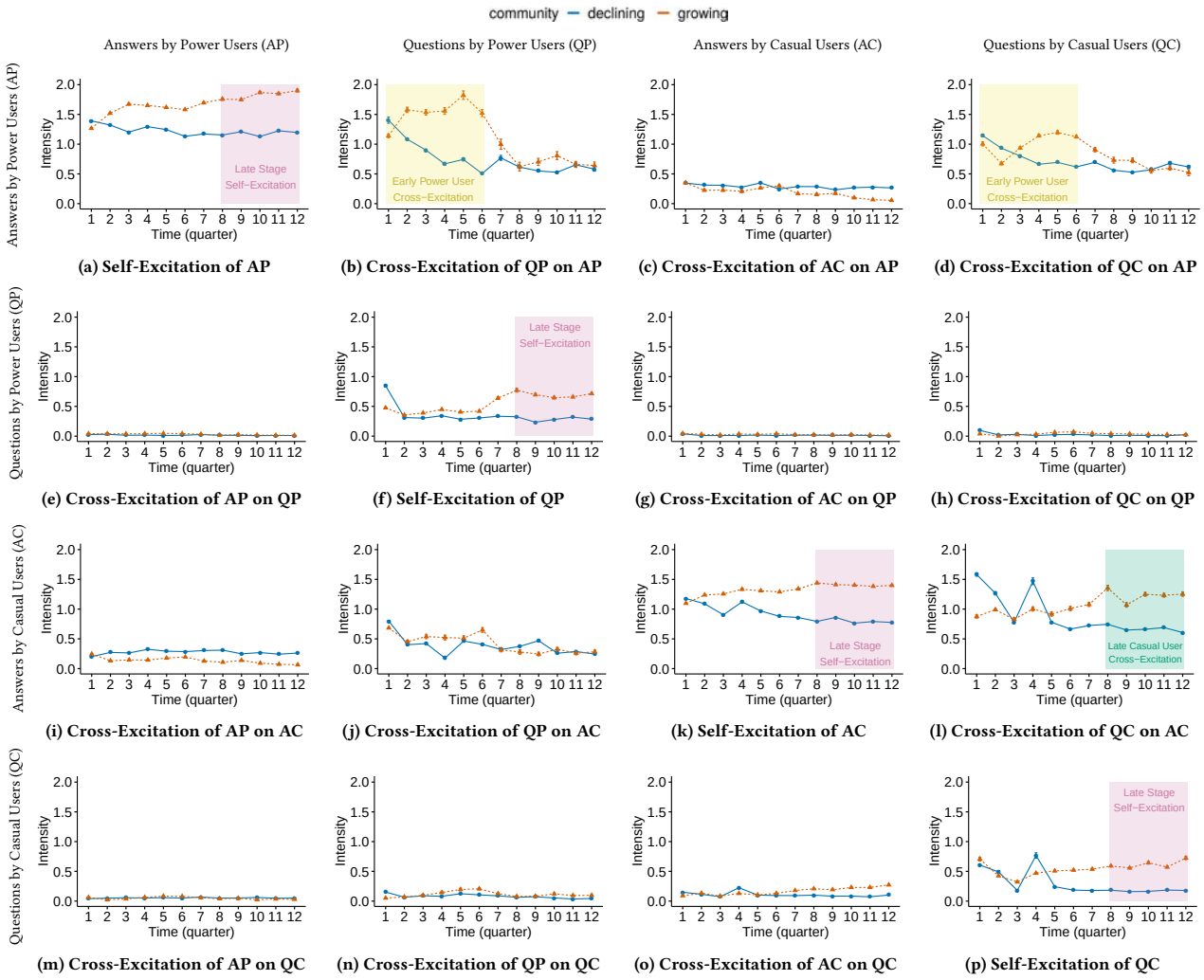
Finding: These observations suggest strong activity by power users, as a response to questions by both power users as well as, crucially, casual users, is related to increased growth in the early stages of Q&A communities. This finding suggests the importance of an active core of users to jumpstart Q&A community development.

Late Casual User Cross-Excitation (■). In Figure 4's *third row* (Figures 4i through 4l), we highlight another type of effect: Answers and discussion by casual users is driven strongly by questions also from casual users, especially in the long-term as highlighted by the green region of Figure 4l. The main difference between *growing* and *declining* instances in this dimension is, besides the intensity magnitude difference, that this cross-excitation effect loses importance in the long-term in the *declining* instances, while overall it does not in *growing* instances. We point out one interesting effect in the *fourth row* (Figures 4m through 4p), which depicts the question intensity dimensions of casual users: In the long term, there is a small increase in questions by casual users after answers also by casual users in *growing* Stack Exchange instances (cf. quarters eight through twelve of Figure 4o).

Finding: Long-term cross-excitation from questions on answers by casual users is a key factor present in *growing* Stack Exchange instances and lacking in *declining* ones. We find contributions by casual users thus attract more participation by casual users, likely helping to sustain and even enhance activity levels. Hence, we identify openness from the community towards casual users in the form of healthy interaction between them as a sign of enduring community growth.

Late Stage Self-Excitation (■). In the diagonal of Figure 4, consisting of Figures 4a, 4f, 4k and 4p, we observe strong and growing self-excitation effects, which dominate over cross-excitation effects in the long-term. We indicate long-term with the pink region marking quarters 8 to 12, the last five quarters we fit. We note this effect is most predominant in *growing* communities. Further, for *growing* instances, notice that, for a given dimension (e.g. answers by power users, Figures 4a through 4d), the timing of the surge in self-excitation coincides in general with a decline in cross-excitation.

Finding: In *growing* Stack Exchange instances, we attribute the phenomenon of higher long-term self-excitation to steadily growing arrivals of questions and answers from power and casual users. As users react to a constantly and regularly growing pool of questions and answers, this makes distinction of direct interaction between



(a) Self-Excitation of AP (b) Cross-Excitation of QP on AP (c) Cross-Excitation of AC on AP (d) Cross-Excitation of QC on AP

(e) Cross-Excitation of AP on QP (f) Self-Excitation of QP (g) Cross-Excitation of AC on QP (h) Cross-Excitation of QC on QP

(i) Cross-Excitation of AP on AC (j) Cross-Excitation of QP on AC (k) Self-Excitation of AC (l) Cross-Excitation of QC on AC

(m) Cross-Excitation of AP on QC (n) Cross-Excitation of QP on QC (o) Cross-Excitation of AC on QC (p) Self-Excitation of QC

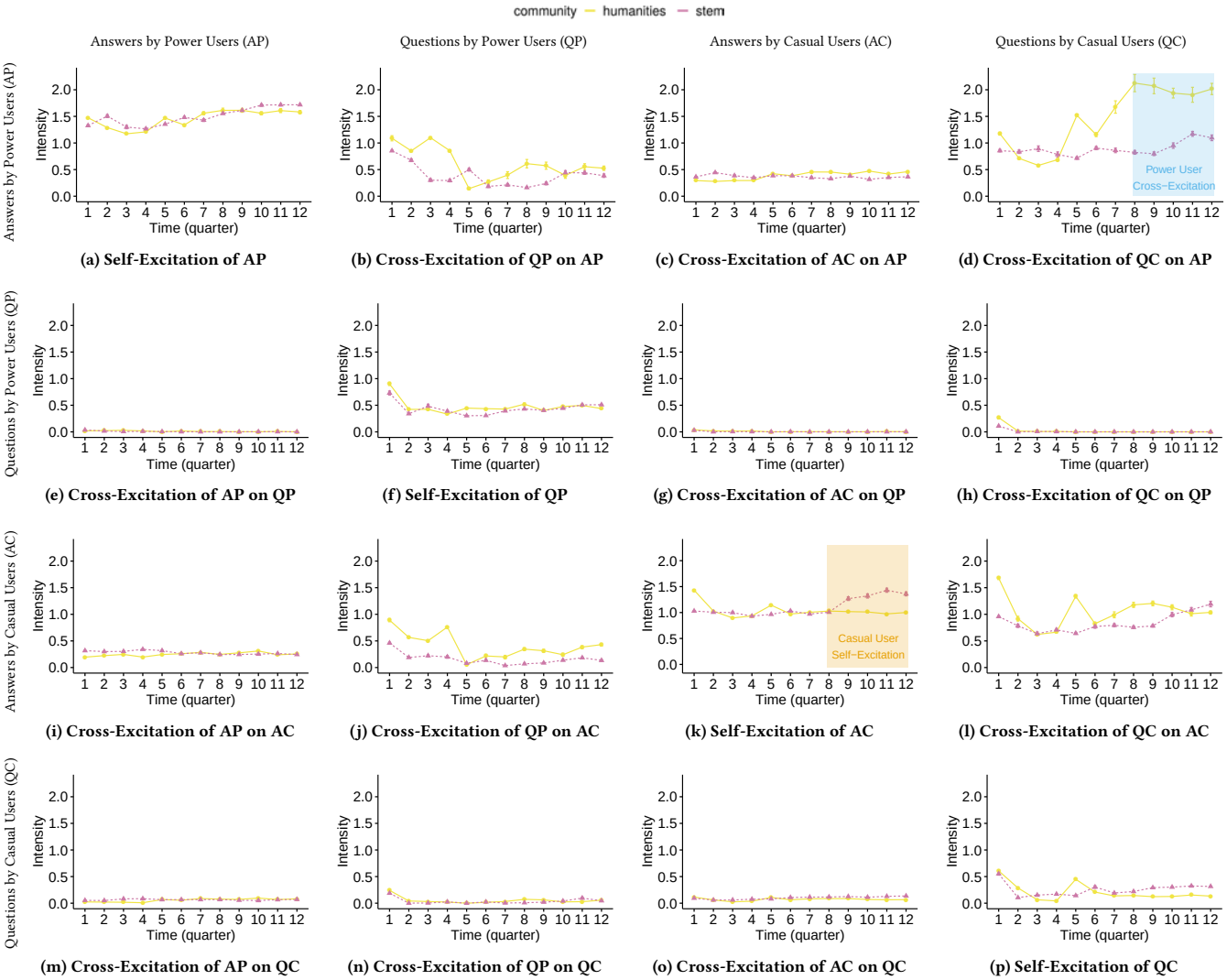
Figure 4: Excitation in *growing* vs. *declining* communities. Given the Hawkes process dimensions questions (Q) and answers (A) per power (P) and casual (C) users, we depict the α matrix of self-excitation (diagonal plots) and cross-excitation (off-diagonal plots) parameters of the Hawkes processes fitted every three months over three years of *growing* (orange lines) and *declining* (blue lines) Stack Exchange instances. The first and second row, respectively, show how answer and question intensities by power users are self- and cross-excited, and the third and fourth the influence on answer and question intensity by casual users as a result of self-excitation and cross-excitation. The yellow highlighted regions ( ) of Figures (b) and (d) depict the effects of questions by power and casual users on answers by power users and thus the crucial importance of power user cross-excitation in driving early stage dynamics of *growing* instances. In Figure (l) we observe another difference between the groups: *growing* communities also thrive off interaction between casual users, as shown by long-term cross-excitation of questions by casual users on answers by casual users (cf. green region of Figure (l)). In the long-term, self-excitation (pink highlighted regions of diagonal entries, i.e. Figures (a), (f), (k), and (p)) is the most dominant form of excitation in all four Hawkes process dimensions.

single questions and corresponding answers harder over time. The timing of this self-excitation surge may be of particular interest for Q&A community managers, who may be concerned about growth should they not observe this effect by the community's third year.

4.2 Comparison by topic

The comparison between *STEM* and *humanities* instances of Figure 5 shares a few commonalities with our previous findings on the *growing* and *declining* instance comparison: roughly constant and relatively low baseline intensities (not depicted due to limitations in space) and comparatively high long-term self-excitation. In this comparison, we obtained $\beta = 2.067$, which corresponds to an intensity half-life of 0.33 hours. These values are comparable to the ones we obtained previously, which may indicate a universal pattern of user activity decline across Stack Exchange instances.

Power User Cross-Excitation ( ) vs. Casual User Self-Excitation ( ). In the light blue highlighted region of Figure 5d,



(a) Self-Excitation of AP (b) Cross-Excitation of QP on AP (c) Cross-Excitation of AC on AP (d) Cross-Excitation of QC on AP

(e) Cross-Excitation of AP on QP (f) Self-Excitation of QP (g) Cross-Excitation of AC on QP (h) Cross-Excitation of QC on QP

(i) Cross-Excitation of AP on AC (j) Cross-Excitation of QP on AC (k) Self-Excitation of AC (l) Cross-Excitation of QC on AC

(m) Cross-Excitation of AP on QC (n) Cross-Excitation of QP on QC (o) Cross-Excitation of AC on QC (p) Self-Excitation of QC

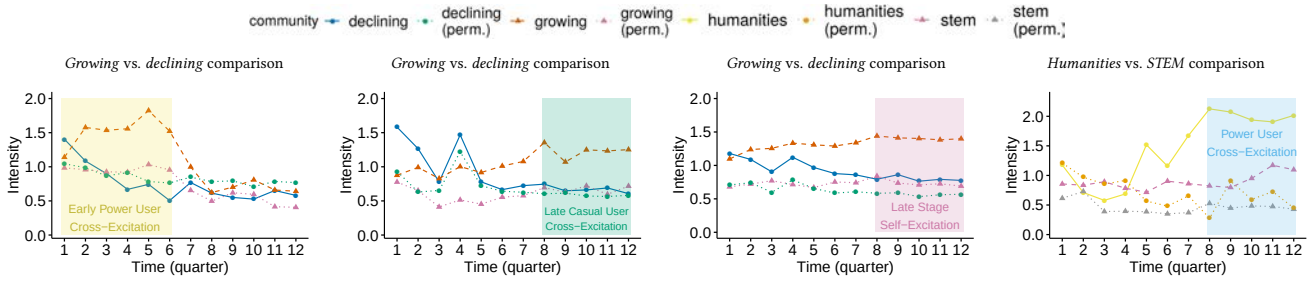
Figure 5: Excitation in *humanities* vs. *STEM*. Using the same notation and format as Figure 4, we depict the parameters of the Hawkes processes fitted every three months over three years of Stack Exchange instances dedicated to *STEM* (purple lines) and *humanities* (yellow lines) topics. We observe a more prominent role by power users in *humanities*, as indicated by by the more important role of power user cross-excitation originating from power users answering questions by casual users (cf. blue highlighted region ■ of Figure (d)). Furthermore, regarding casual user activity in *STEM* vs. *humanities* instances, we note the former's casual users feature more prominently in the long-term in the form of higher answer self-excitation (cf. orange highlighted region ■ of Figure (k)).

we stress the notable role of answer activity by power users in *humanities* instances. We observe answers by power users after questions from casual users is notably higher in *humanities* instances than in *STEM* instances (see Figure 5d). With the light orange region of Figure 5k, we underline the counterpart in casual user activity: There are higher long-term intensities in self-excitation of answers by casual users in *STEM* instances as compared to *humanities*.

Finding: In comparison with *STEM* Stack Exchange instances, *humanities* Stack Exchange instances are more reliant on cross-excitation by power users to address questions by both types of users. We observe more power user centric interactions in Stack Exchange instances in the *humanities*, while activity in *STEM* Stack Exchange appears more focused on casual users. Higher long-term self-excitation by casual users in *STEM* instances indicates stronger interactions between casual users. In turn, casual user activity is less dependent on power users in these instances. Overall, this finding suggests the existence of topic-dependent user type structures, which can be cast as measurable goals for community managers.

5 EVALUATION

In this section, we assess whether differences in excitation effects we observe in the evolution of *growing* vs. *declining* (*STEM* vs. *humanities*) instances result by chance or if there is some causal link between excitation, as measured with the Hawkes processes,



(a) Permuted Cross-Excitation of Questions by Power Users on Answers by Power Users

(b) Permuted Cross-Excitation of Questions by Casual Users on Answers by Casual Users

(c) Permuted Self-Excitation of Answers by Casual Users

(d) Permuted Cross-Excitation of Questions by Casual Users on Answers by Power Users

Figure 6: Permuting event sources destroys observed excitation effects. We illustrate the temporal evolution of selected cross-excitation effects of multivariate Hawkes process models fitted to both the original (solid lines) as well as the permuted event streams (dashed lines) of the *growing* vs. *declining* and *STEM* vs. *humanities* comparisons of Stack Exchange instances. The permuted event streams in the colored regions of this Figure feature only few of the differences of the original event streams, and, for differences that remain (e.g. of Figure (c)), they are of perceptibly lower magnitude. Hence, the absence of effects in permuted event streams strengthen the significance of our main findings.

and community growth (topical focus). Moreover, we evaluate the sizes of the observed effects by quantifying their impact on the future user activity.

Comparison of activity distributions. While the comparison of *growing* vs. *declining* instances aims to distinguish excitation effects in instances of increasing vs. decreasing and thus different total activity volumes, the *STEM* vs. *humanities* comparison is intended towards providing decoupled effects, which ideally should not be confounded with the excitation effects of *growing* vs. *declining* instances. However, in the *STEM* vs. *humanities* instance comparison, we highlight long-term self-excitation in answers by casual users, an effect which could be similar to the late stage self-excitation of *growing* vs. *declining* instances. Furthermore, if *humanities* instances simply featured overall higher answer-based activity levels by power users than in *STEM* instances, power users would also likely react stronger to questions by casual users, as opposed to them being an inherently more important backbone to questions by casual users.

Hence, we verify if the total answer-based activity distributions of both user types are similar in *STEM* and *humanities* instances. We compare the sample distributions of answers-based activity by power (and separately casual) users in *STEM* vs. *humanities* instances with the Kolmogorov-Smirnov two-sample test for their equality. As this test results in a p-value of 0.3855 (0.2305) for power (casual) users' activity distributions, we conclude there is not enough evidence to reject the null hypotheses of the probability distributions being equal at all usual significance levels. In turn, this test result indicates that the power and casual users' activity distributions are comparable, which supports our finding regarding the importance of the role power (casual) users play in *humanities* (*STEM*) Stack Exchange instances.

Permutation tests. To assess the significance of the excitation effects we conduct the following permutation test. First, we randomly permute the association of event types (questions by power users, questions by casual users, answers by power users and answers by casual users) to the corresponding time stamps per time window. This procedure keeps the amount of events per event type constant, but destroys the temporal connection between event types. Then, we refit the multivariate Hawkes processes over windows of these permuted event streams, and we repeat these two steps 100 times. Finally, we compare the difference in mean Hawkes process parameter values fitted on the permuted event streams to the original ones. If there is a notable difference between them, then this indicates that growth (or the evolution of the topical instances) does not come about by chance, but that differences in self-excitation and cross-excitation between *growing* vs. *declining* (*STEM* vs. *humanities*) communities play an important role in their temporal evolution. We depict the result of these permutation tests in Figure 6, in which previously described differences between *growing* vs. *declining* and *STEM* vs. *humanities* Stack Exchange instances are all either remarkably weaker or non-existent. We arrive at similar results with the permutation tests on the other effect not included in Figure 6 (namely casual user self-excitation, in the *STEM* vs. *humanities* comparison). Inspired by Chandrasekharan et al.'s [6] quantification of differences in permutation test distributions, we numerically summarize our permutation tests with a comparison of the absolute difference of *growing* vs. *declining* (*STEM* vs. *humanities*) Hawkes process parameter values fitted on the original event streams with the distribution of absolute differences in parameter values obtained on permuted event streams. If the difference in original values is extreme in relation to the distribution of permuted values in the effects' time spans, then this is further evidence the effects we observe are unlikely to arise by chance. We quantify "extreme" with the p-value, in this case the proportion of values from the permuted difference distribution greater than the original difference. Over the quarters per effect time span, almost all p-values are smaller than or equal to 0.01[3]. Thus, we find the existence of a weak causal link between excitation effects in Stack Exchange instances and their temporal evolution in terms of activity volume.

[3]Single exception: Early Power User Cross-Excitation in quarter one (p-value 0.04).

Table 2: Kolmogorov-Smirnov (K-S) distance between predicted and real interevent times, per predicted quarter, effect type, Hawkes process dimension and model variant. Lower K-S values are better. Distance values marked with an asterisk correspond to not rejecting equality of simulated and real interevent times. The Full model produces forecasts with lowest K-S distances and highest number of non-significant distances. Thus, as the Excitation Effects Removed model features higher K-S distances than the Full model, we find all excitation effects are important for prediction. To quantify the importance of each excitation effect, we observe removing Late Stage Self-Excitation (■) in the Excitation Effects Removed model is most detrimental for prediction performance (cf. high values in mid-section of four rightmost columns). Hence, Late Stage Self-Excitation is most important for prediction, followed by Early Power User (■) and Late Casual User (■) Cross-Excitation.

		Early Power User Cross-Excitation					Late Casual User Cross-Excitation				Late Stage Self-Excitation			
	Prediction Quarter	**2**	**3**	**4**	**5**	**6**	**9**	**10**	**11**	**12**	**9**	**10**	**11**	**12**
Baseline	Answers by Power	0.26	0.26	0.29	0.27	0.24	0.24	0.25	0.23	0.24	0.24	0.25	0.23	0.24
	Questions by Power	0.3	0.28	0.25	0.25	0.24	0.23	0.24	0.23	0.24	0.23	0.24	0.23	0.24
	Answers by Casual	0.24	0.22	0.25	0.21	0.21	0.2	0.18	0.17	0.18	0.2	0.18	0.17	0.18
	Questions by Casual	0.23	0.16	0.18	0.13	0.12	0.12	0.14	0.13	0.12	0.12	0.14	0.13	0.12
Excitation Effects Removed	Answers by Power	0.15	0.13	0.14	0.12	0.13	0.1*	0.11*	0.11*	0.12	0.43	0.42	0.41	0.42
	Questions by Power	0.28	0.27	0.27	0.26	0.25	0.22	0.24	0.23	0.25	0.3	0.33	0.3	0.31
	Answers by Casual	0.22	0.12	0.16	0.15	0.16	0.15	0.14	0.13	0.14	0.42	0.4	0.39	0.4
	Questions by Casual	0.24	0.19	0.21	0.17	0.17	0.15	0.16	0.16	0.16	0.190	0.19	0.19	0.19
Full	Answers by Power	0.16	0.12	0.12	0.11*	0.11*	0.1*	0.1*	0.1*	0.11*	0.1*	0.1*	0.1*	0.11*
	Questions by Power	0.28	0.24	0.23	0.23	0.22	0.22	0.24	0.22	0.24	0.22	0.24	0.22	0.24
	Answers by Casual	0.23	0.11*	0.15	0.14	0.15	0.12	0.11*	0.11*	0.11*	0.12	0.11*	0.11*	0.11*
	Questions by Casual	0.25	0.16	0.18	0.13	0.13	0.12	0.13	0.13	0.12	0.12	0.13	0.13	0.12

Prediction experiment. To quantify the impact of the observed excitation effects on future activity, we design the following prediction experiment. For each three-month time window (quarter) and for each *growing* and *declining* Stack Exchange instance, we fit three variants of the Hawkes process, with the same four dimensions as previously: answers by power users, questions by power users, answers by casual users and questions by casual users. The Hawkes process variants we consider are (i) a multivariate baseline model (i.e., a multivariate Poisson process), consisting of only baseline excitation $\boldsymbol{\mu}$ (Baseline in Table 2), (ii) a reduced model where we fit a full Hawkes process model but set the model parameters corresponding to the observed excitation effect to zero for quarters in which we observe a given excitation effect (i.e., we set cross-excitation of power users to zero for quarters one to five to remove the effects of Early Power User Cross-Excitation effect, then we set cross-excitation of casual users to zero for quarters eight to eleven to remove Late Casual User Cross-Excitation effect, and finally we set self-excitation of all users to zero also for quarters eight to eleven to remove the effects of Late Stage Self-Excitation effect) (Excitation Effects Removed), and (iii) a full Hawkes process model as defined in Equation 2 (Full), which we fit in the same manner as when uncovering excitation effects.

Overall, Hawkes process-based models such as ours are suited to forecast event timings, as classical machine learning approaches cannot make such time predictions (cf. e.g. Kurashima et al. [19]). Hence, for each variant of the Hawkes process, each Stack Exchange instance, and for each quarter that we fit, we predict the next quarter's event times by simulating the fitted process 100 times. To assess the model's performance we first compute the distribution of interevent times as well as event counts in all dimensions for each simulated and for given observed event sequences. Then, for each simulated quarter, we compute the mean of the Kolmogorov-Smirnov (K-S) test statistic to compare distributions of interevent times (and the root mean squared error (RMSE) to compare the event counts) between simulated and real events. We list the K-S test distance values for each predicted quarter in Table 2. We highlight, with an asterisk, values of the K-S test distance which correspond to not rejecting the hypothesis of equality of simulated and real interevent times at all usual significance levels.

We observe best K-S distance values overall for the Full model, indicating the overall importance of observed excitation effects at every developmental stage of a Stack Exchange instance for prediction experiments. Moreover, the very poor performance of the Excitation Effects Removed model, at times worse than the Baseline model, reinforces the importance of the effects we found for modeling and prediction. However, there seems to be a difference in the impact of different observed effects on the prediction performance. In particular, removing two cross-excitation effects (i.e., Early Power User Cross-Excitation and Late Casual User Cross-Excitation effect) from the models does not impair the performance of those models as strongly as the removal of Late Stage Self-Excitation effect. In Table 2's columns corresponding to the two Cross-Excitation effects, a comparison of predictions by the Excitation Effects Removed model with the corresponding predictions by the Full model reveals their differences in K-S distances lie in the interval $[-0.01, 0.04]$. In these cases, the Full model has only slightly better performance. On the other hand, the impact of the Late Stage Self-Excitation effect dramatically impairs the performance of the Excitation Effects Removed model. The differences between K-S distances in this case (cf. predictions by the Full model and the Excitation Effects Removed model in the Late Stage Self-Excitation columns of Table 2) range from 0.06 to 0.32, indicating a larger effect size of self-excitation than that of cross-excitation.

To further validate these results we perform another prediction experiment with a fourth variant of Hawkes processes. In this variant, we fit self-excitation only models by setting all cross-excitation parameters to zero. These additional experiments with a self-excitation model confirm previous observations: A model with only self-excitation achieves performances (as measured by

the K-S distance and by the event count RMSE) in general on par with those of the Full model and, in the Late Stage Self-Excitation effect, even surpassing its performance slightly.

For all model variants we come to comparable conclusions when measuring the RMSE between simulated and real event counts. Limited by space, we summarize these results: The average RMSE of the Full model is 638.17 events, an improvement of 59.91% (43.09%) upon the Excitation Effects Removed (Baseline) model.

6 LIMITATIONS

Although we experimented with slightly different percentiles in the user type distinction and the instance characterization and obtained qualitatively similar results, we recognize those are arbitrary thresholds, which impact the results if changed significantly. Our results are more robust to changes in window size of the activity event stream of the Q&A communities (e.g. to two, four or five months), since this hyperparameter controls for the granularity of our results. Nevertheless, the Hawkes process model itself could include time-varying parameters, as an alternative to this repeated fitting procedure we apply over fixed time windows.

The effect time spans we propose, namely a one-and-a-half-year-long early stage and a late stage starting in the last quarter of the second year of a Q&A community, stem from our empirical observations of large differences in excitation in specific temporal segments in the community comparisons. Pinpointing exact transition dates is beyond the scope of this work, as we focus on learning temporal user excitation effects.

We acknowledge that mapping each user's questions and answers event streams to a Hawkes process dimension may be a more realistic model. However, we argue that such a model would suffer from sparsity, high dimensionality and higher computational cost. Further, such a model might also not improve the excitation effect characterization, as it would also struggle with distinguishing sources of self- and cross-excitation in high-activity regimes.

Note that we avoid a discussion of how casual users become power users with our characterization of power users as the most active each month, regardless of their histories. We believe engagement reward systems such as badges play an important role in casual user's development in particular and user excitation in general [20], but we leave a detailed investigation of the role of reward systems on excitation effects for future work.

We also caution that our work indicates a *temporal* link between (i) specific community structures in terms of user types and their excitation and (ii) the overall development of activity volume in a Stack Exchange community. This work does not establish causality.

7 RELATED WORK

Research on Q&A communities. There is a considerable amount of authors [1, 10, 13, 24, 33, 41] analyzing the roles different types of users play in Web communities such as Q&A websites. In addition, several authors surveyed the motivation and behaviour of individual users [17, 26] of Q&A communities. While Mamykina et al. [24] and Furtado et al. [13] concentrate on uncovering and studying the roles of user types present in thriving Q&A communities, Danescu-Niculescu-Mizil et al. [10] and Yang et al. [41] explicitly focus on specific user types in Web communities and the user types' static and temporal characteristics. More broadly, Yang et al.'s work is part of a larger body of literature [28, 30, 46] on identifying experts in Q&A websites and characterizing their behavior. Our work leverages a comparable user type characterization to infer properties about the temporal evolution of communities themselves.

In approaches methodologically related to Zhang et al.'s [46], multiple authors [3, 25, 31, 38, 39, 43] study evolution dynamics of Web communities by relying on an explicit description of networks underlying a given Web community, and these networks often serve as a basis for dynamical systems models of the communities. In their study of Quora, another Q&A website, Wang et al. [39] analyze the role different social network structures play in Quora's community growth. Ribeiro [31] and Walk et al. [38] model users and activity in diverse Web communities including Q&A communities, with the former focusing on growth and decline of communities and the latter on the model's implications for self-sustainability in a community's activity. Matsubara et al. [25] and Zang et al. [43] study information diffusion and growth dynamics of Web communities.

Similarly to Matsubara et al.'s, Walk et al.'s and Zang et al.'s work, in this paper we also model growth and interaction dynamics of Q&A communities, but we do not assume an underlying network. We focus rather on excitation between groups of users, which we distinguish not on their expertise but on their overall activity levels. Furthermore, by encoding community lifecycles in Hawkes processes fitted to sequences of time windows, we extend the empirical discussion of Web community lifecycles [15, 42] and the critical mass literature [29, 32] to the Q&A community domain with measurable results.

Applications of Hawkes processes. Hawkes processes and their variations, as models for event streams with unequally spaced events in time, have found wide application in literature on different aspects of Web phenomena [12, 16, 36, 37, 44, 47, 48]. One such topic regards content popularity dynamics, in particular how to predict the influence of internal and external aspects of activity in social networks [12] and reshare popularity of items on the Web [47]. To infer causal links between users and user influence from user activity in social networks, Ver Steeg and Galstyan [37], Iwata et al. [16] and Zhou et al. [48] propose point process-related approaches, which cope with high dimensionality in number of users. Further, Upadhyay et al. [36] model the crowdlearning process of Stack Overflow users and characterize different user types by their expertise and learning curves. The work by Zang et al. [44] models and predicts the growth dynamics of individuals' ties in social networks and predicts its evolution.

Our work draws inspiration and methodological know-how from all above mentioned papers to expand on a topic closely related to Zang et al.'s: the development of not just the relatively small circle of an individual's social ties, but of excitation and interaction of user groups in Q&A communities. Furthermore, we contribute, to the growing body of work on fitting Hawkes process kernels [5, 22, 40, 49], a parsimonious Bayesian hyperparameter optimization method for fitting the decay parameter of exponential kernels in Hawkes processes. Finally, our extension of this fitting method to non-stationary multivariate event streams enables the extraction of temporal excitation effects from Q&A communities.

8 CONCLUSIONS

Summary. In this work, we modeled self- and cross-excitation in Q&A communities along several dimensions, including activity type, user engagement level, growth path and topical focus of a given community. We approached this task by fitting multivariate Hawkes processes to stationary temporal segments of Q&A communities' activity volumes. We found stronger cross-excitation of power (casual) users in early (late) stages of growing communities when compared to communities with declining total activity. Further, in growing communities, we observed self-excitation dominates in the long-term. Moreover, we uncovered strong long-term cross-excitation by power (casual) users in Q&A communities dedicated to topics in the fields of the humanities (STEM). We validated the presence of these excitation effects with statistical and permutation tests and we quantified their strength via prediction tasks.

Implications. Our work can support Q&A community managers in their ambition to promote sustainable community structures. To jumpstart community growth in its first six months, engaging a core of power users, for example in community building efforts, appears to be of crucial importance. In the medium- to long-term, we find community developers should carefully monitor and foster participation rather by casual users. While literature on critical mass in Web communities [29, 32] and studies on the user mix in Wikipedia [18, 34] also support this recommendation, we can afford further advice, as our casual cross-excitation analysis specifically underlines the importance of interaction between casual users. In practice, we believe adjusting reward or badge systems to encourage contributions by casual users, perhaps by welcoming newcomers or by easing their adjustment to community rules, would be of value to community development. Furthermore, community managers, which have not observed a surge in self-excitation by the third year of their communities, may have reason to concern over growth. Such excitation effects should be carefully monitored, as Q&A community growth may come at the cost of other community parameters [11]. Furthermore, our results indicate concrete implementations of these suggestions should depend on community topic, as it impacts excitation effects. Overall, our findings thus highlight the impact of timing in the user mix development.

Future work. Comparing other Q&A communities would allow to further generalize the results we obtained on Stack Exchange communities. Our work can be extended to uncover excitation effects in other domains, such as of Q&A instances in other languages or of other contribution types (e.g. open-ended vs. focused question), as our proposed approach is generic and can be readily extended.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable feedback on the manuscript. Tiago Santos is a recipient of a DOC Fellowship of the Austrian Academy of Sciences at the Institute of Interactive Systems and Data Science of the Graz University of Technology.

REFERENCES

[1] Lada A. Adamic, Jun Zhang, Eytan Bakshy, and Mark S. Ackerman. 2008. Knowledge Sharing and Yahoo Answers: Everyone Knows Something. In *Proceedings of the 17th International Conference on World Wide Web (WWW'08)*. ACM, 665–674.

[2] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2013. Steering User Behavior with Badges. In *Proceedings of the 22nd International Conference on World Wide Web (WWW'13)*. ACM, 95–106.

[3] Roy M Anderson, Robert M May, and B Anderson. 1992. *Infectious Diseases of Humans: Dynamics and Control*. Vol. 28. Wiley Online Library.

[4] James Bergstra, Daniel Yamins, and David Cox. 2013. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. In *Proceedings of the 30th International Conference on Machine Learning (ICML'13)*. 115–123.

[5] Qi Cao, Huawei Shen, Keting Cen, Wentao Ouyang, and Xueqi Cheng. 2017. DeepHawkes: Bridging the Gap between Prediction and Understanding of Information Cascades. In *Proceedings of the ACM on Conference on Information and Knowledge Management (CIKM'17)*. ACM, 1149–1158.

[6] Eshwar Chandrasekharan, Umashanthi Pavalanathan, Anirudh Srinivasan, Adam Glynn, Jacob Eisenstein, and Eric Gilbert. 2017. You Can't Stay Here: The Efficacy of Reddit's 2015 Ban Examined Through Hate Speech. In *Proceedings of the 2017 Conference on Computer Supported Cooperative Work (CSCW'17)*. ACM, 31–53.

[7] David Roxbee Cox and Valerie Isham. 1980. *Point processes*. Vol. 12. CRC Press.

[8] Daryl J Daley and David Vere-Jones. 2003. *An Introduction to the Theory of Point Processes: Volume I: Elementary Theory and Methods*. Springer Science & Business Media.

[9] Daryl J Daley and David Vere-Jones. 2008. *An Introduction to the Theory of Point Processes: Volume II: General Theory and Structure*. Springer Science & Business Media.

[10] Cristian Danescu-Niculescu-Mizil, Robert West, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. No Country for Old Members: User Lifecycle and Linguistic Change in Online Communities. In *Proceedings of the 22nd International Conference on World Wide Web (WWW'13)*. ACM, 307–318.

[11] Himel Dev, Chase Geigle, Qingtao Hu, Jiahui Zheng, and Hari Sundaram. 2018. The Size Conundrum: Why Online Knowledge Markets Can Fail at Scale. In *Proceedings of the 2018 World Wide Web Conference (WWW'18)*. ACM, 65–75.

[12] Mehrdad Farajtabar, Nan Du, Manuel Gomez Rodriguez, Isabel Valera, Hongyuan Zha, and Le Song. 2014. Shaping Social Activity by Incentivizing Users. In *Advances in Neural Information Processing Systems (NIPS'14)*. 2474–2482.

[13] Adabriand Furtado, Nazareno Andrade, Nigini Oliveira, and Francisco Brasileiro. 2013. Contributor profiles, their dynamics, and their importance in five q&a sites. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW'13)*. ACM, 1237–1252.

[14] Alan G Hawkes. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* (1971), 83–90.

[15] Alicia Iriberri and Gondy Leroy. 2009. A life-cycle perspective on online community success. *ACM Computing Surveys (CSUR)* 41, 2 (2009), 11.

[16] Tomoharu Iwata, Amar Shah, and Zoubin Ghahramani. 2013. Discovering Latent Influence in Online Social Activities via Shared Cascade Poisson Processes. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13)*. ACM, 266–274.

[17] Jiahua Jin, Yijun Li, Xiaojia Zhong, and Li Zhai. 2015. Why Users Contribute Knowledge to Online Communities: An Empirical Study of an Online Social Q&A Community. *Information & management* 52, 7 (2015), 840–849.

[18] Aniket Kittur, Ed Chi, Bryan A. Pendleton, Bongwon Suh, and Todd Mytkowicz. 2007. Power of the Few vs. Wisdom of the Crowd: Wikipedia and the Rise of the Bourgeoisie. In *Proceedings of the Alt.CHI Conference on Human Factors in Computing Systems (Alt.CHI'07)*.

[19] Takeshi Kurashima, Tim Althoff, and Jure Leskovec. 2018. Modeling Interdependent and Periodic Real-World Action Sequences. In *Proceedings of the 2018 World Wide Web Conference (WWW'18)*. ACM, 803–812.

[20] Tomasz Kusmierczyk and Manuel Gomez-Rodriguez. 2018. On the Causal Effect of Badges. In *Proceedings of the 2018 World Wide Web Conference (WWW'18)*. ACM, 659–668.

[21] Kenneth Levenberg. 1944. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics* 2, 2 (1944), 164–168.

[22] Sha Li, Xiaofeng Gao, Weiming Bao, and Guihai Chen. 2017. FM-Hawkes: A Hawkes Process Based Approach for Modeling Online Activity Correlations. In *Proceedings of the ACM on Conference on Information and Knowledge Management (CIKM'17)*. ACM, 1119–1128.

[23] Thomas Josef Liniger. 2009. *Multivariate Hawkes Processes*. Ph.D. Dissertation. ETH Zurich.

[24] Lena Mamykina, Bella Manoim, Manas Mittal, George Hripcsak, and Björn Hartmann. 2011. Design Lessons from the Fastest Q&A Site in the West. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'11)*. ACM, 2857–2866.

[25] Yasuko Matsubara, Yasushi Sakurai, B Aditya Prakash, Lei Li, and Christos Faloutsos. 2012. Rise and Fall Patterns of Information Diffusion: Model and Implications. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'12)*. ACM, 6–14.

[26] Meredith Ringel Morris, Jaime Teevan, and Katrina Panovich. 2010. What Do People Ask their Social Networks, and Why?: A Survey Study of Status Message Q&A Behavior. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'10)*. ACM, 1739–1748.

[27] Yosihiko Ogata. 1981. On Lewis' Simulation Method for Point Processes. *IEEE Transactions on Information Theory* 27, 1 (1981), 23–31.

[28] Aditya Pal, Shuo Chang, and Joseph A Konstan. 2012. Evolution of Experts in Question Answering Communities. In *Proceedings of the Sixth International Conference on Web and Social Media (ICWSM'12)*. 274–281.

[29] Daphne R Raban, Mihai Moldovan, and Quentin Jones. 2010. An Empirical Study of Critical Mass and Online Community Survival. In *Proceedings of the 2010 Conference on Computer Supported Cooperative Work (CSCW'10)*. ACM, 71–80.

[30] Fatemeh Riahi, Zainab Zolaktaf, Mahdi Shafiei, and Evangelos Milios. 2012. Finding expert users in community question answering. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*. ACM, 791–798.

[31] Bruno Ribeiro. 2014. Modeling and Predicting the Growth and death of membership-based websites. In *Proceedings of the 23rd International Conference on World Wide Web (WWW'14)*. ACM, 653–664.

[32] Jacob Solomon and Rick Wash. 2014. Critical Mass of What? Exploring Community Growth in WikiProjects. In *Proceedings of the Eight International Conference on Web and Social Media (ICWSM'14)*. 476–484.

[33] Ivan Srba and Maria Bielikova. 2016. A Comprehensive Survey and Classification of Approaches for Community Question Answering. *ACM Transactions on the Web (TWEB)* 10, 3 (2016), 18.

[34] Bongwon Suh, Gregorio Convertino, Ed H Chi, and Peter Pirolli. 2009. The Singularity Is Not Near: Slowing Growth of Wikipedia. In *Proceedings of the 5th International Symposium on Wikis and Open Collaboration*. ACM.

[35] Behzad Tabibian, Isabel Valera, Mehrdad Farajtabar, Le Song, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. 2017. Distilling Information Reliability and Source Trustworthiness from Digital Traces. In *Proceedings of the 26th International Conference on World Wide Web (WWW'17)*. ACM, 847–855.

[36] Utkarsh Upadhyay, Isabel Valera, and Manuel Gomez-Rodriguez. 2017. Uncovering the Dynamics of Crowdlearning and the Value of Knowledge. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM'17)*. ACM, 61–70.

[37] Greg Ver Steeg and Aram Galstyan. 2012. Information Transfer in Social Media. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*. ACM, 509–518.

[38] Simon Walk, Denis Helic, Florian Geigl, and Markus Strohmaier. 2016. Activity Dynamics in Collaboration Networks. *ACM Transactions on the Web (TWEB)* 10, 2 (2016), 11.

[39] Gang Wang, Konark Gill, Manish Mohanlal, Haitao Zheng, and Ben Y Zhao. 2013. Wisdom in the Social Crowd: An Analysis of Quora. In *Proceedings of the 22nd International Conference on World Wide Web (WWW'13)*. ACM, 1341–1352.

[40] Hongteng Xu, Mehrdad Farajtabar, and Hongyuan Zha. 2016. Learning Granger Causality for Hawkes Processes. In *Proceedings of the 33rd International Conference on Machine Learning (ICML'16)*. 1717–1726.

[41] Jie Yang, Ke Tao, Alessandro Bozzon, and Geert-Jan Houben. 2014. Sparrows and Owls: Characterisation of Expert Behaviour in StackOverflow. In *User Modeling, Adaptation, and Personalization*. Springer, 266–277.

[42] Colleen Young. 2013. Community Management that Works: How to Build and Sustain a Thriving Online Health Community. *Journal of Medical Internet Research* 15, 6 (2013).

[43] Chengxi Zang, Peng Cui, and Christos Faloutsos. 2016. Beyond Sigmoids: The Nettide Model for Social Network Growth, and its Applications. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16)*. ACM, 2015–2024.

[44] Chengxi Zang, Peng Cui, Christos Faloutsos, and Wenwu Zhu. 2017. Long Short Memory Process: Modeling Growth Dynamics of Microscopic Social Connectivity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'17)*. ACM, 565–574.

[45] Achim Zeileis, Christian Kleiber, Walter Krämer, and Kurt Hornik. 2003. Testing and Dating of Structural Changes in Practice. *Computational Statistics & Data Analysis* 44 (2003), 109–123.

[46] Jun Zhang, Mark S Ackerman, and Lada Adamic. 2007. Expertise Networks in Online Communities: Structure and Algorithms. In *Proceedings of the 16th International Conference on World Wide Web (WWW'07)*. ACM, 221–230.

[47] Qingyuan Zhao, Murat A Erdogdu, Hera Y He, Anand Rajaraman, and Jure Leskovec. 2015. Seismic: A Self-Exciting Point Process Model for Predicting Tweet Popularity. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'15)*. ACM, 1513–1522.

[48] Ke Zhou, Hongyuan Zha, and Le Song. 2013. Learning Social Infectivity in Sparse Low-Rank Networks Using Multi-Dimensional Hawkes Processes. In *Artificial Intelligence and Statistics*. 641–649.

[49] Ke Zhou, Hongyuan Zha, and Le Song. 2013. Learning triggering kernels for multi-dimensional hawkes processes. In *Proceedings of the 30th International Conference on Machine Learning (ICML'13)*. 1301–1309.

Automatic Boolean Query Refinement for Systematic Review Literature Search

Harrisen Scells
The University of Queensland
Brisbane, Australia
h.scells@uq.net.au

Guido Zuccon
The University of Queensland
Brisbane, Australia
g.zuccon@uq.edu.au

Bevan Koopman
CSIRO
Brisbane, Australia
bevan.koopman@csiro.au

ABSTRACT

In the medical domain, systematic reviews are a highly trustworthy evidence source used to inform clinical diagnosis and treatment, and governmental policy making. Systematic reviews must be complete in that *all relevant* literature for the research question of the review must be synthesised in order to produce a recommendation. To identify the literature to screen for inclusion in systematic reviews, information specialists construct complex Boolean queries that capture the information needs defined by the research questions of the systemic review. However, in the quest for total recall, these Boolean queries return many non relevant results.

In this paper, we propose automatic methods for Boolean query refinement in the context of systematic review literature retrieval with the aim of alleviating this high-recall, low-precision problem. To do this, we build upon current literature and define additional semantic transformations for Boolean queries in the form of query expansion and reduction. Empirical evaluation is done on a set of real systematic review queries to show how our method performs in a realistic setting. We found that query refinement strategies produced queries that were more effective than the original in terms of six information retrieval evaluation measures. In particular, queries were refined to increase precision, while maintaining, or even increasing, recall — this, in turn, translates into both time and cost savings when creating laborious and expensive systematic reviews.

KEYWORDS

Systematic Reviews, Query Formulation, Boolean Queries, Query Transformations

ACM Reference Format:
Harrisen Scells, Guido Zuccon, and Bevan Koopman. 2019. Automatic Boolean Query Refinement for Systematic Review Literature Search. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313544

1 INTRODUCTION

Systematic reviews are highly reliable sources of evidence, created by synthesising all relevant studies in a comprehensive literature review for a highly focused research question. Medical systematic reviews are held to a particularly high standard because of the role they play in both evidence-based medicine and health policy as a whole. Systematic reviews both inform how medical professionals diagnose and treat patients, and how governing bodies decide upon health management and policies [14].

With the rise of web-based databases of medical literature, now containing tens of millions of medical citations (often in the form of abstract, title, MeSH headings and metadata, which reference full studies of, for example, randomised controlled trials), it is becoming increasingly difficult and highly costly to identify relevant literature to include in a systematic review [29]. For example, one of the most popular web-based databases, PubMed, contains approximately 28 million citations of medical studies. While recall is paramount in that all potentially relevant citations must be retrieved, precision in the case of systematic review literature retrieval is still critically important. Systematic reviews can cost upwards of a quarter of a million dollars [17] and can take several months to complete. A large portion of this cost can be attributed to the screening phase, where there are typically many false positives (i.e., citations retrieved which are not relevant to the research question of the systematic review). For systematic reviews of highly specific areas of medicine, finding relevant studies requires a highly specific query which cannot be too broad (which minimises the false positive rate). For broad studies such as scoping reviews, the query cannot be too specific as to limit the number of relevant citations that are not retrieved (i.e., minimising the false negative rate). This balance of potentially relevant and potentially non-relevant citations is currently managed by the intuition and knowledge of highly trained and skilled information specialists who are deeply familiar with the search systems in which they formulate their queries, and often, but not always, also with the topic of the systematic reviews they are formulating queries for.

The queries that information specialists formulate are highly complex Boolean queries. Systematic review literature search is dependent on, and has always used, Boolean retrieval for two reasons: (i) the Boolean query syntax enables specialists searchers to have fine-grained control over exactly which citations are retrieved, and (ii) the Boolean query which is published alongside the review, is used for reproducibility purposes; for example, when the recommendation of a systematic review is in doubt or a review needs to be updated. Ultimately, the Boolean query controls the outcomes and decisions reached by a systematic review.

This paper investigates automatic methods for refining queries used to retrieve literature for systematic reviews. Prior work by Scells and Zuccon has investigated syntactic transformations of the Boolean query, and have empirically shown that these could improve the effectiveness of the queries originally formulated by

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313544

expert information specialists [24]. In their approach, a query was modified at the Boolean query language level — for example, by changing an *OR* operator into an *AND* operator for a specific Boolean clause. These transformations did not add or remove the original query terms. They did not perform query expansion or reduction, apart from making use of, or removing, the Medical Subject Headings (MeSH) explosion operator (which allows for the retrieval of citations that contain MeSH concepts subsumed by the concept specified in the query). In this work, we investigate whether semantic transformations (query expansion and reduction by drawing upon semantically similar terms) can further improve the effectiveness of the Boolean queries. Specifically, the context of this work is to refine existing Boolean queries that have been created for the explicit purpose of systematic review literature search. We define a good query refinement process as one where fewer citations are retrieved, but the number of relevant citations stays the same (or with very minor, tolerable reductions). In Information Retrieval terms, this equates to increasing precision, while maintaining or even improving recall. In the case where recall is improved, note that in the case of systematic reviews, a number of studies that are used to synthesise results are often found *outside* the search process, either by snowballing references (i.e., finding relevant studies by examining the references of retrieved citations), or having prior knowledge of the existence of relevant studies.

The methods proposed in this paper are envisioned not to replace the experience and knowledge of human searchers, but instead, to assist them formulate more effective queries (e.g., explicitly recommending more effective queries). To investigate this topic, we pose the following research questions:

RQ1: Which types of transformations are the most effective at refining Boolean queries in systematic review literature search?
RQ2: What impact do unjudged documents have on the effectiveness of refined Boolean queries in systematic review literature search?

To answer **RQ1**, we first devised a set of strategies to generate semantic and syntactic transformations within the Query Transformation Chain Framework of Scells et al. [24]. These were then applied to the original Boolean queries in an iterative fashion so as to generate a set of transformed query candidates. The generated candidate queries were then represented within a feature space, which was then used to automatically select a transformed Boolean query that was predicted to improve over the original query (if no improvement was predicted, then original input query was retained). We then selected the top query from the set of candidates with a query candidate selector function which maximised a target evaluation measure. The effectiveness of the query refinement was determined by comparing the original queries with the ones that were automatically refined.

To answer **RQ2**, we perform evaluation by considering the unjudged documents that may be retrieved by the automatically refined queries. We do so by considering two methods for computing which portion of the unjudged citations (i.e. the residual) should be used for computing relevance. The methods outlined for **RQ1** and **RQ2** are explained in detail in Section 3.

2 RELATED WORK

The modification or transformation of a query, such as query expansion and query reduction, but not limited to any semantic and syntactic modification (e.g., via synonyms of a term, adding related terms, removing unnecessary terms, correcting spelling mistakes) has been shown to significantly improve the effectiveness of a query [3, 6, 18, 30]. Previous work, however, has focused almost exclusively on non-Boolean queries (i.e., keyword queries).

In the professional search setting, Kim et al. [10] was able to generate Boolean queries using a decision tree method and then automatically suggest queries. A decision tree is generated by selecting unigrams from the top-k pseudo-relevant documents and nodes in the tree are selected with a probability of that term appearing or not in the pseudo-relevant set (leading to conjunction and negation). Queries are then suggested by training a learning to rank model on query performance predictor (QPP)-based features. Both general-purpose QPPs and Boolean query-specific QPPs are used. When general-purpose QPPs are used, only the query terms that are associated with conjunctions are considered. In the context of systematic reviews, Scells et al. [22] have shown that QPPs were not effective in predicting the performance of queries in this domain.

Karimi et al. [9] have found that simplifying field restrictions and operators, and removing clauses from keyword queries (not Boolean) provides significant improvements in recall for systematic review queries, while degrading precision only slightly. Keyword queries, however, have a major limitation: they do not capture complex information needs by restricting sets of documents; i.e., they operate in a best-match setting where ranking is important[1]. Another perspective is that keyword queries do not allow the information specialist explicit control over the set of documents retrieved.

Because Boolean queries are so deeply engrained in the scientific methodology of constructing systematic reviews, we focus on modifications or transformations that can be applied to Boolean queries. Scells et al. have also investigated the modifications to queries in a systematic review setting in two studies. One study found that integrating clinical labels into the retrieval model (i.e., annotating both queries and documents) significantly improved precision while slightly degrading recall [25]. The other study focused on improving existing Boolean queries with automatic candidate generation and selection [24]. While Scells et al. [24] use only syntactic transformations, we advance the state-of-the-art by integrating syntactic *and* semantic transformations of queries and then automatically selecting the best candidate.

3 METHODOLOGY

3.1 Query Transformation Chain Framework

A query transformation chain is the repeated application of one or more transformations to a query which results in a set of variations of the original query. A simplified example of a query transformation chain is shown in Figure 1, where three transformations (τ_1, τ_2, and τ_3) are applied to three clauses (c_i) of the original query q. This process of applying transformations to clauses of a new query continues until the stopping criteria of the candidate selection function is met. After the transformations are applied, a final, rewritten

[1] Although another task in this domain – screening prioritisation – would benefit from this.

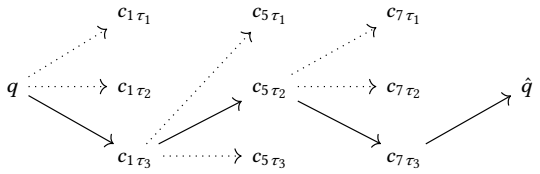


Figure 1: A query q transformed via a query transformation chain into a new query $\hat{q}$. Three transformations (τ_1, τ_2, and τ_3) are applied to clauses (c_i) of a query at each point in the chain. The path of each transformation chosen through the chain is indicated by a solid line. Only the path of transformations taken for $\hat{q}$ are shown.

query $\hat{q}$ is produced. A query chain can either be used to generate all possible variations of a query given the set of transformations, or at each application of transformations, a smaller subset of queries can be chosen. This candidate selection process is described in detail in Section 3.4.

3.2 Query Clause Transformations

Scells et al. [24] considered only syntactic transformations of a query. We extend that work by integrating several semantic transformations. Syntactic transformations are those that modify aspects of the query unrelated to the structure of a query: logical operators, or the fields keywords are restricted to. The other form of transformation is semantic, whereby aspects of a query are modified structurally (through the addition or removal of clauses). In this work, we use six types of transformations: two syntactic, and four semantic. In addition, four of the transformations can be applied to non-atomic clauses and three can be applied to atomic clauses. The six transformations we use in this work are presented in Table 1 with the classification (syntactic or semantic) and the applicability of the transformation (to atomic or non-atomic clauses). Each transformation is described in detail below:

(1) **Logical Operator Replacement** The logical operator replacement transformation syntactically replaces the Boolean operators in a query. For example, `(A **AND** B)→(A **OR** B)`. This transformation only considers `AND` and `OR` operators.
(2) **Field Restrictions** The field restrictions transformation syntactically modifies the fields an atomic clause is restricted to in Boolean search. For example,

```
(cancer[Title] AND lungs[Title])
                ↓
(cancer[Title/Abstract] AND lungs[Title])
```

This transformation produces the following modifications to fields: `title` to `abstract`, `title` to `title & abstract`, `abstract` to `title`, `abstract` to `title & abstract`, `title & abstract` to `title`, and `title & abstract` to `abstract`.
(3) **MeSH Explosion** MeSH is a medical ontology organised in a tree structure. Explosion is the subsumption of all nodes underneath a particular term node in the ontology. MeSH explosion can be considered a semantic transformation in that it implicitly adds terms to a query. This transformation toggles if a MeSH term should be exploded or not. Systematic review query languages all support some way for explosion to be toggled, normally in the retrieval model, thus not explicitly modifying the semantics of a query.
(4) **MeSH Parents** The MeSH parents transformation acts in the opposite direction of the explosion transformation: moving up one level in the tree (if possible), thus selecting a broader MeSH term. For example, if the MeSH term is `Skull Neoplasms`, then this transformation selects the parent term `Bone Neoplasms`.
(5) **Clause Removal** The clause removal transformation semantically modifies a query by removing a non-atomic clause from a query. This transformation will remove atomic clauses from a non-atomic clause until there are no more atomic clauses left (at which point the non-atomic clause that groups them is removed).
(6) **cui2vec Expansion** The cui2vec transformation is a semantic query expansion transformation that uses a pre-trained set of clinical concept embeddings [1] to find related keywords.[2] We first map keywords to concepts using QuickUMLS [27], restricting matches to preferred candidates. Next we find the top-k similar concepts (which are then used as expansions E) using the clinical concept embeddings via cosine similarity; scores are then normalised using the softmax function. A mapping derived by Jimmy et al. [7] is used to map concepts to their most common string. In our experiments, we expand to a maximum of five candidates. To integrate the new expansions into the Boolean query, we replace the original keyword with a logical `OR` operator that groups the expansions. The fields of the expansions are inherited from the original keyword. Finally, the original keyword that was used for expansion is then added to the atomic clauses in the new non-atomic clause.

Table 1: Classification (semantic or syntactic) and Applicability (transformation can be applied to atomic or non-atomic clauses) of transformations used in this work.

Transformation	Classification	Applicability
Logical Operator Replacement	Syntactic	Non-Atomic
Field Restrictions	Syntactic	Atomic
MeSH Explosion	Semantic	Atomic
MeSH Parents (new)	Semantic	Atomic
Clause Removal (new)	Semantic	Atomic
cui2vec Expansion (new)	Semantic	Atomic

3.3 Query Candidate Features

The ranking of query candidates requires discriminative features in order to select the most effective query. We use 39 features in total, described in Table 2. Two notable features which we consider to be novel are Δ, which computes the difference between measurements indicated in Table 2 with †, and ω_i which represents the ith transformation applied to a query in the query chain process.

[2] As the name suggests, the embeddings are for clinical *concepts* (rather than terms), identified as **Concept Unique Identifiers** (CUI) from the UMLS Metathesaurus. A single CUI may represent different names or *strings* in different source vocabularies of the UMLS Metathesaurus.

For performance and effectiveness reasons we do not include general purpose QPPs (Query Performance Predictors) as in similar work [10]. This is because: (i) most QPP measurements rely on costly retrieval statistics which becomes prohibitive with the number of queries that are generated, and (ii) previous work has shown that QPPs are often not correlated with retrieval effectiveness [5, 22] and thus may hamper the quality of ranking.

3.4 Query Candidate Selection

Query candidate selection is the process by which the next candidate to transition to in the query chain is chosen. The formalisation of candidate selection is described as follows. Following the generation of candidate queries $\hat{\mathbb{Q}}_q$ from the initial query q, the next candidate q^* is selected as the next transition in the query chain via the maximisation of function $f(\hat{q})$ where:

$$q^* = \underset{\hat{q} \in \hat{\mathbb{Q}}_q}{\text{argmax}}\, f(\hat{q}) \tag{1}$$

The candidate generation and selection process is repeated until the stopping criteria is met. Here, we define the stopping criteria with two objectives: (i) the query chain reaches a depth of five as in previous research [24], and (ii) the number of retrieved citations for a candidate query is zero.

When generating queries for training purposes, we only considered the stopping criteria when the depth of a chain reached the maximum length of five. We did this so that we would have sufficient negative training examples. In practice, this resulted in a severely imbalanced training set with the majority of examples being negative and only a small portion positive. In training models to maximise the candidate selection objective function, we discovered empirically that removing the majority of negative examples (i.e., rebalancing the positive and negative examples via negative sub-sampling) increased the effectiveness of ranking.

The maximisation function $f(\hat{q})$ can be learnt as a machine learning problem. In this work, we learn $f(\hat{q})$ as both a learning to rank task (similarly as in [24]) and a nearest neighbour task (new to this work). For both tasks, we train a different model for each evaluation measure that we seek to optimise. For the learning to rank task, we use DART: an ensemble of boosted regression trees which incorporates dropouts [21]. In our empirical testing we found this method to be the best for this task when compared to other state of the art techniques such as LambdaMART. For the nearest neighbour task, we create a model which records the most effective query in each topic (given an evaluation measure), and the distance to and reduction in effectiveness for all other queries in that topic. For an unseen set of candidate queries, each target query in the set is ranked by minimising the distance and maximising the score of the queries in the aforementioned model.

3.5 Sampling Query Candidates

In generating training data for automatic query candidate selection methods, we encountered an exponential growth in queries generated. This growth is computationally undesirable: for example, if on average 100 candidates are generated for a query, and the total length of a chain is 5 then a total of 10^9 queries would be generated (n^d where n is the number of candidates and d is the depth). Additionally, if on average it takes 30 seconds to generate a set of queries and features, then for that query it would take over 950 years to complete. For this reason, we sample queries to cut the total number of queries that are included for training. Note that the $O(n^d)$ complexity does not affect testing as we are not exploring the space of transformations for a query, rather we are ranking candidates, selecting and generating transformations for the top-1. Our sampling approach is as follows: for a seed query q, we generate a set of candidate queries $\hat{\mathbb{Q}}_q$; from this set, we randomly sample a minimum of 20 candidate queries and if there are more than 20 queries generated, we sample a further 10%. We then cut further by removing any identical queries that have already been processed. Finally, of the queries that are generated from the previous step, we randomly sample 10% of these new queries and repeat the process.

3.6 Evaluation

We perform evaluation as a standard ad-hoc search task. Relevance assessments are obtained from the studies that are reported in each systematic review as *included* (relevant), and *excluded* (non-relevant). Queries obtained through query refinement may retrieve citations that were not assessed by the reviews (i.e., unjudged); thus, relevance assessments are incomplete. At the first stage we assume all citations which were not judged as non-relevant. Remember, however, that unlike in traditional ad-hoc information retrieval evaluation, no pooling using a large variety of systems was executed: only one query and one system contributed to the relevance assessments — other, non retrieved but relevant citations may very well exist. Thus, later in the analysis of the results, we use two heuristics to bound the effect of unjudged documents on the effectiveness of the studied methods. To this aim, we use and further extend the intuition of residual analysis described for the rank bias precision measure [19]:

3.6.1 Optimistic Residual. The first heuristic consists of assuming all unjudged citations are relevant, as done for the computation of the heuristic in rank bias precision [19]. This value provides the upper bound on the value of the evaluation measure that would be obtained if all citations were assessed.

3.6.2 Maximum Likelihood Residual. The second heuristics attempts to provide a stricter bound than the optimistic residual. To do this, the relevance of an unjudged citation is set according to its probability of being relevant. To estimate this probability, we use the maximum likelihood estimation computed on the set of judged documents:

$$P(\texttt{relevant}|d) = \frac{|\texttt{relevant}|}{|\texttt{relevant}| + |\texttt{non-relevant}|}$$

This could be considered as a rough approximation of the number of relevant and non-relevant citations that would exist, if the relevance assessments for a query were complete. Intuitively, it provides a balance between the existing pessimistic view of unjudged citations and the aforementioned optimistic view of unjudged citations.

4 EXPERIMENTAL SETUP

Experiments were performed on a test collection of 125 systematic reviews [26]. The CLEF2017 TAR collection [8] was not used

Table 2: Features used in the candidate selection task. Applicability denotes to which types of clauses a feature is applicable to: *Atomic* refers to clauses which consist of a single keyword, *Non-Atomic* refers to clauses which group several atomic clauses with a logical operator, and *Both* indicates that the feature can be applied to either type of clause.

Feature	Description	Applicability
Depth	How deep in the query the transformation was made.	Both
Clause Type	The type of clause the transformation was applied to (atomic/non-atomic).	Both
Children Count	Number of sub-clauses a non-atomic clause has.	Non-Atomic
Transformation Type	The type of transformation (Table 1) made.	Both
Logical Operator Replacement Type	The type of operator this type of transformation modified.	Non-Atomic
MeSH Depth	The depth that the MeSH term was exploded from in the MeSH ontology.	Atomic
MeSH Parent	Whether a MeSH parent transformation is made.	Atomic
Restriction Type	Which type of field restriction was made when this transformation was applied.	Atomic
Clause Removal	Whether a clause removal transformation has been made.	Atomic
cui2vec	Whether a cui2vec Expansion has been made.	Atomic
cui2vec Expansions	The number of expansions that were made when this transformation was applied.	Atomic
MeSH Exploded	If the MeSH heading in an atomic clause is exploded or not.	Atomic
Truncated	If an atomic clause has a character replacement expression (e.g., wildcard).	Atomic
Retrieval Size†	Total number of citations retrieved by the query.	Both
# Boolean Clauses†	Total number of Boolean clauses in the query.	Non-Atomic
# Boolean Keywords†	Total number of atomic clauses (keywords) in the query.	Non-Atomic
# Truncated†	Total number of atomic clauses that have a truncated term.	Non-Atomic
# Fields†	Total number of fields in the query.	Both
# MeSH Keywords†	Total number of non-atomic clauses that contain a MeSH term.	Non-Atomic
# MeSH Exploded†	Total number of non-atomic clauses that have an exploded MeSH term.	Non-Atomic
# MeSH Non-Exploded†	Total number of non-atomic clauses that do not have an exploded MeSH term.	Non-Atomic
Average MeSH Depth†	The average depth MeSH terms in the query appear.	Non-Atomic
Maximum MeSH Depth†	The maximum depth MeSH terms in the query appear.	Non-Atomic
Δ	The difference of a feature between q and $\hat{q}$, where applicable (†).	Both
ω_i	n additional features that correspond to the sequence of transformations made.	Both

because that collection contains a smaller number of topics (50 in total) and therefore less training data. To perform the experiments, we used a framework to construct pipelines for each stage of the experiments [23]. These pipelines and the code for the experiments are made available online.[3]

4.1 Query Candidate Generation

In order to train a model to select candidate queries, training data is required. We achieve this by exploring the space of queries using the transformation processes described in Section 3.2 up to a depth of 5. The difference here, however, is that rather than choosing one query as a candidate query, all queries are chosen. That is, rather than choosing one query to transition to in the chain, we explore all possible transitions in order to generate training data. Because this process is exponential in nature, we sample using the method described in Section 3.5. In terms of training statistics, on average 7968 queries were generated from a seed query. The highest number of queries generated for a topic was 129,282 and the lowest was 70. The total number of training data points was 717,160.

[3] https://github.com/ielab/www19refinement

In the testing phase, where candidates for unseen queries are selected, the transformation process is the same as described in Section 3.2.

4.2 Query Candidate Selection

The topics are split into approximately 70% train (90 topics) and 30% test (35 topics). In the training process, 30% of the train set is used for validation. To perform automatic candidate selection, we train two methods: a learning to rank model and a nearest neighbour approach. The queries that are refined by the candidate selection process are evaluated with six information retrieval measures (precision, recall, $F_{0.5}$, F_1, F_3 and work saved over sampling (WSS)). These measures are typically used to evaluate retrieval in the case of systematic reviews [8, 20, 26]. As the candidate selection function seeks to maximise an evaluation measure, a model is trained for each of the evaluation measures considered in this work for both the learning to rank and nearest neighbour candidate selectors, for a total of 12 models.

4.2.1 Learning to Rank. The DART learning to rank model is trained within the QuickRank framework [2]. We parametrise the DART model with a tree size of 100, a drop-out rate of 0.8, with uniform sampling, tree normalisation, a learning rate (shrinkage) of 0.1, and

to optimise the ranking of queries using DCG@1.[4] We perform candidate selection using this model with the same ranking effectiveness metric (DCG@1). The choice of DCG@1 is motivated by the task of identifying the top-1 ranked candidate query to transition to in a query chain. We sample the training data by removing a large portion of negative examples (two-thirds) to train a DART learning to rank model, as we discovered empirically that negative examples degrade the ranking quality.

4.2.2 Nearest Neighbour. Our nearest neighbour approach first records the feature vectors and evaluation measure values for each query in the training set. Next, for a new unseen query, an evaluation measure value is approximated by searching for a query in the training set which minimises the distances between the two feature vectors and maximises the recorded evaluation measure score. The intuition here is to find the most effective query from the training set which is most similar to an unseen query. In order to find the top-1 candidate to continue the query chain, each generated candidate query is ranked by the approximated evaluation measure value described previously. For efficiency reasons, we only store the deltas of distance and score from the most effective query generated from each seed query. The entire training data is used to create a nearest neighbour model, as we empirically discovered that unlike learning to rank, with the nearest neighbour approach negative examples improved the ranking quality.

5 RESULTS AND ANALYSIS

The main results of the experiments are reported in Table 3, where statistical significant differences are computed using a paired two-tailed t-test with $p < 0.01$. Along with the results obtained when unjudged citations were considered not relevant, we also report results obtained by considering the optimistic and probabilistic (maximum likelihood estimation) residuals: these are indicated by the subscripts *r* and *mle*, respectively.

Overall, the results in Table 3 indicate that the automatic candidate selection methods identified refined queries that were better than the original query (Seed), for the chosen evaluation measures. Somewhat surprisingly for this task, the nearest neighbour candidate selector often outperformed the learning to rank selector, choosing candidate queries that more often refined the query, rather than degrading or broadening it. Notably, the nearest neighbour candidate selector that was trained to maximise the F_1 measure (indicated as $F1_n$ in Table 3) was the most effective at selecting queries that improve over the original seed queries in terms of precision, F_1, and F_3.

The summary of the effect each candidate selector has on the refined queries is presented in Table 4. These results suggest that, although the number of transformations that are made to queries is relatively small (sometimes less than five), the queries often differ significantly from the original seed queries. The final clauses included in the queries were impacted by the types of transformations that were allowed. The following mapping indicates which of the six transformations align with which headings in Table 4:

- Clauses, Keywords: *Clause Removal* & *cui2vec Expansion*
- # AND, # OR: *Logical Operator Replacement* (# NOT included for completeness)
- Fields, (Title, Abstract, MeSH, Other): *Field Restrictions*
- Exp. Mesh: *MeSH Explosion*
- Avg. MeSH, Max. MeSH: *MeSH Parents*

The query expansion method *cui2vec* used was not chosen often by the candidate selectors: on average, the selected queries did not contain more non-atomic clauses than the original seed queries (this is explored in more detail in Section 5.1).

To provide further insight into how each candidate selector model affects the resulting selected queries, Table 5 presents the average proportion of fields and Boolean operators in queries. As the number of clauses in each of the selected queries differ from each trained candidate selector, this table seeks to show the changes in the types of fields and types of Boolean operators in proportion to the size of the query (i.e., with respect to the total number of Boolean operators and fields). For example, the $F1_n$ selector (which consistently improved the effectiveness of queries over the original seed queries for all measures) tended to increase the number of Title fields while decreasing the number of Abstract fields, as well as increase the number of AND operators while decreasing the number of OR operators. These refinements to queries align with the intuition that using the Title field will yield fewer results than the Abstract (or both fields combined), and that the AND operator is more restrictive than the OR operator (the intersection of retrieved citations as opposed to the union).

In terms of directly optimising precision and recall, we further analyse the P_n, and R_l selectors, as each recorded the highest increase in precision and recall, respectively. The P_n selector showed a similar proportion of fields to $F1_n$, however the proportion of AND operators was lower and the proportion of OR operators was higher. The R_l selector had fewer Title *and* Abstract fields, and it removed atomic clauses that did not contain MeSH fields (this is implied by the fact that no transformation allowed MeSH terms to be added, and that it was possible for any clause to be removed with the *Clause Removal* transformation). In addition, this selector had a higher number of OR operators than AND operators. Finally, Table 4 shows that R_l is the only selector to increase the average MeSH depth, suggesting that this selector often applied the *MeSH Parent* transformation.

Next, we explore at a topic level the differences between candidate selectors and their effects on evaluation. Figures 4 and 5 present the topic-by-topic effectiveness of the selectors with respect to the target evaluation measures. The differences between the precision candidate selectors (i.e. the learning to rank selector and the nearest neighbour selector) is noticeable. The P_l selector in fact chose many queries which degraded the effectiveness in terms of precision. On the other hand, the P_n selector had (larger) gains for a similar number of topics, but when compared to the P_l selector, it shows both a reduced number of queries for which a loss was recorded, and a reduced amount of loss when losses happened.

On the other hand, the R_l and R_n selectors were more similar, and many of the same topics had improved effectiveness. The results of these figures reflect the intuition that refining a query to increase precision is more difficult than improving recall. To contrast the previous two figures, Figure 6 presents the effect the $F1_n$ selector

[4]The exact parameters used to train the model can be found in the experiment pipeline files.

Table 3: Comparison of the effectiveness of the original queries in the test set (Seed) and each of the candidate selectors. Reported are the average values across topics. Values in bold indicate the highest value in that column. Statistical significance (two-tailed t-test where $p < 0.01$) is indicated with $*$. Evaluation methods are abbreviated in the following manner: P→precision, R→recall, WSS→work saved over sampling. Residual evaluation can be identified with the $_r$ (optimistic) and $_{mle}$ (maximum likelihood) subscripts. The two candidate selection functions can be identified by the evaluation measure they optimise, and a corresponding letter: e.g., P_l corresponds to the learning to rank selector which maximised precision, and $F1_n$ corresponds to the nearest neighbour selector which maximised F1.

	P	P_{mle}	P_r	R	R_{mle}	R_r	F0.5	$F0.5_{mle}$	$F0.5_r$	F1	$F1_{mle}$	$F1_r$	F3	$F3_{mle}$	$F3_r$	WSS	WSS_{mle}	WSS_r
Seed	0.0211	0.0307	0.7344	0.8395	0.8635	0.9831	0.0255	0.0358	0.7353	0.0373	0.0493	0.7366	0.1107	0.1331	0.7384	0.8386	0.8626	0.9822
P_l	0.0068	0.0142	0.7100	0.6989	0.7406	0.9591	0.0084	0.0161	0.7104	0.0129	0.0214	0.7109	0.0468	0.0610	0.7117	0.6849	0.7267	0.9450
P_n	0.0207	0.0432	0.5909	0.6403*	0.6810*	0.8541	0.0248	0.0472	0.5908	0.0355	0.0591	0.5908	0.0977	0.1331	0.5912	0.6396*	0.6802*	0.8533
R_l	0.0115	0.0195	0.7948	0.8833	0.8978	0.9580	0.0139	0.0222	0.7954	0.0201	0.0292	0.7963	0.0573	0.0716	0.7976	0.7946	0.8091	0.8694*
R_n	0.0023	0.0088	**0.7985**	**0.8939**	**0.9111**	0.9865	0.0029	0.0094	**0.7986**	0.0045	0.0113	**0.7988**	0.0183*	0.0270*	0.7990	**0.8551**	**0.8723**	0.9476
$F0.5_l$	0.0113	0.0237	0.7083	0.7826	0.8046	0.9513	0.0139	0.0254	0.7081	0.0213	0.0333	0.7079	0.0793	0.1011	0.7079	0.7802	0.8022	0.9489
$F0.5_n$	0.0191	0.0225	0.8206	0.8245	0.8355	0.9690	0.0231	0.0271	0.8215	0.0336	0.0393	0.8228	0.0965	0.1118	0.8245	0.8125	0.8235	0.9570
$F1_l$	0.0206	0.0229	0.7623	0.8330	0.8503	0.9787	0.0248	0.0275	0.7631	0.0360	0.0396	0.7644	0.1007	0.1090	0.7661	0.8224	0.8397	0.9681
$F1_n$	**0.0416**	**0.0638**	0.6474	0.6726*	0.7103*	0.9188	0.0486	**0.0604**	0.6454	**0.0655**	**0.0795**	0.6430	**0.1472**	**0.1772**	0.6408	0.6719*	0.7095*	0.9181
$F3_l$	0.0344	0.0248	0.6501	0.6860	0.7104*	0.9129	**0.0355**	0.0297	0.6497	0.0436	0.0424	0.6493	0.1041	0.1147	0.6490	0.6847	0.7091*	0.9116
$F3_n$	0.0175	0.0312	0.7279	0.7850	0.8195	0.9583	0.0212	0.0339	0.7191	0.0308	0.0448	0.7155	0.0910	0.1173	0.7144	0.7838	0.8183	0.9570
WSS_l	0.0152	0.0199	0.7666	0.7590	0.7961	0.9881	0.0181	0.0237	0.7665	0.0257	0.0335	0.7663	0.0634	0.0812	**0.7661**	0.7471	0.7842	0.9762
WSS_n	0.0229	0.0277	0.7323	0.7955	0.8177	**0.9867**	0.0275	0.0331	0.7329	0.0397	0.0474	0.7337	0.1131	0.1326	0.7349	0.7916	0.8138	**0.9828**

Table 4: Summary of the query refinement process. Shown in the table is the average difference in change of different aspects of queries obtained by each selector. A negative value indicates a reduction in the corresponding aspect, a positive value indicates an increase, and a zero value indicates that aspect was not changed. For comparison, the average values of the original *Seed* queries are reported as well. *Clauses* refers to the number of Boolean clauses in the query. *# AND*, *# OR*, and *# NOT* indicate how many of the corresponding operators were in the query. *Fields* indicates how many fields were in the query in total; this is further divided into how many of those were Title, Abstract, MeSH, or other (e.g., dates or publication type). *Exp. MeSH* indicates the number of MeSH terms that have been exploded, and *Avg. MeSH* & *Max. MeSH* indicate the average and maximum depth in the ontology of the MeSH terms. Two-tailed statistical significance where $p < 0.01$ between the Seed queries and the queries selected by each model is indicated with $*$.

	Clauses	# AND	# OR	# NOT	Fields	Fields (Title)	Fields (Abstract)	Fields (MeSH)	Fields (Other)	Keywords	Exp. MeSH	Avg. MeSH	Max. MeSH
Seed	12.6000	4.7714	7.3714	0.4571	74.4286	27.8286	29.3143	14.3143	2.9714	41.1714	4.2286	3.1763	7.7143
P_l	-3.3429*	-1.3714*	-1.8857*	-0.0857	-23.9143*	-9.2571	-9.3143	-4.8857*	-0.4571	-12.8857*	-1.5143*	-0.2009	-0.5714
P_n	-0.1714	0.3714	-0.5429*	0.0000	-2.9429*	0.3714	-2.7429*	-0.2000	-0.3714	-0.3429	-0.5143*	-0.2425	-0.2857
R_l	-2.8286	-1.6286	-1.0857	-0.1143	-18.5429	-7.1714	-8.1429	-2.3714	-0.8571	-11.0000	-0.7143	0.3190	-0.3143
R_n	-1.0857*	-1.0286*	0.0000	-0.0571	-5.9714*	-2.5429*	-1.9429*	-1.0000*	-0.4857	-2.6286*	-0.2857	-0.0877	-0.2286
$F0.5_l$	-2.3429*	0.0000	-2.2286*	-0.1143	-13.4000*	-3.7143*	-4.9429*	-4.2571*	-0.4857	-7.2857*	-1.2571*	-0.4887	-1.4571*
$F0.5_n$	-0.3143*	-0.1429	-0.1429	-0.0286	-2.3143*	-0.2571	-1.4000*	-0.2857	-0.3714	-0.3143*	-0.4571*	-0.1620	-0.3429
$F1_l$	-4.4286*	-1.5143*	-2.6286*	-0.2857*	-21.7143*	-6.9143*	-7.5143*	-6.4857*	-0.8000	-11.8000*	-2.0571*	-1.6380*	-3.9143*
$F1_n$	-0.3143	0.5143*	-0.8286*	0.0000	-3.3429*	-0.5714	-2.0286*	-0.4000	-0.3429	-0.7714	-0.7714*	-0.2804	-0.3143
$F3_l$	-0.4857	0.2857	-0.7714*	0.0000	-6.6000	-2.6000	-2.9429	-0.6857	-0.3714	-2.6286	-0.2000	-0.2266	-0.3429
$F3_n$	0.0000	-0.0286	0.0286	0.0000	-1.0571	-0.3429	-0.3429	0.0000	-0.3714	0.0000	-0.8286*	-0.3331*	-0.2857
WSS_l	-2.2857*	-0.8571*	-1.3714*	-0.0571	-12.7429*	-5.0286*	-5.1143*	-2.2000*	-0.4000	-6.1143*	-0.4857*	-0.4084	-0.8857
WSS_n	-0.8571*	-0.2571	-0.5714*	-0.0286	-5.9714*	-2.1143*	-2.4000*	-1.0000*	-0.4571	-2.8000*	-0.6571*	-0.1420	-0.3143

had on each topic in terms of the six evaluation measures considered in this work. While many queries had improved precision and maintained or even increased recall (e.g., topic 83), approximately one third of queries had a reduction in recall. This suggests that there may not have been enough training examples for this selector to choose a candidate query that is likely to cause an increase in recall.

(a) Original query.

```
1. measles.tw.
2. rubeola.tw.
3. morbilli*.tw.
4. exp Measles/
5. exp Measles virus/
6. or/1-5
7. exp Vitamin A/
8. vitamin a.ti,ab,sh.
9. retinol.ti,ab,sh.
10. exp Dietary Supplements/
11. or/7-10
12. bitot*.tw.
13. spot*.tw.
14. 12 and 13
15. vision*.tw.
16. visual*.tw.
17. eye*.tw.
18. sight*.tw.
19. or/15-18
20. xerosis*.tw.
21. exp Vision Disorders/
22. Xerophthalmia/
23. Night Blindness/
24. xerophthalmia*.tw.
25. exp Blindness/
26. keratomalacia.tw.
27. blind*.tw.
28. 14 or 19 or 20 or 21 or 22 or \
23 or 24 or 25 or 26 or 27
29. 6 and 11 and 28
```

(b) Transformed query.

```
1. measles.ti.
2. rubeola.tw.
3. morbilli*.ti.
4. exp Measles/
5. exp Measles virus/
6. or/1-5
7. Retinoids/
8. vitamin a.ti,ab,sh.
9. retinol.ti,ab,sh.
10. exp Dietary Supplements/
11. or/7-10
12. bitot*.tw.
13. spot*.tw.
14. 12 and 13
15. vision*.tw.
16. visual*.tw.
17. eye*.tw.
18. sight*.ab.
19. or/15-18
20. xerosis*.tw.
21. exp Vision Disorders/
22. Xerophthalmia/
23. Night Blindness/
24. xerophthalmia*.tw.
25. exp Blindness/
26. keratomalacia.tw.
27. blind*.ti.
28. 14 or 19 or 20 or 21 or 22 or \
23 or 24 or 25 or 26 or 27
29. 6 and 11 and 28
```

Figure 2: Comparison between the original query (left) and a transformed query (right) for topic 154. The transformed query was chosen using the $F1_n$ selector, and was the least effectively refined query. Note that on line 28 for both queries there is a carriage return as the line was too long.

Finally, we compare two example queries by examining the transformations that have been applied to them. The queries chosen for comparison are from topics 73 and 154. These topics were respectively the best and worst performing ones when query transformations were obtained through the $F1_n$ candidate selector (in the case of topic 154 it was chosen because the query for 106 was too long to fit into a column). The transformed query for topic 73 obtained a large increase in precision, while maintaining recall, while the transformed query for topic 154 obtained a minor decrease in precision and a large decrease in recall.

The original and the transformed query for topic 73 are shown in Figure 3. The transformations made to the query were that lines 59 and 75 were both removed, and lines 40, 46 and 75 (which become line 66 in the transformed query) had their operators changed (from OR to AND). The original and the transformed query for topic 154 are shown in Figure 2. The transformations made to its query were that lines 1, 3, and 27 had their fields changed from Title and Abstract to only Title, and the MeSH term of line 7 was expanded to its parent.

Upon examination of the queries in Figures 3 and 2, it is clear that they considerably differ in length: one may question whether the length difference (in number of clauses) is associated with the effectiveness difference — according to this hypothesis shorter queries would generally perform worse than longer queries. However, upon further analysis we found that this is not the case. In fact, the Pearson's r correlation between the F_1 evaluation measure and the number of Boolean clauses in each query was only $r = 0.175$. Similar results were obtained when considering the other evaluation measures used in this study, thus suggesting that there is no direct relation between query length in terms of number of Boolean clauses and retrieval effectiveness. This finding is similar to what

(a) Original query.

```
1. controlled clinical trial.pt.
2. placebo.ab.
3. clinical trials as topic/
4. randomly.ab.
5. trial.ti.
6. randomized.tw.
7. randomized controlled trial.pt.
8. or/1-7
9. humans/
10. animals/
11. 9 not 10
12. 8 and 11
13. abdominal.tw.
14. abdomen.tw.
15. chest.tw.
16. thoracic.tw.
17. trunk.tw.
18. or/13-17
19. exp Wounds, Penetrating/
20. 18 and 19
21. abdominal.tw.
22. abdomen.tw.
23. chest.tw.
24. thoracic.tw.
25. trunk.tw.
26. or/21-25
27. trauma*.tw.
28. injur*.tw.
29. penetrat*.tw.
30. stab*.tw.
31. or/27-30
32. 26 and 31
33. Splenic.tw.
34. spleen.tw.
35. stomach.tw.
36. gastric.tw.
37. or/33-36
38. rupture*.tw.
39. burst*.tw.
40. 38 or 39
41. 37 and 40
42. heart.tw.
43. cardiac.tw.
44. aortic.tw.
45. aorta*.tw.
46. or/42-45
47. rupture*.tw.
48. 46 and 47
49. exp Abdominal Injuries/
50. exp thoracic injuries/
51. 20 or 32 or 41 or 48 or 49 or 50
52. Blood.tw.
53. Plasma.tw.
54. 52 or 53
55. Autologous.tw.
56. 54 and 55
57. Autohaemotransfusion*.tw.
58. Auto-haemotransfusion*.tw.
59. Autohemotransfusion*.tw.
60. Auto-hemotransfusion*.tw.
61. Autotransfusion*.tw.
62. Auto-transfusion*.tw.
63. or/56-62
64. cell*.tw.
65. blood.tw.
66. 64 or 65
67. transfusion*.tw.
68. salvage.tw.
69. save*.tw.
70. or/67-69
71. 66 adj5 70
72. exp Blood Transfusion, Autologous/
73. exp Blood Loss, Surgical/
74. exp Blood Transfusion/
75. 63 or 71 or 72 or 73 or 74
76. 51 and 75
77. 12 and 76
```

(b) Transformed query.

```
1. controlled clinical trial.pt.
2. placebo.ab.
3. clinical trials as topic/
4. randomly.ab.
5. trial.ti.
6. randomized.tw.
7. randomized controlled trial.pt.
8. or/1-7
9. humans/
10. animals/
11. 9 not 10
12. 8 and 11
13. abdominal.tw.
14. abdomen.tw.
15. chest.tw.
16. thoracic.tw.
17. trunk.tw.
18. or/13-17
19. exp Wounds, Penetrating/
20. 18 and 19
21. abdominal.tw.
22. abdomen.tw.
23. chest.tw.
24. thoracic.tw.
25. trunk.tw.
26. or/21-25
27. trauma*.tw.
28. injur*.tw.
29. penetrat*.tw.
30. stab*.tw.
31. or/27-30
32. 26 and 31
33. Splenic.tw.
34. spleen.tw.
35. stomach.tw.
36. gastric.tw.
37. or/33-36
38. rupture*.tw.
39. burst*.tw.
40. 38 and 39
41. 37 and 40
42. heart.tw.
43. cardiac.tw.
44. aortic.tw.
45. aorta*.tw.
46. and/42-45
47. rupture*.tw.
48. 46 and 47
49. exp Abdominal Injuries/
50. exp thoracic injuries/
51. 20 or 32 or 41 or 48 or 49 or 50
52. Blood.tw.
53. Plasma.tw.
54. 52 or 53
55. Autologous.tw.
56. 54 and 55
57. Autohaemotransfusion*.tw.
58. Auto-haemotransfusion*.tw.
59. Auto-hemotransfusion*.tw.
60. Autotransfusion*.tw.
61. Auto-transfusion*.tw.
62. or/56-61
63. exp Blood Transfusion, Autologous/
64. exp Blood Loss, Surgical/
65. exp Blood Transfusion/
66. and/63-65
67. 51 and 66
68. 12 and 67
```

Figure 3: Comparison between the original query (left) and a transformed query (right) for topic 73. The transformed query was chosen using the $F1_n$ selector, and was the most effectively refined query.

was obtained by Scells et al. when studying query performance predictors for this domain [22].

The next two sections explore our research questions with respect to the impact of semantic transformations in the candidate selection process, and the effect unjudged citations had on the evaluation.

Table 5: Average proportion of clauses for each field, each Boolean operator, and each Exploded MeSH term, across selected queries by each query selector approach. The original queries (Seed) are included for comparison. Statistically significance (two-tailed t-test with $p < 0.01$) between the *Seed* and listed selectors is indicated with $*$.

	Title	Abstract	MeSH	Other	AND	OR	NOT	Exp. MeSH
Seed	0.3327	0.3589	0.2711	0.0374	0.3542	0.6172	0.0286	0.1227
$F1_l$	0.3366*	0.3764*	0.2324*	0.0547	0.3589*	0.6298*	0.0113*	0.0704*
$F1_n$	0.3432	0.3370*	0.2901	0.0297	0.4480*	0.5223*	0.0297	0.0819*
$F3_l$	0.3730	0.3500	0.2472	0.0298	0.4287	0.5417*	0.0296	0.1196
$F3_n$	0.3376	0.3706	0.2625	0.0293	0.3447	0.6267	0.0286	0.0819*
$F0.5_l$	0.3513*	0.3675*	0.2402*	0.0410	0.4406	0.5392*	0.0202	0.1048*
$F0.5_n$	0.3590	0.3501*	0.2621	0.0289	0.3387	0.6359	0.0254	0.1044*
P_l	0.3326	0.3657	0.2656*	0.0360	0.3246*	0.6402*	0.0352	0.1024*
P_n	0.3626	0.3198*	0.2882	0.0294	0.4263	0.5441*	0.0296	0.0936*
R_l	0.3161	0.3357	0.3221	0.0262	0.3196	0.6567	0.0236	0.1153
R_n	0.3252*	0.3822*	0.2647*	0.0280	0.2729*	0.6992	0.0278	0.1223
WSS_l	0.3378*	0.3559*	0.2771*	0.0293	0.3102*	0.6641*	0.0256	0.1358*
WSS_n	0.3314*	0.3504*	0.2899*	0.0283	0.3790	0.5927*	0.0283	0.1049*

5.1 Impact of Semantic Transformations

To answer **RQ1**: '*Which types of transformations are the most effective at refining Boolean queries in systematic review literature search?*', a comparison between candidate selection transformations was made. The summary of query refinement are presented in Table 4. On average, none of the approaches selected queries which performed explicit query expansion. Indeed, a deeper analysis of the results identifies that none of the queries are longer than the original seed. This suggests that, in this context, query expansion may not be an effective way to refine a query. On the other hand, query reduction was chosen very often, more so by the learning to rank selector — which tended to remove clauses from queries more often than the nearest neighbour selector. While the candidate queries that had the *cui2vec Expansion* transformation applied were not chosen by either of the selectors, the *MeSH Parent* transformation was. The R_l selector chose queries with an average MeSH depth of 0.32 higher than the original seed query. This suggests that to maximise an increase in recall, this selector opted to chose a query that broadened the scope of MeSH keyword than to add new, somewhat related text-matching keywords. Overall the results indicate that, among the investigated transformations, *syntactic* transformations play a larger role in query refinement than *semantic* transformations.

5.2 Impact of Unjudged Citations

To answer **RQ2**: '*What effect do unjudged citations have on the effectiveness of refined Boolean queries in systematic review literature search?*', the evaluation measures and the estimates obtained with the two residual heuristics are compared. As reported in Table 3, these results show that the effect of unjudged citations is not significant. When a selector chooses candidate queries that perform

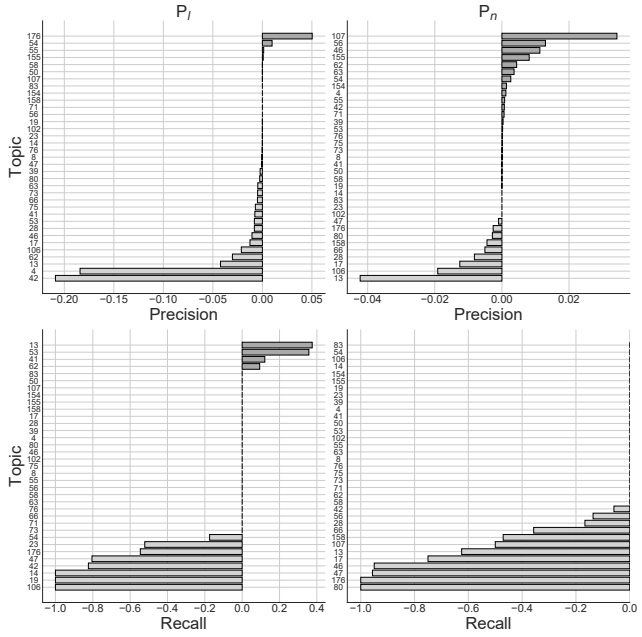


Figure 4: Per-topic gains in precision (top) and recall (bottom) over the Seed query for the candidate selectors which maximised precision (P_l and P_n).

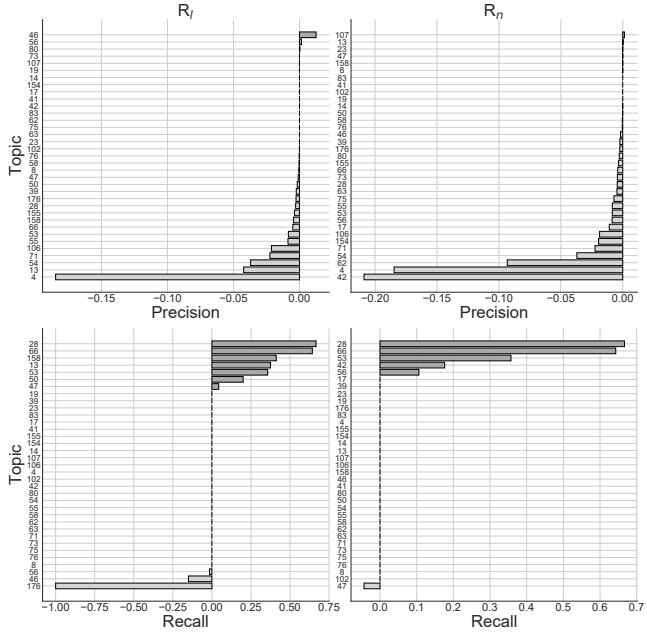


Figure 5: Per-topic gains in precision (top) and recall (bottom) over the Seed query for the candidate selectors which maximised recall (R_l and R_n).

statistically significantly worse for an evaluation measure, the residual *mle* results are also significantly worse. Nonetheless, Table 3

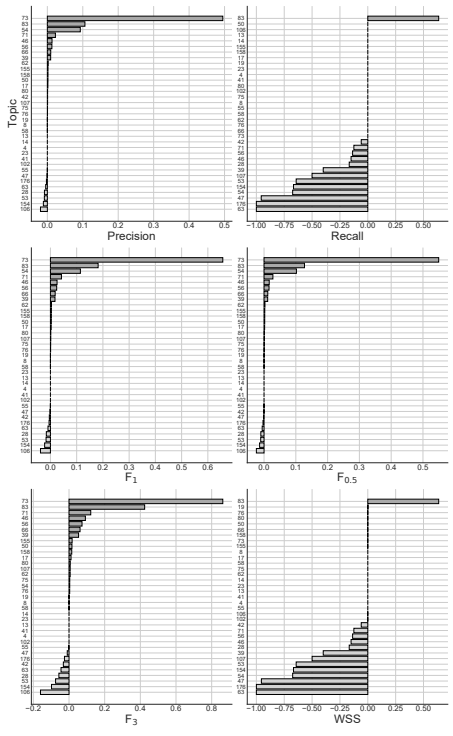


Figure 6: Per-topic gains over the Seed query in the evaluation measures considered in our study. Here we considered the nearest neighbour candidate selector that maximised F1 ($F1_n$.)

still shows that in many cases, effectiveness increases when unjudged citations are factored into evaluation. For example, the P_n selector does not obtain increases in precision over the seed queries, when unjudged citations are considered non-relevant. However, when using the *mle* heuristic, precision is higher, though with the *r* heuristic, precision is also lower. In fact, from analysing the *r* heuristic, we can determine that the P_n selector retrieves many more non-relevant citations than the P_l selector. For these reasons, estimates for residual should be considered when performing high-recall tasks such as this; or in tasks that modify or change queries in some way. This is especially important when relevance assessments are largely incomplete, and no system variation was considered when forming the pool for assessments: this was the case for systematic review collections available for Information Retrieval research [8, 26].

6 CONCLUSIONS

This paper presented three novel semantic transformations that integrate with previous work on systematic review query generation. In addition, we cast the task of query generation from previous work [24] to query refinement — where newly generated queries should improve precision while maintaining or improving recall. To this end, we employed two new candidate selector functions, one based on learning to rank, and one based on a nearest neighbour algorithm. Both candidate selectors were trained to maximise six evaluation measures. When evaluating query refinement, we also revealed that the refinement retrieved unjudged documents; we accounted for these in the evaluation by computing the residuals of the evaluation measures. We did so by considering both an optimistic residual and a probabilistic residual (based on maximum likelihood estimation). Our results showed that the nearest neighbour model was able to more effectively select query refinements than the learning to rank approach. Our experiments have shown that the explicit query expansion method (*cui2vec Expansion*) proposed in this work is not viable for query refinement (as candidates with other transformations outrank it). However, the explicit query reduction method (*Clause Removal*), and to a lesser extent the implicit query expansion method (*MeSH Parents*) can, in fact, be used to help refine queries.

The detailed analysis of how each set of selected queries was transformed by candidate selectors suggested that the syntactic transformations were in fact more suited to refinement than semantic transformations. Our experiments also showed that the effect of unjudged citations on evaluation is significant: unjudged citations should be accounted for in some way when performing evaluation.

Future plans for the task of automatic query refinement for systematic review literature search will focus on exploring other methods for candidate selection and other features, as well as improving the sampling of queries in the exploration/generation phase for collecting training data. With this in mind, the current sampling method leads to a large imbalance of negative to positive training samples. One promising sampling method could be to sample each query with a set of "known-relevant" seed documents. These are used in systematic review literature search by information specialists to guide the query generation process. Recently, a seed-driven approach to ranking has been shown to reduce the workload of the screening task [15]. We also plan to further study other techniques for semantic query transformation, such as query reduction [13]. Query reduction techniques for professional search have been shown to be successful in other contexts, both in the clinical domain [11, 12, 28], and outside of the medical field [4, 16].

The proposed method for automatic query refinement has the potential to vastly reduce the time spent creating systematic reviews. The searching and screening processes contribute to a significant period of time and money in the systematic review creation process. By reducing the total number of citations to screen by automatically refining already highly effective queries, the time spent screening citations and therefore the total cost of a systematic review can be reduced. These reductions lead to faster and more up-to-date evidence based medicine which in turn lead to more accurate clinical decisions and better health outcomes.

ACKNOWLEDGMENTS

Harrisen Scells is the recipient of a CSIRO PhD Top Up Scholarship. Dr Guido Zuccon is the recipient of an Australian Research Council DECRA Research Fellowship (DE180101579) and a Google Faculty Award.

REFERENCES

[1] Andrew L. Beam, Benjamin Kompa, Inbar Fried, Nathan P. Palmer, Xu Shi, Tianxi Cai, and Isaac S. Kohane. 2018. Clinical Concept Embeddings Learned from Massive Sources of Medical Data. *CoRR* abs/1804.01486 (2018). arXiv:1804.01486 http://arxiv.org/abs/1804.01486

[2] Gabriele Capannini, Claudio Lucchese, Franco Maria Nardini, Salvatore Orlando, Raffaele Perego, and Nicola Tonellotto. 2016. Quality versus efficiency in document scoring with learning-to-rank models. *Information Processing & Management* 52, 6 (2016), 1161–1177.

[3] Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys (CSUR)* 44, 1 (2012), 1.

[4] Debasis Ganguly, Johannes Leveling, Walid Magdy, and Gareth JF Jones. 2011. Patent query reduction using pseudo relevance feedback. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 1953–1956.

[5] Claudia Hauff, Djoerd Hiemstra, and Franciska de Jong. 2008. A survey of pre-retrieval query performance predictors. In *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 1419–1420.

[6] Kalervo Järvelin and Jaana Kekäläinen. 2000. IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 41–48.

[7] Jimmy, Guido Zuccon, and Bevan Koopman. 2018. Choices in Knowledge-Base Retrieval for Consumer Health Search. In *Advances in Information Retrieval*, Gabriella Pasi, Benjamin Piwowarski, Leif Azzopardi, and Allan Hanbury (Eds.). Springer International Publishing, Cham, 72–85.

[8] Evangelos Kanoulas, Dan Li, Leif Azzopardi, and Rene Spijker. 2017. CLEF 2017 Technologically Assisted Reviews in Empirical Medicine Overview. In *CLEF'17*.

[9] Sarvnaz Karimi, Stefan Pohl, Falk Scholer, Lawrence Cavedon, and Justin Zobel. 2010. Boolean versus ranked querying for biomedical systematic reviews. *BMC MIDM* 10, 1 (2010), 1.

[10] Youngho Kim, Jangwon Seo, and W Bruce Croft. 2011. Automatic boolean query suggestion for professional search. In *SIGIR'11*.

[11] Bevan Koopman, Peter Bruza, Guido Zuccon, Michael Lawley, and Laurianne Sitbon. 2012. Graph-based Concept Weighting for Medical Information Retrieval. In *Seventeenth Australasian Document Computing Symposium (ADCS 2012)*. Dunedin, New Zealand.

[12] Bevan Koopman, Liam Cripwell, and Guido Zuccon. 2017. Generating clinical queries from patient narratives: A comparison between machines and humans. In *Proceedings of the 40th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 853–856.

[13] Giridhar Kumaran and Vitor R Carvalho. 2009. Reducing long queries using query quality predictors. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 564–571.

[14] John Lavis, Huw Davies, Andy Oxman, Jean-Louis Denis, Karen Golden-Biddle, and Ewan Ferlie. 2005. Towards systematic reviews that inform health care management and policy-making. *JHSRP* 10, 1_suppl (2005), 35–48.

[15] Grace E. Lee and Aixin Sun. 2018. Seed-driven Document Ranking for Systematic Reviews in Evidence-Based Medicine. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. ACM, New York, NY, USA, 455–464. https://doi.org/10.1145/3209978.3209994

[16] Daniel Locke, Guido Zuccon, and Harrisen Scells. 2017. Automatic Query Generation from Legal Texts for Case Law Retrieval. In *Asia Information Retrieval Symposium*. Springer, 181–193.

[17] Jessie McGowan and Margaret Sampson. 2005. Systematic reviews need systematic searchers (IRP). *Journal of the Medical Library Association* 93, 1 (2005), 74.

[18] Mandar Mitra, Amit Singhal, and Chris Buckley. 1998. Improving automatic query expansion. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 206–214.

[19] Alistair Moffat and Justin Zobel. 2008. Rank-biased precision for measurement of retrieval effectiveness. *ACM Transactions on Information Systems (TOIS)* 27, 1 (2008), 2.

[20] Alison O'Mara-Eves, James Thomas, John McNaught, Makoto Miwa, and Sophia Ananiadou. 2015. Using text mining for study identification in systematic reviews: a systematic review of current approaches. *Systematic reviews* 4, 1 (2015), 5.

[21] KV Rashmi and Ran Gilad-Bachrach. 2015. Dart: Dropouts meet multiple additive regression trees. In *International Conference on Artificial Intelligence and Statistics*. 489–497.

[22] Harrisen Scells, Leif Azzopardi, Guido Zuccon, and Bevan Koopman. 2018. Query Variation Performance Prediction for Systematic Reviews. In *SIGIR'18*.

[23] Harrisen Scells, Daniel Locke, and Guido Zuccon. 2018. An Information Retrieval Experiment Framework for Domain Specific Applications. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*.

[24] Harrisen Scells and Guido Zuccon. 2018. Generating Better Queries for Systematic Reviews. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. ACM, New York, NY, USA, 475–484.

[25] Harrisen Scells, Guido Zuccon, Bevan Koopman, Anthony Deacon, Leif Azzopardi, and Shlomo Geva. 2017. Integrating the framing of clinical questions via PICO into the retrieval of medical literature for systematic reviews. In *CIKM'17*.

[26] Harrisen Scells, Guido Zuccon, Bevan Koopman, Anthony Deacon, Leif Azzopardi, and Shlomo Geva. 2017. A Test Collection for Evaluating Retrieval of Studies for Inclusion in Systematic Reviews. In *SIGIR'2017*.

[27] Luca Soldaini, Arman Cohan, Andrew Yates, Nazli Goharian, and Ophir Frieder. 2015. Retrieving medical literature for clinical decision support. In *ECIR'15*. 538–549.

[28] Luca Soldaini, Andrew Yates, and Nazli Goharian. 2017. Learning to reformulate long queries for clinical decision support. *Journal of the Association for Information Science and Technology* 68, 11 (2017), 2602–2619.

[29] Guy Tsafnat, Paul Glasziou, Miew Keen Choong, Adam Dunn, Filippo Galgani, and Enrico Coiera. 2014. Systematic review automation technologies. *SR* 3, 1 (2014), 74.

[30] Jinxi Xu and W Bruce Croft. 1996. Query expansion using local and global document analysis. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 4–11.

Exploiting Diversity in Android TLS Implementations for Mobile App Traffic Classification

Satadal Sengupta
IIT Kharagpur, India
satadal.sengupta@iitkgp.ac.in

Niloy Ganguly
IIT Kharagpur, India
niloy@cse.iitkgp.ac.in

Pradipta De
Georgia Southern University, USA
pde@georgiasouthern.edu

Sandip Chakraborty
IIT Kharagpur, India
sandipc@cse.iitkgp.ac.in

ABSTRACT

Network traffic classification is an important tool for network administrators in enabling monitoring and service provisioning. Traditional techniques employed in classifying traffic do not work well for mobile app traffic due to lack of unique signatures. Encryption renders this task even more difficult since packet content is no longer available to parse. More recent techniques based on statistical analysis of parameters such as packet-size and arrival time of packets have shown promise; such techniques have been shown to classify traffic from a small number of applications with a high degree of accuracy. However, we show that when employed to a large number of applications, the performance falls short of satisfactory. In this paper, we propose a novel set of bit-sequence based features which exploit differences in randomness of data generated by different applications. These differences originating due to dissimilarities in encryption implementations by different applications leave footprints on the data generated by them. We validate that these features can differentiate data encrypted with various ciphers (89% accuracy) and key-sizes (83% accuracy). Our evaluation shows that such features can not only differentiate traffic originating from different categories of mobile apps (90% accuracy), but can also classify 175 individual applications with 95% accuracy.

CCS CONCEPTS

• **Networks** → **Packet classification**; Network monitoring; • **Security and privacy** → *Cryptanalysis and other attacks.*

KEYWORDS

traffic classification; randomness; bit-sequence

ACM Reference Format:
Satadal Sengupta, Niloy Ganguly, Pradipta De, and Sandip Chakraborty. 2019. Exploiting Diversity in Android TLS Implementations for Mobile App Traffic Classification. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313738

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313738

1 INTRODUCTION

Global mobile data traffic has witnessed astonishing growth over the last half-decade; having multiplied 18-fold since 2011, it is forecasted to account for 20% of all IP traffic by 2021 [1]. The sheer scale and enormous diversity of this traffic has engendered exceptional challenges for network administrators and service providers. One major goal of network administrators is providing differentiated quality of service (QoS) based on the traffic category (also known as service differentiation). The decision of how much bandwidth to provide where, may depend on either preferences of users in the network, or the policies adopted by the enterprise served by the administrator. The first step of such a solution is the classification of traffic originating from various applications. This task, however, has been rendered significantly difficult in case of mobile applications, due to the lack of unique traffic signatures.

Unlike in the case of traditional web traffic, mobile app traffic cannot be segregated based on the presence of obvious signatures (e.g., server address, app port number, unique identifiers in packet content, etc. [12]). In fact, the majority of mobile apps tunnel data over HTTP and HTTPS [24, 28], which use the same port (80 and 443 for HTTP and HTTPS respectively) for all traffic. Also, in case of sub-networks, all traffic is addressed to and from a common IP address (that of the gateway device), rendering IP-based traffic classification infeasible. The most difficult hurdle seems to be the increasing adoption of encryption. Encrypted traffic renders DPI-based efforts to identify signatures from packet content ineffective [26].

One popular way of circumventing this problem is using statistical analysis to classify application traffic [7, 27, 46]. Features such as packet-size, inter-arrival time of packets, etc. have been known to prove beneficial in classifying one type of traffic from another. However, these approaches are under the scanner, and techniques have already been proposed to thwart such attempts. For example, a recent study proposed using packet-size and arrival time obfuscation to confuse a trained classifier [41]. The authors implement their system as a proxy which can fragment and regulate packet departure in a way such that one category of traffic looks exactly like another. In the presence of such techniques, it is necessary to devise strategies which are immune towards obfuscation attempts.

In this paper, we propose a novel set of bit-sequence similarity based features, which leverage footprints left by encryption implementations to achieve traffic classification. A recent study reveals that in case of mobile platforms (and specifically the Android OS), there exists significant diversity in the implementation of TLS/SSL, with a vast majority of those implementations falling

short of the adequate security standards [41]. This phenomenon is mainly due to the *app store* model adopted by major platforms such as Google Android and Apple iOS, which enables developers with little to no expertise to implement security in a host of applications. TLS deployment requires configuring a complicated end-to-end architecture, with numerous 'knobs and buttons'; different permutations of these options adopted by the developers result in a wide array of cipher suites, key-sizes, and certificate management styles getting associated with mobile applications. Additionally, security implementations also depend extensively on the computational capabilities of the mobile devices; while high-end smartphones support the most secure standards, lower-end phones may only support weaker cipher suites, or none at all.

We observe that the differences in encryption implementations of mobile applications in the wild, leave footprints in the encrypted data generated by them, which can be harnessed using a set of novel bit-sequence based features. These features leverage the difference in the randomness of bits, when data is encrypted using different cipher algorithms and key-sizes. We argue that the tell-tale differences in randomness, although weak when data bits are assumed to be in a temporal sequence, begin emerging strongly when the sequence is transformed to the frequency domain (using Fast Fourier Transform). We substantiate this claim with a series of *standard randomness tests*[1] on 11 ciphers, the most noteworthy of which is the *Discrete Fourier Transform* based 'Spectral Test' [36]. This test exhibits the highest standard deviation in randomness scores (~ 0.38) across the ciphers, suggesting its suitability in classifying data generated by different ciphers. We proceed to train a classifier using the aforementioned features, on an offline dataset consisting of data encrypted with 11 different ciphers, and 3 different key-sizes, and obtain 89% (for different ciphers) and 83% (for different key sizes) accuracy in the best case, respectively.

Armed with this set of discriminative features, we propose a mechanism based on machine learning, which can enable classification of mobile network data into source applications. We achieve 71% accuracy in classifying packets from 175 Android apps using only these features, which improves to 95% when augmented with previously defined packet-level features. In comparison, a classifier trained only on baseline features, is able to exhibit only 62% accuracy. Additionally, this technique is not only resistant towards obfuscation, but is also simple to implement, since it requires computing the Fast Fourier Transform of a bit-stream (specialized hardware is available for such tasks [6]), and some additional computation.

The key contributions of this paper include:

(1) We propose and validate a set of novel bit-sequence based features, which enable classification among data encrypted with 11 ciphers and 3 different key-sizes, with 89% and 83% accuracy (best case), respectively (§2).
(2) We substantiate the efficacy of the bit-sequence based features in discriminating among diverse security implementations, by performing multi-class classification of network traffic originating from 175 mobile apps. While previously studied packet-level features (which serve as our baseline) account for only 62% accuracy, our proposed bit-sequence based features alone achieve 71% accuracy, which is boosted significantly (95% in the best case) in conjunction with packet-level features (§3).

[1]Recommended by the National Institute of Standards and Technology (NIST), US Department of Commerce for evaluating suitability of a random bit generator in cryptographic applications [4].

2 BIT-SEQUENCE BASED FEATURES

In this section, we substantiate the notion that data encrypted with different cryptographic system configurations yield different degrees of randomness. We show that it is possible to harness these differences in randomness, and construct "features" (in machine learning parlance) which are useful in extracting sensitive information from encrypted data, e.g., the cipher algorithm or the key-size.

2.1 Background

Secure cryptographic systems are characterized with certain desirable properties, namely (1) the *avalanche effect*, and (2) the *completeness* property. The avalanche effect mandates that a small change (even a bit) in the plaintext should create significant changes (alter a large number of bits) in the ciphertext [25]. The completeness property demands that *all* bits in the plaintext should affect *each* bit in the ciphertext [34]. The broader intent behind these properties is the generation of sufficient randomness in the encrypted data, such that an attacker is unable to exploit any patterns useful in deciphering the plaintext or the encryption key.

However, cipher implementations are seldom perfectly random; existing literature shows that different cipher implementations are characterized with different degrees of randomness [8, 23, 35, 44, 52]. An important measure of the strength of a cipher is therefore the amount of randomness it can induce in the data encrypted with it. While a large number of statistical tests are available to determine randomness, the National Institute of Standards and Technology (NIST), USA recommends a specific set of 15. A candidate cipher is evaluated based on whether the randomness (as determined by these tests) in bits generated due to encryption with it, is sufficiently high to be suitable for cryptographic applications [4]. In this section, we perform these 15 tests on data generated by different ciphers, and validate that the degrees of randomness are indeed different.

2.2 Randomness Analysis of Ciphers

We evaluate the randomness induced by different ciphers on data encrypted by them, by subjecting the data to a set of 15 recommended statistical tests, as mentioned before.

Randomness tests: We perform the following randomness tests: (1) Monobit test, (2) Block frequency test, (3) Independent runs test, (4) Longest runs test, (5) Spectral test, (6) Non-overlapping patterns test, (7) Overlapping patterns test, (8) Universal test, (9) Serial test, (10) Approximate entropy test, (11) Cumulative sums test, (12) Random excursions test, (13) Random excursions variant test, (14) Matrix rank test, and the (15) Linear complexity test. These tests have been defined and standardized in the NIST standard *SP 800-22* [4] [2]. The tests evaluate different properties of the test data which may indicate presence of patterns in the data (e.g., high frequency of specific bit-strings, too many zeros or too many ones, etc.). The score reported for each test is the test statistic *p-value*

[2]The descriptions of these tests have been omitted for the sake of brevity.

Table 1 Results of 15 randomness (recommended by NIST [4]) tests on data encrypted with 11 ciphers. The green row represents the randomness scores for the *Spectral Test*, which is touted as the most powerful test of randomness [36] and also shows the higest std. deviation across tests.

Ciphers/Tests	AES	ARC2	ARC4	Blow-fish	CAST	DES	DES3	PKCS#1 OAEP	PKCS#1 v1.5	XOR	Unenc-rypted	*Standard Deviation*
Monobit Test	0.3563	0.8495	0.4129	0.7034	0.7001	0.6347	0.6617	0.0000	0.0000	0.0000	0.0000	**0.3232**
Block Frequency Test	0.6120	0.6700	0.8508	0.8717	0.6739	0.4795	0.0982	0.7956	0.6285	0.0000	1.0000	**0.2973**
Independent Runs Test	0.2928	0.2092	0.5690	0.9011	0.1275	0.5922	0.0351	0.0000	0.0000	0.0000	0.0000	**0.2950**
Longest Runs Test	0.3735	0.5508	0.2576	0.2018	0.3929	0.5349	0.4236	0.5016	0.2965	0.0000	0.0000	**0.1845**
Spectral Test	0.9641	0.9768	0.7489	0.3135	0.9367	0.5588	0.3628	0.1748	0.9982	0.0000	0.0000	**0.3791**
Non Overlapping Patterns Test	0.3824	0.8874	0.5719	0.7193	0.0479	0.7668	0.1335	0.7917	0.5233	0.0000	0.0000	**0.3263**
Overlapping Patterns Test	0.0123	0.0821	0.2920	0.3004	0.1922	0.8939	0.2518	0.1550	0.1712	0.0000	0.0000	**0.2398**
Universal Test	0.3101	0.2031	0.7855	0.4042	0.8567	0.8574	0.8058	0.3027	0.5298	0.0000	0.0000	**0.3141**
Serial Test	0.2937	0.1751	0.6898	0.8080	0.3829	0.6550	0.0151	0.5689	0.0489	0.0000	0.0000	**0.2931**
Approximate Entropy Test	0.3202	0.0553	0.5882	0.5285	0.2862	0.0851	0.3340	0.2450	0.1651	0.0000	0.0000	**0.1909**
Cumulative Sums Test	0.3824	0.9320	0.5635	0.6605	0.8950	0.3800	0.6888	0.0000	0.0000	0.0000	0.0000	**0.3498**
Random Excursions Test	0.8251	0.4261	0.4576	0.9930	0.8197	0.3436	0.4022	0.2735	0.5566	0.7000	0.7000	**0.2208**
Random Excursions Variant Test	0.6991	0.8571	0.3518	0.2654	0.8173	0.2099	0.3670	0.0811	0.4618	1.0000	0.3865	**0.2839**
Matrix Rank Test	0.5624	0.9083	0.5359	0.9268	0.7473	0.5678	0.3579	0.1282	0.8707	0.0000	0.0000	**0.3323**
Linear Complexity Test	0.9053	0.8132	0.8263	0.2563	0.7104	0.0709	0.8156	0.2682	0.8527	0.2072	0.3292	**0.3047**

which is a function of the observed (encrypted) data. The *p-value* indicates the probability that a perfect random number generator would have produced a sequence (in terms of the property assessed by a particular test) less random than the tested sequence. If *p-value* = 1 for a test then the tested sequence is deemed to have perfect randomness, while if *p-value* = 0 then the sequence may be adjudged as completely non-random.

Ciphers: As candidates for the randomness tests, we consider 7 symmetric-key ciphers (e.g., AES, ARC2, ARC4, Blowfish, CAST, DES, and DES3), 2 asymmetric ciphers (e.g., PKCS#1 OAEP and PKCS#1 v1.5), and the XOR cipher. The variety in types of ciphers ensures generalizability. We also consider unencrypted data as the 11^{th} category for these tests (we refer to this as a cipher here onward for ease of explanation).6 The implementations of the considered ciphers are as available in the Python library `PyCrypto` [3].

Setup: We synthesize random data blocks of 1504 bytes each, which is approximately the MTU (maximum transmission unit) of a network packet. We use the `random.SystemRandom` class provided by Python for this purpose. We generate and encrypt 1000 such data blocks (or packets) for each cipher mentioned above. In case of the symmetric ciphers, we use a fixed key sized 128 bits. We use the Cipher-Block Chaining (CBC) encryption mode in all cases (of symmetric ciphers). For the asymmetric ones, we generate a 1024 bit RSA public/private key pair using `OpenSSL`, and then extract the public key; this public key is then used for encryption. The encrypted data generated by each cipher is converted into a separate bit-stream and fed as input to the randomness testers. We implement the randomness testers based on the Python library `r4nd0m` provided by NIST [2].

Results: The results of the randomness tests are provided in Table 1. We report scores, which are *p-values* (as mentioned before), with a precision of only up to 4 places after the decimal point. Therefore, a score of 0.0000 may not indicate a *p-value* of exactly zero, but a value which is lesser than 0.00005 (since we round up). Most tests find unencrypted data and data encrypted with XOR less than satisfactory (very low *p-value*) for usage in practical cryptographic systems, as expected. More importantly, we observe that there are discernible differences in the randomness values for data generated by the 11 ciphers considered, with respect to most of the tests. It is meaningful to ask therefore: *which test should one focus on if she intends to classify data based on the cipher used to encrypt it?* We attempt to answer this question next.

Towards identification of classification features: Of the 15 tests considered in the aforementioned experiment, the test known as the ***Spectral Test*** (also called the *Discrete Fourier Transform (DFT) Test*) is of special significance both in the literature and in this work. The roots of this test lie in a work by Coveyou et al. which suggested employing Fourier analysis to analyze the quality of random number generators [19]. Donald Knuth formalized the *Spectral Test* based on this idea and touted the test as "by far the most powerful test known", since certain weak ciphers which pass all other tests have been known to invariably fail this particular test [36]. We also observe that this test (highlighted with green in Table 1) shows the highest standard deviation in scores across all ciphers, which is indicative of high variability. Buoyed by these observations, we choose the DFT-based *Spectral Test* as the basis of defining classification features for traffic encrypted with different encryption parameters.

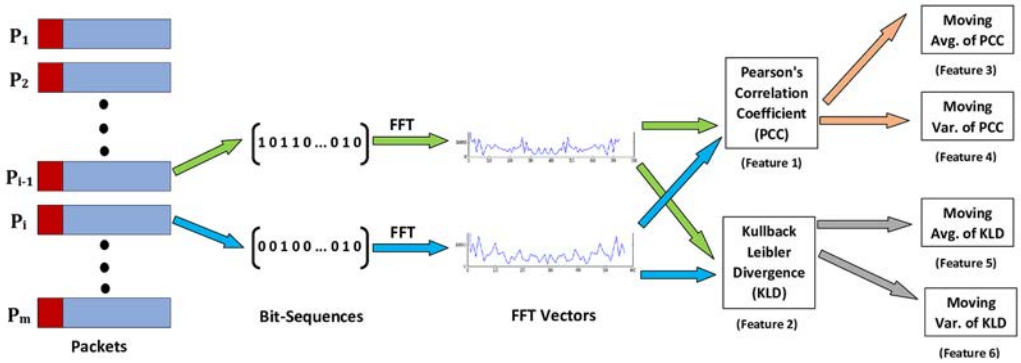
2.3 Defining Bit-Sequence Based Features

Having zeroed in on the basis of feature extraction, i.e., the DFT-based *Spectral Test*, we proceed to examine how we can define features which can effectively classify encrypted data based on encryption parameters.

Role of Fast Fourier Transform (FFT): FFT is an efficient algorithm for computing the DFT of a signal (or sequence). FFT transforms the input sequence (a bit sequence in our case) from the temporal domain to the frequency domain; the result is a vector containing the intensities of each frequency component in the original sequence. Based on our observations in the previous subsection, we already know that analyzing the FFT vectors generated from encrypted data (as done in the *Spectral Test*) is beneficial in gleaning out the randomness present in encrypted data. However, *how does one use FFT effectively to leverage this property and subsequently enable classification?*

The case for packet pairs: Randomness is likely to be induced in an individual app packet from two sources: (1) the content encapsulated in the packet, and (2) encryption. Our aim in this work is to capture the randomness due to encryption only. This requirement drives us to consider a pair of packets instead of individual packets; the idea is to eliminate the randomness due to differences in content,

Figure 1 Feature Extraction: Pictorial description of the methodology used to extract the 6 defined bit-sequence based features



as far as possible. Content similarity is likely to be the maximum in *consecutive packets* generated by an application, rather than any random pair of packets. For example, in the case of a video app, consecutive packets are likely to carry two consecutive frames of a video. Therefore, in this work, we consider consecutive packets for defining our features. We discuss possible scenarios arising out of this requirement from an implementation perspective, and ways to address those, in § 3.2.

Bit-sequence based features: We posit that the randomness due to encryption can be best leveraged by comparing the FFTs of two consecutive packets from a source app. In other words, we determine how similar or dissimilar the FFT vectors of two consecutive app packets are. The similarity (or lack thereof) in randomness is captured using 2 standard metrics employed in measuring the similarity between 2 *distributions*: (1) Pearson's Correlation Coefficient (PCC), and (2) Kullback-Leibler Divergence (KLD). In our case, these *distributions* are the Fourier transformed bit-sequences of encrypted packets generated by a cryptosystem.

In terms of implementation, we first convert the bit-sequence of each network packet into its corresponding bit-sequence in the frequency domain, by performing FFT. Thereafter, we compute the PCC and KLD of such bit-sequences of 2 consecutive packets. In order to not get restricted by sudden changes in PCC and KLD values, we also consider the moving average and moving variance of these values, which are smoother and more consistent. The entire mechanism is illustrated in Fig. 1.

Specifically, we define the following features to quantify bit-sequence similarity (or difference) between packets originating from a cryptosystem:

(1) *Pearson's Correlation Coefficient (PCC)* between the Fast Fourier Transform (FFT) of bit-sequences of 2 consecutive packets.
(2) *Pearson's Correlation Coefficient Average (PCCA)*, which is the moving average of the PCC values obtained.
(3) *Pearson's Correlation Coefficient Variance (PCCV)*, which is the moving variance of the PCC values obtained.
(4) *Kullback-Leibler Divergence (KLD)* between the FFTs of bit-sequences of 2 consecutive packets.
(5) *Kullback-Leibler Divergence Average (KLDA)*, which is the moving average of the KLD values obtained.
(6) *Kullback-Leibler Divergence Variance (KLDV)*, which is the moving variance of the KLD values obtained.

Table 2 Classification results (precision ($\mathcal{P}$), recall ($\mathcal{R}$), f1-score ($\mathcal{F}$), accuracy ($\mathcal{A}$)) obtained using 3 different classifiers for 3 variations of packet-sizes, for packets originating from 11 sources (10 encrypted using different ciphers, and 1 unencrypted). Best classification performance is achieved using the Random Forest Classifier (highlighted in green).

Classifier / Packet-size	**All Equal**	**Equal for Source**	**All Different**
Metrics	$\mathcal{P}, \mathcal{R}, \mathcal{F}, \mathcal{A}$	$\mathcal{P}, \mathcal{R}, \mathcal{F}, \mathcal{A}$	$\mathcal{P}, \mathcal{R}, \mathcal{F}, \mathcal{A}$
Random Forest	0.89, 0.89, 0.89, **0.89**	0.96, 0.96, 0.96, **0.96**	0.90, 0.89, 0.89, **0.89**
Decision Tree	0.88, 0.87, 0.87, **0.87**	0.93, 0.93, 0.93, **0.93**	0.87, 0.86, 0.86, **0.86**
K Nearest Neighbors	0.81, 0.81, 0.81, **0.81**	0.93, 0.93, 0.93, **0.93**	0.72, 0.72, 0.72, **0.72**
Gaussian Naive Bayes	0.73, 0.61, 0.60, **0.61**	0.88, 0.84, 0.82, **0.84**	0.75, 0.71, 0.70, **0.71**
M[1]layer Perceptron (Deep)	0.65, 0.69, 0.66, **0.69**	0.89, 0.89, 0.89, **0.89**	0.57, 0.61, 0.58, **0.61**
Logistic Regression	0.58, 0.62, 0.57, **0.62**	0.69, 0.66, 0.63, **0.66**	0.53, 0.53, 0.50, **0.53**
Support Vector Machine	0.50, 0.46, 0.41, **0.46**	0.77, 0.71, 0.66, **0.71**	0.46, 0.44, 0.39, **0.44**
AdaBoost	0.24, 0.28, 0.21, **0.28**	0.36, 0.29, 0.22, **0.29**	0.34, 0.36, 0.31, **0.36**

Let us denote the FFT vectors corresponding to the n^{th} packet and $(n-1)^{th}$ packet as p^n and p^{n-1} respectively. Pearson's correlation coefficient (PCC) between the FFTs corresponding to the n^{th} and the $(n-1)^{th}$ packets is defined as:

$$PCC_n = \frac{\sum_i (p_i^n - \overline{p^n}).(p_i^{n-1} - \overline{p^{n-1}})}{\sqrt{\sum_i (p_i^n - \overline{p^n})^2 . \sum_i (p_i^{n-1} - \overline{p^{n-1}})^2}} \quad (1)$$

The features derived from PCC, i.e. PCCA and PCCV, corresponding to the n^{th} packet in sequence, are defined as follows:

$$PCCA_n = \frac{1}{n} \sum_{i=0}^{i=n} PCC_i \quad (2) \qquad PCCV_n = \frac{1}{n} \sum_{i=0}^{i=n} (PCC_i - PCCA_i)^2 \quad (3)$$

Kullback-Leibler divergence between the FFTs corresponding to the n^{th} and the $(n-1)^{th}$ packets, is defined as follows:

$$KLD_n = \sum_i p_i^n . log \frac{p_i^n}{p_i^{n-1}} \quad (4)$$

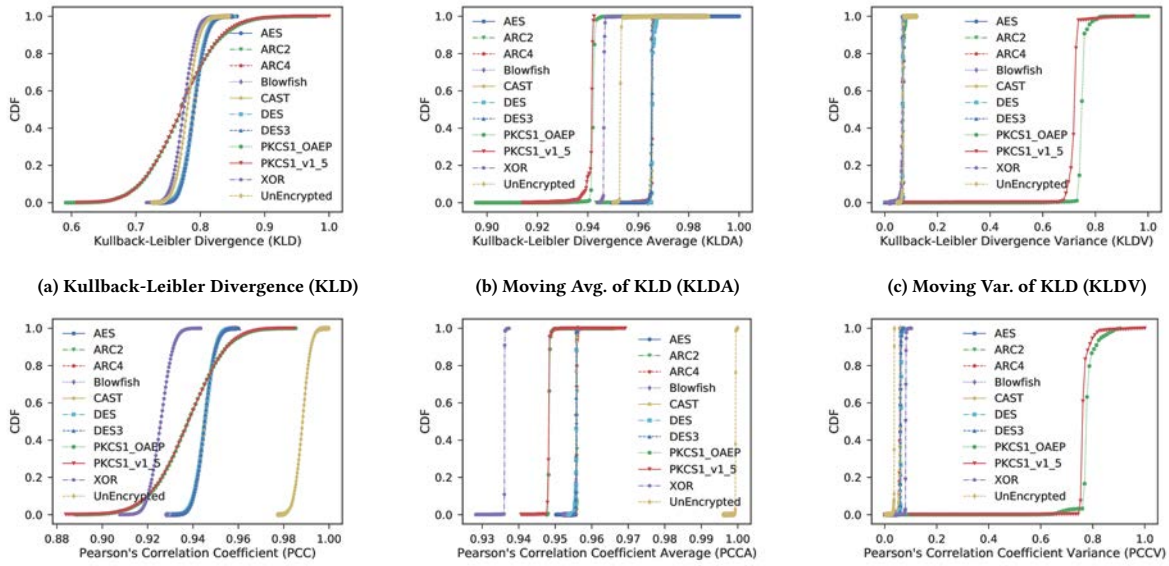
The features derived from KLD, i.e. KLDA and KLDV, corresponding to the n^{th} packet in the traffic flow, are defined as follows:

$$KLDA_n = \frac{1}{n} \sum_{i=0}^{i=n} KLD_i \quad (5) \qquad KLDV_n = \frac{1}{n} \sum_{i=0}^{i=n} (KLD_i - KLDA_i)^2 \quad (6)$$

2.4 Cipher and Key-Size Classification

In this subsection, we present two validation experiments, i.e., cipher classification (expt. 1) and key-size classification (expt. 2), to establish the efficacy of the aforementioned bit-sequence based features.

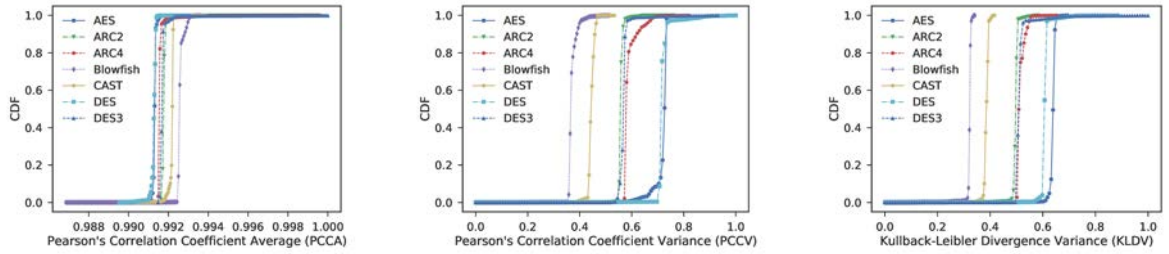
Ciphers: For expt. 1, we consider the same sources as in § 2.2, i.e., 7 symmetric-key ciphers (e.g., AES, ARC2, ARC4, Blowfish, CAST,



(a) Kullback-Leibler Divergence (KLD) (b) Moving Avg. of KLD (KLDA) (c) Moving Var. of KLD (KLDV)

(d) Pearson's Correlation Coeff. (PCC) (e) Moving Avg. of PCC (PCCA) (f) Moving Var. of PCC (PCCV)

Figure 2 Distribution (CDF) of bit-sequence based features: All 6 features – KL divergence (KLD), moving average of KL divergence (KLDA), moving variance of KL divergence (KLDV), Pearson's correlation coefficient (PCC), moving average of Pearson's correlation coefficient (PCCA), and moving variance of Pearson's correlation coefficient (PCCV) – are defined on the FFTs of the bit-sequences of packet payloads. These features capture the similarities (or dissimilarities) among bit-sequences of packet payloads generated using different ciphers (note that PKCS#1 v1.5 and PKCS#1 OAEP are represented as PKCS1_v1_5 and PKCS1_OAEP, respectively, in the plots).



(a) Moving Avg. of PCC (PCCA) (b) Moving Avg. of KLD (KLDA) (c) Moving Var. of KLD (KLDV)

Figure 3 Distribution (CDF) of bit-sequence based features PCCA, KLDA, and KLDV for 7 encryption algorithms, namely AES, ARC2, ARC4, Blowfish, CAST, DES, DES3. It can be observed that the packets generated for one encryption algorithm can be differentiated from others, on the basis of these 3 features.

DES, and DES3), 2 asymmetric ciphers (e.g., PKCS#1 OAEP and PKCS#1 v1.5), the XOR cipher, and unencrypted data acting as the 11^{th} source. We consider only the strongest cipher, i.e., AES, for expt. 2.

Key-sizes: In expt. 1, we use a fixed 128 bit key for symmetric ciphers, and a RSA public key extracted from a 1024 bit public/private key pair for asymmetric ciphers, as described in § 2.2. For expt. 2, we consider 3 different key-sizes of AES, i.e., 128 bits, 192 bits, and 256 bits. The keys are randomly generated using the Python `random.SystemRandom` class.

Setup: As in § 2.2, we used the Python package `PyCrypto` to encrypt each synthesized packet [3]. For both expts. 1 and 2, we generated 10,000 packets from each aforementioned source. In case of expt. 1, we repeated the experiments for 3 variations in terms of packet-size: (1) equal packet-size for all 11 sources, (2) equal packet-size for each source (different for different sources), and (3) random packet-size for each packet, irrespective of the packet source.

Results for cipher classification: We present the classification results for 8 different classifiers in Table 2. Multi-class (11 classes) classification is performed, all 6 features based on bit-sequence similarity are used, and 10-fold cross validation is employed to guard against over-fitting. We observe that the Random Forest Classifier (RFC) achieves best classification performance, with a worst-case accuracy of 89%. We also observe that the distribution of packet-sizes has an impact on classification, i.e., the packets can be more easily distinguished (96% in the best case) when packet-sizes are same across the packets from one source but different from the packets of other sources.

Results for key-size classification: Best performance is once again achieved by RFC, with an accuracy of 83% (with precision,

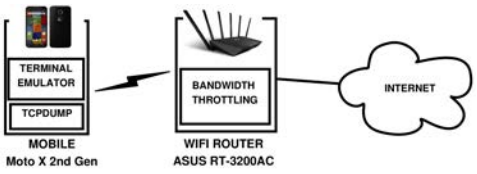


Figure 4 In-laboratory testbed setup for data collection: An ASUS RT-3200AC router connected to the Internet, acts as the gateway device. Bandwidth throttling is performed by setting router configuration parameters appropriately. A Moto X 2^{nd} generation Android smartphone running on Android 6.0 (Marshmallow) is connected to the router over WiFi. Mobile applications are executed on this phone, and data is collected using the *tcpdump* tool.

recall, and F1-score all at 0.83). We omit rest of the results (which are similar as for expt. 1) for brevity.

2.5 Bit-Sequence Features: Detailed Analysis

Having established that our proposed features are able to distinguish among data encrypted with different ciphers and different key-sizes, we perform a deeper analysis of those features. We plot the CDF for all 6 features for packets generated using each of the 11 sources in Fig. 2 (case (2), i.e., equal packet-size for each source is considered; the observations are similar in the other 2 cases). We observe that the distributions for PKCS#1 OAEP, PKCS#1 v1.5, XOR, and unencrypted can be easily differentiated from the rest of the cases (KLDV and PCCA showing the most prominent distinguishability). In order to analyze the other 7 encryption algorithms more closely, we normalize for those and plot the CDF of PCCA, PCCV, and KLDV in Fig. 3. We observe that even these 7 algorithms can be easily distinguished from each other with a classifier of higher arithmetic precision, on the basis of these 3 features.

So far, we have analyzed the effectiveness of our proposed set of features in classifying data encrypted with multiple ciphers/key-sizes. However, this ability is not merely limited to varying ciphers and key-sizes, but generalizes to any customizable parameter in a cryptosystem implementation that may give rise to a discernible difference in the randomness of the encrypted data. Other parameters (besides cipher and key-size) include the default OS library, the TLS library and its version being used, etc. In the following sections, we illustrate how these bit-sequence based features may be used to classify encrypted data generated from a large number of mobile apps, based on their ability to leverage variability in cryptosystem implementations.

3 EXPERIMENTAL SETUP AND DATASET

In this section, we explain our experimental setup for performing large-scale app traffic classification, and also describe the dataset collected using this setup.

3.1 Experimental Setup

We describe the experimental setup adopted by us, and the driving factors behind it, in this subsection.

Setup considerations: In order to generalize our results on mobile app traffic, we collected traffic traces under controlled operating conditions. Important control parameters included background traffic, channel conditions, and traffic from advertisements embedded in the traffic of the intended application. Even if all possible care is taken to avoid background traffic, it can still occur from services that the OS runs for optimizations and communications with the server. Also, variation in channel conditions may lead to differences in the way certain apps behave, e.g., video streaming apps may adapt to a different bit-rate when bandwidth varies. Similarly, advertisements running within the app may contaminate data collected from it. Keeping these factors in mind, we implemented an in-laboratory testbed to enable controlled collection of packet traces from a large number of mobile applications.

Testbed setup: The testbed setup is shown in Fig. 4. The end device connects to the ISP's network through a wireless router. The wireless router is instrumented such that we can control the bandwidth between the router and the end device. We used a Motorola Moto X 2^{nd} generation Android smartphone as the end device where we execute each mobile app separately. This device is rooted to allow trace collection using `tcpdump`. Android `Terminal Emulator` app provides the interface to execute `tcpdump`. The router is an ASUS RT-3200AC router with IEEE 802.11ac Wi-Fi standard. This router provides the *Adaptive QoS* feature, which allows manual configuration of the *Download Bandwidth* field to specify a controlled download rate.

Bandwidth scenarios: We collect data for each app under 4 bandwidth scenarios (by throttling at the router) – 0.5 Mbps, 1 Mbps, 4 Mbps, and 16 Mbps – to capture changes in app behaviour, if any, under varying bandwidth conditions.

Logs maintained during trace collection: We log the following information during data collection:

- ***Log 1:*** IPs for all client devices used in our experiments, and the corresponding app playing on each IP.
- ***Log 2:*** Start and end time for embedded advertisements, during execution of the app.

The aforementioned information enables us to discard noisy data from the accumulated traffic, in the manner described next.

Packet labeling: In order to associate each packet with a distinct label for our experiments, we first differentiate the video traffic flows based on 5-tuple information <*source IP, source port, destination IP, destination port, transport protocol*>, present in the header of each packet. Based on our knowledge of client IP addresses from ***Log 1***, we selectively preserve flows which correspond to download traffic, i.e., from the content server to the client app. We ensure that network control packets are eliminated from our dataset – such packets include ARP, DNS, IGMP, ICMP, NTP, DB-LSP-DISC (Dropbox Lan Sync Protocol), SYN, ACK (non-piggybacked), etc. – these packets do not carry any useful content, but appear in large numbers in the traces. Furthermore, we utilize the logged timing information in ***Log 2*** to discard packets which do not correspond to the actual app content, but to associated events (such as ads). Subsequently, we tag each packet with its source application, based on our knowledge of the client IP-app correspondence – cumulatively, this forms our dataset.

3.2 Android Application Dataset

In this subsection, we describe the dataset collected by us using the setup introduced in § 3.1.

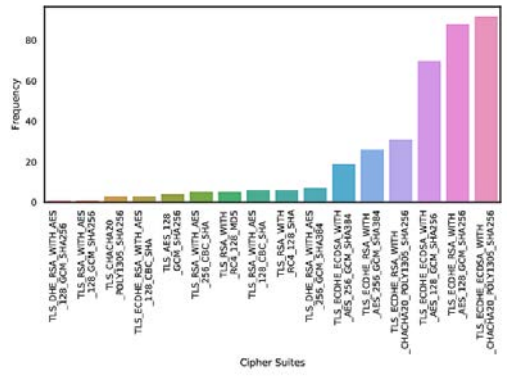


Figure 5 Distribution of cipher suites used by applications in our dataset.

Android app selection: We select 175 Android applications based on the *Top Free in Android Apps* selection in the Google Play Store[3]. The apps comprise a variety of categories, including but not limited to, video, audio, games, messaging, social networking, news, health, and static content-based apps.

Diversity in cipher suites: Razaghpanah et al. established in their analysis that different applications in the Android app store use different cipher suites, TLS libraries, OS defaults, etc. [41]. We perform a similar analysis of our dataset in terms of ciphers suites used and supported. We segregate out the `Server Hello` and `Client Hello` TLS handshake messages from our dataset, and parse them to find out cipher suites used by different applications. Fig. 5 shows the distribution of cipher suites used for encrypting traffic during the course of data collection. Fig. 6 on the other hand, shows the distribution of cipher suites which are supported by the application and may be used depending upon platform support, preference, or random choice. We observe from the figures that there is significant diversity in the cipher suites used for different mobile apps, which corroborates with the analysis in [41]. Furthermore, it is also important to note that cipher suites are not the only encryption parameters which affect randomness induced in the data; OS and TLS libraries, various configurable parameters on the client and server sides, etc. also impacts it. Our features are designed to leverage any such randomness occuring due to any combination of parameters in the cryptographic implementation of an application.

Choosing pairs of packets: We mentioned in § 2.3 that the ideal way of choosing pairs of packets in order to compute our proposed bit-sequence based features is to choose consecutive packets in a flow. We employ this idea (of consecutive packets) in the experiments conducted in this paper. The rationale is that in a real system, even if the client port is common (80 for HTTP and 443 for HTTPS) and the IPs are unrecognized for multiple simultaneously running applications, it would still be possible to segregate a flow based on its 5-tuple combination in most situations. There may be two exceptions, however: (1) when the first packet in a flow is encountered and it cannot be readily attributed to an application, and (2) when simultaneous flows end up with the same 5-tuple combination (e.g., when using a proxy). In both cases, we propose storing and using recent *reference* packets from all candidate applications, and using those for computation of the bit-sequence based features. The application for which the values correlate most closely (as returned by a trained classifier), is the likely target application.

In order to illustrate the frequency of case 1 (first packet of a flow) in our dataset, we plot the number of packets per flow for each application in Fig. 7. We observe that most (above 90%) of apps send atleast 100 packets in a flow, while around 50% send at least 1,000 packets. Around 15% of applications send upward of 10,000 packets in a single flow. This illustrates that in case of most applications the computation of bit-sequence based features does not incur any additional overhead, i.e., first packet instances are few and far between.

[3]This list changes frequently based on Google's app ranking system. Our selection was based on the status of this list on 15-April-2018.

4 EVALUATION

In this section, we evaluate the efficiency of our proposed bit-sequence based features in classifying traffic from a large number of applications.

4.1 Baseline (Packet-Level) Features

We introduce packet-level features in this subsection, such as those based on packet-size and packet inter-arrival time, which have been widely used in the literature for app-traffic classification [7, 27, 42, 46]. These features serve as baselines in our study. Along with the raw packet-size and inter-arrival time, we also consider the moving average and moving variance of these features. This is to account for sudden changes in the raw values, e.g., when the last packet in a burst is sent, it may be of an uncharacteristically lower size, or the first packet in a burst may have a significantly higher inter-arrival time. Moving average and variance values, which are more regular and smoothened, better capture the characteristic nature of app traffic.

Packet-Size Based Features: We use the following packet-size based features:

(1) *Packet-size (PS)*, which is directly the packet-size observed at the network (IP) layer.
(2) *Packet-size Moving Average (PSMA)*, which is the moving average of PS values for a traffic flow.
(3) *Packet-size Moving Variance (PSMV)*, which is the moving variance of PS values for a specific traffic flow.

PSMA and PSMV corresponding to the n^{th} packet in sequence of traffic flow, are formulated as follows:

$$PSMA_n = \frac{1}{n}\sum_{i=0}^{i=n} PS_i \quad (7) \qquad PSMV_n = \frac{1}{n}\sum_{i=0}^{i=n} (PS_i - PSMA_i)^2 \quad (8)$$

Packet Inter-Arrival Time Based Features: We use the following parameters based on temporal traffic behavior:

(1) *Inter-arrival time (IAT)*, which is merely the difference between arrival time of two consecutive packets belonging to the same flow.
(2) *Inter-packet timing average (IPTA)*, which is the moving average of IAT values for a traffic flow.
(3) *Inter-packet timing variance (IPTV)*, which is the moving variance of IAT values for a particular traffic flow.

The features identified above, i.e., IAT, IPTA, and IPTV, of the n^{th} packet in sequence within the flow, are formulated as follows:

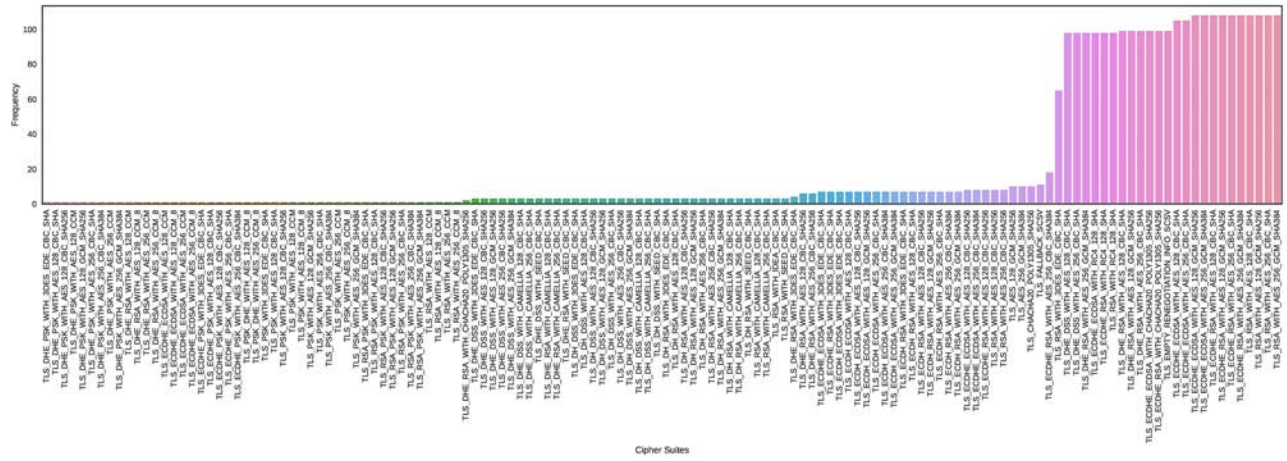


Figure 6 Distribution of cipher suites supported by applications in our dataset.

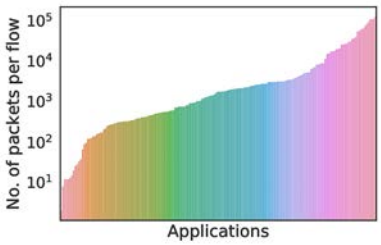


Figure 7 Distribution of number of packets per flow for applications

Table 3 Classification performance (precision ($\mathcal{P}$), recall ($\mathcal{R}$), f1-score ($\mathcal{F}$), accuracy ($\mathcal{A}$)) for classification of traffic from 175 mobile apps. Bit-sequence based features perform better than the baselines in terms of both F1-score and accuracy. The scores boost significantly when our proposed features are used in conjunction with baseline packet-level features.

Features / Classifier	**Random Forest**	**Decision Tree**
Metrics	$\mathcal{P}$, $\mathcal{R}$, $\mathcal{F}$, $\mathcal{A}$	$\mathcal{P}$, $\mathcal{R}$, $\mathcal{F}$, $\mathcal{A}$
Packet-Size Features (PSF)	0.32, 0.27, 0.27, **0.27**	0.35, 0.29, 0.29, **0.29**
Inter-Arrival Time Features (ITF)	0.33, 0.10, 0.12, **0.10**	0.37, 0.11, **0.13**, **0.11**
Packet-Level Features (PSF+ITF)	0.76, 0.62, 0.65, **0.62**	0.78, 0.60, **0.64**, **0.60**
Bit-Sequence Features (BSF)	0.68, 0.71, 0.69, **0.71**	0.66, 0.69, 0.67, **0.69**
Bit-Sequence + Packet-Level Features (PSF+ITF+BSF)	0.97, 0.96, 0.96, **0.96**	0.95, 0.95, 0.95, **0.95**

$$IAT_n = t_n - t_{n-1} \quad (9) \qquad IPTA_n = \frac{1}{n}\sum_{i=0}^{i=n} IAT_i \quad (10)$$

$$IPTV_n = \frac{1}{n}\sum_{i=0}^{i=n} (IAT_i - IPTA_i)^2 \quad (11)$$

where t_n and t_{n-1} are arrival time instances for the n^{th} and $(n-1)^{th}$ packets respectively.

4.2 Classification Methodology

We discuss the classification methodology employed for classification of packets from 175 mobile apps, followed by results.

Overcoming class imbalance: We train our classifiers on 10,000 packets from each app. In order to avoid bias due to class imbalance, we random-sample an equal number of packets (10,000) from the data collected for all those apps, where the total number of packets is higher than 10,000. In the few cases (5 apps) where lesser than 10,000 packets are available, we employ SMOTE [14] and oversample the datasets, so as to balance them.

Guarding against overfitting: In order to minimize the chances of over-fitting the data, we employ k-fold cross validation (we set $k = 10$ in our experiments).

4.3 Classification Results

Overall classification: The classification results for data collected from 175 apps are presented in Table 3. We present results for the following baselines: (1) Packet-size based features only (PS, PSMA, PSMV), (2) Inter-arrival time based features only (IAT, IPTA, IPTV), (3) Bit-sequence based features only (PCC, PCCA, PCCV, KLD, KLDA, KLDV), and (3) All packet-level features, i.e., a combination of packet-size based features and inter-arrival time based features (PS, PSMA, PCCV, IAT, IPTA, IPTV). Results for only the two best performing classifiers (Random Forest and Decision Tree) have been reported. We observe that using only packet-size based features results in an accuracy score of 0.29, while only inter-arrival time based features result in an ever lower accuracy of 0.11 (in the best cases, i.e., using the Decision Tree classifier). When combined, however, the packet-level features together result in a much higher accuracy of 0.60, which is still, however, not satisfactorily high.

Our proposed bit-sequence based features perform significantly better than either packet-size based or inter-arrival time based features, in terms of both F1-score and accuracy. In fact, they perform even better than all packet-level features combined (69% vs. 65% in terms of F1-score, and 71% vs. 62% in terms of accuracy). Considering that we perform multi-class classification with 175 classes, this is a significant result, and substantiates the effectiveness of our proposed features in differentiating apps based on differences in TLS implementations. The classification results are boosted heavily – in fact to 95% F1-score as well as accuracy, in the best case – when packet-level features are employed in conjunction with bit-sequence based features, which is our recommended strategy for mobile app traffic classification. A classifier performing with such

accuracy may be used by a network administrator in fingerprinting and monitoring heterogenous traffic from a vast number of apps.

Category-wise evaluation: We evaluate our proposed features for another important practical use-case. Instead of aggregate traffic, a network administrator may only be interested in identifying traffic from a specific category. For example, in an enterprise setting, she may try to enforce a policy where all video traffic is allocated lesser bandwidth to avoid overburdening critical applications. In this vein, we divide the entire dataset of apps into 5 categories, namely, video, audio, browsing, gaming, and messaging. The *browsing* category consists of apps used for email, download, sync, etc. apart from web browsing; the other 4 category names are self-explanatory. We then attempt to perform inter-category classification (i.e., classify among the 5 categories), as well as intra-category classification (i.e., classify among apps belonging to one particular category). The classification results are presented in Table 4.

We observe that we can achieve 90% accuracy (using only bit-pattern based features) in classifying heterogenous traffic into categories (row 1). Classification performance further improves when traffic from apps within a particular category are classified; minimum accuracy (using RFC) is 93%, which once again is a significant results considering the scale of classification.

Classification running time: Since the classifier needs to run as part of a live system, it is also important to evaluate the running time for per-packet classification. We executed a separate batch of experiments, where we trained the classifiers with 90% of the data, and tested with the remaining 10%. This enabled fine-grained time-keeping, which we achieved using the Python `timeit` module. These experiments were executed in a Ubuntu 16.04 workstation, with 64 GB RAM, and a 3.0 GHz Intel Core i5 processor. We observe that while RFC consumed 14.16 microseconds per packet, the Decision Tree classifier consumed only 0.84 microsecond. We note that these time quanta are well within the bounds of the RTT of packets arriving from a distant content server, which implies that our classification strategy can be employed in a real-life setting, such as in a smart-QoS enabled modern home router.

5 DISCUSSION

In this section, we discuss some of the major learnings from and limitations of this work, as well as the potential impact it may have in inducing further research in the area.

Real-time implementation: We show in § 4.3 that classification at per-packet granularity is compliant with the timing requirements of a live networked system. However, we do not comment on the time consumed in the training process, when either training frequency is high, or training data volume is extremely large. For example, network administrators intent on classifying and blocking malicious apps may need to avail frequent rounds of training, since new apps crop up everyday on the app store. Optimizations such as implementing FFT in hardware could be availed to speed up computations in such scenarios [6].

Attacks around bit-sequence based features: While our use-cases in this paper revolve around the opportunities bit-sequence based features could provide network administrators, the flip-side of the coin is that the same opportunities may be exploited by attackers. For example, these features could be leveraged to infer user actions from mobile phone traffic, which could lead to severe breach of privacy for users [18, 37, 49]. Applications which authenticate users based on their app activity history [21, 30], may also be fooled if app activities could be gleaned out using our proposed features.

Protection against obfuscation: Some works have pointed out the privacy risks involved when statistical analysis of packet-level features is used to differentiate traffic [32, 51]. These studies, including a recent work [13] propose obfuscating such features so that classifiers may not be able to extract useful information out of application traffic. The obfuscation is implemented in a proxy, which fragments incoming packets and regulates arrival rate to make one class of traffic look like another. Our classification mechanism based on randomness of security implementations, is however, immune to such packet-level obfuscation.

Tool for side-channel attacks: It follows from the previous discussion, therefore, that along with packet-level and traffic meta-data (e.g., direction of traffic) based features [17, 45, 48], bit-sequence based features can also pose threats via side-channel attacks. It is important therefore, to further the research agenda in this direction, and devise methods to either exploit or alleviate such attacks, as necessary for the application.

Impact of app updates: App updates may include updates to their encryption implementations as well. While such updates can affect classification performance of our trained model, as long as there exist differences in such implementations, it is possible to retrain the classifier to recover performance. We argue that such differences will continue to exist due to the non-standard nature of app stores, and deficiencies in computational capabilities of low-end devices.

6 RELATED WORK

Classifying traffic from mobile applications has spurred active research recently due to its practical necessity, as well as, the challenges arising from lack of features suitable for traffic classification. First, we look at classification strategies which have addressed generic web traffic. Next, we present studies specific to mobile traffic, and discuss their advantages and limitations.

6.1 Generic Web Traffic Classification

A large body of work has addressed traffic classification on the Internet over a number of years. Continuously changing landscapes in the way data is delivered to the end-users (e.g., content delivery networks (CDNs), cloud), have prompted researchers to revisit old techniques, and invent newer approaches more commensurate with the trends prevalent during the time of their studies.

Deep Packet Inspection (DPI) based Techniques: Several approaches have been suggested for web traffic classification using packet-inspection based methods. Finsterbusch et al. compiled a comprehensive survey of these techniques – however, they note, like many others, that DPI based approaches either perform abysmally, or fail entirely when traffic is encrypted [26].

Statistical Information based Techniques: Most studies which do not rely on packet-inspection, suggest exploiting side-channel information, and statistical properties of traffic, which remain unaffected by encryption. An MIT tech review article highlighted the importance of statistical tricks in extracting user footprints from encrypted communications [5]. Some statistical approaches rely on

Table 4 Classification results (precision ($\mathcal{P}$), recall ($\mathcal{R}$), f1-score ($\mathcal{F}$), accuracy ($\mathcal{A}$)) using only bit-sequence based features for 2 classification tasks: (1) classification of aggregate traffic into app categories (shown in the first row), and (2) classification of category traffic into constituent applications. Best classification performance is achieved using the Random Forest Classifier (highlighted in green).

Clf. task / Classifier	**Random Forest**	**Decision Tree**	**K-Nearest Neighbors**	**Naive Bayes**	**M'layer Perceptron (Deep)**	**Logistic Regression**	**Support Vector**
Metrics	$\mathcal{P}$, $\mathcal{R}$, $\mathcal{F}$, $\mathcal{A}$	$\mathcal{P}$, $\mathcal{R}$, $\mathcal{F}$, $\mathcal{A}$	$\mathcal{P}$, $\mathcal{R}$, $\mathcal{F}$, $\mathcal{A}$	$\mathcal{P}$, $\mathcal{R}$, $\mathcal{F}$, $\mathcal{A}$	$\mathcal{P}$, $\mathcal{R}$, $\mathcal{F}$, $\mathcal{A}$	$\mathcal{P}$, $\mathcal{R}$, $\mathcal{F}$, $\mathcal{A}$	$\mathcal{P}$, $\mathcal{R}$, $\mathcal{F}$, $\mathcal{A}$
App categories (5)	0.90, 0.90, 0.90, **0.90**	0.88, 0.88, 0.88, **0.88**	0.77, 0.77, 0.77, **0.77**	0.47, 0.46, 0.41, **0.46**	0.42, 0.47, 0.42, **0.47**	0.38, 0.42, 0.36, **0.42**	0.37, 0.40, 0.37, **0.40**
Video	0.95, 0.95, 0.95, **0.95**	0.95, 0.95, 0.95, **0.95**	0.72, 0.72, 0.72, **0.72**	0.33, 0.30, 0.27, **0.30**	0.27, 0.29, 0.27, **0.29**	0.22, 0.19, 0.15, **0.19**	0.37, 0.40, 0.37, **0.40**
Audio	0.99, 0.99, 0.99, **0.99**	0.98, 0.98, 0.98, **0.98**	0.91, 0.90, 0.90, **0.90**	0.39, 0.35, 0.32, **0.35**	0.46, 0.43, 0.42, **0.43**	0.41, 0.38, 0.36, **0.38**	0.22, 0.20, 0.12, **0.20**
Browsing	0.94, 0.93, 0.93, **0.93**	0.92, 0.92, 0.92, **0.92**	0.68, 0.69, 0.68, **0.69**	0.20, 0.19, 0.16, **0.19**	0.13, 0.19, 0.13, **0.19**	0.16, 0.16, 0.13, **0.16**	0.22, 0.21, 0.22, **0.21**
Gaming	0.96, 0.96, 0.96, **0.96**	0.94, 0.94, 0.94, **0.94**	0.61, 0.62, 0.60, **0.62**	0.14, 0.15, 0.12, **0.15**	0.11, 0.12, 0.10, **0.12**	0.13, 0.14, 0.12, **0.14**	0.11, 0.10, 0.11, **0.12**
Messaging	0.99, 0.99, 0.99, **0.99**	0.98, 0.98, 0.98, **0.98**	0.91, 0.91, 0.91, **0.91**	0.65, 0.52, 0.52, **0.52**	0.39, 0.41, 0.35, **0.41**	0.48, 0.42, 0.38, **0.42**	0.46, 0.41, 0.38, **0.41**

carefully constructed rules for classification, while others employ machine learning – we report these cases separately.

Rule based Approaches: Sun et al. presented a method for statistical identification of browsed web applications, by studying the various unique web objects (e.g., images, stylesheets, etc.) fetched during the launch of a webpage [47]. This approach is clearly outdated, since most websites serve dynamic content nowadays, where fetched objects would continually change, sometimes even within seconds. Gong et al. proposed using side-channel attacks to determine round-trip times (RTTs), and then clustering traffic with similar RTTs [29]. A similar approach was adopted by Chen et al., where side-channel leaks in terms of packet-size, arrival time, and stateful interactions between servers and clients of web applications, were exploited [15]. Piskac et al. suggested protocol detection from web traffic, again using packet-size and arrival time values [40]. Detection of Domain Generation Algorithm (DGA) based botnets was proposed using DNS traffic analysis in a recent work by Wang et al. [50]. Another recent work studied privacy vulnerabilities in a IoT based smart home system, made possible by traffic analysis [9].

Machine Learning based Approaches: Traffic classification using supervised learning has been a popular approach – the major task in such an approach is to identify discriminative features from encrypted network traffic. Some of the earlier works can be found in a survey by Nguyen et al. in [39], and references therein. Jesudasan et al. identified generic features (based on packet-size and arrival time) across different versions of Skype that could be used to classify Skype flows [33]. Bujlow et al. proposed a learning model with features based on individual and aggregate payload sizes, along with IP and port no. based rules [11]. Recently, Shi et al. have also shown that encrypted, and even tunneled (using a Virtual Private Network (VPN)) video sources, can be identified by analyzing simple features derived from packet-sizes and temporal patterns of packet arrival [10, 43]. As an application of network traffic analysis, a recent work by Das et al. proposed privacy-aware localization using a supervised learning approach [22].

While some statistical observations for generic web traffic may still hold true in case of mobile traffic, there are differences due to buffer and bandwidth constraints on a mobile device, and how service providers tend to optimize traffic for mobile accordingly. We leverage upon the common statistical properties by identifying those (with careful analysis of mobile app traffic), and address the differences by defining novel features that can aid in classification.

6.2 Mobile Traffic Classification

Studies on mobile traffic classification can be broadly divided based on whether those targeted unencrypted or encrypted traffic.

Unencrypted Traffic: Approaches that assume unencrypted traffic rely on inspection of fields in the HTTP header. Hur et al. exploited the user agent field, and other common strings found in the HTTP headers of packets generated by the same application [31]. Dai et al. showed that by executing different apps in an emulator and gathering corresponding network traces, it is possible to build signatures for identifying specific traffic sources [20]. Yao et al. reduced the overhead of building signatures where they used context of signatures to improve identification accuracy and scalability [54]. Automatic generation of classifiers for traffic identification has been presented in [16, 53]. AppPrint also uses extensive traffic observations to build signatures based on header fields [38]. Since these works rely on information that must be read from the packets, they are ill-suited for classifying encrypted traffic.

Encrypted Traffic: Stöber et al. proposed classification of background traffic generated from a smartphone, by studying patterns in packet level features (e.g., packet-size and packet arrival instances), and burst level features (e.g., mean, median, etc. of packet level features) [46]. Alan et al. limited their study to using only launch time traffic for building supervised classifiers for Android app identification [7]. Mobile messaging apps were studied by Fu et al., and a machine learning based solution was provided using packet-level features [27]. Another work defined an end-to-end system to enable fine-grained bandwidth provisioning for video app traffic in absence of background traffic [42].

7 CONCLUSION

Mobile traffic classification is an important task for network administrators in order to achieve policy enforcement and service provisioning. However, due to lack of unique signatures and especially due to encryption, mobile app traffic classification is rendered difficult. While statistical analysis based on packet-size and arrival time of packets has shown a lot of promise, existing techniques do not scale well to traffic from a large number of applications. In this paper, we worked on the notion of exploiting encryption itself (more specifically, TLS implementations) to classify mobile app traffic at a large scale. We proposed a set of novel bit-sequence based features, which can classify data encrypted with different ciphers and key-sizes, with 89% and 83% accuracy (best case), respectively. We show that using these features in conjunction with existing packet-level based features, it is possible to classify traffic from 175 Android applications with a best case accuracy of 95% (as opposed to a baseline accuracy of 62%). The features also perform well in achieving inter-category (90% accuracy) and intra-category traffic classification (min. accuracy of 93%), using only bit-sequence based features.

REFERENCES

[1] [n. d.]. Cisco Visual Networking Index White Paper. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html

[2] [n. d.]. Github Library: r4nd0m. https://github.com/StuartGordonReid/r4nd0m

[3] [n. d.]. Package Crypto: Python Cryptography Toolkit. https://www.dlitz.net/software/pycrypto/api/current/

[4] [n. d.]. Random Bit Generation: Guide to the Statistical Tests. https://csrc.nist.gov/Projects/Random-Bit-Generation/Documentation-and-Software/Guide-to-the-Statistical-Tests

[5] [n. d.]. Statistical Tricks Extract Sensitive Data from Encrypted Communications.

[6] Berkin Akin, Franz Franchetti, and James C Hoe. 2014. Understanding the design space of dram-optimized hardware FFT accelerators. In *Application-specific Systems, Architectures and Processors (ASAP), 2014 IEEE 25th International Conference on*. IEEE, 248–255.

[7] Hasan Faik Alan and Jasleen Kaur. 2016. Can Android Applications Be Identified Using Only TCP/IP Headers of Their Launch Time Traffic?. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. ACM, 61–66.

[8] Mohammed M Alani. 2010. Testing randomness in ciphertext of block-ciphers using DieHard tests. *Int. J. Comput. Sci. Netw. Secur* 10, 4 (2010), 53–57.

[9] Noah Apthorpe, Dillon Reisman, and Nick Feamster. 2017. A Smart Home is No Castle: Privacy Vulnerabilities of Encrypted IoT Traffic. *arXiv preprint arXiv:1705.06805* (2017).

[10] Subir Biswas and Yan Shi. 2016. Protocol independent identification of encrypted video traffic sources using traffic analysis. In *Communications (ICC), 2016 IEEE International Conference on*. IEEE, 1–6.

[11] Tomasz Bujlow, Tahir Riaz, and Jens Myrup Pedersen. 2012. A method for classification of network traffic based on C5. 0 Machine Learning Algorithm. In *Computing, Networking and Communications (ICNC), 2012 International Conference on*. IEEE, 237–241.

[12] Arthur Callado, Carlos Kamienski, Géza Szabó, Balázs Péter Gerö, Judith Kelner, Stênio Fernandes, and Djamel Sadok. 2009. A survey on internet traffic identification. *Communications Surveys & Tutorials, IEEE* 11, 3 (2009), 37–52.

[13] Louma Chaddad, Ali Chehab, Imad H Elhajj, and Ayman Kayssi. 2018. App traffic mutation: Toward defending against mobile statistical traffic analysis. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE.

[14] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.

[15] Shuo Chen, Rui Wang, XiaoFeng Wang, and Kehuan Zhang. 2010. Side-channel leaks in web applications: A reality today, a challenge tomorrow. In *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 191–206.

[16] Yeongrak Choi, Jae Yoon Chung, Byungchul Park, and James Won-Ki Hong. 2012. Automated classifier generation for application-level mobile traffic identification. In *Network Operations and Management Symposium (NOMS), 2012 IEEE*. 1075–1081.

[17] Mauro Conti, Qian Qian Li, Alberto Maragno, and Riccardo Spolaor. 2018. The Dark Side (-Channel) of Mobile Devices: A Survey on Network Traffic Analysis. *IEEE Communications Surveys & Tutorials* (2018).

[18] Mauro Conti, Luigi V Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. 2015. Can't you hear me knocking: Identification of user actions on android apps via traffic analysis. In *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*. ACM, 297–304.

[19] RR Coveyou and Robert D MacPherson. 1967. Fourier analysis of uniform random number generators. *Journal of the ACM (JACM)* 14, 1 (1967), 100–119.

[20] Shuaifu Dai, Alok Tongaonkar, Xiaoyin Wang, Antonio Nucci, and Dong Song. 2013. Networkprofiler: Towards automatic fingerprinting of android apps. In *INFOCOM, 2013 Proceedings IEEE*. IEEE, 809–817.

[21] Sourav Kumar Dandapat, Swadhin Pradhan, Bivas Mitra, Romit Roy Choudhury, and Niloy Ganguly. 2015. Activpass: your daily activity is your password. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 2325–2334.

[22] Aveek K Das, Parth H Pathak, Chen-Nee Chuah, and Prasant Mohapatra. 2017. Privacy-aware contextual localization using network traffic analysis. *Computer Networks* 118 (2017), 24–36.

[23] Ali Doganaksoy, Baris Ege, Onur Koçak, and Fatih Sulak. 2010. Cryptographic Randomness Testing of Block Ciphers and Hash Functions. *IACR Cryptology ePrint Archive* 2010 (2010), 564.

[24] Hossein Falaki, Dimitrios Lymberopoulos, Ratul Mahajan, Srikanth Kandula, and Deborah Estrin. 2010. A first look at traffic on smartphones. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM, 281–287.

[25] Horst Feistel. 1973. Cryptography and Computer Privacy. *Scientific American* 228, 5 (1973), 15–23.

[26] Michael Finsterbusch, Chris Richter, Eduardo Rocha, Jean-Alexander Muller, and Klaus Hanssgen. 2014. A survey of payload-based traffic classification approaches. *Communications Surveys & Tutorials, IEEE* 16, 2 (2014), 1135–1156.

[27] Yanjie Fu, Hui Xiong, Xinjiang Lu, Jin Yang, and Can Chen. 2016. Service usage classification with encrypted internet traffic in mobile messaging apps. *IEEE Transactions on Mobile Computing* 15, 11 (2016), 2851–2864.

[28] Aaron Gember, Ashok Anand, and Aditya Akella. 2011. A comparative study of handheld and non-handheld traffic in campus Wi-Fi networks. In *Passive and Active Measurement*. Springer, 173–183.

[29] Xun Gong, Negar Kiyavash, and Nikita Borisov. 2010. Fingerprinting websites using remote traffic analysis. In *Proceedings of the 17th ACM conference on Computer and communications security*. ACM, 684–686.

[30] Alina Hang, Alexander De Luca, and Heinrich Hussmann. 2015. I know what you did last week! do you?: Dynamic security questions for fallback authentication on smartphones. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 1383–1392.

[31] Min Hur and Myung-Sup Kim. 2012. Towards smart phone traffic classification. In *Network Operations and Management Symposium (APNOMS), 2012 14th Asia-Pacific*. IEEE, 1–4.

[32] Alfonso Iacovazzi and Andrea Baiocchi. 2014. Internet traffic privacy enhancement with masking: Optimization and tradeoffs. *IEEE Transactions on Parallel and Distributed Systems* 25, 2 (2014), 353–362.

[33] Rozanna Nadeera Jesudasan, Philip Branch, and Jason But. 2010. Generic attributes for Skype identification using machine learning. *Centre for Advanced Internet Architectures, Swinburne University of Technology, Melbourne, Australia, Tech. Rep. A* 100820 (2010), 20.

[34] John B. Kam and George I. Davida. 1979. Structured Design of Substitution-Permutation Encryption Networks. *IEEE Trans. Comput.* 10 (1979), 747–753.

[35] Vasilios Katos. 2005. A randomness test for block ciphers. *Applied mathematics and computation* 162, 1 (2005), 29–35.

[36] Donald E Knuth. 1981. *The Art of Programming, vol. 2, Semi-Numerical Algorithms*. Addison Wesley, Reading, MA.

[37] Conti Mauro, Luigi Vincenzo Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. 2016. Analyzing android encrypted network traffic to identify user actions. *IEEE Transactions on Information Forensics and Security* 11, 1 (2016), 114–125.

[38] Stanislav Miskovic, Gene Moo Lee, Yong Liao, and Mario Baldi. 2015. AppPrint: Automatic Fingerprinting of Mobile Applications in Network Traffic. In *Proceedings of Passive and Active Measurement Conference*. 57–69.

[39] Thuy TT Nguyen and Grenville Armitage. 2008. A survey of techniques for internet traffic classification using machine learning. *Communications Surveys & Tutorials, IEEE* 10, 4 (2008), 56–76.

[40] Pavel Piskac and Jiri Novotny. 2011. Using of time characteristics in data flow for traffic classification. *Managing the Dynamics of Networks and Services* (2011), 173–176.

[41] Abbas Razaghpanah, Arian Akhavan Niaki, Narseo Vallina-Rodriguez, Srikanth Sundaresan, Johanna Amann, and Phillipa Gill. 2017. Studying TLS usage in Android apps. In *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies*. ACM, 350–362.

[42] Satadal Sengupta, Vinay Kumar Yadav, Yash Saraf, Harshit Gupta, Niloy Ganguly, Sandip Chakraborty, and Pradipta De. 2017. MoViDiff: Enabling service differentiation for mobile video apps. In *Integrated Network and Service Management (IM), 2017 IFIP/IEEE Symposium on*. IEEE, 537–543.

[43] Yan Shi and Subir Biswas. 2015. Characterization of Traffic Analysis based video stream source identification. In *Advanced Networks and Telecommuncations Systems (ANTS), 2015 IEEE International Conference on*. IEEE, 1–6.

[44] Juan Soto and Lawrence Bassham. 2000. *Randomness testing of the advanced encryption standard finalist candidates*. Technical Report. BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.

[45] Raphael Spreitzer, Veelasha Moonsamy, Thomas Korak, and Stefan Mangard. 2018. Systematic classification of side-channel attacks: a case study for mobile devices. (2018).

[46] Tim Stöber, Mario Frank, Jens Schmitt, and Ivan Martinovic. 2013. Who do you sync you are?: smartphone fingerprinting via application behaviour. In *Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks*. ACM, 7–12.

[47] Qixiang Sun, Daniel R Simon, Yi-Min Wang, Wilf Russell, Venkata N Padmanabhan, and Lili Qiu. 2002. Statistical identification of encrypted web browsing traffic. In *Security and Privacy, 2002. Proceedings. 2002 IEEE Symposium on*. IEEE, 19–30.

[48] Vincent F Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. 2018. Robust smartphone app identification via encrypted network traffic analysis. *IEEE Transactions on Information Forensics and Security* 13, 1 (2018), 63–78.

[49] Qinglong Wang, Amir Yahyavi, Bettina Kemme, and Wenbo He. 2015. I know what you did on your smartphone: Inferring app usage over encrypted data traffic. In *Communications and Network Security (CNS), 2015 IEEE Conference on*. IEEE, 433–441.

[50] Tzy-Shiah Wang, Hui-Tang Lin, Wei-Tsung Cheng, and Chang-Yu Chen. 2017. DBod: Clustering and detecting DGA-based botnets using DNS traffic analysis. *Computers & Security* 64 (2017), 1–15.

[51] Charles V Wright, Scott E Coull, and Fabian Monrose. 2009. Traffic Morphing: An Efficient Defense Against Statistical Traffic Analysis.. In *NDSS*, Vol. 9. Citeseer.

[52] Yue Wu, Joseph P Noonan, and Sos Agaian. 2011. NPCR and UACI randomness tests for image encryption. *Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT)* 1, 2 (2011), 31–38.

[53] Qiang Xu, Yong Liao, Stanislav Miskovic, Mario Baldi, Z. Morley Mao, Antonio Nucci, and Thomas Andrews. 2015. Automatic Generation of Mobile App Signatures from Traffic Observations. In *Proceedings of IEEE INFOCOM 2015 (INFOCOM '15)*.

[54] Hongyi Yao, Gyan Ranjan, Alok Tongaonkar, Yong Liao, and Zhuoqing Morley Mao. 2015. SAMPLES: Self Adaptive Mining of Persistent LExical Snippets for Classifying Mobile Application Traffic. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15)*. 439–451.

BaG: Behavior-aware Group Detection in Crowded Urban Spaces using WiFi Probes

Jiaxing Shen
The Hong Kong Polytechnic University
csjshen@comp.polyu.edu.hk

Jiannong Cao
The Hong Kong Polytechnic University
csjcao@comp.polyu.edu.hk

Xuefeng Liu
Huazhong University of Science and Technology
csxfliu@gmail.com

ABSTRACT

Group detection is gaining popularity as it enables various applications ranging from marketing to urban planning. The group information is an important social context which could facilitate a more comprehensive behavior analysis. An example is for retailers to determine the right incentive for potential customers. Existing methods use received signal strength indicator (RSSI) to detect co-located people as groups. However, this approach might have difficulties in crowded urban spaces since many strangers with similar mobility patterns could be identified as groups. Moreover, RSSI is vulnerable to many factors like the human body attenuation and thus is unreliable in crowded scenarios. In this work, we propose a behavior-aware group detection system (BaG). BaG fuses people's mobility information and smartphone usage behaviors. We observe that people in a group tend to have similar phone usage patterns. Those patterns could be effectively captured by the proposed feature: number of bursts (NoB). Unlike RSSI, NoB is more resilient to environmental changes as it only cares about receiving packets or not. Besides, both mobility and usage patterns correspond to the same underlying grouping information. The latent associations between them cannot be fully utilized in conventional detection methods like graph clustering. We propose a detection method based on collective matrix factorization to reveal the hidden associations by factorizing mobility information and usage patterns simultaneously. Experimental results indicate BaG outperforms baseline approaches by 3.97% ∼ 15.79% in F-score. The proposed system could also achieve robust and reliable performance in scenarios with different levels of crowdedness.

CCS CONCEPTS

• **Information systems → Mobile information processing systems**; **Clustering**.

KEYWORDS

group detection; WiFi; probe request; collective matrix factorization

ACM Reference Format:
Jiaxing Shen, Jiannong Cao, and Xuefeng Liu. 2019. BaG: Behavior-aware Group Detection in Crowded Urban Spaces using WiFi Probes. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313590

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313590

1 INTRODUCTION

Group detection plays an important role in many applications including marketing [29], healthcare [22, 35], and urban planning [3, 39]. The grouping information is an essential contextual feature in multi-target tracking [26] and behavior analysis [12]. This social context facilitates better interpretation of observed events and detection of abnormal behaviors [13]. A typical scenario is for retailers to determine the right incentive for potential customers. For example, "Buy X get Y free" promotions would be more attractive to group customers than individual customers [29]. It is commonly assumed a group could be discerned by people's physical locations and behaviors [6, 23, 29, 34]. Therefore, *group detection* is defined as a task of clustering a set of people into disjoint subsets based on their locations and behaviors.

Existing works exploit *probe requests*[1] to detect co-located people as a group [12, 19]. Co-location is achieved through received signal strength indicator (RSSI) contained in probe requests. Besides, those probes could be collected without difficulty since they are broadcast automatically to seek information about nearby access points (APs). Compared with other approaches, this method requires neither high deployment cost (e.g., deploy cameras [9, 34]) nor user intervention (e.g., carry wearable devices or install apps [18, 24, 29]). The probe-based methods are thus capable of detecting large-scale spontaneous groups in naturalistic environments.

However, current probe-based methods might have some difficulties in crowded environments. First, many strangers are close to each other in crowded spaces [29]. For example, two strangers might walk closely along an aisle. This indicates detecting co-located people as a group in crowded areas is error-prone. Second, RSSI is vulnerable to many factors like device diversity, multipath fading, and body attenuation [30]. It is unreliable to use RSSI to detect groups in dynamic environments crowded with people moving around.

We ask the following question: *can we reliably detect groups with WiFi probes in crowded urban spaces?* In this paper, we provide an affirmative answer by proposing a Behavior-aware Group detection (BaG) system that integrates both mobility information and phone usage behaviors. We have a key observation that people in the same group tend to have similar phone usage patterns. The observation is closely related to the concept of "phubbing" that is rife throughout the world. "Phubbing" is defined as the act of snubbing others in social interactions and instead focusing on one's smartphone [11].

[1] *Probe request*, *WiFi probe*, and *probe* are used interchangeably

According to a survey of 276 participants, the correlation of "phubbing" and "being phubbed" is 0.6 [4], which indicates individuals in a group would have similar phone usage patterns. The authors [4] explained this phenomenon occurs might due to false-consensus effects. Individuals assume that others think and do the same as themselves. Besides, we contribute a new feature (*number of bursts, NoB*) extracted from WiFi probes that could effectively capture phone usage patterns. A *burst* is a set of probes sent over a very short period of time (less than 1 second) [8]. The higher frequency a phone is used, the more NoB is generated. Unlike RSSI, NoB is resilient to crowded environments since it only cares about whether there are probes or not.

The vision of BaG, however, entails sparsity and fusing challenges when applied to real conditions.

1) How to handle spatiotemporal sparsity of WiFi data? First, an area is usually covered by a limited number of APs (spatial sparsity). Second, the probing frequency of a smartphone is affected by many factors like operating systems and users' usage patterns [8] (temporal sparsity). The data in common of two users could be much sparser. Measuring user similarity with such data could lead to highly biased and unreliable results.

2) How to cluster users fusing mobility and behaviors? The integration of mobility and behaviors remains an open issue. Mobility and behaviors have latent associations since both of them are different perspectives of the real grouping information. Simply combing them like calculating their arithmetic mean might limit the potential of hidden associations and thus derive unsatisfactory results.

For the sparsity challenge, we first represent mobility and behavior information into two matrices respectively. Then we apply matrix factorization (MF) to handle data sparsity by decomposing an input matrix into the product of several factor matrices. For the fusing challenge, we consider sparsity-constrained collective nonnegative MF (SCNMF). The advantages are two-fold. First, collective MF (CMF) takes the correlation of both input matrices into consideration by factorizing them simultaneously. Second, the sparsity constraint of CMF makes it an alternative for clustering so that grouping results could be derived directly without extra clustering processes.

According to our experimental evaluation with 82 volunteer groups in a large shopping mall, BaG achieves robust and reliable performance in scenarios with different levels of crowdedness. Compared with baseline approaches, BaG improves F-score of detection by 3.97% $\sim$ 15.79% in labeled data and 6.67% $\sim$ 20.69% in synthetically labeled data, respectively.

Our main contributions are summarized as follows.

- We introduce a new improvement of group detection in crowded environments: phone usage behaviors.
- A new feature (NoB) is extracted from WiFi probes that could effectively capture phone usage behaviors.
- We propose a new group detection method (SCNMF) that fuses mobility and behaviors and derives the grouping results without extra clustering processes.

The remainder of this paper is organized as follows. We present preliminaries in Section 2. Section 3 introduces design details of BaG, followed by experimental evaluation of the system in Section 4. Related work is introduced in Section 5. We conclude this paper and introduce the future work in the last section.

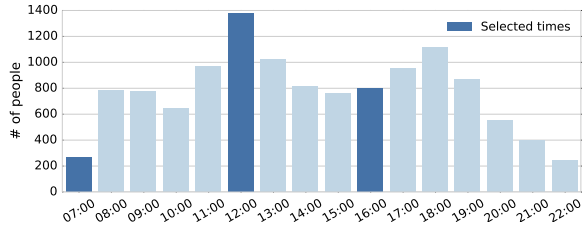


Figure 1: Average number of people in a day in the mall.

2 PRELIMINARIES

First of all, we use a two-day pilot study to illustrate relationships of the following variables in crowded environments: 1) RSS difference vs. groups 2) Phone usage behaviors vs. groups ; 3) Phone usage behaviors vs. NoB.

Experiments are conducted in a large shopping mall where we have deployed 20 APs to sniff WiFi probes. More detailed settings could be found in Section 4. Although it is difficult to define crowdedness directly, different numbers of onsite people could reflect different levels of crowdedness. Figure 1 highlights selected times of the experiment and the average number of people in the mall. The order of crowdedness is 12:00 > 16:00 > 7:00.

In each experiment, we recruit around 10 volunteer groups (each group consists of 2 $\sim$ 5 people) and record their grouping information and MAC addresses. We install an app on their smartphones to record screen on-off states as phone usage events. To ensure the authenticity, volunteers are only told to keep WiFi enabled and behave as normal without knowing the purposes of the experiment.

2.1 RSS Difference vs. Groups

RSSI methods calculate RSS (received signal strength) difference of two users. To this end, we first process users' WiFi data with an interval of 60 seconds, as probes are sent in a granularity of minutes [12]. Then user i's WiFi data in the t-th minute is represented as a vector:

$$\mathbf{v}_i^t = [rss_1, rss_2, \cdots, rss_N], \tag{1}$$

where rss_n represents received signal strength of the n-th AP, N is the number of total APs. If no RSS is available for AP n, then rss_n is set to 0. According to [37], RSS difference of users i and j at time t could be calculated with Eq. 2 and 3:

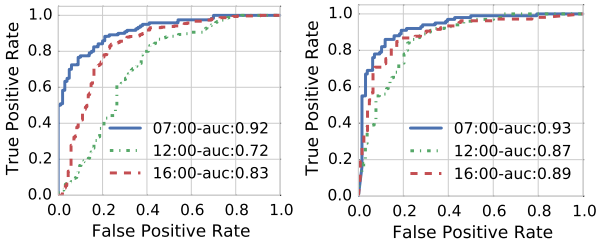
$$d(\mathbf{v}_i^t, \mathbf{v}_j^t) = \frac{1}{|A_i \cap A_j|} \cdot \sum_{a \in A_i \cap A_j} \left| rss_a^i - rss_a^j \right| \tag{2}$$

$$A_i = \left\{ k \middle| rss_k^i \neq 0, k \in [1, N] \right\} \tag{3}$$

The averaged RSS difference $D(i, j)$ over all common time slots is calculated with Eq. 4, where T is a set of common time slots of user i and j.

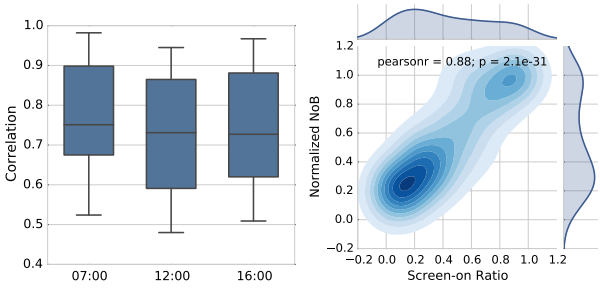
$$D(i, j) = \frac{1}{|T|} \sum_{t \in T} d(\mathbf{v}_i^t, \mathbf{v}_j^t), \tag{4}$$

Then we derive a distribution $\mathbf{X}_g(k)$ of $D(i, j)$ for all group pairs appear in time slot $k \in \{7{:}\,00, 12{:}\,00, 16{:}\,00\}$. Formal definition



(a) RSS difference (b) Behavior difference

Figure 2: ROC curves of different features at selected times. (a) With RSS difference; (b) With behavior difference.



(a) (b)

Figure 3: (a) Boxplot of the correlation for all users at selected times; (b) An example of correlation analysis of *screen-on ratio* and *normalized NoB* for a single user.

of $\mathbf{X}_g(k)$ is shown in Eq. 5, where boolean function $g(i,j)$ returns 'True' when user i and j are from the same group and U_k is a set of users appear in time slot k. Similarly, for all non-group pairs, we have another distribution $\mathbf{X}_{\tilde{g}}(k)$.

$$\begin{aligned}\mathbf{X}_g(k) &= \left\{D(i,j)\middle|g(i,j) = True; i,j \in U_k\right\}\\ \mathbf{X}_{\tilde{g}}(k) &= \left\{D(i,j)\middle|g(i,j) = False; i,j \in U_k\right\}\end{aligned} \tag{5}$$

Usually, $\mathbf{X}_g(k)$ and $\mathbf{X}_{\tilde{g}}(k)$ overlap to some extent, so we cannot easily find a cut-off point that completely separates both distributions. However, the smaller the overlap, the better the separation performance. To compare the effectiveness of using RSS difference more objectively, we exploit Receiver Operating Characteristic (ROC) curve, which is usually used to illustrate the diagnostic ability of a binary classifier system as its discrimination threshold varies. The curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

Figure 2a shows the comparison of ROC curves using RSS difference at selected times. It is clear that the effectiveness of different times varies significantly, indicating RSS difference is not a consistent feature in crowded environments. Besides, the order of effectiveness is 7:00 > 16:00 > 12:00, which is exactly the opposite order of the crowdedness. This implies crowded environment might have a significant negative impact on RSSI methods.

2.2 Smartphone Usage Behaviors vs. Groups

To represent phone usage behaviors, we use a straightforward way. If a smartphone is in use, the screen must be on. Otherwise, the screen is off. To represent user i's usage behaviors, we use a behavior vector $\mathbf{b}_i$:

$$\mathbf{b}_i = \left[\frac{s_1}{60}, \frac{s_2}{60}, \cdots, \frac{s_t}{60}, \cdots\right], \tag{6}$$

where s_t is a total number of seconds when the screen is on in the t-th minute. Behavior difference $D'(i,j)$ of user i and j is measured with Euclidean distance in Eq. 7.

$$D'(i,j) = \frac{1}{60}\sqrt{\sum_{t=1}(s_t^i - s_t^j)^2} \tag{7}$$

The behavior difference distributions $\mathbf{Y}_g(k)$ and $\mathbf{Y}_{\tilde{g}}(k)$ at time k are defined as:

$$\begin{aligned}\mathbf{Y}_g(k) &= \left\{D'(i,j)\middle|g(i,j) = True; i,j \in U_k\right\}\\ \mathbf{Y}_{\tilde{g}}(k) &= \left\{D'(i,j)\middle|g(i,j) = False; i,j \in U_k\right\}\end{aligned} \tag{8}$$

Similarly, we draw ROC curves for $\mathbf{Y}_g(k)$ and $\mathbf{Y}_{\tilde{g}}(k)$. As shown in Figure 2b, the ROC curves at different times are close to each other, indicating behavior difference is stable in different crowdedness settings. Besides, the performance (AUC, area under curve, are all over 0.85) proves behavior difference is effective for group detection.

2.3 Smartphone Usage Behaviors vs. NoB

Although different smartphones have different probing patterns, it is reported that a smartphone will send more probes when the screen is on [8]. Therefore, the more frequently a smartphone is used, the more NoB is generated. A recent research work [15] uses the number of wireless packets to predict the screen on-off states and achieves an accuracy ranging from 93% to 100%. NoB is different from the work in two aspects. First, only WiFi probes are used here. Second, we use the number of bursts instead of the number of packets.

We represent user i's NoB feature with a NoB vector:

$$\mathbf{n}_i = [n_1, n_2, \cdots, n_t, \cdots], \tag{9}$$

where n_t is the number of bursts in the t-th minute. To handle the impact of device diversity, we further calculate a normalized NoB vector with Eq. 10, where $max(.)$ find the largest element of an input vector.

$$\hat{\mathbf{n}}_i = \mathbf{n}_i / max(\mathbf{n}_i) \tag{10}$$

To analyze the correlation between smartphone usage behaviors and NoB, we calculate the Pearson Correlation Coefficient of the normalized NoB vector ($\hat{\mathbf{n}}_i$) and the behavior vector ($\mathbf{b}_i$) for users in all experiments. Figure 3a shows the boxplot of the correlation from all users. It is clear that smartphone usage behaviors and NoB are closely correlated in different crowdedness settings. Furthermore, an example of correlation analysis for a single user is illustrated in Figure 3b. The correlation coefficient and the p-value are 0.88 and $2.1e^{-31}$, respectively, which shows it is significant that screen-on ratio is positively correlated with NoB for the user. Furthermore, we could find that the correlation is higher when both variables are small. This means if the screen is on, it is more likely that the smartphone will send probes.

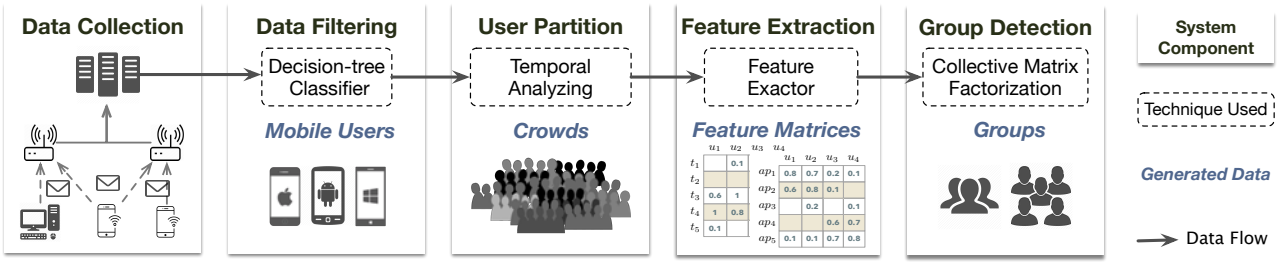


Figure 4: System overview of BaG.

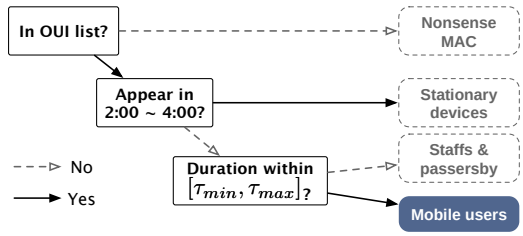


Figure 5: Filtering out WiFi data from non-mobile users.

3 SYSTEM DESIGN

In this section, we elaborate on design details of BaG. The overview of the proposed system is illustrated in Figure 4. First, we passively sniff WiFi probes from different devices (Data Collection). Then we filter out data from non-mobile users (Data filtering) and separate the filtered data into several crowds (User Partition). Besides, mobility and behavior related features are extracted and represented in matrices (Feature Extraction). Last, we use collective matrix factorization to fuse two matrices and derive grouping results (Group Detection) without extra clustering processes.

3.1 Data Collection

We exploit off-the-shelf WiFi APs to sniff WiFi probes from nearby devices and upload the data to a server. The AP configurations are as follows: AR9341 (WLAN chip), 64M (RAM), and 8M (Flash Memory). We use those APs for collecting probe requests only since they have limited computational and storage capacity.

Each AP works under OpenWrt[2] with a virtual network interface enabled in monitor mode. We use Tcpdump (a utility for capturing network traffic) to sniff WiFi probes and upload the collected data to the server at 2:00 AM when the bandwidth is fully available. Once the upload process is complete, local WiFi data will be deleted to make space for the new coming data.

3.2 Data Filtering

Data filtering services as a preprocessing of the raw WiFi data including three steps: combat with MAC randomization, filter out data from non-mobile users, and extract useful data fields.

To prevent third parties from tracking devices with MAC (Media Access Control) address, several vendors have implemented MAC address randomization which implies that probe requests no longer use the real MAC address of the device. However, this mechanism may not work as expected in reality since it is reported defeated in several recent works [8, 21, 38]. Besides, this randomization mechanism is not activated in the majority of old devices, especially for Android systems. We follow the practice described in [21] to defeat MAC randomization.

As the collected data may come from stationary devices (like desktops and IP cameras) and mobile devices, we remove the data of non-mobile users devices with a simple decision-tree classifier as illustrated in Figure 5. If a MAC is not in OUI (Organizationally Unique Identifier) list, it might be a forged MAC or from unregistered companies which makes no sense to us. Then we use the routine of human activities to filter out stationary devices as during 2:00 ~ 4:00, most public places are closed, and most of the people are in sleep. Last, with the help of two scenario-dependent parameters τ_{min} and τ_{max} which represent the minimum and maximum dwell time, we could filter out data from staffs and passersby. In our system, we empirically set $\tau_{min} = 10$ and $\tau_{max} = 240$ which means people stay for at least 10 minutes and up to 4 hours. This setting is based on our observation and is intended to identify customer groups. If we want to detect staff groups, both parameters should be relatively larger. More details could be found in Reference [32].

Lastly, we extract desired fields from WiFi probes. For each WiFi probe, we can get the following data entry:

$$< \text{Timestamp, Device_MAC, AP_MAC} >,$$

where *Timestamp* indicates the time of receiving a probe, *MAC* is MAC address which uniquely represents a smartphone or an AP.

3.3 User Partition

Instead of detecting groups out of all mobile users in a whole day, we first utilize temporal constraint of groups to separate users into different *crowds* and then identify groups out of each crowd. The main advantage of user partition is that the efficiency of group detection can be greatly improved since it breaks down the original problem into many subproblems which can be solved independently.

The idea of the temporal constraint is very simple. The time gap of the appearance of groups members would never be too large. An illustration is shown in Figure 6. For an extreme example, group members would not appear separately in the morning and in the night. Formally, the temporal constraint is defined as

$$max(|ta_i - ta_j|) \leq \psi, \tag{11}$$

[2]OpenWrt is a highly extensible GNU/Linux distribution for embedded devices (typically wireless routers).

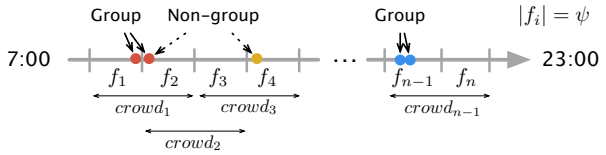


Figure 6: Partitioning users of a day into crowds.

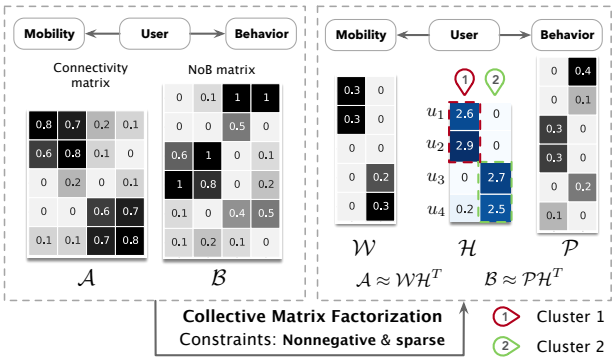


Figure 7: Illustration of using CMF to cluster users.

where ta_i means the appearance time of user i and ψ is a threshold. The physical meaning of parameter ψ is the dwell time of the most people. The selection of ψ is discussed in Section 4.

Then we equally partition the daytime into non-overlapping fragments using ψ. The partition process is depicted in Figure 6. For any two sequential fragments f_i and f_{i+1}, we take out WiFi data entries of users whose start time falls in both fragments. Those users are regarded as a *crowd* and their WiFi data entries are denoted as $crowd_i$. If a person appears in f_i, his/her group mate may appear in the same fragment or the next fragment, but it is barely possible to appear in f_{i+2} since ψ restricts the dwell time of most people.

3.4 Feature Extraction

Given data entries of a crowd, we represent two features as two feature matrices. The first feature is NoB that indicates a user's phone usage behaviors. The other is *AP connectivity* which indicates a user's mobility information.

As introduced in Section 1, NoB refers to how many times a device has sent WiFi probes. AP connectivity means how long a user has been in the coverage area of an AP. It reflects where a user has been to and how long they have stayed. Compared to RSSI, although AP connectivity is more coarse-grained, it is also more robust and reliable. Both AP connectivity and NoB are insensitive to crowded environments as they only care about whether a smartphone sends WiFi probes or not which cannot be readily affected.

We first group WiFi data entries by users' MAC addresses. For each user, we raise the granularity of the data from seconds to minutes, as this practice reduces the impact of sparse WiFi probes. To eliminate the negative impact of device diversity, we normalize connectivity vector and NoB vector for each user. Lastly, we combine all users' connectivity and NoB vectors and generate corresponding matrices.

3.5 Group Detection

Group detection is essentially a *hard clustering* problem which means each user can only belong to a cluster or not. Previous works [12, 29] need to measure pairwise user similarity and then construct a user graph with nodes representing users and the edge between nodes indicating their similarity. After that, an explicit graph clustering process like Markov Cluster Algorithm is applied to the graph to detect groups.

However, it would be inefficient and cumbersome to construct the graph by measuring pairwise user similarity. Instead, we use nonnegative matrix factorization (NMF) to derive the group results directly without extra clustering processes. We choose NMF rather than other matrix factorization like singular value decomposition (SVD) as entries in both original matrices have physical meaning (number of bursts and duration time). Positive factors facilitate direct physical connections. Besides, collective nonnegative matrix factorization (CNMF) is used to jointly factorize connectivity and NoB matrices to get better results as both matrices share the same latent feature: the grouping information. An illustration of the whole process is shown in Figure 7. We have two relation matrices $\mathcal{A}$ (user-mobility matrix) and $\mathcal{B}$ (user-behavior matrix) as input. After collective matrix factorization, there are three factors corresponding to three entities. The user entity matrix $\mathcal{H}$ contains the group results. Each row of $\mathcal{H}$ indicates a potential group with $\mathcal{H}_{ij}$ representing the strength that user i belongs to group j. From $\mathcal{H}$ we can see clearly that the group results are $(u_1, u2)$ and (u_3, u_4).

In the following subsections, we first briefly review the rationale of using NMF for clustering. Then we focus on solving the group detection problem using CNMF.

3.5.1 ***NMF and k-means.*** Given an input matrix $\mathcal{A} \in \mathbb{R}^{m \times n}$ and an integer $k < \min(m, n)$, NMF aims to find two nonnegative factors $\mathcal{W} \in \mathbb{R}^{m \times k}$, $\mathcal{H} \in \mathbb{R}^{n \times k}$ such that $\mathcal{A} \approx \mathcal{W}\mathcal{H}^T$. $\mathcal{W}$ and $\mathcal{H}$ can be found by solving the optimization problem:

$$\min_{\mathcal{W},\mathcal{H}} F_k(\mathcal{W},\mathcal{H}) = \frac{1}{2}\left\|\mathcal{A} - \mathcal{W}\mathcal{H}^T\right\|_F^2 s.t. \quad \mathcal{W},\mathcal{H} \geq 0, \tag{12}$$

where $F_k(\cdot)$ is the loss function and k is the reduced dimension. $\|\mathcal{A}\|_F$ means Frobenius Norm of matrix $\mathcal{A}$. The reasons of using Frobenius Norm are that it has a Guassian noise interpretation and the objective function can be easily transformed to a matrix trace version.

Existing works [17] provide evidence that sparsity constrained NMF is a viable alternative as a clustering method. The objective of k-means is to minimize the sum of squared distances from each data point to its centroid. With $\mathcal{A} = [a_1, ..., a_n] \in \mathbb{R}^{m \times n}$, the objective function J_k with given integer k can be written as:

$$J_k = \sum_{j=1}^{k} \sum_{a_i \in C_j} \left\|a_i - c_j\right\|^2 = \left\|\mathcal{A} - C\mathcal{B}^T\right\|_F^2, \tag{13}$$

where $\mathcal{B} \in \mathbb{R}^{n \times k}$ denotes the clustering assignment. If the i-th observation is assigned to the j-th cluster $\mathcal{B}_{ij} = 1$, otherwise $\mathcal{B}_{ij} = 0$. $C = [c_1, ..., c_k] \in \mathbb{R}^{n \times k}$ is the centroid matrix where c_j is the cluster centroid of cluster C_j. C can also be written as $C = \mathcal{A}\mathcal{B}\mathcal{D}^{-1}$

Algorithm 1 Alternating nonnegative least squares

Require: Initialize $\mathcal{H}$ and $\mathcal{P}$ with random nonnegative values.

1: **while** the convergence criterion is not satisfied **do**

2: $\mathcal{W} \leftarrow \arg\min_{\mathcal{W}} \left\| \begin{pmatrix} \sqrt{\alpha}\mathcal{H} \\ \sqrt{\eta}\mathbf{I}_k \end{pmatrix} \mathcal{W}^T - \begin{pmatrix} \sqrt{\alpha}\mathcal{A}^T \\ \mathbf{0}_{k\times m} \end{pmatrix} \right\|_F^2$

3: $\mathcal{P} \leftarrow \arg\min_{\mathcal{P}} \left\| \begin{pmatrix} \sqrt{1-\alpha}\mathcal{H} \\ \sqrt{\eta}\mathbf{I}_k \end{pmatrix} \mathcal{P}^T - \begin{pmatrix} \sqrt{1-\alpha}\mathcal{A}^T \\ \mathbf{0}_{k\times l} \end{pmatrix} \right\|_F^2$

4: $\mathcal{H} \leftarrow \arg\min_{\mathcal{H}} \left\| \begin{pmatrix} \sqrt{\alpha}\mathcal{W} \\ \sqrt{1-\alpha}\mathcal{P} \\ \sqrt{\beta}\mathbf{e}_{1\times k} \end{pmatrix} \mathcal{H}^T - \begin{pmatrix} \sqrt{\alpha}\mathcal{A} \\ \sqrt{1-\alpha}\mathcal{B} \\ \mathbf{0}_{1\times n} \end{pmatrix} \right\|_F^2$

5: **end while**

where $\mathcal{D}^{-1}$ is a diagonal matrix defined in:

$$\mathcal{D}^{-1} = diag(\frac{1}{|C_1|}, \frac{1}{|C_2|}, ..., \frac{1}{|C_k|}) \in \mathbb{R}^{k\times k} \tag{14}$$

$|C_j|$ denotes the number of data points in cluster j. Since a diagonal matrix can be factored as any two diagonal matrices $\mathcal{D}^{-1} = \mathcal{D}_1\mathcal{D}_2$. Defining $\mathcal{F} = \mathcal{B}\mathcal{D}_1$ and $\mathcal{H} = \mathcal{B}\mathcal{D}_2$, the objective function J_k can be rephrased as:

$$J_k = \left\|\mathcal{A} - \mathcal{A}\mathcal{B}\mathcal{D}_1\mathcal{D}_2\mathcal{B}^T\right\|_F^2 = \left\|\mathcal{A} - \mathcal{A}\mathcal{F}\mathcal{H}^T\right\|_F^2 \tag{15}$$

where $\mathcal{F}$ and $\mathcal{H}$ have exactly one positive element in each row. Factor $\mathcal{H}$ has exactly one nonzero element for each row which represents a hard clustering result of the corresponding data point. If we let $\mathcal{W} = \mathcal{A}\mathcal{F}$, J_k is similar to objective function F_k of NMF. The difference is that NMF formulation does not have the constraints that $\mathcal{H}^T$ is a sparse matrix. To this end, sparse NMF (SNMF) is proposed. The idea is to use ℓ_1-norm regularization to achieve sparsity of the factorization [36].

3.5.2 ***SCNMF for Group Detection***. Now we provide our formulation and solution to the group detection problem. Given two input matrices: connectivity matrix $\mathcal{A} \in \mathbb{R}^{m\times n}$ and NoB matrix $\mathcal{B} \in \mathbb{R}^{l\times n}$, and an integer $k < \min\{m, n, l\}$, sparsity-constrained collective nonnegative matrix factorization (SCNMF) aims to find three nonnegative factors $\mathcal{W} \in \mathbb{R}^{m\times k}$, $\mathcal{H} \in \mathbb{R}^{n\times k}$ (sparse matrix), and $\mathcal{P} \in \mathbb{R}^{l\times k}$ such that:

$$\begin{cases} \mathcal{A} \approx \mathcal{W}\mathcal{H}^T \\ \mathcal{B} \approx \mathcal{P}\mathcal{H}^T \end{cases} \tag{16}$$

The solutions $\mathcal{W}, \mathcal{H}, \mathcal{P}$ can be found by solving the optimization problem:

$$\min_{\mathcal{W},\mathcal{H},\mathcal{P}} \frac{1}{2}\Big[\alpha\left\|\mathcal{A} - \mathcal{W}\mathcal{H}^T\right\|_F^2 + (1-\alpha)\left\|\mathcal{B} - \mathcal{P}\mathcal{H}^T\right\|_F^2 + \beta\sum_{j=1}^{n}\left\|\mathcal{H}_j\right\|_1^2 + \eta(\|\mathcal{W}\|_F^2 + \|\mathcal{P}\|_F^2)\Big] \text{ s.t. } \mathcal{W},\mathcal{H},\mathcal{P} \geq 0, \tag{17}$$

where $\mathcal{H}_j$ is the j-th row vector of $\mathcal{H}$. Parameter $\alpha \in [0, 1]$ is scenario-dependent which weights the relative importance of two input matrices. Parameter $\beta > 0$ balances the trade-off between the accuracy of approximation and the sparseness of $\mathcal{H}$. Parameter $\eta > 0$ controls the size of the elements of $\mathcal{W}$ and $\mathcal{P}$ respectively. It is usually determined by the largest element of input matrices [17].

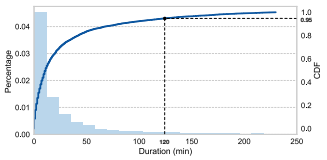


Figure 8: Distribution and CDF of dwell time in the mall.

Although solving Eq. 17 is a non-convex problem [10], it is convex separately in each factor, i.e., finding the optimal factor $\mathcal{W}$ corresponding to fixed factors $\mathcal{H}$ and $\mathcal{P}$ reduce to a convex optimization problem. Algorithms based alternating nonnegative least squares (ANLS) are often used for sparse NMF. Then we use Algorithm 1 to iterate the ANLS until a convergence criterion, which is set to iterate for 200 times [16], is satisfied. In the algorithm, $\mathbf{I}_k$ is an identity matrix of size $k \times k$, $\mathbf{0}_{k\times m}$ is a zero matrix of size $k \times m$, and $\mathbf{e}_{1\times k}$ is a row vector with all components equal to one. More details can be found in [16].

3.5.3 ***Determine the correct number of*** k. Parameter k is an input of the system which corresponds to the potential number of groups. In [17], consistency of clustering algorithms is used to determine the correct number of k from data. Consistency means the consistency clustering results under random initializations. To measure consistency, we construct a matrix $C_k \in \mathbb{R}^{n\times n}$ where n is the number of all users and k is the number of clusters. For pairwise users i and j, $C_k(i, j) = 1$ if they are assigned to the same cluster, otherwise $C_k(i, j) = 0$. Then we calculate the averaged $\hat{C}_k$ over trails. $\hat{C}_k(i, j)$ represents the possibility user i and user j being assigned to the same cluster. If the clustering results were consistent throughout all trails, each element in $\hat{C}_k$ would be close to either 0 or 1. A general quality of consistency is proposed by dispersion coefficient:

$$\rho_k = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} 4\Big[\hat{C}_k(i, j) - \frac{1}{2}\Big]^2 \tag{18}$$

where $\rho_k \in [0, 1]$ and $\rho_k = 1$ indicates the perfect consistent clustering results. After obtaining the values of ρ_k for various k, the number of clusters could be determined by the point k' where $\rho_{k'}$ drops. More detailed information can be found in [17].

4 EXPERIMENTAL EVALUATION

4.1 Settings

4.1.1 ***Setup***. We conduct experiments in a large shopping mall with 4 floors covering an area of $4890m^2$. The experiments here and the experiments conducted in Section 2 are different in terms of purposes and scales.

There are originally 20 APs installed in the mall for customers to access the Internet. We use those APs to collect the WiFi data. Within one week, we recruit 82 volunteer groups including 294 volunteers during 28 experiments at different times of a day. For each experiment, we recruit $2 \sim 4$ volunteer groups with each group containing $2 \sim 5$ people and record their MAC addresses and grouping information. The majority of experiments last from 0.5 hour to 1.5 hours. To ensure authenticity, volunteers are only

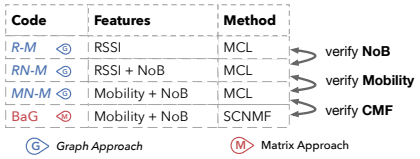


Figure 9: Illustration of baseline approaches.

told to keep their WiFi enabled without knowing the purpose of experiments.

4.1.2 ***Collected data.*** During the week, 4, 184, 778 WiFi probes are received from 59, 282 devices a day on average. According to our data filtering (Section 3.2), only 6.66% (3, 951) of them are from customers' mobile devices.

Figure 8 depicts the distribution of customers' dwell time which obeys the power law distribution. We also draw cumulative distribution function (CDF) of dwell time in Figure 8. It shows that 95% customers stay in the mall for less than 2 hours. This finding is useful because it helps to determine the value of parameter ψ (in Section 3.3).

4.1.3 ***Ground truth.*** BaG takes the collected data of each day as input and output the detected groups. We have no ways of knowing the grouping information for everyone in the collected data. However, we could know the relationships of a subset of those people.

Based on the information of our volunteers, we have prepared two evaluation datasets. The difference between two datasets is the number of non-group pairs. We call the first dataset as *labeled dataset* as it is entirely contributed by volunteers. In other words, we know the relation (group and non-group) of every pair of users. This dataset consists of 438 group pairs (positive samples) and 1, 135 non-group pairs (negative samples).

The second dataset is *synthetically labeled dataset* which includes some non-volunteers. When recruiting volunteer groups, we engage all group members in the experiment. Therefore, user pairs consisting of volunteer and non-volunteers must be non-group. Eventually, we get a dataset of 438 positive samples, and 2, 311 negative samples.

Even though we could find significantly more negative samples, we carefully control such negative samples. One reason is that such synthetic negative sample should also obey the temporal constraint as mentioned in Section 3.3, otherwise it could be too easy to detect. Another reason is that the difference of numbers between positive and negative samples should not be too large, otherwise the evaluation metrics might be affected.

4.1.4 ***Metrics.*** There is no consensus on which metrics should be used to evaluate groups detection [34]. Here we use *Precision*, *Recall* and *F-score* to measure the performance of group detection, which are defined as follows.

$$\begin{cases} \text{Precision(P)} = \dfrac{tp}{tp + fp} \\ \text{Recall(R)} = \dfrac{tp}{tp + fn} \\ \text{F-score} = 2 \cdot \dfrac{\text{P} \cdot \text{R}}{\text{P} + \text{R}} \end{cases}$$

Truth / Detection	g	$\tilde{g}$
g	tp	fp
$\tilde{g}$	fn	tn

g : Group
$\tilde{g}$: Non-group

Table 1: Detailed results of different α when $\beta = 0.2$.

α	precision	recall	F-score
.1	.861	.887	.874
.3	.897	.898	.897
.5	.822	.914	.866
.7	.743	.884	.807
.9	.690	.847	.760

Table 2: Detailed results of different β when $\alpha = 0.5$.

β	precision	recall	F-score
.1	.815	.912	.861
.2	.822	.918	.867
.3	.829	.915	.870
.4	.813	.905	.857
.5	.817	.896	.855

4.1.5 ***Baseline approaches.*** As mentioned in Introduction, two of our contributions are an effective feature (NoB) and a clustering method (CMF) for detecting groups. To verify the effectiveness of both NoB and CMF, we set three baselines R-M, RN-M, and MN-M. A baseline has two important components: feature(s) and method. Figure 9 shows detailed configurations of all approaches. According to the clustering methods used, those approaches can be further classified as graph approach (using Markov Cluster algorithm, MCL) and matrix approach (using SCNMF). Steps of graph approaches: 1) Measure user similarity based on some features. 2) Construct a user graph with users as nodes and their similarities as edge weights. 3) Run graph clustering algorithms to derive clusters or groups.

For different graph approaches, different feature combinations are used to measure user similarity. For multiple features, we combine them with arithmetic mean to measure similarity. Finally, MCL is applied to the constructed graph to detect groups. MCL is a popular graph clustering algorithm used by many existing works [12, 29]. It works well when the cluster size is small and it does not require the number of clusters as an input.

For matrix approach, BaG is a typical one. Therefore, it is unnecessary to repeat the steps again here. The precondition of this approach is to have matrices as input. That is why we do not have a matrix approach with RSSI and NoB since it is non-trivial to find an appropriate way of representing RSSI in the matrix format.

Table 3: Performance in labeled dataset.

Codes	7:00			16:00			12:00		
	P[1]	R[2]	F[3]	P	R	F	P	R	F
R-M	.874	.886	.880	.755	.788	.771	.700	.720	.710
RN-M	.920	.935	.927	.848	.876	.862	.815	.799	.807
MN-M	.905	.922	.914	.856	.886	.871	.817	.849	.832
BaG	.934	.954	.944	.887	.915	.901	.859	.892	.875

[1] Precision [2] Recall [3] F-score

Table 4: Performance in synthetically labeled dataset.

Codes	7:00			16:00			12:00		
	P	R	F	P	R	F	P	R	F
R-M	.751	.806	.778	.652	.715	.682	.618	.655	.636
RN-M	.821	.886	.852	.757	.832	.793	.729	.756	.742
MN-M	.806	.850	.827	.758	.810	.783	.727	.779	.752
BaG	.849	.905	.876	.804	.868	.835	.777	.842	.808

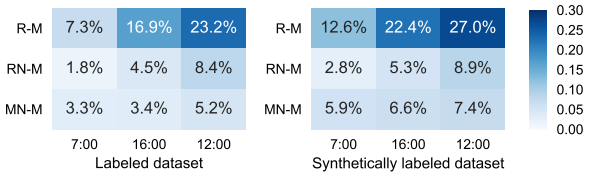


Figure 10: Performance gain ratios of BaG.

4.1.6 ***Parameter selection.*** In this system, there are 3 parameters to determine: ψ for user partition (Section 3.3) and α, β for group detection (Section 3.5).

In user partition, ψ represents the maximum duration time of onsite customers. As discussed in Section 3.3, the physical meaning of ψ is the dwell time of customers. While Figure 8 shows that 95% customers stay in the mall for less than 2 hours. Therefore, we set ψ to 120 minutes in our system.

For group detection, parameter α controls the relative importance of connectivity matrix over NoB matrix. According to the results shown in Table 1, we set α to 0.3. Generally, small α is preferred since connectivity matrix contains coarse-grained mobility information that has limited capacity to detect groups in crowded environments. Parameter β balances the trade-off between accuracy of approximation and sparseness. The performance of different β is illustrated in Table 2. Even though the performance is not that sensitive to β, too big β is undesirable since that might lead to worse approximation [17], so β is set to 0.2.

4.2 Evaluation

As explained in Section 2.1, we use the number of onsite people to represent different crowdedness settings. To have a comprehensive understanding of the performance of BaG, the evaluation is further explained with two datasets, four approaches, three feature combinations, and two clustering methods.

4.2.1 ***Two datasets.*** According to Figure 1, the order crowdedness for selected hours is 12:00 > 16:00 > 7:00. Detailed performance of all approaches on labeled and synthetically labeled datasets are shown in Table 3 and 4, respectively.

Compared with labeled dataset, all approaches have worse performance on the synthetically labeled dataset. This is reasonable since changes of the number of people to be detected might alter the clustering results. Besides, the probability that strangers are close to each other or have similar smartphones usage patterns are likely to increases with the number of people.

Interestingly, the average change of precision (−11.1%) is more significant than that of recall (−0.7%) over all approaches and times. One possible reason is the boost of negative samples increases false detections especially false positive detections (non-groups are detected as groups) and leads to a significant decline in precision.

4.2.2 ***Four approaches.*** Averaging the performance over all times, BaG improves F-score by 3.97% ∼ 15.79% and 6.67% ∼ 20.69% in labeled and synthetically labeled datasets, respectively.

We further demonstrate the performance of BaG by showing the *F-score gain ratio* over baselines. The ratio is defined in Eq. 19 where bl represents a baseline approach.

$$F\text{-score gain ratio} = \frac{F_{\text{BaG}} - F_{\text{bl}}}{F_{\text{bl}}}, \tag{19}$$

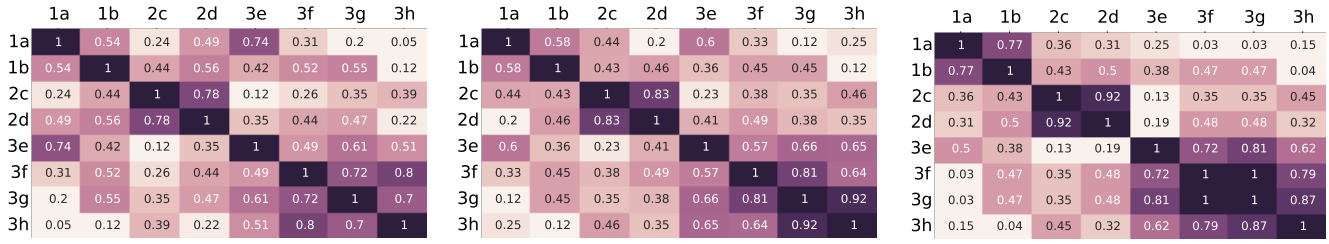
As shown in Figure 10, the ratio increases with the crowdedness indicating BaG is more accurate and reliable in crowded environments. Besides, the improvement is slightly higher in synthetically labeled data.

4.2.3 ***Three feature combinations.*** R-M, RN-M, and MN-M are all based on the same clustering method, so we could match them to compare the performance of different feature combinations.

From the performance tables, we have identified two issues on both datasets. First, the performance of feature combinations with NoB (RN-M and MN-M) are consistently higher than the other approach (R-M) over all times. The reasons might be two-fold. On one hand, the additional feature NoB has been used. On the other, it also proves NoB has the capability to capture the underlying grouping information in different crowdedness settings.

The second issue is that in the least crowded setting RN-M slightly outperforms MN-M, but the situation is just the opposite in the most crowded setting. This phenomenon results from different recall declining rates. When the crowdedness increases, both methods become worse, but recall of RN-M declines faster than that of MN-M. Potential reasons for different declining rates might be as follows. RSSI is more fine-grained than mobility but it is also more unreliable when the environment gets crowded since people frequently move around and hold their smartphones in different ways. In crowded environments, group users might have low similarity based on RSSI distance measurement and thus generate more false negative detections (groups are detected as non-groups) that degrade recall.

To support this, in Figure 11 we show an example of similarity matrices from a single experiment in 12:00 with 3 groups. As we can see in Figure 11a, some group users (like e and f) have low similarity



(a) RSSI **(b) RSSI+NoB** **(c) Mobility+NoB**

Figure 11: User similarity matrix from a single experiment in 12:00 using different feature combinations. Alphabets represent people, numbers reveal their grouping information. For example, $1a$ means user a is in group 1.

using RSSI. Although NoB has a certain effect, the improvement is quite limited as illustrated in Figure 11b. As a contrast in Figure 11c, the combination of mobility and NoB measures user similarity in a better way as mobility is more coarse-grained and thus insensitive to environmental changes.

4.2.4 ***Two clustering methods.*** Two clustering methods refer to graph approach and matrix approach. We could compare them by matching the performance of BaG and MN-M from performance tables and Figure 10. Overall, BaG slightly outperforms MN-M in all situations ranging from 3.3% $\sim$ 7.4%.

The main advantage of the matrix approach relies on the fusion of mobility and NoB. Mobility and NoB are two different perspectives of the real grouping information and thus have some hidden associations. With collective matrix factorization, we could simultaneously factorize them to reveal their hidden associations. While in the graph approach mobility and NoB are simply combined using arithmetic mean which might have limited the potential of the latent association.

5 RELATED WORK

Group detection can be classified as vision-based approaches, sensor-based approaches, and WiFi-based approaches according to different means.

Vision-based approaches regard group detection as a task of clustering set of users' trajectories into disjoint subsets [9, 34]. However, this kind of methods have some apparent limitations. First of all, the biggest issue is privacy erosion. Besides, video surveillance suffers from environmental issues such as non-line-of-sight, and low brightness.

Sensor-based approaches use wearable devices or install apps on smartphones to collect users' behavioral data. Groups are detected through correlation analysis of multiple sensor data. For instance, MIT researchers use specially designed wearable devices called "Sociometric Badges" [24, 25, 33] to measure group behavior through face-to-face interaction and physical proximity. Some research works [18, 20, 29] combine several sensor modalities (WiFi, accelerometer, compass, etc) to measures users similarity. However, these methods might be difficult to collect data on a scale, as they require user intervention which would be cumbersome in some scenarios. Besides, engaging multiple sensors drains smartphone battery more quickly.

WiFi-based approaches utilize the information contained in probe request to detect groups. Compared to other approaches, WiFi-based approaches do not require high deployment cost or user intervention. The probe contains significant information like time stamp, smartphone MAC address, RSSI, and SSID, which enables a wide range of applications like passive tracking [7], crowd counting[28, 40], and facility utilization analysis [27]. SSID and RSSI are two frequently used information to detect groups. Cunche et. al. [1, 2, 5] link different smartphones through SSID similarity. However, 80% of the devices reply with empty SSID list [14], approaches that rely on SSID may not work well anymore. Then researchers' focus transfer to RSSI which indicate users' mobility. Kjærgaard et. al. [19] extract spatial features, signal-strength features, and pseudo-spatial features from signal strength to detect social groups which they call pedestrian flocks. It is found that the performance of spatial features is unreliable, since mapping RSSI into locations is not accurate enough. Besides, the mapping process itself is usually time-consuming and labor-intensive. To avoid the cumbersome mapping process, directly measure the similarity of RSSI fingerprints to detect co-located mobile users. These methods get rid of absolute locations, thus eliminate labor-intensive calibration and protect users' privacy. SocialProbe [12] considers the hardware diversity and uses the normalized RSSI vector to achieve co-location detection. However, hardware difference is not the only factor, other factors like body attenuation, environmental disturbance, and multipath fading can also significantly affect RSSI. Therefore, RSSI based approach may not achieve reliable performance in some scenarios, especially in crowded urban spaces like shopping malls. SNOW [31] focuses on a certain type of group in shopping malls using WiFi data from associated devices rather than WiFi probes.

6 CONCLUSION

In this paper, we present a group detection system (BaG) that is verified reliable and accurate in different crowdedness settings. One of our main contributions is a new feature (NoB) extracted from WiFi probes. NoB could effectively capture phone usage patterns which is a new perspective for detecting groups. The second contribution is a new detection method (SCNMF) that fuses mobility information and usage behaviors. SCNMF could reveal the hidden associations between mobility and behaviors by decomposing them simultaneously. The experimental evaluation in a large shopping

mall demonstrates the effectiveness and robustness of BaG in scenarios with different levels of crowdedness. Compare to baseline approaches, BaG shows a significant improvement by increasing F-score of detection by 3.97% $\sim$ 15.79% in labeled dataset and 6.67% $\sim$ 20.69% in synthetically labeled dataset respectively.

ACKNOWLEDGMENTS

We sincerely appreciate the efforts of the anonymous reviewers and their insightful comments for improving this manuscript. The presented work was supported by National Key R&D Program of China (2018 YFB1004801). It was also partially supported by Shenzhen Basic Research Funding Scheme (JCYJ20170818104222072), and NSFC with project No. 61332004 and 61572218.

REFERENCES

[1] Marco V Barbera, Alessandro Epasto, Alessandro Mei, Vasile C Perta, and Julinda Stefa. [n. d.]. Signals from the crowd: uncovering social relationships through smartphone probes. In *Proceedings of ACM IMC, 2013.*

[2] Ningning Cheng, Prasant Mohapatra, Mathieu Cunche, Mohamed Ali Kaafar, Roksana Boreli, and Srikanth Krishnamurthy. [n. d.]. Inferring user relationship from hidden information in wlans. In *Proceedings of MILCOM, 2012.*

[3] Yohan Chon, Suyeon Kim, Seungwoo Lee, Dongwon Kim, Yungeun Kim, and Hojung Cha. [n. d.]. Sensing WiFi packets in the air: practicality and implications in urban mobility monitoring. In *Proceedings of ACM Ubicomp, 2014.*

[4] Varoth Chotpitayasunondh and Karen M Douglas. 2016. How "phubbing" becomes the norm: The antecedents and consequences of snubbing via smartphone. *Computers in Human Behavior* (2016).

[5] Mathieu Cunche, Mohamed Ali Kaafar, and Roksana Boreli. [n. d.]. I know who you will meet this evening! linking wireless devices using wi-fi probe requests. In *Proceedings of WoWMoM, 2012.*

[6] Marzieh Dashti, Mohd Amiruddin Abd Rahman, Hamed Mahmoudi, and Holger Claussen. [n. d.]. Detecting co-located mobile users. In *Proceedings of IEEE ICC, 2015.*

[7] Adriano Di Luzio, Alessandro Mei, and Julinda Stefa. [n. d.]. Mind your probes: De-anonymization of large crowds through smartphone WiFi probe requests. In *Proceedings of IEEE INFOCOM, 2016.*

[8] Julien Freudiger. [n. d.]. How talkative is your mobile device?: an experimental study of Wi-Fi probe requests. In *Proceedings of ACM WiSec, 2015.*

[9] Weina Ge, Robert T Collins, and R Barry Ruback. 2012. Vision-based analysis of small groups in pedestrian crowds. *IEEE TPAMI* (2012), 1003–1016.

[10] Nicolas Gillis and François Glineur. 2012. A multilevel approach for nonnegative matrix factorization. *J. Comput. Appl. Math.* (2012), 1708–1723.

[11] A Haigh. 2015. Stop phubbing.

[12] Hande Hong, Chengwen Luo, and Mun Choon Chan. [n. d.]. SocialProbe: Understanding Social Interaction Through Passive WiFi Monitoring. In *Proceedings of ACM MobiQuitous, 2016.*

[13] Timothy M Hospedales, Jian Li, Shaogang Gong, and Tao Xiang. 2011. Identifying rare and subtle behaviors: A weakly supervised joint topic model. *IEEE TPAMI* (2011).

[14] Xueheng Hu, Lixing Song, Dirk Van Bruggen, and Aaron Striegel. [n. d.]. Is there wifi yet?: How aggressive probe requests deteriorate energy and throughput. In *Proceedings of ACM IMC, 2015.*

[15] Shuja Jamil, Sohaib Khan, Anas Basalamah, and Ahmed Lbath. [n. d.]. Classifying smartphone screen ON/OFF state based on wifi probe patterns. In *Proceedings of ACM UbiComp, 2016.*

[16] Hyunsoo Kim and Haesun Park. 2008. Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. *SIAM journal on matrix analysis and applications* (2008), 713–730.

[17] Jingu Kim and Haesun Park. 2008. *Sparse nonnegative matrix factorization for clustering.* Technical Report. Georgia Institute of Technology.

[18] Mikkel Baun Kjærgaard, Martin Wirz, Daniel Roggen, and Gerhard Tröster. [n. d.]. Detecting pedestrian flocks by fusion of multi-modal sensors in mobile phones. In *Proceedings of ACM UbiComp, 2012.*

[19] Mikkel Baun Kjærgaard, Martin Wirz, Daniel Roggen, and Gerhard Tröster. [n. d.]. Mobile sensing of pedestrian flocks in indoor environments using wifi signals. In *Proceedings of IEEE PerCom, 2012.*

[20] Youngki Lee, Chulhong Min, Chanyou Hwang, Jaeung Lee, Inseok Hwang, Younghyun Ju, Chungkuk Yoo, Miri Moon, Uichin Lee, and Junehwa Song. [n. d.]. Sociophone: Everyday face-to-face interaction monitoring platform using multi-phone sensor fusion. In *Proceedings of ACM MobiSys, 2013.*

[21] Jeremy Martin, Travis Mayberry, Collin Donahue, Lucas Foppe, Lamont Brown, Chadwick Riggins, Erik C Rye, and Dane Brown. 2017. A study of MAC address randomization in mobile devices and when it fails. *Proceedings on Privacy Enhancing Technologies* (2017).

[22] S Anne Moorhead, Diane E Hazlett, Laura Harrison, Jennifer K Carroll, Anthea Irwin, and Ciska Hoving. 2013. A new dimension of health care: systematic review of the uses, benefits, and limitations of social media for health communication. *Journal of medical Internet research* (2013).

[23] Kazuaki Nakamura, Tsukasa Ono, and Noboru Babaguchi. [n. d.]. Detection of groups in crowd considering their activity state. In *Proceedings of IEEE ICPR, 2016.*

[24] Daniel Olgum Olgun, Peter A Gloor, and Alex Sandy Pentland. [n. d.]. Capturing individual and group behavior with wearable sensors. In *Proceedings of AAAI, 2009.*

[25] Daniel Olguín Olguín, Benjamin N Waber, Taemie Kim, Akshay Mohan, Koji Ara, and Alex Pentland. 2009. Sensible organizations: Technology and methodology for automatically measuring organizational behavior. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* (2009).

[26] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. 2009. You'll never walk alone: Modeling social behavior for multi-target tracking. In *Computer Vision, 2009 IEEE 12th International Conference on.* IEEE.

[27] Thor S Prentow, Antonio J Ruiz-Ruiz, Henrik Blunck, Allan Stisen, and Mikkel B Kjærgaard. 2015. Spatio-temporal facility utilization analysis from exhaustive wifi monitoring. *Pervasive and Mobile Computing* (2015).

[28] Lorenz Schauer, Martin Werner, and Philipp Marcus. [n. d.]. Estimating crowd densities and pedestrian flows using wi-fi and bluetooth. In *Proceedings of EAI MOBIQUITOUS, 2014.*

[29] Rijurekha Sen, Youngki Lee, Kasthuri Jayarajah, Archan Misra, and Rajesh Krishna Balan. [n. d.]. Grumon: Fast and accurate group monitoring for heterogeneous urban spaces. In *Proceedings of ACM SenSys, 2014.*

[30] Souvik Sen, Romit Roy Choudhury, and Srihari Nelakuditi. [n. d.]. SpinLoc: Spin once to know your location. In *Proceedings of Workshop on ACM MobiSys, 2012.*

[31] Jiaxing Shen, Jiannong Cao, Xuefeng Liu, and Shaojie Tang. 2018. SNOW: Detecting Shopping Groups Using WiFi. *IEEE Internet of Things Journal* (2018).

[32] Jiaxing Shen, Jiannong Cao, Xuefeng Liu, and Chisheng Zhang. 2017. DMAD: Data-Driven Measuring of Wi-Fi Access Point Deployment in Urban Spaces. *ACM Transactions on Intelligent Systems and Technology (TIST)* 9, 1 (2017), 11.

[33] Jiaxing Shen, Oren Lederman, Jiannong Cao, Florian Berg, Shaojie Tang, and Alex Pentland. 2018. GINA: Group Gender Identification Using Privacy-Sensitive Audio Data. In *2018 IEEE International Conference on Data Mining (ICDM).* IEEE, 457–466.

[34] Francesco Solera, Simone Calderara, and Rita Cucchiara. 2016. Socially constrained structural learning for groups detection in crowd. *IEEE TPAMI* (2016).

[35] Sandra Soto, Elva M Arredondo, Miguel T Villodas, John P Elder, Elena Quintanar, and Hala Madanat. 2016. Depression and chronic health conditions among latinos: The role of social networks. *Journal of immigrant and minority health* (2016), 1292–1300.

[36] Robert Tibshirani. 2011. Regression shrinkage and selection via the lasso: a retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* (2011), 273–282.

[37] Geert Vanderhulst, Afra Mashhadi, Marzieh Dashti, and Fahim Kawsar. [n. d.]. Detecting human encounters from wifi radio signals. In *Proceedings of ACM MUM, 2015.*

[38] Mathy Vanhoef, Célestin Matte, Mathieu Cunche, Leonardo S Cardoso, and Frank Piessens. 2016. Why MAC address randomization is not enough: An analysis of Wi-Fi network discovery mechanisms. In *Proceedings of ACM ASIACCS, 2016.* ACM.

[39] T Wiesenthal, G Leduc, P Cazzola, W Schade, and J Köhler. 2011. Mapping innovation in the European transport sector. *An assessment of R&D efforts and priorities, institutional capacities, drivers and barriers to innovation. JRC Scientific and Technical Report* (2011).

[40] Wei Xi, Jizhong Zhao, Xiang-Yang Li, Kun Zhao, Shaojie Tang, Xue Liu, and Zhiping Jiang. [n. d.]. Electronic frog eye: Counting crowd using wifi. In *Proceedings of IEEE INFOCOM, 2014.*

SWeG: Lossless and Lossy Summarization of Web-Scale Graphs

Kijung Shin
School of Electrical Engineering, KAIST, Daejeon, South Korea
kijungs@kaist.ac.kr

Amol Ghoting
LinkedIn Corporation, Mountain View, CA, USA
aghoting@linkedin.com

Myunghwan Kim
LinkedIn Corporation, Mountain View, CA, USA
mykim@cs.stanford.edu

Hema Raghavan
LinkedIn Corporation, Mountain View, CA, USA
hraghavan@linkedin.com

ABSTRACT

Given a terabyte-scale graph distributed across multiple machines, how can we summarize it, with much fewer nodes and edges, so that we can restore the original graph exactly or within error bounds?

As large-scale graphs are ubiquitous, ranging from web graphs to online social networks, compactly representing graphs becomes important to efficiently store and process them. Given a graph, *graph summarization* aims to find its compact representation consisting of (a) a *summary graph* where the nodes are disjoint sets of nodes in the input graph, and each edge indicates the edges between all pairs of nodes in the two sets; and (b) *edge corrections* for restoring the input graph from the summary graph exactly or within error bounds. Although graph summarization is a widely-used graph-compression technique readily combinable with other techniques, existing algorithms for graph summarization are not satisfactory in terms of speed or compactness of outputs. More importantly, they assume that the input graph is small enough to fit in main memory.

In this work, we propose SWeG, a fast parallel algorithm for summarizing graphs with compact representations. SWeG is designed for not only shared-memory but also MapReduce settings to summarize graphs that are too large to fit in main memory. We demonstrate that SWeG is **(a) Fast:** SWeG is up to *5400× faster* than its competitors that give similarly compact representations, **(b) Scalable:** SWeG scales to graphs with *tens of billions* of edges, and **(c) Compact:** combined with state-of-the-art compression methods, SWeG achieves up to *3.4*× better compression than them.

ACM Reference Format:
Kijung Shin, Amol Ghoting, Myunghwan Kim, and Hema Raghavan. 2019. SWeG: Lossless and Lossy Summarization of Web-Scale Graphs. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313402

1 INTRODUCTION

Large-scale graphs are everywhere. Graphs are natural representations of the web, social networks, collaboration networks, internet topologies, citation networks, to name just a few. The rapid growth of the web and its applications has led to large-scale graphs of unprecedented size, such as 3.5 billion web pages connected by 129 billion hyperlinks [36], online social networks with 300 billion connections [11], and 1.15 billion query-URL pairs [30].

Consequently, representing graphs in a storage-efficient manner has become important. In addition to the reduction of hardware costs, compact representation may allow large-scale graphs to fit in main memory of one machine, eliminating expensive I/O over the network or to disk. Moreover, it can lead to performance gains by allowing large fractions of the graphs to reside in cache [7, 48].

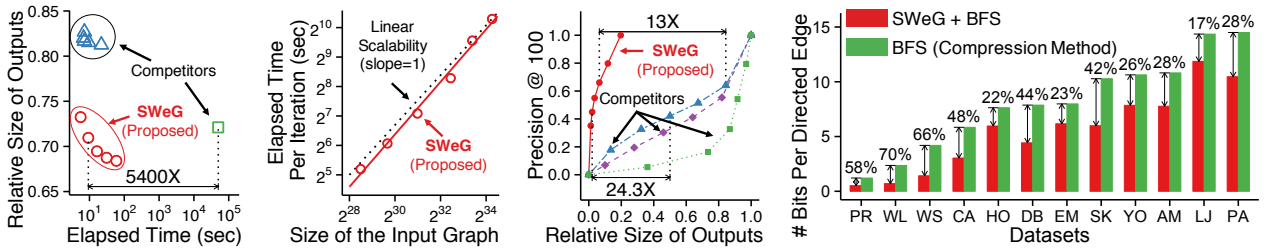
To compactly represent graphs, a variety of graph-compression techniques have been developed, including relabeling nodes [2, 5, 9, 11], employing integer-sequence encoding schemes (e.g., reference, gap, and interval encodings) [5], and encoding common structures (e.g., cliques, bipartite-cores, and stars) with fewer bits [7, 22, 43]. These techniques are either *lossless* and *lossy* depending on whether the input graph can be reconstructed exactly from their outputs.

Graph summarization [38] is a widely-used graph-compression technique. It aims to find a compact representation of a given graph $\mathcal{G}$ consisting of a summary graph and edge corrections. The *summary graph* $\overline{\mathcal{G}}$ is a graph where the nodes are disjoint sets of nodes in $\mathcal{G}$ and each edge indicates the edges between all pairs of nodes in the two sets. The *edge corrections* (i.e., edges to be inserted, and edges to be deleted) are for restoring $\mathcal{G}$ from $\overline{\mathcal{G}}$ exactly (in cases of lossless summarization) or within given error bounds (in cases of lossy summarization). The outputs can be considered as graphs: (a) the summary graph, (b) the graph consisting of the edges to be inserted, and (c) the graph consisting of the edges to be deleted.

As a graph-compression technique, graph summarization has the following desirable properties: **(a) Combinable**: since graph summarization produces three graphs, as explained in the previous paragraph, each of them can be further compressed by any other graph compression technique, **(b) Adjustable**: the trade-off between compression rates and information loss can be adjusted by given error bounds, and **(c) Queryable**: neighbor queries (i.e., finding the neighboring nodes of a given node) can be answered efficiently on the outputs of graph summarization (see Appendix A).

However, existing algorithms for graph summarization are not satisfactory in terms of speed [38, 44] or compactness of output representations [20]. These serial algorithms either have high computational complexity or significantly sacrifice compactness of outputs for lower complexity. Moreover, they assume that the input graph is small enough to fit in main memory. The largest graph used in the previous studies has just about 10 million edges [20, 38, 44]. As

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313402



(a) Fast and Compact (Lossless Summarization)

(b) Scalable (Lossless Summarization)

(c) Compact and Accurate (Lossy Summarization)

(d) Compact (Lossless Compression)

Figure 1: SWeG is fast and scalable with compact and accurate representations. (a) SWeG is faster with more compact representations than the other lossless-summarization methods. (b) SWeG scales linearly with the size of the input graph, successfully scaling to graphs with over 20 billions edges. (c) The lossy version of SWeG yields more compact and accurate representations than the other lossy-summarization methods. (d) Combining SWeG and an advanced compression technique [2] yields up to 3.4× more compact representations than using the technique alone. See Section 4 for details.

large-scale graphs often do not fit in main memory, improving the scalability of graph summarization remains an important challenge.

To address this challenge, we propose SWeG (**S**ummarizing **We**b-Scale **G**raphs), a fast parallel algorithm for summarizing large-scale graphs with compact representations. SWeG is designed for both shared-memory and MapReduce settings. In a nutshell, SWeG repeats (a) dividing the input graph into small subgraphs and (b) processing the subgraphs in parallel without having to load the entire graph in main memory. SWeG also adjusts, in each iteration, how aggressively it merges nodes within subgraphs, and this turns out to be crucial for obtaining compact representations.

Extensive experiments with 13 real-world graphs show that SWeG significantly outperforms existing graph-summarization methods and enhances the best compression techniques, as shown in Figure 1. Specifically, SWeG provides the following advantages:

- **Speed**: SWeG is up to *5, 400× faster* than its competitors that give similarly compact representations (Figure 1(a)).
- **Scalability**: SWeG scales to graphs with *tens of billions* of edges, showing near-linear data and machine scalability (Figure 1(b)).
- **Compression**: Combined with advanced graph-compression methods, SWeG yields up to *3.4× better* compression than the methods (Figure 1(d)).

In Section 2, we provide notations and a formal problem definition. In Section 3, we present our proposed algorithm, SWeG. In Section 4, we provide our experimental results. After discussing related work in Section 5, we offer conclusions in Section 6.

2 NOTATIONS AND PROBLEM DEFINITION

Notations and Concepts. See Table 1 for frequently-used symbols and Figure 2 for an illustration of concepts. Consider a simple undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V}$ and edges $\mathcal{E}$. We denote each node in $\mathcal{V}$ by a lowercase (e.g., v) and each edge in $\mathcal{E}$ by an unordered pair (e.g., $\{u, v\}$). We denote the set of neighbors of a node $v \in \mathcal{V}$ in $\mathcal{G}$ by $N_v \subset \mathcal{V}$.

A *summary graph* of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, denoted by $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, is a graph whose nodes are a partition of $\mathcal{V}$ (i.e., disjoint and exhaustive subsets of $\mathcal{V}$). That is, each node $v \in \mathcal{V}$ belongs to exactly one node in $\mathcal{S}$. We call nodes and edges in $\overline{\mathcal{G}}$ *supernodes* and *superedges*. We denote each supernode by an uppercase (i.e., A). $\mathcal{P}$ may include self-loops, and we let $\mathcal{P}^* \subset \mathcal{P}$ be the set of non-loop superedges.

Given a summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and *corrections* $\mathcal{C} =< \mathcal{C}^+, \mathcal{C}^- >$, where $\mathcal{C}^+$ denotes the set of edges to be inserted and $\mathcal{C}^-$ denotes the set of edges to be deleted, a graph $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$, which we call a *restored graph*, is created by the following steps:

(1) For each superedge $\{A, B\} \in \mathcal{P}$, all pairs of distinct nodes in A and B (i.e.,. $\{\{u, v\} : u \in A, v \in B, u \neq v\}$) are added to $\hat{\mathcal{E}}$,
(2) Each edge in $\mathcal{C}^+$ is added to $\hat{\mathcal{E}}$,
(3) Each edge in $\mathcal{C}^-$ is removed from $\hat{\mathcal{E}}$.

We let the neighbors of each node $v \in \mathcal{V}$ in $\hat{\mathcal{G}}$ be $\hat{N}_v \subset \mathcal{V}$. On $\overline{\mathcal{G}}$ and $\mathcal{C}$, neighbor queries (i.e., finding $\hat{N}_v$ for a query node $v \in \mathcal{V}$) can be answered rapidly without restoring entire $\hat{\mathcal{G}}$ (see Appendix A).

Problem Definition. The large-scale graph summarization problem, which we address in this work, is defined in Problem 1.

Problem 1 (Large-scale Graph Summarization).

(1) **Given**: *a large-scale graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which may or may not fit in main memory, and an error bound ϵ (≥ 0)*
(2) **Find:** *a summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, and corrections $\mathcal{C} =< \mathcal{C}^+, \mathcal{C}^- >$*

Table 1: Table of frequently-used symbols.

Symbol	Definition
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	input graph with nodes $\mathcal{V}$ and edges $\mathcal{E}$
N_v	set of neighbors of node v in $\mathcal{G}$
$\mathcal{C} =< \mathcal{C}^+, \mathcal{C}^- >$	edge corrections (i.e., edges insertions and deletions)
$\mathcal{C}^+$	set of edges to be inserted
$\mathcal{C}^-$	set of edges to be deleted
$\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$	summary graph with supernodes $\mathcal{S}$ and superedges $\mathcal{P}$
$\mathcal{P}^*$	non-loop superedges in $\mathcal{P}$
$\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$	graph restored from $\overline{\mathcal{G}}$ and $\mathcal{C}$
$\hat{N}_v$	set of neighbors of node v in $\hat{\mathcal{G}}$
ϵ	error bound
T	number of iterations
$\theta(t)$	merging threshold in the t-th iteration

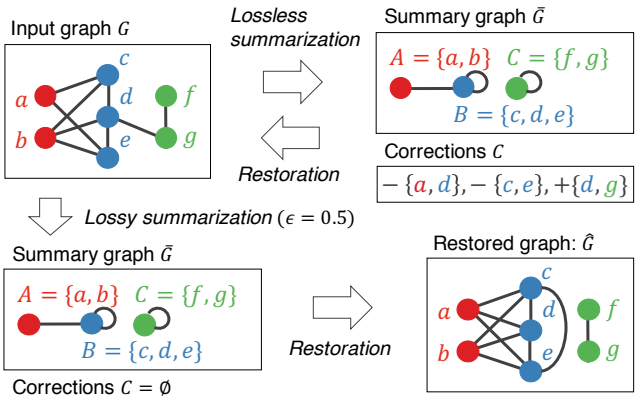


Figure 2: Illustration of graph summarization. Lossless summarization of the input graph (upper left) yields a summary graph and corrections (upper right) from which the input graph is restored exactly. Lossy summarization of the input graph (upper left) yields a summary graph and corrections (lower left). The restored graph (lower right) satisfies Eq. (2). Note that the outputs of the lossless and lossy graph summarization have fewer edges than the input graph.

(3) **to Minimize:**

$$|\mathcal{P}^*| + |C^+| + |C^-| \tag{1}$$

(4) **Subject to:** *the restored graph* $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$ *satisfies*

$$|N_v - \hat{N}_v| + |\hat{N}_v - N_v| \leq \epsilon |N_v|, \ \forall v \in \mathcal{V}. \tag{2}$$

The objective (Eq. (1)), which we aim to minimize, measures the size of the output representation by the count of non-loop (super) edges. We exclude all self-loops in $\mathcal{P}$ from the objective since they can be encoded concisely using 1 bit per supernode regardless of their count. The constraints (Eq. (2)) [38] states that the neighbors N_v of each node $v \in \mathcal{V}$ in the input graph and the node's neighbors $\hat{N}_v$ in the restored graph $\hat{\mathcal{G}}$ should be similar enough so that the size of their symmetric difference (i.e., $(N_v \cup \hat{N}_v) - (\hat{N}_v \cap N_v)$) is at most a certain proportion of the size of the node's neighbors N_v in the input graph. The proportion is given as a parameter ϵ, which we call an *error bound*, and it controls the trade-off between compression rates and the amount of information loss. If $\epsilon = 0$, then $N_v = \hat{N}_v$ holds for every node $v \in \mathcal{V}$ and thus the restored graph $\hat{\mathcal{G}}$ is equal to the input graph $\mathcal{G}$. Thus, Problem 1 is **lossless summarization** if $\epsilon = 0$ and **lossy summarization** if $\epsilon > 0$. See Figure 2 for lossless and lossy summarization of a toy graph.

3 PROPOSED ALGORITHM: SWEG

We present our proposed algorithm, SWeG (**S**ummarizing **We**b-Scale **G**raphs). SWeG provides approximate solutions to the lossless and lossy graph summarization problems (i.e., Problems 1). In Section 3.1, we provide an overview of SWeG then describe each step of it in detail. In Sections 3.2 and 3.3, we discuss parallelizing SWeG in shared-memory and MapReduce settings. In Section 3.4, we analyze its time complexity and memory requirements. In Section 3.5, we present SWeG+, an algorithm for further compression.

Algorithm 1 Overview of SWeG

Input: input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, number of iterations T, error bound ϵ
Output: summary graph $\bar{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, corrections C
1: initialize supernodes $\mathcal{S}$ to $\{\{v\} : v \in \mathcal{V}\}$
2: **for** $t = 1 \ldots T$ **do**
3: divide $\mathcal{S}$ into disjoint groups ▷ Algorithm 2
4: merge some supernodes within each group ▷ Algorithm 3
5: encode edges $\mathcal{E}$ into superedges $\mathcal{P}$ and corrections C ▷ Algorithm 4
6: **if** $\epsilon > 0$ **then** drop some (super) edges from $\mathcal{P}$ and C ▷ Algorithm 5
7: **return** $\bar{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and C

Algorithm 2 Dividing Step of SWeG

Input: input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, current supernodes $\mathcal{S}$
Output: disjoint groups of supernodes: $\{\mathcal{S}^{(1)}, \ldots, \mathcal{S}^{(k)}\}$
1: generate a random bijective hash function $h : \mathcal{V} \to \{1, \ldots, |\mathcal{V}|\}$
2: **for each** supernode $A \in \mathcal{S}$ **do**
3: **for each** node $v \in A$ **do**
4: $f(v) \leftarrow \min(\{h(u) : u \in N_v \text{ or } u = v\})$ ▷ shingle of node v
5: $F(A) \leftarrow \min(\{f(v) : v \in A\})$ ▷ shingle of supernode A
6: divide the supernodes in $\mathcal{S}$ into $\{\mathcal{S}^{(1)}, \ldots, \mathcal{S}^{(k)}\}$ by their $F(\cdot)$ value
7: **return** $\{\mathcal{S}^{(1)}, \ldots, \mathcal{S}^{(k)}\}$

3.1 Description of the Algorithm

We first give an overview of SWeG, and then describe each step of it in detail. In this subsection, for simplicity, we assume that SWeG is executed serially and the input graph is small enough to fit in main memory of one machine. Parallelization and distributed processing are discussed in Sections 3.2 and 3.3.

3.1.1 **Overview (Algorithm 1).** SWeG requires an input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the number of iterations T, and an error bound ϵ, as in Problem 1. SWeG first initializes the set $\mathcal{S}$ of supernodes so that (a) each supernode consists of a node in $\mathcal{V}$ and (b) each node in $\mathcal{V}$ belongs to one supernode (line 1 of Algorithm 1). Then, SWeG updates $\mathcal{S}$ by repeating the following steps T times (line 2):

- **Dividing step (line 3, Section 3.1.2)**: divides $\mathcal{S}$ into disjoint groups, each of which is composed of supernodes with similar connectivity. Different groups are obtained in each iteration.
- **Merging step (line 4, Section 3.1.3)**: merges some supernodes within each group in a greedy manner.

Dividing $\mathcal{S}$ into small groups makes SWeG faster and more memory efficient by allowing it to process different groups in parallel without having to load entire $\mathcal{G}$ in memory (see Sections 3.2 and 3.3). After updating $\mathcal{S}$, SWeG performs the following steps:

- **Encoding step (line 5, Section 3.1.4)**: encodes the edges $\mathcal{E}$ into superedges $\mathcal{P}$ and corrections C so that the count of non-loop (super) edges (i.e., Eq. (1)) is minimized given supernodes $\mathcal{S}$.
- **Dropping step (line 6, Section 3.1.5)**: makes $\mathcal{P}$ and C more compact by dropping some (super) edges from them within the error bounds (i.e., Eq. (2)) in cases of lossy summarization.

Lastly, SWeG returns the summary graph $\bar{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and the corrections C as its outputs (line 7).

3.1.2 **Dividing Step (Algorithm 2).** The dividing step aims to divide the supernodes $\mathcal{S}$ into disjoint groups of supernodes with similar connectivity. To this end, we extend the shingle of nodes, for which it is known that two nodes have the same shingle with

Algorithm 3 Merging Step of SWeG

Input: input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, current supernodes $\mathcal{S}$, current iteration t, disjoint groups of supernodes $\{\mathcal{S}^{(1)}, \ldots, \mathcal{S}^{(k)}\}$
Output: updated supernodes $\mathcal{S}$

1: **for each** group $\mathcal{S}^{(i)} \in \{\mathcal{S}^{(1)}, \ldots, \mathcal{S}^{(k)}\}$ **do**
2: $\quad Q \leftarrow \mathcal{S}^{(i)}$
3: $\quad$ **while** $|Q| > 1$ **do**
4: $\quad\quad$ pick and remove a random supernode A from Q
5: $\quad\quad B \leftarrow \arg\max_{C \in Q} \textit{SuperJaccard}(A, C)$ ▷ Eq. (4)
6: $\quad\quad$ **if** $\textit{Saving}(A, B, \mathcal{S}) \geq \theta(t)$ **then** ▷ Eq. (3) and Eq. (5)
7: $\quad\quad\quad \mathcal{S} \leftarrow (\mathcal{S} - \{A, B\}) \cup \{A \cup B\}$ ▷ merge A and B
8: $\quad\quad\quad \mathcal{S}^{(i)} \leftarrow (\mathcal{S}^{(i)} - \{A, B\}) \cup \{A \cup B\}$
9: $\quad\quad\quad Q \leftarrow (Q - \{B\}) \cup \{A \cup B\}$ ▷ replace B with $A \cup B$
10: **return** $\mathcal{S}$

probability equal to the jaccard similarity of their neighbor sets [6], to supernodes. Specifically, we define the *shingle* $F(A)$ of each supernode $A \in \mathcal{S}$ as the smallest shingle of the nodes in A. Then, two supernodes $A \neq B \in \mathcal{S}$ are more likely to have the same shingle if the nodes in A and those in B have similar connectivity. Formally, given a random bijective hash function $h : \mathcal{V} \to \{1, ..., |\mathcal{V}|\}$,

$$F(A) := \min_{v \in A}(f(v)),$$

where $f(v) := \min_{u \in N_v \text{ or } u=v} h(u)$ is the shingle of node $v \in \mathcal{V}$, and N_v is the set of neighbors of node $v \in \mathcal{V}$ in the input graph $\mathcal{G}$. SWeG generates such a hash function h by shuffling the order of the nodes in $\mathcal{V}$ and mapping each i-th node to i (line 1 of Algorithm 2) and computes the shingle of every supernode in $\mathcal{S}$ (lines 2-5). Lastly, SWeG divides the supernodes $\mathcal{S}$ into disjoint groups $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(k)}\}$ by their shingle and returns the groups (lines 6-7). In cases where large groups exist, Algorithm 2 can be applied recursively to each of the large groups with a new hash function h.

3.1.3 **Merging Step (Algorithm 3).** To describe this step, we first define several concepts. We define the *cost* of each supernode $A \in \mathcal{S}$ given supernodes $\mathcal{S}$, denoted by $\textit{Cost}(A, \mathcal{S})$, as the amount of increase in Eq. (1) (i.e., the count of non-loop (super) edges in outputs) due to the edges adjacent to any node in A (see the encoding step in Section 3.1.4). Then, we define the *saving* due to the merger between supernodes $A \neq B \in \mathcal{S}$ given supernodes $\mathcal{S}$ as

$$\textit{Saving}(A, B, \mathcal{S}) := 1 - \frac{\textit{Cost}(A \cup B, (\mathcal{S} - \{A, B\}) \cup \{A \cup B\})}{\textit{Cost}(A, \mathcal{S}) + \textit{Cost}(B, \mathcal{S})}, \quad (3)$$

where $\textit{Cost}(A, \mathcal{S}) + \textit{Cost}(B, \mathcal{S})$ is the cost of A and B before their merger, and $\textit{Cost}(A \cup B, (\mathcal{S} - \{A, B\}) \cup \{A \cup B\})$ is their cost after their merger. That is, $\textit{Saving}(A, B, \mathcal{S})$ is the ratio of the cost reduction due to the merger and the cost before the merger. Lastly, we define the *supernode jaccard similarity* between supernodes $A \neq B \in \mathcal{S}$ as

$$\textit{SuperJaccard}(A, B) := \frac{\sum_{v \in N_A \cup N_B} \min(w(A, v), w(B, v))}{\sum_{v \in N_A \cup N_B} \max(w(A, v), w(B, v))}, \quad (4)$$

where $N_A := \bigcup_{v \in A} N_v$ is the set of nodes adjacent to any node in supernode $A \in \mathcal{S}$, and $w(A, v) := |\{u \in A : \{u, v\} \in \mathcal{E}\}|$ is the number of nodes in supernode $A \in \mathcal{S}$ adjacent to node $v \in \mathcal{V}$. $\textit{SuperJaccard}(A, B)$ measures the similarity of A and B in terms of their connectivity. Notice that it is 1 if A and B have the same connectivity (i.e., $w(A, v) = w(B, v)$ for every $v \in N_A \cup N_B$) and it is 0 if their nodes have no common neighbors (i.e., $N_A \cap N_B = \emptyset$).

Given disjoint groups of supernodes $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(k)}\}$, the merging step merges some pairs of supernodes within each group $\mathcal{S}^{(i)}$ in a greedy manner, as described in Algorithm 3. Notice that, in line 5, SWeG uses $\textit{SuperJaccard}(A, B)$ instead of $\textit{Saving}(A, B, \mathcal{S})$, which is a more straightforward choice, to find a candidate pair of supernodes $A \neq B \in \mathcal{S}^{(i)}$. This is because (a) $\textit{SuperJaccard}(A, B)$ is cheaper to compute than $\textit{Saving}(A, B, \mathcal{S})$ and (b) intuitively, $\textit{Saving}(A, B, \mathcal{S})$ tends to be high when A and B have similar connectivity. Computing $\textit{SuperJaccard}(A, C)$ instead of $\textit{Saving}(A, C, \mathcal{S})$ for every supernode $C \in Q$ in line 5 leads to a significant improvement in speed with small loss in the compactness of outputs, as shown empirically in Section 4.2. In line 6, the *merging threshold* $\theta(t)$ is set as in Eq. (5) so that SWeG gradually shifts from exploration (of supernodes in the other groups) to exploitation (of supernodes in the same group).

$$\theta(t) := \begin{cases} (1 + t)^{-1} & \text{if } t < T, \\ 0 & \text{if } t = T \end{cases} \quad (5)$$

This decreasing threshold is crucial for obtaining compact output representations, as shown empirically in Section 4.2.

3.1.4 **Encoding Step (Algorithm 4).** Given the supernodes $\mathcal{S}$ from the previous steps, the encoding step encodes the edges $\mathcal{E}$ of the input graph into superedges $\mathcal{P}$ and corrections $C =< C^+, C^- >$. We first describe how to encode edges that connect different supernodes (lines 3-5 of Algorithm 4). For each supernode pair $A \neq B \in \mathcal{S}$, we let $\mathcal{E}_{AB} \subset \mathcal{E}$ be the set of edges connecting A and B, and π_{AB} be the set of all pairs of nodes in A and B. That is,

$$\mathcal{E}_{AB} := \{\{u, v\} \subset \mathcal{V} : u \in A, v \in B, \{u, v\} \in \mathcal{E}\}, \quad (6)$$

$$\pi_{AB} := \{\{u, v\} \subset \mathcal{V} : u \in A, v \in B\}. \quad (7)$$

Recall how a graph is restored from $\mathcal{P}$ and C in Section 2. Then, the two options to encode the edges in $\mathcal{E}_{AB}$ are as follows:

(a) *without superedges*: merge $\mathcal{E}_{AB}$ into C^+,
(b) *with a superedge*: add $\{A, B\}$ to $\mathcal{P}$; merge $(\pi_{AB} - \mathcal{E}_{AB})$ into C^-.

Since (a) and (b) increase our objective (i.e., Eq. (1)) by $|\mathcal{E}_{AB}|$ and $(1 + |\pi_{AB}| - |\mathcal{E}_{AB}|)$, respectively, SWeG chooses (a) if $|\mathcal{E}_{AB}| \leq |\pi_{AB}|/2 = |A| \cdot |B|/2$ (line 4). Otherwise, it chooses (b) (line 5).

SWeG encodes edges between nodes within each supernode in a similar manner (lines 6-7). For each supernode $A \in \mathcal{S}$, we let $\mathcal{E}_{AA} \subset \mathcal{E}$ be the set of edges between nodes within A, and π_{AA} be the set of all pairs of distinct nodes within A. That is,

$$\mathcal{E}_{AA} := \{\{u, v\} \subset \mathcal{V} : u \neq v \in A, \{u, v\} \in \mathcal{E}\}, \quad (8)$$

$$\pi_{AA} := \{\{u, v\} \subset \mathcal{V} : u \neq v \in A\}. \quad (9)$$

Then, the two options to encode the edges in $\mathcal{E}_{AA}$ are as follows:

(c) *without superloops*: merge $\mathcal{E}_{AA}$ into C^+,
(d) *with a superloop*: add $\{A, A\}$ to $\mathcal{P}$; merge $(\pi_{AA} - \mathcal{E}_{AA})$ into C^-.

Since (c) and (d) increase our objective (i.e., Eq. (1)) by $|\mathcal{E}_{AA}|$ and $(|\pi_{AA}| - |\mathcal{E}_{AA}|)$, respectively, SWeG chooses (c) if $|\mathcal{E}_{AA}| \leq |\pi_{AA}|/2 = |A| \cdot (|A| - 1)/4$ (line 6). Otherwise, it chooses (d) (line 7).

3.1.5 **Dropping Step (Algorithm 5).** The dropping step is an optional step for lossy summarization (i.e., when $\epsilon > 0$). This step is skipped if lossless summarization is needed (i.e., when $\epsilon = 0$). Given the summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and corrections $C =< C^+, C^- >$ from the previous step, the dropping step drops some (super) edges from $\mathcal{P}$ and C to make the output representation

Algorithm 4 Encoding Step of SWeG

Input: input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, supernodes $\mathcal{S}$
Output: summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, corrections $C =< C^+, C^- >$

1: $\mathcal{P} \leftarrow \emptyset$; $C^+ \leftarrow \emptyset$; $C^- \leftarrow \emptyset$;
2: **for each** supernode $A \in \mathcal{S}$ **do**
3: **for each** supernode B ($\neq A$) where $\mathcal{E}_{AB} \neq \emptyset$ **do** ▷ Eq. (6)
4: **if** $\mathcal{E}_{AB} \leq \frac{|A|\cdot|B|}{2}$ **then** $C^+ \leftarrow C^+ \cup \mathcal{E}_{AB}$
5: **else** $\mathcal{P} \leftarrow \mathcal{P} \cup \{\{A, B\}\}$; $C^- \leftarrow C^- \cup (\pi_{AB} - \mathcal{E}_{AB})$ ▷ Eq. (7)
6: **if** $\mathcal{E}_{AA} \leq \frac{|A|\cdot(|A|-1)}{4}$ **then** $C^+ \leftarrow C^+ \cup \mathcal{E}_{AA}$ ▷ Eq. (8)
7: **else** $\mathcal{P} \leftarrow \mathcal{P} \cup \{\{A, A\}\}$; $C^- \leftarrow C^- \cup (\pi_{AA} - \mathcal{E}_{AA})$ ▷ Eq. (9)
8: **return** $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and $C =< C^+, C^- >$

more compact (i.e., to further reduce our objective Eq. (1)) without changing more than ϵ of the neighbors of each node (i.e., within the error bounds given in Eq. (2)). SWeG first initializes the *change limit* c_v of each node $v \in \mathcal{V}$ to $\epsilon \cdot |N_v|$, which is the right-hand side of Eq. (2) (line 1 of Algorithm 5). Then, within the change limit of each node, SWeG drops some adjacent edges from C^+, C^-, and $\mathcal{P}$, as described in lines 2-4, lines 5-7, and lines 8-12, respectively. Notice that, when a superedge $\{A, B\}$ is dropped, the total decrement in the change limits is proportional to $|A| \cdot |B|$, which is used to sort the superedges in line 8. Lastly, SWeG returns the updated summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and corrections $C =< C^+, C^- >$ that satisfy the error bounds given in Eq. (2), as shown in Theorem 3.1 (line 13).

Theorem 3.1 (Error Bounds). *Given an input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, and corrections $C =< C^+, C^- >$ satisfying that the restored graph $\hat{\mathcal{G}}$ is equal to $\mathcal{G}$, Algorithm 5 returns $\overline{\mathcal{G}}$ and C that satisfy Eq. (2).*

Proof. From $\hat{\mathcal{G}} = \mathcal{G}$, $\hat{N}_v = N_v$ holds for every node $v \in \mathcal{V}$ at the beginning of the algorithm. Thus, after the change limit c_v of each node $v \in \mathcal{V}$ is initialized to $\epsilon \cdot |N_v|$ in line 1, Eq. (10) holds.

$$c_v \leq \epsilon \cdot |N_v| - |\hat{N}_v - N_v| - |N_v - \hat{N}_v|, \forall v \in \mathcal{V}. \tag{10}$$

Recall how $\hat{\mathcal{G}}$ is constructed from $\overline{\mathcal{G}}$ and C in Section 2. Eq. (10) still holds after C^+ is processed in lines 2-4 since dropping an edge from C^+ decreases c_v by 1, increases $|N_v - \hat{N}_v|$ by at most 1, and keeps $|\hat{N}_v - N_v|$ the same for each adjacent node $v \in \mathcal{V}$. Likewise, Eq. (10) still holds after C^- is processed in lines 5-7 since dropping an edge in C^- decreases c_v by 1, increases $|\hat{N}_v - N_v|$ by at most 1, and keeps $|N_v - \hat{N}_v|$ the same for each adjacent node $v \in \mathcal{V}$. Similarly, we can show that Eq. (10) still holds after $\mathcal{P}$ is processed in lines 8-12. Eq. (11) is enforced by lines 3, 6, and 9.

$$c_v \geq 0, \forall v \in \mathcal{V}. \tag{11}$$

Eq. (10) and Eq. (11) imply Eq. (2). ■

3.2 Parallelization in Shared Memory

We describe how each step of SWeG (Algorithm 1) is parallelized in shared-memory environments. In the dividing step (Algorithm 2), the supernodes in line 2 are processed independently in parallel. In the merging step (Algorithm 3), the groups of supernodes in line 1 are processed in parallel. In lines 6 and 7, the accesses and updates of $\mathcal{S}$ are synchronized. In the encoding step (Algorithm 4), the supernodes in line 2 are processed independently in parallel. Specifically, each thread has its own copies of $\mathcal{P}$, C^+, and C^-; and the copies are merged once, after all supernodes are processed. Lastly, in the dropping step (line 5), parallel merge sort [24] is used when sorting $\mathcal{P}$ in line 8. Although the other parts of the dropping step are executed serially, they take a negligible portion of the total execution time because they are executed only once, while the dividing and merging steps are repeated multiple times.

Algorithm 5 Dropping Step of SWeG (Optional)

Input: input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, error bound ϵ
summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, corrections $C =< C^+, C^- >$
Output: updated summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$,
updated corrections $C =< C^+, C^- >$

1: $c_v \leftarrow \epsilon \cdot |N_v|$ for each node $v \in \mathcal{V}$ ▷ change limits of nodes
2: **for each** edge $\{u, v\} \in C^+$ **do**
3: **if** $c_u \geq 1$ and $c_v \geq 1$ **then**
4: $C^+ \leftarrow C^+ - \{\{u, v\}\}$; $c_u \leftarrow c_u - 1$; $c_v \leftarrow c_v - 1$
5: **for each** edge $\{u, v\} \in C^-$ **do**
6: **if** $c_u \geq 1$ and $c_v \geq 1$ **then**
7: $C^- \leftarrow C^- - \{\{u, v\}\}$; $c_u \leftarrow c_u - 1$; $c_v \leftarrow c_v - 1$
8: **for each** superedge $\{A, B\} \in \mathcal{P}$ in the increasing order of $|A| \cdot |B|$ **do**
9: **if** $A \neq B$ and ($\forall v \in A$, $c_v \geq |B|$) and ($\forall v \in B$, $c_v \geq |A|$) **then**
10: $\mathcal{P} \leftarrow \mathcal{P} - \{\{A, B\}\}$;
11: **for each** $v \in A$ **do** $c_v \leftarrow c_v - |B|$
12: **for each** $v \in B$ **do** $c_v \leftarrow c_v - |A|$
13: **return** $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and $C =< C^+, C^- >$

3.3 Distributed Processing with MapReduce

We describe how each step of SWeG is implemented in the MapReduce framework for large-scale graphs not fitting in main memory. We assume that the input graph $\mathcal{G}$ is stored in a file in a distributed file system where each record describes the nodes in a supernode and the neighbors of the nodes in $\mathcal{G}$. Specifically, the record $R(A)$ for a supernode $A \in \mathcal{S}$ is in the following format:

$$R(A) := (\text{id of } A, |A|, \underbrace{(u, |N_u|, (\overbrace{v, \ldots, w}^{|N_u|})), \ldots, (x, |N_x|, (\overbrace{y, \ldots, z}^{|N_x|}))}_{|A|}), \tag{12}$$

where $(u, |N_u|, (v, \ldots, w))$ describes the neighbors of node $u \in A$.

Dividing and Merging Steps: First, each iteration of the dividing and merging steps (Algorithms 2 and 3) is performed by the following MapReduce job:

- **Map-1**: The hash function h is broadcast to the mappers. Each mapper repeats taking a record $R(A)$, computing $F(A)$ (lines 3-5 of Algorithm 2), and emitting $< F(A), R(A) >$.
- **Reduce-1:** The supernodes $\mathcal{S}$ are broadcast to the reducers. Each reducer repeats taking $\{R(A) | A \in \mathcal{S}^{(i)}\}$ for a group $\mathcal{S}^{(i)}$, updating $\mathcal{S}^{(i)}$ (lines 2-9 of Algorithm 3), and emitting $R(A)$ for each supernode A in the updated $\mathcal{S}^{(i)}$. In the end, each reducer writes the updates in $\mathcal{S}$ to the distributed file system.

Note that updates in $\mathcal{S}$ are not shared among the reducers during the reduce stage. However, in our experiments, the effect of this lazy synchronization on the compactness of output representations was negligible.

Encoding Step: Next, the following map-only job performs the encoding step (Algorithm 4):

- **Map-2:** The supernodes $\mathcal{S}$ are broadcast to the mappers. Each mapper repeats taking a record $R(A)$, encoding the edges adjacent to any node in A (lines 3-7 of Algorithm 4), and emitting

the new (super) edges in $\mathcal{P}$, C^+, and C^-. Different output paths are used for $\mathcal{P}$, C^+, and C^-.

Dropping Step: Lastly, for the dropping step (Algorithm 5), Map-3 initializes the change limit of each node (line 1); and Map-4 and Reduce-4 sort the superedges in $\mathcal{P}$ (line 9).

- **Map-3**: Each mapper repeats taking a record $R(A)$ and emitting $< v, \epsilon \cdot |N_v| >$ for each node $v \in A$.
- **Map-4**: The input file, which is an output of Map-2, lists the superedges in $\mathcal{P}$. Each mapper repeats taking a superedge $\{A, B\}$ and emitting $< (|A| \cdot |B|, \{A, B\}), \emptyset >$.
- **Reduce-4**: A single reducer repeats taking a superedge $\{A, B\} \in \mathcal{P}$ and emitting it. The superedges are sorted in the shuffle stage.

The other parts of the dropping step are processed serially. Specifically, after loading the nodes' change limits (the output of Map-3) in memory, our implementation repeats reading a (super) edge in C^+ (an output of Map-2), C^- (an output of Map-2), and $\mathcal{P}$ (the output of Reduce-4) and writing the (super) edge to the output file if it is not dropped. Note that entire C^+, C^-, or $\mathcal{P}$ is not loaded in memory at once. These serial parts take a small portion of the total execution time because they are executed only once, while the dividing and merging steps are repeated multiple times.

3.4 Complexity Analysis

We analyze the time complexity and memory requirements of SWeG. To this end, we let $\mathcal{E}_A := \{\{u, v\} \in \mathcal{E} : u \in A\}$ be the set of edges adjacent to any node in supernode $A \in \mathcal{S}$ and $\mathcal{E}^{(i)} := \bigcup_{A \in \mathcal{S}^{(i)}} \mathcal{E}_A$ be the set of edges adjacent to any node in any supernode in group $\mathcal{S}^{(i)}$. We also let k be the number of groups of supernodes from the dividing step. Since the groups are disjoint,

$$\sum_{i=1}^{k} |\mathcal{E}^{(i)}| \leq \sum_{A \in \mathcal{S}} |\mathcal{E}_A| \leq 2|\mathcal{E}|. \tag{13}$$

Since the encoding step (Algorithm 4) yields outputs with no more (super) edges than ($\mathcal{P} \leftarrow \emptyset, C^+ \leftarrow \mathcal{E}, C^- \leftarrow \emptyset$),

$$|C^+| + |C^-| + |\mathcal{P}| \leq |\mathcal{E}|. \tag{14}$$

Lastly, we assume that $|\mathcal{V}| \leq |\mathcal{E}|$, for simplicity.

3.4.1 Time Complexity. The dividing step (Algorithm 2) takes $O(|\mathcal{E}|)$ time since its computational bottleneck is to compute the shingles of all supernodes (lines 2-5), which requires accessing every edge in $\mathcal{E}$ twice. The merging step (Algorithm 3) takes $O(\sum_{i=1}^{k} |\mathcal{S}^{(i)}| \cdot |\mathcal{E}^{(i)}|)$ time, since when each group $\mathcal{S}^{(i)}$ is processed (lines 2-9), the number of iterations is $|\mathcal{S}^{(i)}| - 1$ and each iteration takes $O(|\mathcal{E}^{(i)}|)$ time. The encoding step (Algorithm 4) takes $O(\sum_{A \in \mathcal{S}} |\mathcal{E}_A|) = O(|\mathcal{E}|)$ time since to process each supernode A (lines 3-7) takes $O(|\mathcal{E}_{AA}| + \sum_{B(\neq A):\mathcal{E}_{AB} \neq \emptyset} |\mathcal{E}_{AB}|) = O(|\mathcal{E}_A|)$ time. This follows from $|\pi_{AB}| < 2 \cdot |\mathcal{E}_{AB}|$ and $|\pi_{AA}| < 2 \cdot |\mathcal{E}_{AA}|$ in lines 5 and 7, which are due to the conditions in lines 4 and 6. Lastly, the dropping step (Algorithm 5) takes $O(|\mathcal{E}| + |C^+| + |C^-| + |\mathcal{P}|) = O(|\mathcal{E}|)$ time (see Eq. (14)). Note that $\mathcal{P}$ (line 8) can be sorted in $O(|\mathcal{P}| + |\mathcal{E}|)$ time using any linear-time integer sort (e.g., counting sort) since $|A| \cdot |B| \in \{1, ..., 2 \cdot |\mathcal{E}|\}$ for every $\{A, B\} \in \mathcal{P}$ due to the conditions in lines 4 and 6 of Algorithm 4. Thus, all the steps except for the merging step take $O(|\mathcal{E}|)$ time. The merging step, whose complexity is $O(\sum_{i=1}^{k} |\mathcal{S}^{(i)}| \cdot |\mathcal{E}^{(i)}|)$, also takes $O(|\mathcal{E}|)$ time if $\mathcal{S}$ is divided finely in the dividing step so that the size of each group is less than a constant (see Eq. (13)). In such cases, the overall time complexity of SWeG (Algorithm 1) is $O(T \cdot |\mathcal{E}|)$ since the dividing and merging steps are repeated T times. This linear scalability is shown experimentally in Section 4.4.

3.4.2 Memory Requirements. The space required for storing $\mathcal{S}$, $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(k)}\}$, $\{h(v) : v \in \mathcal{V}\}$, and $\{c_v : v \in \mathcal{V}\}$ is $O(|\mathcal{V}|)$, and the space required for storing $\mathcal{G}$, $\overline{\mathcal{G}}$, and C is $O(|\mathcal{E}|)$ (see Eq. (14)). Thus, in shared-memory settings, where all of them are stored in memory, the memory requirements of SWeG (Algorithm 1) are $O(|\mathcal{V}| + |\mathcal{E}|) = O(|\mathcal{E}|)$. In MapReduce settings, as described in Section 3.3, each mapper or reducer requires $O(|\mathcal{V}|)$ memory, which is used to store $\mathcal{S}$, $\{h(v) : v \in \mathcal{V}\}$, or $\{c_v : v \in \mathcal{V}\}$, in all the stages except for Reduce-1. In Reduce-1, in addition to $O(|\mathcal{V}|)$ memory for $\mathcal{S}$, each reducer requires $O(\max_{1 \leq i \leq k} |\mathcal{E}^{(i)}|)$ memory to load $\{R(A) : A \in \mathcal{S}^{(i)}\}$ for each group $\mathcal{S}^{(i)}$ at once. Thus, the memory requirements are $O(|\mathcal{V}| + \max_{1 \leq i \leq k} |\mathcal{E}^{(i)}|)$ per reducer. In real-world graphs, $\max_{1 \leq i \leq k} |\mathcal{E}^{(i)}|$ is much smaller than $|\mathcal{E}|$, as shown experimentally in Appendix B.

Algorithm 6 SWeG+: Algorithm for Further Compression

Input: input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, number of iterations T, error bound ϵ
graph-compression method ALG
Output: compressed $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, $\mathcal{G}^+ = (\mathcal{V}, C^+)$, and $\mathcal{G}^- = (\mathcal{V}, C^-)$
1: run SWeG to get $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and $C =< C^+, C^- >$ ▷ Algorithm 1
2: run ALG on each of $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, $\mathcal{G}^+ = (\mathcal{V}, C^+)$, and $\mathcal{G}^- = (\mathcal{V}, C^-)$
3: **return** the compressed $\overline{\mathcal{G}}$, $\mathcal{G}^+$, and $\mathcal{G}^-$

Table 2: Summary of the real-world datasets that we used.

Name	# Nodes	# Edges	Summary
Caida (CA) [27]	26,475	53,381	Internet
Protein (PR) [18]	6,229	146,160	Protein Interaction
Email (EM) [21]	36,692	183,831	Email
DBLP (DB) [56]	317,080	1,049,866	Collaboration
Amazon (AM) [26]	403,394	2,443,408	Co-purchase
Youtube (YO) [37]	1,134,890	2,987,624	Social
Skitter (SK) [27]	1,696,415	11,095,298	Internet
Web-Small (WS) [5]	862,664	16,138,468	Hyperlinks
Patent (PA) [15]	3,774,768	16,518,947	Citations
LiveJournal (LJ) [56]	3,997,962	34,681,189	Social
Hollywood (HO) [5]	1,985,306	114,492,816	Collaboration
Web-Large (WL) [5]	39,454,463	783,027,125	Hyperlinks
LinkedIn (LI)	> 600 millions	> 20 billions	Social

3.5 Further Compression: SWeG+

We propose SWeG+, an algorithm for further compression. The outputs of SWeG (i.e., $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and $C =< C^+, C^- >$) can be represented as three graphs: $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, $\mathcal{G}^+ = (\mathcal{V}, C^+)$, and $\mathcal{G}^- = (\mathcal{V}, C^-)$. As described in Algorithm 6, SWeG+ further compresses each of the graphs using a given graph-compression method ALG. Any graph-compression method, such as [2, 5, 7, 9, 11, 43], can be used as ALG depending on the objectives of compression.[1] In Section 4.6, we empirically show that for many graph-compression methods, SWeG+ gives significantly more compact representations than directly compressing the input graph using the methods.

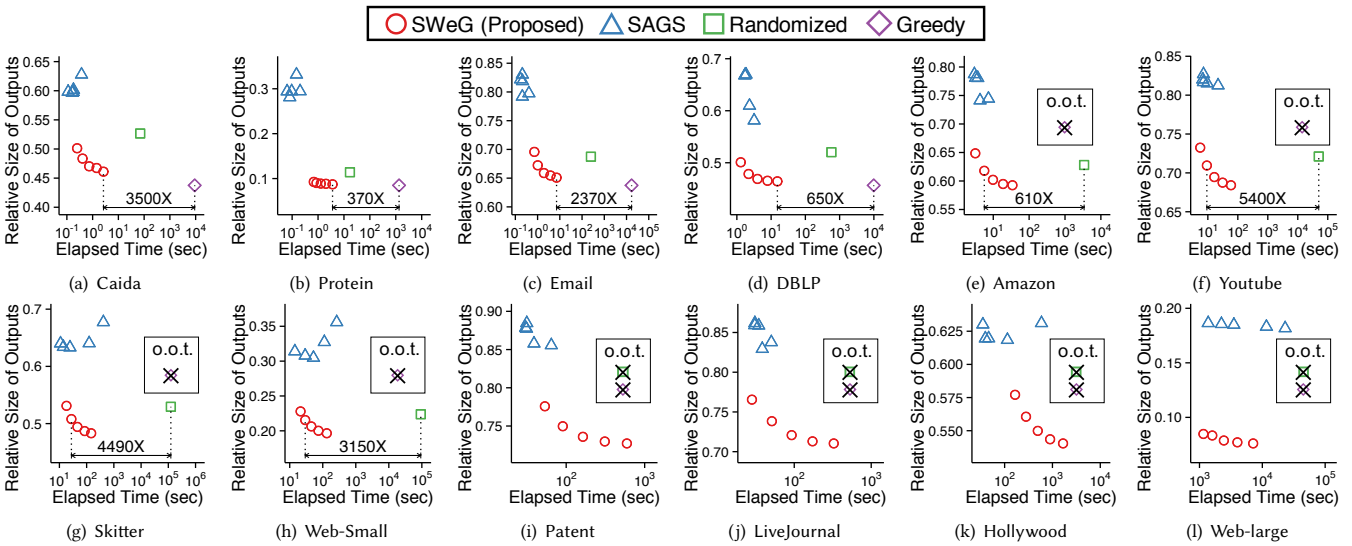


Figure 3: SWeG (lossless and shared-memory) significantly outperforms existing lossless summarization methods. o.o.t.: out of time (> 48 hours). Specifically, SWeG was up to $5,400\times$ faster than the others that give similarly compact outputs.

4 EXPERIMENTS

We review our experiments for answering the following questions:

- **Q1. Lossless Summarization**: Does the lossless version of SWeG yield more compact representations faster than its competitors?
- **Q2. Lossy Summarization**: Does the lossy version SWeG yield more compact and accurate representations than baselines?
- **Q3. Scalability**: How well does SWeG scale as the size of the input graph, the number of machines, and the number of cores grow?
- **Q4. Effects of Parameters**: How do the number of iterations T and the error bound ϵ affect the compactness of outputs?
- **Q5. Further Compression**: How much does SWeG+ improve the compression rates of combined compression methods?

4.1 Experimental Settings

Machines: We ran single instance experiments on a machine with 2.10GHz Intel Xeon E6-2620 CPUs (with 6 cores) and 64GB memory. We ran MapReduce experiments on a private Hadoop cluster.

Datasets: We used the graphs listed in Table 2. We ignored the direction of edges in all of them.

Implementations: We implemented all the considered algorithms in Java 1.8. We implemented the shared-memory version of SWeG using standard Java multithreading, and we set the number of threads to 8 unless otherwise stated. We implemented the MapReduce version of SWeG using Hadoop 2.6.1, and we set the numbers of mappers and reducers to 40 unless otherwise stated. In both implementations, we ran the dividing step recursively so that each group had at most 500 supernodes, as described in Section 3.1.2.

Evaluation Metric: Given an output representation $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and $C = < C^+, C^- >$ of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we measured its compactness using the *relative size of outputs*, defined as

$$(|\mathcal{P}^*| + |C^+| + |C^-|)/|\mathcal{E}|, \tag{15}$$

where the numerator is our objective function (i.e., Eq. (1)) and the denominator is a constant for a given input graph.

[1] If ALG relabels nodes, to keep the labels of $\mathcal{V}$ in $\mathcal{G}^+$ and $\mathcal{G}^-$ the same, the labels obtained when compressing $\mathcal{G}^+$ are used for $\mathcal{G}^-$, which tends to be smaller than $\mathcal{G}^+$.

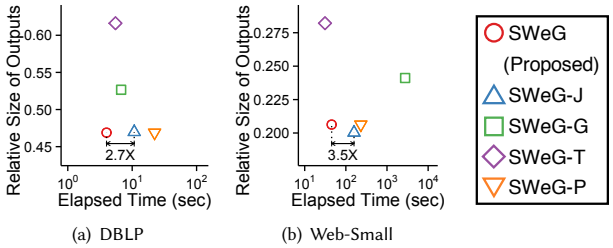


Figure 4: SWeG (lossless and shared-memory) significantly outperforms its variants. This justifies our design choices.

4.2 Q1. Lossless Summarization

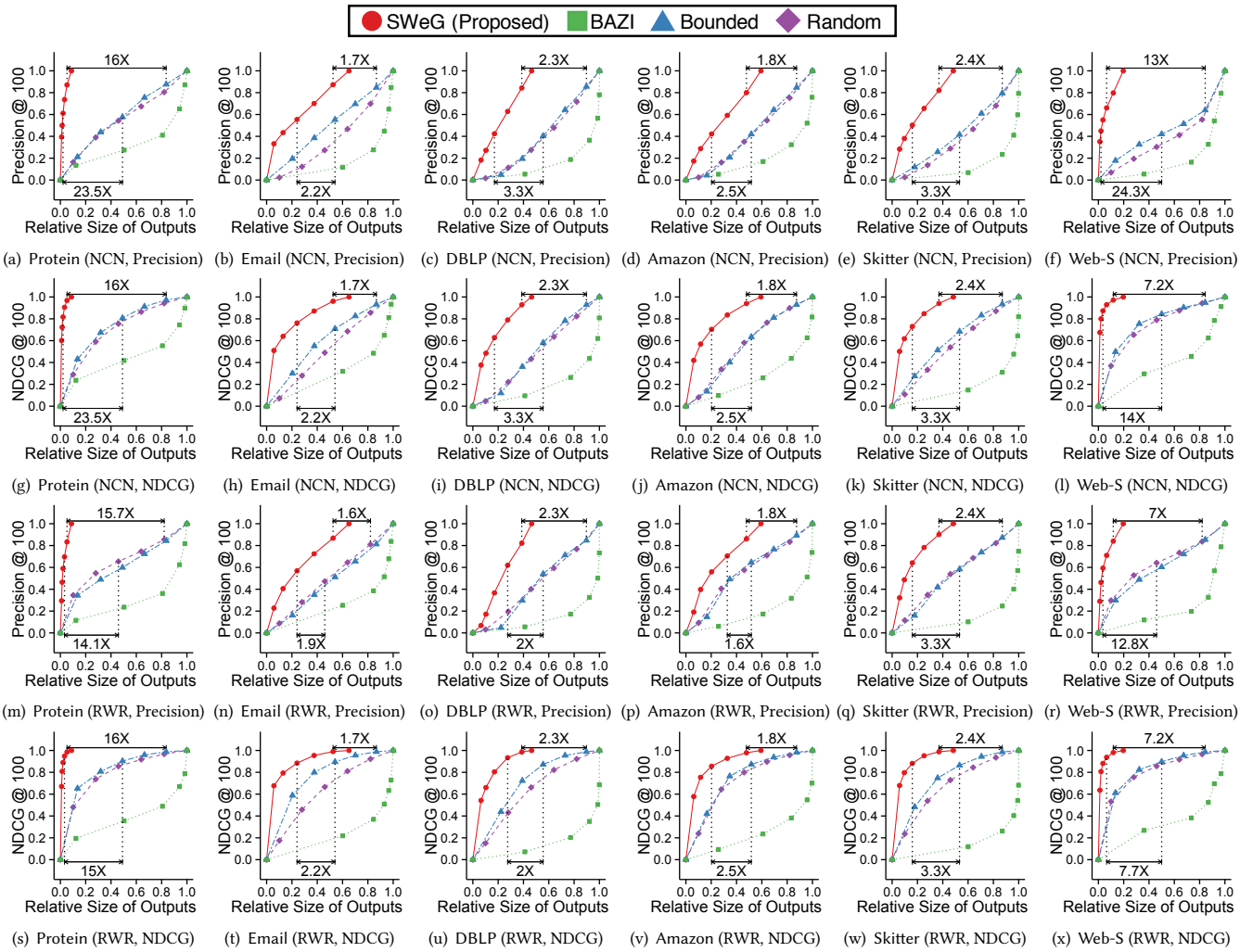
We compared the following lossless graph-summarization methods in terms of speed and compactness of representations: [2]

(a) SWeG (proposed): the shared-memory and lossless version of SWeG with $\epsilon = 0$ and $T = \{5, 10, 20, 40, 80\}$.
(b) SAGS [20]: SAGS with five different parameter settings. [3]
(c) Randomized [38].
(d) Greedy [38].
(e) SWeG-P: a variant of (a) without **p**arallelization.
(f) SWeG-G: a variant of (a) that does not **g**roup the supernodes.
(g) SWeG-J: a variant of (a) that chooses supernodes to merge based on the exact savings instead of **j**accard similarity.
(h) SWeG-T: a variant of (a) where the merging **t**hresohld $\theta(t)$ is 0.

To this end, we measured the elapsed time and the relative size of the outputs (i.e., Eq. (15)) of each algorithm.

SWeG gave the best trade-off between speed and compactness of outputs on all the datasets, as seen in Figure 3. For example, on the Youtube dataset, SWeG was **5,400× faster** with 2% smaller outputs than Randomized and 18% faster with 10% smaller outputs than SAGS. Greedy did not terminate within a reasonable

[2] We slightly modified (b)-(d) so that they aim to minimize the same objective of SWeG (i.e., Eq. (1)). We fixed T to 20 in (e)-(h). (b)-(e) are serial, and the others are parallel.
[3] $\{h = 28, b = 7, p = 0.3\}$, $\{h = 28, b = 7, p = 0.1\}$, $\{h = 28, b = 7, p = 0.2\}$, $\{h = 30, b = 10, p = 0.3\}$,$\{h = 30, b = 15, p = 0.3\}$. p denotes the overlap ratio.



(a) Protein (NCN, Precision) (b) Email (NCN, Precision) (c) DBLP (NCN, Precision) (d) Amazon (NCN, Precision) (e) Skitter (NCN, Precision) (f) Web-S (NCN, Precision)

(g) Protein (NCN, NDCG) (h) Email (NCN, NDCG) (i) DBLP (NCN, NDCG) (j) Amazon (NCN, NDCG) (k) Skitter (NCN, NDCG) (l) Web-S (NCN, NDCG)

(m) Protein (RWR, Precision) (n) Email (RWR, Precision) (o) DBLP (RWR, Precision) (p) Amazon (RWR, Precision) (q) Skitter (RWR, Precision) (r) Web-S (RWR, Precision)

(s) Protein (RWR, NDCG) (t) Email (RWR, NDCG) (u) DBLP (RWR, NDCG) (v) Amazon (RWR, NDCG) (w) Skitter (RWR, NDCG) (x) Web-S (RWR, NDCG)

Figure 5: SWeG (lossy) significantly outperforms baseline methods for lossy graph summarization. Specifically, SWeG yielded up to 24.3× more compact and similarly accurate representations than the other methods.

time (48 hours). In Figure 4, we show that SWeG (with $T = 20$) also outperformed its variants, justifying our design choices in Section 3.

4.3 Q2. Lossy Summarization

We compared the following lossy graph-summarization methods in terms of the compactness and accuracy of output representations:

(a) SWeG (proposed): the shared-memory and lossy version of SWeG with $T = 80$ and $\epsilon = \{0, 0.18, 0.36, 0.54, 0.72, 0.9\}$.
(b) BAZI [3]: BAZI with $s = \log^2 |\mathcal{S}|$, $w = 50$, and $k = \{0.1 \cdot |\mathcal{V}|, 0.28 \cdot |\mathcal{V}|, 0.46 \cdot |\mathcal{V}|, 0.64 \cdot |\mathcal{V}|, 0.82 \cdot |\mathcal{V}|, |\mathcal{V}|\}$.
(c) Bounded: a baseline that performs only the dropping step of SWeG, i.e., SWeG with $T = 0$ and $\epsilon = \{0.18, 0.36, 0.54, 0.72, 0.9\}$.
(d) Random: a baseline that randomly drops $\epsilon = \{0.18, 0.36, 0.54, 0.72, 0.9\}$ of the edges from the input graph.

To this end, for each method, we measured the relative size of outputs (e.g., Eq. (15)) and measured how accurately the outputs preserve the relevances between nodes as follows:

S1. randomly choose 100,000 seed nodes in the input graph.
S2. compute the **true relevances** between each seed node and the other nodes in the input graph.
S3. compute the **approximate relevances** between each seed node and the other nodes in the graph restored from the outputs.
S4. measure how accurate the approximate relevances from S3 are.

In **S2** and **S3**, we used one of the following relevance scores:

- *Random Walk with Restart (RWR)* [53]: each node's RWR score with respect to a seed node is defined as the stationary probability that a random surfer is at the node. The random surfer either moves to a neighboring node of the current node (with probability 0.8) or restarts at the seed node (with probability 0.2).
- *Number of Common Neighbors (NCN)*: each node's NCN score with respect to a seed node is defined as the number of common neighbors of the node and the seed node.

In **S4**, we computed one of the following accuracy measures for every seed node and then averaged them.

- *Precision@100*: Precision@100 is defined as the fraction of the 100 most relevant nodes in terms of the true relevances among those in terms of the approximate relevances.

- *NDCG@*100 [17]: Let $r(i)$ be the true relevance of the i-th most relevant node in terms of the approximate relevances. Then,

$$\text{NDCG@100} := \frac{1}{Z}\sum_{i=1}^{100}\frac{2^{r(i)}}{\log_2(1+i)},$$

where Z is a constant normalizing NDCG@100 to be within $[0, 1]$.

SWeG yielded the most compact and accurate representations on all the considered datasets, as seen in Figure 5. For example, on the Web-Small dataset, SWeG gave a **24.3× more compact** and similarly accurate representation than the other methods. BAZI tended to yield representations with most (super) edges since it aims to reduce the number of supernodes instead of minimizing the number of (super) edges (see Section 5).

4.4 Q3. Scalability

We evaluated the scalability of the shared-memory and MapReduce implementations of SWeG. Specifically, we measured how rapidly their running times change depending on the size of the input graph, the number of threads, and the number of machines (i.e., the numbers of mappers and reducers in the MapReduce framework). In the shared-memory setting, we used graphs with different sizes obtained by sampling different numbers of nodes from the Web-Large dataset. In the MapReduce setting, we used graphs obtained in the same manner using the LinkedIn dataset. When measuring the data scalability, we fixed the number of threads to 8 and the number of machines to 40. When measuring the machine and multi-core scalability, we fixed the size of the input graph.

SWeG scaled linearly with the size of the input graph, as seen in Figures 6(a) and 6(b). The lossless summarization by SWeG as well as the additional dropping step for lossy summarization (with $\epsilon = 0.1$) scaled near linearly with the number of edges in the input graph in both settings. Note that the largest graph used had more than **20 billion edges**.

SWeG achieved significant speedup in the shared-memory and MapReduce settings. As seen in Figure 6(c), the speedup, defined as T_1/T_N where T_N is the running time of SWeG with N threads, increased near linearly with the number of threads in the shared-memory setting. Specifically, SWeG provided a speedup of 3.3 with 4 threads and 5.7 with 8 threads. As seen in Figure 6(d), the speedup of SWeG, defined as T_1/T_N where T_N is the running time of SWeG with N machines, increased near linearly with the number of machines in the MapReduce setting. Specifically, SWeG provided a speedup of 8.3 with 10 machines and 26.4 with 40 machines.

4.5 Q4. Effects of Parameters

We measured how the number of iterations T and the error bound ϵ in SWeG affect the compactness of its output representation using the relative size of outputs (i.e., Eq. (15)). When measuring the effect of T, we fixed ϵ to 0 and changed T from 1 to 80. When measuring the effect of ϵ, we fixed T to 80 and changed ϵ from 0 to 0.5.

The larger the number of iterations, the more compact the output representation. As seen in Figure 7, the size of outputs decreased over iterations and eventually plateaued. **The larger the error bound, the more compact the output representation**. As seen in Figure 8, the size of outputs decreased near linearly as the error bound increased. The relative size of outputs was especially small in web graphs and protein-interaction graphs, where nodes tend to have similar connectivity [10, 23].

4.6 Q5. Further Compression

We measured how much SWeG+ improves the compression rates of the following advanced graph-compression methods:

(a) BP [11]: reordering nodes as suggested in [11] with 20 iterations and using the webgraph framework [5].
(b) Shingle [9]: reordering nodes as suggested in [9] and using the webgraph framework [5].
(c) BFS [2]: running BFS with the default parameter setting at https://github.com/drovandi/GraphCompressionByBFS.
(d) VNMiner [7]: running VNMiner with 80 iterations.

For the webgraph framework in (a)-(b), we used the default parameter setting (i.e., $r = 3$, $W = 7$, $L_{min} = 7$, and ζ_3). We measured the compression rates using the objective function of each compression method. That is, we used the number of bits per directed edge[4] for (a)-(c) and the relative size of outputs (i.e., Eq. (15)) for (d).

SWeG+ achieved significant further compression. As seen in Figures 1(d) and 9, the lossless version of SWeG+ with $T = 80$ and $\epsilon = 0$ yielded up to **3.4× more compact** representations than all the input compression methods on all the datasets. Especially, SWeG+ with ALG = BFS represented the Web-Large dataset using less than **0.7 bits per directed edge** without loss of information.

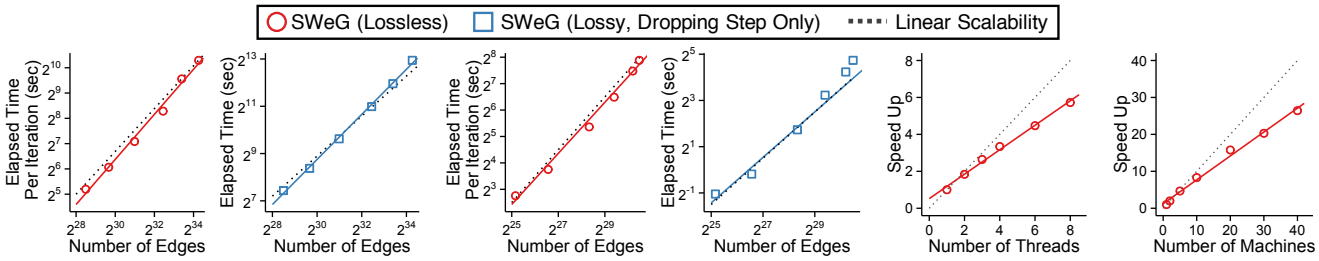
5 RELATED WORK

While this paper addresses Problem 1, the term "graph summarization" has been used for a wider range of problems related to concisely describing static plain graphs [12, 22, 28, 33–35, 38, 39, 42, 46, 52, 58], static attributed graphs [8, 13, 16, 19, 41, 47, 49, 51, 55, 57], and dynamic graphs [1, 29, 40, 45, 50]. We refer the reader to an excellent survey [32] for a review on these problems. In this section, we focus on previous work closely related to Problem 1.

Lossless Graph Summarization. Greedy [38] repeatedly finds and merges a pair of supernodes so that savings in space are maximized given the other supernodes. If there is no pair whose merger leads to non-negative savings, then Greedy creates a summary graph and corrections so that the number of (super) edges is minimized given the current supernodes. Greedy is computationally and memory expensive because it maintains and updates savings for $O(|\mathcal{V}|^2)$ pairs of supernodes. Without maintaining any precomputed savings, Randomized [38], whose time complexity is $O(|\mathcal{V}| \cdot |\mathcal{E}|)$, randomly chooses a supernode first and then chooses another supernode to be merged so that savings in space are maximized given the other supernodes. SAGS [20] chooses supernodes to be merged using locality sensitive hashing without computing savings in space, which are expensive to compute. We empirically show in Section 4.2 that all these serial algorithms are not satisfactory in terms of speed or compactness of outputs. They either have high computational complexity or significantly sacrifice compactness of outputs for lower complexity. More importantly, they cannot handle large-scale graphs that do not fit in main memory.

Lossy Graph Summarization. Apxmdl [38] reduces the problem of choosing edges to be dropped from outputs of Greedy to a maximum b-matching problem and uses GabowâĂŹs algorithm [14], whose time complexity is $O(\min(|\mathcal{E}|^2 \log|\mathcal{V}|, |\mathcal{E}| \cdot |\mathcal{V}|^2))$. Due to this high computational complexity, Apxmdl does not scale

[4]We regarded undirected graphs as symmetric directed graphs.



(a) Data Scalability (MapReduce) (b) Data Scalability (Shared Memory) (c) Multi-Core Scalability (Shared Memory) (d) Machine Scalability (MapReduce)

Figure 6: SWeG is scalable. (a-b) SWeG scaled linearly with the size of the input graph. (c-d) SWeG achieved significant speedup as more machines and CPU cores were used. Note that the largest graph used had more than 20 billion edges.

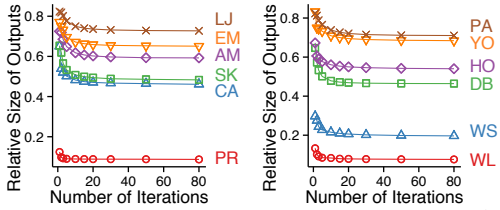


Figure 7: As the number of iterations in SWeG (lossless) increases, the output representations become compact.

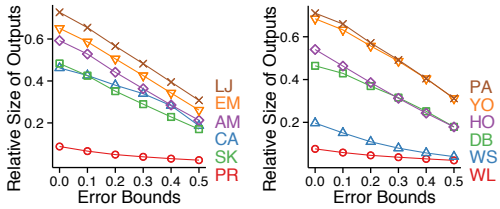


Figure 8: As the error bound in SWeG (lossy) increases, the output representations become compact.

even to moderately-sized graphs. For the same problem, the dropping step of SWeG takes $O(|\mathcal{E}|)$ time (see Section 3.4.1) without sacrificing the compactness of outputs (see Appendix C). In [38], combining Greedy and Apxmdl into a single step was also discussed. Several algorithms [3, 25, 31, 42] have been developed for a problem that is similar but not identical to Problem 1. They aim to find a weighted summary graph with a given number of supernodes so that the difference between the original and restored graphs is minimized without edge corrections. The way of restoring a graph is also different from that in Problem 1. Since these algorithms aim to reduce the number of supernodes, instead of (super) edges, they are not effective for Problem 1 (see Section 4.3).

Combination with Other Compression Techniques. In addition to graph summarization, numerous graph-compression techniques have been developed, including relabeling nodes [2, 5, 9, 11], utilizing encoding schemes for integer sequences (e.g., reference, gap, and interval encodings) [5], and encoding common structures (e.g., cliques, bipartite-cores, and stars) with fewer bits [7, 22, 43]. See [4] for a comprehensive survey on these techniques. As described in Section 3.5 and Section 4.6, SWeG is readily combinable with any compression technique for static plain graphs. In [44], tightly combining two specific summarization and compression algorithms [38, 54] into a single process was presented.

Algorithm 7 Neighbor Query Processing on $\overline{\mathcal{G}}$ and C

Input: summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$, corrections $C =< C^+, C^- >$, query node $v \in \mathcal{V}$
Output: the set $\hat{N}_v$ of v's neighbors in the restored graph $\hat{\mathcal{G}}$

```
1: N̂_v ← ∅; S_v ← the supernode in S where v ∈ S_v
2: if S_v has a self-loop in Ḡ then N̂_v ← N̂_v ∪ (S_v − {v})
3: for each neighbor A (≠ S_v) of S_v in Ḡ do N̂_v ← N̂_v ∪ A
4: N̂_v ← (N̂_v ∪ N_v^+) − N_v^−        ▷ N_v^+: v's neighbors in C^+
5: return N̂_v                          ▷ N_v^− : v's neighbors in C^−
```

6 CONCLUSIONS

We propose SWeG, a fast parallel algorithm for lossless and lossy summarization of large-scale graphs, which may not fit in main memory. We present efficient implementations of SWeG in shared-memory and MapReduce environments. We also propose SWeG+ where SWeG and other graph-compression methods are combined to achieve better compression than individual methods. We theoretically and empirically show the following strengths of SWeG:

- **Speed**: SWeG provides similarly compact representations up to *5, 400× faster* than existing summarization methods (Figure 3).
- **Scalability**: SWeG scales near linearly with the size of the input graph, the number of machines, and the number of CPU cores. It successfully scales to graphs with over *20 billion* edges (Figure 6).
- **Compression**: SWeG+ achieves up to *3.4× better* compression than individual state-of-the-art graph-compression methods that are combined with SWeG (Figures 1(d) and 9).

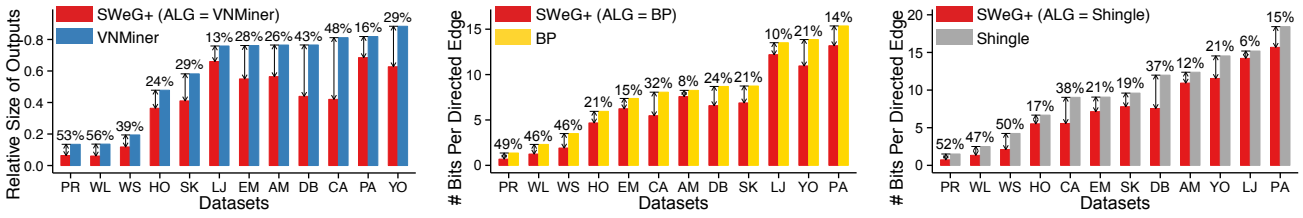
A APPENDIX: NEIGHBOR QUERIES

Algorithm 7 describes how to answer neighbor queries (i.e., finding the neighbors $\hat{N}_v$ of a given node $v \in \mathcal{V}$) efficiently on a summary graph $\overline{\mathcal{G}} = (\mathcal{S}, \mathcal{P})$ and corrections $C =< C^+, C^- >$ without restoring entire $\hat{\mathcal{G}}$. Let N_v^- be the neighbors of node $v \in \mathcal{V}$ in C^-. If there is no redundant edge[5] in C^+ and we use a hash table for $\hat{N}_v$ and adjacency lists for C^+, C^-, and $\mathcal{P}$, then the running time of Algorithm 7 is proportional to $|\hat{N}_v| + 2|N_v^-|$. In every dataset listed in Table 2 in Section 4, when SWeG ($T = 80$) was used, $|C^-|$ was at most 6% of the number of edges in the input graph, regardless of the error bound ϵ. Thus, since $|\hat{N}_v| \leq (1+\epsilon) \cdot |N_v|$ from Eq. (2), $|\hat{N}_v| + 2|N_v^-|$ was at most $(1.12 + \epsilon) \cdot |N_v|$ on average.

B APPENDIX: SIZE OF SUBGRAPHS

We measured the size of the largest subgraphs that need to be loaded in main memory at once in our MapReduce implementation of SWeG (with $T = 80$). As seen in Figure 10, the largest subgraphs had

[5] As in the outputs of SWeG, if $\{A, B\} \in \mathcal{P}$, $u \in A$, and $v \in B$, then $\{u, v\} \notin C^+$.



(a) SWeG+ (lossless) vs. VNMiner (b) SWeG+ (lossless) vs. BP (c) SWeG+ (lossless) vs. Shingle

Figure 9: SWeG+ (lossless) significantly improves the compression rates of state-of-the-art graph-compression algorithms.

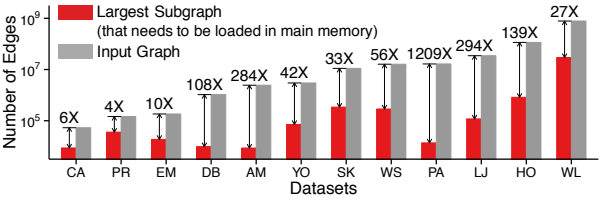


Figure 10: Subgraphs that SWeG (MapReduce) loads in memory at once are significantly smaller than input graphs.

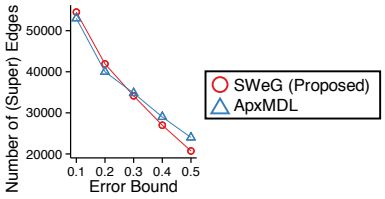


Figure 11: While SWeG has much lower time complexity than Apxmdl, they give similarly compact results.

up to **1,209× fewer** edges than the entire graphs. The experimental settings were the same with those in Section 4.1.

C APPENDIX: COMPARISON WITH APXMDL

To compare SWeG and Apxmdl [38] in terms of the compactness of their output representations, we measured how the number of (super) edges in their outputs, which Apxmdl aims to minimize, changes depending on the error bound ϵ. We used the same small-scale graph used in [38], which is a web graph with 40,000 nodes [5]. SWeG (with $T = 80$) and Apxmdl yielded similarly compact outputs, as seen in Figure 11, where we compare with the numbers reported in [38]. Specifically, SWeG gave more compact outputs than Apxmdl when ϵ was large, while the opposite happened when ϵ was small. SWeG has much lower time complexity than Apxmdl. Specifically, the dropping step of Apxmdl takes $O(\min(|\mathcal{E}|^2 \log |\mathcal{V}|, |\mathcal{E}| \cdot |\mathcal{V}|^2))$ time, while that of SWeG takes $O(|\mathcal{E}|)$ time (see Section 3.4.1).

REFERENCES

[1] Bijaya Adhikari, Yao Zhang, Aditya Bharadwaj, and B Aditya Prakash. 2017. Condensing temporal networks using propagation. In *SDM*.
[2] Alberto Apostolico and Guido Drovandi. 2009. Graph compression by BFS. *Algorithms* 2, 3 (2009), 1031–1044.
[3] Maham Anwar Beg, Muhammad Ahmad, Arif Zaman, and Imdadullah Khan. 2018. Scalable Approximation Algorithm for Graph Summarization. In *PAKDD*.
[4] Maciej Besta and Torsten Hoefler. 2018. Survey and Taxonomy of Lossless Graph Compression and Space-Efficient Graph Representations. *arXiv preprint arXiv:1806.01799* (2018).
[5] Paolo Boldi and Sebastiano Vigna. 2004. The webgraph framework I: compression techniques. In *WWW*.
[6] Andrei Z Broder, Moses Charikar, Alan M Frieze, and Michael Mitzenmacher. 2000. Min-wise independent permutations. *J. Comput. System Sci.* 60, 3 (2000), 630–659.
[7] Gregory Buehrer and Kumar Chellapilla. 2008. A scalable pattern mining approach to web graph compression with communities. In *WSDM*.
[8] Chen Chen, Cindy X Lin, Matt Fredrikson, Mihai Christodorescu, Xifeng Yan, and Jiawei Han. 2009. Mining graph patterns efficiently via randomized summaries. *PVLDB* 2, 1 (2009), 742–753.
[9] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, Michael Mitzenmacher, Alessandro Panconesi, and Prabhakar Raghavan. 2009. On compressing social networks. In *KDD*.
[10] Fan Chung, Linyuan Lu, T Gregory Dewey, and David J Galas. 2003. Duplication models for biological networks. *Journal of Computational Biology* 10, 5 (2003), 677–687.
[11] Laxman Dhulipala, Igor Kabiljo, Brian Karrer, Giuseppe Ottaviano, Sergey Pupyrev, and Alon Shalita. 2016. Compressing graphs and indexes with recursive graph bisection. In *KDD*.
[12] Cody Dunne and Ben Shneiderman. 2013. Motif simplification: improving network visualization readability with fan, connector, and clique glyphs. In *CHI*.
[13] Wenfei Fan, Xin Wang, and Yinghui Wu. 2013. Diversified top-k graph pattern matching. *PVLDB* 6, 13 (2013), 1510–1521.
[14] Harold N. Gabow. 1983. An Efficient Reduction Technique for Degree-constrained Subgraph and Bidirected Network Flow Problems. In *STOC*.
[15] Bronwyn H Hall, Adam B Jaffe, and Manuel Trajtenberg. 2001. *The NBER patent citation data file: Lessons, insights and methodological tools.* Technical Report. National Bureau of Economic Research.
[16] Nasrin Hassanlou, Maryam Shoaran, and Alex Thomo. 2013. Probabilistic graph summarization. In *WAIM*.
[17] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *TOIS* 20, 4 (2002), 422–446.
[18] G Joshi-Tope, Marc Gillespie, Imre Vastrik, Peter D'Eustachio, Esther Schmidt, Bernard de Bono, Bijay Jassal, GR Gopinath, GR Wu, Lisa Matthews, et al. 2005. Reactome: a knowledgebase of biological pathways. *Nucleic Acids Research* 33, suppl_1 (2005), D428–D432.
[19] Kifayat Ullah Khan, Waqas Nawaz, and Young-Koo Lee. 2014. Set-based unified approach for attributed graph summarization. In *BdCloud*.
[20] Kifayat Ullah Khan, Waqas Nawaz, and Young-Koo Lee. 2015. Set-based approximate approach for lossless graph summarization. *Computing* 97, 12 (2015), 1185–1207.
[21] Bryan Klimt and Yiming Yang. 2004. The enron corpus: A new dataset for email classification research. In *ECML*.
[22] Danai Koutra, U Kang, Jilles Vreeken, and Christos Faloutsos. 2014. VOG: Summarizing and Understanding Large Graphs. In *SDM*.
[23] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D Sivakumar, Andrew Tomkins, and Eli Upfal. 2000. Stochastic models for the web graph. In *FOCS*.
[24] Douglas Lea. 2000. *Concurrent programming in Java: design principles and patterns.* Addison-Wesley Professional.
[25] Kristen LeFevre and Evimaria Terzi. 2010. GraSS: Graph structure summarization. In *SDM*.
[26] Jure Leskovec, Lada A Adamic, and Bernardo A Huberman. 2007. The dynamics of viral marketing. *TWEB* 1, 1 (2007), 5.
[27] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *TKDD* 1, 1 (2007), 2.
[28] Cheng-Te Li and Shou-De Lin. 2009. Egocentric information abstraction for heterogeneous social networks. In *ASONAM*.
[29] Yu-Ru Lin, Hari Sundaram, and Aisling Kelliher. 2008. Summarization of social activity over time: people, actions and concepts in dynamic networks. In *CIKM*.
[30] Chao Liu, Fan Guo, and Christos Faloutsos. 2009. Bbm: bayesian browsing model from petabyte-scale data. In *KDD*.
[31] Xingjie Liu, Yuanyuan Tian, Qi He, Wang-Chien Lee, and John McPherson. 2014. Distributed graph summarization. In *CIKM*.
[32] Yike Liu, Tara Safavi, Abhilash Dighe, and Danai Koutra. 2018. Graph Summarization Methods and Applications: A Survey. *CSUR* 51, 3 (2018), 62.
[33] Antonio Maccioni and Daniel J Abadi. 2016. Scalable pattern matching over compressed graphs via dedensification. In *KDD*.
[34] Michael Mathioudakis, Francesco Bonchi, Carlos Castillo, Aristides Gionis, and Antti Ukkonen. 2011. Sparsification of influence networks. In *KDD*.

[35] Yasir Mehmood, Nicola Barbieri, Francesco Bonchi, and Antti Ukkonen. 2013. Csi: Community-level social influence analysis. In *ECML/PKDD*.

[36] Robert Meusel, Sebastiano Vigna, Oliver Lehmberg, and Christian Bizer. 2014. Graph structure in the web—revisited: a trick of the heavy tail. In *WWW*.

[37] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. 2007. Measurement and analysis of online social networks. In *IMC*.

[38] Saket Navlakha, Rajeev Rastogi, and Nisheeth Shrivastava. 2008. Graph summarization with bounded error. In *SIGMOD*.

[39] Manish Purohit, B Aditya Prakash, Chanhyun Kang, Yao Zhang, and VS Subrahmanian. 2014. Fast influence-based coarsening for large networks. In *KDD*.

[40] Qiang Qu, Siyuan Liu, Christian S Jensen, Feida Zhu, and Christos Faloutsos. 2014. Interestingness-driven diffusion process summarization in dynamic networks. In *ECML/PKDD*.

[41] Sriram Raghavan and Hector Garcia-Molina. 2003. Representing web graphs. In *ICDE*.

[42] Matteo Riondato, David García-Soriano, and Francesco Bonchi. 2017. Graph summarization with quality guarantees. *Data Mining and Knowledge Discovery* 31, 2 (2017), 314–349.

[43] Ryan A Rossi and Rong Zhou. 2018. GraphZIP: a clique-based sparse graph compression method. *Journal of Big Data* 5, 1 (2018), 10.

[44] Hojin Seo, Kisung Park, Yongkoo Han, Hyunwook Kim, Muhammad Umair, Kifayat Ullah Khan, and Young-Koo Lee. 2018. An effective graph summarization and compression technique for a large-scaled graph. *The Journal of Supercomputing* (2018), 1–15.

[45] Neil Shah, Danai Koutra, Tianmin Zou, Brian Gallagher, and Christos Faloutsos. 2015. Timecrunch: Interpretable dynamic graph summarization. In *KDD*.

[46] Zeqian Shen, Kwan-Liu Ma, and Tina Eliassi-Rad. 2006. Visual analysis of large heterogeneous social networks by semantic and structural abstraction. *TVCG* 12, 6 (2006), 1427–1439.

[47] Maryam Shoaran, Alex Thomo, and Jens H Weber-Jahnke. 2013. Zero-knowledge private graph summarization.. In *Big Data*.

[48] Julian Shun, Laxman Dhulipala, and Guy E Blelloch. 2015. Smaller and faster: Parallel processing of compressed graphs with Ligra+. In *DCC*.

[49] Qi Song, Yinghui Wu, Peng Lin, Luna Xin Dong, and Hui Sun. 2018. Mining summaries for knowledge graph search. *TKDE* 30, 10 (2018), 1887–1900.

[50] Nan Tang, Qing Chen, and Prasenjit Mitra. 2016. Graph stream summarization: From big bang to big crunch. In *SIGMOD*.

[51] Yuanyuan Tian, Richard A Hankins, and Jignesh M Patel. 2008. Efficient aggregation for graph summarization. In *SIGMOD*.

[52] Hannu Toivonen, Fang Zhou, Aleksi Hartikainen, and Atte Hinkka. 2011. Compression of weighted graphs. In *KDD*.

[53] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast random walk with restart and its applications. In *ICDM*.

[54] Kesheng Wu, Ekow J Otoo, and Arie Shoshani. 2006. Optimizing bitmap indices with efficient compression. *TODS* 31, 1 (2006), 1–38.

[55] Ye Wu, Zhinong Zhong, Wei Xiong, and Ning Jing. 2014. Graph summarization for attributed graphs. In *ISEEE*.

[56] Jaewon Yang and Jure Leskovec. 2015. Defining and evaluating network communities based on ground-truth. *KAIS* 42, 1 (2015), 181–213.

[57] Ning Zhang, Yuanyuan Tian, and Jignesh M Patel. 2010. Discovery-driven graph summarization. In *ICDE*.

[58] Linhong Zhu, Majid Ghasemi-Gol, Pedro Szekely, Aram Galstyan, and Craig A Knoblock. 2016. Unsupervised entity resolution on multi-type graphs. In *ISWC*.

Generative Graph Models based on Laplacian Spectra*

Alana Shine
University of Southern California

David Kempe
University of Southern California

ABSTRACT

We present techniques for generating random graphs whose Laplacian spectrum approximately matches that of a given input graph. The motivation for matching the Laplacian spectrum is that it naturally encodes high-level connectivity information about the input graph; most existing models (e.g., variants of the Configuration Model, Stochastic Block Model, or Kronecker Graphs) focus on local structure or limited high-level partitions.

Our techniques succeed in matching the spectrum of the input graph more closely than the benchmark models. We also evaluate our generative model using other global and local properties, including shortest path distances, betweenness centrality, degree distribution, and clustering coefficients. The graphs produced by our model almost always match the input graph better than those produced by the benchmark models with respect to shortest path distance and clustering coefficient distributions. The performance on betweenness centrality is comparable to the benchmarks, while a worse match on the degree distribution is a price our method pays for more global similarity.

Our results suggest that focusing on spectral properties may lead to good performance for other global properties, at a modest loss in local similarity. Since global connectivity patterns are usually more important than local features for processes such as information flow, spread of epidemics, routing, etc., our main goal is to advocate for a shift in focus from graph generative models matching local properties to those matching global connectivity patterns.

CCS CONCEPTS

• **Mathematics of computing → Spectra of graphs**; Probabilistic algorithms; • **Theory of computation → Generating random combinatorial structures**; *Linear programming*.

KEYWORDS

Random graphs; Spectra of graphs; Generative model

ACM Reference Format:
Alana Shine and David Kempe. 2019. Generative Graph Models based on Laplacian Spectra. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313631

*Our implementation is online at https://github.com/alanadakotashine/spectral_generation.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313631

1 INTRODUCTION

Random graph models for generating "realistic-looking" graphs play a number of important roles in network science and network data mining. Their uses include:

- Produce graphs to evaluate the computational performance of algorithms or protocols (see, e.g., [32]).
- Produce graphs to evaluate the correctness or quality of the *output* of algorithms in a controlled setting.
- Formally prove that certain graphs are likely to have certain properties, and derive (descriptive or prescriptive) insights about real-world networks from these proofs.

In all cases, the insights gained from the model can only be useful if the generated graphs resemble the class of real-world networks about which one is aiming to make statements [46]. More specifically, this resemblance must apply to graph parameters that are *relevant* to the kinds of structures or processes that the algorithm depends on, or the theorem reasons about.

To illustrate this issue, consider the following example. A researcher seeks insight into the spread of epidemics in social networks, many of which he has observed to follow particular degree distributions. He proves (or observes from simulations) that when a graph is drawn uniformly at random subject to this degree distribution, with high probability, an epidemic will spread very fast on it. He concludes that social networks are very susceptible to epidemics.

The concern with this inference is that social networks may have additional properties beyond the observed degree distribution; these properties may be exceedingly rare among the sampled graphs. For instance, it is well known (see, e.g., [8, 15]) that even subject to most fixed degree distributions, almost every graph has high expansion; in turn, high expansion facilitates epidemic spread under many natural models [6, 26]. On the other hand, high expansion is also nearly synonymous with a lack of pronounced community structure, whereas social networks typically do have pronounced community structure. As a result, the observation that epidemics spread fast on a random graph subject to a given degree distribution is likely more due to the expansion of random graphs than the given degree distribution.

With some exceptions (e.g., [52, 53, 55]) discussed below, most[1] generative graph models draw a graph (nearly) uniformly from an ensemble, subject to local properties such as (expected) degrees, joint degrees, or triangle counts, and/or a global partition into communities [5, 8, 14, 15, 27, 29, 30, 34, 41, 43, 44, 46, 49]. Going beyond those basic models, exponential random graph models (ERGMs) [24, 47] draw graphs from a distribution that rewards or penalizes user-specified local features; the Kronecker Graph Model [36] allows recursive specification of "community" structure in a generalization of stochastic block models (SBM). Another line of work

[1] An orthogonal class of models prescribe a generative (growing or rewiring) process based upon a plausible real-world dynamic [4, 37, 50].

uses deep neural networks (DNNs) to generate graphs. The hope is that using DNNs, one can avoid explicitly specifying the (local or global) properties one desires to replicate, instead learning them automatically from a training data set of enough "similar" graphs [17, 39, 54, 56]. An exception is the NetGAN approach [7]. NetGAN trains a deep network from a single input graph, with the goal of reproducing "realistic-looking" random walks, i.e., sequences of node visits. From these sequences, NetGAN then extracts empirical edge frequencies, and uses these to generate random graphs resembling the input graph.

While most constraints imposed by such models are typically[2] *local* in nature, most network properties and dynamics of interest to researchers are *global*. These include the spread of epidemics or information, the prevalence or strength of communities in graphs, the distribution of distances between individuals, or the capacity for information flow, among many others. These graph properties are all closely affected by the graph's expansion; hence, if the random graph model overwhelmingly generates expander graphs, the effects due to other parameters may be overshadowed.

The main high-level point of this paper is the following: *If one is interested in global network properties, then a generative model should try to match these global properties with those of the real-world networks the model is trying to mimic.*

1.1 Approximately Matching Graph Spectra

Our approach to "matching global structure" focuses on the *spectrum* (eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$) of the (normalized) graph Laplacian of the input graph. (Definitions of all key concepts are given in Section 2.) The spectrum of the graph Laplacian is known to capture connectivity properties [21, 23]. Best known is Cheeger's Inequality [13], which relates the spectral gap λ_2 with the graph's conductance. More intricate results [13, 35] show that small values of $\lambda_3, \lambda_4, \ldots$ indicate additional node sets that are sparsely connected to each other.

Concretely, given an (undirected) input graph G^* with spectrum $0 = \lambda_1^* \leq \lambda_2^* \leq \cdots \leq \lambda_n^*$, our goal is to generate random graphs G with spectrum $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ such that $\lambda_i \approx \lambda_i^*$ for all i. The intuition is that a similar spectrum should result in similar global connectivity properties.[3]

We propose intricate heuristics for randomly generating such graphs, and evaluate the heuristics extensively. The central viewpoint exploited throughout is that a (symmetric) matrix is naturally characterized by its spectrum and orthonormal eigenbasis. Since the spectrum is given, finding a graph Laplacian reduces to finding a suitable orthonormal basis. The problem is that for most bases, the resulting "graph adjacency matrix" A will have entries far from the set $\{0, 1\}$. Our approach is to produce the actual adjacency matrix gradually, using the following steps (detailed in Section 3):

(1) Sample a random orthonormal basis to construct a candidate Laplacian matrix with the desired spectrum. Use linear programming to find a matrix M whose Laplacian is approximately the candidate matrix.
(2) Keeping the spectrum fixed, perform a walk on possible orthonormal bases, guided by an objective function that pushes the entries of M close to 0 or 1.
(3) Use an LP-based algorithm to minimally perturb the entries of M to end up inside $[0, 1]$, while maintaining the row sums. This may perturb the spectrum, but the LP's objective is designed to keep the perturbation small.
(4) Use a combination of deterministic rounding across eigencuts with small eigenvalues and randomized rounding of other edges to round M to a graph G.

In Section 7, we evaluate this algorithm on multiple real-world networks. We generate samples using our approach, and compare them to the graphs generated according to the configuration model, stochastic block model, and Kronecker graph model. The comparison is based on the following global and local graph parameters: the spectrum itself, betweenness centralities, clustering coefficients, degree distribution, and the distribution of shortest path lengths.

We observe (not surprisingly) that the spectrum is in fact matched significantly better by our approach. For the other metrics, the spectral approach typically performs comparably (sometimes slightly better, sometimes slightly worse) to the best of the other approaches; for shortest path lengths, it outperforms the other models significantly most of the time.

Our work raises many concrete questions and big-picture directions for future work, discussed in more detail in Section 8. Perhaps most immediately, our algorithms currently scale to graphs of a few thousand nodes, but not beyond. In order to apply them to realistic networks, it is important to speed up the generation by several orders of magnitude. We therefore view this work not as a complete or final solution to the problem, but as a first step. Our goal in this work was to show that (1) it is computationally feasible to generate graphs matching a given graph's spectrum fairly well, and (2) doing so also helps in matching other important global graph properties. We expect future work to yield faster heuristics and different approaches which will allow these ideas to be applied to even more practical instances. Beyond the running time of the algorithm, there are also many natural questions for future work regarding provable guarantees for the objectives we pursue. Our algorithms do not come with provable approximation guarantees, and obtaining such guarantees would be desirable — though it also appears very challenging.

1.2 Related Work

Our goal is directly related to *inverse eigenvalue problems*: constructing matrices with a given spectrum, subject to other structural properties [12]. Ours is an *inexact* inverse eigenvalue problem, because the spectrum only needs to be matched approximately. On the other hand, our task is to generate a *diverse* collection of matrices, making a deterministic construction inadequate.

[2] The SBM prescribes a high-level partition, but typically only into a constant number of partitions; the recursive structure prescribed by the Kronecker model is more intricate, but depends only on a small number of parameters.

[3] It may appear natural to aim to match the eigenvalues exactly: $\lambda_i = \lambda_i^*$ for all (or some) i. There are two reasons to prefer an approximation instead. The first is computational — we are not aware of any way to efficiently decide if *any* graph has a given spectrum, let alone find one. Second, while there are non-isomorphic pairs of graphs with the same spectrum, such pairs are exceedingly rare: for example, Butler and Grout [10], based on computational enumeration, estimate that only roughly one in 10^{23} graphs has a non-isomorphic cospectral graph. Thus, most of the time, a generative model would be forced to always return the input graph G^*.

Typical structural properties considered in the context of inverse eigenvalue problems include symmetric non-negative and self-adjoint matrices, both of which have known constructions [20, 22, 33]. Deciding if a spectrum is realized by a real non-negative matrix is NP-hard [9].

Inverse eigenvalue *graph* problems ask whether a graph exists such that a given associated matrix (e.g., adjacency, random walk, Laplacian) has a given spectrum. Godsil and McKay present a method for generating non-isomorphic co-spectral graphs with respect to the adjacency matrix [28]. There has also been work on generating certain families of co-spectral graphs with respect to the Laplacian matrix [40]. We are interested in the (inexact) inverse eigenvalue graph problem with respect to the symmetric normalized Laplacian; with respect to the normalized Laplacian, Butler and Grout [10] have shown how to construct some families of (exactly) cospectral graphs with special structure.

Spectral graph generative approaches have been proposed in some prior work, e.g., [52, 53, 55]. The idea in these approaches is to consider spectral embeddings of the means and covariances of a *set* of graphs, then interpret these embeddings as samples from a distribution, and sample from this distribution, with the goal of interpolating between graphs. The sampling produces a candidate "Laplacian matrix." As discussed above, in general, it is not guaranteed that such a candidate matrix is the Laplacian of any matrix resembling a graph adjacency matrix, and indeed, a lot of our effort here is focused on actually producing a suitable adjacency matrix, and on choosing a "Laplacian matrix" that lends itself to this transformation.

As far as we can tell, [55] does not describe any such procedure. [52] suggests thresholding the values of the matrix, including as edges of the graph those pairs (i, j) for which the entries of the Laplacian are most negative. We compare two different thresholding approaches in this vein to our algorithms in Section 7.3, and find that they perform significantly inferior in matching the spectrum of the input graph.

In very recent work, [3] proposed an approach very similar to ours. Their goal is also to generate graphs whose spectrum approximately matches that of a given input graph; however, [3] use the Modularity matrix [42] instead of the normalized Laplacian. They use a low-rank approximation of the Modularity matrix, which is then transformed back to an adjacency matrix, noised, scaled, and truncated, to define edge probabilities $p_{i,j}$; subsequently, each edge (i, j) is included independently with probability $p_{i,j}$. The rank of the approximation is a user-specified feature, capturing how different the output graphs are likely to be from the input graph.

2 MODEL AND PRELIMINARIES

Throughout this paper, all graphs are undirected, and all matrices (except basis matrices) are real and symmetric. Vectors are denoted by boldface, including $\mathbf{0}$ and $\mathbf{1}$ for the all-0 and all-1 vectors. Graphs have n nodes, and matrices are of dimension $n \times n$.

For any matrix (adjacency or otherwise) $A = (a_{i,j})_{i,j} \in \mathbb{R}^{n \times n}$, we use $\lambda(A)$ to denote the set of A's eigenvalues $\lambda(A) = \{\lambda_1(A) \leq \lambda_2(A) \leq \cdots \leq \lambda_n(A)\}$, also called the *spectrum* of A. We use $\Lambda(A) = \mathrm{diag}(\lambda_1(A), \lambda_2(A), \ldots, \lambda_n(A))$ to denote the diagonal matrix whose diagonal entries are $\lambda_1(A), \lambda_2(A), \ldots, \lambda_n(A)$.

For any matrix A (in particular, an adjacency matrix), we let $d_i(A) = \sum_j a_{i,j}$, and call it the *degree* of node i in A. The degree matrix D for A is $D(A) = \mathrm{diag}(d_1(A), d_2(A), \ldots, d_n(A))$. The identity matrix is denoted by $I = \mathrm{diag}(1, 1, \ldots, 1)$.

Definition 2.1 (Symmetric Normalized Laplacian Matrix). The symmetric normalized Laplacian matrix $L(A)$ for the symmetric $n \times n$ matrix A is $L(A) = I - (D(A))^{-1/2} A (D(A))^{-1/2}$.

We call $\lambda(L(A))$ the *Laplacian spectrum* of A. It is well known that $\lambda_1(L(A)) = 0$, and $\lambda_n(L(A)) \leq 2$ [13]. Throughout this paper, all eigenvalues we consider will be of Laplacians of matrices; when there is no risk of confusion, we then simply refer to the Laplacian eigenvalues as $\lambda_1, \lambda_2, \ldots, \lambda_n$, omitting the dependence on A. In that case, the corresponding (normalized) eigenvectors are denoted by $\boldsymbol{x}_1, \boldsymbol{x}_2 \ldots, \boldsymbol{x}_n$, and the (orthonormal) matrix of eigenvectors is $X(A) = (\boldsymbol{x}_1, \boldsymbol{x}_2 \ldots, \boldsymbol{x}_n)$. The set of all such matrices, i.e., $n \times n$ matrices of orthonormal column vectors, is called the *Stiefel manifold* [18], and denoted by $\mathcal{S}_n$.

The difference $\lambda_2 - \lambda_1 = \lambda_2$ is called the *spectral gap of* $L(A)$. The corresponding eigenvector $\boldsymbol{x}_2$ is called the *Fiedler vector* [21, 23] of A, and plays an important role in identifying low-conductance cuts.

Given a diagonal matrix Λ of eigenvalues, we can obtain a matrix A with spectrum Λ from any basis matrix X simply as $A = W_\Lambda(X) := X \Lambda X^\intercal$; this is nothing but the standard *eigendecomposition* of A. Conversely, given a symmetric real matrix A, there is always a basis (of A's eigenvectors) such that $A = X \Lambda(A) X^\intercal$.

Definition 2.2 (Computational Goal). The given (undirected and connected) graph on n nodes is denoted by G^*, and has (symmetric) adjacency matrix $A^* \in \{0, 1\}^{n \times n}$. The eigenvalues of $L(A^*)$ are $0 = \lambda_1^* \leq \lambda_2^* \leq \cdots \leq \lambda_n^* \leq 2$.

The goal is to randomly sample a graph G with adjacency matrix $A \in \{0, 1\}^{n \times n}$ whose Laplacian spectrum $0 = \lambda_1(L(A)) \leq \lambda_2(L(A)) \leq \cdots \leq \lambda_n(L(A)) \leq 2$ is similar to that of A^*, in the sense that $\lambda_i(L(A)) \approx \lambda_i^*$ for all i.

Definition 2.2 is not formal, in that it does not quantify the meaning of "$\approx$", i.e., it does not specify how close is "close enough" in terms of the Laplacian spectrum of A. It would be desirable to do so, but we are not aware of ways to sample graphs whose spectrum is *provably* close to a given spectrum. Indeed, the following result of Borobia and Canogar [9] suggests that this may be difficult:

Theorem 2.3 ([9]). *The following problem is NP-hard: given a sequence of n real numbers $\lambda_1, \ldots, \lambda_n$, decide if there exists a real nonnegative matrix A of order n with eigenvalues $\lambda_1, \ldots, \lambda_n$.*

Theorem 2.3 does not apply directly to our problem, in that it (1) asks about the spectrum of A, rather than of $L(A)$, and (2) only requires that all entries of A be non-negative, rather than from $\{0, 1\}$. Typically, requiring integrality of a solution makes problems harder (e.g., linear programming), so we expect the hardness result to apply to our problem as well; however, at this time, we have no proof of NP-hardness with the integrality restriction.

3 ALGORITHM OUTLINE

Our graph generative algorithms can be divided into two stages:

(1) **Relaxed Spectrum Fitting:** Find a *template matrix* M whose Laplacian spectrum $\lambda(L(M))$ is very close to the target spectrum $\lambda(L(G^*))$; the entries $m_{i,j}$ may in principle be far from $\{0,1\}$ (including fractional, negative, or larger than 1), but the goal is for them to be close to $\{0,1\}$.
(2) **Template Matrix Perturbation:** Gradually "round" M, first to a matrix $P \in [0,1]^{n\times n}$, then to an actual adjacency matrix. In the process, aim to minimize perturbations to the Laplacian spectrum.

3.1 Relaxed Spectrum Fitting

The Relaxed Spectrum Fitting phase makes heavy use of the correspondence between matrices with a given spectrum and basis matrices. Its goal is to identify a good candidate orthonormal basis X so that the template matrix M defined by $L(M) = W_{\Lambda^*}(X) = X\Lambda^* X^\intercal$ has entries in (or close to) the interval $[0,1]$; or, even better, close to 0 or 1.

Our method starts by generating a random graph $\hat{G}$ from the configuration model [5] with the degree distribution of G^*. Let $\hat{A}$ be the adjacency matrix of $\hat{G}$, and $\hat{X}$ its eigenbasis. In the basic version of the Relaxed Spectrum Fitting phase, we use a linear programming (LP) based approach to find a template matrix M with $L(M) \approx W_\Lambda(\hat{X})$ (see Section 4).

The matrix M could have many entries far from $[0,1]$. Thus, rather than simply using $\hat{X}$, the extended version of our algorithm uses continuous optimization techniques to search the Stiefel manifold for a basis X such that the corresponding template matrix M has entries closer to 0 and 1. This optimization does not scale to large graphs, but more often than not makes small improvements.

3.2 Template Matrix Rounding

As a first step to rounding the template, M is converted to a matrix $P \in [0,1]^{n\times n}$, whose entries can be viewed as edge strengths or probabilities for randomized rounding. We use the following proxy problem: find an "error" matrix E such that $p_{i,j} = m_{i,j} + e_{i,j} \in [0,1]$ for all i,j and the spectrum of M is close to that of P. We show (in Section 6) how to cast the problem of finding E as a linear program, and prove that the objective function gives an upper bound on the sum of eigenvalue perturbations.

Finally, P needs to be converted to an adjacency matrix A. Small eigenvalues λ_k are particularly sensitive to changes going from M to P or from P to A. We therefore treat them separately. Our goal is to make initial edge modifications judiciously to counteract such effects, and to prevent large changes in λ_k for small k. To do so, we focus on the first approximately balanced[4] *eigencut* $(S_k, \overline{S_k})$, defined by $S_k = \{i \mid x_{k,i} < 0\}$ and $\overline{S}_k = \{i \mid x_{k,i} \geq 0\}$, where $\boldsymbol{x}_k$ are eigenvectors of $L(P)$. For each edge crossing $(S_k, \overline{S_k})$, we estimate how much its addition or removal would affect λ_k. We then select edges (i,j) to round $p_{i,j}$ up or down to correct λ_k as well as possible. We observe experimentally that when $k \neq 2$, if rounding edges across $(S_k, \overline{S}_k)$ shrinks λ_k, the spectral gap is typically lowered as well. There could be other candidate cuts that provide valuable connectivity information, but we restrict our search to eigencuts to keep the computational cost manageable.

In a final step, we round all remaining edges, independently including edge (i,j) with probability $p_{i,j}$.

3.3 Randomness

There are two sources of randomness in our generative process: the random initial generation of an orthonormal matrix X via a configuration model graph $\hat{G}$, and the independent random edge inclusions in the final part of the Template Matrix Rounding phase.

Naturally, the process does not guarantee sampling a graph *uniformly* from an ensemble of graphs with approximately correct spectra; however, experimentally, it leads to significant variation in the graphs that are generated, obviously an important criterion for graph generative processes [48].

4 RELAXED SPECTRUM FITTING

We are given a basis X and associated matrix $A = W_{\Lambda^*}(X)$, and would like to identify a matrix M whose normalized Laplacian is $L(M) = A$; ideally M, should be the adjacency matrix of a graph or close to one. It is obvious how to compute $L(M)$ from M. However, we need to compute M given a candidate $L(M)$. The transformation requires knowing the degrees d_i of nodes in M. The following lemma motivates our LP-based approach for finding d_i:

Lemma 4.1. *The matrix A is the Laplacian of some symmetric matrix M with positive row sums if and only if there exists a vector $\boldsymbol{y} > \mathbf{0}$ (all entries are strictly positive) such that $A\boldsymbol{y} = \mathbf{0}$.*

Proof. For the first direction, assume that $A = L(M) = I - D^{-1/2}MD^{-1/2}$ for some symmetric matrix M with positive row sums. Let $d_i = \sum_j m_{i,j} > 0$ be the degree of i in M, and let $\boldsymbol{y}$ be the vector with $y_i = d_i^{1/2}$. Clearly, $\boldsymbol{y} > \mathbf{0}$, and

$$A\boldsymbol{y} = \boldsymbol{y} - D^{-1/2}MD^{-1/2}\boldsymbol{y} = \boldsymbol{y} - D^{-1/2}M\cdot\mathbf{1} = \boldsymbol{y} - D^{-1/2}\boldsymbol{d} = \mathbf{0}.$$

For the converse direction, let $\boldsymbol{y} > \mathbf{0}$ be a solution to $A\boldsymbol{y} = \mathbf{0}$. Let $Y = \mathrm{diag}(y_1, y_2, \ldots, y_n)$, and define $M = Y^2 - YAY$. The vector of row sums of M is

$$M\cdot\mathbf{1} = (y_i^2)_i - YAY\cdot\mathbf{1} = (y_i^2)_i - YA\boldsymbol{y} = (y_i^2)_i > \mathbf{0}.$$

And because

$$L(M) = I - Y^{-1}MY^{-1} = I - Y^{-1}(Y^2 - YAY)Y^{-1} = I - I + A = A,$$

A is indeed the symmetric normalized Laplacian of M. □

The vector entries y_i in Lemma 4.1 are the square roots of the degrees of node i. During the computation, the basis X may be such that the associated matrix $W_{\Lambda^*}(X)$ is not the Laplacian of any matrix M with positive row sums.[5] In order to compute an approximate and usable matrix M, we relax the constraint that $M\boldsymbol{y} = \mathbf{0}$, and instead aim to minimize $||M\boldsymbol{y}||_\infty$. We approximate the positivity constraint by requiring that $y_i \geq \epsilon$ for all i, for a very small positive constant $\epsilon > 0$; we then obtain the following linear program:

[4] For most inputs, $k = 2$ and we round edges across the Fiedler cut. Occasionally, the Fiedler cut is very unbalanced. In those cases, rounding edges across the Fiedler cut very judiciously is largely misspent effort, since later steps are likely to disconnect the smaller side (plus some nodes) from the rest of the graph.

[5] Because $\lambda_1^* = 0$, the matrix $W_{\Lambda^*}(X)$ is always singular. However, the corresponding eigenvector $\boldsymbol{x}_1$ will typically have negative entries, so we are not guaranteed a vector $\boldsymbol{y} > \mathbf{0}$.

$$\begin{array}{lll} \text{Minimize} & b & \\ \text{subject to} & |\sum_{j=1}^{n} a_{i,j} y_j| \le b & i = 1, \ldots, n \\ & y_j \ge \epsilon & j = 1, \ldots, n. \end{array} \quad (1)$$

The procedure for fitting a graph M is now as follows:

(1) Given an orthonormal basis X, compute $A = W_{\Lambda^*}(X)$.
(2) Solve the LP (1), obtaining a solution $(b, \boldsymbol{y})$.
(3) Output the *template matrix* $M = Y^2 - YW_{\Lambda^*}(X)Y$, where $Y = \text{diag}(\boldsymbol{y})$.
(4) To match the edge density of M to that of the target, set $M := \frac{\sum_i d_i^*}{\sum_i d_i} \cdot M$, where d_i^* are the node degrees in the target graph, and $d_i = \sum_j m_{i,j}$.

If $b = 0$, then $L(M) = W_{\Lambda^*}(X)$, and we have found a template matrix M such that $L(M)$ has exactly the desired spectrum Λ^*. We note that the spectrum is invariant to the scaling in Step (4). We perform the scaling to match the target edge density before perturbing the edges, in order to avoid having to perturb the matrix twice to yield entries in the interval $[0, 1]$. Even in this case, individual entries $m_{i,j}$ of the template matrix may be negative or larger than 1 (and of course fractional): the solution $\boldsymbol{y}$ only guarantees that the *total* degree of each node is positive.

When $b > 0$, we have that $L(M) \neq W_{\Lambda^*}(X)$, meaning that typically $\Lambda(M) \neq \Lambda^*$ as well, i.e., the spectrum can be perturbed. We show that the deviation in spectrum can be bounded using the LP objective b; the lemma also motivates our choice of LP objective.

Lemma 4.2. *Let δ be such that $b \le \delta y_i$ for all i. Then, the perturbed eigenvalues satisfy $|\lambda_i(L(M)) - \lambda_i(A)| \le \frac{\delta}{1-\delta} \cdot |\lambda_i(A)|$.*

Proof. The degrees under M are $d_i = \sum_k m_{i,k} = y_i^2 - y_i \sum_k a_{i,k} y_k$, so the perturbed Laplacian matrix $L = L(M)$ has entries

$$\ell_{i,j} = \frac{a_{i,j} y_i y_j}{(y_i^2 - y_i \sum_k a_{i,k} y_k)^{1/2} (y_j^2 - y_j \sum_k a_{j,k} y_k)^{1/2}}.$$

Writing $z_i = \frac{y_i}{(y_i^2 - y_i \sum_k a_{i,k} y_k)^{1/2}}$ and $Z = \text{diag}(z_1, z_2, \ldots, z_n)$, the perturbed Laplacian therefore satisfies $L = ZAZ^\intercal$.

Applying Theorem 4.3 (below) with $D = D^\intercal = Z$, we get that the relative perturbation in eigenvalues is at most

$$|\lambda_i(L(M)) - \lambda_i(A)| \le |\lambda_i(A)| \cdot ||Z^\intercal Z - I||_2.$$

The norm of a diagonal matrix is its maximum entry, so

$$||Z^\intercal Z - I||_2 = \max_i \frac{\sum_k a_{i,k} y_k}{y_i - \sum_k a_{i,k} y_k} \le \max_i \frac{b}{y_i - b},$$

by the LP's first constraint. Because $b \le \delta y_i$ for all i by assumption, we obtain the claimed bound. □

Theorem 4.3 (Theorem 2.1 of [19]). *Let $\tilde{A} = D^\intercal A D$, where D is a nonsingular matrix. Let λ_i and $\tilde{\lambda}_i$ be the eigenvalues of A and $\tilde{A}$, respectively. Then, $|\tilde{\lambda}_i - \lambda_i| \le |\lambda_i| \cdot ||D^\intercal D - I||_2$, for all i.*

5 STIEFEL MANIFOLD OPTIMIZATION

While the techniques in Section 4 find a good M so $L(M) \approx X\Lambda X^\intercal$, if X was a bad basis, no M will be satisfactory, and the entries will lie far outside $[0, 1]$, or the Laplacian eigenvalues will be perturbed. To improve the basis before committing to it, we can perform a walk on the Stiefel manifold, using known techniques to locally optimize an objective function that rewards template matrices M with entries close to 0 or 1. Specifically, we define the objective function $F(M) = \sum_{i<j} (1 - m_{i,j})^2 m_{i,j}^2$, which has minima when all $m_{i,j}$ are in $\{0, 1\}$ and steeply penalizes entries far from 0 and 1.

For any basis $X \in \mathcal{S}_n$, let $\boldsymbol{y}_X$ be the solution to the linear program (1) (applied to $A = X\Lambda^* X^\intercal$), and $Y_X = \text{diag}(\boldsymbol{y})$. Then, $M(X) = Y_X^2 - Y_X X \Lambda^* X^\intercal Y_X$ is the candidate template matrix for X. The objective is then to find a basis $X \in \mathcal{S}_n$ (approximately) minimizing $F(M(X))$.

While the objective function F itself is differentiable (which would allow for a straightforward application of existing manifold optimization techniques), the transformation $X \mapsto M(X)$ is computed via the solution to a linear program. It is not clear how optimal solutions to the linear program (1) change with X, and in particular whether $X \mapsto F(M(X))$ is continuous or differentiable. Therefore, few guarantees for known optimization techniques apply in our setting. To compensate, we use a small maximum step size of .0001 in the optimization.

To perform optimization over the Stiefel manifold, we use known iterative optimization techniques. Specifically, we use the *Polar Decomposition retraction scheme* [2][6]. Retraction schemes in each step t identify a search direction $\boldsymbol{\eta}_t$ in the tangent bundle of the current basis X_t, such that moving in the direction $\boldsymbol{\eta}_t$ minimizes F. (See [1] for definitions of the notions of *tangent bundle* and *retraction*.) The scheme moves X_t by a step size τ_t in the direction $\boldsymbol{\eta}_t$, then projects it back onto the manifold using a retraction $R_X(\tau)$. A retraction scheme specifies both the retraction $R_X(\tau)$ and search direction $\boldsymbol{\eta}$ such that $R_X(\tau)$ is a descent direction: the derivative $R'_X(\tau)$ must be equal to the projection of $-\text{grad}(F(M(X_t)))$ onto the tangent bundle at X. The step size τ_t is typically chosen according to the *Armajiro-Wolfe Conditions*, which ensure that at the point $X_t + \tau_t$, the decrease $F(X_t) - F(X_t + \tau_t)$ is proportional to τ (*sufficient-decrease*), but also that τ_t is sufficiently large such that one cannot decrease $F(X_t + \tau_t)$ by taking a larger step (*curvature condition*) [45]. One iteration of the Stiefel manifold optimization can be summarized as follows:

(1) Using X_t, solve the template fit LP (1) to compute $M(X_t)$.
(2) Treat the y_{X_t} computed from LP (1) as constant, and using $\text{grad}(F(M(X_t)))$, compute a search direction $\boldsymbol{\eta}_t$ using a retraction method.
(3) Perform a line search technique to find a step size τ_t that obeys the Armajiro-Wolfe conditions [45].
(4) Set $X_{t+1} = X_t + \tau_t \boldsymbol{\eta}_t$.
(5) Repeat until a local optimum is reached.

We implemented and experimented with two different retraction methods: the Cayley Transform Retraction [51] and the Polar-Decomposition Retraction [2]. Our experiments showed that the Polar-Decomposition Retraction performed better most of the time.

6 ROUNDING THE TEMPLATE MATRIX

6.1 Perturbing the Template Matrix

The next step is to compute an additive perturbation matrix E that leaves the row sums (i.e., degrees) of M intact (first constraint of

[6] Manifold optimization techniques can generally be divided into two categories: retraction schemes and geodesic schemes. We use retractions as geodesics are often difficult to compute [1].

LP (2)), and ensures that all entries of $P = M + E$ are in $[0, 1]$ (third constraint). Subject to this, we aim to minimize the weighted sum of absolute perturbations. The variables are $e_{i,j}$ and $q_{i,j} = |e_{i,j}|$ (second constraint) for $i \neq j$ (with $e_{i,i} = 0$ implicitly). $d_i = \sum_j m_{i,j}$ denotes the (fractional) degree of i.

$$\begin{array}{lll} \text{Minimize} & \sum_{i \neq j} \frac{q_{ij}}{d_i^{1/2} d_j^{1/2}} & \\ \text{subject to} & \sum_j e_{i,j} = 0, & i = 1, \ldots, n \\ & q_{i,j} \geq |e_{i,j}| & \text{for all } i \neq j \\ & 0 \leq m_{i,j} + e_{i,j} \leq 1 & \text{for all } i \neq j \\ & e_{i,j} = e_{j,i} & \text{for all } i \neq j \end{array} \tag{2}$$

Keeping the degrees constant ensures that the normalization factors for the Laplacian are the same for P and M. The choice of objective function is justified by the following lemma.

Lemma 6.1. *Let $(e_{i,j})$ be an optimal solution of the LP (2), with objective value b. Then, the eigenvalues of the Laplacians of M and $P = M + E$ are close: $\sum_i |\lambda_i(L(M)) - \lambda_i(L(P))| \leq b$.*

Proof. Consider the perturbation $\Delta = L(M) - L(P)$ of the Laplacian matrix. Applying Theorem 6.2 (below) with $\Phi(x) = |x|$, we obtain that $\sum_i |\lambda_i(L(M)) - \lambda_i(L(P))| \leq \sum_i |\lambda_i(\Delta)|$. In the remainder of the proof, we will show that $\sum_i |\lambda_i(\Delta)| \leq b$.

Because Δ is symmetric and real-valued, $\Delta = U^\intercal \Lambda U$ for a diagonal matrix $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ of eigenvalues λ_i of Δ, and orthonormal U. Let $\hat{\Lambda} = \mathrm{diag}(|\lambda_1|, \ldots, |\lambda_n|)$ be the diagonal matrix of absolute values of Δ's eigenvalues, and $\hat{\Delta} = U^\intercal \hat{\Lambda} U$. Define $\sigma_i = \mathrm{sgn}(\lambda_i)$ (with $\mathrm{sgn}(0) := 1$), and $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$ to be the diagonal matrix of signs of λ_i, so that $\hat{\Lambda} = \Sigma\Lambda$. Then, $\hat{\Delta} = U^\intercal \Sigma U \Delta = U'\Delta$, where $U' = U^\intercal \Sigma U$ is a unitary matrix. Because U' is unitary, its entries are bounded by 1 in absolute value. We can therefore bound

$$\sum_i |\lambda_i(\Delta)| = \mathrm{tr}(\hat{\Delta}) = \mathrm{tr}(U'\Delta) = \sum_i \sum_j u'_{i,j} \delta_{j,i} \leq \sum_{i,j} |\delta_{j,i}|.$$

We have shown that the sum of absolute eigenvalues of a (real symmetric) matrix is bounded by the sum of absolute values of its entries[7]. It remains to bound the sum of absolute entries of Δ. Because the degrees (row sums) are the same, i.e., d_i, for M as for P, we obtain that $\delta_{i,j} = \frac{m_{i,j} - p_{i,j}}{d_i^{1/2} d_j^{1/2}}$. Summing their absolute values over all i, j, this is exactly the objective function of the LP (2), which we assumed to be bounded by b. □

Theorem 6.2 (Theorem 1 of [31]). *If A, B are $n \times n$ Hermitian matrices, their eigenvalues can be enumerated in such a way that for every real-valued convex function Φ on $\mathbb{R}$, we have $\sum_i \Phi(\lambda_i(A) - \lambda_i(B)) \leq \sum_i \Phi(\lambda_i(B - A))$.*

6.2 Cut Rounding

As outlined in Section 3, the cuts corresponding to small λ_k are particularly important for global connectivity properties. Our rounding procedure therefore first focuses on matching those eigenvalues. The main point of comparison is the Fiedler cut $(S^*, \bar{S^*})$ of the input graph G^*; without loss of generality, we assume that $|S^*| \leq n/2$. Ideally, we would like to focus on rounding edges across the Fiedler cut of the fractional graph P, so as to match $\lambda_2(G^*)$. However, it is possible[8] that the Fiedler cut of P is extremely unbalanced, with one side only having a handful of nodes. Efforts to round such unbalanced cuts are largely misplaced.

Instead, we focus on the first approximately balanced cut defined by an eigenvector of P. Let $k \geq 2$ be smallest such that the smaller side of the cut defined by $S_k = \{i \mid x_{k,i} < 0\}$ (again, w.l.o.g. S_k rather than its complement) satisfies $|S_k| \geq \frac{1}{2}|S^*|$. We let $\boldsymbol{x} = \boldsymbol{x}_2, \boldsymbol{x'} = \boldsymbol{x}_k$ and refer to $(S_k, \bar{S}_k)$ as the *critical cut*. We write $d_i = \sum_j p_{i,j}$. Our heuristic is guided by the standard characterization $\lambda_2 = \boldsymbol{x} L(P) \boldsymbol{x}^\intercal = \sum x_i^2 - \sum_{i \neq j} x_i x_j \frac{p_{i,j}}{\sqrt{d_i d_j}}$.

When i and j are on opposite sides of the Fiedler cut,[9] their signs in $\boldsymbol{x}$ are opposite, making the term $-x_i x_j \frac{p_{i,j}}{\sqrt{d_i d_j}}$ positive in $\boldsymbol{x} L(P) \boldsymbol{x}^\intercal$. This motivates scoring each edge (i, j) based on $x_i x_j$ as candidates for removal (rounding $p_{i,j}$ down to 0) or addition (rounding $p_{i,j}$ up to 1). However, the heuristic of rounding edges in such a way suffers from two drawbacks:

(1) The degrees d_i and d_j are affected when the edge (i, j) is removed or added; the changes in the terms for edges (i', j) or (i, j') could offset the rounding progress.
(2) The Fiedler vector (and other eigenvectors) may change after removing or adding an edge (i, j), making it difficult to compute a set of edges to remove or add one by one.

Ideally, after each small change to some $p_{i,j}$, one should recompute the eigenbasis before continuing. This is computationally expensive, so instead we adapt an idea of the *NetMelt* algorithm, which was designed to identify good edge removals/additions to decrease/increase the spectral gap of an *adjacency* matrix [11]. We assign scores for each edge removal and edge addition. When working with the adjacency matrix, each edge removal/addition affects only one entry of the perturbation matrix E. For the Laplacian matrix, an edge removal or addition can change the spectrum also through the changes in the nodes' degrees (and thus the normalization).

To help safeguard against the possibility of perturbing many edges adjacent to the same node, thereby changing the degrees drastically and misestimating additional rounding effects, we designate a *budget* b for how many edges can be removed; once the budget has been exceeded, the Fiedler vector $\boldsymbol{x}$, eigenvector $\boldsymbol{x'}$, and scores are recomputed. In addition, if the (fractional) number of edges crossing the Fiedler cut is below 1, we stop the procedure. To keep the overall edge density reasonably constant, we alternate between edge removals (across the cut) and additions (on some side of the cut).

The central part of the rounding algorithm is the following *Critical Cut Rounding* procedure. It is called repeatedly from the overall rounding procedure (described further below), which handles special cases such as "disconnected" probability matrices P and very unbalanced Fiedler Cuts.

(1) Compute $L(P)$, the Fiedler vector $\boldsymbol{x}_2$, the spectral gap $\lambda_2(L(P))$, $\boldsymbol{x'}$ and $\lambda_k(L(P))$, and let $\delta = \lambda_k(L(P)) - \lambda_k^*(L(P))$ be the

[7] We suspect that this must be a well-known fact, but could not find a reference.

[8] We found that this only happened on input graphs with spectral gap less than .01 and with sparse edge density (for example, the Euro Road graph).

[9] The intuition behind the reasoning applies to $\boldsymbol{x'}$ as well, although the variational characterization of λ_k is more complex.

amount by which we would like to decrease λ_k. If $\delta < c$ (where c is a small constant) or the number of (fractional) edges crossing the critical cut or the Fiedler cut is less than 1, exit the Critical Cut Rounding procedure.

(2) For all (i,j) with $x'_i x'_j < 0$ (on opposite sides of the critical cut), let the score be $s_{i,j} = -\frac{x'_i x'_j p_{i,j}}{\sqrt{d_i d_j}} \geq 0$. All such edges (i,j) are removal candidates; we consider them in order from highest to lowest score.

(3) For all (i,j) with $x'_i x'_j > 0$ (on the same side of the critical cut), let the score be $s_{i,j} = \frac{x'_i x'_j (1-p_{i,j})}{\sqrt{d_i d_j}} \geq 0$. All such edges (i,j) are addition candidates; we consider them in order from highest to lowest score.

(4) Keep track of the following quantities: q is the sum of $p_{i,j}$ for all edges (i,j) that have been rounded, s the sum of all scores of alterations, and y the total change in entries in P (where positive and negative changes cancel out). Initially, all of q, s, y are 0.

(5) Until $q \geq b$ or $s \geq \delta/2$, or all addition and removal candidates have been processed:
- If $y \geq 0$, consider the removal candidate (i,j) with largest score $s_{i,j}$; otherwise, consider the addition candidate (i,j) with largest score $s_{i,j}$. (In either case, remove (i,j) from the candidates for the future.)
- If $s + s_{i,j} \leq \delta$ and (i,j) is not the last candidate edge keeping the critical cut connected, update the values as follows (otherwise, skip (i,j)):
  - If (i,j) was a candidate for removal, then set $p_{i,j} = 0$ and $y = y - p_{i,j}$; otherwise, set $p_{i,j} = 1$ and $y = y + (1 - p_{i,j})$.
  - In both cases, update $s = s + s_{i,j}$ and $q = q + p_{i,j}$.

(6) If any changes were made to P, repeat from Step (1).

$L(P)$ and the eigenvectors are computed only once for each overall iteration of the critical cut rounding procedure. An iteration terminates once the approximate score s reaches $\delta/2$, or the algorithm has reached its budget b. At this point, the critical cut and Fiedler cut are recomputed to check whether the algorithm has indeed reached its goal of ensuring $\lambda_k^* \leq \lambda_k \leq \lambda_k^* + c$, and to adapt should the cuts have changed. In our implementation, we use a constant of $c = .0001$ and a budget b equal to $\frac{1}{4}$ of the number of edges crossing the critical cut. When the Critical Cut Rounding procedure terminates, either all edges across the critical cut have been considered, or λ_k is a good approximation to the target: $\lambda_k^* \leq \lambda_k \leq \lambda_k^* + c$.

While Critical Cut Rounding is the key component to our rounding approach, we need to take care of special cases: disconnected template matrices P and extremely unbalanced Fiedler cuts. This is accomplished by the following *Cut Rounding* procedure.

(1) Consider the graph with edges (i,j) iff $p_{i,j} > 0$. If this graph is disconnected, then permanently discard everything except its largest connected component. For the remainder, consider only the remaining nodes.

(2) Compute $L(P)$, the Fiedler vector $\boldsymbol{x}_2$, and the spectral gap $\lambda_2(L(P))$. While the Fiedler cut is severely unbalanced (having fewer than five vertices on the smaller side of the cut), permanently discard the vertices on the smaller side, and recompute $L(P)$, $\boldsymbol{x}_2$, and $\lambda_2(L(P))$.

(3) If $\lambda_2(L(P)) - \lambda_2^*(L(P)) \geq c$ and the number of (fractional) edges crossing the Fiedler cut is at least 1, then perform Critical Cut Rounding, as described above. If Critical Cut Rounding made any changes to P, then repeat from Step (1).

(4) While the (fractional) number of edges crossing the Fiedler Cut is less than 1, round the largest fractional edge $p_{i,j}$ crossing the Fiedler cut up to 1.

In our experiments, the largest fraction of nodes ever disconnected was .5% (including nodes removed due to unbalanced Fiedler cuts).

When the Critical Cut Rounding procedure made the sum of fractional edges crossing the Fiedler cut less than one, the graph would be likely to become disconnected in the later independent rounding step. To prevent this, the final step rounds the largest fractional entry up to 1.

When there are remaining edges to round, we include them in the independent rounding stage described next; this does not change the expected density of edges across the critical cut.

We also experimented with using a dependent rounding scheme [25] that includes each edge with probability $p_{i,j}$, but correlates the random choices so that the the number of edges crossing the Fiedler cut exactly matches the expectation (up to fractional parts). Experimentally, we found the performance of both approaches comparable, so we did not include dependent rounding in our final algorithm.

6.3 Independent Rounding

The final step is to include each remaining edge (i,j) independently with probability $p_{i,j}$, akin to the generation of $G(n,p)$, SBM, or Kronecker graphs. The following theorem by Chung and Radcliffe bounds the resulting spectral perturbations.

Theorem 6.3 (Theorem 2 of [16]). *Let G be a random graph, generated by including each edge (i,j) independently with probability $p_{i,j}$. Let A be the adjacency matrix of the random graph, and $\delta = \min_i \sum_j p_{i,j}$ the minimum expected degree of any node. For every $\epsilon > 0$, there exists a constant $k = k(\epsilon)$ such that if $\delta > k \ln n$, then with probability at least $1-\epsilon$, all eigenvalues of $L(A)$ and $L(P)$ satisfy $|\lambda_i(L(A)) - \lambda_i(L(P))| \leq 3\sqrt{\frac{3\ln(4n/\epsilon)}{\delta}}$.*

7 EXPERIMENTAL EVALUATION

We evaluated our generative model (*SpectralGen*) against the Configuration Model (*Config.*), Stochastic Block Model (*SBM*), and Kronecker graph model (*Kronecker*), based on many real-world data sets, and using a number of metrics.

7.1 Data, Models, and Metrics

Data Sets. We report on results[10] from running on several standard network data sets, available for download from https://icon.colorado.edu/#!/networks, and https://tuvalu.santafe.edu/~simon/styled-9/styled-10/. They are a network of flights between 500 commercial airports in the US, weights representing the number of

[10] We also ran experiments on several other (smaller) network data sets from the Colorado repository: Political books, Football, High School dynamic contacts, Computer Science Faculty Hiring, Physicians social network, Irvine student forum, Jazz. The results were similar to those reported here.

seats (*Airport*), an email exchange network between members of a university (*Email*), self-reported friendships between high school students (*Health*), a road network made up of European cities and roads (*Euro Road*), and links between Wikipedia pages on editorial norms (*Wiki*). Even when the input network was directed (e.g., Wikipedia), we treated the edges as undirected. In addition to these real-world networks, we include a grid graph in our evaluation to test if our method can prevent generating an expander when the target is far from one.

For all networks (input graphs and generated networks), we focused on the the largest connected component rather than the entire graph. This avoids difficulties in comparing spectra or assigning arbitrary distances to disconnected pairs of nodes. We also removed all self loops. The input graphs used and their basic parameters (of the largest connected component) are summarized in Table 1.

	Num Vertices	Num Edges	Spectral Gap	Fiedler Cut Conductance
Airport	500	2980	0.0274	0.0588
Email	1133	5452	0.1211	0.1617
Health	2535	10455	0.0379	0.0654
Euro Road	1039	1305	0.0005	0.0093
Grid 900	900	1740	0.0029	0.0299
Wiki	1872	15367	0.1295	0.5135

Table 1: Basic properties of the input graphs.

Generative Models. We generated networks using the following four models. For the first three, we implemented the models ourselves in MATLAB.

SpectralGen. Our own model, as described in Sections 3–6.[11]

Config. The configuration model [5, 41], matching the input degree sequence.

SBM. For the SBM [29], we used the Fiedler cut to find a high-level bipartition, then fit the edge densities across this partition and within each set to the input graph. Without knowledge of the number of clusters that best fit the SBM to the ground truth graphs, we use two blocks, and choose the edge densities so as to match the conductance across the Fiedler cut in expectation.

Kronecker. The Kronecker model [36], fit and generated using the Stanford Network Analysis Platform (SNAP) [38], to generate a network of size 2^k which is closest to n (the input graph's size). We used the default initial gradient descent matrix and initiator matrix size and 100 gradient descent iterations.

Evaluation Metrics. We compared the four models along multiple network properties.

Spectrum. The difference between the spectra of the generated graphs and the spectrum of the input graph. Doing well for this metric was the main goal of our work.

Path Length Distribution. Distribution of lengths of shortest paths from s to t for all vertex pairs (s, t).

Clustering Coefficient Distribution. Writing $T(v)$ for the number of triangles that v participates in, the clustering coefficient of v is $\frac{T(v)}{d_v(d_v-1)}$, the fraction of pairs of neighbors of v that are neighbors of each other.

Betweenness Centrality Distribution. Letting $\sigma_{s,t}$ denote the number of shortest paths between s and t, and $\sigma_{s,t}(v)$ the number of shortest paths from s to t that pass through v, the betweenness centrality of v is $\sum_{s,t\neq v} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$.

Degree Distribution. Frequency for each degree.

All graph properties were computed using the Python Networkx library. For betweenness centrality, clustering coefficients, degree distributions, and path length distributions, we evaluated how close the distribution of the generated graph was to the distribution of the input graph using the (one-dimensional) Earth Mover's Distance (EMD). EMD is a well-suited distance measure between distributions for our purpose because it increases when more probability mass needs to be shifted, or it needs to be shifted larger distances. For example, we want to consider graphs more similar if path distances are mostly off by 1 than when they are mostly off by 5.

To compare EMD results across different graph properties, we normalized each distribution to the unit interval: more specifically, when computing the EMD between two empirical distributions, we normalized the elements in the supports of both distributions by dividing them by the largest element in the union of their supports. We compute the loss reductions (or increases) of our method as

$$\Delta = (\text{mean EMD BB} - \text{mean EMD SpecGen})/(\text{mean EMD BB}),$$

where "BB" is the *best* benchmark (having the smallest average EMD).

7.2 Results

Figure 1 shows plots of the first 50 eigenvalues of the graphs generated according to the different models; we plot the average and standard deviation for each eigenvalue across 20 runs. The eigenvalues tend to be tracked closer by our method than by the other generative models.

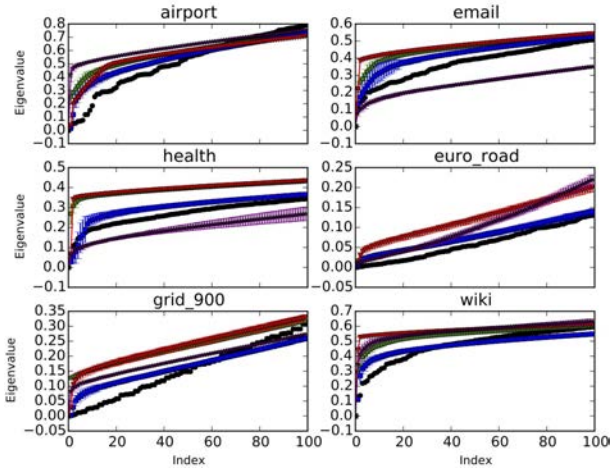


Figure 1: Comparison of spectra. Input graph (black circles), SpectralGen (blue squares), Config (green hexagons), SBM (red triangles), Kronecker (purple stars).

[11] The code is online at https://github.com/alanadakotashine/spectral_generation.

Next, we report the mean and standard deviation EMD[12] of each graph property discussed above for each generative model. These data are reported in Table 2. The spectral generation approach generally outperforms the benchmarks for shortest path lengths and clustering coefficients; however, there are two exceptions.

Our method does not match shortest path lengths on the Wikipedia graph well. This is likely due to the large discrepancy (.384) between the conductance across the Fiedler cut and the spectral gap of the Wikipedia graph, since we observed similar behavior on other graphs with this discrepancy. For graphs with such a large discrepancy, using the density of the Fiedler cut as a stand-in for the graph conductance is a poor approximation.

We included the grid graph in our evaluation, because it is explicitly *not an expander*, while most graph models tend to generate expander graphs. Indeed, our method matches path lengths and betweenness centrality significantly better. It does not do well on clustering coefficients, because it generates too many triangles.

The results are mixed on betweenness centrality. The EMDs for betweenness centrality are smaller than the EMD for other graph properties, which indicates that all methods, including ours, track betweenness centrality reasonably well relative to other graph properties.

The spectral generation approach does not match the degree distribution as well as the configuration model. This is unsurprising considering that the configuration model is designed explicitly to match the degree distribution, making mistakes only when it generates self-loops or multi-edges.

7.3 Evaluating Algorithm Components

Our approach utilizes multiple heuristics. To study the contribution of each heuristic to the final result, in Figure 2, we plot (for two graphs) the spectra obtained by leaving out various steps. We also compare the spectra against those of simple thresholding approaches as they seem to be utilized by [52]. In addition to the spectra of the input graphs, we show the spectra of six matrices.

(1) The result of using the initial template matrix C from the configuration model (with entries $c_{i,j} = d_i d_j / m$), and applying our rounding procedure from Section 6, without first applying LP (1). C is a reasonable candidate for rounding because it matches the degrees of G^*, and all its entries are already in $[0, 1]$.
(2) The template matrix output by the LP (1). This matrix may have entries outside $[0, 1]$.
(3) The matrix output by the LP (2), which forced entries from the template inside $[0, 1]$.
(4) The final rounded output using the Stiefel manifold optimization along with the LP (1) for relaxed spectrum fitting.
(5) The final rounded output without using the Stiefel manifold optimization, but only using the LP (1) for relaxed spectrum fitting.
(6) We derive the following random graph using methods similar to [52]. We combine the spectrum of the target graph and a uniformly random orthonormal matrix X to define the Laplacian matrix $L = X \Lambda X^\intercal$. A graph is produced by simple thresholding: the edge (i, j) is included iff $l_{i,j} < \theta$ (recall that a Laplacian matrix has negative off-diagonal entries for (i, j) that correspond to edges), with the threshold θ chosen so that the number of edges matches the target.
(7) Again in the spirit of [52], we consider a thresholding of our template matrix M. In this case, we include all edges (i, j) with $m_{i,j} > \theta$, again choosing θ such that the number of edges matches the target.

[12]Statistics are taken over $k = 20$ runs for all graphs.

Shortest Paths

	SpecGen	Config	Kronecker	SBM	Δ
Air.	.02 ± .008	.03 ± .001	.04 ± .003	.02 ± .003	.12
Eml.	.02 ± 4e-3	.03 ± 2e-3	.08 ± .01	.03 ± 2e-3	.4
Hlth.	.03 ± 8e-3	.07 ± 3e-4	.04± 6e-3	.05 ± 5e-3	.29
Euro	.14± .02	.15 ± 2e-3	.15 ± 7e-3	.17± .01	.07
Grid	.2 ± 9e-3	.25± 2e-4	.25 ± 8e-4	.25 ± 2e-3	.17
Wiki	.04 ± 3e-3	.01 ± 5e-4	.01 ± 3e-3	.03 ± 5e-4	-2.4

Clustering Coefficients

	SpecGen	Config	Kronecker	SBM	Δ
Air.	.31 ± .02	.39 ± .02	.48 ± .02	.59 ± .003	.25
Eml.	.13 ± .03	.2 ± .002	.21 ± .002	.21 ± 7e-4	.53
Hlth.	.09 ± .006	.14 ± 6e-4	.14 ± .002	.14± 5e-4	.55
Euro	.01± 6e-3	.02± 2e-3	.02 ± .003	.02± 2e-3	.02
Grid	.04 ± .01	.002± .001	.0001 ± 1e-4	.01 ± .001	-39
Wiki	.25 ± .01	.27 ± .003	.3 ± .01	.37 ± 3e-4	.07

Betweenness Centralities

	SpecGen	Config	Kronecker	SBM	Δ
Air.	.01 ± 1e-3	8e-3 ± 4e-4	.01 ± 4e-4	.02 ± 5e-4	-.4
Eml.	.01 ± 2e-3	7e-3 ± 3e-4	.02 ± 5e-3	.04 ± 6e-4	-.6
Hlth.	5e-3 ± 3e-3	.02 ± 9e-5	.04 ± .02	.03 ± 5e-4	.76
Euro	.03± 4e-3	.04 ± 3e-4	.02 ± 2e-3	.03± 2e-3	-.6
Grid	.09 ± .04	.34± 4e-4	.34 ± .02	.32 ± .04	.71
Wiki	2e-3 ± 7e-4	5e-3 ± 9e-4	4e-3 ± 2e-4	6e-3 ± 3e-3	.5

Degree Distribution

	SpecGen	Config	Kronecker	SBM	Δ
Air.	.02 ± 2e-3	.01 ± .08	.03 ± 1e-3	.07 ± 1e-3	-.2
Eml.	8e-3 ± 3e-3	2e-3± 2e-4	.07 ± 5e-4	.06 ± 8e-4	-2.8
Hlth.	.01 ± 7e-3	8e-3 ± 2e-4	.14 ± 7e-3	.04 ± 9e-4	-.25
Euro	.01± 3e-3	.002± 7e-4	.03 ± 3e-3	.05± 5e-3	-5.9
Grid	.06 ± 8e-3	2e-3± 9e-4	.12 ± .01	.12 ± .01	-30.7
Wiki	4e-3 ± 5e-4	3e-3 ± 6e-5	7e-3 ± 2e-4	.03 ± 2e-4	-.4

Table 2: Models' Performance for matching graph property distributions and loss reduction.

Our experimental results show that the template matrix spectrum closely matches the desired spectrum; thus, the deviation in the final spectrum is mostly a result of pushing the template matrix entries into $[0, 1]$, then rounding them. Using C as a template and using our rounding scheme normally suffices to produce strong performance on the spectral *gap*, but fails to match the other eigenvalues. This demonstrates that a careful choice in template helps preserve more of the spectrum. Thresholding the Laplacian and adjacency matrices performs poorly not only on the overall spectrum, but even on the spectral gap. This confirms that a more careful rounding procedure and generation of a suitable template are necessary to generate graphs matching a desired spectrum.

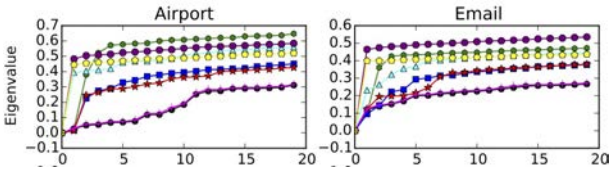


Figure 2: Spectra of matrices produced by employing different sets of heuristics. Input graph (black circles), rounding an unfitted matrix (green hexagons), fitted template matrix (magenta plus), fractional graph (cyan triangles), output graph without Stiefel (blue squares), output graph with Stiefel (red stars), thresholding the Laplacian (yellow pentagons), thresholding the adjacency matrix (purple octagons).

We observe that the Stiefel manifold optimization did not always improve performance of the final result, but more often than not provided small improvements. However, for large graphs, the computation becomes expensive.

8 CONCLUSION

We developed heuristics for randomly generating graphs whose Laplacian spectra approximately match that of a given input graph. Our experiments on multiple real-world network data sets show that for other graph parameters of interest, too, the graphs generated in this way match their real-world counterpart as well as or better compared to several widely used generative models.

Our work is nowhere near the final word on the subject, but should instead be considered as an invitation to the community to focus on spectral properties as an important criterion in random graph generative models.

Among the most obvious directions for future work is to devise faster heuristics. While our approach scales to graphs of several thousand nodes, real-world networks of interest often have millions of nodes; we would like to generate graphs of this size that match global spectral properties of given networks.

A second natural direction is to derive sampling algorithms with *provable* guarantees. The most obvious desirable guarantee would be that the spectrum of the generated graphs be "close" to Λ^* *provably*, or alternatively, that the distribution of deviations can be characterized. An even stronger type of guarantee would be to show that the generated graphs be drawn (nearly) *uniformly* from a suitable class of graphs with spectra similar to Λ^*.

A (perhaps more likely) alternative would be to prove hardness results. Paralleling the two tasks described in the previous paragraph, hardness results could come in two varieties: (1) show that deciding whether graphs exist whose Laplacian spectrum is close to a given Λ^* is hard, or (2) show that sampling nearly uniformly from graphs with the desired Laplacian spectra is hard.

The motivation for our work was that the spectrum naturally characterizes global connectivity properties. However, it is not the only graph metric to do so, and one could consider several others. For example, an alternative approach would be to sample graphs G such that for every partition $(S, \bar{S})$ of the vertices, the conductance (or expansion, or number of edges, or some other parameter) is approximately the same as in G^*. An obvious downside of this approach is that even to verify whether G is similar to G^* in this sense requires inspecting exponentially many cuts instead of linearly many eigenvalues, unless more efficient algorithms are designed.

Finally, the initial motivation for our work raises a very intriguing general direction. We implied (in Section 1) that even when local properties (such as degree distributions, triangle counts, other motifs) of a graph are fixed, generating graphs under such constraints still produces expander graphs with high probability[13]. Without any qualifications, this claim is definitely false — for example, a 3-regular graph in which each node participates in 3 triangles must be the union of disjoint 4-cliques, not an expander. However, under a careful formalization, we believe that such a claim should hold true, and it would be desirable to corroborate the intuition that local motifs do not constrain global structure enough to prevent most graphs from being expanders.

ACKNOWLEDGMENTS

We would like to thank Aaron Clauset, Aram Galstyan, Kristina Lerman, Elchanan Mossel, and Shanghua Teng for useful discussions and pointers, and anonymous reviewers for useful feedback. Work supported in part by NSF grant IIS-1619458 and ARO MURI 72924-NS-MUR.

REFERENCES

[1] P.-A. Absil, Robert Mahony, and Rodolphe Sepulchre. 2009. *Optimization algorithms on matrix manifolds.* Princeton University Press.
[2] P.-A. Absil and Jérôme Malick. 2012. Projection-like retractions on matrix manifolds. *SIAM Journal on Optimization* 22, 1 (2012), 135–158.
[3] Luca Baldesi, Carter T Butts, and Athina Markopoulou. 2018. Spectral Graph Forge: Graph Generation Targeting Modularity. In *Proc. 37th IEEE INFOCOM Conference*. IEEE, 1727–1735.
[4] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512.
[5] Edward A. Bender and E. Rodney Canfield. 1978. The asymptotic number of labelled graphs with given degree sequences. *Journal of Combinatorial Theory (A)* 24 (1978), 296–307.
[6] Noam Berger, Christian Borgs, Jennifer Chayes, and Amin Saberi. 2005. On the Spread of Viruses on the Internet. In *Proc. 16th ACM-SIAM Symp. on Discrete Algorithms*. 301–310.
[7] Aleksandar Bojchevski, Oleksandr Shchur, Daniel Zügner, and Stephan Günnemann. 2018. NetGAN: Generating Graphs via Random Walks. In *Proc. 35th Intl. Conf. on Machine Learning*. 610–619.
[8] Béla Bollobás. 1998. Random Graphs. In *Modern Graph Theory*. Springer, 215–252.
[9] Alberto Borobia and Roberto Canogar. 2017. The real nonnegative inverse eigenvalue problem is NP-hard. *Linear Algebra Appl.* 522 (2017), 127 – 139.
[10] Steve Butler and Jason Grout. 2011. A construction of cospectral graphs for the normalized Laplacian. *The Electronic Journal of Combinatorics* 18, 1 (2011), 231.
[11] Chen Chen, Hanghang Tong, B. Aditya Prakash, Tina Eliassi-Rad, Michalis Faloutsos, and Christos Faloutsos. 2016. Eigen-Optimization on large graphs by edge manipulation. *ACM Transactions on Knowledge Discovery from Data* 10, 4 (2016), 49.
[12] Moody T. Chu and Gene H. Golub. 2005. *Inverse Eigenvalue Problems: Theory, Algorithms, and Applications.* Oxford University Press.
[13] Fan R. K. Chung. 1997. *Spectral Graph Theory.* American Mathematical Society.
[14] Fan R. K. Chung and Linyuan Lu. 2002. The average distance in random graphs with given expected degrees. *Proc. Natl. Acad. Sci. USA* 99 (2002), 15879–15882.
[15] Fan R. K. Chung, Linyuan Lu, and Van Vu. 2003. The spectra of random graphs with given expected degrees. *Proc. Natl. Acad. Sci. USA* 100, 11 (2003), 6313–6318.
[16] Fan R. K. Chung and Mary Radcliffe. 2011. On the spectra of general random graphs. *The Electronic Journal of Combinatorics* 18, 1 (2011), P215.
[17] Nicola De Cao and Thomas Kipf. 2018. MolGAN: An implicit generative model for small molecular graphs. In *ICML 2018 Workshop on Theoretical Foundations and Applications of Deep Generative Models.*

[13] Newman [43] alludes to a similar observation that random graphs tend to be "locally tree-like;" contrary to us, he considers this property to be desirable in a model to facilitate analysis.

[18] Alan Edelman, Tomás A. Arias, and Steven T. Smith. 1998. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.* 20, 2 (1998), 303–353.

[19] Stanley C. Eisenstat and Ilse C. F. Ipsen. 1995. Relative perturbation techniques for singular value problems. *SIAM J. Numer. Anal.* 32, 6 (1995), 1972–1988.

[20] Matthew Fickus, Dustin G. Mixon, Miriam J. Poteet, and Nate Strawn. 2013. Constructing all self-adjoint matrices with prescribed spectrum and diagonal. *Advances in Computational Mathematics* 39, 3-4 (2013), 585–609.

[21] Miroslav Fiedler. 1973. Algebraic connectivity of graphs. *Czechoslovak mathematical journal* 23, 2 (1973), 298–305.

[22] Miroslav Fiedler. 1974. Eigenvalues of nonnegative symmetric matrices. *Linear Algebra Appl.* 9 (1974), 119–142.

[23] Miroslav Fiedler. 1975. A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal* 25, 4 (1975), 619–633.

[24] Ove Frank and David Strauss. 1986. Markov Graphs. *J. Amer. Statist. Assoc.* 81, 395 (1986), 832–842.

[25] Rajiv Gandhi, Samir Khuller, Srinivasan Parthasarathy, and Aravind Srinivasan. 2006. Dependent rounding and its applications to approximation algorithms. *J. ACM* 53, 3 (2006), 324–360.

[26] Ayalvadi Ganesh, Laurent Massoulié, and Donald Towlsey. 2005. The effect of network topology on the spread of viruses. In *Proc. 24th IEEE INFOCOM Conference*. 1455–1466.

[27] Minas Gjoka, Maciej Kurant, and Athina Markopoulou. 2013. 2.5 k-graphs: from sampling to generation. In *Proc. 32nd IEEE INFOCOM Conference*. IEEE, 1968–1976.

[28] Chris D. Godsil and Brendan D. McKay. 1982. Constructing cospectral graphs. *Aequationes Mathematicae* 25, 1 (1982), 257–268.

[29] Paul W. Holland, Kathryn B. Laskey, and Samuel Leinhardt. 1983. Stochastic Blockmodels: Some First Steps. *Social Networks* 5 (1983), 109–137.

[30] Brian Karrer and Mark E. J. Newman. 2011. Stochastic Blockmodels and Community Structure in Networks. *Physical Review E* 83 (2011), 016107.

[31] Tosio Kato. 1987. Variation of Discrete Spectra. *Communications in Mathematical Physics* 111 (1987), 501–504.

[32] Vaishnavy Krishnamurthy, Michalis Faloutsos, Marek Chrobak, Jun-Hong Cui, Li Lao, and Allon G. Percus. 2007. Reducing Large Internet Topologies for Faster Simulations. *Computer Networks* 51, 15 (2007), 4284–4302.

[33] Thomas J. Laffey and Helena Šmigoc. 2007. Construction of nonnegative symmetric matrices with given spectrum. *Linear algebra and its applications* 421, 1 (2007), 97–109.

[34] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. 2008. Benchmark graphs for testing community detection algorithms. *Physical Review E* 78 (2008), 046110.

[35] James R. Lee, Shayan Oveis Gharan, and Luca Trevisan. 2014. Multiway spectral partitioning and higher-order Cheeger inequalities. *J. ACM* 61, 6 (2014), 37.

[36] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. 2010. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research* 11 (2010), 985–1042.

[37] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph Evolution: Densification and Shrinking Diameters. *ACM Transactions on Knowledge Discovery from Data* 1, 1 (2007), 2.

[38] Jure Leskovec and Rok Sosič. 2016. SNAP: A General-Purpose Network Analysis and Graph-Mining Library. *ACM Transactions on Intelligent Systems and Technology* 8, 1 (2016), 1.

[39] Yujia Li, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter Battaglia. 2018. Learning deep generative models of graphs. (2018). arXiv preprint arXiv:1803.03324.

[40] Russell Merris. 1997. Large families of Laplacian isospectral graphs. *Linear and Multilinear Algebra* 43, 1–3 (1997), 201–205.

[41] Michael Molloy and Bruce Reed. 1995. A critical point for random graphs with a given degree sequence. *Random Structures and Algorithms* 6, 2-3 (1995), 161–180.

[42] Mark E. J. Newman. 2006. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E* 74 (2006), 036104.

[43] Mark E. J. Newman. 2009. Random graphs with clustering. *Physical Review Letters* 103, 5 (2009), 058701.

[44] Mark E. J. Newman, Steven H. Strogatz, and Duncan J. Watts. 2001. Random graphs with arbitrary degree distributions and their applications. *Physical Review E* 64, 2 (2001), 026118.

[45] Jorge Nocedal and Stephen Wright. 2006. *Numerical Optimization*. Springer.

[46] Chiara Orsini, Marija M. Dankulov, Pol Colomer-de Simón, Almerima Jamakovic, Priya Mahadevan, Amin Vahdat, Kevin E. Bassler, Zoltán Toroczkai, Marián Boguñá, Guido Caldarelli, Santo Fortunato, and Dmitri Kiroukov. 2015. Quantifying randomness in real networks. *Nature Communications* 6 (2015), 8627.

[47] Garry Robins, Philippa Pattison, Yuval Kalish, and Dean Lusher. 2007. An introduction to exponential random graph (p^*) models for social networks. *Social Networks* 29 (2007), 173–191.

[48] Christian L Staudt, Michael Hamann, Alexander Gutfraind, Ilya Safro, and Henning Meyerhenke. 2017. Generating realistic scaled complex networks. *Applied Network Science* 2, 1 (2017), 36.

[49] Yuchung J. Wang and George Y. Wong. 1987. Stochastic Blockmodels for Directed Graphs. *J. Amer. Statist. Assoc.* 82, 397 (1987), 8–19.

[50] Duncan J. Watts and Steven Strogatz. 1998. Collective dynamics of 'small-world' networks. *Nature* 393 (1998), 440–442.

[51] Zaiwen Wen and Wotao Yin. 2013. A feasible method for optimization with orthogonality constraints. *Mathematical Programming* 142, 1–2 (2013), 397–434.

[52] David White and Richard C. Wilson. 2007. Spectral generative models for graphs. In *Proc. Intl. Conf. on Image Analysis and Processing*. 35–42.

[53] David H. White. 2009. *Generative Models for Graphs*. Ph.D. Dissertation.

[54] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. 2019. A comprehensive survey on graph neural networks. (2019). arXiv preprint arXiv:1901.00596.

[55] Bai Xiao and Edwin R. Hancock. 2006. A spectral generative model for graph structure. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. 173–181.

[56] Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. 2018. GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. In *Proc. 35th Intl. Conf. on Machine Learning*. 5708–5717.

VACCINE: Using Contextual Integrity For Data Leakage Detection

Yan Shvartzshnaider
New York University
yansh@nyu.edu

Zvonimir Pavlinovic
New York University
zvonimir@cs.nyu.edu

Ananth Balashankar
New York University
ananth@nyu.edu

Thomas Wies
New York University
wies@cs.nyu.edu

Lakshminarayanan Subramanian
New York University
lakshmi@nyu.edu

Helen Nissenbaum
Cornell Tech
helen.nissenbaum@cornell.edu

Prateek Mittal
Princeton University
pmittal@princeton.edu

ABSTRACT

Modern enterprises rely on Data Leakage Prevention (DLP) systems to enforce privacy policies that prevent unintentional flow of sensitive information to unauthorized entities. However, these systems operate based on rule sets that are limited to syntactic analysis and therefore completely ignore the semantic relationships between participants involved in the information exchanges. For similar reasons, these systems cannot enforce complex privacy policies that require temporal reasoning about events that have previously occurred.

To address these limitations, we advocate a new design methodology for DLP systems centered on the notion of Contextual Integrity (CI). We use the CI framework to abstract real-world communication exchanges into formally defined information flows where privacy policies describe sequences of admissible flows. CI allows us to decouple (1) the syntactic extraction of flows from information exchanges, and (2) the enforcement of privacy policies on these flows. We applied this approach to built VACCINE, a DLP auditing system for emails. VACCINE uses state-of-the-art techniques in natural language processing to extract flows from email text. It also provides a declarative language for describing privacy policies. These policies are automatically compiled to operational rules that the system uses for detecting data leakages. We evaluated VACCINE on the Enron email corpus and show that it improves over the state of the art both in terms of the expressivity of the policies that DLP systems can enforce as well as its precision in detecting data leakages.

CCS CONCEPTS

• **Security and privacy** → *Usability in security and privacy.*

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313655

KEYWORDS

Contextual Integrity; Data Leakage Detection; DLP; Privacy

ACM Reference Format:
Yan Shvartzshnaider, Zvonimir Pavlinovic, Ananth Balashankar, Thomas Wies, Lakshminarayanan Subramanian, Helen Nissenbaum, and Prateek Mittal. 2019. VACCINE: Using Contextual Integrity For Data Leakage Detection. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313655

1 INTRODUCTION

Modern enterprises rely on Data Leakage Prevention (DLP) systems [12, 20, 27, 28, 39] to prevent the flow of sensitive information such as credit card and other business data to unauthorized entities. Rather than combating malicious behavior by end users, the goal of these systems is to prevent unintentional data leakage. This goal is accomplished by monitoring whether information flows (e.g. extracted from emails) adhere to a set of rules [35] that encode the organizational data handling policies, industry-specific regulations, and general privacy laws, and block or flag those flows that violate the rules. This monitoring process may be performed online as data is in motion, or offline to audit data that is in rest.

We will argue later that a key issue in the design of existing DLP systems is that they conflate the problem of extracting information flows from information exchanges and the problem of checking whether these flows are admissible. As a consequence, the tremendous progress made in natural language processing and machine learning techniques had little impact on modern DLP systems. Moreover, the rules that can be expressed in these systems fail to meet the demands imposed by the increasingly complex privacy policies and regulations that they must enforce. Specifically, they cannot account for contextual information such as consent, which involves reasoning about histories of information flows. The recent General Data Protection Regulation (GDPR) by the European Union is just one example of a privacy policy where such contextual notions are prevalent.

To address these limitations, we advocate a new design methodology for DLP systems centered on the notion of Contextual Integrity

(CI) [30]. The key idea behind this methodology is to view real-world communication exchanges as sequences of information flows as defined by the CI framework. A CI flow represents an atomic information exchange unit identified by the sender, recipient, attribute (i.e. data) being exchanged, and the subject associated with the attribute. CI flows carry the essential information being communicated in information exchanges while abstracting from the low-level details of these exchanges. Privacy policies are modeled as contextual norms that describe flow sequences admissible in a particular privacy context. DLP systems based on this view allow one to take advantage of advances in natural language processing techniques for extracting CI flows on one hand, and advances in formal methods for specifying and automatically enforcing privacy policies on the other hand.

The main technical contribution of this paper is VACCINE, a DLP auditing system for emails that follows this new CI-based design methodology. VACCINE provides users with a simple but expressive language for the declarative specification of complex privacy policies. In particular, it supports policies that require a DLP system to reason about the history of past information flows, which is not supported by other existing DLP systems in this general form. The policies are automatically compiled to sound-by-construction rules, implemented as SQL queries. These queries are used to check the admissibility of the extracted information flows.

VACCINE further leverages advanced techniques in natural language processing [23], such as syntactic dependency parsing [13], name entity recognizers [26], and question answering models [34], to efficiently extract contextual flows from complex email corpora. It combines rule-based and model-based techniques into a single combined model learned end-to-end in a generator-discriminator setting using distant supervision techniques.

We have evaluated VACCINE by applying it to the Enron email corpus [9, 36]. Compared to flow extraction used in existing DLP systems, we achieve improvements in F1 accuracy in detecting attributes by 4% and subject pairs by 32%. We have also used VACCINE to describe and enforce Enron privacy policies extracted from corporate privacy handbooks on the flows extracted from the email corpus. Our experiments demonstrate that VACCINE is effective at detecting violations of complex privacy policies in a real-world corporate setting.

2 DESIGN METHODOLOGY

In this section, we discuss the limits of commercial DLP systems and outline our new design methodology. For a detailed comparison with related work, see Section 5.

To check whether a particular information exchange such as sending an email is admissible, a DLP system has to solve two subproblems, as illustrated in Figure 1. First, it has to extract the (implicit) information flows from the information exchange. For instance, suppose Bob sends an email to Alice containing the phrase: *"You can find Mary's address below. Her credit card number is..."*. This email, thus, contains the implicit flow that Bob sends Mary's credit card data to Alice. Once this flow has been extracted, the second subproblem is then to check whether it is admissible according to the specified rules. For example, if a rule can disallow *any* flow of credit card data via email whatsoever, then Bob's email would be blocked or flagged as a data leakage. However, a more refined rule could include an additional relevant contextual information that, e.g., only disallowed flows with *Alice's* credit card data from Bob to Mary. In this case, the information flow would not be blocked.

Hence, the administrator of a DLP system has to provide two types of inputs to the system: (1) a specification of the information flows to be extracted from information exchanges, and (2) a specification of the rules that determine which of these extracted flows are admissible.

The issue with the design of current DLP systems is that, from the perspective of the administrator, these two types of information are conflated. The rules are specified directly in terms of the underlying information exchange format using combinations of regular expressions, templates, keywords, or text patterns. This severely limits the capabilities of these systems. In particular, the focus on regular expressions means that these systems cannot easily deploy more sophisticated flow extraction techniques based on natural language processing. For instance, the fact that the credit card data sent in Bob's email belongs to Mary cannot be captured by a regular expression because the flow extractor has to establish the identity of "Mary" and the possessive adjective "her" in the two consecutive sentences of the email, which involves a more complex analysis of the grammatical structure of these sentences.

It is therefore not surprising that current DLP systems only consider very basic notions of information flows and rules. While they can reason about senders and receivers of information flows like Bob and Alice, which can be easily extracted from the header of the email, there is no notion of a subject whose information is being transmitted, like Mary in our example. This, in turn, limits the expressivity of the rules that can be checked by these systems. For example, a more sophisticated DLP policy could demand that the flow from Bob to Alice is admissible as long as it is Bob's credit card number that Bob is sharing. Without the notion of a subject of the transferred information, such policies cannot be expressed.

A related limitation of existing systems is that their rules can only refer to a single information exchange (the email from Bob to Alice in our example). This means that they can only take into account very limited contextual information. For instance, a reasonable privacy policy could demand that the flow from Bob to Alice is only admissible if Mary has given explicit consent to sharing her information. Checking such policies requires reasoning about the history of information flows that happened in the past, which cannot be done in existing systems. However, such policies are common in privacy laws such as GDPR [1], which these systems are meant to enforce.

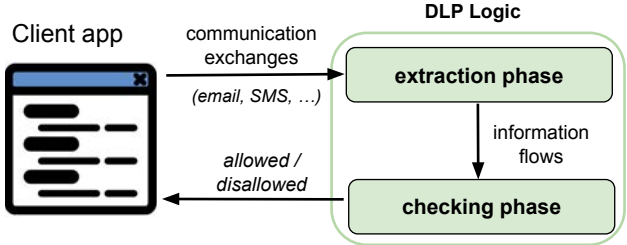


Figure 1: Illustration of an the enforcement mechanism that comprises two main phases: extraction and checking

2.1 Solving the Conflation Problem

Naturally, to solve the conflation problem, we need to separate the extraction and checking phases to make way for a modular DLP system architecture that is parameterized by the solutions to the two phases. A key technical challenge in achieving such separation is in defining a suitable information flow abstraction between the two phases. This abstraction will need to support seamless integration of customized solutions for each of the phases and needs to encompass relevant aspects of the communication exchanges and be amendable to different types of policies. Furthermore, the separation of extraction and checking phases in policy enforcement enables a natural division of labor in the process of implementing rules. For instance, on the one side, NLP experts can focus on devising techniques for extracting information flows from communication exchanges. On the other side, legal scholars and lawyers can focus on the task of checking that information flows obey the policies of interest, not worrying about the low-level details involving actual communication exchanges.

In this regard, the account of privacy [30] as contextual integrity offers a compelling answer to the question of how to tackle the aforementioned issues. It offers a) a suitable abstraction to decouple and interface between the extraction and checking mechanisms, and b) a common language for encoding legal policy and regulations such as FEPRA, HIPAA, and DLP rules into privacy rules. Moreover, CI overcomes the conflation problem by enabling the use of existing techniques tailored for the flow extraction and flow checking phases.

Next, we give a brief introduction to CI and also discuss how the theory and framework can be used to model the data leakage problem.

2.2 Contextual Integrity Overview

The theory of contextual integrity (CI) postulates that informational privacy is not all about secrecy (blocking information) [31] or control [44] but about the appropriateness of information flow within a particular context. Appropriateness of flow means flow that is compliant with contextual norms governing informational flows. To express an informational norm one must specify key parameters: *actors* (senders, recipients and subjects), *attributes* (the type of information at hand) and *transmission principles* (the constraints imposed on a particular information flow). Taken together, these components constitute context-relative informational norms. For instance, in the health context, the patient, acting in his capacity as both the sender and subject of an information flow, could be telling his doctor, the recipient, about his health issues, the attribute. The information flow would be constrained by the transmission principle of confidentiality, which restricts the onward flow of this information to other parties. Exchange the doctor for a friend, and the transmission principle might be reciprocity instead, since friends tend to expect to hear about each other's problems. A patient, by contrast, does not expect to hear about the health issues of his doctor. This is because the context of health and the context of friendship have different overarching goals: the doctor is there to promote the patient's health; friends are there to support each other. An informational norm is breached when an action or practice disrupts the actors, attributes, or transmission principles within a given information flow. Contextual integrity "is preserved when informational norms are respected and violated when informational norms are breached" [30].

It is worth emphasizing one of the most fundamental aspects of CI, namely that, in order to determine whether or not information flows respect or violate privacy expectations within a given context, one must address all three parameters: actors, information types, and transmission principles. Omitting any one of them may lead to an inconclusive or ambiguous description. Accordingly, any formal rendering of information flows, for the purpose of assessing their appropriateness, needs to include independent variables for these parameters. Defining privacy [30] in terms of CI is now a well-established approach in the privacy and computer science research communities. It is used to describe contextual informational norms, to detect infractions of these norms, and in approaches to accountability and enforcement [5, 17].

2.3 Data Leakage Detection using CI

In its core, the theory of CI offers a way to reason about the appropriateness of an information exchange. An *information flow* is the atomic unit of an information (*attribute*) exchange between *actors*. Information flow is formally defined by a tuple:

```
[sender, recipient, subject, attribute]
```

Actors (sender, recipient, subject) take on different roles/objectives that define the capacity in which they function in a given context such as doctors, teachers, or network middleware. Appropriateness of information flow, within a particular context, is defined using contextual informational *norms*. A norm consists of:

```
[Sender, Recipient, Subject, Attribute,
                    Transmission principle]
```

Here `Sender`, `Recipient`, and `Subject` refer to roles and `Attribute` to the type of information exchange under a specified `Transmission principle` condition. For instance, in the educational setting, where information exchange between actors is regulated by the FERPA policies, *a teacher* (sender role) would be allowed to share *grades* (attribute) of a *student* (subject role) to her *parents* (recipient role) if the student has given *explicit written permission* (transmission principle) [42]. If the student has not given such a permission, flows from the teacher to a student's parents involving her grades must not be allowed, i.e., such information exchanges are not appropriate. Thus, it is important to realize that contextual norms specify the appropriateness of a flow subject to conditions that may depend on the history of previous information flows.

We can use the above CI model to define a data leakage. In CI terms, a data leakage occurs within a particular context when an information exchange (flow sequence) is not deemed appropriate by any of the specified contextual norms. We use the above as a system abstraction for building real-world DLP systems that prevent data leakage by only allowing sequences that are consistent with the norms. More formally, we denote the flow sequences that the system deems admissible with T_s and the contextual informational norms of a privacy context as a set of flow sequences, T_n. A DLP system is designed to *correctly* enforce a privacy context by guaranteeing that $T_s \subseteq T_n$. i.e., the system exhibits flow sequences allowed by the given norms.

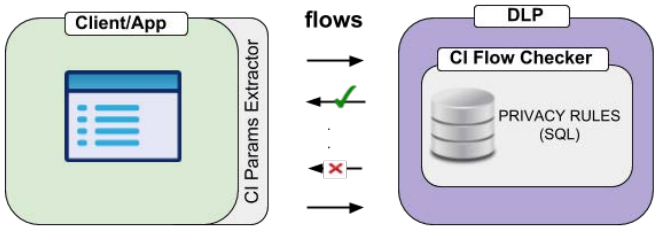


Figure 2: *VACCINE:* Information exchanges are processed by the CI Parameters Extractor that forms corresponding CI flows that are then fed into CI Flow Checker to check for violations with respect to existing privacy norms

3 VACCINE

We next describe our Verifiable and ACtionable Contextual Integrity Norms Engine (VACCINE), a DLP system for email communication based on CI. As illustrated by Figure 2, VACCINE maps information exchanges at the application level to the corresponding CI flows and checks them against the specified CI norms. The information flow abstraction is used to communicate the required contextual information to the checking mechanism.

Specifying norms. Norms can originate from a combination of external sources such as privacy handbooks and context specifications. Other external sources may also be used as a base for CI norms in a specific domain, such as legal and policy documents, professional codes of conduct, findings of empirical social scientists, privacy advocates, ethicists, stipulations of sponsoring organizations, etc. To express contextual norms, VACCINE provides a logic for specifying what flows are admissible based on past events. For example, a norm might state "Allow information flows from Alice to Bob with Mary's credit card information, *provided Mary previously gave Alice permission to do so*".

Enforcing norms. For this task, VACCINE operationalizes norms into a set of actionable privacy rules by automatically compiling norms into SQL queries which then check if a given flow is admissible taking into account the history of previously observed flows. To enforce the norms, VACCINE first extracts the respective flows from the email text with the help of the CI Parameter Extractor and then uses the CI Flow Checker to see whether the extracted flows are admissible by running the respective SQL queries.

VACCINE's approach offers intrinsic benefits to the design of DLP systems. First, it achieves *modularity in design* as a result of decoupling the extraction and checking phases in policy enforcement. Second, it increases the *expressiveness of privacy rules* due to its support for CI based norm enforcement. Finally, it supports *obfuscation of sensitive information* through the use of the CI framework that only captures the type of information being conveyed, not the actual data.

In the following, we describe the two key components of the system in more detail. We also briefly describe the temporal logic that we use for expressing the CI norms that are enforced by the CI Flow Checker.

3.1 CI Flow Extractor

The CI flows comprise the CI parameters extracted by the CI parameter extractor. As depicted in Figure 2, the parameter extractor detects the relevant CI parameters associated with each information exchange such as actors (sender, recipient, subject) and the type of information (attribute) and map them onto CI flows. The extractor retrieves the sender and the intended recipients from the underlying protocol's metadata. This is the straightforward bit compared to the task of identifying attributes and the corresponding subjects, which requires a more sophisticated approach. For example, if the policy states that "John's financial information" should not be transferred without his permission, the CI extractor needs to identify all patterns of text that correspond to financial information like, *bank accounts, SSN*, etc. Similarly, for each attribute type mentioned in the privacy policy text, the CI extractor needs to identify the CI subject, in this case *John*. The resulting CI flow will comprise of the extracted parameters:

```
[sender,
        financial information (attribute),
                John (subject),
                                recipient]
```

To tackle the attribute-subject task, we combine a suite of natural language models using recent advances in distant supervision techniques to identify the mentioned attributes and link them to the appropriate subjects. We assume that the relevant CI parameters are retrieved from the policy either manually or with the assistance of an automatic CI parameter detection algorithm [37].

Extracting attributes. The CI extractor needs to identify the instances of the abstract information types included in the rules (policy). For example, if the rule applies to financial information, the extractor needs to detect all possible related instances in the text such as SSN, bank information, credit card number and so forth. This many-to-one mapping makes it difficult to use a single model to detect all instances of a given attribute. To tackle this issue, we adopt the Snorkel [32] framework to generate a large amount of training data using a small number of annotators with weak supervision to increase accuracy. This approach helps to bootstrap our model to annotate CI parameters in emails using a small number of human curated annotators. As Figure 3 shows, we use the labels outputs of multiple models to train a generative annotator choice model. Namely, we import regular expressions corresponding to each attribute[1]. Secondly, we include name entity recognizers which use a dictionary mapping to known types of nouns to identify numeric and non-numeric attributes [23]. Lastly, we use the syntactic structure to identify the nouns associated with subjects. The outputs from each of the models is used to train a simple logistic classifier on these labels. The resulting unsupervised annotator choice model performs better than the majority vote of all annotators by modeling the agreement between various annotators. The high precision, low coverage probabilistic labels provided by the generative model is trained to discriminate by a fully connected feedforward neural network against gold labels annotated by humans on a small set of information flows.

[1]There can be more than one pattern for attribute like "phone number". See http://regexlib.com/Search.aspx?k=phone

Extracting subjects. For this task, similarly to attribute detection, we use the output of a three model mixture to train a generative annotator choice model for subjects. The annotator models set comprise: 1) a heuristic model that matches the entity closest in syntactical proximity to the attribute using a name entity recognizer (NER) 2) a model to detect matching subject-object using syntactic dependency parse trees, 3) a state-of-the-art conversational question answering model (BiDAF) [34] which encodes the email text and a question whose answer is the desired subject of an attribute into an embedding to perform semantic dimensionality reduction. For example, if the attribute detected is a phone number, the question posed to the QA model is "Who should we call?".

3.2 CI Logic

As explained in Section 2.2, a system execution in VACCINE can be abstractly seen as a timed sequence of CI flows. That is, a set of contextual informational norms describes those flow sequences that are allowed in a given privacy context. We next introduce a logic, `TFlow`, for expressing privacy norms that can be automatically converted into actionable rules used to check whether given information flows obey the specified norms.

Syntax. `TFlow` allows the declarative specification of CI flow sequences where the admissibility of a flow in a sequence can depend on whether other flows have or have not been observed. More precisely, `TFlow` is a variant of metric linear temporal logic [3, 15, 22, 40], which has been used to formalize CI-based privacy frameworks in other settings. We here adapt this logic to the specific setting of DLP and identify several important restrictions that enable the efficient checking of these norms in a DLP system. One important restriction is that we allow only temporal operators that can reason about flows that have happened in the past, which makes the logic suitable for both offline and online monitoring. The syntax of the logic is as follows:

$$\phi \in \texttt{TFlow} ::= \phi_1 \textbf{ and } \phi_2 \mid \phi_1 \textbf{ or } \phi_2 \mid \textbf{not } \phi \mid \textbf{exists } x.\ \phi \mid \phi_1 \textbf{ since } \phi_2 \mid \textbf{flow}(x_1, x_2, x_3, x_4) \textbf{ where } P(\vec{x})$$

Atomic formulas $\textbf{flow}(x_1, x_2, x_3, x_4)$ **where** $P(\vec{x})$ are used to describe flows from a sender x_1 to a receiver x_2 concerning attribute x_3 of subject x_4 such that the formula $P(\vec{x})$ holds. Here P is a Boolean combination of basic predicates, such as equality between variables $x \in Var$ and constants $c \in C$. In the case that $P(\vec{x})$ is simply *true*, we omit the **where** clause for convenience. Constants include numbers, strings, and other common data structures encoding agents and attributes of the target CI context. Temporal formulas of the form ϕ_1 **since** ϕ_2 are used to express that at some point in the past the formula ϕ_2 held after which all later time steps satisfied the formula ϕ_1. The semantics of existential quantification (**exists** $x.\phi$), logical conjunction (**and**), disjunction (**or**), and negation (**not**) are as expected.

Contextual norms are typically expressed as `TFlow` formulas of the form

$$\textbf{flow}(snd, rcp, att, sub) \textbf{ and } \phi(snd, rcp, att, sub)$$

where *snd*, *rcp*, *att*, and *sub* refer to the parameters (agents and attribute) of the flow being checked for admissibility, and ϕ is a formula specifying the transmission principle of the norm.

Examples. As a first example, let us consider a simple generic norm $norm_g$ stating that everybody can send their own data to themselves. This norm can be expressed in `TFlow` as

$$norm_g(snd, rcp, att, sub) \stackrel{\text{def}}{\equiv} \textbf{flow}(snd, rcp, att, sub) \textbf{ where } snd = sub \textbf{ and } snd = rcp.$$

As another example, suppose we want to express the $norm_p$ stating that an agent *snd* can share an attribute of an agent *sub* if *sub* has previously given *snd* an explicit permission to do so, and *sub* has not revoked the permission since then. This norm can be formulated in `TFlow` as

$$norm_p(snd, rcp, att, sub) \stackrel{\text{def}}{\equiv} \textbf{flow}(snd, rcp, att, sub) \textbf{ and } ((\textbf{not flow}(sub, snd, \langle -, permission, att\rangle, sub)) \textbf{ since } \textbf{flow}(sub, snd, \langle +, permission, att\rangle, sub)).$$

Here, we use the term $\langle +, permission, att\rangle$ to indicate that permission of sharing *att* has been granted, and similarly $\langle -, permission, att\rangle$ to indicate that this permission has been revoked.

We point out that existing DLP systems cannot handle the second norm because it involves reasoning about flows that occurred in the past. It is also instructive to point out that the above formulation of the norms does not depend or involve lower-level details of communication exchanges, thereby achieving the level of abstraction needed for separating the checking phase from the extraction phase.

Formal semantics. The formal semantics of `TFlow` formulas can be defined in terms of timed sequences of flows. In this work, we take an equivalent approach where a flow sequence is modeled as a table of flows where each row in the table contains the arguments of one flow in the sequence together with its timestamp indicating when the associated flow occurred. Hence, the flow sequence can be reconstructed from the flow table using this timestamp information.

The satisfiability of a `TFlow` formula ϕ is defined by a satisfiability relation $T, \Gamma, i \models \phi$ which states that ϕ holds at time point i in the flow trace given by flow table T and subject to the variable assignment Γ, which maps the free variables of ϕ to values. For instance, here are the formal definitions of the satisfaction relation for the two most important constructs in the logic:

$$T, \Gamma, i \models \textbf{flow}(x, y, z, w) \textbf{ where } P(\vec{x}) \quad \text{iff } f \in T(i) \text{ and } \Gamma[x \mapsto f(1), y \mapsto f(2), z \mapsto f(3), w \mapsto f(4)](\vec{x}) \in I(P)$$

$$T, \Gamma, i \models \phi_1 \textbf{ since } \phi_2 \quad \text{iff } T, \Gamma, j \models \phi_2 \text{ or } i > 0 \text{ and } T, \Gamma, i \models \phi_1 \text{ and } T, \Gamma, i-1 \models \phi_1 \textbf{ since } \phi_2$$

In the above, we use $T(i)$ to denote the flows in T with the timestamp i. Similar notation is used for accessing constituent elements of a flow f. Lastly, we assume that the function I provides the interpretation for basic predicates expressing constraints on variables

and constants. We omit the full details of the semantics as it is standard [15, 22]. We note that the above semantics directly supports extensions of TFlow to metric temporal logics where formulas can explicitly refer to time points.

3.3 CI Flow Checker

We have implemented our flow checker on top of an SQL database. The database maintains a table FH of all the flows that have been accepted so far. Checking whether a new flow f obeys the norms expressed in TFlow amounts to checking if FH extended with f satisfies any of the norms. Each norm $\phi(\vec{x})$ is automatically compiled to an SQL query that produces a table with one column per free variable in $\vec{x}$ as well as a timestamp column. This table contains one row per timestamp i and valuation Γ for $\vec{x}$ such that $\mathsf{FH}, \Gamma, i \models \phi$ holds. Thus, if the query for any of the norms contains an entry for the current timestamp i, then the new flow is accepted and permanently added to FH. Otherwise, it is rejected.

The algorithm for generating the SQL queries is defined recursively on the structure of formulas and closely follows from the semantics of TFlow. However, a naive encoding into SQL yields queries that do not scale well, in particular if we must deal with nested **since** formulas and unconstrained negation. We therefore discuss several restrictions and optimizations to a naive encoding that considerably improves the performance.

Syntactic restrictions. In order to support the automatic compilation of TFlow formulas into efficient SQL queries, we impose some restrictions on TFlow. First, the free variables in the subformulas of disjunctions have to be identical. Further, all variable must occur as a parameter in some **flow** predicate. Finally, we require that every negative formula **not** ϕ_1 that occurs in a positive context is guarded by a positive formula ϕ_2. A formula is positive if it contains no unguarded negative formulas and a formula occurs in a positive context if it occurs under an even number of **not** operators. A negative formula **not** ϕ_1 is guarded by ϕ_2 if these formulas occur either as (**not** ϕ_1) **and** ϕ_2 or as (**not** ϕ_1) **since** ϕ_2 where all the free variables of ϕ_1 also occur free in ϕ_2.

For instance, in the norm $norm_p$ given in § 3.2, the negative formula (**not** **flow**$(sub, snd, \langle -, permission, att\rangle, sub)$) is guarded by the positive formula **flow**$(sub, snd, \langle +, permission, att\rangle, sub)$ below the **since** operator. This restriction of the logic to a *guarded fragment* is particularly important because it ensures that negative subformulas can only be used to prune the result tables obtained from positive subformulas. This restriction enables us to cache the evaluation of subformulas in dedicated tables which can then be updated incrementally using simple queries as new flows are added to the history table FH. We describe this process in more detail below. Without this restriction, the evaluation of temporal formulas involving the **since** operator cannot be done efficiently (i.e. without reevaluating the entire history of flows each time the norm is checked). For otherwise, to enable caching we would have to know in advance the entire universe of values for agents and attributes that may appear in a future flow. We found that these syntactic restriction did not affect the expressivity of the logic in practice.

Compiling norms to efficient SQL. Our optimized compilation process associates a unique SQL table T_ϕ with every subformula of a norm that needs to be checked. These tables cache the satisfaction relation for ϕ with respect to the current history table FH. Thus, when we check a norm ϕ, we can simply query the associated cache table T_ϕ directly. These cache tables are updated whenever a new flow is added to the history table and time advances to a new timestamp. The updates are computed recursively according to the structure of the subformulas.

For instance, suppose ϕ is of the form (ϕ_1 **since** ϕ_2) and we extend the history table FH with a new incoming flow f at time i. Then we first update the caches for the subformulas ϕ_1 and ϕ_2 recursively to account for the update. To update the cache for ϕ itself, note that according to the semantics of **since**, FH extended with f satisfies ϕ at timestamp i iff either ϕ_2 is satisfied at i or ϕ_1 is satisfied at i and ϕ already holds at the previous timestamp i'. In effect, this means that the table for ϕ can be updated as $T_\phi \triangleq T_\phi \cup T_{\phi_2}(i) \cup (\pi_t(T_\phi(i')) \circ T_{\phi_1}(i))$. Here, $\circ$ stands for inner join (on the common columns), and π_t projects out the timestamp column in a given table. The cache table for an atomic formula $\phi \triangleq$ **flow**(x, y, z, w) **where** $P(\vec{x})$ is updated with the results of the following generic SQL query:

```
SELECT snd AS x, rcp AS y, att AS z, sub AS w
FROM FH WHERE P(x⃗)
```

Here, we assume the case where $\vec{x} = (x, y, z, w)$. If P has free variables other than x, y, z, w, these variables are additionally added as columns in the SELECT clause. Those columns are then populated with all possible values in FH using appropriate subqueries. The syntactic restrictions on free variables in norm formulas ensure that this encoding is both sound and complete.

4 EVALUATION

In this Section, we present results of evaluating the key components of VACCINE in a corporate setting. We first evaluated the CI parameter extractors that form the basis of the combined CI extractor individually on the emails from the Jeb Bush Presidential campaign. We then present an end-to-end case study of data leakage protection using VACCINE on the corpus of emails associated with the Enron corporation [9, 36]. We detail the accuracy of VACCINE parameter extraction techniques on the Enron email dataset, the extraction of contextual norms (expressed in TFlow) for the Enron corporate privacy context, performance numbers on evaluating the norms using the VACCINE SQL-based flow checker, and the benefits of using VACCINE for real-world data leakage protection.

4.1 Extraction Microbenchmark

We compared the efficiency of the CI flow extraction against manually annotated emails from the Jeb Bush Presidential campaign [2]. We manually annotated 436 attributes and 72 subject parameters. Table 1 summarizes the performance of the components used in the Snorkel based CI parameter extraction. Some of the failures of the model can be attributed to genuinely ambiguous natural language in the email. A typical example is the following sentence: *"Can you tell me if you have plans to amend the drivers license laws, so that as I, Florida homeowner from the UK, can have a drivers license for*

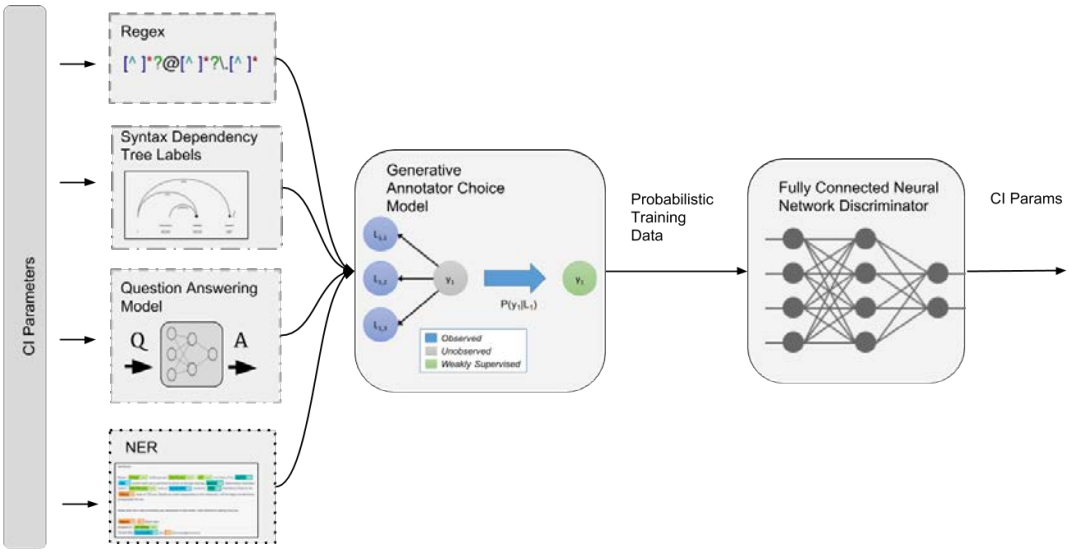


Figure 3: CI Parameters Extractor. We combine outputs from multiple models to train a generative annotator choice model, which we subsequently use to train a fully-connected neural network discriminator for extracting relevant CI parameters.

6 years as I used to have". Here, the manually annotated subject was *"I"*, but the model's output was *"Florida"*. Similarly, for attribute detection, many of the failures are due to the jargons used within an organization which can be confused for affiliations. This makes the case for continuous improvement of the model through active learning once a system like VACCINE is deployed to keep up with the evolving language patterns.

Table 1: Benchmark of CI parameter extraction from Jeb Bush's emails

Model Type	Parameter Type	F1 Accuracy (%)
Regex	Attribute	71.56
NER	Subject	56.94
QA	Subject	61.11
QA+Query Rewrite	Subject	63.88

4.2 Enron Case Study

We have evaluated VACCINE's DLP system performance for offline auditing of email communication. For this experiment, we used the Enron corporation email corpus [9, 36]. Specifically, we aimed to answer the following questions:

(1) How effective is the VACCINE framework in preventing potentially unauthorized flows in a real-world emulated context?
(2) How efficiently can CI flows be checked against the privacy rules automatically generated from `TFlow` norm descriptions?

We chose the Enron corporation for our enterprise context since both its code of ethics [43] as well its email dataset are publicly available [9, 36]. From Enron's code of ethics we extracted CI parameters for an enterprise context. We identified 145 agents with 12 company roles, such as director, manager, and regular employee. We assigned roles to employees, identified by their email addresses, using publicly available data sources [18, 41]. We also identified 13 attributes, such as social security number, visa number, address, etc. Our focus was on emails containing the exchange of sensitive information such as passwords, bank account information, phone numbers, etc. We thus identified 73 emails involving exchange of such information and extracted 87 corresponding CI flows. We used these flows to evaluate the effectiveness and efficiency of VACCINE. All of the experiments were carried out on an `Intel(R) Xeon(R)` machine with eight 3.6GHz cores.

4.2.1 Norm Extraction. We identified 40 Enron contextual norms that focus on access and disclosure of personal identifiable information (PII) in a corporate setting. The norm creation process was conducted over the available documentation of Enron's code of ethics and its organizational structure. Due to the lack of a publicly available (Enron) enterprise privacy handbook that specifies precise norms, we created norms to enforce a simulated *ethical wall* on email communication between Enron employees. An ethical wall [29] refers to establishing an environment that will prevent information exchange between specific departments within the organization to avoid any conflicts of interest. Our norm extraction process focused on each type of PII and how agents in the various enterprise roles can exchange the corresponding attributes. A typical example of a norm is: *Enron employees can send their social security number to administrative managers.* This norm in our policy language `TFlow` is expressed as

$$norm_{SSN}(s,r,a,b) \stackrel{\text{def}}{\equiv} \textbf{flow}(s,r,a,b) \textbf{ where } s = b \wedge a = \text{“}SSN\text{”} \textbf{ and } in_role(s, \text{“}employee\text{”}) \textbf{ and } in_role(r, \text{“}manager\text{”}).$$

Here, the *in_role* predicate is defined using the **since** construct in a similar fashion to the predicate of having a permission in Section 3.2. In contrast, employees are allowed to send their less sensitive information, such as phone number and addresses to anyone without restrictions. This norm is captured in `TFlow` as

$$norm_{PII}(s,r,a,b) \stackrel{\text{def}}{\equiv} \textbf{flow}(s,r,a,b) \textbf{ where } s = b \wedge (a = \text{“}phone_number\text{”} \vee a = \text{“}address\text{”}).$$

Also, the distribution of information belonging to other Enron employees can be done if the subject employee gave a permission for the disclosure of such information. This norm specified in `TFlow` is essentially the norm $norm_g$ introduced in Section 3.2.

4.2.2 Identifying Contextual Information. To evaluate the extraction of CI parameters, we manually annotated our corpus of selected emails (73) with attributes such as phone number and conference call codes and CI subjects corresponding to them. As a baseline accuracy for extracting relevant attributes without taking any contextual information into account, we used regular expression based models (regex) for detecting attributes. These models are similar to the ones currently available in tools like Google DLP and Microsoft Exchange.

Results Table 2 shows that by aggregating multiple regular expressions using a distant supervised framework called Snorkel [32], gives an improvement of 4% points in F1 accuracy. For subjects detection, it is close to 1% F1 accuracy. Combining results from the QA model using the “Who should we call?” question with other NER based models increases the accuracy by almost 4%. Narrowing context by posing a question that includes an attribute, e.g., "Whose phone number is 123-456-7890?", leads to increased accuracy by a further 9%.

Table 2: Accuracy of CI parameter extraction from Enron emails

Model Type	Parameter Type	F1 Accuracy (%)
Regex	Attribute	90.20
Snorkel	Attribute	**94.80**
NER	Subject	48.35
QA	Subject	67.12
Snorkel	Subject	71.83
QA+Query Rewrite	Subject	**80.82**

4.2.3 Leakage Detection Performance. The purpose of our next experiment was to measure the general performance of checking flows against the extracted norms. First, we automatically compiled the extracted Enron norms to SQL queries that we then evaluated against approximately 6700 Enron flows, consisting of our manually extracted flows and initial flows used to set up the organizational structure of the company. For instance, the fact that agent a is an employee is modeled as the flow

$$(enron_inc, enron_inc, \langle +, in_role, employee\rangle, a).$$

Intuitively, this flow encodes that a has been previously given a role of a regular employee in the Enron corporation. We also use the initial flows to incorporate reasonable assumptions about disclosure privileges for PII information, say, that within each department (i.e., role), employees can disclose each other’s phone numbers.

Our implementation of the SQL flow checking engine follows the high-level algorithm introduced in Section 3.3 except that we cache only since formulas in auxiliary tables. We only maintain the rows with the latest timestamp in cache tables as they are the only once needed for proper updating in the future. We use Sqlite3 as our relational database. Lastly, we compressed 40 Enron norms into 12 succinct ones as many of them shared the same structure except that they differ slightly in the specific **where** conditions. For instance, consider a norm stating that managers can disclose employee phone numbers to CEOs and a norm stating that managers can disclose employee phone numbers to directors. We compress these two norms into a single one where managers can disclose employee phone numbers to either CEOs or directors.

Results The performance times are reported in Table 3. We provide measurements for flow checking in general and its breakdown subject to accepted and blocked flows. We also provide information on time spent performing cache updates while checking flows. As it can be seen, average time spent analyzing a flow is reasonable. We point out that rejecting a flow takes more time than accepting it. This is because we implement a form of short-circuit evaluation of the norms where a flow is accepted as soon as we encounter a norm that is satisfied. The remaining norms then no longer need to be evaluated. We note that the initial flows for setting up the organizational structure are always accepted, accounting for the high number of accepted flows.

Table 3: Performance times (in seconds) for SQL checking of flows against the Enron norms

	total	avg	med	min	max	sd
flows (6706)	184	0.027	0.027	0.004	0.054	0.012
accepted (6658)	181	0.027	0.026	0.004	0.053	0.012
blocked (48)	2.54	0.052	0.052	0.051	0.054	0.001
cache upd.	95	0.007	0.0003	0.0002	0.0001	0.008

We also measured how flow checking times vary depending on how the history flow table grows. The results are shown in Figure 4. Flow checking times grow linearly with the size of the table, i.e., the number of accepted flows in the past. The outliers in the graph mostly correspond to the checking of flows that occur at the end of the experiment where we check the flows extracted from the Enron emails containing sensitive information. On the other hand, initial flows are always accepted by exactly the same norm, so the performance of analyzing such flows is more stable. The reported numbers can be improved by employing more sophisticated algorithms used in work on runtime policy monitoring [15, 22].

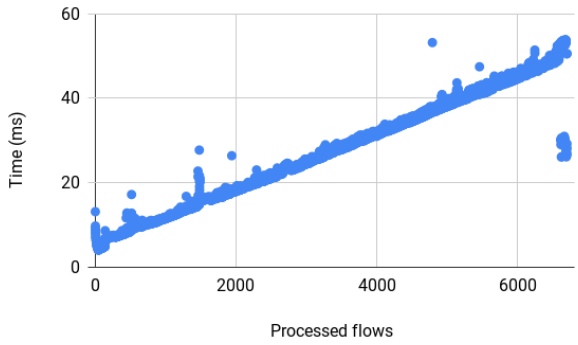


Figure 4: Flow checking performance (in miliseconds) through time

4.2.4 Detecting Information Leakages. The flows rejected in the previous experiment constitute data leakages subject to the identified contextual Enron norms. We note that the flow checking experiment does not suffer from false positives or false negatives. This is due to the fact that the flows used in the experiment were manually extracted and SQL queries implementing the norms are sound by construction (as described in Section 3.3).

In our next experiment, we have analyzed the nature of emails from which the rejected flows were extracted.

Results Arguably, the most interesting case of rejected flows involves visa information. VACCINE rejected a flow between two Enron employees where the sender was sharing visa information about a third party subject. More precisely, the sender was sharing the type of the visa in question, its expiration date, and the information that the subject was granted multiple entries to the United States. To the best of our understanding, the subject had not previously granted the sender with a permission to share such confidential information.

The vast majority (44) of rejected flows involves distribution of passcodes/passwords and phone numbers for company conference calls. Our implementation of VACCINE rejected such flows as we did not extract or create any contextual norms governing the distribution of conference call metadata. In effect, VACCINE conservatively treats such flows as data leakages. VACCINE also prevented a flow where a third party resumé was shared between two Enron employees. Our current norms only allow agents to share their own resumes. Lastly, our implementation also blocked a flow between a third party and one of the higher-ranked Enron employees where the flow subjects were Enron bank accounts. Although careful inspection of the corresponding email did not show any actual leakage of sensitive company information, our norms were deliberately designed to contain the communication exchanges involving confidential company data within Enron.

These experiments show that VACCINE provides an effective tool for data leakage prevention, modulo an effective flow extractor.

4.2.5 Complexity of Email Communication Exchanges. In our last experiment, we measured how many CI flows are typically generated in an email conversation. We note that a single email can generate multiple flows between different participants. This is because an email often quotes other emails that have been sent earlier in the same conversation. In such cases, the new email also generates *transitive flows* involving subjects and attributes from the quoted emails to the new recipients.

Table 4: Statistics for each of the 2000 Enron email conversations

	Total	Min	Avg	Max
Emails	78,386	1	39	15,555
Participants	11,609	1	17	2,428
Threads	4,082	1	2	196
Flows	218,070	1	109	85,280

We grouped the emails in the Enron data set into conversations. We then defined an email conversation as a set of emails with the same email subject (modulo reply and forwarding prefixes, etc.). We extracted from each email a set of quadruples (one per recipient) consisting of the sender email address, a recipient email address, the time the email was sent, and the recipient type which can be `TO`, `CC`, and `BCC`, omitting the email content and other metadata. Here is an example of such a tuple:

```
(tracy.geaccone@enron.com,
 sarah.taylor@enron.com,
                2001-11-27 12:48:05,TO)
```

We identified every unique email address with a unique participant. For emails sent to mailing lists, we simply simulated broadcasting the email to all of the active conversation participants.

Out of the whole Enron email data set, we randomly chose 2,000 email conversations involving approximately 80,000 emails and 11,500 participants in total, as shown in Table 4 (first and second row). We used the email time and recipient type fields to recreate email threads for each conversation. There are two email threads per conversation on average, as shown in Table 4 (third row). Due to some missing information in the data set, we observed cases where participants sent emails in the middle of the conversation although they never received any emails in the conversation up to that point. In our analysis, such emails start a new conversation thread, which explains multiple threads per conversation. We used the extracted email threads to generate CI information flows. To model the worst case scenario in terms of the information flow complexity, whenever an email is sent by some sender in a conversation, we assume the sender creates new content and also transitively sends all of the emails in the conversation up to that point. Also, a CI flow has a subject and attribute. For the purposes of this experiment, the actual subject and attribute values are not important. For transitive flows, however, the subject is the sender of the corresponding email that is being quoted.

Results Given the above dataset, we measure how many flows an email yields in practice. As it can be already inferred from Table 4, a single email yields 2.78 flows on average. Moreover, Figure 5 shows the frequency distribution of emails depending on the number of flows they generate. It can be seen that the flow-per-email ratio is in general low, reaching 15 at maximum. This suggests that the load on a DLP (auditing) system created using VACCINE will be tolerable in practice. This is further evidenced by Figure 6 showing a linear increase in the number of generated flows subject to a total number of thread emails.

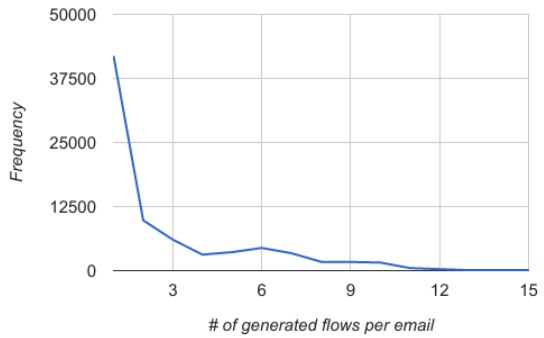


Figure 5: Email frequency distribution based on the number of generated flows per email

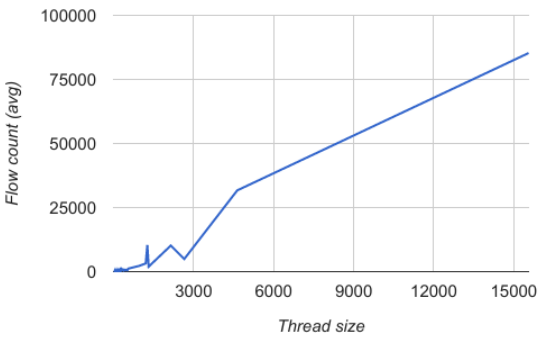


Figure 6: Average number of generated flows per size of the conversation thread

5 RELATED WORK

We already discussed how VACCINE compares to existing commercial DLP solutions in Section 2. The research community has explored sophisticated email DLP methods mostly relying on a combination of machine learning and related classification techniques. These approaches roughly fall under content- and behavior-driven methodologies [35]. Content-based approaches [10, 11, 21, 25] determine the sensitivity of a message based on textual content analysis, either through keyword, regex pattern matching, or machine learning techniques. Behavior-based approaches [24, 38, 45] analyze the organizational structure, common roles and duties, as well as past interaction to identify "normal" behavior patterns and use them to detect unlikely information flows between a pair of sender and recipient. Our work is largely orthogonal to these approaches. While prior solutions explored advancing techniques in identifying insensitive data and malicious behavior patterns, VACCINE instead focuses on providing a robust way to express, administer, and enforce complex DLP policies.

Early work by Armour et.al., looked at designing a privacy compliance engine for emails [4, 7] to prevent privacy violations in large enterprise settings. Similar to our system, their engine comprises an information extraction and violation detection module. The information extraction module relied on an enterprise domain ontology, part-of-speech and semantic tagging to find relevant information in the email text. While the work positions itself as a privacy compliance tool, it does not go beyond facilitating role-based access control to information. Namely, the implicit assumption here is that roles and ownership determine the right to access. In particular, it does not support complex policies involving e.g. notions of consent that require reasoning about the history of observed emails. Furthermore, VACCINE takes advantage of the advances in NLP techniques since this pioneering work. We are not aware of other work that took the ideas in [4, 7] and developed them beyond the preliminary proposal explored there.

Related Work using CI. Defining privacy [30] in terms of CI is now a well-established approach in the privacy and computer science research communities. It is used to describe contextual informational norms, to detect infractions of these norms, and in approaches to accountability and enforcement [5, 17]. Much of the prior work in this space has focused on sophisticated formal logics for describing CI norms that involve complex temporal properties of sequences of information flows such as those involved in HIPAA [6, 14, 16, 19]. We build on these earlier works and adapt them to the specific needs of the DLP setting. Our observation is that in this setting, a simpler logical framework is sufficient to express the relevant norms. With these restrictions, norms can be automatically compiled into SQL queries. The encoding into SQL enables the use of our system in applications where both robustness and efficiency of the DLP engine are important factors.

Compliance-based Approaches. Numerous approaches have been invented to map system constraints to privacy policies that are generated external to these systems. Two examples, are due to Sen, et al. [33] and Breaux and Anton [8]. Sen, et al. have developed LEGALEASE, a language with precise semantics that enables enterprises to express privacy policies against which practices can be checked for compliance. With similar goals, Breaux and Anton have proposed a methodology for extracting implementable software requirements from privacy and security regulatory requirements expressed in legal language that is sometimes vague and ambiguous. The intention in both systems is similar to our own in that it seeks to translate privacy requirements expressed in natural language (assuming this covers legal language) into formally expressed rules that allow for compliance checking. The threat scenarios, however, are quite different.

6 CONCLUSIONS

We presented a new design methodology for building DLP systems based on the theory of Contextual Integrity that offers more precise and modular detection of information flow violations compared to traditional DLP systems.

Using our methodology, we built VACCINE, a system for auditing email communication corpora. VACCINE supports a powerful declarative language for specifying expressive privacy policies that are automatically compiled to sound-by-construction actionable rules, implemented as SQL queries. Furthermore, VACCINE

leverages advanced NLP techniques to efficiently extract relevant information omitted by the existing DLP systems.

In our evaluation, we used VACCINE to audit the Enron emails corpus and achieved significant improvement compared to the methods used by state-of-the-art systems.

7 ACKNOWLEDGEMENTS

We thank Paula Kift and Schrasing Tong for their help in the initial stage of this work. This work is supported by the following National Science Foundation (NSF) under grants CCF-1350574, CNS-1514422, CNS-1801501, CNS-1704527, SES-1537324, SES-1650589, the National Security Agency grant H98230-18-D-006 and the Cisco CG-653005 Research Award. We are also grateful to the NSF I-CORPS 1650769 grant that allowed us to discuss the data leakage problem with large number of companies and ultimately helped us with articulating the VACCINE problem.

REFERENCES

[1] 2018. General Data Protection Regulation. https://gdpr-info.eu/issues/consent/.
[2] 2018. Jeb BushâĂŹs Gubernatorial Email Archive. http://americanbridgepac.org/jeb-bushs-gubernatorial-email-archive/.
[3] Rajeev Alur and Thomas A. Henzinger. 1993. *Information and Computation* 104, 1 (1993), 35–77. https://doi.org/10.1006/inco.1993.1025
[4] Quintin Armour, William Elazmeh, Nour El-Kadri, Nathalie Japkowicz, and Stan Matwin. 2005. Privacy compliance enforcement in email. In *Conference of the Canadian Society for Computational Studies of Intelligence*. Springer, 194–204.
[5] Adam Barth, Anupam Datta, et al. 2006. Privacy and contextual integrity: Framework and applications. In *Proceedings of the Symposium on Security and Privacy*. IEEE, 15–pp.
[6] David Basin, Felix Klaedtke, and Samuel Müller. 2010. Monitoring security policies with metric first-order temporal logic. In *Proceedings of the 15th ACM symposium on Access control models and technologies*. ACM, 23–34.
[7] Narjes Boufaden, William Elazmeh, Yimin Ma, Stan Matwin, Nour El-Kadri, and Nathalie Japkowicz. 2005. PEEP-An Information Extraction base approach for Privacy Protection in Email. In *Proceeding of the Second Conference on Email and Anti-Spam*.
[8] Travis D Breaux, Annie Antón, et al. 2008. Analyzing regulatory rules for privacy and security requirements. *Software Engineering, IEEE Transactions on* 34, 1 (2008), 5–20.
[9] Carnegie Mellon University. 2016. Enron Email Dataset. https://www.cs.cmu.edu/~./enron/.
[10] Vitor R Carvalho, Ramnath Balasubramanyan, and William W Cohen. 2009. Information leaks and suggestions: A case study using mozilla thunderbird. In *Sixth Conference on Email and Anti-Spam (CEAS)*.
[11] Vitor R Carvalho and William W Cohen. 2007. Preventing information leaks in email. In *International Conference on Data Mining*. SIAM, 68–77.
[12] Checkpoint. 2017. Data Loss Prevention Software Blade. https://www.checkpoint.com/products/dlp-software-blade/.
[13] Danqi Chen and Christopher Manning. 2014. A Fast and Accurate Dependency Parser using Neural Networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 740–750. https://doi.org/10.3115/v1/D14-1082
[14] Omar Chowdhury, Andreas Gampe, et al. 2013. Privacy promises that can be kept: A policy analysis method with application to the HIPAA privacy rule. In *Proceedings of the 18th SACMAT*. ACM.
[15] Omar Chowdhury, Limin Jia, Deepak Garg, and Anupam Datta. 2014. Temporal Mode-Checking for Runtime Monitoring of Privacy Policies. In *Computer Aided Verification - 26th International Conference, CAV 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 18-22, 2014*. 131–149. https://doi.org/10.1007/978-3-319-08867-9_9
[16] Omar Chowdhury, Limin Jia, Deepak Garg, and Anupam Datta. 2014. Temporal mode-checking for runtime monitoring of privacy policies. In *International Conference on Computer Aided Verification*. Springer, 131–149.
[17] Natalia Criado and Jose M Such. 2015. Implicit Contextual Integrity in Online Social Networks. *Information Sciences: an International Journal* 325 (2015), 48–69.
[18] Andrew Fiore and Jeff Heer. 2004. UC Berkeley Enron email analysis. http://bailando.sims.berkeley.edu/enron_email.html.
[19] Deepak Garg, Limin Jia, and Anupam Datta. 2011. Policy auditing over incomplete logs: theory, implementation and applications. In *Proceedings of the 18th ACM conference on Computer and communications security*. ACM, 151–162.
[20] Google. 2016. Scan your email traffic using data loss prevention. https://support.google.com/a/answer/6280516?hl=en.
[21] Tanya Goyal, Sanket Mehta, and Balaji Vasan Srinivasan. 2017. Preventing Inadvertent Information Disclosures via Automatic Security Policies. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 173–185.
[22] Hsi-Ming Ho, Joël Ouaknine, and James Worrell. 2014. Online Monitoring of Metric Temporal Logic. In *Proceedings of the Runtime Verification - 5th International Conference*. 178–192. https://doi.org/10.1007/978-3-319-11164-3_15
[23] Matthew Honnibal and Ines Montani. 2018. SpaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. https://spacy.io.
[24] Chetan Kalyan and Krithika Chandrasekaran. 2007. Information leak detection in financial e-mails using mail pattern analysis under partial information. In *Proceedings of the 7th WSEAS International Conference on Applied Informatics and Communications (AIC)*. Citeseer, 104–109.
[25] Gilad Katz, Yuval Elovici, and Bracha Shapira. 2014. CoBAn: A context based model for data leakage prevention. *Information sciences* 262 (2014), 137–158.
[26] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. *CoRR* abs/1603.01360 (2016). arXiv:1603.01360 http://arxiv.org/abs/1603.01360
[27] McAfee. 2017. Total Protection for Data Loss Prevention (DLP). https://www.mcafee.com/us/products/total-protection-for-data-loss-prevention.aspx.
[28] Microsoft. 2017. Data loss prevention in Exchange 2016. https://support.google.com/a/answer/6280516?hl=en.
[29] Microsoft. 2017. Understanding Ethical Walls. https://technet.microsoft.com/en-us/library/aa996850(v=exchg.141).aspx.
[30] Helen Nissenbaum. 2010. *Privacy in Context: Technology, Policy, and the Integrity of Social Life*. Stanford Law Books.
[31] Richard A Posner. 1977. Right of privacy, the. *Ga. L. Rev.* 12 (1977).
[32] Alexander Ratner, Stephen H. Bach, Henry R. Ehrenberg, Jason Alan Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid Training Data Creation with Weak Supervision. *CoRR* abs/1711.10160 (2017). arXiv:1711.10160 http://arxiv.org/abs/1711.10160
[33] Satyaki Sen, Saikat Guha, et al. 2014. Bootstrapping privacy compliance in big data systems. In *Proceedings of the Symposium on Security and Privacy*. IEEE.
[34] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional Attention Flow for Machine Comprehension. *CoRR* abs/1611.01603 (2016).
[35] Asaf Shabtai, Yuval Elovici, and Lior Rokach. 2012. *Data Leakage Detection/Prevention Solutions*. Springer. 17–37 pages.
[36] Jitesh Shetty and Jafar Adibi. 2004. The Enron email dataset database schema and brief statistical report. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.296.9477.
[37] Yan Shvartzshanider, Ananth Balashankar, Thomas Wies, and Lakshminarayanan Subramanian. 2018. RECIPE: Applying Open Domain Question Answering to Privacy Policies. In *Proceedings of the Workshop on Machine Reading for Question Answering*. 71–77.
[38] Salvatore J Stolfo, Shlomo Hershkop, Chia-Wei Hu, Wei-Jen Li, Olivier Nimeskern, and Ke Wang. 2006. Behavior-based modeling and its application to email analysis. *Transactions on Internet Technology (TOIT)* 6, 2 (2006), 187–221.
[39] Symantec. 2017. Data Loss Prevention. https://www.symantec.com/products/information-protection/data-loss-prevention.
[40] Prasanna Thati and Grigore Rosu. 2005. Monitoring Algorithms for Metric Temporal Logic Specifications. *Electronic Notes in Theoretical Computer Science* 113 (2005), 145–162. https://doi.org/10.1016/j.entcs.2004.01.029
[41] The University of Edinburgh. 2017. http://www.inf.ed.ac.uk/teaching/courses/tts/assessed/roles.txt.
[42] US Department of Education. 2016. Family Educational Rights and Privacy Act (FERPA). https://www2.ed.gov/policy/gen/guid/fpco/ferpa/index.html.
[43] Curtis C Verschoor. 2002. It isn't enough to just have a code of ethics.(Ethics). *Strategic Finance* 84, 6 (2002), 22–24.
[44] F Westin Alan. 1967. *Privacy and freedom*. Atheneum. New York.
[45] Polina Zilberman, Shlomi Dolev, Gilad Katz, Yuval Elovici, and Asaf Shabtai. 2011. Analyzing group communication for preventing data leakage via email. In *International Conference on Intelligence and Security Informatics (ISI)*. IEEE, 37–41.

Shapley Meets Uniform: An Axiomatic Framework for Attribution in Online Advertising

Raghav Singal
Columbia University
rs3566@columbia.edu

Omar Besbes
Columbia University
ob2105@gsb.columbia.edu

Antoine Desir
INSEAD
antoine.desir@insead.edu

Vineet Goyal
Columbia University
vg2277@columbia.edu

Garud Iyengar
Columbia University
garud@ieor.columbia.edu

ABSTRACT

One of the central challenges in online advertising is attribution, namely, assessing the contribution of individual advertiser actions including emails, display ads and search ads to eventual conversion. Several heuristics are used for attribution in practice; however, there is no formal justification for them and many of these fail even in simple canonical settings. The main contribution in this work is to develop an axiomatic framework for attribution in online advertising. In particular, we consider a Markovian model for the user journey through the conversion funnel, in which ad actions may have disparate impacts at different stages. We propose a novel attribution metric, that we refer to as *counterfactual adjusted Shapley value*, which inherits the desirable properties of the traditional Shapley value. Furthermore, we establish that this metric coincides with an adjusted "unique-uniform" attribution scheme. This scheme is efficiently computable and implementable and can be interpreted as a correction to the commonly used uniform attribution scheme.

CCS CONCEPTS

• **Information systems** → **Online advertising**; • **Mathematics of computing** → *Markov processes*; • **Applied computing** → *Electronic commerce*; *Operations research*.

KEYWORDS

Digital economy, online advertising, attribution, Markov chain, Shapley value, causality

ACM Reference Format:
Raghav Singal, Omar Besbes, Antoine Desir, Vineet Goyal, and Garud Iyengar. 2019. Shapley Meets Uniform: An Axiomatic Framework for Attribution in Online Advertising. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313731

1 INTRODUCTION

With the rise of the Internet, the digital economy has become a trillion dollar industry accounting for over 6% of the U.S. GDP [19]. A multitude of retailers reach consumers through online sources with the goal of acquiring new customers and managing relationships with the existing ones. Such wide range of economic activities taking place in the digital world has enabled data collection at a massive scale, allowing retailers to better understand customer behavior and enhance service quality using data-driven decisions. With over a billion people with access to the Internet, it is not surprising that advertising has moved to the digital space. The global digital marketing sector witnessed a growth of 21% in 2017, which increased its market size to USD 88 billion [44]. Key decisions in this space include allocating budget to various ad channels and media (e-mail, display media platforms, and paid search for instance) and bidding for ads to push one's product towards the desired customer demographic at the right time. Such problems require the retailer to understand the value of showing an ad via a specific channel at a given time. How can an advertiser assign or *attribute* value to the advertising actions taken across the different channels and media? *Attribution* is one of the central questions in online advertising today. The value of each channel is an important input to media mix optimization, helps build an understanding of the customer journey, and also helps a company to justify its marketing spend [38, 45]. Incorrect understanding of the effectiveness of online channels can result in sub-optimal budget allocation, possibly resulting in lost revenue. Recognizing the importance of this problem, the Marketing Science Institute identifies attribution as the topmost research priority for the period 2016-2018 [23]. Though there exist multiple articles in this domain, there lacks a systematic approach that is both theoretically-sound and tractable.

Attribution is inextricably linked to causality since it involves quantifying the added value of showing an ad over the baseline value of what *would have* happened if no ad was shown (*counterfactual*). In this work, we take the following view of causality, which comes from the pioneering work of Rubin [41]:

> *"Intuitively, the causal effect of one treatment, E, over another, C, for a particular unit and an interval of time from t_1 to t_2 is the difference between what would have happened at time t_2 if the unit had been exposed to E initiated at t_1 and what would have happened at t_2 if the unit had been exposed to C initiated at t_1."*

In the context of digital advertising, treatments E and C are the advertiser showing, and not showing an ad, respectively. Accordingly, attribution involves capturing the causal effect of an ad where the baseline corresponds to not showing an ad.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313731

Causality is an active research area, and there exist paradigms for capturing the causal effect of a treatment. See, e.g. Chockler and Halpern [11], Collins et al. [13], Eells [16], Halpern and Pearl [20], Hitchcock [21], Hume [22], Morgan and Winship [35], Pearl [37] and Rubin [41]. We comment on some of these alternative approaches in our concluding remarks.

As with the existing work in attribution, we also consider the setting where an advertiser is interested in understanding the contributions of various ads to a single product they are promoting. Even with a single product, attribution is a challenging problem. It involves distributing the value generated by a network of actions to each individual action. Such a network might have a mix of interaction effects that one needs to account for when decomposing the network value. Capturing such (possibly non-linear) interaction effects is quite non-trivial.

1.1 Related literature

There exists a large and diverse literature on the attribution problem in online advertising. We present a brief overview of existing approaches and refer the reader to existing surveys [12, 28] for a more complete treatment. Attribution methodologies can be classified into two broad classes: *rule-based* or *algorithmic*. We discuss both the classifications below.

1.1.1 ***Rule-based heuristics.*** Rule-based heuristics include approaches such as *last touch attribution* (LTA), *uniform weights*, and *customized weights* [6, 38–40]. In LTA, an advertiser allocates all the value generated by a user to the last ad directed at her whereas under a uniform weights scheme, all the ads are allocated an equal credit. Ads receive tailored weights under a custom weights scheme. Although such heuristics are transparent and tractable, there is no rationale justifying their appropriateness as a measure for attribution. Metrics such as LTA appear unfair since they do not value the contribution of channels that build product awareness. Uniform or customized weights might appear to be a fix but there is no a priori reason to believe that attribution should be linear.

1.1.2 ***Algorithmic approaches.*** The algorithmic approaches can be classified as using either *incremental value heuristic (IVH)* (or *removal effect*) or *Shapley value (SV)* as a measure for attribution.

IVH. IVH computes the change in the eventual *conversion*[1] probability of a user when a specific ad is removed from her path. This is the most common metric for attribution [2–5, 15, 27, 29, 30]. In this approach, one calibrates a model that predicts the conversion probability as a function of the ad exposure and then, uses the estimated model to compute the incremental value of each ad. The novelty comes from the model proposed to describe user behavior, e.g., an HMM [2] or a neural network [4]. Irrespective of the prediction model's level of complexity, attribution is measured via IVH. There exists little (if any) formal justification for why IVH serves as a good approximation to attribution. In fact, in Section 3.2, we show that IVH can result in inappropriate allocations.

SV. SV [43] is a well-accepted concept for assigning credit to individual players in a cooperative game. The value generated by online advertising can be viewed as the outcome of a cooperative effect of the channels and media platforms. Dalessandro et al. [14] pose attribution as a causal estimation problem and propose SV as an approximation scheme for the causally motivated problem. They also show that SV generalizes the probabilistic model of Shao and Li [42]. Using a stylized model, Berman [8] shows the use of SV for attribution can be beneficial to the advertiser.

Other works. Attribution has been tackled from various other angles. Jordan et al. [26] use a Markovian model to motivate a payment scheme that satisfies incentive compatibility for the advertiser and is fair from the publisher's point-of-view. Xu et al. [46] propose a mutually exciting point process to capture dynamic interactions among various ads. Zhang et al. [47], Ji et al. [25], and Ji and Wang [24] provide an interesting view of attribution via the lens of survival theory. Under a stylized setting, Abhishek et al. [1] analyze attribution contracts used by an advertiser to incentivize two publishers that affect customer acquisition. Zhao et al. [48] propose a regression-based relative importance method to compute the marginal contributions.

1.2 Our approach and contributions

With almost a decade of research, it remains unclear as to what is an "appropriate" or "best" attribution measure. IVH appears to be the most popular; but, to the best of our knowledge, there exists little rationale supporting it. On the other hand, SV has a strong theoretical justification. It has a number of desirable properties such as *efficiency*, *symmetry*, *linearity*, and *null player*; in fact, SV is the unique solution to a cooperative game that has all these properties. However, estimating SV exactly is computationally intractable, and one has to resort to approximations [7, 10, 17, 31–34, 36]. In addition, as we show in Section 4.2, SV is not counterfactual in nature. We seek a metric that has the desirable properties of SV, and yet is tractable and able to accommodate counterfactual reasoning.

Our main contributions are as follows. First, we construct an abstract Markov chain model for the user journey through the conversion funnel that generalizes most of the existing Markovian models in the attribution literature. At every period, the user is in one of the finitely many *states*. The advertiser observes the state and takes an *action*. The user transitions to a random state distributed according to a probability mass function that depends on the current state and the advertising action. We propose attributing value to each state-action pair, which is a generalization of the existing approaches that attribute only to advertising actions. This extension allows us to capture state-specific attribution for each action, the need for which is motivated by Bleier and Eisenbeiss [9].

Second, using our Markovian model, we develop a series of intuitive canonical network structures that serve as a robustness check for any attribution scheme. Using these networks, we show the current attribution metrics (LTA, IVH, and uniform) have serious limitations. Furthermore, we show how one can compute state-specific SV for each action in our Markovian model and highlight that it does not adjust for the counterfactual and hence, is an inappropriate metric for attribution. To the best of our knowledge, this is the first work in the literature to analyze the various existing attribution metrics using a common framework.

Third, we develop an axiomatic framework for attribution in online advertising, which forms the main contribution of this work.

[1] Conversion refers to the event in which a user buys the underlying product.

In particular, we propose a new metric for attribution in the Markovian model of user behavior, that we refer to as *counterfactual adjusted Shapley value*. We show that our proposed metric inherits the desirable axioms of the classical SV. We also demonstrate its robustness to a mix of network structures that highlighted limitations of the existing metrics. In addition, we establish that our metric admits a crisp characterization under our Markovian model. It coincides with a *unique-uniform attribution scheme* that explicitly adjusts for the counterfactual. In turn, it can be interpreted as a correction to the commonly used uniform attribution scheme. Furthermore, we exploit this characterization to shed light on the ease of computability of our metric.

The remainder of this paper is organized as follows. In Section 2, we introduce the Markovian model that describes the user journey as a function of ad exposure. We discuss how the existing attribution schemes (LTA, IVH, and uniform) can be captured on the Markovian model and highlight their limitations in Section 3. In Section 4, we showcase the drawback of directly applying SV to measure attribution and then present our novel metric along with the axioms it satisfies. We then characterize our metric as an adjusted unique-uniform attribution scheme in Section 5 and discuss its ease of computability. We conclude in Section 6 where we also discuss some directions for ongoing and future research. Some technical results are deferred to Appendix A.

2 MODEL

We propose a Markovian model for user behavior. The transitions in user's state are stochastic, and are a function of only the current state and the advertiser action. This process ends when the user quits (leaves the system) or converts (buys the product). In Section 2.1, we define the components of the Markov chain (denoted by $\mathcal{M}$). When constructing the Markov chain, we keep most of its elements abstract to showcase its flexibility. In Section 2.2, we shed light on the abstractness through practical examples. We conclude this section by defining the attribution problem over our Markovian model (Section 2.3).

2.1 Markovian model of user behavior

We first discuss the state space of the Markov chain followed by the arrival process of the users. We then elaborate on the action space of the advertiser. Finally, we define the transition probabilities.

State space. We define $\mathbb{S} := \{s\}_{s=1}^{m}$ as the set of states excluding the two absorbing states (quit q and conversion c) and $\mathbb{S}^+ := \mathbb{S} \cup \{q, c\}$. In order to highlight the flexibility of our model, we do not yet give a concrete meaning to a state.

Arrival process. External traffic arrives at state $s \in \mathbb{S}$ w.p. λ_s (*initial state probability*). We define the vector $\lambda \in \mathbb{R}^m$ as $[\lambda_s]_{s\in\mathbb{S}}$. We assume no external traffic arrives at c and q.

Action space. We define $\mathbb{A} := \{a\}_{a=1}^{n}$ as the set of actions the advertiser can take, such as sending an email or showing a display ad. We include the no-ad action ($a = 1$)[2] in $\mathbb{A}$ and define β_s^a to be the probability that an advertiser takes action $a \in \mathbb{A}$ at state $s \in \mathbb{S}$. We denote by β the collection of all β_s^a values and assume it is known.

Transition probabilities. We denote by $p_{ss'}^a$ the probability a user moves from $s \in \mathbb{S}$ to $s' \in \mathbb{S}$ in one transition given the advertiser takes action $a \in \mathbb{A}$ at s. Also, for all $(s, s') \in \mathbb{S}^2$, we define $p_{ss'}^\beta := \sum_{a\in\mathbb{A}} \beta_s^a p_{ss'}^a$, which denotes the *average* transition probability. To keep the notation concise, for each $a \in \mathbb{A}$, we define the matrix $P^a := [p_{ss'}^a]_{(s,s')\in\mathbb{S}^2} \in \mathbb{R}^{m\times m}$. Furthermore, $P^\beta := [p_{ss'}^\beta]_{(s,s')\in\mathbb{S}^2} \in \mathbb{R}^{m\times m}$ and $B^a := \text{diag}([\beta_s^a]_{s\in\mathbb{S}}) \in \mathbb{R}^{m\times m}$ for each $a \in \mathbb{A}$ is a diagonal matrix. Clearly, P^β represents the transition matrix over the *partial* state space $\mathbb{S}$ and $P^\beta = \sum_{a\in\mathbb{A}} B^a P^a$. For all $s \in \mathbb{S}$, we define the vector $p_s^a \in \mathbb{R}^m$ as the s-th row of P^a for all $a \in \mathbb{A}$ and $p_s^\beta \in \mathbb{R}^m$ as the s-th row of P^β. Next, we state the only assumption we make on our problem primitives.

Assumption 2.1. *The Markov chain with P^β as its transition matrix is absorbing, i.e., the probability each user will eventually either quit or convert from any state equals 1.*

We note that Assumption 2.1 is very mild. So far, we have not discussed the transitions to and from states q and c. We use the notation p_{sc}^a and p_{sc}^β to denote the action-specific and average one-step transition probabilities from $s \in \mathbb{S}$ to c for all $a \in \mathbb{A}$. (For transitions from $s \in \mathbb{S}$ to q, replace the index c by q.) Since both q and c are absorbing states, the transitions from them are self-loops w.p. 1. We define $p_c^a := [p_{sc}^a]_{s\in\mathbb{S}} \in \mathbb{R}^m$ for each $a \in \mathbb{A}$ and $p_c^\beta := [p_{sc}^\beta]_{s\in\mathbb{S}} \in \mathbb{R}^m$. (Note that p_s^a and p_s^β for $s \in \mathbb{S}$ correspond to the probabilities of *leaving* s whereas p_c^a and p_c^β correspond to the probabilities of *entering* c.) Next, we shift our focus to three quantities of interest for this Markov chain.

Expected number of visits. We define $F^\beta \in \mathbb{R}^{m\times m}$ as the *fundamental matrix* of $\mathcal{M}$, i.e., its (i, j)-th entry equals the expected number of visits to state j if the initial state is i. Results in Markov chain theory [18] imply that $F^\beta = (I - P^\beta)^{-1}$ and Assumption 2.1 ensures that F^β exists and is finite.

Effective arrival rate. Let μ_s^β denote the *effective* arrival rate into state $s \in \mathbb{S}$ and $\mu^\beta := [\mu_s^\beta]_{s\in\mathbb{S}} \in \mathbb{R}^m$. It is easy to show that $\lambda^\top + (\mu^\beta)^\top P^\beta = (\mu^\beta)^\top$; hence, $(\mu^\beta)^\top = \lambda^\top F^\beta$.

Eventual conversion probability. We define h_s^β as the probability of eventually being absorbed in c from state $s \in \mathbb{S}$ and the vector $h^\beta := [h_s^\beta]_{s\in\mathbb{S}} \in \mathbb{R}^m$. It is easy to show that $h^\beta = P^\beta h^\beta + p_c^\beta$. Thus, $h^\beta = F^\beta p_c^\beta$. We set $h_q^\beta = 0$ and $h_c^\beta = 1$.

2.2 Model discussion

We have so far defined the state and action spaces in abstract terms. Now, we provide some possible mappings from physical quantities to these.

As a first possibility, the state may be composed of past interactions with the user, e.g., the number of visits to the product website, number of emails received, number of emails opened, number of display ads seen, number of display ads clicked, etc. The number of states may become too large, resulting in potential tractability issues. However, one can quantize the counts to control the size of the state space. The advertiser's action space includes doing nothing (no ad), or sending an email, or showing a display ad, etc.

[2] We are reserving $a = 0$ for future use as it will become transparent in Section 4.2.

As another possibility, the state space may correspond to the widespread conversion funnel used in the marketing literature which captures the journey of the user to the state of being unaware of the product to eventually becoming interested and finally purchasing. Given the fine granularity of data available to advertisers, we believe the advertiser can infer such states using an appropriate statistical model. The action space remains the same as above.

We note that the definitions of state and action spaces can be customized as per the needs of the advertiser (since our attribution framework is developed for the abstract Markovian model). In fact, the state space should very much be context-specific and driven by the user features that are relevant to a particular advertiser.

2.3 The attribution problem

The total value[3] generated by the network equals $\lambda^\top h^\beta$. The goal of an attribution model is to allocate this value to the underlying *agents* (state-action pairs)[4] in the system while accounting for the counterfactual. For any given $(s,a) \in \mathbb{S} \times \mathbb{A}$, we use π_s^a to denote the corresponding attribution. Naturally, a desirable property is that the total value in the system should be credited back to the agents, i.e., $\sum_{s\in\mathbb{S}} \sum_{a\in\mathbb{A}} \pi_s^a = \lambda^\top h^\beta$.

3 CURRENT APPROACHES & LIMITATIONS

We now define in our context the commonly used attribution schemes (LTA, IVH, and uniform) and illustrate their limitations. At a high-level, both LTA and uniform are "backward looking" in the sense that they split the value of a path after observing its realization (from the initial state to the end). LTA allocates all the value to the last interacting agent whereas uniform allocates it equally to all the agents that appeared in the path. On the other hand, IVH is "forward looking" since it attributes to a given agent based on what will happen in the future. It does not require the knowledge of how the actual path will unfold. All it requires is a model that can output the change in eventual conversion probability that results if the agent is removed from the path.

3.1 LTA

In LTA, all the value generated due to a purchase is attributed to the last state-action pair in the path. Denote by $\mathcal{P}$ a random path (over state-action pairs) sampled uniformly from $\mathcal{M}$. For $(s,a) \in \mathbb{S} \times \mathbb{A}$, define

$$\hat{w}_s^a(\mathcal{P}) := \begin{cases} 1 & \text{if } \mathcal{P} \text{ converts and } (s,a) \text{ is the last agent in } \mathcal{P} \\ 0 & \text{otherwise.} \end{cases}$$

The attribution to $(s,a) \in \mathbb{S} \times \mathbb{A}$ under the LTA scheme equals

$$\pi_s^{a,\text{LTA}} := \mathbb{E}_{\mathcal{P}\sim\mathcal{M}} \left[\hat{w}_s^a(\mathcal{P})\right].$$

In other words, (s,a) receives complete credit of each *converted* path it is the last agent of. Trivially, LTA attributes exactly all the value generated by the network (*efficient*) and is easy to implement. However, LTA appears "unfair" since the states are not rewarded for moving the user up in the conversion funnel. We demonstrate this limitation next.

[3]We assume that the sale of one unit of the product generates a value of 1. If the true number is different from 1, the numbers can be scaled accordingly.

[4]One might want to consider actions as the agents. However, we view the state-action pairs as the agents, because the impact of an action is likely to be state dependent.

Example 3.1 (LTA is unfair). Consider the network in Figure 1 with self-loop probability $p = 0$ (*line*). Under LTA, all the value goes to the ad action at state m although state 1 brings in all the traffic and the ad action at each state pushes the user towards conversion.

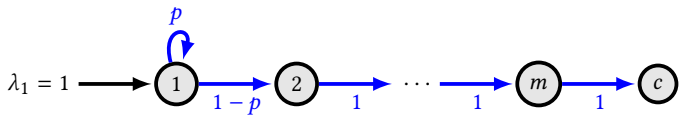


Figure 1: Network for Examples 3.1 and 3.4. The action space consists of two actions: show no-ad and show an ad. The lines denote transitions if an ad is shown. For brevity, we do not show the transitions if an ad is not shown (to quit state w.p. 1). The advertiser shows an ad at all states w.p. 1.

3.2 IVH

IVH allocates to each state-action pair (s,a) the increase in the eventual conversion probability by taking action a in state s as opposed to the no-ad action. For each $a \in \mathbb{A}$, we first define an auxiliary variable that captures the corresponding forward looking increment *conditioned* on the user being at a given state:

$$z^{a,\text{IVH}} := \underbrace{P^a h^\beta + p_c^a}_{\text{action } a} - \underbrace{(P^1 h^\beta + p_c^1)}_{\text{no-ad action}} = \underbrace{(P^a - P^1)h^\beta}_{\text{eventual}} + \underbrace{(p_c^a - p_c^1)}_{\text{immediate}},$$

where $z^{a,\text{IVH}} = [z_s^{a,\text{IVH}}]_{s\in\mathbb{S}} \in \mathbb{R}^m$. The scalar $z_s^{a,\text{IVH}}$ can be interpreted as the allocation at a "trace" level, i.e., if the advertiser observes a user at state s and decides to take action a, the corresponding allocation would be $z_s^{a,\text{IVH}}$. However, we need to scale this metric for it to be seen as attribution over the entire population. In particular, given that at state s, the effective arrival rate is μ_s^β and action a is taken w.p. β_s^a, the IVH attribution is given by

$$\pi_s^{a,\text{IVH}} := \mu_s^\beta \beta_s^a z_s^{a,\text{IVH}}.$$

Although IVH is tractable and somewhat adjusts for the counterfactual, a serious limitation is that it can distribute more value than the network generates because it pays for both eventual and immediate conversions; consequently, each conversion is accounted for several times. We now show that this is, indeed, possible.

Example 3.2 (IVH can over-allocate). Consider the network in Figure 1 with $p = 0$. Under IVH, all states receive an attribution of 1 for the show ad action and 0 for the no-ad action, resulting in a total allocation of m even though the value generated equals 1.

A common workaround to ensure IVH in budget-balanced is to normalize the output by an appropriate constant (*normalized IVH*). Normalizing the numbers in Example 3.2 results in the ad action at each state receiving $1/m$, which appears reasonable. However, as we illustrate in Example 3.3, even normalized IVH can be problematic.

Example 3.3 (Normalized IVH can be unjustified). Consider the network in Figure 2. IVH attributes 1 to state 1 and 1/2 for all other states. Thus, normalized IVH attributes

$$\begin{cases} \frac{1}{1+(m-1)/2} & \text{if state equals 1} \\ \frac{1/2}{1+(m-1)/2} & \text{otherwise.} \end{cases}$$

Clearly, as $m \to \infty$, all the value goes to states 2 to m. This appears inappropriate as state 1 "deserves" at least half the total value since 50% of the users convert immediately after state 1.

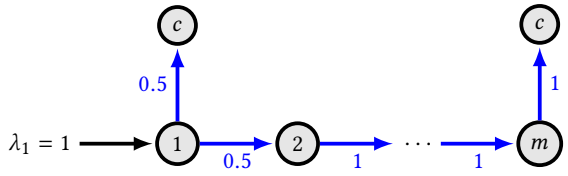


Figure 2: Network for Example 3.3. The action space consists of two actions: show no-ad and show an ad. Solid blue lines denote transitions if an ad is shown. For brevity, we do not show the transitions if an ad is not shown (to quit state w.p. 1). The advertiser shows an ad at all states w.p. 1.

3.3 Uniform attribution

Under a uniform attribution scheme, the credit generated is attributed equally to all the state-action pairs that are encountered in a path. Denote by $\mathcal{P}$ a random path (over state-action pairs) sampled uniformly from $\mathcal{M}$. For $(s, a) \in \mathbb{S} \times \mathbb{A}$, define

$$\bar{w}_s^a(\mathcal{P}) := \begin{cases} \frac{n_s^a}{|\mathcal{P}|} & \text{if } \mathcal{P} \text{ converts and } (s,a) \in \mathcal{P} \\ 0 & \text{otherwise,} \end{cases}$$

where n_s^a equals the number of times (s, a) appears in $\mathcal{P}$ and $|\mathcal{P}|$ denotes the number of state-action pairs in $\mathcal{P}$ (not necessarily unique). The uniform attribution for $(s, a) \in \mathbb{S} \times \mathbb{A}$ is

$$\pi_s^{a,\text{uni}} := \mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[\bar{w}_s^a(\mathcal{P})\right]. \tag{1}$$

In other words, (s, a) receives an "equal cut" of each *converted* path it contributes to. Such a scheme is efficient and scalable. On the line network, it attributes $1/m$ to the ad action at each state, which seems reasonable. However, it does not account for the counterfactual. Furthermore, it creates undesired incentives, which we show next.

Example 3.4 (Undesired incentive). Consider the network in Figure 1 with $p < 1$. For a given path with n_1 occurrences of state 1, the uniform attribution scheme attributes $n_1/(n_1 + m - 1)$ to state 1. As p increases, state 1 receives more credit. However, this seems inappropriate since it gives an undesired incentive to the ad action at state 1 to "game the system" via self-loops. Ideally, an attribution scheme should incentivize the ads to push the user towards conversion.

Remark 3.1 (Unique-uniform). A simple fix to the above drawback is to attribute based on the number of unique agents. For instance, in the example above, the *unique-uniform* scheme would attribute $1/m$ to the ad action at each state. However, this scheme still does not account for the counterfactual, and there is no formal rationale for it. (In the following sections, we will see that adjusting for uniqueness is in fact supported by a strong mathematical rationale and will be part of our prescribed method.)

To summarize this section, the above examples highlight some limitations of existing heuristics but also point to the fact that no single heuristic "dominates the other." In some way, IVH possesses a counterfactual form giving it some practical appeal. At the same time, it is only forward looking, ignoring all past actions and their potential contributions. Uniform, on the other hand, accounts for past actions, but is not counterfactual. Ideally, an attribution scheme would provide the best of both worlds, properly accounting for past actions while also accounting for the counterfactual associated with not advertising.

4 SHAPLEY VALUE

We now study SV as a measure for attribution. We first present a primer on SV (Section 4.1) followed by a discussion on why a direct application to our context does not suffice (Section 4.2). We then adapt SV accordingly (Section 4.3) and conclude this section by discussing the performance of our proposed measure on all the motivating examples (Section 4.4). In this section, we hope to convince the reader that, leaving computational tractability aside, SV (adjusted for the counterfactual) is an appropriate metric for attribution.

4.1 Primer

SV is a well-accepted concept from cooperative game theory that is used to distribute the payoff generated by a *coalition* to the *players* in the coalition [43]. Given a finite set $\mathbb{P}$ of players, the *characteristic function* $v(\cdot)$ maps a coalition $\mathcal{X} \subseteq \mathbb{P}$ to the value (a real number) generated by the coalition. The value $v(\emptyset)$ of the empty coalition is normalized to 0. SV distributes the value $v(\mathbb{P})$ of the grand coalition to a player $r \in \mathbb{P}$ as follows:

$$\pi_r^{\text{Shap}} := \sum_{\mathcal{X}\subseteq\mathbb{P}\setminus\{r\}} w_{|\mathcal{X}|} \times \{v(\mathcal{X}\cup\{r\}) - v(\mathcal{X})\},$$

where

$$w_{|\mathcal{X}|} := \frac{|\mathcal{X}|!(|\mathbb{P}| - |\mathcal{X}| - 1)!}{|\mathbb{P}|!}.$$

The attractiveness of SV is rooted in the fact that it is the *only* solution to a cooperative game that has the following four desirable properties:

1. **Efficiency**: $\sum_{r\in\mathbb{P}} \pi_r = v(\mathbb{P})$.
2. **Symmetry**: Consider players $r, r' \in \mathbb{P}$ such that for any $\mathcal{X} \subseteq \mathbb{P} \setminus \{r, r'\}$, r and r' are *equivalent*, i.e., $v(\mathcal{X} \cup \{r\}) = v(\mathcal{X} \cup \{r'\})$. Then, $\pi_r = \pi_{r'}$.
3. **Linearity**: Consider two characteristic functions $v(\cdot)$ and $w(\cdot)$. Linearity states that for all players $r \in \mathbb{P}$, $\pi_r(v + w) = \pi_r(v) + \pi_r(w)$ and $\pi_r(\alpha v) = \alpha\pi_r(v)$ for all $\alpha \in \mathbb{R}$.
4. **Null player**: Suppose player $r \in \mathbb{P}$ does not add any value to any coalition, i.e., for all $\mathcal{X} \subseteq \mathbb{P}$, $v(X \cup \{r\}) = v(\mathcal{X})$. Then, $\pi_r = 0$.

4.2 Direct application

SV is a natural candidate for decomposing network value because the state-action pairs can be viewed as players participating in a cooperative game to achieve a common goal of converting the users. In particular, given the action intensities β, the network generates a value of $\lambda^\top h^\beta$ and the state-action specific SV $\pi_s^{a,\text{Shap}}$ represents the credit allocated to each state-action pair $(s, a) \in \mathbb{S}\times\mathbb{A}$. We first define the underlying components (players, coalition, and characteristic function) in our context and then discuss an important drawback of such a naive application.

Player. We view each state-action pair $(s, a) \in \mathbb{S} \times \mathbb{A}$ as an underlying player in the cooperative game. Consequently, our SV will not just be action-specific, but also state-specific.

Coalition. A coalition $\mathcal{X} \subseteq \mathbb{S}\times\mathbb{A}$ refers to a collection of players (state-action pairs). We construct a coalition by starting with the *empty coalition*, which corresponds to the Markov chain containing all the states but no action (not even the no-ad action), i.e., the traffic is directed to the quit state w.p. 1. Equivalently, one can view the empty coalition as the one with all the states but *action 0* ("zero-value action") being taken at all of them[5]. If a state-action pair, say (s, a), is added to the empty coalition, then the advertiser takes action a at state s w.p. β_s^a and so on. Therefore, for each coalition $\mathcal{X}$, we obtain a corresponding β, which we denote by $\beta^{\mathcal{X}}$.

Characteristic function. Given $\mathcal{X} \subseteq \mathbb{S} \times \mathbb{A}$, we define

$$v(\mathcal{X}) := \lambda^\top h^{\beta^{\mathcal{X}}}.$$

We define the action probabilities collections $\beta^\emptyset$ and β^* such that they correspond to the empty coalition $\emptyset$ and complete coalition $\mathbb{S} \times \mathbb{A}$, respectively. Thus, $\beta^\emptyset = 0$ and $\beta^* = \{\beta_s^a\}_{(s,a)\in\mathbb{S}\times\mathbb{A}}$. We satisfy $v(\emptyset) = 0$ and $v(\mathbb{S} \times \mathbb{A}) = \lambda^\top h^{\beta^*}$ is the total network value.

Shapley value. The SV for each $(s, a) \in \mathbb{S} \times \mathbb{A}$ is

$$\pi_s^{a,\text{Shap}} := \sum_{\mathcal{X}\subseteq\{\mathbb{S}\times\mathbb{A}\}\setminus\{(s,a)\}} w_{|\mathcal{X}|} \times \{v(\mathcal{X} \cup \{(s, a)\}) - v(\mathcal{X})\}, \quad (2)$$

where

$$w_{|\mathcal{X}|} := \frac{|\mathcal{X}|!(mn - |\mathcal{X}| - 1)!}{(mn)!}. \quad (3)$$

SV depends on our choice of the characteristic function $v(\cdot)$ and the underlying Markov chain $\mathcal{M}$ and hence, we will use the notation $\pi_s^{a,\text{Shap}}(v)$ and $\pi_s^{a,\text{Shap}}(\mathcal{M})$ when the emphasis on $v(\cdot)$ and $\mathcal{M}$ is necessary. Though this view of attribution inherits the desirability of SV, it suffers from a critical flaw: it does not adjust for the counterfactual (Example 4.1).

Example 4.1 (Need for counterfactual). Consider the network in Figure 3. Clearly, showing an ad should not get any attribution since it provides no *additional* value over the counterfactual action (no ad). However, (2) attributes all the value to the ad action. Furthermore, even LTA and uniform attribution fail this sanity check.

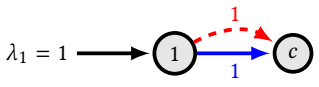


Figure 3: Network for Example 4.1. The action space consists of two actions: show no-ad and show an ad. Solid blue (dashed red) lines denote transition if an ad is shown (not shown). The advertiser shows an ad w.p. 1.

[5] An alternate candidate for the empty coalition is to have the no-ad action being taken at all the states. In other words, the no-ad action fills the "void" left due to the absence of other actions. However, we believe such a view is inappropriate since the no-ad action itself can have some value (either in terms of network effect or in terms of direct conversions). Therefore, using the no-ad action as the "default" action would allow the other actions to free ride on the value generated by the no-ad action, and hence, undermine the contribution due to the no-ad action.

4.3 Our approach

As we alluded to earlier, we seek a measure that provides the best of two worlds: (1) appropriately captures the contributions of the past actions (capturing the feature of, e.g., uniform and SV)[6] and (2) exhibits counterfactual reasoning (capturing a feature of IVH). To this end, we focus on adapting SV to account for the counterfactual and we do so by adhering to the causality framework of Rubin [41]. We first define a *counterfactual player* followed by our novel definition of *counterfactual adjusted Shapley value (CASV)*. We then show the desirability of CASV.

Counterfactual player. Following Rubin [41], the counterfactual to taking action a at state s w.p. β_s^a is to take the no-ad action (action 1) at state s w.p. β_s^a (instead of β_s^1). Accordingly, for a given player $(s, a) \in \mathbb{S} \times \mathbb{A}$, we denote the counterfactual player as $(s, 1)^a$, where the "a" in the superscript captures the dependence on β_s^a.

Counterfactual adjusted Shapley value. The game theoretic setup (player, coalition, characteristic function) is the same as in Section 4.2. For each $(s, a) \in \mathbb{S} \times \mathbb{A}$, we define CASV as

$$\psi_s^{a,\text{Shap}} := \sum_{\mathcal{X}\subseteq\{\mathbb{S}\times\mathbb{A}\}\setminus\{(s,a)\}} w_{|\mathcal{X}|} \times \{v(\mathcal{X} \cup \{(s, a)\}) - v(\mathcal{X} \cup \{(s, 1)^a\})\}, \quad (4)$$

where $w_{|\mathcal{X}|}$ is the same as in (3). We use the symbol ψ instead of π to differentiate CASV from SV. The only change we make in going from $\pi_s^{a,\text{Shap}}$ to $\psi_s^{a,\text{Shap}}$ is that we replace $v(\mathcal{X})$ by $v(\mathcal{X} \cup \{(s, 1)^a\})$, i.e., CASV captures the value added by a player over its counterfactual. Intuitively, it is unclear whether CASV under the current cooperative game is equivalent to SV under a different cooperative game. However, it is possible to view CASV as the difference between two SVs:

$$\psi_s^{a,\text{Shap}}(\mathcal{M}) = \pi_s^{a,\text{Shap}}(\mathcal{M}) - \pi_s^{a,\text{Shap}}(\mathcal{M}_s^a), \quad (5)$$

where $\mathcal{M}$ denotes the original Markov chain and $\mathcal{M}_s^a$ denotes the *counterfactual network* for (s, a), i.e., it is identical to $\mathcal{M}$ except that we replace the transition probabilities of (s, a) by those of $(s, 1)$. Clearly, $\pi_s^{a,\text{Shap}}(\mathcal{M}_s^a)$ is the *counterfactual value* of (s, a), i.e., the value generated if no-ad was shown instead of ad a at state s.

Obviously, it is of interest to check how CASV performs with respect to the desirable properties that motivated SV in the first place. However, one needs to be careful when discussing these axioms in the counterfactual context. To be specific, efficiency should now pertain to the redistribution of additional value generated over the counterfactual value. Similarly, the definition of equivalent players should be adjusted (discussed in Remark 4.1) and null player should correspond to a player with zero value-add. We formalize the desirability of CASV in Theorem 4.2, the proof of which is in Appendix A.

Theorem 4.2 (Axioms). *CASV satisfies the following axioms:*

1. ***Counterfactual efficiency****: The sum of CASVs equals the additional value generated over the counterfactual value, i.e.,*

$$\sum_{(s,a)\in\mathbb{S}\times\mathbb{A}} \psi_s^{a,\text{Shap}} = v(\mathbb{S} \times \mathbb{A}) - \sum_{(s,a)\in\mathbb{S}\times\mathbb{A}} \pi_s^{a,\text{Shap}}(\mathcal{M}_s^a).$$

[6] We will see in Section 5 that uniform and SV are closely linked.

2. ***Counterfactual symmetry***: *If* $(s,a) \in \mathbb{S} \times \mathbb{A}$ *and* $(s',a') \in \mathbb{S} \times \mathbb{A}$ *are counterfactual equivalent, i.e.,*
(A) $v(\mathcal{X} \cup \{(s,a)\}) - v(\mathcal{X} \cup \{(s,1)^a\}) = v(\mathcal{X} \cup \{(s',a')\}) - v(\mathcal{X} \cup \{(s',1)^{a'}\})$ *and*
(B) $v(\mathcal{X} \cup \{(s,1)^a, (s',a')\}) = v(\mathcal{X} \cup \{(s',1)^{a'}, (s,a)\})$
for all $\mathcal{X} \subseteq \{\mathbb{S} \times \mathbb{A}\} \setminus \{(s,a),(s',a')\}$*, then*

$$\psi_s^{a,\,Shap} = \psi_{s'}^{a',\,Shap}.$$

3. ***Linearity***: *Consider two characteristic functions* $v(\cdot)$ *and* $w(\cdot)$*. For all* $(s,a) \in \mathbb{S} \times \mathbb{A}$*, we have*

$$\psi_s^{a,\,Shap}(v+w) = \psi_s^{a,\,Shap}(v) + \psi_s^{a,\,Shap}(w)$$

and for all $\alpha \in \mathbb{R}$*,*

$$\psi_s^{a,\,Shap}(\alpha v) = \alpha \psi_s^{a,\,Shap}(v).$$

4. ***Counterfactual null player***: *Consider a player* $(s,a) \in \mathbb{S} \times \mathbb{A}$ *that has a zero value-add to all coalitions that do not contain* (s,a)*, i.e., for all* $\mathcal{X} \subseteq \{\mathbb{S} \times \mathbb{A}\} \setminus \{(s,a)\}$*,*

$$v(\mathcal{X} \cup \{(s,a)\}) = v(\mathcal{X} \cup \{(s,1)^a\}).$$

Then, $\psi_s^{a,\,Shap} = 0$.

Furthermore, CASV is the unique solution to satisfy these four counterfactual axioms.

Remark 4.1 (Counterfactual equivalent players). In Theorem 4.2, we define $(s,a) \in \mathbb{S} \times \mathbb{A}$ and $(s',a') \in \mathbb{S} \times \mathbb{A}$ to be counterfactual equivalent if they satisfy (A) and (B). To develop an intuitive understanding of these conditions, one can consider the following three conditions for all $\mathcal{X} \subseteq \{\mathbb{S} \times \mathbb{A}\} \setminus \{(s,a),(s',a')\}$:

(A1) $v(\mathcal{X} \cup \{(s,a)\}) = v(\mathcal{X} \cup \{(s',a')\})$,
(A2) $v(\mathcal{X} \cup \{(s,1)^a\}) = v(\mathcal{X} \cup \{(s',1)^{a'}\})$, and
(B) $v(\mathcal{X} \cup \{(s,1)^a, (s',a')\}) = v(\mathcal{X} \cup \{(s',1)^{a'}, (s,a)\})$.

In simple words, consider a coalition that does not contain either of the two given players. Condition (A1) states that adding either of the two players has the same effect on the network value and condition (A2) says so for the counterfactual players. The remaining case of interest is what happens when, on one hand, we add the first player and the counterfactual of the second player whereas, on the other hand, we do the opposite (add the second player and the counterfactual of the first player). Condition (B) states that doing either is equivalent in terms of the characteristic function. Note that we only need conditions (A) and (B) for two players to be counterfactual equivalent, and (A1) and (A2) are sufficient conditions for (A). It is worth mentioning that our definition of counterfactual equivalent players is a generalization of equivalent players from Section 4.1. In particular, equivalent players only need to satisfy condition (A1) and since the notion of a counterfactual player does not exist in the SV context, conditions (A2) and (B) are irrelevant.

We conclude this subsection with a discussion regarding the attribution to the no-ad action. By definition of CASV, the no-ad action at each state receives zero counterfactual credit. However, one can attribute the counterfactual value $\sum_{(s,a)\in\mathbb{S}\times\mathbb{A}} \pi_s^{a,\,\text{Shap}}(\mathcal{M}_s^a)$ ("residual") to the no-ad action in a post hoc fashion. In other words, the post hoc attribution to the no-ad action at state $s \in \mathbb{S}$ equals

$$\sum_{a\in\mathbb{A}} \pi_s^{a,\,\text{Shap}}(\mathcal{M}_s^a) = \pi_s^{1,\text{Shap}}(\mathcal{M}) + \sum_{a=2}^{n} \pi_s^{a,\,\text{Shap}}(\mathcal{M}_s^a), \tag{6}$$

where we use the fact that $\mathcal{M}_s^1 = \mathcal{M}$. The two terms in (6) have an intuitive interpretation. The first term captures the value generated by the no-ad action due to a positive value of β_s^1 whereas the second term captures the value that *would have* been generated if the no-ad action was used in place of the other actions.

4.4 Revisiting canonical examples

We now revisit the networks discussed so far and show that our CASV measure does not suffer from the same limitations as existing heuristics.

Example 4.3. For the network in Figure 1 with $p < 1$, CASV at every state equals $1/m$ for the ad action and 0 otherwise.

Example 4.4. For the network in Figure 2, CASV attributes 0 to the no-ad action at all the states and the following to the ad action:

$$\begin{cases} \frac{1}{2} + \frac{1}{2m} & \text{if } s = 1 \\ \frac{1}{2m} & \text{otherwise.} \end{cases}$$

This seems sensible as half of the paths convert just due to the ad action at state 1 whereas the other half of the paths convert with the help of the ad action at all the m states.

Example 4.5. For the network in Figure 3, CASV allocates zero credit to the ad action, which seems appropriate. However, CASV of the no-ad action also equals zero, which raises the following question: which player gets credit for the value in the system? The answer is, as discussed at the end of Section 4.3, the no-ad action.

In sum, CASV appears to be an appealing measure for attribution: it has a number of desirable properties and appears robust to various network structures. However, similar to SV, computing CASV using (4) requires an exponential runtime in the number of underlying players. This is an important tractability concern that could render CASV impractical in settings with moderately-sized state and action spaces. In the next section, we focus on representing CASV in a manner that is amenable to being computed efficiently, under our Markovian model.

5 CHARACTERIZING COUNTERFACTUAL ADJUSTED SHAPLEY VALUE

We now focus on characterizing CASV by exploiting the structure of the underlying cooperative game. In doing so, we use the fact that CASV can be expressed as a difference of two SVs (see (5)) and hence, analyze SV first. Proposition 5.1 connects the coalition-oriented game-theoretic construct of SV to the paths sampled from $\mathcal{M}$. In particular, we show that, under our Markovian setup, SV is (surprisingly) identical to the unique-uniform attribution scheme (motivated in Remark 3.1). We present the formal proof of Proposition 5.1 in Appendix A and highlight the intuition here.

Proposition 5.1 (SV equals unique-uniform). *Consider* $(s,a) \in \mathbb{S} \times \mathbb{A}$ *and the Markov chain* $\mathcal{M}$*. The SV of* (s,a) *as defined in* (2)

equals

$$\pi_s^{a,\,Shap} = \mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[w_s^a(\mathcal{P})\right],$$

where

$$w_s^a(\mathcal{P}) := \begin{cases} \frac{1}{u(\mathcal{P})} & \text{if } \mathcal{P} \text{ converts and } (s,a)\in\mathcal{P} \\ 0 & \text{otherwise.} \end{cases}$$

The function $u(\cdot)$ returns the number of unique players and $\mathcal{P}$ is a path over players, i.e., state-action pairs.

Proof sketch. There are two keys ideas at interplay here. First, to characterize SV, we observe that the characteristic function for a given coalition $\mathcal{X}\subseteq\mathbb{S}\times\mathbb{A}$ can be expressed as an expectation over the paths sampled from the Markov chain $\mathcal{M}(\mathcal{X})$ in which only the players in $\mathcal{X}$ are "active", i.e.,

$$v(\mathcal{X}) = \mathbb{E}_{\mathcal{P}\sim\mathcal{M}(\mathcal{X})}\left[\mathbb{I}_{\{\mathcal{P}\text{ converts}\}}\right].$$

Using this observation allows us to express SV $\pi_s^{a,\,\text{Shap}}$ as an expectation over the paths from the original Markov chain $\mathcal{M}$:

$$\pi_s^{a,\,\text{Shap}} = \mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[\mathbb{I}_{\{\mathcal{P}\text{ converts}\}}\,(\star)\right],$$

where

$$(\star) := \sum_{\mathcal{X}\subseteq\{\mathbb{S}\times\mathbb{A}\}\setminus\{(s,a)\}} w_{|\mathcal{X}|}\left(\mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\cup\{(s,a)\}\}} - \mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\}}\right).$$

The second key idea is to analyze $(\star)$ rigorously, which forms the bulk of our analysis. We show that $(\star)$ admits the following remarkably simple form:

$$(\star) = \begin{cases} \frac{1}{u(\mathcal{P})} & \text{if } (s,a)\in\mathcal{P} \\ 0 & \text{otherwise.} \end{cases}$$

Putting the two pieces together allows us to finish the proof. □

The above result enables one to characterize CASV, which constitutes the main result of this work.

Theorem 5.1 (CASV equals unique-uniform minus counterfactual). *Consider $(s,a)\in\mathbb{S}\times\mathbb{A}$, the Markov chain $\mathcal{M}$, and the counterfactual Markov chain $\mathcal{M}_s^a$. The CASV of (s,a) as defined in* (4) *equals*

$$\psi_s^{a,\,Shap} = \mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[w_s^a(\mathcal{P})\right] - \mathbb{E}_{\mathcal{P}\sim\mathcal{M}_s^a}\left[w_s^a(\mathcal{P})\right],$$

where $w_s^a(\cdot)$ and $u(\cdot)$ are as defined in Proposition 5.1.

There is a striking resemblance between this characterization of CASV and uniform attribution as defined in (1). In particular, the two expressions are identical except the definition of the weight function and the counterfactual adjustment. The weight function in CASV does not provide an incentive for players to "game the system" as it only rewards based on whether a player appears in the path or not (as opposed to the number of times it appears). In other words, if an ad a had to be shown multiple times (say n_s^a) at the *same* state s to make the user move to another state s', then the CASV weight function only rewards it once (as opposed to rewarding it n_s^a times). This seems very reasonable and, in fact, is the simple fix we proposed in Remark 3.1.

It is noteworthy that the coalition-oriented construct of CASV, which on the surface does not seem to be related to the paths of the underlying Markovian model, reduces to being expressed as a remarkably simple function of such paths. This connection is quite valuable as it helps to gain deeper insights regarding the structure of CASV and hence, build a better understanding. Next, we conclude this section by using this connection to highlight that our scheme is efficiently computable and implementable.

Given a set of user paths $\mathcal{D} := \{\mathcal{P}_i\}_{i=1}^N$ where each user path $\mathcal{P}_i$ consists of various state-action-state (s,a,s') tuples, the action-specific transition probabilities can be estimated using empirical frequencies. To estimate CASV for player (s,a), one can sample paths from the estimated Markov chains $\mathcal{M}$ and $\mathcal{M}_s^a$ and use Theorem 5.1 directly. However, this involves sampling from a different Markov chain $\mathcal{M}_s^a$ to estimate CASV for each player, which we address next.

A simple change of measure allows CASV to be expressed as

$$\psi_s^{a,\,\text{Shap}} = \mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[w_s^a(\mathcal{P})\left(1 - \frac{g_s^a(\mathcal{P})}{g(\mathcal{P})}\right)\right], \tag{7}$$

where $g(\mathcal{P})$ and $g_s^a(\mathcal{P})$ denote the probabilities of observing path $\mathcal{P}$ under Markov chains $\mathcal{M}$ and $\mathcal{M}_s^a$, respectively. The ratio $g_s^a(\mathcal{P})/g(\mathcal{P})$ denotes the *importance weight* and it is easy to show that

$$\frac{g_s^a(\mathcal{P})}{g(\mathcal{P})} = \prod_{s'\in\mathbb{S}^+}\left(\frac{p^1_{ss'}}{p^a_{ss'}}\right)^{n^a_{ss'}(\mathcal{P})},$$

where $n^a_{ss'}(\mathcal{P})$ denotes the number of occurrences of the tuple (s,a,s') in $\mathcal{P}$ for each $s'\in\mathbb{S}^+$. Given (7), it suffices to sample paths from just one Markov chain ($\mathcal{M}$) to estimate CASV for all the players. In fact, each sample can be recycled for multiple players and the scheme can be implemented using a parallel (over both players and samples) architecture.

It is possible to use a Bayesian version of the approach described above. In particular, one can maintain a belief over the transition probability vector $[p^a_{ss'}]_{s'\in\mathbb{S}^+}$ in the form of a Dirichlet distribution for each $a\in\mathbb{A}$. Using Dirichlet-multinomial conjugacy and real data $\mathcal{D}$, the belief can be updated easily. The posterior belief can be used to generate samples of the Markov chain $\mathcal{M}$ itself. For each sample of $\mathcal{M}$, one can use (7) to estimate CASV, resulting in posterior samples of CASV. Naturally, these posterior samples quantify the uncertainty in CASV that arises due to the noise in data and/or lack of data. Furthermore, this Bayesian approach is amenable to parallelization over the samples of $\mathcal{M}$.

6 CONCLUSIONS & FURTHER RESEARCH

In this paper, using a Markovian model for user behavior, we have proposed a new metric (counterfactual adjusted Shapley value) for the attribution problem in online advertising. We have established its grounding and appropriateness in two ways. First, we have shown the robustness of the proposed measure over various canonical settings in which the existing metrics fail. Second, we have provided an underlying axiomatic framework motivated by game theory and causality that supports our choice. Furthermore, we have established a characterization of the proposed metric as a remarkably simple function of the user paths. In particular, we have established, in our Markovian model, that the proposed attribution metric is an adjustment to the unique-uniform attribution scheme. Finally, we have proposed multiple approaches to efficiently estimate our metric.

There are many potential directions for future research, most of which we are pursuing at the present. First, it is of interest to develop more canonical settings to further verify the appropriateness of our proposed measure. Second, though we have briefly touched upon algorithms to estimate the proposed metric, a thorough treatment would involve understanding their statistical efficiency. In addition, it is worthwhile to explore algorithms to estimate the counterfactual adjusted Shapley value that can be implemented in an online fashion while maintaining strong approximation guarantees. Third, with the help of an industry partner, we are working on comparing the output of our scheme to the existing ones on a real world dataset composed of millions of user paths. Fourth, we are interested in applying the methodology developed in this paper to other domains. In particular, the abstract Markovian model presented here can be used to model other applications, where one wishes to understand the contributions of the underlying agents. Finally, it is worthwhile to explore whether attribution metrics can be constructed from other notions of causality. In this work, we modified Shapley value using Rubin's definition of causality [41]. However, there exist several alternative formulations for causality. See, e.g. Chockler and Halpern [11], Collins et al. [13], Eells [16], Halpern and Pearl [20], Hitchcock [21], Morgan and Winship [35], Pearl [37] and Hume [22] for approaches to causality in the computer science and philosophy literature. Chockler and Halpern [11] proposed metrics such as "degree of responsibility" and "blame" in order to *quantify* the causal effect of one variable on another. However, computing these metrics is computationally expensive. Furthermore, these metrics lack axiomatic support. In particular, similar to IVH, these metrics are not budget balanced. In fact, Chockler and Halpern [11] mention adjusting their notions using Shapley value as a potential future work. Accordingly, though we have unified the existing attribution literature in this work, there is more to explore in terms of the set of counterfactuals to consider.

ACKNOWLEDGMENTS

We acknowledge the support of Adobe Digital Marketing Research Award for this project. V. Goyal also acknowledges the support of NSF grants CMMI 1636046 and CMMI 1351838 for this work. We also thank Abhishek Pani, Chen Dong, Manoj Ravi, and Jie Zhang from the Media and Advertising Solutions team at Adobe for useful discussions during the course of this work. Finally, we thank the WWW reviewers for the valuable feedback and pointing us to the responsibility and blame literature.

A TECHNICAL RESULTS

Theorem 4.2 (Axioms). *CASV satisfies the following axioms:*

1. ***Counterfactual efficiency**: The sum of CASVs equals the additional value generated over the counterfactual value, i.e.,*
$$\sum_{(s,a)\in\mathbb{S}\times\mathbb{A}} \psi_s^{a,\,Shap} = v(\mathbb{S}\times\mathbb{A}) - \sum_{(s,a)\in\mathbb{S}\times\mathbb{A}} \pi_s^{a,\,Shap}(\mathcal{M}_s^a).$$
2. ***Counterfactual symmetry**: If $(s,a)\in\mathbb{S}\times\mathbb{A}$ and $(s',a')\in\mathbb{S}\times\mathbb{A}$ are counterfactual equivalent, i.e.,*
(A) $v(\mathcal{X}\cup\{(s,a)\}) - v(\mathcal{X}\cup\{(s,1)^a\}) = v(\mathcal{X}\cup\{(s',a')\}) - v(\mathcal{X}\cup\{(s',1)^{a'}\})$ *and*
(B) $v(\mathcal{X}\cup\{(s,1)^a,(s',a')\}) = v(\mathcal{X}\cup\{(s',1)^{a'},(s,a)\})$
for all $\mathcal{X}\subseteq\{\mathbb{S}\times\mathbb{A}\}\setminus\{(s,a),(s',a')\}$, then
$$\psi_s^{a,\,Shap} = \psi_{s'}^{a',\,Shap}.$$
3. ***Linearity**: Consider two characteristic functions $v(\cdot)$ and $w(\cdot)$. For all $(s,a)\in\mathbb{S}\times\mathbb{A}$, we have*
$$\psi_s^{a,\,Shap}(v+w) = \psi_s^{a,\,Shap}(v) + \psi_s^{a,\,Shap}(w)$$
and for all $\alpha\in\mathbb{R}$,
$$\psi_s^{a,\,Shap}(\alpha v) = \alpha\psi_s^{a,\,Shap}(v).$$
4. ***Counterfactual null player**: Consider a player $(s,a)\in\mathbb{S}\times\mathbb{A}$ that has a zero value-add to all coalitions that do not contain (s,a), i.e., for all $\mathcal{X}\subseteq\{\mathbb{S}\times\mathbb{A}\}\setminus\{(s,a)\}$,*
$$v(\mathcal{X}\cup\{(s,a)\}) = v(\mathcal{X}\cup\{(s,1)^a\}).$$
Then, $\psi_s^{a,\,Shap} = 0$.

Furthermore, CASV is the unique solution to satisfy these four counterfactual axioms.

Proof. Counterfactual efficiency and linearity follow from (5) when used with the efficiency and linearity of SV, respectively. Counterfactual null player follows from (4). For counterfactual symmetry, consider $(s,a)\in\mathbb{S}\times\mathbb{A}$ and $(s',a')\in\mathbb{S}\times\mathbb{A}$ satisfying (A) and (B), define the set $\diamond := \{\mathbb{S}\times\mathbb{A}\}\setminus\{(s,a),(s',a')\}$, and observe that

$$\begin{aligned}
\psi_s^{a,\,\text{Shap}} &= \sum_{\mathcal{X}\subseteq\{\mathbb{S}\times\mathbb{A}\}\setminus\{(s,a)\}} w_{|\mathcal{X}|}\times\{v(\mathcal{X}\cup\{(s,a)\}) - v(\mathcal{X}\cup\{(s,1)^a\})\}\\
&= \sum_{\mathcal{X}\subseteq\diamond} w_{|\mathcal{X}|}\times\{v(\mathcal{X}\cup\{(s,a)\}) - v(\mathcal{X}\cup\{(s,1)^a\})\}\\
&+ \sum_{\mathcal{X}\subseteq\diamond} w_{|\mathcal{X}|+1}\times\{v(\mathcal{X}\cup\{(s,a),(s',a')\}) - v(\mathcal{X}\cup\{(s,1)^a,(s',a')\})\}\\
&= \sum_{\mathcal{X}\subseteq\diamond} w_{|\mathcal{X}|}\times\left\{v(\mathcal{X}\cup\{(s',a')\}) - v(\mathcal{X}\cup\{(s',1)^{a'}\})\right\}\\
&+ \sum_{\mathcal{X}\subseteq\diamond} w_{|\mathcal{X}|+1}\times\left\{v(\mathcal{X}\cup\{(s,a),(s',a')\}) - v(\mathcal{X}\cup\{(s',1)^{a'},(s,a)\})\right\}\\
&= \sum_{\mathcal{X}\subseteq\{\mathbb{S}\times\mathbb{A}\}\setminus\{(s',a')\}} w_{|\mathcal{X}|}\times\left\{v(\mathcal{X}\cup\{(s',a')\}) - v(\mathcal{X}\cup\{(s',1)^{a'}\})\right\}\\
&= \psi_{s'}^{a',\,\text{Shap}}.
\end{aligned}$$

This proves that CASV satisfies the four axioms. To be concise, we defer the uniqueness proof to the journal version of this paper. □

Proposition 5.1 (SV equals unique-uniform). *Consider $(s,a)\in\mathbb{S}\times\mathbb{A}$ and the Markov chain $\mathcal{M}$. The SV of (s,a) as defined in (2) equals*
$$\pi_s^{a,\,Shap} = \mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[w_s^a(\mathcal{P})\right],$$
where
$$w_s^a(\mathcal{P}) := \begin{cases}\frac{1}{u(\mathcal{P})} & \text{if } \mathcal{P} \text{ converts and } (s,a)\in\mathcal{P}\\ 0 & \text{otherwise.}\end{cases}$$
The function $u(\cdot)$ returns the number of unique players and $\mathcal{P}$ is a path over players, i.e., state-action pairs.

Proof. For convenience, we use linearized notation such that $r := (s, a)$ and $\mathbb{P} := \mathbb{S} \times \mathbb{A}$ with the understanding that $\pi_r^{\text{Shap}} := \pi_s^{a,\text{Shap}}$, and $w_r(\cdot) := w_s^a(\cdot)$. The proof is split into three parts.

Step 1: Express $v(\mathcal{X})$ as an expectation over paths:

$$\begin{aligned} v(\mathcal{X}) &= \lambda^\top h^{\beta^{\mathcal{X}}} \\ &= \lambda^\top \left(I + P^{\beta^{\mathcal{X}}} + (P^{\beta^{\mathcal{X}}})^2 + \ldots\right) p_c^{\beta^{\mathcal{X}}} \\ &= \mathbb{E}_{\mathcal{P}\sim\mathcal{M}(\mathcal{X})}\left[\mathbb{I}_{\{\mathcal{P} \text{ converts}\}}\right] \\ &= \mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[\mathbb{I}_{\{\mathcal{P} \text{ converts}\}}\mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\}}\right], \end{aligned}$$

where $\mathcal{M}(\mathcal{X})$ denotes the Markov chain in which only the players in coalition $\mathcal{X}$ are "active".

Step 2: Use the representation from Step 1 in the definition of SV π_r^{Shap} as stated in (2):

$$\begin{aligned} \pi_r^{\text{Shap}} &= \sum_{\mathcal{X}\subseteq\mathbb{P}\setminus\{r\}} w_{|\mathcal{X}|} \times \{v(\mathcal{X}\cup\{r\}) - v(\mathcal{X})\} \\ &= \sum_{\mathcal{X}\subseteq\mathbb{P}\setminus\{r\}} w_{|\mathcal{X}|} \left\{\mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[\mathbb{I}_{\{\mathcal{P} \text{ converts}\}}\left(\mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\cup\{r\}\}} - \mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\}}\right)\right]\right\} \\ &= \mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[\mathbb{I}_{\{\mathcal{P} \text{ converts}\}} \sum_{\mathcal{X}\subseteq\mathbb{P}\setminus\{r\}} w_{|\mathcal{X}|}\left(\mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\cup\{r\}\}} - \mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\}}\right)\right]. \end{aligned}$$

Step 3: Use Lemma A.1, which states

$$\sum_{\mathcal{X}\subseteq\mathbb{P}\setminus\{r\}} w_{|\mathcal{X}|}\left(\mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\cup\{r\}\}} - \mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\}}\right) = \begin{cases} \frac{1}{u(\mathcal{P})} & \text{if } r \in \mathcal{P} \\ 0 & \text{otherwise,} \end{cases}$$

to conclude that

$$\pi_r^{\text{Shap}} = \mathbb{E}_{\mathcal{P}\sim\mathcal{M}}\left[w_r(\mathcal{P})\right]. \tag{8}$$

This completes the proof. □

Lemma A.1 (Unique players). *For any player $r \in \mathbb{P}$ and any path $\mathcal{P}$ over players in $\mathbb{P}$, we have*

$$\sum_{\mathcal{X}\subseteq\mathbb{P}\setminus\{r\}} w_{|\mathcal{X}|}\left(\mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\cup\{r\}\}} - \mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\}}\right) = \begin{cases} \frac{1}{u(\mathcal{P})} & \text{if } r \in \mathcal{P} \\ 0 & \text{otherwise,} \end{cases}$$

where the function $u(\cdot)$ returns the number of unique players.

Proof. First, suppose $r \notin \mathcal{P}$. Then, $\mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\cup\{r\}\}} = \mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\}}$ for all $\mathcal{X} \subseteq \mathbb{P}\setminus\{r\}$ and hence, the l.h.s. equals 0, which equals the r.h.s.

Second, suppose $r \in \mathcal{P}$. Assume w.l.o.g. that path $\mathcal{P}$ has $u+1$ unique players: $r, r_1, \ldots, r_u$. Observe that $\mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\}} = 0$ for all $\mathcal{X} \subseteq \mathbb{P}\setminus\{r\}$ and hence, the l.h.s. equals $\sum_{\mathcal{X}\subseteq\mathbb{P}\setminus\{r\}} w_{|\mathcal{X}|}\mathbb{I}_{\{\mathcal{P}\subseteq\mathcal{X}\cup\{r\}\}}$. Furthermore, since $r \in \mathcal{P}$, we can define a new path $\mathcal{P}' := \mathcal{P}\setminus\{r\}$ and express the l.h.s. as $\sum_{\mathcal{X}\subseteq\mathbb{P}\setminus\{r\}} w_{|\mathcal{X}|}\mathbb{I}_{\{\mathcal{P}'\subseteq\mathcal{X}\}}$. Now, we will count the number of times the combination $(r_1, \ldots, r_u)$ appears in the l.h.s. and weight it by $w_{|\mathcal{X}|}$ to compute the coefficient corresponding to $\mathcal{P}$ in the l.h.s. Trivially, if $|\mathcal{X}| < u$, then the combination will not appear (since there will not be enough number of unique players). For $|\mathcal{X}| \in \{u, \ldots, |\mathbb{P}|-1\}$, the contribution to the combination $(r_1, \ldots, r_u)$ equals

$$\underbrace{w_{|\mathcal{X}|}}_{\text{weight}} \times \underbrace{\binom{|\mathbb{P}|-1-u}{|\mathcal{X}|-u}}_{\#\text{ of times the combination appears in the summation over } \mathcal{X} \text{ for a fixed } |\mathcal{X}|}.$$

Summing over $|\mathcal{X}|$, the total contribution (which is the coefficient we are interested in) equals

$$\sum_{|\mathcal{X}|=u}^{|\mathbb{P}|-1} w_{|\mathcal{X}|} \times \binom{|\mathbb{P}|-1-u}{|\mathcal{X}|-u} = \frac{1}{|\mathbb{P}|-u} \sum_{|\mathcal{X}|=u}^{|\mathbb{P}|-1} \prod_{i=0}^{u-1} \frac{|\mathcal{X}|-i}{|\mathbb{P}|-i} = \frac{1}{u+1},$$

where the last equality follows Lemma A.2. This agrees with the r.h.s. and completes our proof. □

Lemma A.2. *Let k_1 be a non-negative integer and k_2 be a positive integer such that $k_2 \geq k_1 + 1$. Then,*

$$\frac{1}{k_2-k_1} \sum_{j=k_1}^{k_2-1} \prod_{i=0}^{k_1-1} \frac{j-i}{k_2-i} = \frac{1}{k_1+1}.$$

Proof. We will prove this via induction on k_2. For the base case, consider $k_2 = k_1 + 1$. The l.h.s. equals

$$\begin{aligned} \frac{1}{k_1+1-k_1} \sum_{j=k_1}^{k_1+1-1} \prod_{i=0}^{k_1-1} \frac{j-i}{k_1+1-i} &= \prod_{i=0}^{k_1-1} \frac{k_1-i}{k_1+1-i} \\ &= \frac{1}{k_1+1}, \end{aligned}$$

which is what we wanted. For the induction step, we show that if the statement is true for some $k_2 \geq k_1 + 1$, then it is also true for $k_2 + 1$, that is,

$$\frac{1}{k_2+1-k_1} \sum_{j=k_1}^{k_2} \prod_{i=0}^{k_1-1} \frac{j-i}{k_2+1-i} = \frac{1}{k_1+1}.$$

The l.h.s. of the above equation equals

$$\begin{aligned} &\frac{1}{k_2+1-k_1} \sum_{j=k_1}^{k_2} \prod_{i=0}^{k_1-1} \frac{j-i}{k_2+1-i} \\ &= \frac{1}{k_2+1-k_1}\left\{\sum_{j=k_1}^{k_2-1} \prod_{i=0}^{k_1-1} \frac{j-i}{k_2+1-i} + \prod_{i=0}^{k_1-1} \frac{k_2-i}{k_2+1-i}\right\} \\ &= \frac{1}{k_2+1-k_1}\left\{\sum_{j=k_1}^{k_2-1} \prod_{i=0}^{k_1-1} \frac{j-i}{k_2+1-i} + \frac{k_2-k_1+1}{k_2+1}\right\} \\ &= \frac{1}{k_2+1-k_1}\left\{\sum_{j=k_1}^{k_2-1} \prod_{i=0}^{k_1-1} \left(\frac{j-i}{k_2-i} \times \frac{k_2-i}{k_2+1-i}\right) + \frac{k_2-k_1+1}{k_2+1}\right\} \\ &= \frac{1}{k_2+1-k_1}\left\{\sum_{j=k_1}^{k_2-1} \left[\left(\prod_{i=0}^{k_1-1} \frac{j-i}{k_2-i}\right)\left(\prod_{i'=0}^{k_1-1} \frac{k_2-i'}{k_2+1-i'}\right)\right] + \frac{k_2-k_1+1}{k_2+1}\right\} \\ &= \frac{1}{k_2+1-k_1}\left\{\sum_{j=k_1}^{k_2-1} \left[\left(\prod_{i=0}^{k_1-1} \frac{j-i}{k_2-i}\right)\left(\frac{k_2-k_1+1}{k_2+1}\right)\right] + \frac{k_2-k_1+1}{k_2+1}\right\} \\ &= \frac{\frac{k_2-k_1+1}{k_2+1}}{k_2+1-k_1}\left\{\sum_{j=k_1}^{k_2-1} \prod_{i=0}^{k_1-1} \frac{j-i}{k_2-i} + 1\right\} = \frac{1}{k_2+1}\left\{\frac{k_2-k_1}{k_1+1} + 1\right\} = \frac{1}{k_1+1}, \end{aligned}$$

where the second-last equality is true due to the induction hypothesis. This completes the proof. □

REFERENCES

[1] Vibhanshu Abhishek, Stylianos Despotakis, and R Ravi. 2017. Multi-Channel Attribution: The Blind Spot of Online Advertising. Working paper, available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2959778 (2017).

[2] Vibhanshu Abhishek, Peter Fader, and Kartik Hosanagar. 2012. Media exposure through the funnel: A model of multi-stage attribution. Working paper, available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2158421 (2012).

[3] Eva Anderl, Ingo Becker, Florian Von Wangenheim, and Jan Hendrik Schumann. 2016. Mapping the customer journey: Lessons learned from graph-based online attribution modeling. *International Journal of Research in Marketing* 33, 3 (2016), 457–474.

[4] Sai Kumar Arava, Chen Dong, Zhenyu Yan, Abhishek Pani, et al. 2018. Deep Neural Net with Attention for Multi-channel Multi-touch Attribution. Working paper, available at: https://arxiv.org/abs/1809.02230 (2018).

[5] Nikolay Archak, Vahab S Mirrokni, and S Muthukrishnan. 2010. Mining advertiser-specific user behavior using adfactors. In *Proceedings of the 19th International Conference on World Wide Web*. ACM, 31–40.

[6] Joan Arensman and Wilfred Yeung. 2016. Move beyond last click attribution in AdWords. Retrieved Feb 8, 2018 from https://adwords.googleblog.com/2016/05/move-beyond-last-click-attribution.html

[7] Konstantin Avrachenkov, Laura Cottatellucci, and Lorenzo Maggi. 2012. Confidence intervals for Shapley value in Markovian dynamic games. Technical report (Eurecom), available at: http://www.eurecom.fr/publication/3602 (2012).

[8] Ron Berman. 2018. Beyond the last touch: Attribution in online advertising. *Marketing Science* 37, 5 (2018), 771–792.

[9] Alexander Bleier and Maik Eisenbeiss. 2015. Personalized online advertising effectiveness: The interplay of what, when, and where. *Marketing Science* 34, 5 (2015), 669–688.

[10] Javier Castro, Daniel Gómez, and Juan Tejada. 2009. Polynomial calculation of the Shapley value based on sampling. *Computers & Operations Research* 36, 5 (2009), 1726–1730.

[11] Hana Chockler and Joseph Y Halpern. 2004. Responsibility and blame: A structural-model approach. *Journal of Artificial Intelligence Research* 22 (2004), 93–115.

[12] Hana Choi, Carl Mela, Santiago Balseiro, and Adam Leary. 2017. Online Display Advertising Markets: A Literature Review and Future Directions. Working paper, available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3070706 (2017).

[13] John David Collins, Edward Jonathan Hall, Ned Hall, and Laurie Ann Paul. 2004. *Causation and counterfactuals*. MIT Press.

[14] Brian Dalessandro, Claudia Perlich, Ori Stitelman, and Foster Provost. 2012. Causally motivated attribution for online advertising. In *Proceedings of the Sixth International Workshop on Data Mining for Online Advertising and Internet Economy*. ACM, 7.

[15] Peter J Danaher and Harald J van Heerde. 2018. Delusion in Attribution: Caveats in Using Attribution for Multimedia Budget Allocation. *Journal of Marketing Research* 55, 5 (2018), 667–685.

[16] Ellery Eells. 1991. *Probabilistic causality*. Vol. 1. Cambridge University Press.

[17] Shaheen S Fatima, Michael Wooldridge, and Nicholas R Jennings. 2008. A linear approximation method for the Shapley value. *Artificial Intelligence* 172, 14 (2008), 1673–1699.

[18] Charles Miller Grinstead and James Laurie Snell. 2012. *Introduction to probability*. American Mathematical Society.

[19] Shelly Hagan. 2018. Digital economy has been growing at triple the pace of U.S. GDP. Retrieved Oct 11, 2018 from https://www.bloomberg.com/news/articles/2018-03-15/digital-economy-has-been-growing-at-triple-the-pace-of-u-s-gdp

[20] Joseph Y Halpern and Judea Pearl. 2005. Causes and explanations: A structural-model approach. Part I: Causes. *The British Journal for the Philosophy of Science* 56, 4 (2005), 843–887.

[21] Christopher Hitchcock. 1997. Probabilistic causation. (1997). Retrieved Feb 1, 2019 from https://plato.stanford.edu/entries/causation-probabilistic/

[22] David Hume. 2003. *A treatise of human nature*. Courier Corporation.

[23] Marketing Science Institute. 2016. Research Priorities 2016-2018. Retrieved Feb 8, 2018 from https://www.msi.org/uploads/articles/MSI_RP16-18.pdf

[24] Wendi Ji and Xiaoling Wang. 2017. Additional Multi-Touch Attribution for Online Advertising. In *AAAI*. 1360–1366.

[25] Wendi Ji, Xiaoling Wang, and Dell Zhang. 2016. A probabilistic multi-touch attribution model for online advertising. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1373–1382.

[26] Patrick Jordan, Mohammad Mahdian, Sergei Vassilvitskii, and Erik Vee. 2011. The multiple attribution problem in pay-per-conversion advertising. In *International Symposium on Algorithmic Game Theory*. Springer, 31–43.

[27] Lukáš Kakalejčík, Jozef Bucko, Paulo AA Resende, and Martina Ferencova. 2018. Multichannel Marketing Attribution Using Markov Chains. *Journal of Applied Management and Investments* 7, 1 (2018), 49–60.

[28] PK Kannan, Werner Reinartz, and Peter C Verhoef. 2016. The path to purchase and attribution modeling: Introduction to special section. *International Journal of Research in Marketing* 33, 3 (2016), 449–456.

[29] Hongshuang Li and PK Kannan. 2014. Attributing conversions in a multichannel online marketing environment: An empirical model and a field experiment. *Journal of Marketing Research* 51, 1 (2014), 40–56.

[30] Yiyi Li, Ying Xie, and Eric Zheng. 2017. Modeling Multi-Channel Advertising Attribution Across Competitors. Working paper, available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3047981 (2017).

[31] David Liben-Nowell, Alexa Sharp, Tom Wexler, and Kevin Woods. 2012. Computing shapley value in supermodular coalitional games. In *International Computing and Combinatorics Conference*. Springer, 568–579.

[32] Stephen C Littlechild and Guillermo Owen. 1973. A simple expression for the Shapley value in a special case. *Management Science* 20, 3 (1973), 370–372.

[33] Sasan Maleki, Long Tran-Thanh, Greg Hines, Talal Rahwan, and Alex Rogers. 2013. Bounding the estimation error of sampling-based Shapley value approximation. Working paper, available at: https://arxiv.org/abs/1306.4265 (2013).

[34] Tomasz P Michalak, Karthik V Aadithya, Piotr L Szczepanski, Balaraman Ravindran, and Nicholas R Jennings. 2013. Efficient computation of the Shapley value for game-theoretic network centrality. *Journal of Artificial Intelligence Research* 46 (2013), 607–650.

[35] Stephen L Morgan and Christopher Winship. 2014. *Counterfactuals and causal inference*. Cambridge University Press.

[36] Guillermo Owen. 1972. Multilinear extensions of games. *Management Science* 18, 5-part-2 (1972), 64–79.

[37] Judea Pearl. 2009. *Causality*. Cambridge University Press.

[38] Colin Priest. 2017. Multichannel Marketing Attribution with DataRobot. (2017). Retrieved Oct 11, 2018 from https://www.datarobot.com/resource/mktattribution/

[39] Quantcast. 2013. Beyond Last Touch: Understanding Campaign Effectiveness. (2013). Retrieved Oct 11, 2018 from http://info.quantcast.com/rs/quantcast/images/Quantcast%20White%20Paper_Beyond%20Last%20Touch.pdf

[40] Quantcast. 2016. Guide to marketing attribution. (2016). Retrieved Oct 11, 2018 from https://www.iabuk.com/sites/default/files/case-study-docs/Quantcast%20-%20Guide%20to%20Marketing%20Attribution%20-%20EN%20-%202016%20-%20print.pdf

[41] Donald B Rubin. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology* 66, 5 (1974), 688.

[42] Xuhui Shao and Lexin Li. 2011. Data-driven multi-touch attribution models. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 258–264.

[43] Lloyd S Shapley. 1953. A value for n-person games. *Contributions to the Theory of Games* 2, 28 (1953), 307–317.

[44] Sarah Sluis. 2018. Digital ad market soars to $88 billion, Facebook and Google contribute 90% of growth. Retrieved Oct 11, 2018 from https://adexchanger.com/online-advertising/digital-ad-market-soars-to-88-billion-facebook-and-google-contribute-90-of-growth/

[45] Econsultancy Digital Marketers United. 2012. Quarterly Digital Intelligence Briefing: Making Sense of Marketing Attribution (in association with Adobe). (2012). Retrieved Oct 11, 2018 from http://success.adobe.com/assets/en/downloads/whitepaper/Adobe-Quarterly-Digital-Intelligence-Briefing-Digital-Trends-for-2013.pdf

[46] Lizhen Xu, Jason A Duan, and Andrew Whinston. 2014. Path to purchase: A mutually exciting point process model for online advertising and conversion. *Management Science* 60, 6 (2014), 1392–1412.

[47] Ya Zhang, Yi Wei, and Jianbiao Ren. 2014. Multi-touch attribution in online advertising with survival theory. In *2014 IEEE International Conference on Data Mining (ICDM)*. IEEE, 687–696.

[48] Kaifeng Zhao, Seyed Hanif Mahboobi, and Saeed R Bagheri. 2018. Revenue-based attribution modeling for online advertising. *International Journal of Market Research* 61, 2 (2018), 195–209.

Urban Vibes and Rural Charms: Analysis of Geographic Diversity in Mobile Service Usage at National Scale

Rajkarn Singh
The University of Edinburgh
Edinburgh, UK
r.singh@ed.ac.uk

Marco Fiore
CNR-IEIIT
Turin, Italy
marco.fiore@ieiit.cnr.it

Mahesh K. Marina
The University of Edinburgh
Edinburgh, UK
mahesh@ed.ac.uk

Alessandro Nordio
CNR-IEIIT
Turin, Italy
alessandro.nordio@ieiit.cnr.it

Alberto Tarable
CNR-IEIIT
Turin, Italy
alberto.tarable@ieiit.cnr.it

ABSTRACT

We investigate spatial patterns in mobile service consumption that emerge at national scale. Our investigation focuses on a representative case study, *i.e.*, France, where we find that: (*i*) the demand for popular mobile services is fairly uniform across the whole country, and only a reduced set of peculiar services (mainly operating system updates and long-lived video streaming) yields geographic diversity; (*ii*) even for such distinguishing services, the spatial heterogeneity of demands is limited, and a small set of consumption behaviors is sufficient to characterize most of the mobile service usage across the country; (*iii*) the spatial distribution of these behaviors correlates well with the urbanization level, ultimately suggesting that the adoption of geographically-diverse mobile applications is linked to a dichotomy of cities and rural areas. We derive our results through the analysis of substantial measurement data collected by a major mobile network operator, leveraging an approach rooted in information theory that can be readily applied to other scenarios.

CCS CONCEPTS

• **Networks** → **Network services**; • **Social and professional topics** → *Geographic characteristics.*

KEYWORDS

Mobile service demands; mobile network traffic; spatial analysis

ACM Reference Format:
Rajkarn Singh, Marco Fiore, Mahesh K. Marina, Alessandro Nordio, and Alberto Tarable. 2019. Urban Vibes and Rural Charms: Analysis of Geographic Diversity in Mobile Service Usage at National Scale. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313628

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313628

1 INTRODUCTION

As mobile data traffic keeps surging worldwide [9], knowledge of where, when, how and why mobile services are consumed by network subscribers becomes increasingly relevant across research and technology domains, including sociology [4], demography [10], urban planning [18], economy [30], transportation engineering [34], or network management [23]. Despite the importance of the problem and some recent efforts discussed in Section 2, our comprehension of mobile service adoption is currently limited, and many questions remain unanswered, especially when considering the phenomenon at very large geographical scales. In this paper, we focus on one such open question, namely: *"how similar (or different) are demands for mobile services across a whole country?"* We answer by analyzing a real-world dataset of mobile network traffic collected by a major operator that describes the demands for individual services in 10,000 *communes* (*i.e.*, administrative areas) in France. Our study yields the following insights:

- what sets communes apart are not usage patterns of the most popular services, which tend to be similar everywhere in the country, but those of a small set of specific services that still figure in the top-50 services in terms of generated traffic, including operating system updates and long-lived video streaming;
- just 9 (respectively, 50) service consumption patterns are sufficient to retain 23% (respectively, 35%) of the overall usage diversity, implying that a small number of distinct behaviors is sufficient to characterize the many thousands of areas in the whole of France;
- clear correlations exist between different patterns in mobile service consumption and demographics features, which are rooted in higher (or lower) than average usage of specific types of service.

Deriving these results requires overcoming methodological and computational challenges. We face a clustering problem, where communes are to be grouped based on how their inhabitants use mobile services. However, our clustering operates in a multidimensional space of hundreds of mobile services, where a suitable notion of similarity is to be defined. In addition, working at a national scale implies potentially disentangling billions of pairwise relationships between tens of thousands of geographical areas.

We address these issues by adopting an information theoretic approach that builds on the notion of *mutual information* between mobile service demands and geographical locations. The mutual

information measures how much can be inferred about the consumed services by knowing the location, and vice versa. We first leverage it to limit the problem dimension in the mobile service space, by identifying *informative services* that maximize the mutual information, *i.e.*, exhibit significant spatial diversity. Also, we measure the similarity of usage distributions of such informative services between two communes in terms of the loss of mutual information incurred when their distributions are merged. Finally, the fraction of retained mutual information is used to assess the quality of clustering results obtained via a scalable two-phase approach.

Overall, our work sheds light on the limited set of services that are responsible for diversity in mobile data demands across a whole developed country, and on their relation to demographics features. It also provides the research community with a tool[1] for the analysis of patterns in mobile data traffic usage at national scales.

2 RELATED WORK

The vast majority of the literature on mobile network traffic analysis investigates patterns in the aggregate demand, without differentiating among services. In this context, early works have revealed the heterogeneity that characterizes the offered load at radio access in space and time, leading to strong fluctuations of the demand across diverse regions of a same city and during different periods of the day [26]. Especially relevant to our study are works that established links between the temporal dynamics of aggregate traffic and the land use, *i.e.*, the type of human infrastructures and activities present in a given area [8, 16, 32]. Although they focus on aggregate traffic at city scale, these studies have demonstrated for the first time the strong impact that user centric aspects can have on the adoption of mobile applications.

At the individual mobile service level, several works have studied specific applications in depth. Investigations have focused on services such as adaptive-bit-rate video streaming by over-the-top providers [12], Facebook and WhatsApp [15], or cloud storage [22]. These studies address the network-level performance of the examined services, and do not provide insights in terms of the geographical diversity of their usage. Related to these works is also a thorough analysis of web browsing behaviors by mobile users, which finds that a limited number of profiles are sufficient to capture most of the patterns in website visits [20]. However, similarly to the papers above, also in this case the spatial dimension is not considered.

Geographical diversity has been often overlooked also in prior explorations of mobile data traffic considering multiple services. Most works in the literature have a different focus, including differences in mobile application usage in time [35] or across the subscriber population [21]. Attention has also been paid to patterns in the utilization of apps by individual users, finding, *e.g.*, that mobile service usage is very heterogeneous among the user base [13], strongly depends on context [6], is characterized by brief bursts of interactions [14], and is influenced by the type of device used [19]. However, these are orthogonal problems to that of the spatial diversity of the demands for mobile services that we target. Finally, several previous works have observed a strong locality in the usage of applications within a given urban area, *i.e.*, a significant spatial diversity of usage across different city neighborhoods [29, 31]. Our investigation suggests that these dissimilarities in service usage are not significant at a national scale, where the consumption of mobile services is relatively uniform.

Only two previous works investigate the spatial dimension of mobile application usage at a national scale. In a study carried out in the US [33], the authors hint at the existence of local (*i.e.*, US state-related) and nationwide services. We do not find such a dichotomy in our case study, and ascribe it to the federal organization of the US into states, which is not reflected in France. Interestingly, however, the differences in the consumption of nationwide apps is fairly limited in the USA (between 2% and 20%), which is consistent with our findings. The second work is a recent study of mobile service usage in France [24], which unveils the existence of a strong temporal diversity in mobile service demands, but somewhat lower geographical differences. Our in-depth analysis substantiates the observations in [24] with stronger evidence of the limited nationwide diversity of mobile service consumption patterns. Finally, we remark that both works above only provide preliminary geographical results via baseline statistical measures, and do not perform a rigorous analysis based on spatial clustering as the one we propose. Moreover, none investigates the existence of informative services whose distribution is geographically varied, or offers interpretations of the results based on side information.

3 SYSTEM MODEL

As stated at the outset, our framework builds on information theory. In this section we introduce the notation and fundamentals of the proposed approach, and show how they apply to our scenario.

3.1 Probabilistic rendering of service demands

Let $C = \{1, \ldots, N_C\}$ and $\mathcal{S} = \{1, \ldots, N_S\}$ be the set of geographical areas in the target region and the set of mobile services under study, respectively, having cardinality N_C and N_S. A total mobile data traffic of t_i bytes is generated in area i, $i = 1, \ldots, N_C$, over the system observation time, *i.e.*, the time period during which network measurements are performed[2]. Let C be a random variable, with outcome in C, representing the selection of an area with a certain probability. Similarly, let S be a random variable, with outcome in $\mathcal{S}$, representing the selection of a service. In the considered period of time, the probability that a given byte of traffic was generated by service j in area i is the joint probability distribution of services and areas, denoted by $p_{S,C}(j, i)$, $j = 1, \ldots, N_S$, $i = 1, \ldots, N_C$.

Given an area i, the probability of observing a byte generated by service j is denoted by the conditional probability $p_{S|C}(j|i) = \rho_{j,i}$. The vector $\boldsymbol{\rho}_i = [\rho_{1,i}, \ldots, \rho_{N_S,i}]$ thus represents the *service usage distribution* in area i and is such that $\sum_{j=1}^{N_S} \rho_{j,i} = 1$.

Overall, the probability of observing traffic generated by service j, $j = 1, \ldots, N_S$, is represented by the marginal distribution of traffic among services:

$$p_S(j) = \sum_{i=1}^{N_C} p_{S,C}(j, i) = \sum_{i=1}^{N_C} p_C(i)\rho_{j,i},$$

[1] Available at https://github.com/rajkarn/mobdiv.

[2] As our interest is in the spatial diversity of mobile service usage, we primarily consider data accumulated over time. However, we also carried out experiments by partially disaggregating data in time, as discussed at the end of Section 6.2.

where $p_C(i)$, $i = 1, \ldots, N_C$ is the fraction of traffic in area i, among all areas, and so is given by:

$$p_C(i) = \frac{t_i}{\sum_{k=1}^{N_C} t_k}.$$

The amount of information that random variables S and C share is measured by the *mutual information*:

$$I(S;C) = H(S) - H(S|C), \tag{1}$$

where $H(S)$ and $H(S|C)$ are the entropy of S and the conditional entropy of S given C, respectively, expressed as:

$$H(S) = -\sum_{j=1}^{N_S} p_S(j) \log p_S(j),$$

and

$$H(S|C) = -\sum_{i=1}^{N_C} p_C(i) \sum_{j=1}^{N_S} p_{S|C}(j|i) \log p_{S|C}(j|i).$$

The mutual information $I(S;C)$ is a measure of the correlation between how traffic is distributed among services in S and among areas in C; *i.e.*, it captures how much can be inferred about the consumed services by knowing C, or vice versa. As a result, $I(S;C)$ measures how much mobile service usage depends on geographical location. $I(S;C) = 0$ when S and C are independent, *i.e.*, the exact same distribution of mobile service traffic is observed across all areas in C, and the spatial diversity is nil. Non-zero yet low values of mutual information of services and areas imply that sampling a unit of traffic from those services still carries little information on the area it was sampled from, *i.e.*, that services tend to have a quite uniform usage distribution across areas. As $I(S;C)$ grows, the knowledge of C increasingly helps to anticipate the value of S, *i.e.*, geographical regions are characterized by more and more distinctive mobile service usage, hence the spatial diversity rises.

3.2 Application to the France case study

We apply the probabilistic model above to the nationwide case study of France. The information on the traffic demands for individual mobile services in France was collected by a major network operator during one continuous week in late 2016. Deep packet inspection and proprietary fingerprinting techniques[3] were employed on traffic sniffed on the GPRS Tunneling Protocol (GTP), so as to associate single IP-level flows to applications. Overall, the dataset captures the usage of thousands of mobile services by 30 million subscribers in 10,000 *communes*, *i.e.*, local administrative zones with a mean surface of 16 km^2, in France.

The left plot in Fig. 1 provides an illustration of the joint probability $p_{S,C}(j,i)$, where $\mathcal{S}$ is the set of the 50 most popular mobile services[4], and C is the set of 10,000 communes. We remark the high unbalance in the traffic recorded across communes, as well as among services, underscored by the logarithmic scale of the z axis.

The middle plot in Fig. 1 shows the evolution of the mutual information $I(S_k;C)$, computed by limiting the joint distribution to a subset $\mathcal{S}_k \subseteq \mathcal{S}$ of the k services that generate the highest total traffic; equivalently, S_k denotes the random variable representing the selection of a service in $\mathcal{S}_k$. The number k of considered services is on the x-axis. We note that the mutual information remains very low, close to zero, for all top-50 services: according to our previous discussion, this implies that there is very little correlation between the two variables, hence the distribution of mobile service traffic does not vary in a sensible manner across communes.

A more detailed view is provided in the right plot in Fig. 1, which shows the joint distribution $p_{S,C}(j,i)$ in a linearized form where each "period" represents the probabilities associated to one service over all communes. The product of the marginal distributions $p_S(j) \cdot p_C(i)$ is also displayed according to the same format. There is a substantial overlap between the two curves, which confirms that spatial diversity is low also when inspecting the system on a more detailed per-service basis. Indeed, matching curves imply $p_{S,C}(j,i) = p_S(j) \cdot p_C(i)$, hence independence between S and C, or, equivalently, $I(S;C) = 0$ in (1). As discussed previously, such a condition indicates that service consumption is identical everywhere.

Overall, these results lead to our first takeaway message: ***usage patterns of the most popular mobile services tend to be very similar across the whole country under study***. This is quite a surprising outcome, considering the spatial differences in demographic, social and economic features that characterize France. It also leads us to investigate next if specific individual services are in fact geographically diverse in their usage characteristics.

4 DETECTING INFORMATIVE SERVICES

An interesting observation from the right plot of Fig. 1 is that, although the overlap between $p_{S,C}(j,i)$ and $p_S(j) \cdot p_C(i)$ is generally good, some services (denoted by specific "periods") show especially noisy joint distribution curves that are not well captured by the simple product of the marginal distributions. Such services thus appear to be adopted less homogeneously across the country than most other mobile applications. Next, we investigate the existence of *informative services* that are characterized by a non-negligible diversity of usage across geographical areas in the target region.

4.1 Maximizing the mutual information

Consider a subset $\mathcal{S}' = \{j_1, \ldots, j_{|\mathcal{S}'|}\} \subseteq \mathcal{S}$ of services and define the service usage distribution in area i restricted to $\mathcal{S}'$ as $\boldsymbol{\rho}_i(\mathcal{S}') = [\rho_{i1}(\mathcal{S}'), \ldots, \rho_{i,|\mathcal{S}'|}(\mathcal{S}')]$, where

$$\rho_{ik}(\mathcal{S}') = \frac{\rho_{i,j_k}}{\sum_{k'=1}^{|\mathcal{S}'|} \rho_{i,j_{k'}}}.$$

Considering only services that are in $\mathcal{S}'$, let us define the traffic within area i, denoted by $t_i(\mathcal{S}')$, as the sum of the demands within area i for each service in $\mathcal{S}'$. The joint probability of sampling area i and service $j_k \in \mathcal{S}'$ is then given by $p_C(i,\mathcal{S}')\rho_{ik}(\mathcal{S}')$, where we suppose $p_C(i,\mathcal{S}')$ to be the fraction of traffic in area i, *i.e.*,

$$p_C(i,\mathcal{S}') = \frac{t_i(\mathcal{S}')}{\sum_{i'=1}^{N_C} t_{i'}(\mathcal{S}')}.$$

[3] Confidential agreements with the network operator do not allow us to disclose the details of the traffic classification procedure. However, we can mention that 88% of the sessions were correctly detected, according to the operator's performance evaluations. Also, we underscore that the data collection occurred in compliance with regulations in force, and was approved by the French national authority for data privacy (CNIL). All data was aggregated by the operator at the commune level before we could access it, which ensures strong privacy protection as mobile traffic is accumulated over thousands of subscribers. No individual data is used in our study.

[4] We limit plots to the 50 services that generate the highest demands, for the sake of clarity. Such services account for over 93% of the total mobile data traffic in France.

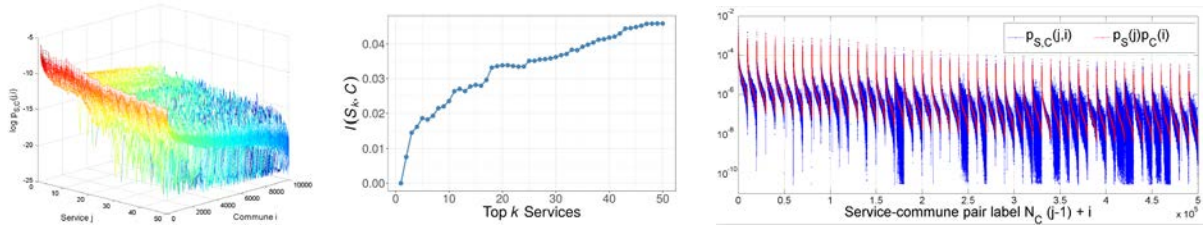


Figure 1: Probabilistic view of mobile service demands in France. Left: joint distribution $p_{S,C}(j,i)$ of services in S and communes in C. Middle: mutual information $I(S_k;C)$ computed on subsets $\mathcal{S}_k$ of the k services generating the most traffic. Right: unidimensional rendering of $p_{S,C}(j,i)$ (blue) and marginal distributions product $p_S(j) \cdot p_C(i)$ (red). Figure best viewed in colors.

We can now define informative services as a subset of all mobile services such that their mutual information with respect to the target spatial areas is maximized. Formally, the subset of informative services defined as above maps to the optimal choice of $\mathcal{S}' \subseteq \mathcal{S}$ that solves the following problem:

$$\mathcal{S}'_{\text{opt}} = \arg \max_{\mathcal{S}' \subseteq \mathcal{S}} I(C|_{\mathcal{S}'}; S|_{\mathcal{S}'}),$$

where $C|_{\mathcal{S}'}$ and $S|_{\mathcal{S}'}$ are the random variables representing the sampled area and service in the restricted scenario where only services in $\mathcal{S}'$ are considered.

Unfortunately, the above combinatorial problem is too complex to be solved exactly for typical data sizes. We thus adopt the following heuristic approach to determine $\mathcal{S}'$. Let us consider the whole set of services and sample service j from an arbitrary probability distribution $p_S(j)$. Then, along the lines of (1), we consider $I(C;S) = H(S) - H(C|S)$, where

$$H(C|S) = -\sum_{j=1}^{N_S} p_S(j) \sum_{i=1}^{N_C} p_{C|S}(i|j) \log p_{C|S}(i|j),$$

and area i is sampled with probability

$$p_{C|S}(i|j) = \frac{t_{i,j}}{\sum_{i'=1}^{N_C} t_{i',j}}.$$

The optimal service distribution p_S^* that weights services according to their informativeness is the one that solves

$$p_S^* = \arg \max_{p_S(j)} I(C;S),$$

which we obtain via the Blahut-Arimoto algorithm [1, 3]. We then sort the services according to decreasing values of p_S^* and we set a threshold θ. Then, $\mathcal{S}'$ contains those services for which $p_S^* > \theta$. The value of θ can be empirically chosen by inspecting the shape of p_S^*, as we will see next in the context of our reference scenario.

4.2 Application to the France case study

When applied to the France reference scenario, the approach returns the result in Fig. 2. Here, mobile services in $\mathcal{S}$ (x axis) are ranked based on their associated weights p_S^*. The differences in weights is striking, and spans tens of orders of magnitude: this implies that some services are significantly more informative than others. Specifically, there is a clear gap in the ranked values after the 13$^{\text{th}}$ service, highlighted by the vertical line, which indicates a substantial reduction of informativeness beyond that point. As the

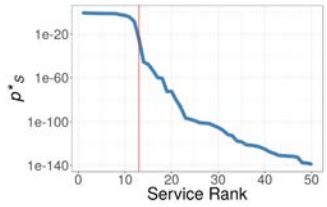


Figure 2: Mobile services ranked by the weighting distribution p_S^* returned by the Blahut-Arimoto approach in France.

Table 1: Set $\mathcal{S}'$ of informative services in France. Values within parentheses denote OS-specific traffic, while the absence of such values indicates OS-independent traffic.

p_S^* rank	Service name	Weekly traffic	/ Overall rank
1	torrent (Android)	8,070 GB	/ 43
2	Netflix (Android)	8,244 GB	/ 42
3	Netflix (iOS)	13,100 GB	/ 33
4	streaming (Android)	6,989 GB	/ 47
5	OS updates (iOS)	10,100 GB	/ 38
6	streaming (Windows Mobile)	11,800 GB	/ 35
7	streaming (iOS)	20,503 GB	/ 25
8	OS updates (Windows Mobile)	42,900 GB	/ 18
9	WhatsApp	8,821 GB	/ 40
10	OS updates (Android)	6,993 GB	/ 46
11	blogging	11,100 GB	/ 37
12	cloud storage (iOS)	11,300 GB	/ 36
13	SoundCloud	7,814 GB	/ 45

first 13 mobile services in the p_S^* ranking yield nearly equivalent informativeness, we include them all in the set $\mathcal{S}'$. These informative services are listed in Tab. 1, where we remark that:

- there exists a notable pattern in the selected mobile services, as they belong to specific classes, *i.e.*, operating system (OS) updates, and audio/video streaming;
- none of these informative services are among the top 15 in terms of total generated traffic, but they are all within the top 50, and amount to substantial network traffic loads of Terabytes per week.

Overall, those identified above are the mobile services that show substantial diversity in usage across France. Indeed, if S' is the random variable representing the selection of a service in $\mathcal{S}'$, the mutual information $I(S';C)$ is equal to 0.236, which is between 5 and 25 times that recorded from the top-k services in $\mathcal{S}_k$, for any $k \leq 50$, in the middle plot of Fig. 1. The findings above convey our second takeaway message: ***there exist a small set of applications that are actually informative of the geographical diversity in mobile service consumption, and which belong to fairly specific service categories.***

5 CLUSTERING MOBILE SERVICE USAGES

In order to understand how the informative services in $\mathcal{S}'$ are linked to the actual geography of France, we cluster the communes in C based on their mobile service usage distribution. Clustering based on distributions itself is non-trivial, and the scale of our scenario adds a layer of complexity. We build upon a recent breakthrough in social segregation analysis [7] to define the *weighted diversity* of two distributions, and use it in a scalable two-phase process.

5.1 Weighted diversity

Our problem essentially involves clustering geographical areas according to their service usage distribution. In our context, a clustering of areas with N_K clusters ($N_K \leq N_C$) is a map $\mathfrak{K} : C \to \mathcal{K}$ where $\mathcal{K} = \{1, \ldots, N_K\}$. We then define $J_{\mathfrak{K}}(k)$, $k = 1, \ldots, N_K$ as

$$J_{\mathfrak{K}}(k) = \{c \in C : \mathfrak{K}(c) = k\},$$

which is the set of geographical areas grouped into cluster k. Also, a clustering $\mathfrak{K}_1$ is a *refinement* of another clustering $\mathfrak{K}_2$ if and only if, for any two $c_1, c_2 \in C$, $\mathfrak{K}_1(c_1) = \mathfrak{K}_1(c_2)$ implies $\mathfrak{K}_2(c_1) = \mathfrak{K}_2(c_2)$.

Let $K = \mathfrak{K}(C)$ be the random variable representing the cluster in which a sampled area belongs to. The distribution p_K of clusters is induced by the distribution of areas p_C in a trivial way, *i.e.*,

$$p_K(k) = \sum_{i \in J_{\mathfrak{K}}(k)} p_C(i).$$

We can now define the mutual information between cluster and service random variables, denoted by $I(K, S)$. By the data processing inequality, we have $I(K; S) \leq I(C; S)$, since K is a deterministic function of C so that the knowledge of C implies the knowledge of K but not vice versa. More generally, we have the proposition:

Proposition 5.1. *If $\mathfrak{K}_1$ is a refinement of $\mathfrak{K}_2$ and $K_i = \mathfrak{K}_i(C)$, $i = 1, 2$, then*

$$I(K_2; S) \leq I(K_1; S)$$

Proof: For the proof, we first consider that $\mathfrak{K}_1$ has $N_K + 1$ clusters, $\mathfrak{K}_2$ has N_K clusters, $J_{\mathfrak{K}_1}(k) = J_{\mathfrak{K}_2}(k)$ for $k = 1, \ldots, N_K - 1$ and $J_{\mathfrak{K}_2}(N_K) = J_{\mathfrak{K}_1}(N_K) \cup J_{\mathfrak{K}_1}(N_K + 1)$. Then, the difference $I(K_1; S) - I(K_2; S)$ is given in (2), where $p_{K_2}(N_K) = p_{K_1}(N_K) + p_{K_1}(N_K + 1)$. Now, since mutual information is convex, we have as a consequence (3). This implies that $I(K_1; S) - I(K_2; S) \geq 0$.

When $\mathfrak{K}_1$ is a general refinement of $\mathfrak{K}_2$, we can always imagine a chain of refinements that starts from $\mathfrak{K}_2$ and ends to $\mathfrak{K}_1$, in which each step consists in splitting a cluster into two. For each of such steps, we can apply the argument above, and conclude again that $I(K_1; S) - I(K_2; S) \geq 0$. ∎

In the above proposition, the information loss $I(K_1; S) - I(K_2; S)$ can be seen as the price to pay for merging two clusters of $\mathfrak{K}_1$ into a single cluster of $\mathfrak{K}_2$. Based on this observation, and given a clustering $\mathfrak{K}$, we can introduce a pairwise cost measure for the joining of any two clusters $J_{\mathfrak{K}}(k_1)$ and $J_{\mathfrak{K}}(k_2)$, which will be central to the design of our information theory-based clustering algorithm. We name such a measure *weighted diversity*, and define it as follows.

Definition 5.1. *Given a spatial clustering $\mathfrak{K}$ with clusters $J_{\mathfrak{K}}(1)$, $\ldots$, $J_{\mathfrak{K}}(N_K)$, let us define a new clustering $\mathfrak{K}_{i,i'}$, with $N_K - 1$ clusters, obtained from $\mathfrak{K}$ by merging clusters $J_{\mathfrak{K}}(i)$ and $J_{\mathfrak{K}}(i')$. Moreover, let $K = \mathfrak{K}(C)$ and $K_{i,i'} = \mathfrak{K}_{i,i'}(C)$. We define the weighted diversity between $J_{\mathfrak{K}}(i)$ and $J_{\mathfrak{K}}(i')$ as*

$$d(i, i') = I(K; S) - I(K_{i,i'}; S). \tag{4}$$

In the above proposition, the information loss $I(K_1; S) - I(K_2; S)$ can be seen as the cost entailed by passing from a more refined description of service usage distribution given by clustering $\mathfrak{K}_1$ to a coarser description corresponding to $\mathfrak{K}_2$. Next, we employ the measure in (4) as the basis for a clustering algorithm.

5.2 Practical clustering algorithm

The weighted diversity measure can be leveraged as a distance metric for practical algorithms that aim at clustering geographical areas based on mobile service usage. It has the following desirable features: (*i*) it depends only on the two considered clusters i and i', and not on the other components of clustering $\mathfrak{K}$, as per (2), hence it is a suitable distance metric for clustering; (*ii*) its definition and properties are independent of how $\mathfrak{K}$ is obtained, hence it can be used in combination with any clustering algorithm; and, (*iii*) it is specifically designed for computing the dissimilarity of two distributions, *i.e.*, the data representation that characterizes our system. In addition, the metric has a clear interpretation in information theory terms, as it maps to the loss of information incurred by merging the mobile service usage distributions of two geographical areas into one. Two areas with identical service usage will be characterized by a null weighted diversity, and merging them into the same cluster will preserve the original distributions without any loss of information. Instead, two areas with very diverse service consumption will have a high weighted diversity, and joining them in a same cluster will lose the specificity of the original distributions.

According to (*ii*) above, we can embed weighted diversity as a similarity measure in any clustering algorithm. In this work, we opt for a greedy *divide-et-impera* solution, which, unlike legacy (*e.g.*, spectral, agglomerative, modularity-based) clustering techniques, avoids computing pairwise weighted diversities for the billion edges in the complete mesh of geographical areas. The algorithm, outlined in Alg. 1, separates the problem in two phases as follows.

5.2.1 Phase I. In the first phase, we start by initializing the clustering $\mathfrak{K}$ to the set of communes **C**, *i.e.*, considering each commune in a separate cluster (line 2). We then compute the dissimilarity matrix **D** among all communes in **C** by initializing all values to ∞ (line 3), and then updating the actual weighted diversity via the expression in (4) only between pairs of adjacent communes (lines 4-6). An important remark is that the matrix **D** is very sparse, since communes typically have a fairly small number of neighboring areas (*e.g.*, less than ten), which is orders of magnitude lower than the cardinality of **C**: therefore, populating **D** with weighted diversities is dramatically faster than computing the weighted diversity for all pairs of areas in the target region. At this point, matrix **D** represents a sparse, weighted graph on which we can run any practical algorithm `cluster` to produce the desired grouping of the original communes in the starting $\mathfrak{K}$ into N_{K_1} clusters (line 7). In our implementation, we opt for a simple hierarchical clustering technique also presented in Alg. 1. This is a traditional greedy approach [2, 11]: until the desired number of clusters is obtained (line 16), it proceeds by identifying the two areas (or clusters) with minimum weighted

$$I(K_1;S) - I(K_2;S) = H(S|K_2) - H(S|K_1)$$
$$= \sum_{k=1}^{N_K} p_{K_2}(k) H(S|K_2 = k) - \sum_{k=1}^{N_K+1} p_{K_1}(k) H(S|K_1 = k)$$
$$= p_{K_2}(N_K) H(S|K_2 = N_K) - p_{K_1}(N_K) H(S|K_1 = N_K) - p_{K_1}(N_K + 1) H(S|K_1 = N_K + 1) \quad (2)$$
$$H(S|K_2 = N_K) \geq \frac{p_{K_1}(N_K)}{p_{K_1}(N_K) + p_{K_1}(N_K + 1)} H(S|K_1 = N_K) + \frac{p_{K_1}(N_K + 1)}{p_{K_1}(N_K) + p_{K_1}(N_K + 1)} H(S|K_1 = N_K + 1) \quad (3)$$

Algorithm 1: Two-phase clustering algorithm pseudocode.

input: **C**, set of geographical areas, *i.e.*, communes
input: **A**, adjacency matrix of areas in **C**
input: N_{K_1}, N_{K_1} target number of clusters in phases I and II

1 **procedure** `twoPhaseClustering`(**C**, **A**)
2 $\quad \mathfrak{K} : C \to C$ s.t. $\mathfrak{K}(c) = c$, $\forall c \in C$
3 $\quad \mathbf{D} \in \mathbb{R}^{N_C \times N_C} \leftarrow \infty$
4 $\quad$ **foreach** $(i, i') \in \mathbf{A}$ **do**
5 $\quad\quad \mathbf{D}(i, i') =$ `weightedDiversity`$(J_{\mathfrak{K}}(i), J_{\mathfrak{K}}(i'))$
6 $\quad$ **end**
7 $\quad \mathfrak{K}_1 \leftarrow$ `cluster`$(\mathfrak{K}, \mathbf{D}, \mathbf{A}, N_{K_1})$
8 $\quad \mathbf{D}_1 \in \mathbb{R}^{N_{K_1} \times N_{K_1}} \leftarrow \infty$
9 $\quad$ **foreach** $(i, i') \in \mathbf{D}_1$ **do**
10 $\quad\quad \mathbf{D}_1(i, i') =$ `weightedDiversity`$(J_{\mathfrak{K}}(i), J_{\mathfrak{K}}(i'))$
11 $\quad$ **end**
12 $\quad \mathfrak{K}_2 \leftarrow$ `cluster`$(\mathfrak{K}_1, \mathbf{D}_1, \mathbf{A}, N_{K_2})$
13 $\quad$ **return** $\mathfrak{K}_2$
14 **end**
15 **procedure** `cluster`$(\mathfrak{K}, \mathbf{D}, \mathbf{A}, N_K)$
16 $\quad$ **while** $|\mathfrak{K}| > N_K$ **do**
17 $\quad\quad (i^*, i'^*) \leftarrow \arg\min_{(i,i')} \mathbf{D}(i, i')$
18 $\quad\quad \mathfrak{K} \leftarrow \mathfrak{K} \setminus J_{\mathfrak{K}}(i^*), J_{\mathfrak{K}}(i'^*)$
19 $\quad\quad J \leftarrow J_{\mathfrak{K}}(i^*) \bigcup J_{\mathfrak{K}}(i'^*)$
20 $\quad\quad \mathfrak{K} \leftarrow \{\mathfrak{K}, J\}$
21 $\quad\quad$ **foreach** $i < |\mathfrak{K}|$ **do**
22 $\quad\quad\quad$ `remove`$(\mathbf{D}, (i, i^*), (i, i'^*))$
23 $\quad\quad\quad$ **if** $\exists (j, j') \in \mathbf{A}$, *s.t.* $j \in J_{\mathfrak{K}}(i)$, $j' \in J$ **then**
24 $\quad\quad\quad\quad \mathbf{D}(i, |\mathfrak{K}|) =$ `weightedDiversity`$(J_{\mathfrak{K}}(i), J_{\mathfrak{K}}(|\mathfrak{K}|))$
25 $\quad\quad\quad$ **end**
26 $\quad\quad$ **end**
27 $\quad$ **end**
28 $\quad$ **return** $\mathfrak{K}$
29 **end**

diversity (line 17), removing them from the current set of clusters (line 18), performing a merge of the geographical areas and associated mobile service usage distributions (line 19), and finally adding the new cluster to the updated $\mathfrak{K}$ (line 20). The dissimilarity matrix is also updated, by removing the weighted diversity values associated to the two merged areas, and computing and adding to **D** the weighted diversities between the newly created cluster J and all adjacent (merged) regions in the current $\mathfrak{K}$ (line 21-26).

At the end of this phase, a specific clustering $\mathfrak{K}_1$ of N_{K_1} merged communes is selected. Typical approaches for picking N_{K_1} rely on expert knowledge of the system, or stopping rules [25]. Instead, we opt for a simple and pragmatic strategy. Each iteration of the `cluster` algorithm decreases N_K by generating a refinement of the previous clustering, which, according to Definition 5.1, retains lower or equal information than that available at the previous step. Thus, a larger N_{K_1} is always a better choice, and N_{K_1} can be straightforwardly set to the order of the largest graph that is computationally manageable during the second phase.

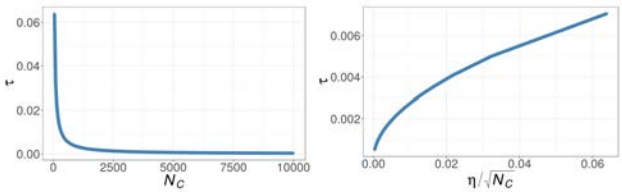


Figure 3: France scenario. Left: variation of τ with number of communes. Right: relationship between τ and $1/\sqrt{N_C}$.

5.2.2 Phase II. In the second phase, we build a clique, *i.e.*, fully connected mesh, of the clusters received from the first phase, and compute the weighted diversity for all pairs of clusters in $\mathfrak{K}_1$ (lines 8-11). Note that, unlike in the first phase, this is now a feasible operation, as N_{K_1} is selected by accounting for computational feasibility.

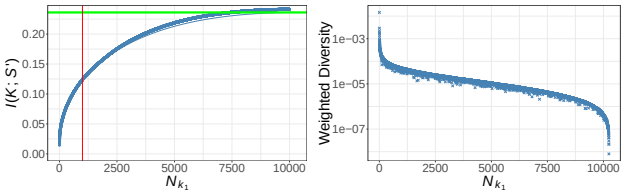
We then run the `cluster` algorithm again on the new graph (line 12). This operation allows breaking the spatial proximity constraint imposed during phase I: areas that are geographically distant and yet show similar mobile service distributions can now be grouped together in a scalable way. Using the weighted diversity as an edge weight metric returns an easily interpretable view of the information loss at each refinement, and allows for an educated choice of the eventual number of clusters to retain, N_{K_2}. We provide an example of this property in the France case study, in Section 5.3.

5.2.3 Complexity analysis. The function `weightedDiversity` operates on the conditional distributions $p_{S|C}(j|i) = \rho_{j,i}$ of S on C. Thus, its complexity is linear with respect to the size of the outcome set of S, *i.e.*, $O(N_S)$ for any pair of geographical areas.

During phase I, the calculation of `weightedDiversity` is first repeated for all non-zero elements of the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N_C \times N_C}$ to obtain **D**, which yields a complexity $O\left(N_S(\tau N_C)^2\right)$. Here, τ is the squared fractional average degree in the adjacency graph described by matrix **A**: it denotes the sparsity of the matrix due by the geographical topology of the target region. Then, each iteration of the `cluster` algorithm requires recomputing all weighted diversities for the neighboring areas of those selected for merging, with complexity $O\left(N_S(\tau N_C)\right)$. Overall, this leads to a complexity of the first phase $O\left(N_S(\tau N_C)^2 + N_S(\tau N_C)\right) = O\left(N_S(\tau N_C)^2\right)$.

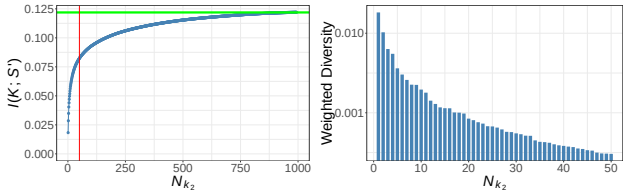
In phase II, the same operations above are repeated on a different, fully connected graph with a reduced number of nodes equal to the clusters from the first phase. Hence, the complexity is $O\left(N_S N_{K_1}^2\right)$. As $N_{K_1} \ll N_C$, we can approximate the total complexity of the algorithm in the two phases as $O\left(N_S(\tau N_C)^2\right)$.

A key remark is that, in planar spatial graphs such as those we consider, τ is typically an extremely low number that scales approximately as $1/\sqrt{N_C}$ when N_C grows [5]. This effectively reduces the complexity of the proposed two-phase algorithm to $O\left(N_S N_C\right)$,



(a) Mutual Information (b) Weighted diversity

Figure 4: Phase I of the clustering algorithm. Left: Mutual information $I(K;S')$ retained by the clustering versus the number of returned clusters N_{K_1}. Right: weighted diversity of the two communes (or clusters) aggregated at each step.



(a) Mutual Information (b) Weighted diversity

Figure 5: Phase II of the clustering algorithm. Left: Mutual information $I(K;S')$ retained by the clustering versus the number of returned clusters N_{K_2}. Right: weighted diversity of the two communes (or clusters) aggregated at each step.

making it extremely efficient in large-scale scenarios. Evidence of the scalability of the approach in is provided in Fig. 3, for the France nationwide scenario. The left plot shows the evolution of τ with respect to N_C in the case of the adjacency matrix of communes in France, when considering an increasingly larger portion of the country, *i.e.*, higher N_C. The curve confirms the scaling property indicated above, which is even more clearly seen in the right plot of Fig. 3, where τ is shown to scale as $\eta/\sqrt{N_C}$ with $\eta = 0.05$.

5.3 Results in the France case study

The two-phase clustering introduced in Section 5.2 above enables the investigation of mobile service usage across the whole country of France. In the light of the results in Section 4.2, it makes sense to limit the analysis to the set of informative services $\mathcal{S}'$ that yield non-negligible geographical diversity.

5.3.1 Phase I analysis. Fig. 4 shows the mutual information $I(K;S')$ retained by all clusterings $\mathfrak{K}_1$ in the hierarchical structure formed by the greedy approach during the first phase of the algorithm. Looking at each of these plots from right to left allows imagining how the algorithm works. Specifically, $I(K;S')$ is illustrated as a function of N_{K_1}, *i.e.*, the number of clusters. In the left plot, we note that $I(K;S')$ grows with N_{K_1}, as expected from Definition 5.1, until it reaches the complete mutual information $I(C;S')$ (green horizontal line) that characterizes the system of $N_{S'}$ (*i.e.*, 13) services and N_C (*i.e.*, 10,000) communes in France, when $N_{K_1} = N_C$.

The curve grows faster at first, to slow down afterwards: this means that the majority of the information is retained by the very first clusters, or, equivalently, that the latest iterations of the agglomerative clustering are those that lose the highest informations. This is highlighted in the right plot of Fig. 4, where the merging of the few tens of clusters (on the left) results in an information loss, quantified by the weighted diversity, which is orders of magnitude higher than that incurred at earlier stages of the algorithm (to the right). The diversity gently degrades across the vast majority of N_{K_1} values, with the exception of the last clusterings: this implies that only the opening aggregations do not lose information.

5.3.2 Phase II analysis. As recommended in Section 5.2, we retain the maximum number of clusters for which we can afford the computationally expensive processing of the second phase. We thus consider $N_{K_1} = 1,000$, which is highlighted by the vertical line in Fig. 4, and captures 52% of the original mutual information. This means that we are bounding the computational complexity of the analysis in phase II to operations on graphs with 500,000 edges.

Table 2: Urbanization levels categorizing French communes. Metropolis, medium- and small-sized cities are based on the number of workplaces. Intermediate classes are obtained by geographical adjacency to the three types of urban areas.

Level	*Urbanization class*	*Workplaces*	*Adjacency*
1	Large metropolis	10,000 and more	-
2	Large metropolis suburbs	-	to 1
3	Large metropolis influence area	-	to 2
4	Medium-sized city	5,000 - 10,000	-
5	Medium-sized city suburbs	-	to 4
6	Small-sized city	1,500 - 5,000	-
7	Small-sized city suburbs	-	to 6
8	Town	-	to 7
9	Rural area	-	-

Equivalent curves to those for the first phase, illustrating the output of the clustering on the fully connected mesh of 1,000 clusters of communes from the first phase, are shown in Fig. 5. Interestingly, the mutual information curve, in the left plot, grows much faster with N_{K_2} than it did with N_{K_1} during phase I. We ascribe this phenomenon to the fact that the clique representation removes all geographical constraints, and grants higher flexibility during the clustering process, allowing matching and merging (clusters of) communes that are spatially distant but showing fairly correlated mobile service distributions. Therefore, the result implicitly proves that similar mobile application usages often occur in areas located at significant geographical distance in France.

The sudden increase of the mutual information also lets us take an easy, informed choice about a reasonable number of clusters: *e.g.*, by selecting $N_{K_2} = 50$ (the red vertical line in Fig. 5), it is possible to retain 67.5% of the mutual information of the 1,000 clusters considered after first phase, and 35% of the total mutual information of the system. Or, a very small $N_{K_2} = 9$ preserves 44% and 23% of the mutual information in the two cases. The detailed weighted diversity values are in the right plot of Fig. 5.

Overall, the observations above let us formulate our third takeaway message: ***a small number of commune clusters (e.g., 9, equal to a fraction 0.0009 of around 10,000 communes considered in the analysis) is sufficient to retain a substantial percentage (e.g., over 20%) of the diversity observed in the usage of mobile services in France***. These figures refer to the set $\mathcal{S}'$ of services that yield the highest spatial heterogeneity. Then, the takeaway above implies that there exists a fairly limited set of typical behaviors in the usage of mobile services across the whole country, even when considering only those applications that show substantial geographic diversity.

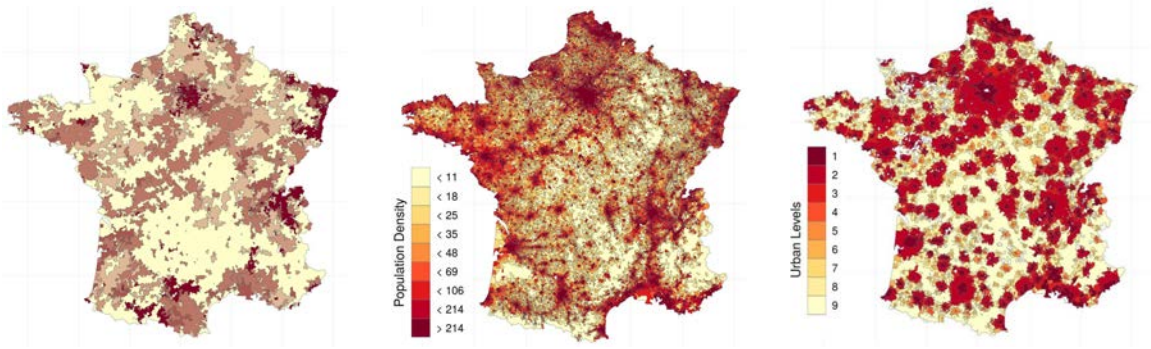
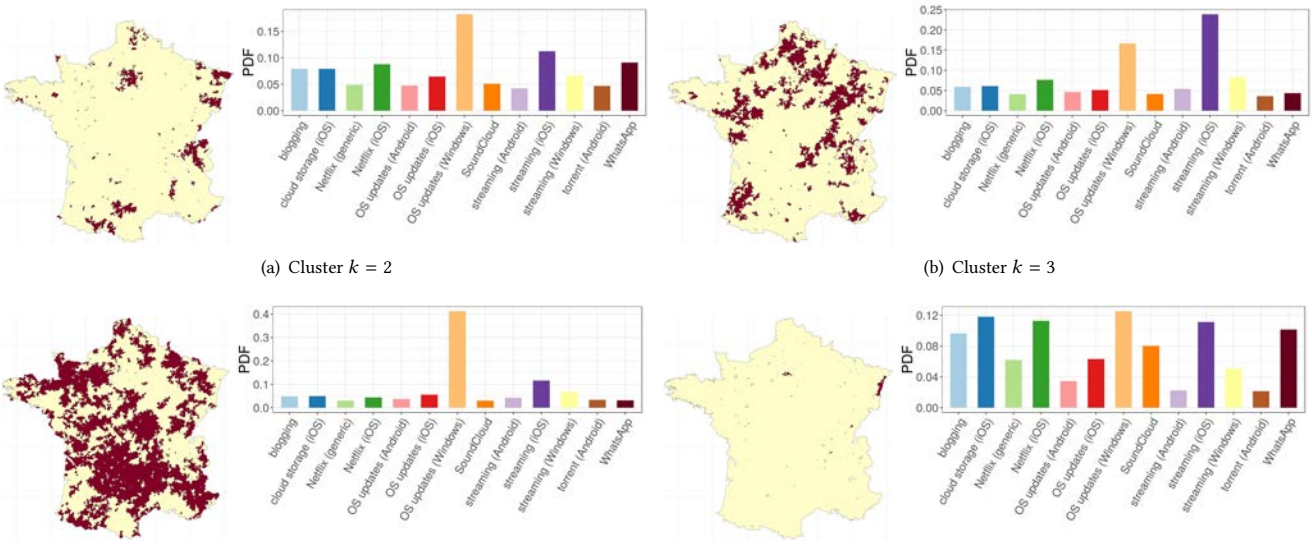


Figure 6: Maps of France. Left: geographical layout of the $N_{K_2} = 9$ clusters produced by our algorithm, each identified by a color. Middle: population density levels (in people per square kilometer). Right: urbanization levels. Figure best viewed in colors.



(a) Cluster $k = 2$

(b) Cluster $k = 3$

(c) Cluster $k = 5$

(d) Cluster $k = 9$

Figure 7: Geographical coverage of four representative clusters, and associated PDF $\rho_{j,k}$ of the informative service demands.

6 INTERPRETATION OF RESULTS

In order to better understand the results obtained so far, we investigate the links between the clustered mobile service usages and the country demographics in the representative case of $N_{K_2} = 9$.

6.1 French geography of mobile service usage

The left plot in Fig. 6 provides an illustration of how clusters are associated to different geographical areas in France. The spatial pattern of clusters is not random. For instance, one cluster clearly tells apart the Paris metropolitan area from its surroundings (dark patch at the center top of the map); or, most of the areas in central France are clustered together in a large continuous region (wide light territory at center bottom of the map).

In fact, a simple visual inspection reveals that the map of clusters yields substantial resemblances to those of two important demographic measures: (*i*) population density, *i.e.*, the number of dwelling units per commune, measured in inhabitants/Km^2; (*ii*) the urbanization level, *i.e.*, a categorization of communes based on number of workplaces and mutual geographical adjacency, which results in the nine categories in Tab. 2 [17]. The likeness is evident when comparing the left plot in Fig. 6 with the middle and right plots in the same figure, which respectively portray the population density and urbanization levels for all communes in France.

Also, it is interesting to observe how the most informative services are consumed in these clusters. Fig. 7 shows the Probability Density Function (PDF) $p_{S'|K}(j|k) = \rho_{j,k}$ of services $j \in \mathcal{S}'$ in a selected subset of clusters $k \in \mathcal{K}$. For the sake of clarity, the PDFs are accompanied by maps displaying the regions included in each cluster. We observe that the distributions yield significant differences, and characterize very heterogeneous surfaces. For instance, clusters 3 and 5 cover large regions all over France that appear to be fairly complementary; the former happens to be characterized by a distinctively high usage of streaming services, while the latter has a high incidence of background traffic generated by smartphones that run Windows Mobile as the operating system. Instead, clusters 2 and 9 cover much smaller areas in France, and have more balanced distributions across all services.

6.2 Linking services and demographics

The qualitative analysis above suggests: (*i*) an apparent correlation between the geographical layout of the clusters and the intensity of human presence (captured by the population density and urbanization level); and (*ii*) striking differences in the way specific services are consumed across clusters. By combining these observations, the identified clusters allow bonding mobile services to demographics.

6.2.1 Methodology. Let $\mathcal{D} = \{1, \ldots, N_D\}$ be a set of demographics levels (either discretized population density levels, or urbanization levels), with cardinality N_D. All geographical areas $i \in C$ are assigned a unique level in $d \in \mathcal{D}$. We can then define as D the random variable with outcome in $\mathcal{D}$ that denotes the probability that a given area i is characterized by a specific demographics level. The joint distribution of demographics levels and clusters is $p_{D,K}(d,k)$, and we can compute conditional probabilities $p_{D|K}(d|k)$ that describe the distribution of individual demographics levels within cluster k.

We now define *significance vectors* $\boldsymbol{\rho}_k^D(d)$ and $\boldsymbol{\rho}_k^{S'}(j)$ from the distributions $p_{D|K}(d|k) = \rho_{d,k}$ and $p_{S'|K}(j|k) = \rho_{j,k}$, as:

$$\boldsymbol{\rho}_k^D(d) = \rho_{d,k} - \frac{1}{N_K - 1} \sum_{k' \in \mathcal{K}, k' \neq k} \rho_{d,k'}, \quad \forall k \in \mathcal{K},$$

$$\boldsymbol{\rho}_k^{S'}(j) = \rho_{j,k} - \frac{1}{N_K - 1} \sum_{k' \in \mathcal{K}, k' \neq k} \rho_{j,k'}, \quad \forall k \in \mathcal{K}.$$

The significance vector $\boldsymbol{\rho}_k^D(d)$ (respectively, $\boldsymbol{\rho}_k^{S'}(j)$) associated to cluster k thus assigns a weight bounded in $[-1, 1]$ to each demographics level (respectively, service). Such weight indicates how much the fraction of communes in the clusters associated to one demographics level (respectively, the demand for a specific service in the cluster) differ, positively or negatively, from the average across all clusters. Example illustrations are provided in Fig. 8 and Fig. 9. Fig. 8 shows the significance vectors $\boldsymbol{\rho}_k^{S'}(j)$ derived from the conditional service distributions at the four representative clusters in Fig. 7. Fig. 9 portrays the significance vectors $\boldsymbol{\rho}_k^D(d)$ of the same four clusters, with respect to urbanization levels. We can observe, for instance, that clusters 2 and 9 show similar patterns in $\boldsymbol{\rho}_k^{S'}(j)$ and are much more present in dense urban areas. Instead, cluster 5 is characterized by very high incidence of Windows Mobile updates, as well as by a striking presence in rural regions.

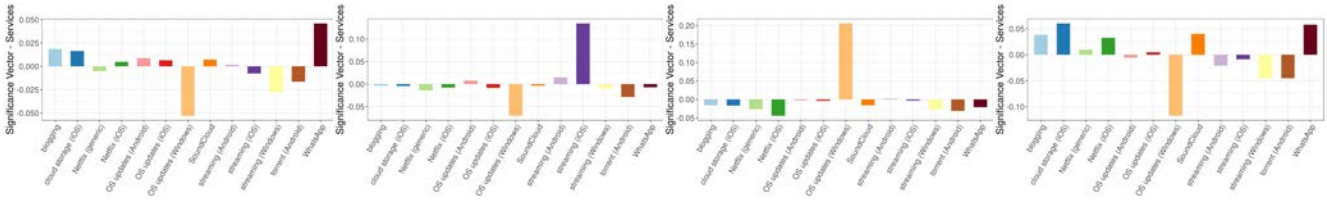
The extent of inter-dependency among services and demographics levels can be computed in a rigorous way for each cluster $k \in \mathcal{K}$, by means of the matrix multiplication $\boldsymbol{\rho}_k^D(d) \left[\boldsymbol{\rho}_k^{S'}(j)\right]^T$, where $[\cdot]^T$ is the matrix transposition operation. This returns an *incidence matrix* $\mathbf{M}_k(d,j) \in \mathbb{R}^{N_D \times N_{S'}}$, where element (d,j) is a value in $[-1,1]$ that indicates the peculiarity of demands for service j in demographics level d, as conveyed by cluster k. Highly positive (resp., negative) values point at abnormally high (resp., low) incidence of the service in the demographics level. Finally, it is possible to derive a single incidence matrix for the overall clustering $\mathfrak{K}$, by simply averaging over all clusters $k \in \mathcal{K}$, *i.e.*, $\mathbf{M}(d,j) = \frac{1}{N_{K_2}} \sum_{k \in \mathcal{K}} \mathbf{M}_k(d,j)$.

6.2.2 Results and discussion. Examples of the matrices $\mathbf{M}(d,j)$ obtained with the clustering $\mathfrak{K}$ portrayed in the left plot of Fig. 6, where $N_{K_2} = 9$, are in the left and middle plots of Fig. 10, for demographics levels obtained with population density percentiles and urbanization levels, respectively. The matrices confirm that some informative services are prone to an higher-than-average use in areas characterized by specific demographics levels; for instance, Windows system updates and streaming services have higher incidence in rural areas, and a lower impact on urbanized areas; conversely, WhatsApp is widely adopted in metropolitan areas, but shows lower-than-average usage in the countryside. Interestingly, the two metrics used to derive demographics levels, in the left and middle plots of Fig. 10, appear to yield consistent views across services. We provide a joint representation in the right plot of Fig. 10, where application $j \in \mathcal{S}'$ is located in the bidimensional space of population density and urbanization levels, by assigning to it coordinates $\frac{1}{N_D} \sum_{d \in \mathcal{D}} d \cdot \mathbf{M}(d,j)$, one for each notion of $\mathcal{D}$. The plot confirms our observation of a clear association of specific informative services to different demographics, providing an interesting and consistent ranking of applications versus urbanization.

Specifically, it is apparent that people living in rural regions of France have a preference to use Windows Mobile devices. Automatic updates for such OS are especially characterizing of the spatial diversity, due to a lower overall traffic per user in rural regions, which makes background traffic stand out. Instead, inhabitants of French metropolitan areas prefer Apple iPhones, as iOS-only services have a higher incidence than normal in cities. Residents in French cities also display a remarkable tendency to significantly use WhatsApp, a popular messaging application, and long-lived streaming services such as Netflix.

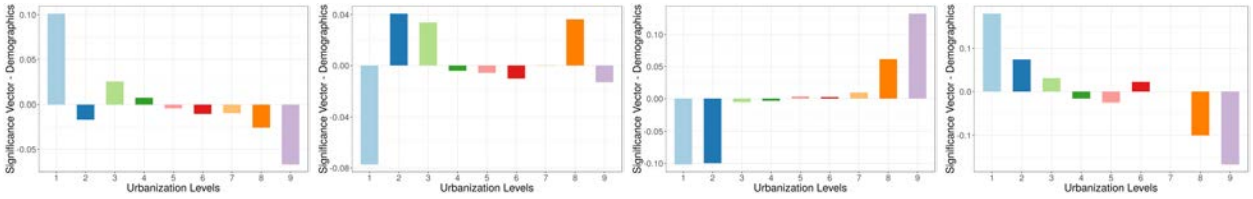
We underscore that the diversity of mobile service usage observed above is not an artifact of the different availability of radio access technolologies in urban and rural areas of France. The observed diversity of OS-specific traffic is a first evidence: there is no reason why lower or higher mobile datarates should affect adoption of Windows Mobile devices rather than Apple iPhones. To further prove our point, we leverage 2G, 3G, and 4G coverage maps provided by ARCEP, the French agency in charge of regulating telecommunications in France[5], as well as data on the deployment of 1.8 million Wi-Fi home access points owned by Free, an incumbent Internet service provider in the country [28]. Both datasets refer to around the same period of the mobile service traffic collection. The left plot in Fig. 11 shows that 4G coverage (dark blue) was already pervasive in France at the end of 2016: most of the national territory enjoyed broadband mobile access, including many regions tagged as suburban and rural by the demographics levels in Fig. 6. In the relatively small portion of geographic surface without 4G access, 3G was available, and 2G-only coverage areas were basically absent. Therefore, the cellular access technology cannot be considered as a discriminant for the geographical diversity of usages observed in our study for specific services. Also, the right plot in Fig. 11 shows that Wi-Fi presence, measured in available access points per person, is largely uniform across communes; it ranges between 10 and 50 people per one Free Wi-Fi router, without any clear correlation with population density. The limited heterogeneity of Wi-Fi presence lets us conclude that also Wi-Fi access alone is insufficient to justify the differences in mobile service consumption across urban and rural areas that we find in the France scenario.

[5] https://www.data.gouv.fr/fr/datasets/monreseaumobile/.



(a) Cluster 2 (b) Cluster 3 (c) Cluster 5 (d) Cluster 9

Figure 8: Significance vectors $\rho_k^{S'}(j)$ for the four clusters $k \in \mathcal{K}$ in Fig. 7.



(a) Cluster 2 (b) Cluster 3 (c) Cluster 5 (d) Cluster 9

Figure 9: Significance vectors $\rho_k^{D}(d)$ for the four clusters $k \in \mathcal{K}$ in Fig. 7. We consider urbanization levels as the levels $d \in \mathcal{D}$.

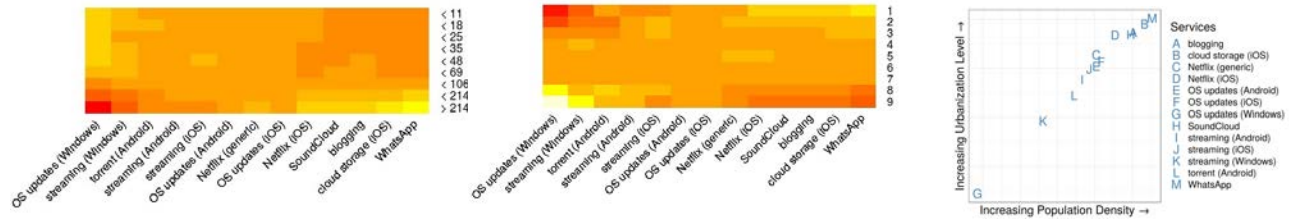


Figure 10: Left, middle: incidence matrices $\mathbf{M}(d, j)$, as heatmaps where light (respectively, dark) colors denote high (respectively, low) values. Demographics levels in $\mathcal{D}$ are derived from discretized population density in terms of number of people per square kilometer (left) and urbanization levels (middle). Services are from the informative set $\mathcal{S}'$. Right: relative positioning of informative mobile services in the space of population density and urbanization levels.

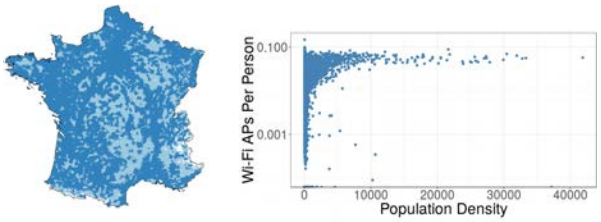


Figure 11: Left: 3G (light) and 4G (dark) coverage in France. Right: Scatterplot of the population density and Wi-Fi access point per person recorded in all communes of France.

Overall, the results in this section lead to our final takeaway message: ***there exist clear interplays between the usage of a specific set of mobile services and the demographics features of the territory, i.e., people tend to consume differently some applications in cities and in the countryside***. Such relationships are time-invariant: tests not detailed here due to space limitations return nearly identical results to those in Fig. 10 when disaggregating the data into night, morning, afternoon, and evening hours.

7 CONCLUSIONS

We unveil that most mobile services are typically consumed in very similar ways across a whole country like France. This suggests that heterogeneity in mobile service usage is observable at citywide scale due to land use [16], and at worldwide scale due to cultural and language differences [27], but it is much weaker at nationwide scale. In the latter case, we show that only a few specific services display spatial diversity, in a way that is strongly linked to urbanization. While our results are for France, the methods used to derive them are general. Also, they are ductile, which leaves space for fine tuning and improvements: for instance, the two-phase algorithm can accommodate any clustering technique as the `cluster` procedure, including more complex solutions than a greedy strategy. Our insights and approach can be useful in mobile networking (for infrastructure planning), sociology (to understand relationships between digital activity and social segregation), or urban planning (to correlate mobile service usage and city structures).

ACKNOWLEDGMENTS

The authors would like to thank Dr. Mathieu Cunche for providing the Wi-Fi deployment data. R. Singh is supported in part by a PhD studentship under the EPSRC Centre for Doctoral Training in Pervasive Parallelism at the University of Edinburgh. R. Singh and M. Marina are also supported in part by The Alan Turing Institute through the PhD Enrichment scheme and Turing Fellowship, respectively. The work of M. Fiore was partially supported by the European Union Horizon 2020 Framework Programme under REA grant agreement no.778305 DAWN4IoE.

REFERENCES

[1] S. Arimoto. 1972. An Algorithm for Computing the Capacity of Arbitrary Discrete Memoryless Channels. *IEEE Transactions on Information Theory* 18, 1 (Jan 1972), 14–20. https://doi.org/10.1109/TIT.1972.1054753

[2] A. Banerjee, S. Merugu, I.S. Dhillon, and J. Ghosh. 2005. Clustering with Bregman Divergences. *J. Mach. Learn. Res.* 6 (Dec 2005), 1705–1749.

[3] R. Blahut. 1972. Computation of Channel Capacity and Rate-distortion Functions. *IEEE Transactions on Information Theory* 18, 4 (Jul 1972), 460–473. https://doi.org/10.1109/TIT.1972.1054855

[4] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, and Lefebvre E. 2008. Fast Unfolding of Communities in Large Networks. *Journal of Statistical Mechanics: Theory and Experiment* 10 (Oct 2008), P10008.

[5] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang. 2006. Complex Networks: Structure and Dynamics. *Physics Reports* 424, 4 (2006), 175 – 308. https://doi.org/10.1016/j.physrep.2005.10.009

[6] M. Böhmer, B. Hecht J. Schöning, A. Krüger, and G. Bauer. 2011. Falling Asleep with Angry Birds, Facebook and Kindle: A large scale study on Mobile Application Usage. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '11)*. ACM, New York, NY, USA, 47–56. https://doi.org/10.1145/2037373.2037383

[7] P. S. Chodrow. 2017. Structure and Information in Spatial Segregation. *Proceedings of the National Academy of Sciences* 114, 44 (2017), 11591–11596. https://doi.org/10.1073/pnas.1708201114

[8] B. Cici, M. Gjoka, A. Markopoulou, and C.T. Butts. 2015. On the Decomposition of Cell Phone Activity Patterns and Their Connection with Urban Ecology. In *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '15)*. ACM, New York, NY, USA, 317–326. https://doi.org/10.1145/2746285.2746292

[9] Cisco. 2017. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021.

[10] M. De Nadai, J. Staiano, R. Larcher, N. Sebe, D. Quercia, and B. Lepri. 2016. The Death and Life of Great Italian Cities: A Mobile Phone Data Perspective. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 413–423. https://doi.org/10.1145/2872427.2883084

[11] I.S. Dhillon, S. Mallela, and R. Kumar. 2003. A Divisive Information Theoretic Feature Clustering Algorithm for Text Classification. *J. Mach. Learn. Res.* 3 (March 2003), 1265–1287.

[12] J. Erman, A. Gerber, K.K. Ramadrishnan, S. Sen, and O. Spatscheck. 2011. Over the Top Video: The Gorilla in Cellular Networks. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference (IMC '11)*. ACM, New York, NY, USA, 127–136. https://doi.org/10.1145/2068816.2068829

[13] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin. 2010. Diversity in Smartphone Usage. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*. ACM, New York, NY, USA, 179–194. https://doi.org/10.1145/1814433.1814453

[14] D. Ferreira, J. Goncalves, V. Kostakos, L. Barkhuus, and A.K. Dey. 2014. Contextual Experience Sampling of Mobile Application Micro-Usage. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services (MobileHCI '14)*. ACM, New York, NY, USA, 91–100. https://doi.org/10.1145/2628363.2628367

[15] P. Fiadino, P. Casas, M. Schiavone, and A. D'Alconzo. 2015. Online Social Networks Anatomy: On the Analysis of Facebook and WhatsApp in Cellular Networks. In *2015 IFIP Networking Conference (IFIP Networking)*. Toulouse, France, 1–9. https://doi.org/10.1109/IFIPNetworking.2015.7145326

[16] A. Furno, M. Fiore, R. Stanica, C. Ziemlicki, and Z. Smoreda. 2017. A Tale of Ten Cities: Characterizing Signatures of Mobile Traffic in Urban Areas. *IEEE Transactions on Mobile Computing* 16, 10 (Oct 2017), 2682–2696. https://doi.org/10.1109/TMC.2016.2637901

[17] Insee Code Officiel Geographique. 2017. https://www.insee.fr/fr/accueil

[18] S. Grauwin, S. Sobolevsky, S. Moritz, I. Gódor, and Ratti C. 2015. Towards a Comparative Science of Cities: Using Mobile Traffic Records in New York, London, and Hong Kong. *Geotechnologies and the Environment* 13 (2015), 363–387.

[19] D. Hintze, P. Hintze, R.D. Findling, and R. Mayrhofer. 2017. A Large-Scale, Long-Term Analysis of Mobile Device Usage Characteristics. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 2, Article 13 (June 2017), 21 pages. https://doi.org/10.1145/3090078

[20] R. Keralapura, A. Nucci, Z.L. Zhang, and L. Gao. 2010. Profiling Users in a 3G Network Using Hourglass Co-clustering. In *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking (MobiCom '10)*. ACM, New York, NY, USA, 341–352. https://doi.org/10.1145/1859995.1860034

[21] H. Li, X. Lu, X. Liu, T. Xie, K. Bian, F.X. Lin, Q. Mei, and F. Feng. 2015. Characterizing Smartphone Usage Patterns from Millions of Android Users. In *Proceedings of the 2015 Internet Measurement Conference (IMC '15)*. ACM, New York, NY, USA, 459–472. https://doi.org/10.1145/2815675.2815686

[22] Z. Li, X. Wang, N. Huang, M.A. Kaafar, Z. Li, J. Zhou, G. Xie, and P. Steenkiste. 2016. An Empirical Analysis of a Large-scale Mobile Cloud Storage Service. In *Proceedings of the 2016 Internet Measurement Conference (IMC '16)*. ACM, New York, NY, USA, 287–301. https://doi.org/10.1145/2987443.2987465

[23] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez. 2018. How Should I Slice My Network?: A Multi-Service Empirical Evaluation of Resource Sharing Efficiency. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom '18)*. ACM, New York, NY, USA, 191–206. https://doi.org/10.1145/3241539.3241567

[24] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, C. Ziemlicki, and Z. Smoreda. 2017. Not All Apps Are Created Equal: Analysis of Spatiotemporal Heterogeneity in Nationwide Mobile Service Usage. In *Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies (CoNEXT '17)*. ACM, New York, NY, USA, 180–186. https://doi.org/10.1145/3143361.3143369

[25] G.W. Milligan and M.C. Cooper. 1985. An Examination of Procedures for Determining the Number of Clusters in a data set. *Psychometrika* 50, 2 (1985), 159–179.

[26] U. Paul, A. P. Subramanian, M. M. Buddhikot, and S. R. Das. 2011. Understanding Traffic Dynamics in Cellular Data Networks. In *2011 Proceedings IEEE INFOCOM*. Shanghai, China, 882–890. https://doi.org/10.1109/INFCOM.2011.5935313

[27] E. Peltonen, E. Lagerspetz, J. Hamberg, A. Mehrotra, M. Musolesi, P. Nurmi, and S. Tarkoma. 2018. The Hidden Image of Mobile Apps: Geographic, Demographic, and Cultural Factors in Mobile Usage. In *Proc. of Int. Conf. on Human-Computer Interaction with Mobile Devices and Services, MobileHCI*. Barcelona, Spain.

[28] P. Rouveyrol, P. Raveneau, and M. Cunche. 2015. Large Scale Wi-Fi tracking using a Botnet of Wireless Routers. In *Workshop on Surveillance & Technology*. Philadelphie, United States. https://hal.inria.fr/hal-01151446

[29] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang. 2012. Characterizing Geospatial Dynamics of Application Usage in a 3G Cellular Data Network. In *2012 Proceedings IEEE INFOCOM*. Orlando, FL, USA, 1341–1349. https://doi.org/10.1109/INFCOM.2012.6195497

[30] C. Smith, A. Mashhadi, and L. Capra. 2013. Ubiquitous Sensing for Mapping Poverty in Developing Countries. In *Proceedings of the 3rd International Conference on the Analysis of Mobile Phone Datasets (NetMob)*.

[31] I. Trestian, S. Ranjan, A. Kuzmanovic, and A. Nucci. 2009. Measuring Serendipity: Connecting People, Locations and Interests in a Mobile 3G Network. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement (IMC '09)*. ACM, New York, NY, USA, 267–279. https://doi.org/10.1145/1644893.1644926

[32] F. Xu, Y. Li, H. Wang, P. Zhang, and D. Jin. 2017. Understanding Mobile Traffic Patterns of Large Scale Cellular Towers in Urban Environment. *IEEE/ACM Transactions on Networking* 25, 2 (April 2017), 1147–1161. https://doi.org/10.1109/TNET.2016.2623950

[33] Q. Xu, J. Erman, A. Gerber, Z. Mao, J. Pang, and S. Venkataraman. 2011. Identifying Diverse Usage Behaviors of Smartphone Apps. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference (IMC '11)*. ACM, New York, NY, USA, 329–344. https://doi.org/10.1145/2068816.2068847

[34] D. Zhang, F. Zhang, J. Huang, C. Xu, Y. Li, and He T. 2014. Exploring Human Mobility with Multi-Source Data at Extremely Large Metropolitan Scales. In *ACM MobiCom*. Maui, HI, 201–212.

[35] Y. Zhang and A. Arvidsson. 2012. Understanding the Characteristics of Cellular Data Traffic. In *Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design (CellNet '12)*. ACM, New York, NY, USA, 13–18. https://doi.org/10.1145/2342468.2342472

Anything to Hide? Studying Minified and Obfuscated Code in the Web

Philippe Skolka
Department of Computer Science
TU Darmstadt

Cristian-Alexandru Staicu
Department of Computer Science
TU Darmstadt

Michael Pradel
Department of Computer Science
TU Darmstadt

ABSTRACT

JavaScript has been used for various attacks on client-side web applications. To hinder both manual and automated analysis from detecting malicious scripts, code minification and code obfuscation may hide the behavior of a script. Unfortunately, little is currently known about how real-world websites use such code transformations. This paper presents an empirical study of obfuscation and minification in 967,149 scripts (424,023 unique) from the top 100,000 websites. The core of our study is a highly accurate (95%-100%) neural network-based classifier that we train to identify whether obfuscation or minification have been applied and if yes, using what tools. We find that code transformations are very widespread, affecting 38% of all scripts. Most of the transformed code has been minified, whereas advanced obfuscation techniques, such as encoding parts of the code or fetching all strings from a global array, affect less than 1% of all scripts (2,842 unique scripts in total). Studying which code gets obfuscated, we find that obfuscation is particularly common in certain website categories, e.g., adult content. Further analysis of the obfuscated code shows that most of it is similar to the output produced by a single obfuscation tool and that some obfuscated scripts trigger suspicious behavior, such as likely fingerprinting and timing attacks. Finally, we show that obfuscation comes at a cost, because it slows down execution and risks to produce code that changes the intended behavior. Overall, our study shows that the security community must consider minified and obfuscated JavaScript code, and it provides insights into what kinds of transformations to focus on. Our learned classifiers provide an automated and accurate way to identify obfuscated code, and we release a set of real-world obfuscated scripts for future research.

CCS CONCEPTS

• **Security and privacy → Web application security**; *Software reverse engineering*; *Intrusion/anomaly detection and malware mitigation.*

KEYWORDS

obfuscation; web security; empirical study; machine learning

ACM Reference Format:
Philippe Skolka, Cristian-Alexandru Staicu, and Michael Pradel. 2019. Anything to Hide? Studying Minified and Obfuscated Code in the Web . In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313752

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313752

1 INTRODUCTION

JavaScript has become the dominant programming language for client-side web applications and nowadays is used in the vast majority of all websites. The popularity of the language makes JavaScript an attractive target for various kinds of attacks. For example, cross-site scripting attacks try to inject malicious JavaScript code into websites [14, 17, 28]. Other attacks aim at compromising the underlying browser [6, 7] or extensions installed in a browser [11, 12], or they abuse particular web APIs [19, 26, 30].

An effective way to hide the maliciousness of JavaScript code are code transformations that preserve the overall behavior of a script while making it harder to understand and analyze. Such transformations affect both manual code inspection, e.g., because the code becomes harder to understand, and automated code analysis, e.g., because the malicious behavior is disguised as apparently harmless operations. There exist a variety of code transformations, ranging from renaming of local variables to more complex code changes that affect the control flow and data flow. We refer to transformations aimed at reducing code size, typically by renaming local variables to shorter names, as *minification*. In contrast, we refer to more complex transformations aimed at hindering the understanding and analysis of code as *obfuscation*. It is important to note that minification and obfuscation may be used for legitimate reasons, such as reducing code size or protecting intellectual property. However, independently of what the reason for transforming code is, it affects the ability of human and automated security analysis.

Despite the potential impact that minification and obfuscation may have on security analysis, little is currently known about how real-world websites use such transformations. A better understanding of what kinds of code transformations are applied in the wild could guide future efforts on making the web more secure. In particular, knowing how widespread minification and obfuscation are helps future analyses to focus on relevant problems. Moreover, understanding what kinds of transformation tools are the most popular enables the development of targeted defense techniques. Unfortunately, to the best of our knowledge, there currently is no comprehensive study of code transformations in the web.

This paper presents a large-scale empirical study of minification and obfuscation in client-side web code. The study involves 967,149 JavaScript files gathered from the top 100,000 websites. We analyze how many of these scripts are transformed through minification and obfuscation, respectively, and which tools are used for these transformations. Moreover, we study which kinds of scripts are transformed particularly often and inspect the runtime behavior

of some obfuscated scripts. Finally, we analyze which costs code transformation may incur by assessing to what extent popular transformation tools influence the performance and correctness of JavaScript code.

Given the large-scale nature of our study, we rely, at least in parts, on automation to answer the above questions. To this end, we present a neural network-based classifier that identifies JavaScript code with particular properties. For example, the classifier can be trained to distinguish transformed from non-transformed code, minified from obfuscated code, and to identify code transformed with particular tools. We show that the classification has very high accuracy for these tasks, providing an effective way to identify particular kinds of scripts across all studied websites.

The study addresses six research questions.

RQ1: How prevalent are minification and obfuscation in client-side JavaScript code? Answering this question is important to determine whether code transformations should be considered by security analyses at all, and what kinds of transformations such analyses should focus on. We find that code transformations are very widespread, affecting 38% of all client-side scripts. The vast majority of the transformed code has been minified, whereas less than 1% of all code is obfuscated. Even though the percentage of obfuscated code is low, the absolute number of obfuscated scripts (2,842) still motivates work on de-obfuscation, as these scripts arguably are the most interesting for security analysis.

RQ2: Which tools are used to obfuscate code in the web? There is a variety of tools available for obfuscating JavaScript code. Understanding which tools and transformation techniques are used most often in practice helps prioritize efforts toward dealing with transformed code. Our study finds that a single obfuscation tool accounts for most obfuscated scripts in the web: 2,551 obfuscated scripts resemble the output of the *Daft Logic Obfuscator*, an online tool that is available free of charge, motivating future work to consider obfuscation techniques implemented by this tool.

RQ3: Does the prevalence of code transformations differ across different kinds of scripts or websites? Since there are many possible reasons for minifying and obfuscating code, it is interesting to ask whether specific kinds of scripts are transformed more often than others. We find that third-party scripts, i.e., code loaded from another website than the one visited by a user, is about twice as likely to be transformed than scripts loaded from the visited website. Possible reasons for this distribution include that content delivery networks minify libraries to reduce network traffic, and that advertisement and tracking code is transformed to protected intellectual property. Studying the prevalence of obfuscation in different categories of websites shows that some categories, e.g., sites with adult content, contain more obfuscated scripts than an average website.

RQ4: What kind of behavior do developers hide behind obfuscation? To further understand the reasons for obfuscating code, we analyze the execution behavior of obfuscated scripts and manually inspect a subset of them. We find that many of the obfuscated scripts access APIs that are typically used for tracking, fingerprinting, cookie syncing, or cookie theft. We also identify a script with an unusually high number of calls to `performance.now`, which could be because the script is exploiting some timing channel.

The final two questions are about potential costs that applying code transformations may incur.

RQ5: How do code transformations affect the performance of code? We find that most obfuscation tools negatively affect performance, i.e., they slow down the execution of the code. In contrast, most minification tools either have no effect on performance or speed up the execution of the code. These findings show that complex code transformations may come at a non-negligible cost, motivating future work on performance-invariant obfuscation.

RQ6: How do code transformations affect the correctness of code? Developers applying an automated code transformation tool may naively assume that the transformation preserves the overall semantics of the code. However, we find that existing tools for both minification and obfuscation often produce corrupt code that behaves differently from the original code. Only about half of the transformed code completely preserves the original semantics, motivating future work on more reliable transformation tools.

The results of our work are relevant for at least four groups of people. First, the study provides insights for developers of security-related program analyses, e.g., static malware checkers and de-obfuscators. In particular, we show that transformed code must be considered by any analysis aimed at real-world, client-side JavaScript code, and we show what kinds of transformations are the most important in practice. Second, the study affects developers of obfuscation and minification tools by highlighting the costs that using state-of-the-art tools imply. Third, the study informs users of transformation tools about the effects that using such tools may have on the performance and correctness of code. Finally, the classifier that we develop to conduct our study enables researchers interested in analyzing real-world JavaScript code to focus on code relevant for their research. For example, the classifier can accurately identify obfuscated code among hundreds of thousands of scripts.

In summary, this paper contributes the following:

- The first large-scale study of minification and obfuscation in real-world, client-side web application code.
- Insights about how code transformations are used in practice, including evidence that minification is widespread, that more complex obfuscation is rather rare yet non-negligible, and that particular obfuscation techniques clearly dominate.
- An automated classification technique that accurately identifies different kinds of transformed code. The technique is useful to select particular scripts, e.g., those with obfuscated code, for further analysis.
- A benchmark of obfuscated JavaScript files gathered from various popular websites, which we provide for future work on de-obfuscating, analyzing, and understanding obfuscated code.

2 CLASSIFICATION OF SCRIPTS

Addressing RQ1, RQ2, RQ3, and RQ4 at the scale of hundreds of thousands of scripts requires an automated technique to determine whether a script has been transformed, and if yes, in what way. While a skilled human could manually label files with high accuracy, that approach does not scale to the amount of JavaScript code considered in our study. One approach to address this challenge would be to define a set of heuristics, e.g., based on idiosyncrasies

of specific transformation tools, and to determine the properties of code based on these heuristics. Unfortunately, the heuristics-based approach relies on human expertise for identifying code properties that are unique to specific transformation techniques, and it cannot be easily adapted to other transformation tools. Instead, we address the challenge of determining properties of code through machine learning. This section presents two machine learning-based approaches for classifying JavaScript code. We use the more effective of the two approaches as the basis of our study.

2.1 Classification Tasks

We address three classification tasks.

TRANS. The first task, called *TRANS*, is to determine whether a given piece of JavaScript code has been transformed by any minification or obfuscation tool. We train classifiers for this task with examples of regular code, which has not been processed by any transformation tool, and with examples of transformed code, which has been processed by a minification or obfuscation tool.

OBFUS. The second task, called *OBFUS*, is to determine whether a given piece of JavaScript code has been obfuscated. We train OBFUS classifiers with examples of regular code, examples of minified but not obfuscated code, and examples of obfuscated code.

TOOL-X. The third task, called *TOOL-X*, is to determine for a given piece of transformed JavaScript code what tool has been used to transform the code. Because popular minification tools apply very similar transformations, we focus on obfuscation tools for this task.

2.2 Training Data for Learning Classifiers

To train the classifiers, we start with a set of human-written, or *regular*, JavaScript files and then create transformed variants of these files.

2.2.1 Corpus of Regular Code. The regular code examples are a subset of a corpus of 150,000 JavaScript files provided by others [24], which consists of human-written, non-transformed code from open-source projects. We remove from this corpus all files with a size less than 1kB, as they provide very little information for the classifiers to make an informed decision, and files with a size greater than 10kB, to keep the memory consumption during training at a manageable level.

2.2.2 Program Transformation Tools. Our study is based on popular minification and obfuscation tools. We select tools that are publicly available and widely used, as reported, e.g., by publicly visible download numbers.

Minification. We consider seven minification tools (the names in parentheses are the abbreviation we use throughout the paper): UglifyJS (uglify), babel-minify (babel), Google Closure Compiler (closure), javascript-minifier.com (jsmincom), Matthias Mullie Minify (mmminify), and YUI Compressor (yui). These tools reduce the program size mainly by performing white spaces reduction and identifiers shortening. In addition, some tools, e.g., closure, also perform optimizations, such as inlining or constant folding.

Table 1: Obfuscation tools and transformation techniques.

Transformation techniques	Obfuscation tools				
	jsobf	jsobfcom	jfogs	JSObfu	daft-logic
String splitting	✓				✓
Keyword substitution					
String concatenation				✓	
Encoding the entire code					✓
Encrypting the entire code					
Identifier encoding	✓	✓	✓	✓	✓
String encoding	✓	✓		✓	
Dead code injection	✓				
Control flow flattening	✓				
String array	✓	✓	✓		✓
Code protecting techniques	✓				

Obfuscation. We consider five obfuscation tools: javascript-obfuscator (jsobf), javascriptobfuscator.com (jsobfcom), DaftLogic Obfuscator (daft-logic), jfogs, and JSObfu. The result of our tool search also included JSFuck and javascript2img.com, but we exclude them as they either are unable to process large code files or produce invalid JavaScript code. Table 1 shows which transformation techniques the tools use, as reported in previous work and in the tool's documentation. The two most common obfuscation techniques are identifier encoding, usually using HEX encoding, and storing strings in a global array.

Configurations. Most tools provide options to configure which transformation techniques to apply. Since different configurations may result in a different transformed code, we use multiple configurations for each tool. In total, we consider 15 configurations for the obfuscation tools and 31 configurations for the minification tools. Together with the regular version of a JavaScript file, this setup yields 47 variants of each file.

2.2.3 Generation of Training Data. As training data for a specific classifier, we randomly sample 10,000 files from the corpus of regular files and apply transformations tools to these files. For all three tasks, we train the classifiers with an even split of two classes of code, i.e., half of the training examples are expected to be classified as positive and negative, respectively. For the TRANS classifier, we apply the minification and obfuscation tools to each code example, using each tool equally often. For the OBFUS classifier, we obtain examples of obfuscated code by applying one of the obfuscation tools to each code example, using each tool equally often. To obtain examples of non-obfuscated code, we use regular and minified code. Since OBFUS gets trained to distinguish obfuscated code from both minified code and regular code, it can be used not only to identify obfuscated code among any code, but also to classify transformed code into minified versus obfuscated code. For the TOOL-X classifiers, we use examples of code transformed by tool X and code examples that are either not transformed or transformed by other tools.

2.3 Classification via Identifier Frequencies

The following describes the first of two approaches to learn classifiers. The approach exploits the fact that both minification and obfuscation tools replace the original identifiers, e.g., names of local variables, with other identifiers. Because many transformation tools use specific identifiers, a skilled human can determine whether a tool has been used and if yes, which tool. The learning approach is based on this observation.

The approach consists of two main steps. The first step is to extract a feature vector for a given JavaScript file. The feature vector summarizes what identifier names occur in the file and how frequent each name is. To this end, we tokenize the code and extract all identifiers. Based on all identifiers that occur in the training data, we determine a vocabulary of the 30,000 most common identifiers. Then, we transform each script into a feature vector of length 30,000, where each element represents a specific identifier. The element that represents a specific identifier is the *tf-idf* value of the identifier, i.e., the result of multiplying the *term frequency* with the *inverted document frequency* of an identifier. To compute the term frequency for a given script, we count the occurrences of each identifier and normalize these with the number of occurrences of the most frequent identifier in the script. For the inverted document frequency we divide, for each identifier, the total number of scripts in the training dataset by the number of scripts that contain the identifier and compute the logarithm of the resulting value.

The second step is to classify the feature vector of a JavaScript file using a support vector machine (SVM). We use SVMs because they are effective for binary classification problems. We achieve the best results for the SVM when using the *radial basis function kernel* and setting the penalty term C to 5. The classifier is implemented in Python using the machine learning library scikit-learn[1].

2.4 Classification via AST Convolution

The identifier-based classifier described above is conceptually simple but limited to a single feature of source code, i.e., identifier names. Our second classification approach addresses this limitation through a neural network that classifies abstract syntax trees (ASTs). ASTs are a useful representation of code because they preserve all relevant information while making the structural relationships between code elements explicit. To classify ASTs, we build upon a neural network architecture proposed by Mou et al. [18]. We adapt their approach by enriching traditional ASTs with additional information that proves useful for our classification tasks.

2.4.1 Background: Tree-based Convolutional Neural Network. We build upon an existing machine learning architecture for classifying trees, e.g., ASTs, based on a convolutional neural network. The network transforms a given tree into a continuous vector representation and then performs the actual classification task on the vector representation. The vector representation is learned in such a way that similar trees are represented by similar vectors. Figure 1 shows the five main steps involved in classifying trees. First, each node is transformed into a vector, where nodes with the same label, e.g., two CallExpression nodes, are mapped to the same vector. Second, a neural network layer ("coding layer") summarizes the children of a node and the node itself into the parent's vector. Third, several convolution layers extract features from the tree by sliding a feature detector over fixed-depth subtrees of the tree, which yields several trees of features. Fourth, a pooling layer summarizes these trees of features into a single tree again. Finally, all nodes of the new tree are passed through a fully connected hidden layer that outputs the classification result. During training with stochastic gradient descent, the parameters of the network are adapted to minimize a loss function that expresses how much the network's classification differs from the expected classification. Our implementation of the tree-based convolutional neural network is based on an implementation by Creston Bunch[2]. The architecture has previously been shown to be effective for identifying code that implements a particular algorithm [18]. We are the first to use it for identifying minified and obfuscated code.

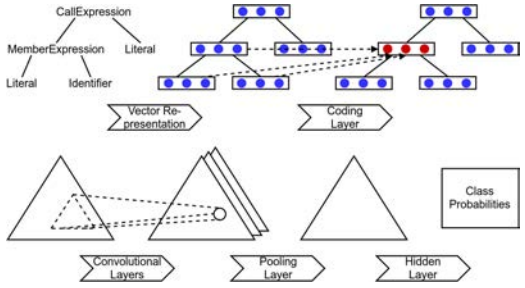


Figure 1: Tree-based convolutional neural network [18]. Rectangles with dots are vector representations of AST nodes; triangles represent trees.

2.4.2 Enriched ASTs. One possible approach is to apply the neural network to standard JavaScript ASTs. During initial experiments, we find this approach to provide a classifier with promising yet not fully satisfying accuracy. In particular, the classifiers struggle to learn from two features that are useful for a human but not well represented in standard ASTs: whitespace and specific properties of identifier names. Motivated by this observation, we enrich the standard JavaScript ASTs in two ways.

Whitespace. The first enrichment is to add information about whitespace into ASTs. Usually, this information is abstracted away as it is irrelevant for most scenarios where ASTs are used. To recognize minified and obfuscated code, though, whitespace is relevant because both minification and obfuscation tools often remove all or at least some whitespace. We enrich ASTs with whitespace information as follows: Whenever two nodes in the AST correspond to successive elements in the source code, we check whether any whitespace exists between the two code elements. If no such whitespace exists, then we insert a new "no whitespace" node between the two adjacent nodes; otherwise, we leave the nodes unchanged.

Length of identifiers. The second enrichment of ASTs is to add information about the length of identifiers, i.e., the number of characters of an identifier name. The motivation is that several obfuscation tools replace identifiers with less understandable identifiers of a fixed length that differs from tool to tool. In contrast, regular code

[1] http://scikit-learn.org/stable

[2] https://github.com/crestonbunch/tbcnn

Table 2: Testing accuracies for different classifiers and classification tasks.

Classifier	Task		
	TRANS	OBFUS	TOOL-X
Identifier frequencies	76.40%	80.07%	64.44%–82.86%
AST convolution	95.06%	99.96%	99.68–100.00%

usually consists of natural identifiers that have variable lengths. We encode this information by modifying every "Identifier" AST node by appending the length of the identifier to the node label, e.g., "Identifier3" for an identifier `foo`. To deal with unusually long identifiers, any length exceeding 30 characters is simply represented by a single, special label.

2.5 Accuracy of Classifiers

To decide which of the two classifiers to use for the study, we measure their classification accuracy on previously unseen sets of validation data. To this end, we randomly select 2,500 files from the non-transformed code (disjoint from the files used for generating training data), and create transformed variants of them, as described in Section 2.2.3. We then measure the accuracy for each classifier and task, i.e., the percentage of predictions that match the expected classification.

Table 2 summarizes the accuracy results for the two classifiers presented in Sections 2.3 and 2.4. Overall, the results show that both classifiers are effective (for comparison, a random decision would achieve 50% accuracy) and that the AST convolution-based classifier has the by far highest accuracy for all three tasks. In particular, the AST convolution-based classifier achieves more than 95% accuracy for all three tasks, and at least 99.68% accuracy for the OBFUS and TOOL-X tasks. In contrast, the classifier based on identifier frequencies performs poorly for some of the TOOL-X tasks, with an accuracy as low as 64.44% for one of the obfuscation tools.

To better understand why one classifier performs better than the other, we check how many of the different kinds of training examples the classifiers identify correctly. We find that the identifier frequency-based classifier is successful at identifying minified and obfuscated code but often fails to correctly identify regular code. For example, for the OBFUS task, the classifier correctly labels 99.92% of all obfuscated examples, but classifies only 59.47% of all non-obfuscated examples correctly. A detailed analysis of the identifiers that occur in obfuscated and non-obfuscated code explains these results: Many obfuscation tools use very characteristic identifiers, whereas regular code contains a wide range of natural identifiers. For example, jfogs creates many variables names `$fog$` followed by some number, and jsobf uses a similar pattern but also hex-encodes the identifiers.

To validate that enriching AST is beneficial over running the neural network on default JavaScript ASTs, we compare the accuracies of both variants of the AST-based classifier. We find that the default ASTs yield significantly lower accuracies than the enriched ASTs. For example, for the OBFUS task, the default ASTs give only 75.43% accuracy, which is not only much lower than with enriched ASTs but also lower than the classifier based on identifier frequencies.

Overall, we conclude from the accuracy results that the AST convolution-based classifier is highly effective at identifying transformed code, obfuscated code, and code obfuscated with a particular tool, making it a solid basis for studying JavaScript code at a large scale.

3 STUDYING DEPLOYED CLIENT-SIDE CODE

Based on the classifiers described above, this section presents the setup and results of our study of minification and obfuscation in deployed, client-side JavaScript code.

3.1 Study Data: Deployed, Client-Side JavaScript Code

To gather a representative set of JavaScript code used in real-world websites, we crawl the top 100,000 most popular websites, as listed by the Majestic Million[3] service. Our crawler visits each website, waits five seconds to enable dynamically loaded code to arrive, and then saves all scripts. We consider both code loaded via .js files and code loaded via inline scripts, i.e., via `<script>` tags without a `src` attribute. For the latter, the crawler copies the code between the tags into a new file. To speed up the loading of websites, we do not fetch resources other than scripts and HTML code.

The crawling yields 2,335,207 scripts from 85,001 websites in total. 14,999 websites are not accessible due to timeout errors and other reasons. Due to the limited size of ASTs that the classifiers can handle in reasonable time, we remove scripts with a size exceeding 40kB. This results in 1,861,489 scripts. We further remove scripts that are smaller than 512 bytes because we observe that such small scripts are hard to classify even for human subjects due to the limited number of clues that can aid the distinction between transformed and original scripts. Finally, we remove all those scripts for which the AST does not contain at least one "CallExpression" node. This is because such scripts are very often just configuration files in the JSON format. Applying these filters yields a set of 967,149 scripts. However, different websites may use the same JavaScript code, e.g., third-party libraries, thus the set of scripts contains duplicate files. In this study we are mostly concerned with the nature of the code on the web and not so much with its frequency on different websites, hence we remove all duplicates. This filtering leaves **a final set of 424,023 unique JavaScript files**, which we use for the study.

3.2 Accuracy of Classifiers on Study Data

Section 2.2.2 has established that our classifiers are highly accurate on code transformed by the tools used to generate the training data. To validate that the classifiers are effective also on the study data, for which we do not know which (if any) tools have been used, and to validate that the classification results match the classification that a skilled human could produce, we perform an experiment with JavaScript developers. The experiment involves five advanced developers, who have extensive experience in writing JavaScript and in understanding real-world JavaScript code.

[3] https://de.majestic.com/reports/majestic-million

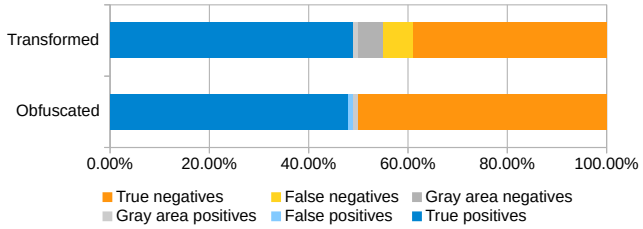


Figure 2: Effectiveness of the TRANS and OBFUS classifiers judged against the ground truth obtained from experienced JavaScript developers. The gray area depicts scripts for which the developers did not agree on the classification label.

The goal of the experiment is to gather ground truth to compare our classifiers against. During the experiment, each developer performs two tasks, which validate the TRANS and OBFUS classifiers, respectively. First, to validate the TRANS classifier, we show to each developer 50 scripts labeled by the classifier as transformed and 50 scripts labeled by the classifier as not transformed. We randomly sample the scripts from all 424,023 scripts in the study data. For each script, the developers are asked to answer the following question: "Is this human-written or generated/transformed code?" Second, to validate the OBFUS classifier, we repeat the same setup with 50 scripts labeled as obfuscated by the classifier and 50 scripts labeled as not obfuscated by the classifier. For these scripts, the developers are asked to answer the question: "Is this obfuscated or not obfuscated code? If the file is transformed but you are not sure if it is obfuscated, try to decide whether the developer transformed it for hindering understanding." To avoid any influence from file names, such as "jquery.min.js", we hide the original file names from the developers and assign some generic names to the files.

After aggregating the results of the experiment, we compute the inter-rater agreement, which quantifies the extent to which the participants agree with each other. More precisely, we compute Cohen's kappa for each pair of participants. The pairwise agreement ranges between 0.64 and 0.93, with an average of 0.81, which is considered a very high agreement. To account for possible mistakes made by individual developers, we consider a 4-out-of-5 majority vote. That is, if at least four of the developers agree on whether a given script is transformed/obfuscated, then we consider this decision as the ground truth. If no such majority is reached, then we consider the scripts to be in a *gray area*, where even humans have a hard time judging whether the script has been transformed/obfuscated.

Figure 2 shows the results of the developer experiment. The blue area depicts true positives, i.e., scripts where the classifier and the developers agree on the scripts being transformed/obfuscated. Likewise, the orange area depicts true negatives, i.e., scripts where the classifier and the developers agree on the scripts being not transformed/obfuscated. Overall, the automated classifiers largely match the decisions by the developers. In addition, the TRANS classifier has 13% false negatives, i.e., it misses a few scripts that developers consider to be transformed, and the OBFUS classifier has 2% false positives, i.e., it sometimes incorrectly labels a non-obfuscated script as obfuscated. The gray parts in the middle of the figure, six scripts for TRANS and one for OBFUS, are the gray area, where the developers disagreed with each other.

Table 3: Prevalence of transformed code.

TRANS on all scripts:	Regular	61.5%
	Transformed	38.5%
OBFUS on all scripts:	Regular or minified	99.33%
	Obfuscated	0.67%
OBFUS on transformed scripts:	Minified	98.31%
	Obfuscated	1.69%

Table 4: Tools used to obfuscate scripts.

Classifier	% detected scripts	% other scripts	# detected scripts
TOOL-JSObfu	0.01%	99.99%	3
TOOL-jsobfcom	0.04%	99.96%	149
TOOL-jfogs	0.02%	99.98%	0
TOOL-daft-logic	0.60%	99.40%	2,551
TOOL-jsobf	0.02%	99.98%	71

Overall, we conclude from the experiment with developers that our automated classifiers match human classifications for the overwhelming majority of scripts. Given the 13% false negative rate of TRANS, one should interpret our results about the number of transformed scripts as a slight under-approximation of the actual number.

3.3 RQ1. Prevalence of Transformed Code

To address the question of how prevalent minified and obfuscated code is in client-side JavaScript code, we classify all downloaded scripts using the TRANS and OBFUS classifiers. Table 3 summarizes the results. We find that 38.5% of all scripts have gone through some kind of transformation, including both minification and obfuscation. In contrast, only 0.67% of all scripts (2,842 scripts) have been obfuscated. Applying the OBFUS classifier to those scripts that are classified as transformed confirms the above numbers: The vast majority of transformed scripts are not obfuscated, i.e., they have only been minified. These numbers show that minification is popular in the web. This finding is in line with the fact that many JavaScript libraries and frameworks include a minification step in their deployment pipeline to reduce the file size and hence the transmission time. A possible explanation for the low number of obfuscated scripts is that obfuscation comes at a cost (discussed in detail in Section 3.7 and 3.8), and therefore it is used only when developers want to hide some behavior. Despite the surprisingly low number of obfuscated scripts, these scripts provide an interesting target for further analysis, and we provide them as a benchmark for future work.[4]

> **Finding 1.** Around 38% of the scripts in the web are transformed, most of which are minified but not obfuscated. Overall, we find a total of 2,842 obfuscated scripts.

3.4 RQ2. Prevalence of Obfuscation Tools

Given the non-negligible number of obfuscated scripts, we next address the question which tools and techniques developers use for obfuscation. Table 4 shows for each tool X how many scripts are detected as obfuscated by this tool according to the corresponding TOOL-X classifier. The by far most popular obfuscator in the web

[4] http://software-lab.org/projects/obfuscation_study.html

Table 5: Overlap between different classifiers.

Pair of classifiers	Number of scripts
TOOL-jsobfcom ∩ OBFUS	147 (98% of TOOL-jsobfcom)
TOOL-daft-logic ∩ OBFUS	2,474 (97% of TOOL-daft-logic)
TOOL-jsobf ∩ OBFUS	71 (100% of TOOL-jsobf)

is DaftLogic Obfuscator, with 2,551 scripts in total. As the only tool that encodes the entire code using the `eval` function, it clearly stands out among the other tools. The study data does not contain any scripts obfuscated with jfogs. A surprising fact from these results is that the apparent popularity on npm of tools like javascript-obfuscator and JSObfu does not transfer to client-side obfuscated code. One reason may be that these obfuscators are more popular for JavaScript code running on Node.js than for client-side code.

> **Finding 2.** DaftLogic Obfuscator is the by far most popular obfuscation tool. The most popular obfuscation technique is to load code at runtime via `eval`.

Having a set of classifiers related to obfuscation (OBFUS and several TOOL-X classifiers) raises the question to what extent the scripts detected by these classifiers overlap. Table 5 shows the overlap of scripts identified by the different obfuscation-related classifiers. We can make three observations. First, the TOOL-X classifiers do not overlap with each other, i.e., each of them precisely identifies scripts originating from a specific tool. This result is particularly remarkable for javascriptobfuscator.com and javascript-obfuscator, as these tools share a list of common obfuscation techniques. Second, almost all scripts detected as obfuscated by a specific tool are also detected as obfuscated by the general OBFUS classifier. Third, some scripts are classified as obfuscated but none of our TOOL-X classifiers can identify the tool used for obfuscating them.

> **Finding 3.** Our generic obfuscation detection classifier successfully identifies most of the obfuscated scripts found by the individual tool classifiers and several additional scripts.

3.5 RQ3. Transformations vs. Kinds of Scripts

3.5.1 Categories of Scripts. We associate scripts with website categories, such as "News", "Travel", and "Education". To this end, we use the *Juniper Test-a-Site*[5] service, which yields a category for a given URL. We associate a category with a script based on the top-level domain from which the script was loaded.

Obfuscated code. We first analyze the prevalence of obfuscated code loaded by websites in specific categories. Figure 3 shows for ten of the categories the percentage of obfuscated scripts among all scripts loaded by a site and an additional entry with the average across all other categories. The results show that the prevalence of obfuscation differs significantly across website categories. Categories with a particularly high percentage of obfuscated scripts include "Adult/Sexually Explicit" and "Glamour & Intimate Apparel", i.e., sites with content that careful users may trust less than an average website.

Transformed code. Figure 4 shows the ratio between transformed and regular code within all script categories in the study data.

[5]http://mtas.surfcontrol.com/mtas/JuniperTest-a-Site.asp

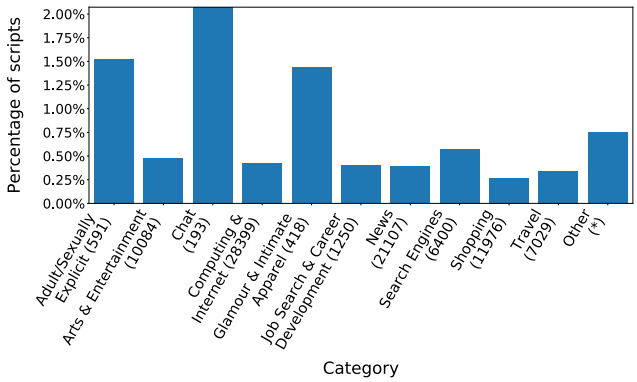


Figure 3: Prevalence of obfuscated code within categories of websites. In parenthesis on the x-axis we show the number of scripts in each category.

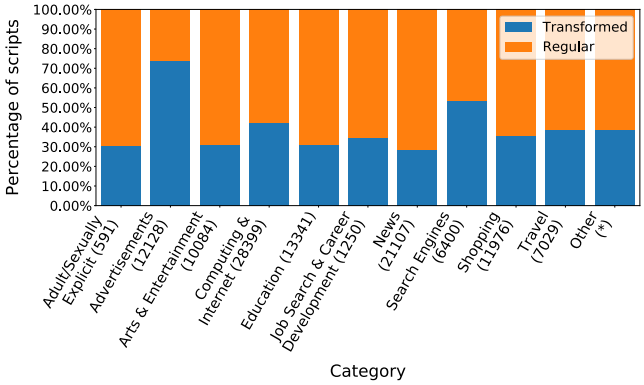


Figure 4: Prevalence of transformed code within categories of websites. In parenthesis on the x-axis we show the number of scripts in each category.

The percentage of transformed code ranges between 28% and 73%. The most significant categories are "Search engines" with 71.7% transformed code and "Advertisements" with 73.7%. We hypothesize that this is the case because websites in these categories serve scripts to many other domains, e.g. tracking scripts, and the size of the delivered scripts has a direct impact in the cost of using these services.

> **Finding 4.** Code obfuscation is particularly common in certain website categories, e.g., adult content. Other code transformations occur particularly often in scripts delivered to many other websites.

3.5.2 Third-party Scripts. The following studies whether transformations are particularly common for scripts loaded from third-party sites. As first-party scripts, we consider all inline scripts and all scripts loaded from the top-level domain of the visited website itself.

All remaining scripts are considered third-party scripts. [6] Based on this grouping of scripts, we analyze how the results of the TRANS and OBFUS classifiers relate to where a script is loaded from.

We find that third-party scripts, with a percentage of 55.38%, are almost twice as frequently transformed than scripts loaded directly from the visited website, which have a percentage of only 30.18% transformed code. These findings seem natural because providers of third-party scripts, e.g., content-delivery networks, often provide a minified version to reduce loading time. For scripts labeled as obfuscated, we do not find a significant difference between third-party and first-party scripts.

> **Finding 5.** Third-party scripts are almost twice as frequently transformed than first-party scripts (55.38% versus 30.18%). Obfuscation is equally uncommon within both categories.

3.6 RQ4. Runtime Behavior of Obfuscated Code

To better understand what the obfuscated scripts actually do, we run them in a custom environment and observe what APIs they try to access. The custom environment consists of self-replicating proxy objects that emulate the browser APIs and other well-known frameworks required by the analyzed scripts. For example, we create a globally accessible `document` proxy object that returns another proxy each time one of its properties is accessed. This environment provides a simple and effective technique for inspecting the behavior of obfuscated scripts.

In total, we analyze 2,924 unique obfuscated scripts which include all scripts classified as obfuscated by either OBFUS or one of the TOOL-X classifiers. We use 13 self-replicating proxies to mock the browser API and we run our analysis in Node.js. After each run, we collect a trace summarizing the property writes, property reads, and function calls observed via the proxies. In total, we collect 3.6 million property accesses, 10,400 writes and 1.4 million function calls. 2,231 scripts access at least one property via the proxies and 1,263 scripts set globally accessible properties. These numbers show that our setup is effective at running the obfuscated scripts and at extracting meaningful information about the APIs they call and the properties they access.

Out of the 2,924 scripts, 341 access the `cookie` object, 316 the `userAgent`, 287 the `location` and 101 the `referrer`. These are all privacy-sensitive APIs that may be accessed to perform cookie theft, cookie syncing, referrer sniffing, or browser fingerprinting. However, a more detailed analysis is needed to confirm this hypothesis. The most frequently invoked method is by far `document.createElement`, used by 346 scripts. This means that in order to fully understand a given obfuscated script, an analysis needs to reason about the HTML code injected in the page, which seem to be a popular idiom. Moreover, we observe that 295 scripts call `document.createElement('script')`, i.e., inject code at runtime, a technique commonly used by malware.

We manually inspect some of the traces and find two security-relevant behavioral patterns. First, several traces contain various API calls known to be used for browser fingerprinting, such as `location.href`, `screen.width`, `screen.colorDepth`, `navigator.plugins` or `window.devicePixelRatio`. There is even a trace in which after multiple such property accesses, a call to `document.createElement('img')` is made, which suggests that the obfuscated script is sending this information over the network. The second interesting case is of a trace containing 336,000 invocations to `performance.now`, which is likely to be part of a timing attack.

An additional observation we make after our sampling-based manual analysis is that some scripts set global properties, such as `SHA256_init`, `jQueryPath`, `mtTracking` or `Fingerprint2`. Their names suggest that some of the scripts register benign functionality, such as SHA hash functions or jQuery, while others register tracking and fingerprinting functionality.

> **Finding 6.** Multiple obfuscated scripts access privacy-sensitive APIs and use dynamic code loading.

[6] This classification may be wrong for websites that split their files across multiple top-level domains. Finding more accurate ways to identify third-party scripts is left for future work.

Table 6: Libraries used to measure performance and correctness.

Library	Category	Number of tests
Bacon	Reactive programming	7,492
async	Asynchronous programming	513
immutable	Immutable data structures	557
lodash	General utility	6,685
math	Math utility	4,063
moment	Date utility	3,232
ramda	Functional programming	954
underscore	General utility	1,569
voca	String utility	409
when	Promise implementation	872

3.7 RQ5. Performance of Transformed Code

Code transformations may not only influence the understandability of code but also its efficiency. To better understand the cost-benefit tradeoff of transformations, we study to what extent transformations affect the performance of code.

3.7.1 Benchmarks. Addressing RQ5 and RQ6 requires JavaScript code for which we can measure both the performance and the correctness. For this purpose, we gather a set of popular client-side JavaScript libraries that have extensive test suites. These tests include assertions to check the correctness of execution behavior, and they provide a reliable way to measure the execution time of code. Table 6 presents the ten libraries we consider, along with the number of tests they provide. All libraries are frequently used in client-side web applications. We execute their unit tests on Node.js, though, because it facilitates the performance measurements, as they are not influenced by opening and initializing a browser.

To study how transformations affect the performance and correctness of the libraries, we apply all 46 transformations (Section 2.2.2) to each library. To measure the performance of a given, possibly transformed, library, we execute its test suite 20 times and measure the overall wall-clock time of each execution. We do not perform a separate warm-up phase before measuring performance, as is common for longer-running benchmarks, to include the time for parsing code into our measurements, as this time affects the user experience on websites. Since closure injects code that is not compatible with our environment and prevents us from running the library tests, we omit this transformation tool for RQ5 and RQ6.

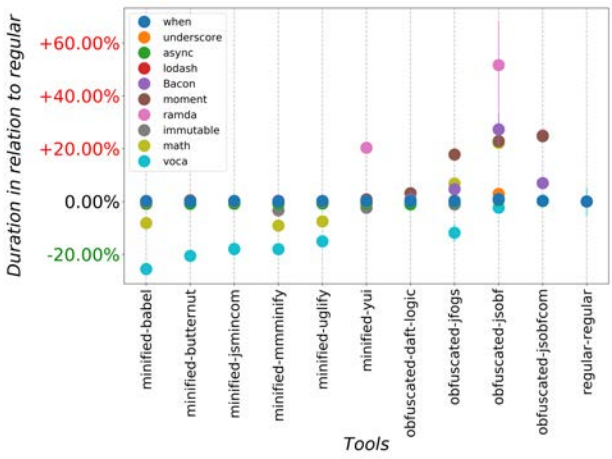


Figure 5: Execution time of transformed code relative to regular code. Each data point shows the average across 20 repetitions and the 95% confidence interval (which is too small to be visible for most libraries).

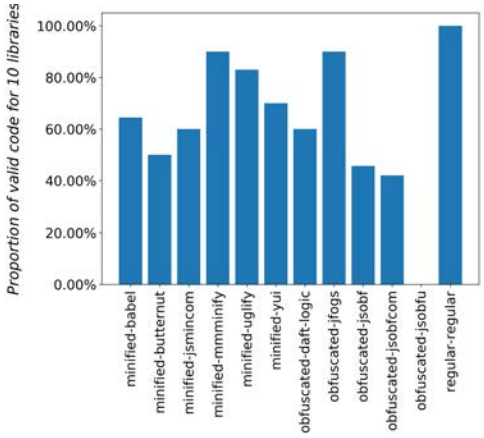


Figure 6: Percentage of correct code among all code transformed by a specific tool.

We run all our performance and correctness measurements on an AMD Phenom II X6 1100T CPU with 16GB of RAM.

3.7.2 Results. Figure 5 shows the execution time of transformed code relatively to regular code for eleven different transformation tools. The figure presents results from the ten libraries listed in Section 3.7.1. Each data point shows the average execution time across 20 repetitions and the 95% confidence interval. The first six columns of data points are for minification tools, whereas the following four columns are for obfuscation tools. The last column shows the baseline, i.e. the performance of the original code.

For minification, the figure shows an overall improvement of efficiency. In particular, for two libraries, voca and math, minification causes performance improvements of over 20% and 8%, respectively. For most of the other libraries, minification causes a measurable but small performance improvement.

In contrast to minification, we observe an overall performance degradation after obfuscating code. For at least four libraries (ramda, moment, math, and Bacon), obfuscation significantly increases the execution time, with average increases between 16% to 37%. For eight out of the ten libraries, the obfuscated code is measurably slower than the regular code. The only outlier is voca, which executes faster for two of the applied obfuscation tools.

> **Finding 7.** Minification either improves performance or has no noticeable effect. In contrast, obfuscation can cause significant performance degradations.

3.8 RQ6. Correctness of Transformed Code

Besides affecting the performance of code, there is another potential cost of applying code transformations: the impact of transformations on the correctness of the code. To assess this impact, we run the test suites of the benchmarks from Section 3.7.1 before and after applying different transformations and measure the percentage of tests that still succeed after the transformation. We say that transformed code is *correct* if this percentage is 100%, i.e., the transformed code passes all tests.

Figure 6 shows the percentage of correct code among all code transformed with a specific tool. Overall, only about 70% of the minified code is correct, and even worse, less than 50% of the obfuscated code is correct. The original code, shown in the right-most column, is by definition 100% correct. A manual analysis of transformed code that fails test cases shows two root causes. First, some transformation tools have implementation-level bugs that get triggered by some code. For example, JSObfu, which consistently creates incorrect code extensively uses the `String.fromCharCode()` method to encode constant string occurring in the code. However, in some cases the method is applied on an undefined object instead of a string, which causes an exception. Second, some transformation tools change the semantics of code in ways that affect rather subtle corner-cases of the JavaScript language. For example, some configurations of UglifyJS replace non-global function names with short and meaningless names. If such a function is a constructor function, this transformation affects code that creates an object with this function and then checks the name of the constructor using JavaScript's reflection APIs. Some of the tests trigger this corner case, e.g., a test of the "immutable" benchmark checks whether objects of type `Record` have a constructor with the same name.

> **Finding 8.** A large portion of code transformed both by minification and obfuscation does not fully preserve the correctness of the code.

We conclude from these results that transformation tools may not only impose a performance cost, but even worse, risk to change the semantics of code. A practical take-away from this finding is that users of such tools must carefully check the correctness of transformed code, instead of blindly relying on the transformation tool. Our results also motivates future work on validating and improving minification and obfuscation tools, e.g., through automated testing.

4 RELATED WORK

This work relates to two active research fields: (i) detection of obfuscated and malicious code and (ii) empirical studies of the web.

4.1 Obfuscated and Malicious Code Detection

Tellenbach et al. [33] propose multiple classifiers for detecting obfuscation. However, they manually specify features of code files, such as the Shannon entropy or the number of characters per line. Similar, Likarish et al. [15] manually extract 60 features from code and propose four different classifiers. In contrast to both, our AST-based classifier does not require any feature engineering.

Wang et al. [36] present a neural network-based classifier for detecting malicious JavaScript code. Since attackers often hide the malicious intent of their scripts using obfuscation, a significant proportion of the dataset used for training their classifier consists of obfuscated code. Comparable to our approach, instead of defining explicit features, they learn the features automatically from the code with the help of multiple layers of stacked denoising autoencoders in the neural network. They transform the code files by replacing each character with a unique binary vector. The vectors require 20,000 dimensions to represent all characters of the dataset. Therefore, Wang et al. reduce the dimensionality to 480 using a dimensionality reduction algorithm. In contrast, we learn a vector representation for AST nodes which only requires 45 dimensions.

Al-Taharwa et al. [1] present a Bayesian-based obfuscation detector aimed at manually performed obfuscation. In contrast, we consider automatically transformed code. Kaplan et al. [10] and Curtsinger et al. [7] extract features from AST nodes while preserving the context of nodes. Their classifiers specialize on obfuscated and on malicious code, respectively. The context-based feature extraction mechanism is comparable to the concept we follow by processing entire ASTs of code files to preserve the context of all AST nodes. However, all three approaches limit the number of allowed contexts and reduce the number of features by applying feature selection. In our case, we do not discard any information from the ASTs so that the neural network can extract the most descriptive features.

There are further static classifier-based approaches which are less related to our approach. Jodavi et al. [9] address the detection of obfuscation with an ensemble of multiple one-class SVM classifiers which are trained to recognize non-obfuscated code using a set of structural and lexical features, such as the number of dynamic code evaluations and the maximum entropy of strings.

Certain researchers address the problem of detecting obfuscated code by using techniques other than machine learning. Xu et al. [38] propose an approach to detect malicious obfuscated code using both static and dynamic analysis. They mostly focus on obfuscation techniques which require the usage of the `eval` function, the `unescape` function, or other related functions. Using static analysis, they gather information about function definitions and invocations in the code. At runtime, they examine if new function definitions and invocations are present to reveal the potentially malicious part of the code. By comparison, we detect if code is obfuscated without assuming a malicious intent because obfuscation allows for different goals, e.g. the protection of intellectual property of code. Furthermore, we include all techniques offered by the obfuscation tools we use for this study instead of focusing only on `eval` based obfuscation. As discovered in Section 2.2.2, most JavaScript obfuscators do not implement `eval` based techniques.

Ceccato et al. [5] propose using out of the box obfuscators for hindering portability of attacks. We also use readily available obfuscators, but for generating training data, not for software diversification.

Deobfuscation is the process of reverse engineering obfuscated code. Techniques for purpose include learning-based approaches [3, 25] and semantics-preserving rewriting of the obfuscated code [16, 39]. This work is complementary to our work and can be applied after detection obfuscated code.

4.2 Empirical Studies

There are a few obfuscation related studies in other programming language contexts. Ceccato et al. [4] analyze the impact of different obfuscation patterns on Java code in terms of the difference between the obfuscated and the original code with the help of metrics. Wang et al. [35] examine the characteristics of obfuscated iOS Apps, the popularity of different obfuscation patterns, and the difficulty of reverse engineering the obfuscated apps. Hammad et al. [8] perform a study concerning the effect of obfuscated Android apps on anti-malware products. They find that the performance of most anti-malware products is significantly impacted by obfuscated apps. Moreover, they report that the tools used for obfuscating the apps frequently result in corrupt apps. Depending on the tool, only 0% to 62% of all 250 apps are runnable after the obfuscation is applied. We observe a similar outcome in the case of JavaScript obfuscation.

Visaggio et al. [34] compare obfuscated and regular JavaScript code using several metrics, including n-gram, entropy, and word size. They find that the two kinds of code differ from each other, in particular, when combining multiple metrics. XU et al. [37] study 510 samples of malicious JavaScript code, its use of obfuscation, and how obfuscation influences whether an anti-virus checker detects the code as malicious. In contrast, our study considers many more scripts, goes beyond obfuscated code, and addresses different research questions.

There have been various JavaScript and web security related studies recently, including a study on the prevalence of the `eval` function [27], on trust relationships between websites that include remote libraries and their corresponding library providers [21], on the communication between websites and embedded frames with 3rd-party content [30], on outdated libraries in the web [13], on XSS vulnerabilities [17], on ReDoS vulnerabilities in JavaScript-based web servers [31], and on performance issues in JavaScript [29]. However, there has been no study focusing on obfuscation and minification in the web outside of malicious code. We conduct the first comprehensive study examining the prevalence, the performance, and the validity of obfuscated and minified JavaScript code in the web.

4.3 JavaScript Analysis

Our work relates to program analyses for JavaScript [2], e.g., to detect conflicts between libraries [22], type inconsistencies [23], code injection attacks [32], or data races [20]. In contrast to that work,

we here address an analysis task by training a machine learning instead of implementing an analysis by hand.

5 CONCLUSION

This paper presents the first large-scale study of minified and obfuscated JavaScript in client-side website applications. Our study leads to several findings that should be of interest for both the web community and the security community. In particular, we show that transformed code is surprisingly common, whereas obfuscation aimed at hindering understanding remains an exception. Yet, there is a non-negligible number of obfuscated scripts, which typically occur in specific website categories and which sometimes expose security-relevant behavior, e.g., fingerprinting or timing attacks. Moreover, we show that obfuscation not only provides benefits to its users but also imposes a cost by negatively impacting both the performance and the correctness of the code. Besides these findings we are releasing the obfuscated code detected during the study to stimulate future research in this area.

ACKNOWLEDGMENTS

This work was supported by the German Federal Ministry of Education and Research and by the Hessian Ministry of Science and the Arts within CRISP, by the German Research Foundation within the ConcSys and Perf4JS projects, and by the Hessian LOEWE initiative within the Software-Factory 4.0 project.

REFERENCES

[1] Ismail Adel AL-Taharwa, Hahn-Ming Lee, Albert B. Jeng, Kuo-Ping Wu, Cheng-Seen Ho, and Shyi-Ming Chen. 2015. JSOD: JavaScript Obfuscation Detector. *Sec. and Commun. Netw.* 8, 6 (April 2015), 1092–1107. https://doi.org/10.1002/sec.1064

[2] Esben Andreasen, Liang Gong, Anders Møller, Michael Pradel, Marija Selakovic, Koushik Sen, and Cristian-Alexandru Staicu. 2017. A Survey of Dynamic Analysis and Test Generation for JavaScript. *Comput. Surveys* (2017).

[3] Rohan Bavishi, Michael Pradel, and Koushik Sen. 2018. Context2Name: A Deep Learning-Based Approach to Infer Natural Variable Names from Usage Contexts. *CoRR* arXiv:1809.05193 (2018).

[4] Mariano Ceccato, Andrea Capiluppi, Paolo Falcarin, and Cornelia Boldyreff. 2015. A large study on the effect of code obfuscation on the quality of Java code. *Empirical Software Engineering* 20, 6 (01 Dec 2015), 1486–1524. https://doi.org/10.1007/s10664-014-9321-0

[5] Mariano Ceccato, Paolo Falcarin, Alessandro Cabutto, Yosief Weldezghi Frezghi, and Cristian-Alexandru Staicu. 2016. Search Based Clustering for Protecting Software with Diversified Updates. In *Search Based Software Engineering - 8th International Symposium, SSBSE 2016, Raleigh, NC, USA, October 8-10, 2016, Proceedings*. 159–175. https://doi.org/10.1007/978-3-319-47106-8_11

[6] Marco Cova, Christopher Krügel, and Giovanni Vigna. 2010. Detection and analysis of drive-by-download attacks and malicious JavaScript code. *International Conference on World Wide Web (WWW)*.

[7] Charlie Curtsinger, Benjamin Livshits, Benjamin Zorn, and Christian Seifert. 2011. ZOZZLE: Fast and Precise In-browser JavaScript Malware Detection. In *Proceedings of the 20th USENIX Conference on Security (SEC'11)*. USENIX Association, Berkeley, CA, USA, 3–3. http://dl.acm.org/citation.cfm?id=2028067.2028070

[8] Mahmoud Hammad, Joshua Garcia, and Sam Malek. 2017. A Large-Scale Empirical Study on the Effects of Code Obfuscations on Android Apps and Anti-Malware Products. *Proceedings of the 2018 IEEE 26th International Conference on Program Comprehension - ICSE 2018*.

[9] Mehran Jodavi, Mahdi Abadi, and Elham Parhizkar. 2015. JSObfusDetector: A binary PSO-based one-class classifier ensemble to detect obfuscated JavaScript code. *Proceedings of the International Symposium on Artificial Intelligence and Signal Processing, AISP 2015* (2015), 322–327. https://doi.org/10.1109/AISP.2015.7123508

[10] Scott Kaplan, Ben Livshits, Ben Zorn, Christian Siefert, and Charlie Cursinger. 2011. *"NOFUS: Automatically Detecting" + String.fromCharCode(32) + "ObFuSCateD ".toLowerCase() + "JavaScript Code"*. Technical Report.

[11] Alexandros Kapravelos, Chris Grier, Neha Chachra, Christopher Kruegel, Giovanni Vigna, and Vern Paxson. 2014. Hulk: Eliciting Malicious Behavior in Browser Extensions. *Proceedings of the 23rd USENIX Conference on Security*.

[12] Rezwana Karim, Mohan Dhawan, Vinod Ganapathy, and Chung-chieh Shan. 2012. An Analysis of the Mozilla Jetpack Extension Framework. *ECOOP 2012 - Object-Oriented Programming - 26th European Conferenc*.

[13] Tobias Lauinger, Abdelberi Chaabane, Sajjad Arshad, William Robertson, Christo Wilson, and Engin Kirda. 2017. Thou Shalt Not Depend on Me: Analysing the Use of Outdated JavaScript Libraries on the Web. In *Network and Distributed System Security Symposium (NDSS)*. The Internet Society.

[14] Sebastian Lekies, Ben Stock, and Martin Johns. 2013. 25 million flows later: large-scale detection of DOM-based XSS. *ACM SIGSAC Conference on Computer and Communications Security (CCS)*.

[15] Peter Likarish, Eunjin Jung, and Insoon Jo. 2009. Obfuscated malicious JavaScript detection using classification techniques. In *2009 4th International Conference on Malicious and Unwanted Software, MALWARE 2009*. 47 – 54.

[16] Gen Lu and Saumya K. Debray. 2012. Automatic Simplification of Obfuscated JavaScript Code: A Semantics-Based Approach. *International Conference on Software Security and Reliability (SERE)*.

[17] William Melicher, Anupam Das, Mahmood Sharif, Lujo Bauer, and Limin Jia. 2018. Riding out DOMsday: Towards Detecting and Preventing DOM Cross-Site Scripting. *Network and Distributed System Security Symposium (NDSS)*.

[18] Lili Mou, Ge Li, Lu Zhang, Tao Wang, and Zhi Jin. 2016. Convolutional Neural Networks over Tree Structures for Programming Language Processing. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, 1287–1293. http://dl.acm.org/citation.cfm?id=3015812.3016002

[19] Keaton Mowery and Hovav Shacham. 2012. Pixel perfect: Fingerprinting canvas in HTML5. *Web 2.0 Security & Privacy, (W2SP)*.

[20] Erdal Mutlu, Serdar Tasiran, and Benjamin Livshits. 2015. Detecting JavaScript Races that Matter. In *European Software Engineering Conference and International Symposium on Foundations of Software Engineering (ESEC/FSE)*.

[21] Nick Nikiforakis, Luca Invernizzi, Alexandros Kapravelos, Steven Van Acker, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2012. You Are What You Include: Large-scale Evaluation of Remote Javascript Inclusions. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM.

[22] Jibesh Patra, Pooja N. Dixit, and Michael Pradel. 2018. ConflictJS: Finding and Understanding Conflicts Between JavaScript Libraries. In *ICSE*. 741–751.

[23] Michael Pradel, Parker Schuh, and Koushik Sen. 2015. TypeDevil: Dynamic Type Inconsistency Analysis for JavaScript. In *International Conference on Software Engineering (ICSE)*.

[24] Veselin Raychev, Pavol Bielik, Martin T. Vechev, and Andreas Krause. 2016. Learning programs from noisy data. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, (POPL)*. ACM.

[25] Veselin Raychev, Martin T. Vechev, and Andreas Krause. 2015. Predicting Program Properties from "Big Code".. In *Principles of Programming Languages (POPL)*. 111–124.

[26] Andreas Reiter and Alexander Marsalek. 2017. WebRTC: your privacy is at risk. *Proceedings of the Symposium on Applied Computing, SAC 2017*.

[27] Gregor Richards, Christian Hammer, Brian Burg, and Jan Vitek. 2011. The Eval That Men Do. In *ECOOP 2011 – Object-Oriented Programming*, Mira Mezini (Ed.). Springer Berlin Heidelberg.

[28] Prateek Saxena, Steve Hanna, Pongsin Poosankam, and Dawn Song. 2010. FLAX: Systematic Discovery of Client-side Validation Vulnerabilities in Rich Web Applications. *Network and Distributed System Security Symposium (NDSS)*.

[29] Marija Selakovic and Michael Pradel. 2016. Performance Issues and Optimizations in JavaScript: An Empirical Study. In *International Conference on Software Engineering (ICSE)*. 61–72.

[30] Sooel Son and Vitaly Shmatikov. 2013. The Postman Always Rings Twice: Attacking and Defending postMessage in HTML5 Websites. In *Network and Distributed System Security Symposium (NDSS)*. The Internet Society.

[31] Cristian-Alexandru Staicu and Michael Pradel. 2018. Freezing the Web: A Study of ReDoS Vulnerabilities in JavaScript-based Web Servers. In *USENIX Security Symposium*. 361–376.

[32] Cristian-Alexandru Staicu, Michael Pradel, and Ben Livshits. 2018. Understanding and Automatically Preventing Injection Attacks on Node.js. In *Network and Distributed System Security Symposium (NDSS)*.

[33] Bernhard Tellenbach, Sergio Paganoni, and Marc Rennhard. 2016. Detecting Obfuscated JavaScripts using Machine Learning . *International Journal on Advances in Security* 9, 3 & 4 (2016), 196–206.

[34] Corrado Aaron Visaggio, Giuseppe Antonio Pagin, and Gerardo Canfora. 2013. An empirical study of metric-based methods to detect obfuscated code. *International Journal of Security and its Applications* 7, 2 (2013), 59–74.

[35] Pei Wang, Qinkun Bao, Li Wang, Shuai Wang, Zhaofeng Chen, and Tao Wei. 2018. Software Protection on the Go: A Large-Scale Empirical Study on Mobile App Obfuscation. In *Proceedings of the 40th International Conference on Software Engineering, ICSE 2018*. ACM, New York, NY, USA, 11.

[36] Yao Wang, Wan-dong Cai, and Peng-cheng Wei. 2016. A deep learning approach for detecting malicious JavaScript code. 9 (02 2016).

[37] Wei Xu, Fangfang Zhang, and Sencun Zhu. 2012. The power of obfuscation techniques in malicious JavaScript code: A measurement study. In *7th International*

Conference on Malicious and Unwanted Software, MALWARE 2012, Fajardo, PR, USA, October 16-18, 2012. 9–16.

[38] W Xu, F Zhang, and S Zhu. 2013. JStill: Mostly static detection of obfuscated malicious javascript code . *Proceedings of the 3rd ACM Conference on Data and Application Security and Privacy (CODASPY)* (2013), 117–128. http://www.cse.psu.edu/{~}szhu/papers/JStill.pdf

[39] Khaled Yakdan, Sergej Dechand, Elmar Gerhards-Padilla, and Matthew Smith. 2016. Helping Johnny to Analyze Malware: A Usability-Optimized Decompiler and Malware Analysis User Study. *IEEE Symposium on Security and Privacy (SP)*.

SciLens: Evaluating the Quality of Scientific News Articles Using Social Media and Scientific Literature Indicators

Panayiotis Smeros
École Polytechnique Fédérale de Lausanne (EPFL)
Lausanne, Switzerland
panayiotis.smeros@epfl.ch

Carlos Castillo
Universitat Pompeu Fabra (UPF)
Barcelona, Catalunya, Spain
carlos.castillo@upf.edu

Karl Aberer
École Polytechnique Fédérale de Lausanne (EPFL)
Lausanne, Switzerland
karl.aberer@epfl.ch

ABSTRACT

This paper describes, develops, and validates SciLens, a method to evaluate the quality of scientific news articles. The starting point for our work are structured methodologies that define a series of quality aspects for manually evaluating news. Based on these aspects, we describe a series of indicators of news quality. According to our experiments, these indicators help non-experts evaluate more accurately the quality of a scientific news article, compared to non-experts that do not have access to these indicators. Furthermore, SciLens can also be used to produce a completely automated quality score for an article, which agrees more with expert evaluators than manual evaluations done by non-experts. One of the main elements of SciLens is the focus on both content and context of articles, where context is provided by (1) explicit and implicit references on the article to scientific literature, and (2) reactions in social media referencing the article. We show that both contextual elements can be valuable sources of information for determining article quality. The validation of SciLens, done through a combination of expert and non-expert annotation, demonstrates its effectiveness for both semi-automatic and automatic quality evaluation of scientific news.

KEYWORDS

ACM Reference Format:
Panayiotis Smeros, Carlos Castillo, and Karl Aberer. 2019. SciLens: Evaluating the Quality of Scientific News Articles Using Social Media and Scientific Literature Indicators. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313657

1 INTRODUCTION

Scientific literacy is broadly defined as a knowledge of basic scientific facts and methods. Deficits in scientific literacy are endemic in many societies, which is why understanding, measuring, and furthering the public understanding of science is important to many scientists [6].

Mass media can be a potential ally in fighting scientific illiteracy. Reading scientific content has been shown to help align public knowledge of scientific topics with the scientific consensus, although in highly politicized topics it can also reinforce pre-existing biases [27]. There are many ways in which mass media approaches science, and even within the journalistic practice there are several sub-genres. Scientific news portals, for instance, include most of the categories of articles appearing traditionally in newspapers [21] such as *editorial*, *op-ed*, and (less frequently) *letters to the editor*. The main category of articles, however, are scientific *news* articles, where journalists describe scientific advances.

Scientific news articles have many common characteristics with other classes of news articles; for instance, they follow the well-known *inverted pyramid* style, where the most relevant elements are presented at the beginning of the text. However, they also differ in important ways. Scientific news are often based on findings reported in scientific journals, books, and talks, which are highly specialized. The task of the journalist is then to *translate* these findings to make them understandable to a non-specialized, broad audience. By necessity, this involves negotiating several trade-offs between desirable goals that sometimes enter into conflict, including appealing to the public and using accessible language, while at the same time accurately representing research findings, methods, and limitations [46].

The resulting portrayal of science in news varies widely in quality. For example, the website "Kill or Cure?"[1] has reviewed over 1,200 news stories published by The Daily Mail (a UK-based tabloid) finding headlines pointing to 140 substances or factors that cause cancer (including obesity, but also Worcestershire sauce), 113 that prevent it (including garlic and green tea), and 56 that both cause and prevent cancer (including rice). Evidently, news coverage of cancer research that merely seeks to classify every inanimate object into something that either causes or prevents cancer does not help to communicate effectively scientific knowledge on this subject.

Our contribution. In this paper we describe *SciLens,* a method for evaluating the quality of scientific news articles. The technical contributions we describe are the following:

- a framework, depicted in Figure 1, for semi-automatic and automatic article quality evaluation (§3);
- a method for contextual data collection that captures the contents of an article, its relationship with the scientific literature, and the reactions it generates in social media (§4);
- a series of automatically-computed *quality indicators* describing:
  - the content of a news article, where we introduce a method to use quotes appearing on it as quality indicators (§5.1),
  - the relationship of a news article with the scientific literature, where we introduce content-based and graph-based similarity methods (§5.2), and

[1] http://kill-or-cure.herokuapp.com

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313657

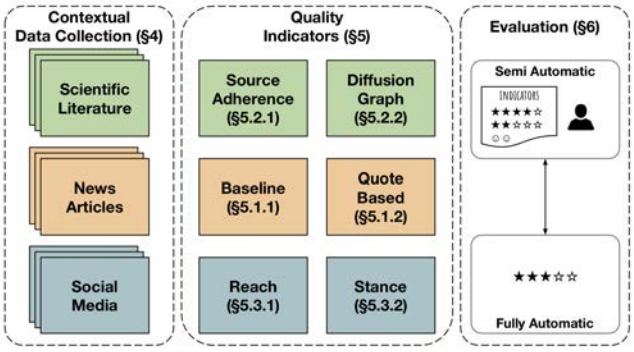


Figure 1: Overview of SciLens, including contextual data collection, quality indicators, and evaluation.

  - the social media reactions to the article, where we introduce a method to interpret their stance (supporting, commenting, contradicting, or questioning) as quality signals (§5.3);
- an experimental evaluation of our methods involving experts and non-experts (§6).

2 RELATED WORK

In this section, we present background information that frames our research (§2.1), previous work on evaluating news quality (§2.2), and methods to extract quality indicators from news articles (§2.3). This is a broad research area where results are scattered through multiple disciplines and venues; our coverage is by no means complete.

2.1 Background on Scientific News

A starting point for understanding communication of science has historically been the "deficit model," in which the public is assumed to have a deficit in scientific information that is addressed by science communication (see, e.g., Gross [25]). In a simplified manner, scientific journalism, as practiced by professional journalists as well as science communicators and bloggers from various backgrounds, can be seen as a translation from a discourse inside scientific institutions to a discourse outside them. However, there are many nuances that make this process much more than a simple translation. For instance, Myers [44], among others, notes that (i) in many cases the gulf between experts and the public is not as large as it may seem, as many people may have some information on the topic; (ii) there is a continuum of popularization through different genres, i.e., science popularization is a matter of degree; and (iii) the scientific discourse is intertwined with other discourses, including the discussion of political and economic issues.

Producing a high-quality article presenting scientific findings to the general public is unquestionably a challenging task, and often there is much to criticize about the outcome. In the process of writing an article, "information not only changes textual form, but is simplified, distorted, hyped up, and dumbed down" [44]. Misrepresentation of scientific knowledge by journalists has been attributed to several factors, including "a tendency to sensationalize news, a lack of analysis and perspective when handling scientific issues, excessive reliance on certain professional journals for the selection of news, lack of criticism of powerful sources, and lack of criteria for evaluating information" [13].

In many cases, these issues can be traced to journalists adhering to journalistic rather than scientific norms. According to Dunwoody [15], this includes (i) a tendency to favor conflict, novelty, and similar news values; (ii) a compromise of accuracy by lack of details that might be relevant to scientists, but that journalists consider uninteresting and/or hard to understand for the public; and (iii) a pursuit of "balance" that mistakenly gives similar coverage to consensus viewpoints and fringe theories. Journalists tend to focus on events or episodic developments rather than long-term processes, which results in preferential coverage to initial findings even if they are later contradicted, and little coverage if results are disconfirmed or shown to be wrong [14]. Furthermore, news articles typically do not include caveat/hedging/tentative language, i.e., they tend to report scientific findings using a language expressing certainty, which may actually have the opposite effect from what is sought, as tentative language makes scientific reporting more credible to readers [32].

2.2 Evaluation of Quality of News

There are many approaches for evaluating the quality of articles on the web; we summarize some of these approaches in Table 1.

Manual Evaluation. The simplest approach for evaluating news article quality relies on the manual work of domain experts. This is a highly subjective task, given that quality aspects such as credibility are to a large extent perceived qualities, made of many dimensions [20]. In the health domain, evaluations of news article quality have been undertaken for both general health topics [53] and specific health topics such as Pancreatic Cancer [57].

Independent, non-partisan *fact-checking portals* perform manual content verification at large scale, typically employing a mixture of professional and volunteer staff. They can cover news articles on general topics (e.g., Snopes.com) or specific topics such as politics (e.g., PolitiFact.com). In the case of science news, ClimateFeedback.org is maintained by a team of experts on climate change with the explicit goal of helping non-expert readers evaluate the quality of news articles reporting on climate change. Each evaluated article is accompanied by a brief review and an overall quality score. Reviews and credibility scores from fact-checking portals have been recently integrated with search results [36] and social media posts [40] to help people find accurate information. Furthermore, they are frequently used as a ground truth to build systems for rumor tracking [54], claim assessment [50], and fake multimedia detection [8]. Articles considered by fact-checking portals as misinformation have been used as "seeds" for diffusion-based methods studying the spread of misinformation [56].

Our approach differs from previous work because it is completely automated and does not need to be initialized with labels from expert- or crowd-curated knowledge bases. For instance, in the *diffusion graph*, which is the graph we construct during contextual data collection (§4) from social media posts and scientific papers, we do not need prior knowledge on the quality of different nodes.

Automatic and Semi-Automatic Evaluation. Recent work has demonstrated methods to automate the extraction of signals or indicators of article quality. These indicators are either expressed at a

Table 1: Summary of selected references describing techniques for evaluating news article quality.

	Fact-checking portals	*Shao et al. [54]*	*Boididou et al. [8]*	*Popat et al. [50]*	*Tambuscio et al. [56]*	*Ciampaglia et al. [11]*	*Urban and Schweiger [58]*	*Zhang et al. [62]*	*Shu et al. [55]*	*Kumar et al. [37]*	*Sbaffi and Rowley [53]*	*Taylor et al. [57]*	**SciLens**
Automatic assessment	✗	✓	✓	✓	✓	✓	✗	✗	✓	✓	✗	✗	✓
No ground-truth needed	✓	✗	✗	✗	✗	✗	✓	✓	✗	✓	✓	✓	✓
Uses article content	✓	✗	✗	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓
Uses reactions from social media	✗	✓	✓	✓	✓	✗	✗	✓	✓	✗	✗	✗	✓
Uses referenced scientific literature	✓	✗	✗	✗	✗	✗	✗	✓	✗	✗	✓	✓	✓
Domain-agnostic	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✗	✗	✓
Web-scale	✗	✓	✓	✓	✓	✓	✗	✗	✓	✓	✗	✗	✓

conceptual level [58] (e.g, *balance of view points*, *respect of personal rights*) or operationalized as features that can be computed from an article [62] (e.g., *expert quotes* or *citations*). Shu et al. [55] describe an approach for detecting fake news on social media based on social and content indicators. Kumar et al. [37] describe a framework for finding hoax Wikipedia pages mainly based on the author's behavior and social circle, while Ciampaglia et al. [11] use Wikipedia as ground truth for testing the validity of dubious claims. Baly et al. [5] describe site-level indicators that evaluate an entire website instead of individual pages.

Our work differs from these by being, to the best of our knowledge, the first work that analyzes the quality of a news article on the web combining its own content with context that includes social media reactions and referenced scientific literature. We provide a framework, *SciLens*, that is scalable and generally applicable to any technical/scientific context at any granularity (from a broad topic such as "health and nutrition" to more specific topics such as "gene editing techniques").

2.3 Indicator Extraction Techniques

Our method relies on a series of indicators that can be computed automatically, and intersects previous literature that describes related indicators used to evaluate article quality or for other purposes.

Quote Extraction and Attribution. The most basic approach to quote extraction is to consider that a quote is a "block of text within a paragraph falling between quotation marks" [16, 51]. Simple regular expressions for detecting quotes can be constructed [45, 52]. Pavllo et al. [48] leverages the redundancy of popular quotes in large news corpora (e.g., highly controversial statements from politicians that are intensely discussed in the press) for building unsupervised bootstrapping models, while Pareti et al. [47] and Jurafsky et al. [34] train supervised machine learning models using corpora of political and literary quotes (Wikiquote, https://www.wikiquote.org, is such a corpus that contains general quotes).

Our work does not rely on simple regular expressions, such as syntactic patterns combined with quotations marks, which in our preliminary experiments performed poorly in quote extraction from science news; instead we use regular expressions based on classes of words. We also do not use a supervised approach as there is currently no annotated corpus for scientific quote extraction. For the research reported on this paper, we built an information extraction model specifically for scientific quotes from scratch, i.e., a "bootstrapping" model, which is based on word embeddings. This is a commonly used technique for information extraction when there is no training data and we can manually define a few high-precision extraction patterns [33].

Semantic Text Similarity. One of the indicators of quality that we use is the extent to which the content of a news article represents the scientific paper(s) it is reporting about. The Semantic Text Similarity task in Natural Language Processing (NLP) determines the extent to which two pieces of text are semantically equivalent. This is a popular task in the *International Workshop on Semantic Evaluation* (SemEval). Three approaches that are part of many proposed methods over the last few years include: (i) *surface-level similarity* (e.g., similarity between sets or sequences of words or named entities in the two documents); (ii) *context similarity* (e.g., similarity between document representations); and (iii) *topical similarity* [26, 38].

In this paper, we adopt these three types of similarity, which we compute at the document, paragraph, and sentence level. The results we present suggest that combining different similarity metrics at different granularities results in notable improvements over using only one metric or only one granularity.

Social Media Stance Classification. Our analysis of social media postings to obtain quality indicators considers their *stance*, i.e., the way in which posting authors position themselves with respect to the article they are posting about. Stance can be binary ("for" or "against"), or be described by more fine-grained types (supporting, contradicting, questioning, or commenting) [28], which is what we employ in this work. Stance classification of social media postings has been studied mostly in the context of online marketing [35] and political discourse and rumors [63].

In our work, we build a new stance classifier based on textual and contextual features of social media postings and replies, annotated by crowdsourcing workers. To the best of our knowledge, there is no currently available corpus covering the scientific domain. As part of our work, we release such corpus.

3 SCILENS OVERVIEW

The goal of SciLens is to help evaluate the quality of scientific news articles. As Figure 1 shows, we consider a contextual data collection, a computation of quality indicators, and an evaluation of the results.

Contextual Data Collection (§4). First, we consider a set of keywords that are representative of a scientific/technical domain; for this paper, we have considered a number of key words and phrases related to health and nutrition. Second, we extract from a social media platform (in this case, Twitter), all postings matching these keywords, as well as public replies to these postings. Third, we follow all links from the postings to web pages, and download such pages; while the majority of them are news sites and blogs of various kinds, we do not restrict the collection by type of site at this point. Fourth, we follow all links from the web pages to URLs in a series of pre-defined domain names from scientific repositories, academic portals and libraries, and universities. Fifth, we clean-up the collection by applying a series of heuristics to de-duplicate articles and remove noisy entries.

Quality Indicators (§5). We compute a series of quality indicators from the content of articles, and from their referencing social media postings and referenced scientific literature.

Regarding the content of the articles, we begin by computing several content-based features described by previous work. Next, we perform an analysis of quotes in articles, which are a part of journalistic practices in general and are quite prevalent in the case of scientific news. Given that attributed quotes are more telling of high quality than unattributed or "weasel" quotes, we also carefully seek to attribute each quote to a named entity which is often a scientist, but can also be an institution.

Regarding the scientific literature, we would like to know the strength of the connection of articles to scientific papers. For this, we consider two groups of indicators: content-based and graph-based. Content-based indicators are built upon various metrics of text similarity between the content of an article and the content of scientific papers, considering that the technical vocabulary is unlikely to be preserved as-is in articles written for the general public. Graph-based indicators are based on a diffusion graph in which scientific papers and web pages in academic portals are nodes connected by links. High-quality articles are expected to be connected through many short paths to academic sources in this graph.

Regarding social media postings, we measure two aspects: reach and stance. Reach is measured through various proxies for attention, that seek to quantify the impact that an article has in social media. The stance is the positioning of posting authors with respect to an article, which can be positive (supporting, or commenting on an article without expressing doubts), or negative (questioning an article, or directly contradicting what the article is saying).

Evaluation (§6). We evaluate the extent to which the indicators computed in SciLens are useful for determining the quality of a scientific news article. We consider that these indicators can be useful in two ways. First, in a semi-automatic setting, we can show the indicators to end-users and ask them to evaluate the quality of a scientific news article; if users that see these indicators are better at this task that users that do not see them, we could claim that the indicators are useful. Second, in a fully automatic setting, we can train a model based on all the indicators that we computed. In both cases, the ground truth for evaluation is provided by experts in communication and science.

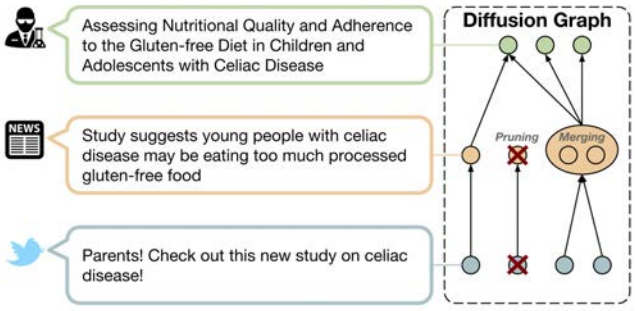


Figure 2: Contextual data collection, including social media postings, which reference a series of news articles, which cite one or more scientific papers. In our diffusion graph, paths that do not end up in a scientific paper or paths that contain unparsable nodes (e.g., malformed HTML pages) are *pruned*, and articles with the same content in two different outlets (e.g., produced by the same news agency) are *merged*.

4 CONTEXTUAL DATA COLLECTION

The contextual data collection in our work seeks to capture all relevant content for evaluating news article quality, including referenced scientific papers and reactions in social media. This methodology can be applied to any specialized or technical domain covered in the news, as long as: (i) media coverage in the domain involves "translating" from primary technical sources, (ii) such technical sources can be characterized by known host/domain names on the web, and (iii) social media reactions can be characterized by the presence of certain query terms. Examples where this type of contextual data collection could be applied beyond scientific news include news coverage of specialized topics such as law or finance.

We consider two phases: a crawling phase, which starts from social media and then collects news articles and primary sources (§4.1), and a pruning/merging phase, which starts from primary sources and prunes/de-duplicates articles and postings (§4.2). This process is depicted in Figure 2 and explained next.

4.1 Crawling of Postings, Articles, and Papers

The crawling phase starts with social media postings, which are identified as candidates for inclusion based on the presence of certain topic-related keywords in them. In the case of this study, we selected "health and nutrition" as our main topic: this is among the most frequent topics in scientific news reporting, which is known to have a medical/health orientation [4, 15, 61]. The initial set of keywords was obtained from Nutrition Facts (https://nutritionfacts.org/topics), a non-commercial and non-profit website that provides scientific information on healthy eating. The list contains over 2,800 keywords and key phrases such as "HDL cholesterol," "polyphenols" and the names of hundreds of foods from "algae" to "zucchini". We further expanded this list with popular synonyms from WordNet [42].

We harvest social media postings from DataStreamer.io (formerly known as Spinn3r.com), covering a 5-year period from June 2013 through June 2018. In this collection, we find 2.5M candidate postings matching at least one of our query terms from which we discard postings without URLs.

Next, we crawl the pages pointed to by each URL found in the remaining postings. These pages are hosted in a wide variety of domains, the majority being news outlets and blogging platforms. We scan these pages for links to scientific papers, which we do identify by domain names. We use a predefined list of the top-1000 universities as indicated by CWUR.org plus a manually curated list of open-access publishers and academic databases obtained from Wikipedia[2] and expanded using the "also visited websites" functionality of SimilarWeb.com. Overall, we obtained a diffusion graph of **2.4M** nodes and **3.7M** edges.

4.2 Pruning and Merging

The initial data collection described in §4.1 is recall-oriented. Now, we make it more precise by pruning and merging items.

Pruning. During the pruning phase, we keep in our collection only documents that we managed to successfully download and parse (e.g., we discard malformed HTML pages and PDFs). We also prune paths that do not end up in a scientific paper i.e., articles that do not have references and all the tweets that point to these articles. This phase helps us eliminate most of the noisy nodes of the diffusion graph that were introduced due to the generic set of seed keywords that we used in the crawling phase (§4.1).

Merging. We notice a large number of duplicate articles across news outlets, which we identify by text similarity i.e, by cosine similarity of more than 90% between the bag-of-words vectors representing the articles. This happens when one outlet re-posts an article originally published in another outlet, or when both syndicate from the same news agency. Once we find such duplicates or near-duplicates, we keep only one of them (the one having more out-links, breaking ties arbitrarily) and remove the redundant ones. Social media postings that point to the duplicates are re-wired to connect to the one that survived after merging, hence we do not lose a potentially important signal of article quality.

The resulting collection is large and mostly composed of elements that are closely related to the topic of health and nutrition: **49K** social media postings, **12K** articles (most of them in news sites and blogs), and **24K** scientific links (most of them peer-reviewed or grey-literature papers). Even after pruning, our collection is possibly more comprehensive than the ones used by systems used to appraise the impact of scientific papers. For instance, when compared to Altmetric.com [1] we find that our collection has more links to scientific papers than what Altmetric counts. In their case, referencing articles seem to be restricted to a controlled list of mainstream news sources, while in our case we often find these references plus multiple references from less known news sources, blogs, and other websites.

[2]https://en.wikipedia.org/wiki/List_of_academic_databases_and_search_engines

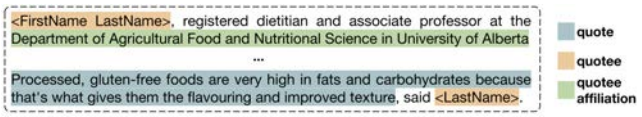


Figure 3: Example of quote extraction and attribution (best seen in color). Quotee has been anonymized.

5 QUALITY INDICATORS

We compute indicators from the content of news articles (§5.1), from the scientific literature referenced in these articles (§5.2), and from the social media postings referencing them (§5.3). The full list of indicators is presented on Table 2.

5.1 News Article Indicators

These indicators are based on the textual content of a news article.

5.1.1 Baseline Indicators. As a starting point, we adopt a large set of content-based quality indicators described by previous work. These indicators are: (i) title deceptiveness and sentiment: we consider if the title is "clickbait" that oversells the contents of an article in order to pique interest [41, 60]; (ii) article readability: indicator of the level of education someone would need to easily read and understand the article [19]; and (iii) article length and presence of author byline [62].

5.1.2 Quote-Based Indicators. Quotes are a common and important element of many scientific news articles. While selected by journalists, they provide an opportunity for experts to directly present their viewpoints in their own words [12]. However, identifying quotes in general is challenging, as noted by previous work (§2.3). In the specific case of our corpus, we observe that they are seldom contained in quotation marks in contrast to other kinds of quotes (e.g., political quotes [51]). We also note that each expert quoted tends to be quoted once, which makes the problem of *attributing* a quote challenging as well.

Quote Extraction Model. To extract quotes we start by addressing a classification problem at the level of a sentence: we want to distinguish between quote-containing and non-containing sentences. To achieve this, we first select a random sample from our dataset, then manually identify quote patterns, and finally, we generalize automatically these patterns to cover the full dataset. As we describe in the related work section (§2.3), this is a "bootstrapping" model built from high-precision patterns, as follows.

The usage of *reporting verbs* is a typical element of quote extraction models [47]. Along with common verbs that are used to quote others (e.g., "say," "claim") we used verbs that are common in scientific contexts, such as "prove" or "analyze." First, we manually create a seed set of such verbs. Next, we extend it with their nearest neighbors in a word embedding space; the word embeddings we use are the *GloVe* embeddings, which are trained on a corpus of Wikipedia articles [49]. We follow a similar approach for nouns related to studies (e.g., "survey," "analysis") and nouns related to scientists (e.g., "researcher," "analyst"). Syntactically, quotes are usually expressed using indirect speech. Thus, we also obtain part-of-speech tags from the candidate quote-containing sentences.

Using this information, we construct a series of regular expressions over *classes* of words ("reporting verbs," "study-related noun," and part-of-speech tags) which we evaluate in §6.1.

Quote Attribution. For the purposes of evaluating article quality, it is fundamental to know not only that an article has quotes, but also their provenance: *who* or *what* is being quoted. After extracting all the candidate quote-containing sentences, we categorize them according to the information available about their quotee.

A quotee can be an *unnamed scientist* or an *unnamed study* if the person or article being quoted is not disclosed (e.g., "researchers believe," "most scientists think" and other so-called "weasel" words). Sources that are not specifically attributed such as these ones are as a general rule considered less credible than sources in which the quotee is named [62].

A quotee can also be a *named entity* identifying a specific person or organization. In this case, we apply several heuristics for quote attribution. If the quotee is a *named person*, if she/he is referred with her/his last or first name, we search within the article for the full name. When the full name is not present in the article, we map the partial name to the most common full name that contains it within our corpus of news articles. We also locate sentences within the article that mention this person together with a named organization. This search is performed from the beginning of the article as we assume they follow an *inverted pyramid* style. In case there are several, the most co-mentioned organization is considered as the affiliation of the quotee.

If the quotee is an *organization*, then it can be either mentioned in full or using an acronym. We map acronyms to full names of organizations when possible (e.g., we map "WHO" to "World Health Organization"). If the full name is not present in an article, we follow a similar procedure as the one used to determine the affiliation of a researcher, scanning all the articles for co-mentions of the acronym and a named organization.

An illustrative example of the extraction and the attribution phase can be shown in Figure 3.

Scientific Mentions. News articles tend to follow journalistic conventions rather than scientific ones [15]; regarding citation practices, this implies they seldom include formal references in the manner in which one would find them in a scientific paper. Often there is no explicit link: journalists may consider that the primary source is too complex or inaccessible to readers to be of any value, or may find that the scientific paper is located in a "pay-walled" or otherwise inaccessible repository. However, even when there is no explicit link to the paper(s) on which an article is based, good journalistic practices still require to identify the information source (institution, laboratory, or researcher).

Mentions of academic sources are partially obtained during the quote extraction process (§5.1.2), and complemented with a second pass that specifically looks for them. During the second pass, we use the list of universities and scientific portals that we used during the *crawling phase* of the data collection (§4.1).

5.2 Scientific Literature Indicators

In this section, we describe content- and graph-based indicators measuring how articles are related to the scientific literature.

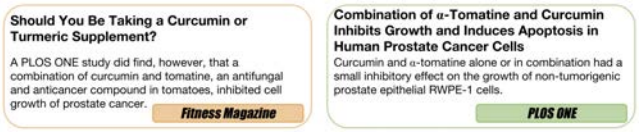


Figure 4: A news article (left) and a scientific paper (right) with Semantic Text Similarity of 87.9%. Indicatively, two passages from these documents, whose conceptual similarity is captured by our method, are presented. In these two passages we can see the effort of the journalist on translating from an academic to a less formal language, without misrepresenting the results from the paper.

5.2.1 Source Adherence Indicators. When there is an explicit link from a news article to the URL where a scientific paper is hosted, we can measure the extent to which these two documents convey the same information. This is essentially a computation of the *Semantic Text Similarity* (STS) between the news article and its source(s).

Supervised Learning for STS. We construct an STS model using supervised learning. The **features** that we use as input to the model consist of the following text similarity metrics: (i) the Jaccard similarity between the sets of named entities (persons and organizations), dates, numbers and percentages of the two texts; (ii) the cosine similarity between the *GloVe* embeddings of the two texts; (iii) the Hellinger similarity [30] between topic vectors of the two texts (obtained by applying LDA [7]); and (iv) the relative difference between the length in words of the two texts. Each of them is computed three times: (1) considering the entire contents of the article and the paper; (2) considering one paragraph at a time, and then computing the average similarity between a paragraph in one document and a paragraph in the other; and (3) considering one sentence at a time, and then computing the average similarity between a sentence in one document and a sentence in the other. In other words, in (2) and (3) we compute the average of each similarity between the Cartesian product of the passages.

The **training data** that we use is automatically created from pairs of documents consisting of a news article and a scientific paper. Whenever a news article has exactly one link to a scientific paper, we add the article and the paper to training data in the positive class. For the negative class, we sample random pairs of news articles and papers. The **learning schemes** used are Support Vector Machine, Random Forests and Neural Networks. Details regarding the evaluation of these schemes are provided in §6.1.2. An example of a highly related pair of documents, as determined by this method, is shown in Figure 4.

Handling Multi-Sourced Articles. When an article has a single link to a scientific paper, we use the STS of them as an indicator of quality. When an article has multiple links to scientific papers, we select the one that has the maximum score according to the STS model we just described. We remark that this is just an indicator of article quality and we do not expect that by itself it is enough to appraise the quality of the article. Deviations from the content of the scientific paper are not always wrong, and indeed a good journalist might consult multiple sources and summarize them in a way that re-phrases content from the papers used as sources.

5.2.2 Diffusion Graph Indicators. We also consider that referencing scientific sources, or referencing pages that reference scientific sources, are good indicators of quality. Figure 2 showing a graph from scientific papers to articles, and from articles to social media postings and from them to their reactions, suggests this can be done using graph-based indicators. We consider the following:

(1) personalized PageRank [29] on the graph having scientific articles and universities as root nodes and news articles as leaf nodes; and
(2) betweenness and degree on the full diffusion graph [22, 23].

Additionally, we consider the traffic score computed by Alexa.com for the website in which each article is hosted, which estimates the total number of visitors to a website.

5.3 Social Media Indicators

We extract signals describing the quantity and characteristics of social media postings referencing each article. Quantifying the amount of reactions in various ways might give us signals about the interest in different articles (§5.3.1). However, this might be insufficient or even misleading, if we consider that false news may reach a larger audience and propagate faster than actual news [59]. Hence, we also need to analyze the content of these postings (§5.3.2).

5.3.1 Social Media Reach. Not every social media user posting the URL of a scientific news article agrees with the article's content, and not all users have sufficient expertise to properly appraise its contents. Indeed, sharing articles and reading articles are often driven by different mechanisms [2]. However, and similarly to citation analysis and to link-based ranking, the volume of social media reactions to an article might be a signal of its quality, although the same caveats apply.

Given that we do not have access to the number of times a social media posting is shown to users, we extract several proxies of the *reach* of such postings. First, we consider the total number of postings including a URL and the number of times those postings are "liked" in their platform. Second, we consider the number of followers and followees of posting users in the social graph. Third, we consider a proxy for international news coverage, which we operationalize as the number of different countries (declared by users themselves) from which users posted about an article.

Additionally, we assume that a level of attention that is sustained can be translated to a larger exposure and may indicate long-standing interest on a topic. Hence, we consider the temporal coverage i.e., the length of the time window during which postings in social media are observed. To exclude outliers, we compute this period for 90% of the postings, i.e., the article's "shelf life" [9].

5.3.2 Social Media Stance. We consider the stance or positioning of social media postings with respect to the article they link to, as well as the stance of the responses (replies) to those postings. According to what we observe in this corpus, repliers sometimes ask for (additional) sources, express doubts about the quality of an article, and in some cases post links to fact-checking portals that contradict the claims of the article. These repliers are, indeed, acting as "social media fact-checkers," as the example in Figure 5 shows. Following



Figure 5: Example in which the stance of social media replies (bottom row) indicates the poor quality of an article promoted through a series of postings (top row).

a classification used for analyzing ideological debates [28], we consider four possible stances: supporting, commenting, contradicting, and questioning.

Retrieving replies. Twitter's API does not provide a programmatic method to retrieve all the replies to a tweet. Thus, we use a web scraper that retrieves the text of the replies of a tweet from the page in which each tweet is shown on the web. The design of this web scraper is straightforward and allows us to retrieve all the *first-level* replies of a tweet.

Classifying replies. To train our stance classifier, we use: (i) a general purpose dataset provided in the context of *SemEval 2016* [43], and (ii) a set of 300 tweets from our corpus which were annotated by crowdsourcing workers. From the first dataset we discard tweets that are not relevant to our corpus (e.g., debates on *Atheism*), thus we keep only debates on *Abortion* and *Climate Change*. The second set of annotated tweets is divided into 97 contradicting, 42 questioning, 80 commenting and 71 supporting tweets. We also have 10 tweets that were marked as "not-related" by the annotators and thus we exclude them from our training process. The combined dataset contains 1,140 annotated tweets. The **learning scheme** we use is a Random Forest classifier based on **features** including the number of: (i) total words, (ii) positive/negative words (using the Opinion Lexicon [31]), (iii) negation words, (iv) URLs, and (v) question/exclamation marks. We also computed the similarity between the replies and the tweet being replied to (using cosine similarity on *GloVe* vectors [49]), and the sentiment of the reply and the original tweet [39]. Details regarding the evaluation are provided in § 6.1.3.

6 EXPERIMENTAL EVALUATION

We begin the experimental evaluation by studying the performance of the methods we have described to extract quality indicators (§6.1). Then, we evaluate if these indicators correlate with scientific news quality. First, we determine if publications that have a good (bad) reputation or track record of rigor in scientific news reporting have higher (lower) scores according to our indicators (§6.2). Second, we use labels from experts (§6.3) to compare quality evaluations done by non-experts with and without access to our indicators (§6.4).

6.1 Evaluation of Indicator Extraction Methods

6.1.1 Quote Extraction and Attribution. The evaluation of our quote extraction and attribution method (§5.1.2) is based on a manually-annotated sample of articles from our corpus. A native English speaker performed an annotation finding 104 quotes (37 quotes attributed to persons, 33 scientific mentions and 34 "weasel" or unattributed quotes) in a random sample of 20 articles.

Table 2: Summary of all the quality indicators provided by the framework SciLens.

Context	Type	Indicator
Article	Baseline	Title [Clickbait, Subjectivity, Polarity], Article Readability, Article Word Count, Article Bylined
	Quote-Based	#Total Quotes, #Person Quotes, #Scientific Mentions, #Weasel Quotes
Sci. literature	Source Adherence	Semantic Textual Similarity
	Diffusion Graph	Personalized PageRank, Betweenness, [In, Out] Degree, Alexa Rank
Social media	Reach	#Likes, #Retweets, #Replies, #Followers, #Followees, [International News, Temporal] Coverage
	Stance	Tweets/Replies [Stance, Subjectivity, Polarity]

We compare three algorithms: (i) a baseline approach based on regular expressions searching for content enclosed in quote marks, which is usually the baseline for this type of task; (ii) our quote extraction method without the quote attribution phase, and (iii) the quote extraction and attribution method, where we consider a quote as correctly extracted if there is no ambiguity regarding the quotee (e.g., if the quotee is fully identified in the article but the attribution finds only the last name, we count it as incorrect).

As we observed, although the baseline approach has the optimal precision, it is unable to deal with cases where quotes are not within quote marks, which are the majority (**100%** precision, **8.3%** recall). Thus, our approach, without the quote attribution phase, improves significantly in terms of recall (**81.8%** precision, **45.0%** recall). Remarkably, the heuristics we use for quote attribution work well in practice and serve to increase both precision and recall (**90.9%** precision, **50.0%** recall). The resulting performance is comparable to state-of-the-art approaches in other domains (e.g., Pavllo et al. [48] obtain **90%** precision, **40%** recall).

6.1.2 Source Adherence. We use the supervised learning method described on §5.2.1 to measure Semantic Text Similarity (STS). We test three different learning models: Support Vector Machine, Random Forests and Neural Networks. The three classifiers use similarities computed at the document, sentence, and paragraph level, and combining all features from the three levels. Overall, the best accuracy (**93.5%**) was achieved by using a Random Forests classifier and all the features from the three levels of granularity, combined.

6.1.3 Social Media Stance. We evaluate the stance classifier described in §5.3.2 by performing 5-fold cross validation over our dataset. When we consider all four possible categories for the stance (supporting, commenting, contradicting and questioning), the accuracy of the classifier is **59.42%**. This is mainly due to confusion between postings expressing a mild support for the article and postings just commenting on the article, which also tend to elicit disagreement between annotators. Hence, we merge these categories into a "supporting or commenting" category comprising postings that do not express doubts about an article. Similarly, we consider "contradicting or questioning" as a category of postings expressing doubts about an article; previous work has observed that indeed false information in social media tends to be questioned more often (e.g., [10]). The problem is then reduced to binary classification.

To aggregate the stance of different postings that may refer to the same article, we compute their weighed average stance considering supporting or commenting as +1 (a "positive" stance) and contradicting or questioning as −1 (a "negative" stance). As weights we consider the popularity indicators of the postings (i.e., the number of likes and retweets). This is essentially a text quantification task [24], and the usage of a classification approach for a quantification task is justified because our classifier has nearly identical pairs of true positive and true negative rates (**80.65%** and **80.49%** respectively), and false positive and false negative rates (**19.51%** and **19.35%** respectively).

6.2 Correlation of Indicators among Portals of Diverse Reputability

We use two lists that classify news portals into different categories by reputability. The first list, by the American Council on Science and Health [3] comprises 50 websites sorted along two axes: whether they produce evidence-based or ideologically-based reporting, and whether their science content is compelling. The second list, by Climate Feedback [17], comprises 20 websites hosting 25 highly-shared stories on climate change, categorized into five groups by scientific credibility, from very high to very low.

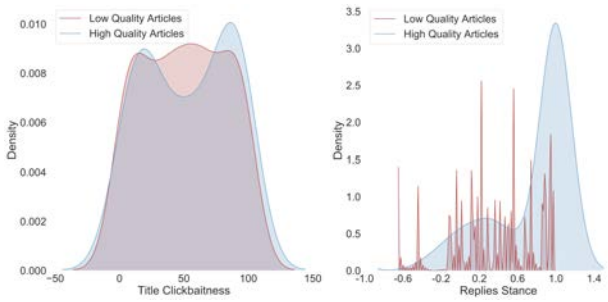
We sample a few sources according to reputability scores among the sources given consistent scores in both lists: high reputability (The Atlantic), medium reputability (New York Times), and low reputability (The Daily Mail). Next, we compare all of our indicators in the sets of articles in our collection belonging to these sources. Two example features are compared in Figure 6. We perform ANOVA [18] tests to select discriminating features. The results are shown on Table 3. Traffic rankings by Alexa.com, scientific mentions, and quotes, are among some of the most discriminating features.

6.3 Expert Evaluation

We ask a set of four external experts to evaluate the quality of a set of articles. Two of them evaluated a random sample of 20 articles about the gene editing technique CRISPR, which is a specialized topic being discussed relatively recently in mass media. The other two experts evaluated a random sample of 20 articles on the effects of Alcohol, Tobacco, and Caffeine (the "ATC" set in the following), which are frequently discussed in science news.

Experts were shown a set of guidelines for article quality based on previous work (§2). Then, they read each article and gave it a score in a 5-point scale, from very low quality to very high quality. Each expert annotated the 20 articles independently, and was given afterwards a chance to cross-check the ratings by the other expert and revise her/his own ratings if deemed appropriate.

The agreement between experts is distributed as follows: (i) *strong agreement*, when the rates after cross-checking are the same (7/20 in ATC, 6/20 in CRISPR); (ii) *weak agreement*, when the rates differ



(a) (b)

Figure 6: Kernel Density Estimation (KDE) of a traditional quality indicator (*Title Clickbaitness* on the left) and our proposal quality indicator (*Replies Stance* on the right). We observe that for both high and low quality articles the distribution of *Title Clickbaitness* is similar, thus the indicator is non-informative. However, most of the high quality articles have *Replies Stance* close to 1.0 which represents the *Supporting/Commenting* class of replies, whereas low quality articles span a wider spectrum of values and often have smaller or negative values representing the *Contradicting/Questioning* class of replies. Best seen in color.

by one point (12/20 in ATC, 10/20 in CRISPR), and (iii) *disagreement*, when the rates differ by two or more points (1/20 in ATC, 4/20 in CRISPR). Annotation results are show on Figure 7, and compared to non-expert evaluations, which are described next.

6.4 Expert vs Non-Expert Evaluation

We perform a comparison of quality evaluations by experts and non-experts. Non-experts are workers in a crowdsourcing platform. We ask for five non-expert labels per article, and employ what our crowdsourcing provider, *Figure Eight* (figure-eight.com), calls tier-3 workers, which are the most experienced and accurate. As a further quality assurance method, we use the agreement among workers to disregard annotators producing consistently annotations that are significantly different from the rest of the crowd. This is done at the worker level, and as a result we remove on average about one outlier judgment per article.

We consider two experimental conditions. On the first condition (**non-expert without indicators**), non-experts are shown the exact same evaluation interface as experts. On the second condition (**non-expert with indicators**), non-experts are shown 7 of the quality indicators we produced, which are selected according to Table 3. Each indicator (except the last two) is shown with stars, with ★☆☆☆☆ indicating that the article is in the lowest quintile according to that metric, and ★★★★★ indicating the article is in the highest quintile. The following legend is provided to non-experts to interpret the indicators:

> **Visitors per day of this news website** (more visitors = more stars)
> **Mentions of universities and scientific portals** (more mentions = more stars)
> **Length of the article** (longer article = more stars)
> **Number of quotes in the article** (more quotes = more stars)
> **Number of replies to tweets about this article)** (more replies = more stars)
> **Article signed by its author** (✓= signed,✗= not signed)
> **Sentiment of the article's title** (☺☺= most positive, ☹☹= most negative)

Results of comparing the evaluation of experts and non-experts in the two conditions we have described are summarized in Figure 7. In the figure, the 20 articles in each set are sorted by increasing expert rating; assessments by non-experts differ from expert ratings, but this difference tends to be reduced when non-experts have access to quality indicators.

In Table 4 we show how displaying indicators leads to a decrease in these differences, meaning that non-expert evaluations become closer to the average evaluation of experts, particularly when experts agree. In the ATC set the improvement is small, but in CRISPR

Table 3: Top five discriminating indicators for articles sampled from pairs of outlets having different levels of reputability (p-value: < 0.005 *, < 0.01 **, < 0.05 *).**

The Atlantic vs. Daily Mail (very high vs. very low)	NY Times vs. Daily Mail (medium vs. very low)
Alexa Rank***	Alexa Rank***
#Scientific Mentions***	Article Bylined***
Article Readability**	#Scientific Mentions***
#Total Quotes*	Article Readability***
Title Polarity	#Total Quotes**
The Atlantic vs. NY Times (very high vs. medium)	**All Outlets (from very high to very low)**
Alexa Rank***	Alexa Rank***
Article Bylined***	Article Bylined***
Article Word Count*	Article Word Count***
#Replies*	#Scientific Mentions***
#Followers	Article Readability***

Table 4: Differences among expert evaluations, evaluations provided by non-experts and fully automatic evaluations provided by the SciLens framework, measured using RMSE (lower is better). ATC and CRISPR are two sets of 20 articles each. Strong agreement indicates cases where experts fully agree, weak agreement when they differed by one point, and disagreement when they differed by two or more points. No-Ind. is the first experimental condition for non-experts, in which no indicators are shown. Ind. is the second experimental condition, in which indicators are shown.

	Experts by agreement	#	Non-Experts No ind.	Non-Experts Ind.	Fully automated
ATC	Strong agreement	7	0.80	**0.45**	1.41
	Weak agreement	12	1.28	1.18	**0.76**
	Disagreement	1	0.40	1.30	**0.00**
	All articles	20	1.10	**1.00**	**1.00**
CRISPR	Strong agreement	6	1.40	1.17	**1.00**
	Weak agreement	10	0.86	0.76	**0.67**
	Disagreement	4	**0.96**	1.22	1.03
	All articles	20	1.96	0.96	**0.85**



(a) Articles on Alcohol, Tobacco, and Caffeine

(b) Articles on CRISPR

Figure 7: Evaluation of two sets of 20 scientific articles. The line corresponds to expert evaluation, while the bars indicate fully automatic evaluation (red), assisted evaluation by non-experts (light blue), and manual evaluation by non-experts (dark blue). Best seen in color.

it is large, bringing non-expert scores about 1 point (out of 5) closer to expert scores.

Table 4 and Figure 7 also includes a fully automated quality evaluation, built using a weakly supervised classifier over all the features we extracted. As weak supervision, we used the lists of sites in different tiers of reputability (§6.2) and considered that *all articles* on each site had the same quality score as the reputation of the site. Then, we used this classifier to annotate the 20 articles in each of the two sets. Results show that this achieves the **lowest** error with respect to expert annotations.

7 CONCLUSIONS

We have described a method for evaluating the quality of scientific news articles. This method, SciLens, requires to collect news articles, papers referenced in them, and social media postings referencing them. We have introduced new quality indicators that consider quotes in the articles, the similarity and relationship of articles with the scientific literature, and the volume and stance of social media reactions. The approach is general and can be applied to any specialized domain where there are primary sources in technical language that are "translated" by journalists and bloggers into accessible language.

In the course of this work, we developed several quality indicators that can be computed automatically, and demonstrated their suitability for this task through multiple experiments. First, we showed several of them are applicable at the site level, to distinguish among different tiers of quality with respect to scientific news. Second, we showed that they can be used by non-experts to improve their evaluations of quality of scientific articles, bringing them more in line with expert evaluations. Third, we showed how these indicators can be combined to produce fully automated scores using weak supervision, namely data annotated at the site level.

Limitations. Our methodology requires access to the content of scientific papers and social media postings. Regarding the latter, given the limitations of the data scrapers we have used only replies to postings and not replies-to-replies. We have also used a single data source for social media postings. Furthermore, we consider a broad definition of "news" to build our corpus, covering mainstream media as well as other sites, including fringe publications. Finally, our methodology is currently applicable only on English corpora.

Reproducibility. Our code uses the following `Python` libraries: `Pandas` and `Spark` for data management, `NetworkX` for graph processing, `scikit-learn` and `PyTorch` for ML, and `SpaCy`, `Beautiful Soup`, `Newspaper`, `TextSTAT` and `TextBlob` for NLP. All the data, code as well as the expert and crowd annotations used in this paper are available for research purposes in ***http://scilens.epfl.ch***.

ACKNOWLEDGMENTS

We would like to thank the external experts who helped us on the evaluation of our framework: **Aina Crosas** (Communication

Officer at the Barcelona Biomedical Research Park), **Andreu Prados** (Pharmacist and Registered Dietitian Nutritionist specialized in Health Communication), **Jose F. Rizo** (Biologist and President of the Fundación Ñamku), and **Dimitra Synefiaridou** (PhD Student in Microbiology at the Veening Lab). This work is partially supported by the Open Science Fund of EPFL (http://www.epfl.ch/research/initiatives/open-science-fund) and the La Caixa project (LCF/PR/PR16/11110009).

REFERENCES

[1] Euan A. Adie and William Roe. 2013. Altmetric: enriching scholarly content with article-level discussion and metrics. *Learned Publishing* 26, 1 (2013), 11–17. https://doi.org/10.1087/20130103

[2] Deepak Agarwal, Bee-Chung Chen, and Xuanhui Wang. 2012. Multi-faceted ranking of news articles using post-read actions. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*, Xue-wen Chen, Guy Lebanon, Haixun Wang, and Mohammed J. Zaki (Eds.). ACM, 694–703. https://doi.org/10.1145/2396761.2396850

[3] Alex Berezow. March 5, 2017. Infographic: The Best and Worst Science News Sites. *American Council on Science and Health* (March 5, 2017). https://acsh.org/news/2017/03/05/infographic-best-and-worst-science-news-sites-10948

[4] Franziska Badenschier and Holger Wormer. 2012. Issue selection in science journalism: Towards a special theory of news values for science news? In *The sciences' media connection–public communication and its repercussions*. Springer, 59–85.

[5] Ramy Baly, Georgi Karadzhov, Dimitar Alexandrov, James R. Glass, and Preslav Nakov. 2018. Predicting Factuality of Reporting and Bias of News Media Sources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, 3528–3539. https://aclanthology.info/papers/D18-1389/d18-1389

[6] Martin W Bauer, Nick Allum, and Steve Miller. 2007. What can we learn from 25 years of PUS survey research? Liberating and expanding the agenda. *Public understanding of science* 16, 1 (2007), 79–95.

[7] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3 (2003), 993–1022. http://www.jmlr.org/papers/v3/blei03a.html

[8] Christina Boididou, Symeon Papadopoulos, Lazaros Apostolidis, and Yiannis Kompatsiaris. 2017. Learning to Detect Misleading Content on Twitter. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval, ICMR 2017, Bucharest, Romania, June 6-9, 2017*, Bogdan Ionescu, Nicu Sebe, Jiashi Feng, Martha Larson, Rainer Lienhart, and Cees Snoek (Eds.). ACM, 278–286. https://doi.org/10.1145/3078971.3078979

[9] Carlos Castillo, Mohammed El-Haddad, Jürgen Pfeffer, and Matt Stempeck. 2014. Characterizing the life cycle of online news stories using social media reactions. In *Computer Supported Cooperative Work, CSCW '14, Baltimore, MD, USA, February 15-19, 2014*, Susan R. Fussell, Wayne G. Lutters, Meredith Ringel Morris, and Madhu Reddy (Eds.). ACM, 211–223. https://doi.org/10.1145/2531602.2531623

[10] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2013. Predicting information credibility in time-sensitive social media. *Internet Research* 23, 5 (2013), 560–588. https://doi.org/10.1108/IntR-05-2012-0095

[11] Giovanni Luca Ciampaglia, Prashant Shiralkar, Luis M. Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. 2015. Computational Fact Checking from Knowledge Networks. *PLOS ONE* 10, 6 (jun 2015), e0128193. https://doi.org/10.1371/journal.pone.0128193

[12] Peter Conrad. 1999. Uses of expertise: Sources, quotes, and voice in the reporting of genetics in the news. *Public Uniderstanding of Science* 8 (1999).

[13] Vladimir De Semir. 2000. Scientific journalism: problems and perspectives. *International Microbiology* 3, 2 (2000), 125–128.

[14] Estelle Dumas-Mallet, Andy Smith, Thomas Boraud, and François Gonon. 2017. Poor replication validity of biomedical association studies reported by newspapers. *PLOS ONE* 12, 2 (02 2017), 1–15. https://doi.org/10.1371/journal.pone.0172650

[15] Sharon Dunwoody. 2014. Science journalism: prospects in the digital age. In *Routledge handbook of public communication of science and technology*. Routledge, 43–55.

[16] David K. Elson and Kathleen R. McKeown. 2010. Automatic Attribution of Quoted Speech in Literary Narrative. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*, Maria Fox and David Poole (Eds.). AAAI Press. http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1945

[17] Emmanuel M. Vincent. 17 Jan 2018. Most popular climate change stories of 2017 reviewed by scientists. *Climate Feedback* (17 Jan 2018). https://climatefeedback.org/most-popular-climate-change-stories-2017-reviewed-scientists/

[18] Ronald Aylmer Fisher. 2006. *Statistical methods for research workers*. Genesis Publishing Pvt Ltd.

[19] Rudolph Flesch. 1948. A new readability yardstick. *Journal of applied psychology* 32, 3 (1948), 221.

[20] B. J. Fogg and Hsiang Tseng. 1999. The Elements of Computer Credibility. In *Proceeding of the CHI '99 Conference on Human Factors in Computing Systems: The CHI is the Limit, Pittsburgh, PA, USA, May 15-20, 1999.*, Marian G. Williams and Mark W. Altom (Eds.). ACM, 80–87. https://doi.org/10.1145/302979.303001

[21] Jim Foust. 2017. *Online journalism: principles and practices of news for the Web*. Taylor & Francis.

[22] Linton C Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* (1977), 35–41.

[23] Linton C Freeman. 1978. Centrality in social networks conceptual clarification. *Social networks* 1, 3 (1978), 215–239.

[24] Wei Gao and Fabrizio Sebastiani. 2016. From classification to quantification in tweet sentiment analysis. *Social Netw. Analys. Mining* 6, 1 (2016), 19:1–19:22. https://doi.org/10.1007/s13278-016-0327-z

[25] Alan G. Gross. 1994. The roles of rhetoric in the public understanding of science. *Public Understanding of Science* 3, 1 (1994), 3–23. https://doi.org/10.1088/0963-6625/3/1/001 arXiv:https://doi.org/10.1088/0963-6625/3/1/001

[26] Lushan Han, Justin Martineau, Doreen Cheng, and Christopher Thomas. 2015. Samsung: Align-and-Differentiate Approach to Semantic Textual Similarity. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, Daniel M. Cer, David Jurgens, Preslav Nakov, and Torsten Zesch (Eds.). The Association for Computer Linguistics, 172–177. http://aclweb.org/anthology/S/S15/S15-2031.pdf

[27] P Sol Hart, Erik C Nisbet, and Teresa A Myers. 2015. Public attention to science and political news and support for climate change mitigation. *Nature Climate Change* 5, 6 (2015), 541.

[28] Kazi Saidul Hasan and Vincent Ng. 2013. Stance Classification of Ideological Debates: Data, Models, Features, and Constraints. In *Sixth International Joint Conference on Natural Language Processing, IJCNLP 2013, Nagoya, Japan, October 14-18, 2013*. Asian Federation of Natural Language Processing / ACL, 1348–1356. http://aclweb.org/anthology/I/I13/I13-1191.pdf

[29] Taher H. Haveliwala. 2003. Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search. *IEEE Trans. Knowl. Data Eng.* 15, 4 (2003), 784–796. https://doi.org/10.1109/TKDE.2003.1208999

[30] Ernst Hellinger. 1909. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik* 136 (1909), 210–271.

[31] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, Won Kim, Ron Kohavi, Johannes Gehrke, and William DuMouchel (Eds.). ACM, 168–177. https://doi.org/10.1145/1014052.1014073

[32] Jakob D Jensen. 2008. Scientific uncertainty in news coverage of cancer research: Effects of hedging on scientists' and journalists' credibility. *Human communication research* 34, 3 (2008), 347–369.

[33] Wei Jin, Hung Hay Ho, and Rohini K. Srihari. 2009. OpinionMiner: a novel machine learning system for web opinion mining and extraction. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*, John F. Elder IV, Françoise Fogelman-Soulié, Peter A. Flach, and Mohammed Javeed Zaki (Eds.). ACM, 1195–1204. https://doi.org/10.1145/1557019.1557148

[34] Dan Jurafsky, Angel X. Chang, Grace Muzny, and Michael Fang. 2017. A Two-stage Sieve Approach for Quote Attribution. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, Mirella Lapata, Phil Blunsom, and Alexander Koller (Eds.). Association for Computational Linguistics, 460–470. https://aclanthology.info/papers/E17-1044/e17-1044

[35] Anand Konjengbam, Subrata Ghosh, Nagendra Kumar, and Manish Singh. 2018. Debate Stance Classification Using Word Embeddings. In *Big Data Analytics and Knowledge Discovery - 20th International Conference, DaWaK 2018, Regensburg, Germany, September 3-6, 2018, Proceedings (Lecture Notes in Computer Science)*, Carlos Ordonez and Ladjel Bellatreche (Eds.), Vol. 11031. Springer, 382–395. https://doi.org/10.1007/978-3-319-98539-8_29

[36] Justin Kosslyn and Cong Yu. April 7, 2017. Fact Check now available in Google Search and News around the world. *Google* (April 7, 2017). http://www.blog.google/products/search/fact-check-now-available-google-search-and-news-around-world

[37] Srijan Kumar, Robert West, and Jure Leskovec. 2016. Disinformation on the Web: Impact, Characteristics, and Detection of Wikipedia Hoaxes. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, Jacqueline Bourdeau, Jim Hendler, Roger Nkambou, Ian Horrocks, and Ben Y. Zhao (Eds.). ACM, 591–602. https://doi.org/10.1145/2872427.2883085

[38] Matthias Liebeck, Philipp Pollack, Pashutan Modaresi, and Stefan Conrad. 2016. HHU at SemEval-2016 Task 1: Multiple Approaches to Measuring Semantic

Textual Similarity. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*, Steven Bethard, Daniel M. Cer, Marine Carpuat, David Jurgens, Preslav Nakov, and Torsten Zesch (Eds.). The Association for Computer Linguistics, 595–601. http://aclweb.org/anthology/S/S16/S16-1090.pdf

[39] Steven Loria. 2018. Sentiment Analysis. (2018). http://textblob.readthedocs.io

[40] Tessa Lyons. May 23, 2018. Hard Questions: What's Facebook's Strategy for Stopping False News? *Facebook* (May 23, 2018). http://newsroom.fb.com/news/2018/05/hard-questions-false-news

[41] Saurabh Mathur. 2017. Clickbait Detector. (2017). http://github.com/saurabhmathur96/clickbait-detector

[42] George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38, 11 (1995), 39–41. https://doi.org/10.1145/219717.219748

[43] Saif M. Mohammad, Parinaz Sobhani, and Svetlana Kiritchenko. 2017. Stance and Sentiment in Tweets. *ACM Trans. Internet Techn.* 17, 3 (2017), 26:1–26:23. https://doi.org/10.1145/3003433

[44] Greg Myers. 2003. Discourse studies of scientific popularization: Questioning the boundaries. *Discourse studies* 5, 2 (2003), 265–279.

[45] Timothy O'Keefe, Silvia Pareti, James R. Curran, Irena Koprinska, and Matthew Honnibal. 2012. A Sequence Labelling Approach to Quote Attribution. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, Jun'ichi Tsujii, James Henderson, and Marius Pasca (Eds.). ACL, 790–799. http://www.aclweb.org/anthology/D12-1072

[46] Chiara Palmerini. 2007. Science reporting as negotiation. In *Journalism, Science and Society*. Chapter 11, 113–122.

[47] Silvia Pareti, Timothy O'Keefe, Ioannis Konstas, James R. Curran, and Irena Koprinska. 2013. Automatically Detecting and Attributing Indirect Quotations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, 989–999. http://aclweb.org/anthology/D/D13/D13-1101.pdf

[48] Dario Pavllo, Tiziano Piccardi, and Robert West. 2018. Quootstrap: Scalable Unsupervised Extraction of Quotation-Speaker Pairs from Large News Corpora via Bootstrapping. In *Proceedings of the Twelfth International Conference on Web and Social Media, ICWSM 2018, Stanford, California, USA, June 25-28, 2018*. AAAI Press, 231–240. https://aaai.org/ocs/index.php/ICWSM/ICWSM18/paper/view/17827

[49] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). ACL, 1532–1543. http://aclweb.org/anthology/D/D14/D14-1162.pdf

[50] Kashyap Popat, Subhabrata Mukherjee, Jannik Strötgen, and Gerhard Weikum. 2017. Where the Truth Lies: Explaining the Credibility of Emerging Claims on the Web and Social Media. In *Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, April 3-7, 2017*, Rick Barrett, Rick Cummings, Eugene Agichtein, and Evgeniy Gabrilovich (Eds.). ACM, 1003–1012. https://doi.org/10.1145/3041021.3055133

[51] Bruno Pouliquen, Ralf Steinberger, and Clive Best. 2007. Automatic detection of quotations in multilingual news. In *Proceedings of Recent Advances in Natural Language Processing*. 487–492.

[52] Andrew Salway, Paul Meurer, Knut Hofland, and Øystein Reigem. 2017. Quote Extraction and Attribution from Norwegian Newspapers. In *Proceedings of the 21st Nordic Conference on Computational Linguistics, NODALIDA 2017, Gothenburg, Sweden, May 22-24, 2017*, Jörg Tiedemann and Nina Tahmasebi (Eds.). Association for Computational Linguistics, 293–297. https://aclanthology.info/papers/W17-0241/w17-0241

[53] Laura Sbaffi and Jennifer Rowley. 2017. Trust and Credibility in Web-Based Health Information: A Review and Agenda for Future Research. *Journal of medical Internet research* 19, 6 (jun 2017), e218. https://doi.org/10.2196/jmir.7579

[54] Chengcheng Shao, Giovanni Luca Ciampaglia, Alessandro Flammini, and Filippo Menczer. 2016. Hoaxy: A Platform for Tracking Online Misinformation. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11-15, 2016, Companion Volume*, Jacqueline Bourdeau, Jim Hendler, Roger Nkambou, Ian Horrocks, and Ben Y. Zhao (Eds.). ACM, 745–750. https://doi.org/10.1145/2872518.2890098

[55] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake News Detection on Social Media: A Data Mining Perspective. *SIGKDD Explorations* 19, 1 (2017), 22–36. https://doi.org/10.1145/3137597.3137600

[56] Marcella Tambuscio, Giancarlo Ruffo, Alessandro Flammini, and Filippo Menczer. 2015. Fact-checking Effect on Viral Hoaxes: A Model of Misinformation Spread in Social Networks. In *Proceedings of the 24th International Conference on World Wide Web Companion, WWW 2015, Florence, Italy, May 18-22, 2015 - Companion Volume*, Aldo Gangemi, Stefano Leonardi, and Alessandro Panconesi (Eds.). ACM, 977–982. https://doi.org/10.1145/2740908.2742572

[57] Joseph W. Taylor, Marie Long, Elizabeth Ashley, Alex Denning, Beatrice Gout, Kayleigh Hansen, Thomas Huws, Leifa Jennings, Sinead Quinn, Patrick Sarkies, Alex Wojtowicz, and Philip M. Newton. 2015. When Medical News Comes from Press ReleasesâĂŤA Case Study of Pancreatic Cancer and Processed Meat. *PLOS ONE* 10, 6 (06 2015), 1–13. https://doi.org/10.1371/journal.pone.0127848

[58] Juliane Urban and Wolfgang Schweiger. 2014. News Quality from the Recipients' Perspective. *Journalism Studies* 15, 6 (2014), 821–840. https://doi.org/10.1080/1461670X.2013.856670 arXiv:https://doi.org/10.1080/1461670X.2013.856670

[59] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science* 359, 6380 (2018), 1146–1151. https://doi.org/10.1126/science.aap9559 arXiv:http://science.sciencemag.org/content/359/6380/1146.full.pdf

[60] Wei Wei and Xiaojun Wan. 2017. Learning to Identify Ambiguous and Misleading News Headlines. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, Carles Sierra (Ed.). ijcai.org, 4172–4178. https://doi.org/10.24963/ijcai.2017/583

[61] Emma Weitkamp. 2003. British newspapers privilege health and medicine topics over other science news. *Public Relations Review* 29, 3 (2003), 321–333.

[62] Amy X. Zhang, Aditya Ranganathan, Sarah Emlen Metz, Scott Appling, Connie Moon Sehat, Norman Gilmore, Nick B. Adams, Emmanuel Vincent, Jennifer Lee, Martin Robbins, Ed Bice, Sandro Hawke, David R. Karger, and An Xiao Mina. 2018. A Structured Response to Misinformation: Defining and Annotating Credibility Indicators in News Articles. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, Pierre-Antoine Champin, Fabien L. Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis (Eds.). ACM, 603–612. https://doi.org/10.1145/3184558.3188731

[63] Arkaitz Zubiaga, Elena Kochkina, Maria Liakata, Rob Procter, Michal Lukasik, Kalina Bontcheva, Trevor Cohn, and Isabelle Augenstein. 2018. Discourse-aware rumour stance classification in social media using sequential classifiers. *Inf. Process. Manage.* 54, 2 (2018), 273–290. https://doi.org/10.1016/j.ipm.2017.11.009

Using Variability as a Guiding Principle to Reduce Latency in Web Applications via OS Profiling

Amoghvarsha Suresh
PACE Lab, Stony Brook University
amsuresh@cs.stonybrook.edu

Anshul Gandhi
PACE Lab, Stony Brook University
anshul@cs.stonybrook.edu

ABSTRACT

Request latency is a critical metric in determining the usability of web services. The latency of a request includes service time – the time when the request is being actively serviced – and waiting time – the time when the request is waiting to be served. Most existing works aim to reduce request latency by focusing on reducing the mean service time (that is, shortening the critical path).

In this paper, we explore an alternative approach to reducing latency – *using variability as a guiding principle when designing web services*. By tracking the service time variability of the request as it traverses across software layers within the user and kernel space of the web server, we identify the most critical stages of request processing. We then determine control knobs in the OS and application, such as thread scheduling and request batching, that regulate the variability in these stages, and demonstrate that tuning these specific knobs can significantly improve end-to-end request latency. Our experimental results with Memcached and Apache web server under different request rates, including real-world traces, show that this alternative approach can reduce mean and tail latency by 30–50%.

ACM Reference Format:

Amoghvarsha Suresh and Anshul Gandhi. 2019. Using Variability as a Guiding Principle to Reduce Latency in Web Applications via OS Profiling. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313406

1 INTRODUCTION

Web applications provide important services, such as online retail, messaging, and search, to end-users on a daily basis [3, 5, 8, 13]. A critical metric for determining the usability of such web applications is the *latency* of user requests; different service providers employ different measures of latency, such as mean, median, 95%ile, and 99%ile [7, 37, 47]. A delay of even a few milliseconds in request latency can lead to significant revenue loss for service providers due to user abandonment [14, 44].

Conceptually, request latency can be broken down into two distinct components, *service time* and *waiting time* [27]. Service time is defined as the time during which the request is being actively serviced. Waiting time is then defined as the remaining time during which the request is waiting to be served. To maintain acceptable latencies, service providers often optimize their web servers to reduce the mean service time of requests, that is, shorten the critical path of request processing. For example, Chronos [24] uses user-level networking with NIC-level request dispatch to reduce lock contention and lower the latency of web applications; Jose et al. [53] make Memcached RDMA-capable to shorten the critical path; Li et al. [31] advocate using a real-time scheduler to reduce request scheduling delay.

An alternative approach to reducing web latencies that we explore in this paper is to minimize the *variability* in request processing. Queuing models show that, in addition to mean service time, the variability of the service time is also important when trying to reduce latency. Surprisingly, there has been very little work on actually reducing the variability in the system [7, 19, 49]. While much of the variability is intrinsic to the workload (e.g., burstiness in customer traffic), some of the variability is due to the *application and software design* (e.g., garbage collection, context switches, CPU scheduling policies, etc.), and can be regulated by making subtle changes to the system.

In this paper, we investigate the following system design question – "*is it worth reducing variability in request processing times at the potential expense of lengthening its critical path?*". While theoretical analysis suggests that this is indeed the case, especially for heavy-tailed distributions (see Section 2), evaluating this idea in practice for web applications is challenging for several reasons:

- Web applications have a *complex processing lifetime*, going through several paths of processing in the kernel, including the TCP stack, parsing of requests, and scheduling of the request on server cores. Identifying the most likely culprit(s) that contributes to service time variability will require low-overhead yet accurate and fine-grained *request tracing* in the user and kernel space.
- There are several control knobs within a web server that can be tuned to reduce service time variability, such as the OS scheduler, page allocation strategy, etc. Prior work has also shown that new application-specific control knobs can be dynamically generated [18]. Given the numerous choices, efficiently finding the right control knob to mitigate variability is challenging, especially since the choice of the optimal control knob may depend on the server and web application configuration. Worse, employing the wrong control knob can *hurt* request latency.
- Most control knobs within the system that can be tuned to reduce service time variability *invariably hurt mean service*

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313406

time. For example, prior work has shown that admission control can reduce service time variability by preventing server overload [21]. But this reduction comes at the expense of lengthening the critical path (since requests are held back), and possibly hurting latency. It is thus important to carefully tune the control knob to balance the trade-off between variability and mean service time.

We address the above challenges in the context of web services and demonstrate the benefits of *using service time variability as a guiding principle* to improve request latency. We employ lightweight, fine-grained request profiling to track service time variability; this allows us to identify components on the critical path of the request-response cycle that exacerbate service time variability. Using variability as our guiding principle, we then determine control knobs in the system and/or application that can be tuned to mitigate service time variability in the identified components. We evaluate our approach by investigating variability in Memcached, a popular in-memory (key-value store) caching service used to speed up web applications [12], and the Apache web server [45], a widely deployed http server application. Specifically, we explore the following use cases:

(1) *Request batching at the Memcached client*: To reduce the network overhead of sending short packets, the Linux kernel at the client batches incoming requests until it gets a response back from the server saying it has received the previous batch (Nagle's algorithm [40]). However, since network conditions can be variable, this default batching behavior leads to bursts of processing at the server, followed by idle times, resulting in significant service time variability. We modify this default batching behavior to closely regulate the time between batches, lowering the variability (by as much as 76%) and improving tail latency by up to 40%.
(2) *Redesigning the LRU management on the Memcached server*: To maintain the Least-Recently-Used (LRU) ordering of items in the cache, the Memcached server needs to constantly reorder items. To shorten the critical path, Memcached moves the LRU maintenance off the request processing path and delegates this responsibility to an LRU management thread that is run periodically. When this thread is active, it interferes with the Memcached request processing, resulting in high service time variability. We redesign the LRU management functionality and include it on the critical path of Memcached request processing in the form of fine-grained slices of LRU work. This counter-intuitive redesign reduces variability and improves latency by about 30%.
(3) *Application thread pinning for the Apache web server*: The Apache web server schedules its various worker threads and processes on any available idle CPU core opportunistically to start serving customer requests as soon as possible. Consequently, threads may move between cores, resulting in context switch overheads and state migration. We explore thread pinning to reduce this overhead and associated variability, though at the expense of possibly delaying request processing when the pinned core is busy. We show that, at high load, this approach can reduce latency by 50%.

We experimentally evaluate the performance improvements for the above three use cases under various workload scenarios, including different levels of load, different inter-arrival time distributions, and different (time-varying) arrival traces. While existing approaches, such as exclusively focusing on reducing mean service time or employing ad-hoc control knobs, can end up *hurting* request latency, we use our alternative approach of focusing on service time variability to substantially reduce mean and tail latency (by up to 30–50%) in all three cases. Importantly, we do so by simply changing the metric that we use to identify the critical stage of request processing and determine the appropriate control knob.

2 MOTIVATION AND SCOPE OF OUR WORK

To motivate our approach of focusing on service time variability, we leverage queueing theory to analyze request latency for a server as a function of variability. Although models are only approximations of today's complex applications, the resulting analysis is instructive [16, 33, 36, 48], and guides our system design in later sections.

2.1 Request latency versus variability

Recent studies at Bing [15, 19], Google [23, 24, 43], and Facebook [3], suggest that modern web applications often experience high variability in inter-arrival time (IAT) and service time (ST). Variability in IAT represents workload variability, such as bursty arrivals. ST variability represents variability in processing times due to differences in work requirements (e.g., reads vs. writes) or differences in the request path (e.g., due to batching or TLB misses), or due to misconfigurations at the server that lead to anomalous behavior or "jitters" [31]. Note that ST is the amount of service required by a request to complete processing; alternatively, ST is the minimum possible request latency, assuming no delays.

To investigate the impact of variability on request latency, we consider a web server with a given inter-arrival time (IAT) distribution and a given service time (ST) distribution. To parameterize variability, we use the squared coefficient of variation (C^2), defined as the ratio of variance and square of the mean. Then, we have:

$$C^2_{IAT} = Var(IAT)/E^2[IAT], \text{ and} \quad (1)$$

$$C^2_{ST} = Var(ST)/E^2[ST], \quad (2)$$

where $E[]$ is the mean and $Var()$ is the variance of the random variable. To examine the full range of variability ($[0, \infty)$), we consider the following distributions:

- D (Deterministic), with $C^2 = 0$, is the ideal case of no variability.
- M (Exponential), with $C^2 = 1$, represents nominal variability.
- H_2 (Hyper-exponential), with $C^2 > 1$ (customizable), represents the case of high variability.

The $M/G/1$ queueing model, with Exponential IAT and generic ST distributions, allows us to analyze mean request latency, $E[T]$, as a function of ST variability, $Var(ST)$, and mean ST, $E[ST]$, via the Pollaczek-Khinchin (P-K) formula [27]:

$$E[T] = Var(ST) \cdot \frac{\lambda}{2(1-\rho)} + E[ST] \cdot \frac{2-\rho}{2(1-\rho)}, \quad (3)$$

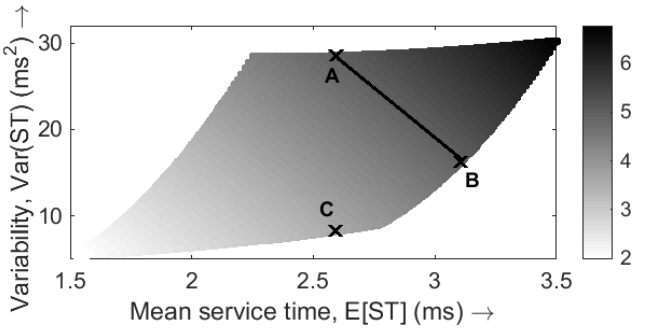


Figure 1: Heatmap of mean latency as a function of mean service time ($E[ST]$) and variability in service time ($Var(ST)$). The black diagonal line denotes the equi-latency line of $5ms$.

where $\lambda = 1/E[IAT]$ is the mean request arrival rate and $\rho = \lambda \cdot E[ST]$ is the normalized system load [27]. Clearly, both $E[T]$ *and* $Var(ST)$ impact request latency.

Consider the $M/G/1$ model where all parameters, including $E[ST]$, are fixed, and only $Var(ST)$ is varied. Let the workload have a mean ST, $E[ST]$, of 1 millisecond, and system load of, say, 60%; thus, request rate, $\lambda = 0.6 \cdot E[ST] = 600$ req/s. Using Eq. (3), we find that $E[T] = 1.7ms$ under Deterministic ST ($C^2_{ST} = 0$). For comparison, for a system with Deterministic IAT and ST, we get $E[T] = 1ms$. Thus, the added variability due to an Exponential IAT already increases mean latency by 70%. For the $M/G/1$ system, if we now consider Exponential ST ($C^2_{ST} = 1$), we get $E[T] = 2.5ms$, a 250% increase over the baseline. If we further increase the variability in ST to $C^2_{ST} = 2$ using a H_2 distribution, we get $E[T] = 3.3ms$, a 330% increase! The increase in request latency with variability is even more pronounced at higher loads.

While the above results highlight the importance of reducing variability in ST, we note that making changes to a system to reduce variability may lead to other performance overheads, resulting in higher $E[ST]$. For example, the OS scheduler may opportunistically move threads between cores to leverage idle cores. However, this introduces ST variability due to the associated state migration and context switches. Disabling this opportunistic behavior can mitigate variability, but may lead to scenarios where threads are waiting on a busy core even though other cores are idle. There is thus a trade-off between reducing $Var(ST)$ and increasing $E[ST]$, which is captured by Eq. (3).

Figure 1 illustrates the impact on mean request latency of the trade-off between $E[ST]$ and $Var(ST)$ for the $M/G/1$ model. We set $\lambda = 100$ req/s, and use an H_2 distributed ST whose variability can be controlled. To highlight the trade-off, we show the equi-latency line of $5ms$ in black. Point B on this line has a *higher* $E[ST]$ but *lower* $Var(ST)$ compared to point A; nonetheless, both have the same $5ms$ latency. Thus, we can afford some overhead in $E[ST]$ when reducing $Var(ST)$ of a system. Finally, point C has the same $E[ST]$ as A, but has much lower $Var(ST)$; as a result, the mean request latency for C is almost 30% lower than that for A.

Key takeaways: *(i) Reducing service time variability can significantly lower request latency (by 2× or higher), and (ii) It is beneficial to reduce variability even if doing so increases the mean service time.*

2.2 Objective and scope of this work

The design of today's web server systems is primarily driven by the intent to shorten the critical path of requests, that is, reducing the mean ST. However, our analytical results above suggest an alternative solution to improving request latencies in systems – *reducing variability*. Variability is often assumed to be intrinsic to the hardware and the workload, and thus not easily controllable. However, this is not always true. While variability in IAT depends on customer request behavior, variability in ST can be regulated to some extent by modifying the application and software stack. The objective of our work is to demonstrate that this alternative viewpoint of "focusing on ST variability for designing web servers" can reveal viable solutions to reduce latency. Note, however, that we are *not* arguing against solutions that focus on reducing ST.

3 SOLUTION OVERVIEW

Given a web application, our goal is to improve its latency by targeting a reduction in service time (ST) variability. However, web applications can be complex, consisting of several processes and threads that work asynchronously. Further, web requests typically pass through several software layers before completing service. Thus, we must first identify the potential software layers or components that significantly contribute to ST variability, and then determine control knobs that can be tuned to reduce ST variability in the identified software layers. Some control knobs are readily available, such as OS thread scheduler, TCP congestion control, etc. However, in some software layers, effective control knobs may not be available, necessitating modifications to existing software.

We use the following methodology to achieve our goal:

(1) *Fine-grained, unobtrusive request tracing*: We employ low-overhead tracing for the web requests. Our tracing encompasses all layers of the software stack that a request goes through during its processing lifetime, including the OS and network stack.
(2) *Identifying the source(s) of service time variability*: We aggregate the tracing information across all requests using low-overhead histograms to identify the software layers with the highest variability that are amenable to modification.
(3) *Modifying the system software to mitigate variability and reduce request latency*: We explore available control knobs in the system that can reduce the observed ST variability in the target software layer, even if this reduction comes at the expense of an increase in mean ST (see Section 2.1). Once the knob is determined, we investigate its optimal setting to minimize end-to-end request latency.

Our choice of "variability of service time" as the guiding principle in our methodology is important. We demonstrate, in Sections 6 – 8, that our methodology allows us to uncover the critical stages of request processing for two popular web applications and guides our selection of control knob. We show later, in Section 9, that naïvely

employing request tracing (for example, by using mean ST as the metric instead of variability) or selecting control knobs in an ad-hoc manner can actually *hurt* application performance.

4 REQUEST TRACING

Our request tracing works by timestamping the request through several layers starting from when it arrives at the host from the server's NIC until immediately before the application transfers the response packet back to the OS. Given our focus on variability in service times, we only trace the request at the host server, and not the client.

To store the timestamps, we append an empty 64-byte buffer to the original request packet, similar to prior works [31]. Then, as the request goes through different stages of processing on the client and the server, timestamps are recorded at appropriate offsets for the stage of processing by writing the system clock time into the buffer; we use the system clock with nanosecond precision to record timestamps within the server. By appending the small buffer to the request, we can record multiple timestamps and track the request to which they correspond without requiring any additional post-processing. The above tracing implementation required modification to the Linux kernel source, network drivers, and the application protocols to write timestamps into the appended buffer at the right offset. The overhead on request latency through all layers is low, about 5%; this overhead is incurred by both the baseline and our approach.

To choose the timestamping locations, we consider all possible components within the host server that may contribute to service time variability; these are locations where significant request processing may occur. For this study, we timestamp at the following locations/events at the host server:

- **e1**: In the host network driver when the packet arrives at the NIC.
- **e2**: At the end of TCP processing.
- **e3**: When the application drains the requests from the socket.
- **e4**: When the application starts processing an individual request.
- **e5**: When the application hands off the response to the kernel.
- **e6**: When the response is dispatched from the host server's NIC.

Timestamping at these different event boundaries provides us an opportunity to analyze the time spent by the requests at different stages, T_{ei}, for $i = 1, 2, \ldots 6$, as shown in Figure 2. However, other fine grained events are used when required, see Section 7.3.

- **driver-to-tcp**: $(T_{e2} - T_{e1})$ is the time spent by the request from driver to TCP layer, representing the network stack processing delay.
- **tcp-to-socket**: $(T_{e3} - T_{e2})$ is the time spent between the TCP layers and the socket, and represents the wakeup/scheduling delay.
- **socket-to-parse**: $(T_{e4} - T_{e3})$ is the time between application processing and socket, and represents the queuing delay at the application level caused by batching of requests.
- **parse-to-response**: $(T_{e5} - T_{e4})$ is the user-space application processing time.

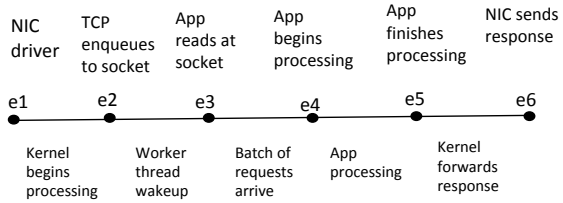


Figure 2: Timestamping locations in our request tracing.

- **response-to-send**: $(T_{e6} - T_{e5})$ is the network stack processing time to dispatch the response from the host server.

To efficiently compute variability at each stage, we maintain a running sum (across requests) of X and X^2, where X is the time spent in a stage. At the end of an experiment, we compute the sample moments $E[X]$ and $E[X^2]$ by dividing the running sums by the number of requests. Finally, we compute $Var(X) = E[X^2] - (E[X])^2$.

In our evaluation, we find that the most significant stages, in terms of variability, are the tcp-to-socket, socket-to-parse, and parse-to-response. We will discuss their role in request processing in detail in the evaluation sections. We also traced the request on the outgoing send path (server socket to driver), but the variability on this path is much smaller, only about 20% of the variability on the incoming path from client to server. Unless otherwise noted, we report $(T_{e6} - T_{e1})$ as the latency of a given request. This definition only includes the server-side latency, which is the focus of our current work.

5 EXPERIMENTAL SETUP

Testbed: We employ two servers in our experiments, one as the client to generate the workload and the other as the server hosting the web application. Each server is equipped with two sockets of Intel Xeon E5-2620 6-core (12 threads, 2.4 GHz) processors, running Ubuntu Linux 14.04 with kernel version 3.16.7. The servers each have 64GB of DRAM (1333 MHz) divided into two 32GB NUMA nodes. The servers each have an Intel I210 1Gb NIC, and are connected via a Quanta LB4M switch. Hyper-threading and power saving mechanisms such as DVFS and sleep states have been disabled in both machines as they are known to cause variability [31].

Web applications: We employ two popular web applications, Memcached [12] and Apache web server [45], for our evaluation.

- *Memcached* is a lightweight and scalable in-memory key-value store, primarily used for accelerating dynamic web applications by acting as a cache for the back-end database. Memcached is employed by several web services, including Facebook [3, 42], Twitter [46], Wikipedia [35], and YouTube [9]. Memcached maintains a list of least-recently-used (LRU) active items in memory to aid the memory reclamation process when an item expires or needs to be evicted. Memcached, from version 1.5.0, has overhauled its LRU maintenance mechanism by using a *dedicated LRU maintainer thread* that speeds up the release of expired items and the allocation of new items [38]. While the dedicated LRU thread

is not on the critical path of request processing, it may interference with requests by requiring a core to run at frequent intervals.

Our Memcached server hosts six million key-value pairs, with the key and value size for each pair set to 20 bytes and 50 bytes, respectively. The workload consists of 90% get and 10% set requests. We use Memcached version 1.5.1 and test two different throughput configurations (Sections 6 and 7).

- *Apache* is a modular, process based web server used by more than 43% of the websites [34]. The Apache web server forks identical processes and can serve a request independent of others. Apache in its newer versions from v2.0 can run in a hybrid multi-process, multi-threaded mode, which improves scalability. We use a newer version (v2.3.4) in our evaluation (Section 8).

Request rate traces: We employ request rate traces in many of our evaluations to drive the application load. Specifically, we use three digitized Memcached traces from Facebook [3], appropriately scaled for our setup, to mimic the load variation experienced by real-world web services. The VAR trace has a gradual drop in request rate, followed by a gradual rise. The APP trace has a steep rise, followed by a relatively constant request rate, and ends with a steep drop. The ETC trace is similar to APP, but is much more bursty. The peak-to-min request rate ratio in all three traces is about 2.

6 EVALUATION 1: REQUEST BATCHING AT THE MEMCACHED CLIENT

Our first evaluation focuses on the Memcached application. In our experiments, we find that Memcached has different sources of variability under high and low throughput configurations. We thus evaluate these configurations separately, starting with high throughput. All the results we report are averaged over 5 experimental runs each.

6.1 Application setup and tracing results

In the high throughput configuration, we run Memcached on the server with six worker threads on six cores of a socket. We use a custom client load generator that sends read/write requests at controllable request rates. In addition to the request rate traces described in Section 5, we also experiment with different inter-arrival time (IAT) distributions, motivated by prior studies on web IATs [2–4, 25]: Deterministic ($C^2_{IAT} = 0$), Exponential ($C^2_{IAT} = 1$), and Bounded Pareto ($C^2_{IAT} = 20$, IAT range of (5μs, 100ms)). The load generator emulates multiple clients by opening several connections sequentially for each worker thread, and sends multiple requests on each connection with the configured IAT. The load generator also supports different popularity distributions for the requested data (keys); unless stated otherwise, we use the random (discrete uniform) distribution.

Figure 3 shows the results of our request tracing for the three stages with the highest variability under Memcached. We see that the *socket-to-parse* stage has the highest variability, by far, compared to others. The socket-to-parse stage at the server involves parsing the drained batch of requests from the socket one request at a time

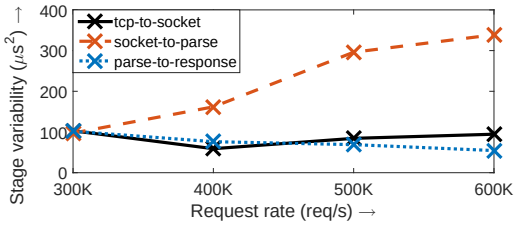
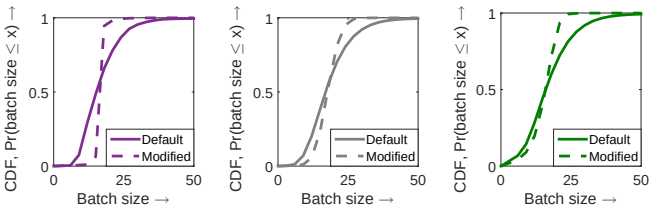


Figure 3: Variability at various request processing stages for Memcached under high throughput setting.

(see Section 4). The time taken in this stage is thus proportional to the size of the batch being served by the application. Consequently, the variability in this stage suggests that there is variability in the request batch sizes as observed by the application at the host server.

6.2 Determining the control knob

To reduce variability in socket-to-parse stage due to request batching, we need to consider the *source* of batching. Requests at the client are batched by the Linux kernel before being sent out so as to avoid the network processing overhead required for each request that is sent ("small-packet problem"). This batching is regulated by Nagle's algorithm [40]. While there are various conditions in the algorithm that trigger the sending of the next batch, we find that it is the condition of getting a response back from the server (kernel) saying it has received the previous batch that triggers the sending of a new batch in our experiments with Memcached. Since network conditions can be variable, the request batch sizes may be variable as well.

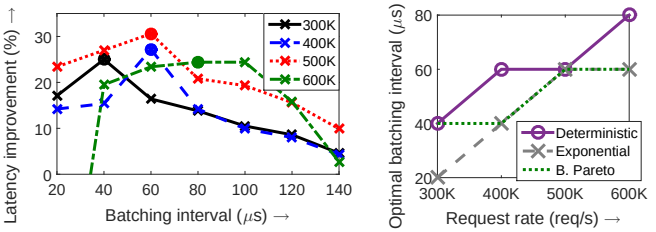


(a) Deterministic IAT. (b) Exponential IAT. (c) Bounded Pareto IAT.

Figure 4: CDF of batch sizes for default and modified Nagle's algorithm under different IAT distributions for 600K req/s.

The solid lines in Figure 4 show our empirical observations for batch sizes under 600K request rate for different IAT distributions. We see that the batch sizes vary from 5 to about 50, with the distribution being more variable for Bounded Pareto IAT. This variance in batch sizes leads to instances where the application is overwhelmed by a large batch of requests, resulting in high tail latencies. Note that once the application finishes serving a large batch of requests and returns to the socket, multiple client-side batches of requests may be waiting in the socket to be drained, resulting in the application finding (server-side) batches as large as 50 requests.

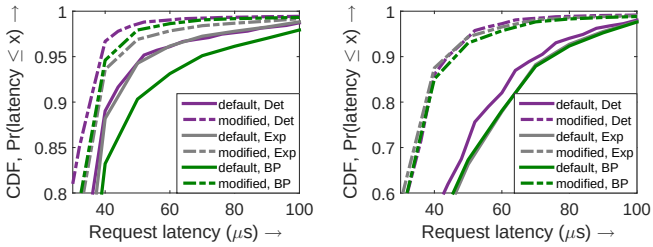
We *modify Nagle's algorithm on the client OS to force batches to be sent out at regular intervals.* This will result in less variable units of work received by the server, indirectly benefiting request latency. Rather than controlling the batch size, we instead control the time between batches, which is easier to implement. When a new request arrives at the client TCP, we append it to the existing



(a) Impact of batch size on request latency improvement under Deterministic IAT.

(b) Optimal batch size for different IAT distributions.

Figure 5: Optimizing the batch size for modified Nagle's.



(a) Request rate: 300K req/s.

(b) Request rate: 600K req/s.

Figure 6: CDF of request latency for default and modified Nagle's algorithm for different IAT distributions.

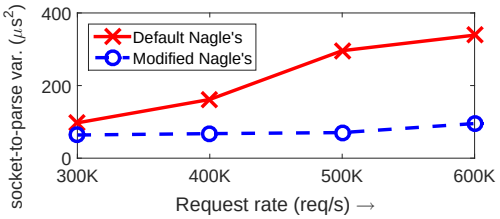


Figure 7: Variability of socket-to-parse stage under the default and modified Nagle's algorithm for Deterministic IAT.

batch and check whether the specified batching interval time has passed since the previous batch was sent. If yes, then we dispatch the current batch; else, we wait for the next request. This modification to Nagle's algorithm reduces the variability in batch sizes, as shown by the dashed lines in Figure 4 that represent the CDF of batch sizes after applying our modification and setting specific batching intervals for 600K req/s. For less variable IAT distributions, such as Deterministic, our modification eliminates almost all variability in batch sizes, as represented by the near-vertical CDF (dashed line) in Figure 4(a). Note that we still batch requests, and so preserve the Nagle's algorithm's functionality of preventing small packets from overwhelming the TCP/IP stack [40].

The "batching interval" length that we specify for sending out batches is a parameter that can be tuned. A large batching interval will result in numerous requests being sent simultaneously in a batch, overwhelming the server. On the other hand, a small interval can overwhelm the network stack with frequent packets. The batching interval must thus be carefully chosen, as we discuss next.

6.3 Evaluation results

Figure 5(a) shows the improvement in mean latency (over default Nagle's batching) for different batching intervals in our experiments under Deterministic IAT. We see that the batching interval does have a significant impact on latency. The optimal batching interval, indicated by the solid circles on each line, improves mean request latency by about 27%, on average, compared to the default Nagle's algorithm.

Figure 5(b) shows the optimal batching interval as a function of request rate for different IAT distributions. In general, the optimal batching interval length increases with request rate. An increase in batching interval leads to larger batch sizes being dispatched to the server, but also results in the server having longer to service each batch. With increasing load, it is likely that the server benefits more from having a longer duration of time to ensure that all requests in the batch are served before taking on a new batch. We find that, compared to the default Nagle's algorithm, our optimal batching leads to more frequent but smaller batches of requests being sent out at predictable intervals.

Figure 6 shows the CDF of request latency under the default and our modified Nagle's algorithm (using the optimal batching interval) for fixed request rates, but with different IAT distributions. Solid lines show the latency for default Nagle's and dashed lines show the latency for our modified algorithm. Given our focus on tail latencies, we intentionally show higher latencies in the CDF figures. For illustration, we show results for the smallest and largest request rate settings in our experiments; results are qualitatively similar for other request rates. We see that the latency reduction is higher for larger request rates. This is because the variable batch sizes overwhelm the server more under high load, and thus our modified batching algorithm has greater opportunity for improvement. The reduction in tail latency (80^{th}–99^{th} percentile) for all distributions ranges from 34-40% for 600K req/s, and from 14-35% for 300K req/s. The improvement in mean request latency likewise ranges from 24-26% for 600K req/s, and from 17-25% for 300K req/s. In general, the improvement is largest for Bounded Pareto IAT (which also has the largest IAT variability), followed by Exponential and Deterministic. This ordering is likely because of the relative variability in batch sizes (dictated by the IAT distribution) for the default case under these distributions.

To validate the efficacy of our chosen control knob in reducing variability, we analyze the socket-to-parse stage variability for the default and modified Nagle's algorithm in Figure 7, for Deterministic IAT; results are similar for other IAT distributions. We see that our modified batching algorithm maintains low variability throughout the request rate range, while the default Nagle's has much higher variability. The reduction in variability in Figure 7 ranges from 34-76%. We find a similar reduction in batch size variability as a function of request rate as well. Note that the reduction in stage variability in Figure 7 is correlated with the latency improvement in Figure 6.

Content popularity: The above results are for random content (key) popularity distribution. We also experiment with the Generalized Pareto distribution using similar parameters as reported by Facebook [3]. In this case, our modified batching reduces tail latency (80^{th}–99^{th} %ile) by about 20-24% and mean latency by about 17%.

Multiple clients: We also experiment with multiple clients, with each new client sending a fixed number of requests over a new connection. Note that only requests within a connection get batched. We vary the number of (sequential) clients per thread from 1 to 50, and the number of requests per client from 100 to 1000. Under our modified batching, the reduction in tail latency (80^{th}–99^{th} %ile) ranges from 13-21% and that for mean latency ranges from 15-17%, across all cases.

Trace-driven: Finally, we also experiment with dynamic trace-driven request rates (see Section 5), with the request rate ranging from 260K–610K req/s. Our modified batching reduces tail latency by about 26-39% and mean latency by about 14-20%, across all traces. The improvement is more pronounced for the ETC trace, likely because it has the highest mean request rate (500K req/s) among all traces.

Our focus on variability of service time in this section led us to request batching as the right control knob for Memcached under high throughput. We show, in Section 9, that request batching is *not* effective for the Apache web server; this is because the socket-to-parse stage variability for Apache is low (see Section 8). It is thus important to consider variability when deciding the right control knob.

7 EVALUATION 2: REDESIGNING THE LRU MANAGEMENT THREAD IN MEMCACHED

7.1 Application setup and tracing results

In the low throughput configuration, we run Memcached with different worker thread and core counts, starting with two worker threads on two cores of a socket. This configuration is representative of typical VM sizes requested by customers in public clouds [6]. We again employ our custom Memcached load generator (see Section 6.1) to vary the request rate (including via traces) and IAT distribution. Unless otherwise stated, we report results for the Deterministic IAT.

Figure 8 shows the results of our request tracing; note the much higher magnitude of variability values here (two core) when compared to the six core configuration in Figure 3. We see that the *tcp-to-socket* stage now has much higher variability compared to the other stages, and the socket-to-parse stage has much lower variability, in contrast to Figure 3. We do not employ the modified batching algorithm here so as to preserve the default application behavior and identify the most significant source of variability under the low throughput configuration. We evaluate the impact of modified batching for the low throughput configuration in Section 9.

The tcp-to-socket stage denotes the time between the TCP processing and the application picking up requests from the socket. Specifically, at the start of this stage, the TCP has enqueued the request(s) in socket, and the kernel now attempts to wake up/schedule the application's worker thread on a core so it can pick up the enqueued requests from the socket. The high variability in this stage indicates that there is variability in the time it takes the application to be scheduled, suggesting that the application is either busy elsewhere or has been scheduled out for other processes. Possible

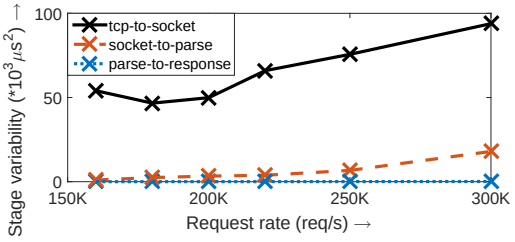


Figure 8: Variability at various request processing stages for Memcached under low throughput setting.

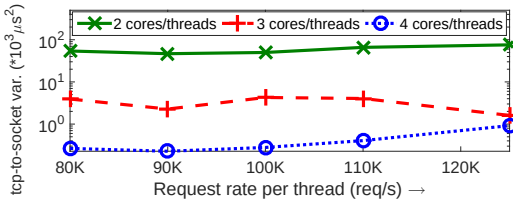


Figure 9: tcp-to-socket variability for different Memcached server configurations. Note the log scale on y-axis.

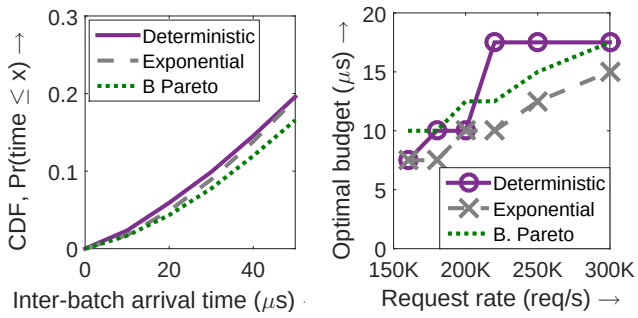
sources of this variability include the application scheduling overhead and competing background threads, such as the dedicated Memcached LRU thread (see Section 5).

To precisely determine the most likely source of variability, we study the change in tcp-to-socket variability as we change the number of cores and threads assigned to Memcached, as shown in Figure 9. Note that the x-axis denotes the request rate *per thread*. We see that the tcp-to-socket variability goes *down* as the number of cores/Memcached threads increases, suggesting that the source of variability is mitigated by increasing the number of cores. This suggests that the dedicated LRU thread interfering with the Memcached worker threads is the likely cause for the increased tcp-to-socket variability. If the source of variability was instead the application scheduling overhead, then the tcp-to-socket variability should not have decreased with the number of threads (since we scale request rate with number of threads), which is contrary to the findings in Figure 9.

7.2 Creating a new control knob

The dedicated LRU thread is an application-specific function and does not expose any tunable knobs. While we can modify the application to do less LRU maintenance work, this would impact the application's functionality. Instead, we wish to preserve the LRU maintenance functionality while reducing variability by mitigating the interference between the LRU thread and the Memcached worker threads.

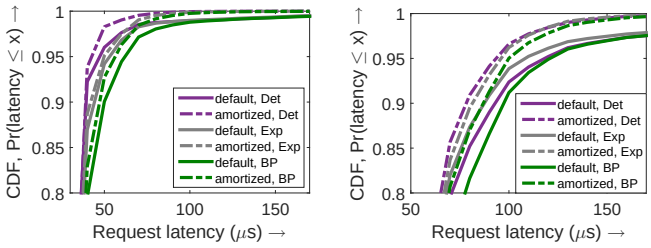
Our key idea is to *offload the LRU maintenance work to the Memcached worker threads and disable the dedicated LRU thread entirely* so it does not cause high variability for the tcp-to-socket stage. Specifically, at the end of each request processing phase and before being scheduled out, we leverage the worker thread itself do some controllable amount of LRU maintenance work. In other words, we *amortize* the LRU maintenance work over all worker thread invocations. We refer to our LRU maintenance approach as the *amortized LRU*.



(a) CDF of inter-batch arrival times at the server for different IAT distributions.

(b) Optimal budget for amortized LRU as a function of request rate.

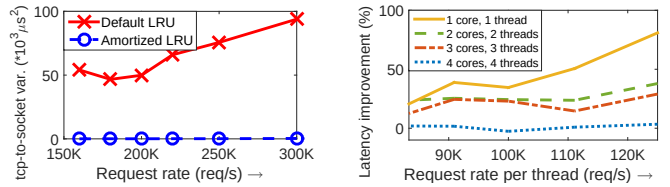
Figure 10: Optimizing the budget for our amortized LRU.



(a) Request rate: 160K req/s.

(b) Request rate: 300K req/s.

Figure 11: CDF of request latency for the default and our amortized LRU mechanisms for different IAT distributions.



(a) Variability of tcp-to-socket stage under the default LRU and our amortized LRU.

(b) Improvement in mean latency under amortized LRU for different configurations.

Figure 12: Reduction in variability and sensitivity analysis.

While it may appear that the LRU maintenance work is now on the critical path, this is not entirely true. By scheduling LRU maintenance only *after* a batch of requests is processed, we are leveraging the idle time between batches to do LRU work. Of course, it is possible that a new batch of requests arrives before our LRU work is completed. To minimize this overstepping, we *regulate the amount of LRU work* done by each worker thread before being scheduled out. We refer to this controllable amount of work as the "budget" of our amortized LRU.

The budget for our amortized LRU must be carefully chosen so as to (i) ensure that the same amount of LRU maintenance work is being done as the dedicated LRU thread, and (ii) minimize the overlap with future request arrivals. While the former constraint implies a larger budget, the latter concern suggests a smaller budget.

7.3 Evaluation results

To determine the optimal budget for our amortized LRU, we analyze the idle time at the server between processing of successive request batches by the application. We timestamp two additional events– e7: When a batch of requests has finished processing (or event e5 for the last request in a batch, see Section 4), and e8: When a new batch of requests has completed its TCP processing and is ready to be serviced (or event e2 for a new batch). Figure 10(a) shows the CDF of inter-batch arrival times, $(T_{e8} - T_{e7})$, for 300K req/s request rate under different IAT distributions; we intentionally focus on a small region of the CDF. We see that almost 90% of the batches have an inter-batch arrival time of at least 30μs. The percentage is even higher for smaller request rates. This suggests that as long as the LRU work is scheduled for no more than 30μs, the negative impact on latency should be small. We thus consider budget values smaller than 30μs to investigate the benefits of amortizing LRU.

We find that the exact value of the budget does not significantly impact request latency. As a result, we seek the *smallest* budget that can still provide an equivalent amount of LRU maintenance as the default dedicated LRU thread. Figure 10(b) shows the optimal budget for different request rates under different IAT distributions. We see that the optimal budget increases with load, as expected, since the amount of LRU maintenance required for items (even in case of the default LRU thread) is proportional to the rate at which they are accessed.

Figure 11 shows the CDF of request latency for the default LRU and our amortized LRU mechanism (using the optimal budget for each request rate) for different IAT distributions and for the smallest and largest request rate setting. Solid lines show the latency for default LRU and dashed lines show the latency for our amortized LRU. The difference in latencies is more pronounced for the higher percentiles, and so we focus on this region in the figures. We see that the latency improvement afforded by our amortized LRU is slightly higher for larger request rates. This is because the default LRU thread interferes with more requests per second when the request rate is higher, and also because the LRU thread has to run for a longer time (to do more maintenance) when the request rate is high; thus, there is more opportunity for improvement at high request rates.

The reduction in tail latency (80^{th}–99^{th} percentile) for all IAT distributions ranges from 4-32% for 300K req/s, and from 6-31% for 160K req/s, with the improvement being slightly higher for Bounded Pareto IAT in general. The improvement in mean request latency across all experiments is about 28%. We also experiment with the Generalized Pareto content popularity distribution. In this case, our amortized LRU reduces tail latency by about 6-23% and mean latency by about 7%. Finally, we also experiment with request rate traces, with the request rate ranging from 90K-200K req/s. Under our amortized LRU, the reduction in tail and mean latency ranges from 7-42% and 22-31%, respectively, across all traces.

Figure 12(a) shows the variability of the target tcp-to-socket stage for the default and amortized LRU mechanisms, for Deterministic IAT. Our amortized LRU significantly reduces the stage variability, with the reduction being more prominent at higher request rates. This is in agreement with the latency improvement trends in Figure 11.

The above results are for the Memcached configuration of 2 cores and 2 threads. We now investigate the results under different configurations of cores and threads, since the impact of interference by the default LRU thread depends on these settings. Figure 12(b) shows the mean latency improvement under Deterministic IAT using our amortized LRU for n cores and n threads, where $n = 1, 2, 3, 4$. The improvement decreases with the number of cores/worker threads; this is because the interference caused by the default LRU thread is shared among all cores, so higher the number of cores, lesser is the impact.

In this section, we demonstrated how non-trivial control knobs can be developed to reduce variability in the critical stages of request processing. By only adding tens of lines of code to change LRU maintenance, we are able to reduce Memcached request latency by about 30%, a significant performance benefit. The evaluations in this and the previous section also highlight the need for request profiling, as different configurations of an application can have different sources of variability and thus different control knobs and solution strategies.

8 EVALUATION 3: APPLICATION THREAD PINNING FOR THE APACHE WEB SERVER

8.1 Application setup and tracing results

We use Apache v2.3.4 with mpm-event module configured to use 6 child processes, running on 6 CPU cores of a socket. We vary the number of threads per process from 5 to 25; unless otherwise stated, we use 25 threads/process. The server hosts a static web page, resulting in a response of 280 bytes. We use httperf [39] as our client driver to experiment with different request rates, including traces. We vary the load by changing the rate at which connections are opened; each connection issues 75 requests using Deterministic IAT. We report load in terms of the total request rate (across all connections).

Figure 13 shows the results of our request tracing for the three stages with the highest variability. We see that the parse-to-response and tcp-to-socket stages have the highest variability; further, the variability increases rapidly with request rate. The parse-to-response stage represents the time taken by the application to process the request, and the tcp-to-socket stage represents the application thread scheduling latency. The high variability in these stages suggests that the worker threads are experiencing unpredictable delays which is affecting their ability to complete the application processing on time (thus the high variability in parse-to-response), and is also delaying their return to pick up requests from the socket. Given that Apache does not have a dedicated maintenance thread like the LRU thread in Memcached, we instead focus on *application thread scheduling* as a possible source of variability. The increase in stage variability with request rate in Figure 13 strengthens our hypothesis since we expect scheduling overhead to increase with load.

8.2 Determining the control knob

By default, the Apache worker threads are not pinned to the cores, and can thus be migrated between cores by the OS scheduler. The

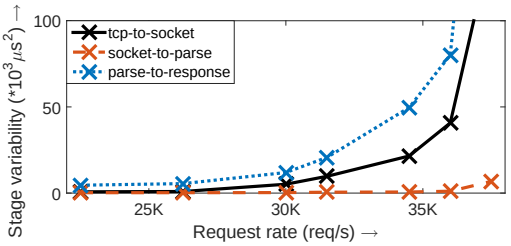


Figure 13: Variability at various stages for Apache web server.

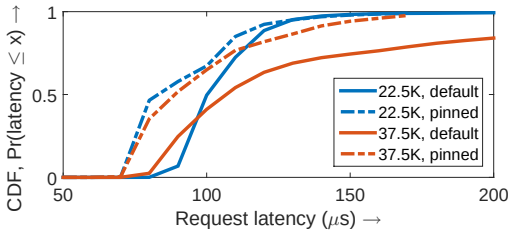


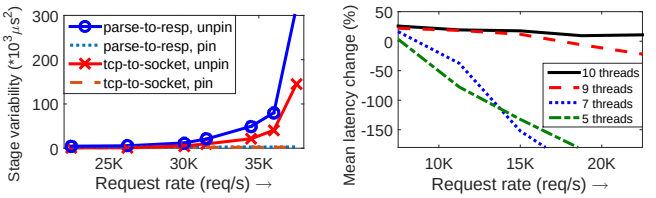
Figure 14: CDF of request latency when threads are unpinned (default) and when threads are pinned.

OS scheduler (CFS, in our case) decides on thread migration based on core availability, cache placement, NUMA affinity, etc. While not pinning the threads has the advantage of leveraging idle cores to reduce mean service time, there is the potential overhead of context switching and associated state migration when migrating threads across cores. This overhead may hinder the progress of worker threads, resulting in unpredictable delays. To reduce variability, we thus *pin the Apache application threads to cores* to prevent thread migration. Specifically, we pin all threads of a process on one core each; thus, each of the six cores handles threads from a specific process.

8.3 Evaluation results

Figure 14 shows the CDF of request latency under the default unpinned case and under the pinned case for the smallest and largest request rate settings in our experiments. We see that pinning threads significantly improves latency over the default case when threads are unpinned (i.e., can run on any of the 6 cores of the socket). The reduction in median and 90%ile latency for 37.5K request rate is 19% and 52%, respectively; the corresponding reduction for 22.5K request rate is 20% and 9%, respectively. The mean latency improvement varies from 15% at low request rates to 50% at high request rates. The higher improvement at high request rates stems from the larger potential in reducing variability (as discussed later). We also experiment with request rate traces, with the request rate ranging from 18K-37K req/s. Pinning threads improves tail latency by about 36-62% and mean latency by about 27-49%; the improvement is highest for the ETC trace.

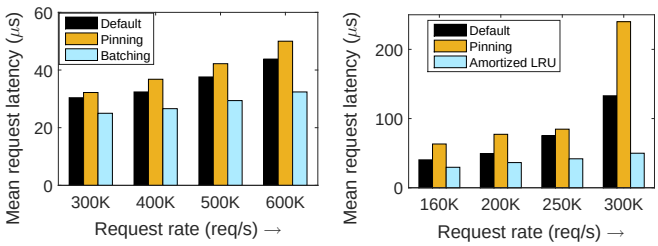
Figure 15(a) shows the variability of the target parse-to-response and tcp-to-socket stages for the unpinned and pinned cases. We see that pinning threads significantly reduces the stage variabilities. This is because pinning reduces the context switching and thread migration penalty, resulting in more predictable delays for



(a) Variability of parse-to-response and tcp-to-socket under unpinned vs pinned.

(b) Change in mean latency under pinning for different number of threads/process.

Figure 15: Reduction in variability and sensitivity analysis.



(a) Memcached, high throughput config.

(b) Memcached, low throughput config.

Figure 16: Using the wrong control knob can hurt latency.

the worker threads. For example, pinning reduces the dTLB and L1-dcache load misses by about 87% and 22%, respectively, compared to the default unpinned case. The reduction is more pronounced at higher request rates. This is likely because the impact of context switching and thread migration penalty under the default unpinned case increases with load. Note the correlation between stage variability reduction in Figure 15(a) and improvement in latency in Figure 14.

The above results for Apache are for the high-throughput case with 25 worker threads/process. We now study the sensitivity of the configuration – number of threads/process – to the improvement in mean latency. Figure 15(b) shows our evaluation results as a function of request rate; we intentionally focus on lower number of threads. We clearly see that the thread configuration has a significant impact on the benefits of pinning over not pinning. For the case of 5 threads/process, pinning does not help for any of the request rates we experiment with, and can lead to a significant increase in latency; this is in contrast to the case of 25 threads/process where pinning is always beneficial. As we increase the number of worker threads, pinning starts outperforming the unpinned case for low request rates. The inflection point occurs at 10 threads, where pinning always helps. This is because the penalty of thread migration under unpinning is higher when there are more threads being scheduled in and out across cores, resulting in cache contents being constantly disrupted; pinning avoids this penalty.

The results in this section highlight the importance of analyzing the sensitivity of the control knob to the application configuration to understand *when* the control knob should be employed, since the benefits (50% latency reduction) and consequences (150% latency increase) of the choice can be significant, as shown in Figures 14 and 15(b). In the next section, we discuss the consequences of selecting the *wrong* control knob for a given application.

9 SIGNIFICANCE OF USING VARIABILITY IN SERVICE TIME AS GUIDING PRINCIPLE

The two key components of our approach, identifying the critical stage and determining the control knob, both rely on using "variability in service time (ST)" as the guiding principle. We now discuss the importance of this metric by comparing with alternative approaches.

Using variability to identify the critical stage:
A natural question to consider in our request profiling approach is – what if we employ "mean ST" as the metric when identifying the target stage instead of "variability of ST"? Consider the Memcached application under low throughput configuration, from Section 7. When employing mean ST as the metric for request profiling, we find that the socket-to-parse stage has the highest ST. Using the arguments from Section 6, we then employ batching as our control knob. However, the impact on request latency when using batching is inconsistent. For low request rates, the optimal batching does improve mean latency by about 28%. However, for higher request rates, even the optimal batching ends up *hurting* mean latency by as much as 32%. By contrast, when using variability of ST as the guiding metric, we consistently reduce mean latency by 26-61% by employing amortized LRU to mitigate variability in the tcp-to-socket stage. This shows that exclusively focusing on reducing mean ST may not always improve latency; a more robust approach is to also consider ST variability.

Using variability to determine the control knob:
Selecting control knobs in an ad-hoc manner is an alternative and sometimes easier approach. Thus, an obvious question is how important is it to select the right control knob?

Consider the Memcached application; an alternative control knob is to pin application threads. Figure 16 shows the resulting mean latency when using the incorrect knob (pinning) and our chosen control knob under Deterministic IAT. For the high throughput configuration, the incorrect knob *hurts* mean latency by about 11.5%, whereas our chosen control knob (batching) improves latency by about 20.9%. Likewise, for the low throughput configuration, the incorrect knob *hurts* latency by about 51.1%, whereas our chosen control knob (amortized LRU) improves latency by about 35.4%.

Similarly, for the Apache web server, another option is to batch requests by modifying the Nagle's algorithm, as in Section 6. However, even under the optimal batching interval for Apache, the improvement in mean request latency is only about 2.7%, whereas that under our chosen control knob (pinning) is about 21.1%.

10 RELATED WORK

Reducing the variability of end-to-end request latency
Dean et al. [7] analyze tail latencies in a production data center that supports their Google search system. The authors acknowledge that latency variability is a problem, and propose application-specific solutions, such as differentiating service classes and reducing head-of-line blocking. While this seminal work established the importance of variability and tail latency in production systems, it focuses on variability in request latency as a whole, and not on specific stages.

Li et al. [31] study sources of tail latency in interactive services by instrumenting applications and the kernel. While our request

profiling approach is inspired by the authors' work, we focus on variability at individual stages of request processing whereas the above work focuses on end-to-end latency. Focusing on per-stage variability can reveal system design opportunities that may otherwise be overlooked. In particular, while the above work also studies the Memcached application, they do not uncover request batching or LRU maintenance as potential optimizations.

Jalaparti et al. [19] focus on the tail latency of Bing web search, and propose application-specific strategies such as reissuing stragglers and speeding up stragglers. Each of these strategies applies to a different "stage", where the concept of stage here refers to a step in the application's workflow and may span thousands of servers; this is in contrast to our definition of "stage" which refers to a fine-grained section of the request's processing path within a single server.

Reducing the latency variability for specific components
Hocko et al. [17] show that making the physical page allocation cache-aware reduces performance non-determinism without significantly increasing mean latency. Pusukuri et al. [30] show that changing the scheduling and migration policies to be aware of cache misses and context switches simultaneously reduces performance variation and improves performance. In addition, several techniques have been developed for promoting fairness in multi-threaded processor cores, shared caches [20, 26, 52], and memory controllers [10, 41], for improving performance by minimizing resource interference.

The above works focus on reducing variability in specific components (such as the OS scheduler), which are known a priori to be the performance bottlenecks. In our work, we profile the request processing path to first determine the potential bottlenecks. Thus, the above works can be employed as control knobs by our approach to reduce the variability at specific stages, *after* those stages have been identified as the potential source of variability.

Shortening the critical path of request processing
Kanev et al. [23] perform a detailed analysis of Google's data center jobs and find that CPU stalls and cache misses are the main causes of low core utilization. To improve utilization, the work calls for shortening the critical path of request processing by employing hardware specialization. Kapoor et al. [24] analyze data center applications and find that kernel overheads contribute significantly to request latency. The authors propose Chronos, a communication framework that bypasses the kernel by employing user-level, zero-copy network functionality, thus shortening the critical path.

The above works are classic examples of approaches that improve performance by reducing the mean ST (shortening the critical path). Our work presents a complementary approach that improves performance by reducing the *variability* of ST.

Request batching
SEDA [51] proposes a staged event-driven architecture for web applications with event batching and thread pool sizing. Yaksha [21] is a control-theoretic admission control proxy for internet services that prevents overload while maintaining high throughput. Elnikety et al. [11] propose black-box admission control for internet services by measuring execution costs and differentiating between request types. While the above works employ batching or admission control to avoid server overload, we precisely control when new requests start being processed by the server to minimize ST variability.

Thread pinning
Kumar et al. [28, 29] consider thread migration to exploit power-performance-area tradeoffs in multi-core machines. Li et al. [32] propose OS scheduling algorithms based on predicting the thread migration overheads. Wang et al. [50] study the interaction between various microarchitectural resources and thread characteristics to efficiently map threads. Thread migration has also been explored for changing thread-level parallelism [1] and for software data spreading [22]. Our work employs thread pinning to specifically reduce ST variability by avoiding state migration across cores.

11 CONCLUSION

Latency is a critical metric for user-facing web services. Existing work typically focuses on shortening the critical path (or mean service time) to improve performance. This paper takes an alternative approach to improving application performance - reducing the variability in request processing. By using "reduction in service time variability" as the guiding principle in web server design, we reveal control knobs that improve request latency (both mean and tail) by up to 28-50% across different scenarios.

Our end-goal is to make the case for using service time variability as a metric to guide system design. The three use cases presented in this paper demonstrate the validity of this approach.

ACKNOWLEDGMENT

This work was supported by NSF grants 1617046, 1717588, & 1750109.

REFERENCES

[1] Murali Annavaram, Ed Grochowski, and John Shen. 2005. Mitigating Amdahl's Law Through EPI Throttling. *SIGARCH Comput. Archit. News* 33, 2 (2005), 298–309.
[2] M. F. Arlitt and C. L. Williamson. 1997. Internet Web servers: workload characterization and performance implications. *IEEE/ACM Transactions on Networking* 5, 5 (1997), 631–645.
[3] Berk Atikoglu, Yuehai Xu, Eitan Frachtenberg, Song Jiang, and Mike Paleczny. 2012. Workload Analysis of a Large-scale Key-value Store. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '12)*. London, England, UK, 53–64.
[4] Theophilus Benson, Aditya Akella, and David A. Maltz. 2010. Network Traffic Characteristics of Data Centers in the Wild. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*. Melbourne, Australia, 267–280.
[5] S. Brin and L. Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. In *Proceedings of the 7th International World-Wide Web Conference*.
[6] Eli Cortez, Anand Bonde, Alexandre Muzio, Mark Russinovich, Marcus Fontoura, and Ricardo Bianchini. 2017. Resource Central: Understanding and Predicting Workloads for Improved Resource Management in Large Cloud Platforms. In *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '17)*. Shanghai, China, 153–167.
[7] Jeffrey Dean and Luiz André Barroso. 2013. The Tail at Scale. *Communications of ACM* 56, 2 (2013), 74–80.
[8] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. 2007. Dynamo: Amazon's Highly Available Key-value Store. In *Proceedings of Twenty-first ACM SIGOPS Symposium on Operating Systems Principles (SOSP '07)*. Stevenson, Washington, USA, 205–220.
[9] Cuong Do. 2007. YouTube Scalability.
[10] Eiman Ebrahimi, Chang Joo Lee, Onur Mutlu, and Yale N. Patt. 2010. Fairness via Source Throttling: A Configurable and High-performance Fairness Substrate for Multi-core Memory Systems. In *Proceedings of the Fifteenth Edition of ASPLOS on*

Architectural Support for Programming Languages and Operating Systems (ASPLOS XV). Pittsburgh, PA, USA, 335–346.

[11] Sameh Elnikety, Erich Nahum, John Tracey, and Willy Zwaenepoel. 2004. A Method for Transparent Admission Control and Request Scheduling in e-Commerce Web Sites. In *Proceedings of the 13th International Conference on World Wide Web (WWW '04)*. New York, NY, USA, 276–286.

[12] Brad Fitzpatrick. 2004. Distributed caching with memcached. *Linux Journal* 2004, 124 (2004).

[13] Phillipa Gill, Martin Arlitt, Zongpeng Li, and Anirban Mahanti. 2007. Youtube Traffic Characterization: A View from the Edge. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement (IMC '07)*. San Diego, CA, USA, 15–28.

[14] Tabb Group. 2008. The Value of a Millisecond: Finding the Optimal Speed of a Trading Infrastructure. http://www.tabbgroup.com/PublicationDetail.aspx?PublicationID=346.

[15] Md E. Haque, Yong hun Eom, Yuxiong He, Sameh Elnikety, Ricardo Bianchini, and Kathryn S. McKinley. 2015. Few-to-Many: Incremental Parallelism for Reducing Tail Latency in Interactive Services. *SIGPLAN Not.* 50, 4 (March 2015), 161–175. https://doi.org/10.1145/2775054.2694384

[16] Mor Harchol-Balter, Bianca Schroeder, Nikhil Bansal, and Mukesh Agrawal. 2003. Size-based Scheduling to Improve Web Performance. *ACM Transactions on Computer Systems* 21, 2 (2003), 207–233.

[17] Michal Hocko and Tomas Kalibera. 2010. Reducing Performance Non-determinism via Cache-aware Page Allocation Strategies. In *Proceedings of the First Joint WOSP/SIPEW International Conference on Performance Engineering (WOSP/SIPEW '10)*. San Jose, CA, USA, 223–234.

[18] Henry Hoffmann, Stelios Sidiroglou, Michael Carbin, Sasa Misailovic, Anant Agarwal, and Martin Rinard. 2011. Dynamic knobs for responsive power-aware computing. In *Proceedings of the 16th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '11)*. Newport Beach, CA, USA, 199–212.

[19] Virajith Jalaparti, Peter Bodik, Srikanth Kandula, Ishai Menache, Mikhail Rybalkin, and Chenyu Yan. 2013. Speeding Up Distributed Request-response Workflows. *SIGCOMM Comput. Commun. Rev.* 43, 4 (Aug. 2013), 219–230. https://doi.org/10.1145/2534169.2486028

[20] Aamer Jaleel, William Hasenplaugh, Moinuddin Qureshi, Julien Sebot, Simon Steely, Jr., and Joel Emer. 2008. Adaptive Insertion Policies for Managing Shared Caches. In *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques (PACT '08)*. ACM, New York, NY, USA, 208–219. https://doi.org/10.1145/1454115.1454145

[21] A. Kamra, V. Misra, and E. M. Nahum. 2004. Yaksha: a self-tuning controller for managing the performance of 3-tiered Web sites. In *Proceedings of the Twelfth IEEE International Workshop on Quality of Service.* Montreal, Canada, 47–56.

[22] Md Kamruzzaman, Steven Swanson, and Dean M. Tullsen. 2010. Software Data Spreading: Leveraging Distributed Caches to Improve Single Thread Performance. In *Proceedings of the 31st ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '10)*. Toronto, Canada, 460–470.

[23] Svilen Kanev, Juan Pablo Darago, Kim Hazelwood, Parthasarathy Ranganathan, Tipp Moseley, Gu-Yeon Wei, and David Brooks. 2015. Profiling a Warehouse-scale Computer. *SIGARCH Comput. Archit. News* 43, 3 (June 2015), 158–169. https://doi.org/10.1145/2872887.2750392

[24] Rishi Kapoor, George Porter, Malveeka Tewari, Geoffrey M. Voelker, and Amin Vahdat. 2012. Chronos: Predictable Low Latency for Data Center Applications. In *Proceedings of the Third ACM Symposium on Cloud Computing (SoCC '12)*. ACM, New York, NY, USA, Article 9, 14 pages. https://doi.org/10.1145/2391229.2391238

[25] T. Karagiannis, M. Molle, M. Faloutsos, and A. Broido. 2004. A nonstationary Poisson view of Internet traffic. In *Proceedings of IEEE INFOCOM 2004*, Vol. 3. 1558–1569.

[26] Seongbeom Kim, Dhruba Chandra, and Yan Solihin. 2004. Fair Cache Sharing and Partitioning in a Chip Multiprocessor Architecture. In *Proceedings of the 13th International Conference on Parallel Architectures and Compilation Techniques (PACT '04)*. Antibes Juan-les-Pins, France, 111–122.

[27] Leonard Kleinrock. 1975. *Queueing Systems, Volume I: Theory.* Wiley-Interscience.

[28] Rakesh Kumar, Keith I. Farkas, Norman P. Jouppi, Parthasarathy Ranganathan, and Dean M. Tullsen. 2003. Single-ISA Heterogeneous Multi-Core Architectures: The Potential for Processor Power Reduction. In *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 36)*. San Diego, CA, USA, 81–92.

[29] Rakesh Kumar, Dean M. Tullsen, Parthasarathy Ranganathan, Norman P. Jouppi, and Keith I. Farkas. 2004. Single-ISA Heterogeneous Multi-Core Architectures for Multithreaded Workload Performance. *SIGARCH Comput. Archit. News* 32, 2 (2004), 64–75.

[30] Kishore Kumar Pusukuri, Rajiv Gupta, and Laxmi N. Bhuyan. 2013. ADAPT: A Framework for Coscheduling Multithreaded Programs. *ACM Trans. Archit. Code Optim.* 9, 4, Article 45 (2013), 45:1–45:24 pages.

[31] Jialin Li, Naveen Kr. Sharma, Dan R. K. Ports, and Steven D. Gribble. 2014. Tales of the Tail: Hardware, OS, and Application-level Sources of Tail Latency. In *Proceedings of the ACM Symposium on Cloud Computing (SOCC '14)*. ACM, New York, NY, USA, Article 9, 14 pages. https://doi.org/10.1145/2670979.2670988

[32] T. Li, D. Baumberger, D. A. Koufaty, and S. Hahn. 2007. Efficient operating system scheduling for performance-asymmetric multi-core architectures. In *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing*. Reno, NV, USA, 1–11.

[33] Zhenhua Liu, Minghong Lin, Adam Wierman, Steven H. Low, and Lachlan L.H. Andrew. 2011. Greening Geographical Load Balancing. In *Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '11)*. San Jose, CA, USA, 233–244.

[34] Netcraft Ltd. 2018 (accessed July 30, 2018). *March 2018 Web Server Survey.* https://news.netcraft.com/archives/2018/03/27/march-2018-web-server-survey.html

[35] mediawiki.org. 2014. memcached. http://www.mediawiki.org/wiki/Memcached.

[36] David Meisner, Brian T. Gold, and Thomas F. Wenisch. 2009. PowerNap: eliminating server idle power. In *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '09)*. Washington, DC, USA, 205–216.

[37] David Meisner, Christopher M. Sadler, Luiz André Barroso, Wolf-Dietrich Weber, and Thomas F. Wenisch. 2011. Power management of online data-intensive services. In *Proceedings of the 38th Annual International Symposium on Computer Architecture (ISCA '11)*. San Jose, CA, USA, 319–330.

[38] memcached 2018 (accessed July 30, 2018). *Memcached documentation new lru.* https://github.com/memcached/memcached/blob/master/doc/new_lru.txt

[39] David Mosberger and Tai Jin. 1998. httperf—A Tool for Measuring Web Server Performance. *ACM Sigmetrics: Performance Evaluation Review* 26 (1998), 31–37.

[40] John Nagle. 1984. Congestion Control in IP/TCP Internetworks. *SIGCOMM Computer Communication Review* 14, 4 (1984), 11–17.

[41] Kyle J. Nesbit, Nidhi Aggarwal, James Laudon, and James E. Smith. 2006. Fair Queuing Memory Systems. In *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 39)*. Orlando, FL, USA, 208–222.

[42] Rajesh Nishtala, Hans Fugal, Steven Grimm, Marc Kwiatkowski, Herman Lee, Harry C. Li, Ryan McElroy, Mike Paleczny, Daniel Peek, Paul Saab, David Stafford, Tony Tung, and Venkateshwaran Venkataramani. 2013. Scaling Memcache at Facebook. In *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*. USENIX, Lombard, IL, 385–398.

[43] Charles Reiss, Alexey Tumanov, Gregory R. Ganger, Randy H. Katz, and Michael A. Kozuch. 2012. Heterogeneity and Dynamicity of Clouds at Scale: Google Trace Analysis. In *Proceedings of the 3rd ACM Symposium on Cloud Computing (SoCC '12)*. San Jose, CA, USA, Article 7.

[44] Eric Schurman and Jake Brutlag. 2009. The User and Business Impact of Server Delays, Additional Bytes, and HTTP Chunking in Web Search. http://velocityconf.com/velocity2009/public/schedule/detail/8523.

[45] The Apache Software Foundation. [n. d.]. Apache HTTP Server Project. https://httpd.apache.org.

[46] Twitter. [n. d.]. Twemcache: Twitter Memcached. https://github.com/twitter/twemcache.

[47] B. Urgaonkar and A. Chandra. 2005. Dynamic Provisioning of Multi-tier Internet Applications. In *Proceedings of the 2nd International Conference on Automatic Computing (ICAC '05)*. Seattle, WA, USA, 217–228.

[48] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer, and A. Tantawi. 2005. An analytical model for multi-tier internet services and its applications. In *Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '05)*. Banff, Alberta, Canada, 291–302.

[49] Q. Wang, C. Lai, Y. Kanemasa, S. Zhang, and C. Pu. 2017. A Study of Long-Tail Latency in n-Tier Systems: RPC vs. Asynchronous Invocations. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. 207–217. https://doi.org/10.1109/ICDCS.2017.32

[50] W. Wang, T. Dey, J. Mars, L. Tang, J. W. Davidson, and M. L. Soffa. 2012. Performance analysis of thread mappings with a holistic view of the hardware resources. In *Proceedings of the 2012 IEEE International Symposium on Performance Analysis of Systems Software*. New Brunswick, NJ, USA, 156–167.

[51] Matt Welsh, David Culler, and Eric Brewer. 2001. SEDA: An Architecture for Well-conditioned, Scalable Internet Services. *SIGOPS Oper. Syst. Rev.* 35, 5 (2001), 230–243.

[52] Yuejian Xie and Gabriel H. Loh. 2009. PIPP: Promotion/Insertion Pseudo-partitioning of Multi-core Shared Caches. *SIGARCH Comput. Archit. News* 37, 3 (2009), 174–183.

[53] M. Zhang, N. S. Islam, S. Sur, H. Subramoni, D. K. Panda, J. Huang, M. Wasi ur Rahman, M. Luo, X. Ouyang, J. Jose, and H. Wang. 2011. Memcached Design on High Performance RDMA Capable Interconnects. In *Proceedings of the 2011 International Conference on Parallel Processing*. Taipei, Taiwan, 743–752.

Aggregating E-commerce Search Results from Heterogeneous Sources via Hierarchical Reinforcement Learning

Ryuichi Takanobu*
Institute for AI, State Key Lab of Intelligent Technology & Systems, DCST, Tsinghua University
Beijing, China
gxly15@mails.tsinghua.edu.cn

Tao Zhuang*
Alibaba Group
Hangzhou, China
zhuangtao.zt@alibaba-inc.com

Minlie Huang†
Institute for AI, State Key Lab of Intelligent Technology & Systems, DCST, Tsinghua University
Beijing, China
aihuang@tsinghua.edu.cn

Jun Feng‡
State Grid Zhejiang Electric Power Co., LTD
Hangzhou, China

Haihong Tang
Alibaba Group
Hangzhou, China

Bo Zheng
Alibaba Group
Hangzhou, China

ABSTRACT

In this paper, we investigate the task of aggregating search results from heterogeneous sources in an E-commerce environment. First, unlike traditional aggregated web search that merely presents multi-sourced results in the first page, this new task may present aggregated results in all pages and has to dynamically decide which source should be presented in the current page. Second, as pointed out by many existing studies, it is not trivial to rank items from heterogeneous sources because the relevance scores from different source systems are not directly comparable. To address these two issues, we decompose the task into two subtasks in a hierarchical structure: a high-level task for *source selection* where we model the sequential patterns of user behaviors onto aggregated results in different pages so as to understand user intents and select the relevant sources properly; and a low-level task for *item presentation* where we formulate a slot filling process to sequentially present the items instead of giving each item a relevance score when deciding the presentation order of heterogeneous items. Since both subtasks can be naturally formulated as sequential decision problems and learn from the future user feedback on search results, we build our model with hierarchical reinforcement learning. Extensive experiments demonstrate that our model obtains remarkable improvements in search performance metrics, and achieves a higher user satisfaction.

CCS CONCEPTS

• **Information systems** → **Combination, fusion and federated search**; *Online shopping*; • **Computing methodologies** → **Reinforcement learning**; *Learning from implicit feedback*.

*Both authors contributed equally to this research.
†Corresponding author: Minlie Huang.
‡Participated in this work while at Tsinghua University.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313455

KEYWORDS

aggregated search, vertical, user feedback, hierarchical reinforcement learning

ACM Reference Format:
Ryuichi Takanobu, Tao Zhuang, Minlie Huang, Jun Feng, Haihong Tang, and Bo Zheng. 2019. Aggregating E-commerce Search Results from Heterogeneous Sources via Hierarchical Reinforcement Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313455

1 INTRODUCTION

The process of aggregating search results from heterogeneous sources is usually referred to as *aggregated search* [17]. Different from a meta-search engine or search federation which assumes that all distributed sources contain homogeneous content and applies the same ranking function to each source for result fusion [1], an aggregated search system is expected to support **heterogeneous information seeking** from different sources. When a user issues a query, the search system should integrate relevant results from each specialized search system, called *vertical*, into one Search Engine Result Page (SERP).

Existing research on search result aggregation has focused on web search [4, 11, 21], while in this paper, we study the task in E-commerce search with one of the largest E-commerce search services in the world: *Taobao.com*. Fig. 1 illustrates that traditional E-commerce search can be augmented with *blog posts* that share purchasing experiences, or *topic groups* that cluster products with the same brand or from the same shop to facilitate finding similar products. As a matter of fact, user study in Taobao Search shows that mixing products with topic groups and blog posts in a SERP can greatly improve the user experiences of product search and online shopping. In this study, we aggregate two other verticals in product search, namely the topic and blog verticals. Given a query, these verticals return topic groups and blog posts respectively.

In web search [4, 11, 21], result aggregation is performed only once at the first page for each query. While in E-commerce search, aggregation is performed multiple times for a query, one for each

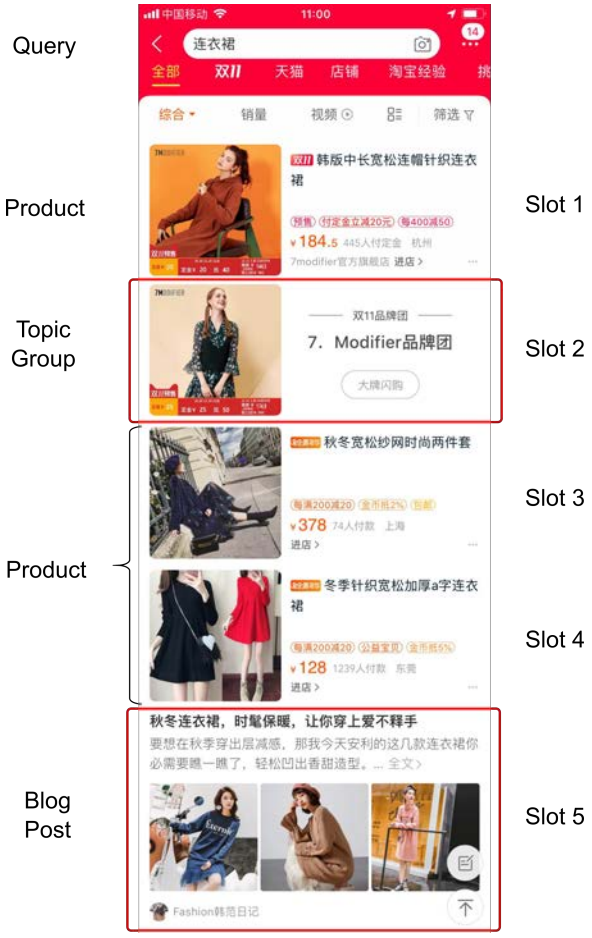


Figure 1: An aggregated search example for the query "dress", where the topic group from the *topic* vertical is shown in the 2nd position, and the blog post from the *blog* vertical in the 5th position.

page. The aim of page-wise aggregation in E-commerce search is to keep diversity to attract and retain diverse customers. This is a major difference from previous web search aggregation in which a single decision is made only for the first page.

In this paper, we decompose search result aggregation in E-commerce search into two subtasks: *source selection* and *item presentation*. *Source selection* decides whether to present the search results of a certain source type in the current page, which depends on the user behaviors onto the items that have been already presented in previous pages. For example, if a user issued the query "sport shoes", and clicked several products with the same brand, presenting a topic group with the same brand in the next page may be a good decision. Then the validity of search results can be examined afterwards by analyzing the long-term gains like click through rate (CTR). So this process is a sequence of decisions and can be modeled with reinforcement learning (RL) [32], which helps better understand user intents to present more relevant aggregated search results.

The other subtask, *item presentation*, is to decide the presentation order of the items from heterogeneous sources, which requires to estimate the relevance scores of all items in a unified way. However, items from different sources have different relevance estimation models [2, 16, 28], thereby making the scores not comparable between different sources. This is termed the *relevance ranking* issue. To avoid this, we formulate the item presentation task as a *slot filling* problem where the training signal comes directly from the user feedback. In this formulation, each display position in a page is regarded as a slot, and the slots in a page will be filled sequentially. At each position, an *item presenter* is trained to choose the most relevant item from the candidate sources, and then fills in the slot with the selected item. Once again, the slot filling process is consistent with the sequential nature of user behaviors, which can be viewed as a sequential decision problem and handled by RL.

Given the above decomposition, we propose a hierarchical reinforcement learning (**HRL**) model [5] consisting of two components: a source selector that decides which sources should be selected at the current page, and an item presenter that decides the presentation order of the items selected from the selected sources in a page. The model fully utilizes the sequential characteristics of user behaviors across different pages to decide both the sources and items to be presented. The hierarchical framework enables the model to utilize the immediate user feedback in the current page as well as the future user feedback on the entire search session.

To summarize, our main contributions are as follows:

- We propose a novel search result aggregation method that formulates a semi-Markov decision process which is composed of a high-level policy for *source selection* and a low-level policy for *item presentation*.
- We present aggregated search results for each page during source selection and capture the sequential patterns of user behaviors onto aggregated items in different pages.
- We formulate the item presentation subtask as a slot filling problem in order to avoid the relevance ranking issue on displaying heterogeneous items.

Results demonstrate that our proposed model can improve user experience and make significant advancement on marketing metrics like CTR and GMV over other baselines.

2 PRELIMINARIES

2.1 Reinforcement Learning

Reinforcement learning (RL) [32] is a learning paradigm that an agent learns from the interactions with the environment through sequential exploration and exploitation. Generally, RL follows a Markov Decision Process (MDP) formulation.

Let an MDP be defined as $< \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma >$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ denotes the set of possible actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ denotes the transition function to generate the next state from current state-action pair, $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ denotes the reward function and $\gamma \in [0, 1]$ is an discount factor . Given an MDP, a trajectory $\tau = s_0, a_0, r_0, s_1, \cdots$ that starts from state s_0 is sampled from a policy $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ that specifies the action. The aim of RL is to find an optimal policy π^* which maximizes the expected cumulative reward (return) $R_t = \mathbb{E}[\sum_{k=0}^{\infty} \gamma^k r_{t+k}]$ at

each time step t. For an agent following a policy π, the Q-value of the state-action pair (s, a) is defined as $Q_\pi(s, a) = \mathbb{E}[R_t|s_t = s, a_t = a, \pi] = \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)}[r + \gamma \mathbb{E}_{a' \sim \pi(s')}[Q_\pi(s', a')]|s, a]$, which measures the average discounted long-term rewards. In the practice of training, the action is often selected by an $\epsilon - greedy$ policy that follows the greedy policy $a = \arg\max_{a'} Q(s, a')$ with probability $1 - \epsilon$ for exploitation and selects a random action with probability ϵ for exploration, in order to collect desired transitions for learning.

With the advance of deep neural networks, deep reinforcement learning (DRL) has been widely used in various search tasks in recent two years [12, 25, 41].

2.2 Options Framework

Options framework [33] extends the usual MDP so that it involves temporal abstractions over the action space in the context of HRL. At each step, the HRL agent chooses either a "one-step" action (primitive action) or a "multi-step" action (option). An option $o = (\pi, \mathcal{I}, \beta) \in O$ defines a policy π over actions (either primitive or option), includes an initiation set $\mathcal{I} \subset \mathcal{S}$ that option o is available *iff* state $s \in \mathcal{I}$, and can be terminated according to a stochastic function $\beta : \mathcal{S} \rightarrow [0, 1]$. An MDP endowed with a set of options is extended to a Semi-Markov Decision Process (SMDP). Given an SMDP, a trajectory τ is sampled from a policy μ over options. Each option o_k launches a new subtask where a sub-trajectory τ_k is sampled from the corresponding policy π over actions. The agent cannot select next option (possibly in a recursive way) until current option terminates according to the termination function β.

Recently, several studies have demonstrated that combining DRL with predefined subgoals delivers promising results in challenging environments like Atari [18], Minecraft [35], relation extraction [34] and task-oriented dialogues [26].

3 RELATED WORK

In recent years, search engines are able to search heterogeneous sources that contain different types of contents (e.g. news, maps, videos, images). Studies have shown that the aggregated view helps expose the content in other sources, thus offering the chance for the user to explore different sources in search[7, 10]. Existing aggregated search systems, which merge all heterogeneous results from various search services in one SERP, mostly adopt a pipeline architecture as illustrated in Fig. 2. In general, the architecture contains two key components, one is source selection which decides the vertical sources from which items should be ranked, and the other is item presentation which decides the presentation order of the heterogeneous items.

3.1 Vertical Selection

The first subtask of aggregated search is vertical selection, also called source selection. The goal of vertical selection is to decide which verticals are to be presented in a SERP, e.g. whether it is necessary to present the videos of "violin" when a user types query "violin". Several works [20, 37, 44] have found that participants rated the system negatively when irrelevant vertical results were presented in an aggregated SERP. It is thus essential to evaluate the relevance of a source to a query.

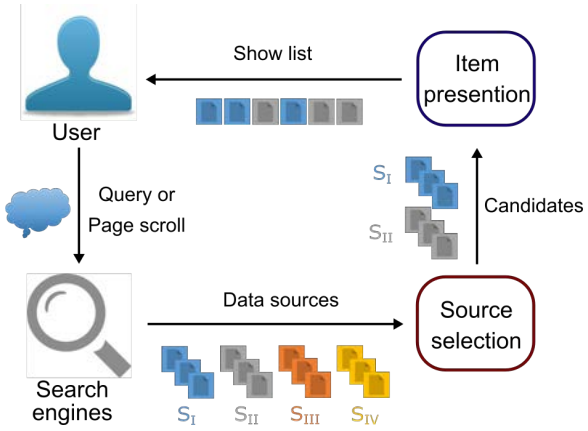


Figure 2: Task flow of search aggregation on heterogeneous sources.

Most existing approaches use a binary classifier for each vertical to make a decision on whether to present a vertical result in a page [15, 19, 36]. From this perspective, each classifier can adopt a different feature representation and focus on the features that are uniquely predictive for its corresponding vertical. Human judges are employed to generate ground-truth labels for the verticals in some approaches [4, 11]. Other approaches [16, 28] leverage user search logs and assign a binary label based on user behaviors.

As previous studies focused on web search, vertical selection is only launched one time when a user issues a query. For instance, the *video* vertical will only appear once in the first SERP for query "Titanic", but no vertical will be shown again in the following pages unless the user types a new query. However, the goal of E-commerce search is quite different from web search [45]. With respect to aggregated search, the main difference between web and E-commerce lies in that the latter needs to present the aggregated search results in each page to meet user search demand, which requires the system to capture the sequential patterns of user behaviors onto aggregated results.

3.2 Vertical Presentation

The second subtask, vertical presentation, or item presentation, is to rank the candidate items selected from verticals. For instance, should the videos of "violin" be positioned above or below the web links in a page? In general, the system presents the most relevant items at the top, which are more likely to be viewed by users [9, 31]. Existing approaches to vertical presentation can be classified into three types: *pointwise*, *pairwise* and *attention-based* approaches.

Pointwise approaches train an independent classifier per vertical to predict the relevance of the corresponding vertical. *Click models* were proposed to predict the user response (i.e. click or skip) to vertical results [16, 38, 39]. FCM [8] considered the rank of the verticals, their visual salience, and the distance between them and the current item. Markov et al. [22] proposed a click model that estimated different probabilities for results above or below the vertical as well. Others directly trained *vertical-specific classifiers* to predict the relevance score of a source to a query, which was used to decide

the position of each vertical [6, 27, 30]. Ponnuswami et al. [28] trained a gradient boosted decision tree to learn relevance scores and ranked each vertical individually using a computed threshold. Although it is simple and intuitive to implement pointwise models, they suffer from the *relevance ranking* issue that the prediction scores from individual classifiers are not directly comparable.

Pairwise approaches learn to predict the *relative preference* between candidate sources to be displayed in a SERP. Arguello et al. [2] trained one binary classifier for user preference per vertical pair, and derived a final ranking from the predicted preferences. The pairwise method solves the *relevance ranking* issue, but in return, it requires a complex ranking principle among verticals and results in a large number of pairwise classifiers.

The recent study [42] adapted an **attention-based** method that applied different weights to different information sources for relevance estimation. It has gained a significant improvement on ranking relevance, however, it requires a large number of data annotations on relevance score to support model training.

4 HIERARCHICAL SEARCH AGGREGATION

We develop a deep hierarchical reinforcement learning algorithm for aggregated search in this paper. The model consists of a high-level source selection policy and a low-level item presentation policy. The entire process works as shown in Fig. 3. When a new search request arrives, the source selector decides which sources to present on the current page. The item presenter then decides the presentation order of the relevant items from selected sources. The user search session continues when the user scrolls to the next page, and ends when he leaves the platform or types a new query. Both policies are trained with the DQN [24] algorithm. The rewards for policy training come from implicit user feedback, instead of any artificial metrics [3, 43], so that it can learn from user behaviors straightforwardly.

In the following subsections, we will present the details of the two policies, deep Q network architecture, and model optimization. The notations used in this paper are summarized in Table 1.

Notation	Description
x	Search request
$\mathcal{SR}_j$	Set of all items of source type j
m_j^k	K-th item of source type j
$n_j(x)$	The number of items of type j given the request x
p, P	Slot position on the SERP, Result presentation
s, $\mathcal{S}$	State, State set
a, $\mathcal{A}$	Primitive action, Primitive action set
o, O	Option, Option set
r^e, r^i	Extrinsic reward, Intrinsic reward
γ	Discount factor
R	Return (discounted cumulative reward)
μ, π	Source selector policy, Item presenter policy

Table 1: Notations of aggregated search and HRL.

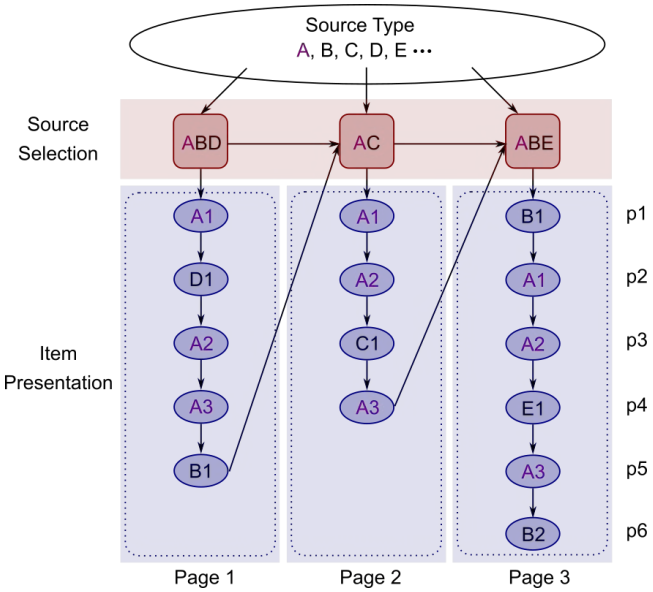


Figure 3: The hierarchical architecture on aggregated search. This example of user search session is composed of three search requests (page 1 to 3). A stands for the products, and $B, C \cdots$ for the items of other source types.

4.1 Source Selection with High-level Policy

The high-level RL process aims to decide which sources should be selected in each page. The high-level policy μ perceives the high-level state s^e and selects an option o that tells what sources to display. The option o will trigger the low-level policy to decide how to present the items from these selected sources. The high-level source selector tries to capture the sequential patterns of user behaviors onto aggregated results in different pages.

4.1.1 Option. An option $o \in O$ refers to a multi-step action [33] that selects the relevant sources to display. The core search, which is the product search in our case, must be selected, while verticals can either be selected or not. So if there are N verticals, the size of the option set is $|O| = 2^N$. The agent samples an option o according to μ, then the control of the agent transfers to the low-level policy. The option lasts until the low-level RL process terminates, then the source selector continues to execute the next option.

4.1.2 State. The state $s_t^e \in \mathcal{S}$ of the high-level RL process at time step t is represented by: 1) the search request x, 2) the triggered search source results $\mathcal{SR}(x)$, and 3) the *latest* option $o_{t'}$ where $t - t'$ is the duration of $o_{t'}$. Then the state vector $\phi(s_t^e)$ is the concatenation of the embedding vector of each part. It should be noted that the vector of source type $\mathcal{SR}_j$ is represented by the mean value of all item vectors of that type, i.e. $\mathbf{e}(\mathcal{SR}_j(x)) = \frac{1}{n_j(x)} \sum_k \mathbf{e}(m_j^k(x))$ where $\mathbf{e}(m)$ is the item vector of m. In particular, the vector of a certain source is represented by zeros if the result of that type is empty. Then the vector of entire source results $\mathbf{e}(\mathcal{SR})$ is derived from the concatenation of $\mathbf{e}(\mathcal{SR}_j)$ from all source types.

4.1.3 Extrinsic reward. The extrinsic reward r^e is given to the source selector to evaluate the global performance. It is derived from the mean value of the intrinsic return from the low-level process, which can be formulated as follows:

$$r_t^e = \frac{1}{l} R_t^i = \frac{1}{l} \sum_{k=0}^{l-1} \gamma^k r_{t+k}^i, \tag{1}$$

where l is the duration of the option o_t. The average return is calculated to prevent the tendency that the source selector greedily chooses all triggered sources to obtain more rewards. The definition of intrinsic reward will be introduced soon later.

4.2 Item Presentation with Low-level Policy

The low-level RL process aims to decide the presentation order of the items from the candidate sources $\overline{\mathcal{SR}}(x)$ chosen by the high-level source selector. Note that the items from each source have been ranked according to its private vertical, and all items from verticals can be interleaved with results from other sources. Each display position is regarded as a slot and the entire presentation subtask starts by filling an item into the top slot of the page, and then the second slot, etc. At each step, the low-level RL policy π takes the corresponding option o and the low-level state s^i as input, and outputs a primitive action a that tells which source to display at the current position. As a result, the agent does not give a relevance score to each item thus avoiding the relevance ranking issue.

Considering that the source selector has notified which sources are selected to display, the number of slots on the page can be determined by $l = \sum_{\mathcal{SR}_j \in \overline{\mathcal{SR}}} n_j(x)$. The agent reaches the subgoal when all the slots in a page have been filled, thus the number of slots is equal to the time duration of the option, and the termination function $\beta : \mathcal{S} \rightarrow \{0, 1\}$ is deterministic in our setting. Once it is terminated, the control of the agent will return to the high-level process.

4.2.1 Primitive action. A primitive action $a \in \mathcal{A}$ refers to a one-step action that chooses a certain source type, so $|\mathcal{A}| = 1 + N$ given the core search and N verticals. Each source can be seen as a stack with its items in descending order of relevance to the search request x. If a certain source is selected according to π, the top item at the stack of this source is displayed in the current position, as shown in Fig. 4. When a source stack is empty, the source is excluded from the action space. In this way, we simplify the relevance ranking problem into a slot filling process to mix different sources: we only need to decide from which source the top item should be selected, instead of evaluating the relevance of each source item. Note that the relative order of the items within the same source type remains unchanged, the agent simply utilizes the existing ranking list.

4.2.2 State. The state $s_t^i \in \mathcal{S}$ of the low-level RL process at time step t is represented by: 1) the search request x, 2) the *top items* of candidate sources $top(\overline{\mathcal{SR}}(x))$, 3) the last primitive action a_{t-1}, and 4) the option o that launches the current subtask, and the state vector $\phi(s_t^i)$ is the concatenation of all vectors of each corresponding part. In order to give an intuitive understanding of *top items*, we take Fig. 4 as an instance. $top(\overline{\mathcal{SR}}) = [A5; B1; C2]$ before the agent presents the item at 6th slot position, then $top(\overline{\mathcal{SR}}) = [A5; B2; C2]$ after it chooses the source type B. So naturally, the vector of candidate sources $\mathbf{e}(top(\overline{\mathcal{SR}}))$ is represented by the concatenation of the top item vector $\mathbf{e}(m_j^{top})$ from each selected source. In addition, similar to [29], we put options into the low-level state representation as an additional input throughout the low-level RL process, to make the selected sources available for the policy π.

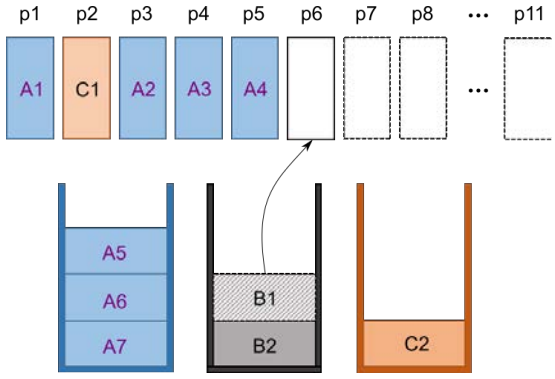


Figure 4: Illustration of the proposed slot filling scheme. There are three sources (A,B,C) and 7+2+2=11 slots in this example.

4.2.3 Intrinsic reward. The intrinsic reward r^i is provided for the item presenter to indicate how well a particular subtask is completed. User clicks and transactions are used to form the intrinsic reward, which is computed as below:

$$r_t^i = \lambda * click + (1 - \lambda) * \min(ln(1 + pay), \delta), \tag{2}$$

where λ is a weight factor, $click \in \{1, -1\}$ indicates whether the user clicks or skips the current slot, and $pay \in [0, +\infty)$ is the price of the product that the user buys. When the user does not buy anything, the second term is zeroed as $pay = 0$. The second term is also clipped by the value δ to reduce the impact of extreme observations. In addition, if there is no click or transaction during the current subtask, a small negative value is added to the intrinsic reward at the last time step as a penalty.

4.3 Q Network Framework

Deep Q-learning uses a multi-layer neural networks with parameters θ to estimate the Q function. Given an n-dimensional state s, it outputs a vector of Q-values over the actions $Q(s, \cdot; \theta) : \mathbb{R}^n \rightarrow \mathbb{R}^{|\mathcal{A}|}$.

In our algorithm, both high-level and low-level policies leverage the same Q network framework, as shown in Fig. 5. We apply DRQN [14] by considering the previous context with a recurrent structure to integrate the underlying correlation between the previous state and current observation. Eq. 3 refers to the hidden layer and RNN layer in Fig. 5.

$$\begin{aligned} \mathbf{w}_t &= \beta(\mathbf{W}_\phi \cdot \phi(s_t) + b_\phi), \\ \mathbf{h}_t &= GRU(\mathbf{w}_t, \mathbf{h}_{t-1}), \end{aligned} \tag{3}$$

where $\phi(s)$ is the vector representation of the state s, β is *Leaky ReLU* activation function.

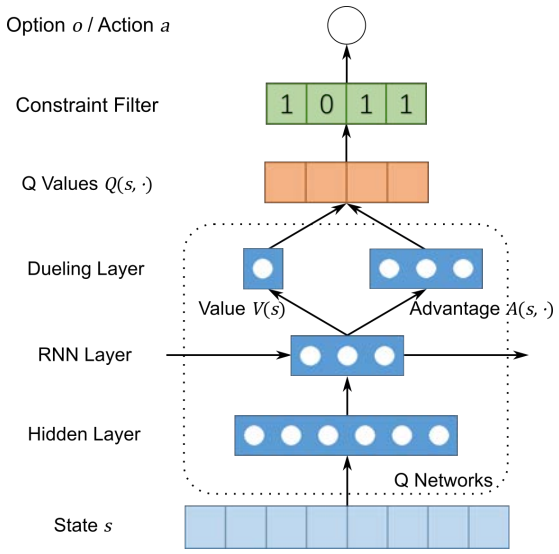


Figure 5: Q network framework used in both high and low level policies.

Besides, the dueling architecture [40] is introduced as well to get a more precise policy evaluation in the presence of many similar-valued actions. It consists of two streams that represent the value and advantage functions, and the two streams are combined afterward to provide a better estimate of Q-values, i.e. the dueling layer of Fig. 5.

$$Q(s,a;\theta) = V(s;\theta) + (A(s,a;\theta) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a';\theta)). \quad (4)$$

The *constraint filter* is to explicitly perform an element-wise mask operation on the Q-values to meet some complicated commercial needs. In this way, commercial constraints in a special situation can be easily satisfied in this framework. For instance, not displaying any blog posts in the first page can be dealt with by the source selector via excluding the corresponding source type from the action space. Similarly, displaying a particular source item at a pre-specified position can be handled by the item presenter via excluding the corresponding slot from the slot sequence.

4.4 Hierarchical Policy Learning

Standard DQN is used to minimize the squared error between the target (Bellman optimality condition) $r + \gamma \max_{a'} Q(s', a'; \theta)$ and its estimate $\hat{Q}(s, a; \theta)$. In the context of HRL, the top-level policy μ tries to minimize the following loss function at each time step t:

$$\begin{aligned} L_\mu(\theta_t^e) &= \mathbb{E}_{(s,o,r,s')\sim D^e}[(y_t^e - Q_\mu(s,o;\theta_t^e))^2], \\ y_t^e &= r + \gamma^l Q_\mu(s', \arg\max_{o'} Q_\mu(s',o';\theta_t^e);\theta_t^{e-}), \end{aligned} \quad (5)$$

where l is the number of steps the option o lasts. Here we apply experience buffer [23] to smooth over changes in the data distribution. We also employ a target Q network [24] with parameters θ^- to decrease the possibility of divergence or oscillations during the training process, and double Q-learning [13] to reduce the observed over-estimations caused by traditional Q-learning. Similarly,

Algorithm 1: Deep hierarchical policy learning for aggregated search

Input: Item sets with multiple source types $\mathcal{SR}$ and their search services.

Initialize the two policies with parameters $\{\theta^e, \theta^i\}$
Initialize replay memories $\{D^e, D^i\}$ respectively
$\theta^{e-} = \theta^e$, $\theta^{i-} = \theta^i$
foreach *user search session* **do**
  Initialize high-level state s^e with current search request x
  repeat
    // Page-level source selection
    Set option $o =$
    $\begin{cases} \text{random sample option } o \text{ from } \mathcal{O}, & random() < \epsilon^e \\ \arg\max_{o'} Q^\mu(s^e, o'; \theta^e), & otherwise \end{cases}$
    Initialize low-level state s^i and option duration l according to o
    for $k = 0$ **to** $l - 1$ **do**
      // Slot-level item presentation
      Set primitive action $a =$
      $\begin{cases} \text{random sample action } a \text{ from } \mathcal{A}, & random() < \epsilon^i \\ \arg\max_{a'} Q^\pi(s^i, o, a'; \theta^i), & otherwise \end{cases}$
      Execute action a, observe intrinsic reward r^i (cf. Eq. 2) and next state s'^i
      Store transition (s^i, o, a, r^i, s'^i) in D^i
      Randomly sample mini-batches from D^i
      Perform gradient descent on $L_\pi(\theta^i)$ (cf. Eq. 6 & 8)
      $s^i = s'^i$
      Assign $\theta^{i-} = \theta^i$ every C^i steps
    end
    Obtain extrinsic reward r^e (cf. Eq. 1) and next state s'^e
    Store transition (s^e, o, r^e, s'^e) in D^e
    Randomly sample mini-batches from D^e
    Perform gradient descent on $L_\mu(\theta^e)$ (cf. Eq. 5 & 7)
    $s^e = s'^e$
    Assign $\theta^{e-} = \theta^e$ every C^e steps
  until *the session ends*
end

we learn the low-level policy π by minimizing the following loss function:

$$\begin{aligned} L_\pi(\theta_t^i) &= \mathbb{E}_{(s,o,a,r,s')\sim D^i}[(y_t^i - Q_\pi(s,o,a;\theta_t^i))^2], \\ y_t^i &= r + \gamma Q_\pi(s', o, \arg\max_{a'} Q_\pi(s',o,a';\theta_t^i);\theta_t^{i-}), \end{aligned} \quad (6)$$

given the option o from μ.

Differentiating the loss function with respect to the weights, we arrive at the following gradient for the high-level policy:

$$\nabla_{\theta_t^e} L_\mu = \mathbb{E}_{(s,o,r,s')\sim D^e}[(y_t^e - Q_\mu(s,o;\theta_t^e))\nabla_{\theta_t^e} Q_\mu(s,o;\theta_t^e)], \quad (7)$$

and similarly, the gradient for the low-level policy yields as below:

$$\nabla_{\theta_t^i} L_\pi = \mathbb{E}_{(s,o,a,r,s')\sim D^i}[(y_t^i - Q_\pi(s,o,a;\theta_t^i))\nabla_{\theta_t^i} Q_\pi(s,o,a;\theta_t^i)]. \quad (8)$$

Note that the initial hidden state of RNN layer should be carried forward from its previous values $\mathbf{h}_{t-1}$ when sampling actions, but we zero it at the start of the update so that we do not need to save $\mathbf{h}_{t-1}$ into replay buffers. The convergence and performance of such complexity reduction are guaranteed [14]. In addition, we clip the error term through *Huber Loss* to improve the stability of the algorithm.

The entire training process is described in Algo. 1.

5 EXPERIMENTAL SETUP

To evaluate the performance of our proposed approach, we carried out experiments on the Taobao Search platform, which is one of the largest E-commerce search services in the world, with over 1 billion user clicks every day. All models in this paper adopt the online-learning paradigm and are trained on the hourly real-time search log streaming data.

The online experiment methodology we adopted is called bucket testing, also known as A/B test. In the bucket testing system of Taobao Search, several test buckets are set up. And all users are randomly hashed into these buckets based on their user ids. Each bucket has the same number of users, and the same distribution of users. Then each algorithm is deployed on one bucket. The performance of an algorithm is estimated using the metrics calculated on the bucket it is deployed on. In this paper, our online bucket testing lasted for two weeks, a period long enough to ensure the statistical stability of our test results.

5.1 Features and Parameter Settings

We aggregated two verticals, namely topic and group verticals, into the results of product search. For each RL policy in our model, the components of state s (i.e. search request x, search source results $\mathcal{SR}(x)$, etc.) are represented by a number of features, which mainly contains 1) the query features: including the word-segmented tokens of the query; 2) user features: including user gender, age, items user clicked in the previous page; 3) page features: the current page number; 4) source features obtained from each search services, including the type and title of a topic group or a blog post.

The parameter settings for the high-level source selector and low-level item presenter are provided in Table 2.

Hyper-parameter	Setting	
	Source selector	Item presenter
Learning rate	1e-2	1e-4
Optimization algorithm	RMSProp	RMSProp
Memory D size	5e4	5e5
Mini-batch size	32	32
Target θ^- update period C	1e3	1e4
State feature vector size	48	56
Hidden layer size	28	24
RNN layer size	16	12
Discount factor γ	0.95	
Weight factor λ in Eq. 2	-	0.3
Clipped to δ in Eq. 2	-	3

Table 2: Hyper-parameter settings

5.2 Baselines and Evaluation Metrics

In order to show the effectiveness of our HRL model, we implemented several aggregation methods for comparison, which can be grouped into two categories: *rule-based* and *learning to aggregate* methods, while the latter can be separated into two types: *one-stage* and *two-stage* methods according to the process flow. The *one-stage* method is introduced to verify the hierarchical decomposition in the aggregate search, and the *two-stage* method is carried out to validate the RL formulation in each subtask. The details of our baselines are introduced as follows:

Rule: a rule-based method, where the rule is manually made for search aggregation according to abundant experience in search product design. The rule claims that the first page should not present aggregated results, and the topic and blog verticals will be alternately displayed at fixed positions from the second page when a topic group or a blog post is available from backends. In a page, the rule displays a topic group at the 4th position, and a blog post at the 9th position. Note that this simple rule is the benchmark for comparison with other methods.

Flat RL: a one-stage method. In this paradigm, the aggregation task is simplified into one stage: for each aggregation request, just choose the optimal template from a set of predefined templates. Seven templates are used in this method and are provided in Fig. 6. Each template clarifies exactly what to fill in each slot. These templates are designed by human experts familiar with Taobao Search to ensure good customer experience. The aggregation problem is solved by a single DQN agent. The state representation and the reward of this DQN are the same with the high-level state and the extrinsic reward defined in subsection 4.1 respectively.

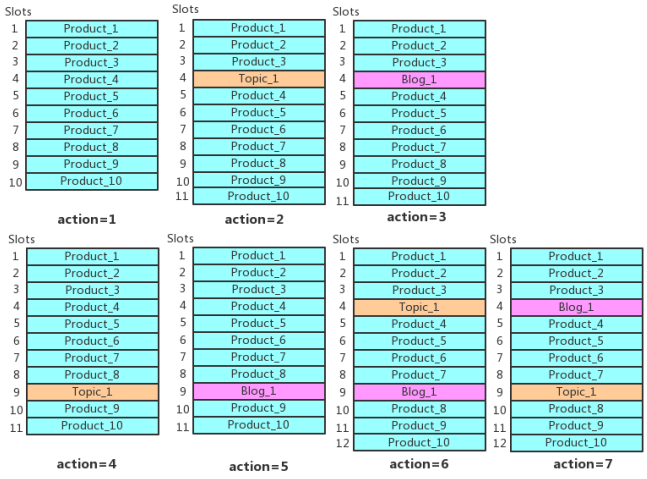


Figure 6: The SERP templates for our 1-stage DQN baseline.

BC+RM: a two-stage method similar to [28]. For *vertical selection*, it uses a binary classifier for each source to decide whether to present this source in a page. The binary classifiers are Neural Networks with 3 hidden layers. The training data for the binary classifiers come from online search log, where each page view is a sample. For a vertical in the page, if it is clicked by a user, then

Method	Topic group						Blog post					
	CTR		ADT		COV		CTR		ADT		COV	
	value (1e-2)	gain	value	gain	value (1e-2)	gain	value (1e-2)	gain	value	gain	value (1e-2)	gain
Rule	5.24	-	10.56	-	5.60	-	3.43	-	75.78	-	6.47	-
Flat RL	6.08	15.99%	10.62	0.54%	5.98	6.79%	3.92	14.29%	76.69	1.20%	6.76	4.48%
BC+RM	5.59	6.76%	10.68	1.09%	6.03	7.68%	3.69	7.58%	76.99	1.59%	6.75	4.33%
BC+RL	5.60	6.77%	10.66	0.92%	6.00	7.32%	3.73	8.75%	76.69	1.21%	6.82	5.41%
RL+RM	5.52	5.29%	10.75	1.75%	6.09	8.75%	3.60	6.12%	76.46	0.91%	6.87	6.18%
HRL	7.34	**40.07**%	10.65	0.89%	5.95	6.25%	4.25	**23.91**%	76.53	0.98%	6.86	6.03%

Table 3: Metrics of online bucket testing on two verticals.

it is a positive sample, otherwise a negative sample. For *vertical presentation*, it uses a 3-hidden-layer Neural Network regression model to score each item. The regression target of the regression model is the same with the intrinsic reward in Eq. 2.

BC+RL: a two-stage method similar to *HRL*, which replaces the high-level source selector with multiple binary classifiers as described in *BC+RM*, but the RL policy for the item presenter remains unchanged.

RL+RM: a two-stage method similar to *HRL*, which replaces the low-level item presenter with a regression model as described in *BC+RM*, but the RL policy for the source selector remains unchanged.

Note that all *learning to aggregate* methods were pre-trained on offline data by *behavioral cloning* prior to online experiments, so as to mitigate the slow learning and poor performance in the early stage of online training. The offline data, which contain 60,000 search sessions covering a broad query topics with an average of 11.3 items per page and 13.4 pages per session, were collected from the user search logs on the Taobao platform.

To evaluate the performance of the algorithms, we use some common evaluation metrics in search aggregation, including:

Click Through Rate (CTR): the ratio of users who click on a specific vertical to the number of total users who view the vertical on a page.

Average Dwell Time (ADT): the mean value of a user's dwell time in seconds on a specific vertical.

Coverage (COV): the ratio of the number of slots occupied by a specific vertical to the total number of slots.

Moreover, we also include an important metric in the E-commerce industry: the *Gross Merchandise Volume* (GMV) of each search service, which is the total revenue of sales made by the product search, or induced by a vertical in SERPs within a fixed time period.

6 RESULTS

6.1 Online Performance

We show the results of online bucket testing in terms of CTR, ADT and COV of different verticals in Table 3. Among these three metrics, the most important one is CTR, which indicates whether users clicked the verticals in the aggregated SERP. For each metric in Table 3, we present its value in the "value" column, and its relative improvement over the *Rule* baseline in the "gain" column. The results in Table 3 show that all the baselines and our HRL model achieve similar improvement rate on ADT and COV. However, it is worth noting that the result on CTR differs remarkably among all the methods.

In general, **HRL** achieves much better CTR improvement than all the baselines on both topic and blog verticals. The comparison to **Flat RL** shows that our hierarchical decomposition of the RL process is very effective, achieving better CTR from the historical user behaviors across different pages as well as the immediate user feedback in the current page. And the result also indicates that it is more sensible to allow a model to choose from any positions when displaying heterogeneous items, rather than restrict the positions where a vertical can be presented in a page. While the *one-stage* method can only use limited aggregation templates to meet the trade-off between the model performance and the exponential number growth of templates due to the number of sources, the hierarchical decomposition of *HRL* overcomes the exponentially large action space problem and allows the agent to explore all possible source combination and item permutation.

The comparisons to *BC+RM*, *BC+RL*, *RL+RM* show that the sequential RL modeling in both high-level and low-level aggregation subtasks is critical for a good performance. A common feature of these discriminant models is that they all follow the vertical selection and presentation pipeline, but make point-wise decisions, without the passing of states between sequential decisions as in *HRL*. **BC+RL** considers the user feedback only based on the current page, while *HRL* tracks the user behaviors from page to page to understand user intents. The result verifies that it is essential to capture sequential patterns of user behavior in aggregated E-commerce search. Similarly, the comparison to **RL+RM** demonstrates that *HRL* also consistently outperforms the baseline that simply ranks items from heterogeneous sources since it suffers from relevance ranking issue by giving scores to each vertical, which concludes that our slot filling process aggregates a more reasonable SERP.

Another important metric, the GMV of the aggregated E-commerce search, that we care about is shown in Table 4. It should be noted that, due to Alibaba's business policy, we temporarily cannot expose the absolute values of GMV. Hence, we report relative GMV increase over *Rule* instead, and this will not affect performance comparison. Because the GMV of aggregated search is the total merchandised volume of all sources, we report not only the GMV of topic and blog verticals, but also the GMV of product search in Table 4.

Method	Topic group	Blog post	Products
Flat RL	38.4%	12.31%	-0.47%
BC+RM	19.32%	5.83%	-0.41%
BC+RL	20.78%	6.00%	-0.37%
RL+RM	17.60%	5.54%	-0.20%
HRL	**56.49%**	**16.93%**	**0.56%**

Table 4: The relative GMV increase over *Rule*. The GMV of a vertical is computed through the sales value guided by the presented items of that vertical.

As shown in Table 4, *HRL* is the only method that increases the GMV of product search. Although other baselines achieve GMV increase on the topic and blog verticals, their GMVs on product search all decrease. For example, *Flat RL* brings a marked GMV improvement on topic and blog verticals, at the expense of the largest GMV drop on product search though. This means that *HRL* provides the best user experience in its aggregated SERP, because even a growth on the COV of topic and blog verticals does not reduce users' purchases in product search. Moreover, *HRL* also achieves the best GMV increase on the topic and blog verticals, leading to the best overall GMV increase among all methods.

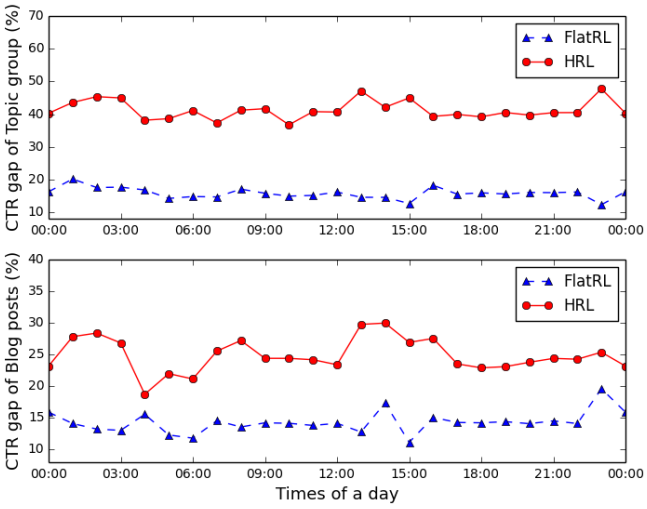


Figure 7: The relative CTR increase over *Rule* in different hours. Each mark indicates the result in the time span between the current time and next full hour.

To provide more insights on the online experiments, we group the two-week statistics by different hours of a day to observe the performance variation over time. For simplicity, we only compare the relative CTR increase of the two most competitive methods: *Flat RL* and *HRL*. Fig. 7 shows the trend of their CTR increase over the baseline *Rule* during different time periods of a day. It can be seen that the CTR increase does not variate too much at different hours of a day, and *HRL* has consistently better CTR improvement than *Flat RL*.

6.2 User Evaluation

In order to study the user experience change caused by our HRL aggregation method, we invite 50 users of Taobao Search to evaluate the quality of our aggregated result against that of the rule-based baseline: *Rule*, which is the old online aggregation method of Taobao Search and has been online serving for several years. The evaluation is performed on a fixed query set: 200 popular queries covering 13 salient product categories in Taobao. This query set is selected by third-party experts on Taobao platform. A user is asked to search each query in both our and the old aggregation environments simultaneously, without knowing which environment is the old one. And the aggregated results of the two environments, designated by A and B, are presented to the user side-by-side. The user can browse the results, click items, scroll to the next page, and so on. And the user provides a grade from {"System A better", "Same/cannot judge", "System B better"} on each vertical for this query. Then we translate the user's grade to {"New System better", "Same/cannot judge", "Old System better"}, where the "New System" refers to *HRL* and the "Old System" refers to *Rule*. The distribution of user grades on each vertical is shown in Table 5.

Vertical	New system better	Same	Old system better
Topic	30.35%	67.98%	1.67%
Blog	26.14%	72.53%	1.33%

Table 5: The distributions of user evaluation grades.

The results in Table 5 show that user experience grades are the same on about 70% of the queries. In spite of this, *HRL* provides better user experience than the baseline most of the times on the rest 30% of the queries.

To understand how *HRL* improves user experience, every time when a user thinks the new system is better, we inquire about his reason for this preference. And we also ask them to group their reasons into three groups as follows:

- Better **behavior relevance**: In *HRL*, the presented verticals are more relevant to the user's previous behaviors.
- Better **query relevance**: In *HRL*, the presented verticals are more relevant to the query.
- **Others**: other reasons.

Vertical	Behavior relevance	Query relevance	Others
Topic	42%	33%	25%
Blog	37%	35%	28%

Table 6: Distributions of reasons why new system is better.

The results in Table 6 show that *HRL* contributes to both user behavior relevance and query relevance. We present two cases here to give some clue that how user experience is improved.

A case for better user behavior relevance is as follows: a user issues the query "running shoes", and clicks several products with the same brand: "Nike" on the first page, then on the next page the topic group titled "Nike shops with discounts" is presented at the

second position in our HRL aggregated result, which is very relevant to the user's behavior. In contrast, no topic group is presented by the baseline in the aggregated result on the next page. In *HRL*, user behaviors in previous pages are encoded into the RNN layer hidden states in the Q network shown in Fig. 5. So user behaviors in previous pages are passed on to the next page for decision making in *HRL*, which helps perceive user behavior relevance. Besides, *HRL* can also capture sequential patterns of user behavior by learning from the long-delayed user feedback across the whole user search session.

A case for better query relevance is as follows: a user issues the query "women's long dress autumn Vero Moda", which is very specific in brand, season and style. The blog post titled "Introduction to the world of women's dresses", which gives a general introduction to women's dresses, is not presented in the aggregated result of *HRL*, whereas it is presented in the baseline's aggregated result. The user prefers our aggregated result because she thinks the blog post is too general thus not meeting her specific needs. With query and blog title features as input, *HRL* is able to identify that the general introduction of the blog post is not relevant enough to the query she inputs.

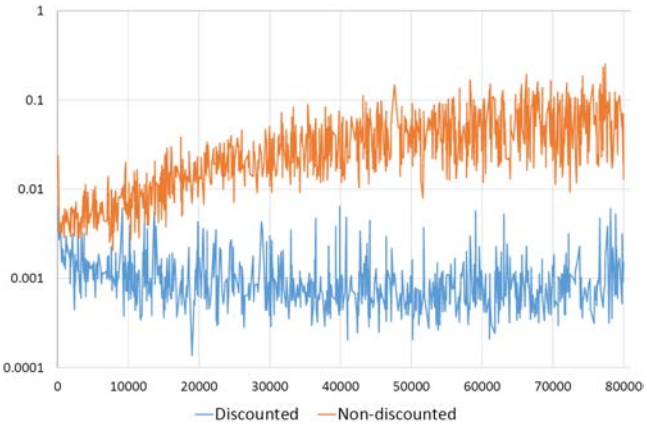


Figure 8: Loss curve of high-level source selector between different training strategies. Blue curve for strategy (I) and orange curve for strategy (II). The horizontal axis is the training iteration, and the vertical axis is the loss value in log scale.

6.3 Training Analysis

To offer a detail training analysis of HRL, we investigate the training procedure of the high-level RL process that takes multi-step actions. Two strategies are designed for the source selector. (I) One is the proposed one in section 4.1 & 4.4 that the extrinsic reward is the mean value of the intrinsic return from the item presenter, i.e. the same as Eq. 1, and the optimization target of Q network (y_t^e in Eq. 5) is discounted by γ^l. (II) The other is to regard the high-level RL process as a simple RL process that takes one-step actions. Under this setting, the extrinsic reward is directly equivalent to the mean value of the intrinsic rewards, i.e. $r_t^e = \frac{1}{l}\sum_{k=0}^{l-1} r_{t+k}^i$, then the training target of source selector policy in Eq. 5 is similar to that in Eq. 6: $y_t^e = r + \gamma Q_\mu(s', \arg\max_{o'} Q_\mu(s', o'; \theta_t^e); \theta_t^{e-})$.

The learning curves of the two strategies are demonstrated in Fig. 8, where the blue and orange curves correspond to strategy (I) and (II) respectively. It is clear that the proposed strategy (I) converges well as the training procedure proceeds while the strategy (II) diverges dramatically. This result shows that it is necessary to handle a multi-step RL process in a proper way to guarantee its convergence. From another perspective, the strategy (I) puts more weight on the items presented in the top slots, which is consistent with the position bias in most search area that top-ranked search results attract more clicks, so the agent can better understand the sequential patterns of user behavior and learn from such data effectively.

7 CONCLUSION AND DISCUSSION

We have studied the search aggregation problem in a large E-commerce search service and proposed an effective HRL aggregation method. Different from web search aggregation, the E-commerce aggregation needs to be performed for each page, leading to multiple and sequential aggregation decisions for one query. In light of this, we propose a hierarchical aggregation model for E-commerce search aggregation. Our HRL model naturally fits the two-stage decomposition of the aggregated search task: a high-level RL for source selection and a low-level RL for item presentation. The source selector models user interactions on aggregated results across pages to capture the sequential patterns of user behavior, and the item presenter formulates a slot filling process that sequentially presents the heterogeneous items to avoid the relevance ranking issue. In this manner, the HRL model can fully learn from user feedback in the current page as well as in the entire search session.

We perform online bucket testing on Taobao Search platform and compare our HRL method with several baselines. The results show that our method contributes a substantial improvement over the baselines for all verticals. Moreover, it also boosts the GMV of product search and achieves the best overall GMV among all the models. User study also demonstrates that it improves user experience by displaying better aggregated results that are relevant to user behaviors and needs. Note that our method can easily combine new verticals into an aggregated search system.

One limitation of the proposed algorithm may lie in the automatic offline evaluation. All we can get from the offline data is the records of user behaviors on the presented results, so it is difficult to infer the user feedback on the unseen SERPs without full annotations of the heterogeneous item relevance as we train our model in a RL setting. However, sufficient experiments including bucket testing and user study indicate that our method indeed shows its effectiveness on the search aggregation. Future research will be focused on the application of our HRL model in different E-commerce verticals and other scenarios.

ACKNOWLEDGEMENTS

This work was jointly supported by the National Key R&D Program of China (Grant No. 2018YFC0830200) and the National Science Foundation of China (Grant No.61876096/61332007). We would also like to thank Mr. Xiaoyi Zeng, our colleague in Alibaba, for constant support and encouragement.

REFERENCES

[1] Jaime Arguello. 2017. Aggregated search. *Foundations and Trends in Information Retrieval* 10, 5 (2017), 365–502.

[2] Jaime Arguello, Fernando Diaz, and Jamie Callan. 2011. Learning to aggregate vertical results into web search results. In *Proc. 20th ACM Int. Conf. Information and Knowledge Management*. 201–210.

[3] Jaime Arguello, Fernando Diaz, Jamie Callan, and Ben Carterette. 2011. A methodology for evaluating aggregated search results. In *Proc. 33rd European Conf. Information Retrieval*. 141–152.

[4] Jaime Arguello, Fernando Diaz, Jamie Callan, and Jean-Francois Crespo. 2009. Sources of evidence for vertical selection. In *Proc. 32nd Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 315–322.

[5] Andrew G Barto and Sridhar Mahadevan. 2003. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems* 13 (2003), 341–379.

[6] Horatiu Bota, Ke Zhou, Joemon M Jose, and Mounia Lalmas. 2014. Composite retrieval of heterogeneous web search. In *Proc. 23rd Int. Conf. World Wide Web*. 119–130.

[7] Marc Bron, Jasmijn Van Gorp, Frank Nack, Lotte Belice Baltussen, and Maarten de Rijke. 2013. Aggregated search interface preferences in multi-session search tasks. In *Proc. 36th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 123–132.

[8] Danqi Chen, Weizhu Chen, Haixun Wang, Zheng Chen, and Qiang Yang. 2012. Beyond ten blue links: Enabling user click modeling in federated web search. In *Proc. 5th ACM Int. Conf. Web Search and Data Mining*. 463–472.

[9] Ye Chen, Yiqun Liu, Ke Zhou, Meng Wang, Min Zhang, and Shaoping Ma. 2015. Does vertical bring more satisfaction?: Predicting search satisfaction in a heterogeneous environment. In *Proc. 24th ACM Int. Conf. Information and Knowledge Management*. 1581–1590.

[10] Aleksandr Chuklin, Anne Schuth, Katja Hofmann, Pavel Serdyukov, and Maarten De Rijke. 2013. Evaluating aggregated search using interleaving. In *Proc. 22nd ACM Int. Conf. Information and Knowledge Management*. 669–678.

[11] Fernando Diaz. 2009. Integration of news content into web results. In *Proc. 2nd ACM Int. Conf. Web Search and Data Mining*. 182–191.

[12] Jun Feng, Heng Li, Minlie Huang, Shichen Liu, Wenwu Ou, Zhirong Wang, and Xiaoyan Zhu. 2018. Learning to collaborate: Multi-scenario ranking via multi-agent reinforcement learning. In *Proc. 27th Int. Conf. World Wide Web*. 1939–1948.

[13] Hado van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double Q-Learning. In *Proc. 30th AAAI Conf. Artificial Intelligence*. 2094–2100.

[14] Matthew Hausknecht and Peter Stone. 2015. Deep recurrent Q-Learning for partially observable MDPs. In *Proc. 29th AAAI Conf. Artificial Intelligence, Fall Symp. Series, Sequential Decision Making for Intelligent Agents*. 29–37.

[15] Dzung Hong, Luo Si, Paul Bracke, Michael Witt, and Tim Juchcinski. 2010. A joint probabilistic classification model for resource selection. In *Proc. 33rd Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 98–105.

[16] Luo Jie, Sudarshan Lamkhede, Rochit Sapra, Evans Hsu, Helen Song, and Yi Chang. 2013. A unified search federation system based on online user feedback. In *Proc. 19th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*. 1195–1203.

[17] Arlind Kopliku, Karen Pinel-Sauvagnat, and Mohand Boughanem. 2014. Aggregated search: A new information retrieval paradigm. *Comput. Surveys* 46, 3 (2014), 41.

[18] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. 2016. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Proc. 30th Annu. Conf. Neural Information Processing Systems*. 3675–3683.

[19] Or Levi, Ido Guy, Fiana Raiber, and Oren Kurland. 2018. Selective cluster presentation on the search results page. *ACM Transactions on Information Systems* 36, 3 (2018), 28.

[20] Zeyang Liu, Yiqun Liu, Ke Zhou, Min Zhang, and Shaoping Ma. 2015. Influence of vertical result in web search examination. In *Proc. 38th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 193–202.

[21] Bo Long and Yi Chang. 2014. *Relevance ranking for vertical search engines*. Morgan Kaufmann Publishers Inc.

[22] Ilya Markov, Eugene Kharitonov, Vadim Nikulin, Pavel Serdyukov, Maarten De Rijke, and Fabio Crestani. 2014. Vertical-aware click model-based effectiveness metrics. In *Proc. 23rd ACM Int. Conf. Information and Knowledge Management*. 1867–1870.

[23] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. In *Proc. 27th Annu. Conf. Neural Information Processing Systems, Deep Learning Workshop*.

[24] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[25] Harrie Oosterhuis and Maarten de Rijke. 2018. Ranking for relevance and display preferences in complex presentation layouts. In *Proc. 41st Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 845–854.

[26] Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam-Fai Wong. 2017. Composite Task-Completion Dialogue Policy Learning via Hierarchical Deep Reinforcement Learning. In *Proc. 22nd Conf. Empirical Methods in Natural Language Processing*. 2231–2240.

[27] Ashok Kumar Ponnuswami, Kumaresh Pattabiraman, Desmond Brand, and Tapas Kanungo. 2011. Model characterization curves for federated search using click-logs: predicting user engagement metrics for the span of feasible operating points. In *Proc. 20th Int. Conf. World Wide Web*. 67–76.

[28] Ashok Kumar Ponnuswami, Kumaresh Pattabiraman, Qiang Wu, Ran Gilad-Bachrach, and Tapas Kanungo. 2011. On composition of a federated web search result page: Using online users to provide pairwise preference for heterogeneous verticals. In *Proc. 4th ACM Int. Conf. Web Search and Data Mining*. 715–724.

[29] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. 2015. Universal value function approximators. In *Proc. 32nd Int. Conf. Machine Learning*. 1312–1320.

[30] Luo Si and Jamie Callan. 2003. Relevant document distribution estimation method for resource selection. In *Proc. 26th Int. ACM SIGIR Conf. Research and Development in Informaion Retrieval*. 298–305.

[31] Shanu Sushmita, Hideo Joho, Mounia Lalmas, and Robert Villa. 2010. Factors affecting click-through behavior in aggregated search interfaces. In *Proc. 19th ACM Int. Conf. Information and Knowledge Management*. 519–528.

[32] Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction*. MIT press.

[33] Richard S Sutton, Doina Precup, and Satinder Singh. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112 (1999), 181–211.

[34] Ryuichi Takanobu, Tianyang Zhang, Jiexi Liu, and Minlie Huang. 2019. A Hierarchical Framework for Relation Extraction with Reinforcement Learning. In *Proc. 33rd AAAI Conf. Artificial Intelligence*.

[35] Chen Tessler, Shahar Givony, Tom Zahavy, Daniel J Mankowitz, and Shie Mannor. 2017. A deep hierarchical approach to lifelong learning in Minecraft. In *Proc. 31st AAAI Conf. Artificial Intelligence*. 1553–1561.

[36] Gilad Tsur, Yuval Pinter, Idan Szpektor, and David Carmel. 2016. Identifying web queries with question intent. In *Proc. 25th Int. Conf. World Wide Web*. 783–793.

[37] Lauren Turpin, Diane Kelly, and Jaime Arguello. 2016. To blend or not to blend?: Perceptual speed, visual memory and aggregated search. In *Proc. 39th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 1021–1024.

[38] Chao Wang, Yiqun Liu, Min Zhang, Shaoping Ma, Meihong Zheng, Jing Qian, and Kuo Zhang. 2013. Incorporating vertical results into search click models. In *Proc. 36th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 503–512.

[39] Yue Wang, Dawei Yin, Luo Jie, Pengyuan Wang, Makoto Yamada, Yi Chang, and Qiaozhu Mei. 2016. Beyond ranking: Optimizing whole-page presentation. In *Proc. 9th ACM Int. Conf. Web Search and Data Mining*. 103–112.

[40] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. 2016. Dueling Network Architectures for Deep Reinforcement Learning. In *Proc. 33rd Int. Conf. Machine Learning*. 1995–2003.

[41] Long Xia, Jun Xu, Yanyan Lan, Jiafeng Guo, Wei Zeng, and Xueqi Cheng. 2017. Adapting Markov decision process for search result diversification. In *Proc. 40th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 535–544.

[42] Junqi Zhang, Yiqun Liu, Shaoping Ma, and Qi Tian. 2018. Relevance Estimation with Multiple Information Sources on Search Engine Result Pages. In *Proc. 27th ACM Int. Conf. Information and Knowledge Management*. 627–636.

[43] Ke Zhou, Ronan Cummins, Mounia Lalmas, and Joemon M Jose. 2012. Evaluating aggregated search pages. In *Proc. 35th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 115–124.

[44] Ke Zhou, Ronan Cummins, Mounia Lalmas, and Joemon M Jose. 2013. Which vertical search engines are relevant?. In *Proc. 22nd Int. Conf. World Wide Web*. 1557–1568.

[45] Tao Zhuang, Wenwu Ou, and Zhirong Wang. 2018. Globally Optimized Mutual Influence Aware Ranking in E-Commerce Search. In *Proc. 27th Int. Joint Conf. Artificial Intelligence*. 3725–3731.

Towards Neural Mixture Recommender for Long Range Dependent User Sequences

Jiaxi Tang*†
School of Computing Science, Simon Fraser University, Canada
jiaxit@sfu.ca

Francois Belletti†
Google
belletti@google.com

Sagar Jain
Google
sagarj@google.com

Minmin Chen
Google
minminc@google.com

Alex Beutel
Google
alexbeutel@google.com

Can Xu
Google
canxu@google.com

Ed H. Chi
Google
edchi@google.com

ABSTRACT

Understanding temporal dynamics has proved to be highly valuable for accurate recommendation. Sequential recommenders have been successful in modeling the dynamics of users and items over time. However, while different model architectures excel at capturing various temporal ranges or dynamics, distinct application contexts require adapting to diverse behaviors.

In this paper we examine how to build a model that can make use of different temporal ranges and dynamics depending on the request context. We begin with the analysis of an anonymized Youtube dataset comprising millions of user sequences. We quantify the degree of long-range dependence in these sequences and demonstrate that both short-term and long-term dependent behavioral patterns co-exist. We then propose a neural Multi-temporal-range Mixture Model (*M3*) as a tailored solution to deal with both short-term and long-term dependencies. Our approach employs a mixture of models, each with a different temporal range. These models are combined by a learned gating mechanism capable of exerting different model combinations given different contextual information. In empirical evaluations on a public dataset and our own anonymized YouTube dataset, *M3* consistently outperforms state-of-the-art sequential recommendation methods.

CCS CONCEPTS

• **Information systems** → **Personalization**; **Recommender systems**; • **Computing methodologies** → **Neural networks**.

*Work done while interning at Google
†The two authors contributed equally to this work

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313650

KEYWORDS

Recommender System; User Modeling; Sequential Prediction

ACM Reference Format:
Jiaxi Tang, Francois Belletti, Sagar Jain, Minmin Chen, Alex Beutel, Can Xu, and Ed H. Chi. 2019. Towards Neural Mixture Recommender for Long Range Dependent User Sequences. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313650

1 INTRODUCTION

Across the web and mobile applications, recommender systems are relied upon to surface the right items to users at the right time. Some of their success can be attributed to advances in modeling as well as the ingenuity of applied researchers in adopting and inventing new techniques to solve this important problem [19, 30, 44, 46]. Fundamentally, recommenders match users in a particular context with the best personalized items that they will engage with [17, 34]. In order to do this effectively, recommenders need to understand the users, typically based on their previous actions, and to understand items, most often based on the users who previously interacted with them. This presents a fundamental challenge: users' preferences and items' perception are continuously changing over time, and the recommender system needs to understand these dynamics.

A significant amount of research has recognized forms of this problem. Sequence information has been generally shown to improve recommender performance [18, 45]. Koren [29] identified multiple user and item dynamics in the Netflix Prize competition, and incorporated these dynamics as biases in a collaborative filtering model. [7, 56] demonstrated that Recurrent Neural Networks (RNNs) could learn many of these patterns, and likewise [20] demonstrated that RNNs can learn patterns in individual sessions. Despite these successes, RNNs are known to have difficulties learning long-range dependent temporal patterns [4].

We observe and study an open challenge for such sequential recommender systems: *while different applications and contexts require different temporal ranges and patterns, model architectures are typically designed to capture a particular temporal dynamic.* For

example, when a user comes to the Amazon home page they may be looking for something new to buy or watch, but on an item specific page they may be looking for other items that are closely related to recently browsed items. *How can we design a model that works, simultaneously, across all of these contexts and temporal ranges?*

Contributions: We address the issue of providing a single model adapted to the diversity of contexts and scales of temporal dependencies in sequential recommendations through data analysis and the design of a Multi-temporal-range Mixture Model, or *M3* for short. We make the following contributions to this problem:

- **Data-driven design:** We demonstrate that in real world recommendation tasks there are significant long-range temporal dependencies in user sequence data, and that previous approaches are limited in their ability to capture those dynamics. M3's design is informed by this quantitative analysis.
- **Multi-range Model:** We offer a single model, M3, which is a mixture model consisting of three sub-models (each with a distinct manually designed architecture) that specialize in capturing different ranges of temporal dependencies. M3 can learn how to dynamically choose to focus on different temporal dynamics and ranges depending on the application context.
- **Empirical Benefits and Interpretability:** We show on both public academic and private data that our approach provides significantly better recommendations. Further, using its interpretable design, we analyze how M3 dynamically switches between patterns present at different temporal ranges for different contexts, thus showing the value in enabling context-specific multi-range modeling. Our private dataset consists in anonymized user sequences from YouTube. To the best of our knowledge this paper is the first to focus on sequential patterns in such a setting.

2 RELATED WORK

Before we describe our sequential recommendation problem and provide the quantitative insights orienting the design of a novel sequential neural model based on a mixture of models, we briefly introduce the reader to some key pre-existing related work.

Matrix factorization [30] is among the most popular techniques used in classic recommender research, in which a similarity score for each user-item pair is learned by building latent user and item representations to recover historical user-item interactions. The predicted similarity score is then used to indicate the *relatedness* and find the most relevant items to recommend to a user. Followup work on introducing auxiliary sources of information beyond user-item interactions have been proven successful [11], especially for cold-start problems. Pazzani and Billsus [39] use item content (*e.g.,* product image, video's visual/audio content, etc) to provide a better item representation.

Neural Recommender Systems. Deep neural networks have gained tremendous success in the fields of Computer Vision [28, 31] and Natural Language Processing [2, 36]. In recommender research, we have witnessed growing interest of using deep neural networks to model complex contextual interactions between user and items, which surpass classic factorization-based methods [30, 43]. Auto-encoders[33, 47, 57] constitute an early example of success for a framework based on neural networks to better infer un-observed user/item affinities in a recommendation problem. He et al. [19] also proved that traditional Collaborative Filtering methods can be effectively generalized by a deep neural network. Besides,

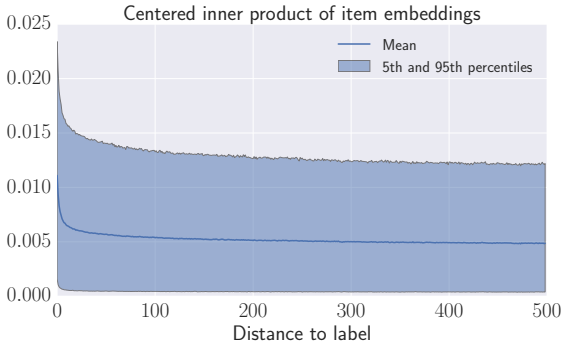


Figure 1: Trace of covariance (i.e. centered inner product similarity) of item embeddings between the last item in user sequence and the item located L steps before (100K samples).

For the specific problem of sequential recommendation using neural networks, RNNs [20, 56] have become a common choice. Other methods based on Convolutional Neural Networks (CNNs) [51, 59], Attention Models [61] have also been explored. While most of existing methods developed for sequential recommendations perform well [18, 20, 45, 49, 51], they still have some limitations when dealing with long user sequences found in production recommender systems. As we shall discuss in Section 3, such approaches do not scale well to very long sequences.

Mixture of Models. Despite being simpler and more elegant, monolithic models are in general less effective than mixtures of models to take advantage of different model capacities and architectural biases. Gehring et al. [15] used an RNN in combination with an attention model for neural machine translation which provided a substantial performance gain. Pinheiro and Collobert [40] proposed to combine a CNN with an RNN for scene labeling. In the field of sequential recommendation, an earlier work on mixing of a Latent Factor Model (LFM) and a Factorized Markov Chain (FMC) has been shown to offer superior performance than each individual one [45]. A similar trend was observed in [18, 60]. While sharing similar spirit to these aforementioned methods, we designed our mixture of models with the goal to model varying ranges of dependence in long user sequences found in real production systems. Unlike model ensembles [13, 62, 63] that learn individual models separately prior to ensembling them, a mixture of models learns individual models as well as combination logic simultaneously.

3 QUANTIFYING LONG RANGE TEMPORAL DEPENDENCIES IN USER SEQUENCES

We first present some findings on our anonymized proprietary dataset which uncover properties of behavioral patterns as observed in extremely-long user-item interaction sequences. We then pinpoint some limitations of existing methods which motivate us to design a better adapted solution.

Sequential Recommendation Problem: We consider a sequential recommendation problem [18, 20, 45, 49, 51] defined as follows: assume we have a set of users $u \in \mathcal{U}$, a set of items $v \in \mathcal{V}$, and for each user we have access to a sequence of user historical events $\mathcal{E}^u = (e_1^u, e_2^u, \cdots)$ ordered by time. Each e_τ^u records the item consumed at time τ as well as context information of the interaction. Given the historical interactions, our goal is to recommend to each user a subset of items in order to maximize a performance metric such as user satisfaction.

3.1 Hidden Values in Long User Sequences

We now describe how we developed a better understanding of long user sequences in our proprietary dataset through quantitative data exploration. To quantify how past events can influence a user's current behavior in our internal dataset, *i.e.* measure the range of temporal dependency within a sequence of events, one can examine the covariance matrix of two events L-step apart [5, 41], where step denotes the relative order of events within sequence. In particular, we look at the trace of the covariance matrix as a measurement of dependency:

$$\mathrm{Dep}_L = \mathrm{tr}\left(\mathrm{Cov}\left(Q_{e_N}, Q_{e_{N-L}}\right)\right)$$

where e_N is the item in last event in a logged user/item interaction sequence and e_{N-L} is the item corresponding to the interaction that occurred L time steps before the last event. We focus on the trace of the covariance matrix as it equals the sum of the eigenvalues of the covariance matrix and its rate of decay is therefore informative of the rate of decay of these eigenvalues as a whole.

We utilize the embeddings Q that have been learned by a pre-existing model—in our case an RNN-based sequential recommender which we describe later as one of M3's sub-models. Dep_L here measures the similarity between the current event and the event L steps back from it. To estimate Dep_L for a particular value of L we employ a classic empirical averaging across user sequences in our dataset. From Figure 1, we can extract multiple findings:

- The dependency between two events decreases as the time separating their consumption grows. This suggests that recent events bear most of the influence of past user behavior on a user's future behavior.
- The dependency slowly approaches zero even as the temporal distance becomes very large (*i.e.* $L > 100$). The clear hyperbolic-decay of the level of temporal dependencies indicates the presence of long-range-dependent patterns existing in user sequences [41]. In other words, a user's past interactions, though far from the current time step, still cumulatively influence their current behavior significantly.

These findings suggest that users do have long-term preferences and better capturing such long-range-dependent pattern could help predicting their future interests. In further work, we plan to use off-policy correction methods such as [8, 16] to remove presentation bias when estimating correlations.

3.2 Limitations of Existing Sequential Models

The previous section has demonstrated the informational value of long-range temporal patterns in user sequences. Unfortunately, it is still generally challenging for existing sequential predictive models to fully utilize information located far into the past.

Most prior models have difficulties when learning to account for sequential patterns involving long-range dependence. Existing sequential recommenders with factorized Markov chain methods [18] or CNNs [51] arguably provide reliable sequential recommendation strategies. Unfortunately they are all limited by a short window of significant temporal dependence when leveraging sequential data to make a recommendation prediction. RNNs [7, 20, 24] and their variants [12, 42] are widely used in sequential recommendation. RNN-based models, though effective for short user sequences (*e.g.* short event sequences within a session), are challenged by long-range dependent patterns in long user sequences. Because of the way they iterate over sequential items [36] and their use of saturating non-linear functions such as tanh to propagate information through time, RNNs tend to have difficulties leveraging the information contained in states located far into the past due to gradient propagation issues [4, 38]. Even recent architectures designed to facilitate gradient propagation such as Gated Recurrent Unit [10] and Long-short Term Memory [21, 50] have also been shown to suffer from the same problem of not being able to provably account for long-range dependent patterns in sequences [4].

A second challenge in sequential recommendations is learning user latent factors P_u explicitly from data, which has been observed to create many difficulties [9, 18, 45, 51]. In the corresponding works, users' long-term preferences have been modeled through learning a set of latent factors P_u for each user. However, learning P_u explicitly is difficult in large-scale production systems. As the number of users is usually several magnitudes higher than the number of items, building such a large user vocabulary and storing the latent factors in a persistent manner is challenging. Also, the long-tail users (*a.k.a* cold users) and visitor users (*i.e.* users who are not logged in) could have much worse recommendations than engaged users [6].

3.3 Limitations of Single Monolithic Models

Figure 1 clearly indicates that although the influence of past user events on future interactions follows a significant decaying trend, significant predictive power can still be carried by events located arbitrarily far in the past. Very recent events (*i.e.* $1 \leq L \leq 10$) have large magnitude similarities with the current user behavior and this similarity depends strongly on the sequential order of related events. As the distance L grows larger, the informative power of previously consumed items on future user behavior is affected by more uncertainty (*e.g.* variance) and is less sensitive to relative sequential position. That is, the events from 100 steps ago and from 110 steps ago may have a generally similar influence on future user decisions regardless of their relative temporal location. Therefore, *for the kind of sequential signals we intend to leverage, in which different scales of temporal dependencies co-exist, it may be better to no longer consider a single model.* While simple monolithic models such as Deep Neural Network (DNN) with pooling and dropout [11, 55] are provably robust to noise, they are unfortunately not sensitive to sequential order (without substantial modifications). On the other hand, RNNs [7, 20] provide cutting-edge sequential modeling capabilities but they are heavily sensitive to noise in sequential patterns. Therefore, it is natural to choose a mixture of diverse models which would then complement each other to provide better overall predictive power.

4 MULTI-TEMPORAL-RANGE MIXTURE MODEL FOR LONG USER SEQUENCES

Motivated by our earlier analyses, we now introduce a novel method aimed at addressing the shortcoming of pre-existing approaches for long user/item interaction sequences: Multi-temporal-range Mixture Model (M3) and its two variants (M3R/M3C). For simplicity, we omit the superscripts related to users (*i.e.* e_τ^u will now be denoted e_τ) and use a single user sequence to describe the neural architecture we introduce.

4.1 Overview

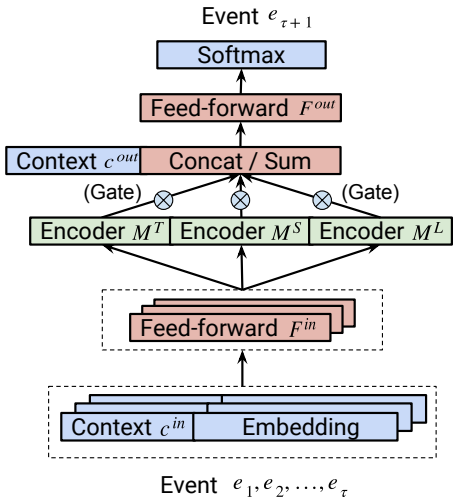


Figure 2: An overview of the proposed M3 model.

Figure 2 gives a general schematic depiction of M3. We will now introduce each part of the model separately in a bottom-up manner, starting from the inputs and progressively abstracting their representation which finally determines the model's output. When predicting the behavior of a user in the next event $e_{\tau+1}$ of their logged sequence, we employ item embeddings and context features (optional) from past events as inputs:

$$x_\tau = [Q_\tau \oplus c_\tau^{\text{in}}], \tag{1}$$

where $\oplus$ denotes the concatenation operator. To map the raw context features and item embeddings to the same high-dimensional space for future use, a feed-forward layer F^{in} is used:

$$Z_\tau^{\text{in}} = \{z_i^{\text{in}}\}_{i=1\cdots\tau} \text{ where } z_\tau^{\text{in}} = F^{\text{in}}(x_\tau) \tag{2}$$

here $z_\tau^{\text{in}} \in \mathbb{R}^{1\times d_{\text{in}}}$ represents the input processed at step τ and $Z_\tau^{\text{in}} \in \mathbb{R}^{\tau\times d_{\text{in}}}$ stands for the collection of all processed inputs before step τ (included). Either the identity function or a ReLU [37] can be used to instantiate the feed-forward layer F^{in}.

In the previous section, we assessed the limitations of using a single model on long user sequence. To circumvent the issues we highlighted, we employ in M3 three different sequence models (encoders) in conjunction, namely M^T, M^S and M^L, on top of the processed input Z_τ^{in}. We will later explain their individual architectures in details. The general insight is that we want each of these sub-models to focus on different ranges of temporal dependencies in user sequences to provide a better representation (*i.e.,* embedding) of the sequence. We want the sub-models to be architecturally diverse and address each other's shortcomings. Hence

$$SE_\tau^T = M^T(Z_\tau^{\text{in}}),\ SE_\tau^S = M^S(Z_\tau^{\text{in}}),\ SE_\tau^L = M^L(Z_\tau^{\text{in}}), \tag{3}$$

which yields three different representations, one produced by each of the three sequence encoders. The three different sub-model encoders are expected to produce outputs—denoted by d_{enc}—of identical dimension. By construction, each sequential encoder produces its own abstract representation of a given user's logged sequence, providing diverse latent semantics for the same input data.

Our approach builds upon the success of Mixture-of-Experts (MOE) model [23]. One key difference is that our 'experts' are constructed to work with different ranges of temporal dependencies, instead of letting the cohort of 'experts' specialize by learning from data. As shown in [48], heavy regularization is needed to learn different experts sharing the same architecture in order to induce specialization and prevent starvation when learning (only one expert performs well because it is the only one to learn which creates a self-reinforcing loop when learning with back-propagation).

Informed by the insights underlying the architecture of MOE models, we aggregate all sequence encoders' results by weighted-concatenate or weighted-sum, with weights G_τ computed by a small gating network. In fact, we concatenate the outputs with

$$SE_\tau = (G_\tau^T \times SE_\tau^T) \oplus (G_\tau^S \times SE_\tau^S) \oplus (G_\tau^L \times SE_\tau^L), \tag{4}$$

where $G_\tau \in \mathbb{R}^3$ corresponds to the outputs of our gating network. We can also aggregate outputs with a weighted-sum:

$$SE_\tau = (G_\tau^T \times SE_\tau^T) + (G_\tau^S \times SE_\tau^S) + (G_\tau^L \times SE_\tau^L). \tag{5}$$

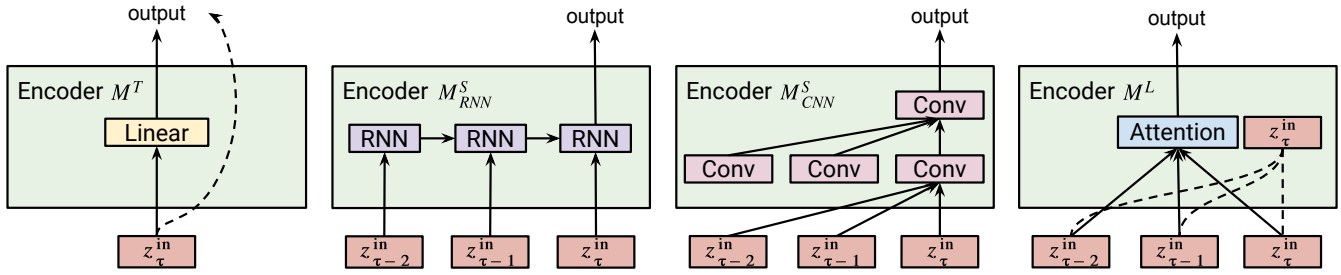
Note that there is no theoretical guarantee whether concatenation is better than summation or not. The choice of aggregation, as well as the choice of activation functions, is determined by observing a given model's performance from a validation set extracted from different datasets. Such a procedure is usual in machine learning and will help practitioners determine which variant of the model we propose is best suited to their particular application.

Because of its MOE-like structure, our model can adapt to different recommendation scenarios and provide insightful interpretability (as we shall see in Section 5). In many recommendation applications, some features annotate each event and represent the context in which the recommendation query is produced. Such features are for instance indicative of the page or device on which a user is being served a recommendation. After obtaining a sequence encoding at step τ (*i.e.* SE_τ), we fuse it with the annotation's context features (optional) and project them to the same latent space with another hidden feed-forward layer F^{out}:

$$z_\tau^{\text{out}} = F^{\text{out}}([SE_\tau \oplus c_\tau^{\text{out}}]) \tag{6}$$

where c^{out} is a vector encoding contextual information to use after the sequence has been encoded. Here the $z_\tau^{\text{out}} \in \mathbb{R}^{1\times d_{\text{out}}}$ is what we name *user representation*, it is computed based on the user's history as it has been gathered in logs. Finally, a user similarity score r_v is predicted for each item via an inner-product (which can be changed to another similarity scoring function):

$$r_v = z_\tau^{\text{out}} \cdot Q_v' \tag{7}$$



(a) Tiny-range encoder **(b) Short-range encoder (RNN)** **(c) Short-range encoder (CNN)** **(d) Long-range encoder**

Figure 3: The sequence encoders of *M3*. The solid lines are used to denote the data flow. The dotted line in (a) means an identity copy whereas in (d) it means the interaction of attention queries and keys.

where Q'_v is a vector representing the item. For a given user, item similarity scores are then normalized by a softmax layer which yields a recommendation distribution over the item vocabulary. After training M3, the recommendations for a user at step τ are served by sorting the similarity scores r_v obtained for all $v \in \mathcal{V}$ and retrieving the items associated with the highest scores.

4.2 Three Different Range Encoders

Item Co-occurrence as a Tiny-range Encoder The Tiny-range encoder M^T only focuses on the user's last event e_τ, ignoring all previous events. In other words, given the processed inputs from past events Z^{in}_τ, this encoder will only consider z^{in}_τ. As in factorizing Markov chain (FMC) models [45], M^T makes predictions based on item range-1 co-occurrence within observed sequences. For example, if most of users buy iPhone cases after purchasing an iPhone, then M^T should learn this item-to-item co-occurrence pattern. As shown in Figure 3a, we compute M^T's output as:

$$M^T(Z^{\text{in}}_\tau) = \phi(z^{\text{in}}_\tau), \text{ where } \phi(x) = \begin{cases} xW^{(T)} + b^{(T)}, & \text{if } d_{\text{in}} \neq d_{\text{enc}}, \\ x, & \text{otherwise.} \end{cases} \tag{8}$$

That is, when the dimensionality of processed input and encoder output are the same, the tiny-range encoder performs a role of residual for the other encoders in mixture. If $d_{\text{in}} \neq d_{\text{enc}}$, it is possible to down-sample (if $d_{\text{in}} > d_{\text{enc}}$) or up-sample (if $d_{\text{in}} < d_{\text{enc}}$) from z^{in}_τ by learned parameters $W^{(T)} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{enc}}}$ and $b^{(T)} \in \mathbb{R}^{d_{\text{enc}}}$.

In summary, the tiny-range encoder M^T can only focus on the last event by construction, meaning *it has a temporal range of 1 by design*. If we only use the output of M^T to make predictions, we obtain recommendations results based on item co-occurrence.

RNN/CNN as Short-range Encoder As discussed in Section 3, the recent behavior of a user has substantial predictive power on current and future interactions. Therefore, to leverage the corresponding signals entailed in observations, we consider instantiating a short-range sequence encoder that puts more emphasis on recent past events. Given the processed input from past events Z^{in}_τ, this encoder, represented as M^S, focuses by design on a recent subset of logged events. Based on our quantitative data exploration, we believe it is suitable for M^S to be highly sensitive to sequence order. For instance, we expect this encoder to capture the purchasing pattern iPhone → iPhone case → iPhone charger if it appears frequently in user sequences. As a result, we believe the Recurrent Neural Network (RNN [36]) and the Temporal Convolutional Network ([3, 53, 59]) are fitting potential architectural choices. Such neural architectures have shown superb performances when modeling high-order causalities. Beyond accuracy, these two encoders are also order sensitive, unlike early sequence modeling method (*i.e.* Bag-of-Word [27]). As a result we develop two interchangeable variants of M3: *M3R* and *M3C* using an RNN and a CNN respectively.

To further describe each of these options, let us introduce our RNN encoder M^S_{RNN}. As shown in Figure 3b we obtain the output of M^S_{RNN} by first computing the *hidden state* of RNN at step τ:

$$h_\tau = \text{RNN}(z^{\text{in}}_\tau, h_{\tau-1}), \tag{9}$$

where RNN$(\cdot)$ is a *recurrent cell* that updates the hidden state at each step based on the previous hidden state $h_{\tau-1} \in \mathbb{R}^{1 \times d_{\text{in}}}$ and the current RNN input z^{in}_τ. Several choices such as Gated Recurrent Unit (GRU) [10] or Long Short Term Memory (LSTM) [21] can be used. The output is then computed as follows:

$$M^S_{\text{RNN}}(Z^{\text{in}}_\tau) = h_\tau W^{(R)}, \text{ where } W^{(R)} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{enc}}} \tag{10}$$

where $W^{(R)}$ maps the hidden state to the encoder output space. We design our CNN encoder M^S_{CNN} as a Temporal Convolutional Networks which has provided state-of-art sequential modeling performance [3, 15, 53]. As shown in Figure 3c, this encoder consists of several stacked layers. Each layer computes

$$h^{(1)}_\tau = \text{Conv}(Z^{\text{in}}_\tau), \ldots, h^{(k)}_\tau = \text{Conv}(h^{(k-1)}_\tau), \tag{11}$$

where k indicates the layer number. The Conv$(\cdot)$ is a 1-D convolutional operator (combined with non-linear activations, see [3] for more details), which contains d_{enc} convolutional filters and operates on the convolutional inputs. With K layers in our CNN encoder, the final output will be:

$$M^S_{\text{CNN}}(Z^{\text{in}}_\tau) = h^{(K)}_\tau. \tag{12}$$

As highly valuable signals exist in the short-range part of user sequence, we propose two types of encoders to capture them. Our model can be instantiated in its first variant, *M3R*, if we use RNN encoder or *M3C* if a CNN is employed. Here M3C and M3R are totally interchangeable with each other and they show comparable results in our experiments (see Section 5.2.1). We believe such

flexibility will help practitioners adapt their model to the hardware they intend to use, *i.e.* typically using GPU for faster CNN training or CPU for which RNNs are better suited. In terms of temporal range, the CNN only considers a limited finite window of inputs when producing any output. The RNN, although it does not have a finite receptive field, is hampered by difficulties when learning to leverage events located further back into the past (to leverage an event located L observations ago the RNN needs $L-1$ steps). Regardless of the choice of a CNN or an RNN, our short-range encoder M^S *has a temporal range greater than 1, although it is challenging for this sub-model to capture signals too far away from current step.* This second encoder is specifically designed to capture sequence patterns that concern recent events.

Attention Model as Long-range Encoder The choice of an attention model is also influenced by our preliminary quantitative analysis. As discussed in Section 3, as the temporal distance grows larger, the uncertainties affecting the influence of item consumption on future events get larger as well. Moreover, as opposed to the recent part of a given user's interaction sequence, relative position does not matter as much when it comes to capturing the influence of temporally distant events. As we take these properties into account, we choose to employ *Attention Model* [2, 54] as our long-range sequence encoder. Usually, an attention model consists of three parts: attention *queries*, attention *keys* and attention *values*. One can simply regard an attention model as weighted-sum over attention values with weights resulting from the interaction between attention queries and attention keys. In our setting, we use (1) the last event's processed input z_τ^{in} as attention queries, (2) all past events' processed inputs Z_τ^{in} as keys and values and (3) scaled dot-product [54] as the similarity metric in the attention softmax. For instance, if a user last purchased a pair of shoes, the attention mechanism will focus on footwear related previous purchases.

So that all encoders have the same output dimensionality, we need to transform[1] our processed input first as follows:

$$\tilde{Z}_\tau^{\text{in}} = Z_\tau^{\text{in}} W^{(A)}, \tag{13}$$

where $W^{(A)} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{enc}}}$ is a learned matrix of parameters. Then for each position $i \in [1, \tau]$, we obtain its raw attention weights, with respect to the processed input $\tilde{z}_i^{\text{in}}$, as follows:

$$\omega_{\tau,i} = \frac{\tilde{z}_\tau^{\text{in}} \cdot \tilde{z}_i^{\text{in}}}{\sqrt{d_{\text{enc}}}}, \tag{14}$$

where $\omega_{\tau,i}$ is the raw weight at position i. Similarly, we compute the raw attention weights $\boldsymbol{\omega}_\tau \in \mathbb{R}^{1\times\tau}$ for all positions $\boldsymbol{\omega}_\tau = \{\omega_i\}_{i=1..\tau}$ and normalize them with a softmax$(\cdot)$ function. Finally, we acquire the output of our long-range encoder as follows:

$$M^L(Z_\tau^{\text{in}}) = \text{softmax}(\boldsymbol{\omega}_\tau) Z_\tau^{\text{in}}. \tag{15}$$

Our long-range encoder borrows several advantages from the attention model. First, it is not limited by a certain finite temporal range. That is, *it has an unlimited temporal range and can 'attend' to anywhere in user's sequence with O(1) steps.* Second,because it computes its outputs as a weighted sum of inputs, the attention-based encoder is not as sensitive to sequential order as an RNN or a CNN as each event from the past has an equal chance of influencing the prediction. Third, the attention model is robust to noisy inputs due to its normalized attention weights and weighted-sum aggregation.

Gating Network Borrowing the idea from from Mixture-of-Experts model [23, 35], we build a gating network to aggregate our encoders' results. The gate is also helpful to better understand our model (see Section 5). To produce a simpler gating network, we use a feed-forward layer F^g on the gating network's inputs:

$$G_\tau = [G_\tau^T, G_\tau^S, G_\tau^L] = \text{sigmoid}(F^g(G_\tau^{\text{in}})), \tag{16}$$

where G_τ^{in} is the input we feed into our gating network. We will discuss how the model performs overall with different choices of gate inputs in Section 5.4. The resulting $G_\tau \in \mathbb{R}^3$ contains the gate value modulating each encoder. More importantly, an element-wise sigmoid function is applied to the gate values which allows encoders to 'corporate' with each other [4]. Note that a few previous works [26, 35, 48] also normalize the gate values, but we found this choice led to the degeneracy of our mixture model as it would learn to only use M^S which in turn hampers model performance.

Summary M3 is able to address limitations of pre-existing models as shown in Table 1: (1) M3 has a mixture of three encoders with different temporal ranges which can capture sequential patterns located anywhere in user sequences. (2) Instead of learning a set of latent factor P_u for each user, M3 represents the long-term user preferences by using a long-range sequence encoder that provides a representation of the entire history of a user. Furthermore, M3 is efficient in both model size and computational cost. In particular M3 does not introduce any extra parameters under certain settings (*i.e.* $d_{\text{in}} = d_{\text{enc}}$), and the computation of M^T and M^L are very efficient when using specialized hardware such as a GPU. With its simple gate design, M3 also provides good interpretability and adaptability.

- **Effectiveness.** Given our analysis on user sequences, we assume M3 to be effective. As compared to past works, M3 is capable to capture signals from the whole sequence, it also satisfies the properties we found in different parts of sequence. Moreover, our three encoders constitute a diverse set of sequential encoder and, if well-trained, can model user sequence in a multi-scale manner, which is a key to success in past literature [53, 58].
- **Efficiency.** In terms of model size, M3 is efficient. As compared to existing works which use short-range encoder only, though uses two other encoders, our M3 model doesn't introduce any extra parameters (if $d_{\text{in}} = d_{\text{enc}}$). In terms of computational efficiency, our M3 is good as well, as both M^T and M^L are nothing other than matrix multiplication, which is cheap when computed with optimized hardwares like Graphics Processing Unit (GPU).
- **Interpretability.** Model's interpretability is critical for diagnosing purpose. As we shall see later, with the gate network, we are able to visualize our network transparently by observing the gate values.
- **Adaptability.** One issue in production recommender system is modeling users for different recommendation scenarios, as people may behave very differently. Two typical scenarios are *HomePage* recommendation and product *DetailPage* recommendation. However, as we shall introduce in later

[1] It is unnecessary if d_{in} is same as d_{enc}.

Table 1: A summary of relationships and differences between sequence encoders in *M3*

	Base model	Temporal range	Model size	Sensitive to order	Robustness
M^T	Item Co-occurrence	1	small (or 0)	very high	no
M^S_{RNN}	Recurrent Neural Nets	unknown	large	high	no
M^S_{CNN}	Temporal Convolution Nets	limited	large	high	no
M^L	Attention Model	unlimited	small (or 0)	no	high

section, M3 is able to adapt to these scenarios if we use the scenario information as our gate input.

5 EXPERIMENTAL STUDIES

In this section, we study the two variants of M3 against several baseline state-of-the-art methods on both a publicly available dataset and our large-scale Youtube dataset.

Datasets We use MovieLens 20M[2], which is a publicly available dataset, along with a large-scale anonymized dataset from YouTube to which we have access because we are employees of Google working on improving YouTube as a platform. The dataset is private, anonymized and accessible only internally by few employees whose work is directly related to Youtube.

5.1 Experiments on MovieLens Dataset

As in previous works [18, 51], we process the MovieLens data by first converting numeric ratings to 1 values, turning them into implicit logged item consumption feedback. We remove the items with less than 20 ratings. Such items, because of how little user feedback is available for them, represent another research challenge — cold start — which is outside the scope of the present paper.

To focus on long user sequences, we filtered out users who had a sequence length of less than $\delta_{\text{min}} = 20$ item consumed, while we didn't filter items specifically. The maximum sequence length in the dataset being 7450, we follow the method proposed in [18, 51] and employ a sliding window of length $\delta_{\text{win}} = 300$ to generate similarly long sequences of user/item interactions in which we aim to capture long range dependent patterns. Some statistics can be found in the first row of Table 3.

We do not use contextual annotations for the MovieLens data.

Evaluation protocol We split the dataset into training and test set by randomly choosing 80% of users for training and the remaining 20% for validation (10%) and testing (10%). As with the training data, a sliding window is used on the validation and test sets to generate sequences. We measure the mean average precision (mAP) as an indicator for models' performances [6, 52]. We only focus on the top positions of our predictions, so we choose to use mAP@n with $n \in \{5, 10, 20\}$. There is only one target per instance here and therefore the mAP@n is expected to increase with n which is consistent with [4] but differs from [51].

Details on model architectures We keep architectural parameters consistent across all experiments on MovieLens. In particular, we use identical representation dimensions: $d_{\text{in}} = d_{\text{enc}} = d_{\text{out}} = 32$. Such a choice decreases the number of free parameters as the sub-models M^T and M^L will not have learned parameters. A GRU cell is employed for the RNN while 2 stacked temporal convolution layers [3] of width 5 are used in the CNN. A ReLU activation function is employed in the feed-forward layers F^{in} and F^{out}. Item embeddings of dimension 64 are learned with different weights on the input side (*i.e.,* Q in Eq. 1) and output side (*i.e.,* Q' in Eq. 7). Although previous work [18] has constrained such embeddings to be identical on the input and output side of the model, we found that increasing the number of degrees of freedom led to better results.

Baselines We compare our two variants, *i.e.,* M3R and M3C, with the following baselines:

- **FMC**: The Factorizing model for the first-order Markov chain (FMC) [45] is a simple but strong baseline in sequential recommendation task [7, 49, 51]. As discussed in Section 1, we do not want to use explicit user representations. Therefore, we do not compare the personalized version of this model (FPMC).
- **DeepBoW**: The Deep Bag-of-word model represent user by averaging item embeddings from all past events. The model then makes predictions through a feed-forward layer. In our experiments, we use a single hidden layer with size of 32 and ReLU as activation function.
- **GRU4Rec**: Originally presented in [20], this method uses a GRU RNN over user sequences and is a state-of-the-art model for sequential recommendation with anonymized data.
- **Caser**: The Convolutional Sequence Embeddings model [51] applying horizontal and vertical convolutional filters over the embedding matrix and achieves state-of-the-art sequential recommendation performance. We try $\{2, 4, 8\}$ vertical filters and $\{16, 32, 64\}$ horizontal filters of size $(3, 5, 7)$. In order to focus on the sequential encoding task, we discard the user embedding and only use the sequence embedding of this model to make predictions.

In the models above, due to the large number of items in input and output dictionaries, the learned embeddings comprise most of the free parameters. Therefore, having set the embedding dimension to 64 in all the baselines as well as in M3R and M3C, we consider models with similar numbers of learned parameters. The other hyperparameters mentioned above are tuned by looking at the mAP@20 on validation set. The training time of M3R/M3C is comparable with others and can be further improved with techniques like model compression [52], quantization [22], etc.

5.1.1 Overall Performances. We report each model's performance in Table 2. Each metric is averaged across all user sequences in test set. The best performer is highlighted in bold face. The results show that both M3C and M3R outperform other baselines by a large margin. Among the baselines, GRU4Rec achieves the best performance

[2] https://grouplens.org/datasets/movielens/20m/

and DeepBoW worst one, suggesting the sequence order plays a very important predictive role. FMC performs surprisingly well, suggesting we could get considerable results with a simple model only taking the last event into account. The poor results of Caser may be caused by its design which relies on vertical filters of fixed size. Caser performs better in the next subsection which considers sequences whose lengths vary less within the training data.

Table 2: Performance comparison on MovieLens 20M. M3C and M3R outperform the baselines significantly.

	mAP@5	mAP@10	mAP@20
FMC	0.0256	0.0291	0.0317
DeepBoW	0.0065	0.0079	0.0093
GRU4Rec	0.0256	0.0304	0.0343
Caser	0.0225	0.0269	0.0304
M3C	0.0295	0.0342	0.0379
M3R	**0.0315**	**0.0367**	**0.0421**
Improv.	+23.4%	+20.7%	+22.7%

5.1.2 Investigating the influence of sequence length through variants of MovieLens. The previous results have shown strong performance gains achieved by the models we intruced: M3C and M3R. We now investigate the origin of such improvements. The design of these models was inspired by an attempt to capture sequential patterns with different characteristic temporal extents. To check whether the models we introduced achieve this aim we construct multiple variants of MovieLens with different sequence lengths.

We vary the sequence length by having a maximum cutoff threshold $\delta_{\max}$ which complements the minimal sequence length threshold $\delta_{\min}$. A sequence with more than $\delta_{\max}$ only has its latest $\delta_{\max}$ observations remained. We vary the values of $\delta_{\min}$, $\delta_{\max}$ and the sequence generation window size. Table 3 summarizes the properties of the four variants of the MovieLens dataset we construct. It is noteworthy that such settings make Caser perform better as the sequence length is more consistent within each dataset variant.

GRU4Rec and Caser outperform the other baselines in the present setting and therefore we only report their performance. Figure 4 shows the improvements of M3C and M3R over the best baselines on four MovieLens variants. The improvement of each model is computed by its mAP@20 against the best baseline. In most cases, M3C and M3R can outperform the highest performing baseline. Specifically, on ML20M-S and ML20M-M, Caser performs similarly to GRU4Rec while both M3C and M3R have good performance. This is probably due to the contribution of the tiny-range encoder.

5.2 Anonymized YouTube dataset

For the YouTube dataset, we filtered out users whose logged sequence length was less than 150 ($\delta_{\min} = 150$) and keep each user's last 300 events ($\delta_{\max} = 300$) in their item consumption sequence. In the following experiments, we exploit contextual annotations such as user device (*e.g.,* from web browser or mobile App), time-based features (*e.g.,* dwelling time), etc. User sequences are all anonymized and precautions have been taken to guarantee that users cannot be re-identified. In particular, only public videos with enough views have been retained.

Neural recommender systems attempt at foreseeing the interest of users under extreme constraints of latency and scale. We define the task as predicting the next item the user will consume given a recorded history of items already consumed. Such a problem setting is indeed common in collaborative filtering [34, 46] recommendations. We present here results obtained on a dataset where only about 2 million items are present that correspond to most popular items. While the user history can span over months, only watches from the last 7 days are used for labels in training and watches in the last 2 days are used for testing. The train/test split is 90/10%. The test set does not overlap with the train set and corresponds to the last temporal slice of the dataset. In all, we have more than 200 million training sequences and more than 1 million test sequences, and with overall average sequence length approximately being 200.

The neural network predicts, for a sample of negatives, the probability that they are chosen and classically a negative sampling loss is employed in order to leverage observations belonging to a very large vocabulary [25]. The loss being minimized is

$$\sum_{l \in \text{Labels}} w_l \times \text{CrossEntropy}(\text{SampledSoftmax}(\xi(t+1)))$$

where the SampledSoftmax [25] uses 20000 randomly sampled negatives and w_l is the weight of each label.

Evaluation metrics To test the models' performances, we measure the mean average precision (mAP) as in [7, 51]. We only focus on the top positions of our predictions, so we choose to use mAP@n with $n \in \{5, 10, 20\}$. The mAP is computed with the entire dictionary of candidate items as opposed to the training loss which samples negatives. There is only one target per instance here and therefore the mAP@n is expected to increase with n which is consistent with [4] but differs from [51].

Baselines In order to make fair comparisons with all previous baselines, we used their contextual counterparts if they are proposed or compared in literature.

- **Context-FMC**: The Context-FMC condition the last event's embedding on last event's context features by concatenating them and having a feed-forward layer over them.
- **DeepYouTube**: Proposed by [11], the DeepYoutube model is a state-of-the-art neural model for recommendation. It concatenates: (1) item embedding from users' last event, (2) item embeddings averaged by all past events and (3) context features. The model then makes predictions through a feedforward layer composed of several ReLU layers.
- **Context-GRU**: We used the contextual version of GRU proposed in [49]. Among the three conditioning paradigms on context, we used the concatenation as it gives us better performances.

All models are implemented by TensorFlow [1] and by Adagrad [14] over a parameter server [32] with many workers.

Model details In the following experiments, we keep the dimensions of processed input d_{in} and encoder outputs d_{enc} identical for all experiments conducted on the same dataset. Once more, we also want to share some of our architectural parameters so that they are consistent across the two datasets. Again, by doing this, we make the parametrization of our models more parsimonious,

Table 3: Statistics of the variants of the MovieLens dataset.

	Min. Length	Max. Length	Window Size	Avg. Length	#Sequences	#Items
ML20M	20	∞	300	144.1	138.4K	13.1K
ML20M-S	20	50	20	42.8	138.4K	13.1K
ML20M-M	50	150	50	113.6	85.2K	13.1K
ML20M-L	150	300	150	250.7	35.8K	12.9K
ML20M-XL	300	∞	300	605.5	16.3K	12.5K

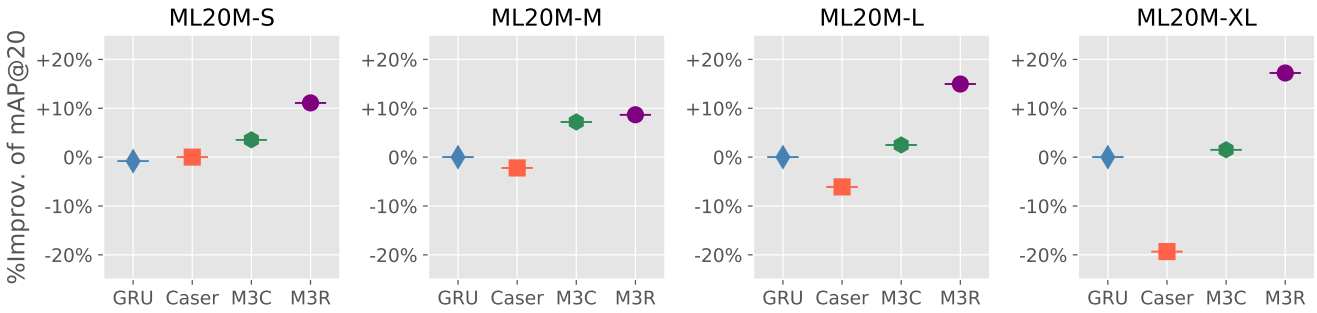


Figure 4: Uplifts with respect to the best baselines on four MovieLens variants. The improvement percentage of each model is computed by its relative mAP@20 gain against the best baseline. For all variants, M3R significantly outperforms the two baselines we consider according to a one-tail paired t-test at level 0.01, while M3C outperforms the other two significantly only on ML20M-M. Note that the standard error of all uplifts gets higher as we use a MovieLens variant with longer sequences. The standard error reaches 2.3% on ML20M-XL.

Table 4: Performance comparison on the anonymized YouTube dataset. M3C and M3R outperform the baselines significantly.

	mAP@5	mAP@10	mAP@20
Context-FMC	0.1103	0.119	0.1240
DeepYouTube	0.1295	0.1399	0.1455
Context-GRU	0.1319	0.1438	0.1503
M3C	0.1469	0.1591	0.1654
M3R	**0.1541**	**01670**	**0.1743**
Improv.	+16.8%	+16.1%	+16.0%

because the sub-models M^T and M^L will be parameter-free. For the RNN cell, we use a GRU on both datasets for its effectiveness as well as efficiency. For the CNN version, we stacked 3 layers of temporal convolution [3], with no dilation and width of 5. For the feed-forward layers F^{in} and F^{out}, we used ReLU as their activation functions, whereas they contains different number of sub-layers. For item embeddings on the input side (*i.e.,* Q in Eq. 1) and on the output side (*i.e.,* Q' in Eq. 7), we learn them separately which improves all results.

5.2.1 Overall Results. We report each model's performance on the private dataset in Table 4. The best performer is highlighted in bold face. As can be seen from this table, on our anonymized YouTube dataset, the Context-FMC performs worse followed by DeepYoutube while Context-GRU performs best among all baselines. The DeepYouTube and Context-GRU perform better than Context-FMC possibly because they have longer temporal range, which again shows that the temporal range matters significantly in long user sequences. One can therefore improve the performance of a sequential recommender if the model is able to leverage distant (long-range dependent) informantion in user sequences.

On both datasets, we observed our proposed two model variants M3R and M3C significantly outperform all other baselines. Within these two variants, the M3R preforms marginally better than the M3C, and it improves upon the best baselines by a large margin (more than 20% on MovieLens data and 16.0% on YouTube data).

5.3 Ablation Study of Mixture of Models

To demonstrate how each encoder contributes to the overall performance, we now present an ablation test on our M3R model (results from M3C are similar) on our proprietary data. We use T, S, L to denote M^T, M^S and M^L respectively. The results are described in Table 5. When we only enable single encoder for M3R, the best performer is M3R-T on MovieLens data and M3R-S on the YouTube data. This result is consistent with the results in Section 5.2.1. With more encoders involved in M3R the model performs better. In particular, when all encoders are incorporated, our M3R-TSL performs best on both datasets, indicating all three encoders matter for performance.

Table 5: mAP@20 vs. different components of M3R on both datasets, where T,S,L stands for M^T, M^S and M^L respectively.

	MovieLens 20M	YouTube Dataset
M3R-T	0.0269	0.1406
M3R-S	0.0363	0.1673
M3R-L	0.0266	0.1359
M3R-TS	0.0412	0.1700
M3R-TL	0.0293	0.1485
M3R-SL	0.0403	0.1702
M3R-TSL	0.0421	0.1743

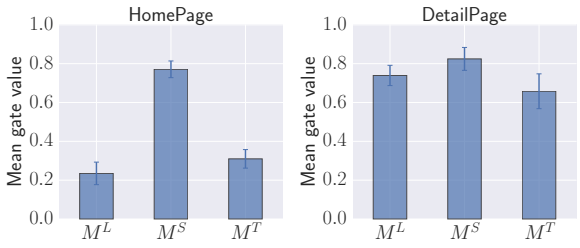


Figure 5: Average gate values of M3R in different scenarios. The model learns to use different combination of encoders in different recommendation scenarios.

5.4 Role of Gating Network

We now begin to study our gating network in order to answer the following questions: (1) Is the gating network beneficial to the overall model performance? (2) How do different gating network inputs influence the model performance? and (3) How can the gating network make our model more adaptable and interpretable?

Fixed gates versus learned gates: First of all, we examine the impact of our gating network by comparing it with a set of fixed gate values. More precisely, we fixed the gate values to be all equal to 1.0 during the model training: $G_\tau = \mathbf{1}$, here $\mathbf{1} \in \mathbb{R}^3$ is a vector. The first row of Table 6 shows the result of this fixed-gate model. We found that the fixed models are weaker than the best performing version of M3R (*i.e.,* mAP@20 of 0.1743) and M3C (*i.e.,* mAP@20 of 0.1654). This reveals that the gating network consistently improves M3-based models' performances.

Influence of different gate inputs: In this paragraph we investigate the potential choices of inputs for the gating network, and how they result in different performance scores. In the existing Mixture-of-Experts (MOE) literature, the input for the gating network G_τ^{in} can be categorized into Contextual-switch and Bottom-switch. The Contextual-switch, used in [4], uses context information as gate input:

$$G_\tau^{\text{in}} = [c_\tau^{\text{in}} \oplus c_\tau^{\text{out}}], \tag{17}$$

where c_τ^{in} and c_τ^{out} are context features from input and output side. Intuitively, this suggests how context may influence the choices of different encoders. If no context information is available, we can still use the output of a shared layer operating before the MOE layer [35, 48] as gate input, *i.e.,* Bottom-switch:

$$G_\tau^{\text{in}} = z_\tau^{\text{in}}. \tag{18}$$

The shared layer contains high-level semantic knowledge from the last event, which can also enable gate switching.

Table 6: mAP@20 vs. different types of gating network on the two datasets for M3R. 'Fixed' indicates we fix gate values to 1.0, 'Contextual-switch' means that we use context features c^{in} and c^{out} as gate input and 'Bottom-switch' corresponds to the use of z_τ^{in} as gate input.

	MovieLens	YouTube
Fixed	0.0413	0.1715
Bottom-switch	0.0421	0.1734
Contextual-switch	/	0.1743

On the MovieLens data, we used Bottom-switched gate for all the results above because of the absence of contextual annotations. On the YouTube dataset, the last two rows from Table 6 provide the comparison results between Contextual-switched gate and Bottom-switched gate. We observe that context information is more useful to the gates than a shared layer. In other words, the decision of whether to focus more on recent part (*i.e.* large gate values for M^T and M^S) or on the distant part (*i.e.* large values for M^L) from user sequence is easier to make based on contextual annotations.

5.5 Discussion on model architecture

The model architecture we design is based on quantitative findings and has two primary goals: capturing co-existing short-range and long-range behavioral patterns as well as serving recommendations given in different contexts with a single model.

We know for recommender systems in most applications (*e.g.* e-commerce like Amazon, streaming services like Netflix) that recommendations commonly occur in at least two different contexts: either a *HomePage* or a *DetailPage*. The Homepage is the page shown when users open the website or open the mobile App, while DetailPage is the page shown when users click on a certain item. User behaviors are different depending on which of these two pages they are browsing. Users are more likely to be satisfied by a recommendation related to recent events, especially the last event, when they are on a DetailPage. A straightforward solution to deal with these changing dynamics is to train two different models.

We now demonstrate that with the multi-temporal-range encoders architecture and gating mechanism in M3, we can have a single adaptive end-to-end model that provides good performance in a multi-faceted recommendation problem. To that end we analyze the behavior of our gating network and show the adaptability of the model as we gain a better understanding of its behavior.

What we observe in Figure 5 is that when contextual information is available to infer the recommendation scenario, the gating network can effectively automatically decide how to combine the results from different encoders in a dynamic manner to further improve performance. Figure 5 shows how gate values of M3R change *w.r.t.* across different recommendation scenarios. It is clear that M3R puts more emphasis on M^S when users are on the HomePage,

while it encourages all three encoders involved when users are on DetailPage. This result shows that the gating network uses different combinations of encoders for different recommendation scenarios.

As a result, we can argue that our architectural design choices do meet the expectations we set in our preliminary analysis. It is noteworthy that the gating mechanism we added on top of the three sub-models is helpful to improve predictive performance and ease model diagnosis. We have indeed been able to analyze recommendation patterns seamlessly.

6 CONCLUSION

M3 is an effective solution to provide better recommendations based on long user sequences. M3 is a neural model that avoids most of the limitations faced by pre-existing approaches and is well adapted to cases in which short term and long term temporal dependencies coexist. Other than effectiveness, this approach also provides several advantages such as the absence of a need extra parameters and interpretability. Our experiments on large public dataset as well as a large-scale production dataset suggest that M3 outperforms the state-of-the-art methods by a large margin for sequential recommendation with long user sequences. One shortcoming of the architecture we propose is that all sub-models are computed at serving time. As a next step, we plan to train a sparse context dependent gating network to address this shortcoming.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. [n. d.]. Tensorflow: a system for large-scale machine learning.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[3] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271* (2018).

[4] Francois Belletti, Alex Beutel, Sagar Jain, and Ed Chi. 2018. Factorized Recurrent Neural Architectures for Longer Range Dependence. In *International Conference on Artificial Intelligence and Statistics*. 1522–1530.

[5] Francois Belletti, Evan Sparks, Alexandre Bayen, and Joseph Gonzalez. 2017. Random projection design for scalable implicit smoothing of randomly observed stochastic processes. In *Artificial Intelligence and Statistics*. 700–708.

[6] Alex Beutel, Ed H Chi, Zhiyuan Cheng, Hubert Pham, and John Anderson. 2017. Beyond globally optimal: Focused learning for improved recommendations. In *International Conference on World Wide Web*. 203–212.

[7] Alex Beutel, Paul Covington, Sagar Jain, Can Xu, Jia Li, Vince Gatto, and Ed H Chi. 2018. Latent Cross: Making Use of Context in Recurrent Recommender Systems. In *International Conference on Web Search and Data Mining*. ACM, 46–54.

[8] Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. 2019. Top-K Off-Policy Correction for a REINFORCE Recommender System. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 456–464.

[9] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *International Conference on Web Search and Data Mining*. ACM, 108–116.

[10] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[11] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *ACM Conference on Recommender Systems*.

[12] Robin Devooght and Hugues Bersini. 2017. Long and short-term recommendations with recurrent neural networks. In *Conference on User Modeling, Adaptation and Personalization*. ACM, 13–21.

[13] Thomas G Dietterich. 2000. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*. Springer, 1–15.

[14] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[15] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122* (2017).

[16] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline a/b testing for recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 198–206.

[17] Carlos A Gomez-Uribe and Neil Hunt. 2016. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)* 6, 4 (2016), 13.

[18] Ruining He and Julian McAuley. 2016. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *International Conference on Data Mining*. IEEE.

[19] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *International Conference on World Wide Web*. ACM, 173–182.

[20] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[21] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[22] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2017. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research* 18, 1 (2017), 6869–6898.

[23] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation* 3, 1 (1991), 79–87.

[24] Dietmar Jannach and Malte Ludewig. 2017. When Recurrent Neural Networks meet the Neighborhood for Session-Based Recommendation. In *ACM Conference on Recommender systems*. ACM, 306–310.

[25] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2014. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007* (2014).

[26] Michael I Jordan and Robert A Jacobs. 1994. Hierarchical mixtures of experts and the EM algorithm. *Neural computation* 6, 2 (1994), 181–214.

[27] Dan Jurafsky and James H Martin. 2014. *Speech and language processing*. Vol. 3. Pearson London.

[28] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. 2014. Large-scale video classification with convolutional neural networks. In *IEEE conference on Computer Vision and Pattern Recognition*.

[29] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM.

[30] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[31] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. 1097–1105.

[32] Mu Li, David G Andersen, and Jun Woo Park. [n. d.]. Scaling Distributed Machine Learning with the Parameter Server.. In *Proceedings of USENIX Symposium on Operating Systems Design and Implementation*.

[33] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 689–698.

[34] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* (2003).

[35] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. 2018. Modeling Task Relationships in Multi-task Learning with Multi-gate Mixture-of-Experts. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1930–1939.

[36] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model.. In *Interspeech*.

[37] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning*. 807–814.

[38] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*. 1310–1318.

[39] Michael J Pazzani and Daniel Billsus. 2007. Content-based recommendation systems. In *The adaptive web*. Springer, 325–341.

[40] Pedro HO Pinheiro and Ronan Collobert. 2014. Recurrent convolutional neural networks for scene labeling. In *International Conference on Machine Learning*.

[41] Vladas Pipiras and Murad S Taqqu. 2017. *Long-range dependence and self-similarity*. Vol. 45. Cambridge university press.

[42] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *ACM Conference on Recommender Systems*. ACM, 130–137.

[43] Steffen Rendle. 2010. Factorization machines. In *International Conference on Data Mining*. IEEE, 995–1000.

[44] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 452–461.

[45] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *International Conference on World Wide Web*. ACM, 811–820.

[46] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *International Conference on World Wide Web*. ACM, 285–295.

[47] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *International Conference on World Wide Web*. ACM, 111–112.

[48] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538* (2017).

[49] Elena Smirnova and Flavian Vasile. [n. d.]. Contextual Sequence Modeling for Recommendation with Recurrent Neural Networks. *arXiv preprint arXiv:1706.07684* ([n. d.]).

[50] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*.

[51] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 565–573.

[52] Jiaxi Tang and Ke Wang. 2018. Ranking Distillation: Learning Compact Ranking Models With High Performance for Recommender System. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2289–2298.

[53] Aäron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W Senior, and Koray Kavukcuoglu. 2016. WaveNet: A generative model for raw audio.. In *SSW*. 125.

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[55] Chen Wu and Ming Yan. 2017. Session-aware information embedding for e-commerce product recommendation. In *ACM on Conference on Information and Knowledge Management*. ACM, 2379–2382.

[56] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent Recommender Networks. In *International Conference on Web Search and Data Mining*. ACM, 495–503.

[57] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *International Conference on Web Search and Data Mining*. ACM, 153–162.

[58] Fisher Yu and Vladlen Koltun. 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122* (2015).

[59] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2019. A Simple Convolutional Generative Network for Next Item Recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 582–590.

[60] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2018. Deep Interest Evolution Network for Click-Through Rate Prediction. *arXiv preprint arXiv:1809.03672* (2018).

[61] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1059–1068.

[62] Zhi-Hua Zhou, Jianxin Wu, and Wei Tang. 2002. Ensembling neural networks: many could be better than all. *Artificial intelligence* 137, 1-2 (2002), 239–263.

[63] Yu Zhu, Junxiong Zhu, Jie Hou, Yongliang Li, Beidou Wang, Ziyu Guan, and Deng Cai. 2018. A Brand-level Ranking System with the Customized Attention-GRU Model. *arXiv preprint arXiv:1805.08958* (2018).

Leveraging Peer Communication to Enhance Crowdsourcing

Wei Tang
Washington University in St. Louis
w.tang@wustl.edu

Chien-Ju Ho
Washington University in St. Louis
chienju.ho@wustl.edu

Ming Yin
Purdue University
mingyin@purdue.edu

ABSTRACT

Crowdsourcing has become a popular tool for large-scale data collection where it is often assumed that crowd workers complete the work *independently*. In this paper, we relax such independence property and explore the usage of *peer communication*—a kind of direct interactions between workers—in crowdsourcing. In particular, in the crowdsourcing setting with peer communication, a *pair* of workers are asked to complete the same task together by first generating their initial answers to the task independently and then freely discussing the task with each other and updating their answers after the discussion. We first experimentally examine the effects of peer communication on individual microtasks. Our results conducted on three types of tasks consistently suggest that work quality is significantly improved in tasks with peer communication compared to tasks where workers complete the work independently. We next explore how to utilize peer communication to optimize the requester's total utility while taking into account higher data correlation and higher cost introduced by peer communication. In particular, we model the requester's online decision problem of whether and when to use peer communication in crowdsourcing as a constrained Markov decision process which maximizes the requester's total utility under budget constraints. Our proposed approach is empirically shown to bring higher total utility compared to baseline approaches.

CCS CONCEPTS

• **Human-centered computing → Empirical studies in HCI**; **Computer supported cooperative work**.

KEYWORDS

crowdsourcing; peer communication; collaboration

ACM Reference Format:
Wei Tang, Chien-Ju Ho, and Ming Yin. 2019. Leveraging Peer Communication to Enhance Crowdsourcing. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313554

1 INTRODUCTION

Crowdsourcing has gained increasing popularity recently as a scalable data collection tool for various purposes, such as obtaining labeled data for training machine learning algorithms and getting high-quality yet cheap transcriptions for audio files. On a typical crowdsourcing platform like Amazon Mechanical Turk (MTurk), task requesters can post small jobs (e.g., image labeling or audio transcription tasks) as "microtasks" along with the specified payment for completing each task. Workers then can browse all available tasks on the platform and decide which ones to work on. Crowd workers are often assumed to complete tasks *independently*, and a substantial amount of crowdsourcing research has been focused on how to make better use of the independent workers. For example, a rich body of research has explored how to aggregate independent contributions from multiple workers by inferring task difficulties, worker skills, and correct answers simultaneously [10, 41, 49, 53]. Moreover, given a limited budget, researchers have further examined how to intelligently decide the number of independent workers needed for each task at the first place [8, 21, 34].

However, the validity and value of this independence assumption in crowdsourcing has been challenged recently. Through a combination of ethnographic and experimental methodologies, researchers have found that crowd workers, in fact, communicate and collaborate with each other through both online forums (like TurkerNation[1] and MTurkCrowd[2]) and one-on-one channels [19, 20, 25, 37, 44, 51]. Different from such collaboration which is organically arisen within the crowd and mostly about exchanging meta-level information related to crowd work (e.g., how to find well-paid tasks), an increasing number of studies in the human-computer interaction community have started to design certain level of interactions between workers in their actual work, which is shown to improve crowdsourcing outcomes in many cases. For example, various *workflows* are developed to coordinate workers to work on *different* subtasks and interact with each other through the pre-defined input-output handoffs [4, 9, 30–32, 35, 40, 42], which enable the crowd to jointly complete complex tasks.

More recently, worker interactions are further introduced between workers of the *same* task: Drapeau et al. [18] and Chang et al. [7] showed that in image/text labeling tasks, workers can improve their labeling accuracy when *indirect* interactions—in the form of showing each worker the alternative answer and associated justification produced by another worker who works on the same task—are enabled, and Schaekermann et al. [45] observed that in text classification tasks, worker performance increases when they can debate their answers through *direct, real-time* deliberation with one another. While these research show the promise of an alternative way to structure crowd work that leads to higher performance, they also raise a number of open questions.

First, on the "micro" level, it is important to empirically examine whether adding interactions between workers working on the same task leads to an increase in work quality for *individual* tasks of *different* types, especially for those tasks with a large number of possible answers (rather than just a few options as in image labeling and text classification tasks). Indeed, when the number of possible

[1] http://turkernation.com/
[2] https://www.mturkcrowd.com/

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313554

answers in a task becomes large, workers may hardly agree with each other so it is unclear whether interactions between them would be meaningful and effective. It is also impractical for workers to argue against all alternative answers during their interactions, which may imply the need for new formats of interactions beyond providing justification and argumentation.

Furthermore, from a more "macro" point of view, requesters typically have a *large batch* of tasks at hand and need to solicit answers from multiple workers for each task. Their goal is to optimize their *overall* utility such as maximizing the quality obtained across *all* the tasks under a fixed budget. Yet, compared to independent work, allowing worker interactions in a task can bring up not only work quality improvement in that task, but also higher cost and higher correlation in workers' answers. Thus, for requesters to make better use of worker interactions, a critical problem to address is that given a limited budget, *whether* and *when* should worker interactions be used in each task, such that after combining the possibly correlated answers together for each task, the work quality for the entire batch of tasks is maximized when the budget is exhausted.

In this paper, we attempt to answer these two questions. In particular, inspired by the concept of peer instruction in education [12], we focus on studying a specific format of worker interactions that we refer to as *peer communication*—a *pair* of workers working on the same task are asked to first provide an independent answer each, then freely discuss the task with each other, and finally provide an updated answer, again independently, after the discussion. Compared to worker interaction formats used in the early research (e.g., justification and argumentation), we consider peer communication as a kind of direct and synchronous interaction that can be generalized to different types of tasks more easily. Our goal is to better understand not only whether and how peer communication would affect the outcome of crowd work for various types of tasks, but also how requesters can use algorithmic approaches to better utilize the potential benefits brought up by peer communication.

To understand the effects of peer communication on crowd work, we design and conduct randomized experiments with three different types of tasks: image labeling, optical character recognition, and audio transcription. For all types of tasks in our experiments, regardless of how large the number of possible answers in the task is, we have consistently observed an increase in work quality when workers can talk with their peers in the task compared to workers who work independently. Yet, we do not observe any spillover effects of such quality improvement when workers who have engaged in peer communication work on similar tasks again independently.

Moreover, to examine how peer communication can be better utilized, we propose an algorithmic framework to help requesters make *online decisions* on whether and when to use peer communication for each task in their batch, with the goal of maximizing the overall work quality produced in all tasks given a budget constraint. One of the key challenges here is how to infer the correct answer for a task given multiple answers solicited from workers, where some of them may be produced following the peer communication procedure and thus may be correlated. To this end, we introduce the notions of *meta-workers* and *meta-labels* to describe a pair of workers who have engaged in a task with peer communication and the pair of answers produced by them. Such notions enable us to characterize the possible correlation in data, which further allow us to solve the requester's online decision-making problem by modeling it as a constrained Markov decision process.

We evaluate the effectiveness of the proposed algorithmic approach on real data collected through our experimental study. Results show that using our approach to decide the usage of peer communication in tasks, the requester can achieve higher overall quality across all her tasks when the budget is exhausted, compared to when baseline approaches are adopted where peer communication is always used in all tasks or never used in any of the tasks, or when correlation in data is not explicitly considered. In addition, through two sets of simulated experiments, we further examine how the proposed algorithmic approach performs in various scenarios when the differences in work quality and cost between hiring pairs of communicating workers and hiring independent workers vary, and when answers produced by pairs of communicating workers are correlated to different extent.

In summary, we make the following contributions:

- We introduce peer communication, a general mechanism adapted from the concept of peer instruction in education for including worker interactions in crowd work.
- We empirically show that on different types of tasks, compared to independent work, peer communication consistently leads to a 32%–47% improvement in work quality for individual tasks.
- We propose an algorithmic framework to help requesters dynamically decide whether and when to use peer communication for each task in their batch, so as to maximize the overall quality obtained across all tasks given a budget constraint.
- Through evaluations on both real data from crowd workers and synthetic data, we demonstrate that compared to baseline approaches, using our proposed algorithm to determine the deployment of peer communication leads to higher requester utility.

2 RELATED WORK

Our work joins a long line of research on improving the quality of crowd work. In traditional settings where it is assumed that workers independently complete tasks, various methods have been proposed to address this problem, including post-hoc aggregation of workers' answers [10, 14, 15, 26, 41, 49, 53], designing effective extrinsic incentives [23, 24, 38, 50] and intrinsic motivation [33, 43, 47], appropriate assignment of tasks to workers [22, 27, 28], etc.

We explore how to improve the quality of crowd work from a different angle, that is, by adding interactions between workers. Researchers have previously designed *workflows* for complex tasks to allow workers to work on different subtasks while indirectly interacting with one another through the pre-defined input-output handoffs [4, 9, 30–32, 35, 40, 42]. Different from these workflow-based approaches, we consider the addition of interactions between workers of the *same* task. A few previous studies [7, 18, 45] have showed that enabling interactions between workers working on the same task, in the form of asking workers to provide justification for their answers, can lead to improvement in work quality, but these studies only test this idea on classification tasks. We aim to extend this idea to a wider range of tasks, especially for tasks with a large number of possible answers.

In this paper, we study a specific format of interactions, *peer communication*, which is adapted from "peer instruction" [12] and

"think-pair-share" strategies [36] in the educational settings. There is a rich literature in the collaborative learning community suggesting that asking students to discuss conceptual questions with other students after they independently answer the questions leads to higher levels of understanding and post-test performance [11, 16, 48]. We thus design the peer communication procedure as first asking a pair of workers to provide an independent answer each, then allowing them to *freely* discuss the task with each other, and finally independently update their answers. While evidence in the collaborative learning community and results for adding argumentation in classification tasks seem to indicate peer communication would lead to higher work quality, other studies showed that allowing workers to chat during work doesn't change work quality [52]. Thus, it is necessary to re-examine the impact of peer communication on the quality of crowd work, if any. In addition, as prior research on inter-task effects [1, 6, 39] suggests that when working on a sequence of tasks, workers' responses for later tasks could be influenced by the earlier tasks, we further examine whether peer communication brings any "spillover" effect on work quality. That is, whether workers produce higher independent work quality after engaging in similar tasks with peer communication.

Besides empirically showing the benefits of peer communication on work quality, we further provide an algorithmic framework for helping requesters better utilize such benefits. Early work has explored how indirect interactions between workers of different tasks that are embedded in certain workflows can be algorithmically controlled in order to maximize requester utility [13]. In contrast, the purpose of our algorithmic framework is to dynamically decide whether and when to deploy peer communication between workers of the *same* task for each task in requesters' batch to maximize their utility. Our framework is built on top of the work by Chen et al. [8], in which they used a Markov decision process to sequentially decide which task in requesters' batch needs an additional worker to work on given a budget constraint. However, our framework has a few key differences: First, in addition to choose which task needs further work, we also decide on how to design that piece of work—hiring one independent worker or two communicating workers? Second, when making inference for each task, we need to consider the possible correlation in the answers for this task. Finally, since peer communication and independent work incurs *different cost*, this decision-making problem does *not* degenerate into a finite-horizon Markov decision process. Thus, we explicitly model the problem as a *constrained* Markov decision process.

3 EXAMINING PEER COMMUNICATION VIA REAL-WORLD EXPERIMENTS

In this section, we first present our experimental study, in which we carefully examine the effects of introducing peer communication between pairs of workers on the quality produced in individual tasks through a set of randomized experiments conducted on Amazon's Mechanical Turk (MTurk). In particular, we ask:

- **Question 1 (Q1)**: Do workers produce higher work quality in tasks with peer communication compared to that in tasks where workers work independently?

Previous studies on the effects of adding worker interactions in image and text classification tasks [7, 18, 45] seem to imply a positive answer for Q1. Compared to these studies, our study has two key differences that warrant a re-examination of Q1: (1) the main format of interaction in peer communication is a *synchronous, free-form chat* rather than required justification and argumentation; (2) we consider different types of tasks beyond classification, especially tasks with a large number of possible answers so workers can hardly agree with each other or argue against all alternative answers. Moreover, we are also interested in examining whether there is any "spillover" effects of the impact of peer communication on work quality. Specifically:

- **Question 2 (Q2)**: Do workers produce higher independent work quality after engaging in similar tasks with peer communication, compared to workers who always complete tasks on their own?

Both positive and negative answers might be possible for Q2: On the one hand, if communication between workers in tasks allow them to resolve misconception about the tasks or learn useful problem-solving strategies from each other, we might expect a positive answer; on the other hand, if the benefits of peer communication are mostly due to workers being able to exchange their confidence levels on a task and eventually converge to the more confident answer [2], the answer for Q2 would likely be negative.

3.1 Independent Tasks vs. Discussion Tasks

In our experiments, we considered two ways to structure the tasks:

- **Independent tasks** (tasks without peer communication). In an independent task, workers are instructed to complete the task on their own.
- **Discussion tasks** (tasks with peer communication). In discussion tasks, we designed a procedure which guides workers to communicate with each other and complete the task together. Specifically, each worker is paired with another "co-worker" on a discussion task. Both workers in the pair are first asked to work on the task and submit their answers independently. Then, the pair enters a chat room, where they can see each other's independent answer. Workers are instructed to freely discuss the task with their co-workers for two minutes; for example, they can explain to each other why they believe their answers are correct. After the discussion, both workers get the opportunity to independently update and submit their final answers.

3.2 Experimental Treatments

The most straight-forward experimental design would include two treatments, where workers in one treatment are asked to work on a sequence of independent tasks while workers in the other treatment complete a sequence of discussion tasks. However, if we adopt such a design, the different nature of independent and discussion tasks (e.g., discussion tasks require more time and effort from workers but can be more interesting to workers) implies the possibility of observing severe self-selection biases in the experiments (i.e., workers may self-select into the treatment that they can complete tasks faster or find more enjoyable).

To overcome the drawback of this simple design, we design our experiments in a way that each treatment consists of the same number of independent tasks *and* discussion tasks, so neither treatment appears to be obviously more time-consuming or enjoyable. Figure 1 illustrates the two treatments used in our experiments. In

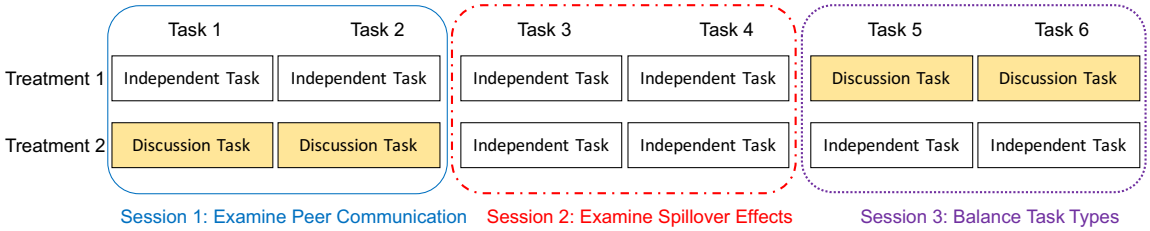


Figure 1: The two experimental treatments. This design enables us to examine whether peer communication improves the quality of crowd work (by comparing work quality in Session 1) and if so, does the improvement spill over to the following independent tasks (by comparing work quality in Session 2), while not creating significant differences between the two treatments (by adding Session 3 to make the two treatments containing equal number of independent and discussion tasks).

particular, we bundle 6 tasks in each HIT (i.e., Human Intelligence Task on MTurk). When a worker accepted our HIT, she was told that there are 4 independent tasks and 2 discussion tasks in the HIT. There are two treatments in our experiments:

- *Treatment 1*: Workers are asked to complete 4 independent tasks followed by 2 discussion tasks.
- *Treatment 2*: Workers are asked to complete 2 discussion tasks followed by 4 independent tasks.

Importantly, we did *not* tell workers the ordering of the 6 tasks, which helps us to minimize the self-selection biases as the two treatments look the same to workers. We refer to the first, middle, and last two tasks in the sequence as Session 1, 2, 3 of the HIT, respectively. Thus, we can answer Q1 by comparing the work quality produced in Session 1 between the two treatments, while a comparison of work quality in Session 2 between the two treatments would allow us to answer Q2. Finally, Session 3 is used for balancing the number of independent and discussion tasks in each HIT.

3.3 Experimental Tasks

We conducted our experiments on three types of tasks:

- **Image labeling.** In each task, the worker is asked to identify whether the dog shown in an image is a Siberian Husky or a Malamute. Dog images we use are collected from the Stanford Dogs dataset [29].
- **Optical character recognition (OCR).** In each task, the worker is asked to transcribe a vehicle's license plate numbers from photos. The photos are taken from the dataset provided by Shah and Zhou [46].
- **Audio transcription.** In each task, the worker is asked to transcribe an audio clip which contains about 5 seconds of speech. The audio clips are collected from VoxForge[3].

We decided to conduct our experiments on these three types of tasks for two main reasons: First, these tasks are all very common types of tasks on crowdsourcing platforms [17], so experimenting with them would allow us to better understand the effects of peer communication on typical kind of crowd work. Second, in terms of the number of possible answers, these tasks span a wide spectrum from two (image labeling) to infinitely many (audio transcription), enabling us to both confirm the effects of peer communication in tasks with just a few possible answers and explore its effects in tasks with many possible answers. As a final note, tasks we bundled in the same HIT had a certain degree of similarity[4], hence a spillover effect of peer communication on work quality is not impossible as knowledge/strategy that workers may learn in one task can potentially be transferred to another task.

3.4 Experimental Procedure

Enabling synchronous work among crowd workers is quite challenging, as discussed in previous research on real-time crowdsourcing [3, 5]. We address this challenge by dynamically matching pairs of workers and sending them to simultaneously start working on the same sequence of tasks. In particular, when each worker arrived at our HIT, we first checked whether there was another worker in our HIT who didn't have a co-worker yet—if yes, she would be matched to that worker and assigned to the same treatment and task sequence as that worker, and the pair then started working on their sequence of tasks together. Otherwise, the worker would be *randomly* assigned to one of the two treatments as well as a *random* sequence of tasks, and she would be asked to wait for another co-worker to join the HIT for a maximum of 3 minutes. In the case where no other workers arrived at our HIT within 3 minutes, we asked the worker to decide whether she was willing to complete all tasks in the HIT on her own (and we dropped the data for the analysis but still paid her accordingly) or get a 5-cent bonus to keep waiting for another 3 minutes.

We provided a base payment of 60 cents for all our HITs. In addition to the base payments, workers were provided with the opportunity to earn performance-based bonuses, that is, workers can earn a bonus of 10 cents in a task if the final answer they submit for that task is correct. Our experiment HITs were open to U.S. workers only, and each worker was only allowed to take one HIT for each type of tasks.

3.5 Experimental Results

In total, we have 388, 382, and 250 workers who successfully formed pairs and completed the image labeling, OCR, and audio transcription tasks in our experiments, respectively. We then answer Questions 1 and 2 separately for each type of task by analyzing experimental data from Session 1 and 2 in the HIT, respectively.[5]

[3] http://www.voxforge.org

[4] For example, image labeling tasks are all about the key concept of distinguishing Siberian Husky from Malamute, OCR tasks have similar image quality, and audio transcription tasks contain similar accents.

[5] On a side note, analyzing the data collected in Session 3 leads to conclusions that are consistent with our findings reported below, and including such data only strengthens our results. However, since we have decided not to use it in the experiment design phase, we do not include the data in the analysis. The reason of the decision is that

3.5.1 Work Quality Metrics. For all three types of tasks, we evaluate the work quality using the notion of *error*. In the image labeling task, we define error as the binary classification error—the error is 0 for correct labels and 1 for incorrect labels. For OCR and audio transcription tasks, we define error as the edit distance between the worker's answer and the correct answer, divided by the number of characters in the correct answer. Naturally, for all tasks, a lower rate of error implies higher work quality.

3.5.2 Q1: Peer Communication Improves Work Quality. In Figure 2, We plot the average error rate for workers' final answers in the first two tasks (i.e., Session 1) of Treatment 1 and 2 using white and black bars, respectively. Visually, it is clear that for all three types of tasks, the work quality is higher in discussion tasks (i.e., Session 1 of Treatment 2 HITs) when workers are able to communicate with others about the work, compared to that in independent tasks (i.e., Session 1 of Treatment 1 HITs) where workers need to complete the work on their own. Indeed, we observe a substantial 37%, 32%, and 47% deduction in the average error rate for image labeling, OCR, and audio transcription tasks when peer communication is enabled. We further conduct two-sample t-tests to check whether these changes are statistically significant, and p-values are 2.42×10^{-4}, 5.02×10^{-3}, and 1.95×10^{-11} respectively, suggesting that introducing peer communication in crowd work can significantly improve the work quality produced for various types of tasks.

We then look into the chat logs to gain some insights on how and what workers have communicated with each other during the discussion. On average, the length of the discussions in image labeling, OCR and audio transcription tasks are 4.2, 5.1 and 5.4 turns[6], yet the amount of discussion is not correlated to the quality of worker's final answers after discussion. Furthermore, by looking into the content of discussions, we find several types of information workers are exchanging during their communication:

- *Providing Justification*: e.g., "triangle ears that stand erect are traits of a Siberian Husky" (image labeling)
- *Communicating Confidence*: e.g., "I'm pretty sure about UR to start, but not very sure after that" (OCR); "I had no idea what the last word was" (audio transcription)
- *Exchanging Strategy*: e.g., "If you can zoom in on it you will see what I mean" (OCR); "He pronounces 'was' with a v-sound instead of the w-sound" (audio transcription)
- *Expressing Agreement*: e.g., "I agree"; "Listening to it again, I think you are right" (audio transcription)
- *Collaborative Work*: the pair of workers work together to solve the task, e.g., guessing a digit on the car plate for the OCR task that neither worker can recognize independently

Interestingly, as an anecdotal observation, we notice that in image labeling tasks the majority of workers tend to provide justifications for their answers. In OCR and audio transcription tasks, instead of "defending" their own answers, many more workers choose to team up with their co-workers to solve the task together.

workers' conditions in Session 3 of the two treatments differ to each other both in terms of whether they have communicated with other workers about the work in previous tasks and whether they can communicate with other workers in the current tasks, making it difficult to draw any causal conclusions on the effect of peer communication.

[6]We count each chunk of sentences a worker entered in the chat room as a "turn."

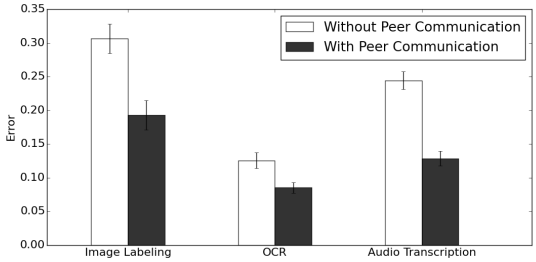


Figure 2: Comparisons of work quality produced in tasks with or without peer communication. Error bars indicate the mean ± one standard error.

3.5.3 Q2: There are no spillover effects. We now move on to Q2: Compared to workers who always complete tasks independently, do workers who have participated in tasks with peer communication continue to produce work of higher quality in future tasks of the same type, even if they need to complete those tasks on their own? To answer this question, we compare the work quality produced in Session 2 (i.e., the middle two independent tasks) of the two treatments for all three types of tasks. For image labeling, OCR, and audio transcription tasks, the average error rates for Session 2 in Treatment 1 (workers never engage in peer communication) are 0.324, 0.175, and 0.209 respectively, while the average error rates for Session 2 in Treatment 2 (workers have previously engaged in peer communication) are 0.334, 0.168, and 0.244. Thus, we do not observe any spillover for the effects of peer communication on work quality, that is, the quality improvement brought up by peer communication does not carry on to future independent work.

3.5.4 Discussions. Results of our experimental study suggest that peer communication improves the quality of crowd work for various types of tasks, even when the number of possible answers in the task is very large, yet such effect does not spill over to later independent work. Cautions should be used when generalizing these results to substantially different contexts, such as when workers can interact with each other for an extended period of time rather than just 2 minutes, when the tasks are significantly more complex or more subjective, or when workers engage in peer communication for a longer sequence of tasks. It is, thus, an important future direction to obtain an thorough understanding on how tuning various parameters of the design space (e.g., length of interactions, complexity/subjectivity of tasks) would change the effects of peer communication.

4 AN ALGORITHMIC FRAMEWORK FOR UTILIZING PEER COMMUNICATION

Our experimental study focuses on understanding the impact of peer communication on *individual tasks*. Now, we turn to our next question, that is, for a requester who has a *large batch* of tasks, how can he better utilize peer communication to improve the *overall utility* that he can obtain across all the tasks? In particular, we address the following research question: *Given a budget and a batch of tasks to complete, whether and when should a requester deploy peer communication in each task to maximize his total utility?*

To answer this question, there are two main challenges. First, when peer communication is deployed, workers communicate with

each other before submitting their answers. Therefore, their answers might be correlated. Yet, existing aggregation methods all assume each worker complete the work independently, making it necessary for us to develop new ways to address the data correlation issue. Second, deploying peer communication incurs higher cost, since it requires us to hire workers in pairs to work for longer period of time and may need additional effort for worker synchronizations. Therefore, even though peer communication produces higher work quality for individual tasks, it is not clear deploying peer communication is always beneficial for the overall utility.

In this section, we focus on the setting in which a requester aims to collect labels for a batch of binary classification tasks with a fixed budget, and the "utility" to maximize is the average accuracy the requester obtains across all classification tasks[7]. In the following, we first discuss how to deal with the data correlation issue, that is, how to infer the correct label for a task given multiple labels solicited from workers, where some of the labels may be correlated. We then describe our algorithmic framework, a constrained Markov decision process, which adaptively decides whether and when peer communication should be deployed in each task under the budget constraint while taking into account data correlation and differing cost in deploying peer communication.

4.1 Dealing with Data Correlation

When peer communication is used in a task, a pair of workers directly interact with each other. Naturally, their contributions (e.g., labels in image labeling tasks) might be correlated. For categorical tasks with a finite number of labels, we could use the covariance notion to measure the correlation of workers' contributions. Formally, let X, Y be the random variables representing the answers generated by a pair of workers for the same task. The correlation of workers' answers can be formulated using covariance $cov(X, Y)$, defined as $cov(X, Y) = E[XY] - E[X]E[Y]$. By definition, when a pair of answers X, Y are independent, the covariance should be 0.

4.1.1 Measuring Data Correlation. To see whether the answers from a pair of workers are correlated when they work together on a task with peer communication, we examine workers' answers in Session 1 of both treatments in our experiments on image labeling tasks. For each of the 20 images in the experiments (with labels in $\{0, 1\}$), we calculate the covariance between pairs of labels generated in independent tasks (Session 1 of Treatment 1) and discussion tasks (Session 1 of Treatment 2)[8], in which we use the empirical average to replace the expectation in the definition. The results are shown in Figure 3. Perhaps not surprisingly, data collected in independent tasks is mostly independent (with covariance close to 0), while data collected in discussion tasks is correlated to various degrees.

We also calculate the covariance of workers' answers in Session 2 of both treatments to see if the data correlation caused by peer communication has any spillover effect. We find the covariance is close to 0 for both treatments, indicating the correlation in data caused by peer communication does not carry on to later independent work.

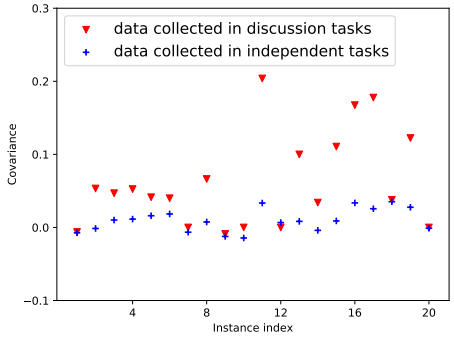


Figure 3: Covariance for data collected in independent tasks and discussion tasks in Session 1 in the image labeling HITs.

4.1.2 Meta-Workers and Meta-Labels. The above observations confirm that workers' answers are indeed correlated when peer communication is deployed. To deal with data correlation, we introduce the notions of *meta-workers* and *meta-labels*. In particular, we denote a pair of workers who talk with each other through the peer communication procedure as a meta-worker, and the pair of labels they generate as a meta-label. As no communication happens between different pairs of workers, we assume each meta-label is drawn independently.

Formally, for a binary classification task, let the true label $z \in \{0, 1\}$. When peer communication is deployed in a task, we obtain a pair of labels, which can be $\{1, 1\}$, $\{0, 1\}$, or $\{0, 0\}$, and we use the *meta-label* s_{11}, s_{01}, and s_{00} to denote them, respectively. Moreover, denote s_1 and s_0 as the label 1 and 0 obtained from a single worker who works independently.

To simplify the discussion, we assume workers are homogeneous. We propose a model to characterize the correlation in data produced in tasks with peer communication as follows: Denote p as the probability of independent workers providing correct labels, i.e., $p = P(s_1|z = 1) = P(s_0|z = 0)$. Additionally, we denote p_+, p_0, p_- as the probability for workers in tasks with peer communication to contribute two correct labels, one correct and one incorrect label, and two incorrect labels[9]:

$$p_+ = P(s_{11}|z = 1) = P(s_{00}|z = 0)$$
$$p_- = P(s_{00}|z = 1) = P(s_{11}|z = 0)$$
$$p_0 = P(s_{01}|z = 1) = P(s_{01}|z = 0)$$

This model provides a principled way to capture different levels of correlation. For example, when the pair of labels are independent, and the probability for each worker in the pair to submit a correct label is still p, we should have $p_+ = p^2, p_- = (1 - p)^2$, and $p_0 = 2p(1 - p)$. When the correlation between a pair of labels is 1 (i.e., the two labels are always the same), we have $p_0 = 0$.

4.1.3 Utilizing Meta-labels. The idea of introducing meta-workers and meta-labels are intuitive but powerful. Below we use maximum likelihood aggregation as an example to demonstrate how the

[7] Our discussion can be extended to classification tasks with any finite number of labels. Extending our results to general types of tasks (such as transcription tasks) requires a well-defined utility notion that can quantify the total utility for any given set of worker contributions. It is an interesting and important future direction.

[8] Recall that in our experiment, we always send a pair of workers to work on the same sequence of tasks. Thus, an independent task is also completed by a pair of workers except that they don't communicate with each other. This allows us to directly calculate the covariance for labels generated in independent tasks.

[9] This is the extension to the standard one-coin model in crowdsourcing literatures. Extending the discussion to model the confusion matrix (e.g., using two different probability values for $P(s_{11}|z = 1)$ and $P(s_{00}|z = 0)$) is straightforward. We do not include the discussion here due to space constraints.

concepts of meta-workers and meta-labels can be incorporated in standard aggregation methods, which provides us with key insights on how to utilize these concepts in our algorithmic framework (we will detail this in Section 4.2).

For a task with unknown true label $z \in \{0, 1\}$, given a set of N labels (or meta-labels) $\boldsymbol{L} = \{l_1, ..., l_N\}$, where $l_i \in \{s_{11}, s_1, s_{01}, s_0, s_{00}\}$, the maximum likelihood estimator for the value of z is defined as:

Definition 1. *Let the ground truth of the task be z. Given a set of labels $\boldsymbol{L} = \{l_1, \ldots, l_N\}$. $\hat{z}$ is a maximum likelihood estimator if*

$$\hat{z} = \begin{cases} 1 & \text{if } P(\boldsymbol{L}|z = 1) \geq P(\boldsymbol{L}|z = 0), \\ 0 & \text{otherwise.} \end{cases}$$

We assume p, p_+, p_0, and p_- are all known. Note that in our algorithmic framework as explained in Section 4.2, we adopt a Bayesian setting to learn how to aggregate the data over time without prior knowledge on values of these parameters. However, when such prior knowledge is available, a weighted majority voting rule can lead to maximum likelihood estimation:

Lemma 1. *Given a set of labels $\boldsymbol{L}$. Let $n_{11}, n_1, n_{01}, n_0, n_{00}$ denote the number of labels $s_{11}, s_1, s_{01}, s_0, s_{00}$ in $\boldsymbol{L}$. Consider the following weighted majority voting rule that generates an aggregation $\hat{z}$*

$$\hat{z} = \begin{cases} 1 & \text{if } w_{11}n_{11} + w_1 n_1 \geq w_{00}n_{00} + w_0 n_0 \\ 0 & \text{if } w_{11}n_{11} + w_1 n_1 < w_{00}n_{00} + w_0 n_0 \end{cases}$$

This weighted majority voting rule leads to maximum likelihood estimation when the weights are set as: $w_{11} = w_{00} = \ln \frac{p_+}{p_-}$, and $w_1 = w_0 = \ln \frac{p}{1-p}$.

Proof. We can write the probabilities on both sides as follows:

$$P(\boldsymbol{L}|z = 1) = p_+^{n_{11}} p^{n_1} p_0^{n_{01}} (1-p)^{n_0} p_-^{n_{00}}$$
$$P(\boldsymbol{L}|z = 0) = p_-^{n_{11}} (1-p)^{n_1} p_0^{n_{01}} p^{n_0} p_+^{n_{00}}$$

Therefore, we have

$$\frac{P(\boldsymbol{L}|z = 1)}{P(\boldsymbol{L}|z = 0)} = \left(\frac{p_+}{p_-}\right)^{n_{11}} \left(\frac{p}{1-p}\right)^{n_1} \left(\frac{1-p}{p}\right)^{n_0} \left(\frac{p_-}{p_+}\right)^{n_{00}}$$

Note that, in maximum likelihood estimator, $\hat{z} = 1$ if $P(\boldsymbol{L}|z = 1)/P(\boldsymbol{L}|z = 0) \geq 1$. Therefore, $\hat{z} = 1$ if

$$\left(\frac{p_+}{p_-}\right)^{n_{11}} \left(\frac{p}{1-p}\right)^{n_1} \geq \left(\frac{p}{1-p}\right)^{n_0} \left(\frac{p_+}{p_-}\right)^{n_{00}}$$

The proof is completed by taking logarithm on both sides. □

As a sanity check, we can see that when a pair of labels are independent (i.e., $p_+ = p^2$ and $p_- = (1-p)^2$), we have $w_{11} = w_{00} = 2w_1 = 2w_0$, implying that the weight of $\{1, 1\}$ label is twice as the weight of $\{1\}$ label, and this is essentially a simple majority voting.

Note that in the maximum likelihood aggregation, the number of meta-label s_{01} does *not* play a role in the aggregation process. In other words, we may interpret the generation of meta-labels as follows: with probability p_+ (or p_-), a meta-worker generates a correct label s_{11} (or incorrect label s_{00}), while with probability p_0 she generates *no* label at all. The above weighted majority voting rule then simply indicates that different weights need to be used for labels generated by independent workers or meta-workers. Following a similar idea, in the following algorithmic framework, we only take the meta-label s_{00} and s_{11} into consideration and discard the meta-label s_{01} when inferring the correct labels of a task from a collection of labels and meta-labels.

4.2 Our Algorithmic Framework

With the notions of meta-workers and meta-labels in place, we have a principled way to deal with correlated data in peer communication. However, we still need to address the second challenge of balancing the quality and cost. In particular, while introducing peer communication leads to a significant improvement in work quality for individual microtasks, such improvement comes with extra cost, such as the financial payment incurred to recruit more workers (e.g., at least two workers are needed for peer communication to happen), the compensation for longer task completion time due to discussions, and the additional administrative costs for synchronizing the work pace of worker pairs. As a result, a requester needs to face the quality-cost tradeoff when deploying peer communication.

We now describe our algorithmic framework, built on the constrained Markov decision process (CMDP), that adaptively decides for a requester with a limited budget, whether and when peer communication should be deployed in each of his tasks with the goal of maximizing his total utility (i.e., the average accuracy for all classification tasks), while taking into account data correlation and differing costs for deploying peer communication.

4.2.1 Problem Setup. Our problem setup is inspired by the method by Chen et al. [8] to optimally allocate budget among task instances in crowdsourcing data collection. Our setup differs from theirs in two fundamental ways due to the presence of peer communication strategy. First, they don't and don't need to consider the issue of data correlation. Second, in their setting, the cost for acquiring labels is fixed, while we need to deal with the differing costs when peer communication is deployed in a task. Therefore, instead of modeling the decision-making problem as a Markov decision process framework (as in Chen et al. [8]), we adopt a constrained Markov decision process framework and include the meta-label concept in our formulation.

Formally, suppose a requester gets a budget of $\mathcal{B}$ and a batch of K binary classification tasks, and he needs to estimate the label for each of these tasks. The goal of the requester is to maximize the average accuracy of the estimated labels across all tasks through spending the budget to solicit labels from crowd workers and then aggregating the collected labels. We describe the setting in which workers are homogeneous (however, their performance might be different when working independently or when working with peer communication). Extensions to settings with heterogeneous workers are straightforward as described by Chen et al. [8].

Assume the K tasks are independent from each other, and $Z_k \in \{0, 1\}$ represents the true label for task k ($1 \leq k \leq K$). We use the notations $\theta_k \in [0, 1]$, $\alpha_s \in [0, 1]$, and $\alpha_p \in [0, 1]$ to model the label generation process, where θ_k characterizes the difficulty of task k, α_s and α_p characterize workers' performance when working independently and working with peer communication. In particular, we denote $p_{k,s,1}$ (or $p_{k,s,0}$) as the probability for a single worker (who works independently) to provide label 1 (or 0) for task k. We define $p_{k,s,1} = \alpha_s\theta_k + (1-\alpha_s)(1-\theta_k)$ and $p_{k,s,0} = 1 - p_{k,s,1}$. To obtain intuition for the parameters of the model, assume $\alpha_s = 1$, we can see that θ_k captures the difficulty of task k: When θ_k is

close to 0.5, workers are effectively making random guess (hence the task is difficult), and when θ_k is close to 0 or 1, independent workers can consistently provide the same label (hence the task is easy). Similarly, α_s can then be interpreted as the worker skill and a larger α_s implies a higher skill. We assume θ_k is consistent with the label Z_k, which means $Z_k = 1$ if and only if $\theta_k \geq 0.5$.

Recall that we denote a meta-worker as a pair of workers in tasks with peer communication. We use α_p to denote the skill of meta-workers, and the probability for a meta-worker to generate a meta-label s_{01} for task k is denoted as q_k. Conditioned on a meta-worker contributing a meta-label other than s_{01}, α_p is similarly defined as α_s. That is, when $p_{k,p,1}$ and $p_{k,p,0}$ is the probability for a meta-worker to generate meta-labels s_{11} and s_{00}, we have $p_{k,p,1} = (1-q_k)(\alpha_p\theta_k + (1-\alpha_p)(1-\theta_k))$ and $p_{k,p,0} = 1 - p_{k,p,1} - q_k$.

After describing the data generation model, we formulate the online decision problem faced by the requester. The requester recruits workers to label his tasks in a sequential manner. Specifically, at each time step t, the requester decides on a task k_t to work on, and he can solicit label(s) from crowd workers on this task using one of the two strategies (the strategy is denoted as x_t): first, the requester can recruit a *single* worker to work on the task ($x_t = 0$), and thus obtain a label for that task; second, the requester may recruit a meta-worker (i.e., a *pair* of workers following the peer communication procedure) to work on the task ($x_t = 1$), and thus obtain a meta-label for the task. We denote c_s as the cost for recruiting a single worker and c_p ($c_p > c_s$) as the cost of recruiting a meta-worker through peer communication strategy.

Naturally, the requester's activity in each time step can be summarized through the tuple (k_t, x_t). We also denote y_t as the label (or meta-label) obtained by the requester at time t for task k_t. By the time $t_{\mathcal{B}}$ that the requester exhausts his budget, his activity history is $\mathcal{H}_{\mathcal{B}} = \{(k_0, x_0, y_0), ..., (k_{t_{\mathcal{B}}}, x_{t_{\mathcal{B}}}, y_{t_{\mathcal{B}}})\}$. The requester then aggregates the data he has collected and infers the true labels for each of the K tasks such that the expected accuracy across all K tasks, conditioned on the activity history $\mathcal{H}_{\mathcal{B}}$, is maximized.

4.2.2 A Constrained Markov Decision Process Formulation. We now formally model the requester's decision-making problem as a constrained Markov decision process:

- **States**: the state s_t is a $K \times 4$ matrix, where $s_t(k, \cdot)$ is a 1×4 vector with each entry representing before time t, the number of label (or meta-labels) s_0, s_1, s_{00}, s_{11} obtained for task k. Note that following the idea that we have discussed in Section 4.1.3, we consider the meta-label s_{01} to contributes zero utility to the requester and thus we do not include the count of it in the state.
- **Actions**: $a_t = (k_t, x_t)$, where k_t is the task to work on at time t, and $x_t \in \{0, 1\}$ represents the worker recruiting strategy, with 0 being recruiting a single worker working independently and 1 being recruiting a pair of workers to follow the peer communication procedure.
- **Transition probabilities**: When $a_t = (k_t, x_t = 0)$,

$$Pr(s_{t+1}|s_t, a_t) = \begin{cases} p_{k_t,s,1} & \text{if } s_{t+1} = s_t + (\mathbf{0}, \mathbf{e_{k_t}}, \mathbf{0}, \mathbf{0}) \\ p_{k_t,s,0} & \text{if } s_{t+1} = s_t + (\mathbf{e_{k_t}}, \mathbf{0}, \mathbf{0}, \mathbf{0}) \\ 0 & \text{otherwise} \end{cases}$$

where $\mathbf{e_{k_t}}$ is a $K \times 1$ vector with value 1 at the k_t-th entry and 0 at all other entries. On the other hand, when $a_t = (k_t, x_t = 1)$,

$$Pr(s_{t+1}|s_t, a_t) = \begin{cases} p_{k_t,p,1} & \text{if } s_{t+1} = s_t + (\mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{e_{k_t}}) \\ 1 - p_{k_t,p,1} - q_{k_t} & \text{if } s_{t+1} = s_t + (\mathbf{0}, \mathbf{0}, \mathbf{e_{k_t}}, \mathbf{0}) \\ q_{k_t} & s_{t+1} = s_t \\ 0 & \text{otherwise} \end{cases}$$

- **Rewards**: We adopt the same reward function as that used by Chen et al. [8]. Specifically, we assume the parameters $\theta_k, \alpha_s, \alpha_p$ are sampled from three separate Beta prior distributions, and we update the posteriors of these distributions through variational approximation where hyper-parameters are decided by moment matching. Doing so, we can then define the reward as $R(s_t, a_t) = \mathbb{E}(h(P_{k_t}^{t+1}) - h(P_{k_t}^t))$, where P_k^t is the probability of the parameter θ_k taking on a value of at least 0.5 given the posterior of θ_k at time t, $h(x) = \max(x, 1-x)$, and the expectation is taken over all possible label y_t observed after action a_t.
- **Constraint**: Different from the setting in the work by Chen et al. [8], as different actions imply different costs, we need to explicitly characterize the budget constraint for our problem. Formally, the requester needs to ensure the budget constraint is satisfied. $\sum_{t=0}^{t_{\mathcal{B}}} c_s \mathbf{1}(x_t = 0) + c_p \mathbf{1}(x_t = 1) \leq \mathcal{B}$, where $\mathbf{1}(.)$ is the indicator function.

4.2.3 Proposed Algorithm. We adopt the method of Lagrangian multipliers to solve the above constrained optimization problem, which converts the problem of maximizing the total reward (i.e., $\sum_{t=0}^{t_{\mathcal{B}}} R(s_t, a_t)$) under the budget constraint into a simpler problem of maximizing the auxiliary function $\sum_{t=0}^{t_{\mathcal{B}}} R(s_t, a_t) - \lambda \sum_{t=0}^{t_{\mathcal{B}}} (c_s \mathbf{1}(x_t = 0) + c_p \mathbf{1}(x_t = 1))$. Notice this optimization problem is equivalent to solve a (unconstrained) Markov decision process where reward in each step is redefined as $R'(s_t, a_t) = R(s_t, a_t) - \lambda(c_s \mathbf{1}(x_t = 0) + c_p \mathbf{1}(x_t = 1))$. We use the optimistic knowledge gradient technique introduced by Chen et al. [8] to solve the optimal policy of this MDP, which produces a single-step look-ahead policy that maximizes the highest reward at each step. Note that in theory, we can compute the optimal value of λ by solving the dual of the constrained MDP. In practice, we have experimented with multiple different λ values and find that the choice of λ has limited influence on the performance of our algorithmic approach.

4.3 Evaluations

We evaluate the effectiveness of our algorithmic approach using both real-world data and synthetic data.

4.3.1 Experiments on Real Data. Using the real data that we collected in image labeling tasks of our experimental study, we compare the performance of our algorithm with a couple of baseline algorithms. In our evaluation, we set the cost of recruiting a single worker as $c_s = 1.0$ and the cost of recruiting a pair of workers to work on a task with peer communication (i.e., a meta-worker) $c_p = 2.5$. Note that we have examined a range of different values of c_p from 1.5 to 3.5 and the results are qualitatively similar. The prior distribution for θ_k is set as Beta(1, 1), where the prior distributions for α_s and α_p are all set to be Beta(4, 1). For this evaluation, we only considered the final labels that workers in our experimental

study submit in the *first two tasks* of the image labeling HIT[10]. Thus, when $a_t = (k_t, 0)$, we randomly sampled a label from Treatment 1 workers who had completed task k_t in their first two (independent) tasks, and when $a_t = (k_t, 1)$, we randomly sampled a label from Treatment 2 workers who had completed task k_t in their first two (discussion) tasks.

The performance of our algorithmic approach is compared against the following baseline approaches:

- *Round robin*: in each round, the requester decides which task to work on in a round robin fashion, and he always recruit a single worker to work on that task independently.
- *Single workers only*: in each round, the requester recruits a single worker to work on a task independently, and this task is optimally decided (effectively by considering only actions with $x_t = 0$ in our algorithm).
- *Peer communication only*: in each round, the requester recruits a pair of workers to work on a task with peer communication, and this task is optimally decided (effectively by considering only actions with $x_t = 1$ in our algorithm).
- *Our algorithm [No correlation]*: in each round, the requester uses our algorithm to decide whether to deploy peer communication and which task to work. The only difference is that this baseline treats the two labels from peer communication as two independent labels while our algorithm incorporates the concept of meta-labels to deal with data correlation.

We conduct this evaluation on a range of budget level from 20 to 400 with an interval of 20. At each budget level, we implement each of the decision-making strategies for 100 times, and we report the average level of overall accuracy the requester obtains across the 20 tasks when she exhausts the budget in Figure 4.

As shown in Figure 4, our proposed algorithm outperforms all baseline strategies. In particular, we make a few observations as follows. First, comparing the performance of our algorithm and that of the "No Correlation" strategy, it is clear that incorporating meta-labels to deal with data correlation has improved the requester's overall utility. In fact, even for the "peer communication only" strategy, we also implement two versions, and the version for which the idea of meta-labels is used also outperforms the other version treating two labels generated by pairs of workers as independent. In the following discussion, unless otherwise specified, we adopt the meta-worker ideas in our implementation when peer communication is used. Second, strategies involving peer communication converge to a better overall accuracy than strategies without peer communication does. This is due to the fact that for some tasks in our experiment, the majority of workers who work independently provide incorrect answers while the majority of workers with peer communication provide correct answers. Third, adaptively determining whether and when to deploy peer communication outperforms fixed recruiting strategies, as illustrated by the superior performance of our algorithm over both the "single workers only" and "peer communication only" strategies. Finally, adaptively deciding which task to label next significantly improves the total utility than random task assignment does, e.g., through observing the significantly worse performance of the baseline "round robin" strategy.

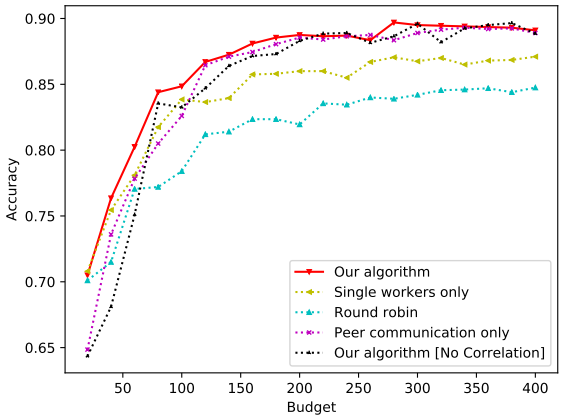


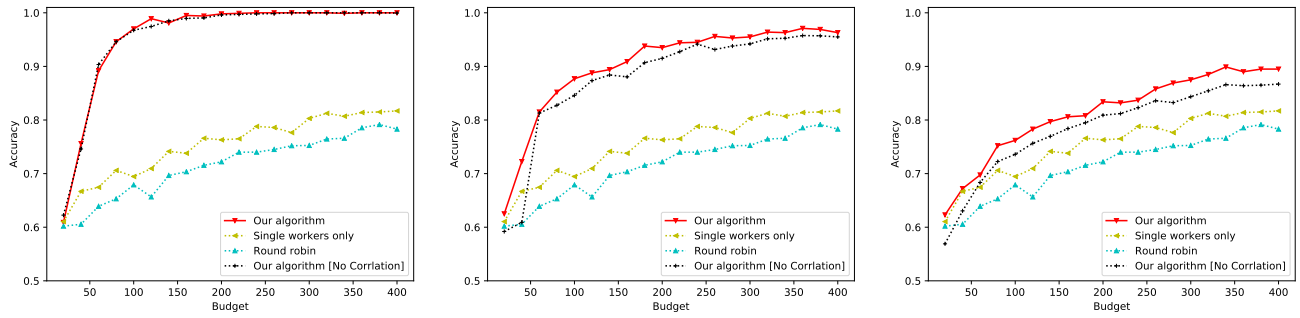
Figure 4: Evaluating the performance of the proposed approach on real datasets.

4.3.2 Experiments on Synthetic Data. To the best of our knowledge, our dataset is the only dataset that deploys peer communication for crowdsourcing data collection. Therefore, to further investigate the properties of our proposed algorithm, we generate synthetic data to evaluate our algorithm. In particular, we explore how the performance of our algorithm changes along the following three dimensions: 1) the level of data correlation of workers' answers in tasks with peer communication, 2) the performance gap between workers who work independently and workers who discuss with others via peer communication, and 3) the cost differences of hiring a single worker and hiring a pair of workers for peer communication. In the base setup, we set θ_k to be uniformly drawn from $[0.5, 1]$, α_s drawn from a normal distribution with mean 0.7 and variance 0.01, α_p drawn from a normal distribution with mean 0.9 and variance 0.1. We also set $c_s = 1$ and $c_p = 2.5$.

We first modify the level of data correlation of workers' answers in peer communication. This can be done by changing the value of q_k, i.e., the probability of a pair of workers in peer communication to generate the meta-label s_{01}. In strong correlation, we set $q_k = 0$, which means the two workers are entirely correlated (always generating the same label). In no correlation, we set q_k to the value such that the two labels in a meta-label are independently generated (i.e., $q_k = 2\sqrt{p_{k,p,1}p_{k,p,0}}$, which can be derived using our model discussed in Section 4.1.2). In weak correlation, q_k is uniformly drawn between the above two values. We compare the performance between our algorithm and "our algorithm [no correlation]", which treats the two labels from a meta-label as independent labels. As shown in Figure 5, the performance gap becomes larger as the correlation becomes stronger[11]. This validates the benefits of incorporating meta-labels in our framework when there is data correlation.

[10]Recall that we set out to examine the effects of peer communication using the first two tasks in each HIT.

[11]As a side note, the overall performance is lower in strong correlation since we fixed α_p in all three plots; two independent labels brings more information than two correlated labels. Since our goal is to measure the gap between two algorithms, we didn't tune the parameter to normalize the algorithm performance.



(a) No Correlation (b) Weak Correlation (c) Strong Correlation

Figure 5: The performance comparison under different levels of correlation in peer communication.

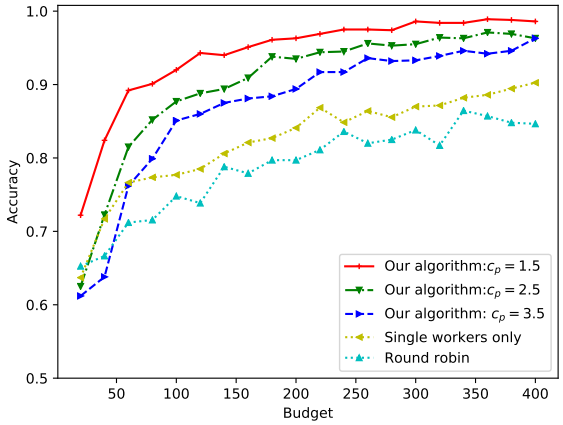


Figure 6: Modify the cost of peer communication.

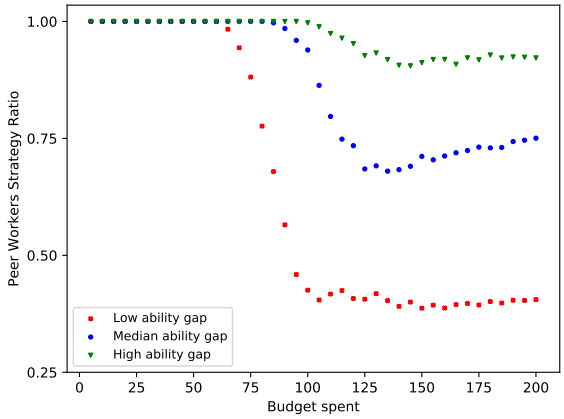


Figure 7: Ratio of peer communication strategies deployed.

We then change the cost of deploying peer communication c_p to be from $\{1.5, 2.5, 3.5\}$. As shown in Figure 6, our algorithm performance decreases as c_p increases. However, even when the cost of peer communication is pretty large (i.e., $c_p = 3.5$), utilizing peer communication is still beneficial. Next, we vary the performance gap between single workers (who work independently) and meta-workers by fixing the mean of α_p to be 0.9 and set the mean of α_s to be 0.8, 0.7, and 0.6. The results are qualitatively similar to changing c_p (e.g., larger skill gap corresponds to smaller c_p). To provide more insights for our algorithm, we demonstrate this result in a different plot. In particular, we fix the budget to be 200 and run our algorithm 100 times. We record the worker recruiting strategy (hiring single workers or deploying peer communication) our algorithm takes at every step, and then calculate the ratio of peer communication strategy as a function of the budget spent so far. As shown in Figure 7, our algorithm always starts by deploying peer communication. When the marginal rewards for hiring peer communication is not high enough to justify the higher cost, our algorithm gradually switches to hire single workers. These two figures demonstrate that our algorithm brings in benefits under a wide range of settings and has stronger benefits when c_p is small or when the performance gap between single workers and meta-workers (with peer communication) are higher.

5 CONCLUSION

In this paper, we relax the common assumption of data independence in crowdsouring data collection. In particular we explore peer communication, in which a pair of crowd workers directly communicate when producing the data. We first examine the effects of peer communication on the quality of crowd work produced. Randomized experiments conducted on Amazon Mechanical Turk demonstrate that the work quality significantly improves in tasks with peer communication. We then study how to utilize peer communication in crowdsourcing data collection. In particular, we introduce the notions of meta-workers and meta-labels to deal with data correlation caused by peer communication. We then develop an algorithmic framework, built on constrained Markov decision process, to optimally determine whether and when to deploy peer communication in crowdsourcing data collection, with the goal of maximizing the total utility of requesters while satisfying budget constraints. Experiments conducted on both real data and synthetic data demonstrate the advantage of our proposed algorithms over baseline approaches.

Our results suggest the potential benefits of incorporating peer communication in crowdsourcing and provide a framework for better utilizing these benefits. We hope this work could open more discussions on designing and leveraging more complex, useful worker interactions to further enhance crowdsourcing.

REFERENCES

[1] Alan Aipe and Ujwal Gadiraju. 2018. SimilarHITs: Revealing the Role of Task Similarity in Microtask Crowdsourcing. In *Proceedings of the 29th on Hypertext and Social Media*.

[2] Bahador Bahrami, Karsten Olsen, Peter E Latham, Andreas Roepstorff, Geraint Rees, and Chris D Frith. 2010. Optimally interacting minds. *Science* 329, 5995 (2010), 1081–1085.

[3] Michael S. Bernstein, Joel Brandt, Robert C. Miller, and David R. Karger. 2011. Crowds in Two Seconds: Enabling Realtime Crowd-powered Interfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST)*.

[4] Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. 2010. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. ACM, 313–322.

[5] Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C. Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, and Tom Yeh. 2010. VizWiz: Nearly Real-time Answers to Visual Questions. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology (UIST)*.

[6] Carrie J. Cai, Shamsi T. Iqbal, and Jaime Teevan. 2016. Chain Reactions: The Impact of Order on Microtask Chains. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*.

[7] Joseph Chee Chang, Saleema Amershi, and Ece Kamar. 2017. Revolt: Collaborative Crowdsourcing for Labeling Machine Learning Datasets. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI)*.

[8] Xi Chen, Qihang Lin, and Dengyong Zhou. 2013. Optimistic knowledge gradient policy for optimal budget allocation in crowdsourcing. In *International Conference on Machine Learning (ICML)*.

[9] Lydia B Chilton, Greg Little, Darren Edge, Daniel S Weld, and James A Landay. 2013. Cascade: Crowdsourcing taxonomy creation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1999–2008.

[10] Sharath R. Cholleti, Sally A. Goldman, Avrim Blum, David G. Politte, and Steven Don. 2008. Veritas: Combining expert opinions without labeled data. In *Proceedings 20th IEEE international Conference on Tools with Artificial intelligence (ICTAI)*.

[11] Derrick Coetzee, Seongtaek Lim, Armando Fox, Bjorn Hartmann, and Marti A Hearst. 2015. Structuring interactions for large-scale synchronous peer learning. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. ACM, 1139–1152.

[12] Catherine Crouch and Eric Mazur. 2001. Peer instruction: Ten years of experience and results. *Am. J. Phys.* 69, 9 (September 2001), 970–977.

[13] Peng Dai, Christopher H. Lin, Mausam, and Daniel S. Weld. 2013. POMDP-based Control of Workflows for Crowdsourcing. *Artif. Intell.* 202, 1 (Sept. 2013), 52–85.

[14] A. P. Dawid and A. M. Skene. 1979. Maximum likeihood estimation of observer error-rates using the EM algorithm. *Applied Statistics* 28 (1979), 20–28.

[15] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B* 39 (1977), 1–38.

[16] Louis Deslauriers, Ellen Schelew, and Carl Wieman. 2011. Improved learning in a large-enrollment physics class. *science* 332, 6031 (2011), 862–864.

[17] Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, Panagiotis G Ipeirotis, and Philippe Cudré-Mauroux. 2015. The dynamics of micro-task crowdsourcing: The case of amazon mturk. In *Proceedings of the 24th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 238–247.

[18] Ryan Drapeau, Lydia B. Chilton, Jonathan Bragg, and Daniel S. Weld. 2016. MicroTalk: Using Argumentation to Improve Crowdsourcing Accuracy. In *Fourth AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*.

[19] Mary L Gray, Siddharth Suri, Syed Shoaib Ali, and Deepti Kulkarni. 2016. The crowd is a collaborative network. In *Proceedings of the 19th ACM conference on computer-supported cooperative work & social computing*. ACM, 134–147.

[20] Neha Gupta, David Martin, Benjamin V Hanrahan, and Jacki O'Neill. 2014. Turk-life in India. In *Proceedings of the 18th International Conference on Supporting Group Work*. ACM, 1–11.

[21] Danna Gurari and Kristen Grauman. 2017. CrowdVerge: Predicting If People Will Agree on the Answer to a Visual Question. In *Proceedings of the 2017 Conference on Human Factors in Computing Systems (CHI)*.

[22] Chien-Ju Ho, Shahin Jabbari, and Jennifer Wortman Vaughan. 2013. Adaptive Task Assignment for Crowdsourced Classification. In *The 30th International Conference on Machine Learning (ICML)*.

[23] Chien-Ju Ho, Aleksandrs Slivkins, Siddharth Suri, and Jennifer Wortman Vaughan. 2015. Incentivizing High Quality Crowdwork. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*.

[24] John Joseph Horton and Lydia B. Chilton. 2010. The labor economics of paid crowdsourcing. In *Proceedings of the 11th ACM conference on Electronic commerce (EC)*.

[25] Lilly C Irani and M Silberman. 2013. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 611–620.

[26] Rong Jin and Zoubin Ghahramani. 2003. Learning with multiple labels. In *Advances in Neural Information Processing Systems (NIPS)*.

[27] David R. Karger, Sewoong Oh, and Devavrat Shah. 2011. Budget-Optimal Crowdsourcing using Low-rank Matrix Approximations. In *Proc. 49th Annual Conference on Communication, Control, and Computing (Allerton)*.

[28] David. R. Karger, Sewoong Oh, and Devavrat Shah. 2011. Iterative learning for reliable crowdsourcing systems. In *The 25th Annual Conference on Neural Information Processing Systems (NIPS)*.

[29] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. 2011. Novel Dataset for Fine-Grained Image Categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*. Colorado Springs, CO.

[30] Juho Kim, Phu Tran Nguyen, Sarah Weir, Philip J Guo, Robert C Miller, and Krzysztof Z Gajos. 2014. Crowdsourcing step-by-step information extraction to enhance existing how-to videos. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 4017–4026.

[31] Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E. Kraut. 2011. CrowdForge: Crowdsourcing Complex Work. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST)*.

[32] Anand Kulkarni, Matthew Can, and Björn Hartmann. 2012. Collaboratively Crowdsourcing Workflows with Turkomatic. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work CSCW*.

[33] Edith Law, Ming Yin, Joslin Goh, Kevin Chen, Michael A. Terry, and Krzysztof Z. Gajos. 2016. Curiosity Killed the Cat, but Makes Crowdwork Better. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI)*.

[34] Qi Li, Fenglong Ma, Jing Gao, Lu Su, and Christopher J Quinn. [n. d.]. Crowdsourcing high quality labels with a tight budget. In *Proceedings of the ninth acm international conference on Web Search and Data Mining (WSDM)*.

[35] Greg Little, Lydia B. Chilton, Max Goldman, and Robert C. Miller. 2010. TurKit: Human Computation Algorithms on Mechanical Turk. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology (UIST)*.

[36] Frank Lyman. 1987. Think-Pair-Share: An Expanding Teaching Technique. *MAA-CIE Cooperative News* (1987).

[37] David Martin, Benjamin V Hanrahan, Jacki O'Neill, and Neha Gupta. 2014. Being a turker. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. ACM, 224–235.

[38] Winter Mason and Duncane Watts. 2009. Financial Incentives and the "Performance of Crowds". In *Proceedings of the 1st Human Computation Workshop (HCOMP)*.

[39] Edward Newell and Derek Ruths. 2016. How One Microtask Affects Another. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*.

[40] Jon Noronha, Eric Hysen, Haoqi Zhang, and Krzysztof Z Gajos. 2011. Platemate: crowdsourcing nutritional analysis from food photographs. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 1–12.

[41] Vikas Raykar, Shipeng Yu, Linda Zhao, Gerardo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *Journal of Machine Learning Research* 11 (2010), 1297–1322.

[42] Daniela Retelny, Sébastien Robaszkiewicz, Alexandra To, Walter S. Lasecki, Jay Patel, Negar Rahmati, Tulsee Doshi, Melissa Valentine, and Michael S. Bernstein. 2014. Expert Crowdsourcing with Flash Teams. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST)*.

[43] Jakob Rogstadius, Vassilis Kostakos, Aniket Kittur, Boris Smus, Jim Laredo, and Maja Vukovic. 2011. An assessment of intrinsic and extrinsic motivation on task performance in crowdsourcing markets. *ICWSM* 11 (2011), 17–21.

[44] Niloufar Salehi, Lilly C Irani, Michael S Bernstein, Ali Alkhatib, Eva Ogbe, Kristy Milland, et al. 2015. We are dynamo: Overcoming stalling and friction in collective action for crowd workers. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*. ACM, 1621–1630.

[45] Mike Schaekermann, Joslin Goh, Kate Larson, and Edith Law. 2018. Resolvable vs. Irresolvable Disagreement: A Study on Worker Deliberation in Crowd Work. In *Proceedings of the 21st ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW)*.

[46] Nihar Bhadresh Shah and Denny Zhou. 2015. Double or Nothing: Multiplicative Incentive Mechanisms for Crowdsourcing. In *The 29th Annual Conference on Neural Information Processing Systems (NIPS)*.

[47] Aaron D. Shaw, John J. Horton, and Daniel L. Chen. 2011. Designing incentives for inexpert human raters. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work (CSCW)*.

[48] Michelle K Smith, William B Wood, Wendy K Adams, Carl Wieman, Jennifer K Knight, Nancy Guild, and Tin Tin Su. 2009. Why peer discussion improves student performance on in-class concept questions. *Science* 323, 5910 (2009), 122–124.

[49] Jacob Whitehill, Ting fan Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. 2009. Whose vote should count more: Optimal integration of labels

from labelers of unknown expertise. In *Advances in Neural Information Processing Systems (NIPS)*.

[50] Ming Yin, Yiling Chen, and Yu-An Sun. 2013. The Effects of Performance-Contingent Financial Incentives in Online Labor Markets. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI)*.

[51] Ming Yin, Mary L Gray, Siddharth Suri, and Jennifer Wortman Vaughan. 2016. The communication network within the crowd. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1293–1303.

[52] Lixiu Yu, Paul André, Aniket Kittur, and Robert Kraut. 2014. A comparison of social, learning, and financial strategies on crowd engagement and output quality. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. ACM, 967–978.

[53] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. 2017. Truth inference in crowdsourcing: Is the problem solved? *Proceedings of the VLDB Endowment* 10, 5 (2017), 541–552.

Joint Modeling of Dense and Incomplete Trajectories for Citywide Traffic Volume Inference

Xianfeng Tang
The Pennsylvania State University, USA
tangxianfeng@outlook.com

Boqing Gong
Tencent AI Lab
boqinggo@outlook.com

Yanwei Yu
Yantai University, China
yuyanwei0530@gmail.com

Huaxiu Yao
The Pennsylvania State University, USA
yhx662012@gmail.com

Yandong Li
University of Central Florida, USA
lyndon.leeseu@outlook.com

Haiyong Xie
University of Science and Technology of China, China
haiyong.xie@ieee.org

Xiaoyu Wang
Intellifusion Inc.
fanghuaxue@gmail.com

ABSTRACT

Real-time traffic volume inference is key to an intelligent city. It is a challenging task because accurate traffic volumes on the roads can only be measured at certain locations where sensors are installed. Moreover, the traffic evolves over time due to the influences of weather, events, holidays, etc. Existing solutions to the traffic volume inference problem often rely on dense GPS trajectories, which inevitably fail to account for the vehicles which carry no GPS devices or have them turned off. Consequently, the results are biased to taxicabs because they are almost always online for GPS tracking. In this paper, we propose a novel framework for the citywide traffic volume inference using both dense GPS trajectories and incomplete trajectories captured by camera surveillance systems. Our approach employs a high-fidelity traffic simulator and deep reinforcement learning to recover full vehicle movements from the incomplete trajectories. In order to jointly model the recovered trajectories and dense GPS trajectories, we construct spatiotemporal graphs and use multi-view graph embedding to encode the multi-hop correlations between road segments into real-valued vectors. Finally, we infer the citywide traffic volumes by propagating the traffic values of monitored road segments to the unmonitored ones through masked pairwise similarities. Extensive experiments with two big regions in a provincial capital city in China verify the effectiveness of our approach.

CCS CONCEPTS

• **Information systems** → **Spatial-temporal systems**; **Data mining**; *Information extraction.*

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313621

KEYWORDS

Traffic volume inference; spatiotemporal; smart city transportation

ACM Reference Format:
Xianfeng Tang, Boqing Gong, Yanwei Yu, Huaxiu Yao, Yandong Li, Haiyong Xie, and Xiaoyu Wang. 2019. Joint Modeling of Dense and Incomplete Trajectories for Citywide Traffic Volume Inference. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313621

1 INTRODUCTION

With the vast deployment of sensors, such as loop detectors and surveillance cameras, and the rapid development of data storage techniques, an intelligent city can collect and store a large amount of traffic volume data on the daily basis. It is vital to not only monitor "visible" traffic volumes in real time but also infer those unseen by the sensors. Information extracted from the traffic volume data may contribute to a wide range of urban applications, such as route selection, traffic control, and urban planning, etc.

However, the sensors are often displaced away from each other to a certain distance, and some of them could fail now and then. As a result, the citywide traffic volume data inevitably have lots of missing values, causing challenges to many downstream applications. In this paper, we aim to develop techniques for the citywide traffic volume inference — in other words, to estimate the traffic volume of each and every road segment in every time interval in the city.

The citywide traffic volume inference is a very challenging problem for two major reasons. Firstly, sensors to monitor the traffic are not only sparse but also non-uniform, resulting in relatively low spatial coverage [15]. It is difficult to obtain any historic traffic data for the currently unmonitored roads. Secondly, traffic patterns evolve over time and could suddenly change rapidly due to big events. The high dynamism of traffic makes it difficult to infer the traffic volume data by drawing similarities between different road segments or across various time intervals.

Existing works on inferring the traffic volume mostly fall into the category of missing (spatiotemporal) data inference. A basic and representative framework is to estimate the missing values by linear interpolation [3, 9, 27, 44]. However, those approaches fail to maintain the global consistency of the inferred results and assume simple structures underlying the historic spatiotemporal data. Another frequently used method is collaborative filtering [40]. However, this approach fails for the unmonitored road segments which have no historical information.

Some recent studies use dense trajectories from mobile devices for traffic volume inference. Zhan et al. estimate the citywide traffic volume by combining GPS trajectories with road networks, point-of-interest information, and weather conditions. Meng et al. characterize the similarities between roads using travel speeds extracted from taxi trajectories. However, the dense trajectories, no matter extracted from GPS or taxi, are biased representations of the real traffic. After all, taxi is only a small portion of the total vehicles and not all vehicles carry or turn on the GPS devices.

Partially observed trajectories, which are captured by the static and discrete sensors (e.g., a vehicle tracked by a camera network), are notably under-explored by the existing works. Unlike the dense trajectories (e.g., generated by "active" GPS devices), the partially observed trajectories are incomplete because they are obtained from the "passive" static sensors (e.g., cameras). As a result, they are available for almost all kinds of vehicles no matter they have GPS devices installed or not. Compared with dense trajectories, such incomplete trajectories are collected in a much bigger scale everyday. For instance, the surveillance system recognizes and provides the incomplete trajectories of more than 1.1 million unique vehicles in a provincial capital city in China every single day at the time of this paper written. To the best of our knowledge, the incomplete trajectories have not been utilized before for the traffic volume inference.

Of course, it is more involved to model the spatiotemporal patterns of incomplete trajectories than their dense counterparts. The uncertainties between monitor points have to be taken care of, as well as the noisy measurements at the monitor points. There exist some research tackling the uncertainty problem. For example, Zheng et al. investigate how to reduce the uncertainty in low-sampling-rate trajectories by inferring the possible routes therein. Banerjee et al. employ road-network constraints to summarize all probable routes in a holistic manner. Li et al. use the GPS snippets to help recover the full trajectories. However, these methods require that every road segment has appeared at least once in the past in order to guide the recovering procedure of the full trajectories. Besides, their performance significantly degrades when the traffic pattern changes, e.g., due to big events or severe weather.

To address the aforementioned challenges, we propose a novel framework to **J**ointly **M**odel the **D**ense and **I**ncomplete trajectories (JMDI) for the citywide traffic volume inference. Specifically, we first recover detailed vehicle movements given the incomplete trajectories using a high-fidelity city traffic simulator. We tune the simulator's parameters using deep reinforcement learning so as to automatically identify the proper simulator state which matches the real data. After that, two spatiotemporal graphs are constructed using the recovered trajectories and the dense GPS trajectories, respectively. The graphs are a natural choice for modeling the dynamism of traffic volumes of road segments and over time. Meanwhile, the graphs enable us to conveniently embed the road segments (nodes of the graph) to continuous-value vectors. Finally, we use these embeddings to construct pairwise similarities between the road segments and then propagate the known traffic volumes to the road segments of unknown traffic.

We conduct extensive experiments on large-scale real-world traffic datasets collected from a provincial capital city in China. The dense trajectories are available for GPS-enabled taxi. For the incomplete trajectories, we extract partially observed information from the surveillance cameras as a result of the plate number identification technique [7]. We evaluate the proposed framework JMDI against competitive baselines on two selected regions of the city and during different time periods. We also study some ablations of our approach to highlight the effectiveness of main components of JMDI.

We summarize our main contribution as follows.

- We propose a novel framework, called JMDI, to infer citywide traffic volumes using both dense and incomplete traffic trajectories.
- We propose to use a high-fidelity traffic simulator to recover full vehicle movements from incomplete trajectories. Moreover, in order to efficiently tune the simulator towards the real data, we design a deep reinforcement learning algorithm to automatically set the parameters of the simulator.
- We present a joint embedding method to learn meaningful representations of road segments using spatiotemporal graphs. We further use a semi-supervised propagation method to infer citywide traffic volume values.
- We conduct extensive experiments on large-scale real-world traffic datasets to validate the proposed approach.

To the best of our knowledge, this work is the first to jointly model the dense and incomplete trajectories for the citywide traffic volume inference. Since the dense GPS trajectories are biased to taxi, the incomplete trajectories obtained from other sensors show strong complement to their dense counterparts. Jointly, they give rise to better results than either individually.

2 RELATED WORK

2.1 Traffic Volume Inference

Some studies [9, 27, 44] use linear regression models to infer missing traffic speed or travel time based on taxi trajectories. Aslam et al. [3] learn a regression model with taxi GPS trajectories to estimate traffic volume. *However, regression methods require a great amount of labeled training data, which is unavailable in our problem setting.*

Another category of prior studies apply principal component analysis (PCA) (e.g., [2, 13, 23, 24]) or collaborative filtering (CF) (e.g., [26, 31, 40]) to fill in missing values in spatiotemporal data. PCA-based methods extract traffic patterns from observed data using various PCA techniques, such as Bayesian PCA [24], Probabilistic PCA [13, 23] and FPCA [2]. CF-based methods recover missing values by decomposing spatiotemporal data into the product of low-rank matrices. Yi et al. [40] fill missing values in geo-sensory time series using CF from multiple spatial and temporal perspectives. Ruan et al. [26] use tensor completion to recover missing values in spatiotemporal sensory data. Wang et al. [31] propose a tensor

factorization method to estimate the missing travel time for drivers on road segments. *However, both PCA-based and CF-based methods rely on historical data when filling in. They are unable to handle our problem since the traffic volume of unmonitored road segments are totally missing.*

Graph-based Semi-supervised learning (SSL) method [6] has been widely applied for unlabeled data inference, which can be used for inferring missing values. Label propagation [47], a classic semi-supervised method, infers unobserved labels by propagating existing labels on an affinity graph. Other SSL based methods [1, 36, 46] have been proposed to model the similarities of vertices in the affinity graph. *Although Graph-based SSL methods can be applied to our problem, they only consider the similarities between vertices. Therefore, they fail to utilize rich traffic flow information for volume inference.*

Other existing work aim to infer traffic volume values of road segments using loop detector [12, 17, 35], surveillance cameras [41], or float car trajectories [3, 8, 42]. Studies [12, 35, 41] tackle the volume estimation of a single road segment with loop detectors or surveillance cameras. Thus their methods cannot infer citywide traffic volume. Zhan et al. [42] propose a method to estimate citywide traffic volume using probe taxi trajectories. They estimate travel speeds for volume inference using full taxi trajectories. Recently, Meng et al. [17] propose ST-SSL that predicts city-wide traffic volume values using loop detector incorporating taxi trajectories. They first build a spatiotemporal affinity graph based on travel speed pattern and spatial correlations extracted from loop detector and taxi trajectories. Then they infer traffic volume using semi-supervised learning method on the spatiotemporal affinity graph. *However, both [42] and ST-SSL [17] require full observation of trajectories, thus they are unable to utilize incomplete counterparts.*

2.2 Trajectory Recovery

Several methods (e.g., [4, 34, 45]) have been proposed to predict complete trajectories from partial observations. Zheng et al. [45] investigate how to reduce the uncertainty in low-sampling-rate trajectories. This approach requires every road segment appearing in historical data, which is impossible for incomplete trajectories. Banerjee et al. [4] infer uncertain trajectories from network-constrained partial observations by summarizing all probable routes in a holistic manner. However, their method ignores many important factors such as vehicle interaction and road constraints. *More important, these methods require historical trajectories as input, which cannot be applied to recovering routes in our problem.*

3 NOTATIONS AND PROBLEM STATEMENT

Let $\{r_1, r_2, \ldots, r_m\}$ denote road segments of a road network in a city region. Each road segment is a directed short portion of a road. The traffic volume of a road segment can be represented using the total number of vehicles traversing through it during a fixed time interval. We split the whole time period (e.g, one week) into equal-length continuous time intervals and use vector $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ to denote the traffic volume values of the road segment r_i during the past n continuous time intervals. Note that the traffic volume values are only available at monitored road segments.

Different sources of trajectories (i.e., incomplete observations from cameras or GPS trackers of taxi) are denotes by $\Delta^{\mathrm{S}} = \{\delta_i\}$, where the superscript S stands for the source S. A trajectory $\delta_i = p_{i,1} \rightarrow p_{i,2} \rightarrow \cdots \rightarrow p_{i,s}$ is a list of s points in chronological order, where each point $p = \langle l, t \rangle$ is a location coordinate l (i.e., latitude and longitude) and a time stamp t. We denote by Δ^{D} and Δ^{I} dense and incomplete trajectories, respectively. Note that points contained in the incomplete trajectories Δ^{I} are restricted to monitor points — therefore they form a fixed finite set.

The citywide traffic volume inference problem is defined as follows: Given the road network, observed traffic volume values $\{x_{it}\}$ at monitored road segments, and multiple sources of trajectories Δ^{S} ($\mathrm{S} \in \{\mathrm{D}, \mathrm{I}\}$), our goal is to infer the traffic volume values of any unmonitored road segment at any time interval.

4 APPROACH

Figure 1 illustrates the proposed framework JMDI. First, we design an algorithm to recover full trajectory from incomplete observations using a traffic simulator, whose parameters are tuned by deep reinforcement learning. Next, we construct spatiotemporal graphs from the dense and incomplete trajectories. In order to model the spatiotemporal dynamics of traffic volume, we then employ a multi-view graph embedding technique [25] to learn representation vectors for road segments. Finally, we present a propagation method which infers the citywide traffic volume values according to pairwise similarities between different road segments. In the following sections, we introduce each major component of our solution in detail.

4.1 Trajectory Recovery

In this section, we introduce the recovery process for incomplete trajectories. Instead of data-driven algorithms, we incorporate a high-fidelity traffic simulator [11] for the recovery. Different from previous trajectory recovery methods (e.g., [4, 45]), the simulator jointly models the influence of traffic rules as well as vehicle interactions. Furthermore, we propose a novel deep reinforcement learning [20] algorithm that improves the accuracy of trajectory recovery by tuning time-aware hyper-parameters of the simulator.

4.1.1 Trajectory Recovery via Simulator. We first estimate the missing portions of incomplete trajectories by using rich spatiotemporal movement patterns. Unlike the existing methods [4, 14, 37, 45], our solution relies on a traffic simulator [11], which naturally observes road network constraints (e.g., no turn on red at some intersections) and the global consistency of traffic in the road network. It also captures the influence of vehicle interactions.

Specifically, we simulate all vehicles simultaneously in real-world road networks using the traffic simulator named *SUMO* [11], conditioning on observations in the incomplete trajectories. Given an incomplete trajectory $\delta_i = p_{i,1} \rightarrow p_{i,2} \rightarrow \cdots \rightarrow p_{i,s}$, the simulator initializes a vehicle at the corresponding time and location of each point $p_{i,j}$. Then, the vehicle moves toward $p_{i,j+1}$ along the most possible route selected by the simulator. Multiple factors, such as distance, speed, traffic jams, etc., are considered jointly to find the best route. The simulated results for all points $p_{i,j} \rightarrow p_{i,j+1}$ are collected to fill in the missing values in the incomplete trajectories.

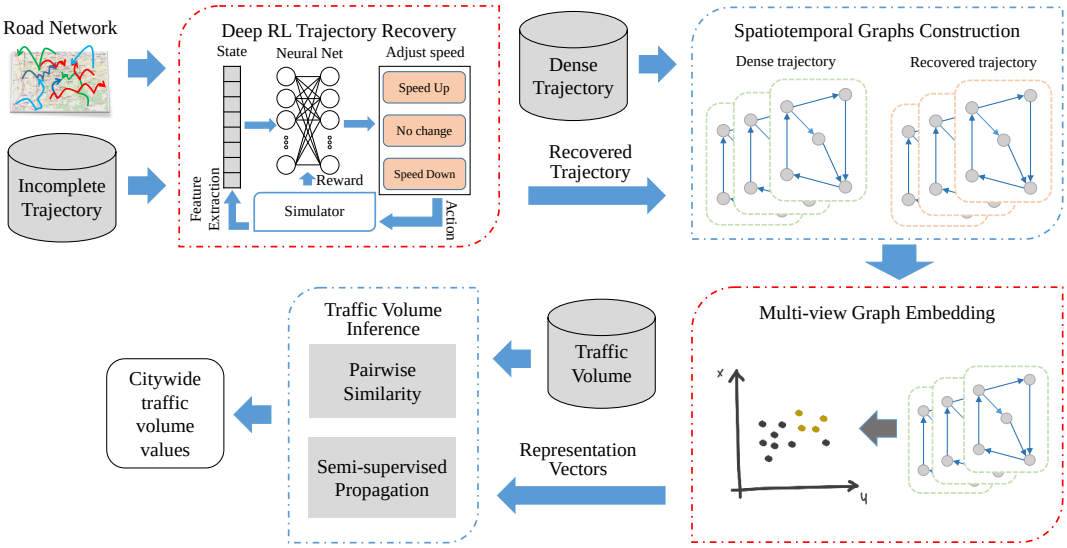


Figure 1: Overview of the proposed JMDI framework.

However, blindly using the simulator for trajectory recovery without careful validation is not rigorous. Therefore, a novel metric is proposed for trajectory recovery evaluation. Taking incomplete trajectory δ_i as an example, we first measure the relative arrival time $t_{i,j}$ for $p_{i,j} \in \delta_i$, using time stamps provided by δ_i. Here $t_{i,j}$ equals to the time consumption of traversing between $p_{i,j}$ and $p_{i,1}$ ($t_{i,1} = 0$) captured by real-world sensors. Since vehicles would traverse through each monitor point successively in the simulating environment, we compute the relative arrival time at each $p_{i,j}$ again, using time stamps from simulating results. Similarly, let $t^{\text{simu}}_{i,j}$ denote the relative arrival time at $p_{i,j}$ of the simulated vehicle. Given those two kinds of relative arrival time $t_{i,j}$ and $t^{\text{simu}}_{i,j}$, the trajectory recovery accuracy can be estimated using the following equation:

$$err = \sum_{\delta_i \in \Delta^{\text{I}}} \sum_{p_{i,j} \in \delta_i} |t^{\text{simu}}_{i,j} - t_{i,j}| / (\sum_{\delta_i \in \Delta^{\text{I}}} \sum_{p_{i,j} \in \delta_i} 1). \quad (1)$$

The intuition behind Equation 1 is that larger errors will be generated if the recovered trajectories differ a lot from the real ones (i.e., larger $|t^{\text{simu}}_{i,j} - t_{i,j}|$). And the recovered trajectories have low err if the simulated vehicles arrive at monitor points sharply.

4.1.2 *Deep Reinforcement Learning for Parameter Tuning.*

There are a few major parameters of the simulator (see Table 1) we have to tune in order to achieve good simulation results (in other words, small err). For example, changing the speed of vehicles could directly affect the arrival time at each monitored point in the simulating environment, and modifying the lane-changing rules could lead to traffic jams. In order to simulate as realistic as possible, parameter tuning is key. Unfortunately, it is nontrivial to tune the parameters because they are inter-correlated and their effects on the traffic volume are delayed. For example, we may speed up a vehicle anytime but the results of the acceleration cannot be validated until the vehicle arrives at the next monitor point. Besides, the change of speed of one vehicle may change the speed of others.

Table 1: Examples of parameters in *SUMO*.

Parameters	Descriptions
stop	Lets the vehicle stop at the given edge
change lane	Change to the lane with the given index
slow down	Change the speed smoothly
speed	Sets the vehicle speed to the given value
change target	Change the vehicle's destination edge
change route	Assigns the vehicle's new route
speed factor	Sets the vehicle's speed factor
max speed	Sets the vehicle's maximum speed

To address the above challenges, we propose a reinforcement learning (RL) algorithm to tune the simulator's parameters such that the simulated trajectories can better match the sparsely monitored real data. The RL algorithm successfully avoids the manual configuration of the simulator in a tedious trial-and-error manner.

The general goal of reinforcement learning is to maximize the expected cumulative rewards in a sequential decision making process. Given a set of states $\{s_i\}$ and actions $\{a_i\}$, an action a is taken by the agent following a certain policy π after seeing the current state s in each step. The expected return of each potential action a under state s is evaluated using the Bellmen equation [32],

$$Q_\pi(s, a) = \mathbf{E}(\sum_{i=1}^{\infty} \gamma^{i-1} R_i | S_0 = s, A_0 = a, \pi), \quad (2)$$

where γ is the discount factor for future rewards. The goal of reinforcement learning is to maximize the expectation of accumulative future rewards through the sequential process.

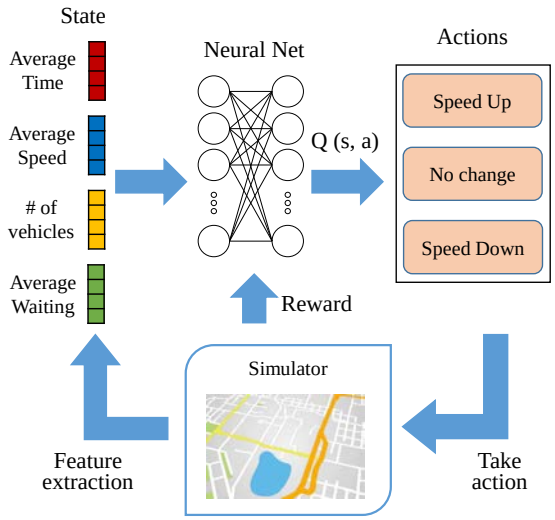


Figure 2: Deep reinforcement learning architecture for tuning the vehicles' speeds in the simulator.

Parameter tuning of the simulator can be naturally considered as a sequential decision making process, where states are the snapshots of the simulating environment and actions are all possible ways of parameter tuning. We learn a policy that continuously takes positive action to change the parameters of the simulator in each step. Note that it is intractable to use the Bellmen Equation (Equation 2) to directly optimize the Q-value as it has to track all possible state-action combinations. Instead, we employ Deep Q-Learning (DQN) [20] which approximates the Q-value using a deep neural network. Given a state s, the output of DQN is the approximated Q-values for the actions $Q(s, a; \theta)$, where θ are neural network weights in DQN. The greedy action is chosen as $\arg\max_{a \in A} Q(s, a; \theta)$. In our context, we use this action to tune the simulator's parameters.

Figure 2 illustrates the DQN framework for trajectory recovery. Real-valued feature vectors extracted from the simulator are used to describe the states. We construct a two-layer fully-connected neural network for DQN to approximate the Q-value. During the training process, weights of the neural network are optimized iteratively to reduce the discrepancy between the optimal Q-value and the predicted one, using gradient descent of the loss function. The details of states, actions, rewards, and the training are described below:

- **States**: Many existing deep reinforcement learning methods (e.g. [16, 30, 33, 43]) use real-valued vector to represent state. Following the same practice, we first extract representative features using the API of the simulator. Since the number of vehicles on the road segments directly describes the traffic volume distribution, we first retrieve their values from the simulator. Next, average traversing time and speed are acquired as the traversing speed and time are closely related to the traffic volume. In addition, we retrieve average waiting time, which measures the average stopping time of every vehicle in the immediate past simulating step. Those four types of values are retrieved from each road segment and then concatenated into a real-valued vector as the state representation. Given m road segments, the dimension of the state feature vector is $4m$.
- **Actions**: Recall that the action with the largest expected cumulative reward (Q-value) will be selected by the agent in each simulating step. To construct the action set, we change the speeds of vehicles which are one of the most important factors that affect the traversing trajectory. However, directly tuning the speed for every vehicle in the simulator is prohibitive because there could be thousands of vehicles traveling simultaneously. To simplify the setting, we do not change the speed continuously for every vehicle. Instead, the speed limits (meters/second) for different vehicles are modified within a certain range in each simulation step. The actions include increasing the speed limit by +1, 0, or −1 (i.e., decrease by 1) for each vehicle. To reduce the action space, we group the vehicles according to their sizes (i.e., sedans, SUVs, and large trucks) and assign the same speed limit to a group of vehicles. Hence, the action space consists of 9 possible actions pairing the speed limits with the groups. Such a setting actually changes the behavior of every group of vehicles by limiting their speeds into a certain range.
- **Rewards**: In reinforcement learning, positive or negative rewards are provided by the environment for each action by the agent. The agent can utilize the reward to calculate the optimal Q-value, and to optimize parameters in the neural network using the discrepancy induced by the Bellman equation. We construct short-term rewards according to the following equation:

$$R = \sum_{v \in \Omega} (e^{-|t_v^{\text{simu}} - t_v|}), \tag{3}$$

  where Ω denotes vehicles that have arrived at the monitor points at the simulating step. That is, the simulating results of those vehicles can be evaluated at this time. The arrival time stamps t_v^{simu} and t_v are of the simulated ones and the real vehicles, respectively. The definition is straightforward. Obviously, lower absolute time differences are received if and only if more vehicles arrive sharply, and the agent receives larger rewards under such conditions.
- **Training**: A simulating step is set to lasting for one minute. A two-layer neural network is initialized and trained to estimate the Q-value. Weights θ of the network are updated every step by gradient descent, using the discrepancy between estimated Q-value and the feedback from the simulating environment. Before the training, we first set up a replay memory $\mathcal{M}$ to store state transitions (i.e., (s, a, s', R) where s' denotes the next state following s after taking action a). The deep reinforcement learning algorithm keeps sampling mini-batches from the replay memory and updates its network weights in each simulating step by minimizing the following loss function:

$$\mathcal{L}(\theta) = \mathbf{E}_{s,a}[(Q^* - Q(s, a; \theta))^2], \tag{4}$$

Algorithm 1: Training of the Deep RL Architecture

Randomly initialize network parameters θ and the replay memory $\mathcal{M}$;
for *each episode* **do**
  for *each simulation step* **do**
    /* Action and simulating */
    if *random()* $< \epsilon$ **then**
      Select a random action a;
    else
      Select $a = \max_a Q(s, a; \theta)$;
    end
    Obtain short-term reward R and new state s' by executing a in the simulator;
    Store state transition (s, a, s', R) in $\mathcal{M}$;
    /* Updating Q-network parameters θ */
    Sample mini-batch state transitions $\{(s, a, s', R)\}$ from $\mathcal{M}$;
    if *simulator terminated* **then**
      $Q^* = R$;
    else
      $Q^* = R + \gamma \max_{a'} Q(s', a'; \theta)$;
    end
    Use Equation 5 to perform gradient descent on the loss $\mathcal{L}(\theta)$ based on Q^*;
  end
end

where $Q^* = R + \gamma \max_{a'} Q(s', a'; \theta)$ is the target optimal Q-value, which is the sum of the short-term reward R and the optimal Q-value of the next step. The decay term γ gives penalty to future rewards. The expectation is computed over the mini-batches. Furthermore, we adopt ϵ-greedy strategy [21] to balance the trade-off between exploration and exploitation. That is, with probability ϵ, a random action is selected for exploration during training, instead of the optimal one. After calculating the loss function, the network weights θ are updated using gradient descent. The gradient of θ is given as follows,

$$\nabla_\theta \mathcal{L}(\theta) = \mathbb{E}_{s,a}[(Q^* - Q(s, a; \theta))\nabla_\theta Q(s, a; \theta)]. \quad (5)$$

The overall training procedure of the deep reinforcement learning architecture is given in Algorithm 1. Once the simulator is stable, we fill in the missing values of the real and incomplete trajectories using the traffic volume values in the simulator *SUMO*. Compared with taxi trajectories collected from GPS devices, these recovered trajectories, denoted by Δ^{I+}, are more representative of a variety of vehicle types. Both kinds of trajectories will be employed in the following steps.

4.2 Spatiotemporal Graph Construction

Temporal correlations of different road segments change over time. For example, an expressway connecting business areas and residential areas may exhibit reversed traffic flows from morning to evening. Moreover, the spatial relationship between road segments is extremely complicated (e.g., overpasses at the intersection of the main road). To model such dynamism of traffic in the road network, we construct spatiotemporal graphs, as shown in Figure 3 and detailed below.

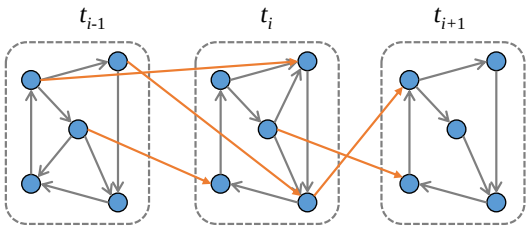


Figure 3: An example of spatiotemporal graph. Three layers are shown in the figure. Gray arrows denote edges within the same layer and yellow arrows denote edges between current and next layers.

A spatiotemporal graph in this paper is a multi-layer graph $G = \langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V} = \{v_i^t\}$ denotes time-enhanced nodes representing road segments $\{r_i\}$, and $\mathcal{E}$ represents edges between nodes. Nodes in the same time interval are grouped to the same *layer*. To preserve spatial restrictions of the road network, edges in $\mathcal{E}$ are limited to $v_i^t \rightarrow v_j^{\{t+1\}}$ where vehicles could travel from the road segment r_i to r_j. The edge's weight is assigned using the traffic flow value, i.e., number of trajectories traversing through the edge.

To differentiate mobility behaviors and traffic patterns in different sources of trajectories, two spatiotemporal graphs are constructed respectively using the dense trajectories Δ^D and recovered Δ^{I+} from incomplete trajectories.

4.3 Multi-view Graph Embedding

The objective of citywide traffic volume inference is to estimate the unknown traffic volume values from the observed ones. This problem is challenging due to multiple interwoven factors such as the road network, vehicles' physical condition, road condition, time, etc. Our solution relies on the spatiotemporal graphs constructed from the macro trajectories, effectively factoring out those individual factors and allowing us to conduct the inference on the graphs instead.

We first embed the time-enhanced road segments (nodes of the graph) to real-valued vectors using the two spatiotemporal graphs obtained above. The embedding captures multi-hop correlations [28] between road segments at different times. Moreover, joint embeddings from the two graphs naturally account for both the GPS-tracked taxi and other vehicles monitored in the incomplete trajectories. We first introduce the embedding process on a single graph, followed by the joint embedding algorithm from the two graphs by a multi-view graph embedding method [25].

We adopt the Skip-gram model [18] to learn the embedding vector $\mathbf{u}_i^t$ for each node v_i^t in a spatiotemporal graph $G = \langle \mathcal{V}, \mathcal{E} \rangle$. Given a node in the graph, the skip-gram model aims to predict the probability of the node's context. We first define a path in G as $P_i = v_{i_1} v_{i_2} \dots v_{i_\sigma}$, where v_{i_j} in P_i is corresponding to a

node in G. We omit the time superscript here, which can be easily acquired from each layer. Given a specific node v_i^t, the set of all paths containing v_i^t formulates a set $\mathcal{P}(v_i^t)$. We define set $\mathcal{N}(v_i^t) = \{v_c | \exists P_i \in \mathcal{P}(v_i^t), v_c \in P_i\} \backslash \{v_i^t\}$ as the context of node v_i^t, which refers to all multi-hop neighbors of v_i^t (i.e., nodes closely surround v_i^t in space and time) [29]. Specifically, the following log-probability is maximized:

$$O = \sum_{v_i^t \in G} \log Pr(\mathcal{N}(v_i^t) | v_i^t)) = \sum_{v_i^t \in G} \log \prod_{v_c \in \mathcal{N}(v_i^t)} P(v_c | v_i^t)$$
$$= \sum_{v_i^t \in G} \sum_{v_c \in \mathcal{N}(v_i^t)} \log P(v_c | v_i^t).$$

and $P(v_c|v_i^t)$ is further estimated by $P(v_c|v_i^t) = \sigma(\mathbf{u}_i^{t\mathrm{T}} \mathbf{u}_c')$ where $\sigma(\cdot)$ is the sigmoid function, and $\mathbf{u}_c'$ denotes the auxiliary vector of v_c when it is treated as "context" of other nodes.

To achieve better performance and reduce computational costs, negative sampling technique [19] is applied. For each node v_i^t and a neighbor node v_c, we sample negative nodes $NG(v_i^t) = \{z | z \notin \mathcal{N}(v_i^t)\}$ which denotes unrelated road segments. The objective function for network embedding with negative sampling is:

$$O = \sum_{v_i^t \in G} \sum_{v_c \in \mathcal{N}(v_i^t)} \left\{ \log P(v_c|v_i^t) + \sum_{z \in NG(v_i^t)} \log(1 - P(v_c|v_i^t)) \right\}$$
$$= \sum_{v_i^t \in G} \sum_{v_c \in \mathcal{N}(v_i^t)} \left\{ \log(\sigma(\mathbf{u}_i^{t\mathrm{T}} \mathbf{u}_c')) + \sum_{z \in NG(v_i^t)} \log(1 - \sigma(\mathbf{u}_i^{t\mathrm{T}} \mathbf{u}_z')) \right\}.$$

The objective reads that, for each node v_i^t, we aim to maximize the probability of its co-occurrence with nodes in $\mathcal{N}(v_i^t)$, while minimize the probability of it being neighbor with nodes from $NG(v_i^t)$.

Since we have two spatiotemporal graphs of the same set of road segments, we may learn the embedding by a trade-off between the objective functions above over the two graphs. Namely,

$$O = \alpha O_{G^{\mathrm{D}}} + (1 - \alpha) O_{G^{\mathrm{I+}}}, \tag{6}$$

where G^{D} and $G^{\mathrm{I+}}$ are the graphs constructed from dense and recovered trajectories, respectively, and α balances the two terms.

4.4 Citywide Traffic Volume Inference

After learning representations for road segments that preserve spatiotemporal dynamics, we borrow a semi-supervised learning formulation [47] to propagate the traffic volume values to the unknown road segments. At first glance, one may train a regression model taking as input the road segment's embedding to predict the missing traffic volume data. However, such methods fail when only a small portion of the entire road network is monitored. We instead employ a formulation which is popular in semi-supervised learning for the inference.

We first define pairwise similarities between any two road segments. The similarity of r_i at time interval t and r_j at time interval t' is measured by:

$$sim(r_i^t, r_j^{t'}) := \omega_{i,j}^{t,t'} = \langle \mathbf{u}_i^t \cdot \mathbf{u}_j^{t'} \rangle, \tag{7}$$

where $\langle \cdot \rangle$ denotes the inner product of two vectors.

We then prune out irrelevant pairs of road segments by constructing 0-1 masks based on geospatial features (i.e., absolute distance between road segments) as well as temporal distance. The intuition behind such pruning is that traffic volume values of faraway locations usually do not have similarities, and the temporal dependencies of traffic pattern at the same location will decrease as the time lag becomes longer [39]. The "masked" pairwise similarities between road segments are defined as:

$$\omega^{\star\, t,t'}_{i,j} = \begin{cases} \omega_{i,j}^{t,t'}, & \text{if } adj(r_i, r_j) \wedge |t - t'| \leq 1, \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

Here $adj(r_i, r_j)$ is *true* if and only if r_i and r_j are adjacent on the road network. The above mask ensures only the correlations between road segments that are close enough in time and space are considered. The citywide traffic volume values $\{x_{it}\}$ are inferred by minimize the following with respect to (w.r.t.) the unknown values:

$$\mathcal{L}^\star = \sum_{1 \leq i,j \leq m, 1 \leq t,t' \leq n} \omega^{\star\, t,t'}_{i,j} (x_{it} - x_{jt'})^2. \tag{9}$$

Since the objective function is convex w.r.t. the unknown traffic volume values, gradient descent is applied to find the global solutions.

5 EXPERIMENTS

In this section, we conduct extensive experiments to answer the following research questions.

- How does the proposed JMDI perform compared with the baseline methods?
- How do each component of JMDI affect the overall performance?
- How accurate is the trajectory recovery via the deep RL architecture?
- How to balance different types of trajectories through the hyper-parameter α?
- How is the inference error distributed over space (i.e., different road segments) and time (i.e., different time periods of a day)?

5.1 Datasets

We compile a traffic volume dataset using the vehicle counting reported by the city surveillance system from a large provincial capital city in China. The incomplete trajectories are acquired using plate numbers identified by surveillance cameras (with an average F1-score of 90%). Dense trajectories are collected from GPS sensors equipped by public taxicabs. The average reporting frequency of the taxicab-carried GPS devices is 5 seconds. All those datasets are collected over the period of Aug. 2016. The road network is extracted from *Openstreetmap.org* [22]. We split roads into segments using intersections as natural cut-points and set the length of the time interval to 5 minutes. Moreover, geospatial features such as starting/ending locations, road segment length, road width, number of lanes, road type, speed limit, etc., are constructed for each road segment. We select two regions in the city. The detailed data descriptions are shown in Table 2.

5.2 Experiment Settings

We cut trajectories recursively at any $p_{i,j} \rightarrow p_{i,j+1}$ if the time interval between $p_{i,j}$ and $p_{i,j+1}$ is larger than 30 minutes. Because

Table 2: Dataset statistics.

	Region 1	Region 2
Time period	Aug. 1 - Aug 4	Aug. 14 - Aug. 20
Time interval	5 minutes	5 minute
Longitude range	116.8988 - 116.9457	117.0635 - 117.1053
Latitude range	36.6507 - 36.6926	36.6624 - 36.7052
# of all road seg.	322	143
# of monitored seg.	45	32
# of incomplete traj.	366,645	1,798,782
# of dense taxi traj.	12,562	97,315

vehicles may stop and park during such long duration, their behaviors are different from the constantly moving vehicles.

We randomly split the road segments with traffic information into training (80%) and testing (20%), respectively. We further select 20% of the training randomly as validation. The same split is used for all experiments. We run each baseline and ablation 10 times to report the average results.

We set the memory buffer size to $10,000$ and the discount factor γ to 0.8 for the deep RL framework. A feed-forward neural network with two 256-dimension hidden layers is trained using the Adam optimizer [10]. The size of a mini-batch is 128 and the learning rate is 0.0001. Parameters of the network are updated at every step of the simulator. The probability ϵ in the deep RL training is reduced linearly from 0.5 to 0.01 over $2,000$ epochs. The speed limits for every type of vehicle are restricted between $1m/s$ and $40m/s$ in the simulating environment. We set the dimension size for embedding vectors to 50. The window size for sampling in Skip-gram is 10.

5.3 Metrics

We use Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) to evaluate the performance of all methods.The detailed definitions of the two metrics are stated as below:

$$RMSE = \sqrt{\frac{1}{s}\sum_{l=1}^{s}(x_l - \hat{x}_l)^2}, \quad MAPE = \frac{1}{s}\sum_{l=1}^{s}\frac{|x_l - \hat{x}_l|}{\hat{x}_l},$$

where $x_l \in \{x_{it}\}$ is a test sample, $\hat{x}_l$ is the ground truth of x_l, and s denotes the total number of test samples.

While RMSE receives more penalties from larger values, MAPE focuses on the error of smaller values. Therefore, combining these two metrics can lead to more comprehensive conclusions. Similar to [38, 39], to evaluate relative error using MAPE, we filter out samples whose traffic volume is lower than 5 in testing, since road segments with very low traffic volume are of little interest for our problem.

5.4 Baselines

Our solution is compared with the following baselines: (1) two geospatial-based methods, (2) two supervised learning methods, and (3) two semi-supervised learning methods.

- **Spatial *k*NN** selects top k nearest road segments with traffic data and uses the average of their volume values at each time interval as the traffic volume prediction for unmonitored road segments.
- **Contextual Average** (CA) uses the averaged volume values from same-type road segments (e.g., primary, expressway, etc.) as the prediction.
- **Linear Regression** (LR) is trained on all road segments with traffic volume using geospatial features. We train regression models for each time interval.
- **XGBoost** [5] is a boosting-tree-based method which is popular in data mining community. We train XGBoost separately on each time interval.
- **Graph SSL** Classic graph-based semi-supervised learning method with loss function $\mathcal{L} = \sum_t \sum_{i,j} a_{i,j}(x_{it} - x_{jt})^2$, which is implemented following Zhu et al.. The similarity weight $a_{i,j}$ is defined using the Euclidean distance between the geospatial feature vectors r_i and r_j.
- **ST-SSL** [17] is the state-of-the-art method for inferring citywide traffic volume using dense trajectories. ST-SSL utilizes static features and estimated speed patterns from taxi trajectories to model the correlations between road segments. The citywide traffic volume is then inferred by a semi-supervised learning method.

5.5 Result Analysis of Different Methods

The comparison of JMDI with other methods in two regions is shown in Table 3. As we can see, JMDI achieves best performances in both regions. In particular, JMDI significantly outperforms geospatial-based methods due to its capability of well modeling the dynamism of traffic volume. In addition, JMDI achieves tremendous improvement compared with supervised methods. This is because supervised methods are trained on road segments with traffic data, they ignore correlations between other road segments. Moreover, both geospatial and regression-based methods fail to model spatiotemporal patterns in multi-sources trajectories, and do not well utilize information from unmonitored road segments.

JMDI significantly outperforms both Graph SSL and ST-SSL. There are two potential reasons. First, JMDI utilizes vast incomplete trajectory data to recover spatiotemporal similarities among road segments. The proposed deep reinforcement learning recovery method successfully transforms partial observations to complete route samples. Second, the joint embedding algorithm enhances the representation of road segments. Both multi-hop correlations and multi-view graph information are preserved in learned embedding vector, resulting in better performance. Furthermore, Graph SSL achieves lower errors compared with ST-SSL, a possible explanation is that the strong assumption of speed similarity does not hold in those regions. For example, most drivers will drive close to the speed limit unless the road capacity reaches the upper limit. In such condition, speeds are almost stationary while traffic volume could change depending on the number of vehicles. Moreover, it suffers from the spatial sparsity of taxi trajectories since the coverage of taxi trajectory in the city is limited.

5.6 Ablation Study

To validate each component of JMDI, the following variations of JMDI are further studied. Despite the changed part(s), all variations

Table 3: Comparison of baseline methods.

Methods	Region 1		Region 2	
	RMSE	MAPE	RMSE	MAPE
Spatial kNN	7.1404	0.6348	8.5259	0.6053
CA	8.0866	0.5251	8.1317	0.5423
LR	13.2857	1.0911	11.5781	1.0219
XGBoost	9.2157	0.6711	9.2847	0.6954
Graph SSL	6.1743	0.4457	6.3640	0.4643
ST-SSL	6.5603	0.5631	7.1528	0.5975
JMDI (Ours)	**5.5877**	**0.3675**	**5.6680**	**0.3915**

have the same framework structure and parameter settings. The results of all variations are listed in Table 4.

- **JMDI-Reg** trains XGBoost models using representation of road segments. It can be considered as appending embedding vectors to the geospatial features in previous XGBoost.
- **JMDI-Tr** replaces the trajectory recovery part using algorithm in [4]. The method infers uncertain trajectories from network-constrained partial observations by summarizing all probable routes in a holistic manner.
- **JMDI-Df** recovers trajectories using default simulator, i.e., we directly collect results from *SUMO* without training the deep reinforcement learning component.
- **JMDI-Semi** directly applies graph-based semi-supervised learning [47] on the spatiotemporal graphs to propagate known traffic volume without the embedding process. The similarity between any time-enhanced road segments (nodes of the graph) is assigned according to the edge weights (i.e., traffic flow value from trajectory data).
- **JMDI-Um** proceeds semi-supervised learning without pruning out irrelevant pairs of road segments. Namely, the raw similarity in Equation (7) is used.

JMDI-Reg performs the worst among all variations. However, it still outperforms the original XGBoost. In addition, the performance gap between JMDI-Reg and JMDI reflects the advantage of the semi-supervised method, which successfully leverages large amounts of unmonitored road segments.

The comparison between JMDI-Tr and JMDI highlights the effectiveness of the proposed trajectory recovery method. Sampling-based methods are unable to handle complex road network constraints, vehicle interactions, and other factors, thus fail to recovery highly accurate trajectories. Moreover, JMDI-Df performs worse compared with JMDI. A possible explanation is that default stationery (time independent) parameters do not describe real-world vehicle movements correctly. Furthermore, since there are fewer vehicles in the simulator, some parameters should still be altered even if the ground truth is known. For example, vehicles traverse faster and arrive earlier in a simulated environment, resulting in a higher error of trajectory recovery. In such condition, speed limits could be explicitly reduced to adjust vehicles' movement.

The performance of JMDI-Semi is obviously overtaken by JMDI. This is because the joint embedding enhances the representation capacity from multi-hop spatiotemporal correlations and multi-view graph information.

Table 4: Comparison of variations.

Methods	Region 1		Region 2	
	RMSE	MAPE	RMSE	MAPE
JMDI-Reg	7.0851	0.5182	6.9187	0.5155
JMDI-Tr	6.5091	0.5127	6.8262	0.5390
JMDI-Df	6.1445	0.4104	6.4006	0.4378
JMDI-Semi	7.9121	0.5673	7.5165	0.5366
JMDI-Um	6.0357	0.3781	6.2272	0.4057
JMDI	**5.5877**	**0.3675**	**5.6680**	**0.3915**

Table 5: Averaged arrival time error (s) of simulated vehicles.

Region	Region 1	Region 2
Default Parameters	27.2356	29.2141
Deep RL	**22.2554**	**25.2231**

The comparison of JMDI-Um and JMDI indicates the effectiveness of proposed 0-1 masks that explicitly remove unrelated road segment pairs. The propagation of traffic volume focuses on relevant pairs of road segments, improving the overall performance of JMDI distinctly.

5.7 Accuracy of Trajectory Recovery

We evaluate the trajectory recovery algorithm. Since the spatial ground truth for incomplete trajectories is unavailable, we construct validation from arrival times at monitor points. First, we manually tune speed limit parameters using greedy search approach over each hour of the simulating procedure to minimize the arrival time error. Those speed parameters are used as the default setting of the compared method. Note that we do not use the default speed limit parameters in *SUMO* (i.e., $22.4m/s$). Following Equation (1), the averaged arrival time error (in second) of the default simulator and that of the optimized simulator are compared. The results are demonstrated in Table 5. The proposed deep reinforcement learning method reduces averaged arrival error significantly, which is relatively lower by 18.29% in Region 1 and 13.61% in Region 2. Moreover, the improvement in Region 1 is relatively higher than in Region 2. A possible reason is that Region 1 has a relatively simpler environment with less traffic. Therefore, modeling the state-action relationship could be easier for the DQN.

5.8 Hyper-Parameter

We further study the hyperparameter α which balances weights between different spatiotemporal graphs. More emphasis is put to recovered trajectories when α is closer to 0, and vise versa when α is closer to 1. The performance of JMDI w.r.t different α is shown in Figure 4. As we can observe, the inference error first drops and then rises in both regions. Moreover, JMDI achieves relatively low inference errors when α falls into a certain range. However, the performance is worse when α is closer to 0 or 1. In addition, the performance of JMDI is significantly improved in both regions by leveraging incomplete trajectories. Compared with the ablation that only uses taxi data (i.e., $\alpha = 1$), JMDI reduces the RMSE value by 8.84%, 5.47%, and the MAPE value by 9.76%, 7.41% in region 1 and

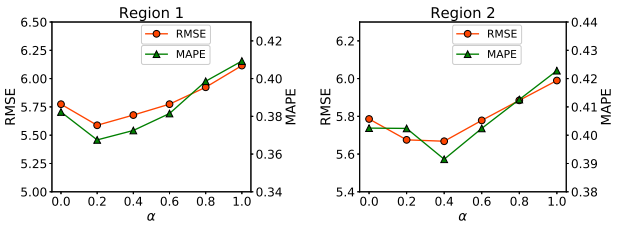


Figure 4: Performance w.r.t different α.

Table 6: Error distribution over different groups of road segments.

Group	Region 1		Region 2	
	RMSE	MAPE	RMSE	MAPE
Group 1	6.5151	0.4102	6.6741	0.4170
Group 2	3.5488	0.3368	3.4101	0.3274

region 2, respectively. This suggests the importance of modeling multi-source trajectories jointly. Besides, the relative improvement from recovered trajectories in Region 1 is larger compared with Region 2. A potential explanation is that the taxi trajectories in Region 1 are very sparse, thus brings biased spatiotemporal correlation and results in larger inference error. Moreover, recovered trajectories also contain noises and uncertainties, and that's why JMDI achieves the best performance by combining them together.

5.9 Error Distribution Interpretation

We also explore the spatiotemporal performance of JMDI, i.e., error distribution over different types of roads, and over different time periods of a day. Without loss of generality, Region 2 is selected for the interpretation.

5.9.1 Error distribution over the space. To study the performance of JMDI over different road segments, we first split road segments from testing set into two categories using road type information extracted from the map. Namely, Group 1 (i.e., major roads such as highway and primary way) and Group 2 (secondary roads like tertiary route). Generally, road segments from Group 1 have more lanes and higher average traffic volume values.

Table 6 provides a detailed illustration of the spatial performance of JMDI. As we can see, the RMSE values are higher for road segments from Group 1, as they have higher traffic volume values. RMSE values are positively correlated to the original variables themselves. Meanwhile, we also observe that the MAPE values of road segments from Group 2 is relatively lower than those of road segments from Group 1. One potential reason is that traffic volume of road segments from Group 1 has higher uncertainty. For example, traffic jams occur when the traffic volume exceeds road capacity, and this usually happens on major roads. Moreover, local events have a higher probability to happen near major roads. Those factors increase the uncertainty of traffic volume on road segments from Group 1, resulting in a high relative error (i.e., MAPE) of the framework.

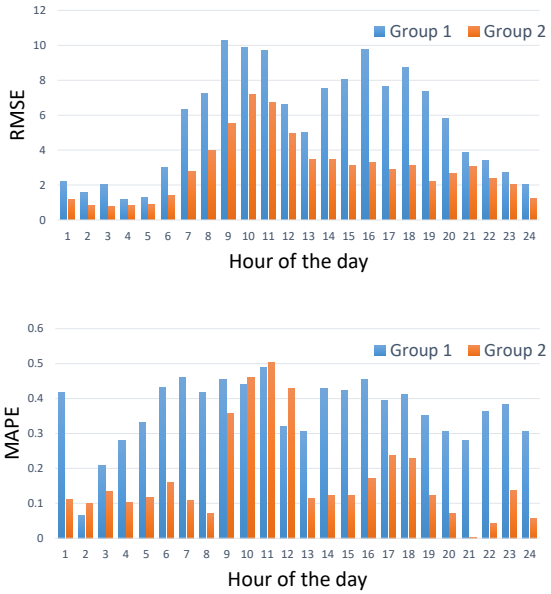


Figure 5: Error distribution over different hours of a day. Number i on x-axis indicates the i-th hour of a day. For example, 7 means the 7th hour, i.e., 6:00 to 7:00.

5.9.2 Error distribution over different time periods of the day. Since traffic volume value series are closely related to different times of a day, and peak hours are more interesting, we further explore the performance of JMDI at different hours. The averaged RMSE and MAPE values are calculated over each hour of a day in the testing set. Detailed values are illustrated in Figure 5. To better interpret those errors, we still use the same group category in Section 5.9.1.

As it reads from the figure, the RMSE values for both groups of road segments have clear patterns. The values are higher during the busy time, such as rush hours in the morning and in the afternoon. This is reasonable since common practice tells us that peak hours would have more traffic. And RMSE values are positively correlated to the ground truth values.

On the other hand, the relative error of JMDI (i.e., the MAPE values) are pretty stable for Group 1 road segments, where the values of MAPE in rush hours are slightly higher than other ones. This is again due to traffic patterns during rush hours are more uncertain, and more accidental events could change the traffic volume dramatically. Those are reasons why inference difficulty is increased, leading to higher percentage error. At the same time, JMDI achieve very low MAPE values for Group 2 road segments at around 0.1 during off-peak hours. And its MAPE values during rush hours in the morning is reasonable. This is explainable since accidental events or traffic jams are less likely to happen near roads from Group 2, because of their less traffic volume.

6 CONCLUSION

In this paper, we propose a novel framework JMDI to citywide traffic volume inference using dense and incomplete trajectories.

JMDI combines high-fidelity traffic simulator with deep reinforcement learning to recover full vehicle movements from incomplete trajectories. The masked semi-supervised approach enhanced by multi-view graph embedding is introduced for citywide traffic volume inference. We evaluate JMDI in two regions in a provincial capital city in China to demonstrate its effectiveness.

REFERENCES

[1] Andreas Argyriou, Mark Herbster, and Massimiliano Pontil. 2006. Combining graph Laplacians for semi–supervised learning. In *Advances in Neural Information Processing Systems*. 67–74.

[2] Muhammad Tayyab Asif, Nikola Mitrovic, Justin Dauwels, and Patrick Jaillet. 2016. Matrix and tensor based methods for missing data estimation in large traffic networks. *IEEE Transactions on Intelligent Transportation Systems* 17, 7 (2016), 1816–1825.

[3] Javed Aslam, Sejoon Lim, Xinghao Pan, and Daniela Rus. 2012. City-scale traffic estimation from a roving sensor network. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*. ACM, 141–154.

[4] Prithu Banerjee, Sayan Ranu, and Sriram Raghavan. 2014. Inferring uncertain trajectories from partial observations. In *Data Mining (ICDM), 2014 IEEE International Conference on*. IEEE, 30–39.

[5] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 785–794.

[6] Mark Culp and George Michailidis. 2008. Graph-based semisupervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 1 (2008), 174–179.

[7] Tran Duc Duan, TL Hong Du, Tran Vinh Phuoc, and Nguyen Viet Hoang. 2005. Building an automatic vehicle license plate recognition system. In *Proc. Int. Conf. Comput. Sci. RIVF*. Citeseer, 59–63.

[8] Astrid Gühnemann, Ralf-Peter Schäfer, Kai-Uwe Thiessenhusen, and Peter Wagner. 2004. Monitoring traffic and emissions by floating car data. (2004).

[9] Huiqi Hu, Guoliang Li, Zhifeng Bao, Yan Cui, and Jianhua Feng. 2016. Crowdsourcing-based real-time urban traffic speed estimation: From trends to speeds. In *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*. IEEE, 883–894.

[10] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[11] Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker. 2012. Recent Development and Applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements* 5, 3&4 (December 2012), 128–138.

[12] Jaimyoung Kwon, Pravin Varaiya, and Alexander Skabardonis. 2003. Estimation of truck traffic volume from single loop detectors with lane-to-lane speed correlation. *Transportation Research Record: Journal of the Transportation Research Board* 1856 (2003), 106–117.

[13] Li Li, Yuebiao Li, and Zhiheng Li. 2013. Efficient missing data imputing for traffic flow by considering temporal and spatial dependence. *Transportation research part C: emerging technologies* 34 (2013), 108–120.

[14] Mu Li, Amr Ahmed, and Alexander J Smola. 2015. Inferring movement trajectories from GPS snippets. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. ACM, 325–334.

[15] Weizi Li, Dong Nie, David Wilkie, and Ming C Lin. 2017. Citywide estimation of traffic dynamics via sparse gps traces. *IEEE Intelligent Transportation Systems Magazine* 9, 3 (2017), 100–113.

[16] Yandong Li, Liqiang Wang, Tianbao Yang, and Boqing Gong. 2018. How Local is the Local Diversity? Reinforcing Sequential Determinantal Point Processes with Dynamic Ground Sets for Supervised Video Summarization. In *The European Conference on Computer Vision (ECCV)*.

[17] Chuishi Meng, Xiuwen Yi, Lu Su, Jing Gao, and Yu Zheng. 2017. City-wide Traffic Volume Inference with Loop Detector Data and Taxi Trajectories. In *Proceedings of ACM International Conference on Advances in Geographical Information Systems*.

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[20] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[21] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[22] OpenStreetMap contributors. 2017. OpenStreetMap. https://www.openstreetmap.org.

[23] Li Qu, Li Li, Yi Zhang, and Jianming Hu. 2009. PPCA-based missing data imputation for traffic flow volume: A systematical approach. *IEEE Transactions on intelligent transportation systems* 10, 3 (2009), 512–522.

[24] Li Qu, Yi Zhang, Jianming Hu, Liyan Jia, and Li Li. 2008. A BPCA based missing value imputing method for traffic flow volume data. In *Intelligent Vehicles Symposium, 2008 IEEE*. IEEE, 985–990.

[25] Meng Qu, Jian Tang, Jingbo Shang, Xiang Ren, Ming Zhang, and Jiawei Han. 2017. An Attention-based Collaboration Framework for Multi-View Network Representation Learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1767–1776.

[26] Wenjie Ruan, Peipei Xu, Quan Z Sheng, Nickolas JG Falkner, Xue Li, and Wei Emma Zhang. 2017. Recovering Missing Values from Corrupted Spatio-Temporal Sensory Data via Robust Low-Rank Tensor Completion. In *International Conference on Database Systems for Advanced Applications*. Springer, 607–622.

[27] Zhenyu Shan, Danna Zhao, and Yingjie Xia. 2013. Urban road traffic speed estimation for missing probe vehicle data based on multiple linear regression model. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*. IEEE, 118–123.

[28] Hongjian Wang and Zhenhui Li. 2017. Region Representation Learning via Mobility Flow. 10 (2017). https://doi.org/10.1145/3132847.3133006

[29] Hongjian Wang, Xianfeng Tang, Yu-Hsuan Kuo, Daniel Kifer, and Zhenhui Li. 2019. A simple baseline for travel time estimation using large-scale trip data. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (2019), 19.

[30] Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. 2018. MathDQN: Solving Arithmetic Word Problems via Deep Reinforcement Learning. (2018).

[31] Yilun Wang, Yu Zheng, and Yexiang Xue. 2014. Travel time estimation of a path using sparse trajectories. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 25–34.

[32] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8, 3-4 (1992), 279–292.

[33] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. 2018. Intellilight: A reinforcement learning approach for intelligent traffic light control. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2496–2505.

[34] Ling-Yin Wei, Yu Zheng, and Wen-Chih Peng. 2012. Constructing popular routes from uncertain trajectories. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 195–203.

[35] David Wilkie, Jason Sewall, and Ming Lin. 2013. Flow reconstruction for data-driven traffic animation. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 89.

[36] Yuto Yamaguchi, Christos Faloutsos, and Hiroyuki Kitagawa. 2015. OMNI-Prop: Seamless Node Classification on Arbitrary Label Correlation.. In *AAAI*. 3122–3128.

[37] Ning Yang and Philip S Yu. 2016. Efficient Hidden Trajectory Reconstruction from Sparse Data. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 821–830.

[38] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. 2019. Revisiting Spatial-Temporal Similarity: A Deep Learning Framework for Traffic Prediction. *2019 AAAI Conference on Artificial Intelligence (AAAI'19)*.

[39] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. 2018. Deep Multi-View Spatial-Temporal Network for Taxi Demand Prediction. *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence* (2018).

[40] Xiuwen Yi, Yu Zheng, Junbo Zhang, and Tianrui Li. 2016. ST-MVL: filling missing values in geo-sensory time series data. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. AAAI Press, 2704–2710.

[41] Xianyuan Zhan, Ruimin Li, and Satish V Ukkusuri. 2015. Lane-based real-time queue length estimation using license plate recognition data. *Transportation Research Part C: Emerging Technologies* 57 (2015), 85–102.

[42] Xianyuan Zhan, Yu Zheng, Xiuwen Yi, and Satish V Ukkusuri. 2017. Citywide Traffic Volume Estimation Using Trajectory Data. *IEEE Transactions on Knowledge and Data Engineering* 29, 2 (2017), 272–285.

[43] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A Deep Reinforcement Learning Framework for News Recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 167–176.

[44] Jiangchuan Zheng and Lionel M Ni. 2013. Time-dependent trajectory regression on road networks via multi-task learning. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*. AAAI Press, 1048–1055.

[45] Kai Zheng, Yu Zheng, Xing Xie, and Xiaofang Zhou. 2012. Reducing uncertainty of low-sampling-rate trajectories. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. IEEE, 1144–1155.

[46] Denny Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *Advances in neural information processing systems*. 321–328.

[47] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*. 912–919.

Listening between the Lines: Learning Personal Attributes from Conversations

Anna Tigunova
Max Planck Institute for Informatics
Saarbrücken, Germany
tigunova@mpi-inf.mpg.de

Andrew Yates
Max Planck Institute for Informatics
Saarbrücken, Germany
ayates@mpi-inf.mpg.de

Paramita Mirza
Max Planck Institute for Informatics
Saarbrücken, Germany
paramita@mpi-inf.mpg.de

Gerhard Weikum
Max Planck Institute for Informatics
Saarbrücken, Germany
weikum@mpi-inf.mpg.de

ABSTRACT

Open-domain dialogue agents must be able to converse about many topics while incorporating knowledge about the user into the conversation. In this work we address the acquisition of such knowledge, for personalization in downstream Web applications, by extracting personal attributes from conversations. This problem is more challenging than the established task of information extraction from scientific publications or Wikipedia articles, because dialogues often give merely implicit cues about the speaker. We propose methods for inferring personal attributes, such as profession, age or family status, from conversations using deep learning. Specifically, we propose several *Hidden Attribute Models*, which are neural networks leveraging attention mechanisms and embeddings. Our methods are trained on a per-predicate basis to output rankings of object values for a given subject-predicate combination (e.g., ranking the doctor and nurse professions high when speakers talk about patients, emergency rooms, etc). Experiments with various conversational texts including Reddit discussions, movie scripts and a collection of crowdsourced personal dialogues demonstrate the viability of our methods and their superior performance compared to state-of-the-art baselines.

CCS CONCEPTS

• **Computing methodologies** → *Natural language processing*; **Information extraction**; **Supervised learning by classification**; **Neural networks**.

KEYWORDS

Information extraction; personal knowledge; user profiling; conversational text; neural networks; attention mechanisms

ACM Reference Format:
Anna Tigunova, Andrew Yates, Paramita Mirza, and Gerhard Weikum. 2019. Listening between the Lines: Learning Personal Attributes from Conversations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313498

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313498

1 INTRODUCTION

Motivation and Problem: While interest in dialogue agents has grown rapidly in recent years, creating agents capable of holding personalized conversations remains a challenge. For meaningful and diverse dialogues with a real person, a system should be able to infer knowledge about the person's background from her utterances. Consider the following example where *H* stands for a human and *A* for an agent:

H: *What's the best place for having brekky?*
A: *The porridge at Bread and Cocoa is great.*
H: *Any suggestions for us and the kids later? We already visited the zoo.*
A: *There's the San Fransisco Dungeon, an amusement ride with scary city history.*

From the word 'brekky' in the first *H* utterance, the system understands that the user is Australian and may thus like porridge for breakfast. However, the cue is missed that the user is with pre-teen children (talking about kids and the zoo), and the resulting suggestion is inappropriate for young children. Instead, with awareness of this knowledge, a better reply could have been:

A: *I bet the kids loved the sea lions, so you should see the dolphins at Aquarium of the Bay.*

A possible remedy to improve this situation is to include user information into an end-to-end learning system for the dialogue agent. However, any user information would be bound to latent representations rather than explicit attributes. Instead, we propose to capture such attributes and construct a *personal knowledge base (PKB)* with this information, which will then be a distant source of background knowledge for personalization in downstream applications such as Web-based chatbots and agents in online forums.

The advantages of an explicit PKB are twofold: it is an easily re-usable asset that benefits all kinds of applications (rather than merely fixing the current discourse), and it provides a convenient way of explaining the agent's statements to the human whenever requested. Constructing a PKB involves several key issues:

- What is personalized knowledge about users?
- How can this knowledge be inferred from the users' utterances in conversations?
- How should such background knowledge be incorporated into a dialogue system?

As for the first issue, interesting facts about a user could be attributes (age, gender, occupation, etc), interests and hobbies, relationships to other people (family status, names of friends, etc) or sentiments towards certain topics or people. In this paper's experiments, we focus on crisp attributes like profession and gender.

The second issue is concerned with information extraction from text. However, prior works have mostly focused on well comprehensible text genres such as Wikipedia articles or news stories. These methods do not work as well when conversations are the input. Compared to formal documents, dialogues are noisy, utterances are short, the language used is colloquial, topics are diverse (including smalltalk), and information is often implicit ("between the lines"). This paper addresses these problems by proposing methods that identify the terms that are informative for an attribute and leverage these terms to infer the attribute's value.

A detailed exploration of the third issue is beyond the scope of this paper, which focuses on inferring personal attributes. However, we do partially address the issue of integrating background knowledge by arguing that such information should be captured in an explicit PKB. In addition to being independent of downstream applications, an explicit PKB can provide transparency by allowing users to see what is known about them as well as giving users the opportunity to consent to this data being stored.

State of the Art and its Limitations: Currently the most successful dialogue agents are task-oriented, for instance, supporting users with car navigation or delivery orders (e.g., [23, 25]). General-purpose chatbot-like agents show decent performance in benchmarks (e.g., [13, 20, 37]), but critically rely on sufficient training data and tend to lack robustness when users behave in highly varying ways. Very few approaches have considered incorporating explicit knowledge on individual users, and these approaches have assumed that personal attributes are explicitly mentioned in the text [15, 21, 47].

To illustrate that identifying explicit mentions of attributes is insufficient, we developed an oracle to obtain an upper bound on the performance of pattern-based approaches, such as [21]. This oracle, which is described in Section 5.2, assumes that we have perfect pattern matching that correctly extracts an attribute value every time it is mentioned. (When multiple attribute values are mentioned, we assume the oracle picks the correct one.) This oracle routinely performs substantially worse than our proposed methods, demonstrating that extracting information from utterances requires *listening between the lines* (i.e., inferring the presence of attribute values that are never explicitly stated).

On the other hand, many efforts have considered the problem of profiling social media users in order to predict latent attributes such as age, gender, or regional origin (e.g., [3, 4, 8, 9, 17, 34–36, 41]). While social media posts and utterances are similar in that both are informal, the former can be associated with many non-textual features that are unavailable outside of the social media domain (e.g., social-network friends, likes, etc. and explicit self-portraits of users). We consider several user profiling baselines that rely on only textual features and find that they do not perform well on our task of inferring attributes from conversational utterances.

Approach and Contributions: We devise a neural architecture, called *Hidden Attribute Model (HAM)*, trained with subject-predicate-object triples to predict objects on a per-predicate basis, e.g., for a subject's profession or family status. The underlying neural network learns to predict a scoring of different objects (e.g., different professions) for a given subject-predicate pair by using attention within and across utterances to infer object values. For example, as illustrated later in Table 7, our approach infers that a subject who often uses terms like *theory*, *mathematical*, and *species* is likely to be a *scientist*, while a subject who uses terms like *senate*, *reporters*, and *president* may be a *politician*.

Our salient contributions are the following:

- a viable method for learning personal attributes from conversations, based on neural networks with novel ways of leveraging attention mechanisms and embeddings,
- a data resource of dialogues from movie scripts, with ground truth for professions, genders, and ages of movie characters,
- a data resource of dialogue-like Reddit posts, with ground truth for the authors' professions, genders, ages, and family statuses
- an extensive experimental evaluation of various methods on Reddit, movie script dialogues, and crowdsourced personalized conversations (PersonaChat),
- an experimental evaluation on the transfer learning approach: leveraging ample data from user-generated social media text (Reddit) for inferring users' latent attributes from data-scarce speech-based dialogues (movie scripts and PersonaChat).

2 RELATED WORK

Personal Knowledge from Dialogues: Awareness of background knowledge about users is important in both goal-oriented dialogues [6, 16] and chitchat settings [20, 47]. Such personal knowledge may range from users' intents and goals [6, 42] to users' profiles, with attributes such as home location, gender, age, etc. [16, 20, 22], as well as users' preferences [25, 47].

Prior works on incorporating background knowledge model *personas* and enhance dialogues by injecting personas into the agent's response-generating model [20, 22, 47] or by adapting the agent's speech style [16]. These approaches typically use latent representations of user information. In contrast, there is not much work on capturing explicit knowledge based on user utterances. Li et al. encode implicit speaker-specific information via distributed embeddings that can be used to cluster users along some traits (e.g., age or country) [20]. Zhang et al. describe a preliminary study on predicting simple speaker profiles from a set of dialogue utterances [47]. The most related prior work by Garera and Yarowsky captures latent biographic attributes via SVM models from conversation transcripts and email communication, taking into account *contextual features* such as partner effect and sociolinguistic features (e.g., % of "yeah" occurences), in addition to n-grams [12].

Several works explore information extraction from conversational text to build a personal knowledge base of speakers given their utterances, such as extracting *profession*: *software engineer* and *employment_history*: *Microsoft* from *"I work for Microsoft as a software engineer"*, using maximum-entropy classifiers [15] and sequence-tagging CRFs [21]. However, this approach relies on user attributes being *explicitly mentioned* in the utterances. In contrast,

our method can infer attribute values from implicit cues, e.g., *"I write product code in Redmond."*

Social Media User Profiling: The rapid growth of social media has led to a massive volume of user-generated informal text, which sometimes mimics conversational utterances. A great deal of work has been dedicated to automatically identify latent demographic features of online users, including age and gender [3, 4, 8, 9, 17, 34–36, 41], political orientation and ethnicity [26, 32–34, 41], regional origin [8, 34], personality [14, 36], as well as occupational class that can be mapped to income [10, 31]. Most of these works focus on user-generated content from Twitter, with a few exceptions that explore Facebook [35, 36] or Reddit [8, 14] posts.

Most existing studies on social media to capture users' latent attributes rely on classification over hand-crafted features such as word/character n-grams [2, 4, 34], Linguistic Inquiry and Word Count (LIWC) [27] categories [14, 32, 33], topic distributions [9, 26, 31] and sentiment/emotion labels of words derived from existing emotion lexicon [14, 26, 32, 33]. The best performing system [2] in the shared task on author profiling organized by the CLEF PAN lab [11, 30] utilizes a linear SVM and word/character n-gram features.

There have been only limited efforts to identify attributes of social media users using neural network approaches. Bayot and Gonçalves explore the use of Convolutional Neural Networks (CNNs) together with word2vec embeddings to perform user profiling (age and gender) of English and Spanish tweets [3]. Kim et al. employ Graph Recursive Neural Networks (GRNNs) to infer demographic characteristics of users [17]. Vijayaraghavan et al. exploit attention-based models to identify demographic attributes of Twitter users given multi-modal features extracted from users' profiles (e.g., name, profile picture, and description), social network, and tweets [41].

The vast majority of these works rely on features specific to social media such as hashtags, users' profile descriptions and social network structure, with only [2, 3, 31] inferring users' latent attributes based solely on user-generated text. In our evaluation we consider these three methods as baselines.

Neural Models with Attention: Recently, neural models enhanced with *attention* mechanisms have boosted the results of various NLP tasks [38, 44, 48], particularly in neural conversation models for generating responses [43, 45] and in predicting user intents [6]. While our attention approach bears some similarity to [43] in that both consider attention on utterances, both our approach's architecture and our use case differ substantially. The role of attention weights has been studied for various neural models, including feed-forward networks [40], CNNs, [46] and RNNs [1].

3 METHODOLOGY

In this section we propose Hidden Attribute Models (HAMs) for ranking object values given a predicate and a sequence of utterances made by a subject. For example, given the movie script excerpt shown in Figure 1, the *profession* predicate, and the subject *Edwards*, *policeman* should be ranked as the subject's most likely profession.

More formally, given a subject S and a predicate P, our goal is to predict a probability distribution over object values O for the predicate based on the subject's utterances from a dialogue corpus (e.g., a movie script). Each subject S is associated with a sequence of N utterances $[U_1, U_2, ..., U_N]$ containing M terms each, $U_1 = [U_{1,1}, U_{1,2}, ..., U_{1,M}]$. Each term $U_{n,m}$ is represented as a d-dimensional word embedding.

EDWARDS
Put down the gun and put your hands on the counter!

KAY
I warned him.

EDWARDS
Drop the weapon!

KAY
You warned him.

EDWARDS
You are under arrest.
You have the right to remain silent.

Figure 1: An excerpt from *Men in Black* (1997).

HAMs can be described in terms of three functions and their outputs:

(1) f_{utter} creates a representation R_n^{utter} of the nth utterance given the terms in the utterance:

$$R_n^{utter} = f_{utter}(U_{n,1}, U_{n,2}, ..., U_{n,M}) \tag{1}$$

(2) f_{subj} creates a subject representation R^{subj} given the sequence of utterance representations:

$$R^{subj} = f_{subj}(R_1^{utter}, R_2^{utter}, ..., R_N^{utter}) \tag{2}$$

(3) f_{obj} outputs a probability distribution over object values O given the subject representation:

$$O = f_{obj}(R^{subj}) \tag{3}$$

In the following sections we describe our proposed HAMs by instantiating these functions.

3.1 Hidden Attribute Models

HAM_{avg} illustrates the most straightforward way to combine embeddings and utterances. In this model,

$$avg(X) = \sum_{i=1}^{|X|} X_i \tag{4}$$

serves as both f_{utter} and f_{subj}; the nth utterance representation R_n^{utter} is created by averaging the terms in the nth utterance and the subject representation R^{subj} is created by averaging the N utterance representations together. Two stacked fully connected layers serve as the function f_{obj},

$$FC(x) = \sigma(Wx + b) \tag{5}$$

where σ is an activation function and W and b are learned weights. The full HAM_{avg} model is then

$$R_n^{utter} = avg(U_n) \tag{6}$$

$$R^{subj} = avg(R^{utter}) \tag{7}$$

$$O = FC_1(FC_2(R^{subj})) \tag{8}$$

where FC_2 uses a sigmoid activation and FC_1 uses a softmax activation function in order to predict a probability distribution over object values.

HAM$_{2attn}$ extends HAM$_{avg}$ with two attention mechanisms allowing the model to learn which terms and utterances to focus on for the given predicate. In this model the utterance representations and subject representations are computed using attention-weighted averages,

$$attn\text{-}avg(X, \alpha) = \sum_{i=1}^{|X|} X_i \alpha_i \tag{9}$$

with the attention weights calculated over utterance terms and utterance representations, respectively. That is, $f_{utter}(X) = attn\text{-}avg(X, \alpha^{term})$ and $f_{subj}(X) = attn\text{-}avg(X, \alpha^{utter})$, where the attention weights for each term in an utterance U_i are calculated as

$$w_i^{term} = \sigma(W^{term} U_i + b^{term}) \tag{10}$$

$$\alpha_{i,j}^{term} = \frac{exp(w_{i,j}^{term})}{\sum_j exp(w_{i,j}^{term})} \tag{11}$$

and the utterance representation weights α^{utter} are calculated analogously over R^{utter}. Given these attention weights, the HAM$_{2attn}$ model is

$$R_n^{utter} = attn\text{-}avg(U_n, \alpha^{term}) \tag{12}$$

$$R^{subj} = attn\text{-}avg(R^{utter}, \alpha^{utter}) \tag{13}$$

$$O = FC(R^{subj}) \tag{14}$$

where f_{obj} function FC uses a softmax activation function as in the previous model.

HAM$_{CNN}$ considers n-grams when building utterance representations, unlike both previous models that treat each utterance as a bag of words. In this model f_{utter} is implemented with a text classification CNN [18] with a ReLU activation function and k-max pooling across utterance terms (i.e., each filter's top k values are kept). A second k-max pooling operation across utterance representations serves as f_{subj}. As in the previous model, a single fully connected layer with a softmax activation function serves as f_{obj}.

HAM$_{CNN\text{-}attn}$ extends HAM$_{CNN}$ by using attention to combine utterance representations into the subject representation. This mirrors the approach used by HAM$_{2attn}$, with $f_{subj} = attn\text{-}avg(X, \alpha^{utter})$ and α^{utter} computed using equations 10 and 11 as before. This model uses the same f_{utter} and f_{obj} as HAM$_{CNN}$. That is, utterance representations are produced using a CNN with k-max pooling, and a single fully connected layer produces the model's output.

3.2 Training

All HAMs were trained with gradient descent to minimize a categorical cross-entropy loss. We use the Adam optimizer [19] with its default values and apply an L2 weight decay (2e-7) to the loss.

4 DATA ACQUISITION AND PROCESSING

MovieChAtt dataset. Following prior work on personalized dialogue systems, we explore the applicability of fictional dialogues from TV or movie scripts to approximate real-life conversations [20, 22]. Specifically, we compile a Movie Character Attributes (MovieChAtt) dataset consisting of characters' utterances and the characters' attributes (e.g., profession).

To create MovieChAtt, we labeled a subset of characters in the Cornell Movie-Dialogs Corpus[1] [7] of 617 movie scripts. From each movie, we derive a sequence of utterances for each character, excluding characters who have less than 20 lines in the movie. Each utterance is represented as a sequence of words, excluding stop words, the 1,000 most common first names[2], and words that occur in fewer than four different movies. The latter two types of words are excluded in order to prevent the model from relying on movie-specific or character-specific signals that will not generalize.

We extracted characters' *gender* and *age* attributes by associating characters with their entries in the Internet Movie Database (IMDb) and extracting the corresponding actor or actress' attributes at the time the movie was filmed. This yielded 1,963 characters labeled with their genders and 4,548 characters labeled with their ages. We discretized the age attribute into the following ranges: (i) 0–13: *child*, (ii) 14–23: *teenager*, (iii) 24–45: *adult*, (iv) 46–65: *middle-aged* and (v) 66–100: *senior*. In our data the distribution of age categories is highly imbalanced, with *adult* characters dominating the dataset (58.7%) and *child* being the smallest category (1.7%).

To obtain the ground-truth labels of characters' *profession* attributes, we conducted a Mechanical Turk (MTurk) crowdsourcing task to annotate 517 of the movies in our corpus. Workers were asked to indicate the professions of characters in a movie given the movie's Wikipedia article. Workers were instructed to select professions from a general predefined list if possible (e.g., doctor, engineer, military personnel), and to enter a new profession label when necessary. We manually defined and refined the list of professions based on several iterations of MTurk studies to ensure high coverage and to reduce ambiguity in the options (e.g., *journalist* vs *reporter*). We also included non-occupational "professions" that often occur in movies, such as *child* and *criminal*.

Fleiss' kappa for the crowdworkers' inter-annotator agreement is 0.47. Disagreement was oftentimes caused by one character having multiple professions (Batman is both a *superhero* and a *businessman*), or a change of professions in the storyline (from *banker* to *unemployed*). We kept only characters for which at least 2 out of 3 workers agreed on their profession, which yielded 1405 characters labeled with 43 distinct professions. The highly imbalanced distribution of professions reflects the bias in our movie dataset, which features more *criminals* and *detectives* than *waiters* or *engineers*.

PersonaChat dataset. We also explore the robustness of our models using the PersonaChat corpus[3] [47], which consists of conversations collected via Mechanical Turk. Workers were given persona descriptions consisting of 5-sentence-long descriptions (e.g., "I am an artist", "I like to build model spaceships") and asked to incorporate these facts into a short conversation (up to 8 turns) with another worker. We split these conversations by persona, yielding a sequence of 3 or 4 utterances for each persona in a conversation.

We automatically labeled personas with *profession* and *gender* attributes by looking in the persona description for patterns "I am/I'm a(n) ⟨*term*⟩", where ⟨*term*⟩ is either a profession label or a gender-indicating noun (woman, uncle, mother, etc). We manually labeled

[1] http://www.cs.cornell.edu/ cristian/Cornell_Movie-Dialogs_Corpus.html
[2] Removed to prevent overfitting. http://catalog.data.gov/dataset/baby-names-from-social-security-card-applications-national-level-data
[3] http://convai.io/#personachat-convai2-dataset

attribute	pattern
profession	"(I|i) (am|'m)"+[one of the profession names]
gender	"(I|i) (am|'m)"+[words, indicating gender (*lady*, *father*, etc.)]
age	"(I|i) (am|'m)"+number (5-100)+"years old" "(I|i) (am|'m|was) born in"+number (1920-2013)
family status	"(I|i) (am|'m)"+[*married*, *divorced*, *single*, ...] "(M|m)y"|"(I|i) have a"+[*wife*, *boyfriend*, ...]

Table 1: Patterns for extracting ground truth labels from Reddit posts.

persons with *family status* by identifying persona descriptions containing related words (single, married, lover, etc) and labeling the corresponding persona as single or not single. Overall, we collected 1,147 personas labeled with profession, 1,316 with gender, and 2,302 labeled with family status.

Reddit dataset. As a proxy for dialogues we used discussions from Reddit online forums. We used a publicly available crawl[4] spanning the years from 2006 to 2017. Specifically, we tapped into two subforums on Reddit: "iama" where anyone can ask questions to a particular person, and "askreddit" with more general conversations. In selecting these subforums we followed two criteria: 1) they are not concerned with fictional topics (e.g. computer games) and 2) they are not too topic-specific, as this could heavily bias the classification of user attributes.

To create ground-truth labels for users, we searched for posts that matched specific patterns. The list of patterns for all four attributes is given in Table 1. For the case of *profession*, we created a list of synonyms and subprofessions for better coverage. For example, the list for *businessperson* includes *entrepreneur*, *merchant*, *trader*, etc.

We selected only posts of length between 10 and 100 words by users who have between 20 and 100 posts. Also, we removed users who claim multiple values for the same attribute, as we allow only for single-label classification. To further increase data quality, we rated users by language style, to give preference to those whose posts sound more like the utterances in a dialogue. This rating was computed by the fraction of a user's posts that contain personal pronouns, as pronouns are known to be abundant in dialogue data.

The test set, disjoint from training data and created in the same manner, was further checked by manual annotators, because the above mentioned patterns may also produce false positives, for example, the wrong profession from utterances such as "they think I am a doctor" or "I dreamed I am an astronout", or the wrong age and family status from "I am 10 years old boy's mother" or "I am a single child". The final Reddit test set consists of approximately 400 users per predicate.

Limitations. The predicates we consider are limited by the nature of our datasets. We do not consider the *family status* predicate for MovieChAtt, because the necessary information is often not easily available from Wikipedia articles. Similarly, we do not consider the *age* predicate on the PersonaChat dataset, because this attribute is not easily found in the persona descriptions. More generally, all users in our datasets are labeled with exactly one attribute value for each predicate. In a real setting it may be impossible to infer any attribute value for some users, whereas other users may have multiple correct values. We leave such scenarios for future work.

Our code and dataset annotations are available online.[5][6]

[4] https://files.pushshift.io/reddit/

5 EXPERIMENTAL SETUP

5.1 Data

We randomly split the MovieChAtt and PersonaChat datasets into training (90%) and testing (10%) sets. The Reddit test set consists of approximately 400 users per predicate whose label extractions were manually verified by annotators. We tuned models' hyperparameters by performing a grid search with 10-fold cross validation on the training set.

For the binary predicates family status and gender, we balanced the number of subjects in each class. For the multi-valued profession and age predicates, which have very skewed distributions, we did not balance the number of subjects in the test set. During training we performed downsampling to reduce the imbalance; each batch consisted of an equal amount (set to 3) of training samples per class, and these samples were drawn randomly for each batch. This both removes the class imbalance in training data and ensures that ultimately the model sees all instances during training, regardless of the class size.

Note that all three datasets are somewhat biased regarding the attribute values and not representative for real-life applications. For example, the professions in MovieChAtt are dominated by the themes of entertaining fiction, and gender distributions are not realistic either. The data simply provides a diverse range of inputs for fair comparison across different extraction methods.

5.2 Baselines

We consider the following baselines to compare our approach against.

Pattern matching oracle. Assuming that we have a perfect sequence tagging model (e.g., [21]) that extracts a correct attribute value every time one appears in an utterance, we can determine the upper-bound performance of such sequence tagging approaches. Note that, as mentioned in the introduction, this type of model assumes attribute values explicitly appear in the text. In order to avoid vocabulary mismatches between our attribute value lists and the attribute values explicitly mentioned in the data, we augment our attribute values with synonyms identified in the data (e.g., we add terms like 'soldier' and 'sergeant' as synonyms of the the *profession* attribute's value 'military personnel'). For both MRR and accuracy, a subject receives a score of 1 if the correct attribute value appears in any one of a subject's utterances. If the correct attribute value never appears, we assume the model returns a random ordering of attribute values and use the expectation over this list (i.e., given $|V|$ attribute values, the subject receives a score of $\frac{1}{0.5|V|}$ for MRR and $\frac{1}{|V|}$ for accuracy). This oracle method does not provide class confidence scores, so we do not report AUROC with this method.

[5] https://github.com/Anna146/HiddenAttributeModels

[6] https://mpi-inf.mpg.de/departments/databases-and-information-systems/research/personal-knowledge-base

Embedding similarity. Given an utterance representation created by averaging the embeddings of the words within the utterance, we compute the cosine similarity between this representation and the embeddings of each attribute value.

Logistic regression. Given an averaged utterance representation (as used with embedding similarity), we apply a multinomial logistic regression model to classify the representation as one of the possible attribute values. This model obtains a ranking by ordering on the per-class probabilities of its output.

Multilayer Perceptron (MLP). Given an averaged utterance representation (as used with embedding similarity), we apply an MLP with one hidden layer of size 100 to classify the utterance representation as one of the possible attribute values. This model can be used to obtain a ranking by considering the per-class probabilities.

The embedding similarity, logistic regression, and MLP baselines are distantly supervised, because the subject's labels are applied to each of the subject's utterances. While this is necessary because the baselines do not incorporate the notion of a subject with multiple utterances, it results in noisy labels because it is unlikely that every utterance will contain information about the subject's attributes. We address this issue by using a window of $k = 4$ (determined by a grid search) concatenated utterances as input to each of these methods. With these distantly supervised models, the label prediction scores are summed across all utterances for a single subject and then ranked.

CNN [3]. We consider the Convolutional Neural Network (CNN) architecture proposed by Bayot and Gonçalves for the task of predititng the age and gender of Twitter users. This approach is a simpler variant of HAM_{CNN} in which f_{utter} is implemented with a *tanh* activation function and max pooling (i.e., $k = 1$) and f_{obj} is a fully connected layer with dropout ($p = 0.5$) and a softmax activation function. The CNN is applied to individual utterances and the majority classification label is used for the user, which differs from the in-model aggregation performed by HAM_{CNN}.

New Groningen Author-profiling Model (N-GrAM) [2]. Following the best performing system at CLEF 2017's PAN shared task on *author profiling* [11], we implemented a classification model using a linear Support Vector Machine (SVM) that utilizes the following features: character n-grams with $n = 3, 4, 5$, and term unigrams and bigrams with sublinear tf-idf weighting.

Neural Clusters (W2V-C) [31]. We consider the best classification model reported by Preoţiuc-Pietro et al. for predicting the occupational class of Twitter users, which is a Gaussian Process (GP) classifier with *neural clusters* (W2V-C) as features. Neural clusters were obtained by applying spectral clustering on a word similarity matrix (via cosine similarity of pre-trained word embeddings) to obtain $n = 200$ word clusters. Each post's feature vector is then represented as the ratio of words from each cluster.

For both N-GrAM and W2V-C baselines, flattened representations of the subject's utterances are used. That is, the model's input is a concatenation of all of a given user's utterances.

5.3 Hyperparameters

Hyperparameters were chosen by grid search using ten-fold cross validation on the training set. Due to the limited amount of data, we found a minibatch size of 4 users performed best on MovieChAtt and PersonaChat. All models were trained with a minibatch size of 32 on Reddit. We used 300-dimensional word2vec embeddings pretrained on the Google News corpus [24] to represent the terms in utterances. We set the number of utterances per character $N = 40$ and the number of terms per utterance $M = 40$, and truncated or zero padded the sequences as needed.

- $\textbf{HAM}_{\textbf{avg}}$ uses a hidden layer of size 100 with the sigmoid activation function. The model was trained for 30 epochs.
- $\textbf{HAM}_{\textbf{CNN}}$ uses 178 kernels of size 2 and k-max pooling with $k = 5$. The model was trained for 40 epochs.
- $\textbf{HAM}_{\textbf{2attn}}$ uses a sigmoid activation with both attention layers and with the prediction layer. The model was trained for 150 epochs.
- $\textbf{HAM}_{\textbf{CNN-attn}}$ uses 128 kernels of size 2. The model was trained for 50 epochs.

5.4 Evaluation metrics

For binary predicates (*gender* and *family status*), we report models' performance in terms of accuracy. Due to the difficulty of performing classification over many attribute values, a ranking metric is more informative for the multi-valued predicates (*profession* and *age category*). We report Mean Reciprocal Rank for these predicates.

Mean Reciprocal Rank (MRR) measures the position of the correct answer in a ranked list of attribute values provided by the model. We obtain a ranking of attribute values for a movie character or a persona by considering the attribute value probabilities output by a model. Given one ranked list of attribute values per character, MRR is computed by determining the reciprocal rank of the correct attribute value in each list, and then taking the average of these reciprocal ranks. We report both *macro MRR*, in which we calculate a reciprocal rank for each attribute value before averaging, and *micro MRR*, which averages across each subject's reciprocal rank.

Area under the ROC Curve (AUROC) measures the performance of a model as a function of its true positive rate and false positive rate at varying score thresholds. We report micro AUROC for all predicates. For multi-valued predicates, we binarize the labels in a one-vs-all fashion.

6 RESULTS AND DISCUSSION

6.1 Main Findings

In Table 2, 3 and 4 we report results for the HAMs and the baselines on all datasets (MovieChAtt, PersonaChat and Reddit) for all considered attributes (*profession*, *gender*, *age* and *family status*). We do not report results for the *pattern oracle* baseline in the tables as we evaluated the baseline solely on the MovieChAtt dataset, because the oracle essentially replicates the way we labeled persona descriptions and posts in the PersonaChat and Reddit datasets, respectively. The pattern oracle baseline yields 0.21/0.20 micro/macro MRR for *profession*, 0.67 accuracy for *gender*, and 0.41/0.40 micro/macro MRR for *age*. HAMs significantly outperform this baseline, indicating

Models	profession						gender					
	MovieChAtt		PersonaChat		Reddit		MovieChAtt		PersonaChat		Reddit	
	MRR micro / macro	AUROC	MRR micro / macro	AUROC	MRR micro / macro	AUROC	Acc	AUROC	Acc	AUROC	Acc	AUROC
Embedding sim.	0.22* / 0.14*	0.60*	0.30* / 0.25*	0.63*	0.15* / 0.13*	0.59*	0.52*	0.54*	0.49	0.50	0.61*	0.60*
Logistic reg.	0.46* / 0.20*	0.76*	0.81* / 0.77*	0.58*	0.13* / 0.19*	0.57	0.59	0.62	0.86	0.93	0.69*	0.75*
MLP	**0.47** / 0.20	0.75	0.86* / 0.77*	0.97	0.46 / 0.23	0.78	0.57*	0.60*	0.80	0.87	0.71	0.77
N-GrAM [2]	0.21* / 0.16*	0.62*	0.83* / 0.83	0.88	0.17 / 0.26	0.64*	0.57	0.58	0.86	0.87	0.66*	0.71*
W2V-C [31]	0.25* / 0.13*	0.74*	0.59 / 0.46	0.89	0.27* / 0.17*	0.74*	0.62	0.66	0.73*	0.80*	0.64*	0.73*
CNN [3]	0.19* / 0.20*	0.66*	0.77* / 0.77*	0.81*	0.26* / 0.24*	0.76*	0.60	0.60	0.72*	0.73*	0.61*	0.61*
HAM_{avg}	0.39* / 0.37*	0.81*	0.86* / 0.91*	0.98*	0.34* / 0.22*	0.82*	0.72	0.82	0.79	0.87	**0.86**	0.92
HAM_{CNN}	0.42 / 0.37	0.83	**0.96** / **0.94**	**0.99**	0.36* / 0.37*	0.86*	0.75	**0.85**	0.95	**0.99**	**0.86**	0.93*
$HAM_{CNN\text{-}attn}$	0.43 / **0.50**	**0.85**	0.90 / 0.93	**0.99**	**0.51** / 0.40	**0.9**	**0.77**	0.84	**0.96**	0.97	0.85	**0.94**
HAM_{2attn}	0.39 / 0.34	0.84	0.94 / 0.93	**0.99**	0.43 / **0.42**	0.89	0.69	0.77	0.94	0.98	0.80	0.91

Table 2: Comparison of models on all datasets for *profession* and *gender* attributes. Results marked with * significantly differ from the best method (in bold face) with $p < 0.05$ as measured by a paired t-test (MRR) or McNemar's test (Acc and AUROC).

that identifying explicit mentions of attribute values is insufficient in our dialogue setting.

HAMs outperform the distantly supervised models (i.e., *embedding similarity*, *logistic regression* and *Multilayer Perceptron (MLP)*) in the vast majority of cases. MLP and logistic regression perform best in several occasions for *profession* and *age* attributes when micro MRR is considered. However, their macro MRR scores fall behind HAMs, showing that HAMs are better at dealing with multi-valued attributes having skewed distribution. The low performance of these distantly supervised methods may be related to their strong assumption that every sequence of four utterances contains information about the attribute being predicted.

Comparing with baselines from prior work, HAMs significantly outperform N-GrAM [2] in many cases, suggesting that representing utterances using word embeddings, instead of merely character and word n-grams, is important for this task. Using neural clusters (W2V-C) as features for the classification task [31] works quite well for the *age* attribute, where different 'topics' may correlate with different age categories (e.g. 'video game' for *teenager* and 'office' for *adult*). However, W2V-C is often significantly worse for the *profession*, *gender*, and *family status* attributes, which may be caused by similar discriminative words (e.g., 'husband'/'wife' for *gender*) being clustered together in the same topic. The CNN baseline [3] is significantly worse than the best method in the majority of cases. Furthermore, it generally performs substantially worse than HAM_{CNN}, further illustrating the advantage of aggregating utterances within the model.

In general, HAM_{avg} performs worse than the other HAMs[7], demonstrating that simple averaging is insufficient for representing utterances and subjects. In most cases HAM_{CNN} performs slightly worse than $HAM_{CNN\text{-}attn}$, demonstrating the value of exploiting an attention mechanism to combine subject's utterances. $HAM_{CNN\text{-}attn}$ and HAM_{2attn} achieve the strongest performance across predicates, with $HAM_{CNN\text{-}attn}$ generally performing better. $HAM_{CNN\text{-}attn}$ performs particularly well on the *gender* and *family status* attributes, where detecting bigrams may yield an advantage. For example, HAM_{2attn} places high attention weights on terms like 'family' and 'girlfriend' where the previous term may be a useful signal (e.g., 'my family' vs. 'that family').

The gap between the baselines and HAMs is often smaller on PersonaChat compared with on the other two datasets, illustrating the simplicity of crowdsourced-dialogue as compared to movie scripts or Reddit discussions. This is also supported by the fact that the maximum metrics on PersonaChat are much higher. There are several factors that may be responsible for this: *(1)* the dialogue in PersonaChat was created by crowdworkers with the goal of stating facts that were given to them, which often leads to the facts being stated in a straightforward manner (e.g., "I am an author"); *(2)* PersonaChat utterances are much shorter and there are far fewer utterances per character (i.e., a maximum of 4 in PersonaChat vs. a minimum of 20 in MovieChAtt), leading to a higher density of information related to attributes; and *(3)* the persona descriptions in PersonaChat are used for many dialogues, giving models an opportunity to learn specific personas.

6.2 Study on word embeddings

In the previous experiments, we represented terms using embeddings from a word2vec skip-gram model trained on Google News. [24] In this study we compare the Google News embeddings with word2vec embeddings trained on Reddit posts, GloVe [28] embeddings trained on Common Crawl, and GloVe embeddings trained on Twitter. We also consider *ELMo* [29], a contextualized embedding model. To capture semantic variations, this model creates a contextualized character-based representation of words using a bidirectional language model. We use AllenNLP's small ELMo model[8] trained on the 1 Billion Word Benchmark of news crawl data from WMT 2011 [5].

[7] For the sake of brevity we neither instantiate nor report results for LSTM-based HAMs, such as $f_{utter} = LSTM$ and $f_{subj} = attn\text{-}avg$ or $f_{subj} = LSTM$. These models were unable to outperform HAM_{avg}, with the best variant obtaining a micro MRR of only 0.31 after grid search (profession predicate on MovieChAtt; Table 2). This is in line with recent results suggesting that RNNs are not ideal for identifying semantic features [39].

[8] https://allennlp.org/elmo

Models	age			
	MovieChAtt		Reddit	
	MRR micro / macro	AU-ROC	MRR micro / macro	AU-ROC
Embedding sim.	0.45* / 0.45*	0.61*	0.55* / 0.44*	0.56*
Logistic reg.	0.65* / 0.49*	0.76	**0.80** / 0.61	0.87
MLP	0.64* / 0.48*	0.83	0.78 / 0.48	0.88
N-GrAM [2]	0.69 / 0.47	0.85	0.48* / 0.53*	0.55*
W2V-C [31]	0.67 / 0.45	0.86	0.75 / 0.51	0.88
CNN [3]	0.66* / 0.62*	0.83	0.68* / 0.65*	0.79*
HAM_{avg}	0.62* / 0.59	0.76*	0.67 / 0.67	0.77*
HAM_{CNN}	0.73* / **0.63**	0.84	0.73* / 0.61*	0.89*
$HAM_{CNN\text{-}attn}$	0.73 / 0.60	**0.86**	0.79 / **0.68**	**0.90**
HAM_{2attn}	**0.74** / 0.6	0.85	0.72 / 0.6	0.82

Table 3: Comparison of models on all datasets for *age* attribute. Results marked with * significantly differ from the best method (in bold face) with $p < 0.05$ as measured by a paired t-test (MRR) or McNemar's test (Acc and AUROC).

Models	family status			
	PersonaChat		Reddit	
	Acc	AUROC	Acc	AUROC
Embedding sim.	0.41*	0.49*	0.42*	0.47*
Logistic reg.	0.75*	0.84*	0.71	0.74
MLP	0.70	0.80	0.62*	0.60*
N-GrAM [2]	0.85	0.86	0.45*	0.47*
W2V-C [31]	0.74*	0.82*	0.70	0.78
CNN [3]	0.74	0.74	0.69	0.69
HAM_{avg}	0.80	0.91	0.67	0.72
HAM_{CNN}	**0.93**	**0.99**	0.52*	0.62*
$HAM_{CNN\text{-}attn}$	0.92	0.98	**0.70**	**0.78**
HAM_{2attn}	0.88	0.94	0.64	0.67

Table 4: Comparison of models on all datasets for *family status* attribute. Results marked with * significantly differ from the best method (in bold face) with $p < 0.05$ as measured by a paired t-test (MRR) or McNemar's test (Acc and AUROC).

Given space limitations and the higher model variance on the *profession* attribute on MovieChAtt, we restrict the study to this predicate and corpus. We evaluated the two best performing HAMs, i.e., $HAM_{CNN\text{-}attn}$ and HAM_{2attn}. Table 5 shows the results obtained with the various embedding methods trained on different datasets. The difference in performance does not greatly vary across embedding models and datasets, with Google News embeddings performing best in terms of macro MRR and Reddit embeddings performing best in terms of micro MRR. Despite their strong performance on some NLP tasks, the ELMo contextualized embeddings do not yield a performance boost for any method or metric. We view this observation as an indicator that the choice of term embedding method is not very significant for this task compared to the method used to combine terms into an utterance representation.

Model	Corpus	$HAM_{CNN\text{-}attn}$		HAM_{2attn}	
		MRR micro / macro	AU-ROC	MRR micro / macro	AU-ROC
word2vec (skip-gram)	Google News	0.42 / **0.44**	0.77	0.39 / 0.37	**0.83**
	Reddit	**0.43** / 0.37	**0.82**	**0.50** / 0.37	**0.83**
GloVe	Common Crawl	0.40 / 0.37	0.76	0.40 / **0.39**	0.82
	Twitter	0.39 / 0.35	0.67	0.36 / 0.34	0.81
ELMo	WMT News	0.38 / 0.32	0.76	0.37 / 0.37	0.83

Table 5: Comparison of embedding models trained on different datasets, for identifying *profession* attribute.

	MRR micro / macro	AUROC
HAM_{2attn}	**0.57** / **0.42**	**0.84**
– attention on terms	0.49 / 0.40	0.81
– attention on R^{utter}	0.48 / 0.34	0.82

Table 6: Ablation study for the *profession* attribute.

6.3 Ablation study

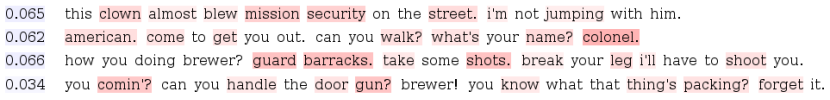
We performed an ablation study in order to determine the performance impact of the HAMs' components. Given space limitations and the higher model variance on the *profession* attribute on MovieChAtt, we restrict the study to this predicate and dataset. Ablation results for HAM_{2attn} using cross validation on the training set are shown in Table 6. Replacing either representation function (i.e., f_{utter} or f_{subj}) with an averaging operation reduces performance, as shown in the last two lines. Attention on utterance representations (R^{utter}) is slightly more important in terms of MRR, but both types of attention contribute to HAM_{2attn}'s performance. Similarly, removing both types of attention corresponds to HAM_{avg}, which consistently underperforms HAM_{2attn} in Table 2, 3 and 4.

Removing attention from $HAM_{CNN\text{-}attn}$ yields HAM_{CNN}, which consistently performs worse than $HAM_{CNN\text{-}attn}$ in Table 2, 3 and 4, supporting the observation that attention is important for performance on our task. Intuitively, attention provides a useful signal because it allows the model to focus on only those terms that provide information about an attribute.

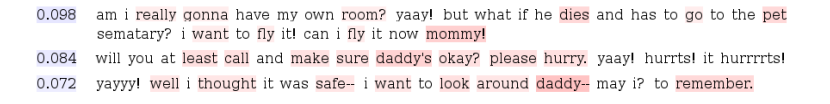
6.4 Case study on attention weights

In order to illustrate the types of terms the model is looking for, we display HAM_{2attn}'s term and utterance weights for *profession* and *age* attributes (on MovieChAtt) in Figure 2, as well as *gender* and *family status* attributes (on Reddit) in Figure 3. While HAM_{2attn} is sometimes outperformed by $HAM_{CNN\text{-}attn}$, this model is more interpretable because individual terms are considered in isolation. Note that all these dialogue snippets come from the specific datasets in our experiments, partly from fictional conversations. Some are skewed and biased, and not representative for the envisioned downstream applications.

When predicting *military personnel* as the *profession* (Figure 2a), the model focuses on military terms such as *mission*, *guard*, *barracks*,

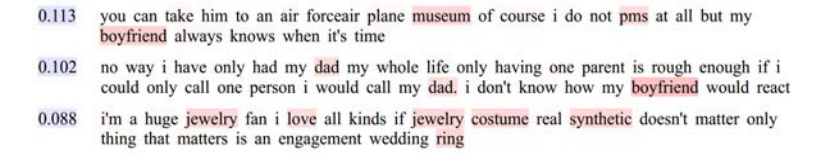


(a) *profession*: military personnel

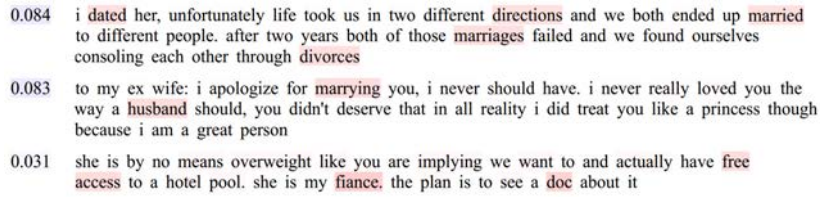


(b) *age* (category): child

Figure 2: Attention visualization for *profession* and *age* predicates on MovieChAtt.



(a) *gender*: female



(b) *family status*: married

Figure 3: Attention visualization for *gender* and *family status* predicates on Reddit.

and *colonel*. When predicting *child* as the *age category* (Figure 2b), on the other hand, the model focuses on terms a child is likely to use, such as *pet*, *mommy*, and *daddy*. According to Reddit posts, *female* gender is suggested by terms such as *boyfriend*, *pms* and *jewelry* (Figure 3a). Meanwhile, *married* users were identified through terms such as *dated*, *fiance* and *divorces*, along with obvious terms like *marrying* and *marriages* (Figure 3b). These examples illustrate how the model is able to infer a subject's attribute by aggregating signals across utterances.

In addition to looking at specific utterances, we investigated which terms the model is strongly associating with a specific attribute. To do so, we computed attribute value probabilities for each term in the corpus, and kept the top terms for each attribute value. The results using HAM_{2attn} are shown in Table 7, which is divided into words that appear informative (top section) and words that do not (bottom section). In the case of informative words, there is a clear relationship between the words and the corresponding profession. Many of the uninformative words appear to be movie-specific, such as names (e.g., xavier, leonard) and terms related to a *waiter's* role in a specific movie (e.g., rape, stalkers). Reducing the impact of setting-specific signals like this is one direction for future work.

profession	**significant words**
scientist	characteristics, theory, mathematical, species, changes
politician	governors, senate, secretary, reporters, president
detective	motel, spotted, van, suitcase, parked
military personnel	captured, firepower, guard, soldiers, attack
student	playing, really, emotional, definitely, unbelievable
photographer	xavier, leonard, collins, cockatoo, burke
waiter	rape, stalkers, murdered, overheard, bothering

Table 7: Top-5 words from HAM_{2attn} characterizing each profession.

Models	**profession**		**gender**		**age**	
	MRR micro / macro	AU-ROC	Acc	AU-ROC	MRR micro / macro	AU-ROC
$HAM_{CNN-attn}$	0.19 / 0.18	0.58	0.56	0.58	**0.57** / **0.54**	**0.69**
HAM_{2attn}	**0.21** / **0.21**	**0.67**	**0.61**	**0.64**	0.45 / 0.41	0.45

Table 8: Transfer learning performance of pre-trained Reddit models on MovieChAtt.

6.5 Insights on transfer learning

To investigate the robustness of our trained HAMs, we tested the best performing models (i.e., HAM_{2attn} and $HAM_{CNN-attn}$) on a transfer learning task between our datasets. Specifically, we leveraged user-generated social media text (Reddit) available in abundance for inferring personal attributes of subjects in speech-based

Models	profession		gender		family status	
	MRR micro / macro	AU-ROC	Acc	AU-ROC	Acc	AU-ROC
$HAM_{CNN\text{-}attn}$	0.20 / 0.16	0.58	**0.52**	0.50	**0.74**	**0.74**
HAM_{2attn}	**0.21** / **0.18**	**0.71**	0.51	**0.54**	0.62	0.64

Table 9: Transfer learning performance of pre-trained Reddit models on PersonaChat.

dialogues (MovieChAtt and PersonaChat). We report the results in Table 8 and 9 respectively.

While the scores on PersonaChat are low compared to those in Table 2 and 4, the HAMs' performance on MovieChAtt is often comparable with the baselines' performance in Table 2, 3 and 4. This difference may be caused by the fact that PersonaChat is a smaller, more synthetic dataset as discussed in Section 6.1. On MovieChAtt with the *profession* predicate, both HAMs match the performance of all six baselines in terms of macro MRR. Similarly, HAM_{2attn} matches the performance of five of the six baselines on the *gender* predicate (accuracy), and $HAM_{CNN\text{-}attn}$ matches the performance of four of the six baselines on the *age* predicate (macro MRR). Particularly for the *profession* attribute, missing training subjects in the Reddit dataset for certain professions such as *astronaut* or *monarch* contribute to the decreasing performance, although the models still make a reasonable prediction of *scientist* for *astronaut*. The methods do not perform as well in terms of micro MRR, which may be due to the substantially different attribute value distributions between datasets (i.e., the professions common in movies are uncommon in Reddit). Improving the HAMs' transfer learning performance is a direction for future work.

6.6 Profession misclassification study

In this section we investigate common misclassifications on the MovieChAtt dataset for the *profession* predicate, which is both the most challenging predicate and the predicate with the most possible object values (i.e., 43 on MovieChAtt). A confusion matrix for HAM_{2attn} is shown in Figure 4. Dotted lines indicate several interesting misclassifications: *policemen* are often confused with *detectives* and *special agents* (red line); *scientists* are confused with *astronauts* (yellow line), because sci-fi films often feature characters who arguably serve in both roles; and a *child* is often labeled as a *student* or a *housewife* (green line) because they sometimes use similar terms (e.g., 'school' is used by both children and students, and 'mommy' is used by both children and housewives). Finally, many occupations are confused with *criminal*, which is the most common profession in MovieChAtt.

To further compare the performance of the model on direct and transfer learning tasks we computed the confusion matrix for HAM_{2attn} trained on the Reddit dataset and using MovieChAtt as the test corpus. Interesting misclassifications include the following: artistic professions (*actor*, *painter*, *musician*, *director*) are often mixed up (red lines); *banker* is confused with *manager* (green lines); *policeman* and *airplane pilot* are confused with *military personnel* (purple lines); and *stewardess* is often confused with *nurse* as they both are related to caring and serving tasks (yellow line).

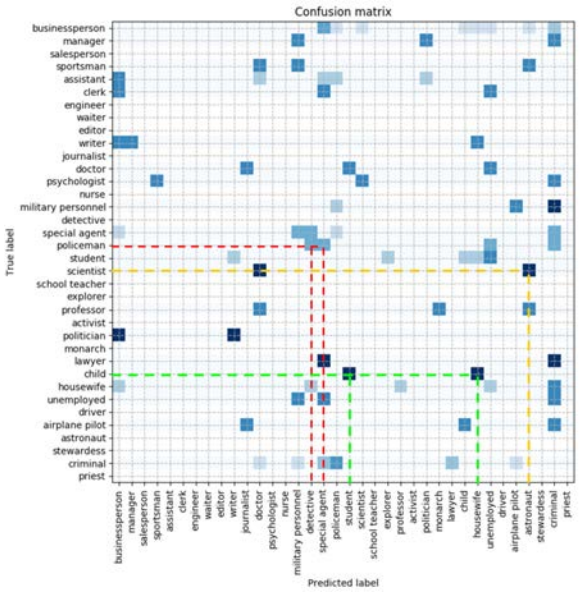


Figure 4: Confusion matrix computed with HAM_{2attn}. True positives are not shown. Darker squares indicate more misclassifications.

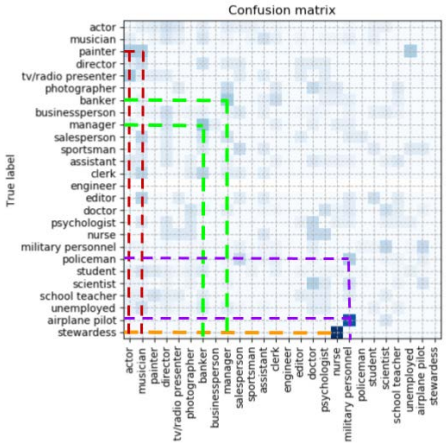


Figure 5: Confusion matrix of $Reddit_{train} \rightarrow MovieChAtt_{test}$ computed with HAM_{2attn}. True positives are not shown. Darker squares indicate more misclassifications.

7 CONCLUSION

We proposed Hidden Attribute Models (HAMs) for inferring personal attributes from conversations, such as a person's profession. We demonstrated the viability of our approach in extensive experiments considering several attributes on three datasets with diverse characteristics: Reddit discussions, movie script dialogues and crowdsourced conversations. We compared HAMs against a variety of state-of-the-art baselines, and achieved substantial improvements over all of them, most notably by the $HAM_{CNN\text{-}attn}$ and HAM_{2attn} variants. We also demonstrated that the attention weights assigned by our methods provide informative explanations of the computed output labels.

As a stress test for our methods, we investigated transfer learning by training HAMs on one dataset and applying the learned models to other datasets. Although we observed degradation in output quality, compared to training on in-domain data, it is noteworthy that the transferred HAMs matched the MovieChAtt performance of the baselines when trained on in-domain data. We plan to further investigate this theme of transfer learning in our future work on construction personal knowledge bases and leveraging them for personalized Web agents.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR'15*.

[2] Angelo Basile, Gareth Dwyer, Maria Medvedeva, Josine Rawee, Hessel Haagsma, and Malvina Nissim. 2017. N-GrAM: New Groningen Author-profiling Model—Notebook for PAN at CLEF 2017. In *Working Notes Papers of the CLEF 2017 Evaluation Labs*.

[3] Roy Khristopher Bayot and Teresa Gonçalves. 2018. Age and Gender Classification of Tweets Using Convolutional Neural Networks. In *Machine Learning, Optimization, and Big Data*. Springer International Publishing, Cham.

[4] John D. Burger, John Henderson, George Kim, and Guido Zarrella. 2011. Discriminating Gender on Twitter. In *Proceedings of EMNLP'11*.

[5] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. *One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling*. Technical Report. Google.

[6] Yun-Nung Chen, Dilek Hakkani-TÃijr, Gokhan Tur, Jianfeng Gao, and Li Deng. 2016. End-to-End Memory Networks with Knowledge Carryover for Multi-Turn Spoken Language Understanding. In *Proceedings of Interspeech'16*. https://doi.org/10.21437/Interspeech.2016-312

[7] Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs.. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011*.

[8] Benjamin Fabian, Annika Baumann, and Marian Keil. 2015. Privacy on Reddit? Towards Large-scale User Classification. In *Proceedings of ECIS'15*.

[9] Lucie Flekova, Jordan Carpenter, Salvatore Giorgi, Lyle Ungar, and Daniel Preoţiuc-Pietro. 2016. Analyzing Biases in Human Perception of User Age and Gender from Text. In *Proceedings of ACL'16 (Volume 1: Long Papers)*.

[10] Lucie Flekova, Daniel Preoţiuc-Pietro, and Lyle Ungar. 2016. Exploring Stylistic Variation with Age and Income on Twitter. In *Proceedings of ACL'16 (Volume 2: Short Papers)*.

[11] Francisco Manuel, Rangel Pardo, Paolo Rosso, Martin Potthast, and Benno Stein. 2017. Overview of the 5th Author Profiling Task at PAN 2017: Gender and Language Variety Identification in Twitter. In *Working Notes Papers of the CLEF 2017 Evaluation Labs*.

[12] Nikesh Garera and David Yarowsky. 2009. Modeling Latent Biographic Attributes in Conversational Genres. In *Proceedings of ACL/IJCNLP'09*.

[13] Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen tau Yih, and Michel Galley. 2018. A Knowledge-Grounded Neural Conversation Model. In *Proceedings of AAAI'18*.

[14] Matej Gjurković and Jan Šnajder. 2018. Reddit: A Gold Mine for Personality Prediction. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media, NAACL-HLT'18*.

[15] Hongyan Jing, Nanda Kambhatla, and Salim Roukos. 2007. Extracting Social Networks and Biographical Facts From Conversational Speech Transcripts. In *Proceedings of ACL'07*.

[16] Chaitanya K. Joshi, Fei Mi, and Boi Faltings. 2017. Personalization in Goal-Oriented Dialog. In *Proceedings of Conversational AI Workshop, NIPS'17*.

[17] Sunghwan Mac Kim, Qiongkai Xu, Lizhen Qu, Stephen Wan, and Cecile Paris. 2017. Demographic Inference on Twitter using Recursive Neural Networks. In *Proceedings of ACL'17 (Volume 2: Short Papers)*.

[18] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of EMNLP'14*.

[19] Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR'15*.

[20] Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A Persona-Based Neural Conversation Model. In *Proceedings of ACL'16 (Volume 1: Long Papers)*.

[21] Xiang Li, Gökhan Tür, Dilek Z. Hakkani-Tür, and Qi Li. 2014. Personal knowledge graph population from user utterances in conversational understanding. In *Proceedings of IEEE Spoken Language Technology Workshop (SLT)*.

[22] Grace I. Lin and Marilyn A. Walker. 2011. All the World's a Stage: Learning Character Models from Film. In *Proceedings of AIIDE'11*.

[23] Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2018. Mem2Seq: Effectively Incorporating Knowledge Bases into End-to-End Task-Oriented Dialog Systems. In *Proceedings of ACL'18 (Volume 1: Long Papers)*.

[24] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of NIPS'13*.

[25] Kaixiang Mo, Yu Zhang, Shuangyin Li, Jiajun Li, and Qiang Yang. 2018. Personalizing a Dialogue System With Transfer Reinforcement Learning. In *Proceedings of AAAI'18*.

[26] Marco Pennacchiotti and Ana-Maria Popescu. 2011. A Machine Learning Approach to Twitter User Classification. In *Proceedings of ICWSM'11*.

[27] James W Pennebaker, Martha E Francis, and Roger J Booth. 2001. Linguistic inquiry and word count: LIWC 2001. *Mahway: Lawrence Erlbaum Associates* 71 (2001).

[28] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of EMNLP'14*.

[29] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of NAACL'18, Volume 1 (Long Papers)*.

[30] Martin Potthast, Francisco Rangel, Michael Tschuggnall, Efstathios Stamatatos, Paolo Rosso, and Benno Stein. 2017. Overview of PAN'17: Author Identification, Author Profiling, and Author Obfuscation. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction. 7th International Conference of the CLEF Initiative (CLEF 17)*. Berlin Heidelberg New York.

[31] Daniel Preoţiuc-Pietro, Vasileios Lampos, and Nikolaos Aletras. 2015. An analysis of the user occupational class through Twitter content. In *Proceedings of ACL/IJCNLP'15 (Volume 1: Long Papers)*.

[32] Daniel Preoţiuc-Pietro, Ye Liu, Daniel Hopkins, and Lyle Ungar. 2017. Beyond Binary Labels: Political Ideology Prediction of Twitter Users. In *Proceedings of ACL'17 (Volume 1: Long Papers)*.

[33] Daniel Preoţiuc-Pietro and Lyle Ungar. 2018. User-Level Race and Ethnicity Predictors from Twitter Text. In *Proceedings of COLING'18*.

[34] Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. 2010. Classifying Latent User Attributes in Twitter. In *Proceedings of SMUC'10*.

[35] Maarten Sap, Gregory Park, Johannes Eichstaedt, Margaret Kern, David Stillwell, Michal Kosinski, Lyle Ungar, and Hansen Andrew Schwartz. 2014. Developing Age and Gender Predictive Lexica over Social Media. In *Proceedings EMNLP'14*.

[36] H. Andrew Schwartz, Johannes C. Eichstaedt, Margaret L. Kern, Lukasz Dziurzynski, Stephanie Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin E. P. Seligman, and Lyle H. Ungar. 2013. Personality, Gender, and Age in the Language of Social Media: The Open-Vocabulary Approach. In *PloS one*.

[37] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. In *Proceedings of NAACL-HLT'15*.

[38] Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep Semantic Role Labeling With Self-Attention. In *Proceedings of AAAI'18*.

[39] Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. 2018. Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures. In *Proceedings of EMNLP'18*.

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[41] Prashanth Vijayaraghavan, Soroush Vosoughi, and Deb Roy. 2017. Twitter Demographic Classification Using Deep Multi-modal Multi-task Learning. In *Proceedings of ACL'17 (Volume 2: Short Papers)*.

[42] Kei Wakabayashi, Johane Takeuchi, Kotaro Funakoshi, and Mikio Nakano. 2016. Nonparametric Bayesian Models for Spoken Language Understanding. In *Proceedings of EMNLP'16*.

[43] Chen Xing, Yu Wu, Wei Wu, Yalou Huang, and Ming Zhou. 2018. Hierarchical Recurrent Attention Network for Response Generation. In *Proceedings of AAAI'18*.

[44] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical Attention Networks for Document Classification. In *Proceedings of NAACL-HLT'16*. https://doi.org/10.18653/v1/N16-1174

[45] Kaisheng Yao, Baolin Peng, Geoffrey Zweig, and Kam-Fai Wong. 2016. An Attentional Neural Conversation Model with Improved Specificity. *CoRR* abs/1606.01292 (2016).

[46] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *TACL* 4 (2016).

[47] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing Dialogue Agents: I have a dog, do you have pets too?. In *Proceedings of ACL'18 (Volume 1: Long Papers)*.

[48] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. In *Proceedings of ACL'16 (Volume 2: Short Papers)*.

Dynamic Deep Multi-modal Fusion for Image Privacy Prediction

Ashwini Tonge
Department of Computer Science
Kansas State University
atonge@ksu.edu

Cornelia Caragea
Department of Computer Science
University of Illinois at Chicago
cornelia@uic.edu

ABSTRACT

With millions of images that are shared online on social networking sites, effective methods for image privacy prediction are highly needed. In this paper, we propose an approach for fusing object, scene context, and image tags modalities derived from convolutional neural networks for accurately predicting the privacy of images shared online. Specifically, our approach identifies the set of most competent modalities on the fly, according to each new target image whose privacy has to be predicted. The approach considers three stages to predict the privacy of a target image, wherein we first identify the neighborhood images that are visually similar and/or have similar sensitive content as the target image. Then, we estimate the competence of the modalities based on the neighborhood images. Finally, we fuse the decisions of the most competent modalities and predict the privacy label for the target image. Experimental results show that our approach predicts the sensitive (or private) content more accurately than the models trained on individual modalities (object, scene, and tags) and prior privacy prediction works. Also, our approach outperforms strong baselines, that train meta-classifiers to obtain an optimal combination of modalities.

CCS CONCEPTS

• **Security and privacy → Social network security and privacy**.

KEYWORDS

Image privacy prediction; fusion of modalities; decision-level fusion

ACM Reference Format:
Ashwini Tonge and Cornelia Caragea. 2019. Dynamic Deep Multi-modal Fusion for Image Privacy Prediction. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313691

1 INTRODUCTION

Technology today offers innovative ways to share photos with people all around the world, making online photo sharing an incredibly popular activity for Internet users. These users document daily details about their whereabouts through images and also post pictures of their significant milestones and private events, e.g., family photos and cocktail parties [28]. Furthermore, smartphones and other mobile devices facilitate the exchange of information in content sharing sites virtually at any time, in any place. Although current social networking sites allow users to change their privacy preferences, this is often a cumbersome task for the vast majority of users on the Web, who face difficulties in assigning and managing privacy settings [30]. Even though users change their privacy settings to comply with their personal privacy preference, they often misjudge the private information in images, which fails to enforce their own privacy preferences [35]. Thus, new privacy concerns [13] are on the rise and mostly emerge due to users' lack of understanding that semantically rich images may reveal sensitive information [1, 35, 47, 57]. For example, a seemingly harmless photo of a birthday party may unintentionally reveal sensitive information about a person's location, personal habits, and friends. Along these lines, Gross and Acquisti [17] analyzed more than 4,000 Carnegie Mellon University students' Facebook profiles and outlined potential threats to privacy. The authors found that users often provide personal information generously on social networking sites, but they rarely change default privacy settings, which could jeopardize their privacy. Employers often perform background checks for their future employees using social networking sites and about 8% of companies have already fired employees due to their inappropriate social media content [54]. A study carried out by the Pew Research center reported that 11% of the users of social networking sites regret the content they posted [33].

Motivated by the fact that increasingly online users' privacy is routinely compromised by using social and content sharing applications [58], recently, researchers started to explore machine learning and deep learning models to automatically identify private or sensitive content in images [35, 45, 49–52, 57]. Starting from the premise that the objects and scene contexts present in images impact images' privacy, many of these studies used objects, scenes, and user tags, or their combination (i.e., feature-level or decision-level fusion) to infer adequate privacy classification for online images.

However, we conjecture that simply combining objects, scenes and user tags modalities using feature-level fusion (i.e., concatenation of all object, scene and user tag features) or decision-level fusion (i.e., aggregation of decisions from classifiers trained on objects, scenes and tags) may not always help to identify the sensitive content of images. Figure 1 illustrates this phenomenon through several images. For example, let us consider image (a) in the figure. Both feature-level and decision-level fusion models yield very low private class probabilities (Feature-level fusion: 0.21 and decision-level fusion: 0.33). Interestingly, a model based on the scene context (bedroom) of the image outputs a high probability of 0.62, showing that the scene based model is competent to capture the sensitive content of the image on its own. Similarly, for the image (b) (self-portrait) in Figure 1, where scene context is seldom in the visual content, the objects in the image (the "persons," "cocktail dress") are more useful (0.78) to predict appropriate image's privacy. Moreover, for images such as "personal documents" (image (c)), user-annotated

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313691

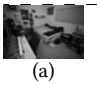
Single modality is correct				Multiple modalities are correct			
Image	Tags	Probabilities		Image	Tags	Probabilities	
		Base classifiers	Fusion			Base classifiers	Fusion
 (a)	bed, studio dining table speakers, music	**scene: 0.62** object: 0.5 tags: 0.29	Feature-level: 0.21 Decision-level: 0.33	 (d)	girl, baby indoor, people canon	scene: 0.49 **object: 0.87** **tags: 0.97**	**Feature-level: 0.77** **Decision-level: 0.67**
 (b)	birthday night party, life	scene: 0.57 **object: 0.78** tags: 0.39	Feature-level: 0.21 Decision-level: 0.33	 (e)	people, party awesome, tea bed, blanket	**scene: 0.92** object: 0.38 **tags: 0.7**	**Feature-level: 0.69** **Decision-level: 0.67**
 (c)	toeic, native speaker, text document, pen	scene: 0.02 object: 0.15 **tags: 0.86**	Feature-level: 0.27 Decision-level: 0.33	 (f)	indoor, fun party people	**scene: 0.92** **object: 0.73** **tags: 0.77**	**Feature-level: 0.89** **Decision-level: 1**

Figure 1: Anecdotal evidence of private images and their tags. The feature-level fusion is given as the concatenation of all the features (object, scene, tag) and the decision-level fusion is obtained by averaging the predictions.

tags provide broader context (such as type and purpose of the document), capturing the sensitive content (0.86), that objects and scene obtained through images' content failed to capture. On the other hand, in some cases, we can find more than one competent model for an image (e.g., for image (d)). To this end, we propose a novel approach that dynamically fuses multi-modal information of online images, derived through Convolutional Neural Networks (CNNs), to adequately identify the sensitive image content. In summary, we make the following contributions:

- Our significant contribution is to estimate the competence of object, scene and tag modalities for privacy prediction and dynamically identify the most competent modalities for a target image whose privacy has to be predicted.
- We derive "competence" features from the neighborhood regions of a target image and learn classifiers on them to identify whether a modality is competent to accurately predict the privacy of the target image. To derive these features, we consider privacy and visual neighborhoods of the target image to bring both sensitive and visually similar image content closer.
- We provide an in-depth analysis of our algorithm in an ablation setting, where we record the performance of the proposed approach by removing its various components. The analysis outline the crucial components of our approach.
- Our results show that we identify images' sensitive content more accurately than single modality models (object, scene, and tag), multi-modality baselines and prior approaches of privacy prediction, depicting that the approach optimally combines the multi-modality for privacy prediction.

2 RELATED WORK

We briefly review the related work as follows.

Ensemble models and Multi-Modality. Several works used ensemble classifiers (or bagging) to improve image classifications [11, 31, 39]. Bagging is an ensemble technique that builds a set of diverse classifiers, each trained on a random sample of the training data to improve the final (aggregated) classifier confidence [4, 43]. Dynamic ensembles that extend bagging have also been proposed [7, 9, 10] wherein a pool of classifiers are trained on a single feature set (single modality) using the bagging technique [4, 43], and the competence of the base classifiers is determined dynamically.

Ensemble classifiers are also used in the multi-modal setting [18, 38], where different modalities have been coupled, e.g., images and text for image retrieval [23] and image classification [18], and audio and visual signals for speech classification [34]. Zahavy et al. [56] highlighted that classifiers trained on different modalities can vary in their discriminative ability and urged the development of optimal unification methods to combine different classifiers. Besides, merging the Convolutional Neural Network (CNN) architectures corresponding to various modalities, that can vary in depth, width, and the optimization algorithm can become very complex. However, there is a potential to improve the performance through multi-modal information fusion, which intrigued various researchers [16, 22, 23, 32]. For example, Frome et al. [14] merged an image network [25] with a Skip-gram Language Model to improve classification on ImageNet. Zahavy et al. [56] proposed a policy network for multi-modal product classification in e-commerce using text and visual content, which learns to choose between the input signals. Feichtenhofer et al. [12] fused CNNs both spatially and temporally for activity recognition in videos to take advantage of the spatio-temporal information present in videos. Wang et al. [53] designed an architecture to combine object networks and scene networks, which extract useful information such as objects and scene contexts for event understanding. Co-training approaches [3] use multiple views (or modalities) to "guide" different classifiers in the learning process. However, co-training methods are semi-supervised and assume that all views are "sufficient" for learning. In contrast with the above approaches, we aim to capture different aspects of images, obtained from multiple modalities (object, scene, and tags), with each modality having a different competence power, and perform dynamic multi-modal fusion for image privacy prediction.

Online Image Privacy. Several works are carried out to study users' privacy concerns in social networks, privacy decisions about sharing resources, and the risk associated with them [15, 19, 24, 36, 37, 42]. Ahern et al. [1] examined privacy decisions and considerations in mobile and online photo sharing. They explored critical aspects of privacy such as users' consideration for privacy decisions, content and context based patterns of privacy decisions, how

different users adjust their privacy decisions and user behavior towards personal information disclosure. The authors concluded that applications, which could support and influence user's privacy decision-making process should be developed. Jones and O'Neill [21] reinforced the role of privacy-relevant image concepts. For instance, they determined that people are more reluctant to share photos capturing social relationships than photos taken for functional purposes; certain settings such as work, bars, concerts cause users to share less. Besmer and Lipford [2] mentioned that users want to regain control over their shared content, but meanwhile, they feel that configuring proper privacy settings for each image is a burden. Buschek et al. [6] presented an approach to assign privacy to shared images using metadata (location, time, shot details) and visual features (faces, colors, edges). Zerr et al. [57] developed the PicAlert dataset, containing Flickr photos, to help detect private images and also proposed a privacy-aware image classification approach to learn classifiers on these Flickr photos. Authors considered image tags and visual features such as color histograms, faces, edge-direction coherence, and SIFT for the privacy classification task. Squicciarini et al. [45, 46] found that SIFT and image tags work best for predicting sensitivity of user's images. Given the recent success of CNNs, Tran et al. [52], and Tonge and Caragea [49, 50] showed promising privacy predictions compared with visual features such as SIFT and GIST. Yu et al. [55] adopted CNNs to achieve semantic image segmentation and also learned object-privacy relatedness to identify privacy-sensitive objects.

Spyromitros-Xioufis et al. [44] used features extracted from CNNs to provide personalized image privacy classification, whereas Zhong et al. [59] proposed a Group-Based Personalized Model for image privacy classification in online social media sites. Despite that an individual's sharing behavior is unique, Zhong et al. [59] argued that personalized models generally require large amounts of user data to learn reliable models, and are time and space consuming to train and to store models for each user, while taking into account possible deviations due to sudden changes of users' sharing activities and privacy preferences. Orekondy et al. [35] defined a set of privacy attributes, which were first predicted from the image content and then used these attributes in combination with users preferences to estimate personalized privacy risk. The authors used official online social network rules to define the set of attributes, instead of collecting real user's opinions about sensitive content and hence, the definition of sensitive content may not meet a user's actual needs [29]. Additionally, for privacy attribute prediction, the authors fine-tuned a CNN pre-trained on object dataset. In contrast, we proposed a dynamic multi-modal fusion approach to determine which aspects of images (objects, scenes or tags) are more competent to predict images' privacy.

3 MULTI-MODALITY

The sensitive content of an image can be perceived by the presence of one or more objects, the scenes described by the visual content and the description associated with it in the form of tags [45, 49–51]. We derive features (object, scene, tags) corresponding to the multi-modal information of online images as follows.

Object (F^o): Detecting objects from images is clearly fundamental to assessing whether an image is of private nature. For example, a single element such as a firearm, political signs, may be a strong indicator of a private image. Hence, we explore the image descriptions extracted from VGG-16 [41], a CNN pre-trained on the ImageNet dataset [40] that has 1.2M+ images labeled with 1,000 object categories. The VGG-16 network implements a 16 layer deep network; a stack of convolutional layers with a very small receptive field: 3×3 followed by fully-connected layers. The architecture contains 13 convolutional layers and 3 fully-connected layers. The input to the network is a fixed-size 224×224 RGB image. The activation of the fully connected layers capture the complete object contained in the region of interest. Hence, we use the activation of the last fully-connected layer of VGG-16, i.e., fc_8 as a feature vector. The dimension of object features F^o is 1000.

Scene (F^s): As consistently shown in various user-centered studies [1], the context of an image is a potentially strong indicator of what type of message or event users are trying to share online. These scenes, e.g., some nudity, home, fashion events, concerts are also often linked with certain privacy preferences. Similar to object features, we obtain the scene descriptors derived from the last fully-connected layer of the pre-trained VGG-16 [25] on the Places2 dataset which contains 365 scene classes within 2.5 million images [60]. The dimension of scene features F^s is 365.

Image Tags (F^t): For image tags, we employ the CNN architecture of Collobert et al. [8]. The network contains one convolution layer on top of word vectors obtained from an unsupervised neural language model. The first layer embeds words into the word vectors pre-trained by Le and Mikolov [27] on 100 billion words of Google News, and are publicly available. The next layer performs convolutions on the embedded word vectors using multiple filter sizes of 3, 4 and 5, where we use 128 filters from each size and produce a tag feature representation. A max-pooling operation over the feature map is applied to capture the most important feature of length 256 for each feature map. To derive these features, we consider two types of tags: (1) user tags, and (2) deep tags. Because not all images on social networking sites have user tags or the set of user tags is very sparse [48], we predict the top d object categories (or deep tags) from the probability distribution extracted from CNN.

Object + Scene + Tag (F^{ost}): We use the combination of the object, scene, and tag features to identify the neighborhood of a target image. We explore various ways given in [12] to combine the features. For example, we use fc_7 layer of VGG to extract features of equal length of 4096 from both object-net and scene-net and consider the max-pooling of these vectors to combine these features. Note that, in this work, we only describe the combination of features that worked best for the approach. We obtain high-level object "F^o" and scene "F^s" features from fc_8 layer of object-net and scene-net respectively and concatenate them with the tag features as follows: $F^{ost} = f^{cat}(F^o, F^s, F^t)$. $F^{ost} = F^o(i), 1 \leq i \leq 1000, F^{ost}(i + 1000) = F^s(i), 1 \leq i \leq 365, F^{ost}(i + 1365) = F^t(i), 1 \leq i \leq 256$.

4 PROPOSED APPROACH

We seek to classify a given image into one of the two classes: *private* or *public*, based on users' general privacy preferences. To achieve this, we depart from previous works that use the same model on all image types (e.g., portraits, bedrooms, and legal documents), and

propose an approach called "Dynamic Multi-Modal Fusion for Privacy Prediction" (or DMFP), that effectively fuses multi-modalities (object, scene, and tags) and dynamically captures different aspects or particularities from image. Specifically, the proposed approach aims to estimate the competence of models trained on these individual modalities for each target image (whose privacy has to be predicted) and dynamically identifies the subset of the most "competent" models for that image. The rationale for the proposed method is that for a particular type of sensitive content, some modalities may be important, whereas others may be irrelevant and may simply introduce noise. Instead, a smaller subset of modalities may be significant in capturing a particular type of sensitive content (e.g., objects for portraits, scenes for interior home or bedroom, and tags for legal documents, as shown in Figure 1).

The proposed approach considers three stages to predict the privacy of a target image, wherein we first identify the neighborhood images that are visually similar and/or have similar sensitive content as the target image (Section 4.1). Then, using a set of base classifiers, each trained on an individual modality, we estimate the competence of the modalities by determining which modalities classify the neighborhood images correctly (Section 4.2). The goal here is to select the most competent modalities for a particular type of images (e.g., scene for home images). Finally, we fuse the decisions of the most competent base classifiers (corresponding to the most competent modalities) and predict the privacy label for the target image (Section 4.3).

Our approach considers two datasets, denoted as $\mathcal{D}^T$ and $\mathcal{D}^E$, that contain images labeled as *private* or *public*. We use the dataset $\mathcal{D}^T$ to train a base classifier for each modality to predict whether an image is public or private. Particularly, we train 3 base classifiers $\mathcal{B} = \{B^o, B^s, B^t\}$ on the corresponding modality feature sets from $\mathcal{F}$. Note that we use the combination of feature sets F^{ost} only for visual content similarity and do not train a base classifier on it. The competences of these base classifiers are estimated on the $\mathcal{D}^E$ dataset. We explain the stages of the proposed approach as follows. The notation used is shown in Table 1.

4.1 Identification of Neighborhoods

The competence of a base classifier is estimated based on a local region where the target image is located. Thus, given a target image T, we first estimate two neighborhoods for T: (1) visual similarity based (N_V^T) and (2) privacy profile based (N_P^T) neighborhoods.

The neighborhood N_V^T of target image T consists of the k_v most similar images from $\mathcal{D}^E$ using visual content similarity. Specifically, using the F^{ost} features obtained by concatenating object, scene, and tag features (as explained in Section 3), we determine the k_v most visually similar images to T by applying the K-Nearest Neighbors algorithm on the $\mathcal{D}^E$ dataset.

The neighborhood N_P^T of target image T consists of k_p most similar images to T by calculating the cosine similarity between the privacy profile of T and images from the dataset $\mathcal{D}^E$. We define privacy profile (denoted by $\overline{T}$) of image T as a vector of posterior privacy probabilities obtained by the base classifiers $\mathcal{B}$ i.e., $\overline{T} = \bigcup_{B_i \in \mathcal{B}} \{P(Y_T = private|T, B_i), P(Y_T = public|T, B_i)\}$. For image (a) in Figure 1, $\overline{T} = [0.62, 0.38, 0.5, 0.5, 0.29, 0.71]$. We consider the privacy profile of images because images of particular image content (bedroom images or legal documents) tend to possess similar privacy probabilities with respect to the set of base classifiers $\mathcal{B}$. For instance, irrespective of various kinds of bedroom images, the probabilities for a private class obtained by base classifiers $\mathcal{B}$, would be similar. This enables us to bring sensitive content closer irrespective of their disparate visual content. Moreover, we consider two different numbers of nearest neighbors k_v and k_p to find the neighborhoods since the competence of a base classifier is dependent on the neighborhood and estimating an appropriate number of neighbors for the respective neighborhoods reduces the noise.

Table 1: Mathematical notations.

Notation	Description
$\mathcal{D}^T$	$= \{(I_1, Y_1), \cdots, (I_m, Y_m)\}$ a dataset of labeled images for base classifier training.
$\mathcal{D}^E$	$= \{(X_1, Y_1), \cdots, (X_n, Y_n)\}$ a dataset of labeled images for competence estimation.
T	A target image with an unknown privacy label.
$\mathcal{F}$	$= \{F^o, F^s, F^t, F^{ost}\}$ a collection of modality feature sets of object, scene, tag, and object+scene+tag, respectively.
$\mathcal{B}$	$= \{B^o, B^s, B^t\}$ a set of base classifiers trained on corresponding modality feature sets from $\mathcal{F}$ (e.g., B^o is trained on F^o).
N_V^T	The visual similarity based neighborhood of image T estimated using visual content features F^{ost}, i.e., the set of most similar images to T based on visual content.
k_v	The size of N_V^T, where $1 \leq k_v < n$.
N_P^T	The privacy profile based neighborhood of target T, i.e., the set of most similar images to T based on images' privacy profiles.
k_p	The size of N_P^T, where $1 \leq k_p < n$.
C	$= \{C^o, C^s, C^t\}$ a set of "competence" classifiers corresponding to the base classifiers from $\mathcal{B}$ (e.g., C^o for B^o).
Φ	$= \{\phi^o, \phi^s, \phi^t\}$ a set of "competence" feature vectors for training the "competence" classifiers.

4.2 "Competence" Estimation

We now describe how we estimate the "competence" of a base classifier. For instance, for the image (a) in Figure 1, scene model has a higher competence than the others, and here, we capture this competence through "competence" features and "competence" classifiers. Specifically, we train a competence classifier for each base classifier that predicts if the base classifier is competent or not for a target image T. The features for learning the competence classifiers and the competence learning are described below.

4.2.1 Derivation of "Competence" Features. We define three different sets of "competence" features wherein each set of these features captures a different criterion to estimate the level of competence of base classifiers dynamically. The first competence feature ϕ_1, for image T, is derived from the neighborhood N_V^T (based on visual similarity) whereas the second competence feature ϕ_2 is obtained from the neighborhood N_P^T (based on privacy profile). The third competence feature ϕ_3 captures the level of confidence of base classifiers for predicting the privacy of the image (T) itself. We create a "competence" feature vector by concatenating all these competence features $\phi = \{\phi_1 \cup \phi_2 \cup \phi_3\}$ into a vector of length $|\phi| = k_v + k_p + 1$. We extract such competence vectors corresponding to each base

Algorithm 1 The "Competence" Learning

1: **Input**: A dataset $\mathcal{D}^E = \{(X_1, Y_1), \cdots, (X_n, Y_n)\}$ of labeled images; $F^{ost}_{X_j}$ combination of modality feature sets for X_j; a set of base classifiers $\mathcal{B} = \{B^o, B^s, B^t\}$.
2: **Output**: A set of "competence" classifiers $C = \{C^o, C^s, C^t\}$.
3: $\mathcal{D} = \{D^o, D^s, D^t\} \leftarrow \varnothing$; // Datasets for training competence classifiers, initially empty.
4: $C \leftarrow \varnothing$; // A set of competence classifiers, initially empty.
5: **for all** $X_j \in \mathcal{D}^E$ **do**
6: $N_V^{X_j} \leftarrow IdentifyVisualNeighborhood(k_v, X_j, F^{ost}_{X_j}, \mathcal{D}^E)$; // k_v nearest neighbors of X_j obtained using visual content similarity.
7: $\overline{X_j} \leftarrow ComputePrivacyProfile(X_j, \mathcal{B})$; //Privacy profile.
8: $N_P^{X_j} = IdentifyPrivacyNeighborhood(k_p, \overline{X_j}, \overline{\mathcal{D}^E})$; // k_p most similar images of $\overline{X_j}$ obtained using privacy profile similarity.
9: **for all** $B_i \in \mathcal{B}$ **do** // Iterate through the set of base classifiers.
10: $\phi_{i,j} \leftarrow CompetenceFeatures(X_j, N_V^{X_j}, N_P^{X_j}, B_i)$;
11: **if** $Predict(B_i, X_j) = Y_j$ **then** // predicted correctly.
12: $L_{i,j} \leftarrow 1$; // B_i is competent for X_j.
13: **else**
14: $L_{i,j} \leftarrow 0$; // B_i is not competent for X_j.
15: **end if**
16: $D^i \leftarrow D^i \cup \{(\phi_{i,j}, L_{i,j})\}$
17: **end for**
18: **end for**
19: **for all** $D^i \in \mathcal{D}$ **do** // Train competence classifiers.
20: $C_i \leftarrow TrainCompetenceClassifier(D^i)$;
21: $C \leftarrow C \cup C_i$
22: **end for**
23: **return** C;

Algorithm 2 Dynamic Fusion of Multi-Modality

1: **Input**: A target image T; $\mathcal{D}^E = \{(X_1, Y_1), \cdots, (X_n, Y_n)\}$ a dataset of labeled images; F^{ost}_T combination of modality feature sets for T; a set of base classifiers $\mathcal{B} = \{B^o, B^s, B^t\}$; and a set of competence classifiers $C = \{C^o, C^s, C^t\}$.
2: **Output**: Privacy label Y_T.
3: $B' \leftarrow \varnothing$; // the subset of most competent base classifiers.
4: $CS \leftarrow \varnothing$; // the set of competence scores.
5: $T_{ba} \leftarrow Agreement(\mathcal{B}, T)$; // Base classifiers' agreement on T's label.
6: **if** $T_{ba} \leq |\mathcal{B}|$ **then**
7: $N_V^T \leftarrow IdentifyVisualNeighborhood(k_v, T, F^{ost}_T, \mathcal{D}^E)$; // k_v nearest neighbors of T obtained using visual content similarity.
8: $\overline{T} \leftarrow ComputePrivacyProfile(T, \mathcal{B})$; //Privacy profile of T.
9: $N_P^T = IdentifyPrivacyNeighborhood(k_p, \overline{T}, \overline{\mathcal{D}^E})$; // k_p most similar images of $\overline{T}$ obtained using privacy profile similarity.
10: **for all** $B_i \in \mathcal{B}$ & $C_i \in C$ **do** // Iterate through the set of base and competence classifiers.
11: $\phi_i \leftarrow CompetenceFeatures(T, N_V^T, N_P^T, B_i)$;
12: $CS_i \leftarrow PredictCompetence(F_i, C_i)$; // Predict competence score for base classifier B_i.
13: **if** $CS_i > 0.5$ **then** // If the predicted competence score is greater than 0.5 then the base classifier B_i is competent.
14: $B' \leftarrow B' \cup \{B_i\}$
15: $CS \leftarrow CS \cup \{CS_i\}$
16: **end if**
17: **end for**
18: $Y_T = WeightedMajorityVote(T, B', CS)$ // Votes are first weighted by the competence score and then majority vote is taken.
19: **end if**
20: **return** Y_T

classifier in $\mathcal{B}$ (e.g., ϕ^o for B^o, refer to Figure 2). We extract these "competence" features as follows.

ϕ_1: A vector of k_v entries that captures the correctness of a base classifier in the visual neighborhood region N_V^T. An entry j in ϕ_1 is 1 if a base classifier $B_i \in \mathcal{B}$ accurately predicts privacy of image $X_j \in N_V^T$, and is 0 otherwise, where $j = 1, \cdots, k_v$. For the target image in Figure 2, $\phi_1 = \{1, 1, 0, 1, 0, 1, 1\}$, obtained by B^o.

ϕ_2: A vector of k_p entries that captures the correctness of a base classifier in the privacy profile neighborhood region N_P^T. An entry j in ϕ_2 is 1 if a base classifier $B_i \in \mathcal{B}$ accurately predicts privacy of image $X_j \in N_P^T$, and is 0 otherwise, where $j = 1, \cdots, k_p$. For the target image in Figure 2, $\phi_2 = \{1, 1, 1, 1, 1\}$, obtained using B^o.

ϕ_3: We capture a degree of confidence of base classifiers for target image T. Particularly, we consider the maximum posterior probability obtained for target image T using base classifier B_i i.e. $Max(P(Y_T = Private|T, B_i), P(Y_T = Public|T, B_i))$, where $B_i \in \mathcal{B}$. For the target image in Figure 2, $\phi_3 = 0.67$, obtained using B^o.

4.2.2 "Competence" Learning. We learn the "competence" of a base classifier by training a binary "competence" classifier on the dataset $\mathcal{D}^E$ in a Training Phase. A competence classifier predicts whether a base classifier is competent or not for a target image. Algorithm 1 describes the "competence" learning process in details. Mainly, we consider images from $\mathcal{D}^E$ as target images (for the training purpose only) and identify both the neighborhoods (N_V, N_P) from the dataset $\mathcal{D}^E$ itself (Alg. 1, lines 6–8). Then, we extract "competence" features for each base classifier in $\mathcal{B}$ based on the images from these neighborhoods (Alg. 1, line 10). To reduce noise, we extract "competence" features by considering only the images belonging to both the neighborhoods. On these "competence" features, we train a collection of "competence" classifiers C corresponding to each base classifier in $\mathcal{B}$ (Alg. 1, lines 19–22). Precisely, we train 3 competence classifiers $C = \{C^o, C^s, C^t\}$. To train "competence" classifier $C_i \in C$, we consider label $L_i = 1$ if base classifier $B_i \in \mathcal{B}$ predicts the correct privacy of a target image (here, $X_j \in \mathcal{D}^E$), otherwise 0 (Alg. 1, lines 11–16).

4.3 Dynamic Fusion of Multi-Modality

In this stage, for given target image T, we dynamically determine the subset of most competent base classifiers. We formalize the process of base classifier selection in Algorithm 2. The algorithm first checks the agreement on the privacy label between all the base classifiers in $\mathcal{B}$ (Alg. 2, line 5). If not all the base classifiers agree, then we estimate the competence of all the base classifiers and identify the subset of most competent base classifiers for the target image as follows. Given target image T, Algorithm 2 first identifies both the neighborhoods (N_V^T, N_P^T) using the visual features F^{ost}

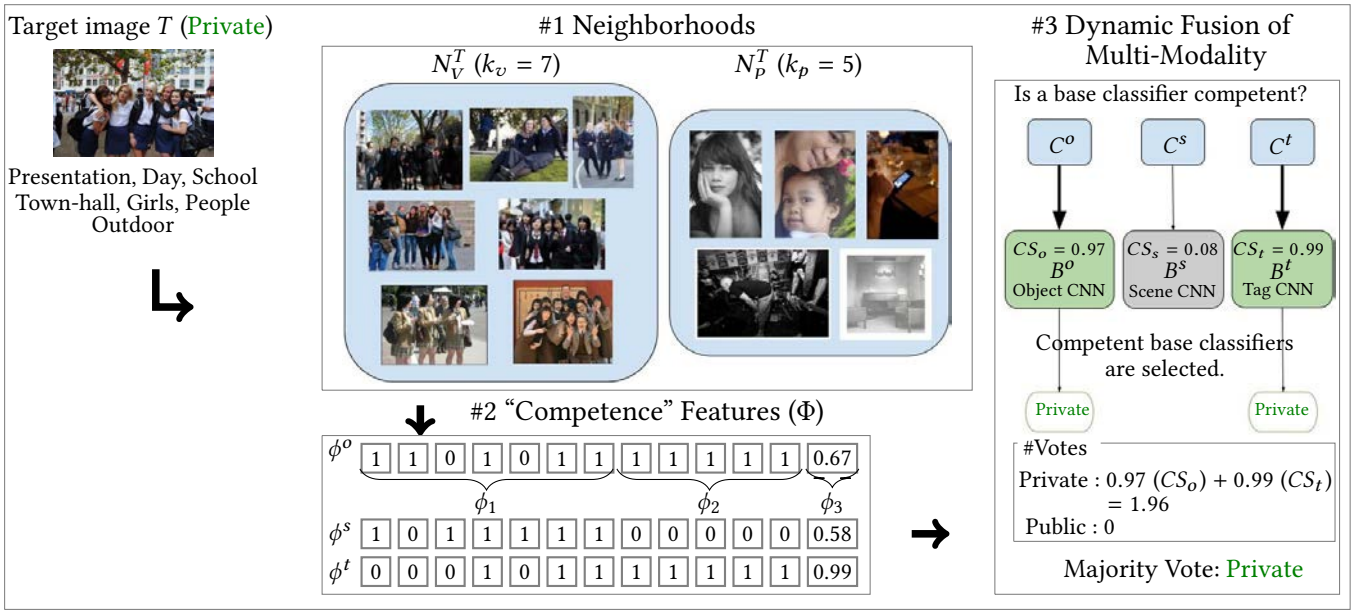


Figure 2: Illustration of the proposed approach using an example.

and privacy profile from $\mathcal{D}^E$ dataset (Alg. 2, lines 7–9). Using these neighborhoods, we extract "competence" feature vectors (explained in Section 4.2) and provide them to the respective "competence" classifiers in C (learned in the Training Phase) to predict competence score of base classifier B_i. If the competence score is greater than 0.5, then base classifier B_i is identified as competent to predict the privacy of target image T (Alg. 2, lines 10–17). Finally, we weight votes of the privacy labels predicted by the subset of most competent base classifiers by their respective "competence" score and take a majority vote to obtain the final privacy label for target image T (Alg. 2, line 18). A "competence" score CS_i is given as a probability of base classifier B_i being competent. We consider the majority vote of the most competent base classifiers because certain images (e.g., vacation) might require more than one base classifiers (object and scene) to predict the appropriate privacy. If both the privacy classes (private and public) get the same number of votes, then the class of a highest posterior probability is selected.

Illustration of the Proposed Approach . Figure 2 shows the illustration of the proposed approach through an anecdotal example. We consider a target image T whose privacy has to be predicted. For T, we first identify two neighborhoods: (1) visual content (N_V^T), 2. privacy profile (N_P^T). For N_V^T, we use visual content features F^{ost} to compute the similarity between target image T and the images from the dataset $\mathcal{D}^E$. The top $k_v = 7$ similar images for N_V^T are shown in the figure (left blue rectangle). Likewise, for N_P^T, we compute the similarity between privacy profile of the target image $\overline{T}$ and privacy profiles of images in $\mathcal{D}^E$. We show the top $k_p = 5$ similar images for N_P^T in the right blue rectangle of Figure 2. From these neighborhoods, we derive a "competence" feature vector ϕ for each base classifier in $\mathcal{B}$ (e.g., ϕ^o for B^o). We show these "competence" features in the figure as a matrix of feature values. We input these features to the respective "competence" classifiers from C (e.g., ϕ^o to C^o), that predict whether a base classifier $B_i \in B$ is competent to predict correct privacy label of the target image (T). The "competence" classifiers (C^o, C^s, C^t) are shown as blue rectangles on the right side of Figure 2. The base classifiers B^o and B^t are predicted as competent and hence are selected to obtain the final privacy label for the target image. The competent base classifiers are shown in green rectangles on the right side of Figure 2. Once we select the competent base classifiers, we take a weighted majority vote on the privacy labels, predicted by these base classifiers. For example, in this case, the competent base classifiers B^o and B^t predict the privacy of T as "private," and hence, the final privacy label of T is selected as "private." It is interesting to note that the target image (T) contains "outdoor" scene context that is not useful to predict the correct privacy label and hence, the scene model B^s is not selected by the proposed approach for the target image.

5 DATASET

We evaluate our approach on a subset of 32,000 Flickr images sampled from the PicAlert dataset, made available by Zerr et al. [57]. PicAlert consists of Flickr images on various subjects, which are manually labeled as *public* or *private* by external viewers. The guideline to select the label is given as: private images belong to the private sphere (like self-portraits, family, friends, someone's home) or contain information that one would not share with everyone else (such as private documents). The remaining images are labeled as public. The dataset of 32,000 images is split in $\mathcal{D}^T$, $\mathcal{D}^E$ and *Test* sets of 15,000, 10,000 and 7,000 images, respectively. Each experiment is repeated 5 times with a different split of the three subsets (obtained using 5 different random seeds) and the results are averaged across the five runs. The public and private images are in the ratio of 3:1 in all subsets.

Table 2: Exploratory analysis.

Test	Pr(%)	Pu(%)	O(%)
Object is correct	49	95.7	84.8
Scene is correct	51	94.7	84.4
Tag is correct	57	91.1	83
All are correct	30	87.3	73.9
All are wrong	27	1.5	7.4
Atleast one modality is correct	**73**	**98.5**	**92.6**

6 EXPERIMENTS AND RESULTS

We evaluate the privacy prediction performance obtained using the proposed approach DMFP, where we train a set of base classifiers $\mathcal{B}$ on images in the dataset $\mathcal{D}^T$, and dynamically estimate the "competence" of these base classifiers for target images in *Test* by identifying neighborhoods (N_V, N_P) using images in $\mathcal{D}^E$. We first consider various values of neighborhood parameters k_v and k_p and show their impact on the performance of the proposed approach. Then, we compare the performance of the proposed approach with the performance obtained using three types of mechanisms: (1) components of the proposed approach, that are used to fuse the multi–modal characteristics of online images, (2) the state-of-the-art approaches for privacy prediction, and (3) strong baselines that select models based on their competence (e.g., Zahavy et al. [56]) and that attempt to yield the optimal combination of base classifiers (for instance, using stacked ensemble classifiers).

Evaluation Setting. We train base classifiers ($\mathcal{B}$) using the Calibrated linear Support Vector Machine (SVM) implemented in Scikit-learn library to predict more accurate probabilities. We use 3-fold cross-validation on the dataset $\mathcal{D}^T$ to fit the linear SVM on the 2–folds, and the remaining fold is used for calibration. The probabilities for each of the folds are then averaged for prediction. We train "competence" classifiers (C) on the dataset $\mathcal{D}^E$ using logistic regression to predict "competence" scores between $0-1$ for base classifiers. If base classifier B_i gets a "competence" score greater than 0.5 then the base classifier is considered competent. To derive features from CNN, we use pre-trained models presented by the VGG-16 team in the ILSVRC-2014 competition [41] and the CAFFE framework [20]. For deep tags, we consider top $d = 10$ object labels as $d = 10$ worked best.

Exploratory Analysis. We provide exploratory analysis in Table 2 to highlight the potential of merging object, scene and tag modality for privacy prediction. We predict privacy for images in the *Test* set using base classifiers in $\mathcal{B}$ and obtain "private" (Pr), "public" (Pu) and "overall" (O) accuracy for: (a) a modality is correct (e.g., object), (b) all modalities are correct, (c) all modalities are wrong, and (d) at least one modality is correct. Table 2 shows that out of the three base classifiers (top 3 rows), the tag model yields the best accuracy for the private class (57%). Interestingly, the results for "at least one modality is correct" (73%) show that using multi-modality, there is a huge potential (16%) to improve the performance of the private class. This large gap is a promising result for developing multi-modality approaches for privacy prediction. Next, we evaluate DMFP that achieved the best boost in the performance for the private class using these modalities.

6.1 Impact of Parameters k_v and k_p on DMFP

We show the impact of neighborhood parameters, i.e., k_v and k_p on the privacy prediction performance obtained by the proposed approach DMFP. k_v and k_p are used to identify visual (N_V) and privacy profile (N_P) neighborhoods of a target image, respectively (Alg. 2 lines 7–8). We experiment with a range of values for both the parameters $k_v, k_p = \{10, 20, \cdots, 100, 200, \cdots, 1000\}$, in steps of 10 upto 100 and then in steps of 100. We also experiment with larger k_v and k_p values, but for better visualization, we only show the values with significant results. Figure 3 shows the F1-measure obtained (using 3-fold cross-validation on the $\mathcal{D}^E$ dataset) for the private class for various k_v and k_p values. We notice that when we increase the k_v parameter the performance increase whereas when we increase k_p parameter, the performance increases upto $k_p = 200$, then the performance decreases gradually. The results show that the performance is quite sensitive to changes in the privacy neighborhood (N_P) parameter k_p, but relatively insensitive to changes in the visual neighborhood (N_V) parameter k_v. We get the best performance for $k_v = 900$ and $k_p = 100$. We use these parameter values in the next experiments.

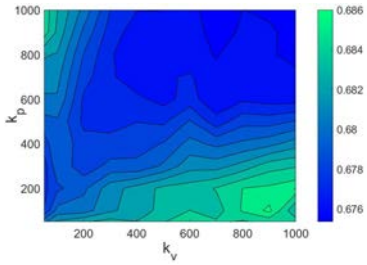


Figure 3: F1-measure obtained for various k_v and k_p values.

6.2 Evaluation of the Proposed Approach

We evaluate the proposed approach DMFP for privacy prediction in an ablation experiment setting. Specifically, we remove a particular component of the proposed approach DMFP and compare the performance of DMFP before and after the removal of that component. We consider excluding several components from DMFP: (1) the visual neighborhood N_V (DMFP$-N_V$), (2) the privacy profile neighborhood N_P (DMFP$-N_P$), (3) "competence" features (e.g., DMFP$-\phi_1$), and (4) base classifier selection without "competence" learning (e.g., $N_V - CL$). For option (4), we consider a simpler version of the proposed algorithm, in which we do not learn a competence classifier for a base classifier; instead, we rely solely on the number of accurate predictions of the samples from a neighborhood. We evaluate it using images from three regions: (a) neighborhood N_V only ($N_V - CL$), (b) neighborhood N_P only ($N_P - CL$), and (c) both the neighborhoods N_P and N_V ($\{N_P + N_V\} - CL$).

Table 3 shows the class-specific (private and public) and overall performance obtained by the proposed approach (DMFP) and after removal of its various components detailed above. Primarily, we wish to identify whether the proposed approach characterizes the private class effectively as sharing private images on the Web with everyone is not desirable. We observe that the proposed approach achieves the highest recall of 0.627 and F1-score of 0.684 (private class), which is better than the performance obtained by eliminating the essential components (e.g., neighborhoods) of the proposed approach. We notice that if we remove either of the neighborhood

Table 3: Evaluation of dynamic multi-modal fusion for privacy prediction (DMFP).

Features	Private			Public			Overall			
	Precision	Recall	F1-score	Precision	Recall	F1-score	Accuracy (%)	Precision	Recall	F1-score
DMFP	0.752	**0.627**	**0.684**	**0.891**	0.936	**0.913**	**86.36**	**0.856**	**0.859**	**0.856**
Components of the proposed approach										
DMFP−N_V	0.763	0.575	0.655	0.879	0.945	0.91	85.79	0.85	0.852	0.847
DMFP−N_P	0.74	0.572	0.645	0.877	0.938	0.907	85.21	0.843	0.847	0.841
$N_V - CL$	**0.79**	0.534	0.637	0.87	**0.956**	0.911	85.71	0.85	0.851	0.843
$N_P - CL$	0.788	0.537	0.639	0.87	**0.956**	0.911	85.73	0.85	0.851	0.843
$\{N_V + N_P\} - CL$	**0.79**	0.534	0.637	0.87	**0.956**	0.911	85.71	0.85	0.851	0.843
"Competence" Features										
DMFP−ϕ_1	0.777	0.553	0.646	0.874	0.951	0.911	85.74	0.849	0.852	0.844
DMFP−ϕ_2	0.74	0.565	0.641	0.875	0.939	0.906	85.11	0.842	0.846	0.84
DMFP−ϕ_3	0.752	0.627	0.683	0.891	0.936	0.913	86.35	0.856	0.859	0.856

N_V or N_P, the recall and F1-score drop by 5% and 4%. This suggests that both the neighborhoods (N_V, N_P) are required to identify an appropriate local region surrounding a target image. It is also interesting to note that the performance of DMFP−N_P (removal of N_P) is slightly lower than the performance of DMFP−N_V (removal of N_V), depicting that neighborhood N_P is helping more to identify the competent base classifier(s) for a target image. The N_P neighborhood brings images closer based on their privacy probabilities and hence, is useful to identify the competent base classifier(s) (this is evident in Figure 2). We also show that when we remove competence learning (CL) i.e., $N_V - CL$, $N_P - CL$, and $\{NV + NP\} - CL$, the precision improves by 4% (private class), but the recall and F1-score (private class) drops by 9% and 5% respectively, showing that competence learning is necessary to achieve the best performance.

We also remove the "competence" features one by one and record the performance of DMFP to understand which competence features are essential. Table 3 shows that when we remove feature ϕ_1 corresponding to the neighborhood N_V, the performance drops significantly ($\approx$ 4%). Likewise, when we remove ϕ_2 (feature corresponding the N_P region), we notice a similar decrease of 4% in the F1-score of private class. Note that, when we remove the "competence" features corresponding to their neighborhoods (such as ϕ_1 for N_V and ϕ_2 for N_P), we get nearly similar performance as we remove the respective neighborhoods from the proposed approach (DMFP−N_V and DMFP−N_P); implying that removing "competence" features (e.g., ϕ_1) is as good as removing the corresponding neighborhood (N_V). However, a close look at the performance suggests that the performance obtained using DMFP−ϕ_1 (recall of 0.553) is slightly worse than the performance of DMFP−N_V (recall of 0.57). Similarly, for DMFP−ϕ_2, the performance (recall) decrease from 0.572 obtained using DMFP−N_P to 0.565. The performance decrease can be explained as when we remove the neighborhood N_V or N_P, the respective "competence" features are empty, and that might be helpful for some cases (as zero-valued ϕ_2 feature of ϕ^s was helpful in Figure 2). Additionally, the recall of DMFP−N_V and DMFP−N_P are similar whereas the recall of DMFP−ϕ_1 (0.553) is slightly worse than the recall of DMFP−ϕ_2 (0.565). The results suggests that the neighborhood N_V is more dependent on the "competence" features as compared to the neighborhood N_P. We experimented with the probability based "competence" features (instead of boolean features), but did not yield improvements in the performance.

Model	 (a)	 (b)	 (c)	 (d)
DMFP	✓	✓	✓	✗
Object	✗	✓	✓	✗
Scene	✓	✗	✓	✗
Tags	✓	✓	✗	✗

Figure 4: Predictions for private images.

6.3 Proposed Approach vs. Base Classifiers

We compare privacy prediction performance obtained by the proposed approach DMFP with the set of base classifiers $\mathcal{B}$: 1. object (B^o), 2. scene (B^s), and 3. image tags (B^t).

Table 4 compares the performance obtained by the proposed approach (DMFP) and base classifiers. We achieve the highest performance as compared to the base classifiers and show a maximum improvement of $\approx$ 10% in the F1-score of private class. We notice that our approach based on multi-modality yields an improvement of 11% over the recall of object and scene models and an improvement of $\approx$ 6% over the recall of the tag model, that is the best-performing single modality model obtained for the private class from the exploratory analysis (refer to Table 2). Still, our approach makes some errors (See Table 2 and 3, 73% vs. 62%). A close look at the errors discovered that a slight subjectivity of annotators could obtain different labels for similar image subjects (e.g., food images are very subjective).

Error Analysis. We perform error analysis to further analyze the results of the proposed approach. We first determine the errors generated by all the base classifiers in $\mathcal{B}$ and corrected by the proposed approach DMFP. We calculate the percentage of corrected errors for private class, public class and overall (considering both the classes) and show them in Table 6. We compute the percentage of corrected errors as the number of corrected errors of private (or public) class over the total number of private (or public) class errors. We calculate the fraction of overall corrected errors by considering both public and private classes. The table shows that we correct 14% − 27% of private class errors, 18% − 58% of public class errors and overall we eliminate 16% − 37% errors. Note that errors generated for the private class are much larger than the public class

Table 4: Dynamic multi-modal fusion for privacy prediction (DMFP) vs. base classifiers of DMFP.

Features	Private			Public			Overall			
	Precision	Recall	F1-score	Precision	Recall	F1-score	Accuracy (%)	Precision	Recall	F1-score
DMFP	0.752	**0.627**	**0.684**	**0.891**	0.936	**0.913**	**86.36**	**0.856**	**0.859**	**0.856**
Object (B^o)	**0.772**	0.513	0.616	0.864	**0.953**	0.907	84.99	0.843	0.85	0.838
Scene (B^s)	0.749	0.51	0.606	0.863	0.947	0.903	84.45	0.836	0.844	0.833
Image tags (B^t)	0.662	0.57	0.612	0.873	0.91	0.891	83.03	0.823	0.83	0.826

Table 5: Dynamic multi-modal fusion for privacy prediction (DMFP) vs. baselines.

Features	Private			Public			Overall			
	Precision	Recall	F1-score	Precision	Recall	F1-score	Accuracy (%)	Precision	Recall	F1-score
DMFP	0.752	**0.627**	**0.684**	**0.891**	0.936	**0.913**	**86.36**	**0.856**	**0.859**	**0.856**
Zahavy et al. [56]	0.662	0.568	0.612	0.873	0.911	0.891	83.02	0.82	0.825	0.821
Majority Vote	**0.79**	0.534	0.637	0.87	**0.956**	0.911	85.71	0.85	0.851	0.843
Decision-level Fusion	0.784	0.555	0.65	0.874	0.953	0.912	85.94	0.852	0.853	0.846
Stacked–En	0.681	0.59	0.632	0.879	0.915	0.897	83.86	0.829	0.834	0.831
Cluster–En	0.748	0.429	0.545	0.845	**0.956**	0.897	83.17	0.822	0.831	0.814

Table 6: Errors corrected (%).

$\mathcal{B}$	overall	private	public
object	16.52	14.00	23.75
scene	27.71	21.78	42.63
Tags	37.02	26.90	58.79

(See Table 4) and hence, even a comparatively smaller percentage of corrected errors constitute to a significant improvement. We also analyze results by showing predictions of samples in Figure 4, for which at least one base classifier fails to predict the correct privacy of an image. For instance, for example (b), scene model failed to predict the correct privacy of the image; however, DMFP identifies the competent base classifiers, i.e., object, and tag and predict the correct privacy label. We also show an example (image (d)) for which all the base classifiers fail to predict the correct privacy class and hence, the proposed approach also fails to predict the correct label. The image of a food is very subjective and hence, generic base classifiers will not be sufficient to predict the correct labels of such images. In the future, these generic models can be extended to develop hybrid approaches, that consider both generic and subjective privacy notions to predict personalized privacy labels.

6.4 Proposed Approach vs. Baselines

We compare the performance of the proposed approach DMFP with multi-modality based baselines described below.

1. Model Selection by Zahavy et al. [56]: The authors proposed a deep multi-modal architecture for product classification in e-commerce, wherein they learn a decision-level fusion policy to choose between image and text CNN for an input product. Specifically, the authors provide class probabilities of a product as input to the policy trained on a validation dataset and use it to predict whether image CNN (or text CNN) should be selected for the input. In other words, policy determines the competence of the CNNs for its input and thus, we consider it as our baseline. For a fair comparison, we learn 3 policies (corresponding to the competence classifiers C), wherein each policy (say object policy) predicts whether the respective base classifier (object) should be selected for a target image. Note that we learn these policies on the $\mathcal{D}^E$ dataset. Finally, we take a majority vote of the privacy label predicted by the selected base classifiers (identified by the policies) for a target image.

2. Majority Vote: We consider a majority vote as another baseline, as we use it for final selection of privacy label for a target image. Unlike our approach, a vote is taken without any pre-selection of base classifiers. We predict privacy of a target image using base classifiers in $\mathcal{B}$ and select a label having highest number of votes.

3. Decision-level Fusion: Fixed rules, that average the predictions of the different CNNs [25] or select the CNN with the highest confidence [38]. The first rule is equivalent to the majority vote baseline, and hence, we show the results for the second rule only. The second rule is given as: $Y_T = argmax_i([{P_i}^o + {P_i}^s + {P_i}^t]/3)$, where $i = 0$ (public), 1 (private). P^o, P^s, and P^t denotes the posterior probabilities (private or public) obtained using object (B^o), scene (B^s) and tag (B^t) modality respectively.

4. Stacked Ensemble (Stacked–en): Stacking learns a meta-classifier to find an optimal combination of the base learners [5, 26]. Unlike bagging and boosting, stacking ensembles robust and diverse set of base classifiers together, and hence, we consider it as one of the baselines. We use the same set of base classifiers $\mathcal{B}$ to encode images in $\mathcal{D}^T$ using posterior probabilities $P(Y_I = private|I, B_i)$ and $P(Y_I = public|I, B_i)$ where $B_i \in \mathcal{B}$. We train a meta-classifier on this encoded $\mathcal{D}^T$ dataset using calibrated SVM classifier. We use this meta-classifier to predict privacy class of an encoded target image T (using the posterior probabilities obtained by the base classifiers $P(Y_T = private|T, B_i)$ and $P(Y_T = public|T, B_i)$). As we use $\mathcal{D}^E$ only to learn "competence" classifiers, we do not consider it for training a meta-classifier for a fair comparison.

5. Clusters-based Models (Cluster–en): We create 5 clusters of $\mathcal{D}^T$ dataset using hierarchical clustering mechanism and the combination of object, scene and tag features (F^{ost}). We train a calibrated SVM model on each cluster using the combination of

features F^{ost}. For target image T, the most relevant cluster is identified using $k = 15$ nearest neighbors, and the model trained on that cluster is used to predict the privacy of the target image. We consider this as another baseline, because clustering images that are shared online, brings similar image types (e.g., portraits) together and models trained on these clusters can be competent to predict privacy of target images of respective image types. The number of clusters and neighbors are estimated based on the $\mathcal{D}^E$ dataset.

Table 5 compares the performance obtained by the proposed approach DMFP with the performance obtained using the baseline models. We observe that DMFP learns better privacy characteristics than baselines with respect to private class by providing improvements of 4.5%−14% and 4%−20% in the F1-score and recall of private class. When we learn the "competence" of the base classifiers ($\mathcal{B}$) on the $\mathcal{D}^E$ dataset without identifying the neighborhoods (the first baseline, Zahavy et al. [56]), the precision, recall and F1-score drop by 9%, ≈ 6%, ≈ 7%. It is interesting to note that the precision of DMFP−CL (Refer Table 3, $N_V - CL$, $N_P - CL$, $\{N_V + N_P\} - CL$), i.e., 0.79 is better than the first baseline (Zahavy et al. [56]), i.e., 0.662 whereas the recall of the first baseline (0.568) is better than DMFP−CL (0.534). However, when we combine the neighborhoods ($\{N_V + N_P\}$) and the first baseline (competence learning), i.e., the proposed approach DMFP, we get better performance than each of these methods. Another detail to note that the performance of the first baseline (Zahavy et al. [56]) is very close to the image tags model (see Table 5, 4) and even though the baseline uses multi-modality, the performance does not exceed significantly over the individual base classifiers (object, scene, image). Zahavy et al. [56] performed well for product classification, but it failed to yield improved results for privacy prediction because unlike product images or ImageNet images (that contains single object in the image), images that are shared online are much more complex (containing multiple objects, and scene) and diverse (having different image subjects such as self-portraits, personal events). The results suggest that it is hard to generalize the competency of base classifiers on all types of image subjects and hence, the competence of the base classifiers needs to be determined dynamically based on the neighborhoods of a target image.

Table 5 also shows that F1-measure of private class improves from 0.637 achieved by majority vote (the second baseline), 0.65 obtained by decision-level fusion (the third baseline), 0.636 obtained by stacked−en (the fourth baseline), and 0.545 obtained by cluster−en (the fifth baseline) to 0.684 obtained by DMFP. Additionally, we notice that the proposed approach is able to achieve a performance higher than 85% in terms of all compared measures. Note that a naive baseline which classifies every image as "public" obtains an accuracy of 75%. With a paired T-test, the improvements over the baseline approaches for F1-measure of a private class are statistically significant for p-values < 0.05.

6.5 Proposed Approach vs. Prior Image Privacy Prediction Works

We compare the privacy prediction performance obtained by the proposed approach DMFP with the state-of-the-art works of privacy prediction: **1. object** [49, 50] (B^o), **2. scene** [51] (B^s), **3. image tags** [45, 49, 50] (B^t), 4. PCNH privacy framework [52], and 5. Concatenation of all features [51]. Note that the first three works are the feature sets of DMFP and are evaluated in the Experiment 6.3. We describe the remaining prior works (i.e., 4 and 5) in what follows. **4. PCNH privacy framework** [52]: The framework combines features obtained from two architectures: one that extracts convolutional features (size = 24), and another that extracts object features (size = 24). The object CNN is a very deep network of 11 layers obtained by appending three fully-connected layers of size 512, 512, 24 at the end of the fully-connected layer of AlexNet [25]. The PCNH framework is first trained on the ImageNet dataset [40] and then fine-tuned on a privacy dataset. **5. Combination of Object, Scene and User Tags (Concat)** [51]: Tonge et al. [51] combined object and scene tags with user tags and achieved an improved performance over the individual sets of tags. Thus, we compare the proposed approach with the SVM models trained on the combination of all feature sets (F^{ost}) to show that it will not be adequate to predict the privacy of an image accurately. In our case, we consider object and scene visual features instead of tags and combine them with user tags to study multi-modality with the concatenation of visual and tag features.

Table 7 compares the performance obtained by the proposed approach (DMFP) and prior works. We achieve the highest performance as compared to the prior works and show a maximum improvement of ≈ 10% in the F1-score of private class. We notice that our approach based on multi-modality yields an improvement of 11% over the recall of almost all the prior works (Refer Table 4 and 7). Particularly, we show improvements in terms of all measures over the PCNH framework, that uses two kinds of features object and convolutional. We found that adding high-level descriptive features such as scene context and image tags to the object features help improve the performance. In addition to the individual feature sets, we also outperform the concatenation of these feature sets (denoted as "Concat"), showing that "Concat" could not yield an optimal combination of multi-modality. We notice that the performance of "Concat" is slightly lower than the performance of base classifiers (Refer Tables 4 and 7). We find this is consistent with Zahavy et al. [56] results, that concatenated various layers of image and tag CNN and trained the fused CNN end-to-end but did not yield a better performance than the individual CNN (image or tag).

7 CONCLUSIONS AND FUTURE WORK

In this work, we estimate the competence of object, scene and image tag modalities, derived through convolutional neural networks and dynamically identify the set of most competent modalities for a target image to adequately predict the class of the image as *private* or *public*. The proposed approach contains three stages wherein we first identify neighborhoods for a target image based on visual content similarity and privacy profile similarity. Then, we derive "competence" features from these neighborhoods and provide them to the "competence" classifiers to predict whether a modality is competent for the target image. Lastly, we select the subset of the most competent modalities and take a majority vote to predict privacy class of the target image. Experimental results show that our approach predicts the sensitive (or private) content more accurately than the models trained on an individual modality (object, scene, and tags), multi-modality baselines and prior privacy prediction

Table 7: Dynamic multi-modal fusion for privacy prediction (DMFP) vs. prior image privacy prediction works.

	Private			Public			Overall			
Features	Precision	Recall	F1-score	Precision	Recall	F1-score	Accuracy (%)	Precision	Recall	F1-score
DMFP	**0.752**	**0.627**	**0.684**	**0.891**	**0.936**	**0.913**	**86.36**	**0.856**	**0.859**	**0.856**
PCNH [52]	0.689	0.514	0.589	0.862	0.929	0.894	83.15	0.819	0.825	0.818
Concat [51] (F^{ost})	0.671	0.551	0.605	0.869	0.917	0.892	83.09	0.82	0.826	0.821

approaches. Also, our approach could aid other applications such as event understanding, image classification, to on the fly decide which CNN (object, scene or tag) to use based on a target image.

In the future, it will be interesting to study dynamic multi-modal fusion in personalized privacy setting. Also, other types of competence learning approaches and competence features can be developed for estimating the competence of base classifiers.

ACKNOWLEDGMENTS

This research is supported by the NSF grant #1421970. The computing for this project was performed on Amazon Web Services.

REFERENCES

[1] Shane Ahern, Dean Eckles, Nathaniel S. Good, Simon King, Mor Naaman, and Rahul Nair. 2007. Over-exposed?: privacy patterns and considerations in online and mobile photo sharing. In *CHI '07*.
[2] Andrew Besmer and Heather Lipford. 2009. Tagged photos: concerns, perceptions, and protections. In *CHI '09*.
[3] Avrim Blum and Tom Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory (COLT' 98)*. ACM, New York, NY, USA, 92–100.
[4] Leo Breiman. 1996. Bagging predictors. *Machine Learning* 24, 2 (1996), 123–140.
[5] Leo Breiman. 1996. Stacked Regressions. *Machine Learning* 24, 1 (1996), 49–64.
[6] Daniel Buschek, Moritz Bader, Emanuel von Zezschwitz, and Alexander De Luca. 2015. Automatic Privacy Classification of Personal Photos. In *INTERACT 2015*.
[7] Paulo Rodrigo Cavalin, Robert Sabourin, and Ching Y. Suen. 2011. Dynamic selection approaches for multiple classifier systems. *Neural Computing and Applications* 22 (2011), 673–688.
[8] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* 12 (Nov. 2011), 2493–2537.
[9] Rafael Cruz, Robert Sabourin, and George Cavalcanti. 2018. Dynamic classifier selection: Recent advances and perspectives. *Information Fusion* 41 (05 2018).
[10] Rafael Cruz, Robert Sabourin, George Cavalcanti, and Tsang Ing Ren. 2015. META-DES: A dynamic ensemble selection framework using meta-learning. *Pattern Recognition* 48 (05 2015).
[11] Olivier Debeir, Isabelle Van Den Steen, Patrice Latinne, Philippe Van Ham, and Eleonore Wolff. 2002. Textural and Contextual Land-Cover Classification Using Single and Multiple Classifier Systems. *Photogrammetric Engineering and Remote Sensing* 68 (2002), 597–605.
[12] C. Feichtenhofer, A. Pinz, and A. Zisserman. 2016. Convolutional Two-Stream Network Fusion for Video Action Recognition. In *IEEE CVPR*.
[13] Findlaw. 2017. Is It Safe to Post Photos of Your Kids Online? http://consumer.findlaw.com/online-scams/ is-it-safe-to-post-photos-of-your-kids-online.html.
[14] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc' Aurelio Ranzato, and Tomas Mikolov. 2013. DeViSE: A Deep Visual-Semantic Embedding Model. In *NIPS*. 2121–2129.
[15] Kambiz Ghazinour, Stan Matwin, and Marina Sokolova. 2013. Monitoring and Recommending Privacy Settings in Social Networks. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*. 5.
[16] Yunchao Gong, Liwei Wang, Micah Hodosh, Julia Hockenmaier, and Svetlana Lazebnik. 2014. Improving Image-Sentence Embeddings Using Large Weakly Annotated Photo Collections. In *Computer Vision – ECCV 2014*. 529–545.
[17] Ralph Gross and Alessandro Acquisti. 2005. Information Revelation and Privacy in Online Social Networks. In *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*. 71–80.
[18] Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. 2010. Multimodal semi-supervised learning for image classification. , 902-909 pages.
[19] Panagiotis Ilia, Iasonas Polakis, Elias Athanasopoulos, Federico Maggi, and Sotiris Ioannidis. 2015. Face/Off: Preventing Privacy Leakage From Photos in Social Networks. In *Proceedings of the 22Nd ACM Conf. on Computer and Comm. Security*.
[20] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. In *Proceedings of the ACM International Conference on Multimedia*. 675–678.
[21] Simon Jones and Eamonn O'Neill. 2011. Contextual dynamics of group-based sharing decisions *(CHI '11)*. 10.
[22] A. Kannan, P. P. Talukdar, N. Rasiwasia, and Q. Ke. 2011. Improving Product Classification Using Images. In *2011 IEEE 11th Int. Conf. on Data Mining*.
[23] Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. 2014. Multimodal Neural Language Models. In *Proceedings of the 31st International Conf. on ML*, Vol. 32.
[24] Balachander Krishnamurthy and Craig E. Wills. 2008. Characterizing Privacy in Online Social Networks. In *Proceedings of the First Workshop on Online Social Networks (WOSN '08)*. ACM, New York, NY, USA, 37–42.
[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*. 1097–1105.
[26] Mark Laan, Eric C Polley, and Alan Hubbard. 2007. Super Learner. *Statistical applications in genetics and molecular biology* 6 (02 2007), Article 25.
[27] Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *ICML (JMLR Workshop and Conf. Proceedings)*, Vol. 32. 1188–1196.
[28] Yann LeCun. 2017. Facebook Envisions AI That Keeps You From Uploading Embarrassing Pics. https://www.wired.com/2014/12/fb/all/1.
[29] Yifang Li, Wyatt Troutman, Bart P. Knijnenburg, and Kelly Caine. 2018. Human Perceptions of Sensitive Content in Photos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.
[30] Heather Richter Lipford, Andrew Besmer, and Jason Watson. 2008. Understanding Privacy Settings in Facebook with an Audience View. In *Proceedings of the 1st Conference on Usability, Psychology, and Security (UPSEC'08)*. 2:1–2:8.
[31] D. Lu and Q. Weng. 2007. A survey of image classification methods and techniques for improving classification performance. *Int. Journal of Remote Sensing* (2007).
[32] Corey Lynch, Kamelia Aryafar, and Josh Attenberg. 2016. Images Don'T Lie: Transferring Deep Visual Semantic Features to Large-Scale Multimodal Learning to Rank. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. 541–548.
[33] Mary Madden. 2012. Privacy management on social media sites. http://www.pewinternet.org/2012/02/24/privacy-management-on-social-media-sites.
[34] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng. 2011. Multimodal Deep Learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning (ICML'11)*.
[35] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2017. Towards a Visual Privacy Advisor: Understanding and Predicting Privacy Risks in Images. In *IEEE International Conference on Computer Vision, ICCV 2017*. 3706–3715.
[36] J. Parra-Arnau, A. Perego, E. Ferrari, J. FornÃľ, and D. Rebollo-Monedero. 2014. Privacy-Preserving Enhanced Collaborative Tagging. *IEEE Transactions on Knowledge and Data Engineering* 26, 1 (Jan 2014), 180–193.
[37] Javier Parra-Arnau, David Rebollo-Monedero, Jordi Forné, Jose L. Muñoz, and Oscar Esparza. 2012. Optimal tag suppression for privacy protection in the semantic Web. *Data Knowl. Eng.* 81-82 (2012), 46–66.
[38] Soujanya Poria, Erik Cambria, Newton Howard, Guang-Bin Huang, and Amir Hussain. 2016. Fusing Audio, Visual and Textual Clues for Sentiment Analysis from Multimodal Content. *Neurocomput.* 174, PA (Jan. 2016), 50–59.
[39] Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael L. Littman. 2005. Activity Recognition from Accelerometer Data. In *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence*. 1541–1546.
[40] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *IJCV* (April 2015), 1–42.
[41] Karen Simonyan and Andrew Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* abs/1409.1556 (2014).
[42] Andrew Simpson. 2008. On the Need for User-defined Fine-grained Access Control Policies for Social Networking Applications. In *Proceedings of the Workshop on Security in Opportunistic and SOCial Networks (SOSOC '08)*. Article 1, 8 pages.
[43] Marina Skurichina and Robert P.W. Duin. 1998. Bagging for linear classifiers. *Pattern Recognition* 31, 7 (1998), 909 – 930.

[44] Eleftherios Spyromitros-Xioufis, Symeon Papadopoulos, Adrian Popescu, and Yiannis Kompatsiaris. 2016. Personalized Privacy-aware Image Classification. In *Proceedings of the ACM on Int. Conf. on Multimedia Retrieval (ICMR '16)*. 71–78.

[45] Anna Squicciarini, Cornelia Caragea, and Rahul Balakavi. 2014. Analyzing Images' Privacy for the Modern Web *(HT '14)*. ACM, New York, NY, USA, 136–147.

[46] Anna Squicciarini, Cornelia Caragea, and Rahul Balakavi. 2017. Toward Automated Online Photo Privacy. *ACM TWeb* 11, 1, Article 2 (2017), 2:1–2:29 pages.

[47] Anna Squicciarini, Smitha Sundareswaran, Dan Lin, and Josh Wede. 2011. A3P: Adaptive Policy Prediction for Shared Images over Popular Content Sharing Sites. In *Proceedings of the 22Nd ACM Conf. on Hypertext and Hypermedia*. 261–270.

[48] H. Sundaram, L. Xie, M. De Choudhury, Y.R. Lin, and A. Natsev. 2012. Multimedia Semantics: Interactions Between Content and Community. *Proc. IEEE* (2012).

[49] Ashwini Tonge and Cornelia Caragea. 2016. Image Privacy Prediction Using Deep Features. In *AAAI '16*.

[50] Ashwini Tonge and Cornelia Caragea. 2018. On the Use of "Deep" Features for Online Image Sharing. In *Proceedings of The Web Conference Companion*.

[51] Ashwini Tonge, Cornelia Caragea, and Anna Squicciarini. 2018. Uncovering Scene Context for Predicting Privacy of Online Shared Images. In *AAAI '18*.

[52] Lam Tran, Deguang Kong, Hongxia Jin, and Ji Liu. 2016. Privacy-CNH: A Framework to Detect Photo Privacy with Convolutional Neural Network Using Hierarchical Features. In *AAAI '16*. 7.

[53] L. Wang, Zhe Wang, Wenbin Du, and Yu Qiao. 2015. Object-Scene Convolutional Neural Networks for Event Recognition in Images. *CoRR* abs/1505.00296 (2015).

[54] Susan Waters and James Ackerman. 2011. Exploring Privacy Management on Facebook: Motivations and Perceived Consequences of Voluntary Disclosure. *Journal of Computer-Mediated Communication* 17, 1 (2011), 101–115.

[55] Jun Yu, Baopeng Zhang, Zhengzhong Kuang, Dan Lin, and Jianping Fan. 2017. iPrivacy: Image Privacy Protection by Identifying Sensitive Objects via Deep Multi-Task Learning. *IEEE Trans. Information Forensics and Security* 12, 5 (2017).

[56] Tom Zahavy, Abhinandan Krishnan, Alessandro Magnani, and Shie Mannor. 2018. Is a Picture Worth a Thousand Words? A Deep Multi-Modal Architecture for Product Classification in E-Commerce. In *AAAI*. AAAI Press.

[57] Sergej Zerr, Stefan Siersdorfer, Jonathon Hare, and Elena Demidova. 2012. Privacy-aware image classification and search. In *ACM SIGIR*. ACM, NY, USA.

[58] Elena Zheleva and Lise Getoor. 2009. To Join or Not to Join: The Illusion of Privacy in Social Networks with Mixed Public and Private User Profiles. In *WWW '09*.

[59] Haoti Zhong, Anna Squicciarini, David Miller, and Cornelia Caragea. 2017. A Group-Based Personalized Model for Image Privacy Classification and Labeling. In *IJCAI' 17*. 3952–3958.

[60] B. Zhou, A. Khosla, A. Lapedriza, Antonio Torralba, and Aude Oliva. 2016. Places: An Image Database for Deep Scene Understanding. *arXiv:1610.02055* (2016).

Signed Distance-based Deep Memory Recommender

Thanh Tran
Department of Computer Science
Worcester Polytechnic Institute
Massachusetts, USA
tdtran@wpi.edu

Xinyue Liu
Department of Computer Science
Worcester Polytechnic Institute
Massachusetts, USA
xliu4@wpi.edu

Kyumin Lee
Department of Computer Science
Worcester Polytechnic Institute
Massachusetts, USA
kmlee@wpi.edu

Xiangnan Kong
Department of Computer Science
Worcester Polytechnic Institute
Massachusetts, USA
xkong@wpi.edu

ABSTRACT

Personalized recommendation algorithms learn a user's preference for an item by measuring a distance/similarity between them. However, some of the existing recommendation models (e.g., matrix factorization) assume a linear relationship between the user and item. This approach limits the capacity of recommender systems, since the interactions between users and items in real-world applications are much more complex than the linear relationship. To overcome this limitation, in this paper, we design and propose a deep learning framework called *Signed Distance-based Deep Memory Recommender*, which captures non-linear relationships between users and items *explicitly* and *implicitly*, and work well in both general recommendation task and shopping basket-based recommendation task. Through an extensive empirical study on six real-world datasets in the two recommendation tasks, our proposed approach achieved significant improvement over ten state-of-the-art recommendation models.

KEYWORDS

Memory recommender; signed distance; metric-based attention.

ACM Reference Format:
Thanh Tran, Xinyue Liu, Kyumin Lee, and Xiangnan Kong. 2019. Signed Distance-based Deep Memory Recommender. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313460

1 INTRODUCTION

Recommender systems [1] have been deployed in many online applications such as e-commerce, music/video streaming services, social media, *etc.* They have played a vital role for users to explore new items and for companies to increase their revenues. Most of recommendation algorithms model user preferences and item properties based on observed interactions (*e.g.,* clicks, reviews, ratings) between users and items [20, 21, 30]. In a perspective, we can view most of the recommendation models as a measurement of similarity or distance between a user and an item. For instance, the well known latent factor (i.e., matrix factorization) models [19] usually employ an inner product function to approximate the similarity between the user and the item. Although the latent factor models achieved competitive performance in some datasets, they did not correctly capture complex (i.e., non-linear) relationships between users and items because the inner product function follows limited linear nature.

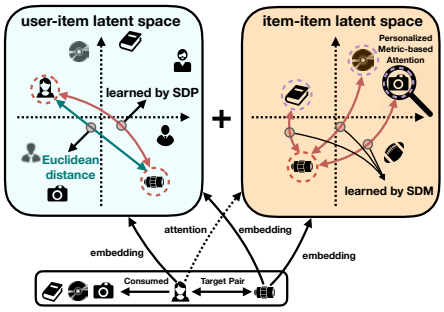


Figure 1: We consider a recommender as a signed distance approximator, and decompose the signed distance between a user and an item into two parts: the left box learns an explicitly signed distance between the user and item (i.e., the *camera lens*), the right box learns an implicitly signed distance between the user and the item via the user's recently consumed items (i.e., the *book*, *CD* and *camera*). Our novel personalized metric-based soft attention is applied to the consumed items to optimize their contributions to the output signed distance score. Then the two parts are combined to obtain a final score. Most of linear latent factor models are equivalent to simply measuring the linear Euclidean distance in the user-item latent space (shown as the green line).

Existing recommendation algorithms faced difficulties in finding good kernels for different data patterns [30], only focused on user-item latent space without considering the item-item latent space together [12, 24, 25, 43, 57], or required additional auxiliary information (e.g., item description, music content, reviews) [4, 17, 29, 31, 53]. To overcome the drawbacks, in this paper we aim to propose and build a deep learning framework to learn a non-linear relationship

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313460

between a user and a target item by measuring a distance from the observed data. In particular, we propose *Signed Distance-based Deep Memory Recommender* (SDMR), which captures non-linear relationship of the user and item *explicitly* and *implicitly*, combines *explicitly* and *implicitly* measured relationship to produce a final distance score for the recommendation, and performs well in both general recommendation task and shopping basket-based recommendation task.

SDMR internally combines two signed distances, each of which is measured by our proposed *Signed Distance-based Perceptron* (SDP) and *Signed Distance-based Memory Network* (SDM). On one hand, SDP explicitly measures a non-linear signed distance between the user and the item. Many existing models [13, 16] rely on a pre-defined metric such as Euclidean distance (the green line in Figure 1) which is much more limited than the customized non-linear signed distance learned from the data (the red curves in Figure 1). On the other hand, SDM implicitly measures a non-linear signed distance between the user and the item via the user's recently consumed items. SDM is similar to the item neighborhood-based recommender [36, 42] in nature. However, it is more advanced in several aspects, as shown in the right side of Figure 1. First, SDM only focuses on a set of recently consumed items of the target user (*e.g.,* the *book*, *CD* and *camera* in Figure 1) as context items. Second, it employs additional memories to learn a novel personalized metric-based attention on the consumed items. The goal of our proposed attention is to compute weights of each consumed item *w.r.t.* the target item (i.e., the *camera lens*). In the example, the attention module assigns higher weights on the *camera* and lower weights on the *book* and *CD*. Unlike our approach, most of the existing neighborhood-based models consider contribution of consumed items to the target item equally, leading to suboptimal results. Last but not the least, we update the attention weights via a gated multi-hop to build a long-term memory within SDM. This multi-hop design helps refine our attention module and produces more accurate attentive scores.

The contributions of this work are summarized as follows:

- We design a deep learning framework which can tackle both general recommendation task and shopping basket-based recommendation task.
- We propose SDMR that combines two signed distance scores internally measured by SDP and SDM, which capture non-linear relationship between a user and an item explicitly and implicitly.
- To better balance the weights among consumed items of the user, we propose a novel multi-hop memory network with a personalized metric-based attention mechanism in SDM.
- Extensive experiments on six datasets in two different recommendation tasks demonstrate the effectiveness of our proposed methods against ten baselines.

2 RELATED WORK

Latent Factor Models (LFM) have been extensively studied in the literature, which include Matrix Factorization [16], Bayesian Personalized Ranking [39], fast matrix factorization for implicit feedbacks (eALS) [13], *etc.* Despite their success, LFM suffer from several limitations. First, LFM overlook associations between the user's previously consumed items and the target item (e.g. mobile phones and phone cases). Second, LFM usually rely on inner product function, whose linearity limits the capability of modeling complex user-item interactions. To address the second issue, several non-linear latent factor models have been proposed, with the help of Gaussian process [23] or kernels [30, 62]. However, they either require expensive hyper-parameter tuning or face difficulties in finding good kernels for different data patterns.

Neighborhood-based models [36, 42] are usually based on the principle that similar users prefer similar items. The problem turns into finding the neighbors of a user or an item based on a pre-defined distance/similarity metric, such as *cosine vector similarity* [3, 22], *Person Correlation similarity* [6], *etc.* The recommendation quality highly depends on a chosen metric, but finding a good pre-defined metric is usually very challenging. Furthermore, these models are also sensitive to the selection of neighbors. Our proposed SDM is similar to neighborhood-based models in nature, but it exploits a novel personalized metric-based attention for assigning attentive weights to context items. Therefore, our approach is more robust and less sensitive than conventional neighborhood-based models.

NeuMF [12] is a neural network that generalizes matrix factorization via Multi Layer Perceptron (MLP) for learning non-linear interaction functions. Similarly, some other works [24, 25, 43, 57] substitute MLP with auto-encoder architecture. It is worth noting that all these approaches are limited by only considering the user-item latent space, and overlook the correlations in the item-item latent space. Besides, some deep learning based works [32, 34, 45, 51] employ auxiliary information such as item description [17], music content [53], item visual features [4, 29], reviews [31] to address the cold-start problem. However, this auxiliary information is not always available, and it limits their applicability in many real-world systems. Another line of works use deep neural networks to model temporal effects of consumed items [14, 37, 49, 56]. Although our proposed methods do not explicitly consider the temporal effects, SDM utilizes the time information to select a set of recently consumed items as the context items of the target item.

The most closely related work to our work is recently proposed (*Collaborative Memory Network* (CMN) [7]). In this work, Memory Network [48] is adapted to measure similarities between users and user neighbors. Key differences between our work and CMN are as follows: (i) First, we follow an item neighborhood based design, whereas CMN follows a user neighborhood based design. The prior work showed that item neighborhood based models slightly outperformed user neighbor based models [27, 42]; (ii) Second, our proposed SDM model uses our proposed personalized metric-based attention mechanism and produces signed distance scores as output, whereas CMN exploited a traditional inner product based attention; (iii) Third, we use a gated multi-hop architecture [28], which was shown to perform better than the original multi-hop design [48].

3 PROBLEM STATEMENT

In this section, we describe two recommendation problems: (i) general recommendation task; and (ii) shopping basket-based recommendation task. In following sections, we focus on solving them.

General recommendation task: Given a whole item set $V = \{v_1, v_2, ..., v_{|V|}\}$, and a whole user set $U = \{u_1, u_2, ..., u_{|U|}\}$. Each user $u_i \in U$ may consume several items $\{v_{i1}, v_{i2}, ..., v_{ik}\}$ in V,

denoted as a set of context items c. In this task, given previously consumed items of a user u_i, a recommendation model predicts a next target item v_j that u_i may prefer, denoting this task as estimating $P(u_i, v_j|c)$. Note that some existing works assume independent relationships between v_j and context items in the set $\boldsymbol{c}$, leading to $P(u_i, v_j|c) = P(u_i, v_j)$ [12, 13]. In our work, we model the u_i's preference on v_j in two steps: (i) an explicit preference of u_i on v_j in a signed distance based perceptron, and (ii) an implicit preference of u_i on v_j via summing attentive effects of context items toward target item v_j in a signed distance based memory network.

Shopping Basket-based recommendation task: This problem is based on the fact that users go shopping offline/online and add some items into a basket/cart together. Each shopping basket/cart is seen as a transaction, and each user may shop once or multiple times, leading to one or multiple transactions. Let $T^{(u)} = \{t_1, t_2, ..., t_{|T^{(u)}|}\}$ as a set of the user u's transactions, where $|T^{(u)}|$ denotes the number of user u's transactions. Each transaction $t_i = \{v_1, v_2, ..., v_{|t_i|}\}$ consists of several items in the whole item set V. In this problem, it is assumed that all the items in t_i are inserted into the same basket at the same time, ignoring the actual order of the items being inserted and considering t_i's transaction time as each item's insertion time. Given a target item $v_j \in t_i$, the rest of the items in t_i will be seen as the context items of v_j, denoted as c (i.e. $c = t_i \backslash \{v_j\}$). Then, given the set of context items c, a recommendation model predicts a conditional probability $P(u, v_j|c)$, which is interpreted as the conditional probability that u will add the item v_j into the same basket with the other items c.

Both of the recommendation tasks above are popular in the literature [8, 12, 37, 40]. The *general recommendation task* differs from the *shopping basket-based recommendation task* because there is no specific context items of the target item in the *general recommendation task*. Note that the two tasks are *personalized* recommendation problems. In fact, there are *non-personalized* recommendation problems such as *session-based recommendation* [14], where users (i.e. user IDs) are not available in transactions. However, in this paper, we focus on *personalized* recommendation tasks because they are more preferred in the literature [8, 37, 40].

4 PROPOSED METHODS

Our proposed *Signed Distance-based Deep Memory Recommender* (SDMR) consists of two major components: *Signed Distance-based Perceptron* (SDP) and *Signed Distance-based Memory network* (SDM). We first describe an overview of our models as follows:

- Given a target user i and a target item j as two one-hot vectors, we pass the two vectors through the user and item embedding spaces to get user embedding u_i and item embedding v_j.
- On one hand, our proposed *Signed Distance-based Perceptron* (SDP) will measure a signed distance score between u_i and v_j by a multi-layer perceptron network.
- On the other hand, given target user i, target item j, and the user i's recently consumed context items s as the input, our *Signed Distance-based Memory network* (SDM) will measure a signed distance score between user i and item j via attentive distances between context items s and target item j.

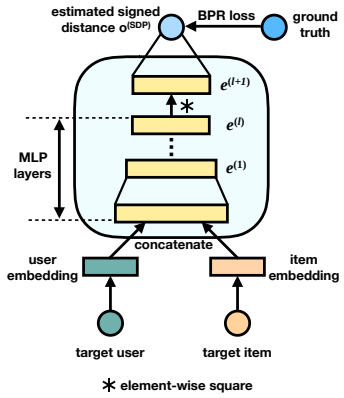


Figure 2: The illustration of our SDP model.

- Then, the *Signed Distance-based Deep Memory Recommender* (SDMR) model will measure a total distance between user i and item j by learning a combination of SDP and SDM. The smaller the total distance is, the more likely user i will consume item j.

Next, we describe SDP, SDM, and SDMR in detail.

4.1 Signed Distance-based Perceptron (SDP)

We first propose *Signed Distance-based Perceptron* (SDP) that *explicitly* learns a signed distance between a target user i and a target item j. An illustration of SDP is shown in Figure 2. Let the embedding of a target user i be $\boldsymbol{u}_i \in \mathbb{R}^d$, and the embedding of a target item j be $\boldsymbol{v}_j \in \mathbb{R}^d$, where d is the number of dimensions in each embedding. First, SDP takes a concatenation of these two embeddings as the input and proceeds as follows:

$$\boldsymbol{e}^{(1)} = f_1(\mathbf{W}^{(1)} \begin{bmatrix} \boldsymbol{u}_i \\ \boldsymbol{v}_j \end{bmatrix} + \boldsymbol{b}^{(1)}) \tag{1}$$

$$\boldsymbol{e}^{(2)} = f_2(\mathbf{W}^{(2)} \boldsymbol{e}^{(1)} + \boldsymbol{b}^{(2)}) \tag{2}$$

$$\dots \tag{3}$$

$$\boldsymbol{e}^{(\ell)} = f_\ell(\mathbf{W}^{(\ell)} \boldsymbol{e}^{(\ell-1)} + \boldsymbol{b}^{(\ell)}) \tag{4}$$

$$\boldsymbol{e}^{(\ell+1)} = square(\boldsymbol{e}^{(\boldsymbol{\ell})}) \tag{5}$$

$$o^{(SDP)} = \boldsymbol{w}^{(o)\top} \boldsymbol{e}^{(\ell+1)} + \boldsymbol{b}^{(o)} \tag{6}$$

where $f_l(\cdot)$ refers to a non-linear activation function at the layer l^{th} (e.g. `sigmoid`, `ReLu` or `tanh`), and $square(\cdot)$ denotes an element-wise square function (e.g $square([2, 3]) = [6, 9]$). Through experimental results, we choose `tanh` as the activation function because it yields slightly better results than `ReLu`. From now on, we will use $f(\cdot)$ to denote the `tanh` function. It can be easily observed that Eq. (1) – (4) form a trivial Multi-Layer Perceptron (MLP) network, which is a popular design [12, 60] to learn a complex and non-linear interaction between user embedding $\boldsymbol{u}_i$ and item embedding $\boldsymbol{v}_j$. Our new design starts at Eq. (5) – Eq. (6). In Eq. (5), we apply the element-wise squared function $square(\cdot)$ to the output vector $\boldsymbol{e}^{(l)}$ of the MLP and obtain a new output vector $\boldsymbol{e}^{(l+1)}$. Next, in Eq. (6), we use a fully connected layer $\boldsymbol{w}^{(o)}$ to combine different dimensions in $\boldsymbol{e}^{(l+1)}$ and yields a final distance value $o^{(SDP)}$. Our idea of using $\boldsymbol{w}^{(o)}$ in here is that after applying the element-wise square function $square(\cdot)$ in Eq. (5), all the dimensions in $\boldsymbol{e}^{(l+1)}$ will be

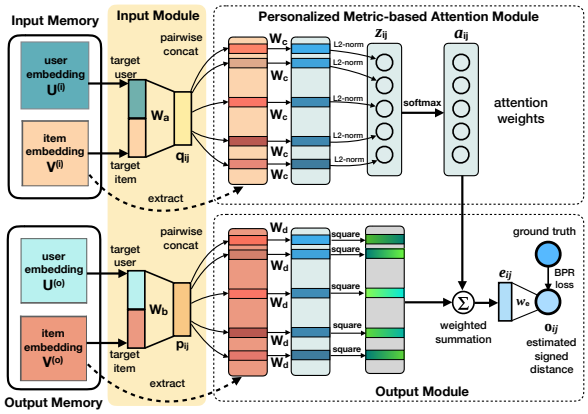


Figure 3: The illustration of a single-hop SDM, which consists of a memory module, an input module, an attention module, and an output module.

non-negative. Thus, we consider each dimension of $\boldsymbol{e}^{(l+1)}$ as a distance value. The edge weights $\boldsymbol{w}^{(o)}$ will then be used to combine those distant dimensions to provide a more fine-grained distance.

We note that SDP can be reduced to a squared Euclidean distance with the following setting: at Eq. (1), $\mathbf{W}^{(1)} = [\mathbb{1}, -\mathbb{1}]$ with $\mathbb{1}$ denotes an identity matrix and so $\mathbf{W}^{(1)} \begin{bmatrix} \boldsymbol{u}_i \\ \boldsymbol{v}_j \end{bmatrix} = \boldsymbol{u}_i - \boldsymbol{v}_j$; the activation $f(\cdot)$ is an identity function; the number of MLP layers $\ell = 1$; the edge-weights layer at Eq. (6): $\boldsymbol{w}^{(o)} = \mathbf{1}$ (e.g. the all-ones matrix), bias $\boldsymbol{b}^{(o)} = \mathbf{0}$. Note that if $\boldsymbol{w}^{(o)}$ in Eq. (6) is an all-negative layer, it will yield a negative value, which we name as a signed distance[1] score. If we see each user $\boldsymbol{i}$ as a point in multi dimensional space, and the user's preference space is defined by a boundary Ω, we can interpret this signed distance score as follows: When the item $\boldsymbol{j}$ is out of the user $\boldsymbol{i}$'s preference boundary Ω, the distance $d(\boldsymbol{i}, \boldsymbol{j})$ between them is positive (i.e. $d(\boldsymbol{i}, \boldsymbol{j}) > 0$) and it reflects that user $\boldsymbol{i}$ does not prefer item $\boldsymbol{j}$. When the distance between user $\boldsymbol{i}$ and item $\boldsymbol{j}$ is shortened and $\boldsymbol{j}$ is right on the boundary Ω, the distance between them is zero and it indicates user $\boldsymbol{i}$ likes item $\boldsymbol{j}$. As $\boldsymbol{j}$ is coming inside Ω, the distance between them becomes negative and reflects a higher preference of user $\boldsymbol{i}$ on item $\boldsymbol{j}$. In short, we can see SDP as a signed distance function, which could learn a complex signed distance between a user and an item via a MLP architecture with non-linear activations and an element-wise square function $square(\cdot)$. In the recommendation domain, the signed distances will provide more fine-grained distance values, thus, reflecting users' preferences on items more accurately.

4.2 Signed Distance-based Memory Network (SDM)

We propose a multi-hop memory network, *Signed Distance-based Memory network* (SDM), to model *implicit* preference of a user on the target item via the user's previously consumed items (i.e., context items). The implicit preference is represented as a signed distance. First, we describe a single-hop SDM, and then describe how to extend it into a multi-hop design. Following the traditional architecture of a memory network [28, 48, 58], our proposed single-hop SDM has four main components: a memory module, an input module, an attention module, and an output module. The overview of SDM's architecture is presented in Figure 3. We will go into details of each SDM's module as follows:

4.2.1 Memory Module: We maintain two memories called input memory and output memory. The input memory contains two embedding matrices $\mathbf{U}^{(i)} \in \mathbb{R}^{M \times d}$ and $\mathbf{V}^{(i)} \in \mathbb{R}^{N \times d}$, where M and N are the number of users and the number of items in the system, respectively. d denotes the embedding size of each user and each item. Similarly, the output memory also contains two embedding matrices $\mathbf{U}^{(o)} \in \mathbb{R}^{M \times d}$ and $\mathbf{V}^{(o)} \in \mathbb{R}^{N \times d}$. As shown in Figure 3, the input memory will be used to calculate attention weights of a user's consumed items (i.e., context items), whereas the output memory will be used to measure a final signed distance between the target user and the target item via the user's context items.

Given a target user i, a target item j and a set of user i's consumed items as context items $\mathcal{T}_j^{\,i}$, the output of this module is the embeddings of user i, item j, and all context items $k \in \mathcal{T}_j^{\,i}$: $(\boldsymbol{u}_i, \boldsymbol{v}_j, <\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_k>)$. Since this module has a separated input memory and output memory, we obtain $(\boldsymbol{u}_i^{(i)}, \boldsymbol{v}_j^{(i)}, <\boldsymbol{v}_1^{(i)}, \boldsymbol{v}_2^{(i)}, ..., \boldsymbol{v}_k^{(i)}>)$ as the output of the input memory, and $(\boldsymbol{u}_i^{(o)}, \boldsymbol{v}_j^{(o)}, <\boldsymbol{v}_1^{(o)}, \boldsymbol{v}_2^{(o)}, ..., \boldsymbol{v}_k^{(o)}>)$ as the output of the output memory. It is obvious that $\boldsymbol{u}_i^{(i)}$ is the i-th row of $\mathbf{U}^{(i)}$, $\boldsymbol{v}_j^{(i)}$ and $\boldsymbol{v}_k^{(i)}$ are the corresponding j-th and k-th row of $\mathbf{V}^{(i)}$. A similar explanation is applied to $\boldsymbol{u}_i^{(o)}$ $\boldsymbol{v}_j^{(o)}$, and $\boldsymbol{v}_k^{(o)}$.

4.2.2 Input Module: The goal of the input module is to form a non-linear combination between the target user embedding and the target item embedding. Given the target user embedding $\boldsymbol{u}_i^{(i)}$ and the target item embedding $\boldsymbol{v}_j^{(i)}$ from the input memory in the memory module, following the widely adopted design in multimodal deep learning work [47, 61], the input module simply concatenates the two embeddings, and then applies a fully connected layer with a non-linear activation $f(\cdot)$ (i.e. `tanh` function) to obtain a coherent hidden feature vector as follows:

$$\boldsymbol{q}_{ij} = f\left(\mathbf{W}_a \begin{bmatrix} \boldsymbol{u}_i^{(i)} \\ \boldsymbol{v}_j^{(i)} \end{bmatrix} + \boldsymbol{b}_a\right) \tag{7}$$

where $\mathbf{W}_a \in \mathbb{R}^{d \times 2d}$ is the weights of input module. Note that $q_{ij} \in \mathbb{R}^d$ can be seen as a query embedding in Memory Network [48].

Similarly, if the inputs of the input module are the target user embeddings $\boldsymbol{u}_i^{(o)}$ and the target item embeddings $\boldsymbol{v}_j^{(o)}$ from the output memory, we can form a non-linear combination between $\boldsymbol{u}_i^{(o)}$ and $\boldsymbol{v}_j^{(o)}$ (i.e. an output query), denoted as $\boldsymbol{p}_{ij}$, as follows:

$$\boldsymbol{p}_{ij} = f\left(\mathbf{W}_b \begin{bmatrix} \boldsymbol{u}_i^{(o)} \\ \boldsymbol{v}_j^{(o)} \end{bmatrix} + \boldsymbol{b}_b\right) \tag{8}$$

4.2.3 Attention Module: The goal of the attention module is to assign attentive scores to different context items (or candidates) given the combined vector (or a query) $\boldsymbol{q}_{ij}$ of the target user i and target item j obtained in Eq. (7). First, we calculate the squared $\mathcal{L}2$ distance between $\boldsymbol{q}_{ij}$ and each candidate item $\boldsymbol{v}_k^{(i)}$ as follows:

[1] https://en.wikipedia.org/wiki/Signed_distance_function

$$z_{ijk} = \left\| f\left(\mathbf{W}_c \begin{bmatrix} \boldsymbol{q}_{ij} \\ \boldsymbol{v}_k^{(i)} \end{bmatrix} + \boldsymbol{b}_c\right)\right\|_2^2 \quad (9)$$

where $||\cdot||_2$ refers to the $\mathcal{L}2$ distance (or Euclidean distance), which is widely used in previous works to measure similarity among items [8] or between users and items [15]. To better understand our intuition in Eq. (9), we will break it into smaller parts and explain them. First, similar to the intuition of Eq. (7), we have $f\left(\mathbf{W}_c \begin{bmatrix} \boldsymbol{q}_{ij} \\ \boldsymbol{v}_k^{(i)} \end{bmatrix} + \boldsymbol{b}_c\right)$ component to define a non-linear combination between the input query $\boldsymbol{q}_{ij}$ and each context item embeddings $\boldsymbol{v_k}^{(i)}$. Then, $||\cdot||_2^2$ will measure the squared $\mathcal{L}2$ distance of the combined vector. It is worth to note that with a following setting: $\boldsymbol{W}_a = [\mathbf{0}, \mathbb{I}]$ where $\mathbb{I}$ refers to an identity matrix and $\mathbf{0}$ is an all-zeros matrix; $f(\cdot)$ is an identity function; $\boldsymbol{W}_c = [\mathbb{I}, -\mathbb{I}]$; bias terms $\boldsymbol{b}_a = \boldsymbol{b}_c = 0$. Then, in Eq. (7), $\boldsymbol{q}_{ij} = f\left(\mathbf{W}_a \begin{bmatrix} \boldsymbol{u}_i^{(i)} \\ \boldsymbol{v}_j^{(i)} \end{bmatrix} + \boldsymbol{b}_a\right) = \boldsymbol{v}_j^{(i)}$; in Eq. (9), $f\left(\mathbf{W}_c \begin{bmatrix} \boldsymbol{q}_{ij} \\ \boldsymbol{v}_k^{(i)} \end{bmatrix} + \boldsymbol{b}_c\right) = \boldsymbol{v}_j^{(i)} - \boldsymbol{v}_k^{(i)}$, and $z_{ijk} = ||(\boldsymbol{v}_j^{(i)} - \boldsymbol{v}_k^{(i)})||_2^2$, which simply generalizes a squared $\mathcal{L}2$ distance between the target item j and the context item k. Additionally, with another setting: $\boldsymbol{W}_a = [\mathbb{I}, -\mathbb{I}]$; $f(\cdot)$ is an identity function; $\boldsymbol{W}_c = [\mathbb{I}, \mathbb{I}]$; bias terms $\boldsymbol{b}_a = \boldsymbol{b}_c = 0$. Then, in Eq. (7), $\boldsymbol{q}_{ij} = f\left(\mathbf{W}_a \begin{bmatrix} \boldsymbol{u}_i^{(i)} \\ \boldsymbol{v}_j^{(i)} \end{bmatrix} + \boldsymbol{b}_a\right) = \boldsymbol{u}_i^{(i)} - \boldsymbol{v}_j^{(i)}$, in Eq. (9), $f\left(\mathbf{W}_c \begin{bmatrix} \boldsymbol{q}_{ij} \\ \boldsymbol{v}_k^{(i)} \end{bmatrix} + \boldsymbol{b}_c\right) = \boldsymbol{u}_i^{(i)} - \boldsymbol{v}_j^{(i)} + \boldsymbol{v}_k^{(i)}$, and $z_{ijk} = ||(\boldsymbol{v}_k^{(i)} + \boldsymbol{u}_i^{(i)} - \boldsymbol{v}_j^{(i)})||_2^2$, which simply generalizes a squared $\mathcal{L}2$ distance between the target item j and the context item k where the user i plays as a translator [9]. The two examples above show that our proposed design can learn a more generalized distance between target and context items.

The output squared $\mathcal{L}2$ distance in Eq. (9) will show how similar the target item j and the context item k are. The lower the distance score is, the more similar two items j and k are. Next, we use the Softmax function to normalize and obtain attentive score between j and k as follows:

$$a_{ijk} = \frac{exp(-z_{ijk})}{\sum_{p \in \mathcal{T}_j^i} exp(-z_{ijp})} \quad (10)$$

where $\mathcal{T}_j^i$ is the set of user $\boldsymbol{i}$'s neighborhood items. The **minus** sign in Eq. (10) is used to assign a higher attention score for a lower distance between two items (j, k).

We note that the $\mathcal{L}2$ distance (or Euclidean distance) satisfies four conditions of a metric [2]. While the crucial triangle inequality property of a metric was shown to provide a better performance compared to the inner product [15, 38, 46] in recommendation domains, to our best of knowledge, most of existing attention designs [2, 5, 26, 33, 44, 54, 59] adopted the inner product for measuring attentive scores. Hence, this proposed attention design is the **first attempt** to bring metric properties into the attention mechanism.

Similar to [50], we limit the number of considering context items by choosing the user $\boldsymbol{i}$'s $\boldsymbol{s}$ most recently consumed items before target item $\boldsymbol{j}$ as the context items of target item $\boldsymbol{j}$. Here, $\boldsymbol{s}$ can be selected via tuning with a development dataset. The soft attention vector containing attentive contribution scores of $\boldsymbol{s}$ context items toward the target item $\boldsymbol{j}$ of a user $\boldsymbol{i}$ is given as follows:

$$\boldsymbol{a}_{ij} = \left[a_{ij1}, \cdots, a_{ijs}\right]^T \quad (11)$$

4.2.4 Output Module: Given the attentive scores $\boldsymbol{a}_{ij}$ in Eq.(11) and the combined vector $\boldsymbol{p}_{ij} \in \mathbb{R}^d$ of the user embedding $\boldsymbol{u}_i^{(o)}$ and item embedding $\boldsymbol{v}_j^{(o)}$ from the output memory $\boldsymbol{U}^{(o)}$ and $\boldsymbol{V}^{(o)}$, the goal of this output module is to measure a total output distance $\boldsymbol{o}_{ij}^{(SDM)}$ between the output target item embeddings $\boldsymbol{v}_j^{(o)}$ and all the user $\boldsymbol{i}$'s output context item embeddings $\boldsymbol{v}_k^{(o)} (k \in T_j^i)$ using attention weights $\boldsymbol{a}_{ij}$ and the output query $\boldsymbol{p}_{ij}$ as follows:

$$o_{ij}^{(SDM)} = \boldsymbol{w}_e^\top \boldsymbol{e}_{ij} + b_e \quad (12)$$

where $\boldsymbol{e}_{ij} \in \mathbb{R}^d$ is calculated as follows:

$$\boldsymbol{e}_{ij} = \sum_{k \in \mathcal{T}_j^i} \boldsymbol{a}_{ijk} \times square\left(f\left(\mathbf{W}_d \begin{bmatrix} \boldsymbol{p}_{ij} \\ \boldsymbol{v}_k^{(o)} \end{bmatrix} + \boldsymbol{b}_d\right)\right) \quad (13)$$

In here, let $\boldsymbol{r}_{ijk} = f\left(\mathbf{W}_d \begin{bmatrix} \boldsymbol{p}_{ij} \\ \boldsymbol{v}_k^{(o)} \end{bmatrix} + \boldsymbol{b}_d\right)$. Similar to the previously discussed intuition in Eq (9), $\boldsymbol{r}_{ijk}$ is a flexible combination between $\boldsymbol{p}_{ij}$ and each output context item embeddings $\boldsymbol{v}_k^{(o)}$; $square(\cdot)$ is an element-wise squared function. Our idea in Eq. (12), (13) is similar to the idea in Eq. (5), (6) of the SDP model. First, in Eq. (13), each context item k will attentively contribute to the target item j via a squared Euclidean measure. Second, in Eq. (12), each non-negative dimension in e_{ij} will be considered as a distance dimension and we use an edge-weights layer $\boldsymbol{w}_e$ to combine them flexibly. When there is only one context item in $\mathcal{T}_j^i$, then in Eq. (13), the attention score $\boldsymbol{a}_{ijk}$=1.0, leading to $\boldsymbol{e}_{ij} = square(\boldsymbol{r}_{ijk})$, which is similar to Eq. (5). In this case, SDM will measure the distance between target item j and context item k in the same way as SDP model does. Note that Eq. (13) is similar to Eq. (6) so SDM can also learn a signed distance value, which also provides a more fine-grained distance compared to a general distance value.

4.2.5 Multi-hop SDM:. Inspired by previous work [48] where the multi-hop design helped to refine the attention module in Memory Network, we also integrate multiple hops to further extend our SDM model to build a deeper network (Figure 4). As the gated multi-hop

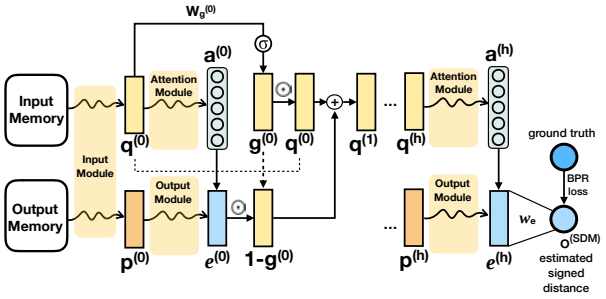


Figure 4: The illustration of our multi-hop SDM.

[2] https://en.wikipedia.org/wiki/Metric_(mathematics)

design [28] was shown to perform better than the original multi-hop design with a simple residual connection in [48], we employ this gated memory update from hop to hop as follows:

$$\boldsymbol{g}^{(h-1)} = \sigma(\mathbf{W}_g^{(h-1)}\boldsymbol{q}^{(h-1)} + \boldsymbol{b}_g^{(h-1)}) \tag{14}$$

$$\boldsymbol{q}^{(h)} = (1 - \boldsymbol{g}^{(h-1)}) \odot \boldsymbol{e}^{(h-1)} + \boldsymbol{g}^{(h-1)} \odot \boldsymbol{q}^{(h-1)} \tag{15}$$

where $\boldsymbol{q}^{(h-1)}$ is the input query embedding as shown in Eq. (7) at hop $h-1$, $\mathbf{W}_g^{(h-1)}$ and bias $\boldsymbol{b}_g^{(h-1)}$ are hop-specific parameters, σ is the sigmoid function, $e^{(h-1)}$ is the output of Eq. (13) at hop $h-1$, $\boldsymbol{q}^{(h)}$ is the input query embedding at the next hop h. So the attention could be updated at hop h accordingly using $\boldsymbol{q}^{(t)}$ as follows:

$$\alpha_{ijk}^{(h)} = \frac{exp(-z_{ijk}^{(h)})}{\sum_{p \in \mathcal{T}_j{}^i} exp(-z_{ijp}^{(h)})}, \text{ where } z_{ijk}^{(h)} = \left\| f\left(\mathbf{W}_c^{(h)} \begin{bmatrix} \boldsymbol{q}_{ij}^{(h)} \\ \boldsymbol{v}_k^{(i)} \end{bmatrix} + \boldsymbol{b}_c\right)\right\|_2^2 \tag{16}$$

The multi-hop architecture with gated design further refines the attention for different users based on the previous output from hop to hop. Hence, if the final hop is h then the SDM model with h hops, denoted as *SDM-h*, will use $\boldsymbol{a}_{ij}^{(h)}$ to yield a final signed distance score as follows:

$$o_{ij}^{(SDM-h)} = \boldsymbol{w}_e^\top \boldsymbol{e}_{ij}^{(h)} + b_e^{(h)} \tag{17}$$

where $\boldsymbol{e}_{ij}$ is calculated as:

$$\boldsymbol{e}_{ij}^{(h)} = \sum_{k \in \mathcal{T}_j{}^i} \boldsymbol{a}_{ijk}^{(h)} \times square\left(f\left(\mathbf{W}_d^{(h)} \begin{bmatrix} \boldsymbol{p}_{ij}^{(h)} \\ \boldsymbol{v}_k^{(o)} \end{bmatrix} + \boldsymbol{b}_d^{(h)}\right)\right) \tag{18}$$

Weight constraints in multi-hop SDM model: To save memory, we use the global weight constraint in multi-hop SDM. Particularly, input memory $\boldsymbol{U^{(i)}}, \boldsymbol{V^{(i)}}$ and output memory $\boldsymbol{U^{(o)}}, \boldsymbol{V^{(o)}}$ are shared among different hops. All the weights are shared from hop to hop $\boldsymbol{W}_a^{(1)} = \boldsymbol{W}_a^{(2)} = ... = \boldsymbol{W}_a^{(h)}; \boldsymbol{W}_b^{(1)} = \boldsymbol{W}_b^{(2)} = ... = \boldsymbol{W}_b^{(h)}; \boldsymbol{W}_c^{(1)} = \boldsymbol{W}_c^{(2)} = ... = \boldsymbol{W}_c^{(h)}; \boldsymbol{W}_d^{(1)} = \boldsymbol{W}_d^{(2)} = ... = \boldsymbol{W}_d^{(h)}$; and so do all bias terms. The gate weights are also global weights: $\boldsymbol{W}_g^{(1)} = \boldsymbol{W}_g^{(2)} = ... = \boldsymbol{W}_g^{(h)}$.

4.3 Signed Distance-based Deep Memory Recommender (SDMR)

Now we propose Signed Distance-based Deep Memory Recommender (SDMR), a hybrid network that combines SDP and SDM. The first approach to combine them is to employ a weighted summation of the output scores from SDP and SDM as follows:

$$o = \beta o^{(\text{SDP})} + (1 - \beta) o^{(\text{SDM})} \tag{19}$$

where $o^{(\text{SDP})}$ is the signed distance score obtained at Eq. (6), $o^{(\text{SDM})}$ is the signed distance score obtained at Eq. (17), and $\beta \in [0, 1]$ is a hyper-parameter to control the contribution of SDP and SDM. When β=0, SDMR becomes SDM. When β=1, SDMR becomes SDP.

However, to avoid tuning an additional hyper-parameter β, we do not use Eq. (19) for SDMR. Instead, we let SDMR self-learns the combination of SDM and SDM as follows:

$$o = ReLU\left(\boldsymbol{w}_u^\top \begin{bmatrix} \boldsymbol{e}^{(\ell+1)} \\ \boldsymbol{e}^{(h)} \end{bmatrix} + b_u\right) \tag{20}$$

where $\boldsymbol{e}^{(\ell+1)}$ is the final layer embedding from SDP and is obtained at Eq. (5), $\boldsymbol{e}^{(h)}$ is the final hop output from the multi-hop SDM obtained at Eq. (18). We note that SDP and SDM are first pre-trained separately using the BPR loss function (see the next section). Then, we obtain $\boldsymbol{e}^{(\ell+1)}$ from SDP, and $\boldsymbol{e}^{(h)}$ from SDM, and keep them fixed in Eq. (20) to learn $\boldsymbol{w}_u$ and b_u. We use ReLU in Eq. (20) because ReLU encourages sparse activations and helps to reduce over-fitting when combining the two components SDP and SDM.

4.4 Loss Functions

We adopt the Bayesian Personalized Ranking (BPR) as our loss function, which is similar to the idea of AUC (area under the curve):

$$\mathcal{L} = \underset{\theta}{\text{argmin}} \left(- \sum_{(u, i^+, i^-)} \log \sigma(o_{ui^-} - o_{ui^+}) + \lambda \|\theta\|^2 \right) \tag{21}$$

where we uniformly sample tuples in a form of (u, i^+, i^-) for user u with positive item (consumed) i^+ and negative item (unconsumed) i^-. λ is a hyper-parameter to control the regularization term, and $\sigma(\cdot)$ is the sigmoid function. Note that other pairwise probability functions could be plugged in Eq. (21) to replace $\sigma(\cdot)$. Both SDP and SDM are end-to-end differentiable since we uses soft attention over the output memory. Hence, we can utilize back-propagation to learn our models with stochastic gradient descent or Adam [18].

5 EMPIRICAL STUDY

We evaluate our SDP, SDM, SDMR models against ten state-of-the-art baselines in two recommendation tasks: (i) *general recommendation task*, and (ii) *shopping basket-based recommendation task*. We mainly aim to answer the following research questions (RQs):

- **RQ1:** How do SDP, SDM, and SDMR perform compared to other state-of-the-art models in both general recommendation task and shopping basket-based recommendation task?
- **RQ2:** Why/How does the multi-hop design help to improve the proposed models' performance?

5.1 Datasets

General recommendation task: In this task, we evaluate our proposed models and state-of-the-art methods using different datasets with various density levels as follows:

- **Movielens** [41]: It is a widely adopted benchmark dataset for collaborative filtering evaluation. We use two versions of this benchmark dataset, namely MovieLens100k (or ML-100k) and MovieLens1M (or ML-1M).
- **Netflix Prize** [3]: It is a real-world dataset collected by Netflix. This dataset was collected from 1999 to 2005, and consists of 463,435 users and 17,769 items with 56.9M of interactions. Since the dataset is extremely large, we subsample the Netflix dataset by randomly picking one-month data for evaluation.
- **Epinions** [35] [4]: It is an online rating dataset where users can share product feedback by giving explicit ratings and reviews.

In preprocessing preparation, we adopted a popular k-core preprocessing step [11, 25, 52] (with *k-core* = 5) to filter out inactive users with less than five ratings and items which are consumed

[3] https://www.netflixprize.com/
[4] http://www.trustlet.org/downloaded_epinions.html

Table 1: Statistics of the four datasets in the general recommendation task.

Statistics	ML-100k	ML-1M	Netflix	Epinions
# of users	943	6,040	1,888	23,137
# of items	1,682	3,706	3,724	23,585
# of interactions	100,000	1,000,209	103,254	461,982
Density (%)	6.3%	4.5%	1.5%	0.08%

Table 2: Statistics of the two real-world transactional datasets in the shopping basket-based recommendation task.

Statistics	IJCAI-15	Tafeng
# of users	2,433	22,851
# of items	4,534	22,291
avg # of items in a transaction	6.28	9.28
# of generated instances	15,422	523,653
Density (%)	0.14%	0.10%

by less than five users. Since ML-100k and ML-1M are already preprocessed, we only apply 5-core preprocessing step on the Netflix and Epinions datasets. We also binarize the rating scores as implicit feedback by converting all observed rating scores as positive interactions and the remaining as negative interactions. The statistics of the four datasets are summarized in Table 1.

Shopping basket-based recommendation task: We use two real-world transaction datasets as follows:

- **IJCAI-15** [5]: It consists of shopping logs of users from Tmall [6]. Since the original dataset is extremely large scale. We subsample IJCAI-15 by randomly picking 20k transactions for evaluation.
- **Tafeng** [7]: It is a grocery store transaction data. It contains four month transaction data from November 2000 to February 2001 by T-Feng supermarket.

Users in both IJCAI-15 and Tafeng datasets are logged under four types of actions: *click*, *add-to-cart*, *purchase*, and *add-to-favourite*. We consider all the four types as the *click* action. We only keep transactions with at least five items. This is because we will take one item out for testing, another item for development. In the remaining three items, one will be taken out as a target item and the two items will be used as the context items. Attentive scores will be assigned to the context items. In each of original transactions, we generate data instances of the format $< \mathbf{c}, v_c >$ where v_c is the target/predicting item and $\mathbf{c}$ is a set of all other items in the same transaction with v_c. In particular, in each transaction t, each time we pick one item out as a target item and leave the rest of items in t as corresponding context items. Subsequently, for each transaction t containing $|t|$ items, we can generate $|t|$ data instances. The statistics of the two transactional datasets are summarized in Table 2.

For an easy reference, we call (ML-100k, ML-1M, Netflix, Epinions) as *Group-1 dataset* and (IJCAI-15, Ta-Feng) as *Group-2 datasets*.

[5]https://tianchi.aliyun.com/datalab/dataSet.htm?id=1
[6]https://www.tmall.com
[7]http://stackoverflow.com/questions/25014904/download-link-for-ta-feng-grocery-dataset

5.2 Baselines and State-of-the-art Methods

We compared our proposed models against several strong baselines in the general recommendation task as follows:

- **Item**KNN [42]: It is an item neighborhood-based collaborative filtering method. It exploited cosine item-item similarities to produce recommendation results.
- **Bayesian Personalized Ranking (MF-BPR)** [39]: It is a state-of-the-art pairwise matrix factorization method for implicit feedback datasets. It minimizes $\sum_i \sum_{j^+,j^-} -log\sigma(u_i^T v_{j^+} - u_i^T v_{j^-}) + \lambda(||u_i||^2 + ||v_{j^+}||^2)$ where (u_i, v_{j^+}) is a positive interaction and (u_i, v_{j^-}) is a negative sample.
- **Sparse LInear Method (**SLIM**)** [36]: It learns a sparse item-item similarity matrix by minimizing the squared loss $||A - AW||^2 + \lambda_1||W|| + \lambda_2||W||^2$, where A is a $m \times n$ user-item interaction matrix and W is a $n \times n$ sparse matrix of aggregation coefficients of context items.
- **Collaborative Metric Learning (CML)** [15]: It is a state-of-the-art collaborative metric-based model that utilizes Euclidean distance to measure similarities between users and items. For fair comparison, we learn CML with BPR loss by minimizing $-\sum_{i,j^+,j^-} log(\sigma(||u_i - v_{j^-}||_2^2 - ||u_i - v_{j^+}||_2^2))$, where $||\cdot||_2^2$ is a squared Euclidean distance, (u_i, v_{j^+}) is a positive interaction and (u_i, v_{j^-}) is a negative sample.
- **Neural Collaborative Filtering (NeuMF++)** [12]: It is a state-of-the-art matrix factorization method using deep learning architecture. We use a pre-trained NeuMF to achieve its best performance, and denote it as NeuMF++.
- **Collaborative Memory Network (CMN++)** [7]: It is a state-of-the-art memory network based recommender. Its architecture follows traditional user neighborhood based collaborative filtering approaches. It adopts a memory network to assign attentive weights for other similar users.

Even though our approaches do not model the order of consumed items in the user's purchase history (e.g. rigid orders of items), since we consider latest s items as the context items to predict the next item, we still compare our models with some key sequential models to further show our models' effectiveness as follows:

- **Personalized Ranking Metric Embedding (PRME)** [8]: Given a user u, a target item j, and a previous consumed item k, it models a personalized first-order Markov behavior with two components: $d_{ujk} = \alpha||v_u - v_j||^2 + (1-\alpha)||v_k - v_j||^2$, where $||\cdot||_2^2$ is a squared $\mathcal{L}2$ distance. Then PRME is learned by minimizing BPR loss.
- **PRME_s:** It is our extension of PRME, where the distance between the target item j and the previous consumed item k is replaced by the average distance between j and each of previous s items: $d_{ujs} = \alpha||v_u - v_j||^2 + (1-\alpha)\frac{1}{|s|}\sum_{k \in s}||v_k - v_j||^2$. We use BPR loss to learn PRME_s.
- **Translation-based Recommendation (TransRec)** [9]: It uses first-order Markov and considers a user u as a translator of his/her previous consumed item k to a next item j. In another word, $prob(j|u,k) \propto \beta_j - d(u + v_k - v_j)$ where β_j is an item bias term, d is a distance function (e.g. $\mathcal{L}1$ or $\mathcal{L}2$ distance). We use $\mathcal{L}2$ distance because it was shown to

perform better than $\mathcal{L}1$ [9]. TransRec is then learned with BPR loss.

- **Convolutional Sequence Embedding Recommendation (Caser)** [49]: It is a state-of-the-art sequential model. It uses convolution neural network with many horizontal and vertical kernels to capture the complex relationships among items.

The strong sequential baselines above surpassed many other sequential models such as: TransRec outperformed FMC[40], FPMC [40], HRM [55]; Caser surpassed GRU4Rec [14] and Fossil [10], so we exclude them in our evaluation.

Comparison: In the general recommendation task, we compare our proposed models with all **ten** strong baselines listed above. In the shopping basket-based recommendation task, since the sequential models often work better than general recommendation-based models (see Table 3), we only compared our proposed models with sequential baselines. We name general recommendation baselines (i.e. ItemKNN, BPR, SLIM, CML, NeuMF++, CMN++) as *Group-1 baselines*, and call sequential baselines (i.e. PRME, PRME_s, TransRec, Caser) as *Group-2 baselines* for an easy reference.

5.3 Experimental Settings

Protocol: We adopt the widely used *leave-one-out* setting [12, 60], in which for each user, we reserve her last interaction as the test sample. If there are no timestamps available in the dataset, then the test sample is randomly drawn. Among the remaining data, we randomly hold one interaction for each user to form the development set, while all others are utilized as the training set. Since it is very time-consuming and unnecessary to rank all the unobserved items for each user, we follow the standard strategy to randomly sample 100 unobserved items for each user. Then, we rank them together with the test item [12, 19].

Assigning item orders: Sequential models need rigid orders of consumed items but consumed items in the same transaction (in IJCAI-15 and TaFeng datasets) are assigned the same timestamp of the transaction containing these items. Hence, we assigned the item timestamps where the orders of items are kept as in the original dataset. This may give credits to sequential models but not our methods (because our methods will use all consumed items in the same transaction as context items and do not model the item orders).

Hyper-parameters selection: We perform a grid search for the embedding size d from $\{8, 16, 32, 64, 128\}$ and regularization terms from $\{0.1, 0.01, 0.001, 0.0001, 0.00001\}$ in all the models. We select the best number of hops for CMN++ and our SDM from $\{1, 2, 3, 4\}$. In NeuMF++, we select the best number of MLP layers from $\{1, 2, 3\}$. In our models, we fix the batch size to 256. We adopt Adam optimizer [18] with a fixed learning rate of 0.001. Similar to CMN++ and NeuMF++, the number of negative samples is set to 4. We use one layer perceptron for SDP (more complex datasets may need more than one layer to get better results). In the four datasets used in general recommendation task (e.g ML-100k, ML-1M, Netflix, Epinions), to avoid too many *zero paddings* for users with a smaller number of consumed items or too many context items are kept in the memory, which unnecessarily slow down the model's execution, we follow [50] to limit the number of context items using latest s consumed items. We search s in {5, 10, 20}. In the two shopping basket-based recommendation datasets (i.e. IJCAI-15 and TaFeng), since the maximum number of items in a transaction is small (e.g. 13 in IJCAI-15, and 18 in TaFeng), we consider all the other items in the same transaction with the target item as its context items. All the hyper-parameters are tuned using the development dataset. Our source code is available at: *https://github.com/thanhdtran/SDMR*.

Evaluation Metrics: We evaluate all models' performance by two widely used metrics: Hit Ratio (*hit@k*), and Normalized Discounted Cumulative Gain (NDCG@k), where k is a truncated number or *top-k* item recommendation. Intuitively, *hit@k* shows whether the test item is in the *top-k* list or not, while *NDCG@k* accounts for the position of the hits by assigning higher scores to the hits at top ranks and downgrading the scores to hits by log_2 at lower ranks.

5.4 Experimental Results

RQ1: Overall results in general recommendation task: The performance of our proposed models and the baselines are shown in Table 3. First, we observe that SDP significantly outperformed BPR in all four datasets in *Group-1 datasets*, improving *hit*@10 from 8.33~41.19%, and NDCG@10 from 10.44~44.56%. Although SDP and BPR shared the same loss function, the difference between them is SDP measured a signed distance score between a target user and a target item via a MLP which modeled a non-linear interaction between them, while BPR used Matrix Factorization with inner product. This result confirms the effectiveness of using signed distance based similarity over inner product in the general recommendation task. Second, we compare SDP with CML. CML worked by trying to minimize the squared Euclidean distance scores between target users and target items. Our SDP, in another hand, works by minimizing signed distance scores of non-linear interactions (via non-linear activation functions) between target users and target items. We observe that SDP performed better than CML in all *Group-1 datasets*, improving *hit*@10 from 8.33~11.19%, and NDCG@10 from 7.06~12.24%. On average, SDP improved *hit*@10 by 7.5% and *NDCG*@10 by 9.5% compared to CML. Our SDP even gain competitive results compared to NeuMF++ and CMN++. On average, SDP is just slightly worse than NeuMF++ and CMN++ by -2.67% for *hit*@10, and -1.68% for *NDCG*@10. All of these results show the effectiveness of using signed distance in our SDP model.

Next, we compare SDM with neighborhood-based baselines. Both SLIM and item-KNN used previously consumed items of a user to make the prediction for the next item. SDM significantly outperformed both baselines, improving *hit*@10 from 20.53~130.92% and NDCG@10 from 39.05~106.35% compared with SLIM. It is an obvious result because the neighborhood-based baselines barely measured linear similarities between the target item and the user's consumed items. In contrast, our SDM produced signed distance scores and assigned personalized metric-based attention weights to each of consumed items that contribute to the target item.

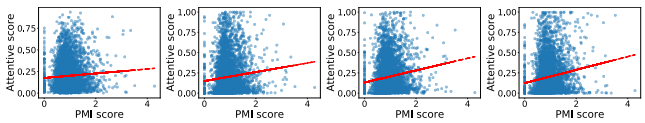
We then compare SDM with CMN++ and NeuMF++. SDM outperformed CMN++ in all *Group-1 datasets*, improving *hit*@10 from 11.71~35.93% and NDCG@10 from 26.51~43.38%. On average, it improves *hit*@10 by 18.63% and *NDCG@10* by 32.84% compared to CMN++. This result shows the effectiveness of our personalized metric-based attention with signed distance and item-based neighborhood design over the traditional inner product-based attention

Table 3: General Recommendation Task: Overall performance of the baselines, and our proposed SDP, SDM, and SDMR on four datasets. The last four lines show the relative improvement of the SDM and SDMR over the best baseline method in General Recommenders (Group 1) and Sequential Recommenders (Group 2), respectively.

Method type	Method	ML-100k		ML-1M		Netflix		Epinions	
		hit@10	NDCG@10	*hit*@10	NDCG@10	*hit*@10	NDCG@10	*hit*@10	NDCG@10
General Recommenders (Group 1)	Item-KNN	0.166	0.073	0.235	0.110	0.039	0.019	0.121	0.096
	SLIM	0.520	0.298	0.677	0.420	0.358	0.212	0.249	0.189
	MF-BPR	0.554	0.316	0.595	0.352	0.352	0.193	0.384	0.232
	CML	0.596	0.326	0.662	0.390	0.447	0.254	0.376	0.237
	NeuMF++	0.623	0.341	0.716	0.438	0.509	0.279	0.428	0.274
	CMN++	0.620	0.344	0.729	0.442	0.523	0.293	0.423	0.272
Sequential Recommenders (Group 2)	PRME	0.638	0.381	0.724	0.486	0.509	0.329	0.538	0.346
	PRME_s	0.674	0.398	0.734	0.491	0.539	0.348	0.380	0.244
	TransRec	0.684	0.402	0.770	0.524	0.511	0.345	0.551	0.357
	Caser	0.674	0.386	**0.826**	0.606	0.480	0.253	0.326	0.268
Ours	SDP	0.616	0.349	0.694	0.424	0.497	0.279	0.416	0.266
	SDM	**0.713**	0.435	**0.816**	0.584	**0.584**	0.379	**0.575**	0.390
	SDMR	**0.695**	**0.562**	**0.810**	**0.662**	**0.592**	**0.449**	**0.568**	**0.423**
Compared to Group 1	Imprv. of SDM	14.54%	26.51%	11.93%	32.13%	11.71%	29.32%	34.35%	42.34%
	Imprv. of SDMR	11.65%	63.44%	11.11%	49.77%	13.24%	53.20%	32.71%	54.38%
Compared to Group 2	Imprv. of SDM	4.24%	8.21%	-1.21%	-3.63%	8.35%	8.91%	4.36%	9.24%
	Imprv. of SDMR	1.61%	39.80%	-1.94%	9.24%	9.83%	29.02%	3.09%	18.49%

Table 4: Shopping basket-based Recommendation Task: Overall performance of the baselines, and our proposed models on two datasets. The last two lines show the relative improvement of the SDM and SDMR over the best baseline.

Method	IJCAI-15		Ta-Feng	
	hit@10	NDCG@10	*hit*@10	NDCG@10
PRME	0.276	0.177	0.594	0.365
PRME_s	0.229	0.133	0.590	0.355
TransRec	0.262	0.168	0.622	0.401
Caser	0.173	0.096	0.605	0.373
SDP	0.323	0.201	0.633	0.401
SDM	0.316	0.189	**0.646**	0.439
SDMR	**0.336**	**0.222**	0.627	**0.559**
Imprv. of SDM	14.49%	6.78%	3.86%	9.48%
Imprv. of SDMR	21.74%	25.42%	0.80%	39.40%



(a) Hop 1. **(b) Hop 2.** **(c) Hop 3.** **(d) Hop 4.**

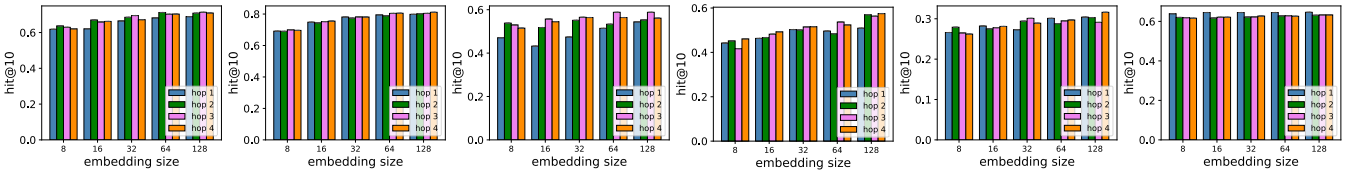
Figure 5: ML-100K: Scatter plots of PMI scores and attentive scores generated by SDM with h hops (h={1, 2, 3, 4} from left to right). The red lines are the linear trend lines. The Pearson correlation between two scores increases when h increased.

in a user-based neighborhood design in CMN++. SDM also outperformed NeuMF++, improving *hit*@10 from 13.97~34.35%, and NDCG@10 from 27.42~42.34%. On average, in all *Group-1 datasets*, SDM outperformed all the baselines in "General Recommenders" (Group 1), improved *hit*@10 by 18.13% and *NDCG*@10 by 32.58% compared to the best baseline in *Group 1*.

Finally, we look at the performance of SDMR model, which is the proposed fusion of SDP and SDM. Compared to SDM, our SDMR insignificantly downgrades SDM on *hit*@10 measurement with a very small amount, but it does help a lot in refining the ranking of items and boosting *NDCG*@10 results. As shown in Table 3, SDMR improved from 8.46~29.20% for *NDCG*@10, and by 17.37% for *NDCG*@10 on average compared to SDM in *Group-1 datasets*. SDMR also surpassed all the methods in *Group 1*. On average, SDMR improved *hit*@10 by 17.18% and NDCG@10 by 55.20% compared to the best model in *Group 1*.

We also compared our models with some strong sequential models in Table 3. Sequential models exploited consuming time of items and model their rigid orders, which often lead to a much improved performance compared to general recommendation models in *Group-1 baselines*. As such, compared to the best sequential baseline model, on average, SDM improves *hit*@10 by 3.94% and *NDCG*@10 by 5.68% , and SDMR improves *hit*@10 by 3.15% and *NDCG*@10 by 24.14% compared to the best sequential model reported in Table 3.

Overall results in shopping basket-based recommendation task: Table 4 shows the performance of our models and sequential baselines in *Group-2 datasets*. Again, our models outperformed all the sequential baselines. On average, SDM improved *hit*@10 by 9.2% and *NDCG*@10 by 8.1%, SDMR improved *hit*@10 by 11.3% and *NDCG*@10 by 32.4% compared to the best reported baseline.



(a) ML-100K. (b) ML-1M. (c) Netflix. (d) Epinions. (e) IJCAI-15. (f) TaFeng.

Figure 6: Comparison of varying the number of hops regarding different embeddings sizes in the six datasets.

5.5 RQ2: Understanding our multi-hop personalized metric-based attention design?

In the previous section, we see that our models outperformed many strong baselines in **six** different datasets of the **two** different recommendation problems. In this part, we explore why did we achieve those better results? As "attention is all you need" [54], the core reason brought us an surpassed performance accredit to the metric-based attention which are further refined via multi-hop design. Therefore, we want to explore quantitatively and qualitatively how our attention with multi-hop design worked by answering two smaller research questions: (i) what did our metric-based attention with multi-hop design learn?, (ii) did the metric-based attention with multi-hop design improve recommendation results? Without a special mention, since our SDMR model just learned a combination between SDP and SDM without re-learning the learned-already parameters in SDP and SDM, we explore SDM in this section to understand how attention with multi-hop design works. Note that we conduct this analysis for ML-100k only due to space limitation and the availability of movies genre in ML-100k (for visualization in Figure 7).

What did our metric-based attention with multi-hop design learn? To answer this question, we first measure the *point-wise mutual information* (PMI) between two certain items j and k as:

$$PMI(j,k) = log\frac{P(j,k)}{P(j) \times P(k)} \tag{22}$$

where $P(j,k)$ is the joint probability between two items j and k, which shows how likely j and k are co-preferred ($P(j,k) = \frac{\#(j,k)}{|D|}$, where D denotes a collection of all item-item pairs, and $|D|$ is the total number of item-item co-occurrence pairs in D). Similarly, $P(j)$ and $P(k)$ are the probabilities of the item j and k in D, respectively (e.g. $P(j) = \frac{\#(j)}{|D|}$, $P(k) = \frac{\#(k)}{|D|}$). Intuitively, a PMI score between two items shows how likely the two items are co-purchased/co-preferred. The higher the PMI score between j and k is, the more likely the user will purchase j if k was purchased before.

We denote SDM-h is the SDM model with h hops. Now, given a target item j and the user's context items k, *SDM-h* will assign attentive scores for all (j,k) pairs. We also get PMI scores (from Eq. (22)) of (j,k) pairs. Next, we plot a scatter plot of PMI scores and attentive scores for all (j,k) pairs to see the relationship between the two scores. Our results for ML-100k dataset is shown in Figure 5.

In Figure 5, the Pearson correlation between PMI scores and attentive scores are 0.059, 0.097, 0.143, and 0.146 for SDM-1, SDM-2, SDM-3 SDM-4, respectively. It indicates that as we increase the number of hops in SDM model, PMI scores and attentive scores are more positively correlated. In another word, as we increase the number of hops, our metric-based attention with multi-hop design will assign higher weights for co-purchased items, which is what we desire. Furthermore, scatter plots in Figure 5a presents that there is a high density of points with small attentive scores. This indicates that attention in SDM-1 is distributed to several items (which is somewhat close to equally focusing on context items). However, when we increase the number of hops h, the density spreads up to the top, indicating that the model tends to give a higher attention to some context items, which can be more relevant than others. This observation is consistent with "learning to attend" in [2, 59].

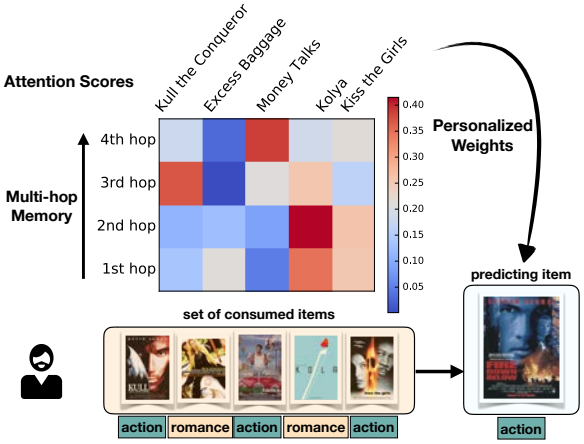


Figure 7: Multi-hop Attention visualization.

Did the metric-based attention with multi-hop design improve recommendation results? We answer this research question by showing the results of SDM model when varying number of hops h from {1, 2, 3, 4} with different embedding sizes and visualize attention scores of SDM-h with a random observation as follows:

Varying number of hops with different embedding sizes: The performance of SDM-h regarding *hit*@10 with h from {1, 2, 3, 4} and embedding size from {8, 16, 32, 64, 128} is presented in Figure 6. We see that more hops tend to give additional improvement in all 6 datasets, except in Tafeng dataset where SDM with more hops over-fitted. In ML-100k and ML-1M, the optimal number of hops are 3 or 4. In Netflix, SDM with 3 hops performed well. In Epinions and IJCAI-15, SDM-4 tends to achieve better results. Overall, the selection of the number of hops depends on the dataset complexity, and it varies from datasets to datasets.

Attention Visualization: Lastly, to visualize how the personalized metric-based attention with multi-hop design works, we chose one user from ML-100K data. The learned weights at each hop of SDM is shown in Figure 7. The target item in this example is an action movie called *Fire Down Below* (1997). The first two hops of SDM

assigned high weights to two romance movies, and the lowest score to the action movie *Money Talks* (1997). The 3rd-hop and 4th-hop attention refined the weights of movies to better reflect the correlations and similarities *w.r.t* the target movie. At last, *Money Talks* (1997) was assigned with the highest weight 0.386, and the total weights of two romance movies decreased to less than 0.2. This result shows the effectiveness of our multi-hop SDM model.

6 CONCLUSION

In this paper, we have studied the top-k recommendation problem in a signed distance learning perspective. Different from previous works, we have considered two independent signed distance models for measuring user-item and item-item similarities respectively via deep neural networks. Extensive experiments have been performed on six real-world datasets in general recommendation and shopping basket-based recommendation task. We presented that our proposed SDMR outperformed **ten** baselines in all two recommendation tasks. To an extension, future works can integrate position embeddings [54] in our models, which give our models a sense of which position they are dealing with, which can further improve our model's performance when rigid orders of items are available.

ACKNOWLEDGMENT

This work was supported in part by NSF grant CNS-1755536, Google Faculty Research Award, Microsoft Azure Research Award, AWS Cloud Credits for Research, and Google Cloud. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors.

REFERENCES

[1] Charu C Aggarwal. 2016. *Recommender systems*. Springer.
[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).
[3] Daniel Billsus and Michael J Pazzani. 2000. User modeling for adaptive news access. *User modeling and user-adapted interaction* 10, 2-3 (2000), 147–180.
[4] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 335–344.
[5] Heeyoul Choi, Kyunghyun Cho, and Yoshua Bengio. 2018. Fine-grained attention mechanism for neural machine translation. *Neurocomputing* 284 (2018), 171–176.
[6] Mukund Deshpande and George Karypis. 2004. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems* 22, 1 (2004), 143–177.
[7] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative Memory Network for Recommendation Systems. In *Proceedings of the 41st ACM International Conference on Research and Development in Information Retrieval*.
[8] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized Ranking Metric Embedding for Next New POI Recommendation. In *International Joint Conference on Artificial Intelligence*, Vol. 15. 2069–2075.
[9] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 161–169.
[10] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. 191–200.
[11] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. 507–517.
[12] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.
[13] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 549–558.
[14] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).
[15] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative metric learning. In *Proceedings of the 26th International Conference on World Wide Web*. 193–201.
[16] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. 263–272.
[17] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 233–240.
[18] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[19] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 426–434.
[20] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 447–456.
[21] Yehuda Koren. 2010. Collaborative filtering with temporal dynamics. *Commun. ACM* 53, 4 (2010), 89–97.
[22] Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *International Conference on Machine Learning*. 331–339.
[23] Neil D Lawrence and Raquel Urtasun. 2009. Non-linear matrix factorization with Gaussian processes. In *Proceedings of the 26th Annual International Conference on Machine Learning*. 601–608.
[24] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep collaborative filtering via marginalized denoising auto-encoder. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 811–820.
[25] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. (2018).
[26] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130* (2017).
[27] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 1 (2003), 76–80.
[28] Fei Liu and Julien Perez. 2017. Gated end-to-end memory networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Vol. 1. 1–10.
[29] Qiang Liu, Shu Wu, and Liang Wang. 2017. DeepStyle: Learning user preferences for visual recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 841–844.
[30] Xinyue Liu, Chara Aggarwal, Yu-Feng Li, Xiaugnan Kong, Xinyuan Sun, and Saket Sathe. 2016. Kernelized matrix factorization for collaborative filtering. In *Proceedings of the 2016 SIAM International Conference on Data Mining*. 378–386.
[31] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Coevolutionary Recommendation Model: Mutual Learning between Ratings and Reviews. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. 773–782.
[32] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Convolutional Matrix Factorization for Recommendation Explanation. In *Proceedings of the 23rd International Conference on Intelligent User Interfaces Companion*. 34.
[33] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).
[34] Chen Ma, Peng Kang, Bin Wu, Qinglong Wang, and Xue Liu. 2019. Gated Attentive-Autoencoder for Content-Aware Recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 519–527.
[35] Paolo Massa and Paolo Avesani. 2007. Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*. 17–24.
[36] Xia Ning and George Karypis. 2011. Slim: Sparse linear methods for top-n recommender systems. In *2011 11th IEEE International Conference on Data Mining*. 497–506.
[37] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 130–137.
[38] Parikshit Ram and Alexander G Gray. 2012. Maximum inner-product search using cone trees. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 931–939.
[39] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. 452–461.

[40] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. 811–820.

[41] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. 175–186.

[42] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[43] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*. 111–112.

[44] Paul Hongsuck Seo, Zhe Lin, Scott Cohen, Xiaohui Shen, and Bohyung Han. 2016. Hierarchical attention networks. *arXiv preprint arXiv:1606.02393* (2016).

[45] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 297–305.

[46] Anshumali Shrivastava and Ping Li. 2014. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In *Advances in Neural Information Processing Systems*. 2321–2329.

[47] Nitish Srivastava and Ruslan R Salakhutdinov. 2012. Multimodal learning with deep boltzmann machines. In *Advances in Neural Information Processing Systems*. 2222–2230.

[48] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems*. 2440–2448.

[49] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 565–573.

[50] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent relational metric learning via memory-based attention for collaborative ranking. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. 729–739.

[51] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Multi-Pointer Co-Attention Networks for Recommendation. *arXiv preprint arXiv:1801.09251* (2018).

[52] Thanh Tran, Kyumin Lee, Yiming Liao, and Dongwon Lee. 2018. Regularizing Matrix Factorization with User and Item Embeddings for Recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 687–696.

[53] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep content-based music recommendation. In *Advances in Neural Information Processing Systems*. 2643–2651.

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 6000–6010.

[55] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *ACM SIGIR Conference on Research and Development in Information Retrieval*. 403–412.

[56] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent recommender networks. In *Proceedings of the tenth ACM international conference on web search and data mining*. 495–503.

[57] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. 153–162.

[58] Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *International Conference on Machine Learning*. 2397–2406.

[59] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*. 2048–2057.

[60] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep Matrix Factorization Models for Recommender Systems. In *Proceeding of the 26th International Joint Conference on Artificial Intelligence*. 3203–3209.

[61] Hanwang Zhang, Yang Yang, Huanbo Luan, Shuicheng Yang, and Tat-Seng Chua. 2014. Start from scratch: Towards automatically identifying, modeling, and naming visual attributes. In *Proceedings of the 22nd ACM International Conference on Multimedia*. 187–196.

[62] Tinghui Zhou, Hanhuai Shan, Arindam Banerjee, and Guillermo Sapiro. 2012. Kernelized probabilistic matrix factorization: Exploiting graphs and side information. In *Proceedings of the 2012 SIAM International Conference on Data Mining*. 403–414.

Rating Worker Skills and Task Strains in Collaborative Crowd Computing: A Competitive Perspective

George Trimponias*
Huawei Noah's Ark Lab
Hong Kong

Xiaojuan Ma*
Hong Kong University of Science and Technology
Hong Kong

Qiang Yang
Hong Kong University of Science and Technology
Hong Kong

ABSTRACT

Collaborative crowd computing, e.g., human computation and crowdsourcing, involves a team of workers jointly solving tasks of varying difficulties. In such settings, the ability to manage the workflow based on workers' skills and task strains can improve output quality. However, many practical systems employ a simple additive scoring scheme to measure worker performance, and do not consider the task difficulty or worker interaction. Some prior works have looked at ways of measuring worker performance or task difficulty in collaborative settings, but usually assume sophisticated models. In our work, we address this question by taking a competitive perspective and leveraging the vast prior work on competitive games. We adapt TrueSkill's standard competitive model by treating the task as a fictitious worker that the team of humans jointly plays against. We explore two fast online approaches to estimate the worker and task ratings: (1) an ELO rating system, and (2) approximate inference with the Expectation Propagation algorithm. To assess the strengths and weaknesses of the various rating methods, we conduct a human study on Amazon's Mechanical Turk with a simulated ESP game. Our experimental design has the novel element of pairing a carefully designed bot with human workers; these encounters can be used, in turn, to generate a larger set of simulated encounters, yielding more data. Our analysis confirms that our ranking scheme performs consistently and robustly, and outperforms the traditional additive scheme in terms of predicted accuracy.

CCS CONCEPTS

• **Human-centered computing → HCI design and evaluation methods**; **Collaborative interaction**; **HCI theory, concepts and models**; *Collaborative and social computing*.

KEYWORDS

Human computation; crowdsourcing; ranking, rating; collaborative.

ACM Reference Format:
George Trimponias, Xiaojuan Ma, and Qiang Yang. 2019. Rating Worker Skills and Task Strains in Collaborative Crowd Computing: A Competitive Perspective. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313569

*Equal contributions.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313569

1 INTRODUCTION

Human computation [38] and crowdsourcing [14] leverage the human processing power of online crowds to solve tasks that are beyond the capabilities of computer algorithms. Even though most of these tasks involve independent human workers, a significant portion of them requires the collaborative effort of two or more workers. One representative example is multi-player human computation games [40], originally proposed by Luis von Ahn as a money-free alternative to incentivize humans to perform such diverse tasks as image annotation [39], music tagging [27], collecting common sense facts [41], or locating objects in images [42]. These games usually require the agreement of the players[1] for data verification. Collaboration thus serves as a quality control mechanism. For example, in the ESP game [39], two randomly paired-up players simultaneously provide labels for a given image without communicating one with another. Collaboration also occurs in highly complex crowdsourcing tasks such as multilingual translation, to exploit complementary knowledge from different workers to generate high quality results [23, 32].

The performance of a joint task mainly depends on three factors. The first is task difficulty, which intuitively measures how hard it is for workers to generate outcomes intended by requesters. For example, in the ESP Game, a picture of a lion would be easy to name, since the majority of players are able to give specific labels, e.g., "lion". In contrast, an image of a tapir is more challenging, given that most people can only identify it as an "animal". The second factor is the skill of individual workers, which can be affected by their education, personal experience, etc. Normally, skilled workers are able to generate outputs of a higher quality compared to the low-skilled ones. The third, also the most distinctive factor of collaborative crowd computing, concerns the inter-worker interaction, i.e., the mechanism to determine the collective output given individual contributions. For instance, several human computation games rely on the agreement between two or more players as a verification mechanism. As a result, low-skilled players may have a greater impact on the game outcome. In the tapir image example, a knowledgeable player may key in the exact name of the tropical mammal, which, however, will not be accepted by the game if the partner fails to recognize it. Such crowd computing tasks can be best characterized as weak-link games [13]. The dual type, strong-link interactions in which more skilled workers may have a greater impact on the group outcome, also occurs in collaborative crowd computing. Crowd translation for instance only takes the

[1] Whenever there is no ambiguity, we use the terms worker and human interchangeably. In game contexts, we additionally use all three terms worker, player, and human interchangeably.

best translation derived from various answers generated by a team of workers [23].

To achieve the desired output quality and maintain worker engagement in collaborative crowd computing, one approach is to rate human participants based on their skill and tasks based on their difficulty. First, a rating system gives workers the incentive to devote more efforts to the tasks for potentially higher monetary rewards. This can, in turn, lead to better outcomes. Second, it allows for balanced collaborations, teaming up workers of similar skill levels. Third, it enables more proper task assignments, i.e., workers receive tasks that match their skills. Existing crowd computing projects often use an additive scoring scheme to assess worker performance [27, 39, 41]. Upon the completion of a collaborative task, each member of the group receives a score according to the joint performance. A project can rank all its workers based on their total scores summed over all the tasks they have performed, possibly in different groups. This scoring scheme is not an accurate indicator of personal skills, as a low-skilled worker that finishes significantly more tasks than a more skillful counterpart may probably end up with a higher score. Taking the average score can avoid this issue, but it still suffers from two significant shortcomings as a performance measure. First, it does not take into account the task difficulty, only the outcome. Intuitively, a high-quality output of a challenging task should be valued more than that of an easy one. Second, it ignores worker interaction, which can be a critical factor. Motivated by its pivotal role and the aforementioned shortcomings, some prior work has looked at ways of jointly rating workers and tasks in collaborative crowd computing [4, 7, 11, 20, 21, 33, 34, 36, 43–45]. These works typically assume fairly sophisticated models; the ratings of workers and/or tasks are subsequently inferred by maximizing a proper log-likelihood function.

In this work, we adopt an alternative perspective: instead of coming up with new schemes, we seek to leverage the state-of-the-art work on competitive games. Our model is the standard Bayesian rating scheme of TrueSkill for competitive games [12], which additionally treats the task as a fictitious worker that the team of humans plays against. We further discuss two online methods to estimate the ratings under this model: (1) an ELO rating scheme [9] that iteratively updates the ratings based on the expected and the actual outcome; and (2) Expectation Propagation (EP) [30, 31], a message-passing algorithm that performs approximate Bayesian inference. We evaluate the efficacy of the proposed rating framework using data from a simulated ESP Game with 100 participants conducted on Amazon Mechanical Turk (MTurk). We empirically show that our framework is more robust than the average scoring scheme, with the ELO scheme having the most consistent performance. We further assess the strengths and weaknesses of all methods. Our contributions can be summarized as follows:

- We show how to leverage TrueSkill's standard rating model in competitive games for the purpose of rating participants and tasks in collaborative projects. In this direction, we derive, for the first time, the update equations for the online schemes of ELO and EP in the collaborative scenario.
- We perform a human study on MTurk by properly adjusting the ESP game. Our study allows us to empirically assess the strengths and weaknesses of our proposed methods to the average scoring scheme, commonly employed in several human computation projects.
- We propose a novel experimental design, where we pair a human player with a carefully designed bot instead of a human player. This allows us, in turn, to generate a large set of encounters from a smaller dataset at a low cost.

The rest of the paper is structured as follows. We first survey related work. We then introduce our Bayesian TrueSkill-based rating model, and explore worker and task rating estimation via (1) a simple ELO rating scheme, and (2) an Expectation Propagation message passing algorithm. Following that, we present our human study on MTurk and the experimental results. Finally, we conclude the paper with discussions and future work.

2 RELATED WORK

We review existing scoring schemes in collaborative crowd computing, as well as rating and ranking in competitive games.

2.1 Scoring in Collaborative Crowd Computing

Additive scoring is the most common way to assess worker performance in existing crowd computing projects. In the ESP game for instance, a match results in a score varying from 50 to 150, with more specific labels rewarding a higher score. Similarly, in TagATune for tagging music clips [27] and Verbosity for collecting commonsense facts [41], both players earn 1 point if they reach an agreement or 0 points otherwise. These games order players according to their total scores across all the tasks they have performed, and thus the number of jobs completed may outweigh individual skill in the ranking. Computing the average score per task can avoid this issue, but still fails to consider the influence of task difficulty and intra-group dynamics. In contrast, although under a different setting, the more complicated schemes for ranking or rating players proposed for conventional competitive games may provide useful insights for collaborative crowd computing.

2.2 Ranking and Rating in Competitive Games

Assume a game with a finite set of players. A ranking is defined as a total preorder $\preceq$ on the set of players. For two players i, j, the relationship $i \preceq j$ denotes that player i ranks at most as high as player j. A rating, on the other hand, is a quantitative assessment index that denotes the skill level of a player. Higher-skilled players are associated with higher ratings in contrast to those with lower skills. Compared to rating, ranking merely shows that one player is better than another, but how much better remains unknown.

For ranking, there are three main approaches [3]. (1) The model-based approach determines the player ratings and subsequently calculates the induced ranking by sorting the computed ratings. (2) The score-based approach measures player performance with a scoring scheme (e.g., total or average score), and ranks players based on their score. (3) The axiomatic approach is inspired by axiomatic economic theory [35]. The idea is to determine a ranking scheme based on a set of axioms that must be satisfied.

Model-based rating methods usually view skill as a latent variable that characterizes the players. They assume that a fixed probability distribution (model) governs the outcome of an encounter between two players of given skills [3]. In particular, $F(r_i, r_j)$ denotes the

probability that player i with skill r_i beats player j with skill r_j; note that $F(r_i, r_j) = 1 - F(r_j, r_i)$. The probability distribution is computed a priori based on historical data. It primarily follows two models: the Thurstone Case V model [37] assumes the Gaussian distribution $F(r_i, r_j) = \Phi(r_i - r_j)$, where $\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$, whereas the Bradley-Terry model [2] assumes the logistic variant $F(r_i, r_j) = \frac{1}{1+e^{-(r_i - r_j)}}$. Subsequently, one can use a maximum likelihood approach to estimate the players' skills based on paired-comparison data, and rank them accordingly. In practice, the skill parameters are called ratings, and the function F is the rating function. Under the linear paired comparison model [6], we assume that the result of an encounter between any two players depends only on their rating difference.

Given that maximum likelihood estimation is complex and requires the complete paired-comparison data, an alternative is the ELO rating system [9]. Under this online scheme, every player is associated with an ELO rating. Whenever an encounter between two players occurs, the scheme computes the probability that each of the two players wins. It subsequently rewards or punishes players, depending on whether they performed better or worse than expected. The ELO rating system has been widely deployed for player ranking in real-world competitive games such as chess, association football, American football, basketball, Major League Baseball, Scrabble, and snooker.

The above rating methods do not directly handle the uncertainties over the ratings. Hence, prior research has proposed Bayesian rating schemes to tackle this [10]. The idea is to model the belief about a player's skill as a Gaussian belief distribution characterized by a mean μ and a standard deviation σ. Recently, TrueSkill has been introduced as a more general online Bayesian rating system which can further handle (1) teams of players rather than just individual players, and (2) settings where more than two teams compete simultaneously [12].

Of course, the topic of ranking and rating from pairwise comparisons with items of various difficulties has attracted considerable attention in the machine learning community, see e.g., [4, 7, 11, 20, 21, 33, 34, 36, 43–45]. In particular, several sophisticated studies within the machine learning community investigate the problem from a theoretical perspective. In this work, we take a different approach by looking into fast online schemes inspired by competitive games that are relatively simple to implement and deploy. Our interest in TrueSkill is due to its enormous success in both identifying and tracking the skills of gamers in Xbox Live by Microsoft, and its ability to properly match gamers into competitive matches. Our goal is thus to examine how the most popular and successful rating schemes in competitive games can be adapted to ameliorate ranking in collaborative crowd computing in a fast online manner and empirically assess them, rather than investigate new theory and propose novel sophisticated ranking schemes.

Finally, note that there has been extensive research within the crowdsourcing community aimed at improving worker performance, including different filtering mechanisms, the use of golden standards, explicitly verifiable questions as well as gamification with scoring mechanisms aimed at improving data quality [1, 8, 15, 24, 25, 29]. Such work is complementary to ours, since our sole focus in on schemes for jointly rating the workers' skills and task strains.

3 BAYESIAN RATING MODEL

In this section, we introduce a TrueSkill-based Bayesian rating model and two online rating estimation methods for assessing works and tasks in collaborative crowd computing. Our model takes into account both the worker interaction and the task difficulty.

3.1 High-level Overview

Before examining our model in detail, we first provide a high-level overview. Since both the worker skills and the task strains are unknown, we represent them as random variables, which are characterized by two important parameters, the mean and the variance. Higher values for the former are associated with a belief for a higher worker skill (or task strain) compared to lower values. On the other hand, the variance describes the degree of certainty in our belief, with lower values corresponding to higher certainty.

Given the random variables, a model describes how the various variables are related to each other. In collaborative tasks, the model has in general two parts: (1) the first part describes how the performance of the team of workers depends on their individual skills; (2) the second part describes when the team of workers solves a task of a given strain successfully or not.

Since the random variables are unknown, our goal is to then infer their values given the model from a set of encounters, where teams of workers try to solve a task. There are two general families of methods. Online methods update the random variables after each encounter, whereas offline methods do the update based on the entire dataset. Offline methods can in principle lead to optimal estimates, but online methods have the advantage of being faster and simpler. Practical rating schemes for competitive games have traditionally relied on online methods (see section 2.2), and we take a similar approach in our work.

3.2 Model

Our scheme is inspired by TrueSkill [12] and by prior work on question difficulty estimation in community question answering services [28].

Let i and j be the two workers[2] with ratings r_i and r_j, respectively, that collaboratively try to solve human computation task t. We denote the encounter between i and j when they perform task t as $< i, j, t >$. TrueSkill assumes that the belief about the rating r_i of player i follows a normal distribution $\mathcal{N}(r_i; \mu_i, \sigma_i^2)$ with mean μ_i and variance σ_i^2. When i performs a task t, its performance p_i will follow a normal distribution $\mathcal{N}(p_i; r_i, \delta^2)$. The variance δ^2 is due to the fact that the performance of a player may vary due to assorted factors that we cannot directly model. When players i and j collaboratively perform task t, their joint performance p will depend on the type of interaction between them. As we argued in the Introduction, in some human computation games, the inter-group interaction is asymmetric and takes the form of a weak-link game, whereas certain crowdsourcing tasks such as collaborative design can be modeled as the dual strong-link game. In the former

[2]For teams consisting of just two workers, we use the terms team and pair interchangeably.

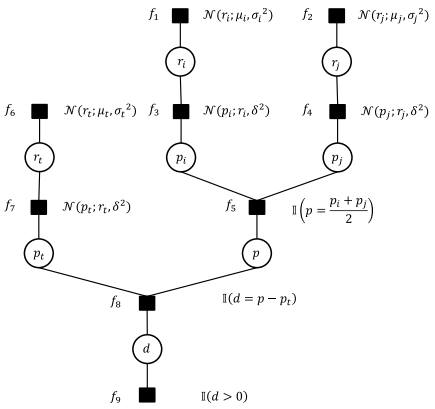


Figure 1: Factor graph for an encounter $< i, j, t >$ where the pair of workers (i, j) correctly solves task t.

(latter) case, workers of higher (lower) skill have a higher impact on the task output. In our work, we adopt a simpler symmetric approach where the interaction dynamics is simply modeled as the average of the player performances. This is similar to the model that TrueSkill considers to measure the performance of teams consisting of several players. Despite being less accurate, this model has the advantage of being linear (as opposed to the non-linear minimum or maximum operators), and allows for tractable inference.

In order to take into account the second feature of collaborative human computation, i.e., the task difficulty, we properly adapt the approach proposed in [28]. The idea is to model each task as an additional player, whose rating and performance also follow normal distributions. After encounter $< i, j, t >$ has taken place, we review the task output that the players jointly produced. If the pair of workers produced high quality output (e.g., informative or specific labels in the ESP game), we assume that the pair beat the fictitious worker that represents task t. In the opposite case, the pair loses to the fictitious worker. We do not consider the case of a draw. This approach actually turns the collaborative task into a competitive game. Note that we use the same variance δ^2 for the performance of the human players and the task for simplicity, but we could in principle also adopt different variances depending on the collaborative project at hand.

Figure 1 schematically depicts an example factor graph in the case where the pair of workers answers the question correctly, or "beats" the question. It has the form of a factor graph [26], which is basically a bipartite graph that consists of variables (circles) and nine factor nodes $f_1, \ldots, f_9$ (squares). The structure of the factor graph gives information about the dependencies of the factors.

Note that we can extend the model of Figure 1 in three ways. First, we can model the fact that ratings vary over time by assuming a Gaussian drift of skill between time steps given by $Pr(r_i^t | r_i^{t-1}) = \mathcal{N}(r_i^t; r_i^{t-1}, \tau^2)$ [5, 12]. Second, we can handle the case of more players by just adding one rating node r_k and one performance node p_k for each additional player k, and then defining the joint performance to be equal to the average of the individual player performances, i.e., $p = \frac{p_i+p_j+p_k+\cdots}{n}$, where n the number of players. Third, we can handle situations where the task output is not binary (correct versus incorrect or specific versus generic), but may take on one of $L > 2$ possible values (e.g., low, medium, or high specificity). To accommodate this, we can extend Figure 1 by considering $L-1$ tasks rather than just one, where each task also consists of a rating node r_t^l and one performance node p_t^l, where $2 \leq l \leq L$. Furthermore, we add $L-1$ difference nodes d^l that correspond to the difference between the team performance p and task performance p_t^l, where $2 \leq l \leq L$. Whenever an encounter occurs, we determine the level l^o of the outcome that the team of workers produced. Subsequently, we assume that the team beats all the tasks that correspond to level equal to or less than l^o, and loses to all task nodes that correspond to levels greater than l^o, i.e., $d_l > 0$ if and only if $l \leq l^o$. The natural interpretation for this is that if a team of workers generates an outcome of level l^o, where $1 \leq l^o \leq L$, this is equivalent to assuming that the team has (1) beaten all task levels lower than or equal to level l^o, and (2) has lost to all task levels greater than l^o. Note also that we do not need a node for the lowest level, since the team of workers is always guaranteed to beat that level; for instance, in a correct versus incorrect scenario, the team of workers can do no worse than giving an incorrect answer.

As a final remark, our model assumes that each worker is characterized by a single rating $\mathcal{N}(r_i; \mu_i, \sigma_i^2)$. In reality, a worker may have different skills for different tasks; for instance, they may be very knowledgeable about sports but know little about books. It is possible to consider multiple ratings per worker, one per task. In the remainder of the paper, as a proof of concept, we use a single rating per worker, which can be viewed as an indicator of the worker's average ability to provide high quality output on any task.

3.3 ELO Rating Scheme

Let $< i, j, t >$ be the encounter between i and j when they jointly perform task t. The task serves as a fictitious participant that plays against the human pair as shown in the graph model of Figure 1. Before the encounter takes place, assume the ratings of the workers are r_i and r_j, and the rating of the task node is r_t. The combined rating for the pair of workers will be according to our model $r = \frac{r_i+r_j}{2}$. The task node exhibits a performance p_t that is normally distributed around r_t with a fixed variance δ^2, while the performance p of the human pair node is normally distributed around r with a fixed variance $\frac{\delta^2+\delta^2}{4} = \frac{\delta^2}{2}$. But then the random variable $p - p_t$ will be normally distributed around $r - r_t$ with a variance $\delta^2 + \frac{\delta^2}{2} = \frac{3\delta^2}{2}$. The probability that the pair of workers beats the task will then be equal to $Pr(p > p_t | r, r_t) = \Phi(\frac{\sqrt{2}(r-r_t)}{\sqrt{3}\delta})$.

Our online ELO rating scheme will update the pair rating and the task rating after the encounter to reflect the game outcome, based on the premise that an entity should be "punished" if it did worse than expected, or boosted if it exceeded expectations. The expected outcome (in terms of the pair of workers beating the task node) will be $expected^{<i,j,t>} = \Phi(\frac{\sqrt{2}(r-r_t)}{\sqrt{3}\delta})$. The actual outcome $actual^{<i,j,t>}$ will be 1 if the pair of humans beats the task, and 0 otherwise. The resulting ELO update is:

$$r' = r + K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>})$$
$$r'_t = r_t - K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>})$$

Parameter $K > 0$ is a positive real number called the K-factor fixed in advance that is related to the maximum adjustment from a single encounter. Note that we increase the rating of an entity if it does better than expected, or we decrease it in the opposite case; the adjustment is proportional to (1) the K-factor, and (2) the difference between the actual and the expected outcome.

The update rule for the human pair can be rewritten as:

$$\frac{r_i' + r_j'}{2} = \frac{r_i + r_j}{2} + K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>}) \Leftrightarrow$$
$$r_i' + r_j' = r_i + r_j + 2 \cdot K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>})$$

But then we can split the $2 \cdot K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>})$ update amount equally to the two workers, so that we have:

$$r_i' = r_i + K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>})$$
$$r_j' = r_j + K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>})$$

Regarding convergence, in the presence of balanced matches, the ELO ratings converge exponentially fast to the actual player ratings [16]. If, on the other hand, the schedule is unbalanced, convergence can be very slow and a plethora of equilibria can emerge, depending on the exact initial distribution of initial ratings. Note that despite these theoretical concerns, ELO is a very robust scheme and widely used in several popular competitive games (see section 2.2). Algorithm 1 describes the update rules for ELO rating.

ALGORITHM 1: ELO Rating for Bayesian Factor Graph of Figure 1.

Input: An encounter $< i, j, t >$, prior ratings r_i, r_j, r_t, performance variance δ^2, and K-factor K.

Output: Updated ratings r_i', r_j', r_t' after encounter $< i, j, t >$.

Compute:

$$expected^{<i,j,t>} = \Phi(\frac{\sqrt{2}(r - r_t)}{\sqrt{3}\delta})$$
$$actual^{<i,j,t>} = \begin{cases} 1, & \text{if pair } (i, j) \text{ beats task } t \\ 0, & \text{otherwise} \end{cases}$$

Update:

$$r_i' = r_i + K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>})$$
$$r_j' = r_j + K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>})$$
$$r_t' = r_t - K \cdot (actual^{<i,j,t>} - expected^{<i,j,t>})$$

3.4 Bayesian Rating Scheme

We can estimate the unknown ratings with Expectation Propagation (EP), a deterministic approximate inference technique [30, 31]. Our goal is to determine the posterior distributions over the variables r_i and r_j, after encounter $< i, j, t >$ has occurred. EP is essentially an online message-passing algorithm for the factor graph of Figure 1.

Before going into the details, we briefly examine inference algorithms on factor graphs, i.e., graphs where the global function can be factorized into a product of local functions (factors) that depend on the node variables. The main algorithm for this class of graphs is the sum-product algorithm [26], which operates on the factor graph and attempts to compute various marginal functions. The basic idea is that if the graph is acyclic (as in our case), then it is possible to execute a message-passing algorithm where messages are passed from factor nodes to variable nodes, and vice versa. The algorithm terminates once two messages have passed over every edge, one in each direction. The message passing procedure for continuous variables is described by the following equations:

$$\textit{marginal distribution: } p(v) = \prod_{f \in n(v)} m_{f \to v}(v)$$
$$\textit{variable to factor: } m_{v \to f}(v) = \prod_{h \in n(v) - f} m_{h \to v}(v)$$
$$\textit{factor to variable: } m_{f \to v}(v) = \int \cdots \int f(V) \prod_{u \in n(f) - v} m_{u \to f}(u) dV_{-v}$$

where $m_{v \to f}(v)$ the message sent from variable v to factor f, $m_{f \to v}(v)$ the message sent from factor f to variable v, $n(x)$ is the set of neighbors of node x, V the set of variables connected to factor f, and V_{-v} the components of vector V except for variable v.

Of particular importance in our framework are the sum and difference of normal distributions. Assume two normal distributions $\mathcal{N}(x_1; \mu_1, \sigma_1^2)$ and $\mathcal{N}(x_2; \mu_2, \sigma_2^2)$. Their sum will be distributed according to $\mathcal{N}(x_1 + x_2; \mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$, and their difference according to $\mathcal{N}(x_1 - x_2; \mu_1 - \mu_2, \sigma_1^2 + \sigma_2^2)$. Moreover, in the message-passing algorithm, we often need to compute the product of two edge messages, or the ratio of a node message to an edge message. Since the messages in our framework are normal densities, given two normal densities $m_1 = (\mu_1, \sigma_1^2)$ and $m_2 = (\mu_2, \sigma_2^2)$, their product will be $m_1 \cdot m_2 = (\frac{\sigma_2^2 \cdot \mu_1 + \sigma_1^2 \cdot \mu_2}{\sigma_1^2 + \sigma_2^2}, \frac{\sigma_1 \cdot \sigma_2}{\sqrt{\sigma_1^2 + \sigma_2^2}})$ and their ratio $m_1 \div m_2 = (\frac{\sigma_2^2 \cdot \mu_1 - \sigma_1^2 \cdot \mu_2}{\sigma_2^2 - \sigma_1^2}, \frac{\sigma_1 \cdot \sigma_2}{\sqrt{|\sigma_2^2 - \sigma_1^2|}})$.

The only complication is created by factor f_9, which is non-Gaussian. To tackle this, we approximate the non-Gaussian posterior marginal $p(d)$ of node d with a normal distribution, using the technique of moment matching [31]. This technique consists in computing the mean and variance of the non-Gaussian message, and subsequently approximating it with a Gaussian distribution with the same first two moments. Moment matching provably minimizes the Kullback-Leibler divergence for Gaussian distributions. Assuming a message $m_{f_8 \to d} = (\mu_{f_8 \to d}, \sigma_{f_8 \to d}^2)$, the approximate posterior marginal $\hat{p}(d)$ will have mean and variance [19]:

$$\hat{\mu}_d = \mu_{f_8 \to d} + \sigma_{f_8 \to d} \cdot u\Big(-\frac{\mu_{f_8 \to d}}{\sigma_{f_8 \to d}}\Big) \quad (1)$$
$$\hat{\sigma}_d^2 = \sigma_{f_8 \to d}^2 \cdot \Big(1 - w\Big(-\frac{\mu_{f_8 \to d}}{\sigma_{f_8 \to d}}\Big)\Big) \quad (2)$$

where

$$u(x) = \begin{cases} \frac{\phi(x)}{1 - \Phi(x)}, & \text{if } x < 0 \\ -\frac{\phi(x)}{\Phi(x)}, & \text{if } x > 0 \end{cases}, w(x) = u(x) \cdot (u(x) - x).$$

Note that ϕ and Φ denote the probability density function and the cumulative distribution function of the standard normal distribution $\mathcal{N}(0, 1)$, respectively. Algorithm 2 presents the message-passing inference process.

Finally, note that even though Figure 1 and Algorithm 2 assume a single encounter, it is possible to assume multiple encounters, as long as the underlying factor graph does not have cycles [30, 31].

ALGORITHM 2: Approximate Inference for Bayesian Factor Graph of Figure 1.

Input: An encounter $< i, j, t >$, the prior ratings (μ_i, σ_i^2), (μ_j, σ_j^2), (μ_t, σ_t^2), and the performance variance δ^2.
Output: The posteriors $p(r_i \mid < i, j, t >)$, $p(r_j \mid < i, j, t >)$, and $p(r_t \mid < i, j, t >)$.

//Forward messages
1. $m_{f_1 \to r_i} = (\mu_i, \sigma_i^2)$, $m_{f_2 \to r_j} = (\mu_j, \sigma_j^2)$, $m_{f_6 \to r_t} = (\mu_t, \sigma_t^2)$
2. $m_{r_i \to f_3} = m_{f_1 \to r_i}$, $m_{r_j \to f_4} = m_{f_2 \to r_j}$, $m_{r_t \to f_7} = m_{f_6 \to r_t}$
3. $m_{f_3 \to p_i} = (\mu_{r_i \to f_3}, \sigma^2_{r_i \to f_3} + \delta^2)$, $m_{f_4 \to p_j} = (\mu_{r_j \to f_4}, \sigma^2_{r_j \to f_4} + \delta^2)$, $m_{f_7 \to p_t} = (\mu_{r_t \to f_7}, \sigma^2_{r_t \to f_7} + \delta^2)$
4. $m_{p_i \to f_5} = m_{f_3 \to p_i}$, $m_{p_j \to f_5} = m_{f_4 \to p_j}$, $m_{p_t \to f_8} = m_{f_7 \to p_t}$
5. $m_{f_5 \to p} = m_{p_i \to f_5} + m_{p_j \to f_5}$, $m_{p \to f_8} = m_{f_5 \to p}$
6. $m_{f_8 \to d} = m_{p \to f_8} - m_{p_t \to f_8}$
7. $\hat{p}(d) = (\hat{\mu}_d, \hat{\sigma}_d^2)$ (Equations 1 and 2)

//Backward messages
8. $m_{d \to f_8} = \frac{\hat{p}(d)}{m_{f_8 \to d}}$
9. $m_{f_8 \to p} = m_{p_t \to f_8} + m_{d \to f_8}$, $m_{f_8 \to p_t} = m_{p \to f_8} - m_{d \to f_8}$
10. $m_{p \to f_5} = m_{f_8 \to p}$, $m_{p_t \to f_7} = m_{f_8 \to p_t}$
11. $m_{f_5 \to p_i} = 2 \cdot m_{p \to f_5} - m_{p_j \to f_5}$, $m_{f_5 \to p_j} = 2 \cdot m_{p \to f_5} - m_{p_i \to f_5}$
12. $m_{p_i \to f_3} = m_{f_5 \to p_i}$, $m_{p_j \to f_4} = m_{f_5 \to p_j}$
13. $m_{f_3 \to r_i} = (\mu_{p_i \to f_3}, \sigma^2_{p_i \to f_3} + \delta^2)$, $m_{f_4 \to r_j} = (\mu_{p_j \to f_4}, \sigma^2_{p_j \to f_4} + \delta^2)$, $m_{f_7 \to r_t} = (\mu_{p_t \to f_7}, \sigma^2_{p_t \to f_7} + \delta^2)$

//Posterior rating marginals
14. $p(r_i \mid < i, j, t >) = m_{f_3 \to r_i} \cdot m_{f_1 \to r_i}$, $p(r_j \mid < i, j, t >) = m_{f_4 \to r_j} \cdot m_{f_2 \to r_j}$, $p(r_t \mid < i, j, t >) = m_{f_7 \to r_t} \cdot m_{f_6 \to r_t}$

However, in practice the player encounters can form cycles, for instance when every player is paired with every other player. For this reason, we favor our simpler model in Figure 1, which considers a single encounter and avoids cycles in the underlying factor graph.

4 EXPERIMENT

We test our framework using the ESP game as a case study, as it is typically viewed as the most influential human computation game.

4.1 System and Task Design

Figure 2 depicts the graphical user interface of the game used in our experiment. It closely resembles the original Google Image Labeler, the commercially licensed version of the ESP game [17]. Our implementation is web-based and written in JavaScript and PHP, and also uses an Apache HTTP web server.

The rules in our implementation have both similarities and differences, compared to the original version. First, players have 40 seconds to match on a label for any image, while in the original game players were given 120 seconds to try to match as many pictures as possible. This is related to the fact that we do not include a passing mechanism, as we discuss later. Second, if a pair of players match on a label, they both earn a score ranging from 10 to 100 depending on the label specificity, which they can immediately see upon agreement. Third, if time is out, players win 0 points each (no agreement reached), and they move on to the next image. Fourth, for every picture we additionally include a short list of 0 to 2 taboo (off-limits) labels that players are not allowed to provide. As opposed to the original game, our version does not allow players to skip an image and move on to the next one, because we need to know their performances in all tasks to be able to rate them better.

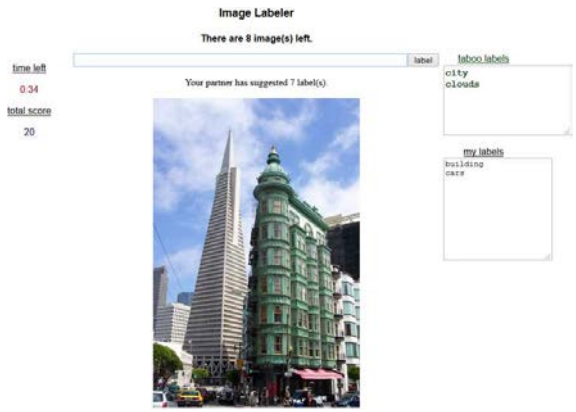


Figure 2: The interface of our simulated ESP game.

Regarding the actual tasks, we select 10 pictures from the Internet (Figures 3a-3j). Our tasks satisfy two properties: (1) they cover various common themes in the original ESP game such as animals, food, Hollywood, architecture, and science, and (2) they have both a generic and a specific component. For instance, Tapir accepts generic labels such as "animal" or "large" as well as the specific animal name. Transamerica can be labeled with the generic label "skyscraper" or the specific label "San Francisco".



Figure 3: Pictures from our MTurk experimental evaluation.

4.2 Experiment Design

An interesting design of our experiment is that we do not directly ask two humans to play with each other. Instead, we develop a bot that simulates a high-skilled player; a human worker is then paired with our bot. The motivation behind this decision is threefold. First,

a bot allows us to control the label scores. It is practically impossible to know in advance all possible labels that two human players may agree on and assign scores to them. A bot with a predefined set of labels with associated scores can tackle this issue. Of course, this requires that the label set is large enough and carefully selected; we discuss this point in more detail later. Second, it is easier to infer human players' true skill as the baseline, since the bot does not act as the weak link in the collaboration. Third, as we discuss below it is possible to simulate a very large number of human encounters based on the bot responses, which would be very difficult and costly to do directly. Interestingly, our design novelty is appropriate for any collaborative task where the goal is to generate many encounters at a lower cost. This method can be applied to real collaborative crowd computing tasks in cold-start situations to generate an initial assessment of task difficulty and worker skill.

We did not mention to the players they are playing against a bot; instead, we instructed them that they would be paired with another human player. In addition, we generate the time period between two consecutive labels randomly from 1 to 5 seconds to give the human workers the impression that they are playing with a real human instead of a bot, especially given that they only play together for 10 images. The bot provides 12-14 representative labels per image that we carefully select so that: (1) they range from very specific to very generic in decreasing order to prioritize high-quality labels, and (2) they cover the main objects or semantic aspects of the image. This ensures that both high-skilled and low-skilled players can reach an agreement with the bot. In addition, we created variations of these representative labels, such as singular and plural form, capitalized first letter for names etc. For each image, the bot generates the labels in decreasing order of specificity, because in game-theoretical terms it is a dominant strategy for a rational player to output the labels in that order [18]; otherwise, the pair of players might first match on a label with lower specificity (and thus lower score). The label selection was based on a preliminary pilot study on MTurk, where we asked 20 workers to provide both generic and specific labels for each picture. Following that, we asked two researchers to rank the labels of each image from the most generic to the most specific independently. The Spearman's coefficient of 0.82 (p-value < 0.05) suggests strong inter-rater reliability. We invited a third researcher into the group discussion to resolve disagreement and come up with the final assessment. The top six or seven labels from the generic and specific categories then form the label set for each image. We subsequently assigned scores using even intervals between consecutive terms, so that the most generic picture gets a score of 10 and the most specific a score of 100, and we rounded the remaining scores to the nearest multiple of 10.

Finally, we upload our game as a human intelligence task (HIT) on Amazon Mechanical Turk (MTurk). Upon successful completion of the HIT, i.e., reaching an agreement in at least eight out of the ten images presented in a randomized order, workers earn \$0.25, for an average hourly rate per worker of about \$3.0. To motivate workers to put more effort into the game, we include a bonus of \$5 to the 5 workers with the highest total scores. For quality control purposes, we only allowed MTurk workers with acceptance rates for submitted tasks of at least 80%, and with at least 10 submitted tasks in their history. Note that a worker may be truthful but still have low skill, the two notions are not contradictory. Given that our study involves specific labels, we restrict our tasks to MTurk workers in the US.

4.3 Data Collection

In total, we accepted data from 100 workers. For each worker, we recorded the set of labels provided for each image until either a match or a timeout occurred, as well as the corresponding timestamp. From these results, we were able to generate two datasets. (1) The first dataset, which we call BOT-PARTNER, contains the 10×100=1,000 encounters between the population of the 100 players and the bot. (2) The second dataset, which we name HUMAN-PARTNER, contains the (hypothetical) encounters between any two players from the 100 workers. It is indeed possible to simulate these encounters, based on the (label, timestamp) recorded information. One problem is that not all encounters are valid. For instance, assume that one player reaches an agreement with the bot in the first 5 seconds, whereas it takes 35 seconds for another player to reach an agreement for the same image. In the simulated encounter where these two players are paired together, there may be no match because the first player only played for a limited time. To account for this, we ignore (invalid) encounters that result in no match, when at least one of the two players played the corresponding image for less than 40 seconds (this happens when the player did not manage to match on any label). By construction, HUMAN-PARTNER is not an unbiased dataset: encounters involving players who match with the bot in BOT-PARTNER may be ignored; on the other hand, encounters where both players did not match with the bot can never be ignored (they either match on a new label or they do not match yielding a score of 0). Given this bias, we expect any given method will not produce identical rankings on the two datasets.

In particular, HUMAN-PARTNER contains 17,357 (valid) encounters in total (instead of $10 \times \binom{100}{2} = 49,500$); furthermore, by construction it has a bias toward low-skilled workers, as it takes these workers longer to come to a match with the bot and as a result their encounters are more likely to be kept. This is an inherent limitation of this design scheme that we also discuss in Section 5.4. Note that in BOT-PARTNER a player can only generate a match on the predetermined 10-12 labels that the bot is programmed to provide; in contrast, in HUMAN-PARTNER matches occur on labels that both human workers provided, which may be different from the predetermined labels of the bot. The total number of matched labels over all 17,357 encounters that were different from the bot's labels were 129: 22 for "Pumpernickel", 10 for "Tapir", 13 for "Hunger Games", 7 for "Captcha", 31 for "Declaration", 10 for "Soba Noodles", 14 for "Transamerica", 8 for "Albert Einstein", 1 for "Cheetah", and 13 for "Phi-beta-sigma". Note that intuitively easier tasks such as "Cheetah" or "Albert Einstein" resulted in fewer new matches than harder tasks such as "Pumpernickel", or tasks that are associated with a plethora of objects and/or concepts such as "Declaration". In order to assign a score to matched labels not in the predetermined set, we asked the previous three raters that we used to select labels for the bot to score these new labels from 10 to 100 in multiples of 10 according to how specific they were. The value of 60 was used as a cutoff: labels with specificity 10-50 were classified as generic, whereas labels with specificity 60-100 were classified as specific. The average of the three values was then selected as the label's

score. For labels where the three values differed significantly, we invited a fourth rater to confirm and finalize the score.

Table 1 provides descriptive statistics about the matched labels (specific, generic, or no match) in BOT-PARTNER for each of the ten tasks. Note that the first three columns sum up to 100, i.e. the number of workers since in BOT-PARTNER each worker is involved in one encounter per task. Images where fewer players managed to match on a specific label are usually associated with a higher average matching time (the Pearson's correlation coefficient is -70.65%). These images naturally correspond to harder tasks; for instance, "Transamerica" or "Pumpernickel" are harder than "Albert Einstein" or "Cheetah". Alternatively, the Specific:Generic ratio can serve as an indicator of the image's difficulty. Easier tasks tend to have high values for that ratio; for instance, for "Cheetah" this ratio stands at 24. Difficult tasks, on the other hand, are associated with a lower ratio; for example, for "Transamerica" it stands at 0.59.

Table 1: Statistics for the ten tasks in BOT-PARTNER

	Specific	Generic	No match	Matching time (sec)
Pumpernickel	45	40	15	12.71 ± 7.34
Tapir	76	22	2	11.44 ± 6.95
Hunger Games	90	3	7	8.93 ± 7.68
Captcha	61	36	3	9.69 ± 8.06
Declaration	69	20	11	14.82 ± 8.58
Soba Noodles	71	23	6	9.15 ± 6.00
Transamerica	35	59	6	13.33 ± 12.08
Albert Einstein	93	6	1	8.07 ± 7.93
Cheetah	96	4	0	7.15 ± 8.37
Phi-beta-sigma	76	20	4	11.78 ± 8.79

Table 2 summarizes the descriptive statistics of matched labels for HUMAN-PARTNER. We have added a column "Invalid" that contains the number of invalid encounters. Note that there are totally $\binom{100}{2} = 4,950$ encounters for every image; hence, the first four columns sum up to 4,950. It is not surprising that most encounters are invalid, given that the majority of encounters in BOT-PARTNER result in a match and thus last less than 40 seconds. A higher Specific:Generic ratio is again associated with easier tasks, which are in turn associated with lower matching times. However, we notice in Tables 1 and 2 that for any image the ratios between the specific and generic matches are not necessarily the same in BOT-PARTNER and HUMAN-PARTNER. The percentage of generic matches over all 10 tasks is 24.5% in the former but 28.23% in the latter dataset. The reason for this discrepancy is that in the first dataset the bot simulates a highly skilled player, so encounters are more likely to occur on specific labels. Thus, even though the two datasets are based on the same 10 tasks and the same set of 100 workers, they may result in different workers' skills and task strains.

Note that the above tables do not contain information about how individual workers did across the 10 tasks. Intuitively, we expect that high-skilled workers tend to produce more specific tasks. To test for this, we considered the ELO rating (see Section 5) for the 100 workers based on BOT-PARTNER. We then isolate the top-25 workers as well as the bottom-25 ones, and subsequently simulate encounters among the top 25 workers only; or, among the top 25 and the bottom 25 workers; or, among the bottom 25 workers only.

Table 2: Statistics for the ten tasks in HUMAN-PARTNER

	Specific	Generic	No match	Invalid	Matching time
Pumpernickel	413	451	41	4045	14.65±8.16
Tapir	819	978	6	3147	11.66±7.63
Hunger Games	1405	33	20	3492	10.02±5.28
Captcha	1043	400	20	3487	11.84±7.25
Declaration	806	199	83	3862	18.19±9.30
Soba Noodles	1163	922	5	2860	11.45±7.37
Transamerica	437	1497	51	2965	15.12±9.30
Albert Einstein	2580	61	4	2305	8.73±6.56
Cheetah	1771	66	0	3113	8.42±5.68
Phi-beta-sigma	1816	213	54	2867	13.08±9.13

We only keep the valid encounters. The percentage of encounters resulting in specific matches stands at 88.31%, 61.20%, and 31.44%, respectively, which is in full accordance with our intuition. We also observe that the percentage of valid encounters that result in no match is 0.57%, 3.44%, and 9.44%, respectively. This suggests that encounters involving low-skilled workers are significantly more likely to result in a generic match or no match at all.

5 RATING RESULTS AND ANALYSIS

For each encounter in either dataset, we asked three raters from our research group who were blind to the conditions to vote whether the quality of the matched label (if a match occurred) is high (e.g., informative and specific) or low; we combine the answers with majority voting. We then rate the workers with the following three methods: (1) AVERAGE score, where a specific match yields 1 point and a generic match or a no match yields 0 points, (2) ELO rating of Algorithm 1 with $K = 24.0$, and (3) EP of Algorithm 2 with $\mu_0 = 25, \sigma_0 = \frac{25}{3}, \delta = \frac{\sigma_0}{2}$ (based on [12]). In BOT-PARTNER, we consider that each team consists of a single human player, whereas in HUMAN-PARTNER each team consists of a pair of workers. We compare our proposed rating framework with the baseline average scoring scheme in terms of consistency, accuracy, and robustness.

5.1 Output Similarity across Rating Schemes

First, we investigate how similar the three methods perform on each dataset. To compare the methods, we consider the rankings that the worker ratings and task strains induce, and subsequently calculate the similarities between them using Spearman's rank correlation coefficient[3] [22]. The correlation metric in this context makes sense as a way of capturing the degree to which the different methods agree on the generated rankings; intuitively, they should be positively correlated since they rank based on the same data.

Table 3 contains our results. In general, HUMAN-PARTNER is more challenging for AVERAGE, because (1) not all encounters are valid, and (2) it has a bias toward low-skilled workers. As a result, the task schedules are unbalanced, and the number and type of encounters for each worker varies. In comparison, rankings produced by the three methods have stronger correlations in BOT-PARTNER, where the schedule is balanced, i.e., each player plays with the bot in all 10 tasks for a total number of 10 encounters as

[3]The p-values of all reported correlation coefficients are very low due to the high sample size and we do not report them here.

Table 3: Spearman's correlation coefficient between worker rankings and task rankings from different methods

	BOT-PARTNER		HUMAN-PARTNER	
	WORKER	TASK	WORKER	TASK
AVERAGE-ELO	95.66%	89.09%	73.89%	67.27%
AVERAGE-EP	75.90%	53.94%	50.85%	30.91%
ELO-EP	85.94%	76.97%	85.63%	74.55%

opposed to the less regular second dataset. ELO and EP are generally strongly correlated, since they both rely on the same Bayesian factor graph of Figure 1. One possible reason for the weaker correlation between AVERAGE and EP is that we perform EP after each individual encounter instead of simultaneously optimizing over multiple encounters. This may make EP less accurate, as it needs both the mean and variance. In contrast, the simpler update rules of ELO using only the mean can produce results closer to those of AVERAGE. We also note that the rankings that different methods produce are less correlated for the images than for the workers, possibly due to the very low number of tasks (10) compared to the higher number of workers (100). This is more pronounced for EP.

Next, we investigate whether the same method yields consistent rankings when applied to BOT-PARTNER and HUMAN-PARTNER based on Spearman's correlation coefficient. Using the rank correlation coefficient to compare rankings in different datasets is tricky, because a higher correlation coefficient does not necessarily imply a better ranking. For example, using a random but fixed ranking across different datasets yields a correlation coefficient of 1, even though the underlying random ranking may be totally unrelated to the ground truth. Despite this, Spearman's correlation coefficient can serve as a necessary condition for consistency: given the two datasets refer to the same set of workers and tasks, we intuitively expect similar rankings in the two datasets.

Table 4 shows the correlation coefficients for worker and task rankings for all methods. We observe that worker rankings are positively correlated at around 65%-70%, while task rankings are very strongly correlated at above 90%, with AVERAGE being the least consistent. Task rankings are probably more consistent because a task participates in many more encounters than a worker, so its strain can be estimated more accurately. The positive correlations suggest that dataset HUMAN-PARTNER that we generated by emulation is consistent with the original dataset BOT-PARTNER and can be used in conjunction with it.

Table 4: Spearman's correlation coefficient between worker rankings and task rankings in two datasets

	WORKER RANKINGS	TASK RANKINGS
AVERAGE	65.61%	90.30%
ELO	67.60%	95.15%
EP	71.65%	92.73%

To gain insight into the different rankings, we consider the image "Phi-beta-sigma" in HUMAN-PARTNER. AVERAGE ranks it as the 5th easiest, whereas ELO ranks it as the 3rd easiest. The reason for this discrepancy is that AVERAGE only considers the specificity of the outputs, whereas ELO additionally considers information about the workers involved. We manually verified that there were five players who did not perform well in other tasks, yet managed to score high in "Phi-beta-sigma". This is because these workers provided specific labels such as "fraternity", "frat", or "sorority". Furthermore, three players that did well in other tasks did not perform well in that image, possibly because they were unaware of both the exact Greek characters or their connection to fraternities and sororities. Given that low-rating workers provided specific answers, ELO and EP adjust the image rating to be lower (easier).

The above example additionally demonstrates that the various methods can coexist and play a complementary role. When they are in agreement, this suggests very small uncertainty over the ranking. Disagreement, on the other hand, may be indicative of some interesting fact about the task or the workers.

5.2 Prediction Accuracy

Ratings can be used for balanced task assignment. To confirm this, we investigate whether the worker and task ratings of the three methods can accurately predict output quality. In particular, we predict that a worker (pair of workers) in BOT-PARTNER (HUMAN-PARTNER) will match on an informative label, if its rating (their average rating) is greater than the task rating. Otherwise, we predict that the worker (pair of workers) will agree on a generic label. Table 5 contains the accuracy for each method on each dataset.

Table 5: Prediction accuracy of task output

	BOT-PARTNER	HUMAN-PARTNER
AVERAGE	60.83%	64.56%
ELO	75.63%	70.08%
EP	62.29%	58.13%

Overall, ELO performs significantly better than EP in both datasets. This may be because (1) the EP update occurs after each individual encounter instead of optimizing over all encounters simultaneously, and (2) EP considers both the mean and the variance while ELO only needs the mean. Furthermore, we note that ELO achieves consistently higher accuracy than AVERAGE. It is telling that even in HUMAN-PARTNER with the 17,357 encounters, AVERAGE does not manage to outperform ELO. This demonstrates our original claim that even in large datasets AVERAGE does not capture the complex dynamics of the worker interaction and the task strain, which can hurt performance.

5.3 Effect of Worker Interaction

We next investigate the effect of worker interaction. For this purpose, we consider a modified Bayesian rating scheme, where a player's performance is only determined by itself and not the joint effort (this only affects ELO and EP). In other words, even though the team of players jointly produces the output, we assume in this experiment that each player's performance in an encounter depends only on their individual skill. In BOT-PARTNER, this was already the case since we considered teams consisting of a single worker. We now assume this to also be the case in HUMAN-PARTNER, and we investigate how the correlation coefficients for worker and task rankings between AVERAGE, ELO, and EP change. We show our

results in Table 6. We make two observations. First, the correlation of both ELO and EP compared to AVERAGE decreases substantially. This is because we assume a less accurate model that ignores worker interaction. Second, ELO and EP are still highly correlated, since they both estimate the ratings for the same (wrong) model.

Table 6: Correlation coefficient between worker rankings and task strains in HUMAN-PARTNER without interaction

	WORKER RANKINGS	TASK STRAINS
AVERAGE-ELO	44.44%	58.79%
AVERAGE-EP	30.27%	39.39%
ELO-EP	88.46%	92.73%

Finally, we study the robustness of the various schemes by looking at the effect of unbalanced task schedules. For this purpose, we consider the dataset BOT-PARTNER, but this time we assume that each encounter occurs with a fixed probability p. Trivially, the case of p=1 corresponds to the full dataset where all encounters are considered. In Figure 4, we calculate Spearman's coefficient between the ranking that each method produces when p=1 and the ranking it generates as p decreases from 0.1 to 0.9. For each value of p, we randomly sample 20 subsets of encounters, and report the average. We observe that AVERAGE has consistently the worst performance, in the sense that the ranking it produces changes more significantly as p decreases. ELO is the most robust, followed by EP. This confirms our claim that the average score is very good as long as the task schedules are balanced, whereas ELO and EP are more robust and superior for unbalanced schedules.

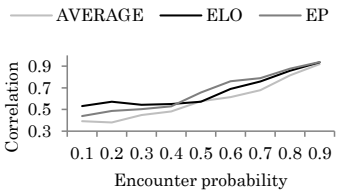


Figure 4: Spearman's correlation coefficient for worker rankings in terms of encounter probability in BOT-PARTNER.

5.4 Discussion

Our proposed experimental design based on simulated encounters has the potential to generate very large datasets from a small initial dataset that acts as a seed in cold-start situations. It is thus a cost-efficient solution, tailored for costly human studies involving large numbers of participants. Furthermore, it has the advantage that we can control how large the generated dataset will be by adjusting the encounter probability p. In principle, we can even generate a larger number of smaller datasets by doing many simulations with small values of p. A downside is that the generated dataset can contain a large number of invalid encounters. Furthermore, our scheme is not necessarily unbiased, since it is more likely to keep encounters with low-skilled workers as it takes these workers longer to match with the bot. Note that this is not a general property of our proposed scheme but related to the specifics of the ESP game where players immediately move on to the next task as soon as a match is reached.

All three methods are online and thus fast to implement in any collaborative crowd computing scenario. Relaxing the original model without considering the worker interaction leads to very inconsistent rankings and discrepancy between ELO and AVERAGE. This suggests that worker interaction is indeed an important ingredient of our model and leads to better rankings. ELO has the highest prediction accuracy of over 70% out of the three methods, followed by AVERAGE. Furthermore, ELO leads to the most consistent worker ranking even as the encounter probability decreases, followed by EP. This suggests that ELO is a very robust algorithm that generally outperforms AVERAGE. Its strengths lie in the facts that (1) it takes into account the worker dynamics and the task strains, and (2) it has a simple update rule that only relies on the mean. Compared to ELO, EP converges much faster [12]. It also leads to consistent rankings, and in some cases beats AVERAGE. EP tries to optimize over both the first two moments, i.e., the mean and the variance information. ELO, on the other hand, is based on the same model but uses a simpler update rule with only the mean, making it generally more robust when we do not simultaneously optimize over multiple encounters. The fact that our Bayesian graph model is approximate and does not capture the true dynamics may also contribute to the robustness of the simpler ELO scheme.

In general, all three methods are valuable and can convey important information about the worker ratings and task strains. Of particular interest are cases of disagreement, where the models produce very different ratings. This may indicate that low-skilled workers have produced high-quality output for a task that workers of higher rating have failed, or that a task has been surprisingly hard for high-skilled workers. The service requester can utilize this information to generate better assignments and improve the output quality. Even though AVERAGE is generally a resilient method, especially as the number of encounters grows, our empirical evaluation demonstrates that a collaborative ELO rating system can outperform it in terms of consistency, accuracy, and robustness. This implies that ELO could serve as the primary rating system.

6 CONCLUSION

Collaborative human computation and crowdsourcing are characterized by tasks of varying difficulties and involve interactions among human workers. In this work, we show how to leverage the standard rating schemes in competitive games for rating participants and tasks in collaborative crowd computing. Experimental results demonstrate that collaborative ranking using our TrueSkill-inspired rating framework can lead to better results than the traditional additive scoring scheme. In the future, we would like to explore how the specific type of interaction (e.g., weak-link, strong-link, linear) affects the rating process in our model. We will also consider the uncertainty over the output that the workers jointly produce. This is especially true for human computation games, where the game service provider relies on the player agreement to accept the output, but has no other way to know for sure whether it is correct or specific, etc. It would thus be valuable to extend our frameworks so that they can handle cases where there is additional uncertainty over the worker output.

REFERENCES

[1] Lora Aroyo and Chris Welty. 2014. The Three Sides of CrowdTruth. *Journal of Human Computation* 1 (2014), 31–34. Issue 1.

[2] Ralph Allan Bradley and Milton E. Terry. 1952. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika* 39, 3/4 (1952), 324–345.

[3] Miguel Brozos-VÃązquez, Marco Antonio Campo-Cabana, JosÃĺ Carlos DÃŋaz-Ramos, and Julio GonzÃąlez-DÃŋaz. 2008. Ranking participants in tournaments by means of rating functions. *Journal of Mathematical Economics* 44, 11 (2008), 1246 – 1256.

[4] Xi Chen, Paul N. Bennett, Kevyn Collins-Thompson, and Eric Horvitz. 2013. Pairwise Ranking Aggregation in a Crowdsourced Setting. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining (WSDM '13)*. 193–202.

[5] Pierre Dangauthier, Ralf Herbrich, Tom Minka, and Thore Graepel. 2007. TrueSkill Through Time: Revisiting the History of Chess. In *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS'07)*. 337–344.

[6] Herbert A. David. 1988. *The method of paired comparisons* (2nd. ed.). Chapman and Hall, London.

[7] A. P. Dawid and A. M. Skene. 1979. Maximum Likelihood Estimation of Observer Error-Rates Using the EM Algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28, 1 (1979), 20–28.

[8] Julie S. Downs, Mandy B. Holbrook, Steve Sheng, and Lorrie Faith Cranor. 2010. Are Your Participants Gaming the System?: Screening Mechanical Turk Workers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. 2399–2402.

[9] Arpad E. Elo. 1978. *The rating of chess players, Past and Present.* Arco Publishing, New York.

[10] Mark E. Glickman. 2001. Dynamic paired comparison models with stochastic variances. *Journal of Applied Statistics* 28, 6 (2001), 673–689.

[11] Bruce Hajek, Sewoong Oh, and Jiaming Xu. 2014. Minimax-optimal Inference from Partial Rankings. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14)*. 1475–1483.

[12] Ralf Herbrich, Tom Minka, and Thore Graepel. 2007. TrueSkill™: A Bayesian Skill Rating System. In *Advances in Neural Information Processing Systems 19*. 569–576.

[13] Jack Hirshleifer. 1983. From Weakest-Link to Best-Shot: The Voluntary Provision of Public Goods. *Public Choice* 41, 3 (1983), 371–386.

[14] Jeff Howe. 2006. The rise of crowdsourcing. *Wired magazine* 14, 6 (2006), 1–4.

[15] Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality Management on Amazon Mechanical Turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation (HCOMP '10)*. 64–67.

[16] Pierre-Emmanuel Jabin and StÃIphane Junca. 2015. A Continuous Model For Ratings. *SIAM J. Appl. Math.* 75, 2 (2015), 420–442.

[17] Nassim Jafarinaimi. 2012. Exploring the Character of Participation in Social Media: The Case of Google Image Labeler. In *Proceedings of the 2012 iConference (iConference '12)*. 72–79.

[18] Shaili Jain and David C. Parkes. 2008. A Game-Theoretic Analysis of Games with a Purpose. In *Proceedings of the 4th International Workshop on Internet and Network Economics (WINE '08)*. 342–350.

[19] N. L. Johnson, S. Kotz, and N. Balakrishnan. 1995. *Continuous Univariate Distributions, Vol. 1-2* (2nd. ed.). John Wiley and Sons, New York.

[20] David R. Karger, Sewoong Oh, and Devavrat Shah. 2011. Iterative Learning for Reliable Crowdsourcing Systems. In *Advances in Neural Information Processing Systems 24*. 1953–1961.

[21] David R. Karger, Sewoong Oh, and Devavrat Shah. 2014. Budget-Optimal Task Allocation for Reliable Crowdsourcing Systems. *Oper. Res.* 62, 1 (2014), 1–24.

[22] Maurice J. Kendall. 1970. *Rank correlation methods* (4th. ed.). Hafner Press, New York.

[23] Aniket Kittur. 2010. Crowdsourcing, Collaboration and Creativity. *XRDS* 17, 2 (2010), 22–26.

[24] Aniket Kittur, Ed H. Chi, and Bongwon Suh. 2008. Crowdsourcing User Studies with Mechanical Turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. 453–456.

[25] Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E. Kraut. 2011. CrowdForge: Crowdsourcing Complex Work. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. 43–52.

[26] F. R. Kschischang, B. J. Frey, and H. A. Loeliger. 2006. Factor Graphs and the Sum-product Algorithm. *IEEE Trans. Inf. Theor.* 47, 2 (2006), 498–519.

[27] Edith Law and Luis von Ahn. 2009. Input-agreement: A New Mechanism for Collecting Data Using Human Computation Games. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. 1197–1206.

[28] Jing Liu, Quan Wang, Chin-Yew Lin, and Hsiao-Wuen Hon. 2013. Question Difficulty Estimation in Community Question Answering Services. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 85–90.

[29] Panagiotis Mavridis, David Gross-Amblard, and Zoltán Miklós. 2016. Using Hierarchical Skills for Optimized Task Assignment in Knowledge-Intensive Crowdsourcing. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. 843–853.

[30] Thomas P. Minka. 2001. Expectation Propagation for Approximate Bayesian Inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI '01)*. 362–369.

[31] Thomas P. Minka. 2001. *A Family of Algorithms for Approximate Bayesian Inference.* Ph.D. Dissertation.

[32] Robert Munro. 2010. Crowdsourced translation for emergency response in Haiti: the global collaboration of local knowledge.

[33] Sahand Negahban, Sewoong Oh, and Devavrat Shah. 2012. Iterative ranking from pair-wise comparisons. In *Advances in Neural Information Processing Systems 25*. 2474–2482.

[34] Habibur Rahman, Saravanan Thirumuruganathan, Senjuti Basu Roy, Sihem Amer-Yahia, and Gautam Das. 2015. Worker Skill Estimation in Team-based Tasks. *Proc. VLDB Endow.* 8, 11 (2015), 1142–1153.

[35] Ariel Rubinstein. 1980. Ranking the Participants in a Tournament. *SIAM J. Appl. Math.* 38, 1 (1980), 108–111.

[36] Nihar B. Shah, Sivaraman Balakrishnan, Joseph Bradley, Abhay Parekh, Kannan Ramchandran, and Martin J. Wainwright. 2016. Estimation from Pairwise Comparisons: Sharp Minimax Bounds with Topology Dependence. *J. Mach. Learn. Res.* 17, 1 (2016), 2049–2095.

[37] L. L. Thurstone. 1927. A law of comparative judgment. *Psychological Review* 34, 4 (1927), 273–286.

[38] Luis Von Ahn. 2005. *Human Computation.* Ph.D. Dissertation.

[39] Luis von Ahn and Laura Dabbish. 2004. Labeling Images with a Computer Game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04)*. 319–326.

[40] Luis von Ahn and Laura Dabbish. 2008. Designing Games with a Purpose. *Commun. ACM* 51, 8 (2008), 58–67.

[41] Luis von Ahn, Mihir Kedia, and Manuel Blum. 2006. Verbosity: A Game for Collecting Common-sense Facts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. 75–78.

[42] Luis von Ahn, Ruoran Liu, and Manuel Blum. 2006. Peekaboom: A Game for Locating Objects in Images. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. 55–64.

[43] Jacob Whitehill, Ting fan Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. 2009. Whose Vote Should Count More: Optimal Integration of Labels from Labelers of Unknown Expertise. In *Advances in Neural Information Processing Systems 22*. 2035–2043.

[44] Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I. Jordan. 2016. Spectral Methods Meet EM: A Provably Optimal Algorithm for Crowdsourcing. *J. Mach. Learn. Res.* 17, 1 (2016), 3537–3580.

[45] Dengyong Zhou, John C. Platt, Sumit Basu, and Yi Mao. 2012. Learning from the Wisdom of Crowds by Minimax Entropy. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS'12)*. 2195–2203.

Multimodal Review Generation for Recommender Systems

Quoc-Tuan Truong
Singapore Management University
Singapore
qttruong.2017@smu.edu.sg

Hady W. Lauw
Singapore Management University
Singapore
hadywlauw@smu.edu.sg

ABSTRACT

Key to recommender systems is learning user preferences, which are expressed through various modalities. In online reviews, for instance, this manifests in numerical rating, textual content, as well as visual images. In this work, we hypothesize that modelling these modalities jointly would result in a more holistic representation of a review towards more accurate recommendations. Therefore, we propose Multimodal Review Generation (MRG), a neural approach that simultaneously models a rating prediction component and a review text generation component. We hypothesize that the shared user and item representations would augment the rating prediction with richer information from review text, while sensitizing the generated review text to sentiment features based on user and item of interest. Moreover, when review photos are available, visual features could inform the review text generation further. Comprehensive experiments on real-life datasets from several major US cities show that the proposed model outperforms comparable multimodal baselines, while an ablation analysis establishes the relative contributions of the respective components of the joint model.

ACM Reference Format:
Quoc-Tuan Truong and Hady W. Lauw. 2019. Multimodal Review Generation for Recommender Systems. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313463

1 INTRODUCTION

One classic formulation for recommender system is rating prediction [13]. Increasingly ratings are no longer the sole modality for expressing one's preference. The most common scenario now is when rating is but a component of an online review, which also contains a textual description of one's experiences with a product or a place. Figure 1 illustrates a review from Yelp.com for *Antica Ristorante*, a popular Italian restaurant in the Financial District of NYC. The writer *Adam W.* assigned it the highest rating of 5 stars, and articulated his impression through sentences stating supporting factors such as amazing and flavorful clam linguini, attentive servers, and homemade grappa. Evident from the example is how vividly the textual description embodies multi-faceted user's preferences, with the rating being a culminating assessment.

We start off with the premise that the textual content of a review would be useful to rating prediction. Upon perusing the existing literature on content-based recommender systems (see Section 2),

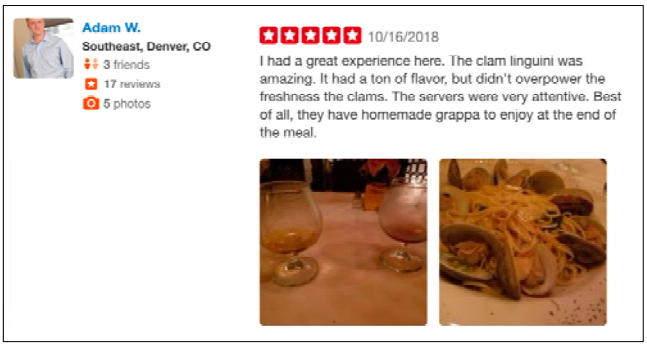


Figure 1: Example of *Yelp* review for Antica Ristorante

we observe a couple of distinctions in our intended setting here. First, when textual content was used for recommendation, in most cases it was in association with the item (e.g., product description, paper abstract), rather than with the user-item tuple (i.e., a review). Even in those cases that relied on review texts, the approach was usually to aggregate all the reviews belonging to an item into a document [19, 20]. Second, textual content was commonly seen as observations or features, rather than as a generative output.

Problem We express the problem as *multimodal review generation*. Given a user and an item, we seek to generate both the rating and the review text. Prior works tend to address one given the other. For instance, several content-based recommendations [10, 33] generate ratings based on observed text. On the other hand, product review generation [5] seeks to generate the review text based on user, item, and the observed rating. To showcase the benefits of joint modeling, we will compare to these classes of baselines. For an additional benefit, the generated text could also potentially serve as explanations [7, 37] to the rating-based recommendation.

Moreover, multimodality for a review goes beyond numerical rating and textual content. Given the proliferation of multimedia smartphones, arming virtually everyone with a camera on hand, online reviews frequently also contain photos. When such photos are available, they often provide an illustrative backdrop to the various facets mentioned in the text, as attested to by Figure 1. Hence, we hypothesize that photos could be helpful to review generation, and consider review photos as another input to the problem.

Approach We propose a model called *Multimodal Review Generation* or *MRG*, which joins rating prediction with non-linearity and review text generation with LSTM. The joint modeling implies that the users and items' representations would be sensitive to the text generation process, and the generated text would be personalized and item-sensitive. Another novel aspect to *MRG* is incorporating visual features in the text generation. The selection of words to describe an item is challenging given the large variance among items.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313463

To supplement the sentiment features encoded in users and items' representations, we will use visual features extracted from photos to provide additional clues. Intuitively, words that could describe a photo in a review would be suitable to describe some facet of the item of interest. However, we make the deliberate design decision to use image features to help generate review text, rather than to also attempt generating the images themselves. This is because our primary intention is to model user's subjective preferences, which would be expressed much more evidently in the rating and the text.

Contributions We design the *MRG* model (see Section 3), which jointly models rating prediction and text generation at the review level by incorporating LSTM cells with a novel fusion gate as a kind of soft attention to weigh the relative contributions of sentiment features and visual features that provide context to the text generation. We also describe the learning and inference algorithms respectively. In Section 4 we conduct comprehensive experiments against baselines for both rating prediction and review generation that showcase the benefits of their joint modeling and incorporation of visual features, as well as an ablation analysis that establishes the contributions of the various components of our model.

2 RELATED WORK

In this section, we review the two groups of literature most related to our problem, namely: content-based recommendation and text generation (particularly relating to online reviews).

Content-Based Recommendation. The vast majority of recommendation works are based on modeling ratings [31], and most rely on matrix factorization [13]. These include the popular methods such as Probabilistic Matrix Factorization or PMF[22], Non-negative Matrix Factorization or NMF [15], and Singular Value Decomposition with neighborhood information or SVD++ [12], which we will use as baselines to validate the benefits of content modeling. Note that we focus on models that fit rating values, rather than pairwise rankings [8, 29] that are generally seen as a different formulation.

Later approaches seek to model textual content to supplement ratings. One direction is to apply topic modeling to item content, and relate an item's topic distribution to its latent factors for use in matrix factorization-like rating prediction. Such approaches [19, 20, 32] outperform rating-only methods. The more recent direction is to rely on neural approaches. For instance, [33] improves upon [32] by replacing the topic model with a Stacked Denoising AutoEncoder (SDAE) to learn item latent factors. ConvMF [10] learns a Convolutional Neural Network to extract latent features for an item from its text reviews. DeepCoNN [38] models user and item's review texts separately using siamese CNN and joins the two sets of features in the manner of Factorization Machine [28]. The latter three are state-of-the-art, having been compared favorably to the prior ones based on topic models, and will be used as baselines.

There are two key distinctions between ours and the approaches in this category. First, we model review-level textual content, i.e., the text is associated with the user-item tuple, rather than with the item or the user alone. Second, we seek to generate a piece of natural language text as an output, as opposed to feature vectors.

Review Text Generation. Learning to generate text is intensively investigated in natural language processing, where it can be applied to language modeling, machine translation, or speech recognition. The idea is to learn a model which can generate a sequence of text based on sequential dependencies and contexts. Long Short-Term Memory (LSTM) [9], a gated version of recurrent neural networks, is capable of capturing long-term dependencies and is demonstrably an effective approach [3, 6, 31].

The review text generation technique closest to ours is Att2Seq [5]. The work models user, item, and rating as given attributes, and learns to generate a review text through an attention mechanism. Note that we do not take the rating as a given; instead we seek to generate the rating concurrently. In Section 4, we will compare to Att2Seq as one of the review generation baselines, assessing their performance in both scenarios when the test rating is unknown (our setting) and when it is given (their original setting). Other review generation methods are not directly comparable. For instance, some expect extraneous inputs not applicable to our setting, such as generating tips based on a review [16], generating review based on pre-defined phrases [25], or considering the neighborhood reviews of a user [34]. In turn, [24] is based on pairwise ranking, which is usually seen as a different formulation from rating prediction. None makes use of visual features from images.

By incorporating visual features in the text generation, our problem is related to image captioning [35, 36]. The key distinction is our incorporation of user and item latent factors, which implies that in our case the "captioning" is effectively item-sensitive and personalized. In Section 4 we will compare to [35] as a image-based text generation representative not affected by either user or item.

Table 1: List of notations

Symbols	Description
$\mathcal{X}^r$	set of rating observations
$\mathcal{X}^s$	set of review text with images
$\mathcal{U}$	set of users
$\mathcal{I}$	set of items
$\mathcal{A}$	set of image annotations
$\mathbf{P}$	user embedding matrix
$\mathbf{Q}$	item embedding matrix
$\mathbf{E}$	word embedding matrix
$\mathbf{W}$	weight projection matrix
b	bias term
Θ	set of neural parameters

3 MULTIMODAL REVIEW GENERATION

We now describe our proposed *Multimodal Review Generation* or *MRG*, first the model architecture, then the learning and inference. Table 1 reproduces the notations for ease of reference.

Problem. The universal sets of users and items are denoted $\mathcal{U}$ and $\mathcal{I}$. We are given $\mathcal{X}^r$ and $\mathcal{X}^s$ for learning. $\mathcal{X}^r$ is the set of rating observations, whereby each observation is a triple (u, i, r) indicating that a user $u \in \mathcal{U}$ assigns to an item $i \in \mathcal{I}$ a rating score of $r \in \mathbb{R}$. In turn, $\mathcal{X}^s$ is the set of review content observations, whereby each observation is a quadruple (u, i, d, m) with d being a review text document and m is a corresponding image. We further assume that for each observation in $\mathcal{X}^r$, there is at least one corresponding observation in $\mathcal{X}^s$ for the same user and item, i.e., each rating is accompanied by at least a review text and an image.

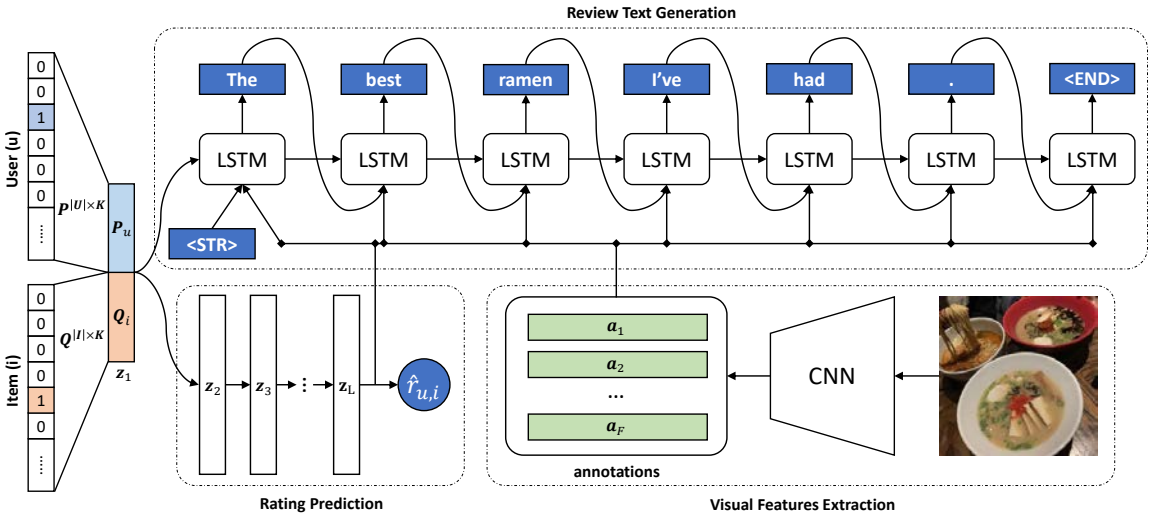


Figure 2: Overall Architecture of Multimodal Review Generation (MRG) model

The problem of multimodal review generation can thus be stated as follows. From $\mathcal{X}^r$ and $\mathcal{X}^s$, we seek to learn a model, which could output a predicted rating score $\hat{r}$ and a synthesized review document $\hat{d}$ given a user u, an item i, and an image m. The image is not a must, but could be useful. One use case is when a user is observed to have an image of an item (such as an Instagram posting), and we seek to assess the likely rating and review text. Alternatively, we could maintain a set of images reflecting different facets of an item, which could then be sampled for a given user. As we will see shortly, even when image is unavailable, the model is still capable of producing both a rating prediction and a synthesized review text.

3.1 Model Architecture

Fig. 2 shows the neural model architecture, with several components including rating prediction and review text generation, as well as visual features extraction. We describe each component in turn.

Rating Prediction with Non-Linearity. Traditionally, matrix factorization [13] models rating via a *bilinear* function, such as:

$$\hat{r}_{u,i} = \mathbf{p}_u^T \mathbf{q}_i + b_u + b_i + \mu \tag{1}$$

where $\mathbf{p}_u, \mathbf{q}_i$ are user and item latent factors, b_u, b_i are user and item biases, and μ is the global bias (average rating).

An emerging idea is to use *non-linearity* to capture the interactions between users and items. Non-linear transformations promise to learn better representations as shown for various tasks [6, 14, 21]. We propose to utilize Multilayer Perceptron or MLP as the rating prediction component. In a different formulation, MLP was combined with matrix factorization to learn pairwise ranking scores [8].

User u and item i are encoded as one-hot vectors. With the help of user embedding matrix $\mathbf{P}^{|\mathcal{U}|xK}$ and item embedding matrix $\mathbf{Q}^{|\mathcal{I}|xK}$, we project the one-hot vectors onto the user embedding vector $\mathbf{P}_u$ and item embedding vector $\mathbf{Q}_i$. These are concatenated and projected through a number L of neural layers followed by non-linear transformations, as shown below. The model outputs a rating $\hat{r}_{u,i}$ based on the representation learned from the last layer.

$$\mathbf{z}_1 = \begin{pmatrix} \mathbf{P}_u \\ \mathbf{Q}_i \end{pmatrix} \tag{2}$$

$$\mathbf{z}_2 = \phi(\mathbf{W}_2 \mathbf{z}_1 + b_2) \tag{3}$$

$$\ldots \tag{4}$$

$$\mathbf{z}_L = \phi(\mathbf{W}_L \mathbf{z}_{L-1} + b_L) \tag{5}$$

$$\hat{r}_{u,i} = \mathbf{W}_r \mathbf{z}_L + b_r \tag{6}$$

Above, $\mathbf{W}_l$ is learned projection matrix, b_l is bias of the neural layer l. In turn, ϕ is hyperbolic tangent ($tanh$) function, and $\mathbf{W}_r, b_r$ are projection matrix and bias of the output layer.

Review Text Generation with Sensitivity to User and Item. The text generation component (upper part of Fig. 2) is built on LSTM. First, we describe how we generate text as a language model with only textual contexts. Then, we introduce the notion of user and item with sentiment information from the rating prediction component. Later on, we extend it further with visual information.

Let $d = \{w_1, \ldots, w_T\}$ be a review text document, where w_t is the word at position t. We encode each word w_t as an one-hot vector $\mathbf{y}_t \in \mathbb{R}^V$, where V is the vocabulary size. Thus, document d becomes a sequence of one-hot vectors $\mathbf{y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_T\}$. We implement our LSTM (see Fig. 3) according to following equations:

$$\mathbf{x}_t = \mathbf{E}\mathbf{y}_{t-1} \tag{7}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{fx}\mathbf{x}_t + \mathbf{W}_{fh}\mathbf{h}_{t-1} + b_f) \tag{8}$$

$$\mathbf{i}_t = \sigma(\mathbf{W}_{ix}\mathbf{x}_t + \mathbf{W}_{ih}\mathbf{h}_{t-1} + b_i) \tag{9}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{ox}\mathbf{x}_t + \mathbf{W}_{oh}\mathbf{h}_{t-1} + b_o) \tag{10}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \phi(\mathbf{W}_{cx}\mathbf{x}_t + \mathbf{W}_{ch}\mathbf{h}_{t-1} + b_c) \tag{11}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \phi(\mathbf{c}_t) \tag{12}$$

where $\mathbf{E} \in \mathbb{R}^{mxV}$ is the learned word embedding matrix initialized randomly or from pre-trained word embeddings [21, 27], $\mathbf{W}_*, b_*$ are learned projection matrices and biases initialized randomly, and σ, ϕ are *sigmoid* and *tanh* activation functions respectively.

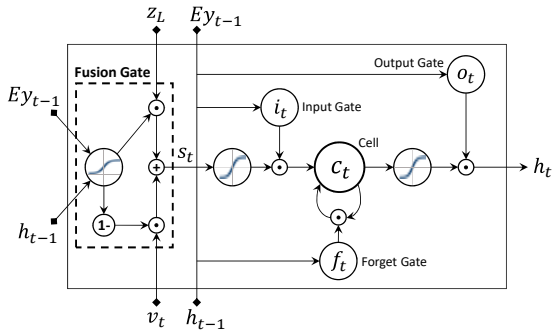


Figure 3: LSTM Cell with Fusion Gate

Because we seek to generate a document that would be relevant to the item at hand yet sensitive to user preferences, we initialize the LSTM with user embedding $\mathbf{P}_u$ and item embedding $\mathbf{Q}_i$.

$$\mathbf{c}_0 = \phi(\mathbf{W}_{c0}\mathbf{z}_1 + b_{c0}) \tag{13}$$

$$\mathbf{h}_0 = \phi(\mathbf{W}_{h0}\mathbf{z}_1 + b_{h0}) \tag{14}$$

Above, $\mathbf{z}_1$ is the concatenated vector of $\mathbf{P}_u$ and $\mathbf{Q}_i$ (Eq. 2). $\mathbf{W}_{c0}, \mathbf{W}_{h0} \in \mathbb{R}^{n\times 2K}$ and b_{c0}, b_{h0} are learned projection matrices and biases. n is the number of hidden dimensions of the LSTM state.

By sharing the user and item embeddings between the rating prediction and review text generation components, we seek shared representations that will be beneficial to both components [2, 23]. Moreover, in addition to sequential dependencies, we supply the LSTM with sentiment information. In other words, at each time step t, in addition to the generated word from $t-1$, we use $\mathbf{z}_L$ (Eq. 5) as another input source of the LSTM by simply concatenating it with previous generated word embedding:

$$\mathbf{x}_t = \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{z}_L \end{pmatrix} \tag{15}$$

The intuition to use $\mathbf{z}_L$ is because the final representation of the rating prediction component is hypothesized to be the most representative features of the sentiment of the review.

Review Text Generation with Visual Features. Based on the previously described components, we would already meet the dual objectives of rating prediction and review text generation. In some cases, additional information in the form of photo(s) may be available. For instance, a user may not have reviewed an item, but she may have an Instagram posting about the item. In such cases, we hypothesize that visual features would provide additional clues that could improve the review text generation further.

To extract feature vectors, also referred to as *annotations*, from images, we use convolutional neural network or CNN. For each image m, we get a set of annotation vectors $\mathcal{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_F\}, a_j \in \mathbb{R}^D$, where F is the number of annotations.

$$\mathcal{A} = \text{CNN}(m) \tag{16}$$

Our *MRG* architecture employs 19-layer Oxford VGGNet [30] as the visual feature extractor. We use the pre-trained model of VGGNet on ImageNet [4] without finetuning, and treat the $14 \times 14 \times 512$ feature map output from the fifth convolutional layer (conv5_3) as our image annotations. In consequence, each image will be transformed into 196 annotation vectors $\mathbf{a}_j$ with $D = 512$.

At each time step t, we want to supply our LSTM a context vector as a visual clue for generating the next word which is relevant to the image. Each annotation $\mathbf{a}_j$ is a compression of a small region of the image. Not all annotations are contributing visual information equally. Therefore, we apply soft attention mechanism [3, 35] to allow the model to weigh on the more relevant annotations. Thus, the visual context vector $\mathbf{v}_t$ is obtained by the following equations:

$$\mathbf{e}_{tj} = \phi(\mathbf{W}_{ea}\mathbf{a}_j + \mathbf{W}_{eh}\mathbf{h}_{t-1} + b_e) \tag{17}$$

$$\alpha_{tj} = \frac{\exp(\mathbf{W}_\alpha \mathbf{e}_{tj})}{\sum_{k=1}^{F}\exp(\mathbf{W}_\alpha \mathbf{e}_{tk})} \tag{18}$$

$$\mathbf{v}_t = \sum_{j=1}^{F} \alpha_{tj}\mathbf{a}_j \tag{19}$$

where $\mathbf{W}_{ea} \in \mathbb{R}^{D\times D}, \mathbf{W}_{eh} \in \mathbb{R}^{D\times n}, \mathbf{W}_\alpha \in \mathbb{R}^{1\times D}$, and b_e are learned projection matrices and bias randomly initialized.

Fusion Gate as Soft Attention between Sentiment and Visual Information. In addition to $\mathbf{z}_L$, the visual context vector $\mathbf{v}_t$ is treated as another source of input for our LSTM. Although we hypothesize that each generated word is either based on sentiment or visual information, we relax that constraint by allowing the model to make soft decisions. We introduce a fusion gate (Fig. 3), which can be seen as an attention mechanism for the LSTM. The fusion gate will learn to give the importance weights for sentiment features $\mathbf{z}_L$ and visual features $\mathbf{v}_t$ and combine them into a final context vector $\mathbf{s}_t$. More information from visual features means less information from sentiment features going through the gate.

$$\gamma_t = \sigma(\mathbf{W}_{\gamma y}\mathbf{E}\mathbf{y}_{t-1} + \mathbf{W}_{\gamma h}\mathbf{h}_{t-1} + b_\gamma) \tag{20}$$

$$\mathbf{s}_t = \gamma_t \mathbf{z}_L + (1-\gamma_t)\mathbf{v}_t \tag{21}$$

$$\mathbf{x}_t = \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{s}_t \end{pmatrix} \tag{22}$$

$\mathbf{W}_{\gamma y} \in \mathbb{R}^{1\times m}, \mathbf{W}_{\gamma h} \in \mathbb{R}^{1\times n}$, and b_γ are learned projection matrices and bias randomly initialized. Constructing the fusion gate requires the number of dimensions of $\mathbf{z}_L$ and $\mathbf{v}_t$ to be equal. Thus, we make sure the rating prediction component outputs $\mathbf{z}_L \in \mathbb{R}^D$.

The output probabilities are computed based on previous word, the LSTM state, and the context vector from fusion gate:

$$\mathbf{g}_t = \mathbf{W}_g\phi(\mathbf{W}_{gy}\mathbf{E}\mathbf{y}_{t-1} + \mathbf{W}_{gh}\mathbf{h}_t + \mathbf{W}_{gs}\mathbf{s}_t) \tag{23}$$

$$p(\mathbf{y}_{tj}|\mathbf{x}_t) = \frac{\exp(\mathbf{g}_{tj})}{\sum_{k=1}^{V}\exp(\mathbf{g}_{tk})} \tag{24}$$

$\mathbf{W}_g \in \mathbb{R}^{V\times m}, \mathbf{W}_{gy} \in \mathbb{R}^{m\times m}, \mathbf{W}_{gh} \in \mathbb{R}^{m\times n}, \mathbf{W}_{gs} \in \mathbb{R}^{m\times D}$ are learned projection matrices randomly initialized. The word with the highest output probability will be chosen as the generated one.

$$\mathbf{y}_t^* = \underset{\mathbf{y}_j, j\in V}{\text{argmax}}\, p(\mathbf{y}_{tj}|\mathbf{x}_t) \tag{25}$$

Discussion. The relative generality of *MRG* can be seen from how the full-fledged model could be reduced into simplified models that work with less information. As earlier stated, the model is still capable of generating rating prediction and synthesizing review text without visual features. Without rating prediction, the model is effectively solving a personalized image captioning problem where the generated text will describe the image based on the user's experience of the item. If both visual features and rating prediction

Algorithm 1 Parameter Learning with Stochastic Gradient Descent

Input: $\mathcal{U}, \mathcal{I}, \mathcal{X}^r, \mathcal{X}^s$, learning rate η
Output: learned model parameters $\{\mathbf{P}, \mathbf{Q}, \mathbf{E}, \Theta_r, \Theta_s\}$

initialization
  $\mathbf{P}, \mathbf{Q}, \Theta_r, \Theta_s \leftarrow$ randomly initialized
  $\mathbf{E} \leftarrow$ randomly initialized / pre-trained embeddings
while not converged **do**
  for all $(u, i, r) \in \mathcal{X}^r$ **do**
    $\mathbf{P}_u = \mathbf{P}_u - \eta \cdot \frac{\partial}{\partial \mathbf{P}_u} \mathcal{L}^r(u, i, r)$
    $\mathbf{Q}_i = \mathbf{Q}_i - \eta \cdot \frac{\partial}{\partial \mathbf{Q}_i} \mathcal{L}^r(u, i, r)$
    $\Theta_r = \Theta_r - \eta \cdot \frac{\partial}{\partial \Theta_r} \mathcal{L}^r(u, i, r)$
    for all $(j, k, d, m) \in \mathcal{X}^s$ **where** $(j == u)$ & $(k == i)$ **do**
      $\mathbf{P}_j = \mathbf{P}_j - \eta \cdot \frac{\partial}{\partial \mathbf{P}_j} \mathcal{L}^s(j, k, d, m)$
      $\mathbf{Q}_k = \mathbf{Q}_k - \eta \cdot \frac{\partial}{\partial \mathbf{Q}_k} \mathcal{L}^s(j, k, d, m)$
      $\Theta_s = \Theta_s - \eta \cdot \frac{\partial}{\partial \Theta_s} \mathcal{L}^s(j, k, d, m)$
      $\mathbf{E}_{w \in d} = \mathbf{E}_{w \in d} - \eta \cdot \frac{\partial}{\partial \mathbf{E}_{w \in d}} \mathcal{L}^s(j, k, d, m)$
    end for
  end for
end while
return $\{\mathbf{P}, \mathbf{Q}, \mathbf{E}, \Theta_r, \Theta_s\}$

are removed, the model is solving the problem of personalized review text generation without any further information. If the goal is solely to predict ratings, both visual features and review text can be removed, leaving the model with only rating prediction. In Section 4, we will investigate the respective contributions of various components towards the final model as well as the performance of different architectural variants in an ablation analysis.

3.2 Model Learning and Inference

Learning. Our model consists of two supervised learnable components. To learn the parameters based on rating information, the model minimizes the regularized square error on the set $\mathcal{X}^r$:

$$\mathcal{L}^r = \frac{1}{2} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbb{1}_{u,i} (r_{u,i} - \hat{r}_{u,i})^2 + \frac{\lambda_{wr}}{2} \sum_{\theta \in \Theta_r} \|\theta\|_2^2 \|\theta\|_2^2 + \frac{\lambda_{ur}}{2} \sum_{u \in \mathcal{U}} \|\mathbf{P}_u\|_F^2 + \frac{\lambda_{ir}}{2} \sum_{i \in \mathcal{I}} \|\mathbf{Q}_i\|_F^2$$

$\mathbb{1}_{u,i}$ is the indicator function which is equal to 1 if user u rated item i and equal to 0 otherwise, $\|.\|$ denotes the Frobenius norm of matrix, Θ_r is the set of parameters of rating prediction component, and $\lambda_{wr}, \lambda_{ur}, \lambda_{ir}$ are hyper-paramenters for regularization.

To learn the review text generator, we update the parameters by minimizing the regularized negative log-likehood on the set $\mathcal{X}^s$:

$$\mathcal{L}^s = \sum_{(u,i,d,m) \in \mathcal{X}^s} \sum_{w \in d} -\log p(\mathbf{y}_w | \mathbf{x}) + \frac{\lambda_{ws}}{2} \sum_{\theta \in \Theta_s} \|\theta\|_2^2 + \frac{\lambda_{us}}{2} \sum_{u \in \mathcal{U}} \|\mathbf{P}_u\|_F^2 + \frac{\lambda_{is}}{2} \sum_{i \in \mathcal{I}} \|\mathbf{Q}_i\|_F^2$$

where Θ_s is the set of parameters of rating prediction component, and $\lambda_{ws}, \lambda_{us}, \lambda_{is}$ are hyper-paramenters for regularization.

Algorithm 2 Multimodal Review Generation with Greedy Search

Input: user (u), item (i), image (m)
Output: estimated rating ($\hat{r}$), generated review text ($\hat{d}$)

initialization
  $\mathbf{P}_u, \mathbf{Q}_i \leftarrow$ user and item embeddings
  $t = 0$, $\mathbf{T} \leftarrow$ maximum review length
  $\Pi = [] \leftarrow$ generated word array
  $w_0 =$ "<start>", $\mathbf{y}_0 = \mathbf{y}_{w_0} \leftarrow$ starting word
  $\mathbf{c}_0, \mathbf{h}_0 \leftarrow$ LSTM initialized states from $[\mathbf{P}_u; \mathbf{Q}_i]$
  $\mathcal{A}_m = \text{CNN}(m) \leftarrow$ generated annotations from image m
$\mathbf{z}_L \leftarrow$ generated sentiment features from $[\mathbf{P}_u; \mathbf{Q}_i]$
$\hat{r} = \mathbf{W}_r \mathbf{z}_L + b_r$
while $t < T$ **do**
  $t = t + 1$
  $\mathbf{E}\mathbf{y}_{t-1} \leftarrow$ input word embedding
  $\mathbf{v}_t \leftarrow$ generated visual context from $\mathcal{A}_m$
  $\mathbf{s}_t \leftarrow$ generated fusion context from $\mathbf{z}_L$ and $\mathbf{v}_t$
  $\mathbf{x}_t = [\mathbf{E}\mathbf{y}_{t-1}; \mathbf{s}_t] \leftarrow$ input of the LSTM
  $\mathbf{y}_t = \underset{\mathbf{y}_j, j \in V}{\text{argmax}}\, p(\mathbf{y}_{tj} | \mathbf{x}_t) \leftarrow$ greedy selection
  $w_t = \text{map_to_word}(\mathbf{y}_t)$
  if $w_t ==$ "<end>" **then**
    break
  end if
  Π.append(w_t)
end while
$\hat{d} = \text{join}(\Pi)$
return $\{\hat{r}, \hat{d}\}$

The final objective of the model is the weighted combination of $\mathcal{L}^r$ and $\mathcal{L}^s$:

$$\mathcal{J} = \underset{\mathbf{P}, \mathbf{Q}, \mathbf{E}, \Theta_r, \Theta_s}{\text{minimize}} \Big(\frac{\lambda_r}{2} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbb{1}_{u,i} (r_{u,i} - \hat{r}_{u,i})^2 + \lambda_s \sum_{(u,i,d,m) \in \mathcal{X}^s} \sum_{w \in d} -\log p(\mathbf{y}_w | \mathbf{x}) + \frac{\lambda_{wr}}{2} \sum_{\theta \in \Theta_r} \|\theta\|_2^2 + \frac{\lambda_{ws}}{2} \sum_{\theta \in \Theta_s} \|\theta\|_2^2 + \frac{\lambda_u}{2} \sum_{u \in \mathcal{U}} \|\mathbf{P}_u\|_F^2 + \frac{\lambda_i}{2} \sum_{i \in \mathcal{I}} \|\mathbf{Q}_i\|_F^2 \Big)$$

λ_r, λ_s are hyper-parameters to balance the importance between two components, $\lambda_{wr}, \lambda_{ws}, \lambda_u, \lambda_i$ are hyper-parameters for regularization of Θ_r, Θ_s, user embeddings, item embeddings respectively.

We derive a learning algorithm based on stochastic gradient descent using backpropagation (Alg. 1). Each triple (u, i, r) in $\mathcal{X}^r$ may be associated with quadruples (u, i, d, m) in $\mathcal{X}^s$. For each observation in $\mathcal{X}^r$, we update user, item embeddings and parameters in Θ_r by minimizing $\mathcal{L}^r$. With each $(u, i, r) \in \mathcal{X}^r$, we find corresponding $(u, i, d, m) \in \mathcal{X}^s$, and update user, item embeddings, parameters in Θ_s, word embeddings by minimizing $\mathcal{L}^s$. One might argue that the model can be updated based on observations from $\mathcal{X}^s$ before $\mathcal{X}^r$. We update rating prediction first because review text generation requires features from the rating, supplying the LSTM with more accurate sentiment signals. We can speed up the training with mini-batch gradient descent yielding a faster rate of convergence.

Table 2: Data Statistics

Data	#users	#items	#ratings	#docs	#images
CH	2,908	2,725	19,453	82,283	38,978
LA	36,918	23,601	355,553	1,539,355	680,892
NY	21,474	15,160	199,723	818,682	382,368
SF	7,999	3,375	65,228	320,165	139,014

Inference. To generate a review given a user u, an item i and an image m, we derive an inference algorithm for our *MRG* model (see Alg. 2). The rating $\hat{r}_{u,i}$ is first estimated using rating prediction component that also outputs the sentiment features $\mathbf{z}_L$. The review text generation is based on greedy strategy, where at each time step t, the word with the highest probability is chosen for the review.

4 EXPERIMENTS

Our experimental objective is to investigate the following research questions with respect to the proposed model *MRG*.

- **RQ#1:** Does factoring content via the review generation task improve the performance of *MRG* in rating prediction compared to rating-only matrix factorization approaches?
- **RQ#2:** Does coupling rating prediction and text generation at the review level help *MRG* to outperform comparable content-based recommender systems in rating prediction?
- **RQ#3:** Does factoring sentiment and visual features help *MRG* to generate review texts closer to the ground truth?
- **RQ#4:** How do the different components of *MRG* contribute to its performance? How do different architecture variants perform in rating prediction and review text generation?

4.1 Experimental Setup

Datasets. We use datasets of online reviews crawled from *Yelp* covering 4 US cities, namely: Chicago (CH), Los Angeles (LA), New York (NY), and San Francisco (SF). Table 2 shows some statistics about these datasets. Within each online review, there is a rating, a review text, and one or more images taken by the user. The set $\mathcal{X}^r$ consists of all rating observations. The set $\mathcal{X}^s$ is constructed as follows. First, we split the review text into shorter passages; in this case, each sentence is considered a review document d. A review image m could be paired with multiple documents. To identify the best-matching documents to an image, we rank sentences within the same review based on the cosine similarity of their TF-IDF scores to that of the user-provided image description, and consider up to 5 highest-ranked documents above a threshold (0.1) to be relevant. These would then form the (u, i, d, m) quadruples in $\mathcal{X}^s$.

Task #1: Rating Prediction. One evaluation task is rating prediction. Since *Yelp* uses rating scores $r_{u,i}$ from 1 to 5, our model learns to estimate ratings within that range. We rely on two standard metrics for rating prediction, namely: Mean Absolute Error (*MAE*) and Root Mean Square Error (*RMSE*) as shown below.

$$MAE = \frac{1}{N} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbb{1}_{u,i} |r_{u,i} - \hat{r}_{u,i}|$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}} \mathbb{1}_{u,i} (r_{u,i} - \hat{r}_{u,i})^2}$$

Table 3: Hyper-parameter settings

Method	Hyper-parameters
PMF	latent factors = 10, $\lambda_U = \lambda_V = 0.1$
NMF	latent factors = 30
SVD++	latent factors = 10, $\lambda = 0.01$
CDL	$K = 50, \lambda_v = 10, \lambda_n = 1000, \lambda_u = \lambda_w = 0.1$
ConvMF	$k = 50, p = 200, \lambda_u = 1, \lambda_v = 100$
DeepCoNN	$\lvert x_u \rvert = \lvert y_i \rvert = 50, c = 200, t = 3$
Att2Seq	$m = 200, r = 200, n = 256, L = 2$
SAT	$m = 200, n = 256, \lambda = 1$
MRG	$K = 200, m = 200, n = 256, L = 2, \lambda_r = \lambda_s = 1.0, \lambda_{wr} = \lambda_{ws} = \lambda_u = \lambda_i = 0.0001$

$\mathbb{1}_{u,i}$ is the indicator function which is equal to 1 if user u rated item i in the testing set and equal to 0 otherwise, N indicates the total number of rating observations in the testing set.

For RQ#1 to validate the use of content, we compare *MRG* with the following rating-based matrix factorization methods.

- **PMF**: Probabilistic Matrix Factorization [22] models matrix factorization under probabilistic framework with the assumptions of Gaussian distribution for rating values.
- **NMF**: Non-negative Matrix Factorization [15] ensures that the factorized user and item matrices are non-negative.
- **SVD++**: Singular Value Decomposition [12] learns from ratings as well as considers neighborhood information.

For RQ#2 to validate the modeling of content at review level, we compare *MRG* with the following content-based recommendation methods that incorporate textual content at the item level.

- **CDL**: Collaborative Deep Learning [33] uses Stacked Denoising AutoEncoder (SDAE) to learn item features.
- **ConvMF**: Convolutional Matrix Factorization [10] replaces the SDAE with a convolutional neural network (CNN).
- **DeepCoNN**: [38] models both user and item features from aggregated review texts using siamese CNN.

Task #2: Review Text Generation. The other evaluation task is generating review text. BLEU [26] is a well-known metric for evaluating the quality of generated text, which has been widely used for machine translation and image captioning. We use smoothed BLEU [18] and report the results of BLEU scores from 1 to 4. Another metric is ROUGE [17] which has been extensively used for text summarization. We report the F-measure, which is the geometric mean of the precision and recall of ROUGE-1 (covering 1-grams) and ROUGE-L (covering the longest subsequences) respectively.

A review may be associated with multiple photos. We therefore evaluate the generated text under two scenarios: photo level and rating level. Photo level means the generated text for each photo is evaluated individually, and we report the average results over all the photos. Rating level will first aggregate (i.e., average) the results associated with all the photos of the same review, and then we report the average results across all the ratings. For the latter scenario, the ground-truth references corresponding to the photos of the same rating are merged. The purpose is to make the comparison fair for the baselines not using visual information.

Table 4: Review text generation performance evaluated @ photo level (*higher is better*)

	Metric	Att2Seq		SAT	MRG
		-Rating	*+Rating*		
Chicago	**BLEU-1**	29.07	29.49	29.89	**33.34**
	BLEU-2	15.57	15.79	15.46	**17.28**
	BLEU-3	11.87	11.88	11.75	**12.53**
	BLEU-4	10.51	10.41	10.42	**10.77**
	ROUGE-1	22.55	22.19	24.13	**25.17**
	ROUGE-L	17.21	16.72	17.77	**18.43**
Los Angeles	**BLEU-1**	30.11	30.67	31.63	**32.37**
	BLEU-2	16.95	17.22	17.20	**17.98**
	BLEU-3	12.98	13.15	13.16	**13.64**
	BLEU-4	11.40	11.55	11.57	**12.01**
	ROUGE-1	21.86	22.99	23.38	**24.22**
	ROUGE-L	16.65	17.71	17.96	**18.50**
New York	**BLEU-1**	28.95	29.40	29.68	**30.67**
	BLEU-2	16.24	15.75	16.09	**16.92**
	BLEU-3	12.46	11.85	12.30	**12.86**
	BLEU-4	11.00	10.26	10.82	**11.28**
	ROUGE-1	22.25	22.11	23.46	**24.09**
	ROUGE-L	16.96	16.65	17.83	**18.35**
San Francisco	**BLEU-1**	30.19	30.74	30.64	**32.24**
	BLEU-2	16.86	17.15	16.66	**17.76**
	BLEU-3	12.90	12.94	12.55	**13.31**
	BLEU-4	11.34	11.25	10.90	**11.60**
	ROUGE-1	22.17	21.83	22.25	**25.20**
	ROUGE-L	16.77	16.63	16.91	**18.39**

Table 5: Review text generation performance evaluated @ rating level (*higher is better*)

	Metric	Att2Seq		SAT	MRG
		-Rating	*+Rating*		
Chicago	**BLEU-1**	36.84	37.12	37.75	**42.00**
	BLEU-2	19.60	19.74	19.30	**21.74**
	BLEU-3	14.47	14.35	14.19	**15.16**
	BLEU-4	12.52	12.27	12.31	**13.04**
	ROUGE-1	22.41	21.96	24.03	**24.93**
	ROUGE-L	17.11	16.55	17.64	**18.18**
Los Angeles	**BLEU-1**	36.86	37.76	38.26	**39.38**
	BLEU-2	20.72	21.21	20.58	**21.94**
	BLEU-3	15.47	15.79	15.38	**16.30**
	BLEU-4	13.32	13.58	13.29	**14.07**
	ROUGE-1	21.61	22.89	22.66	**23.74**
	ROUGE-L	16.41	17.52	17.34	**18.08**
New York	**BLEU-1**	35.94	36.09	36.37	**38.03**
	BLEU-2	20.18	19.33	19.53	**20.97**
	BLEU-3	15.11	14.15	14.58	**15.51**
	BLEU-4	13.08	12.00	12.62	**13.32**
	ROUGE-1	21.88	21.96	22.84	**23.79**
	ROUGE-L	16.63	16.55	17.33	**18.11**
San Francisco	**BLEU-1**	39.55	39.83	38.78	**41.51**
	BLEU-2	22.10	22.25	20.81	**22.96**
	BLEU-3	16.39	16.26	15.20	**16.75**
	BLEU-4	14.03	13.76	12.93	**14.29**
	ROUGE-1	22.19	22.12	21.86	**25.54**
	ROUGE-L	16.76	16.85	16.62	**18.55**

For RQ#3, to validate review text generation, we compare with two text generation methods constructing three baselines.

- **Att2Seq**: Attribute-to-Sequence [5] treats user, item, and rating as three attributes and generates review text with the attention-based multilayer LSTMs. We compare with two versions of Att2Seq which are: without observing rating (*-Rating*) that is similar to our setting and with observed rating (*+Rating*) that is the authors' original setting (which confers it an advantage since the ground-truth rating is given).
- **SAT**: Show, Attend, and Tell [35] is the state-of-the-art method for image captioning, which exploits visual features using attention mechanism and generates the captions using LSTM.

Experimental Settings. For each dataset in Table 2, we reserve 10% of the data for model validation. The remaining 90% is split into training and testing sets. For evaluating rating prediction, we report results for different training percentages (80%, 60%, 40%, and 20%). This is to examine the effects of sparsity on rating prediction, which is a crucial issue for recommender systems. For evaluating review text generation, we use the same split, yet we report the results for 80% training and 20% testing due to space limitations.

We tune all the comparative models for their respective best performance. Table 3 indicates the discovered hyper-parameter settings used for different methods. For PMF, NMF, and SVD++, we use grid search to find the number of latent factors from $\{10, \ldots, 100\}$ where we end up with 10 for PMF, 30 for NMF, and 10 for SVD++. The regularization hyper-parameters are searched from $\{0.001, 0.01, 0.1, 1.0\}$.

For CDL, ConvMF, and DeepCoNN, the number of item features is grid-searched from $\{10, 50, 100, 200\}$ and 50 gives the best performance and fair comparison between three methods, whereas $\{100, 200\}$ cause overfitting. Regularization hyper-parameters of the models are tuned using the validation set.

For LSTM-based models Att2Seq, SAT, and *MRG*, we use the same number of dimensions of the LSTM cells (256). SAT and *MRG* use one-layer LSTM whereas Att2Seq uses two layers of LSTMs as we respect the original design of the authors. In practice, we observe negligible improvements of using more than one layer for *MRG* so we keep using one layer for the gain in efficiency. Att2Seq and *MRG* use the same number of dimensions for user and item embeddings as well as rating embeddings for the former. Other hyper-parameters are tuned using the validation set. In testing, the maximum length of generated text is set at $T = 20$ for all the methods. *MRG* [1] is implemented using Tensorflow [1] and trained with batch size of 64. We update the parameters using Adam [11] rule with learning rate $\eta = 0.0003$, $\beta_1 = 0.9$, $\beta_2 = 0.99$, and $\epsilon = 10^{-8}$.

The vocabulary is built from words appearing at least 3 times in training and validation sets. Infrequent words are replaced with special <UNK> token. All models using word embeddings are initialized from Glove [27] pre-trained word embeddings of 200 dimensions.

[1] https://code.preferred.ai/mrg

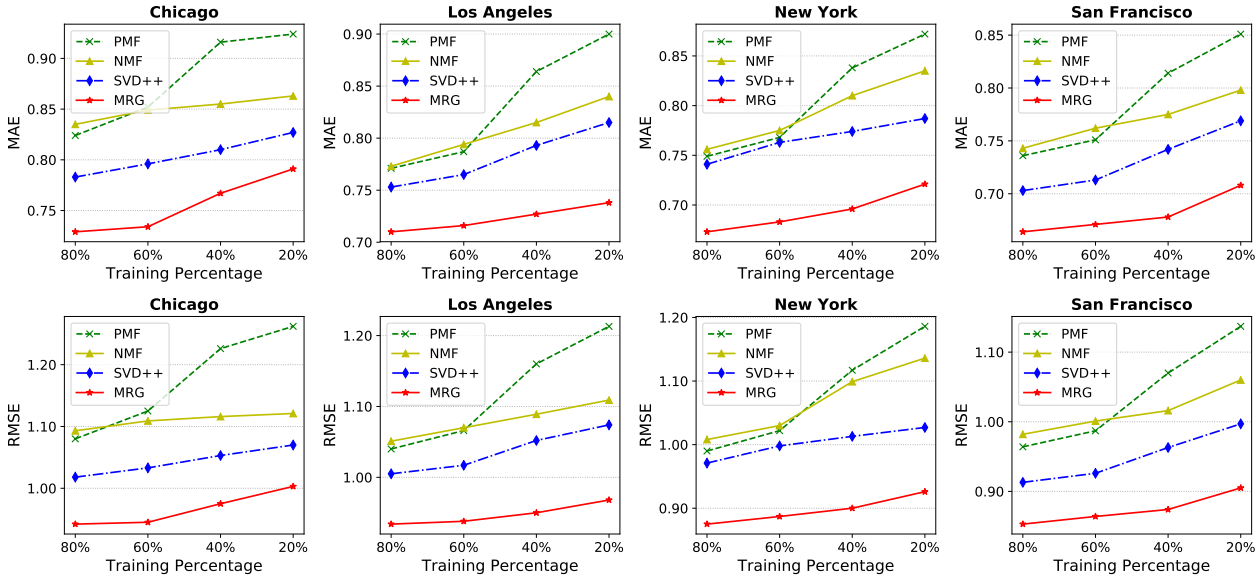


Figure 4: Rating prediction comparision with content-based approaches (*lower is better*) (The results are statistically significant with $p < 0.01$ based on the paired sample t-test)

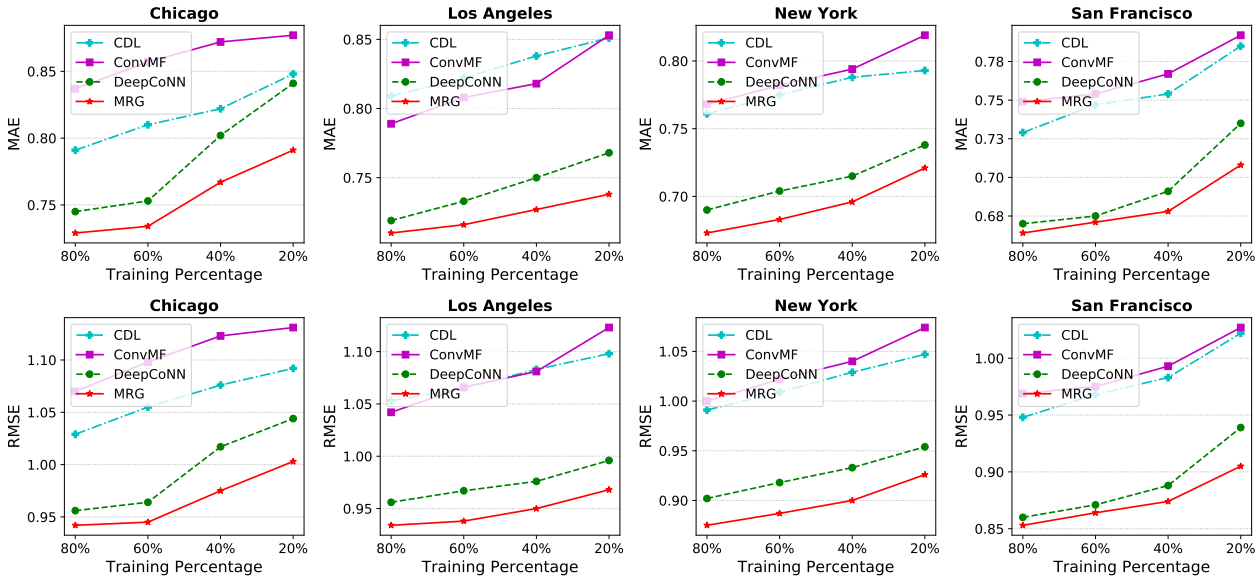


Figure 5: Rating prediction comparision with content-based approaches (*lower is better*) (The results are statistically significant with $p < 0.01$ based on the paired sample t-test)

4.2 Comparison to Baselines (RQ#1 to RQ#3)

We now compare the relative performance of *MRG* with respect to the baselines, corresponding to the first three research questions.

RQ#1. First, we consider the effects of using content for rating prediction. Figure 4 shows the performance of *MRG* in rating prediction compared to matrix factorization approaches as we vary the degree of sparsity. Lower MAE or RMSE implies higher accuracy. Expectedly, with less training data, the performance generally worsens because greater sparsity may cause overfitting.

In relative terms, the clear consensus across four datasets and two metrics is that *MRG* outperforms the baselines. This validates the advantage of modeling content in addition to ratings.

Among the three baselines, SVD++ with the advantage of modeling neighborhood information tends to perform better than PMF and NMF. In turn, PMF performs slightly better than NMF with 80% data in training, but deteriorates quickly with more sparsity.

RQ#2. We now compare *MRG* with content-based recommendation methods, namely: CDL, ConvMF, and DeepCoNN. Figure 5 illustrates the performance of these models. Here, it is also evident that *MRG* outperforms the baselines, which we attribute to its modeling content at the review level. In contrast, CDL and ConvMF that model content at item level (aggregating the reviews of an item into one document) tend to perform similarly across the datasets, with the exception of Chicago where CDL seems to have an edge. In turn, DeepCoNN performs better than both CDL and ConvMF, because DeepCoNN exploits review texts for both item and user.

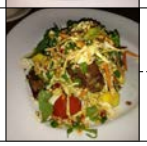
RQ#3. Table 4 and Table 5 show the results of review text generation at photo level and rating level respectively with 80% training. The baselines are text generation methods Att2Seq that relies on user, item, rating, and SAT that relies on image. For all the datasets, *MRG* consistently has higher BLEU and ROUGE scores than the baselines, indicating that it synthesizes review texts that are closer to the ground-truth references. This helps to validate the joint modeling of rating prediction and review text generation.

Examining the baselines, we see that for Chicago, Los Angeles, and San Francisco, Att2Seq with rating (*+Rating*) performs slightly better than without rating (*-Rating*). That indicates that sentiment information encoded in the rating does contribute to review text generation. In some cases, SAT that uses image information is slightly better than Att2Seq (no image). However, this is an imperfect comparison as Att2Seq factors in user and item, whereas SAT does not. For a clearer sense, we conduct an ablation analysis.

4.3 Ablation Analysis (RQ#4)

To investigate the respective contribution of our architecture components to the overall performance of *MRG*, we conduct an ablation analysis. Table 6 shows the rating prediction performance of our model when we systematically remove different components as discussed at the end of Section 3.1. The ticks indicate which information is included. The model with rating prediction alone (*User + Item*) is the worst, performing at similar levels as the matrix factorization baselines. The review text generation component (*Text*) sharply boosts the results because of the strongly evident preference information from text. When the visual features (*Photo*) are introduced, the sentiment features are aligned more accurately with the sentiment words during the text generation process, which in turn benefits the learned representations of the rating prediction. The improvements are even clearer as the data becomes sparser.

Table 7 and Table 8 show the review text generation quality at the photo level and the rating level respectively. The model with only (*User + Item*) performs the worst, as the generated text becomes generic because of the loss of the guiding information during generation process. Adding either the sentiment signal (*Rating*) or the visual signal (*Photo*) individually improves the results, indicating that these components are useful. The gain of adding *Rating* is relatively higher than adding *Photo*. The final model that combines both signals consistently outperforms the rest, demonstrating the utility of joining these components in the overall architecture. Due to space limitation, we omit BLEU-2 and BLEU-3 measurements as both of them show the same trend as discussed.

	Photo	Rating	Review
Image 1		**4.1**	**the steak was cooked perfectly .**
		4.0	order the medium rare my favorite and you will have yourself a big , fat , and juicy steak to shove between your big smile .
Image 2		**4.1**	**the beef was cooked perfectly .**
		4.0	i 'll recommend the thai steak and noodle salad to you .

(a) Case Study #1: A user (Ronald "Exotic food consumer" L.) reviews an item (Hillstone) with different images.

	Photo	Rating	Review
Ellen "FuZe" Z.		**4.5**	**the clam chowder was good .**
		5.0	best clam chowder i 've ever had .
Young Y.		**3.4**	**the clam chowder was a bit too salty .**
		3.0	the boston clam chowder was pretty salty and i 've had lots of clam chowder before .

(b) Case Study #2: An item (Tadich Grill) is reviewed by two users with different sentiments.

	Photo	Rating	Review
Philz Coffee		**4.6**	**i 've had a few times for the best breakfast sandwich .**
		4.0	the avocado toast was surprisingly good .
A16		**4.2**	**i was n't sure to try the pizza .**
		3.0	i think i might want to try their other pizzas which might be better tasting than their funghi .

(c) Case Study #3: A user (Rodney "Hungry Trikker" H.) reviews two different items.

Figure 6: Multimodal review generation. The first line next to each photo (bold) is generated rating & text, and the second line is the ground truth. Photos are best seen in color.

4.4 Case Studies

To gain an intuitive sense of the workings of the *MRG* model, we illustrate several examples of the rating prediction as well as the synthesized text generated by *MRG*. The first case study in Fig. 6a examines the text generated for two photos by the same user for an item. The predicted rating (bold) is close to the ground-truth and is the same for both photos since it is for the same item. The predicted texts (bold) also have some similarity to the ground-truth. Interestingly, the texts corresponding to the photos are different

Table 6: Ablation analysis: rating prediction performance (*lower is better*)
(The results are statistically significant with $p < 0.01$ based on the paired sample t-test)

	Dataset	Chicago			Los Angeles			New York			San Francisco		
	User + Item	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Text		✓	✓		✓	✓		✓	✓		✓	✓
	Photo			✓			✓			✓			✓
MAE	Training 80%	0.751	0.741	**0.729**	0.749	0.713	**0.710**	0.695	0.677	**0.673**	0.685	0.667	**0.664**
	Training 60%	0.779	0.751	**0.734**	0.766	0.719	**0.716**	0.717	0.693	**0.683**	0.703	0.678	**0.671**
	Training 40%	0.832	0.788	**0.767**	0.801	0.732	**0.727**	0.754	0.704	**0.696**	0.743	0.686	**0.678**
	Training 20%	0.920	0.817	**0.791**	0.881	0.746	**0.738**	0.820	0.740	**0.721**	0.811	0.728	**0.708**
RMSE	Training 80%	1.006	0.952	**0.942**	0.985	0.936	**0.934**	0.912	0.881	**0.875**	0.889	0.857	**0.853**
	Training 60%	1.032	0.961	**0.945**	1.004	0.943	**0.938**	0.937	0.900	**0.887**	0.941	0.872	**0.864**
	Training 40%	1.083	1.010	**0.975**	1.043	0.957	**0.950**	0.981	0.910	**0.900**	0.989	0.879	**0.874**
	Training 20%	1.165	1.039	**1.003**	1.126	0.988	**0.968**	1.082	0.953	**0.926**	1.058	0.932	**0.905**

Table 7: Ablation analysis: review text generation performance evaluated @ photo level (*higher is better*)

	User + Item	✓	✓	✓	✓
	Rating		✓		✓
	Photo			✓	✓
CH	**BLEU-1**	29.98	32.16	32.19	**33.34**
	BLEU-4	9.37	10.32	9.83	**10.77**
	ROUGE-1	23.87	23.82	24.05	**25.17**
	ROUGE-L	17.29	17.72	17.61	**18.43**
LA	**BLEU-1**	30.37	32.04	31.41	**32.37**
	BLEU-4	11.23	11.57	11.22	**12.01**
	ROUGE-1	22.07	23.41	23.17	**24.22**
	ROUGE-L	17.19	17.97	17.76	**18.50**
NY	**BLEU-1**	29.23	29.95	29.45	**30.67**
	BLEU-4	10.68	10.98	10.90	**11.28**
	ROUGE-1	22.77	23.67	23.10	**24.09**
	ROUGE-L	17.59	18.23	17.73	**18.35**
SF	**BLEU-1**	29.64	31.56	31.09	**32.24**
	BLEU-4	10.86	11.19	10.61	**11.60**
	ROUGE-1	21.34	22.74	22.08	**25.20**
	ROUGE-L	16.49	17.30	16.89	**18.39**

Table 8: Ablation analysis: review text generation performance evaluated @ rating level (*higher is better*)

	User + Item	✓	✓	✓	✓
	Rating		✓		✓
	Photo			✓	✓
CH	**BLEU-1**	36.64	40.42	39.63	**42.00**
	BLEU-4	10.65	12.04	11.22	**13.04**
	ROUGE-1	23.76	23.83	23.34	**24.93**
	ROUGE-L	17.10	17.52	17.06	**18.18**
LA	**BLEU-1**	37.08	39.18	37.97	**39.38**
	BLEU-4	13.01	13.45	12.85	**14.07**
	ROUGE-1	21.89	23.21	22.59	**23.74**
	ROUGE-L	17.00	17.73	17.32	**18.08**
NY	**BLEU-1**	35.87	37.04	36.19	**38.03**
	BLEU-4	12.44	12.95	12.85	**13.32**
	ROUGE-1	22.40	23.55	22.58	**23.79**
	ROUGE-L	17.32	18.09	17.38	**18.11**
SF	**BLEU-1**	38.51	40.67	39.50	**41.51**
	BLEU-4	13.19	13.64	12.63	**14.29**
	ROUGE-1	21.44	22.96	22.25	**25.54**
	ROUGE-L	16.58	17.52	17.06	**18.55**

in their objects ("steak" vs. "beef") rather than in their sentiment, signalling the utility of images in differentiating content in the text generation. The second case study in Fig. 6b examines two users with different sentiments for an item, as reflected by their ratings. Here, both photos depict clam chowder, but the generated texts are different: "good" for the more positive user, and "salty" for the more negative user. Finally, the third case study in Fig. 6c depicts how the same user may review two items. In one case, the object of interest is a toast or sandwich. In the other, it is pizza. These are examples of how rating prediction interacts with review text generation sharing some sentiment signals, and how the visual features help to align with aspect words allowing the sentiment features to align with the sentiment words.

5 CONCLUSION

We propose *MRG*, a neural model for multimodal review generation that joins a rating prediction component and a review text generation component informed by visual features. Through comprehensive experiments and ablation analysis, we establish its utility for recommendation tasks, outperforming both content-based recommender systems and review text generation baselines. We attribute the outperformance to a more holistic representation of various signals within a review, yielding more accurate recommendations.

ACKNOWLEDGMENTS

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its NRF Fellowship Programme (Award No. NRF-NRFF2016-07).

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

[2] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. 2007. Multi-task feature learning. In *Advances in neural information processing systems*. 41–48.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, 248–255.

[5] Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. 2017. Learning to generate product reviews from attributes. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Vol. 1. 623–632.

[6] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 6645–6649.

[7] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 1661–1670.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[10] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 233–240.

[11] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[12] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.

[13] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[15] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*. 556–562.

[16] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. Neural rating regression with abstractive tips generation for recommendation. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 345–354.

[17] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out* (2004).

[18] Chin-Yew Lin and Franz Josef Och. 2004. Orange: a method for evaluating automatic evaluation metrics for machine translation. In *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, 501.

[19] Guang Ling, Michael R Lyu, and Irwin King. 2014. Ratings meet reviews, a combined approach to recommend. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 105–112.

[20] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 165–172.

[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[22] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.

[23] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. 2011. Multimodal deep learning. In *Proceedings of the 28th international conference on machine learning (ICML-11)*. 689–696.

[24] Jianmo Ni, Zachary C Lipton, Sharad Vikram, and Julian McAuley. 2017. Estimating reactions and recommending products with generative models of reviews. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 783–791.

[25] Jianmo Ni and Julian McAuley. 2018. Personalized Review Generation by Expanding Phrases and Attending on Aspect-Aware Representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 706–711.

[26] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 311–318.

[27] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[28] Steffen Rendle. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 995–1000.

[29] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. 452–461.

[30] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[31] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009).

[32] Chong Wang and David M Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 448–456.

[33] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[34] Zhongqing Wang and Yue Zhang. 2017. Opinion recommendation using neural memory model. In *Proceedings of the 2017 conference on empirical methods in natural language processing (EMNLP)*. 1626–1637.

[35] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. 2048–2057.

[36] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4651–4659.

[37] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 83–92.

[38] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 425–434.

Revisiting Wedge Sampling for Triangle Counting

Ata Turk
Boston University
Boston, Massachusetts
ataturk@bu.edu

Duru Türkoğlu
Loyola University Chicago
Chicago, Illinois
dturkoglu@luc.edu

ABSTRACT

Triangle counts of massive graphs can provide important information regarding the structure of the networks these graphs model. Exact triangle counting can be expensive and thus researchers have proposed a number of approximation approaches. The state-of-the-art triangle count approximation techniques depend on wedge (two-path) sampling. In this paper we offer a mechanism to significantly improve wedge sampling for triangle counting. We shrink the sampling space by eliminating wedges that are less likely to participate in triangles. Experiments over large-scale real-world graphs show that proposed mechanism provides five- to a few hundred-folds sampling space reduction. When compared against the state-of-the-art approaches, it requires as low as ∼100× less sampling to provide the same accuracy, or makes as low as ∼8× less error when used with the same sampling ratio.

CCS CONCEPTS

• **Mathematics of computing** → **Graph algorithms**; **Approximation algorithms**; • **Information systems** → Social networks.

KEYWORDS

Triangle count estimation; Wedge sampling; Approximation algorithms

ACM Reference Format:
Ata Turk and Duru Türkoğlu. 2019. Revisiting Wedge Sampling for Triangle Counting. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313534

1 INTRODUCTION

The number of triangles in a graph is an important metric that represents the transitivity of the relations and the degree of clustering of the entities modeled by the graph. This metric has use in applications such as social network analysis [40], gene expression microarray data analysis [18], or text summarization [2].

Currently, practically used exact triangle counting and listing algorithms have a complexity of $O(m^{3/2})$ [30], where m is the number of edges. However, target graphs of interest for triangle counting can have billions of edges and hundreds of billions of triangles, which render exact algorithms costly. Furthermore, for many settings the exact triangle count is not necessary and approximations are sufficient as the graphs are dynamic. Thus, a number of approximation solutions for estimating the triangle count have been proposed [6, 7, 9, 10, 14, 16, 21, 22, 24, 28, 31–33, 36–38].

The state-of-the-art approximation approaches utilize wedge sampling. Wedge-sampling-based solutions have shown to be fast and accurate in estimating triadic measures [32, 38]. These approaches sample wedges from a graph, check whether sampled wedges are "closed" (form a triangle in the original graph) or not, and based on the ratio of closed and open wedges, estimate the triangle count in the original graph. Unfortunately, in large-scale power-law graphs there are too many wedges compared to the number of triangles. The number of wedges in such graphs can easily be in the order of trillions and the chances of finding a "closed" wedge can be as low as one in ten thousand. Estimating the number of triangles available with confidence while sampling from such a large wedge-space requires a significant number of samplings.

In this paper we revisit the wedge-sampling approaches for estimating triangle counts in large graphs. We first show that during sampling it is not necessary to consider all of the wedges in a graph and it is possible to reduce the sampling space significantly. In a preprocessing step, we first define a partial ordering of vertices, and then use that ordering to categorize wedges into three groups. We show that sampling from any one of these groups is sufficient for triangle counting. We use this finding to restrict the sampling-space for triangle counting, which leads to more accurate estimations with far fewer sampled wedges when compared to existing aproaches.

We theoretically prove that our mechanism guarantees at least a three-fold reduction in the sampling space, thus guaranteeing an accuracy no worse than standard wedge sampling, and we show via experimental analysis that we achieve one to two orders of magnitude reduction in practice for real-world graphs. Note that such reduction can pave the way for more accurate analysis and estimations over larger graphs (e.g. in some cases, the number of candidate wedges for sampling decrease so much that running exact algorithms become feasible).

Our contributions can be listed as follows:

- We show that any total order on the vertices of a graph can be used to divide the wedges in a graph into three categories, all of which can be separately and accurately used for triangle count estimation.
- We provide a greedy total ordering heuristic that divides the wedges in a graph such that the number of wedges in a pre-specified category is guaranteed to be at most one third of the total number of wedges.
- We prove that the number of wedges sampled using our greedy heuristic is a 2-approximation, i.e., it is at worst twice as much as that of an optimal ordering for the considered category.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313534

- We provide a wedge-sampling based algorithm that utilizes our categorization for improved triangle count estimation.
- We provide closed form expressions for the Relative Standard Error of our algorithm to allow for comparison against previously proposed algorithms over known graphs.
- We evaluate the triangle counting performance of the proposed algorithm against the state-of-the-art over real-world large-scale graphs. Results indicate that proposed sampling algorithm outperforms the state-of-the-art significantly, requiring between two to hundred-fold less sampling to provide the same estimation accuracy.

In the remainder of the paper, we first provide a brief background and the related work (Section 2), then explain our improved wedge sampling approach (Section 3), and then we evaluate our approach (Section 4) and conclude (Section 5).

2 RELATED WORK

Given an undirected graph $G = (V, E)$, where V is a set of n vertices, and E is a set of m edges, we would like to find out T, the number of triangles (three edge cycles) in G. The triangle counting problem can be considered a special case of the subgraph counting [20] or specific length cycle counting [3] problems.

The most efficient algorithm for exact triangle counting uses the fact that if A is the adjacency matrix of G, then the diagonal elements of A^3 contain (two times) the number of triangles of the corresponding vertex. The complexity of this algorithm is $O\left(m^{2\gamma/(\gamma+1)}\right)$ [3], where m is the number of edges in the graph and γ is the exponent of the used matrix multiplication algorithm. Considering a low known $\gamma < 2.38$ for matrix multiplication [25], this complexity can be listed as $\approx O(m^{1.407})$.

Practically efficient exact triangle counting algorithms generally have a complexity of $O(m^{3/2})$ and we refer [30] for a thorough practical comparison of efficient in-memory sequential counting and listing implementations. As most graphs of interest do not fit in memory, some triangle counting studies focus on reducing the number of disk I/Os performed [12, 19] during counting, while others focus on parallelizing the counting process [4, 11, 39].

In cases where the exact count is not needed and an "accurate enough" approximation to the actual count is sufficient, significantly faster triangle-count approximation approaches can be utilized. These approaches can be further divided into the streaming settings where graph components arrive as a data stream [1, 6, 9, 14, 16, 17, 27, 35], and the static settings where the entire graph is available for analysis [32, 33, 37, 38]. Our focus on this work is on the static scenario, where the sampling approaches randomly sample edges [10, 37] or adjacent edge pairs (wedges) [33, 38] to estimate the number of triangles in G based on the statistics obtained from the sampled graph.

State-of-the-art approximation solutions to triangle counting for static settings are based on wedge sampling [33, 38]. The wedge sampling solution proposed in [33] uniformly samples wedges and multiplies one third of the ratio of closed-wedges among the sampled wedges with the total number of wedges in the graph to approximate the total number of triangles in the graph. The solution proposed in [38] note that in graphs with power-law degree distribution, the triangles are not uniformly distributed across wedges and uniform sampling leads to bias necessitating more sampling. They suggest starting with uniform edge sampling and then converting the sampled edges to wedges via further sampling another edge from the connected edges of the first sampled edge.

In this work we also focus on wedge-sampling based triangle count approximation. Our contribution is in proving that for the problem of triangle counting the space of wedges can be significantly reduced (at least three times provably and one to two orders of magnitude in experimented real-world graphs). This reduction in turn leads to more accurate estimation with less sampling. Our space reduction approach has the additional benefit of eliminating the bias in wedges for power law-graphs. Once the space is reduced, we use the uniform wedge sampling approach described in [33] to estimate the triangle count.

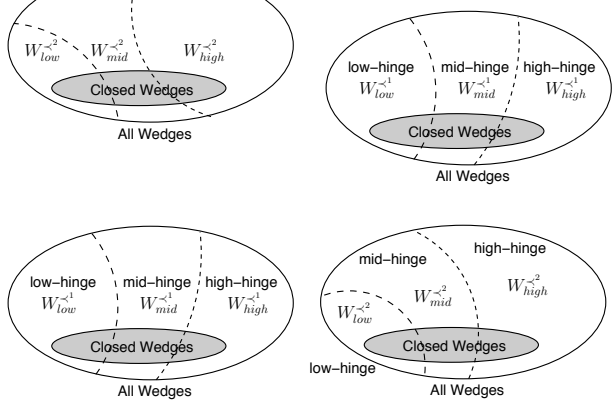
In addition to triangle counting, sampling techniques have been used for estimating counts of other important patterns in graphs such as 4-paths [15], diamonds [5], butterfly patterns in bipartite graphs [29], and cliques of any size [13]. Among these works, 4-path sampling work presented in [15] is of most interest to this work. It utilizes a similar approach to what we propose to first order vertices and use this ordering to reduce the set of 4-paths investigated as an implementation improvement. Unfortunately they only provide empirical analysis of this approach without providing detailed theoretical proofs of its search space reduction implications. In this work we provide a better ordering approach for better search space reduction than their approach, and provide theoretical proofs regarding the quality of the proposed approach. We believe this work sheds further light into why they observe significant improvements in their empirical tests when they use an ordering.

3 REVISITING WEDGE SAMPLING

In this section we first show that given a total ordering of vertices in a graph, it is possible to categorize the wedges in the graph into three groups such that sampling wedges only from one group is sufficient for accurate triangle count estimation. We then offer a simple greedy mechanism to obtain a total ordering such that the number of wedges in one of the categories is significantly smaller than the total number of wedges in the graph. We prove that via the proposed ordering, the number of wedges in the category we choose is at least three times less than the total number of wedges in the graph. We also prove that proposed ordering leads to at most two times more wedges in the target group compared to an optimal ordering for that group.

3.1 Categorizing Wedges and Using One Category for Triangle Sampling and Counting

Let $\prec$ define a total ordering on the vertices of a graph $G = (V, E)$. Using this total ordering, we can categorize wedges (two-paths) in G into three groups based on the hinge vertex of the wedge as defined below. We say that the *hinge vertex*, or *hinge* for short, of any given wedge w_{ijk} formed using the edges $\{v_i, v_j\}$ and $\{v_j, v_k\}$ is the vertex common to both edges, i.e., v_j. Then, we categorize the wedge w_{ijk} as a *high-hinge*, *mid-hinge*, or *low-hinge* wedge if the hinge vertex v_j is the highest, middle, or the lowest respectively



(a) Categorization due to $\prec^1$ (b) Categorization due to $\prec^2$

Figure 1: Orderings $\prec^1$ and $\prec^2$ divide the wedges into low-, mid-, and high-hinge categories. For both orderings, any category can be used for sampling triangles since the number of closed wedges in each category is equal to the number of triangles (closed wedges), i.e., $\overline{W}^{\prec^1}_{low}$=$\overline{W}^{\prec^1}_{mid}$=$\overline{W}^{\prec^1}_{high}$=$\overline{W}^{\prec^2}_{low}$=$\overline{W}^{\prec^2}_{mid}$=$\overline{W}^{\prec^2}_{high}$=$T$. Even though the number of triangles in each category is the same, the ratio of triangles to wedges ($\overline{W}/W$) is different for each category. Using the low-hinge category due to $\prec^2$ would be the best among these six options as that option has the highest triangle-to-wedge ratio.

when compared to the other two vertices of the wedge with respect to the total ordering $\prec$.

To estimate the triangle count of a graph G, it is sufficient to sample wedges only from any one of the wedge categories described above. For a given ordering $\prec$ of V, any triangle in G contains exactly one low-hinge, one mid-hinge, and one high-hinge wedge. To observe this, assume a triangle composed of the edges $\{v_i, v_j\}$, $\{v_i, v_k\}$, and $\{v_j, v_k\}$. Without loss of generality, let $v_i \prec v_j \prec v_k$. Clearly, the wedges w_{jik}, w_{ijk}, and w_{ikj} that have hinge vertices v_i, v_j, and v_k are low-hinge, mid-hinge, and high-hinge wedges respectively.

No matter which category of wedges one samples from, all triangles in G can be found during sampling. Thus, wedge-based sampling algorithms proposed in [33] for triangle count estimation can be applied using only one category of wedges for sampling. Intuitively, the smaller the number of wedges in the target category that will be used for sampling, the higher the improvement achieved over sampling wedges from all categories. This is due to the fact that the number of wedges in a category is equal to the number of triangles plus the number of so-called *open* wedges, which do not form triangles, in that category. This intuition is supported by our theoretical analysis as we show that the sampling error reduces as the ratio of open wedges reduces in a category.

As the number of wedges in each category depend on the ordering, we try to find an ordering that reduces the number of wedges in one category. In our approach, we focus on reducing the number of wedges in the low-hinge category. Note that we could have easily selected the high-hinge category as well since the number of low-hinge wedges in an ordering and the number of high-hinge wedges in the reverse ordering are the same.

For a given total order $\prec$, and any given vertex $v \in V$, let $d^{\prec}_{+}(v)$ be the number of vertices adjacent to v that are ordered after v, i.e., $d^{\prec}_{+}(v) = |\{u \in V \mid \{v,u\} \in E, v \prec u\}|$. Similarly, let $d^{\prec}_{-}(v)$ be the number of vertices adjacent to v that are ordered before v, i.e., $d^{\prec}_{-}(v) = |\{u \in V \mid \{v,u\} \in E, u \prec v\}|$. Clearly, the degree of v is $d(v) = d^{\prec}_{+}(v) + d^{\prec}_{-}(v)$. Given this notation, we define the number of low-hinge wedges in the graph G as $W^{\prec}_{low} = \sum_{v \in V} \binom{d^{\prec}_{+}(v)}{2}$. Similarly, we define the number of high-hinge wedges as $W^{\prec}_{high} = \sum_{v \in V} \binom{d^{\prec}_{-}(v)}{2}$, and the number of mid-hinge wedges as $W^{\prec}_{mid} = \sum_{v \in V} (d^{\prec}_{+}(v) \times d^{\prec}_{-}(v))$. Summing up, the total number of wedges is $W = \sum_{v \in V} \binom{d(v)}{2} = W^{\prec}_{low} + W^{\prec}_{mid} + W^{\prec}_{high}$. Also note that for any total ordering $\prec$, there is exactly one low-hinge, one mid-hinge, and one high-hinge wedge in any triangle. Therefore, the number of *closed* low-hinge wedges ($\overline{W}^{\prec}_{low}$), the number of *closed* mid-hinge wedges ($\overline{W}^{\prec}_{mid}$), and the number of *closed* high-hinge wedges ($\overline{W}^{\prec}_{high}$) are all equal to the number of triangles, i.e., $\overline{W}^{\prec}_{low} = \overline{W}^{\prec}_{mid} = \overline{W}^{\prec}_{high} = T$.

3.2 A Greedy Ordering Heuristic for Small Low-Hinge Wedge Count

As shown in Fig. 1 different orderings lead to different number of wedges in low-, mid-, high-hinge categories. For better estimation via smaller number of samplings, it is ideal to use an ordering that provides the smallest number of wedges in one category. However, ordering itself should not be more expensive than the sampling algorithm. In this section we describe a greedy ordering heuristic that provides a categorization where the number of wedges in the low-hinge category is at most one third of the total number of wedges. Furthermore, we prove that the number of low-hinge wedges produced by our greedy ordering is at most twice as much as the minimum number of low-hinge wedges any ordering can produce.

The intuition of our ordering mechanism stems from the observation that the earlier the order of a vertex, the higher the chances of that vertex to be the hinge of a low-hinge wedge, as a low-hinge wedge can only be formed with vertices ordered after that vertex. Thus, it stands to reason to place vertices that have low degrees early on in the ordering to reduce the number of wedges in the low-hinge category. Intuitively, ordering the vertices in G based on their vertex degrees reduces the number of wedges in the low-hinge category significantly. This is exemplified in Fig. 2, where the total number of low-hinge wedges are an order of magnitude (14× precisely) less than the total number of wedges.

In Fig. 2, we compare the cumulative distribution of wedges, closed wedges, low-hinge wedges, and closed low-hinge wedges for the real world Citeseer citation graph [23] in a log-log graph. Note that the cumulative closed low-hinge wedge figure (shown with lower triangles) saturates very fast and after the vertices of degrees up to 30 are considered, reaches very close to the total number of triangles T, which is its final value. This suggests that most triangles have a low degree vertex (e.g. a vertex with a degree smaller or equal to 30), which is a common pattern observed in real-world power-law graphs. Furthermore, the number of wedges (shown with capital W) starts to deviate from the number of low-hinge wedges significantly when we consider vertices of degree 20 and higher. This confirms our intuition explained in the above paragraph that placing vertices that have low degrees early on in the ordering reduces the number of low-hinge wedges. The number of wedges, as expected, increase quadratically as the degree increases.

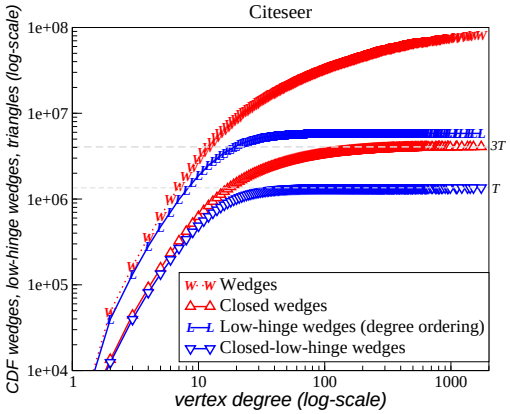


Figure 2: Cumulative distribution function for wedges, closed wedges, low-hinge wedges based on the degree ordering, and closed low-hinge wedges for the Citeseer dataset.

Note the mismatch between the wedge and triangle distributions. As the degree of vertices increase, the number of wedges increase much faster than the number of triangles. Thus a uniform sampling across wedges will be biased towards high-degree vertices. On the other hand, note the similarity between the distributions of the number of low-hinge wedges and the number of closed low-hinge wedges. A uniform low-hinge wedge sampling approach would not have the bias that the uniform wedge-sampling approach has.

The ordering based on the degrees alone can be further improved by taking into account the following observation: the only vertices that matter for a given vertex are those that appear after that given vertex. In this regard, if the vertices are ordered incrementally, placing a vertex into the total order increases the probability of its unordered neighbors becoming the hinge of low-hinge wedges, because their remaining degrees decreases by one. Taking advantage of this observation, we provide a *GreedyOrder* method to describe a total order (Algorithm 1). In *GreedyOrder*, we maintain a min heap priority queue over the degrees of vertices (lines 2-3). In a loop we remove the next smallest degree vertex from the queue (line 5) and reduce the degrees of the all the vertices incident to the removed vertex (lines 7-10). This loop continues till the queue gets empty. Vertices are ordered based on the removal order.

Complexity of *GreedyOrder* is $O(m \log n)$ as all edges are traversed only once by the algorithm and the complexity of the an update (*DecreaseKey*) is $O(\log n)$. Note that the complexity of the simple vertex degree ordering is $O(n \log n)$.

3.2.1 GreedyOrder Reduces Search Space Threefold. Here we prove that for a graph ordered by *GreedyOrder* the number of low-hinge wedges is at most one third of the total number of wedges.

Theorem 1. *Given a graph G and any ordering $\prec_{gr}$ obtained via GreedyOrder(G), the number of low-hinge wedges is at most one third of the total number of wedges, i.e., $W_{low}^{\prec_{gr}} \leq \frac{1}{3}W$.*

Proof. To count W, we will count the total number of wedges in which a given vertex v is the lowest order vertex, and then we will sum these counts for each vertex. Thus, every wedge will be counted exactly once and this sum will be equal to W. When counting, we will also rely on the following invariant that *GreedyOrder* guarantees: any vertex u appearing after another vertex v in the order $\prec_{gr}$ has a remaining degree of at least $d_+^{\prec_{gr}}(v)$ when v is removed from the priority queue. Note that, $d_+^{\prec_{gr}}(v)$ is the degree of v when v is removed from the priority queue in *GreedyOrder*(G).

Let us consider a vertex v when it is removed from the priority queue, i.e., when the order of v is determined. Also, let us consider another vertex u among the $d_+^{\prec_{gr}}(v)$ adjacent vertices of v still in the queue. Using the invariant above, the remaining degree of u is at least $d_+^{\prec_{gr}}(v)$ at the time v is removed from the priority queue. Thus, there are at least $d_+^{\prec_{gr}}(v) - 1$ wedges for which u is the hinge vertex and v is the lowest order vertex. Since there are $d_+^{\prec_{gr}}(v)$ vertices in the priority queue that are adjacent to v, the total number of mid-hinge and high-hinge wedges whose lowest order vertex is v is at least $d_+^{\prec_{gr}}(v) \times (d_+^{\prec_{gr}}(v) - 1)$. Also, recall that $W_{low}^{\prec_{gr}} = \sum_{v \in V} \binom{d_+^{\prec_{gr}}(v)}{2}$. Therefore, we have $W \geq \sum_{v \in V} \binom{d_+^{\prec_{gr}}(v)}{2} + d_+^{\prec_{gr}}(v)(d_+^{\prec_{gr}}(v)-1)$, which implies $W \geq 3W_{low}^{\prec_{gr}}$ proving the theorem. □

Algorithm 1 *GreedyOrder*$(G = (V, E))$

```
1:  L ← ∅                                   ▷ Empty list that will contain the ordering
2:  D ← ComputeVertexDegrees(V, E)          ▷ O(m)
3:  Q ← BuildMinHeap(D)
4:  while Q ≠ ∅ do                          ▷ O(m log n)
5:      x ← VertexID(Q.ExtractMin())
6:      L.Append(v_x)
7:      for all v_y ∈ Adjacency(v_x) do
8:          d_y ← d_y − 1                   ▷ reduce degree of v_x's neighbors
9:          Adjacency(v_y) ← Adjacency(v_y) − {v_x}
10:         Q.DecreaseKey(v_y, d_y)
11: return L                                ▷ return ordered vertex list
```

We note that a three-fold space reduction is very important as in normal wedge sampling the number of closed wedges is three times the number of triangles, i.e. 3T, whereas when sampling low-hinge wedges, the number of closed low-hinge wedges is equal to the number of traingles, i.e. T. If $W < 3W_{low}^{\prec_{gr}}$ was possible, then this would have meant that for some graphs it is more likely to find closed wedges when sampling from W compared to sampling from $W_{low}^{\prec_{gr}}$. With the above proof, we show that sampling from $W_{low}^{\prec_{gr}}$ is always as good as or better than sampling from W. We also note that for certain families of graphs, such as complete graphs, the above theorem is tight, i.e., $W_{low}^{\prec_{gr}} = \frac{1}{3}W$, the number of wedges in the low-hinge category is exactly 1/3 of the total number wedges in the graph.

3.2.2 GreedyOrder is 2−Approximate. Even though Theorem 1 proves that the improvement obtained using Alg. 1 to reduce $W_{low}^{\prec_{gr}}$ is at least threefold, as we will discuss in the evaluation of our algorithms, in practice we observe one to two order of magnitude reduction in the wedge space when we apply this algorithm on various graphs.

Regardless of the reduction ratio, we also prove that our algorithm produces a total ordering of the vertices that is provably very close to the best ordering that minimizes the number of low-hinge wedges. More precisely, we prove that if the best ordering $\prec_{opt}$ has $W_{low}^{\prec_{opt}}$ low-hinge wedges, then the low-hinge wedge count, $W_{low}^{\prec_{gr}}$, of the ordering $\prec_{gr}$ generated by Alg. 1 is at most twice that of the optimal ordering.

To prove our approximation bound we first show that any ordering produced by *GreedyOrder* will satisfy another important invariant. To define this invariant we first expand the ordering notation $\prec$ for a set of vertices as follows. We say that the vertex set X is ordered before vertex set Y if all vertices in X are ordered before all vertices in Y. That is, $X \prec Y$ if $\forall x \in X$ and $\forall y \in Y$, we have $x \prec y$. Using this definition, we show that there exists a paritition of the vertex set V independent of the ordering $\prec_{gr}$ that *GreedyOrder* generates and yet the sets of the partition are ordered with respect to $\prec_{gr}$:

Lemma 1. *Let $V_0, V_1, \ldots, V_k$ be the partition of the set of vertices V where V_i is the smallest subset that ensures that the minimum modified degree in the subgraph obtained by the removal of the vertices $\cup_{j=0}^{i} V_j$ is greater than i. Then, for any total ordering $\prec_{gr}$ generated by GreedyOrder, we have $V_0 \prec_{gr} V_1 \ldots \prec_{gr} V_k$.*

Proof. We prove the following statement on ℓ by induction: Letting $U_\ell = \cup_{j=0}^{\ell} V_j$, for any two vertices $v \in U_\ell$ and $v' \in V \setminus U_\ell$, we have $v \prec v'$ in the ordering provided by *GreedyOrder*. Proving this statement for all $\ell = 0, \ldots, k$ proves our lemma statement.

For the case of $\ell = 0$, the statement holds trivially since all vertices in V_0 have degree 0 and will be processed the earliest by *GreedyOrder* before any other vertex with positive degree.

Assuming the statement true for ℓ, we prove the statement for $\ell + 1$. Using the statement for ℓ, for any two vertices $v \in U_\ell$ and $v' \in V \setminus U_{\ell+1} \subset V \setminus U_\ell$, we have $v \prec v'$ in our ordering. Therefore, all we need to prove is that for any two vertices $v \in V_{\ell+1}$ and $v' \in V \setminus U_{\ell+1}$, we have $v \prec v'$ in our ordering. Assume towards a contradiction that we have $v' \prec v$ instead. Then, consider the earliest vertex $y \in V \setminus U_{\ell+1}$ with respect to the ordering $\prec$. From above, we have $y \prec v$. Furthermore, by definition of V_i, we know that $d_+^{\prec_{gr}}(y)$, the modified degree of y, must be at least $\ell + 1$ when y is ordered by *GreedyOrder*. However, this means that every vertex u satisfying $y \prec u$ must have a remaining degree of at least $\ell + 1$ at the time of ordering y. This contradicts the definition of V_ℓ in that it was the smallest such set. This is because there exists a proper subset of V_ℓ whose removal still guarantees that all other vertices have degree larger than ℓ. The contradiction proves the statement and therefore the lemma. □

Using the partitioning from the above lemma, we can now show that the number of low-hinge wedges of any total order $\prec_{gr}$ generated by *GreedyOrder* is at most twice that of any other ordering, and in particular at most twice that of the optimal ordering.

Theorem 2. *$W_{low}^{\prec_{gr}} \leq 2W_{low}^{\prec}$ for any total ordering $\prec_{gr}$ generated by GreedyOrder, and any total ordering $\prec$.*

Proof. Let $V_0, V_1, \ldots, V_k$ be the partitioning of the set of vertices V as defined in Lemma 1. Then, the statement of the lemma can be written as:

$$2W_{low}^{\prec} - W_{low}^{\prec_{gr}} = \sum_{i=0}^{k} \sum_{v \in V_i} 2\binom{d_+^{\prec}(v)}{2} - \binom{d_+^{\prec_{gr}}(v)}{2} \geq 0 \quad (1)$$

To prove this, for $i = 0, \ldots, k$, let $n_i = |V_i|$ and let $\mu_i(\prec)$ be the average low-degree of the vertices in V_i in the ordering $\prec$; i.e., $\mu_i(\prec) = \sum_{v \in V_i} d_+^{\prec}(v)/n_i$. Using the inequality between root-mean square and arithmetic mean, we get:

$$\begin{aligned}
\sum_{v \in V_i} 2\binom{d_+^{\prec}(v)}{2} &= \sum_{v \in V_i} (d_+^{\prec}(v))^2 - \sum_{v \in V_i} d_+^{\prec}(v) \\
&\geq n_i \left(\sum_{v \in V_i} d_+^{\prec}(v)/n_i \right)^2 - n_i \mu_i(\prec) \\
\sum_{v \in V_i} 2\binom{d_+^{\prec}(v)}{2} &\geq n_i \mu_i(\prec)(\mu_i(\prec) - 1) \quad (2)
\end{aligned}$$

Note that $\mu_i(\prec_{gr}) = \sum_{v \in V_i} d_+^{\prec_{gr}}(v)/n_i$. Also note that in (1), the term $\sum_{v \in V_i} \binom{d_+^{\prec_{gr}}(v)}{2}$ is maximized if we let $d_+^{\prec_{gr}}(v)$ to take the extreme values, the highest value can be i and for simplicity of the proof, we will assume the lowest value to be 1 (a longer proof follows if we allow 0 low-degree vertices). Since the average of $d_+^{\prec_{gr}}(v)$ is $\mu_i(\prec_{gr})$, we can assume a ratio of $i - \mu_i(\prec_{gr})$ vertices with $d_+^{\prec_{gr}}(v) = 1$ and $\mu_i(\prec_{gr}) - 1$ vertices with $d_+^{\prec_{gr}}(v) = i$ in the worst case. Then, we can state:

$$\begin{aligned}
\sum_{v \in V_i} \binom{d_+^{\prec_{gr}}(v)}{2} &\leq \frac{n_i}{i-1}(i - \mu_i(\prec_{gr}))\binom{1}{2} \\
&\quad + \frac{n_i}{i-1}(\mu_i(\prec_{gr}) - 1)\binom{i}{2} \\
\sum_{v \in V_i} \binom{d_+^{\prec_{gr}}(v)}{2} &\leq n_i(\mu_i(\prec_{gr}) - 1)\frac{i}{2} \quad (3)
\end{aligned}$$

Substituting in (1) the bounds we obtain in (2) and (3), we get $2W_{low}^{\prec} - W_{low}^{\prec_{gr}} \geq \sigma = \sum_{i=1}^{k} \sigma_i$, where σ_i is defined below, and we note that it suffices to show $\sigma \geq 0$ to prove the theorem:

$$\sigma_i = n_i \left(\mu_i(\prec)(\mu_i(\prec) - 1) - (\mu_i(\prec_{gr}) - 1)\frac{i}{2} \right)$$

Letting $\mu_i(\prec) = \mu_i(\prec_{gr}) + \phi_i/n_i$, we rewrite σ_i to obtain:

$$\sigma_i = \phi_i(\mu_i(\prec) + \mu_i(\prec_{gr}) - 1) + n_i(\mu_i(\prec_{gr}) - 1)\left(\mu_i(\prec_{gr}) - \frac{i}{2}\right) \quad (4)$$

To prove that $\sigma \geq 0$, we make the following observations:
(i) $\mu_i(\prec_{gr}) \geq \frac{i}{2}$,
(ii) for any $\ell = 1, \ldots, k$, $\sum_{i=1}^{\ell} \phi_i \leq 0$, and
(iii) $\sum_{i=1}^{k} \phi_i = 0$.

The first observation can be shown by the fact that every vertex $v \in V_i$ has degree at least i in the graph restricted to the vertices $\cup_{j=i}^{k} V_j$, which shows that there are at least $\frac{in_i}{2}$ edges incident to the vertices in V_i, and hence $\mu_i(\prec_{gr}) \geq \frac{i}{2}$. For the second observation, the definition of V_1 maximizes $\sum_{v \in V_1} d_+^{\prec_{gr}}(v)$, and in general, the definition of V_i maximizes $\sum_{v \in \cup_{j=1}^{i} V_j} d_+^{\prec_{gr}}(v)$. Therefore, for

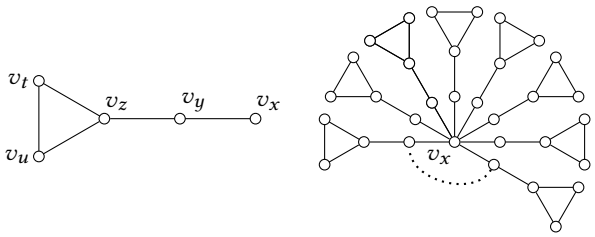


Figure 3: Illustration of proof of Lemma 2.

any ordering $\prec$, and for any $\ell = 1, \ldots, k$, we obtain $\sum_{i=1}^{\ell} \phi_i \leq 0$. Finally, the third observation is correct because each edge in the entire graph is counted exactly once, hence the sum of the modified degrees must be the same regardless of the ordering $\prec$.

First, we simplify the terms σ_j for those indices j with non-negative ϕ_j values. Using the first observation that $\mu_j(\prec_{gr}) \geq \frac{j}{2}$, the property that $\mu_j(\prec) \geq \mu_j(\prec_{gr})$ (directly implied by $\phi_j \geq 0$), and using (4), we can bound σ_j by:

$$\sigma_j \geq \phi_j(j-1)$$

Using the second and the third observations, we know that each reduction in an earlier vertex set V_i must be corrected as an increment in a later vertex set V_j for some $j > i$. Then, we prove $\sigma \geq 0$ by showing that each of the terms σ_i for those indices i with negative ϕ_i values satisfies $\sigma_i \geq i\phi_i$, or equivalently $\hat{\sigma}_i = \sigma_i - i\phi_i \geq 0$. Using the equation (4), we simplify $\hat{\sigma}_i$ to obtain the desired result:

$$\begin{aligned}\hat{\sigma}_i =& (\mu_i(\prec) - i + \mu_i(\prec_{gr}) - 1)\phi_i \\ &+ n_i(\mu_i(\prec_{gr}) - 1)\left(\mu_i(\prec_{gr}) - \frac{i}{2}\right) \\ =& -\frac{i}{2}\phi_i + n_i(\mu_i(\prec) - 1)\left(\mu_i(\prec) - \frac{i}{2}\right)\end{aligned}$$

If $\mu_i(\prec) \geq \frac{i}{2}$, then the above immediately implies $\hat{\sigma}_i \geq 0$ proving the theorem. Otherwise, we have $\mu_i(\prec) < \frac{i}{2}$. Furthermore, since $\mu_i(\prec_{gr}) \geq \frac{i}{2}$, we have $\mu_i(\prec) - \frac{i}{2} \geq \frac{\phi_i}{n_i}$. Still, we can show

$$\hat{\sigma}_i \geq \phi_i \left(\mu_i(\prec) - \frac{i}{2} - 1\right) \geq 0$$

This shows that $\hat{\sigma}_i \geq 0$ in either case, proving the theorem. □

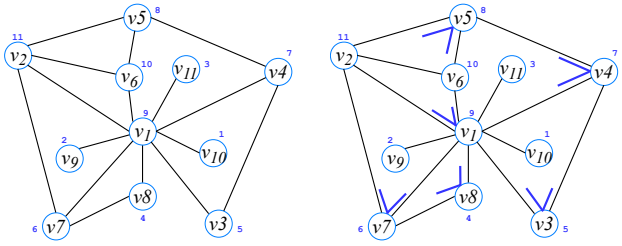
We also show that the 2-approximation of *GreedyOrder* is tight, that is, we can construct examples of graphs and orderings $\prec$ of its vertex set for which $W_{low}^{\prec_{gr}} / W_{low}^{\prec}$ gets arbitrarily close to 2.

Lemma 2. *For any given $\varepsilon > 0$, there exists a graph and an ordering $\prec$ of its vertices where $W_{low}^{\prec_{gr}} > (2-\varepsilon) W_{low}^{\prec}$.*

Proof. Consider a gadget of five vertices as shown in Figure 3. The gadget includes a triangle v_u, v_t, v_z and a two path v_z–v_y–v_x. Also, assume that the degree of v_x is high enough that our ordering algorithm chooses first the vertex v_y followed by v_z, v_t, v_u. For these four vertices our low wedge count is increased by two. However, the ordering $\prec$ where the vertices of this gadget are ordered v_u, v_t, v_z, and v_y would have resulted in a wedge count of one. To magnify the impact of such savings and prove Lemma 2, we consider the graph with at least $c > 1/\varepsilon$ copies of this gadget such that the vertex v_x is shared among the gadgets as shown in Figure 3. In this example graph, *GreedyOrder* can produce two low-hinge wedges for all gadgets but the last one, for which it will produce a single wedge. Therefore, we get $W_{low}^{\prec_{gr}} = 2c - 1$ in the worst case. Clearly the ordering $\prec$ would produce one wedge per gadget resulting with $W_{low}^{\prec} = c$, proving $W_{low}^{\prec_{gr}} > (2-\varepsilon) W_{low}^{\prec}$ as desired. □

Algorithm 2 *LowHingeWedgeSample*$(G = (V, E), \prec_{gr}, k)$

1: $S \leftarrow 0$
2: **for** $v \in V$ **do**
3: $\quad p_v(\prec_{gr}) \leftarrow \binom{d_+^{\prec_{gr}}(v)}{2} / W_{low}^{\prec_{gr}}$
4: **for** $i = 1, \ldots, k$ **do**
5: $\quad$ Select v_i from V w.r.t. probability distribution $\left\{p_{v_i}(\prec_{gr})\right\}$
6: $\quad H_{v_i} \leftarrow \{\, x \mid x \in adj(v_i) \wedge v_i \prec_{gr} x \,\}$
7: $\quad$ Select u, w uniformly at random from H_{v_i}
8: $\quad$ **if** $(u, w) \in E$ **then**
9: $\quad\quad S \leftarrow S + 1$
10: **return** $S/k \times W_{low}^{\prec_{gr}}$ ▷ return prediction



(a) Example graph with 11 vertices, 16 edges, 56 wedges, and 5 triangles. Vertex ordering (shown with blue indexes) is performed by *GreedyOrder*.

(b) Low-hinge wedges based on the ordering in (a) are marked with blue wedges. There are 6 low-hinge wedges 5 of which are closed.

Figure 4: Illustration of low-hinge wedges on an example graph.

3.3 Triangle Count Estimation via Sampling Low-Hinge Wedges

In this section we describe how we estimate the number of triangles in a graph by taking advantage of the properties of the ordering $\prec_{gr}$. Proposed approach is presented in Alg. 2. As seen in the algorithm, we first assign every vertex v a probability $p(v)$ equal to $\binom{d_+^{\prec_{gr}}(v)}{2} / W_{low}^{\prec_{gr}}$ (lines 2–3). Then, we sample k random vertices from the vertex set V with repetitions according to the assigned probability distribution (line 5). For each sampled vertex v_i, we sample a wedge out of the $\binom{d_+^{\prec_{gr}}(v_i)}{2}$ wedges of v_i (lines 6-7). If the sampled wedge is closed, i.e., the third edge is in the original graph, then we increment a sum S by one (lines 8-9). Finally, we output our triangle count prediction by multiplying S/k (the ratio of discovered closed wedges to the sampled wedges) with $W_{low}^{\prec_{gr}}$, the number of low-hinge wedges (line 10).

Figure 4(a) shows a sample graph with 11 vertices, 16 edges, 56 wedges, and 5 triangles[1]. A vertex ordering using a run of the *GreedyOrder* is indicated with a blue index number placed next to each vertex. Figure 4(b) indicates the low-hinge wedges of G based on the ordering in Figure 4(a). Out of the 6 low-hinge wedges, 5 of them are closed. The wedge space reduction is ~ 9× (from 56 to 6). As an example, if we run *LowHingeWedgeSample* on the sample graph with $k = 3$, we'll find 2 or 3 closed wedges, which will lead to a prediction of 2/3*6=4 or 3/3*6=6 triangles.

3.4 Expected Value, Variance, and RSE of Low-Hinge Wedge Sampling

In this section we analyze and provide closed form expressions for the expected value, the variance, and the relative standard error of our algorithm to be able to reason about its performance on graphs whose properties are known.

Considering every wedge selection step using $i = 1$ to k, and numbering every triangle in the graph using $j = 1$ to T, we introduce the following indicator variables for our analysis: let w_{ij} be the indicator variable whose value is 1 if and only if the low-hinge wedge for the j^{th} triangle is chosen at the i^{th} wedge selection step; and 0 otherwise[2]. Using these indicator variables, the output of our algorithm, S, can be expressed as:

$$S = \sum_{i=1}^{k} \sum_{j=1}^{T} w_{ij}.$$

We express the output of our algorithm in terms of the ratio of triangles T in the graph to the number of considerable wedges $W_{low}^{\prec gr}$, i.e., $R^{gr} = T/W_{low}^{\prec gr}$. Note that intuitively, as the ratio R^{gr} gets larger, the accuracy of our algorithm should increase and the number of samples we need to pick to achieve a given error rate in estimation should decrease. We can express $\mathbb{E}(S)$, the expected value of S, and $var(S)$, the variance of S, in terms of R^{gr} and in terms of k, the number of samples:

Lemma 3. *The expected value of S is kR^{gr}.*

Proof. We know that $\mathbb{E}(w_{ij}) = 1/W_{low}^{\prec gr}$, therefore, we get

$$\mathbb{E}(S) = \sum_{i=1}^{k} \sum_{j=1}^{T} \mathbb{E}(w_{ij}) = \sum_{i=1}^{k} \sum_{j=1}^{T} 1/W_{low}^{\prec gr} = kT/W_{low}^{\prec gr} = kR^{gr}$$

as desired, proving the lemma. □

Lemma 4. *The variance of S is $kR^{gr}(1 - R^{gr})$.*

Proof. For brevity, in the following analysis we will use the notation Λ to indicate $W_{low}^{\prec gr}$. We have $var(w_{ij}) = \frac{1}{\Lambda} - \frac{1}{\Lambda^2}$. Also, for any $i \neq i'$, we have $covar(w_{ij}, w_{i'j'}) = 0$ since the two selection processes at the two steps i and i' are independent of each other. Finally, for any $j \neq j'$, we have $covar(w_{ij}, w_{ij'}) = 0 - \frac{1}{\Lambda^2}$ since only one wedge is selected at each selection step. Therefore, we have:

$$\begin{aligned}
var(S) &= \sum_{i=1}^{k} \sum_{j=1}^{T} var(w_{ij}) + \sum_{i=1}^{k} \sum_{j \neq j'} covar(w_{ij}, w_{ij'}) \\
&= \sum_{i=1}^{k} \sum_{j=1}^{T} \left(\frac{1}{\Lambda} - \frac{1}{\Lambda^2}\right) + \sum_{i=1}^{k} \sum_{j \neq j'} \left(-\frac{1}{\Lambda^2}\right) \\
&= kT\frac{1}{\Lambda} - kT\frac{1}{\Lambda^2} - kT(T-1)\frac{1}{\Lambda^2} \\
&= k\frac{T}{\Lambda}\left(1 - \frac{1}{\Lambda} - \frac{T-1}{\Lambda}\right) \\
&= kR^{gr}\left(1 - \frac{T}{\Lambda}\right) \\
&= kR^{gr}\,(1 - R^{gr})
\end{aligned}$$

proving the variance stated in the lemma. □

We compare our method to previous work using the Relative Standard Error (RSE) values of all of the methods and base our arguments on the RSE values. Applying the formula for the relative standard error $RSE(S) = \sqrt{var(S)}/\mathbb{E}(S)$, we obtain:

$$RSE(S) = \frac{\sqrt{kR^{gr}(1 - R^{gr})}}{kR^{gr}} = \sqrt{\frac{1 - R^{gr}}{kR^{gr}}} \tag{5}$$

4 EVALUATION

In this section we provide an experimental analysis of the proposed methodology.

4.1 Experimental Setup

We compare the performance of our low-hinge-based wedge sampling algorithm (LWS) with the state-of-the-art wedge sampling [33] (WS) and edge-based wedge sampling [38] (EWS) approaches using large-scale graphs modeling real-world networks. As a comparison metric (and to verify our theoretical analysis presented in Section 3.4), we observe the relative standard error (RSE) values of different algorithms. We also compare these values with the theoretical analysis we provided in Section 3.4. In the figures, in addition to the RSE estimation we provide for LHS, we draw the estimations for WS and EWS using the formulations for the expected RSE values of these algorithms presented in [38]. It is important to note that the formulation for WS (shown below) has a striking similarity to the formulation in Eq. 5, allowing us to compare WS and LHS using the clustering coefficients (C) and the R values directly:

$$RSE(WS) = \sqrt{\frac{1 - C}{kC}} \tag{6}$$

In order to provide comparable results, as was done in [10, 38], we report the corresponding RSE values of the three algorithms for a given sampling probability. We vary the sampling probability until one of the algorithms achieves RSE ≤ 0.01. Considering the randomized nature of the sampling based approximation algorithms, we calculate the experimental RSE values observed with k different runs of the algorithms using the formula:

$$RSE = \frac{\sqrt{\frac{1}{k}\sum_{i=1}^{k}(\Delta_i - \mu)}}{T},$$

[1] We used the same sample graph presented in [38].

[2] We would like to thank the anonymous reviewer who pointed out the error in our initial indicator variable defition.

Table 1: Datasets used in experiments and their features.

Dataset	n	m	Description
Ego-Facebook [26]	4K	88K	Online social netw. Facebook
Enron-email [23]	36K	183K	Email comm. netw. in Enron
Brightkite [26]	58K	214K	Online social netw. Brightkite
Dblp-coauthor [26]	317K	1049K	Co-authorship netw. in DBLP
Amazon [26]	334K	925K	Co-purchasing from Amazon
Web-NotreDame [23]	325K	1090K	Web graph of Notre Dame
Citeseer [23]	384K	1736K	Citation in Citeseer
Dogster [23]	426K	8543K	OSN from dogster.com website
Web-Google [23]	875K	4322K	Web graph from Google
YouTube [26]	1134K	2987K	Online social netw. in Youtube
DBLP [23]	1314K	5362K	Co-authorship netw. in DBLP
As-skitter [26]	1696K	11095K	Internet connections Skitter
Flicker [23]	2302K	22838K	Online social netw. in Flicker
Orkut [23]	3072K	117185K	Online social netw. in Orkut
LiveJournal [26]	3997K	34681K	OSN in LiveJournal
Orkut2 [8]	11514K	327036K	Online social netw. in Orkut
Web-Arabic [8]	22743K	553903K	Web graph Arabian countries
MicrosoftAG [34]	46742K	528463K	Citation in Microsoft Academic
Twitter [23]	41652K	1202513K	Online social netw. in Twitter
Friendster [23]	65608K	1806067K	OSN from website Friendster

where Δ_i is the estimate obtained in the i-th run and μ is the mean of all of the k runs, i.e., $\mu = \frac{1}{k}\sum_{i=1}^{k}\Delta_i$. In all of our experiments, we use $k = 1000$.

We conducted the experiments on a server with 16 CPUs and 64 GB of RAM. The implementation was done on C++. We experimented over 20 real-world datasets with varying sizes that were made available in [10]. Largest of the datasets contain more than 40 million vertices and 30 billion triangles.

Features of the datasets (the number of vertices (n), edges (m), and description) used in the experiments are listed in Table 1. Other interesting features of the datasets such as the number of triangles (T), the number of wedges (W), and the clustering coefficients (C), as well as the number of low-hinge wedges ($W_{low}^{\prec_{gr}}$) and the R values for the greedy (R^{gr}) and degree ordering (R^{dg}) schemes, which can be used to estimate the RSE values using Eq. 5 are presented in Table 2.

4.2 Empirical Analysis

As seen in Table 2, the number of low-hinge wedges $W_{low}^{\prec_{gr}}$ are in general one to two orders of magnitude lower than the total number of wedges W for all graphs, indicating a significantly reduced sampling space for LHS. Also we can see in the table that for all datasets both R^{gr} and R^{dg} are significantly larger than C indicating that our approach has a much higher chance of hitting a triangle than in WS. Furthermore, for all datasets, R^{gr} is larger than R^{dg} showing that *GreedyOrder* reduces the sampling space more than a degree ordering solution would do. We observe from Equations (5) and (6) that the higher (and closer to 1) the R or C value is, the lower the RSE is expected to be. Given that R^{gr} is larger than C and R^{dg} for all datasets, LHS using the greedy ordering is expected to lead to lower RSE compared to other approaches for all datasets.

More strikingly, in Table 2 we observe that the value of R^{gr} is very close to one for many datasets (e.g. Web-NotreDame, Web-Arabic), which indicate that our approach shrank the sampling space so much that the number of low-hinge wedges is almost equal to the number of triangles and almost all samplings lead to closed wedges. Simply, by sampling very few low-hinge wedges (e.g., approximately 34 low-hinge wedges to get an RSE of 0.05 for Web-NotreDame or for Web-Arabic) and by mutliplying the number of closed wedge ratio with the number of low-hinge wedges, we can accurately estimate the total number of triangles (e.g., Web-NotreDame and Web-Arabic has more than 8 million and 36 billion triangles respectively). Note here that, irrespective of the utilized approach, to offer a correct sampling, the sampling space cannot be reduced to a size less than the total number of triangles, which shows how effective our reduction is.

Figure 5, provides an empirical comparison of the relative standard errors of WS, EWS and LHS over the 20 datasets listed in Table 1. The figure also depicts the RSE estimations indicated by [38] for WS and EWS, and by Equation 5 for LHS. In the figure the x-axis indicates different sampling ratios experimented with (in terms of number of sampled wedges) and the y-axis indicates the corresponding RSE values obtained by the three algorithms. Both the x- and y-axis is presented in log-scale and the x-axis range is selected such that the RSE values for LHS will range between 0.20 and 0.01.

First we note that for all datasets depicted in Figure 5, the empirical RSE observations we make for LHS perfectly match the theoretical bounds we compute using Equation 5. This proves the validity of our theoretical analysis. We also observe that, as expected, the datapoints in Figure 5 for WS, EWS and LHS all exhibit parallel trends in the log-log graph irrespective of the sampling ratio. In other words, all lines in the figure have the same slope with different constants defining their difference. This is expected as WS and LHS practically run the same algorithm albeit on different sampling spaces and all of WS, EWS, and LHS depend linearly on the sampling ratio, therefore, the ratio of their RSE values remain the same across different sampling probabilities.

In all of the datasets LHS performs significantly better than the other algorithms and offers highly confidant estimations with very small number of samplings. We note that since Figure 5 is a log-log figure, the constant difference between the lines in fact indicate a multiplicative difference. i.e. for all datasets, LHS offers two to eight times lower RSE at the same sampling ratio and this multiplicative distance remains fixed per dataset at all sampling ratios.

Note that the parallel nature of the figures will continue even if more samplings are done and the RSE gets lower. For any sampling ratio LHS will provide more accurate results, and for any desired RSE value LHS will be able to achieve that value with lower number of samplings. For example, in the As-skitter dataset, to achieve an RSE of around 0.11, LHS needs to sample just 165 wedges whereas EWS needs to sample 9476 wedges (a 57× difference). This difference is more profound in Web-Arabic where LHS needs 113× less sampling. LHS offers the highest RSE improvement ($\sim$ 8×) at the same sampling ratio over the best of WS and EWS at the Web-Arabic and Web-NotreDame datasets. This is expected as on these datasets, LHS achieves close to one R^{gr}, which indicate that LHS performs close to optimal wedge-space reduction.

5 CONCLUSION

We proposed a methodology to significantly improve the performance of wedge-sampling based triangle counting approaches by reducing the size of sampling space. We achieved this by ordering

Table 2: Number of low-hinge wedges and R values with greedy and degree orderings.

Dataset	T	W	$W_{low}^{\prec gr}$	$W/W_{low}^{\prec gr}$	$C=\frac{3T}{W}$	$R^{gr}=T/W_{low}^{\prec gr}$	$R^{dg}=T/W_{low}^{\prec dg}$
Ego-Facebook [26]	1.6M	9.3M	1.9M	4.9	0.5192	0.8552	0.8385
Enron-email [23]	0.7M	25.6M	1.4M	18.3	0.0853	0.5264	0.4982
Brightkite [26]	0.5M	13.4M	0.9M	14.9	0.1106	0.5195	0.4862
Dblp-coauthor [26]	2.2M	21.8M	2.7M	8.1	0.3064	0.8145	0.7830
Amazon [26]	0.7M	9.8M	1.0M	9.8	0.2052	0.6414	0.5859
Web-NotreDame [23]	8.9M	304.9M	9.7M	31.4	0.0877	0.9220	0.9114
Citeseer [23]	1.4M	81.7M	5.1M	16.0	0.0496	0.2665	0.2342
Dogster [23]	83.5M	17,554.4M	266.4M	65.9	0.0143	0.3134	0.2923
Web-Google [23]	13.4M	727.4M	15.9M	45.7	0.0552	0.8427	0.8085
YouTube [26]	3.1M	1,474.5M	13.9M	106.1	0.0062	0.2194	0.1947
DBLP [23]	12.2M	214.7M	18.7M	11.5	0.1703	0.6503	0.6231
As-skitter [26]	28.8M	16,021.7M	87.1M	184.0	0.0054	0.3302	0.3044
Flicker [23]	837.6M	23,342.9M	2,030.7M	11.5	0.1076	0.4125	0.3978
Orkut [23]	627.6M	45,625.5M	3,787.8M	12.0	0.0413	0.1657	0.1553
LiveJournal [26]	177.8M	4,255.8M	422.6M	10.1	0.1253	0.4208	0.3954
Orkut2 [8]	223.1M	2,543,767.1M	35,698.9M	71.3	0.0003	0.0062	0.0054
Web-Arabic [8]	36,895.4M	3,531,929.7M	40,075.6M	88.1	0.0313	0.9206	0.9135
MicrosoftAG [34]	578.2M	115,067.1M	5,724.7M	20.1	0.0151	0.1010	0.0919
Twitter [23]	34,824.9M	123,435,589.6M	136,394.5M	905.0	0.0008	0.2553	0.2314
Friendster [23]	4,173.7M	720,655.0M	76,910.8M	9.4	0.0174	0.0543	0.0507

the vertices of the graph, using this ordering to categorize wedges, only using low-hinge wedges for sampling. If degrees of vertices are available, we showed that a simple degree ordering can significantly reduce the size of sampling space. We proved that the reduction achieved by a slightly more complex ordering mechanism is at least three-fold in theory, is a 2-approximation to the best achievable reduction for low-hinge wedges, and provides one to two orders of reduction in practice. Through rigorous analysis over large-scale real-world graphs, we show that our approach offers highly confident estimations always outperforming the state-of-the-art approaches two to eight times in terms of accuracy or up to 100× in terms of sampling size for the same level of accuracy.

REFERENCES

[1] Nesreen K Ahmed, Nick Duffield, Theodore L Willke, and Ryan A Rossi. 2017. On sampling from massive graph streams. *Proceedings of the VLDB Endowment* 10, 11 (2017), 1430–1441.

[2] Yazan Alaya AL-Khassawneh, Naomie Salim, and Obasa Adekunle Isiaka. 2014. Extractive Text Summarisation using Graph Triangle Counting Approach: Proposed Method. In *1 st International Conference of Recent Trends in Information and Communication Technologies in Universiti Teknologi Malaysia, Johor, Malaysia*. 300–311.

[3] N. Alon, R. Yuster, and U. Zwick. 1997. Finding and counting given length cycles. *Algorithmica* 17, 3 (01 Mar 1997), 209–223.

[4] A. Azad, A. BuluÃğ, and J. Gilbert. 2015. Parallel Triangle Counting and Enumeration Using Matrix Algebra. In *2015 IEEE International Parallel and Distributed Processing Symposium Workshop*. 804–811.

[5] Grey Ballard, Tamara G Kolda, Ali Pinar, and C Seshadhri. 2015. Diamond sampling for approximate maximum all-pairs dot-product (MAD) search. In *Data Mining (ICDM), 2015 IEEE International Conference on*. IEEE, 11–20.

[6] Ziv Bar-Yossef, Ravi Kumar, and D. Sivakumar. 2002. Reductions in Streaming Algorithms, with an Application to Counting Triangles in Graphs. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '02)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 623–632. http://dl.acm.org/citation.cfm?id=545381.545464

[7] Luca Becchetti, Paolo Boldi, Carlos Castillo, and Aristides Gionis. 2008. Efficient Semi-streaming Algorithms for Local Triangle Counting in Massive Graphs. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*. ACM, New York, NY, USA, 16–24. https://doi.org/10.1145/1401890.1401898

[8] Paolo Boldi, Marco Rosa, Massimo Santini, and Sebastiano Vigna. 2011. Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks. In *Proceedings of the 20th international conference on World wide web*. ACM, 587–596.

[9] Luciana S. Buriol, Gereon Frahling, Stefano Leonardi, Alberto Marchetti-Spaccamela, and Christian Sohler. 2006. Counting Triangles in Data Streams. In *Proceedings of the Twenty-fifth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '06)*. ACM, New York, NY, USA, 253–262. https://doi.org/10.1145/1142351.1142388

[10] Roohollah Etemadi, Jianguo Lu, and Yung H. Tsin. 2016. Efficient Estimation of Triangles in Very Large Graphs. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM '16)*. ACM, New York, NY, USA, 1251–1260. https://doi.org/10.1145/2983323.2983849

[11] Oded Green, Pavan Yalamanchili, and Lluís-Miquel Munguía. 2014. Fast Triangle Counting on the GPU. In *Proceedings of the 4th Workshop on Irregular Applications: Architectures and Algorithms (IA3 '14)*. 1–8.

[12] Xiaocheng Hu, Yufei Tao, and Chin-Wan Chung. 2014. I/O-Efficient Algorithms on Triangle Listing and Counting. *ACM Trans. Database Syst.* 39, 4 (2014), 27:1–27:30.

[13] Shweta Jain and C Seshadhri. 2017. A Fast and Provable Method for Estimating Clique Counts Using Turán's Theorem. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 441–449.

[14] Madhav Jha, C. Seshadhri, and Ali Pinar. 2013. A Space Efficient Streaming Algorithm for Triangle Counting Using the Birthday Paradox. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '13)*. ACM, New York, NY, USA, 589–597. https://doi.org/10.1145/2487575.2487678

[15] Madhav Jha, C Seshadhri, and Ali Pinar. 2015. Path sampling: A fast and provable method for estimating 4-vertex subgraph counts. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 495–505.

[16] Hossein Jowhari and Mohammad Ghodsi. 2005. New Streaming Algorithms for Counting Triangles in Graphs. In *Proceedings of the 11th Annual International Conference on Computing and Combinatorics (COCOON'05)*. Springer-Verlag, Berlin, Heidelberg, 710–716. https://doi.org/10.1007/11533719_72

[17] John Kallaugher and Eric Price. 2017. A hybrid sampling scheme for triangle counting. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 1778–1797.

[18] Gabriela Kalna and Desmond J. Higham. 2007. A Clustering Coefficient for Weighted Networks, with Application to Gene Expression Data. *AI Commun.* 20, 4 (Dec. 2007), 263–271. http://dl.acm.org/citation.cfm?id=1365534.1365536

[19] Hyeonji Kim, Juneyoung Lee, Sourav S. Bhowmick, Wook-Shin Han, JeongHoon Lee, Seongyun Ko, and Moath H.A. Jarrah. 2016. DUALSIM: Parallel Subgraph Enumeration in a Massive Graph on a Single Machine. In *Proceedings of the 2016 International Conference on Management of Data (SIGMOD'16)*. 1231–1245.

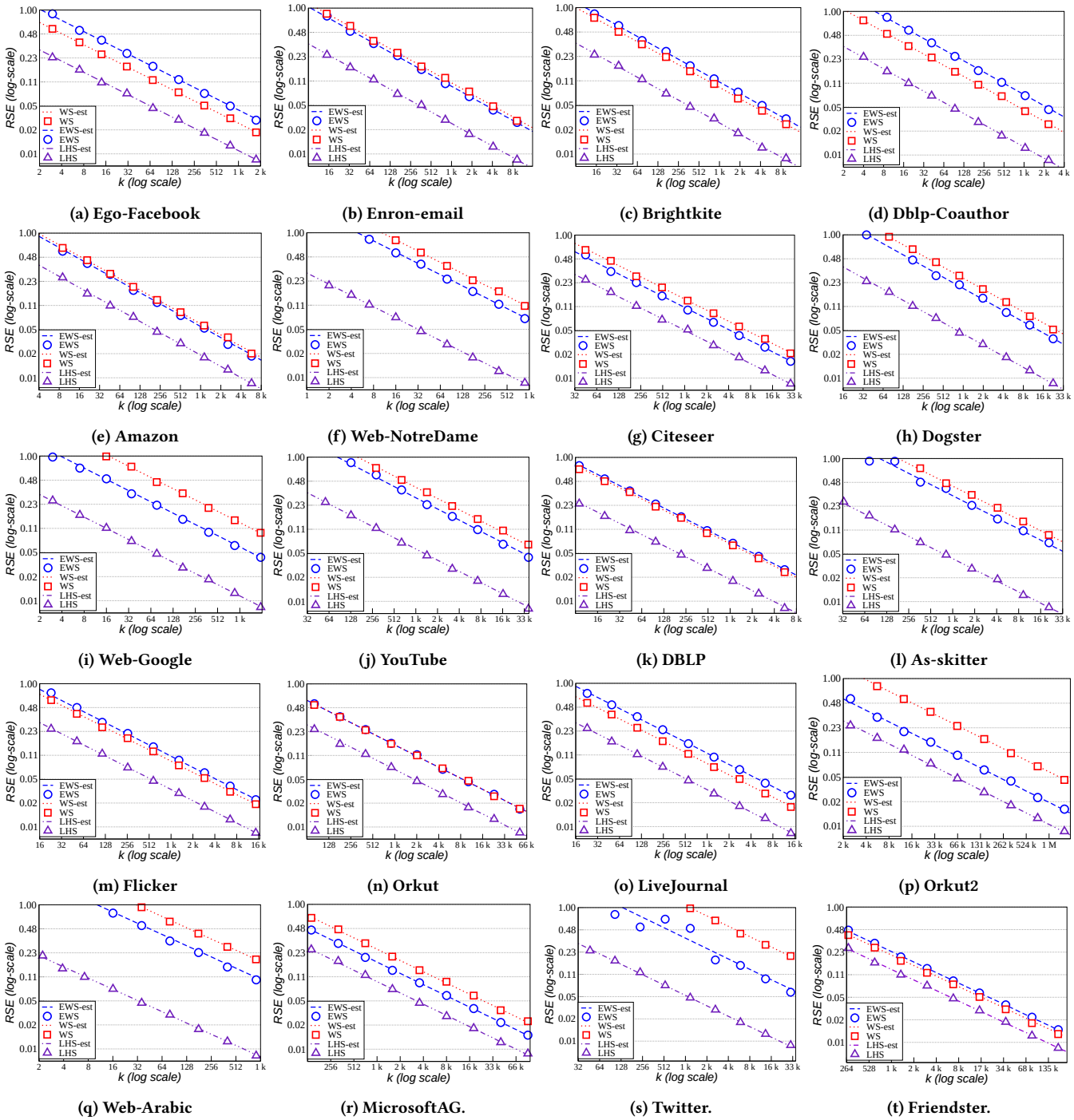


Figure 5: *RSE* **vs** k **for the datasets.**

[20] T. Kloks, D. Kratsch, and H. Müller. 2000. Finding and counting small induced subgraphs efficiently. *Inform. Process. Lett.* 74, 3 (2000), 115 – 121.

[21] Tamara G Kolda, Ali Pinar, Todd Plantenga, C Seshadhri, and Christine Task. 2014. Counting triangles in massive graphs with MapReduce. *SIAM Journal on Scientific Computing* 36, 5 (2014), S48–S77.

[22] Mihail N. Kolountzakis, Gary L. Miller, Richard Peng, and Charalampos E. Tsourakakis. 2010. *Efficient Triangle Counting in Large Graphs via Degree-Based Vertex Partitioning.* Springer Berlin Heidelberg, Berlin, Heidelberg, 15–24. https://doi.org/10.1007/978-3-642-18009-5_3

[23] Jérôme Kunegis. 2013. Konect: the koblenz network collection. In *Proceedings of the 22nd International Conference on World Wide Web.* ACM, 1343–1350.

[24] Matthieu Latapy. 2008. Main-memory triangle computations for very large (sparse (power-law)) graphs. *Theoretical Computer Science* 407, 1-3 (2008), 458–473.

[25] François Le Gall. 2014. Powers of Tensors and Fast Matrix Multiplication. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation (ISSAC '14)*. ACM, New York, NY, USA, 296–303. https://doi.org/10.1145/2608628.2608664

[26] Jure Leskovec and Andrej Krevl. 2015. SNAP Datasets: Stanford Large Network Dataset Collection.

[27] Yongsub Lim and U Kang. 2015. Mascot: Memory-efficient and accurate sampling for counting local triangles in graph streams. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 685–694.

[28] Rasmus Pagh and Charalampos E. Tsourakakis. 2012. Colorful triangle counting and a MapReduce implementation. *Inform. Process. Lett.* 112, 7 (2012), 277 – 281. https://doi.org/10.1016/j.ipl.2011.12.007

[29] Seyed-Vahid Sanei-Mehri, Ahmet Erdem Sariyuce, and Srikanta Tirthapura. 2018. Butterfly Counting in Bipartite Networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2150–2159.

[30] Thomas Schank and Dorothea Wagner. 2005. Finding, Counting and Listing All Triangles in Large Graphs, an Experimental Study. In *Proceedings of the 4th International Conference on Experimental and Efficient Algorithms (WEA'05)*. 606–609.

[31] Thomas Schank and Dorothea Wagner. 2005. Finding, Counting and Listing All Triangles in Large Graphs, an Experimental Study. In *Experimental and Efficient Algorithms: 4th International Workshop, WEA*, Sotiris E. Nikoletseas (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 606–609.

[32] Comandur Seshadhri, Ali Pinar, and Tamara G Kolda. 2013. Triadic measures on graphs: The power of wedge sampling. In *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 10–18.

[33] C. Seshadhri, Ali Pinar, and Tamara G. Kolda. 2014. Wedge sampling for computing clustering coefficients and triangle counts on large graphsâĂă. *Statistical Analysis and Data Mining* 7, 4 (2014), 294–307. https://doi.org/10.1002/sam.11224

[34] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June (Paul) Hsu, and Kuansan Wang. 2015. An Overview of Microsoft Academic Service (MAS) and Applications. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15 Companion)*. 243–246.

[35] Lorenzo De Stefani, Alessandro Epasto, Matteo Riondato, and Eli Upfal. 2017. Triest: Counting local and global triangles in fully dynamic streams with fixed memory size. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 4 (2017), 43.

[36] Siddharth Suri and Sergei Vassilvitskii. 2011. Counting Triangles and the Curse of the Last Reducer. In *Proceedings of the 20th International Conference on World Wide Web (WWW '11)*. ACM, New York, NY, USA, 607–614. https://doi.org/10.1145/1963405.1963491

[37] Charalampos E. Tsourakakis, U. Kang, Gary L. Miller, and Christos Faloutsos. 2009. DOULION: Counting Triangles in Massive Graphs with a Coin. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*. ACM, New York, NY, USA, 837–846. https://doi.org/10.1145/1557019.1557111

[38] Duru Türkoglu and Ata Turk. 2017. Edge-Based Wedge Sampling to Estimate Triangle Counts in Very Large Graphs. In *2017 IEEE International Conference on Data Mining, ICDM 2017*. 455–464.

[39] Leyuan Wang, Yangzihao Wang, Carl Yang, and John D. Owens. 2016. A Comparative Study on Exact Triangle Counting Algorithms on the GPU. In *Proceedings of the ACM Workshop on High Performance Graph Processing (HPGP '16)*.

[40] Zhi Yang, Christo Wilson, Xiao Wang, Tingting Gao, Ben Y Zhao, and Yafei Dai. 2014. Uncovering social network sybils in the wild. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 8, 1 (2014), 2.

RAQ: Relationship-Aware Graph Querying in Large Networks

Jithin Vachery
Computer Science and Engineering,
Indian Institute of Technology, Madras
jithin@cse.iitm.ac.in

Akhil Arora
EPFL
akhil.arora@epfl.ch

Sayan Ranu
Computer Science and Engineering,
Indian Institute of Technology, Delhi
sayanranu@cse.iitd.ac.in

Arnab Bhattacharya
Computer Science and Engineering,
Indian Institute of Technology, Kanpur
arnabb@cse.iitk.ac.in

ABSTRACT

The phenomenal growth of graph data from a wide variety of real-world applications has rendered graph querying to be a problem of paramount importance. Traditional techniques use structural as well as node similarities to find matches of a given query graph in a (large) target graph. However, almost all existing techniques have tacitly ignored the presence of *relationships* in graphs, which are usually encoded through interactions between node and edge labels. In this paper, we propose *RAQ—**R**elationship-**A**ware Graph **Q**uerying*—to mitigate this gap. Given a query graph, RAQ identifies the k best matching subgraphs of the target graph that encode similar relationships as in the query graph. To assess the utility of RAQ as a graph querying paradigm for knowledge discovery and exploration tasks, we perform a *user survey* on the Internet Movie Database (IMDb), where an overwhelming 86% of the 170 surveyed users preferred the relationship-aware match over traditional graph querying. The need to perform subgraph isomorphism renders RAQ NP-hard. The querying is made practical through *beam stack search*. Extensive experiments on multiple real-world graph datasets demonstrate RAQ to be effective, efficient, and scalable.

ACM Reference Format:
Jithin Vachery, Akhil Arora, Sayan Ranu, and Arnab Bhattacharya. 2019. RAQ: Relationship-Aware Graph Querying in Large Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313448

1 INTRODUCTION

Graphs act as a natural choice to model data from several domains. Examples include social networks [13], knowledge graphs [8, 11], and protein-protein interaction networks (*PPIs*) [27, 28]. Consequently, graph-based searching and querying have received significant interest in both academia [15,21,22,32,35,40] and industry (e.g., Facebook's Graph Search [3] and Google's Knowledge Graph [4]).

One of the most common queries in these frameworks is to find similar embeddings of a query graph q in a much larger target graph G. More formally, given a query graph q, a target graph G and a similarity function $sim(q, g)$, the goal is to identify the top-k most similar subgraphs $g \subseteq G$ to the query graph q. Traditional similarity functions consider two graphs as similar if they are structurally similar and they contain similar nodes [21, 22, 29, 48]. Two nodes are said to be similar if they are represented by similar feature vectors; the most common form being just a node label. Despite significant advancements in the field of graph querying [15, 22, 48], this commonly adopted definition of similarity is oblivious to the presence of *relationships* in the graphs. In this work, we bring the power of relationships to graph querying.

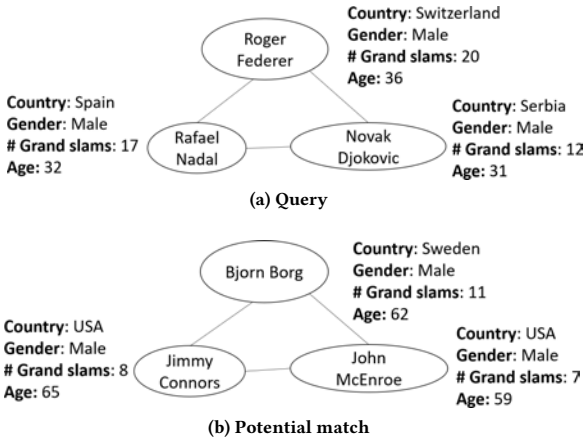


Figure 1: Two subgraphs from the Wikipedia knowledge graph. For simplicity, we drop the directionality and model them as an undirected graph.

To give an example of a relationship, consider the entity tuples ⟨Rafael Nadal, Roger Federer⟩ and ⟨John McEnroe, Bjorn Borg⟩ from the Wikipedia knowledge graph. Although the entities in these two tuples are different, both tuples encode a similar relationship; they represent players who played during the same era and won multiple grand slam tournaments between them. Annotating knowledge graphs with relationship information in the form of edge labels has been shown to expose a higher level of semantics than simple entity-to-entity matching [19, 44].

Fig. 1a presents a query subgraph from Wikipedia knowledge graph containing well-known tennis players of the current era. Wikipedia further characterizes each node with a bunch of summary features as shown in the figure. Now, consider the task of identifying a subgraph from Wikipedia similar to the query in Fig. 1a. Consider Fig. 1b, which depicts another group of well-known tennis players: Bjorn Borg, Jimmy Connors and John McEnroe. Would a tennis fan consider these two groups to be similar? There is a strong possibility that she would since both the groups respectively represent players

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313448

who played during the same era and won multiple grand slam tournaments among them.

Let g_1 and g_2 be two graphs. Most of the existing techniques [18, 22, 29, 47, 48] rely on mapping each node in g_1 to some node in g_2, and the cumulative similarity of the mapped node pairs dictate the overall similarity score between g_1 and g_2. When such a strategy is applied to the above two subgraphs (Fig. 1a and 1b), none of the possible node mappings would return a high score since their feature vectors are not similar. For example, Federer is not similar to any of the nodes in Fig. 1b, since he is from a different generation as indicated by his "Age", and belongs to a different country. The "Gender" and "# Grand slams" will fetch some similarity, however, that may not be enough to offset the dissimilarity in the other two features. More fundamentally, although a graph contains a *group/collection* of nodes, *existing techniques* treat each node as an *independent entity* while comparing two graphs. Consequently, *group-level relationships*, that can be identified only by analyzing the *group-level interactions* between node feature vectors, are not reflected in the similarity score. For example, the relationship that each group contains players from roughly the same generation, as hinted by the "Age" feature, cannot be inferred through node similarity alone, where nodes are treated independently. This observation forms the core motivation behind our work to move beyond node similarity, and match subgraphs based on the *relationships encoded through non-trivial interactions between node features*.

Several important questions arise at this juncture. *What is the precise definition of a relationship? How do we quantify the strength of a relationship? Are all relationships equally important?* In this paper, we answer these questions. In sum, we make the following key contributions:

- We propose a novel graph querying paradigm, *RAQ (Relationship-Aware Graph Querying)*, that incorporates the notion of *relationship* in its similarity function (Sec. 3). The proposed formulation automatically mines the relationships manifested in the query graph and ranks their importance based on their *statistical significance* (Sec. 4).
- We design the RAQ search algorithm, which employs *beam stack search* on an R-tree to identify good *candidates*. These candidates are further refined through *neighborhood signatures* constructed using random walks with restarts to prune the search space and compute the *optimal* answer set. (Sec. 5).
- Empirical evaluation (Sec. 6) on real datasets establishes three key properties. (1) RAQ produces results that are *useful*. This is validated by two user surveys. (2) RAQ *complements* existing graph similarity metrics by finding results that traditional techniques are unable to find. (3) RAQ is up to 5 times faster than baseline strategies and scales to million-sized networks.

  The code is available at: https://github.com/idea-iitd/RAQ.

2 RELATED WORK

Research work done in both exact and approximate (sub-)graph querying have employed a plethora of similarity functions. The most prominent of them being graph edit distance (GED) [47, 48], maximum and minimum common subgraph [12, 16], edge misses [41], structural similarity [15, 21, 22, 38, 40, 42], node-label mismatches [38, 40], and statistical significance of node-label distributions [14]. However, all these methods operate oblivious to the presence of *relationships in the query graph*.

Query-by-example Paradigm: The need to query based on relationships is touched upon by the recent line of work on Query-by-example [19, 24–26]. All these techniques take an edge or a set of edges as input, and this input is treated as an *example* of what the user wants to find. Each exemplar edge connects two entities of the knowledge graph and the edge label denotes the relationship between these entities. Typical relationships in a knowledge graph are "friendOf", "locatedIn" etc. Given the exemplar edges, the common goal in this line of work is to identify subgraphs from the target graph that are connected by the same edge labels as expressed in the exemplars. Our work is motivated by the same line of thought and revamps this paradigm through the following innovations.

(1) **Mining relationships:** While [19, 24–26] assume that relationships are explicitly provided in the form of edge labels, we *mine* these relationships from feature-value interactions observed between a pair of nodes. Thus, RAQ is not constrained by the explicit availability of edge relationships. This results in a more powerful model since it is hard to capture all possible relationships through a small set of edge labels. Furthermore, unlike existing query-by-example techniques, RAQ is not limited to edge-labeled (relationship-annotated) graphs alone, and exposes a more generic framework capable of operating on graphs with both node and edge labels.
(2) **Multiple relationships:** Existing techniques assume each edge to be associated with only one relationship. In our formulation, each edge encodes d relationships corresponding to each of the d features characterizing a node.
(3) **Relationship importance:** All relationships in the exemplar edges are assumed to be of equal importance. In the proposed paradigm, we analyze all of the identified relationships and quantify their importance through statistical significance. As we will see later in Sec. 6.3, users find this weighting relevant.
(4) **Relationship similarity:** Existing query-by-example techniques are incapable of computing similarity between relationships. This is a natural consequence of the fact that relationships are reduced to *categorical edge labels*. Thus, relationships between two pairs of nodes are either considered *identical or different*. In contrast, *RAQ is not limited to a binary matching*. For example, the relationship with respect to "# Grand slams" between Federer and Nadal is similar to that between Bjorg and McEnroe, but not identical.

Overall, the proposed paradigm is more generic, and therefore more challenging to solve algorithmically.

3 PROBLEM FORMULATION

The input to our problem is a *target graph*, a *query graph*, and the value k corresponding to the top-k matches that need to be identified.

Definition 1 (Target Graph). *The target graph $G = (V, E, \mathbb{F})$ contains a node set V, an edge set E, and a set of d-dimensional feature*

vectors $\mathbb{F}$, where each feature vector $f(v) = [f_1(v), \ldots, f_d(v)]$ characterizes a node $v \in V$. Optionally, each edge may also be annotated with an edge label.

Without loss of generality, we assume that for each feature value $f_i(v)$, $0 \leq f_i(v) \leq 1$. A query graph $q = (V_q, E_q, \mathbb{F}_q)$ follows the same notational structure as the target graph. Both the target graph and the query graph can be either directed or undirected. Our goal is to identify *isomorphic* embeddings of the query graph within the target graph, such that they encode relationships similar to that manifested in the query. Graph $g_1 = (V_1, E_1, \mathbb{F}_1)$ is *isomorphic* to $g_2 = (V_2, E_2, \mathbb{F}_2)$ if there exists a bijection ϕ such that for every vertex $v \in V_1$, $\phi(v) \in V_2$ and for every edge $e = (u, v) \in E_1$, $\phi(e) = (\phi(u), \phi(v)) \in E_2$ and the labels of e and $\phi(e)$ are the same. Among all isomorphic embeddings of the query graph q in the target graph G, we want to select the k embeddings that possess the highest *relationship-aware similarity (RAS)* to q (defined formally in Def. 3).

Problem 1 (Top-k Relationship-aware Querying). *Given a query graph q, a target graph G, and a positive integer k, identify the graphs $\mathbb{A} = \{g_1, \cdots, g_k\}$, such that (1) $\forall g_i \in \mathbb{A}$, g_i is a subgraph of G, (2) $\forall g_i \in \mathbb{A}$, g_i is isomorphic to q, and (3) $\nexists g' \subseteq G$ such that $RAS(g', q) > RAS(g_i, q)$ for some $g_i \in \mathbb{A}$.*

The main task is to design the relationship-aware similarity function $RAS(g, q)$ for any two isomorphic graphs g and q.

4 RELATIONSHIP-AWARE SIMILARITY

We first define what a relationship is. Intuitively, each feature represents the characteristics of a node and the relationship between two nodes is inferred by studying the interplay between their feature values. For example, considering Djokovic and Nadal, both entities are of the same gender, similar age, have won a large number of grand slam tournaments, but are citizens of different countries. Interactions between each corresponding feature of the connected nodes generate a relationship, and hence, a d-dimensional feature vector would produce d relationships per edge. We capture these relationships in the form of a *relationship vector* $s(e)$ corresponding to each edge $e = (u, v)$.

Definition 2 (Relationship Vector). *The relationship vector, $s(e)$, of an edge $e = (u, v)$ captures the features that are preserved in the relationship between u and v. Formally, $s(e) = [s(f_1, e), \ldots, s(f_d, e)]$ where*

$$s(f_i, e) = \begin{cases} \Gamma(f_i(u), f_i(v)), & \text{if } f_i \text{ is real valued} \\ 1, & \text{if } f_i \text{ is categorical and } f_i(u) = f_i(v) \\ 0, & \text{if } f_i \text{ is categorical and } f_i(u) \neq f_i(v) \end{cases}$$

$$\text{where } \Gamma(x, y) = \begin{cases} 1, & \text{if } x = y = 0 \\ \min\{x, y\} / \max\{x, y\}, & \text{otherwise} \end{cases} \quad (1)$$

Example 1. *The relationship vector $s(e)$ for the edge e connecting Nadal and Djokovic in Fig. 1a is $[0, 1, 0.71, 0.97]$. The second dimension is 1 since it is a categorical feature and both are males. The third dimension is non-categorical and thus, the similarity between them is $\frac{12}{17}$. Similarly, the relationship vector of the edge between Connors and McEnroe in Fig. 1b is $[1, 1, 0.88, 0.91]$.*

From its definition, each dimension in the relationship vector takes a value in the range $[0, 1]$. A value close to 1 indicates that both endpoints of the edge share similar values. Two edges e and e' are considered similar if they encode similar relationships. Thus, to quantify their similarity, we compute the similarity between their relationship vectors:

$$eSim(e, e') = \sum_{i=1}^{d} w(i) \cdot \Gamma\left(s(f_i, e), s(f_i, e')\right) \quad (2)$$

$eSim(e, e')$ is a weighted *min-max* similarity between the two relationship vectors. The *weight* $w(i)$ of the i^{th} relationship (feature) represents its relative importance with respect to the other relationships, and the function $\Gamma\left(s(f_i, e), s(f_i, e')\right)$ operates as defined in Eq. 1. We add the constraint that $\sum_{i=1}^{d} w_i = 1$.

Example 2. *Assume that each relationship is of equal importance and, therefore, $\forall i$, $w_i = \frac{1}{d}$. The similarity between the edges Nadal-Djokovic and Connors-McEnroe is, thus, $0.25 \times 0 + 0.25 \times 1 + 0.25 \times \frac{0.71}{0.88} + 0.25 \times \frac{0.91}{0.97} = 0.68$.*

Given two isomorphic subgraphs q and g, and an isomorphic mapping ϕ from the edges in q to the edges in g, the *relationship-aware similarity (RAS)* with respect to the mapping ϕ is the similarity between the mapped edges from q to g:

$$RAS_{\phi}(q, g) = \sum_{\forall e_q = (u_q, v_q) \in E_q} eSim\left(e_q, \phi(e_q)\right) \quad (3)$$

Definition 3 (Relationship-aware Similarity). *The relationship-aware similarity (RAS) between q and g is the maximum RAS under all possible isomorphic mappings:*

$$RAS(q, g) = \max_{\phi} \left\{RAS_{\phi}(q, g)\right\} \quad (4)$$

Quantifying Relationship Importance: The obvious question now is how to infer the weights in $eSim(e, e')$ (Eq 2). One option is to ask the user to provide the weights. However, such a strategy may not be practical. Specifically, each edge may encode a large number of relationships corresponding to features characterizing the participating nodes, and providing a weight with respect to each relationship is hard for general users. Also, providing a large number of input parameters is a cumbersome procedure. Hence, ideally, the weights should be automatically inferred.

The simplest approach is to assume that all relationships are of equal importance, i.e. $\forall i, w_i = \frac{1}{d}$. This assumption however, may be unrealistic. To elaborate, *is the relationship that both Nadal and Djokovic are males as important as the relationship that both of them have won a large number of grand slam tournaments?* Intuitively, such an assumption appears to be odd.

More importantly, should the importance be constant under all situations and should it ignore the feature values. For example, consider an edge between two tennis players who have won no grand slams. In such a case, both would have a similarity of 1 in the "# Grand Slams" dimension. In that case, is the importance of grand slams the same between these two players as compared to Nadal and Djokovic? From a human psychological point of view, we value events that are rare. The relationship between Nadal and Djokovic stands out since they have won so many grand slams. We capture this intuition through *statistical significance*. Higher the statistical significance of a relationship, more is its weight.

4.1 Statistical Significance of Relationships

Statistical significance tests quantify whether an event occurred due to chance alone or is a result of some additional factor. Several statistical tests exist [20], such as the chi-square test [7], p-value [31,33,34], g-test [45], Kruskal-Wallis test, Anderson-Darling test, Shapiro-Wilk test, etc., to quantify the statistical significance of an event. However, none of the techniques work directly on both categorical data (e.g., country) as well as continuous-valued data (e.g., age). Thus, we need to either map continuous data into a discrete space or convert discrete, categorical data into a continuous domain. Since converting categorical data such as gender and country into a continuous space is not feasible, we discretize continuous data into bins. We use the *chi-squared statistic* [36] to measure statistical significance since it is known to produce robust results [7,37] and is efficient to compute.

Converting continuous data into discrete bins has been extensively studied [23]. While any discretization technique can be used, we use kernel-density estimation based method [10] since it is unsupervised and non-parametric and, therefore, easy to apply on any continuous valued variable.

Let x be some event in a random experiment and S be the sample space. For all outcomes $y \in S$, let O_y represent the number of times outcome y has occurred. The chi-square statistic, X^2, measures the statistical significance of the event x through the deviation of the *observed* frequency of all possible outcomes O_y from their *expected* frequency E_y given by the *null model*:

$$X^2 = \sum_{\forall y \in S} \frac{(O_y - E_y)^2}{E_y}. \tag{5}$$

The higher the chi-square, the more statistically significant the observed event is.

In our problem, an event corresponds to a relationship. A relationship is manifested in the form of feature values assumed by the two endpoints of an edge. Thus, we should compute the observed frequency of a relationship and compare it to its expected frequency as per the chi-square protocol. Towards that end, each edge $e = (u, v)$ gives rise to a *relationship tuple* $\tau = (f_i(u), f_i(v))$ corresponding to feature f_i. Let $\mathcal{V}(f_i)$ represent the set of unique tuples assumed by feature f_i across all edges in a graph g. Then, the *observed frequency*, O_τ, for any relationship tuple $\tau \in \mathcal{V}(f_i)$ is simply the number of times τ is encountered in g.

Example 3. *Let us revisit Fig. 1a. As a pre-processing step, we would first discretize "# Grand slams" and "Age" into discrete categories. Let us assume that grand slam wins are bucketed into the following bins – Poor: 0, Good: 1-5, Very Good: 6-10, Great: 11-15, All time great (ATG):> 15. Therefore, $\mathcal{V}$("# Grand slam") contains the tuples (ATG,ATG) and (ATG,Great) with observed frequencies 1 and 2 respectively. Similarly, $\mathcal{V}$("Gender") = {(Male, Male)}, where observed frequency $O_{(Male,Male)} = 3$.*

Null Model: We assume that relationship tuples in the query graph are drawn independently and randomly from the distribution of tuples in the target graph G. If the distribution of a relationship tuple in the query graph deviates significantly from the distribution in target graph, we call the corresponding relationship statistically significant.

Let $P(f_i) = \{p_1, \ldots, p_{|\mathcal{V}(f_i)|}\}$ denote the probability distribution of relationship tuples for feature f_i in the target graph $G = (V, E, \mathbb{F})$. Here, $p_j \in P(f_i)$ represents the probability of finding tuple $\tau_j \in \mathcal{V}(f_i)$ in the target graph and is equal to the ratio $\frac{freq(\tau_j)}{|E|}$, where $freq(\tau_j)$ denotes the frequency of tuple τ_j in the target graph. Since each relationship tuple of feature f_i in the query graph is drawn randomly and independently from $P(f_i)$, the *expected frequency* E_τ of any tuple τ in the query graph $q = (V_q, E_q, \mathbb{F}_q)$ is

$$E_\tau = |E_q|.p_\tau \tag{6}$$

The chi-square value of f_i with observed frequency O_τ is, therefore,

$$X^2_{f_i} = \sum_{\forall \tau \in \mathcal{V}(f_i)} \frac{(O_\tau - E_\tau)^2}{E_\tau} \tag{7}$$

Example 4. *Let us compute the significance of the grand slam and gender relationships in Fig. 1a. Since very few of the tennis players have won a grand slam tournament, we assume that in the target graph, 5% of the edges have the relationship tuple (ATG, Great) and 1% have the tuple (ATG, ATG). Therefore, the expected frequency of (ATG, Great), (ATG, ATG) and all other tuples in Fig. 1a is $0.05 \times 3 = 0.15$, $0.01 \times 3 = 0.03$ and $0.94 \times 3 = 2.82$ respectively. Consequently,*

$$X^2_{\#Grandslam} = \frac{(2-0.15)^2}{0.15} + \frac{(1-0.03)^2}{0.03} + \frac{(0-2.82)^2}{2.82} = 57$$

In the gender relationship, let us assume that in the target graph 50% of the edges contain the tuple (Male, Male), 40% contain (Female, Female) and 10% contain (Female, Male). Therefore, the expected frequency of (Male, Male), (Female, Female) and (Female, Male) in Fig. 1a are 1.5, 1.2 and 0.3 respectively. Consequently, $X^2_{Gender} = 3$. Therefore, we conclude that the grand slam relationship is much more significant than the relationship that all players are males.

Since the importance of each relationship is proportional to its statistical significance, the weight w_i of the i^{th} feature in Eq. (2) is

$$w_i = \frac{X^2_{f_i}}{\sum_{j=1}^{d} X^2_{f_j}} \tag{8}$$

5 RELATIONSHIP-AWARE QUERYING (RAQ)

The naïve approach to solve a top-k query is to first enumerate all possible subgraphs of the target graph, identify those that are isomorphic to the query, and then rank these isomorphic subgraphs based on their relationship-aware similarity. However, this approach is prohibitively expensive since the target graph contains exponential number of subgraphs. In fact, the problem is NP-complete since finding isomorphic embeddings of a query graph within the target graph reduces to *subgraph isomorphism* [18]. We, therefore, need to design a more scalable searching algorithm.

5.1 Bottom-Up Exploration

We first analyze a bottom-up exploration strategy. First, we pick an arbitrary edge e_q from the query graph q and map it to an arbitrary edge e of the target graph. We call e_q the *seed edge*. This forms an 1-edge common subgraph of q and G. Next, we try to grow it into a 2-edge subgraph by extending both e_q and e through one of their neighboring edges such that the two extended subgraphs are isomorphic and, therefore, a common subgraph of q and G.

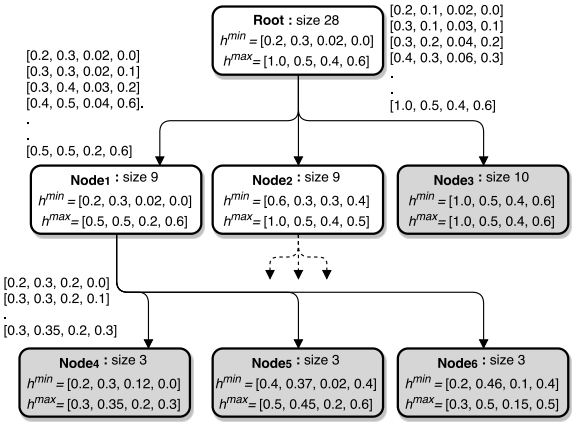


Figure 2: The structure of the R-tree. The leaf nodes are colored in grey. We do not explicitly show the relationship vectors in the leaf nodes due to space limitations.

This procedure of growing using a neighboring edge continues iteratively till either of the following two cases become true:

(1) The subgraph cannot be grown any further such that it remains a subgraph of both q and G. In this case, we fail to find an isomorphic embedding of q.
(2) The subgraph becomes isomorphic to q. We compute its similarity to q and store it as a candidate.

Once the growing procedure terminates, the search restarts by mapping the seed query edge to another edge of G.

Once all possible mappings of the seed edge are explored, we pick another query edge as the seed and repeat the procedure. Finally, the k highest scoring isomorphic embeddings are returned.

The bottom-up exploration is brute-force in nature and explores all possible combinations. However, it offers some concrete directions to improve.

(1) **Seed selection:** Can the seed edge be more strategically selected so that good candidates are found early?
(2) **Seed mapping:** Can the seed edge be mapped to an edge e in the target graph more intelligently so that with a high likelihood an isomorphic embedding is found around e, which is also highly similar to query graph q?
(3) **Pruning bad candidates:** Can a candidate subgraph be pruned early by guaranteeing that it cannot be in the top-k?

5.2 Seed Selection and Mapping

Intuitively, we should explore the mapping that has the highest chance of growing into high-scoring isomorphic match to the query. An obvious approach is to compute edge similarity (Eq. 2) between all pairs of query edges and target graph edges, and choose the most similar pair. The computation cost for this is $O(|E_q| \cdot |E|)$. Since $|E|$ is a large number, we need a more efficient algorithm for this task. The similarity between two edges is essentially a weighted min-max similarity between two high-dimensional points, since each edge is represented by a d-dimensional relationship vector. Indexing high-dimensional points have been extensively studied [9], and we choose R-tree [17] for our framework. R-tree recursively partitions the entire dataset of relationship vectors into *MBRs*.

Definition 4 (Minimum Bounding Rectangle, MBR). *Given a set of d-dimensional relationship vectors, $\mathbb{D} = \{s(e_1), \cdots, s(e_m)\}$, an MBR $\mathcal{H}$ on $\mathbb{D}$ is the smallest axis-parallel d-dimensional hyper-rectangle that encloses all vectors in $\mathbb{D}$.*

An MBR $\mathcal{H}$ can be uniquely represented by the co-ordinates of its *smallest* and *largest* diagonal points h^{min} and h^{max} respectively. More specifically, $h^{min} = [h_1^{min}, \ldots, h_d^{min}]$, where $h_i^{min} = \min_{\forall s(e) \in \mathbb{D}} \{s(f_i, e)\}$, and h^{max} is defined analogously. An example of an R-tree is shown in Fig. 2.

We use the notation $e \in \mathcal{H}$ to denote that the relationship vector $s(e)$ of e is contained within MBR $\mathcal{H}$. Mathematically, this means $\forall i,\ h_i^{min} \leq s(f_i, e) \leq h_i^{max}$. We define the similarity between a relationship vector $s(e_q)$ of an edge e_q and an MBR $\mathcal{H}$ as follows:

$$eSim(e_q, \mathcal{H}) = \max_{\forall e \in \mathcal{H}} \{eSim(e_q, e)\}$$
$$= \sum_{i=1}^{d} w(f_i, e_q) \cdot \max_{s(f_i,e) \in [h_i^{min}, h_i^{max}]} \left\{\Gamma(s(f_i, e_q), s(f_i, e))\right\}$$

Simply put, $eSim(e, \mathcal{H})$ provides the maximum similarity between e and any possible edge contained within $\mathcal{H}$. This similarity value can be used to provide the following upper bound on $\mathbb{D}$.

Theorem 1. *If $\mathcal{H}$ is an MBR on a set of d-dimensional relationship vectors $\mathbb{D}$ and e_q is a query edge then,*

$$\max_{\forall s(e) \in \mathbb{D}} \{eSim(e_q, e)\} \leq eSim(e_q, \mathcal{H}).$$

We use Theorem 1 to search for the most similar edge to the query edge e_q using the *best-first* search. Specifically, starting from the root node, we prioritize all child nodes based on the distance of their MBRs to the query edge. The best child node is then chosen to explore next and the process is recursively applied till we reach a leaf node (MBR without a child). Once a leaf node is reached, we compute the similarity to all nodes in this MBR and retain the highest scoring one. The *best-first* search procedure stops when all MBRs that remain to be explored have a maximum possible similarity smaller than the highest scoring target edge we have found till now. We discuss this more formally in Sec. 5.4.

5.3 Avoiding Local Maxima

While the above algorithm is efficient in locating a similar edge, it is prone to getting stuck in a local maxima. Specifically, searching for similar edges may lead us to a good match in a bad neighborhood (local maxima), and such leads should be avoided. A neighborhood is bad if one or both of the following are true:

(1) The structure around the mapped edges is different.
(2) The distributions of relationships around the mapped edges are different and, therefore, even if an isomorphic mapping is found, the similarity among the mapped edges (relationship vectors) is likely to be low.

Can we avoid dissimilar neighborhoods without explicitly performing isomorphism tests?

Neighborhood Signature: We answer the above question by mapping the neighborhood of an edge into a feature vector such that if two neighborhoods are similar, then their feature vectors are similar as well. This is achieved using *Random Walks with Restarts (RWR)*. RWR starts from a source node and iteratively jumps from

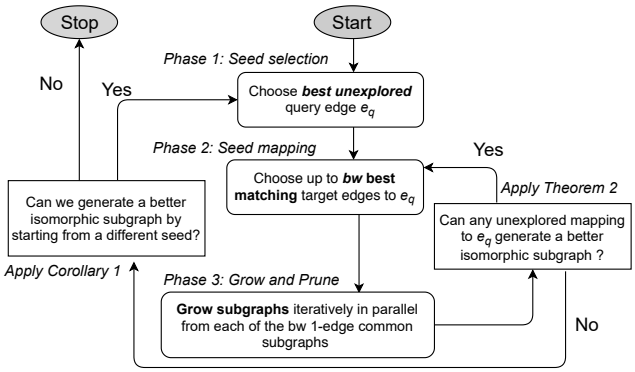


Figure 3: Flowchart of the searching algorithm in RAQ.

the current node to one of its neighbors with equal probability. To bound the walker within a δ-hop neighborhood with a high likelihood, RWR employs a restart probability of $\alpha = \frac{1}{\delta}$. At any given state, with α probability, the walker returns to the source. Otherwise, the walker jumps to a neighbor.

Let e be the edge whose neighborhood needs to be captured. We arbitrarily choose one of the endpoints of e as the source node. RWR is initiated from it and for each edge e' that is traversed, a *counter* $c(e')$ records the number of times e' have been traversed. The RWR is performed for a large number of iterations after which, we compute the *neighborhood signature* of edge e as a d-dimensional feature vector $N(e) = [n_1, \cdots, n_d]$, where

$$n_i = \sum_{\forall e' \in \mathbb{E}} \frac{c(e')}{C} \times s(f_i, e') \tag{9}$$

where $\mathbb{E}$ is the set of all edges visited in the RWR and s is as defined in Def. 2 and C is the total number of iterations.

The signature captures the distribution of relationships around the source edge e. Edges that are closer to e have a higher say. Consequently, the signature not only captures the relationship vectors in the neighborhood, but is also structure sensitive. While it is still possible for two dissimilar neighborhoods to have similar signatures, the chances that two dissimilar neighborhoods produce similar signatures across all d dimensions is small.

We construct the neighborhood signature of all edges in the target graph as part of the index construction procedure. The signatures of the edges in the query graph are computed at query time. The neighborhood hop parameter δ is typically a small value since query graphs for knowledge discovery tasks are normally small [25]. In our empirical evaluation, we set δ to 3.

5.4 RAQ Search Algorithm

Fig. 3 outlines the flowchart and Alg. 1 presents the pseudocode. There are three major phases: selecting the most promising query edge e_q for exploration (*seed selection phase*), identifying the best matching target edges (*seed mapping phase*), and growing subgraphs from these initial seeds in bottom-up manner (*growth phase*).

Before we execute any of the phases, two operations are performed. First, we compute the neighborhood signatures of each edge in the query (line 1). Second, we use a priority queue called *Ans*, which stores the k most similar subgraphs identified till now (line 2). Initially, *Ans* is empty.

5.4.1 Phase 1: Seed Selection. We select the unexplored query edge $e_q \in E_q$ that has the highest similarity to the root node of R-tree (line 5). An edge is *unexplored* if it has not already been selected in this phase in some earlier iteration.

5.4.2 Phase 2: Seed Mapping. We use *best-first search* to find the leaf node in R-tree that has the highest similarity to e_q. In this leaf node, we perform a *beam stack search* [49] with *beam width* bw (lines 13-20). Instead of mapping e_q to the most similar target edge, beam stack search selects the bw target edges that have the highest neighborhood similarity to e_q and explores them further. The *neighborhood similarity* between two edges e_q and e is the weighted L_1 distance between their neighborhood signatures.

$$\text{Neighborhood_Similarity}(e_q, e) = \sum_{i}^{d} w_i \cdot \|N(i, e_q) - N(i, e)\|$$

Recall that w_i is the importance of the i^{th} relation and more important relations have a higher say in defining the neighborhood similarity.

5.4.3 Phase 3: Grow and Prune. Each of the bw edges selected in the previous phase generates a common 1-edge subgraph of the query and the target graph. We now grow them one edge at time to larger subgraphs (line 19). On any such subgraph $g = (V_g, E_g)$ we can apply the following bound.

Theorem 2. *The maximum similarity of any isomorphic match to query graph q formed by growing g further is $\widehat{RAQ}(q, g) = RAQ(q, g) + |E_q| - |E_g|$.*

Proof. The maximum similarity between any two edges is 1 (Eq. 2). Therefore, the maximum similarity contributed from the edges yet to be added is at most $|E_q| - |E_g|$. The theorem follows. □

Corollary 1. *Let e_q be a query edge and $\mathcal{H}$ be an MBR in the R-tree. The maximum similarity of any isomorphic subgraph formed by mapping e_q to any edge contained within $\mathcal{H}$ is $\widehat{RAQ}(e_q, \mathcal{H}) = eSim(e_q, \mathcal{H}) + |E_q| - 1$.*

Equipped with the above upper bounds, we prioritize each of the initial 1-edge common subgraphs based on $\widehat{RAQ}(q, g)$ (Theorem 2). More specifically, we initialize another priority queue PQ and insert all 1-edge subgraphs in PQ (line 14). The subgraph g with the highest upper bound is popped from PQ. We check if $\widehat{RAQ}(q, g)$ is larger than the k^{th} highest similarity score in *Ans* (line 17). If yes, we explore all possible single edge extensions of g to create common subgraphs with one more edge (line 20). If g becomes isomorphic to q after extension, then we add it to *Ans*. Otherwise, we insert it back to PQ (line 20). Otherwise, if the check (at line 17) fails, then we are guaranteed that none of the unexplored subgraphs in PQ can lead to a better solution than what we already have and, hence, Phase 3 completes (lines 7-18).

5.5 Properties of RAQ Framework

Correctness Guarantee: The RAQ framework provides the *optimal answer set*. We do not prune off any possibility without applying

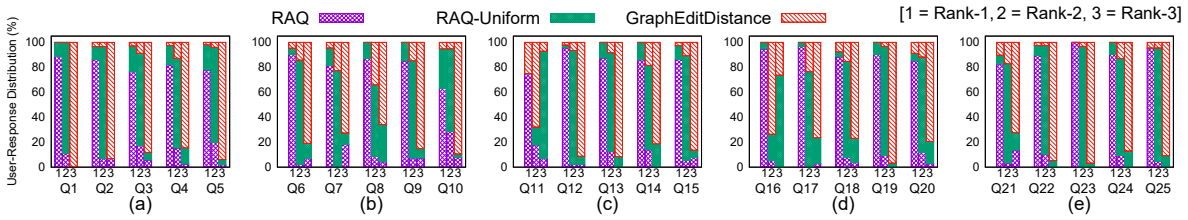


Figure 4: The distribution of the received user responses.

Algorithm 1 RAQ Search Algorithm

Input: $G = (V, E, \mathbb{F})$, $q(V_q, E_q, \mathbb{F}_q)$, k, $RAQIndex$
Output: Top-k subgraphs of G with highest relationship-aware similarity to q.

Compute neighborhood signature for each edge in q
$Ans \leftarrow \emptyset$ {Priority queue of maximum size k. Stores isomorphic matches in descending order of similarity.}
$Cands \leftarrow \emptyset$
while $\exists e \in E_q$, $\widehat{RAQ}(e, rootNode.MBR) > Ans.leastValue()$ and e has not been explored **do**
  $e_q \leftarrow$ unexplored query edge with highest similarity to MBR in root node
  Initialize $Cands$ with the root node of R-tree
  while $Cands$ is not empty **do**
    $n \leftarrow Cands.poll()$
    $max \leftarrow \widehat{RAQ}(e_q, n.MBR)$.
    if $max \leq Ans.leastValue()$ **then**
      break
    if n is a leaf node **then**
      while $\widehat{RAQ}(e_q, n.MBR) > Ans.leastValue()$ and n contains unexplored edges **do**
        $PQ \leftarrow$ Priority queue containing bw unexplored edges with similar neighborhoods to e_q
        while PQ is not empty **do**
          $g \leftarrow PQ.poll()$
          if $\widehat{RAQ}(g, q) \leq Ans.leastValue()$ **then**
            break
          $\mathbb{G} \leftarrow$ Grow g through all possible 1-edge extensions to form common subgraphs
          Add all isomorphic matches in $\mathbb{G}$ to Ans and remaining subgraphs in $\mathbb{G}$ to PQ
    else
      Add all children of n to $Cands$.
return Ans

Theorem 2 or Corollary 1. Consequently, we do not lose out on any candidate that can be in the top-k answer set.

Index construction cost: The cost of constructing the R-tree is $O(d|E|\log(|E|))$. We also compute and store the neighborhood signature of each edge through random walk with restarts. This step can be performed in $O(|V|^2)$ time for the entire graph [43, 46]. Thus, the total computation cost is $O(|V|^2 + d|E|\log(|E|))$.

Memory Footprint: R-tree consumes $O(d|E|)$ space since it stores d-dimensional relationship vectors of all edges. The neighborhood signatures are also d-dimensional feature vectors and hence the total cost is bounded by $O(d|E|)$.

Querying Time: As stated earlier, the problem is NP-complete. Hence, akin to most existing graph search algorithms [18, 22, 25], the worst case running time remains exponential. However, with the application of efficient heuristics and effective pruning bounds, we compute the top-k answer set in milliseconds as elaborated next.

6 EXPERIMENTS

In this section, we evaluate the RAQ paradigm through user surveys, case studies, and scalability experiments.

6.1 Datasets

We consider a mix of various real (large) graphs (Table 1).

• The **IMDb** dataset [5] contains a collaboration network of movie artists, i.e., actors, directors, and producers. Each node is an artist featuring in a movie, and two artists are connected by an edge if they have featured together in at least one movie. Each node possesses 4 features: (i) year of birth, (iii) the set of all the movies the artist has featured in, (iv) the most prevalent genre among his/her movies, and (v) the median rating of the movie set.

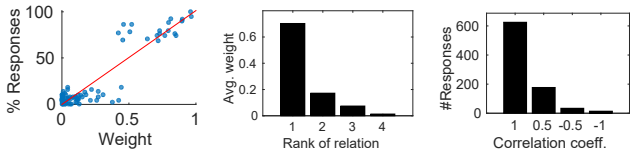
• The **DBLP** dataset [2] represents the citation network of research papers published in computer science. Each node is a paper and a directed edge from u to v denotes paper u being cited by paper v. Each node possesses 5 features: (i) publication venue, (ii) the set of authors, (iii) year of publication, (iv) the rank of the venue, and (v) the subject area of the venue. The publication year and venue rank are numerical features while the rest are categorical in nature. The first three features are obtained from the DBLP database while the last two are added from the CORE rankings portal [1].

• The **DBLP (co-author)** dataset [2] represents the co-authorship network of the research papers. Each node is an author of a paper and edges denote collaboration between two authors. Each node possesses 5 features: (i) number of papers published by the author, (ii) the number of years for which the author has been active (calculated as the difference between the latest and the first paper), (iii) the set of venues in which the author has published, (iv) the set of subject area(s) of the venues in which the author has published, and (v) the median rank of the venue (computed as the median of the ranks of all the venues where the author has published). The number of papers published, number of years active, and the median rank are numerical features, while the rest are categorical.

• The **Pokec** dataset [6] presents data collected from an online social network in Slovakia. Each node is a user of the social network and a directed edge $e = (u, v)$ denotes that u has a friend v (but not necessarily vice versa). Each node is characterized by six features: (i) gender, (ii) age, (iii) joining date, (iv) timestamp of last login (v) public: a boolean value indicating whether the friendship information of a user is public, and (vi) profile completion

Table 1: Summary of datasets used in our experiments.

Dataset	# Nodes	# Edges	Type	Mean Degree	# Features
IMDb	88K	781K	Undirected	17.59	7
DBLP	3.2M	6.8M	Directed	2.13	5
DBLP (co-author)	1.8M	7.4M	Undirected	8.44	5
Pokec	1.6M	30.6M	Directed	19.13	6



(a) Weight Correlation (b) Relation Weights (c) Rank Correlation

Figure 5: (a) Correlation between weights and user-perceived relationship importance. Variation in (b) average weights from most to least important relationship, (c) Correlation across user responses.

percentage. While public and gender are categorical features, the rest are numerical.

6.2 Experimental Setup

All experiments are performed using codes written in Java on an Intel(R) Xeon(R) E5-2609 8-core machine with 2.5 GHz CPU and 512 GB RAM running Linux Ubuntu 14.04.

Query graph: Query graphs are generated by selecting random subgraphs from the target graph. Following convention in graph querying [22, 25], we considered queries with size up to 10 edges. All reported results were averaged over 30 random query graphs.

Baselines: To measure the quality of graphs returned by RAQ, we compare its performance to three baselines:

• **RAQ-uniform:** To answer the question whether setting the weight of a relationship in proportion to its statistical significance helps, we implement RAQ-uniform as the setting where all relationships are considered equally important.

• **Graph edit distance (GED):** The *edit distance* between graphs q and g is the minimum *cost* of *edits* required to convert q to g [18]. An edit corresponds to deleting a node or an edge, adding a node or an edge, or substituting a node. Each edit has a cost. Deletions and additions of nodes or edges have a cost of 1. The cost of substituting a node u with v is proportional to the distance between them. We use the *min-max distance* between the feature vectors of two nodes to compute the distance. The replacement cost $rep(u, v)$ is:

$$rep(u, v) = 1 - sim(u, v)$$

$$sim(u, v) = \frac{1}{d}\sum_{i=1}^{d} s_i(u, v) \qquad \text{where}$$

$$s_i(u, v) = \begin{cases} \frac{\min\{f_i(u), f_i(v)\}}{\max\{f_i(u), f_i(v)\}} & \text{if} \quad f_i \text{ is real valued} \\ 1 \quad \text{if} \quad f_i \text{ is categorical and } f_i(u) = f_i(v) \\ 0 \quad \text{if} \quad f_i \text{ is categorical and } f_i(u) \neq f_i(v) \end{cases}$$

$rep(u, v)$ is always within the range $[0, 1]$.

• **Query by example (QBE):** We choose [26] to compare against the QBE paradigm (described in Sec. 2). To make it compatible to our framework, we model the node features as edge labels. Specifically, if u and v are connected in the original network, we introduce d edges between them. The i^{th} edge between u and v corresponds to the i^{th} relationship (feature), and this relationship is captured in the form of edge label by concatenating the feature values $f_i(u)$ and $f_i(v)$. All numerical features are discretized into bins [10] since otherwise, there would be infinitely many edge labels.

Parameters: The default value of k in a top-k query is set to 10. The branching factor in R-tree is $b = 4$.

6.3 User Survey: IMDb Dataset

To assess the relevance of the results retrieved using the proposed RAQ paradigm, we conducted two surveys spanning 170 users each, using collaboration patterns (subgraph queries) from the IMDb dataset. The user-base was carefully chosen to ensure *diversity* in various aspects such as profession (engineers, researchers, homemakers, doctors, business executives etc.), educational background (from undergraduates to doctorates in science, engineering, medicine etc.), gender (approximately equal distribution between females and males), and age (from 20 to 70 years old). Each user was presented with 5 queries randomly sampled from the master query pool, which led to a total of 850 user-responses for each user survey. To enable the users in understanding the tasks better, they were briefed on the capabilities of the RAQ framework and the semantic interpretation of a graph.

Relationship Importance: The first user survey studies whether statistical significance is consistent with human intuition. For this, we constructed a query pool of 20 edges (representing collaboration between two artists) along with their node features. For each query edge, we randomly choose two relationships (features), and the users were asked to identify the relatively more important relationship. A large number, 692/850 (81.4%), of the received user-responses considered the relationship with higher statistical significance to be more important. Considering the null hypothesis that either relationship is equally probable to be chosen as important, this observed outcome possesses a p-value of 1e−74 and is, therefore, highly statistically significant.

Moving ahead, we also analyze the correlation between the weight (normalized X^2 value) of a relationship and the proportion of times the feature is chosen as the preferred one in the survey. Fig. 5a shows that the higher the weight of a relationship, the more likely it is to be chosen as important in the user-responses. This is also indicated by their significantly high Pearson's correlation coefficient [30] of 0.923. Fig. 5b presents the average weight of the k^{th} ranked feature, where k is varied in the x-axis. We observe an exponential decay in weights, with the rank-1 relationship possessing a weight of 0.72 on average. This result, in conjunction with Fig. 5a, shows that the highest ranked relationship has a much higher likelihood of being perceived as important in the user-responses. Overall, this survey provides substantive evidence to support the use of statistical significance in quantifying relationship importance.

Ranking Results based on Relevance: The second survey evaluates the relevance of the results retrieved by RAQ and the three baselines discussed above. For this task, we considered a query pool consisting of 25 query subgraphs (representing group-level artist collaborations) possessing a judicious mix of node features (artist types, prominence, genres, etc.). For each query, the users were presented with top-1 results returned by the four algorithms being studied, and were asked to rank the results based on the user-perceived relevance to the query. To eliminate bias, the mapping of which result comes from which algorithm was masked from the user, and the order of the results presented were randomized. For ensuring simplicity and brevity, each query was annotated with

Table 2: Comparing the quality of RAQ with the considered baselines using both user-level and query-level responses.

Retrieved	User-level Responses (%)			Query-level Responses (%)		
Rank	RAQ	Uniform	GED	RAQ	Uniform	GED
1	**86.0**	10.6	3.4	**100.0**	0.0	0.0
2	10.0	**75.4**	14.6	0.0	**92.0**	8.0
3	4.0	14.0	**82.0**	0.0	8.0	**92.0**

the most important relationship, which was shown to correspond well with human perceived importance in the first survey.

Interestingly, across all 25 queries, QBE returned an empty answer set since it did not find any subgraph from IMDb, other than the query itself, which had nodes connected by the same relationships as in the query. This result is not surprising. As we discussed in Sec. 2, QBE operates in a binary world where two relationships are either identical or different. This result indicates that while QBE is an intuitive and powerful framework for relationship-annotated knowledge graphs, it is not well-suited for graphs where the relationships are encoded through interactions between feature vectors characterizing each node. Owing to this result, the user survey reduced to a comparison among three techniques.

Table 2 summarizes the user responses and in Fig. 4, we plot the distribution of the received responses within each retrieved rank for each query. An overwhelming 731 (86%) of the 850 received user responses deemed RAQ results to be the most relevant (rank-1 match) to the input query. Considering the null hypothesis where any one of the three results is equally likely to be chosen as the preferred one, this observed outcome is highly statistically significant (p-value of 2e−232). Moving beyond the analysis of rank-1 responses, we observe a high rank-correlation between the user-responses and the expected outcome, i.e., RAQ, RAQ-Uniform, and GED at ranks 1, 2, and 3 respectively, indicated by the average Spearman's coefficient $\rho = 0.803$ [39]. It is evident from Fig. 5c that $\approx 74\%$ of the user-responses perfectly match the expected output ranking with $\rho = 1$.

In addition to analyzing the user-level responses, we performed a query-level analysis as well. Specifically, for each query-rank combination, we identified the algorithm preferred by the highest number of users. As is clear from Table 2, RAQ results were considered to be the most relevant (rank-1) for all 25 (100%) queries.

6.4 Case Study: DBLP Dataset

In this section, we showcase results from the DBLP co-author dataset and discuss how RAQ is able to identify relationship patterns that node-based distance measures like GED are unable to. Each node in this experiment is characterized by three features, *number of papers published*, *median venue rank*, and *subject area*. The subject area corresponds to the area of the venue where the author has published most of her works.

The query presented in Fig. 6a represents a collaboration pattern comprising prolific authors working in the field of *data mining*. Fig. 6b presents the matched subgraph by RAQ, which is a collaboration pattern among prolific authors working in the field of *databases*. As can be seen, all authors in Fig. 6b possess the median venue rank as "A*" and the subject area as "DB". It is evident that RAQ captures the relationship that each group contains prolific authors from the same community. The result presented in Fig. 6b is not even within the top-10,000 matches produced by GED. This is natural since the two groups do not contain similar nodes.

Fig. 6c demonstrates RAQ's ability to adapt with the query. More specifically, while the subject area was a preserved relationship in query 1, query 2 presents collaborations among authors with high paper counts, but from diverse backgrounds. The result, Fig 6d, is a collaboration pattern among authors from diverse backgrounds as well, but all prolific with a large number of papers.

Correspondence between RAQ and edit distance: In this section, we show that RAQ retrieves results that node-based techniques are unable to find. We perform a top-5 query on random query graphs with sizes varying from 3 to 10. For each of the top-5 RAQ results, we find its rank using GED. We plot the average rank in GED for each of the query graph sizes (Fig. 7a). Since it is hard to scale beyond top-10,000 in GED, if a RAQ result does not appear within top-10,000, we set its rank to 10,000. The GED rank of the top-5 RAQ matches is close to 10,000 on average. This behavior is not a coincidence. Node-based similarity functions match graphs by considering each node as an independent entity and remains blind to the patterns found when the nodes are considered as a group. Consequently, they miss out on similarities that RAQ is able to discover.

6.5 Efficiency and Scalability

We next compare the running time of RAQ with RAQ-uniform and QBE [26] (Fig. 7b). We omit GED since for query graphs sizes beyond 5 it is impractically slow. RAQ is up to 5 times faster than QBE which suffers due to replication of each edge d times corresponding to d relationships. RAQ is faster than RAQ-uniform, since in RAQ, the relationships weights are skewed. Hence, pruning strategies are more effective than RAQ-uniform, where all relationships are equally important.

Fig. 7c demonstrates the growth of querying time against query graph size. As expected, the growth of is exponential since the search space grows exponentially with query graph size. Nonetheless, RAQ limits the running time to a maximum of only 5 s.

Notice that RAQ is fastest in Pokec even though Pokec is the largest network containing more than 30 million edges. The variance in feature values in Pokec is low. Consequently, the chance that a randomly picked subgraph is identical to the query is larger. Owing to this property, the search for the query graph converges quickly. Overall, these results establish that RAQ is a fast and practical search mechanism.

Next, we study the impact of target graph size on the querying time. For this experiment, we extract four different subgraphs of the entire network covering 25%, 50%, 75% and 100% of the nodes (and all edges between them) of the entire network (Figs. 7d, 7e) In DBLP, the querying time increases with increase in target graph size. In Pokec, we do not observe a similar trend. Since we employ best-first search, the search procedure stops as soon as the upper bound of an unexplored candidate is smaller than the k^{th} most similar subgraph we have already found. While an increase in target graph size increases the search space, it also enhances the chance of the k^{th} subgraph being more similar and therefore an earlier termination

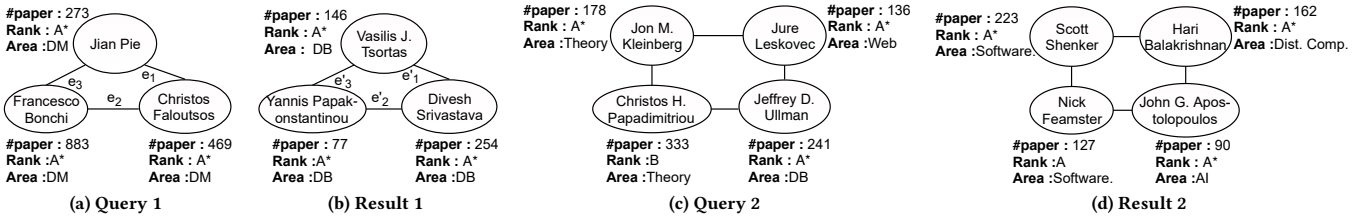


Figure 6: Matched subgraphs by RAQ.

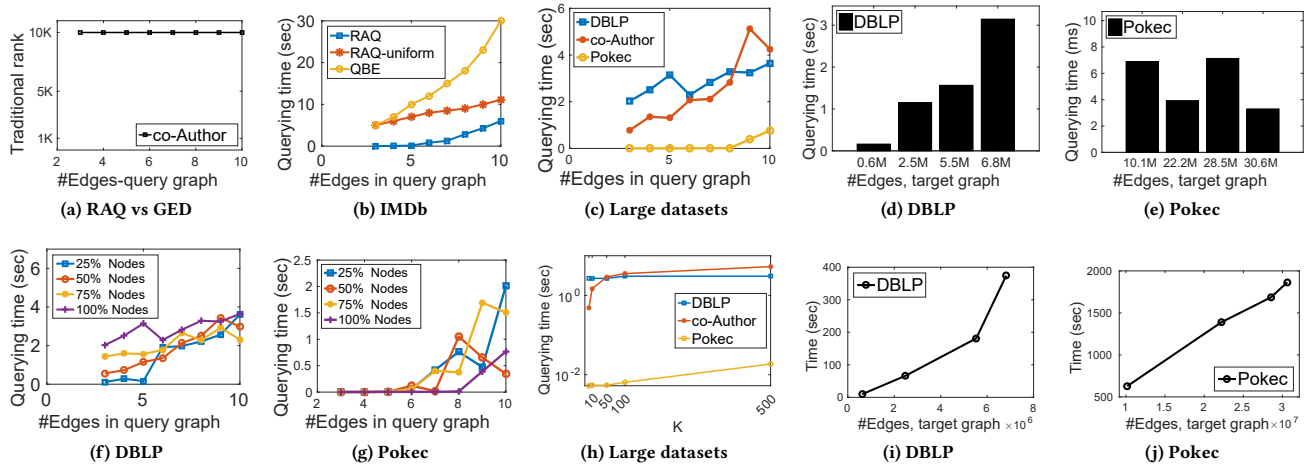


Figure 7: (a) Comparison of traditional graph search with the proposed relationship-aware similarity function. (b) Comparison of querying time between RAQ and the baseline technique. (c) Growth of querying time against query graph size and (d-e) target graph size. (f-g) Impact of target graph size on queries of various sizes. (h) Growth of querying time against k. (i-j) Growth of index construction time against network size.

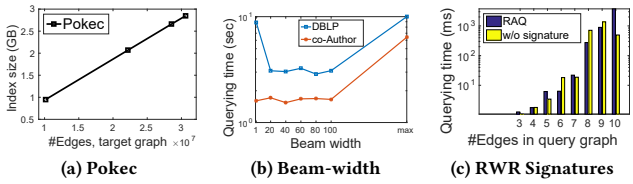


Figure 8: (a) Growth of RAQ's memory footprint against network size. Impact of (b) beam width and (c) neighborhood signatures on querying time.

of the best-first search. Due to this conflicting nature of our search operation, we do not see a consistent trend in the running time against the target graph size.

Figs. 7f and 7g study the impact of query graph size on the querying time. In general, we see any upward trend in the querying time as the size of the query graph increases.

Fig. 7h presents the variation of querying time across various values of k. We notice that the querying time flattens out very rapidly. This is due to the fact that the number of subgraphs explored till the answer set converges remains relatively same even for larger values of k.

Indexing Costs: Figs. 7i, 7j present the results of indexing costs on DBLP and Pokec. Even on Pokec, which contains more than 30 million edges, RAQ completes index construction within 33 minutes. RAQ index structures have a linear space complexity as reflected in Fig. 8a where we depict the memory footprint.

6.6 Optimization

The efficiency of the search algorithm RAQ is dependent on two major components: *neighborhood signatures* for identifying similar neighborhoods and *beam-stack search* to grow and prune initial candidates. In this section, we analyze their impact on RAQ.

Beam-width controls the number of simultaneous subgraphs we grow to form isomorphic matches to the query. We vary this parameter and plot the corresponding querying times in Fig. 8b. A beam width of 1 reduces to depth-first search, where an initial bad choice needs to be explored till the end before moving to a different one. A larger beam width allows other promising candidates explored earlier. A very large value, however, keeps switching too often among the candidates. Overall, a beam width between 40 and 60 provides the highest speed-up.

We study the importance of neighborhood signatures. We compare the time taken for top-k search using two techniques: RAQ with neighborhood prioritization through signatures and RAQ without neighborhood prioritization. Fig. 8c presents the results in the Pokec dataset. We observe that for smaller query graphs, RAQ without neighborhood optimization performs better. However, for larger query graphs neighborhood, typically, optimization helps. For smaller query graphs, the neighborhood of a query edge itself is very small and, thus, they lack much discriminative information. When the query size grows, the neighborhood of an edge is larger and the chances of getting stuck in a local maxima increases. Signatures allows us to escape these regions.

7 CONCLUSIONS

In this paper, we addressed the problem of graph querying under a novel *Relationship-Aware Querying (RAQ)* paradigm. RAQ captures the relationships present among nodes in the query graph and uses these relationships as an input to the similarity function. Majority of the existing techniques consider each node as an independent entity and, hence, remain blind to the patterns that exist among them when considered as a whole. RAQ not only mines these patterns in the form of relationships but also weighs their importance in proportion to their statistical significance. An overwhelming majority of 86% of the users surveyed preferred RAQ over existing metrics. To address the computational challenges posed by graph querying, we designed a flexible searching algorithm, which performs beam-stack search on an R-tree to prune the search space. Empirical studies on real-world graph datasets established that RAQ is scalable and fast. Overall, RAQ opens a new door in the graph querying literature by surfacing useful results that would otherwise remain hidden.

REFERENCES

[1] Core rankings portal. http://portal.core.edu.au/conf-ranks/.
[2] DBLP: Computer Science Bibliography. http://dblp.uni-trier.de/.
[3] Facebook Graph Search. https://en.wikipedia.org/wiki/Facebook_Graph_Search.
[4] Google Knowledge Graph. https://www.google.com/intl/es419/insidesearch/features/search/knowledge.html.
[5] IMDB: The Internet Movie Data Base. https://www.imdb.com/interfaces/.
[6] SNAP Datasets. https://snap.stanford.edu/data/.
[7] A. Arora, M. Sachan, and A. Bhattacharya. Mining Statistically Significant Connected Subgraphs in Vertex Labeled Graphs. In *SIGMOD*, pages 1003–1014, 2014.
[8] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *ISWC*, pages 722–735, 2007.
[9] A. Bhattacharya. *Fundamentals of Database Indexing and Searching*. CRC Press, 2014.
[10] M. Biba, F. Esposito, S. Ferilli, N. Di Mauro, and T. M. A. Basile. Unsupervised discretization using kernel density estimation. In *IJCAI*, pages 696–701, 2007.
[11] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250, 2008.
[12] H. Bunke and K. Shearer. A graph distance metric based on the maximal common subgraph. *Pattern recognition letters*, 19(3):255–259, 1998.
[13] V. Chaoji, S. Ranu, R. Rastogi, and R. Bhatt. Recommendations to boost content spread in social networks. In *WWW*, pages 529–538, 2012.
[14] S. Dutta, P. Nayek, and A. Bhattacharya. Neighbor-aware search for approximate labeled graph matching using the chi-square statistics. In *WWW*, pages 1281–1290, 2017.
[15] C. Faloutsos, D. Koutra, and J. T. Vogelstein. DELTACON: A principled massive-graph similarity function. In *SDM*, pages 162–170, 2013.
[16] M.-L. Fernández and G. Valiente. A graph distance metric combining maximum common subgraph and minimum common supergraph. *Pattern Recognition Letters*, 22(6):753–758, 2001.
[17] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD*, pages 47–57, 1984.
[18] H. He and A. K. Singh. Closure-tree: An index structure for graph queries. In *ICDE*, 2006.
[19] N. Jayaram, A. Khan, C. Li, X. Yan, and R. Elmasri. Querying knowledge graphs by example entity tuples. *TKDE*, 27(10):2797–2811, Oct 2015.
[20] G. K. Kanji. *100 Statistical Tests*. Sage, 2006.
[21] A. Khan, N. Li, X. Yan, Z. Guan, S. Chakraborty, and S. Tao. Neighborhood based fast graph search in large networks. In *SIGMOD*, pages 901–912, 2011.
[22] A. Khan, Y. Wu, C. C. Aggarwal, and X. Yan. Nema: Fast graph search with label similarity. *PVLDB*, 6(3):181–192, 2013.
[23] S. Kotsiantis and D. Kanellopoulos. Discretization techniques: A recent survey.
[24] M. Lissandrini, D. Mottin, T. Palpanas, and Y. Velegrakis. Multi-example search in rich information graphs. In *ICDE*, 2018.
[25] D. Mottin, M. Lissandrini, Y. Velegrakis, and T. Palpanas. Exemplar queries: Give me an example of what you need. *PVLDB*, 7(5):365–376, 2014.
[26] D. Mottin, M. Lissandrini, Y. Velegrakis, and T. Palpanas. Exemplar queries: a new way of searching. *VLDBJ*, 25(6):741–765, Dec 2016.
[27] D. Natarajan and S. Ranu. A scalable and generic framework to mine top-k representative subgraph patterns. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 370–379. IEEE, 2016.
[28] D. Natarajan and S. Ranu. Resling: a scalable and generic framework to mine top-k representative subgraph patterns. *Knowledge and Information Systems*, 54(1):123–149, 2018.
[29] G. Nikolentzos, P. Meladianos, and M. Vazirgiannis. Matching node embeddings for graph similarity. In *AAAI*, pages 2429–2435, 2017.
[30] K. Pearson. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58:240–242, 1895.
[31] S. Ranu, B. T. Calhoun, A. K. Singh, and S. J. Swamidass. Probabilistic substructure mining from small-molecule screens. *Molecular Informatics*, 30(9):809–815, 2011.
[32] S. Ranu, M. Hoang, and A. Singh. Answering top-k representative queries on graph databases. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1163–1174. ACM, 2014.
[33] S. Ranu and A. K. Singh. Graphsig: A scalable approach to mining significant subgraphs in large graph databases. In *2009 IEEE 25th International Conference on Data Engineering*, pages 844–855. IEEE, 2009.
[34] S. Ranu and A. K. Singh. Mining statistically significant molecular substructures for efficient molecular classification. *Journal of chemical information and modeling*, 49(11):2537–2550, 2009.
[35] S. Ranu and A. K. Singh. Indexing and mining topological patterns for drug discovery. In *Proceedings of the 15th International Conference on Extending Database Technology*, pages 562–565. ACM, 2012.
[36] T. R. C. Read and N. A. C. Cressie. *Goodness-of-fit Statistics for Discrete Multivariate Data*. Springer, 1988.
[37] M. Sachan and A. Bhattacharya. Mining statistically significant substrings using the chi-square statistic. In *PVLDB*, pages 1052–1063, 2012.
[38] R. Singh, J. Xu, and B. Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. In *PNAS*, pages 12763–12768, 2008.
[39] C. Spearman. The proof and measurement of association between two things. *The American journal of psychology*, 15(1):72–101, 1904.
[40] Y. Tian, R. C. Mceachin, C. Santos, D. J. States, and J. M. Patel. Saga: A subgraph matching tool for biological graphs. *Bioinf.*, 23(2):232–239, 2007.
[41] Y. Tian and J. M. Patel. Tale: A tool for approximate large graph matching. In *ICDE*, pages 963–972, 2008.
[42] H. Tong, C. Faloutsos, B. Gallagher, and T. Eliassi-Rad. Fast best-effort pattern matching in large attributed graphs. In *KDD*, pages 737–746, 2007.
[43] H. Tong, C. Faloutsos, and J.-Y. Pan. Fast random walk with restart and its applications. In *ICDM*, pages 613–622, 2006.
[44] N. Voskarides, E. Meij, M. Tsagkias, M. de Rijke, and W. Weerkamp. Learning to explain entity relationships in knowledge graphs. In *ACL*, 2015.
[45] X. Yan, H. Cheng, J. Han, and P. S. Yu. Mining significant graph patterns by leap search. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 433–444, 2008.
[46] W. Yu and X. Lin. Irwr: incremental random walk with restart. In *SIGIR*, pages 1017–1020, 2013.
[47] Z. Zeng, A. K. H. Tung, J. Wang, J. Feng, and L. Zhou. Comparing stars: On approximating graph edit distance. In *PVLDB*, pages 25–36, 2009.
[48] W. Zheng, L. Zou, X. Lian, D. Wang, and D. Zhao. Graph similarity search with edit distance constraint in large graph databases. In *CIKM*, pages 1595–1600, 2013.
[49] R. Zhou and E. A. Hansen. Beam-stack search: Integrating backtracking with beam search. In *ICAPS*, pages 90–98, 2005.

BOLT-K: Bootstrapping Ontology Learning via Transfer of Knowledge

Nikhita Vedula
The Ohio State University
vedula.5@osu.edu

Pranav Maneriker
The Ohio State University
maneriker.1@osu.edu

Srinivasan Parthasarathy
The Ohio State University
srini@cse.ohio-state.edu

ABSTRACT

Dynamically extracting and representing continually evolving knowledge entities is an essential scaffold for grounded intelligence and decision making. Creating knowledge schemas for newly emerging, unfamiliar, domain-specific ideas or events poses the following challenges: (i) detecting relevant, often previously unknown concepts associated with the new domain; and (ii) learning ontological, semantically accurate relationships among the new concepts, despite having severely limited annotated data. To this end, we propose a novel LSTM-based framework with attentive pooling, BOLT-K, to learn an ontology for a target subject or domain. We bootstrap our ontology learning approach by adapting and transferring knowledge from an existing, functionally related source domain. We also augment the inadequate labeled data available for the target domain with various strategies to minimize human expertise during model development and training. BOLT-K first employs semantic and graphical features to recognize the entity or concept pairs likely to be related to each other, and filters out spurious concept combinations. It is then jointly trained on knowledge from the target and source domains to learn relationships among the target concepts. The target concepts and their corresponding relationships are subsequently used to construct an ontology. We extensively evaluate our framework on several, real-world bio-medical and commercial product domain ontologies. We obtain significant improvements of 5-25% F1-score points over state-of-the-art baselines. We also examine the potential of BOLT-K in detecting the presence of novel kinds of relationships that were unseen during training.

ACM Reference Format:
Nikhita Vedula, Pranav Maneriker, and Srinivasan Parthasarathy. 2019. BOLT-K: Bootstrapping Ontology Learning via Transfer of Knowledge. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313511

1 INTRODUCTION

Ontologies, taxonomies or knowledge graphs represent an effective way of organizing massive amounts of real world information into a structured format. In particular, domain-specific ontologies are valuable resources that formally model the conceptual vocabulary of a given domain. Building accurate ontologies from trustworthy sources [36, 51, 65, 66, 69] with sufficient coverage of concepts and relationships among them is time-consuming and labor-intensive. A number of approaches to automate this process have been proposed. For instance, building ontologies based on statistical, linguistic and graphical features [45, 68, 76]; enriching existing ontologies with domain-specific information [63]; and embedding-based methods to complete knowledge graphs [2, 23, 31, 47, 61, 62, 70, 73, 79, 80]. Nevertheless, they only utilize textual corpora and other accompanying information from the specific domain under consideration. They do not leverage the abundant, hierarchically structured knowledge that might be available in functionally and/or semantically *similar* or *related* subjects or domains. This is especially valuable in fields ranging from epidemiology to crisis response, and from bio-medicine to commercial product catalogs, where emerging knowledge can be frequent and crucial. For instance, there is no semantically coherent ontology associated with the recently surfaced human disease of *Zika fever*, knowledge of which is evolving to-date. It will therefore be highly useful to take advantage of its connections to similar vector-borne diseases like *Dengue* or *Malaria*, for which well organized and annotated information from domain experts is available. Further, there are multiple challenges associated with building comprehensive ontologies for e-commerce product platforms [11, 29]. Numerous closely related product catalogs often have incomplete or incorrectly labeled attributes. Using annotations from semantically similar product listings (such as products from the same category) can help embellish existing listings with missing attributes, as well as detect errors in the labeled attributes. In our work, we *bootstrap* the task of learning ontologies for novel domains or subjects, by adapting and *transferring* existing knowledge from related domain ontologies. This also alleviates the requirement of human expertise to obtain sufficient labeled data on the newly emergent subject.

A crucial task in constructing ontologies is learning hierarchical relationships among multiple concepts from unstructured text. This task requires large amounts of annotated training data associated with ontological concepts and their corresponding relationships. This process is time-consuming, expensive, and necessitates a significant amount of expert knowledge to categorize associations in niche domains that a layperson is unlikely to know about. To resolve this issue, Mintz et al [42] heuristically aligned texts with knowledge graphs via distant supervision to automatically generate training examples. Lin et al [32] applied distant supervision with sentence-level selective attention, while Zeng et al [79] coupled it with multi-instance learning to learn relationships. However such efforts either do not address the problem of the highly imbalanced occurrence of different relationships [53], or require a sizable number of sentences connecting related entities. Another popular strategy to address the issue of insufficient labeled data is *data augmentation*. This technique artificially expands labeled training

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313511

sets by generating new data points or transforming the existing ones, such that the class label properties are preserved [22, 35, 52].

In this paper, we propose a framework *BOLT-K* – Bootstrapping Ontology Learning via Transfer of Knowledge. It uses a long short term memory (LSTM) neural network with attentive pooling, to learn an ontology hierarchy for a given *target* domain. We first obtain all the concepts that constitute the target ontology. We subsequently devise an approach based on semantic and topological attributes to identify the concept pairs likely to be connected by a relationship, and eliminate the remaining spurious combinations. To address the issue of limited relationship-labeled training data, we utilize publicly available textual corpora and ontological information from a functionally similar *source* domain. We also employ data augmentation techniques to generate additional training examples for the target domain. We train our model jointly on the target and source hierarchy information, by sharing the hidden feature representations and appropriate model parameters among them. Finally, we predict ontological relationships between the concepts of the target domain, to construct an ontology for it. We extensively evaluate our framework on several real-world datasets, highlighting the transferability of concepts across comparable subjects. We show that BOLT-K can significantly improve the quality of learned ontologies over state-of-the-art baselines, when bootstrapped with relevant knowledge from a similar subject or domain.

To summarize, the key contributions of our work are:

- We develop a flexible and generalizable framework, BOLT-K, to automatically learn ontologies for contemporary novel or emergent sub-domains of rapidly evolving fields such as bio-medicine, epidemiology, e-commerce and crisis response.
- We propose to transfer existing knowledge from functionally similar domains, and augment the insufficient labeled target training data. This significantly lessens the need for manual expertise during model development and training.
- We extensively evaluate BOLT-K on real-world datasets for various sub-domains within bio-medicine and product graphs. We also show BOLT-K's capability in detecting novel types of relationships that were unseen during training.

2 RELATED WORK

The following lines of research are related to our work: (i) augmentation of textual training data; (ii) learning ontological relationships; and (iii) transfer learning for natural language processing tasks.

Training data augmentation: This is a common and successful technique in the image processing literature [12, 52, 55], and is slowly gaining popularity in NLP applications [22, 35, 52]. A caveat though, is that supplementing text data via transformations, interpolations or affine perturbations is not as straightforward as performing these enhancements for images. Kafle et al [24] proposed two methods of data augmentation for visual question answering: (i) using semantic segmentation annotations with labels to synthesize certain kinds of questions; and (ii) training a stacked LSTM model to generate questions about images. A common technique of augmenting text is to replace words or phrases with their synonyms from a thesaurus [82], or with appropriate n-grams from a language model [52]. Another technique is to translate sentences into a second language, and then translate them back into the original language to obtain a slight variation of the original sentence [13, 54, 71]. Progress has also been made in developing generative models based on variational autoencoders for sentence generation [7, 21, 58]. In this work, we employ a combination of text substitutions for data augmentation, as specified in Section 4.2.

Relation Extraction: Deep learning models such as convolutional neural networks (CNNs), recurrent neural networks (RNNs) and graph-RNNs have been successful in automatically learning features for extracting semantic relations between a pair of entities [49, 57, 72, 79]. A number of approaches have also employed attention mechanisms in conjunction with RNNs for relation extraction and classification [32, 84]. Another way of solving this problem is by formulating it as a link prediction problem in knowledge graphs [2, 23, 31, 47, 61, 70, 73]. Since our method addresses relation extraction at the level of a sentence or a group of sentences, we find that a bidirectional LSTM with attention-based pooling works well for our purpose. Similar to our work, there have been efforts to extract relations with an insufficient number of labeled examples per relationship type. Yuan et al [78] formulated this as a one-shot classification problem and solve it using a convolutional siamese neural network. Levy et al [30] turned it into a reading comprehension problem by associating multiple textual questions with each relation type and learning answers for them.

Transfer learning: Zhang et al [81] have surveyed various cross-dataset transfer learning techniques. This includes the kind of knowledge transfer paradigm that our work addresses, i.e. minimizing the generalization error in the target domain with the help of training instances from two disjoint source and target domains. Transfer learning has been highly useful in low-resource domains such as bio-medicine [26, 40]. Ganin et al [17] introduced a representation learning approach for domain adaptation by adding a gradient reversal layer in a feed forward neural network. They trained their model on a document sentiment classification task using labeled data from a source domain and unlabeled data from a target domain. Long et al [34] presented another approach that learns an unsupervised residual function to adapt classifiers from a source domain to a target domain. Similar to our work, prior efforts addressed the sharing of structural parameters across multiple task domains [1]. Yang et al [74] developed a framework for three different types of transfer learning paradigms in hierarchical RNNs for a sequence tagging task; namely cross-domain, cross-application and cross-lingual transfer. Knowledge transfer has also been used to inform inter-related NLP tasks such as named entity recognition, part-of-speech tagging, chunking and word segmentation [9, 48].

3 PROBLEM FORMULATION

We propose to learn an ontology for a *target* domain for which labeled information is very limited, by transferring potentially useful semantic and ontological knowledge from a distinct but related *source* domain. We refer to each element of the ontologies associated with these domains as an *entity* or a *concept*.

The inputs to our problem are:

(1) A knowledge ontology S consisting of entities, categories and labeled relationships S_R among them, for a source domain

(2) A small number of concept pairs that will be part of the ontology T for the target domain. These have been labeled with their respective relationship type from a set of target relationship types T_R. Since the target is an emergent domain with little to no labeled information available, we only require at least one labeled concept pair per relation type in T_R. Note that though S and T are from related domains, there may or may not be relationship types common to both S and T. This is not a requirement for our method.

(3) Corpora of text documents whose sentences contain occurrences of the concepts in S and concepts associated with the target domain, which are to be part of T. These documents can either be expert-authored such as research papers, or can be from public resources such as news articles or encyclopedias. We utilize the text corpora associated with the target domain to extract the set of concepts to be inserted into T, as explained in Section 4.1.

We extract the sentences containing co-occurrences of the concept pairs linked by relationship types in S_R as part of training data from the source. However, it is unlikely for a whole lot of information to be available about the emergent, target domain concepts in the corpora. Thus as part of the target training data, we restrict BOLT-K to use at least one and at most five labeled instances per relationship type in T_R. These would contain co-occurrences of a small number of target concept pairs. We employ data augmentation techniques to enhance this limited amount of training data (Section 4.2). Our BOLT-K approach thus makes use of (i) the abundant labeled relationship information from S and; (ii) the minimal amount of labeled relation information from T, to learn relationships between various pairs of concepts for the target domain.

4 THE BOLT-K FRAMEWORK

In this section we describe our proposed BOLT-K approach (Figure 1). Among the set of concepts that are to be part of the target ontology T, we first identify the pairs of concepts that are likely to be related to each other in T, and filter out the remaining spurious pairs. Next, we augment the limited number of available training instances that are labeled with the target relationships from T_R. We then use this dataset along with the annotated information from S to learn the relations between the remaining target concepts in T. The identified concept pairs for T and the relationship types between them that have been predicted by our model can be used to construct a complete ontology for the target domain T.

We now describe each step of our BOLT-K framework in detail.

4.1 Filtering Unrelated Target Concept Pairs

This is the first step of our algorithmic pipeline. We acquire the entity and category concepts from the given source ontology structure S. We assume for the purpose of simplification, that a list of the concepts that need to be a part of the target ontology T has been provided to us by a domain expert. Alternatively, we can also use emerging entity extraction algorithms [10, 14] to locate and extract concepts from the target text document corpus. This corpus can consist of news reports, laboratory records or research articles authored by domain experts, related to the target topic. It is highly inefficient (quadratic complexity) and unnecessary to consider all

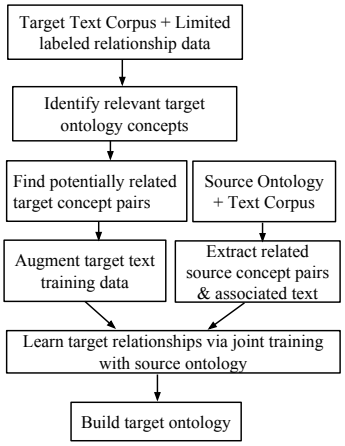


Figure 1: Pipeline of our BOLT-K framework

possible pairs of target concepts while learning relationships between them. Hence, our next step is to identify the concept pairs likely to be linked by a relationship and filter out the remaining pairs. Our empirical investigations (Figure 3) show that using only semantic information is insufficient to capture the likelihood of a relationship between pairs of target concepts. Hence, we also incorporate structural information by modeling the target concepts as a graph. We construct a weighted concept graph T_G, whose nodes consist of the concepts that are to be a part of the target ontology T. We link two nodes by an edge if they have co-occurred together at least once in a document in the target text corpus, within W words of each other. Inspecting a sample of documents in our corpora showed that related concepts often co-occur not in the same sentence, but in consecutive sentences. Since the length of an average sentence is about 10-12 words, we use $W = 25$ to indicate adjacent sentences. The weight of an edge is given by the pointwise mutual information (PMI) between the two concept nodes of the edge:

$$PMI(c_1, c_2) = log\Big(\frac{num(c_1, c_2)}{num(c_1) \cdot num(c_2)}\Big)$$

where $num(c)$ (or $num(c_1, c_2)$) is defined as the number of occurrences of a particular concept (or co-occurrences of a pair of concepts within W words of each other) in the target text corpus.

The PMI metric semantically gives us a good sense of the possibly related target concepts. Nevertheless, it yields a number of false positives which we seek to eliminate, by utilizing additional topological properties of the concept graph. We examine if the local neighborhood of a concept is a sufficient indicator of its potential relationships. However, we empirically find that local structural information is insufficient to account for concept relationships (Figure 3). Therefore, we employ a more global measure for this purpose that takes into account the overall concept graph topology, namely edge-based random walk betweenness centrality [46] (also called current-flow betweenness centrality [3]). For each edge in the weighted graph T_G, it measures approximately how often a node is traversed by a random walker going from any node in the network to another. If a concept c_1 appears often on random walks from concept c_2, then it is likely to be related to c_2. Finally, we learn a global threshold based on the betweenness centrality values, and

consider only those target concept pairs as potentially related if their edge betweenness centrality value is above the threshold.

We now have the set of source concept pairs that are related via labels in S_R to each other from the source ontology S. We have also obtained the set of target concept pairs that are likely to be related to each other and which will form the target ontology T. We now generate a training dataset for both the source and target domains, which will serve as input to the next step of BOLT-K's pipeline, i.e. learning the relationship between the pairs of target concepts. From the source and target document corpora, we extract sentences or blocks of sentences in which potentially related concepts co-occur within W words of each other. Henceforth, for ease of understanding, we call a group of W consecutive words a *sentence*, even though they may physically span more than one sentence of text. We mark the occurrences of the related concepts or entities in each sentence. As mentioned earlier, the target dataset may not have an abundant amount of labeled data available. To account for this, we limit BOLT-K to use at most 5 labeled concept pairs (and hence at most 5 sentences containing them) per target relationship type for training.

4.2 Data Augmentation for the Target Domain

A crucial requirement of building a predictive model is the availability of a sufficient amount of labeled training data for the relationship types of the target domain. Hence, to control generalization error and avoid overfitting to the limited training data available, we adopt the technique of *training data augmentation*. It artificially enhances labeled training datasets by transforming the available data items such that the class label properties are preserved. This lends a major advantage to our approach, i.e. the ability to transfer properties from one taxonomic hierarchy to another at no additional annotation cost. Some effective methods of performing text data augmentation with minimal human effort could be to create sentence paraphrases, or to substitute specific words or phrases with likely candidate words (e.g. synonyms). In our work, we augment the relationship-annotated target training sentences by replacing chosen words in them. We want to do this as diversely as possible, such that the syntactic and semantic equivalence between the original and altered sentence is maintained. Inspired by Ratner et al [52], we first identify the noun, verb and adjective terms in the target training sentences using the StanfordCoreNLP part-of-speech tagger [39]. We filter out the terms that do not occur above a learned frequency threshold. Out of these selected terms, we then iteratively sample a term occurring to the left, in between, and to the right of a pair of entities or concepts in each target sentence for substitution. We do not replace more than two terms in a single sentence at a time, to preserve the meaning and grammatical correctness of the modified sentence. We replace the chosen terms in the following three ways.

The first is by constructing an n-gram language model [52]. It is built by recording the frequencies of n-gram occurrences in the source and target corpora, filtering out the less frequent n-grams, and applying Laplace smoothing to the n-gram counts. This model samples words conditioned on the words preceding them. It identifies the n-gram n_x preceding the word or phrase x to be replaced. It then finds from the corpora a list of terms l_x, following n_x, sorted in descending order based on frequency. It finally replaces x with the term at index i in l_x. i is picked based on a geometric distribution $P[i] \sim p^i$, where a more frequent term has a higher probability of being chosen as a substitute [83]. The value of p is fixed at 0.5. The n-gram model falls back to using bigrams (or unigrams), in case the required trigram (or bigram) was filtered out.

For our second technique, we replace the chosen words in each sentence with one of their synonyms from their synset gloss in WordNet [15]. Since synonyms in a gloss are ranked according to how frequently they are observed in natural language, we use a similar geometric distribution as mentioned above to pick a synonym substitute. For our third substitution strategy, we use a pre-trained word2vec [41] word embedding model induced on texts from PubMed, PMC and the English Wikipedia [43]. After obtaining vector representations for each of the terms to be replaced, we substitute them with the terms most similar to their vector representations in the embedding space. We present a comparative evaluation of the three augmentation strategies in Table 3. Once we enhance the labeled dataset for the target domain, we use this data in combination with the labeled examples from the source domain to train the core model of BOLT-K, as described in Section 4.3.

4.3 BOLT-K Core Model

We now introduce the core model architecture of BOLT-K. It utilizes human-annotated knowledge on ontological concept relationships from a source domain, and minimally labeled and artificially augmented data from a functionally similar target domain, to learn a hierarchical ontology for the target domain. We build an LSTM-based model with attentive pooling to learn an ontology for the target concepts. We transfer ontological relationship knowledge from the source to the target domain in this model, by sharing the hidden layer representation and some of the model parameters between the two domains. We also combine the objective functions of both domains for effective training. Such models have been used in the literature for transfer learning in various applications [74, 81].

4.3.1 Base Model Components. Figure 2 presents an overview of the core model of our proposed BOLT-K framework. It consists of two parts: a *base* model architecture which is shared among both the source and target domains, followed by domain-specific neural network layers. We first describe each of the base model components in detail.

The input to our model is a set of sentences obtained from the source or target text corpora. Each sentence consists of n words $[x_1, x_2, ..., x_n]$. These words include a pair of ontology entities linked by a relation label. Recent literature (e.g. [16, 25, 64, 67, 77]) has seen immense success in transforming words into high-dimensional embeddings for diverse applications. The next layer of our model is thus an embedding layer, which represents every input word x_i as an embedding d_i. d_i is formed by concatenating the character-level representation of x_i with its word embedding from a pre-trained word2vec model [43]. This word2vec model has been trained on texts from PubMed, PMC and the English Wikipedia. Words which lack embeddings in the word2vec model are given a random representation. We obtain the character-level representation of each word using a CNN. CNNs have been shown to encode useful morphological information like word prefixes and suffixes from the characters of a word [8, 37, 56]. Our CNN model consists

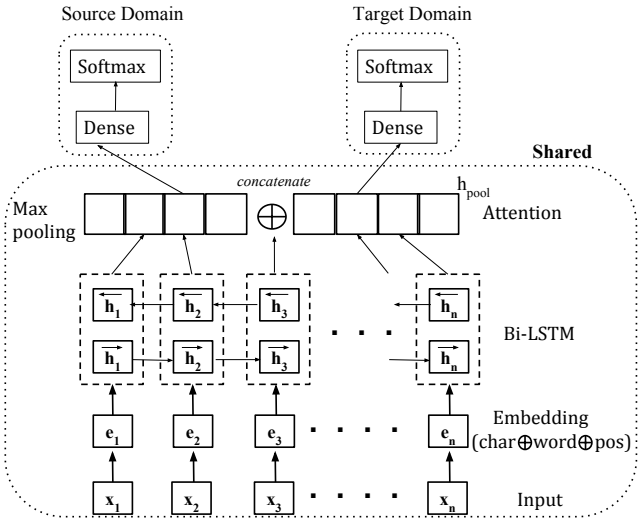


Figure 2: Proposed BOLT-K model architecture

of a convolution layer followed by dropout and max pooling. Its input is randomly initialized for each character of a given word x_i, and its output is a character-level representation of x_i. We form a final embedding e_i by appending two additional position indicators to each d_i. These are the normalized word-distances of word x_i from both the related concepts in their respective sentence. The embedding matrix $[e_1, e_2, ..., e_n]$ is updated during model training and serves as input to the next layer, i.e. the bidirectional LSTM.

Bidirectional LSTM [18] based models are used for a variety of sequence modeling tasks where it is often beneficial to utilize both the past and future context. These networks extend the traditional uni-directional LSTM [20] units that only consider past sequential information, by accounting for temporal context information from future time steps. At the core of the LSTM unit is a memory cell controlled by three sigmoidal gates: the input gate i_t deciding whether the unit retains its current input x_t or not, the forget gate f_t to enable the unit to forget its previous memory context c_{t-1}, and the output gate o_t controlling the context transferred to the hidden state h_t. The recurrences for the LSTM are defined as:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ c_t &= f_t * c_{t-1} + i_t * \tanh(W_c x_t + U_c h_{t-1} + b_c) \\ h_t &= o_t * \tanh(c_t) \end{aligned}$$

where σ is the sigmoid function, tanh is the hyperbolic tangent function and $*$ represents the product with the gate value. W, U and b are matrices of network parameters to be trained. As shown in Figure 2, the Bi-LSTM layer combines its forward ($\overrightarrow{h_t}$) and backward ($\overleftarrow{h_t}$) sequence contexts using the concatenation operation ($\oplus$). The output h_t of this layer at time step t is given by:

$$h_t = \overrightarrow{h_t} \oplus \overleftarrow{h_t}$$

The Bi-LSTM layer generates a sequence of word-level representations $[h_1, h_2, ..., h_n]$ utilizing past and future context, where $h_{j,t}$ denotes the j-th element of h_t. We perform one-dimensional max pooling to obtain a fixed length vector from the Bi-LSTM output.

$$m_{pool,j} = \max_{1 \le t \le n} [h_{j,t}]$$

The max pooling operation assumes that the important and relevant latent semantic features in the sentence are present at the positions [28] containing the maximum value of h_t. However, this might not always be the case. Hence, to focus on words that are crucial in predicting the relationship between entities which might lie anywhere in a sentence, and to give less importance to irrelevant information in the sentences, we utilize an *attention* mechanism. It highlights the important tokens in a given input sequence, which are responsible for performing feature selection for the model as well as for its predictions. Inspired by Yang et al [75], we introduce a word-level attention layer to capture the similarity of a word token with respect to its neighboring context tokens in an input sequence. It assigns weights to the hidden outputs h_t from the Bi-LSTM layer as follows:

$$\begin{aligned} z_{j,t} &= \tanh(W_z h_{j,t} + b_z) \\ \alpha_{j,t} &= \frac{exp(W_a^T z_{j,t})}{\sum_t exp(W_a^T z_{j,t})} \\ att_{pool,j} &= \textstyle\sum_t \alpha_{j,t} h_{j,t} \end{aligned}$$

Here, W_z and b_z are the weight matrix and bias vector respectively associated with the hidden state h_t from the Bi-LSTM layer, and $h_{j,t}$ represents the j-th sentence. Their non-linear transformation yields z_t, for which W_a is the corresponding weight matrix. $\alpha_{j,t}$ are the normalized attention weights representing token importances from a softmax function at time step t. $att_{pool,j}$ is the attention-focused hidden state representation, given by the linear combination of the Bi-LSTM output h_t and the attention weights.

The final output of the base model $h_{pool,j}$ is the concatenation ($\oplus$) of the outputs of the max pooling layer and the attention layer.

$$h_{pool,j} = m_{pool,j} \oplus att_{pool,j}$$

4.3.2 Knowledge Transfer Components. All the layers in the base model described in Section 4.3.1, namely the input, embedding, bidirectional LSTM, attention and pooling layers are shared and informed by training data from both the source and target domains. The output from the base model or the shared block in Figure 2 serves as input to two separate fully connected dense layers, followed by two softmax layers, one each for the source and target domains. The softmax function predicts the relationship type between a pair of concepts in an input sentence.

4.3.3 Training. We next outline how we transfer information from the source to the target domain by training our model jointly for both domains. We adopt a training procedure similar to that described by Yang et al. [74]. At each iteration, we sample one of the domains from the source and target based on a binomial distribution, where the binomial probability is a tuned parameter. We then optimize the objective function of the chosen domain by training on a sampled batch of labeled instances. The parameters of the shared block are thus updated due to training inputs from both domains, while the fully connected layers are only affected by their corresponding domain. We repeat this procedure until convergence, with early stopping based on the target domain performance.

We used 200-dimensional LSTM units with L2 regularization in our framework. We optimized the cross-entropy error between the true and predicted labels using Adam [27] with gradient clipping. We set the initial learning rate to 0.001 with a decay of 0.05. Dropout [60] was applied to the Bi-LSTM and pooling layers with a probability value of 0.5. Computational support for all our experiments was provided by the Ohio Supercomputer Center [5].

5 EVALUATION

5.1 Data Collection

We used the Open Biological and Biomedical Ontology (OBO) Foundry [59] and the National Center for Biomedical Ontology portal [44] which are collaborative repositories of science based ontologies, to obtain a family of ontologies related to the bio-medical domain. Each ontology has been curated with the important and relevant concepts associated with its specific sub-domain or subject, as well as the relationship types among the concepts. We also created and used commercial product ontologies from the open-source Web Data Commons project [50]. It contains structured data extracted from the web on different topics. Table 1 shows the statistics of various ontologies we have tested our BOLT-K framework on. These include for each ontology the number of concepts present in it, the number of related concept pairs, the total number of relationship types, and the median number of sentences available for a single relationship type. The first six rows are based on various human diseases and disorders. The next two rows show ontologies of flowering plants (*Angiosperms*) and non-flowering plants (*Gymnosperms*). The final three rows are based on three popular kinds of commercial products that are frequently manufactured, bought and sold, namely, *Earphones, Phones* (mobile phones) and *Television sets*. These ontologies are diverse and contain a total of less than 10K concepts, and less than 20 unique kinds of relationships.

We picked ontologies of sub-domains that are based on related subjects shown in Table 1, and used them interchangeably as both *source* and *target* ontologies to test our framework. For instance, *Dengue* and *Malaria* are both vector-borne diseases transmitted by mosquitoes and share some causes, symptoms and effects, so we use them as a source-target pair. Likewise, *Alzheimers, Multiple Sclerosis, Depression* and *Anxiety* are mental health disorders; *Gymnosperms* and *Angiosperms* are two classes of plant varieties; *Earphones* are an accessory of *Phones*; and *Phones* and *Televisions* are electronic devices sharing some common properties. Hence, we also used these as source-target ontology pairs to test BOLT-K.

As mentioned in Section 3, we require a corpus of text documents containing sentences associated with the concepts in the source and target ontologies S and T. To fulfill this requirement for the bio-medical sub-domains, we used a combination of PubMed [4], PubMed Central (PMC) and the English Wikipedia. For an ontology subject $\mathfrak{C}$, we consider all those articles from PubMed, PMC and Wikipedia as part of a text corpus associated with $\mathfrak{C}$ if they contain the term $\mathfrak{C}$ either in their title or abstract. For the last three rows of product hierarchies in Table 1, we constructed a text corpus from the product information and descriptions extracted from the Amazon Product Dataset [19]. This dataset includes information about the numerous commodities sold online on www.amazon.com.

Table 1: Statistics of various domain ontologies used

Ontology sub-domain name	No. of concepts	No. of concept pairs	No. of relations	Median no. of sentences per relation
Dengue	5035	5923	11	6010
Malaria	2643	3556	11	5070
Alzheimers	5738	5961	2	9602
Multiple sclerosis	9036	11310	2	16518
Depressive disorder	2008	4576	3	3025
Anxiety disorder	1978	4194	3	3637
Gymnosperms	539	502	9	716
Angiosperms	306	302	10	590
Earphones	115	146	11	28
Phones	189	337	16	256
Television sets	72	87	11	8

5.2 Results

5.2.1 Identifying Related Concept Pairs. We first present in Figure 3 the performance of BOLT-K as well as two other baselines on the first step of identifying potentially related concepts for the target ontology, as described in Section 4.1. We consider each ontology in Table 1 as a target ontology.

Our first baseline (yellow bars of *context similarity* in Figure 3) associates a context with each target concept. This is a set of nearby words around the mention of the concept in the target text corpus. If a target concept has multiple mentions (and hence multiple contexts) in the corpus, we pick the context associated with a randomly selected mention. We then compute an embedding for each concept using a normalized term-frequency (*tf*-) weighted sum of the embeddings of its context terms. We use a pre-trained word2vec model [43] to get the term embeddings, ignoring the context terms that do not have embeddings in this model. The *tf*- weights are obtained from the frequencies of occurrence of the chosen context terms, in the available contexts associated with every mention of the target concept. Once we generate an aggregated embedding of each concept, we compute the pairwise cosine similarity between all pairs of concept embeddings. We filter out all those concept pairs that do not have a similarity value above a learned threshold, and consider the remaining pairs as potentially related. Our second baseline (green bars of *2-hop jaccard* in Figure 3) estimates structurally similar concepts as possibly related to each other. It computes the 2-hop Jaccard index between pairs of connected concepts in the concept graph defined in Setion 4.1. For concepts c_1 and c_2, the Jaccard index is given by the number of neighbor concepts common to both c_1 and c_2 and reachable from them in 2 hops, divided by the union of the neighbors of c_1 and c_2. The pink bars of *RWBC* in Figure 3 denote the edge-based random walk betweenness centrality technique used in BOLT-K. The orange portions seen in the last three bars show an overlap between the result numbers of the context-based similarity and the 2-hop Jaccard strategies.

The RWBC measure considers a more global view of the concept graph along with semantic attributes. We find that this causes a marked improvement over merely using either semantic information (the context similarity baseline), or the local structural neighborhood of concepts in the weighted concept network (the Jaccard index baseline). Our random walk betweenness centrality measure

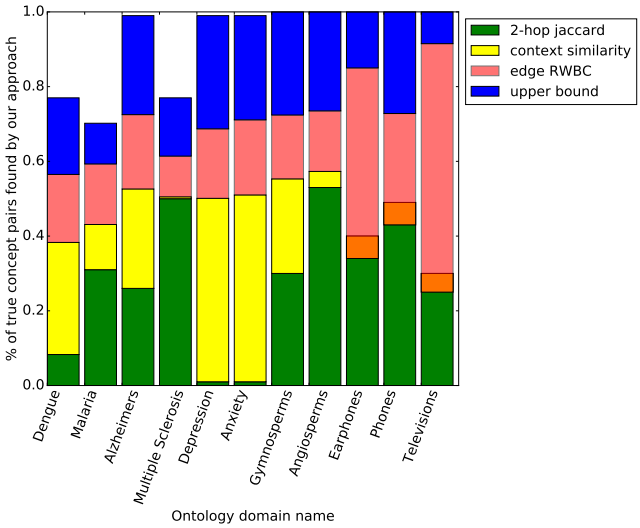


Figure 3: Performance of multiple approaches on identifying the concepts that are to constitute the target ontology

correctly identifies as *related*, about 60-75% of the linked concept pairs for the bio-medical ontologies. For the product hierarchies, it identifies more than 70% of the related concept pairs. We note here that every pair of related concepts that is present in the considered ontologies may not occur in the text corpora, i.e. may not be accompanied by textual information. There is thus a limitation on the number of potentially related concept pairs that can be found by our technique. The blue bars show this *upper bound* on the identifiable pairs of linked concepts for each ontology instance. This value is 76-80% for *Dengue*, *Malaria* and *Multiple Sclerosis*, and nearly 100% for the remaining hierarchies. Changing the scale of the result of BOLT-K's edge-based betweenness centrality strategy based on this upper bound, we observe an accuracy of more than 75% in finding potentially related concept pairs for the ontology instances.

5.2.2 Constructing Domain Ontologies via Knowledge Transfer. To the best of our knowledge, no existing work learns ontological relationships for a new target subject by levering prior knowledge from a distinct but related source subject. Thus, we compare our method with existing state-of-the-art techniques that only use information from the target domain. We evaluate all approaches on the task of ontological relationship prediction for the target ontology in Table 2. We reiterate that BOLT-K *does not* assume that the source and target ontologies have the same types of relationships. It uses at most 5 relationship-labeled concept pair training examples per target domain. We compare BOLT-K with the following baselines:

(1) PCNN-Att [79]: It employs a piecewise CNN with a sentence-level attention mechanism and distant supervision.
(2) Path-Max [80]: It considers CNNs coupled with probabilistic relation paths learnt from the sentences between entities.
(3) BLSTM-Att [84]: It uses a Bi-LSTM model without pooling, and with an attention mechanism different from ours.
(4) ComplEx [61]: It uses low-rank matrix factorization to learn complex-valued embeddings for entities and relations. We found ComplEx to outperform multiple other knowledge graph completion techniques like TransR [31], TransD [23], DistMult [73] and HolE [47], so we report only its results.
(5) BOLT-K ($t_{sr} = 0.6$) - no Att: This is a variant of our model at a target sub-domain data sampling rate $t_{sr} = 0.6$ (the probability of the model being trained on data from the target sub-domain), on using max pooling without attention.
(6) BOLT-K ($t_{sr} = t_{sr}$): The last five rows of Table 2 are variants of BOLT-K at target data sampling rates $t_{sr} = \{0, 0.4, 0.6, 0.8, 1\}$.

The first column of Table 2 shows the different approaches, and the subsequent columns show the ontology datasets. The $S \rightarrow T$ and $S \leftarrow T$ sub-columns for each column $S \leftrightarrow T$ indicate the direction of knowledge transfer, from ontology S to T and T to S respectively for our approach. The 'arrows' have no significance for the other approaches, merely indicating the ontology for which relations are being learnt (T in case of $S \rightarrow T$ and S in case of $S \leftarrow T$). These results have been computed after applying all augmentation strategies from Section 4.2 to the target training dataset.

We observe that BOLT-K obtains a 5-25% F1-score gain over the baselines in learning relationships for the different ontologies, with the best performance at a target training data sampling rate of 0.6. This reinforces our hypothesis of the utility of levering prior source knowledge in learning ontologies for a newly emergent target domain or sub-domain. BOLT-K is outperformed by ComplEx [61] by about 1% and 4% F1 score points respectively on the *Earphone* $\rightarrow$ *Phone* and the *Phone* $\rightarrow$ *Earphone* product ontology pairs. We believe this is because though these two products are somewhat related, their distinct characteristics may falsely inform our model and detract from its performance. BOLT-K performs better on the product pair of *Phones* and *Televisions* which share relatively more common attributes. It is interesting to observe the difference in performance on interchanging the source and target subject ontologies. For example, there is more value in transferring knowledge from the *Phone* to the *Television* domain, compared to the reverse.

We further note that the overall performance of all approaches on the bio-medical ontologies is significantly better than on the commercial product ontologies. A reason for this could be the quality of textual data available. The text descriptions are often quite generic and repetitive across different product relation types (e.g. *"details about apple iphone 6 16gb - at&t - gold - great condition"*). The variation and uniqueness in context and sentence structure is much lesser than the text data for the bio-medical ontology datasets that come from research, news or encyclopedia articles. Since ComplEx [61] utilizes less information from textual descriptions compared to the other methods, it performs better on the product datasets which have lower quality text input. We also analyze the topological structure of the concept graph (from Section 4.1) for the various ontology datasets. We find multiple well-separated connected components in case of the bio-medical ontologies, signifying a higher separability of their relationship categories. However, the product datasets have few (≤ 3) connected components for more than 10 relation categories. This implies that it is harder to distinguish between the product relationship types based on the current information.

5.2.3 Role of Target Training Data Size. Augmenting the limited amount of available target training data is a crucial step in our approach. Figure 4 (top) shows the change in F1-score of BOLT-K at the

Table 2: Baseline F1 scores on the ontology pairs of Dengue ↔ Malaria, Alzheimers ↔ Multiple Sclerosis, Gymnosperms ↔ Angiosperms, Earphones ↔ Phones, and Phones ↔ Televisions. We use at most 5 target training sentences per relation type with data augmentation. t_{sr} is the target sub-domain sampling probability. The first sub-column of every S ↔ T source-target ontology pair denotes knowledge transfer from S → T and the second denotes S ← T (i.e. knowledge transfer from T → S).

Approach	Deng ↔ Mal		Alz ↔ Mult Scl		Depr ↔ Anxi		Gymno ↔ Angio		Earph ↔ Phone		Phone ↔ TV	
	S→T	S←T	S→T	S←T	S→T	S←T	S→T	S←T	S→T	S←T	S→T	S←T
PCNN-Att	0.468	0.51	0.501	0.45	0.577	0.63	0.555	0.6	0.44	0.47	0.4	0.43
Path-Max	0.6	0.589	0.578	0.55	0.655	0.72	0.624	0.602	**0.501**	0.5	0.471	0.469
BLSTM-Att	0.601	0.62	0.505	0.49	0.567	0.6	0.592	0.63	0.43	0.45	0.41	0.399
ComplEx	0.561	0.626	0.54	0.5	0.66	0.701	0.58	0.54	0.48	**0.575**	0.514	0.455
BOLT-K (t_{sr}=0.6) - no Att	0.679	0.68	0.572	0.54	0.701	0.7	0.698	0.687	0.466	0.5	0.531	0.476
BOLT-K (t_{sr}=1)	0.66	0.64	0.502	0.53	0.582	0.68	0.631	0.67	0.47	0.5	0.5	0.47
BOLT-K (t_{sr}=0.8)	0.67	0.66	0.53	0.53	0.609	0.71	0.667	0.65	0.467	0.47	0.51	0.46
BOLT-K (t_{sr}=0.4)	0.644	0.63	0.545	0.5	0.65	0.66	0.63	0.62	0.44	0.45	0.46	0.44
BOLT-K (t_{sr}=0)	0.527	0.51	0.499	0.48	0.576	0.6	0.55	0.58	0.398	0.401	0.41	0.399
BOLT-K (t_{sr}=0.6)	**0.713**	**0.728**	**0.61**	**0.562**	**0.748**	**0.747**	**0.724**	**0.725**	0.487	0.53	**0.551**	**0.498**

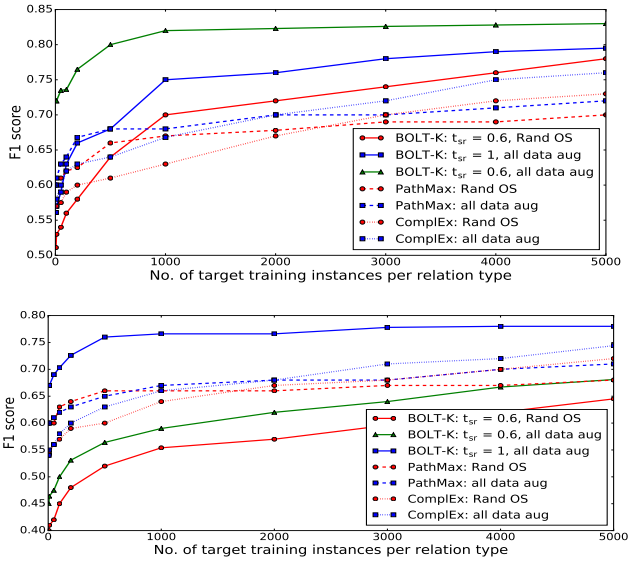


Figure 4: F1-score vs training data size for Dengue → Malaria (top) and Dengue → Angiosperms (bottom) ontology pairs.

target data sampling rates $t_{sr} = \{0.6, 1\}$ for the *Dengue → Malaria* ontology pair. We compare BOLT-K to the best performing baselines from Table 2 (*PathMax* [80] and *ComplEx* [61]), as the number of training instances per relation type are increased. We also show in Figure 4 (bottom) an attempt to learn an ontology for *Angiosperms* using information from a dissimilar and unrelated subject, *Dengue*. The solid, thick dashed and thin dashed line plots denote BOLT-K, PathMax and ComplEx respectively. The red plots use only random oversampling to augment the training data, while the blue and green plots use all strategies in Section 4.2. We observe that in case of *Dengue → Malaria* (top plot), as expected, the F1-score of each model rises with the amount of training data it receives. This gain is particularly significant at the left of the x-axis, from 5 to about 1000 training instances per relation category. Employing better data augmentation strategies than random oversampling contributes to the performance. Transferring knowledge from the source ontology lends us a consistent advantage of at least 5%, even at the maximum number of target training instances per relation type. In case of *Dengue → Angiosperms* (bottom plot), BOLT-K performs the best at $t_{sr} = 1$, i.e. without any input from the unrelated source (blue solid line plot). It obtains a 4% gain over ComplEx at the maximum number of training examples per relation class. However, training input from the dissimilar source *Dengue* causes a performance drop (red and green solid line plots).

We next examine in Table 3 the impact of different data augmentation strategies on the performance of BOLT-K, on the *Dengue → Malaria* ontology pair. We observe similar trends on the other datasets as well. The first two rows show the performance using simple random oversampling and Synthetic Minority Oversampling (SMOTE) [6] on the target training examples, followed by the three augmentation strategies described in Section 4.2. The last row of Table 3 shows that these three strategies complement one another. Employing them all together with random oversampling gives us a performance advantage of about 4-8% over any one of them.

5.2.4 Role of Attention. Table 2 indicates that attentive pooling lends BOLT-K an F1 score gain of nearly 5%. Further drilling down, Figure 5 shows a heatmap of the attention weight values learned by BOLT-K for sentences belonging to different relation types. The cells of the heatmap contain the sentence word they represent. The background color highlights the importance of a word with respect to its neighboring context, and consequently, how it affects our model's decision. The words in boldface are the concept phrases between which a relation is being predicted. For instance, the first row of the heatmap demonstrates that the words 'refers', 'to', and 'a' (dark background) are crucial in deciding the presence of an *is-a* relation between the concept phrases *virion assembly* and *process of dengue virus*. Similarly, in the fifth row, the words 'begins', 'after' and 'follows' influence the detection of the *preceded-by* relation between the concept phrases *incubation period* and *dengue disease course*. These examples show that BOLT-K has correctly learnt the relevant semantics needed to detect ontological relations between emergent concept phrases.

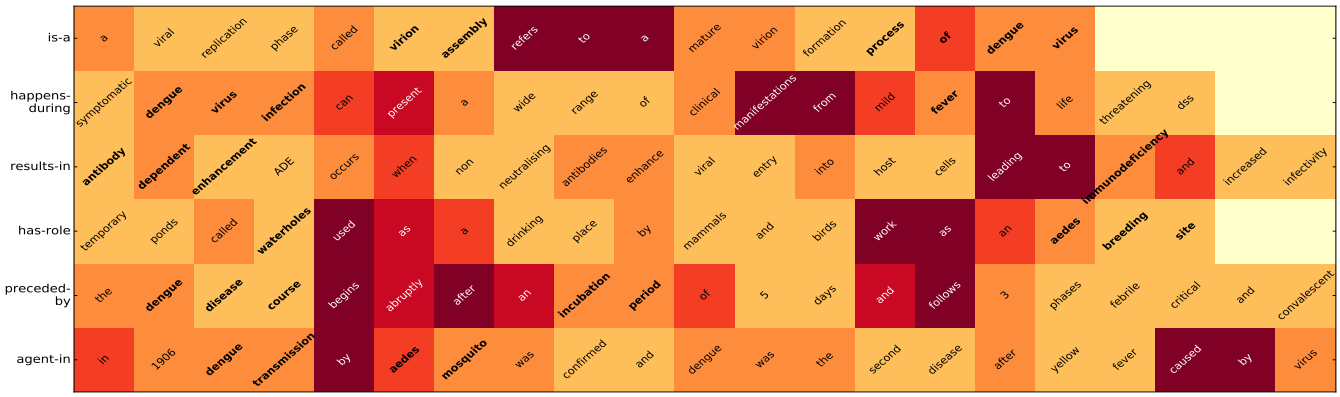


Figure 5: Visualizing word attention weights for sentences of certain relation types, for the Dengue ontology. The related concept phrase pairs in the sentence are in boldface, and a darker background corresponds to a higher attention weight.

Table 3: Assessing various training data augmentation methods for BOLT-K, on the Dengue → Malaria ontology pair

Augmentation Technique	Target F1 score
Random oversampling (Rand OS)	0.511
SMOTE oversampling	0.458
WordNet synonym based replacement + Rand OS	0.66
Word embedding based replacement + Rand OS	0.637
Trigram model based replacement + Rand OS	0.69
WordNet + Word embedding + Trigram + Rand OS	**0.713**

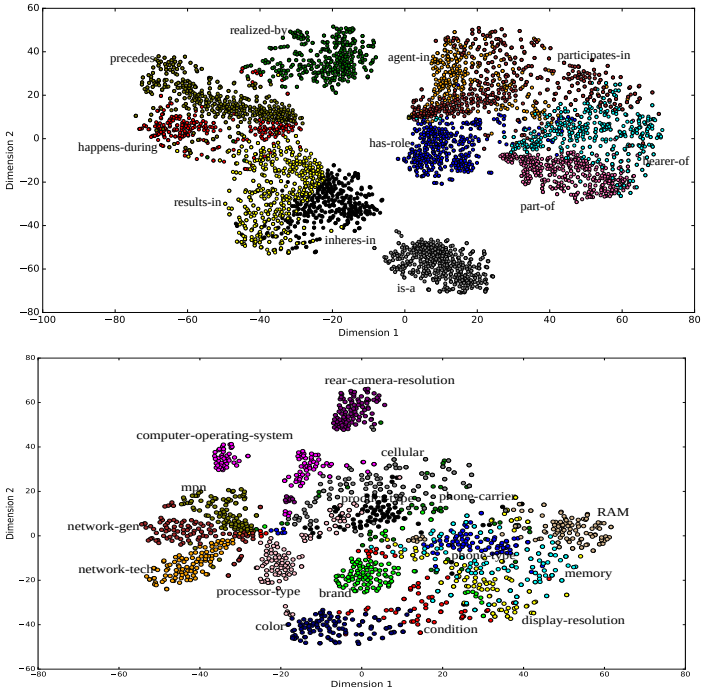


Figure 6: Visualizing learnt concept pair representations for Dengue → Malaria (top) and Televisions → Phones (bottom).

5.3 Discussion

We now dive deeper into the kind of ontological relationship information that our model can transfer across domains. Figure 6 displays the 2-dimensional t-SNE [38] visualizations of the representations learnt by BOLT-K for the sentence inputs of various relation classes. We observe that the relationships are largely well separable in case of the *Dengue → Malaria* ontology pair (Figure 6 top), with semantically similar relations co-located in their vector space. For example, the relations *participates-in, agent-in, has-role, part-of* and *bearer-of* are situated nearby, and away from the relatively dissimilar relations *results-in* and *inheres-in*. We do not see as clear a segregation between the relation categories for *Televisions → Phones* (Figure 6 bottom). But we do find some relations alike in meaning close by in the vector space too, e.g. *RAM* and *memory*, *network-gen* and *network-tech*.

Table 4 shows the relation categories dominantly mispredicted by our model and the relation type that they are frequently mispredicted as, for different ontology datasets. It also reports for each mispredicted relation category *r*, the percentage of test instances of category *r* that were mispredicted. We find that a common reason for mispredictions is the high similarity in the meaning of certain relationships, due to which they can be used interchangeably in natural language. For instance, BOLT-K is unable to distinguish between the relations *is-a* and *type-of* in case of *Depression ↔ Anxiety*, *participates-in* and *has-role* in case of *Dengue ↔ Malaria* and *preceded-by* and *develops-from* in case of *Gymnosperms ↔ Angiosperms*. We also observe this trend in some cases of the product ontology pairs (last four rows of Table 4), such as *RAM* and *memory*, *brand* and *phone-type*, and *total-size* and *viewable-size*.

As part of further analysis, we seek to understand how well our model can recognize hitherto unseen relation types. For this purpose, we remove or "leave out" one relationship class (i.e. all concept pairs and sentences associated with it) completely from the training dataset of the target sub-domain (and from the training data of the source sub-domain if present). We train our model with

Table 4: Frequently mispredicted relation types for a set of ontology pairs. For a relation type *r*, the parentheses show the percentage of mispredicted test instances of type *r*.

Source → Target	Mispred. rel type: Frequently mispred. as
Dengue → Malaria	*part-of* (35%): *is-a*; *inheres-in* (21%): *part-of*; *participates-in* (16%): *has-role*; *agent-in* (7%): *participates-in*; *bearer-of* (6%): *agent-in*
Malaria → Dengue	*part-of* (17%): *is-a*; *has-role* (17%): *part-of*; *happens-during* (8%): *part-of*; *precedes* (6%): *happens-during*
Alzheimers → Multiple Sclerosis	*type-of* (40%): *has-type*; *has-type* (38%): *type-of*
Multiple Sclerosis → Alzheimers	*type-of* (47%): *has-type*; *has-type* (41%): *type-of*
Depression → Anxiety	*is-a* (36%): *type-of*; *is-a* (12%): *has-type*
Anxiety → Depression	*is-a* (24%): *type-of*; *type-of* (18%): *has-type*
Gymnosperms → Angiosperms	*part-of* (19%): *is-a*; *adjacent-to* (20%): *part-of*; *has-part* (21%): *develops-from*; *located-in* (36%): *part-of*
Angiosperms → Gymnosperms	*preceded-by* (16%): *develops-from*; *develops-from* (11%): *part-of*; *has-participant* (11%): *part-of*; *part-of* (7%): *is-a*
Earphones → Phones	*brand* (42%): *phone-type*; *phone-type* (38%): *brand*; *display-res* (29%): color; *phone-carrier* (25%): *network-gen*; *RAM* (25%): *memory*
Phones → Earphones	*model* (44%): *product-type*; *brand* (30%): *model*; *color* (30%): *brand*; *additional-features* (22%): *compatibility*; *tagline* (20%): *brand*
Phones → Televisions	*refresh-rate* (39%): *display-type*; *model* (29%): *product-type*; *viewable-size* (29%): *total-size*; *display-res* (23%): *display-type*
Televisions → Phones	*RAM* (58%): *memory*; *cellular* (34%): *mpn*; *phone-type* (32%): *brand*; *display-res* (27%): *phone-type*; *cellular* (27%): *phone-carrier*

Table 5: Novel relation detection for Dengue → Malaria pair.

Relation type *r* left out during training	% of left out concept pairs detected as novel	Novel concept pairs missed. *s*(x%) means x% concept pairs related by *r* were mispredicted as having relation *s*.
is-a	76%	*part-of(8%), has-role(5%)*
part-of	51%	*has-role(19%), is-a(10%)*
happens-during	66%	*results-in(11%), part-of(6%)*
precedes	70%	*results-in(12%), happens-during(9%)*
results-in	61%	*precedes(14%), happens-during(9%)*
has-role	39%	*participates-in(21%), agent-in(17%)*
inheres-in	58%	*part-of(17%), bearer-of(10%)*
agent-in	41%	*participates-in(22%), bearer-of(19%)*
participates-in	46%	*has-role(23%), part-of(17%)*
bearer-of	64%	*agent-in(13%), part-of(9%)*
realized-by	67%	*results-in(9%), precedes(6%)*

the remaining training data as described in Section 4. We then use the Isolation Forest algorithm [33] with the authors' best case parameter settings to see if BOLT-K can detect concept pairs with the unseen relation type as *novel*, i.e. linked by a novel or unseen relationship. Isolation Forest takes as input the d-dimensional features obtained from BOLT-K's penultimate layer before the softmax layer. This novelty detection experiment shows our model's capability in identifying new or unseen relation categories which cannot be easily substituted by another, previously seen relation type.

The results are shown in Table 5, for the *Dengue* → *Malaria* ontology pair. The second column shows the fraction of concept pairs (whose relation class was left out while training) that have been detected as novel. For distinctive relationship types such as *is-a*, we find that more than 75% of the concept pairs which are linked by this relation in the ground truth are predicted as novel. But for left out relations which are not as semantically distinct from the other relationship types, BOLT-K is unable to satisfactorily identify that they were part of an unseen class. For instance, BOLT-K is unable to flag as novel more than 50% of the concept pairs linked by *has-role* and *participates-in*, when they were left out during training. For the concept pairs that were connected by the left-out relationship yet were not detected as novel, we investigated the relationship categories that BOLT-K was predicting them as. The third column of Table 5 shows the two dominant relation types that most concept pairs (linked by the left-out relationship) are mispredicted as belonging to. We found these predictions to be largely logical, when compared with the actual ground truth relationships. For example, 17% of the concept pairs linked by an *inheres-in* relation that was unseen during training were mispredicted as being connected by a *part-of* relation. 21% of the concept pairs connected by a *has-role* relation were mispredicted as having a *participates-in* relation. These mispredictions are semantically plausible due to the similarity in meaning of the misunderstood relationship groups.

6 CONCLUSION AND FUTURE WORK

We present BOLT-K, a Bi-LSTM framework with attentive pooling. It identifies concepts relevant to newly emerging subjects or events in various domains from their textual accounts, and automatically learns structured ontologies for them. We bootstrap this process by transferring knowledge from an existing, related sub-domain. We also leverage training data augmentation to accentuate the limited expert-labeled data available for these emergent sub-domains, at no further annotation cost. We extensively evaluate BOLT-K on real-life bio-medical and commercial product ontologies.

BOLT-K currently involves manual intervention to define functionally or logically related domains for knowledge transfer during ontology learning. In future, we plan to automate this task. Since the meaning and validity of concepts and their relationships may change over time, we seek to enrich BOLT-K by learning concept and relation representations that can effectively capture their evolving dynamics. This can enable the prediction of time instances at which concepts or relations are likely to appear, disappear or reappear with respect to an ontology snapshot. We also aim to improve BOLT-K's scalability so it can handle larger and deeper ontologies.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation grant EAR-1520870. The authors would also like to thank Xin Luna Dong for suggestions on product datasets. All content presented represents the opinion of the authors, and is not necessarily endorsed by their sponsors.

REFERENCES

[1] Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research* 6, Nov (2005), 1817–1853.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*. 2787–2795.

[3] Ulrik Brandes and Daniel Fleischer. 2005. Centrality measures based on current flow. In *Annual Symposium on Theoretical Aspects of Computer Science*. Springer, 533–544.

[4] Kathi Canese and Sarah Weis. 2013. PubMed: the bibliographic database. (2013).

[5] Ohio Supercomputer Center. 1987. Ohio Supercomputer Center. http://osc.edu/ark:/19495/f5s1ph73. (1987).

[6] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* 16 (2002), 321–357.

[7] Yu Chen and Mohammed J Zaki. 2017. Kate: K-competitive autoencoder for text. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 85–94.

[8] Jason PC Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional LSTM-CNNs. *arXiv preprint arXiv:1511.08308* (2015).

[9] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[10] Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*. 140–147.

[11] Xin Luna Dong. 2018. Challenges and innovations in building a product knowledge graph. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2869–2869.

[12] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. 2014. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in Neural Information Processing Systems*. 766–774.

[13] Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. *arXiv preprint arXiv:1705.00440* (2017).

[14] Michael Färber, Achim Rettinger, and Boulos El Asmar. 2016. On emerging entity detection. In *European Knowledge Acquisition Workshop*. Springer, 223–238.

[15] WordNet Fellbaum. 1998. An Electronic Lexical Database (Language, Speech, and Communication). (1998).

[16] Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning semantic hierarchies via word embeddings. In *ACL Vol 1 Long Papers*.

[17] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research* 17, 1 (2016), 2096–2030.

[18] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6645–6649.

[19] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 507–517.

[20] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[21] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. *arXiv preprint arXiv:1703.00955* (2017).

[22] Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. *arXiv preprint arXiv:1804.06059* (2018).

[23] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 687–696.

[24] Kushal Kafle, Mohammed Yousefhussien, and Christopher Kanan. 2017. Data augmentation for visual question answering. In *Proceedings of the 10th International Conference on Natural Language Generation*. 198–202.

[25] A. Karpathy and L. Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *IEEE CVPR*.

[26] J-D Kim, Tomoko Ohta, Yuka Tateisi, and JunâĂŹichi Tsujii. 2003. GENIA corpus: a semantically annotated corpus for bio-textmining. *Bioinformatics* 19, suppl_1 (2003), i180–i182.

[27] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[28] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification.. In *AAAI Conference on Artificial Intelligence*, Vol. 333. 2267–2273.

[29] Ig-hoon Lee, Suekyung Lee, Taehee Lee, Sang-goo Lee, Dongkyu Kim, Jonghoon Chun, Hyunja Lee, and Junho Shim. 2005. Practical issues for building a product ontology system. In *Proceedings of the International Workshop on Data Engineering Issues in E-Commerce*. IEEE, 16–25.

[30] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115* (2017).

[31] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion.. In *AAAI Conference on Artificial Intelligence*, Vol. 15. 2181–2187.

[32] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 2124–2133.

[33] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 413–422.

[34] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. 2016. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*. 136–144.

[35] Xinghua Lu, Bin Zheng, Atulya Velivelli, and ChengXiang Zhai. 2006. Enhancing text categorization with semantic-enriched representation and training data augmentation. *Journal of the American Medical Informatics Association* 13, 5 (2006), 526–535.

[36] Fenglong Ma, Yaliang Li, Qi Li, Minghui Qiu, Jing Gao, Shi Zhi, Lu Su, Bo Zhao, Heng Ji, and Jiawei Han. 2015. Faitcrowd: Fine grained truth discovery for crowdsourced data aggregation. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 745–754.

[37] Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354* (2016).

[38] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[39] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. 55–60.

[40] Suyu Mei, Wang Fei, and Shuigeng Zhou. 2011. Gene ontology based transfer learning for protein subcellular localization. *BMC bioinformatics* 12, 1 (2011), 44.

[41] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[42] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*. Association for Computational Linguistics, 1003–1011.

[43] SPFGH Moen and Tapio Salakoski2 Sophia Ananiadou. 2013. Distributional semantics resources for biomedical text processing. In *Proceedings of the 5th International Symposium on Languages in Biology and Medicine, Tokyo, Japan*. 39–43.

[44] Mark A Musen, Natalya F Noy, Nigam H Shah, Patricia L Whetzel, Christopher G Chute, Margaret-Anne Story, Barry Smith, and NCBO team. 2011. The national center for biomedical ontology. *Journal of the American Medical Informatics Association* 19, 2 (2011), 190–195.

[45] Roberto Navigli, Paola Velardi, and Stefano Faralli. 2011. A graph-based algorithm for inducing lexical taxonomies from scratch. In *IJCAI: International Joint Conference on Artificial Intelligence*.

[46] Mark EJ Newman. 2005. A measure of betweenness centrality based on random walks. *Social networks* (2005).

[47] Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, and others. 2016. Holographic Embeddings of Knowledge Graphs.. In *AAAI Conference on Artificial Intelligence*, Vol. 2. 3–2.

[48] Nanyun Peng and Mark Dredze. 2016. Improving named entity recognition for chinese social media with word segmentation representation learning. *arXiv preprint arXiv:1603.00786* (2016).

[49] Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-sentence n-ary relation extraction with graph lstms. *arXiv preprint arXiv:1708.03743* (2017).

[50] Petar Petrovski, Anna Primpeli, Robert Meusel, and Christian Bizer. 2016. The WDC gold standards for product feature extraction and product matching. In *International Conference on Electronic Commerce and Web Technologies*. Springer, 73–86.

[51] Kashyap Popat, Subhabrata Mukherjee, Jannik Strötgen, and Gerhard Weikum. 2017. Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 1003–1012.

[52] Alexander J Ratner, Henry Ehrenberg, Zeshan Hussain, Jared Dunnmon, and Christopher Ré. 2017. Learning to Compose Domain-Specific Transformations for Data Augmentation. In *Advances in Neural Information Processing Systems*. 3239–3249.

[53] Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 148–163.

[54] Itsumi Saito, Jun Suzuki, Kyosuke Nishida, Kugatsu Sadamitsu, Satoshi Kobashikawa, Ryo Masumura, Yuji Matsumoto, and Junji Tomita. 2017. Improving Neural Text Normalization with Data Augmentation at Character-and Morphological Levels. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 257–262.

[55] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. 2016. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Advances in Neural Information Processing Systems*. 1163–1171.

[56] Cicero D Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 1818–1826.

[57] Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. *arXiv preprint arXiv:1504.06580* (2015).

[58] Xiaoyu Shen, Hui Su, Yanran Li, Wenjie Li, Shuzi Niu, Yang Zhao, Akiko Aizawa, and Guoping Long. 2017. A conditional variational framework for dialog generation. *arXiv preprint arXiv:1705.00316* (2017).

[59] Barry Smith, Michael Ashburner, Cornelius Rosse, Jonathan Bard, William Bug, Werner Ceusters, Louis J Goldberg, Karen Eilbeck, Amelia Ireland, Christopher J Mungall, and others. 2007. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature biotechnology* 25, 11 (2007), 1251.

[60] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[61] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*. 2071–2080.

[62] Luu Tuan, Yi Tay, Siu Hui, and See Ng. 2016. Learning Term Embeddings for Taxonomic Relation Identification Using Dynamic Weighting Neural Network. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

[63] Nikhita Vedula, Patrick K Nicholson, Deepak Ajwani, Sourav Dutta, Alessandra Sala, and Srinivasan Parthasarathy. 2018. Enriching Taxonomies With Functional Domain Knowledge. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 745–754.

[64] Nikhita Vedula and Srinivasan Parthasarathy. 2017. Emotional and linguistic cues of depression from social media. In *Proceedings of the 2017 International Conference on Digital Health*. ACM, 127–136.

[65] Nikhita Vedula, Srinivasan Parthasarathy, and Valerie L Shalin. 2016. Predicting trust relations among users in a social network. On the roles of influence, cohesion and valence. *Proceedings of ACM SKGKDD WISDOM* (2016).

[66] Nikhita Vedula, Srinivasan Parthasarathy, and Valerie L Shalin. 2017. Predicting trust relations within a social network: A case study on emergency response. In *Proceedings of the 2017 ACM on Web Science Conference*. ACM, 53–62.

[67] Nikhita Vedula, Wei Sun, Hyunhwan Lee, Harsh Gupta, Mitsunori Ogihara, Joseph Johnson, Gang Ren, and Srinivasan Parthasarathy. 2017. Multimodal Content Analysis for Effective Advertisements on YouTube. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1123–1128.

[68] Paola Velardi, Stefano Faralli, and Roberto Navigli. 2013. Ontolearn reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics* 39, 3 (2013), 665–707.

[69] VG Vydiswaran, ChengXiang Zhai, and Dan Roth. 2011. Content-driven trust propagation framework. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 974–982.

[70] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes.. In *AAAI Conference on Artificial Intelligence*, Vol. 14. 1112–1119.

[71] John Wieting, Jonathan Mallinson, and Kevin Gimpel. 2017. Learning paraphrastic sentence embeddings from back-translated bitext. *arXiv preprint arXiv:1706.01847* (2017).

[72] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation Learning of Knowledge Graphs with Entity Descriptions.. In *AAAI Conference on Artificial Intelligence*. 2659–2665.

[73] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575* (2014).

[74] Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. *arXiv preprint arXiv:1703.06345* (2017).

[75] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[76] Jianxing Yu, Zheng-Jun Zha, Meng Wang, Kai Wang, and Tat-Seng Chua. 2011. Domain-assisted product aspect hierarchy generation: towards hierarchical organization of unstructured consumer reviews. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 140–150.

[77] Zheng Yu, Haixun Wang, Xuemin Lin, and Min Wang. 2015. Learning Term Embeddings for Hypernymy Identification.. In *IJCAI*.

[78] Jianbo Yuan, Han Guo, Zhiwei Jin, Hongxia Jin, Xianchao Zhang, and Jiebo Luo. 2017. One-shot learning for fine-grained relation extraction via convolutional siamese neural network. In *2017 IEEE International Conference on Big Data*. IEEE, 2194–2199.

[79] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1753–1762.

[80] Wenyuan Zeng, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2016. Incorporating relation paths in neural relation extraction. *arXiv preprint arXiv:1609.07479* (2016).

[81] Jing Zhang, Wanqing Li, and Philip Ogunbona. 2017. Transfer learning for cross-dataset recognition: a survey. *arXiv preprint arXiv:1705.04396* (2017).

[82] Xiang Zhang and Yann LeCun. 2015. Text understanding from scratch. *arXiv preprint arXiv:1502.01710* (2015).

[83] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*. 649–657.

[84] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 207–212.

Learning Resolution Parameters for Graph Clustering

Nate Veldt
Mathematics Department,
Purdue University
West Lafayette, Indiana
lveldt@purdue.edu

David F. Gleich
Computer Science Department,
Purdue University
West Lafayette, Indiana
dgleich@purdue.edu

Anthony Wirth
School of Computing and
Information Systems,
The University of Melbourne
Parkville, VIC, Australia
awirth@unimelb.edu.au

ABSTRACT

Finding clusters of well-connected nodes in a graph is an extensively studied problem in graph-based data analysis. Because of its many applications, a large number of distinct graph clustering objective functions and algorithms have already been proposed and analyzed. To aid practitioners in determining the best clustering approach to use in different applications, we present new techniques for automatically learning how to set clustering resolution parameters. These parameters control the size and structure of communities that are formed by optimizing a generalized objective function. We begin by formalizing the notion of a parameter fitness function, which measures how well a fixed input clustering approximately solves a generalized clustering objective for a specific resolution parameter value. Under reasonable assumptions, which suit two key graph clustering applications, such a parameter fitness function can be efficiently minimized using a bisection-like method, yielding a resolution parameter that fits well with the example clustering. We view our framework as a type of *single-shot* hyperparameter tuning, as we are able to learn a good resolution parameter with just a single example. Our general approach can be applied to learn resolution parameters for both local and global graph clustering objectives. We demonstrate its utility in several experiments on real-world data where it is helpful to learn resolution parameters from a given example clustering.

KEYWORDS

Graph clustering; community detection; resolution parameters

ACM Reference Format:
Nate Veldt, David F. Gleich, and Anthony Wirth. 2019. Learning Resolution Parameters for Graph Clustering. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313471

1 INTRODUCTION

Partitioning a collection of items into groups of similar items – that is, clustering – is a fundamental computational task. So commonly applied, there is a large and still-growing suite of objective functions, algorithms, and techniques for identifying *good* clusters. One powerful mathematical model for clustering is the graph, comprising nodes and (undirected) edges. For a broad overview of graph clustering, refer to any one of a number of surveys [11, 13, 28, 31]. Nearly all clustering approaches favor clusters with high internal edge density and a low external edge density. A related, but not identical, notion is that a good cluster is a set of nodes with a small cut (i.e., few edges leaving the set), and a nontrivial size (e.g., a large number of nodes, or many internal edges).

Although most clustering approaches follow these general principles, there are many different ways to formalize such goals mathematically. In practice, this array of objective functions yields a large variety of different output clusterings. Indeed, many existing (theoretical) approaches to graph clustering assume that the user knows *a priori* which objective function is appropriate for their context or job. The main design task, leading to a practical solution, is then to develop good algorithms that exactly, or approximately, optimize the objective. However, we propose that it is more natural to assume that the user starts with some a priori knowledge about the desired *structure* of clusters in a given application domain. More specifically, they can provide at least *one* example of what a *good* clustering should look like. The revised goal is to find an objective function whose optimization yields the desired type of output.

Our approach. In this article, we show how to bootstrap from this one quality clustering to learn the appropriate objective function, chosen from a parameterized family of objective functions. The efficiency of our technique relies on the clustering objective function being linear, with linear constraints, tuned by a single parameter β. Given this type of objective, we formalize the notion of a parameter fitness function, which relies on a fixed example clustering C_x of a network, and takes a parameter β as input. The fitness function computes the ratio between the objective score of C_x, when β is chosen as the input parameter, and a lower bound on the optimal clustering objective for that β. Specifically, we use a concave piecewise linear lower bound over a wide family of what we refer to as *relaxed clusterings*. Minimizing the fitness function produces a parameter (and a corresponding objective function) that C_x exactly or at least *nearly* optimizes. Thus the aim, which can be realized via a guided binary search, is to identify the parameter setting in which C_x most stands out. In the remainder of this introduction, we flesh out the context for our technique.

Parameters. There are multiple families of objective functions whose members are specified by a tunable resolution parameter. This parameter controls the size and structure of clusters that are formed by optimizing the objective. Key examples of such *generalized clustering objective functions* include the Hamiltonian objective studied by Reichardt and Bornholdt [29], the stability objective of Delvenne et al. [10], and a multi-resolution variant of the map equation [32].

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313471

In this manuscript we focus on a related clustering framework that we developed in previous work [37], based on correlation clustering [5]. This framework is named LambdaCC, after its resolution parameter λ, which implicitly controls both the internal edge density as well as the cut sparsity of nodes formed by minimizing the objective. Furthermore, LambdaCC generalizes several well-studied objectives such as modularity clustering, sparsest cut, normalized cut, and cluster deletion. All of these objectives can be viewed as special cases of the objective for appropriate settings of λ.

Global and local. The above objectives are specifically designed for global clustering, in which the goal is to find a multi-cluster partitioning of an input graph. Local clustering objectives relying on resolution parameters also exist; these focus on finding a single cluster in a localized region of a graph. Flow-based methods such as FlowImprove [4], LocalImprove [26], and SimpleLocal [35] fit into this category. These methods repeatedly solve minimum s-t cut problems for different values of a parameter α, in order to minimize a ratio-style objective related to a cluster quality measure called *conductance*. This α can be viewed as a resolution parameter that balances a trade-off between forming a cluster with a small cut, and forming a cluster with a large overlap with a seed set in the graph.

Given the unifying power and versatility of generalized clustering objective functions, the challenge of finding the right clustering technique for a specific application can often be reduced to finding an appropriate resolution parameter. However, very little work has addressed how to set these parameters in practice, in particular to capture the specific clustering structure present in a certain application domain. In the past, solving generalized objective functions for a range of resolution parameters has been used to detect hierarchical clusterings in a network [29], or as a way to identify *stable* clusterings, which consistently optimize the objective over a range of parameter values [10, 17, 32]. While both are important applications of resolution-based clustering, the ability to detect a specified type of clustering structure is important regardless of a clustering's stability or the hierarchical structure of a network. Finally, while tuning hyperparameters is a standard procedure in the broader machine learning literature, most existing approaches are not specifically designed for tuning graph clustering resolution parameters. Furthermore hyperparameter tuning techniques typically rely on performing cross validation over a large number of training examples. We are concerned with learning good resolution parameters from a *single* example clustering that represents a meaningful partitioning in a certain application domain.

Our Contributions. In this paper we develop an approach for learning how to set resolution parameters for both local and global graph clustering problems. Our results for global graph clustering rely heavily on the LambdaCC framework we developed in past work [37]. We begin by formally defining a parameter fitness function for a given clustering. We then prove that under reasonable assumptions on the input clustering and clustering objective function used, we can find the minimizer of such a fitness function to within arbitrary precision using a simple bisection-like method. Our approach can be viewed as a type of *single-shot* hyperparameter tuning, as we are able to learn an appropriate setting of a resolution parameter when given a single example clustering. We display the utility of our approach in several local and global graph clustering experiments. Our approach allows us to obtain improved community detection results on synthetic and real-world networks. We also show how our method can be used to measure the correlation between metadata attributes and community structure in social networks.

2 GRAPH CLUSTERING BACKGROUND

This section reviews the global LambdaCC [37] clustering objective and a local clustering objective that is based on regionally biased minimum cut computations [4, 26]. While there do exist many other objectives for local and global clustering, we focus on these two as they both rely crucially on resolution parameters.

Basic Notation. In this paper we consider unweighted and undirected graphs $G = (V, E)$, though many of the ideas can be extended to weighted graphs. Global graph clustering separates G into disjoint sets of nodes so that every node belongs to exactly one cluster. For local clustering, one is additionally given a set of reference or seed nodes $R \subset V$ and the objective is to find a good cluster that shares a nontrivial overlap with R. The degree of a node $i \in V$ is the number of edges incident to it; we denote this by d_i. The volume of a set $S \subseteq V$ is given by $\mathbf{vol}(S) = \sum_{i \in S} d_i$ and $\mathbf{cut}(S)$ measures how many edges cross from S to its complement set $\bar{S} = V \backslash S$. Further notation will be presented as needed in the paper.

2.1 Global Clustering with LambdaCC

The LambdaCC objective is a special case of correlation clustering (CC) [5], a framework for partitioning signed graphs. In correlation clustering, each pair of nodes (u, v) in a signed graph is associated with either a positive edge or a negative edge, as well as a nonnegative edge weight e_{uv} indicating the strength of the relationship between u and v. Given this input, the goal is to produce a clustering which minimizes the weight of *disagreements* or *mistakes*, which occur when positive edges are placed between clusters or negative edges are placed inside clusters.

The LambdaCC framework takes an unsigned graph $G = (V, E)$, a resolution parameter $\lambda \in (0, 1)$, and node weights w_u for each $u \in V$. It converts this input into a signed graph over which the correlation clustering objective can then be minimized. The signed graph $\tilde{G} = (V, E^+, E^-)$ is constructed as follows: for every $(u, v) \in E$, if $1 - \lambda w_u w_v \geq 0$, form a positive edge (u, v) in $\tilde{G}$, otherwise form a negative edge. In either case, the weight of this edge is $e_{uv} = |1 - \lambda w_u w_v|$. For every non-edge in the original graph ($(u, v) \notin E$), form a negative edge $(u, v) \in E^-$ in $\tilde{G}$ with weight $e_{uv} = \lambda w_u w_v$. The LambdaCC objective function then corresponds to the correlation clustering objective applied to $\tilde{G}$:

$$\min \sum_{(u,v) \in E^+} e_{uv}(1 - \delta_{uv}) + \sum_{(i,j) \in E^-} e_{uv}\delta_{uv}, \tag{1}$$

where δ_{uv} is a zero-one indicator function which encodes whether a clustering has placed nodes u, v together ($\delta_{uv} = 1$), or apart ($\delta_{uv} = 0$). There are two main choices for node weights: $w_u = 1$ for all $u \in V$ is the *standard* LambdaCC objective. For this simple case, we note that a node pair (u, v) which defines an edge in G will always correspond to a positive edge in G. In some applications it is useful to consider a *degree-weighted* version in which $w_u = d_u$. In this case, if $\lambda \leq 1/(d_{max}^2)$ then we can still guarantee that $E = E^+$.

However, for larger values of λ it may be possible that an edge in G gets mapped to a negative edge in $\tilde{G}$.

As a generalization of standard unweighted correlation clustering, LambdaCC is NP-hard, though many approximation algorithms and heuristics for correlation clustering have been developed in practice [2, 5, 8, 9]. In our previous work [37], we showed that a 3-approximation for standard LambdaCC can be obtained for any $\lambda \geq 1/2$ by rounding the following LP relaxation of objective (1):

$$\begin{array}{lll} \text{minimize} & \sum_{(u,v)\in E}(1-\lambda)x_{uv} + & \sum_{(u,v)\notin E}\lambda(1-x_{uv}) \\ \text{subject to} & x_{uv} \leq x_{uw} + x_{vw} & \text{for all } u,v,w \\ & 0 \leq x_{uv} \leq 1 & \text{for all } u < v. \end{array} \tag{2}$$

Furthermore, even when a priori approximations are not guaranteed, solving the LP relaxation can be a very useful way to obtain empirical lower bounds for the objective in polynomial time. In follow up work [15], we provided improved approximations for $\lambda < 1/2$ based on rounding the LP, but noted an $\Omega(\log n)$ integrality gap for some small value of λ.

Equivalence Results. LambdaCC generalizes and unifies a large number of other clustering approaches. When $\lambda = 1/(2|E|)$, the degree-weighted version is equivalent to the popular maximum modularity clustering objective [23, 24]. Standard LambdaCC interpolates between the sparsest cut objective for a graph-dependent small value of λ, and the cluster deletion problem when $\lambda > |E|/(1+|E|)$. Given its relationship to modularity, LambdaCC is known to also be related to the stochastic block model [25] and a multi-cluster normalized cut objective [42].

2.2 Local Clustering Objectives

We next consider a class of clustering objectives that share some similarities with (1), but are designed for finding a single local cluster in a specific region of a large graph. With the input graph $G = (V, E)$, we additionally identify a set of seed or *reference* nodes R around which we wish to form a good community. One common measure for the "goodness" of a cluster S is the conductance objective:

$$\phi(S) = \mathbf{cut}(S)/(\min\{\mathbf{vol}(S), \mathbf{vol}(\bar{S})\}), \tag{3}$$

which is small when S is connected very well internally but shares few edges with $\bar{S}$. A number of graph clustering algorithms have been designed to minimize local variants of (3). These substitute the denominator of (3) with a measure of the overlap between an output cluster S and the reference set R. One such objective is the following local conductance measure:

$$\phi_R(S) = \frac{\mathbf{cut}(S)}{\mathbf{vol}(R \cap S) - \varepsilon\mathbf{vol}(\bar{R} \cap S)}, \tag{4}$$

which is minimized over all sets S such that the denominator of $\phi_R(S)$ is positive. This objective includes a *locality* parameter ε that controls how much overlap there should be between the seed set and output cluster. For a general overview of this clustering paradigm and its relationship to spectral and random-walk based techniques, we refer the reader to the work of Fountoulakis et al. [14]. Specific algorithms which minimize variants of (4) include FlowImprove [4], which always uses parameter $\varepsilon = \mathbf{vol}(R)/\mathbf{vol}(\bar{R})$, and LocalImprove [26] and SimpleLocal [35], both of which choose larger values of ε in order to keep computations more local. In the extreme case where we consider $\varepsilon = \infty$, the problem reduces to finding the minimum conductance *subset* of a reference set R, which can be accomplished by the Minimum Quotient Improvement (MQI) algorithm of Lang and Rao [20].

Objective (4) can be efficiently minimized by repeatedly solving a minimum s-t cut problem on an auxiliary graph constructed from G, which introduces a sink node s attached to nodes in R, and a source node t attached to nodes in $\bar{R} = V \backslash R$. Edges are weighted with respect to the locality parameter ε and another parameter α. In order to detect whether there exists some set S with $\phi_R(S) \leq \alpha$, one can solve a local clustering objective corresponding to the minimum s-t cut objective on the auxiliary graph. We refer to this simply as the *local flow clustering objective*:

$$\min \; f_\alpha(S) = \mathbf{cut}(S) + \alpha\mathbf{vol}(R \cap \bar{S}) + \alpha\varepsilon\mathbf{vol}(\bar{R} \cap S). \tag{5}$$

If the set S minimizing f_α satisfies $f_\alpha(S) < \alpha\mathbf{vol}(R)$, then rearranging terms one can show that $\phi_R(S) < \alpha$. Thus, by performing binary search over α or repeatedly solving (5) for smaller and smaller α, one can minimize the local conductance measure (4).

Previous research has largely treated α as a temporary parameter used in one step of a larger algorithm seeking to minimize (4). Algorithms which minimize (4) do so by finding the smallest α such that the minimum of (5) is $\alpha\mathbf{vol}(R)$. We depart from this approach by instead treating α as a tunable resolution parameter for balancing two conflicting goals: finding clusters with a small cut, and finding clusters that have a large overlap with the seed set R. In the case where ε is treated as infinitely large and we are simply looking for subsets of a seed set R satisfying $\mathbf{vol}(R) \leq \mathbf{vol}(\bar{R})$, then in effect we are trying to solve the optimization problem:

$$\min \mathbf{cut}(S) - \alpha\mathbf{vol}(S) + \alpha\mathbf{vol}(R) \;\; \text{such that } S \subseteq R. \tag{6}$$

This goal is related to, but ultimately should be contrasted with, the goal of minimizing the ratio $\mathbf{cut}(S)/\mathbf{vol}(S)$. The objectives are similar in that they both tend to prefer sets with small cut and large volume. We argue that treating α as a tunable parameter is in fact more versatile than simply minimizing the ratio score. In multiple applications it may be useful to find clusters with small cut and large volume, but different applications may put a different weight on each aspect of the objective. We observe that ε also plays an important role in the size and structure of the output community when it is less than ∞. For simplicity, in this paper we can treat this as a fixed constant, and in our experimental section we simply focus on objective (6).

2.3 Parametric Linear Programs

Before moving on we provide key background on parametric linear programming which will be important in our theoretical results. A standard linear program is a problem of the form

$$\min_{\mathbf{x}} \; \mathbf{c}^T\mathbf{x} \text{ such that } \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{7}$$

where $\mathbf{c}, \mathbf{b}$ are vectors and $\mathbf{A}$ is a constraint matrix. A *parametric* linear program is a related problem of the form

$$\min_{\mathbf{x}} \; \mathbf{c}^T\mathbf{x} + \beta(\Delta\mathbf{c})^T\mathbf{x} \text{ such that } \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{8}$$

where $\Delta\mathbf{c}$ is another vector of the same length as $\mathbf{c}$ and β is a parameter controlling the difference between (7) and (8). We state a well-known result about the solutions of (8) for different β. This result is not new; it follows directly from Proposition 2.3b from [1].

Theorem 1. *Let $L(\beta)$ be the minimum of (8) for a fixed β. If we are given bounds a and b such that $L(\beta) \in \mathbb{R}$ for all $\beta \in [a, b]$, then L is a piecewise linear and concave function in β over this interval.*

Parametric LPs in Graph Clustering Applications. In our work it is significant to note that the linear programming relaxation of LambdaCC is a parametric linear program in λ. Furthermore, the local flow clustering objective can be cast as a parametric linear program in α, since this objective corresponds simply to a special case of the minimum s-t cut problem, which can be cast as an LP.

2.4 Related Work

Our work builds on previous results that introduced generalized objective functions with resolution parameters, including the Hamiltonian objective [29], clustering stability [10], a multiscale variant of the map equation [32], and the LambdaCC framework [37]. Recently Jeub et al. [17] introduced a technique for sampling values of a resolution parameter and applying hierarchical consensus clustering techniques. Our work on learning clustering resolution parameters differs from theirs in that we do not aim to provide hierarchical clusterings of a network. Instead we assume that there is a known fixed clustering, for which we wish to learn a single specific resolution parameter.

There exist many techniques for localized community detection based on seed set expansion. Among numerous others, these include spectral and random-walk based methods [3, 33], flow-based methods [4, 20, 26, 35], and other approaches which perform diffusions from a set of seed nodes and round embeddings via a sweep cut procedure [18, 39]. We build on these by interpreting hyperparameters associated with such methods as resolution parameters which can be learned to produce clusters of a certain type.

3 THEORETICAL RESULTS

The major theoretical contribution of our work is a new framework for learning clustering resolution parameters based on minimizing a parameter fitness function for a given example clustering. We present results for a generic clustering objective and fitness function, and later show how to apply our results to LambdaCC and local flow clustering.

3.1 Problem Formulation

Let $\mathcal{C}$ denote a set of valid clusterings for a graph $G = (V, E)$. We consider a generic clustering objective function $f_\beta : \mathcal{C} \to \mathbb{R}_{\geq 0}$ that depends on a resolution parameter β. The function takes as input a clustering $C \in \mathcal{C}$, and outputs a nonnegative clustering quality score for C. We assume that smaller values of f_β are better. We intentionally allow f_β to be very general in order to develop broadly applicable theory. For intuition, one can think of f_β as being the LambdaCC function (1) with $\beta = \lambda$. Alternatively, one can picture f_β to be the local flow objective (5) with $\beta = \alpha$ and with $\mathcal{C}$ representing the set of bipartitions, i.e. for any $C \in \mathcal{C}$, $C = \{S, \bar{S}\}$ for some set $S \subset V$.

Given some objective function f_β, a standard clustering paradigm is to assume that an appropriate value of β has already been chosen, and then the goal is to produce some clustering C that exactly or approximately minimizes f_β. In our work, we address an inverse question: given an example clustering C_x, how do we determine a parameter β such that C_x approximately minimizes f_β? Ideally we would like to solve the following problem:

Goal 1: Find $\beta > 0$ such that $f_\beta(C_x) \leq f_\beta(C)$ for all $C \in \mathcal{C}$. (9)

In practice, however, C_x may not exactly minimize a generic clustering objective for any choice of resolution parameter. Thus we relax this to a more general and useful goal:

Goal 2: Find the minimum $\Delta \geq 1$ such that for some $\beta > 0$
$$f_\beta(C_x) \leq \Delta f_\beta(C) \text{ for all } C \in \mathcal{C}. \quad (10)$$

This second goal is motivated by the study of approximation algorithms for clustering. In effect this asks: if we are given a certain clustering C_x, is C_x a good approximation to f_β for any choice of β? Note that this generalizes (9): if β can be chosen to satisfy Goal 1, then the same β will satisfy Goal 2 with $\Delta = 1$. Furthermore, it has the added advantage that, if solved, Goal 2 will produce a value Δ which communicates how well clusterings like C_x can be detected using variants of the objective function f_β. If Δ is near 1, it means that f_β is able to produce similar clusterings for a correct choice of β, whereas if Δ is very large this indicates that C_x will be difficult to find even for an optimal β, and thus a different approach will be necessary for detecting clusterings of this type.

Clustering Relaxations. While Goal 2 is a more reasonable target than Goal 1, it may still be a very challenging problem to solve when objective f_β is hard to optimize, e.g., if it is NP-hard. We thus consider one final relaxation that is slightly weaker than (10), but will be more feasible to work with. Let $\hat{\mathcal{C}}$ denote a superset of $\mathcal{C}$ which includes not only clusterings for G, but also some notion of a relaxed clustering, and let $g_\beta : \hat{\mathcal{C}} \to \mathbb{R}_{\geq 0}$ be an objective that assigns a score for every $C \in \hat{\mathcal{C}}$. Furthermore, assume g_β represents a lower bound function for f_β: $g_\beta(C) \leq f_\beta(C)$ for all β and all $C \in \mathcal{C}$. Our consideration of g_β is motivated by the fact that many NP-hard clustering objectives permit convex relaxations, which can be optimized in polynomial time over a larger set of relaxed clusterings that contain all valid clusterings of G as a subset. For example, the LambdaCC objective is NP-hard to optimize for every $\lambda \in (0, 1)$, but the linear programming relaxation for every λ can be solved in polynomial time, and is defined over relaxed clusterings in which pairs of nodes are assigned distances between 0 and 1. These relaxations can be rounded to produce good approximations to the original NP-hard objective [8, 9]. Since g_β is indeed easier to optimize than f_β, the following goal will be easier to approach but still provide strong guarantees for learning a good value of β:

Goal 3: Find the minimum $\Delta \geq 1$ such that for some $\beta > 0$
$$f_\beta(C_x) \leq \Delta g_\beta(C) \text{ for all } C \in \hat{\mathcal{C}}. \quad (11)$$

If we can solve (11), this still guarantees that C_x is a Δ-approximation to f_β for an appropriately chosen β. For problems where f_β is very challenging to optimize, but g_β is not, this will be a much more feasible approach. In the next section we will focus on developing theory for addressing Goal 3, though we note that in applying this theory we can still choose $g_\beta = f_\beta$ and therefore instead address the stronger Goal 2 whenever this is feasible. We will take this approach when applying our theory to the local flow objective.

3.2 Parameter Fitness Function

We now present a parameter fitness function whose minimization is equivalent to solving (11). Functions f_β and g_β take a clustering or relaxed clustering as input and output an objective score. However, we wish to view β as an input parameter and we treat an example clustering C_x as a fixed input. Thus for convenience we introduce new related functions:

$$F(\beta) = f_\beta(C_x) \tag{12}$$

$$G(\beta) = \min_{C \in \hat{C}} g_\beta(C) \tag{13}$$

The ratio of these two functions defines the *parameter fitness function* that we seek to minimize:

$$\mathcal{P}(\beta) = \frac{F(\beta)}{G(\beta)}. \tag{14}$$

Observe that this function is always greater than or equal to 1 since $G(\beta) \leq F(\beta)$ for any β. The minimizer of $\mathcal{P}$ is a resolution parameter β that minimizes the ratio between the clustering score of a fixed C_x and a lower bound on f_β. Thus, by minimizing (14) we achieve Goal 3 in (11) with $\Delta = \min_\beta \mathcal{P}(\beta)$.

In Section 2.3, we noted that the local flow clustering objective can be characterized as a parametric linear program, as can the LP relaxation of LambdaCC. Furthermore, for a fixed clustering, both objective functions can be viewed as a linear function in terms of their resolution parameter. Motivated by these facts, we present a theorem which characterizes the behavior of the parameter fitness function $\mathcal{P}$ under certain reasonable conditions on the functions F and G. In the subsequent section we will use this result to show that $\mathcal{P}$ can be minimized to within arbitrary precision using an efficient bisection-like method.

Theorem 2. *Assume $F(\beta) = a + b\beta$ for nonzero real numbers a and b. Let G be concave and piecewise linear in β, and assume $F(\beta) \geq G(\beta) \geq 0$ for all $\beta \in [\ell, r]$ where ℓ and r are nonnegative lower and upper (i.e. left and right) bounds for β. Then $\mathcal{P}$ satisfies the following two properties:*

(a) If $\beta^- < \beta < \beta^+$, then $\mathcal{P}(\beta)$ cannot be strictly greater than both $\mathcal{P}(\beta^-)$ and $\mathcal{P}(\beta^+)$.

(b) If $\mathcal{P}(\beta^-) = \mathcal{P}(\beta^+)$, then $\mathcal{P}$ achieves its minimum in $[\beta^-, \beta^+]$.

Proof. Note that for some $\gamma \in (0, 1)$, $\beta = (1-\gamma)\beta^+ + \gamma\beta^-$. By concavity of G and linearity of F, we know

$$\mathcal{P}(\beta) = \frac{F((1-\gamma)\beta^+ + \gamma\beta^-)}{G((1-\gamma)\beta^+ + \gamma\beta^-)} \leq \frac{(1-\gamma)F(\beta^+) + \gamma F(\beta^-)}{(1-\gamma)G(\beta^+) + \gamma G(\beta^-)}$$
$$\leq \max\left\{\frac{(1-\gamma)F(\beta^+)}{(1-\gamma)G(\beta^+)}, \frac{\gamma F(\beta^-)}{\gamma G(\beta^-)}\right\} = \max\left\{\mathcal{P}(\beta^+), \mathcal{P}(\beta^-)\right\},$$

which proves the first property. Now assume that $\mathcal{P}(\beta^-) = \mathcal{P}(\beta^+)$. Using property 1, we know as β increases from its lower to upper limit, $\mathcal{P}$ cannot increase and then decrease. Thus, either $\mathcal{P}$ attains its minimum on $[\beta^-, \beta^+]$, else $\mathcal{P}$ is a constant for all $\beta \in [\beta^-, \beta^+]$. If the latter is true, then for some $\beta \in [\beta^-, \beta^+]$ and some sufficiently small $\epsilon > 0$, G must be linear in the range $(\beta - \epsilon, \beta + \epsilon)$, since we know that G is piecewise linear. Therefore, $G(\beta) = c + d\beta$ and

$$\mathcal{P}(\beta) = (a + b\beta)/(c + d\beta) = constant \tag{15}$$

for $\beta \in (\beta - \epsilon, \beta + \epsilon)$ and for some $c, d \in \mathbb{R}$. This ratio of linear functions can only be a constant if $a = c = 0$, or $b = d = 0$, or if $a = c$ and $b = d$. Since we assumed a and b were nonzero, the last case must hold, and thus $\mathcal{P}(\beta) = 1$ for every $\beta \in [\beta^-, \beta^+]$, so the minimizer is obtained in this case, since $\mathcal{P}(\beta) \geq 1$ for all β. □

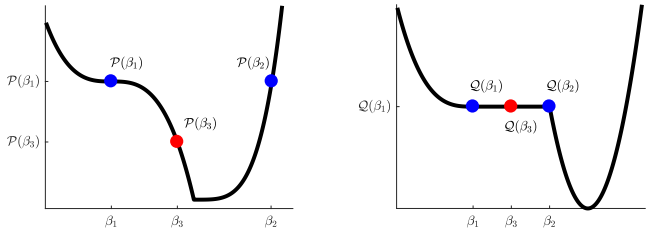


Figure 1: Function $\mathcal{P}$ satisfies both properties (a) and (b) in Theorem 2. If $\mathcal{P}(\beta_1) = \mathcal{P}(\beta_2)$, querying $\mathcal{P}$ at any point $\beta_3 \in [\beta_1, \beta_2]$ gets us closer to a minimizer. Function Q only satisfies property (a). If $Q(\beta_1) = Q(\beta_2)$, we can get stuck making queries inside a flat region of Q not near a minimizer.

In the next section we present a method for finding the minimizer of a function satisfying properties (a) and (b) in Theorem 2 to within arbitrary precision. Before doing so, we highlight the importance of ensuring that *both* properties hold. In Figure 1 we plot two toy functions, $\mathcal{P}$ and Q. Although both satisfy property (a), only $\mathcal{P}$ additionally satisfies (b). Assume we do not have explicit representations of either function, but we can query them at specific points to help find their minimizers. Consider Figure 1. If we query $\mathcal{P}$ at points β_1 and β_2 to find that $\mathcal{P}(\beta_1) = \mathcal{P}(\beta_2)$, then choosing any third point $\beta_3 \in (\beta_1, \beta_2)$ will get us closer to the minimizer. However, if $Q(\beta_1) = Q(\beta_2)$ for some β_1, β_2, we cannot be sure these points are not part of a flat region of Q somewhere far from the minimizer. It thus becomes unclear how to choose a third point β_3 at which to query Q. If we choose some $\beta_3 \in (\beta_1, \beta_2)$ and find that $Q(\beta_3) = Q(\beta_2) = Q(\beta_1)$, the minimizer may be within $[\beta_1, \beta_2]$, within $[\beta_2, \beta_3]$, or in a completely different region. Thus it is important for the denominator of a parameter fitness function to be piecewise linear in addition to being concave, since this piecewise linear assumption guarantees property (b) will hold.

3.3 Minimizing $\mathcal{P}$

We now outline an approach for finding a minimizer of $\mathcal{P}$ to within arbitrary precision when Theorem 2 holds. Our approach is closely related to the standard bisection method for finding zeros of a continuous function f. Recall that standard bisection starts with a and b such that $sign(f(a)) \neq sign(f(b))$, and then computes $f(c)$ where $c = (a + b)/2$. Checking the sign of $f(c)$ allows one to determine whether the zero of f is located within the interval $[a, c]$ or $[b, c]$. Thus each new query of the function f halves the interval in which a zero must be located.

Assume $\mathcal{P}$ satisfies properties (a) and (b) in Theorem 2 over an interval $[\ell, r]$. To satisfy Goal 3, given in (11) in Section 3.1, it suffices to find any minimizer of $\mathcal{P}$, which we do by repeatedly halving the interval in which the minimizers of $\mathcal{P}$ must lie. Our approach differs from standard bisection in that we are trying to find a *minimizer* instead of the zero of some function. The key algorithmic difference is that querying $\mathcal{P}$ at a single point between two bounds will not always be sufficient to cut the search space in

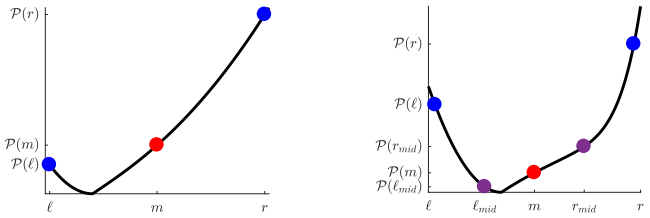


Figure 2: We evaluate $\mathcal{P}$ at left and right bounds (blue points), and at a midpoint m (red point). Left: If $\mathcal{P}(\ell) < \mathcal{P}(m) < \mathcal{P}(r)$, then we know the minimizer of $\mathcal{P}$ is in $[\ell, m]$, and we recursively call the *one-branch* phase (Algorithm 1) with new bounds ℓ and m. Right: If $\mathcal{P}(m) < \mathcal{P}(\ell) < \mathcal{P}(r)$, we don't know if the minimizer is in the left branch $[\ell, m]$ or right branch $[m, r]$. Evaluating $\mathcal{P}$ at the midpoint of each branch (purple points), we rule out branch $[m, r]$ and recursively call Algorithm 2 with new endpoints ℓ and m and midpoint ℓ_{mid}.

half. Consider Figure 2. Our method starts in a *one-branch* phase in which we know a minimizer lies between ℓ and r. If we compute $m = (\ell + r)/2$ and find that $\mathcal{P}(m)$ is between $\mathcal{P}(\ell)$ and $\mathcal{P}(r)$, this does in fact automatically cut our search space in half, as this implies that $\mathcal{P}$ is monotonic on either $[\ell, m]$ or $[m, r]$. However, if $\mathcal{P}(m) < \min\{\mathcal{P}(\ell), \mathcal{P}(r)\}$, then it is possible for the minimizer to reside within either the left branch $[\ell, m]$ or the right branch $[m, r]$. In this case, the method enters a *two-branch* phase in which it takes the midpoint of each branch ($\ell_{mid} = (\ell+m)/2$ and $r_{mid} = (m+r)/2$) and evaluates $\mathcal{P}(\ell_{mid})$ and $\mathcal{P}(r_{mid})$. If $\mathcal{P}$ returns the same value for two of the inputs (e.g., $\mathcal{P}(\ell) = \mathcal{P}(m)$), then by property (b) we have found a new interval containing the minimizer(s) of $\mathcal{P}$ that is at most half the length of $[\ell, r]$. Otherwise, we can use property (a) to deduce that the minimizer will be located within $[\ell, m]$, $[m, r]$, or $[\ell_{mid}, r_{mid}]$, and we recurse on the two-branch phase.

Algorithms 1 and 2 handle the one- and two-branch phases of the method respectively. The guarantees of our method are summarized in Theorem 3. We omit the full proof, since it follows directly from considering different simple cases and applying properties of $\mathcal{P}$ to halve the search space as outline above.

Theorem 3. *Consider a fixed clustering C_x and a corresponding parameter fitness function $\mathcal{P}_{C_x}$ satisfying the assumptions of Theorem 2. Running Algorithm 1 with input ℓ, r and a tolerance ϵ will produce a resolution parameter $\tilde{\beta}$ that is within ϵ of the minimizer of $\mathcal{P}_X$ over the interval $[\ell, r]$, in at most $\log_2((r-\ell)/\epsilon)$ recursive calls.*

4 APPLICATION TO SPECIFIC OBJECTIVES

Theorem 2 and our approach for minimizing $\mathcal{P}$ can be immediately applied to learn resolution parameters for the LambdaCC global clustering objective and the local flow clustering objective.

4.1 Local Clustering

For local clustering we consider the objective function f_α given in (5) and note that the set of valid clusterings C is the set of bipartitions. The example clustering we are given at the outset of the problem is $C_x = \{X, \bar{X}\}$ where $X \subset V$ is some nontrivial set of nodes representing a "good" cluster for a given application. We assume we

Algorithm 1 CheckOneBranch(ℓ, r, ϵ)

Base case:
if $r - \ell < \epsilon$ **then**
  return ℓ
Recursive call:
Midpoint: $m = (\ell + r)/2$
switch ℓ, m, r **do**
  case $\mathcal{P}(\ell) = \mathcal{P}(m) = \mathcal{P}(r)$
    return m
  case $\mathcal{P}(\ell) \leq \mathcal{P}(m) < \mathcal{P}(r)$
    return CheckOneBranch(ℓ, m, ϵ)
  case $\mathcal{P}(\ell) > \mathcal{P}(m) \geq \mathcal{P}(r)$
    return CheckOneBranch(m, r, ϵ)
  case $\mathcal{P}(\ell) > \mathcal{P}(m) < \mathcal{P}(r)$
    return CheckTwoBranches(ℓ, m, r, ϵ)

Algorithm 2 CheckTwoBranches(ℓ, m, r, ϵ)

Base case:
if $r - \ell < \epsilon$ **then**
  return m
Recursive call:
Left midpoint: $\ell_{mid} = (\ell + m)/2$
Right midpoint: $r_{mid} = (m + r)/2$
switch ℓ_{mid}, m, r_{mid} **do**
  case $\mathcal{P}(\ell_{mid}) = \mathcal{P}(m) = \mathcal{P}(r_{mid})$
    return m
  case $\mathcal{P}(\ell_{mid}) = \mathcal{P}(m) \neq \mathcal{P}(r_{mid})$
    return CheckOneBranch$(\ell_{mid}, m, \epsilon)$
  case $\mathcal{P}(\ell_{mid}) \neq \mathcal{P}(m) = \mathcal{P}(r_{mid})$
    return CheckOneBranch(m, r_{mid}, ϵ)
  case $\mathcal{P}(\ell_{mid}) < \mathcal{P}(m) < \mathcal{P}(r_{mid})$
    return CheckTwoBranches$(\ell, \ell_{mid}, m, \epsilon)$
  case $\mathcal{P}(\ell_{mid}) > \mathcal{P}(m) > \mathcal{P}(r_{mid})$
    return CheckTwoBranches$(m, r_{mid}, r, \epsilon)$
  case $\mathcal{P}(\ell_{mid}) > \mathcal{P}(m) < \mathcal{P}(r_{mid})$
    return CheckTwoBranches$(\ell_{mid}, m, r_{mid}, \epsilon)$

are also given a reference set R (with $\mathbf{vol}(R) \leq \mathbf{vol}(\bar{R})$) that defines a region of the graph in which we are searching for clusters. As noted previously, f_α can be viewed as a parametric linear program, and furthermore it will evaluate to a non-negative finite number for any $\alpha > 0$. Thus by Theorem 1, $G(\alpha) = \min_S f_\alpha(S)$ is concave and piecewise linear and we can apply Theorem 2. More explicitly, the local clustering parameter fitness function is

$$\mathcal{P}_X(\alpha) = \frac{\mathbf{cut}(X) + \alpha\mathbf{vol}(\bar{X} \cap R) + \alpha\varepsilon\mathbf{vol}(X \cap \bar{R})}{\min_S[\mathbf{cut}(S) + \alpha\mathbf{vol}(\bar{S} \cap R) + \alpha\varepsilon\mathbf{vol}(S \cap \bar{R})]}. \quad (16)$$

If we focus on finding clusters that are subsets of R, using objective (6), we have a simplified fitness function:

$$\mathcal{P}_X(\alpha) = \frac{\mathbf{cut}(X) - \alpha\mathbf{vol}(X) + \alpha\mathbf{vol}(R)}{\min_{S \subseteq R}[\mathbf{cut}(S) - \alpha\mathbf{vol}(S) + \alpha\mathbf{vol}(R)]}. \quad (17)$$

When we apply Algorithm 1 to minimize (16) or (17), we can query $\mathcal{P}_X$ in the time it takes to evaluate a linear function and the time it

takes to solve the s-t cut problem (5). This can be done extremely quickly using localized min-cut computations [20, 26, 35, 38].

Functions (16) and (17) should be minimized over $\alpha \in [\alpha^*, \mathbf{cut}(R)]$, where α^* is either the minimum of (4) if we are minimizing (16), or is the minimum conductance for a subset of R if we are minimizing (17). One can show that for any α outside this range, objectives (5) and (6) will be trivially minimized by $S = R$, so it is not meaningful to optimize these objectives for these α. In practice one can additionally set stricter upper and lower bounds if desired.

4.2 Global Clustering Approach

We separately consider the standard and degree-weighted versions of LambdaCC when applying Theorem 2 to global graph clustering.

Standard LambdaCC. For the standard objective, it is useful to consider the scaled version of LambdaCC obtained by dividing (1) by $1-\lambda$ and substituting for a new resolution parameter $\gamma = \lambda/(1-\lambda)$. Then the objective is

$$\min \ \sum_{(u,v)\in E}(1-\delta_{uv}) + \sum_{(u,v)\notin E}\gamma\delta_{uv}. \tag{18}$$

The denominator of the parameter fitness function for this scaled LambdaCC problem would be

$$G(\gamma) = \min_{\mathbf{x}\in\mathcal{X}} \ \sum_{(u,v)\in E} x_{uv} + \sum_{(u,v)\notin E}\gamma(1-x_{uv}) \tag{19}$$

where $\mathcal{X}$ represents the set of linear constraints for the linear program (2). Note that $G(\gamma)$ will be finite for every $\gamma \geq 0$, so Theorem 1 holds. Thus G is concave and piecewise linear as required by Theorem 2. Next, for a fixed clustering C_x, let P_x be the number of positive mistakes (pairs of nodes that are separated despite sharing an edge) and N_x be the number of negative mistakes (pairs of nodes that are clustered together but share no edge). Then objective (18) for this clustering is $P_x + \gamma N_x$, and we see that this fits the linear form given in Theorem 2 as long as the example clustering satisfies $P_x > 0$ and $N_x > 0$, which will be the case for nearly any nontrivial clustering one might consider. Finally, note that the parameter fitness function for (18) would be exactly the same as the parameter fitness function for the standard LambdaCC objective, since scaling by $(1-\lambda)$ makes no difference if we are going to minimize the ratio between the clustering objective and its LP relaxation. The parameter fitness function for standard LambdaCC is therefore

$$\mathcal{P}_{C_x}(\lambda) = \frac{(1-\lambda)P_x + \lambda N_x}{\min_{\mathbf{x}}\left[\sum_{uv\in E}(1-\lambda)x_{uv} + \sum_{uv\notin E}\lambda(1-x_{uv})\right]} \tag{20}$$

and it satisfies the assumptions of Theorem 2 as long as $P_x > 0$, $N_x > 0$, and we optimize over $\lambda \in (0,1)$.

Degree-weighted LambdaCC. Showing how Theorem 2 applies to degree-weighted LambdaCC requires slightly more work, though the same basic principles hold. The LP-relaxation of the objective is still a parametric linear program, thus is still concave and piecewise linear in λ over the interval $(0,1)$. The denominator of the parameter fitness function in this case would be:

$$\min \ \sum_{(u,v)\in E^+} e_{uv}(1-\delta_{uv}) + \sum_{(u,v)\in E^-} e_{uv}\delta_{uv}. \tag{21}$$

where e_{uv} is defined in the degree-weighted fashion (see Section 2.1). For a fixed example clustering C_x encoded by a function $\delta_x = (\delta_{uv})$, we can rearrange this into the form $a + \lambda b$ where $a = \sum_{(u,v)\in E}(1-\delta_{uv})$ and $b = \sum_{(u,v)\notin E} d_u d_v \delta_{uv} - \sum_{(u,v)\in E} d_u d_v (1-\delta_{uv})$. These values are simple to compute, and as long as they are both nonzero, the results of Theorem 2 apply. In some extreme cases it is possible that $a = 0$ or $b = 0$, but we expect this to be rare. Furthermore, our general approach may still work even when $a = 0$ or $b = 0$, Theorem 2 simply does not analyze this case. We leave it as future work to develop more refined sufficient and necessary conditions such that Algorithm 1 is guaranteed to minimize $\mathcal{P}$.

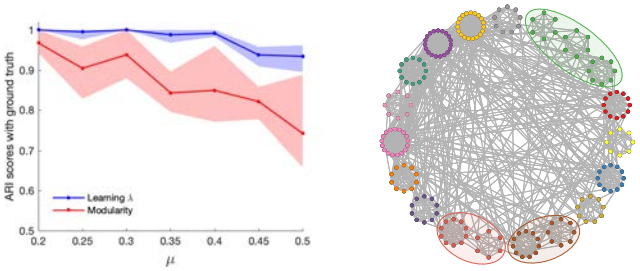


Figure 3: Left: ARI scores for detecting ground truth in LFR graphs. Solid lines indicate mean scores, and colored regions show the range of scores across 5 test graphs for each μ. Right: one of the 5 LFR test graphs for $\mu = 0.3$. Modularity ($\lambda = 1/(2|E|)$) makes mistakes by putting distinct ground truth clusters together (highlighted). For this example our approach perfectly detects the ground truth.

5 EXPERIMENTS

We consider several local and global clustering experiments in which significant benefit can be gained from learning resolution parameters rather than using previous off-the-shelf algorithms and objective functions. We implement Algorithms 1 and 2 in the Julia programming language for both local and global parameter fitness functions. Computing the LambdaCC linear programming relaxation can be challenging due to the size of the constraint set. For our smaller graphs we apply Gurobi optimization software, and for larger problems we use recently developed memory-efficient projection methods [30, 36]. For the local-flow objective we use a fast Julia implementation we developed in recent work [38]. Our experiments were run on a machine with two Intel Xeon E5-2690 v4 processors. Code for our experiments and algorithms are available at https://github/nveldt/LearnResParams.

5.1 Learning Parameters for Synthetic Datasets

Although modularity is a widely-applied objective function for community detection, Fortunato and Barthélemy [12] demonstrated that it is unable to accurately detect communities below a certain size threshold in a graph. In our first experiment we demonstrate that learning resolution parameters for LambdaCC allows us to overcome the *resolution limit* of modularity, and better detect community structure in synthetic networks. We generate a large number of synthetic LFR benchmark graphs [19], in a parameter regime that is chosen to be difficult for modularity. All graphs contain 200 nodes, average degree 10, max degree 20, and community sizes between 5 and 20 nodes. We test a range of mixing parameters μ, which controls the fraction of edges that connect nodes in different communities ($\mu = 0$ means all edges are inside the communities).

For each μ from 0.2 to 0.5, in increments of 0.05, we generate six LFR networks, one for training and five for testing. On the training graph, we minimize the degree-weighted LambdaCC parameter fitness function to learn a resolution parameter λ_{best}. This takes between roughly half an hour (for $\mu = 0.2$) to just over three hours (for $\mu = 0.5$), solving the underlying LambdaCC LP with Gurobi software. We then cluster the five test LFR examples using a generalized version of the Louvain method [7], as implemented by Jeub et al. [16]. We separately run the method with two resolution parameters: $\lambda = 1/(2|E|)$, the standard setting for modularity, and $\lambda = \lambda_{best}$. Learning λ_{best} significantly improves adjusted Rand index (ARI) scores for detecting the ground truth (see Figure 3).

5.2 Local Community Detection

Next we demonstrate that a small amount of semi-supervised information about target communities in real-world networks can allow us to learn good resolution parameters, leading to more robust community identification. Additionally, minimizing the parameter fitness function provides a way to measure the extent to which *functional* communities in a network correspond to topological notions of community structure in networks.

Data. We consider four undirected networks, DBLP, Amazon, Orkut, LiveJournal, which are all available on the SNAP repository [22], and come with sets of nodes that can be identified as "functional communities" (see Yang and Leskovec [40]). For example, members of the social network Orkut may explicitly identify as being part of a user-formed group. Such user groups can be viewed simply as metadata about the network, though these still correspond to some notion of community organization that may be desirable to detect. Following an approach taken in previous work [38], we specifically consider the ten largest communities from the 5000 best functional communities as identified by Yang and Leskovec [40].

Experimental Setup and Results. We treat each functional community as an example cluster X. We build a superset of nodes R by growing X from a breadth first search until we have a superset of size $5|X|$, breaking ties arbitrarily. The size of R is chosen so that it comprises a localized region of a large graph, but is still significantly larger than the target cluster X hidden inside of it. We compare two approaches for detecting X within R. As a baseline approach we extract the best conductance subset of R. Then as our new approach we assume we are given $\mathbf{cut}(X)$ and $\mathbf{vol}(X)$ as additional semi-supervised information. This allows us to minimize the parameter fitness function (17), without knowing what X is. This outputs a resolution parameter α_X, and we then minimize $\mathbf{cut}(S) - \alpha_X \mathbf{vol}(S) + \alpha \mathbf{vol}(R)$ over $S \subseteq R$ to output a set S_X.

Table 1 reports conductance, set size, runtimes, and F1 scores for both approaches, averaged over the ten communities in each network. Learning resolution parameters leads to significantly better F1 scores on every dataset. Additionally, learning resolution parameters for local clustering can be done much more quickly than learning λ for LambdaCC.

New Insights. In addition to improving semi-supervised community detection, minimizing $\mathcal{P}_X$ allows us to measure how well a functional community matches the topological notion of a cluster.

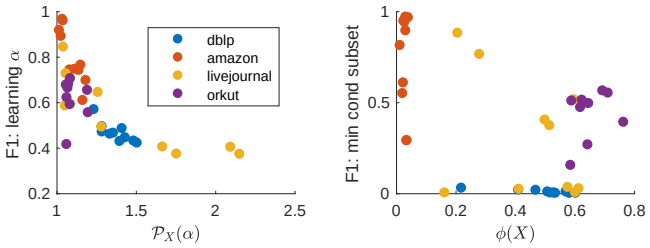


Figure 4: Left: F1 scores for detecting clusters X by learning α vs. the minimum of $\mathcal{P}_X$. Right: F1 scores obtained by finding $\min_{S \subseteq R} \phi(S)$ vs. $\phi(X)$. See the main text for discussion.

Figure 4 shows a scatter plot of F1 community recovery scores against the minimum of $\mathcal{P}_X$ for each experiment from Table 1. We note a downward sloping trend: small values of $\mathcal{P}_X$ near 1 tend to indicate that a cluster is highly "detectable," whereas a higher value of $\mathcal{P}_X$ gives some indication that the functional community may not in fact correspond to a good structural community. We also plot the F1 recovery scores for finding the minimum conductance subset of R against the conductance of functional communities. In this case we do not see any clear pattern, and we learn very little about the relationship between structural and functional communities.

5.3 Meta-Data and Global Clustering

Next we use our techniques to measure how strongly metadata attributes in a network are associated with actual community structure. In general, sets of nodes sharing metadata attributes should not be viewed as "ground truth" clusters [27], although they may still shed light on the underlying clustering structure of a network.

Email Network. We first consider the largest connected component of an email network [21, 41]. Each of the 986 nodes in the graph represents a faculty member at a European university, and edges represent email correspondence between members. We remove edge weights and directions, and consider an example clustering C_x formed by assigning faculty in the same academic department to the same cluster. We use our bisection method to approximately minimize the global parameter fitness function for the degree-weighted LambdaCC objective. We run our method until we find the best resolution parameter to within a tolerance of 10^{-8}, yielding a resolution parameter $\lambda_x = 6.5 \times 10^{-5}$ and a fitness score of $\mathcal{P}_{C_x}(\lambda_x) = 1.34$.

To assess how good or bad a score of 1.34 is for this particular application, we construct a new fake metadata attribute by performing a random permutation of the department labels, which gives a clustering C_{fake}. Approximately minimizing $\mathcal{P}_{C_{fake}}$ yields a resolution parameter $\lambda_{fake} = 3.25 \times 10^{-5}$ and a score $\mathcal{P}_{C_{fake}}(\lambda_{fake}) = 2.16$. The gap between the minima of $\mathcal{P}_{C_{fake}}$ and $\mathcal{P}_{C_x}$ indicates that although the true metadata partitioning does not perfectly map to clustering structure in the network, it nevertheless shares some meaningful correlation with the network's connectivity patterns. To further demonstrate this, we run the generalized Louvain algorithm [7, 16], using the resolution parameters λ_x and λ_{fake}. Running the clustering heuristic with λ_x outputs a clustering that has a normalized mutual information score (NMI) of 0.71 and an adjusted Rand index (ARI) score of 0.55 with C_x. Using λ_{fake}, we get NMI and ARI scores of only 0.05 and 0.003 respectively when comparing with C_{fake}.

Table 1: Output statistics on SNAP datasets for finding the minimum conductance (*mc*) subset, and learning a resolution parameter (*lr*). Runtime is given in seconds. We also provide average size ($|T|$) and conductance ($\phi(T)$) for target communities.

Graph	$\|V\|$	$\|E\|$	$\|T\|$	$\phi(T)$	F1		ϕ		run.		size	
					mc	*lr*	*mc*	*lr*	*mc*	*lr*	*mc*	*lr*
DBLP	317,080	1,049,866	3902	0.49	0.01	**0.47**	0.02	0.16	4.9	11.4	31	11680
Amazon	334,863	925,872	190	0.03	0.73	**0.80**	0.00	0.02	0.3	0.7	142	288
LiveJournal	3,997,962	34,681,189	988	0.45	0.30	**0.54**	0.06	0.10	13.7	31.3	1556	2940
Orkut	3,072,441	117,185,083	3877	0.65	0.44	**0.62**	0.42	0.46	129.8	272.6	2353	5727

Social Networks. We repeat the above experiment on the smallest social network in the Facebook 100 datasets [34], Caltech36. This network is a subset of Facebook with $n = 769$ nodes, defined by users at the California Institute of Technology at a certain point in September 2005. Every node in the network comes with anonymized metadata attributes reporting student/faculty status, gender, major, second major, residence, graduation year, and high school. We treat each metadata attribute as an example clustering C_x. Any node with a value of 0 for an attribute we treat as its own cluster, as this indicates the node has no label for the given attribute. We do not run Algorithm 1 for each individual C_x, since this would involve redundant computations of the LambdaCC LP relaxations for many of the same values of λ. Instead, we evaluate the denominator of $\mathcal{P}$, which is the same for all example clusterings, at 20 equally spaced λ values between $1/(8|E|)$ and $2/(|E|)$. We set values of λ to be inversely proportional to the number of edges, since we expect the effect of a resolution parameter to depend on a network's size. We note for example that the resolution parameter corresponding to modularity is $\lambda = 1/(2|E|)$, which is also inversely proportional to $|E|$. Computing all of the LP bounds is the bottleneck in our computations, and takes just under 2.5 hours using a recently developed parallel solver for the correlation clustering relaxation [30].

Having evaluated the denominator of $\mathcal{P}$ at these values, we can quickly find the minimizer of $\mathcal{P}$ for each metadata attribute and a permuted fake metadata attribute to within an error of less than 10^{-5}. The smallest values of the parameter fitness function $\mathcal{P}$ for both real and permuted (fake) metadata attributes are given below:

	S/F	Gen	Maj.	Maj. 2	Res.	Yr	HS
$\min \mathcal{P}_{real}$	1.30	1.73	2.03	2.12	1.35	1.57	2.11
$\min \mathcal{P}_{fake}$	1.65	1.80	2.12	2.12	2.11	2.09	2.12

We note that the smallest values of $\mathcal{P}$, as well as the largest gap between $\mathcal{P}$ for true and fake metadata clusterings, are obtained for the student/faculty status, residence, and graduation year attributes. This indicates that these attributes share the strongest correlation with the community structure at this university, which is consistent with independent results on the Facebook 100 datasets [34, 37].

5.4 Local Clustering in Social Networks

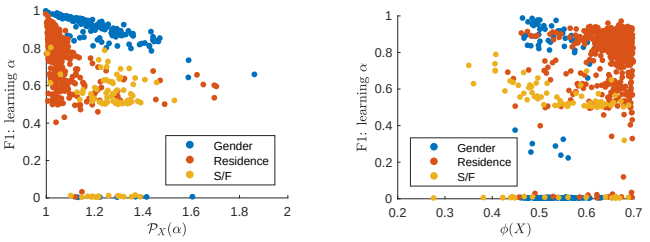
In our final experiment we continue exploring the relationship between metadata and community structure in Facebook 100 datasets. We find that minimizing a *local* parameter fitness function $\mathcal{P}$ can be a much better way to measure the community structure of a set of nodes than simply considering the set's conductance.

Data. We perform experiments on all Facebook 100 networks, focusing on the student/faculty status, gender, residence, and graduation year metadata attributes. For the Caltech dataset in the last experiment, these attained the lowest scores for a *global* parameter fitness function, and furthermore these are the only attributes with a significant number of sets with nontrivial conductance. For the graduation year attribute, we focus on classes between 2006 to 2009, since these correspond to the four primary classes on each campus when the networks were crawled in September of 2005 [34].

Experimental Setup. We return to an approach similar to our first experiment. For each network and metadata attribute, we consider sets of nodes identified by the same metadata label, e.g., X may represent all students in the class of 2008 at the University of Chicago. We will refer to these simply as *metadata sets*. A label of zero indicates no attribute is known, so we ignore these sets. We also discard sets that are larger than half the graph, or smaller than 20 nodes. We restrict to considering metadata sets with conductance at most 0.7, since conductance scores too close to 1 indicate that a set has little to no meaningful connectivity pattern. For each remaining metadata set X, we grow a superset R around X using a breadth first search, and stop growing when R contains half the nodes in the graph or is three times the size of X. We then minimize $\mathcal{P}_X$ as given by (17) to learn a resolution parameter α_X. This allows us to find $S_X = \operatorname{argmin}_{S \subseteq R} \mathbf{cut}(S) - \alpha_X \mathbf{vol}(S)$, and we then compute the F1 score between S_X and X. Our goal here is not to develop a new method for community detection. Rather, computing the F1 score and the minimum of $\mathcal{P}_X$ provide ways to measure how well a metadata set conforms to a topological notion of community structure, and how detectable the set is from an algorithmic perspective.

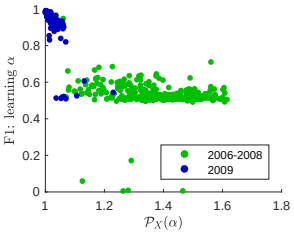
Results. While computing conductance scores provides a good first order measure of a node's community structure, we find that minimizing $\mathcal{P}$ provides more refined information for the detectability of clusters. In Figure 5 we show scatter plots of F1 detection scores against both $\min \mathcal{P}_X$ as well as $\phi(X)$ for each metadata set X. We see that especially for the gender and residence metadata sets across all networks, there is a much clearer relationship between F1 scores and $\min \mathcal{P}_X$. Values of $\mathcal{P}_X$ very close to 1 map to F1 scores near 1, and as $\mathcal{P}_X$ increases we see a downward sloping trend in F1 scores. In the conductance plot we do not see the same trend.

Figures 5c and 5d show results for metadata sets associated with the 2006-2009 graduation years. For this attribute there appears to be a relationship between both conductance and the $\min \mathcal{P}$ scores. Furthermore, in both plots we see a separation of the points roughly into two clusters. A deeper exploration of these trends reveals that

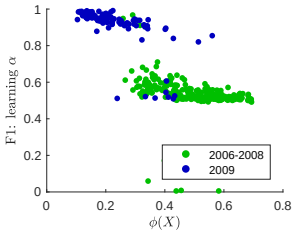


(a) $\mathcal{P}_X$: Gender, S/F, Residence

(b) $\phi(X)$: Gender, S/F, Residence

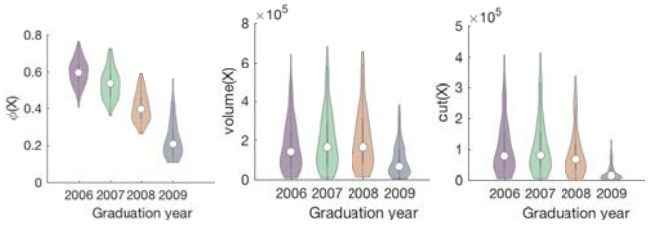


(c) $\mathcal{P}_X$: Grad Year



(d) $\phi(X)$: Grad Year

Figure 5: Minimizing $\mathcal{P}_X$ gives insight into the connectivity structure of different metadata sets in Facebook 100 datasets. Plotting F1 detection scores against the minimum of $\mathcal{P}_X$ shows especially clear trends for the gender, residence, and year attributes. Plots (c) and (d) highlight an anomaly in the connectivity patterns of the 2009 graduating year classes.



(a) Conductance (b) Volume (c) Cut

Figure 6: Small conductance and cut scores in Facebook 100 datasets indicate that freshman in 2005 were largely connecting on Facebook with people in their same class.

the 2009 graduation class accounts for the majority of one of these two clusters, and there appears to be an especially clear trend between F1 detection scores and both $\phi(X)$ and $\mathcal{P}_X$ for this class.

New Insights. Figure 6 shows violin plots for $\phi(X)$, **cut**(X), and **vol**(X) for metadata sets associated with graduation years from 2006 to 2009. Overall, conductance decreases as graduation year increases. We notice that class sizes for the 2009 graduation year are much smaller on average. When these datasets were generated, Facebook users needed a .edu email address to register an account. Thus, in September 2005, the graduation class of 2009 was made up primarily of new freshman who just started college, many of whom had not registered a Facebook account yet. Interestingly, we see a slight decrease in the median cut score from 2007 to 2008, and a significant decrease from 2008 to 2009 (Figure 6c). This suggests that although there were fewer freshman on Facebook at the time, on average they had a greater tendency to establish connections on Facebook among peers in their same graduation year.

Figure 6 suggests that in the early years of Facebook, with each new year, students in the same graduating class tended to form tighter Facebook circles with members in their own class. To further explore this hypothesis, for each of the 100 Facebook datasets we consider each node from a graduating class between 2006 and 2009. In each network we compute the average *in-class connection ratio*, i.e., the number of Facebook friends each person has inside the same graduating class, divided by the total number of Facebook connections that the person has across the entire university. In 97 out of 100 datasets (all networks except Caltech36, Hamilton46, and Santa74), this ratio strictly increases as graduation year increases. For Hamilton46 and Santa74, the ratio is still significantly higher for the 2009 graduation class than any other class. If we average this ratio across all networks, as the graduation year increases from 2006 to 2009, the ratios strictly increase: 0.39 for 2006, 0.45 for 2007, 0.57 for 2008, and 0.75 for the class of 2009. Previous work on has already highlighted the influence of the graduation year attribute on the connectivity structure of Facebook 100 datasets on the whole [34]. Our exploration of these datasets has further allowed us to uncover interesting new connectivity patterns that exist between different individual graduation years.

6 DISCUSSION AND FUTURE WORK

We have introduced a new framework and theory for learning resolution parameters based on minimizing a fitness function associated with a single example clustering of interest. There are several open questions for improving our specific approach. Our bisection-like algorithm is designed to be general enough to minimize a large class of functions to within arbitrary precision. However, by making additional assumptions on either specific clustering objectives or the fixed example clustering, one may be able to develop improved algorithms for minimizing the parameter fitness function in practice. Another open question is to study which other graph clustering objectives can fit into out framework, beyond just the LambdaCC global objective and the local flow clustering objective, and whether, for example, the approach can be applied to clustering in directed graphs.

Our work can be viewed as one approach to the more general goal of learning objective functions for graph clustering applications. This general goal could involve more techniques than simply learning resolution parameters. For example, in future work we wish to explore how to learn small *motif* subgraph patterns [6] in an example clustering that may be indicative of a desirable type of clustering structure in an application of interest.

ACKNOWLEDGMENTS

The authors thank several funding agencies: Nate Veldt is supported by NSF award IIS-154648, David Gleich is supported by the DARPA SIMPLEX program, the Sloan Foundation, and NSF awards CCF-1149756, CCF-0939370, and IIS-154648. Anthony Wirth is funded by the Melbourne School of Engineering.

REFERENCES

[1] Ilan Adler and Renato D.C. Monteiro. 1992. A geometric view of parametric linear programming. *Algorithmica* 8 (1992), 161–176.

[2] Nir Ailon, Moses Charikar, and Alantha Newman. 2008. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)* 55, 5 (2008), 23.

[3] Reid Andersen, Fan Chung, and Kevin Lang. 2006. Local Graph Partitioning using PageRank Vectors. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*. http://www.math.ucsd.edu/~fan/wp/localpartition.pdf

[4] Reid Andersen and Kevin Lang. 2008. An Algorithm for Improving Graph Partitions. In *Proceedings of the 19th annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2008)*. 651–660.

[5] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. 2004. Correlation Clustering. *Machine Learning* 56 (2004), 89–113.

[6] Austin R. Benson, David F. Gleich, and Jure Leskovec. 2016. Higher-order organization of complex networks. *Science* 353, 6295 (2016), 163–166. https://doi.org/10.1126/science.aad9029 arXiv:http://science.sciencemag.org/content/353/6295/163.full.pdf

[7] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008, 10 (2008), P10008. http://stacks.iop.org/1742-5468/2008/i=10/a=P10008

[8] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. 2005. Clustering with qualitative information. *J. Comput. System Sci.* 71, 3 (2005), 360 – 383. Learning Theory 2003.

[9] Shuchi Chawla, Konstantin Makarychev, Tselil Schramm, and Grigory Yaroslavtsev. 2015. Near optimal LP rounding algorithm for correlation clustering on complete and complete k-partite graphs. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*. ACM, 219–228.

[10] J.-C. Delvenne, Sophia N Yaliraki, and Mauricio Barahona. 2010. Stability of graph communities across time scales. *Proceedings of the National Academy of Sciences* 107, 29 (2010), 12755–12760.

[11] Santo Fortunato. 2010. Community detection in graphs. *Physics Reports* 486, 3 (2010), 75 – 174. https://doi.org/10.1016/j.physrep.2009.11.002

[12] Santo Fortunato and Marc Barthélemy. 2007. Resolution limit in community detection. *Proceedings of the National Academy of Sciences* 104, 1 (2007), 36–41. https://doi.org/10.1073/pnas.0605965104 arXiv:https://www.pnas.org/content/104/1/36.full.pdf

[13] Santo Fortunato and Darko Hric. 2016. Community detection in networks: A user guide. *Physics Reports* 659 (2016), 1 – 44. https://doi.org/10.1016/j.physrep.2016.09.002 Community detection in networks: A user guide.

[14] K. Fountoulakis, D. F. Gleich, and M. W. Mahoney. 2017. An Optimization Approach to Locally-Biased Graph Algorithms. *Proc. IEEE* 105, 2 (Feb 2017), 256–272. https://doi.org/10.1109/JPROC.2016.2637349

[15] David F. Gleich, Nate Veldt, and Anthony Wirth. 2018. Correlation Clustering Generalized. In *29th International Symposium on Algorithms and Computation (ISAAC 2018)*, Vol. 123. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 44:1–44:13. https://doi.org/10.4230/LIPIcs.ISAAC.2018.44

[16] Lucas G. S. Jeub, Marya Bazzi, Inderjit S. Jutla, and Peter J. Mucha. 2011-2017. A generalized Louvain method for community detection implemented in MATLAB. http://netwiki.amath.unc.edu/GenLouvain

[17] Lucas G. S. Jeub, Olaf Sporns, and Santo Fortunato. 2018. Multiresolution Consensus Clustering in Networks. *Scientific Reports* 8, 1 (2018), 3259. https://doi.org/10.1038/s41598-018-21352-7

[18] Kyle Kloster and David F. Gleich. 2014. Heat Kernel Based Community Detection. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2014)*. 1386–1395. https://doi.org/10.1145/2623330.2623706

[19] Andrea Lancichinetti and Santo Fortunato. 2009. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E* 80 (Jul 2009), 016118. Issue 1. https://doi.org/10.1103/PhysRevE.80.016118

[20] Kevin Lang and Satish Rao. 2004. A Flow-Based Method for Improving the Expansion or Conductance of Graph Cuts. In *Integer Programming and Combinatorial Optimization*. Lecture Notes in Computer Science, Vol. 3064. Springer Berlin Heidelberg, 325–337. https://doi.org/10.1007/978-3-540-25960-2_25

[21] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph Evolution: Densification and Shrinking Diameters. *ACM Trans. Knowl. Discov. Data* 1, 1, Article 2 (March 2007). https://doi.org/10.1145/1217299.1217301

[22] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[23] Mark EJ Newman. 2006. Finding community structure in networks using the eigenvectors of matrices. *Physical review E* 74, 3 (2006), 036104.

[24] Mark EJ Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical review E* 69, 026113 (2004).

[25] M. E. J. Newman. 2016. Equivalence between modularity optimization and maximum likelihood methods for community detection. *Phys. Rev. E* 94 (Nov 2016), 052315. Issue 5. https://doi.org/10.1103/PhysRevE.94.052315

[26] Lorenzo Orecchia and Zeyuan Allen Zhu. 2014. Flow-Based Algorithms for Local Graph Clustering. In *Proceedings of the 25th ACM-SIAM Symposium on Discrete Algorithms (SODA 2014)*. 1267–1286. http://arxiv.org/abs/1307.2855

[27] Leto Peel, Daniel B. Larremore, and Aaron Clauset. 2017. The ground truth about metadata and community detection in networks. *Science Advances* 3, 5 (2017). https://doi.org/10.1126/sciadv.1602548 arXiv:http://advances.sciencemag.org/content/3/5/e1602548.full.pdf

[28] Mason A Porter, Jukka-Pekka Onnela, and Peter J Mucha. 2009. Communities in networks. *Notices of the AMS* 56, 9 (2009), 1082–1097.

[29] Jörg Reichardt and Stefan Bornholdt. 2006. Statistical mechanics of community detection. *Physical Review E* 74, 016110 (2006).

[30] Cameron Ruggles, Nate Veldt, and David Gleich. 2019. A Parallel Projection Method for Metric Constrained Optimization. *arXiv preprint arXiv:1901.10084* (2019).

[31] Satu Elisa Schaeffer. 2007. Graph clustering. *Computer Science Review* 1, 1 (2007), 27 – 64. https://doi.org/10.1016/j.cosrev.2007.05.001

[32] Michael T. Schaub, Renaud Lambiotte, and Mauricio Barahona. 2012. Encoding dynamics for multiscale community detection: Markov time sweeping for the map equation. *Phys. Rev. E* 86 (Aug 2012), 026112. Issue 2. https://doi.org/10.1103/PhysRevE.86.026112

[33] Daniel A. Spielman and Shang-Hua Teng. 2013. A Local Clustering Algorithm for Massive Graphs and Its Application to Nearly Linear Time Graph Partitioning. *SIAM J. Comput.* 42, 1 (2013), 1–26. https://doi.org/10.1137/080744888

[34] Amanda L. Traud, Peter J. Mucha, and Mason A. Porter. 2012. Social structure of Facebook networks. *Physica A: Statistical Mechanics and its Applications* 391, 16 (2012), 4165–4180. https://doi.org/10.1016/j.physa.2012.03.0

[35] Nate Veldt, David Gleich, and Michael Mahoney. 2016. A Simple and Strongly-Local Flow-Based Method for Cut Improvement. In *Proceedings of The 33rd International Conference on Machine Learning (ICML 2016)*, Vol. 48. PMLR, New York, New York, USA, 1938–1947. http://proceedings.mlr.press/v48/veldt16.html

[36] Nate Veldt, David Gleich, Anthony Wirth, and James Saunderson. 2018. A Projection Method for Metric-Constrained Optimization. *arXiv preprint arXiv:1806.01678* (2018).

[37] Nate Veldt, David F. Gleich, and Anthony Wirth. 2018. A Correlation Clustering Framework for Community Detection. In *Proceedings of the 2018 World Wide Web Conference (WWW 2018)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 439–448. https://doi.org/10.1145/3178876.3186110

[38] Nate Veldt, Christine Klymko, and David F. Gleich. 2019. Flow-Based Local Graph Clustering with Better Seed Set Inclusion. In *Proceedings of the 2019 SIAM International Conference on Data Mining*.

[39] Di Wang, Kimon Fountoulakis, Monika Henzinger, Michael W. Mahoney, and Satish Rao. 2017. Capacity Releasing Diffusion for Speed and Locality. In *Proceedings of the 34th International Conference on Machine Learning (ICML 2017)*, Vol. 70. PMLR, International Convention Centre, Sydney, Australia, 3598–3607. http://proceedings.mlr.press/v70/wang17b.html

[40] Jaewon Yang and Jure Leskovec. 2015. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* 42, 1 (01 Jan 2015), 181–213. https://doi.org/10.1007/s10115-013-0693-z

[41] Hao Yin, Austin R. Benson, Jure Leskovec, and David F. Gleich. 2017. Local Higher-Order Graph Clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2017)*. ACM, New York, NY, USA, 555–564. https://doi.org/10.1145/3097983.3098069

[42] Linbin Yu and Chris Ding. 2010. Network Community Discovery: Solving Modularity Clustering via Normalized Cut. In *Proceedings of the Eighth Workshop on Mining and Learning with Graphs (MLG 2010)*. ACM, New York, NY, USA, 34–36. https://doi.org/10.1145/1830252.1830257

Auditing Offline Data Brokers via Facebook's Advertising Platform

Giridhari Venkatadri
Northeastern University

Piotr Sapiezynski
Northeastern University

Elissa M. Redmiles
University of Maryland

Alan Mislove
Northeastern University

Oana Goga
Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG

Michelle L. Mazurek
University of Maryland

Krishna P. Gummadi
MPI-SWS

ABSTRACT

Data brokers such as Acxiom and Experian are in the business of collecting and selling data on people; the data they sell is commonly used to feed marketing as well as political campaigns. Despite the ongoing privacy debate, there is still very limited visibility into data collection by data brokers. Recently, however, online advertising services such as Facebook have begun to partner with data brokers—to add additional targeting features to their platform— providing avenues to gain insight into data broker information.

In this paper, we leverage the Facebook advertising system—and their partnership with six data brokers across seven countries—in order to gain insight into the extent and accuracy of data collection by data brokers today. We find that a surprisingly large percentage of Facebook accounts (e.g., above 90% in the U.S.) are successfully linked to data broker information. Moreover, by running controlled ads to 183 crowdsourced U.S.-based volunteers, we find that at least 40% of data broker sourced user attributes are not at all accurate, that users can have widely varying fractions of inaccurate attributes, and that even important information such as financial information can have a high degree of inaccuracy. Overall, this paper provides the first fine-grained look into the extent and accuracy of data collection by offline data brokers, helping to inform the ongoing privacy debate.

ACM Reference Format:
Giridhari Venkatadri, Piotr Sapiezynski, Elissa M. Redmiles, Alan Mislove, Oana Goga, Michelle L. Mazurek, and Krishna P. Gummadi. 2019. Auditing Offline Data Brokers via Facebook's Advertising Platform. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313666

1 INTRODUCTION

Data brokers such as Acxiom [5] and Experian [23] have traditionally collected, aggregated, and linked information about people's activities, based on a variety of sources (e.g., voter records, vehicle registries, loyalty cards, and so forth). Their business model is then selling this information to third-parties such as banks, insurance companies, political campaigns, and marketers. The collection practices of data brokers have been the subject of an ongoing privacy debate. For example, data brokers typically only make data available to clients who purchase it, and not to the users who it is actually about [20]. Even worse, public-facing web sites run by data brokers [10] that purport to reveal the data only report a fraction of what they actually have [19, 47]. In fact, outside of a few niche areas (e.g., credit reports), people in the U.S. have limited if any rights to determine the provenance of, correct, or even view the data these companies have on them [38]. As a result, researchers and society at large still have a very limited understanding of the extent, accuracy, or provenance of this data collection ecosystem.

In parallel, *online services* such as Facebook and Google have been collecting information about people's online activities. Their business model is to build advertising platforms that use this data to provide advertisers with fine-grained targeting features [29, 31]. Recently, data brokers and online services have begun partnering together, allowing for the data collected about users online to be linked against data collected offline. This enables online services to provide advertisers with targeting features that concern users' offline information (e.g., advertisers can then target users based on their net worth, purchase behavior, and so forth [4, 34]). While this linking of offline and online data may seem to only further stoke the privacy debate, it does have one significant benefit: it offers a unique opportunity to gain visibility into the data broker ecosystem. Specifically, advertising platforms often provide advertisers with statistics about any audience an advertiser can target, *including those created using data broker-provided attributes.* This provides the first opportunity to gain insight into the extent of data collection by offline data brokers.

In this paper, we use the Facebook targeted advertising platform to study the *coverage* and *accuracy* of data collected by four offline data brokers (Acxiom, Epsilon, Experian, and Oracle Data Cloud - formerly Datalogix) across seven countries (U.S., U.K, Australia, Germany, France, Japan, and Brazil). [1] To examine the coverage of data brokers (i.e., the fraction of the population they have data on), we collect statistics from the Facebook advertising interface on how many Facebook users possess each of more than 600 data broker-provided attributes. Our results demonstrate that a large

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313666

[1] While Facebook reported having partnerships with six different brokers [4], we were able to find targeting attributes corresponding to only four of these brokers on Facebook's advertising platform.

percentage of Facebook accounts (e.g., ranging from over 48% of users in Japan to over 90% of users in the U.S.) are indeed linked to some data broker information, thus demonstrating for the first time the extent of offline-online data linkage, in addition to demonstrating the coverage of offline data brokers. Furthermore, we devise a methodology to study the coverage at a fine granularity based on the targeting mechanisms and statistics commonly provided by online advertising platforms in Section 3. We find that data broker coverage generally increases with the age of users, and can be significantly lower in U.S. counties with higher poverty levels. We also find that data brokers have high levels of coverage for potentially sensitive user attributes such as income (76.9% of all U.S. users), net worth (67.6%), and purchase behavior (86.7%).

Facebook provides users with an Ads Preferences Page [28] where they can see and correct what data Facebook has inferred about them; however, this page does not reveal to users *any* of the attributes sourced from data-brokers [2]. To study the accuracy of these attributes we use a recently-proposed transparency mechanism [53], which consists of running ads that reveal information about their targeting to the targeted users. Such ads, called *Transparency-enhancing advertisements* (Treads), work because advertising platforms by their very functionality only show targeted ads to users who satisfy the targeting of that ad. Thus, by running one distinct Tread targeting each individual data-broker-sourced attribute offered by the ad platform, we can reveal to users which data broker attributes they have.

To study the accuracy of attributes sourced from data brokers, we run Treads to a set of crowdworkers who installed a browser extension that collects the Treads seen by the worker, and surveys them about the accuracy of the inferred broker information. We use the results from 183 workers—corresponding to 1,432 individual inferred data broker attributes—to analyze the overall accuracy of broker and Facebook information, the per-user accuracy of broker and Facebook information, and the accuracy of broker information across various information categories. We find that the wide coverage of data broker information comes at the cost of mediocre accuracy, with around 40% of attributes being reported as "Not at all accurate" by workers. We also find that inaccuracy does not occur in an all-or-none fashion (i.e., users can have widely varying percentages of inaccurate data broker attributes). Finally, we find that even potentially sensitive attributes like financial information have mediocre accuracy, with for example 56% of attributes about investments reported as "Not at all accurate".

Taken together, our results present the first detailed look into the coverage and accuracy data broker ecosystem; our methodology could also be used to help trace the provenance of collected data, as well as to study how data collection practices change over time.

2 BACKGROUND

We begin by providing some background about offline data brokers, before explaining relevant parts of Facebook's advertising interface.

2.1 Offline data brokers

While some data brokers such as Acxiom [5], Epsilon [21], and Datalogix (now part of Oracle Data Cloud [39]) are geared towards aggregating consumer data to sell to marketers and advertisers, other companies such as Experian [23] have a greater focus on creating credit reports for businesses.

Kinds of data collected The data collected by data brokers is diverse and often very sensitive. It can include demographic information such as ethnicity and occupation, household characteristics such as age and number of children, financial information such as income and net worth, life events such as divorces, and credit-related information about property, mortgages, and investments [9, 26]. Other information collected can include sensitive health-related information such as ailments, medications, visual impairments, and health indicators [8, 26]. Publicly available detailed catalogs of data offered by data brokers are hard to find; we were only able to find the following data catalogs corresponding to Experian [26], Oracle Data Cloud [40], and Acxiom [6]. These catalogs show the diversity, sensitivity, and fine-grained nature of the information collected by these data brokers, highlighting the importance of understanding their data collection better.

Extent of data collection Data brokers reveal only high-level information about the extent of the data they collect. For example, Acxiom reveals that its data collection "now encompasses more than 62 countries, 2.5 billion addressable consumers and more than 10,000 attributes", for a "comprehensive representation of 68 percent of the world's online population" [7]. Similarly Epsilon reveals that its data covers "virtually every U.S. household" [22], and Experian reveals that it "maintains credit information on approximately 220 million U.S. consumers and 25 million active U.S. businesses", "demographic information on approximately 235 million consumers in 117 million living units across the U.S.", and "information on more than 650 million vehicles in the U.S. and Canada" [24]. However, except to their clients, these data brokers usually do not reveal detailed lists of what kinds of information they collect, and how many users are covered by different kinds of information.

Some data brokers do allow individuals to gain some insight into the data collected about them, such as Acxiom's "About the Data" site [10] and Experian's free credit reports [25]. However, these are only sanitized versions of their data and only show a fraction of the actual data collected [19, 47].

2.2 Facebook's advertising platform

Facebook's advertising platform is one of the largest and most mature online advertising platforms. It leverages Facebook's user data to allow advertisers to target *audiences* of users (i.e., sets of users with particular attributes) with ads.

Targeting parameters When specifying an audience, advertisers must specify the location (e.g., countries, states, or postal codes), age range (between 13 and 65+), and gender of users to target. In addition, advertisers can choose to include or exclude users who have certain attributes, choosing from a list containing 1,121 attributes; of these, 614 attributes come from Facebook and are present in all countries, while the others come from data brokers and can vary across countries [2]. In addition, advertisers are able to create targeting formulas of boolean expressions over attributes [11].

Size estimates Once the advertiser specifies the particular attributes of users to target, Facebook then gives the advertiser a size estimate (called the *potential reach*) that represents the number of

monthly active Facebook users who meet the targeting criteria.[2] Prior work found that the potential reach was calculated by rounding the number of matching users to two significant digits [52].

Partner categories Facebook refers to the offline information obtained by linking its profiles with data brokers (also called Facebook Marketing Partners) as *partner categories* [4]. Facebook reports that partner categories are "available to people targeting audiences located in the United States, Brazil, France, Germany, the United Kingdom, Australia, and Japan"; this includes data from Acxiom, Acxiom Japan, CCC Marketing (for Japan), Epsilon, Experian, Oracle Data Cloud, and Quantium [4]. Facebook recently announced that it would soon be terminating these partnerships and removing these categories [49] in the wake of the Cambridge Analytica scandal, and ultimately did so in October 2018.

Linking partner data One key concern with partner data is the *linking* process: essentially, identities in the partner's database need to be correctly linked with Facebook's user database to enable accurate targeting. While the exact linking methodology is not public, Facebook has stated that it is based on users' unique identifiers [34]:

> For each Facebook user, the company computes a hash code of the person's phone number, email address and other major identifiers and transmit those to Acxiom, Oracle, Epsilon and the other data brokers it works with and requests that they return all available marketing segments they offer for that user.

2.3 Related work

We next review closely related work on data brokers, linkage between offline and online identities, and online advertising platforms.

Coverage of data brokers To the best of our knowledge, there have been no studies analyzing the coverage of data brokers in a fine-grained manner; as previously mentioned, this has been due to the difficulty of conducting such a study owing to the opaque practices of data brokers. While the U.S. Federal Trade Commission (FTC) published a report in 2014 describing data brokers, their sources of data, and their clients [20], they were unable to report on the coverage of data by these data brokers at a fine-grained level.

Accuracy of data brokers There have been a number of anecdotal reports that show that data collected by data brokers can be inaccurate. For example, a journalist obtained a copy of their information held by Oracle Data Cloud [34], and examined their data as provided by Acxiom's "About the Data" site, finding that more than 70% of their attributes from either source were inaccurate. Similarly another journalist [13] found their information from Acxiom highly inaccurate, while yet another [37] found nearly 50% of their personal information purchased for a $50 fee from an undisclosed company inaccurate. While these results are intriguing, they each represent data for only a single individual.

A recent white paper [42] presented a small-scale study of 107 Deloitte U.S. employees, asking them to review their data revealed by a "leading consumer data broker" with "a publicly available, web-based portal that presents users with a variety of personal and household data." These authors restrict themselves to a small, arbitrary subset of 30 attributes, finding that nearly half of the attributes are incorrect for more than 50% of users who have them. They do not report how many of the 107 participants actually had data about them and responded to the survey. Another small-scale study [45] surveyed 8 graduate students about their information collected by Oracle BlueKai as revealed by its transparency feature, with all participants finding some inaccuracies in their profiles. These two studies focus on a non-representative sample of users, on one data broker, and (in the case of the former study) on a small set of attributes provided by the broker; in comparison, our methodology allows us to study a more general sample of users, covering multiple data brokers, avoiding the use of a sanitized version of the data, and focusing on all 507 data broker attributes that Facebook makes available through its advertising platform.

Online-offline linkage Prior work [36] has explored the potential for data brokers to link online data (Facebook profiles) and offline data (voter registration records) in target cities in the U.S., presenting one method to link such profiles. Since data brokers might potentially use more sophisticated methods to accomplish such linkage, our work makes the complementary contribution of empirically demonstrating the extent of such online-offline linkage.

Online advertising platforms A number of recent studies have looked at the privacy implications of online advertising platforms. For example, multiple projects [27, 33, 52] have discovered serious privacy leaks on Facebook's ad platform, and others [48] demonstrated the potential for abusing Facebook's ad platform to launch hard-to-detect discriminatory advertising on Facebook's ad platform. Other researchers [17, 46] have demonstrated that these weaknesses are not confined to Facebook, and are present in Google's ad platform as well. The recent inclusion of data broker-derived data—often consisting of sensitive information such as financial information—could exacerbate such ill effects.

On the other hand, recent work [2] studied Facebook's transparency mechanism that purports to show users why they were targeted with a particular ad. They found that Facebook's transparency feature does reveal information about Facebook's attributes, but *does not* reveal any information about data broker attributes that have been collected about users. Our work is therefore the first to shed light on the usage of data broker information in this ecosystem.

A number of recent studies have used Facebook's advertising platform for various demographic studies, either estimating the coverage of sensitive attributes computed internally by Facebook [16], studying the distribution of health conditions using various indirect markers [35], studying the gender divide on Facebook [30], or studying the migration of populations [55]. Our paper, on the other hand, estimates the extent of linkage between online and offline identities, and the coverage achieved by offline data brokers, using Facebook's advertising platform.

Finally, a number of recent studies have examined the accuracy of information revealed by the transparency mechanisms of online advertising platforms. While one study [50] found that participants' age and gender as revealed by Google's transparency mechanism were accurate for 65% - 74% of participants, and had missing values for 18%-29% of participants; another study [14] found that over 52% of participants reported less than half of their information as listed by the transparency mechanisms of Google, Facebook, and a

[2]Facebook previously defined potential reach as measuring the number of *daily* active users [52]; however, the definition of potential reach has always captured the notion of active users.

small broker Nielsen eXelate as "relevant". Finally, 27% of users in a recent report [1] found their information revealed by Facebook's transparency mechanism inaccurate. Our work instead chiefly focuses on the coverage and accuracy of information aggregated by offline data brokers.

3 COVERAGE

We next describe our methodology for examining the coverage of data collected by data brokers, before analyzing the results.

3.1 Methodology for studying coverage

Obtaining data broker attributes Similar to prior work [2], we use the "web inspector" feature of our web browser to identify the API call made by Facebook's ad interface to retrieve the list of targeting attributes shown to an advertiser.[3] These attributes are organized in a hierarchy, with the highest level categories being "Interests", "Demographics", and "Behaviors". These are then subdivided into sub-categories; the targeting attributes themselves are the leaf nodes of this hierarchy. For all attributes, the result of the API call contains the name of each targeting attribute, its parents in the hierarchy, and the total number of users with that attribute. For partner attributes, the result also contains the name of the partner (data broker) that the attribute was sourced from, and a brief description about the attribute.

We notice that the list of targeting attributes shown depends on the country in which the advertising account is created (and not, for example, on the country of the audience being targeted). Thus, to obtain the list of targeting attributes across different countries, we create Facebook accounts while logged in to an Amazon Web Services (AWS) instance based in that country, and then obtain the corresponding list of attributes shown. We collected these lists of targeting attributes in April 2017 for all seven countries (U.S., U.K, Australia, Germany, France, Japan, and Brazil) where Facebook reports it offers partner categories [4].

We found partner categories in all seven countries, with a varying number of data brokers in these different countries as can be seen from the first two columns of Table 1. These included all but two brokers—Quantium (for Australia) and CCC Marketing (for Japan)—reported by Facebook [4]; this may be because Facebook allows advertisers to request additional categories on a case-by-case basis from certain partners (as opposed to offering them to all advertisers via the ad interface). In addition, we collected these lists of targeting attributes for all three other countries (Canada, South Korea, and India) where we could create AWS instances; as expected, we found no partner categories corresponding to these countries.

Measuring extent of data broker coverage While it would be ideal to measure the percentage of *all* Facebook identities that have been linked to data broker information, the platform offers no way of estimating this. Instead we focus on the set of Facebook identities that are *targetable* by advertisements, since Facebook provides us estimates of the size of this set (via the *potential reach* estimates shown when creating an ad).[4]

[3]One only needs a Facebook account in order to be an advertiser and use Facebook's ad interface.

[4]These estimates could vary over longer intervals of time as they count active users [52]; thus, we ensure that all measurements corresponding to that population are made within short intervals of time. Also, these estimates are rounded to have two significant digits [52]; we verified that the magnitude of this measurement error does not affect our findings.

Country	Partner	Attribute count	Targetable		Percent
			Overall	Partner	
U.S.	**All**	507	210M	190M	90.5%
	Acxiom	128	210M	160M	76.2%
	Datalogix	350	210M	160M	76.2%
	Others[5]	10	210M	150M	71.4%
	Experian	5	210M	140M	66.7%
	Epsilon	14	210M	130M	61.9%
Australia	**All**	58	16M	13M	81.3%
	Experian	34	16M	12M	75.0%
	Acxiom	24	16M	9.1M	56.9%
U.K.	**All**	139	39M	29M	74.4%
	Acxiom	103	39M	22M	56.4%
	Datalogix	19	39M	17M	43.6%
	Experian	17	39M	15M	38.5%
Germany	Acxiom	60	31M	20M	64.5%
France	Acxiom	21	32M	18M	56.3%
Brazil	Experian	20	120M	61M	50.8%
Japan	Acxiom	17	25M	12M	48.0%

Table 1: Coverage of different data brokers across countries with partner categories. We show the total number of broker attributes, the number of Facebook identities that are targetable (Overall), the number of these identities that have at least one attribute from that broker (Partner), and the resulting coverage. Countries with more than one broker have a row indicating the coverage of all the brokers together (All).

We then estimate the coverage achieved by a broker within a particular population: *First*, we obtain the number of targetable identities corresponding to all users in that population (e.g., by obtaining the potential reach targeting that population). *Second*, we obtain the number of targetable identities within that population that have at least one attribute from that particular broker (we can do this since Facebook's ad interface allows us to target an OR of attributes, as mentioned in Section 2). While each of these numbers represents subsets of the overall population that are targetable, their ratio provides an estimate of the coverage in the overall population.

We are then able to study the coverage across different subpopulations (e.g., across genders, ages, locations, etc.), as Facebook allows us to additionally filter by those attributes (see Section 2).

Limitations Our methodology has a few limitations worth discussing. *First*, the population of targetable identities could be a biased sample of both the overall population of Facebook users, and of the overall population of that targeted country as a whole. We briefly explore this bias by focusing on the U.S., comparing the distribution of targetable identities with those of the overall population of the country (sourced from the U.S. Census Bureau's 2017 American Community Survey [ACS] 1-Year Estimates [3]). From Table 2, we see that the distribution across gender is similar for both targetable identities and for the U.S. population. However, we can see that the population of targetable identities is biased towards lower ages, with a larger percentage of people aged 44 or below than in the overall U.S. population. Nevertheless, this sample can still provide us a unique opportunity to study the coverage of data broker information across a large sample of users.

	Value	U.S. Census	Targetable	Surveyed (§4)
Age	15-24	13.3%	17.9%	22.9%
	25-34	13.8%	26.2%	37.1%
	35-44	12.6%	17.9%	22.4%
	45-54	13.0%	14.6%	8.7%
	55-64	12.9%	11.7%	7.1%
	65+	15.6%	10%	0.0%
Gender	Male	48.7%	47.8%	40.9%
	Female	51.3%	52.2%	56.8%

Table 2: Demographics of the entire U.S. population (U.S. Census), all targetable U.S. Facebook users (Targetable), and the crowdsourced users surveyed in Section 4 (Surveyed).

Second, data brokers might provide only a subset of their information to Facebook. However, Facebook has revealed that it requests "all available marketing segments they offer" from its partners [34], suggesting this is not the case. *Third*, the measured coverage of data brokers depends on the extent of linkage between Facebook profiles and offline data broker information, and thus can only provide a lower bound on the actual coverage of the data brokers. In the next section, we first demonstrate that there is a high extent of linkage between Facebook profiles and offline data broker information, thus suggesting our lower bounds are tight.

3.2 Analysis of coverage

We now examine the results on data broker coverage obtained using the methodology just described.

Extent of online-offline identity linkage We begin by examining the fraction of Facebook accounts that are linked with *any* data broker information. To do so, for each country, we measure the percentage of targetable Facebook identities that have *at least one* attribute from any of the brokers offering attributes within that country. This percentage reveals the extent of linkage of Facebook identities with offline information and serves as a lower bound.

Table 1 shows the percentage of targetable identities in different countries that have at least one data broker attribute; for countries with multiple data brokers, we include an **All** line that matches users from any of the data brokers. We make two observations: *First*, the extent of linkage is surprisingly high for the U.S., with over 90% of targetable Facebook identities having at least one data broker-provided attribute; similar high percentages of linkage are observed for Australia (81.3%), and for the U.K. (74.4%). *Second*, the extent of linkage is higher for countries where Facebook links to a larger number of data brokers (and obtains a larger number of attributes), indicating that even for the other countries, Facebook might have the ability to link a larger percentage of identities if it simply partnered with more data brokers. Taken together, these results suggest that offline and online data are being linked at massive scale. Having observed that our methodology can yield good lower bounds, we move on to further characterize the coverage.

Coverage of individual data brokers We measure the overall coverage achieved by various individual data brokers[5] in each of the seven countries we study; the results are presented in Table 1.

[5]We observed that 10 particular targeting attributes for the U.S. were labeled as coming from "data providers", indicating they are sourced from multiple brokers; in a previous crawl earlier in 2017, we observed that these were exclusively sourced from Acxiom. We mark these as "Others" when reporting results.

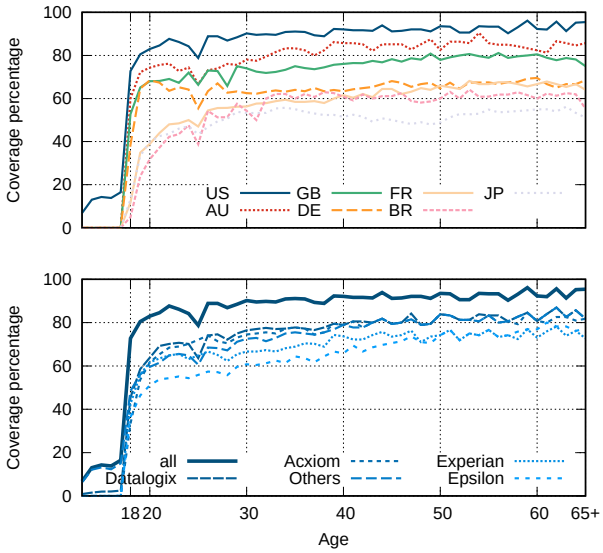


Figure 1: Breakdown of data broker coverage by age across multiple countries (top) and only within the U.S. (bottom).

We make three observations: *First*, we see that almost all the brokers achieve a coverage of at least half the targetable population in the country, irrespective of the country, showing that data brokers are successful in aggregating information about users at a global level. *Second*, we also see that the coverage can be as high as 76.2% (for Acxiom and Datalogix (now part of Oracle Data Cloud [39]) in the U.S.) and 75% (for Experian in Australia). *Third*, we see that the coverage achieved by the same broker can vary significantly (e.g., Experian covers over 66%, 75%, 38%, and 50% in the U.S, Australia, U.K., and Brazil, respectively).

Variation in coverage across ages One powerful feature of Facebook's advertising service is the ability to sub-divide any audience by additional features. As an example, we briefly study the variation of overall coverage of all data brokers in a country with users' age at the top of Figure 1. Similarly, we study the variation in coverage of individual data brokers from the U.S. at the bottom of Figure 1.

In either case, we study variation of coverage across ages taken from the set {13, 14, 15, ..., 65+}; these correspond to the ages that the interface allows us to target. We firstly observe that across countries, or across brokers in the U.S, the coverage rapidly increases between the ages of 18 and 20; this might be because these ages roughly correspond to the ages when people start acting with financial independence, and indicate that data brokers are rapidly able to cover people once they start acting as independent adults. We secondly observe that the coverage generally increases up to the age of 30, after which the rate of increase with age slows down; one exception is Epsilon in the U.S, whose coverage increases from 60.8% at age 30 to over 77.2% at age 65+. Finally, we observe that while the coverage for people below 18 years of age is zero for countries other than the U.S.; the corresponding coverage for the U.S. is non-zero, not dropping below 6.8% for the minimum studied age of 13. We observe that the data brokers in the U.S. with non-zero coverage for these users are Oracle Data Cloud (dropping from a

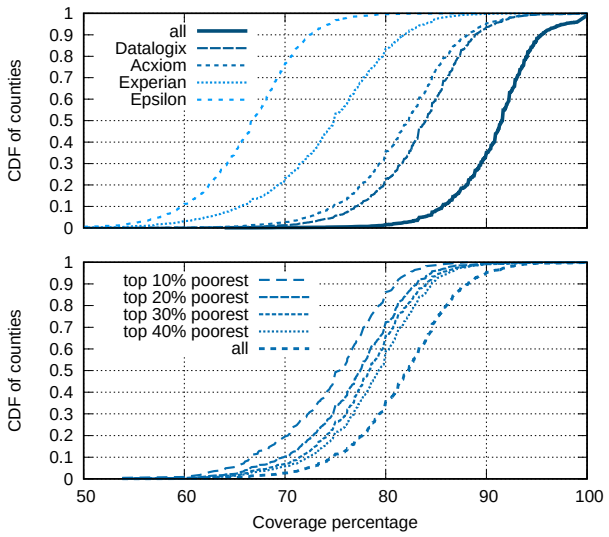


Figure 2: Cumulative distribution function (CDF) of data broker coverage across different U.S. counties (top), and of Acxiom for counties ranked by population below poverty level, as rated by the U.S. Census (bottom).

Category	Coverage
Behaviors → Purchase behavior	86.7%
Demographics → Financial → Income	76.9%
Demographics → Home → Home Ownership	72.3%
Behaviors → Automotive	68.7%
Demographics → Financial → Net Worth	67.6%
Behaviors → Financial → Spending methods	65.6%
Demographics → Home → Home Type	45.6%
Behaviors → B2B → Company size	44.1%
Demographics → Home → Home Type → Home value	43.1%
Behaviors → B2B → Industry	40.5%
Behaviors → Charitable donations	40.5%
Behaviors → Financial → Investments	40.0%
Demographics → Financial → Net Worth → Liquid assets	38.5%
Behaviors → Media → Television → Show Genre	28.7%
Behaviors → Travel	26.1%

Table 3: List of all 15 categories of data broker attributes provided on Facebook, along with the coverage (i.e., the fraction of targetable U.S. Facebook users who have at least one attribute in that category). We observe impressive coverage, with a significant variance across categories.

2.4% coverage for age 17 to a 0.8% coverage for age 13), and the brokers labeled by Facebook as "data providers", marked "Others" in the plot (dropping from 15% for age 17 to 6.3% for age 13).

Variation in coverage across U.S. counties We move on to study the variation of data broker categories across fine-grained geographic areas, plotting the CDF of the variation across different U.S. counties in Figure 2 (top). Since we cannot directly target counties on Facebook's ad platform—but we can target ZIP codes—we obtain a mapping from ZIP codes to counties [51]; we filter out military ZIP codes, P.O. box ZIP codes, and unique ZIP codes (dedicated for use by large organizations), giving us 29,916 ZIP codes in total. We further only consider ZIP codes from counties from the 50 states of the U.S. (3,110 counties in total). In the figure, we observe that the broker coverage varies widely across counties, roughly varying between 65% and 95% for Acxiom and Oracle Data Cloud, 50% and 80% for Epsilon, and 50% and 90% for Experian.

While on one hand, increased data broker coverage can have negative privacy implications, on the other hand there are a number of benefits to consumers from data broker coverage, such as better fraud prevention, improved product offerings, and more relevant (tailored) advertisements [20]. Thus, we study whether the extent of data broker coverage is correlated with the socioeconomic development of counties, specifically measured by the percentage of population in a county that is below the poverty level as per the 2012-2016 American Community Survey 5-Year Estimates [12]. In Figure 2 (bottom), we plot the CDF of coverage of one data broker (Acxiom) across the top percentiles of counties (in terms of having the highest fraction of population below poverty line), in addition to plotting the CDF of coverage across all counties. While we only show results for one data broker (owing to space constraints), we find similar results with other data brokers, and with the overall coverage of all data brokers taken together. We see that the coverage by Acxiom is indeed lower for counties with higher fractions of population below poverty line; for example, while 64.3% of all counties have at least 80% of their population covered by Acxiom, only 13.8% of those in the top 10th percentile have a similar coverage by Acxiom. This shows that data brokers indeed achieve lower coverage in counties with higher poverty rates.

Variation in coverage across attributes To understand the variation in coverage across various kinds of information, we group the 507 broker attributes available for the U.S., according to the second-to-last level of the hierarchy presented by Facebook, which we refer to as a *category*. Doing so gives us 15 categories in total where the "leaf" attributes are provided by data brokers. For example, the category Demographics → Financial → Net Worth → Liquid assets represents peoples' total liquid assets; all of the leaf attributes in this category together cover 100M people, and there are nine different leaf attributes including $1-$25K (covering 23M people), $25K-$50K (14M people), ..., up to >$3M (2.7M people). Similarly, the category Behaviors → Purchase behavior represents the kinds of purchases people make; it covers 169M people and has 175 different leaf attributes including Coupon users (100M people), Childrens' cereals (24M people), and Over-the-counter medication (43M people).

For each category, Table 3 shows the percentage of the U.S. population that is covered by at least one attribute corresponding to that category. We find that the coverage can vary significantly with the kind of targeting information, and can be as high as 86.7% for purchase behavior, and higher than 65% for sensitive categories like net worth, spending methods, home ownership, and income, but much lower 38.5% for net worth in the form of liquid assets.

4 ACCURACY

We move on to study the accuracy of data broker attributes, focusing on U.S.-based users. We first discuss the methodology we use before diving into the results.

4.1 Methodology for studying accuracy

Revealing user attributes to users While Facebook reveals Facebook-derived attributes, it has been shown to not reveal users' data broker attributes [2]. Thus, we leverage a recently-proposed mechanism that uses Facebook's advertising platform to enforce transparency on itself. Called *Transparency-enhancing advertisements* (Treads), the mechanism allows an advertiser to reveal platform-collected information about users by targeting them with ads that contain information about their targeting [53]. Specifically, we reveal to users (study participants) whether they have a particular data broker attribute by running one ad targeting all study participants, and another ad targeting only those participants who have that particular data broker attribute. If a user receives the first ad, we know they are targetable; if they receive the second as well, we know they have the data broker attribute.

To be able to target participants, we ask them to 'Like' our study's Facebook page. We then target Treads to the set of users who liked our Facebook page, with one Tread for each of the 507 data broker categories in the U.S (from Section 3.1). Since explicitly including the categories in the ad may be against Facebook's advertising policies [53], we encode each attribute by a unique numeric identifier, which we then translate into innocuous-looking text. For example, we might assign a particular attribute the identifier 852, which is then translated into "We have 8 and 52 ideas. Stay tuned."

Data collection In order to decode these Treads and to infer a user's data broker attributes, we built a browser extension for Google Chrome. The extension scans for our ads on Facebook as the user normally browses, and locally stores only the numeric identifiers in the Treads that are seen (for privacy reasons).

We then use our extension to survey users about the accuracy of their data broker attributes. As a baseline, we also survey users about the accuracy of information about them collected internally by Facebook. To do so, we gather the user's Facebook ad preferences page [28] just prior to the survey; we call these *Facebook attributes*.

When surveying participants, we randomly shuffle the data broker attributes and Facebook attributes, without revealing which is which to users. In order to avoid user fatigue, we survey them on a random sample of at most 25 data broker attributes and at most 30 attributes overall. For each user attribute, we show the user the attribute, and ask them questions about its accuracy. Specifically, we ask users whether the inferred attribute is "Not at all accurate", "Somewhat accurate", "Mostly accurate", or "Completely accurate." For data broker attributes, we also show Facebook's description of the attribute to the user.[6] For Facebook attributes, this description is not available; hence, we only show the attribute. Finally, we ask about their demographic information. Users are asked to answer only those questions they feel comfortable answering.

Changed targeting attributes Early-on during our experiments in this section (in August 2018), we found Facebook stopped supporting 120 out of the 507 original targeting attributes. [7] By freshly obtaining the set of targeting attributes from the API (in the same way as in Section 3.1), we found that each of the above 120 attributes was replaced by a similar attribute. However, the partner for all these attributes was now described as a generic "data license", perhaps indicating they were being sourced from multiple data brokers. In any case, we ran an additional 120 Treads for these attributes, and report accuracy for these under the column of "Others."

[6]For example, the attribute Behaviors → Charitable donations → Veterans , has the description, "People who are interested in donating to veteran causes."

[7]Of these 120 attributes, 96 were sourced from Acxiom (corresponding to information such as net worth, investments, spending methods, home type, charitable donations etc.), 21 were from Oracle (corresponding to information such as travel), two from Experian (corresponding to information such as home ownership) and one from Epsilon (corresponding to the length of residence).

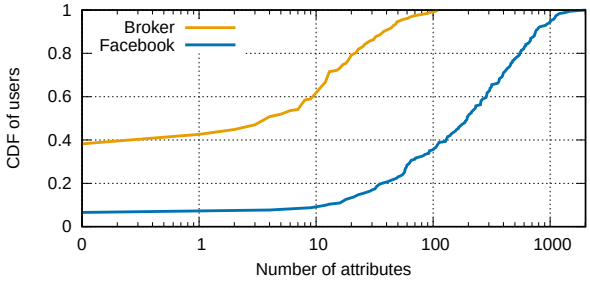


Figure 3: Distribution of inferred broker and Facebook attributes across the 183 participants who took the survey.

4.2 User recruitment

Our user study was reviewed and approved by Northeastern University's Institutional Review Board (IRB). We recruited users from Prolific [44], a crowdsourcing platform geared towards researchers; the quality of data obtained from Prolific users has been found to be comparable to that of the dominant crowdsourcing platform, Amazon Mechanical Turk [41]. We used Prolific's filters to selectively target our study at users aged 18 and above, residing in the U.S., and who use Facebook on a regular basis (at least once a month). We also followed Prolific's guidelines to selectively obtain high-quality participants [18], only considering users who had at least 10 prior submissions on Prolific, and an approval rating of at least 95 out of 100. This gave us a pool of over 4,000 potential participants, from which we recruited 300 users.

Participants were invited to participate in a research study about their reactions to their Facebook information (we did not mention data brokers until the end of the study so as not to bias participants). Participants were first asked to install our extension, for which they were paid $2. Participants were also asked to browse Facebook as they normally would in the subsequent days, and were informed how our extension works. After a period of a month, we followed up with the 231 users that still had our extension correctly installed, and requested them to take the survey (via the extension). We compensated participants $8 for completing the survey. 183 users completed the survey; this attrition rate of 39% (compared to the initial 300 users recruited) is well within the range of attrition rates observed in longitudinal studies on Prolific [18].[8]

Figure 3 shows the distribution of the number of data broker and Facebook attributes for these users who took the survey.[9] While around 61% of these users had at least one data broker attribute, 41% of the users had 10 or more data broker attributes, with the median user having four attributes. The median participant had 196

[8]The attrition was partially due to delays (of over a week) on our part in setting up the survey due to technical issues, and due to some users mistakenly assuming that the study had ended.

[9]Due to a bug in our extension in the early stages of our extension (where it failed to scan some users' Facebook home pages for our Treads, we potentially missed some users' data broker attributes. Thus, the plot potentially shows a lower bound on the actually revealed users' broker attributes.

Facebook attributes; given our limits on the number of attributes surveyed, the median number that we surveyed them on was 22.

Participant demographics To quantify the bias in the set of surveyed participants, we compare their demographics with those of the overall U.S. population [3]. For age/gender, we compare against the population of targetable Facebook identities (see Section 3).

Table 2 compares the age and gender distribution of participants. We find 56.8% of participants are female, as opposed to 52.2% of targetable Facebook identities and 51.3% of the entire U.S. population. We also find 82.4% of participants are younger than 45, as opposed to 62% of the targetable Facebook identities, and 39.7% of the offline population. Overall, this indicates a slight bias towards women, and a larger bias towards younger people, that is qualitatively similar to the bias of targetable Facebook identities.

We find that 63.4%, 8.1%, and 8.7% of our participants reported a single race of White, Black, and Asian, respectively. Thus our participant pool has a smaller percentage of White and Black participants and a larger percentage of Asian participants compared to the general U.S. population which has 72.3%, 12.7%, and 5.6% respectively of these races. 12.6% of our participants reported themselves as "Hispanic or Latino", as opposed to 18.1% of the overall population. Finally, our partipants are spread across 39 U.S. states.

Limitations Inaccuracy in data broker information revealed via Facebook's advertising platform can arise from two sources: (i) inaccuracy in the information collected by data brokers, and (ii) inaccuracy in linking data broker information to Facebook accounts. Thus, our methodology measures the cumulative effect of these two sources of inaccuracy, providing an upper bound on the inaccuracy of information collected by data brokers. However, as discussed in Section 2, Facebook only uses unique identifiers such as phone numbers and email addresses to match data broker information [34], thus indicating that the inaccuracy introduced due to the linking process is likely limited. Thus, our upper bounds are likely tight.

Additionally, the information exposed by Facebook via its advertising platform may only cover a subset of information held by data brokers. However, recent reports have stated that Facebook receives "all available marketing segments offered for [each] user" from data brokers [34]. Moreover, since our methodology relies on Treads to reveal user attributes, not all data broker attributes of each user might be successfully revealed (e.g., if some Treads that are not delivered). Finally, our methodology relies on the accuracy of attributes as reported by participants; while each participant is the best source for evaluating the accuracy of their own data, they might misunderstand what certain attributes mean, may lie, or might be subject to fatigue (which we try to limit). Besides, participants might not recall past behavior such as purchases, or imperfectly estimate information such as their financial data.

4.3 Analysis of accuracy

We next analyze the responses from participants in order to understand the accuracy of information collected by data brokers.

Overall accuracy of attributes We begin by analyzing the overall accuracy of the collected data broker attributes, shown in Table 4. We find a largely bimodal distribution: 40.6% of the 1,432 surveyed broker attributes are reported "Completely accurate", while 40.5% of them are reported "Not at all accurate." By contrast, a higher percentage (proportion test, $p < 0.001$) of Facebook attributes are reported "Completely accurate" (51.7% of the 3,464 surveyed attributes). Also, a lower percentage ($p < 0.001$) of Facebook attributes are reported "Not at all accurate" (25.3% of the 3,464 surveyed attributes). This may be due to the significantly larger amount of data that Facebook can directly observe in users' activities, when compared to data brokers. The low accuracy of data broker information has significant implications, given the widespread use of data broker information for various purposes, including for background checks.

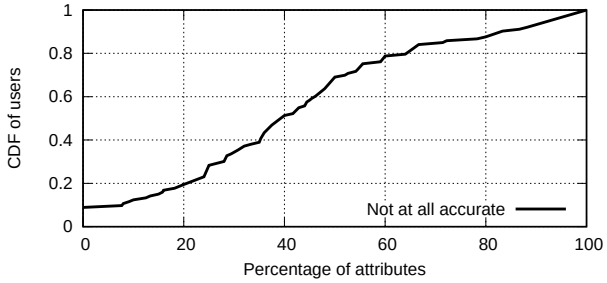


Figure 4: CDF of the fraction of users' data broker attributes rated as "Not at all accurate." We see a wide distribution.

Are inaccurate attributes stale? To study whether data broker attributes are simply inaccurate, or were previously accurate but is now out of date (*stale*), we ask users whether attributes they rated as anything other than "Completely Accurate" were accurate in the past. Specifically, we show the distribution of user responses to the question "Do you think this attribute would have accurately described you at a previous time? (For example, would it have been more accurate last year?)" in Table 5. We see that around 24% of incorrect attributes from data brokers are stale (with users responding with either "Yes" or "Probably yes"). In contrast, the percentage of incorrect attributes from Facebook that are stale is higher (32.3%). Thus, data broker attributes appear to suffer from many inaccuracies, only partially explained by out-of-date data.

User-level accuracy Inaccuracy in a user's attributes might often be because of data brokers' inability to uniquely or correctly identify users from different source databases (e.g., the user might have a very common name). Thus, it might be expected that users' attributes are inaccurate in an all-or-none fashion (i.e., that either almost all a user's attributes are incorrect, or almost all a user's attributes are correct). To study whether this is the case, we show the variation in accuracy by user in Figure 4.

Contrary to our expectation, we see that users typically have a subset of their broker attributes inaccurate, with the percentage of attributes marked "Not at all accurate" varying widely between 0% and 100%, with the median user having around 40% of their broker attributes marked "Not at all accurate." Besides, more than 90% of users who have data broker attributes have a non-zero fraction of their attributes marked "Not at all accurate." This means that inaccuracy of data broker attributes is a problem potentially affecting most people (rather than an unfortunate few).

Variation in accuracy across attributes Finally, to study how the accuracy of broker attributes varies across types or categories

	Data Brokers	Acxiom	Datalogix	Experian	Epsilon	Others	Facebook
Responses	1,432	95	728	55	18	536	3,464
Not at all accurate	40.5% ± 2.5%	27.4% ± 9.0%	42.0% ± 3.6%	21.8% ± 10.9%	55.6% ± 23.0%	42.2% ± 4.2%	25.3% ± 1.4%
Somewhat accurate	13.6% ± 1.8%	14.7% ± 7.1%	14.4% ± 2.6%	20.0% ± 10.6%	0.0% ± 0.0%	12.1% ± 2.8%	15.2% ± 1.2%
Mostly accurate	5.2% ± 1.2%	4.2% ± 4.0%	6.2% ± 1.7%	12.7% ± 8.8%	5.6% ± 10.6%	3.4% ± 1.5%	7.9% ± 0.9%
Completely accurate	40.6% ± 2.5%	53.7% ± 10.0%	37.4% ± 3.5%	45.5% ± 13.2%	38.9% ± 22.5%	42.4% ± 4.2%	51.7% ± 1.7%

Table 4: Aggregated user-reported accuracies for attributes from different data brokers and Facebook. The first column shows the accuracies for attributes from all data brokers taken together. 95% confidence intervals are shown for sample percentages.

	Data Brokers	Acxiom	Datalogix	Experian	Epsilon	Others	Facebook
Responses	833	44	445	28	11	305	1,596
Yes/Probably yes	24.0% ± 2.9%	29.5% ± 13.5%	23.8% ± 4.0%	32.1% ± 17.3%	36.4% ± 28.4%	22.3% ± 4.7%	32.3% ± 2.3%
No/Probably not	76.0% ± 2.9%	70.5% ± 13.5%	76.2% ± 4.0%	67.9% ± 17.3%	63.6% ± 28.4%	77.7% ± 4.7%	67.7% ± 2.3%

Table 5: Aggregated user-responses about the past accuracies for attributes from different data brokers and from Facebook. 95% confidence intervals are shown for sample percentages.

of information, we consider the variation of accuracy across the 15 higher-level "categories" in Facebook's hierarchy (from Section 3.2). Since we have only a relatively small number of responses for some categories, we group together categories that have fewer than 30 responses and which have a common ancestor in the hierarchy. For example, we group together Income , Net worth , and Liquid assets , which have a common ancestor (Demographics → Financial).

We show the distribution of responses in Table 6. We make multiple observations: *First*, we observe mediocre accuracy across all categories. Indeed, even categories involving important financial information such as income/net worth/liquid assets, or home ownership/home type/home value can have a high degree of inaccuracy, with 47.2% and 42.3% of respective attributes marked "Not at all accurate." This has important implications as this information might be used in important situations such as credit or background checks.

Second, we observe that some categories have higher accuracies than others. For example, the Purchase behavior category has a higher percentage of attributes marked "Completely accurate" than the Automotive category (proportion test, $p < 0.001$), and even the Income/Net worth/Liquid assets category ($p < 0.002$). While it is hard to explain why this might be the case, one potential reason is that purchase behavior is often tracked via loyalty cards provided by the data broker themselves [15], thus tracking users more directly.

Third, we find a less bimodal distribution for some categories than others. For example, the Purchase behavior , Income/Net worth/Liquid assets , and Television → Show Genre categories all have a larger percentage of intermediate accuracy levels ("Somewhat accurate" or "Mostly accurate") than the Automotive , and Spending methods categories (with $p < 0.006$ for all six pairwise comparisons). We explain why this might be the case in the following section.

Qualitative analysis of inaccuracy To better characterize how certain attributes are inaccurate, we briefly analyze participants' (free-text) responses to the question "What is incorrect about it?", asked for attributes that they did not mark "Completely accurate." Participants provided responses for 822 out of the 833 inaccurate broker attributes; however, a majority of these responses simply stated that the attribute did not correspond to the participant. We make a number of qualitative observations from the remaining responses (which are thus limited to a subset of all responses).

First, participants often reported errors of degree for categories like Purchase behavior , Income/Net worth/Liquid assets , and Television → Show Genre . For example, for attributes such as "food enthusiast" or "gadget enthusiast" that fall under the Purchase behavior category, participants responded that they might like food or occasionally purchase gadgets, but not to the extent of an enthusiast. Similarly, participants indicated that they only occasionally watched shows of a particular television show genre. For attributes under the Income/Net worth/Liquid assets category, which typically specify a range of income or net worth, participants indicated that the ranges were incorrect by different margins (ranging from "close" to "an order of magnitude below"). This explains why such categories might have less bimodal distributions compared to other categories such as Automotive , and Spending methods , which typically have attributes that are more likely to incur binary errors (e.g., make of car owned).

Second, we observed some anecdotal reasons from participants explaining how particular attributes were stale (i.e., accurate in the past but no longer so). Some responses indicated users changing their purchase behavior in response to changed life circumstances: two participants had purchased particular children's food items when they had children, or cared for children in the past. Similarly, a few participants had purchased pet products when they owned a pet. Finally, one participant had stopped buying a particular kind of frozen food in order to be more healthy. Other responses indicated reasons for changes in finance-related information such as home value (no longer accurate due to appreciation), or particular kinds of investments (participant no longer had that kind of investment).

Third, for categories like Income , Net worth , Liquid assets , or Home value , participants reported having both higher and lower values compared to their data broker attributes (which show a particular range of values for that category). While two participants reported having higher and lower incomes each, six participants reported having lower values for net worth. Nine participants reported having fewer liquid assets (of which two reported that their family

Category	Total	Not at all accurate	Somewhat accurate	Mostly accurate	Completely accurate
Purchase behavior	541	26.1% ± 3.7%	18.7% ± 3.3%	7.0% ± 2.2%	48.2% ± 4.2%
Income/Net worth/Liquid assets	53	47.2% ± 13.4%	20.8% ± 10.9%	5.7% ± 6.2%	26.4% ± 11.9%
Home Ownership/Home type/Home value	111	42.3% ± 9.2%	6.3% ± 4.5%	1.8% ± 2.5%	49.5% ± 9.3%
Automotive	252	60.7% ± 6.0%	8.7% ± 3.5%	4.0% ± 2.4%	26.6% ± 5.5%
Spending methods	141	35.5% ± 7.9%	6.4% ± 4.0%	2.1% ± 2.4%	56.0% ± 8.2%
B2B Company size/Industry	48	66.7% ± 13.3%	6.2% ± 6.8%	8.3% ± 7.8%	18.8% ± 11.0%
Charitable donations	27	40.7% ± 18.5%	18.5% ± 14.7%	3.7% ± 7.1%	37.0% ± 18.2%
Financial → Investments	50	56.0% ± 13.8%	14.0% ± 9.6%	4.0% ± 5.4%	26.0% ± 12.2%
Television → Show Genre	119	37.8% ± 8.7%	21.0% ± 7.3%	5.9% ± 4.2%	35.3% ± 8.6%
Behaviors → Travel	38	39.5% ± 15.5%	10.5% ± 9.8%	5.3% ± 7.1%	44.7% ± 15.8%

Table 6: Aggregate user-reported accuracies for attributes corresponding to different "categories" of information. The first column shows a description of the different categories of information, while the remaining columns show the aggregate distribution of user responses. 95% confidence intervals are shown for sample percentages.

might collectively have a matching amount of liquid assets), while three participants reported having more liquid assets. Finally, two participants reported having homes with higher values.

Fourth, we found some egregious errors (either in magnitude or sensitivity). For example, one unemployed student was inferred to be a C-Suite (i.e., top level) executive, while another two unemployed participants (one of whom was a student) were inferred to be corporate executives. Finally, a few participants who were non-drinkers were inferred to be likely to buy alcoholic beverages.

5 DISCUSSION

Implications of findings Recent policy is moving in the direction of requiring greater transparency and accuracy of data collection; for example, the principles of the General Data Protection Regulation (GDPR) in the European Union require organizations collecting personal data to ensure "lawfulness, fairness and transparency" and to ensure "the personal data is accurate and up-to-date, having regard to the purposes for which it's processed, and correct it if not" [43]. In this light, the findings in this paper about the coverage and lack of accuracy of information collected by data brokers calls for greater efforts from the brokers to (a) allow people to control and correct what data is collected about them, and (b) warn consumers of their data about cases where the data may not be accurate.

Future applicability of methodology Facebook removed partner categories as of October 2018 [4], meaning that it is no longer possible to study the coverage and accuracy of data brokers via Facebook's advertising platform. However, our methodology is still applicable on other platforms such as Twitter, which offers similar categories sourced from Acxiom and Oracle Data Cloud [32].

In addition, our methodology could also be used to study the coverage and accuracy of information internally gathered by online advertising platforms such as Facebook, Google, and Twitter, themselves. The existing transparency mechanisms of these platforms, revealing collected user information, have been shown to reveal an incomplete view to users [2, 17, 54]. Thus, our methodology for studying accuracy using Treads could potentially reveal the accuracy of the data that these platforms have actually collected.

Ethics All our experiments were performed as per community ethical standards. Our experiments measuring the coverage of data brokers only rely on aggregate statistics about users, and thus do not involve collection of data about individual users.

For our experiments measuring the accuracy of data broker information (approved by Northeastern University's IRB), we take multiple precautions to ensure the privacy of participants. *First*, our extension stores inferred attributes about participants locally (using their browser's local storage), and does not collect any personally identifying information from users. *Second*, participants are only required to answer questions they are comfortable answering; any attributes that participants are not comfortable answering questions about are not shared with our server. *Third*, we store the completed surveys in anonymized form. *Fourth*, users who unexpectedly see ads that explicitly state or imply their personal attributes might consider this a privacy violation. We address this concern by targeting the Treads that we run only to participants who have read our consent forms and signed up for our study, and obfuscating the targeting information in these Treads (as opposed to explicitly stating the targeting information directly).

6 CONCLUSION

The extent of data collection by online sites and offline data brokers, and the extent of linkage between online and offline identities of users are both crucial inputs to ongoing debates about user privacy. This paper used Facebook's advertising platform to (a) measure for the first time the actual extent of linkage between online (Facebook) identities and offline identities (as compiled by data brokers), (b) measure for the first time the coverage of data brokers from seven countries in a fine-grained manner, and (c) measure the accuracy of information from data brokers for U.S.-based users. Taken together, our analysis and results of the fine-grained coverage of data brokers provide new insights into the workings of data brokers, that can feed into ongoing privacy debates. We hope this paper serves as a starting point for more such analyses in the future.

7 ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their helpful comments. This research was supported in part by the Data Transparency

Lab, by NSF grant CNS-1616234, by the French National Research Agency (ANR) through the ANR-17-CE23-0014 grant, by the ERC Advanced Grant "Foundations for Fair Social Computing" (No. 789373), and by a UMIACS contract under the partnership between the University of Maryland and DoD. Additionally, Elissa Redmiles acknowledges support from the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 1322106 and a Facebook Fellowship.

REFERENCES

[1] Facebook Algorithms and Personal Data. *PewResearchCenter*, 2019. http://www.pewinternet.org/2019/01/16/facebook-algorithms-and-personal-data/.

[2] A. Andreou, G. Venkatadri, O. Goga, K. P. Gummadi, P. Loiseau, and A. Mislove. Investigating Ad Transparency Mechanisms in Social Media: A Case Study of Facebook's Explanations. *NDSS*, 2018.

[3] ACS Demographic and Housing Estimates (2017). https://factfinder.census.gov/faces/tableservices/jsf/pages/productview.xhtml?pid=ACS_17_1YR_DP05&prodType=table.

[4] About Partner Categories. https://web.archive.org/web/20180911174110/https://www.facebook.com/business/help/298717656925097.

[5] Acxiom. https://www.acxiom.com/.

[6] Acxiom Data Catalog. http://www.stephenfortune.net/projects/puppetry/data_products_catalog.pdf.

[7] Acxiom Global Coverage. https://www.acxiom.com/news/acxiom-launches-global-data-navigator-tool-offering-marketers-visibility-into-global-audiences/.

[8] Acxiom Healthcare Portfolio. https://www.acxiom.com/news/acxiom-expands-healthcare-solutions-portfolio-new-patients-insights-package/.

[9] Acxiom InfoBase. https://www.acxiom.com/what-we-do/infobase/.

[10] Acxiom transparency website. https://aboutthedata.com/.

[11] Advanced Targeting and Placement. https://developers.facebook.com/docs/marketing-api/targeting-specs.

[12] American Fact Finder. https://factfinder.census.gov/faces/nav/jsf/pages/index.xhtml.

[13] T. Bergin. How a data mining giant got me wrong. https://www.reuters.com/article/us-data-privacy-acxiom-insight/how-a-data-mining-giant-got-me-wrong-idUSKBN1H513K.

[14] M. A. Bashir, U. Farooq, M. Shahid, M. F. Zaffar, and C. Wilson. Quantity vs. Quality: Evaluating User Interest Profiles Using Ad Preference Managers. *Proceedings of the Network and Distributed System Security Symposium (NDSS 2019)*, 2019.

[15] J. Constine. Datalogix Raises $25M to Pump Juicy Offline Purchase Data Into Google and Facebook. https://techcrunch.com/2013/04/25/datalogix-offline-purchase-data/.

[16] J. G. Cabañas, A. Cuevas, and R. Cuevas. Unveiling and Quantifying Facebook Exploitation of Sensitive Personal Data for Advertising Purposes. *27th USENIX Security Symposium (USENIX Security 18)*, USENIX Association, 2018.

[17] A. Datta, M. C. Tschantz, and A. Datta. Automated Experiments on Ad Privacy Settings: A Tale of Opacity, Choice, and Discrimination. *PETS*, 2015.

[18] K. Damer. What is your dropout / attrition rate in longitudinal studies? http://help.prolific.ac/getting-started-launching-your-first-study/what-is-your-dropout-attrition-rate-in-longitudinal-studies.

[19] Data Broker Axciom's Flawed Transparency Effort. *MIT Technology Review*. https://www.technologyreview.com/s/519111/data-broker-axcioms-flawed-transparency-effort/.

[20] Data Brokers: A Call for Transparency and Accountability. https://www.ftc.gov/reports/data-brokers-call-transparency-accountability-report-federal-trade-commission-may-2014.

[21] Epsilon. https://www.epsilon.com.

[22] Epsilon U.S. Coverage. https://us.epsilon.com/data-driven-marketing-solutions/people-based-marketing-data-solution.

[23] Experian. https://www.experian.com.

[24] Experian Corporate Factsheet. https://www.experian.com/corporate/experian-corporate-factsheet.html.

[25] Experian Free Credit Report. https://www.experian.com/consumer-products/free-credit-report.html.

[26] Experian Services Catalog. https://www.experian.com/assets/data-university/brochures/ems-list-services-catalog.pdf.

[27] I. Faizullabhoy and A. Korolova. Facebook's Advertising Platform: New Attack Vectors and the Need for Interventions. *CoRR*, https://arxiv.org/abs/1803.10099, Workshop on Technology and Consumer Protection (ConPro), 2018.

[28] Facebook Ads Preferences. https://www.facebook.com/ads/preferences.

[29] Facebook Advertising. https://www.facebook.com/ads/create/.

[30] D. Garcia, Y. M. Kassa, A. Cuevas, M. Cebrián, E. Moro, I. Rahwan, and R. Cuevas. Facebook's gender divide. *CoRR*, https://arxiv.org/abs/1710.03705, 2017.

[31] Google AdWords. https://adwords.google.com/.

[32] Introducing partner audiences (Twitter). https://blog.twitter.com/marketing/en_us/a/2015/introducing-partner-audiences.html.

[33] A. Korolova. Privacy Violations Using Microtargeted Ads: A Case Study. *Journal of Privacy and Confidentiality*, 3(1), 2011.

[34] K. Leetaru. The Data Brokers So Powerful Even Facebook Bought Their Data - But They Got Me Wildly Wrong. Forbes, 2018. https://www.forbes.com/sites/kalevleetaru/2018/04/05/the-data-brokers-so-powerful-even-facebook-bought-their-data-but-they-got-me-wildly-wrong/.

[35] A. Matheus, M. Yelena, W. Ingmar, and B. Fabricio. Using Facebook Ads Audiences for Global Lifestyle Disease Surveillance: Promises and Limitations. *ACM WebSci*, 2017.

[36] T. Minkus, Y. Ding, R. Dey, and K. W. Ross. The City Privacy Attack: Combining Social Media and Public Records for Detailed Profiles of Adults and Children. *ACM COSN*, 2015.

[37] C. R. Miller. I Bought a Report on Everything That's Known About Me Online. https://www.theatlantic.com/technology/archive/2017/06/online-data-brokers/529281/.

[38] Mapping, and Sharing, the Consumer Genome. http://www.nytimes.com/2012/06/17/technology/acxiom-the-quiet-giant-of-consumer-database-marketing.html.

[39] Oracle Data Cloud. https://www.oracle.com/applications/customer-experience/data-cloud/.

[40] Oracle Data Cloud Data Directory. http://www.oracle.com/us/solutions/cloud/data-directory-2810741.pdf.

[41] E. Peer, L. Brandimarte, S. Samat, and A. Acquisti. Beyond the Turk: Alternative platforms for crowdsourcing behavioral research. *Journal of Experimental Social Psychology*, 70, 2017.

[42] Predictably inaccurate: The prevalence and perils of bad big data. https://www2.deloitte.com/insights/us/en/deloitte-review/issue-21/analytics-bad-data-quality.html.

[43] Principles of the GDPR. https://ec.europa.eu/info/law/law-topic/data-protection/reform/rules-business-and-organisations/principles-gdpr_en.

[44] Prolific - Home Page. 2018. https://prolific.ac/.

[45] A. Rao, F. Schaub, and N. M. Sadeh. What do they know about me? Contents and Concerns of Online Behavioral Profiles. *CoRR*, abs/1506.01675, 2015.

[46] L. Sweeney. Discrimination in Online Ad Delivery. *SSRN*, 2013.

[47] N. Singer. Acxiom Lets Consumers See Data It Collects. https://www.nytimes.com/2013/09/05/technology/acxiom-lets-consumers-see-data-it-collects.html.

[48] T. Speicher, M. Ali, G. Venkatadri, F. N. Ribeiro, G. Arvanitakis, F. Benevenuto, K. P. Gummadi, P. Loiseau, and A. Mislove. On the Potential for Discrimination in Online Targeted Advertising. *FAT**, 2018.

[49] Shutting Down Partner Categories. https://newsroom.fb.com/news/h/shutting-down-partner-categories/.

[50] M. C. Tschantz, S. Egelman, J. Choi, N. Weaver, and G. Friedland. The accuracy of the demographic inferences shown on Google's Ad Settings. *WPES*, 2018.

[51] United States ZIP Codes. https://www.unitedstateszipcodes.org/.

[52] G. Venkatadri, Y. Liu, A. Andreou, O. Goga, P. Loiseau, A. Mislove, and K. P. Gummadi. Privacy Risks with Facebook's PII-based Targeting: Auditing a Data Broker's Advertising Interface. *IEEE S&P*, 2018.

[53] G. Venkatadri, A. Mislove, and K. P. Gummadi. Treads: Transparency-Enhancing Ads. *HotNets*, 2018.

[54] C. E. Wills and C. Tatar. Understanding What They Do with What They Know. *WPES*, 2012.

[55] E. Zagheni, I. Weber, and K. Gummadi. Leveraging Facebook's Advertising Platform to Monitor Stocks of Migrants. *Popul. Dev. Rev.*, 43(4), 2017.

"Data Strikes": Evaluating the Effectiveness of a New Form of Collective Action Against Technology Companies

Nicholas Vincent
Northwestern University
Evanston, IL, USA,
nickvincent@u.northwestern.edu

Brent Hecht
Northwestern University
Evanston, IL, USA,
bhecht@northwestern.edu

Shilad Sen
Macalester College
Saint Paul, MN, USA,
ssen@macalester.edu

ABSTRACT

The public is increasingly concerned about the practices of large technology companies with regards to privacy and many other issues. To force changes in these practices, there have been growing calls for "data strikes." These new types of collective action would seek to create leverage for the public by starving business-critical models (e.g. recommender systems, ranking algorithms) of much-needed training data. However, little is known about how data strikes would work, let alone how effective they would be. Focusing on the important commercial domain of recommender systems, we simulate data strikes under a wide variety of conditions and explore how they can augment traditional boycotts. Our results suggest that data strikes can be effective and that users have more power in their relationship with technology companies than they do with other companies. However, our results also highlight important trade-offs and challenges that must be considered by potential organizers.

CCS CONCEPTS

• **Human-centered computing~Human computer interaction (HCI)** • Information systems~Recommender systems

KEYWORDS

Data strikes, recommender systems, online collective action

ACM Reference format:

Nicholas Vincent, Brent Hecht and Shilad Sen. 2019. "Data Strikes": Evaluating the Effectiveness of a New Form of Collective Action Against Technology Companies. In *Proceedings of WWW '19: The Web Conference (WWW '19), May 13, 2019, San Francisco, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3308558.3313742

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05
https://doi.org/10.1145/3308558.3313742

1 Introduction

Large technology companies are facing a growing wave of public criticism. Just in the last year, these companies have been condemned for a wide range of practices, including those related to privacy [13, 55], harassment [20], addiction [10], effects on democracy [12], and automation [1]. The breadth and scale of the public concerns about tech companies has even led to the popularization of the term "Big Tech" [25], an adaptation of the terms "Big Oil" and "Big Tobacco" [3, 28].

However, these same companies that anger the public often are *dependent on the public* in new ways. Specifically, in addition to needing users and customers to generate revenue, *tech companies often rely on the public's "**data labor**" [44] to power mission-critical **intelligent technologies***. For example, Google requires user clicks to train its ranking algorithm [45]. Similarly, the highly-profitable recommender systems employed by companies like Amazon and Netflix require large amounts of data from users (i.e. ratings, clicks, and views) [17, 53].

Seen through the lens of the public's concerns about tech companies, these companies' dependence on user data to fuel their intelligent technologies can be understood as a potentially powerful source of new leverage for the public. To help the public action this leverage, several authors have proposed the notion of "**data strikes**" (e.g. [2, 39, 44, 52]), in which users halt their data labor [44]. The basic logic that motivates data strikes is straightforward: if users withhold their data labor from a tech company, some of the company's essential services will suffer, and this would then force the company to make concessions that are desired by the public. These concessions could range from improved privacy policies to profit sharing [2, 15].

Despite the growing discussion around data strikes, little is known about how this new type of collective action would work or about how data strikes relate to standard forms of collective action like traditional consumer **boycotts** (a type of collective action classified as political consumption [32, 42]). Additionally, as data strikes increasingly enter the realm of feasibility (see below), there is little empirical information about how effective data strikes could be, let alone the data strike configurations that would be most effective. Activists seeking to organize a data strike have no guidance regarding the number of users that would need to join them, the kinds of services most vulnerable, the types of users that would allow them to be most successful, or even whether strikes can be successful at all. Similarly, tech

companies are not aware of the potential damage that could be inflicted through data strikes.

This paper seeks to improve our basic understanding of data strikes and provide much-needed empirical information about their effectiveness. We first situate data strikes in relationship to traditional boycotts, in which a user stops patronizing a company entirely. Through the introduction of a lightweight framework that (partially) describes **collective action** in a technology company context, we highlight that most traditional boycotts against a company operating data labor-dependent intelligent technologies will implicitly also include a data strike, but that data strikes can also occur independently from boycotts. For example, a consumer who continues to purchase products from an online retailer could engage in a data strike by using private browsing windows and not providing product ratings.

Next, focusing on the domain of recommender systems, a family of intelligent technologies that are critical drivers of revenue [17, 51], we introduce a novel evaluation procedure for understanding collective action campaigns against technology companies. Our procedure uses a metric called **surfaced hits,** which can capture the effects of both a traditional boycott and a data strike. Leveraging surfaced hits, we examine how model performance changes depending on 1) the size of the participating group, 2) whether the participating group is a random group of users or a homogeneous group of users who share some characteristics (e.g. women, people interested in documentary movies), and 3) whether or not the group is conducting an independent data strike or are doing so as part of a traditional boycott.

Our results confirm that users' **data labor power** – which is mostly unique to online platforms and is manifest in a data strike – provides users with a new source of leverage in their relationships with technology companies. For small recommenders and in specific product spaces, this added leverage can be particularly substantial. A moderately-sized data strike alone – even when not part of a traditional boycott – can significantly harm the performance of a recommender system. Indeed, for moderately-sized data strikes, we observe recommender accuracy decreasing to the levels that defined the state-of-the-art in recommender algorithms in 1999. This power comes from the reduced performance for both non-striking users (who receive recommendations trained on less data) and striking users themselves (who receive recommendations that are not personalized). Additionally, our work shows that data strikes that occur as part of a traditional boycott add data labor power to the standard **consumer power** from a boycott, increasing the overall power of the collective action campaign.

Finally, our work also highlights that data strikes that are not part of a traditional boycott represent a *fundamentally new type of collective action*, one in which the *barrier to entry is much lower than in a boycott*. Most notably, we observe that data strike participants can substantially reduce the utility of a recommender system without sacrificing access to the underlying products and services. Given that it has proven difficult for people with limited financial resources to participate in political consumption activities like boycotts [32], the demonstrated effectiveness of data strikes could democratize access to these activities (e.g. users who cannot afford to use expensive alternatives to online platforms can still strike). This is analogous to an offline boycott in which a user who cannot afford expensive pizza could still participate in collective action against a local low-price pizza chain while continuing to buy their products.

Below, we adopt a standard structure to motivate, explain, and expand on our findings. We first cover related work, then discuss methods, followed by results. We close with a discussion of the issues identified in our results and by highlighting limitations.

2 Related Work

In this section, we describe how this research draws motivation from four areas in particular: the growing discussion related to data strikes, research on the relationships between tech companies and volunteer-created content, studies that generally seek to quantify the financial value of user data, and studies that looks specifically at ways to manipulate recommender system outputs.

2.1 Data Strikes

This research was most directly motivated by growing calls for collective action campaigns that force changes in technology platforms by leveraging the value of user data to these platforms [2, 11, 16, 23, 37, 44]. These growing calls use different, potentially conflicting framings of *data as capital* or *data as labor* [1]. With regard to the former (data as capital), collective action is framed in terms of a boycott, in which users stop their consumptive activities (e.g. purchasing products through a web platform, using a social media platform) which in turn prevents the flow of their capital (data, related revenue like advertising revenue) to the platform. This framing is exemplified by very recent boycotts put into practice against Facebook (e.g. [42]). The data labor view suggests that data "unions" should protect the interests of those who produce data (i.e. users) [2, 16, 44]. Just as traditional labor unions have implemented (and threatened) strikes to gain leverage when collectively bargaining, Lanier and others [16, 23] have written that data unions might similarly engage in a "strike". These authors point out that users can leverage their data in ways that resemble both traditional boycotts and strikes.

The diverse understandings of collective action campaigns that use data leverage needed to be integrated in order to make these campaigns concrete enough to simulate. Below in the Framework section, we enumerate one possible integration and use the corresponding framework to inform the design of our experiments.

Ideas about collective action campaigns that use data leverage often imagine a future in which people can "delete their data" from an online platform, and this future is becoming increasingly realistic thanks to developments and discussions in the policy domain. For instance, the European Union recently adopted the General Data Protection Regulation (GDPR) [30],

which includes a provision ensuring the right to erasure. Barring special circumstances (e.g. data critical to public health research), individuals covered by the GDPR will have the right to request that their personal data be deleted. As such, the GDPR potentially empowers activists to engage in more powerful data labor-related collective action than previously possible, in particular by erasing old data instead of just stopping the flow of new data. While it remains to be seen how often and how effectively the right to erasure will be used in practice – and how it might apply to campaigns that use data leverage specifically – the inclusion of this right highlights a large shift in regulatory practices towards data usage. The GDPR could trailblaze the way for similar or even stronger provisions by other regulatory bodies (e.g. California's State Government [6]).

In keeping with this policy trajectory and with the typical vision of collective action campaigns that use data leverage, we simulate campaigns in which people can "delete their data" when they participate. However, our approach also applies to other contexts, for instance domains in which there is *no existing data* like reviews about a new television show or location data used to predict traffic (and less directly to contexts in which participants cannot delete data, but do not execute new data labor). Furthermore, very recent research suggests that simple tools like browser extensions may help web users successfully join web-based collective action campaigns with low overhead for the user, which could help stop the flow of implicit behavioral data [39].

Finally, it is important to note that the social science literature can give some guidance as to how large one could expect the campaigns we examine to be. Data from Europe and the U.S. found that between 28% and 35% of consumers had engaged in an act of political consumption [32, 42], which includes either boycotting or "buycotting" (aligning one's purchases with a company that is perceived to align with one's political preferences).

2.2 Tech Companies and Volunteer-Created Content

A related area of research that also helped motivate this study is work that has sought to understand the dependence of technology companies on volunteer-created content like Wikipedia articles. McMahon et al. [40] showed that Google search effectiveness drops substantially when Wikipedia links are silently removed from search results, highlighting how important Wikipedia is to the success of search engines. Outside of Google's relationship to Wikipedia, Vincent et al. found that Stack Overflow and Reddit receive substantial benefits from Wikipedia in the form of impactful links and references [56]; they showed that these benefits come in the form of both increased engagement from users and advertising revenue. Although not originally intended as such, these studies can be seen as simulating a form of data-related campaign in which companies are somehow prevented from using Wikipedia content. Such campaigns would be highly unlikely given Wikipedia's content license [58] and other factors, but the design of these studies helped to inform our methodological approach outlined below.

2.3 Financial Value of Data

This work was more generally motivated by a broad body of literature seeking to understand the financial value of data and highlighting the importance of this understanding. This research includes efforts to provide individual users with transparency into the value they create, such as the Facebook Data Valuation Tool created by González Cabañas et al. [18], as well as efforts to broadly understand the value of that data at a macroeconomic scale (e.g. the value of Wikipedia to GDP statistics [4]). On the policy front, the World Economic Forum has identified data as a new "asset class" and suggests that thinking about data economics demands a new understanding of the personal data ecosystem [49].

2.4 Recommender System Manipulation

Within the recommender system literature, there has been research into various ways recommender systems might be manipulated. For instance, prior work has examined how recommender systems might be "shilled", i.e. misled so as to promote a particular product (e.g. [7, 36]). Like potential strikes, shilling attacks are an adversarial approach to manipulating the outputs of a recommender. As we explain below, our experiments specifically focus on campaigns that withhold data, so findings related to shilling are not directly applicable. However, in practice collective action participants might be able to adopt techniques from shilling, in which case this body of literature may be useful to both users and the companies against which they seek to gain leverage.

Recent research from Wen et al. explored recommender performance under conditions in which users filter some portion of their preference history to increase privacy and/or recommender accuracy [57]. Specifically, the authors found that users can filter out some of their preference history from the recommender while maintaining, or even improving, performance for implicit recommendations. This research has direct implications for data-related collective action: although users might be able to perform a "partial strike" by deleting some of their data without suffering decreased performance, these partial strikes are unlikely to be very effective, as in some cases, partial strikes by all users may improve population-level performance. In our experiments, we focus on directly simulating conditions in which users withhold *all* their data, and we also explicitly consider both data strikes and traditional boycotts. However, exploring the interplay between strikes, boycotts and data-filtering tools will be an important area of future research that is mutually beneficial to both problem contexts. In particular, the data filtering interfaces described by Wen et al. could be another outlet for users to actuate data strikes.

3 FRAMEWORK

As mentioned above, although collective action campaigns that use data leverage against technology companies have been discussed as a theoretical possibility, they have not been characterized in detail. Indeed, in the context of collective action against technology companies, the distinction between data strikes, boycotts, and combinations of the two can be unclear.

In order to simulate data strikes, we need to first concretely define data strikes and their relationship to traditional boycotts. To do so, we turned to the divergent theoretical underpinnings of the boycott and strike terms. In a boycott, participants are *consumers* who cut off the flow of an asset (e.g. money from purchases) to a firm. In a strike, participants are *laborers* who stop performing work for a firm. Users of an online platform can therefore boycott the platform by refusing to use the platform as consumers (e.g. not buying from an e-commerce site, not visiting a news site or video site, etc.) and strike against the platform by refusing to provide data (e.g. deleting data, preventing the flow of new data by using private browsing features or other privacy techniques like ad blockers or Mozilla's new Facebook Container [14]).

In most cases, *boycotts against tech companies implicitly include a data strike.* For instance, users of a video platform like YouTube who boycott (refuse to visit the website) are also implicitly conducting a data strike by cutting off the flow of behavioral data like views, likes, and comments. However, *it is often possible to participate in a data strike without boycotting the platform.* This occurs if someone continues to access a website but withholds data using privacy-preserving techniques (e.g. private browsing), leverages data management options made possible through data protection regulation, or – critically for our context – *refuses to comment on products, rate products, or review products.* The inverse, a boycott without a data strike, is less ecologically valid in the context of our work. Someone who boycotts Amazon products is unlikely to submit product reviews and ratings. While there are more nuanced situations in which this could occur, in this paper, we simulate boycotts in concert with a strike.

To put the nuanced relationship between boycotts and data strikes as defined above into better context, we consulted the literature to identify the various specific means by which these types of collective action campaigns could affect company revenue (e.g. [2, 44]). We identified four such pathways to revenue impacts:

- The **direct data labor effects** (e.g. algorithmic performance decreases leading to loss in sales)
- The **indirect data labor effects** (e.g. because algorithmic performance goes below some threshold, users quit the platform leading to loss in sales) [54].
- The **direct consumer effects** (e.g. people stop buying products or viewing ads) [32, 42].
- The **indirect consumer effects** (e.g. a large number of customers stop buying products or viewing ads, so there is a loss of economy of scale advantages) [50].

The **consumer effects** above (which make up **consumer power**) are those that exist in traditional boycotts and have been felt by targeted businesses since well before intelligent technologies came into common use. The **data labor effects** (which make up **data labor power)** are specific to collective action campaigns against companies that use data-hungry intelligent technologies. While a traditional boycott only includes direct and indirect consumer effects, a simultaneous data strike and boycott includes all four of the above components.

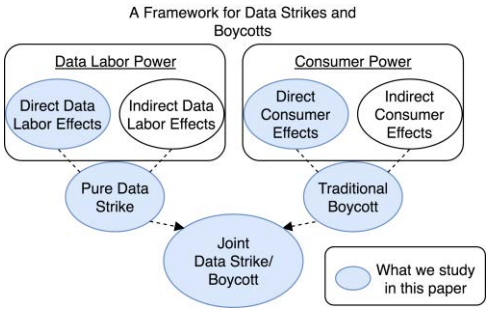


Figure 1: Graphical summary of our framework for defining data strikes and boycotts. Blue indicates aspects on which we focus.

This simple framework, depicted in Figure 1, provides a much-needed lens into the ontology of these types of collective action campaigns, but it also highlights that researchers – especially those outside a technology platform – can only simulate a portion of the effects of data strikes and boycotts. For instance, without exact sales numbers, both the direct and indirect sales effects are very difficult to study externally. Similarly, the indirect long-term effects of reduced recommender performance would be difficult to capture both for researchers external or internal to a platform (although there is at least one case of this being done internally in the search literature using an A/B test framework [54]).

Fortunately, in this paper, we show by focusing on the direct data labor effects while still considering the direct consumer effects, we can still learn a great deal about collective action campaigns that use data leverage. As we discuss below, our results support the effectiveness of data strikes, suggesting the unique relationship between users and technology companies can empower users beyond what would be the case in other contexts. We also discuss how our results can be interpreted as a *lower bound* for the effects of data strikes and boycotts against tech companies because we cannot measure indirect effects.

As discussed above, the data strikes we simulate correspond to a person "deleting their account" and using the recommender as a "guest", with no account history. Similarly, the boycotts we simulate (which include a joint strike) correspond to someone deleting their account and not using the system at all. In both cases, users not participating in a strike or boycott receive recommendations from a model that has been trained without strikers' or boycotters' data. As we discuss in Limitations, there

are many other configurations one can imagine - especially those related to strikes that do not involve the deletion of past data and play out over time - and we believe these to be important directions of future work.

4 METHODOLOGY

In this section, we first describe aspects of our methods that were consistent across all our experimental configurations. We then describe the two broad types of collective action campaigns we simulated: "general" groups comprised of randomly selected users and "homogenous groups" of users who share some characteristic (e.g. power users, fans of comedy movies).

4.1 Design of Experiments

In each experiment, we evaluate recommender systems under a variety of simulated data strike and boycott conditions. While the campaigning groups differ by experiment, the basic methods are the same.

4.1.1 Datasets. Our primary dataset was MovieLens-1M (ML-1M), which consists of 1 million "1 to 5 star" ratings for 3,706 movies provided by 6,040 users with self-reported demographic data [22]. To better understand performance against large recommenders, we additionally performed experiments with the much newer MovieLens-20M (ML-20M) dataset [22], which contains 20x more ratings but no demographic data about users. The MovieLens datasets have been hugely influential and have been central to recommender system research for decades [22].

4.1.2 Algorithm Choice and Implementation. For each experiment, we focus on the well-known and high-performing Singular Value Decomposition (SVD) recommender algorithm [41]. As validation, we also performed a smaller set of ML-1M experiments with an older and mathematically distinct algorithm: item-based K-Nearest Neighbors (k-NN) [48] adjusting for item and user baselines as described by Koren [33]. Both algorithms are implemented in the open-source Python library Surprise [27], which we extended for our experiments. All code used for our experiments and analyses is available for replication and extension on GitHub[1]. We validated the accuracy of the SVD implementation by ensuring results were comparable to published results on the ML-1M dataset [26, 34, 38, 46]. These successful comparisons are summarized in the linked GitHub repository.

4.1.3 Evaluation Procedure. We evaluated the recommender with five-fold cross-validation; for each evaluation fold, 20% of total data is available for testing and each rating is tested in exactly one fold. While data that is held out because of a simulated campaign cannot be used for training, it can be used for testing. This means we can consider results from the perspective of striking users, who will receive non-personalized recommendations which are based on each movie's average rating. This is a standard baseline for producing recommendations for users who lack personalized data and it is a widely-used way to provide preference information in a non-personalized fashion (e.g. displaying a movie's average rating instead of predicted rating for a given user).

4.1.4 Metrics for Evaluating Strikes and Boycotts. When evaluating the accuracy of explicit rating predictions for recommender systems, one common approach is to measure the error in individual predictions, e.g. through an accuracy metric such as root-mean-squared error (RMSE) which was used for the well-known Netflix Prize [41], or using information retrieval metrics such as gain (NDCG) [5] or precision. Retrieval metrics have been favored in recent years because of their ecological validity [8, 31] (e.g. "Top Ten Movies for You").

However, these standard metrics only capture performance for users who receive recommendations, and do not capture the consumer effects of users leaving due to boycotts. Therefore, this approach is not well-suited to understand boycotts from the perspective of the system owner, because the loss in revenue from boycotting users will not be visible in traditional metrics. For instance, if we simulate an 80% boycott and measure the RMSE of predicted ratings for the remaining 20% of users, the change in RMSE does not account for the users who *left the system entirely*.

To understand the relationship between data strikes and boycotts, it is critical to capture both the direct consumer effects of boycotting users and the direct data labor effects of striking users. To do so, we introduce a new metric, which we call **surfaced hits**. The metric measures the fraction of hits (defined as a rating of at least 4.0, as is common in prior work, e.g. [34]) across an entire group of users (perhaps all users, or non-boycotting users). The underlying assumption, that one hit corresponds to one unit of value for a recommender system, is supported by the widespread use of analytic metrics such as click-through-rate in online systems [17]. A perfect algorithm will surface all hits, and therefore have a surfaced hits value of 1.0. This metric can be effectively viewed as a variant of precision that sums (rather than averages) across all users and sets individual thresholds for precision equal to how many positive ratings each user has. More explicitly,

for each user $\boldsymbol{u}$:
$\boldsymbol{t_u}$ = u's test ratings
$\boldsymbol{n_u}$ = number of true ratings >= 4 in t_u
$\boldsymbol{p_u}$ = top n_u of t_u ordered by predicted rating
$\boldsymbol{h_u}$ = number of true ratings >= 4 in p_u
surfaced hits = sum($\{h_u\}$) / sum($\{n_u\}$)

As stated earlier, for both data strikes and boycotts combined with data strikes, participating users' ratings are withheld from all training data sets. Users who boycott contribute zero hits to the numerator of surfaced hits for their test ratings ($h_u = 0$). In other words, the surfaced hits value is "penalized" by marking all positive ratings for the user as a non-hit. For users in a data strike, we included the user's test ratings in the calculation of surfaced hits, but "penalize" via non-personalized recommendations (i.e. movie averages) because the user's training data is not available due to the strike.

Overall, the surfaced hits metric has three useful properties for studying data strikes and boycotts. First, when users boycott, surfaced hits is reduced proportionally to the number of positive

[1] https://github.com/nickmvincent/surprise_sandbox

ratings in the boycotting group. In other words, if enough users boycott to remove half of all the "hits" in the dataset, surfaced hits will be reduced by at least half. This allows us to understand the effects of strikes and boycotts from within a single reference frame. Second, this metric also accounts for differences in user behavior: a user with 1000 hits in their rating history has 10x the impact of a user with 100 hits in their history. This captures the potentially disproportional economic value of more active users. Third, we can calculate surfaced hits for different subsets of users (e.g. all users, non-striking users, users similar to striking users) to understand the effects of collective action from different perspectives. We verified that these metrics perform very similarly to well-established list metrics including NDCG and precision while at the same time capturing the damage that occurs when boycotting users leave a system (see Section 5.3 and the code repository).

4.2 Campaign Configurations

4.2.1 General campaigns. To get a general understanding of the relationship between campaign size and recommender system performance, we first simulated a series of "general" campaigns with random user selection. In these campaigns, the demographic make-up of the groups approximates the distribution of all users. We selected a sequence of 16 different group sizes ranging from 0.01% of users to 99% of users. For each group size, we randomly selected a group of that size to participate in the campaign. To reduce noise associated with different random configurations, we repeated each of these 16 experiments 250 times with a new random user sample for the ML-1M dataset and 40 times for the ML-20M dataset.

4.2.2 Homogeneous campaigns. We also simulated campaigns executed by "homogeneous" groups defined by shared patterns in rating behavior or shared demographic information. More specifically, we created five types of homogenous campaigns: campaigns by "fans" of specific movie genres, campaigns by three categories of demographic groups, and campaigns defined by rating behavior. We created groups of "fans" for each movie genre by identifying all users who rated at least ten movies of that genre and have an average rating for the genre of four or higher. To simulate demographically-defined campaigns, we created groups based on user-reported demographics, specifically male/female, age bracket, and occupation. For rating behavior campaigns, we created campaigns for "power users," defined as the top 10% of raters and "low frequency" users, the bottom 10% of raters.

For each of the five types of homogeneous groups, we simulated campaigns in which 50% of users within a given group participated. For example, we simulated campaigns with groups such as 50% of all women, or 50% of all comedy movie fans, and so on. Importantly, 50% participation allows us to simulate what happens to similar users who do not participate. In other words, if some women participate in a data strike, what happens to women who do not? We also viewed 50% group participation as more realistic than full participation.

Our homogenous experiments focus on the data labor effects of data strikes. The consumer effects of a homogenous boycott scale with the size of the boycott as measured by the number of positive ratings, and as we show in our general campaign experiments, this effect is substantially larger than that of strikes, but this does not negate a strike's value.

For each homogenous campaign configuration, we performed experiments with 50 sampled groups. We also compared the observed campaign effects to the expected effects for a random campaign with the same number of ratings. In order to obtain a relatively simple estimate of the "expected" effect of a data strike of some size, we computed a quadratic interpolation of our results shown in Figure 2. Our homogeneous experiments only consider the ML-1M dataset due to the lack of demographic information in ML-20M.

5 RESULTS

In this section, we first describe the relationship we observed between recommender performance and campaign size in the case of general (random users) collective action campaigns, focusing on comparing pure data strikes to strikes coupled with traditional boycotts and examining the overall effectiveness of campaigns across datasets. Next, we describe the key findings from our homogeneous group experiments, focusing on the finding that unique homogenous groups, defined demographically or behaviorally, can exert their data labor power to disproportionately affect similar users, indicating the potential for data strikes that target specific preference spaces to boost their effectiveness.

This section uses the surfaced hits metric, described above. We focus on the popular SVD algorithm because the item-based k-NN algorithm behaved similarly in our initial experiments (see GitHub repository).

5.1 General Campaign Experiments

We begin by examining the effect of general data strikes and boycotts (i.e. with random users) from the perspective of the system owner (e.g. Google, Facebook, operators of MovieLens). Next, still focusing on the perspective of system owners, we specifically zoom in on performance changes relative to un-personalized results.

Figure 2 shows the effect of data strikes (blue line) and joint data strikes and boycotts (green line) on surfaced hits (y-axis) across the system for both ML-1M (left column) and ML-20M (right column). As a reminder, a value of 1.0 would correspond to an algorithm that produces perfect ranked lists for every user.

These plots include dotted horizontal lines that provide important context: the black line shows performance of SVD with full access to the dataset (which gives 77.4% of hits), the red line shows the results of "MovieMean," which gives completely un-personalized ratings (movies are ranked in order their mean rating) and the gold line shows the results of completely randomly ranked lists (i.e. worst-case performance). Note the high number of hits associated with random lists: this is because MovieLens users tend to give movies high ratings, so when evaluating even random lists many of the items suggested for a given user will be hits. We address this phenomenon below by

focusing not only on raw changes in surfaced hits, but also on performance *relative to* un-personalized performance, i.e. we zoom in on performance change between the black and red lines.

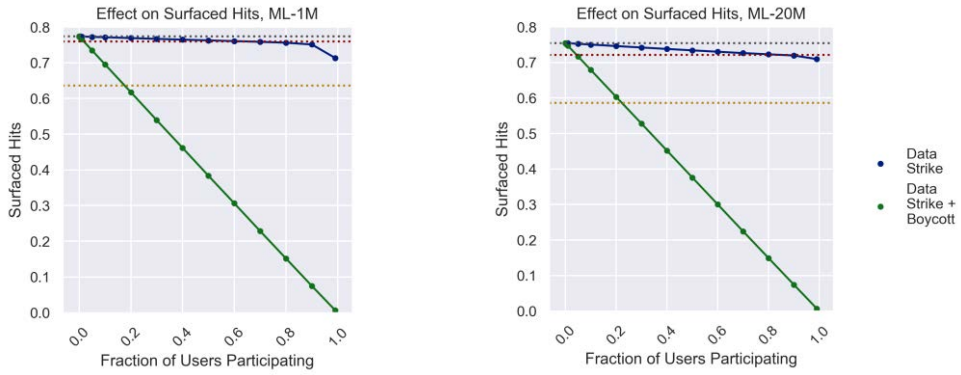


Figure 2: The relationship between campaign size and surfaced hits. Surfaced hits include all users (including strikers and boycotters), and therefore reflect the perspective of the system owner. Dotted horizontal lines provide comparisons: black (uppermost) shows fully personalized SVD, red (middle) shows un-personalized results, and gold (bottom) shows random results.

The most significant trend in Figure 2 is that boycotts are substantially more effective than data strikes. For instance, while a 30% boycott of ML-1M reduces hits to from 77.4% to 53.9%, a 30% data strike only reduces hits by 0.7% to 76.7% (ML-20M, right column, shows a similar trend). Furthermore, for both datasets a boycott of about 20% of users reduces surfaced hits to the amount expected for completely randomized recommendations. This result means that at first glance, the loss in hits caused by users who leave the system strongly outweighs the loss in hits from reduced algorithmic performance. Importantly, this finding does not fully explicate the potential of data strikes, as we will describe below.

We note that the gaps in Figure 2 between un-personalized results (red lines) and fully personalized results (black lines) appear to be small and correspond to a loss of 1.4% of surfaced hits for ML-1M and 3.3% for ML-20M. This reflects the non-linear value of recommender algorithms; the small margin between un-personalized and personalized algorithms corresponds to a large amount of value for a recommender system operator. For example, in Netflix's case, the margin between un-personalized algorithms and personalized algorithms accounts for a 2-4x increase in engagement with recommended items and $1 billion in revenue [17] (see below). Thus, we now specifically focus on the change in performance relative to un-personalized results to inspect how data strikes leverage direct data labor effects to lower recommendation performance towards un-personalized levels.

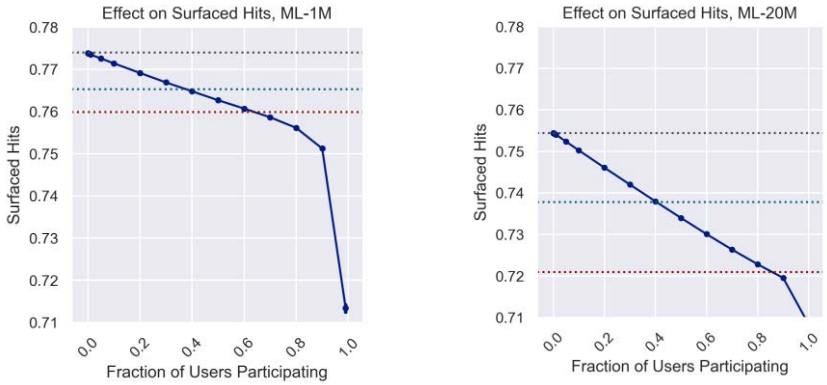


Figure 3: The relationship between data strike size and surfaced hits. Dotted horizontal lines provide comparisons: black (top) shows fully personalized SVD, cyan (middle) shows item-based k-NN (1999), and red (bottom) shows un-personalized results.

Figure 3 zooms in on surfaced hits during a data strike using the same y-axis as Figure 2. Again, the black horizontal line marks personalized performance and the red horizontal line marks un-personalized performance. Additionally, in Figure 3 the horizontal cyan line shows the performance of simple item-based k-NN (which we evaluated with full access to each

dataset), a technique that was introduced in 1999. This context shows the ability of campaigns to essentially set recommender system performance "back in time".

In Figure 3, the potential power of data strikes becomes clearer. The left side of the figure shows that campaigns had substantial effects on recommender performance for ML-1M relative to non-personalized ratings. For instance, a strike with 30% of users (which is a realistic size based on research on political consumption; see Related Work) degrades performance such that users lose roughly half the benefits of personalization. These results also illustrate the power of collective action to potentially negate decades of algorithmic advances. Looking again at MovieLens-1M, a strike by 37.5% of users can roll back hits to a level equivalent to the classic item-based k-NN algorithm introduced in 1999 [48] (cyan dotted line in Figure 3).

The ML-20M results (right column), however, suggest a somewhat more complicated story for recommenders using larger datasets. When the dataset size is increased by a factor of 20, a strike by the same percentage of users becomes somewhat less effective. If 30% of ML-1M strike (1800 users), we see a 50.2% reduction in the benefits of personalization, but if 30% of ML-20M strike (41400 users), it would only cause a 37.0% reduction.

A likely explanation of why the relationship between strike size and strike power differs between ML-1M and ML-20M lies in the two effects of a data strike, the effect on the strikers themselves and the effect on non-strikers. The first effect captures how the removal of the strikers' data lowers performance of the recommender for non-strikers (i.e. the ability of strikers to affect the experience of non-strikers). While we see similar directional relationships between ML-1M and ML-20M, this effect is more pronounced for ML-1M. For instance, at 30% strike participation, we see a 25.4% reduction in personalization for non-striking ML-1M users but just a 4.2% reduction in personalization for ML-20M (analysis in GitHub repo). ML-20M does not see an equivalent reduction in personalization for non-strikers until strike participation rates hit 77%. The explanation for this difference is likely straightforward: ML-20M has more redundant encodings of preference patterns due to its sheer size, so an equivalent percentage of strikers cannot have as much of an effect on the experience of non-strikers.

The second factor driving the strike results is the fact that, during a data strike (rather than boycott), by definition, striking users can still use the system. Because *these users must still receive a ranked list of items* (e.g. to power a Netflix-style interface), what used to be a personalized ranked list must now become a non-personalized ranked list. As noted above (Section 4.1.3), we implement this non-personalized ranked list in the most ecologically valid way possible: using the average movie rating from other users (item mean). As such, for each new user who strikes, the recommender will inch towards item mean by default, regardless of the effect on non-striking users. In other words, *even in the face of large amounts of training data, users can hurt the system by refusing to provide the input data needed to make predictions for themselves.*

Finally, we reiterate that evidence from industry suggests that in commercial systems, a change in surfaced hits has a non-linear value to recommender operators. In other words, the visually-small performance change between the red and black horizontal lines in Figures 2 and 3 may have an outsized effect on platform revenue. As mentioned before, the small surfaced hits improvement due to personalization may correspond to a 2-4x increase engagement with recommended items in other contexts [17]. Similarly, in 2010 YouTube published findings that suggest recommendations add substantial value: almost 30% of video views came from their recommender system and the recommender was the main source of views for most videos [59]. An industry report from consulting company McKinsey estimates that recommender systems account for 35% of Amazon purchases and 70% of Netflix views [24]. Taken together, this means that in many contexts, the revenue effects of data strikes seen above would be magnified relative to their appearance in the figures.

5.2 Homogeneous Campaign Group Experiments

In our homogenous campaign experiments, we examine what would occur if a demographics- or taste-defined group engaged in a data strike (e.g. women or documentary fans). Specifically, for the reasons defined above, we simulate the effects of 50% of the group striking, examining the impact on the remaining 50% in the group, as well as on the recommender overall. We use the term **Similar Users** to describe non-striking members of the striking campaign group (e.g. non-striking women in a strike by women) and **Not Similar Users** for all other non-striking users (e.g. non-women in a strike by women). Then, we define the **Similar User Effect Ratio** as the percent change in surfaced hits for Similar Users divided by the percent change in surfaced hits for Not Similar Users. One challenge with analyzing these homogenous boycotts is that the various groups are very different in size (see Table 1). Therefore, this set of analyses focuses specifically on percent change in surfaced hits, which partially mitigates the challenge in comparing data strikes by groups that are very different in size. Furthermore, when looking at the aggregate effects of homogeneous campaigns, we specifically look at the perspective of non-participating users, as both the direct consumer effects and the effects from striking users seeing un-personalized recommendations are functions of campaign size.

The results from our homogenous campaign experiments show that data strikes may be especially effective at impacting recommendations within targeted topical domains. Specifically, we observe that if homogeneous groups of people strike, non-striking users that share the same characteristic as the homogenous striking group will experience larger reductions in recommender accuracy than other users. For example, striking horror movie fans can make a large movie recommender suffer for other horror fans, potentially giving a competing movie site an opening. A secondary, related observation from these experiments, which we return to at the end of this section, is that homogenous groups' aggregate effects on non-participating users are not consistent: some groups have an outsized aggregate effect relative to their size, and vice versa.

Table 1: Examples of homogenous groups, the number of ratings in each group, percent change in surfaced hits, and the Similar User Effect Ratio.

Name	# Ratings	% change surfaced hits, Similar Users	% change surfaced hits, Not Similar Users	Similar User Effect Ratio
men	753769	-0.71	-0.64	1.11
women	246440	-0.38	-0.09	4.24
fans of horror	48464	-0.24	-0.03	7.16
under 18	27211	-0.27	-0.03	9.02
25-34	395556	-0.38	-0.3	1.28
56+	38780	-0.18	-0.03	6.97
artist	50068	0.15	-0.07	-2.26
power users	381407	-0.76	-0.45	1.71

Table 1, which includes a variety of example groups, highlights the primary finding that some groups are especially effective at lowering performance for Similar Users compared to other users. For instance, looking at Table 1, we see that when half of women strike, surfaced hits decreased for non-striking women by 0.38% while surfaced hits decreased for non-women by 0.09%. Therefore, the Similar User Effect Ratio is 0.38 / 0.09 = 4.24. This effect is exaggerated even further in the case of the "under 18" group, which has a Similar User Effect Ratio of 9.02. On the other hand, the "25-34" age group has a more or less "flat" ratio of 1.28, and the same is the case for men.

It is clear from Table 1 that some homogenous groups are able to "punch above their weight" with respect to affecting the experience of non-striking users, at least those within their homogenous group. This is particularly important with respect to large recommenders, which we saw above are more robust to this component of the data strike effect (the effect of strikers on the experience of non-strikers). Regardless of the scale of data resources available to a recommender, it appears that recommendation quality for some users may always be vulnerable to targeted data strikes by Similar Users.

One hypothesis that explains why some data strikes hurt Similar Users more than other users lies in a holistic view of the user preference space. If a group of users has substantial uniqueness– or more specifically, mathematical independence – in their preferences when compared to other groups, a campaign by that group is less likely to hurt users not in that group. At the extreme, a group whose preferences are completely orthogonal to every other group may be able to execute a campaign without substantially affecting personalization for any other group (one realistic example might be groups based on language proficiency).

To understand this relationship, we compared a group's Similar User Effect Ratios from Table 1 to a measure of preference independence based on overlap in rate/no-rate behaviors. To calculate preference independence, we first create a vector with a column for each movie (3706 columns) and value equal to the proportion of users in the group who have rated the movie (i.e. the group implicit rating vector). A group's preference independence is the cosine distance of that group's vector to the similarly calculated vector for the all groups (i.e. the centroid). We focus on groups with over 20k ratings, ignoring the six very small groups for which Similar User effects are extremely noisy.

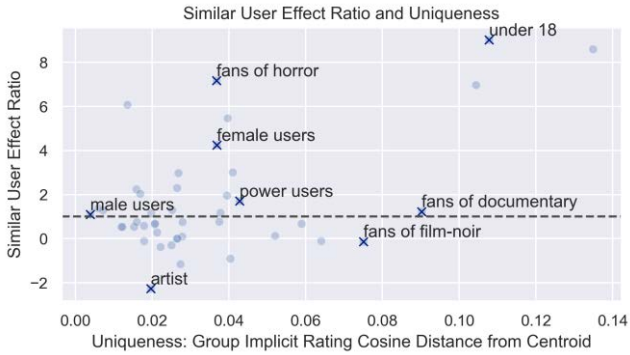


Figure 4: Scatterplot showing how homogeneous data strikes (with > 20k ratings) affect similar users differently than the general population. Along the x-axis, groups are organized by increasing uniqueness, defined by the cosine distance between the group implicit rating vector for the group and the general population. The y-axis shows the Similar User Effect Ratio. Gray dotted line shows ratio of 1. Pearson correlation is 0.55. "X" markers indicate labeled examples.

Figure 4 shows a full scatterplot of Similar User Effect Ratio for all groups with over 20k ratings. On the x-axis, we plot preference independence as defined above and on the y-axis, we plot Similar User Effect Ratio (note that negative values correspond to strikes that improve surfaced hits for Similar Users). Our experiments suggest a moderate positive relationship between a group's preference independence and the effect of a strike on non-participating Similar Users (Pearson's $r = 0.55$, $p < 0.001$), although this is likely driven by a non-trivial number of strong outliers (e.g. "under 18", "fans of horror" in Figure 4) given the Spearman correlation ($r = 0.18$, $p > 0.05$) is not statistically significant. The full interplay between preference independence and strike effectiveness is a fertile ground for research, and future work could more closely study how preference spaces might be operationalized for the purpose of data strikes.

The outsized Similar User effects for many homogenous strikes point to both strategic advantages and effects that may limit adoption. For example, if women data strike against a company to affect some lasting change (e.g. ending discriminatory practices, launching profit sharing), a homogenous strike that focuses on recruiting other women may be especially effective, with large Similar User strike effects potentially driving women away from the company to competitors and substantially reducing company revenue.

However, these effects may decrease strike adoption in some contexts. For instance, returning to a strike by women against a major tech company, imagine that the tech company provides important services to large populations (e.g. online shopping in areas with few brick-and-mortar stores, low-cost communication in areas without similarly priced options). Many potential participants may be unwilling to participate in a campaign that will disproportionately damage these services for other women, thus limiting the adoption of homogenous campaigns in certain contexts.

A related secondary observation from these experiments is that homogenous groups have varying effects on all non-strikers (rather than just Similar Users) compared to the "expected" effects based on their number of ratings (determined by a quadratic interpolation of our general results, as described above). In other words, homogenous strikes vary in their ability to "punch above their weight" with respect to their capacity to decrease performance for all non-participating users: In 34 of 50 homogeneous groups, surfaced hits for non-participating users decreased more than would be expected for a random group with an equal number of ratings. Ratios of observed to expected effect ranged from 0.52 for the group of users whose occupation was "retired" to 1.84 for "power users", with other examples including a 0.7 ratio for "fans of comedy" and a 1.35 ratio for "fans of fantasy" (see GitHub repository for more details).

Based on this secondary finding, it seems that homogenous campaigns will not always be effective at damaging the general population of users because some groups are under-performing, and even over-performing groups are limited by size in their ability to affect the general population. However, organizing campaigns with preference spaces in mind (i.e. campaigns that are homogenous in topical preferences) likely will be effective and, critically, may provide a way to challenge recommenders when the number of strikers is not sufficient to cause more general damage to performance.

5.3 Generalizing Beyond "Surfaced Hits" and SVD

In presenting our results, we have focused on our "surfaced hits" metrics. However, we also computed the more typical recommender systems evaluation metrics of RMSE, NDCG with all items, NDCG@k, Precision@k, and Recall@k for k = {5, 10}. We additionally calculated these metrics when only including "long-tail" (i.e. unpopular) movies. All these metrics produce similar results regarding the effects of various data strikes configurations, although they did not afford us the ability to analyze boycotts. The full dataset of results is available in our GitHub repository. We also note that in our early experiments using item-based k-NN, ML-1M, and traditional metrics, we observed very similar general trends, e.g. the effect of data strikes on the NDCG@10 of an item-based k-NN recommender mirrored effects on other traditional metrics for SVD.

6 DISCUSSION

In this paper, we have taken the previously hypothetical notion of data strikes, identified a wide variety of realistic campaign configurations (including when they are combined with traditional boycotts), and simulated the effect of these configurations in the recommender systems domain using best-practice evaluations. Comparing these campaign configurations and looking specifically at direct effects, we find that while the consumer power of boycotts still substantially outweighs the data labor power of data strikes, data strikes represent a potentially powerful new form of leverage. Moreover, we saw that strike organizers might specifically target preference spaces within a recommender to achieve especially effective data strikes within those spaces. Below, we discuss some of the more general implications of these results.

6.1 Barriers to Entry vs. Impact

Our results suggest that collective action organizers targeting companies that operate intelligent technologies like recommender systems have more options than is the case in traditional collective action. Specifically, these organizers have the ability to optimize for impact or for barrier to entry. Our results show that boycotts have a larger impact, but they require all participants bear the cost of not using a potentially valuable service. Strikes, on the other hand, allow users to continue to benefit from the use of technology platforms without completely losing the ability to collectively bargain. Historically, participation in political consumption has been easier for affluent groups [32] - data strikes represent an approach that may be substantially more accessible. Notably, our results suggest that new technologies focused on privacy (e.g. initiatives from Mozilla [14]) and online political consumption (e.g. recent work from Li et al. [39]) are a promising approach to empowering more individuals to participate in collective action.

This notion of low-barrier-to-entry collective action echoes early research on digital activism by Earl and Kimport, who argued that the web reduces costs for participating in protest behavior like boycotts, petitions, and email campaigns [9]. Specifically, data strikes can be seen as another tool in the toolbox of low-barrier-to-entry techniques, offering an additional avenue for people to take action against technology companies. However, we also observed that this increased accessibility is coupled with reduced power.

6.2 Towards a Holistic View of Data Strikes

Although our results give us important insight into the potential impact of data strikes and boycotts, our work likely only captures a portion of the real-world effect of a collective action against an intelligent technology. In particular, as is discussed in the Framework section, we cannot measure directly the indirect effects of traditional boycotts or data strikes. This means that our results should be interpreted as a lower bound on the effects of any data labor-related collective action campaign.

A related point that emerges from our results viewed with the lens of our framework is that collective action against

technology companies will largely be more powerful than collective action against non-technology companies. The effect of a boycott against, for instance, a clothing company, would largely not include either direct or indirect loss of data labor value (excluding edge cases like long-term sales and marketing data). Since our results suggest that these factors will be non-trivial in most tech company boycotts, a user boycotting a tech company is likely to have a greater effect on revenue than would be expected in a boycott with a more traditional type of target. We expand on these power dynamics further below.

6.3 The Power of Algorithms vs. The Power of Public Data

Outside of the context of boycotts and data strikes, our results can also be viewed as a means to better understand the power of data provided by the public relative to the power of algorithms. Namely, we observed that moderately-sized strikes can bring recommender accuracy down to the levels of early recommender systems from 1999. These results, along with the work of McMahon et al. [40] and others [18, 23], emphasize the data leverage that the public has in its relationship with data-hungry intelligent technologies and the companies that operate them. While the public perception of intelligent technologies like recommender systems is that they are largely the accomplishment of tech companies and the computer scientists they employ, these technologies are in fact a highly cooperative project between the public and companies. Without the companies and computer scientists, the intelligent technologies do not exist. But the same is also true for the public's contributions of data (i.e. data labor). This implies a much different power dynamic than is currently assumed by most people on both sides of this relationship.

6.4 Limitations

This work has several limitations not yet discussed above that should be highlighted. First and foremost, this paper focused on recommender systems which, while a business-critical family of intelligent technologies, is only one family of intelligent technologies that could be vulnerable to collective action campaigns. Future work should seek to replicate our research for other intelligent technologies, for instance search ranking algorithms (e.g. [45]), "newsfeed"-style technologies (e.g. [43]), traffic prediction (e.g. [21]), and wi-fi geolocation (e.g. [19]).

As noted above, we simulate boycotts in concert with strikes owing to that being the more ecologically valid choice in the context of our study. We note that one could imagine boycotts coupled only with partial strikes: e.g. someone who boycotts a system but does not delete their past ratings. Exploring these types of configurations - and longitudinal considerations in general - is an important direction of future work.

While we used best-practice evaluation techniques in the recommender systems community [8, 22, 46], these techniques have several limitations that also affect the large literature of recommender systems research that employs them. In particular, we considered only explicit ratings and did not consider implicit preferences expressed through user behavior (which are not available in the MovieLens dataset). We also only considered our recommenders in an offline environment (as opposed to in a live experiment). Finally, to gain more insight into the nuances of recommenders, it will be valuable explore other recommender system datasets, particularly datasets from industry contexts.

Another important limitation is that in our experiments, we had to operationalize male/female as a binary variable due to the data available in the MovieLens dataset. Similarly, we were not able to test other types of demographic groups (e.g. LGBT communities, political groups). Relatedly, our use of the term "homogenous" refers to a specific demographic or topical dimension; it does not consider the diversity within our "homogenous" groups, and doing so would be a fruitful area of future work.

Finally, this paper focused on understanding the effect of collective action campaigns of various sizes and types, but it did not consider the collective action problem of organizing or actuating these campaigns. Fortunately, this problem maps to a deep body of work within social computing and related fields on sociotechnical strategies for motivating collective action online (e.g. [35, 47]). An obvious direction of future work in this research space involves building tools to organize data strikes and boycotts that leverages this body of work (either using GDPR or restricting new data collection). Recent research suggests that user-friendly tools like browser extensions may be an effective approach for making collective action campaigns easy to join and conduct [39].

6.5 Potential Negative Impacts

In response to calls for the computing community to better engage with the negative impacts of our research [29], we wish to highlight two major concerns with this work. First, we emphasize that our findings may be equally useful to organizers of campaigns as to they are to companies interested in mitigating the effectiveness of such campaigns. Relatedly, it is entirely possible that using a simulated data strike methodology, companies could identify which groups of users are and are not "useful to the algorithm", i.e. they could rank groups based on their utility to some intelligent technology and use this ranking to justify ignoring the interests of some groups. If this occurred along demographic lines, this could lead to troubling societal outcomes, e.g. if majority groups can collectively bargain with tech companies and minority groups cannot. Technologies to organize data strikes and boycotts could help mitigate this issue by recruiting users from a variety of demographic groups (perhaps, guided by future work, specifically targeting some preference space like Comedy movies or electronics products). Our results suggest that this should be a priority in the design of these technologies.

Moreover, our ability to perform simulated campaigns was predicated on the public availability of the MovieLens dataset. Substantially more accurate simulations could be run using much richer datasets available only to corporations, so in any "data strike simulation arms race", there will be a clear advantage for corporations. This means that corporations may be

able to prepare models in advance to counteract boycotts or strikes. This might be mitigated through crowdsourced data collection or other means, a ripe area for future work.

7 CONCLUSION

In this paper, we have done the work of advancing the notion of data strikes from abstract discussion point to concrete campaigns that can be simulated. Through these simulations, we provided critical early empirical information to help advance the discussion around data strikes. In doing so, we first detailed a framework that describes the key elements of collective action campaigns that action data leverage (i.e. data strikes and traditional boycotts, which respectively leverage *data labor power* and *consumer power*). We found that these campaigns can be effective, with relatively small strikes wiping away significant portions of the value of recommender systems relative to simpler techniques. However, as datasets grow larger, data strikes become less effective, and strategies that target specific groups of users or preference spaces may become necessary. We discussed the implications of our results for those seeking to organize data strikes and companies seeking to understand potential effects on their core functionality.

8 ACKNOWLEDGMENTS

We are thankful to our anonymous reviewers, Loren Terveen, Rohan Pitre, Michael Ekstrand, and Lisa Lendway for the comments and feedback on the work.

REFERENCES

[1] 1 million jobs will disappear by 2026. How to prepare for automation-commentary: 2018. *https://www.cnbc.com/2018/02/02/automation-will-kill-1-million-jobs-by-2026-what-we-need-to-do-commentary.html.*

[2] Arrieta Ibarra, I. et al. 2018. Should We Treat Data as Labor? Moving Beyond "Free." *American Economic Association Papers & Proceedings*. 1, 1 (2018).

[3] Big Tobacco - Wikipedia: *https://en.wikipedia.org/wiki/Big_Tobacco.*

[4] Brynjolfsson, E. and McAfee, A. 2014. *The second machine age: Work, progress, and prosperity in a time of brilliant technologies*. WW Norton & Company.

[5] Burges, C. et al. 2005. Learning to Rank Using Gradient Descent. *Proceedings of the 22Nd International Conference on Machine Learning* (New York, NY, USA, 2005), 89–96.

[6] California just passed one of the toughest data privacy laws in the country - The Verge: 2018. *https://www.theverge.com/2018/6/28/17509720/california-consumer-privacy-act-legislation-law-vote.*

[7] Chirita, P.-A. et al. 2005. Preventing shilling attacks in online recommender systems. *Proceedings of the 7th annual ACM international workshop on Web information and data management* (2005), 67–74.

[8] Cremonesi, P. et al. 2010. Performance of Recommender Algorithms on Top-n Recommendation Tasks. *Proceedings of the Fourth ACM Conference on Recommender Systems* (New York, NY, USA, 2010), 39–46.

[9] Earl, J. and Kimport, K. 2011. *Digitally enabled social change: Activism in the internet age*. Mit Press.

[10] Early Facebook and Google Employees Form Coalition to Fight What They Built - The New York Times: 2018. *https://www.nytimes.com/2018/02/04/technology/early-facebook-google-employees-fight-tech.html.*

[11] Faceblock campaign urges users to boycott Facebook for a day: 2018. https://www.theguardian.com/technology/2018/apr/07/faceblock-campaign-urges-users-boycott-facebook-for-one-day-protest-cambridge-analytica-scandal.

[12] Facebook admits social media sometimes harms democracy - The Washington Post: 2018. https://www.washingtonpost.com/news/the-switch/wp/2018/01/22/facebook-admits-it-sometimes-harms-democracy/?utm_term=.f0cf046c1ebe.

[13] Facebook and Cambridge Analytica: What You Need to Know as Fallout Widens - The New York Times: 2018. *https://www.nytimes.com/2018/03/19/technology/facebook-cambridge-analytica-explained.html?mtrref=www.google.com&gwh=DB24EEF2DB7C9762BA867E70C9C6DB2C&gwt=pay.*

[14] Facebook Container Extension: Take control of how you're being tracked | The Firefox Frontier: 2018. *https://blog.mozilla.org/firefox/facebook-container-extension/.*

[15] Granholm, J. and Eldred, C. Facebook owes you money (opinion) - CNN: 2018. https://www.cnn.com/2018/04/11/opinions/facebook-should-pay-us-for-using-our-data-granholm-eldred/index.html.

[16] Facebook users unite! "Data Labour Union" launches in Netherlands | Reuters: 2018. https://www.reuters.com/article/us-netherlands-tech-data-labour-union/facebook-users-unite-data-labour-union-launches-in-netherlands-idUSKCN1IO2M3.

[17] Gomez-Uribe, C.A. and Hunt, N. 2015. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Trans. Manage. Inf. Syst.*6, 4 (Dec. 2015), 13:1–13:19. DOI:https://doi.org/10.1145/2843948.

[18] González Cabañas, J. et al. 2017. FDVT: Data Valuation Tool for Facebook Users. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (2017), 3799–3809.

[19] Google, Apple tap crowdsourcing to map out WiFi locations | IT Business: 2010. *https://www.itbusiness.ca/news/google-apple-tap-crowdsourcing-to-map-out-wifi-locations/15658.*

[20] Google sees major claims of harassment and discrimination as lawsuits proceed | Technology | The Guardian: 2018. *https://www.theguardian.com/technology/2018/mar/28/google-sexual-harassment-pay-gap-lawsuits-proceed.*

[21] Graunke, G.L. 2001. *Using predictive traffic modeling*. Google Patents.

[22] Harper, F.M. and Konstan, J.A. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.*5, 4 (Dec. 2015), 19:1–19:19. DOI:https://doi.org/10.1145/2827872.

[23] Hecht, B. 2017. HCI and the U.S. Presidential Election: A Few Thoughts on a Research Agenda. *CHI '18 Panel Presentation* (Denver, CO, 2017).

[24] How retailers can keep up with consumers: 2013. https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers.

[25] How Silicon Valley became "Big Tech.": http://www.slate.com/articles/technology/technology/2017/11/how_silicon_valley_became_big_tech.html.

[26] Huang, S. et al. 2015. Listwise Collaborative Filtering. *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2015), 343–352.

[27] Hug, N. 2017. *Surprise, a Python library for recommender systems*. http://surpriselib.com

[28] Inside the Big Oil Game - TIME: *http://content.time.com/time/magazine/article/0,9171,920328,00.html.*

[29] It's Time to Do Something: Mitigating the Negative Impacts of Computing Through a Change to the Peer Review Process - ACM FCA: *https://acm-fca.org/2018/03/29/negativeimpacts/.*

[30] Key Changes with the General Data Protection Regulation: *https://www.eugdpr.org/the-regulation.html.*

[31] Koenigstein, N. 2017. Rethinking Collaborative Filtering: A Practical Perspective on State-of-the-art Research Based on Real World Insights. *Proceedings of the Eleventh ACM Conference on Recommender Systems* (2017), 336–337.

[32] Koos, S. 2012. What drives political consumption in Europe? A multi-level analysis on individual characteristics, opportunity structures and globalization. *Acta Sociologica*. 55, 1 (2012), 37–57.

[33] Koren, Y. 2010. Factor in the Neighbors: Scalable and Accurate Collaborative Filtering. *ACM Trans. Knowl. Discov. Data.* 4, 1 (Jan. 2010), 1:1–1:24. DOI:https://doi.org/10.1145/1644873.1644874.

[34] Koyejo, O. et al. 2013. Retargeted Matrix Factorization for Collaborative Filtering. *Proceedings of the 7th ACM Conference on Recommender Systems* (New York, NY, USA, 2013), 49–56.

[35] Kraut, R.E. et al. 2012. *Building successful online communities: Evidence-based social design*. Mit Press.

[36] Lam, S.K. and Riedl, J. 2004. Shilling recommender systems for fun and profit. *Proceedings of the 13th international conference on World Wide Web* (2004), 393–402.

[37] Lanier, J. 2014. *Who owns the future?*. Simon and Schuster.

[38] Lawrence, N.D. and Urtasun, R. 2009. Non-linear Matrix Factorization with Gaussian Processes. *Proceedings of the 26th Annual International Conference on Machine Learning* (New York, NY, USA, 2009), 601–608.

[39] Li, H. et al. Out of Site: Empowering a New Approach to Online Boycotts. *Proceedings of the 2018 Computer-Supported Cooperative Work and Social Computing (CSCW'2018 / PACM).*

[40] McMahon, C. et al. 2017. The Substantial Interdependence of Wikipedia and Google: A Case Study on the Relationship Between Peer Production Communities and Information Technologies. *ICWSM* (2017), 142–151.

[41] Netflix Update: Try This at Home: *http://sifter.org/simon/journal/20061211.html.*

[42] Newman, B.J. and Bartels, B.L. 2011. Politics at the checkout line: Explaining political consumerism in the United States. *Political Research Quarterly*. 64, 4 (2011), 803–817.

[43] Paek, T. et al. 2010. Predicting the Importance of Newsfeed Posts and Social Network Friends. *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence* (Atlanta, Georgia, 2010), 1419–1424.

[44] Posner, E.A. and Weyl, E.G. 2018. *Radical Markets: Uprooting Capitalism and Democracy for a Just Society*. Princeton University Press.

[45] Radlinksi, F. and Joachims, T. 2005. Query Chains: Learning to Rank from Implicit Feedback. *KDD '05: 11th ACM Conference on Knowledge Discovery and Data Mining* (Chicago, IL, 2005).

[46] Said, A. and Bellogín, A. 2014. Comparative Recommender System Evaluation: Benchmarking Recommendation Frameworks. *Proceedings of the 8th ACM Conference on Recommender Systems* (New York, NY, USA, 2014), 129–136.

[47] Salehi, N. et al. 2015. We Are Dynamo: Overcoming Stalling and Friction in Collective Action for Crowd Workers. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (New York, NY, USA, 2015), 1621–1630.

[48] Sarwar, B. et al. 2001. Item-based Collaborative Filtering Recommendation Algorithms. *Proceedings of the 10th International Conference on World Wide Web* (New York, NY, USA, 2001), 285–295.

[49] Schwab, K. et al. 2011. Personal data: The emergence of a new asset class. *An Initiative of the World Economic Forum* (2011).

[50] Shapiro, C. et al. 1999. Information rules: a strategic guide to the network economy. *Journal of Economic Education*. 30, (1999), 189–190.

[51] Sharma, A. et al. 2015. Estimating the Causal Impact of Recommendation Systems from Observational Data. *Proceedings of the Sixteenth ACM Conference on Economics and Computation* (New York, NY, USA, 2015), 453–470.

[52] Should internet firms pay for the data users currently give away? - The digital proletariat: 2018. *https://www.economist.com/news/finance-and-economics/21734390-and-new-paper-proposes-should-data-providers-unionise-should-internet*.

[53] Smith, B. and Linden, G. 2017. Two decades of recommender systems at Amazon. com. *IEEE Internet Computing*. 21, 3 (2017), 12–18.

[54] Song, Y. et al. 2013. Evaluating and Predicting User Engagement Change with Degraded Search Relevance. *Proceedings of the 22Nd International Conference on World Wide Web* (New York, NY, USA, 2013), 1213–1224.

[55] The Latest: Facebook users await word on privacy scandal - The Washington Post: 2018. https://www.washingtonpost.com/national/the-latest-facebook-users-await-word-on-privacy-scandal/2018/04/09/37e8d82a-3c2a-11e8-955b-7d2e19b79966_story.html?utm_term=.1247b93f6998.

[56] Vincent, N. et al. 2018. Examining Wikipedia With a Broader Lens: Quantifying the Value of Wikipedia's Relationships with Other Large-Scale Online Communities. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (2018), 566.

[57] Wen, H. et al. 2018. Exploring Recommendations Under User-Controlled Data Filtering. *ACM Conference on Recommender Systems (RecSys' 18)* (2018).

[58] Wikipedia:Reusing Wikipedia content - Wikipedia: *https://en.wikipedia.org/wiki/Wikipedia:Reusing_Wikipedia_content*.

[59] Zhou, R. et al. 2010. The impact of YouTube recommendation system on video views. *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (2010), 404–410.

Learning Semantic Models of Data Sources Using Probabilistic Graphical Models

Binh Vu
USC Information Sciences Institute
Marina del Rey, CA
binhvu@isi.edu

Craig A. Knoblock
USC Information Sciences Institute
Marina del Rey, CA
knoblock@isi.edu

Jay Pujara
USC Information Sciences Institute
Marina del Rey, CA
jpujara@isi.edu

ABSTRACT

A semantic model of a data source is a representation of the concepts and relationships contained in the data. Building semantic models is a prerequisite to automatically publishing data to a knowledge graph. However, creating these semantic models is a complex process requiring considerable manual effort and can be error-prone. In this paper, we present a novel approach that efficiently searches over the combinatorial space of possible semantic models, and applies a probabilistic graphical model to identify the most probable semantic model for a data source. Probabilistic graphical models offer many advantages over existing methods: they are robust to noisy inputs and provide a straightforward approach for exploiting relationships within the data. Our solution uses a conditional random field (CRF) to encode structural patterns and enforce conceptual consistency within the semantic model. In an empirical evaluation, our approach outperforms state of the art systems by an average 8.4% of F_1 score, even with noisy input data.

CCS CONCEPTS

• **Information systems → Information integration**; • **Computing methodologies → Learning in probabilistic graphical models**.

KEYWORDS

Semantic models, knowledge graph, probabilistic graphical models, semantic web, linked data, ontology

ACM Reference Format:
Binh Vu, Craig A. Knoblock, and Jay Pujara. 2019. Learning Semantic Models of Data Sources Using Probabilistic Graphical Models. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313711

1 INTRODUCTION

There is an enormous number of data sources available on the web. Integrating information from multiple data sources enables interesting applications that can have a large social impact [27]. However, because data publishers usually publish data in different data formats and using different conventions, harvesting information can be challenging. The task of semantic modeling addresses this challenge by building a model that maps the information found in data sources to semantically meaningful structures.

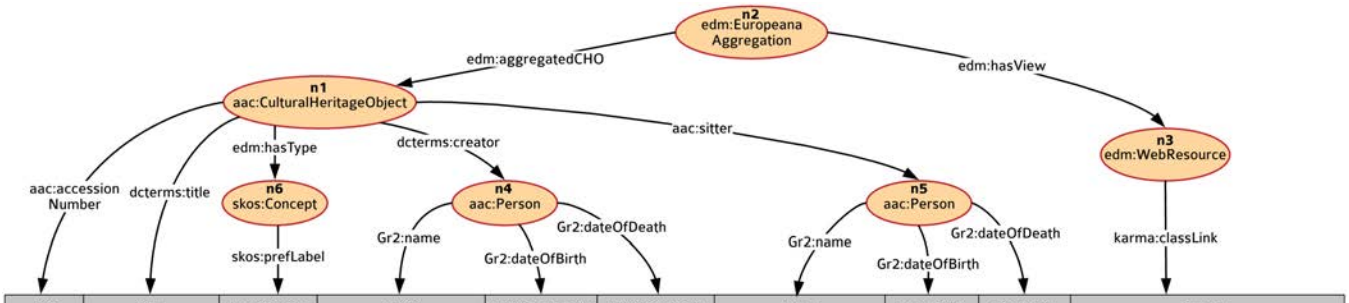
A semantic model is a graph where ontology classes and source attributes are nodes and ontology properties are edges. Figure 1 depicts a semantic model for a sample data source about artworks, which contain information about paintings, their creators and sitters. By explicitly encoding the semantic type and relationship between source attributes in the graph, the model precisely describes the intended meaning of the data source. Moreover, it can be used to automatically generate R2RML rules to transform the data source to RDF triples for publishing to knowledge graphs [7, 25].

Since creating semantic models, or semantic modeling, requires significant effort and expertise, it is desirable to have an automated or semi-automated system to support users. There is previous work in predicting the semantic types of source attributes [18, 20, 22], called semantic labeling. A semantic type is a pair of ontology class and predicate. For instance, in Figure 1, the semantic type of attribute *name* is ⟨*aac:Person*, *Gr2:name*⟩. However, semantic labeling alone is insufficient for describing and publishing data as RDF because it fails to capture the relationships between attributes. Other work focuses directly on semantic modeling [29, 30]. The state-of-the-art methods use predicted semantic types of source attributes from semantic labeling systems, then build a minimum-weighted tree (Steiner Tree) that connects the attributes together. Heuristic functions and other machine learning methods, such as frequent pattern mining, are used to assign cost to edges of the tree.

Generating semantic models automatically still remains challenging for several reasons: the data of attributes may be very similar, and there is more than one possible relationship between entities. For example, it is easy to confuse a *date* attribute with created date of paintings or creators' birth date. If we simply look at one signal from the cost of the edge, it would be hard to decide which one is correct because they are similar. However, there are also multiple weak signals from its relationships with other remaining attributes that can assist our decision making. For instance, if we group values of *date* attribute by painting id and find that one painting appears to have been painted at two different points in time, then it is likely that this *date* attribute is not the created date.

To address these issues, we present a new approach for semantic modeling that uses a probabilistic graphical model (PGM). A PGM allows us to express and capture the weak signals within the data. From a set of known semantic models, we auto-generate positive and negative examples representing many possible descriptions of sources. Using this training set, the PGM is trained to distinguish between good and bad models. To predict the most probable semantic model for a data source, we use beam search to explore the

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313711



ref	title	classification	artist	ArtistBornDate	ArtistDiedDate	name	bornDate	diedDate	image
NY270002	Chester Alan Arthur	Painting	Eastman Johnson	29 Jul 1824	5 Apr 1906	Chester Alan Arthur	5 Oct 1829	18 Nov 1886	http://npgportraits.si.edu/.../ny270002.jpg
NPG.70.36	Apollo 11 Crew	Painting	Ronald B. Anderson	1929	-	Neil Alden Armstrong	5 Aug 1930	25 Aug 2012	http://npgportraits.si.edu/.../7000062a.jpg
						Edwin Eugene Aldrin, Jr.	20 Jan 1930	-	
						Michael Collins	31 Oct 1930	-	
NPG.67.62	Chester A. Arthur	Painting	Ole Peter Hansen Balling	13 Apr 1823	1 May 1906	Chester Alan Arthur	5 Oct 1829	18 Nov 1886	http://npgportraits.si.edu/.../6700114a.jpg

Figure 1: NPG data source and its semantic model

space of possible models and evaluate the likelihood of the model using the trained PGM as a scoring function.

The contribution of this paper is a novel way to explore relationships within data sources and semantic models using a PGM, as well as a flexible framework that can easily be extended to incorporate new features. We present experimental results on two museum datasets published in Taheriyan et al. [29]. In the experiments, we control different levels of noise by using four different semantic labeling methods. These experiments show that our approach outperforms state-of-the-art systems by an average of 8.4% F_1 score, even with noisy input data.

2 MOTIVATING EXAMPLE

In this section, we provide an example to describe the typical process of building a semantic model for a sample data source of the National Portrait Gallery (NPG[1]) museum in Figure 1. On top of the table is a semantic model of the NPG. The source attributes are data nodes or leaf nodes. The ontology classes are class nodes, and links between nodes are ontology predicates. We also say that links between two class nodes are class links. Additionally, data links are links between class nodes and data nodes. For example, *aac:Person* and *aac:CulturalHeritageObject* are class nodes, while the source attributes *title* and *classification* are data nodes. The ontology classes and predicates are from DCTerms, SKOS, ElementsGr2, AAC, EDM ontologies[2].

The first step in building a semantic model is semantic labeling. A user or an automated semantic labeling system annotates each source attribute with semantic types. For example, the attributes *title* and *image* are labeled as ⟨*aac:CulturalHeritageObject*, *dcterms:title*⟩, ⟨*edm:WebResource*, *karma:classLink*⟩, respectively. Note that we use a special predicate *karma:classLink* to indicate that the data node *image* contains URIs of *edm:WebResource*.

The second step is specifying relationships between nodes in the model. This is a complicated problem, as there are several paths to connect two nodes in the domain ontology. For instance, *aac:Person* can be *dcterms:creator* or *aac:sitter* of *aac:CulturalHeritageObject*. Another example is that *bornDate* and *artistBornDate* do not belong to same person. If we only know the scores of semantic types of attributes and how frequent a predicate is used to connect two class nodes, we may not know which path is correct.

When users model a data source, they use various information to make decisions. For example, in Figure 1, we have three people who are listed under attribute *name* and are in the "Apollo 11 Crew" painting. As a painting is usually painted by one individual, a person would conclude that these are the names of the sitters rather than the names of the creators. Attributes *artist* and *ArtistBornDate* are two columns next to each other. This is weak evidence indicating that the two columns contain the name and birth date of a person. If the *bornDate* is mislabeled as death date of *aac:Person*, we would be skeptical about having Eastman Johnson draw a painting of Chester A. Arthur when he was 5 years old. Although these signals are useful, they are not always strong and correct. Can we learn to combine and apply the signals to build a correct semantic model?

3 PROBABILISTIC GRAPHICAL MODELS FOR SEMANTIC MODELING

Our problem of learning semantic models is defined as follow: suppose we have a set of domain ontologies O, a set of sources $\{s_1, s_2, ..., s_n\}$ and their semantic models $\{sm(s_1), sm(s_2), ..., sm(s_n)\}$. Given a target source $s(a_1, a_2, ..., a_n)$, in which a_i is a source attribute, we want to automatically build its corresponding semantic model $sm(s)$.

Our approach for building semantic models is similar to the procedure humans use. We model source attributes one at a time. For every attribute, we rank its modeling options based on the likelihood of the overall semantic model. Then, we select the attribute that has the highest rank. The process ends when there is no attribute left.

As we have discussed, users can exploit multiple signals to assist their decisions. For example, in Figure 1, given two options for relation ⟨n_1_*CulturalHeritageObject*, n_4_*Person*⟩: *dcterms:creator*

[1] http://npg.si.edu/home/national-portrait-gallery
[2] These ontologies are used in the evaluation datasets

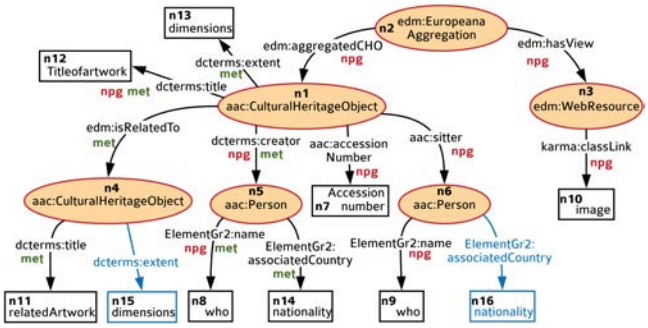


Figure 2: A example of integration graph[3]. This is built from sm(npg) and sm(met). Some nodes and properties are hidden for readability.

and *aac:sitter*, users can use the fact that name of n_4_*Person* is linked to the attribute *artist* to select predicate *dcterms:creator* over *aac:sitter*. Those signals are often collective and structural. They can be captured and represented naturally using a PGM, which is a very powerful framework that captures dependencies between relations and is robust to noise.

In Section 3.1, we describe the procedure to build a semantic model using search. We then explain how we apply and train a PGM as a ranking function in Section 3.2 and 3.3.

3.1 Searching for semantic models

Let $D_0 = sm(\{\})$ be a start state which is an empty model, $f : D_0 \rightarrow D_1$ be a transition function to move from one state to another state by adding one source attribute. Then, in general, constructing a semantic model can be solved using search algorithms by repeatedly invoking the transition function f until we reach the goal state, a tree connecting all source attributes.

Algorithm 1: TransitionFunction

Input: A search state: s_t
A set of source attributes: $A = \{a_0, ..., a_n\}$
An integration graph: G_{int}
Output: List of next search states

```
1 nextStates ← []
2 A′ ← A \ s_t.leafNodes
3 for a_i ← A′ do
4     trees ← MergeIntoTree (s_t, G_int, a_i)
5     for s_{t+1} ← trees do
6         nextStates.append(s_{t+1})
7 return nextStates
```

The transition function we use is defined in algorithm 1. We start with each unmerged attribute (line 2 - 3) and generate next states by merging them into current state (line 4). Algorithm 2 is used to combine an attribute with the current semantic model. The intuition of algorithm 2 is that we find all possible connecting paths between source attribute and leaves or root of the tree. Each path with a current tree forms a new state we want to generate.

[3]This example is based on an example from [29]

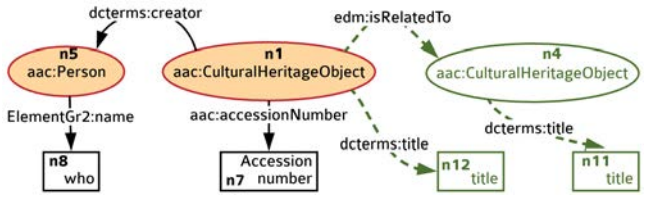


Figure 3: Two possible paths to add *title* attribute into current tree

To find the paths connecting source attribute and the existing tree, we use an integration graph from Taheriyan's work [29]. An integration graph is a directed weighted graph built from known semantic models, and represents a space of plausible semantic models we have seen in the training data. In Figure 2, we show an integration graph built from two data sources *npg* and *met*. Each link is annotated with its source name to specify where it comes from. When the graph is used in prediction, it is expanded by adding semantic types depicted in blue color. For example, in a new data source table, we have a column called *dimensions* tagged with a semantic type ⟨*aac:CulturalHeritageObject*, *dcterms:extent*⟩, we add one data node n_{15} and one link *dcterms:extent* from n_4 to n_{15}. However, we do not need to update node n_1 because we have already had link (n_1, *dcterms:extent*, n_{13}).

Now given an integration graph $G_{int} = \{V_{int}, E_{int}\}$, suppose that we are in the state s_t that we have four nodes $\{n_1, n_5, n_7, n_8\}$ as in Figure 3 and we want to merge attribute a_t = *title*. First, we need to align a_t to nodes in V_{int} that have same semantic type (line 2 in algorithm 2). The two nodes that match are n_{12} and n_{11}. For each node, we find all paths that connect the node to s_t, each path generates different tree. For example, there are two paths ($n_1 \rightarrow n_{12}$) and ($n_1 \rightarrow n_4 \rightarrow n_{11}$). Therefore, we have two next states. Each next state will be a tree that combines s_t and a merge path.

Algorithm 2: MergeIntoTree

Input: A search state: s_t
An integration graph: G_{int}
An attribute: a_t
Output: List of trees which contain the given attribute a_t

```
1 newTrees ← []
2 mntPts ← G_int.findAttribute(a_t.semanticType)
3 for mntPt ← mntPts do
4     connectedPaths ← all paths that connect leaves or root of
       s_t with mntPt
5     for path ← connectedPaths do
6         newTree ← merge mntPt to s_t using path
7         newTrees.append(newTree)
8 return newTrees
```

In addition to using the transition function in the beam search, we can also use it in interactive semantic modeling systems such as Karma [11]. In particular, from a current curated semantic model, the transition function is invoked to generate all semantic model

candidates. Then, the candidates are ranked using a scoring function to produce the top-k candidates for users to choose from. A user can also make corrections if there is no suitable option. In our approach, the scoring function directly affects the quality of our predicted semantic models. In the next section, we will describe how we use PGM to learn a scoring function.

3.2 Using Graphical Models as the score function

A probabilistic graphical model (PGM) is a well-known and effective framework to represent a probability distribution of random variables [13]. In order to apply PGMs in semantic modeling, we choose two kinds of random variables: input variables $\boldsymbol{x}$ that are links between nodes in a semantic model, and output variables $\boldsymbol{y}$ that are labels of the links. A label of a link has possible values in $\mathcal{V} = \{true, false\}$. A link with label *true* means it is correct and present in the gold semantic model. Since we are interested in predicting labels of links: $P(\boldsymbol{y}|\boldsymbol{x})$, it is natural to use a conditional random field (CRF). In particular, our standard CRF has following form:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x}_c)} \prod_{C_p \in C'} \prod_{\Psi_c \in C_p} \Psi_c(\boldsymbol{y}_c, \boldsymbol{x}_c; \theta_p)$$

$$Z(\boldsymbol{x}_c) = \sum_{\boldsymbol{y}} \prod_{c \in C'} \prod_{\Psi_c \in C_p} \Psi_c(\boldsymbol{y}_c, \boldsymbol{x}_c; \theta_p)$$

where $\Psi_c(\boldsymbol{y}_c, \boldsymbol{x}_c; \theta_p)$ is a factor function that takes two set of variables $\boldsymbol{y}_c$ and $\boldsymbol{x}_c$ with parameters θ_p, and $Z(\boldsymbol{x}_c)$ is a partition function acted as a normalization factor to ensure the distribution $P(\boldsymbol{y}|\boldsymbol{x})$ sum to 1. The factors that have same input structure are grouped into $C' = \{C_1, C_2, ..., C_n\}$, where each C_i is a set of factors called *clique template* [26]. Factors in a clique template C_p share same parameters θ_p. A common choice of factor functions is exponential function, and each factor is log-linear with a set of feature functions f as follows:

$$\Psi_c(\boldsymbol{y}_c, \boldsymbol{x}_c, \theta_p) = \exp\left\{ \sum_k \theta_{pk} f_{pk}(\boldsymbol{y}_c, \boldsymbol{x}_c) \right\}$$

To simplify the notation, for every equation, x or y refers to one random variable, and $\boldsymbol{x}$ or $\boldsymbol{y}$ (bolded) refers to sets of random variables. Additionally, x and y symbols are used to denote input and output of a same link, respectively. For a complete guide about CRF, readers can refer to Sutton et al. [26].

Our ultimate goal is to construct a semantic model where all links are true. So the likelihood of the model being correct is $P(\forall y \in \boldsymbol{y}; y = true|\boldsymbol{x})$, and can be used as a score function for ranking. With the CRF framework above, in the following subsections, we will describe different clique templates and their features, each capture different relations between data sources and semantic models.

3.2.1 Link between nodes. These are factors over one link representing the likelihood of the link being correct or not. In particular, the factors have this form: $\Psi(y, x)$. The first feature of the factor is the confidence score of a semantic type, which is one of the outputs from the semantic labeling step. The features are defined as:

$$f_{ij}^{SS}(y, x) = \mathbf{1}_{\{y=i\}} \mathbf{1}_{\{st(x)=j\}} \text{st_score}(x); \forall i \in \mathcal{V}, j \in \mathcal{ST}$$

$$f_{ij}^{\overline{SS}}(y, x) = \mathbf{1}_{\{y=i\}} \mathbf{1}_{\{st(x)=j\}} (1 - \text{st_score}(x)); \forall i \in \mathcal{V}, j \in \mathcal{ST}$$

where $\mathbf{1}_{\{*\}}$ is an indicator function, $\mathcal{ST}$ is set of all possible semantic types, $st(x)$, $st_score(x)$ are the semantic type and confidence score of link x, respectively. These are typical features used in graphical models, where each feature has its own weight and is non-zero only when a particular condition is met.

Note that if we only use $f_{ij}^{SS}(y, x)$, then $\Psi(y = true, x) - \Psi(y = false, x) = \text{st_score}(x)(\theta_{true,j} - \theta_{false,j})$, then $\Psi(y = true, x) > \Psi(y = false, x)$ is independent of $\text{st_score}(x)$ but only weights of the features. By introducing $f_{ij}^{\overline{SS}}(y, x)$, we are able to assign higher energy to $\Psi(y = true, x)$ when confidence score is high, and higher energy $\Psi(y = false, x)$ when confidence score is low (assuming that the higher confidence score, the more chance y is true). The simple and yet efficient technique above is applied to many other signals which have continuous value in this paper.

The confidence score signal is only applied for data links. For class links, we estimate the probability of a link being correct as follows:

$$P(s, x, o) = P(s) * P(o) * P(x|s, o)$$

$$P(x|s, o) = \frac{count(s, x, o)}{count(s, o)}$$

where s, o are source node and target node of a link x in the semantic model, respectively. In the equation, the probability of nodes $P(s)$ and $P(o)$ are estimated using a logistic regression (LR) model, which is trained on the same training set of our graphical model. The intuition of two features used in the LR model is described below:

- Total confidence score of semantic types of all child data nodes of n: a class node is likely to be present in the model if it links to many data nodes.
- The merging cost of node n with another node n' that has the same label in the graph: two class nodes with disjoint sets of properties should be merged together to simplify the model.

Features of class links are defined similar to f_{ij}^{SS} as follow.

$$f_{ij}^{CL}(y, x) = \mathbf{1}_{\{y=i\}} \mathbf{1}_{\{\text{triple}(x)=j\}} P(\text{triple}(x)); \forall i \in \mathcal{V}, j \in \mathcal{T}$$

$$f_{ij}^{\overline{CL}}(y, x) = \mathbf{1}_{\{y=i\}} \mathbf{1}_{\{\text{triple}(x)=j\}} (1 - P(\text{triple}(x))); \forall i \in \mathcal{V}, j \in \mathcal{T}$$

where $\text{triple}(x) = (s, x, o)$, $\mathcal{T}$ is set of all triples.

Beyond the two features above, we also use other features which depend mostly on x. To define these features, we define a series of observation functions $o_t(x)$. The corresponding features have the form:

$$f_{it}^{LO}(y, x) = \mathbf{1}_{\{y=i\}} o_t(x); \forall i \in \mathcal{V}$$

The set of observation functions we used are listed in table 1.

3.2.2 Cardinality relationships. The cardinality constraint between two source attributes can be used to discover an incorrect model. For example, a painting should have only one primary title, i.e the id and primary title of a painting have a one-to-one relationship. If we observe that the relationship is one-to-many, then one of the properties is linked to the wrong attribute. We express the heuristic

Table 1: Observation functions used in factor of link of nodes

Features	Explanation
Delta_SS(x=j) $\forall j \in \mathcal{ST}$	Difference between score of current semantic type and the remained best semantic type
SS_rank(x)	Rank of a predicted semantic type (determined by semantic labeling method
P(x|s)	Frequency of a link x given its source node s
NoDataChild	The source node has only child class nodes
NoSiblings	The source node has only one child node
Prior	Prior probability of a link

above using factors between two outgoing links of class nodes x_1 and x_2: $\Psi(y_1, y_2, x_1, x_2)$.

Cardinality relationship between attribute a_2 and attribute a_1 is computed by averaging the number of unique values of a_2 grouped by a_1. If the average number is greater than a threshold $\alpha = 1 + |\epsilon|, \epsilon \gtrapprox 0$, we consider this to be either a one-to-many or a many-to-many relationship, otherwise, it is a one-to-one relationship. The threshold is chosen to be slightly greater than 1 in order to make this feature robust to noise in the data source (e.g., missing values). In particular, α is set to 1.05 based on experimental results. For example, in Figure 1, the cardinality of attribute *name* and attribute *ref* is $CRD(\text{ref}, \text{name})$ = 1-to-many because $\frac{1+3+1}{3} = 1.67 > \alpha$. If two attributes belong to two separate nested lists, which occurs infrequently, we do not compute its cardinality relationship and set it to NULL. The cardinality features of this clique template are defined as:

$$f^{CRD}_{ijkmn}(y_1, y_2, x_1, x_2) = \mathbf{1}_{\{y_1=i\}}\mathbf{1}_{\{y_2=j\}}\mathbf{1}_{\{st(x_1)=k\}}\mathbf{1}_{\{st(x_2)=m\}} \mathbf{1}_{\{CRD(x_1,x_2)=n\}} ; \forall i, j \in \mathcal{V}, k, m \in \mathcal{ST}, n \in \mathcal{CRD}$$

where $\mathcal{CRD}$ = {many-to-many, 1-to-many, 1-to-1, NULL}.

3.2.3 Properties co-occurrence. This clique template gives a boost to properties that are likely to occur together. For example, birth date and death date of an artist have higher co-occurrence than birth date and creation date of a painting. If we have two dates attributes, the chance of their being birth date and creation date will be lower than the chance that they are birth date and death date.

A co-occurrence frequency matrix is estimated from known semantic models and are used as features in this clique template:

$$f^{OCC}_{ijmn}(y_1, y_2, x_1, x_2) = \mathbf{1}_{\{y_1=i\}}\mathbf{1}_{\{y_2=j\}}\mathbf{1}_{\{st(x_1)=m\}}\mathbf{1}_{\{st(x_2)=n\}} \text{co_occurrence}(x_1, x_2) ; \forall i, j \in \mathcal{V}, m, n \in \mathcal{ST}$$

$$f^{\overline{OCC}}_{ijk}(y_1, y_2, x_1, x_2) = \mathbf{1}_{\{y_1=i\}}\mathbf{1}_{\{y_2=j\}}\mathbf{1}_{\{st(x_1)=m\}}\mathbf{1}_{\{st(x_2)=n\}} (1 - \text{co_occurrence}(x_1, x_2)) ; \forall i, j \in \mathcal{V}, m, n \in \mathcal{ST}$$

where x_1 and x_2 are sibling links of a class node.

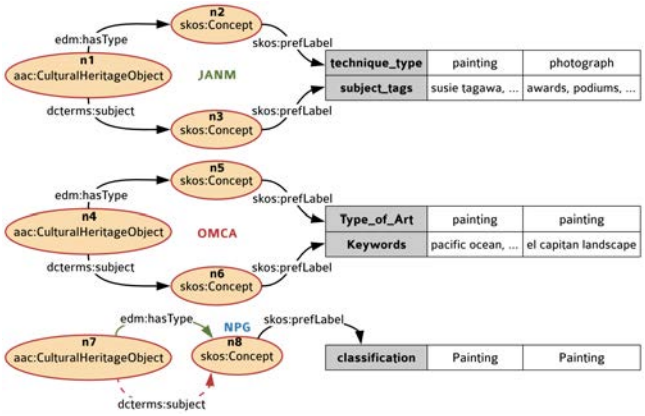


Figure 4: Excerpt from semantic models of the JANM, OMCA[4] and NPG data sources

3.2.4 Duplicated properties. An entity usually has many properties, but a few of them are duplicated. An example of a duplicated property is a class node *Person* with two links *worksFor* to two data nodes *organization1* and *organization2*. As semantic labeling systems often output the same semantic types for similar attributes, knowing which properties could be duplicated helps us identify incorrect links.

Factors in this clique template take a set of same label outgoing links of a class node and have the following features:

$$f^{DP}_{i0}(\boldsymbol{y_t}, \boldsymbol{x_t}) = \mathbf{1}_{\{|\{\forall y_i \in \boldsymbol{y_t} | y_i = true\}|=1\}}\mathbf{1}_{\{st(\boldsymbol{x_t})=i\}}; \forall i \in \mathcal{ST}$$

$$f^{DP}_{i1}(\boldsymbol{y_t}, \boldsymbol{x_t}) = \mathbf{1}_{\{|\{\forall y_i \in \boldsymbol{y_t} | y_i = true\}|>1\}}\mathbf{1}_{\{st(\boldsymbol{x_t})=i\}}; \forall i \in \mathcal{ST}$$

3.2.5 Grouping properties. In hierarchical data sources, properties of an entity are usually grouped under the same nested object. For example, birth date and name of person are properties of the *artist* object and *artist* is in a record object in a data source: {"artist": {"birth_date": "..", "name": ".."}, "title": ".."}.

Let $\text{SameScope}(x_1, x_2)$ be a function of two sibling links that output *true* if their target data nodes are under the same nested object and *false* otherwise. Then, the features of this clique template are defined as:

$$f^{GP}_{ij}(y_1, y_2, x_1, x_2) = \mathbf{1}_{\{y_1=i\}}\mathbf{1}_{\{y_2=i\}}\text{SameScope}(x_1, x_2); \forall i, j \in \mathcal{V}$$

$$f^{\overline{GP}}_{ij}(y_1, y_2, x_1, x_2) = \mathbf{1}_{\{y_1=i\}}\mathbf{1}_{\{y_2=i\}}\neg\text{SameScope}(x_1, x_2); \forall i, j \in \mathcal{V}$$

3.2.6 Structural similarity. There are many possible ways to link two class nodes. Class nodes involved in the same relationship often connect to a similar set of data nodes. For example, in Figure 4, *Concept* nodes n_2, n_5 and n_8 link with *CulturalHeritageObject* nodes via the predicate *hasType* have similar values, which are types of artworks. While *Concept* nodes n_3 and n_6 involved in different relations (*dcterms:subject*) contain values about subjects in artworks. This heuristic can be used to predict the likelihood of links between class nodes. In particular, we define factor functions taking two links x_1 and x_2, which are class link and data link, respectively. The input x_1 and x_2 form a substructure. We say two substructures are partially matched if they both have the same link x_2. For example, in Figure 4,

[4]http://www.janm.org/ and http://museumca.org/

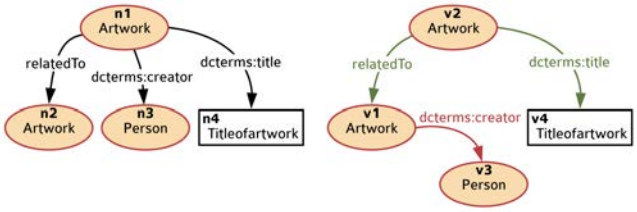


Figure 5: Example of a gold model and predicted model. The gold model is on the left side and the other is the predicted model. The green lines in the predicted model show correct links and red ones are incorrect links.

we have two substructures: $s_1 = \langle CulturalHeritageObject, hasType, Concept, prefLabel, classification\rangle$ and $s_2 = \langle CulturalHeritageObject, subject, Concept, prefLabel, classification\rangle$, in which $s2$ and $s1$ are partially matched.

The likelihood of a substructure is the maximum similarity score between its source attribute with other source attributes of other identical substructures in the training set. The reward of the substructure, Reward(x_1, x_2), is defined to be the difference between the likelihood of current substructure with the highest likelihood of other partial matched substructures. For example, in Figure 4, assuming the similarity score between attribute *classification* and attributes *technique_type*, *subject_tags*, *type_of_art*, *keywords* are 0.7, 0.5, 0.8, 0.2, respectively. Then, the likelihood of substructure $s_1 = max(0.7, 0.8) = 0.8$, and the likelihood of substructure $s_2 = 0.5$. The reward of the substructure $s_1 = 0.8 - 0.8 = 0$ while reward of $s_2 = 0.5 - 0.8 = -0.3$. Hence, our CRF will prefer s_1 to s_2.

Using Reward(x, y) function, the features of this clique template are defined as follow:

$$f_{ij}^{SrS}(y_1, y_2, x_1, x_2) = \mathbf{1}_{\{y_1=i\}}\mathbf{1}_{\{y_2=i\}}\text{Reward}(x_1, x_2); \forall i, j \in \mathcal{V}$$

$$f_{ij}^{\overline{SrS}}(y_1, y_2, x_1, x_2) = \mathbf{1}_{\{y_1=i\}}\mathbf{1}_{\{y_2=i\}} - \text{Reward}(x_1, x_2); \forall i, j \in \mathcal{V}$$

3.2.7 Error propagations. The factors of this clique template are used to remove class nodes in which all of their outgoing links are labeled *false* because they are redundant. In particular, the factors need to output high energy when all links of a class node are incorrect, and low energy when all of the outgoing links have label *false* but the incoming link is labeled *true*. Features of this clique template are defined as follow:

$$f_i^{EP}(y_0, \boldsymbol{y}_c, x_0, \boldsymbol{x}_c) = \mathbf{1}_{\{y_0=i\}}\mathbf{1}_{\{y=\text{false};\forall y\in\boldsymbol{y}_c\}}; \forall i \in \mathcal{V}$$

where x_0 is the incoming link, $\boldsymbol{x}_c$ are the outgoing links.

3.3 Training the Graphical Model

To train the graphical model, beside positive examples obtained from set of known semantic models, we also need negative examples. A typical approach to generate more training data is collecting generated semantic models from the search procedure. Then, every link in the generated models is labeled *true* or *false* using an automated labeling procedure.

3.3.1 Auto-label examples. Define rel(*sm*) is a set of triples (u, e, v) of semantic model *sm*, in which u, v is a source node and target node of a link e, respectively. The labeling procedure works as follow. First, we find the best mapping from node in the predicted model sm' to node in known model *sm* such that it maximizes the number of overlapped triples between rel(*sm*) and rel(sm'). For example, in Figure 5, if the mapping is $\{n_1 \to v_2, n_2 \to v_1, n_3 \to v_3, n4 \to v_4\}$, we have 1 overlapping triple $\langle Artwork, dcterms{:}title, Titleofartwork\rangle$. If we switch the mapping between two nodes n_1 and n_2, we have 2 overlapping triples and this is the best mapping we can have. After the alignment step, each link e in predicted model sm' is assigned a true label if its triple (u, e, v) is in rel(*sm*); otherwise, it is assigned *false*.

3.3.2 Generate training examples.

Algorithm 3: EliminateDataNodes

Input: A gold semantic model: *sm*
Output: List of new semantic models in which some data nodes have been removed

```
startStates ← []
sm_new ← remove data nodes in sm, which semantic labeling doesn't predict their semantic types correctly
for u ← sm_new.dataNodes do
    for stype ← PredictedSemanticTypes (u) do
        V_match ← all the data nodes have semantic type stype
        if |V_match | = 0 then
            state ← remove u out of sm_new
            add state to startStates
        for v ← V_match do
            if u is not v then
                state ← remove u and v out of sm_new
                add state to startStates
return startStates
```

To make the process of generating training examples efficient, instead of starting from an empty semantic model, we start from a known semantic model, remove some data nodes, then invoke the search algorithm in Section 3.1 to add back the removed attributes. The algorithm 3 describes how we remove data nodes. The goal is instead of removing random nodes, we remove the nodes that have same semantic types, so that when we add back a data node, there is always more than one possible way we can merge the data node into the model, hence generate a hard example for our graphical model to learn. For example, in Figure 1, suppose that we are removing the attribute *artist*, and *artist* is tagged with two possible semantic types: $\langle CulturalHeritageObject, title\rangle$ and $\langle Person, name\rangle$. After removing, we have two started states: the first one is the graph missing the links $\langle n_1_CulturalHeritageObject, title\rangle$ and $\langle n_4_Person, name\rangle$, the second state is omitting the links $\langle n_4_Person, name\rangle$ and $\langle n_5_Person, name\rangle$.

Note that as there are many possible examples that can be generated, each example is associated with a score computed using Taheriyan et al. [29] method, then we randomly sample the examples using the score to reduce the number of training examples. In our experiments, the number of examples is set to 300 per gold semantic model.

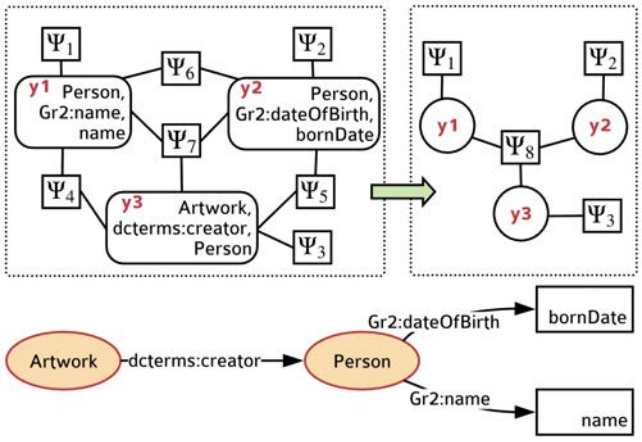


Figure 6: Factor graphs before and after eliminating cycles. In the factor graphs, we omit random variables $\boldsymbol{x}$ because their values are fixed.

3.3.3 Inference and Parameter Learning. We perform exact inference on CRF using Belief Propagation [26]. Undirected graphical models are often represented as factor graphs in which random variables and factor functions are nodes. An edge is drawn between a random variable and a factor if the variable is input of the factor function. Since we can only apply Belief Propagation to acyclic factor graphs, we eliminate cycles in our factor graphs by merging cliques which form cycles into single cliques. With the choices of factors in Section 3.2, we only need to merge factors, whose inputs are incoming links and/or outgoing links of same class nodes. For example, in the factor graph in Figure 6, the graph becomes acyclic after the four factors Ψ_4 to Ψ_7, which form cycles, are merged and replaced by a new factor:

$$\Psi_8(y_1, y_2, y_3, x_1, x_2, x_3) = \Psi_4(y_1, y_3, x_1, x_3)\Psi_5(y_2, y_3, x_2, x_3) \Psi_6(y_1, y_2, x_1, x_2)\Psi_7(y_1, y_2, y_3, x_1, x_2, x_3)$$

The problem with this approach is that the merged factors may have many variables, which results in the number of values of $\boldsymbol{y}$ we need to enumerate increasing exponentially with n: 2^n, where $n = |\boldsymbol{y}|$. Fortunately, it is not common to have a class node containing too many properties. In order to ensure the inference algorithm has reasonable runtime, we could control the number of variables in the factor by keeping top $n' < n$ most frequently used predicates. In the experiments, we set $n = 11$ so there are only maximum 2048 combinations of $\boldsymbol{y}$ values.

Our CRF is trained using the standard maximum log-likelihood and Belief Propagation. For more details about Belief Propagation and training CRF, readers can refer to the tutorial in Sutton et al. [26].

4 EXPERIMENTAL EVALUATION

4.1 Dataset and experimental setup

Our objective is to assess the ability of our method to infer correct semantic models with and without noisy data. We evaluate our approach, which is called PGM-SM, on two different datasets: ds_{edm} and ds_{crm} from Taheriyan et al. [29]. Both datasets contain data sources from different art museums in US, and in different data formats (CSV, XML, JSON, etc). However, domain experts use two different well-known data models in the museum domain: Europeana Data Model (EDM), and CIDOC Conceptual Reference Model (CIDOC-CRM) for knowledge representation. The details of two datasets are listed in Table 2. Note that we updated the semantic models in the ground-truth datasets to fix some mistakes based on the recommendation[5] from domain experts [12]. The updated datasets, experimental results and our code are available on Github[6].

Table 2: The evaluation datasets ds_{edm} and ds_{crm}

	ds_{edm}	ds_{crm}
#data sources	28	28
#classes in domain ontologies	119	147
#properties in domain ontologies	351	409
#nodes in the gold-standard models	444	970
#data nodes in the gold-standard models	310	552
#class nodes in the gold-standard models	134	418
#links in the gold-standard models	416	942

Table 3: MRR scores of three semantic labeling methods

Datasets	DSL	Serene	SemTyper
ds_{edm}	0.886	0.912	0.830
ds_{crm}	0.896	0.910	0.628

Semantic labeling methods are evaluated using standard mean reciprocal rank (MRR) [5]. We assess the quality of predicted semantic models by comparing with gold semantic models in term of precision and recall as in Taheriyan et al. [29]:

$$\text{precision} = \frac{|rel(sm) \cap rel(f^*(sm'))|}{|rel(f^*(sm'))|}$$

$$\text{recall} = \frac{|rel(sm) \cap rel(f^*(sm'))|}{|rel(sm)|}$$

$$f^* = \underset{f}{\text{argmax}}\, |rel(sm) \cap rel(f(sm'))|$$

where $rel(sm)$ is a set of triples (u, e, v) of a semantic model sm, f is a mapping function that maps nodes in sm' to nodes in sm (Section 3.3.1).

In our experiments, we use half of the dataset for training and the other half for testing. We repeat the process by three times (3-folds) and average the results. Parameters of our CRF are learned using the ADAM-AMSGRAD optimization method [21] for 60 epoches with batch size 200 and learning rate 0.05. Our experiments are run on a single machine with Intel Xeon E5-2620 v4 and 32GB RAM.

4.2 Automatic semantic modeling

In this experiment, we evaluate our method on four different semantic labelers: SemTyper [20], DSL [18], Serene [22] and Oracle.

[5]The recommendation documentation can be found at: http://review.americanartcollaborative.org/

[6]https://github.com/binh-vu/semantic-modeling

Table 4: Performances of semantic modeling systems with respect to different semantic labelers

Datasets	Labelers	Precision			Recall			F1		
		Taheriyan	Serene	PGM-SM	Taheriyan	Serene	PGM-SM	Taheriyan	Serene	PGM-SM
ds_{edm}	SemTyper	0.726	0.689	0.772	0.702	0.701	0.767	0.712	0.693	0.768
	DSL	0.656	0.708	0.812	0.618	0.734	0.820	0.635	0.719	0.815
	Serene	0.808	0.777	0.822	0.800	0.803	0.837	0.803	0.789	0.829
	Oracle	0.883	0.876	0.945	0.892	0.896	0.927	0.887	0.885	0.935
ds_{crm}	SemTyper	0.695	0.710	0.809	0.559	0.624	0.660	0.618	0.663	0.725
	DSL	0.699	0.714	0.851	0.693	0.687	0.839	0.695	0.698	0.844
	Serene	0.779	0.736	0.886	0.770	0.773	0.875	0.774	0.753	0.880
	Oracle	0.869	0.838	0.965	0.847	0.846	0.928	0.857	0.840	0.944

Table 5: Average F_1 score of PGM-SM with respect to different training size

Datasets	25% of dataset	50% of dataset
ds_{edm}	0.808	0.837
ds_{crm}	0.823	0.848

Table 6: Average running time of PGM-SM on two different datasets.

Datasets	Training time	Testing time
ds_{edm}	162.47s	9.54s
ds_{crm}	315.74s	59.37s

Each semantic labeler has different performance and characteristic, thus provides different levels of noise to our system. Of the three methods, SemTyper has the lowest MRR score indicating that its output includes many incorrect semantic types. While DSL has higher MRR score, thus provides fewer incorrect semantic types. However, the confidence score between predicted semantic types is similar since DSL predicts semantic labels based on a similarity notion. Serene has the best MRR performance, and the confidence score between correct and incorrect semantic types is very different. The Oracle semantic labeler is a labeler that always outputs the correct semantic type, thus provides no noise to the semantic modeling system. The performance of these semantic labelers is reported in Table 3.

We compare our method with two state-of-the-art semantic modeling systems: Taheriyan et al. [29] and Uña et al. [30] (Serene). The experiment results are reported in Table 4. Generally, the F_1 score increases with less noise for three systems. Among them, our approach outperforms the two baseline methods with the average increase by 6.53% and 10.92% on ds_{edm} and ds_{crm}, respectively. As both baseline methods predict semantic models by finding the minimum-weighted tree, they would be affected if the weights of edges are similar. Therefore, they are more sensitive to the noise of DSL. Our approach, however, uses the collective signals and achieves significant improvement, which indicates it is more robust to noise.

Table 5 shows the average F_1 score of PGM-SM when the number of known semantic models is 25% and 50% of the datasets. As the training size increases, the performance of PGM-SM increases. The average percentage of performance gain is 2.7%, which suggests that the model achieves reasonable performance even when the training size is small (25%). Comparing to the two baseline methods Taheriyan et al. [29] and Serene in the same scenario, PGM-SM improves the F_1 score on average 7.68% and 9.20%, respectively.

In addition, we measure the running time of our approach in terms of training and testing time listed in Table 6. The training time starts from generating training examples to finishing training PGM-SM. The testing time is total time to predict semantic models of the testing data sources. Since the size of semantic models in ds_{crm} are twice as big as the models in ds_{edm} (Table 2), we can see that the training time of ds_{crm} is nearly double ds_{edm}. The testing time does not increase correspondingly because it depends not only on the runtime of PGM but also the runtime of the search algorithm, which increases linearly with the number of attributes in the data sources.

4.3 Feature analysis

To understand the impact of each feature on the result, we perform ablation analysis by removing the factors described in Section 3.2. As the results in Table 7 have shown, dropping some factors sometimes slightly increases the F_1 score. In particular, STRUCT-SIM and ERROR-PROP are important in dataset ds_{crm}, while they are not in dataset ds_{edm}. The reason of this phenomenon is that the semantic models in ds_{crm} are very complex and structural while semantic models in ds_{edm} are not. For example, in ds_{edm}, a creation date of an artwork is modeled as a literal date time value. While in ds_{crm}, the date is represented as a class *E52_Time-Span* to capture the fact that it could take an artist many years to create an artwork. As those factors have little effect in ds_{edm}, dropping them actually makes the CRF model to learn better parameters of the remaining factors. On the other hand, CO-OCCUR and GROUPING factors

Table 7: Ablation analysis of the factors in PGM-SM. Each cell value is a difference in F_1 score upon dropping one factor.

Factor	Factor name	ds_{edm}				ds_{crm}			
		SemTyper	DSL	Serene	Oracle	SemTyper	DSL	Serene	Oracle
Structural Similarity	STRUCT-SIM	-0.009	0.004	0.020	0.008	-0.006	-0.013	-0.001	-0.015
Error Propagation	ERROR-PROP	0.005	0.005	0.004	0.000	-0.006	-0.016	-0.001	0.003
Co-occurrence	CO-OCCUR	-0.006	-0.014	-0.008	-0.005	0.024	-0.008	0.010	0.004
Grouping properties	GROUPING	0.008	-0.011	-0.003	0.008	-0.007	-0.012	0.004	0.003
Duplicated properties	DUP-PROP	-0.003	-0.009	-0.005	0.000	-0.007	-0.008	-0.003	-0.007
Cardinality relationships	CARDINALITY	-0.006	-0.020	-0.013	-0.023	-0.014	-0.049	-0.022	-0.032

show the opposite trend. DUP-PROP and CARDINALITY have positive impact on the F_1 score for all datasets and CARDINALTIY is the most important feature in our CRF model.

5 RELATED WORK

There are many studies to automatically integrate data from multiple sources. Although their goals are similar, they are different in terms of inputs and assumptions and can be clustered into three groups.

The first group focuses on problems of schema matching and mapping [3]. Schema matching generates correspondences between elements of two schemas [4, 6, 19]. For instance, Madhavan et al. [15] combine information from schema constraints (data types, foreign keys), structure and values in data sources to find one to one correspondence between elements in source schema and target schema. Schema mapping generates mapping formulas from schema matches to transform data from source schema to target schema [1, 2, 8, 9, 16, 23]. In some recent studies, Kimmig et al. [10] use probabilistic soft logic to explore information from both data examples and schema constraints to generate the mappings. The semantic modeling problem, however, is different from schema matching and mapping [29]. First, in addition to finding mappings from attributes in a data source to ontology classes and properties, we also generate relationships between the attributes. Moreover, as data sources on the Web are often lack of schema constraints and have different conventions, semantic modeling provides a straightforward approach to represent and publish the data to knowledge graph.

The second group focuses on describing semantics of web tables. Limaye et al. [14] use a probabilistic graphical model and an external knowledge base YAGO to annotate tables with entities for cells, semantic types for columns and binary relationships between columns. Mulwad et al. [17] share similar ideas with Limaye et al., in which they leverage Linked Open Data (LOD) such as DBpedia, YAGO and Wikitology to link table cells with LOD entities. After generating initial lists of candidates for cell values, they perform inference to compute the assigned types of column, cell linkage and relations between columns. Instead of using available knowledge graphs, Venetis et al. [31] build two databases of class of entities and relationships between the entities from text documents available on the Web. The column labels are assigned to maximize the probability of column values given class labels of entities. All of these approaches rely on external knowledge bases which provide possible links between cell values and known entities. Hence, performance of these approaches highly depends on the quality of the linkages. Furthermore, as not all data attributes have corresponding entities in knowledge base, e.g: birth date, phone, age; these approaches may not be applicable to a broad range of data sources and be restricted to domains where online data is widely available.

Our problem is in the final group, in which semantic models are used to precisely describe meanings of data sources and publish data to knowledge graph. The semantic models can represent a wide range of data sources and in various formats such as CSV, XML, JSON, etc. To address the semantic modeling problem, Taheriyan et al. [28, 29] build an integration graph that represents space of plausible semantic models for a new data source. Each link in the integration graph is associated with a weight representing the frequency of use of the link in previous models. First, attributes of a new data source are associated with data nodes in the integration graph. Then, the semantic model is built by finding the Steiner Tree that connects the data nodes. Uña et al. [30] also predict semantic models by finding Steiner Tree. However, to solve the Steiner Tree problem, instead of using an approximation algorithm as in Taheriyan et al. [29], they use constraint programming, an exact algorithm, which allows it to incorporate more features as additional constraints. Instead of solving the Steiner Tree problem, our approach uses a PGM to exploit collective signals from a data source. The PGM allows us to express complex dependencies between the features of good and bad semantic models. Therefore, we are able to obtain better performance than the previous state-of-the-art methods.

6 CONCLUSION AND FUTURE WORK

In this paper, we presented a novel approach for semantic modeling that uses a probabilistic graphical model to exploit relationship within data sources and semantic models. From a set of data sources and their semantic models, we train a conditional random field (CRF) to distinguish between good and bad semantic models. Then, we use the CRF to efficiently search over the combinatorial space of possible semantic models to identify the most probable model for a data source. The evaluation shows that by exploring the relationships within data, our approach generates better semantic models and is more robust to noise than previous approaches.

Because labeled data is costly to obtain, a future direction of our work is to exploit transfer learning to leverage a vast amount of available linked datasets in the Linked Open Data. For example, we can train a CRF on linked datasets, which are represented using the same domain ontologies, and reuse them as a pre-trained model. We can also enrich our training set by applying ontology matching techniques to align linked data from different ontologies to our ontologies.

Another direction for future work is to integrate semantic modeling with web data extraction. Unsupervised web data extraction methods such as Trinity [24] can extract structural data from a set of web pages. However, the output of these methods often includes many redundant data attributes. By leveraging more information from the extraction step such as the position of extracted attributes in the web pages and collective information between data attributes, we can potentially improve performance of systems that automatically extract and semantically annotate web data.

ACKNOWLEDGMENTS

This material is based upon the work supported by the Defense Advanced Research Projects Agency (DARPA) under award W911NF-18-1-0027. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA. We thank the anonymous reviewers for their helpful comments on the paper.

REFERENCES

[1] Bogdan Alexe, Balder Ten Cate, Phokion G Kolaitis, and Wang-Chiew Tan. 2011. Designing and refining schema mappings via data examples. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 133–144.

[2] David Aumueller, Hong-Hai Do, Sabine Massmann, and Erhard Rahm. 2005. Schema and ontology matching with COMA++. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. Acm, 906–908.

[3] Zohra Bellahsène, Angela Bonifati, and Erhard Rahm. 2011. *Schema matching and mapping*. Springer.

[4] Philip A Bernstein, Jayant Madhavan, and Erhard Rahm. 2011. Generic schema matching, ten years later. *Proceedings of the VLDB Endowment* 4, 11 (2011), 695–701.

[5] Nick Craswell. 2009. Mean reciprocal rank. In *Encyclopedia of Database Systems*. Springer, 1703–1703.

[6] Robin Dhamankar, Yoonkyong Lee, AnHai Doan, Alon Halevy, and Pedro Domingos. 2004. iMAP: discovering complex semantic matches between database schemas. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. ACM, 383–394.

[7] Anastasia Dimou, Miel Vander Sande, Jason Slepicka, Pedro Szekely, Erik Mannens, Craig A. Knoblock, and Rik Van de Walle. 2014. Mapping hierarchical sources into RDF using the RML mapping language. In *IEEE International Conference on Semantic Computing (ICSC)*. IEEE, 151–158.

[8] Ronald Fagin, Laura M Haas, Mauricio Hernández, Renée J Miller, Lucian Popa, and Yannis Velegrakis. 2009. Clio: Schema mapping creation and data exchange. In *Conceptual modeling: foundations and applications*. Springer, 198–236.

[9] Ronald Fagin, Phokion G Kolaitis, Renée J Miller, and Lucian Popa. 2005. Data exchange: semantics and query answering. *Theoretical Computer Science* 336, 1 (2005), 89–124.

[10] Angelika Kimmig, Alexander Memory, Renee J Miller, and Lise Getoor. 2018. A Collective, Probabilistic Approach to Schema Mapping Using Diverse Noisy Evidence. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[11] Craig A. Knoblock, Pedro Szekely, Jose Luis Ambite, Shubham Gupta, Aman Goel, Maria Muslea, Kristina Lerman, Mohsen Taheriyan, and Parag Mallick. 2012. Semi-Automatically Mapping Structured Sources into the Semantic Web. In *Proceedings of the Extended Semantic Web Conference*. Crete, Greece.

[12] Craig A. Knoblock, Pedro Szekely, Eleanor Fink, David Newbury Duane Degler, Robert Sanderson, Kate Blanch, Sara Snyder, Nilay Chheda, Nimesh Jain, Ravi Raju Krishna, Nikhila Begur Sreekanth, and Yixiang Yao. 2017. Lessons Learned in Building Linked Data for the American Art Collaborative. In *ISWC 2017 - 16th International Semantic Web Conference*.

[13] Daphne Koller and Nir Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*.

[14] Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. 2010. Annotating and Searching Web Tables Using Entities, Types and Relationships. *Proc. VLDB Endow.* 3, 1-2 (Sept. 2010), 1338–1347. https://doi.org/10.14778/1920841.1921005

[15] Jayant Madhavan, Philip A Bernstein, and Erhard Rahm. 2001. Generic schema matching with cupid. In *vldb*, Vol. 1. 49–58.

[16] Bruno Marnette, Giansalvatore Mecca, Paolo Papotti, Salvatore Raunich, Donatello Santoro, et al. 2011. ++ Spicy: an Open-Source Tool for Second-Generation Schema Mapping and Data Exchange. *Clio* 19 (2011), 21.

[17] Varish Mulwad, Tim Finin, and Anupam Joshi. 2013. Semantic message passing for generating linked data from tables. In *International Semantic Web Conference*. Springer, 363–378.

[18] Minh Pham, Suresh Alse, Craig Knoblock, and Pedro Szekely. 2016. Semantic labeling: A domain-independent approach. In *ISWC 2016 - 15th International Semantic Web Conference*.

[19] Erhard Rahm and Philip A Bernstein. 2001. A survey of approaches to automatic schema matching. *the VLDB Journal* 10, 4 (2001), 334–350.

[20] S.K. Ramnandan, Amol Mittal, Craig A. Knoblock, and Pedro Szekely. 2015. Assigning Semantic Labels to Data Sources. In *Proceedings of the 12th ESWC 2015*.

[21] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. 2018. On the Convergence of Adam and Beyond. In *ICLR 2018 : International Conference on Learning Representations 2018*.

[22] Natalia Rümmele, Yuriy Tyshetskiy, and Alex Collins. 2018. Evaluating approaches for supervised semantic labeling. (2018).

[23] Satya S. Sahoo, Juan Sequeda, and Ahmed Ezzat. 2009. A Survey of Current Approaches for Mapping of Relational Databases to RDF. (2009).

[24] Hassan A Sleiman and Rafael Corchuelo. 2014. Trinity: on using trinary trees for unsupervised web data extraction. *IEEE Transactions on Knowledge and Data Engineering* 26, 6 (2014), 1544–1556.

[25] Jason Slepicka, Chengye Yin, Pedro Szekely, and Craig A. Knoblock. 2015. KR2RML: An Alternative Interpretation of R2RML for Heterogenous Sources. In *Proceedings of the 6th International Workshop on Consuming Linked Data (COLD 2015)*.

[26] Charles A. Sutton and Andrew McCallum. 2012. An Introduction to Conditional Random Fields. *arXiv preprint arXiv:1011.4088* 4, 4 (2012), 267–373.

[27] Pedro Szekely, Craig A. Knoblock, Jason Slepicka, Andrew Philpot, Amandeep Singh, Chengye Yin, Dipsy Kapoor, Prem Natarajan, Daniel Marcu, Kevin Knight, David Stallard, Subessware S. Karunamoorthy, Rajagopal Bojanapalli, Steven Minton, Brian Amanatullah, Todd Hughes, Mike Tamayo, David Flynt, Rachel Artiss, Shih-Fu Chang, Tao Chen, Gerald Hiebel, and Lidia Ferreira. 2015. Building and Using a Knowledge Graph to Combat Human Trafficking. In *Proceedings of the 14th International Semantic Web Conference (ISWC 2015)*.

[28] Mohsen Taheriyan, Craig Knoblock, Pedro Szekely, and Jose Luis Ambite. 2016. Leveraging Linked Data to Discover Semantic Relations within Data Sources. In *ISWC 2016 - 15th International Semantic Web Conference*.

[29] Mohsen Taheriyan, Craig A. Knoblock, Pedro Szekely, and Jose Luis Ambite. 2016. Learning the semantics of structured data sources. *Journal of Web Semantics* (2016).

[30] Diego De Uña, Nataliia Rümmele, Graeme Gange, Peter Schachte, Peter J. Stuckey, and Peter J. Stuckey. 2018. Machine Learning and Constraint Programming for Relational-To-Ontology Schema Mapping. In *IJCAI 2018: 27th International Joint Conference on Artificial Intelligence*. 1277–1283.

[31] Petros Venetis, Alon Halevy, Jayant Madhavan, Marius Paşca, Warren Shen, Fei Wu, Gengxin Miao, and Chung Wu. 2011. Recovering semantics of tables on the web. *Proceedings of the VLDB Endowment* 4, 9 (2011), 528–538.

Generalists and Specialists: Using Community Embeddings to Quantify Activity Diversity in Online Platforms

Isaac Waller
University of Toronto
walleris@cs.toronto.edu

Ashton Anderson
University of Toronto
ashton@cs.toronto.edu

ABSTRACT

In many online platforms, people must choose how broadly to allocate their energy. Should one concentrate on a narrow area of focus, and become a specialist, or apply oneself more broadly, and become a generalist? In this work, we propose a principled measure of how generalist or specialist a user is, and study behavior in online platforms through this lens. To do this, we construct highly accurate *community embeddings* that represent communities in a high-dimensional space. We develop sets of community analogies and use them to optimize our embeddings so that they encode community relationships extremely well. Based on these embeddings, we introduce a natural measure of activity diversity, the GS-score.

Applying our embedding-based measure to online platforms, we observe a broad spectrum of user activity styles, from extreme specialists to extreme generalists, in both community membership on Reddit and programming contributions on GitHub. We find that activity diversity is related to many important phenomena of user behavior. For example, specialists are much more likely to stay in communities they contribute to, but generalists are much more likely to remain on platforms as a whole. We also find that generalists engage with significantly more diverse sets of users than specialists do. Furthermore, our methodology leads to a simple algorithm for community recommendation, matching state-of-the-art methods like collaborative filtering. Our methods and results introduce an important new dimension of online user behavior and shed light on many aspects of online platform use.

KEYWORDS

generalist and specialists, community embeddings, activity diversity, community recommendation

ACM Reference Format:
Isaac Waller and Ashton Anderson. 2019. Generalists and Specialists: Using Community Embeddings to Quantify Activity Diversity in Online Platforms. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313729

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313729

1 INTRODUCTION

In many domains of human endeavor, people must choose how broadly to allocate their energy. Should one be a *specialist*, and concentrate in depth on a narrow area of focus, or be a *generalist*, and apply oneself more broadly? In the past century, there has been a steady movement towards specialization in professional domains, including business, medicine, and academia [5, 12, 22], while in the past decade, there has been a rise of generalists, especially in entrepreneurship and in the tech industry [6, 21].

The trade-off between breadth and depth of activity is also a fundamental characteristic of user behavior in online platforms. A user can concentrate on a narrow range of activities and reap the rewards of specializing in some area of the platform, or one can maintain a more diverse set of interests at the cost of not engaging as deeply with them. Whether the choice is explicitly made or not, the scope of a user's activity ranges between being narrow and broad. There exists a continuum of possibilities between extreme specialists on one side and extreme generalists on the other, and in general users will be somewhere in the middle.

Activity Diversity in Online Platforms. The question of how diverse users' activities are illuminates a number of important issues. First, the extent to which users are generalists or specialists indicates how connected an online platform is on a global level. If most users lean towards being specialists, then the platform will tend to consist of isolated communities with little interaction between them, whereas if most users lean towards being generalists, then the platform will tend to be more cohesive and support broader engagement between members of different communities. Second, users' activity diversity styles can be indicative of the underlying incentives they face. As in medicine and academia, where the prevailing incentives once rewarded generalists but now typically steer professionals towards specialization, a preponderance of specialists or generalists in online platforms could reveal how users perceive their incentive structures. Third, the relation between breadth and depth of activity and resulting success and longevity has been the source of rich debates in other contexts [2, 18]. The study of how this plays out in online platforms could contribute to this discussion.

Despite the fundamental nature of the activity diversity problem, and its relative simplicity to state, it is difficult to study rigorously for several reasons. In the simplest form of the problem, we are given a set of activities supported by an online platform, and empirical data on what activities individual users choose to pursue. The main obstacles in measuring which users are generalists and which are specialists are twofold: first, coming up with measures of similarity between all pairs of activities, and second, ensuring that these similarities are consistent with each other. In prior work, researchers have mainly used entropy as a measure of activity diversity [7]. Entropy is very sensible in many scenarios, and is sensitive

to both the number and distribution of activities. However, it is agnostic to the similarities or differences between activities. Thus, a user whose activities are all very similar to each other would be treated the same as a user with the same distribution of engagement across activities that are very different from each other, contradicting the intuition that the former is more of a specialist.

The Present Work: Community Embeddings and the GS-Score. In this work, we present a methodology to study generalists and specialists in online platforms, and we analyze, characterize, and predict user behavior on Reddit and GitHub through this lens. On Reddit, a discussion-based content aggregation platform, we are interested in how users participate in various communities (*subreddits*) as defined by commenting on posts in those communities; and on GitHub, a social coding platform, we are interested in how users interact with (contribute, star, or watch) various code repositories.

To solve the problem of deriving high-fidelity, consistent similarities between communities, we develop and considerably extend *community embeddings*. Following previous work [11, 15], we adapt word2vec, a word embedding algorithm that assigns a vector to each word in a training corpus such that relationships between words are preserved, to assign a vector to each community in our data such that relationships between communities are preserved. In word embeddings, analogical reasoning can surprisingly be done with simple vector arithmetic. Our first contribution is developing a set of ground-truth community analogies for both Reddit and GitHub that can be used to optimize the embedding algorithm hyperparameters and vastly enhance the quality of the resulting community embeddings. In our final embeddings, analogy questions like `toronto : torontoraptors :: chicago : ___?` are answered correctly (`chicagobulls`) with extremely high accuracy.

Using our community embeddings, we are able to quantify the similarity between any two communities as the cosine similarity between their respective vector representations. Furthermore, these similarities are mutually consistent, as a result of being derived from the geometry of the same vector space. We represent a user's activities as a distribution over points in this vector space. Intuitively, a specialist's activity is concentrated in the space, and a generalist's activity is diffuse in the space (see Figure 1 for a schematic depiction). We calculate a user's *center of mass* as being the centroid of the communities they contribute to, and we define the *Generalist-Specialist score* of a user's activity to be the average cosine similarity between their communities and their center of mass, weighted by activity. The GS-score is minimized by specialists who contribute to a tight cluster of communities and maximized by generalists who contribute to communities that are far apart. We developed an online quiz to elicit human judgments about generalist and specialist users and found that they correlate very well with the GS-score. This principled measure of activity diversity in online platforms, using high-fidelity and validated community similarities calculated from community embeddings, is a key contribution of our work.

Overview of Results. Our results fall into two categories, according to our two main units of analysis: the user and the community.

We first focus on users. Applying the GS-score to Reddit and GitHub users, we observe a great diversity of user activity styles, ranging from extreme generalists to extreme specialists, at every level of activity. Compared to a random baseline, there are many more specialists than expected. On Reddit, we analyze how users' comments are voted on, and find that specialists do about 20% better in a controlled analysis. We then turn to studying user longevity, both in individual communities and on the platform as a whole. We find that specialists are much more likely than generalists to convert into long-term members of a particular community, suggesting that specialists are more selective. However, we also observe that generalists are much more likely to remain on the platform, with the effect being particularly stark on Reddit. These results have interesting implications for community managers and platform designers; on the one hand a community manager should probably seek out specialists to join their community, but on the other hand platform designers should try to be amenable to generalists and encourage exploration. Finally, we measure the diversity of the subpopulations users interact with, and find that specialists interact with significantly less diverse sets of users than generalists do.

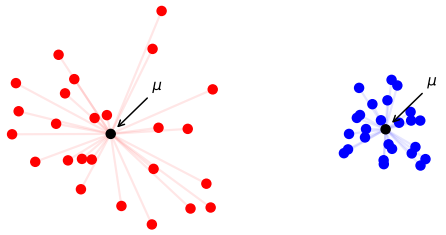


Figure 1: A schematic depicting the vector representations of communities contributed to by a generalist (left) and a specialist (right). The generalist's communities are spread out, and the specialist's communities are clustered together.

We then turn our attention to communities, and make the observation that we can define the activity diversity of a *community* to be the average activity diversity of its users. We find this *community activity diversity* score to be remarkably stable, remaining relatively constant for at least three years, even when the underlying user base turns over significantly. We also analyze which users make up the "elites" in a community and find that they skew generalist.

Finally, we apply our measure in a prediction framework and focus on two concrete, important tasks. The first is community recommendation, in which we predict which communities users will join. We develop a simple algorithm to recommend communities based on their proximity to a user's center of mass and demonstrate that it performs at least as well as state-of-the-art collaborative filtering. Furthermore, we find that specialists are much more predictable than generalists. The second problem is user retention, in which we predict if users will stay on the platform. We show that using activity diversity alone performs very well.

The rest of the paper is laid out as follows. In Section 2, we discuss the datasets, develop community embeddings, and introduce our metric of activity diversity, the GS-score; in Section 3 we analyze behavior in Reddit and GitHub through the lens of activity diversity; in Section 4 we perform our prediction tasks; in Section 5 we outline related work; and in Section 6 we discuss our work and conclude.

2 METHODOLOGY

The main goal of our work is to precisely define a measure of activity diversity that smoothly interpolates between generalists

and specialists, and apply it to the study of user behavior in online platforms. In this Section, we describe our datasets, develop our methodology to construct high-fidelity community embeddings, validate these embeddings as being accurate representations of the platforms we study, and introduce our measure of activity diversity.

2.1 Data

First we introduce the online platform datasets we use throughout the rest of the paper. We study two online platforms in depth: Reddit, a discussion-based content aggregation platform, and GitHub, a collaborative coding platform.

Reddit. Our main dataset is derived from Reddit, an online platform where users can share, view, and discuss content from around the Web. Every post is either a piece of free-standing text or a URL, and there is a discussion page where users can discuss the post. Reddit is subdivided into communities called *subreddits* that are organized around some theme; for example `r/gardening` is about gardening and `r/learnprogramming` is for people learning how to program. We are interested in user participating in these communities by *commenting* in them, i.e. by participating in discussions on discussion pages. Comments can also be voted on by other users, and each has a vote score associated with it.

Our Reddit dataset is comprised of all Reddit user comments in 2017, totalling 900M comments authored by 11.4M distinct users in 232K subreddits. We restrict our attention to the top 10,000 subreddits by activity, which account for 96.8% of all comments. The data we used is publicly available and can be downloaded from `pushshift.io` [3].

GitHub. We supplement our Reddit analyses with a dataset of GitHub activities. GitHub is an online platform where users can collaborate on coding projects that are divided into *repositories* of self-contained projects. Popular repositories frequently support a vibrant community of contributors. We treat repositories as communities, and consider adding code and bookmarking to be user contributions to these communities.

Our GitHub dataset is comprised of all GitHub commits, pull requests, forks, watches, and stars in 2017, totalling 413M actions by 8.3M distinct users in 26M repositories. Since the vast majority of repositories have very few contributors (usually 1), we restrict our attention to the top 40,000 communities by number of stars. The data we used is publicly available from the GH Archive and can be downloaded from `gharchive.org`.

2.2 Community Embeddings

As mentioned in the Introduction, a chief difficulty in formulating a measure of activity diversity is coming up with accurate similarities between all pairs of communities. Without these, it is hard to distinguish between a user who is active in two very related communities and a user who is active in two very different communities, despite the fact that intuitively the former should be treated as more specialist than the latter. The most commonly used metric for capturing user diversity among activities is entropy, which is agnostic to community similarities, and therefore treats these two users as being equally diverse in their activities.

To solve this problem, we construct high-fidelity *community embeddings*. Drawing inspiration from the major successes achieved by applying word embeddings to various natural language processing tasks, and following recent work [11, 15], we adapt the word2vec word embedding algorithm to apply to user-community interaction data to generate community embeddings. Treating communities as "words" and users who comment in them as "contexts", we embed communities into a high-dimensional vector space. Two communities are close together if users frequently comment in them both, and are far apart if they are rarely commented in by the same users. We use the skip-gram model with negative sampling and train over all pairs (c_i, u_j) of user u_j commenting in community c_i.

Using the same hyperparameters as those used in the original word2vec model, we generated an initial community embedding. It seems to work reasonably well — for example, the nearest communities to `r/Fishing` (those with highest cosine similarity with `r/Fishing`) are focused on other outdoor pursuits, e.g. `r/discgolf`. However, it is unclear how to objectively evaluate the fidelity of this embedding.

This problem of objectively measuring embedding quality also arises with word embeddings. There, researchers and practitioners solve this problem by making use of the surprising fact that embeddings preserve semantic relationships. For example, in a well-tuned word embedding, the analogy `man : king :: woman : ___`? can be correctly solved with `queen` by performing vector arithmetic on the corresponding word vectors ($\overrightarrow{\text{king}} - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}} \approx \overrightarrow{\text{queen}}$). By generating a set of word analogies and then measuring how many of them a word embedding can solve by performing vector arithmetic, one can measure the quality of a word embedding.

Community analogies. To evaluate our community embeddings, we adapt this methodology to our community setting and develop large sets of ground-truth *community analogies*, which can be used to objectively evaluate the quality of a community embedding. Each analogy is a tuple (a, a', b, b') of communities such that $a : a' :: b : b'$, and it should be the case that $\overrightarrow{a'} - \overrightarrow{a} + \overrightarrow{b} \approx \overrightarrow{b'}$. We found three different classes of objective semantic relationships on Reddit communities that we can generate analogies with: university → city, sports team → city, and sports team → sport. For example, for the first class of analogies, university communities (e.g. `r/Harvard`) should all have roughly the same vector relationship with the communities of the city they are located in (e.g. `r/CambridgeMA`). Thus, it should be the case that e.g., $\overrightarrow{\texttt{r/Harvard}} - \overrightarrow{\texttt{r/CambridgeMA}} \approx \overrightarrow{\texttt{r/mcgill}} - \overrightarrow{\texttt{r/montreal}}$. Example analogies of each type are shown in Table 1. For each class of analogies, we generated a list of pairs of communities with the same semantic relationship, then took the Cartesian product of these pairs to construct all possible analogies for this class. Across our three classes, we constructed 4,392 analogies in total[1].

We can now validate our embedding. For each analogy, we rank all vector representations of communities in ascending order of distance to the point $\overrightarrow{a'} - \overrightarrow{a} + \overrightarrow{b}$. The accuracy metric Precision@k measures the fraction of analogies for which $\overrightarrow{b'}$ is among the k closest communities, and the special case Precision@1 measures what fraction of analogies are perfect, i.e. $\overrightarrow{b'}$ is the closest community to

[1]We have made our sets of ground-truth community analogies available at https://osf.io/jcz3p/.

Table 1: Example community analogies from our collection.

a	a'	b	b'	
UCSD	sandiego	georgetown	washingtondc	university to city
Harvard	CambridgeMA	mcgill	montreal	
Northwestern	evanston	UCL	london	
vancouver	canucks	Detroit	DetroitRedWings	city to team
toronto	torontoraptors	chicago	chicagobulls	
Seattle	Seahawks	oakland	oaklandraiders	
SFGiants	baseball	49ers	nfl	team to sport
lakers	nba	Dodgers	baseball	

the point $\overrightarrow{a'} - \overrightarrow{a} + \overrightarrow{b}$. Our initial embedding scores 30% Precision@1 on our set of 4,392 validation analogies.

Hyperparameter search. To optimize our embedding, we conducted a sweep over the space of word2vec parameters, generating a separate community embedding for every combination of parameter values and evaluating all of these embeddings with our analogy sets. The parameters we varied are the learning rate α, the dimensionality of the vector space, the number of negative samples, and the sampling rate. How analogy performance varies with parameters is shown in Figure 2. Optimizing the parameters in this way drastically improves the resulting community embedding. Our final embedding scores 72% Precision@1 and 96% Precision@5 on our analogy test, a massive improvement over our initial embedding scores. We show a two-dimensional projection of our final embedding in Figure 3 centered on `r/Fishing`.

Optimizing the embedding with analogies has achieved a difference in kind in the fidelity of the resulting embedding; now all of the closest communities are extremely related (e.g. `r/bassfishing`, `r/kayakfishing`, and `r/flyfishing` are the top three closest communities). We emphasize that the fact this is even possible was not a priori obvious. Community embeddings preserve semantic relationships between communities much as word embeddings do with words. Furthermore, similarly to word embeddings, optimizing community embeddings with analogies drawn from a small number of classes of semantic relationships seems to generally align the entire vector space, even for relationships we didn't train on. The vast difference in quality between the neighborhoods of `r/Fishing` in the optimized and unoptimized embeddings is typical. Also, we observe that other types of analogies suddenly work, even though they weren't explicitly trained; for example, the analogy $\overrightarrow{\text{swimming}} + \overrightarrow{\text{cycling}} + \overrightarrow{\text{running}} \approx \overrightarrow{\text{triathlon}}$ is present in our embedding, despite the fact that we didn't train any ternary relationships at all. Finally, we note that there are interesting patterns in embedding quality in the hyperparameter space — even small changes in parameters can lead to sharp drops in embedding quality, as can be seen in the ridge that cuts across the middle of Figure 2. This suggests it is important to understand where in the hyperparameter space one's embedding is.

Generating community embeddings helps us overcome two major obstacles to measuring activity diversity. First, we can now measure the similarity between any pair of communities by calculating the cosine similarity between their vector representations.

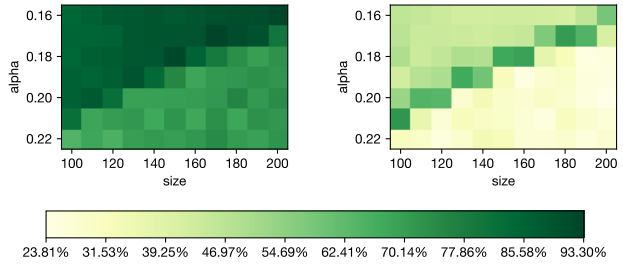


Figure 2: Heatmap of Precision@1 analogy test scores as a function of embedding algorithm hyperparameters on university to city (left) and North American sports (right).

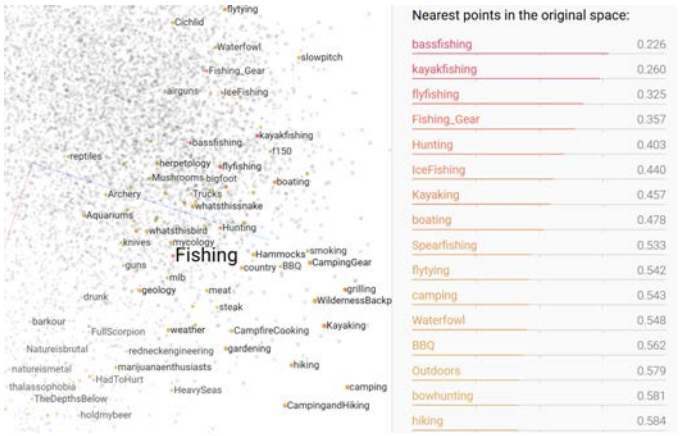


Figure 3: Two-dimensional projection of our final embedding, centered on `r/Fishing`, after performing hyperparameter search with our community analogies.

This similarity is entirely data-driven, and is in a way crowdsourced: two communities are similar if and only if many people choose to comment in them both. Other methods, such as hand-crafted taxonomies or expert classifications, are potentially biased by subjective opinions. Second, our similarities are consistent with each other, since they are derived from cosine similarities between vectors embedded in the same vector space.

2.3 Measuring Activity Diversity

We are now ready to define our measure of activity diversity. Consider a user who is active in some subset of communities. Now, if we consider their activity through the lens of our community embedding, their activity is distributed through the space according to where their communities are. Intuitively, a specialist user's activity will be concentrated in this space, as their communities are similar to each other, and a generalist user's activity will be diffuse in this space, as their communities are different from each other.

With this intuition in mind, we formulate the *Generalist-Specialist score*, or *GS-score*, as follows. Say user u_i makes w_j contributions to community c_j, and in a slight abuse of notation let $\vec{c_j}$ denote c_j's vector representation in our community embedding (note we normalize all vectors to be unit length). First we define u_i's *center of mass* to be $\vec{\mu_i} = \sum_j w_j \vec{c_j}$. Then the definition of u_i's activity diversity is very natural — it is simply the average cosine similarity between u_i's communities and his center of mass, weighted by number of contributions by community:

$$GS(u_i) = \frac{1}{J} \sum_j w_j \frac{\vec{c_j} \cdot \vec{\mu_i}}{||\mu_i||}$$

where J is the number of communities u_i is active in. When the context is clear we will refer to u_i's GS-score as GS_i. A simple interpretation of the GS-score is that it is equal to the expected cosine similarity between a user's contributions and his center of mass in the community embedding (where we say a contribution is located at the vector representation of the community it was contributed to). The GS-score ranges between -1 and 1, where -1 is an extreme generalist and 1 is an extreme specialist. In practice, the GS-score typically ranges between 0.5 and 1.

This definition fulfills a number of desiderata. First, as the communities one contributes to become more similar to each other, the measure increases and moves towards the specialist end of the spectrum. Second, as users contribute more heavily in one community, they become more specialist. Third, fixing a user's contributions and adding another contribution to a new community pushes the measure towards the generalist end.

Interestingly, the GS-score is related to another natural definition of activity diversity. Instead of comparing contributions to the center of mass, we could imagine comparing communities to each other. Consider the *all-pairs community similarity*: $\frac{1}{C} \sum_i \sum_j \vec{c_i} \cdot \vec{c_j}$, where here the sums are over contributions, $\vec{c_i}$ is the vector representing the community of the i-th contribution, and C is the total number of contributions. This quantity measures the expected similarity between two of a user's contributions. This definition is also intuitive: the closer a user's contributions are on average, the most specialist the user. After simple algebra, we derive that the all-pairs community similarity is equivalent to the square of the GS-score. This is encouraging: two different ways of defining activity diversity are rank-equivalent, which indicates that we are capturing a real property of a user's distribution over activities.

Validating the GS-score. The GS-score is natural, intuitive, and is rank-equivalent with another intuitive definition. How well does it capture human judgments of generalists and specialists? To answer this question, we developed an online quiz asking people to compare two Reddit users at a time and decide which one they think is more specialist. By eliciting human judgments in this way, we can compare human responses against our measure and measure the extent to which the two agree.

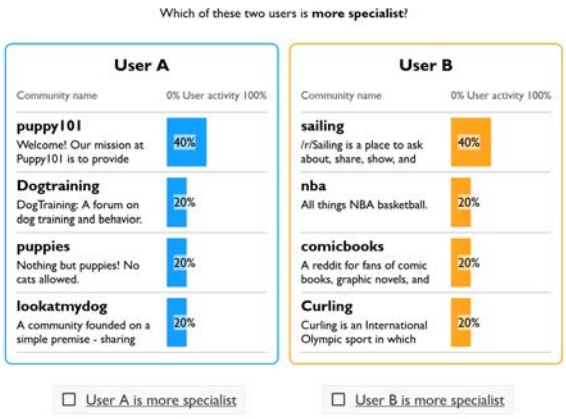


Figure 4: Screenshot of the quiz asking people to rank which user is more specialist.

In order to elicit high-quality feedback, we developed the quiz as a game, rather than paying for responses on a crowdsourcing platform, under the assumption that people would be more inclined to answer in good faith if they were internally motivated than if they were externally motivated. When a user arrived at the quiz, they were shown an initial screen explaining, "In this quiz, you will be presented with a list of communities (subreddits) someone is a part of. The bar shows what percentage of their activity is in this community. At each step, you will be shown two users. Click the button below the user which is **more of a specialist**. A specialist is **someone whose interests focus on a particular topic**." On every subsequent screen, they were presented with one question, for example the one shown in Figure 4. Every question compares two users with exactly 4 communities each, shows the name and description of each user's communities, and the distribution of their contributions among these communities. The quiz-taker is then asked to select which user they think is more specialist.

In all, we elicited judgments on 1,807 pairs of users from 144 unique quiz-takers. Examining the relationship between how these judgments compare with the GS-score, we observe a strong correlation between the difference in GS-scores of pairs of users and the proportion of quiz-takers who agree with the GS-score (i.e. say the user with the higher GS-score is more specialist). As the difference in GS-scores grows, human judgments become more in line with the score. This is exactly what we want: as two users' activity diversity scores become closer together, it is more of a close call, and thus human judgments are less unanimous. As they become further apart, human judgments are more unanimous, and approach 90% agreement with the GS-score. Recall that even when the GS-scores of a pair of users are similar, the underlying communities they are contributing to are often very different. Thus it is reassuring that when we predict their activity diversity to be quite close, humans have a correspondingly difficult time telling them apart. The Pearson correlation between the difference in GS-scores and proportion

of quiz-takers agreeing with the GS-score is 0.94. The GS-score thus accords very well with human judgments of activity diversity.

3 ACTIVITY DIVERSITY IN ONLINE PLATFORMS

Now that we have defined and validated a measure of activity diversity in online platforms, we apply our metric and study user behavior through this lens.

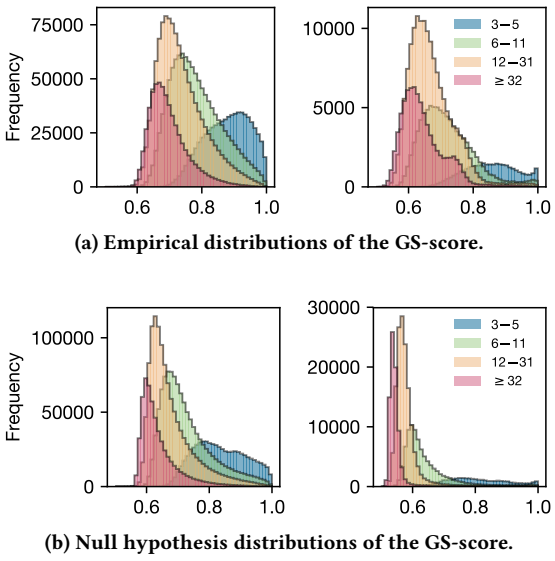
3.1 User Activity Diversity

Distribution of GS-scores. First, we calculate the GS-scores of all users in our Reddit and GitHub datasets. Since we're interested in the activity diversity of those who are users of the platform itself, as opposed to "drive-by" users who may only be aware of individual communities and not the entire platform, we restrict to users who have contributed to at least three communities in their lifetime[2]. The distributions of users' GS-scores broken down by number of communities are shown in Figure 5 (top).

First and foremost, we observe a broad diversity of user types, ranging from extreme generalists to extreme specialists, on both Reddit and GitHub. For example, a Reddit user with GS = 0.96 contributed to communities `wedding`, `weddingplanning`, `JustEngaged`, whereas another user with GS = 0.63 contributed to communities `zelda`, `australia`, `bisexual`, `horror`, `AskMen`, `homeland`. We also see that the GS-score is related to the number of communities one participates in, as desired. Generally speaking, the more communities one contributes to, the more generalist one is. However, this is not always the case, also as desired. In fact, we observe a broad diversity of GS-scores even when we fix the number of communities. All of the distributions in Reddit, for example, have support that is almost as broad support as all of them together — even if you contribute to 30 communities, say, you can still be an extreme generalist or an extreme specialist, depending on how similar the communities you contribute to are. We see many examples where users with 32+ communities are more specialist than users with 5 communities.

To check that these results are meaningful, we compare against a null hypothesis. We conduct the following permutation test: for each user, we keep their number of communities the same, but draw the communities independently from the overall distribution of communities weighted by popularity. Are the empirical results we observe simply produced by the distributions of community popularity and number of communities by user?

In fact, they are not. In Figure 5 (bottom), we plot the distributions of GS-scores produced by this null hypothesis. Comparing the empirical distributions with the null hypothesis distributions, we observe several major differences. We see that there are significantly more specialists at every number of communities, especially on GitHub, in the empirical data than in the permuted data. We also observe much more overlap between the GS-scores of users with different numbers of communities in the real data. Finally, we see that the most generalist constructed users are only marginally more generalist than the most generalist real users, indicating that real



(a) Empirical distributions of the GS-score.

(b) Null hypothesis distributions of the GS-score.

Figure 5: Empirical (top) and null hypothesis (bottom) distributions of GS-scores over users, broken down by number of communities, in Reddit (left) and GitHub (right).

users are approaching the mathematical limit of how generalist they can be given the number of communities they are in.

Relationship with entropy. In previous work, the most popular method of measuring activity diversity is using the entropy of a user's activity distribution. We examine the relationship between the GS-score and entropy. As desired, they are related: the more specialist users have lower entropy. But there is still significant overlap in the entropy distributions of different GS-scores; the GS-score can distinguish between users with similar activity distributions but over sets of communities with different levels of coherence. The R^2 between GS-score and entropy is 0.69 in both Reddit and GitHub, meaning that a significant amount of the variance in GS-score (31%) is unexplained by entropy.

Feedback success. On Reddit, every comment can be upvoted or downvoted by other users, indicating the audience's reaction to it. We use this as a proxy for success and ask how success relates with activity diversity. The raw distribution of success as a function of user GS-score is very flat, indicating that the range of viable strategies on Reddit is broad. However, there are many potential confounding factors influencing a comment's vote score: the post it is in reply to, how long after the original post it was written, its parent comment's score, the audience that sees it, etc. Thus, we consider the following question: does a comment get a higher score than its parent? (As comments on Reddit are threaded, they have some parent they are replying to, except for top-level comments). This elegantly controls for several factors. Parent and child comments usually occur close together in time, are in direct conversation with each other and so are more likely to share topics, and if the child outscores the parent (the parent usually scores higher due to its increased exposure) it can be considered a success. We examine the probability that a comment outscores its parent as a

[2]One could use the GS-score to study the activity diversity of those who only contributed to one or two communities, but it isn't the focus of our work here.

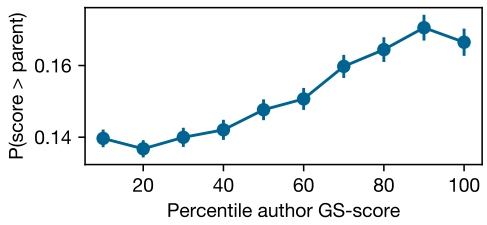


Figure 6: Comment success versus author GS-score.

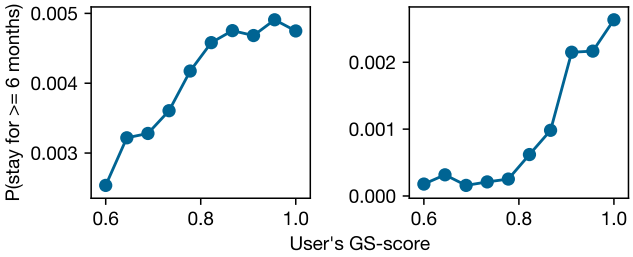


Figure 7: Likelihood of being engaged in a community for six consecutive months after first contributing, as a function of user GS-score on Reddit (left) and GitHub (right).

function of the author's GS-score in Figure 6. There is a significant, moderate effect, with specialists being 20% more likely than generalists to author a comment that outscores its parent. This is consistent with specialists being more engaged with, and more successful in, their home communities.

Longevity in communities. How long do users stay engaged in a community that they've joined? Is there a difference between how likely specialists and generalists are to stay? To answer this question, we examine all the occurrences of when Reddit and GitHub users join a new community, and calculate the likelihood that they subsequently make contributions in that community for six months in a row. Of course, this probability will be low on an absolute scale, but we are interested in whether generalist and specialists act differently. We plot this relationship in Figure 7. There is a clear, monotonically increasing relationship between the user's GS-score and the likelihood that they remain engaged for at least six months. The difference is quite large; on Reddit, specialists are about twice as likely to stay as generalists, and on GitHub the difference is even larger, with specialists about 10 times more likely than generalists to remain engaged. This reveals one way in which generalists and specialists are systematically different: generalists explore, specialists exploit. Generalists are more likely to try a new community but then leave, whereas specialists are more likely to stay in communities that they join. This has implications for community managers: given that a specialists have a much higher chance of converting into long-time members, it may be worth expending more effort or resources in recruiting specialists.

Longevity on the platform. We have just seen that specialists are more likely to stay in communities they join. What about the entire platform itself? Since activity is such a major factor in whether people will stay (as it is a sign that they are engaged), we control for activity level. Concretely, we consider users' activity in the first

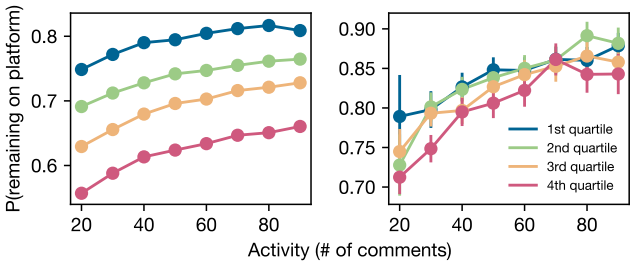


Figure 8: Likelihood of remaining on the platform versus user GS-score on Reddit (left) and GitHub (right).

half of 2015, and measure their GS-scores on this activity only. We then measure the probability that they are active two years later, in the second half of 2017, as a function of GS-score and activity.

These results are shown in Figure 8, in which there are a number of notable findings. On Reddit, there is a very large difference between generalists and specialists in the probability of leaving the platform. For any given level of activity, specialists (e.g. 4th quartile of GS-score, shown in red) are much less likely to remain on the platform than generalists (e.g. 1st quartile of GS-score, shown in blue). For example, among users with 20 contributions during the first half of 2015, generalists remain on Reddit 75% of the time while specialists remain only 55% of the time. Even more strikingly, small differences in the GS-score can trump large differences in activity. For example, users with 4th-quartile GS-scores (specialists) and 90 comments only remain 66% of the time, whereas users with 2nd-quartile GS-scores and only 20 comments remain 69% of the time. Finally, we note that the probability of remaining is monotonically increasing in activity for every quartile of GS-score. On GitHub, the differences between generalists and specialists are considerably less stark. However, specialists are still significantly less likely than generalists to remain on the platform, given activity level.

At a high level, these two groups of results suggest there is an interesting interplay between activity diversity and longevity. While specialists are more likely to stay in a given community once they've tried it, generalists are more likely to stay on the entire platform. These results have important implications for designers of online platforms. Community managers may prefer exposure to specialists to maximize the chance of acquiring new long-term members. But all else equal, generalists are more likely to be long-standing members of the platform itself. Thus, it may be in the platform's interest to expose users to a more diverse set of communities. Also, designing online platforms to attract generalists and be conducive to community exploration could be beneficial.

Diversity of exposure. Every user in a platform engages with some subset of the platform's population. Depending on the communities one contributes to, the subpopulation one is exposed to can vary dramatically, and can be quite different from the platform population as a whole. How does the diversity of the group one is exposed to vary with activity diversity? We conduct an analysis on Reddit, where for each user i we consider the set of users P_i that they replied to (comments on Reddit are threaded, and thus are often in reply to each other). We call this population of users one interacts with the "parent-universe". A key benefit of our definition of activity diversity is that it can be applied to more than a user.

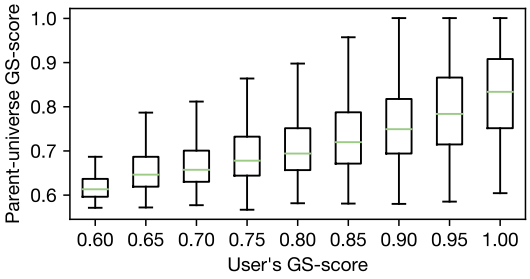


Figure 9: Distribution of the parent-universe GS-scores as a function of user GS-score.

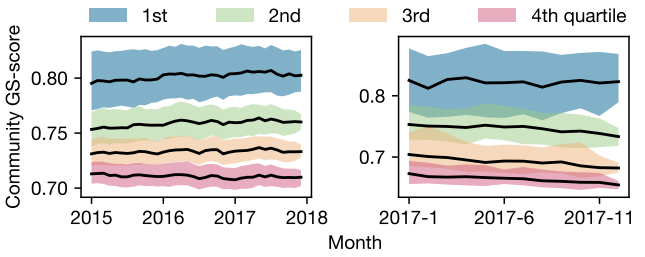


Figure 10: Stability of community GS-scores over time on Reddit (left) and GitHub (right).

Recall that the definition is an average over contributions, and so far we have been considering the set of contributions made by a user. Here we consider all contributions by users in the set P_i, and measure the activity diversity of this whole set. This is the diversity of the subpopulation user i is exposed to.

In Figure 9, we show how distributions of $GS(P_i)$ vary as a function of $GS(u_i)$ — how the diversity of the subpopulation one interacts with changes a function of one's own activity diversity. We observe a clear, strong trend: generalists are exposed to a diverse set of users, whereas specialists are exposed to more homogeneous sets of users. Users with first-decile GS-scores are exposed to a subpopulation of users with combined $GS(P_i)$ around 0.61, whereas users with tenth-decile GS-scores are exposed to a subpopulation of users with combined $GS(P_i)$ around 0.84, a huge difference. Specialists, by selecting small, coherent sets of communities to contribute to, limit themselves to relatively narrow "echo chambers", where the users they interact with are themselves very specialist in their collective interests. While this trend is intuitive, the magnitude of the effect is striking.

In this paper, we are focusing on the entire set of a user's contributions, but there are many interesting analyses to be done by concentrating on a topical subset. For example, one could straightforwardly apply the GS-score to only user contributions to overtly political subreddits. Repeating this analysis would then measure the extent to which generalists and specialists are living in political echo chambers online. We leave this and other analyses as examples of promising future work using our methodology and score.

3.2 Community Activity Diversity

We measure a user's activity diversity by considering how similar their communities are. Once we understand the activity diversity of every user in a community, we can use this to also understand the activity diversity of a *community*. Here, we define the GS-score $GS(c_i)$ of a community c_i as the weighted average over its users: $GS(c_i) = \frac{1}{N} \sum_j w_j \cdot GS(u_j)$, where u_j makes w_j contributions to c_i and there are $N = \sum_j w_j$ contributions in total. Specialist (generalist) communities are those with specialist (generalist) users on average. What kinds of communities tend to be generalist or specialist? Examining the distributions over community GS-scores, we observe a broad range of community activity diversity on both Reddit and GitHub, with most communities skewing generalist.

Community GS-score stability. Does the community GS-score capture a real property of a community? One way to test this is to measure if it is at least stable. In Figure 10, we plot how community GS-scores change over time. We separate the communities into quartiles based on their initial GS-score, then plot how the distribution of their GS-scores evolves over a 3-year period on Reddit and a 1-year period on GitHub (GitHub communities are more volatile and don't last as long on average). In both Reddit and GitHub, they are strikingly consistent. The quartiles are essentially non-overlapping in their community GS-scores for the entire period. We see strong evidence that generalist communities remain generalist and specialist communities remain specialist. Furthermore, we see evidence that communities have stable GS-scores even when the underlying user population changes dramatically. For example, the subreddit `r/gardening` has highly cyclical usage patterns over the year, being much more popular during the summer. Despite the volume of active users changing by an order of magnitude throughout the year, the community GS-score is very consistent.

Size vs. community GS-score. Finally, we observe how community size (number of users) and community GS-score relate. In Figure 11, we show a scatterplot of this relationship on Reddit (GitHub is similar). First, we observe a broad range of community GS-scores for almost every level of community size. Even communities with tens of thousands of users can have vastly different GS-scores. For example, on Reddit, `r/The_Donald` and `r/books` have exactly the same number of users during 2017 (241K), but `r/books` has a GS-score of 0.72 and `r/The_Donald` has a GS-score of 0.86, which are on opposite ends of the community activity diversity spectrum. Inspecting the communities, the GS-scores are intuitive: generalist communities are typically about very broad topics like `r/technology` or have wide appeal like `r/funny`, whereas specialist communities are typically centered around particular interests or niches like `r/FIFA` and `r/exmormon`. On GitHub, generalist communities are general-use frameworks, such as `d3` (a web-based data visualization framework), whereas specialist communities are more deeply technical, such as `grpc` (a remote procedure call system).

Elite users vs. community GS-score. Finally, we combine the user GS-score and community GS-score. Who are the top users in generalist and specialist communities? In each community, we rank users by the average comment vote score, and consider the top 5% as the "elites". Are the generalists or specialists the elites, and how does this vary with community GS-score? The relationship, shown in Figure 12, indicates that generalists are consistently more likely to be among the elites. The line $y = x$ indicates what would happen under a random baseline, and clearly the distributions are always below this line. We speculate that this is due to generalists having

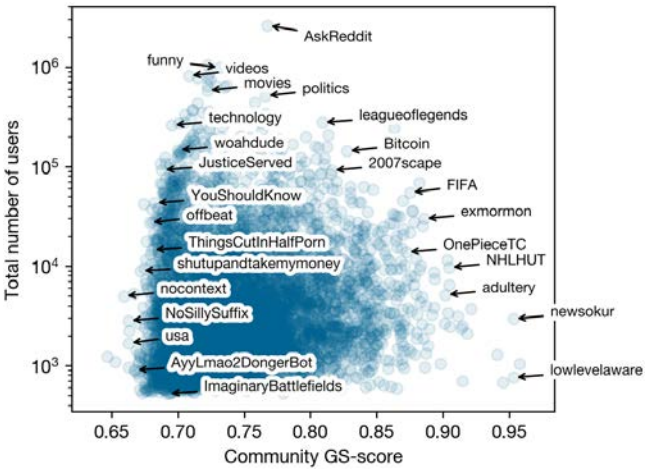


Figure 11: Community size versus community GS-score.

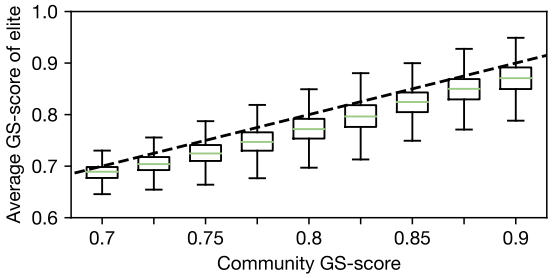


Figure 12: Elite users' GS-scores versus community GS-score.

more chances at being in the elite of a community. By our measure, a user can be in a community's elite with a small number of highly-rated comments. Since generalists are more likely to spread their activity out over more communities, they have more opportunities to be in the elite.

4 PREDICTION TASKS

In the previous Sections, we introduced our methodology for quantifying activity diversity and applied it to study user behavior in online platforms. Now we demonstrate that our insights are directly applicable to two important prediction problems in online platforms: community recommendation and user retention. We perform these tasks on our main dataset, Reddit.

4.1 Community Recommendation

Predicting which communities a user is likely to join next is a ubiquitous and important prediction task for online platforms. Our first task is to predict which new communities users will join after an initial observation period. On this task, every model outputs a ranking of communities for each user sorted by the likelihood of that user joining. We trained the models on all data from 2017 and tested them on the communities users joined for the first time during January 2018. We evaluated four models:

- **Random**: a baseline that randomly ranks communities.
- **Popularity**: communities ranked by popularity

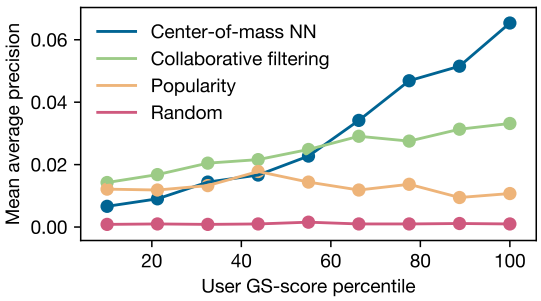


Figure 13: Performance of community recommendation models on Reddit versus user GS-score percentile.

- **Collaborative Filtering**: a state-of-the-art model (Bayesian Personalized Ranking [20]) for doing collaborative filtering with implicit data.
- **Center-of-Mass Nearest Neighbors**: a novel method using community embedding, which simply ranks communities by how cosine-similar they are to a user's center of mass during the training period (2017). Note that a community embedding trained on only 2017 data was used to ensure that the CF model and our embedding-based model had access to the exact same training data.

To assess model performance, we looked at users who were active in the training period and the test period. On Reddit, there were 20K such users with 24K total community joins. We use mean average precision (MAP) to assess the quality of community rankings produced by each model. The overall results are: 0.001 for random, 0.013 for popularity, 0.023 for CF, and 0.027 for Center-of-Mass NN. Despite being an extremely simple model, Center-of-Mass NN slightly outperforms a state-of-the-art collaborative filtering method, which demonstrates the fidelity of our community embeddings. Notice that our method doesn't even take popularity into account (which CF does).

Beyond this proof of concept, we are interested in how predictability varies with user GS-score. This relationship is shown in Figure 13, and there are two notable findings. First, it is clear that Center-of-Mass NN is competitive with CF among all users, but especially outperforms it on specialists. Second, it is remarkable that both CF and Center-of-Mass NN perform much better for specialist users. The fact that some users are so much more predictable than others, and that this correlates so well with the GS-score, is interesting and has important implications. For example, the value of recommendations could be much higher for specialists, which should inform their cost.

4.2 Platform User Retention

Predicting whether users will remain or leave the platform is another ubiquitous and important problem. Here, we demonstrate that the GS-score is valuable in solving this problem. Concretely, the task is to consider all users in an initial period and predict whether or not they will be present on the platform in a later period. We used the first half of 2015 as the initial period and the second half of 2017 as the later period, and randomly sample a balanced set

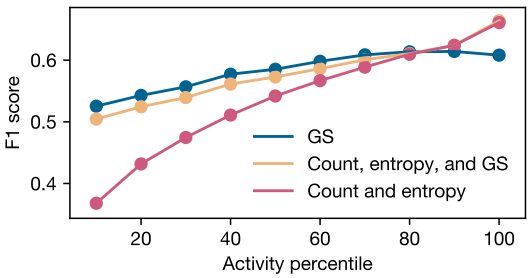


Figure 14: Performance of the Reddit longevity prediction models on users of varying activity levels.

of 464K users to consider (so that 50% stay and 50% leave). We split this into a training set and test set of equal size, and measure our models with the F1 score. We train a simple logistic regression model with comment count, entropy, and user GS-score as features. We compare against entropy since it is the main metric others have used in past work to quantify activity diversity.

We see that the GS-score alone outperforms the other models. Over most of the range of activity, but particularly for low-activity users, the improvement over entropy is quite large: for users in the lowest decile of activity, the F1 score of activity and entropy is 0.36 but the GS-score alone achieves an F1 score of 0.53 (see Figure 14). Count and entropy outperform the GS-score for the highest decile since activity is such a strong predictor of engagement.

5 RELATED WORK

Our work draws on three areas of previous research: embedding models, analyses of user behavior in social platforms, and work investigating generalists and specialists more broadly.

Embedding models. Our work draws upon the embedding methods that have precipitated revolutions in many fields, starting with natural language processing. Word embedding models like word2vec showed how to use modern model architectures to leverage large datasets and create embeddings that preserve semantic relationships [16, 17]. Subsequent work by Levy and Goldberg improved our understanding of these models, and also provided software that we used to generate our community embeddings [13, 14]. Kumar et al. applied word embedding algorithms to develop community embeddings in their study of community conflict on Reddit [11], and early steps in this direction were taken by Martin [15]. Embedding methods have also been developed for use in the context of social networks [9, 19, 24].

User behavior in social platforms. Our work is part of a large body of work investigating user behavior in social platforms. Reddit in particular has been the subject of a rich line of research. Tan and Lee study the trajectory of communities that Reddit users post in and find that users settle after an initial period of exploration [23]. They (and others) use entropy as a measure of topical concentration; as mentioned in Section 2, entropy is agnostic to the similarity or dissimilarity of different activities. Our measure and analysis builds on this work by taking community similarity into account. Hamilton et al. study community loyalty, both from the user and community perspectives [10], and Zhang et al. study community identity on Reddit, finding differences between communities with high and low churn rates [26].

Generalists and specialists. The notion of a spectrum spanning from generalists to specialists is ancient. It begins with the epigraph that opens this paper—"The fox knows many things; but the hedgehog knows one big thing."—written by 7th-century BC Greek poet Archilocus. This quote was the inspiration for the classic essay *The Hedgehog and the Fox* by philosopher Isaiah Berlin, in which he classifies writers and thinkers as either hedgehogs, those who have a single unifying idea, or foxes, those who draw on a wide array of experiences [4]. His work, in turn, brought the idea of generalists and specialists back into mainstream thought, and it has served as a useful frame ever since. As one example, Tetlock, in his studies of expert judgment and forecasters, analyzed the predictive abilities of generalists and specialists ("hedgehogs" and "foxes"), finding that specialists performed worse, especially on long-term forecasts within their specialty [25]. The problem of how broadly to allocate one's energy is ubiquitous: in biology, species are broadly categorized into generalist and specialist species [8]. Generalist species can thrive in a wide variety of environments, whereas specialist species are limited to a narrow range of environments.

In the study of activity diversity in online platforms, the closest work is Adamic et al.'s research on the relationship between individual focus and contribution on Wikipedia [1]. Our work builds on theirs by introducing a higher-fidelity metric for activity similarity based on community embeddings, studying other online platforms, assigning activity diversity scores to communities, and investigating the relationship between activity diversity and additional outcome metrics. Anderson studied how skill diversity correlates with worker pay in an online crowdwork platform [2].

6 CONCLUSION

In this work, we have introduced a methodology to quantify activity diversity in online platforms, and analyzed several important phenomena around user behavior through this lens on Reddit and GitHub. We developed high-quality *community embeddings*, which encode similarities between communities as defined by their members, as well as a set of community analogies to validate them. Based on these embeddings, we introduced the *GS-score*, a principled measure of activity diversity. We found a broad spectrum of user styles, from extreme generalists to extreme specialists, and observed that specialists are more likely to produce higher-quality replies. We also observed that specialists are much more likely to stay in communities they contribute to, but generalists are much more likely to remain on the platform as a whole. We applied activity diversity to study an important question: how diverse are the subpopulations users engage with? We found that generalists engage with a significantly more diverse group of people, whereas specialists are exposed to much narrower segments of the population. We also studied the activity diversity of communities, and found that the community GS-score is remarkably stable over time, even when the underlying user base changes a lot. In our prediction tasks, we found that specialists are more predictable than generalists in which communities they will join, and that the user GS-score is much more predictive of user retention than entropy.

There are several promising directions for future work. First, the GS-score can be directly applied to answer important questions about *subsets* of user activity in online platforms. For example, it would be interesting to conduct our results on echo chambers in the subset of political activity on Reddit. The GS-score then becomes a measure of political diversity. Second, as our GS-score metric is robust and conceptually simple, it could be broadly applicable beyond the online platforms we considered in this work. It would be illuminating to apply the GS-score to other domains as well.

Acknowledgments. This work was partially supported by NSERC. We thank Josef Waller, Shunzhe Yu, and especially Will Hamilton for fruitful discussions, and we thank Jason Baumgartner and the GH Archive for providing public access to the Reddit and GitHub datasets, respectively.

REFERENCES

[1] Lada A Adamic, Xiao Wei, Jiang Yang, Sean Gerrish, Kevin K Nam, and Gavin S Clarkson. 2010. Individual focus and knowledge contribution. *arXiv preprint arXiv:1002.0561* (2010).
[2] Katharine A Anderson. 2017. Skill networks and measures of complex human capital. *Proceedings of the National Academy of Sciences (PNAS)* (2017).
[3] Jason Baumgartner. 2017. pushshift.io Reddit archive. https://pushshift.io/. (2017). Accessed: 2018-07-23.
[4] Isaiah Berlin. 1953. *The hedgehog and the fox.* Weidenfeld & Nicolson.
[5] Hendrik Bode, Frederick Mosteller, John W Tukey, and Charles Winsor. 1949. The education of a scientific generalist. *Science* (1949).
[6] Elisabeth Bublitz and Florian Noseleit. 2014. The skill balancing act: when does broad expertise pay off? *Small Business Economics* 42, 1 (2014).
[7] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).*
[8] Nicholas B Davies, John R Krebs, and Stuart A West. 2012. *An introduction to behavioural ecology.* John Wiley & Sons.
[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NIPS).*
[10] William L Hamilton, Justine Zhang, Cristian Danescu-Niculescu-Mizil, Dan Jurafsky, and Jure Leskovec. 2017. Loyalty in online communities. In *Proceedings of the International AAAI Conference on Weblogs and Social Media (ICWSM).*
[11] Srijan Kumar, William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2018. Community interaction and conflict on the web. In *Proceedings of the 2018 World Wide Web Conference (WWW).*
[12] Erin Leahey. 2007. Not by productivity alone: How visibility and specialization contribute to academic earnings. *American Sociological Review* 72, 4 (2007), 533–561.
[13] Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers).*
[14] Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems (NIPS).*
[15] Trevor Martin. 2017. community2vec: Vector representations of online communities encode semantic relationships. In *Proceedings of the Second Workshop on NLP and Computational Social Science.*
[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS).*
[18] Scott Page. 2007. *The Difference: How the Power of Diversity Creates Better Groups, Firms, Schools, and Societies.* Princeton University Press.
[19] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).*
[20] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI).*
[21] Kurt C Stange. 2009. The generalist approach. *The Annals of Family Medicine* 7, 3 (2009).
[22] Rosemary Stevens. 2017. *Medical practice in modern England: the impact of specialization and state medicine.* Routledge.
[23] Chenhao Tan and Lillian Lee. 2015. All who wander: On the prevalence and characteristics of multi-community engagement. In *Proceedings of the 24th International Conference on World Wide Web (WWW).*
[24] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web (WWW).*
[25] Philip E Tetlock. 2005. *Expert political judgment: How good is it? How can we know?* Princeton University Press.
[26] Justine Zhang, William L Hamilton, Cristian Danescu-Niculescu-Mizil, Dan Jurafsky, and Jure Leskovec. 2017. Community identity and user engagement in a multi-community landscape. In *Proceedings of the International AAAI Conference on Weblogs and Social Media (ICWSM).*

A Family of Fuzzy Orthogonal Projection Models for Monolingual and Cross-lingual Hypernymy Prediction

Chengyu Wang
School of Computer Science and Software Engineering, East China Normal University
chywang2013@gmail.com

Yan Fan
School of Computer Science and Software Engineering, East China Normal University
eileen940531@gmail.com

Xiaofeng He*
School of Computer Science and Software Engineering, East China Normal University
xfhe@sei.ecnu.edu.cn

Aoying Zhou
School of Data Science and Engineering, East China Normal University
ayzhou@dase.ecnu.edu.cn

ABSTRACT

Hypernymy is a semantic relation, expressing the "is-a" relation between a concept and its instances. Such relations are building blocks for large-scale taxonomies, ontologies and knowledge graphs. Recently, much progress has been made for hypernymy prediction in English using textual patterns and/or distributional representations. However, applying such techniques to other languages is challenging due to the high language dependency of these methods and the lack of large training datasets of lower-resourced languages.

In this work, we present a family of fuzzy orthogonal projection models for both monolingual and cross-lingual hypernymy prediction. For the monolingual task, we propose a Multi-Wahba Projection (MWP) model to distinguish hypernymy vs. non-hypernymy relations based on word embeddings. This model establishes distributional fuzzy mappings from embeddings of a term to those of its hypernyms and non-hypernyms, which consider the complicated linguistic regularities of these relations. For cross-lingual hypernymy prediction, a Transfer MWP (TMWP) model is proposed to transfer the semantic knowledge from the source language to target languages based on neural word translation. Additionally, an Iterative Transfer MWP (ITMWP) model is built upon TMWP, which augments the training sets of target languages when target languages are lower-resourced with limited training data. Experiments show i) MWP outperforms previous methods over two hypernymy prediction tasks for English; and ii) TMWP and ITMWP are effective to predict hypernymy over seven non-English languages.

CCS CONCEPTS

• **Computing methodologies** → **Lexical semantics**; *Ontology engineering*; • **Information systems** → *Data mining*.

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313439

KEYWORDS

hypernymy prediction, Multi-Wahba Projection, cross-lingual transfer learning

ACM Reference Format:
Chengyu Wang, Yan Fan, Xiaofeng He, and Aoying Zhou. 2019. A Family of Fuzzy Orthogonal Projection Models for Monolingual and Cross-lingual Hypernymy Prediction. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313439

1 INTRODUCTION

Hypernymy is a type of basic semantic relations, expressing the "is-a" relation between a concept (hypernym) and its instances (hyponyms). The automatic extraction of hypernymy relations is vital for building large-scale taxonomies, ontologies and knowledge graphs [10, 33, 51]. Such relations also benefit a variety of Web-scale applications including query understanding [48], post-search navigation [14], personalized recommendation [56], etc.

Due to its importance in NLP and real-world applications, remarkable progress has been made for hypernymy prediction. As summarized in Wang et al. [45], pattern based and distributional approaches are two major types of methods for hypernymy harvesting. Pattern based methods employ lexical patterns (e.g., Hearst patterns [13]) to extract hypernymy relations from text corpora [30]. Distributional methods can be divided as unsupervised and supervised. Unsupervised methods refer to hypernymy measures, modeling the degree of the existence of a hypernymy relation between two terms [15, 31]. Supervised algorithms classify a term pair as hypernymy or non-hypernymy based on the distributional representations of the two terms [20, 54]. While there exists some disagreement in the NLP community on which type of methods is more effective [45], Shwartz et al. [34] show that both methods can be combined via an integrated neural network, further improving the performance.

Despite the significant success, there is much room for improvement, due to the following reasons. i) The linguistic regularities of hypernymy relations are complicated to model [10, 44, 53]. For example, semantics of hypernymy relations between concept-concept pairs (subclass-of) and concept-instance pairs (instance-of) are different. Such pairs, however, are often mixed in real-world scenarios.

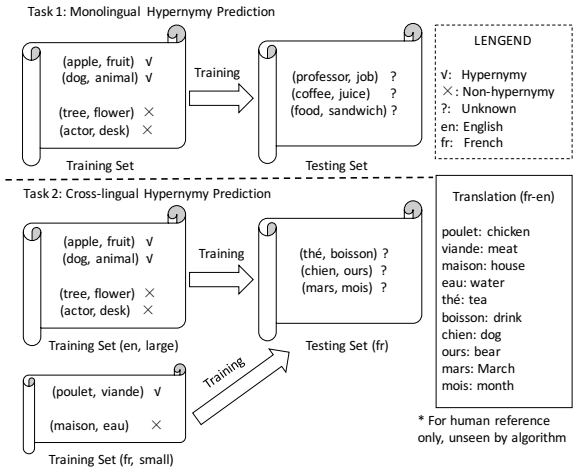


Figure 1: A toy example of monolingual and cross-lingual hypernymy prediction.

ii) Supervised distributional methods are reported to have high performance, but suffer from the "lexical memorization" problem [18]. iii) Most existing methods focus on monolingual hypernymy prediction in English only and are highly language dependent. Some analytical languages (e.g., Chinese, Vietnamese) are flexible in expressions and lack fixed patterns to extract hypernymy relations with high recall [10, 46]. Hence, most existing methods are difficult to apply to other languages. iv) Several distributional methods require additional knowledge (e.g., taxonomies and semantic hierarchies) to learning high-quality hypernymy embeddings [20, 26, 54], which are difficult to obtain for some low-resourced languages.

In this work, a family of fuzzy orthogonal projection models is presented to address the tasks of both monolingual and cross-lingual hypernymy prediction. A toy example of the two tasks is illustrated in Figure 1. For the monolingual task, we explicitly model how the embeddings of a term are mapped to those of its hypernyms or non-hypernyms, respectively, in order to avoid the "lexical memorization" problem [18]. To deal with complicated linguistic regularities, for each latent component of hypernymy and non-hypernymy relations, a fuzzy orthogonal mapping is established in the embedding space[1]. The learning process of the mappings is closely related to a generalized version of the Wahba's problem in mathematics [23], named Multi-Wahba Projection (MWP). In this work, we derive a closed-form solution to MWP and propose an MWP based neural network to distinguish hypernymy vs. non-hypernymy relations.

For cross-lingual hypernymy prediction, the goal is to predict hypernymy relations of a target language with limited training data, given a (relatively) large training set of a source language as input. Neural machine translation techniques are able to transfer knowledge across languages, but usually require a large parallel corpus, which is not feasible for some lower-resourced languages [55]. To derive a general solution for any languages, we present a Transfer MWP (TMWP) model from a source language to target languages based on neural word translation. To handle the small size of training sets of target languages, an Iterative Transfer MWP (ITMWP) model is built upon TMWP to support semi-supervised learning via training data augmentation.

In summary, we make the following major contributions:

- We propose a Multi-Wahba Projection (MWP) model to distinguish monolingual hypernymy vs. non-hypernymy relations. A closed-form solution is derived for generalized Wahba's problem.
- We propose a Transfer Multi-Wahba Projection (TMWP) model for cross-lingual hypernymy prediction. An iterative data augmentation method is introduced, named the Iterative Transfer Multi-Wahba Projection (ITMWP) model.
- We conduct extensive experiments to show that i) MWP outperforms state-of-the-art over two hypernymy prediction tasks for English, and ii) TMWP and ITMWP are effective to predict hypernymy over seven non-English languages.

The rest of this paper is organized as follows. Section 2 summarizes related work. MWP, TMWP and ITMWP models are presented in Section 3 and Section 4. Experiments are shown in Section 5 and Section 6, with the conclusion drawn in Section 7.

2 RELATED WORK

This section presents a brief overview on monolingual and cross-lingual hypernymy prediction.

2.1 Monolingual Hypernymy Prediction

In the NLP community, "hypernymy prediction" is not a standalone task, but has been evaluated under a series of tasks, e.g., unsupervised hypernymy classification [15], supervised hypernymy detection [20, 54], graded lexical entailment [41], etc. Although evaluation steps and metrics may differ, pattern based and distributional methods are two major paradigms to address these tasks [45].

Pattern based approaches use lexical patterns to extract hypernymy relations from texts. Hearst patterns [13] are the most influential patterns in English, which are employed to build a large lexical taxonomy Probase [51]. A recent study [30] reveals that using simple Hearst patterns can result in high performance for hypernymy detection. Similar Hearst pattern-like features are used in a variety of methods [28, 35]. The semantics of more generalized textual patterns can be captured by LSTM-based neural networks [34]. The reason of the success is that they are precise and have high coverage of hypernymy relations in English. For other languages, Hearst-style patterns are not necessarily effective. For example, Chinese has flexible expressions and lacks fixed patterns for hypernymy detection [10, 46]. Because we aim at providing a general solution for all languages, pattern-based approaches are difficult to apply to other languages, without language-specific modifications.

Distributional methods can be divided as supervised and unsupervised. **Unsupervised distributional approaches** are primarily based on hypernymy measures, modeling the degree of the existence of a hypernymy relation within a term pair. Typical hypernymy measures are based on distributional inclusion hypothesis [17], distributional informativeness hypothesis [31] and selective distributional inclusion hypothesis [29]. Readers can refer

[1]For hypernymy, relations related to different domains are usually treated as different components. The "subclass-of" and "instance-of" relations are treated as different components. Components of non-hypernymy relations include co-hyponymy, synonymy, meronymy, etc. Refer to [10, 49, 53] for detailed discussion.

to a comprehensive overview and evaluation on hypernymy measures in [32]. In **supervised methods**, each pair is modeled as an embedding vector as the input of a classifier to predict the relation, such as the Concat model, the Diff model, etc [29, 49]. Recently, several algorithms are proposed to learn hypernymy embeddings, which consider the taxonomic structure of concepts [6, 20, 26, 54]. For example, Yu et al. [54] learn the hypernym and hyponym embeddings for a term in a max-margin neural network based on Probase [51]. Nguyen et al. [26] propose the hierarchical embeddings trained over a text corpus and the WordNet concept hierarchy. Additionally, Drozd et al. [9] study relationships between semantic relations and word embeddings.

Projection based approaches are variants of previous methods, which learn how to map the embeddings of a term to those of its hypernyms directly. A notable model is the piecewise linear projection model [10]. It is also improved by Yamane et al. [53], which learns the number of linear projection models and the parameters of projection models jointly, and by Biemann et al. [3], Wang and He [44], which consider explicit negative samples. Wang et al. [46] learn the representations of both hypernymy and non-hypernymy in a transductive learning framework. Our work improves these approaches by multiple fuzzy orthogonal projections, and also employs a neural network for hypernymy relation classification. Hence, it takes advantages of both classification and projection approaches. Apart from the previous methods, many approaches aim at extracting hypernymy relations from various Web data sources. Gupta et al. [12], Ponzetto and Strube [27], Suchanek et al. [36] learn hypernymy from the Wikipedia category system based on lexical patterns, syntactic constraints and rule-based inference. Liu et al. [19] construct a taxonomy purely from keywords. We do not further elaborate.

2.2 Cross-lingual Hypernymy Prediction

The task of **cross-lingual hypernymy prediction** has not been sufficiently studied. Some researchers aim at building taxonomies based on multi-lingual Wikipedia [21, 47]. YAGO 3 [21] is the extension of YAGO [36], which integrates a multi-lingual taxonomy derived from Wikipedia. Wang et al. [47] propose a cross-lingual knowledge validation technique to improve the accuracy of cross-lingual hypernymy prediction from links among multi-lingual versions of Wikipedia. Wu et al. [50] introduce a bilingual topic model to align taxonomies of different languages.

In the NLP community, a few works [38, 42] determine if a word in one language is a hypernym of a word in another language. For example, these methods predict if the English word "fruit" is the hypernym of the French word "pomme" (apple). The difference between these methods and ours is that our method is capable of learning hypernymy relations where both hypernyms and hyponyms are in the target language. Hence, our method can extract culture-specific hypernymy relations where both hypernyms and hyponyms are not present in the source language.

3 MWP: THE MONOLINGUAL MODEL

In this section, we introduce the MWP model for monolinugal hypernymy prediction and give the closed-form solution of the generalized Wahba's problem.

3.1 Task Definition

Let $\vec{x}_i$ be the normalized embedding of the term x_i, pre-trained using any neural language models. Denote $y_i^{(+)}$ and $y_i^{(-)}$ as a hypernym and a non-hypernym of x_i, respectively. The task of monolingual hypernymy prediction is defined as follows:

Definition 3.1. (Monolingual Hypernymy Prediction) The goal is to train a classifier f over a hypernymy relation set $D^{(+)} = \{(x_i, y_i^{(+)})\}$ and a non-hypernymy relation set $D^{(-)} = \{(x_i, y_i^{(-)})\}$, to predict hypernymy relations $U = \{(x_i, y_i)\}$ of the same language.

3.2 Learning Hypernymy Projections

Following [10, 46, 53], hypernymy projection models assume that there exists a $d \times d$ projection matrix $\mathbf{M}^{(+)}$ such that $\mathbf{M}^{(+)}\vec{x}_i \approx \vec{y}_i^{(+)}$ where d is the dimension of word embeddings and $(x_i, y_i^{(+)}) \in D^{(+)}$. In this work, MWP further forces that the cosine similarity of the projected word embedding $\mathbf{M}\vec{x}_i$ and the true hypernym embedding $\vec{y}_i^{(+)}$ should be maximized, i.e., $\max \sum_{i=1}^{|D^{(+)}|} \cos(\mathbf{M}\vec{x}_i, \vec{y}_i^{(+)})$. Because all the word embeddings are normalized, the optimization function can be re-written as: $\max \sum_{i=1}^{|D^{(+)}|} (\mathbf{M}\vec{x}_i)^T \vec{y}_i^{(+)}$. We can see that $\mathbf{M}$ should be orthogonal to guarantee that $\mathbf{M}\vec{x}_i$ is normalized [52]. For ease of the model training process, we re-formulate the objective function as below with $\mathbf{I}$ being the $d \times d$ identity matrix:

$$\min \sum_{i=1}^{|D^{(+)}|} \|\mathbf{M}\vec{x}_i - \vec{y}_i^{(+)}\|^2 \text{ s. t. } \mathbf{M}^T\mathbf{M} = \mathbf{I}$$

However, this setting does not consider the complicated linguistic regularities of hypernymy relations. As seen in [10, 44], different types of hypernymy relations may correspond to different projection matrices. Denote K as the number of clusters, where each cluster corresponds to a hypernymy component. We apply K-means to $D^{(+)}$ using the vector offset $\vec{x}_i - \vec{y}_i^{(+)}$ as features. The cluster centroids are denoted as $\vec{c}_1^{(+)}, \cdots, \vec{c}_K^{(+)}$. We define $a_{i,j}^{(+)}$ as the weight of $(x_i, y_i^{(+)})$ over the dataset $D^{(+)}$ w.r.t. the jth cluster:[2]

$$a_{i,j}^{(+)} = \frac{\cos(\vec{x}_i - \vec{y}_i^{(+)}, \vec{c}_j^{(+)})}{\sum_{i'=1}^{|D^{(+)}|} \cos(\vec{x}_{i'} - \vec{y}_{i'}^{(+)}, \vec{c}_j^{(+)})}$$

Considering the orthogonal constraint, we minimize the weighted projection errors on the jth cluster with $\mathbf{M}_j^{(+)}$ being a $d \times d$ projection matrix w.r.t. the jth cluster:

$$\begin{aligned} \min J(\mathbf{M}_j^{(+)}) =& \frac{1}{2} \sum_{i=1}^{|D^{(+)}|} a_{i,j}^{(+)} \|\mathbf{M}_j^{(+)}\vec{x}_i - \vec{y}_i^{(+)}\|^2 \\ \text{s. t. } & \mathbf{M}_j^{(+)T}\mathbf{M}_j^{(+)} = \mathbf{I}, \sum_{i=1}^{|D^{(+)}|} a_{i,j}^{(+)} = 1 \end{aligned} \tag{1}$$

[2] We have also experimented with several fuzzy clustering algorithms, such as Gaussian Mixture Model, Fuzzy c-Means, etc. The cluster membership probabilities are related to the weights $a_{i,j}^{(+)}$. However, they perform poorly due to the high dimensionality of word embeddings. Hence, we employ K-means to generate clusters and use a heuristic approach to compute $a_{i,j}^{(+)}$.

Let $\mathcal{M}^{(+)} = \{\mathbf{M}_1^{(+)}, \mathbf{M}_2^{(+)}, \cdots, \mathbf{M}_K^{(+)}\}$ be the set of projection parameters of all the K clusters. Learning hypernymy projections is equivalent to minimizing the following objective function:

$$\begin{aligned} \min \tilde{J}(\mathcal{M}^{(+)}) =& \frac{1}{2}\sum_{j=1}^{K}\sum_{i=1}^{|D^{(+)}|} a_{i,j}^{(+)} \|\mathbf{M}_j^{(+)}\vec{x}_i - \vec{y}_i^{(+)}\|^2 \\ & \text{s. t. } {\mathbf{M}_j^{(+)}}^T\mathbf{M}_j^{(+)} = \mathbf{I}, \sum_{i=1}^{|D^{(+)}|} a_{i,j}^{(+)} = 1, j = 1, \ldots, K \end{aligned} \tag{2}$$

We refer to this model as the Multi-Wahba Projection (MWP) model. In mathematics, the Wahba's problem deals with three-dimensional vector observations between two coordinate systems, extensively applied in satellite attitude determination [23]. The MWP model is an extension to the Wahba's problem, considering d dimensional ($d > 3$) projections with K components. Because the values of different $\mathbf{M}_j^{(+)}$ ($j = 1, 2, \cdots, K$) are independent, the optimal solution to MWP is the same as the combination of minimizing K objectives $J(\mathbf{M}_j^{(+)})$. In this work, we pay attention to a closed-form solution to the Wahba's problem [22] based on Singular Value Decomposition (SVD) and extend it to d dimensions ($d > 3$):

Theorem 3.2. *The d-dimensional Wahba's problem has a closed-form solution as follows:*

(1) $\mathbf{B}_j = \sum_{i=1}^{|D^{(+)}|} a_{i,j}^{(+)} \vec{y}_i^{(+)} \vec{x}_i^T$;
(2) $SVD(\mathbf{B}_j) = \mathbf{U}_j\mathbf{\Sigma}_j\mathbf{V}_j^T$;
(3) $\mathbf{R}_j = diag(\underbrace{1, \ldots, 1}_{d-1}, det(\mathbf{U}_j)det(\mathbf{V}_j))$;
(4) $\mathbf{M}_j^{(+)} = \mathbf{U}_j\mathbf{R}_j\mathbf{V}_j^T$;

Proof. For simplicity, we omit the subscript j and the superscript (+) of all variables. Eq. (1) can be re-written as:

$$J(\mathbf{M}) = \frac{1}{2}\sum_i a_i \|\mathbf{M}\vec{x}_i - \vec{y}_i\|^2 \text{ s. t. } \mathbf{M}^T\mathbf{M} = \mathbf{I}$$

Here, each pair (x_i, y_i) has the weight a_i over the entire dataset. Because $\sum_i a_i = 1$, based on the proof of the original Wahba's problem [43], we re-write the objective function as follows:

$$J(\mathbf{M}) = 1 - \sum_i a_i \vec{y}_i^T\mathbf{M}\vec{x}_i = 1 - \text{tr}(\mathbf{M}\mathbf{B}^T) \tag{3}$$

where $\mathbf{B} = \sum_i a_i\vec{y}_i\vec{x}_i^T$. The SVD of the matrix $\mathbf{B}$ is given by: $\text{SVD}(\mathbf{B}) = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ with the singular value matrix $\mathbf{\Sigma} = \text{diag}(\lambda_1, \ldots, \lambda_d)$.

Based on the result of SVD, we define two orthogonal matrices and one diagonal matrix:

$$\mathbf{U}_+ = \mathbf{U}\text{diag}(\underbrace{1, \ldots, 1}_{d-1}, \det(\mathbf{U})) \quad \mathbf{V}_+ = \mathbf{V}\text{diag}(\underbrace{1, \ldots, 1}_{d-1}, \det(\mathbf{V}))$$

$$\mathbf{\Sigma}' = \text{diag}(\lambda_1, \ldots, \lambda_{d-1}, \lambda_d\det(\mathbf{U})\det(\mathbf{V}))$$

Due to the orthogonality of $\mathbf{U}$ and $\mathbf{V}$, we have $\det(\mathbf{U})\det(\mathbf{V}) = \pm 1$. Hence, matrix $\mathbf{B}$ can be re-decomposed as: $\mathbf{B} = \mathbf{U}_+\mathbf{\Sigma}'\mathbf{V}_+^T$.

For simplicity, let $\mathbf{W} = \mathbf{U}_+^T\mathbf{M}\mathbf{V}_+$. Based on the cyclic invariance property of the trace, we have

$$\text{tr}(\mathbf{M}\mathbf{B}^T) = \text{tr}(\mathbf{M}\mathbf{V}_+\mathbf{\Sigma}'\mathbf{U}_+^T) = \text{tr}(\mathbf{\Sigma}'\mathbf{U}_+^T\mathbf{M}\mathbf{V}_+) = \text{tr}(\mathbf{\Sigma}'\mathbf{W})$$

Therefore, we re-write Eq. (3) as: $J(\mathbf{M}) = 1 - \text{tr}(\mathbf{\Sigma}'\mathbf{W})$.

Using the Euler axis/angle parameterization for the orthogonal matrix $\mathbf{W} = \mathcal{R}(\mathbf{e}, \phi)$, we have:

$$\begin{aligned} J(\mathbf{M}) = 1 - (\sum_{i=1}^{d-1}\lambda_i + \lambda_d\det(\mathbf{U})\det(\mathbf{V})) + (1 - \cos\phi)[\sum_{i=2}^{d-1}\lambda_i \\ + \lambda_d\det(\mathbf{U})\det(\mathbf{V}) + \sum_{i=2}^{d-1}(\lambda_1 - \lambda_i)e_i^2 + (\lambda_1 - \lambda_d\det(\mathbf{U})\det(\mathbf{V}))e_d^2] \end{aligned}$$

where $\mathbf{e} = [e_1, e_2, \cdots, e_d]$ is a unit vector and ϕ is a rotation angle. It is easy to show that $J(\mathbf{M})$ is minimized when $\cos\phi = 1$. We have:

$$\min J(\mathbf{M}) = 1 - (\sum_{i=1}^{d-1}\lambda_i + \lambda_d\det(\mathbf{U})\det(\mathbf{V})) = 1 - \text{tr}(\mathbf{\Sigma}')$$

In this condition, we have $\mathbf{W} = \mathbf{I}$. This gives the optimal solution:

$$\mathbf{M}_{opt} = \mathbf{U}_+\mathbf{V}_+^T = \mathbf{U}\text{diag}(\underbrace{1, \ldots, 1}_{d-1}, \det(\mathbf{U})\det(\mathbf{V}))\mathbf{V}^T$$

□

Complexity analysis. For simplicity, let $n = |D^{(+)}|$. The construction of the data matrix $\mathbf{B}_j$ takes $O(nd^2)$ time. Subsequent steps include the SVD, the determinant computation and the matrix multiplication of $d \times d$ matrices. The time complexity is $O(d^3)$. Hence, the total time complexity is $O(nd^2 + d^3)$. Because d is a small constant, usually around $50 \sim 300$ in practice, and n is the size of the training set, the algorithm is sufficiently efficient for the task.

3.3 Learning Non-hypernymy Projections

In the semantic relation classification task, non-hypernyms of a term can be co-hyponyms, synonyms, etc [49]. Hence, MWP is also suitable for modeling the semantics of different non-hypernymy relations. The objective is defined as follows:[3]

$$\begin{aligned} \min \tilde{J}(\mathcal{M}^{(-)}) =& \frac{1}{2}\sum_{j=1}^{K}\sum_{i=1}^{|D^{(-)}|} a_{i,j}^{(-)} \|\mathbf{M}_j^{(-)}\vec{x}_i - \vec{y}_i^{(-)}\|^2 \\ & \text{s. t. } {\mathbf{M}_j^{(-)}}^T\mathbf{M}_j^{(-)} = \mathbf{I}, \sum_{i=1}^{|D^{(-)}|} a_{i,j}^{(-)} = 1, j = 1, \ldots, K \end{aligned}$$

with $\vec{c}_1^{(-)}, \cdots, \vec{c}_K^{(-)}$ as non-hypernymy cluster centroids, and $\mathcal{M}^{(-)} = \{\mathbf{M}_1^{(-)}, \mathbf{M}_2^{(-)}, \cdots, \mathbf{M}_K^{(-)}\}$ as projection parameters. The centroids are generated by K-means over $D^{(-)}$, using $\vec{x}_i - \vec{y}_i^{(-)}$ as features.

3.4 Hypernymy Prediction Classifier

This part describes how MWP benefits hypernymy prediction by training a neural network classifier.

Due to the effectiveness of the Diff model for hypernymy classification [26], given a pair $(x_i, y_i) \in D^{(+)} \cup D^{(-)}$, we generate two groups of features:[4]

$$\mathcal{F}^{(+)}(\vec{x}_i, \vec{y}_i) = (\mathbf{M}_1^{(+)}\vec{x}_i - \vec{y}_i) \oplus \cdots \oplus (\mathbf{M}_K^{(+)}\vec{x}_i - \vec{y}_i)$$

$$\mathcal{F}^{(-)}(\vec{x}_i, \vec{y}_i) = (\mathbf{M}_1^{(-)}\vec{x}_i - \vec{y}_i) \oplus \cdots \oplus (\mathbf{M}_K^{(-)}\vec{x}_i - \vec{y}_i)$$

[3] Note that the numbers of clusters in hypernymy and non-hypernymy relations can also be different. For simplicity, we set the numbers of clusters in hypernymy and non-hypernymy relations uniformly as K.
[4] For simplicity, we omit the superscripts (+) and (−) for y_i here.

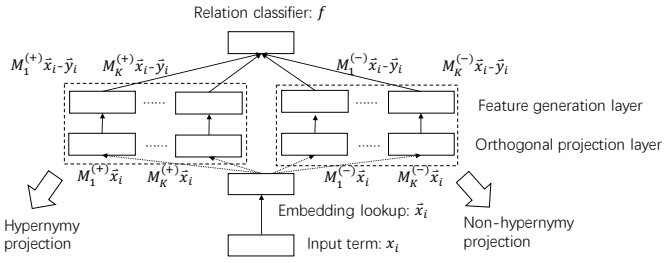


Figure 2: The MWP based neural network architecture for monolingual hypernymy prediction.

where $\oplus$ is the vector concatenation operator.

Figure 2 shows the MWP based neural network for hypernymy relation classification. For each pair $(x_i, y_i) \in D^{(+)} \cup D^{(-)}$, the model generates features $\mathcal{F}^{(+)}(\vec{x}_i, \vec{y}_i) \cup \mathcal{F}^{(-)}(\vec{x}_i, \vec{y}_i)$. A binary classifier f is trained over $D^{(+)}$ and $D^{(-)}$.[5]

The reason for designing such architecture is as follows. In distributional semantics, using embeddings $\vec{x}_i$ and $\vec{y}_i$ to predict the relation between x_i and y_i can cause the "lexical memorization" problem [18]. For example, if the classifier receives positive pairs "(dog, animal)", "(cat, animal)" and "(sheep, animal)" during training, it will "memorize" the property of the embeddings of "animal", where "animal" is regarded as a "prototypical hypernym" by [18]. In consequence, the classifier is likely to predict there is a hypernymy relation in "(rose, animal)" when seeing the term "animal".

In our approach, if there is a hypernymy relation in (x_i, y_i), the norms of the features $\mathcal{F}^{(+)}(x_i, y_i)$ are likely to be small based on Eq. (2). The norms of features $\mathcal{F}^{(-)}(x_i, y_i)$ are likely to be large. Hence the combined features $\mathcal{F}^{(+)}(x_i, y_i) \cup \mathcal{F}^{(-)}(x_i, y_i)$ are discriminative for distinguishing hypernymy vs. non-hypernymy relations. By modeling the semantics of hypernymy and non-hypernymy, this method avoids "lexical memorization". The high-level training procedure is summarized in Algorithm 1.

Algorithm 1 Monolingual Hypernymy Prediction

```
Perform K-means over D^(+), with features as x⃗_i − y⃗_i^(+);
Perform K-means over D^(−), with features as x⃗_i − y⃗_i^(−);
for j = 1 to cluster number K do
    Learn M_j^(+) and M_j^(−) by the closed-form solution;
end for
for each pair (x_i, y_i^(+)) ∈ D^(+) do
    Compute F^(+)(x⃗_i, y⃗_i^(+)) and F^(−)(x⃗_i, y⃗_i^(+));
end for
for each pair (x_i, y_i^(−)) ∈ D^(−) do
    Compute F^(+)(x⃗_i, y⃗_i^(−)) and F^(−)(x⃗_i, y⃗_i^(−));
end for
Train neural network classifier f over D^(+) and D^(−);
```

4 CROSS-LINGUAL MODELS

In this section, we introduce two projection models (i.e., TMWP and ITMWP) based on MWP for cross-lingual hypernymy prediction.

[5]We have added more hidden layers between the feature generation layer and the output layer with no improvement. Hence, an output layer is directly connected to the feature generation layer.

4.1 Transfer MWP

Let $D_S^{(+)}$, $D_S^{(-)}$, $D_T^{(+)}$ and $D_T^{(-)}$ be the training sets of hypernymy and non-hypernymy relations from the source and target languages, respectively. For low-resourced languages, the training sets are usually highly limited in size. Hence, we impose two assumptions, i.e., $|D_S^{(+)}| \gg |D_T^{(+)}|$ and $|D_S^{(-)}| \gg |D_T^{(-)}|$. We define the task of cross-lingual hypernymy prediction as follows:

Definition 4.1. (Cross-lingual Hypernymy Prediction) The goal is to train a classifier f over $D_S^{(+)}$, $D_S^{(-)}$ of the source language and $D_T^{(+)}$, $D_T^{(-)}$ of the target language, to predict hypernymy relations $U_T = \{(x_i, y_i)\}$ of the target language.

Recently, significant progress has been made in neural machine translation [39]. However, such translation technique is not sufficient to solve our task by translating training sets of the source language to the target language directly. The reasons are twofold. i) The training of neural machine translation models require a large amount of high-quality bilingual data in order to achieve high accuracy. For low-resourced languages, acquiring such data is difficult, or sometimes even infeasible [55]. ii) Cross-lingual hypernymy prediction requires the translation of words without any contexts, rather than complete sentences.

In this work, we adopt Conneau et al. [8] to learn the mappings among words across different languages. Before we train TMWP or ITMWP models, we follow the work [8] and learn a $d \times d$ projection matrix $\mathbf{S}$, which maps the embedding vector $\vec{x}$ in the source language space to $\vec{x}^*$ in the target language space such that x and x^* are terms of two languages share the same meaning.

4.1.1 Learning Hypernymy Projections. Similar to the monolingual case, the word pairs from $D_S^{(+)}$ and $D_T^{(+)}$ are grouped into K clusters. If $(x_i, y_i^{(+)}) \in D_T^{(+)}$, the features that we use for clustering are $\vec{x}_i - \vec{y}_i^{(+)}$; otherwise, we use $\mathbf{S}\vec{x}_i - \mathbf{S}\vec{y}_i^{(+)}$ as features. The objective function w.r.t. the jth cluster is as follows:

$$\begin{aligned}\min J(\mathbf{M}_j^{(+)}) =& \frac{\beta}{2} \sum_{i=1}^{|D_S^{(+)}|} a_{i,j}^{(+)} \gamma_i^{(+)} \|\mathbf{M}_j^{(+)}\mathbf{S}\vec{x}_i - \mathbf{S}\vec{y}_i^{(+)}\|^2 \\ &+ \frac{1-\beta}{2} \sum_{i=1}^{|D_T^{(+)}|} a_{i,j}^{(+)} \|\mathbf{M}_j^{(+)}\vec{x}_i - \vec{y}_i^{(+)}\|^2 \\ \text{s. t. } &\mathbf{M}_j^{(+)T}\mathbf{M}_j^{(+)} = \mathbf{I},\ \sum_{i=1}^{|D_S^{(+)}|} a_{i,j}^{(+)}\gamma_i^{(+)} = 1,\ \sum_{i=1}^{|D_T^{(+)}|} a_{i,j}^{(+)} = 1\end{aligned} \tag{4}$$

where $\gamma_i^{(+)}$ is a weight factor that imposes different scores to hypernymy relations in $D_S^{(+)}$. $\beta \in (0, 1)$ is a pre-defined balance factor that gives different importance to losses of the source and target languages. We define the unnormalized score $\tilde{\gamma}_i^{(+)}$ by:

$$\tilde{\gamma}_i^{(+)} = \cos(\mathbf{S}\vec{x}_i - \mathbf{S}\vec{y}_i^{(+)}, \frac{1}{|D_T^{(+)}|} \sum_{(x_j, y_j^{(+)}) \in D_T^{(+)}} \vec{x}_j - \vec{y}_j^{(+)})$$

which qualifies the semantic similarity between a pair $(x_i, y_i^{(+)}) \in D_S^{(+)}$ and all the hypernymy relations in $D_T^{(+)}$. Because $\sum_{i=1}^{|D_S^{(+)}|} a_{i,j}^{(+)} \gamma_i^{(+)} = 1$, $\gamma_i^{(+)}$ is computed by normalizing $\tilde{\gamma}_i^{(+)}$: $\gamma_i^{(+)} = \frac{\tilde{\gamma}_i^{(+)}}{\sum_{i'=1}^{|D_S^{(+)}|} a_{i',j}^{(+)} \gamma_{i'}^{(+)}}$.

After all the weights are computed, $\mathbf{M}_j^{(+)}$ can be learned by minimizing $J(\mathbf{M}_j^{(+)})$. We present the following theorem to compute the optimal solution of $\mathbf{M}_j^{(+)}$ in Eq. (4):

Theorem 4.2. *Eq. (4) is a variant of the d-dimensional Wahba's problem, which has a closed-form solution as follows:*

(1) $\mathbf{B}_j = \beta \sum_{i=1}^{|D_S^{(+)}|} a_{i,j}^{(+)} \gamma_i^{(+)} \mathbf{S}\vec{y}_i^{(+)} (\mathbf{S}\vec{x}_i)^T + (1-\beta) \sum_{i=1}^{|D_T^{(+)}|} a_{i,j}^{(+)} \vec{y}_i^{(+)} \vec{x}_i^T$;
(2) $SVD(\mathbf{B}_j) = \mathbf{U}_j \mathbf{\Sigma}_j \mathbf{V}_j^T$;
(3) $\mathbf{R}_j = diag(\underbrace{1, \ldots, 1}_{d-1}, det(\mathbf{U}_j) det(\mathbf{V}_j))$;
(4) $\mathbf{M}_j^{(+)} = \mathbf{U}_j \mathbf{R}_j \mathbf{V}_j^T$;

Proof. We omit the subscript j and the superscript $(+)$ of all variables in the objective function. Eq. (4) can be re-written as:

$$\min J(\mathbf{M}) = \frac{\beta}{2} \sum_{i=1}^{|D_S|} a_i \gamma_i \|\mathbf{M}\mathbf{S}\vec{x}_i - \mathbf{S}\vec{y}_i\|^2 + \frac{1-\beta}{2} \sum_{i=1}^{|D_T|} a_i \|\mathbf{M}\vec{x}_i - \vec{y}_i\|^2$$
$$\text{s. t. } \mathbf{M}^T\mathbf{M} = \mathbf{I}, \sum_{i=1}^{|D_S|} a_i \gamma_i = 1, \sum_{i=1}^{|D_T|} a_i = 1$$

Hence, $\beta \sum_{i=1}^{|D_S|} a_i \gamma_i + (1-\beta) \sum_{i=1}^{|D_T|} a_i = 1$. We can see that each pair $(x_i, y_i) \in D_S$ has the weight $\beta a_i \gamma_i$, and each pair $(x_i, y_i) \in D_T$ has the weight $(1-\beta) a_i$. Define the matrix $\mathbf{B}$ as:

$$\mathbf{B} = \beta \sum_{i=1}^{|D_S|} a_i \gamma_i \mathbf{S}\vec{y}_i (\mathbf{S}\vec{x}_i)^T + (1-\beta) \sum_{i=1}^{|D_T|} a_i \vec{y}_i \vec{x}_i^T$$

The objective $J(\mathbf{M})$ in Eq. (4) can be re-written as follows: $J(\mathbf{M}) = 1 - \text{tr}(\mathbf{M}\mathbf{B}^T)$. Therefore, we turn Eq. (4) into the d-dimensional Wahba's problem. The rest of the steps are the same as the previous proof, which are omitted here. □

Complexity analysis. For simplicity, let $n_1 = |D_S^{(+)}|$ and $n_2 = |D_T^{(+)}|$. The construction of $\mathbf{B}_j$ takes $O((n_1+n_2)d^2)$ time. The total time complexity is $O((n_1+n_2)d^2 + d^3)$.

Combining the objective functions w.r.t. all K clusters, the objective function for learning projected hypernymy embeddings in the cross-lingual case is derived as follows:

$$\min \tilde{J}(\mathcal{M}^{(+)}) = \frac{\beta}{2} \sum_{j=1}^{K} \sum_{i=1}^{|D_S^{(+)}|} a_{i,j}^{(+)} \gamma_i^{(+)} \|\mathbf{M}_j^{(+)} \mathbf{S}\vec{x}_i - \mathbf{S}\vec{y}_i^{(+)}\|^2$$
$$+ \frac{1-\beta}{2} \sum_{j=1}^{K} \sum_{i=1}^{|D_T^{(+)}|} a_{i,j}^{(+)} \|\mathbf{M}_j^{(+)} \vec{x}_i - \vec{y}_i^{(+)}\|^2$$
$$\text{s. t. } {\mathbf{M}_j^{(+)}}^T \mathbf{M}_j^{(+)} = \mathbf{I}, \sum_{i=1}^{|D_S^{(+)}|} a_{i,j}^{(+)} \gamma_i^{(+)} = 1, \sum_{i=1}^{|D_T^{(+)}|} a_{i,j}^{(+)} = 1,$$
$$j = 1, \ldots, K$$

which is referred as the Transfer Multi-Wahba Projection (TMWP) model.

4.1.2 Learning Non-hypernymy Projections. Similarly, to learn non-hypernymy projections, we group $D_S^{(-)}$ and $D_T^{(-)}$ into K clusters, using $\vec{x}_i - \vec{y}_i^{(-)}$ and $\mathbf{S}\vec{x}_i - \mathbf{S}\vec{y}_i^{(-)}$ as features, respectively. The objective function of TMWP is defined as follows:

$$\min \tilde{J}(\mathcal{M}^{(-)}) = \frac{\beta}{2} \sum_{j=1}^{K} \sum_{i=1}^{|D_S^{(-)}|} a_{i,j}^{(-)} \gamma_i^{(-)} \|\mathbf{M}_j^{(-)} \mathbf{S}\vec{x}_i - \mathbf{S}\vec{y}_i^{(-)}\|^2$$
$$+ \frac{1-\beta}{2} \sum_{j=1}^{K} \sum_{i=1}^{|D_T^{(-)}|} a_{i,j}^{(-)} \|\mathbf{M}_j^{(-)} \vec{x}_i - \vec{y}_i^{(-)}\|^2$$
$$\text{s. t. } {\mathbf{M}_j^{(-)}}^T \mathbf{M}_j^{(-)} = \mathbf{I}, \sum_{i=1}^{|D_S^{(-)}|} a_{i,j}^{(-)} \gamma_i^{(-)} = 1, \sum_{i=1}^{|D_T^{(-)}|} a_{i,j}^{(-)} = 1,$$
$$j = 1, \ldots, K$$

The learning process of the parameters $\mathcal{M}^{(-)}$ is the same as $\mathcal{M}^{(+)}$.

4.1.3 Hypernymy Prediction Classifier. After $\mathcal{M}^{(+)}$ and $\mathcal{M}^{(-)}$ are learned, we train the hypernymy relation classifier f for the cross-lingual case, with the procedure illustrated in Algorithm 2.

Algorithm 2 Cross-lingual Hypernymy Prediction (TMWP)

```
Perform K-means over D_S^(+) and D_T^(+), with features as S x_i − S y_i^(+) and x_i − y_i^(+), respectively;
Perform K-means over D_S^(−) and D_T^(−), with features as S x_i − S y_i^(−) and x_i − y_i^(−), respectively;
for j = 1 to cluster number K do
    Learn M_j^(+) and M_j^(−) by the closed-form solution;
end for
for each pair (x_i, y_i^(+)) ∈ D_S^(+) do
    Compute F^(+)(S x_i, S y_i^(+)) and F^(−)(S x_i, S y_i^(+));
end for
for each pair (x_i, y_i^(+)) ∈ D_T^(+) do
    Compute F^(+)(x_i, y_i^(+)) and F^(−)(x_i, y_i^(+));
end for
for each pair (x_i, y_i^(−)) ∈ D_S^(−) do
    Compute F^(+)(S x_i, S y_i^(−)) and F^(−)(S x_i, S y_i^(−));
end for
for each pair (x_i, y_i^(−)) ∈ D_T^(−) do
    Compute F^(+)(x_i, y_i^(−)) and F^(−)(x_i, y_i^(−));
end for
Train neural network classifier f over D_S^(+), D_S^(−), D_T^(+) and D_T^(−);
```

4.2 Iterative Transfer MWP

Because the sizes of $D_T^{(+)}$ and $D_T^{(-)}$ are highly limited, TMWP can encode very little knowledge from the target language. As different languages may have culture-specific words whose semantics can not be acquired using transfer learning, the performance is likely to suffer [7].

To further improve the performance, we introduce the Iterative Transfer Multi-Wahba Projection (ITMWP) model. Recall that $U_T = \{(x_i, y_i)\}$ is the collection of unlabeled term pairs of the target

Relation	BLESS	ENTAILMENT
Hypernymy	1,337	1,385
Other relations (Non-hypernymy)	13,210	1,385

Table 1: Statistic summarization of two general-domain hypernymy datasets.

language. The algorithmic procedure is shown in Algorithm 3. In each iteration, after $\mathcal{M}^{(+)}$ and $\mathcal{M}^{(-)}$ are learned, for each pair $(x_i, y_i) \in U_T$, we compute the confidence score as follows:

$$conf(x_i, y_i) = \frac{|\|\mathcal{F}^{(+)}(\vec{x}_i, \vec{y}_i)\|_2 - \|\mathcal{F}^{(-)}(\vec{x}_i, \vec{y}_i)\|_2|}{\max\{\|\mathcal{F}^{(+)}(\vec{x}_i, \vec{y}_i)\|_2, \|\mathcal{F}^{(-)}(\vec{x}_i, \vec{y}_i)\|_2\}}$$

We use the confident score instead of the output of the classifier, because modern neural networks do not generate calibrated probabilistic distributions [11]. Given a threshold τ, if $conf(x_i, y_i) > \tau$, we add the pair (x_i, y_i) to the training set (either $D_T^{(+)}$ or $D_T^{(-)}$, depending on the prediction label). The TMWP models are iteratively trained over enlarged datasets until the performance does not increase over the development set. The detailed procedure of ITMWP is shown in Algorithm 3.

Algorithm 3 Cross-lingual Hypernymy Prediction (ITMWP)

```
Train TMWP over D_S^(+), D_S^(-), D_T^(+) and D_T^(-) by Algorithm 2;
while not converge do
    for each pair (x_i, y_i) ∈ U_T do
        if conf(x_i, y_i) > τ then
            if f(x_i, y_i) = HYPERNYMY then
                Update D_T^(+) = D_T^(+) ∪ {(x_i, y_i)};
            else
                Update D_T^(-) = D_T^(-) ∪ {(x_i, y_i)};
            end if
            Update U_T = U_T \ {(x_i, y_i)}
        end if
    end for
    Update TMWP over D_S^(+), D_S^(-), D_T^(+) and D_T^(-) by Algorithm 2;
end while
```

5 MONOLINGUAL EXPERIMENTS

In this section, we conduct extensive experiments to evaluate MWP over the tasks of supervised hypernymy detection and unsupervised hypernymy relation classification. We compare it with state-of-the-art approaches to make the convincing conclusion.

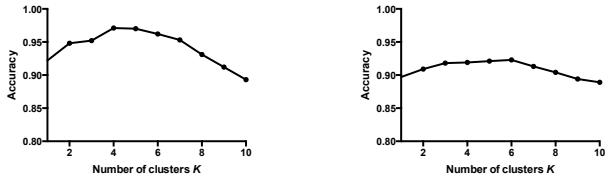
5.1 General-domain Supervised Hypernymy Detection

5.1.1 Task Description. We first evaluate MWP by supervised hypernymy detection in the general domain, which aims at classifying whether the hypernymy relation holds in a term pair. Experiments are conducted over two benchmark datasets that are frequently used in the NLP community: BLESS [2] and ENTAILMENT [1], consisting of 14,547 and 2,770 word pairs with labeled relations (i.e., hypernymy relations or other relations), respectively. The statistics of the two datasets are summarized in Table 1.

To obtain all the term embeddings for learning projection matrices and the hypernymy relation classifier, we use the same procedure as in Bojanowski et al. [4] to train fastText word embeddings over the entire Wikipedia corpus, with the dimensionality of word embeddings set to $d = 300$.

Method	BLESS	ENTAILMENT
Mikolov et al. [24]	0.84	0.83
Yu et al. [54]	0.90	0.87
Luu et al. [20]	0.93	0.91
Nguyen et al. [26]	0.94	0.91
MWP (Non-orthogonal)	**0.95**	0.90
MWP	**0.97**	**0.92**

Table 2: Performance comparison of general-domain supervised hypernymy detection in terms of accuracy.



(a) Dataset: BLESS (b) Dataset: ENTAILMENT

Figure 3: Parameter analysis of K over two datasets.

For evaluation, we follow exactly the same "leave-one-out" evaluation protocols that have been used in a series of NLP papers [20, 26, 54]. For the BLESS dataset [2], because the relations are related to 200 most frequent nouns in WordNet [25], we randomly select relations w.r.t. one noun for testing, and train the MWP model and the relation prediction classifier on others. For the ENTAILMENT dataset [1], one hypernymy relation is randomly selected for testing and the model is trained over others. All the experimental results are reported in averaged accuracy.

5.1.2 Baselines. To evaluate whether the learned features are useful to distinguishing hypernymy vs. non-hypernymy relations, we compare MWP against five hypernymy embedding methods:[6]

- Word2Vec [24]: It is the standard Skip-gram model, trained via the negative sampling technique.
- Yu et al. [54]: It learns hypernymy embeddings for terms based on the Probase taxonomy [51]. The algorithm is optimized by a max-margin neural network.
- Luu et al. [20]: It improves hypernymy embedding learning by a dynamic weighting neural model.
- HyperVec [26]: It combines the negative sampling based Skip-gram model [24] and the WordNet concept hierarchy [25] to learn hypernymy embeddings.
- MWP (Non-orthogonal): It is a variant of the MWP model, without orthogonal constraints on projection matrices.

5.1.3 Experimental Results. The experimental results are summarized in Table 2. In the implementation, as a default setting, we fix the number of clusters of MWP as $K = 4$. The parameter analysis of K will be presented further. From the results, we can see that MWP outperforms all previous baseline approaches. Simple models such as Word2Vec [24] do not produce satisfactory results because

[6]Based on the respective papers, the features of [20, 24, 54] are the hypernymy embeddings of two terms and the vector difference. The features of [26] are the vector difference and cosine similarity between two terms' embeddings, together with the magnitudes of the two embeddings. An SVM classifier is trained over all the features to make the prediction.

Statistics	ANIMAL	PLANT	VEHICLE
# Distinct terms	659	520	117
# Hypernymy relations	4,164	2,266	283
# Random word pairs	8,471	4,520	586

Table 3: Statistic summarization of three domain-specific hypernymy datasets.

Method	ANIMAL	PLANT	VEHICLE
Mikolov et al. [24]	0.80	0.81	0.82
Yu et al. [54]	0.67	0.65	0.70
Luu et al. [20]	0.89	0.92	0.89
Nguyen et al. [26]*	0.83	0.91	0.83
MWP (Non-orthogonal)	**0.90**	**0.92**	0.87
MWP	**0.92**	**0.94**	**0.90**

Table 4: Performance comparison of domain-specific supervised hypernymy detection in terms of accuracy. * refers to the result based on our own implementation.

the semantics of hypernymy relations are not explicitly modeled. Hence, they are probable to suffer from the "lexical memorization" problem. Compared to the strongest competitor HyperVec [26], the accuracy of MWP is higher by 3% over BLESS and higher by 1% over ENTAILMENT. The comparison between the MWP model and the variant MWP (Non-orthogonal) shows that the orthogonal constraint improves projection learning in MWP.

As there is no gold standard to determine the number of "latent components" in a hypernymy relation dataset, the choice of parameter K in MWP is mostly heuristic. Here, we investigate how the changes of K affect the performance of MWP. We set K from 1 to 10 and report the averaged accuracy over the two datasets. The results are shown in Figure 3. As seen, the performance is not very sensitive to K if the value of K is not extremely large or small. This is because we introduce the weights $a_{i,j}^{(+)}$ and $a_{i,j}^{(-)}$ in the model, resulting in the situations where all the projections are "fuzzy". Additionally, by using the clustering technique, the accuracy of MWP is boosted by over 5% and 2%, respectively.

5.2 Domain-specific Supervised Hypernymy Detection

5.2.1 Task Description. We further evaluate the effectiveness of MWP method over domain-specific datasets. In this set of experiments, we use three datasets derived from the following domain-specific taxonomies: ANIMAL, PLANT and VEHICLE [40]. The respective three evaluation datasets are constructed by extracting all possible taxonomic relations from taxonomies as possible samples and randomly pairing two terms that are not involved in any hypernymy relations as negative samples. We use the same datasets that have been generated and released by Luu et al. [20]. The statistics are summarized in Table 3.

For evaluation, we also follow the settings of Luu et al. [20]. Each time, we hold out relations w.r.t. one term for testing and train our model on the remaining terms. The results are also reported in averaged accuracy. Because a few terms in the three datasets are multi-word expressions (e.g., American tree, half track), we treat these terms as a whole to train the fastText word embeddings [4]. The default experimental settings and baselines are the same as in the general-domain experiments. Hence, we do not repeat the details again.

5.2.2 Experimental Results. From the experimental results in Table 4, it can be concluded that the proposed MWP model has high performance for hypernymy detection in specific domains. Specifically, the non-orthogonal version of MWP outperforms state-of-the-art methods over two domain-specific datasets (PLANT and ANIMAL) and is comparable to the strongest baseline Luu et al. [20] over the other one (VEHICLE). The full implementation of MWP outperforms all the baselines over the three datasets. Another interesting observation is that methods that use general corpora to training word embeddings (i.e., [20, 24] and ours) have relatively higher performance than methods that only consider the taxonomy data (i.e., Yu et al. [54]). Nguyen et al. [26] learn hypernymy embeddings using the general WordNet concept hierarchy, but still have relatively low performance in specific domains. This is because concepts in the taxonomy Probase [51] or in the WordNet concept hierarchy usually have low converge for specific domains, leading to low prediction performance.

5.3 Unsupervised Hypernymy Relation Classification

5.3.1 Task Description and Evaluation Protocols. This set of experiments involves two studies that compare MWP against various hypernymy measures. We follow the evaluation framework of Nguyen et al. [26], Roller et al. [30] over two benchmark datasets: BLESS and WBLESS, constructed by Kiela et al. [15] and Weeds et al. [49], respectively. In the first experiment, we aim at predicting the directionality of all the 1,337 hypernymy relations in BLESS. All the hypernymy relations are treated as positive samples and all the reverse-hypernymy relations are treated as negative samples. WBLESS is derived from BLESS, including a subset of 1,168 pairs. The experiment over WBLESS is more challenging, which is a binary classification task, aiming at distinguishing hypernymy relations and other relations (including a mixture of reverse-hypernymy, meronymy, co-hyponymy relations and randomly matched nouns).

The baseline hypernymy measures that we consider in this work are summarized as follows. Santus et al. [31],Weeds et al. [49], Kiela et al. [15] and Nguyen et al. [26] employ distributional methods for unsupervised hypernymy relation classification. Roller et al. [30] is the state-of-the art path-based measure by utilizing generalized Hearst patterns in a large text corpus.

In contrast to these hypernymy measures, the MWP model is a supervised relation classification model, instead of an unsupervised measure. Hence, MWP can not be directly applied to this task. Instead of using the MWP based classifier f, we design a hypernymy score based on the features:

$$\tilde{s}(x_i, y_i) = \|\mathcal{F}^{(-)}(\vec{x}_i, \vec{y}_i)\|_2 - \|\mathcal{F}^{(+)}(\vec{x}_i, \vec{y}_i)\|_2 \quad (5)$$

where the projection matrices w.r.t. $\mathcal{F}^{(+)}(x_i, y_i)$ are learned over hypernymy relations and the projection matrices w.r.t. $\mathcal{F}^{(-)}(x_i, y_i)$ are learned over others. For a term pair (x_i, y_i), if x_i is the hyponym and y_i is the hypernym, the norm of $\mathcal{F}^{(+)}(x_i, y_i)$ is likely to be smaller than that of $\mathcal{F}^{(-)}(x_i, y_i)$. Hence, we use $\tilde{s}(x_i, y_i) > \tilde{s}(y_i, x_i)$ to predict y_i is the hypernym of x_i.

Relation↓ Language →	fr	zh	ja	it	th	fi	el
# Hypernymy relations	4,035	2,962	1,448	3,034	1,156	7,157	2,612
# Non-hypernymy relations	8,947	6,382	3,203	6,081	1,977	9,433	1,454

Table 5: Statistics of hypernymy and non-hypernymy relation datasets of seven non-English languages. Language abbreviations: French (fr), Chinese (zh), Japanese (ja), Italian (it), Thai (th), Finnish (fn) and Greek (el).

Measure	BLESS	WBLESS
Santus et al. [31]	0.87	-
Weeds et al. [49]	-	0.75
Kiela et al. [15]	0.88	0.75
Nguyen et al. [26]	0.92	0.87
Roller et al. [30]	0.96	0.87
MWP (Non-orthogonal)	0.95	**0.89**
MWP	**0.97**	**0.92**

Table 6: Performance comparison of unsupervised hypernymy relation classification in terms of accuracy.

Because all the projection matrices require to be pre-trained before we can compute the features $\mathcal{F}^{(+)}(\vec{x}_i, \vec{y}_i)$ and $\mathcal{F}^{(-)}(\vec{x}_i, \vec{y}_i)$, we employ 14,135 hypernymy relations from the dataset Shwartz [34] as the training data, after removing pairs appearing in BLESS to avoid overfitting. Note that the proposed hypernymy measure in Eq. (5) is unsupervised because the Shwartz data is only employed to learn projection matrices. We do not train the hypernymy relation classifier to make the prediction over BLESS and WBLESS, nor do we use any BLESS or WBLESS data to learn projection matrices.

5.3.2 Experimental Results. The experimental results are summarized in Table 6. We can see that among all the baselines, Roller et al. [30] has the highest performance, with the accuracies as 96% and 87%, respectively. Compared to Roller et al. [30], the MWP model outperforms the method by 1% and 5% in terms of accuracy. Although the MWP model is primarily built for supervised hypernymy relation classification, by adding some slight modifications, it is also is highly effective for predicting the hypernymy relations in an unsupervised manner.

Note that we only focus on the binary classification of hypernymy relations in this work. In the future, we will extend our work to the multi-way classification of semantic relations, and evaluate it over other multi-way classification datasets (e.g., BIBLESS [26]).

6 CROSS-LINGUAL EXPERIMENTS

In this section, we evaluate two cross-lingual models (TMWP and ITMWP) over two tasks: cross-lingual hypernymy direction classification and cross-lingual hypernymy detection.

6.1 Datasets and Experimental Settings

We select English as the source language due to its wide usage and the availability of large training sets. We refer to the survey [45], and combine the five human-labeled datasets as the underlying training set of the source language: BLESS [2], Shwartz [34], Kotlerman [16], Turney [37] and ENTAILMENT [1]. After removing of duplicates and multi-word expressions, we create a large English dataset, consisting of 85,234 word pairs, containing 17,394 hypernymy relations and 67,930 non-hypernymy relations (including a mixture of other relations). All the datasets are available at https://chywang.github.io/data/www2019.zip.

For non-English languages, we utilize the Open Multilingual Wordnet project [5] to create training and testing sets[7]. We take seven non-English languages as target languages: French, Chinese, Japanese, Italian, Thai, Finnish and Greek. Versions of Wordnets are Wordnet Libre du Français (French), Chinese Open Wordnet, Japanese Wordnet, ItalWordnet (Italian), Thai Wordnet, FinnWordnet (Finnish) and Greek Wordnet. Hypernymy relations are generated by randomly sampling hypernymy relations from the Multilingual Wordnet concept hierarchies. Non-hypernymy relations consist of a mixture of holonymy, synonymy and randomly matched word pairs. The statistics of all the seven datasets are summarized in Table 5. For seven non-English languages, word embeddings are taken from pre-trained fastText word embeddings [4]. The cross-lingual mapping matrices are trained using the original code of Conneau et al. [8] over multi-lingual Wikipedia corpora with default parameter settings. The dimensions of word embeddings of all languages are set to $d = 300$.

6.2 Evaluation Protocols

We evaluate the two cross-lingual models (i.e., TMWP and ITMWP) over two cross-lingual hypernymy prediction tasks. The first is cross-lingual hypernymy direction classification. Its goal is to predict the directionality of hypernymy relations for the target language given the training sets in English and the target language. We take hypernymy relations as positive data, and reverse-hypernymy relations as negative data to train and evaluate our model. The second task is cross-lingual hypernymy detection. It aims at distinguishing hypernymy vs. non-hypernymy relations for the target language based on both datasets in English and the target language. In the experiments, we use all the data in English and 20% of the non-English language datasets for training, 20% for development and the rest 60% for testing, partitioned randomly. By rotating the 5-fold subsets of the non-English language datasets, we report the performance of all the models in terms of averaged accuracy.

Because pattern-based methods and a few distributional methods for hypernymy prediction is highly language dependent, they are not suitable for cross-lingual hypernymy prediction over arbitrary languages. Hence, we follow the evaluation protocols that used in Shwartz et al. [34]. We employ several state-of-the-art distributional approaches as baselines, introduced as follows:

- Santus et al. [31]: It is an entropy-based hypernymy measure SLQS that characterizes the semantic generality of terms.
- Kiela et al. [15]: It is a distributional generality measure that models the hierarchical property of hypernymy relations.
- Weeds et al. [49]: It is a supervised distributional model based on vector offsets of term pairs.
- Shwartz et al. [34]: It is a neural network architecture for relation classification. Because several non-English languages

[7] http://compling.hss.ntu.edu.sg/omw/

Method	fr	zh	ja	it	th	fi	el
Task: cross-lingual hypernymy direction classification							
Santus et al. [31]	0.65	0.65	0.68	0.61	0.63	0.70	0.62
Weeds et al. [49]	0.76	0.71	0.77	0.76	0.72	0.77	0.70
Kiela et al. [15]	0.67	0.65	0.71	0.68	0.65	0.70	0.62
Shwartz et al. [34]	0.79	0.67	0.71	0.72	0.66	0.75	0.66
TMWP (N)	0.78	0.71	0.75	0.76	0.73	0.76	0.71
TMWP	0.80	0.72	0.76	0.78	0.75	0.78	0.73
ITMWP (N)	**0.82**	0.72	0.76	0.78	0.75	**0.81**	0.72
ITMWP	0.81	**0.74**	**0.78**	**0.81**	**0.78**	**0.81**	**0.75**
Task: cross-lingual hypernymy detection							
Santus et al. [31]	0.67	0.63	0.67	0.62	0.64	0.62	0.64
Weeds et al. [49]	0.74	0.66	0.68	0.71	0.62	0.68	0.69
Kiela et al. [15]	0.70	0.61	0.65	0.68	0.57	0.61	0.67
Shwartz et al. [34]	0.72	0.66	0.69	0.64	0.66	0.69	0.70
TMWP (N)	0.72	0.67	0.70	0.70	0.68	0.71	0.70
TMWP	0.75	0.71	0.76	0.72	0.69	0.72	0.71
ITMWP (N)	0.72	**0.74**	0.77	**0.74**	0.67	0.71	0.72
ITMWP	**0.76**	0.73	**0.78**	**0.74**	**0.72**	**0.73**	**0.73**

Table 7: Performance comparison of two cross-lingual hypernymy prediction tasks in terms of accuracy. TMWP (N) and ITMWP (N) are stand for TMWP (Non-orthogonal) and ITMWP (Non-orthogonal).

(e.g., Chinese, Thai) lack high-quality hypernymy patterns. The path-based sub-networks are not implemented.

- TMWP (Non-orthogonal) and ITMWP (Non-orthogonal): They are the variants of TMWP and ITMWP without the orthogonal constraint on projection matrices, respectively.

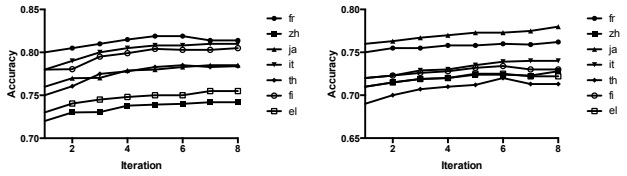
Apart from the proposed two models and their variants, the remaining competing approaches [15, 31, 34, 49] are not designed for cross-lingual hypernymy prediction. To implement all the baselines for the two cross-lingual hypernymy prediction tasks, we employ Conneau et al. [8] to translate the embeddings of terms in the source language to those of the target languages. Next, we train all the baseline models over the English dataset after translating it to the target language and the training set of the target language. For simplicity, we set $K = 4$ and $\beta = 0.5$ for TMWP and ITMWP and their variants in all the cross-lingual experiments.

6.3 Experimental Results

The experimental results of both tasks over all the seven non-English languages are summarized in Table 7.

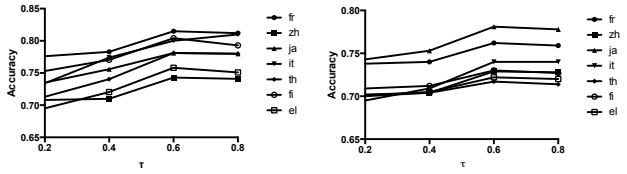
From the results, we can draw the following conclusions. i) For both tasks, ITMWP and TMWP outperform all the previous methods over all seven non-English languages. Overall, the proposed models boost the accuracy by 2% $\sim$ 9% for cross-lingual hypernymy direction classification, depending on different languages. The performance of our models for cross-lingual hypernymy detection is similar to that of the direction classification task. ii) By using unsupervised neural word translation and the orthogonal constraint, knowledge from the source languages can be employed to improve the performance of hypernymy prediction of the target language. iii) The performance of cross-lingual hypernymy direction classification is generally higher than cross-lingual hypernymy detection, indicating the latter task is more challenging.

We further investigate how ITMWP improves the performance. We fix $\tau = 0.7$ and run our algorithm for 8 iterations over the two tasks. The results are illustrated in Figure 4. As seen, the accuracy increases steadily during the first few iterations. The performance becomes relatively stable because there are no sufficient number of relations with high confidence that can be added to training sets.



(a) Task: cross-lingual hypernymy direction classification (b) Task: cross-lingual hypernymy detection

Figure 4: Performance of ITMWP over two cross-lingual hypernymy prediction tasks.



(a) Task: cross-lingual hypernymy direction classification (b) Task: cross-lingual hypernymy detection

Figure 5: Parameter analysis of τ over two cross-lingual hypernymy prediction tasks.

Additionally, we tune the parameter τ in ITMWP and run the algorithm in 5 iterations. The results are summarized in Table 5. It shows that the changes of performance trend are similar across different languages. The value of τ reflects the trade-off between the number of pairs to be added to the training set and the accuracies of these pairs. When τ is small, the algorithm tends to add more pairs to the training set, unavoidably introducing errors to the training set. In contrast, when τ is large, the effect of training data augmentation is reduced. We suggest that the algorithm achieves the best performance when τ is set around 0.7.

7 CONCLUSION AND FUTURE WORK

In this paper, we propose a family of fuzzy orthogonal projection models for monolingual and cross-lingual hypernymy prediction. It includes three models: MWP, TMWP and ITMWP. MWP distinguishes hypernymy vs. non-hypernymy relations based on distributional fuzzy mappings from embeddings of a term to those of its hypernyms and non-hypernyms. TMWP and ITMWP are designed to transfer the semantic knowledge from the source language to target languages for cross-lingual hypernymy prediction. Experiments illustrate the effectiveness of our models over both monolingual and cross-lingual hypernymy prediction. In the future, we plan to i) extend our method to predict multiple types of semantic relations over multiple languages, and to ii) improve cross-lingual hypernymy prediction via multi-lingual embeddings.

ACKNOWLEDGMENTS

This work is supported by the National Key Research and Development Program of China under Grant No. 2016YFB1000904.

REFERENCES

[1] Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*. 23–32.

[2] Marco Baroni and Alessandro Lenci. 2011. How we BLESSed distributional semantic evaluation. In *Proceedings of the Gems 2011 Workshop on Geometrical MODELS of Natural Language Semantics*. 1–10.

[3] Chris Biemann, Dmitry Ustalov, Alexander Panchenko, and Nikolay Arefyev. 2017. Negative Sampling Improves Hypernymy Extraction Based on Projection Learning. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. 543–550.

[4] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *TACL* 5 (2017), 135–146.

[5] Francis Bond and Ryan Foster. 2013. Linking and Extending an Open Multilingual Wordnet. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. 1352–1362.

[6] Haw-Shiuan Chang, ZiYun Wang, Luke Vilnis, and Andrew McCallum. 2018. Distributional Inclusion Vector Embedding for Unsupervised Hypernymy Detection. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 485–495.

[7] Nurendra Choudhary, Rajat Singh, and Manish Shrivastava. 2018. Cross-Lingual Task-Specific Representation Learning for Text Classification in Resource Poor Languages. *CoRR* abs/1806.03590 (2018). http://arxiv.org/abs/1806.03590

[8] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word Translation Without Parallel Data. *CoRR* abs/1710.04087 (2017). arXiv:1710.04087 http://arxiv.org/abs/1710.04087

[9] Aleksandr Drozd, Anna Gladkova, and Satoshi Matsuoka. 2016. Word Embeddings, Analogies, and Machine Learning: Beyond king - man + woman = queen. In *Proceedings of the 26th International Conference on Computational Linguistics*. 3519–3530.

[10] Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning Semantic Hierarchies via Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. 1199–1209.

[11] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*. 1321–1330.

[12] Amit Gupta, Francesco Piccinno, Mikhail Kozhevnikov, Marius Pasca, and Daniele Pighin. 2016. Revisiting Taxonomy Induction over Wikipedia. In *Proceedings of the 26th International Conference on Computational Linguistics*. 2300–2309.

[13] Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*. 539–545.

[14] Matthias Keller, Patrick Mühlschlegel, and Hannes Hartenstein. 2013. Search result presentation: supporting post-search navigation by integration of taxonomy data. In *Proceedings of the 22nd International World Wide Web Conference, Companion Volume*. 1269–1274.

[15] Douwe Kiela, Laura Rimell, Ivan Vulic, and Stephen Clark. 2015. Exploiting Image Generality for Lexical Entailment Detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*. 119–124.

[16] Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering* 16, 4 (2010), 359–389.

[17] Alessandro Lenci and Giulia Benotto. 2012. Identifying hypernyms in distributional semantic spaces. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*. 75–79.

[18] Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. Do Supervised Distributional Methods Really Learn Lexical Inference Relations?. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 970–976.

[19] Xueqing Liu, Yangqiu Song, Shixia Liu, and Haixun Wang. 2012. Automatic taxonomy construction from keywords. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1433–1441.

[20] Anh Tuan Luu, Yi Tay, Siu Cheung Hui, and See-Kiong Ng. 2016. Learning Term Embeddings for Taxonomic Relation Identification Using Dynamic Weighting Neural Network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 403–413.

[21] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. 2015. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *Proceedings of the Seventh Biennial Conference on Innovative Data Systems Research*.

[22] F Landis Markley. 1988. Attitude determination using vector observations and the singular value decomposition. *Journal of the Astronautical Sciences* 36, 3 (1988), 245–258.

[23] F Landis Markley and John L Crassidis. 2014. *Fundamentals of spacecraft attitude determination and control*. Vol. 33. Springer.

[24] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013). arXiv:1301.3781 http://arxiv.org/abs/1301.3781

[25] George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[26] Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017. Hierarchical Embeddings for Hypernymy Detection and Directionality. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 233–243.

[27] Simone Paolo Ponzetto and Michael Strube. 2007. Deriving a Large-Scale Taxonomy from Wikipedia. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*. 1440–1445.

[28] Stephen Roller and Katrin Erk. 2016. Relations such as Hypernymy: Identifying and Exploiting Hearst Patterns in Distributional Vectors for Lexical Entailment. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2163–2172.

[29] Stephen Roller, Katrin Erk, and Gemma Boleda. 2014. Inclusive yet Selective: Supervised Distributional Hypernymy Detection. In *Proceedings of the 25th International Conference on Computational Linguistics*. 1025–1036.

[30] Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst Patterns Revisited: Automatic Hypernym Detection from Large Text Corpora. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 358–363.

[31] Enrico Santus, Alessandro Lenci, Qin Lu, and Sabine Schulte im Walde. 2014. Chasing Hypernyms in Vector Spaces with Entropy. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. 38–42.

[32] Enrico Santus, Vered Shwartz, and Dominik Schlechtweg. 2017. Hypernyms under Siege: Linguistically-motivated Artillery for Hypernymy Detection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. 65–75.

[33] Jiaming Shen, Zeqiu Wu, Dongming Lei, Chao Zhang, Xiang Ren, Michelle T. Vanni, Brian M. Sadler, and Jiawei Han. 2018. HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2180–2189.

[34] Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving Hypernymy Detection with an Integrated Path-based and Distributional Method. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 2389–2398.

[35] Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2004. Learning Syntactic Patterns for Automatic Hypernym Discovery. In *Advances in Neural Information Processing Systems 17*. 1297–1304.

[36] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*. 697–706.

[37] Peter D. Turney and Saif M. Mohammad. 2015. Experiments with three approaches to recognizing lexical entailment. *Natural Language Engineering* 21, 3 (2015), 437–476.

[38] Shyam Upadhyay, Yogarshi Vyas, Marine Carpuat, and Dan Roth. 2018. Robust Cross-Lingual Hypernymy Detection Using Dependency Context. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 607–618.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*. 6000–6010.

[40] Paola Velardi, Stefano Faralli, and Roberto Navigli. 2013. OntoLearn Reloaded: A Graph-Based Algorithm for Taxonomy Induction. *Computational Linguistics* 39, 3 (2013), 665–707.

[41] Ivan Vulic, Daniela Gerz, Douwe Kiela, Felix Hill, and Anna Korhonen. 2017. HyperLex: A Large-Scale Evaluation of Graded Lexical Entailment. *Computational Linguistics* 43, 4 (2017).

[42] Yogarshi Vyas and Marine Carpuat. 2016. Sparse Bilingual Word Representations for Cross-lingual Lexical Entailment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1187–1197.

[43] Grace Wahba. 1965. A least squares estimate of satellite attitude. *SIAM review* 7, 3 (1965), 409–409.

[44] Chengyu Wang and Xiaofeng He. 2016. Chinese Hypernym-Hyponym Extraction from User Generated Categories. In *Proceedings of the 26th International Conference on Computational Linguistics*. 1350–1361.

[45] Chengyu Wang, Xiaofeng He, and Aoying Zhou. 2017. A Short Survey on Taxonomy Learning from Text Corpora: Issues, Resources and Recent Advances. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1190–1203.

[46] Chengyu Wang, Junchi Yan, Aoying Zhou, and Xiaofeng He. 2017. Transductive Non-linear Learning for Chinese Hypernym Prediction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. 1394–1404.

[47] Zhigang Wang, Juanzi Li, Shuangjie Li, Mingyang Li, Jie Tang, Kuo Zhang, and Kun Zhang. 2014. Cross-Lingual Knowledge Validation Based Taxonomy Derivation from Heterogeneous Online Wikis. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. 180–186.

[48] Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen. 2015. Query Understanding through Knowledge-Based Conceptualization. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*. 3264–3270.

[49] Julie Weeds, Daoud Clarke, Jeremy Reffin, David J. Weir, and Bill Keller. 2014. Learning to Distinguish Hypernyms and Co-Hyponyms. In *Proceedings of the 25th International Conference on Computational Linguistics*. 2249–2259.

[50] Tianxing Wu, Lei Zhang, Guilin Qi, Xuan Cui, and Kang Xu. 2017. Encoding Category Correlations into Bilingual Topic Modeling for Cross-Lingual Taxonomy Alignment. In *Proceedings of the 16th International Semantic Web Conference*. 728–744.

[51] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Qili Zhu. 2012. Probase: a probabilistic taxonomy for text understanding. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 481–492.

[52] Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized Word Embedding and Orthogonal Transform for Bilingual Word Translation. In *Proccedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1006–1011.

[53] Josuke Yamane, Tomoya Takatani, Hitoshi Yamada, Makoto Miwa, and Yutaka Sasaki. 2016. Distributional Hypernym Generation by Jointly Learning Clusters and Projections. In *Proceedings of the 26th International Conference on Computational Linguistics*. 1871–1879.

[54] Zheng Yu, Haixun Wang, Xuemin Lin, and Min Wang. 2015. Learning Term Embeddings for Hypernymy Identification. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*. 1390–1397.

[55] Poorya Zaremoodi, Wray L. Buntine, and Gholamreza Haffari. 2018. Adaptive Knowledge Sharing in Multi-Task Learning: Improving Low-Resource Neural Machine Translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 656–661.

[56] Yuchen Zhang, Amr Ahmed, Vanja Josifovski, and Alexander J. Smola. 2014. Taxonomy discovery for personalized recommendation. In *Proceedings of the Seventh ACM International Conference on Web Search and Data Mining*. 243–252.

Modeling Item-Specific Temporal Dynamics of Repeat Consumption for Recommender Systems

Chenyang Wang
BNRist, DCST, Tsinghua University
Beijing, China
wangcy18@mails.tsinghua.edu.cn

Min Zhang*
BNRist, DCST, Tsinghua University
Beijing, China
z-m@tsinghua.edu.cn

Weizhi Ma
BNRist, DCST, Tsinghua University
Beijing, China
mawz14@mails.tsinghua.edu.cn

Yiqun Liu
BNRist, DCST, Tsinghua University
Beijing, China
yiqunliu@tsinghua.edu.cn

Shaoping Ma
BNRist, DCST, Tsinghua University
Beijing, China
msp@tsinghua.edu.cn

ABSTRACT

Repeat consumption is a common scenario in daily life, such as repurchasing items and revisiting websites, and is a critical factor to be taken into consideration for recommender systems. Temporal dynamics play important roles in modeling repeat consumption. It is noteworthy that for items with distinct lifetimes, consuming tendency for the next one fluctuates differently with time. For example, users may repurchase milk weekly, but it is possible to repurchase mobile phone after a long period of time. Therefore, how to adaptively incorporate various temporal patterns of repeat consumption into a holistic recommendation model has been a new and important problem.

In this paper, we propose a novel unified model with introducing Hawkes Process into Collaborative Filtering (CF). Different from most previous work which ignores various time-varying patterns of repeat consumption, the model explicitly addresses two item-specific temporal dynamics: (1) *short-term effect* and (2) *life-time effect*, which is named as **S**hort-Term and **L**ife-Time **R**epeat **C**onsumption (SLRC) model. SLRC learns importance of the two factors for each item dynamically by interpretable parameters. According to extensive experiments on four datasets in diverse scenarios, including two public collections, SLRC is superior to previous approaches for repeat consumption modeling. Moreover, due to the high flexibility of SLRC, various existing recommendation algorithms are shown to be easily leveraged in this model to achieve significant improvements. In addition, SLRC is good at balancing recommendation for novel items and consumed items (exploration and exploitation). We also find that the learned parameters is highly interpretable, and hence the model is able to be leveraged to discover items' lifetimes, and to distinguish different types of items such as durable and fast-moving consumer goods.

KEYWORDS

Recommender system, Repeat consumption, Temporal dynamics, Collaborative filtering, Hawkes process

*Contact author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313594

ACM Reference Format:
Chenyang Wang, Min Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. 2019. Modeling Item-Specific Temporal Dynamics of Repeat Consumption for Recommender Systems. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313594

1 INTRODUCTION

Recommender system plays a crucial role in users' engagements with web-services, which makes it easier for users to explore endless items that they have not consumed (novel items) [19, 23, 36]. Meanwhile, repeat consumptions come to account for a large proportion of user-item interactions and become increasingly important in recent years. Therefore, properly recommending items user consumed before becomes a vital part of recommender systems nowadays. For example, an online shopping website can recommend the milk powder user consumed before when it is time to be used up. It is widely agreed that providing consumed items at the right time can greatly improve both users' satisfaction and sellers' profit [3, 11, 12, 40].

Temporal dynamics is an important factor to accurately model repeat consumption. Previous study about user consumption sequence presents a holistic model of sequential consumption [4]. The model takes time factor, distance factor, quality factor into consideration and the time factor turns out to be the most important one. Du et al. [16] utilize temporal point process to model recurrent user activities. That is to say, previous consumptions are likely to trigger repeat consumptions for those items (self-excitation) and such excitation decays with time.

However, existing studies about repeat consumption (e.g. [3–5, 8, 16, 32]) do not focus on modeling how the tendency of repeat consumption fluctuates with time, especially the difference between items. As a result, they fail to capture various temporal dynamics across distinct items. Actually, it is noteworthy that different items demonstrate diverse temporal patterns of repeat consumption. In particular, there are both (1) *short-term* and (2) *life-time* effects. To illustrate these two factors, Figure 1 gives an example of two items' purchase tendencies (baby diaper and feeding bottle) drifting with time. For a new mother, both baby diaper and feeding bottle are needed, so the base purchase tendencies are similar at first. After she buys both items simultaneously, their purchase tendencies increase immediately and diminish with time (*short-term effect*), which means the mother probably consumes these two items again in a short term for some reasons such as future usage. On the

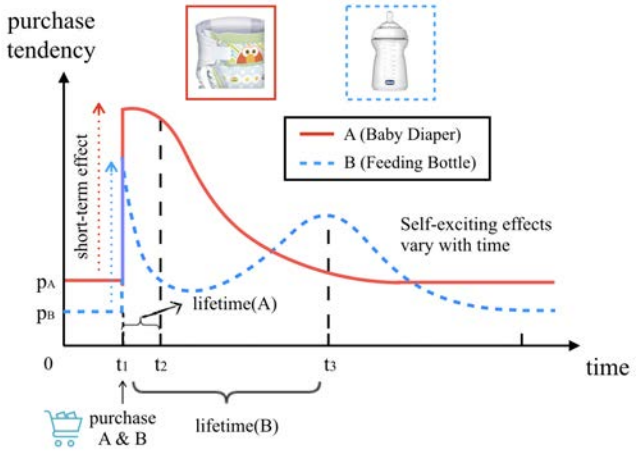


Figure 1: Illustration of two items' purchase tendencies drifting with time. There existing *short-term effect* for both baby diaper and feeding bottle after purchase, and such effect decays with time. On the other hand, *life-time effect* peaks after different periods of time, which is short for baby diaper (superposed with short-term effect) but long for feeding bottle. Both items' purchase tendencies finally decay to their basic values as time goes by.

other hand, users also tend to purchase the item again near the end of item's lifetime [20, 30]. In Figure 1, the purchase tendency of feeding bottle has another spike around t_3 (*life-time effect*), which is probably the time to change a new one. With regard to baby diaper, lifetimes of such fast-moving consumer goods are so short that the corresponding effect is superposed with short-term effect, resulting in higher tendency shortly after purchasing. As time goes by, the self-exciting effect diminishes and both items' purchase tendencies return to their basic values. Such difference between items also holds for many other cases such as snacks for A and mobiles for B.

Current methods modeling repeat consumption do not jointly model both two characteristics for different items. However, the lack of variety of temporal factors can lead to lower predictive ability. For example, when life-time effect is not considered, the dominated short-term effect will result in continuous recommendation for mobiles shortly after purchasing one, which would be annoying to users. And mobiles could not be recommended when it is the time to change to a new one.

In this paper, firstly we conduct a detailed analysis of repeat consumption and temporal dynamics on four real-world datasets spanning over 30 months. Two key factors are revealed to model the complex time-varying patterns of repeat consumption: (1) *short-term effect* and (2) *life-time effect*. We find there are many repeat consumption shortly after previous purchase even for durable goods like baby walker (short-term effect). Besides, a consumption is likely to trigger next consumptions for the same item when item's lifetime runs out (life-time effect). Moreover, the importance of short-term and life-time effect can vary wildly between distinct items.

Then, based on these findings, we propose a holistic model that can recommends both novel and consumed items simultaneously, called SLRC referring to **S**hort-Term and **L**ife-Time **R**epeat **C**onsumption model. SLRC innovatively combines collaborative filtering (CF) [24] and Hawkes Process [17], with the two temporal characteristics of repeat consumption modeled by the kernel function in Hawkes Process. The CF part of the model projects users and items to the same space, in which similar representations stand for similar properties. As a result, the CF part helps to better capture users' preferences and explore novel items. While Hawkes process utilizes patterns of repeat consumption to properly rank consumed items higher at specific time (when users need items previously consumed again), leading to a holistic model with a good balance of exploration and exploitation. Moreover, the item-specific modeling of repeat consumption makes it possible to reveal fine-grained temporal patterns and automatically discover distinct items' lifetimes, which often need to be predefined in previous work [27]. The main contributions of this work can be summarized as follows:

- Two temporal dynamics of repeat consumption: (1) short-term effect and (2) life-time effect are revealed through empirical study, which vary a lot for distinct types of items.
- A unified model, **S**hort-Term and **L**ife-Time **R**epeat **C**onsumption model (SLRC), is proposed with innovative combination of two techniques: Collaborative Filtering (CF) and Hawkes Process. The model adaptively incorporates two item-specific temporal dynamics of repeat consumption into kernel function of Hawkes Process. And the integration of CF helps SLRC to better capture users' intrinsic preference, achieving a good balance of exploration and exploitation.
- Comparative experimental results on four datasets indicate SLRC is superior to previous models of repeat consumption. SLRC is also flexible to leverage different state-of-the-art recommendation algorithms and further enhance the performance and interpretability of originals.
- With high interpretability, SLRC is able to discover distinct items' lifetimes, and hence distinguish different kinds of items such as durable goods and fast-moving consumer goods.

2 RELATED WORK

2.1 Repeat Consumption

Repeat consumption has been studied in various domains, including web revisitation [2, 9, 29, 45], repeated web search queries [40, 41], music listening [20], time-limited coupons [32], and predicting consumption rate [28]. All the work demonstrates that repeat consumption is common and important in various fields.

Early explorations of repeat consumption utilize marginal utility in economics to make prediction [42]. The *Law Of Diminishing Marginal Utility* [7] states that an item's marginal utility decreases as its quantity grows. However, this law may not always hold especially in item purchasing scenario. For example, users may buy fast-moving consumer goods again and again regularly although the accumulated quantities are growing.

There is a series of work focusing on some simplified tasks related to repeat consumption to better model intrinsic patterns. Some work predicts whether a purchase will be a repeat consumption according to item features in current time window [11]. And some other work predicts which consumed item user will prefer given the fact that current consumption is a repeat consumption [3, 5, 12]. Especially, Anderson et al. find *recency* is a key factor with a cache-based method, indicating that users tend to purchase items recently

consumed [3]. Different from these studies, we focus to build a unified and holistic model that can recommend both consumed items and novel items simultaneously, where consumed items are only properly selected at the right time.

Further, Benson et al. [4] presents a holistic model of sequential repeat consumption, which introduces time factor, distance factor (*recency*) and quality factor. Most recently, Cai et al. [8] and Du et al. [16] utilize revamped Hawkes Process to model sequential online interactive behaviors and recurrent user activities, respectively. Although these methods can make holistic recommendation, they either estimate temporal patterns over all items or move one more step to model item-specific effect of previously consumed items (self-excitation or mutual-excitation), but the degree of excitation simply decays globally with time. All these studies are lack in taking various temporal dynamics of repeat consumption into consideration. Moreover, without integrating collaborative filtering methods, they may fail to accurately capture users' personalized preference.

2.2 Collaborative Filtering and Point Process

There are other two lines of research related to our work: *Collaborative Filtering* and *Point Process*.

Collaborative Filtering analyzes relationships between users and interdependencies among products to identify new user-item associations, which is domain free and more accurate than content-based techniques [25]. Latent factor model is one of the primary methods of collaborative filtering, in which users and items are characterized on some specific number of factors inferred from the interaction patterns. A series of latent factor models have been proposed to improve performance of item recommendation [6, 10, 18, 19, 23, 35, 36]. The utilization of mutual relationships between users and items enables these methods to better capture users' preference and discover novel items users may be interested in.

Point Process is good at modeling sequential events localized in time [14]. There have been many applications of point process, including modeling user influence in social network [39, 46], predicting earthquakes [33], paper citation count [44] and user return times [38]. As a variant of temporal point process, Hawkes Process [17] explicitly models the effect of previous events (self-excitation). The triggering kernel of intensity function in Hawkes Process controls how excitation varies with time. There have been several studies leveraging recurrent neural network (RNN) to derive intensity or replace triggering kernel [15, 34, 43], but it is hard for RNN to capture very long dependencies of repeat consumption when spanning over a long period of time. The lack of interpretability is also the main weakness. Recent work utilizes temporal point process to predict high-level human actions such as eating and biking, which accounts for three time-sensitive characteristics [26]. Compared to only 8-10 actions in the work, there are millions of items in personalized recommendation task, in which case their model contains intractably numerous parameters and will fail to explore endless novel items without integrating collaborative filtering.

We further build a holistic model with combination of Collaborative Filtering and Hawkes Process to take full advantage of both techniques. The CF part of the model captures users' inherent preference and helps to explore novel items, while two temporal dynamics of repeat consumption modeled by Hawkes Process makes it possible to properly exploit consumed items, leading to a good balance of exploration and exploitation.

Table 1: Statistics of Datasets.

Dataset	#user ($\|\mathcal{U}\|$)	#item ($\|\mathcal{I}\|$)	#entry ($\sum_u \|S_u\|$)	repeat ratio	time span
Baby	25.4k	7.6k	238.7k	17.8%	2012.09 - 2014.05
Order	34.6k	86.2k	820.4k	14.0%	2016.12 - 2018.03
Recsys2017	113.8k	44.7k	1240.4k	21.2%	2016.11 - 2017.02
BrightKite	29.7k	105.2k	3591.9k	90.0%	2008.03 - 2010.10

3 EMPIRICAL STUDY ON REPEAT CONSUMPTION

In this section, we first describe the task we are tackling. Then we make several observations about repeat consumption on four real-world datasets, which serve as the foundation of our model.

3.1 Task Definition

Given a target time $\hat{t}$ and a user's consumption history before $\hat{t}$, our task is to predict top-k items the user may be interested in at target time. Note that the recommendation list here can contain both novel items and consumed items. Formally, let $\mathcal{U}$ be the set of all users. Each user $u \in \mathcal{U}$ has a consumption sequence $S_u = \{(i_1, t_1), (i_2, t_2), \cdots, (i_{N_u}, t_{N_u})\} \in \mathcal{S}$ with N_u interactions. Each interaction consists of an item $i_n \in \mathcal{I}$ and corresponding time $t_n \in \mathbb{R}^+$ $(0 \leq t_n \leq T)$, which means user u consumed item i_n at time t_n. T is the total time span observed. $\mathcal{S}$ represents all the consumption sequence in data. The sequence is sorted by time in an ascending order, i.e., $t_n \leq t_{n'}$ for any $n < n'$. For simplicity, we denote a user's consumption sequence up to time t as $S_t^u = \{(i', t') | (i', t') \in S_u \wedge t' < t\}$.

Then, we define *repeat consumption* as below:

Definition 3.1. The interaction (i, t) conducted by user u is a *repeat consumption* if and only if $\exists (i', t') \in S_t^u : \ i' = i \wedge t' < t$

3.2 Dataset Description

To illustrate critical temporal patterns of repeat consumption, we use four datasets in distinct scenarios. The datasets are described below and the consumption statistics after filtering are summarized in Table 1.

Baby. This dataset contains online purchasing information within baby category from an e-commerce retailer.

Order. This dataset contains mobile payment records in supermarkets and convenient stores. We have provided the anonymized dataset online[1].

Recsys2017. Recsys Challenge 2017[2] is a competition for job recommendation, where the item is job posting and the consumption is clicking. Then a repeat consumption is that a user clicks the same job posting again after previous click. We use click interactions in open training data (offline phase).

[1] https://github.com/THUwangcy/SLRC/tree/master/data
[2] https://www.recsyschallenge.com/2017/

BrightKite. BrightKite was a location-based social networking website, where users could check in to physical locations. The data is publicly available[3] [13].

Each record in these datasets contains user-item pair and corresponding timestamp. Users and items with less than 5 associated interactions are filtered out. Besides, we remove two types of users: (1) interactions take place within one day (likely to be attracted by promotions); (2) consumes too frequently (more likely a merchant but a normal user). These four datasets cover consumptions in different scenarios, including both online and offline consumptions, as well as different scene of shopping, job seeking and check-in. Besides the ratio of repeat consumption varies across datasets. For *BrightKite*, 90% consumptions are repeat consumption, while the repeat ratio is moderate in other datasets.

3.3 Temporal Dynamics of Repeat Consumption

To reveal the temporal dynamics of repeat consumption, we try to figure out how previous consumption triggers next purchasing for the same item and how repurchasing tendency changes with time. However, it is hard to observe actual self-exciting pattern, which is latent and we can only observe consumption behavior as a result of the latent factor. Here, we resort to study the inter-consumption gap, which means the time interval between a user's two adjacent consumptions for the same item. Inter-consumption gap is a good indicator of how the tendency for consuming an item again varies with time. For example, if there is a lot of people purchasing milk one day after previous consumption, it indicates the tendency of repurchasing milk is high after one day.

Through investigating overall and item-level distributions of inter-consumption gap, we make two important empirical observations about repeat consumption. First, consumptions for most items are likely to trigger next consumption for the same item in the short term, which we call *short-term effect*. Second, some items tend to be consumed again centrally after a period of time, which we call *life-time effect*. These two types of effects serve as the foundation of the proposed SLRC model in the next section.

3.3.1 Short-term effect. Figure 2 shows the overall distribution of inter-consumption gap in the datasets. Except for *Recsys2017*, all the distributions monotonically decrease with a heavy tail. The distribution in *Recsys2017* also diminishes quickly in the beginning, which means generally there are a lot of repeat consumptions happening shortly after previous consumption. Such short-term effect is very common in daily life. Figure 3 shows different items' inter-consumption gap distributions in *Baby* and *Order* datasets. Take milk powder in Figure 3(b) for example, one may occasionally buy a kind of milk powder online and appreciates it very much. As a result, she purchases the milk powder again recently. Even for items that are not fast-moving consumer goods, like baby care cream in Figure 3(a), it is possible to repurchase them in a short term just as a gift to others or for other reasons. As a result, the time-decaying short-term effect can be leveraged to basically model the tendency to repurchase.

3.3.2 Life-time effect. Apart from short-term effect, there is also life-time effect for repeatedly consuming items. The distribution in

[3]https://snap.stanford.edu/data/loc-brightkite.html

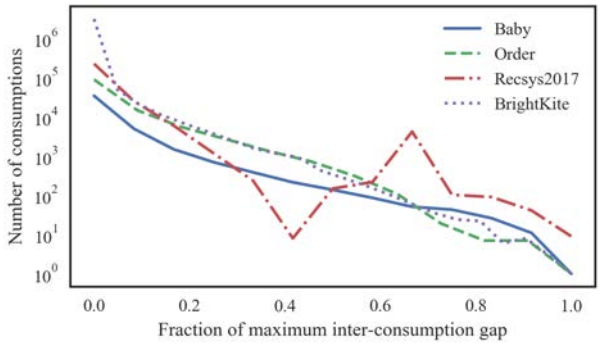


Figure 2: Distributions of inter-consumption gap. On the whole, most repeat consumptions happen shortly after the previous, while life-time effect is obvious in *Recsys2017* dataset.

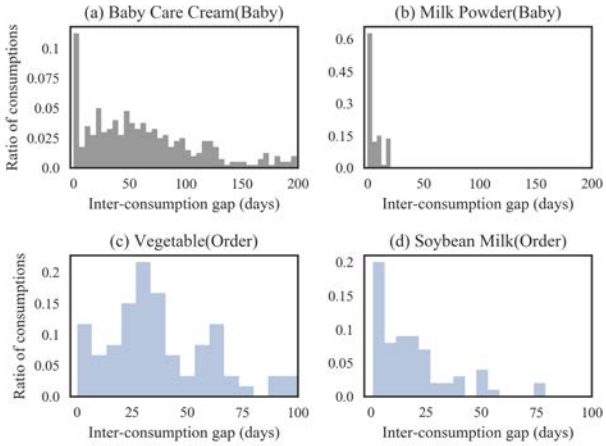


Figure 3: Inter-consumption gap distributions of different items. Various types of items inherently have distinct temporal dynamics of repeat consumption.

Recsys2017 dataset presents another spike after a long-period time, which means users may forget the information of previously clicked posts and click them again. Almost all items have their inherent lifetimes and demonstrates different time-varying patterns. For other datasets, the monotonically decreasing form on the whole can be a superimposed result of distinct distributions for various items. In Figure 3, we can see time-varying patterns differ from each other greatly. For example, in Figure 3(a)(c), baby care cream runs out in about 50 days and some families may go to supermarket monthly to resupply vegetables. When item's lifetime is going to run out, users will be more likely to purchase it again. Therefore, when taking life-time effect into consideration, we can make more accurate prediction at the right time (e.g. recommending durable goods consumed before after a long period when user tends to repurchase them).

In summary, two temporal patterns of repeat consumption: (1) *short-term effect* and (2) *life-time effect* are uncovered through investigation into inter-consumption gap distribution. More importantly, the importance of these two factors varies wildly across different items. Therefore, addressing item-specific temporal dynamics of repeat consumption will help to make more relevant recommendation.

4 SLRC MODEL

Based on the empirical observations above, we propose a holistic model with combination of collaborative filtering and Hawkes Process, which is named as **S**hort-Term and **L**ife-Time **R**epeat **C**onsumption (SLRC) model. Some preliminaries about temporal point process are introduced first. Then we detailedly describe model definition and parameter learning.

4.1 Preliminaries About Temporal Point Process

Formally, a temporal point process is a random process of which the realization consists of a list of discrete events localized in time, $\{t_n\}_{n\in\mathbb{N}}$ with the time $t_n \in \mathbb{R}^+$. In the scene of item consumption, $\{t_n\}_{n\in\mathbb{N}}$ represents a series of time that a user purchases some specific item. Given the history time of past events S_t, temporal point process introduces conditional intensity function $\lambda(t|S_t)$ representing a stochastic model for the time of the next event given all the times of previous events. For simplicity, we omit conditional sign and denote $\lambda(t)$ as conditional intensity function. Then, the probability for the occurrence of a new event given the history time S_t within a small time window $[t, t+dt)$ can be expressed as:

$$\lambda(t)dt = \mathbb{P}\{\text{event in } [t, t+dt) \mid S_t\} \,. \tag{1}$$

Besides, the conditional probability that an event does not happen since last observed time t_n is

$$U(t) = \exp\left(-\int_{t_n}^{t} \lambda(\tau)d\tau\right) \,. \tag{2}$$

Combining these two components, we have the conditional density function $f(t|S_t) = \lambda(t)U(t)$ that an event happens exactly at time t given all previous consumptions S_t [1].

The conditional intensity function $\lambda(t)$ takes various functional forms. A constant intensity leads to homogeneous Poisson Process, while non-homogeneous Poisson Process has time-varying intensity function but the events are still independent. Hawkes Process models excitations between events whose intensity function takes the form of:

$$\lambda(t) = \lambda_0 + \alpha \sum_{t_j < t} \gamma(t - t_j) \,, \tag{3}$$

where λ_0 represents the base intensity and every history event has an addictive self-exciting effect to current intensity, which varies with time and temporal characteristics are controlled by triggering kernel γ. α represents the degree of excitation.

4.2 Model Definition

Inspired by the intensity function in Hawkes Process (Eq. 3), we integrate collaborative filtering in base intensity and address both short-term and life-time effects in kernel function, leading to an intensity function in the following form:

$$\lambda^{u,i}(t) = \overbrace{\lambda_0^{u,i}}^{base} + \overbrace{\alpha_i \sum_{(t',i')\in S_t^u} I(i'=i)\gamma_i(t-t')}^{self-excitation} \,, \tag{4}$$

where the total intensity can be seen as a summation of base intensity $\lambda_0^{u,i}$ and the self-excitation[4] of all previous consumptions for item i.

4.2.1 Base intensity. Base intensity $\lambda_0^{u,i}$ models inherent personalized preference for different items. To accurately model the basic preference, we leverage collaborative filtering methods to derive the base intensity, such as Bayesian Personalized Ranking (BPRMF) and Neural Collaborative Filtering (NCF). Collaborative filtering methods model user's preference based on what she has consumed in the past and assume similar users like similar items, which helps to explore novel items. In the case of BPRMF, there is a K-dimensional latent factor for each user and item, p_u and q_i respectively. Then base intensity

$$\lambda_0^{u,i} = p_u^T q_i + b \,. \tag{5}$$

In the case of NCF, the base intensity is derived by a multi-layer neural network, which can be formulated as

$$\lambda_0^{u,i} = \phi_{out}(\phi_X(\cdots\phi_2(\phi_1(p_u^T, q_i^T))\cdots)) \,, \tag{6}$$

where ϕ_{out} and ϕ_x respectively denote the mapping function for the output layer and x-th neural collaborative filtering layer, and there are X neural CF layers in total.

We will show it is easy for SLRC model to leverage many state-of-the-art algorithms with joint parameters learning.

4.2.2 Self-excitation. In self-excitation part, $I(c)$ is an indicator function and returns 1 when c is true, otherwise 0. The triggering kernel $\gamma_i(\cdot)$ controls the temporal characteristics of self-excitation and varies with inter-consumption time interval $\Delta t = t - t'$. To explicitly address the two factors, we resolve $\gamma_i(\cdot)$ as a mixture distribution modeling short-term effect with Exponential distribution and life-time effect with Gaussian mixture distribution:

$$\gamma_i(\Delta t) = \overbrace{\pi_0^i E(\Delta t|\ 1/\beta^i)}^{short-term} + \overbrace{\sum_{z\in[1,Z]} \pi_z^i N(\Delta t|\ \mu_z^i, \sigma_z^i)}^{life-time} \,. \tag{7}$$

$E(x|\lambda)$ represents Exponential distribution with parameter λ, while $N(x|\mu,\sigma)$ is a Gaussian distribution with mean μ and standard deviation σ. The coefficient π_z^i satisfy the limit that $\sum_{z\in[0,Z]} \pi_z^i = 1$. From Eq. 7, we can see there are two factors in self-excitation:

Short-term effect. As shown in Section 3.3.1, short-term effect is really common and important for modeling repeat consumption. From Figure 2, the short-term effect diminishes quickly as time goes by, in which case an item-specific Exponential distribution is suitable for modeling it.

Life-time effect. Life-time effect also greatly influences user's repurchasing tendency, which often causes a raise of tendency after specific time intervals (e.g. 50 days after purchasing baby care cream when it is time to run out). As a result, it's natural to choose Gaussian mixture distribution to model life-time effect, for sometimes item's lifetime can be multimodal. But in our experiments,

[4]Here we only consider self-excitation (influence by the same item) because we mainly focus on repeat consumption. We leave the mutual-excitation caused by other items as future work.

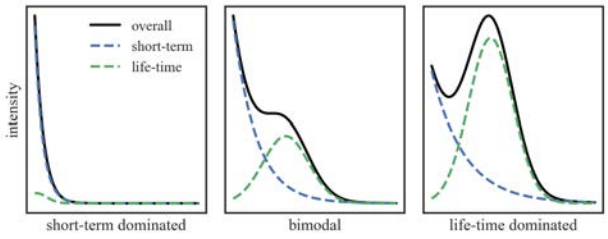


Figure 4: Illustration on the joint impact of various temporal dynamics on time-varying trends of overall intensity, characted by the triggering kernel.

we find a single Gaussian distribution is strong enough, with its mean standing for item's expected lifetime. Therefore we choose to set $Z = 1$ without loss of generality. If items' lifetimes demonstrate more complex patterns in datasets, larger Z can be chosen through cross-validation. When $Z = 1$, the triggering kernel $\gamma_i(\cdot)$ can be simplified as:

$$\gamma_i(\Delta t) = (1 - \pi^i)E(\Delta t|\ 1/\beta^i) + \pi^i N(\Delta t|\ \mu^i, \sigma^i),\ \pi^i \in [0, 1]\ . \quad (8)$$

All the parameters related to self-excitation $\Theta_s = \{\boldsymbol{\alpha}, \boldsymbol{\pi}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\sigma}\}$ are item-specific. We make the degree of excitation α^i a summation of global degree and item bias: $\alpha^i = \alpha + \alpha^i_b$, in which case global α can capture average degree of excitation. Figure 4 conceptually visualize how we separately model two temporal dynamics in self-excitation part, which shows different time-varying trends of overall intensity (black) can be represented as the summation of short-term effect (blue) and life-time effect (green).

As intensity $\lambda^{u,i}(t)$ indicates the tendency of consumption, the recommendation list can be derived by the followings: we calculate the intensity value of each candidate item (including novel items and consumed items) for user u at time t firstly. Then, all candidate items are ranked according to their intensity value. The top-k items in the ranked list will be recommended to users.

Different from previous work [5, 8, 31, 32, 43], we focus to build a holistic model that can recommend both novel items and consumed items. The integration of collaborative filtering methods helps to better balance recommendation of the two kinds of items. Besides, we address two temporal dynamics of repeat consumption with specifically designed item-specific triggering kernel, which is concise and powerful with high interpretability. There may be other methods to address these two factors, but what we want to do is preliminarily exploring how to incorporate item-specific temporal dynamics of repeat consumption into recommender systems. Therefore we leave other forms of intensity function as future work.

4.3 Parameter Learning

As all candidate items are ranked according to intensity value $\lambda^{u,i}(t)$, we can adopt pairwise ranking loss for optimization. For each consumption (u, i_n, t_n), a negative item i_n^- is randomly sampled from items user haven't consumed, i.e. $\mathcal{I} \backslash I_u$ (I_u is a set of unique items consumed by user u). The pairwise ranking loss is defined as follows:

$$\mathcal{L}(\Theta|\mathcal{S}) = -\sum_{n\in\mathcal{U}}\sum_{n=1}^{N_u} \log\sigma\left(\lambda^{u,i_n}(t_n) - \lambda^{u,i_n^-}(t_n)\right) + \Omega(\Theta_b)\ . \quad (9)$$

Remind that the base intensity λ_0^{u,i_n^u} can be derived by different methods. We denote Θ_b as parameters related to calculating base intensity (e.g. $\Theta_b = \{\boldsymbol{p}, \boldsymbol{q}, b\}$ for BPRMF). Θ_b may have various forms when our model leverages different algorithms. We add a $\mathcal{L}_2$ regularizer $\Omega(\Theta_b)$ to prevent the base intensity overfitting data. On the other hand, parameters related to self-excitation can be denoted as Θ_s ($\Theta = \Theta_b \cup \Theta_s$). Θ_b and Θ_s are learned together with the unified loss (Eq. 9).

Due to the success of Adam algorithm [22] in many recommendation models' parameter learning procedures, we use Adam as the learning algorithm to minimize the pairwise ranking loss. Note that such optimization method makes it easy and efficient for joint learning of parameters when leveraging different algorithms to derive base intensity.

5 EXPERIMENTS

5.1 Experimental Settings

5.1.1 Datasets and evaluation protocols. We use the same datasets described in Section 3.2, including consumptions in various scenarios. For each consumption sequence $S_u \in \mathcal{S}$, we leave the interactions at the latest time as test dataset and the interactions at the secondly latest time as validation dataset. All the remaining data are used for training. The repeat consumption ratio of each test dataset is similar with overall repeat ratio (as shown in Table 1). And for each test or validation case, we recommend top-k ($k = 5, 10$) items from a given candidate set containing ground-truth items. Considering it is too time-consuming to rank all items for some methods when dataset is large, we randomly sample a certain number of negative items besides ground-truth items to construct candidate set. The negative items are randomly sampled from all the items except for ground truth, which means the candidate set may contain both items that user has never consumed and previously consumed items (but not bought this time, which are also negative examples for this consumption). The candidate sets stay the same for all models' training and testing. As for the size of candidate set, we make it relevant to the number of unique items in the dataset (500, 5000, 2500, 5000 for *Baby*, *Order*, *Recsys2017*, *BrightKite* respectively).

To evaluate the quality of recommendation, we use Recall and *Normalized Discounted Cumulative Gain* (NDCG) as evaluation metrics. Recall measures how many ground-truth items are recommended, while NDCG concerns about whether ground-truth items are ranked higher than others by accounting for the position of hit. We calculate both metrics for each user in test or validation dataset and report the average score.

5.1.2 Baseline Methods. We compare SLRC to nine baselines in different aspects. The first three baselines are collaborative filtering models and do not take repeat consumption into consideration:

- **BPRMF**. This method apply a pairwise ranking loss to optimize Matrix Factorization (MF) model, which is competitive for item recommendation [36].

Table 2: Experimental results (higher is better). Bold face indicates the best result of each column in a particular metric. Best baseline in each metric is underlined. The relative improvement SLRC gains compared to originals is listed within parentheses. Value is significantly better than the strongest baseline ($p < 0.01$).**

Method	*Baby*				*Order*				*Recsys2017*				*BrightKite*			
	topk=5		topk=10		topk=5		topk=10		topk=5		topk=10		topk=5		topk=10	
	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG
BPRMF	0.3686	0.2786	0.4496	0.3058	0.2307	0.2168	0.2796	0.2371	0.5754	0.5191	0.5879	0.5231	0.7609	0.7143	0.7872	0.7229
Tensor	0.4587	0.3543	0.5708	0.3923	0.2214	0.1971	0.2874	0.2240	0.5561	0.4655	0.5758	0.4719	0.7816	0.7280	0.8106	0.7375
NCF	0.3600	0.2717	0.4450	0.3002	0.2241	0.1987	0.2834	0.2231	0.5926	0.5322	0.6080	0.5372	0.7431	0.6771	0.7663	0.6847
FPMC	0.3835	0.3034	0.4655	0.3311	0.2346	0.2188	0.2897	0.2414	0.6180	0.5838	0.6324	0.5885	0.7369	0.6927	0.7651	0.7018
UtilitySVD	0.3431	0.2677	0.4197	0.2933	0.2282	0.2214	0.2734	0.2401	0.5730	0.5053	0.5850	0.5092	0.7453	0.7151	0.7610	0.7202
HCM	0.3703	0.2736	0.4824	0.3118	0.1445	0.1345	0.1863	0.1518	0.5633	0.4277	0.6326	0.4505	0.8019	0.7540	0.8276	0.7590
TSR-Hawkes	0.3797	0.2998	0.4576	0.3260	0.2565	0.2544	0.2907	0.2689	0.6629	0.5888	0.6658	0.5898	0.7611	0.7447	0.7622	0.7451
Neu-Hawkes	0.4203	0.3298	0.5088	0.3550	0.2645	0.2577	0.3087	0.2804	0.7078	0.6356	0.7280	0.6405	0.7801	0.7511	0.8006	0.7520
LS-Hawkes	0.4304	0.3457	0.5356	0.3786	0.2604	0.2554	0.2978	0.2723	0.7103	0.6389	0.7292	0.6409	0.7803	0.7513	0.8098	0.7587
$\text{SLRC}_{\text{BPRMF}}$	0.4014 (+8.9%)	0.3217 (+15.5%)	0.4833 (+7.5%)	0.3492 (+14.2%)	**0.2790**** (+20.9%)	**0.2723**** (+25.6%)	**0.3301**** (+18.0%)	**0.2935**** (+23.8%)	0.7058 (+22.7%)	0.6337 (+22.1%)	0.7250 (+23.3%)	0.6399 (+22.3%)	**0.8083**** (+6.2%)	**0.7778**** (+8.9%)	**0.8286**** (+5.3%)	**0.7844**** (+8.5%)
$\text{SLRC}_{\text{Tensor}}$	**0.4894**** (+6.7%)	**0.3919**** (+10.6%)	**0.5991**** (+5.0%)	**0.4290**** (+9.3%)	0.2622 (+18.4%)	0.2572 (+30.5%)	0.3057 (+6.4%)	0.2752 (+22.9%)	0.6650 (+19.6%)	0.6034 (+29.6%)	0.6724 (+16.8%)	0.6058 (+28.4%)	0.7923 (+1.4%)	0.7658 (+5.2%)	0.8119 (+0.2%)	0.7722 (+4.7%)
SLRC_{NCF}	0.3924 (+9.0%)	0.3124 (+15.0%)	0.4709 (+5.8%)	0.3387 (+12.8%)	0.2751 (+22.8%)	0.2683 (+35.0%)	0.3219 (+13.6%)	0.2876 (+28.9%)	**0.7207**** (+21.6%)	**0.6450**** (+21.2%)	**0.7364**** (+21.1%)	**0.6501**** (+21.0%)	0.8044 (+8.3%)	0.7768 (+14.7%)	0.8219 (+7.3%)	0.7824 (+14.3%)

- **Tensor**. Tensor factorization is widely used in context-aware recommendation [21]. Here we split the continuous time into bins and handle a three dimensional tensor (user-item-time).
- **NCF**. This is a state-of-the-art collaborative filtering method that utilizes neural networks to capture complex relationship between users and items in latent space [18].

Another three baselines incorporate repeat consumption information and can make holistic item recommendation:

- **FPMC**. This method combines MF and factorized Markov Chains with a user-specific transition matrix, which utilizes the sequence information of each user [37].
- **UtilitySVD**. This method utilizes marginal utility to model repeat consumption, where users' preference for items will change according to consumed quantity [42].
- **HCM**. It is a holistic generative model for sequential repeated consumption, which takes time, quality and distance factor into consideration. We use the last two parts of the model to conduct top-k recommendation [4].

The last three are related to Hawkes Process with similar task:

- **TSR-Hawkes**. This model aims at Time-Sensitive Recommendation with low-rank Hawkes Process, in which core parameters are assumed to have low-rank structure. The triggering kernel decays globally with time [16].
- **Neu-Hawkes**. Mei et al. [34] propose a neural Hawkes process model that uses recurrent layer to derive intensity according to events history. A novel continuous-time LSTM architecture is utilized where each memory cell exponentially decays toward some steady-state value.
- **LS-Hawkes**. The model captures long- and short-term dependency between actions with two different Hawkes Process, while both of the triggering kernels simply decay in a global way [8].

The first three baselines will further be leveraged to derive the base intensity $\lambda_0^{u,i}$ in our SLRC model. We denote the unified model as $\text{SLRC}_{\text{method}}$, such as $\text{SLRC}_{\text{BPRMF}}$.

As for other work about repeat consumption [3, 5, 11, 12] referred to in Section 2, they are used either to classify repeat consumption or to predict only items consumed before. Neither of them focuses to recommend both consumed items and novel items simultaneously. So they cannot be taken as baseline methods.

5.1.3 Implement details. We implement SLRC model in *TensorFlow* and the code has been open source[5]. We tune hyperparameters based on NDCG@10 in the validation dataset and report the results in the test dataset. All models are trained with Adam optimizer until converge with a maximum of 200 epochs. Each experiment is repeated 5 times with different random seed and the average result is reported.

For fair comparison, the batch size is fixed to 128 and the latent dimension for all models is 100. As for parameter initialization, global excitation α and parameters in mixture distribution $\boldsymbol{\beta}, \boldsymbol{\sigma}$ are initialized with 1; item bias of excitation degree α_b^i is all initialized with 0. $\boldsymbol{\pi}$ is normally initialized with 0.5 mean; $\boldsymbol{\mu}$ is also normally initialized and the mean is global average inter-consumption time interval. Other parameters are normally initialized with 0 mean. All normal initializers have 0.01 standard deviation.

5.2 Performance

Table 2 shows the performance of all baselines and the improved methods when incorporating SLRC model.

First, the performances of different baselines vary across datasets. Tensor, Neu-Hawkes, LS-Hawkes, HCM performs the best among all baselines in *Baby*, *Order*, *Recsys2017*, *BrightKite*, respectively, which shows capacities of different models in various scenarios. Tensor performs quite well in *Baby*, which may be the result of long time span and time sensitivity in this dataset. As state-of-the-art models based on Hawkes Process, Neu-Hawkes and LS-Hawkes performs robustly in all the datasets, and become the most competitive baselines in *Order* and *Recsys2017*. As for HCM, it achieves the best performance among baselines in *Brightkite* because of its high repeat consumption ratio (90%). Explicitly predicting whether current consumption will be repeat consumption in HCM greatly narrow the range of candidate items to consider. But in other datasets with

[5] https://github.com/THUwangcy/SLRC

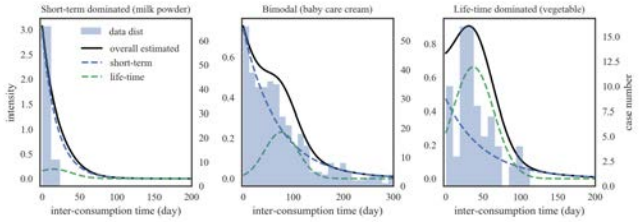


Figure 5: Some examples to validate the parametric modeling assumptions. For different types of items, overall intensity (black) closely approximates real distributions of inter-consumption gap (bar). Moreover, Exponential (blue) and Gaussian (green) distribution fit short-term and life-time effects well, respectively.

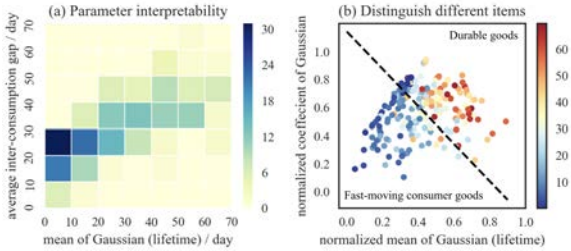


Figure 6: Correlation between item's inter-consumption gap and the learned lifetime parameter μ^i in Gaussian distribution (left), where color represents the number of items. And joint distribution of parameter μ^i and π^i (right), where red and blue stand for longer and shorter inter-consumption gap in average, respectively.

lower repeat consumption ratio (e.g. *Order*), due to the lack of collaborative filtering information, mainly modeling repeat consumption results in bad performance.

Second, although collaborative filtering and Hawkes Process methods gain encouraging results in different datasets, SLRC performs consistently better than other baselines in all datasets (the best combination with collaborative filtering method may be different in distinct datasets). This shows the proposed model is capable of utilizing both the strengths of collaborative filtering and temporal dynamics of repeat consumption in Hawkes Process, leading to great overall performance improvement consequently. Aside from the performance improvement over models about repeat consumption (i.e. FPMC, UtilitySVD, HCM), SLRC outperforms methods based on Hawkes Process without concern about item-specific temporal dynamics of repeat consumption (i.e. TSR-Hawkes, Neu-Hawkes and LS-Hawkes), which indicates the importance of modeling both short-term and life-time effects in kernel function. The integration of collaborative filtering methods also plays an important role, which estimates users' inherent preference for items more accurately and helps to better recommend novel items that meet users' interests.

Third, the experimental results demonstrate that the incorporation of SLRC leads to a big gap of improvement (up to 35.0%) compared to the original collaborative filtering baselines. This shows the information of repeat consumption and its item-specific temporal dynamics are critical to recommender systems. Notice that the improvement for traditional method BPRMF is extremely large in most datasets. SLRC introduces history sequence information and temporal factors into the model, so BPRMF may attentively model inherent preference more accurately. For other baselines that take more information into consideration themselves, SLRC also brings consistent performance gain on all datasets. Besides, the encouraging results on all the datasets in different scenarios show the scalability and flexibility of our model.

5.3 Parametric Assumptions Validating

Here we want to validate whether the learned parameters have interpretable meaning as we assume when designing the model.

First, we study how well the learned intensity function fits real data. Figure 5 gives some examples from different datasets. We draw both the distribution of inter-consumption gap in real data and how estimated intensity drifting with time[6]. The results show that the overall intensity (black) closely approximates real distribution of inter-consumption gap. Moreover, the *short-term effect* (blue) and *life-time effect* (green) capture the temporal characteristics of repeat consumption well, respectively. Take vegetable as an example (the right figure), we find there are numerous repurchasing behavior for vegetables around 30 days after previous consumption in Section 3.3. As expected, the Gaussian distribution for vegetable learned in SLRC shows around 30 days mean ($\mu^i = 36$ days) and the importance of *life-time effect* is high ($\pi^i = 0.65$), indicating *life-time effect* is in dominated position.

Next, remind that the mean (μ^i) and the coefficient (π^i) of life-time effect in Eq. 7 are item-specific. μ^i can be seen as item's expected lifetime and π^i is the importance of life-time effect. Figure 6 (a)[7] shows the correlation between item's inter-consumption gap in average and corresponding parameter μ^i. Parameters are trained in *Baby* dataset with $\text{SLRC}_{\text{BPRMF}}$ model. Other combinations in other datasets yield similar results. Note that the information of inter-consumption gap is not introduced in SLRC explicitly, but μ^i is reasonably in direct proportion to average inter-consumption gap (Pearson Correlation Coefficient: 0.7468), showing that items that are often consumed again after a long period of time (long lifetime) get larger μ^i just as assumed.

Further, we show the joint distribution of μ^i and π^i in Figure 6 (b), which demonstrates that items with higher average inter-consumption gap (red in color) almost have larger μ^i and π^i. The dashed line in figure obviously divides items into two kinds: fast-moving consumer goods (below) and durable goods (above). Some representative items above the dashed line are toys, baby walkers, beds and so on, while fast- moving consumer goods like paper diaper and milk powder are mostly under the dashed line. Another convincing and interesting observation is that most feeding bottles appear above the dashed line while nipples only appear below the dashed line, for the reason that users often change nipples but seldom change feeding bottles.

[6]Here we assume previous consumption happens at the origin of coordinates and show the temporal trend of self-excitation.

[7]We discard items that are never repeatedly consumed. And items with extremely high or low number of average inter-consumption gap are also removed.

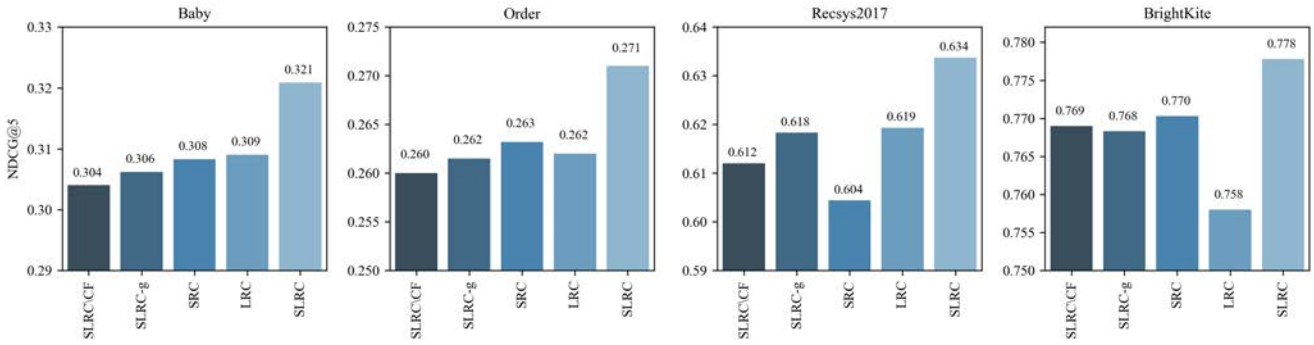


Figure 7: Performance comparison between SLRC and methods with no consideration for CF or variety of temporal dynamics. SLRC performs the best again various variations. All improvements are statically significant ($p < 0.05$).

Table 3: Difference comparison between variants of SLRC.

Variants	CF	Item-specific	Life-time	Short-term
SLRC\CF		✓	✓	✓
SLRC-g	✓		✓	✓
SRC	✓	✓		✓
LRC	✓	✓	✓	
SLRC	✓	✓	✓	✓

To sum up, SLRC gains remarkable performance improvement and the estimated parameters can further explain various temporal patterns of repeat consumption across different items perfectly.

6 FURTHER ANALYSIS

6.1 Ablation Study

To verify the effect of CF part and the two item-specific temporal dynamics (short-term and life-time effects) addressed in the model, we compare SLRC to a series of baselines without consideration for variety of temporal characteristics of repeat consumption.

- **SLRC\CF**. This model removes the CF part of SLRC and uses an item-specific parameter to estimate the base intensity.
- **SLRC-g**. This model does not concern the variety across different items. Parameters related to self-excitation are not item-specific (parameters in $\Theta_s = \{\boldsymbol{\alpha}, \boldsymbol{\pi}, \boldsymbol{\beta}, \boldsymbol{\mu}, \boldsymbol{\sigma}\}$ are global values).
- **SRC**. This model only consider short-term effect without life-time effect, where the triggering kernel in intensity function is an item-specific Exponential distribution.
- **LRC**. Similar with SRC, LRC only takes life-time effect into consideration, where an item-specific Gaussian distribution acts as the triggering kernel in intensity function.

Table 3 intuitively shows the difference between SLRC and its variants. Combinations with different baselines yield similar results, so here we only show results when combined with BPRMF (i.e. base intensity is calculated by BPRMF for SRC, LRC, SLRC-g). The results shown in Figure 7 indicate that although variants of SLRC achieve consistently better performance than BPRMF, none of them can outperform the full SLRC model.

Without the integration of CF, SLRC\CF leads to loss of performance, which demonstrates that the CF part of SLRC plays an important role for holistic recommendation task. As for SLRC-g, it neglects inherent properties (like lifetime) of distinct items. The short-term and life-time effects estimated over all items can be inaccurate when evaluating some specific items. Therefore the difference in temporal dynamics across items are critical to model repeat consumption behavior.

On the other hand, SRC and LRC do not take both temporal factors into consideration and hence get worse results, which demonstrates that the lack of variety of temporal factors can lead to lower predictive ability (e.g. continuous recommendation for mobiles shortly after purchasing one and mobiles could not be recommended when it is the time to change a new one). It is noteworthy that the performances of SRC and LRC vary across datasets. The results are similar in *Baby* and *Order* for the reason that both short-term and life-time effects are common in item purchasing scenario. But LRC is better than SRC in *Recsys2017*, which may be the result of obvious life-time effect in this dataset as shown in Section 3.3. The short-term effect is weaker compared with other datasets because users seldom click a job posting again in a short time. While in *BrightKite*, SRC outperforms LRC greatly. It is also reasonable because users often revisit hotels they appreciate recently but seldom go to the one visited long time ago. The short-term effect is much stronger than life-time effect in such scenario.

Further, dynamically modeling short-term and life-time effects for different items, our SLRC model is able to utilize strengths of CF and capture complex patterns of repeat consumption, achieving the best performance in all the datasets.

6.2 Discussion on Exploration and Exploitation

Besides the great performance improvement, we want to figure out how well our model balances the recommendation of novel items and consumed items. The self-excitation addressed in the model should not result in recommending more consumed items under all circumstances. Therefore, we construct a series of subsets (the same size) from the test dataset with different ratios of repeat consumption to simulate exploration dominated and exploitation dominated scenarios. This can be done by randomly sampling a certain ratio (namely, repeat ratio r) of cases from repeat consumption set and filling the rest with cases randomly sampled from the other part

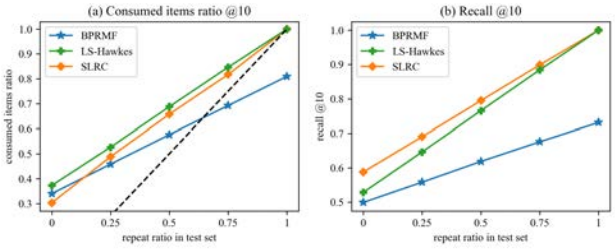


Figure 8: Results on test subsets with different ratios of repeat consumption case. In the left figure, the dashed line shows expected trend of consumed items ratio. SLRC shows a better balance of exploration and exploitation, and gains better result all the time.

containing no repeat consumptions. When repeat ratio r is small, exploration cases are in dominant position (if $r = 0$, all cases in the subset are consumptions for novel items), otherwise exploitation cases dominate the subset.

Figure 8 shows consumed items ratio and Recall@10 on subsets with different repeat ratio r. Consumed items ratio means the proportion of test cases whose corresponding recommendation list contains consumed items. Ideally, model should recommend less consumed items when repeat ratio r is small (i.e. users tend to seek novel items) but to retrieve more consumed items when r is large (i.e. users want to purchase items consumed before). In the left figure, the dashed line shows expected trend of consumed items ratio. We conduct top-10 recommendation in *Recsys2017* dataset and use BPRMF as the base method. Several observations from these results are worth highlighting:

- All the models recommend more consumed items when repeat ratio gets larger (i.e. users express more needs for consumed items) and yield better performance.
- SLRC recommends less items when repeat ratio is small, while recommends more consumed items otherwise, leading to a better balance of exploration and exploitation compared with BPRMF and LS-Hawkes.
- SLRC achieves better results under all circumstances. The improvement gap compared to BPRMF gets larger as repeat ratio increases. And only when all test cases are repeat consumption, LS-Hawkes can achieve similar performance with SLRC.

The main conclusion is that addressing both short-term and lifetime effects helps SLRC to well balance exploration and exploitation. SLRC will not recommend extra consumed items when users do not need. While in the cases when users express demands for consumed items, our model properly recommends more consumed items. Notice that the performance of LS-Hawkes gets closer to SLRC's when repeat ratio increases, the reason may be that LS-Hawkes wrongly gives higher scores to some consumed items without considering various temporal dynamics and CF information, so novel items cannot be recommended for some exploration cases. As a result, only when there are all repeat consumption cases, LS-Hawkes can achieve similar performance with SLRC.

In summary, the results demonstrate that SLRC clearly captures users' need for consumed items, achieving a good balance of exploration and exploitation. Furthermore, in real-world scenarios where repeat consumption takes a moderate part, SLRC model performs significantly better than other methods.

7 CONCLUSION AND FUTURE WORK

Repeat consumption comes to be increasingly important recently. It is necessary to model repeat consumption behavior so that recommender system can recommend items user consumed before at proper time. In this work, we focus to build a holistic model that can recommend novel items and consumed items simultaneously. Two factors are revealed for modeling various time-varying patterns of repeat consumption: *short-term effect* and *life-time effect*. To address this variety, we propose a novel model, **S**hort-Term and **L**ife-Time **R**epeat **C**onsumption (SLRC), which explicitly takes the two factors into consideration with combination of two techniques: Collaborative Filtering (CF) and Hawkes Process. According to extensive experiments, significant performance improvement is achieved in four real-world datasets compared to state-of-the-art baselines. Besides, with high interpretability, SLRC is able to discover distinct items' lifetimes. And different types of items such as fast-moving consumer goods and durable goods can even be distinguished by the learned parameters, showing the model can perfectly captures various item-specific dynamics of repeat consumption. Generally, our model is able to be leveraged as a framework that utilize repeat consumption information to enhance existing recommender systems based on collaborative filtering and hence achieve better recommendation results in real-work applications.

In the future, we will turn to category level and consider mutual-excitation of similar items. Intuitively, a previous consumption not only has influence on the item itself, but also influences purchase tendency of similar items. The item-specific modeling method in SLRC may suffer from data sparsity greatly. Further, simultaneous consumption has some inherent relationship with repeat consumption. How to combine these two together is an interesting but seldom explored problem.

ACKNOWLEDGMENTS

This work is supported by Natural Science Foundation of China (Grant No. 61672311, 61532011) and The National Key Research and Development Program of China (2018YFC0831900). We would like to thank Maarten de Rijke and Diane Kelly for their valuable comments of the work.

REFERENCES

[1] Odd O. Aalen, ÃŸrnulf Borgan, and HÃ¢kon K. Gjessing. 2008. *Survival and Event History Analysis.* Springer New York. 457–491 pages.
[2] Eytan Adar, Jaime Teevan, and Susan T. Dumais. 2008. Large scale analysis of web revisitation patterns. In *Sigchi Conference on Human Factors in Computing Systems.* 1197–1206.
[3] Ashton Anderson, Ravi Kumar, Andrew Tomkins, and Sergei Vassilvitskii. 2014. The dynamics of repeat consumption. In *Proceedings of the 23rd international conference on World wide web.* ACM, 419–430.
[4] Austin R. Benson, Ravi Kumar, and Andrew Tomkins. 2016. Modeling User Consumption Sequences. In *International Conference on World Wide Web.* 519–529.
[5] Rahul Bhagat, Srevatsan Muralidharan, Alex Lobzhanidze, and Shankar Vishwanath. 2018. Buy It Again: Modeling Repeat Purchase Recommendations. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* ACM, 62–70.
[6] Preeti Bhargava, Thomas Phan, Jiayu Zhou, and Juhan Lee. 2015. Who, what, when, and where: Multi-dimensional collaborative recommendations using tensor

factorization on sparse user-generated data. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 130–140.

[7] Stanley L Brue, Campbell R Mcconnell, and McGrawHill. 2011. Economics: principles, problems and policies. *Economics Principles Problems and Policies* (2011).

[8] Renqin Cai, Xueying Bai, Zhenrui Wang, Yuling Shi, Parikshit Sondhi, and Hongning Wang. 2018. Modeling Sequential Online Interactive Behaviors with Temporal Point Process. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 873–882.

[9] Lara D Catledge and James E Pitkow. 1995. Characterizing Browsing Strategies in the World-Wide Web. In *International World Wide Web Conference*. 1065âĂŞ1073.

[10] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural Attentional Rating Regression with Review-level Explanations. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. 1583–1592.

[11] Jun Chen, Chaokun Wang, and Jianmin Wang. 2015. Will You" Reconsume" the Near Past? Fast Prediction on Short-Term Reconsumption Behaviors.. In *AAAI*. 23–29.

[12] Jun Chen, Chaokun Wang, Jianmin Wang, and S Yu Philip. 2016. Recommendation for repeat consumption from user implicit feedback. *IEEE Transactions on Knowledge and Data Engineering* 28, 11 (2016), 3083–3097.

[13] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. 2011. Friendship and mobility:user movement in location-based social networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, Ca, Usa, August*. 1082–1090.

[14] David Roxbee Cox and Valerie Isham. 1980. Point processes. *Monographs on Statistics and Applied Probability* 65, 432 (1980), 47–98.

[15] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. 2016. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1555–1564.

[16] Nan Du, Yichen Wang, Niao He, and Le Song. 2015. Time-sensitive recommendation from recurrent user activities. In *International Conference on Neural Information Processing Systems*. 3492–3500.

[17] J. Durbin and G. S. Watson. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* 58, 1 (1971), 83–90.

[18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[19] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee, 263–272.

[20] Komal Kapoor, Karthik Subbian, Jaideep Srivastava, and Paul Schrater. 2015. Just in time recommendations: Modeling the dynamics of boredom in activity streams. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. ACM, 233–242.

[21] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. 2010. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 79–86.

[22] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *Computer Science* (2014).

[23] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.

[24] Yehuda Koren and Robert Bell. 2015. Advances in collaborative filtering. In *Recommender systems handbook*. Springer, 77–118.

[25] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[26] Takeshi Kurashima, Tim Althoff, and Jure Leskovec. 2018. Modeling Interdependent and Periodic Real-World Action Sequences. *arXiv preprint arXiv:1802.09148* (2018).

[27] Guokun Lai, Wei Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. (2018).

[28] Moshe Lichman and Padhraic Smyth. 2018. Prediction of Sparse User-Item Consumption Rates with Zero-Inflated Poisson Regression. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 719–728.

[29] Jie Liu, Chun Yu, Wenchang Xu, and Yuanchun Shi. 2012. Clustering web pages to facilitate revisitation on mobile devices. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*. 249–252.

[30] Xinyue Liu, Yuanfang Song, Charu Aggarwal, Yao Zhang, and Xiangnan Kong. 2017. BiCycle: Item Recommendation with Life Cycles. In *IEEE International Conference on Data Mining*. 297–306.

[31] Dixin Luo, Hongteng Xu, Yi Zhen, Xia Ning, Hongyuan Zha, Xiaokang Yang, and Wenjun Zhang. 2015. Multi-task multi-dimensional hawkes processes for modeling event sequences. In *International Conference on Artificial Intelligence*. 3685–3691.

[32] Emaad Manzoor and Leman Akoglu. 2017. RUSH!: Targeted Time-limited Coupons via Purchase Forecasts. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1923–1931.

[33] D Marsan and O LenglinÃĪ. 2008. Extending earthquakes' reach through cascading. *Science* 319, 5866 (2008), 1076–9.

[34] Hongyuan Mei and Jason M Eisner. 2017. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*. 6754–6764.

[35] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.

[36] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[37] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 811–820.

[38] Mingxuan Sun, Mingxuan Sun, Tao Ye, and Tao Ye. 2014. A hazard based approach to user return time prediction. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1719–1728.

[39] Yusuke Tanaka, Takeshi Kurashima, Yasuhiro Fujiwara, Tomoharu Iwata, and Hiroshi Sawada. 2016. Inferring Latent Triggers of Purchases with Consideration of Social Effects and Media Advertisements. In *ACM International Conference on Web Search and Data Mining*. 543–552.

[40] Jaime Teevan, Eytan Adar, Rosie Jones, and Michael Potts. 2006. History repeats itself: repeat queries in Yahoo's logs. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 703–704.

[41] Sarah K. Tyler and Jaime Teevan. 2009. Large scale query log analysis of re-finding. In *ACM International Conference on Web Search and Data Mining*. 191–200.

[42] Jian Wang and Yi Zhang. 2011. Utilizing marginal net utility for recommendation in e-commerce. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 1003–1012.

[43] Shuai Xiao, Junchi Yan, Stephen M. Chu, Xiaokang Yang, and Hongyuan Zha. 2017. Modeling The Intensity Function Of Point Process Via Recurrent Neural Networks. (2017).

[44] Shuai Xiao, Junchi Yan, Changsheng Li, Bo Jin, Xiangfeng Wang, Xiaokang Yang, Stephen M Chu, and Hongyuan Zha. 2016. On Modeling and Predicting Individual Paper Citation Count over Time.. In *IJCAI*. 2676–2682.

[45] Haimo Zhang and Shengdong Zhao. 2011. Measuring web page revisitation in tabbed browsing. In *Sigchi Conference on Human Factors in Computing Systems*. 1831–1834.

[46] Ke Zhou, Hongyuan Zha, and Le Song. 2013. Learning triggering kernels for multi-dimensional hawkes processes. In *International Conference on International Conference on Machine Learning*. III–1301.

Understanding the Evolution of Mobile App Ecosystems: A Longitudinal Measurement Study of Google Play

Haoyu Wang
Beijing University of Posts and Telecommunications
Beijing, China
haoyuwang@bupt.edu.cn

Hao Li
OrangeAPK, Inc.
Beijing, China
lihao_0823@yeah.net

Yao Guo
MOE Key Lab of HCST, School of EECS, Peking University
Beijing, China
yaoguo@pku.edu.cn

ABSTRACT

The continuing expansion of mobile app ecosystems has attracted lots of efforts from the research community. However, although a large number of research studies have focused on analyzing the corpus of mobile apps and app markets, little is known at a comprehensive level on the evolution of mobile app ecosystems. Because the mobile app ecosystem is continuously evolving over time, understanding the dynamics of app ecosystems could provide unique insights that cannot be achieved through studying a single static snapshot. In this paper, we seek to shed light on the dynamics of mobile app ecosystems. Based on 5.3 million app records (with both app metadata and apks) collected from three snapshots of Google Play over more than three years, we conduct the first study on the evolution of app ecosystems from different aspects. Our results suggest that although the overall ecosystem shows promising progress in regard of app popularity, user ratings, permission usage and privacy policy declaration, there still exists a considerable number of unsolved issues including malicious apps, update issues, third-party tracking threats, improper app promotion behaviors, and spamming/malicious developers. Our study shows that understanding the evolution of mobile app ecosystems can help developers make better decision on developing and releasing apps, provide insights for app markets to identifying misbehaviors, and help mobile users to choose desired apps.

CCS CONCEPTS

• **Information systems** → **Web mining**; • **Security and privacy** → **Mobile and wireless security**; • **Networks** → **Mobile networks**; • **Software and its engineering** → *Software libraries and repositories.*

KEYWORDS

App ecosystem; Android; Google Play; evolution

ACM Reference Format:
Haoyu Wang, Hao Li, and Yao Guo. 2019. Understanding the Evolution of Mobile App Ecosystems: A Longitudinal Measurement Study of Google Play. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313611

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313611

1 INTRODUCTION

It has been over 10 years since the first Android-based smartphone was announced in 2008 [18, 27]. Android has evolved significantly in the past decade, and has become the most popular mobile operating system with 85.9% market share as of 2017 [55]. Google reported that there were over 2 billion monthly active devices using Android as of May 2017 [48]. Millions of mobile apps with myriad features have become the key to boost the evolution of smartphones and Android. Google Play, the official Android app market, was first introduced in 2008 [72]. Since its launch, (Android) mobile apps have impacted our society at an astounding pace and scale. It is now one of the largest app stores in the world, with over 2.5 million apps available for download in October 2018[1]. The total number of app downloads in Google Play has surpassed 200 billion by 2017 [70].

Billions of mobile users, millions of mobile apps and app developers, thousands of devices manufacturers, and hundreds of app markets have formed a symbiotic relationship, which we refer to as the ***mobile app ecosystem***. The global app economy is also poised to explode, which is estimated to worth 6.3 trillion dollars by 2021 [45]. As a result, the continuing evolution of mobile app ecosystems has increased the complexity and myth of the whole ecosystem, which itself needs considerable efforts to understand its characteristics and working mechanism.

The mobile app ecosystem has attracted a lot of efforts from the research community. A large number of studies have focused on analyzing the mobile apps and app markets, including exploring the characteristics of mobile apps [11, 32, 35, 43, 59], understanding and comparing app markets [7, 14, 60, 67, 69, 70], security and privacy analysis of mobile apps [10, 25, 26, 41], understanding app developers and mobile users [16, 22, 33, 40, 74], etc.

However, little is known at a comprehensive level on the **evolution of mobile app ecosystems**. Although several previous work have performed large-scale measurement studies on the app ecosystem and analyzed millions of apps [60, 70], they only studied the ecosystem based on a static snapshot collected in a given time slot. For example, PlayDrone [60] performed the characterization of 1.1 million Google Play apps crawled in 2013, and Wang *et al.* [69] studied the ecosystem from the perspective of app developers on over 1.2 million Google Play apps and 320,000 developers collected in 2015. To the best of our knowledge, no previous work have studied the evolution of app ecosystems comprehensively at a large scale.

Because the Android ecosystem is evolving over time, studying the dynamics of the app ecosystem could gain unique insights

[1]It is reported that the number of apps in Google Play has surpassed 3 million in 2017. However, Google has been removing a significant number of apps from time to time.

that cannot be achieved by studying static app market snapshots. From app developers' perspective, understanding the evolution of app ecosystems could help them make better decision on developing and releasing apps, e.g., realizing which app categories are most competitive and the trend, and understanding the impact that developer actions will likely have on the success of their apps. From the view of app markets, studying the evolution of the app ecosystem could help them with monitoring and further improving the app market, as the number of apps keeps increasing. Previous studies suggest that malicious apps and fraudulent developers are prevalent in app markets [60, 70], thus lessons learned from a longitudinal analysis can provide insights into identifying fraudulent app promotion and malicious behaviors. Besides, a longitudinal measurement study of the ecosystem is a valuable indicator of understanding the real threats and challenges. From mobile users' perspective, understanding the dynamics of app markets could help them make better decision when choosing desired apps with similar functionality, e.g., avoiding the apps that have not been updated for years, which may contain vulnerabilities and have compatibility issues.

This Study. In this paper, we make the first effort towards understanding the evolution of mobile app ecosystems. Specifically, we seek to shed light on the evolution of Google Play, the official and the most popular Android app market. To this end, we have collected three snapshots of Google Play, which were crawled in March 2014, March 2015 and September 2017, respectively (Section 3). This dataset includes 5.3 million Android app entities in total, with all the metadata and apks. We first use this dataset to provide a high-level characterization of all the apps and analyze the evolving trends, including the distribution of free and paid apps, the evolution of app downloads and app rating, the ranking and competitiveness of app categories, the evolution of permission usage and privacy policy declaration, and the presence of third-party tracking and advertising libraries (Section 4). Next, for the 743,530 long-lasting apps (appeared in all three snapshots) in our dataset, we further investigate their growth rate, app updates, and permission changes (Section 5). Third, we provide an in-depth analysis of malicious and deceptive behaviors across the three snapshots, discussing the presence of malicious apps and aggressive app promotion behaviors (Section 6). At last, we analyze the evolution from the developers' perspective. We have classified the developers into different groups based on the number of apps they released and analyzed their evolution, and further pinpoint the spamming and malicious developers (Section 7). Among many interesting results and observations, the following are the most prominent:

- The app monetization scheme evolves during the evolution of the mobile app ecosystem. The number of paid apps has decreased significantly in Google Play, while mobile advertising is getting more popular.
- Although the popularity and overall quality of apps published in Google Play are getting better, the overall ecosystem shows a typical Pareto effect all the time, i.e., dominated by a small number of apps and developers.
- There exists a trend that the percentage of apps that request sensitive permissions are on the decline. Google's regulation on privacy policy has greatly improved the privacy protection of mobile users, e.g., the percentage of apps declaring privacy policies in 2017 is four times of the number in 2014.
- Over 61% of the long-lasting apps have not received any updates for three years, which indicates that these apps cannot utilize new features introduced by new Android versions, and may have compatibility issues and vulnerabilities.
- The proportion of potentially malicious apps in Google Play is decreasing. However, spamming apps that use misleading references to other apps are always prevalent in Google Play, many using more sophistic methods to insert keywords.
- Although Google Play removes spamming developers and malicious developers regularly, the number of developers with misbehaviors is nonetheless increasing.

To the best of our knowledge, this is the first comprehensive longitudinal study of the mobile app ecosystem at scale. Our dataset is by far one of the largest Google Play app repositories in the research community, with all the metadata and apks. We believe that our efforts and the revealed insights can benefit the research community, as well as various stakeholders of the mobile app ecosystem.

2 RELATED WORK

2.1 Mobile App Ecosystem Analysis

A number of research efforts have focused on analyzing the mobile app ecosystem [7, 28, 29, 31, 34, 38, 46, 60, 69, 70]. They usually first collect a large number of apps from Google Play or alternative markets, and then measure the app ecosystem from various dimensions. We have summarized the most related work on app ecosystem analysis, as shown in Table 1. Two representative studies are AndroidZoo [38] and the work of Wang *et al.* [70]. AndroZoo focused on crawling a large-scale dataset of APKs, which has enabled a number of studies focusing on malware analysis and app repository mining [31]. Wang *et al.* [70] performed a large-scale comparative study that covers more than 6 million Android apps downloaded from 16 alternative markets and Google Play to understand the catalog similarity across app stores and malicious behaviors. Although these studies have measured the mobile app ecosystem in large scale, they only captured a single snapshot of the ecosystem and did not consider the longitudinal evolution.

2.2 App Evolution Analysis

One line of research is analyzing the app evolution. Potharaju *et al.* [49] performed a longitudinal study of 160K apps, however, they only studied the evolution of metadata (e.g., price and downloads) and only covered 10% of apps in Google Play. Calciati *et al.* [13] and Taylor *et al.* [56, 57] analyzed the evolution of apps' permission requests, and found that apps tend to request more permissions in their evolution. Taylor and Martinovic [56] studied the evolution of financial apps from permission usage and vulnerabilities. Gao *et al.* [23] analyzed the vulnerability evolution of Android apps based on a set of Android app lineages in large scale, and found that apps do not get safer as they get updated. Wang *et al.* [67] found that a large portion of apps in Google Play are removed from time to time. They presented a large-scale study of 791,138 removed Google Play apps to identify potential reasons for app removal. Although these studies have analyzed app evolution from specific

Table 1: Related work on mobile app ecosystem analysis.

Paper	Market	# Apps (year)	Dimensions
PlayDrone [60]	Google Play	1,107,476 (2013)	Similar app, Library usage, Authentication schemes
Petsas et al. [46]	Five alternative markets	316,143 (2013)	Download patterns, popularity trends
Wang et al. [69]	Google Play	1,501,555 (2015)	Characterizing App Developers
Ali et al. [7]	Apple App Store, Google Play	80,000 pairs (2015)	App-store attributes Comparison
Ishii et al. [31]	Google Play, Alternative Markets	4,761,283 (2016)	Malware, Security Management
Want et al. [70]	Google Play, Chinese Markets	6,277,247 (2017)	Catalog Similarity, App Clones, Malware
AndroidZoo++ [38]	Google Play, Alternative Markets	Over 7 Million (2017)	App Repository

perspectives, these studies lack of a comprehensive understanding on the evolution of the app ecosystem.

2.3 Mobile App Analysis

A large number of studies focus on mobile app analysis, including security and privacy analysis [10, 25, 30, 39, 41, 42, 65, 66, 68], app repackaging detection [15, 63], app quality analysis [37, 50, 54, 58], third-party library detection [12, 44, 53, 62, 64] and mobile ad network analysis [17, 20], etc. Most of these studies focus on one specific issue and lack of the measurement study of the app ecosystem, although similar approaches could be used to understand the evolution of specific issues in the mobile app ecosystem.

3 STUDY DESIGN

3.1 Data Collection

We have created three snapshots of Google Play, which were crawled in 2014, 2015 and 2017, respectively. Note that we use the term **snapshot** to refer to the entire state of the market, i.e., it contains meta-information of (almost) all the apps and the corresponding apks. This dataset is representative enough to study the evolution of app ecosystem, as the timeline in our study period covers three major Android system updates (version 5.0, 6.0 and 7.0).

First Snapshot. It was created in March 2014. Our crawling strategy started from top 500 Google Play apps in each category (considered as seeds), and 12,500 apps that belong to 25 general categories in total. Then, we use a breadth-first-search approach to crawl (1) "Similar Apps" shown on the app web pages recommended by Google Play and (2) other apps released by the same developer by retrieving the developers' Google Play web pages. Note that we downloaded all the apk files of free apps through the Google Play API. Furthermore, We have taken efforts to crawl all the app metadata (e.g., app names, descriptions, version names, developer names, user ratings, app installs, the privacy policy links, etc). We have crawled over 1.5 million apps, which represent almost all the apps that can be crawled from Google Play at that time [60].

Second Snapshot. In March 2015 (1 year after the first snapshot), we repeated the same process as described above to crawl Google Play apps, except that we take the previous 1.5 million crawled apps as our searching seeds. It could ensure that we check all the 1.5 million apps crawled in 2014. Overall, we are able to collect more than 1.6 million Android apps in this snapshot.

Third Snapshot. We repeated the same process in September 2017 (2.5 years after the second snapshot), and we take the 1.6 million apps crawled in the second snapshot as searching seeds. At last, we have crawled 2.1 million apps in total.

Table 2: Summary of our dataset.

	# Apps	# Free	#Paid	Installs	# Developers
Google Play 2014	1,522,730	1,292,064	230,666	55.1B	413,933
Google Play 2015	1,650,895	1,408,593	242,302	90.2B	398,855
Google Play 2017	2,144,733	2,012,904	131,829	193.5B	541,105

We launched crawlers in parallel via Aliyun Cloud servers that were located in North America. In order to avoid potential regional/language bias, we instrumented our crawlers in the default mode that supports English. This is the largest longitudinal Android app dataset in our research community that has collected the detailed information of both apks and app metadata.

3.2 Dataset Characteristics

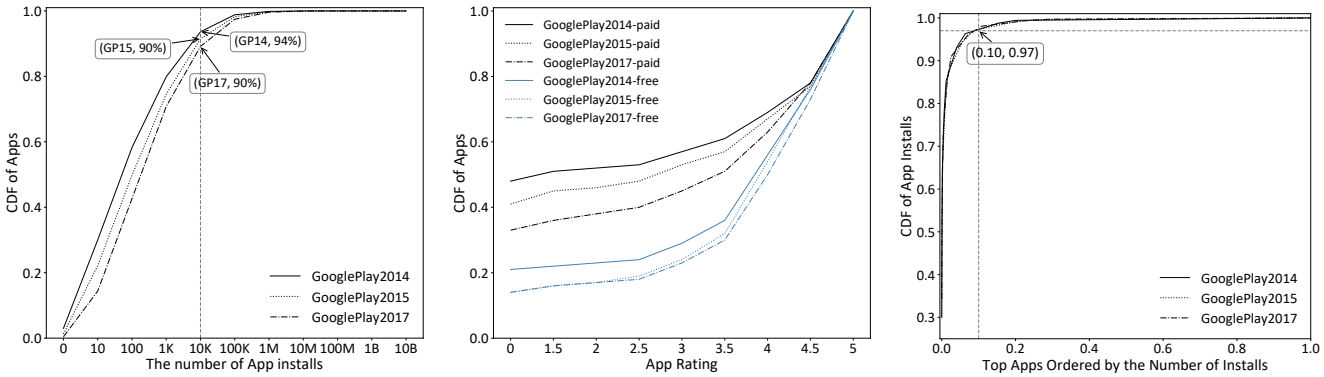
Table 2 reports the number of apps we harvested in the three snapshots. We could observe some general trends here: (1) From 2014 to 2017, the number of free apps in Google Play increases greatly, from roughly 1.3 million to the 2 million mark. In contrast, the number of paid apps decreases rapidly since 2015. The number of paid apps in 2017 is roughly half of the number in 2015. (2) The overall number of app installs in Google Play grows steadily. The number of app installs in 2015 is twice the number of 2014, while in 2017 the number has doubled compared to the 2015 snapshot. (3) The number of app developers in Google Play has not seen significant increase, which remains stable in the range of 400K to 500K. It is interesting to see that the number of app developers decrease in the 2015 snapshot compared to 2014.

4 OVERALL TREND OF APP EVOLUTION

4.1 Free vs. Paid Apps

It is a trend that the number of paid apps has decreased significantly in Google Play. The percentage of paid apps dropped by more than 60% (from 15.2% in 2014 to 6.1% in 2017). As to app installs, the proportion of paid apps is nearly negligible. Free apps occupy more than 99.7% of total app installs across snapshots.

There is a sharp decrease in the number of available paid apps from 2015 to 2017. To further investigate the reason, we analyzed the catalog similarity of paid apps between the three snapshots, and get two main observations: (1) More than 95% of the paid apps in 2017 are originated from 2015, while only 6,252 new paid apps have emerged in the two-year period. More than 99.5% of new apps crawled in the 2017 (1.28 million) snapshot are free apps. (2) Only 52% of paid apps (125,577) in 2015 remained as paid apps in 2017. For the remaining 116,725 apps, 6,989 of them (6%) became free apps (with the same app name and package name) in 2017, and roughly 94% of them had been removed/discontinued/renamed, so that we cannot locate them by searching the names in Google Play.



(a) The distribution of app installs. (b) The distribution of app ratings. (c) A few apps account for most of the downloads.

Figure 1: The evolution of app installs and app ratings.

Implication. One possible reason is that mobile ads and in-purchase become the most popular monetization schemes in Go play apps (cf. Section 4.7). As a result, new developers should ret their app releasing and monetization strategies.

4.2 App Installs

We further investigate the distribution of app installs across three snapshots. Figure 1(a) reports the cumulative distribu of app installs, which suggests that ***the popularity and qua of apps published in Google Play is getting better***. Over of apps have less than 10K installs by 2014, while the numb reduced to 89% by 2017. The average number of app installs raised from 36,183 (2014) to 90,209 (2017).

Then we characterize the app popularity distribution for all apps ordered by the number of installs, as illustrated in Figure Our results highlight a typical **Pareto effect**, with top 0.1% of a in Google Play represent roughly 60% of total installs, while the 1% of apps account for 80% of total installs. This observation is sistent across all the three snapshots. It suggests that although overall market shows a promising trend in regard of app popula the market is still dominated by a small percentage of apps.

4.3 App Ratings

We then investigate the distribution of app ratings in Google P across the snapshots, as shown in Figure 1(b). In general, the distribution of app ratings for free apps is better than paid apps. The app ratings are getting better for both free and paid apps, i.e., more apps are getting higher user ratings. This trend is more obvious for paid apps. For the 2017 dataset, over 55% of paid apps received app ratings higher than 3-star, while the number in the 2014 dataset is 43%. For the free apps, the distribution of app ratings in the 2015 and 2017 dataset are almost the same, which are slightly better than in 2014.

4.4 App Categories

For the 2014 and 2015 snapshots, there are generally 25 categories, as listed in Figure 2. While, in 2017, 8 new categories were introduced, including "Beauty" and "Dating". As the distribution of apps and developers is fairly diverse for different categories, we examined the differences between each category in order to understand

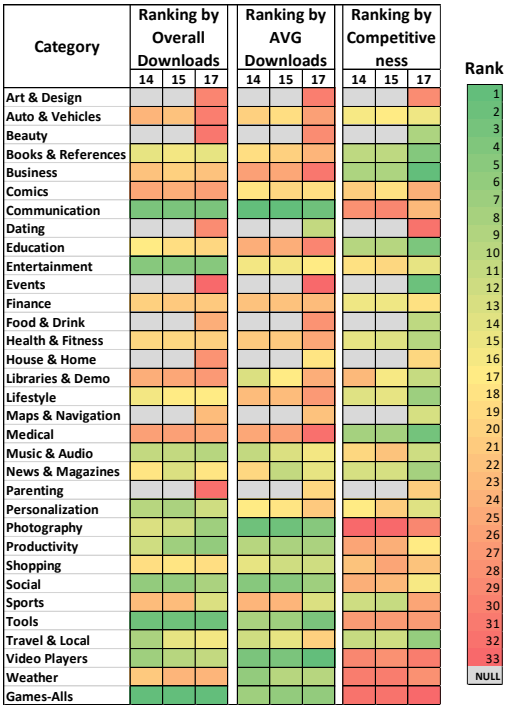


Figure 2: Distribution of app categories.

which categories are more popular and more competitive over time. This is also important to understand the trend of popular apps.

To this end, we compare the app categories based on (1) the total number of app installs, (2) the average number of app installs, and (3) the percentage of popular apps in each category, which can be regarded as an indicator of category competitiveness. Note that we regarded the apps that belong to the top 1% of the most downloaded apps as **popular apps**. The result is shown in Figure 2.

Category Ranking by Total Installs. Besides the "Games" Category, "Tools" received the most number of app installs (e.g., 25 billion till 2017) across the three years. The ranking of most categories does not vary significantly during the period. Five out of the 33 categories have received higher ranking from 2014 to 2017, for

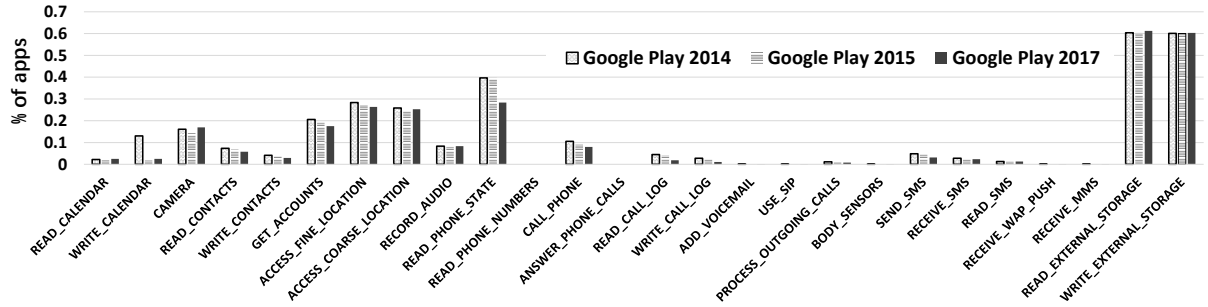


Figure 3: Evolution of requested sensitive permissions.

example, the ranking of category "Sports" went up to 11 from 21. Six categories (e.g., "Travel & Local") have received lower rankings.

Category Ranking by Average Downloads. We then compare app categories based on the average app installs, which is an indicator of app quality and user adoption. As shown in Figure 2, the general trend is stable, except for a few categories. Although category "Entertainment" receives quite high number of overall installs (rank-4 across the period), it only ranks No. 14 on average app downloads. Although the "Weather" category only ranks 19 to 22 in regard of total installs, the average app installs in this category are quite high, which ranks 5 to 8 during the evolution.

Ranking by Competitiveness. We then rank app categories based on the percentage of popular apps in each category. "Business" and "Medical" are the most competitive categories, while the total installs and average installs of these two categories rank low among the 33 categories. "Video Player" and "Communication" categories are the least competitive categories, while they have the most number of average app installs. This result could offer insights to new app developers on their developing and releasing strategies.

4.5 Permission Usage

We then investigate how sensitive permission usage by Google Play apps has evolved over time. Permission requests protect sensitive information available from a device, while Google requires developers to use sensitive permissions only when it is necessary for the functioning of the app. We seek to investigate how the permission usage in the whole ecosystem has evolved. For each app, we extract its requested permissions. Note that we only focus on the so-called *dangerous permissions* as listed by Android [8], i.e., the 26 system permissions used to guard accesses to sensitive data.

Figure 3 shows the overall result. The trend is that the number of apps requesting sensitive permissions are on the decline. More than 13% of apps requested the WRITE_CALENDER permission in 2014, while the numbers in 2015 and 2017 are only 1.7% and 2.5%, respectively. As another example, roughly 40% of apps requested the READ_PHONE_STATE permission in 2014 and 2015, while the number went down to 28% in 2017. Note that we will discuss the evolution of per-app permission request in Section 5.

One possible reason leading to this result is that Google Play requires all developers to provide a privacy policy since 2017 when an app requests sensitive permissions, which may in turn help apps respect users' privacy (i.e., eliminate unnecessary permissions).

4.6 Privacy Policies

Since 2017, Google Play requires developers to provide a valid privacy policy when an app requests or handles sensitive information [47]. It is reported that Google Play will remove apps with no privacy policies [1]. Thus, we can compare the developers' practices on privacy policy declaration along with the timeline. Note that the dataset we crawled in 2014 does not contain the privacy policy URL of the corresponding app, thus we only compare the apps in the 2015 and 2017 snapshots.

Our empirical analysis suggests that Google Play apps may declare privacy policy in either the app description or in the field of privacy policy shown on web pages. Thus, for the apps that request sensitive permissions, we check whether it declares privacy policies in its app description by (1) searching keywords "Privacy Policy" or (2) inspecting the privacy policy field shown on app pages. Note that we only consider whether the privacy policy exists, i.e., it should declare a privacy policy as long as it requests sensitive information, as required by Google Play. We did not check whether the privacy policy is consistent with its actual behaviors, which is another interesting topic that is out of the scope of this paper [73].

Figure 4 shows the analysis result for each sensitive permission. For the apps that request a specific permission, we calculate the percentage of apps with privacy policies. The results show that, in 2015, less than 20% of apps (11% on average) declared privacy policies even if they have requested dangerous permissions. For example, 216,160 apps requested the "CAMERA" permission in 2015, while only 31,814 of them (14.7%) declared privacy policies. This scenario has been greatly improved in 2017. For most permissions, more than half of the apps that used these permissions have declared privacy policies. The percentage is generally four times higher than that in 2015. For example, more than 62.8% of apps that use the "CAMERA" permission now have declared privacy policies.

Implication. This result suggests that the guidelines of Google Play on privacy policies have greatly improved the privacy protection of mobile users. Nevertheless, we should notice that a considerable number of apps still have not declared privacy policies, even if they request access to users' sensitive data, which indicates that they may violate Google Play's policy but are not removed.

4.7 Mobile Advertisement Services

Third-party libraries form an integral part of the mobile ecosystem, helping ease app development and enable features such as

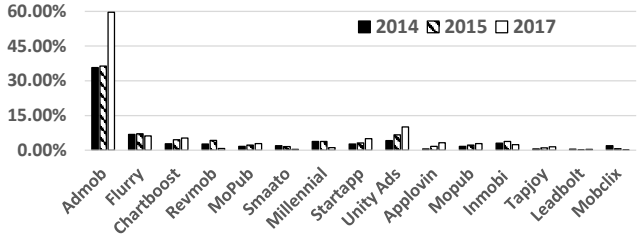


Figure 5: Top 15 mobile advertisement libraries.

analytics, social networking, and app monetization through advertisements. However, previous work suggested that third-party tracking services, especially the mobile advertisement services, may bring security and privacy issues to mobile users.

One important question here we want to explore is *whether more Google Play apps have embedded advertisement services during our investigation period.* Thus, we first need to identify third-party libraries in the apps we crawled. Here we extend a state-of-the-art clustering-based approach [43] to the millions of apps we crawled, and build a complete list of third-party libraries used in Google Play apps. The key idea is that, code that belongs to third-party libraries would be grouped in a large cluster. Then we manually labelled ad libraries based on several existing classifications, including AppBrain library classification [9], PrivacyGrade classification [6], and Common Library classification [36]. At last, we have manually labeled over 100 ad libraries.

Overall, 43.8% of apps in 2014 used at least one ad library, while over 68% of apps in 2017 embedded ad libraries. It means that more and more Google Play apps tend to use advertisement libraries, which is consistent with our findings in Section 4.1 that the app monetization schemes have changed over time, i.e., the number of paid apps is decreasing and most free apps rely on mobile advertisement to make money. Figure 5 further shows the top 15 mobile advertisement libraries used in Google Play apps. It is no doubt that Admob is dominating Google Play all the time. Over 35% of apps in the 2014 snapshot use the Admob library, while the percentage has raised to roughly 60% in our 2017 dataset.

5 LONG-LASTING APPS

In this section, we focus on the life cycle of apps. We define "**long-lasting apps**" as those available in all three Google Play snapshots, and we define "**removed**" apps as those apps that were removed from Google Play in either the 2015 or 2017 snapshot. In our dataset, 743,530 apps are flagged as "long-lasting apps", i.e., we have crawled three snapshots of them. It indicates that over 50% of the apps released in 2014 have been removed during the four-year period. We will next investigate them from different perspectives, including *app growth rate*, *app updates* and *permission evolution.*

5.1 Growth Rate

It is important to study the growth rate of app installs for the long-lasting apps, which could offer insights on how to develop successful apps. Thus, we first classify the number of app installs into 10 ranges, e.g., $[1, 10)$, $[10, 100)$, $[100, 1K)$, etc. Then, we analyze how many of them keep staying in the original ranges, and how many of them have evolved to higher install ranges.

Figure 6(a) presents the results. Overall, more than 57% of apps did not move ahead to a higher install ranges, especially for the apps with installs higher than 10K. For the apps with less than 1K downloads in the 2014 snapshot, roughly 40% of them have moved into a higher range during our study period. Interestingly, very few unpopular apps (roughly 0.04%) would become popular (with more than 1 million installs) later, as almost all of them would remain unpopular and with relative low downloads. This result suggests that the *initial status of the app would determine whether this app could be successful to a great extent.*

5.2 App Updates

We then study the statistics on app updates, which form a critical part of the app life cycle and would greatly affect app quality. It is important to understand how apps update over time, especially as several new Android versions was introduced during the period of our snapshots. For example, Lollipop (version 5.0) was released in October 2014, Marshmallow (version 6.0) was introduced in May 2015, and Nougat (version 7.0) was announced in August 2016.

Percentage of Updated Apps. For each long-lasting app, we have analyzed its three snapshots by comparing the hash values of the apks and the extracted version names and version code. To our surprise, **roughly 61% of apps have not received any update within our observation period**. Only 11.44% of them have been updated at least twice[2]. This means that more than 60% apps did not release new versions for almost four years, even if the Android system has evolved over several versions, and introduced several key features such as runtime permission enforcement [2].

App Updates across Categories. We then investigate whether there is a correlation between app updates and app categories, as

[2]Our sampling only covered 3 snapshots, while the apps may update more than twice.

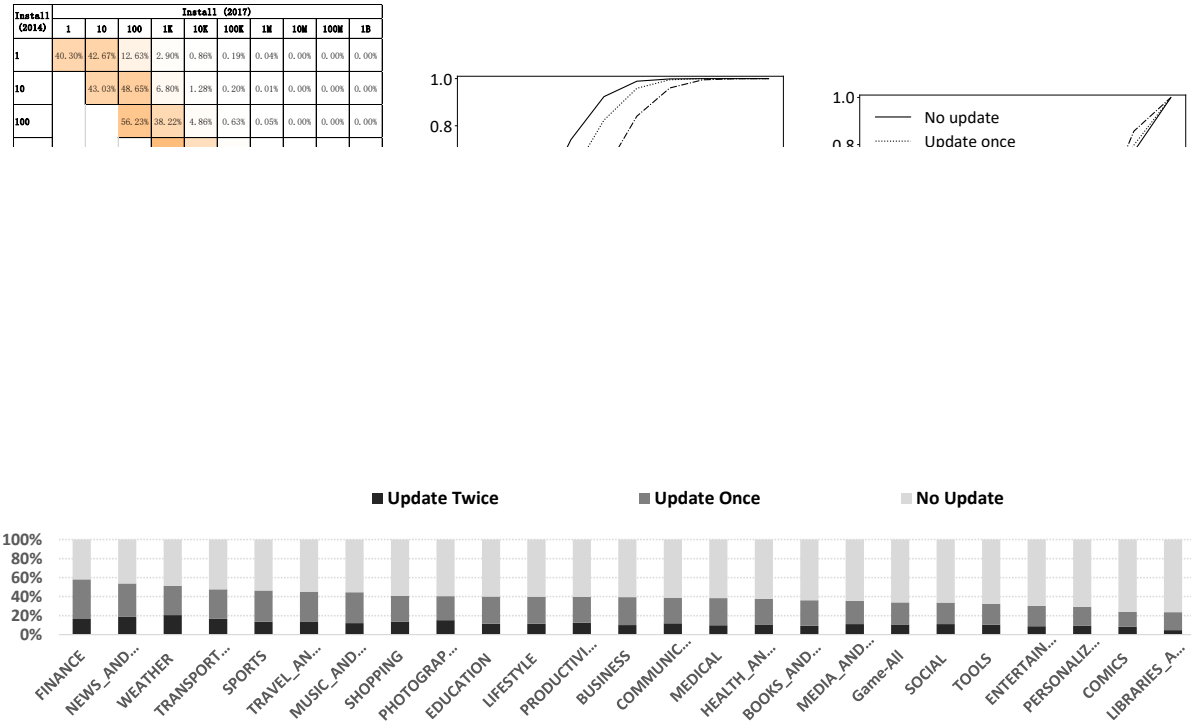


Figure 7: The distribution of app updates across categories.

shown in Figure 7. Apps in "Finance" and "News and Magazine" categories have received the highest number of updates, with roughly 60% of them have been updated at least once. In contrast, categories "Comics" and "Libraries and Demo" are the most inactive, with roughly only a quarter of apps in these categories have been updated at least once. One possible reason is that apps in categories like "Libraries and Demo" contain mostly static content that do not need to be updated, while apps in categories like "Finance" contain mostly dynamic content and they need to be updated frequently in order to provide new features. Nevertheless, the percentage of apps that received frequent updates is relatively low.

App Updates vs. App Popularity We then investigate whether popular apps received more updates than the unpopular ones, or whether app updates have positive effects on app installs. Figure 6(b) shows the distribution of app installs at different "update frequency levels". It is obvious that there is a positive correlation between app updates and app popularity. Over 40% of apps in the "update twice" group have received more than 10K installs, while the percentage in the "no update" group is only 7%. We further analyze the user ratings of long-lasting apps, in order to investigate whether users are more satisfied with apps that received more updates. As shown in Figure 6(c), there is no strong correlation between app rating and app update, although the distribution of app ratings for apps updated twice is slightly better.

5.3 Permission Evolution

We also analyzed permission changes at the per-app level to understand how many permissions each individual app had added or removed during the evolution. For a given app, its permission usage between snapshots falls into the four categories: (1) new permissions, (2) removed permissions, (3) both new and removed permissions, (4) permissions unchanged. Between the three snapshots we created, there are two intervals where we could study permission evolution.

Table 3: The distribution of permission evolution patterns.

		15-17			
		new	removed	both	unchanged
14-15	new	1.03%	1.76%	1.50%	1.70%
	removed	4.12%	1.17%	1.66%	0.97%
	both	0.50%	0.53%	1.16%	0.36%
	unchanged	21.21%	38.95%	20.49%	2.90%

Changes in Permission Usage. We first investigate how many long-lasting apps changed their permission usage during the evolution. Note that our previous observation suggests that over 60% of apps did not received any updates, thus we eliminate these apps when studying the changes in permission usage. As shown in the Table 3, each cell has two corresponding values, which represent the permission evolution states from 2014 to 2015 and from 2015 to 2017, respectively. Roughly 2.9% of the apps have no permission changes during the period, while only 0.4% of apps have requested more permissions all the time. Note that over 80% of apps do not have permission changes from 2014 to 2015, while roughly 95% of apps changed their permissions from 2015 to 2017. Over 42% of apps removed sensitive permissions and did not introduce new permissions from the year 2015 to 2017.

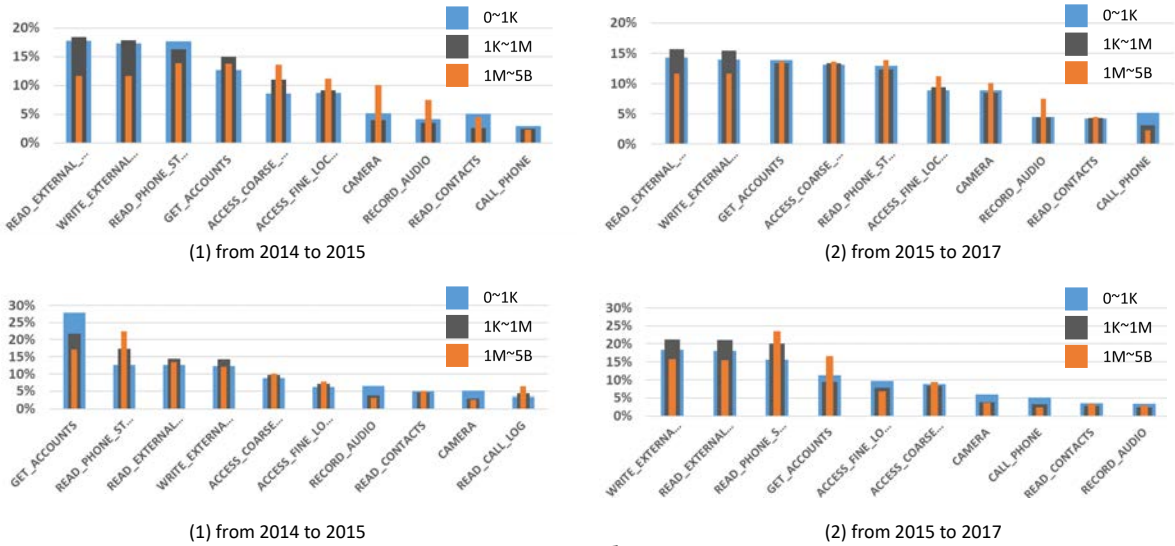


Figure 9: Removed permissions.

Changed Permissions We then analyze the permissions that were added to apps when they were updated, to understand the potential privacy risks. Figure 8 shows the breakdown of the most commonly added permissions for apps with different download levels. The top 2 permissions added are "READ_EXTERNAL_ STORAGE" and "WRITE_EXTERNAL_STORAGE", especially for the apps with low installs. Many apps added the "READ_EXTERNAL_STORAGE" permission while they have already declared the corresponding write permission. What is apparently unclear to these developers is that, apps granted with the write permission have implicitly obtained the read access as well, so there is no need to ask for the read permission. Besides, popular apps tend to request more sensitive permissions such as "CAMERA" and "RECORD_AUDIO" during the evolution than the apps with lower app installs. Figure 9 shows which permissions were removed the most. "GET_ACCOUNTS" is the most commonly removed permission from 2014 to 2015, and the external storage permissions are the most frequently added and removed permissions from 2015 to 2017.

6 THE EVOLUTION OF SECURITY RISKS

In this section, we study the evolution of security risks and developing misbehaviors, i.e., the presence of malicious apps, and the spamming apps using ranking fraud techniques.

6.1 Malicious Apps

As a measure to eliminate malicious apps, Google Play has adopted some vetting process to keep malware from entering the market, e.g., the Bouncer service [52] and machine-learning based approaches [19]. However, it is reported that malicious apps are recurrently found in Google Play [4, 5, 21, 61], especially for new malware variants [3]. As Google Play is removing malware from time to time, we want to investigate whether Google Play is moving towards higher security levels during its evolution.

Table 4: Distribution of potential malware over time.

Year	AV-rank (# apps, % apps)		
	≥ 1	≥ 10	≥ 20
Google Play 2014	261,480 (17.17%)	44,664 (2.93%)	9,324 (0.61%)
Google Play 2015	250,484 (15.17%)	37,111 (2.25%)	8,716 (0.53%)
Google Play 2017	179,184 (8.35%)	19,459 (0.91%)	3,854 (0.18%)

To this end, we upload all the apps we collected from the three snapshots to VirusTotal, a popular online malware analysis service that aggregates more than 60 anti-virus engines. Note that, some anti-virus engines on VirusTotal may not always report reliable results [10, 71]. To reduce false positives, we use "AV-Score" to represent how many engines flag an app as malware. We empirically select the threshold of "$AV - Score = 10$" as a robust choice for identifying malware, as suggested by previous studies [10, 70].

Distribution of Malicious Apps We first analyze the presence of malicious apps in the three snapshots, as reported in Table 4. We are glad to report that *the proportion of potentially malicious apps in Google Play is decreasing year by year.* Roughly 50% of the apps in snapshot 2014 are flagged at least by one anti-virus engine, while the percentage for snapshot 2017 is significantly lower (8.35%). According to the threshold of "$AV - Score = 10$", around 0.91% of the apps in snapshot 2017 are labeled as malware, while the percentage in snapshot 2014 is three times higher.

Evolution of Malware Families We also analyzed the malware families that are prevalent on Google Play. We resort to AVClass [51], a widely used malware labeling tool to obtain the family name of each identified malware, and Figure 10 shows the results. The top malware families are relatively stable across the three years. "revmob" and "airpush" are the most popular malware families, which are well-known aggressive adware.

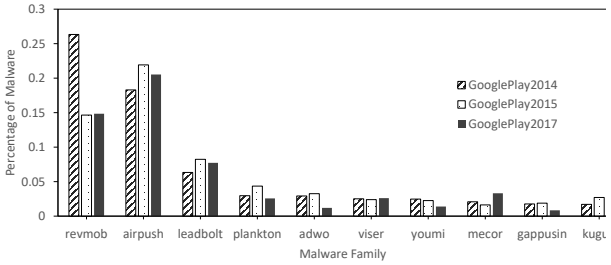


Figure 10: The distribution of the top 10 malware famili

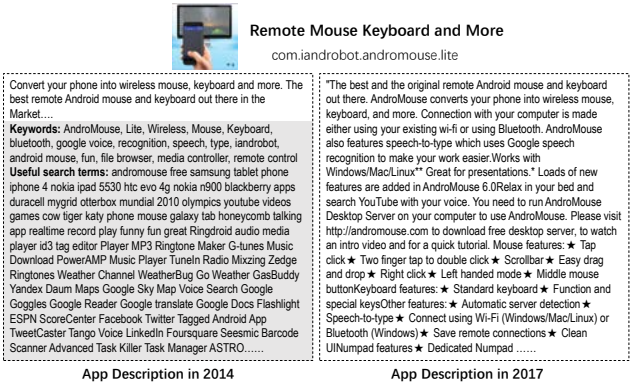


Figure 11: An example of misleading references to popular apps (left). The description has been improved later (right).

6.2 Spamming Apps

Google Play has updated its developer policy on metadata to make app listing accurate and relevant in October 2016 [24]. Google asked the developers to avoid user testimonials, excessive details, misleading references to other apps and repetitive, excessive or irrelevant keywords. The motivation is that some developers would like to insert irrelevant keywords (e.g., the names of popular apps and popular searching words) in their app descriptions, so that their apps would appear popular (highly ranked) in the search results, which is a common spamming technique and even used by some app store optimization (ASO) providers.

Approach. Therefore, it is important to investigate whether the situation of spamming apps has been improved with the evolution. To this end, we resort to the most straightforward strategy to identify spamming apps with irrelevant descriptions, which is to identify apps that insert names of popular apps in their descriptions. This strategy proposed by Seneviratne *et al.* [54] is of interest to spamming developers. As a result, we first collect the top 100 popular app names, then we count the number of popular app names mentioned in the descriptions of all the apps we harvested. Note that popular app names could be common words, such as "Video Player", "Weather" and "Music", thus we also excluded 10 such common words from this study.

Result. Table 5 summarizes the result. In general, during the evolution, more apps embed these popular app names in their descriptions. Since mobile apps may communicate with Facebook and Twitter to provide social networking functionalities, having those keywords in the app descriptions does not directly mean that these apps are spamming apps. By manually inspecting some cases, we believe that incorporating more than 10 popular app names in the description are mostly spamming apps.

The result suggests that the 2017 snapshot contains more possible spamming apps, while the aggressive level is not as serious as that in snapshot 2014 and 2015. For example, there are 56 apps in snapshot 2015 that have listed more than 40 keywords in their app descriptions (the most aggressive one has 80 keywords), while the most aggressive one in 2017 only has 25 keywords. We further explore the evolution of these spamming app candidates. About 62% of these apps identified in 2014 and 2015 snapshots were removed in 2017, and for the remaining apps, all of their app descriptions have been improved. For example, as shown in Figure 11, the app "com.iandrobot.andromouse.lite" has inserted more than 200 search keywords (named "Useful search terms") related to other popular

Table 5: The distribution of the number of misleading references to other popular apps.

Year	References (# apps)				
	≥ 1	≥ 5	≥ 10	≥ 15	≥ 20
Google Play 2014	489,795	18,479	593	122	68
Google Play 2015	792,130	39,827	1,232	261	96
Google Play 2017	1,261,613	86,730	3,013	148	20

apps in the description we crawled in 2015 (left), while the description has been improved as in 2017 (right). Nevertheless, there are still over 3,000 possible spamming apps in the 2017 snapshot. We manually examined some of them, and found that they tend to use some sophistic methods to insert keywords. Rather than listing keywords without logic, they seek to embed the keywords into real sentences. For example, "com.aditya.hexawhite" is an icon replacement app, which has incorporated lots of popular app names and keywords in its description using statements such as "compatible with [A list of App Names]" and "Icon List: [A list of App Names]"

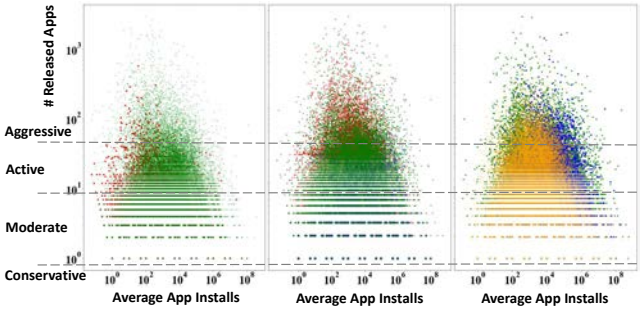


Figure 12: The distribution of app developers in Google Play over time. Green, Blue and Yellow dots represent new developers in 2014, 2015 and 2017 respectively. Red dot represents the developers removed in the next snapshot.

7 THE EVOLUTION OF APP DEVELOPERS

This section investigates the dynamics of app developers. Table 2 shows that there are roughly 400K to 500K developers in Google

Table 6: Developer groups.

Snapshot	Category (# developers, % Percentage)			
	Conservative	Moderate	Active	Aggressive
GP2014	239,331 (57.8%)	151,541 (36.6%)	20,219 (4.9%)	2,869 (0.7%)
GP2015	211,029 (52.9%)	156,961 (39.4%)	27,737 (7.0%)	3,128 (0.8%)
GP2017	306,996 (56.7%)	194,544 (36.0%)	35,066 (6.5%)	4,499 (0.8%)

Play during the span of our study. However, as we plotted in Figure 12, developers change greatly during the evolution, as a considerable number of developers have disappeared, while new developers have emerged. Only 42% of developers in the 2017 snapshot are originated from 2014. Next, we first study the developer popularity, then we classify developer into different groups, and further characterize spamming and malicious developers.

7.1 Top Developers

We characterize the developer popularity distribution for all the developers ordered by the number of accumulated installs, as illustrated in Figure 13(a). This result suggests that the distribution of developer installs also follows the Pareto effect. Top 1% of developers have occupied roughly 83% of total installs across the three snapshots. We further compare the catalog similarity of the top 1% of developers, and found that more than 80% of them are the same. Thus, although the active developers change greatly across snapshots, the most popular ones are relative stable, so that Google Play is dominated by a small number of developers all the time.

7.2 Developer Categorization

Previous work [69] proposed to classify developers into four groups based on the number of apps they released, including Conservative (1 app), Moderate (2 to 9 apps), Active (10 to 49 apps) and Aggressive (more than 50 apps). In this work, we follow this classification to study the evolution of app developers.

Table 6 and Figure 12 shows the distribution of different developer groups. The developer distribution follows the similar pattern. Conservative developers account for more than half of the community, while less than 1% of the developers are aggressive developers.

Spamming Developers. Previous work identified the prevalence of spamming developers in Google Play, who have released a large number of low-quality apps (mostly are clones) with almost no downloads. This study tries to study the evolution of spamming developers, i.e., whether spamming developers left or are removed. Following the same criteria (aggressive developers with no popular apps and with an average install number lower than 10,000), we flag each developer as spamming or not. We have identified the number of spamming developers in these three snapshots, with numbers at 2,378, 2,474 and 3,299 respectively.

Removal of Spamming Developers. Figure 12 indicates that a large number of developers in the 2014 and 2015 snapshots are removed. We are interested in whether spamming developers were removed or not, because many of them released hundreds of low-quality and least popular apps, which should be removed. Thus we analyze the percentage of removed apps for spamming developers we found in the the 2014 and 2015 snapshots, as shown in Figure 13(b). We can see that, more than 40% of the spamming developers are removed from Google Play, with some of them having released hundreds of apps. More than 40% of the released apps have been removed for roughly 60% of the spamming developers. For example, developer "ReverbNation Artists" had released 2,634 apps in total (with average installs of 226) in the snapshot 2015, while it was removed along with all its 2,634 apps in 2017.

7.3 Malicious Developers

We are also interested in the malicious developers who have released malware to Google Play. Thus, we regard the developers who have created at least one malicious app (with AV-Score >=10) as a malicious developer. The number of malicious developers is 13,488 (3.26%), 12,894 (3.23%) and 8,621 (1.59%), respectively. The most aggressive developers have released hundreds of malware. For example, "Apps Ministry LLC" has distributed 611 malware, and "AppShareNI" released 448 malware. We further analyze whether these malicious apps are dominated by malicious developers, as shown in Figure 13(c). We can see that, top 10% malicious developers account for roughly 50% to 60% of the malicious apps.

8 DISCUSSION

We believe that our efforts and the revealed insights can contribute to different stakeholders of the mobile app ecosystem.

App Markets. For market maintainers, because Google Play may remove low-quality and policy-violation apps regularly, the lessons learned from this longitudinal study could help app markets understand real threats and challenges, and help them further improve the markets. For example, market maintainers should pay more attention to the fraudulent app promotion activities mentioned in this paper. Besides, although the percentage of malicious apps and malicious developers in Google Play are decreasing year by year, there still exist new emerging malicious apps, while aggressive developers may still be able to release a large number of malware to the market.

App Developers and App Users. From the perspective of app developers, various findings revealed in this paper could help them make better decision on how to develop and release apps, e.g., app categories, app monetization schemes, and app growth rate, etc. For mobile users, it is better to avoid apps that have not been updated for years, and avoid the potentially risk/low-quality apps that were released by malicious and spamming developers.

9 CONCLUDING REMARKS

We conduct a large-scale longitudinal study to understand the evolution of mobile app ecosystems. Specifically, our analysis covers over 5 million app records collected from three different snapshots of Google Play over a period of more than three years. Overall, our findings suggest that the overall ecosystem shows a promising progress in regard of app popularity, user ratings and privacy respecting. However, there still exists a considerable number of unsolved issues including malware, app update issues, aggressive app promotion behaviors, and spamming/malicious developers. We believe that our research efforts can positively contribute to benefiting app users and developers, attracting the focus of the research community and regulators, and advocating best operational practices across app market operators.

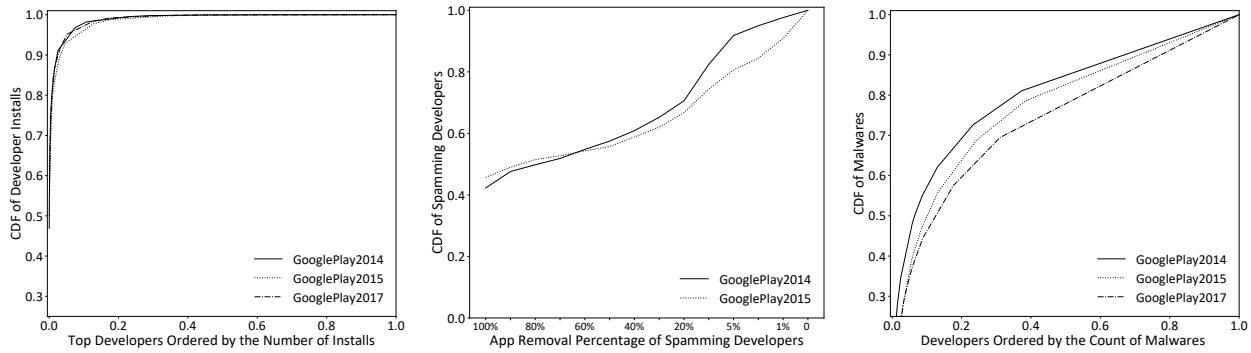


Figure 13: Statistics of app developers. (a) Top developers. (2) Removal of spamming developers. (c) Malicious developers.

ACKNOWLEDGEMENT

This work is supported by the National Key Research and Development Program of China (grant No.2017YFB0801903) and the National Natural Science Foundation of China (grants No.61702045 and No.61772042). Yao Guo is the Corresponding author.

REFERENCES

[1] 2017. Google To Delete Android Apps over Privacy Policy. https://www.techrepublic.com/article/google-will-soon-delete-apps-with-no-privacy-policies-from-play-store/.

[2] 2017. Request App Permissions. https://developer.android.com/training/permissions/requesting.

[3] 2018. Google Play is hosting a disturbing amount of cryptocurrency malware. https://thenextweb.com/hardfork/2018/04/20/google-play-cryptocurrency-apps-malware/.

[4] 2018. How Malware Keeps Sneaking Past Google Play's Defenses. https://www.wired.com/story/google-play-store-malware/.

[5] 2018. New Android Trojan malware discovered in Google Play. https://blog.malwarebytes.com/cybercrime/2017/11/new-trojan-malware-discovered-google-play/.

[6] 2018. PrivacyGrade: Grading The Privacy Of Smartphone Apps. privacy-grade.org/.

[7] Mohamed Ali, Mona Erfani Joorabchi, and Ali Mesbah. 2017. Same app, different app stores: A comparative study. In *Proceedings of the 4th International Conference on Mobile Software Engineering and Systems*. 79–90.

[8] Android. 2018. Permissions overview. https://developer.android.com/guide/topics/permissions/overview#permission-groups.

[9] AppBrain. 2018. Android Ad Network statistics and market share. https://www.appbrain.com/stats/libraries/ad-networks.

[10] Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, Konrad Rieck, and CERT Siemens. 2014. DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket.. In *The Network and Distributed System Security Symposium (NDSS '14)*. 23–26.

[11] Steven Arzt, Siegfried Rasthofer, Christian Fritz, Eric Bodden, Alexandre Bartel, Jacques Klein, Yves Le Traon, Damien Octeau, and Patrick McDaniel. 2014. Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps. *Acm Sigplan Notices* 49, 6 (2014), 259–269.

[12] Michael Backes, Sven Bugiel, and Erik Derr. 2016. Reliable third-party library detection in android and its security applications. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 356–367.

[13] Paolo Calciati and Alessandra Gorla. 2017. How do apps evolve in their permission requests?: a preliminary study. In *Proceedings of the 14th International Conference on Mining Software Repositories*. 37–41.

[14] Bogdan Carbunar and Rahul Potharaju. 2015. A longitudinal study of the Google app market. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*. ACM, 242–249.

[15] Kai Chen, Peng Liu, and Yingjun Zhang. 2014. Achieving accuracy and scalability simultaneously in detecting application clones on android markets. In *Proceedings of the 36th International Conference on Software Engineering*. 175–186.

[16] Ning Chen, Jialiu Lin, Steven CH Hoi, Xiaokui Xiao, and Boshen Zhang. 2014. AR-miner: mining informative reviews for developers from mobile app marketplace. In *Proceedings of the 36th International Conference on Software Engineering*. ACM, 767–778.

[17] Jonathan Crussell, Ryan Stevens, and Hao Chen. 2014. Madfraud: Investigating ad fraud in android applications. In *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*. 123–134.

[18] Christian de Looper. 2018. From Android 1.0 to Android 9.0, here's how Google's OS evolved over a decade. https://www.digitaltrends.com/mobile/android-version-history/.

[19] Christian de Looper. 2018. Google's AI has cut down Google Play's malware by more than half. https://www.digitaltrends.com/mobile/google-play-malware-artificial-intelligence/.

[20] Feng Dong, Haoyu Wang, Li Li, Yao Guo, Tegawendé F Bissyandé, Tianming Liu, Guoai Xu, and Jacques Klein. [n. d.]. Frauddroid: Automated ad fraud detection for android apps. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 257–268.

[21] Jon Fingas. 2018. Android malware returned to Google Play with just a name change. https://www.engadget.com/2018/05/14/android-malware-returned-to-google-play-with-just-a-name-change/.

[22] Bin Fu, Jialiu Lin, Lei Li, Christos Faloutsos, Jason Hong, and Norman Sadeh. 2013. Why people hate your app: Making sense of user feedback in a mobile app store. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1276–1284.

[23] Jun Gao, Li Li, Pingfan Kong, Tegawendé F Bissyandé, and Jacques Klein. 2018. Poster: On Vulnerability Evolution in Android Apps. In *2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion)*. 276–277.

[24] GooglePlay. 2018. Store Listing and Promotion. https://play.google.com/intl/en-US/about/storelisting-promotional/metadata/index.html.

[25] Michael Grace, Yajin Zhou, Qiang Zhang, Shihong Zou, and Xuxian Jiang. 2012. Riskranker: scalable and accurate zero-day android malware detection. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 281–294.

[26] Michael C Grace, Wu Zhou, Xuxian Jiang, and Ahmad-Reza Sadeghi. 2012. Unsafe exposure analysis of mobile in-app advertisements. In *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks*. 101–112.

[27] JERRY HILDENBRAND. 2018. 10 years of Android: How the best mobile OS was made. https://www.androidcentral.com/10-years-android-how-best-mobile-os-was-made.

[28] Yangyu Hu, Haoyu Wang, Li Li, Yao Guo, Guoai Xu, and Ren He. 2019. Want to Earn a Few Extra Bucks? A First Look at Money-making Apps. In *IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER 2019)*.

[29] Yangyu Hu, Haoyu Wang, Yajin Zhou, Yao Guo, Li Li, Bingxuan Luo, and Fangren Xu. 2018. Dating with scambots: Understanding the ecosystem of fraudulent dating applications. *arXiv preprint arXiv:1807.04901* (2018).

[30] Muhammad Ikram, Narseo Vallina-Rodriguez, Suranga Seneviratne, Mohamed Ali Kaafar, and Vern Paxson. 2016. An analysis of the privacy and security risks of android vpn permission-enabled apps. In *Proceedings of the 2016 Internet Measurement Conference*. ACM, 349–364.

[31] Yuta Ishii, Takuya Watanabe, Fumihiro Kanei, Yuta Takata, Eitaro Shioji, Mitsuaki Akiyama, Takeshi Yagi, Bo Sun, and Tatsuya Mori. 2017. Understanding the security management of global third-party Android marketplaces. In *Proceedings of the 2nd ACM SIGSOFT International Workshop on App Market Analytics*. 12–18.

[32] Mona Erfani Joorabchi, Ali Mesbah, and Philippe Kruchten. 2013. Real challenges in mobile app development. In *The ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*. 15–24.

[33] Hammad Khalid, Emad Shihab, Meiyappan Nagappan, and Ahmed E Hassan. 2015. What do mobile app users complain about? *IEEE Software* 32, 3 (2015), 70–77.

[34] Chengze Li, Haoyu Wang, Junfeng Wang, Qi Li, Jianbo Yu, Jingyi Guo, Guoai Xu, and Yanhui Guo. 2017. CRSPR: PageRank for Android apps. *IEEE Access* 5 (2017), 18004–18015.

[35] Li Li, Alexandre Bartel, Tegawendé F Bissyandé, Jacques Klein, and Yves Le Traon. 2015. Apkcombiner: Combining multiple android apps to support inter-app analysis. In *IFIP International Information Security Conference*. Springer, 513–527.

[36] Li Li, Tegawendé F Bissyandé, Jacques Klein, and Yves Le Traon. 2016. An investigation into the use of common libraries in android apps. In *The 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*. 403–414.

[37] Li Li, Tegawendé F Bissyandé, Haoyu Wang, and Jacques Klein. 2018. Cid: Automating the detection of api-related compatibility issues in android apps. In *Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ACM, 153–163.

[38] Li Li, Jun Gao, Médéric Hurier, Pingfan Kong, Tegawendé F Bissyandé, Alexandre Bartel, Jacques Klein, and Yves Le Traon. 2017. AndroZoo++: Collecting Millions of Android Apps and Their Metadata for the Research Community. *arXiv preprint arXiv:1709.05281* (2017).

[39] Yuanchun Li, Yao Guo, and Xiangqun Chen. 2016. Peruim: Understanding mobile application privacy with permission-ui mapping. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 682–693.

[40] Soo Ling Lim and Peter J Bentley. 2012. How to be a successful app developer: Lessons from the simulation of an app ecosystem. *ACM SIGEVOlution* 6, 1 (2012), 2–15.

[41] Jialiu Lin, Shahriyar Amini, Jason I Hong, Norman Sadeh, Janne Lindqvist, and Joy Zhang. 2012. Expectation and purpose: understanding users' mental models of mobile app privacy through crowdsourcing. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. ACM, 501–510.

[42] Minxing Liu, Haoyu Wang, Yao Guo, and Jason Hong. 2016. Identifying and analyzing the privacy of apps for kids. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications*. 105–110.

[43] Ziang Ma, Haoyu Wang, Yao Guo, and Xiangqun Chen. 2016. LibRadar: fast and accurate detection of third-party libraries in Android apps. In *Proceedings of the 38th International Conference on Software Engineering Companion*. 653–656.

[44] Ziang Ma, Haoyu Wang, Yao Guo, and Xiangqun Chen. 2016. LibRadar: fast and accurate detection of third-party libraries in Android apps. In *Proceedings of the 38th international conference on software engineering companion*. ACM, 653–656.

[45] Sarah Perez. 2017. App economy to grow to $6.3 trillion in 2021, user base to nearly double to 6.3 billion. https://techcrunch.com/tag/app-ecosystem/.

[46] Thanasis Petsas, Antonis Papadogiannakis, Michalis Polychronakis, Evangelos P Markatos, and Thomas Karagiannis. 2017. Measurement, modeling, and analysis of the mobile app ecosystem. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)* 2, 2 (2017), 7.

[47] Google Play. 2018. Privacy, Security, and Deception. https://play.google.com/about/privacy-security-deception/.

[48] Ben Popper. 2017. Google announces over 2 billion monthly active devices on Android. https://www.theverge.com/2017/5/17/15654454/android-reaches-2-billion-monthly-active-users.

[49] Rahul Potharaju, Mizanur Rahman, and Bogdan Carbunar. 2017. A Longitudinal Study of Google Play. *IEEE Transactions on computational social systems* 4, 3 (2017), 135–149.

[50] Israel J Mojica Ruiz, Meiyappan Nagappan, Bram Adams, Thorsten Berger, Steffen Dienst, and Ahmed E Hassan. 2016. Examining the rating system used in mobile-app stores. *IEEE Software* 33, 6 (2016), 86–92.

[51] Marcos Sebastián, Richard Rivera, Platon Kotzias, and Juan Caballero. 2016. Avclass: A tool for massive malware labeling. In *International Symposium on Research in Attacks, Intrusions, and Defenses*. 230–253.

[52] TrendLabs Security. 2018. A Look at Google Bouncer. https://blog.trendmicro.com/trendlabs-security-intelligence/a-look-at-google-bouncer/.

[53] Suranga Seneviratne, Harini Kolamunna, and Aruna Seneviratne. 2015. A measurement study of tracking in paid mobile applications. In *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. ACM, 7.

[54] Suranga Seneviratne, Aruna Seneviratne, Mohamed Ali Kaafar, Anirban Mahanti, and Prasant Mohapatra. 2015. Early detection of spam mobile apps. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. 949–959.

[55] Statista. 2018. Android - Statistics & Facts. https://www.statista.com/topics/876/android/.

[56] Vincent F Taylor and Ivan Martinovic. 2017. A longitudinal study of financial apps in the Google Play Store. In *2017 International Conference on Financial Cryptography and Data Security*.

[57] Vincent F Taylor and Ivan Martinovic. 2017. To update or not to update: Insights from a two-year study of android app evolution. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 45–57.

[58] Yuan Tian, Meiyappan Nagappan, David Lo, and Ahmed E Hassan. 2015. What are the characteristics of high-rated apps? a case study on free android applications. In *Software Maintenance and Evolution (ICSME), 2015 IEEE International Conference on*. 301–310.

[59] Alok Tongaonkar, Shuaifu Dai, Antonio Nucci, and Dawn Song. 2013. Understanding mobile app usage patterns using in-app advertisements. In *International Conference on Passive and Active Network Measurement*. 63–72.

[60] Nicolas Viennot, Edward Garcia, and Jason Nieh. 2014. A measurement study of google play. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 42. ACM, 221–233.

[61] Jaikumar Vijayan. 2018. Google Removes 145 Malware-Laden Apps From Play Store. http://www.eweek.com/mobile/google-removes-145-malware-laden-apps-from-play-store.

[62] Haoyu Wang and Yao Guo. 2017. Understanding third-party libraries in mobile app analysis. In *Software Engineering Companion (ICSE-C)*. 515–516.

[63] Haoyu Wang, Yao Guo, Ziang Ma, and Xiangqun Chen. 2015. WuKong: a scalable and accurate two-phase approach to Android app clone detection. In *Proceedings of the 2015 International Symposium on Software Testing and Analysis*. 71–82.

[64] Haoyu Wang, Yao Guo, Ziang Ma, and Xiangqun Chen. 2017. Automated Detection and Classification of Third-Party Libraries in Large Scale Android Apps. *Journal of Software* 28, 6 (2017), 1373–1388.

[65] Haoyu Wang, Yao Guo, Zihao Tang, Guangdong Bai, and Xiangqun Chen. 2015. Reevaluating android permission gaps with static and dynamic analysis. In *2015 IEEE Global Communications Conference (GLOBECOM)*. 1–6.

[66] Haoyu Wang, Jason Hong, and Yao Guo. 2015. Using text mining to infer the purpose of permission use in mobile apps. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 1107–1118.

[67] Haoyu Wang, Hao Li, Li Li, Yao Guo, and Guoai Xu. 2018. Why are Android apps removed from Google Play?: a large-scale empirical study. In *Proceedings of the 15th International Conference on Mining Software Repositories*. ACM, 231–242.

[68] Haoyu Wang, Yuanchun Li, Yao Guo, Yuvraj Agarwal, and Jason I Hong. 2017. Understanding the purpose of permission use in mobile apps. *ACM Transactions on Information Systems (TOIS)* 35, 4 (2017), 43.

[69] Haoyu Wang, Zhe Liu, Yao Guo, Xiangqun Chen, Miao Zhang, Guoai Xu, and Jason Hong. 2017. An explorative study of the mobile app ecosystem from app developers' perspective. In *Proceedings of the 26th International Conference on World Wide Web*. 163–172.

[70] Haoyu Wang, Zhe Liu, Jingyue Liang, Narseo Vallina-Rodriguez, Yao Guo, Li Li, Juan Tapiador, Jingcun Cao, and Guoai Xu. 2018. Beyond Google Play: A Large-Scale Comparative Study of Chinese Android App Markets. In *The Internet Measurement Conference (IMC '18)*.

[71] Fengguo Wei, Yuping Li, Sankardas Roy, Xinming Ou, and Wu Zhou. 2017. Deep ground truth analysis of current android malware. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. 252–276.

[72] Wikipedia. 2018. Google Play - Wikipedia. https://en.wikipedia.org/wiki/Google_Play.

[73] Le Yu, Xiapu Luo, Xule Liu, and Tao Zhang. 2016. Can we trust the privacy policies of android apps?. In *The 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN '16)*. 538–549.

[74] Hengshu Zhu, Hui Xiong, Yong Ge, and Enhong Chen. 2015. Discovery of ranking fraud for mobile apps. *IEEE Transactions on knowledge and data engineering* 27, 1 (2015), 74–87.

Multi-Task Feature Learning for Knowledge Graph Enhanced Recommendation

Hongwei Wang[1,2], Fuzheng Zhang[3], Miao Zhao[4], Wenjie Li[4], Xing Xie[2], Minyi Guo[1*]
[1]Shanghai Jiao Tong University, Shanghai, China [2]Microsoft Research Asia, Beijing, China
[3]Meituan-Dianping Group, Beijing, China [4]The Hong Kong Polytechnic University, Hong Kong, China
wanghongwei55@gmail.com,zhangfuzheng@meituan.com,{csmiaozhao,cswjli}@comp.polyu.edu.hk,
xingx@microsoft.com,guo-my@cs.sjtu.edu.cn

ABSTRACT

Collaborative filtering often suffers from sparsity and cold start problems in real recommendation scenarios, therefore, researchers and engineers usually use side information to address the issues and improve the performance of recommender systems. In this paper, we consider knowledge graphs as the source of side information. We propose **MKR**, a **M**ulti-task feature learning approach for **K**nowledge graph enhanced **R**ecommendation. MKR is a deep end-to-end framework that utilizes knowledge graph embedding task to assist recommendation task. The two tasks are associated by cross&compress units, which automatically share latent features and learn high-order interactions between items in recommender systems and entities in the knowledge graph. We prove that cross&compress units have sufficient capability of polynomial approximation, and show that MKR is a generalized framework over several representative methods of recommender systems and multi-task learning. Through extensive experiments on real-world datasets, we demonstrate that MKR achieves substantial gains in movie, book, music, and news recommendation, over state-of-the-art baselines. MKR is also shown to be able to maintain satisfactory performance even if user-item interactions are sparse.

KEYWORDS

Recommender systems; Knowledge graph; Multi-task learning

ACM Reference Format:
Hongwei Wang[1,2], Fuzheng Zhang[3], Miao Zhao[4], Wenjie Li[4], Xing Xie[2], Minyi Guo[1]. 2019. Multi-Task Feature Learning for Knowledge Graph Enhanced Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313411

1 INTRODUCTION

Recommender systems (RS) aims to address the information explosion and meet users personalized interests. One of the most popular recommendation techniques is collaborative filtering (CF) [11], which utilizes users' historical interactions and makes recommendations based on their common preferences. However, CF-based methods usually suffer from the sparsity of user-item interactions and the cold start problem. Therefore, researchers propose using *side information* in recommender systems, including social networks [10], attributes [30], and multimedia (e.g., texts [29], images [40]). *Knowledge graphs* (KGs) are one type of side information for RS, which usually contain fruitful facts and connections about items. Recently, researchers have proposed several academic and commercial KGs, such as NELL[1], DBpedia[2], Google Knowledge Graph[3] and Microsoft Satori[4]. Due to its high dimensionality and heterogeneity, a KG is usually pre-processed by *knowledge graph embedding* (KGE) methods [27], which embeds entities and relations into low-dimensional vector spaces while preserving its inherent structure.

Existing KG-aware methods

Inspired by the success of applying KG in a wide variety of tasks, researchers have recently tried to utilize KG to improve the performance of recommender systems [31, 32, 39, 40, 45]. Personalized Entity Recommendation (PER) [39] and Factorization Machine with Group lasso (FMG) [45] treat KG as a heterogeneous information network, and extract meta-path/meta-graph based latent features to represent the connectivity between users and items along different types of relation paths/graphs. It should be noted that PER and FMG rely heavily on manually designed meta-paths/meta-graphs, which limits its application in generic recommendation scenarios. Deep Knowledge-aware Network (DKN) [32] designs a CNN framework to combine entity embeddings with word embeddings for news recommendation. However, the entity embeddings are required in advance of using DKN, causing DKN to lack an end-to-end way of training. Another concern about DKN is that it can hardly incorporate side information other than texts. RippleNet [31] is a memory-network-like model that propagates users' potential preferences in the KG and explores their hierarchical interests. But the importance of relations is weakly characterized in RippleNet, because the embedding matrix of a relation $\mathbf{R}$ can hardly be trained to capture the sense of importance in the quadratic form $\mathbf{v}^\top\mathbf{R}\mathbf{h}$ ($\mathbf{v}$ and $\mathbf{h}$ are embedding vectors of two entities). Collaborative Knowledge base Embedding (CKE) [40] combines CF with structural knowledge, textual knowledge, and visual knowledge in a unified framework. However, the KGE module in CKE (i.e., TransR [13]) is more suitable for in-graph applications (such as KG

*M. Guo is the corresponding author. This work was partially sponsored by the National Basic Research 973 Program of China (2015CB352403) and National Natural Science Foundation of China (61272291).

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313411

[1]http://rtw.ml.cmu.edu/rtw/
[2]http://wiki.dbpedia.org/
[3]https://developers.google.com/knowledge-graph/
[4]https://searchengineland.com/library/bing/bing-satori

completion and link prediction) rather than recommendation. In addition, the CF module and the KGE module are loosely coupled in CKE under a Bayesian framework, making the supervision from KG less obvious for recommender systems.

The proposed approach

To address the limitations of previous work, we propose MKR, a multi-task learning (MTL) approach for knowledge graph enhanced recommendation. MKR is a generic, end-to-end deep recommendation framework, which aims to utilize KGE task to assist recommendation task[5]. Note that the two tasks are not mutually independent, but are highly correlated since an item in RS may associate with one or more entities in KG. Therefore, an item and its corresponding entity are likely to have a similar proximity structure in RS and KG, and share similar features in low-level and non-task-specific latent feature spaces [15]. We will further validate the similarity in the experiments section. To model the shared features between items and entities, we design a *cross&compress unit* in MKR. The cross&compress unit explicitly models high-order interactions between item and entity features, and automatically control the cross knowledge transfer for both tasks. Through cross&compress units, representations of items and entities can complement each other, assisting both tasks in avoiding fitting noises and improving generalization. The whole framework can be trained by alternately optimizing the two tasks with different frequencies, which endows MKR with high flexibility and adaptability in real recommendation scenarios.

We probe the expressive capability of MKR and show, through theoretical analysis, that the cross&compress unit is capable of approximating sufficiently high order feature interactions between items and entities. We also show that MKR is a generalized framework over several representative methods of recommender systems and multi-task learning, including factorization machines [22, 23], deep&cross network [34], and cross-stitch network [18]. Empirically, we evaluate our method in four recommendation scenarios, i.e., movie, book, music, and news recommendations. The results demonstrate that MKR achieves substantial gains over state-of-the-art baselines in both click-through rate (CTR) prediction (e.g., 11.6% *AUC* improvements on average for movies) and top-K recommendation (e.g., 66.4% *Recall*@10 improvements on average for books). MKR is also shown to maintain satisfactory performance even when user-item interactions are sparse.

Contribution

It is worth noticing that the problem studied in this paper can also be modelled as *cross-domain recommendation* [26] or *transfer learning* [21], since we care more about the performance of recommendation task. However, the key observation is that though cross-domain recommendation and transfer learning have single objective for the target domain, their loss functions still contain constraint terms for measuring data distribution in the source domain or similarity between two domains. In our proposed MKR, the KGE task serves as the constraint term *explicitly* to provide regularization for recommender systems. We would like to emphasize that the major contribution of this paper is exactly modeling the problem as multi-task learning: We go a step further than cross-domain recommendation and transfer learning by finding that the inter-task similarity is helpful to not only recommender systems but also knowledge graph embedding, as shown in theoretical analysis and experiment results.

[5]KGE task can also benefit from recommendation task empirically as shown in the experiments section.

2 OUR APPROACH

In this section, we first formulate the knowledge graph enhanced recommendation problem, then introduce the framework of MKR and present the design of the cross&compress unit, recommendation module and KGE module in detail. We lastly discuss the learning algorithm for MKR.

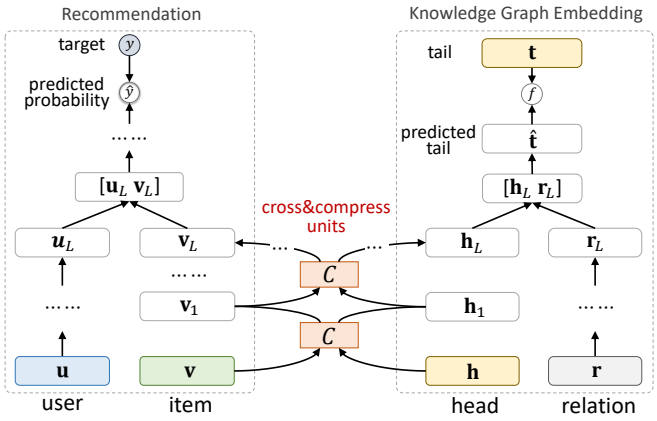
2.1 Problem Formulation

We formulate the knowledge graph enhanced recommendation problem in this paper as follows. In a typical recommendation scenario, we have a set of M users $\mathcal{U} = \{u_1, u_2, ..., u_M\}$ and a set of N items $\mathcal{V} = \{v_1, v_2, ..., v_N\}$. The user-item interaction matrix $\mathbf{Y} \in \mathbb{R}^{M \times N}$ is defined according to users' implicit feedback, where $y_{uv} = 1$ indicates that user u engaged with item v, such as behaviors of clicking, watching, browsing, or purchasing; otherwise $y_{uv} = 0$. Additionally, we also have access to a knowledge graph $\mathcal{G}$, which is comprised of entity-relation-entity triples (h, r, t). Here h, r, and t denote the head, relation, and tail of a knowledge triple, respectively. For example, the triple (*Quentin Tarantino*, *film.director.film*, *Pulp Fiction*) states the fact that Quentin Tarantino directs the film Pulp Fiction. In many recommendation scenarios, an item $v \in \mathcal{V}$ may associate with one or more entities in $\mathcal{G}$. For example, in movie recommendation, the item "Pulp Fiction" is linked with its namesake in a knowledge graph, while in news recommendation, news with the title "Trump pledges aid to Silicon Valley during tech meeting" is linked with entities "Donald Trump" and "Silicon Valley" in a knowledge graph.

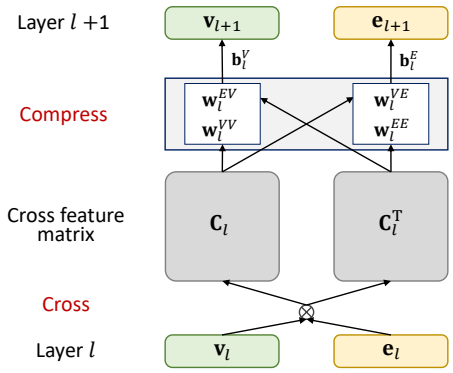
Given the user-item interaction matrix $\mathbf{Y}$ as well as the knowledge graph $\mathcal{G}$, we aim to predict whether user u has potential interest in item v with which he has had no interaction before. Our goal is to learn a prediction function $\hat{y}_{uv} = \mathcal{F}(u, v|\Theta, \mathbf{Y}, \mathcal{G})$, where $\hat{y}_{uv}$ denotes the probability that user u will engage with item v, and Θ is the model parameters of function $\mathcal{F}$.

2.2 Framework

The framework of MKR is illustrated in Figure 1a. MKR consists of three main components: recommendation module, KGE module, and cross&compress units. (1) The recommendation module on the left takes a user and an item as input, and uses a multi-layer perceptron (MLP) and cross&compress units to extract short and dense features for the user and the item, respectively. The extracted features are then fed into another MLP together to output the predicted probability. (2) Similar to the left part, the KGE module in the right part also uses multiple layers to extract features from the head and relation of a knowledge triple, and outputs the representation of the predicted tail under the supervision of a score function f and the real tail. (3) The recommendation module and the KGE module are bridged by specially designed cross&compress units. The proposed unit can automatically learn high-order feature interactions of items in recommender systems and entities in the knowledge graph.



(a) Framework of MKR



(b) Cross&compress unit

Figure 1: (a) The framework of MKR. The left and right part illustrate the recommendation module and the KGE module, respectively, which are bridged by the cross&compress units. (b) Illustration of a cross&compress unit. The cross&compress unit generates a cross feature matrix from item and entity vectors by *cross* operation, and outputs their vectors for the next layer by *compress* operation.

2.3 Cross&compress Unit

To model feature interactions between items and entities, we design a cross&compress unit in MKR framework. As shown in Figure 1b, for item v and one of its associated entities e, we first construct $d \times d$ pairwise interactions of their latent feature $\mathbf{v}_l \in \mathbb{R}^d$ and $\mathbf{e}_l \in \mathbb{R}^d$ from layer l:

$$\mathbf{C}_l = \mathbf{v}_l \mathbf{e}_l^\top = \begin{bmatrix} v_l^{(1)} e_l^{(1)} & \cdots & v_l^{(1)} e_l^{(d)} \\ \cdots & & \cdots \\ v_l^{(d)} e_l^{(1)} & \cdots & v_l^{(d)} e_l^{(d)} \end{bmatrix}, \tag{1}$$

where $\mathbf{C}_l \in \mathbb{R}^{d \times d}$ is the cross feature matrix of layer l, and d is the dimension of hidden layers. This is called the *cross* operation, since each possible feature interaction $v_l^{(i)} e_l^{(j)}, \forall (i,j) \in \{1, ..., d\}^2$ between item v and its associated entity e is modeled explicitly in the cross feature matrix. We then output the feature vectors of items and entities for the next layer by projecting the cross feature matrix into their latent representation spaces:

$$\begin{aligned} \mathbf{v}_{l+1} &= \mathbf{C}_l \mathbf{w}_l^{VV} + \mathbf{C}_l^\top \mathbf{w}_l^{EV} + \mathbf{b}_l^V = \mathbf{v}_l \mathbf{e}_l^\top \mathbf{w}_l^{VV} + \mathbf{e}_l \mathbf{v}_l^\top \mathbf{w}_l^{EV} + \mathbf{b}_l^V, \\ \mathbf{e}_{l+1} &= \mathbf{C}_l \mathbf{w}_l^{VE} + \mathbf{C}_l^\top \mathbf{w}_l^{EE} + \mathbf{b}_l^E = \mathbf{v}_l \mathbf{e}_l^\top \mathbf{w}_l^{VE} + \mathbf{e}_l \mathbf{v}_l^\top \mathbf{w}_l^{EE} + \mathbf{b}_l^E, \end{aligned} \tag{2}$$

where $\mathbf{w}_l^{\cdot\cdot} \in \mathbb{R}^d$ and $\mathbf{b}_l^{\cdot} \in \mathbb{R}^d$ are trainable weight and bias vectors. This is called the *compress* operation, since the weight vectors project the cross feature matrix from $\mathbb{R}^{d \times d}$ space back to the feature spaces $\mathbb{R}^d$. Note that in Eq. (2), the cross feature matrix is compressed along both horizontal and vertical directions (by operating on $\mathbf{C}_l$ and $\mathbf{C}_l^\top$) for the sake of symmetry, but we will provide more insights of the design in Section 3.2. For simplicity, the cross&compress unit is denoted as:

$$[\mathbf{v}_{l+1}, \mathbf{e}_{l+1}] = C(\mathbf{v}_l, \mathbf{e}_l), \tag{3}$$

and we use a suffix [v] or [e] to distinguish its two outputs in the following of this paper. Through cross&compress units, MKR can adaptively adjust the weights of knowledge transfer and learn the relevance between the two tasks.

It should be noted that cross&compress units should only exist in low-level layers of MKR, as shown in Figure 1a. This is because: (1) In deep architectures, features usually transform from general to specific along the network, and feature transferability drops significantly in higher layers with increasing task dissimilarity [38]. Therefore, sharing high-level layers risks to possible negative transfer, especially for the heterogeneous tasks in MKR. (2) In high-level layers of MKR, item features are mixed with user features, and entity features are mixed with relation features. The mixed features are not suitable for sharing since they have no explicit association.

2.4 Recommendation Module

The input of the recommendation module in MKR consists of two raw feature vectors $\mathbf{u}$ and $\mathbf{v}$ that describe user u and item v, respectively. $\mathbf{u}$ and $\mathbf{v}$ can be customized as one-hot ID [8], attributes [30], bag-of-words [29], or their combinations, based on the application scenario. Given user u's raw feature vector $\mathbf{u}$, we use an L-layer MLP to extract his latent condensed feature[6]:

$$\mathbf{u}_L = \mathcal{M}(\mathcal{M}(\cdots \mathcal{M}(\mathbf{u}))) = \mathcal{M}^L(\mathbf{u}), \tag{4}$$

where $\mathcal{M}(\mathbf{x}) = \sigma(\mathbf{W}\mathbf{x}+\mathbf{b})$ is a fully-connected neural network layer[7] with weight $\mathbf{W}$, bias $\mathbf{b}$, and nonlinear activation function $\sigma(\cdot)$. For item v, we use L cross&compress units to extract its feature:

$$\mathbf{v}_L = \mathbb{E}_{e \sim \mathcal{S}(v)} \left[C^L(\mathbf{v}, \mathbf{e})[\mathbf{v}] \right], \tag{5}$$

where $\mathcal{S}(v)$ is the set of associated entities of item v.

After having user u's latent feature $\mathbf{u}_L$ and item v's latent feature $\mathbf{v}_L$, we combine the two pathways by a predicting function f_{RS},

[6] We use the exponent notation L in Eq. (4) and following equations in the rest of this paper for simplicity, but note that the parameters of L layers are actually different.
[7] Exploring a more elaborate design of layers in the recommendation module is an important direction of future work.

for example, inner product or an H-layer MLP. The final predicted probability of user u engaging item v is:

$$\hat{y}_{uv} = \sigma\big(f_{RS}(\mathbf{u}_L, \mathbf{v}_L)\big). \tag{6}$$

2.5 Knowledge Graph Embedding Module

Knowledge graph embedding is to embed entities and relations into continuous vector spaces while preserving their structure. Recently, researchers have proposed a great many KGE methods, including translational distance models [2, 13] and semantic matching models [14, 19]. In MKR, we propose a deep semantic matching architecture for KGE module. Similar to the recommendation module, for a given knowledge triple (h, r, t), we first utilize multiple cross&compress units and nonlinear layers to process the raw feature vectors of head h and relation r (including ID [13], types [36], textual description [35], etc.), respectively. Their latent features are then concatenated together, followed by a K-layer MLP for predicting tail t:

$$\begin{aligned} \mathbf{h}_L &= \mathbb{E}_{v \sim \mathcal{S}(h)} \Big[C^L(\mathbf{v}, \mathbf{h})[\mathbf{e}] \Big], \\ \mathbf{r}_L &= \mathcal{M}^L(\mathbf{r}), \\ \hat{\mathbf{t}} &= \mathcal{M}^K \left(\begin{bmatrix} \mathbf{h}_L \\ \mathbf{r}_L \end{bmatrix} \right), \end{aligned} \tag{7}$$

where $\mathcal{S}(h)$ is the set of associated items of entity h, and $\hat{\mathbf{t}}$ is the predicted vector of tail t. Finally, the score of the triple (h, r, t) is calculated using a score (similarity) function f_{KG}:

$$score(h, r, t) = f_{KG}(\mathbf{t}, \hat{\mathbf{t}}), \tag{8}$$

where $\mathbf{t}$ is the real feature vector of t. In this paper, we use the normalized inner product $f_{KG}(\mathbf{t}, \hat{\mathbf{t}}) = \sigma(\mathbf{t}^\top \hat{\mathbf{t}})$ as the choice of score function [18], but other forms of (dis)similarity metrics can also be applied here such as KullbackâĂŞLeibler divergence.

2.6 Learning Algorithm

The complete loss function of MKR is as follows:

$$\begin{aligned} \mathcal{L} =& \mathcal{L}_{RS} + \mathcal{L}_{KG} + \mathcal{L}_{REG} \\ =& \sum_{u \in \mathcal{U}, v \in \mathcal{V}} \mathcal{J}(\hat{y}_{uv}, y_{uv}) \\ & - \lambda_1 \Big(\sum_{(h,r,t) \in \mathcal{G}} score(h, r, t) - \sum_{(h', r, t') \notin \mathcal{G}} score(h', r, t') \Big) \\ & + \lambda_2 \|\mathbf{W}\|_2^2. \end{aligned} \tag{9}$$

In Eq. (9), the first term measures loss in the recommendation module, where u and v traverse the set of users and the items, respectively, and $\mathcal{J}$ is the cross-entropy function. The second term calculates the loss in the KGE module, in which we aim to increase the score for all true triples while reducing the score for all false triples. The last item is the regularization term for preventing overfitting, λ_1 and λ_2 are the balancing parameters.[8]

Note that the loss function in Eq. (9) traverses all possible user-item pairs and knowledge triples. To make computation more efficient, following [17], we use a negative sampling strategy during training.

The learning algorithm of MKR is presented in Algorithm 1, in which a training epoch consists of two stages: recommendation task (line 3-7) and KGE task (line 8-10). In each iteration, we repeat training on recommendation task for t times (t is a hyper-parameter and normally $t > 1$) before training on KGE task once in each epoch, since we are more focused on improving recommendation performance. We will discuss the choice of t in the experiments section.

[8] λ_1 can be seen as the ratio of two learning rates for the two tasks.

Algorithm 1 Multi-Task Training for MKR

Require: Interaction matrix $\mathbf{Y}$, knowledge graph $\mathcal{G}$
Ensure: Prediction function $\mathcal{F}(u, v|\Theta, \mathbf{Y}, \mathcal{G})$

1: Initialize all parameters
2: **for** number of training iteration **do**
   // *recommendation task*
3: **for** t steps **do**
4: Sample minibatch of positive and negative interactions from $\mathbf{Y}$;
5: Sample $e \sim \mathcal{S}(v)$ for each item v in the minibatch;
6: Update parameters of $\mathcal{F}$ by gradient descent on Eq. (1)-(6), (9);
7: **end for**
   // *knowledge graph embedding task*
8: Sample minibatch of true and false triples from $\mathcal{G}$;
9: Sample $v \sim \mathcal{S}(h)$ for each head h in the minibatch;
10: Update parameters of $\mathcal{F}$ by gradient descent on Eq. (1)-(3), (7)-(9);
11: **end for**

3 THEORETICAL ANALYSIS

In this section, we prove that cross&compress units have sufficient capability of polynomial approximation. We also show that MKR is a generalized framework over several representative methods of recommender systems and multi-task learning.

3.1 Polynomial Approximation

According to the Weierstrass approximation theorem [25], any function under certain smoothness assumption can be approximated by a polynomial to an arbitrary accuracy. Therefore, we examine the ability of high-order interaction approximation of the cross&compress unit. We show that cross&compress units can model the order of item-entity feature interaction up to exponential degree:

Theorem 1. *Denote the input of item and entity in MKR network as* $\mathbf{v} = [v_1 \cdots v_d]^\top$ *and* $\mathbf{e} = [e_1 \cdots e_d]^\top$*, respectively. Then the cross terms about* $\mathbf{v}$ *and* $\mathbf{e}$ *in* $\|\mathbf{v}_L\|_1$ *and* $\|\mathbf{e}_L\|_1$ *(the L1-norm of* $\mathbf{v}_L$ *and* $\mathbf{e}_L$*) with maximal degree is* $k_{\boldsymbol{\alpha},\boldsymbol{\beta}} v_1^{\alpha_1} \cdots v_d^{\alpha_d} e_1^{\beta_1} \cdots e_d^{\beta_d}$*, where* $k_{\boldsymbol{\alpha},\boldsymbol{\beta}} \in \mathbb{R}$*,* $\alpha_i, \beta_i \in \mathbb{N}$ *for* $i \in \{1, \cdots, d\}$*,* $\alpha_1 + \cdots + \alpha_d = 2^{L-1}$*, and* $\beta_1 + \cdots + \beta_d = 2^{L-1}$ *(*$L \geq 1, \mathbf{v}_0 = \mathbf{v}, \mathbf{e}_0 = \mathbf{e}$*).*

In recommender systems, $\prod_{i=1}^d v_i^{\alpha_i} e_i^{\beta_i}$ is also called *combinatorial* feature, as it measures the interactions of multiple original features. Theorem 1 states that cross&compress units can automatically model the combinatorial features of items and entities for sufficiently high order, which demonstrates the superior approximation capacity of MKR as compared with existing work such as Wide&Deep [3], factorization machines [22, 23] and DCN [34]. The

proof of Theorem 1 is provided in the Appendix. Note that Theorem 1 gives a theoretical view of the polynomial approximation ability of the cross&compress unit rather than providing guarantees on its actual performance. We will empirically evaluate the cross&compress unit in the experiments section.

3.2 Unified View of Representative Methods

In the following we provide a unified view of several representative models in recommender systems and multi-task learning, by showing that they are restricted versions of or theoretically related to MKR. This justifies the design of cross&compress unit and conceptually explains its strong empirical performance as compared to baselines.

3.2.1 Factorization machines. Factorization machines [22, 23] are a generic method for recommender systems. Given an input feature vector, FMs model all interactions between variables in the input vector using factorized parameters, thus being able to estimate interactions in problems with huge sparsity such as recommender systems. The model equation for a 2-degree factorization machine is defined as

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d} \sum_{j=i+1}^{d} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j, \qquad (10)$$

where x_i is the i-th unit of input vector $\mathbf{x}$, $w_\cdot$ is weight scalar, $\mathbf{v}_\cdot$ is weight vector, and $\langle \cdot, \cdot \rangle$ is dot product of two vectors. We show that the essence of FM is conceptually similar to an 1-layer cross&compress unit:

Proposition 1. *The L1-norm of $\mathbf{v}_1$ and $\mathbf{e}_1$ can be written as the following form:*

$$\|\mathbf{v}_1\|_1 \ (or \ \|\mathbf{e}_1\|_1) = \left| b + \sum_{i=1}^{d} \sum_{j=1}^{d} \langle w_i, w_j \rangle v_i e_j \right|, \qquad (11)$$

where $\langle w_i, w_j \rangle = w_i + w_j$ is the sum of two scalars.

It is interesting to notice that, instead of factorizing the weight parameter of $x_i x_j$ into the dot product of two vectors as in FM, the weight of term $v_i e_j$ is factorized into the sum of two scalars in cross&compress unit to reduce the number of parameters and increase robustness of the model.

3.2.2 Deep&Cross Network. DCN [34] learns explicit and high-order cross features by introducing the layers:

$$\mathbf{x}_{l+1} = \mathbf{x}_0 \mathbf{x}_l^\top \mathbf{w}_l + \mathbf{x}_l + \mathbf{b}_l, \qquad (12)$$

where $\mathbf{x}_l$, $\mathbf{w}_l$, and $\mathbf{b}_l$ are representation, weight, and bias of the l-th layer. We demonstrate the link between DCN and MKR by the following proposition:

Proposition 2. *In the formula of $\mathbf{v}_{l+1}$ in Eq. (2), if we restrict $\mathbf{w}_l^{VV}$ in the first term to satisfy $\mathbf{e}_l^\top \mathbf{w}_l^{VV} = 1$ and restrict $\mathbf{e}_l$ in the second term to be $\mathbf{e}_0$ (and impose similar restrictions on $\mathbf{e}_{l+1}$), the cross&compress unit is then conceptually equivalent to DCN layer in the sense of multi-task learning:*

$$\begin{aligned} \mathbf{v}_{l+1} &= \mathbf{e}_0 \mathbf{v}_l^\top \mathbf{w}_l^{EV} + \mathbf{v}_l + \mathbf{b}_l^V, \\ \mathbf{e}_{l+1} &= \mathbf{v}_0 \mathbf{e}_l^\top \mathbf{w}_l^{VE} + \mathbf{e}_l + \mathbf{b}_l^E. \end{aligned} \qquad (13)$$

It can be proven that the polynomial approximation ability of the above DCN-equivalent version (i.e., the maximal degree of cross terms in $\mathbf{v}_l$ and $\mathbf{e}_l$) is $O(l)$, which is weaker than original cross&compress units with $O(2^l)$ approximation ability.

3.2.3 Cross-stitch Networks. Cross-stitch networks [18] is a multi-task learning model in convolutional networks, in which the designed cross-stitch unit can learn a combination of shared and task-specific representations between two tasks. Specifically, given two activation maps x_A and x_B from layer l for both the tasks, cross-stitch networks learn linear combinations $\tilde{x}_A$ and $\tilde{x}_B$ of both the input activations and feed these combinations as input to the next layers' filters. The formula at location (i, j) in the activation map is

$$\begin{bmatrix} \tilde{x}_A^{ij} \\ \tilde{x}_B^{ij} \end{bmatrix} = \begin{bmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{bmatrix} \begin{bmatrix} x_A^{ij} \\ x_B^{ij} \end{bmatrix}, \qquad (14)$$

where α's are trainable transfer weights of representations between task A and task B. We show that the cross-stitch unit in Eq. (14) is a simplified version of our cross&compress unit by the following proposition:

Proposition 3. *If we omit all biases in Eq. (2), the cross&compress unit can be written as*

$$\begin{bmatrix} \mathbf{v}_{l+1} \\ \mathbf{e}_{l+1} \end{bmatrix} = \begin{bmatrix} \mathbf{e}_l^\top \mathbf{w}_l^{VV} & \mathbf{v}_l^\top \mathbf{w}_l^{EV} \\ \mathbf{e}_l^\top \mathbf{w}_l^{VE} & \mathbf{v}_l^\top \mathbf{w}_l^{EE} \end{bmatrix} \begin{bmatrix} \mathbf{v}_l \\ \mathbf{e}_l \end{bmatrix}. \qquad (15)$$

The transfer matrix in Eq. (15) serves as the cross-stitch unit $[\alpha_{AA}\ \alpha_{AB};\ \alpha_{BA}\ \alpha_{BB}]$ in Eq. (14). Like cross-stitch networks, MKR network can decide to make certain layers task specific by setting $\mathbf{v}_l^\top \mathbf{w}_l^{EV}$ (α_{AB}) or $\mathbf{e}_l^\top \mathbf{w}_l^{VE}$ (α_{BA}) to zero, or choose a more shared representation by assigning a higher value to them. But the transfer matrix is more fine-grained in cross&compress unit, because the transfer weights are replaced from scalars to dot products of two vectors. It is rather interesting to notice that Eq. (15) can also be regarded as an *attention mechanism* [1], as the computation of transfer weights involves the feature vectors $\mathbf{v}_l$ and $\mathbf{e}_l$ themselves.

4 EXPERIMENTS

In this section, we evaluate the performance of MKR in four real-world recommendation scenarios: movie, book, music, and news[9].

4.1 Datasets

We utilize the following four datasets in our experiments:

- **MovieLens-1M**[10] is a widely used benchmark dataset in movie recommendations, which consists of approximately 1 million explicit ratings (ranging from 1 to 5) on the MovieLens website.
- **Book-Crossing**[11] dataset contains 1,149,780 explicit ratings (ranging from 0 to 10) of books in the Book-Crossing community.
- **Last.FM**[12] dataset contains musician listening information from a set of 2 thousand users from Last.fm online music system.

[9]The source code is available at https://github.com/hwwang55/MKR.
[10]https://grouplens.org/datasets/movielens/1m/
[11]http://www2.informatik.uni-freiburg.de/~cziegler/BX/
[12]https://grouplens.org/datasets/hetrec-2011/

Table 1: Basic statistics and hyper-parameter settings for the four datasets.

Dataset	# users	# items	# interactions	# KG triples	Hyper-parameters
MovieLens-1M	6,036	2,347	753,772	20,195	$L = 1$, $d = 8$, $t = 3$, $\lambda_1 = 0.5$
Book-Crossing	17,860	14,910	139,746	19,793	$L = 1$, $d = 8$, $t = 2$, $\lambda_1 = 0.1$
Last.FM	1,872	3,846	42,346	15,518	$L = 2$, $d = 4$, $t = 2$, $\lambda_1 = 0.1$
Bing-News	141,487	535,145	1,025,192	1,545,217	$L = 3$, $d = 16$, $t = 5$, $\lambda_1 = 0.2$

- **Bing-News** dataset contains 1,025,192 pieces of implicit feedback collected from the server logs of Bing News[13] from October 16, 2016 to August 11, 2017. Each piece of news has a title and a snippet.

Since MovieLens-1M, Book-Crossing, and Last.FM are explicit feedback data (Last.FM provides the listening count as weight for each user-item interaction), we transform them into implicit feedback where each entry is marked with 1 indicating that the user has rated the item positively, and sample an unwatched set marked as 0 for each user. The threshold of positive rating is 4 for MovieLens-1M, while no threshold is set for Book-Crossing and Last.FM due to their sparsity.

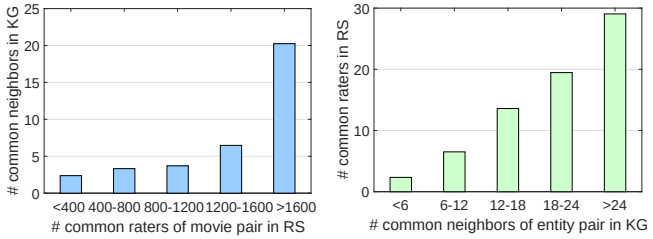
We use Microsoft Satori to construct the KG for each dataset. We first select a subset of triples from the whole KG with a confidence level greater than 0.9. For MovieLens-1M and Book-Crossing, we additionally select a subset of triples from the sub-KG whose relation name contains "film" or "book" respectively to further reduce KG size.

Given the sub-KGs, for MovieLens-1M, Book-Crossing, and Last.FM, we collect IDs of all valid movies, books, or musicians by matching their names with tail of triples (*head, film.film.name, tail*), *(head, book.book.title, tail)*, or *(head, type.object.name, tail)*, respectively. For simplicity, items with no matched or multiple matched entities are excluded. We then match the IDs with the head and tail of all KG triples and select all well-matched triples from the sub-KG. The constructing process is similar for Bing-News except that: (1) we use entity linking tools to extract entities in news titles; (2) we do not impose restrictions on the names of relations since the entities in news titles are not within one particular domain. The basic statistics of the four datasets are presented in Table 1. Note that the number of users, items, and interactions are smaller than original datasets since we filtered out items with no corresponding entity in the KG.

4.2 Baselines

We compare our proposed MKR with the following baselines. Unless otherwise specified, the hyper-parameter settings of baselines are the same as reported in their original papers or as default in their codes.

- **PER** [39] treats the KG as heterogeneous information networks and extracts meta-path based features to represent the connectivity between users and items. In this paper, we use manually designed user-item-attribute-item paths as features, i.e., "user-movie-director-movie", "user-movie-genre-movie", and "user-movie-star-movie" for MovieLens-20M; "user-book-author-book" and "user-book-genre-book" for Book-Crossing; "user-musician-genre-musician", "user-musician-country-musician", and "user-musician-age-musician" (age is discretized) for Last.FM. Note that PER cannot be applied to news recommendation because it's hard to pre-define meta-paths for entities in news.
- **CKE** [40] combines CF with structural, textual, and visual knowledge in a unified framework for recommendation. We implement CKE as CF plus structural knowledge module in this paper. The dimension of user and item embeddings for the four datasets are set as 64, 128, 32, 64, respectively. The dimension of entity embeddings is 32.
- **DKN** [32] treats entity embedding and word embedding as multiple channels and combines them together in CNN for CTR prediction. In this paper, we use movie/book names and news titles as textual input for DKN. The dimension of word embedding and entity embedding is 64, and the number of filters is 128 for each window size 1, 2, 3.
- **RippleNet** [31] is a memory-network-like approach that propagates usersâĂŹ preferences on the knowledge graph for recommendation. The hyper-parameter settings for Last.FM are $d = 8$, $H = 2$, $\lambda_1 = 10^{-6}$, $\lambda_2 = 0.01$, $\eta = 0.02$.
- **LibFM** [23] is a widely used feature-based factorization model. We concatenate the raw features of users and items as well as the corresponding averaged entity embeddings learned from TransR [13] as input for LibFM. The dimension is {1, 1, 8} and the number of training epochs is 50. The dimension of TransR is 32.
- **Wide&Deep** [3] is a deep recommendation model combining a (wide) linear channel with a (deep) nonlinear channel. The input for Wide&Deep is the same as in LibFM. The dimension of user, item, and entity is 64, and we use a two-layer deep channel with dimension of 100 and 50 as well as a wide channel.



(a) RS to KG **(b) KG to RS**

Figure 2: The correlation between the number of common neighbors of an item pair in KG and their number of common raters in RS.

[13] https://www.bing.com/news

Table 2: The results of *AUC* and *Accuracy* in CTR prediction.

Model	MovieLens-1M		Book-Crossing		Last.FM		Bing-News	
	AUC	*ACC*	*AUC*	*ACC*	*AUC*	*ACC*	*AUC*	*ACC*
PER	0.710 (-22.6%)	0.664 (-21.2%)	0.623 (-15.1%)	0.588 (-16.7%)	0.633 (-20.6%)	0.596 (-20.7%)	-	-
CKE	0.801 (-12.6%)	0.742 (-12.0%)	0.671 (-8.6%)	0.633 (-10.3%)	0.744 (-6.6%)	0.673 (-10.5%)	0.553 (-19.7%)	0.516 (-20.0%)
DKN	0.655 (-28.6%)	0.589 (-30.1%)	0.622 (-15.3%)	0.598 (-15.3%)	0.602 (-24.5%)	0.581 (-22.7%)	0.667 (-3.2%)	0.610 (-5.4%)
RippleNet	**0.920** (+0.3%)	0.842 (-0.1%)	0.729 (-0.7%)	0.662 (-6.2%)	0.768 (-3.6%)	0.691 (-8.1%)	0.678 (-1.6%)	0.630 (-2.3%)
LibFM	0.892 (-2.7%)	0.812 (-3.7%)	0.685 (-6.7%)	0.640 (-9.3%)	0.777 (-2.5%)	0.709 (-5.7%)	0.640 (-7.1%)	0.591 (-8.4%)
Wide&Deep	0.898 (-2.1%)	0.820 (-2.7%)	0.712 (-3.0%)	0.624 (-11.6%)	0.756 (-5.1%)	0.688 (-8.5%)	0.651 (-5.5%)	0.597 (-7.4%)
MKR	0.917	**0.843**	**0.734**	**0.704**	**0.797**	**0.752**	**0.689**	**0.645**
MKR-1L	-	-	-	-	0.795 (-0.3%)	0.749 (-0.4%)	0.680 (-1.3%)	0.631 (-2.2%)
MKR-DCN	0.883 (-3.7%)	0.802 (-4.9%)	0.705 (-4.3%)	0.676 (-4.2%)	0.778 (-2.4%)	0.730 (-2.9%)	0.671 (-2.6%)	0.614 (-4.8%)
MKR-stitch	0.905 (-1.3%)	0.830 (-1.5%)	0.721 (-2.2%)	0.682 (-3.4%)	0.772 (-3.1%)	0.725 (-3.6%)	0.674 (-2.2%)	0.621 (-3.7%)

4.3 Experiments setup

In MKR, we set the number of high-level layers $K = 1$, f_{RS} as inner product, and $\lambda_2 = 10^{-6}$ for all three datasets, and other hyper-parameter are given in Table 1. The settings of hyper-parameters are determined by optimizing *AUC* on a validation set. For each dataset, the ratio of training, validation, and test set is $6:2:2$. Each experiment is repeated 3 times, and the average performance is reported. We evaluate our method in two experiment scenarios: (1) In click-through rate (CTR) prediction, we apply the trained model to each piece of interactions in the test set and output the predicted click probability. We use *AUC* and *Accuracy* to evaluate the performance of CTR prediction. (2) In top-K recommendation, we use the trained model to select K items with highest predicted click probability for each user in the test set, and choose *Precision@K* and *Recall@K* to evaluate the recommended sets.

4.4 Empirical study

We conduct an empirical study to investigate the correlation of items in RS and their corresponding entities in KG. Specifically, we aim to reveal how the number of common neighbors of an item pair in KG changes with their number of common raters in RS. To this end, we first randomly sample 1 million item pairs from MovieLens-1M. We then classify each pair into 5 categories based on the number of their common raters in RS, and count their average number of common neighbors in KG for each category. The result is presented in Figure 2a, which clearly shows that *if two items have more common raters in RS, they are likely to share more common neighbors in KG*. Figure 2b shows the positive correlation from an opposite direction. The above findings empirically demonstrate that *items share the similar structure of proximity in KG and RS*, thus the cross knowledge transfer of items benefits both recommendation and KGE tasks in MKR.

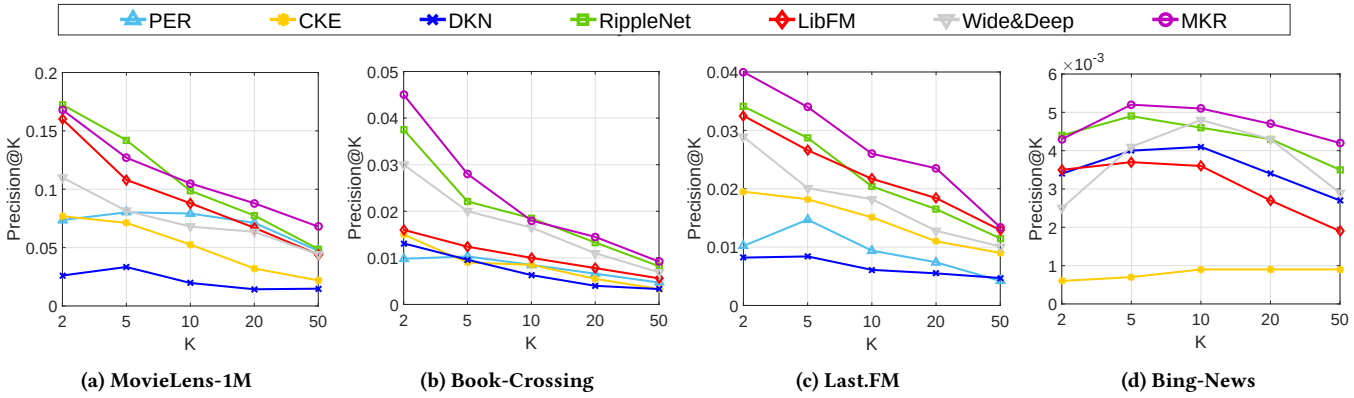
4.5 Results

4.5.1 Comparison with baselines. The results of all methods in CTR prediction and top-K recommendation are presented in Table 2 and Figure 3, 4, respectively, from which we have the following observations:

- PER performs poor on movie, book, and music recommendation because the user-defined meta-paths can hardly be optimal in reality. Moreover, PER cannot be applied to news recommendation since entities in news titles are not within one particular domain.
- CKE performs better in movie, book, and music recommendation than news. This may be because MovieLens-1M, Book-Crossing, and Last.FM are much denser than Bing-News, which is more favorable for the collaborative filtering part in CKE.
- DKN performs best in news recommendation compared with other baselines, but performs worst in other scenarios. This is because movie, book, and musician names are too short and ambiguous to provide useful information.
- RippleNet performs best among all baselines, and even outperforms MKR on MovieLens-1M. This demonstrates that RippleNet can precisely capture user interests, especially in the case where user-item interactions are dense. However, RippleNet is more sensitive to the density of datasets, as it performs worse than MKR in Book-Crossing, Last.FM, and Bing-News. We will further study their performance in sparse scenarios in Section 4.5.3.
- In general, our MKR performs best among all methods on the four datasets. Specifically, MKR achieves average *Accuracy* gains of 11.6%, 11.5%, 12.7%, and 8.7% in movie, book, music, and news recommendation, respectively, which demonstrates the efficacy of the multi-task learning framework in MKR. Note that the top-K metrics are much lower for Bing-News because the number of news is significantly larger than movies, books, and musicians.

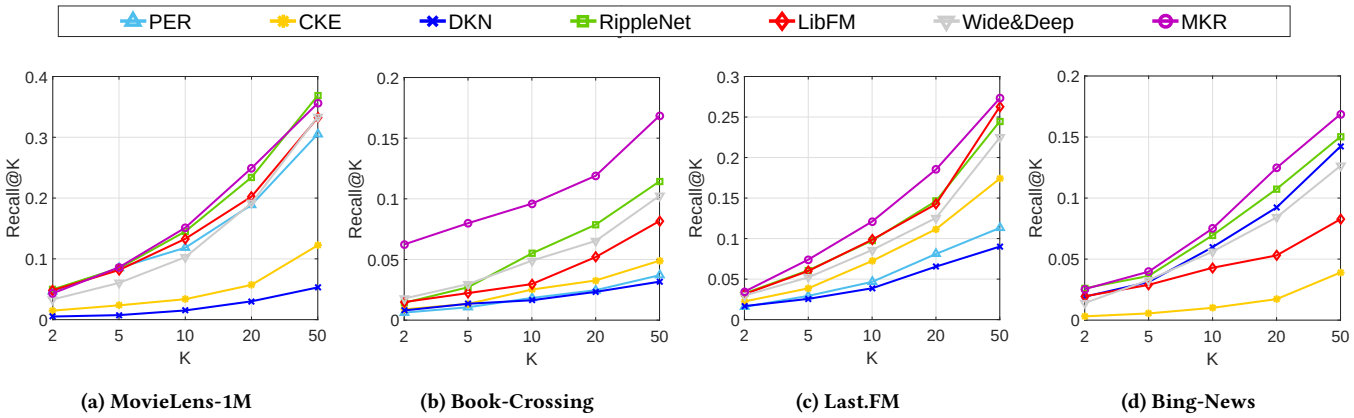
4.5.2 Comparison with MKR variants. We further compare MKR with its three variants to demonstrate the efficacy of cross&compress unit:

- MKR-1L is MKR with one layer of cross&compress unit, which corresponds to FM model according to Proposition 1. Note that MKR-1L is actually MKR in the experiments for MovieLens-1M.
- MKR-DCN is a variant of MKR based on Eq. (13), which corresponds to DCN model.
- MKR-stitch is another variant of MKR corresponding to the cross-stitch network, in which the transfer weights in Eq. (15) are replaced by four trainable scalars.



(a) MovieLens-1M (b) Book-Crossing (c) Last.FM (d) Bing-News

Figure 3: The results of *Precision@K* in top-*K* recommendation.



(a) MovieLens-1M (b) Book-Crossing (c) Last.FM (d) Bing-News

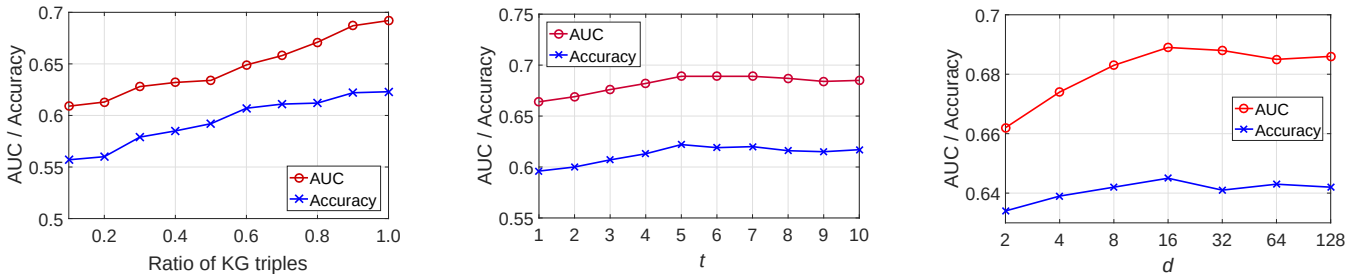
Figure 4: The results of *Recall@K* in top-*K* recommendation.

Table 3: Results of *AUC* on MovieLens-1M in CTR prediction with different ratios of training set *r*.

Model	*r*									
	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
PER	0.598	0.607	0.621	0.638	0.647	0.662	0.675	0.688	0.697	0.710
CKE	0.674	0.692	0.705	0.716	0.739	0.754	0.768	0.775	0.797	0.801
DKN	0.579	0.582	0.589	0.601	0.612	0.620	0.631	0.638	0.646	0.655
RippleNet	0.843	0.851	0.859	0.862	0.870	0.878	0.890	0.901	0.912	**0.920**
LibFM	0.801	0.810	0.816	0.829	0.837	0.850	0.864	0.875	0.886	0.892
Wide&Deep	0.788	0.802	0.809	0.815	0.821	0.840	0.858	0.876	0.884	0.898
MKR	**0.868**	**0.874**	**0.881**	**0.882**	**0.889**	**0.897**	**0.903**	**0.908**	**0.913**	0.917

From Table 2 we observe that MKR outperforms MKR-1L and MKR-DCN, which shows that modeling high-order interactions between item and entity features is helpful for maintaining decent performance. MKR also achieves better scores than MKR-stitch. This validates the efficacy of fine-grained control on knowledge transfer in MKR compared with the simple cross-stitch units.

4.5.3 Results in sparse scenarios. One major goal of using knowledge graph in MKR is to alleviate the sparsity and the cold start problem of recommender systems. To investigate the efficacy of the KGE module in sparse scenarios, we vary the ratio of training set of MovieLens-1M from 100% to 10% (while the validation and test set are kept fixed), and report the results of *AUC* in CTR prediction for all methods. The results are shown in Table 3. We observe that the performance of all methods deteriorates with the reduce of the training set. When $r = 10\%$, the *AUC* score decreases by 15.8%, 15.9%, 11.6%, 8.4%, 10.2%, 12.2% for PER, CKE, DKN, RippleNet, LibFM, and Wide&Deep, respectively, compared with



(a) KG size (b) Training frequency (c) Dimension of embeddings

Figure 5: Parameter sensitivity of MKR on Bing-News w.r.t. (a) the size of the knowledge graph; (b) training frequency of the RS module t; and (c) dimension of embeddings d.

Table 4: The results of *RMSE* on the KGE module for the three datasets. "KGE" means only KGE module is trained, while "KGE + RS" means KGE module and RS module are trained together.

dataset	KGE	KGE + RS
MovieLens-1M	0.319	**0.302**
Book-Crossing	0.596	**0.558**
Last.FM	0.480	**0.471**
Bing-News	0.488	**0.459**

the case when full training set is used ($r = 100\%$). In contrast, the *AUC* score of MKR only decreases by 5.3%, which demonstrates that MKR can still maintain a decent performance even when the user-item interaction is sparse. We also notice that MKR performs better than RippleNet in sparse scenarios, which is accordance with our observation in Section 4.5.1 that RippleNet is more sensitive to the density of user-item interactions.

4.5.4 Results on KGE side. Although the goal of MKR is to utilize KG to assist with recommendation, it is still interesting to investigate whether the RS task benefits the KGE task, since the principle of multi-task learning is to leverage shared information to help improve the performance of all tasks [42]. We present the result of *RMSE* (rooted mean square error) between predicted and real vectors of tails in the KGE task in Table 4. Fortunately, we find that the existence of RS module can indeed reduce the prediction error by 1.9% $\sim$ 6.4%. The results show that the cross&compress units are able to learn general and shared features that mutually benefit both sides of MKR.

4.6 Parameter Sensitivity

4.6.1 Impact of KG size. We vary the size of KG to further investigate the efficacy of usage of KG. The results of *AUC* on Bing-News are plotted in Figure 5a. Specifically, the *AUC* and *Accuracy* is enhanced by 13.6% and 11.8% with the KG ratio increasing from 0.1 to 1.0 in three scenarios, respectively. This is because the Bing-News dataset is extremely sparse, making the effect of KG usage rather obvious.

4.6.2 Impact of RS training frequency. We investigate the influence of parameters t in MKR by varying t from 1 to 10, while keeping other parameters fixed. The results are presented in Figure 5b. We observe that MKR achieves the best performance when $t = 5$. This is because a high training frequency of the KGE module will mislead the objective function of MKR, while too small of a training frequency of KGE cannot make full use of the transferred knowledge from the KG.

4.6.3 Impact of embedding dimension. We also show how the dimension of users, items, and entities affects the performance of MKR in Figure 5c. We find that the performance is initially improved with the increase of dimension, because more bits in embedding layer can encode more useful information. However, the performance drops when the dimension further increases, as too large number of dimensions may introduce noises which mislead the subsequent prediction.

5 RELATED WORK

5.1 Knowledge Graph Embedding

The KGE module in MKR connects to a large body of work in KGE methods. KGE is used to embed entities and relations in a knowledge into low-dimensional vector spaces while still preserving the structural information [33]. KGE methods can be classified into the following two categories: (1) Translational distance models exploit distance-based scoring functions when learning representations of entities and relations, such as TransE [2], TransH [35], and TransR [13]; (2) Semantic matching models measure plausibility of knowledge triples by matching latent semantics of entities and relations, such as RESCAL [20], ANALOGY [19], and HolE [14]. Recently, researchers also propose incorporating auxiliary information, such as entity types [36], logic rules [24], and textual descriptions [46] to assist KGE. The above KGE methods can also be incorporated into MKR as the implementation of the KGE module, but note that the cross&compress unit in MKR needs to be redesigned accordingly. Exploring other designs of KGE module as well as the corresponding bridging unit is also an important direction of future work.

5.2 Multi-Task Learning

Multi-task learning is a learning paradigm in machine learning and its aim is to leverage useful information contained in multiple related tasks to help improve the generalization performance of all the tasks [42]. All of the learning tasks are assumed to be related to

each other, and it is found that learning these tasks jointly can lead to performance improvement compared with learning them individually. In general, MTL algorithms can be classified into several categories, including feature learning approach [34, 41], low-rank approach [7, 16], task clustering approach [47], task relation learning approach [12], and decomposition approach [6]. For example, the cross-stitch network [41] determines the inputs of hidden layers in different tasks by a knowledge transfer matrix; Zhou et. al [47] aims to cluster tasks by identifying representative tasks which are a subset of the given m tasks, i.e., if task T_i is selected by task T_j as a representative task, then it is expected that model parameters for T_j are similar to those of T_i. MTL can also be combined with other learning paradigms to improve the performance of learning tasks further, including semi-supervised learning, active learning, unsupervised learning,and reinforcement learning.

Our work can be seen as an asymmetric multi-task learning framework [37, 43, 44], in which we aim to utilize the connection between RS and KG to help improve their performance, and the two tasks are trained with different frequencies.

5.3 Deep Recommender Systems

Recently, deep learning has been revolutionizing recommender systems and achieves better performance in many recommendation scenarios. Roughly speaking, deep recommender systems can be classified into two categories: (1) Using deep neural networks to process the raw features of users or items [5, 28–30, 40]; For example, Collaborative Deep Learning [29] designs autoencoders to extract short and dense features from textual input and feeds the features into a collaborative filtering module; DeepFM [5] combines factorization machines for recommendation and deep learning for feature learning in a neural network architecture. (2) Using deep neural networks to model the interaction among users and items [3, 4, 8, 9]. For example, Neural Collaborative Filtering [8] replaces the inner product with a neural architecture to model the user-item interaction. The major difference between these methods and ours is that MKR deploys a multi-task learning framework that utilizes the knowledge from a KG to assist recommendation.

6 CONCLUSIONS AND FUTURE WORK

This paper proposes MKR, a multi-task learning approach for knowledge graph enhanced recommendation. MKR is a deep and end-to-end framework that consists of two parts: the recommendation module and the KGE module. Both modules adopt multiple nonlinear layers to extract latent features from inputs and fit the complicated interactions of user-item and head-relation pairs. Since the two tasks are not independent but connected by items and entities, we design a cross&compress unit in MKR to associate the two tasks, which can automatically learn high-order interactions of item and entity features and transfer knowledge between the two tasks. We conduct extensive experiments in four recommendation scenarios. The results demonstrate the significant superiority of MKR over strong baselines and the efficacy of the usage of KG.

For future work, we plan to investigate other types of neural networks (such as CNN) in MKR framework. We will also incorporate other KGE methods as the implementation of KGE module in MKR by redesigning the cross&compress unit.

APPENDIX

A Proof of Theorem 1

Proof. We prove the theorem by induction:

Base case: When $l = 1$,

$$\begin{aligned}\mathbf{v}_1 =& \mathbf{v}\mathbf{e}^\top \mathbf{w}_0^{VV} + \mathbf{e}\mathbf{v}^\top \mathbf{w}_0^{EV} + \mathbf{b}_0^V \\ =& \left[v_1 \sum_{i=1}^{d} e_i w_0^{VV(i)} \ \cdots \ v_d \sum_{i=1}^{d} e_i w_0^{VV(i)} \right]^\top \\ &+ \left[e_1 \sum_{i=1}^{d} v_i w_0^{EV(i)} \ \cdots \ e_d \sum_{i=1}^{d} v_i w_0^{EV(i)} \right]^\top \\ &+ \left[b_0^{V(0)} \ \cdots \ b_0^{V(d)} \right]^\top .\end{aligned}$$

Therefore, we have

$$\begin{aligned}\|\mathbf{v}_1\|_1 &= \left| \sum_{j=1}^{d} v_j \sum_{i=1}^{d} e_i w_0^{VV(i)} + \sum_{j=1}^{d} e_j \sum_{i=1}^{d} v_i w_0^{EV(i)} + \sum_{i=1}^{d} b_0^{V(d)} \right| \\ &= \left| \sum_{i=1}^{d} \sum_{j=1}^{d} (w_0^{EV(i)} + w_0^{VV(j)}) v_i e_j + \sum_{i=1}^{d} b_0^{V(d)} \right|.\end{aligned}$$

It is clear that the cross terms about $\mathbf{v}$ and $\mathbf{e}$ with maximal degree is $k_{\boldsymbol{\alpha},\boldsymbol{\beta}} v_i e_j$, so we have $\alpha_1 + \cdots + \alpha_d = 1 = 2^{1-1}$, and $\beta_1 + \cdots + \beta_d = 1 = 2^{1-1}$ for $\mathbf{v}_1$. The proof for $\mathbf{e}_1$ is similar.

Induction step: Suppose $\alpha_1 + \cdots + \alpha_d = 2^{l-1}$ and $\beta_1 + \cdots + \beta_d = 2^{l-1}$ hold for the maximal-degree term x and y in $\|\mathbf{v}_l\|_1$ and $\|\mathbf{e}_l\|_1$. Since $\|\mathbf{v}_l\|_1 = \left|\sum_{i=1}^{d} v_l^{(i)}\right|$ and $\|\mathbf{e}_l\|_1 = \left|\sum_{i=1}^{d} e_l^{(i)}\right|$, without loss of generosity, we assume that x and y exist in $v_l^{(a)}$ and $e_l^{(b)}$, respectively. Then for $l + 1$, we have

$$\|\mathbf{v}_{l+1}\|_1 = \sum_{i=1}^{d} \sum_{j=1}^{d} (w_l^{EV(i)} + w_l^{VV(j)}) v_l^{(i)} e_l^{(j)} + \sum_{i=1}^{d} b_l^{V(d)}.$$

Obviously, the maximal-degree term in $\|\mathbf{v}_{l+1}\|_1$ is the cross term xy in $v_l^{(a)} e_l^{(b)}$. Since we have $\alpha_1 + \cdots + \alpha_d = 2^{l-1}$ and $\beta_1 + \cdots + \beta_d = 2^{l-1}$ for both x and y, the degree of cross term xy therefore satisfies $\alpha_1 + \cdots + \alpha_d = 2^{(l+1)-1}$ and $\beta_1 + \cdots + \beta_d = 2^{(l+1)-1}$. The proof for $\|\mathbf{e}_{l+1}\|_1$ is similar. □

B Proof of Proposition 1

Proof. In the proof of Theorem 1 in Appendix A, we have shown that

$$\|\mathbf{v}_1\|_1 = \left| \sum_{i=1}^{d} \sum_{j=1}^{d} (w_0^{EV(i)} + w_0^{VV(j)}) v_i e_j + \sum_{i=1}^{d} b_0^{V(d)} \right|.$$

It is easy to see that $w_i = w_0^{EV(i)}$, $w_j = w_0^{VV(j)}$, and $b = \sum_{i=1}^{d} b_0^{V(d)}$. The proof is similar for $\|\mathbf{e}_1\|_1$. □

We omit the proofs for Proposition 2 and Proposition 3 as they are straightforward.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*. 2787–2795.

[3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 7–10.

[4] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.

[5] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*.

[6] Lei Han and Yu Zhang. 2015. Learning tree structure in multi-task learning. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 397–406.

[7] Lei Han and Yu Zhang. 2016. Multi-Stage Multi-Task Learning with Reduced Rank.. In *AAAI*. 1638–1644.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.

[9] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *CIKM*. ACM, 2333–2338.

[10] Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the 4th ACM conference on Recommender systems*. ACM, 135–142.

[11] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).

[12] Giwoong Lee, Eunho Yang, and Sung Hwang. 2016. Asymmetric multi-task learning based on task relatedness and loss. In *International Conference on Machine Learning*. 230–238.

[13] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion.. In *The 29th AAAI Conference on Artificial Intelligence*. 2181–2187.

[14] Hanxiao Liu, Yuexin Wu, and Yiming Yang. 2017. Analogical Inference for Multi-Relational Embeddings. In *Proceedings of the 34th International Conference on Machine Learning*. 2168–2178.

[15] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and S Yu Philip. 2017. Learning Multiple Tasks with Multilinear Relationship Networks. In *Advances in Neural Information Processing Systems*. 1593–1602.

[16] Andrew M McDonald, Massimiliano Pontil, and Dimitris Stamos. 2014. Spectral k-support norm regularization. In *Advances in Neural Information Processing Systems*. 3644–3652.

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*. 3111–3119.

[18] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3994–4003.

[19] Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, et al. 2016. Holographic Embeddings of Knowledge Graphs.. In *The 30th AAAI Conference on Artificial Intelligence*. 1955–1961.

[20] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In *Proceedings of the 28th International Conference on Machine Learning*. 809–816.

[21] Sinno Jialin Pan, Qiang Yang, et al. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[22] Steffen Rendle. 2010. Factorization machines. In *Proceedings of the 10th IEEE International Conference on Data Mining*. IEEE, 995–1000.

[23] Steffen Rendle. 2012. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3, 3 (2012), 57.

[24] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. 2015. Injecting logical background knowledge into embeddings for relation extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1119–1129.

[25] Walter Rudin et al. 1964. *Principles of mathematical analysis*. Vol. 3. McGraw-hill New York.

[26] Jie Tang, Sen Wu, Jimeng Sun, and Hang Su. 2012. Cross-domain collaboration recommendation. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1285–1293.

[27] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Graphgan: Graph representation learning with generative adversarial nets. In *AAAI*. 2508–2515.

[28] Hongwei Wang, Jia Wang, Miao Zhao, Jiannong Cao, and Minyi Guo. 2017. Joint Topic-Semantic-aware Social Recommendation for Online Voting. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 347–356.

[29] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[30] Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. 2018. Shine: Signed heterogeneous information network embedding for sentiment link prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 592–600.

[31] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM.

[32] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep Knowledge-Aware Network for News Recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1835–1844.

[33] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.

[34] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & Cross Network for Ad Click Predictions. In *Proceedings of the ADKDD'17*. ACM, 12.

[35] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph and text jointly embedding. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1591–1601.

[36] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2016. Representation Learning of Knowledge Graphs with Hierarchical Types.. In *IJCAI*. 2965–2971.

[37] Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. 2007. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research* 8, Jan (2007), 35–63.

[38] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks?. In *Advances in Neural Information Processing Systems*. 3320–3328.

[39] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*. 283–292.

[40] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 353–362.

[41] Wenlu Zhang, Rongjian Li, Tao Zeng, Qian Sun, Sudhir Kumar, Jieping Ye, and Shuiwang Ji. 2015. Deep model based transfer and multi-task learning for biological image analysis. In *21st ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2015*. Association for Computing Machinery.

[42] Yu Zhang and Qiang Yang. 2017. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114* (2017).

[43] Yu Zhang and Dit-Yan Yeung. 2012. A convex formulation for learning task relationships in multi-task learning. *arXiv preprint arXiv:1203.3536* (2012).

[44] Yu Zhang and Dit-Yan Yeung. 2014. A regularization approach to learning task relationships in multitask learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 8, 3 (2014), 12.

[45] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-graph based recommendation fusion over heterogeneous information networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 635–644.

[46] Huaping Zhong, Jianwen Zhang, Zhen Wang, Hai Wan, and Zheng Chen. 2015. Aligning knowledge and text embeddings by entity descriptions. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 267–272.

[47] Qiang Zhou and Qi Zhao. 2016. Flexible Clustered Multi-Task Learning by Learning Representative Tasks. *IEEE Trans. Pattern Anal. Mach. Intell.* 38, 2 (2016), 266–278.

Towards Efficient Sharing: A Usage Balancing Mechanism for Bike Sharing Systems

Shuai Wang
Key Lab of Computer Network and Information Integration
Southeast University
shuaiwang@seu.edu.cn

Tian He
University of Minnesota
tianhe@umn.edu

Desheng Zhang
Rutgers University
desheng.zhang@cs.rutgers.edu

Yunhuai Liu
Peking University
yunhuai.liu@gmail.com

Sang H. Son
Daegu Gyeongbuk Institute of Science and Technology
don@dgist.ac.kr

ABSTRACT

With the rapid development of sharing economy, massive sharing systems such as Uber, Airbnb, and bikeshare have percolated into people's daily life. The sharing economy, at its core, is to achieve efficient use of resources. The actual usage of shared resources, however, is unclear to us. Little measurement or analysis, if any, has been conducted to investigate the resource usage status with the large-scale data collected from these sharing systems. In this paper, we analyze the bike usage status in three typical bikeshare systems based on 140-month multi-event data. Our analysis shows that the most used 20% of bikes account for 45% of usage, while the least used 20% of bikes account for less than 1% of usage. To efficiently utilize shared bikes, we propose a usage balancing design called eShare which has three components: (i) a statistical model based on archived data to infer historical usage; (ii) an entropy-based prediction model based on both real-time and archived data to infer future usage; (iii) a model-driven optimal calibration engine for bike selection to dynamically balance usage. We develop an ID swapping based evaluation methodology and measure the efficiency of eShare with data from three systems including the world's largest bikeshare system with 84,000 bikes and 3,300 stations. Our results show that eShare not only fully utilizes shared bikes but also improves service quality.

CCS CONCEPTS

• **Human-centered computing → Ubiquitous and mobile computing**.

KEYWORDS

Bike sharing; usage balancing; sharing economy; urban data

ACM Reference Format:
Shuai Wang, Tian He, Desheng Zhang, Yunhuai Liu, and Sang H. Son. 2019. Towards Efficient Sharing: A Usage Balancing Mechanism for Bike Sharing Systems. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313441

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW'19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313441

1 INTRODUCTION

Bike sharing is widely adopted in major cities around the world as a new way of travel [7]. In a bikeshare system, people pick up a bike from one station and drop off it at another station, providing great convenience to people's daily commute. As of June 2014, bikeshare systems have been launched in 712 cities, operating approximately 806,200 bicycles at 37,500 stations [22].

As a sharing platform, one of the key concerns is to efficiently utilize the shared bike resources. However, our data analysis on three representative bikeshare systems shows that inefficient bike usage is a common phenomenon in existing bikeshare systems. For example, the statistic results on NiceRide bikeshare system show that the most used 20% of bikes account for 45% of the total rentals, while the least used 20% of bikes account for less than 1% of the total rentals. This is because bikes in current systems are arbitrarily assigned to users, ignoring both cumulative service time of bikes and potential rental duration of users.

The current bike sharing mechanism has several drawbacks. First, a large amount of bikes in the system are rarely utilized but continually introducing maintenance costs while the unsatisfied status of the frequently used bikes leads to complaints on service quality. Second, the irregular or excessive bike usage contributes to bike depreciation, leading to a huge amount of damaged bikes. Per the statistics of Capital bikeshare system, 9,284 bikes are damaged because of heavy riding from Nov 2014 to Oct 2015, which costs about 3 million US dollars in inspecting/repairing those bikes [5]. In addition, massive damaged bikes appear in busy seasons (e.g., from May to August in Capital bikeshare) and operators are too busy to inspect/repair them, which further reduce the service quality.

To overcome these drawbacks, a straightforward strategy is to manage bikes to be balanced used. This strategy helps existing bikeshare systems make full use of all bikes. In addition, while bikes are balanced used, we should avoid the situation that all bikes wear out at the same time which overwhelms system operators. We thus propose *batch balancing* which gracefully degrades bikes in batches so that (i) shared bikes are fully utilized and (ii) the usage

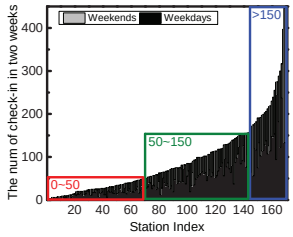


Figure 1: Demand Dist.

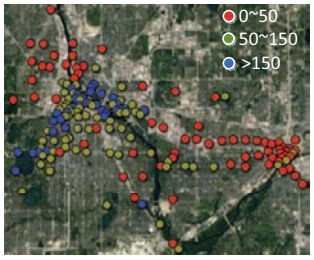


Figure 2: Demand Map

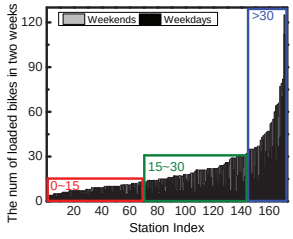


Figure 3: Service Dist.

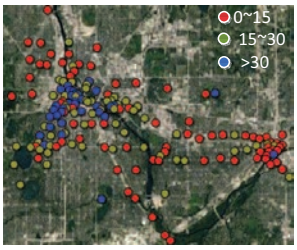


Figure 4: Service Map

status meets operators' demands (e.g., to fit the bike usage status with bike repair/upgrade schedule).

Managing shared bikes to be efficiently used seems straightforward because we can intuitively assign bikes to users based on lengths of their trips when they rent bikes. However, the key challenge is that we can only know their trip lengths when they *return* bikes, not when they *rent* bikes. As a result, naively assigning a newer bike to a passenger may increase usage unbalance because other older bikes may have to be assigned to later passengers with longer trips, and vice versa. Furthermore, it is more challenging to achieve batch balancing which has to meet operators' demands.

Although controlling the usage status is crucial in sharing platforms, most existing work is focused on demand/supply modeling *at station levels* [1, 4, 9, 15] and bike redistribution among stations [16, 17, 25]. To the best of our knowledge, little work, if any, has been conducted to investigate the usage issue *at individual bike levels*. In this paper, we present the first data-driven usage balancing mechanism called eShare to analyze, quantify and (batch) balance the shared usage status at individual bike levels based on large scale data from three bikeshare systems. Specifically, the key contributions of this paper are as follows:

• We conduct the first data analysis on usage balancing for bikeshare systems. To our knowledge, our analysis on bikeshare systems is based on the most comprehensive long-term multi-site data until now, i.e., 140-month multi-event data related to bikes (i.e., rental records, station status, rebalancing and repairs records) for 3 major cities with different system scales and population (i.e., 3.3 million, 6.1 million, and 9.1 million) in both developed and developing countries.

• Built upon the identified features, we design a data-driven usage balancing mechanism, called eShare, which consists of three technical components: (i) a statistical model based on archived data to infer historical usage; (ii) an entropy-based prediction model based on both real-time and archived data to infer future bike usage; (iii) a usage calibration engine based on these two models to select bikes for users to optimize bike usage at individual levels. The calibration engine is the core of achieving efficient usage balancing.

• We develop an ID swapping based measurement methodology and evaluate the performance of eShare through extensive data-driven experiments based on the data from three bikeshare systems including the largest bikeshare system in the world, with 84,000 bikes and 3,300 stations in Hangzhou, as well as the second largest bikeshare system in the US. with 3,700 bikes and 429 stations in Washington D.C. The evaluation results show that the standard deviation of global bike usage is reduced by 90% with our eShare design when the number of batch equals to one.

2 ANALYSIS OF SHARED USAGE

In this section, we first analyze the current usage status of shared bikes in existing systems. Then, we introduce the benefit of usage balancing in efficient sharing.

2.1 Current Bike Sharing Status

By analyzing the usage status in existing bikeshare systems, this subsection shows the need for bike usage balancing. The analysis results from three bikeshare systems in three cities show that inefficient usage is common phenomenon. The basic information about these three systems is shown in Table 1. Details of the data are in Section 3.

Unbalanced Demand: We first introduce the unbalanced demand phenomenon in existing bikeshare systems. Figure 1 plots each station's total number of rentals in two weeks in NiceRide bikeshare system. We find that (i) about 17.6% of the stations (marked as "blue " in Figure 2) are extremely busy which have more than 150 rentals in two weeks; (ii) about 40% of the stations (marked as "red" in

Table 1: Statistic of Three Bikeshare Systems

	Capital	Hangzhou	NiceRide
Launched Time	2010.9	2008.5	2010.6
Num of Stations	429	3.4×10^3	190
Num of Bikes	3,700	8.4×10^4	1,700
Ridership/Day	5.8×10^3	2.3×10^5	1.9×10^3

the demand map in Figure 2) have a low user demand, i.e., fewer than 50 rentals in two weeks. Statistically, 17.6% of the busiest stations satisfy 44.5% of the user demand. In addition, we find that the user demand at some stations varies dramatically on weekdays and weekends, e.g., the user demand at the stations near campus increases on weekdays and drops on weekends.

Unbalanced Service: To meet user demand, the operators in bikeshare systems redistribute bikes among stations. Figure 3 shows the number of bikes loaded to each station in two weeks in NiceRide bikeshare system. We find that the service level at different stations is quite different as in Figure 4: (i) some stations have a high service level with more than 30 loaded bikes (marked as "blue circle"); (ii) however, some stations have a service level with fewer than 15 loaded bikes (marked as "red circle"); (iii) most of the stations with high service levels locate in the downtown and the campus areas.

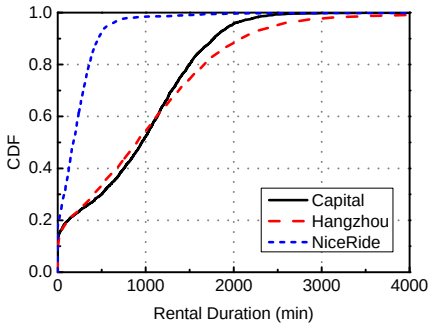


Figure 5: The CDFs of accumulated rental duration

Unbalanced Usage: We investigate the bike usage in Capital, Hangzhou, and NiceRide bikeshare systems. We find that inefficient bike usage is a common phenomenon in existing bikeshare systems. Figure 5 plots the CDF of bikes' rental durations in two weeks in the three bikeshare systems. We find that the bike usage is inefficient in all three systems. For example, in Capital bikeshare system, about 20% of bikes are used more than 1500 minutes while 20% of bikes are used less than 120 minutes in two weeks. As shown in Figure 5, similar results are found in Hangzhou and NiceRide bikeshare systems as well.

2.2 The Benefit of Bike Usage Balancing

From the above discussion, we learn that inefficient bike usage is a common phenomenon in existing bikeshare systems. Bike usage balancing helps existing bikeshare systems improve sharing efficiency. Specifically, with the help of usage balancing, users are able to use those rarely-used new bikes which improves service quality. Inefficient bike usage also causes heavy riding on a portion of bikes which leads to massive damaged bikes. In addition, compared with traditional bikes, shared bikes are designed for sharing purpose and most of them are equipped with special sensors or communication devices. These shared bikes are much more expensive than traditional bikes and need to be batch customized. Bike usage balancing has great potential to help operators achieve efficient system maintenance by letting the system-wide bike usage meet operators' repair/upgrade schedule.

Furthermore, investigating bike usage helps system operators obtain information including but not limited to (i) the current status of a bike, (ii) the residual service time of a bike, (iii) whether a bike needs to be repaired, and (iv) whether bikes are efficiently used. This information can help operators manage bikes and improve the quality of sharing service.

3 ESHARE OVERVIEW

Leveraging upon the share usage analysis results, we aim to develop a novel usage balancing mechanism for efficient sharing. In the following, we first introduce the infrastructure of bikeshare systems and the collected data. Then, we propose design objectives, followed by design overview.

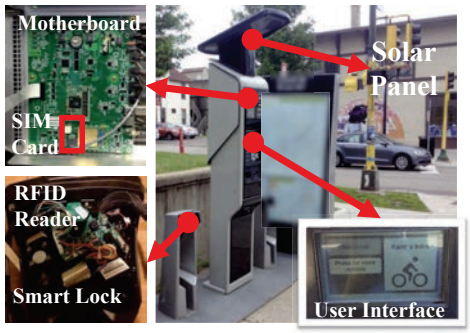


Figure 6: The infrastructure of bikeshare systems

3.1 Bikeshare Infrastructure

Leveraging experiences we gained in our project, we provide our expertise and are part of a team to design, deploy and maintain NiceRide bikeshare system as shown in Figure 6.

From the figure, we can see that passive RFID reader along with smart dock is installed in each dock in a station. When a user comes to borrow a bike, she/he needs to use her/his access key (i.e., RFID tags) to unlock a bike. Once passive RFID reads the tag, the user's information along with current timestamp is sent to central server through the communication device installed in the kiosk (as shown in Figure 6). In our bikeshare system, T-Mobile is adopted to transmit these data. Once user's information is verified by the central server, authorization message is sent back to the station. The smart lock will unlock the bike and the user thus can take the bike.

There is also a RFID tag in the head tube of each bike. When a user finishes her/his trip and returns the bike to a dock, the RFID tag

will be close enough to the passive RFID reader which will recognize the return. The real-time return information is then transmitted to the central server which will compute rental fees based on the rental duration.

3.2 Data Sets

Our infrastructure collects multi-event data as follows.

• **User Trip Data:** Each record includes user ID, borrowed bike ID, trip duration, start/end date and time, and start/end station. These data are used to infer user's habit (e.g., origin/destination and rental time) and bike's usage status.

• **Station Status Data:** Each record includes time, station ID/name, station GPS with available bikes or docks. These data help us obtain real-time station status and decide which bike should be selected for a user.

• **Operators' Rebalancing Data:** A rebalancing record includes operators' ID, bike ID, loading/unloading, station ID, time. Combined with the user trip data, these data help us obtain bikes' IDs in a station, and are also used to infer repaired bikes after operators return them.

• **Bike Repair Data:** Different from the above data sets, bike repair data cannot be extracted from the sensing data directly, but they are essential to our design. We build a new data set by manually recording each repaired bike's ID and its corresponding date from 08/03/15 to 10/05/16 in NiceRide bikeshare system. We use this data set as the ground truth for evaluation purposes. Based on the repaired bike data, we can infer (i) the current status of a bike, (ii) the residual service time of a bike, and (iii) whether a bike needs to be repaired.

The above data are typically collected by most of existing bikeshare systems in order to monitor and improve service quality. As a result, our eShare design is helpful to other bikeshare systems to improve the usage efficiency.

3.3 Design Objectives

Leveraging the multi-event data, we aim to design a data-driven usage balancing mechanism to help operators efficiently maintain the shared bikes. Specifically, eShare is designed to achieve the following two objectives:

• **Better Service Quality:** eShare should avoid excessive wearing on a small portion of bikes and fully exploit those rarely-used bikes to improve the overall service quality. In addition, some bikeshare systems impose a mandatory decommission day for operational and maintenance concerns. For example, the mandatory decommission deadline for sharing bikes is three years in China. In the US, although there is no mandatory decommission deadline, most of bikeshare systems upgrade bikes every two to four years. In these cases, eShare aims at controlling the bike usage, so that all bikes are evenly used before the decommissioning/upgrade day. Such an even usage improves overall service quality for riders.

• **Efficient Maintenance:** While bike usage is controlled to be evenly-used, eShare should avoid the situation that all bikes wear out at the same time which may overwhelm repair staff. The concept of *batch balancing* can gracefully degrade bikes in batches so that the workload of repairs is aligned with the staff capacity. To achieve this batch balancing objective, eShare needs to have the capability to statistically control the number of repairing bikes to meet operators' repair capability. Furthermore, it would be desirable to control the interval between routine repairs, so that bike parts can be purchased in batch and labors can be scheduled efficiently.

3.4 Design Overview

Keeping these design objectives in mind, we establish eShare – a usage balancing mechanism which consists of three modules. Although the description of these modules is specific to the bikeshare system, our approach is generically applicable to other sharing systems (e.g., rideshare).

• A historical usage model: It is a statistical model responsible for quantifying the bike historical usage with archived data.

• A future usage model: It is a predictive model responsible for inferring bike usage associated with future rentals from potential passengers.

• A usage calibration engine: It is based on the historical usage as well as the potential usage of the incoming rental, and chooses a bike for coming users to minimize the system-wide usage variance. The usage calibration engine is the core of achieving batch balancing.

eShare is designed to achieve batch balancing control, e.g., controlling the usage of bikes in the same batch to reach the usage threshold for routine repair. In the following, we introduce these three modules as well as how to achieve batch balancing with these modules in detail.

For the sake of clarity, in Section 4, we first introduce our basic design where we use rental duration to quantify bike usage. Thus, our historical usage model is simply to add all cumulated historical rental durations for the same bike together as its historical usage. We directly introduce the other two modules, i.e., the predictive future usage model and the system-wide usage calibration model in Section 4 to show our key idea. Then, in Section 5, we propose an advanced design to quantify bike usage by incorporating more factors (e.g., origin and destination, terrain features, etc.) in addition to rental duration.

4 BASIC DESIGN

This section introduces future usage prediction model, followed by the system-wide usage calibration model. Then, we introduce how to achieve usage balancing with these models.

4.1 Future Usage Prediction

This section describes the predictive future usage model which estimates coming users' rental duration, thus to obtain potential bike usage. Given both historical and real-time user trip data, we propose a fine-grained Bayesian model to estimate users' rental duration. In existing bikeshare systems, when users swipe their cards to rent a bike, the bikeshare system obtains users' ID, start station, and start time. Based on users' historical rental records, we use this real-time information as a condition to predict users' rental duration.

In the following, we first demonstrate that a user's rental duration between a pair of origin (i.e., start station) and destination (i.e., end station) is stable. Then we show that the entropy of a user's destination is low, especially when the start station and start time are given.

• **Property 1: Stable Rental Duration.** For a user u renting a bike at start station S_u and returning the bike at destination D_u, the rental duration R_u is stable. Moreover, if the hour of the start time t_u is given, the standard deviation of the rental duration, denoted by $\sigma(R_u)$, will be further reduced.

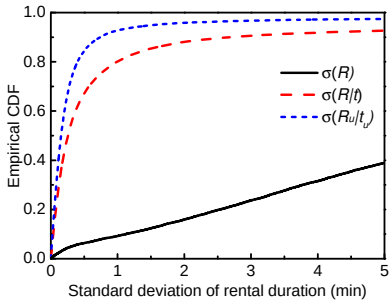


Figure 7: The CDFs of the standard deviation of rental duration with the same origin and destination given start time and user information.

Figure 7 shows the CDFs of (i) $\sigma(R)$, i.e., the standard deviation of rental duration with the same origin and destination, (ii) $\sigma(R|t)$, i.e., the standard deviation of all users' rental duration with the same origin and destination given the hour of the rental start time t, and (iii) $\sigma(R_u|t_u)$, i.e., the standard deviation of user u's rental duration with the same origin and destination pair given user u's start time t_u. The statistical result is based on 58,330 users' 483,239 rental records from 04/01/2015 to 11/01/2015 in NiceRide bikeshare system. From the figure, we can clearly see that the rental duration becomes more stable when the hour of the rental start time and the user information is given. With the start time information, 90% of $\sigma(R|t)$ are lower than three minutes. With both the start time and user information, 90% of $\sigma(R_u|t_u)$ are lower than one minute.

• **Property 2: Low Entropy Destination** The entropy of user u's destination $H(D_u)$ is low. The conditional entropy $H(D_u|S_u, t_u)$ is much lower when the start station S_u and start time t_u are given.

Figure 8 plots user u's CDFs of $H(D_u)$, $H(D_u|S_u)$, and $H(D_u|S_u, t_u)$ based on all her/his trip records during eight months, which are computed as follows,

$$H(D_u) = -\sum_{D_u \in \Psi} p(D_u) \log p(D_u),$$
$$H(D_u|S_u) = \sum_{D_u, S_u \in \Psi} p(S_u, D_u) \log \frac{p(S_u)}{p(S_u, D_u)},$$
$$H(D_u|S_u, t_u) = \sum_{\substack{D_u, S_u \in \Psi, \\ t_u \in \chi}} p(S_u, D_u, t_u) \log \frac{p(S_u, t_u)}{p(S_u, D_u, t_u)},$$

where Ψ is the set of bikeshare stations, and $\chi = \{[00:00:00, 01:00:00), [01:00:00, 02:00:00), \cdots, [23:00:00, 24:00:00)\}$ is the set of hours in a day.

Figure 8 plots the CDFs of (i) $H(D_u)$, i.e., the entropy of user u's destination, (ii) $H(D_u|S_u)$, i.e., the conditional entropy of user u's destination when the start station S_u is given, and (iii) $H(D_u|S_u, t_u)$, i.e., the conditional entropy of user u's destination when both start station S_u and start time t_u are given. The statistical result is based on 43,316 users' rental data who have at least two rental records.

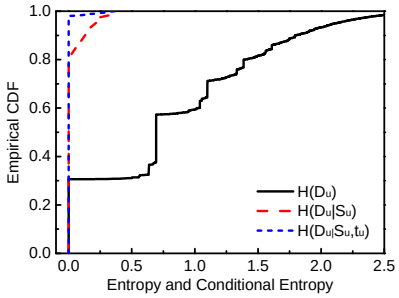


Figure 8: The CDFs of entropy & conditional entropy of destination given start station and time.

15,014 users' rental data are removed in this statistics since these users have only one rental record and thus the entropy of their destination is always zero. As shown in Figure 8, with user's start station and start time information, the values of conditional entropy $H(D_u|S_u)$ and $H(D_u|S_u, t_u)$ are greatly reduced. $H(D_u|S_u, t_u)$ is almost zero, which means user u's destination is almost fixed when his/her start station S_u and start time t_u are given. From the result in Figure 8, we find the destination distribution of a user's coming rental can be accurately predicted given historical user trip records and real-time start station and time.

With the stable duration and low-entropy destination distribution, user u's expected duration $E(R_u)$ is predicted by

$$E(R_u) = \sum_{D_u^i \in \Psi} p(D_u^i) R_u(D_u^i), \quad (1)$$

where $D_u^i \in \Psi$ is user u's possible end station given the start station S_u and start time t_u. $p(D_u^i)$ is the probability that user u will go to end station D_u^i, and $R_u(D_u^i)$ is the rental duration between start station S_u and end station D_u^i.

We use station-wide average rental duration to predict a user's rental duration when the user information is unknown. The predicted rental duration $E(R)$ is given by

$$E(R) = \frac{1}{|U_t^S|} \sum_{u_j \in U_t} E(R_{u_j}), \quad (2)$$

where $E(R_{u_j})$ is computed using Eq.(1), and U_t^S is the set of users who rent bikes in start station S at start time t.

4.2 Usage Calibration Engine

After obtaining the historical bike usage and the usage for the coming rental, this section introduces the system-wide usage calibration engine which is designed to select bikes for users to minimize system-wide usage variance. Although the usage calibration design itself only balances the system-wide usage, it can be used as an engine to achieve batch balancing by introducing weight to historical usage, which will be introduced in Section 4.3. In the following, we first propose the rationale behind our design. Then, we introduce the calibration algorithm.

4.2.1 Design Rationale. When a user u comes to a station to rent a bike, let the available bikes in this station be $\{b_1, b_2, ..., b_m\}$. We calculate the historical usage of these bikes $\{H_{b_1}, H_{b_2}, ..., H_{b_m}\}$ using the usage statistic model (i.e., the rental duration in the basic version of eShare). We then apply future usage prediction model to infer user u's future usage $E(R_u)$. For an arbitrary bike b_i in the station,

if it is selected for user u, then this bike's usage is updated to the sum of its historical usage and future usage, i.e., $R_{b_i} = H_{b_i} + E(R_u)$.

When minimizing the system-wide usage variance, it will be helpful if the usage calibration engine obtains *station-wide* usage statistics. We use an example to illustrate this idea. There are two bikes $\{b_1, b_2\}$ in the station. Bike b_1's historical usage is three hours while bike b_2's historical usage is two hours. When a user u_1 comes to a station to rent a bike, the bikeshare system obtains the user's real-time and historical usage after the user swiped his/her card. The system applies the usage prediction algorithm and obtains u_1's potential usage is one hour. On the other hand, the station-wide average usage is two hour.

Without considering the station-wide statistics, the calibration engine will select bike with minimum usage, i.e., bike b_2, and assign it to user u_1 to minimize system-wide usage variance. Bike b_2's usage is thus updated to $R_{b_2} = H_{b_2} + E_{R_{u_1}} = 2 + 1 = 3$. Since the station-wide average usage is two hour, in statistics, bike b_1's usage is given by $R_{b_1} > 3 + 2 = 5$. The usage variance is thus greater than $(3 - \frac{3+5}{2})^2 + (5 - \frac{3+5}{2})^2 = 2$.

However, if station-wide average usage is considered, the calibration engine will select the bike b_1 for user u_1. The bike usage is updated to $R_{b_1} = 3 + 1 = 4$ and $R_{b_2} > 2 + 2 = 4$, and the usage variance is greater than zero. Comparing these two cases, we find that it is better to consider station-wide usage statistics in bike selection since it helps to achieve smaller usage variance.

4.2.2 Calibration Algorithm. We now introduce how to utilize station-wide usage statistics to let the calibration engine decide the most suitable bike for a user u to minimize the system-wide usage variance. We assume that there are m bikes in the station when user u_1 comes to rent a bike. In the usage calibration design, without losing generality, we consider k potential users, i.e., $u_1, u_2, \ldots, u_k$, where $1 \le k \le m$ and $[u_2, \ldots, u_k]$ are users coming after u_1. Note that the problem formulation with a generic k is helpful for practical settings especially when coming users' reservation information is provided. Let each user u_i's expected usage be $E(R_{u_i})$, where $E(R_{u_1})$ is calculated based on Eq.(1). $E(R_{u_i})$, where $2 \le i \le k$, is estimated based on Eq.(2) when coming users' information is vacant.

Since the variety of bike selection at a station does not change the usage variance of bikes at other stations, the goal of minimizing the system-wide variance is equivalent to

$$\min \sum_{i=1}^{k} \sum_{j=1}^{m} \|(H_{b_j} + E(R_{u_i}))x_{ij} - \overline{R}\|^2, \tag{3}$$

s.t.

$$\begin{aligned} &\sum_{i=1}^{k} x_{ij} \le 1 \quad j = 1, \ldots, m, \\ &\sum_{j=1}^{m} x_{ij} = 1 \quad i = 1, \ldots, k, \\ &\sum_{i=1}^{k} \sum_{j=1}^{m} x_{ij} = k, \qquad x_{ij} = 0 \text{ or } 1, \end{aligned} \tag{4}$$

where $x_{ij} = 1$ if bike b_j for user u_i is selected, 0 if not. In the optimization target, H_{b_j} represents bike b_j's historical usage, $E(R_{u_i})$ represents the expected usage when bike b_j for user u_i is selected, and $H_{b_j} + E(R_{u_i})$ represents the total usage when bike b_j is selected for user u_i. $\overline{R}$ is the system-wide average bike usage and $\overline{R} = \frac{1}{n}(\sum_{k=1}^{n} H_{b_k} + \sum_{i=1}^{k} E_{R_i})$. Then the optimization target represents choosing bikes at station S for coming users to minimize the usage variance. The first set of constraints ensures that no more than one user is assigned to any bike. The second set of constraints ensures that one bike is selected for every user. The third constraint ensures that k of the bikes for all the k users are selected.

We find that it is difficult to solve the nonlinear usage calibration problem with the four constraints directly. In the following, we transform the above nonlinear usage calibration problem to a linear programming problem.

Lemma 4.1. ***(Problem Transformation)*** *The nonlinear usage calibration problem in formula (3)-(4) is equivalent to the linear programming problem with the optimization target* $\min \sum_{i=1}^{k} \sum_{j=1}^{m} \|H_{b_j} + E(R_{u_i}) - \overline{R}\|^2 x_{ij}$ *and the same constraints in formula (4).*

We find that this transformed problem is a variation of the classic assignment problem, named k-cardinality assignment problem, in which there are k rows and m columns, but only k rows are assigned to k columns to minimize the sum of corresponding costs. The k-cardinality assignment problem has optimal solutions that can be achieved by the rectangular assignment algorithm with time complexity $O(k^2 \log k)$.

In the usage calibration process, on the one hand, the system-wide usage variance can be further reduced when we have exact reservation information of more coming users (i.e., a larger k). The estimation of the expected usage of coming users contains prediction errors, which may lead to an inefficient calibration to increase the usage variance. When the estimated user information is adopted, we run the usage calibration algorithm with all possible k, where k is an integer and $k \in [1, m]$, to choose the k which achieves the minimal system-wide variance. By solving the usage calibration problem in formula (3)-(4) with the optimal k, the calibration algorithm chooses the bike which is assigned to u_1.

• **Special Case:** Note that the above usage calibration algorithm includes the special case that the algorithm has no knowledge about the coming users, i.e., $k = 1$ in the usage calibration problem formulation.

Lemma 4.2. ***(Lower Bound)*** *When the algorithm has no knowledge about the coming users, the usage calibration algorithm chooses the bike with minimum usage in the station and reaches the lower bound.*

From this lemma, the performance of the usage calibration algorithm will not be worse than the performance of the algorithm which always selects the bike with minimum usage in a station.

4.3 Batch Balancing

The above usage calibration engine minimizes the system-wide bike usage variance. This engine provides us the capability to achieve batch balancing (e.g., balancing the usage of bikes in the same batch to reach the threshold calling for repair or other maintenance

activities at the same time). With the help of the usage calibration engine, we are able to manage the number of wear-out bikes to meet the maintenance capability by simply adjusting bikes' historical usage to either accelerate or postpone bike usage.

Let the batch of controlled bikes be $\{cb_1, cb_2, \ldots, cb_i, \ldots\}$. We update these bikes' usage with $\alpha \times R(cb_i)$, where $R(cb_i)$ is bike cb_i's real usage. We then apply $\alpha R(cb_i)$ as bike cb_i's usage in the usage calibration algorithm. Recall that the calibration model minimizes the system-wide bike usage variance. When other bikes' average usage is $\overline{R}$, the batch of controlled bikes's real usage is $\frac{\overline{R}}{\alpha}$. When $0 < \alpha < 1$, the usage of the batch of controlled bikes are accelerated. These bikes' usage are postponed when $\alpha > 1$. In real-world control applications in bikeshare systems, when the expected usage after control $\{C(cb_1), C(cb_2), \ldots, C(cb_i), \ldots\}$ is provided, then the value of α for bike cb_i is given by $\alpha(cb_i) = \frac{\sum_{i=1}^{n} C(cb_i)}{n \cdot C(cb_i)}$.

5 ADVANCED DESIGN

In the basic version of eShare, we quantify bike usage using rental duration which accounts for riding time. In this section, we provide an advanced design to quantify the bike usage by exploiting the depreciation caused by each rental trip (accounting for terrain features) from user trip data and bike repair data.

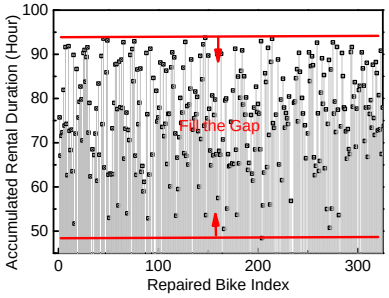


Figure 9: Accumulated Duration of Damaged Bikes

In the basic version of eShare, we use "rental duration" to quantify the bike usage. It works well under the situation that the depreciation caused by rental trips with same rental time are similar or the same. But in some scenarios, even with the same rental time, some rental trips may cause more bike depreciation than others, which leads to a higher repair cost. For example, in Bay Area bikeshare system in San Francisco, the rental trips locate in steep slopes cause more depreciation than those trips locate in flat roads. Under such a scenario, simply using "rental duration" may not accurately decide the status of a bike and the repair frequency. To provide some empirical insights, we use a repair event as an indicator of high usage for a bike. Then, we can explore the rental time distribution of repaired bikes to investigate if accumulated rental duration can be effectively used to indicate usage or not. Figure 9 shows the rental duration of the repaired bikes in NiceRide bikeshare system from 08/03/2015 to 11/01/2015. From this figure, we find that a bike that is repaired due to high usage can have a rental duration as short as 50 hours and as long as 95 hours . Thus, the rental duration alone is not enough to quantify usage.

In the following, we propose an enhanced usage quantification method by leveraging the depreciation caused by each rental trip from the large repaired bike data set. Let the differentiated source-destination pairs in the user trip records be $SD = \{sd_1, sd_2, \ldots, sd_l\}$. For each source-destination pair $sd_j \in SD$, we introduce depreciation rate w_j, where $w_j > 0$. Let the bikes in the repair records be $[rb_1, rb_2, \ldots, rb_N]$. For each repaired bike rb_i, we extract all its rental records from the user trip data within the repair-cycle. For bike rb_i's each differentiated source-destination pair $sd_j^{rb_i}$, we sum up the rental duration under this source-destination pair and obtain $A_j^{rb_i}$, which is accumulated rental duration under trip sd_j. Then, for each repaired bike rb_i, we obtain its adjusted total service duration $\sum_{j=1}^{l} w_j A_j^{rb_i}$.

Since we have a large repaired bike data set, we obtain a large observation of variables about the repaired bikes' total service duration, i.e., $X = [x_1\ x_2\ \ldots\ x_i\ \ldots]^T$, where $x_i = \sum_{j=1}^{l} w_j A_j^{rb_i}$. We assume that the total bike service duration subjects to normal (Gaussian) distribution, i.e., $X \sim N(\mu, \sigma^2)$, where μ is the expected average bike service duration. To reduce the variance of repaired bikes' total service duration, we aim to

$$\min \sum_{i=1}^{N} (\sum_{j=1}^{l} w_j A_j^{rb_i} - X'[i])^2, \tag{5}$$

where $X' = [x'_1\ x'_2\ \ldots\ x'_i\ \ldots]^T$, $X' \sim N(\mu, \sigma'^2)$ and $\sigma' \to 0$.

Lemma 5.1. *The minimum variance of repaired bikes' total service duration will be achieved when*

$$\hat{\mathbf{w}} = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{X}'. \tag{6}$$

Eq.(6) provides an optimal solution to the problem in Eq.(5) when the total bike service duration follows normal distribution. From Lemma 5.1, we obtain the depreciation rate w_j for each differentiated trip sd_j. The depreciation rate w_j quantifies each source-destination pair's rental cost and fills the gap among repaired bikes' rental duration. The enhanced usage statistics model thus builds a connection between the usage and rental costs by inferring repaired bikes from the accumulated rental duration. Besides, this depreciation rate provides the operators the basis of a new charging scheme about rental fees with fine-grained depreciation rate.

6 EVALUATION: ID SWAPPING

This section evaluates the performance of eShare with real world data sets from three bikeshare systems. The basic information of these data sets are shown in Table 2. The format of these data sets is introduced in Section 3.2.

6.1 Evaluation Methodology

This section introduces the ID swapping based evaluation methodology which utilizes long-term multi-event data, including user trip, operator rebalancing and bike repair data, to evaluate the performance of eShare. The existing systems' bike usage information, including user ID, bike ID, time and station, can be found directly from the user trip data. Recall that eShare selects a bike from the available bikes in the station for each rent to minimize the system-wide usage variance. We first recover the available bikes (IDs) in a station at anytime. We observe that station status changes if and only if a user borrows/returns a bike or a system operator unloads/loads a bike. In other words, a bike is always in a station in the

Table 2: Basic Information of Collected Data from Three Bikeshare Systems.

Data Set Name	Accessed Periods	Data Size	Number of Records	Recording Frequency
Capital_User_Trip	10/01/10 - 12/31/15	1.4GB	11.8×10^6	Event Driven
Capital_Station_Status	08/04/14 - 12/31/15	1.9GB	269×10^6	Every 1min
Capital_Repair_Bike	01/01/11 - 09/30/15	297KB	3.14×10^3	Event Driven
HangZhou_User_Trip	01/01/13 - 12/31/13	4.2GB	103×10^6	Event Driven
HangZhou_Station_Status	01/01/13 - 12/31/13	1.2GB	248×10^3	Every 30min
HangZhou_Rebalancing	01/01/13 - 12/31/13	109MB	8.5×10^6	Event Driven
NiceRide_User_Trip	06/10/10 - 11/01/15	189MB	1.8×10^6	Event Driven
NiceRide_Rebalancing	04/01/12 - 11/01/15	31.9MB	465×10^3	Event Driven
NiceRide_Repair_Bike	08/03/15 - 10/05/16	272KB	3,724	Event Driven

duration after it is returned/loaded by the previous user/operator and before the next borrow/unload activity comes.

Based on this observation, eShare exploits the available bikes (IDs) for the bike selection of each rent. By utilizing both historical usage and real time card swiping information, eShare selects a bike for each rent in the user trip data. If the selected bike ID is different from the bike ID in the rent record in user trip data, we swap these two bikes' IDs and create a new user trip data with the bike (ID) selected by eShare. In other words, the selected bike by eShare will replace the original bike in the rent record. *Note that such a swap will not change the station status and thus will not change either user demand or service level.* Based on the newly-built user trip data, we are able to do statistics on eShare's system-wide bike usage.

•**Compared Algorithms:** To show the effectiveness of usage balancing in sharing platforms, we compare eShare with (i) the ground truth, which is the bike selection strategy extracted from the user trip data set and (ii) the algorithm which always selects the bike with minimum usage in a station, indicated as "Minimum Usage" in the figures.

•**Performance Metric:** We evaluate the bike usage efficiency from both single-bike and system-wide perspectives:

(i) **Service Timespan:** from single-bike perspective, we investigate each bike's service timespan which is defined as the time duration between the first use of a bike and the last use before a repair.

(ii) **Usage Variance:** from system-wide perspective, we investigate usage variance which is given by $\frac{1}{n}\sum_{i=1}^{n}(R_i - \overline{R})^2$, where n is the total number of bikes in the system, R_i is each bike's usage and $\overline{R}$ is the average bike usage.

The bike usage is defined as the total rental duration in service timespan multiplying by the depreciation rate. In the basic version of eShare, the depreciation rate is set to one.

6.2 Main Performance

In this subsection, we evaluate the performance of eShare by investigating bikes' service timespan and usage variance. Figure 10 shows the each bike's usage in two weeks in Capital bikeshare system with different bike selection strategies. In the figures, the bike usage of ground truth varies dramatically. The bike usage in Capital bikeshare system varies from 0 to 2000 minutes. As a comparison, Figure 10 also shows each bike's usage with eShare and "minimum usage" algorithm. From the figures, we can see that all the bikes in the systems are relatively fully utilized and the usage variance with

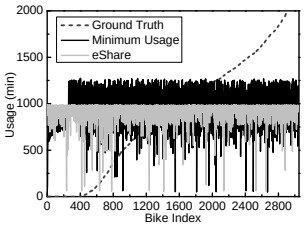


Figure 10: eShare main Performance

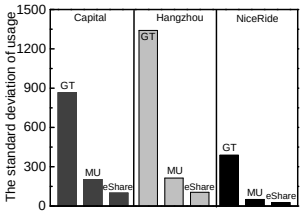


Figure 11: Bike usage Deviation

eShare is significantly reduced. In addition, eShare always achieves better performance than the "minimum usage" algorithm which is a special case of the eShare design. Figure 11 shows the standard deviation of usage with different algorithms in the three bikeshare systems. On average, the standard deviation of usage is reduced by 90% after eShare is applied to existing systems.

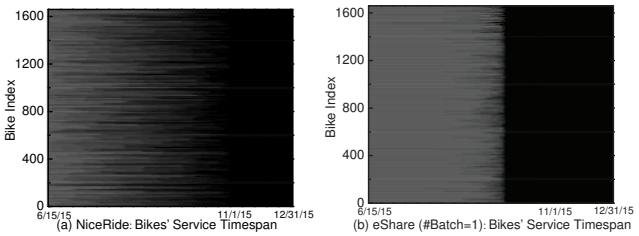


Figure 12: Service Timespan: (a) NiceRide system and (b) eShare

When bike usage is balanced, each bike's service timespan is changed accordingly. Figure 12 plots 1658 bikes' service timespan before (Figure 12(a)) and after (Figure 12(b)) eShare is applied in NiceRide bikeshare system. In the figures, when the color turns to black, it means that the bike's accumulated usage reaches the threshold for routine repair. From Figure 12(b), we find that the end of the service timespan is close when the bikes in the system are balanced used. It is helpful when the end of service time sits in a period when the bikeshare system is idle or in mandatory-decommission/bike-upgrade day.

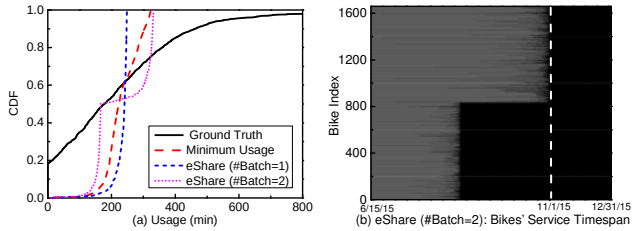


Figure 13: Capability of Batch Balancing:(a) CDFs of Usage and (b) Service Time.

•**Capability of Batch Balancing:** With eShare, we are able to achieve batch balancing. In this experiment, we accelerate the usage of half of the bikes by setting these bikes' usage to half of the real one in bike selection algorithm. Figure 13(a) shows the CDF of two-week bike usage with "Ground Truth", "Minimum Usage", "eShare ($\#Batch = 1$)", and "eShare ($\#Batch = 2$)" in NiceRide bikeshare system. From the figure, we can see that "eShare ($\#Batch = 2$)" put the 18% barely used bikes into use.

On the other hand, there is a usage gap between two portions of bikes. Figure 13(b) shows the corresponding service timespan under the bike usage with "eShare ($\#Batch = 2$)". From the figure, we can see that "eShare ($\#Batch = 2$)" makes portion of bikes' end-of-service-time sit in the idle period (e.g., November) of NiceRide bikeshare system. Figure 13 only shows eShare's capability on batch balancing while a specific control model is required to control the number of repairing bikes to fit with operators' repair capability.

6.3 Design Insight Analysis

In this subsection, we reveal design insights to illustrate why we achieve the above performance.

6.3.1 Performance of Usage Prediction. We perform the evaluation of our method with one-year user trip data from NiceRide bikeshare system which has 483,239 rental records. In this evaluation, we compare the method in eShare with the historical average (HA) method and the "HA" method with start time ("HA w/ ST"). In the "HA" method, we use the average rental duration of a source destination pair's all records to predict a new coming user's rental duration. In the "HA w/ ST" method, we use the average rental duration of a source-destination pair's records with the same hour of the start time.

The metric we adopt to measure the result is error rare which is given by $\frac{1}{N}\sum_{i=1}^{N}\frac{|\widehat{R_i}-R_i|}{R_i}$, where N is the total number of checked rental records, R_i is the ground truth of the rental duration in the i^{th} rental record and $\widehat{R_i}$ is the corresponding prediction value.

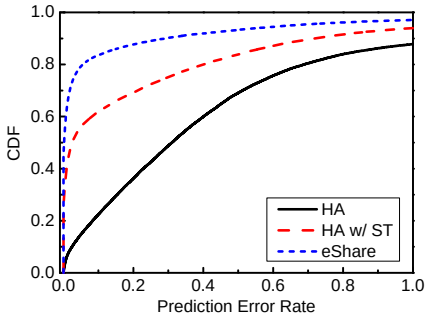


Figure 14: The CDFs of the prediction error rate

Figure 14 plots the CDFs of the prediction error rate of the "HA", "HA w/ ST", and eShare. As we can see, 80% of the predictions with our method have an error rate less than 5.5%. Compared with "HA" and "HA w/ ST", our prediction method significantly reduces the prediction error rate. The rationality behind this improvement is that our method fully exploits fine-grained user rental information from both historical and real-time user trip data in the prediction.

6.3.2 Performance of Usage Statistics. This section evaluates the performance of the enhanced usage model. In NiceRide bikeshare

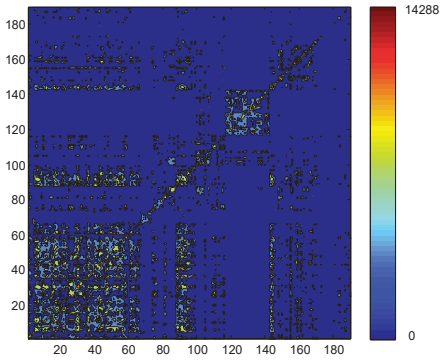


Figure 15: The four-year system-wide trip distribution with all origin-destination pairs in NiceRide bike sharing system.

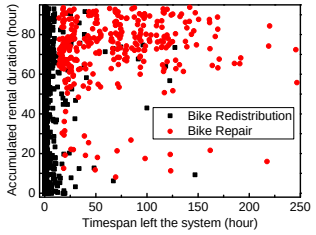


Figure 16: Timespan of Redis. and Repair

system, there are 190 stations and there are 190^2 possible origin-destination pairs. Figure 15 shows four-year system-wide trip distribution with all origin-destination pairs. There are 1,469,882 rental trips in total while the busiest origin-destination has 14,288 trips. In Figure 15, a origin-destination pair has more trips when its color is closer to red. From the figure, we can clearly see that there are many partial red dots in the diagonal. It means that there are a large number of trips which have the same origin and destination. That's the major reason why we do not adopt "mileage" as the measure of usage in this paper. Since the bikes in current bikeshare systems are not equipped with GPS, simply utilizing station distance to measure bike usage will introduce great errors in usage statistics because of the zero distance of the trips with same origin-destination.

We find that a large portion of areas are marked "blue" which means that these origin-destination pairs have almost no traffic. According to our statistics, 3,291 origin-destination pairs have more than one trip in two weeks. That's because that people normally take bike as a short-distance transport tool and only ride bikes to nearby stations.

In the following, we compute the depreciation rate $\hat{\mathbf{w}}$ for these 3,291 origin-destination pairs which requires at least 3,291 bike repair records. We manually record 3,724 bike repair logs with bike id and repair date from 08/03/15 to 10/05/16. Since the usage statistics model will be more accurate when we have more bike repair records, we use the collect repair logs as ground truth and build a model to infer repaired bikes from operator rebalancing data automatically.

For each bike unloading behavior (by operators), we compute this bike's accumulated usage from the user trip data as well as the timespan the bike left the system to classify this unloading behavior is for bike redistribution or for repair. Figure 16 plots the timespan of a bike left the system and the bike's accumulated rental duration when this bike is unload from the system. From the figure, we can see that the bike unloading for repair has long departure timespan and accumulated rental duration.

Table 3: Performance of Classification

Algorithm	TP Rate	FP Rate	Precision	Recall
KNN	0.987	0.137	0.987	0.987
Naive Bayes	0.977	0.346	0.975	0.977
Decision Trees	0.990	0.046	0.991	0.990
RandomForest	0.989	0.065	0.990	0.989
AdaBoost	0.991	0.022	0.992	0.991

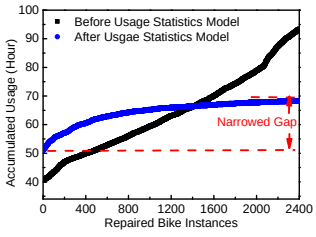


Figure 17: Model Validation

We apply classification algorithms to the data set. Table 3 reports the true positive (TP) rate, false positive (FP) rate, precision, and recall with KNN, Naive Bayes, RandomForest and AdaBoost by 10-fold cross-validation. From the experiment results, we can see that the repaired bike can be accurately predicted from user trip and operator rebalancing data. AdaBoost achieves the best performance since the data set has few outliers which can be found from Figure 16. We apply AdaBoost to four-year user trip and operator rebalancing data where we recover 12,662 bike repair records. Based on these bike repair records, we apply our usage statistics model. Its performance is shown in Figure 17 where we find that the gap of repair bikes' accumulated usage is narrowed. This is because that the usage statistics model leverages the depreciation (caused by terrain features) which reduces the variance of bikes' accumulated usage time.

7 RELATED WORK

The sharing platforms, e.g., Uber [13, 23], Airbnb [12, 20] and Bikeshare, have been on an exponential growth curve over recent years. In sharing platforms, it is crucial to investigate the usage of shared resources to provide an efficient sharing service. To the best of knowledge, most of existing research focuses on participation analysis [13, 23], incentive mechanism and benefits analysis [8, 12, 20] in the sharing economy. Little research, if any, has been conducted to study the fine-grained usage status among shared resources with large-scale data collected from these platforms.

Existing works on bike sharing systems focus on (i) demand analysis [1, 2, 9–11, 14, 15, 24–27], (ii) service analysis [3, 18, 19, 21], and (iii) redistribution scheduling [6, 16, 17]. Compared with previous studies on demand analysis, service analysis, and redistribution scheduling in bikeshare systems, this paper studies a completely new class of problem, i.e., usage balancing for efficient sharing.

8 CONCLUSION

In this paper, we design eShare to achieve usage efficient sharing. We provide a usage statistics model which quantifies (i) each bike's usage status and residual service time and (ii) each rental trip's depreciation rate. These information provides system operator valuable insights on managing bike resources and improving the quality of sharing service. We also propose a fine-grained usage prediction model for a specified user by leveraging his/her historical and real-time bike usage information collected by existing bikeshare systems. In addition, an optimal usage calibration algorithm is proposed to choose the most suitable bike to optimize the system-wide bike usage. The performance of eShare is evaluated through the data sets collected from three representative bikeshare systems. The evaluation results show that our eShare design helps existing bikeshare systems fully utilize shared bikes, improve service quality and reduce maintenance cost.

ACKNOWLEDGMENTS

The work is supported by China National Key R&D Program 2017YFB1003000, US NSF Grants CNS-1446640 and DGIST Research and Development Program funded by MSIP.

REFERENCES

[1] A. Bargar, A. Gupta, S. Gupta, and D. Ma. 2014. Interactive visual analytics for multi-city bikeshare data analysis. In *Proceedings of the 3rd Urbcomp*.

[2] P. Borgnat, C. Robardet, P. Abry, P. Flandrin, J. Rouquier, and N. Tremblay. 2013. A dynamical network view of Lyon's Velo'v shared bicycle system. *Dynamics on and of Complex Networks* 2 (2013), 267–284.

[3] Longbiao Chen, Daqing Zhang, Gang Pan, Xiaojuan Ma, Dingqi Yang, Kostadin Kushlev, Wangsheng Zhang, and Shijian Li. 2015. Bike Sharing Station Placement Leveraging Heterogeneous Urban Open Data. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*. 571–575.

[4] Longbiao Chen, Daqing Zhang, Leye Wang, Dingqi Yang, Xiaojuan Ma, Shijian Li, Zhaohui Wu, Gang Pan, Thi-Mai-Trang Nguyen, and Jérémie Jakubowicz. 2016. Dynamic Cluster-based Over-demand Prediction in Bike Sharing Systems. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*. 841–852.

[5] A. Cohen, D. Simons, M. Martignoni, J. Olson, and C. Holben. 2014. The Bikeshare Planning Guide. *Institute for Transportation and Development Policy (ITDP)* (2014).

[6] G. Dantzig and J. Ramser. 1959. The Truck Dispatching Problem. *Management Science* 6, 1 (1959), 80–91.

[7] P DeMaio. 2009. Bike-sharing: History, Impacts, Models of Provision, and Future. *Journal of Public Transportation* 12, 4 (2009).

[8] T. R. Dillahunt, X. Wang, E. Wheeler, H. F. Cheng, B. Hecht, and H. Zhu. 2017. The Sharing Economy in Computing: A Systematic Literature Review. In *Proceedings of the ACM on Human-Computer Interaction*.

[9] Côme Etienne and Oukhellou Latifa. 2014. Model-Based Count Series Clustering for Bike Sharing System Usage Mining: A Case Study with the VÉLib' System of Paris. *ACM Trans. Intell. Syst. Technol.* 5, 3 (2014), 39:1–39:21.

[10] Jon Froehlich, Joachim Neumann, and Nuria Oliver. 2009. Sensing and Predicting the Pulse of the City Through Shared Bicycling. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*. 1420–1426.

[11] Andreas Kaltenbrunner, Rodrigo Meza, Jens Grivolla, Joan Codina, and Rafael Banchs. 2010. Urban Cycles and Mobility Patterns: Exploring and Predicting Trends in a Bicycle-based Public Transport System. *Pervasive Mob. Comput.* 6, 4 (2010), 455–466.

[12] Qing Ke. 2017. Service Providers of the Sharing Economy: Who Joins and Who Benefits? *Proceedings of the ACM on Human-Computer Interaction* 1 (2017), 57:1–57:17.

[13] Farshad Kooti, Mihajlo Grbovic, Luca Maria Aiello, Nemanja Djuric, Vladan Radosavljevic, and Kristina Lerman. 2017. Analyzing Uber's Ride-sharing Economy. In *Proceedings of the 26th International Conference on World Wide Web Companion (WWW)*.

[14] N. Lathia, S. Ahmed, and L. Capra. 2012. Measuring the impact of opening the London shared bicycle scheme to casual users. *Transportation Research Part C: Emerging Technologies* 22 (2012), 88–102.

[15] Yexin Li, Yu Zheng, Huichu Zhang, and Lei Chen. 2015. Traffic Prediction in a Bike-sharing System. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 33:1–33:10.

[16] Junming Liu, Leilei Sun, Weiwei Chen, and Hui Xiong. 2016. Rebalancing Bike Sharing Systems: A Multi-source Data Smart Optimization. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1005–1014.

[17] Benchimol M., Benchimol P., Chappert B., Taille A. D. L., Laroche F., Meunier F., and Robinet L. 2011. Balancing a dynamic public bike-sharing system. *RAIRO – Operations Research* 45, 1 (2011), 37–61.

[18] Rahul Nair and Elise Miller-Hooks. 2011. Fleet Management for Vehicle Sharing Operations. *Transportation Science* 45, 4 (2011), 524–540.

[19] R. Naira, R. Hampshire E. Miller-Hooks, and A. Busic. 2013. Large-Scale Vehicle Sharing Systems: Analysis of Velib. *International Journal of Sustainable Transportation* 7, 1 (2013), 85–106.

[20] Giovanni Quattrone, Davide Proserpio, Daniele Quercia, Licia Capra, and Mirco Musolesi. 2016. Who Benefits from the "Sharing" Economy of Airbnb. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*.

[21] Tal Raviv, Tzur Michal, and I. Forma. 2013. Static Repositioning in a Bike-Sharing System: Models and Solution Approaches. *EURO Journal on Transportation and Logistics* 2, 3 (2013), 187–229.

[22] Susan A. Shaheen, Elliot W. Martin, Nelson D. Chan, Adam P. Cohen, and Mike Pogodzinski. 2014. Public Bikesharing In North America During a period Of Rapid Expansion: Understanding Business Models, Industry Trends And User Impacts. *Mineta Transportation Institute* (2014).

[23] Jacob Thebault-Spieker, Loren Terveen, and Brent Hecht. 2017. Toward a Geographic Understanding of the Sharing Economy: Systemic Biases in UberX and TaskRabbit. *ACM Trans. Comput.-Hum. Interact.* 24, 3 (2017), 21:1–21:40.

[24] P. Vogel and D. Mattfeld. 2011. Strategic and operational planning of bike-sharing systems by data mining- a case study. In *Proceedings of the 2nd International Conference on Computational Logistics*. 127–141.

[25] S. Wang, T. He, D. Zhang, Y. Shu, Y. Liu, Y. Gu, C. Liu, H. Lee, and S. Son. 2018. BRAVO: Improving the Rebalancing Operation in Bike Sharing with Rebalancing Range Prediction. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*.

[26] Zidong Yang, Ji Hu, Yuanchao Shu, Peng Cheng, Jiming Chen, and Thomas Moscibroda. 2016. Mobility Modeling and Prediction in Bike-Sharing Systems. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. 165–178.

[27] Ji Won Yoon, Fabio Pinelli, and Francesco Calabrese. 2012. Cityride: A Predictive Bike Sharing Journey Advisor. In *Proceedings of the 13th International Conference on Mobile Data Management*. 306–311.

Heterogeneous Graph Attention Network

Xiao Wang, Houye Ji
Beijing University of Posts and Telecommunications
Beijing, China
{xiaowang,jhy1993}@bupt.edu.cn

Chuan Shi*, Bai Wang
Beijing University of Posts and Telecommunications
Beijing, China
{shichuan,wangbai}@bupt.edu.cn

Peng Cui, P. Yu
Tsinghua University
Beijing, China
{cuip,psyu}@tsinghua.edu.cn

Yanfang Ye
West Virginia University
WV, USA
yanfang.ye@mail.wvu.edu

ABSTRACT

Graph neural network, as a powerful graph representation technique based on deep learning, has shown superior performance and attracted considerable research interest. However, it has not been fully considered in graph neural network for heterogeneous graph which contains different types of nodes and links. The heterogeneity and rich semantic information bring great challenges for designing a graph neural network for heterogeneous graph. Recently, one of the most exciting advancements in deep learning is the attention mechanism, whose great potential has been well demonstrated in various areas. In this paper, we first propose a novel heterogeneous graph neural network based on the hierarchical attention, including node-level and semantic-level attentions. Specifically, the node-level attention aims to learn the importance between a node and its meta-path based neighbors, while the semantic-level attention is able to learn the importance of different meta-paths. With the learned importance from both node-level and semantic-level attention, the importance of node and meta-path can be fully considered. Then the proposed model can generate node embedding by aggregating features from meta-path based neighbors in a hierarchical manner. Extensive experimental results on three real-world heterogeneous graphs not only show the superior performance of our proposed model over the state-of-the-arts, but also demonstrate its potentially good interpretability for graph analysis.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning approaches**.

KEYWORDS

Social Network, Neural Network, Graph Analysis

ACM Reference Format:
Xiao Wang, Houye Ji, Chuan Shi*, Bai Wang, Peng Cui, P. Yu, and Yanfang Ye. 2019. Heterogeneous Graph Attention Network. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313562

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313562

1 INTRODUCTION

The real-world data usually come together with the graph structure, such as social networks, citation networks, and the world wide web. Graph neural network (GNN), as a powerful deep representation learning method for such graph data, has shown superior performance on network analysis and aroused considerable research interest. For example, [10, 20, 24] leverage deep neural network to learn node representations based on node features and the graph structure. Some works [6, 14, 18] propose the graph convolutional networks by generalizing the convolutional operation to graph. A recent research trend in deep learning is the attention mechanism, which deals with variable sized data and encourages the model to focus on the most salient parts of data. It has demonstrated the effectiveness in deep neural network framework and is widely applied to various applications, such as text analysis [1], knowledge graph [25] and image processing [38]. Graph Attention Network (GAT) [35], a novel convolution-style graph neural network, leverages attention mechanism for the homogeneous graph which includes only one type of nodes or links.

Despite the success of attention mechanism in deep learning, it has not been considered in the graph neural network framework for heterogeneous graph. As a matter of fact, the real-world graph usually comes with multi-types of nodes and edges, also widely known as heterogeneous information network (HIN) [28]. For convenience, we uniformly call it heterogeneous graph in this paper. Because the heterogeneous graph contains more comprehensive information and rich semantics, it has been widely used in many data mining tasks. Meta-path [32], a composite relation connecting two objects, is a widely used structure to capture the semantics. Taking the movie data IMDB[1] shown in Figure 1(a) as an example, it contains three types of nodes include movie, actor and director. A relation between two movies can be revealed by meta-path Movie-Actor-Movie (*MAM*) which describes the co-actor relation, while Movie-Director-Movie (*MDM*) means that they are directed by the same director. As can be seen, depending on the meta-paths, the relation between nodes in the heterogeneous graph can have different semantics. Due to the complexity of heterogeneous graph, traditional graph neural network cannot be directly applied to heterogeneous graph.

Based on the above analysis, when designing graph neural network architecture with attention mechanism for heterogeneous graph, we need to address the following new requirements.

[1] https://www.imdb.com

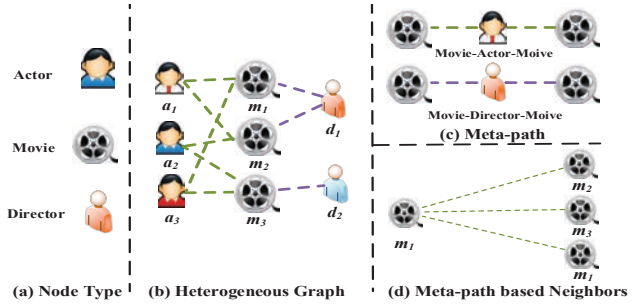


Figure 1: An illustrative example of a heterogenous graph (IMDB). (a) Three types of nodes (i.e., actor, movie, director). (b) A heterogenous graph IMDB consists three types of nodes and two types of connections. (c) Two meta-paths involved in IMDB (i.e., Moive-Actor-Moive and Movie-Director-Movie). (d) Moive m_1 and its meta-path based neighbors (i.e., m_1, m_2 and m_3).

Heterogeneity of graph. The heterogeneity is an intrinsic property of heterogeneous graph, i.e., various types of nodes and edges. For example, different types of nodes have different traits and their features may fall in different feature space. Still taking IMDB as an example, the feature of an actor may involve in sex, age and nationality. On the other hand, the feature of movie may involve in plot and actors. How to handle such complex structural information and preserve the diverse feature information simultaneously is an urgent problem that needs to be solved.

Semantic-level attention. Different meaningful and complex semantic information are involved in heterogeneous graph, which are usually reflected by meta-paths [32]. Different meta-paths in heterogeneous graph may extract diverse semantic information. How to select the most meaningful meta-paths and fuse the semantic information for the specific task is an open problem [4, 19, 26]. Semantic-level attention aims to learn the importance of each meta-path and assign proper weights to them. Still taking IMDB as an example, *The Terminator* can either connect to *The Terminator 2* via Movie-Actor-Movie (both starred by *Schwarzenegger*) or connect to *Birdy* via Movie-Year-Movie (both shot in *1984*). However, when identifying the genre of the movie *The Terminator*, *MAM* usually plays more important role, rather than *MYM*. Therefore, treating different meta-paths equally is unpractical and will weaken the semantic information provided by some useful meta-paths.

Node-level attention. In a heterogeneous graph, nodes can be connected via various types of relation, e.g., meta-path. Given a meta-path, each node has lots of meta-path based neighbors. How to distinguish the subtle difference of there neighbors and select some informative neighors is required. For each node, node-level attention aims to learn the importance of meta-path based neighbors and assign different attention values to them. Still taking IMDB as an example, when using the meta-path Movie-Director-Moive (the movies are with the same director), *The Terminator* will connect to *Titanic* and *The Terminator 2* via director *James Cameron*. To better identify the genre of *The Terminator* as sci-fi movie, the model should pay more attention to *The Terminator 2*, rather than *Titanic*. Therefore, how to design a model which can discover the subtle differences of neighbors and learn their weights properly will be desired.

In this paper, we propose a novel **H**eterogeneous graph **A**ttention **N**etwork, named HAN, which considers both of node-level and semantic-level attentions. In particular, given the node features as input, we use the type-specific transformation matrix to project different types of node features into the same space. Then the node-level attention is able to learn the attention values between the nodes and their meta-path based neighbors, while the semantic-level attention aims to learn the attention values of different meta-paths for the specific task in the heterogeneous graph. Based on the learned attention values in terms of the two levels, our model can get the optimal combination of neighbors and multiple meta-paths in a hierarchical manner, which enables the learned node embeddings to better capture the complex structure and rich semantic information in a heterogeneous graph. After that, the overall model can be optimized via backpropagation in an end-to-end manner.

The contributions of our work are summarized as follows:

• To our best knowledge, this is the first attempt to study the heterogeneous graph neural network based on attention mechanism. Our work enables the graph neural network to be directly applied to the heterogeneous graph, and further facilitates the heterogeneous graph based applications.

• We propose a novel heterogeneous graph attention network (HAN) which includes both of the node-level and semantic-level attentions. Benefitting from such hierarchical attentions, the proposed HAN can take the importance of nodes and meta-paths into consideration simultaneously. Moreover, our model is high efficiency, with the linear complexity with respect to the number of meta-path based node pairs, which can be applied to large-scale heterogeneous graph.

• We conduct extensive experiments to evaluate the performance of the proposed model. The results show the superiority of the proposed model by comparing with the state-of-the-art models. More importantly, by analysing the hierarchical attention mechanism, the proposed HAN demonstrates its potentially good interpretability for heterogeneous graph analysis.

2 RELATED WORK

2.1 Graph Neural Network

Graph neural networks (GNNs) which aim to extend the deep neural network to deal with arbitrary graph-structured data are introduced in [10, 24]. Yujia Li et al. [20] proposes a propagation model which can incorporate gated recurrent units to propagate information across all nodes. Recently, there is a surge of generalizing convolutional operation on the graph-structured data. The graph convolutional neural work generally falls into two categories, namely spectral domain and non-spectral domain. On one hand, spectral approaches work with a spectral representation of the graphs. Joan Bruna et al. [2] extends convolution to general graphs by finding the corresponding Fourier basis. Michaël et al. [6] utilizes K-order Chebyshev polynomials to approximate smooth filters in the spectral domain. Kipf et al. [18] proposes a spectral approach, named Graph Convolutional Network, which designs a graph convolutional network via a localized first-order approximation of spectral graph convolutions. On the other hand, we also have non-spectral

approaches, which define convolutions directly on the graph, operating on groups of spatially close neighbors. Hamilton et al. [14] introduces GraphSAGE which performs a neural network based aggregator over a fixed size node neighbor. It can learn a function that generates embeddings by aggregating features from a node's local neighborhood.

Attention mechanisms, e.g., self-attention [34] and soft-attention [1], have become one of the most influential mechanisms in deep learning. Some previous works introduce the attention mechanism for graph based applications, e.g., the recommendation [15, 16]. Inspired by attention mechanism, Graph Attention Network [35] is proposed to learn the importance between nodes and its neighbors and fuse the neighbors to perform node classification. However, the above graph neural network cannot deal with various types of nodes and edges and can only be applied to the homogeneous graphs.

2.2 Network Embedding

Network embedding, i.e., network representation learning (NRL), is proposed to embed network into a low dimensional space while preserving the network structure and property so that the learned embeddings can be applied to the downstream network tasks. For example, the random walk based methods [12, 23], the deep neural network based methods [36], the matrix factorization based methods [22, 37], and others, e.g., LINE [33]. However, all these algorithms are proposed for the homogeneous graphs. Some elaborate reviews can be found in [5, 11].

Heterogeneous graph embedding mainly focuses on preserving the meta-path based structural information. ESim [26] accepts user-defined meta-paths as guidance to learn vertex vectors in a user-preferred embedding space for similarity search. Even through ESim can utilize multiple meta-paths, it cannot learn the importance of meta-paths. To achieve the best performance, ESim needs to conduct grid search to find the optimal weights of hmeta-paths. It is pretty hard to find the optimal combination for specific task. Metapath2vec [7] designs a meta-path based random walk and utilizes skip-gram to perform heterogeneous graph embedding. However, metapath2vec can only utilize one meta-path and may ignore some useful information. Similar to metapath2vec, HERec [27] proposes a type constraint strategy to filter the node sequence and capture the complex semantics reflected in heterogeneous graph. HIN2Vec [9] carries out multiple prediction training tasks which learn the latent vectors of nodes and meta-paths simultaneously. Chen et al. [3] proposes a projected metric embedding model, named PME, which can preserve node proximities via Euclidian Distance. PME projects different types of node into the same relation space and conducts heterogeneous link prediction. To study the problem of comprehensive describe heterogeneous graph, Chen et al. [29] proposes HEER which can embed heterogeneous graph via edge representations. Fan et al. [8] proposes a embedding model metagraph2vec, where both the structures and semantics are maximally preserved for malware detection. Sun et al. [30] proposes meta-graph-based network embedding models, which simultaneously considers the hidden relations of all meta information of a meta-graph. In summary, all these aforementioned algorithms do not consider the attention mechanism in heterogeneous graph representation learning.

3 PRELIMINARY

A heterogeneous graph is a special kind of information network, which contains either multiple types of objects or multiple types of links.

Definition 3.1. **Heterogeneous Graph** [31]. A heterogeneous graph, denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consists of an object set $\mathcal{V}$ and a link set $\mathcal{E}$. A heterogeneous graph is also associated with a node type mapping function $\phi : \mathcal{V} \rightarrow \mathcal{A}$ and a link type mapping function $\psi : \mathcal{E} \rightarrow \mathcal{R}$. $\mathcal{A}$ and $\mathcal{R}$ denote the sets of predefined object types and link types, where $|\mathcal{A}| + |\mathcal{R}| > 2$.

Example. As shown in Figure 1(a), we construct a heterogeneous graph to model the IMDB. It consists of multiple types of objects (Actor (A), Movie (M), Director (D)) and relations (shoot relation between movies and directors, role-play relation between actors and movies).

In heterogeneous graph, two objects can be connected via different semantic paths, which are called meta-paths.

Definition 3.2. **Meta-path** [32]. A meta-path Φ is defined as a path in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1}$ (abbreviated as $A_1A_2 \cdots A_{l+1}$), which describes a composite relation $R = R_1 \circ R_2 \circ \cdots \circ R_l$ between objects A_1 and A_{l+1}, where $\circ$ denotes the composition operator on relations.

Example. As shown in Figure 1(a), two movies can be connected via multiple meta-paths, e.g., Movie-Actor-Movie (*MAM*) and Movie-Director-Movie (*MDM*). Different meta-paths always reveal different semantics. For example, the *MAM* means the co-actor relation, while Movie-Director-Movie (*MDM*) means they are directed by the same director.

Given a meta-path Φ, there exists a set of meta-path based neighbors of each node which can reveal diverse structure information and rich semantics in a heterogeneous graph.

Definition 3.3. **Meta-path based Neighbors.** Givien a node i and a meta-path Φ in a heterogeneous graph, the meta-path based neighbors N_i^{Φ} of node i are defined as the set of nodes which connect with node i via meta-path Φ. Note that the node's neighbors includes itself.

Example. Taking Figure 1(d) as an example, given the meta-path Movie-Actor-Movie, the meta-path based neighbors of m_1 includes $m1$ (itself), m_2 and m_3. Similarly, the neighbors of m_1 based on meta-path Movie-Director-Movie includes m_1 and m_2. Obviously, meta-path based neighbors can exploit different aspects of structure information in heterogeneous graph. We can get meta-path based neighbors by the multiplication of a sequences of adjacency matrices.

Graph neural network has been proposed to deal with arbitrary graph-structured data. However, all of them are designed for homogeneous network [18, 35]. Since meta-path and meta-path based neighbors are two fundamental structures in a heterogeneous graph, next, we will present a novel graph neural network for heterogeneous graph data, which is able to exploit the subtle difference of nodes and meta-paths. The notations we will use throughout the article are summarized in Table 1.

Table 1: Notations and Explanations.

Notation	Explanation
Φ	Meta-path
$\mathbf{h}$	Initial node feature
$\mathbf{M}_{\phi}$	Type-specific transformation matrix
$\mathbf{h}'$	Projected node feature
e_{ij}^{Φ}	Importance of meta-path based node pair (i,j)
$\mathbf{a}_{\Phi}$	Node-level attention vector for meta-path Φ
α_{ij}^{Φ}	Weight of meta-path based node pair (i,j)
$\mathcal{N}_i^{\Phi}$	Meta-path based neighbors
$\mathbf{Z}_{\Phi}$	Semantic-specific node embedding
$\mathbf{q}$	Semantic-level attention vector
w_{Φ}	Importance of meta-path Φ
β_{Φ}	Weight of meta-path Φ
$\mathbf{Z}$	The final embedding

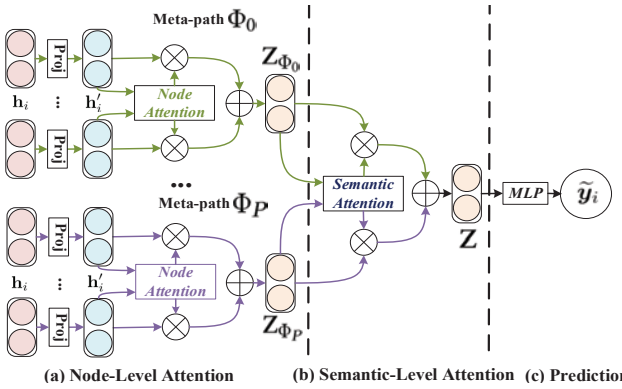


Figure 2: The overall framework of the proposed HAN. (a) All types of nodes are projected into a unified feature space and the weight of meta-path based node pair can be learned via node-level attention. (b) Joint learning the weight of each meta-path and fuse the semantic-specific node embedding via semantic-level attention. (c) Calculate the loss and end-to-end optimization for the proposed HAN.

4 THE PROPOSED MODEL

In this section, we propose a novel semi-supervised graph neural network for heterogeneous graph. Our model follows a hierarchical attention structure: node-level attention → semantic-level attention. Figure 2 presents the whole framework of HAN.

4.1 Node-level Attention

Before aggregating the information from meta-path neighbors for each node, we should notice that the meta-path based neighbors of each node play a different role and show different importance in learning node embedding for the specific task. Here we introduce node-level attention can learn the importance of meta-path based neighbors for each node in a heterogeneous graph and aggregate the representation of these meaningful neighbors to form a node embedding.

Due to the heterogeneity of nodes, different types of nodes have different feature spaces. Therefore, for each type of nodes (e.g.,node with type ϕ_i), we design the type-specific transformation matrix $\mathbf{M}_{\phi_i}$ to project the features of different types of nodes into the same feature space. Unlike [13], the type-specific transformation matrix is based on node-type rather than edge-type. The projection process can be shown as follows:

$$\mathbf{h}_i' = \mathbf{M}_{\phi_i} \cdot \mathbf{h}_i, \tag{1}$$

where $\mathbf{h}_i$ and $\mathbf{h}_i'$ are the original and projected feature of node i, respectively. By type-specific projection operation, the node-level attention can handle arbitrary types of nodes.

After that, we leverage self-attention [34] to learn the weight among various kinds of nodes. Given a node pair (i, j) which are connected via meta-path Φ, the node-level attention e_{ij}^{Φ} can learn the importance e_{ij}^{Φ} which means how important node j will be for node i. The importance of meta-path based node pair (i, j) can be formulated as follows:

$$e_{ij}^{\Phi} = att_{node}(\mathbf{h}_i', \mathbf{h}_j'; \Phi). \tag{2}$$

Here att_{node} denotes the deep neural network which performs the node-level attention. Given meta-path Φ, att_{node} is shared for all meta-path based node pairs. It is because there are some similar connection patterns under one meta-path. The above Eq. (2) shows that given meta-path Φ, the weight of meta-path based node pair (i, j) depends on their features. Please note that, e_{ij}^{Φ} is asymmetric, i.e., the importance of node i to node j and the importance of node j to node i can be quite difference. It shows node-level attention can preserve the asymmetry which is a critical property of heterogenous graph.

Then we inject the structural information into the model via masked attention which means we only calculate the e_{ij}^{Φ} for nodes $j \in \mathcal{N}_i^{\Phi}$, where $\mathcal{N}_i^{\Phi}$ denotes the meta-path based neighbors of node i (include itself). After obtaining the importance between meta-path based node pairs, we normalize them to get the weight coefficient α_{ij}^{Φ} via softmax function:

$$\alpha_{ij}^{\Phi} = softmax_j(e_{ij}^{\Phi}) = \frac{\exp\left(\sigma(\mathbf{a}_{\Phi}^{\mathrm{T}} \cdot [\mathbf{h}_i' \| \mathbf{h}_j'])\right)}{\sum_{k \in \mathcal{N}_i^{\Phi}} \exp\left(\sigma(\mathbf{a}_{\Phi}^{\mathrm{T}} \cdot [\mathbf{h}_i' \| \mathbf{h}_k'])\right)}, \tag{3}$$

where σ denotes the activation function, $\|$ denotes the concatenate operation and $\mathbf{a}_{\Phi}$ is the node-level attention vector for meta-path Φ. As we can see from Eq. (3), the weight coefficient of (i, j) depends on their features. Also please note that the weight coefficient α_{ij}^{Φ} is asymmetric which means they make different contribution to each other. Not only because the concatenate order in the numerator, but also because they have different neighbors so the normalize term (denominator) will be quite difference.

Then, the meta-path based embedding of node i can be aggregated by the neighbor's projected features with the corresponding coefficients as follows:

$$\mathbf{z}_i^{\Phi} = \sigma\left(\sum_{j \in \mathcal{N}_i^{\Phi}} \alpha_{ij}^{\Phi} \cdot \mathbf{h}_j'\right). \tag{4}$$

where $\mathbf{z}_i^{\Phi}$ is the learned embedding of node i for the meta-path Φ. To better understand the aggregating process of node-level, we

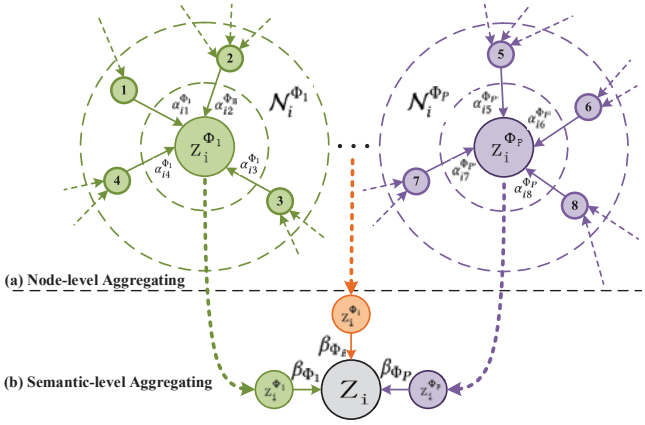


Figure 3: Explanation of aggregating process in both node-level and semantic-level.

also give a brief explanation in Figure 3 (a). Every node embedding is aggregated by its neighors. Since the attention weight α_{ij}^{Φ} is generated for single meta-path, it is semantic-specific and able to caputre one kind of semantic information.

Since heterogeneous graph present the property of scale free, the variance of graph data is quite high. To tackle the above challenge, we extend node-level attention to multihead attention so that the training process is more stable. Specifically, we repeat the node-level attention for K times and concatenate the learned embeddings as the semantic-specific embedding:

$$\mathbf{z}_i^{\Phi} = \Big\Vert_{k=1}^{K} \sigma\Big(\sum_{j\in\mathcal{N}_i^{\Phi}} \alpha_{ij}^{\Phi} \cdot \mathbf{h}_j'\Big). \tag{5}$$

Given the meta-path set $\{\Phi_0, \Phi_1, \ldots, \Phi_P\}$, after feeding node features into node-level attention, we can obtain P groups of semantic-specific node embeddings, denoted as $\{\mathbf{Z}_{\Phi_0}, \mathbf{Z}_{\Phi_1}, \ldots, \mathbf{Z}_{\Phi_P}\}$.

4.2 Semantic-level Attention

Generally, every node in a heterogeneous graph contains multiple types of semantic information and semantic-specific node embedding can only reflect node from one aspect. To learn a more comprehensive node embedding, we need to fuse multiple semantics which can be revealed by meta-paths. To address the challenge of meta-path selection and semantic fusion in a heterogeneous graph, we propose a novel semantic-level attention to automatically learn the importance of different meta-paths and fuse them for the specific task. Taking P groups of semantic-specific node embeddings learned from node-level attention as input, the learned weights of each meta-path $(\beta_{\Phi_0}, \beta_{\Phi_1}, \ldots, \beta_{\Phi_P})$ can be shown as follows:

$$(\beta_{\Phi_0}, \beta_{\Phi_1}, \ldots, \beta_{\Phi_P}) = att_{sem}(\mathbf{Z}_{\Phi_0}, \mathbf{Z}_{\Phi_1}, \ldots, \mathbf{Z}_{\Phi_P}). \tag{6}$$

Here att_{sem} denotes the deep neural network which performs the semantic-level attention. It shows that the semantic-level attention can capture various types of semantic information behind a heterogeneous graph.

To learn the importance of each meta-path, we first transform semantic-specific embedding through a nonlinear transformation (e.g., one-layer MLP). Then we measure the importance of the semantic-specific embedding as the similarity of transformed embedding with a semantic-level attention vector $\mathbf{q}$. Furthermore, we average the importance of all the semantic-specific node embedding which can be explained as the importance of each meta-path. The importance of each meta-path, denoted as w_{Φ_i}, is shown as follows:

$$w_{\Phi_i} = \frac{1}{|\mathcal{V}|} \sum_{i\in\mathcal{V}} \mathbf{q}^{\mathrm{T}} \cdot \tanh(\mathbf{W} \cdot \mathbf{z}_i^{\Phi} + \mathbf{b}), \tag{7}$$

where $\mathbf{W}$ is the weight matrix, $\mathbf{b}$ is the bias vector, $\mathbf{q}$ is the semantic-level attention vector. Note that for the meaningful comparation, all above parameters are shared for all meta-paths and semantic-specific embedding. After obtaining the importance of each meta-path, we normalize them via softmax function. The weight of meta-path Φ_i, denoted as β_{Φ_i}, can be obtained by normalizing the above importance of all meta-paths using softmax function,

$$\beta_{\Phi_i} = \frac{\exp(w_{\Phi_i})}{\sum_{i=1}^{P} \exp(w_{\Phi_i})}, \tag{8}$$

which can be interpreted as the contribution of the meta-path Φ_i for specific task. Obviously, the higher β_{Φ_i}, the more important meta-path Φ_i is. Note that for different tasks, meta-path Φ_i may has different weights. With the learned weights as coefficients, we can fuse these semantic-specific embeddings to obtain the final embedding $\mathbf{Z}$ as follows:

$$\mathbf{Z} = \sum_{i=1}^{P} \beta_{\Phi_i} \cdot \mathbf{Z}_{\Phi_i}. \tag{9}$$

To better understand the aggregating process of semantic-level, we also give a brief explanation in Figure 3 (b). The final embedding is aggregated by all semantic-specific embedding. Then we can apply the final embedding to specific tasks and design different loss fuction. For semi-supervised node classification, we can minimize the Cross-Entropy over all labeled node between the ground-truth and the prediction:

$$L = -\sum_{l\in\mathcal{Y}_L} \mathbf{Y}^l \ln(\mathbf{C} \cdot \mathbf{Z}^l), \tag{10}$$

where $\mathbf{C}$ is the parameter of the classifier, $\mathcal{Y}_L$ is the set of node indices that have labels, $\mathbf{Y}^l$ and $\mathbf{Z}^l$ are the labels and embeddings of labeled nodes. With the guide of labeled data, we can optimize the proposed model via back propagation and learn the embeddings of nodes. The overall process of HAN in shown in Algorithm 1.

4.3 Analysis of the Proposed Model

Here we give the analysis of the proposed HAN as follows:

• The proposed model can deal with various types of nodes and relations and fuse rich semantics in a heterogeneous graph. The information can transfer from one kind of nodes to another kind of nodes via diverse relation. Benefitted from such a heterogeneous graph attention network, different types of node embedding can enhance the mutual integration, mutual promotion and mutual upgrade.

• The proposed HAN is highly efficient and can be easily parallelized. The computation of attention can compute individually across all nodes and meta-paths. Given a meta-path Φ, the time

Table 2: Statistics of the datasets.

Dataset	Relations(A-B)	Number of A	Number of B	Number of A-B	Feature	Training	Validation	Test	Meta-paths
DBLP	Paper-Author	14328	4057	19645	334	800	400	2857	*APA*
	Paper-Conf	14328	20	14328					*APCPA*
	Paper-Term	14327	8789	88420					*APTPA*
IMDB	Movie-Actor	4780	5841	14340	1232	300	300	2687	*MAM*
	Movie-Director	4780	2269	4780					*MDM*
ACM	Paper-Author	3025	5835	9744	1830	600	300	2125	*PAP*
	Paper-Subject	3025	56	3025					*PSP*

Algorithm 1: The overall process of HAN.

Input : The heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$,
The node feature $\{\mathbf{h}_i, \forall i \in \mathcal{V}\}$,
The meta-path set $\{\Phi_0, \Phi_1, \ldots, \Phi_P\}$.
The number of attention head K,
Output : The final embedding $\mathbf{Z}$,
The node-level attention weight α ,
The semantic-level attention weight β .

for $\Phi_i \in \{\Phi_0, \Phi_1, \ldots, \Phi_P\}$ **do**
  for $k = 1 \ldots K$ **do**
    Type-specific transformation $\mathbf{h}'_i \leftarrow \mathbf{M}_{\phi_i} \cdot \mathbf{h}_i$;
    for $i \in \mathcal{V}$ **do**
      Find the meta-path based neighbors $\mathcal{N}_i^\Phi$;
      for $j \in \mathcal{N}_i^\Phi$ **do**
        Calculate the weight coefficient α_{ij}^Φ ;
      end
      Calculate the semantic-specific node embedding $\mathbf{z}_i^\Phi \leftarrow \sigma\left(\sum_{j \in \mathcal{N}_i^\Phi} \alpha_{ij}^\Phi \cdot \mathbf{h}'_j\right)$;
    end
    Concatenate the learned embeddings from all attention head $\mathbf{z}_i^\Phi \leftarrow \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i^\Phi} \alpha_{ij}^\Phi \cdot \mathbf{h}'_j\right)$;
  end
  Calculate the weight of meta-path β_{Φ_i} ;
  Fuse the semantic-specific embedding $\mathbf{Z} \leftarrow \sum_{i=1}^{P} \beta_{\Phi_i} \cdot \mathbf{Z}_{\Phi_i}$;
end
Calculate Cross-Entropy $L = -\sum_{l \in \mathcal{Y}_L} \mathbf{Y}_l \ln(\mathbf{C} \cdot \mathbf{Z}_l)$;
Back propagation and update parameters in HAN;
return Z, α, β.

complexity of node-level attention is $O(V_\Phi F_1 F_2 K + E_\Phi F_1 K)$, where K is the number of attention head, V_Φ is the number of nodes, E_Φ is the number of meta-path based node pairs, F_1 and F_2 are the numbers of rows and columns of the transformation matrix, respectively. The overall complexity is linear to the number of nodes and meta-path based node pairs. The proposed model can be easily parallelized, because the node-level and semantic-level attention can be parallelized across node paris and meta-paths, respectively. The overall complexity is linear to the number of nodes and meta-path based node pairs.

• The hierarchical attention is shared for the whole heterogeneous graph which means the number of parameters is not dependent on the scale of a heterogeneous graph and can be used for the inductive problems [14]. Here inductive means the model can generate node embeddings for previous unseen nodes or even unseen graph.

• The proposed model has potentionally good interpretability for the learned node embedding which is a big advantage for heterogeneous graph analysis. With the learned importance of nodes and meta-paths, the proposed model can pay more attention to some meaningful nodes or meta-paths for the specific task and give a more comprensive description of a heterogeneous graph. Based on the attention values, we can check which nodes or meta-paths make the higher (or lower) contributions for our task, which is beneficial to analyze and explain our results.

5 EXPERIMENTS

5.1 Datasets

The detailed descriptions of the heterogeneous graph used here are shown in Table 2.

• **DBLP**[2]. We extract a subset of DBLP which contains 14328 papers (P), 4057 authors (A), 20 conferences (C), 8789 terms (T). The authors are divided into four areas: *database, data mining, machine learning, information retrieval*. Also, we label each author's research area according to the conferences they submitted. Author features are the elements of a bag-of-words represented of keywords. Here we employ the meta-path set {*APA*, *APCPA*, *APTPA*} to perform experiments.

• **ACM**[3]. We extract papers published in KDD, SIGMOD, SIGCOMM, MobiCOMM, and VLDB and divide the papers into three classes (*Database, Wireless Communication, Data Mining*). Then we construct a heterogeneous graph that comprises 3025 papers (P), 5835 authors (A) and 56 subjects (S). Paper features correspond to elements of a bag-of-words represented of keywords. We employ the meta-path set {*PAP*, *PSP*} to perform experiments. Here we label the papers according to the conference they published.

• **IMDB.** Here we extract a subset of IMDB which contains 4780 movies (M), 5841 actors (A) and 2269 directors (D). The movies are divided into three classes (*Action, Comedy, Drama*) according

[2]https://dblp.uni-trier.de
[3]http://dl.acm.org/

Table 3: Qantitative results (%) on the node classification task.

Datasets	Metrics	Training	DeepWalk	ESim	metapath2vec	HERec	GCN	GAT	HAN$_{nd}$	HAN$_{sem}$	HAN
ACM	Macro-F1	20%	77.25	77.32	65.09	66.17	86.81	86.23	88.15	89.04	**89.40**
		40%	80.47	80.12	69.93	70.89	87.68	87.04	88.41	89.41	**89.79**
		60%	82.55	82.44	71.47	72.38	88.10	87.56	87.91	**90.00**	89.51
		80%	84.17	83.00	73.81	73.92	88.29	87.33	88.48	90.17	**90.63**
	Micro-F1	20%	76.92	76.89	65.00	66.03	86.77	86.01	87.99	88.85	**89.22**
		40%	79.99	79.70	69.75	70.73	87.64	86.79	88.31	89.27	**89.64**
		60%	82.11	82.02	71.29	72.24	88.12	87.40	87.68	**89.85**	89.33
		80%	83.88	82.89	73.69	73.84	88.35	87.11	88.26	89.95	**90.54**
DBLP	Macro-F1	20%	77.43	91.64	90.16	91.68	90.79	90.97	91.17	92.03	**92.24**
		40%	81.02	92.04	90.82	92.16	91.48	91.20	91.46	92.08	**92.40**
		60%	83.67	92.44	91.32	92.80	91.89	90.80	91.78	92.38	**92.80**
		80%	84.81	92.53	91.89	92.34	92.38	91.73	91.80	92.53	**93.08**
	Micro-F1	20%	79.37	92.73	91.53	92.69	91.71	91.96	92.05	92.99	**93.11**
		40%	82.73	93.07	92.03	93.18	92.31	92.16	92.38	93.00	**93.30**
		60%	85.27	93.39	92.48	93.70	92.62	91.84	92.69	93.31	**93.70**
		80%	86.26	93.44	92.80	93.27	93.09	92.55	92.69	93.29	**93.99**
IMDB	Macro-F1	20%	40.72	32.10	41.16	41.65	45.73	49.44	49.78	**50.87**	50.00
		40%	45.19	31.94	44.22	43.86	48.01	50.64	52.11	50.85	**52.71**
		60%	48.13	31.68	45.11	46.27	49.15	51.90	51.73	52.09	**54.24**
		80%	50.35	32.06	45.15	47.64	51.81	52.99	52.66	51.60	**54.38**
	Micro-F1	20%	46.38	35.28	45.65	45.81	49.78	55.28	54.17	55.01	**55.73**
		40%	49.99	35.47	48.24	47.59	51.71	55.91	56.39	55.15	**57.97**
		60%	52.21	35.64	49.09	49.88	52.29	56.44	56.09	56.66	**58.32**
		80%	54.33	35.59	48.81	50.99	54.61	56.97	56.38	56.49	**58.51**

to their genre. Movie features correspond to elements of a bag-of-words represented of plots. We employ the meta-path set {*MAM*, *MDM*} to perform experiments.

5.2 Baselines

We compare with some state-of-art baselines, include the (heterogeneous) network embedding methods and graph neural network based methods, to verfify the effectiveness of the proposed HAN. To verify the effectiveness of our node-level attention and semantic-level attention, respectively, we also test two variants of HAN.

• DeepWalk [23]: A random walk based network embedding method which designs for the homogeneous graphs. Here we ignore the heterogeneity of nodes and perform DeepWalk on the whole heterogeneous graph.

• ESim [26]: A heterogeneous graph embedding method which can capture semantic information from multiple meta-paths. Because it is difficult to search the weights of a set of meta-paths, we assign the weights learned from HAN to ESim.

• metapath2vec [7]: A heterogeneous graph embedding method which performs meta-path based random walk and utilizes skip-gram to embed the heterogeneous graphs. Here we test all the meta-paths for metapath2vec and report the best performance.

• HERec [27]: A heterogeneous graph embedding method which designs a type constraint strategy to filter the node sequence and utilizes skip-gram to embed the heterogeneous graphs. Here we test all the meta-paths for HERec and report the best performance.

• GCN [18]: It is a semi-supervised graph convolutional network that designed for the homogeneous graphs. Here we test all the meta-paths for GCN and report the best performance.

• GAT [35]: It is a semi-supervised neural network which considers the attention mechanism on the homogeneous graphs. Here we test all the meta-paths for GAT and report the best performance.

• HAN$_{nd}$: It is a variant of HAN, which removes node-level attention and assigns the same importance to each neighbor.

• HAN$_{sem}$: It is a variant of HAN, which removes the semantic-level attention and assigns the same importance to each meta-path.

• HAN: The proposed semi-supervised graph neural network which employs node-level attention and semantic-level attention simultaneously.

5.3 Implementation Details

For the proposed HAN, we randomly initialize parameters and optimize the model with Adam [17]. For the proposed HAN, we set the learning rate to 0.005, the regularization parameter to 0.001, the dimension of the semantic-level attention vector $\mathbf{q}$ to 128, the number of attention head K to 8, the dropout of attention to 0.6. And we use early stopping with a patience of 100, i.e. we stop training if the validation loss does not decrease for 100 consecutive epochs. To make our experiments repeatable, we make our dataset and codes publicly available at website[4].For GCN and GAT, we optimize their parameters using the validation set. For semi-supervised graph neural network, including GCN, GAT and HAN, we split exactly

[4]https://github.com/Jhy1993/HAN

the same training set, validation set and test set to ensure fairness. For random walk based methods include DeepWalk, ESim, metapath2vec, and HERec, we set window size to 5, walk length to 100, walks per node to 40, the number of negative samples to 5. For a fair comparison, we set the embedding dimension to 64 for all the above algorithms.

5.4 Classification

Here we employ KNN classifier with $k = 5$ to perform node classification. Since the variance of graph-structured data can be quite high, we repeat the process for 10 times and report the averaged *Macro-F1* and *Micro-F1* in Table 3.

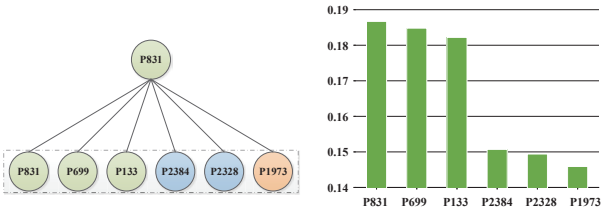
Based on Table 3, we can see that HAN achieves the best performance. For traditional heterogeneous graph embedding method, ESim which can leverage multiple meta-paths performs better than metapath2vec. Generally, graph neural network based methods which combine the structure and feature information, e.g., GCN and GAT, usually perform better. To go deep into these methods, compared to simply average over node neighbors, e.g., GCN and HAN_{nd}, GAT and HAN can weigh the information properly and improve the performance of the learned embedding. Compared to GAT, the proposed HAN, which designs for heterogeneous graph, captures the rich semantics successfully and shows its superiority. Also, without node-level attention (HAN_{nd}) or semantic-level attention (HAN_{sem}), the performance becomes worse than HAN, which indicates the importance of modeling the attention mechanism on both of the nodes and semantics. Note that in ACM and IMDB, HAN improves classification results more significantly than in DBLP. Mainly because *APCPA* is the much more important than the rest meta-paths. We will explain this phenomenon in Section 5.7 by analyzing the semantic-level attention.

Through the above analysis, we can find that the proposed HAN achieves the best performance on all datasets. The results demonstrate that it is quite important to capture the importance of nodes and meta-paths in heterogeneous graph analysis.

5.5 Clustering

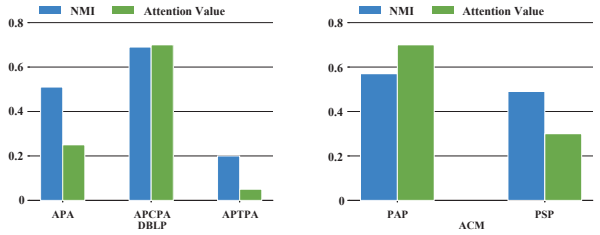
We also conduct the clustering task to evaluate the embeddings learned from the above algorithms. Once the proposed HAN trained, we can get all the node embedding via feed forward. Here we utilize the KMeans to perform node clustering and the number of clusters K is set to the number of classes. We use the same ground-truth as in node classification. And we adopt *NMI* and *ARI* to assess the quality of the clustering results. Since the performance of KMeans is affected by initial centroids, we repeat the process for 10 times and report the average results in Table 4.

As can be seen in Table 4, we can find that HAN performs consistently much better than all baselines. Also, graph neural network based algorithms usually achieve better performance. Besides, without distinguishing the importance of nodes or meta-paths, metapath2vec and GCN cannot perform well. With the guide of multiple meta-paths, HAN performs significantly better than GCN and GAT. On the other hand, without node-level attention (HAN_{nd}) or semantic-level attention (HAN_{sem}), the performance of HAN has shown various degrees of degeneration. It demonstrates that



(a) Meta-path based neighbors of P831 (b) Attention values of P831's neighbors

Figure 4: Meta-path based neighbors of node P831 and corresponding attention values (Different colors mean different classes, e.g., *green* means Data Mining, *blue* means Database, *orange* means Wireless Communication).



(a) NMI values on DBLP (b) NMI values on ACM

Figure 5: Performance of single meta-path and corresponding attention value.

via assigning the different importance to nodes and meta-paths, the proposed HAN can learn a more meaningful node embedding.

Based on the above analysis, we can find that the propsed HAN can give a comprehensive description of heterogeneous graph and achieve a significant improvements.

5.6 Analysis of Hierarchical Attention Mechanism

A salient property of HAN is the incorporation of the hierarchical mechanism, which takes the importance of node neighbors and meta-paths into consideration in learning representative embedding. Recall that we have learned the node-level attention weight α_{ij}^{Φ} and the semantic-level attention weight β_{Φ_i}. To better understand the importance of the neighbors and meta-paths, we provide a detailed analysis on the hierarchical attention mechanism.

Analysis of node-level attention. As mentioned before, given a specific task, our model can learn the attention values between nodes and its neighbors in a meta-path. Some important neighbors which are useful for the specific task tend to have larger attention values. Here we take the paper P831 [5] in ACM dataset as an illustrative example. Given a meta-path Paper-Author-Paper which describes the co-author of different papers, we enumerate the meta-path based neighbors of paper P831 and their attention values are

[5]Xintao Wu, Daniel Barbara, Yong Ye. Screening and Interpreting Multi-item Associations Based on Log-linear Modeling, KDD'03

Table 4: Qantitative results (%) on the node clustering task.

Datasets	Metrics	DeepWalk	ESim	metapath2vec	HERec	GCN	GAT	HAN$_{nd}$	HAN$_{sem}$	HAN
ACM	NMI	41.61	39.14	21.22	40.70	51.40	57.29	60.99	61.05	**61.56**
	ARI	35.10	34.32	21.00	37.13	53.01	60.43	61.48	59.45	**64.39**
DBLP	NMI	76.53	66.32	74.30	76.73	75.01	71.50	75.30	77.31	**79.12**
	ARI	81.35	68.31	78.50	80.98	80.49	77.26	81.46	83.46	**84.76**
IMDB	NMI	1.45	0.55	1.20	1.20	5.45	8.45	9.16	10.31	**10.87**
	ARI	2.15	0.10	1.70	1.65	4.40	7.46	7.98	9.51	**10.01**

shown in Figure 4. From Figure 4(a), we can see that P831 connects to P699 [6] and P133 [7], which all belong to *Data Mining*; conects to P2384 [8] and P2328 [9] while P2384 and P2328 both belong to *Database*; connects to P1973 [10] while P1973 belongs to *Wireless Communication*. From Figure 4(b), we can see that paper P831 gets the highest attention value from node-level attention which means the node itself plays the most important role in learning its representation. It is reasonable because all information supported by neighbors are usually viewed as a kind of supplementary information. Beyond itself, P699 and P133 get the second and third largest attention values. This is because P699 and P133 also belong to *Data Mining* and they can make significant contribution to identify the class of P831. The rest neighbors get minor attention values that because they do not belong to *Data Mining* and cannot make important contribution to identify the P831's class. Based on the above analysis, we can see that the node-level attention can tell the difference among neighbors and assign higher weights to some meaningful neighbors.

Analysis of semantic-level attention. As mentioned before, the proposed HAN can learn the importance of meta-paths for the specific task. To verify the ability of semantic-level attention, taking DBLP and ACM as examples, we report the clustering results (*NMI*) of single meta-path and corresponding attention values in Figure 5. Obviously, there is a positive correlation between the performance of a single meta-path and its attention value. For DBLP, HAN gives *APCPA* the largest weight, which means that HAN considers the *APCPA* as the most critical meta-path in identifying the author's research area. It makes sense because the author's research area and the conferences they submitted are highly correlated. For example, some natural language processing researchers mainly submit their papers to ACL or EMNLP, whereas some data mining researchers may submit their papers to KDD or WWW. Meanwhile, it is difficult for *APA* to identify the author's research area well. If we treat these meta-paths equally, e.g., HAN$_{sem}$, the performance will drop significantly. Based on the attention values of each meta-path, we can find that the meta-path *APCPA* is much more useful than *APA* and *APTPA*. So even the proposed HAN can fuse them, *APCPA* still plays a leading role in identifying the author's research area while *APA* and *APTPA* do not. It also explains why the performance of HAN in DBLP may not be as significant as in ACM and IMDB. We get the similar conclusions on ACM. For ACM, the results show that HAN gives the most considerable weight to *PAP*. Since the performance of *PAP* is slightly better than *PSP*, so HAN$_{sem}$ can achieve good performance by simple average operation. We can see that semantic-level attention can reveal the difference between these meta-paths and weights them adequately.

5.7 Visualization

For a more intuitively comparation, we conduct the task of visualization, which aims to layout a heterogeneous graph on a low dimensional space. Specifically, we learn the node embedding based on the proposed model and project the learned embedding into a 2-dimensional space. Here we utilize t-SNE [21] to visualize the author embedding in DBLP and coloured the nodes based on their research areas.

From Figure 6, we can find that GCN and GAT which design for the homogeneous graphs do not perform well. The authors belong to different research areas are mixed with each other. Metapath2vec performs much better than the above homogeneous graph neural networks. It demonstrates that the proper meta-path(e.g., *APCPA*) can make a significant contribution to heterogeneous graph analysis. However, since metapath2vec can only take only one meta-path into consideration, the boundary is still blurry. From Figure 6, we can see that the visualization of HAN peform best. With the guide of multiple meta-paths, the embedding learned by HAN has high intra-class similarity and separates the authors in different research area with distinct boundaries.

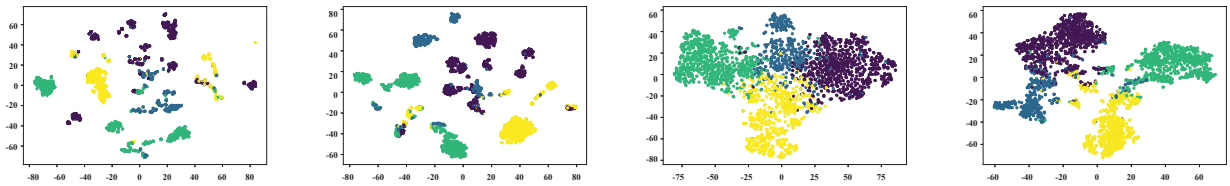
5.8 Parameters Experiments

In this section, we investigate the sensitivity of parameters and report the results of clustering (*NMI*) on ACM dataset with various parameters in Figure 7.

• **Dimension of the final embedding Z.** We first test the effect of the dimension of the final embedding $\mathbf{Z}$. The result is shown in Figure 7(a). We can see that with the growth of the embedding dimension, the performance raises first and then starts to drop slowly. The reason is that HAN needs a suitable dimension to encode the semantics information and larger dimension may introduce additional redundancies.

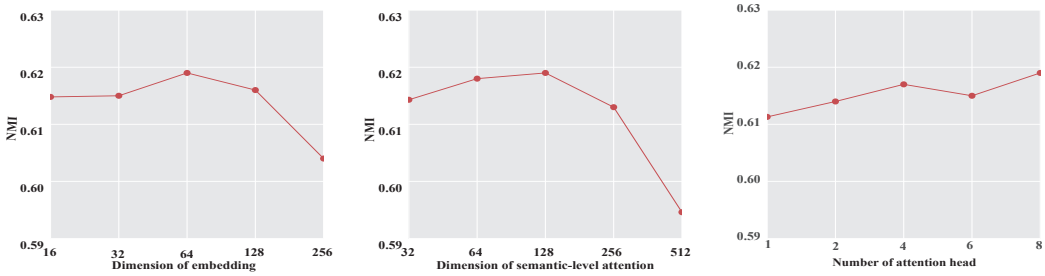
• **Dimension of semantic-level attention vector q.** Since the ability of semantic-level attention is affected by the dimension of the semantic-level attention vector $\mathbf{q}$, we explore the experimental results with various dimension. The result is shown in Figure 7(b). We can find that the performance of HAN grows with the

[6]Xintao Wu, Jianpin Fan, Kalpathi Subramanian. B-EM: a classifier incorporating bootstrap with EM approach for data mining, KDD'02
[7]Daniel Barbara, Carlotta Domeniconi, James P. Rogers. Detecting outliers using transduction and statistical testing, KDD'06
[8]Walid G. Aref, Daniel Barbara, Padmavathi Vallabhaneni. The Handwritten Trie: Indexing Electronic Ink, SIGMOD'95
[9]Daniel Barbara, Tomasz Imielinski. Sleepers and Workaholics: Caching Strategies in Mobile Environments, VLDB'95
[10]Hector Garcia-Holina, Daniel Barbara. The cost of data replication, SIGCOMM'81



(a) GCN (b) GAT (c) metapath2vec (d) HAN

Figure 6: Visualization embedding on DBLP. Each point indicates one author and its color indicates the research area.



(a) Dimension of the final embedding $\mathbf{Z}$ (b) Dimension of the semantic-level attention vector $\mathbf{q}$ (c) Number of attention head K.

Figure 7: Parameter sensitivity of HAN w.r.t. Dimension of the final embedding Z, Dimension of the semantic-level attention vector q and Number of attention head K.

dimension of semantic-level attention vector and achieves the best performance when the dimension of $\mathbf{q}$ is set to 128. After that, the performance of HAN starts to degenerate which may because of overfitting.

• **Number of attention head** K**.** In order to check the impact of multihead attention, we explore the performance of HAN with various number of attention head. The result is shown in Figure 7(c). Note that the multihead attention is removed when the number of attention head is set to 1. Based on the results, we can find that the more number of attention head will generally improve the performance of HAN. However, with the change of attention head, the performance of HAN improve only slightly. Meanwhile, we also find that multihead attention can make the training process more stable.

6 CONCLUSION

In this paper, we tackle several fundamental problems in heterogeneous graph analysis and propose a semi-supervised heterogeneous graph neural network based solely on attention mechanism. The proposed HAN can capture complex structures and rich semantics behind heterogeneous graph. The proposed model leverages node-level attention and semantic-level attention to learn the importance of nodes and meta-paths, respectively. Meanwhile, the proposed model utilizes the structural information and the feature information in a uniform way. Experimental results include classification and clustering demonstrate the effectiveness of HAN. By analyzing the learned attention weights include both node-level and semantic-level, the proposed HAN has proven its potentially good interpretability.

7 ACKNOWLEDGMENTS

This work is supported in part by the National Natural Science Foundation of China (No. 61702296, 61772082, 61532006), the Beijing Municipal Natural Science Foundation (4182043), and the CCF-Tencent Open Fund.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR* (2015).
[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203* (2013).
[3] Hongxu Chen, Hongzhi Yin, Weiqing Wang, Hao Wang, Quoc Viet Hung Nguyen, and Xue Li. 2018. PME: Projected Metric Embedding on Heterogeneous Networks for Link Prediction. In *SIGKDD*. 1177–1186.
[4] Ting Chen and Yizhou Sun. 2017. Task-guided and path-augmented heterogeneous network embedding for author identification. In *WSDM*. 295–304.
[5] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2018. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering* (2018).
[6] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*. 3844–3852.
[7] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *SIGKDD*. 135–144.
[8] Yujie Fan, Shifu Hou, Yiming Zhang, Yanfang Ye, and Melih Abdulhayoglu. 2018. Gotcha-sly malware!: Scorpion a metagraph2vec based malware detection system. In *SIGKDD*. 253–262.

[9] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. HIN2Vec: Explore Meta-paths in Heterogeneous Information Networks for Representation Learning. In *CIKM*. 1797–1806.

[10] Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. In *IJCNN*, Vol. 2. 729–734.

[11] Palash Goyal and Emilio Ferrara. 2017. Graph embedding techniques, applications, and performance: A survey. *arXiv preprint arXiv:1705.02801* (2017).

[12] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *SIGKDD*. 855–864.

[13] Will Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. 2018. Embedding logical queries on knowledge graphs. In *Advances in Neural Information Processing Systems*. 2030–2041.

[14] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*. 1024–1034.

[15] Xiaotian Han, Chuan Shi, Senzhang Wang, S Yu Philip, and Li Song. 2018. Aspect-Level Deep Collaborative Filtering via Heterogeneous Information Networks.. In *IJCAI*. 3393–3399.

[16] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging Meta-path based Context for Top-N Recommendation with A Neural Co-Attention Model. In *SIGKDD*. 1531–1540.

[17] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *ICLR* (2015).

[18] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[19] Xiang Li, Yao Wu, Martin Ester, Ben Kao, Xin Wang, and Yudian Zheng. 2017. Semi-supervised clustering in attributed heterogeneous information networks. In *WWW*. 1621–1629.

[20] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2016. Gated graph sequence neural networks. *ICLR* (2016).

[21] Laurens Van Der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, 2605 (2008), 2579–2605.

[22] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *SIGKDD*. 1105–1114.

[23] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*. 701–710.

[24] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks* 20, 1 (2009), 61–80.

[25] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*. Springer, 593–607.

[26] Jingbo Shang, Meng Qu, Jialu Liu, Lance M. Kaplan, Jiawei Han, and Jian Peng. 2016. Meta-Path Guided Embedding for Similarity Search in Large-Scale Heterogeneous Information Networks. *CoRR* abs/1610.09769 (2016).

[27] Chuan Shi, Binbin Hu, Xin Zhao, and Philip Yu. 2018. Heterogeneous Information Network Embedding for Recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[28] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and Philip S. Yu. 2017. A Survey of Heterogeneous Information Network Analysis. *IEEE Transactions on Knowledge and Data Engineering* 29 (2017), 17–37.

[29] Yu Shi, Qi Zhu, Fang Guo, Chao Zhang, and Jiawei Han. 2018. Easing Embedding Learning by Comprehensive Transcription of Heterogeneous Information Networks. In *SIGKDD*. ACM, 2190–2199.

[30] Lichao Sun, Lifang He, Zhipeng Huang, Bokai Cao, Congying Xia, Xiaokai Wei, and S Yu Philip. 2018. Joint embedding of meta-path and meta-graph for heterogeneous information networks. In *2018 IEEE International Conference on Big Knowledge (ICBK)*. 131–138.

[31] Yizhou Sun and Jiawei Han. 2013. Mining heterogeneous information networks: a structural analysis approach. *Acm Sigkdd Explorations Newsletter* 14, 2 (2013), 20–28.

[32] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *VLDB* 4, 11 (2011), 992–1003.

[33] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW*. 1067–1077.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*. 5998–6008.

[35] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *ICLR* (2018).

[36] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *SIGKDD*. 1225–1234.

[37] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community Preserving Network Embedding.. In *AAAI*. 203–209.

[38] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*. 2048–2057.

Aspect-level Sentiment Analysis using AS-Capsules

Yequan Wang*
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems
Beijing National Research Center for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
tshwangyequan@gmail.com

Aixin Sun
School of Computer Science and Engineering, Nanyang Technological University, Singapore
axsun@ntu.edu.sg

Minlie Huang
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems
Beijing National Research Center for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
aihuang@tsinghua.edu.cn

Xiaoyan Zhu
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems
Beijing National Research Center for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
zxy-dcs@tsinghua.edu.cn

ABSTRACT

Aspect-level sentiment analysis aims to provide complete and detailed view of sentiment analysis from different aspects. Existing solutions usually adopt a two-staged approach: first detecting aspect category in a document, then categorizing the polarity of opinion expressions for detected aspect(s). Inevitably, such methods lead to error accumulation. Moreover, aspect detection and aspect-level sentiment classification are highly correlated with each other. The key issue here is how to perform aspect detection and aspect-level sentiment classification jointly, and effectively. In this paper, we propose the *aspect-level sentiment capsules* model (**AS-Capsules**), which is capable of performing aspect detection and sentiment classification simultaneously, in a joint manner. AS-Capsules utilizes the correlation between aspect and sentiment through shared components including capsule embedding, shared encoders, and shared attentions. AS-Capsules is also capable of communicating with different capsules through a shared Recurrent Neural Network (RNN). More importantly, AS-Capsules model does not require any linguistic knowledge as additional input. Instead, through the attention mechanism, this model is able to attend aspect related words and sentiment words corresponding to different aspect(s). Experiments show that the AS-Capsules model achieves state-of-the-art performances on a benchmark dataset for aspect-level sentiment analysis.

ACM Reference Format:
Yequan Wang, Aixin Sun, Minlie Huang, and Xiaoyan Zhu. 2019. Aspect-level Sentiment Analysis using AS-Capsules. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313750

*This work was done when Yequan was a visiting Ph.D student at School of Computer Science and Engineering, Nanyang Technological University, Singapore.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313750

1 INTRODUCTION

Sentiment analysis aims to analyze people's sentiments, opinions, evaluations, attitudes, and emotions from human languages [17, 24]. Current researches focus on document-level (*e.g.,* document, paragraph, and sentence), or in-depth aspect-level analysis. As a fine-grained task, aspect-level sentiment analysis provides complete and detailed view of sentiments from different aspects. In general, this task requires aspect detection and aspect-level sentiment classification. Most existing solutions first detect aspect category in a sentence, and then categorize the polarity of opinion expressions with respect to the detected aspect(s). In other words, the two subtasks are tackled separately. As a result, errors made in aspect detection would affect aspect-level sentiment classification. On the other hand, the two subtasks, *i.e.,* aspect detection and aspect-level sentiment classification, are highly correlated with each other.

In this paper, we propose the *aspect-level sentiment capsules* (**AS-Capsules**) model. This model utilizes the correlation between aspects and corresponding sentiments. Hence, we jointly perform the two subtasks: aspect detection and aspect-level sentiment classification.

The Research Problem. In aspect-level sentiment analysis, we have a predefined set of aspects $\mathcal{A} = \{a_1, a_2, \cdots, a_M\}$, and a predefined set of sentiment polarities $\mathcal{P} = \{o_1, o_2, \cdots, o_P\}$. Given a piece of text (*e.g.,* a sentence or a paragraph), denoted by $S = [w_1, w_2, \ldots, w_N]$, the task is to predict the aspect(s) and the corresponding sentiment(s), *i.e.,* aspect-sentiment pairs $\{\langle a_i, o_i \rangle\}$, expressed in the text.

Consider the task is to be conducted on restaurant reviews. The set of aspects $\mathcal{A}$ can be {*food, service, price, ambience, anecdote*}, and the set of sentiment polarities $\mathcal{P}$ may include {*positive*, *neutral*, *negative*}. Each restaurant review is considered a piece of text S.

Given an example review "Staffs are not that friendly, but the taste covers all.", the expected output will be aspect-sentiment pairs {⟨food, positive⟩, ⟨service, negative⟩}. Accordingly, such pairs are available for sample inputs as training data, where a supervised sentiment detection algorithm could learn from.

Recently, sentiment analysis has attracted wide attention. Aspect detection has been studied as a subtask of aspect-level sentiment analysis. The goal of aspect detection is to identify the aspect categories in sentence instead of extracting aspect terms. The aspect category set is predefined in advance. Most existing researches focus on classical machine learning models using classifiers with rich features, or deep learning models. LR and SVM are among the popular and effective classifiers. Unigram, Bigram and Lexicon features are the most important features for aspect detection. Many neural network models have been proposed for various tasks including sentiment analysis, such as Recursive Auto Encoder (RAE)[36, 37], Recurrent Neural Network (RNN) [19, 38], Convolutional Neural Network (CNN) [13, 14, 16], and more. The common solutions typically train deep learning models by using the basic models, and fine-tuning the word representations pre-trained by word2vec [20] or glove [25]. Attention-based LSTM with aspect embedding (ATAE-LSTM) [44] is shown effective to enforce the neural model to attend the related part of a sentence, with response to a specific aspect. Some variants of RNN and attention are proposed to improve the performance of aspect-level sentiment classification. However, ATAE-LSTM and its variants need aspect category as input, which limits the application. As we mentioned before, how to take advantage of the relationship between aspect detection and sentiment classification is the key.

In spite of the great success of neural network models, linguistic knowledge [30] is often required to achieve the best performance for sentiment analysis. However, linguistic knowledge is domain specific and costly to obtain. For example, the words to express positive and negative opinions in restaurant reviews will be very different from the words used in movie reviews. Further, many neural network models cannot provide explanations on how and why the predictions are made. Very recently, RNN-Capsule [45] demonstrates state-of-the-art accuracy on sentence-level sentiment classification. It does not require linguistic knowledge and is capable of outputting meaningful words with sentiment tendency. Inspired by RNN-Capsule, we propose AS-Capsules model for aspect-level sentiment analysis.

The AS-Capsules Model. The concept of "capsule" was proposed by Hinton *et al.* in 2011 [8]. A capsule is a group of neurons that "perform some quite complicated internal computations on their inputs and then encapsulate the results of these computations into a small vector of highly informative outputs" [8]. Following this high-level concept, each capsule in RNN-Capsule was designed to predict one sentiment polarity (*e.g.,* positive, negative, and neutral) [45]. In this work, we follow the same high-level concept of capsule as in RNN-Capsule, to design the AS-Capsules model for aspect-level sentiment analysis.[1]

Specifically, each individual capsule in the AS-Capsules model contains *an attribute*, *a state*, *a capsule embedding*, and *four modules*. The four modules are 'aspect representation module', 'aspect probability module', 'sentiment representation module', and 'sentiment distribution module'. For each predefined aspect, we build a capsule whose attribute is the same as the aspect category (*e.g., food* or *price*). Given a piece of text, we represent the given text by the low-level hidden vectors encoded by an encoder RNN. All capsules take the low-level hidden representations as their input, and each capsule outputs: (i) the aspect probability computed by its aspect probability module, and (ii) the sentiment distribution computed by its sentiment distribution module. All capsules utilize a shared RNN to communicate with each other to prevent capsules from attending conflict parts. The hidden representation of shared RNN is high-level because its input is the output of low-level encoder RNN. All attentions in capsule rely on the capsule embedding, which is capable of enforcing the model to focus on the correlated parts with respect to aspect. Aspect representation module and sentiment representation module share one representation generated by a component known as shared attention.

Compared with most existing neural network models for sentiment analysis, the AS-Capsules model does not heavily rely on the quality of input instance representation. The RNN layer to encode the given text input can be realized through the widely used LSTM models, GRU models or their variants. The model does not require any linguistic knowledge. Instead, each capsule is capable of outputting two kinds of attended words, one kind of words to reflect its assigned aspect category, and the other to reflect the sentiment tendency. Both sets of words are learned through the attention mechanisms. Experiments show that the words attended by each capsule well reflect the capsule's aspect category and sentiment tendency. We observe that the attended words cover a wide range of words from high frequency words to low frequency words. As low frequency words are not usually covered in sentiment lexicon, the domain-dependent aspect and sentiment words could be extremely useful sense making from the feedbacks to services or products. To summarize, the main contributions of this work are as follows:

- We propose AS-Capsules model to simultaneously perform aspect detection and aspect-level sentiment classification. A capsule is easy to build with input representations encoded by RNN. Each capsule contains an attribute, a state, a capsule embedding, and four modules known as aspect representation module, aspect probability module, sentiment representation module, and sentiment distribution module.
- The proposed AS-Capsules model does not require any linguistic knowledge to achieve state-of-the-art performance. Instead, the model is able to attend both sentiment words and aspect words reflecting the aspect knowledge of the dataset.
- We conduct experiments on a benchmark dataset (SemEval 2014 Task 4 dataset) to compare our model with strong baselines. Results show that our model is competitive and robust. We further show that our model trained on the SemEval dataset could be directly applied on Yelp reviews and output meaningful results.

2 RELATED WORK

Early approaches for sentiment analysis are mostly based on feature engineering and manually defined rules [35]. Recently, neural

[1] Our AS-Capsules model is different from the idea of Capsule Network (CapsNet) [34].

networks become the mainstream for sentiment analysis. Most current studies focus on improving the quality of vector representation of input instance using different models *e.g.,* RNN, RAE, CNN. We briefly review the related works on aspect detection, and aspect-level sentiment classification.

Aspect Detection. Aspect detection aims at identifying aspects about which users express their sentiments. A popular approach for aspect detection features a frequency-based method [10], where single nouns and compound nouns are considered possible aspects. Hence, only explicit aspects are detected. The authors in [6] then employ association rules mining to find implicit aspects. Instead of focusing on frequencies, syntax-based methods have also been used to detect aspects by means of syntactical relations [48, 49]. In general, this kind of models operates in an unsupervised manner. In [29], the authors propose a hybrid model where pointwise mutual information is used to find possible aspects, which are then fed into a Naive Bayes classifier to output a set of explicit aspects. There are also works like [11] to formulate aspect detection as a labeling problem, and a linear chain Conditional Random Field (CRF) is utilized.

Recently, relevant aspects are identified by employing word embedding techniques [32]. The method utilizes semantic and syntactic relationships in word embedding vectors in order to improve the extraction of multiple words aspects and distinguish conflict aspects. The effectiveness of word embeddings is investigated for aspect-level sentiment analysis in [1], in which both semantic and sentiment information are encoded. A semi-supervised word embedding algorithm is proposed in [50] to obtain continuous word embeddings on a large set of reviews. Then the word representations could be used to generate deeper and hybrid features to predict the aspect category.

To improve the performance of aspect detection, additional convolutional neural network features are extended in [42] besides extracting lots of features including lexicon, syntactic and cluster. Nevertheless expensive human effort and CNN operations limit its application.

Aspect-level Sentiment Analysis. Aspect-level sentiment classification deals with fine-grained classification with respect to specific aspect(s). Traditional approaches are to design a set of features manually. There are lexicon-based features built for sentiment analysis [22] with the abundance of sentiment lexicons [12, 26, 31]. Many studies focus on building sentiment classifiers with bag-of-words, sentiment lexicons, and other features, using SVM [23] or other classifiers. However, the results highly depend on the quality of features, and feature engineering is expensive.

Recently, many neural network models have been developed to tackle sentiment analysis. Transferring knowledge from existing public datasets [7] or pre-annotated information [39] improves the performance of aspect-term level sentiment classification. There are methods using memory network or linguistic knowledge to improve the performance of aspect-level sentiment classification. An innovative model named CEA is proposed in [46] using context memory, entity memory, and aspect memory.

Attention mechanism has shown to be effective in many applications including machine translation [2, 43], sentiment analysis [40, 44], summarization [33], and more. Given a sentence and the corresponding aspect(s), Attention-based LSTM with Aspect Embedding (ATAE-LSTM) [44] is able to predict the sentiment polarity at the aspect level. ATAE-LSTM is the first model to predict different sentiment tendencies with respect to different aspects in the same text. Aspect embedding and aspect based attention mechanism are designed in ATAE-LSTM to utilize aspect information effectively. The attention mechanism is well-designed to attend the different parts of a sentence when considering different aspects. To improve the performance of attention mechanism, [41] proposes a method named Aspect Fusion LSTM for incorporating aspect information for learning attentions. Motivated by similar reason, [18] proposes content attention with two enhancing attention mechanism. Multi-Head attention [43] jointly attends to information from different representation subspaces at different positions. In the proposed AS-Capsules model, we design attentions for every representation module in each capsule, which benefit from both low-level representation and high-level representation through different levels of encoder. In addition, a shared attention is able to generating shared representation as a part of both aspect and sentiment representations.

3 ASPECT-LEVEL SENTIMENT CAPSULES

The aspect-level sentiment capsules (AS-Capsules) model has its root in RNN-Capsule. Next we briefly describe RNN-Capsule, then detail the design of AS-Capsules model and its optimization method.

3.1 Preliminary: RNN-Capsule

Recurrent Neural Network. As the name suggests, RNN-Capsule is based on RNN. A recurrent neural network (RNN) is able to exhibit dynamic temporal behavior for a time sequence through connections between units. A unit can be realized by an LSTM model, a GRU model, or their variants. RNNs can be bi-directional, by using a finite sequence to predict or label each element in the sequence based on the element's past and future contexts. This is achieved by concatenating the outputs of two RNNs, one processes the sequence from left to right, and the other from right to left.

Briefly speaking, in an RNN realized by LSTM, the hidden states h_t and memory cell c_t in LSTM is a function of the previous h_{t-1} and c_{t-1}, and input vector x_t, or formally as follows:

$$c_t, h_t = \text{LSTM}(c_{t-1}, h_{t-1}, x_t) \tag{1}$$

The hidden state h_t denotes the representation of position t while encoding the preceding contexts of the position. More details about LSTM are given in [9].

RNN-Capsule. RNN-Capsule is designed to predict the sentiment category (*e.g.,* positive, negative, and neutral) of a given piece of text. The input text is encoded by RNN and the hidden vector representations are input to all capsules. One capsule is built for one sentiment category and each capsule contains an attribute, a state, and three modules. The attribute of a capsule reflects its dedicated sentiment category (*e.g.,* positive). The three modules are: (i) representation module for building capsule representation using attention mechanism, (ii) probability module for predicting the capsule's state probability based on its representation, and (iii) reconstruction module for rebuilding the representation of the input

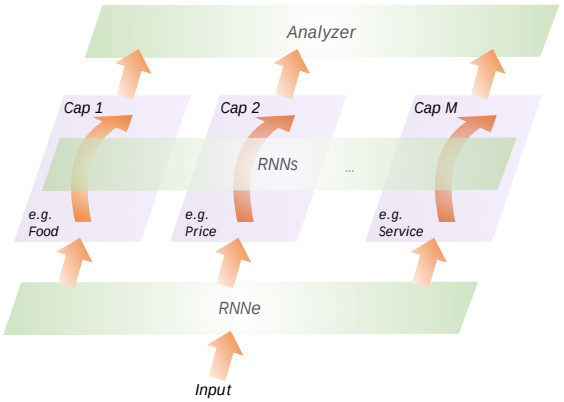


Figure 1: Architecture of AS-Capsules Model. Number of capsules equals the number of aspect categories. The hidden vectors H_1 are encoded by RNN_e, which encodes the input text. All capsules take H_1 as input and each capsule outputs its aspect probability p and sentiment distribution $\mathcal{P}$.

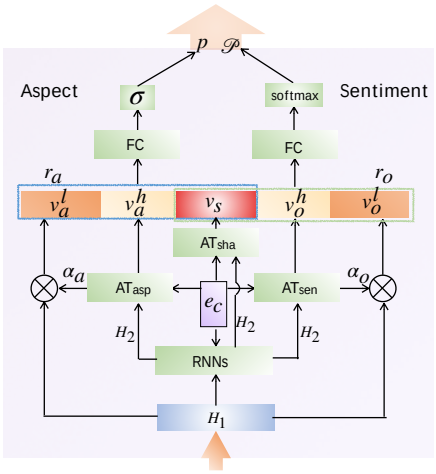


Figure 2: The architecture of a single capsule. The input to a capsule is the hidden vectors H_1 from RNN_e. e_c is the capsule embedding. The output is aspect probability p and sentiment distribution $\mathcal{P}$ of this capsule.

instance. A capsule's state is 'active' if the output of its probability module is the largest among all capsules, and 'inactive' otherwise.

There are two learning objectives in RNN-Capsule network. The first is to maximize the state probability of the capsule corresponding to the groundtruth sentiment, and to minimize the state probabilities of other capsule(s). The second is to minimize the distance between the input representation and the reconstruction representation of the capsule corresponding to the ground truth, and to maximize such distances for other capsule(s).

RNN-Capsule is not designed for aspect-level sentiment analysis and each capsule in RNN-Capsule corresponds to one sentiment category. The sentiment category predicted by RNN-Capsule therefore does not reflect the sentiment on any particular aspect. Aspect-level sentiment classification relies on aspect category heavily, so well-designed shared components for the two subtasks will benefit a lot. The high-level shared RNN between capsules is important, it allows capsules to cooperate to prevent capsules from attending conflict parts.

3.2 The AS-Capsules Model

The architecture of the proposed AS-Capsules model is depicted in Figure 1. The number of the capsules M equals to the number of predefined aspect categories. For example, we need 5 capsules to model a set of 5 categories {*Food*, *Price*, *Service*, *Ambience*, *Anecdote/miscellaneous* }, and a capsule is built for each aspect category. As in RNN-Capsule [45], we use an encoder RNN named RNN_e to encode the input text.

Given a piece of text (*e.g.*, a sentence, a paragraph, a document), RNN_e encodes the given instance and outputs the hidden representations H_1. Briefly speaking, in the encoder RNN, the hidden matrix H_1 are function of RNN_e and input word representations W, or formally:

$$H_1 = \text{RNN}_e(W) \tag{2}$$

The word representations $W = [w_1, w_2, \ldots, w_N]$ are obtained from glove, and N is the length of input text. The base unit of RNN is an LSTM, GRU, or their variants, *e.g.*, bi-directional LSTM.

All the capsules take the same hidden vectors H_1 as their input, and they share an RNN named RNN_s. Obviously, different aspect ought to attend different parts of input text, which is not well-designed in RNN-Capsule [45]. RNN_s allows capsules to communicate with each other, which is capable of preventing capsules from attending conflict parts. The high-level hidden representations are the hidden vectors of RNN_s. High-level here means that the input of RNN_s is the hidden representation of RNN_e. There are special designed attentions in capsule, and we will explain them in Section 3.3. Each capsule outputs an aspect probability and a sentiment distribution, through its aspect probability module and sentiment distribution module, respectively. Analyzer refers to the training objective strategy, which is capable of utilizing the correlation between aspect detection and aspect-level sentiment classification to improve the performance.

3.3 Structure of A Single Capsule

The structure of a single capsule is shown in Figure 2. A capsule contains *an attribute*, *a state*, *a capsule embedding*, and *four modules* (aspect representation module, aspect probability module, sentiment representation module, and sentiment distribution module).

- The attribute of a capsule reflects its dedicated aspect category, which is pre-assigned when we build the capsule. Depending on the number of aspect categories in a given problem, the same number of capsules are built to reflect each aspect category.
- The state of a capsule, *i.e.,* 'active' or 'inactive', is determined by aspect probability. To learn the model, a capsule's state is active if the current aspect appears in input text. Then AS-Capsules maximizes the aspect probability of the current active capsule. In testing, a capsule's state will be active if

its aspect probability p is above a predefined threshold *e.g.*, 0.5. Note that, a given piece of text may contain opinions on multiple aspects, then multiple capsules will be active.

- Capsule embedding e_c is a vector representation of the current capsule learned during training as in [44]. Recall that each capsule is assigned to learn one particle embedding and different aspects often demonstrate different word features for expressing the aspect and sentiment tendency. For example, "the pizza is delicious" demonstrates *food* aspect with positive sentiment. Here, the words "pizza" and "delicious" will not be applicable for other aspects like *service* or *price*. Therefore, for each capsule, we learn its capsule embedding.
- Aspect representation module learns the aspect representation r_a including three parts $[v_a^l, v_a^h, v_s]$, as shown in Figure 2. Given the hidden representations H_1 as input, we compute the high-level representation H_2 through RNN_s. The first part v_a^h of r_a is the weighted representation using the aspect-based attention AT_{asp} with capsule embedding e_c and H_2 as input. Utilizing the attention scores α_a of AT_{asp} and H_2, we can get the second representation v_a^l for aspect. The shared representation v_s, which is the output of the shared attention AT_{sha}, is the last part of r_a. AT_{sha} is shared with sentiment representation module. The aspect probability module then predicts the capsule's aspect probability p based on r_a. Similarly, sentiment representation module computes sentiment representation r_o of this capsule through similar method. Based on r_o, the sentiment distribution module generates sentiment distribution $\mathcal{P}$ on the predefined sentiment categories *e.g.*, positive, negative, neutral.

The essence of the AS-Capsules model is the four modules briefed above. Next, we detail the four modules.

Aspect Representation Module. Given the hidden vectors H_1 encoded by RNN_e, we are able to compute high-level hidden representations H_2 using RNN_s, which is shared by all capsules. Then, we use capsule embedding e_c and two attention mechanisms known as aspect-based attention AT_{asp} and shared attention AT_{sha}, to construct aspect representation r_a inside each capsule. The aspect-based attention attends the indicative words based on the aspect detection task. The shared attention benefits both aspect detection and sentiment classification. Our formulation is inspired by [2, 4, 44, 47]. Specifically, given H_1 as the low-level representations of the input text, we get a high-level hidden representations through

$$H_2 = \text{RNN}_s(H_1, e_c), \tag{3}$$

where, RNN_s is the shared encoder. The input of RNN_s is the concatenation of H_1 and e_c to enforce the RNN to attend the aspect correlated parts, inspired by AE-LSTM [44]. After getting H_2, we compute the high-level aspect representation v_a^h and attention weights α_a through aspect-based attention AT_{asp}, or formally:

$$v_a^h, \alpha_a = \text{AT}_{\text{asp}}(H_2, e_c) \tag{4}$$

Aspect-based attention, sentiment-based attention and shared attention have the same structure, so here we only detail the attention mechanism in AT_{asp}:

$$M_a = \tanh(W_h H_2 + W_c e_c \otimes N) \tag{5}$$

$$\alpha_a = \text{softmax}(w_a^T M_a) \tag{6}$$

$$v_a^h = H_2 \alpha_a^T \tag{7}$$

Here, $e_c \otimes N$ is the operator that repeatedly concatenates e_c for N times. W_h, W_c and w_a are the parameters of the current capsule for the aspect-based attention layer. The attention importance score α_a is obtained by multiplying the representations with the weight matrix, and then normalizing to a probability distribution over the words. Lastly, the high-level aspect representation vector v_a^h is a weighted summation over all the positions using the attention importance scores as weights.

The shared attention AT_{sha} is computed in a similar manner as the aspect-based attention.

$$v_s = \text{AT}_{\text{sha}}(H_2, e_c) \tag{8}$$

To obtain the original, or low-level aspect representation v_a^l from the input of capsule, we use the attention weights from aspect-based attention to weight the low-level input H_1.

$$v_a^l = H_1 \alpha_a^T \tag{9}$$

Lastly, we get the aspect representation by concatenating v_a^h, v_a^l and v_s,

$$r_a = [v_a^h, v_a^l, v_s], \tag{10}$$

where square brackets refer to the concatenation of vectors.

Aspect Probability Module. The aspect probability p is computed by a sigmoid function after getting the aspect representation r_a.

$$p = \sigma(W_p r_a + b_p), \tag{11}$$

where W_p and b_p are the parameters for the aspect probability module of the current capsule.

Sentiment Representation Module. The sentiment representation module is computed in a similar manner as the aspect representation module. The sentiment representation r_o contains three parts, known as high-level sentiment representation v_o^h, low-level sentiment representation v_o^l and shared representation v_s. The high-level sentiment representation v_o^h is computed in sentiment-based attention:

$$v_o^h, \alpha_o = \text{AT}_{\text{sen}}(H_2, e_c) \tag{12}$$

After getting the attention importance scores α_o in sentiment representation module, we utilize the low-level information through

$$v_o^l = H_1 \alpha_o^T \tag{13}$$

In the two formulas above, AT_{sen} is the sentiment-based attention. e_c is the capsule embedding. H_1 and H_2 are the hidden representations of RNN_e and RNN_s. The attention importance score for each position for sentiment is α_o. The sentiment representation vector r_o is obtained by concatenating v_o^h, v_o^l and v_s.

Note that, both the aspect representation and the sentiment representation obtained from attention layer are high-level encodings of the entire input text. The attention mechanism designed in the model is for improving the model's capability and robustness.

Sentiment Distribution Module. The sentiment distribution $\mathcal{P}$ is computed by a softmax function after getting the sentiment representation vector r_o.

$$\mathcal{P} = \text{softmax}(W'_p r_o + b'_p), \tag{14}$$

where W'_p and b'_p are the parameters for the sentiment distribution module of the current capsule.

The above four modules complement each other in the AS-Capsules model. From a macro perspective *i.e.,* the full dataset, the words attended by different capsules match the capsules' attribute. From micro perspective *i.e.,* a piece of input text, the state 'active' or 'inactive' of a capsule is determined by its aspect probability p. The sentiment distribution is with respect to the aspect category of the current capsule.

There are two different ways to utilize the correlation between aspect and sentiment. One way is to draw support from the capsule structure. In our first attempt, we design the capsule structure with two independent attentions based on their own embeddings named aspect embedding and sentiment embedding. However, the two independent embeddings have the same effect, and cannot benefit from the correlation between aspect and sentiment effectively. Motivated by shared embedding [44] and multi-head attention [43], in our current design, aspect representation module and sentiment representation module share one embedding known as capsule embedding. Specially, we design aspect-based attention, sentiment-based attention and shared attention. Aspect-based attention and sentiment-based attention are shared by low-level and high-level RNN encoders to get hierarchical representations of input text. Shared attention is used to generate shared representation for both aspect detection and sentiment classification. The other way to utilize the correlation considers the co-occurrence of aspect and sentiment pair. We use the idea of mask. That is, we only keep the cross entropy of aspect(s) that appear in the input text and ignore the irrelevant aspect(s) in the learning objective.

3.4 Training Objective

The training of the proposed AS-Capsules model has two objectives. One is to maximize the aspect probability of active capsule(s) matching the ground truth and minimize aspect probability of the inactive capsule(s). The other is to minimize the cross-entropy of sentiment distribution of active capsules.

Aspect Probability Objective. A given text may express sentiments on multiple aspects. Hence, we have both positive sample(s) (*i.e.,* the active capsule(s)) and negative sample(s) (*i.e.,* the inactive capsule(s)). Recall that our objective is to maximize the aspect probability of active capsules and to minimize the aspect probability of inactive capsules. The classification objective J can be formulated by cross entropy loss:

$$J(\theta) = \sum \frac{1}{M} \sum_{i=1}^{M} \text{cross-entropy}(y_a^i, [p_i, 1 - p_i]), \tag{15}$$

where p_i is the aspect probability of the capsule i. For a given training instance, $y_a^i = 1$ for an active capsule (*i.e.,* the corresponding aspect occurs in the given text), and $y_a^i = 0$ for an inactive capsule. M is the number of aspect categories, where is 5.

Table 1: The statistics of restaurant reviews on SemEval 2014 Task 4 dataset. The sentiment category 'conflict' is not used.

Aspect	Positive		Negative		Neural	
	Train	Test	Train	Test	Train	Test
Food	867	302	209	69	90	31
Price	179	51	115	28	10	1
Service	324	101	218	63	20	3
Ambience	263	76	98	21	23	8
Anecdote	546	127	199	41	357	51
Total	2179	657	839	222	500	94

Aspect-level Sentiment Classification Objective. The other objective is to ensure the accuracy of aspect-level sentiment classification of the active capsule(s). Similarly, the unregularized objective U can be formulated as cross entropy loss:

$$U(\theta) = \sum \frac{1}{\sum_{\forall y_a^i=1} 1} \sum_{\forall y_a^i=1} \text{cross-entropy}(y_o^i, \mathcal{P}_i) \tag{16}$$

We only utilize the cross-entropy loss of the 'active' capsule(s). y_o^i is the groundtruth sentiment of the 'active' capsule i. $\mathcal{P}_i$ is the predicted sentiment distribution of the capsule i.

Considering both objectives, our final objective function L is obtained by adding J and U:

$$L(\theta) = J(\theta) + U(\theta) \tag{17}$$

4 EXPERIMENT

We now evaluate the proposed AS-Capsules for aspect detection and aspect-level sentiment classification, against baselines.

4.1 Dataset and Model Implementation Details

Dataset. We conduct experiments on SemEval 2014 Task 4 dataset [27].[2] The dataset consists of customer reviews for laptops and restaurants, but only restaurant reviews are annotated with aspect-specific polarity. Hence we conduct experiments on restaurant reviews.

The restaurant reviews consist of about 3K English sentences from [5]. We randomly cut out $\frac{1}{8}$ as validation dataset, and the rest is the training dataset. Additional restaurant reviews, not in the original dataset of [5], are used as test data. The set of aspects $\mathcal{A}$ is {*food, service, price, ambience, anecdote*}. There are four sentiment categories: *positive*, *neutral*, *negative* and *conflict*. Each sentence is assigned one or more aspects together with a polarity label for each aspect, *e.g.,* "Staffs are not that friendly, but the taste covers all." would be assigned the aspect-sentiment pairs {⟨food, positive⟩, ⟨service, negative⟩}. In our experiments, we use the first three sentiment categories, as in most other studies.

Table 1 presents the statistics of the SemEval 2014 Task 4 dataset. The two aspects *food* and *anecdote* have the largest number of instances. However, it is hard to think about words that are indicative for aspect *anecdote*.

Implementation Details. In our experiments, all word vectors are initialized by Glove [25].[3] The pre-trained word embeddings and

[2]Refer to http://alt.qcri.org/semeval2014/task4/ for more details.
[3]http://nlp.stanford.edu/projects/glove/

Table 2: Micro F_1 of methods on SemEval 2014 Task 4 dataset. Best results are in bold face and second best underlined.

Model	micro-F_1	Model	micro-F_1
KNN	63.9	HLBL	69.7
LR	66.0	C&W	72.5
SVM	80.8	word2vec	83.3
SemEval-Average	73.8	Hybrid-WRL-300	88.6
NRC-Lexicon	84.1	Hybrid-WRL-Best	**90.1**
NRC	88.6	AS-Capsules	<u>89.6</u>

Table 3: The average F_1 and F_1 of different aspects for Subtask 1: aspect detection.

Model	Average	Food	Price	Service	Ambience	Anecdote
Bi-LSTM	83.0	92.6	79.2	88.8	74.8	<u>79.6</u>
AE-LSTM	84.5	**93.8**	83.3	88.8	78.2	78.5
AT-LSTM	84.5	91.6	83.0	85.6	<u>83.0</u>	79.2
RNN-Capsule	<u>85.5</u>	**93.8**	<u>85.4</u>	<u>89.4</u>	80.6	78.3
AS-Capsules	**87.2**	<u>93.4</u>	**85.9**	**91.0**	**83.3**	**82.4**

capsule embedding have dimensions of 300 and 256, respectively. The dimension of hidden vectors encoded by RNN is 256 (hence 512 if RNN is bidirectional). We use a single layer bidirectional LSTM in AS-Capsules. The model is trained with a batch size of 16 examples, and there is a checkpoint every 8 mini-batch due to the small size of dataset. Dropout is 0.5 for word embedding and linear layer in aspect probability module and sentiment distribution module.

We implement our model on Pytorch (version 0.4).[4] Model parameters are randomly initialized. Adam [15] is utilized to optimize our model, and we use $1e-3$ as learning rate for model parameters except word vectors, and $1e-4$ for word vectors.

4.2 Evaluation on Three Subtasks

SemEval 2014 task 4 dataset is widely used for aspect-level sentiment analysis. Because of the detailed annotation, multiple evaluations can be conducted. We report our experiments on three subtasks: aspect detection, sentiment classification on given aspects, and aspect-level sentiment analysis.

4.2.1 Subtask 1: Aspect Detection. Given a piece of text, the task of aspect detection is to predict the existence of predefined aspects, which is a typical multi-label classification task. We compare AS-Capsules with state-of-the-art baselines designed for aspect detection. Specially, we compare detailed F_1 of all categories with effective joint deep learning baselines including Bi-LSTM, AE-LSTM, AT-LSTM, and RNN-Capsule.

KNN is the baseline provided by SemEval official [28]. First, the Dice coefficient is used to calculate the similarity to find k most similar sentences in training dataset for given test sentence. Then, m most frequent aspect categories of the k retrieved sentences will be assigned to the test sentence. m is the number of most frequent aspect categories per sentence among the k sentences. Logistic Regression (LR) and Support Vector machine (SVM) are used as classifiers using unigram and bigram features. SemEval-Average is the average result of all the systems in SemEval 2014. NRC, the best system in SemEval 2014, adopts SVM as the classifier with some well-designed features including n-grams, stemmed n-grams, character n-grams, non-contiguous n-grams, word cluster n-grams and lexicons. NRC-Lexicon is the result without the lexicon feature.

Some word representation methods are compared with our AS-Capsules. C&W [3] and word2vec [20] are the powerful and accepted generally methods for word representation learning. HLBL [21] is a hierarchical model, which performs well using a carefully constructed hierarchy over words. Hybrid-WRL [50] is a word representation learning method using hybrid features including shared-features and aspect-specific features.

Table 2 lists the micro F_1 of aspect detection on restaurants reviews. Our proposed AS-Capsules model achieves the second best. Notice that, there are two kinds of Hybrid-WRL methods. The difference between them is the size of word representation. Hybrid-WRL-300 uses 300 as word representation size, 600 for Hybrid-WRL-Best. Our AS-Capsules adopts 300 as the dimension of word representation, improves 1 percentage than Hybrid-WRL at the same size of word representation. Obviously, there are more complexity and parameters with bigger size of word representation. For classical machine learning methods, SVM performs better than KNN and LR. NRC achieves the best SemEval result with the textual features and lexicons features using SVM as classifier. Without Lexicons features, NRC-Lexicon is 4 percentage lower than NRC. Word representation methods perform better than KNN, LR and SVM, however they perform worse than NRC with well-designed features.

Although classical machine learning methods have shown their effectiveness, it is better to compare with neural network methods. Bidirectional-LSTM (Bi-LSTM) is a variant of LSTM which is introduced in Section 3.1. It is capable of utilizing the content information through the bidirectional structure. Aspect Embedding LSTM (AE-LSTM) is proposed in [44], where aspect embeddings are concatenated with word vectors. AE-LSTM considers aspect information so it is expected to perform better than Bi-LSTM. Different parts of input sentence have different importance, so attention is a powerful way to address this problem. Attention based LSTM (AT-LSTM) [44] further uses attention mechanism, which performs better than other baselines. RNN-Capsule uses each capsule to detect one category, sentiment category in its original paper [45], and aspect in this experiment.

Reported in Table 3, our proposed AS-Capsules is the best performing method, followed by RNN-Capsule. Specifically, AS-Capsules achieves the best average F_1, best results on four aspects except aspect *food*. The gap between AS-Capsules and the best model in aspect *food* is very small. AE-LSTM and AT-LSTM deliver similar results and Bi-LSTM performs the poorest.

4.2.2 Subtask 2: Sentiment Classification on Given Aspects. Given a piece of text and also the annotated aspect, the task is to predict the sentiment expressed in the text on the given aspect. For this subtask, we compare AS-Capsules with four baselines: Bi-LSTM,

[4] https://github.com/pytorch

Table 4: Accuracy and F_1 of sentiment categories for Subtask 2: sentiment classification on given aspects.

Model	Accuracy	F_1-Positive	F_1-Neutral	F_1-Negative
Bi-LSTM	82.1	89.2	49.7	73.2
AE-LSTM	82.8	89.8	54.2	72.6
AT-LSTM	83.0	89.8	47.3	75.8
ATAE-LSTM	84.3	90.1	**61.9**	77.5
AS-Capsules	**85.0**	**91.2**	50.7	**78.7**

Table 5: Accuracy and F_1 of different sentiment categories for Subtask 3: aspect-level sentiment analysis.

Model	Accuracy	F_1-Positive	F_1-Neutral	F_1-Negative
Bi-LSTM	62.3	80.2	43.6	56.6
AE-LSTM	64.7	82.4	50.3	55.7
AT-LSTM	65.6	82.4	46.6	57.7
AS-Capsules	**68.1**	**83.3**	**53.6**	**61.6**

AE-LSTM, AT-LSTM, and Attention-based LSTM with Aspect Embedding (ATAE-LSTM) [44]. ATAE-LSTM unitizes aspect information to attend the important words in sentence with respect to a specific aspect. Note that RNN-Capsule cannot be applied to this subtask because RNN-Capsule is self-attentive, and it cannot take aspect as an additional input.

From the results reported in Table 4, we observe that AS-Capsules achieves the best accuracy of 85.0. It outperforms all baselines with respect to the F_1 results on the *positive* and *negative* sentiment categories. Due to less neutral data in dataset, AS-Capsules performs not good enough. Among the baseline methods, ATAE-LSTM outperforms the rest. AE-LSTM and AT-LSTM perform better than Bi-LSTM. Among three sentiment categories, *positive* is much easier to predict and *neutral* is the most difficult category due to the smallest size of data.

4.2.3 Subtask 3: Aspect-level Sentiment Analysis. Given a piece of text, subtask 3 requires a method to detect $\langle aspect, sentiment \rangle$ pair(s) from the input text. A detected pair is considered correct if both components in the pair are correctly identified. We compare AS-Capsules with Bi-LSTM, AE-LSTM and AT-LSTM. Again, RNN-Capsule is not designed to classify sentiment with given aspect(s) and ATAE-LSTM needs aspect as additional input; hence these two models are not applicable to this task.

Reported in Table 5, AS-Capsules model delivers the best results on accuracy. It also outperforms all baselines with respects to the F_1 results on the three sentiment categories. AE-LSTM and AT-LSTM perform similarly and both outperform Bi-LSTM. Similar to earlier observations, sentiment *positive* is relatively easier to detect and *neutral* is the hardest category.

5 EXPLAINABILITY ANALYSIS

We have shown that AS-Capsules model outperforms all baseline models for aspect-level sentiment analysis. Because of the attention mechanism, AS-Capsules model is able to attend meaningful words in the aspect representation module and sentiment representation module. Specifically, each word is assigned an attention weight in aspect representation module and sentiment representation module by multiplying *aspect probability* with *aspect attention weight* and *sentiment attention weight*, respectively.

Table 6: Word attended by sentiment representation module in Capsule *food*. Significant words and low frequency words are ranked by *average attention weight* $\times \log$*(word frequency)* and *average attention weight*, respectively.

No.	Significant words	Freq	Low frequency words	Freq
1	delicious	22	divine	1
2	great	67	satisfying	2
3	tasty	8	favourites	1
4	good	55	scrumptious	2
5	best	24	terrific	1
6	fresh	23	yummy	5
7	yummy	5	greatest	1
8	excellent	23	frosty	1
9	amazing	7	delicious	22
10	outstanding	5	fave	1
11	fantastic	5	luscious	1
12	wonderful	8	tasty	8
13	mouth	5	unexpected	1
14	superb	3	winner	1
15	delectable	3	refreshing	2
16	recommend	8	recomend	1
17	satisfying	2	flavor	2
18	scrumptious	2	highlight	1
19	perfect	6	lemons	1
20	sweet	5	delicate	1

As a case study, we show the words attended by AS-Capsules during test. That is, after all test instances are evaluated, we obtain two lists of attended words from each capsule with their attention weights for aspect and sentiment respectively. Due to page limit, we can only display a small number of words with some ranking criteria. A straightforward ranking is by the *averaged attention weight* of a word. By this ranking, most top-ranked words are of low frequency. That is, some words have significant attention weight (or strong aspect or sentiment tendencies) but do not appear very often. Another way of ranking is by the product of *averaged attention weight* and *the logarithm of word frequency*.

Words Attended for Aspect Detection. Table 7a lists the top-ranked 20 words by the product of average attention weight and logarithm of word frequency, for the five aspects. Table 7b lists the top-ranked 20 words by average attention weight.

From the two tables, we observe that almost attended words are self-explanatory for the assigned aspect category. For instance, the attended words by capsule *food* are mostly food categories or ingredients. Capsule *service* attends words for personnel involved and their attitude. Without the need of any linguistic knowledge, the AS-Capsules model is able to identify words that reflect the aspect categories. This provides an easy way to build domain specific lexicons on domain specific data.

Words Attended for Aspect-level Sentiment Classification. We show the sentiment words attended by AS-Capsules for capsule *food* with *positive* sentiment tendency. Other aspects are omitted

Table 7: Words attended by aspect representation module in all capsules on SemEval dataset.

(a) Top ranked words by average attention weight $\times$ log(word frequency).

No.	Food	Freq	Price	Freq	Service	Freq	Ambience	Freq	Anecdote	Freq
1	food	150	prices	18	service	76	atmosphere	21	!	23
2	sushi	22	cheap	10	staff	28	decor	5	meal	3
3	pizza	14	price	11	waiter	10	ambiance	5	.	172
4	meal	12	priced	7	waiters	7	space	4	restaurant	20
5	menu	23	value	5	bartender	6	music	5	sushi	3
6	desserts	8	inexpensive	3	attentive	8	cozy	6	food	7
7	portions	7	expensive	4	waitress	4	ambience	3	menu	6
8	pasta	5	bill	4	owner	5	interior	2	experience	11
9	wine	14	overpriced	2	servers	3	room	4	place	28
10	sauce	11	money	3	hostess	3	intimate	4	italian	4
11	shrimp	7	affordable	2	friendly	22	quiet	3	in	43
12	soup	9	pay	3	courteous	3	clean	2	the	98
13	dessert	5	over	3	greeted	3	chic	2	dining	4
14	dishes	9	reasonable	5	politely	2	crowded	2	at	26
15	cheese	10	the	81	accomodating	2	scene	2	money	3
16	seafood	6	for	23	rude	9	downstairs	2	pizza	2
17	chicken	17	your	4	prompt	5	,	78	for	36
18	cuisine	4	reasonably	4	bartenders	2	place	21	night	7
19	crab	5	worth	4	manager	4	laid-back	4	here	17
20	crust	4	at	14	asked	3	like	8	this	53

(b) Words ranked by average attention weight.

No.	Food	Freq	Price	Freq	Service	Freq	Ambience	Freq	Anecdote	Freq
1	truffles	1	overpriced	2	service	76	claustrophobic	1	thai	1
2	hotdogs	1	pricey	1	politely	2	chairs	1	cost	1
3	meat	1	inexpensive	3	accomodating	2	bathroom	1	meal	3
4	meats	1	cheap	10	lady	1	patio	1	agave	1
5	wines	2	deal	1	servers	3	decor	5	service	1
6	sangria	2	prices	18	hostess	3	chill	1	desserts	1
7	pancakes	1	price	11	solicitous	1	decoration	1	delicacy	1
8	mojitos	1	priced	7	brusquely	1	landscaping	1	sum	1
9	codfish	1	value	5	bartenders	2	pretentious	1	pizza	2
10	crepes	1	wallet	1	courteous	3	singer	1	tacos	1
11	pizzas	1	expensive	4	port	1	setting	1	sushi	3
12	breads	1	cost	1	waitress	4	lighting	1	martinis	1
13	guacamole	1	bill	4	greeted	3	atmosphere	21	delivary	1
14	tequila	1	spend	1	polite	2	space	4	crepes	1
15	cookies	1	14	1	staff	28	bumping	1	steak	1
16	pudding	1	cheaper	1	awful	1	rooftop	1	astoria	1
17	marscapone	1	investment	1	waiters	7	uncomfortably	1	taste	1
18	meatball	1	50	1	gracious	1	air	1	brunch	2
19	bbq	2	affordable	2	bartender	6	ambiance	5	diamond	1
20	chili	2	6.25	1	owner	5	interior	2	stock	1

due to limited space. Similarly, Table 6 lists the top-ranked 20 significant words and low frequency words ranked by *average attention weights* $\times$ *logarithm of word frequency* and *average attention weight*, respectively. These words are consistent with sentiment lexicons identified in related studies [12, 26, 31]. All of those words reflect the positive sentiment tendency significantly. The attention weights of all significant words are above 0.35. More importantly, the words are used commonly. For low frequency words, the attention weights are over 0.77. Even though the words are used not so often, they belong to small but beautiful word group. 'delicious' exists in both columns due to a very positive sentiment tendency and common use. The *neutral* sentiment attends lots of punctuation marks and meaningless words *e.g.*,, 'and', 'a' and 'is', so they are not shown. *Negative* sentiment attends significant words including 'terrible', 'worst' and 'disappointed' whose attention weights are over 0.1.

5.1 Applying AS-Capsules on Yelp Reviews

To the best of our knowledge, the SemEval 2014 Task 4 dataset is the only dataset that comes with aspect-level sentiment annotations. This limits the evaluation of our model on aspect-level sentiment

Table 8: Words attended by aspect representation module on Yelp dataset.

No.	Ranked by average attention weight × log(word frequency)					Ranked by average attention weight				
	Food	Price	Service	Ambience	Anecdote	Food	Price	Service	Ambience	Anecdote
1	food	prices	service	atmosphere	!	chilaquiles	pricy	managers	jazz	gelato
2	beers	cheap	staff	decor	craving	porridge	affordable	station	scene	pies
3	sauce	price	friendly	patio	again	terrine	overpriced	staffs	smoke	greeted
4	beer	overpriced	polite	cramped	breakfast	fillings	overcharged	unattentive	pretension	catering
5	menu	priced	server	ambiance	sushi	oatmeal	prices	waitstaff	cramped	panini
6	foods	expensive	waitress	music	eaten	miso	inexpensive	politely	noisy	intriguing
7	pizza	pricey	employees	vibe	pasta	sauces	cheap	greeting	venues	eww
8	meal	deal	courteous	crowded	burger	milkshakes	priced	apologetic	cozy	ahhh...
9	cheese	cost	greeted	bathroom	buying	wintermelon	expensive	answering	rowdy	cakes
10	meat	buy	attentive	rooms	dinner	concepts	pricey	polite	atmosphere	sandwiches
11	chicken	cheaper	bartender	seating	eat	beers	price	courteous	decor	burgers
12	sushi	dollar	servers	cozy	grocery	unagi	cheaper	marketing	nondescript	treated
13	beef	pay	manager	walls	menu	shakes	deal	service	crowded	soeur
14	dessert	affordable	waiter	relaxing	sandwiches	cheeses	45	greet	pretentious	tidy
15	burger	bucks	desk	floor	service	foods	costs	handled	pop	salty
16	sauces	million	patient	noisy	dining	appetite	cost	staff	interior	pasta
17	pork	pricing	waitstaff	space	lunch	concoctions	600	servers	jukebox	donut
18	ingredients	buck	apologetic	loud	heaven	maya	pricing	manager	claustrophobic	digress
19	burgers	charged	answering	air	brunch	tartare	50	employees	ornate	nooo...
20	lemonade	inexpensive	prompt	interior	steakhouse	pig	25	greeted	mellow	bagel

Table 9: Word attended by sentiment representation module in Capsule *food* on Yelp dataset.

No.	Significant words	Freq	Low frequency words	Freq
1	delicious	108	dynamic	1
2	tasty	51	delicious	108
3	yummy	31	yummy	31
4	excellent	28	adventurous	3
5	fresh	51	friendliest	1
6	fantastic	37	satisfying	6
7	great	203	superb	2
8	good	390	tasty	51
9	amazing	71	meaty	1
10	flavorful	13	palates	1
11	flavor	79	heavenly	6
12	satisfying	6	scrumptious	4
13	flavour	13	chai	1
14	incredible	14	efficient	1
15	heavenly	6	loves	3
16	awesome	34	delectable	1
17	enjoy	44	excellent	28
18	flavors	20	palate	3
19	enjoying	9	flavorful	13
20	goodness	9	bite	6

analysis. On the other hand, many restaurant reviews are available from other domains without aspect-level manual annotation. The Yelp dataset[5] is an example. As most reviews on Yelp are for restaurants, we can directly apply the trained AS-Capsules to conduct a qualitative evaluation. Because there are no aspect-level annotations on Yelp dataset, we are unable to report quantitative measures like accuracy or F_1. However, it is interesting to observe whether the AS-Capsules model learned on SemEval 2014 Task 4 dataset can be used to identify meaningful words on Yelp dataset, to reflects its aspects and sentiment categories.

In this case study, we take the first 1,000 reviews from Yelp dataset. As Yelp reviews are relatively long and each review has several paragraphs so we split the reviews by paragraphs and consider each paragraph a test input to AS-Capsules model. As a result, we have 3,776 test instances from Yelp dataset.

Words Attended for Aspect Detection on Yelp. Table 8 lists the top ranked words identified by AS-Capsules for the five aspects. We provide two rankings following our earlier ranking criteria. Observe that regardless word frequencies, most of them well reflect the corresponding aspect category. For example, 'beers', 'sauce' and 'pizza' are identified as important words for *food*. Interestingly, most low frequency words are meaningful to the related aspects. For example, 'porridge', 'terrine' and 'fillings' are attended by capsule *food*. More interestingly, some numbers *e.g.,* 45, 50 and 20, are attended in capsule *price*. The phenomenon reflects that our AS-Capsules model performs well for low frequency words.

Words Attended for Aspect-level Sentiment Classification on Yelp. We now present the words attended by sentiment representation module. Table 9 lists the top ranked significant words and low frequency words identified by capsule *food* for *positive* sentiments. Most of the words attended by *neutral* are not very meaningful, so they are not shown. Most of the attended words fit their sentiment tendency in sentiment lexicons. Specially, we observe that most of them are able to reflect *food* aspect. As we mentioned before, 'delicious' is a very common word for describing food, it is also shown in low frequency words because we rank the attended words by their attention weight.

[5]https://www.yelp.com/dataset/challenge

6 CONCLUSION

In this paper, we study aspect-level sentiment analysis and propose aspect-level sentiment capsules (AS-Capsules) model. The key idea of AS-Capsules model is to use capsule structure to focus on each aspect category. Each capsule outputs its aspect probability and sentiment distribution on the targeted aspect. The objective of learning is to maximize the aspect probability of the capsule(s) matching the groundtruth and to minimize its (their) sentiment cross entropy loss. Through shared components including capsule embedding, shared encoders and shared attentions, our model utilizes the correlation between aspects and corresponding sentiments effectively. Experiments show that the proposed AS-Capsules model achieves state-of-the-art performance without the need of linguistic knowledge. We show that the capsules are able to identify words best reflect the aspect category and sentiment tendency. We also show that the model can be directly applied to restaurant reviews on Yelp, demonstrating its effectiveness and robustness.

ACKNOWLEDGMENTS

This work was partially supported by Singapore Ministry of Education Research Fund MOE2014-T2-2-066, the National Key R&D Program of China (Grant No. 2018YFC0830200), the National Science Foundation of China (Grant No. 61876096/61332007), and China Scholarship Council.

REFERENCES

[1] Abdulaziz Alghunaim, Mitra Mohtarami, Scott Cyphers, and Jim Glass. 2015. A Vector Space Approach for Aspect Based Sentiment Analysis. In *Proc. NAACL HLT*. 116–122.
[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014).
[3] Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proc. ICML*. 160–167.
[4] Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proc. EMNLP*. 1615–1625.
[5] Gayatree Ganu, Noemie Elhadad, and Amélie Marian. 2009. Beyond the Stars: Improving Rating Predictions using Review Text Content. In *Proc. WebDB*.
[6] Zhen Hai, Kuiyu Chang, and Jung-jae Kim. 2011. Implicit Feature Identification via Co-occurrence Association Rule Mining. In *Proc. Computational Linguistics and Intelligent Text Processing - International Conference, CICLing, Part I*. 393–404.
[7] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2018. Exploiting Document Knowledge for Aspect-level Sentiment Classification. In *Proc. ACL, Volume 2: Short Papers*. 579–585.
[8] Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. 2011. Transforming Auto-Encoders. In *Proc. ICANN, Part I*. 44–51.
[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.
[10] Minqing Hu and Bing Liu. 2004. Mining Opinion Features in Customer Reviews. In *Proc. National Conference on Artificial Intelligence, Conference on Innovative Applications of Artificial Intelligence*. 755–760.
[11] Niklas Jakob and Iryna Gurevych. 2010. Extracting Opinion Targets in a Single and Cross-Domain Setting with Conditional Random Fields. In *Proc. EMNLP*. 1035–1045.
[12] Nobuhiro Kaji and Masaru Kitsuregawa. 2007. Building Lexicon for Sentiment Analysis from Massive Collection of HTML Documents. In *Proc. EMNLP-CoNLL*. 1075–1083.
[13] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A Convolutional Neural Network for Modelling Sentences. In *Proc. ACL, Volume 1: Long Papers*. 655–665.
[14] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proc. EMNLP*. 1746–1751.
[15] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).
[16] Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2015. Molding CNNs for text: non-linear, non-consecutive convolutions. In *Proc. EMNLP*. 1565–1575.
[17] Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.
[18] Qiao Liu, Haibin Zhang, Yifu Zeng, Ziqi Huang, and Zufeng Wu. 2018. Content Attention Model for Aspect Based Sentiment Analysis. In *Proc. WWW*. 1023–1032.
[19] Tomáš Mikolov. 2012. Statistical language models based on neural networks. *Presentation at Google, Mountain View, 2nd April* (2012).
[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proc. NIPS*. 3111–3119.
[21] Andriy Mnih and Geoffrey E. Hinton. 2008. A Scalable Hierarchical Distributed Language Model. In *Proc. NIPS*. 1081–1088.
[22] Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets. In *Proc. Workshop on Semantic Evaluation, SemEval@NAACL-HLT*. 321–327.
[23] Tony Mullen and Nigel Collier. 2004. Sentiment Analysis using Support Vector Machines with Diverse Information Sources. In *Proc. EMNLP*. 412–418.
[24] Bo Pang and Lillian Lee. 2007. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval* 2, 1-2 (2007), 1–135.
[25] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proc. EMNLP*. 1532–1543.
[26] Verónica Pérez-Rosas, Carmen Banea, and Rada Mihalcea. 2012. Learning Sentiment Lexicons in Spanish. In *Proc. LREC*. 3077–3081.
[27] Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 Task 4: Aspect Based Sentiment Analysis. In *Proc. Workshop on Semantic Evaluation, SemEval@COLING*. 27–35.
[28] Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 Task 4: Aspect Based Sentiment Analysis. In *Proc. SemEval@COLING*. 27–35.
[29] Ana-Maria Popescu and Oren Etzioni. 2005. Extracting Product Features and Opinions from Reviews. In *Proc. HLT/EMNLP*. 339–346.
[30] Qiao Qian, Minlie Huang, JinHao Lei, and Xiaoyan Zhu. 2017. Linguistically Regularized LSTMs for Sentiment Classification. In *Proc. ACL*, Vol. 1. 1679–1689.
[31] Delip Rao and Deepak Ravichandran. 2009. Semi-Supervised Polarity Lexicon Induction. In *Proc. EACL*. 675–682.
[32] Seyyed Aref Razavi and Masoud Asadpour. 2017. Word embedding-based approach to aspect detection for aspect-based summarization of persian customer reviews. In *Proc. IML*. 33:1–33:10.
[33] Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proc. EMNLP*. 379–389.
[34] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. 2017. Dynamic Routing Between Capsules. In *Proc. NIPS*. 3859–3869.
[35] Kim Schouten and Flavius Frasincar. 2016. Survey on Aspect-Level Sentiment Analysis. *IEEE Trans. Knowl. Data Eng.* 28, 3 (2016), 813–830.
[36] Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *Proc. EMNLP*. 151–161.
[37] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. EMNLP*, Vol. 1631. Citeseer, 1642.
[38] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Proc. ACL, Volume 1: Long Papers*. 1556–1566.
[39] Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Target-Dependent Sentiment Classification with Long Short Term Memory. *CoRR* abs/1512.01100 (2015).
[40] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document Modeling with Gated Recurrent Neural Network for Sentiment Classification. In *Proc. EMNLP*. 1422–1432.
[41] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Learning to Attend via Word-Aspect Associative Fusion for Aspect-Based Sentiment Analysis. In *Proc. AAAI*. 5956–5963.
[42] Zhiqiang Toh and Jian Su. 2016. NLANGP at SemEval-2016 Task 5: Improving Aspect Based Sentiment Analysis using Neural Network Features. In *Proc. NAACL HLT*. 282–288.
[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proc. NIPS*. 6000–6010.
[44] Yequan Wang, Minlie Huang, Li Zhao, and Xiaoyan Zhu. 2016. Attention-based LSTM for Aspect-level Sentiment Classification. In *Proc. EMNLP*. 606–615.
[45] Yequan Wang, Aixin Sun, Jialong Han, Ying Liu, and Xiaoyan Zhu. 2018. Sentiment Analysis by Capsules. In *Proc. WWW*. 1165–1174.
[46] Jun Yang, Runqi Yang, Chongjun Wang, and Junyuan Xie. 2018. Multi-Entity Aspect-Based Sentiment Analysis With Context, Entity and Aspect Memory. In *Proc. AAAI*. 6029–6036.

[47] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical Attention Networks for Document Classification. In *Proc. NAACL HLT*. 1480–1489.

[48] Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O'Brien-Strain. 2010. Extracting and Ranking Product Features in Opinion Documents. In *Proc. COLING, Posters Volume*. 1462–1470.

[49] Yanyan Zhao, Bing Qin, Shen Hu, and Ting Liu. 2010. Generalizing Syntactic Structures for Product Attribute Candidate Extraction. In *Proc. HLT NAACL*. 377–380.

[50] Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. 2015. Representation Learning for Aspect Category Detection in Online Reviews. In *Proc. AAAI*. 417–424.

Quality-Sensitive Training! Social Advertisement Generation by Leveraging User Click Behavior

Yongzhen Wang
Indiana University Bloomington
Bloomington, Indiana, USA
kuadmu@163.com

Heng Huang
Alibaba Group
Hangzhou, Zhejiang, China
gongchong.hh@taobao.com

Yuliang Yan
Alibaba Group
Hangzhou, Zhejiang, China
yuliang.yyl@alibaba-inc.com

Xiaozhong Liu*
Indiana University Bloomington
Bloomington, Indiana, USA
liu237@indiana.edu

ABSTRACT

Social advertisement has emerged as a viable means to improve purchase sharing in the context of e-commerce. However, humanly generating lots of advertising scripts can be prohibitive to both e-platforms and online sellers, and moreover, developing the desired auto-generator will need substantial gold-standard training samples. In this paper, we put forward a novel seq2seq model to generate social advertisements automatically, in which a quality-sensitive loss function is proposed based on user click behavior to differentiate training samples of varied qualities. Our motivation is to leverage the clickthrough data as a kind of quality indicator to measure the textual fitness of each training sample quantitatively, and only those ground truths that satisfy social media users will be considered the eligible and able to optimize the social advertisement generation. Specifically, under the qualified case, the ground truth should be utilized to supervise the whole training phase as much as possible, whereas in the opposite situation, the generated result ought to preserve the semantics of original input to the greatest extent. Simulation experiments on a large-scale dataset demonstrate that our approach achieves a significant superiority over two existing methods of distant supervision and three state-of-the-art NLG solutions.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Natural language processing**; **Natural language generation**; • **Applied computing** → **Electronic commerce**.

KEYWORDS

Social Advertising; Natural Language Generation; Clickthrough Data; Neural Networks; Sequence-to-Sequence

*Corresponding Author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313536

ACM Reference Format:
Yongzhen Wang, Heng Huang, Yuliang Yan, and Xiaozhong Liu. 2019. Quality-Sensitive Training! Social Advertisement Generation by Leveraging User Click Behavior. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313536

1 INTRODUCTION

In the era of e-commerce, the purchase sharing via social networks is an organic form of sale promotion generated by the public[1], when a growing number of people make their buying decisions from the shared product links[2]. For instance, Alibaba users share product information over 150M times per day through social media sites like Weibo and WeChat, on which those users may one-click-share what they bought to relatives or friends conveniently. While these deliveries can be just plain URLs, online stores ideally want to present attractive advertising scripts to not only explain product features adequately, but also convert audiences into potential customers, as shown in Fig. 1. However, writing such eye-catching narratives does require substantial marketing skills. Even though content experts could be employed to create the social advertisements, merchants have to struggle with the cost incurred by an army of commodities.

As a solution to save the workforce, traditional software heavily relies on settled examples to auto-generate product descriptions, which in turn challenges both personalization and copyright. More recently, neural networks enable a data-driven architecture *sequence-to-sequence* (seq2seq) for the *natural language generation* (NLG) [1, 2], in which an encoder reads a sequence of tokens into a context vector, and then from that a decoder yields a sequence of specific outcomes. But unlike the machine translation with an end-to-end nature, a majority of e-commerce sales lack a pair of concise yet appropriate texts to construct the gold-standard training samples with respect to the seq2seq paradigm. Especially in the context of mobile social apps, online sellers tend to serve shoppers with a brief title as well as many detail images to post product information for the sake of *search engine optimization* (SEO), when too lengthy presentations are not suitable for both display and edit on the limited screens of cellphones. Under this circumstance, the

[1]https://www.bigcommerce.com/ecommerce-answers/what-are-social-share-buttons-and-how-do-they-impact-conversions
[2]https://sproutsocial.com/insights/social-media-ecommerce

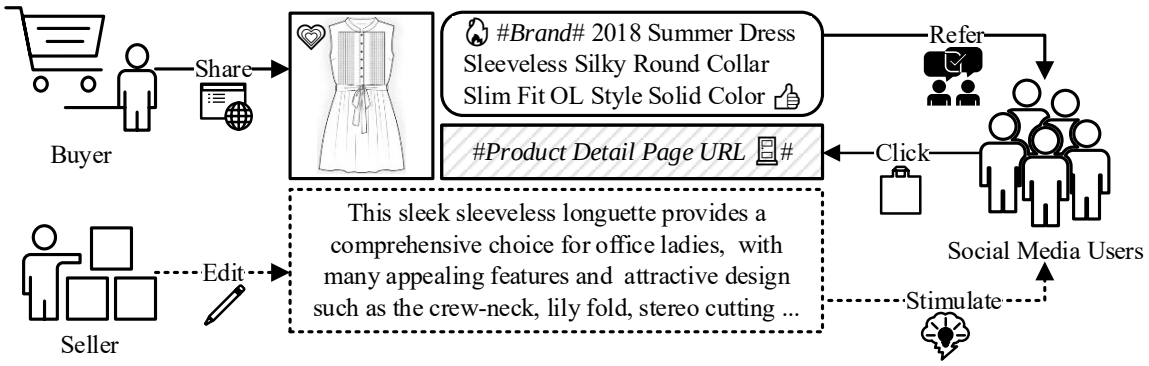


Figure 1: The illustration of the purchase sharing, in which a social advertisement (rounded textbox) is prepared for each product title (dashed textbox) accompanied by a raw URL (shaded textbox).

title often merely covers a few key attributes only, and moreover, drawing useful scripts from those photos is far more difficult.

Concerning the seq2seq NLG, a qualified training sample always requires some textual fitness to hold between its input and output, such as the semantic equivalence for machine translation, and the language logic for question answering. To address the qualities of training samples, latest studies try a widely used toolkit ROUGE [21] to calculate word co-occurrences from inputs to outputs exactly [6, 7, 24, 34]. Meanwhile, another strand of works attempt to retrieve approximate outputs as soft templates to offer additional guidelines [5]. Nonetheless, these methods of distant supervision [23] can hardly apply to the *social advertisement generation* (SAG) for the following reasons. First, due to vocabulary variation, counting the lexical overlaps could barely present the correct relationships between product titles (most readily available) and social advertisements (composed by heuristics). Second, the rough advertising templates would inevitably bring about irrelevant product details, which might confuse social media users and thus cause a decrease in the related purchase conversion rate. Last but most important, the performance of a social advertisement is highly dependent on how well it attracts audiences to click-return, and this type of quality assessment should be intrinsically dynamic because of the evolving interest of social media users outside. These reasons motivate us to re-investigate the measurement for textual fitness, and take the user-centric qualities of training samples into consideration.

In this study, we aim at the SAG to envision an innovative NLG application, in which user click behavior is leveraged to estimate the actual textual fitness of a candidate social advertisement to a particular product title. The basic idea lies in that the human-friendly advertising has a greater potential to improve the *clickthrough rate* (CTR) of shared product links for social media sites. In other words, the varying CTR can be utilized as a kind of quality indicator to characterize the matching degrees of corresponding paired texts, for which the desired evaluation involves relevance, diversity, and interestingness three criteria that are not easy to judge by computers. Note that the former two indicate the properties of generated results themselves, whereas the third one reveals the up-to-date preference of social media users accordingly. Specifically, we integrate the source paired texts with their CTR logs to establish a latent semantic space, where any pair of the product title and social advertisement could be mapped into two embeddings respectively so that the textual fitness may be measured therefrom. More formally, this measurement can be interpreted as a certain genre of supervision strength under the view of Internet crowd. With such a distant supervision, we are able to assess the qualities of training samples quantitatively, and the better the textual fitness grows, the higher the quality becomes.

The contribution of this paper is threefold. First, we explore the textual fitness measurement via a latent semantic space that features user click behavior, which enables the user-centric qualities of training samples for the seq2seq NLG. Second, we develop a novel seq2seq model to generate social advertisements automatically, where a quality-sensitive loss function is proposed to differentiate the training samples of varied qualities. Third, we conduct extensive experiments on 1.18M samples with their ground truths extracted by heuristics, and simulation results demonstrate that our approach outperforms two existing methods of distant supervision and three state-of-the-art NLG solutions.

2 RELATED WORK

This study touches on several strands of research within e-commerce NLG, click logs mining, and distant supervision as follows.

E-commerce NLG It is well acknowledged that the NLG offers dynamic improvements for e-commerce by turning product data into reader- and SEO-friendly narratives[3]. As one of the typical applications, Wang et al. [32] present a statistical framework that generates product descriptions from related attributes, in which some selected templates are combined with writing knowledge to decide what and how to say. More recently, Wang et al. [33] propose the first data-driven solution to compress product titles for the sake of SEO, which makes use of a pointer network-based model under the settings of multi-tasks. Subsequently, Gong et al. [15] and Sun et al. [28] further put forward two methods to summarize product titles, where the former utilizes three different categories of features in parallel, and the latter introduces a knowledge encoder as the auxiliary tool. Not long ago, Mathur et al. [22] employ several seq2seq models on a multi-lingual dataset, and thereby create a joint

[3]https://multichannelmerchant.com/blog/natural-language-generation-coming-ecommerce

model to produce product titles in various languages. Nonetheless, these works all conform to one formulation that the information of outputs is entirely contained in that of inputs, which differs from the SAG that parts of the social advertisements are free from or beyond their associated product titles.

Click Logs Mining In the field of information retrieval, click logs mining has proved to be a powerful means toward optimizing search engines [18, 20]. The most well-known model on query-document matching is the latent semantic analysis [9], by which a document or query can be mapped into a low-dimensional concept by the single value decomposition of corresponding document-term or query-term matrix. On this basis, Gao et al. [12, 13] develop a document ranking model trained on clicked query-document pairs, which ranks the documents by the likelihood of given query being a semantics-based translation of each document, with the assumption that the two sources of texts share the same distribution over semantic topics. As a further research, Huang et al. [17] propose a *deep structured semantic model* (DSSM) that projects both queries and documents into a common low-dimensional space respectively. This model is discriminatively trained by maximizing the conditional likelihoods of clicked documents regarding a particular query, and is considered the state-of-the-art in comparison to most previous works. However, despite these positive examples of exploiting click logs, few studies have applied the user click behavior to the NLG domain for either locating or sampling important contents.

Distant Supervision Over the past years, distant supervision has made a great success in a series of classification tasks featuring a weakly labeled training dataset [10, 23]. Inspired by this observation, Cao et al. [6] develop a seq2seq model for extractive summarization, in which each candidate sentence with comparatively high ROUGE-2 [21] score is annotated as a positive training sample. Similarly, Cheng and Lapata [7], Nallapati et al. [24], and Wang et al. [34] all adopt this labeling strategy and further raise three different seq2seq summarizers separately, where the sentence labels serve as ground truths to offer the essential guidelines for training samples. Most recently, Cao et al. [5] utilize *Lucene*[4] to retrieve approximate summaries as pseudo annotations for abstractive summarization, which enhances the textual fitness between the input and output inside each training sample. Unfortunately, although these studies improve the qualities of training samples in diverse ways, they lack an effective mechanism to integrate such kind of supervision information into the training process of the seq2seq paradigm.

In this study, our proposed seq2seq model derives from the established pointer-generator network [25], in which we put forward a quality-sensitive loss function to differentiate the training samples of varied qualities. In particular, the involved quality assessment builds upon the concept of the DSSM [17], which integrates the source paired texts with their CTR logs to develop a latent semantic space for measuring the textual fitness of a candidate social advertisement to a particular product title. To the best of our knowledge, we make the first attempt to address the problem of generating social advertisements automatically, and the practice of applying CTR data into the NLG domain is also less explored. Besides, this work is launched on a simulation environment, where a total of 180K experimental samples are constructed and hence make our results more convincing.

[4]https://lucene.apache.org

3 METHODOLOGY

This section formulates the SAG into a seq2seq NLG problem, as shown in Fig. 2, where the input and output are respectively a product title (green mark) and its paired social advertisement (red mark). To be detailed, Sect. 3.1 presents the encoder integrating with contextual features, Sect. 3.2 introduces the decoder making use of a pointer-generator mechanism, Sect. 3.3 elaborates how to set up the latent semantic space, and Sect. 3.4 describes the implementations for quality-sensitive training.

3.1 Contextual Encoder with NER

The goal of encoder is to transform an input sequence into a dense representation. Consider a product title of n words $T = \{t_{1:n}\}$, and each token $t_i \in \mathbb{V}$ (vocabulary list) can be represented by a word embedding $\boldsymbol{e}(t_i) \in \mathbb{R}^{d_e}$. In the NLG domain, previous studies have illustrated the superiority of *bidirectional long-short-term memory* (Bi-LSTM) in presenting sequential data, because of its capability to handle the long-range dependency and avoid the gradient vanishing [3, 26]. The Bi-LSTM encoder consists of both forward and backward two *recurrent neural networks* (RNNs), in which the hidden state at time step i can be denoted as follows:

$$\boldsymbol{h}_i = \left[\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i\right] \tag{1}$$

where each hidden state $\boldsymbol{h}_i \in \mathbb{R}^{2d_h}$ can be regarded as a local representation with focusing on the current and surrounding (before $\overrightarrow{\boldsymbol{h}}_{i-1}$ and after $\overleftarrow{\boldsymbol{h}}_{i+1}$, as Eq. 2-3) inputs simultaneously.

$$\overrightarrow{\boldsymbol{h}}_i = \text{LSTM}\left(\boldsymbol{e}(t_i), \overrightarrow{\boldsymbol{h}}_{i-1}\right) \tag{2}$$

$$\overleftarrow{\boldsymbol{h}}_i = \text{LSTM}\left(\boldsymbol{e}(t_i), \overleftarrow{\boldsymbol{h}}_{i+1}\right) \tag{3}$$

Despite the robustness of Bi-LSTM, as Fig. 1 shows that even though a product title conforms to a temporal sequence in a certain sense, a host of the phrases therein could swap their positions without any divergence, especially for those adjectives such as "*Sleeveless*" and "*Silky*". To better adapt the seq2seq paradigm, we draw out the fine-grained information from product titles via *named entity recognition* (NER) so as to enhance the sequential dependency of encoding stage. This empirical practice is derived from the contextual LSTM of [14], in which the extracted named entities are used as contextual features to capture the segment structures within text sequences. For instance, the *brand-name* is frequently followed by the *season-to-market*, whereas those adjacent modifiers seldom to restrict their appearance orders. More formally, an original token t_i is combined with its named entity $r_i \in \mathbb{V}$ as a composite input $\left[\boldsymbol{e}(t_i); \boldsymbol{e}(r_i)\right] \in \mathbb{R}^{2d_e}$ to feed the Bi-LSTM encoder, by which no additional parameters are introduced herefrom.

3.2 Attentional Decoder with Pointer

The decoder is used to generate an output sequence in terms of settled encoded representations. Given the hidden states of encoder

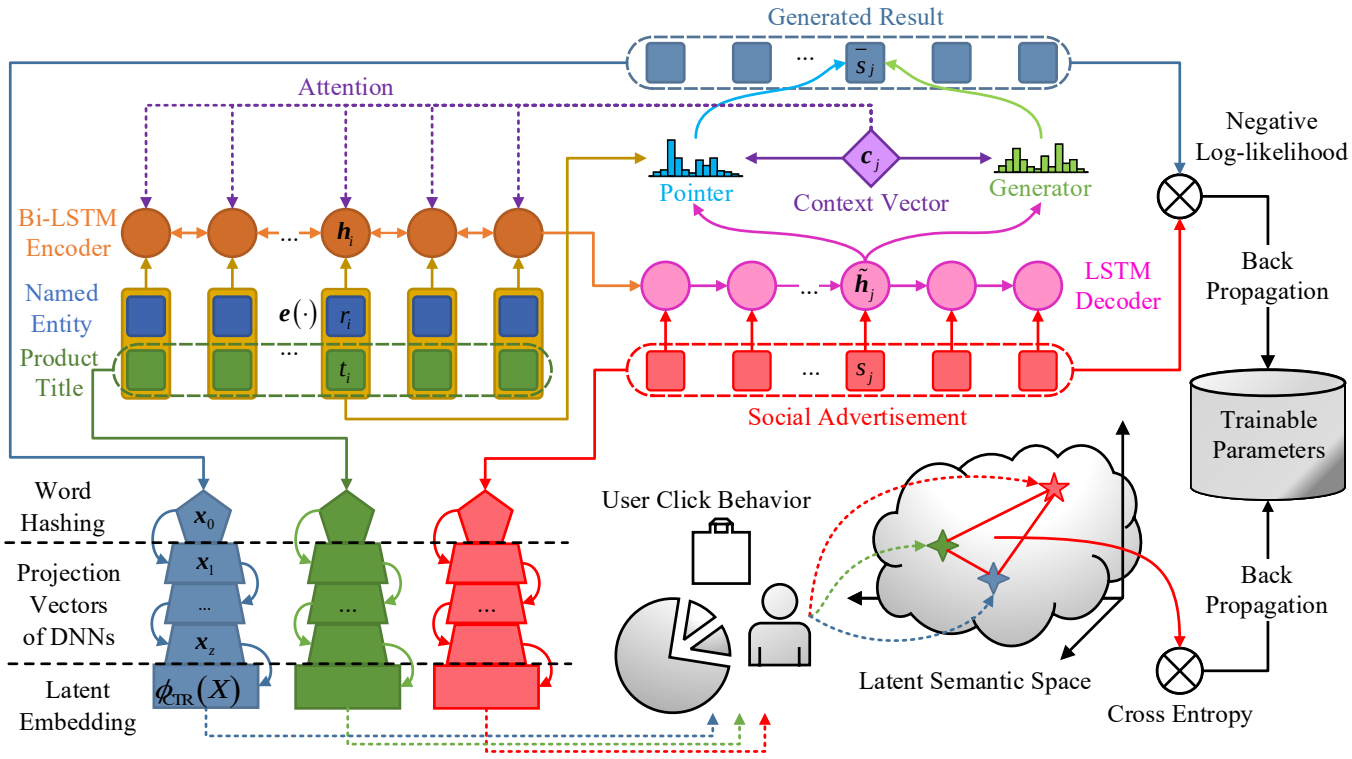


Figure 2: The seq2seq NLG framework formulated by the SAG.

$H = \{\boldsymbol{h}_{1:n}\}$, we apply an attention-based LSTM to sequentially compose a social advertisement $S = \{s_{1:m}\}$. To be specific, each hidden state of decoder $\tilde{\boldsymbol{h}}_j \in \mathbb{R}^{d_\text{h}}$ (as Eq. 5) is made by both the previous state and the last sampled word $\bar{s}_j \in \mathbb{V}$[5]. Next, this hidden state is concatenated with a context vector $\boldsymbol{c}_j \in \mathbb{R}^{2d_\text{h}}$ to feed through a linear layer and produce the vocabulary distribution $\Pr^{\text{voc}}_{j,1:|\mathbb{V}|}$ as below:

$$\Pr^{\text{voc}}_{j,1:|\mathbb{V}|} = \text{Softmax}\Big(\boldsymbol{W}_\text{v}\big[\tilde{\boldsymbol{h}}_j; \boldsymbol{c}_j\big] + \boldsymbol{b}_\text{v}\Big) \tag{4}$$

where $\boldsymbol{W}_\text{v} \in \mathbb{R}^{|\mathbb{V}|\times 3d_\text{h}}$ indicates a learnable matrix, and $\boldsymbol{b}_\text{v} \in \mathbb{R}^{|\mathbb{V}|}$ denotes a bias term. In particular, $\boldsymbol{c}_j$ (as Eq. 6) is the weighted sum of encoder hidden states, and the attention weight $\alpha^i_j \in [0, 1]$ (as Eq. 7) demonstrates how much the title token t_i contributes to sample the advertisement word s_j. This mechanism actually acts as an intermediate phase to determine which parts of the encoder to highlight for current samplings [1], in which a bi-linear learner $\boldsymbol{M} \in \mathbb{R}^{d_\text{h}\times 2d_\text{h}}$ is employed to measure such kind of relationship.

$$\tilde{\boldsymbol{h}}_j = \text{LSTM}\Big(\boldsymbol{e}(\bar{s}_j), \tilde{\boldsymbol{h}}_{j-1}\Big) \tag{5}$$

$$\boldsymbol{c}_j = \sum_{i=1}^{n} \alpha^i_j \boldsymbol{h}_i \tag{6}$$

[5]If during the training process, $\bar{s}_j$ is the current token of ground truth, or the last emission of decoder otherwise.

$$\alpha^i_j = \frac{\exp\left(\tilde{\boldsymbol{h}}^\top_j \boldsymbol{M}\boldsymbol{h}_i\right)}{\sum_{k=1}^{n}\exp\left(\tilde{\boldsymbol{h}}^\top_j \boldsymbol{M}\boldsymbol{h}_k\right)} \tag{7}$$

$$\Pr^{\text{opt}}_j = \text{Sigmoid}\Big(\boldsymbol{W}_\text{o}\big[\tilde{\boldsymbol{h}}_j; \boldsymbol{c}_j; \boldsymbol{e}(\bar{s}_j)\big] + \boldsymbol{b}_\text{o}\Big) \tag{8}$$

Unlike conventional NLG problems, the neologisms including coming trends and emerging fashions bloom in e-commerce every day, which results in a lot of *out-of-vocabulary* (OOV) words necessarily. To this end, we introduce an established pointer from [25] to transport title contents into our target advertisements directly. More formally, a probability $\Pr^{\text{opt}}_j \in \mathbb{R}$ (as Eq. 8, where $\boldsymbol{W}_\text{o} \in \mathbb{R}^{3d_\text{h}+d_\text{e}}$ and $\boldsymbol{b}_\text{o} \in \mathbb{R}$) is utilized to switch on the option for whether making a copy or not, and hence the final distribution $\Pr^{\text{dis}}_{j,1:|\mathbb{V}|}$ is obtained by adding a pointer component $\Pr^{\text{ptr}}_{j,1:|\mathbb{V}|}$ to a generator basis $\Pr^{\text{voc}}_{j,1:|\mathbb{V}|}$ as follows:

$$\Pr^{\text{dis}}_{j,1:|\mathbb{V}|} = \Pr^{\text{opt}}_j \times \Pr^{\text{voc}}_{j,1:|\mathbb{V}|} + \Big(1 - \Pr^{\text{opt}}_j\Big) \times \Pr^{\text{ptr}}_{j,1:|\mathbb{V}|} \tag{9}$$

$$\Pr^{\text{ptr}}_{j,k} = \begin{cases} \alpha^i_j, & \text{if} \quad k = \mathbb{V}_{t_i} \\ 0, & \text{otherwise} \end{cases} \qquad \forall k \in \mathbb{N}_+ \wedge k \le |\mathbb{V}| \tag{10}$$

where $\mathbb{V}_{t_i} \in \mathbb{N}_+$ returns the index of t_i in $\mathbb{V}$.

3.3 Latent Semantic Space Featuring CTR

Numerous works have confirmed the CTR as one of the most promising channels to access human evaluation on the Internet [20, 31]. In this study, we aim at the SAG to verify this hypothesis, by leveraging user click behavior to measure the textual fitness between product titles and social advertisements.

Generally speaking, a well-designed product title T is pieced by the main characteristics of sale promotion due to length limits. This messy display would certainly affect audiences' returning experiences, for which an appropriate social advertisement S is required to stimulate their shopping impulses effectively. Since evaluating such textual fitness is fairly computer-hard, we take the CTR as an alternative means to estimate the matching degrees of corresponding paired texts, as shown in Eq. 11.

$$\mathcal{Q}(T,S) = \text{Cosine}(T,S \mid \text{CTR}) = \frac{\phi_{\text{CTR}}(T)^\top \phi_{\text{CTR}}(S)}{\left\|\phi_{\text{CTR}}(T)\right\| \times \left\|\phi_{\text{CTR}}(S)\right\|} \tag{11}$$

where $\text{Cosine}(T,S \mid \text{CTR})$ represents the affinity of T to S within the latent semantic space $\phi(\cdot)_{\text{CTR}} \in \mathbb{R}^{d_{\text{s}}}$ featuring the CTR, which equates to promote an ocean of social media users to infer the textual fitness between T and S from semantics. In this manner, the consumer preference for social advertisements can be depicted to help assess the user-centric qualities of training samples quantitatively.

Specifically, we follow the mature practice of [17] to apply a stack of z *deep neural networks* (DNNs) for deriving the latent embedding of a text sequence X (T or S) as below:

$$\boldsymbol{x}_0 = \text{Hashing}(X) \tag{12}$$

$$\boldsymbol{x}_l = \tanh\left(\boldsymbol{W}_l \boldsymbol{x}_{l-1} + \boldsymbol{b}_l\right), \quad \forall l \in \{1, 2, \cdots, z\} \tag{13}$$

$$\phi_{\text{CTR}}(X) = \tanh\left(\boldsymbol{W}_{\text{s}} \boldsymbol{x}_z + \boldsymbol{b}_{\text{s}}\right) \tag{14}$$

where $\text{Hashing}(\cdot)$ indicates a function that converts from the one-hot representation X into the letter n-gram denotation $\boldsymbol{x}_0$ [27], $\boldsymbol{x}_{1:z} \in \mathbb{R}^{d_{\text{x}}}$ represents z projection vectors at different layers of the DNNs, and $\boldsymbol{W}_l \in \mathbb{R}^{d_{\text{x}} \times d_{\text{x}}}$, $\boldsymbol{b}_l \in \mathbb{R}^{d_{\text{x}}}$, $\boldsymbol{W}_{\text{s}} \in \mathbb{R}^{d_{\text{s}} \times d_{\text{x}}}$, $\boldsymbol{b}_{\text{s}} \in \mathbb{R}^{d_{\text{s}}}$ denote $2z + 2$ tensors for the linear maps within Eq. 13-14. In practice, the pairwise ranking strategy of [8] is adopted to tune all these parameters, in which a training sample with its own CTR above the threshold $\Theta \in \mathbb{R}$ is selected as a qualified individual (T, S^+), and otherwise as an unqualified one (T, S^-), with the purpose of assigning the qualified higher scores in comparison to the unqualified. More formally, our objective is to minimize the cost function $\mathcal{L}_{\text{rank}}$ defined as follows:

$$\mathcal{L}_{\text{rank}} = \text{Max}\Big(0, \Omega - \mathcal{Q}(T,S^+) + \mathcal{Q}(T,S^-)\Big) \tag{15}$$

where $\Omega \in \mathbb{R}$ denotes the marginal threshold. Once set up, the optimized latent semantic space will be responsible for inference only, and no longer get updated during training the SAG subsequently.

3.4 Quality-sensitive Loss Function

Conceptually, the seq2seq paradigm can be considered as a kind of conditional language model for the transduction tasks like machine translation [19, 29], in which the ground truths simply play the roles of data labels while their contained semantics are consistently ignored. It is fine if the every training process is quite perfect, but what if a significant portion of the training samples are not satisfactory? In this study, we aim at the SAG to put forward a quality-sensitive loss function $\mathcal{L}_{\text{qs}}$, as shown in Eq. 16.

$$\mathcal{L}_{\text{qs}} = \mathcal{Q}(T,S) \times \mathcal{L}_{\text{neg}} + \Big(1 - \mathcal{Q}(T,S)\Big) \times \mathcal{L}_{\text{ce}} \tag{16}$$

where $\mathcal{L}_{\text{neg}}$ and $\mathcal{L}_{\text{ce}}$ represent two different losses for the conventional negative log-likelihood (as Eq. 17) and the introduced cross entropy (as Eq. 18) respectively. Note that $\mathcal{L}_{\text{ce}}$ achieves the minimum value if and only if $\mathcal{Q}(T,S) = \mathcal{Q}(S,\bar{S})$.

$$\mathcal{L}_{\text{neg}} = -\log \Pr(S \mid T, \bar{S}) = -\sum_{j=1}^{m} \log \Pr^{\text{dis}}_{j,\mathbb{V}_{s_j}} \tag{17}$$

$$\begin{aligned} \mathcal{L}_{\text{ce}} = &-\mathcal{Q}(T,S) \times \log \mathcal{Q}(S,\bar{S}) \\ &- \Big(1 - \mathcal{Q}(T,S)\Big) \times \log\Big(1 - \mathcal{Q}(S,\bar{S})\Big) \end{aligned} \tag{18}$$

The rationale behind our loss allocation is as follows. Regarding the qualified case $\mathcal{Q}(T,S) \uparrow$, the ground truth S should be utilized as much as possible to supervise the whole training phase, and consequently reproduce itself into the generated result $\bar{S}$, namely $\bar{S} \cong S \rightarrow \mathcal{L}_{\text{neg}} \downarrow$. However, in the opposite situation $\mathcal{Q}(T,S) \downarrow$, we would like $\bar{S}$ to be as similar as the original input T within the latent semantic space $\phi_{\text{CTR}}(\cdot)$, namely $\mathcal{Q}(T,\bar{S}) \cong 1 \rightarrow \bar{S} \cong T \rightarrow \mathcal{Q}(T,S) \cong \mathcal{Q}(S,\bar{S}) \rightarrow \mathcal{L}_{\text{ce}} \downarrow$. In particular, $\mathcal{L}_{\text{qs}}$ will be immediately reduced to the common quality-insensitive training mode once $\mathcal{Q}(T,S) = 1$. To sum up, $\mathcal{Q}(T,S)$ is exactly a conditioner for the model outcome $\bar{S}$ to decide between the expected answer S and the trivial response T. Note that $\phi_{\text{CTR}}(\cdot)$ is employed here to not only assess the user-centric qualities of training samples, but also forecast the textual fitness of S and newly observed $\bar{S}$.

In essence, the proposed quality-sensitive training mode equips the seq2seq model with a potential pipeline to exploit source paired texts from the semantic level. Especially for the unqualified case, we cannot expect that the ground truth therein could supply positive word-by-word supervision during the training period, and instead it should serve as a benchmark to restrict the semantic scopes of all results to be produced, i.e., to prevent the decoding stage from being too random and thus avoid $\mathcal{Q}(T,S) \geq \mathcal{Q}(T,\bar{S})$. It is mainly because that an incompatible social advertisement is often prone to puzzle audiences and further lessen the related purchase conversion rate, for which we only hope that $\bar{S}$ could preserve the semantics of T to the greatest extent. This is somewhat analogous to impose a specific orientation into the common beam search [29] that is frequently embedded at decoder.

4 EXPERIMENT

4.1 Experimental Setup

This section presents the experimental setup for assessing our approach, including 1) the dataset used for training and testing, 2)

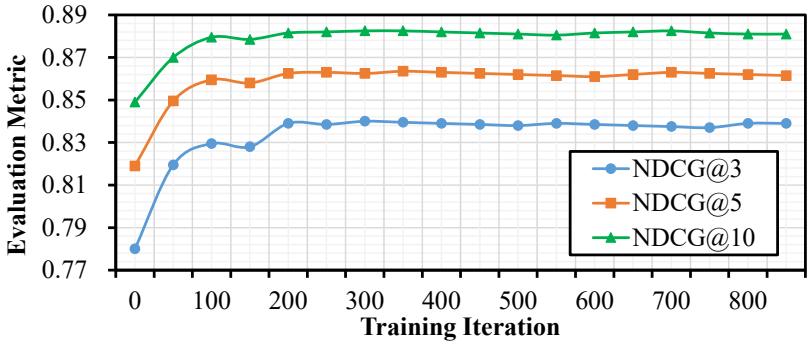


Figure 3: The optimization process of the latent semantic space, where the *normalize discounted cumulative gain* (NDCG) is applied to measure the accuracy of the pairwise ranking strategy.

the implementation details, 3) the evaluation metrics, and 4) the contrast methods.

Dataset We conduct extensive experiments on a dataset created from *Taobao*[6], in which all social advertisements are extracted from the associated product displays on search guide tabs (the heuristic rule: > 100 words and contain corresponding product names). Specifically, we select the clothing category (most actively traded) as our experimental subject and the dataset includes a total of 1.18M samples, where 1M of which are applied to build the latent semantic space, and the remaining 180K are used for developing the proposed seq2seq model. To be detailed, the 180K samples split into approximately 90% for training, 5% for validation, and 5% for testing. Once constructed, these samples are deployed in the context of purchase sharing via an online simulation environment (an instant messaging social app), and their CTR logs within a week are traced down to monitor the user click behavior. As a side note, we choose this dataset for the reason that no other open source data satisfy the condition to access an ocean of the CTR logs.

Implementation We use *Tensorflow*[7] for our implementations, in which the dimensions of word embeddings and hidden states are 128 and 256 respectively. More concretely, a vocabulary list of 70K words is built according to all training samples, and no pre-train operation is applied to the word embeddings in advance. Meanwhile, the maximum time step of encoder is set to 400, whereas that of decoder is fixed to 100 for training, and 35 for both validating and testing. Note that every LSTM parameter is randomly initialized over a uniform distribution within $[-0.02, 0.02]$. Based on these settings, we optimize the proposed seq2seq model using the AdaGrad [11] with a learning rate of 0.001, and perform the mini-batch gradient descent with a batch size of 16 for around 190K iterations (20 epochs in total). Moreover, as for the latent semantic space, we follow the established setup from [17] to utilize a stack of 3-layered DNNs for deriving the hierarchical text features, and the pairwise ranking strategy of [8] is adopted to tune the parameters therein. As can be seen from Fig. 3, the optimized space reaches an accuracy of at least 80% to discriminate the textual fitness for any pair of the product title and social advertisement.

Evaluation We take the widely used toolkit ROUGE [21] to evaluate the generated social advertisements automatically. To this end, we report both ROUGE-1 and ROUGE-2 (lexical overlaps for unigram and bigram) as a way to assess the informativeness, and ROUGE-L (the longest common subsequence) as a means to determine the fluency, in terms of the bytes of ground truths.

In addition, we conduct a human evaluation through the online simulation environment, on which a number of crowdsourcing workers are employed from *Datacrowd*[8] to compare between a list of social advertisements produced by all methods. Specifically, three criteria considered independently are as follows: 1) How *relevant* are the social advertisements to the corresponding product titles? 2) How *diverse* are the social advertisements for readers to easily grasp the expression changes? 3) How *interesting* are the social advertisements for audiences to proactively know about the product features? Note that we do not allow any correlation among workers during this evaluation, and each criterion is assigned with a 2-point scale of 0 (fail) and 1 (pass). By this means, each worker browses a set of 10 social advertisements of the same origin at random per session, and subsequently submits a score to judge every aspect of that paired method. To be exact, a total of 10 workers are responsible for each generated social advertisement, together with the Majority Voting [4] function to aggregate a final decision. Meanwhile, we also ask those workers to mark the ground truths so as to examine the raw qualities of whole experimental samples.

Benchmark To validate the proposed textual fitness, we compare our approach (denoted as P._{CTR}) against its four variants. The first one is to eliminate the cross entropy loss from Eq. 16, and thus adopt the common quality-insensitive training mode (denoted as P._{None}), which equates to setting $Q(T, S) = 1$ by default, namely no measurement for the textual fitness. The other three are all derived from the heuristic practice of [24], which applies the ROUGE as a substitute means to calculate the textual fitness from inputs to outputs, denoted as $\text{P.}_{\text{ROUGE-1}}$, $\text{P.}_{\text{ROUGE-2}}$, and $\text{P.}_{\text{ROUGE-L}}$ respectively.

To better verify the effectiveness of our approach, we also select five representative NLG methods as another benchmark group, including: 1) Re3Sum: using approximate outputs as soft templates to guide the seq2seq model [5]; 2) Transformer: a sequence transduction model based entirely on the self-attention [30]; 3) PGenerator:

[6]https://www.taobao.com
[7]https://www.tensorflow.org
[8]https://datacrowd.taobao.com

a hybrid pointer-generator network with the trade-off between copying and producing [25]; 4) CopyNet: the first work to incorporate copying into the neural network-based seq2seq learning [16]; and 5) Seq2ASeq: a standard seq2seq model with the attention mechanism [1].

For clarity, $P_{\cdot\text{ROUGE-1}}$, $P_{\cdot\text{ROUGE-2}}$, $P_{\cdot\text{ROUGE-L}}$ and Re3Sum belong to the two categories of existing solutions to address the qualities of training samples, where the former three focus on computing lexical overlaps, and the latter one turns to obtain additional information from similar outputs. Meanwhile, Transformer, PGenerator and CopyNet are three of the state-of-the-art solutions in the NLG domain, each with a different copying fashion to handle the OOV problem. Besides, Seq2ASeq is a classic model that integrates the seq2seq paradigm with the attention mechanism, which has witnessed a great success in various NLG applications like the machine translation.

4.2 Results and Discussion

Table 1 reports the ROUGE comparison between our approach and its four variants, in which $P_{\cdot\text{CTR}}$ gets the highest score on all three evaluation metrics, whereas $P_{\cdot\text{ROUGE-1}}$ receives the lowest ranking of the whole group. To be specific, we further draw the following conclusions:

1) $P_{\cdot\text{None}}$ vs. $P_{\cdot\text{CTR}}$: The use of the proposed textual fitness does contribute to improve both the informativeness (ROUGE-1 and ROUGE-2) and fluency (ROUGE-L) of the generated social advertisements.

2) $P_{\cdot\text{None}}$ vs. $P_{\cdot\text{ROUGE-1}}$: The unigram overlaps cannot provide enough evidence to characterize the textual fitness for any pair of the product title and social advertisement that might experience a drastic vocabulary variation.

3) $P_{\cdot\text{ROUGE-1}}$ vs. $P_{\cdot\text{ROUGE-2}}$ vs. $P_{\cdot\text{ROUGE-L}}$: The larger-grained word co-occurrences may present a deeper level of natural language understanding to capture the semantic affinity between product titles and social advertisements, and ROUGE-2 is the best choice among all its related versions to calculate the textual fitness, which conforms to the previous studies of [6, 7, 24, 34].

4) $P_{\cdot\text{ROUGE-2}}$ vs. $P_{\cdot\text{CTR}}$: The CTR indeed has a better ability than the literal alignment ROUGE to distinguish whether a candidate social advertisement suits to a particular product title, owing to its reflecting consumer preference for the advertising.

To better illustrate the power of user click behavior, Fig. 4 visualizes the value distributions of various textual fitness over all experimental samples. From this graph, the curve of the proposed measurement is extremely similar to the Gaussian distribution, whereas those of the other counterparts have a relatively narrow range of span, and thus would suffer difficulties to differentiate the training samples of varied qualities accurately. In addition, Table 2 summarizes the correlation tests between the proposed textual fitness and each of ROUGE-1, ROUGE-2, and ROUGE-L separately. It can be seen that ROUGE-2 obtains a bit greater coefficient than the others, which could partly explain the reason why $P_{\cdot\text{ROUGE-2}}$ ranks the second in Table 1.

Table 3 demonstrates the superiority of our approach over five representative NLG solutions, in which $P_{\cdot\text{CTR}}$ gains the highest ranking of all six methods. Specifically, $P_{\cdot\text{CTR}}$ achieves the maximum value on ROUGE-1 and ROUGE-2 simultaneously, but is slightly surpassed by Seq2ASeq and PGenerator respectively on ROUGE-L. However, the two counterparts' leading-edge is not statistically significant, and they both fall far behind our approach in terms of the informativeness (ROUGE-1 and ROUGE-2). To further explore the performance comparison, Table 4 presents the human evaluation on a total of 4,500 testing samples of varied qualities, where each percentage indicates the proportion of crowdsourcing workers who pass the corresponding criterion. Obviously, $P_{\cdot\text{CTR}}$ still occupies the first place of the whole group, which means that the conclusion drawn by Table 3 is sustained. More concretely, we can reach the following findings:

1) Ground Truth ranks the third only, and therefore is by no means the gold standard for developing an ideal seq2seq model, which is consistent with our above analysis, i.e., constructing qualified training samples toward the SAG is not an easy matter.

2) Nearly no diversity and interestingness (below 5%) exists in the social advertisements generated by Seq2ASeq, CopyNet, and PGenerator. In addition, we also find that these three generic RNN-based solutions repeatedly return several frequent patterns like "*This is the newest*" of evidently sequential dependency, which results in their higher scores on ROUGE-L (fluency) in Table 3.

3) With the ability to draw global dependencies from inputs to outputs, Transformer significantly improves the scores on all three criteria in comparison to the former baselines. However, because of adopting the conventional quality-insensitive training mode, this method will more or less be polluted by those unqualified (what social media users are disinteresting of) training samples, which consequently causes an obvious gap in the interestingness with either $P_{\cdot\text{CTR}}$ or Ground Truth.

4) While Re3Sum leverages the rough advertising templates to enhance both the diversity and interestingness, it still gets trapped into the poor relevance apparently due to introducing some irrelevant product details.

5) It is noteworthy that $P_{\cdot\text{CTR}}$ obtains significantly better diversity than Ground Truth. This is partly because that $P_{\cdot\text{CTR}}$ requires the ground truths to offer additional (user-centric) semantic constraints instead of merely word-by-word supervision for generating the social advertisements, which is particularly useful when facing the training samples with imbalanced textual fitness.

6) There is a large discrepancy in the evaluation results between ROUGE and crowdsourcing. For example, the ranking of Transformer in Table 4 is clearly higher than that in Table 3, and meanwhile it exceeds the lower expectation Seq2ASeq successfully, which to a certain degree proofs the usefulness and importance of human evaluation (user click behavior) in measuring the textual fitness.

4.3 Case Study

To better understand the effect of quality-sensitive training mode, Fig. 5 displays two typical examples from both the qualified and unqualified samples separately. For simplicity, we only compare the top three methods ($P_{\cdot\text{CTR}}$, Transformer, Re3Sum) in Table 4 against Ground Truth, and reasonably come to the following analysis:

1) While the rough advertising templates can offer a majority of proper details on sale promotion, Re3Sum still carries the risk to

Table 1: The ROUGE evaluation (%) of testing samples regarding our approach and its four variants.

Method \ Metric	ROUGE-1	ROUGE-2	ROUGE-L	Ranking
$P_{\text{-None}}$	23.64*	6.46*	16.38*	2.0000
$P_{\text{-ROUGE-1}}$	23.59*	6.45*	16.20*	1.0000
$P_{\text{-ROUGE-2}}$	24.04*	6.50*	16.51*	3.8333
$P_{\text{-ROUGE-L}}$	23.84*	6.50*	16.49*	3.1667
$P_{\text{-CTR}}$	**24.39**	**7.14**	**17.10**	**5.0000**

1) the Ranking is returned by the Friedman test based on ROUGE-1, ROUGE-2, and ROUGE-L, and the higher the better;
2) * indicates the Wilcoxon signed-rank test with $p < 0.01$, in comparison to $P_{\text{-CTR}}$.

Table 2: The correlation coefficients between the proposed textual fitness and each of ROUGE-1, ROUGE-2, and ROUGE-L separately.

Metric	ROUGE-1	ROUGE-2	ROUGE-L
the Proposed	0.1748	**0.2482**	0.2309

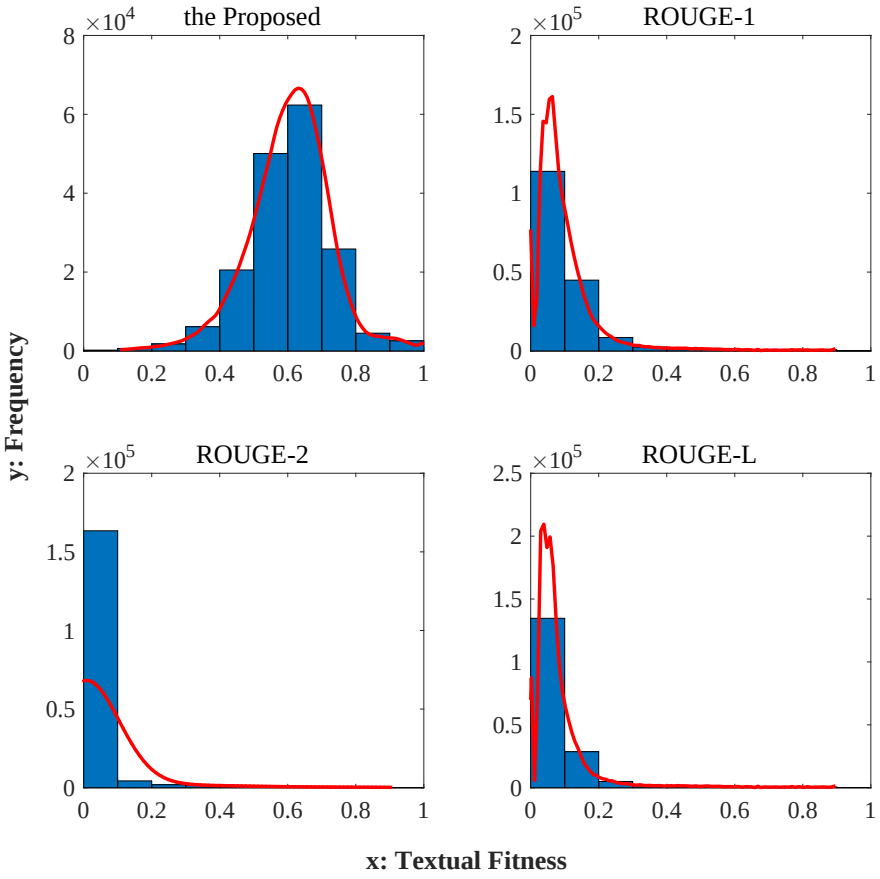


Figure 4: The frequency histograms and kernel density estimations of the proposed textual fitness, ROUGE-1, ROUGE-2, and ROUGE-L over all experimental samples.

Table 3: The ROUGE evaluation (%) of testing samples regarding our approach and five representative counterparts.

Method \ Metric	ROUGE-1	ROUGE-2	ROUGE-L	Ranking
Seq2ASeq	20.70*	4.37*	17.39	3.6667
CopyNet	20.24*	3.89*	16.79*	2.3333
PGenerator	20.85*	3.82*	**17.42**	3.6667
Transformer	21.60*	3.81*	15.53*	2.0000
Re3Sum	24.18	6.71*	16.69*	4.0000
$P_{\cdot CTR}$	**24.39**	**7.14**	17.10	**5.3333**

1) the Ranking is returned by the Friedman test based on ROUGE-1, ROUGE-2, and ROUGE-L, and the higher the better;
2) * indicates the Wilcoxon signed-rank test with $p < 0.01$, in comparison to $P_{\cdot CTR}$.

Table 4: The human evaluation (%) of testing samples regarding our approach and five representative counterparts.

Method \ Metric	Relevance	Diversity	Interestingness	Ranking
Seq2ASeq	72.52*	2.81*	4.10*	2.2500
CopyNet	25.48*	4.81*	0.37*	1.5000
PGenerator	96.91	2.90*	1.30*	3.2500
Transformer	**98.28**	89.80	53.73*	5.7500
Re3Sum	28.87*	70.69*	44.24*	3.5000
$P_{\cdot CTR}$	95.41	**92.48**	**79.58**	**6.2500**
Ground Truth	95.74	75.30*	74.15	5.5000

1) the Ranking is returned by the Friedman test based on Relevance, Diversity, and Interestingness, and the higher the better;
2) * indicates the Wilcoxon signed-rank test with $p < 0.01$, in comparison to $P_{\cdot CTR}$.

replace the given product names with the relevant yet inaccurate ones, just as in both the cases where "*hoodie*" and "*down pant*" respectively change into "*blouse*" and "*trousers*" of much higher generality.

2) Even though the self-attention can capture the correlation between specific products and associated descriptions, Transformer tends to produce some more common modifiers, such as "*round collar design → hoodie*" and "*with whatever T-shirts on → down pant*", which lacks of a few specialized adjectives toward the given product names. The reason is that, in terms of the conventional quality-insensitive training mode, the universal outcomes usually possess a lower damage to affect those unqualified training samples at decoding end.

3) Regarding the qualified case (left), $P_{\cdot CTR}$ can almost acquire an amount of product details as equal as Ground Truth does. But surprisingly, the former creates a far better narrative than the latter in the opposite situation (right), where Ground Truth yields the correct product names only, whereas $P_{\cdot CTR}$ generates several customized phrases with respect to the corresponding title contents. For instance, "*fits any figure well → slim*" and "*keeps warming → winter*". To sum up, the proposed quality-sensitive training mode provides the decoder a chance to take original inputs as a kind of backup ground truths and further consume the information therein.

5 CONCLUSION

In this paper, we highlight user click behavior in the context of the SAG via a latent semantic space that features the CTR, and on this basis, measure the textual fitness to quantitatively assess the user-centric qualities of training samples toward the seq2seq NLG. Notably, while social media users' interest varies over time, both the latent semantic space and quality assessment can change correspondingly. More specifically, we develop a novel seq2seq model to generate social advertisements automatically, in which a quality-sensitive loss function is proposed to characterize the quality distinction among training samples. In particular, for the qualified case, the ground truth should be utilized as much as possible to supervise the entire training process, but in the opposite situation, the generated result ought to retain the semantics of original input to the greatest extent. To sum up briefly, after an ocean of training samples are carefully projected into the CTR-based latent semantic space, only those relevant, diverse, and interesting contents will be oversampled to optimize the social advertisement generation. This strategy can be especially important for the purchase sharing concerning social media sites. Extensive experiments with real users demonstrate the validity of the proposed textual fitness, and the superiority of our approach over two representative methods of distant supervision and three state-of-the-art NLG solutions.

In future work, an appealing direction is to generalize our approach for other NLG applications as long as the user behavior information is available. For instance, in an explainable search scenario where the CTR logs are already prepared, we will be able to tailor the proposed seq2seq model so as to enhance the dynamic performance of content service.

Training Sample \ Quality Level		Qualified √	Unqualified ×
Product Title		 秋季时尚字母 Logo 印花连帽套头卫衣 Autumn Fashion Hoodie Printed with Letter Logos	 冬款白鹅绒通勤显瘦百搭羽绒裤 Winter Commuter Slim Casual Down Pant Made by White Goose Down
Social Advertisement	Ground Truth	纯色套头卫衣，简单大方，搭配胸前 Logo 印花，彰显时尚品味。 This is a solid-color simple and elegant hoodie, printed with letter logos on the chest to show your fashion taste.	找了一款羽绒裤的效果图供你们参考。 This is an effect drawing about the down pant for your reference.
	P_{CTR}	潮范儿十足的一款连帽卫衣，胸前的字母印花设计更显时尚潮流。 This is a hoodie full of fashion style, on which the letter printing design at chest highlights the latest trends.	休闲时尚的百搭羽绒裤，任何身形都可以驾驭，穿起来非常保暖。 This is a casual down pant featuring leisure fashion, which fits any figure well and keeps warming when in wearing.
	Transformer	这款卫衣采用了经典的圆领设计，让你的穿搭变得更加的时尚。 This hoodie has a classic round collar design, which makes your outfit more in fashion.	这款羽绒裤的版型是属于窄管的，随便搭配一件 T 恤就很好看。 This down pant follows a skinny shape and looks pretty with whatever T-shirts on.
	Re3Sum	遮肉显瘦的一款小衫，穿上就能很好的衬托出一整个人的时尚感。 This blouse helps you look slim and brings out a whole sense of fashion when dressed.	这款长裤采用的是鹿皮绒的面料，打造属于时髦达人的冬季服饰。 This pair of trousers is made from suede fabrics to tailor-make winter clothes for fashionistas.

Figure 5: The case study, in which the phrases within each generated social advertisement are marked in red if they are directly related to the corresponding product title.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous referees for their valuable comments and helpful suggestions.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. 2016. End-to-end attention-based large vocabulary speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*. IEEE, Shanghai, China, 4945–4949.

[3] Yi Bin, Yang Yang, Fumin Shen, Ning Xie, Heng Tao Shen, and Xuelong Li. 2018. Describing video with attention-based bidirectional LSTM. *IEEE Transactions on Cybernetics* PP(99) (2018), 1–11.

[4] Caleb Chen Cao, Jieying She, Yongxin Tong, and Lei Chen. 2012. Whom to Ask? Jury Selection for Decision Making Tasks on Micro-blog Services. *Proceedings of the Vldb Endowment* 5, 11 (2012), 1495–1506.

[5] Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. ACL, Melbourne, Australia, 152–161.

[6] Ziqiang Cao, Wenjie Li, Sujian Li, Furu Wei, and Yanran Li. 2016. Attsum: Joint learning of focusing and summarization with neural attention. In *Proceedings of the International Conference on Computational Linguistics*. ACM, Osaka, Japan, 547–556.

[7] Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the Annual Meeting of the Association for*

Computational Linguistics. ACL, Berlin, Germany, 484–494.

[8] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[9] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science* 41, 6 (1990), 391–407.

[10] Jan Deriu, Maurice Gonzenbach, Fatih Uzdilli, Aurelien Lucchi, Valeria De Luca, and Martin Jaggi. 2016. Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (Workshop)*. ACL, San Diego, CA, 1124–1128.

[11] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[12] Jianfeng Gao, Xiaodong He, and Jian-Yun Nie. 2010. Clickthrough-based translation models for web search: from word models to phrase models. In *Proceedings of the International Conference on Information and Knowledge Management*. ACM, New York, USA, 1139–1148.

[13] Jianfeng Gao, Kristina Toutanova, and Wen-tau Yih. 2011. Clickthrough-based latent semantic models for web search. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, Beijing, China, 675–684.

[14] Shalini Ghosh, Oriol Vinyals, Brian Strope, Scott Roy, Tom Dean, and Larry Heck. 2016. Contextual lstm (clstm) models for large scale nlp tasks. *arXiv preprint arXiv:1602.06291* (2016).

[15] Yu Gong, Xusheng Luo, Kenny Q Zhu, Shichen Liu, and Wenwu Ou. 2018. Automatic generation of Chinese short product titles for mobile display. *arXiv preprint arXiv:1803.11359* (2018).

[16] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393* (2016).

[17] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the International Conference on Information & Knowledge Management*. ACM, San Francisco, USA, 2333–2338.

[18] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*. ACM, New York, USA, 133–142.

[19] Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. ACL, Seattle, USA, 1700–1709.

[20] Liangda Li, Hongbo Deng, Anlei Dong, Yi Chang, Ricardo Baeza-Yates, and Hongyuan Zha. 2017. Exploring query auto-completion and click logs for contextual-aware web search and query suggestion. In *Proceedings of the International World Wide Web Conferences*. ACM, Western, Australia, 539–548.

[21] Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*. ACL, Stroudsburg, USA, 71–78.

[22] Prashant Mathur, Nicola Ueffing, and Gregor Leusch. 2018. Multi-lingual neural title generation for e-Commerce browse pages. *arXiv preprint arXiv:1804.01041* (2018).

[23] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. ACL, Stroudsburg, USA, 1003–1011.

[24] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, San Francisco, USA, 3075–3081.

[25] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. ACL, Vancouver, Canada, 1073–1083.

[26] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, San Francisco, USA, 3295–3301.

[27] Henning Sperr, Jan Niehues, and Alex Waibel. 2013. Letter n-gram-based input encoding for continuous space language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. ACL, Sofia, Bulgaria, 30–39.

[28] Fei Sun, Peng Jiang, Hanxiao Sun, Changhua Pei, Wenwu Ou, and Xiaobo Wang. 2018. Multi-source pointer network for product title summarization. *arXiv preprint arXiv:1808.06885* (2018).

[29] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the Annual Conference on Neural Information Processing Systems*. MIT Press, MontrÃľal, Canada, 3104–3112.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Annual Conference on Neural Information Processing Systems*. MIT Press, Long Beach, USA, 5998–6008.

[31] Gang Wang, Xinyi Zhang, Shiliang Tang, Christo Wilson, Haitao Zheng, and Ben Y Zhao. 2017. Clickstream user behavior models. *ACM Transactions on the Web* 11, 4 (2017), 21.

[32] Jinpeng Wang, Yutai Hou, Jing Liu, Yunbo Cao, and Chin-Yew Lin. 2017. A statistical framework for product description generation. In *Proceedings of the International Joint Conference on Natural Language Processing (Short Paper)*. ACL, Taipei, 187–192.

[33] Jingang Wang, Junfeng Tian, Long Qiu, Sheng Li, Jun Lang, Luo Si, and Man Lan. 2018. A Multi-task learning approach for improving product title compression with user search log data. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, New Orleans, USA, 451–458.

[34] Yongzhen Wang, Xiaozhong Liu, and Zheng Gao. 2018. Neural Related Work Summarization with a Joint Context-driven Attention Mechanism. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. ACL, Brussels, Belgium, 1776–1786.

Demographic Inference and Representative Population Estimates from Multilingual Social Media Data

Zijian Wang*
Stanford University
zijwang@stanford.edu

Scott A. Hale
University of Oxford & Alan Turing Institute
scott.hale@oii.ox.ac.uk

David Adelani
MPI-SWS & Saarland University
dadelani@mpi-sws.org

Przemyslaw A. Grabowicz
MPI-SWS & University of Massachusetts
grabowicz@cs.umass.edu

Timo Hartmann
GESIS Cologne
timo.hartmann@gesis.org

Fabian Flöck
GESIS Cologne
fabian.floeck@gesis.org

David Jurgens†
University of Michigan
jurgens@umich.edu

ABSTRACT

Social media provide access to behavioural data at an unprecedented scale and granularity. However, using these data to understand phenomena in a broader population is difficult due to their non-representativeness and the bias of statistical inference tools towards dominant languages and groups. While demographic attribute inference could be used to mitigate such bias, current techniques are almost entirely monolingual and fail to work in a global environment. We address these challenges by combining multilingual demographic inference with post-stratification to create a more representative population sample. To learn demographic attributes, we create a new multimodal deep neural architecture for joint classification of age, gender, and organization-status of social media users that operates in 32 languages. This method substantially outperforms current state of the art while also reducing algorithmic bias. To correct for sampling biases, we propose fully interpretable multilevel regression methods that estimate inclusion probabilities from inferred joint population counts and ground-truth population counts.

In a large experiment over multilingual heterogeneous European regions, we show that our demographic inference and bias correction together allow for more accurate estimates of populations and make a significant step towards representative social sensing in downstream applications with multilingual social media.

CCS CONCEPTS

• **Human-centered computing** → **Social media**; • **Computing methodologies** → **Natural language processing**; **Object identification**; **Neural networks**.

KEYWORDS

Demographics; Post-stratification; Social Media; Latent Attribute Inference; Inclusion Probabilities; Multilingual; Deep Learning

*Work performed while at the University of Michigan
†Corresponding senior author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313684

ACM Reference Format:
Zijian Wang, Scott A. Hale, David Adelani, Przemyslaw A. Grabowicz, Timo Hartmann, Fabian Flöck, and David Jurgens. 2019. Demographic Inference and Representative Population Estimates from Multilingual Social Media Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313684

1 INTRODUCTION

Data representing the attitudes and behaviors of a (national) base population is of great importance to policy making, social science research and commercial prediction tasks, but representative surveys are expensive and infrequent. Social media data has been proposed as a real-time and inexpensive way to measure social phenomena—this area of research is known as "social sensing." Social media data has been used, with various levels of success, in areas such as infectious disease [30, 31, 51], migration and tourism [7, 50, 81], and box office takings for films [21, 55].

However, despite social media data being easily accessed, large in scale, and detailed, it is generally not representative of indicator variables measured in any broader (offline) population due to various biases [43, 67]. Especially the self-selection of users joining and participating on any given social media platform is a large source of error. In the UK for instance, Twitter users are more likely to be male and young compared to the national population [71], and in the U.S. men and residents of densely populated areas are overrepresented [35], while there exists a mix of over- and undersampling on Twitter for users with specific racial backgrounds [56]. In general, however, the exact inclusion probabilities for a specific demographic group on any platform in a given country or region are unknown. And while not all social sensing tasks require representative data [3], recent studies have pointed out the fallacies in predicting phenomena via social sensing without controlling for sampling biases in social media data [28, 29, 44], with research applications aiming to draw inferences for nation-wide target populations often relying on such attribute data. Consequently, a systematic methodology for estimating them is needed.

Survey analysis researchers have dealt with non-representative polls and non-response through sample re-weighting—with post-stratification being a well-known technique [10, 37]. These methods make use of basic demographics attributes: most prominently age, gender and location [60, 77]. Where demographic details are provided, post-stratification has been a valuable technique [77, 81].

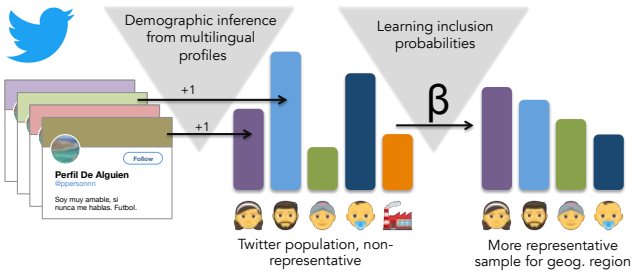


Figure 1: Our two-stage approach debiases non-representative Twitter data by (i) inferring demographics with state-of-the-art accuracy from multilingual, profile-only data and (ii) learning inclusion probabilities to create more representative samples for Europe-wide subregions

However, most social media platforms do not provide demographic data about their users, making it difficult to know or correct for such biases. Although promising techniques for extracting demographic attributes from social media have been proposed, there exists no robust multilingual approach for this task. We address these challenges by inferring basic demographic attributes and correcting for selection biases on a large sample of multilingual profile data from Twitter (Figure 1), one of the most used platforms for social sensing given its ease for obtaining data.

This work offers the following three contributions. First, we introduce a new multilingual, multimodal, multi-attribute deep learning system for inferring demographics of users (§2). This pipeline is built from the ground up to enable inference in 32 different languages, addressing the need for such methods beyond English and outperforming state-of-the-art methods on the tasks of predicting the age, gender and is-organization state of Twitter users. We show this system reduces algorithmic bias with respect to skin tone over the best-performing commercial systems. Second, we formalize a statistical framework for models that debias non-representative samples (§3). Our models explicitly learn per-stratum inclusion probabilities from data, whereas typical post-stratification methods either assume their knowledge or focus on obtaining a post-stratified estimator of a response variable. Third, in real-world evaluations, we show that our debiasing models significantly reduce the error rate in comparison to a model without post-stratification on the inferred demographic attributes (§4). In these ways, we make a significant step towards *representative social sensing* in downstream applications using multilingual social media.

2 DEMOGRAPHIC INFERENCE

We propose a new demographic inference model for three attributes: gender, age, and a binary organization indicator ("is-organization"). The first two attributes, gender and age, are widely reported in census data and are core features to accurately measure demographic biases. The third attribute was selected based on a known confounder for people-based studies of online platforms: the presence of accounts belonging to organizations [2, 54], which shall be distinguished from individual accounts. Next, we describe the model, its data, and its evaluation.

2.1 Classification Task

We consider the task of estimating population count in Twitter given a stream of user tweets. This scenario is motivated by the common research use case for social sensing on Twitter, where a researcher consumes one of the Twitter streams (e.g., the 10% or 1% sample streams) and desires to make some analysis from the tweets of a broader phenomena that extends beyond Twitter. In such a scenario, many users are expected to appear rarely–possibly only once–which hinders the use of existing demographic methods that rely upon large amounts of text to infer gender [13, 17, 18, 25, 45, 49, 72, 73, 82] and age [33, 58, 59, 63, 65, 68].

Instead, we avoid making inferences based on tweets and focus on using only the information associated with a user's account, which allows our method to easily scale to large volumes of users without the need for significant quantities of text. Not stratifying users based on tweets has the added advantage of preventing downstream demographic biases to any social sensing tasks that analyze the same language. Ultimately, our method uses four sources of information, username, screen name, biography, and profile image, and is designed to operate effectively on multilingual data.

We formalize the three classification tasks as follows. Gender and is-organization are modeled as binary classification tasks.[1] Age is known to be a difficult inference task in social media [58, 82]. Here, we opt to categorize age in four levels: ≤ 18, $(18, 30)$, $[30, 40)$, $[40, 99)$. Our choice is motivated by (i) our ability to align these age ranges with census data reports and surveys [74, 75], and (ii) the difficulty of the task (even for humans) when finer-grained age classes are used [23, 76], while (iii) still making the resulting classes amenable to downstream tasks. While other attributes are possible for use in debiasing (e.g., education, income, etc.), ground truth population statistics for these attributes are not widely available in censuses, making it difficult to train and evaluate any model.

2.2 Method and Training Procedure

Demographic attributes are often expressed in both image and text. We propose a new multimodal inference model with a novel training technique that leverages both of these sources to significantly augment our predictive accuracy. We first describe the model and each modality and then discuss the training procedure.

2.2.1 Model. The full model comprises two separate pipelines for processing a profile image and each of the three text sources of information, as shown in Figure 2. Here, we use a shared pipeline for all three attributes, using multi-task learning for the output. We refer to this as the **M3** model, after its multimodal, multilingual, and multi-attribute abilities. Following, we describe the architectures of each modality and how they are combined.

Image Model The image classifier was constructed using DenseNet [38] based on initial performance experiments over current state-of-the-art vision models on our data. We scale all profile images to 224x224 to meet the input size of the model.

[1] We base our notion of gender as one of performance [22], in which individuals adapt their style, name, and picture to (de)emphasize certain aspects of their gender identity [24]. While our training data uses self-declared binary gender identities, the M3 model reports gender identity as a continuum using prediction probability along a graded scale. We recognize that the current approach does not yet support non-binary gender identities, which we view as an important future task for full representation.

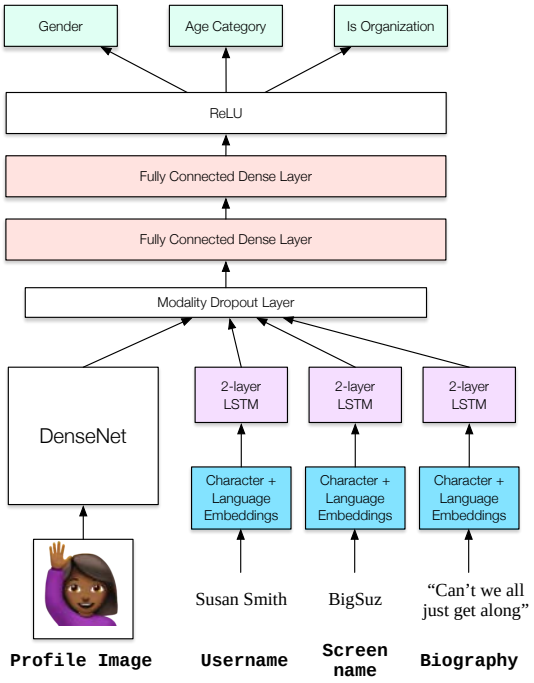


Figure 2: The M3 model for inferring gender, age, and organization-identity from image and text data.

Text Model Separate character-based neural models were trained for each text input and then a multi-input model was tuned based on the weights from each single model. Embeddings for username and biography were limited to the 3000 most frequent characters in our training corpus, with remaining characters represented with an encoding for their unicode category; screen name are embedded in the ASCII range. We use 2-stack bidirectional character-level Long-short Term Memory (LSTM) architecture [36] to capture necessary information. To incorporate potential different meanings in different languages for the same character, we concatenate a separate trainable language embedding to each character embedding and add a fully connected layer before being fed into an LSTM. The fusion of character-based input with language embeddings provides two critical benefits that let us scale to the linguistic diversity seen in global platforms like twitter. (1) Character-based RNNs are able to capture morphological regularity within a language and substantially reduce the parameter space due to the reduced vocabulary size of the model (i.e., the set of characters) as compared with word-based models with orders-of-magnitude larger vocabularies [15, 27, 46]. (2) By fusing language embeddings with character embeddings, the joint embedding lets us represent shared linguistic regularity *across* languages, which lets us better scale to the multilingual environment.

Full Multimodal Model To construct the full model, separate models are fully trained for image and text for all attributes. Then, the softmax output layer of each is removed and the two models are joined into a new model with a modality drop-out layer (described next), two fully connected layers of size 2048, a Rectified Linear Unit (ReLU) layer [57], and separate output layers for each task. This full model is then fine-tuned using the pretrained single-modality models' parameters for initialization.

2.2.2 Training Procedure. Rather than training M3 end-to-end initially, we combine three procedures for the training process: co-training, multilingual data augmentation, and modality drop out.

Co-Training Both image and text can have clear signal of demographics. Given the potentially-limited training data for speakers of less common languages, we take advantage of the multiple views of the same users by incorporating co-training: a semi-supervised learning technique. Co-training first learns a separate classifier for each view using any labeled examples. The most confident predictions of each classifier on the unlabeled data are then used to iteratively construct additional labeled training data [11]. Ardehaly and Culotta [4] recently showed co-training was effective for demographic classification in social media data using images and text in a graph-based co-training learning procedure.

In comparison to the variance of gender and age expressions in textual data across sociolinguistic contexts [6, 70], image data provides a more universal depiction, acting as an analog to a universal pivot language in machine translation [80]. Thus, we train the image-based portion of the M3 model first, which identifies high confidence labeled data from an unlabeled multilingual dataset. Second, the text-based portion of the M3 model is trained using the original dataset plus the text data associated with any high-confidence image-based classifications. Finally, we train the full M3 model with the training dataset and all high-confidence predictions from the unlabeled data. All models are trained as multi-task, where a common input representation and architecture is used to predict each output tasks. The probabilities for each task were represented using a top softmax layer.

Multilingual Data Augmentation The M3 model is designed to operate in online environments on tens of languages. However, few approaches and little labeled data exists for this task for languages other than English, with some exceptions [16, 32, 62]. Even with co-training, the relative differences in number of speakers between languages (e.g., English vs. Slovenian) create a substantial class imbalance which would prevent the model from effectively learning signals of each attribute in users' biographies written in less-frequent languages. Therefore, we perform data augmentation by using automatic machine translation of the training data.

Given the scale of our training data (described later in §2.3), the use of online translation services is prohibitive both in terms of time and cost. Instead, we opt to use word-based translation using the MUSE bilingual dictionaries [20]. For each instance in the training data, all words in the bilingual dictionary are replaced with the translation. A translated instance is only kept if 80% or more of its words could be translated. While word substitution typically produces translations that are more incorrect and less fluent, we observed that the remaining biography translations were often of reasonable quality due to the declarative nature of biographies, which often consist primarily of lists of readily-translatable noun descriptions or attributes of the user without using full grammatical sentences. Such translations also largely capture topical and semantic regularity, even if the morphology is not precise.

Modality Drop-out The co-training model uses a conjoining architecture with pre-trained weights from the image model and the

multi-input text model. To facilitate the model using each source of information, we introduce a new technique, modality dropout, that can fully hide a source of information (e.g., the biography) during training. We add an extra input dropout with a probability of 0.25 (i.e., keep 3 out of 4 input sources on average) to better regularize the model and support use cases where not all 4 input sources are available. When doing dropout, images were replaced by a random matrix while text was turned to a specific empty embedding.

2.3 Data

Training and evaluating M3 uses five distinct datasets: (1) a large dataset of Twitter profiles whose gender and age are heuristically identified, predominantly in English, (2) a curated dataset of Twitter accounts belonging to organizations, (3) an image dataset of faces from IMDB and Wikipedia, (4) a massive unlabeled set of users, and (5) a crowdsourced dataset for all three attributes that spans all 32 languages in our study. We describe each next.

2.3.1 Heuristically-identified users. Heuristically-identified users were drawn from a 10% sample of Twitter from 2014 to 2017 on Twitter, where roughly 40% of the users are English speakers [34]. Here, we identify gender and age from fixed expression in users' biographies that signal age or gender, e.g., "mother of two wonderful kids" or "26 y/o dude" using a gender- or age-signaling word. Age data was further augmented by identifying tweets wishing another user a happy birthday (cf. Al Zamal et al. [1]). These pattern-based approaches are known to be highly precise for certain attributes when properly constructed [8, 9]. Patterns were created for five languages: English, Spanish, French, German, and Swedish.[2] Organization accounts were manually curated by identifying 676 Twitter lists that primarily contained organizations, such as non-profits, local clubs, companies, or municipal services. These lists resulted in 59.92K unique organization accounts. For heuristically-identified users and organization accounts, we collect their current profile image, screen name, username, and biography. User age is adjusted to the present day. For heuristically-identified users, the biography is altered to replace the gender- or age-indicating word with a special token so that the model is forced to look for additional cues to recognize each attribute. The IMDB-WIKI dataset [66] consists of 523,051 images from headshots of actors in the IMDB and profile pictures in Wikipedia. This dataset provides an auxiliary source of information for fine-tuning the image-based part of the M3 model. These three datasets constitute our *initial data* prior to performing the three-step annotation procedure. Table 1 shows the detailed statistics of the data for each step.

Co-training depends on having access to a large set of unlabeled users whose high-confidence labels from one modality (e.g., profile image) can be used to augment the training set for another modality. Here, we collect 36.97M profiles speaking one of the 32 languages commonly spoken in Europe but for which we have no groud truth label. Languages were identified using CLD2 [53]. Note that users in this unlabeled dataset will only become training instances if one view's classifier (text or image) labels them with high confidence.

2.3.2 Crowdsourcing Data. Current gender, age, and organization datasets are nearly all produced for English-speaking users, which prevents us from evaluating the M3 model in the multilingual environment. Therefore, we constructed a new dataset of up to 200 randomly-selected Twitter users speaking one of each of the 32 languages in our study. Three annotators on Figure Eight[3] were shown the usernames, screen names, short biographies, and profile images of each Twitter account. We instructed workers to determine gender and age of a given user from two drop-down menus based only on this information. The selection options regarding gender also included a category for profiles belonging to multiple persons, and to a non-person (e.g., organization/bot). Regarding age, annotators were given seven categories to choose from: <= 18, $(18, 30)$, $[30, 40)$, $[40, 50)$, $[50, 60)$, $[60, 70)$, >= 69. They could select "don't know" for all questions, if they felt not enough information was available. We employed Figure Eight's selection mechanism for speakers of the target language of the set to be annotated, or chose workers by their country of residence, if one of that country's main spoken languages was identical with the target language of the task. Additionally, each job included a set of 30 test questions, annotated as a gold standard by the authors. These were used to screen for spam but also for any non-speakers of the target language remaining after our first filter; to this end, at least 10 questions were only solvable by sufficiently understanding the target language.

Annotation reliability was measured for each Figure Eight job using Krippendorff's α [48]. For the computations of α, we excluded all English test-questions from the 31 non-English jobs, profiles with only one annotation, and included "don't know" answers as an answer option. To establish a measure of how well workers did on the age classification in respect to the age brackets classified by M3, we collapsed age bracket encodings over 40 into one category. Across all languages, the mean α for the gender annotations lies at 0.81 and at 0.57 regarding the age brackets, hinting at the high difficulty of the Twitter profile age classification task for humans. For a separate measure, we recoded all gender and organization annotations into the two categories "is organization" and "is not organization" and calculated Krippendorff's α based on these data, resulting in a mean of approximately 0.75 across all languages, showing that workers could identify non-personal accounts reasonably well. The final dataset is constructed from all instances where at least two annotators agreed on the label for a particular attribute (majority vote), resulting in 4,732 profiles across 32 languages.

2.3.3 Data Partitions. All self-reported labeled data is partitioned into 80% train, 10% development, and 10% test splits. No official

Table 1: Sizes of datasets used for training and testing. Incremental numbers are shown for the outputs of data-augmentation from co-training and translation.

Dataset	**Gender**	**Age**	**Is-Org.**
Initial	3.98M	1.20M	59.92K
After Image Co-Training	+2.26M	+0.62M	+6.31M
After Translation Augmentation	+5.74M	+1.54M	+4.74M
After Text Co-Training	+8.28M	+0.78M	+11.87M
Final	14.53M	2.61M	23.86M
Held-out Evaluation	0.38M	0.36M	6.91K

[2]Languages were selected to capture linguistic diversity and where high precision demographic-identify patterns could be created easily.

[3]https://www.figure-eight.com/

train/test/dev splits are available for the IMBD-WIKI datasets so we perform our own partition with the same percentages as for self-reported. During model training, only the development set for self-reporting users are used for model selection. Owing to its small size, crowdsourcing data was only used for testing purposes. For the organizational data, we create a full dataset with non-organizational accounts by randomly sampling accounts from the heuristically labeled data to attain a 1:9 ratio, following the report in McCorriston et al. [54] that organizations make up 10% of the accounts on Twitter.

2.3.4 Co-Training Setup. The training data is balanced for each stage, where we oversampled gender and organization status, and undersampled the two most-frequent classes (0-18 and 19-29) while oversampling the other two age classes. High confidence thresholds were set to 0.9 for gender and organization status in both the image and text modalities. As a four-class classification task, the high-confidence for age is set to 0.7 in the image classifier, based on early examination of the prediction quality. We observed that age is more readily predictable in text and therefore used a threshold of 0.9 to keep quality high while also providing a sufficient number of new accounts for co-training.

Separate image and text models are each trained for 10 epochs. Due to the imbalanced number of instances per attribute (Table 1), each epoch contains 10000 steps where each step contains a mini-batch of 128 instances balanced across the three attributes. This setup ensures the model sees instances of the smaller datasets (age and organization). The image model is trained first, and its high-confidence instances are then used to train the text model. The final co-training model was trained for 5 epochs with input dropout and another 3 epochs without input dropout for better convergence. Amsgrad optimizers [64] were used in all training process, with a learning rate of 0.001 for ground-up models and 0.0005 for fine-tuning models. Text models were trained on one GPU, while the image and co-training models were trained in parallel on 4 GPUs. The full pipeline was built in PyTorch [61].

2.4 Evaluations

Accurate inference of representative attitudes from social media relies upon accurate stratification approaches. Here, we test the M3 model against current state of the art systems to describe the potential for error to affect demographic bias.

2.4.1 Comparison Systems and Data. The M3 model is compared with state-of-the-art systems from each modality and attribute. For images, we compare with Face++ [42] and Microsoft Face API [5] on age and gender performance. For text, we compare with three current state-of-art systems for inferring gender from usernames, genderperformr [78], demographer [47] and Jaech et al. [39], which could feasibly operate on names from multiple languages.

No multilingual organization recognition system exists; so, we limit our evaluation to the only publicly-available dataset of person-vs-organization which was scored by the Humanizr [54] and Demographer [79] systems. This data consists of a uniform sample across Twitter accounts, which were approximately 10% organizations. Here, we recollect the user profiles for their 20,273 accounts, of which 18,587 (91.6%) were still available as of October 2018,

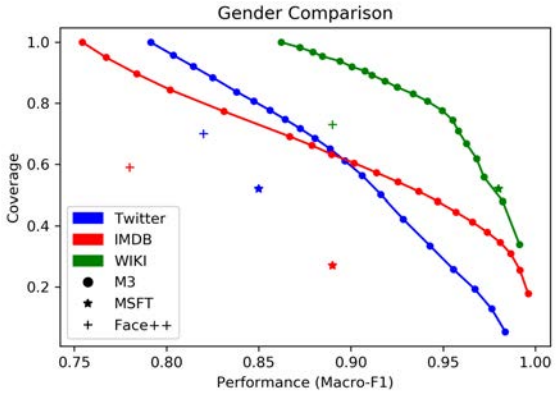


Figure 3: Model performances on gender classification for the three image datasets, as measured through F1. Coverage equals % of returned results at varying confidence levels for M3 and is the maximum available for the commercial APIs.

which matched the distribution originally reported. Their model was evaluated on the dataset using cross-validation, whereas we treat the entire data as a test set and report performance.

2.4.2 Gender Recognition. M3 produces state-of-the-art gender recognition for each of the three image-based datasets (IMDB, WIKI, and Twitter), as seen in Figure 3. M3 provides significant improvements in performance (F1) at each comparison system's level of recall.[4] The one exception to this trend is the performance of the MSFT classifier on the WIKI dataset, which has highly similar performance to M3. Given the MSFT's model large jump in performance, we hypothesize that their model may have been trained on Wikipedia data, although this cannot be confirmed. We observed that the increase in coverage of M3 is due in part to the model learning non-facial attributes associated with more than one gender, e.g., certain sports with the male gender, which allow M3 to process profile photos that other models cannot.

On the text-based data, M3 outperforms all systems except one when using just the username and masking all other information, shown in Table 2. When the model is allowed to see all other text information, M3 performance improves substantially, indicating it can effectively fuse several sources of information. While this latter setup uses more information than comparison systems, in practice, the Twitter API includes all text information used by M3 by default; so the only additional step is downloading the profile image.

2.4.3 Age Recognition. Age recognition is a significantly more difficult task in social media, as seen by the performances in Figure 4. The M3 model offers similar performance to commercial models on the IMDB and WIKI datasets, which primarily feature face-forward headshots in good lighting. However, in real-world Twitter profiles (shown in blue), the M3 model substantially outperforms both in F1 at each of their respective coverage levels, with 0.16 and 0.11 absolute improvements over Face++ and Microsoft, respectively.

[4]The coverage reported is the maximum available from the commercial APIs, which did not find a face or did not provide a gender/age estimate for each photo.

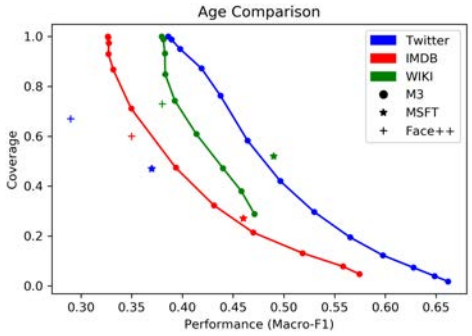


Figure 4: Model performances on age classification for three datasets, as measured through F1 on classified images and Coverage on the % of images classified.

2.4.4 Organization Recognition. For any task that draws inferences about human actors, non-human and specifically organizational accounts are a major source of error. The results of Table 3 show that our model is able to identify these accounts with 16.3% higher accuracy than the next closest system with no drop in performance at recognizing humans. Results on our test set from our organization data (§2.3) in Table 4 (top right) show similar performance, with an overall F1 of 0.898 that indicates high performance for both classes.

2.4.5 Multilingual Performance. As a full test of M3, we evaluate it against our crowdsourced account labels in 32 languages, shown in Table 4 top right. Performance in this multilingual setting is on par with the performance on the primarily-English heuristically-labeled data, though age classification remains the most difficult task. Figure 5 shows the performance per language for each attribute. Performance at predicting gender and is-organization are similar for most languages, while age performance varies significantly by language from 0.28 F1 for Bosnian to 0.73 for Slovenian and Welsh. These results together indicate that M3 is sufficiently accurate in the multilingual European environment.

Table 2: Performance comparison on gender using text methods. †Demographer only has a coverage of 0.974.

Method	Macro-F1
M3: Full Text	**0.907**
M3: Username Only	0.828
GenderPerformr [78]	0.835
Demographer [47]	0.781†
Jaech and Ostendorf [39]	0.763

Table 3: Performance at recognizing organizational accounts, measured as accuracy per class, following [54]

Method	Person	Organization
M3	**0.986**	**0.807**
Demographer [79]	0.973	0.644
Humanizr [54]	0.982	0.586

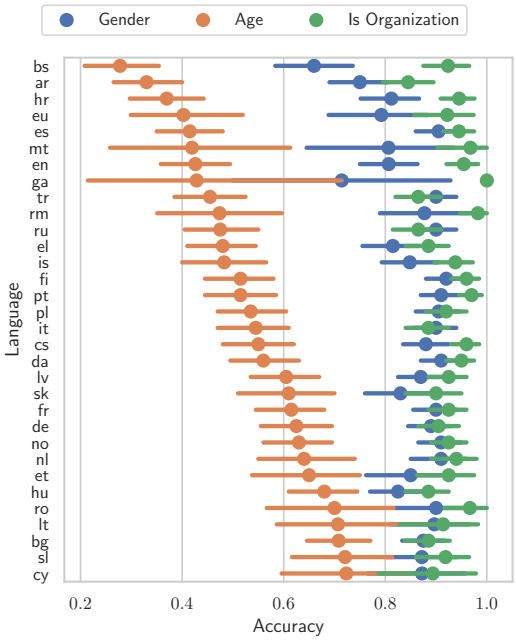


Figure 5: M3 performance on the multilingual dataset in 32 languages. Bars show 95% confidence intervals of accuracy.

2.4.6 Ablation Study. To examine which parts of the system are contributing to performance, we perform an ablation study by (i) restricting the model to one modality and (ii) training a model without using co-training or translation for data augmentation. We test each model on (i) the heuristically-labeled and organisation data and (ii) the multilingual data, which is representative of the performance on the data used in our downstream task.

The results shown in Table 4 reveal two main trends. First, the model benefits from both modalities. The removal of textual information causes the biggest performance drop. Second, although co-training and translation hurt performance in the heuristically-labeled data, they produced a substantially better model when evaluated on multilingual data. Since the heuristically-labeled data is primarily in English, translation potentially adds noise and forces the model to represent a larger space of inputs that are not represented in the test data. This performance difference indicates that the two data augmentation techniques are highly beneficial when bootstrapping a model from mostly monolingual data.

Table 4: Performance (Macro-F1) for (1) full M3 model, (2-3) model with modality dropouts, (4) model trained without co-training or translation, (5-6) Random and and Majority-class baselines.

	Heuristically-labeled			Multilingual		
M3 Model	Gender	Age	Org-Status	Gender	Age	Org-Status
Full	**0.918**	**0.522**	0.898	**0.915**	**0.425**	**0.898**
w/ Text Dropout	0.743	0.349	0.837	0.837	0.359	0.892
w/ Image Dropout	0.905	0.493	0.823	0.862	0.372	0.822
w/o Co-T. & Trans.	**0.918**	0.449	**0.917**	0.875	0.325	0.660
Random	0.494	0.181	0.500	0.494	0.211	0.466
Majority	0.377	0.214	0.500	0.334	0.187	0.210

Table 5: Performance on the Gender Shades dataset (cf. Table 4 in [12]). Gender classification performance as measured by the positive predictive value (PPV), true positive rate (TPR), and false positive rate (FPR).

Classifier	Metric	All	F	M	Darker	Lighter	DF	DM	LF	LM
MSFT	PPV	93.7	89.3	97.4	87.1	99.3	79.2	94.0	98.3	100.0
	TPR	93.7	96.5	91.7	87.1	99.3	92.1	83.7	100.0	98.7
	FPR	6.3	8.3	3.5	12.9	0.7	16.3	7.9	1.3	0.0
Face++	PPV	90.0	78.7	99.3	83.5	95.3	65.5	99.3	94.0	99.2
	TPR	90.0	98.9	85.1	83.5	95.3	98.8	76.6	98.9	92.9
	FPR	10.0	14.9	1.1	16.5	4.7	23.4	1.2	7.1	1.1
M3	PPV	96.7	99.4	94.5	94.0	99.3	99.6	89.3	99.3	99.2
	TPR	96.5	92.8	99.6	93.4	99.3	86.0	99.7	99.0	99.5
	FPR	4.2	0.4	7.2	7.7	0.8	0.3	14.0	0.5	1.0

2.4.7 Test for Algorithmic Bias. Image-based models for gender recognition are known to suffer from algorithmic bias in recognizing darker-skinned individuals [12]. This bias is thought to be a result of non-representative training data the underreprentation of darker skin tones in the training data. As our focus is on European countries, such a bias could be present in the M3 data. However, because our co-training procedure uses a large unlabeled set of users from across the globe, these users can potentially provide a more representative sample and reduce algorithmic disparity. To test for algorithmic bias, we compare the M3 model's performance on the Gender Shades dataset [12], which contains 3964 gender-annotated facial images balanced across light and dark skin tones. In this earlier algorithmic audit, the Microsoft (MSFT) and Face++ gender classifiers performed substantially worse on women, especially darker skinned women.

The gender inference performance, shown in Table 5, reveals that M3 has substantially less algorithmic bias relative to the two best-performing commercial systems also tested on Gender Shades. Our model significantly improves performance on dark-skinned women in comparison to the other two systems which had the largest performance disparity on that demographic. However, M3 is least accurate on darker-skinned males, which indicates that additional work is needed to reach performance parity in skin tones. Nevertheless, this analysis also indicates that M3 is more suitable than existing systems for operating in the global environment and increases the robustness of downstream social sensing applications.

3 LEARNING INCLUSION PROBABILITIES

In social sensing studies, the results of measurements performed on a given platform, e.g., a social media site, are studied to understand the behavior of a population. Often, such measurements are biased [67], as individuals with certain demographics are more likely to join these platforms, e.g., young people may be more likely to join Twitter. Obtaining representative estimates in these scenarios is challenging, as the probability of an individual with given demographics to be on a given platform, also referred to as *inclusion probability* in sampling methodology, is typically unknown.

Here, we estimate these inclusion probabilities as a function of demographics. To this end, we learn the debiasing coefficients on the grounds of statistical survey analysis with missing data [10, 52, 69]. Specifically, we derive estimators for the numbers of individuals by making an assumption that per-strata inclusion probabilities are equal for individuals within a group (e.g., a country) and different between groups (e.g., between countries). This assumption allows us to get both across-groups global debiasing coefficients (i.e., the biases of Twitter users) and the group-specific debiasing coefficients (i.e., per-country demographics on Twitter, e.g., developed countries may have more and diverse users on Twitter). Our approach learns the inclusion probabilities, while typical post-stratification methods either assume these probabilities are known or focus on obtaining a post-stratified estimator of a response variable.

3.1 Formulation of Debiasing Models

Consider a population U of $N = |U|$ individuals with certain demographics. For simplicity, we focus on the case of only two discrete demographic variables, say age a and gender g, which are distributed following $P_{\mathrm{N}}(a, g)$, but the following reasoning applies to other demographics as well. Out of these N individuals, say, $M \leq N$ joined a certain online platform with a probability depending on their demographics. For instance, individuals joined Twitter and younger people were more likely to join than older ones.

In survey analysis, the probability that an individual with certain demographics joined a certain platform corresponds to the inclusion probability of the stratum representing these demographics [10, 69]. In the simplest scenario, when the probability of joining the platform is homogeneous in time and across individuals, this probability can be expressed as the ratio of the number of Twitter users with the demographics a and g to all individuals, i.e., $\pi(a, g) = \frac{M(a,g)}{N(a,g)} = \frac{MP_{\mathrm{M}}(a,g)}{NP_{\mathrm{N}}(a,g)}$, where $P_{\mathrm{M}}(a, g)$ and $P_{\mathrm{N}}(a, g)$ are the distributions of demographics of Twitter users and the overall population, respectively. However, the inclusion probability may vary between individuals. To account for this, we discuss the homogeneity of inclusion probability for a given partition of the population. A partition of the population U is defined through non-overlapping and non-empty subsets U_i of the population that together constitute U, i.e., $\bigcup_i U_i = U$. Typically such subsets will have a certain meaning, e.g., a natural partition of a population is a split by countries, regions, or cities. The total number of individuals in the subset i is N^i and the number of individuals having particular demographics is $N^i(a, g)$.

3.1.1 Homogeneous bias. The inclusion probability, which governs the bias, is homogeneous with respect to a given set of demographics and a given partition of the population, if and only if $\pi^i(a, g) = \pi(a, g)$ for each subset i of the partition, i.e., the inclusion probability does not depend on the elements of the partition.

If the inclusion probability is homogeneous, then we can write

$$N^i = \sum_a \sum_g N^i(a, g) = \sum_a \sum_g \frac{M^i(a, g)}{\pi(a, g)}. \tag{1}$$

Thus, to obtain the inclusion probabilities, we regress N^i against $M^i(a, g)$. If the condition of homogeneity holds, then the regression coefficients are equal to $1/\pi(a, g)$.

3.1.2 Inhomogeneous bias. If the inclusion probability is inhomogenous for given demographics and partition, then we shall model each subset of the partition separately or use a different model that relaxes the homogenity assumption by specifying the functional form of inhomogeneity. Here, we consider the inhomogeneity of

the form

$$\pi^i(a,g) = M^i(a,g)^v f_1(a) f_2(g), \quad (2)$$

where v is an unknown exponent and f_1 and f_2 are unknown functions.[5] Under this assumption, $\log N^i(a,g) = \log \frac{M^i(a,g)}{\pi^i(a,g)} = (1-v)\log M^i(a,g) - \log f_1(a) - \log f_2(g)$. This time, to obtain the debiasing coefficients and corresponding inclusion probabilities, we regress the ground-truth population, $\log N^i(a,g)$, against the biased measurements, $\log M^i(a,g)$, and the demographic indicator variables, i.e.,

$$\log N^i(a,g) = \beta_1 \log M^i(a,g) + \sum_{\tilde{a}} \beta_{\tilde{a}} \delta_{\tilde{a}a} + \sum_{\tilde{g}} \beta_{\tilde{g}} \delta_{\tilde{g}g} \quad (3)$$

where δ_{ij} is Kronecker delta, i.e., $\delta_{ij} = 0$ if $i \neq j$ and $\delta_{ij} = 1$ if $i = j$, $\beta_1 = (1-v)$, $\beta_a = -\log f_1(a)$, and $\beta_g = -\log f_2(g)$. From these regression coefficients we can obtain the inclusion probability $\pi^i(a,g)$ of a set of samples i with the demographics a and g via Equation 2. This debiasing model has been proposed recently by Zagheni et al. to predict the number of migrants based on Facebook data [81], but its derivation and explicit formal interpretation have not been provided until now, to the best of our knowledge. Note that this estimation method of inclusion probabilities requires the ground-truth joint counts $N^i(a,g)$ at the time of training, whereas the model based on the homogeneity assumption requires only the total counts N^i. The availability of the joint counts is often limited in practice because of insufficient number of samples per stratum.

3.2 Discussion

We derived debiasing models by making homogeneity or inhomogeneity assumptions. These models predict population size, so they can be evaluated by measuring the error of predictions in cross-validation settings or via model selection methods. In this way, we test which assumption is closer to reality and find the most accurate method for the given dataset. We evaluate these models in the next section using the population of regions in EU countries.

4 EUROPEAN POPULATION INFERENCE FROM TWITTER DATA

Here, we use the debiasing models formulated in the previous section to obtain the debiasing coefficients and the corresponding inclusion probabilities for Twitter users of different countries. To this end, we regress the country-level ground-truth number of people living in a certain location against the number of Twitter users of different demographics.

4.1 Debiasing Models for Population Inference

More specifically, we evaluate five models requiring different amounts of data. The first model is a baseline, the next three models are based on the assumption of homogeneous inclusion probabilities (Equation 1), whereas the last one is based on the inhomogeneity assumption (Equation 3):

- $\mathbf{N} \sim \mathbf{M}$ is our base model that uses only the total population count from the census (N) and Twitter (M).
- $\mathbf{N} \sim \sum_{\mathbf{g}} \mathbf{M(g)}$ uses gender marginal counts only (i.e., the total counts of males and females not broken down by ages).
- $\mathbf{N} \sim \sum_{\mathbf{a}} \mathbf{M(a)}$ uses age marginal counts only.
- $\mathbf{N} \sim \sum_{\mathbf{a,g}} \mathbf{M(a,g)}$ uses the joint histograms inferred from Twitter but only the total population values from the census.
- $\mathbf{\log N(a,g) \sim \log M(a,g) + a + g}$ uses the joint histograms inferred from Twitter and the joint histograms from the census.

Note that Twitter users are biased in various ways: the platform is more accessible to tech-savvy individuals and citizens of various countries use it to a different extent [34]. To distinguish the global effect of Twitter on the overall bias from the local effect of country, we use multilevel models. Namely, all slopes and intercepts in the introduced models have random effects specific to a country. Thus, from fixed effects of the model, we obtain global debiasing coefficients (i.e., the biases of Twitter users) and, from random effects, the country-specific debiasing coefficients (i.e., a given country has its own bias towards Twitter).

Note that the homogeneity and inhomogeneity assumptions apply within each country separately. Namely, we group samples by regions of a country, described next. Then, the homogeneity assumption translates into the same inclusion probability for a given demographic across all regions of a country, whereas the inhomogeneity assumption translates into a dependence of the inclusion probability on regions of a country that follows Equation 2.

4.2 Data

We retrieved joint population distributions for age and gender on a regional level (NUTS3) for 26 countries of the European Union plus four EFTA members as made available through the CensusHub system of Eurostat, the European Statistical Office.[6] All census data is from 2011, the most recent year for which comprehensive census data was collectively reported to Eurostat. NUTS3 regions are the finest-grained level of the "Nomenclature of Territorial Units for Statistics" used as a standard for statistical reporting in EU member states [26]. They are based on existing local administrative boundaries and are usually at the level of local districts. Although sizes differ to a certain extent per country, NUTS3 provide the most standardized cross-country geographical units to subdivide populations. We use the 2010 iteration of the NUTS3 regions as these correspond with the 2011 census subdivisons and is the most recent census available.

Our sample of social media users is derived from the random 10% stream of tweets from Twitter. We recorded all users observed from September 2015 to January 2016 and downloaded profile photos for all users in spring 2018. The location of each user is inferred using the method of Compton et al. [19]. This model predicts a user's latitude and longitude and was shown to be least biased with respect to urban and rural areas [41], which is important given the diversity in population centers in our study. The model was trained on a social network of 781M edges constructed from Twitter data spanning 2012 to 2017. Five-fold cross validation reports a median inference error of 7.9km, which is sufficiently accurate for the geographic granularity we use here. Each user is assigned to a NUTS3 region in Europe if the inferred latitude-longitude pair of

[5] Note that the aforementioned homogeneity assumption is a special case of this assumption, i.e., this assumption relaxes the homogeneity assumption.

[6] https://ec.europa.eu/CensusHub2/: Luxembourg and Belgium were not available. Additional EFTA countries: Switzerland, Iceland, Liechtenstein and Norway.

that user lies within the boundaries of the NUTS3 regions; users not in these regions are discarded. Ultimately, we obtain a dataset of 3,202,964 users within the NUTS3 regions for our study. In the remainder, we will refer to NUTS3 regions simply as "regions."

The age, gender, and is-organization variables for each user in our dataset are inferred using M3. As an additional experiment to quantify the impact of including non-human accounts, we ignore any organization classification and treat all accounts as humans, grouping them according to their inferred demographics.

4.3 Results

We evaluate the debiasing models in the following cross-validation settings: leave one region out, leave one country out (i.e., leave out all regions from a given country), and leave one stratum out (e.g., leave out only males aged 30-39). Per case, we measure the mean absolute percentage error of the population estimates for the left-out samples, i.e., $\text{MAPE}(N) = \frac{100\%}{n} \sum_i \frac{|\hat{N}_i - N_i|}{|N_i|}$, where $\hat{N}_i$ and N_i are the predicted and actual population sizes, respectively, and the sum is over all regions. Note that the model $\log N(a,g) \sim \log M(a,g) + a + g$ operates naturally in the log space; hence, before calculating the error of this model we first exponentiate the predicted population sizes to make the results more comparable across all models. Its results should further be compared to the remaining models with care, as the effects of moving to log space cannot be clearly untangled from gains due to learning on joint age and gender attributes.

The results of the leave-one-region-out evaluation show a clear benefit to debiasing based on the inferred demographics (Figure 6). For the $N \sim M$ model without demographics, MAPE is 88%. The inclusion of the inferred gender or age demographics in the debiasing models, via $N \sim \sum M(a)$ and $N \sim \sum M(g)$, decreases MAPE to 59% and 61%, respectively. Including inferred joint distributions, e.g., the number of males of age 30-39, in the model $N \sim \sum M(a,g)$, decreases MAPE further to 54%, even though we do not use joint-distribution data from the census for training. In fact, the use of census joint distributions at the stage of model training along with a move to log space, via $\log N(a,g) \sim \log M(a,g) + a + g$, further improves the accuracy, bringing the MAPE down to 33%. To understand the effect of accounts belonging to organizations on this population prediction task, we compare the results of the debiasing models with and without removing the organizations (Figure 6). Removing organizations results in a small reduction in error for all but the model trained with only marginal gender counts, though these differences are not statistically significant. A further analysis shows that the presence of organizations is heavily skewed towards populous metropolitan cities and most regions have very few organizations (i.e., organizations are not equally geographically distributed). However, for populous regions, organizations can create significant error.

To gain further insights into the introduced models, we show scatter plots of true and predicted population sizes for each model (Figure 7). The model with joint Twitter and census counts is noticeably closer to the $y = x$ line, likely because this models is trained in log space. For this model, we also depict the geographical variation of MAPE across the EU regions (Figure 8).

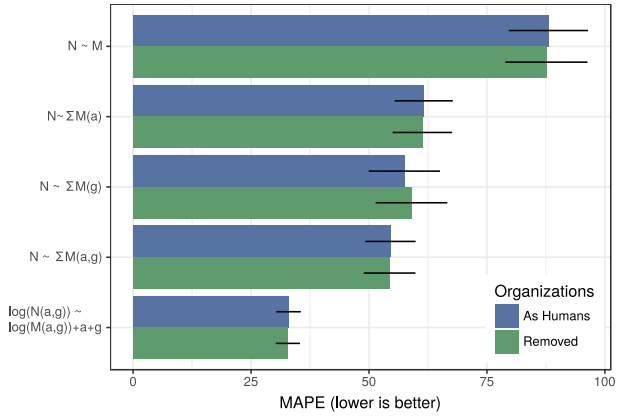


Figure 6: Performance on leave-one-region-out population inference across different debiasing models. Bars show mean MAPE(N) with 95% confidence intervals. The results of the last model are exponentiated to compute MAPE(N).

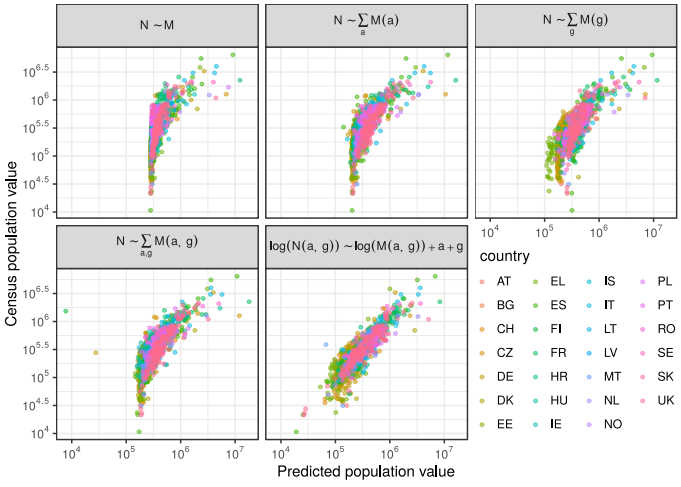


Figure 7: Comparing the census population and debiased estimates for leave-one-region-out evaluations shows that the model with joint Twitter and census distributions correlates best with the real values for large and small regions.

The above results test predictions when one region is left out and its population is predicted. However, in different circumstances, more or less census information could be available. We test two other cross-validation settings: leave one stratum out (e.g., leave out only females aged 30–39) and leave one country out (i.e., leave out all regions from a given country). The latter reflects the generalizability of the model to completely unseen countries where platform adoption probabilities and country-specific biases are not known. The evaluation results for all three cross-validation settings are plotted in Figure 9 for the most accurate model, i.e., $\log N(a,g) \sim \log M(a,g) + a + g$. From the first additional cross-validation setting, we learn that hiding the population sizes for a specific stratum in all regions results in a minimal penalty to the prediction accuracy. For the second additional cross-validation setting, we see that the error rate nearly doubles with an average MAPE of 81%. This result suggests that knowledge of country-specific

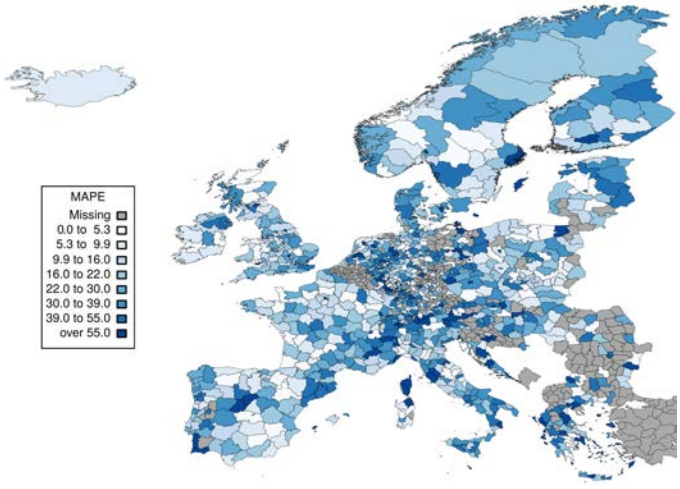


Figure 8: MAPE for $\log N(a,g) \sim \log M(a,g)+a+g$ model across regions shows that many regions have population estimate errors below 10% of the true population, with high-error regions found infrequently in all countries. For greyed regions we either miss NUT3 data or Twitter counts are zero.

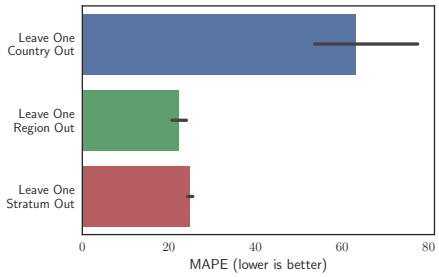


Figure 9: Comparisons of the model with joint Twitter and census distributions as increasing amounts of information are held out from training: (i) leaving out one stratum, (ii) leaving out all the strata from one region, and (iii) leaving out all regions in one country.

platform biases is important for accurate estimates and that at least some regions within the country should be seen during a model's training time to reach high performance.

4.4 Discussion of Debiasing Results and Potential Sources of Error

Debiasing social media data samples is a difficult task, but our results show that automatic post-stratification with respect to inferred age and gender notably improves population estimation results. Furthermore, our predictive models are fully interpretable, which allows us to estimate the inclusion probabilities and share them with the research community for future reuse.

Even when joint distributions are not given by the census, inferring joint distributions from social media data with the model $N \sim M(a,g)$ provides a significant increase in accuracy in population prediction tasks compared to the baseline without debiasing.

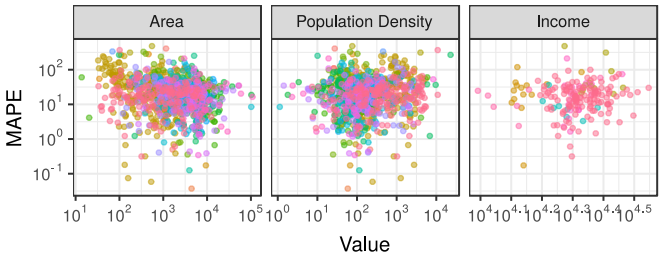


Figure 10: Comparing the model with joint Twitter and census distributions MAPE to region area (km^2), population density (people/km^2), and income (USD per capita) shows that the model performance is not biased towards particular types of regions. Colors represent countries as in Figure 7.

However, the model $\log N(a,g) \sim \log M(a,g) + a + g$ is notably more accurate, suggesting that the assumption of homogeneity does not hold for the partition of citizens of a particular country, gender, and age into regions. Note, however, that the predictions of the most accurate model are not perfect. This may be caused by the time mismatch of about 5 years between the Twitter and NUTS3 datasets. On the other hand, this point opens the door for developing debiasing models based on other inhomogeneity assumptions and searching better partitions, e.g., our multilevel model could have more levels to capture biases shared by smaller regions within a country.

To learn more about potential confounders, we compare the MAPE of the model $\log N(a,g) \sim \log M(a,g) + a + g$ to three variables: the area of the region, its population density, and average income (Figure 10). NUTS3 regions are defined by each country and vary considerably in land area from city-states in Germany to large regions in Sweden (up to 98,000 km^2). We do not find correlations of land area with the MAPE error. Hecht and Stephens [35] found a clear urban–rural bias in social media data: there tend to be more users, more information, and higher quality information per capita within metropolitan areas. Despite this bias, we find our estimates not heavily correlated with population density, suggesting our models do well at debiasing these differences in inclusion probabilities. Only two countries (Czech Republic and Norway) show a correlation that is statically different from zero across our models. Finally, income has been suggested as an important explanatory variable for who uses Internet-based platforms and hence could be a potential confound to our estimates [14, 40]. However, we similarly find that income and MAPE are not correlated significantly in our models.

5 CONCLUSION

The everyday opinions expressed in social media provide promising opportunities for measuring population-level statistics for health metrics, political outcomes, or general attitudes. However, social media is a non-representative sample of the population due to demographic skew in usage frequencies and access rates. As such, any direct estimate from a platform is likely biased towards certain demographics. This work provides a holistic solution to this problem by developing a novel method for assigning users to demographic strata and exploiting it in a regression framework for debiasing that allows direct estimation of the probability of an individual

with given demographics to be on the given social media platform. Our work provides three main contributions. First, we introduce a state-of-the-art neural system for multi-attribute classification in 32 languages. This contribution also includes the system release and the creation of a new dataset of gender, age, and is-organization annotations in 32 languages. Second, we derive a series of models to debias social media measurements and provide their explicit formal interpretations. Third, in a massive study of all of Europe, we show that our two methods are able to infer regional population counts accurately and provide demographic corrections for all downstream measurements on the grounds of the estimated inclusion probabilities. These results pave the way for more accurate social sensing by laying a foundation of representative population sampling in social media. Code, software, and debiasing coefficients pertaining to this work are released for public use at https://github.com/euagendas/.

ACKNOWLEDGEMENTS

This research has received funding through the Volkswagen Foundation.

REFERENCES

[1] Faiyaz Al Zamal, Wendy Liu, and Derek Ruths. 2012. Homophily and Latent Attribute Inference: Inferring Latent Attributes of Twitter Users from Neighbors. In *Proceedings of ICWSM*.

[2] Sultan Alzahrani, Chinmay Gore, Amin Salehi, and Hasan Davulcu. 2018. Finding Organizational Accounts Based on Structural and Behavioral Factors on Twitter. In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*. Springer, 164–175.

[3] Jisun An and Ingmar Weber. 2015. Whom should we sense in "social sensing"—Analyzing which users work best for social media now-casting. *EPJ Data Science* 4, 1 (30 Nov 2015), 22. https://doi.org/10.1140/epjds/s13688-015-0058-9

[4] Ehsan Mohammady Ardehaly and Aron Culotta. 2017. Co-training for Demographic Classification Using Deep Learning from Label Proportions. In *Data Mining Workshops (ICDMW), 2017 IEEE International Conference on*. IEEE, 1017–1024.

[5] Microsoft Azure. 2018. Cognitive Services. https://azure.microsoft.com/en-us/services/cognitive-services/

[6] David Bamman, Jacob Eisenstein, and Tyler Schnoebelen. 2014. Gender identity and lexical variation in social media. *Journal of Sociolinguistics* 18, 2 (2014), 135–160.

[7] Daniele Barchiesi, Helen Susannah Moat, Christian Alis, Steven Bishop, and Tobias Preis. 2015. Quantifying International Travel Flows Using Flickr. *PLOS ONE* 10, 7 (07 2015), 1–8. https://doi.org/10.1371/journal.pone.0128470

[8] Charley Beller, Rebecca Knowles, Craig Harman, Shane Bergsma, Margaret Mitchell, and Benjamin Van Durme. 2014. I'm a belieber: Social roles via self-identification and conceptual attributes. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 181–186.

[9] Shane Bergsma and Benjamin Van Durme. 2013. Using Conceptual Class Attributes to Characterize Social Media Users. In *Proc. ACL*.

[10] Jelke G Bethlehem and Wouter J Keller. 1987. Linear weighting of sample survey data. *Journal of Official Statistics* 3, 2 (1987), 141–153.

[11] Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*. ACM, 92–100.

[12] Joy Buolamwini and Timnit Gebru. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on Fairness, Accountability and Transparency*. 77–91.

[13] Xin Chen, Yu Wang, Eugene Agichtein, and Fusheng Wang. 2015. A Comparative Study of Demographic Attribute Inference in Twitter.. In *Proceedings of ICWSM*, Vol. 15. 590–593.

[14] Menzie D Chinn and Robert W Fairlie. 2007. The determinants of the global digital divide: a cross-country analysis of computer and internet penetration. *Oxford Economic Papers* 59, 1 (2007), 16–44.

[15] Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A character-level decoder without explicit segmentation for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 1693–1703.

[16] Morgane Ciot, Morgan Sonderegger, and Derek Ruths. 2013. Gender inference of Twitter users in non-English contexts. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1136–1145.

[17] Jennifer Coates. 1998. *Language and gender: A reader*. Wiley-Blackwell.

[18] Jennifer Coates. 2015. *Women, men and language: A sociolinguistic account of gender differences in language*. Routledge.

[19] Ryan Compton, David Jurgens, and David Allen. 2014. Geotagging One Hundred Million Twitter Accounts with Total Variation Minimization. In *IEEE Conference on BigData*.

[20] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word Translation Without Parallel Data. *arXiv preprint arXiv:1710.04087* (2017).

[21] Brian de Silva and Ryan Compton. 2014. Prediction of Foreign Box Office Revenues Based on Wikipedia Page Activity. *CoRR* abs/1405.5924 (2014). arXiv:1405.5924 http://arxiv.org/abs/1405.5924

[22] Victoria Pruin DeFrancisco, Catherine Helen Palczewski, and Danielle D McGeough. 2013. *Gender in communication: A critical introduction*. Sage Publications.

[23] Hedwige Dehon and Serge Brédart. 2001. An 'other-race' effect in age estimation from faces. *Perception* 30, 9 (2001), 1107–1113.

[24] Penelope Eckert. 2008. Variation and the indexical field. *Journal of sociolinguistics* 12, 4 (2008), 453–476.

[25] Penelope Eckert and Sally McConnell-Ginet. 2003. *Language and gender*. Cambridge University Press.

[26] European Commission. [n. d.]. NUTS—Nomenclature of Territorial Units for Statistics. https://ec.europa.eu/eurostat/web/nuts/background.

[27] Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. 2016. Morphological inflection generation using character sequence to sequence learning. In *Proceedings of EMNLP*.

[28] Daniel Gayo-Avello. 2012. " I Wanted to Predict Elections with Twitter and all I got was this Lousy Paper"–A Balanced Survey on Election Prediction using Twitter Data. *arXiv preprint arXiv:1204.6441* (2012).

[29] Daniel Gayo-Avello. 2013. A meta-analysis of state-of-the-art electoral prediction from Twitter data. *Social Science Computer Review* 31, 6 (2013), 649–679.

[30] Nicholas Generous, Geoffrey Fairchild, Alina Deshpande, Sara Y. Del Valle, and Reid Priedhorsky. 2014. Global Disease Monitoring and Forecasting with Wikipedia. *PLOS Computational Biology* 10, 11 (11 2014), 1–16. https://doi.org/10.1371/journal.pcbi.1003892

[31] Jeremy Ginsberg, Matthew H. Mohebbi, Rajan S. Patel, Lynnette Brammer, Mark S. Smolinski, and Larry Brilliant. 2008. Detecting influenza epidemics using search engine query data. *Nature* 457 (nov 2008), 1012. https://doi.org/10.1038/nature0763410.1038/nature07634

[32] Rob Goot, Nikola Ljubešić, Ian Matroos, Malvina Nissim, and Barbara Plank. 2018. Bleaching Text: Abstract Features for Cross-lingual Gender Prediction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 383–389.

[33] Sumit Goswami, Sudeshna Sarkar, and Mayur Rustagi. 2009. Stylometric analysis of bloggersâĂŹ age and gender. In *Proceedings of ICWSM*.

[34] Scott A. Hale. 2014. Global Connectivity and Multilinguals in the Twitter Network. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 833–842. https://doi.org/10.1145/2556288.2557203

[35] Brent Hecht and Monica Stephens. 2014. A Tale of Cities: Urban Biases in Volunteered Geographic Information. In *Proceedings of ICWSM*. https://www.aaai.org/ocs/index.php/ICWSM/ICWSM14/paper/view/8114

[36] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[37] D. Holt and T. M. F. Smith. 1979. Post Stratification. *Journal of the Royal Statistical Society. Series A (General)* 142, 1 (1979), 33–46. http://www.jstor.org/stable/2344652

[38] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely Connected Convolutional Networks.. In *CVPR*, Vol. 1. 3.

[39] Aaron Jaech and Mari Ostendorf. 2015. What your username says about you. *arXiv preprint arXiv:1507.02045* (2015).

[40] Bernard J Jansen. 2010. *Use of the internet in higher-income households*. Pew Research Center Washington, DC.

[41] Isaac Johnson, Connor McMahon, Johannes Schöning, and Brent Hecht. 2017. The Effect of Population and Structural Biases on Social Media-based Algorithms: A Case Study in Geolocation Inference Across the Urban-Rural Spectrum. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 1167–1178.

[42] Soon-Gyo Jung, Jisun An, Haewoon Kwak, Joni Salminen, and Bernard J Jansen. 2017. Inferring Social Media Users' Demographics from Profile Pictures: A Face++ Analysis on Twitter Users. In *Proceedings of The 17th International Conference on Electronic Business*. 140–145).

[43] Andreas Jungherr. 2017. Normalizing digital trace data. *Digital Discussions: How Big Data Informs Political Communication* (2017).

[44] Andreas Jungherr, Pascal Jürgens, and Harald Schoen. 2012. Why the pirate party won the german election of 2009 or the trouble with predictions: Tumasjan, A., Sprenger, T. O., Sander, P. G., & Welpe, I. M. "Predicting elections with twitter:

What 140 characters reveal about political sentiment". *Social science computer review* 30, 2 (2012), 229–234.
[45] Shari Kendall, Deborah Tannen, et al. 1997. Gender and language in the workplace. *Gender and Discourse. London: Sage* (1997), 81–105.
[46] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-Aware Neural Language Models.. In *AAAI*. 2741–2749.
[47] Rebecca Knowles, Josh Carroll, and Mark Dredze. 2016. Demographer: Extremely simple name demographics. In *Proceedings of the First Workshop on NLP and Computational Social Science*. 108–113.
[48] Klaus Krippendorff. 2011. Computing Krippendorff's alpha-reliability. *University of Pennsylvania Departmental papers (ASC)* (2011).
[49] Robin Tolmach Lakoff and Mary Bucholtz. 2004. *Language and woman's place: Text and commentaries*. Vol. 3. Oxford University Press, USA.
[50] Fabio Lamanna, Maxime Lenormand, María Henar Salas-Olmedo, Gustavo Romanillos, Bruno Gonçalves, and José J Ramasco. 2018. Immigrant community integration in world cities. *PloS one* 13, 3 (2018), e0191612.
[51] David Lazer, Ryan Kennedy, Gary King, and Alessandro Vespignani. 2014. The Parable of Google Flu: Traps in Big Data Analysis. *Science* 343, 6167 (2014), 1203–1205.
[52] Roderick Little and Donald Rubin. 2002. *Statistical analysis with missing data, Second edition*. 408 pages.
[53] Michael McCandless. 2010. Accuracy and performance of Google's compact language detector. http://blog.mikemccandless.com/2011/10/accuracy-and-performance-of-googles.html.
[54] James McCorriston, David Jurgens, and Derek Ruths. 2015. Organizations Are Users Too: Characterizing and Detecting the Presence of Organizations on Twitter.. In *Proceedings of ICWSM*. 650–653.
[55] Márton Mestyán, Taha Yasseri, and JÃąnos Kertész. 2013. Early Prediction of Movie Box Office Success Based on Wikipedia Activity Big Data. *PLOS ONE* 8, 8 (08 2013), 1–8. https://doi.org/10.1371/journal.pone.0071226
[56] Alan Mislove, Sune Lehmann, Yong-Yeol Ahn, Jukka-Pekka Onnela, and J Niels Rosenquist. 2011. Understanding the Demographics of Twitter Users. *Proceedings of ICWSM* 11, 5th (2011), 25.
[57] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 807–814.
[58] Dong Nguyen, Rilana Gravel, Dolf Trieschnigg, and Theo Meder. 2013. "How Old Do You Think I Am?" A Study of Language and Age in Twitter.. In *Proceedings of ICWSM*.
[59] Dong Nguyen, Noah A Smith, and Carolyn P Rosé. 2011. Author age prediction from text using linear regression. In *Proc. of the Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*. Association for Computational Linguistics, 115–123.
[60] David K Park, Andrew Gelman, and Joseph Bafumi. 2004. Bayesian multilevel estimation with poststratification: state-level estimates from national polls. *Political Analysis* 12, 4 (2004), 375–385.
[61] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS-W*.
[62] Francisco Rangel, Paolo Rosso, Manuel Montes-y Gómez, Martin Potthast, and Benno Stein. 2018. Overview of the 6th author profiling task at PAN 2018: Multimodal gender identification in Twitter. *Working Notes Papers of the CLEF* (2018).
[63] Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. 2010. Classifying latent user attributes in twitter. In *Proc. of the 2nd International Workshop on Search and Mining User-generated Contents*. ACM, 37–44.
[64] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. 2018. On the convergence of adam and beyond. (2018).
[65] Sara Rosenthal and Kathleen McKeown. 2011. Age prediction in blogs: A study of style, content, and online behavior in pre-and post-social media generations. In *Proc. ACL*. Association for Computational Linguistics, 763–772.
[66] Rasmus Rothe, Radu Timofte, and Luc Van Gool. 2016. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision (IJCV)* (July 2016).
[67] Derek Ruths and Jürgen Pfeffer. 2014. Social media for large studies of behavior. *Science* 346, 6213 (2014), 1063–1064.
[68] Maarten Sap, Gregory Park, Johannes Eichstaedt, Margaret Kern, David Stillwell, Michal Kosinski, Lyle Ungar, and H. Andrew Schwartz. 2014. Developing Age and Gender Predictive Lexica over Social Media. In *Proc. EMNLP*. Association for Computational Linguistics, 1146–1151.
[69] Carl-Erik Särndal and Sixten Lundström. 2005. *Estimation in Surveys with Nonresponse*. John Wiley & Sons, Ltd, Chichester, UK. 1–199 pages. https://doi.org/10.1002/0470011351
[70] Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. 2006. Effects of age and gender on blogging.. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, Vol. 6. 199–205.
[71] Luke Sloan. 2017. Who Tweets in the United Kingdom? Profiling the Twitter Population Using the British Social Attitudes Survey 2015. *Social Media + Society* 3, 1 (2017), 2056305117698981.
[72] Deborah Tannen. 1991. *You just don't understand: Women and men in conversation*. Virago London.
[73] Deborah Tannen. 1993. *Gender and conversational interaction*. Oxford University Press.
[74] UK Office for National Statistics 2015. *Harmonised Concepts and Questions for Social Data Sources - Primary Principles*. UK Office for National Statistics. http://www.ons.gov.uk/ons/guide-method/harmonisation/primary-set-of-harmonised-concepts-and-questions/demographic-information--household-composition-and-relationships.pdf
[75] United States Department of Education 2009. *Implementation guidelines: Measures and methods for the national reporting system for adult education*. United States Department of Education. http://www.air.org/sites/default/files/downloads/report/ImplementationGuidelines_0.pdf
[76] Nicola Van Rijsbergen, Katarzyna Jaworska, Guillaume A Rousselet, and Philippe G Schyns. 2014. With age comes representational wisdom in social signals. *Current Biology* 24, 23 (2014), 2792–2796.
[77] Wei Wang, David Rothschild, Sharad Goel, and Andrew Gelman. 2015. Forecasting elections with non-representative polls. *International Journal of Forecasting* 31, 3 (2015), 980–991.
[78] Zijian Wang and David Jurgens. 2018. It's going to be okay: Measuring Access to Support in Online Communities. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 33–45.
[79] Zach Wood-Doughty, Praateek Mahajan, and Mark Dredze. 2018. Johns Hopkins or johnny-hopkins: Classifying Individuals versus Organizations on Twitter. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*. 56–61.
[80] Hua Wu and Haifeng Wang. 2007. Pivot language approach for phrase-based statistical machine translation. *Machine Translation* 21, 3 (2007), 165–181.
[81] Emilio Zagheni, Ingmar Weber, and Krishna Gummadi. 2017. Leveraging Facebook's Advertising Platform to Monitor Stocks of Migrants. *Population and Development Review* 43, 4 (2017), 721–734.
[82] Jinxue Zhang, Xia Hu, Yanchao Zhang, and Huan Liu. 2016. Your Age Is No Secret: Inferring Microbloggers' Ages via Content and Interaction Analysis.. In *Proceedings of ICWSM*. 476–485.

Iterative Discriminant Tensor Factorization for Behavior Comparison in Massive Open Online Courses

Xidao Wen
University of Pittsburgh
Pittsburgh, PA
xidao.wen@pitt.edu

Yu-Ru Lin*
University of Pittsburgh
Pittsburgh, PA
yurulin@pitt.edu

Xi Liu
Texas A&M University
College Station, TX
xiliu.tamu@gmail.com

Peter Brusilovsky
University of Pittsburgh
Pittsburgh, PA
peterb@pitt.edu

Jordan Barría Pineda
University of Pittsburgh
Pittsburgh, PA
jab464@pitt.edu

ABSTRACT

The increasing utilization of massive open online courses has significantly expanded global access to formal education. Despite the technology's promising future, student interaction on MOOCs is still a relatively under-explored and poorly understood topic. This work proposes a multi-level pattern discovery through hierarchical discriminative tensor factorization. We formulate the problem as a hierarchical discriminant subspace learning problem, where the goal is to discover the shared and discriminative patterns with a hierarchical structure. The discovered patterns enable a more effective exploration of the contrasting behaviors of two performance groups. We conduct extensive experiments on several real-world MOOC datasets to demonstrate the effectiveness of our proposed approach. Our study advances the current predictive modeling in MOOCs by providing more interpretable behavioral patterns and linking their relationships with the performance outcome.

ACM Reference Format:
Xidao Wen, Yu-Ru Lin, Xi Liu, Peter Brusilovsky, and Jordan Barría Pineda. 2019. Iterative Discriminant Tensor Factorization for Behavior Comparison in Massive Open Online Courses. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313713

1 INTRODUCTION

While massive open online courses (MOOCs) have been attracting an ever-increasing number of students, the low completion rate (between 5%-10% [19]) has been a major obstacle to the transformative potentials of MOOCs. The predictive analysis of student performance thus emerged an an important research topic offering insights to platform developers and instructors in arranging proper learning support and allocating resources to students. To find informative predictors, researchers have focused on extracting features from students' interaction with the MOOC platform, such as watching videos, working on assignments, and viewing or contributing to discussion forums. Applied predictive models range from standard machine learning methods [23, 28] to more advanced ones such as deep learning [7]. These prediction models could be useful in predicting learning outcomes but are notably limited in helping understand the underlying learning behavior.

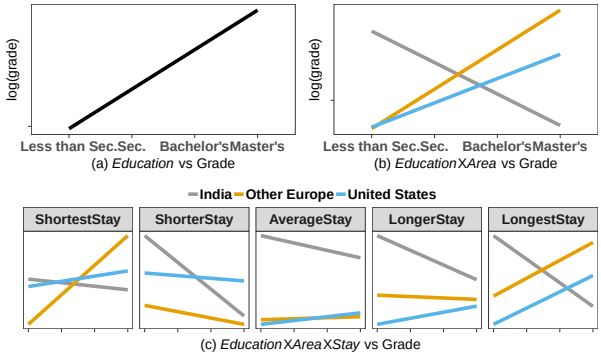


Figure 1: Association analysis of student performance on MOOCs. (a) shows the positive association between the grade and education level; (b) shows the mixed associations when including the area where the student is from; (c) further breaks down the observed groups into different level of activity (five quantiles w.r.t. the number of days students remained active on the platform).

There is abundant work that aims to better understand the behavior patterns that relate to the learning outcomes. For example, Coleman et al. [4] correlates each "behavior topic" to the learning outcomes based on topic modeling. However, the research to date fails to consider the multi-dimensional nature of the features and their potential interactions in outcome learning. Meanwhile, the famous Simpson's paradox points out that the direction of an association at the population-level may be reversed within the subgroups comprising that population [20]. To further explain this in the context of the MOOC platform, we use the Edx MOOC dataset [14] and investigate the factors associated with the students' grade on it.

We extract the education, area (the region where the student is from), the number of active days and the final grade of each student in the course "Introduction to Computer Science and Programming" offered by MITx in Spring 2013. The course had more than 44,000 online participants. Figure 1 shows the association between the selected factors and the final grades of the students. By comparing

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313713

Figure 1(a) and Figure 1(b), we observe that when the factor of area is considered, mixed associations occur. For example, the subgroup of Indian participants exhibited a negative association between education level and grade, which potentially suggests a more conservative understanding of the relationship between educational background and course outcome. Moreover, when we consider the number of active days, as in Figure 1(c), we notice that for the participants from the "other European" area, the positive association has a strong presence— but only with *ShortestStay* and *LongestStay*. Thus, in comparison to typical correlation analysis, a prediction model that can take advantage of the multi-way interactions of the features could potentially yield better performance.

A growing body of research seeks to resolve Simpson's paradox through causality inference [34]. However, causality is not the focus of this study as we aim for a data-driven approach to subgroup comparisons and explorations. In many cases, this can be interesting and important even in non-causal settings. A straightforward solution is to perform regression with feature interactions, or use Factorization Machines [37] that allow for the estimation of high-order interaction effects. However, the drawback of these methods is that they offer little understanding of the underlying multi-way learning behavior dynamics and their relationship to the learning outcomes. On the other hand, Factorization models like Matrix Factorization (2-way) and Tensor Factorization (m-way) are able to provide an in-depth understanding of meaningful behavior dynamics [9], but there are a few drawbacks preventing them from being more widely adopted by researchers in the field. First, the associations are isolated, with each of them capturing a certain trend of the behaviors separately (e.g., Figure 2(c)). Second, conventional pattern discovery through factorization models provides little support for contrasting pattern exploration that aims to identify the shared and discriminative behavior characteristics among different groups of users. Being able to do this can tremendously improve knowledge of user behaviors in the context of user group analysis.

In this work, we formulate the problem of understanding learning behavior in MOOCs as (1) the simultaneous factorization of the association between students' multi-aspect features and their performance, and (2) the iterative discovery of interpretable shared and discriminative patterns at multiple levels. The critical challenge is how to utilize the multi-way interaction of the features while providing interpretable patterns to help domain experts understand the learning dynamics. We propose a tensor-based learning method—iterative Discriminative tensor factorization (`iDisc`)—that discovers the common and discriminative learning patterns at multiple levels, and based on which we project users to a latent space (i.e. *embedding* for the downstream prediction tasks) to identify the association between the multi-way interaction of the features and the students' performance. To this end, we first represent the behaviors of the students from the opposite performance groups as *coupled tensors*. Since the coarse-grained joint factorization of these behavior tensors may not be capable of revealing behavior patterns at the subgroup level, `iDisc` iteratively performs discriminative pattern discovery at multiple levels. To increase the interpretability of the entire pattern space, we also introduce the inference of pattern hierarchy. To make the solution capable of handling unseen students, we project the students' behavior tensors into a latent space, by considering the multi-way interactions at different levels as the loading matrix. The empirical studies with the dataset from different MOOC platforms have shown the promising results on the effectiveness and efficiency of `iDisc`.

Contributions Our contributions can be summarized as follows:

- We formulate the problem of identifying the multi-way feature interaction with interpretable pattern discovery for understanding user behavior on the MOOC platforms.
- We propose a framework of iterative discriminant factorization for multi-way data. By factorizing the residual tensors at each level, our method enables the discovery of common and discriminative patterns at different granular levels. To ensure the parsimony of the discovered structure, we employ sparse learning to effectively capture enforcing relationships between the top-level and bottom-level patterns.
- We perform extensive experimentation of our methodology using several real-world datasets, and show the efficiency and interpretability of our proposed method.

2 RELATED WORK

Predictive Modeling in MOOCs. There are several types of predictive models in MOOCs that are closely related to this work. One direction is to utilize more complex feature types, including higher-order n-gram representations of learner activity data. For instance, features are constructed using the occurrence of pre-defined sequential activities [44], or from sequential pattern mining [11, 17, 26, 40]. Another line of work proposes to utilize the temporal nature of the activity data for student success prediction. Qiu et al. [36] propose a latent dynamic factor graph (LadFG) to model and predict learning behavior in MOOCs. LadFG captures the dynamic information and homophily correlations between students. It also projects students' learning behavior into a latent continuous space for predicting student performance. Another approach is the latent variable modeling as a way of inferring complex relationships between predictors [13, 25, 27]. For instance, Halawa et al. [13] explore the use of count-based learning activity features to predict dropout; this approach suggests that both observable learner activity and dropout are driven by latent, unobservable "persistence" factors.

Common and Discriminative Subspace Learning. The increasing availability of data from diverse sources has enabled the study on joint analysis of heterogeneous data. Gupta et al. [12] propose a joint Non-negative Matrix Factorization (NMF) approach to separate the common and discriminative subspace. Following the same idea, Kim et al. [22] relax the framework by requiring the shared subspace to be *similar* rather than strictly identical. Wen et al. [46, 47] propose a data-driven method of jointly factorizing the paired tensors with auxiliary tensors that preserve the common and distinctive signals. However, existing works focus primarily on pattern discoveries. This limits their use in downstream tasks (e.g., prediction or classification).

Multi-Level Tensor Factorization. Multi-Level Tensor Factorization addresses the problem of approximating the hierarchical low-rank tensor format. This process allows the representation of the tensors in a nested subspace, in one of Tree-Tucker format [30], tensor train format [29], or tensor networks format [3]. Huang et al. [18] employ a tree-guided learning via tensor decomposition

Table 1: Description of Notations.

Symbol	Meaning
R	the rank of tensor decomposition
l	the level index
R_l	the rank at l-th level
$\mathbf{v}^l$	the tensor-based user embedding at l-th level
$\mathbf{P}^l$	the projection matrix that maps from the l-th level patterns to the patterns at the $(l-1)$-th level
$\mathcal{X}$, $\mathbf{X}$, $\mathbf{x}$, x	Tensor, matrix, vector, scalar
$\mathbf{X}_{(i,j)}$	the scalar at the $\{i, j\}$ position of matrix $\mathbf{X}$
$\mathbf{X}_{(r,:)}$	the r-th row of matrix $\mathbf{X}$
$\mathbf{X}_{(m)}$	the mode-m unfolding of tensor $\mathcal{X}$
$\mathbb{U}$	the set of factor matrices
$\mathbf{U}_c^{(m)}$	the m-th mode factor matrix from the tensor of class c
$\mathbf{U}_c^l$	the l-th level factor matrix from the tensor of class c
I_1,..., I_M	the dimensionality of mode 1, ..., M

and matrix factorization in the context of experts recommendation in multiple areas simultaneously. However, there is limited research that discovers hierarchical nested subspace in the tensor subspace [31]. Özdemir et al. [31] construct a data-dependent multi-scale subspace to better represent the data. To do so, the authors first construct a tree structure by partitioning the tensor into a collection of permuted sub-tensors, and then construct the multi-scale subspace by applying HoSVD to each sub-tensor.

Summary. The existing predictive analytics on MOOCs considers various ways of constructing a matrix-based feature space. We argue that tensors could be a more suitable representation for student behavioral modeling due to their flexibility in representing the multi-way interaction of the behavioral data. In this regard, Sahebi et al. [39] have shown success in using a tensor-based approach to model the students' learning process, and predict student performance. However, the multi-way interactions as behavior patterns have not been discussed, and a more interpretable pattern discovery that can support a comprehensive understanding of student behaviors is missing. On the other hand, most hierarchical tensor factorization methods tend to recursively decompose the tensor modes by a pre-specified dimension tree [3, 29]. Our work, instead, is closer to the multi-level tensor factorization approach by [32], which recursively factorizes the residual tensors to obtain a multi-level representation of the subspace. However, to the best of our knowledge, there has been no work yet that discovers the common and discriminative patterns at multiple levels, especially in the area of predictive modeling in educational data mining.

3 PROBLEM FORMULATION

In this section, we start with a brief introduction to tensor notions and operations and then formulate the problem considered in this study. Table 1 summarizes the notations used in this paper.

3.1 Preliminaries

3.1.1 Tensors. A tensor is a multidimensional array. Let x denote a scalar, $\mathbf{x}$ a vector, $\mathbf{X}$ a matrix, and then $\mathcal{X}$ is the extension of these concepts to higher dimensions. The *order* of the tensor is the number of modes (or ways) in the tensor.

3.1.2 Tensor Operations. The basic tensor operations that we use in this study are:

Mode-n unfolding, or matricization, is the process of re-ordering elements of tensor $\mathcal{X}$ to form a matrix $\mathbf{X}_{(n)}$, where n is the dimension index upon which the unfolding performs. In the case of the three-way tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, its matricization in the first mode can be denoted as $\mathbf{X}_{(1)} \in \mathbb{R}^{I_1 \times (I_2 \times I_3)}$.

CP Decomposition [15] or CANDECOMP/PARAFAC decomposition expresses an m-way tensor $\mathcal{X}$ of size $I_1 \times \cdots \times I_M$ as $\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \circ \cdots \circ \mathbf{u}_r^{(M)}$, where $\mathbf{u}_r^m \in \mathbb{R}^{I_m}$ for $m = 1, \ldots, M$ and $r = 1, \ldots, \mathbf{R}$. By using "kruskal operator," [24], we can shorthand the CP decomposition as $\mathcal{X} \approx \left[\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(M)}\right]$, where the factor matrix $\mathbf{U}^m$ is defined as $\mathbf{U}^m = \left[\mathbf{u}_1^{(m)} \cdots \mathbf{u}_R^{(m)}\right]$.

3.2 Problem Formulation with Tensors

To motivate the problem in the context of a real-world dataset, we discuss the application of NMF, Non-negative Tensor Factorization (NTF), discriminative NTF, and hierarchical NTF with a toy dataset. This dataset was extracted from one of the most popular courses in the XueTangX dataset (full dataset details in Section 5.1). Each student event, or *activity*, is associated with three attributes: *time* (d_1, d_2), *source* (s_1, s_2), and *type* ($t_1, t_2, \ldots, t_7$).

3.2.1 NMF. Let matrix $\mathbf{X}$ denote the aggregated activities that users have been recorded engaging in on the XueTangX platform for this course. The nature of matrix $\mathbf{X}$ is a two-dimensional array, which restricts its capability of integrating further information [38]. In this way, we can either drop one of the attributes or force the third dimension to be combined with the second dimension. Figure 2(a) shows the case where $\mathbf{X}'$ contains only *source*×*type*, and Figure 2(b) shows the case where $\mathbf{X}''$ contains *source*×(*type*+*type*), where (*type*+*type*) can be considered as a repeated vector to jointly represent the event activity and the day.

With the behavior described by $\mathbf{X}$, the bottom part of Figure 2(a-b) show the respective low-rank factor matrices approximated by NMF, the *source* factors (left), and the *type* factors (right), since they provide the low-dimensional representations of each source and each activity, respectively. Compared to $\mathbf{X}'$, $\mathbf{X}''$ has the additional advantage of revealing the low-dimensional representation of each activity on different days.

3.2.2 NTF. Alternatively, we can use a tensor to represent the same dataset (Figure 2(c)). With the given data of a ternary relation nature [43], we could use a third-order tensor $\mathcal{X}$ to denote a *source*×*day*×*type* activity.

NTF techniques can be applied to obtain three low-dimensional representations: *source* factors, *day* factors, and *type* factors, as $\mathcal{X} \approx \left[\mathbf{S}, \mathbf{D}, \mathbf{T}\right]$. As a result, each pattern comes with a set: a between-activity vector $\mathbf{t}$ to describe the activity dynamics; a between-source vector $\mathbf{s}$ to describe the usage tendency between different sources; and an across-day vector $\mathbf{d}$ to describe the temporal dynamics (e.g., p_1 in Figure 2(c)). Compared to factorizing the unfolding matrix $\mathbf{X}''$ (Figure 2(b)), NTF introduces the day-specific factors. This significantly increases the presentation capability of the patterns by revealing a more direct across-day (temporal) dynamics. Each pattern now represents the interplay of three factors, describing the tendency from different perspectives. With the rich attributes in the behavioral dataset on MOOCs, we thus use tensors to model

the behaviors, with the hope that doing so can provide behavioral patterns with the interpretations from different aspects.

3.2.3 Discriminant NTF. Standard NTF provides meaningful patterns for simultaneously analyzing the behaviors from multiple aspects. This substantially increases the capability of studying and interpreting the behaviors on MOOC platforms with rich dynamics. However, this still does not sufficiently serve the desire to comprehensively understand and investigate student behaviors on these platforms. For example, one of the most interesting questions is which behavior patterns are shared by *completed* students and *dropout* students, and which differentiate the two groups (Figure 2(d)). Through understanding the commonality and differences, researchers can better design course interactions and content to help more students successfully complete the course.

One could easily use NTF to fit the behavior tensors from each group of users, separately. However, this approach does not take advantage of any shared behavior patterns between the two groups. As the behavior moves to high-dimensional tensor space, this could potentially lead to the under-fitting problem. Besides, with patterns generated for each data tensor separately, it needs to perform an additional post-hoc analysis to determine common and discriminative patterns. This is a non-trivial attempt to align the common and discriminative patterns, in the case of each pattern being represented by multiple vectors from different aspects. In this regard, discriminative NTF is set to jointly factorize the tensors constructed from different groups of users with the following objective considering CP decomposition:

$$\mathcal{L}_{\text{disc}} = \left\|\mathcal{X}_{\text{Completed}} - \left[\mathbf{S}, \mathbf{D}, \mathbf{T}\right]\right\|^2 + \left\|\mathcal{X}_{\text{Dropout}} - \left[\mathbf{S}', \mathbf{D}', \mathbf{T}'\right]\right\|^2 + \Omega(\mathbf{S}, \mathbf{D}, \mathbf{T}, \mathbf{S}', \mathbf{D}', \mathbf{T}'), \tag{1}$$

where $\Omega(\cdot)$ is the function to promote the simultaneous discovery of the common and discriminative patterns [22, 47].

3.2.4 Hierarchical NTF. Previous works on NTF or discriminative NTF for unsupervised pattern discovery focus on finding a set of patterns at equal granularity, or in a flat structure. Although they are adequately expressive to reveal the behavior dynamics from different aspects, they can not provide the relations between patterns (such as parent-child and sibling relations).

A hierarchical non-negative tensor factorization (HierNTF) is more desirable than a set of "flat" patterns, because one can work with the pattern exploration in a hierarchy. As opposed to going through each pattern individually, this results in more efficient pattern understanding. HierNTF can be analogous to hierarchical topic modeling, such as hierarchical Latent Dirichlet Allocation (hLDA) [10], where patterns at higher levels in the hierarchy present "abstract" behavior topics, and ones at lower levels reveal more "specific" behavior topics.

3.2.5 Problem Statement. Our problem falls into the combination of the discriminative NTF and HierNTF. We would like to identify common and discriminative patterns nested at multiple levels for a deeper understanding of the relationship between students' multi-way behavior dynamics and their course performance. Before we give the formulation of the studied problem, we would like to first clarify some basic concepts used later.

Definition 3.1. *Individual Behavior Tensor.* Let $\mathcal{X}^{(u)} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_M}$ be an M-way tensor representing an individual user u, with each entry in the $\mathcal{X}^{(u)}$ being an activity from user u that is jointly described by M attributes.

The attributes can be data or platform dependent, such as a time-varying attribute tensor constructed from demographic and behavior attributes associated with users at different time stamps [36]. The individual behavior tensor can be considered as a multi-way representation of each student. With that, for each performance group, we can compute the *collective behavior tensor*.

Definition 3.2. *Collective Behavior Tensor.* Let $\mathcal{X}_c \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_M}$ be an M-way tensor that users M attributes to describe the collective activities from a group of users indexed by c.

Students at each performance group can be jointly represented by a collective behavior tensor that captures the full multi-way feature interactions of their activities. Then, we can combine the tensors for the two opposite performance groups to construct the *coupled tensors*.

Definition 3.3. *Coupled Tensors.* Coupled Tensors $\boldsymbol{\mathcal{X}} = \{\mathcal{X}_c\}$ is a pair of tensors with identical attributes, i.e., $I_1^c = I_1^{\bar{c}}, I_2^c = I_2^{\bar{c}}, \ldots, I_M^c = I_M^{\bar{c}}$, where c is the index of user group that tensor $\mathcal{X}$ is constructed from and $\bar{c}$ represents the counterpart class.

The coupled tensors $\boldsymbol{\mathcal{X}}$ can be constructed in various ways, depending on the performance metric selected, such as $c \in$ {dropouts, completion} or $c \in$ {certificates, no-certificates}. Inspired by [35], we construct the coupled tensor as follows:

$$\mathcal{X}_c = \frac{1}{|U_c|} \sum_{u \in U_c} \mathcal{X}^{(u)}, \tag{2}$$

where U_c is the subset of users u with u belonging to class c. While each individual behavior tensor captures the full-order feature interactions explained by her activities, the full interplay between the M attributes for each group of students can be contained within the tensor structure corresponding to the group.

Definition 3.4. *Multi-way Behavior Pattern.* A multi-way behavior pattern is a a collection of M vectors $(x^{(1)}, x^{(2)}, \ldots, x^{(M)}), M \geq 2$, where $x^{(m)} \in \mathbb{R}^{I_m}$ is a vector to describe the pattern with the m-th attribute.

Definition 3.5. *Pattern Hierarchy.* Let $\mathbf{P}^l \in \mathbb{R}^{R_l \times R_{l-1}}$ denote a pattern hierarchy that specifies the relationships between the behavior patterns in the consecutive levels, i.e., level l and level $l-1$, where $1 \leq l \leq L$. $\mathbf{P}^l$ can be considered as a projection matrix that maps the pattern from the l-th level to the ones at the $(l-1)$-th level.

Definition 3.6. *Tensor-based User Embedding.* A tensor-based user embedding $v \in \mathbb{R}^R$ is the student's vector representation that preserves each student's behavior tensor with a lower-dimensional feature space $\mathbb{R}^R$, given the tensor's rank R.

With the definitions above, we define the iterative common and discriminative pattern analysis as follows:

Problem 1. Given a set of individual behavior tensors $\mathcal{X}^{(u)} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_M}, (u = 1, 2, \ldots, N)$ corresponding to C categories, $C = 2$, and a set of unseen test data $\mathcal{X}^{(t)} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_M}, (t = 1, 2, \ldots, T)$, our goal is to iteratively identify a set of patterns that reveal the

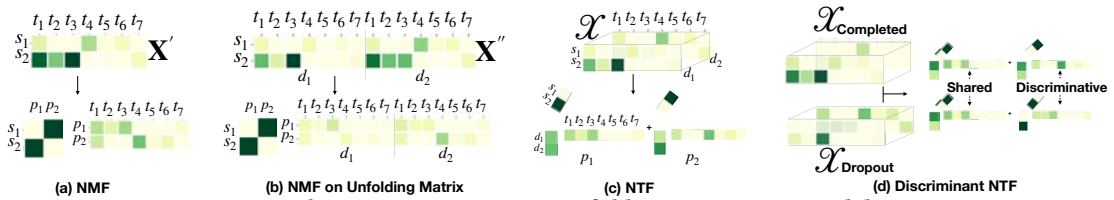


Figure 2: Comparison between NMF, NMF on unfolding Matrix, NTF and discriminant NTF.

common and discriminative behaviors at multiple levels, and then use the learned patterns as bases to infer the user embeddings for prediction of the group membership from the unseen students.

Specifically, the task is threefold: (1) collective behavior pattern inference for discovering the common and discriminative patterns; (2) iterative pattern discovery at multiple levels with hierarchy; and (3) embedding projection for test samples based on the iterative patterns for classification. In other words, the first two tasks are aimed at revealing interpretable patterns that could explain the interplays between the behavior attributes, and the last task is to discover the relationship between student performance and the multi-way patterns.

4 SOLUTIONS

In this section, we introduce an iterative tensor factorization method named `iDisc` for the coupled tensors, $\boldsymbol{\mathcal{X}} = \{\mathcal{X}_c\}$. Figure 3 illustrates the overview of `iDisc`. There are two-stages: (1) iterative application of discriminative tensor subspace learning; and (2) representation learning for the unseen student based on the multi-level patterns.

4.1 Iterative Discriminative Tensor Subspace Learning

This component iteratively applies the following two-step approach: (1) discriminant low-rank tensor approximation, followed by (2) computing and passing the residual tensor into the next level.

4.1.1 Discriminant Tensor Factorization. Conventional tensor factorization seeks a set of N factor matrices $\left[\mathbb{U}\right] = \left[\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}\right]_c^l$ from a behavior tensor $\mathcal{X}_c^l$ at level l for class c. One such example is:

$$\mathcal{L}^l = \underbrace{\sum_c \left\| {\mathcal{X}_c}^l - \left[\mathbb{U}\right]_c^l \right\|^2}_{\text{Loss for Coupled Tensors Factorization}} . \tag{3}$$

Through Eq. 3, we could obtain a set of independent factor matrices (or behavior patterns) for each of the performance groups, respectively. However, this does not consider the commonality and differences among the coupled tensors.

In order to take the commonality between the two behavior tensors into consideration and allow discrimination against each other, we introduce two sets of auxiliary tensors $\mathcal{S}_c^l$ and $\mathcal{Z}_c^l$ that capture the shared behavior and the discriminative behaviors among the coupled tensors. Inspired by [47], $\mathcal{S}_c^l$ and $\mathcal{Z}_c^l$ are computed based on the coupled tensors $\boldsymbol{\mathcal{X}}$ with the clamping function (Eq. 7 and Eq. 9 in [47]). The rational behind the auxiliary tensors is that; we would like to have discriminative tensors that contain only the unique signals for each class, and common tensors to hold what is shared among the coupled tensors. A collective tensor factorization framework is then leveraged to jointly factorize the coupled tensors and the auxiliary tensors as follows:

$$\begin{aligned}\mathcal{J}^l = \mathcal{L}^l + \underbrace{\lambda_0\Big(\sum_c \left\| {\mathcal{Z}_c}^l - \left[\mathcal{W}_{\mathcal{Z}}; \mathbb{U}\right]_c^l \right\|^2 + \sum_c \left\| {\mathcal{S}_c}^l - \left[\mathcal{W}_{\mathcal{S}}; \mathbb{U}\right]_c^l \right\|^2\Big)}_{\text{Loss for Auxiliary Tensor Factorization}} \\ + \underbrace{f\big(\mathbf{U}_c^l, \mathbf{U}_{\bar{c}}^l, \mathcal{W}_{\mathcal{S}}^l\big)}_{\text{Loss for Pattern Alignment}} + \underbrace{g\big(\mathbf{U}_c^{l-1}, \mathbf{U}_c^l, \mathbf{P}^l\big)}_{\text{Loss for Pattern Hierarchy}} + \underbrace{h\big(\mathbf{P}^l\big)}_{\text{L1 penalty}} \\ s.t. \left\|\mathbf{U}_c^l\right\|_2 = 1, \forall\, c,\end{aligned} \tag{4}$$

where:

- $\mathcal{W}_{\mathcal{Z}}$ and $\mathcal{W}_{\mathcal{S}}$ are the core tensors with super diagonal entries;
- $f\big(\cdot\big)$ is the function to enforce the similar components to be aligned correspondingly and defined as:

$$f\big(\mathbf{U}_c^l, \mathbf{U}_{\bar{c}}^l, \mathcal{W}_{\mathcal{S}}^l\big) = \lambda_1 \sum_m \left(\left\| \mathrm{diag}\big(\mathcal{W}_{\mathcal{S}q}\big)\mathbf{U}_c^m - \mathrm{diag}\big(\mathcal{W}_{\mathcal{S}\bar{c}}\big)\mathbf{U}_{\bar{c}}^m \right\|^2 \right); \tag{5}$$

- $g\big(\cdot\big)$ is the function that learns the shared mapping $\mathbf{P}^l$ by the coupled tensors, between the patterns at the consecutive levels. We can consider this operation as performing a matrix decomposition from $\mathbf{U}_c^{l-1}$ to $\mathbf{U}_c^l$ and $\mathbf{P}^l$ as:

$$g\big(\mathbf{U}_c^{l-1}, \mathbf{U}_c^l, \mathbf{P}^l\big) = \lambda_2 \left\| \mathbf{U}_c^{l-1} - \mathbf{U}_c^l \mathbf{P}^l \right\|^2, \tag{6}$$

  assuming that we already have the values of $\mathbf{U}_c^{l-1}$ for l-th level pattern discovery;
- $h\big(\cdot\big)$ is an $L1$ penalty function that encourages sparsity in $\mathbf{P}^l$ to promote the exclusive mapping between the factor matrices at the consecutive levels. Considering a more interpretable pattern hierarchy, we use $L1$-norm, since it can function as a proxy for the $L0$ norm, to minimize the number of nonzero elements while maintaining the convexity of the cost function when estimating P the others fixed. In this manner, we ensure the higher-level patterns are mapped to exclusive lower-level ones; and
- $\lambda_0, \lambda_1, \lambda_2, \lambda_3$ are the respective parameters to weigh in each objective.

With Eq. 4, common and discriminative patterns discovery at the l-th level becomes an optimization objective as:

$$\theta^l = \mathrm{argmin}_\theta^l \; \mathcal{J}^l, \tag{7}$$

where $\theta^l = \{\mathbb{U}, \mathcal{W}_{\mathcal{Z}}, \mathcal{W}_{\mathcal{S}}, \mathbf{P}\}_c^l$.

4.1.2 Obtain the residual tensors. Once θ^l is determined, the reconstructed tensor can be obtained by:

$$\hat{\mathcal{X}}_c^l \approx \left[\mathbb{U}_c^l\right], \tag{8}$$

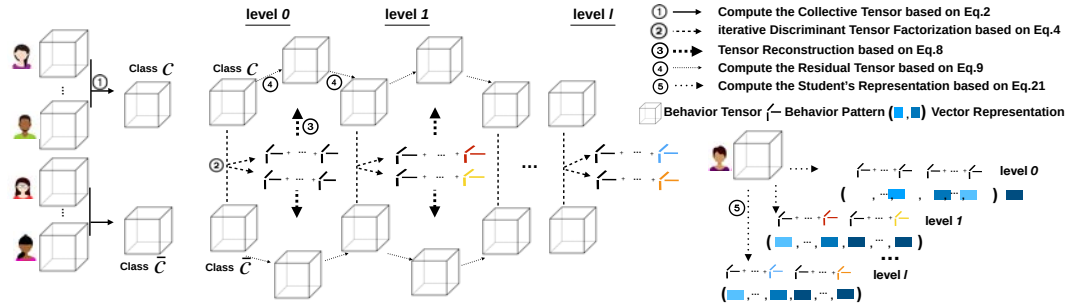


Figure 3: The overview of iDisc's workflow.

and therefore the residual tensor can be computed as:

$$\mathcal{E}_c^l = \mathcal{X}_c^l - \hat{\mathcal{X}}_c^l, \forall\, c. \tag{9}$$

Let $\mathcal{X}_c^{l+1}$ denote the tensors for the identification of common and discriminative patterns at the next level $l+1$. We first obtain $\mathcal{X}_c^{l+1}$ as: $\mathcal{X}_c^{l+1} = \mathcal{E}_c^l$, where $\mathcal{E}_c^l$ is the residual tensor as aforementioned. With $\mathcal{X}_c^{l+1}$, we can further identify the common and discriminative patterns at level $l+1$ with Eq. 4.

Algorithm 1: Algorithm iDisc for discovering the shared and discriminative subspace from coupled tensors at multiple levels.

Input : Coupled tensors $\mathcal{X}$, $\{R\}_l^L$, and $\{\lambda_0, \lambda_1, \lambda_2, \lambda_3\}$
Output: $\{\theta\}_l^L$, where $\theta = \{\mathbb{U}, \mathcal{W}_\mathcal{Z}, \mathcal{W}_\mathcal{S}, \mathbf{P}\}_c$

```
Let l = 0;
while l < L do
    Construct {Z_c}^l and {S_c}^l from {X_c}^l based on Eq.7 and Eq.9 in [47];
    Obtain θ^l by solving Eq. 4;
    Reconstruct {X_c}^l based on Eq. 8;
    Compute E_c^l based on Eq. 9;
    Let l = l + 1;
    Let X_c^l = E_c^l
end
```

4.1.3 Parameter Optimization of iDisc. For simplicity of notation, we omit the level l in all notations since the optimization is performed per level with the focus on the unknown θ^l and the θ^{l-1} are learned at level $l-1$. We also omit the mode notation m because all modes share the same optimization process. Let $\mathbf{U}_c$ represent the mode-m factor matrix at level-l, instead of the rather complex form of $\{\mathbf{U}_c^{(m)}\}_l$. We use $\overline{\mathbb{U}}_c$ to denote the set of factor matrices for $\mathcal{X}_c$ that correspond to modes other than m, and $\bar{c}$ to denote the class that is not c.

Since objective function $\mathcal{J}$ is not convex with respect to θ, we aim to find a local minimum for $\mathcal{J}$ by iteratively updating each in θ.

1. Update $\mathbf{U}_c$*, fix others.* The optimization of $\mathbf{U}_c$ is equivalent to the following least squares loss functions [47]:

$$\begin{aligned}\mathbf{U}_c \leftarrow \underset{\mathbf{U}_c \geq 0}{\operatorname{argmin}} \frac{1}{2}\Big(\frac{1}{n_c}\Big\|\mathbf{X}_c - \mathbf{U}_c\,(\odot\overline{\mathbb{U}}_c)^T\Big\|_F^2\Big) \\ + \lambda_0\Big(\Big\|\mathbf{Z}_c - \mathbf{U}_c\Lambda_{\mathbf{w}_{Z_c}}(\odot\overline{\mathbb{U}}_c)^T\Big\|_F^2 + \Big\|\mathbf{S}_c - \mathbf{U}_c\Lambda_{\mathbf{w}_{S_c}}(\odot\overline{\mathbb{U}}_c)^T\Big\|_F^2\Big) \\ + \lambda_1\Big\|\mathbf{U}_c\Lambda_{\mathbf{w}_{S_c}} - \mathbf{U}_{\bar{c}}\Lambda_{\mathbf{w}_{S_{\bar{c}}}}\Big\|_F^2 + \lambda_2\Big\|\mathbf{U}_c^{l-1} - \mathbf{U}_c\mathbf{P}\Big\|^2,\end{aligned} \tag{10}$$

where $\mathbf{X}_c$ is the mode-m unfolding of tensor $\mathcal{X}_c$. Then the gradient update of $\mathbf{U}_c$ can be computed as:

$$\begin{aligned}\nabla_{\mathbf{U}_c}\mathcal{J} = \frac{1}{n_c}\Big(\mathbf{U}_c(\odot\overline{\mathbb{U}}_c)^T - \mathbf{X}_c\Big)(\odot\overline{\mathbb{U}}_c) \\ + \lambda_0\Big(\Big(\mathbf{U}_c\Lambda_{\mathbf{w}_{Z_c}}(\odot\overline{\mathbb{U}}_c)^T - \mathbf{Z}_c\Big)(\odot\overline{\mathbb{U}}_c)\Lambda_{\mathbf{w}_{Z_c}}^T + \Big(\mathbf{U}_c\Lambda_{\mathbf{w}_{S_c}}(\odot\overline{\mathbb{U}}_c)^T - \mathbf{S}_c\Big)(\odot\overline{\mathbb{U}}_c)\Lambda_{\mathbf{w}_{S_c}}^T\Big) \\ + \lambda_1\Big(\mathbf{U}_c\Lambda_{\mathbf{w}_{S_c}} - \mathbf{U}_{\bar{c}}\Lambda_{\mathbf{w}_{S_{\bar{c}}}}\Big)\Lambda_{\mathbf{w}_c} - \lambda_2\mathbf{P}\Big(\mathbf{U}_c^{l-1} - \mathbf{U}_c\mathbf{P}\Big).\end{aligned} \tag{11}$$

2. Update $\mathcal{W}_{\mathcal{Z}_c}$*, fix others.*
Let $\mathbf{w}_{Z_c}$ denote $\mathbf{w}$ for tensor $\mathcal{X}_c$. The optimization of $\mathbf{w}_{Z_c}$ is equivalent to the following problem:

$$\mathbf{w}_{Z_c} \leftarrow \underset{\mathbf{w}_c \geq 0}{\operatorname{argmin}}\ \lambda_0\Big\|\mathbf{Z}_c - \Lambda_{\mathbf{w}_{Z_c}}(\odot\mathbb{U}_c)^T\Big\|^2, \tag{12}$$

where $\Lambda_{\mathbf{w}_{Z_c}} \in \mathbb{R}^{R\times R\times R}$ is the tensor with $\mathbf{w}_{Z_c}$ as its super-diagonal entries. The gradient update of $\mathbf{w}_c$ is:

$$\nabla_{\mathbf{w}_{Z_c}}\mathcal{J} = \lambda_0\Big(\Lambda_{\mathbf{w}_{Z_c}}(\odot\mathbb{U}_c)^T - \mathcal{Z}_c\Big)(\odot\mathbb{U}_c). \tag{13}$$

3. Update $\mathcal{W}_{\mathcal{S}_c}$*, fix others.*
Let $\mathbf{w}_{S_c}$ denote $\mathbf{w}$ for tensor $\mathcal{X}_c$. The optimization of $\mathbf{w}_{S_c}$ is equivalent to the following problem:

$$\mathbf{w}_{S_c} \leftarrow \underset{\mathbf{w}_{S_c} \geq 0}{\operatorname{argmin}}\ \lambda_0\Big\|\mathbf{S}_c - \Lambda_{\mathbf{w}_{S_c}}(\odot\mathbb{U}_c)^T\Big\|^2 + \lambda_1\Big\|\mathbf{U}_c\Lambda_{\mathbf{w}_{S_c}} - \mathbf{U}_{\bar{c}}\Lambda_{\mathbf{w}_{S_{\bar{c}}}}\Big\|^2. \tag{14}$$

The gradient update of $\mathbf{w}_{S_c}$ can be derived as:

$$\nabla_{\mathbf{w}_{S_c}}\mathcal{J} = \lambda_0\Big(\Lambda_{\mathbf{w}_{S_c}}(\odot\mathbb{U}_c)^T - \mathcal{S}_c\Big)(\odot\mathbb{U}_c) - \lambda_1\Big(\mathbf{U}_c{}^T\Big(\Lambda_{\mathbf{w}_{S_c}}\mathbf{U}_c - \Lambda_{\mathbf{w}_{S_{\bar{c}}}}\mathbf{U}_{\bar{c}}\Big)\Big) \tag{15}$$

4. Update $\mathbf{P}$*, fix others.* The optimization of $\mathbf{P}$ is equivalent to a co-regularized collective matrix factorization problem with sparsity constraints [6, 41]:

$$\mathbf{P} \leftarrow \underset{\mathbf{P}\geq 0}{\operatorname{argmin}}\ \lambda_2\Big(\frac{1}{2}\Big\|\mathbf{U}_c^{l-1} - \mathbf{U}_c\mathbf{P}\Big\|^2 + \frac{1}{2}\Big\|\mathbf{U}_{\bar{c}}^{l-1} - \mathbf{U}_{\bar{c}}\mathbf{P}\Big\|^2\Big) + \lambda_3\|\mathbf{P}\|_1. \tag{16}$$

Since the sparsity is applied to each row of $\mathbf{P}$, each row $\mathbf{P}_{(r,:)}$ can be updated based on the following gradient:

$$\nabla_{\mathbf{P}_{(r,:)}}\mathcal{J} = -\lambda_2\Big(\mathbf{U}_{c\,(r,:)}^{T}(\mathbf{U}_{c}^{l-1} - \mathbf{U}_{c\,(:,r)}\mathbf{P}_{(r,:)}) + \mathbf{U}_{\bar{c}\,(r,:)}^{T}(\mathbf{U}_{\bar{c}}^{l-1} - \mathbf{U}_{\bar{c}\,(:,r)}\mathbf{P}_{(r,:)})\Big) + \lambda_3 \mathrm{sgn}(\mathbf{P}_{(r,:)}). \tag{17}$$

The details of iDisc are summarized in Alg. 1 [1].

4.1.4 Time Complexity Analysis. The time complexity is mainly consumed by updating each factor matrix $\mathbf{U}_c$ in iDisc from computing $\nabla_{\mathbf{U}_c}\mathcal{J}$. From Eq. 11, we need to compute $\mathbf{U}_c(\odot\overline{\mathbb{U}}_c)^T(\odot\overline{\mathbb{U}}_c)$ and $\mathbf{X}_c(\odot\overline{\mathbb{U}}_c)$ in the first term. Note $\mathbf{U}_c \in \mathbb{R}^{I_m \times R}$ and $\mathbf{X}_c \in \mathbb{R}^{I_m \times \prod_{i\neq m} I_i}$ and therefore we have $\mathbf{U}_c(\odot\overline{\mathbb{U}}_c)^T(\odot\overline{\mathbb{U}}_c) \in \mathbb{R}^{I_m \times R}$ and $\mathbf{X}_c(\odot\overline{\mathbb{U}}_c) \in \mathbb{R}^{I_m \times R}$. The operation of matricized tensor times Khatri-Rao product ($\mathbf{X}_c(\odot\overline{\mathbb{U}}_c)$) is often considered a bottleneck for CP decomposition due to the expensive computational cost [48]. In practice, the sparsity of the tensor is leveraged for an efficient computation for this operation [2, 48]. Particularly, the complexity can be reduced by only considering the computation for nonzero observations in $\mathcal{X}_c$. Let x_h denote the h-th nonzero observation in $\mathcal{X}_c$ and its subscripts in $\mathcal{X}_c$ as $(I_{1_h}, I_{2_h}, \ldots, I_{M_h})$. If there are H non-zeros, i.g, $H = nnz(\mathcal{X}_c)$, we would just need an H-vector to store the real values of $\mathcal{X}_c$. In this case, the element-wise computation for $\mathbf{X}_c(\odot\overline{\mathbb{U}}_c)$ can be written as:

$$\Big(\mathbf{X}_c(\odot\overline{\mathbb{U}}_c)\Big)_{(i,r)} = \sum_{\substack{h=1 \\ I_{m_h}=i}}^{H} x_h \prod_{\substack{m'=1 \\ m'\neq m}}^{M} \mathbf{U}^{(m')}_{(I_{m'_h},r)}, \quad \text{for } i = 1, \ldots, I_m \text{ , and } r = 1, \ldots, R, \tag{18}$$

where the computation of Khatri-Rao product can be ignored when $x_h = 0$. Therefore, the time complexity for computing $\mathbf{X}_c(\odot\overline{\mathbb{U}}_c)$ for each mode per iteration is $O(nnz(\mathcal{X}_c)I_m R)$. The element-wise of $\mathbf{U}_c(\odot\overline{\mathbb{U}}_c)^T(\odot\overline{\mathbb{U}}_c)$ can be efficiently computed as [1, 42]:

$$\Big(\mathbf{U}_c(\odot\overline{\mathbb{U}}_c)^T(\odot\overline{\mathbb{U}}_c)\Big)_{(i,r)} = \sum_{j=1}^{R}\left(\mathbf{U}^{(m)}_{(i,j)} \prod_{\substack{m'=1 \\ m'\neq m}}^{M} \sum_{i=1}^{I_{m'}} \mathbf{U}^{(m')^T}_{(i,j)} \mathbf{U}^{(m')}_{(i,r)}\right). \tag{19}$$

Therefore, the time complexity as $O(\hat{I}\,R^2)$, where $\hat{I} = \sum_{m'=1}^{M} I_{m'} - I_m$ and the overall time complexity for the above two terms is $O(HI_mR + \hat{I}\,R^2)$. Similarly, the time complexity for terms involving tensors $\mathcal{Z}$ and $\mathcal{S}$ becomes $O(H' I_m R^2 + \hat{I}\,R^3)$ due to the additional loop introduced by the weight vector $\mathbf{w}$, where H' is the respective nonzero observations in the auxiliary tensors. Since $H' \leq H$, with $M \ll H$, $R \ll H$, and $\hat{I} \ll H$, we can see that the running time is expected to scale linearly with the number of nonzero observations in $\mathcal{X}_c$.

4.2 Embedding Learning for the Unseen Student

This section explains the inference of the student's embedding in the latent space anchored by the factor matrices. The individual behavior tensor $\mathcal{X}_t$ for unseen student t with an unknown class is constructed based on his or her logs with the system. The mode settings of tensor $\mathcal{X}_t$ are the same as for the collective behavior tensors (e.g., $\mathcal{X}_c$). With the students' individual tensor and the factor matrices, we follow iDisc to first obtain the corresponding auxiliary $\mathcal{Z}$ tensors at l-th level, $\mathcal{Z}^l_{t_{\hat{c}}}, \forall\, \hat{c} \in \{c, \bar{c}\}$, for student t, by following the clapping function in Eq. 7 in [47]. Then, the equation to compute the embedding becomes:

$$\mathbf{v}^l_{t_{\hat{c}}} = \mathcal{Z}^l_{t_{\hat{c}}} \times_m \left\{\mathbf{U}^{(m)}\right\}^l_{\hat{c}}, \forall\, \hat{c} \in \{c, \bar{c}\}, \tag{20}$$

which follows a typical computation of the core tensor, given the data tensor and its factor matrices.

It is worth noting that since PARAFAC decomposition does not enforce the orthogonal property in each of the factor matrices, direct computation of Eq. 20 is not feasible. Recent work by [8] proposes an efficient estimation of the core tensor for PARAFAC decomposition. By following the algorithm 1 in [8] to efficiently estimate $\mathbf{v}^l_{c_t}$ and then the embedding of student s at l-th level $\mathbf{v}^l_t \in \mathbb{R}^{2R_l}$ can be obtained by:

$$\mathbf{v}^l_t = \left[\mathbf{v}^l_{t_c} | \mathbf{v}^l_{t_{\bar{c}}}\right]. \tag{21}$$

5 EXPERIMENTS

In this section, we conduct systematic experiments to evaluate the quality of iDisc. In following, we first describe the data used for the experiments. The content of the rest of this section is structured to answer the following questions:

- Can iDisc reveal meaningful patterns?
- How does iDisc perform in comparison with state-of-art methods from predictive modeling in learning analytics?
- Can iDisc scale for the dataset at a massive scale?

5.1 Data

We experiment with nine courses/sessions from three publicly-available MOOC platforms: edX, ASSISTments, and XueTangX. The statistics of the datasets are provided in Table 2.

Table 2: Dataset and Tensor Modes Description.

Dataset	Course	#Users	#Activity	#Areas	#Education	Stay	
edX	Course A	57,715	-				
	Course B	66,731	-	34	5	5	
	Course C	169,621	-				
				#Days	#Events	#Source	
XueTangX	Course 1	12,004	652,701				
	Course 2	10,321	877,805	14	7	2	
	Course 3	9,382	907,118				
				#Problems	#KC	#Views	#Duration
ASSISTments	Year 2004	912	580,785	376	58	7	10
	Year 2005	2,392	521,751	266	59	4	10
	Year 2006	2,584	686,868	409	69	4	10

edX. The edX [33] dataset is comprised of de-identified data from "Introduction to Computer Science" (Fall 2012, Spring 2013, and Summer 2013) from MITx (Course A and B in Table 2) and HarvardX (Course C). This dataset does not have detailed event logs. However, the data are aggregated records, where each record represents the summary statistics for one individual's activity in the edX course with her demographic information. We select the three most popular courses from this dataset. For this dataset, we construct a 34x5x5 tensor as *Area*×*Education*×*Stay*. Our goal is to predict whether the course completion certificate is earned by a student at the end of the course.

[1] Code is available at https://github.com/picsolab/iDisc

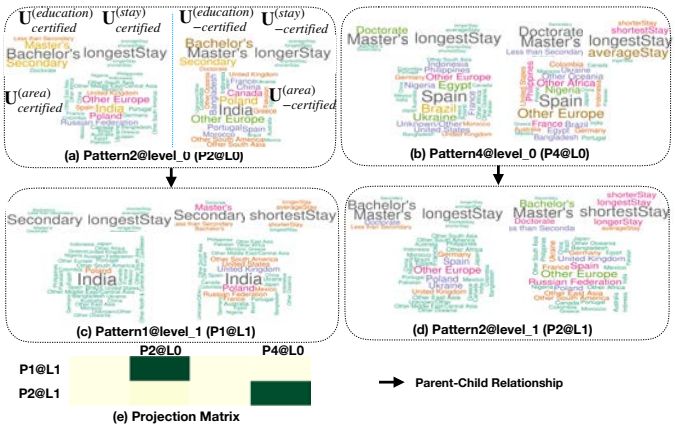


Figure 4: Output generated by `iDisc` on the MITx dataset. (e) shows the relationships between patterns in the first two levels; (a-d) show the associated patterns in the hierarchy.

XueTangX. XueTangX [50] is one of the largest MOOC platforms in China. The full dataset includes 79,186 students enrolled in 39 classes. Each enrollment is associated with a log of the student's activities, including watching lecture videos, working on course problems, accessing course modules, and so on. In total, there are 8,157,277 activity logs, and the longest lifetime of enrollment is five weeks. We take the three most popular courses from this dataset. The dataset statistics are shown in Table 2. We use the first two weeks to learn the factor matrices by constructing a 14x7x2 tensor as *Day×EventType×EventSource*. The goal is to correctly predict course completion.

ASSISTments. ASSISTments [16] is an online tutoring system used by more than 50,000 students around the world [5]. On ASSITments, students attempt to solve problems and receive feedback on those attempts. To assist the learning process, each problem is also associated with multiple knowledge components. We take the public dataset of the Math course on ASSISTments over three years (2004, 2005, and 2006) and the dataset characteristics are shown in Table 2. For each dataset, we construct a four-way tensor as *Problem×KnowledgeComponent×ProblemView ×ActionDuration*. Our aim is to classify the students as over-performing students and under-performing students, in terms of error-rate [2].

5.2 Qualitative Examination of the Patterns

In this section, we use "Introduction to Computer Science" on MITx during Spring 2013 to illustrate the outputs of `iDisc`. We first qualitatively examine the patterns, as well as the pattern hierarchy generated and then explain the relationship between the patterns and the performance outcome.

5.2.1 Common and Discriminative Pattern Discovery. Figure 4 describes the outputs by `iDisc`, with $\{R\}_l^L = \{4, 2\}$ (rank-4 at the first level and rank-2 at the second level). Particularly, Figure 4(e) shows the project matrix $\mathbf{P}^1 \in \mathbb{R}^{2\times4}$ that represents the hierarchical relationships between the set of patterns at the first two levels, where darker colors indicate stronger associations. We observe that pattern #1 at level 1(P1@L1) is strongly associated with pattern #2 at level 0(P2@L0). This suggests that P1@L1 could be a child pattern for P2@L0. Similarly, we observe the parent-child relationship between P4@L0 and P2@L1. Since the projection matrix is shared by the factor matrices from the coupled tensors, it is important to note that each of figures 4(a) through 4(d) refers to both patterns for the certified and non-certified group (i.e. as shown left and right in Figure 4(a)).

[2] https://pslcdatashop.web.cmu.edu/help?page=terms#error_rate

Table 3: Regression results for course end performance. Explanatory variables are the students' embeddings that correspond to the patterns presented in Figure 4 (a-d) (e.g., $\mathbf{v}^0_{certified}(2)$ refers to the values in students' embedding vector v for P2@L0 from the certified group). *Note:* **:p<.05; ***:p<.01.

	Certified (M1)		Certified (M2)
$\mathbf{v}^0_{\text{certified}}(2)$	−0.058 (0.046)	$\mathbf{v}^1_{\text{certified}}(1)$	−0.282 (0.188)
$\mathbf{v}^0_{\text{certified}}(4)$	−0.004 (0.046)	$\mathbf{v}^1_{\text{certified}}(2)$	2.285^{***} (0.649)
$\mathbf{v}^0_{\text{-certified}}(2)$	-0.113^{**} (0.048)	$\mathbf{v}^1_{\text{-certified}}(1)$	-0.620^{***} (0.080)
$\mathbf{v}^0_{\text{-certified}}(4)$	0.026 (0.046)	$\mathbf{v}^1_{\text{-certified}}(2)$	0.733^{***} (0.061)

Due to the space limit, we only discuss the details of two sets of multi-way patterns as word clouds in Figure 4(a) for P2@L0 and Figure 4(c) for P1@L1. Figure 4(a) describes a subgroup of students that are from the most populated countries (e.g., the United States, India, and Poland, based on $\mathbf{U}^{(area)}$) with education background mostly from Secondary to Master's ($\mathbf{U}^{(education)}$), and most of whom tend to stay on the edX platform ($\mathbf{U}^{(stay)}$) for a relatively long period. While the pattern from the certified group of students shares almost identical distributions in the area and educational background, what makes them slightly different was primarily the time spent on the platform; certified students($\mathbf{U}^{(stay)}_{\text{certified}}$) spend relatively longer than un-certified students ($\mathbf{U}^{(stay)}_{\text{-certified}}$). The set of patterns in Figure 4(c) is the child patterns of the Figure 4(a). Compared to their counterpart in level 0, they primarily describe the students from Indian (although the area distribution from the un-certified group of students spans more countries ($\mathbf{U}^{(area)}_{\text{-certified}}$)) with more focus on the middle level of education background (e.g., Secondary($\mathbf{U}^{(education)}_{\text{certified}}$)). The difference in their length of stay on the platform was more prominent in this set of patterns, where certified students have the longest stays with the edX platform ($\mathbf{U}^{(stay)}_{\text{certified}}$), and un-certified students generally have the least ($\mathbf{U}^{(stay)}_{\text{-certified}}$).

5.2.2 Simpson's Paradox Revisited. We performed multivariate logistic regression analysis to identify the patterns that can explain students' variation in obtaining the certificate for each level (M1 for level 0 and M2 for level 1). The dependent variable is whether or not the users are certified at the end of the course. The explanatory variables include the students' embeddings for the aforementioned patterns in Figure 4(a-d) in each level (e.g., $\mathbf{v}^1_{\text{certified}}(1)$ refers to the embeddings corresponding to P1@L1$_{\text{certified}}$). The embeddings are standardized to facilitate comparison among different variables.

Table 3 shows the estimated coefficients for M1 and M2. The only significant variable in M1 is the embeddings $\mathbf{v}^0_{\text{-certified}}(2)$ ($\beta = -0.113$, $p < .05$). This suggests that users who have shown more

activities in line with pattern P2@L0$_{\text{-certified}}$ appear to have less chance to earn the certificate. M2 shows that both embeddings $\mathbf{v}^1_{\text{certified}}(2)$ ($\beta = 2.285, p < .01$) and embeddings $\mathbf{v}^1_{\text{-certified}}(2)$ ($\beta = 0.733, p < .01$) reveal a significant and positive effect towards earning the certificate, with $\mathbf{v}^1_{\text{certified}}(2)$ having a much larger effect size. On the other hand, M2 also shows a significant and negative effect of having a larger value in $\mathbf{v}^1_{\text{-certified}}(1)$ ($\beta = -0.620, p < .01$).

We can consider multi-way patterns as principal components in PCA that bridge the original feature interactions in the high-dimensional space and the associated *loadings*. In this case, we would expect students from one class to have higher loading scores associated with patterns that are extracted from the same class. The regression result in M1 shows that the un-certified group of students does have larger loading scores. However, that is true only in $\mathbf{v}^0_{\text{-certified}}(2)$ for P2@L0 . This is not surprising, because in level 0 the two sets of patterns are in fact very similar to each other, as shown in Figure 4. The results in M2 confirm this expectation, with significantly higher loading scores $\mathbf{v}^1_{\text{certified}}(2)$ for certified students and significant higher loading scores $\mathbf{v}^1_{\text{-certified}}(1)$ for un-certified students. Contrary to our expectation, the certified group of students also have higher loading scores in $\mathbf{v}^1_{\text{-certified}}(2)$ (although much lower than $\mathbf{v}^1_{\text{certified}}(2)$). This could be explained by our observation in Figure 1, where P2@L1 captures a cluster of highly-educated students from certain European areas. They have a much higher chance of obtaining the certificate, regardless of having the longest stay or shortest stay with the platform.

Summary. We qualitatively examine the outputs generated by `iDisc`. The results show interesting properties of the proposed method. Our model reveals common and discriminative patterns at each level with their relationship explained via the projection matrix. Our regression analysis first explains the discriminative capability of the students' embeddings based on this set of patterns. More importantly, the analysis validates that the students' embeddings can be used to measure the relationship between the performance outcome with multi-way patterns from `iDisc`.

5.3 Quantitative Comparison

In this section, we report the results from the quantitative experiments in comparison with existing work commonly used in predictive analytics. Specifically, we conduct a classification task, in which the goal is to predict the students' performance at the end of the course defined in Section 5.1.

5.3.1 Baselines. We include baselines that are commonly seen in the area of predictive analytics in educational data mining as:

- Raw. We use the raw activity counts each day as features to train classifiers for prediction. This is the most common approach in predictive modeling for MOOCs.
- LDA. Coleman et al. [4] use LDA to capture the temporal element of the behavior data. We first discover the latent behavior patterns from Raw features with a varying number of topics, and use the topic membership of each student for the classification task.
- LadFG [36]. As one of the most cited works in MOOC predictive modeling, LadFG is a latent dynamic factor graph model that finds a mapping from students' time-varying attribute tensor to the observed learning outcome. We only evaluate the performance of LadFG in XueTangX dataset because it is the only one of the three that contains the necessary temporal dynamics.
- Factorization machines (FM) [37]. Factorization machines have been proposed and successfully applied to recommendation and prediction tasks. As the factorization model projects the input feature space into a latent space, it enables the learning of more complex interactions between features. We first convert each dimension as dummy variables for each student, and then concatenate all dimensions as a wide feature matrix.

It is worth noting the recent use of Deep Neural Nets (DNN) and their variants have shown promising performance compared to conventional machine learning approaches (e.g., [21, 45, 49]). However, the lack of interpretability of these models prevents their further application in problems driven by both interpretations and performance gains. We also compare `iDisc` with the existing work on discovering the common and discriminative patterns from multi-way data.

- `SDCDNTF`. `SDCDNTF` extends [22] and learns the common and discriminative patterns with different ranks. The input to the model consists of the rank and the number of shared patterns, along with the coupled tensors.
- `PairFac`. `PairFac` [47] learns the common and discriminative patterns with different ranks. Comparing to `SDCDNTF`, it does not require the input of the split.

We would like to point out that standard tensor factorization could serve as another baseline to compare with, in which students or their class reside as one of the dimensions and the corresponding factor matrix can naturally become features for downstream prediction tasks. However, we did not include it for two reasons: first, because the aforementioned baselines work as inductive models, where unseen students can be predicted based on the learned parameters, while simple tensor factorization serves as a transductive model and only predicts for the students that are available in the factorization; and second, because student populations on MOOC platforms can be of any size, from small to very large, the efficiency of standard tensor factorization with a large dimension size could be a practical problem for its real-world application.

5.3.2 Experiment Settings. For each dataset, we draw a training set of students from each class with replacement, and then obtain the embeddings of the out-of-bootstrap students. For this set of students, we perform a five-fold cross-validation with a k Nearest Neighbors classifier. We conduct five independent trials of this experiment and report the average classification accuracy. We select accuracy since both the training and testing dataset are constructed in a way that each class has an equal amount of students. For `SDCDNTF`, we experiment with α, β and $\gamma \in \{10^{-5}, 10^{-4}, 10^{-3}\}$. Finally, we set α and β in the same range, and $R = 6$ for both `PairFac` and `SDCDNTF` and derive two versions of `SDCDNTF` using $K \in \{2, 4\}$. To make a fair comparison with `PairFac` and `SDCDNTF` , we use two-level pattern discovery with rank-4 and rank-2 in each level for `iDisc`, respectively. For LDA and FM, we experiment with a varying number of topics /factors and report the best performance. For LadFG, we keep the suggested parameters from their paper.

Table 4: Classification Results in Accuracy.

Dataset		edX Course A	edX Course B	edX Course C	XueTangX Course 1	XueTangX Course 2	XueTangX Course 3	ASSISTments Year 2004	ASSISTments Year 2005	ASSISTments Year 2006
Baselines	Raw	90.50	91.55	87.34	61.99	69.15	69.37	64.23	57.20	60.36
	LDA	76.83	75.04	75.17	66.93	71.39	70.44	62.79	66.50	66.44
	FM	93.98	94.21	88.89	65.31	70.50	69.43	74.10	69.32	70.20
	LadFG	-	-	-	68.35	72.63	73.56	-	-	-
	SDCDNTF2	94.42	96.55	93.43	67.23	71.50	71.27	61.52	59.79	60.94
	SDCDNTF4	94.37	96.47	93.46	67.15	72.56	71.19	61.49	62.32	60.88
	PairFac	94.44	95.54	93.19	67.40	71.96	72.22	69.54	67.59	70.26
Proposed Method	iDisc-1st	94.10	95.28	93.01	66.56	71.24	72.36	72.82	69.47	66.50
	iDisc-2nd	**95.37**	**96.86**	**94.58**	**69.39**	73.27	73.13	**78.37**	**72.84**	**73.26**
	iDisc-Comb.	94.79	96.14	93.76	69.35	**74.00**	**74.12**	77.47	72.65	71.49

5.3.3 Experiment Results. Table 4 shows the classification results. The raw features perform poorly, especially in XueTangX and ASSISTments dataset, with the score on accuracy in the range of 60-70%, which suggests the difficulty of the prediction task. LDA saw different performance, with noticeable drops in the edX dataset in comparison to raw features. We conjecture that the construction of the raw features results in a high dimensional and sparse feature space, which could potentially cause LDA to suffer from learning merely meaningful latent topics. Factorization machines slightly improve the performance. FMs can be considered as a generalization of tensor factorization with the additional modeling of interactions within each dimension [37]. Although we observe noticeable gains from FMs over Raw features and LDA, FMs do not perform as well as tensor-based methods. We suspect there might be two reasons for this: 1) the current feature space might not be as well tuned for general prediction tasks as it is for the more commonly seen recommendation tasks; 2) compared to tensor-based methods that only consider the interactions between different dimensions, FMs could potentially over-fit the interaction effects between and within dimensions in the training data. We observe that LadFG achieves large gains over the raw features for the XueTangX dataset. This indicates there could exist some hidden patterns that can capture the temporal elements of the behavior data. We also notice that tensor-based models such as PairFacand SDCDNTFperform better than LDA, especially in Course 2 and Course 3. This suggests that by systematically considering the multi-way interactions, the performance could be further improved. Finally, the best performance of iDisc is statistically comparable with the state-of-art LadFG and significantly better than the rest of the baselines. While LadFG is geared towards student performance predictions, iDisc can provide comparable prediction performance as well as meaningful patterns.

Summary. iDisc constructs students' embeddings that integrate relations between the multi-way interactions and the performance outcome. The quantitative experiment demonstrates the discriminative capability of iDisc, and iDisc outperforms the baselines in nine datasets from three MOOC platforms. Higher-level embeddings from iDisc have shown stronger discriminative powers over ones from the lower levels, which we will discuss in next section. Since there is no trivial solution in determing the rank of the tensor decomposition, we experimented with different rank settings for iDisc (e.g., {2, 4}, {3, 3}) and this observation still holds.

Table 5: Running time for varying number of observations.

#observations	10^2	10^3	10^4	10^5	10^6
running time/epoch	0.31s	0.35s	1.13s	1.23s	2.05s

5.4 Scalability

Since many of the education platforms have seen exponential growth in usage, scalable solutions of learning analytic are another critical aspect of adoption. In this section, we test the scalability of iDisc. In this experiment, we choose the ASSISTments dataset for the year of 2006, since it has the largest tensor settings in our experiment. We run iDisc with a varying number of entries in the data tensor, from $\{10^2, 10^3, 10^4, 10^5, 10^6\}$. Table 5 reports the average running time per epoch, and we observe that the running time scales almost linearly with the exponential increase in observations in the tensor. This result is consistent with the analysis in Section 4.1.4.

6 CONCLUSION

In this paper, we present a tensor-based learning framework, iDisc, to perform common and discriminative pattern discovery at multiple levels for understanding of high-dimensional student behavior and performance prediction in MOOCs. We first use tensors to represent each user's behavior, and construct coupled tensors to aggregate behavior for users with contrasting performance groups. Then, we iteratively identify the shared and distinct behavioral patterns at various levels, while revealing the hierarchical relationship between them to further increase the interpretability of the output. Finally, we use these patterns as anchors to generate the students' latent representation for down-stream performance prediction. Our qualitative examination of the patterns has shown the multi-level, multi-aspect and hierarchical characteristics of behavior patterns on the edX platform. The quantitative experiments, compared to both traditional predictive methods as well as existing discriminative tensor factorization models, suggest promising results by iDisc in several datasets from different MOOC platforms.

To the best of our knowledge, this is the first attempt to tackle the joint problem of discriminant tensor factorization and hierarchical pattern discovery for understanding such behavior on MOOC platforms. This enables the in-depth comprehension of students' multi-way behavior dynamics, as well as its association with course performance. Nevertheless, one of the limitations is that it merely

provides the relationships between the latent multi-way interaction and the performance outcome, with no intention to draw causal reasoning between them. In practice, iDisc can be developed as a plugin for MOOC platforms, where instructors can examine the multi-aspect contrasting behavior and connect the difference to the course outcome. Considering the XueTangX platform, one of the multi-aspect patterns could refer to a set of events at the beginning of the course that trigger from the server. if iDisc reveals its positive association to the success of students' course end performance, this pattern can be used as guidance of promotions for both the instructor and the platform to improve the students' learning outcome. Last, but not least, compared to other tensor factorization methods, iDisc provides a more efficient exploration of the multi-aspect patterns due to its multi-level nature. However, we understand that the interpretation of the multi-aspect pattern itself is not straightforward in general. In our future work, we would like to follow a more human-centric approach and develop a visual analytic system that helps domain experts interpret and understand the multi-aspect patterns.

ACKNOWLEDGEMENT

The authors would like to acknowledge the support from NSF #1634944, #1637067, and #1739413. Any opinions, findings, and conclusions or recommendations expressed in this material do not necessarily reflect the views of the funding sources.

REFERENCES

[1] Evrim Acar, Daniel M Dunlavy, and Tamara G Kolda. 2011. A scalable optimization approach for fitting canonical tensor decompositions. *Journal of Chemometrics* 25, 2 (2011), 67–86.

[2] Joon Hee Choi and S Vishwanathan. 2014. DFacTo: Distributed factorization of tensors. In *Proceedings of the 27th International Conference on Neural Information Processing Systems, NIPS'14*. 1296–1304.

[3] Andrzej Cichocki. 2014. Tensor networks for big data analytics and large-scale optimization problems. *arXiv preprint arXiv:1407.3124* (2014).

[4] Cody A Coleman, Daniel T Seaton, and Isaac Chuang. 2015. Probabilistic use cases: Discovering behavioral patterns for predicting certification. In *Proceedings of the Second ACM Conference on Learning at Scale*. ACM, 141–148.

[5] Christopher Donnelly. 2015. Enhancing Personalization Within ASSISTments. (2015).

[6] Michael Elad. 2010. From exact to approximate solutions. In *Sparse and Redundant Representations*. Springer, 79–109.

[7] Mi Fei and Dit-Yan Yeung. 2015. Temporal models for predicting student dropout in massive open online courses. In *Proceedings of the 2015 IEEE International Conference on Data Mining Workshop (ICDMW)*. IEEE, 256–263.

[8] Xiao Fu, Kejun Huang, Evangelos E Papalexakis, Hyun-Ah Song, Partha Pratim Talukdar, Nicholas D Sidiropoulos, Christos Faloutsos, and Tom Mitchell. 2016. Efficient and distributed algorithms for large-scale generalized canonical correlations analysis. In *Proceedings of the 16th International Conference on Data Mining (ICDM)*. IEEE, 871–876.

[9] Nabeel Gillani, Rebecca Eynon, Michael Osborne, Isis Hjorth, and Stephen Roberts. 2014. Communication communities in MOOCs. *arXiv preprint arXiv:1403.4640* (2014).

[10] Thomas L Griffiths, Michael I Jordan, Joshua B Tenenbaum, and David M Blei. 2003. Hierarchical topic models and the nested chinese restaurant process. In *Proceedings of the 16th International Conference on Neural Information Processing Systems, NIPS'03*. 17–24.

[11] Julio Guerra, Shaghayegh Sahebi, Yu-Ru Lin, and Peter Brusilovsky. [n. d.]. The Problem Solving Genome: Analyzing Sequential Patterns of Student Work with Parameterized Exercises. In *Proceedings of the 7th International Conference on Educational Data Mining, EDM 2014*. 153–160.

[12] Sunil Kumar Gupta, Dinh Phung, Brett Adams, and Svetha Venkatesh. 2013. Regularized nonnegative shared subspace learning. *Data mining and knowledge discovery* 26, 1 (2013), 57–97.

[13] Sherif Halawa, Daniel Greene, and John Mitchell. 2014. Dropout prediction in MOOCs using learner activity features. In *Proceedings of the Second European MOOC Stakeholder Summit*. 58–65.

[14] John D Hansen and Justin Reich. 2015. Socioeconomic status and MOOC enrollment: enriching demographic information with external datasets. In *Proceedings of the Fifth International Conference on Learning Analytics and Knowledge*. ACM.

[15] R.A. Harshman. 1970. Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multimodal factor analysis. (1970).

[16] Neil T Heffernan and Cristina Lindquist Heffernan. 2014. The ASSISTments Ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education* 24, 4 (2014), 470–497.

[17] Roya Hosseini, Peter Brusilovsky, Michael Yudelson, and Arto Hellas. 2017. Stereotype modeling for Problem-Solving performance predictions in MOOCs and traditional courses. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. ACM, 76–84.

[18] Chaoran Huang, Lina Yao, Xianzhi Wang, Boualem Benatallah, Shuai Zhang, and Manqing Dong. 2018. Expert recommendation via tensor factorization with regularizing hierarchical topical relationships. In *Proceedings of the International Conference on Service-Oriented Computing*. Springer, 373–387.

[19] Katy Jordan. 2014. Initial trends in enrolment and completion of massive open online courses. *The International Review of Research in Open and Distributed Learning* 15, 1 (2014).

[20] Rogier Kievit, Willem Eduard Frankenhuis, Lourens Waldorp, and Denny Borsboom. 2013. Simpson's paradox in psychological science: a practical guide. *Frontiers in psychology* 4 (2013), 513.

[21] Byung-Hak Kim, Ethan Vizitei, and Varun Ganapathi. 2018. GritNet 2: Real-Time Student Performance Prediction with Domain Adaptation. *arXiv preprint arXiv:1809.06686* (2018).

[22] Hannah Kim, Jaegul Choo, Jingu Kim, Chandan K Reddy, and Haesun Park. 2015. Simultaneous discovery of common and discriminative topics via joint nonnegative matrix factorization. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 567–576.

[23] Marius Kloft, Felix Stiehler, Zhilin Zheng, and Niels Pinkwart. 2014. Predicting MOOC dropout over weeks using machine learning methods. In *Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs*.

[24] Tamara Gibson Kolda. 2006. *Multilinear operators for higher-order decompositions*. Technical Report. Sandia National Laboratories.

[25] Stephan Lorenzen, Niklas Hjuler, and Stephen Alstrup. 2018. Tracking Behavioral Patterns among Students in an Online Educational System. In *Proceedings of the 11th International Conference on Educational Data Mining, EDM 2018, Buffalo, NY, USA, July 15-18, 2018*.

[26] Jorge Maldonado-Mahauad, Mar Pérez-Sanagustín, Pedro Manuel Moreno-Marcos, Carlos Alario-Hoyos, Pedro J Muñoz-Merino, and Carlos Delgado-Kloos. 2018. Predicting Learners' Success in a Self-paced MOOC Through Sequence Patterns of Self-regulated Learning. In *Proceedings of the European Conference on Technology-Enhanced Learning*. Springer, 355–369.

[27] Charalampos Mavroforakis, Isabel Valera, and Manuel Gomez-Rodriguez. 2017. Modeling the Dynamics of Learning Activity on the Web. In *Proceedings of the 26th International Conference on World Wide Web*. 1421–1430.

[28] Saurabh Nagrecha, John Z Dillon, and Nitesh V Chawla. 2017. MOOC dropout prediction: lessons learned from making pipelines interpretable. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 351–359.

[29] Ivan V Oseledets. 2009. Compact matrix form of the d-dimensional tensor decomposition. In *Proceedings of the International Symposium on Nonlinear Theory and its Applications*.

[30] Ivan V Oseledets and Eugene E Tyrtyshnikov. 2009. Breaking the curse of dimensionality, or how to use SVD in many dimensions. *SIAM Journal on Scientific Computing* 31, 5 (2009), 3744–3759.

[31] Alp Özdemir, Mark A Iwen, and Selin Aviyente. 2016. Multiscale tensor decomposition. In *Proceedings of the 2016 50th Asilomar Conference on Signals, Systems and Computers*. IEEE, 625–629.

[32] Alp Özdemir, Mark A Iwen, and Selin Aviyente. 2017. Multiscale Analysis for Higher-order Tensors. *arXiv preprint arXiv:1704.08578* (2017).

[33] Laura Pappano. 2012. The Year of the MOOC. *The New York Times* 2, 12 (2012).

[34] Judea Pearl et al. 2009. Causal inference in statistics: An overview. *Statistics surveys* 3 (2009), 96–146.

[35] Anh Huy Phan and Andrzej Cichocki. 2010. Tensor decompositions for feature extraction and classification of high dimensional datasets. *Nonlinear theory and its applications, IEICE* 1, 1 (2010).

[36] Jiezhong Qiu, Jie Tang, Tracy Xiao Liu, Jie Gong, Chenhui Zhang, Qian Zhang, and Yufei Xue. 2016. Modeling and predicting learning behavior in MOOCs. In *Proceedings of the 9th ACM international conference on web search and data mining*.

[37] Steffen Rendle. 2010. Factorization machines. In *Proceedings of the 10th International Conference on Data Mining (ICDM)*. IEEE, 995–1000.

[38] Giuseppe Ricci, Marco de Gemmis, and Giovanni Semeraro. 2014. Mathematical methods of tensor factorization applied to recommender systems. In *New Trends in Databases and Information Systems*. Springer, 383–388.

[39] Shaghayegh Sahebi, Yu-Ru Lin, and Peter Brusilovsky. 2016. Tensor factorization for student modeling and performance prediction in unstructured domain. In *Proceedings of the 9th International Conference on Educational Data Mining*.

[40] John Saint, Dragan Gašević, and Abelardo Pardo. 2018. Detecting Learning Strategies Through Process Mining. In *Proceedings of the European Conference on Technology-Enhanced Learning*. Springer, 385–398.
[41] Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 650–658.
[42] Age Smilde, Rasmus Bro, and Paul Geladi. 2005. *Multi-way analysis: applications in the chemical sciences*. John Wiley & Sons.
[43] Panagiotis Symeonidis. 2016. Matrix and tensor decomposition in recommender systems. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 429–430.
[44] Feng Wang and Li Chen. 2016. A Nonlinear State Space Model for Identifying At-Risk Students in Open Online Courses. In *Proceedings of the 9th International Conference on Educational Data Mining, EDM 2016, Raleigh, North Carolina, USA, June 29 - July 2, 2016*. 527–532.
[45] Wei Wang, Han Yu, and Chunyan Miao. 2017. Deep Model for Dropout Prediction in MOOCs. In *Proceedings of the 2nd International Conference on Crowd Science and Engineering*. ACM, 26–32.
[46] Xidao Wen, Yu-Ru Lin, and Konstantinos Pelechrinis. 2016. Pairfac: Event analytics through discriminant tensor factorization. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM.
[47] Xidao Wen, Yu-Ru Lin, and Konstantinos Pelechrinis. 2018. Event Analytics via Discriminant Tensor Factorization. *ACM Trans. Knowl. Discov. Data* (2018).
[48] Jibing Wu, Zhifei Wang, Yahui Wu, Lihua Liu, Su Deng, and Hongbin Huang. 2017. A Tensor CP decomposition method for clustering heterogeneous information networks via stochastic gradient descent algorithms. *Scientific Programming* 2017 (2017).
[49] Wanli Xing and Dongping Du. 2018. Dropout Prediction in MOOCs: Using Deep Learning for Personalized Intervention. *Journal of Educational Computing Research* (2018).
[50] Tiantian Zhang and Bo Yuan. 2016. Visualizing MOOC User Behaviors: A Case Study on XuetangX. In *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, 89–98.

Dynamic Ensemble of Contextual Bandits to Satisfy Users' Changing Interests

Qingyun Wu
University of Virginia
Charlottesville, VA , USA
qw2ky@virginia.edu

Huazheng Wang
University of Virginia
Charlottesville, VA , USA
hw7ww@virginia.edu

Yanen Li
Snap Inc.
Los Angeles, CA, USA
yanen.li@snap.com

Hongning Wang
University of Virginia
Charlottesville, VA, USA
hw5x@virginia.edu

ABSTRACT

Recommender systems have to handle a highly non-stationary environment, due to users' fast changing interests over time. Traditional solutions have to periodically rebuild their models, despite high computational cost. But this still cannot empower them to automatically adjust to abrupt changes in trends caused by timely information. It is important to note that the changes of reward distributions caused by a non-stationary environment can also be *context dependent*. When the change is orthogonal to the given context, previously maintained models should be reused for better recommendation prediction.

In this work, we focus on contextual bandit algorithms for making adaptive recommendations. We capitalize on the unique context-dependent property of reward changes to conquer the challenging non-stationary environment for model update. In particular, we maintain a dynamic ensemble of contextual bandit models, where each bandit model's reward estimation quality is monitored regarding given context and possible environment changes. Only the admissible models to the current environment will be used for recommendation. We provide a rigorous upper regret bound analysis of our proposed algorithm. Extensive empirical evaluations on both synthetic and three real-world datasets confirmed the algorithm's advantage against existing non-stationary solutions that simply create new models whenever an environment change is detected.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Theory of computation** → **Sequential decision making**; **Online learning algorithms**; **Regret bounds**;

KEYWORDS

Non-stationary bandits; recommender systems; regret analysis

ACM Reference Format:
Qingyun Wu, Huazheng Wang, Yanen Li, and Hongning Wang. 2019. Dynamic Ensemble of Contextual Bandits to Satisfy Users' Changing Interests. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313727

This paper is published under the Creative Commons Attribution 4.0 International (CC BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution. In case of republication, reuse, etc., the following attribution should be used: "Published in WWW2019 Proceedings © 2019 International World Wide Web Conference Committee, published under Creative Commons CC BY 4.0 License."
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313727

1 INTRODUCTION

The overwhelming volume of online content make personalized recommendation an indispensable component in modern information service systems. Traditional solutions, including content-based filtering [17, 26], collaborative filtering [6, 30], and hybrid approaches [22], provide recommendations by leveraging users' interests as demonstrated in their past activities. However, in many practical applications, such as news recommendation, both content popularity and user interests evolve frequently over time, which make the traditional offline learning approaches incompetent [18].

In recent years, multi-armed bandits, and more specifically contextual bandits, have become a referenced online learning solution to deal with this dynamic nature of recommendations [1, 20, 27, 32, 33, 36, 37]. Contextual bandit solutions explore the unknowns by collecting users' feedback in real time to estimate the utility/reward of new content with available side-information or context information. They provide a principled way to find optimal trade-offs between exploration and exploitation [3, 4], and have been successfully deployed in many important practical scenarios [20, 23].

However, most existing contextual bandit algorithms assume a stationary environment [4, 20], where the expected reward on each arm is drawn from an unknown yet fixed reward mapping function based on the given context. This assumption, however, rarely holds in many real-world applications, where the underlying reward mapping undergoes slow or abrupt changes due to various factors. For example, Liu et al. [24] reported that readers' preferences over news articles shift with time and events in Google News. By analyzing more than 20,000 twitter users over a four-months period, Abel et al. [2] found that the interests of individual users into a topic evolve differently over time. More importantly, the duration during which users are interested in an event-like topic differs significantly among each other. In other words, users' interest change over time but the change is unknown beforehand.

In this work, we focus on the setting where there are unknown (to the learner) and abrupt changes in terms of user preferences. Between consecutive changes, the reward distribution remains stationary yet unknown, i.e., *piecewise stationary*. Existing bandit algorithms address such a dynamic environment by either introducing a forgetting mechanism to downweight historical observations [12, 15], or creating a bandit model for each newly detected stationary period [7, 14, 35, 38]. These strategies, nevertheless, failed to recognize an important property of this non-stationary environment: the changes of user interests can be *context dependent*. Even though one's interest could change frequently, his/her preference on a particular type of items might be stable over a longer period

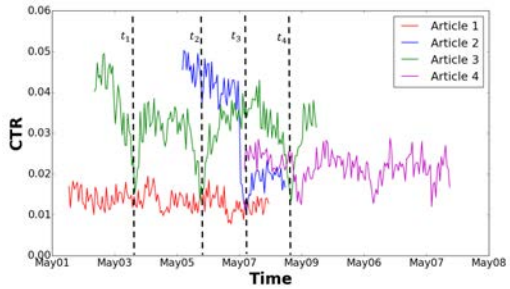


Figure 1: Real-time click-through-rate of four sample news articles collected from Yahoo user click log dataset [20, 21].

of time. For example, in the news recommendation scenario mentioned above, users' preferences over sports news may change with sport seasons. However, at the same time, their preferences over political news may stay stationary, independent of the sport season.

This phenomenon can be verified in Figure 1, where we collect real-time click-through-rate (CTR) of four sample news articles from the public Yahoo front-page user click log [20, 21] over a week's period. In the figure, each point denotes average CTR over 4000 log records. We can clearly observe the changes of user interest on article 2 at time t_3 and on article 3 at time t_1, t_2 and t_4 in this period. In the meanwhile, the CTR of article 1 and 4 remain quite stable, i.e., the expected reward for recommending these two articles remain unchanged. As a result, the experience recorded in old models can still be used to make accurate reward estimations for these two articles. On the contrary, strategies that discount observations or abandon the "old" models must regain confidence in their newly estimated parameters, which many lead to higher regret due to redundant explorations. The key is thus to recognize the reward change in a per-arm basis regarding the context.

To capitalize on such a unique property of the changing environment, we develop our contextual bandit algorithm that adapts its arm selection and model update strategy with regard to users' interest changes in a context-dependent manner. Our solution consists of a dynamic set of contextual bandit models, collectively referred to as *bandit experts*, which are maintained to estimate the underlying reward distribution. In the meanwhile, we monitor each bandit expert's reward estimation quality regarding specific context through another bandit model, referred to as *bandit auditors*. The auditor predicts whether a bandit expert is 'admissible' to a specific arm under the given context. At each round of interaction, an ensemble of admissible bandit experts is created to estimate the reward of each arm; and the arm with the highest upper confidence bound of estimated reward will be selected. The acquired feedback is used to update all admissible bandit experts for this chosen arm, and their corresponding auditors. When no admissible bandit expert exists, a new bandit expert will be created and added to the set before evaluating the recommendation candidates.

We rigorously prove a sublinear upper regret bound of our proposed algorithm, which guarantees that the number of sub-optimal recommendations from our algorithm reduces rapidly over time. We also show that if one fails to model the context dependent changes, a much worse upper regret bound is inevitable.Extensive empirical evaluations in one synthetic and three real-world datasets also confirmed the effectiveness of the proposed algorithm, especially in handling the changing popularity and user preferences in practice.

2 RELATED WORK

In this work, we focus on the stochastic setting of multi-armed bandit problems [29]. Under this setting, various exploration strategies have been proposed to conquer the exploit-explore dilemma, such as upper confidence bound [3, 4, 20], epoch greedy [19] and posterior sampling [9, 16]. However, most of these algorithms assume a static environment, which is often violated in practice. A number of bandit models hence have been proposed to deal with non-stationary environments, and there are mainly two types of environment changes studied: gradual changes and abrupt changes.

For gradual changes, Whittle [34] introduced the restless bandits, where the states of arms can change in each step according to a stochastic transition function. But this setting is notoriously intractable [5, 25, 31].

Our work concerns abrupt changes of environment, where most solutions only focus on context-free bandits. Hartland et al. [15], and Garivier and Moulines [12] proposed a γ-Restart algorithm and a discounted-UCB algorithm, in which a discount factor is introduced to exponentially decay the effect of past observations. Both algorithms need properly designed hyper-parameters, such as the discount factor. Yu and Mannor [38] proposed a windowed mean-shift detection algorithm to detect the changes in the environment. They assume that at each iteration, the agent can not only get feedback from the selected arms but also query a subset of arms for additional observations, which would be very expensive in practice. Raj and Kalyani proposed discounted Thompson Sampling in [28]. Recently Cao et al. [7] proposed a UCB based algorithm with change detection module to detect changes and restart exploration accordingly. Liu et al. [10] proposed to use cumulative sum and Page-Hinkley test to detect sudden changes in the environment. A sublinear upper regret bound is proved for a simplified Bernoulli bandit environment with a strong detectability assumption. All of the aforementioned solutions are non-contextual bandit solutions, which are incapable to leverage the rich side information in real-world applications.

Some recent works have realized the lack of contextual bandits in a non-stationary environment, and extended corresponding solutions. Hariri et al. [14] proposed a contextual Thompson sampling algorithm with a change detection module, which involves iteratively applying a combination of cumulative sum charts and bootstrapping to capture potential changes. But due to its empirical nature, no theoretical property is known about it. Wu et al. [35] developed a two-level hierarchical bandit algorithm, which detects and adapts to changes in the environment by maintaining a suite of contextual bandit models. Regret analysis is provided under a strong assumption about the change. However, to the best of our knowledge, among the existing non-stationary bandit solutions, no work utilizes the context dependent property of reward changes in a piece-wise stationary environment. Hence, none of them is able to exploit the existence of *context dependent changes*. Our algorithm

avoids the unnecessary penalties in regret incurred by indiscriminately dismissing existing bandit models and tries to reuse the existing models when judged 'admissible.'

3 METHODOLOGY

In this section, we first introduce our notations and assumptions about the non-stationary environment, then illustrate our proposed algorithm, followed with a rigorous regret bound analysis.

3.1 Problem Setups

In a multi-armed bandit problem, a learner sequentially selects an arm a_t from a candidate pool $\mathcal{A} = \{a_1, a_2, ..., a_k\}$ to interact with the environment, and receives the corresponding reward r_{a_t}. The goal of the learner is to maximize its accumulated reward over a finite time horizon T. In a stochastic contextual bandit setting, each candidate arm a is associated with a feature vector $\mathbf{x}_a$ (assume $\|\mathbf{x}_a\|_2 \leq 1$ without loss of generality), referred as context. The corresponding reward r_a is determined by the context vector and an underlying bandit parameter $\boldsymbol{\theta}^*$ (assume $\|\boldsymbol{\theta}^*\|_2 \leq 1$ without loss of generality). When applying to the recommendation scenario, the bandit parameter $\boldsymbol{\theta}^*$ can be interpreted as the underlying parameter that controls users' interests and $\mathbf{x}_a$ is the available context information about the candidate item a. Most existing contextual bandit algorithms consider $\boldsymbol{\theta}^*$ as constant over time [1, 20], which imposes a strong stationary assumption about the environment.

In this work, firstly, we relax this stationary assumption by allowing abrupt changes in $\boldsymbol{\theta}^*$: the ground-truth $\boldsymbol{\theta}^*$ changes arbitrarily at unknown time points, but remains constant between any two consecutive change points [12, 14, 15] as follows:

$$\underbrace{r_0, r_1, \cdots, r_{t_{c_1}-1}}_{\text{governed by } \boldsymbol{\theta}^*_{c_0}}, \underbrace{r_{t_{c_1}}, r_{t_{c_1}+1}, \cdots, r_{t_{c_2}-1}}_{\text{governed by } \boldsymbol{\theta}^*_{c_1}}, \cdots, \underbrace{r_{t_{c_{\Gamma_T}}}, r_{t_{c_{\Gamma_T}}+1}, \cdots, r_T}_{\text{governed by } \boldsymbol{\theta}^*_{c_{\Gamma_T}}}$$

where the change points $\{t_c\}_{c=c_1}^{c_{\Gamma_T}}$ and the corresponding bandit parameters $\{\boldsymbol{\theta}^*_c\}_{c=c_0}^{c_{\Gamma_T}}$ are unknown to the learner. We only assume there are at most Γ_T change points in the environment up to time T, with $\Gamma_T \ll T$. To simplify the discussion of our developed algorithm, a linear reward structure is postulated, but it can be readily extended to more complicated structures, such as generalized linear models [11]. Specifically, we have $r_{a,t} = \mathbf{x}_a^\intercal \boldsymbol{\theta}^*_t + \eta_t$, where η_t is Gaussian noise drawn from $N(0, \sigma^2)$.

Secondly, we capitalize on the unique property that in a contextual bandit setting the changes of reward distribution are context-dependent. Thus we categorize arms into two types, *change-invariant* and *change-sensitive*, between any two stationary periods. For stationary periods i and j with their ground-truth bandit parameters $\boldsymbol{\theta}^*_i$ and $\boldsymbol{\theta}^*_j$, the arm that satisfies $|\mathbf{x}_a^\intercal \boldsymbol{\theta}^*_i - \mathbf{x}_a^\intercal \boldsymbol{\theta}^*_j| \leq \Delta_L$ (with $\Delta_L > 0$) is referred as a change-invariant arm; otherwise as a change-sensitive arm. Δ_L is a parameter to introduce flexibility and accommodate stochastic noise, which relaxes the requirement that the change has to be completely orthogonal to the context vector of a change-invariant arm. This makes our problem setting more general than those in [10, 38]. An illustration of these two types of arms is provided in Figure 2: when the ground-truth bandit parameter changes from $\boldsymbol{\theta}^*_c$ to $\boldsymbol{\theta}^*_{c+1}$, although there are *change-sensitive arms*, for example arm a_1, whose expected reward changes

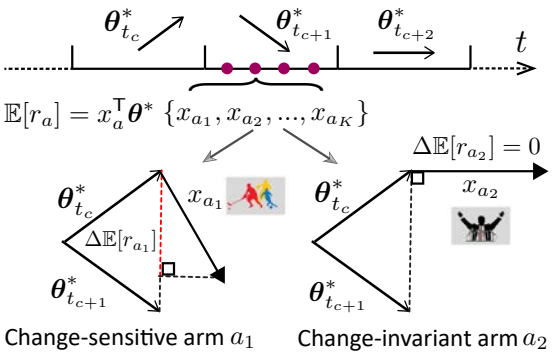


Figure 2: An illustrative example of context-dependent changes in a piecewise stationary environment. A linear reward assumption is postulated to simplify the illustration.

dramatically, there are also *change-invariant arms*, for example arm a_2, whose context vectors are orthogonal to the change, i.e., $\mathbf{x}_a^\intercal(\boldsymbol{\theta}^*_c - \boldsymbol{\theta}^*_{c+1}) = 0 < \Delta_L$, so that their expected reward does not change significantly even after the environment changes. For example, mapping it back to our previous news recommendation example, if the reward change in a user was caused by the change of sports season, the sports news and political news could be considered as the change-sensitive and change-invariant arms respectively for this user.

To differentiate the actual reward change from stochastic noise, we impose the following assumption about the non-stationary environment, which characterizes the detectability of reward changes between the stationary period i and j,

Assumption 1. *Among the change-sensitive arms between stationary period i and j, there are at least ρ portion of them satisfying $|\mathbf{x}_a^\intercal \boldsymbol{\theta}^*_i - \mathbf{x}_a^\intercal \boldsymbol{\theta}^*_j| > \Delta_H$.*

This assumption requires that between any two stationary periods, there are a number of change-sensitive arms undergo perceivable reward changes, which differentiate these two periods.

3.2 A Dynamic Ensemble of Bandits

In the non-stationary environment specified above, where the changes of users' interests occasionally happen at unknown time points, new bandit models should be rebuilt in accordance with the changes of underlying user interest. In addition, the possible existence of change-invariant arms urges us to reuse the bandit models estimated for those earlier periods, so that more accurate reward estimation on such arms can be achieved sooner so as to obtain reduced regret in a new stationary period.

In order to achieve these goals, three challenges have to be addressed: 1) as the changes of environment are unknown to the learner, how to detect the potential change of user interests and create new bandit models to account for the change-sensitive arms in a new environment? 2) as the reward changes in each arm are context-dependent, how to recognize the change-invariant arms at current period such that experience from old models can be fully utilized? and 3) which arm to choose given multiple bandit models might exist at the same time? To avoid any potential ambiguity in our later discussions, we refer to the contextual bandit models created for reward estimation as *bandit experts*.

We address the first two challenges by creating a companion bandit model for each bandit expert to monitor its reward estimation quality. We refer to this companion bandit model as a *bandit auditor*. In a nutshell, a bandit expert, who works with the context features and reward collected from its chosen arms, is responsible for reward estimation regarding its corresponding environment. Its companion bandit auditor works with the context features of the expert's choices and the observed prediction errors from the bandit expert. The auditor is responsible for assessing the expert's prediction accuracy. At each round, every bandit auditor evaluates whether the monitored bandit expert is *admissible* to make an accurate reward estimation for a given arm, with respect to potential environment changes. A bandit expert being identified as admissible to a particular arm indicates with a high probability that, either 1) no change has happened since the creation of this bandit expert, or 2) the environment has changed but the arm is change-invariant between this bandit expert's estimated reward distribution and the current period's underlying reward distribution. This addresses the second challenge. When no admissible bandit expert exists for a given arm, it is thus highly likely to be a change-sensitive arm in a new environment, and a new bandit expert is needed. This addresses the first challenge.

To address the third challenge, at each round of interaction, following the principle of optimism in the face of uncertainty [1, 20], an arm is chosen by the upper confidence bound of reward estimation based on an ensemble of all its admissible bandit experts. The acquired feedback for the selected arm/item is used to update all corresponding bandit experts and their auditors. We name the resulting bandit algorithm as Dynamic Ensemble of Bandit Experts, or DenBand in short. We describe DenBand in Algorithm 1 [1] and discuss the key components of it in details as follows.

Bandit Expert. Define t_m as the time when bandit expert m is created. Each bandit expert m maintains an estimated bandit parameter $\hat{\boldsymbol{\theta}}_t(m)$ for the stationary period at t_m. Define $\mathcal{I}_t^{\theta}(m)$ as a set of timestamps when observation $(\mathbf{x}_{a_i}, r_{a_i,i})$ is assigned to the bandit expert m for model update till time t (in line 24-25 of Algorithm 1). $r_{a_i,i}$ is the observed reward on arm a_i at time i. Because of our linear reward structure, $\hat{\boldsymbol{\theta}}_t(m)$ can be readily estimated by $\hat{\boldsymbol{\theta}}_t(m) = \mathbf{A}_t^{-1}(m)\mathbf{b}_t(m)$, in which $\mathbf{A}_t(m) = \lambda\mathbf{I} + \sum_{i\in\mathcal{I}_t^{\theta}(m)} \mathbf{x}_{a_i}\mathbf{x}_{a_i}^{\mathsf{T}}$, $\mathbf{I}$ is a $d \times d$ identity matrix, λ is the regularization coefficient in the least square regression, and $\mathbf{b}_t(m) = \sum_{i\in\mathcal{I}_t^{\theta}(m)} \mathbf{x}_{a_i} r_{a_i,i}$.

Bandit Auditor. Denote the reward estimation error of bandit expert m on arm a at time t as $e_{a,t}(m) = \hat{r}_{a,t}(m) - r_{a,t}$, in which $\hat{r}_{a,t}(m) = \mathbf{x}_a^{\mathsf{T}}\hat{\boldsymbol{\theta}}_t(m)$. We have $\mathbb{E}[e_{a,t}(m)] = \mathbf{x}_a^{\mathsf{T}}(\hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}_t^*)$, which is referred as 'badness' of bandit expert m on arm a at time t and leads to a linear structure for badness estimation. We create a new bandit model with the target parameter for estimation as $\boldsymbol{\beta}_t^*(m) = \hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}_t^*$, and refer to it as the bandit auditor of bandit expert m. We maintain and update the bandit auditors in a similar manner as that in bandit experts. Denote $\mathcal{I}_t^{\beta}(m)$ as a set of timestamps when observation $\left(\mathbf{x}_{a_i}, e_{a_i,i}(m)\right)$ is assigned to the bandit auditor for bandit expert m up to time t (line 21-23 in Algorithm 1). The bandit auditor estimates $\boldsymbol{\beta}_t^*(m)$ by $\hat{\boldsymbol{\beta}}_t(m) = \mathbf{C}_t^{-1}(m)\mathbf{d}_t(m)$, in which

[1] Open source implementation of DenBand can be found in https://github.com/huazhengwang/BanditLib

Algorithm 1 Dynamic Ensemble of Bandit Experts (DenBand)

1: **Inputs:** $\alpha \in \mathbb{R}_+$, $\lambda > 0$, δ_1, $\delta_2 \in (0, 1)$, τ, Δ_L
2: **Initialize:** Create and initialize bandit expert m: $\mathbf{A}_1(m) = \lambda\mathbf{I}$, $\mathbf{b}_1(m) = \mathbf{0}$, $\hat{\boldsymbol{\theta}}_1(m) = \mathbf{0}$, and its auditor: $\mathbf{C}_1(m) = \lambda\mathbf{I}$, $\mathbf{d}_1(m) = \mathbf{0}$, $\hat{\boldsymbol{\beta}}_1(m) = \mathbf{0}$. Initialize the bandit expert set $\mathcal{M}_1 = \{m\}$
3: **for** $t = 1$ to T **do**
4: **for** $a \in \mathcal{A}_t$ **do**
5: Create an admissible model set for arm a: $\mathcal{M}_t^a = \emptyset$
6: **for** $m \in \mathcal{M}_t$ **do**
7: $\hat{e}_{a,t}(m) = \mathbf{x}_a^{\mathsf{T}}\hat{\boldsymbol{\beta}}_t(m)$
8: Compute $B_{a,t}^{\beta}(m)$ and $B_{a,t}^{\theta}(m)$ by Eq (1) and (2)
9: **if** $|\hat{e}_{a,t}(m)| < B_{a,t}^{\theta}(m) + B_{a,t}^{\beta}(m) + \Delta_L$ **then**
10: Add m into $\mathcal{M}_t^a$
11: **end if**
12: **end for**
13: **if** $|\mathcal{M}_t^a| = 0$ **then**
14: Create and initialize a new bandit expert m and its auditor as in Line 2; Add m to $\mathcal{M}_t^a$ and $\mathcal{M}_t$
15: **end if**
16: Compute $\text{UCB}_{a,t}$ of arm a with bandit experts in $\mathcal{M}_t^a$
17: **end for**
18: Select an arm by $a_t = \arg\max_{a\in\mathcal{A}} \text{UCB}_{a,t}$
19: Observe reward $r_{a_t,t}$
20: **for** $m \in \mathcal{M}_t^{a_t}$ **do**
21: $\hat{r}_{a_t,t}(m) = \mathbf{x}_{a_t}^{\mathsf{T}}\hat{\boldsymbol{\theta}}_t(m)$, $e_{a_t,t}(m) = \hat{r}_{a_t,t}(m) - r_{a_t,t}$
22: Update $\{e_{a_i,i}\}_{i\in\mathcal{I}_t^{\beta}(m)}$ according to $\hat{\boldsymbol{\theta}}_t(m)$, and keep the size of $\mathcal{I}_t^{\beta}(m)$ to τ
23: $\mathbf{C}_{t+1}(m) = \mathbf{C}_t(m) + \mathbf{x}_{a_t}\mathbf{x}_{a_t}^{\mathsf{T}}$, $\mathbf{d}_{t+1}(m) = \sum_{i\in\mathcal{I}_t^{\beta}(m)} \mathbf{x}_{a_i} e_{a_i,i}$
24: **if** $m \in \mathcal{M}_t^{a_t}$ and $|\hat{r}_{a_t,t}(m) - r_{a_t,t}| \le B_{a_t,t}^{\theta}(m) + \Delta_L + \epsilon$ **then**
25: $\mathbf{A}_{t+1}(m) = \mathbf{A}_t(m) + \mathbf{x}_{a_t}\mathbf{x}_{a_t}^{\mathsf{T}}$, $\mathbf{b}_{t+1}(m) = \mathbf{b}_t(m) + \mathbf{x}_{a_t} r_{a_t,t}$
26: **else**
27: $\mathbf{A}_{t+1}(m) = \mathbf{A}_t(m)$, $\mathbf{b}_{t+1}(m) = \mathbf{b}_t(m)$
28: **end if**
29: $\hat{\boldsymbol{\theta}}_{t+1}(m) = \mathbf{A}_{t+1}(m)^{-1}\mathbf{b}_{t+1}(m)$
30: $\hat{\boldsymbol{\beta}}_{t+1}(m) = \mathbf{C}_{t+1}(m)^{-1}\mathbf{d}_{t+1}(m)$
31: **end for**
32: **end for**

$\mathbf{C}_t(m) = \lambda\mathbf{I} + \sum_{i\in\mathcal{I}_t^{\beta}(m)} \mathbf{x}_{a_i}\mathbf{x}_{a_i}^{\mathsf{T}}$ and $\mathbf{d}_t(m) = \sum_{i\in\mathcal{I}_t^{\beta}(m)} \mathbf{x}_{a_i} e_{a_i,i}(m)$. Intuitively, the bandit auditor for expert m evaluates whether an arm a at time t is change-invariant to the reward distributions specified by $\boldsymbol{\theta}_t^*$ and $\boldsymbol{\theta}_{t_m}^*$. The definition of badness requires us to update $e_{a,t}(m)$ of all observations in $\mathcal{I}_t^{\beta}(m)$ whenever $\hat{\boldsymbol{\theta}}_t(m)$ is updated or $\boldsymbol{\theta}_t^*$ is changed. But as the environment change is unknown to the learner, this update is infeasible. We decide to only accumulate the most recent τ observations in $\mathcal{I}_t^{\beta}(m)$ for auditor update, and prove this still provides us a high probability bound of each bandit auditor's badness estimation in our regret analysis,

$$|\hat{e}_{a,t}(m) - \mathbb{E}[e_{a,t}(m)]| \le B_{a,t}^{\beta}(m) \tag{1}$$

where $B_{a,t}^{\beta}(m) = \left(\sigma^2\sqrt{d\ln(\frac{\lambda+|\mathcal{I}_t^{\beta}(m)|}{\lambda\delta_2})} + \sqrt{\lambda}\right)\|\mathbf{x}_a\|_{\mathbf{C}_t^{-1}(m)}$.

Bandit Expert Selection. Based on Eq (1) and our badness definition, we have $|\hat{e}_{a,t}(m)| \le \mathbb{E}[e_{a,t}(m)] + B_{a,t}^{\beta}(m) \le |\mathbf{x}_a^{\mathsf{T}}\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^{\mathsf{T}}\boldsymbol{\theta}_{t_m}^*| + |\mathbf{x}_a^{\mathsf{T}}\boldsymbol{\theta}_{t_m}^* - \mathbf{x}_a^{\mathsf{T}}\boldsymbol{\theta}_t^*| + B_{a,t}^{\beta}(m)$. The first part on the right-hand

side of this inequality is the reward estimation quality of bandit expert m, which can be bounded by,

$$|\mathbf{x}_a\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a\boldsymbol{\theta}^*_{t_m}| \leq B^{\theta}_{a,t}(m,a) \tag{2}$$

where $B^{\theta}_{a,t}(m,a) = \left(3\sigma^2\sqrt{d\ln(\frac{\lambda+|\mathcal{I}^{\theta}_t(m)|}{\lambda\delta_1})} + \sqrt{\lambda}\right)\|\mathbf{x}_a\|_{\mathbf{A}^{-1}_t(m)}$. We should note that this bound is different from those for classical linear bandit algorithms (e.g., [1]), since it also has to account for the possible contamination from change-invariant arms (as we do not restrict Δ_L to zero). The second part on the right-hand side can be bounded by Δ_L, if no change has happened since t_m or the arm a is change-invariant between t_m and t. As a result, the condition $|\hat{e}_{a,t}(m)| < B^{\theta}_{a,t}(m) + B^{\beta}_{a,t}(m) + \Delta_L$ in line 9 of Algorithm 1 determines if the bandit expert m is admissible to arm a at time t (supported by Lemma 1 and 2 in Section 3.3). When there is no admissible bandit expert for arm a, a new bandit expert needs to be created to account for the detected change of environment (line 13-15 in Algorithm 1).

Arm Selection. We appeal to the Upper Confidence Bound (UCB) principle [1, 20] to select an arm from the candidate arm pool (line 18 of Algorithm 1). With the above bandit expert selection strategy, for arm a at time t, we might collect a set of admissible bandit experts $\mathcal{M}^a_t$, and each admissible bandit expert m gives us an upper confidence bound of reward estimation for arm a: $\text{UCB}_{a,t}(m) = \mathbf{x}^{\intercal}_a\hat{\boldsymbol{\theta}}_t(m) + B^{\theta}_{a,t}(m)$. Therefore, we need to integrate $\text{UCB}_{a,t}(m)$ from multiple bandit experts (line 16 of Algorithm 1). We propose two strategies for this purpose, each of which has its own advantages; but both of them lead to the same upper regret bound (proved in Theorem 2 of Section 3.3).

Option 1: Average ensemble. For each arm, as in principle every admissible bandit expert is 'legitimate' to give a reward prediction for this arm, we compute an average upper confidence bound of reward based on all admissible expert models: $\text{UCB}_{a,t} = \frac{1}{|\mathcal{M}^a_t|}\sum_{m\in\mathcal{M}^a_t}\text{UCB}_{a,t}(m)$. This ensemble helps reduce variance in reward estimation among the admissible bandit experts, but might be disturbed by some least qualified bandit experts.

Option 2: Lower confidence bound of badness. Although every admissible bandit expert is guaranteed to have small enough badness to make an accurate reward prediction on the chosen arm, the uncertainty of their auditors' badness estimation may differ, which introduces another level of trade-off between exploitation and exploration in bandit expert selection. By applying the Lower Confidence Bound (LCB) principle (as we want to minimize the badness of chosen bandit experts), we select a bandit expert from $\mathcal{M}^a_t$ by the LCB of its auditor's estimated badness: $\tilde{m}_{a,t} = \arg\min_{m\in\mathcal{M}^a_t}\left(|\hat{e}_{a,t}(m)| - B^{\beta}_{a,t}(m)\right)$, and compute the UCB of arm a using the selected bandit expert $\tilde{m}_{a,t}$.

Model Update. Once the feedback $(\mathbf{x}_{a_t}, r_{a_t})$ is obtained from the environment on the selected arm a_t, we update the bandit experts and auditors in this arm's admissible model set (line 20-31 in Algorithm 1). In particular, we compare the acquired feedback against each bandit expert's estimation (line 24) to decide whether we should update the bandit expert to improve its reward estimation or the bandit auditor to improve its badness estimation. This decision comes from two factors that cause the observed reward estimation error: 1) large noise from the environment; 2) the arm is actually not change-invariant. Large noise may happen, but with a very small probability, as it follows a Gaussian distribution. Define $\epsilon = \sqrt{2}\sigma\text{erf}^{-1}(\delta_1 - 1)$, in which σ is the standard deviation of the Gaussian noise in reward feedback, and $\text{erf}(\cdot)$ is the Gauss error function. Violating the condition $|\hat{r}_{a,t}(m) - r_{a,t}| \leq B^{\theta}_{a,t}(m) + \Delta_L + \epsilon$ suggests that the chosen arm might not be change-invariant, such that the bandit expert should not be updated but the bandit auditor should be. This selective update strategy helps reduce erroneous observations in both bandit experts and auditors.

3.3 Regret Analysis

The accumulated (pseudo) regret of a bandit algorithm up to time T is formally defined as $R(T) = \sum_{t=1}^{T}(\mathbb{E}[r_{a^*_t,t}] - \mathbb{E}[r_{a_t,t}])$, in which a^*_t is the best arm according to the oracle at time t, and a_t is the arm selected by the algorithm to be evaluated. We first provide high probability bounds of bandit experts' reward estimation and bandit auditor's badness estimation in the following theorem.

Theorem 1 (Confidence Bounds). *We define S_{min} as the length of the shortest stationary period up to time T and $t_c(m)$ as the first change point in the environment after bandit expert m is created. When Assumption 1 is satisfied with $\Delta_H > 4\sqrt{\lambda} + \Delta_L$ and $\Delta_L \leq \frac{\sigma^2\sqrt{d\lambda\ln\frac{\lambda+T}{\lambda\delta_1}}}{T}$, and $1 < \tau \leq S_{min}\frac{\sigma^2\sqrt{d\lambda\ln\frac{\lambda+T}{\lambda\delta_1}}}{2\rho T}$, at time t for any $\delta_1 \in (0,1)$, $\delta_2 \in (0,1)$ and $\delta_3 = 1-(1-\delta_1)\left[(1-\delta_2)(1-\delta_1)\rho\right]^{\max\{t-t_c(m),0\}}$, with a probability at least $1-\delta_3$, for any arm a all the bandit experts in Algorithm 1 satisfy,*

$$|\mathbf{x}^{\intercal}_a\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}^{\intercal}_a\boldsymbol{\theta}^*_{t_m}| \leq B^{\theta}_{a,t}(m,a) \tag{3}$$

If there is no environment change in the past τ iterations, with a probability at least $1-\delta_2$, all bandit auditors at time t satisfy,

$$|\mathbf{x}^{\intercal}_a\hat{\boldsymbol{\beta}}_t(m) - \mathbf{x}^{\intercal}_a\boldsymbol{\beta}^*_t| \leq B^{\beta}_{a,t}(m,a) \tag{4}$$

And with a probability at least $1-\delta_3$, all the selected bandit experts in $\mathcal{M}^a_t$ for arm a satisfy,

$$|\mathbf{x}^{\intercal}_a\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}^{\intercal}_a\boldsymbol{\theta}^*_t| \leq \Delta_L + B^{\theta}_{a,t}(m) \tag{5}$$

This theorem specifies the threshold of minimum reward change for change-sensitive arms, i.e., Δ_H, which decides whether a change is detectable by our algorithm. We can further relax the threshold to $2B^{\theta}_{a,t}(m) + 2B^{\beta}_{a,t}(m) + \Delta_L$, which is shrinking over time and related to the current model uncertainties of the bandit expert and auditor. This would allow us to recognize more subtle reward changes across different stationary periods so as to improve the model estimation quality on the fly. On the other hand, Δ_L is the threshold deciding whether an arm is change-invariant with respect to two stationary periods. It is thus the resolution of our bandit experts in recognizing the ground-truth bandit parameters of their designated periods. Note, Δ_L is a parameter of the environment, rather than a parameter of our model. The parameter τ determines the number of most recent observations used for estimating the bandit auditors. Although a larger τ naturally leads to better badness estimation quality in the auditors, it cannot be arbitrarily large as it brings in out-of-date observations to the auditors. Theorem 1 imposes an upper bound of τ to guide practical use of our algorithm. Proofs this this theorem can be found in Appendix 7.1.

In the following two lemmas, we bound the probability of false negative and false positive selection of bandit experts, which proves the validity of our designed bandit expert selection strategy.

Lemma 1. *A false negative selection happens when the bandit expert m is not selected in its designated period or for the truly change-invariant arms to it in other periods. Denote the probability of a false negative selection as P_{FN}. At time t, we have $P_{FN} = \mathbb{P}\big(|\hat{e}_{a,t}(m)| > B^{\theta}_{a,t}(m) + B^{\beta}_{a,t}(m) + \Delta_L \,\big|\, |\mathbf{x}_a^\top \boldsymbol{\theta}^*_t - \mathbf{x}_a^\top \boldsymbol{\theta}^*_{t_m}| \le \Delta_L\big) \le (1-\delta_2)(1-\delta_3)$, in which δ_2 and δ_3 are defined in Theorem 1.*

Lemma 2. *A false positive selection happens when the environment has changed and the current arm is change-sensitive to a particular bandit expert m, but the bandit expert is mistakenly selected. Denote the probability of a false negative selection as P_{FP}. When Assumption 1 holds and $\Delta_H > 4\sqrt{\lambda} + \Delta_L$, we have at time $t > t_c(m)$, $P_{FP} = \mathbb{P}\big(|\hat{e}_{a,t}(m)| \le B^{\theta}_{a,t}(m) + B^{\beta}_{a,t}(m) + \Delta_L \,\big|\, |\mathbf{x}_a^\top \boldsymbol{\theta}^*_t - \mathbf{x}_a^\top \boldsymbol{\theta}^*_{t_m}| > \Delta_L\big) \le 1 - \big((1-\delta_1)(1-\delta_2)\rho\big)^{t-t_c(m)}$, in which δ_1 and δ_2 are defined in Theorem 1.*

Intuitively, a false negative selection of bandit experts happens when both the reward and badness estimations on a change-invariant arm exceed their confidence bounds; and a false positive selection happens when a change-sensitive arm undergoes substantial reward change but both the expert's reward estimation and the auditor's badness estimation still stay within their confidence bounds. Putting all these analyses together, we have the following regret bound for our proposed algorithm.

Theorem 2 (Regret bound of DenBand). *Under the same condition as stated in Theorem 1, and $(\delta_2 + \delta_3 - \delta_2\delta_3) \le \frac{1}{2S_{max}}$, using any bandit expert in $\mathcal{M}^a_t$ for UCB-based arm selection guarantees that with a probability at least $(1-\delta_3)(1-\delta_2)$ the expected accumulated regret of DenBand up to time T can be bounded as,*

$$R(T) \le 2\Gamma_T\Big(3\sigma^2\sqrt{d\ln\frac{\lambda+U_T}{\lambda\delta_1}} + \sqrt{\lambda}\Big)\sqrt{S_{max}d\ln(\lambda + \frac{S_{max}}{d})} + (\frac{1}{\rho}+2)\sigma^2\sqrt{d\lambda\ln\frac{\lambda+T}{\lambda\delta_1}}$$

where S_{max} is the length of the longest stationary period up to time T, Γ_T is the total number of environment changes till T, and $U_T = \max\{|\mathcal{I}^{\theta}_T(m)|\}_{m\in\mathcal{M}_T}$.

U_T is the maximum number of updates taken among any of the bandit experts; and it is clearly smaller than T. Hence the accumulated regret is in the order of $O(\Gamma_T\sqrt{S_{\max}\ln T\ln S_{\max}})$, which is arguably a very tight upper regret bound without any further assumption about the environment. Detailed proof and interpretation of Theorem 2 can be found in Appendix 7.2.

Theorem 2 provides a general upper regret bound of our DenBand in the order of $O(\Gamma_T\sqrt{S_{\max}\ln T\ln S_{\max}})$. To better interpret this upper regret bound under different circumstances, we consider the following two special environment cases. **Case 1:** When the changes are evenly (or almost evenly) distributed, i.e., $S_{\max} = S = \frac{T}{\Gamma_T}$, the resulting regret bound can be rewritten as $O(\sqrt{\Gamma_T T}\ln T)$, which matches the general lower regret bound in an abruptly changing environment proved in [12] without further assumptions about the environment. **Case 2:** When the distribution of changes are highly unbalanced such that there is a single very long stationary period and many very short stationary periods, by denoting the short periods with a superscript 's', the final upper regret bound can be rewritten as $O(\sqrt{S_{\max}\ln T\ln S_{\max}} + \Gamma_T\sqrt{S^s_{\max}\ln T\ln S^s_{\max}})$ where $S^s_{\max} \ll S_{\max}$. For example, when $S^s_{\max} \ll \frac{S_{\max}}{\Gamma_T^2}$, the regret can be further upper bounded by $O(\sqrt{S_{\max}\ln T\ln S_{\max}})$, which matches and is better than the upper regret bound of running a single contextual bandit over the whole period (as $S_{\max} < T$).

Our regret analysis in those two special cases supports the validity of DenBand, and we can further generalize our analysis of it under other reward assumptions. In particular, our theoretical analysis supports that any contextual bandit algorithm can be used as a bandit expert in DenBand, as long as its reward estimation error is bounded with a high probability, which corresponds to $B^{\theta}_t(m,a)$ in Eq (3). The overall regret of DenBand will only be a factor of the actual number of changes in the environment, which is arguably inevitable without further assumptions about the environment.

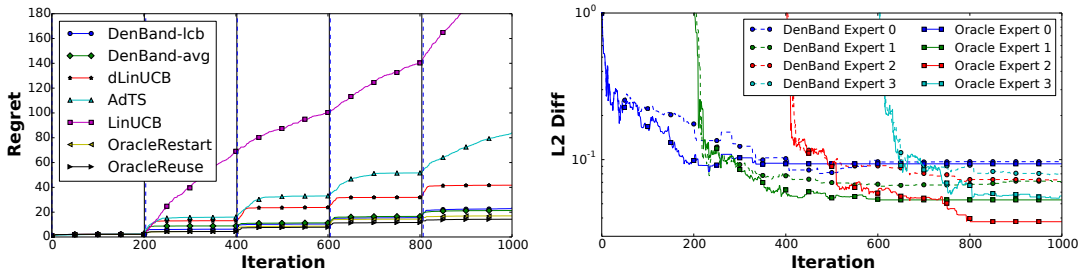
4 EXPERIMENTS

We tested DenBand on a comprehensive set of evaluation datasets, which include a synthetic dataset and three real-world recommendation datasets. On all datasets, we compared against the following bandit baselines: LinUCB [20], a reference stationary linear contextual bandit algorithm; dLinUCB [35], AdTS [14], WMDUCB1 [38], and Meta-Bandit [15], which are state-of-the-art bandit algorithms for piece-wise stationary environment. For DenBand, we tested two variants of it: DenBand-lcb, which uses lower confidence bound of badness to select a bandit expert, and DenBand-avg, which takes the average reward UCBs from all admissible bandit experts.

4.1 Experiments on Synthetic Dataset

• **Simulation Settings.** In simulation, we generate a size-K (K = 1000) arm pool $\mathcal{A}$, in which each arm a is associated with a d-dimensional (d = 10) feature vector $\mathbf{x}_a \in \mathbb{R}^d$ with $\|\mathbf{x}_a\|_2 \le 1$. Similarly, we create a set of ground-truth bandit parameters $\boldsymbol{\theta}^* \in \mathbb{R}^d$ with $\|\boldsymbol{\theta}^*\|_2 \le 1$, which are not disclosed to the learners. The standard deviation of Gaussian noise σ on the reward is set to 0.05 by default. To simulate an abruptly changing environment, after every S rounds, we randomize $\boldsymbol{\theta}^*$ with respect to the constraint that at least ρ portion of arms in $\mathcal{A}$ satisfy $|\mathbf{x}_a^\top\boldsymbol{\theta}^*_{t_{c_j}} - \mathbf{x}_a^\top\boldsymbol{\theta}^*_{t_{c_{j+1}}}| > \Delta_H$. And by default, we set S to 200, Δ_H to 0.7, and ρ to 0.2.

• **Empirical Regret Comparisons.** Under this simulation setting, we execute all algorithms 1000 iterations and report their accumulated regret in Figure 3 (a). Because LinUCB imposes a stationary assumption about the environment, it suffers almost linearly increasing regret after the first change point. Both AdTS and dLinUCB can react to the environment changes, but they are slow in doing so and thus accumulate increasing regret. Both of our proposed algorithms, DenBand-lcb and DenBand-avg, can quickly identify the changes and create corresponding bandit experts to capture the new reward distributions. The solid and dashed vertical lines in Figure 3 (a) show the actual and detected change points by DenBand-lcb respectively. We can clearly notice that DenBand can almost immediately respond to the changes in the environment. To improve visibility of Figure 3 (a), WMDUCB and Meta-Bandit are excluded.



(a) Accumulated regret over time

(b) Parameter estimation quality of bandit experts

Figure 3: Performance comparison on a synthetic dataset.

Table 1: Accumulated regret with different settings of the non-stationary environment.

$(\sigma, \Delta_H, S, \rho)$	(0.05, 0.7, 200, 0.8)	(0.1, 0.7, 200, 0.8)	(0.05, 0.7, 400, 0.8)	(0.05, 0.5, 200, 0.8)	(0.05, 0.5, 200, 0.5)	(0.05, 0.5, 200, 0.2)
DenBand-lcb	**22.48 ± 3.12**	36.68 ± 6.64	16.55 ± 1.48	**23.24 ± 3.01**	**18.96 ± 1.96**	22.45 ± 2.32
DenBand-avg	25.54 ± 3.88	**33.87 ± 4.13**	**13.08 ± 2.12**	23.51 ± 2.55	20.40 ± 2.06	**19.87 ± 1.85**
dLinUCB	35.12 ± 2.20	54.58± 3.56	22.33 ± 2.83	35.72 ± 2.05	34.45 ± 2.83	36.65 ± 2.10
AdTS	65.16± 20.30	67.43 ± 21.49	38.61 ± 5.08	70.74 ± 26.60	58.59 ± 14.13	52.45 ± 14.21
LinUCB	253.22 ± 7.12	263.14 ± 11.76	257.50 ± 8.07	216.60 ± 11.32	206.29 ± 15.71	164.68 ± 8.98
WMDUCB1	519.73 ± 21.53	528.48 ± 23.63	473.13 ± 43.29	484.23 ± 17.73	404.17 ± 10.95	372.30 ± 9.68
Meta-Bandit	585.70 ± 3.72	585.97 ± 4.62	499.83 ± 5.14	454.56 ± 4.29	412.30 ± 2.68	362.96 ± 2.17

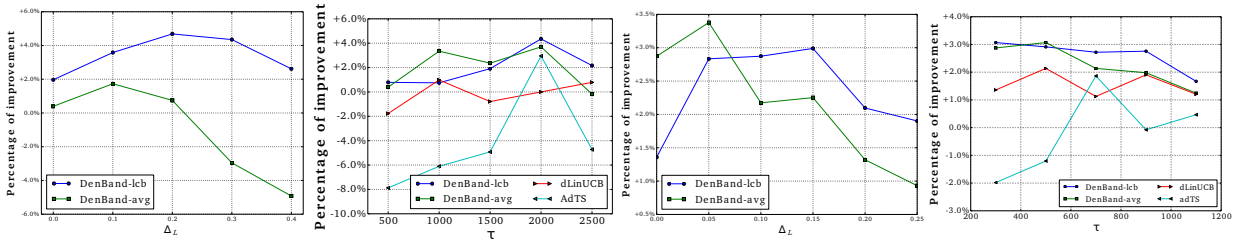
Table 2: Accumulated regret with different hyperparameter configurations.

(τ, Δ_L)	(30, 0.0)	(50, 0.0)	(80, 0.0)	(100, 0.0)	(100, 0.025)	(100, 0.05)	(100, 0.1)	(100, 0.15)
DenBand-lcb	24.36 ± 4.23	22.48 ± 3.12	19.33 ± 2.65	18.36 ± 3.27	19.27 ± 1.74	19.58 ± 1.38	21.08 ± 3.17	23.75 ± 5.96
DenBand-avg	25.73 ± 1.53	21.19 ± 3.29	18.87± 2.59	19.04 ± 1.65	19.52 ± 2.10	22.78 ± 2.19	26.84 ± 4.03	31.81 ± 6.08
dLinUCB	35.08 ± 2.59	35.12 ± 2.20	40.28 ± 2.52	42.86 ± 2.23	-	-	-	-
AdTS	94.90 ± 13.08	65.16 ± 20.30	123.05 ± 14.94	91.31 ± 20.08	-	-	-	-

In addition, we also include two oracle algorithms in Figure 3 (a). One is to start a new bandit expert at each change point and only uses it in this period, named as *OracleRestart*. The other, named as *OracleReuse*, maintains one bandit expert for each stationary period, and uses all ground-truth reusable experts for reward estimation in change-invariant arms. DenBand performs very closely to such optimal algorithms, although it does not get access to the ground-truth environment changes. Figure 3 (b) shows the parameter estimation quality (i.e., the L2 difference between the estimated parameter and the ground truth parameter) for four dynamically created bandit experts in DenBand-lcb. For comparison, we also include the estimation quality of experts from the OracleReuse algorithm. The good estimation quality of each bandit expert in DenBand further verifies our conclusion in Theorem 1 about the confidence bound of admissible bandit experts' estimation quality.

• **Sensitivity to Environment Settings.** According to our regret analysis, the performance of DenBand depends on the environment settings: including standard deviation σ in the Gaussian noise, length S of stationary period, proportion ρ of change-sensitive arms, and magnitude Δ_H of reward change. We varied them in simulation to investigate their influence on the algorithms. We ran all algorithms for 10 times and reported the mean and standard derivation of obtained accumulated regret in Table 1. DenBand consistently achieved the best performance against all baselines in all environment settings. As expected, a larger noise level σ leads to worse regret in almost all algorithms. Since T is fixed in our simulation, a smaller S leads to a larger Γ_T, which linearly scales DenBand's regret. When Δ_H becomes smaller, the regret of the two variants of DenBand and AdTS are further reduced. This is because once Δ_H satisfies the requirement in Theorem 1, the change can be confidently detected; and in the meanwhile, because of a smaller Δ_H, the added regret from using a false-positive bandit expert becomes smaller. Lastly, ρ does not seriously affect the performance of DenBand, which indicates that the algorithm is robust to ρ as long as Assumption 1 is satisfied.

• **Sensitivity to Hyper-Parameters.** To verify the robustness of our proposed algorithm, we studied the effect of two important hyper-parameters in Table 2: τ, whic determines the number of most recent observations used in bandit auditors, and Δ_L, which introduces flexibility and accounts for the existence of change-invariant arms with infinitesimal reward shift. As a comparison, we also studied the effect of τ on dLinUCB and AdTS, which also use a sliding window for change detection. From Table 2 we can find that both variants of DenBand are robust to its parameter τ as long as it falls into the required range. This result verified our theoretical analysis about the role of τ in Theorem 1, which defines an upper and lower bound of τ for the proved confidence bound. On the contrary, the baselines, especially AdTS, are very sensitive to the setting of τ. For the parameter Δ_L, DenBand-lcb is very robust to it, but DenBand-avg is not. This can be explained from two perspectives: 1). As Δ_L accounts for the existence of change-invariant arms with infinitesimal reward shift, it should be upper bounded as required in Theorem 1. 2). When Δ_L is set too large, inadmissible bandit experts may be mistakenly selected. In this



(a) Effect of Δ_L on Yahoo dataset

(b) Effect of τ on Yahoo dataset

(c) Effect of Δ_L on Snapchat dataset

(d) Effect of τ on Snapchat dataset

Figure 4: Effect of hyper-parameters on DenBand on two real-world datasets

case, DenBand-avg suffers from a contaminated average ensemble of bandit experts in reward prediction. DenBand-lcb reduces the risk by selecting a bandit expert by its lower confidence bound of estimated badness.

4.2 Experiments on Real-World Datasets

• **Datasets.** We tested all algorithms on the following three real-world recommendation datasets.

The first dataset is a large collection of Yahoo frontpage recommendation log, made available by the Yahoo Webscope program [20]. It contains 45,811,883 user visits to Yahoo Today Module in a ten-day period in May 2009. Unbiased offline evaluation is performed on this dataset following the offline evaluation protocol used in [20, 21].

The second dataset is a user click log collected from the Snapchat lens recommendation platform over a one-week period. This dataset contains hundred millions of anonymous observations related to 495 unique lenses and more than five hundred thousand randomly selected users. Each observation contains an anonymous user id, a lens id which is recommended by the logging policy, and corresponding user actions on this lens. Each lens is associated with an 8-dimensional feature vector constructed by historical click statistics. The sharing-related actions on items are considered as positive user click feedback. Due to the sparsity of observations, users are grouped into 22 groups according to their demographic information, and estimate one bandit model in each user group. The same offline evaluation protocol is used here as in the Yahoo news recommendation dataset.

The third data is extracted from the music streaming service Last.fm, which is made available on the HetRec 2011 workshop [2]. This dataset contains 1892 users and 17632 items (artists). We treat the 'listened artists' in each user as positive feedback. Following the settings in [8, 32, 36], we pre-processed the dataset in order to fit them into a contextual bandit setting. And we followed [15, 35] to simulate a non-stationary environment: we ordered observations chronologically inside each user, and built a single hybrid user by merging different users. Hence, the boundary between two consecutive batches of observations from two original users is treated as the preference change of the hybrid user.

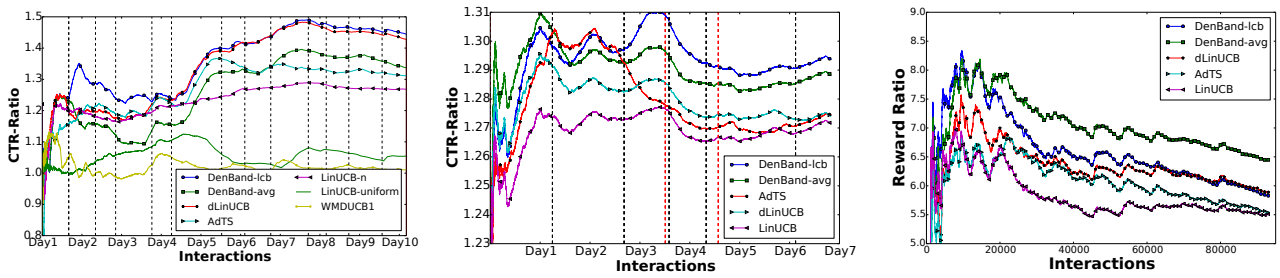
• **Sensitivity of Hyper-Parameters.** To test the proposed algorithms' sensitivity to hyper-parameters and identify the best hyper-parameter setting, we perform parameter tuning using one day's data from yahoo dataset and two days' data from Snapchat dataset. We vary Δ_L and τ, and report the relative performance improvement compared with LinUCB in Figure 4. From Figure 4 (a) and (c), we notice that DenBand-lcb is very robust to the setting of Δ_L, while the performance of DenBand-avg can be bad when Δ_L is too large. As discussed in our study of hyper-parameters in Section 4.1, the reason is that for a larger Δ_L, DenBand is more likely to include false positive models as admissible bandit experts. In this case, the reward estimation quality may become inaccurate in DenBand-avg, which evenly trust all admissible experts' output. As for τ, in Figure 4 (b) and (d) we observed that AdTS is very sensitive to the choice of τ, while both variants of DenBand are quite robust to it as long as it is within a reasonable range.

• **Recommendation Quality** In Figure 5 (a), we report normalized Click-Through Rate (CTR) from different algorithms based on the corresponding logged random strategy's CTR on the Yahoo dataset. We set τ to 2,000 and Δ_L to 0.3 based on the hyper-parameter tuning results. From Figure 5 (a), we can find that DenBand-lcb achieves significant improvement compared with all baselines, especially at the beginning of the testing period. WMDUCB1 performs the worst as it cannot utilize any available context information. We also looked into the detected change points by DenBand-lcb, and found that it detected 25 changes in total, and we plotted 12 of them in Figure 5 (a) using vertical lines (to increase visibility of the figure). We can find close correspondence between its detected change points and its performance improvement.

Using the same evaluation protocol, we report the CTR ratio between different algorithms and the logging policy on Snap dataset in Figure 5 (b). We set τ to 500 and Δ_L to 0.1. On this dataset, the context-free algorithms perform significantly worse than the contextual ones, so that we excluded them from the comparisons. The results show that our algorithms achieve a 29% - 31% improvement comparing to the logging policy, and a 3.1% improvement against LinUCB in a per-user basis. Comparing to AdTS, although our proposed algorithm performs similarly at the beginning, it caught up very quickly later. We plot the detected change points for two largest groups of users (users in the same group share the same set of bandit experts) in Figure 5 (b) using vertical lines of different colors, which well correlate with performance improvement during the adaptive recommendation process.

The ratio between reward from the bandit algorithms and that from a random selection policy on LastFM dataset is reported in Figure 5 (c), where Δ_L is set to 0.1 and τ is set to 300. From this result, we can see that DenBand, especially DenBand-avg outperform all

[2] http://grouplens.org/datasets/hetrec-2011



(a)Yahoo dataset (b) Snapchat dataset (c) LastFM dataset

Figure 5: Normalized CTR comparison on three real-world datasets.

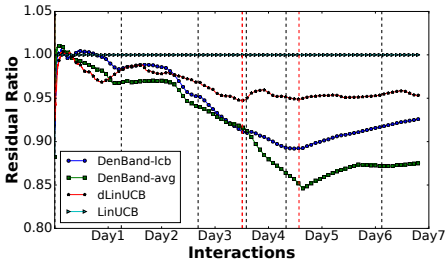


Figure 6: Residual ratio on Snapchat dataset

the baselines. LinUCB performs the worst as it failed to capture the non-stationarity of the environment. Since the distribution of items is highly skewed [8], the context-free bandits perform very poorly on this dataset. We therefore decide to exclude them from comparison in the figure.

• **Reward estimation quality and qualitative study.** In order to investigate the effectiveness of DenBand in reward estimation and illustrate the source of its performance improvement, we looked into its reward prediction error on the selected items on Snapchat dataset. In Figure 6, we normalize the accumulated residuals (i.e., reward prediction error) from different algorithms by that from LinUCB. It shows that both variants of DenBand have smaller reward estimation error on the selected items. In addition, we illustrate the detected change points for the two largest user groups using vertical lines. The detected changes correlate well with the turning points of residual ratio in DenBand. For example, at the first and last detected change points, the residual ratio is increasing, which indicates that the old bandit model is becoming bad; and thus new bandit experts are created to account for the new environment.

To reveal how DenBand recognizes the changes of users' interests in a context-dependent manner, we looked into the high reward items, change-invariant items, and change-sensitive items according to our sequentially created bandit experts and their corresponding auditors on the LastFM dataset. For each bandit expert and auditor pair, we used the learnt models in the bandit expert to get the top 500 high reward items, and used its auditor to get the top 500 change-invariant items and top 500 change-sensitive items separately. As each item is associated with some short text descriptions provided by the users, we then generated word clouds for each group of those selected items to summarize the learnt bandit models in Figure 7. It is interesting to find that the change-invariant items tend to be related with geographical regions. For example, the change-invariant items in group 1 are mostly about Brazilian

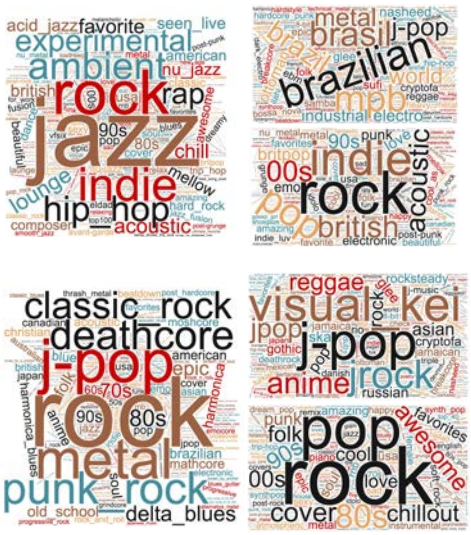


Figure 7: Word cloud visualization of two bandit experts and their auditors on LastFM. In each of the word cloud groups, the left one shows the tags from high reward items selected by the bandit expert; the upper right one shows tags from change-invariant items, and the lower right one shows tags collected from change-sensitive items according to the bandit auditor, respectively.

music, and those in group 2 are related to Japanese music. The change-sensitive items are mostly related to those more common genres of music. And the high reward items are a mix of these two types. Hence recognizing the changing and stable users' interest is essential in making satisfactory recommendations.

5 CONCLUSIONS & FUTURE WORK

We studied contextual bandits for adaptive recommendation in a piece-wise stationary environment. Capitalizing on the context-dependent property of environment changes, bandit models are dynamically ensembled and reused to conquer the non-stationary environment. Rigorous regret analysis validates the convergence of the proposed solution, and extensive empirical evaluations on simulation and three large real-world datasets verified the effectiveness and reliability of the proposed algorithm.

In this work, we treat bandit learners as independent from each other. As existing works have shed light on collaborative bandit

learning [13], it is meaningful to study non-stationary bandits in a collaborative manner, e.g., monitoring the change and reusing models across users. It is also important to study a continuously changing environment, such as Brownian motion or periodical changes, where the context-dependent changes help us better maintain and create bandit models.

6 ACKNOWLEDGMENTS

This work was supported in part by National Science Foundation Grant IIS-1553568 and IIS-1618948, and Bloomberg Data Science PhD Fellowship.

7 APPENDIX

7.1 Proof of Theorem 1

Proof of Eq (3)**:** For the bandit expert m created at time t_m, we split its training instances in $\mathcal{I}_t^\theta(m)$ up to time t into two sets $\mathcal{H}_t(m)$ and $\tilde{\mathcal{H}}_t(m)$. The instances in $\mathcal{H}_t(m)$ are all from the reward distribution governed by $\boldsymbol{\theta}^*_{t_m}$, while the instances in $\tilde{\mathcal{H}}_t(m)$ are not. According to the first order optimum condition in ridge regression and the definition of $\mathbf{A}_t(m)$, the following equation can be obtained,

$$\hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}^*_{t_m} = \mathbf{A}_t^{-1}\Big(\sum_{i\in\mathcal{I}_t^\theta(m)} \mathbf{x}_i\eta_i - \lambda\boldsymbol{\theta}^*_{t_m} - \sum_{i\in\tilde{\mathcal{H}}_t(m)} \mathbf{x}_i\mathbf{x}_i^\intercal(\boldsymbol{\theta}^*_{t_m} - \boldsymbol{\theta}^*_i)\Big)$$

Based on the self-normalized martingale bound in Theorem 1 of [1], for any $\delta_1 \in (0,1)$, with a probability at least $1-\delta_1$, we have

$$\|\hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}^*_{t_m}\|_{\mathbf{A}_t} \le \sigma^2\sqrt{d\ln\frac{\lambda+|\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}} + \sqrt{\lambda} + \|\sum_{i\in\tilde{\mathcal{H}}_t(m)} \mathbf{x}_i\mathbf{x}_i^\intercal(\boldsymbol{\theta}^*_t - \boldsymbol{\theta}^*_i)\|_{\mathbf{A}_t^{-1}}$$

To simplify notations, we define $F_t(m) = \sum_{i\in\tilde{\mathcal{H}}_t(m)} \mathbf{x}_i\mathbf{x}_i^\intercal(\boldsymbol{\theta}^*_t - \boldsymbol{\theta}^*_i)$. Then we have,

$$\begin{aligned}|\mathbf{x}_a^\intercal\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^\intercal\boldsymbol{\theta}^*_{t_m}| &\le \|\hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}^*_{t_m}\|_{\mathbf{A}_t}\|\mathbf{x}_{a_t}\|_{\mathbf{A}_t^{-1}} \qquad (6)\\ &\le \Big(\sigma^2\sqrt{d\ln\frac{\lambda+|\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}} + \sqrt{\lambda} + \|F_t(m)\|_{\mathbf{A}_t^{-1}}\Big)\|\mathbf{x}_{a_t}\|_{\mathbf{A}_t^{-1}}\end{aligned}$$

According to the definition of $F_t(m)$ and $\tilde{H}_t(m)$, $F_t(m)$ can be understood as a form of contamination in bandit expert m's estimation of the ground-truth bandit parameter $\boldsymbol{\theta}^*_{t_m}$. Denote $t_c(m)$ as the time index of the first change point after the bandit expert m is created. When $t > t_c(m)$, essentially, the contamination in $F_t(m)$ comes from two sources: First, erroneous updates from change-sensitive arms, which is referred as false positive selection of bandit experts. In Lemma 2, we proved that with a high probability there will not be any false positive selection up to time t when the auditor's estimation is not contaminated (best case scenario). But we may have additional erroneous updates when the auditor's observations contain change-sensitive arms collected after the change points. Therefore, the possible erroneous updates are at most $2\frac{(t-t_m)}{S_{\min}}\tau$. This also explains why the sliding observation window τ of the bandit auditors cannot be arbitrarily large when we explained in the requirement of this Theorem. The second source of error is the small contamination from change-invariant arms, which can be bounded by $\big(t - t_c(m)\big)\Delta_L$, due to the fact that for change-invariant arms $|\mathbf{x}_a^\intercal(\boldsymbol{\theta}^*_{t_m} - \boldsymbol{\theta}^*_i)| \le \Delta_L$. As a result, we have $F_t(m) \le 2\frac{t-t_m}{S_{\min}}\tau\rho + \big(t - t_c(m)\big)\Delta_L$. Thus when $\Delta_L \le \frac{\sigma^2\sqrt{d\lambda\ln\frac{\lambda+|\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}}}{t-t_c(m)}$ and $\tau \le S_{\min}\frac{\sigma^2\sqrt{d\lambda\ln\frac{\lambda+|\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}}}{2\rho(t-t_m)}$, we have $\|F_t(m)\|_{\mathbf{A}_t^{-1}} \le 2\sigma^2\sqrt{d\ln\frac{\lambda+|\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}}$. Substituting this inequaity into Eq (6) concludes the proof.

Proof of Eq (4)**:** The training instances for $\hat{\boldsymbol{\beta}}_t(m)$ are determined by the bandit expert's estimation about the ground-truth reward distribution in the environment. Every time when $\hat{\boldsymbol{\theta}}_t(m)$ gets updated, we will revise the historical training instances in $\hat{\boldsymbol{\beta}}_t(m)$, i.e., compute $e_{a,t}(m)$ by the updated reward estimation. In addition, the bandit auditors only accumulate badness observations in the most recent τ interactions. When there is no environment change in this time window, Eq (4) can be easily derived based on [1]. For the case in which there is environment change among the τ observations, wrong selection and update of bandit experts may happen but the number is relatively small, since the number of mistakes is at most τ in each stationary period. The effect of these mistakes will be taken into account in both the bandit expert's reward estimation and the final regret in Theorem 2.

Proof of Eq (5)**:** When $t \le t_c(m)$, Eq (5) is equivalent to Eq (3). When $t > t_c(m)$, we have $|\mathbf{x}_a^\intercal\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^\intercal\boldsymbol{\theta}^*_t| \le |\mathbf{x}_a^\intercal\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^\intercal\boldsymbol{\theta}^*_{t_m}| + |\mathbf{x}_a^\intercal\boldsymbol{\theta}^*_{t_m} - \mathbf{x}_a^\intercal\boldsymbol{\theta}^*_t|$, in which the first term can be bounded by $B^\theta_{a,t}(m)$ according to Eq (3). With Lemma 2, when the bandit expert m is selected for arm a, with a high probability the arm is change-invariant, which means that $|\mathbf{x}_a^\intercal\boldsymbol{\theta}^*_{t_m} - \mathbf{x}_a^\intercal\boldsymbol{\theta}^*_t| \le \Delta_L$. Putting them together concludes the proof.

7.2 Proof of Theorem 2

Proof of Theorem 2. According to the reward confidence bound proved in Eq (5) of Theorem 1, and the UCB arm selection strategy in Algorithm 1, the regret at time t can be upper bounded by,

$$\begin{aligned}\mathbb{E}[r_{a^*_t,t}] - \mathbb{E}[r_{a_t,t}] &\le \mathbf{x}_{a_t}\hat{\boldsymbol{\theta}}_t(m_{a_t,t}) + B^\theta_{t,a_t}(m_{a_t,t}) + \Delta_L - \mathbf{x}_{a_t}\boldsymbol{\theta}^*_t\\ &\le 2\Delta_L + 2B^\theta_{t,a_t}(m_{a_t,t}) \qquad (7)\end{aligned}$$

Combining with the additional regret caused by the events when a bandit auditor's most recent τ observations contain environment changes, the cumulative regret can be upper bounded by $R(T) \le 2\Gamma_T\tau + 2T\Delta_L + \sum_{m\in\mathcal{M}_T}\sum_{i\in\Omega_m} 2B^\theta_{i,a_i}(m)$, in which Ω_m is the set of time indices when the bandit expert m is used for arm selection up to time T. According to the proof of Theorem 3 in [1], we have,

$$\sum_{i\in\Omega_m} B^\theta_{i,a_i}(m) \le \Big(3\sigma^2\sqrt{d\ln\frac{\lambda+|\mathcal{I}_T^\theta(m)|}{\lambda\delta_1}} + \sqrt{\lambda}\Big)\sqrt{|\Omega_m|d\ln(\lambda + \frac{|\Omega_m|}{d})}$$

Since $\sum_{i\in\Omega_m} 2B^\theta_{a_i,i}(m)$ is a concave function with respect to $|\Omega_m|$, according to the Jensen's inequality $\sum_{m\in\mathcal{M}_T}\sum_{i\in\Omega_m} 2B^\theta_{i,a_i}(m) \le |\mathcal{M}_T|\Big(3\sigma^2\sqrt{d\ln\frac{\lambda+U_T}{\lambda\delta_1}} + \sqrt{\lambda}\Big)\sqrt{S_{\max}d\ln(\lambda + \frac{S_{\max}}{d})}$, in which $U_T = \max\{|\mathcal{I}_T^\theta(m)|\}_{m\in\mathcal{M}_T}$.

Regarding to the number of bandit experts in $\mathcal{M}_T$: denote the possible number of false negative detection as k_{FN}, which follows a binomial distribution $B(T, P_{FN})$. According to Lemma 1 and the tail bound of Binomial distribution, it can be proved that with a high probability, $|\mathcal{M}_T| \le \Gamma_T + k_{FN} \le \Gamma_T + 1$. Combining all the

conclusions above we have,

$$R(T) \leq 2\Gamma_T \tau + 2T\Delta_L + \sum_{m \in \mathcal{M}_T} \sum_{i \in \mathcal{S}_m} 2B^{\theta}_{i,a_i}(m) \leq (\frac{1}{\rho} + 2)\sigma^2 \sqrt{d\lambda \ln \frac{\lambda + T}{\lambda \delta_1}}$$

$$+ (\Gamma_T + 1)\Big(3\sigma^2 \sqrt{d \ln \frac{\lambda + U_T}{\lambda \delta_1}} + \sqrt{\lambda}\Big) \sqrt{S_T d \ln(\lambda + \frac{S_T}{d})}$$

which finishes the proof. □

Proofs of Lemma 1 and Lemma 2 are omitted here due to space limit. They will be provided in a longer version of this paper.

REFERENCES

[1] Yasin Abbasi-yadkori, Dávid Pál, and Csaba Szepesvári. 2011. Improved Algorithms for Linear Stochastic Bandits. In *NIPS*. 2312–2320.

[2] Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. 2011. Analyzing temporal dynamics in twitter profiles for personalized recommendations in the social web. In *Proceedings of the 3rd International Web Science Conference*. ACM, 2.

[3] Peter Auer. 2002. Using Confidence Bounds for Exploitation-Exploration Tradeoffs. *Journal of Machine Learning Research* 3 (2002), 397–422.

[4] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. *Mach. Learn.* 47, 2-3 (May 2002), 235–256. https://doi.org/10.1023/A:1013689704352

[5] Dimitris Bertsimas and José Niño Mora. 2000. Restless Bandits, Linear Programming Relaxations, and a Primal-Dual Index Heuristic. *Oper. Res.* 48, 1 (Jan. 2000), 80–90. https://doi.org/10.1287/opre.48.1.80.12444

[6] John S. Breese, David Heckerman, and Carl Kadie. 1998. *Empirical Analysis of Predictive Algorithms for Collaborative Filtering*. Technical Report MSR-TR-98-12. Microsoft Research. 18 pages.

[7] Yang Cao, Zheng Wen, Branislav Kveton, and Yao Xie. 2018. Nearly Optimal Adaptive Procedure for Piecewise-Stationary Bandit: a Change-Point Detection Approach. *https://arxiv.org/abs/1802.03692* (2018).

[8] Nicolò Cesa-Bianchi, Claudio Gentile, and Giovanni Zappella. 2013. A Gang of Bandits. *In Pro. NIPS* (2013).

[9] Olivier Chapelle and Lihong Li. 2011. An Empirical Evaluation of Thompson Sampling. In *Advances in Neural Information Processing Systems*. 2249–2257.

[10] Joohyun Lee Fang Liu and Ness Shroff. 2018. A Change-Detection based Framework for Piecewise-stationary Multi-Armed Bandit Problem *(AAAI'18)*.

[11] Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. 2010. Parametric bandits: The generalized linear case. In *NIPS*. 586–594.

[12] Aurélien Garivier and Eric Moulines. 2008. On upper-confidence bound policies for non-stationary bandit problems. *arXiv preprint arXiv:0805.3415* (2008).

[13] Claudio Gentile, Shuai Li, Purushottam Kar, Alexandros Karatzoglou, Giovanni Zappella, and Evans Etrue. 2017. On Context-Dependent Clustering of Bandits. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Doina Precup and Yee Whye Teh (Eds.), Vol. 70. PMLR, International Convention Centre, Sydney, Australia, 1253–1262.

[14] Negar Hariri, Bamshad Mobasher, and Robin Burke. 2015. Adapting to User Preference Changes in Interactive Recommendation. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*. AAAI Press, 4268–4274.

[15] Cédric Hartland, Sylvain Gelly, Nicolas Baskiotis, Olivier Teytaud, and Michèle Sebag. 2006. Multi-armed Bandit, Dynamic Environments and Meta-Bandits. (Nov. 2006). https://hal.archives-ouvertes.fr/hal-00113668 working paper or preprint.

[16] Jaya Kawale, Hung H Bui, Branislav Kveton, Long Tran-Thanh, and Sanjay Chawla. 2015. Efficient Thompson Sampling for Online Matrix-Factorization Recommendation. In *Advances in Neural Information Processing Systems*. 1297–1305.

[17] Michal Kompan and Mária Bieliková. 2010. Content-Based News Recommendation. In *E-Commerce and Web Technologies*, Francesco Buccafurri and Giovanni Semeraro (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 61–72.

[18] Georgia Koutrika. 2018. Recent Advances in Recommender Systems: Matrices, Bandits, and Blenders. In *EDBT*.

[19] John Langford and Tong Zhang. 2008. The Epoch-Greedy Algorithm for Multi-armed Bandits with Side Information. In *NIPS*. 817–824.

[20] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of 19th WWW*. ACM, 661–670.

[21] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of 4th WSDM*. ACM, 297–306.

[22] Lei Li, Dingding Wang, Tao Li, Daniel Knox, and Balaji Padmanabhan. 2011. SCENE: A Scalable Two-stage Personalized News Recommendation System. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '11)*. ACM, New York, NY, USA, 125–134. https://doi.org/10.1145/2009916.2009937

[23] Wei Li, Xuerui Wang, Ruofei Zhang, Ying Cui, Jianchang Mao, and Rong Jin. 2010. Exploitation and exploration in a performance based contextual advertising system. In *Proceedings of 16th SIGKDD*. ACM, 27–36.

[24] Jiahui Liu, Peter Dolan, and Elin Rønby Pedersen. 2010. Personalized news recommendation based on click behavior. In *Proceedings of the 15th international conference on Intelligent user interfaces*. ACM, 31–40.

[25] Christos H. Papadimitriou and John N. Tsitsiklis. 1999. The Complexity of Optimal Queuing Network Control. *Math. Oper. Res.* 24, 2 (May 1999), 293–305. https://doi.org/10.1287/moor.24.2.293

[26] Owen Phelan, Kevin McCarthy, Mike Bennett, and Barry Smyth. 2011. Terms of a Feather: Content-based News Recommendation and Discovery Using Twitter. In *Proceedings of the 33rd European Conference on Advances in Information Retrieval (ECIR'11)*. Springer-Verlag, Berlin, Heidelberg, 448–459. http://dl.acm.org/citation.cfm?id=1996889.1996947

[27] Yi Qi, Qingyun Wu, Hongning Wang, Jie Tang, and Maosong Sun. 2018. Bandit Learning with Implicit Feedback. In *Advances in Neural Information Processing Systems*. 7287–7297.

[28] Vishnu Raj and Sheetal Kalyani. 2017. Taming non-stationary bandits: A Bayesian approach. *arXiv preprint arXiv:1707.09727* (2017).

[29] Herbert Robbins. 1952. Some aspects of the sequential design of experiments. *Bull. Amer. Math. Soc.* 58, 5 (09 1952), 527–535. http://projecteuclid.org/euclid.bams/1183517370

[30] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of 10th WWW*. ACM, 285–295.

[31] Alex Slivkins and Eli Upfal. 2008. Adapting to a Changing Environment: the Brownian Restless Bandits, In 21st Conference on Learning Theory (COLT). 343–354.

[32] Huazheng Wang, Qingyun Wu, and Hongning Wang. 2016. Learning hidden features for contextual bandits. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1633–1642.

[33] Huazheng Wang, Qingyun Wu, and Hongning Wang. 2017. Factorization bandits for interactive recommendation. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[34] P. Whittle. 1988. Restless bandits: activity allocation in a changing world. *Journal of Applied Probability* 25, A (1988), 287–298. https://doi.org/10.1017/S0021900200040420

[35] Qingyun Wu, Naveen Iyer, and Hongning Wang. 2018. Learning Contextual Bandits in a Non-stationary Environment. In *The 41st International ACM SIGIR Conference on Research Development in Information Retrieval (SIGIR '18)*. ACM, 495–504.

[36] Qingyun Wu, Huazheng Wang, Quanquan Gu, and Hongning Wang. 2016. Contextual Bandits in a Collaborative Environment. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 529–538.

[37] Qingyun Wu, Hongning Wang, Liangjie Hong, and Yue Shi. 2017. Returning is believing: Optimizing long-term user engagement in recommender systems. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1927–1936.

[38] Jia Yuan Yu and Shie Mannor. 2009. Piecewise-stationary Bandit Problems with Side Observations. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09)*. ACM, New York, NY, USA, 1177–1184. https://doi.org/10.1145/1553374.1553524

Dual Graph Attention Networks for Deep Latent Representation of Multifaceted Social Effects in Recommender Systems

Qitian Wu
Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University
echo740@sjtu.edu.cn

Hengrui Zhang
Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University
sqstardust@sjtu.edu.cn

Xiaofeng Gao
Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University
gao-xf@cs.sjtu.edu.cn

Peng He
WeChat, Tencent Inc.
paulhe@tencent.com

Paul Weng
UM-SJTU Joint Institute, Shanghai Jiao Tong University
paul.weng@sjtu.edu.cn

Han Gao
WeChat, Tencent Inc
alangao@tencent.com

Guihai Chen
Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University
gchen@cs.sjtu.edu.cn

ABSTRACT

Social recommendation leverages social information to solve data sparsity and cold-start problems in traditional collaborative filtering methods. However, most existing models assume that social effects from friend users are static and under the forms of constant weights or fixed constraints. To relax this strong assumption, in this paper, we propose *dual graph attention networks* to collaboratively learn representations for two-fold social effects, where one is modeled by a user-specific attention weight and the other is modeled by a dynamic and context-aware attention weight. We also extend the social effects in user domain to item domain, so that information from related items can be leveraged to further alleviate the data sparsity problem. Furthermore, considering that different social effects in two domains could interact with each other and jointly influence users' preferences for items, we propose a new policy-based fusion strategy based on *contextual multi-armed bandit* to weigh interactions of various social effects. Experiments on one benchmark dataset and a commercial dataset verify the efficacy of the key components in our model. The results show that our model achieves great improvement for recommendation accuracy compared with other state-of-the-art social recommendation methods.

KEYWORDS

Recommender Systems, Graph Neural Network, Contextual Multi-Armed Bandit, Representation Learning, Social Influence Analysis

ACM Reference Format:
Qitian Wu, Hengrui Zhang, Xiaofeng Gao, Peng He, Paul Weng, Han Gao, and Guihai Chen. 2019. Dual Graph Attention Networks for Deep Latent Representation of Multifaceted Social Effects in Recommender Systems. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313442

1 INTRODUCTION

Recommender systems, which aim at filtering and suggesting items of potential interests to targeted users, have extensive applications and can bring up a huge amount of commercial benefits, especially when the information overload problem has been pervasive with the rapid development of the World Wide Web. Conventional collaborative filtering (CF) methods for recommendation suffer from data sparsity and cold start problems. To this end, social recommendation, which attempts to leverage the social networks of users to incorporate more useful information, has been proposed to alleviate these two problems [5, 12, 19, 20].

Prior Works and Limitations. Nowadays, there are many composite platforms which integrate item recommendation and social network services (SNS). Users on these platforms can click on or

*Xiaofeng Gao is the corresponding author. This work was supported by the National Key R&D Program of China [2018YFB1004703]; the National Natural Science Foundation of China [61872238, 61672353]; the Shanghai Science and Technology Fund [17510740200]; the Huawei Innovation Research Program [HO2018085286]; and the State Key Laboratory of Air Traffic Management System and Technology [SKLATM20180X]; the Tencent Social Ads Rhino-Bird Focused Research Program.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313442

rate items as well as interact with their friends. A user's preference for one item could be impacted by her friends, which motivates us to probe into such social effect to improve recommendation quality. Previous studies for social recommendation attempt to model social effects in various ways, such as by trust propagation [5, 12, 22], regularization loss [20, 33, 40], matrix factorization [8, 18, 38], network embedding [17, 34], causal inference [16, 32], and graph neural network [36, 41].

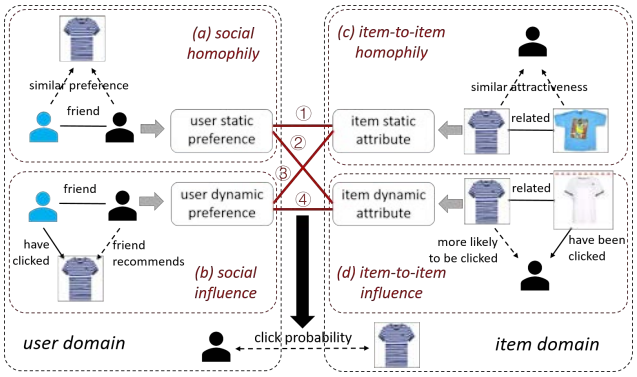


Figure 1: Illustration of the two-fold social effects, i.e., homophily effect and influence effect, in user social networks as well as among related items. The four social effects jointly affect a user's decision on one item. In this paper, we aim to build an unfied model to represent and weigh four effects given different user-item pairs.

However, these studies share several common limitations. **First**, most studies assume that linked users all share similar preferences. This assumption cannot suit well contemporary SNS since there could be various types of online friends, such as close friends, casual friends, and event friends [44]. Therefore, directly equating friendship on an SNS with preference similarity may not be sound. **Second**, most works model friends' influences statically under the forms of constant weights or fixed constraints. This assumption ignores the dynamic pattern of social effects. In fact, users could be influenced by a specific group of friends when it comes to specific items, which makes the social effect dynamic and dependent of specific contexts. **Third**, previous methods lack interpretability for social effects, i.e., they cannot explicitly indicate how a user's preference for one item is influenced by their friends.

Motivations and Rationales. In the real world, people's decisions could be influenced by various factors, and their behaviors are often the results of multifaceted causes. To this end, we probe into four different social effects in recommender systems, including two-fold effects in user domain and the other symmetric two-fold ones in item domain, as illustrated in Fig. 1. In user domain, friends could influence each other in two ways. For one thing, users tend to possess similar preferences to their friends, which is called as *social homophily* [24] (Fig. 1.(a)). Social homophily often leads to intrinsic effect for user preference which stays unchanged and independent of external contexts. For another, a user's friend who has purchased one item may recommend it to that user who then may be more likely to click such item, which is termed as *social influence* [21] (Fig. 1.(b)). Actually, social influence contributes to behavior-level effect for user preference which may change dynamically with specific contexts. *For example, a commercial analyst may be influenced by different communities of friends when it comes to an entertainment article or a commercial news.* Therefore, we can distinguish two components that may impact a user's preference: a static effect from social homophily and a dynamic effect from social influence[1], and we call corresponding user's preference as static and dynamic preference, respectively (left part of Fig. 1).

Moreover, in item domain, there exist analogous 'social effects'. For one thing, some related items tend to possess similar attractiveness when exposed to users. We term this as *item-to-item homophily*, and call the component of an item attribute affected by such effect as static attribute (Fig. 1.(c)). For another, if one item is popular among a certain community, then other related items would become more likely to be clicked by users in this social group. Such phenomenon can be termed as *item-to-item influence*, which depends on specific contexts (Fig. 1.(d)). *For example, a businessman and a researcher both read one article talking about the development of AI, but the businessman would prefer another article about financial information of AI companies, while the researcher would tend to click on another article about cutting-edge breakthroughs in AI.* We call the component of an item attribute affected by such effect as dynamic attribute. Finally, as shown by ①~④ in Fig. 1, the two-fold social effects in both user domain and item domain may jointly influence a user's decision on one item. Such phenomenon is pervasive on Epinion, Twitter, Facebook, and Netflix, etc.

Methodologies and Results. To these ends, in this paper, we propose *DANSER (Dual graph Attention Networks for modeling multifaceted Social Effects in Recommender systems)*, a novel architecture that extends graph attention networks (GAT) [31]. DANSER is composed of two dual GATs: one dual GAT for users—including a GAT to capture social homophily and a GAT to capture social influence—and the other dual GAT for items. Specifically, in user domain, one GAT to capture social homophily attentively aggregates adjacent users' embedding that reflects ones' preferences, and outputs representations for static user preference. Another GAT to capture social influence aims at convolution on adjacent users' context-aware preferences, which characterize relevance of users' rated items to candidate item, and outputs representations for dynamic user preference. In item domain, two analogous GATs as a dual architecture are designed to model items' static and dynamic attributes under effects of item-to-item homophily and influence, respectively, among related items. Such dual mechanism possesses two advantages: i) GATs to capture homophily and influence could collaboratively model two-fold social effects, taking advantage of both global and local views to investigate user-item interactions; ii) two GATs in item domain incorporate information from related items, which could further alleviate data sparsity problem.

Then, since user static (resp. dynamic) preference as well as item static (resp. dynamic) attribute could jointly affect a user's decision on one item (shown by ①~④ in Fig. 1) and the importance of these four interactions would vary for distinct user-item pairs, we propose a policy-based strategy to dynamically weigh the four interactions. Specifically, we model the problem as a contextual

[1]Previous studies tend to interchangeably use social homophily and social influence. In this paper, we discriminate these two notions to better investigate the dynamic social effect.

multi-armed bandit [1], and treat the weighing strategy as a policy conditional on the context (the targeted user-item pair). Then our goal is to optimize a reward (w.r.t predicted loss). For model training, we use stochastic policy gradient to update our neural network-based policy unit, and design a local-graph aware regularization technique to reduce computational cost for regularization. To verify our model, we conduct experiments on one benchmark datasets from *Epinions* and a commercial dataset from *WeChat Top Story*. The results show that DANSER outperforms state-of-the-art models under both explicit feedback settings (2.9% MAE improvement) and implicit feedback settings (4.5% AUC improvement).

Our contributions can be summarized as follows:

i) **General Aspects:** We distinguish the social homophily and social influence notions in view of static and dynamic effects. Also, we extend the two-fold social effects in user domain to item domain, and therefore investigate four social effects in recommender systems. These general aspects enable the model to capture more information and improve its capacity.

ii) **Novel Methodologies:** We propose DANSER with two dual GATs and a policy-based fusion unit. The dual GATs can collaboratively model four social effects in both user and item domains, while the policy unit, based on a contextual multi-armed bandit, dynamically weighs four interactions of social effects in two domains according to specific contexts.

iii) **Multifaceted Experiments:** We deploy DANSER on one benchmark dataset and a commercial dataset. The experiment results verify the superiority of DANSER over state-of-the-art techniques, the effectiveness of the proposed components, as well as its good interpretability for modeling social effects.

2 PRELIMINARY AND BACKGROUND

In this paper, we consider a user-item interaction matrix $\mathbf{R} = \{r_{ui}\}_{M\times N}$, where M and N are numbers of users and items respectively. For a recommender system with *implicit* feedback, $\mathbf{R}$ consists of 0 and 1, where $r_{ui} = 1$ if user u has clicked item i, and $r_{ui} = 0$ otherwise. For a recommender system with *explicit* feedback, r_{ui} represents the rating value given by user u on item i (In Epinions, rating values range from 1 to 5 and user could give a high rating if she likes the item), and $r_{ui} = 0$ if user u has not rated item i. We use $R_I(u)$ and $R_U(i)$ to respectively denote the set of items rated by user u and the set of users who have rated item i. Moreover, we assume the existence of a trust or friendship network for users, which is a graph $G_U = (V_U, E_U)$ where V_U is the set of users and E_U is the set of edges that connect two users. We use $F_U(u)$ to denote the set of nodes adjacent to u in G_U. Also, we consider interaction frequencies between user u and user v as edge features, denoted by $\mathbf{e}_{uv} \in \mathbb{R}^C$, where each element in $\mathbf{e}_{uv}$ denotes the frequency of one-type interaction between u and v, and C is the number of interaction types. The edge features reflect link strengths between two users.

Problem Formulation. The social recommendation problem can be defined as [28]: given observed interaction records in $\mathbf{R}$ and trust relationship G_U, we are to estimate unobserved interactions in $\mathbf{R}$, i.e., the probability of a targeted user u^+ clicking an unobserved candidate item i^+ (for implicit feedback) or the rating value user u^+ will score item i^+ (for explicit feedback).

3 METHODOLOGIES

We first introduce the model framework. For each part, we start from the motivations and intuitions, and then introduce the technical details. Secondly, we present some training techniques to improve the model performance. Finally, we discuss two key units in our model for justification.

3.1 Model Framework

Fig. 2 provides an overview of DANSER. We will go into the details of each part in order.

Raw Input and Item Implicit Network. The model requires user-item interaction matrix R and user social network G_U as input. Social recommendation methods focus on leveraging the user social network to solve the data sparsity problem in recommender systems. Most existing methods treat items independently since there is no prior information that explicitly expresses the relationship between items. One way to calculate the similarity or relevance between two items is by the common users who clicked or rated them [26]. For any item i and item j, we define their similarity coefficient s_{ij} as the number of users who clicked both items. These coefficients induce an equivalence relation over items as follows: item i is related to item j if $s_{ij} > \tau$ with τ a fixed threshold. We define the item implicit network as the graph $G_I = (V_I, E_I)$ where V_I is the set of items and E_I is the set of edges that connects two related items.

Embedding Layer. The raw input of each user (resp. item) is a one-hot vector with high dimension, and the embedding operation is to project each user to a low-dimensional representation. Inspired by [15] and [2], each user can be represented by a user-specific embedding and the items rated by her. The former representation reflects user's interests while the latter one (called *item-based user embedding*) captures implicit influence of her rated history on current decision. Such embedding could deal with false-negative samples and improve model accuracy [15]. Similarly, each item can be represented by an item-specific embedding and the users who rated it (called *user-based item embedding*).

For the user-specific embedding, we define $\mathbf{P} = \{\mathbf{p}_u\}_{D\times M}$, where D is the embedding dimension and $\mathbf{p}_u$ denotes the embedding vector for user u. For the item-based user embedding, we define another embedding lookup $\mathbf{Y} = \{\mathbf{y}_i\}_{D\times N}$, where $\mathbf{y}_i$ denotes the item latent factor. Then each item in $R_I(u)$ will be mapped via $\mathbf{Y}$ and outputs $\mathcal{Y}_u = \{\mathbf{y}_j | j \in R_I(u)\}$, which will be used to compute the item-based user embedding, a D-dimensional vector representation, in the next layer. The user embeddings reflect a user's inherent preference (the 'inherent' notion emphasizes that it contains no social information and only depends on the user oneself). Similarly, for the item-specific embedding, we define $\mathbf{Q} = \{\mathbf{q}_i\}_{D\times N}$, where $\mathbf{q}_i$ denotes the embedding vector for item i. For user-based item embedding, we define $\mathbf{X} = \{\mathbf{x}_u\}_{D\times M}$, where $\mathbf{x}_u$ is user latent factor, and each user in $R_U(i)$ will be mapped via $\mathbf{X}$ and outputs $\mathcal{X}_i = \{\mathbf{x}_v | v \in R_U(i)\}$, which will be used to compute user-based item embedding, a D-dimensional vector representation, in the next layer. Also, the item embeddings reflect item's inherent attribute.

Dual GCN/GAT Layer. Graph convolution network (GCN) [14] conducts local convolutional operation over neighbor nodes in graph and outputs a new representation for each node, in order to encode the graph structure information as low-dimensional node

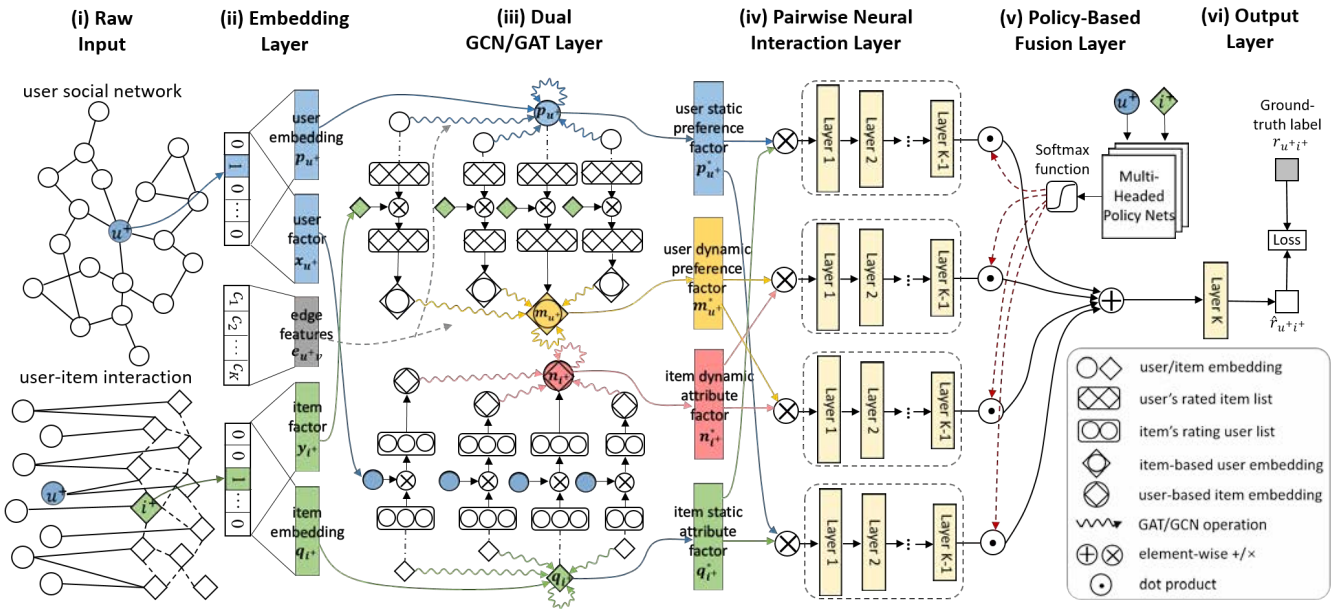


Figure 2: DANSER framework. Blue circles denote a targeted user u^+ while green rhombuses denote a candidate item i^+. i) The model requires user-item interaction records and user social network as raw input. We use common users who rate both items to calculate the item-item relevance and link the related items to form an item implicit network (the dotted lines between items). ii) In the embedding layer, we represent one user (resp. item) as a low-dimensional embedding vector and a latent factor. Besides, interaction frequencies between users are used as edge features. iii) In the dual GCN/GAT layer, four different graph attention networks are to capture the two two-fold social effects, where the upper (resp. lower) two of them output representations for user (resp. item) static and dynamic preferences (resp. attributes) under the effect of homophily and influence, respectively. iv) These four deep factors will be pairwisely combined as four interacted features, which are then fed into four independent neural networks to obtain more condensed representations. v) Then a policy net with the input of item i^+'s and user u^+'s embeddings as context information outputs weights for four interacted features, which will be aggregated as one synthetic vector. vi) Finally, the synthetic vector is input into the output layer to give the final predicted score $\hat{r}_{u^+i^+}$.

representations. Such operation can be viewed as an extension of convolution neural network (CNN) from a grid structure to general graphs. GCN equally aggregates the neighbors' embedding in each convolution and treats each neighbor nodes with equal importance. In contrast, Graph Attention Network (GAT) [31] leverages attention mechanism to consider different weights from neighbor nodes, which enables the model to filter out noises and focus on important adjacent nodes.

In order to exploit user social network G_U and item implicit network G_I, we propose dual graph attention networks, an extension of GCN/GAT as a means to non-linearly aggregate the embeddings of the neighbor nodes in both networks, so that the social effects from friends and related items can propagate to adjacent nodes through the graphs. As discussed in Section 1, we can distinguish two types of social effects for users (resp. items) in social recommender systems: *homophily* effect and *influence* effect. They jointly affect user preferences and item attributes in different ways. DANSER uses two dual GATs to collaboratively learn different deep representations for user static/dynamic preference and item static/dynamic attribute.

I. GAT to capture social homophily (marked as blue in Fig. 2). Firstly, via user embedding, we have $\mathbf{P}$ as representation of inherent user preference factor. Then the GCN/GAT operation could output a new representation, the user static preference factor $\mathbf{P}^*$, by

$$\mathbf{P}^* = \sigma(\mathbf{A}_P(G_U)\mathbf{P}\mathbf{W}_P^T + \mathbf{b}_P), \tag{1}$$

where $\sigma, \mathbf{W_P}, \mathbf{b_P}$ are activation function, weight matrix, bias vector respectively, and $\mathbf{P}^*$ is the updated representation of users, which incorporates social information by using the attention weights $\mathbf{A_P(G_U)} = \{\alpha^P_{uv}\}_{M\times M}$ obtained from G_U. Its elements are defined as follows:

$$\alpha^P_{uv} = \frac{attn_U(\mathbf{W}_P\mathbf{p}_u, \mathbf{W}_P\mathbf{p}_v, \mathbf{W}_E\mathbf{e}_{uv})}{\sum_{w\in\Gamma_U(u)} attn_U(\mathbf{W}_P\mathbf{p}_u, \mathbf{W}_P\mathbf{p}_w, \mathbf{W}_E\mathbf{e}_{uv})}, v \in \Gamma_U(u),$$

where $\mathbf{W}_E$ is weight matrix, $\Gamma_U(u) = \{u\} \cup F(u)$, $attn_U(x, y, z) = LeakyReLu(\mathbf{w}_U^T z \otimes (x||y))$, and $\mathbf{w}_U$ is weight vector. Here $\otimes$ and $||$ denote the element-wise product and concatenation, respectively. Note that the above GAT weight α^P_{uv} remains unchanged given user u and v, which means factor matrix $\mathbf{P}^*$ is fixed for users. This design conforms to the intuition that social homophily contributes to static effect for user's preference, this is why we call it static preference factor.

II. GAT to capture social influence (marked as yellow in Fig. 2). In contrast to social homophily, social influence effect is often context-aware, and the model needs to output different attention weights for friends w.r.t different candidate items. Through the embedding layer, we have $\mathcal{Y}_u$ as the embedding of the items clicked by user

u. Then we let each item clicked by user u interact with candidate item i^+,

$$\mathcal{Y}_u^{i^+} = \{\mathbf{y}_j \otimes \mathbf{y}_{i^+} | j \in R_I(u)\}.$$

This product operation can help to focus on the candidate item and model dynamic social influence under a specific context. We define the item-based user embedding $\mathbf{M}_{i^+} = \{\mathbf{m}_u^{i^+}\}_{D\times M}$, which depends on candidate item i^+ with max pooling to select the most dominating features for D dimensions:

$$m_{ud}^{i^+} = \max_{j\in R_I(u)} \{y_{jd} \cdot y_{i^+d}\} \quad \forall d = 1, \ldots, D$$

where $m_{ud}^{i^+}$, y_{i^+d}, and y_{jd} are the d-th feature of $\mathbf{m}_u^{i^+}$, $\mathbf{y}_{i^+}$, $\mathbf{y}_j$ respectively. The max pooling operation can help to focus on the most important value and alleviate noises in users' clicked history. The item-based embedding $\mathbf{m}_u^{i^+}$ includes two information: i) user u's context-aware preference (w.r.t candidate item i^+), and ii) inherent representation for u (independent of social information). In order to define the user dynamic preference factor $\mathbf{M}_{i^+}^*$, we proceed to incorporate the social information from friend users,

$$\mathbf{M}_{i^+}^* = \sigma(\mathbf{A}_M(G_U)\mathbf{M}\mathbf{W}_M^T + \mathbf{b}_M), \mathbf{A}_M(G_U) = \{\alpha_{uv,i^+}^M\}_{M\times M},$$

$$\alpha_{uv,i^+}^M = \frac{attn_U(\mathbf{W}_M\mathbf{m}_u^{i^+}, \mathbf{W}_M\mathbf{m}_v^{i^+}, \mathbf{W}_E\mathbf{e}_{uv})}{\sum_{w\in\Gamma_U(u)} attn_U(\mathbf{W}_M\mathbf{m}_u^{i^+}, \mathbf{W}_M\mathbf{m}_w^{i^+}, \mathbf{W}_E\mathbf{e}_{uv})},$$

for $v \in \Gamma_U(u)$. Note that the above attention weight α_{uv,i^+}^M depends on the user's history of rated items as well as specific candidate item i^+, which indicates that factor matrix $\mathbf{M}_{i^+}^*$ would change dynamically with different contexts. This design conforms to the intuition that social influence contributes to context-aware effect for a user's preference, so we term it as dynamic preference factor.

III. GAT to capture item-to-item homophily (marked as green in Fig. 2). Similarly, we use the item embedding $\mathbf{Q}$ as representation of inherent item attribute factor, and then leverage GAT to incorporate social information,

$$\mathbf{Q}^* = \sigma(\mathbf{A}_Q(G_I)\mathbf{Q}\mathbf{W}_Q^T + \mathbf{b}_Q), \mathbf{A}_Q(G_I) = \{\alpha_{ij}^Q\}_{N\times N},$$

$$\alpha_{ij}^Q = \frac{attn_I(\mathbf{W}_Q\mathbf{q}_i, \mathbf{W}_Q\mathbf{q}_j)}{\sum_{k\in\Gamma_I(i)} attn_I(\mathbf{W}_Q\mathbf{q}_i, \mathbf{W}_Q\mathbf{q}_k)}, j \in \Gamma_I(i),$$

where $attn_I(x, y) = LeakyRelu(\mathbf{w}_I^T(x||y))$, and $\mathbf{w}_I$ is a weight vector. The GAT weight α_{ij}^Q remains unchanged given item i and j. Correspondingly, item-to-item homophily contributes to static effect for item's attribute, and we call it static attribute factor.

IV. *GAT to capture item-to-item influence (marked as red in Fig. 2).* We now model item-to-item influence, which is context-aware and depends on the specific targeted user. Hence, our model needs to output different attention weights for distinct related items w.r.t different targeted users. Similarly to social influence, the user-based item representation $N_{u^+} = (n_i^{u^+})_{D\times N}$ for a given targeted user u^+ can be defined as follows:

$$\mathcal{X}_i^{u^+} = \{\mathbf{x}_v \otimes \mathbf{x}_{u^+} | v \in R_I(i)\},$$

$$n_{id}^{u^+} = \max_{v\in R_U(i)} \{x_{vd} \cdot x_{u^+d}\} \quad \forall d = 1, \ldots, D,$$

where $n_{id}^{u^+}$, x_{u^+d}, and x_{vd} are the d-th feature of $\mathbf{n}_i^{u^+}$, $\mathbf{x}_{u^+}$, $\mathbf{x}_v$ respectively. Then the representation for item dynamic attribute $\mathbf{N}_{u^+}^*$ can be computed as follows:

$$\mathbf{N}_{u^+}^* = \sigma(\mathbf{A}_N(G_U)\mathbf{N}_{u^+}\mathbf{W}_N^T + \mathbf{b}_N), \mathbf{A}_N(G_U) = \{\alpha_{ij,u^+}^N\}_{N\times N},$$

$$\alpha_{ij,u^+}^N = \frac{attn(\mathbf{W}_N\mathbf{n}_i^{u^+}, \mathbf{W}_N\mathbf{n}_j^{u^+})}{\sum_{k\in\Gamma_I(i)} attn(\mathbf{W}_N\mathbf{n}_i^{u^+}, \mathbf{W}_N\mathbf{n}_k^{u^+})}, j \in \Gamma_I(i).$$

Pairwise Neural Interaction Layer. Since user's decision on one item often depends on both user preference and item attribute, we can let two preference factors and two attribute factors pairwisely interact with each other. Then we borrow the idea from [11], feeding the four results into different neural networks indexed by $a \in \{1, 2, 3, 4\}$:

$$\mathbf{s}_a = \phi_K^a(\cdots\phi_2^a(\phi_1^a(\mathbf{z}_0[a]))),$$
$$\phi_k^a(\mathbf{z}_{k-1}) = tanh(\mathbf{W}_k^a\mathbf{z}_{k-1}^a + \mathbf{b}_k^a), k \in [1, K-1],$$
$$\mathbf{z}_0 = [\mathbf{p}_u^* \oplus \mathbf{q}_i^*, \mathbf{p}_u^* \oplus \mathbf{n}_i^*, \mathbf{m}_u^* \oplus \mathbf{q}_i^*, \mathbf{m}_u^* \oplus \mathbf{n}_i^*],$$

Here we employ a tower structure for each network, where higher layers have smaller number of neurons [11].

Policy-Based Fusion Layer. We further fuse the four interacted features $\mathbf{s}_a$ into a synthetic one. Note that homophily effect and influence effect could jointly affect user preference and item attribute, but for distinct users and items, the importance of the two-fold social effects could be different. To this end, we propose a new policy-based fusion strategy to dynamically allocate weights to the four interacted features according to specific user-item pairs.

We model the weight allocation as a contextual multi-armed bandit problem[1],where an action, denoted by $\gamma \in \{1, 2, 3, 4\}$, indicates which feature to choose, a context is a user-item pair and the reward after playing an action represents a recommendation loss. In this problem, a stochastic policy can be written as the conditional probability $p(\gamma|\mathbf{p}_u, \mathbf{q}_i)$, which could characterize the selecting probability or importance weights for different combinations of social effects given a specific pair (u, i).

To make the problem solvable, we approximate $p(\gamma|\mathbf{p}_u, \mathbf{q}_i)$ by a neural network (called policy network):

$$e_\gamma = \mathbf{W}_F^2 \tanh(\mathbf{W}_F^1(\mathbf{p}_u||\mathbf{q}_i) + \mathbf{b}_F^1) + \mathbf{b}_F^2.$$

$$p(\gamma|\mathbf{p}_u, \mathbf{q}_i) = \frac{\exp(e_\gamma)}{\sum_{a=1}^4 \exp(e_a)}.$$

Then the synthetic representation can be expressed as

$$\mathbf{s} = \mathbb{E}_{\gamma\sim p(\gamma|\mathbf{p}_u,\mathbf{q}_i)}(\mathbf{s}_\gamma) = \sum_{\gamma=1}^4 p(\gamma|\mathbf{p}_u, \mathbf{q}_i) \cdot \mathbf{s}_\gamma.$$

We call the above strategy *single-headed* policy-based fusion. Inspired by [30], we can extend it to a *multi-headed* version. We harness L different independent policy networks and the final weights are given by the averaged weights given by each policy net. The training of policy networks is by stochastic policy gradient, which will be discussed in Section 3.2.

Output Layer. Then probability of user u clicking item i can be predicted by

$$\hat{r}_{ui} = nn(\mathbf{s}).$$

If a clicking probability is required (for implicit feedback), $nn(\cdot)$ can be a fully-connected layer with a sigmoid activation function.

If a rating value is needed (for explicit feedback), $nn(\cdot)$ can be a fully-connected layer without activation function.

Loss Function. The loss function measures the discrepancy between predicted value and ground-truth value. For implicit feedback, the most widely adopted loss function is the cross-entropy defined as

$$\mathcal{L}_1 = -\sum_{(u,i)} r_{ui} \log \hat{r}_{ui} + (1 - r_{ui}) \log(1 - \hat{r}_{ui}).$$

For explicit feedback, we adopt the mean square loss

$$\mathcal{L}_1 = \sum_{(u,i)} \|\hat{r}_{ui} - r_{ui}\|^2.$$

Here, we concentrate on pointwise loss. Also, other pairwise loss functions can be used and we leave them for future investigation.

3.2 Training

In this subsection, we introduce some training techniques to improve the model performance.

Mini-Batch Training. We also leverage mini-batch training to calculate the gradient. For each iteration, we consider B user-item pairs. For each pair (u, i), the neighbor nodes $F_U(u)$, $F_I(i)$ in two networks as well as the rated history $R_I(v)$, $v \in \Gamma_U(u)$, and $R_U(j)$, $j \in \Gamma_I(i)$ will be together input into the model to compute the gradient. Once an epoch is finished, the batch partition will be randomly reset to incorporate enough noise during optimization. In each mini-batch, since numbers of friends are different for users, we need to pad zeros and input a matrix with dimension of max friend number in one batch. We observe that number of friends tends to obey a long-tail distribution, so the padding would lead to both high space and time complexity. Therefore, we use a sample technique: 1) for friend number more than F, we uniformly sample F friends as input data; 2) for friend number less than F, we pad zeros to obtain a F-dimensional vector. We also experiment other sample methods like random walk with restart (RWR) [29] on a sub-graph near central node, but we find that uniformly sampling over adjacent nodes can achieve competitive performance. This observation could be justified by one argument in sociology research that influence from users always propagates to adjacent users in social networks [7]. Thus we adopt the simpler method to reduce time complexity. In experiment part, we will investigate the impact of sample size F on the performance.

Local-Graph Aware Regularization. The other issue is regularization to avoid over-fitting problem. In our model, we adopt L1 regularization to constrain the embedding parameters to sparse forms. The regularization loss can be expressed as

$$\mathcal{L}_2 = \sum_u (\|p_u\| + \|x_u\|) + \sum_i (\|q_i\| + \|y_i\|).$$

However, the above loss introduces high computational complexity, since for each batch training we need to calculate the gradient of regularization over all users and items. Such operation can not generalize to large-scale data set. A recent work [45] proposes a mini-batch aware regularization trick to solve this problem. In each batch training, only the embedding parameters for user-item pair (u, i) in the batch will be used to calculate the regularization loss. Here we extend this idea to graph structure and call it as local-graph aware regularization. In each batch training, we constrain the embedding parameters of users, items as well as their neighbor nodes in social network and item implicit network. The new regularization loss can be written as

$$\mathcal{L}_2 = \frac{1}{2} \sum_{(u,i)} \Big[\|\mathbf{p}_u\| + \|\mathbf{x}_u\| + \sum_{v \in \Gamma_U(u)} \frac{1}{|F_U(v)|} (\|\mathbf{p}_v\| + \|\mathbf{x}_v\|) + \|\mathbf{q}_i\| + \|\mathbf{y}_i\| + \sum_{j \in \Gamma_I(i)} \frac{1}{|F_I(j)|} (\|\mathbf{q}_j\| + \|\mathbf{y}_j\|) \Big].$$

In fact, the local-graph aware regularization is to split the regularization loss into small parts for each mini-batch and guarantees that once an epoch is finished, each user and each item can be regularized equally. To sum up, the final loss function is

$$\mathcal{L} = \mathcal{L}_1 + \lambda \mathcal{L}_2, \tag{2}$$

where λ is a trade-off parameter between accuracy and complexity.

Besides, for parameters in neural networks (weight matrix and bias vectors), we adopt dropout strategy to replace the traditional regularization. Our experiments will further discuss the influence of different dropout probabilities ρ.

Policy Gradient. In training stage, we train the policy networks in a stochastic way. Assume $p_l(\gamma|\mathbf{p}_u, \mathbf{q}_i)$ to be the active probability given by l-th policy network, and random variable γ obeys a multinomial distribution. For each policy network, we draw $\gamma \sim Multi(p_l(\gamma|\mathbf{p}_u, \mathbf{q}_i))$, and feed $\mathbf{s}_\gamma$ into output layer, which gives prediction and returns loss. We leverage the loss to train the policy network. This design can be formulated as an actor-critic terminology: i) given the environment (user u and item i), the l-th agent (policy net) needs to execute an action according to policy $(p_l(\gamma|\mathbf{p}_u, \mathbf{q}_i))$; ii) every action would generate a reward (recommendation loss) that could lead the training of policy network. We define the reward as subtraction of loss generated by selected interacted feature, i.e., $\mathcal{R}(p_u, q_i, \gamma) = -\mathcal{L}(\mathbf{s}_\gamma)$ (where $\mathcal{L}(\mathbf{s}_\gamma)$ denotes the loss when we input $\mathbf{s}_\gamma$ into the output layer). We aim at maximizing the expected reward $\mathbb{E}_{\gamma \sim p(\gamma|p_u, q_i)}(\mathcal{R}(p_u, q_i, \gamma))$. The policy gradient method REINFORCE [35] can be used to update parameters in one policy network (denoted as θ), and the gradient can be derived as follows:

$$\begin{aligned} &\nabla_\theta \mathbb{E}_{\gamma \sim p_\theta(\gamma|\mathbf{p}_u, \mathbf{q}_i)}(\mathcal{R}(\mathbf{p}_u, \mathbf{q}_i, \gamma)) \\ \simeq &\frac{1}{4} \sum_\gamma \nabla_\theta \log p_\theta(\gamma|\mathbf{p}_u, \mathbf{q}_i) \mathcal{R}(\mathbf{p}_u, \mathbf{q}_i, \gamma). \end{aligned} \tag{3}$$

Training Algorithm. The training of DANSER updates policy networks and feedforward networks (including embedding, dual GAT, pairwise neural interaction and output layers) iteratively, which is presented in Alg. 1. First, we update feedforward networks for n_p steps (one mini-batch for one step) by optimizing (2). Then we train L policy networks independently by conducting corresponding policy gradient (3).

3.3 Discussions

In this subsection, we discuss on effectiveness of our key units, dual GAT layer and policy-based fusion layer, in order to shed more insights on DANSER.

Algorithm 1: Training for DANSER

REQUIRE: $\mathbf{R}$, observed user-item interaction. G_U, user social network. G_I, item implicit network.
REQUIRE: η, ζ, learning rates.
while *not converged* **do**
  for $i = 1, \cdots, n_p$ **do**
    Sample B user-item pairs.;
    For each pair (u, i), uniformly sample F neighbors from $F_U(u)$ and $F_I(i)$, respectively;
    Draw $\gamma_l \sim Multi(p^l_{\theta_l}|\mathbf{p}_u, \mathbf{q}_i), l = 1, \cdots, L$;
    $w \leftarrow \eta \nabla_w \sum_l L(\mathbf{s}_{\gamma_l})$;
  $\theta_l \leftarrow \zeta \nabla_{\theta_l} \mathbb{E}_{\gamma \sim p^l_{\theta_l}(\gamma|\mathbf{p}_u, \mathbf{q}_i)}(\mathcal{R}(\mathbf{p}_u, \mathbf{q}_i, \gamma)), l = 1, \cdots, L$;

Justification of Dual GATs. During the training process, the GAT to capture the homophily effect would give larger weights to friends whose preferences for items are globally similar to the targeted user (resp. related items whose attractiveness to users are globally similar to candidate item), so the static preference (resp. attribute) factor captures static effect of friend users (resp. related items) from a global view. Differently, the weights given by the GAT to capture the influence effect depend on the targeted user (resp. item), so the corresponding dynamic factor captures context-aware effects from a local view. In each domain, a dual GAT collaboratively models the two-fold effects and takes both global and local views to conduct a more accurate investigations into distinct users and items.

To put it more symbolically, we illustrate its effectiveness by considering the following two cases for the recommendation of candidate item i^+ to targeted user u^+: item i^+ is similar to item i_1 and i^+ is similar to i_2 (see left and right columns of Fig. 3). Here, we assume that user u^+ has only clicked on i_1 and i_2 (and no other items), and users u_1 to u_4 are u^+'s friends. Previous works for social recommendation consider static effect of friends in an average manner. As illustrated in Fig. 3.(a), for friend u_1 who clicked on both items, the attention weight would be large, and the opposite for friend u_4. For u_2 and u_3, their weights are medium since they are both partially similar to u^+. Therefore static weights cannot discriminate between the influence of user u_2 and u_3 when it comes to item i^+ even if it is known that it is similar to i_1 or i_2. In contrast, DANSER contains two different attention weights, where one captures user-specific static effect (for homophily) and the other can dynamically change under different contexts (i.e., for influence). When candidate item i^+ is similar to i_1 (resp. i_2), the social influence GAT would focus more on u_1 and u_2 (resp. u_1 and u_3). Then the resultant force of the two GATs would make u^+ pay more attention to u_2 when i^+ is similar to i_1 and to u_3 in the other case. The global and local views of DANSER work collaboratively for selecting the 'right' neighbors that are similar to the targeted user under a specific context, which improves model expressiveness. In Section 4.5, we will verify the above argument via a real-case study.

Justification of Policy-Based Fusion. The policy-based fusion layer acts like a stochastic network while training, and the REINFORCE algorithm can help to explore the optimal weights distribution over four interacted features. Since the interacted feature with larger active probability is more likely to be selected, it would be more frequent for the interaction neural layer that gives the 'best' feature to be trained. Such mechanism brings up two major benefits. Firstly, the policy networks can filter 'right' user-item samples for training of corresponding interaction neural layer as well as previous GCN/GAT and embedding layers. Second, the active probability introduces random pattern to the training process, which can help to escape from local optimums. Our ablation study in experiment part also validate the superiority of the policy-based fusion compared with other straightforward strategies.

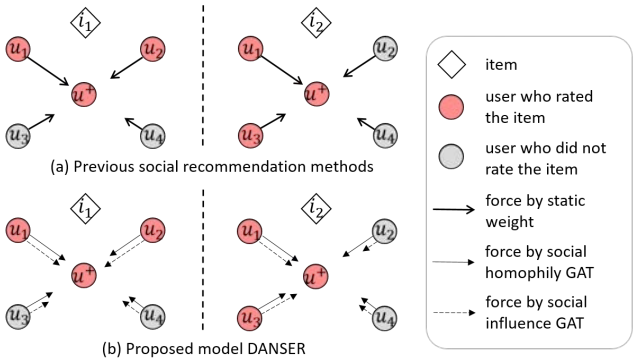


Figure 3: Illustration of effectiveness of dual GATs and comparison with previous social recommendation methods.

4 EXPERIMENTS

To comprehensively evaluate proposed model DANSER[2], we conduct experiments to answer the following research questions:
RQ1 How does DANSER perform compared with state-of-the-art models for recommendation and social recommendation?
RQ2 Are the key components in DANSER, such as dual GATs and policy-based fusion, necessary for improving performance?
RQ3 How do hyper-parameters in DANSER impact recommendation performance?
RQ4 How can DANSER interpret the four different social effects from real-world data?

4.1 Experiment Setup

4.1.1 Data sets. We apply our model to one public benchmark datasets for social recommendation *Epinions* and a commercial dataset *WeChat Top Story*. We provide some statistics of these two datasets in Table 1. The basic information about the two datasets is summarized as follows:

i) *Epinions:* Epinions is a consumer review websites, where users can rate some items and add other users in their trust lists. *Epinions* dataset [23] contains two parts of information: user-item interaction pairs, where items are rated from 1 to 5 (explicit feedback), as well as the directed trust relationships between users. The dataset has been widely used as benchmarks for social recommendation. Following other works [10, 17, 33, 38], we randomly choose 80% user-item interactions as training set and the remaining 20% as test set for each dataset.

[2]The codes are released at https://github.com/echo740/DANSER-WWW-19.

ii) *WeChat:* We also deploy our model on a real-world article recommender system, WeChat Top Story. This dataset is constructed by user-article clicking records on this platform. Different from *Epinions*, this dataset only contains implicit feedback, i.e., we only know whether a user clicked an article or not. Besides, the friend relationships are bidirectional. We chronologically order the user-article clicking records, and use the first 90% records to train our model and the remaining records to evaluate the prediction performance. Since the positive samples and negative samples are fairly unbalanced in this dataset, we uniformly sample the negative ones such that the numbers of positive and negative samples are the same for one user.

Table 1: Statistics of three datasets.

Dataset	#users	#items	#interactions	#relationships
Epinions	49,290	139,738	664,824	487,181
WeChat	~200,000	~100,000	~4,000,000	~2,000,000

4.1.2 Implementation Details. We use Tensorflow to implement our model and deploy it on GTX 1080 GPU with 8G memory. The hyper-parameter settings for *Epinions* are as follows: batch size $B = 64$, embedding dimension $D = 10$, dropout ratio $\tau = 0.5$, regularization coefficient $\lambda = 0.001$, sample size $F = 30$, truncation length $C_t = 30$, multi-head number $L = 4$, policy gradient period $n_p = 1000$. The learning rates $\eta = 0.1$, $\zeta = 0.01$. The Leaky ReLu slope is 0.2. In pairwise interaction neural layer, we adopt three-layer neural networks and neuron numbers are $10 - 16 - 8 - 4$.

4.1.3 Competitors. We choose several comparative methods, including some state-of-the-art models for recommendation and social recommendation, to evaluate the performance of our DANSER. The optimal hyper-parameter settings for each method are determined either by our experiments or suggested by previous works.

The recommendation models include *SVD++*[15] and *DELF*[2]. These two methods only leverage user-item interactions as model input. They can be used to evaluate the effectiveness of other models that use social information.

- *SVD++* [15] is a basic model-based recommendation method which use both user-specific and item-based user embedding to represent user's preferences.
- *DELF* [2] is a state-of-the-art CF method which proposes dual embedding for users and items, and adopts deep neural networks to capture complex non-linear information.

The social recommendation models include *TrustPro*, *TrustMF*, *TrustSVD*, *NSCR*, *SREPS*. These methods all leverage both user-item interactions and user friendships as input information.

- *TrustPro* [39] is a trust propagation method, using rating of friends to deduce rating of targeted user.
- *TrustMF* [38], as one matrix factorization method, optimizes user embedding to retrieve the trust matrix.
- *TrustSVD* [10] is another matrix factorization-based method, incorporating friends' embedding vectors into targeted user's predicted rating.
- *NSCR* [33], as a strong baseline for social recommendation, adopts deep neural networks to learn latent representations of users and items, and leverages graph regularization to constrain the embedding of adjacent users to be similar.
- *SREPS* [17] is another strong baseline, using a network embedding approach to encode social network.

We also construct several variants of DANSER as ablation study. To evaluate the necessity of dual GAT layer, we simplify our model as *DualEMB*, *DualGCN*, *userGAT*, and *itemGAT*. *DualEMB* removes the convolution over adjacent nodes, while *DualGCN* replace GAT by GCN. *userGAT* removes two GATs in item domain, and *itemGAT* removes two GATs in user domain. Moreover, to evaluate the necessity of policy-based fusion layer, we consider the following variants. *DANSER-m* and *DANSER-a* leverage max and average pooling to fuse four interacted features, respectively. *DANSER-c* concatenates the four interacted features as the input of output layer. *DANSER-w* removes the policy net, set four sharing fusion weights, and update the weights together with training of feed-forward networks.

4.1.4 Evaluation Protocol. We adopt different metrics to evaluate recommendation performance. Since two datasets possess different feedbacks, we consider different metrics for them. For *Epinions* with explicit feedback, we use *MAE* and *RMSE*, which are widely adopted by other works, as evaluation metrics. For *WeChat* with implicit feedback, we use *Precision@k* (short as P@k) and *AUC*, two universally acknowledged metrics for 0-1 classification, to evaluate the performance. We repeated each experiment ten times and report the average performance.

4.2 Comparative Results: RQ1

We report experiment results of DANSER and other comparative methods in Table 2. As we can see, our model DANSER outperforms other comparative methods and achieves great improvements, i.e., 2.87% MAE impv. for Epinions and 4.48% AUC impv. for WeChat. The results verify the superiority of proposed model for social recommendation. Also, there are some findings in these comparative experiments. First, matrix factorization methods like TrustMF and TrustSVD can achieve good accuracy in Epinions with explicit feedback, but perform poorly in WeChat with implicit feedback. Second, NSCR performs quite better than other linear methods, which indicates that neural network plays an important role in capturing complex non-linear information. Third, even though DELF has not leveraged any social information, it outperforms some social recommendation methods like TrustSVD and NSCR. This is possibly due to the facts: i) the dual embedding and neural interaction layer in DELF can better exploit information in user-item interactions; ii) other social recommendation methods like TrustSVD and NSCR do not make good use of social information. Indeed, TrustSVD assumes equal importance of each friend user while NSCR uses graph regularization that constrains representations for adjacent users in a static way. These settings resort to a prior assumption that friend users' preferences are all similar, which may not be true for contemporary SNS (also discussed in Section 1).

4.3 Ablation Study: RQ2

In order to verify the effectiveness of some components in our model, we do some ablation studies and the results are shown in

Table 2: Comparative results for Epinions and WeChat. For MAE, RMSE, the smaller value is better, and vice versa for P@10, AUC.

	Epinions		WeChat	
	MAE	RMSE	P@10	AUC
SVD++ [15]	0.8321	1.0772	0.0653	0.7304
DELF [2]	0.8115	1.0561	<u>0.0752</u>	<u>0.7818</u>
TrustPro [39]	0.9130	1.1124	0.0561	0.6482
TrustMF [38]	0.8214	1.0715	0.0625	0.7005
TrustSVD [10]	0.8144	1.0492	0.0664	0.7325
NSCR [33]	0.8044	1.0425	0.0736	0.7727
SREPS [17]	<u>0.8014</u>	<u>1.0393</u>	0.0725	0.7745
DANSER	**0.7781**	**1.0268**	**0.0823**	**0.8165**
Impv.[1]	2.87%	1.25%	9.33%	4.48%

[1]The improvement compares DANSER with the best competitor (underlined).

Table 3. We first compare DANSER with DualEMB and DualGCN. As we can see, GCNs can capture enough social information hidden in graph structure and improve recommendation accuracy (about 1.0% impv. for MAE over DualEMB), while GATs can further improve performance by differentiate importance weights for friend users (about extra 0.8% impv. for MAE over DualGCN). We secondly compare DANSER with userGAT and itemGAT. The results show that incorporation of user and item networks can both bring significant performance improvement, and the benefits brought by user social network are greater.

We also evaluate the necessity of policy-based fusion layer. The results show that our policy-based fusion and policy gradient method can improve the accuracy a lot. Also, for Epinions, the improvement is quite significant. One possible reason is that the social effects in Epinions are more heterogeneous than those in WeChat, so to dynamically weigh different social effects are more effective for Epinions. More importantly, since DualEMB can be seen as the extension of DELF [2], where we replace the concatenation of four embedding by our policy unit. Comparing DualEMB in Table 3 with DELF in Table 2, we find that the policy unit do help to improve its performance and is superior to straightforward concatenation.

In Section 1, we emphasized that social recommendation is proposed to solve the data sparsity and cold-start problem in recommender systems. Here we would like to investigate the performance of DANSER for cold-start users. Fig. 4 shows the results on Epinions and WeChat for users with different number of clicked items and friends. We present the median, the 75− and 25−percentile MAE/AUC. As we can see, with number of clicked items increasing, the gap between upper and lower bounds becomes narrow, which indicates that performance becomes better. Generally, DANSER can achieve better overall performance as well as lower variation of MAE/AUC. For users with different friend numbers, as depicted in the figure, when friend number goes up, the variation trends for three methods are different. For DualEMB, the performance improves a little. For DualGCN, there is an obvious performance decline, while performance of DANSER becomes better. This is

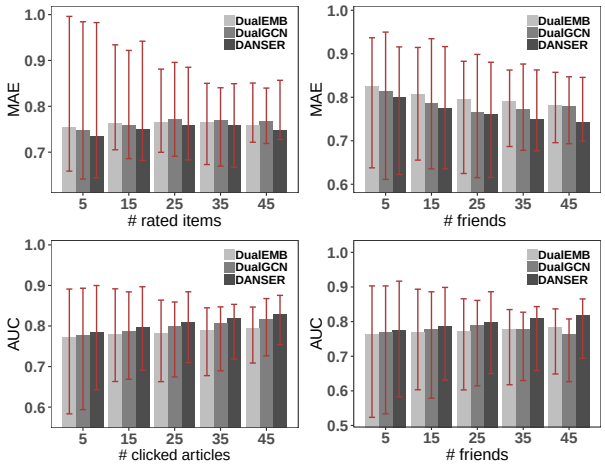


Figure 4: MAE/AUC of DANSER, DualGCN, DualEMB on Epinions/WeChat w.r.t users with different number of clicked items and friends.

Table 3: Ablation study of components in proposed method.

	Epinions		WeChat	
	MAE	RMSE	P@10	AUC
DualEMB	.7920(1.7%)[1]	1.0363(0.7%)	.0794(3.6%)	.7992(2.2%)
DualGCN	.7840(0.7%)	1.0335(0.4%)	.0814(1.1%)	.8102(0.8%)
userGAT	.7858(0.9%)	1.0364(0.7%)	.0813(1.2%)	.8136(0.4%)
itemGAT	.7919(1.7%)	1.0335(0.4%)	.0813 (1.2%)	.8138(0.3%)
DANSER-w	.8191(4.9%)	1.0659(3.4%)	.0820(0.4%)	.8151(0.2%)
DANSER-m	.8211(5.2%)	1.0681(3.6%)	.0815(1.0%)	.8144(0.3%)
DANSER-a	.8232(5.4%)	1.0710(3.9%)	.0814 (1.1%)	.8140(0.3%)
DANSER-c	.8091(3.8%)	1.0659(3.4%)	.0809(1.7%)	.8118(0.6%)
DANSER	**0.7787**	**1.0292**	**0.0823**	**0.8165**

[1]The ratio indicates the impv. comparing DANSER with corresponding variant.

possibly because for users with large number of friends, the GCN operation may incorporate more misleading information, and instead, GAT can attentively filter the noise and concentrate more on important neighbor nodes.

4.4 Parameter Sensitivity: RQ3

We study the performance variation for our model w.r.t some hyper-parameters including regularization parameter λ, dropout rate ρ, embedding dimension D, sample size F. (The other hyper-parameters have little impact on model performance, so we skip discussions on them for space limit.) The results are shown in Fig. 5 and enlighten us that. i) The regularization parameter plays an important role, and if it is too large, the model cannot focus on minimizing the recommendation loss. ii) If the dropout ratio is too large, the random pattern would hinder the training. If ρ is too small, it would lose the regularization efficacy. iii) A proper embedding size is needed. If it is too small, the model lacks expressiveness, while if it is too large, the representation vector would become so sparse, which leads to performance decline. iv) With the sample

size increasing, the accuracy improves but the computational cost increases as well. Hence an appropriate sample size can keep a good balance between complexity and accuracy.

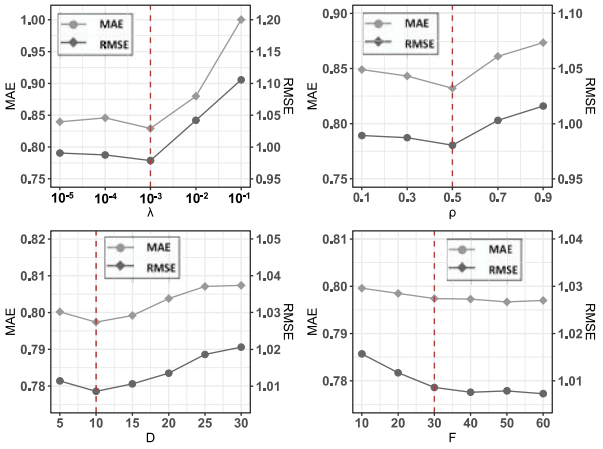


Figure 5: MAE of DANSER on Epinions w.r.t different hyper-parameters. The vertical dotted lines mark the values set in this paper.

4.5 Case Study: RQ4

We conduct a case study of four user-item pairs and show the weights given by four GATs by node size in Fig. 6. As we can see, the weights of two homophily GATs stay unchanged for the same user or item, while the weights of two influence GATs vary for different user-item pairs. Moreover, influence GATs incline to allocate larger weights to nodes with the same labels, which echoes the argument in Section 3.3 that the influence GATs tend to concentrate more on the similar nodes under a specific context. Also, the policy weights play a part in selecting the most dominant interacted features from four GATs representations.

More importantly, we can decompose the model interpretability for social effects in recommender systems by a two-level illustration: i) for a specific user-item pair (u, i), the policy weights first indicate the dominant combinations of social effects in user domain and item domain, ii) the corresponding GATs for four social effects further give out the influential degree of each friend given item i as well as the influential degree of each related item given user u.

5 RELATED WORKS

Social Recommendation. In online social networks, friends tend to have a large impact on users' decisions and behaviors [9, 27]. There are plenty of studies for social effects, such as homophily[24] and influence[21], over graph structures in social network analysis. For recommender systems, [13] is one of the pioneer who takes advantage of social networks to alleviate the data sparsity problem for item recommendation. Then many following studies begin to model social effects in recommender systems from various perspectives. [5, 12, 22] propose trust propagation methods by leveraging rating of friends to deduce rating of targeted user. This kind of approach is quite heuristic and is called *similarity-based* method. Then a more modern approach, *model-based* method, is proposed by other studies. [20, 40] use regularization to constrain the neighbor

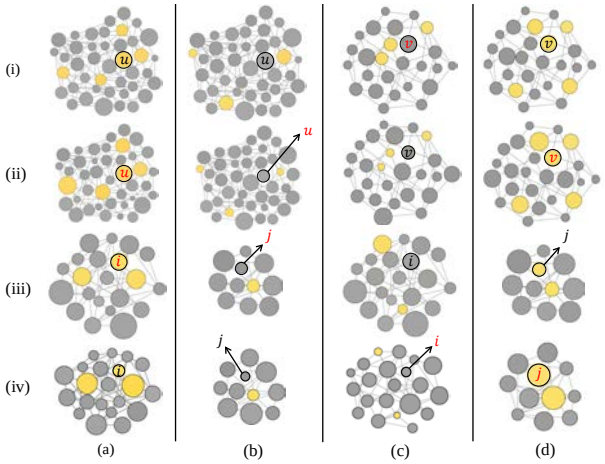


Figure 6: Visualization of the four GATs' weights for a case study. There are four user-item pairs: (a) (u, i), (b) (u, j), (c) (v, i), (d) (v, j). Row (i) and row (ii) are for social homophily and social influence GATs' weights, respectively, while row (iii) and row (vi) are for item-to-item homophily and item-to-item influence GATs' weights, respectively. Gray nodes are negative samples ($r_{ui} = 0$) while orange nodes are positive ones ($r_{ui} = 1$). In each line, the interacted feature of two factors with red symbol corresponds to the largest active probability given by policy networks.

users' embedding vectors to be similar and [8, 18, 38] adopt matrix factorization to incorporate the adjacent matrix into user embedding. Since the above methods can only capture linear information, some recent studies like [4, 33] and [17, 34, 43] leverage deep neural networks and network embedding approaches, respectively, to learn a more complex representation for graph structures. However, comparing with dual GATs used in this paper, the common limitations of existing studies lie in: i) they assume neighbors' influences to be equally important or statically constrained, ii) they ignore the social effects from related items, iii) modeling of social effects lacks interpretability.

Graph Convolution/Attention Network. GCN and GAT, as two powerful techniques to encode a complex graph into low-dimensional representations, are extensively applied into various problems involved with graph data since their proposals [3, 14, 31]. Using GCN and GAT to solve semi-supervised classification problem in graph could achieve state-the-of-art performance (GAT can be seen as an extension of GCN and provide better performance according to [31]), and its good scalability enables it to tackle large-scale dataset. Existing studies leverage GAT to tackle social influence analysis[25], graph node classification[6], conversation generation[46], relevence matching[43]. For recommendation, some recent studies like [42] adopt GCN to convolve on user-item network (a bipartite graph) to obtain better representations for items, and there are also some works using GCN to capture social information for recommendation [41][36]. We are the first to use GAT for social recommendation task, and our new architecture, dual GATs, can capture social information in both user and item networks.

Dual Mechanism. There are many dual phenomenons in real-life which inspire several dual structures in model design. [37] proposes a dual-model paradigm to merge the training of two dual tasks. [47] extends GCN to dual structures, combining global and local consistencies in graph. [2] proposes DELF conducting dual embedding for users and items in recommender systems. Compared to [2], our model possesses several key differences: i) for the item-based user embedding (resp. the user-based item embedding), DANSER considers dynamic embedding under specific candidate items (resp. targeted users), while DELF treats embedding statically given a user (resp. item); ii) in dual GCN/GAT layer, DANSER harnesses two dual GATs to capture social information in user social network and item implicit network, while DELF considers no user-to-user and item-to-item mutual influences; iii) in the policy-based fusion layer, DANSER dynamically weighs the importances of the four GAT representations, while DELF straightforwardly concatenates four embedding vectors.

6 CONCLUSION

In this paper, we proposed DANSER, which includes two dual graph attention networks, to learn deep representations for social effects in recommender systems. Our method is equipped with good expressiveness because: i) it can collaboratively model homophily and influence effects from both friend users and related items; ii) it can investigate heterogeneous importance distributions among different interactions of the two two-fold social effects for distinct user-item pairs. Our comparative experiments and ablation studies on a benchmark dataset and a commercial dataset showed that DANSER can learn effective representations of multifaceted social effects and leverage this information to significantly improve recommendation accuracy.

As extension of our work, we believe that the construction of item implicit network could leverage extra attribute features or distill more high-level connections through knowledge graphs if the information is available. More generally, one could extend DANSER to other information retrieval tasks, like question answering, where DANSER could model 'social effects' from related questions and similar answers given the corresponding networks of them.

REFERENCES

[1] Sébastien Bubeck and Nicolò Cesa-Bianchi. 2012. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends in Machine Learning* 5, 1 (2012), 1–122.

[2] Weiyu Cheng, Yanyan Shen, Yanmin Zhu, and Linpeng Huang. 2018. DELF: A Dual-Embedding based Deep Latent Factor Model for Recommendation. In *IJCAI*. 3329–3335.

[3] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NIPS*. 3837–3845.

[4] Wenqi Fan, Qing Li, and Min Cheng. 2018. Deep Modeling of Social Relations for Recommendation. In *AAAI*. 8075–8076.

[5] Jennifer Golbeck. 2006. Generating Predictive Movie Recommendations from Trust in Social Networks. In *iTrust*. 93–104.

[6] Liyu Gong and Qiang Cheng. 2018. Adaptive Edge Features Guided Graph Attention Networks. *CoRR* abs/1809.02709 (2018).

[7] Amit Goyal, Francesco Bonchi, and Laks V. S. Lakshmanan. 2010. Learning influence probabilities in social networks. In *WSDM*. 241–250.

[8] Yulong Gu, Jiaxing Song, Weidong Liu, Lixin Zou, and Yuan Yao. 2018. CAMF: Context Aware Matrix Factorization for Social Recommendation. *Web Intelligence* 16, 1 (2018), 53–71.

[9] Adrien Guille and Hakim Hacid. 2012. A predictive model for the temporal dynamics of information diffusion in online social networks. In *WWW*. 1145–1152.

[10] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. 2015. TrustSVD: Collaborative Filtering with Both the Explicit and Implicit Influence of User Trust and of Item Ratings. In *AAAI*. 123–129.

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*. 173–182.

[12] Mohsen Jamali and Martin Ester. 2009. TrustWalker: a Random Walk Model for Combining Trust-based and Item-based Recommendation. In *KDD*. 397–406.

[13] Henry A. Kautz, Bart Selman, and Mehul A. Shah. 1997. Referral Web: Combining Social Networks and Collaborative Filtering. *Communication of ACM* 40, 3 (1997), 63–65.

[14] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[15] Yehuda Koren. 2008. Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model. In *KDD*. 426–434.

[16] Dawen Liang, Laurent Charlin, James McInerney, and David M. Blei. 2016. Modeling User Exposure in Recommendation. In *WWW*. 951–961.

[17] Chun-Yi Liu, Chuan Zhou, Jia Wu, Yue Hu, and Li Guo. 2018. Social Recommendation with an Essential Preference Space. In *AAAI*. 346–353.

[18] Hao Ma, Irwin King, and Michael R. Lyu. 2009. Learning to Recommend with Social Trust Ensemble. In *SIGIR*. 203–210.

[19] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. 2008. SoRec: Social Recommendation using Probabilistic Matrix Factorization. In *CIKM*. 931–940.

[20] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender Systems with Social Regularization in Social Networks. In *WSDM*. 287–296.

[21] Peter V Marsden and Noah E Friedkin. 1994. Network studies of social influence. *Sociological Methods & Research* 22, 1 (1994), 127–151.

[22] Paolo Massa and Paolo Avesani. 2007. Trust-Aware Recommender Systems. In *KDD*. 17–24.

[23] Paolo Massa and Paolo Avesani. 2007. Trust-aware recommender systems. In *RecSys*. 17–24.

[24] Miller Mcpherson, Lynn Smith-Lovin, and James M. Cook. 2001. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology* 27, 1 (2001), 415–444.

[25] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. 2018. DeepInf: Social Influence Prediction with Deep Learning. In *KDD*. 2110–2119.

[26] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*. 285–295.

[27] Hossam Sharara, William Rand, and Lise Getoor. 2011. Differential Adaptive Diffusion: Understanding Diversity and Learning whom to Trust in Viral Marketing. In *ICWSM*.

[28] Jiliang Tang, Xia Hu, and Huan Liu. 2013. Social recommendation: a review. *Social Netw. Analys. Mining* 3, 4 (2013), 1113–1133.

[29] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast Random Walk with Restart and Its Applications. In *ICDM*. 613–622.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*. 6000–6010.

[31] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[32] Menghan Wang, Xiaolin Zheng, Yang Yang, and Kun Zhang. 2018. Collaborative Filtering With Social Exposure: A Modular Approach to Social Recommendation. In *AAAI*. 2516–2523.

[33] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item Silk Road: Recommending Items from Information Domains to Social Users. In *SIGIR*. 185–194.

[34] Yufei Wen, Lei Guo, Zhumin Chen, and Jun Ma. 2018. Network Embedding Based Recommendation Method in Social Networks. In *WWW*. 11–12.

[35] Ronald J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* 8 (1992), 229–256.

[36] Le Wu, Peijie Sun, Richang Hong, Yanjie Fu, Xiting Wang, and Meng Wang. 2018. SocialGCN: An Efficient Graph Convolutional Network based Model for Social Recommendation. *CoRR* abs/1811.02815 (2018).

[37] Yingce Xia, Xu Tan, Fei Tian, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2018. Model-Level Dual Learning. In *ICML*. 5379–5388.

[38] Bo Yang, Yu Lei, Dayou Liu, and Jiming Liu. 2013. Social Collaborative Filtering by Trust. In *IJCAI*. 2747–2753.

[39] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *SIGIR*. 325–334.

[40] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2010. A Matrix Factorization Technique with Trust Propagation for Recommendation in Social Networks. In *RecSys*. 135–142.

[41] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD*. 974–983.

[42] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD*. 974–983.
[43] Ting Zhang, Bang Liu, Di Niu, Kunfeng Lai, and Yu Xu. 2018. Multiresolution Graph Attention Networks for Relevance Matching. In *CIKM*. 933–942.
[44] Xingang Zhang, Qijie Gao, Christopher S.G. Khoo, and Amos Wu. 2013. Categories of Friends on Social Networking Sites: An Exploratory Study. In *A-LIEP*. 244–259.
[45] Guorui Zhou, Xiaoqiang Zhu, Chengru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *KDD*. 1059–1068.
[46] Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense Knowledge Aware Conversation Generation with Graph Attention. In *IJCAI*. 4623–4629.
[47] Chenyi Zhuang and Qiang Ma. 2018. Dual Graph Convolutional Networks for Graph-Based Semi-Supervised Classification. In *WWW*. 499–508.

Grid-based Evaluation Metrics for Web Image Search

Xiaohui Xie
BNRist, DCST, Tsinghua University
Beijing, China
xiexh_thu@163.com

Jiaxin Mao
BNRist, DCST, Tsinghua University
Beijing, China
maojiaxin@gmail.com

Yiqun Liu*
BNRist, DCST, Tsinghua University
Beijing, China
yiqunliu@tsinghua.edu.cn

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

Yunqiu Shao
BNRist, DCST, Tsinghua University
Beijing, China
shaoyunqiu14@gmail.com

Zixin Ye
Beihang University
Beijing, China
zixinye612@gmail.com

Min Zhang
BNRist, DCST, Tsinghua University
Beijing, China
z-m@tsinghua.edu.cn

Shaoping Ma
BNRist, DCST, Tsinghua University
Beijing, China
msp@tsinghua.edu.cn

ABSTRACT

Compared to general web search engines, web image search engines display results in a different way. In web image search, results are typically placed in a grid-based manner rather than a sequential result list. In this scenario, users can view results not only in a vertical direction but also in a horizontal direction. Moreover, pagination is usually not (explicitly) supported on image search search engine result pages (SERPs), and users can view results by scrolling down without having to click a "next page" button. These differences lead to different interaction mechanisms and user behavior patterns, which, in turn, create challenges to evaluation metrics that have originally been developed for general web search. While considerable effort has been invested in developing evaluation metrics for general web search, there has been relatively little effort to construct grid-based evaluation metrics.

To inform the development of grid-based evaluation metrics for web image search, we conduct a comprehensive analysis of user behavior so as to uncover how users allocate their attention in a grid-based web image search result interface. We obtain three findings: (1) "Middle bias": Confirming previous studies, we find that image results in the horizontal middle positions may receive more attention from users than those in the leftmost or rightmost positions. (2) "Slower decay": Unlike web search, users' attention does not decrease monotonically or dramatically with the rank position in image search, especially within a row. (3) "Row skipping": Users may ignore particular rows and directly jump to results at some distance. Motivated by these observations, we propose corresponding user behavior assumptions to capture users' search interaction processes and evaluate their search performance. We show how to derive new metrics from these assumptions and demonstrate that they can be adopted to revise traditional list-based metrics like Discounted Cumulative Gain (DCG) and Rank-Biased Precision (RBP). To show the effectiveness of the proposed grid-based metrics, we compare them against a number of list-based metrics in terms of their correlation with user satisfaction. Our experimental results show that the proposed grid-based evaluation metrics better reflect user satisfaction in web image search.

CCS CONCEPTS

• **Information systems → Evaluation of retrieval results**.

KEYWORDS

Web image search, User behavior, Evaluation metrics

ACM Reference Format:
Xiaohui Xie, Jiaxin Mao, Yiqun Liu, Maarten de Rijke, Yunqiu Shao, Zixin Ye, Min Zhang, and Shaoping Ma. 2019. Grid-based Evaluation Metrics for Web Image Search. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313514

1 INTRODUCTION

Image search has been shown to be very important within web search. Existing work shows that queries with an image search intent are the most popular on mobile phone devices and the second most popular on desktop and tablet devices [27]. In web image search a different type of search result placement is used compared to general web search, which results in differences in interaction mechanisms and user behavior. Let us consider the image search search engine result page (SERP) in Figure 1 to highlight three important differences: (1) An image search engine typically places results on a grid-based panel rather than in a one-dimensional ranked list. As a result, users can view results not only vertically but also horizontally. (2) Users can view results by scrolling down without having to click on the "next-page" button because the image search engine does not have an explicit pagination feature. (3) Instead of a snippet, i.e., a query-dependent abstract of the landing page, an image snapshot is shown together with metadata

*Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313514

about the image, which is typically only available when a cursor hovers on the result.

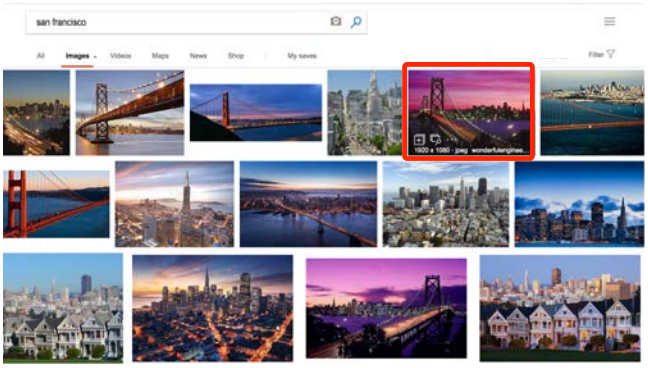


Figure 1: An example SERP from a popular image search engine. For the image that is highlighted using the red box, metadata is displayed when the user hovers over the image.

Evaluation metrics encapsulate assumptions about user behavior [10, 20] and, hence, differences in user behavior should lead to differences in the design of evaluation metrics in image search. Previous work on evaluation metrics [5, 15, 21] focuses on general web search scenarios where results are placed in a list manner. Among the evaluation metrics proposed, Rank-Biased Precision (RBP) [21] assumes that users will examine each result with a persistence probability p from top to bottom; users with a higher value of p are more patient to interact with search results. Discounted Cumulative Gain (DCG) [15] measures the gain of a document based on its position in the result list; the gain is accumulated from the top of the result list to the bottom and is discounted at lower ranks. Although these models work well to assess a result list in general web search, it is not obvious how to adapt them to image search where results are placed in a grid-based manner. Zhang et al. [39] show that the performance of these evaluation metrics is not promising in image search environments in the sense that they do not correlate well with user satisfaction. While the importance of different presentation formats has been recognized [23], there have been very few attempts to construct grid-based evaluation metrics.

As a first step towards designing better evaluation metrics for web image search, we conduct a comprehensive user behavior analysis using data from a lab-based user study so as to obtain a deeper understanding of the underlying user behavior, especially how users allocate their attention.

To summarize, we have three major findings through the analysis:

(1) Similar with the findings in [34], a middle position bias of users attention is observed in the user study data.
(2) The attention of image search users is not discounted monotonically and dramatically along with the rank positions, which means that more attention might not always be allocated to the higher rank positions. Also, the attention allocated to results within a row shows less variance than the attention among different rows.
(3) Users display row-skipping behavior on image SERPs. They may directly jump to results at some distance and ignore particular rows. A two-stage model can be used to depict this process in which users will judge the whole row first, and then decide to skip this row or view the details of results in this row.

Motivated by these observations, we propose corresponding user behavior assumptions to simulate users' interaction processes on SERPs. As shown in [3], evaluation metrics can be generalized as a function of gain and stopping probability, that is the sum over all ranks of the gain (e.g., relevance) accumulated by examining that far, times the probability that this is where the user stops examining the results. The basic idea of our proposed assumptions is to revise the stopping probability by incorporating grid-based position information. We show how we derive new evaluation metrics from these assumptions and how to adopt them to revise well-known list-based metrics.

We conduct extensive experiments to test the proposed assumptions. By using a large-scale commercial image search log, we show that incorporating grid-based features can help user behavior models to better predict the stopping position. We also use data from a field study, in which users' explicit satisfaction feedback and assessors' relevance judgments are available, to measure the performance of the grid-based evaluation metrics. We demonstrate that in image search, existing list-based metrics do not correlate well with user satisfaction while the proposed grid-based evaluation metrics can better reflect user satisfaction.

In summary, we make the following contributions:

- We thoroughly investigate how users allocate their attention on a grid-based interface in image search. We have three major findings of user behavior, i.e., "Middle bias," "Slower decay," and "Row skipping."
- Motivated by our findings on how attention is allocated, we propose corresponding user behavior assumptions to simulate users' search processes. We then derive new grid-based evaluation metrics based on these assumptions.
- We conduct extensive experiments to test the performance of our proposed grid-based evaluation metrics. Experimental results demonstrate that they better reflect user satisfaction and the assumptions behind them are closer to practical user behavior than the assumptions underlying competing models.

2 RELATED WORK

Related work comes in two areas: image search and evaluation metrics.

2.1 Image search

As result placement and interaction mechanisms in image search are different from general web search, user behavior in image search is different from user behavior in general web search. There exists a number of studies on user behavior analysis of image search engines. One line of prior research focuses on characterizing general user behavior based on search logs [2, 12, 28, 32]. Compared with general web search, important differences in user behavior (e.g., shorter queries, a tendency to be more exploratory, and to browse deeper) have been observed. Another line of research investigates more fine-grained user interactions with image SERPs. Xie et al. [34] observe a different browsing model on image SERPs and show a *middle position bias* of users' examination behavior. The observation "Middle bias" in this paper accords with their findings. Also,

interaction behavior such as cursor hovering has been shown to be a valuable additional signal for relevance [22, 35]. User behavior that is unique to image search has motivated various attempts at user behavior modeling that aim to improve the performance of image search engines [14, 35, 37].

Differences in user behavior also have an impact on evaluation. Previous work on evaluation of image search mainly adopts existing list-based evaluation metrics to measure the performance of models developed for image search by simply joining results together [11, 14]. Sanderson [24] introduces evaluation measures used in ImageCLEF, an evaluation forum for cross-language annotation and retrieval of images. However, these metrics still follow those in general web search. Zhang et al. [39] find that existing metrics in web search do not correlate well with user satisfaction in image search. The construction of evaluation metrics that do correlate well with user satisfaction in the context of grid-based interfaces for image search still remains an open question and deserves more attention.

2.2 Evaluation metrics

Evaluation sits at the center of IR research. In order to approximate the system's performance and users' search satisfaction, two components are needed. One is a search result collection labeled with query-dependent relevance levels and the other is a well-designed user model used to simulate the search process [25]. A number of effective evaluation metrics have been designed for general web search [7]. These metrics mainly follow the assumption that users scan ranked results from top to bottom before they stop [9]. One of these, RBP [21], assumes that users examine the $(i+1)$-th result after examining the i-th result with persistence p and will end their examination with probability $1-p$. Järvelin and Kekäläinen [15] propose a metric, DCG, that formalizes user gain from a result list as a discounting process. Besides considering the position impact, Expected Reciprocal Rank (ERR) [5] takes result relevance into consideration and defines the probability that a user is satisfied with a document to be related with relevance of the document. More sophisticated measures have been developed recently. Zhang et al. [38] try to model the search process based on upper limits for both benefit and cost, and propose a Bejeweled Player Model. Also, Wicaksono and Moffat [30] provide a detailed discussion of continuation probabilities (e.g., the persistence p in RBP) in user behavior models that underlie evaluation metrics.

In information retrieval, user satisfaction can be understood as the fulfillment of a specified desire or goal [16]. Satisfaction can be considered as the golden standard in search performance evaluation and is used to reflect users' actual feelings about the system [1, 13]. Correlation with actual user satisfaction is often taken to be the ultimate test for newly proposed evaluation metrics. Indeed, there exists a number of studies investigating different evaluation methods and the correlation between these methods and satisfaction [6, 19, 20, 26]. In this paper, we follow the same principle and also measure the performance of the proposed evaluation metrics by considering their correlation with actual user satisfaction.

What we add on top of the work discussed above is the following. List-based metrics have shown their effectiveness in estimating users' search satisfaction and measuring the performance of general web search engines – but they are list-based. However, in image search a grid-based result placement is adopted. We show that considering grid-based position information as part of the design of evaluation metrics can be beneficial. No previous research has investigated grid-based evaluation metrics for web image search.

3 IMAGE SEARCH USER BEHAVIOR PATTERNS

In order to gain a better understanding of user behavior in image search we examine the attention allocation mechanisms of search users in image search. The findings of this examination will help us to formulate an image search user model that will underlie our proposed grid-based evaluation metric.

We use two publicly available datasets, of image search and web search respectively, in this paper. The image search dataset has been created using data collected in a lab-based user study in image search scenarios [34]. A total of 40 participants have been recruited to complete 20 image search tasks in this study. A Tobbi eye-tracker with default settings has been used to record the examination behavior of participants; the participants' fixation points and fixation dwell time were recorded and certain image being examined was recorded by the built-in algorithms. The general web search dataset has been created using data collected in another user study conducted in general web search scenarios [18]. This dataset involves 32 participants who have been recruited to complete 30 web search tasks. Participants' fixation points on general web SERPs were recorded used the eye-tracker with the same settings and built-in algorithms as in the first dataset described above. Based on these two datasets, we cannot only investigate examination behavior in image search but compare image search with general web search.

We obtain three major findings of user examination behavior on image SERPs. They are "Middle bias," "Slower decay," and "Row skipping." The first one ("Middle bias") is mainly column-based and share the same observations with [34]. Starting from reviewing this finding, we introduce two new observations ("Slower decay" and "Row skipping") which are mainly row-based.

3.1 "Middle bias"

In image search, results are placed in a grid-like manner. Hence, users cannot only examine results vertically, as in web search, but also horizontally, within a row. It is important to investigate how users allocate their attention within a row. For the first dataset, similar to [34], we use the absolute position instead of the border of images to segment SERPs since the number of images in each row may be different (see the SERP example in Figure 1). Each SERP can be equally divided into 5 columns. We then draw a heat map with 10 rows and 5 columns of the distribution of examination durations (averaged over tasks and users); see Figure 2. Here, the examination duration of an image is defined as the dwell time during which a user gazes at the image. Gaze is the externally-observable indicator of human visual attention [17].

By examining the heat map in Figure 2 we re-confirm the observations from [34]: the middle positions in each row receive more attention than other positions, i.e., the leftmost or rightmost positions.

Based on these observations, we propose our first hypothesis:

Hypothesis 1 – Middle bias

Image search results in the middle position may attract more attention from search users than results in the left-most or rightmost position.



Figure 2: Distribution of examination duration (in seconds) in the first 10 rows in image search (0–9); rows are split into 5 columns (0–4).

Hypothesis 1 is not new: Xie et al. [34] already apply a linear mixed model to justify that the middle-position bias is significant statistically. That is, eye gaze behaviors are related to the location of an image within a row, and placing an image in the middle columns has a significant impact on fixation duration. However, they didn't adopt it to construct new image search evaluation metrics.

After Hypothesis 1, which concerns user examination behavior within a row, we introduce two other new observations and hypotheses that concern inter-row examination behavior patterns of image search users.

3.2 "Slower decay"

In image search, users can view results by scrolling down without having to click the "next page" button, which brings less cost to users and results in more exploratory search and deeper browsing depths [32]. We use the eye-tracking user study datasets to investigate how users examine SERPs in image search and general web search.

As shown in [35, 39], different within-row directions have little impact on user behavior modeling in image search. Define the *rank position* in a grid by following the top to bottom and left to right order. We calculate the examination duration for each cell in the grid (in the same way as was used in Section 3.1) and plot the distribution of the top 10 rank positions of image results in Figure 3. For the second dataset, we calculate the examination duration for each result and also plot the duration distribution in Figure 3 for comparison with image search.

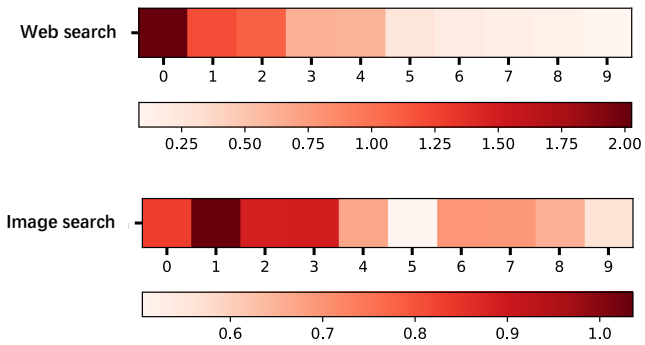


Figure 3: Distribution of examination duration (in seconds) in the first 10 rank positions of general web search and image search.

From Figure 3, the first observation is that users' examination duration does not decrease dramatically with the rank position in image search, especially within the same row (To note, there are five cells within a row.). Also, the difference of values between positions at different rows is smaller than the difference in web search. The second observation is that the change in examination duration in image search is not always monotonic, which is also different from web search. Position 7 (0.694s) receives a longer fixation than position 4 (0.671s) and position 5 (0.505s). In the case of web search, attention decreases in a monotonic way and at a higher speed than in the case of image search.

This leads to our second hypothesis:

Hypothesis 2 – Slower decay

Users' attention does not decrease monotonically and dramatically with the rank position. In the case of image search attention decays at a slower speed than in general web search.

To verify Hypothesis 2, we first take "two distributions in web search and Image search are similar" as the null hypothesis and then we use Pearson's chi-squared test, which is used to determine whether there is a significant difference between the expected distribution and the observed distribution, to determine whether the null hypothesis is true. The result shows that the p-value is less than 0.001. Hence, we can reject the null hypothesis and say that the difference in examination duration distribution between web search and image search in Figure 3 is significant. Also, we define "decay speed" as the result of dividing examination duration in position i by examination duration in position $i + 1$. We calculate the average decay speed based on the data shown in Figure 3. Results show that the average decay speed of image search (1.06) is much lower than of general web search (1.48).

3.3 "Row skipping"

We look deeper into examination sequences of search users using the eye-tracking data. We find that users will not examine every row one-by-one from top to bottom, which means they will skip rows and examine results at some distance. This "Row skipping"

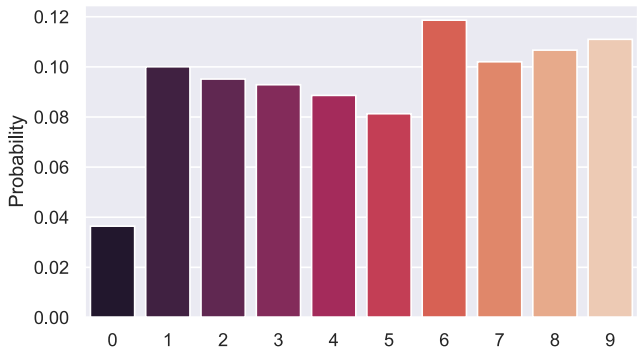


Figure 4: Probability of row-skipping behavior in different rows.

behavior can be formalized as: Right after a user examines results in the i-th row, she/he examines results in the j-th row where $j > i+1$. We define the probability of row-skipping behavior in a certain row (row i) as:

$$P(i) = \frac{S(i)}{S(i) + E(i)}. \tag{1}$$

Here, $E(i)$ is the number of cases where row i is being examined right after row $i-1$ has been examined. And $S(i)$ is the number of cases that users examine results at a row with a row number larger than i after examining row $i-1$. We define "search begin" as the row before row 0. That is, row 0 being skipped means the first examined row is not row 0. We show the probability of row-skipping behavior in the first 10 rows in Figure 4. There exists row-skipping behavior in image search. The highest probability is about 12% in the 6-th row in the first dataset. Also, the row-skipping probability in the 0-th row is much smaller than in later rows, which means users rarely skip the first row in image SERPs. Assuming that participants in a lab-based user study are more patient than users in real-life environments, the probability of row-skipping in real-life can be even higher.

Thus, we propose our third hypothesis:

Hypothesis 3 – Row skipping

Users may ignore particular rows and directly jump to results at some distance.

We take "the frequency of cases that row i being examined after row $i-1$ has been examined (i.e., E(i)) accords with the frequency of all cases that previous examined row is $i-1$ (i.e., S(i)+E(i))" as the null hypothesis. We also perform a chi-squared test and find that the p-value is less than 0.001. Therefore, we can reject the null hypothesis and say that row-skipping behavior does exist in user examination process.

To sum up, we have presented three hypotheses concerning user behavior in image search based on the observations made during eye-tracking user studies. Statistical tests have been conducted to verify the hypotheses and show the significance.

Although our first observation (i.e., "Middle bias") is not new, it has not been adopted in the design of image search evaluation metrics. we devote our attention to it as well as to two other, new observations, since considering both interaction processes in the horizontal direction ("Middle bias") and in the vertical direction ("Slower decay" and "Row skipping") as part of the construction of a grid-based evaluation metrics is beneficial in this two-dimensional environment.

4 GRID-BASED EVALUATION METRICS

In this paper, we construct grid-based evaluation metrics based on the user behavior hypotheses proposed in Section 3. We first introduce a uniform framework from which existing list-based evaluation metrics can be instantiated. We then propose three modeling assumptions motivated by the hypotheses in Section 3. Based on these assumptions, we derive new grid-based metrics by making revisions on the uniform structure.

4.1 Evaluation framework

Given a result set generated in response to a query, we can estimate users' satisfaction based on the relevance score of each query-result pair and a particular user model followed by users when they interact with this result set. Existing list-based evaluation metrics mainly follow an interaction process where users scan ranked results one-by-one from top to bottom before they stop. This interaction process can be regarded as a *cascade model* [9]. Following the cascade assumption, Moffat et al. [20] define a framework that captures a user's expected utility to generalize arbitrary list-based evaluation metrics (M) as:

$$M = \sum_{i=0}^{\infty} W_i R_i, \tag{2}$$

where R_i is the relevance score of the i-th result, and W_i is the metric-specific weight at rank position i. For example, for RBP with persistence probability p, $W_i = (1-p)p^{i-1}$ and for DCG, the metric-specific weight W_i would be $1/\log_2(i+2)$. To note, W_∞ is set to 0 for existing metrics.

Similar to work reported in [3, 38], we construct a uniform framework by considering user continuation and stopping probability. That is, users have a continuation probability C_i at position i to examine the $(i+1)$-th result and with probability S_i they stop at position i and leave this search or issue another query. Thus, S_i can be represented as:

$$S_i = \prod_{j=0}^{i-1} C_j(1 - C_i). \tag{3}$$

As shown in [3], the conditional probability of continuing past the i-th result, i.e., C_i, relates to the metric-specific weight, which can be computed as:

$$C_i = \frac{W_{i+1}}{W_i}. \tag{4}$$

We can transfer the framework mentioned in Eq. 2 to uniform framework depicting user stopping behavior and accumulated gain (relevance) as:

$$\overline{M} = \sum_{i=0}^{\infty}\left(S_i \sum_{j=0}^{i} R_j\right) = \sum_{i=0}^{\infty}\left(\prod_{j=0}^{i-1} C_j(1 - C_i) \sum_{j=0}^{i} R_j\right). \tag{5}$$

We refer to $\overline{M}$ as the *total user expected utility*. Next, we show that $\overline{M}$ and M are equivalent (i.e., $\overline{M} \sim M$):

$$\begin{aligned}\overline{M} &= \sum_{i=0}^{\infty}\left(\prod_{j=0}^{i-1} C_j(1-C_i)\sum_{j=0}^{i} R_j\right)\\ &= \frac{1}{W_1}\sum_{i=0}^{\infty}\left((W_i - W_{i+1})\sum_{j=0}^{i} R_j\right)\\ &= \frac{1}{W_1}\sum_{i=0}^{\infty}\left(R_i\sum_{j=i}^{\infty}(W_j - W_{j+1})\right)\\ &= \frac{1}{W_1}\sum_{i=0}^{\infty} R_i W_i\\ &\sim M.\end{aligned} \tag{6}$$

The last equivalence holds because W_1 is a constant given a certain evaluation metric. The framework detailed in Eq. 5 can take the user interaction process into consideration more naturally than the framework depicted in Eq. 2 which mainly models the metric-specific weight and obtained gain for each rank. We therefore make revisions on this framework by incorporating grid-based assumptions. For convenience, we use a triple $(i, r(i), c(i))$ to represent the index of an image result. As we discuss in Section 3.2, we predefine the examination order of search users in image search to be from top to bottom and from left to right. Based on this order, we can obtain the rank position i of a certain image which is in the $r(i)$-th row and $c(i)$-th column.

We are now in a position to introduce the grid-based modeling assumptions which are among the contributions of our work. The order in which we propose our assumptions is the same as the order used to present observations of user behavior patterns in Section 3, i.e., "Middle bias" followed by "Slower decay" and "Row skipping."

4.2 Middle bias assumption

The first assumption, named "Middle bias," focuses on the interaction within a single row, i.e., it is column-based. As mentioned in Section 3.1, users have a higher probability to examine results in the middle position. In this paper, we simulate this bias by considering users' continuation examination, in which we increase the stopping probability in the middle position and lower it in the leftmost or the rightmost positions. We assume that users will have a higher probability to finally stop at the middle position within a row. Hence, we can use a column-based function $f(c)$ to modify the stopping probability S_i. For the image at rank position i with the column number $c(i)$, we design the function $f(c(i))$ as follows:

$$f(c(i)) = e^{g(c(i))}, \tag{7}$$

where $g(c(i))$ is a normal distribution with mean μ and standard deviation σ as:

$$g(c(i)) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(c(i)-MP-\mu)^2}{2\sigma^2}}, \tag{8}$$

where MP denotes the column index of the middle position in row $r(i)$. We leave explorations of other functions (such as, e.g., a quadratic function) as future work. In a normal distribution, the mean is the central tendency of the distribution; it defines the location of the peak for normal distributions. And the standard deviation is a measure of variability; it defines the width of the normal distribution. Since we simulate users' middle bias in this assumption, we set μ to be a constant number 0 to further simplify the parameter estimation process, which means the "location" of the normal distribution will be right in the middle of the column. Thus, σ is then the only parameter needed to estimate in Eq. 8. Hence, based on the middle bias assumption, the total user expected utility ($\overline{M}$) can be represented as:

$$\overline{M_{MB}} = \sum_{i=0}^{\infty}\left(\prod_{j=0}^{i-1} f(c(i))C_j(1-C_i)\sum_{j=0}^{i} R_j\right). \tag{9}$$

4.3 Slower decay assumption

As mentioned in Section 3.2, the "Slower decay" observation shows that users are more patient in image search than in web search. Their attention decreases more slowly, especially on results within a row. Thus, simply adopting existing evaluation metrics, developed for web search, to image search scenarios is not promising. In this paper, we utilize the row information of image results. We assume that users' stopping probability will increase along with the row. Hence, we can revise the stopping probability S_i in the proposed evaluation framework by multiplying S_i with a row-based function $I(r)$. Considering a result at rank position i with row number $r(i)$ and column number $c(i)$, the revised probability of stopping at this result can be computed as:

$$\overline{S_{i,r(i),c(i)}} = I(r(i)) \cdot S_{i,r(i),c(i)}, \tag{10}$$

where $S_{i,r(i),c(i)}$ is the original stopping probability of a certain list-based metric; $I(r)$ is a monotonically increasing function. In this paper, we define $I(r)$ as an exponential function with a base β larger than 1. Then, we can rewrite Eq. 5 as:

$$\overline{M_{SD}} = \sum_{i=0}^{\infty}\left(\prod_{j=0}^{i-1} \beta^{r(i)} C_j(1-C_i)\sum_{j=0}^{i} R_j\right). \tag{11}$$

Adding the parameter β can slow down the speed of decreasing the stopping probability along with rows, since users might still have a relatively high probability of examining results at a lower rank (see Figure 3). Also, the stopping probability of results within a row will multiply the same value according to Eq. 10, which attempts to control the variance between the stopping probability of results in the same row. When $\beta = 1$, Eq. 10 models the stopping probability of existing list-based metrics. We show how different values of β affect the estimated stopping probability distribution in Section 6.

4.4 Row skipping assumption

The third assumption is motivated by the "Row skipping" observation which suggests that users may skip particular rows and jump to results at some distance. In this paper, we model this process by considering a two-stage browsing process. In the first stage, users briefly browse the whole row; we can join image results within a row together to an imaginary "united image". By viewing this "united image," users will make a decision for the second stage where they either skip this row or examine results in this row in detail. We arrive at this two-stage browsing process motivated by a

neuroimaging study [36], which gives important hints about the multistage mechanisms of visual perceptual learning in the brain.

We are now in a position to describe our row skipping evaluation metric ($\overline{M_{RS}}$). We use a parameter γ to depict the probability with which users skip the next row after examining the current row; γ is also a trainable parameter with a value between 0 and 1. Then, the stopping probability of users at rank position i can be computed as follows:

$$S_{i(RS)} = \underbrace{\prod_{k=0}^{r(i)-1}\left((1-\gamma)\prod_{j=S(k)}^{S(k)+N(k)-1}C_j+\gamma\right)}_{\text{two-stage browsing assumption}}\cdot(1-\gamma)\prod_{j=S(r(i))}^{i-1}C_j(1-C_i), \tag{12}$$

where $N(k)$ is the number of images in the k-th row and $S(k)$ is the total number of images before the k-th row. The first part of Eq. 12, before the multiplication sign, depicts the two-stage browsing assumption. We simply assume that with a probability $(1-\gamma)$, users will examine all the image results within this row. Since users stop at row $r(i)$, they will not skip row $r(i)$. Thus, there is the probability $(1-\gamma)$ in the second part after the multiplication sign in Eq. 12.

The row skipping assumption also has an impact on the accumulated gain (i.e., $\sum_{j=0}^{i} R_j$). Since users have a probability γ to skip a certain row, the gain received from this row should be discounted by multiplying $(1-\gamma)$. Hence, the total user expected utility ($\overline{M}$) based on the row skipping assumption can be computed as:

$$\overline{M_{RS}} = \sum_{i=0}^{\infty} S_{i(RS)} \sum_{j=0}^{i}(1-\gamma)R_j. \tag{13}$$

In this section, we have proposed three grid-based assumptions. According to these assumptions, we revise the formula expressing the continuation probability, stopping probability and also the accumulated gain in the uniform evaluation framework (see Eq. 5). To sum up, we modify the stopping probability at different columns, increase the value in the middle position, by considering a normal distribution according to the "Middle bias" assumption. We modify the stopping probability at each rank by increasing the value of the probability along with the rank according to the "Slower decay" assumption. And based on the "Row skipping" assumption, we consider a two-stage browsing process in which users have a skipping probability to ignore a certain row. Thus, the accumulated gain of a certain row is also modified by multiplying the probability that users browse this row.

5 EXPERIMENTAL SETUP

We evaluate the proposed grid-based evaluation metrics using search logs from a commercial image search engine and data from a field study, in which query-level satisfaction feedback and assessors' relevance judgments for query-image pairs are available. Since there is a user behavior model, which depicts the stopping behavior of search users, behind each proposed assumption, we first perform a sanity check, that is, an experiment to test if incorporating grid-based features can help the underlying user behavior model to better predict the stopping position (in terms of mean log-likelihood). As mentioned in Section 2, user satisfaction can be considered as the golden standard in search performance evaluation. In the same way as in [19, 38], we compare our proposed grid-based evaluation metrics against existing list-based metrics in terms of their correlation with user satisfaction to show the effectiveness of proposed grid-based assumptions.

In this section we first introduce the datasets and then describe the design of the two main experiments in this paper.

Table 1: Statistics of the datasets used in our experiments ("#" refers to "number of").

Dataset	#Tasks	#Participants	#Queries	#Sessions
Search log	–	–	82,629	100,000
Field study	555	50	1,212	1,212

5.1 Datasets

Two image search datasets are used to conduct the experiments. Descriptive statistics of these two datasets can be found in Table 1.

The first dataset ("Search log") is randomly sampled from a search log in October 2017 from the Sogou image search engine, which is popular in China. In this dataset, the grid-based information (i.e., row and column number of image results) and user interaction behavior (i.e., click and cursor hovering) are available. We keep query sessions that have at least one click to make sure we can estimate the user's stopping position, since the last clicked rank can be used to approximate the users' actual stopping rank as shown in [3]. The number of search sessions used in this paper is 100K in total, with 80,000+ distinct queries. We split all query sessions into training and test sets at a ratio of 8:2.

The second dataset ("Field study") consists of data collected from a one-month field study, which is publicly available (see [31]). In this field study, participants are asked to provide explicit satisfaction feedback for their search experience. To note, they can decide which query sessions they want to give the explicit feedback on without having to annotate all search sessions and they are also asked to provide a description about the task they conduct when issuing a specific query. Query-level satisfaction scores on a 5 point scale are gathered. Besides user behavior data recorded using a browser extension and explicit feedback from participants, relevance scores of query-image pairs are annotated by assessors on a crowdsourcing platform. Each query-image pair has at least five relevance annotation scores in the range of 0 to 100. We use the average of these annotation scores in our experiment as the label of a certain query-image pair. Also, assessors are recruited to assign a user intent tag to each task (i.e., the "Locate, Learn, Entertain" taxonomy proposed by Xie et al. [33]). Since image search users usually have deeper browsing depths, we test the performance of evaluation metrics at depths of 5, 10 as in [39] and 15 as well. Thus, we keep query sessions in which the number of the last browsing row is not less than 15, which leads to 1,212 query sessions in total in our dataset.

5.2 Experiment 1: Behavior prediction

Experiment 1 is aimed at testing whether the proposed grid-based user behavior assumptions (considering the continuation and stopping behavior in a grid-based interface) are closer to real-life user

Table 2: Grid-search values of hyper-parameters that need to be estimated and the grid-based assumptions these parameters belong to for Experiment 1.

Parameter	Assumption	Grid-search values
p	–	$\{0.1, 0.2, \ldots, 0.9\}$
σ	Middle bias	$\{1, 2, \ldots, 10\}$
β	Slower decay	$\{1.1, 1.2, \ldots, 2.0\}$
γ	Row skipping	$\{0.1, 0.2, \ldots, 0.9\}$

behavior than list-based assumptions. As mentioned in Section 4, the proposed grid-based assumptions revise users' stopping probability at different rank positions by incorporating row and column information. To validate the user behavior assumptions underlying the proposed evaluation metrics, we test the performance of these assumptions on predicting users' actual stopping positions.

We use RBP as our baseline model that naturally takes users' continuation and stopping into consideration. In RBP, a persistence probability p is used to depict users' continuation probability at each rank. Based on formulas introduced in Section 4, we can calculate the stopping probability at different ranks, estimated by RBP as well as by grid-based RBPs with different proposed assumptions (i.e., "MD": Middle bias; "SD": Slower decay; "RS": Row skipping). For example, the stopping probability at rank position $(i, r(i), c(i))$ estimated by grid-based RBP with the "Slower decay" assumption according to Eq. 11 can be computed as:

$$S_i(RBP - SD) = \prod_{j=0}^{i-1} \beta^{r(i)} p(1-p). \tag{14}$$

We regard the last click position to be users' stopping position on SERPs in the same way as in [3]. And we use log-likelihood to show how well the stopping probability distributions estimated by different models approximate the actual user stopping behavior.

We use a grid-search algorithm to estimate the best parameter(s) for each model to minimize the mean log-likelihood of the training data (80%) in our first dataset. We then test the performance of these models with the pre-trained hyper-parameter(s) in the test data (20%). We show the details of the bounds and discretization of the different parameters needed to be estimated using grid-search in Table 2.

5.3 Experiment 2: Correlation with user satisfaction

In Experiment 2, we measure the performance of our grid-based assumptions by testing the correlation between grid-based evaluation metrics, derived from our assumptions, and user satisfaction. We first conduct experiments on RBP-based metrics. We show Pearson's correlation results of RBP with different assumptions (the original list-based and the proposed grid-based assumptions). We also construct a t-statistic to test the significance of the difference between two dependent correlation coefficients [8]. The p-value level is reported if a significant difference is observed. We then look deeper into the effect of different settings of our proposed assumptions (e.g., different starting rows to consider row-skipping assumption, different number of rows of results being modeled in the evaluation metrics). After that, we report results of the grid-based evaluation metrics, under the best settings, based on other list-based prototype metrics (i.e., ERR and DCG). Comparisons are also made between different prototype metrics.

6 RESULTS

We first report the results of Experiment 1, behavior prediction of user behavior models that are based on different grid-based assumptions. Then, in Experiment 2 we show the performance of grid-based evaluation metrics in terms of their correlation with user satisfaction. We compare the parameter selection in different tasks and discuss the optimal settings to perform our proposed grid-based evaluation metrics. Additional comparisons are made between different grid-based evaluation metrics based on different prototype list-based evaluation metrics.

6.1 Evaluation of behavior prediction

Table 3 shows the minimized mean log-likelihood of each user behavior model as well as the value of the best parameters and improvements over the baseline model (RBP). Here, the improvement of the log-likelihood of model A over model B is computed as $\frac{LL(B)-LL(A)}{LL(B)}$. We also perform pairwise t-tests to determine the significance of the observed difference between grid-based models and the baseline model.

Table 3: Outcomes of Experiment 1. Minimized mean Log-likelihood of user behavior models. **: significantly better than the RBP model with p-value < 0.01.

Model	Parameter(s)	Log-likelihood	Improvement
RBP	$p(0.8)$	-0.542	–
RBP-MB	$p(0.8)$, $\sigma(2)$	-0.513^{**}	5.4%
RBP-SD	$p(0.8)$, $\beta(1.9)$	-0.473^{**}	12.7%
RBP-RS	$p(0.5)$, $\gamma(0.2)$	-0.469^{**}	13.5%

Compared against the baseline model (RBP), our grid-based models with the proposed assumptions achieve better performance on behavior prediction, i.e., users' stopping behavior, in terms of mean log-likelihood. Also, all observed differences are significant. The best grid-based model RBP-RS obtains a 13.5% (significant) improvement over the list-based model RBP. Thus, incorporating grid-based information into the construction of a user behavior model is beneficial and results show that search user behavior in a grid-based environment differs from that in a list-based environment.

Compared to the "Slower decay" and "Row skipping" assumptions, both of which help RBP to better predict user stopping behavior, RBP with the "Middle bias" assumption has a smaller improvement over the baseline model on behavior prediction. The reason can be two-fold:

(1) The method used to depict middle position bias of search users may not be optimal. The practical distribution of users' stopping probability within a row may follow more complex distributions. We leave an investigation on methods to more accurately model "Middle bias" behavior as future work.
(2) Users' stopping behavior correlates more with row information than with column information. Thus, the row-based assumptions ("Slower decay" and "Row skipping") achieve better results

than the column-based assumptions ("Middle bias") on behavior prediction.

We also show the value of the best parameters in Table 3. We can see the performance of the baseline model with a fixed continuation probability is not promising, which indicates that in image search users' continuation probability may be affected by other factors like the position information of the current examined result.

When considering the "Middle bias" assumption, the value of the best parameter σ is 2. For a normal distribution, a small standard deviation (σ) produces a distribution that is more tight. Thus, a difference in stopping probability between middle position and other positions is observable.

By incorporating the additional parameter β (when considering the "Slower decay" assumption), we are able to consider the possibility that users' stopping probability will increase along with the row. In this setting, the probability of the results at lower ranks being examined will be higher than in the list-based setting, which might indicate that in image search users have deeper browsing depth (confirming [32]); the stopping probability will decrease slowly.

In RBP with the "Row skipping" assumption, we observe that the probability to ignore certain rows is 0.2. This observation accords with the results shown in Figure 4 while the row-skipping probability in the search log is slightly higher than in the user study data. This may be caused by the fact that participants in a lab-based user study may be more patient due to the phenomenon that their behavior will be recorded. Thus, the probability of row-skipping of real-life users can be higher.

In summary, Experiment 1 has shown that the grid-based assumptions proposed in this paper are closer to natural user behavior than list-based assumptions. User behavior models underlying the grid-based assumptions achieve better performance in predicting real-life user behavior, i.e., users' stopping behavior. The value of the estimated parameters of grid-based assumptions further confirms the observations introduced in Section 3.

6.2 Evaluation of user satisfaction correlation

As explained in Section 5.3, we first consider RBP-based evaluation metrics at top 10 rows, in the same way as in [39]. Table 4 shows the coefficients of Pearson's correlation between RBP-based metrics and user satisfaction. As shown in Figure 4, the row-skipping probability in the 0-th row is much smaller; we also compare the different settings of where we start to perform "Row skipping" in this table.

We can observe from Table 4 that with the help of the proposed grid-based assumptions, RBP-based evaluation metrics can achieve better correlation with user satisfaction than the original RBP that follows the list-based assumption.

Since the optimization target is different from the target in Section 6.1, we fit the best parameters of different evaluation metrics to gain the best correlation with users satisfaction in this experiment. We can observe from Table 3 and Table 4 that the best parameters in these two tasks are slightly different. The reason can be two-fold:

(1) We consider a fixed number of rows in this experiment to calculate the correlation, since a predefined scale of results being measured is required for offline evaluation metrics [15, 21]. However, in Experiment 1, we compute the log-likelihood based on the rank users stop at. Real users may have different depths of browsing due to their search dwell time.
(2) In the field study, participants can freely decide the feedback of which query session to be recorded by the browser extension. Thus, the search intent distribution may be slightly different between the field study data and the search log. This difference has previously been observed by [33]. Different search intents have an impact on user behavior and satisfaction [31].

Table 4: Outcomes of Experiment 2. Pearson's Correlation between RBP-based evaluation metrics (@top 10 rows) and user satisfaction in the field study dataset. "S@n": "Start performing Row skipping assumption at row n". ↑ (↓): the grid-based evaluation metric achieves better (worse) performance than the baseline model RBP. All correlations are significant at the $p < 0.001$ level.

Metric	Parameter(s)	PC coefficient	Performance
RBP	$p(0.7)$	0.333	–
RBP-MB	$p(0.7)$ $\sigma(1)$	0.341	↑
RBP-SD	$p(0.7)$ $\beta(1.2)$	**0.342**	↑
RBP-RS(S@0)	$p(0.8)$ $\gamma(0.1)$	0.322	↓
RBP-RS(S@1)	$p(0.7)$ $\gamma(0.2)$	0.336	↑
RBP-RS(S@2)	$p(0.7)$ $\gamma(0.2)$	0.334	↑
RBP-RS(S@3)	$p(0.7)$ $\gamma(0.2)$	0.333	–

Table 5: Outcomes of Experiment 2. Pearson's Correlation between RBP-based evaluation metrics at different number of rows (top 5, 10 and 15 respectively) and user satisfaction in field study dataset. "S@n" refers to "Start performing row skipping at row n". All correlations are significant at the $p < 0.001$ level.

Metric	Top 5 rows	Top 10 rows	Top 15 rows
RBP	0.331	0.333	0.333
RBP-MD	0.340	0.341	0.341
RBP-SD	0.342	0.342	0.342
RBP-RS(S@1)	0.331	0.336	0.337

We also investigate how the choice of different first rows to which to apply the "Row skipping" assumption affects the performance of the grid-based evaluation metrics. Results are shown in Table 4; they indicate that applying the "Row skipping" assumption at the very beginning is not promising. When we apply the "Row skipping" assumption at the second row (i.e., RBP-RS(S@1)), we observe a better result, with a higher correlation with user satisfaction than list-based RBP. We also show the results of other RBP-RSs with different starting rows (with a row number larger than 1) to apply the "Row-skipping" assumption in Table 4. We find that although these metrics are better than the list-based metric, the improvement of them over the baseline decrease along with the row number of the starting rows. RBP-RS(S@3) has no observable improvement. Thus, it is optimal to consider "Row skipping" starting from the second row. This finding accords with the results plotted in Figure 4, which shows that users rarely skip the first row on an image SERPs.

Since we need to define the number of rows of results being considered in the evaluation metrics before using a certain metric, we also discuss the optimal setting of the row scale. For each query

session, we test the performance of different RBP-based models at the top 5, 10 and 15 rows respectively. Results are shown in Table 5. We have two findings from this table:

(1) When only a small number of rows is considered, grid-based evaluation metrics with the "Row skipping" assumption, which mainly takes row-based information into consideration, cannot achieve improvements over the list-based competitor. However, the column-based evaluation metric (i.e, RBP-MD) is still better than the baseline model: RBP-MD mainly considers user behavior within a row: changes in the number of rows have less effect on it. For the "Row skipping" assumption, the reason that we only observe small differences may be that the stopping probability at a lower rank is too small to be affected. Thus, the improvement over RBP obtained by considering "Row skipping" is achieved mainly from the top rows.
(2) The more rows are being considered in the evaluation metrics, the better the correlation with user satisfaction that can be achieved, for all evaluation metrics (see, e.g., how "Top 10 rows" compares with "Top 5 rows"). However, the difference between "Top 10 rows" and "Top 15 rows" is small which indicates there exists an upper bound on the performance. Hence, considering the annotation expense, we regard "Top 10 rows" as the best setting of the row scale.

Armed with the best settings ("S@1", "Top 10 rows") observed from the experiments conducted on the RBP-based evaluation metrics, we further test the effectiveness of our proposed grid-based assumptions on other list-based prototype metrics. We perform experiments on two other list-based prototype metrics, i.e., DCG and ERR. Recall that DCG is also a position-based model, like RBP. The difference is that the continuation probability of the result at rank i in DCG is rank-dependent; it can be computed as:

$$C_i(DCG) = \frac{W_{i+1}}{W_i} = \frac{\log_2(i+2)}{\log_2(i+3)}. \tag{15}$$

In addition, we consider ERR. Unlike DCG and RBP, the stopping criterion of ERR is affected by the gain (G) of the currently examined result. Following [5], the probability that a user stops at rank i can be represented as:

$$S_i(ERR) = \prod_{j=0}^{i-1}(1-G_j)G_i, \tag{16}$$

where G_i is the gain that correlates with the relevance score of the current result at rank i, which has the following form:

$$G_i(ERR) = \frac{2^r - 1}{2^{r_{\max}}}. \tag{17}$$

where r is the relevance score of the i-th result. ERR and DCG have been used in previous evaluation tasks on image search [35, 39].

We are now in a position to test the performance of our grid-based assumptions on these two evaluation metrics. The results are presented in Table 6. The proposed grid-based assumptions can help ERR and DCG to achieve better correlation with user satisfaction, while an exception is observed (ERR with the MB assumption). All ERR-based evaluation metrics obtain a poor correlation with user satisfaction, confirming a similar result by Zhang et al. [39]. The reason may be that ERR focuses more on the user gain rather than the examined position. As shown in [11, 35], users' judgments about image results depend largely on image attractiveness. Only considering the effect of relevance on user stopping may not be promising. Furthermore, since position information is not explicitly modeled in the stopping probability in ERR, a grid-based version of ERR cannot achieve promising results.

Table 6: Outcomes of Experiment 2. Pearson's Correlation between evaluation metrics (DCG and ERR @top 10 rows) and user satisfaction in field study dataset. "(u@0.9)" refers to "Upper bound of continuation probability is 0.9". All correlations are significant at the $p < 0.001$ level. ‡ (†): the difference is significant comparing to the corresponding list-based metrics at the $p < 0.01$ (0.05) level.

Metric	List-based	MB	SD	RS(S@1)
ERR	0.169	0.152	0.169	0.180
DCG	0.225	0.260‡	0.224	0.295‡
DCG (u@0.9)	0.291	0.308‡	0.305†	0.303

For the DCG-based evaluation metrics, we can observe the expected results that most grid-based DCG metrics perform better than the list-based DCG, demonstrating the effectiveness of our grid-based assumptions. We also observe a similar performance of DCG and DCG-SD. This may be explained by the fact that the original continuation probability, which is shown in Eq. 15, approaches 1 quickly along with the rank which results in a small stopping probability approaching 0. Thus, the parameter β of "Slower decay" assumption has limited effect on the stopping probability. We also consider an upper bound of the continuation probability of DCG-based evaluation metrics. The results are also shown in Table 6 (last row). All grid-based DCG metrics obtain better correlation with user satisfaction than the list-based DCG. Also, setting an upper bound on the continuation probability improves the performance of all DCG-based metrics, which confirms the observation that users' attention decays at a slower speed. Simply adopting assumptions of list-based DCG is not promising in image search scenarios. Importantly, the best parameters of grid-based DCG are almost the same as for RBP shown in Table 4, i.e., $\sigma(1)$, $\beta(1.1)$ and $\gamma(0.2)$, where the different setting for β may be caused by the different continuation probability settings between RBP and DCG. The results shown in Table 6 indicate that the proposed grid-based assumptions help increase the correlation of position-based models of user satisfaction (e.g., RBP and DCG).

In summary, Experiment 2 has shown that the proposed grid-based assumptions can help existing list-based evaluation metrics, especially position-based evaluation metrics (e.g., RBP and DCG), to better reflect user satisfaction. We find that: (1) performing the "Row-skipping" assumption beginning at the second row rather than the first row can help RBP-RS to achieve better performance; and (2) a result grid limited to the top 10 rows in RBP-based evaluation metrics is optimal considering the trade-off between metric performance and annotation cost.

7 CONCLUSION AND FUTURE WORK

In this paper, we have conducted a comprehensive user behavior analysis using data from a lab-based user study so as to understand the attention allocation mechanisms of search users in image search.

We obtain three major findings through our analysis: (1) User attention follows a middle position bias within a row ("Middle bias"). (2) User attention in the case of image search decays more slowly than in general web search ("Slower decay"). (3) Users may skip particular rows and jump to results at some distance ("Row skipping").

We have proposed three grid-based assumptions. Our experimental results show that user behavior models underlying these grid-based assumptions are closer to real-life user behavior. Existing evaluation metrics (e.g., RBP and DCG) can achieve better performance in terms of correlation with user satisfaction by taking grid-based assumptions into consideration.

Our work is the first attempt to construct grid-based evaluation metrics for Web image search. The research outputs of this paper can guide the optimization of image search engines (e.g., in result ranking and UI design) and are also meaningful to inform user behavior modeling in grid-based environments (not only image search but also video search and e-commerce).

Limitations of the proposed grid-based assumptions which may guide future work: (1) The proposed grid-based assumptions mainly consider the effect of the position. It may be beneficial to also take appearance bias (the effect of image attractiveness) into consideration. (2) The way to model grid-based user behavior may not be optimal, e.g., using the normal distribution to simulate the "Middle bias." Methods to encode grid-based user behavior and combine different user behavior assumptions need further investigation [4]. (3) We test the performance of grid-based assumptions on a small group of evaluation metrics only. Experiments conducted on further evaluation metrics are called for. (4) As the effectiveness of evaluation metrics may vary with tasks [29], we will try to investigate the performance of proposed grid-based evaluation metrics across search tasks and intents.

Code

To facilitate reproducibility of our results, we share the code used to run our experiments at https://github.com/THUxiexiaohui/grid-based-evaluation-metrics.

Acknowledgements

This work is supported by Natural Science Foundation of China (Grant No. 61622208, 61732008, 61532011), The National Key Research and Development Program of China (2018YFC0831700), Ahold Delhaize, the Association of Universities in the Netherlands (VSNU), and the Innovation Center for Artificial Intelligence (ICAI). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Azzah Al-Maskari, Mark Sanderson, and Paul Clough. 2007. The relationship between IR effectiveness measures and user satisfaction. In *30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 773–774.

[2] Paul André, Edward Cutrell, Desney S Tan, and Greg Smith. 2009. Designing novel image search interfaces by understanding unique characteristics and usage. In *IFIP Conference on Human-Computer Interaction*. Springer, 340–353.

[3] Leif Azzopardi, Paul Thomas, and Nick Craswell. 2018. Measuring the utility of search engine result pages: an information foraging based measure. In *41st International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 605–614.

[4] Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. 2016. A neural click model for web search. In *WWW 2016: 25th International World Wide Web Conference*. ACM, 531–541.

[5] Olivier Chapelle, Donald Metzler, Ya Zhang, and Pierre Grinspan. 2009. Expected reciprocal rank for graded relevance. In *18th ACM conference on Information and knowledge management*. ACM, 621–630.

[6] Aleksandr Chuklin and Maarten de Rijke. 2016. Incorporating clicks, attention and satisfaction into a search engine result page evaluation model. In *25th ACM Conference on Information and Knowledge Management*. ACM, 175–184.

[7] Aleksandr Chuklin, Pavel Serdyukov, and Maarten de Rijke. 2013. Click model-based information retrieval metrics. In *SIGIR '13: 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 493–502.

[8] Patricia Cohen, Stephen G West, and Leona S Aiken. 2014. *Applied multiple regression/correlation analysis for the behavioral sciences*. Psychology Press.

[9] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In *2008 International Conference on Web Search and Data Mining*. ACM, 87–94.

[10] Marco Ferrante, Nicola Ferro, and Maria Maistro. 2014. Injecting user models and time into precision via Markov chains. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 597–606.

[11] Bo Geng, Linjun Yang, Chao Xu, Xian-Sheng Hua, and Shipeng Li. 2011. The role of attractiveness in web image search. In *19th ACM International Conference on Multimedia*. ACM, 63–72.

[12] Abby Goodrum and Amanda Spink. 1999. Visual information seeking: A study of image queries on the World Wide Web. In *ASIST Annual Meeting*, Vol. 36. 665–74.

[13] Scott B Huffman and Michael Hochster. 2007. How well does result relevance predict session satisfaction?. In *30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 567–574.

[14] Vidit Jain and Manik Varma. 2011. Learning to re-rank: query-dependent image re-ranking using click data. In *20th International Conference on World Wide Web*. ACM, 277–286.

[15] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[16] Diane Kelly. 2009. Methods for evaluating interactive information retrieval systems with users. *Foundations and Trends in Information Retrieval* 3, 1–2 (2009), 1–224.

[17] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. 2016. Eye tracking for everyone. In *IEEE Conference on Computer Vision and Pattern Recognition*. 2176–2184.

[18] Yiqun Liu, Zeyang Liu, Ke Zhou, Meng Wang, Huanbo Luan, Chao Wang, Min Zhang, and Shaoping Ma. 2016. Predicting search user examination with visual saliency. In *39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 619–628.

[19] Cheng Luo, Yiqun Liu, Tetsuya Sakai, Fan Zhang, Min Zhang, and Shaoping Ma. 2017. Evaluating mobile search with height-biased gain. In *40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 435–444.

[20] Alistair Moffat, Paul Thomas, and Falk Scholer. 2013. Users versus models: What observation tells us about effectiveness metrics. In *22nd ACM international conference on Information & Knowledge Management*. ACM, 659–668.

[21] Alistair Moffat and Justin Zobel. 2008. Rank-biased precision for measurement of retrieval effectiveness. *ACM Transactions on Information Systems (TOIS)* 27, 1 (2008), Article 2.

[22] Neil O'Hare, Paloma De Juan, Rossano Schifanella, Yunlong He, Dawei Yin, and Yi Chang. 2016. Leveraging user interaction signals for web image search. In *39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 559–568.

[23] Harrie Oosterhuis and Maarten de Rijke. 2018. Ranking for relevance and display preferences in complex presentation layouts. In *41st international ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 845–854.

[24] Mark Sanderson. 2010. Performance measures used in image information retrieval. In *ImageCLEF*. Springer, 81–94.

[25] Mark Sanderson. 2010. Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval* 4, 4 (2010), 247–375.

[26] Mark Sanderson, Monica Lestari Paramita, Paul Clough, and Evangelos Kanoulas. 2010. Do user preferences and evaluation measures line up?. In *33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 555–562.

[27] Yang Song, Hao Ma, Hongning Wang, and Kuansan Wang. 2013. Exploring and exploiting user search behavior on mobile and tablet devices to improve search relevance. In *22nd International Conference on World Wide Web*. ACM, 1201–1212.

[28] Dian Tjondronegoro, Amanda Spink, and Bernard J Jansen. 2009. A study and comparison of multimedia Web searching: 1997–2006. *Journal of the American Society for Information Science and Technology* 60, 9 (2009), 1756–1768.

[29] Andrew Turpin and Falk Scholer. 2006. User performance versus precision measures for simple search tasks. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 11–18.

[30] Alfan Farizki Wicaksono and Alistair Moffat. 2018. Empirical evidence for search effectiveness models. In *27th ACM International Conference on Information and Knowledge Management*. ACM, 1571–1574.

[31] Zhijing Wu, Yiqun Liu, Qianfan Zhang, Kailu Wu, Min Zhang, and Shaoping Ma. 2019. The influence of image search intents on user behavior and satisfaction. In *12th ACM International Conference on Web Search and Data Mining*. ACM.

[32] Zhijing Wu, Xiaohui Xie, Yiqun Liu, Min Zhang, and Shaoping Ma. 2017. A study of user image search behavior based on log analysis. In *China Conference on Information Retrieval*. Springer, 69–80.

[33] Xiaohui Xie, Yiqun Liu, Maarten de Rijke, Jiyin He, Min Zhang, and Shaoping Ma. 2018. Why people search for images using web search engines. In *11th ACM International Conference on Web Search and Data Mining*. ACM, 655–663.

[34] Xiaohui Xie, Yiqun Liu, Xiaochuan Wang, Meng Wang, Zhijing Wu, Yingying Wu, Min Zhang, and Shaoping Ma. 2017. Investigating examination behavior of image search users. In *40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 275–284.

[35] Xiaohui Xie, Jiaxin Mao, Maarten de Rijke, Ruizhe Zhang, Min Zhang, and Shaoping Ma. 2018. Constructing an interaction behavior model for web image search. In *41st International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 425–434.

[36] Yuko Yotsumoto, Takeo Watanabe, and Yuka Sasaki. 2008. Different dynamics of performance and brain activation in the time course of perceptual learning. *Neuron* 57, 6 (2008), 827–833.

[37] Jun Yu, Dacheng Tao, Meng Wang, and Yong Rui. 2015. Learning to rank using user clicks and visual features for image retrieval. *IEEE Transactions on Cybernetics* 45, 4 (2015), 767–779.

[38] Fan Zhang, Yiqun Liu, Xin Li, Min Zhang, Yinghui Xu, and Shaoping Ma. 2017. Evaluating web search with a bejeweled player model. In *40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 425–434.

[39] Fan Zhang, Ke Zhou, Yunqiu Shao, Cheng Luo, Min Zhang, and Shaoping Ma. 2018. How well do offline and online evaluation metrics measure user satisfaction in web image search?. In *41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 615–624.

Sarcasm Detection with Self-matching Networks and Low-rank Bilinear Pooling

Tao Xiong*
Ant Financial Security and Risk Management
Hangzhou, China
weilue.xt@alibaba-inc.com

Peiran Zhang†
Alibaba Group
Hangzhou, China
peiran.zpr@alibaba-inc.com

Hongbo Zhu
Alibaba Group
Hangzhou, China
xiaofeng.zhb@alibaba-inc.com

Yihui Yang*
Afterpay
Melbourne, Australia
yihui.y@outlook.com

ABSTRACT

Sarcasm is sophisticated linguistic expression and is commonly observed on social media and e-commerce platforms. Failure to detect sarcastic expressions in natural language processing tasks, such as opinion mining and sentiment analysis, leads to poor model performance. Traditional approaches rely heavily on discrete handcrafted features and will incur enormous human costs. It was not until recent that scholars began to employ neural networks to address these limitations and have achieved new state-of-the-art performance. In this work, we propose a novel self-matching network to capture sentence *"incongruity"* information by exploring word-to-word interactions. In particular, we calculate the joint information in each word-to-word pair in the input sentence to build a self-matching attention vector, based on which we attend the sentence and build its representation vector. Such a network allows sentence to match within itself word by word and cater to the words of conflict sentiments. In addition, we incorporate a bi-directional LSTM network into our proposed network to retain compositional information. We concatenate incongruity information and compositional information through a Low-rank Bilinear Pooling method to control for potential information redundancy without losing discriminative power. Experiment results on publicly available datasets demonstrate that our model significantly outperforms extant baselines on standard evaluation metrics including precision, recall, $F1$ score and accuracy.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**.

KEYWORDS

Sarcasm detection, self-matching, Bi-LSTM, co-attention

ACM Reference Format:
Tao Xiong, Peiran Zhang, Hongbo Zhu, and Yihui Yang. 2019. Sarcasm Detection with Self-matching Networks and Low-rank Bilinear Pooling. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313735

*This work was finished when the first and the fourth authors were was working at Alibaba Group.
†Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313735

1 INTRODUCTION

Sarcasm, which refers to *"a sharp and often satirical or ironic utterance designed to cut or give pain"*[1], is not unusual in online communities such as social media and e-commerce platforms. Sarcasm detection is significant for many language understanding tasks, such as opinion mining and sentiment classification [27]. According to Ghosh and Veale, human's ability to detect sarcasm mainly relies on social cues such as sentiment, scenario and people's intentions [10]. However, as a computational task, using learning models to detect such a complicated linguistic expression is challenging. The sophistication of automatic sarcasm detection mainly comes from the difficulty in identifying its proper context dependency, such as humorous scenario [11], or discriminating the contradictory between literal expression and author's underlying intention, such as irony and metaphor [35]. Following are some sarcastic examples collected from Twitter by Riloff et al. [34]:

- *"A wonderful day of starting work at 6am"*.
- *"I love finals week! #justkidding #stressed"*
- *"I am working hard to be this poor"*.
- *"Nothing makes me happier than getting on the highway and seeing break lights light up like a Christmas tree"*.
- *"Oh thank GOD, our entire office email system is down"*.

These examples demonstrate a strong disparity between authors' intended emotions and true contexts. For example, *"starting work at 6am"* and *"finals week"* are less likely to be enjoyed. Joshi et al. defines such a disparity as *"incongruity"*, referring to contradictory implications between words and phrases [16].

Sarcasm is a widely studied topic in cognitive science and linguistics [7, 8, 14]. However, the study of sarcasm in computational linguistics community is still at its nascent stage [10]. Traditional machine learning based sarcasm detection approaches typically consider it as a binary classification task and emphasize the importance of exploring predictive discrete features [42], including both content features (e.g. n-grams, sentiment lexicons, incongruity) [29, 33, 34], and contextual features (e.g. historical behavior, contextual environment) [2, 30]. These approaches incur nonnegligible research efforts on feature engineering and the modeling process is usually inefficient. In recent times, scholars began to explore neural features based on word embeddings [26, 28] and cultivate deep

[1]Defined by Merriam-Webster dictionary at https://www.merriam-webster.com.

networks, such as Convolutional Neural Networks(CNN), Gated Recurrent Units (GRU), Long Short Term Memory (LSTM) networks, and Attention networks to improve modeling efficiency as well as model performance [1, 10, 35, 42].

Current state-of-the-art sarcasm learning models mainly rely on identifying and exploiting the incongruity patterns embedded in sentences [34, 35]. For example, Riloff et al. address a common incongruity pattern that a sentence contains a positive sentiment which is adhere to a negative scenario or vice versa [34]. A few other studies utilize the incongruity information between target sentence and authors' historical behaviors [1] or contextual information[11]. Approaches utilizing historical information or contextual information rely on extra inputs other than the target sentence itself. In the most recent state-of-the-art model, Tay et al. suggest that the incongruity can be identified by comparing word-to-word interactions and they compute the embedding similarity between every word-to-word pair in the input sentence to search for conflict sentiments [35].

In this work, we aim to predict sarcasm by using sentence information exclusively to save data collecting efforts. We follow the word-to-word comparison approach suggested by Tay et al. [35] to search for incongruity within input sentence. However, Tay et al. compute the joint information between words simply by using the inner product between word embeddings [35]. In our proposed self-matching network, we improve the word-to-word comparison approach by proposing a novel self-matching network based on the "co-attention" mechanism suggested by Lu et al. [23]. The original co-attention mechanism is designed to attend two input feature maps simultaneously, and we employ a similar approach to attend embeddings simultaneously to capture the joint information between every word-to-word pair in the input sentence. Such a comprehensive comparison mechanism allows us to fully capture any potential incongruity information generated by the sentiment conflicts between words. In addition, as Tay et al. suggest, compositional and sequential information of sentence is also important for sarcasm detection and existing state-of-the-art models rely on deep and sequential neural networks such as GRU and LSTM [35]. Therefore, we also incorporate sentence compositional information as prediction input by employing a bi-directional LSTM encoder [13], to complement the self-matching network. Our proposed network generates two sentence representations separately: one captures incongruity information, the other captures compositional information. We do not input them directly into the final prediction layer because of potential information redundancy. Instead, we propose a bilinear pooling method based on Low-rank Bilinear Pooling (LRBP) [21] to reduce input dimension without losing sentence representation's discriminative power.

We evaluate model performance by conducting extensive experiments on several publicly available datasets. Experiment results demonstrate two major findings: first, our proposed self-matching network outperforms most of extant neural network based sarcasm detection models on standard evaluation metrics including precision, recall, $F1$ score and accuracy, and on some datasets, we notice that our model improves $F1$ score and accuracy of existing state-of-the-art performance by over 10%. Second, sequential networks such as LSTM are known for its power on capturing compositional and sequential information, we compare the self-matching network with and without the bi-directional LSTM encoder to evaluate the effectiveness of cultivating compositional information on sarcasm detection task. Experiment results demonstrate that in most circumstances, employing a sequential neural network encoder to complement the self-matching network helps improve model performance.

The rest of this paper is organized as follows: in section 2, we briefly review extant related studies; in section 3 we describe our proposed model architecture in detail, specifically, we illustrate how self-matching network, bi-directional LSTM and the low-rank bilinear pooling method works; in section 4 we demonstrate our experimental settings and discuss experiment results; we conclude this paper in section 5.

2 RELATED WORKS

Sarcasm is a widely studied linguistic phenomenon by cognitive science and linguistic scholars [7, 8, 14], the study of sarcasm in the computational linguistic community is still at the beginning[10]. Extant studies on sarcasm detection can be classified into three main streams: (1) rule based approaches; (2) machine learning approaches based on discrete features; and (3) neural network approaches.

Rule based approaches are the most straightforward methods for sarcasm detection. Researchers attempt to identify sarcastic expressions based on typical linguistic rules and representative indicators. Among these obvious rules and routines, hash-tags attached by authors themselves, such as *#sarcasm*, *#not true*, *#just kidding*, are widely explored by previous studies [9, 25, 37]. Veale and Hao suggest that some unique simile types, such as sentences containing the pattern of "as xx as a xx" (e.g., *"as private as a park-bench"*, *"as useful as a chocolate teapot"*), can be a strong predictor for ironic expressions [38]. In social contexts, sarcasm is usually expressed verbally with heavy tonal stress or certain body gestures such as eye rollings [5], these pieces of behavioral information are unable to be recovered in automatic sarcasm detection though, certain syntactic and semantic text patterns, such as hyperboles identification [5] and interjections and punctuations [8] can still be utilized.

Rule based approaches solely rely on fixed patterns. In order to make improvements, scholars began to exploit a wide range of discrete features including pattern features and semantic features. Pattern features such as n-grams features and occurrence patterns, usually relate to words' syntactic patterns without considering words' implications [3, 16, 17, 22, 31, 33]. For example, Reyes and Rosso use n-grams, which are sequences combining 2 up to 7 words, to search for a set of recurrent words carrying sarcasm information [31]; Barbieri et al. propose an Italian sarcasm detection model based on bag-of-words [3]. Semantic features rely on words' implications and meanings. For example, ambiguity is an important aspect of sarcasm because sarcastic expressions usually present words with ambiguous meanings [4, 32]. Scholar also explore the relationship between contexts and behaviors [19, 30, 40]. Among these various semantic features, incongruity based features are the most widely explored ones, incongruity information is usually identified by searching for negative words in positive context or vice versa [3, 6, 12]. It is worth pointing out that traditional machine learning approaches usually employ various pattern features and semantic

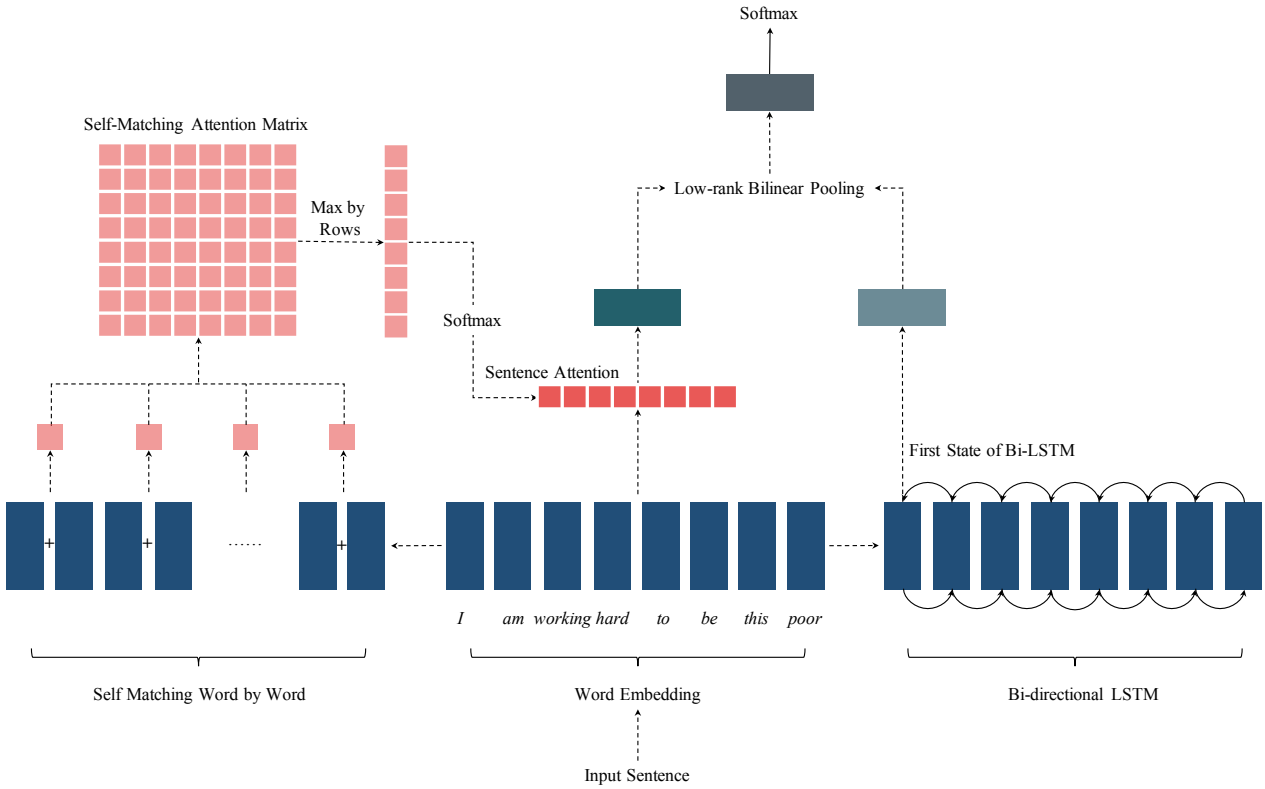


Figure 1: Model architecture. The left part is the self-matching network which is used to generate an attended sentence feature vector based on the self-matching process to search for incongruity information; and the right part is a bi-directional LSTM network used to capture sentence compositional information. We concatenate these two sentence feature vectors by using a low-rank bilinear pooling method to reduce the dimension of the final sentence representation vector.

features simultaneously. Joshi et al. provide a more comprehensive overview on studies based on feature engineering[15] .

Discrete feature based approaches incur nonnegligible research efforts on feature engineering. In recent years, scholars began to employ neural network approaches to detect sarcastic expressions. Joshi et al. are among the very first few studies demonstrating the effectiveness of word embeddings on sarcasm classification tasks [18]. Zhang et al. design a bi-directional Gated Recurrent Neural Network (GRNN) based on neural features, and they also explore how to automatically learn content and context embeddings to make improvements[42]. Amir et al. employ a Convolutional Neural Network (CNN) model to automatically learn *"user embeddings"* from users' historical behavior (e.g. previous tweets), and they detect sarcasm by searching for the incongruity between user embeddings and target content embeddings [1]. Ghosh and Veale combine Deep Neural Networks (DNN) with Long Short-Term Memory(LSTM) to capture sentences' sequential information[10]. In their following study, Ghosh and Veale incorporate user's mood information as well as contextual information into a neural architecture containing two CNN layers and one LSTM layer and achieve improved model accuracy [11] .

Neural network approaches achieve the state-of-the-art performance for sarcasm detection, according to Tay et al., sarcasm detection relies on semantic relationships between words and phrases, yet most sequential neural networks such as CNN, RNN and LSTM, still lack the capability to capture the incongruity and contrast between words, especially when those words are located relatively far from each other. Therefore, they propose a neural network to attend a sentence representation by comparing word-to-word embeddings, and an LSTM encoder to capture sequential information. The so-called *intra-attention* mechanism allows their model to search for conflict sentiments as well as maintain compositional information. This model achieves the new state-of-the-art performance on sarcasm detection [35].

Inspired by the concept of searching for words' contrast and incongruity [29, 33–35], we propose a novel Self-matching Sarcasm Detection model (SMSD) based on a modified "co-attention" mechanism [23]. The major difference between our approach and Tay et al. on capturing word-to-word information is that Tay et al. use inner product operation to capture the joint information between words [35], however, we believe that the inner product operation dilutes information across different words and it only captures words' co-occurrence patterns without considering their sentiment information. As an improvement, we calculate the joint information by introducing a parameter matrix between two word embedding vectors, it helps capture the correlation between words

and is more sensitive on attending sentiment conflicts. We will compare our experiment results with [35] to validate our model performance.

3 MODEL DESCRIPTION

In this section, we describe our proposed model in detail. Model architecture is depicted in Figure 1.

3.1 Sentence Representation

We begin by training a word embedding space $\mathbf{E} \in \mathbb{R}^{V \times k}$, where V is the size of vocabulary and k is the size of dimension for each word vector. For any word in our vocabulary, we assign a vector $\mathbf{e} \in \mathbb{R}^k$ as its feature vector. Feature vectors are associated with rows in the embedding space $\mathbf{E}$. For any sentence s, we build a feature map by combining its word embeddings: $\mathbf{S} = [\mathbf{e}_1^{\mathrm{T}}, \mathbf{e}_2^{\mathrm{T}}, \ldots, \mathbf{e}_n^{\mathrm{T}}]$, $\mathbf{S} \in \mathbb{R}^{k \times n}$, where n is the number of words in the input sentence.

3.2 Self-matching Network

The purpose of our proposed self-matching network is to generate a attended feature vector for the input sentence: $\boldsymbol{f}_a = \mathbf{S} \cdot \boldsymbol{a}$, where $\boldsymbol{a} \in \mathbb{R}^n$ is the self-matched attention vector. As previous studies have found that semantic incongruity is a strong predictor for sarcasm [16], therefore, attention vector $\boldsymbol{a}$ is designed to capture sentence incongruity. In this section, we discuss the specifics of how we calculate the self-matched attention vector.

In the recent state-of-the-art model, Tay et al. propose that the incongruity embedded in a sentence mainly comes from conflict sentiments between words, and they take a deep exploration into this concept by considering the interaction between every word-to-word relationship. In their proposed *"intra-attention"* network, they calculate inner product value for every word-to-word pair as the interaction information. For example, the interaction information $w_{i,j}$ between word i and j is computed as:

$$w_{i,j} = \mathbf{e}_i \cdot \mathbf{e}_j^{\mathrm{T}} \tag{1}$$

where $\mathbf{e}_i$ and $\mathbf{e}_j$ are word embeddings for i and j [35]. In our self-matching network, we employ a similar approach to search for potential incongruity by comparing word-to-word information. However, using inner product operation to compute joint information absorbs and dilutes too much information across different words. Arithmetically, inner product operation aims to compare the similarity between two feature vectors, it lacks the ability to take sentiment information into consideration. Word embedding aims to capture the occurrence patterns between words, not exclusively cater to the sentiment. For example, in a pre-trained word embedding space, the inner product between the vector for *"Loud"* and the vector for *"Silent"* usually is larger than the inner product between *"Loud"* and *"Noisy"*, yet the inner product between *"Loud"* and *"New"* also is likely to be significant because these two words are less likely to have similar occurrence patterns. Therefore, the simple inner product operation might not be able to fully capture sentiment conflicts between words. In the self-matching network, we introduce a parameter matrix based on "co-attention" approach to address this limitation.

Lu et al. propose a "co-attention" network to address the Visual Question Answering (VQA) task [23]. In particular, they introduce an affinity matrix $\mathbf{C}$ to attend input picture feature map $\mathbf{V}$ and text question representation $\mathbf{Q}$. $\mathbf{C}$ is calculated by:

$$\mathbf{C} = tanh(\mathbf{Q} \cdot \mathbf{W}_a \cdot \mathbf{V}) \tag{2}$$

where $\mathbf{W}_a$ contains attention weights based on which Lu et al. adopt a joint activation approach (e.g. maximize by rows and columns) to adjust attention weights for $\mathbf{V}$ and $\mathbf{Q}$ simultaneously. We modify this approach by introducing a weight matrix between word-to-word pair to improve the flexibility of capturing joint information between words.

Specifically, for word pair $(\mathbf{e}_i, \mathbf{e}_j)$, we compute the joint feature vector $w_{i,j}$ as follows:

$$w_{i,j} = tanh(\mathbf{e}_i \cdot \mathbf{M}_{i,j} \cdot \mathbf{e}_j^{\mathrm{T}}) \tag{3}$$

where $\boldsymbol{w}_{i,j} \in \mathbb{R}$ measuring the joint information between word i and word j, and $\mathbf{M}_{i,j} \in \mathbb{R}^{k \times k}$ is a parameter matrix.

We build a self-matching information matrix $\mathbf{W}$ based on all joint information $w_{i,j}, i, j \in (1, 2, ..., n)$:

$$\mathbf{W} = \begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,1} & w_{2,2} & \cdots & w_{n,n} \end{pmatrix} \tag{4}$$

After having $\mathbf{W}$, we use a maximization activation approach to calculate the self-matched attention vector $\boldsymbol{a}$. We first calculate an intermediate vector $\boldsymbol{m} \in \mathbb{R}^n$ by maximizing elements in $\mathbf{W}$ by rows:

$$m_i = max(w_{i,1}, w_{i,2}, \ldots, w_{i,n}), \forall i \in (1, 2, \cdots, n) \tag{5}$$

and we input $\boldsymbol{m}$ into a standard softmax function to calculate $\boldsymbol{a}$: $\boldsymbol{a} = Softmax(\boldsymbol{m})$. Softmax function is adopted for the purpose of normalization.

3.3 Bi-direction LSTM

In our experiment we first propose an approach by inputing the attended sentence feature vector generated by the self-matching network $\boldsymbol{f}_a$ into the prediction layer (denoted as SMSD). However, we notice that the self-matching network only aims to capture incongruity information contained in sentence, it lacks the ability to cultivate sentence's compositional information, which as suggested by previous studies, is also critical for sarcasm detection [35]. Therefore, as a complement to the self-matching network, we propose a second approach which employs a standard Bi-directional LSTM encoder (Bi-LSTM)[13] to capture the compositional information for each input sentence (denoted as SMSD-BiLSTM). Bi-LSTM contains a forward LSTM operation ($\overrightarrow{LSTM}$) which reads the clause information from word 1 to word n; and a backward LSTM operation ($\overleftarrow{LSTM}$) which reads the clause information from word n to word i. Hidden states generated from forward LSTM and backward LSTM are concatenated as the hidden states for Bi-LSTM [2].

For each input sentence, we feed its original word embeddings into the Bi-LSTM encoder and we define the hidden state at the ith

[2]LSTM is based on a non-linear parameterized gating mechanism, as it's one of the most popular neural networks in NLP community, we skip the technical detail of LSTM and Bi-LSTM in this paper for brevity.

time step as:

$$\boldsymbol{h}_i = [\overrightarrow{LSTM}(\mathbf{e}_i), \overleftarrow{LSTM}(\mathbf{e}_i)], \forall i \in (1, 2, \cdots, n) \tag{6}$$

where n is the maximum number of words in the input sentence, $\mathbf{e}_i$ is the feature vector for the corresponding word and $\boldsymbol{h}_i \in \mathrm{R}^d$ is the hidden output of the bi-directional LSTM encoder at time-step i. Hyperparameter d is the dimension of the output vector.

Instead of using all hidden states generated from the Bi-LSTM encoder, we follow the approach proposed by Tay et al. and only adopt the first hidden state as the output of the Bi-LSTM encoder [35][3]. We define the sentence feature vector generated by Bi-LSTM encoder as follows:

$$\boldsymbol{f}_l = \boldsymbol{h}_1 \tag{7}$$

where $\boldsymbol{f}_l \in \mathbb{R}^d$, and d is a hyperparameter.

3.4 Low-rank Bilinear Pooling

We have two sentence feature vectors: $\boldsymbol{f}_a$ generated by the self-matching network, and $\boldsymbol{f}_l$ generated by the Bi-LSTM encoder. We can simply concatenate these two feature vectors for the final prediction. Self-matching network aims to capture word-to-word interactions and it is not intended to capture sequential information, but it might still contain compositional information because of the self-matching operation between adjacent words. Therefore, the two feature vectors $\boldsymbol{f}_a$ and $\boldsymbol{f}_l$ are of high possibility to have redundant sentence information and it is unnecessary to input all of them into the final prediction. In this case, we employ a Low-rank Bilinear Pooling (LRBP) method based on Hadamard product to reduce the dimension of the final input vector to control for potential information redundancy without reducing feature vector's discriminative power [21].

Considering the two feature vectors $\boldsymbol{x} \in \mathbb{R}^N$ and $\boldsymbol{y} \in \mathbb{R}^M$, standard bilinear pooling method uses a quadratic expansion of linear transformation to pool information from input vectors [21] as follows:

$$f_i = \boldsymbol{x}^{\mathrm{T}} \cdot \mathbf{W}_{p,i} \cdot \boldsymbol{y} + g_i \tag{8}$$

where $\mathbf{W}_{p,i} \in \mathbb{R}^{N \times M}$ is a weight matrix for output f_i and g_i is the bias. If L is the number of the output features, the total number of parameters including bias is $L \times (N \times M + 1)$. As an improvement, Kim et al. propose a Low-rank Bilinear Pooling (LRBP) method to reduce the rank of the weight matrix $\mathbf{W}_p$ with less parameters to be learned. Specifically, they decompose the matrix $\mathbf{W}_{p,i}$ as $\mathbf{W}_{p,i} = \mathbf{U}_i \cdot \mathbf{V}_i^{\mathrm{T}}$, where $\mathbf{U}_i \in \mathbb{R}^{N \times d}$ and $\mathbf{V}_i \in \mathbb{R}^{M \times d}$. Based on this concept, they propose the following equation to calculate a output feature vector $\boldsymbol{f} \in \mathbb{R}^c$:

$$\boldsymbol{f} = \mathbf{P}^{\mathrm{T}} \cdot (\mathbf{U}^{\mathrm{T}} \cdot \boldsymbol{x} \circ \mathbf{V}^{\mathrm{T}} \cdot \boldsymbol{y}) + \boldsymbol{g} \tag{9}$$

where $\circ$ represents the Hadamard product, the input of which are two matrices of the same dimension, and it produces another matrix of the same dimension where each element (i, j) in the output matrix is the arithmetic product of elements (i, j) in the original two matrices. $\mathbf{P} \in \mathbb{R}^{d \times c}$, $\mathbf{U} \in \mathbb{R}^{k \times c}$ and $\mathbf{V} \in \mathbb{R}^{d \times c}$ are parameters to be learned and $\boldsymbol{g} \in \mathbb{R}^c$ is bias, c and d are hyperparameters.

[3]Tay et al. adopts standard LSTM encoder in their network and use the last hidden state as the output. In this paper we adopt the Bi-LSTM encoder, therefore, the first hidden state contains the final information of the Bi-LSTM operation.

Table 1: Descriptive statistics

Dataset	Train	Test	Total
Reddit(/r/movies)	13,515	1,504	15,019
Reddit(/r/technology)	12,136	1,349	13,485
IAC-V1	1,794	200	1,794
IAC-V2	4,078	454	4532
Tweets(Riloff)	1,368	588	1,956
Tweets(Ghosh)	51,189	3,742	54,931

In this work, we follow the concept of LRBP to pool information from two input feature vectors: $\boldsymbol{f}_a \in \mathbb{R}^k$ and $\boldsymbol{f}_l \in \mathbb{R}^d$. Considering the fact that sentence representation vector usually doesn't employ very large number of dimensions, we simplify the approach by removing the parameter matrix $\mathbf{P}$ in equation (9) to reduce the parameters to be learned and do not further reduce the input dimension. We calculate the final projection feature vector for the input sentence as:

$$\boldsymbol{f} = \mathbf{U}^{\mathrm{T}} \cdot \boldsymbol{f}_a \circ \mathbf{V}^{\mathrm{T}} \cdot \boldsymbol{f}_l + \boldsymbol{g} \tag{10}$$

where $\boldsymbol{f} \in \mathbb{R}^c$, and c is a hyperparameter.

We use $\boldsymbol{f}$ as the final feature vector for the input sentence and we input $\boldsymbol{f}$ into a standard softmax classification layer to make the sarcasm prediction:

$$\boldsymbol{p}_i = Softmax(\mathbf{W}_f \cdot \boldsymbol{f} + \boldsymbol{b}), \tag{11}$$

where $\boldsymbol{p}_i \in \mathbb{R}^2$ representing the probability of whether the input sentence is sarcastic or not, and $\mathbf{W}_f \in \mathbb{R}^{2 \times c}$, $\boldsymbol{b} \in \mathbb{R}^2$ are parameters to be learned.

3.5 Training Objective

For the final sarcasm classification task, we optimize a standard cross-entropy loss function:

$$J(\theta) = -\sum_{i=1}^{N} [y_i \cdot log p_i + (1 - y_i) \cdot log(1 - p_i)] + \lambda \cdot R \tag{12}$$

where N is the size of training dataset, y_i is the true label for sentence i. $\theta = \{\mathbf{M}_{i,j}, \mathbf{U}, \mathbf{V}, \boldsymbol{g}, \mathbf{W}_f, \boldsymbol{b}\}$ are model parameters. $R = \|\theta\|_{L2}$ is the standard L2 regularization term and λ is a hyperparameter measuring the weight of regularization term. d is the hyperparameter in the bi-directional LSTM encoder referring to the number of dimension of the output feature vector. c is the hyperparameter in the low-rank bilinear pooling process referring to the number of dimension of the final sentence representation vector.

4 EXPERIMENTS

4.1 Datasets

To validate our model performance and make it comparable with baseline models, we follow previous studies and evaluate our model on two datasets collected from Reddit and another two datasets collected from IAC (Internet Argument Corpus) [10, 35, 42]. In addition, we notice that existent studies also test models on two popular Twitter datasets provided by Riloff et al. [34] and Ptáček et al. [29]. However, these two studies only provide Tweets IDs, and we are only able to recollect a comparable dataset with Riloff et al.[34]

but fail to match the dataset provided by Ptáček et al [29] [4]. As a complement, we test our model on a recent collected Twitter dataset provided by Ghosh and Veale [11]. Reddit and Twitter datasets mainly contain short texts, while IAC datasets focus on long texts. Descriptive statistics for all six datasets are summarized in Table 1.

- **Reddit**: Reddit is one of the world's largest online communities with millions of users and hundreds and thousands of topics. Provided by Khodak et al. [20], posts in Reddit dataset are annotated based on the '/s' tag left by authors themselves. We use two subsets: */r/movies* and */r/technology* that are tested by Tay et al. [35] to make our experiment results comparable [5].
- **IAC**(Internet Argument Corpus): the original corpus was collected by Walker et al. [39] from the online debate forum *4forums.com* to study political debates on online communities, Lukin and Walker annotate this corpus for sarcasm detection [24]. We test our model on the same two versions of the dataset used by Tay et al. [34], denoted as IAC-V1 and IAC-V2 respectively[6].
- **Tweets**: Twitter is the world's biggest microblogging platform where users can share any of their news, ideas and thoughts with their followers. Two twitter based datasets are widely used by NLP scholars: Riloff et al. provide a dataset with sentences manually annotated and checked, it is also known as the very first public tweets sarcasm dataset [34]; Ghosh and Veale collect a dataset in which tweets are labeled by hash-tags such as *#sarcasm*, *#sarcastic* and *#ironie* [29], and they also employ a feedback-based system which allows them to contact tweet authors to examine the validity of the sarcasm labels.

4.2 Baseline Models

We compare our model with following neural network based models to evaluate our model:

- **NBOW** is a baseline model using the average of word embedding vectors as sentence representation, which is adopted as the input of a standard logistic regression.
- **ATT-LSTM** is an attention based LSTM network proposed by Yang et al. [41], it employs an attention mechanism on all hidden states of the LSTM encoder. Although this model is not exclusively designed for sarcasm detection, it achieves outstanding performance on this task.
- **GRNN** is a bi-directional gated recurrent neural network model proposed by Zhang et al. [42], it employs a gated pooling method to extract content features from a gated recurrent neural network, and also employs a standard pooling method to extract contextual features. This model is among the very first models utilizing neural features for sarcasm detection and demonstrates improved model accuracy compared to traditional discrete feature engineering approaches.
- **CNN-LSTM-DNN** is proposed by Ghosh and Veale [10], it combines Convolutional Neural Network(CNN), Long Short-Term Memory(LSTM) and Deep Neural Network(DNN) into its model architecture. It employs two CNN layers with 1D convolution and two LSTM layers for feature engineering and employs a DNN layer for prediction. it was once the state-of-the-art model for sarcasm detection before Tay et al. [35].
- **SIARN** and **MIARN** are the state-of-the-art models for sarcasm detection [35] . It utilizes an intra-attention mechanism to overcome limitations from sequential neural networks and could better capture words' incongruities. **SIARN** employs a single-dimension intra-attention network and **MIARN** employs a multi-dimension intra-attention network.

4.3 Experimental Settings

To make our experiment results comparable with existent models, we align our experimental settings with Tay et al. [35]. We preprocess our datasets by removing words that appear only once in datasets, and sentences and documents with less than 5 tokens or contain hyper-links are also removed. Word embedding space in our model is generated by GloVe [28] with the dimension of word feature vector k fixed at 100 and we fine-tune the embedding during the training process. Considering the typical length of reviews and documents contained in our six datasets, we fix the length of each input sentence at (20, 20, 60) for datasets from Reddit, Twitter and IAC respectively.

All models are developed using TensorFlow and are trained on a GTX 1080Ti GPU. For the training process, we use the RMSProp optimizer[36] to optimize model parameters with learning rate equals to 0.01 and we train maximum 200 epochs for each model. The training batch size for all datasets is tuned amongst {64, 256, 512}. The L2 regularization is set to 10^{-5} for short texts contained in Reddit and Twitter datasets, and 10^{-3} for long texts covered in IAC datasets. Hyperparameter d in the Bi-LSTM encoder and hyperparameter c in the LRBP process are both fixed at 100.[7]

4.4 Experiment Results

We compare our model with baselines based on standard evaluation metrics including precision, recall, $F1$ score and the accuracy[8]. Experiment results on different datasets are reported in Table 2, 3 and 4.

On the two datasets collected from Reddit, we observe our models improve the state-of-the-art performance substantially. SMSD and SMSD-BiLSTM dominate previous baselines in precision, recall, $F1$ score and accuracy. The specific improvements differ in terms of evaluation metrics and datasets: on the Reddit(/r/movies) dataset, compared with the best performing baseline model, SMSD and SMSD-BiLSTM improve model precision by around 13%, improve recall by around 15-16%, improve the $F1$ score by around 11%

[4]We find the original Riloff dataset on a personal Github page and the dataset can be downloaded at https://github.com/przhang/Sarcasm-Detection. However, we are unable to find the Ptáček dataset from any reliable sources, and after 4 years passed, a substantial amount of original tweets are deleted and we are unable to collect them from Twitter.

[5]We download Reddit datasets from http://nlp.cs.princeton.edu/SARC/2.0/.

[6]We download IAC datasets from https://nlds.soe.ucsc.edu/sarcasm1 and https://nlds.soe.ucsc.edu/sarcasm2 respectively.

[7]As an important baseline, models proposed by Tay et al. [35] are not open-sourced yet, therefore, we align most of our hyperparameters with Tay et al. to make experiment results comparable.

[8]Precision, recall, $F1$ are calculated based on positive samples and accuracy is calculated by averaging both the accuracies in positive and negative samples.

Table 2: Experiment results on Reddit datasets. Best result in previous baselines is in *italic*, and best result in our approach is in bold. Experiment results of baseline models are collected from Tay et al. [35].

	Reddit(/r/movies)				Reddit(/r/technology)			
Model	Precision	Recall	$F1$	Acc	Precision	Recall	$F1$	Acc
NBOW	67.33	66.56	66.82	67.52	65.45	65.62	65.52	66.55
ATT-LSTM (Yang et al. [41])	68.11	67.87	67.94	68.37	68.20	68.78	67.44	67.22
GRNN (Zhang et al. [42])	66.16	66.16	66.16	66.42	66.56	66.73	66.66	67.65
CNN-LSTM-DNN (Ghosh and Veale [11])	68.27	67.87	67.95	68.50	66.14	66.73	65.74	66.00
SIARN (Tay et al. [35])	69.59	*69.48*	69.52	69.84	*69.35*	*70.05*	*69.22*	69.57
MIARN (Tay et al. [35])	*69.68*	69.37	*69.54*	*69.90*	68.97	69.30	69.09	*69.91*
SMSD	**83.31**	74.42	78.62	77.79	89.33	**82.83**	85.96	81.54
SMSD-BiLSTM	82.09	**76.01**	**78.93**	**78.52**	**91.09**	81.88	**86.24**	**81.62**

Table 3: Experiment results on IAC datasets. Best result in previous baselines is in *italic*, and best result in our approach is in bold. Results of baseline models are from Tay et al. [35]

	IAC-V1				IAC-V2			
Model	Precision	Recall	$F1$	Acc	Precision	Recall	$F1$	Acc
NBOW	57.17	57.03	57.00	57.51	66.01	66.03	66.02	66.09
ATT-LSTM (Yang et al. [41])	58.98	57.93	57.23	59.07	70.04	69.62	69.63	69.96
GRNN (Zhang et al. [42])	56.21	56.21	55.96	55.96	62.26	61.87	61.21	61.37
CNN-LSTM-DNN (Ghosh and Veale [11])	55.50	54.60	53.31	55.96	64.31	64.33	64.31	64.38
SIARN (Tay et al. [35])	*63.94*	63.45	62.52	62.69	72.17	71.81	71.85	72.10
MIARN (Tay et al. [35])	63.88	*63.71*	*63.18*	*63.21*	*72.92*	*72.93*	*72.75*	*72.75*
SMSD	73.68	58.82	65.42	63.00	**88.76**	64.81	**74.92**	67.40
SMSD-BiLSTM	**75.79**	**59.50**	**66.67**	**64.00**	82.73	**67.54**	74.37	**68.72**

Table 4: Experiment results on Tweets datasets. Best result in previous baselines is in *italic*, and best result in our approach is in bold. Results of baseline models are from Ghosh and Veale [11] and Tay et al. [35]

	Tweets(Riloff)				Tweets(Ghosh)			
Model	Precision	Recall	$F1$	Acc	Precision	Recall	$F1$	Acc
NBOW	71.28	62.37	64.13	79.23	-	-	-	-
ATT-LSTM (Yang et al. [41])	68.78	68.63	68.71	77.69	-	-	-	-
GRNN (Zhang et al. [42])	66.32	64.74	65.40	76.41	-	-	-	-
CNN-LSTM-DNN (Ghosh and Veale [11])	69.76	66.62	67.81	78.72	*73.30*	*71.70*	*72.50*	-
SIARN (Tay et al. [35])	*73.82*	*73.26*	*73.24*	*82.31*	-	-	-	-
MIARN (Tay et al. [35])	73.34	68.34	70.10	80.77	-	-	-	-
SMSD	95.70	**100.00**	97.80	**99.32**	76.39	**72.56**	**74.42**	**80.09**
SMSD-BiLSTM	**96.77**	98.90	**97.83**	**99.32**	**78.65**	68.80	73.40	78.37

and model accuracy by 8-9%. On the Reddit(/r/technology) dataset, SMSD and SMSD-BiLSTM achieve around 10-11% increasing on precision, 12-13% increasing on recall, 16% increasing on $F1$ score and 11% increasing on model accuracy.

On the two datasets collected from IAC, SMSD and SMSD-BiLSTM also achieve outstanding performance. On the IAC-V1 dataset, MIRAN proposed by Tay et al. [35] has the highest recall though, SMSD-BiLSTM achieves around 11% increasing on precision, 3% increasing on $F1$ score and 1% increasing on model accuracy. On the IAC-V2 dataset, MIARN has higher recall and accuracy, yet SMSD has higher precision and $F1$ score. In addition, we also notice that our models are more robust in model performance across two different IAC datasets. For example, the difference of model accuracy between these two datasets is 4.4% for the SMSD, 4.7% for SMSD-BiLSTM, while the difference for SIARN and MIARN are over 9%. Among these baselines, only GRNN has similar stability with our model, while its $F1$ score and accuracy are much lower than ours.

Moreover, SMSD and SMSD-BiLSTM achieve state-of-the-art performance on the two datasets collected from Twitter. On the Rillof dataset, SMSD and SMSD-BiLSTM almost achieve perfect model performance, the accuracy and $F1$ score for both of our

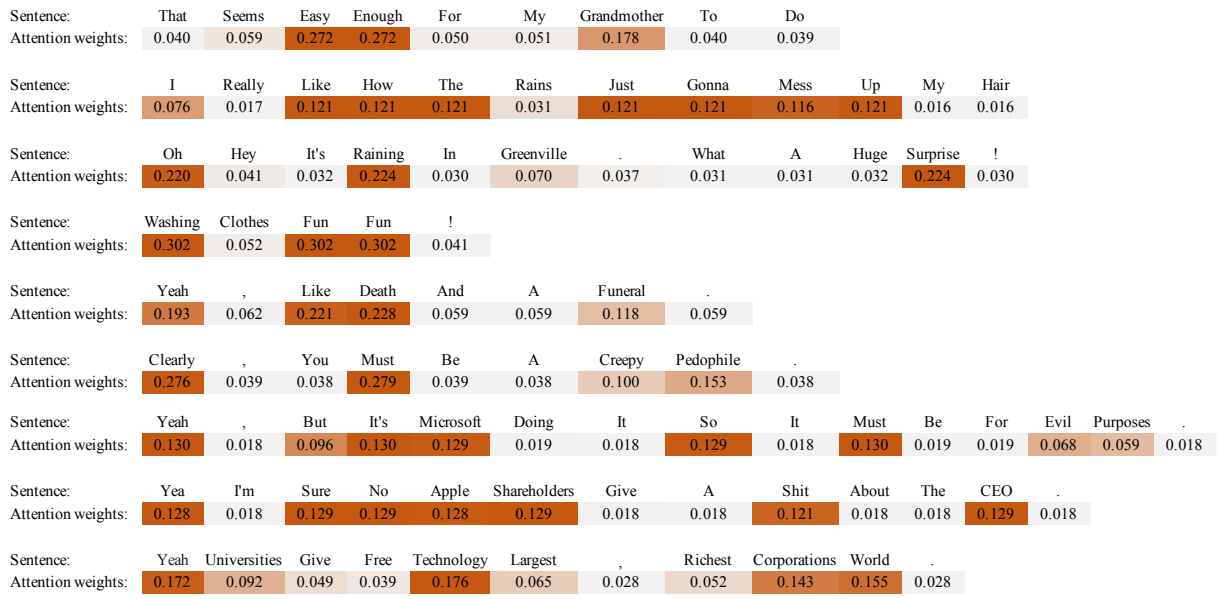


Figure 2: Visualization of Self-matching Results. For each sample, the first sentence shows the original input sentence, and the second sentence demonstrates the attention weights generated from the self-matching network, the darkness of the background varies according to the value of the corresponding attention weight.

models are over 97% [9]. The Ghosh dataset is not widely tested by extant studies yet, we only compare our model performance with the original paper [11] and we still find that our models outperform it, the SDSM-BiLSTM improves precision by around 5%, and SMSD improves recall, $F1$ score by 1% and 2% respectively.

As we have discussed in model description section, we design a self-matching network to compare word-to-word information aiming to search for potential incongruity information; and we also incorporate a bi-directional LSTM encoder to help capture sequential information. We report model performance with and without the Bi-LSTM separately to help us discriminate the value of these two components. As experiment results have demonstrated, in most circumstances, incorporating a bi-directional LSTM encoder will improve model performance. We test our models on six datasets and we find that on the IAC-V1 dataset, the SMSD-BiLSTM outperforms SMSD on four evaluation metrics. SMSD-BiLSTM generally have better performance than SMSD on two Reddit datasets except that SMSD has better precision on Reddit(/r/movies) and better recall on Reddit(/r/technology). These two models have comparable results on IAC-V2 and Tweets(Riloff) considering four evaluation metrics, except that SMSD-BiLSTM has higher accuracy and recall on IAC-V2. We notice that SMSD outperforms SMSD-BiLSTM on Tweets(Ghosh) in terms of recall, $F1$ score and accuracy. These comparison results indicate that the a bi-directional LSTM encoder indeed enriches model input by capturing sentence compositional information, and it helps to generate better sarcasm detection result.

4.5 Model Visualization

In this section we use a few model prediction results to showcase the mechanism of the self-matching network. The self-matching network attends word-to-word information simultaneously and the joint information are captured to attend the original sentence feature map. Therefore, words of conflict sentiments are more likely to be attended by our proposed model. In Figure 2 we depict a few sample sentences with their self-matched attention weights reported. We also demonstrate some bad cases in the prediction result to better understand under what circumstances our models would fail (see Figure 3)[10].

Examples demonstrate that the self-matching network is highly effective on attending words of incongruity information. For example. in the sentence *"That seems easy enough for my grandmother to do"* we notice that the phrase *"easy enough"* and the word *"grandmother"* have the highest attention weights; in sentence *"I really like how the rains just gonna mess up my hair"*, *"like"* and *"mess up"* have relatively high attention weights. These most attended words are usually of opposite sentiments compared to each other and are less likely to be found simultaneously in non-sarcastic expressions. The same pattern can be found across all the examples we provide in Figure 2. Therefore, the self-matching network has a strong power on capturing word pairs of incongruity information that can be taken as a strong indicator for the appearance of sarcasm.

[9]Indeed, the significant improvements on this dataset is beyond our expectation. As aforementioned, Rillof dataset is not provided officially and we collect the Rillof dataset from a personal Github link (as provided in previous section). We test the reliability of the dataset by checking the descriptive statistics of our collected dataset with the statistics reported in previous studies, such as Tay et al. [35] and we find no significant difference between these two datasets. We also notice that the Rillof dataset is unbalanced and evaluation metrics might be biased by prediction result on the dominant class. Therefore, we further examine the values of evaluation metrics from positive samples and negative samples respectively, and we find the results are highly similar.

[10]The authors want to thank two anonymous reviewers for the suggestion of providing a qualitative bad case analysis.

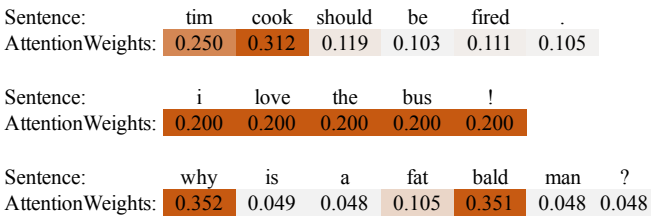


Figure 3: Bad case analysis. Three model prediction bad cases are reported, the first two samples are false negative, and the third sample is false positive.

In Figure 3 we report three bad cases: the first two are false negative and last one is false positive. The first case shows that our model fails to capture *tim cook* as a celebrity name and the sarcasm embedded in this sample replies heavily on external knowledge. In the second case, there are no conflict sentiments between those four words, and we have not further input to know that the author actually hates bus. In addition, the third false positive case *"why"* and *"bald"* have the highest attention weights, this sample sentence is a rhetorical question expression, which is highly similar with sarcastic expression. Therefore, our model is most likely to fail when the sarcasm relies on external knowledge or when the expression is a rhetorical question.

5 CONCLUSION

In this work we propose a novel self-matching network based on a modified co-attention mechanism to cater to sentence incongruity. The self-matching network allows us to capture the interaction between words to search for potential conflict sentiments. We also incorporate a bi-directional LSTM encoder to utilize sentence's compositional information. We conduct extensive experiments on six publicly available datasets. Experiment results demonstrate a strong evidence that our proposed model outperforms most of extant baselines in terms of standard evaluation metrics including precision, recall, $F1$score and accuracy. Our model even improve the state-of-the-art performance on some datasets dramatically. In addition, we also find supportive empirical evidence on utilizing sequential neural networks and low-rank bilinear pooling method can help improve sarcasm prediction result.

6 ACKNOWLEDGMENTS

This research is supported by Alibaba Group.

REFERENCES

[1] Silvio Amir, Byron C Wallace, Hao Lyu, and Paula Carvalho Mário J Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *arXiv preprint arXiv:1607.00976* (2016).
[2] David Bamman and Noah A Smith. 2015. Contextualized Sarcasm Detection on Twitter. *ICWSM* 2 (2015), 15.
[3] Francesco Barbieri, Francesco Ronzano, and Horacio Saggion. 2014. Italian irony detection in twitter: a first approach. In *The First Italian Conference on Computational Linguistics CLiC-it*. 28.
[4] Francesco Barbieri, Horacio Saggion, and Francesco Ronzano. 2014. Modelling sarcasm in twitter, a novel approach. In *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. 50–58.
[5] Santosh Kumar Bharti, Korra Sathya Babu, and Sanjay Kumar Jena. 2015. Parsing-based sarcasm sentiment recognition in twitter data. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*. ACM, 1373–1380.
[6] Mondher Bouazizi and Tomoaki Ohtsuki. 2015. Opinion mining in Twitter: How to make use of sarcasm to enhance sentiment analysis. In *Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on*. IEEE, 1594–1597.
[7] Elisabeth Camp. 2012. Sarcasm, pretense, and the semantics/pragmatics distinction. *Noûs* 46, 4 (2012), 587–634.
[8] John D Campbell and Albert N Katz. 2012. Are there necessary conditions for inducing a sense of sarcastic irony? *Discourse Processes* 49, 6 (2012), 459–480.
[9] Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the fourteenth conference on computational natural language learning*. Association for Computational Linguistics, 107–116.
[10] Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis*. 161–169.
[11] Aniruddha Ghosh and Tony Veale. 2017. Magnets for Sarcasm: Making sarcasm detection timely, contextual and very personal. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 482–491.
[12] Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in Twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers-Volume 2*. Association for Computational Linguistics, 581–586.
[13] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 6645–6649.
[14] Stacey L Ivanko and Penny M Pexman. 2003. Context incongruity and irony processing. *Discourse Processes* 35, 3 (2003), 241–279.
[15] Aditya Joshi, Pushpak Bhattacharyya, and Mark J Carman. 2017. Automatic sarcasm detection: A survey. *ACM Computing Surveys (CSUR)* 50, 5 (2017), 73.
[16] Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing context incongruity for sarcasm detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 757–762.
[17] Aditya Joshi, Vaibhav Tripathi, Pushpak Bhattacharyya, and Mark J Carman. 2016. Harnessing Sequence Labeling for Sarcasm Detection in Dialogue from TV SeriesFriends'. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 146–155.
[18] Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark Carman. 2016. Are Word Embedding-based Features Useful for Sarcasm Detection? *arXiv preprint arXiv:1610.00883* (2016).
[19] Jihen Karoui, Farah Benamara, Véronique Moriceau, Nathalie Aussenac-Gilles, and Lamia Hadrich Belguith. 2015. Towards a contextual pragmatic model to detect irony in tweets. Association for Computational Linguistics (ACL).
[20] Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2017. A large self-annotated corpus for sarcasm. *arXiv preprint arXiv:1704.05579* (2017).
[21] Jin-Hwa Kim, Kyoung-Woon On, Woosang Lim, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. 2017. Hadamard product for low-rank bilinear pooling. In *Proceedings of the 2017 International Conference on Learning Representation*.
[22] CC Liebrecht, FA Kunneman, and APJ van Den Bosch. 2013. The perfect solution for detecting sarcasm in tweets# not. (2013).
[23] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical question-image co-attention for visual question answering. In *Advances In Neural Information Processing Systems*. 289–297.
[24] Stephanie Lukin and Marilyn Walker. 2017. Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. *arXiv preprint arXiv:1708.08572* (2017).
[25] DG Maynard and Mark A Greenwood. 2014. Who cares about sarcastic tweets? investigating the impact of sarcasm on sentiment analysis. In *LREC 2014 Proceedings*. ELRA.
[26] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, L Sutskever, and G Zweig. 2013. word2vec. *URL https://code. google. com/p/word2vec* (2013).
[27] Bo Pang, Lillian Lee, et al. 2008. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval* 2, 1–2 (2008), 1–135.
[28] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.
[29] Tomáš Ptáček, Ivan Habernal, and Jun Hong. 2014. Sarcasm detection on czech and english twitter. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. 213–223.
[30] Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. 2015. Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. ACM, 97–106.
[31] Antonio Reyes and Paolo Rosso. 2012. Making objective decisions from subjective data: Detecting irony in customer reviews. *Decision Support Systems* 53, 4 (2012), 754–760.

[32] Antonio Reyes, Paolo Rosso, and Davide Buscaldi. 2012. From humor recognition to irony detection: The figurative language of social media. *Data & Knowledge Engineering* 74 (2012), 1–12.

[33] Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in twitter. *Language resources and evaluation* 47, 1 (2013), 239–268.

[34] Ellen Riloff, Ashequl Qadir, Prafulla Surve, Lalindra De Silva, Nathan Gilbert, and Ruihong Huang. 2013. Sarcasm as contrast between a positive sentiment and negative situation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 704–714.

[35] Yi Tay, Luu Anh Tuan, Siu Cheung Hui, and Jian Su. 2018. Reasoning with Sarcasm by Reading In-between. *arXiv preprint arXiv:1805.02856* (2018).

[36] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4, 2 (2012), 26–31.

[37] Oren Tsur, Dmitry Davidov, and Ari Rappoport. 2010. ICWSM-A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Online Product Reviews.. In *ICWSM*. 162–169.

[38] Tony Veale and Yanfen Hao. 2010. Detecting ironic intent in creative comparisons.. In *ECAI*, Vol. 215. 765–770.

[39] Marilyn A Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. 2012. A Corpus for Research on Deliberation and Debate.. In *LREC*. 812–817.

[40] Byron C Wallace, Eugene Charniak, et al. 2015. Sparse, contextually informed models for irony detection: Exploiting user communities, entities and sentiment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 1035–1044.

[41] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[42] Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Tweet sarcasm detection using deep neural network. In *Proceedings of COLING 2016, The 26th International Conference on Computational Linguistics: Technical Papers*. 2449–2460.

A First Look at Deep Learning Apps on Smartphones

Mengwei Xu
Key Lab of High-Confidence Software Technologies (Peking University), MoE, Beijing, China
mwx@pku.edu.cn

Jiawei Liu
Key Lab of High-Confidence Software Technologies (Peking University), MoE, Beijing, China
jiaweiliu@pku.edu.cn

Yuanqiang Liu
Key Lab of High-Confidence Software Technologies (Peking University), MoE, Beijing, China
yuanqiangliu@pku.edu.cn

Felix Xiaozhu Lin
Purdue ECE
West Lafayette, Indiana, USA
xzl@purdue.edu

Yunxin Liu
Microsoft Research
Beijing, China
yunxin.liu@microsoft.com

Xuanzhe Liu
Key Lab of High-Confidence Software Technologies (Peking University), MoE, Beijing, China
xzl@pku.edu.cn

ABSTRACT

To bridge the knowledge gap between research and practice, we present the first empirical study on 16,500 the most popular Android apps, demystifying how smartphone apps exploit deep learning in the wild. To this end, we build a new static tool that dissects apps and analyzes their deep learning functions. Our study answers threefold questions: what are the early adopter apps of deep learning, what do they use deep learning for, and how do their deep learning models look like. Our study has strong implications for app developers, smartphone vendors, and deep learning R&D. On one hand, our findings paint a promising picture of deep learning for smartphones, showing the prosperity of mobile deep learning frameworks as well as the prosperity of apps building their cores atop deep learning. On the other hand, our findings urge optimizations on deep learning models deployed on smartphones, protection of these models, and validation of research ideas on these models.

CCS CONCEPTS

• **General and reference → Empirical studies**; • **Human-centered computing → Ubiquitous and mobile computing**.

KEYWORDS

Empirical Study; Deep Learning; Mobile Apps

ACM Reference Format:
Mengwei Xu, Jiawei Liu, Yuanqiang Liu, Felix Xiaozhu Lin, Yunxin Liu, and Xuanzhe Liu. 2019. A First Look at Deep Learning Apps on Smartphones. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313591

1 INTRODUCTION

Smartphones are undoubtedly among the most promising platforms for running deep learning (DL) based applications [3, 6, 8, 27]. Such a huge market is driven by continuous advances in DL, including the introduction of latest neural-network (NN) hardware [45, 47, 57, 103], improvements in DL algorithms [53, 77, 82, 89], and the increased penetration in huge information analytics [49, 54, 79, 81]. The research community has built numerous DL-based novel apps [44, 68, 69, 76, 83, 102]. The industry has also tried to utilize DL in their mobile products. For example, in the latest released Android 9 Pie OS, Google introduces a small feed-forward NN model to enable Smart Linkify, a useful API that adds clickable links when certain types of entities are detected in text [38].

The year 2017 marked the dawn of DL for smartphones. Almost simultaneously, most major vendors roll out their DL frameworks for smartphones, or *mobile DL framework* for short. These frameworks include TensorFlow Lite (TFLite) from Google [37] (Nov. 2017), Caffe2 from Facebook [9] (Apr. 2017), Core ML from Apple [12] (Jun. 2017), ncnn from Tencent [35] (Jul. 2017), and MDL from Baidu [24] (Sep. 2017). These frameworks share the same goal: executing the DL inference task solely on smartphones. Compared to offloading the DL inference from smartphones to the cloud [2, 19, 21], the on-device DL inference better protects user privacy without uploading sensitive data, continues to operate in the face of poor Internet connectivity, and relieves app authors from affording the cost of running DL in the cloud [45, 63, 66, 67, 67, 69, 74, 100, 103].

Following the DL framework explosion, there emerges the first wave of smartphone apps that embrace DL techniques. We deem it crucial to understand these apps and in particular how they use DL, because the history has demonstrated that such *early adopters* heavily influence or even decide the evolution of new technologies [86] – smartphone DL in our case.

To this end, we present the first empirical study on how real-world Android apps exploit DL techniques. Our study seeks to answer *threefold* questions: *what are the characteristics of apps that have adopted DL, what do they use DL for*, and *what are their DL models*. Essentially, our study aims findings on how DL is being used by smartphone apps *in the wild* and the entailed implications, filling a key gap between mobile DL research and practice.

For the study, we have examined an extensive set of Android apps from the official Google Play appstore. We take two snapshots of the app market in early Jun. 2018 and early Sep. 2018 (3 months apart), respectively. Each snapshot consists of 16,500 the most popular apps covering 33 different categories listed on Google Play. We derive

Xuanzhe Liu is the paper's corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW'19, May 2019, San Francisco, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313591

insights by inspecting individual apps as well as by comparing the two snapshots. To automate the analysis of numerous Android apps, we build a new analyzer tool that inspects app installation packages, identifies the apps that use DL (dubbed "DL apps"), and extracts DL models from these apps for inspection. To realize such a tool, we eschew from looking for specific code pattern and instead identify the usage of known DL frameworks, based on a rationale that most DL apps are developed atop DL frameworks.

Our key findings are summarized as follows.

Early adopters are top apps (§4.2) We have found 211 DL apps in the set of apps collected in Sep. 2018. Only 1.3% of all the apps, these DL apps collectively contribute 11.9% of total downloads of all the apps and 10.5% of total reviews. In the month of Sep. 2018, the 211 DL apps are downloaded for around 13,000,000 times and receive 9,600,000 reviews. DL apps grow fast, indicating a 27% increase in their numbers over the 3 months in our study.

DL is used as core building blocks (§4.3) We find that 81% DL apps use DL to support their core functionalities. That is, these apps would fail to operate without their use of DL. The number of such DL apps grow by 23% over the period of 3 months.

Photo beauty is the top use (§4.3) DL is known for its diverse applications, as confirmed by the usage discovered by us, e.g. emoji prediction and speech recognition. Among them, photo beauty is the most popular use case: 94 (44.5%) DL apps use DL for photo beauty; 61 (29%) DL apps come from the photography category.

Mobile DL frameworks are gaining traction (§4.4) While full-fledged DL frameworks such as TensorFlow are still popular among DL apps due to their momentum, DL frameworks designed and optimized for constrained resources are increasingly popular. For instance, the number of DL apps using TFLite has grown by 258% over the period of 3 months.

Most DL models miss obvious optimizations (§5.2) Despite well-known optimizations, e.g. quantization which can reduce DL cost by up to two orders of magnitude with little accuracy loss [58], we find only 6% of DL models coming with such optimizations.

On-device DL is lighter than one may expect (§5.3) Despite the common belief that the power of DL models comes from rich parameters and deep layers, we find that DL models used in apps are rather small, with the median memory usage of 2.47 MB and the inference computation of 10M FLOPs, which typically incurs the inference delay of tens of milliseconds. These models are not only much lighter than full models designed for cloud/servers (e.g. ResNet-50 with 200 MB memory and 4G FLOPs inference computations) but also lighter than well-known models specifically crafted for smartphones, e.g. MobileNet with 54 MB memory and 500M FLOPs inference computations.

DL models are poorly protected (§5.4) We find that only 39.2% discovered models are obfuscated and 19.2% models are encrypted. The remaining models are a bit trivial to extract and therefore subject to unauthorized reuse.

Summary of implications: Overall, our findings paint a promising picture of DL on smartphones, motivating future research and development. Specifically, the findings show strong implications for multiple stakeholders of the mobile DL ecosystem. **To app developers:** our findings show that DL can be sufficiently affordable on smartphones; developers, especially individuals or small companies, should have more confidence in deploying DL in their apps; interested developers should consider building DL capability atop mobile DL frameworks; a few app categories, notably photography, are most likely to benefit from DL techniques. **To DL framework developers:** our findings encourage continuous development of frameworks optimized for smartphones; our findings also show the urgent need for model protection as the first-class concern of frameworks. **To hardware designers:** our findings motivate DL accelerator designs to give priority to the layers popular among mobile DL models. **To DL researchers:** our findings suggest that new proposal for optimizing DL inference should be validated on lightweight models that see extensive deployment on smartphones in the wild.

In summary, our contributions are as follows.

- We design and implement a tool for analyzing the DL adoption in Android apps. Capable of identifying the DL usage in Android apps and extracting the corresponding DL models for inspection, our tool enables automatic analysis of numerous apps for DL.
- We carry out the **first** large-scale study of 16,500 Android apps for their DL adoption. Through the empirical analysis, we contribute new findings on the first wave of apps that adopt DL techniques. In the dawn of DL explosion for smartphones, our findings generate valuable implications to key stakeholders of the mobile ecosystem and shed light on the evolution of DL for smartphones. We also publicize our tools and datasets[1].

2 BACKGROUND

DL models and frameworks DL has revolutionized many AI tasks, notably computer vision and natural language processing, through substantial boosts in algorithm accuracy. In practice, DL algorithms are deployed as two primary parts. The first one is *DL models*, which often comprise neuron layers of various types, e.g. convolution layers, pooling layers, and fully-connected layers. Based on the constituting layers and their organizations, DL models fall into different categories, e.g., Convolutional Neural Network (CNN) containing convolution layers, and Recurrent Neural Network (RNN) processing sequential inputs with their recurrent sub-architectures. The second part is *DL frameworks* that produce DL models (i.e. training) and execute the models over input data (i.e. inference). Since a commodity DL framework often entails tremendous engineering efforts, most app developers tend to exploit existing frameworks by major vendors, such as *TensorFlow* from Google.

Deploying mobile DL As training models is intensive in both data and computing [73], smartphone developers often count on cloud servers for modeling training offline prior to app deployment. At app installation time, the trained models are deployed as part of the app installation package. At runtime, apps perform inference

[1] https://github.com/xumengwei/MobileDL

with the trained models by invoking DL frameworks, and therefore execute AI tasks such as face recognition and language translation.

Inference: on-cloud vs. on-device Towards enabling DL on smartphones, model inference can be either offloaded to the cloud or executed solely on smartphones. Offloading to the cloud is a classical use case of Software-as-a-Service (SaaS), and has been well studied in prior work [43, 62, 85, 101]. The mobile devices upload data and retrieve the inference results, transparently leveraging rich data center resources as server-class GPU. Yet, we have observed that the on-device DL inference is quickly gaining large popularity due to its unique advantages of stronger privacy protection, resilience against poor Internet connectivity, and lower cloud cost to app developers. We will present more evidence in the paper. In this work, we focus our empirical study on such on-device deep learning for smartphones.

3 GOAL AND METHODOLOGY

3.1 Research Goal

The goal of our study is to demystify how smartphone apps exploit DL techniques in the wild. Our study focuses on two types of subjects: i) smartphone apps that embrace DL, and ii) the DL frameworks and models used in practice. Accordingly, we characterize the apps, the frameworks, and the models. We will present the results in Section 4 and Section 5 respectively.

Scope We focus our analysis on Android apps, as Android represents a dominant portion of smartphone shipment (88% in the second quarter of 2018) and hence serves a good proxy for the entire smartphone app population [18].

Datasets We retrieve from the Google Play store the full dataset used in this work. We select 16,500 apps in total, which consist of the top 500 free apps with most downloads from each of the 33 categories defined by Google Play[2]. We have crawled two datasets at different moments, June 2018 and September 2018, which are three months apart. The two app datasets have more than 2/3 overlapped apps. For each app, we download its apk file and crawl its meta information (e.g. app description and user rating) from the Google Play web page for analysis. Our analysis primarily focuses on the newer dataset, i.e., Sep. 2018, and the difference between the two data sets, unless specified otherwise.

3.2 Workflow Overview

We design and implement an analyzing tool to enable our research goal on large-scale Android apps. The tool runs in a semiautomatic way, as illustrated in Figure 1.

The very first step of the analyzing tool is identifying DL apps among a given set of Android apps as input. This is achieved via the module named DL Sniffer. The core idea of DL Sniffer, is detecting the usage of popular DL frameworks, instead of directly finding the usage of DL. After identifying DL apps, it performs analysis on those apps. During analysis, we use the manifest files extracted from DL apps via tool aapt [4] and the meta information crawled from the corresponding Google Play web page. The manifest files include information such as package name, app version, required permissions, etc. The web pages include information such as app description, user rating, app developer, etc.

The analyzing tool further extracts DL models from those DL apps. This extraction is achieved via a module called Model Extractor. After extracting DL models, it performs analysis on them. However, we here face the challenge that the models are mostly in different formats. Although developers are investing substantial efforts in integrating different model formats, such as designing a standardized one [25], the ecosystem of DL frameworks is still broken and fragmented nowadays. Thus, when looking into the internal structures of DL models, we substantially leverage the available tools and source of different frameworks. Fortunately, most of the frameworks we investigated (details in Table 2) are open-source and well-documented.

We discuss more details of DL Sniffer and Model Extractor in Section 4.1 and Section 5.1, respectively.

4 APPLICATION ANALYSIS

This section presents our analysis of smartphone DL apps. We first describe our methodology in Section 4.1 and then the following three major aspects of the DL apps:

- The characteristics of DL apps (§4.2): their popularity, their difference from non-DL apps, and their developers.
- The role of DL (§4.3): the popular usage of DL in apps, the categories of DL apps, and evidence that DL is already used as core building blocks of apps.
- An analysis of DL frameworks (§4.4): which frameworks are used, the cost, and their adoption trend over time.

4.1 Methodology: finding DL apps

As a part of our analyzing tool (Section 3.2), DL Sniffer takes apk file(s) as input, and outputs which of them use DL technique. Detecting DL usage is difficult, instead DL Sniffer mainly detects the usage of popular DL frameworks with Android support. Currently, DL Sniffer supports the detection of 16 popular DL frameworks, including *TensorFlow, Caffe, TFLite,* etc, and the details of those frameworks will be presented later in Section 4.4 & Table 2. DL Sniffer uses two ways to mine the usage of DL frameworks: (1) For those who provide native C++ APIs, DL Sniffer first decomposes the apk files via Apktool [5], and extracts the native shared libraries (with suffix ".so"). DL Sniffer then searches for specific strings on the *rodata* section of those libraries. Those strings can be regarded as identifications of corresponding frameworks, and are pre-defined by us. For example, we notice that the shared libraries that use *TensorFlow* always have *"TF_AllocateTensor"* in its *rodata* section. (2) For those who only support Java APIs, DL Sniffer first converts the apk file into smali code via dex2jar [14]. The smali code, which is a disassembled version of the DEX binary used by Android's Davik VM, enables us to carry out static program analysis. DL Sniffer statically goes through the class/method names of smali code and checks whether certain APIs exist. For example, the class *MultiLayerConfiguration* is used in almost every app that embeds DeepLearning4J framework.

Besides detecting the usage of DL frameworks, we also try to identify DL apps that don't use the frameworks listed in Table 2 (called "*no lib*" in this work). Similarly, this is achieved by searching

[2]Different app categories on Google Play can be visited via url https://play.google.com/store/apps/category/XXX, where XXX can be GAME or other category names.

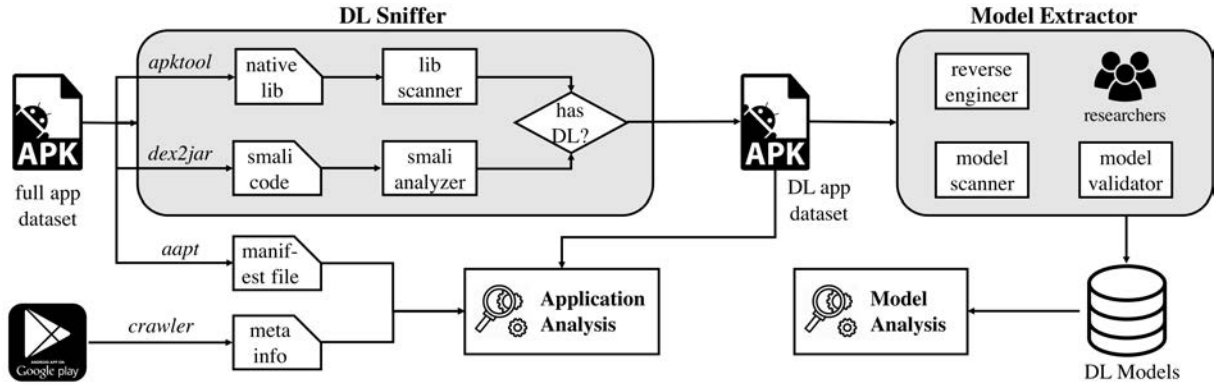


Figure 1: The overall workflow of our analyzing tool.

specific strings on the *rodata* section of native libraries as mentioned above, but the strings we use here are pre-defined such as *"neural network", "convolution", "lstm"*, etc, rather than extracted from existing frameworks. We then manually check the detection correctness through reverse engineering, filtering those don't really have DL usage (false positive). This manual check is also performed on other DL apps detected using the aforementioned approach, to ensure good accuracy in DL identification.

4.2 Characteristics of DL apps

• **Is DL gaining popularity on smartphones?** Our study shows: over our investigation period (June 2018 – Sept. 2018), the total amount of DL apps has increased by 27.1%, from 166 to 211. We further investigate the new downloads and reviews of DL apps within one month of Sept. 2018: in that period, the 211 DL apps are downloaded for around 13,000,000 times and receive 9,600,000 new reviews. The results indicate that a substantial amount of smartphones are running DL apps nowadays.

• **How are DL apps different from non-DL apps?** We investigate the following three aspects with results illustrated in Figure 2.

Downloads and Reviews are typical signal of apps' popularity [75]. As observed, the median number of downloads and reviews of DL apps are 5,000,000 and 41,074 respectively, much larger than non-DL apps, i.e., 100,000 and 1,036 respectively. We also count the download rankings of each DL apps within the corresponding category. The median number of such ranking is 89 among total 500 apps for each category. We deem the above statistics as strong evidences that *top apps are early adopters in deploying DL in mobile apps.* Such phenomenon can be explained that making DL work in the wild, though appealing, takes a lot of engineering efforts. The cycle of developing DL functionality on smartphones includes model construction, data collection, model training, offline/online testing, etc. Thus, many small companies or individual developers lack the resources to exploit DL on their apps.

Ratings show how much appreciation users give to apps. The Figure shows that DL apps and non-DL apps have similar ratings from users, with the same median number 4.3.

App size: as shown in Figure 2, DL apps have much larger apk files than non-DL apps (median number: 35.5MB vs 12.1MB). This

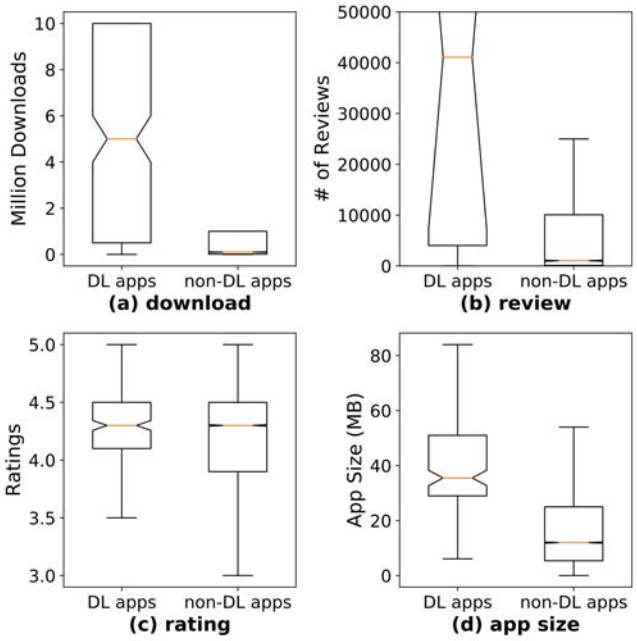


Figure 2: Comparisons between DL apps and non-DL apps on various aspects (a)–(d). Each box shows the 75th percentile, median, and 25th percentile from top to bottom. We manually set the y-axis limits for better presentation in (b): the missed out 75th percentile of DL apps is 324,044.

is reasonable since having DL not only adds DL frameworks and models to the apps, it also confidently indicates that the apps have much richer features.

• **Who are the developers of DL apps?** We also study the developers of DL apps. The results show that the identified 211 DL apps belong to 172 developers (companies), among which 27 developers have more than one DL apps. The developers with most DL apps are "Google LLC" (10) and "Fotoable,Inc" (6). We observe many big companies own more than one DL app, including Google, Adobe, Facebook, Kakao, Meitu, etc. This suggests that those big

Table 1: The DL usage in different apps. Note: as one app may have multiple DL uses, the sum of detailed usage (column 2) might exceed the corresponding coarse usage (column 1).

usage	detailed usage	as core feature
image: 149	photo beauty: 97	94 (96.9%)
	face detection: 52	44 (84.6%)
	augmented reality: 19	5 (26.3%)
	face identification: 8	7 (87.5%)
	image classification: 11	6 (54.5%)
	object recognition: 10	9 (90%)
	text recognition:11	4 (36.3%)
text:26	word&emoji prediction: 15	15 (100%)
	auto-correct: 10	10 (100%)
	translation: 7	3 (42.8%)
	text classification: 4	2 (50%)
	smart reply: 2	0 (0%)
audio: 24	speech recognition: 18	7 (38.9%)
	sound recognition: 8	8 (100%)
other: 19	recommendation: 11	2 (18.1%)
	movement tracking: 9	4 (44.4%)
	simulation: 4	4 (100%)
	abnormal detection: 4	4 (100%)
	video segment: 2	1 (50%)
	action detection: 2	0 (0%)
total: 211		171 (81.0%)

companies are pioneers in adopting DL into their products. We also notice that DL apps from the same developer often have identical DL frameworks. For example, four products from Fotoable Inc use the exactly same native library called *libncnn_style.0.2.so* to support DL technique. This is because that DL frameworks and even the DL models are easily reusable: a good nature of DL technique that can help reduce the engineering efforts of developers.

Implications: *The popularity of DL among top smartphone apps, especially ones developed by big companies, should endow smaller companies or independent developers with strong confidence in deploying DL in their apps.*

4.3 The roles of DL in apps

• **What are the popular uses of DL?** To understand the roles played by DL, we manually classify the usage of DL on different apps. This is achieved by looking into the app description and app contents. The results are shown in Table 1. Each app has one or more usages, and the usage is represented in two different levels (coarse and detailed). 10 apps are left out since we cannot confirm their DL usage.

Overall, image processing is the most popular usage of DL on smartphones, far more than text and audio processing (149 vs. 26 & 24). This is not surprising since computer vision is the field where DL starts the revolution [64], and the progress on this field has been lasting since then [72]. In more details, photo beauty (97) and face detection (52) are mostly widely used in DL apps, usually found in photo editing and camera apps to beautify pictures. In text field, word & emoji prediction (15) and auto-correct (10) are also

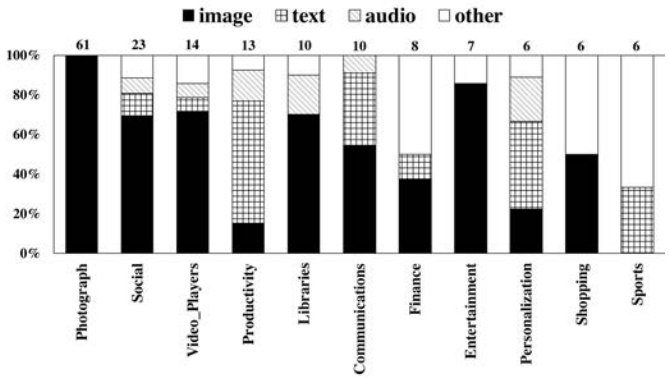


Figure 3: Distributions of DL apps over categories defined by Google Play. Numbers on top: the counts of DL apps in the corresponding categories. Apps in each category are further broken down by DL usage (see Table 1 for description). Categories with fewer than 5 DL apps are not shown.

popular, usually found in input method apps like GBoard. For audio processing, DL is mainly used for speech recognition (14). Besides, there are other types of usage such as recommendation (11) which is often found in shopping apps.

• **Which categories do DL apps come from?** Figure 3 summarizes the number of DL apps in different categories. As observed, almost 29% DL apps (61 out of 211) are in category photograph, all of which use DL for image processing. Social category is another hotspot with 23 DL apps in total, 78% of which use DL for image processing while others use it for text, audio, etc. The category of productivity also contains 13 DL apps, but most of them (62%) use DL for text processing. Overall, we can see that the DL usage is somehow diverse, with 11 categories has more than 5 DL apps among the top 500. Such diversity gives credits to the good generality of DL technique.

Implications: *Our findings encourage developers of certain types of apps, notably the ones with photo beauty, to embrace DL. Our findings also motivate encapsulating DL algorithms within higher level abstractions that cater to popular uses in apps. For instance, companies such as SenseTime already starts to ship DL-based face detection libraries to app developers. Masking the details of DL models and frameworks, such abstractions would make DL more friendly to developers.*

• **Is DL a core building block?** We also manually tag each DL usage as core feature or not. We define the DL functionality as apps' core feature if and only if two conditions are satisfied: (1) *hot*: the DL functionality is very likely to be invoked every time the apps are opened and used by users, (2) *essential*: without the DL functionality, the apps' main functionality will be severely compromised or even become infeasible. For example, DL on *text recognition* is treated as core feature in a scanner app (Adobe Scan) that helps users translate an image into text, but not in a payment app (Alipay) that uses it to scan ID card for identification. Similarly, DL on *photo beauty* is treated as core feature in a camera app (Meitu), but not in a social app (Facebook Messenger Kids).

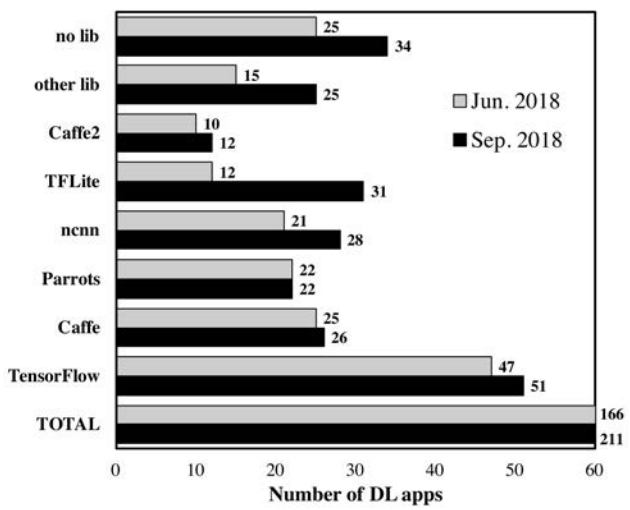


Figure 4: Numbers of DL apps using various mobile DL frameworks. "other lib": the DL apps developed on the frameworks in Table 2 but not itemized here, e.g. *mace, SNPE, and xnn*. "no lib": apps with DL functions but using no DL frameworks from Table 2. Note that the number in "TOTAL" is lower than the sum of others since some DL apps have integrated multiple frameworks.

Overall, 171 out of 211 (81%) apps use DL to support core features. Specifically, since photo beauty (96.9%) and face detection (84.6%) are primarily used in photo & camera apps, their usage is essential. Similarly, word & emoji prediction (100%) and auto-correct (100%) are treated as core features in keyboard apps, helping users input more efficiently and accurately. However, recommendation (18.1%) is often provided as complementary feature to others such as shopping apps, thus not treated as core feature.

Implications: *Our findings support future investment on R&D of mobile DL, as the core user experience on smartphones will probably depend on DL performance [67] and security [80, 92].*

4.4 DL frameworks

As mentioned in Section 2, DL frameworks are critical to DL adoption, as most developers use those frameworks to build their DL apps. In this subsection, we investigate into how those frameworks are used in DL apps: the numbers, the sizes, the practice, etc.

• **A glance over popular DL frameworks** We first make an investigation into popular DL frameworks, and the results are summarized in Table 2. We select those 21 frameworks for their popularity, e.g., forks and stars on GitHub, gained attention on StackOverflow and other Internet channels. Among those 21 frameworks, 16 frameworks support Android platform via Java (official language on Android) and/or C++ (native support via cross-compilation). Most of them are open-source, while others are either provided to public as a binary SDK (*SNPE, CoreML*), or only accessible by the providers' (collaborators') own products (*xNN, Parrots*). Most of them use customized format to store and represent the model files, but some



Figure 5: The number of check-in and check-out DL apps.

leverage existing approaches, such as ProtoBuf [28]. We also notice a trend on lightweight DL inference frameworks, which are designed specifically for mobile apps but have no training-support back-end (*ncnn, FeatherCNN, MACE*, etc). Those frameworks cannot train DL models, but can predict with pre-trained models via other frameworks such as *TensorFlow* or *Caffe*. Note that our later analysis substantially relies on the openness of DL frameworks: it enables us to use the existing tools to analyze or even visualize the DL models such as *TensorFlow*, or build our own interpreting scripts to analyze them based on the open code such as *ncnn*.

• **What is the adoption of DL frameworks?** As summarized in Figure 4, the most popular DL frameworks used in Sep. 2018 are *TensorFlow* (51), *TFLite* (31), and *ncnn* (28), as they contribute to almost 50% of the total number of DL apps. Other popular frameworks include *Caffe*, *Parrots*, and *Caffe2*. We have made several observations from those 6 dominant frameworks as following.

(1) All these frameworks are developed by big companies (e.g. Google), AI unicorns (e.g. SenseTime), or renowned universities (e.g. Berkeley).
(2) 5 out of these 6 frameworks are open-source, except *Parrots* which is provided as SDK to consumers. In fact, it is believed that openness is already an important feature in machine learning, especially DL society, as it supposes to [91]. It helps developers reproduce the state-of-the-art scientific algorithms, customize for personal usage, etc. As a result, for example, *TensorFlow* has more than 1,670 contributors up to Oct. 2018, going far beyond the community of Google.
(3) Most (4 out of 6) frameworks are optimized for smartphones, except *Caffe* and *TensorFlow*. Those mobile DL frameworks are designed and developed specifically for mobile devices, usually without training back-end, so that the resulted libraries can be faster and more lightweight. As an example, *TFLite* stems from *TensorFlow*, but is designed for edge devices and reported to have lower inference time and smaller library size than *TensorFlow* [106]. Besides those popular frameworks, we identify 34 (16.1%) DL apps that don't use any framework in Table 2. These apps use self-developed engines to support DL functionality.

• **Are mobile DL frameworks gaining traction?** As shown in Figure 4, DL frameworks optimized for smartphones, such as *TFLite* and *ncnn*, quickly gain popularity: the number of *TFLite*-based DL apps has increased from 12 to 31; that of *ncnn* increases from 21 to 28. We deem it as the trend of mobile DL ecosystem: *To train a model offline, use large, mature, and generic frameworks that focus on developer friendliness and feature completeness. To deploy a model on edge devices, switch to mobile-oriented frameworks that focus on performance (inference latency, memory footprint, and library size).*

Table 2: An overview of popular deep learning frameworks and their smartphone support at the time of writing (Nov. 2018).

Framework	Owner	Supported Mobile Platform	Mobile API	Is Open-source	Supported Model Format	Support Training
TensorFlow [36]	Google	Android CPU, iOS CPU	Java, C++	✓	ProtoBuf (*.pb, .pbtxt*)	✓
TF Lite [37]	Google	Android CPU, iOS CPU	Java, C++	✓	FlatBuffers (*.tflite*)	✗
Caffe [65]	Berkeley	Android CPU, iOS CPU	C++	✓	customized, json (*.caffemodel, .prototxt*)	✓
Caffe2 [9]	Facebook	Android CPU, iOS CPU	C++	✓	ProtoBuf (*.pb*)	✓
MxNet [46]	Apache Incubator	Android CPU, iOS CPU	C++	✓	customized, json (*.json, .params*)	✓
DeepLearning4J [13]	Skymind	Android CPU	Java	✓	customized (*.zip*)	✓
ncnn [35]	Tencent	Android CPU, iOS CPU	C++	✓	customized (*.params, .bin*)	✗
OpenCV [26]	OpenCV Team	Android CPU, iOS CPU	C++	✓	TesnorFlow, Caffe, etc	✗
FeatherCNN [16]	Tencent	Android CPU, iOS CPU	C++	✓	customized (*.feathermodel*)	✗
PaddlePaddle [24]	Baidu	Android CPU, iOS CPU & GPU	C++	✓	customized (*.tar*)	✓
xNN [40]	Alibaba	Android CPU, iOS CPU	unknown	✗	unknown	unknown
superid [34]	SuperID	Android CPU, iOS CPU	unknown	✗	unknown	unknown
Parrots [30]	SenseTime	Android CPU, iOS CPU	unknown	✗	unknown	unknown
MACE [23]	XiaoMi	Android CPU, GPU, DSP	C++	✓	customized (*.pb, .yml, .a*)	✗
SNPE [31]	Qualcomm	Qualcomm CPU, GPU, DSP	Java, C++	✗	customized (*.dlc*)	✗
CNNDroid [70]	Oskouei et al.	Android CPU & GPU	Java	✓	MessagePack (*.model*)	✗
CoreML [12]	Apple	iOS CPU, GPU	Swift, OC	✗	customized, ProtoBuf (*.proto, .mlmodel*)	✓
Chainer [10]	Preferred Networks	/	/	✓	customized (*.chainermodel*)	✓
CNTK [22]	Microsoft	/	/	✓	ProtoBuf (*.proto*)	✓
Torch [39]	Facebook	/	/	✓	customized (*.dat*)	✓
PyTorch [29]	Facebook	/	/	✓	customized, pickle (*.pkl*)	✓

We also investigate into the DL check-in and check-out behavior in mobile apps. We define the check-in DL apps as those that have no DL usage in earlier version (Jun. 2018) but add the DL usage in newer version (Sep. 2018), and the check-out vice versa. Note that the app list of our two datasets are not identical since we crawl the most popular ones, but the popularity is changing. So we only consider the apps that exist in both lists (11,710 in total), and conclude the results in Figure 5. As observed, 48 out of the 190 (25.3%) DL apps in newer version are checked in between Jun. 2018 and Sep. 2018. We also notice that some DL apps in old version checked out, but the number is much smaller (5). The reasons of check-out can be that the developers remove the corresponding functionality or just replace the DL with other approaches. Overall, the statistics support the fact that DL technique is increasingly adopted in mobile apps.

• **What is the storage overhead of frameworks?** Figure 6 shows the sizes of DL libs, i.e. the physical incarnation of DL frameworks. As shown, the average size of DL libs is 7.5MB, almost 6 times compared to the non-DL libs. Here, we only use the non-DL libs found within DL apps. The results show that DL libs are commonly heavier than non-DL libs, because implementing DL functionality, even without training backend, is quite complex. Looking into different frameworks, using *TensorFlow* and *Caffe* results in larger DL libs, i.e., 15.3MB and 10.1MB respectively, while others are all lower than 5MB. The reason is that mobile supports of *TensorFlow* and *Caffe* are ported from the original frameworks and substantially reuse the code base from them. However, these two frameworks are designed for distributed on-cloud DL. As comparison, other frameworks in Figure 6 are specifically designed for mobile devices to the purpose of good performance.

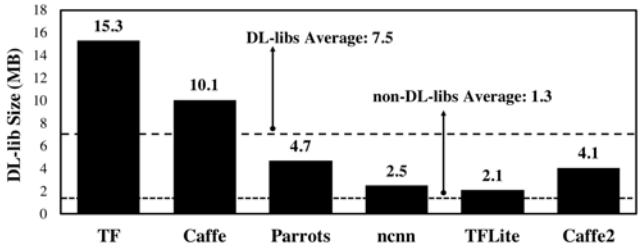


Figure 6: The binary library sizes of DL frameworks.

One app may incorporate multiple DL frameworks. Surprisingly, we find that 24 DL apps integrate more than one DL framework. For example, AliPay, the most popular payment app in China, has both *xnn* and *ncnn* inside. We deem such multi-usage as (potentially) bad practice, since it unnecessarily increases the apk size and memory footprint when these frameworks need to be loaded simultaneously. According to our statistics, the overhead is around 5.4MB, contributing to 13.6% to the total apk size on average. Such overhead can be avoided by running different tasks based on one framework, since most DL frameworks are quite generic and can support various types of DL models. Even if not, they can be easily extended to support the missing features [1]. The reason of such multi-usage behavior can be twofold. First, one app might be

developed by different engineers (groups), who introduce different frameworks for their own DL purpose. Second, the developers may just reuse existing code and models for specific tasks, without merging them together in one DL implementation.

Implications: *Our findings highlight the advantages and popularity of mobile DL frameworks, encouraging further optimizations on them. Our findings also motivate app developers to give these frameworks priority considerations in choosing the incarnation of DL algorithms.*

5 MODEL ANALYSIS

This section focuses on the model-level analysis of DL technique. We first describe the methodology details, e.g., the design of Model Extractor in Section 5.1. Then, we show the analysis results on those DL models from three main aspects.

- The structures of DL models: the model types, layer types, and optimizations used (Section 5.2).
- The resource footprint of DL models: storage, memory, execution complexity, etc (Section 5.3).
- The security of DL models: using obfuscation and encryption to protect models from being stolen (Section 5.4).

5.1 Model Extractor: finding DL models

As a part of our analyzing tool (Section 3.2), Model Extractor takes DL apps which we have already identified as input, and outputs the DL model(s) used in each app. Model Extractor scans the *assets* folder of each decomposed DL apps, tries to validate each model file inside. Since DL frameworks use different formats to store their model files, Model Extractor has a validator for each of supported framework. However, we observe that many models are not stored as plaintext inside apk files. For example, some of them are encrypted on storage, and decrypted when apps running on devices. For such cases, Model Extractor tries to reverse engineer the apps, and extract the analyzable models.

Overall results We extract DL models based on the most popular frameworks, i.e., *TFLite*, *TensorFlow*, *ncnn*, *Caffe*, or *Caffe2*. In summary, we successfully extract 176 DL models, which come from 71 DL apps. The reasons why we cannot extract models from the other DL apps could be (i) the models are well protected and hidden in the apk files; (ii) the models are retrieved from Internet during runtime. Among the extracted models, we can analyze 98 of them, which come from 42 DL apps. The other extracted models cannot be parsed via our framework currently because (i) the models are in format which we have no insights into, such as *Parrots*-based models, since the corresponding frameworks are not open-source; (ii) the models are encrypted.

5.2 Model Structures

• **DL model types** Among the DL models extracted, 87.7% models are CNN models, 7.8% models are RNN models, while others are not confirmed yet. The CNN models are mostly used in image/video processing and text classification. The RNN models are mostly used in text/voice processing, such as word prediction, translation, speech recognition, etc. The results are consistent with the conventional wisdom: CNN models are good at capturing visual characteristics from images, while RNN models are powerful at processing sequential data with temporal relationships such as text and audio.

Table 3: Layers used in DL models. "% of models" shows how many models contain such layer, while "# in each model" shows the median/mean numbers of occurrences in each model that contains such layer. "conv" and "fc" are short for convolution and fully-connect.

Layer type	% of models	# in each model	Layer type	% of models	# in each model
conv	87.7	5 / 14.8	relu	51.0	6 / 16.3
pooling	76.5	2 / 2.8	split	46.9	1 / 7.5
softmax	69.1	1 / 1.1	prelu	32.1	4 / 4.6
fc	60.5	3 / 5.6	reshape	28.4	2 / 24.1
add	56.8	9.5 / 23.8	dropout	21.0	1 / 1.0

Table 4: Optimizations applied on DL models.

	1-bit Quan.	8-bit Quan.	16-bit Quan.	Sparsity
TF	unsupported	4.78%	0.00%	0.00%
TFLite	unsupported	66.67%	unsupported	unsupported
Caffe	unsupported	0.00%	unsupported	unsupported
Caffe2	unsupported	0.00%	unsupported	unsupported
ncnn	unsupported	0.00%	unsupported	unsupported
Total	*0.00%*	*6.32%*	*0.00%*	*0.00%*

• **DL layer types** We then characterize different types of layers used in DL models. As shown in Table 3, convolution (conv) is the most commonly used type. 87.7% models have at least one convolutional layer, and the median (mean) number of convolutional layers used in those models is 5 (14.8). This is not surprising since convolution is the core of CNN models, and CNN is the dominant model architecture used in vision tasks. Our previous analysis already demonstrates that image processing is the most popular use case of mobile DL. Similarly, pooling is also an important layer type in CNN models, included in 76.5% DL models. Besides, softmax is also frequently used (69.1%), but don't repeatedly show up in one single model. This is because softmax layer usually resides at the end of DL models to get the probabilities as output. As a comparison, fully-connected (fc) layers are less common, only used in 60.5% DL models. A possible reason is that fully-connected layer is known to be very parameter- and computation-intensive, and can be replaced by other layers such as convolution [17]. Other frequently used layer types include add, split, relu, prelu, dropout, and reshape.

Implications: *Our findings motivate framework and hardware vendors who are interested in optimizing mobile DL to focus on the popular DL layers that we discovered in deployed models, e.g. convolution.*

We also notice that a small number (5) of DL models contain customized layer types. Such customization is made as an extension to existing DL frameworks [1]. The result indicates that the functionalities of current DL frameworks are mostly complete enough.

• **Model optimizations** Various techniques have been proposed to optimize the DL models in consideration of their sizes and computation complexity. Here, we study the usage of two most popular techniques in the wild: quantization and sparsity. **Quantization** compresses DL models by reducing the number of bits required

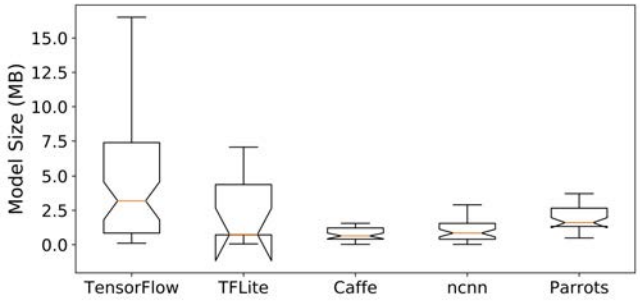


Figure 7: The size of DL models in different frameworks. We leave out *Caffe2* since we only successfully extract one model in *Caffe2* format.

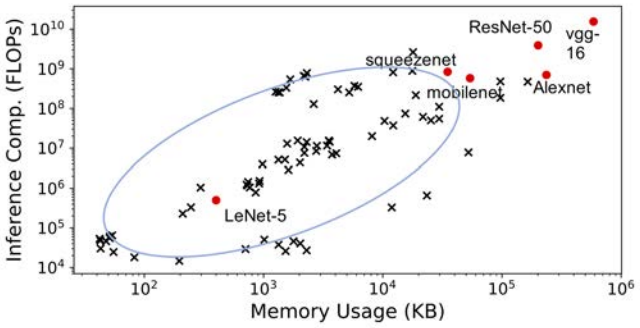


Figure 8: The cost of memory and computation of DL models extracted from apps. Red dots: classical CNN architectures as references. Black crosses: extracted DL models.

to represent each weight. The quantization has different levels, including 16-bit [56], 8-bit [95], and even 1-bit [48, 84], while the original models are usually presented in 32-bit floating points. **Sparsity** [71, 96] has also been extensively studied in prior literatures as an effective approach to make compact DL models. It's mainly used in matrix multiplication and convolutional layers to reduce parameters. Such optimizations are known to reduce DL cost by up to two orders of magnitude without compromising model accuracy [58].

Table 4 summarizes the optimizations applied on DL models. Here we focus on the DL models for 5 popular frameworks, i.e., *TensorFlow (TF), TFLite, Caffe, Caffe2*, and *ncnn*. Overall, most of these frameworks only support 8-bit quantization, except *TensorFlow* who has 16-bit quantization and sparsity support. However, only a fraction of DL models apply the optimization techniques: 6.32% models are quantized into 8-bit, while others are non-optimized.

Implications: *The findings that well-known DL optimizations are missing in real-world deployment suggest the efficiency potential of mobile DL is still largely untapped. The findings also urge immediate actions to fix the missing optimizations.*

5.3 Model Resource Footprint

• **Model size.** Figure 7 illustrates the size of DL models (in storage). Overall, we find that the extracted DL models are quite small (median: 1.6MB, mean: 2.5MB), compared to classical models such as VGG-16 [90] (around 500MB) and MobileNet [63] (around 16MB). The models in *TensorFLow* format (median: 3.2MB) are relatively larger than the models in other formats such as *TFLite* (median: 0.75MB) and *ncnn* (median: 0.86MB).

• **Runtime overhead.** We then study the runtime performance of DL models. Here, we focus on two aspects: memory and computations. The memory usage includes both the model parameters and the generated intermediate results (feature maps). For computation complexity, we use floating point operations (FLOPs) during one inference as the metric. Here we use only part of models in *TensorFlow* and *ncnn* formats since some others don't have fixed input sizes, e.g., image size, so that the computation complexity can only be determined at runtime [15]. We also include the performance of some other classical CNN models such as AlexNet, MobileNet, etc.

As illustrated in Figure 8, the black crosses represent the DL models we have extracted, while the red dots represent the classical CNN architectures. Overall, the results show that *in-the-wild DL models are very lightweight in consideration of memory usage and computation complexity, with median value of 2.47 MB and 10M FLOPs respectively.* Running such models on mobile processors is inexpensive. For example, as estimated on the CPU of Snapdragon 845[3], the execution time of 80% models are less than 15ms which is translated to 67 FPS [32]. To be compared, ResNet-50, one of the state-of-the-art models in image classification task, has around 200MB memory usage and 4GFLOPs computations. Even MobileNet and SqueezeNet, which are designed and optimized for mobile scenarios, require more memory usage and computations than 90% those mobile DL models that we have discovered.

Implications: *Our findings of dominant lightweight DL models on smartphones give app developers a valuable assurance: DL inference can be as cheap as a few MBs of memory overhead and tens of ms execution delay. Our findings challenge existing research on DL inference, which are typically centered on full-blown models (e.g. VGG) and validated on these models. Given the significance of smartphones as DL platforms, future DL algorithm proposals should consider applicability on lightweight models and resource constraints concern.*

5.4 Model Security

Finally, we investigate into how DL models are protected. We deem model protection as an important step to AI system/app security, because if attackers can acquire the model, they can (i) steal the intellectual property by reusing the model file or re-train a new model; (ii) easily attack the DL functionality via adversarial attack [80]. We focus on two practical protection mechanisms.

- **Obfuscation** is a rather shallow approach to prevent attackers from gaining insights into the model structures by removing any meaningful text, e.g., layer names.
- **Encryption** is better in security by avoiding attackers from getting the model structures/parameters, but also causes inevitable overhead for apps to decrypt the models in memory. Here, we deem encrypted models as always obfuscated too.

We investigate into how obfuscation and encryption are employed on DL models that we have extracted. We analyze the DL models

[3] A typical mobile chip used by many popular smartphones such as Galaxy S8.

extracted from apps using *TensorFLow, TFLite, ncnn, caffe*, and *Caffe2*. In total, we confirm the security level of 120 DL models. Note that here encryption doesn't necessarily mean encryption algorithm, but also includes cases where developers customize the model format so that the model cannot be parsed via the DL framework.

The results show that among the 120 DL models, we find 47 (39.2%) models are obfuscated and 23 (19.2%) models are encrypted. Note that these two sets of apps are overlapped: encrypted apps are also obfuscated. The results indicate that *most DL models are exposed without protection, thus can be easily extracted and utilized by attackers.* In fact, only few frameworks in Table 2 support obfuscation, e.g., *ncnn* can convert models into binaries where text is all striped [33], and *Mace* can convert a model to C++ code [11]. What's worse, no framework provides help in model encryption as far as we know. Thus, developers have to implement their own encryption/decryption mechanism, which can impose non-trivial programming overhead.

Implications: *The grim situation of model security urges strong protection over proprietary models in a way similar to protecting copyright digital contents on smartphones [41, 78, 87, 104]. This necessitates a synergy among new tools, OS mechanisms and policies, and hardware mechanisms such as ARM TrustZone [7].*

6 DISCUSSIONS

Limitations of our analyzing tool Although we carefully design our analyzer to capture as many DL apps as possible, and involve a lot of manual efforts to validate the results, we can still have false identifications. For example, those DL apps that neither depend on any popular DL frameworks, nor have any string patterns in the native libraries, will be missed out. In addition, the apps that have integrated DL frameworks but don't really use them will be falsely classified as DL apps. For the first case, we plan to mine the code pattern of DL implementation and use the pattern to predict more DL apps that might involve DL. For the second one, we plan to further enhance our analyzer with advanced static analysis technique [42] so that it can detect whether the API calls (sinks) of DL libraries will be invoked or not. Our tool can be further enhanced via dynamic analysis, e.g., running the extracted models on smartphones and inspecting the system behavior.

More platforms In this work, we only analyze the adoption of DL on Android apps. Though Android is quite representative of the mobile ecosystem, more interesting findings might be made by expanding our study on other platforms such as iOS and Android Wear. We believe that comparing the DL adoption on different platforms can feed in more implications to researchers and developers.

7 RELATED WORK

Mobile DL Due to their ubiquitous nature, mobile devices can create countless opportunities for DL tasks. Researchers have built numerous novel applications based on DL [44, 68, 69, 76, 83, 98]. Besides, various optimization techniques have been proposed to reduce the overhead of DL on resource-constrained mobile devices, e.g., model compression [50, 59, 67, 74, 97], hardware customizations [45, 47, 57, 103], and cloud offloading [61, 66, 99, 105]. These studies are usually carried out under lab environments, based on classical models such as VGG and ResNet, in lack of real-world workloads and insights. Thus, our work is motivated by those enormous efforts that try to bring DL to mobile devices, and fill the gap between the academic literature and industry products.

ML/DL as cloud services Besides on-device fashion, the DL functionality, or in a broader scope of Machine Learning (ML), can also be accessed as cloud services. The service providers include Amazon [2], Google [19], Microsoft Azure [21], etc. There are some prior analyzing studies focusing on such MLaaS (machine learning as a service) platforms. Yao *et al.* [101] comprehensively investigate into effectiveness of popular MLaas systems, and find that with more user control comes greater risk. Some other literature studies [52, 88, 94] focus on the security issues of those platforms towards different types of attacks.

Empirical study of DL Prior empirical analysis mainly focuses on assisting developers to build better DL apps/systems/models. Zhang *et al.* [107] characterize the defects (bugs) in DL programs via mining the StackOverflow QA pages and GitHub projects. Consequently, the results are limited in only open-source, small-scale projects. Fawzi *et al.* [51] analyze the topology and geometry of the state-of-the-art deep networks, as well as their associated decision boundary. These studies mostly focus on classical and small-scale DL models proposed in previous literature, while our study mine the knowledge from large-scale in-the-wild mobile apps.

DL Model protection Some recent efforts have been investigated in protecting DL models. For example, various watermarking mechanisms [41, 78, 87, 104] have been proposed to protect intellectual property of DL models. This approach, however, cannot protect models from being extracted and attacked. As a closer step, some researchers [55, 60, 93] secure the DL systems/models based on secure execution environments (SEE) such as Intel SGX [20]. Some DL frameworks also provide mechanisms for model protection. For example, *Mace* [23] supports developers to convert models to C++ code [11]. However, our results show that a large number of DL models are exposed without secure protection.

8 CONCLUSIONS

In this work, we have carried out the first empirical study to understand how deep learning technique is adopted in real-world smartphones, as a bridge between the research and practice. By mining and analyzing large-scale Android apps based on a static tool, we have reached interesting and valuable findings. Our findings also provide strong and valuable implications to multiple stakeholders of the mobile ecosystem, including developers, hardware designers, and researchers.

ACKNOWLEDGMENT

The authors affiliated with Peking University were supported by the National Key R&D Program of China under the grant number 2018YFB1004800, the Beijing Municipal Science and Technology Project under the grant number Z171100005117002, the National Natural Science Foundation of China under grant numbers 61725201, 61528201, and 61529201. Felix Xiaozhu Lin was support in part by a Google faculty award.

REFERENCES

[1] 2018. Add a new op in TensorFlow. https://www.tensorflow.org/guide/extend/op.

[2] 2018. Amazon Machine Learning. https://aws.amazon.com/machine-learning.
[3] 2018. An Exploration of Mobile First AI. https://medium.com/swlh/an-exploration-of-mobile-first-ai-576c944efd36.
[4] 2018. Android aapt. http://elinux.org/Android_aapt.
[5] 2018. Apktool: A tool for reverse engineering Android apk files. https://ibotpeaches.github.io/Apktool/.
[6] 2018. Apple COO: Smartphone is a 'major platform' for future of AI. https://www.techrepublic.com/article/apple-coo-smartphone-is-a-major-platform-for-future-of-ai/.
[7] 2018. Arm TrustZone. https://developer.arm.com/technologies/trustzone.
[8] 2018. Artificial Intelligence Next Key Growth Area for Smartphones as Numbers Top Six Billion by 2020, IHS Markit Says. https://news.ihsmarkit.com/press-release/technology/artificial-intelligence-next-key-growth-area-smartphones-numbers-top-six-bi.
[9] 2018. Caffe2 deep learning framework. https://github.com/caffe2/caffe2.
[10] 2018. Chainer. https://chainer.org/.
[11] 2018. Converting model to C++ code. https://mace.readthedocs.io/en/latest/user_guide/advanced_usage.html.
[12] 2018. CoreML by Apple. https://developer.apple.com/documentation/coreml.
[13] 2018. Deep Learning for Java. https://deeplearning4j.org/.
[14] 2018. dex2jar. https://github.com/pxb1988/dex2jar.
[15] 2018. Dynamic shapes in TensorFlow. https://www.tensorflow.org/guide/tensors.
[16] 2018. FeatherCNN. https://github.com/Tencent/FeatherCNN.
[17] 2018. Fully-connected Layers in Convolutional Neural Networks). https://cs231n.github.io/convolutional-networks/.
[18] 2018. Global mobile OS market share in sales to end users. https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/.
[19] 2018. Google Prediction API. https://cloud.google.com/prediction.
[20] 2018. Intel Software Guard Extensions. https://software.intel.com/en-us/sgx.
[21] 2018. Microsoft Azure ML Studio. https://azure.microsoft.com/en-us/services/machine-learning.
[22] 2018. Microsoft Cognitive Toolkit (CNTK). https://github.com/Microsoft/CNTK.
[23] 2018. Mobile AI Compute Engine. https://github.com/XiaoMi/mace.
[24] 2018. Mobile deep learning. https://github.com/baidu/mobile-deep-learning.
[25] 2018. Open neural network exchange format. https://onnx.ai/.
[26] 2018. Open Source Computer Vision Library. https://opencv.org/.
[27] 2018. Over Half of Smartphone Owners Use Voice Assistants. https://voicebot.ai/2018/04/03/over-half-of-smartphone-owners-use-voice-assistants-siri-leads-the-pack/.
[28] 2018. Protocol Buffer. https://developers.google.com/protocol-buffers/.
[29] 2018. pytorch. http://pytorch.org/.
[30] 2018. SenseTime. https://www.sensetime.com/?lang=en-us.
[31] 2018. Snapdragon Neural Processing Engine. https://developer.qualcomm.com/software/snapdragon-neural-processing-engine.
[32] 2018. Snapdragon performance. https://www.anandtech.com/show/12420/snapdragon-845-performance-preview/2.
[33] 2018. Strip visible string in ncnn. https://github.com/Tencent/ncnn/wiki/how-to-use-ncnn-with-alexnet.
[34] 2018. SuperID Android SDK. https://github.com/SuperID/superid-android-sdk.
[35] 2018. Tencent ncnn deep learning framework. https://github.com/Tencent/ncnn.
[36] 2018. TensorFlow. https://www.tensorflow.org/.
[37] 2018. TensorFlow Lite. https://www.tensorflow.org/mobile/tflite/.
[38] 2018. The Machine Learning Behind Android Smart Linkify. https://ai.googleblog.com/2018/08/the-machine-learning-behind-android.html.
[39] 2018. torch. http://torch.ch/.
[40] 2018. xNN deep learning framework. https://myrgzn.gitee.io/rgzn/news/page100.html.
[41] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. 2018. Turning Your Weakness Into a Strength: Watermarking Deep Neural Networks by Backdooring. *arXiv preprint arXiv:1802.04633* (2018).
[42] Steven Arzt, Siegfried Rasthofer, Christian Fritz, Eric Bodden, Alexandre Bartel, Jacques Klein, Yves Le Traon, Damien Octeau, and Patrick McDaniel. 2014. Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps. *Acm Sigplan Notices* 49, 6 (2014), 259–269.
[43] Davide Bacciu, Stefano Chessa, Claudio Gallicchio, and Alessio Micheli. 2017. On the need of machine learning as a service for the internet of things. In *Proceedings of the 1st International Conference on Internet of Things and Machine Learning, IML 2017*. 22:1–22:8.
[44] Michael Barz and Daniel Sonntag. 2016. Gaze-guided Object Classification Using Deep Neural Networks for Attention-based Computing. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp'16)*. 253–256.
[45] Tianshi Chen, Zidong Du, Ninghui Sun, Jia Wang, Chengyong Wu, Yunji Chen, and Olivier Temam. 2014. DianNao: a Small-footprint High-throughput Accelerator for Ubiquitous Machine-Learning. In *Architectural Support for Programming Languages and Operating Systems (ASPLOS'14)*. 269–284.
[46] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. 2015. MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. *CoRR* abs/1512.01274 (2015).
[47] Yu-Hsin Chen, Joel S. Emer, and Vivienne Sze. 2016. Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks. In *43rd ACM/IEEE Annual International Symposium on Computer Architecture, (ISCA'16)*. 367–379.
[48] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. 2015. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in neural information processing systems*. 3123–3131.
[49] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 191–198.
[50] Emily L. Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. 2014. Exploiting Linear Structure Within Convolutional Networks for Efficient Evaluation. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems (NIPS'14)*. 1269–1277.
[51] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. 2018. Empirical study of the topology and geometry of deep networks. In *IEEE CVPR*.
[52] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 1322–1333.
[53] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.
[54] Mihajlo Grbovic and Haibin Cheng. 2018. Real-time Personalization using Embeddings for Search Ranking at Airbnb. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 311–320.
[55] Zhongshu Gu, Heqing Huang, Jialong Zhang, Dong Su, Ankita Lamba, Dimitrios Pendarakis, and Ian Molloy. 2018. Securing Input Data of Deep Learning Inference Systems via Partitioned Enclave Execution. *arXiv preprint arXiv:1807.00969* (2018).
[56] Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. 2015. Deep learning with limited numerical precision. In *International Conference on Machine Learning*. 1737–1746.
[57] Song Han, Xingyu Liu, Huizi Mao, Jing Pu, Ardavan Pedram, Mark A. Horowitz, and William J. Dally. 2016. EIE: Efficient Inference Engine on Compressed Deep Neural Network. In *43rd ACM/IEEE Annual International Symposium on Computer Architecture, (ISCA'16)*. 243–254.
[58] Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149* (2015).
[59] Seungyeop Han, Haichen Shen, Matthai Philipose, Sharad Agarwal, Alec Wolman, and Arvind Krishnamurthy. 2016. MCDNN: An Approximation-Based Execution Framework for Deep Stream Processing Under Resource Constraints. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys'16)*. 123–136.
[60] Lucjan Hanzlik, Yang Zhang, Kathrin Grosse, Ahmed Salem, Max Augustin, Michael Backes, and Mario Fritz. 2018. MLCapsule: Guarded Offline Deployment of Machine Learning as a Service. *arXiv preprint arXiv:1808.00590* (2018).
[61] Johann Hauswald, Yiping Kang, Michael A. Laurenzano, Quan Chen, Cheng Li, Trevor N. Mudge, Ronald G. Dreslinski, Jason Mars, and Lingjia Tang. 2015. DjiNN and Tonic: DNN as a service and its implications for future warehouse scale computers. In *Proceedings of the 42nd Annual International Symposium on Computer Architecture, Portland, OR, USA, June 13-17, 2015*. 27–40.
[62] Ehsan Hesamifard, Hassan Takabi, Mehdi Ghasemi, and Rebecca N. Wright. 2018. Privacy-preserving Machine Learning as a Service. *PoPETs* 2018, 3 (2018), 123–142.
[63] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *CoRR* abs/1704.04861 (2017).
[64] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. 2016. SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and <1MB Model Size. *arXiv preprint arXiv:1602.07360* (2016).
[65] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross B. Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. In *Proceedings of the ACM International Conference on Multimedia, MM '14, Orlando, FL, USA, November 03 - 07, 2014*. 675–678.
[66] Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor N. Mudge, Jason Mars, and Lingjia Tang. 2017. Neurosurgeon: Collaborative Intelligence

Between the Cloud and Mobile Edge. In *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'17)*. 615–629.

[67] Nicholas D. Lane, Sourav Bhattacharya, Petko Georgiev, Claudio Forlivesi, Lei Jiao, Lorena Qendro, and Fahim Kawsar. 2016. DeepX: A Software Accelerator for Low-power Deep Learning Inference on Mobile Devices. In *15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN 2016)*. 23:1–23:12.

[68] Nicholas D. Lane, Sourav Bhattacharya, Petko Georgiev, Claudio Forlivesi, and Fahim Kawsar. 2015. An Early Resource Characterization of Deep Learning on Wearables, Smartphones and Internet-of-Things Devices. In *Proceedings of the 2015 International Workshop on Internet of Things towards Applications (IoT-App'15)*. 7–12.

[69] Nicholas D. Lane, Petko Georgiev, and Lorena Qendro. 2015. DeepEar: Robust Smartphone Audio Sensing in Unconstrained Acoustic Environments Using Deep Learning. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp'15)*. 283–294.

[70] Seyyed Salar Latifi Oskouei, Hossein Golestani, Matin Hashemi, and Soheil Ghiasi. 2016. CNNdroid: GPU-Accelerated Execution of Trained Deep Convolutional Neural Networks on Android. In *Proceedings of the 2016 ACM on Multimedia Conference*. 1201–1205.

[71] Vadim Lebedev and Victor Lempitsky. 2016. Fast convnets using group-wise brain damage. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2554–2564.

[72] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[73] Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J Smola. 2014. Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th international conference on Knowledge discovery and data mining (KDD'14)*. 661–670.

[74] Sicong Liu, Yingyan Lin, Zimu Zhou, Kaiming Nan, Hui Liu, and Junzhao Du. 2018. On-Demand Deep Model Compression for Mobile Devices: A Usage-Driven Model Selection Framework. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys'18)*. 389–400.

[75] Xuanzhe Liu, Huoran Li, Xuan Lu, Tao Xie, Qiaozhu Mei, Feng Feng, and Hong Mei. 2018. Understanding Diverse Usage Patterns from Large-Scale Appstore-Service Profiles. *IEEE Trans. Software Eng.* 44, 4 (2018), 384–411.

[76] Gaurav Mittal, Kaushal B. Yagnik, Mohit Garg, and Narayanan C. Krishnan. 2016. SpotGarbage: Smartphone App to Detect Garbage Using Deep Learning. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp'16)*. 940–945.

[77] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[78] Yuki Nagai, Yusuke Uchida, Shigeyuki Sakazawa, and ShinâĂŹichi Satoh. 2018. Digital watermarking for deep neural networks. *International Journal of Multimedia Information Retrieval* 7, 1 (2018), 3–16.

[79] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based news recommendation for millions of users. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1933–1942.

[80] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*. 372–387.

[81] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'14)*. 701–710.

[82] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[83] Valentin Radu, Nicholas D. Lane, Sourav Bhattacharya, Cecilia Mascolo, Mahesh K. Marina, and Fahim Kawsar. 2016. Towards Multimodal Deep Learning for Activity Recognition on Mobile Devices. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp'16)*. 185–188.

[84] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*. 525–542.

[85] Mauro Ribeiro, Katarina Grolinger, and Miriam A. M. Capretz. 2015. MLaaS: Machine Learning as a Service. In *14th IEEE International Conference on Machine Learning and Applications, ICMLA 2015*. 896–902.

[86] Everett M Rogers. 2010. *Diffusion of innovations*. Simon and Schuster.

[87] Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. [n. d.]. DeepSigns: A Generic Watermarking Framework for Protecting the Ownership of Deep Learning Models. ([n. d.]).

[88] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *Security and Privacy (SP), 2017 IEEE Symposium on*. 3–18.

[89] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of Go without human knowledge. *Nature* 550, 7676 (2017), 354.

[90] Karen Simonyan and Andrew Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* abs/1409.1556 (2014).

[91] SÃķren Sonnenburg, Mikio L Braun, Cheng Soon Ong, Samy Bengio, Leon Bottou, Geoffrey Holmes, Yann LeCun, Klaus-Robert MÃžller, Fernando Pereira, Carl Edward Rasmussen, et al. 2007. The need for open source software in machine learning. *Journal of Machine Learning Research (JMLR)* 8, Oct (2007), 2443–2466.

[92] Ion Stoica, Dawn Song, Raluca Ada Popa, David Patterson, Michael W Mahoney, Randy Katz, Anthony D Joseph, Michael Jordan, Joseph M Hellerstein, Joseph E Gonzalez, et al. 2017. A berkeley view of systems challenges for ai. *arXiv preprint arXiv:1712.05855* (2017).

[93] Florian Tramer and Dan Boneh. 2018. Slalom: Fast, Verifiable and Private Execution of Neural Networks in Trusted Hardware. *arXiv preprint arXiv:1806.03287* (2018).

[94] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. 2016. Stealing Machine Learning Models via Prediction APIs.. In *USENIX Security Symposium*. 601–618.

[95] Vincent Vanhoucke, Andrew Senior, and Mark Z Mao. 2011. Improving the speed of neural networks on CPUs. In *Proc. Deep Learning and Unsupervised Feature Learning NIPS Workshop*, Vol. 1. 4.

[96] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*. 2074–2082.

[97] Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Jian Cheng. 2016. Quantized Convolutional Neural Networks for Mobile Devices. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, (CVPR'16)*. 4820–4828.

[98] Mengwei Xu, Feng Qian, Qiaozhu Mei, Kang Huang, and Xuanzhe Liu. 2018. DeepType: On-Device Deep Learning for Input Personalization Service with Minimal Privacy Concern. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 4 (2018), 197.

[99] Mengwei Xu, Feng Qian, Mengze Zhu, Feifan Huang, Saumay Pushp, and Xuanzhe Liu. 2019. DeepWear: Adaptive Local Offloading for On-Wearable Deep Learning. *IEEE Transactions on Mobile Computing* (2019).

[100] Mengwei Xu, Mengze Zhu, Yunxin Liu, Felix Xiaozhu Lin, and Xuanzhe Liu. 2018. DeepCache: Principled Cache for Mobile Deep Vision. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. 129–144.

[101] Yuanshun Yao, Zhujun Xiao, Bolun Wang, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. 2017. Complexity vs. performance: empirical analysis of machine learning as a service. In *Proceedings of the 2017 Internet Measurement Conference, IMC 2017, London, United Kingdom, November 1-3, 2017*. 384–397.

[102] Xiao Zeng, Kai Cao, and Mi Zhang. 2017. *MobileDeepPill*: A Small-Footprint Mobile Deep Learning System for Recognizing Unconstrained Pill Images. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys'17)*. 56–67.

[103] Chen Zhang, Peng Li, Guangyu Sun, Yijin Guan, Bingjun Xiao, and Jason Cong. 2015. Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks. In *Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA'15)*. 161–170.

[104] Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph Stoecklin, Heqing Huang, and Ian Molloy. 2018. Protecting Intellectual Property of Deep Neural Networks with Watermarking. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*. 159–172.

[105] Qingchen Zhang, Laurence T. Yang, and Zhikui Chen. 2016. Privacy Preserving Deep Computation Model on Cloud for Big Data Feature Learning. *IEEE Trans. Computers* (2016), 1351–1362.

[106] Xingzhou Zhang, Yifan Wang, and Weisong Shi. 2018. pCAMP: Performance Comparison of Machine Learning Packages on the Edges. (2018).

[107] Yuhao Zhang, Yifan Chen, Shing-Chi Cheung, Yingfei Xiong, and Lu Zhang. 2018. An Empirical Study on TensorFlow Program Bugs. (2018).

Constrained Local Graph Clustering by Colored Random Walk

Yaowei Yan
yxy230@ist.psu.edu
Pennsylvania State University

Yuchen Bian
yub31@ist.psu.edu
Pennsylvania State University

Dongsheng Luo
dul262@ist.psu.edu
Pennsylvania State University

Dongwon Lee
dlee@ist.psu.edu
Pennsylvania State University

Xiang Zhang
xzhang@ist.psu.edu
Pennsylvania State University

ABSTRACT

Detecting local graph clusters is an important problem in big graph analysis. Given seed nodes in a graph, local clustering aims at finding subgraphs around the seed nodes, which consist of nodes highly relevant to the seed nodes. However, existing local clustering methods either allow only a single seed node, or assume all seed nodes are from the same cluster, which is not true in many real applications. Moreover, the assumption that all seed nodes are in a single cluster fails to use the crucial information of relations between seed nodes. In this paper, we propose a method to take advantage of such relationship. With prior knowledge of the community membership of the seed nodes, the method labels seed nodes in the same (different) community by the same (different) color. To further use this information, we introduce a color-based random walk mechanism, where colors are propagated from the seed nodes to every node in the graph. By the interaction of identical and distinct colors, we can enclose the supervision of seed nodes into the random walk process. We also propose a heuristic strategy to speed up the algorithm by more than 2 orders of magnitude. Experimental evaluations reveal that our clustering method outperforms state-of-the-art approaches by a large margin.

CCS CONCEPTS

• **Mathematics of computing** → **Graph algorithms**; • **Information systems** → **Clustering**; • **Theory of computation** → **Random walks and Markov chains**.

KEYWORDS

community detection; local graph clustering; random walk; non-Markovian

ACM Reference Format:
Yaowei Yan, Yuchen Bian, Dongsheng Luo, Dongwon Lee, and Xiang Zhang. 2019. Constrained Local Graph Clustering by Colored Random Walk. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313719

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313719

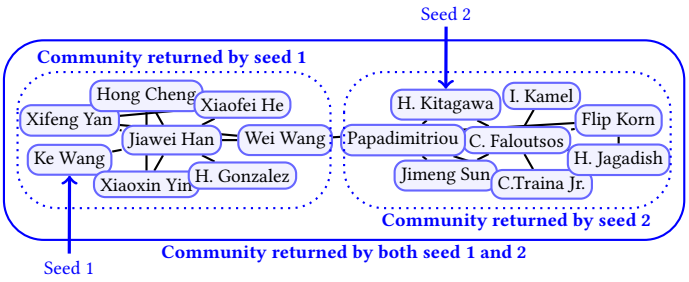


Figure 1: Example of an academic social network. All nodes are data science researchers and form two subgroups by their collaborations. Query from each side returns a single subgroup (in dotted boxes). The joint query returns two subgroups together (in the solid box).

1 INTRODUCTION

The graph/network is a natural representation of real-world relations. Graph clustering (or community detection) is a fundamental problem and is the base of many applications, such as online recommendation, medical diagnosis, social network and biological network analysis [11, 14, 15, 18]. The clustering of a graph aims to find groups of nodes that are closely related to each other in each group. Unlike global graph clustering which retrieves all clusters/communities from a graph, local graph clustering focuses on the subgraph within the neighborhood of the seed/query nodes and is applicable to very large data [17]. With the increasing size of networks, recently local clustering has attracted lots of research interest [6, 8, 9, 22].

The seed nodes of local graph clustering can be single or multiple. Existing local graph clustering methods do not accept multiple seed nodes (e.g., Random Walk with Restart [19]), or simply assume that all these seeds are from the same cluster (e.g., Personalized PageRank [1]). We argue that knowing the relations of seed nodes (in the same community or not) are important, as the network data may be noisy or lossy to be clustered accurately. Furthermore, it is a relatively easy task to label the seed nodes by users or domain experts, since the number of seed nodes is small.

Knowing seeds nodes are in the same community can help to merge divided subgroups to a whole community. Fig. 1 is part of a collaboration network used by a conference program committee (PC) member recommendation system. Given that Dr. Ke Wang and Dr. Kitagawa are prospectively favorable PC members, we want the system to recommend more researchers as candidates.

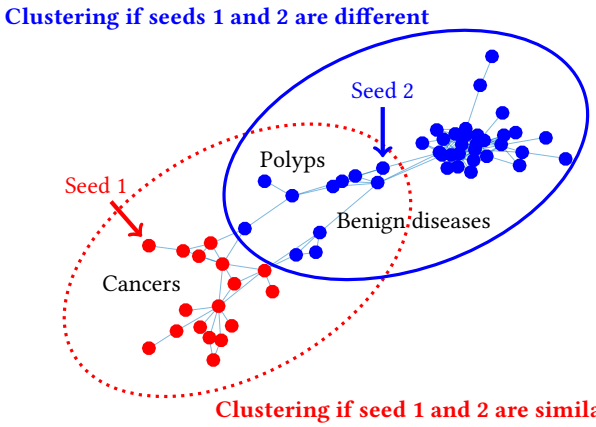


Figure 2: Example of a stomach disease similarity network. Vertices are diseases and edges are similarities between diseases. Red nodes are malignant diseases and blue nodes are benign ones. The dotted red circle contains diseases detected by a single querying seed 1. The solid blue circle encloses clustering result by knowing seed 1 and seed 2 are from different communities.

However, local clustering from either Dr. Wang or Dr. Kitagawa returns their own collaboration subgroup, which is not appropriate due to diversity concerns. On the other hand, assuming Dr. Wang and Dr. Kitagawa are in the same research community and querying jointly from both of them may return a larger community with diverse researchers relevant to them. The final recommendation can be decided by the proximity score of each researcher with respect to the two query seeds.

More importantly, knowing that seed nodes are different can separate a dense subgraph into semantically meaningful communities. Fig. 2 is a network of stomach diseases where red nodes are malignant tumors and blue nodes denote benign stomach diseases such as gastritis and ulcer. By conventional local clustering method, either query from seed 1 or from both seed 1 and 2 will enclose polyps into group of malignant diseases, which is not correct. On the contrary, with domain knowledge that seed 2 is benign and different from seed 1, stomach polyps can be well divided from malignant tumors by our proposed method. This helps the automatic medical diagnosis system to correctly identify a specific disease [13].

Generally, existing local graph clustering methods cannot take advantage of seed node labels, and prefer well-structured networks [8]. For networks without clear community structures, the performance of such methods is limited. In this paper, we introduce a method to leverage supervision of labeled seed node. If a clustering task has multiple seed nodes, we assume that the relationships of seed nodes are already known: we know that which seeds should be in the same community, and which should not. We name this local graph clustering problem with labeled seed nodes as constrained local graph clustering.

To solve the constrained local graph clustering problem, we propose a colored random walk (CRW) algorithm, which is motivated by random walk based local graph clustering [1]. In addition to using random walkers to explore a network, we label seed nodes and random walkers by colors, where seed nodes believed to be in the same community are marked with identical color. The algorithm launches colored random walkers for each color starting from the seed nodes with the corresponding color.

Unlike previous random walk methods [3, 20] where walkers change nothing to the graph, the colored random walker will leave colors to every node it visited. The colors on each node will affect the transition probability of subsequent walkers. In general, a walker is more likely to visit nodes with the same color. In other words, walkers are attracted by nodes with the same color and repulsed by nodes with different colors. The attraction and repulsion are affected by the amount of colors accumulated on each node. By this means, communities with different color of seed nodes are divided, even in unclear network structures.

For local graph clustering with single seed node, the proposed colored random method still outperforms the traditional approaches. This is due to the attraction of walkers by the community members which are assigned with the same color, especially by the central influential nodes with large degrees in each community. These nodes cumulate a higher amount of color by being visited more frequently (Fig. 9), and the cumulative color, in turn, prevents the random walkers from fleeing their community. Due to the same reason, our method is robust to the position of the seed nodes. No matter where is the seed node, the influential nodes remain the same, and so do the corresponding communities. This is different with the previous approaches, in which the performance degrades if seed nodes are close to cluster boundaries. Please refer to Section 5 for more details.

We formulate the colored random walk algorithm by power iterations, and introduce a heuristic speedup strategy to reduce the running time. By the nature of local graph clustering, nodes far away from seeds are not important. The strategy can speed up the searching process by more than 100 times while increasing the local clustering accuracy at the same time. Experiments show that the proposed algorithm outperforms the state-of-the-art baseline methods on both accuracy and speed.

The rest of the paper is organized as follows. Related works are discussed in Section 2. Section 3 explains the detail of our method. In Section 4, extensive experimental results are presented and analyzed. Section 5 provides two case studies for deeper insight of the proposed method, and finally Section 6 draws the conclusion.

2 RELATED WORK

The current local graph clustering methods have no restriction on seed nodes. Random Walk with Restart [20] uses Markovian random walkers to explore the networks. With certain probability, the walkers jump back to seed nodes. The local clusters are cut from nodes frequently visited by the walkers. Heat Kernel Diffusion [8] instead adopts values diffused from seed nodes to evaluate how close the nodes are to the seeds. Query-biased Densest Connect Subgraph method [22] weights nodes by random walk. Then the local community is selected to minimize a predefined goodness function. All these methods assume seed nodes are from a single cluster.

Semi-supervised learning, including semi-supervised clustering with constrains, gained success as they do not require a large amount of human annotations [4]. In semi-supervised learning, the labels are propagated from labeled seed instances by random-walk-like mechanisms through the similarity graph, and then a classifier is trained to categorize all other instances [24, 25]. However, traditional semi-supervised learning focuses more on classification than clustering, and is a heavy framework since it considers both local and global consistency during the learning process.

The colored random walk is also relevant to the Vertex-reinforced Random Walk (VRW), which views graph nodes as urns [16]. Each time the random walker visits a node, it puts a ball with its color into the corresponding urn. The further visiting probability of the random walker is reinforced by the numbers and colors of balls in each node. Though it is proved that single-color VRW is guaranteed to converge to a distribution [2], the converged distribution is not unique due to the randomness of the walker, which diminishes the performance of VRW in local graph clustering. Moreover, the theories of multi-color VRW are still underdeveloped.

3 OUR PROPOSAL

Local graph clustering requires proximity measurement to decide which nodes should be in a local cluster. Random walk based approaches such as Random Walk with Restart (a.k.a. Personalized PageRank) are commonly used to evaluate the node proximity in graphs [20]. Compared with other methods, random walk takes advantage of local network structure and reports better performance [9]. In this work, we extend the random walk method to colored random walk to solve the problem of local graph clustering with constrains.

3.1 Preliminaries

The local graph clustering targets on a graph $G = (V, E)$, where V is the node set and E is the edge set. The adjacency matrix of graph G is denoted by $\mathbf{A}$, whose element $\mathbf{A}_{ij}$ is the edge weight between node V_i and node V_j. $\mathbf{P}$ is the transposed column-stochastic transition matrix with entries $\mathbf{P}_{ji} = \mathbf{A}_{ij} / \sum_j \mathbf{A}_{ij}$. The main symbols and their definitions can be found in Table 1.

The core function of local graph clustering is to measure the proximity of each node with the seed node(s). In Random Walk with Restart (RWR), a random walker starts from the seed node q, and randomly walks to a neighbor node according to edge weights. At each time point $(t + 1)$, the walker continues walking with probability α or returns to the query node q with probability $(1-\alpha)$. The proximity score of node p with respect to node q is defined as the converged probability that the walker visits node p:

$$\mathbf{c}^{(t+1)} = \alpha \cdot \mathbf{P} \cdot \mathbf{c}^{(t)} + (1 - \alpha) \cdot \mathbf{s} \tag{1}$$

where $\mathbf{s}$ is the column vector of the initial distribution with q^{th} entry as 1 and all other entries 0, and $\mathbf{c}$ is the current distribution vector whose p^{th} entry is the visiting probability of node p.

RWR can be extended to accommodate multiple seed nodes. For n seed nodes, the initial vector $\mathbf{s}$ consists of entries with value $1/n$ for each seed node. However, if seed nodes are from multiple clusters, RWR cannot use such information. One may run RWR multiple times, each for a single seed node, yet this approach fails to take full advantage the prior knowledge of cluster membership of the seed nodes. For example, if two seed nodes are known from different clusters, walkers of each seed node should avoid walking into the other seed's cluster. We give the formal definition of constrained local graph clustering as follows.

Table 1: Main symbols

Symbol	Definition
$G = (V, E)$	graph G with node set V and edge set E
$\mathbf{A}, \mathbf{P}$	adjacency matrix and transition matrix
$\mathbf{c}, \mathbf{s}$	current and initial color distributions
α, θ	forward probability and color threshold
$\psi(t)$	decaying factor at time t
λ_1, λ_2	attraction and repulsion coefficient

Definition 3.1 (Constrained local graph clustering). Given graph $G = (V, E)$ and seed node set with K different colors, the constrained local graph clustering aims to find K local communities containing seed nodes of each color.

3.2 The Colored Random Walk

To solve the constrained local graph clustering problem, we propose a Colored Random Walk (CRW) method in this section, which takes advantage of seed node labels. Seed nodes of the same cluster are assigned with identical color, and seed nodes from different clusters have different colors. For each color, the algorithm sends out a colored walker from the corresponding seed nodes to explore the network. During the exploration, the walker leaves a certain amount of colored values to the visited nodes, similar to the RWR method in Eq. 1.

The main difference between CRW and conventional RWR is that the movement of walkers is affected by existing colors of nodes. Generally, a walker is more likely to visit nodes with identical colors, and is less likely to visit nodes with distinct colors. Since color values are changed at each time point t, the transition matrix $\mathbf{P}$ should be updated accordingly.

For a constrained local graph clustering task with K colors, the CRW algorithm maintains K transition matrix $\mathbf{P}^{(k,t)}$ for color k at time t. Then the Eq. 1 expands to a new power iteration formula

$$\mathbf{c}^{(k,t+1)} = \alpha \cdot \mathbf{P}^{(k,t)} \cdot \mathbf{c}^{(k,t)} + (1 - \alpha) \cdot \mathbf{s}^{(k)} \tag{2}$$

where $\mathbf{c}^{(k,t)}$ is the distribution of color k at time t, and $\mathbf{s}^{(k)}$ is the initial vector of color k, respectively.

3.3 Transition Matrix Reinforcement

The transition matrix $\mathbf{P}^{(k,t)}$ in Eq. 2 is based on the original transition matrix $\mathbf{P}$, and is reinforced by all current color distributions $\mathbf{c}^{(\ell,t)}$, $1 \le \ell \le K$. The reinforcement is done by

$$\mathbf{P}^{(k,t)} = \mathbf{P} \circ (1 + \mathbf{R}^{(k,t)}) \tag{3}$$

where operator $\circ$ is the Hadamard (entry-wise) product of two matrices with the same size [7]. By Hadamard product, we avoid adding new edges to graph G, and the reinforcement is done by weight adjustment of existing edges. In the Eq. 3, '1' is used to keep the original transition matrix $\mathbf{P}$ as part of the reinforced transition

matrix $\mathbf{P}^{(k,t)}$. $\mathbf{R}$ is a coefficient matrix defined by its entry

$$\mathbf{R}_{ij}^{(k,t)} = \sum_{\ell} \mathbf{c}_i^{(\ell,t)} \cdot \lambda_{k\ell}, \quad \lambda_{k\ell} = \begin{cases} \lambda_1 & k = \ell \\ -\lambda_2 & k \neq \ell \end{cases} \tag{4}$$

In Eq. 4, if the factor $\lambda_{k\ell}$ is defined as positive when color k equals color ℓ, and negative otherwise. This definition increases the likelihood of a walker going to the node with identical color, and decreases the likelihood of walking to the node with different colors. The mechanism can be understood as that, a walker is attracted by nodes with the same color, and repulsed by nodes with different colors. Here we denote by λ_1 the attraction coefficient and by λ_2 the repulsion coefficient.

By Eq. 3 and 4, the entries of $\mathbf{P}^{(k,t)}$ are possibly negative. To make $\mathbf{P}^{(k,t)}$ as a transition matrix meaningful, it is normalized before used in Eq. 2. The normalization includes two parts. First, the negative entries of $\mathbf{P}^{(k,t)}$ are set to be zero: $\mathbf{P}_{ij}^{(k,t)} \leftarrow 0$ if $\mathbf{P}_{ij}^{(k,t)} < 0$, and then each entry of the matrix is divided by the sum of its column to make the matrix column stochastic: $\mathbf{P}_{ij}^{(k,t)} \leftarrow \mathbf{P}_{ij}^{(k,t)} / \sum_i \mathbf{P}_{ij}^{(k,t)}$.

3.4 Convergence

Unlike the traditional random walk methods which have stationary transition matrix $\mathbf{P}$, the transition matrix $\mathbf{P}^{(k,t)}$ of colored random walk varies at each time t. To guarantee the convergence of Eq. 2, we introduce a decaying mechanism changing $\mathbf{P}^{(k,t)}$ to $\mathbb{P}^{(k,t)}$ (Eq. 5), where the value of function $\psi(t)$ approaches to 0 when t increases. Then the colored random walk is adapted to Eq. 6.

$$\mathbb{P}^{(k,t)} = \begin{cases} \mathbf{P} & t = 0 \\ \psi(t) \cdot \mathbf{P}^{(k,t)} + (1 - \psi(t)) \cdot \mathbb{P}^{(k,t-1)} & t \geq 1 \end{cases} \tag{5}$$

$$\mathbf{c}^{(k,t+1)} = \alpha \cdot \mathbb{P}^{(k,t)} \cdot \mathbf{c}^{(k,t)} + (1 - \alpha) \cdot \mathbf{s}^{(k)} \tag{6}$$

Here the definition of $\mathbf{P}$ remains the same with Eq. 3 and Eq. 4. Intuitively, when $\psi(t) \to 0$, $\mathbb{P}^{k,t} \to \mathbb{P}^{k,t-1}$, meaning that $\mathbb{P}^{k,t}$ converges. With a converged $\mathbb{P}^{k,t}$, the proximity score vector $\mathbf{c}^{(k,t)}$ converges, similar to the conventional random walk. The formal proof of convergence of $\mathbf{c}^{(k,t)}$ is given by Theorem 3.2.

Theorem 3.2 (Convergence of colored random walk). *Both $\mathbb{P}^{(k,t)}$ and $\mathbf{c}^{(k,t)}$ in Eq. 5 and Eq. 6 converge if $\psi(t) = \Omega(\alpha^t)$.*

Proof. To ease the description, we combine distributions, reinforced transition matrices, and initial vectors of all color at time t by

$$\mathbf{c}^{(t)} = \begin{bmatrix} \mathbf{c}^{(1,t)} \\ \mathbf{c}^{(2,t)} \\ \vdots \\ \mathbf{c}^{(K,t)} \end{bmatrix}, \mathbf{P}^{(t)} = \begin{bmatrix} \mathbf{P}^{(1,t)} & & & 0 \\ & \mathbf{P}^{(2,t)} & & \\ & & \ddots & \\ 0 & & & \mathbf{P}^{(K,t)} \end{bmatrix}, \mathbf{s} = \begin{bmatrix} \mathbf{s}^{(1)} \\ \mathbf{s}^{(2)} \\ \vdots \\ \mathbf{s}^{(K)} \end{bmatrix}$$

Evidently, Eq. 7 equals to Eq. 6 for all color $1 \leq k \leq K$. Similarly we define $\mathbb{P}^{(t)}$ as diagonal combination of all $\mathbb{P}^{(t,k)}$.

$$\mathbf{c}^{(t+1)} = \alpha \cdot \mathbb{P}^{(t)} \cdot \mathbf{c}^{(t)} + (1 - \alpha) \cdot \mathbf{s} \tag{7}$$

For convergence of decayed transition matrix $\mathbb{P}^{(t)}$, it is easy to know that for $t \geq 1$,

$$\|\mathbb{P}^{(t)} - \mathbb{P}^{(t-1)}\|_1 = \psi(t)\|\mathbf{P}^{(t)} - \mathbb{P}^{(t-1)}\|_1 \leq 2\psi(t)$$

Algorithm 1: The Colored Random Walk

Input : $\mathbf{P}, \mathcal{S}, \alpha, \lambda_1, \lambda_2, T, \psi(t)$
Output: $\mathbf{c}$

```
K ← |S|;
for k ← 1 to K do
    s^(k) ← 0;
    for each x ∈ S[k] do s_x^(k) ← 1/|S[k]| ;
s ← [s^(1); s^(2); ...; s^(K)];
c ← s;
P^(0) ← I_K ⊗ P;
ℙ ← P^(0);
Λ ← I_N · λ_1;
Set all non-diagonal entries of Λ as −λ_2;
Λ ← Λ ⊗ I_K;
for t ← 0 to T do
    c ← α · ℙ · c + (1 − α)s;
    r ← Λ · c;
    R ← r ⊗ 1^T_{N·K};
    P̂ ← P^(0) ∘ (1 + R);
    for each i and j do P̂_ij ← 0 for P̂_ij < 0;
    for each i and j do P̂_ij ← P̂_ij / Σ_x P̂_xj;
    ℙ ← ψ(t) · P̂ + (1 − ψ(t)) · ℙ;
```

For convergence of the color distribution $\mathbf{c}^{(t)}$, we define $\Delta^{(t)} = \|\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)}\|_1$. Then

$$\begin{aligned}
\Delta^{(t+1)} &= \|\mathbf{c}^{(t+1)} - \mathbf{c}^{(t)}\|_1 = \|\alpha\mathbb{P}^{(t+1)}\mathbf{c}^{(t)} - \alpha\mathbb{P}^{(t)}\mathbf{c}^{(t-1)}\|_1 \\
&= \alpha\|\mathbb{P}^{(t+1)}\mathbf{c}^{(t)} - \mathbb{P}^{(t)}\mathbf{c}^{(t-1)}\|_1 \\
&= \alpha\|\mathbb{P}^{(t+1)}\mathbf{c}^{(t)} - \mathbb{P}^{(t+1)}\mathbf{c}^{(t-1)} + \mathbb{P}^{(t+1)}\mathbf{c}^{(t-1)} - \mathbb{P}^{(t)}\mathbf{c}^{(t-1)}\|_1 \\
&= \alpha\|\mathbb{P}^{(t+1)}(\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)}) + (\mathbb{P}^{t+1} - \mathbb{P}^{t})\mathbf{c}^{t-1}\|_1 \\
&\leq \alpha\|\mathbb{P}^{(t+1)}(\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)})\|_1 + \alpha\|(\mathbb{P}^{(t+1)} - \mathbb{P}^{(t)})\mathbf{c}^{t-1}\|_1 \\
&\leq \alpha\|\mathbb{P}^{(t+1)}\|_1\|\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)}\|_1 + \alpha\|\mathbb{P}^{(t+1)} - \mathbb{P}^{(t)}\|_1\|\mathbf{c}^{t-1}\|_1 \\
&\leq \alpha\|\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)}\|_1 + \alpha\|\mathbb{P}^{(t+1)} - \mathbb{P}^{(t)}\|_1 \\
&\leq \alpha\Delta^{(t)} + 2\alpha\psi(t+1) \leq \alpha^2\Delta^{(t-1)} + 2\alpha^2\psi(t) + 2\alpha\psi(t+1) \\
&\leq \alpha^t\Delta^{(1)} + 2\sum_{\tau=1}^{t} \alpha^\tau\psi(t-\tau+2)
\end{aligned}$$

To make $\Delta^{(t+1)}$ converge to 0, a sufficient condition is that $\psi(t) = \Omega(\alpha^t)$, which means $0 \leq c\psi(t) \leq \alpha^t$ for any $t > t_0$ and $c > 0$. Given $\psi(t) \leq 2\alpha^{t-1}$ for $t > t_0$,

$$\begin{aligned}
\Delta^{(t+1)} &\leq 2\alpha^t + 2\sum_{\tau=t+2-t_0}^{t} \alpha^\tau + 2\sum_{\tau=1}^{t+1-t_0} \alpha^{t+1} \\
&< 2\alpha^t + 2(t_0 - 1)\alpha^{t-t_0+2} + 2(t - t_0 + 1)\alpha^{t+1} \\
&< 2t_0\alpha^{t-t_0+2} + 2(t - t_0 + 1)\alpha^{t+1} \to 0
\end{aligned}$$

□

3.5 The Algorithm

By Eq. 7, we can formulate the colored random walk as shown in Algorithm 1. The input of the algorithm includes original transition matrix $\mathbf{P}$, seed node sets $\mathcal{S}$ of all colors, penalty factor α, attraction coefficient λ_1 and repulsion coefficient λ_2, together with the number of iterations T and the decaying faction $\psi(t)$. The output of the algorithm is the color distribution vector $\mathbf{c}$ for all colors. The details of the algorithm are as follows.

Lines 1 – 11 initialize the algorithm. Specifically, in line 1 we get the number of colors K by the number of seed node sets in $\mathcal{S}$. Lines 3 and 4 prepare for the initial color vectors $\mathbf{s}^{(k)}$ for each color k. From line 5 to line 8 we combine initial distributions and transition matrix of all colors together by $\mathbf{s}$ and $\mathbf{P}^{(0)}$, as explained in Section 3.4, and initialize the color distribution vector $\mathbf{c}$ and the modified transition matrix $\mathbb{P}$.

To calculate the color reinforcement for each node, we create a $N \times N$ factor matrix $\mathbf{\Lambda}$ (lines 9 to 11), which originally has the value of λ_1 on all diagonal entries and $-\lambda_2$ on all non-diagonal entries. The size of matrix $\mathbf{\Lambda}$ is then amplified (line 11) by Kronecker Product [7] with an identity matrix $\mathbf{I}_K$, to multiply with vector $\mathbf{c}$ (line 14), which is a matrix-form equivalence to the operation in Eq. 4.

The for loop from line 12 to 19 is the power iteration process. We first generate the color distribution for next iteration (line 13), and then generate the reinforcement vector $\mathbf{r}$ (line 14). Vector $\mathbf{r}$ is repeated to form a matrix $\mathbf{R}$ (line 15), which means that the reinforcement is fair to every node. After generating new reinforced matrix $\hat{\mathbf{P}}$ (line 16), we remove negative entries (line 17) and normalize it to column stochastic (line 18), and update the decayed transition matrix for the next iteration (line 19).

The power iteration part of the algorithm is critical in analyzing the time complexity, where all operations including calculations of color distribution (line 13), reinforcement (line 14) and decayed transition matrix (line 19) take $O(K|E|)$, if we base the operations on sparse matrices. Therefore, the overall time complexity is $O(TK|E|)$. As empirically reported in [9], $T = 10$ is sufficient for the convergence of distribution $\mathbf{c}$.

With the distribution $\mathbf{c}$ for each color, we rank nodes by their colors and cut local communities by sweeping from the top-rank node and find a cluster with minimum conductance [1].

3.6 Speedup Strategy

Time complexity of the proposed colored random walk algorithm is proportional to the number of edges $|E|$, implying that the algorithm searches the entire graph for local clusters. This is undesirable as the task of local clustering focuses only on the nodes close to the seeds.

In this section, we propose a localized variant of Algorithm 1, with the basic idea that random walkers only search the influential nodes around the seed nodes. Unlike Algorithm 1 where the walkers go as far as T-hops from the seeds, we only consider nodes with colors above a given tolerance threshold θ.

As the random process in Eq. 2 can be viewed as the diffusion of colors from each node to its neighbors, the key point of Algorithm 2 is that only colors with values above a threshold θ will diffuse to next iteration. The intuition behind this is that small color amount will not affect colors of next iteration very much. When a walker reaches a node with color value less than θ, it stops there and does not go to the node's neighbors, which prevents the process from searching too many further nodes. We also neglect the decaying function $\psi(t)$ and reinforce the node on demand (with color value greater than θ) to accelerate the algorithm.

Algorithm 2: The Localized Colored Random Walk

```
   Input  : P, S, α, λ1, λ2, T, θ
   Output : c
1  for each color k do
2      c^(k) ← 0;
3      for each node i ∈ S[k] do c_i^(k) ← 1/|S[k]| ;
4  for t ← 1 to T do
5      ĉ^(k) ← 0;
6      for each color k do
7          for each c_i^(k) > θ do
8              p ← 0;
9              for each neighbor j of node i do
10                 p_j ← P_ij^(k,t);
11                 for each color ℓ do
12                     if ℓ = k then p_j ← p_j + λ1 · c_j^(ℓ) ;
13                     else p_j ← p_j − λ2 · c_j^(k) ;
14                 if p_j < 0 then p_j ← 0 ;
15             for each p_j > 0 do ĉ_j^(k) ← ĉ_j^(k) + α · c_i^(k) · p_j / Σ p ;
16         for each seed node i ∈ S[k] do
17             ĉ_i^(k) ← ĉ_i^(k) + (1 − α)/|S[k]|;
18     for each color k do c^(k) ← ĉ^(k) ;
```

In Algorithm 2, lines 1 – 3 initialize the color distribution according to seed nodes, and lines 4 – 18 simulate the power iteration in a heuristic way. In this heuristic algorithm, we remove the process of decaying, as it is shown that the result of experiments without decaying is similar. Specifically, at each iteration (line 4) the current color scores are in $\mathbf{c}^{(k)}$ for color k, and the new scores for next iteration are calculated and stored in $\hat{\mathbf{c}}^{(k)}$. Note that, $\mathbf{c}^{(k)}$ is changed to $\hat{\mathbf{c}}^{(k)}$ after the scores of all colors are updated (line 18). Note that the color scores are updated simultaneously.

By this strategy, though color values diminish at each iteration, the summation of colors remains its majority value. The difference turns out to be less than 1%.

4 EXPERIMENTAL VALIDATION

Extensive experiments are conducted on both synthetic and real-world data to evaluate the effectiveness and efficiency of the proposed algorithms. All experiments are performed on a server with Redhat OS, Xeon 2.6GHz CPU, and 64G memory. The core functions are implemented by C++.

Each experimental result is gathered on average of 10,000 trials. The querying seeds are randomly selected. The accuracy is measured by F1-score, which is defined as $2pr/(p + r)$, where p and r

are precision and recall, respectively.

4.1 Non-overlapping Clusters

We use the following networks with non-overlapping ground truth clusters to test the effectiveness of our proposed method. The clusters are large enough to test the searching ability of the colored random algorithm algorithm.

Disease networks. The networks contain a disease similarity network [13] with 9,721 diseases. For this network, there are two sets of clustering ground truth for the disease network, with 17 classes and 47 classes, respectively. We use both sets of ground truth, and denote the experiments by Disease 1 and Disease 2, accordingly. Disease 3 is a phenotype similarity network with 5,080 diseases in 20 categories [21].

Gene network. The gene network represents the functional relationship between 8,503 genes [21] with 200 pathway labels.

Author network. This network comprises 20,111 authors from 4 research areas: Data Mining, Data Base, Information Retrieval and Machine Learning. The edge of this network is the number of papers co-published by the authors.

Conference network. This is a full graph comprises 20 main conferences from the same 4 areas with the author network. The edges are cosine similarity of keywords from the publication of the conferences. We use research areas as the ground truth for both author network and conference network.

Since no previous method is designed for the constrained local graph clustering problem, we compare the proposed colored random walk (CRW) method with the following five state-of-the-art local clustering methods, which support multiple seed node querying: Random Walk with Restart (RWR) [20] uses Eq. 1 to calculate the proximity score of each node; Heat Kernel Diffusion (HKD) [8] measures node proximity by heat kernel network diffusion; Query-biased Densest Connect subgraph (QDC) [22] assigns weights to nodes using RWR scores and finds the query biased densest subgraph; Vertex-reinforced Random Walk (VRW) [12] reinforces the next-hop visiting probability by previous visiting counts. For our method CRW, we use the localized colored random walk (Algorithm 2) with $\alpha = 0.9$, $\lambda_1 = 1000$, $\lambda_2 = 10$, $\theta = 10^{-5}$.

Effectiveness evaluations are conducted on the selected networks. Since the accuracy difference between our two algorithms is small, we present the results of Algorithm 2 only. Table 2 shows the result of single seed node querying. By column "CRW" we only use one seed node with a single color, so there is no repulsion of different colors, which serves as a baseline of our method. From the results in Table 2, it can be seen that even the basic CRW can outperform the state-of-the-art methods in terms of clustering accuracy.

In Table 2 we add an extra column of CRW_1^1, which randomly picks up two seed nodes of different colors to examine the impact of color repulsion. Comparison of CRW and CRW_1^1 shows that resistance of distinct colors helps to improve the clustering accuracy significantly. On all datasets CRW_1^1 achieves the best performance.

Table 3 shows the result of 2-seed-node query on the same 6 datasets. For each method except CWR_2^2 we use two seed nodes from the same cluster. CWR_2^2 is similar to CWR_1^1 by choosing 2 colors, each having two seeds. Though initially VRW is not designed for multiple seed nodes, we make it work by adding extra random walkers that are reinforced by rules of the original VRW. Comparing Table 3 with Table 2, the conclusion is that almost all methods improve clustering accuracy by having more seeds. Furthermore, CWR gains more improvement on average. Again CRW_1^1 and CRW_2^2 have the best performance among all chosen methods.

Table 2: 1-query accuracy for non-overlapping clusters

F1-score	RWR	HKD	QDC	VRW	CRW	CRW_1^1
Disease 1	0.33	0.27	0.11	0.08	**0.34**	**0.37**
Disease 2	0.42	0.48	0.32	0.25	**0.49**	**0.53**
Disease 3	0.24	0.30	0.14	0.13	**0.30**	**0.38**
Gene	0.05	0.06	0.09	0.15	**0.16**	**0.22**
Author	0.27	0.03	0.01	0.01	**0.30**	**0.40**
Conference	0.42	0.38	0.40	0.57	**0.66**	**0.71**

Table 3: 2-query accuracy for non-overlapping clusters

F1-score	RWR	HKD	QDC	VRW	CRW	CRW_2^2
Disease 1	0.40	0.30	0.11	0.10	**0.41**	**0.44**
Disease 2	0.43	0.54	0.38	0.29	**0.57**	**0.61**
Disease 3	0.26	0.33	0.19	0.10	**0.38**	**0.39**
Gene	0.05	0.07	0.08	0.17	**0.21**	**0.35**
Author	0.32	0.02	0.01	0.01	**0.35**	**0.41**
Conference	0.43	0.45	0.40	0.57	**0.89**	**0.91**

4.2 Overlapping Clusters

Intuitively colored random walk works with non-overlapping clusters. However, due to its color-agglomerating nature, it also works for detecting overlapping clusters. Table 4 lists the statistics 4 of networks with overlapping ground truth communities (Amazon, DBLP, YouTube, LiveJournal). These datasets are publicly available at http://snap.stanford.edu. For the quality of clusters, we select clusters with size larger than 10, and within $\pm 3\sigma$ range of the average cluster size. The last column of Table 4 shows the mean and standard deviation (σ) of the chosen clusters.

Table 4: Networks with overlapping communities

Name	$\lvert V\rvert$	$\lvert E\rvert$	# clusters	avg (σ)
Amazon	334,863	925,872	75,149	13.23 (11.24)
DBLP	317,080	1,049,866	13,477	10.87 (20.41)
YouTube	1,134,890	2,987,624	8,385	10.22 (18.81)
LiveJournal	3,997,962	34,681,189	287,512	23.67 (24.75)

Table 5: 1-query accuracy on overlapping communities

F1-score	RWR	HKD	QDC	VRW	CRW	CRW_1^1
Amazon	0.92	**0.93**	0.90	0.70	0.92	**0.93**
DBLP	0.41	0.59	0.65	0.57	**0.72**	**0.76**
YouTube	0.05	0.06	0.19	0.13	**0.27**	**0.30**
LiveJournal	0.69	0.74	0.74	0.42	**0.81**	**0.84**

Table 6: 2-query accuracy on overlapping communities

F1-score	RWR	HKD	QDC	VRW	CRW	CRW_2^2
Amazon	0.93	0.94	0.91	0.79	**0.95**	**0.96**
DBLP	0.42	0.61	0.73	0.68	**0.78**	**0.81**
YouTube	0.05	0.06	0.21	0.16	**0.38**	**0.42**
LiveJournal	0.69	0.75	0.79	0.45	**0.84**	**0.86**

The clustering results are listed in both Table 5 and Table 6. It shows that 2-seed quires are generally better than single seed node. Among all methods HKD is relatively good since it is refined for the small overlapping communities. However, more seed nodes do not improve its performance very much (only 1% on average). The probability-reinforced methods VRW and CRW gain the biggest improvement by adding an extra seed. Once again our proposed method leads the accuracy measurements. For seed nodes more than 2, Fig. 3 shows that most methods perform better with more seeds. However, CRW, VRW and QDC take better advantage of more seed nodes, as they all have mechanisms of reinforcement on node weights.

Fig. 3 gives the results where seed nodes are from a single community. However, when seed nodes are from distinct communities, the other methods cannot use this relationship, so their performances are very low (Fig. 4). On the contrary, our method takes advantage of this cannot-link information, resulting in the increased accuracy with more communities (colors).

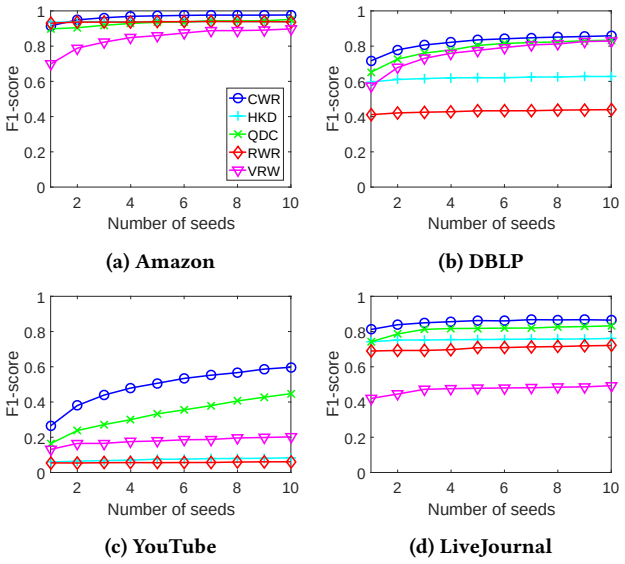
We also evaluate the efficiency of the selected methods on datasets provided in Table 4, since networks are relatively large. To be fair, the running time is measured on single color and single node on every method. Fig. 5 shows that CRW is one of the fastest methods of all, comparable to the HKD method, and is generally 100 times faster than the other approaches. Generally, CRW takes no more than 200 milliseconds in finding a target local cluster. For the scalability of colors and seed nodes, the running time of CRW does not increase with number of seed nodes, and is proportional with the number of colors.

4.3 Impact of Color Attraction and Repulsion

To analyze the sensitivity of the main parameters of our method (λ_1, λ_2, θ), we generate synthetic networks by the standard LFR benchmark [10].

We first test the impact of attraction and repulsion of Algorithm 1, by varying factor λ_1 and λ_2 on the synthetic graph of Fig. 6. The results are presented in Fig. 7, in terms of accuracy vs. attraction λ_1 and repulsion λ_2. We set $\psi(t) = 0.9^t$ is this experiment.

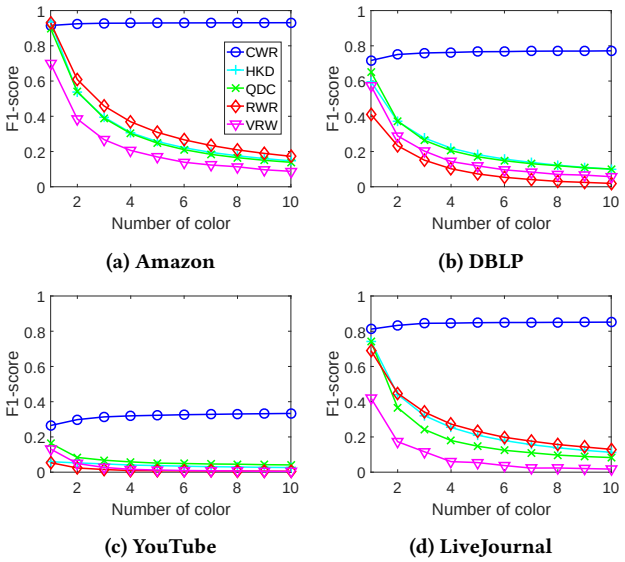
First we try our algorithm on the traditional problem of local graph clustering, where there is only one target community and accordingly only one color (Fig. 7a). We randomly choose 1 seed node (1×1) or 2 seed nodes (2×1) from a community, or 4 seed nodes from 2 different communities (2×2). Since there is no different color, we use attraction coefficient λ_1 only and set $\lambda_2 = 0$. In Fig. 7a the x-axis is the logarithmic value of λ_1 and y-axis is the F1-score. When λ_1 is close to 0, our algorithm degenerates into the conventional random walk based local clustering. By this means we can compare our methods with the basic random walk, as well as measuring the sensitivity of parameters λ_1 and λ_2.



(a) Amazon (b) DBLP

(c) YouTube (d) LiveJournal

Figure 3: Impact of adding more seed nodes



(a) Amazon (b) DBLP

(c) YouTube (d) LiveJournal

Figure 4: Impact of adding more colors

From curve of 1×1 we can see that the clustering accuracy increases obviously with the parameter λ_1. This indicates that even on traditional graph clustering problem, our algorithm still outperforms the basic random walk method. For curve 2×1, initially its accuracy is higher than 1×1 because of the information introduced by more seed nodes. However, the accuracy drops slightly after $\lambda_1 > 10^3$, because excessively strong attraction around the two individual seed nodes separates the target community. The attraction even works when seeds nodes are not from the same community, but labeled as the same color (curve 2×2).

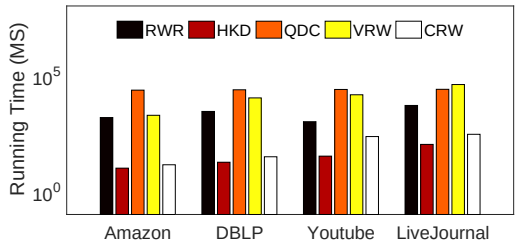


Figure 5: Running time comparison

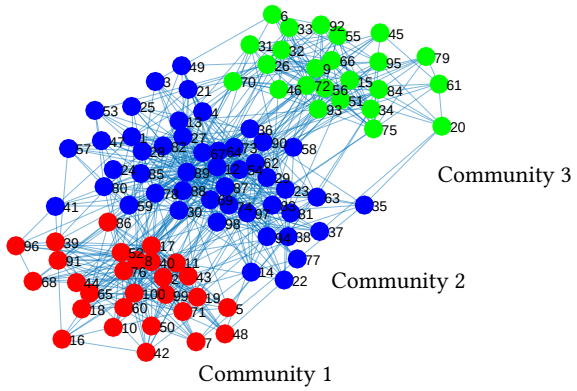


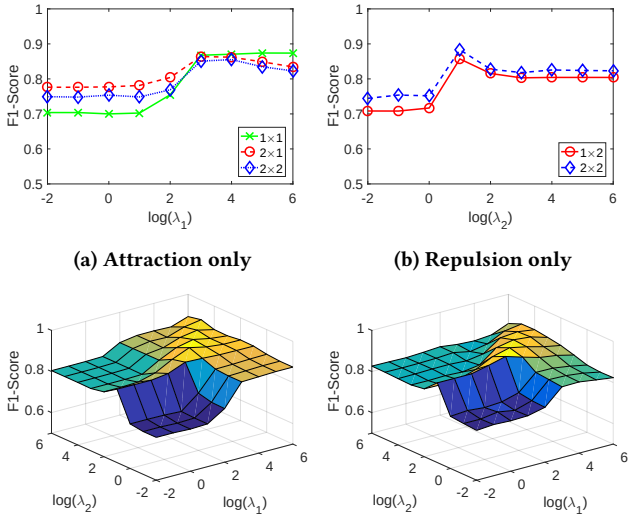
Figure 6: A 100-node graph generated by the LFR benchmark. The red, blue, green nodes are from 3 distinct ground-truth communities.

Similar results are observed by using repulsion force only and set $\lambda_1 = 0$. In experiments shown in Fig. 7b, we select 1 or 2 nodes from two adjacent communities and label them with different colors. The figures illustrate that the repulsion enhances clustering accuracy, as walkers from the two different communities are divided by distinct colors. However, overdose of repulsion ($\lambda_2 > 10$) drives walkers from the border of communities and decreases the accuracy.

Finally, we combine both attraction and repulsion forces together (shown in Fig. 7c and 7d). The best accuracy results come with $\lambda_1 = 10^4$ and $\lambda_2 = 10^2$, and the performance are maintained when these two parameters keep on increasing. The favorable λ_1 and λ_2 are large because of the stabilizer "1" in color reinforcement (Eq. 3), which requires large λ_1 and λ_2 to change the behavior of random walkers.

4.4 Evaluation of Speedup Strategy

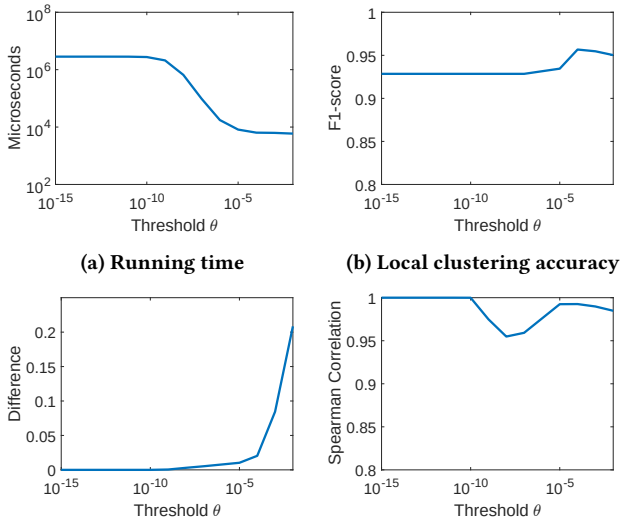
We generate a network with 1 million nodes and 15 million edges by LFR benchmark to test the impact of threshold θ on Algorithm 2. The results are presented in Fig. 8. When the threshold θ rises from 10^{-15} to 10^{-5}, the average running time drops dramatically (Fig. 8a). Note that when $\theta \to 0$, Algorithm 2 reduces to a non-decaying version of Algorithm 1. The total speedup is more than 2 orders of magnitudes, when threshold increases to above 10^{-5}.



(a) Attraction only **(b) Repulsion only**

(c) 1×2 attraction and repulsion **(d) 2×2 attraction and repulsion**

Figure 7: Impact of color attraction and repulsion. λ_1 is the attraction coefficient and λ_2 is the repulsion coefficient.



(a) Running time **(b) Local clustering accuracy**

(c) Vector difference **(d) Spearman correlation**

Figure 8: Impact of threshold θ on Algorithm 2

In the meantime, the clustering accuracy remains intact and even increases from $\theta = 10^{-5}$. Looking closely, the recall of clustering remains stable while the precision rises after $\theta > 10^{-5}$. This is because the nodes of the target local clustering (true positive) are not faraway from seed nodes. Larger threshold θ makes the search more local: thus irrelevant nodes will not be enclosed, which reduces the false positive.

We also examine the difference between the output vectors from Algorithm 1 and Algorithm 2. From Fig. 8d, the summation of absolute difference value is only 1.03% at $\theta = 10^{-5}$ (Fig. 8c). Does

this difference substantially change the orders of nodes, which is crucial to clustering result? Spearman's rank correlation reveals that it is not the case (Fig. 8d). In calculating correlation, we only compare non-zero entries from both vectors. Though the correlation drops a little between 10^{-10} and 10^{-5} due to the noise introduced by the speedup strategy, it returns to above 0.98 after $\theta > 0.5$, where the correlation is limited to local clusters only. This indicates that the heuristic of Algorithm 2 has very little deflection on the order of nodes in the target local cluster.

In summary, the introduction of threshold θ largely reduces the running time of the proposed algorithm, and the clustering accuracy is enhanced. We adopt $\theta = 10^{-5}$ as a trade-off between running time and accuracy in our other experiments.

5 CASE STUDY

In this section we present two case studies to examine nature of the proposed colored random walk algorithm, explain why CRW works with even a single color, and provide more examples on how CRW divides dense subgraph into meaningful partitions.

5.1 Karate Club Social Network

In this section we look into the colored random walk (CRW) method by the karate club network [23], to see how CRW improves the clustering accuracy even by a single seed node.

The karate club network contains 34 members (nodes) (Fig. 9). Edges are interactions among members. A conflict between the club instructor (node 1) and the president (node 34) divides the club into two separate groups (labeled by blue and red, respectively), which serves as clustering ground truth. It is obvious that node 1 and node 34 are influential nodes with large degrees in the two groups, connecting to many other members.

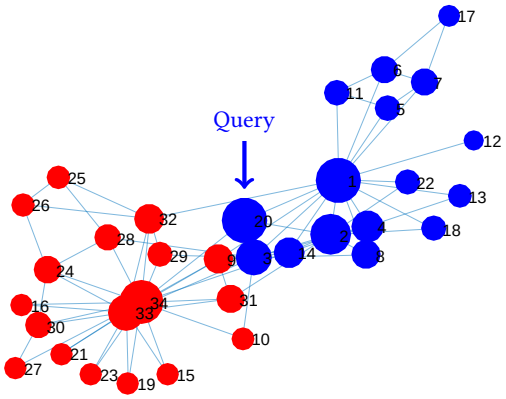
We choose node 20 as the seed node for both CRW and random walk with restart (RWR). It is a hard task as node 20 is on the border of these two clusters, though it is categorized to the blue cluster. The proximity scores are represented by size of nodes in Fig. 9.

From Fig. 9a, we can see that the proximity score distribution is relatively even on all nodes. This is not desirable since it is not easy to distinguish the node membership by the flat proximity scores. In contrast, proximity scores concentrate on the blue community, especially the central influential nodes 1 (Fig. 9b). These nodes accumulate more colors (proximity scores) and have more attraction to the colored walkers, and essentially helps to divide the two clusters.

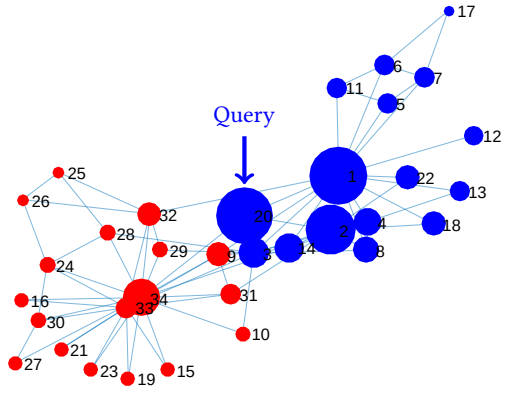
5.2 Brain Activity Network

Fig. 10a depicts another example where prior knowledge about the difference of seed nodes is critical. It is part of a brain activity network where each node is a cortical area and edges are functional associations [5]. The nodes in this network are from Broadman area 6 (BA6) and Broadman area 13 (BA13). Since these two areas are closely connected with each other, conventional methods cannot divide these areas. Once knowing the difference of seed nodes (e.g., node 7 and 14), it becomes easier to separate the two Broadman areas. The clustering result can be viewed by 3D coordinates[1] in Fig. 10b.

[1]http://www.talairach.org/



(a) Proximity scores by random walk with restart (RWR), $\alpha = 0.9$, $T = 10$



(b) Proximity scores by colored random walk (CRW), $\alpha = 0.9$, $T = 10$, $\lambda_1 = 70$, $\lambda_2 = 0$, $\psi(t) = 1$

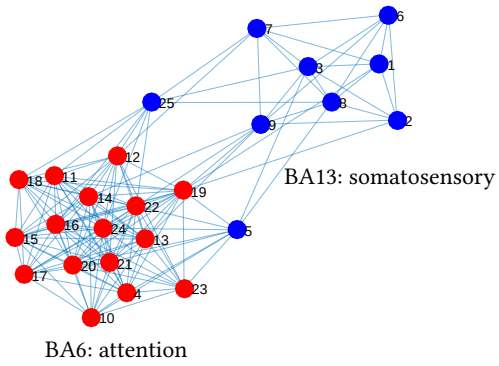
Figure 9: Comparison of proximity scores of RWR and CRW on the karate club network

6 CONCLUSION

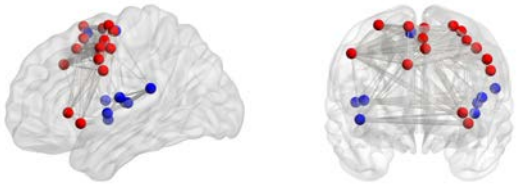
In this paper we advance a colored random walk method to solve the constrained local graph clustering problem, where the communities membership of seed nodes are known as prior knowledge. The proposed method dynamically adjusts node weights for each color during the random walk process, and the updated node weights help to trap the colored random walkers into the corresponding communities. Experiments show that the performance of our method is above the state-of-the-art approaches, even for conventional single-color and single-seed scenario. With more seed nodes and colors available, the performance of the proposed method is significantly better.

ACKNOWLEDGEMENT

This work was partially supported by the National Science Foundation grant IIS-1707548.



(a) Part of brain activity network



(b) 3D coordinates of the network

Figure 10: Each node is an area in brain. Red and blue nodes are clusters divided with prior knowledge that node 3 and node 24 are different

REFERENCES

[1] Reid Andersen, Fan Chung, and Kevin Lang. 2006. Local graph partitioning using pagerank vectors. In *FOCS*.

[2] Michel Benaïm et al. 1997. Vertex-reinforced random walks and a conjecture of Pemantle. *The Annals of Probability* 25, 1 (1997), 361–392.

[3] Yuchen Bian, Jingchao Ni, Wei Cheng, and Xiang Zhang. 2017. Many Heads are Better than One: Local Community Detection by the Multi-Walker Chain. In *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 21–30.

[4] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. 2006. Semi-Supervised Learning. Adaptive Computation and Machine Learning series.

[5] Nicolas A Crossley, Andrea Mechelli, Petra E Vértes, Toby T Winton-Brown, Ameera X Patel, Cedric E Ginestet, Philip McGuire, and Edward T Bullmore. 2013. Cognitive relevance of the community structure of the human brain functional coactivation network. *Proceedings of the National Academy of Sciences* 110, 28 (2013), 11583–11588.

[6] Wanyun Cui, Yanghua Xiao, Haixun Wang, and Wei Wang. 2014. Local search of communities in large graphs. In *SIGMOD*.

[7] Roger A Horn, Roger A Horn, and Charles R Johnson. 1990. *Matrix analysis*. Cambridge university press.

[8] Kyle Kloster and David F Gleich. 2014. Heat kernel based community detection. In *SIGKDD*.

[9] Isabel M Kloumann and Jon M Kleinberg. 2014. Community membership identification from small seed sets. In *KDD*.

[10] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. 2008. Benchmark graphs for testing community detection algorithms. *Physical review E* 78, 4 (2008), 046110.

[11] Rui Liu, Wei Cheng, Hanghang Tong, Wei Wang, and Xiang Zhang. 2015. Robust Multi-Network Clustering via Joint Cross-Domain Cluster Alignment. In *ICDM*.

[12] Qiaozhu Mei, Jian Guo, and Dragomir Radev. 2010. Divrank: the interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. Acm, 1009–1018.

[13] Jingchao Ni, Hongliang Fei, Wei Fan, and Xiang Zhang. 2017. Automated Medical Diagnosis by Ranking Clusters Across the Symptom-Disease Network. In *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 1009–1014.

[14] Jingchao Ni, Hongliang Fei, Wei Fan, and Xiang Zhang. 2017. Cross-Network Clustering and Cluster Ranking for Medical Diagnosis. In *ICDE*.

[15] Jingchao Ni, Mehmet Koyuturk, Hanghang Tong, Jonathan Haines, Rong Xu, and Xiang Zhang. 2016. Disease gene prioritization by integrating tissue-specific molecular networks using a robust multi-network model. *BMC bioinformatics* 17, 1 (2016), 453.

[16] Robin Pemantle et al. 2007. A survey of random processes with reinforcement. *Probability surveys* 4 (2007), 1–79.

[17] Satu Elisa Schaeffer. 2007. Graph clustering. *Computer science review* 1, 1 (2007), 27–64.

[18] Mauro Sozio and Aristides Gionis. 2010. The community-search problem and how to plan a successful cocktail party. In *KDD*.

[19] Hanghang Tong, Christos Faloutsos, Brian Gallagher, and Tina Eliassi-Rad. 2007. Fast best-effort pattern matching in large attributed graphs. In *KDD*.

[20] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast random walk with restart and its applications. (2006).

[21] Marc A Van Driel, Jorn Bruggeman, Gert Vriend, Han G Brunner, and Jack AM Leunissen. 2006. A text-mining analysis of the human phenome. *European journal of human genetics* 14, 5 (2006), 535–542.

[22] Yubao Wu, Ruoming Jin, Jing Li, and Xiang Zhang. 2015. Robust local community detection: on free rider effect and its elimination. *Proceedings of the VLDB Endowment* 8, 7 (2015), 798–809.

[23] Wayne W Zachary. 1977. An information flow model for conflict and fission in small groups. *Journal of anthropological research* 33, 4 (1977), 452–473.

[24] Denny Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *Advances in neural information processing systems*. 321–328.

[25] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*. 912–919.

Revisiting User Mobility and Social Relationships in LBSNs: A Hypergraph Embedding Approach

Dingqi Yang, Bingqing Qu, Jie Yang and Philippe Cudre-Mauroux
University of Fribourg, Switzerland
{firstname.lastname}@unifr.ch

ABSTRACT

Location Based Social Networks (LBSNs) have been widely used as a primary data source to study the impact of mobility and social relationships on each other. Traditional approaches manually define features to characterize users' mobility homophily and social proximity, and show that mobility and social features can help friendship and location prediction tasks, respectively. However, these handcrafted features not only require tedious human efforts, but also are difficult to generalize. In this paper, by revisiting user mobility and social relationships based on a large-scale LBSN dataset collected over a long-term period, we propose LBSN2Vec, a hypergraph embedding approach designed specifically for LBSN data for automatic feature learning. Specifically, LBSN data intrinsically forms a hypergraph including both user-user edges (friendships) and user-time-POI-semantic hyperedges (check-ins). Based on this hypergraph, we first propose a random-walk-with-stay scheme to jointly sample user check-ins and social relationships, and then learn node embeddings from the sampled (hyper)edges by preserving n-wise node proximity ($n = 2$ or 4). Our evaluation results show that LBSN2Vec both consistently and significantly outperforms the state-of-the-art graph embedding methods on both friendship and location prediction tasks, with an average improvement of 32.95% and 25.32%, respectively. Moreover, using LBSN2Vec, we discover the asymmetric impact of mobility and social relationships on predicting each other, which can serve as guidelines for future research on friendship and location prediction in LBSNs.

CCS CONCEPTS

• **Information systems** → *Data mining*; *Web applications*.

KEYWORDS

Mobility, Social relationship, Location based social network, Link prediction, Hypergraph, Embeddings

ACM Reference Format:
Dingqi Yang, Bingqing Qu, Jie Yang and Philippe Cudre-Mauroux. 2019. Revisiting User Mobility and Social Relationships in LBSNs: A Hypergraph Embedding Approach. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313635

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313635

1 INTRODUCTION

Understanding the correlation between human mobility and social relationships is crucial for studying human dynamics, which is also a key ingredient for friendship and location prediction tasks. Location Based Social Networks (LBSNs), such as Foursquare, have attracted millions of users and generated a considerable amount of digital footprints from their daily life. As such, they have become a primary data source to study the impact of mobility and social relationships on each other. Specifically, in LBSNs, users can share their real-time presences with their friends by checking-in at a Point of Interest (POI), such as a restaurant or a gym. In addition to such fine-grained and semantic user mobility information, the social network of the corresponding users is also available.

Based on LBSN data, empirical analyses have shown various socio-spatial properties of user activities, in particular the correlation between user mobility characteristics (e.g., distance, co-location rate, etc.) and social networks [8, 26–28, 42]. Based on these findings, two main applications have been investigated, i.e., friendship prediction (a.k.a. link prediction) and location prediction. *Friendship prediction* aims at recommending social relationships that are likely to be established in the future [28], while *location prediction* tries to predict which POI a user will go to in a given context (e.g., at a given time) [8]. Existing work has shown that considering the correlation between user mobility and social relationships can improve the performance of both friendship prediction [7, 25, 28, 32, 42] and location prediction [8, 12, 18, 21, 25]. More precisely, these approaches usually select a set of hand-crafted features either from user mobility data or from the corresponding social network, and then show the impact of one on the other. For example, social features usually involve different network proximity metrics including common neighbor, Adamic-Adar [1], Katz index [17], etc. For mobility features, co-location rate is a widely used metric to measure the homophily between two users in terms of mobility traces; it has different variations such as normalized/unnormalized, weighted/unweighted, spatial only/spatiotemporal [28, 33]. However, such a manual feature engineering process not only requires tedious efforts from domain experts, but also shows less generalizability [13] (see Section 5.2.2 and 5.3.2 for more detail).

To overcome the limitation of hand-crafted features, automatic feature learning (a.k.a. representation learning) has been proposed [3]. When applied to networks or graphs, this paradigm is typically known as *graph (or network) embedding* [5], which represents nodes of a graph in a low-dimensional vector space while preserving key structural properties of the graph (e.g., topological proximity of the nodes). Based on such embeddings, graph analysis tasks (e.g., link prediction) can then be efficiently performed. As shown in Figure 1, LBSN data in this context intrinsically form a hypergraph consisting of four key data domains, i.e., spatial, temporal, semantic and

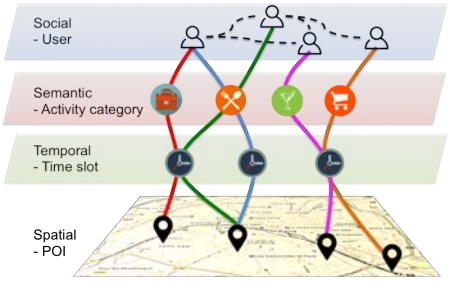


Figure 1: LBSN hypergraph containing both user mobility data (check-ins) and the corresponding social network. A friendship is represented by an edge (a black dotted line) linking two user nodes. A check-in is represented by a hyperedge (a colored thick line) linking four nodes, i.e., a user, an activity type, a time stamp and a POI.

social. This graph contains not only classical edges (i.e., friendships between two users in the social network), but also hyperedges (i.e., check-ins linking four nodes, one from each domain, representing a *user*'s presence at a *POI* at a specific *time* along with the *semantic* information about the user's activity there).

However, existing graph embedding techniques cannot fully grasp the complex data structure of LBSNs. First, most of the existing techniques were developed for classical graphs [6, 13, 22–24, 29, 30], where an edge links two nodes only; the node embeddings are then learnt such that the pairwise node proximity is preserved. However, preserving the pairwise node proximity cannot fully capture the information from the check-in hyperedges. Even though a hypergraph can be transformed into a classical graph by breaking each hyperedge into multiple classical edges, such an irreversible process causes a certain information loss, leading to degraded performance of the learnt node embeddings on different tasks (see Section 5.2 and 5.3 for more detail). Second, there are also a few techniques studying the hypergraph embedding problem, but they either focus on a n-uniform hypergraph (where all hyperedges contain a fixed number n of nodes) and thus capture only fixed-n-wise node proximity [2, 31, 44], or learn from hyperedges for heterogeneous events only (hyperedges linking nodes from different data domains) while ignoring edges within a data domain [14]. However, as shown in Figure 1, the LBSN hypergraph typically contains both classical friendship edges (n=2) within the user domain and check-in hyperedges (n=4) across all four data domains. Subsequently, these techniques cannot be directly applied to the full LBSN hypergraph. Even though these techniques could be applied on check-in hyperedges (n=4) only, ignoring social relationship indeed shows suboptimal performance for different tasks (see Section 5.3 for more detail).

Against this background, we revisit human mobility and social relationships in LBSNs in this paper and propose LBSN2Vec, a hypergraph embedding approach designed specifically for LBSN data with the special consideration of learning from *both* friendship edges *and* check-in hyperedges at the same time. Our contributions are three-fold:

- We collect a large-scale LBSN dataset over a long-term period and conduct a rigorous empirical analysis to systematically reveal the correlations between user check-ins and the corresponding social network. Specifically, existing publicly available LBSN datasets collected from Foursquare [12], Gowalla [8] or Brightkite [8], usually contain check-in data from a set of users over a period of time, and one snapshot of the corresponding user social network. However, it is often unclear when the social network snapshot is collected (e.g., before, during or after the check-in data collection period); such chronological information is important in studying the impact of user mobility and the corresponding social network on each other. For example, to investigate the impact of social networks on user mobility patterns, only friendship formed before a check-in should be considered. Therefore, our collected dataset contains not only check-ins of a set of users over a two-year period, but also two snapshots of the corresponding user social network before and after the check-in data collection period, respectively. Based on this dataset, we conduct a rigorous empirical analysis of the impact of user mobility and social network on each other, showing that each of the four data domains has a clear impact on both friendship and location prediction tasks.
- We propose LBSN2Vec, a hypergraph embedding approach that can efficiently learn node embeddings from both friendship edges and check-in hyperedges in a LBSN hypergraph. To this end, we first propose a random-walk-with-stay scheme to jointly sample friendship and check-ins from a LBSN hypergraph. To balance the impact of social relationships and user mobility on the learnt node embeddings, we incorporate a tunable parameter to control the portion of social relationships and check-ins in the learning process in a probabilistic manner. Moreover, to learn node embeddings from hyperedges containing n nodes, we design LBSN2Vec to preserve the n-wise node proximity by simultaneously optimizing the proximity of the n nodes from a hyperedge. More precisely, we first compute the best-fit-line of the n nodes in cosine space, and then iteratively maximize the cosine similarity between each node of the hyperedge and the best-fit-line.
- We conduct a thorough evaluation using our collected dataset. Our results show that LBSN2Vec outperforms state-of-the-art graph embedding techniques on both friendship and location prediction tasks, with an average improvement of 32.95% and 25.32%, respectively. Moreover, using LBSN2Vec, we discover the asymmetric impact of user mobility and social networks on predicting each other. We find that node embeddings learnt from 80% social and 20% mobility data results in the best performance on the friendship prediction task, while those learnt from 40% mobility and 60% social data give the best performance on the location prediction task. Such an observation can serve as guidelines for future research on friendship and location prediction.

2 RELATED WORK

2.1 Human Mobility and Social Relationships

The interaction between human mobility and social relationships has been widely studied using different data sources over the past decade. In the earlier stage, call detail records have been often used to study human mobility, social ties and link prediction problems

[8, 33]. Although these studies have revealed several interesting properties of human mobility and social ties, such data shows two obvious limitations. First, it contains only coarse-grained user location information on a cell tower level. Second, it does not contain actual information about social relationships between users; the corresponding social network are usually built from the users' communication activities based on certain heuristics. For example, a friendship is assumed between a pair of users when there is at least one call between them [33] (or 10 calls in [8]).

The emergence of Location Based Social Networks provides a novel opportunity to collect both large-scale user mobility data (i.e., check-ins) and the corresponding social network. Using LBSN data, positive correlations between social proximity and mobility homophily have been universally found [8, 33]. However, existing publicly available LBSN datasets (Foursquare [12], Gowalla [8] or Brightkite [8]) mainly contain check-in data of a set of user over a period of time, and one snapshot of the corresponding social network without mentioning when the social network is collected[1]; such chronological information is important in studying the impact of user mobility and the corresponding social network on each other. Therefore, in this paper, we collected our own dataset from Foursquare, containing not only check-ins of a set of users over about two years, but also two snapshots of the corresponding user social network before and after the check-in data collection period, respectively. Based on this dataset, we investigate the impact of user check-ins and the corresponding social network on both friendship and location prediction tasks.

First, friendship prediction is a classical problem in social network analysis. It predicts the potential friendship between two users based on the existing social network [19]. Using LBSN data, friendship prediction approaches [7, 25, 28, 32, 42] often combine social proximity and mobility pattern similarity to achieve better prediction performance. For example, Scellato et al. [28] investigated a set of hand-crafted features and showed that besides social proximity, mobility pattern similarity is also a strong indicator to predict future friendship. Sadilek et al. [25] manually combined features from social proximity, mobility similarity, and textual similarity extracted from users' Tweets for friendship prediction.

Second, location prediction is a typical problem in human mobility modeling, which predicts the location of a user under a certain context based on the user's historical mobility traces. Using LBSN data, location prediction approaches incorporate social factors in mobility models, showing an improved prediction performance [8, 12, 18, 21, 25]. For example, Gao et al. [12] combined check-in patterns and social ties by considering mobility similarity between friends (co-location rate measured by cosine distance). Noulas et al. [21] investigated the next place prediction problem and found that besides a user's own check-in history, a "social filtering" feature serves as an important predictor for the user's next location.

However, the manual feature engineering process adopted by the existing work not only requires a lot of human efforts, but also shows less generalizability [13]. In this paper, we propose LBSN2Vec, a hypergraph embedding approach to automatically learn the node embeddings from a LBSN hypergraph, based on which both friendship and location prediction tasks can be efficiently performed. Moreover, our LBSN2Vec can also model the impact of user mobility and social relationships on each other.

[1]Note that one of the existing work [28] used multiple social network snapshots to study the friendship prediction problem, but the dataset is not publicly accessible. Moreover, it used hand-crafted features for friendship prediction.

2.2 Graph Embeddings

Most existing graph embedding approaches focus on preserving pairwise node proximity in a classical graph, which can be further classified into two categories according to the embedding learning process. First, *factorization based approaches* [6, 22, 24] measure pairwise node proximity as a matrix using a certain network proximity metric, such as common neighbor or Adamic-Adar, and then factorize this proximity matrix using matrix factorization techniques to learn the node embeddings. However, factorization-based approaches have an intrinsic scalability limitation, due to the quadratic complexity of matrix factorization algorithms [4]. Second, *graph-sampling based approaches* [13, 16, 23, 29, 30] sample node pairs (directly or via random walks) from a graph, and then design specific models to learn node embeddings from the sampled node pairs via stochastic optimization. These graph-sampling based approaches are able to scale up to large datasets, as their complexity mainly depends on the number of the sampled node pairs.

Moreover, there are also a few approaches studying the hypergraph embedding problem, but they either focus on a n-uniform hypergraph (where all hyperedges contain a fixed number n of nodes) and thus capture only fixed-n-wise node proximity [2, 31, 44], or learn from hyperedges for heterogeneous event only (hyperedges linking nodes from different data domains) while ignoring edges within a data domain [14]. However, in this paper, as shown in Figure 1, our LBSN hypergraph contains both classical friendship edges (n=2) within the user domain and check-in hyperedges (n=4) across all four data domains. Subsequently, these techniques cannot be directly applied to our full LBSN hypergraph. Even though they can be applied only on the check-in hyperedges (n=4) for example, ignoring social relationships indeed yields suboptimal performance (see Section 5.3 for more detail). Against this background, we propose LBSN2Vec to efficiently learn node embeddings from the LBSN hypergraph by preserving the n-wise node proximity encoded by the hyperedges (where $\{n \in \mathbb{Z}^+ | n \geq 2\}$).

We also note that there are several works exploited embedding techniques on LBSN data, but they mostly focus on check-in data only (without using social networks) for specific tasks, such as semantic annotation of POIs [34], POI recommendation [10, 35], next visitor prediction [11]. Only one recent work [43] used both social network and check-in data on LBSNs for learning embeddings for users and POIs, but it overlooks the temporal and semantic information of check-ins. Moreover, none of these works have investigated the correlations between human mobility and social relationships.

3 EMPIRICAL DATA ANALYSIS

In this section, we first present our dataset collection and its main characteristics, followed by empirical studies on the impact of mobility on the social network and vice versa.

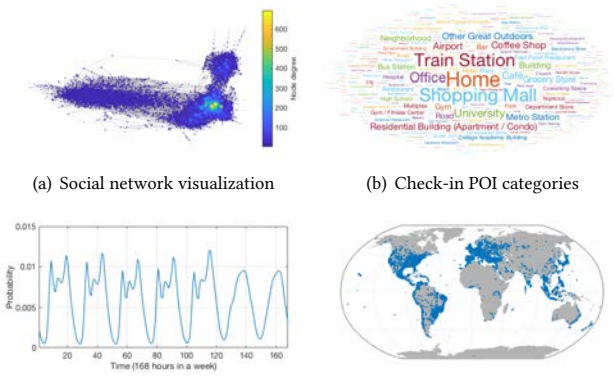
3.1 Dataset Collection and Characteristics

Our collected Foursquare dataset contains a set of global-scale check-ins over about two years (from Apr. 2012 to Jan. 2014), and two snapshots of the corresponding user social network before (in Mar. 2012) and after (in May 2014) the check-in data collection period. The check-in data is collected from Foursquare, which has been widely used as a key data source for studying location based services [39–41]. Specifically, as Foursquare user's check-ins can only be accessed from the user's social circle, they are not publicly available. However, many Foursquare users choose to also share their check-ins via Twitter [37]. Therefore, we collect global-scale check-in data by searching Foursquare-tagged Tweets from Twitter Public Streams for about two years (from Apr. 2012 to Jan. 2014). We keep only active users (who have performed at least one check-in per month). User social relationships are collected from Twitter, as Foursquare friendship information is not publicly available. Specifically, for each user, we first crawl her "followers" (users who follow this user) and "friends" (users who are followed by this user) on Twitter. We then build the corresponding user social network by keeping only reciprocal relationships [38], i.e., a friendship is assumed between two users if the two users follow each other. We collect two snapshots of such social network before (in Mar. 2012) and after (in May 2014) the check-in data collection period. After filtering out isolated users (i.e., users without social links), we keep only users appearing in both social network snapshots. In summary, our dataset contains 22,809,624 check-ins from 114,324 users on 3,820,891 POIs, and two snapshots of the corresponding user social network before (363,704 friendships) and after (607,333 friendships) the check-in data collection period. Our dataset is available here[2].

Figure 2 shows a series of visualizations of our dataset in all four data domains. For the social domain, Figure 2(a) shows the old social network with a color bar indicating user node degrees. We observe only a few nodes with high degrees, while most of the nodes are of low degrees, which suggests a heavy-tailed distribution. We do not show the new social network as its plot shows a very similar visual structure as the old one. For the semantic domain, Figure 2(b) shows the categorical distribution of check-in POIs, where we have a total number of 453 POI categories (i.e., activity categories). We find that daily-routine related activities including "Shopping Mall", "Home", "Train station" and "Office", are among the most frequent categories. For the temporal domain, Figure 2(c) plots the weekly temporal traffic pattern of check-ins over 168 hours in a week. We observe not only a clear daily periodicity, but also the difference between weekday (with three peaks in the morning, afternoon and evening, respectively) and weekend (with only one flat peak during the daytime). Based on this observation, we define the time granularity (time slot) as 168 hours (corresponding to a full week) for this study. For the spatial domain, we visualize the GPS coordinates of POIs on a world map in Figure 2(d). We observe POIs from our dataset spread all over the world, with urban areas concentrating many of the POIs.

In addition, we take a closer look at the dataset from a statistical point of view. Figure 3 shows the Complementary Cumulative Distribution Function (CCDF) of different statistics of the collected dataset. First, we observe a heavy-tailed distribution of all statistics,

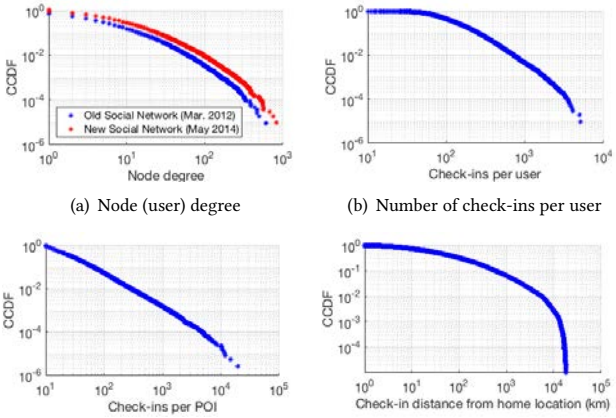
[2]https://sites.google.com/site/yangdingqi/home/foursquare-dataset



(a) Social network visualization (b) Check-in POI categories

(c) Temporal traffic of check-ins (d) World map of check-in POIs

Figure 2: Visualization of the collected LBSN data



(a) Node (user) degree (b) Number of check-ins per user

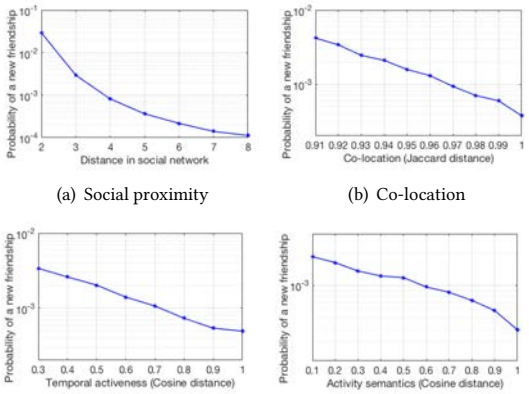
(c) Number of check-ins per POI (d) Check-in distance from home

Figure 3: Complementary Cumulative Distribution Function (CCDF) of various statistics in the collected dataset

including user node degrees in Figure 3(a), number of check-ins per user in Figure 3(b), number of check-ins per POI in Figure 3(c), and check-in distance from home location [8] in Figure 3(d), which implies a large variance in these statistics. For example, the heavy-tailed distribution of check-in distance from home shows that 70% of the check-ins happen less than 100km from home. Moreover, we observe that the user social network significantly expanded over the two years in Figure 3(a); the average degree increases from 3.38 to 5.56. In the following, we study the impact of mobility on the corresponding social network and vice versa.

3.2 Impact of Mobility on New Friendships

We investigate the probability of establishing a *new* friendship (appearing in the new but not in the old social network), w.r.t. various social and mobility factors in the old social network and user check-in data. Figure 4 shows the results. On one hand, the existing social network is unsurprisingly the primary predictor of future friendships [19]. Figure 4(a) shows the impact of the social proximity between users in the old social network. We observe that pairs of users at closer social distance are more likely to establish a new friendship. On the other hand, we use the following three metrics to



(a) Social proximity (b) Co-location

(c) Temporal activeness (d) Activity semantics

Figure 4: Impact of social and mobility factors on establishing new friendships

measure the similarity of user mobility patterns on spatial, temporal and semantic domains, respectively.

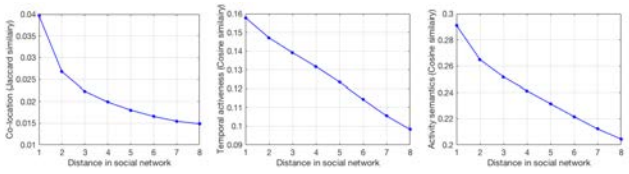
- **Co-location** [33]: By modeling each user's check-in data as a vector, where each element in the vector indicates whether the user has checked-in at the corresponding POI or not, we measure the Jaccard distance between two users' check-in vectors.
- **Temporal activeness** [36]: We model a user's check-in data by a vector encoding the user's check-in frequency over 168 hours in a week (as shown in Figure 2(c)), and then measure the Cosine distance between two users' temporal activeness vectors.
- **Activity semantics** [40]: We model a user's check-in data by a vector encoding the categorical distribution of a user's checked POIs (i.e., the distribution on POI categories as shown in Figure 2(b)), and then measure the Cosine distance between two users' activity semantic vectors.

Figure 4(b), 4(c) and 4(d) show the impact of the above three metrics on the probability of establishing a new friendship, respectively. We consistently observe that pairs of users with similar mobility patterns (measured in any of the three domains) are more likely to establish a new friendship in the future. *These observations suggest that all four data domains should be considered in the friendship prediction task. Note that we only use these metrics to investigate which data domains should be considered in the LBSN2Vec, yet none of these metric is required in our node embedding learning process.*

3.3 Impact of Friendship on Mobility

We investigate the similarity of user mobility patterns w.r.t. the social proximity between users in the *old* social network. Figure 5 shows the results. We consistently observe that pairs of users at closer social proximity are more likely to exhibit similar mobility patterns measured by all the three metrics, i.e., co-location, temporal activeness and activity semantics. Such an observation suggests that social relationships can be used to help predict user mobility.

In addition, we also investigate the impact of a user's historical mobility traces on her future mobility. To achieve this goal, we chronologically split our check-in data (of about two years) into



(a) Co-location (b) Temporal activeness (c) Activity semantics

Figure 5: Impact of social proximity on mobility factors

two parts of the same duration, i.e., a first-year check-in dataset and a second-year check-in dataset. Subsequently, we compute the similarity between each user's mobility patterns in the first and the second year, measured by the aforementioned three metrics. The average similarity is 0.12, 0.32 and 0.38 for co-location, temporal activeness and activity semantics, respectively. Compared to the mobility similarity between friends (the distance in the old social network is 1 in Figure 5, i.e., 0.04, 0.16 and 0.29 for co-location, temporal activeness and activity semantics, respectively), we find that a user's own mobility history is indeed the primary predictor of her future mobility. *Subsequently, we also consider all four data domains in our LBSN2Vec for node embedding learning in the location prediction task. We note again that as an automatic feature learning approach, LBSN2Vec does not involve any of the metrics defined in this section.*

4 LBSN2VEC

In this paper, we adopt a graph-sampling based embedding paradigm due to its intrinsic scalability advantage as discussed in Section 2.2. Specifically, our method first uses a novel random-walk-with-stay scheme to jointly sample friendships and check-in hyperedges[3] from the LBSN hypergraph, and then learn node embeddings from these hyperedges to preserve the n-wise node proximity by optimizing the proximity of the n nodes from a hyperedge simultaneously.

4.1 Random Walk with Stay

As shown in Figure 1, the LBSN hypergraph consists of four data domains, i.e., spatial, temporal, semantic and social domain, and two types of edges, i.e., classical edges (friendships) linking pairs of user nodes and hyperedges (check-ins) linking four nodes, one from each domain. To sample (hyper)edges from this graph, we propose a random-walk-with-stay scheme to jointly sample friendship and check-in hyperedges. As shown in Figure 6, our random-walk-with-stay scheme performs classical random walk only on user nodes based on their friendships in social domain, while for each encountered user node, it stays on the user node to sample a set of hyperedges (check-ins) from the corresponding user. Subsequently, in the node embedding learning process, when iterating over each user node in a random walk sequence, we not only learn from two user nodes appearing with a context window of length k, but also stay on the corresponding user node to learn from its check-in hyperedges. In other words, the node embedding learning process alternates between these two types of edges (i.e., friendships and check-ins). To perform classical random walk on the user social

[3] A classical edge linking two user nodes (i.e., a friendship) can be regarded as a special case of a hyperedge consisting of only two nodes. In the following, besides the check-in hyperedges, we also use the term "hyperedge" to describe a pair of user nodes.

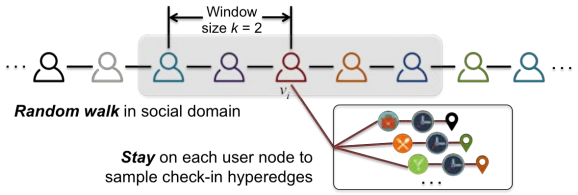
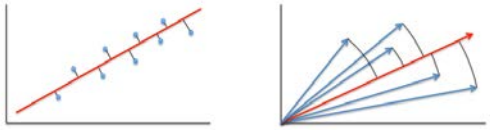


Figure 6: Random walk with stay on the LBSN hypergraph



(a) Euclidean space with perpendicular distance (b) Cosine space with cosine distance

Figure 7: Examples of the best-fit-lines in Euclidean and Cosine spaces. The given data points are represented in blue, while the corresponding best-fit-line is shown in red.

network, we use the same strategy as used by existing works [13, 23], generating r walks of length l rooted on each user node.

Moreover, in order to balance the impact of friendships and check-ins on the learnt node embeddings, LBSN2Vec incorporates a tunable parameter α to control the portion of learning edges of each type. Specifically, for a given context window of length k, we iterate over $2k$ contexts for each user node v_i (k context nodes on each side), resulting in $2k$ pairs of user nodes. Figure 6 shows an example where the context window size is 2. Subsequently, we sample for each user node the same number ($2k$) of check-ins. Afterwards, we learn from each check-in with a probability α, while from each user node pair with a probability $(1-\alpha)$, where α and $(1-\alpha)$ actually specify the impact of mobility and social network on the learnt node embeddings, respectively. In summary, for each node v_i in a random walk, the expected total number of learnt edges is $2k$, including an expected number $2k\alpha$ of check-ins and an expected number of $2k(1-\alpha)$ of user node pairs. A small value of α gives less importance on the check-in data and more on social network, and vice versa. In the following, we present our node embedding learning process preserving the n-wise node proximity for hyperedges.

4.2 Learning from Hyperedges

We design a novel embedding model in LBSN2Vec to preserve the n-wise node proximity by maximizing the similarity between the nodes of a hypergraph and their best-fit-line under cosine similarity. Specifically, our embedding model needs to learn from hyperedges linking either 2 or 4 nodes. We further generalize this problem to hyperedges containing n nodes, where $\{n \in \mathbb{Z}^+ | n \geq 2\}$, and design our model that can preserve the n-wise node proximity from those hyperedges. As the hyperedges are actually sampled and learnt one by one from a random walk, we focus in the following on the learning process for one hyperedge.

To learn node embeddings from a hyperedge containing n nodes, we borrow the idea of best-fit-line (a.k.a. line of best fit) from linear regression [15]. In general, a best-fit-line is a straight line that is the best approximation of the given data points. Figure 7(a) illustrates an example of the best-fit-line in Euclidean space that minimizes the sum of perpendicular distances, which can be computed via linear least squares. In this paper, following existing graph embedding techniques [6, 13, 22–24, 29, 30], we learn node embeddings in cosine space where the proximity between nodes are measured using cosine similarity (or dot product of the normalized embedding vectors). Moreover, we show below that the computation of the best-fit-line in cosine space can be significantly simplified. Formally, the best-fit-line of a set of node embeddings is the vector that minimizes the sum of cosine distances between each node embedding and the best-fit-line, as shown in Figure 7(b), which can be efficiently computed using Proposition 1:

Proposition 1. *For a set of nodes (in a hyperedge) $\{v_i | i = 1, 2, \ldots, n\}$, the corresponding best-fit-line can be computed as $\vec{v}_b = \sum_{i=1}^{n} \frac{\vec{v}_i}{\|\vec{v}_i\|}$, where $\vec{v}_i$ refer to the embedding vector of node v_i.*

Proof. The problem of finding the best-fit-line $\vec{v}_b$ that minimizes the sum of cosine distances can be formulated as follows:

$$\underset{\vec{v}_b}{\text{argmin}} \sum_{i=1}^{n} (1 - \cos(\vec{v}_i, \vec{v}_b)) \tag{1}$$

As minimizing the cosine distance is equivalent to maximizing the cosine similarity, we obtain:

$$\underset{\vec{v}_b}{\text{argmax}} \sum_{i=1}^{n} \cos(\vec{v}_i, \vec{v}_b) = \underset{\vec{v}_b}{\text{argmax}} \left(\sum_{i=1}^{n} \frac{\vec{v}_i}{\|\vec{v}_i\|} \right) \cdot \frac{\vec{v}_b}{\|\vec{v}_b\|} \tag{2}$$

Let $\vec{v}_a = \sum_{i=1}^{n} \frac{\vec{v}_i}{\|\vec{v}_i\|}$, we obtain:

$$\underset{\vec{v}_b}{\text{argmax}}\ \vec{v}_a \cdot \frac{\vec{v}_b}{\|\vec{v}_b\|} = \underset{\vec{v}_b}{\text{argmax}} \|\vec{v}_a\| \frac{\vec{v}_a \cdot \vec{v}_b}{\|\vec{v}_a\| \|\vec{v}_b\|} \tag{3}$$

where $\|\vec{v}_a\|$ is a constant for the given set of node. Subsequently, the above function actually maximizes the cosine similarity between $\vec{v}_a$ and $\vec{v}_b$. As the maximal cosine similarity between any two vectors is achieved if and only if the two vectors have the same orientation, the best-fit-line $\vec{v}_b$ can be simply computed as:

$$\vec{v}_b = \vec{v}_a = \sum_{i=1}^{n} \frac{\vec{v}_i}{\|\vec{v}_i\|} \tag{4}$$

This completes the proof. □

Therefore, we optimize the n-wise node proximity by iteratively maximizing the cosine similarity between each node embedding $\vec{v}_i$ from the hyperedge $\{v_i | i = 1, 2, \ldots, n\}$ and the best-fit-line $\vec{v}_b$.

$$\Theta = \sum_{i=1}^{n} \cos(\vec{v}_i, \vec{v}_b) \tag{5}$$

Note that here we keep using cosine similarity rather than dot product (which is widely used by existing techniques), because cosine similarity is not affected by the norm of the input vectors, in particular when the norm of the best-fit-line $\vec{v}_b$ computed by Eq. 4 has a large variance due to the variant number of the nodes n in different hyperedges ($n = 2$ or 4 in our LBSN hypergraph).

In addition, similar to other graph-sampling based embedding techniques [13, 23, 29, 30], we also adopt a negative sampling technique to maximize the cosine distance between each negative sample node v_N and the best-fit-line vector $\vec{v}_b$. Finally, we want

Table 1: Statistics of the selected datasets

Dataset	NYC	TKY	IST	JK	KL	SP
#User	4,024	7,232	10,367	6,395	6,432	3,954
#POI	3,628	10,856	12,693	8,826	10,817	6,286
#Check-ins	105,961	699,324	908,162	378,559	526,405	249,839
#Friendships(Before)	8,723	37,480	21,354	11,207	16,161	9,655
#Friendships(After)	10,545	51,704	36,007	16,950	31,178	14,402

to maximize the following objective function for one hyperedge $\{v_i|i = 1, 2, \ldots, n\}$:

$$\Theta = \sum_{i=1}^{n} (\cos(\vec{v}_i, \vec{v}_b) + \gamma \cdot \mathbb{E}_{v_N}[1 - \cos(\vec{v}_N \cdot \vec{v}_b)]) \tag{6}$$

where $\gamma \in \mathbb{Z}^+$ is the number of negative samples. Note that as the LBSN hypergraph contains four data domains, the negative samples for a node v_i from one data domain are uniformly sampled from the nodes in the same data domain.

The above objective function can be optimized using stochastic gradient descent techniques. Note that we only need to update $\vec{v}_i$ and $\vec{v}_N$ when learning from one specific hyperedge, as the best-fit-line vector $\vec{v}_b$ is fixed. The gradient of the above objective function with respect to $\vec{v}_i$ and $\vec{v}_N$ is computed as follows:

$$\frac{\partial \Theta}{\partial \vec{v}_i} = \frac{\vec{v}_b}{\|\vec{v}_i\|\|\vec{v}_b\|} - \frac{\vec{v}_i \cos(\vec{v}_i, \vec{v}_b)}{\|\vec{v}_i\|^2} \tag{7}$$

$$\frac{\partial \Theta}{\partial \vec{v}_N} = -\frac{\vec{v}_b}{\|\vec{v}_N\|\|\vec{v}_b\|} + \frac{\vec{v}_N \cos(\vec{v}_N, \vec{v}_b)}{\|\vec{v}_N\|^2} \tag{8}$$

In summary, to learn node embeddings from the LBSN hypergraph, we first sample a set of friendship and check-in hyperedges using our proposed random-walk-with-stay scheme, and then iteratively learn from each hyperedge $\{v_i|i = 1, 2, \ldots, n\}$ ($n = 2$ for friendships or $n = 4$ for check-ins) by optimizing the objective function in Eq. 6.

5 EXPERIMENTS

In this section, we evaluate LBSN2Vec on both friendship and location prediction tasks using our collected dataset. In the following, we first present our experimental setup, and then discuss the results on individual tasks, followed by a parameter sensitivity study.

5.1 Experimental Setup

5.1.1 Dataset. In this study, we investigate the friendship and location prediction at an urban scale. Without loss of generality, we select six cities with a large number of check-ins, while also considering the cultural diversity of the selected cities: New York City (**NYC**), Tokyo (**TKY**), Istanbul (**IST**), Jakarta (**JK**), Kuala Lumpur (**KL**), Sao Paulo (**SP**). We select users in the largest connected components of the old social network. Table 1 summarizes the statistics of the selected datasets.

5.1.2 Baselines. As our objective is automatic feature learning from LBSN data, we compare our method to the following state-of-the-art graph embedding methods.

- **DeepWalk** [23] first generates node sequences using random walks, and then feeds them to a SkipGram model to output node embeddings. We set the walk length to $l = 80$, the number of walks per node to $r = 10$, and the context window size for the SkipGram model to $k = 10$.
- **Node2vec** [13] extends DeepWalk by introducing a parameterized random walk method to balance the breadth-first search (return parameter p) and depth-first search (in-out parameter q) strategies, to capture richer graph structures. Following the suggestions in the original paper, we tune p and q with a grid search over $p, q \in \{0.25, 0.05, 1, 2, 4\}$. We keep the other parameters l, r, k similar as for DeepWalk.
- **LINE** [29] explicitly defines 1st and 2nd-order node proximity to learn graph embeddings. Specifically, to learn node embeddings of dimension d, it first learns two sets of $d/2$-dimension node embeddings preserving 1st and 2nd-order node proximity, respectively, and then concatenates them together.
- **HEBE** [14] learns node embeddings for *heterogeneous event data* using tensor modeling, where a heterogeneous hyperedge links nodes across different domains only and is modeled as an entry in an high-order tensor.
- **DHNE** [31] uses a deep autoencoder to learning node embeddings from a *n-uniform heterogeneous hypergraph*. It preserves both fixed-n-wise node proximity encoded by the hyperedges, and also the high-order node proximity defined as the similarity between nodes' neighborhood structures.

As the baseline methods cannot directly take the LBSN hypergraph as input graph, we propose three settings to adapt our LBSN hypergraph:

- **(S)**: Only the Social network is considered by keeping nodes in user domain and their friendship edges. This setting can be applied to classical graph embedding techniques (preserving pairwise node proximity), i.e., **DeepWalk**, **Node2vec** and **LINE**.
- **(M)**: Only the user Mobility is considered by keeping check-in hyperedges only. This setting can be directly applied to **HEBE** and **DHNE** which can take heterogeneous check-in hyperedges as input. In addition, we also apply this setting to **DeepWalk**, **Node2vec** and **LINE** by breaking each check-in hyperedge into multiple classical edges linking each pair of nodes in the hyperedge (e.g., user-time, time-POI, etc.).
- **(S&M)**: The full LBSN hypergraph containing both friendship edges and check-in hyperedges. This setting can be directly applied to our **LBSN2Vec**. In addition, we also apply this setting to **DeepWalk**, **Node2vec** and **LINE** by mapping the full LBSN hypergraph onto a classical graph by keeping friendship edges and also breaking each check-in hyperedge into multiple classical edges linking each pair of nodes in the hyperedge (same as for the M setting).

For our method LBSN2Vec, we keep the same random walk parameters as for DeepWalk and Node2vec ($l = 80$, $r = 10$, $k = 10$), and tune the parameter α within $[0, 1]$ with a step of 0.1 to balance the impact of social and mobility on the learnt embeddings (more discussion on this point in Section 5.4). The dimension of the node embeddings d is set to 128 and the number of negative samples γ is set to 10 for all methods in all experiments, if not specified otherwise. The implementation of LBSN2Vec is available here[4].

[4] https://github.com/eXascaleInfolab/LBSN2Vec

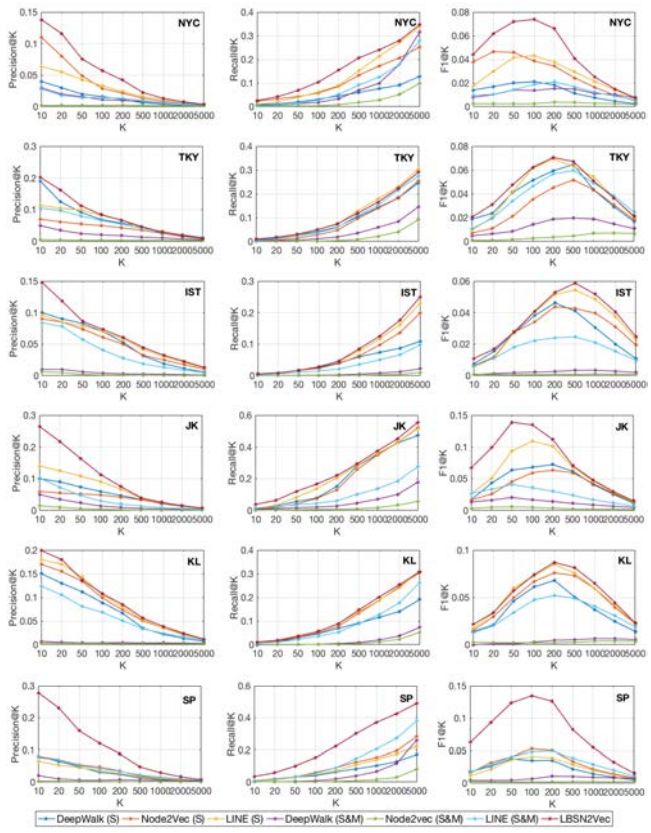


Figure 8: Friendship prediction performance comparison with other graph embedding methods. Each row shows the results on one dataset, while each column shows the results for one evaluation metric.

5.2 Friendship (link) Prediction

Friendship prediction recommends friendships that will probably be established in the future. In this study, we advocate an unsupervised friendship prediction approach [19, 20] which generates a ranking list of potential links between pairs of users. Specifically, after learning the node embeddings based on the *old* social network and check-ins, we rank pairs of user nodes (not being friends in the old social network) according to the cosine similarity between their embeddings. We then evaluate the obtained ranking list against the new friendships (appearing in the new but not in the old social network), using three common metrics for evaluating ranked results, i.e., precision, recall, and F1-score on the top K predicted friendships [19]. As the number of candidate pairs of nodes is too large, we randomly sample 1% pairs of nodes for the evaluation, and report the average results from 10 repeated trials.

5.2.1 Comparison with other graph embedding methods. We report only the methods including social networks (setting S and S&M), since the social proximity in the old social network is indeed the primary predictor of new friendships (see Figure 4(a) for more details) and since we found out that methods using setting M perform very poorly on this task. Figure 8 shows the results. We clearly observe that LBSN2Vec achieves the highest precision, recall and F1-score in general. For example, we find that LBSN2Vec outperforms all baselines with an average improvement of 32.95% on precision@10 over the best baselines across different datasets.

One interesting observation is that for all the baseline methods, considering both the social network and mobility (setting S&M) leads to worse performance than considering only the social network (setting S). In other words, for all baseline methods, adding user mobility information decreases the friendship prediction performance. Despite the counter-intuitive nature of this observation, we notice that these methods fail to balance the impact of social network and user mobility on the learnt node embeddings. More precisely, as the number of check-ins is usually much larger than the number of friendships, the S&M graph is dominated by the edges representing check-ins. When the baseline methods uniformly sample edges from the S&M graph to learn the node embeddings, the sampled edges are also dominated by the edges representing check-ins, which naturally imposes a strong impact of mobility on the learnt node embeddings. In fact, as mentioned for Figure 4(a) in Section 3.2, the social proximity in the old social network is indeed the primary predictor of new friendships (our parameter sensitivity study in Section 5.4 also verifies this point below). Therefore, the learnt node embeddings from those baseline methods (with setting M) result in the worst performance in friendship prediction.

5.2.2 Comparison with hand-crafted features. We compare LBSN2Vec with the following hand-crafted features suggested specifically for the friendship prediction task by previous work:

- *Katz Index (**Katz**)* [17] is the best performing social feature suggested by [33]. It is defined as the weighted sum of all paths between two users on the social networks, where the weight of a path decays exponentially with its length. $Katz(u,v) = \sum_{l=1}^{\infty} \beta^l \cdot |path^l_{u,v}|$, where $path^l_{u,v}$ is the set of all paths with length l from u to v. We set $\beta = 0.05$ as suggested by [33].
- *Adamic-Adar (**AA**)* [1] is the best performing social feature suggested by [28]. It is the weighted sum of the common neighbors between two users. The weight for each common neighbor is the inverse logarithm of its degree. $AA(u,v) = \sum_{z \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log |\Gamma(z)|}$, where $\Gamma(u)$ is the set of neighbors (friends) of user u.
- *Spatial Cosine Similarity (**SCos**)* is the best performing mobility feature suggested by [33]. It is defined as the cosine similarity between two users' check-in vectors C_u and C_v, i.e., $\cos(C_u, C_v)$, where each element $C_u(i)$ is u's check-in count on POI i.
- *Minimum place entropy (**Min_ent**)* is the best performing mobility feature suggested by [28]. It is defined as the minimum POI entropy (i.e., the entropy of the empirical distribution of check-ins at that POI over users) that two users share.

As combining social and mobility features is suggested by both [33] and [28], we consider all the combinations of the above social and mobility features, including **Katz+SCos**, **Katz+Min_ent**, **AA+Min_ent** and **AA+SCos**. Specifically, as these features have different value ranges, we first generate one ranking list of predicted friendships using each feature, and then merge two ranking

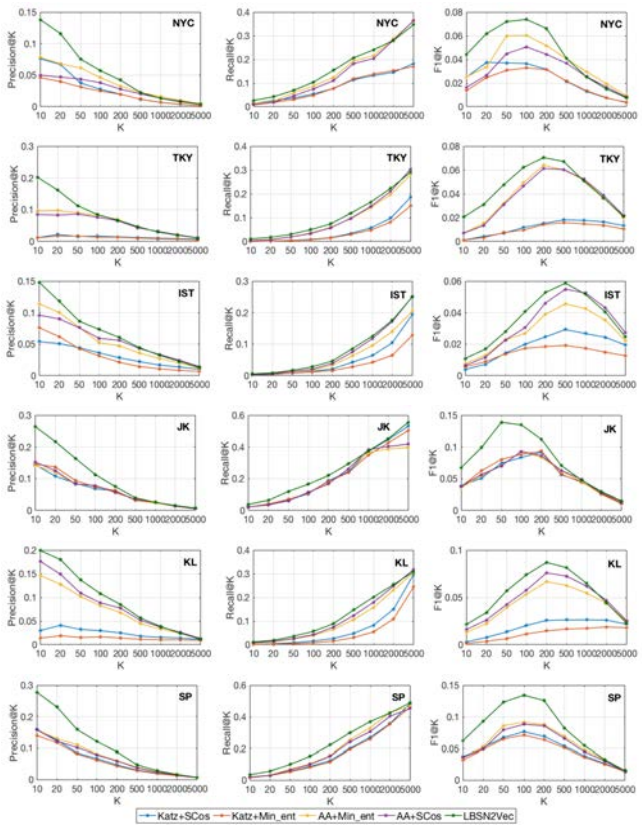


Figure 9: Friendship prediction performance comparison with hand-crafted features.

lists using reciprocal rank fusion [9], which is a popular rank fusion method widely used in information retrieval research. Figure 9 shows the results. We clearly observe that LBSN2Vec outperforms the hand-crafted feature based methods in general, and we find an average improvement of 66.44% on precision@10 over the best baselines across different datasets. Moreover, We observe that none of the four combinations can consistently outperform others over all datasets, which shows the generalization limitations of the hand crafted features across different datasets.

5.3 Location Prediction

Location prediction tries to predict the POI a user will be located in at a given time slot. To implement this task, we chronologically split our check-in data into two parts, i.e., the first 80% for training and the last 20% for testing. We then learn node embeddings from the old social network and the training check-in data. Based on the learnt node embeddings, for each test check-in (containing a user, a time slot and a POI), we rank all POIs according to the sum of two cosine similarities: one between the node embeddings of the user and a candidate POI, and one between the node embeddings of the time slot and the candidate POI. For DHNE, we use its own similarity function for ranking POIs. Subsequently, we evaluate the obtained POI ranking list against the actual POI in the test check-in. We report the average accuracy@10 over all test check-ins.

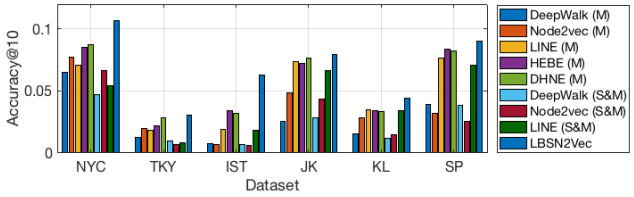


Figure 10: Location prediction performance comparison with other graph embedding methods on all test check-ins

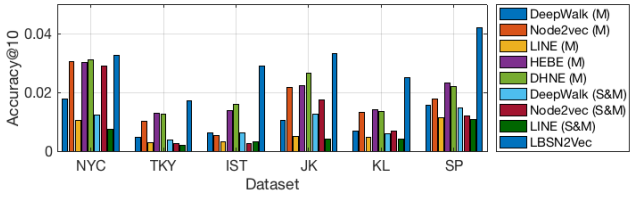


Figure 11: Location prediction performance comparison with other graph embedding methods on new POIs.

5.3.1 Comparison with other graph embedding methods. Figure 10 shows the results. Note that only settings involving check-in data (setting M and S&M) are eligible for this task. First, we observe that our LBSN2Vec consistently outperforms all baselines with an average improvement of 25.32% on accuracy@10 over the best baselines across different datasets. Second, we see that using setting M, hypergraph embedding techniques (HEBE and DHNE) achieve better performance than classical graph embedding techniques (preserving only pairwise node proximity) in general, as the irreversible process of transforming a hypergraph into a classical graph causes a certain information loss, leading to degraded performance for the location prediction task. Finally, similar to friendship prediction, we also find that for all baselines considering both the social network and mobility (setting S&M) results in worse performance than considering mobility (M) only, as the baseline methods fail to balance the impact of the social network and user mobility on the learnt node embeddings. However, the performance drop is relatively smaller than that of the friendship prediction task, as the S&M graph is actually dominated by the edges from the check-ins.

In addition, we also investigate the prediction performance on new POIs only, i.e., the POIs appearing in the test check-ins but not in the training check-ins. As historical (training) check-ins of a user are represented as hyperedges in the LBSN hypergraph, check-ins with new POIs actually refer to the non-observed hyperedges that will be established in the future. Therefore, this task is more difficult as it requires to explore the LBSN hypergraph structure for predicting potential hyperedges, rather than preserving the observed hypergraph structure. Figure 11 shows the accuracy@10 for only new POIs. We observe that compared to the case of all testing POIs, the improvement of our method over the baselines are much higher for new POIs. We observe an average improvement of 50.17% on accuracy@10 in the case of only new POIs, compared to an average improvement of 25.32% in the case of all test POIs. Such an observation demonstrates the strong advantage of LBSN2Vec in predicting potential check-in hyperedges.

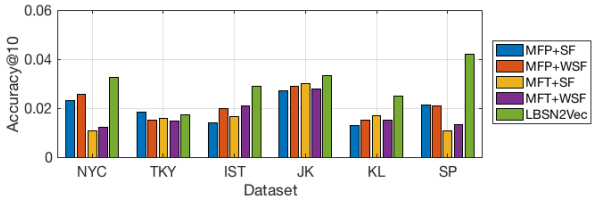


Figure 12: Location prediction performance comparison with hand-craft features on new POIs.

5.3.2 Comparison with hand-crafted features. We also compare LBSN2Vec with the following hand-crafted features suggested specifically for the location prediction task by previous work:

- *Most Frequent POI (**MFP**)* is the best performing mobility feature suggested by [21]. For each user, it ranks a POI according to the number of the user's check-ins at that POI in the training dataset.
- *Most Frequent Time (**MFT**)* is suggested by [40]. For each user and each time slot, it ranks a POI according to the number of user's check-ins at that POI and at that time slot in the training dataset.
- *Social Filtering (**SF**)* is suggested by [21]. For a user u and his friends $\Gamma(u)$, it ranks a POI according to the total number of check-ins that any friend v ($v \in \Gamma(u)$) of the user has performed at the POI.
- *Weighted Social Filtering (**WSF**)* is suggested by [12]. It is a weighted version of SF, where each friend's check-in count is weighed by the mobility similarity between the user and that friend. It ranks a POI according to the weighted sum of the number of check-ins that any her friend v, $v \in \Gamma(u)$, has performed at that POI.

Similar to the friendship prediction task, we combine the results from mobility and social features using reciprocal rank fusion [9], resulting in four combinations (i.e., **MFP+SF**, **MFP+WSF**, **MFT+SF**, **MFT+WSF**). Figure 12 shows the results on new POIs. We observe that our method outperforms the hand-crafted feature in most cases (expect for the TKY dataset where MFP+SF is slightly better). In addition, similar to the case of friendship prediction, none of the four combinations consistently achieve higher accuracy than others, showing the generalization limitations of hand-crafted features.

5.4 Parameter Sensitivity Study

5.4.1 Balance trade-off between social and mobility with α. We investigate the trade-off between the social network and mobility patterns on both the friendship and location prediction tasks by varying α within $[0, 1]$ with a step of 0.1. A small value of α gives more importance on the social network and less on check-in data when learning node embeddings, and vice versa. Figure 13 shows the results. We see a clear trade-off between the social network and mobility on all datasets. On one hand, when increasing α from 0, the friendship prediction performance slightly increases and reaches its peak around $\alpha = 0.2$, as considering mobility factors indeed helps the friendship prediction task. When further increasing α, the performance start to decrease, and we observe a sharp drop around $\alpha = 0.5$. On the other hand, when decreasing α from 1, the location prediction performance slightly increases and reaches its

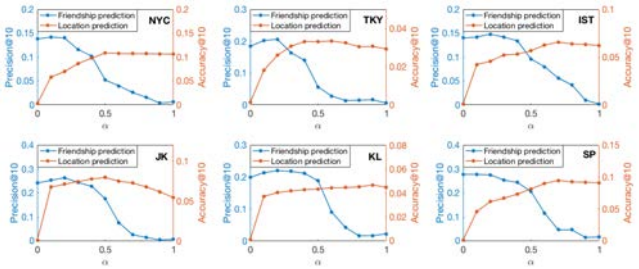


Figure 13: Impact of α on friendship and location prediction tasks

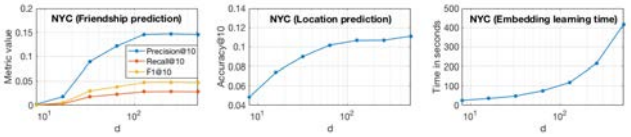


Figure 14: Impact of the node embedding dimension d on the NYC dataset

peak around $\alpha = 0.6$ in most cases (expect for KL dataset), meaning that considering social factors can also help the location prediction task. When further decreasing α, the performance start to decrease, but we observe a sharp drop around $\alpha = 0.1$ or 0.2. Interestingly, by comparing the values of α giving the peak performance in the two tasks, we find the asymmetric impact of social and mobility factor on each other, which seems to be universal across different datasets. More precisely, combining 80% social with 20% mobility data results in the best performance for the friendship prediction task, while combining 40% mobility with 60% social data gives the best performance for the location prediction task.

5.4.2 Dimension of the node embeddings d. We study the impact of the node embedding dimension d on both the friendship and location prediction tasks, as well as on the node embedding learning time on a commodity PC (Intel Core i7-4770HQ@2.20GHz, 16GB RAM, Mac OS X). Figure 14 shows the results on the NYC dataset. First, we see that a small d leads to poor performance on both tasks, as the resulting node embeddings cannot capture sufficient information from the LBSN hypergraph. Then, the performance increases with increasing d, and converges around $d = 128$, meaning that further increasing d leads to little performance improvement. In contrast, the embedding learning time always increases with increasing d, as a higher dimension of node embedding requires more computation in the node embedding learning process. Therefore, we select $d = 128$ for all previous experiments.

6 CONCLUSION

In this paper, by revisiting user mobility and social relationships in LBSNs, we propose LBSN2Vec, a hypergraph embedding approach designed specifically for LBSN data. First, by collecting a large-scale LBSN dataset over a long-term period, we conducted a rigorous empirical analysis to reveal the correlations between user check-ins and the corresponding social network. Then, we designed LBSN2Vec to learn node embeddings from the LBSN hypergraph. It performs random-walk-with-stay to jointly sample user mobility patterns and social relationships from the LBSN hypergraph,

and then learns node embeddings from the sampled hyperedges by preserving the n-wise node proximity captured by a hyperedge containing n nodes. Using the collected LBSN data, our extensive evaluation shows that LBSN2Vec consistently outperforms the state-of-the-art graph embedding methods on both the friendship and location prediction tasks, with an average improvement of 32.95% and 25.32%, respectively. Moreover, using LBSN2Vec, we discover the asymmetric impact of mobility and social relationships on predicting each other, which can serve as guidelines for future research on friendship and location prediction in LBSNs.

In the future, we plan to extend LBSN2Vec by further considering sequential patterns in the user check-in data.

ACKNOWLEDGMENTS

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement 683253/GraphInt).

REFERENCES

[1] Lada A Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social networks* 25, 3 (2003), 211–230.
[2] Amin Bahmanian and Mike Newman. 2016. Embedding Factorizations for 3-Uniform Hypergraphs II: r-Factorizations into s-Factorizations. *The Electronic Journal of Combinatorics* 23, 2 (2016), 2–42.
[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.
[4] Matthew Brand. 2003. Fast online svd revisions for lightweight recommender systems. In *SDM*. SIAM, 37–46.
[5] Hongyun Cai, Vincent W Zheng, and Kevin Chang. 2018. A comprehensive survey of graph embedding: problems, techniques and applications. *IEEE Transactions on Knowledge and Data Engineering* (2018).
[6] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. Grarep: Learning graph representations with global structural information. In *CIKM*. ACM, 891–900.
[7] Ran Cheng, Jun Pang, and Yang Zhang. 2015. Inferring friendship from check-in data of location-based social networks. In *ASONAM*. ACM, 1284–1291.
[8] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *KDD*. ACM, 1082–1090.
[9] Gordon V Cormack, Charles LA Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *SIGIR*. ACM, 758–759.
[10] Ruifeng Ding and Zhenzhong Chen. 2018. RecNet: a deep neural network for personalized POI recommendation in location-based social networks. *International Journal of Geographical Information Science* (2018), 1–18.
[11] Shanshan Feng, Gao Cong, Bo An, and Yeow Meng Chee. 2017. POI2Vec: Geographical Latent Representation for Predicting Future Visitors.. In *AAAI*. 102–108.
[12] Huiji Gao, Jiliang Tang, and Huan Liu. 2012. Exploring social-historical ties on location-based social networks.. In *ICWSM*.
[13] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*. ACM, 855–864.
[14] Huan Gui, Jialu Liu, Fangbo Tao, Meng Jiang, Brandon Norick, and Jiawei Han. 2016. Large-scale embedding learning in heterogeneous event data. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 907–912.
[15] John Hopcroft and Ravi Kannan. 2014. *Foundations of data science*.
[16] Rana Hussein, Dingqi Yang, and Philippe Cudré-Mauroux. 2018. Are Meta-Paths Necessary?: Revisiting Heterogeneous Graph Embeddings. In *CIKM*. ACM, 437–446.
[17] Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (1953), 39–43.
[18] Defu Lian, Xing Xie, Vincent W Zheng, Nicholas Jing Yuan, Fuzheng Zhang, and Enhong Chen. 2015. CEPR: A collaborative exploration and periodically returning model for location prediction. *TIST* 6, 1 (2015), 8.
[19] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *journal of the Association for Information Science and Technology* 58, 7 (2007), 1019–1031.
[20] Linyuan Lü and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications* 390, 6 (2011), 1150–1170.
[21] Anastasios Noulas, Salvatore Scellato, Neal Lathia, and Cecilia Mascolo. 2012. Mining user mobility features for next place prediction in location-based services. In *ICDM*. IEEE, 1038–1043.
[22] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *KDD*. ACM, 1105–1114.
[23] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*. ACM, 701–710.
[24] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *WSDM*. ACM, 459–467.
[25] Adam Sadilek, Henry Kautz, and Jeffrey P Bigham. 2012. Finding your friends and following them to where you are. In *WSDM*. ACM, 723–732.
[26] Salvatore Scellato, Cecilia Mascolo, Mirco Musolesi, and Vito Latora. 2010. Distance Matters: Geo-social Metrics for Online Social Networks.. In *WOSN*.
[27] Salvatore Scellato, Anastasios Noulas, Renaud Lambiotte, and Cecilia Mascolo. 2011. Socio-spatial properties of online location-based social networks. *ICWSM* 11 (2011), 329–336.
[28] Salvatore Scellato, Anastasios Noulas, and Cecilia Mascolo. 2011. Exploiting place features in link prediction on location-based social networks. In *KDD*. ACM, 1046–1054.
[29] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW*. ACM, 1067–1077.
[30] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2018. VERSE: Versatile Graph Embeddings from Similarity Measures. In *WWW*. International World Wide Web Conferences Steering Committee, 539–548.
[31] Ke Tu, Peng Cui, Xiao Wang, Fei Wang, and Wenwu Zhu. 2017. Structural Deep Embedding for Hyper-Networks. In *AAAI*.
[32] Jorge Valverde-Rebaza, Mathieu Roche, Pascal Poncelet, and Alneu de Andrade Lopes. 2016. Exploiting social and mobility patterns for friendship prediction in location-based social networks. In *ICPR*. IEEE, 2526–2531.
[33] Dashun Wang, Dino Pedreschi, Chaoming Song, Fosca Giannotti, and Albert-Laszlo Barabasi. 2011. Human mobility, social ties, and link prediction. In *KDD*. Acm, 1100–1108.
[34] Yan Wang, Zongxu Qin, Jun Pang, Yang Zhang, and Jin Xin. 2017. Semantic Annotation for Places in LBSN through Graph Embedding. In *CIKM*. ACM, 2343–2346.
[35] Min Xie, Hongzhi Yin, Hao Wang, Fanjiang Xu, Weitong Chen, and Sen Wang. 2016. Learning graph-based poi embedding for location-based recommendation. In *CIKM*. ACM, 15–24.
[36] Dingqi Yang, Daqing Zhang, Longbiao Chen, and Bingqing Qu. 2015. NationTelescope: Monitoring and visualizing large-scale collective behavior in LBSNs. *Journal of Network and Computer Applications* 55 (2015), 170–180.
[37] Dingqi Yang, Daqing Zhang, and Bingqing Qu. 2016. Participatory cultural mapping based on collective behavior data in location-based social networks. *TIST* 7, 3 (2016), 30.
[38] Dingqi Yang, Daqing Zhang, Zhiyong Yu, and Zhu Wang. 2013. A sentiment-enhanced personalized location recommendation system. In *HT*. ACM, 119–128.
[39] Dingqi Yang, Daqing Zhang, Zhiyong Yu, and Zhiwen Yu. 2013. Fine-grained preference-aware location search leveraging crowdsourced digital footprints from LBSNs. In *UbiComp*. ACM, 479–488.
[40] Dingqi Yang, Daqing Zhang, Vincent W Zheng, and Zhiyong Yu. 2015. Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs. *TSMC* 45, 1 (2015), 129–142.
[41] Zhiwen Yu, Huang Xu, Zhe Yang, and Bin Guo. 2016. Personalized travel package with multi-point-of-interest recommendation based on crowdsourced user footprints. *IEEE Transactions on Human-Machine Systems* 46, 1 (2016), 151–158.
[42] Yang Zhang and Jun Pang. 2015. Distance and friendship: A distance-based model for link prediction in social networks. In *APWeb*. Springer, 55–66.
[43] Wayne Xin Zhao, Feifan Fan, Ji-Rong Wen, and Edward Y Chang. 2018. Joint Representation Learning for Location-Based Social Networks with Multi-Grained Sequential Contexts. *ACM Transactions on Knowledge Discovery from Data* 12, 2 (2018), 22.
[44] Yu Zhu, Ziyu Guan, Shulong Tan, Haifeng Liu, Deng Cai, and Xiaofei He. 2016. Heterogeneous hypergraph embedding for document recommendation. *Neurocomputing* 216 (2016), 150–162.

Scalpel-CD: Leveraging Crowdsourcing and Deep Probabilistic Modeling for Debugging Noisy Training Data

Jie Yang
University of Fribourg
Fribourg, Switzerland
jie.yang@unifr.ch

Alisa Smirnova
University of Fribourg
Fribourg, Switzerland
alisa.smirnova@unifr.ch

Dingqi Yang
University of Fribourg
Fribourg, Switzerland
dingqi.yang@unifr.ch

Gianluca Demartini
University of Queensland
St Lucia, Australia
g.demartini@uq.edu.au

Yuan Lu
ING Domestic Bank Nederland
Amsterdam, Netherlands
joyce.yuan.lu@gmail.com

Philippe Cudré-Mauroux
University of Fribourg
Fribourg, Switzerland
pcm@unifr.ch

ABSTRACT

This paper presents Scalpel-CD, a first-of-its-kind system that leverages both human and machine intelligence to debug noisy labels from the training data of machine learning systems. Our system identifies potentially wrong labels using a deep probabilistic model, which is able to infer the latent class of a high-dimensional data instance by exploiting data distributions in the underlying latent feature space. To minimize crowd efforts, it employs a data sampler which selects data instances that would benefit the most from being inspected by the crowd. The manually verified labels are then propagated to similar data instances in the original training data by exploiting the underlying data structure, thus scaling out the contribution from the crowd. Scalpel-CD is designed with a set of algorithmic solutions to automatically search for the optimal configurations for different types of training data, in terms of the underlying data structure, noise ratio, and noise types (random vs. structural). In a real deployment on multiple machine learning tasks, we demonstrate that Scalpel-CD is able to improve label quality by 12.9% with only 2.8% instances inspected by the crowd.

CCS CONCEPTS

• **Information systems** → **Crowdsourcing**; • **Computing methodologies** → **Neural networks**; **Latent variable models**.

KEYWORDS

Debugging training data; crowdsourcing; deep probabilistic models

ACM Reference Format:
Jie Yang, Alisa Smirnova, Dingqi Yang, Gianluca Demartini, Yuan Lu, and Philippe Cudré-Mauroux. 2019. Scalpel-CD: Leveraging Crowdsourcing and Deep Probabilistic Modeling for Debugging Noisy Training Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313599

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313599

1 INTRODUCTION

The success of machine learning techniques — deep learning in particular — heavily relies on the quality and quantity of labeled training data [49]. While a growing body of research has addressed the data *quantity* issue [2, 9, 15, 31, 47], relatively little work has been focused on the data *quality* issue. Due to the lack of transparency and accountability of deep learning models [11, 19, 25], incorrect labels in the training data are generally difficult to identify when the prediction goes wrong; consequently, label noise has become a main obstacle for developing, deploying, and improving deep learning models. The significance of the problem is more obvious in critical domains such as health or justice [34]. For instance, in a medical context, models trained on drug instances that are wrongly labeled as successfully *treating* a certain condition would likely produce erroneous and damaging effects. We therefore argue that debugging noisy labels is of key importance for deep learning-based systems, both in terms of performance and security.

The main existing approach in that context leverages data distributions for debugging noisy labels (see Section 5 for more discussion on that point). The basic assumption is that data points distributed close to each other are more likely to have the same label. Existing methods [1, 35, 39], however, suffer from two major limitations. First, they only model data distributions in the *low-level* feature space with oversimplified structures (e.g., bag-of-words). As for most language and vision problems, the low-level distributions learned by these methods are limited compared to the true distributions that present complex dependencies among features. The second limitation, perhaps the most important one, is that all existing methods are automatic methods that rely on machine intelligence. As a result, the performance of such methods is limited by two factors: 1) the learning capability of the method, and 2) the predictive power of data distributions for true labels, a limiting factor of any data-driven method that relies on data distributions for debugging noisy labels.

Compared to machines, humans are more reliable in justifying decisions and verifying issues [3, 27]. We, therefore, advocate a *human-in-the-loop* approach to leverage both human and machine intelligence when debugging noisy labels. In such an approach, automatic methods can be used to identify data instances with potentially wrong labels in large training data and infer their latent classes, while crowd workers can be engaged to inspect data

instances for which the automatic method are insufficient. This approach can, therefore, exploit the complementary strength of humans and machines for debugging noisy labels. Despite its obvious potential, developing such an approach is however challenging. The first challenge is to develop a machine learning model that can learn high-quality data structures to capture data distributions with higher fidelity for identifying potentially wrong labels. Second, it is important to reduce crowd efforts in debugging large training data due to the difficulty of scaling out the crowd. This requires to select not only data points for which the inference is unreliable, but also those most *representative* data points, such that the labels verified or fixed by the crowd can be propagated to similar data instances.

This paper introduces `Scalpel-CD`, a new human-in-the-loop system that takes advantage of *deep* and *probabilistic* models for inferring the latent classes and human computation for improving the model's inference. Unlike existing probabilistic methods, our deep probabilistic model adopts deep neural networks to parameterize data distributions, thus is able to model complex feature relationships in the data and learn high-quality latent features. The model therefore benefits both from the flexibility of neural networks in learning the underlying data structure and from the expressiveness of probabilistic modeling in capturing data distributions. An additional advantage, which is of key importance when involving the crowd, is that the learned data structure and the reliability of model inference can be used to select the most representative data instances that need to be inspected by humans. To do so, `Scalpel-CD` employs a data sampler that considers both data representativeness and model reliability for data selection. To involve crowd workers, `Scalpel-CD` dynamically creates micro label inspection tasks and publishes them on a popular crowdsourcing platform. The collected results are distilled through aggregation algorithms and the corrected labels are propagated to similar data instances in the training data in order to scale out the contribution from the crowd.

A key design principle of `Scalpel-CD` is that each of its components seeks an optimal tradeoff between different factors that influence system performance on different types of machine learning datasets, in terms of the underlying data structure, noise ratio, and noise types (e.g., random vs. structural). These factors include 1) the level of trust to put in the data distribution and the existing noisy labels; 2) the importance between the representativeness of the data instances and the reliability of model inference in data sampling; and 3) the efficiency and the accuracy of label propagation. We model the tradeoffs between these important system factors as system parameters that are easy to understand; furthermore, we introduce a set of algorithmic solutions that allow to easily search for the optimal system configurations on different datasets. Our system is therefore, generic in the sense that it applies to a wide range of machine learning tasks.

In summary, we make the following key contributions:

- We introduce the notion of debugging noisy training data for learning tasks via a human-in-the-loop approach;
- We present a new system architecture that orchestrates both human and machine intelligence in debugging training data;
- We propose a deep probabilistic model that identifies potentially wrong labels and a set of algorithms for scaling out the contribution of the crowd while determining optimal system parameters.

To the best of our knowledge, we are the first to combine human and machine intelligence for debugging noisy labels. Extensive evaluation on multiple real-world datasets shows that our system is able to improve label quality by 12.9% (accuracy) and 15.8% (AUC) while requiring only 2.8% data to be inspected by the crowd.

2 ARCHITECTURE

In this section, we first give an overview of `Scalpel-CD`, and then describe in more detail some of its components.

2.1 System Overview

Figure 1 presents a simplified architecture of `Scalpel-CD`. It takes as input a noisy dataset, e.g., a corpus of documents each associated with a noisy label that represents the polarity (*sentiment*) of the text. The dataset is first passed to a *deep probabilistic model*, which infers the latent class for each data instance and a high-level latent feature representation of it. Both outputs are used as input for the *data sampler*, which calculates the reliability of the model's inference and the representativeness of the data instance. The data sampler then selects representative data instances whose labels need to be inspected the most. The selected data instances, together with the noisy labels, are passed to the *micro-task manager*, which dynamically creates human computation tasks and publishes them on a crowdsourcing platform.

The noisy label for each selected data instance is then examined by multiple human workers. Once workers complete their inspection, the results are passed to a *label aggregator*. The label aggregator aggregates multiple workers' annotations to obtain the true class for each inspected data instance. The data instances, together with their verified/fixed labels are then fed into a *label propagator*, which propagates the labels to similar data instances identified through their latent features (inferred by the deep probabilistic model). The data sampler and label aggregator can be seen as down-sampling and up-sampling methods to reduce crowd annotation efforts and amplify crowd contributions, respectively. Together, they help to scale out human contributions, which is of critical importance in a human-in-the-loop approach where human workers are often the bottleneck in terms of scalability or computational efficiency.

2.2 Components

C1: Deep Probabilistic Model. The deep probabilistic model receives a noisy dataset and simultaneously infers two types of latent variables, i.e., the latent class and the latent features. The latent class represents what the model believes is the true label for the data instance. The latent features, represented as low-dimensional vectors, essentially capture the underlying data structure. They are useful in identifying representative data instances for data sampling and in identifying similar data instances for label propagation. The inference relies on both the existing noisy label and the latent features. The basic idea is that data instances distributed close to each other in the latent feature space are likely to belong to the same class. Consider for example a data instance whose surrounding neighbors are all positively labeled; it is likely to be positive also, even when its existing noisy label is actually negative. In most cases, the surrounding neighbors are partially labeled as positive and partially as negative. The deep probabilistic model is able to

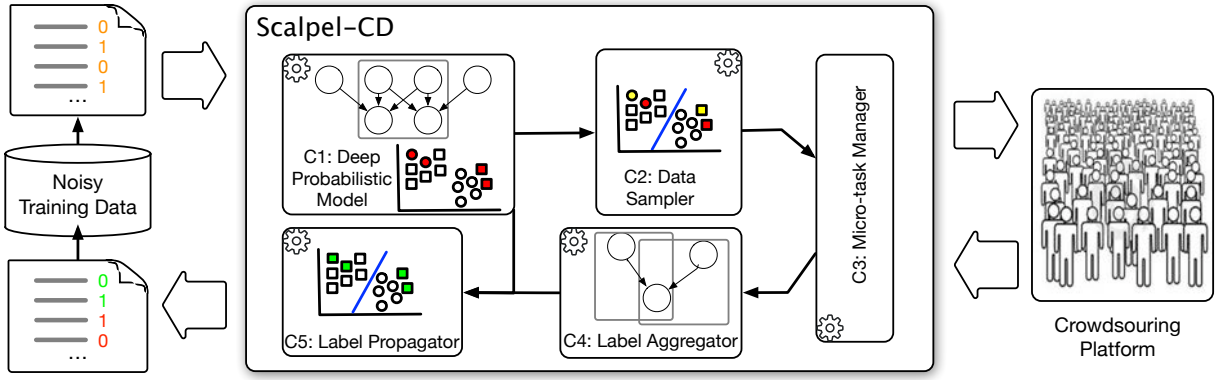


Figure 1: The architecture of `Scalpel-CD`: our system takes as input a noisy training dataset and identifies data instances with potentially wrong labels; it samples from the data representative instances and involves crowd workers to verify and fix wrong labels; the fixed labels are then propagated to similar data instances in order to scale out the contributions of the crowd.

strike a balance between the latent features and the noisy label to obtain a reasonable estimate of the latent class. This component is described in detail in Section 3.1 and 3.2.

C2: Data Sampler. The data sampler is critical in reducing the annotation effort of the crowd workers as it helps to select representative data instances for which the deep probabilistic model's inference is unreliable. The model reliability is approximated by the (inverse) uncertainty of the inference model. The representativeness of a data instance is estimated based on the underlying data structure. Specifically, the data sampler makes use of the latent features inferred by the deep probabilistic model to cluster the data instances. Within each cluster, data instances are supposed to be similar to each other. The data sampler then uses an algorithm to weight the importance of model reliability and data representativeness when selecting data. More details on the sampling algorithm are presented in Section 3.3.

C3: Micro-Task Manager. The micro-task manager is responsible for dynamically creating human computation tasks that are then published on a crowdsourcing platform. It takes as input data instances (together with their original noisy labels) selected by the data sampler, and builds a Web page to be published on the crowdsourcing platform using a templating engine. Once published, the micro-tasks can be selected by workers on the crowdsourcing platform, who are then asked to verify the noisy labels and fix them if deemed incorrect. The results are sent back to the micro-task manager, which in turns inserts them into the label aggregator.

C4: Label Aggregator. Label aggregation is a central problem in crowdsourcing. Given multiple human workers contributing to individual data instances, the label aggregator is designed to infer the true label through a probabilistic latent variable model. A large body of work can be found on this topic, e.g., the classic Dawid-Skene model [8] or the GLAD model [42] (see Section 5). Our work focuses more on addressing the scalability issue of the crowd rather than on quality issues. However, our system supports arbitrary label aggregation methods through a generic interface.

C5: Label Propagator. The label propagator transfers manually examined labels to similar data instances. This is achieved by exploiting the underlying data structure: the noisy labels of data instances identified by the deep probabilistic model are matched to the verified labels of the data instances in a cluster. For data instances with different verified labels in the same cluster, label matching is performed using a nearest neighbor search. Coupled with the data sampler, the label propagator serves as an amplifier for maximizing the utility of the crowd. The label propagator is designed to optimize two criteria: it propagates labels to as many data instances as possible while ensuring that the labels are propagated to the right instances. A detailed introduction to the propagation algorithm is given in Section 3.4.

3 DEEP PROBABILISTIC MODELING AND CROWDSOURCING TRADEOFFS

`Scalpel-CD` orchestrates human and machine intelligence via a set of components described in the previous section. These components intrinsically seek a tradeoff between different key factors of our system. In this section, we characterize the tradeoff space into three parts, relevant to the deep probabilistic model, the data sampler, and the label propagator, respectively; for each part, we introduce an algorithmic solution that simplifies the search for the optimal tradeoff as the search for a single system parameter. Note that the label aggregator is also concerned with a tradeoff: the number of crowd contributors involved and the accuracy of result aggregation. However, this has been extensively studied (see [2, 23]). In the following, we first briefly describe our deep probabilistic model, then introduce the algorithmic solutions to the three components.

3.1 Deep Probabilistic Model

To infer the latent class of a data instance, the basic intuition of our deep probabilistic model is that data instances similar to each other are more likely to belong to the same class. We consider a generative model to fully capture the data distribution of a large training dataset. Given a noisy dataset $\mathcal{D} = \{(\mathbf{x}_i, \hat{y}_i)\}_{i=1}^{N}$, where $\mathbf{x}_i$ is a data instance and $\hat{y}$ is its corresponding noisy label (we will omit the index i whenever it is clear that we are referring to a single data instance), the deep probabilistic model encompasses the generative modeling of both data and labels, described as follows.

- For each data instance $(\mathbf{x}_i, \hat{y}_i) \in \mathcal{D}$,

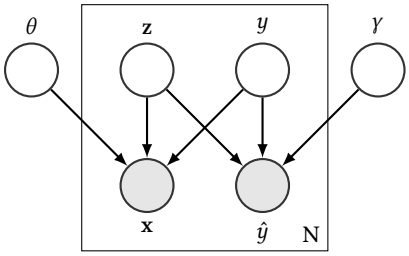


Figure 2: Graphical model of our deep probabilistic model. The data instance $\mathbf{x}$ and the noisy label $\hat{y}$ are both generated from distributions conditioned on the latent feature vector $\mathbf{z}$ and the latent class y. θ and γ are the parameters of the likelihood function for generating the data and noisy labels, respectively (priors of $\mathbf{z}$ and y are omitted).

- Draw a latent feature vector $\mathbf{z}_i \sim P(\mathbf{z})$ where $P(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$ is a standard Gaussian distribution;
- Draw a latent class $y_i \sim P(y)$ where $P(y) = \text{Cat}(y|\boldsymbol{\pi})$ is a Multinoulli distribution, where $\boldsymbol{\pi} = \{1/K, \ldots, 1/K\}^\top$ (K is the number of classes);
- Draw a data instance $\mathbf{x}_i \sim P_\theta(\mathbf{x}|\mathbf{z}, y)$;
- Draw a noisy label $\hat{y} \sim P_\gamma(\hat{y}|\mathbf{z}, y)$.

Our overall model is depicted in Figure 2. The data instance and the noisy label are both dependent on the latent feature vector $\mathbf{z}$ and the latent class y, which captures the class specification. $\mathbf{z}$ can contain both class and non-class related features. For example, for a sentence $\mathbf{x}$, the latent feature vector $\mathbf{z}$ can represent topics related to a certain class, and additionally, it can capture the author's writing style that are not class-related. The non-class related features, when mistaken as class-related, are likely to lead to wrong labels during the labeling process. $\mathbf{z}$ and y are conditionally dependent given the observed data instance $\mathbf{x}$ and the noisy label $\hat{y}$.

Formally, the deep probabilistic model is expressed by the following factorization:

$$P(\mathbf{x}, \hat{y}, \mathbf{z}, y|\theta, \gamma) = P_\theta(\mathbf{x}|\mathbf{z}, y)P_\gamma(\hat{y}|\mathbf{z}, y)P(\mathbf{z})P(y). \tag{1}$$

The likelihood functions $P_\theta(\mathbf{x}|\mathbf{z}, y)$ and $P_\gamma(\hat{y}|\mathbf{z}, y)$ are parameterized by deep neural networks to accurately capture the distributions of the data and the noisy labels. Depending on the specific form of the data, different likelihood functions are to be used for $P_\theta(\mathbf{x}|\mathbf{z}, y)$: Gaussian likelihood is suitable for image data, while Multinomial likelihood is more suitable for textual data (see [18, 29]). $P_\gamma(\hat{y}|\mathbf{z}, y)$ is represented by a Multinoulli likelihood.

Inference and Learning. The inference for the latent feature vector $\mathbf{z}$ and the latent class y is closely related to the learning of the deep probabilistic model parameters. The parameters are learned by maximizing the log likelihood of the observed data instances and the associated noisy labels:

$$\log P(\mathbf{x}, \hat{y}|\theta, \gamma) = \log \iint P(\mathbf{x}, \hat{y}, \mathbf{z}, y|\theta, \gamma)\, d\mathbf{z}dy. \tag{2}$$

Due to the intractability of the integral, we approximate the true posteriors of $\mathbf{z}$ and y with variational ones, denoted by $Q_\phi(\mathbf{z}|\mathbf{x}, y)$ and $Q_\psi(y|\mathbf{x})$, respectively. We use a Gaussian distribution for $Q_\phi(\mathbf{z}|\mathbf{x}, y)$ and a Multinoulli distribution for $Q_\psi(y|\mathbf{x})$. Given the complex dependencies among the low-level features of data instances and their relationships with the latent variables, these distributions are again parameterized by deep neural networks. The parameters, including those of the generation networks θ and γ and those of the inference networks ψ and ϕ, are then learned by maximizing the evidence lower bound (ELBO) [4] of the objective as follows:

$$\mathbb{E}_{Q_{\phi,\psi}(\mathbf{z},y|\mathbf{x})}[\log P_{\theta,\gamma}(\mathbf{x}, \hat{y}|\mathbf{z}, y)] - D_{KL}[Q_{\phi,\psi}(\mathbf{z}, y|\mathbf{x})\|P(\mathbf{z}, y)], \tag{3}$$

where $\mathbb{E}(\cdot)$ is expectation and $D_{KL}[\cdot\|\cdot]$ is the KL-divergence between two distributions.

3.2 Inferring the Latent Class

From a neural network perspective, the first part of the ELBO (Equation 3) can be interpreted as the sum of the (negative) reconstruction errors of the observed feature $\mathbf{x}$ and noisy label $\hat{y}$, while the second part is the (negative) regularization term preventing overfitting. Hence, we rely both on the data distribution and the existing noisy labels when inferring latent classes.

The original ELBO weights both types of reconstruction errors as equally important. While this can result in a powerful generative model for data and label generation, our goal is different: we are interested in inferring the latent classes, for which the data distribution and the existing noisy labels are likely to be differently important. Furthermore, their importance can vary across different datasets: for datasets whose data distributions are highly indicative of the true class and the noise ratio of the existing labels is high, the data distribution should be more trusted in the inference; otherwise, the existing label should be more trusted.

It is, therefore, natural to extend the ELBO by introducing a parameter β to weight the importance of the two types of reconstruction errors:

$$\begin{aligned}&\mathbb{E}_{Q_{\phi,\psi}(\mathbf{z},y|\mathbf{x})}[\log P_{\theta,\gamma}(\mathbf{x}, \hat{y}|\mathbf{z}, y)] = \\ &\mathbb{E}_{Q_{\phi,\psi}(\mathbf{z},y|\mathbf{x})}[\log P_\gamma(\hat{y}|\mathbf{z}, y)] + \beta \cdot \mathbb{E}_{Q_{\phi,\psi}(\mathbf{z},y|\mathbf{x})}[\log P_\theta(\mathbf{x}|\mathbf{z}, y)].\end{aligned} \tag{4}$$

In case $\beta \neq 1$, we are no longer optimizing the ELBO on the log marginal likelihood (Equation 2). If $\beta < 1$, we would trust more the existing labels than the data distribution; consequently, the model is less able to determine label noise by resorting to the data distribution. Otherwise, if $\beta > 1$, we are weakening our trust on the existing noisy labels while putting more trust on the data distribution in determining label noise. While this allows for more flexibility, it could also bring additional noise to the labels due to the non-class related features in the data distribution. Finding a proper value of β is important to achieve a good performance in inferring latent classes.

The resulting optimization algorithm is given in Algorithm 1. It iteratively goes over two steps, i.e., the forward and the backward step. In each iteration, the forward step (row 5) computes the latent variables given the current parameters; the backward step (row 6-8) then updates the parameters by backpropagating the gradients of the errors. In the calculation of the gradients (row 6), we use an adapted version of the ELBO (Equation 4).

3.3 Data Sampling

The data sampler selects representative data instances for which our deep probabilistic model's inference is most unreliable. The

Algorithm 1: Learning the Deep Probabilistic Model

Input: the set of N i.i.d. data instances $\mathcal{D} = \{\mathbf{x}_i, \hat{y}_i\}_{i=1}^{N}$, ELBO adapter β, and the maximum number of iterations $Iter$

1 Initialize ϕ, ψ, θ, γ;
2 **for** $t = 1; t \leq Iter; t++$ **do**
3 Sample a batch of data instances;
4 **forall** $\mathbf{x}_i \in$ *the batch* **do**
5 Compute y_i and $\mathbf{z}_i$;
6 Compute the noisy gradient $\Delta_{\phi,\psi,\theta,\gamma}\mathcal{L}$;
7 Average noisy gradients from batch;
8 Update ϕ, ψ, θ, γ with gradient descent;
9 **if** *ELBO has converged* **then**
10 break;

Algorithm 2: Data Sampling

Input: Data instances represented by latent features and inferred classes $\mathcal{D} = \{\mathbf{z}_i, P(y_i)\}_{i=1}^{N}$, budget B, #clusters c, validation instances with ground truth labels $\mathcal{V}$

Output: Data instances sampled $\mathcal{S} \subset \mathcal{D}$, clusters $\mathcal{C} = \{C_j\}_{j=1}^{c}$

1 $\mathcal{C} \leftarrow$ Cluster data instances $\mathcal{D}$ based on latent features;
2 $\mathcal{S} \leftarrow \emptyset$;
3 Decide the sampling strategy using $\mathcal{V}$;
4 **foreach** $C_j \in \mathcal{C}$ **do**
5 Rank data instances in C_j according to the sampling strategy;
6 $n_j \leftarrow \left\lfloor \frac{|C_j|}{N} \times B \right\rfloor$;
7 $\mathcal{S}_j \leftarrow$ Pick n_j top-ranking data instances;
8 $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{S}_j$;

model reliability is approximated by the inverse of the model's *uncertainty*, measured by Shannon entropy [13, 37]:

$$H[y|x] = -\sum_{C=1}^{K} p(y = C|x) \log p(y = C|x) \tag{5}$$

where C is the class and K is the number of classes. We hypothesize that uncertainty sampling is effective for datasets with random noise: for data instances far away from the true decision boundary, the deep probabilistic model can take advantage of the majority label of data instances and provide correct inference; however for data instances close to the decision boundary, the deep probabilistic model will have a high uncertainty due to the mix of data instances from different classes and will likely result in some wrong inference.

However, for a dataset with structural noise, since there can exist certain regions of the data distribution where the majority of the labels are incorrect, the inference of the deep probabilistic model can be totally off while being highly probable according to the model. Therefore, we also consider random sampling, which is independent of the deep probabilistic model. The effectiveness of the two sampling methods can be evaluated using a set of validation instances with ground truth labels: a better sampling method is the one from which the sampled data instances covers more validation instances where the deep probabilistic model's inference is wrong.

An additional criterion for data sampling is the representativeness of the data instances, which is important for label propagation. We apply a clustering method to the data instances in the latent feature space as uncovered by the deep probabilistic model, i.e., $\mathbf{z}_i$ $(1 \leq i \leq N)$. Data instances falling into the same clusters are supposed to be more similar with each other.

We now describe our data sampling algorithm (see Algorithm 2). It starts by clustering data instances based on their latent features (row 1), then chooses a sampling strategy between uncertainty and random sampling based on the coverage of validation instances for which model inference is wrong (row 3). Then, given a budget B, i.e., the predefined number of data instances to be inspected by the crowd, the data sampler first ranks the data instances in each cluster according to the sampling strategy (row 5). It then selects from each cluster the top-ranking data instances (row 7). The number of data instances selected in each cluster is proportional to the size of the cluster (row 6). Note that for random sampling, selecting data from clusters proportionally is equivalent to stratified random sampling.

A key parameter of the algorithm is the number of clusters, denoted by $c \leq B$. By setting this parameter to different values, the data sampler weights the importance of data representativeness and model reliability differently. Consider the extreme case when $c = 1$, i.e., no data partitioning, the data sampler fully relies on uncertainty or random sampling for selecting data instances; when $c = B$, i.e., a single data instance is selected from each cluster, the data sampler relies more on data representativeness for selecting data instances. Our algorithm is not restricted to any specific clustering methods. The only requirement is the flexibility for users to specify the number of clusters c. As an example, we consider the widely used K-means method for clustering. Comparison of different clustering methods is left for future work.

3.4 Label Propagation

The label propagator amplifies the crowd contributions by propagating corrected labels to similar data instances. The similarity between data instances is calculated as the Euclidean distance between the latent feature vectors of the instances. A key parameter of the label propagating algorithm is the number of data instances to propagate the label to. Given that the size of the clusters are different and that data instances within each cluster are similar, we use a relative number for the parameter to represent the fraction of the data instances for label propagation in each cluster. We denote this parameter as p. A small value of p will affect fewer data instances, thus being less efficient in scaling out human contributions; on the contrary, a big value of p, while being efficient, might increase the noise ratio of the dataset as labels can get wrongly propagated. A proper setting of p is, therefore, of key importance for scaling out human contributions in a reliable way.

Our label propagation algorithm is presented in Algorithm 3. It performs label propagation in each cluster identified by the data sampler. For each inspected data instance in the cluster (row 1-2), the label propagator first ranks the data instances that have not been inspected by crowd workers according to their distance to the inspected instances (row 3-5). It then picks the top ranking instances with shortest distances (row 6). Finally, for each of the selected data instances in the cluster, the label of the nearest inspected data instance is propagated (row 7-8).

Algorithm 3: Label Propagation

Input: Data instances represented by latent features $\mathcal{D} = \{\mathbf{z}_i, \hat{y}_i\}_{i=1}^{N}$, data instances with fixed labels $\mathcal{S} = \{\mathbf{z}_h, y_h\}_{h=1}^{H}$, clusters $C = \{C_j\}_{j=1}^{c}$, propagation fraction p

Output: Fixed labels $\mathcal{D} = \{y_i\}_{i=1}^{N}$

foreach $C_j \in C$ **do**
  foreach $h \in \mathcal{S} \wedge h \in C_j$ **do**
    foreach $i \in C_j \wedge i \notin \mathcal{S}$ **do**
      $d_{i,h} \leftarrow$ distance between $\mathbf{z}_h$ and $\mathbf{z}_i$;
    Rank data instance i ($i \in C_j \wedge i \notin \mathcal{S}$) by $d_{i,h}$;
    $\mathcal{P}_h \leftarrow p \times |C_j|$ top ranking instances with smallest $d_{i,h}$;
  foreach $i \in \bigcup_{h=1}^{H} \mathcal{P}_h$ **do**
    $y_i \leftarrow y_{h^*}$ where $h^* = \arg\min d_{i,h}$ ($i \in \mathcal{P}_h$);

4 EXPERIMENTS AND RESULTS

This section presents experimental results for evaluating the performance of Scalpel-CD. We start by presenting an evaluation of each of its components by answering the following questions:

- **Q1:** How well does the deep probabilistic model perform when inferring the latent class from noisy data?
- **Q2:** How well do the uncertainty and random sampling perform on datasets of different noise types?
- **Q3:** How effective is label propagation (together with data sampling) in scaling out the contributions of crowd workers?

From a human computation perspective, while we mainly consider improving the scalability of the crowdsourcing process, we also investigate the quality of the work performed by the crowd, which we report when addressing Q3. Subsequently, we evaluate the system as a whole by considering the following issue:

- **Q4:** Does our proposed Scalpel-CD system outperform the state-of-the-art methods for debugging noisy labels?

In addition, we also investigate how much Scalpel-CD helps in improving the performance of a machine learning system. In the following, we start by introducing our experimental setup, before answering each of the above questions in a separate subsection.

4.1 Experimental Setup

Datasets. We experiment with three datasets created with different noise types, namely MovieReview (random noise), PoliDying (structural noise), and NYT (structural noise). These datasets are representative for sentiment analysis, event detection, and relation extraction, respectively. The MovieReview dataset [30] is a subjectivity classification dataset of movie reviews from Rotten Tomatoes. This dataset contains ground truth labels for all data instances; we, therefore, simulate wrong labels by randomly selecting a certain fraction (10% to 50%) of the data instances and flipping the ground truth labels. The PoliDying dataset is a tweet relevance classification dataset for detecting politician-dying events. Following previous studies [20, 36], the positive tweets were retrieved with detailed information on 11 seed events, including event name and date, while negative tweets were retrieved with "politician"- and "death"-related keywords. A set of 500 tweets were manually annotated for evaluation. The NYT dataset [33] is a relation extraction dataset for extracting relations from New York Times articles. We focus on the relation "location-contains". The dataset is labeled by distant supervision using Freebase as the labeling source. In this dataset, there are 395 manually annotated data instances.

Table 1: Statistics of the datasets in our experiments.

Dataset	MovieReview	PoliDying	NYT
#Data instances	10,000	17,030	125,267
#Test instances	2,000	500	395

For each of the three datasets, the data sampler selects a number (i.e., the budget B) of data instances for crowdsourcing. We experiment with the budget varying in {50, 100, 200, 500, 1000}. Key statistics from these datasets are presented in Table 1.

Comparison Methods. Due to the lack of existing human-in-the-loop approaches in debugging training data, we compare the following automatic methods and our proposed system. 1) **Ratio** [5]: a ratio-based method that finds the most predictive features and identifies data instances with the most uncertain labels as those containing such features yet labeled differently from the label indicated by the features. The predictive power of a feature is calculated as the ratio between the number of data instances of a certain class containing such a feature and the overall number of data instances. 2) **Pattern** [39]: a generative model designed to capture the labeling process as a generative process from the latent classes. The model facilitates the inference of the posterior of the latent class given the observed features. However, unlike our deep probabilistic model that infers the latent class by exploiting the data distribution in the latent feature space, Pattern is a probabilistic model that directly models the generative process of low-level features. 3) **HierTopic** [1]: a hierarchical topic model that assumes a latent topic-word hierarchy for the data generation process. Unlike our probabilistic model which learns complex hidden data structures, HierTopic is only capable of learning a hierarchical structure. For our proposed system, we compare an automated variant and the full system involving human workers: 4) **DPM**, our proposed deep probabilistic model, which is used in isolation to automatically correct wrong labels. 5) **Scalpel-CD**, our proposed human-in-the-loop system which makes use of both the deep probabilistic model and crowd workers.

To demonstrate the effectiveness of our system on machine learning tasks, we compare the following training data configurations: 1) the original noisy training data; 2) training data denoised by the deep probabilistic model; and 3) training data denoised by Scalpel-CD. We apply and evaluate state-of-the-art machine learning models on these different data configurations: we use Convolutional Neural Network [16] for MovieReview and PoliDying and Piecewise Convolutional Neural Network (PCNN) [48] for NYT.

Parameter Settings and Model Training. We empirically set optimal parameters based on a held-out validation set that contains 10% of the test data. These include the hyperparameters of the deep probabilistic model and the parameters of the trade-off algorithms, i.e., β, c, and p. For the deep probabilistic model, we use a shared embedding layer as the first layer of the inference networks, i.e., Q_ϕ and Q_ψ, to encode the textual features. We apply a grid search in {32, 64, 128, 256, 512} for the dimension of the embeddings and that of the hidden layers, and we explore multi-layer perceptron

networks with 0, 1, and 2 hidden layers for both the generation and inference models, subject to the constraint that P_θ and Q_ϕ have a symmetric architecture (similar to an auto-encoder). We use Adam [17] for model training with mini-batches of 128 data samples for all datasets. The deep probabilistic model is trained for 200 epochs on each dataset and the one with the best performance on the validation set is kept for evaluation.

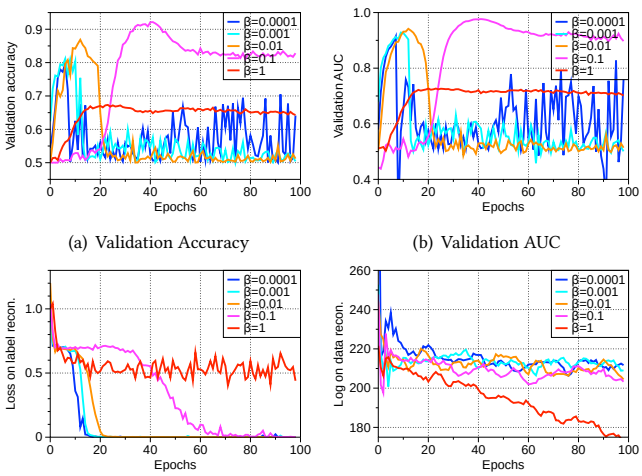
Evaluation Protocols. We separately evaluate the performance of `Scalpel-CD` on debugging noisy labels and its effect on the final machine learning tasks using different protocols. For the former task, we take the entire dataset as input. This includes the data instances in the test set together with their noisy labels. The system debugs labels in the whole dataset. The denoised labels of data instances in the test set are then compared against the ground truth. For the final machine learning tasks, our system debugs labels in the training set only. A state-of-the-art deep learning model is then trained on the denoised training set and evaluated on the test set. Two metrics are used to measure the performance of our proposed system in debugging noisy labels and its effect on the final machine learning tasks: accuracy and Area Under the ROC Curve (AUC). Higher accuracy and AUC values indicate better performance.

4.2 Inferring the Latent Class (Q1)

To understand the effectiveness of our deep probabilistic model in inferring the latent class, we analyze the relationship between capturing the data distribution and inferring the latent class. The analysis is carried out by comparing the performance of the deep probabilistic model with different settings of β, which controls the importance of the reconstruction errors of data instances and the noisy labels in the objective function. We analyze the dynamics of our deep probabilistic model on the validation set during the training process, with β selected from $\{0.0001, 0.001, 0.01, 0.1, 1\}$. Results on the MovieReview dataset with a noise ratio $NR = 20\%$ are shown in Figure 3. We discuss below the influence of different datasets and noise ratios on the performance of the model.

From Figures 3(a-b), we observe a general pattern across different settings of β: when the training proceeds, the performance of the deep probabilistic model first increases then decreases. The difference between the performance across β settings indicates that the optimal performance is hit at different performance levels and epochs during the training process: from $\beta = 0.0001$ to 0.1, the optimal performance grows and is reached later in the training process for larger values of β; then from $\beta = 0.1$ to 1, the optimal performance decreases and is reached earlier. Given such results, we speculate that $\beta = 0.1$ best infers the latent classes as it strikes a balance between minimizing the reconstruction errors of the data and the noisy labels. In addition, we further hypothesize that there is a consistent relationship between the level of optimal performance and the time at which it is reached in the training process.

These hypotheses are confirmed by Figures 3(c-d). For small values of β ($0.0001-0.01$), the deep probabilistic model quickly reaches the minimum reconstruction error of noisy labels. As a result, the inferred latent classes are biased towards existing noisy labels, which is evidenced by the optimal accuracy of $\beta = \{0.0001, 0.001\}$ being only slightly higher than 0.8 (Figure 3(a); the performance of $\beta = 0.01$ is slightly higher yet still not optimal). For $\beta = 1$, the



(a) Validation Accuracy (b) Validation AUC

(c) Loss on Label Reconstruction (d) Loss on Data Reconstruction

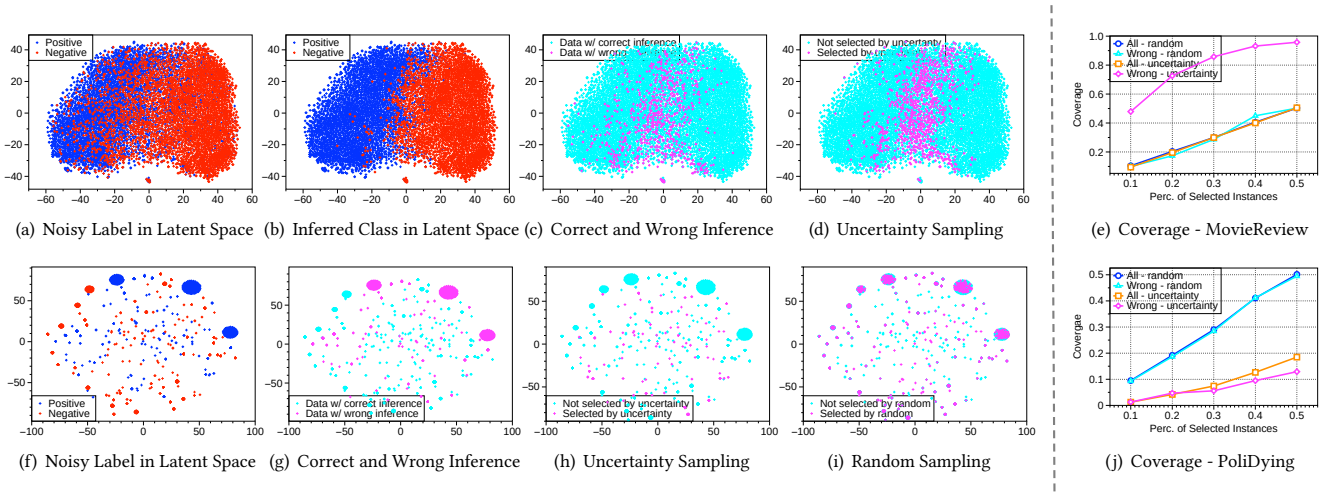
Figure 3: Comparison of performance against loss in data and label reconstruction with different settings of β on the MovieReview dataset. Upper figures are the dynamics of accuracy (a) and AUC (b) on the validation set during the training process; lower figures are the reconstruction loss (negative log-likelihood) of noisy labels (c) and data instances (d).

reconstruction error of the data is significantly reduced (more than for other settings, see Figure 3(d)); the reconstruction error of noisy labels is however not reduced as much as in the other settings (Figure 3(c)). The inferred latent classes therefore cannot benefit from those existing correct labels. To infer the latent class in an unbiased manner, sufficient model training has to be performed to reduce both types of reconstruction errors. This implies that the optimal setting of β is most likely the one that takes most of the time to reach the optimal performance during the training process.

Impact of Datasets and Noise Ratios. We observe similar results with the other datasets. The optimal settings of β are 1 and 0.01 for PoliDying and NYT, respectively. The variation across datasets implies different utility of the data distribution in inferring the latent class, which highlights the importance of searching for the optimal β separately for different datasets. Actual performance using these β values will be reported in Section 4.5. We note that our model is robust with respect to different noise ratios. It achieves accuracy scores > 0.9 even when the noise ratio of MovieReview goes up to 40%. Compared with noise ratios, model performance is more affected by the predictive power of the data distribution. We remark, however, that the model can always benefit from the data distribution and achieve better performance than $1 - NR$.

4.3 Uncertainty and Random Sampling (Q2)

The data sampling component of `Scalpel-CD` allows for two sampling strategies, i.e., uncertainty sampling and random sampling, which are model-dependent and model-independent, respectively. We therefore hypothesize that random sampling is more effective than uncertainty sampling for datasets with structural noise. To verify our hypothesis, we conduct a comparative analysis between the MovieReviews dataset and the PoliDying and NYT datasets,



(a) Noisy Label in Latent Space (b) Inferred Class in Latent Space (c) Correct and Wrong Inference (d) Uncertainty Sampling (e) Coverage - MovieReview

(f) Noisy Label in Latent Space (g) Correct and Wrong Inference (h) Uncertainty Sampling (i) Random Sampling (j) Coverage - PoliDying

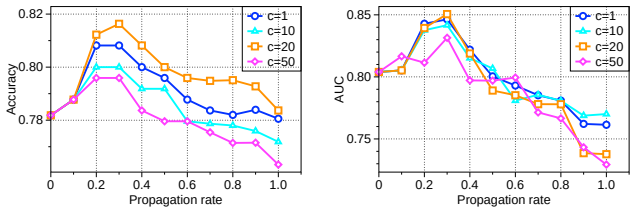
Figure 4: Effects of uncertainty and random sampling on the MovieReview dataset (upper figures) and the PoliDying dataset (lower figures). From left to right, upper figures are the visualization of the noisy label (a) and the inferred latent class (b) in the two dimensional latent feature space; visualization of the data instances with correct and wrong inference of the latent class (c) and those selected (top-10%) and not selected by uncertainty sampling (d). Lower figures are the visualization of the noisy labels (f) and the data instances with correct and incorrect inference of the latent class (g) in the latent feature space; visualization of the (top-)10% selected and not selected data instances by uncertainty sampling (h) and random sampling (i). Figures on the right part are the coverage of the uncertainty sampling and random sampling on the entire validation set and the subset where the inference is wrong for the MovieReview dataset (e) and the PoliDying dataset (j).

as the two types of noisy datasets. The comparison is carried out by 1) visually locating in the latent feature space the selected data instances using the two sampling strategies and the data instances where the inference of the latent classes is wrong, and 2) statistically comparing the coverage of the selected data instances on data instances where the inference is wrong. (For rigorousness, we use the validation set for comparison, however similar results are observed in the test set.)

We start by investigating the effectiveness of uncertainty sampling on the MovieReview dataset. We apply t-SNE [26] to embed the latent features of the data instances into a two-dimensional space, and then visualize the noisy label and the inferred latent class in Figures 4(a-b). Given our previous results (high performance of our deep probabilistic model in latent class inference in Figures 3(a-b)), we observe from Figures 4(a-b) that the inferred latent features are highly indicative of the latent class: data instances of different classes are located separately in the latent feature space. Figure 4(c) then shows the data instances for which the inference is correct and wrong in two colors, respectively. We observe that our model can effectively recover the true classes of data instances located far away from the model's decision boundary. In contrast, the majority of the data instances with wrong class inference are located close to the decision boundary. This is due to the fact that at the decision boundary, data instances of different classes are mixed with each other, leading to high model uncertainty. This is confirmed by Figure 4(d), which visualizes the top-10% data instances selected by uncertainty sampling. Comparing Figures 4(c) and 4(d), we observe that uncertainty sampling is a highly effective method in selecting data instances when the model inference is wrong.

For comparison, we show the effect of uncertainty sampling and random sampling on the PoliDying dataset created with structural noise. Figure 4(f) visualizes the noisy labels in the latent feature space and 4(g) shows the data instances with correct and wrong inference in two colors. We can observe that the deep probabilistic model can be totally wrong in its inference for a set of data instances close with each other (i.e., those about the same topic), e.g., the red clusters in Figure 4(g). Comparing such a result to Figure 4(f), we observe that the noisy labels for these data instances are almost all incorrect. These highly consensual yet wrong labels not only make the inference wrong, but also lead to high model confidence (i.e., low uncertainty). This is validated by Figure 4(h), which shows that uncertainty sampling can hardly identify data instances with structural errors. In contrast, Figure 4(i) shows a similar number of data instances (both 10%) selected by random sampling, which is able to cover data instances with wrong inference scattered at different regions of the latent feature space.

Impact of Noise Types. To quantitatively compare uncertainty and random sampling, Figures 4(e,j) show for both datasets the coverage of the selected data instances (from 10% to 50%) on the entire validation set and the subset where the model inference is wrong. On one hand, we observe that for MovieReview, uncertainty sampling is highly capable of selecting those data instances with wrong inference (80% covered by 20% selected instances). On the other hand, random sampling performs much better for PoliDying. These results highlight the distinct capabilities of uncertainty sampling at tackling random noise and of random sampling at tackling structural noise. Finally, we note that for the NYT dataset the two sampling strategies perform comparably well. This is likely due to the variety of the entity pairs (tens of thousands) in the external database used for labeling, which comes in contrast to the PoliDying dataset that uses a limited number of seed events and keywords as labeling sources.



(a) Expert - Accuracy (b) Expert - AUC

(c) Crowd - Accuracy (d) Crowd - AUC

Figure 5: Effect of label propagation with different settings of the propagation rate and the number of clusters on the PolyDying dataset. Upper figures are the results with expert-contributed labels measured by accuracy (a) and AUC (b); lower figures are the results with crowd-contributed labels measured by accuracy (c) and AUC (d).

4.4 Scaling out Human Contributions (Q3)

We now investigate the effectiveness of data sampling and label propagation for scaling out human contributions. The relevant system parameters are the number of clusters c, which controls the importance of the representativeness of data instances in data sampling, and the fraction of data instances p in each cluster to be considered for label propagation. We analyze the impact of different settings of p from 0 to 1 with a step of 0.1, and c selected from {1, 10, 20, 50}. Furthermore, we conduct a comparative study between labels verified by experts and crowd workers to understand the impact of imperfect human work. We report results on the PoliDying dataset with a budget $B = 500$. Similar results are observed on the NYT dataset; results for the MovieReview dataset and the impact of different budgets will be analyzed below.

Figures 5(a-b) show the results of expert verified labels. When p increases from 0 to 1, we observe inverse U-shaped curves across all different settings of c and the two metrics: the performance first increases then decreases, with optimal performance reached at $p = 0.3$. This indicates that label propagation does help in correcting more data instances not inspected by experts. The performance decreases from $p = 0.3$ to 1 since the propagation accuracy decreases when labels are propagated to data instances semantically different from the manually verified ones. Comparing the different settings of c, we observe that label propagation reaches its best performance at $c = 20$, which significantly outperforms other settings. This confirms the importance of finding an optimal setting for c to make the best partitioning of the data and the optimal tradeoff between the different criteria for data sampling.

For our crowdsourcing experiment, we solicit from the crowd three labels for each data instance. Results show that crowd labels match expert labels on 84.73% data instances. With such an accuracy, we do not observe consistent patterns of performance variation in label propagation. A simple filtering of data instances where all three crowd workers agree results in 81.50% remaining data instances, whose labels match expert labels 90.20% of the time. Figures 5(c-d) show the label propagation results on the filtered labels. We observe similar patterns with expert labels (Figures 5 (a-b)). Overall, label propagation enhances system performance by 3.5% (accuracy) and 4.6% (AUC) for expert labels and by 2.1% (accuracy) and 3.8% (AUC) for crowd labels.

Impact of Budgets and Datasets. We observe similar results for budget $B = 1000$ on both the PoliDying and NYT datasets. However, for $B <= 200$ no explicit patterns are observed. This is likely due to the under-representativenes of the inspected data instances. For MovieReview, crowd contributions improve the resulting accuracy from 0.925 to 0.956 and the AUC from 0.977 to 0.987. However, the performance gets worse for $p >= 0.1$. This is mainly due to the effectiveness of the deep probabilistic model in making use of data distributions to correct erroneous labels in a dataset with random noise (see Figures 3(a-b) and Section 4.5), which makes it difficult to further improve label quality through label propagation.

4.5 Performance on Debugging Labels (Q4)

We now compare the performance of our proposed system to state-of-the-art methods for debugging noisy labels. Table 2 ("Debugging") reports their performance on our three datasets. From these results, we make the following observations.

Ratio works well on the MovieReview dataset, however fails on PoliDying and NYT (compared to the original labels). Pattern, as a probabilistic method, is more robust: it improves label quality for both MovieReview and PoliDying. Yet, it still fails on the NYT dataset. Recall that both methods use the low-level features (i.e., words) for label inference. These results indicate that probabilistic methods that capture data distributions based on low-level features do not necessarily work for debugging noisy labels. In contrast, HierTopic, which models a hierarchical structure from the data using a probabilistic framework, consistently improves the label quality across all datasets. These results confirm the importance of considering data structures in debugging noisy labels.

Our proposed deep probabilistic model outperforms all the other methods by a large margin: it yields an average improvement of 7.0% in accuracy and 3.9% in AUC compared to HierTopic; compared with the original noisy labels, it improves the label quality by 9.1% in accuracy and 11.8% in AUC. These results clearly indicate the effectiveness of our model in debugging noisy labels. In particular, the improvements over HierTopic shows that deep neural networks are effective in learning the underlying data structure. Furthermore, the effectiveness of the model on PolyDying and NYT indicates that, overall, the model works well for datasets with structural noise, even if the inference can be unreliable for data instances distributed in certain regions of the latent feature space (Section 4.3).

`Scalpel-CD` achieves the best performance across all datasets. It outperforms the deep probabilistic model by 3.8% in accuracy and 3.9% in AUC on average. This highlights the value of human input in the process. Compared to the original noisy labels, our system improves label quality by 12.9% in accuracy and 15.8% in AUC, with only 2.8% data instances manually inspected by the crowd.

Table 2: Performance of the compared methods on two settings: debugging noisy labels and machine learning. Results on the MovieReview dataset are reported for $NR = 20\%$. Results of our proposed systems are obtained with a budget $B = 500$. The best performance for each dataset in each setting is boldfaced; the runner up is labelled with "*".

	Methods	MovieReview		PoliDying		NYT	
		Acc.	AUC	Acc.	AUC	Acc.	AUC
Debugging	Original labels	0.800	0.800	0.625	0.633	0.701	0.722
	Ratio	0.891	0.949	0.615	0.616	0.605	0.664
	Pattern	0.849	0.897	0.669	0.672	0.631	0.646
	HierTopic	0.852	0.929	0.629	0.693	0.706	0.770
	DPM	0.925*	0.977*	0.742*	0.742*	0.731*	0.791*
	Scalpel-CD	**0.956**	**0.987**	**0.812**	**0.831**	**0.745**	**0.810**
Learning	SOTA	0.798	0.902	0.608	0.670	0.665	0.733
	W/ DPM	0.845*	0.913*	0.746*	0.793*	0.690*	0.754*
	W/ Scalpel-CD	**0.883**	**0.949**	**0.828**	**0.896**	**0.719**	**0.765**

Impact on Learning Tasks. The benefits of using our system on machine learning tasks is shown in Table 2 ("Learning"). The performance of state-of-the-art learning methods improves significantly when using our deep probabilistic model or our system as a whole. On average, machine learning performance is improved by 12.0% in accuracy and 10.2% in AUC using Scalpel-CD. These results clearly highlight the benefits of debugging noisy labels and the effectiveness of Scalpel-CD on machine learning tasks.

5 RELATED WORK

This section gives an overview of relevant work from the emerging field of human-in-the-loop paradigm for artificial intelligence (HITL-AI) and from label noise reduction for training data.

HITL-AI. Human computation and machine learning are increasingly interweaving with each other [41]. The main intersection is crowdsourced training data creation, e.g., the ImageNet dataset for computer vision [10] and many crowd annotated datasets for various natural language processing tasks [14, 28]. A closely related line of work in the machine learning community is "learning from crowds", where researchers study models that can learn from noisy labels contributed by crowds [32, 40, 46, 47]. Unlike the conventional learning setting, models developed in this context are concerned with learning parameters of the annotation process (e.g., annotator expertise [32] and task difficulty [46]) and inferring true labels from noisy ones. Such an idea has been extended to incorporate (deep) active learning to reduce annotation efforts [12, 40, 45, 47].

In contrast to creating high-quality training data from scratch, e.g., through active learning from the crowd, fast creation of training data with noise tolerance is becoming more popular [2, 31]. Existing work on debugging machine learning systems mainly focuses on identifying the weakest component. Much fewer approaches have considered debugging training data, which is a key bottleneck for improving machine learning systems [6]. Existing work is mainly developed within the topic of interpretability and transparency of machine learning models [34]. These work mainly uses perturbation based approaches to identify data instances that are likely to cause incorrect predictions [7, 19, 24]. Unlike these work that requires incorrect predictions to be observed, our deep probabilistic method is a self-discovering method as the identification of erroneous labels only relies on data distributions; more importantly, our system involves humans for reliable label correction.

Label Noise Reduction. Existing methods have been developed separately for distant supervision and crowdsourcing. Methods in the former context mainly leverage data distributions in the low-level feature space, e.g., the factor graph model by Riedel et al. [35] and the generative model by Takamastu et al. [39]. Alfonseca et al. [1] introduce a hierarchical model to capture the data distribution in the topic space. In comparison, our deep generative model is flexible to capture data distributions with more complex data structures. In crowdsourcing, label noise reduction has been a central problem as worker annotations are often noisy. Typical methods assume a redundancy of worker annotations [22, 38], e.g., majority voting and those based on Expectation Maximization (EM) [8]. EM methods simultaneously estimate the true labels and parameters of the annotation process, e.g., worker reliability by Dawid and Skene [8] and task difficulty by Whitehill et al. [42]. These methods are, however, not applicable to most machine learning scenarios where each data instance is labeled only once. Our work is different in that we consider the broader scope of noisy training data, where labels are not necessarily redundant.

Finally, our work is also related to the broader domain of data cleaning, mainly studied by the database community [21, 43, 44]. Yakout et al. [44] study an approach to automatically generate data cleaning rules for relational databases. Krishnan et al. [21] specifically focus on cleaning training sets for convex statistical models (e.g., SVM, Logistic Regression). They introduce an SGD-like method that leverages expected model improvement to identify the most beneficial data instances to be cleaned. The method, however, does not apply to deep learning models. Most importantly, all existing work does not explicitly consider the human's role in data cleaning and does not address the potential issues of human involvement (e.g., scalability and work quality).

6 CONCLUSION

We presented Scalpel-CD, a new human-in-the-loop system that exploits the complementary strength of humans and machines for debugging noisy training data. Scalpel-CD leverages deep probabilistic modeling to identify data instances with potentially wrong labels and infer their latent classes, and engages humans to inspect specific data instances for which the model inference is of low reliability. Scalpel-CD is a generic system applicable to any type of training data and can be easily optimized for specific datasets. Our extensive evaluation on multiple machine learning datasets with different noise types demonstrates that Scalpel-CD significantly improves label quality and machine learning performance with minimum human involvement. As future work, we plan to study how to best involve humans to increase the explainability of the models and to reduce potential biases in automated decisions.

ACKNOWLEDGEMENT

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement 683253/GraphInt).

REFERENCES

[1] Enrique Alfonseca, Katja Filippova, Jean-Yves Delort, and Guillermo Garrido. 2012. Pattern learning for relation extraction with a hierarchical topic model. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association of Computational Linguistics, 54–59.

[2] Alexander Ratner Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment (PVLDB)* 11, 3 (2017).

[3] Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. 2015. Soylent: a word processor with a crowd inside. *Commun. ACM* 58, 8 (2015), 85–94.

[4] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. 2017. Variational inference: A review for statisticians. *J. Amer. Statist. Assoc.* 112, 518 (2017), 859–877.

[5] Andrew Carlson, Justin Betteridge, Richard C Wang, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Coupled semi-supervised learning for information extraction. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 101–110.

[6] Aleksandar Chakarov, Aditya Nori, Sriram Rajamani, Shayak Sen, and Deepak Vijaykeerthy. 2016. Debugging machine learning tasks. *arXiv preprint arXiv:1603.07292* (2016).

[7] Anupam Datta, Shayak Sen, and Yair Zick. 2016. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 598–617.

[8] Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics* (1979), 20–28.

[9] Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W Bruce Croft. 2017. Neural ranking models with weak supervision. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 65–74.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 248–255.

[11] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).

[12] Meng Fang, Xingquan Zhu, Bin Li, Wei Ding, and Xindong Wu. 2012. Self-taught active learning from crowds. In *Proceedings of the 12th IEEE International Conference on Data Mining (ICDM)*. IEEE, 858–863.

[13] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep bayesian active learning with image data. *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 1183–1192.

[14] Michael Heilman and Noah A Smith. 2010. Rating Computer-generated questions with Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. 35–40.

[15] Armand Joulin, Laurens van der Maaten, Allan Jabri, and Nicolas Vasilache. 2016. Learning visual features from large weakly supervised data. In *European Conference on Computer Vision (ECCV)*. Springer, 67–84.

[16] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[17] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*.

[19] Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*. 1885–1894.

[20] Alexander Konovalov, Benjamin Strauss, Alan Ritter, and Brendan O'Connor. 2017. Learning to extract events from knowledge base revisions. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*. IW3C2, 1007–1014.

[21] Sanjay Krishnan, Jiannan Wang, Eugene Wu, Michael J Franklin, and Ken Goldberg. 2016. ActiveClean: interactive data cleaning for statistical modeling. *Proceedings of the VLDB Endowment (PVLDB)* 9, 12 (2016), 948–959.

[22] Edith Law and Luis von Ahn. 2011. Human computation. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 5, 3 (2011), 1–121.

[23] Hongwei Li, Bin Yu, and Dengyong Zhou. 2013. Error rate analysis of labeling by crowdsourcing. In *ICML Workshop: Machine Learning Meets Crowdsourcing. Atalanta, Georgia, USA*.

[24] Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220* (2016).

[25] Zachary C Lipton. 2016. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490* (2016).

[26] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.

[27] Tyler McDonnell, Matthew Lease, Mucahid Kutlu, and Tamer Elsayed. 2016. Why is that relevant? Collecting annotator rationales for relevance judgments. In *Fourth AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*. 139–148.

[28] Bart Mellebeek, Francesc Benavent, Jens Grivolla, Joan Codina, Marta R Costa-Jussa, and Rafael Banchs. 2010. Opinion mining of Spanish customer comments with non-expert annotations on Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. 114–121.

[29] Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *International Conference on Machine Learning (ICML)*. 1727–1736.

[30] Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics (ACL*. Association for Computational Linguistics, 271–278.

[31] Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. In *Advances in Neural Information Processing Systems (NIPS)*. 3567–3575.

[32] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *Journal of Machine Learning Research* 11, Apr (2010), 1297–1322.

[33] Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. CoType: Joint extraction of typed entities and relations with knowledge bases. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*. IW2C2, 1015–1024.

[34] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 1135–1144.

[35] Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*. Springer, 148–163.

[36] Alan Ritter, Evan Wright, William Casey, and Tom Mitchell. 2015. Weakly supervised extraction of computer security events from twitter. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*. IW3C2, 896–905.

[37] Burr Settles. 2010. Active Learning Literature Survey. *University of Wisconsin, Madison* 52, 55-66 (2010), 11.

[38] Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. 2008. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 614–622.

[39] Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association of Computational Linguistics, 721–729.

[40] Yuandong Tian and Jun Zhu. 2012. Learning from crowds in the presence of schools of thought. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 226–234.

[41] Jennifer Wortman Vaughan. 2018. Making better use of the crowd: How crowdsourcing can advance machine learning research. *Journal of Machine Learning Research* 18, 193 (2018), 1–46.

[42] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems (NIPS)*. 2035–2043.

[43] Mohamed Yakout, Laure Berti-Équille, and Ahmed K Elmagarmid. 2013. Don't be SCAREd: use SCalable Automatic REpairing with maximal likelihood and bounded changes. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (SIGMOD)*. ACM, 553–564.

[44] Mohamed Yakout, Ahmed K Elmagarmid, Jennifer Neville, Mourad Ouzzani, and Ihab F Ilyas. 2011. Guided data repair. *Proceedings of the VLDB Endowment (PVLDB)* 4, 5 (2011), 279–289.

[45] Yan Yan, Glenn M Fung, Rómer Rosales, and Jennifer G Dy. 2011. Active learning from crowds. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*. 1161–1168.

[46] Yan Yan, Rómer Rosales, Glenn Fung, Mark W Schmidt, Gerardo H Valadez, Luca Bogoni, Linda Moy, and Jennifer G Dy. 2010. Modeling annotator expertise: Learning when everybody knows a bit of something. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 932–939.

[47] Jie Yang, Thomas Drake, Andreas Damianou, and Yoelle Maarek. 2018. Leveraging crowdsourcing data for deep active learning - An application: learning intents in Alexa. In *Proceedings of the 2018 Web Conference (WWW)*. IW3C2, 23–32.

[48] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association of Computational Linguistics, 1753–1762.

[49] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2016. Understanding deep learning requires rethinking generalization. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

How Intention Informed Recommendations Modulate Choices: A Field Study of Spoken Word Content

Longqi Yang
Cornell Tech, Cornell University
ylongqi@cs.cornell.edu

Michael Sobolev
Cornell Tech, Cornell University
michael.sobolev@cornell.edu

Yu Wang
Himalaya Media
yu.wang@himalaya.com

Jenny Chen
The City University of New York
jennychen0905@gmail.com

Drew Dunne
Cornell University
asd222@cornell.edu

Christina Tsangouri
The City University of New York
christinatsangouri@gmail.com

Nicola Dell
Cornell Tech, Cornell University
nixdell@cornell.edu

Mor Naaman
Cornell Tech, Cornell University
mor.naaman@cornell.edu

Deborah Estrin
Cornell Tech, Cornell University
destrin@cornell.edu

ABSTRACT

People's content choices are ideally driven by their intentions, aspirations, and plans. However, in reality, choices may be modulated by recommendation systems which are typically trained to promote popular items and to reinforce users' historical behavior. As a result, the utility and user experience of content consumption can be affected implicitly and undesirably. To study this problem, we conducted a 2×2 randomized controlled field experiment (105 urban college students) to compare the effects of intention informed recommendations with classical intention agnostic systems. The study was conducted in the context of spoken word web content (podcasts) which is often consumed through subscription sites or apps. We modified a commercial podcast app to include (1) a recommender that takes into account users' stated intentions at onboarding, and (2) a Collaborative Filtering (CF) recommender during daily use. Our study suggests that: (1) intention-aware recommendations can significantly raise users' interactions (subscriptions and listening) with channels and episodes related to intended topics by over 24%, even if such a recommender is only used during onboarding, and (2) the CF-based recommender doubles users' explorations on episodes from not-subscribed channels and improves satisfaction for users onboarded with the intention-aware recommender.

KEYWORDS

User intention; Recommendation; Field study; Podcast

ACM Reference Format:
Longqi Yang, Michael Sobolev, Yu Wang, Jenny Chen, Drew Dunne, Christina Tsangouri, Nicola Dell, Mor Naaman, and Deborah Estrin. 2019. How Intention Informed Recommendations Modulate Choices: A Field Study of Spoken Word Content. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313540

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313540

1 INTRODUCTION

People make choices about what content to consume on a day-to-day basis, such as selecting music or podcasts to listen to and identifying articles to read. Ideally, users make choices according to their intentions, plans, and aspirational preferences [31]. For example, a person may use a search engine to find an article or a piece of music with a specific title. However, choosing content on real world platforms is more complex, in part because users' choices are often sub-optimal and focus on the short-term [32], and these immediate choices then get reinforced by recommendation systems that expose users to biased sets of items. The bias of item presentations mainly comes from two sources: (1) recommenders often hold a partial and skewed view of users' preferences that are learned from observational interaction records [40, 51], and (2) recommenders are typically subject to popularity bias [51], which hinders the system from presenting relevant items. When subject to regular exposure to these biased item sets, users' original intention-related choices may be altered — on the one hand, users may explore more content, on the other hand, they may end up consuming trendy but mediocre or irrelevant content with low utility to them.

Prior recommendation systems literature was focused on *how many* [21, 41] and *what* [16, 46] items people choose but rarely addressed *why* people choose them. For example, are the choices a result of people's original intentions or their interactions with recommendation systems? In other words, how recommendations may change users' consumption from what they might have chosen, or aspire to choose? These under-explored questions are critical for recommender systems to listen to users and support users' needs, intentions, and desires [14, 26].

In this paper, we investigate the above mentioned questions, specifically, *how intention informed recommendations modulate users' choices, as compared to intention agnostic systems?* To answer this question, we designed a randomized controlled field study [27] in the domain of podcasts, where we leveraged the **topics of interest** as an indicator of user intentions. The field study is a 2×2 experiment where two factors are two stages of app usage, and two interventions within each factor are different recommendation algorithms. First, during onboarding, users expressed their topics of interest and subscribed to a set of podcast channels through a

website, where we compared a popularity-based recommender to a recommender that takes into account users' intentions (**intention-aware recommender**) in presenting channel candidates. Then, during the remainder of their participation (app usage in the field), users used a customized commercial mobile app without constraint. During this stage of the study, we compared a subscription-based recommender to a Collaborative Filtering (CF)-based recommender in populating the home feed that users interacted with everyday. Finally, participants were invited to complete a post-study survey where they gave ratings in terms of four aspects of satisfaction.

We choose podcasts as the study domain for two main reasons. First, traditional podcast content consumption is typically based on subscriptions and therefore clearly relates to user intentions — users subscribe to RSS feeds of the channels they plan to listen to and then regularly consume released episodes from those channels. Second, recommendation systems for podcasts is of growing importance but currently under-explored (Section 2.4).

We conducted the study with 105 urban college students, which consists of 52.5 hours of one-by-one onboarding, four weeks of field experiments with daily communications and weekly reminders, and a follow-up survey with each participant. Our key findings include:

- **Effects of onboarding recommendations:** Compared to commonly used popularity-based ranking of channels, intention-aware recommendations for user onboarding significantly raised the ratio of channel subscription and episode listening that were aligned with users' topic-wise aspirations (improvements: 72.1% and 36.5% in terms of subscriptions at onboarding and in the field, and 24.9% in terms of listening time).
- **Effects of field recommendations:** Home feeds that were populated by the CF-based recommendations significantly increased the ratio of episode listening to not-subscribed channels by 127.5%, as compared to the traditional home feeds that were filled purely with episodes from subscribed channels.
- **Interaction effects:** User satisfaction was jointly affected by the recommendation algorithms used in the two stages — the CF-based recommender improved satisfaction for users onboarded with the intention-aware recommender, whereas for others, the CF-based recommender was shown to have negative effects.

These findings suggest that recommendations can implicitly but significantly modulate users' intention-related choices — they can encourage or discourage users to pursue their aspirations and intentions. The positive modulation effects can be leveraged to support healthy behavior and benefit an individual's aspired long-term growth, as discussed in Section 5. Also, our study suggests a hybrid form of recommender for podcasts and subscription-based media, consisting of an intention-aware recommender for onboarding and a CF-based recommender for home feed generation. Together, these recommenders support user aspirations, encourage content exploration, and provide satisfying user experiences.

Through our study, we also find that signals regarding the utility of user engagement is not reflected in intention-agnostic statistics (e.g., total listening time and total number of subscriptions) that are commonly employed to understand user experiences (Section 4.1). This highlights the importance of using metrics conditioned on individual intentions to complement the understanding of recommendation effects (Section 5.5).

2 RELATED WORK

Our work builds on and contributes to four lines of research: (1) studying the effects of recommendations, (2) investigating user intentions in using intelligent systems, (3) building recommendation systems beyond optimizing for accuracy, and (4) analyzing and leveraging spoken word content on the web.

2.1 Effects of recommendations

Recommendation systems were shown to increase traffic and user engagement [41], but it was recently recognized in the research community that they can also significantly affect end users' behavior and the structure of a society. Prior work in studying the effects of recommendations mainly focused on the social network structures [11, 42, 43] and the filtering bubble problem [3, 7, 16, 20, 34, 36]. For example, the former line of research demonstrated that introducing friend-based recommendations into social network platforms exacerbates popularity bias (i.e., rich gets richer) [43] and establishes an algorithmic ceiling for minority groups of users [42]. The latter line of research illustrated how recommendations affect users' information exposure by either limiting users' information exposure to a biased scope [16, 36] or enabling users to explore ideologically diverse opinions [3, 16]. As a result, consumers and users may be fragmented [20]. Most recently, Chaney et al. [8] used a simulation to show that recommendations may lead to a homogenization of users' choices. For contextual-aware recommendations, prior work has raised the concern about their potential alternation of users' content consumption context [1].

Although prior research has revealed significant effects of recommendations in the global and individual levels, these effects are user intention-agnostic and are measured and interpreted from system designers' and experts' perspectives. It is unknown whether recommendations' effects are aligned with or deviated from users' own intentions. Our study measures effects from users' angle and contributes findings that are critical to the future user-centric recommendation systems [14, 21, 26].

2.2 User intentions

Understanding and leveraging user intentions is an important theme in designing intelligent systems. For example, in the context of web search, previous research [12, 29, 39, 44, 48] discovered diverse user intents in using search engines [44], e.g., for the same query, users may look for different information. The understanding and prediction of users' intents is an essential component for personalized search experience [12, 44, 48]. In other domains, such as arts and fashion [10], and psychology [15, 38], user intentions were also investigated and were shown to be predictable from behavior logs [10]. In the context of recommendation systems, prior work leveraged interactive systems to elicit signals about user intentions, such as conversation-based [25], survey-based [52], and critique-based [9] systems. Recently, Tomkins et al. [46] presented a system that recommends appropriate products for users who intend to maintain a sustainable behavior.

However, when incorporating users' intentions, the intelligent systems were often evaluated against intention-agnostic metrics, such as click-through rate, dwelling time, and etc., which do not answer the questions of how these systems alter users' choices from

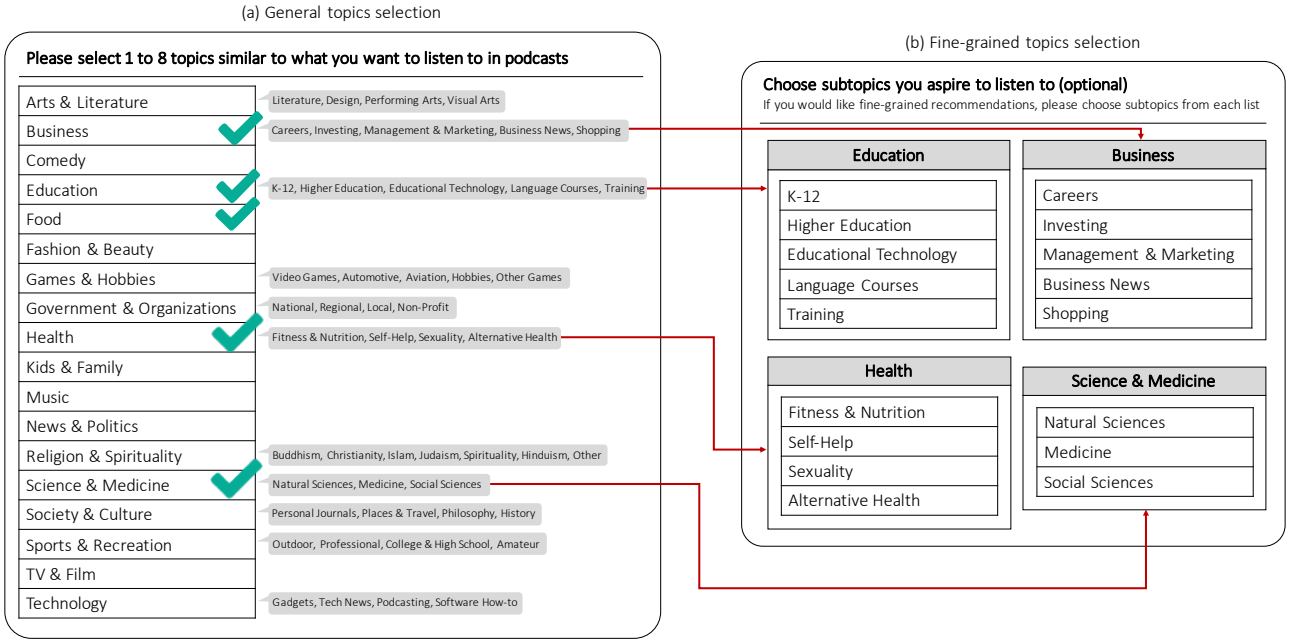


Figure 1: The web user interface designed for participants to indicate their topic-wise intentions. Participants first (a) select up to eight general topics they want to listen to and then (b) optionally select fine-grained topics. The topics are defined using podcast categories in iTunes.

what they might have chosen, and how much of users' intentions were satisfied. As argued by Knijnenburg et al. [26] and Ekstrand et al. [14], future recommenders should be able to satisfy *what users want* and *what they want to achieve*. Our research takes a step further and investigates how intention informed recommenders would in turn affect users' intention-related choices, which closes the feedback loop between choices and recommendation systems.

2.3 Recommendations beyond accuracy

Our work contributes to the increasing recognition and interests in building recommender systems for objectives beyond accuracy [14, 26, 50], such as diversity [21, 53], fairness [13], novelty [45], sustainability [46], and unbiasedness [40, 51]. These objectives were motivated by the observation that recommender systems purely optimized for accuracy may have various negative effects on end users, as reviewed in Section 2.1. These enable recommendations to serve users with different needs and intents. Nevertheless, similar to the limitations discussed in Section 2.2, prior work optimized these systems using hand-crafted or expert-designed metrics (such as categorical accuracy [13]), which may or may not be aligned with users' intentions and goals. Our study reveals the extent to which users' choices are related to their intentions, which can be used to inform future design of recommenders beyond accuracy.

2.4 Web spoken word content

We conducted the field study in the domain of spoken word content (podcasts) — an emerged channel for information and entertainment [37]. In the web community, prior research was mainly focused on building web search engines [5, 17, 19, 33, 35], which index podcast metadata and audio files so as to match given text queries to audio. However, there has been very little work addressing the podcast recommendation problem. The only work we recognized was from Tsagkias et al. [47] that predicted users' podcast preference using hand-crafted preference indicators, which can hardly be applied in the wild because of the heterogeneity of users and content. With the interests from major media companies to serve podcasts, research is needed to build recommenders that better expose users to content beyond passive receiving. Our study contributes a hybrid form of podcast recommender that serves users' intentions, encourages exploration and results in higher user satisfaction. Our paper also presents key guidelines for the design of podcast recommenders, which can be applied to other subscription-based media platforms as well.

3 STUDY DESIGN

Our study design included collecting consumption intentions from all participants and randomly assigning participants to four independent experimental conditions. This design allowed us to conduct within-subject comparisons to understand the discrepancy between users' content consumption and intentions, and between-subject comparisons to measure the effects of different recommendations. Specifically, our study consisted of two phases: an **one-by-one video onboarding** (30 minutes) and a **field study** (four weeks), corresponding to the prominent settings under which podcast listeners are exposed to recommendations in the wild (i.e., when they first begin to use an application, and during the daily usage). Our design for both phases of the study allowed participants to

interact with recommendations naturally. During onboarding, participants were instructed to subscribe to a set of podcast channels they wanted to listen to from a ranked list of candidates; and in the field, participants were provided with a customized commercial podcast mobile app (available on both Android and IOS) to listen to podcasts naturally and without study constraints. The experiment used a full 2×2 factorial design where the two factors were recommendations made in the two study stages, i.e., onboarding (**ONB**) and field (**FIE**) recommendations, and the two interventions within each factor were specific algorithms that presented channels or episodes in different orders. Below, we describe detailed design of each phase.

3.1 Onboarding (ONB)

We onboarded participants one-by-one using remote video conferencing software. Participants were instructed to complete two tasks during onboarding: (1) indicate their topic-wise intentions and interests, and (2) subscribe to channels that they want to listen to in the field. Participants were directed to use a website we developed to complete both tasks.

Indicating topic-wise intentions. We collected participants' listening aspirations in the form of podcast topics (Fig. 1). This topic selection approach is a common practice adopted by major content platforms (e.g., Pinterest and Medium) to elicit user preferences during onboarding. We used podcast categories defined by iTunes[1] as topics, which consists of two levels: general and fine-grained. Through the website, participants first picked 1-8 general topics (Fig. 1-a), and then optionally chose fine-grained topics within the selected general ones (Fig. 1-b). To help participants make sense of general topics, fine-grained topics were shown side-by-side.

Subscribing to channels. Each participant was then asked to subscribe to up to ten podcast channels from a list of recommendations (Fig. 2). The recommendation list was subject to the control or experimental setting, according to the participants' assignments in the study. The control intervention implemented a standard user onboarding strategy that ordered channels based on their popularity on iTunes (**POP**) (Fig. 2-a), whereas the experimental intervention ranked channels by the degree to which they related to participants' aspirations[2] (**ASP**) (Fig. 2-b). The relevance of a channel c for a user u is characterized by a score $s(c|u)$ calculated as follows.

$$s(c|u) = |\mathcal{S}_c \cap \mathcal{A}_u| + \mathbb{1}\left[m_c \in \mathcal{A}_u\right] \tag{1}$$

where $\mathcal{S}_c$ is the set of topics (general and fine-grained) that the channel c belongs to, m_c is the channel's primary topic ($m_c \in \mathcal{S}_c$), and $\mathcal{A}_u$ is the set of topics that users aspired to listen to. Both $\mathcal{S}_c$ and m_c were scraped via iTunes RSS API. As shown in the above equation, when calculating $s(c|u)$, we placed an additional weight on the primary topic.

For both groups, participants were instructed to browse the website freely and make decisions at any point of time. To prepare channels for recommendations, we scraped all top channels returned by the iTunes RSS feed, and made a join with our podcast database. Eventually, 2231 channels were used.

[1]Podcast directory: https://itunes.apple.com/us/genre/podcasts/id26?mt=2
[2]To break ties, channel popularity on iTunes was used.

3.2 Field Study (FIE)

After onboarding, each participant was provided with a podcast mobile app and a pre-registered account to use for four weeks in the field. The app was pre-loaded with the channels for which the participant subscribed during onboarding. We customized a popular commercial app for our study. The app (shown in Fig. 3 and Fig. 4) has three main pages: (1) a **home** page (Fig. 3-b,c) that presented a personalized list of new podcast episodes, and is the default page when opening the app, (2) a **library** page (Fig. 3-a) that showed the channels to which a user has subscribed, and (3) a **discover** page (Fig. 4) that listed channels based on categories and popularity, which were not personalized. In addition, the app allowed users to directly search for content (through the icon at the top-right corner), and users can also consume episodes from a channel's page (by clicking on the channel's thumbnail).

Similar to onboarding, the field intervention was applied to recommendations on the mobile home page, which chronologically listed episodes from a personalized set of channels and was refreshed daily for newly-released episodes. For the control group, the personalized set contained channels to which a user has subscribed (**SUB**); whereas for the experimental intervention, the set additionally mixed five not-subscribed channels (**MIX**). These channels were retrieved by a matrix factorization based recommendation model, which we built as follows:

- **Dataset collection**. We scraped the most recent 500 reviews of 29K popular podcast channels on iTunes to train a recommendation model. In order to conduct recommendations based on users' channel subscriptions, which are binary signals, we disregarded rating scores and treated iTunes reviews as positive-only feedback. The final training dataset contained 702K user-channel interactions from 137K iTunes users.
- **Recommendation model**. We used OpenRec [50] to build a Weighted Regularized Matrix Factorization (WRMF) [22] based recommender, which is a representative implicit-feedback-based recommendation model and is optimized to minimize the following objective function:

$$\min_{x_*, y_*} \sum_{u \in \mathcal{U}, i \in \mathcal{I}} w_{ui}(p_{ui} - x_u^T y_i)^2 + \lambda \|\Theta\|^2 \tag{2}$$

  where Θ is a set of model parameters, x_u and y_i are latent factor representations for user u (among all iTunes users $\mathcal{U}$) and channel i (among all iTunes channels $\mathcal{I}$) respectively, and p_{ui} is a binary indicator for user preferences ($p_{ui} = 1$ if user u subscribed to channel i, and $p_{ui} = 0$ otherwise). In addition, WRMF uses w_{ui} to control models' confidence levels on p_{ui}. We set w_{ui} such that $w_{ui} = 1$ if $p_{ui} = 1$, and $w_{ui} = 0.01$ otherwise. These parameter settings achieved the best validation results in our dataset. When applying the WRMF model, we discarded x_u since it corresponds to users from iTunes, and fixed trained channel representations y_i. For a participant u', we derived an analytic expression of the optimal user representation $x_{u'}$ by differentiating the objective function (eqn. 2):

$$x_{u'} = \frac{1}{\lambda + \sum_i w_{u'i} y_i^T y_i} \sum_i w_{u'i} p_{u'i} y_i \tag{3}$$

- **Not-subscribed channel retrieval**. For any participant u' in the experimental group, we retrieved the top 5 not-subscribed

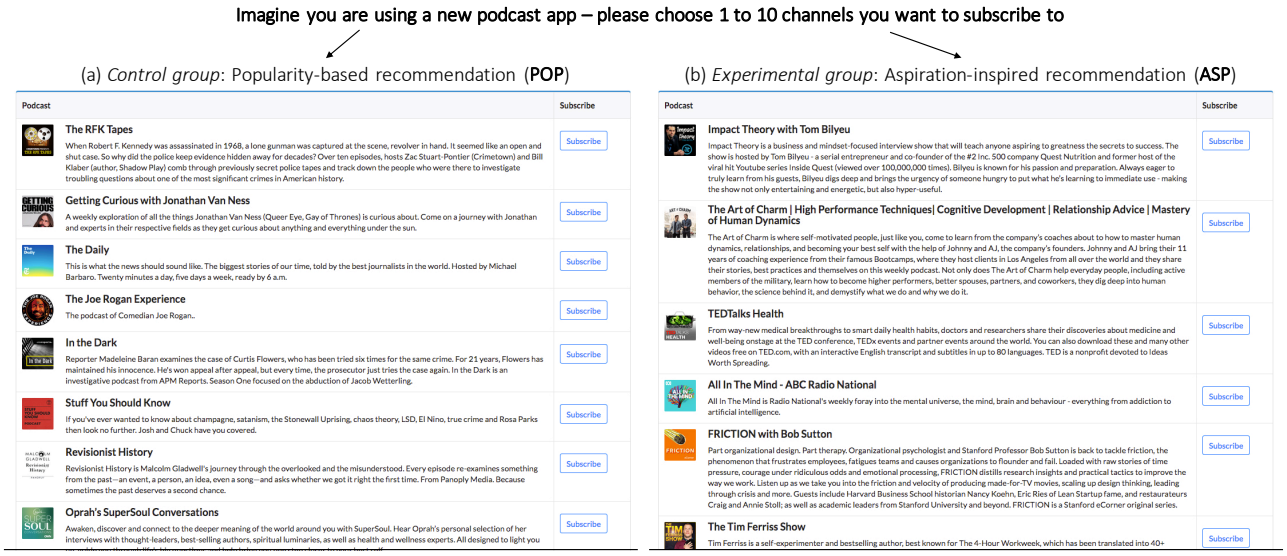
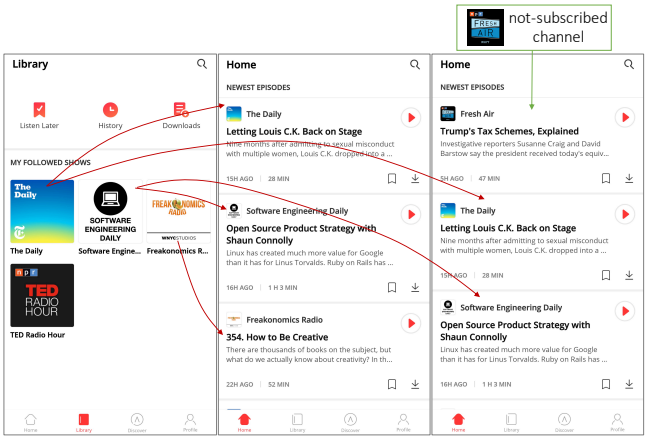


Figure 2: The web user interface designed for participants to subscribe to channels during onboarding. The interface presented a list of podcast shows, and participants were instructed to subscribe to up to ten of them. For the control group (POP), channels were ordered by their popularity on iTunes, whereas for the experimental group (ASP), the ordering was determined by channels' alignment to participants' topic-wise intentions. Both groups shared the same set of candidate content.

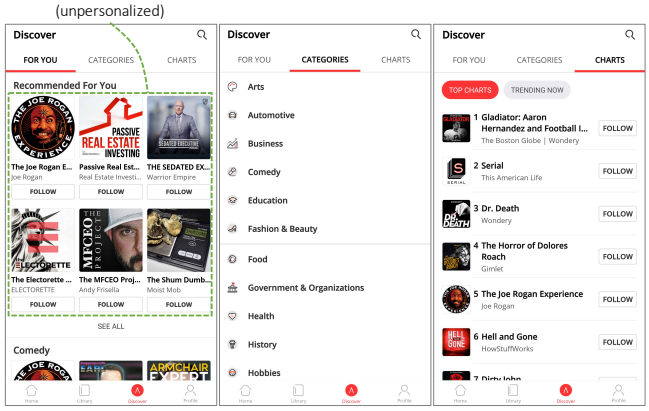


(a) Library showing subscribed channels

(b) *Control group*: episodes from subscribed channels (**SUB**)

(c) *Experimental group*: episodes from mixed channels (**MIX**)

Figure 3: The library and home pages of the customized podcast mobile app. The library page showed the channels to which a user has subscribed, and the home page chronologically presented a list of episodes. For the control group (SUB), the episodes were retrieved from subscribed channels, whereas for the experimental group (MIX), those episodes were mixed with the ones from selected not-subscribed channels based on a CF recommendation model.



(a) Randomly sampled popular channels

(b) Categorical listing of channels

(c) Popular/Trending channels

Figure 4: The discover page of the customized podcast mobile app. The page grouped channels into topic-wise categories and presented a trending chart that ordered channels according to their popularity on iTunes. This page allowed users to readily explore and subscribe to new channels.

channels that had the highest dot product scores (i.e., $x_{u'}y_i, i \in \{i|p_{u'i} = 0 \wedge i \in \mathcal{I}\}$). Although the recommendation model was fixed throughout the study, retrieved channels were adaptively updated whenever participants subscribed to new shows.

3.3 Post-study Survey

After participants finished the 4-weeks field study, we conducted a post-study survey through email to elicit user satisfaction. The survey questions follow a template: "How satisfied were you with

Total number of participants: 105, **unreported:** 26	
Gender:	Female: 50 Male: 29
Age (years):	Max: 43 Min: 17 Mean: 21
Device:	IOS: 49 Android: 30
Major:	Computing and Information Science: 20 Arts & Sciences: 22 Life Sciences: 10 Medicine: 2 Business: 16 Engineering: 9

Table 1: Participants' demographic information including gender, age, primary mobile device, and college major.

___?", and the aspects we surveyed include *the app*, *the experiment*, *your current podcast channel subscriptions*, and *the home feed in the app*. For each question, participants were instructed to give a likert-scale rating (i.e., *not at all satisfied*, *slightly satisfied*, *neutral*, *very satisfied*, and *extremely satisfied*).

3.4 Participant Recruitment

We recruited 105 full-time undergraduate students who were studying in New York City and were from diverse background. The demographic information of the participants is summarized in Table. 1. Participants were compensated with $30 after completing both phases of the study. To encourage app usage in the field, we provided an additional $20 bonus for those who used the mobile app for at least five days a week, and reminded all participants to listen to new episodes weekly. Finally, participants were randomly assigned to one of the 2×2 conditions (POP-SUB: 25, POP-MIX: 26, ASP-SUB:29, ASP-MIX:25), and two research personnel who were blind to condition assignments managed and executed participants onboarding and the field study. The study was approved by the Institutional Review Board (IRB) under the protocol #1507005739.

4 STUDY RESULTS

Our study recorded the choices that participants made at onboarding and in the field including both channel subscriptions and episode listening. In addition, we recorded satisfaction ratings that participants gave to questions in the final survey. Eventually, 99 out of 105 participants completed the study (POP-SUB: 24, POP-MIX: 23, ASP-SUB:28, ASP-MIX:24). We summarize and present our study results in four dimensions: general usage patterns (Section 4.1), choices related to topic-wise intentions (Section 4.2), exploratory choices (Section 4.3), and user satisfaction (Section 4.4).

4.1 General usage patterns

To understand the usability and user experience with our podcast content platform, we investigate a type of commonly used metrics, user activity level [28]. We count the number of subscriptions that each user made in the field, and the amount of time that each user spent listening to episodes. The distributions of these measures over users are illustrated in Fig. 5. Overall, participants were fairly active in using the mobile app in the field with 8.8 average number of subscriptions and 4.58-hour average listening time. Participants' activity level is also distributed within a range and has rare outliers (Fig. 5-b,c). In Fig. 5-a, we also plot the distribution of the number of onboarding subscriptions, which is shown to spread from one to ten (maximum allowance) with an average of 7.4. To test whether two experimental factors affect the three measures in Fig. 5, we conduct a general nonparametric factorial analysis using the Aligned Rank Transform (**ART**) [49] (by treating the three measures as responses). We use ART because our study contains more than one factor, and all the measures are not normally distributed over users[3]. In the rest of this paper, if not specified, the ART is used to conduct statistical significance tests (notations: $***$: $p < 0.001$, $**$: $p < 0.01$, $*$: $p < 0.05$). The ART reports no significant effect from ONB, FIE, or ONB×FIE for all the three measures. However, as shown in Section 4.2 and 4.3, ONB and FIE have significant effects on users' podcast consumption patterns, although they are not captured in the general user activity measures. We discuss the limitations of these traditional measures in Section 5.5.

In addition to aggregate users' activities (subscriptions and listening time) on a per-user basis, we also cluster the activities into hour of day (Fig. 6-a), day of week (Fig. 6-b), and distinct channels (Fig. 7). Temporal distributions of listening instances (Fig. 6) reveal several diurnal and weekly listening patterns, such as decreased listening during night and over weekends. However, no statistical evidence shows significant effects of experimental factors on these temporal patterns. Regarding the channel-wise user activity distributions (Fig. 7), they demonstrate that (1) during onboarding (Fig. 7-a), participants' channel subscriptions manifested significant popularity bias under the POP treatment, i.e., the majority of user subscriptions were concentrated on a small number of channels, whereas under the ASP treatment, subscriptions were spread out to more channels and tended to be uniformly distributed, and (2) in the field, users interacted with a broader set of podcast channels than during onboarding, but both experimental factors have no significant effect on the number of interactions that each channel received. In the Appendix, we additionally visualize top channels subscribed by participants.

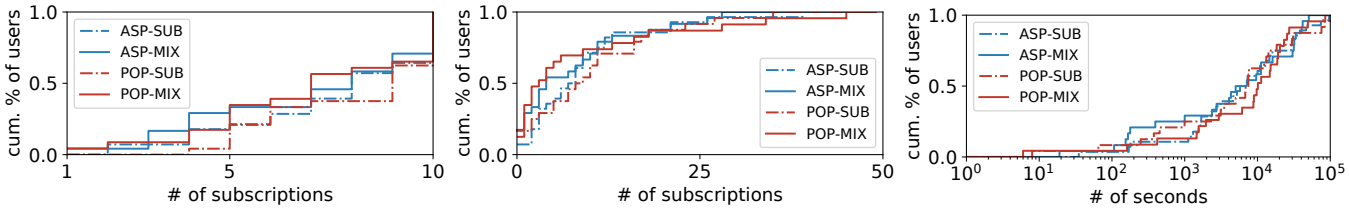
4.2 Choices related to topic-wise intentions

The distribution of the topics that participants intended to consume is shown in Fig. 8, which shows the diversity of the topics of interest chosen by participants — the intended topics in the population were spread across 53 distinct general and fine-grained categories, and most of the topics were selected by less than half of the population. Such a wide range of selected topics is partially attributable to the diverse background of our recruited participants (Table. 1). To show how users' choices related to topic-wise intentions may be modulated by two stages of recommendations, we define a topic-wise intention ratio $r_{\text{topic}}(c|u)$ of a channel c for the user u as follows:

$$r_{\text{topic}}(c|u) = \frac{|\mathcal{S}_c \cap \mathcal{A}_u|}{|\mathcal{S}_c|} \tag{4}$$

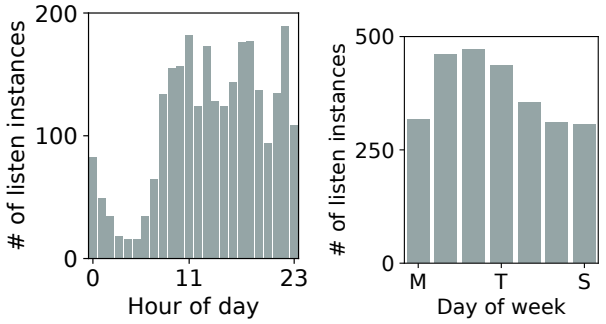
where we use the notations from eqn. 1. The value of $r_{\text{topic}}(c|u)$ corresponds to the proportion of a channel's content that aligns with a user's intended topics. Then using $r_{\text{topic}}(c|u)$, we calculate the average alignment of a user u's subscriptions, $P^{\text{sub}}_{\text{topic}}(u)$, as the

[3]The normality test is conducted via the Shapiro-Wilk normality test



(a) Onboarding subscriptions. (b) Field subscriptions. (c) Listening time.

Figure 5: The cumulative distributions of users over number of subscriptions and listening time. These figures show the extent to which participants were actively subscribing and listening to podcasts throughout the study. A vertical line in these figures represents a group of users with a similar activity level. We note that these commonly-used aggregated measures are not statistically different across the four groups. In other words, they do not reflect the different composition of content consumption across these groups (Section 4.2 and 4.3). These differences are critical to understand the effects of recommendations on individual growth and experience.



(a) Hour of day distribution. (b) Day of week distribution.

Figure 6: The distribution of podcast listening instances over hour of day and day of week. The aggregation is across all participants. Again we note that no statistical difference is observed across the four groups.

average $r_{\text{topic}}(c|u)$ over all followed channels $\mathcal{F}_u$, i.e.,

$$P^{\text{sub}}_{\text{topic}}(u) = \frac{\sum_{c\in\mathcal{F}_u} r_{\text{topic}}(c|u)}{|\mathcal{F}_u|} \tag{5}$$

and calculate the average alignment of a user u's listening, $P^{\text{listen}}_{\text{topic}}(u)$, as the weighted average of $r_{\text{topic}}(c|u)$ over listened channels $\mathcal{L}_u$ with the weight proportional to the listening duration d_c, i.e.,

$$P^{\text{listen}}_{\text{topic}}(u) = \frac{\sum_{c\in\mathcal{L}_u} r_{\text{topic}}(c|u)d_c}{\sum_{c\in\mathcal{L}_u} d_c} \tag{6}$$

We show the cumulative distributions of users over $P^{\text{sub}}_{\text{topic}}(u)$ and $P^{\text{listen}}_{\text{topic}}(u)$ in Fig. 9, and the 2×2 groupwise averages in Fig. 10. These graphs and corresponding ART tests demonstrate that under all scenarios, the ASP intervention significantly improves the ratio of content consumption that matches users' topic-wise intentions — during onboarding, ASP increases $\overline{P}^{\text{sub}}_{\text{topic}}$ by 72.1% (ONB:**), and in the field, ASP improves $\overline{P}^{\text{sub}}_{\text{topic}}$ and $\overline{P}^{\text{listen}}_{\text{topic}}$ by 36.5% (ONB:***) and 24.9% (ONB:*) respectively. It is worth noting that although improvements are larger at onboarding when the intervention is directly applied, ASP is shown to have significant indirect effects on users' content consumption in the field as well. The statistical test does not show significant effects from the FIE factor and the interaction (i.e., ONB×FIE).

4.3 Exploratory choices

To investigate how participants' exploratory choices were affected by recommendations, we divided their podcast listening into subscribed listening (exploitation) and not-subscribed listening (exploration). We define the exploratory ratio $r_{\text{explore}}(c|u)$ as a counterpart for $r_{\text{topic}}(c|u)$ (Section 4.2). This exploratory ratio is calculated as follows.

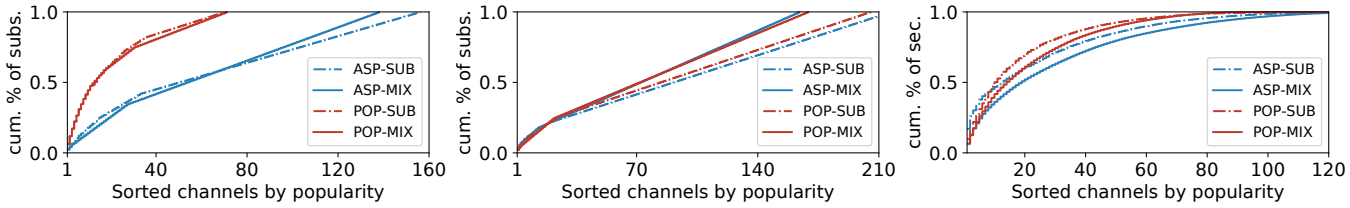
$$r_{\text{explore}}(c|u) = 1 - \mathbb{1}\left[c \in \mathcal{F}^t_u\right] \tag{7}$$

where $\mathbb{1}$ is an indicator function, and $\mathcal{F}^t_u$ is the set of channels that the user u subscribed to at time t when the channel c was consumed. Essentially, $r_{\text{explore}}(c|u) = 1$ if the channel was not subscribed when consumed, otherwise $r_{\text{explore}}(c|u) = 0$. We then substitute $r_{\text{topic}}(c|u)$ in eqn. 6 with $r_{\text{explore}}(c|u)$ and derive an exploratory measure of a user u' s listening, denoted as $P^{\text{listen}}_{\text{explore}}(u)$. From another angle, $P^{\text{listen}}_{\text{explore}}(u)$ can be viewed as the percentage of time that the user u explored new information channels.

We show the distributions of users over $P^{\text{listen}}_{\text{explore}}(u)$ and the groupwise average scores in Fig. 11. Both figures and ART statistical tests demonstrate that the MIX intervention significantly increases $\overline{P}^{\text{listen}}_{\text{explore}}$ by 127.5% (FIE:*). In other words, the MIX feeds significantly encouraged participants to explore beyond existing and potentially narrow information channels. The onboarding recommendations (ONB) and the interaction between the two stages of recommendations (ONB×FIE) do not have significant effects.

4.4 User satisfaction

Four satisfaction indicators were surveyed and reported by participants after the study was over (Section 3.3). Among 99 valid participants, 89 of them responded to our email survey (Response rate: 89.9%). Both experimental factors and their interaction do not have significant effects on whether or not a participant responded to the survey. To quantitatively analyze survey results, following



(a) Onboarding subscriptions. **(b) Field subscriptions.** **(c) Listening time.**

Figure 7: The cumulative distributions of subscriptions and listening time over channels ordered by popularity. The popularity is defined as the number of subscription (a, b) and the amount of listening (c). These figures show the extent to which participants' content consumption was concentrated on a small set of popular items. A linear line in the figure represents uniformly distributed consumption over all channels. During onboarding, the POP intervention resulted in significant popularity bias in participants' subscriptions, but in the field, no significant effect from experimental factors is observed.

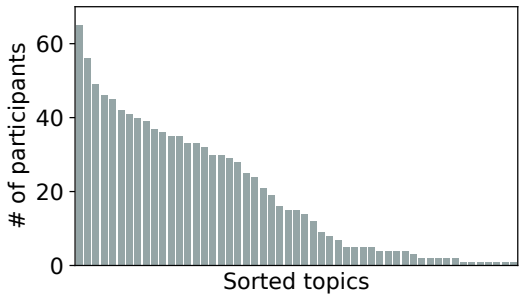


Figure 8: The distribution of user intentions over podcast topics (categories). Topics are sorted by their popularity in a descending order. Participants' intended topics were diversely spread across 53 categories, with most of the topics liked by less than half of the participants.

the common practice [21], we convert the five options in each survey question, i.e., *not at all satisfied*, *slightly satisfied*, *neutral*, *very satisfied*, and *extremely satisfied*, to 1–5 numerical ratings.

We found that satisfactions for all indicators are highly correlated. Therefore, we aggregated them into one factor by taking the average of the ratings. The distributions of the aggregated satisfaction ratings and the groupwise average values are shown in Fig. 12. Participants' satisfaction is significantly affected by the interaction between two factors (ONB×FIE: $*$); and the post-hoc differences of differences test [6, 30] confirms the effects of one factor given the other. In other words, if participants were onboarded with the popularity-based recommender, applying the CF-based recommender to populate users' home feeds significantly degraded users' satisfaction, whereas if participants were initially presented with a channel list ranked by their intentions, the CF-based recommender used in the field showed positive effects. These findings have important implications as to how the reinforcing nature of recommendations may improve or degrade utility and user experience, as discussed in Section 5.

5 IMPLICATIONS AND DISCUSSIONS

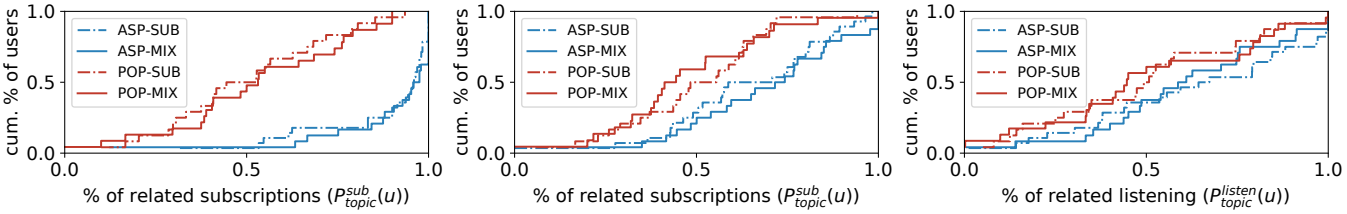
Our study results indicate significant interactions between recommendations and intentions. We discuss our findings in light of theoretical and empirical research on human decision making and suggest directions for designing better recommendation systems that benefit end users.

5.1 Employing planning and intentions

Individuals face self-control challenges when making decisions about content consumption, just as they do when managing diet or finance [32]. People have troubles translating their intentions and goals into actions when facing real-world decision-making problems. For example, prior research showed that people rent documentaries in line with their "aspirational self" but were less likely to actually consume this type of movie compared to more affective movies such as action films [32]. Filter bubbles [36] are another example in which users' long-term interests do not match with short-term consumption of news. To help people choose according to their long-term interests, our study suggests to employ a deliberative thinking via planning in the form of preference elicitation. As shown in Section 3.1, our onboarding system leveraged a preference elicitation-based interaction technique and an intention-aware recommender system that allowed for the explicit inclusion of user intentions. Such a design was shown to have significantly positive effects as users subscribed according to their elicited intentions during onboarding and later followed up on their plans when listening in the wild. Similar strategies were examined in behavioral science literature suggesting that people planning ahead are more likely to act on their intentions and to exhibit aspirational behavior in line with their long term interests [18].

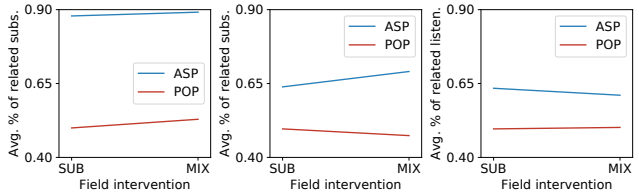
5.2 Encouraging exploration

Classical recommendation systems based on collaborative filtering and click-based metrics are often criticized since they are likely to be overly optimized to reinforce past behavior and preferences [24]. As a result, measures such as novelty and diversity are increasingly explored both in research papers and industry practice in recent years [21, 45, 53]. Finding the right mix of novel and familiar items can be challenging as it is not clear to what extent a certain quality characteristic like novelty is truly desired in a given application for a specific user and at a certain time. In the social and behavioral science literature this is often formulated as the exploration-exploitation trade-off [2, 4, 23]. Our results demonstrate



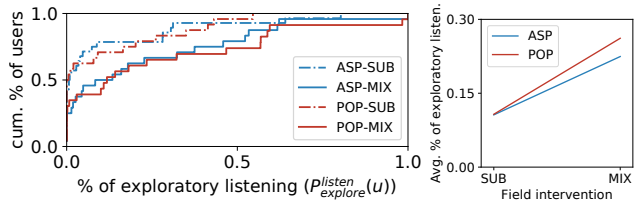
(a) Onboarding subscriptions (ONB: $**$). **(b) Field subscriptions (ONB: $***$).** **(c) Listening time (ONB: $*$).**

Figure 9: Cumulative distributions of users over the percentage of the topicwise intention-related subscriptions and listening. In the above figures, an $x = 1.0$ curve denotes that all users' consumption is related to their topicwise intentions, while an $x = 0.0$ curve denotes that none are related. The ASP intervention during onboarding is shown to significantly increase the topic-related onboarding subscriptions, topic-related field subscriptions, and topic-related field listening. The FIE factor and the interaction ONB×FIE have no significant effect.



(a) Onboard. subs. (ONB:$**$) **(b) Field subs. (ONB:$***$)** **(c) Listening time (ONB:$*$)**

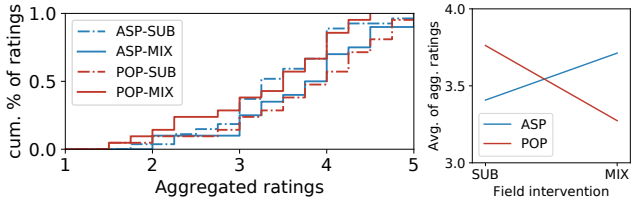
Figure 10: The groupwise average percentage of the topicwise intention-related subscriptions and listening. The ASP intervention significantly improves the topic-relatedness of onboarding subscriptions, field subscriptions, and field episode listening by 72.1%, 36.5%, and 24.9% respectively. The FIE and the interaction (ONB×FIE) have no significant effect.



(a) The cumulative distribution over users (FIE: $*$). **(b) The groupwise average (FIE: $*$).**

Figure 11: The percentage of subscriptions and listening from not-subscribed channels — (a) cumulative distributions over users, and (b) groupwise average. In (a), a $x = 1.0$ curve denotes that users do not listen to episodes from subscribed channels, while a $x = 0.0$ curve denotes that all listening comes from subscribed channels. The MIX intervention is shown to significantly increase the exploration rate by 127.5%. The ONB factor and the interaction (ONB×FIE) have no significant effect.

that introducing recommendation systems in content platforms that were mainly driven by user intentions provided benefits in the form of user exploration, because recommendations helped people find



(a) The cumulative distributions of aggregated ratings (ONB×FIE:$*$). **(b) The average ratings (ONB×FIE:$*$).**

Figure 12: Participants' satisfaction (the averaged ratings of all indicators) — (a) cumulative distributions of aggregated ratings, and (b) groupwise average ratings. The interaction between two factors (ONB×FIE) significantly affects satisfaction — MIX improves satisfaction if participants were onboarded with the ASP, otherwise MIX shows negative effects. No single factor alone has a significant effect.

choice alternatives that they were not aware of. However, how recommendation systems may influence the explore-exploit dilemma in the long term is an open question for future research.

5.3 Understanding user satisfaction

Our results show that users were satisfied when CF-based recommendations (MIX) were delivered based on intention-driven subscriptions (ASP) at onboarding. This can be explained by the benefits of reinforcing users' long-term interests. Whereas when users' initial subscriptions were only driven by channels' popularity and not aspirational, CF-based recommendations ignored their intentions and left them dissatisfied. Another possible explanation is the explainability and trust of recommendations. People are more likely to follow recommendations they trust, and explaining recommendations is shown to increase the trust [54]. Since the ASP-MIX hybrid recommender systems were informed by stated users' preferences, recommendations were implicitly explained and were easier to be perceived and understood by users. Whereas when the POP-MIX systems were used, the explainability and trust of field recommendations was expected to be low.

Additionally, as shown in Section 4.4, we also observe high user satisfaction under the POP-SUB interventions, in which users were

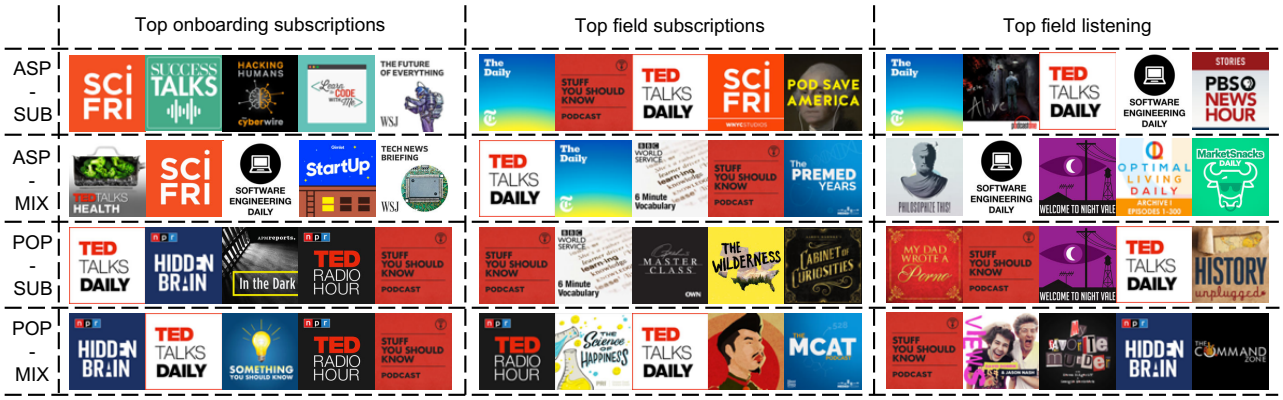


Figure 13: Top five most interacted content source during onboarding and in the field, categorized by 2×2 groups. Each square icon represents a podcast channel. These qualitative results further demonstrate how users' content consumption in the field was jointly affected by users' intentions and recommendation systems.

left in their information bubble populated with self-chosen popular items. This may be explained by people's inherent motivation to chase popular items [51] even if these items were misaligned with users' stated intentions; such content satisfies an important, if implicit, aspect of people's information needs and desires.

5.4 Optimizing for multiple objectives

Our study reveals benefits of jointly optimizing people's information consumption for multiple objectives. For example, for podcasts and other subscription-based media, service providers should consider a hybrid form of recommender that contains an intention-aware recommender for onboarding and a CF-based recommender for field listening. This combination can support users' intentions while encouraging them to explore beyond existing channels. As a result, users are likely to be more satisfied. More generally, with a global view of the recommendations that people are increasingly exposed to, we can jointly optimize recommendation systems to support an individual's aspirations and satisfaction in other domains such as diet and time management.

5.5 Limitations of intention-agnostic metrics

Commonly-used metrics that quantify user experiences are often agnostic to people's intentions. As a result, these metrics mainly reflect the extent to which recommendations engage people but overlook the utility of those engagements. For example, in our study, total listening time and total number of subscriptions show that people were equally active across different groups (Section 4.1), but in reality, people in certain groups were less exposed to new information, guided away from their aspirations, and less satisfied. Therefore, when probing and evaluating the performance of recommendation systems, it is important to condition metrics on individual intentions.

6 CONCLUSIONS AND FUTURE WORK

We presented a randomized controlled field experiment that studied the effects of recommendations on people's content choices related to intentions. Our study revealed how recommendations (1) modulated people's choices of topically relevant content, (2) affect the likelihood that people explore beyond their existing information sources, and (3) jointly affected user satisfaction. We discussed the implications and applications of our study findings on the design, evaluation and understanding of recommendation systems. Our study confirms the suspected importance of recommendations beyond discovering relevant information; in particular, that these systems implicitly alter online behavior in a manner that can have profound implications for individuals and society [1]. Future work is needed to study the generalization of these effects to wider demographic groups, and explore broader and longer term effects of recommendations through offline evaluation, simulations, and larger scale field experiments.

A QUALITATIVE USAGE RESULTS

We show the channels that were most-subscribed and listened during onboarding and in the field (Fig. 13). During onboarding, the subscriptions made in ASP-$*$ groups were much more diverse compared to the POP-$*$ groups. The subscriptions from POP-$*$ groups were mostly concentrated on trendy channels such as TED Talks Daily, TED Radio Hour, and Hidden Brain. However, in the field, all groups manifested diverse content consumption patterns, and the top subscribed and listened channels contained both trendy and long-tail items. These qualitative results further illustrate how users' podcast content consumption was driven by users' intentions and at the same time affected by recommendation systems.

ACKNOWLEDGMENTS

This research was supported by the National Science Foundation under grant IIS-1700832 and by Yahoo Research (via the Connected Experiences Laboratory at Cornell Tech). The work was further supported by the small data lab at Cornell Tech, which received funding from NSF, NIH, RWJF, UnitedHealth Group, Google, and Adobe. The City University of New York provided generous support in recruiting participants. We thank the anonymous reviewers for their insightful comments and suggestions.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2011. Context-aware recommender systems. In *Recommender systems handbook*. Springer, 217–253.

[2] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Raghu Ramakrishnan. 2013. Content recommendation on web portals. *Commun. ACM* 56, 6 (2013), 92–101.

[3] Eytan Bakshy, Solomon Messing, and Lada A Adamic. 2015. Exposure to ideologically diverse news and opinion on Facebook. *Science* 348, 6239 (2015), 1130–1132.

[4] Oded Berger-Tal, Jonathan Nathan, Ehud Meron, and David Saltz. 2014. The exploration-exploitation dilemma: a multidisciplinary framework. *PloS one* 9, 4 (2014), e95693.

[5] Jana Besser, Martha Larson, and Katja Hofmann. 2010. Podcast search: User goals and retrieval technologies. *Online information review* (2010).

[6] Robert J Boik. 1979. Interactions, partial interactions, and interaction contrasts in the analysis of variance. *Psychological Bulletin* 86, 5 (1979), 1084.

[7] Danah Boyd. 2010. Streams of content, limited attention: The flow of information through social media. *Educause Review* 45, 5 (2010), 26.

[8] Allison JB Chaney, Brandon M Stewart, and Barbara E Engelhardt. 2017. How Algorithmic Confounding in Recommendation Systems Increases Homogeneity and Decreases Utility. *arXiv preprint arXiv:1710.11214* (2017).

[9] Li Chen and Pearl Pu. 2012. Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction* 22, 1-2 (2012), 125–150.

[10] Justin Cheng, Caroline Lo, and Jure Leskovec. 2017. Predicting intent using activity logs: How goal specificity and temporal range affect user behavior. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 593–601.

[11] Elizabeth M Daly, Werner Geyer, and David R Millen. 2010. The network effects of recommending social connections. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 301–304.

[12] Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 581–590.

[13] Michael D Ekstrand, Mucun Tian, Ion Madrazo Azpiazu, Jennifer D Ekstrand, Oghenemaro Anuyah, David McNeill, and Maria Soledad Pera. 2018. All The Cool Kids, How Do They Fit In?: Popularity and Demographic Biases in Recommender Evaluation and Effectiveness. In *Conference on Fairness, Accountability and Transparency*. 172–186.

[14] Michael D Ekstrand and Martijn C Willemsen. 2016. Behaviorism is not enough: better recommendations through listening to users. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 221–224.

[15] Andrew J Elliot and Judith M Harackiewicz. 1994. Goal setting, achievement orientation, and intrinsic motivation: A mediational analysis. *Journal of personality and social psychology* 66, 5 (1994), 968.

[16] Seth Flaxman, Sharad Goel, and Justin M Rao. 2016. Filter bubbles, echo chambers, and online news consumption. *Public opinion quarterly* 80, S1 (2016), 298–320.

[17] Marguerite Fuller, E Tsagkias, Eamonn Newman, Jana Besser, Martha Larson, Gareth JF Jones, M Rijke, et al. 2008. Using term clouds to represent segment-level semantic content of podcasts. (2008).

[18] Peter M Gollwitzer. 1999. Implementation intentions: strong effects of simple plans. *American psychologist* 54, 7 (1999), 493.

[19] Masataka Goto and Jun Ogata. 2011. PodCastle: Recent advances of a spoken document retrieval service improved by anonymous user contributions. In *Twelfth Annual Conference of the International Speech Communication Association*.

[20] Kartik Hosanagar, Daniel Fleder, Dokyun Lee, and Andreas Buja. 2013. Will the global village fracture into tribes? Recommender systems and their effects on consumer fragmentation. *Management Science* 60, 4 (2013), 805–823.

[21] Cheng-Kang Hsieh, Longqi Yang, Honghao Wei, Mor Naaman, and Deborah Estrin. 2016. Immersive recommendation: News and event recommendations using personal digital traces. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee.

[22] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee, 263–272.

[23] Michael Inzlicht, Brandon J Schmeichel, and C Neil Macrae. 2014. Why self-control seems (but may not be) limited. *Trends in cognitive sciences* 18, 3 (2014), 127–133.

[24] Dietmar Jannach, Paul Resnick, Alexander Tuzhilin, and Markus Zanker. 2016. Recommender systems-beyond matrix completion. *Commun. ACM* 59, 11 (2016), 94–102.

[25] Jie Kang, Kyle Condiff, Shuo Chang, Joseph A Konstan, Loren Terveen, and F Maxwell Harper. 2017. Understanding how people use natural language to ask for recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 229–237.

[26] Bart P Knijnenburg, Saadhika Sivakumar, and Daricia Wilkinson. 2016. Recommender systems for self-actualization. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 11–14.

[27] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M Henne. 2009. Controlled experiments on the web: survey and practical guide. *Data mining and knowledge discovery* 18, 1 (2009), 140–181.

[28] Mounia Lalmas and Liangjie Hong. 2018. Tutorial on Metrics of User Engagement: Applications to News, Search and E-Commerce. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 781–782.

[29] Uichin Lee, Zhenyu Liu, and Junghoo Cho. 2005. Automatic identification of user goals in web search. In *Proceedings of the 14th international conference on World Wide Web*. ACM, 391–400.

[30] Leonard A Marascuilo and Joel R Levin. 1970. Appropriate post hoc comparisons for interaction and nested hypotheses in analysis of variance designs: The elimination of type IV errors. *American Educational Research Journal* 7, 3 (1970), 397–421.

[31] Katherine L Milkman, Dolly Chugh, and Max H Bazerman. 2009. How can decision making be improved? *Perspectives on psychological science* 4, 4 (2009), 379–383.

[32] Katherine L Milkman, Todd Rogers, and Max H Bazerman. 2008. Harnessing our inner angels and demons: What we have learned about want/should conflicts and how that knowledge can help us reduce short-sighted decision making. *Perspectives on Psychological Science* 3, 4 (2008), 324–338.

[33] Junta Mizuno, Jun Ogata, and Masataka Goto. 2008. A similar content retrieval method for podcast episodes. In *Spoken Language Technology Workshop, 2008. SLT 2008. IEEE*. IEEE.

[34] Tien T Nguyen, Pik-Mai Hui, F Maxwell Harper, Loren Terveen, and Joseph A Konstan. 2014. Exploring the filter bubble: the effect of using recommender systems on content diversity. In *Proceedings of the 23rd international conference on World wide web*. ACM, 677–686.

[35] Jun Ogata and Masataka Goto. 2009. PodCastle: Collaborative training of acoustic models on the basis of wisdom of crowds for podcast transcription. In *Tenth Annual Conference of the International Speech Communication Association*.

[36] Eli Pariser. 2011. *The filter bubble: How the new personalized web is changing what we read and how we think*. Penguin.

[37] Edison Research. 2017. The Podcast Consumer 2017. http://www.edisonresearch.com/the-podcast-consumer-2017/

[38] William W Ronan, Gary P Latham, and SB Kinne. 1973. Effects of goal setting and supervision on worker behavior in an industrial situation. *journal of Applied Psychology* 58, 3 (1973), 302.

[39] Daniel E Rose and Danny Levinson. 2004. Understanding user goals in web search. In *Proceedings of the 13th international conference on World Wide Web*. ACM, 13–19.

[40] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: debiasing learning and evaluation. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*. JMLR. org.

[41] Amit Sharma, Jake M Hofman, and Duncan J Watts. 2015. Estimating the causal impact of recommendation systems from observational data. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*. ACM, 453–470.

[42] Ana-Andreea Stoica, Christopher Riederer, and Augustin Chaintreau. 2018. Algorithmic Glass Ceiling in Social Networks: The effects of social recommendations on network diversity. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 923–932.

[43] Jessica Su, Aneesh Sharma, and Sharad Goel. 2016. The effect of recommendations on network structure. In *Proceedings of the 25th international conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1157–1167.

[44] Jaime Teevan, Susan T Dumais, and Daniel J Liebling. 2008. To personalize or not to personalize: modeling queries with variation in user intent. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 163–170.

[45] Choon Hui Teo, Houssam Nassif, Daniel Hill, Sriram Srinivasan, Mitchell Goodman, Vijai Mohan, and SVN Vishwanathan. 2016. Adaptive, personalized diversity for visual discovery. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 35–38.

[46] Sabina Tomkins, Steven Isley, Ben London, and Lise Getoor. 2018. Sustainability at scale: towards bridging the intention-behavior gap with sustainable recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 214–218.

[47] Manos Tsagkias, Martha Larson, and Maarten De Rijke. 2010. Predicting podcast preference: An analysis framework and its application. *Journal of the American Society for information Science and Technology* 61, 2 (2010), 374–391.

[48] Ryen W White and Steven M Drucker. 2007. Investigating behavioral variability in web search. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 21–30.

[49] Jacob O Wobbrock, Leah Findlater, Darren Gergle, and James J Higgins. 2011. The aligned rank transform for nonparametric factorial analyses using only anova procedures. In *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 143–146.

[50] Longqi Yang, Eugene Bagdasaryan, Joshua Gruenstein, Cheng-Kang Hsieh, and Deborah Estrin. 2018. OpenRec: A Modular Framework for Extensible and

Adaptable Recommendation Algorithms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 664–672.

[51] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin. 2018. Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 279–287.

[52] Longqi Yang, Cheng-Kang Hsieh, Hongjian Yang, John P Pollak, Nicola Dell, Serge Belongie, Curtis Cole, and Deborah Estrin. 2017. Yum-me: a personalized nutrient-based meal recommender system. *ACM Transactions on Information Systems (TOIS)* 36, 1 (2017), 7.

[53] Longqi Yang, Michael Sobolev, Christina Tsangouri, and Deborah Estrin. 2018. Understanding user interactions with podcast recommendations delivered via voice. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 190–194.

[54] Mike Yeomans, Anuj Shah, Sendhil Mullainathan, and Jon Kleinberg. 2016. Making sense of recommendations. *Preprint at http://scholar. harvard. edu/files/sendhil/files/recommenders55_01. pdf* (2016).

Learning from Multiple Cities: A Meta-Learning Approach for Spatial-Temporal Prediction

Huaxiu Yao
Pennsylvania State University
huaxiuyao@psu.edu

Yiding Liu
Nanyang Technological University
liuy0130@e.ntu.edu.sg

Ying Wei
Tencent AI Lab
judyweiying@gmail.com

Xianfeng Tang
Pennsylvania State University
tangxianfeng@outlook.com

Zhenhui Li
Pennsylvania State University
jessieli@ist.psu.edu

ABSTRACT

Spatial-temporal prediction is a fundamental problem for constructing smart city, which is useful for tasks such as traffic control, taxi dispatching, and environment policy making. Due to data collection mechanism, it is common to see data collection with unbalanced spatial distributions. For example, some cities may release taxi data for multiple years while others only release a few days of data; some regions may have constant water quality data monitored by sensors whereas some regions only have a small collection of water samples. In this paper, we tackle the problem of spatial-temporal prediction for the cities with only a short period of data collection. We aim to utilize the long-period data from other cities via transfer learning. Different from previous studies that transfer knowledge from one single source city to a target city, we are the first to leverage information from multiple cities to increase the stability of transfer. Specifically, our proposed model is designed as a spatial-temporal network with a meta-learning paradigm. The meta-learning paradigm learns a well-generalized initialization of the spatial-temporal network, which can be effectively adapted to target cities. In addition, a pattern-based spatial-temporal memory is designed to distill long-term temporal information (i.e., periodicity). We conduct extensive experiments on two tasks: traffic (taxi and bike) prediction and water quality prediction. The experiments demonstrate the effectiveness of our proposed model over several competitive baseline models.

CCS CONCEPTS

• **Information systems → Data Mining**; • **Computing methodologies → Transfer learning**.

KEYWORDS

Spatial-temporal prediction, periodicity, meta-learning

ACM Reference Format:
Huaxiu Yao, Yiding Liu, Ying Wei, Xianfeng Tang, and Zhenhui Li. 2019. Learning from Multiple Cities: A Meta-Learning Approach for Spatial-Temporal Prediction. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313577

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313577

1 INTRODUCTION

Recently, the construction of smart cities significantly changes urban management and services [12, 13, 25, 28, 42, 43]. One of the most fundamental techniques in building smart city is accurate spatial-temporal prediction. For example, a traffic prediction system can help the city pre-allocate transportation resources and control traffic signal intelligently. An accurate environment prediction system can help the government develop environment policy and then improve the public's health.

Traditionally, basic time series models (e.g., ARIMA, Kalman Filtering and their variants) [10, 15, 19], regression models with spatial-temporal regularizations [6, 44] and external context features [30, 35] are used for spatial-temporal prediction. Recently, advanced machine learning methods (e.g., deep neural network based models) significantly improve the performance of spatial-temporal prediction [34, 35, 41] by characterizing non-linear spatial-temporal correlations more accurately. Unfortunately, the superior performance of these models is conditioned on large-scale training data which are probably inaccessible in real-world applications. For example, there may exist only a few days of GPS traces for traffic prediction in some cities.

Transfer learning has been studied as an effective solution to address the data insufficiency problem, by leveraging knowledge from those cities with abundant data (e.g., GPS traces covering a few years). In [29], the authors proposed to transfer semantically-related dictionaries learned from a data-rich city, i.e., a source city, to predict the air quality category in a data-insufficient city, i.e., a target city. The method proposed in [27] aligns similar regions across source and target cities for finer-grained transfer. However, these methods, transferring the knowledge from only a single source city, would cause unstable results and the risk of negative transfer. If the underlying data distributions are significantly different between cities, the knowledge transfer will make no contribution or even hurt the performance.

To reduce the risk, in this paper, we focus on transferring knowledge from *multiple source cities* for the spatial-temporal prediction in a target city. Compared with single city, the knowledge extracted from multiple cities covers more comprehensive spatial-temporal correlations of a city, e.g., temporal dependency, spatial closeness, and region functionality, and thus increases the stability of transfer. However, this problem faces two key challenges.

- **How to adapt the knowledge to meet various scenarios of spatial-temporal correlations in a target city?** The spatial-temporal correlations in the limited data of a target city may vary considerably from city to city and even from time to time. For example, the knowledge to be transferred to New York is expected to differ from that to Boston. In addition, the knowledge to be transferred within the same city also differs from weekdays to weekends. Thus a sufficiently flexible algorithm capable of adapting the knowledge learned from multiple source cities to various scenarios is required.
- **How to capture and borrow long-period spatial-temporal patterns from source cities?** It is difficult to capture long-term spatial-temporal patterns (e.g., periodicity) accurately in a target city with limited data due to the effects of special events (e.g. holiday) or missing values. These patterns as indicators of region functionality, however, are crucial for spatial-temporal prediction [33, 45]. Take the traffic demand prediction as an instance. The traffic demand in residential areas is periodically high in the morning when people transit to work. Thus, it is promising but challenging to transfer such long-term patterns from source cities to a target city.

To address the challenges, we propose a novel framework for spatial-temporal prediction, namely **MetaST**. It is the first to incorporate the meta-learning paradigm into spatial-temporal networks (ST-net). The ST-net consists of a local CNN and an LSTM which jointly capture spatial-temporal features and correlations. With the meta-learning paradigm, we solve the first challenge by learning a well-generalized initialization of the ST-net from a large number of prediction tasks sampled from multiple source cities, which covers comprehensive spatial-temporal scenarios. Subsequently, the initialization can easily be adapted to a target city via fine-tuning, even when only few training samples are accessible. Second, we learn a global pattern-based spatial-temporal memory from all source cities, and transfer it to a target city to support long-term patterns. The memory, describing and storing long-term spatial-temporal patterns, is jointly trained with the ST-net in an end-to-end manner.

We evaluate the proposed framework on several datasets including taxi volume, bike volume, and water quality. The results show that our proposed MetaST consistently outperforms several baseline models. We summarize our contributions as follows.

- To the best of our knowledge, we are the first to study the problem of transferring knowledge from multiple cities for the spatial-temporal prediction in a target city.
- We propose a novel MetaST framework to solve the problem by combining a spatial-temporal network with the meta-learning paradigm. Moreover, we learn from all source cities a global memory encrypting long-term spatial-temporal patterns, and transfer it to further improve the spatial-temporal prediction in a target city.
- We empirically demonstrate the effectiveness of our proposed MetaST framework on three real-world spatial-temporal datasets.

The rest of this paper is organized as follows. We first review and discuss the related work in Section 2. Then, we define some notations and formulate the problem in Section 3. After that, we introduce details of the framework of MetaST in Section 4. We apply our model on three real-world datasets from two different domains and conduct extensive experiments in Section 5. Finally, we conclude our paper in Section 6.

2 RELATED WORK

In this section, we briefly review the works in two categories: some representative works for spatial-temporal prediction and knowledge transferring.

2.1 Spatial-Temporal Prediction

The earliest spatial-temporal prediction models are based on basic time series models (e.g., ARIMA, Kalman Filtering) [10, 15, 19]. Recent studies further utilize external context data (e.g., weather condition, venue types, holiday, event information) [26, 30, 35] to enhance the prediction accuracy. Also, spatial correlation is taken into consideration by designing regularization of spatial smoothness [21, 31, 44].

Recently, various deep learning methods have been used to capture complex non-linear spatial-temporal correlations and predict spatial-temporal series, such as stacked fully connected network [24, 35], convolutional neural network (CNN) [23, 41] and recurrent neural network (RNN) [39]. Several hybrid models have been proposed to model both spatial and temporal information [7, 33, 34]. These methods combine CNN and RNN, and achieve the state-of-the-art performance on spatial-temporal prediction. In addition, based on the road network structure, another type of hybird models combines graph aggregation mechanism and RNN for spatial-temporal prediction [9, 38, 40]

Different from previous studies of deep spatial-temporal prediction which all rely on a large set of training samples, we aim to transfer learned knowledge from source cities to improve the performance of spatial-temporal prediction in a target city with limited data samples.

2.2 Knowledge Transfer and Reuse

Transfer learning and its related fields utilize previously learned knowledge to facilitate learning in new tasks when labeled data is scarce [16, 17]. Previous transfer learning methods transfer different information from a source domain to a target domain, such as parameters [32], instances [1], manifold structures [3, 4], deep hidden feature representations [22, 37]. Recently, meta-learning (a.k.a., learning to learn) transfers shared knowledge from multiple training tasks to a new task for quick adaptation. These techniques include learning a widely generalizable initialization [2, 11], optimization trace [18], metric space [20], transfer learning skills [36].

However, only a few attempts have been made on transferring knowledge on the space. [29] proposes a multi-modal transfer learning framework for predicting air quality category, which combines multi-modal data by learning a semantically coupled dictionary for multiple modalities in a source city. *However, this method works on multimodal features instead of spatial-temporal sequences we focus on. Therefore, it cannot be applied to solve the problem.* For predicting traffic flow, [27] leverages the similarity of check-in records/spatial-temporal sequences between a source city and a target city to construct the similarity regularization for knowledge transfer. *Different from this method that intelligently learns the correlation which could be linear or non-linear. Compared with both*

methods above, in addition, our model transfers the shared knowledge from multiple cities to a new city, which increase the stability of transfer and prediction.

3 DEFINITIONS AND PROBLEM FORMULATION

In this section, we define some notations used in this paper and formally define the setup of our problem.

Definition 1 (Region) Following previous works [34, 41], we spatially divide a city c into an $I_c \times J_c$ grid map which contains I_c rows and J_c columns. We treat each grid as a region r_c, and define the set of all regions as $\mathcal{R}_c$.

Definition 2 (Spatial-Temporal Series) In city c, we denote the current/latest timestamp as k_c, and the time range as a set $\mathcal{K}_c = \{k_c - |\mathcal{K}_c| + 1, ..., k_c\}$ consisting of $|\mathcal{K}_c|$ evenly split non-overlapping time intervals. Then, the spatial-temporal series in city c is represented as

$$\mathcal{Y}_c = \{y_{r_c,k_c} | r_c \in \mathcal{R}_c, k_c \in \mathcal{K}_c\}, \tag{1}$$

where y_{r_c,k_c} is the spatial-temporal information to be predicted (e.g., traffic demand, air quality, climate value).

Problem Definition Suppose that we have a set of source cities $C_s = \{c_1, ..., c_S\}$ and a target city c_t with insufficient data (i.e., $\forall s \in \{1, \cdots, S\}$, $|\mathcal{K}_{c_s}| \gg |\mathcal{K}_{c_t}|$), our goal is to predict the spatial-temporal information of the next timestamp $k_{c_t} + 1$ in the testing dataset of the target city c_t, i.e.,

$$y^*_{r_{c_t},k_{c_t}+1} = \arg\max_{y_{r_{c_t},k_{c_t}+1}} p(y_{r_{c_t},k_{c_t}+1} | \mathcal{Y}_{c_t}, f_{\theta_0}), \tag{2}$$

where f represents the ST-net which serves as the base model to predict the spatial-temporal series. Detailed discussion about ST-net is in given Section 4.1. More importantly, in the meta-learning paradigm, θ_0 denotes the initialization for all parameters of the ST-net, which is adapted from $\mathcal{Y}_{c_1}, \cdots, \mathcal{Y}_{c_S}$.

4 METHODOLOGY

In this section, we elaborate our proposed method **MetaST**. In particular, we first introduce the ST-net as the base model f for spatial-temporal prediction, and then present our proposed spatial-temporal meta-learning framework for knowledge transfer.

4.1 Spatial-Temporal Network

Recently, hybrid models combining convolution neural networks (CNN) [8] and LSTM [5] have achieved the state-of-the-art performances on spatial-temporal prediction. Thus, following [34], we adopt a CNN to capture the spatial dependency between regions, and an LSTM to model the temporal evolvement of each region.

In city c, for each region r_c at time k_c, we regard the spatial-temporal value of this region and its surrounding neighbors as an $N \times N$ image with v channels $\mathbf{Y}_{r_c,k_c} \in \mathbb{R}^{N \times N \times v}$, where region r_c is in the center of this image. For instance, when N=3, we are considering a center cell as well as 8 adjacent grid cells of the cell, which is a 3*3 size neighborhood. The number of channels v depends on a specific task. For example, in taxi volume prediction, we jointly predict taxi pick-up volume and drop-off volume, so that the number of channels equals two, i.e., $v = 2$. Taking $\mathbf{Y}_{r_c,k_c}$ as input $\mathbf{Y}^0_{r_c,k_c}$, a local CNN computes the q-th layer progressively:

$$\mathbf{Y}^q_{r_c,k_c} = \text{ReLU}(\mathbf{W}^q_r * \mathbf{Y}^{q-1}_{r_c,k_c} + \mathbf{b}^q_r), \tag{3}$$

where $*$ represents the convolution operation, $\mathbf{W}^q_r$ and $\mathbf{b}^q_r$ are learnable parameters. After Q convolutional layers, a fully connected layer following a flatten layer is used to infer the spatial representation of region r_c as $\mathbf{s}_{r_c,k_c}$ eventually.

Then, for predicting y_{r_c,k_c+1}, we model the temporal unfolding of region r_c by passing all the spatial representations along the time span $\{k_c - |\mathcal{K}_c| + 1, ..., k_c\}$ through an LSTM, which can be formulated as

$$\begin{aligned}
\mathbf{i}_{r_c,k_c} &= \sigma(\mathbf{U}_i[\mathbf{s}_{r_c,k_c}; \mathbf{e}_{r_c,k_c}] + \mathbf{W}_i \mathbf{h}_{r_c,k_c-1} + \mathbf{b}_i), \\
\mathbf{f}_{r_c,k_c} &= \sigma(\mathbf{U}_f[\mathbf{s}_{r_c,k_c}; \mathbf{e}_{r_c,k_c}] + \mathbf{W}_f \mathbf{h}_{r_c,k_c-1} + \mathbf{b}_f), \\
\mathbf{d}_{r_c,k_c} &= \sigma(\mathbf{U}_d[\mathbf{s}_{r_c,k_c}; \mathbf{e}_{r_c,k_c}] + \mathbf{W}_d \mathbf{h}_{r_c,k_c-1} + \mathbf{b}_d), \\
\hat{\mathbf{c}}_{r_c,k_c} &= \tanh(\mathbf{U}_c[\mathbf{s}_{r_c,k_c}; \mathbf{e}_{r_c,k_c}] + \mathbf{W}_c \mathbf{h}_{r_c,k_c-1} + \mathbf{b}_c), \\
\mathbf{c}_{r_c,k_c} &= \mathbf{f}_{r_c,k_c} \circ \mathbf{c}_{r_c,k_c-1} + \mathbf{i}_{r_c,k_c} \circ \hat{\mathbf{c}}_{r_c,k_c}, \\
\mathbf{h}_{r_c,k_c} &= \tanh(\mathbf{c}_{r_c,k_c}) \circ \mathbf{d}_{r_c,k_c},
\end{aligned} \tag{4}$$

where $\circ$ denotes Hadamard product, $\mathbf{U}_j$, $\mathbf{W}_j$, and $\mathbf{b}_j$ ($j \in \{i, f, d, c\}$) are learnable parameters, $\mathbf{i}_{r_c,k_c}$, $\mathbf{f}_{r_c,k_c}$, and $\mathbf{d}_{r_c,k_c}$ are forget gate vector, input gate vector, and output gate vector of the i-th context feature, respectively. $\mathbf{h}_{r_c,k_c}$ denotes the spatial-temporal representation of region r_c, and $|\mathcal{K}_c|$ is the number of time steps we use to consider the temporal information. Note that $\mathbf{e}_{r_c,k_c}$ represents other external features (e.g., weather, holiday) that can be incorporated, if applicable. By doing these, $\mathbf{h}_{r_c,k_c}$ encodes both the spatial and temporal information of region r_c. As a result, the value of the next time step of spatial-temporal series, i.e., $\hat{y}_{r_c,k_c+1}$, can be predicted by

$$\hat{y}_{r_c,k_c+1} = \tanh(\mathbf{W}_n \mathbf{h}_{r_c,k_c} + \mathbf{b}_n), \tag{5}$$

where $\mathbf{W}_n$ and $\mathbf{b}_n$ are learnable parameters. The output is scaled to (-1,1) via a $\tanh(\cdot)$ function, to be consistent with the normalized spatial-temporal values. We later denormalize the prediction to get the actual demand values. We formulate the loss function of ST-net for each city c as:

$$\mathcal{L}_c = \sum_{r_c} \sum_{k_c} (\hat{y}_{r_c,k_c+1} - y_{r_c,k_c+1})^2, \tag{6}$$

so as to enforce the estimated spatial-temporal value to be as close to the ground-truth y_{r_c,k_c+1} as possible. As introduced previously, we denote all the parameters of the ST-net as θ, and the ST-net parameterized by θ as f_θ. For better illustration, we visualize the structure of the spatial-temporal network (**ST-net**) in Figure 1.

4.2 Knowledge Transfer

Next, we propose a meta-learning framework that enables our ST-net model to borrow knowledge from multiple cities. The framework consists of two parts: adapting the initialization and learning the spatial-temporal memory. We present the whole framework in Figure 2.

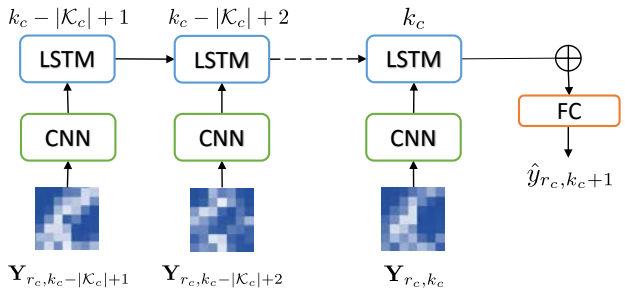


Figure 1: The framework of the ST-net. The heatmaps are exemplar spatial-temporal values. CNN is used to capture spatial dependency and then LSTM is used to handle temporal correlation.

4.2.1 Adapting the Initialization. As we described before, we are supposed to increase the stability of prediction by transferring knowledge from multiple source cities. Since the spatial-temporal correlation of limited data in a target city may vary considerably from city to city and even from time to time. For example, in traffic prediction, the knowledge to be transferred to Boston with limited weekend data is expected to differ from that to Chicago with limited weekday data. Thus, the extracted knowledge from multiple cities is expected to include comprehensive spatial-temporal correlations such as spatial closeness and temporal dependency, so that we can adapt the knowledge to a target city with limited data under different scenarios.

In ST-net, the parameters θ are exactly the knowledge which encrypts spatial-temporal correlations. To effectively adapt the parameters to a target city, as suggested in model-agnostic meta-learning (MAML) [2], initialization of θ from multiple source cities, i.e., θ_0, so that the ST-net initialized by θ_0 achieves the minimum of the average of generalization losses over all source cities, i.e.,

$$\theta_0 = \min_{\theta_0} \sum_{c_s \in C_s} \mathcal{L}'_{c_s}(f_{\theta_0 - \alpha \nabla_\theta \mathcal{L}_{c_s}(f_\theta)}). \tag{7}$$

Here we would note that $\mathcal{L}_{c_s}(f_{\theta_{c_s}})$ denotes the training loss on the training set of a city c_s sampled from C_s, i.e., $\mathcal{D}_{c_s}$ (refer to S-train in Figure 2). We illustrate the iterative update of the parameters θ_{c_s} during the training process (shown as the yellow solid arrow in Figure 2), by showing one exemplar gradient descent, i.e.,

$$\theta_{c_s} = \theta_0 - \alpha \nabla_\theta \mathcal{L}_{c_s}(f_\theta). \tag{8}$$

In practice, we can use several steps of gradient descent to update from the initialization θ_0 to θ_{c_s}. For each city c_s, the training process is repeated on batches of series sampled from D_{c_s}.

Since Eq. (7) minimizes the generalization loss, $\mathcal{L}'_{c_s}(\cdot)$ evaluates the loss on the test set of the city c_s, i.e., $\mathcal{D}'_{c_s}$ (refer to S-test in Figure 2). By optimizing the problem in Eq. (7) using stochastic gradient descent (shown as the purple solid arrow in Figure 2), we obtain an initialization which can generalize well on different source cities. Therefore, it is widely expected that transferring the initialization θ_0 to a target city c_t would also bring superior generalization performance, which we will detail in the end of this section.

4.2.2 Spatial-Temporal Memory. In spatial-temporal prediction, long-term spatial-temporal patterns (e.g., periodic patterns) play an important role [33, 45]. These patterns reflect the spatial functionality of each region and can be regarded as the global property shared by different cities. An example of spatial-temporal patterns and their corresponding regions are shown in Figure 3. In this figure, one region around NYU-Tardon in New York City and another one around Georgetown University in Washington DC are selected. The averaged 5-days' taxi demand distributions of both regions are daily periodic and similar, whose value are higher in the afternoon when students and faculties go back to home. The similarity of distributions between different cities verifies our assumption, i.e., spatial functionality is globally shared. However, in a target city, these patterns are hard to be captured with limited data due to missing values or the effects of special events (e.g., holidays). Thus, we propose a global memory, named **ST-mem**, to store long-term spatial-temporal patterns and further transfer to improve prediction accuracy in target cities. The framework of ST-Mem is illustrated in Figure 4.

Based on spatial-temporal patterns, we first cluster all the regions in source cities to G categories. The categories G of regions represent different region functionalities. For region r_c, the clustering results are denoted as $\mathbf{o}_{r_c} \in \mathbb{R}^G$. If region r_c belongs to cluster g, $\mathbf{o}_{r_c}(g) = 1$, otherwise $\mathbf{o}_{r_c}(g) = 0$. Then, we construct a parameterized spatial-temporal memory $\mathcal{M} \in \mathbb{R}^{G \times d}$. Each row of the memory stores the pattern representation of a category, and the dimension of the pattern representation is d.

Next, we utilize the knowledge of patterns stored in memory $\mathcal{M}$, and distill this knowledge for prediction via attention mechanism [14]. Since we only have short-term data in a target city, we use the representation of short-term data to query ST-mem. In particular, we linearly project the spatial-temporal representation $\mathbf{h}_{r_c,k_c}$ of ST-net to get the query vector for the attention mechanism, which is formulated as:

$$\mathbf{v}_{r_c,k_c} = \mathbf{W}_l \mathbf{h}_{r_c,k_c} + \mathbf{b}_l. \tag{9}$$

Then, we use the query vector $\mathbf{v}_{r_c,k_c} \in \mathbb{R}^d$ to calculate the similarity score between it and the pattern representation for each category g. Formally, the similarity score is defined as

$$\mathbf{p}_{r_c,k_c}(g) = \frac{\exp(\langle \mathbf{v}_{r_c,k_c}, \mathcal{M}(g) \rangle)}{\sum_{g'=1}^{G} \exp(\langle \mathbf{v}_{r_c,k_c}, \mathcal{M}(g') \rangle)}, \tag{10}$$

where $\mathcal{M}(g)$ means the g-th row of memory (i.e., for the g-th pattern category). We calculate the representation of spatial-temporal pattern $\mathbf{z}_{r_c,k_c}$ as:

$$\mathbf{z}_{r_c,k_c} = \sum_{g=1}^{G} \mathbf{p}_{r_c,k_c}(g) * \mathcal{M}(g). \tag{11}$$

Then, we concatenate the representation $\mathbf{z}_{r_c,k_c}$ of spatial-temporal patterns with the representation $\mathbf{h}_{r_c,k_c}$ of ST-net. And the input of the final layer $\mathbf{h}_{r_c,k_c}$ in Eq. (5) is replaced by the enhanced representation i.e.,

$$\hat{y}_{r_c,k_c+1} = \tanh(\mathbf{W}'_n[\mathbf{h}_{r_c,k_c}; \mathbf{z}_{r_c,k_c}] + \mathbf{b}'_n). \tag{12}$$

where $\mathbf{W}'_n$ and $\mathbf{b}'_n$ are learnable parameters.

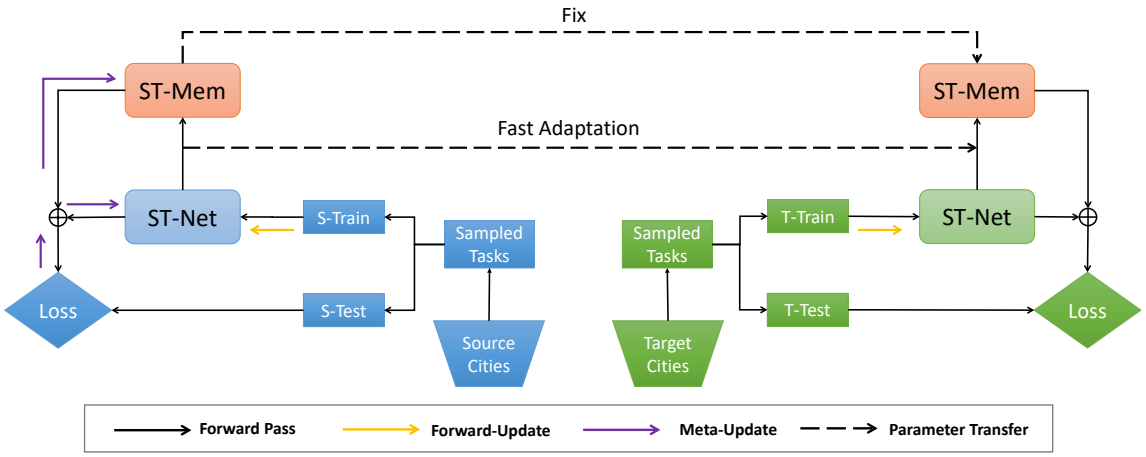


Figure 2: The framework of proposed MetaST. ST-net and ST-mem mean spatial-temporal network and spatial-temporal memory. S-train and S-test, T-train and T-test represent the training and testing set of source tasks (i.e., source cities) and target tasks (i.e., source cities), respectively. In the process of knowledge transfer, the parameters of ST-net will be updated by the training set in target city (i.e., T-train), while the parameters of ST-mem are fixed.

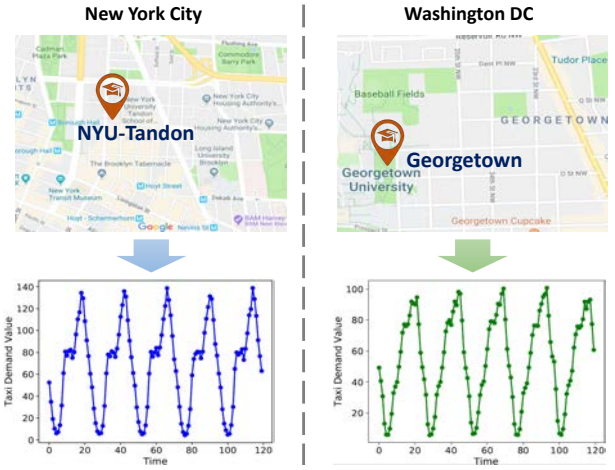


Figure 3: The illustration of spatial-temporal patterns of two regions and their corresponding region functionality.

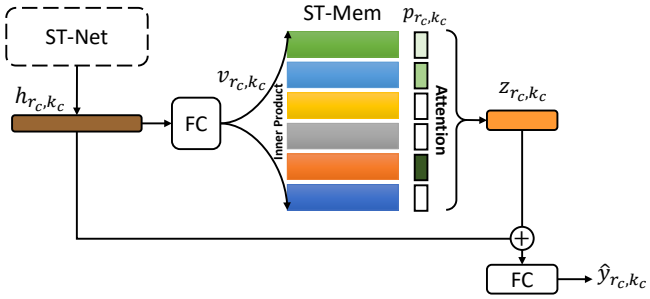


Figure 4: The stucture of ST-mem. The spatial-temporal representation $\mathbf{h}_{r_c,k_c}$ is projected as a query vector $\mathbf{v}_{r_c,k_c}$. Then, the attention mechanism is used to get the pattern representation $\mathbf{z}_{r_c,k_c}$. The short-term spatial-temporal representation and pattern representation are concatenated together for prediction.

In order to learn the memory $\mathcal{M}$, we construct the clustering loss of city c_s, which enforce the attention scores to be consistent with previous clustering results $\mathbf{o}_{r_c}$. The formulation of the clustering loss is as follows:

$$\mathcal{L}_{clu} = -\sum_{r_c}\sum_{k_c} \mathbf{o}_{r_c} \log(\mathbf{p}_{r_c,k_c}). \tag{13}$$

Additionally, the memory $\mathcal{M}$ is also updated when we train the MetaST framework, together with the initialization θ_0. Thus, we revise the loss function in Eq. (7) by adding the clustering loss. Then, our final objective function is:

$$\theta_0, \mathcal{M} = \min_{\theta_0, \mathcal{M}} \sum_{c_s \in C_s} \gamma \mathcal{L}_{clu}(f_{\theta_{c_s}}) + \mathcal{L}'_{c_s}(f_{\theta_{c_s}}), \tag{14}$$

where γ is a trade-off hypeparameter and is used to balance the effect of each part. $\theta_{c_s} = \theta_0 - \alpha\nabla_\theta \mathcal{L}_{c_s}(f_\theta)$ is defined in Eq. (8). By using testing set $\mathcal{D}'_{c_s}$ of each city c_s, the memory $\mathcal{M}$ and initialization θ_0 are updated by gradient descent. Note that, as we discussed before, the spatial-temporal patterns are common property between cities. We do not update memory $\mathcal{M}$ when training a specific task (i.e., Eq. (8)). In Figure 2, the purple solid arrow indicates the process of meta-update. In Eq. (14), the initialization θ_0 and memory $\mathcal{M}$ are mutually constrained. Since the memory $\mathcal{M}$ provides region-level relationship via spatial-temporal patterns, it can also help learn the initialization θ_0 of ST-net.

4.2.3 Transfer Knowledge to Target Domain. To improve the prediction in target cities, we transfer the ST-mem $\mathcal{M}$ and initialization θ_0 of ST-net. The black dotted line in Figure 2 shows the process of knowledge transfer. For each new city c_t sampled from C_t, the

Algorithm 1: Framework of MetaST

Input: Set of target cities C_t; Set of source cities C_s; hyperparameter γ
Output: Spatial-temporal predictions of each target city

Randomly initialize θ_0 and $\mathcal{M}$;
Cluster all regions in C_s and get $\mathbf{o}_{r_c}$ for each region r_c;
/* learning θ_0 and $\mathcal{M}$ on source cities */
while *not done* **do**
  Sample a batch of cities from C_s;
  for *each city* c_s **do**
    Sample a set $\mathcal{D}_{c_s}$ from c_s;
    Evaluate $\nabla\mathcal{L}_{c_s}(f_{\theta_{c_s}})$ using $\mathcal{D}_{c_s}$ by Eq. (6);
    Compute adapted parameters θ_{c_s} with gradient descent by Eq. (8);
    Sample a new set $\mathcal{D}'_{c_s}$ from c_s;
    Evaluate $\mathcal{L}'_{c_s}(f_{\theta_s})$, $\mathcal{L}_{clu}(f_{\theta_{c_s}})$ by $\mathcal{D}'_{c_s}$;
  end
  Update θ_0, $\mathcal{M}$ by gradient descent;
end
/* Evaluate our model on target cities */
for *each city* c_t *in* C_t **do**
  Sample a sample set $\mathcal{D}_{c_t}$ from c_t;
  Compute adapted parameters θ_{c_t} with gradient descent by Eq. (15);
  Sample new series $\mathcal{D}'_{c_t}$ and predict;
end

memory is fixed and the parameters θ_{c_t} are trained with initialization θ_0 and training samples $\mathcal{D}_{c_t}$ (i.e., T-train in Figure 2), which is defined as:

$$\theta_{c_t} = \theta_0 - \alpha\nabla_\theta\mathcal{L}_{c_t}(f_\theta), \tag{15}$$

where $\mathcal{L}_{c_t}$ is the loss function of training set in target city t and the formulation is:

$$\mathcal{L}_{c_t} = \sum_{r_t}\sum_{k_t}(\hat{y}_{r_t,k_t+1} - y_{r_t,k_t+1})^2. \tag{16}$$

The predicted value $\hat{y}_{r_t,k_t+1}$ is calculated via Eq. (12). Hence, we distill knowledge of spatial-temporal patterns from source cities to target city via ST-mem $\mathcal{M}$. Finally, we evaluate the model $f_{\theta_{c_t}}$ on testing set $\mathcal{D}'_{c_t}$ (i.e., T-test in Figure 2) of city c_t and get the prediction value of this city. The whole framework of MetaST is shown in Algorithm 1.

5 EXPERIMENT

In this section, we conduct extensive experiments for two domain applications to answer the following research questions:

- Whether MetaST can outperform baseline methods for inference tasks, i.e., traffic volume prediction in and water quality (pH value) prediction?
- How do various hyper-parameters, e.g., the dimensions of each cluster in ST-mem or trade-off factor, impact the model's performance?
- Whether ST-mem can detect distinguished spatial-temporal patterns?

Table 1: Data Statistics of Traffic Prediction

Data	City	Time Span (m/d/y)	Trips	Size
Taxi	NYC	1/1/15-7/1/15	6,748,857	10×20
	DC	5/1/15-1/1/16	8,151,077	16×16
	Porto	7/1/13 - 6/30/14	1,710,671	20×10
	CHI	9/1/13-11/1/14	124,820	15×18
	BOS	10/1/12 - 10/31/12	839,897	18×15
Bike	NYC	1/1/17-12/31/17	16,364,475	10×20
	DC	1/1/17-12/31/17	3,732,263	16×16
	CHI	1/1/17-2/1/17	106,165	15×18

5.1 Application-*I*: Traffic Prediction

5.1.1 Problem Overview. We first conduct experiments on two traffic prediction tasks, i.e., taxi volume prediction and bike volume prediction. Similar as the previous traffic prediction task in [33, 41], each individual trip departs from a region, and then arrives at the destination region. Thus, our task is to predict the pick-up (start) and drop-off (end) volume of taxi (and bike) at each time interval for each region. The time inveral of traffic prediction task is one hour. We use *Root Mean Square Error* (RMSE) to evaluate our model, which is defined as:

$$\text{RMSE} = \frac{1}{|N|}\sqrt{\sum_{r_t}\sum_{k_t}(\hat{y}_{r_t,k_t+1} - y_{r_t,k_t+1})_2^2}, \tag{17}$$

where $\hat{y}_{r_t,k_t+1}$ and y_{r_t,k_t+1} represent predicted value and actual value, respectively. $|N|$ means the number of all samples in testing set.

5.1.2 Data Description. For taxi volume prediction, we collect five representative mobility datasets from five different cities to evaluate the performance of our proposed model, i.e., New York City (NYC)[1], Washington DC (DC), Chicago (CHI), Porto[2], and Boston (BOS). We use NYC, DC, Porto as the source cities and CHI, BOS as the target cities. Note that the Boston data does not have drop-off records, and thus we only report the results on predicting pick-up volume.

For bike volume prediction, we collect three datasets from four cities, i.e., NYC[3], DC[4], and CHI[5]. NYC and DC are used as source cities, and CHI are used as target city. All trips in the above datasets contain time and geographic coordinates of pick-up and drop-off. For each city above, as discussed before, we spatially divide them to a grid map. The grid map size of NYC, DC, CHI, BOS, Porto is 10×20, 16×16, 15×18, 18×15, 20×10, respectively. Detailed statistics of these datasets are listed in Table 1.

In addition, for each source city, we select 80% data for training and validation, and the rest for testing. For each target city, we select the 1-day, 3-day and 7-day data for training, and the rest for testing.

[1] http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml
[2] https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i/data
[3] https://www.citibikenyc.com/system-data
[4] https://www.capitalbikeshare.com/system-data
[5] https://www.divvybikes.com/system-data

5.1.3 Compared Baselines. We compare our model with the following two categories of methods: non-transfer methods and transfer methods. Note that, for non-transfer baselines, we only use the training data of target cities (limited training data) to train the model and use the testing data to evaluate it. For transfer baselines, we transfer the learned knowledge from source cities to improve the prediction accuracy.

Non-transfer baselines:

- **Historical Average (HA)**: For each region, HA predicts spatial-temporal value based on the average value of the previous relative time. For example, if we want to predict the pick-up volume at 9:00am-10:00am, HA is the average value of all time intervals from 9:00am to 10:00am in training data.
- **ARIMA**: Auto-Regressive Integrated Moving Average (ARIMA) is a traditional time-series prediction model, which considers the temporal relationship of data.
- **ST-net**: This method simply use the spatial-temporal neural network defined in Section 4.1 to predict traffic volume. Both pick-up and drop-off volume are predicted together.

Transfer baselines:

- **Fine-tuning Methods**: We include two types of fine-tuning methods: (1) Single-source fine-tuning (**Single-FT**): train ST-net in one source city (e.g., NYC, DC or Porto in taxi data) and fine-tune the model for target cities; and (2) multi-source fine-tune (**Multi-FT**): mix all samples from all source cities and fine-tune the model in target cities.
- **RegionTrans [27]**: RegionTrans transfers knowledge from one city to another city for traffic flow prediction. Since we do not have auxiliary data, we compare the S-Match of RegionTrans. For each region in target city, RegionTrans uses short period data to calculate the linear similarity value with each region in source city. Then, the similarity is used as regularization for fine-tuning. For fair comparison, the base model of RegionTrans (i.e., the ST-net) is same as MetaST.
- **MAML [2]**: an state-of-the-art meta-learning method, which learns a better initialization from multiple tasks to supervise the target task. MAML uses the same base model (i.e., the ST-net) as MetaST.

5.1.4 Experimental Settings.

Hyperparameter Setting. For ST-net, we set the number of filters in CNN as 64, the size of map in CNN as 7×7, the number of steps in LSTM as 8, and the dimension of hidden state of LSTM as 128. In the training process of taxi volume prediction, we set the learning rate of inner loop and outer loop as 10^{-5} and 10^{-5} respectively. For bike volume prediction, we set the learning rate of inner loop and outer loop as 10^{-5} and 10^{-6} respectively. The parameter γ is set as 10^{-4}. The number of updates for each task is set as 5. All the models are trained by Adam. The training batch size for each meta-iteration is set as 128, and the maximum of iteration of meta-learning is set as 20000.

Spatial-temporal Clustering. In addition, since the pattern of traffic volume usually repeats every day. Thus, in this work, we use averaged 24-hour patterns of each region to represent its spatial-temporal pattern. We use K-means to cluster these patterns to 4 groups. Furthermore, in this work, we do not use other external features, which means that $[\hat{\mathbf{y}}_{r_c,k_c};\mathbf{e}_{r_c,k_c}]$ in Eq. (4) equals to $\hat{\mathbf{y}}_{r_c,k_c}$. We set the size of pattern representation in memory as 8.

5.1.5 Results. We implement our model and compare it with the baselines on taxi and bike-sharing datasets. We run 20 testing times and report the average values. The results are shown in Table 2 and Table 3, respectively. According to these tables, we draw the following conclusions.

- Comparing with ST-net and some single-FT models, in some cases (e.g., 1-day training data), traditional time-series prediction methods (i.e., HA and ARIMA) achieves competitive performance in this problem. The reason is that traffic data show a strong daily periodicity, so that if we only have limited traffic data, we can still use periodicity to predict traffic volume.
- Comparing with ST-net, all transfer learning models (i.e., fine-tune models (including Single-FT and Multi-FT models), MAML, RegionTrans, MetaST) significantly improves the performance (i.e., lower the RMSE values). The results indicate that (1) it is difficult to train a model from random initialization with limited data; (2) the knowledge transfer between cities is effective for prediction.
- In most cases, Multi-FT outperforms Single-FT. One possible reason is that Multi-FT increases the diversity of source domain. In other cases (e.g., Chicago pick-up prediction with 3-day training data), the best performance of Single-FT outperforms Multi-FT. The results implicitly indicates that if there exists a source city that is optimally transferable to the target city, simply mixing other cities may hurt the performance.
- RegionTrans models only slightly outperform their corresponding fine-tuning models (i.e., RegionTrans from NYC to Chicago v.s., Single-FT from NYC to Chicago). The results suggest that regional linear similarity regularization may not capture complex relationship between regions. In addition, since the data from different cities are collected from different time, regional similarity calculations may be inaccurate. Thus, it is not an effective and flexible way to transfer knowledge via regional similarity regularization.
- MAML and MetaST achieve better performance than fine-tuning methods and RegionTrans. This is because fine-tuning methods and RegionTrans cannot be adapted to different scenarios, and thereby decreasing the stability of transfer. However, MAML and MetaST not only learn the initialization based on multiple cities, but also achieve the best performance in every scenario sampled from these cities.
- Our proposed MetaST achieves the best performance in all experimental settings. Comparing with MAML, the averaged relative improvement is 5.0%. This is because our model can also capture and transfer long-term information, besides learning a better initialization. Moreover, the long-term pattern memory helps learn a further enhanced initialization. The stability of knowledge transfer increases to the highest degree.
- Finally, we compare the results under different training data size in target city (i.e., 1-day, 3-day, and 7-day data). For every learning models (except HA and ARIMA), the performance improves with more training data. Our proposed MetaST still outperforms all baselines.

Table 2: Comparing with baselines for taxi volum prediction

Taxi Data		Chicago						Boston		
		Pick-up			Drop-off			Pick-up		
		1-day	3-day	7-day	1-day	3-day	7-day	1-day	3-day	7-day
HA		2.83	2.36	2.18	2.67	2.28	2.13	11.07	9.13	7.71
ARIMA		3.19	2.76	2.71	2.93	2.43	2.41	11.02	10.25	9.36
ST-net		10.51	6.04	3.89	11.22	6.42	3.99	30.01	17.02	13.28
Single-FT	NYC	2.72	2.06	1.76	2.84	2.75	1.89	11.71	10.62	7.76
	DC	3.90	2.62	2.05	4.17	2.19	2.15	14.39	12.32	9.19
	Porto	2.57	1.87	1.60	2.87	2.03	1.74	14.12	11.56	7.63
Multi-FT		2.18	1.89	1.60	2.20	2.08	1.69	9.90	9.41	6.68
RegionTrans	NYC	2.53	2.01	1.69	2.83	2.56	1.72	11.12	10.49	7.49
	DC	3.87	2.51	2.04	3.95	2.16	2.03	13.98	11.83	8.87
	Porto	2.45	1.83	1.60	2.85	1.98	1.73	13.05	10.97	7.43
MAML		2.01	1.78	1.52	2.10	1.92	1.66	8.13	7.59	5.88
MetaST		**1.95****	**1.70****	**1.48****	**2.04****	**1.79****	**1.65***	**7.48****	**7.15****	**5.67****

** (*) means the result is significant according to Student's T-test at level 0.01 (0.05) compared to MAML

Table 3: Comparing with baselines for Bike Volume Prediction.

Bike Data		Chicago					
		Pick-up			Drop-off		
		1-day	3-day	7-day	1-day	3-day	7-day
HA		4.97	3.69	3.64	4.96	3.67	3.63
ARIMA		4.86	4.89	4.79	4.83	4.97	4.86
ST-net		7.61	5.57	3.83	8.03	5.45	3.51
SFT[1]	NYC	2.52	2.49	1.95	2.49	2.41	1.87
	DC	1.88	1.99	1.69	2.03	2.20	1.67
Multi-FT		1.97	2.05	1.62	1.90	1.98	1.59
RT[1]	NYC	2.50	2.23	1.87	2.36	2.18	1.76
	DC	1.86	1.95	1.66	1.98	2.09	1.63
MAML		1.85	1.87	1.62	1.85	1.79	1.56
MetaST		**1.76****	**1.73****	**1.45****	**1.83****	**1.71****	**1.46****

[1]Due to the space limitation, in this table, SFT, RT mean single-FT and RegionTrans respectively.

** (*) means the result is significant according to Student's T-test at level 0.01 (0.05) compared to MAML.

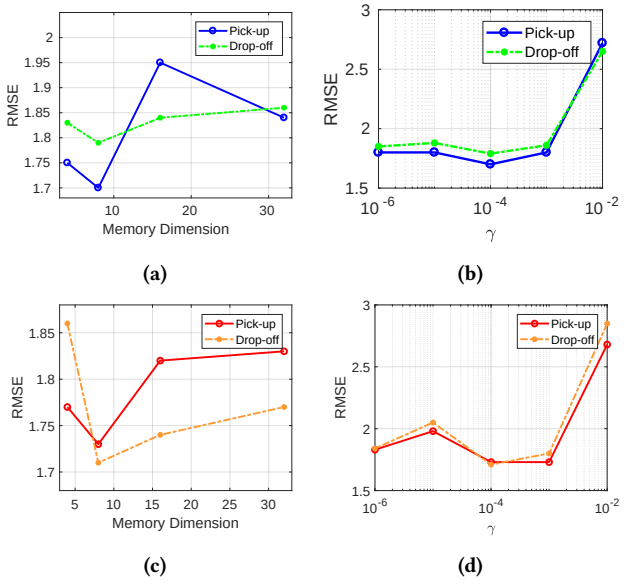


Figure 5: (a) (c) RMSE with respect to the dimension of spatial-temporal memory in Chicago taxi/bike volume prediction, respectively; (b) (d) RMSE with respect to the value of γ in Chicago taxi/bike volume prediction, respectively.

5.1.6 Parameter Sensitivity. MetaST involves a number of hyper-parameters. In this subsection, we evaluate how different selections of hyper-parameters impact our model's performance. Specifically, we analyze the impacts of two key parameters of spatial-temporal memory, i.e., the dimension d of pattern representation and the trade-off factor γ of two losses in the joint objective. All other hyperparameters are set as introduced in Section 5.1.4. We use the scenario of 3-day data for sensitivity analysis.

For the the dimension d of pattern representation, we change the dimension of pattern representation from 4 to 32 in spatial-temporal memory. The performance of Chicago taxi and bike volume prediction are shown in Figure 5a and Figure 5c, respectively. We find that the performance increases in the beginning but decreases later. One potential reason is that the spatial-temporal memory provides too little information when the dimension is too small, while it can

include too much irrelevant information when the dimension is too large. Both of the scenarios hurt the performance. Another experiment of trade-off factor γ also demonstrates similar assumption. We change the parameter γ in Eq. (14) from 10^{-6} to 10^{-2}. Higher value of γ means higher importance of spatial-temporal memory. The results of Chicago taxi and bike volume prediction are shown in Figure 5b and Figure 5d, respectively. We can see the similar change of the performance, increasing at first but decreasing later.

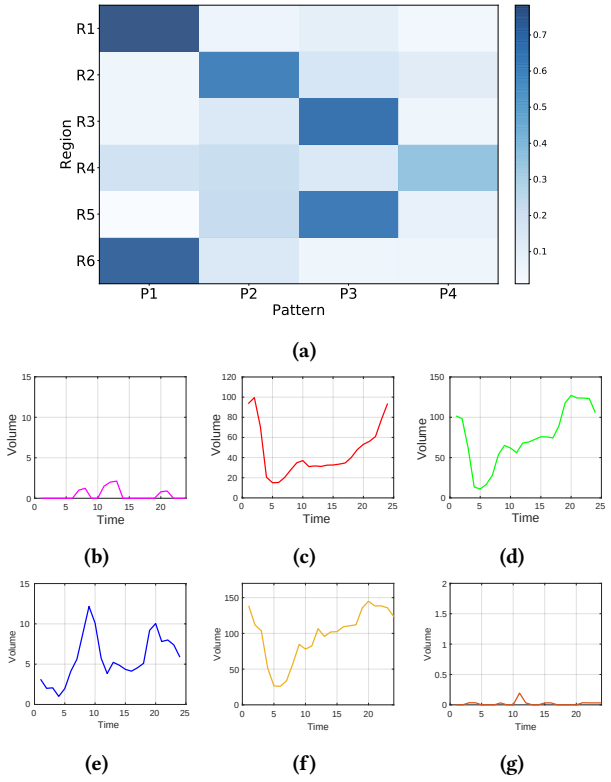


Figure 6: Spatial-temporal patterns detected by the memory $\mathcal{M}$ in Boston Taxi data. (a): probability of 4 pattern values of 6 regions; (b),(c),(d),(e),(f),(g) denote actual patterns of R1, R2, R3, R4, R5, R6, respectively.

5.1.7 Case Study: Visualization of Detected Patterns. To intuitively demonstrate the efficacy of the usage of the Spatial-temporal memory, we visualize patterns detected from Boston taxi pick-up volume prediction. We also use 3-day data for this case study. We randomly select six regions and show the similarity values with respect to each pattern category in the memory $\mathcal{M}$. The results are shown in Figure 6a. The corresponding actual patterns of each region are shown in Figure 6b, Figure 6c, Figure 6d, Figure 6e, Figure 6f, and Figure 6g respectively. We can see that the taxi pick-up volume of R1 and R6 is almost zero and their attention weights are also similar (Pattern 1 is activated). The volume distribution in R2, R3 and R5 have one peak. The peak time of R2 is around 1:00am - 2:00am (Pattern 2 is activated). The pattern and attention weights of R3 and R5 are similar and the peak time is around 8:00pm - 9:00pm (Pattern 3 is activated). In R4, the volume distribution has two peaks (Pattern 4 is activated). One peak is around 9:00am - 10:00am, another one is 8:00pm - 9:00pm. The results indicate that the memory can distinguish regions with different patterns.

5.2 Application-*II*: Water Quality Prediction

5.2.1 Problem Overview. The second application studied in this work is water quality prediction task. We also conduct a water quality prediction experiment. In this scenario, the water quality is represented by pH value of water, because pH value is easier to measure than other chemical metrics of water quality. We aim at predicting pH value in a specific location of next month (i.e., the time interval of water quality prediction is one month), as the changing of pH indicates the relative changes in the chemical composition in water. RMSE is still used as the evaluation metric in this task.

5.2.2 Dataset. The data used in this experiment is collected from the Water Quality Portal[6] of the National Water Quality Monitoring Council. It spans about 52 years from 1966 to 2017. Each record represents one surface water sample with longitude, latitude, date and pH value in a standard unit. The continental U.S. area is roughly split to six areas: Pacific, West, Midwest, Northeast, Southwest, South. Note that, in the water quality prediction task, the areas are treated as the cities in previous descriptions.

In addition, due to the sparsity of sampling points, we split each area into a grid region map, the size of each grid being 0.5°of latitude by 0.5°of longitude. Thus, the map sizes of all the six areas are 25×50, 30×25, 35×25, 30×25, 50×25, 45×25, respectively. The pH value of each region is represented by the median value of all sampling points in this region. We select Pacific, West, Midwest as source areas, and Northeast, Southwest, South as target areas.

5.2.3 Baselines. We use the same baselines as in the experiments for traffic prediction. Both non-transfer methods and transfer methods are used to evaluated our algorithm. Note that, when calculating HA, the relative time is monthly. For example, if we want to predict the water quality at May, HA is the average value of all pH values at May in training data.

5.2.4 Experimental Settings.

Hyperparameter Setting. Similar as the traffic prediction application, we do have external features in the water quality prediction task. For the learning process of spatial-temporal framework, we set the maximum of iterations as 5000, the number of filters in CNN as 32, the dimension of hidden states of LSTM as 64 and the size of memory representation as 4. Other hyperparameters are the same as traffic prediction.

Spatial-temporal Clustering. By analyzing the data, pH in the current month is strongly correlated with pH in the same month of previous year. Thus, we use the 12-month periodic pattern of each region. Similar as traffic prediction task, K-means is also used to cluster these patterns to 3 groups. DTW distance is used to measure the distance of K-means.

[6] https://www.waterqualitydata.us/

Table 4: Comparing with baselines for water quality prediction

pH Data		Northeast			Southwest			South		
		1-year	3-year	7-year	1-year	3-year	7-year	1-year	3-year	7-year
HA		2.302	2.112	2.016	3.051	2.811	2.770	2.402	2.141	2.028
ARIMA		2.309	2.328	2.212	3.153	3.178	3.103	2.372	2.363	2.209
ST-net		4.536	3.806	1.850	2.694	2.094	1.008	4.237	3.951	1.662
Single-FT	West	1.236	1.128	0.862	0.935	0.716	0.683	1.138	1.043	0.837
	Midwest	1.048	1.004	0.806	0.791	0.653	0.622	0.970	0.951	0.793
	Pacific	1.249	1.140	0.854	0.928	0.711	0.671	1.172	1.031	0.835
Multi-FT		1.010	0.987	0.798	0.706	0.587	0.567	0.909	0.898	0.730
RegionTrans	West	1.233	1.115	0.853	0.924	0.698	0.682	1.121	0.993	0.826
	Midwest	1.047	0.988	0.796	0.783	0.651	0.619	0.965	0.938	0.769
	Pacific	1.243	1.098	0.851	0.916	0.693	0.652	1.128	1.012	0.813
MAML		0.997	0.955	0.784	0.701	0.579	0.559	0.907	0.897	0.710
MetaST		**0.903****	**0.898****	**0.758****	**0.649****	**0.541****	**0.514****	**0.820****	**0.803****	**0.650****

** (*) means the result is significant according to Student's T-test at level 0.01 (0.05) compared to MAML

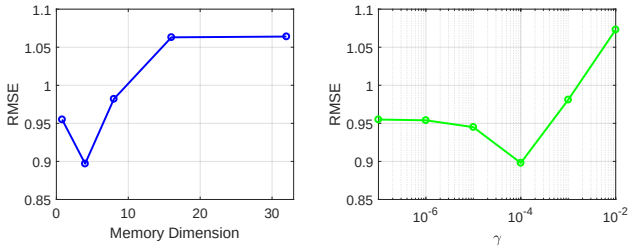
5.2.5 Results. We implement our model and compare with baselines on the water quality prediction task. We run 20 testing tasks and report the average values in Table 4. Most experiment results and their explanation are similar to traffic prediction. Besides, from this table, we observe that:

- Comparing with Multi-FT model, the performance of MAML only slightly improves in most cases. The potential reason is that the regions in the source areas may be homogeneous in short-term performance. Thus, compared to MAML which learning a initialization, simply mixing all samples (i.e., Multi-FT) may not significantly hurt the performance significantly.
- MetaST achieves the best performance compared with all baselines with the averaged relative improvement as 7.7%. Since MetaST provides more detailed long-term information by spatial-temporal memory and distills the long-term information to target city, which explicitly increases the diversity of regions in source domain.

5.2.6 Parameter Sensitivity. Following the same step of traffic prediction task, we investigate the effect of the dimension d of pattern representation and the trade-off factor γ of two losses in the joint objective on MetaST performance. The performance of d from 2 to 32 and γ from 10^{-7} to 10^{-2} on water quality value prediction is shown in Figure 7a and Figure 7b, respectively. Both Figure 7a and Figure 7b are evaluated on Northeast water quality prediction with 3-year training data. Accordingly, MetaST achieves the best performance when $d = 4$ and $\gamma = 10^{-4}$. Similar as the reasons in traffic prediction, the results indicate that suitable selection of d and γ lead to the best performance.

6 CONCLUSION AND DISCUSSION

In this paper, we study the problem of transferring knowledge from multiple cities for spatial-temporal prediction. We propose a novel MetaST model which leverages learned knowledge from multiple cities to help with the prediction in target data-scarce cities. Specifically, the proposed model learns a well-generalized initialization of spatial-temporal prediction model for easier adaptation. Then, MetaST a global pattern-based spatial-temporal memory from all source cities. We test our model on two spatial-temporal prediction problem from two different domains: traffic prediction and environment prediction. The results show the effectiveness and of our proposed model.



(a) (b)

Figure 7: (a) RMSE with respect to the dimension of spatial-temporal memory; (b) RMSE with respect to the value of γ.

For future work, we plan to investigate from two directions: (1) We plan to further consider network structure (e.g., road structure) and combine it with our proposed model. For example, a simple extension is that we can use graph convolutional network as our base model; (2) We plan to explain the black-box transfer learning framework, and analyze which information is transferred (e.g., spatial correlation, region functionality).

ACKNOWLEDGMENTS

We thank Xiuwen Yi for feedback on an early draft of this paper. The work was supported in part by NSF awards #1652525, #1618448, and #1639150. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] Wenyuan Dai, Ou Jin, Gui-Rong Xue, Qiang Yang, and Yong Yu. 2009. Eigentransfer: a unified framework for transfer learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 193–200.

[2] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *International Conference on Machine Learning*. 1126–1135.

[3] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. 2012. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*. IEEE, 2066–2073.

[4] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. 2011. Domain adaptation for object recognition: An unsupervised approach. In *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 999–1006.

[5] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[6] Tsuyoshi Idé and Masashi Sugiyama. 2011. Trajectory regression on road networks. In *AAAI Conference on Artificial Intelligence*. AAAI Press, 203–208.

[7] Jintao Ke, Hongyu Zheng, Hai Yang, and Xiqun Michael Chen. 2017. Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach. *Transportation Research Part C: Emerging Technologies* 85 (2017), 591–608.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[9] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *Sixth International Conference on Learning Representations*.

[10] Marco Lippi, Matteo Bertini, and Paolo Frasconi. 2013. Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. *IEEE TIST* 14, 2 (2013), 871–882.

[11] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, and Yi Yang. 2018. Transductive Propagation Network for Few-shot Learning. *arXiv preprint arXiv:1805.10002* (2018).

[12] Yiding Liu, Tuan-Anh Nguyen Pham, Gao Cong, and Quan Yuan. 2017. An experimental evaluation of point-of-interest recommendation in location-based social networks. *Proceedings of the VLDB Endowment* 10, 10 (2017), 1010–1021.

[13] Yiding Liu, Kaiqi Zhao, and Gao Cong. 2018. Efficient Similar Region Search with Deep Metric Learning. In *SIGKDD*. ACM, 1850–1859.

[14] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[15] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. 2013. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems* 14, 3 (2013), 1393–1402.

[16] Devang K Naik and RJ Mammone. 1992. Meta-neural networks that learn by learning. In *Neural Networks, 1992. IJCNN., International Joint Conference on*, Vol. 1. IEEE, 437–442.

[17] Sinno Jialin Pan, Qiang Yang, et al. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[18] Sachin Ravi and Hugo Larochelle. 2016. Optimization as a Model for Few-Shot Learning. *ICLR* (2016).

[19] Shashank Shekhar and Billy Williams. 2008. Adaptive seasonal time series models for forecasting short-term traffic flow. *Transportation Research Record: Journal of the Transportation Research Board* 2024 (2008), 116–125.

[20] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*. 4077–4087.

[21] Yongxin Tong, Yuqiang Chen, Zimu Zhou, Lei Chen, Jie Wang, Qiang Yang, and Jieping Ye. 2017. The Simpler The Better: A Unified Approach to Predicting Original Taxi Demands on Large-Scale Online Platforms. In *KDD*. ACM.

[22] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *CVPR*.

[23] Bao Wang, Duo Zhang, Duanhao Zhang, P Jeffery Brantingham, and Andrea L Bertozzi. 2017. Deep learning for real time crime forecasting. *arXiv preprint arXiv:1707.03340* (2017).

[24] Dong Wang, Wei Cao, Jian Li, and Jieping Ye. 2017. DeepSD: Supply-Demand Prediction for Online Car-Hailing Services Using Deep Neural Networks. In *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*. IEEE, 243–254.

[25] Hongjian Wang and Zhenhui Li. 2017. Region representation learning via mobility flow. In *CIKM*. ACM, 237–246.

[26] Hongjian Wang, Huaxiu Yao, Daniel Kifer, Corina Graif, and Zhenhui Li. 2017. Non-Stationary Model for Crime Rate Inference Using Modern Urban Data. *IEEE Transactions on Big Data* (2017).

[27] Leye Wang, Xu Geng, Xiaojuan Ma, Feng Liu, and Qiang Yang. 2018. Crowd Flow Prediction by Deep Spatio-Temporal Transfer Learning. *arXiv preprint arXiv:1802.00386* (2018).

[28] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. 2018. Intellilight: A reinforcement learning approach for intelligent traffic light control. In *SIGKDD*. ACM, 2496–2505.

[29] Ying Wei, Yu Zheng, and Qiang Yang. 2016. Transfer knowledge between cities. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1905–1914.

[30] Fei Wu, Hongjian Wang, and Zhenhui Li. 2016. Interpreting traffic dynamics using ubiquitous urban data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 69.

[31] Jianpeng Xu, Pang-Ning Tan, Lifeng Luo, and Jiayu Zhou. 2016. Gspartan: a geospatio-temporal multi-task learning framework for multi-location prediction. In *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 657–665.

[32] Jun Yang, Rong Yan, and Alexander G Hauptmann. 2007. Adapting SVM classifiers to data with shifted distributions. In *Data Mining Workshops, 2007. ICDM Workshops 2007. Seventh IEEE International Conference on*. IEEE.

[33] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, Yanwei Yu, and Zhenhui Li. 2019. Revisiting Spatial-Temporal Similarity: A Deep Learning Framework for Traffic Prediction. *AAAI Conference on Artificial Intelligence* (2019).

[34] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. 2018. Deep Multi-View Spatial-Temporal Network for Taxi Demand Prediction. *AAAI Conference on Artificial Intelligence* (2018).

[35] Xiuwen Yi, Junbo Zhang, Zhaoyuan Wang, Tianrui Li, and Yu Zheng. 2018. Deep Distributed Fusion Network for Air Quality Prediction. In *KDD*.

[36] Wei Ying, Yu Zhang, Junzhou Huang, and Qiang Yang. 2018. Transfer learning via learning to transfer. In *International Conference on Machine Learning*. 5072–5081.

[37] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks?. In *Advances in neural information processing systems*. 3320–3328.

[38] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2017. Spatio-temporal Graph Convolutional Neural Network: A Deep Learning Framework for Traffic Forecasting. *arXiv preprint arXiv:1709.04875* (2017).

[39] Rose Yu, Yaguang Li, Ugur Demiryurek, Cyrus Shahabi, and Yan Liu. 2017. Deep Learning: A Generic Approach for Extreme Condition Traffic Forecasting. In *Proceedings of SIAM International Conference on Data Mining*.

[40] Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. 2018. GaAN: Gated Attention Networks for Learning on Large and Spatiotemporal Graphs. *arXiv preprint arXiv:1803.07294* (2018).

[41] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction. *AAAI* (2017).

[42] Xiangyu Zhao and Jiliang Tang. 2017. Modeling temporal-spatial correlations for crime prediction. In *CIKM*. ACM, 497–506.

[43] Xiangyu Zhao, Tong Xu, Yanjie Fu, Enhong Chen, and Hao Guo. 2017. Incorporating Spatio-Temporal Smoothness for Air Quality Inference. In *ICDM*. IEEE, 1177–1182.

[44] Jiangchuan Zheng and Lionel M Ni. 2013. Time-dependent trajectory regression on road networks via multi-task learning. In *AAAI Conference on Artificial Intelligence*. AAAI Press, 1048–1055.

[45] Ali Zonoozi, Jung-jae Kim, Xiao-Li Li, and Gao Cong. 2018. Periodic-CRN: A Convolutional Recurrent Model for Crowd Density Prediction with Recurring Periodic Patterns.. In *IJCAI*. 3732–3738.

STFNets: Learning Sensing Signals from the Time-Frequency Perspective with Short-Time Fourier Neural Networks

Shuochao Yao[1], Ailing Piao[2], Wenjun Jiang[3], Yiran Zhao[1], Huajie Shao[1], Shengzhong Liu[1], Dongxin Liu[1], Jinyang Li[1], Tianshi Wang[1], Shaohan Hu[4], Lu Su[3], Jiawei Han[1], Tarek Abdelzaher[1]
[1]University of Illinois at Urbana-Champaign [2]University of Washington [3]State University of New York at Buffalo [4]IBM
[1]{syao9, zhao97, hshao5, sl29, dongxin3, jinyang7, tianshi3, hanj, zaher}@illinois.edu
[2]alpiao@uw.edu [3]{wenjunji, lusu}@buffalo.edu [4]shaohan.hu@ibm.com

ABSTRACT

Recent advances in deep learning motivate the use of deep neural networks in Internet-of-Things (IoT) applications. These networks are modelled after signal processing in the human brain, thereby leading to significant advantages at perceptual tasks such as vision and speech recognition. IoT applications, however, often measure *physical phenomena*, where the underlying physics (such as inertia, wireless signal propagation, or the natural frequency of oscillation) are fundamentally a function of signal frequencies, offering better features in the *frequency domain*. This observation leads to a fundamental question: For IoT applications, can one develop a new brand of neural network structures that synthesize features inspired not only by the biology of human perception but also by the fundamental nature of physics? Hence, in this paper, instead of using conventional building blocks (*e.g.*, convolutional and recurrent layers), we propose a new foundational neural network building block, the Short-Time Fourier Neural Network (STFNet). It integrates a widely-used time-frequency analysis method, the Short-Time Fourier Transform, into data processing to learn features directly in the frequency domain, where the physics of underlying phenomena leave better footprints. STFNets bring additional flexibility to time-frequency analysis by offering novel nonlinear learnable operations that are spectral-compatible. Moreover, STFNets show that transforming signals to a domain that is more connected to the underlying physics greatly simplifies the learning process. We demonstrate the effectiveness of STFNets with extensive experiments on a wide range of sensing inputs, including motion sensors, WiFi, ultrasound, and visible light. STFNets significantly outperform the state-of-the-art deep learning models in all experiments. A STFNet, therefore, demonstrates superior capability as the fundamental building block of deep neural networks for IoT applications for various sensor inputs [1].

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Computer systems organization** → **Embedded and cyber-physical systems**; • **Human-centered computing** → **Ubiquitous and mobile computing**.

[1]Code is available on https://github.com/yscacaca/STFNets

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313426

KEYWORDS

Deep learning; time frequency analysis; Internet of Things; IoT

ACM Reference Format:
Shuochao Yao, Ailing Piao, Wenjun Jiang, Yiran Zhao, Huajie Shao, Shengzhong Liu, Dongxin Liu, Jinyang Li, Tianshi Wang, Shaohan Hu, Lu Su, Jiawei Han, Tarek Abdelzaher. 2019. STFNets: Learning Sensing Signals from the Time-Frequency Perspective with Short-Time Fourier Neural Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313426

1 INTRODUCTION

Motivated by the needs of IoT applications, this paper presents a principled way of designing deep neural networks that learn (from IoT sensing signals) features inspired by the fundamental properties of the underlying domain of measurements; namely, properties of physical signals. We refer by IoT applications to those where sensors measure some physical quantities, generating (possibly complex and multi-dimentional) time-series data, typically reflecting some underlying physical process. The human brain (whose wiring inspires the structure of conventional neural networks) extracts features well-suited for *external perceptual* tasks, which explains the great success of such networks at those tasks. In contrast, the internal *physical processes* underlying sensor measurements in IoT systems have properties (such as physical intertia, characteristics of wireless signal propagation, and signal resonance) that depend more on signal *frequency*, motivating feature extraction in the *frequency domain*. It is no coincidence that much of classical signal processing literature works by transforming time-series data to the frequency domain first. To help capture signatures of internal physical processes the way a brain captures their externally perceived properties, this paper develops a new neural network block designed specifically for learning in the frequency domain.

The design of neural network structures greatly influences efficiency of signal modelling and ease of extraction of hidden patterns. Convolutional neural networks (CNNs) for image recognition, for example, align perfectly with biological studies of the visual cortex [13] and with domain knowledge in digital image processing [7]. We thus ask a fundamental question: what structures are well-suited for the domain of physical sensor measurements, which we henceforth call the domain of IoT?

Previous research on customizing deep learning models to the needs of IoT applications [16, 26, 31] mainly focused on designing neural network structures that integrate conventional deep learning components, such as convolutional and recurrent layers, to extract spatial and temporal properties of inputs. On the other hand, since the

physics of measured phenomena are best expressed in the frequency domain, decades of research on signal processing developed powerful techniques for time-frequency analysis of signals, including motion sensor signals [12, 23], radio frequency signals [19, 25], acoustic signals [3, 8], and visible light signals [17]. A popular transform that maps time-series measurements to the frequency domain is the Short-Time Fourier Transform (STFT). We, therefore, propose a new neural network model, namely, Short-Time Fourier Neural Networks (STFNets) that operate directly in the frequency domain.

One potential approach for learning in the frequency domain might simply be to convert sensing signals into the frequency domain first, and then apply conventional neural network components, possibly extending them to support operations on complex-numbers so they can represent frequency-domain quantities [24]. These approaches miss two key opportunities for improvement, described below, that we take advantage of in this work. As a result, our work leads to more accurate results, as shown in the evaluation section. The two reasons that account for our improvements are as follows.

First, different from traditional neural networks, where the internal representations constitute features with no physical meaning, the internal representations in STFNet leverage frequency domain semantics that encode time and frequency information. All operations and learnable parameters we propose are explicitly made compatible with the basic properties of spectral data, and align corresponding frequency and time components. In our design, we categorize spectral manipulations into three main types: filtering, convolution, and pooling. Filtering refers to the general spectral filtering and global template matching operation; convolution refers to the local motif detection including shift detection and local template detection; and pooling refers to dimension reduction over the frequency domain. We then design the spectral-compatible parameters and operating rules for these three manipulation categories respectively, which have shown superior performance in evaluations compared to the application of *conventional* neural networks in the domain of complex-numbers.

Second, transforming signals to the frequency domain is governed by the *uncertainty principle* [22]. The transformed representation cannot achieve both a high frequency resolution and a high time resolution at the same time. In STFT, the time-frequency resolution is controlled by the length of the sliding window (the length of the part of the time-series being converted at a time). With a longer window, we can obtain a finer-grained frequency representation. However, we then cannot achieve a time resolution smaller than the window size. The uncertainty principle causes a dilemma in traditional time-frequency analysis. One often needs to guess the best time-frequency resolution using trial and error. In STFNet, we circumvent this dilemma by simultaneously computing multiple STFTs with different time-frequency resolutions. The representations with different time-frequency resolutions are then mutually enhanced in a data-driven manner. The network then automatically learns the best resolution or resolutions, where the most useful features are present. STFNet defines a formal way to extract features from multiple time-frequency transformations with the same set of spectral-compatible operations and parameters, which greatly reduces model complexity while improving accuracy.

We demonstrate the effectiveness of STFNet through extensive experiments with various sensing modalities. During the evaluation, we focus on device-based and device-free human activity recognition with a broad range of sensing modalities, including motion sensors (accelerometer and gyroscopes), WiFi, ultrasound, and visible light. The experimental results validate the design settings of STFNets and illustrate their superior accuracy compared to the state-of-the-art deep learning frameworks for IoT applications.

Broadly speaking, the main contributions of this paper to the general research landscape of deep learning and IoT are twofold:

(1) STFNet presents a principled way of designing neural networks that reveal the key properties of physical processes underlying the sensing signals from the time-frequency perspective.
(2) STFNet unveils the benefit of incorporating domain-specific analytic modelling and transformation techniques into the neural network design.

The rest of paper is organized as follows. Section 2 introduces related work on deep learning in the context of mobile sensing as well as deep learning for spectral representations. We introduce the detailed technical design of STFNet in Section 3. The evaluation is presented in Section 4. Finally, we discuss the results in Section 5 and conclude in Section 6.

2 RELATED WORK

The impressive achievements in image classification using deep neural networks at the turn of the decade [15] precipitated a re-emergence of interest in deep learning. Deep neural networks have achieved significant accuracy improvements in a broad spectrum of areas, including computer vision [11, 21], natural language processing [1, 4], and network analysis [14, 18].

Recent efforts applied deep learning in the context of IoT. In order to improve the predictive accuracy of IoT applications, researchers employed deep learning to model complicated sensing tasks [16, 26]. In order to improve system efficiency at executing neural networks on low-end IoT devices, efforts have been made to compress model parameters and/or structures in a manner that does not entail (almost any) accuracy loss [2, 10, 27, 32]. Recent work in the context of IoT also addressed mathematical foundations for quantifying confidence in deep learning predictions to support mission-critical applications. The work produced deep neural networks that offer well-calibrated uncertainty estimates in results [5, 6, 28, 30]. Finally, the challenge of insufficient labeling of IoT data was addressed by introducing semi-supervised approaches for deep learning that allow neural network training using mostly *unlabeled* data [29]. However, none of the aforementioned IoT-inspired efforts addressed the customization of learning machinery to a different signal space inspired by the physics of measured processes; namely, the frequency domain.

To fill the above gap, recent work in machine learning focused on extending deep neural networks to complex numbers and spectral representations. Trabelsi et al. propose deep complex networks, investigating the complex-value neural network structure [24]. However, they mainly concentrate on the problems of initialization, normalization, and activation functions when extending real-valued operations directly into the complex-value domain. Their designs focus more on complex-value representations than spectral representations, and do not take the properties of spectral data into consideration. Rippel et al. study spectral representations for convolutional neural networks [20]. However, their study focuses on spectral parametrizing

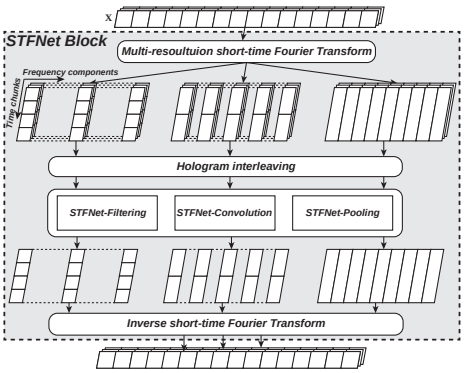


Figure 1: The overview design of STFNet block.

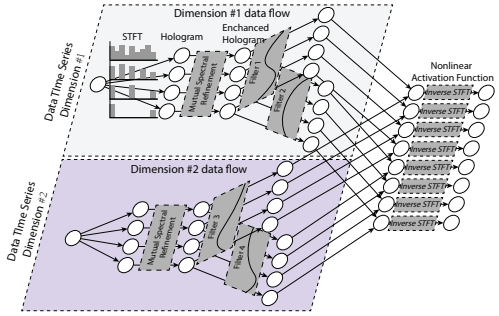


Figure 2: Data Flow within a block of STFNet.

of standard CNNs, instead of designing operations customized for spectral data. In addition, their work treats input data fully from the frequency perspective instead of the time-frequency perspective. Yao et al. propose a neural network that takes short-time Fourier transformed data as inputs [26]. Yet their design uses traditional CNNs and RNNs, combining the real and imagery parts of complex-value inputs as additional features.

To the best of our knowledge, STFNet is the first work that integrates neural networks with traditional time-frequency analysis, and designs fundamental spectral-compatible operations for Fourier-transformed representations. Our study shows that the approach leads to improved accuracy compared to the state of the art. It implies that integrating neural networks with domain-inspired transformation techniques (in our case, the Fourier Transform of physical time-series signals) projects input signals into a space that significantly facilitates the learning process.

3 SHORT-TIME FOURIER NEURAL NETWORKS

We introduce the technical details of STFNets in this section. We separate the technical descriptions into six parts. In the first two subsections, we provide some background followed by a high-level overview of STFNet components, including (i) hologram interleaving, (ii) STFNet-filtering, (iii) STFNet-convolution, and (iv) STFNet-pooling. In the remaining four subsections, we describe the technical details of each of these components, respectively.

3.1 Background and STFNet Overview

IoT devices sample the physical environment generating time-series data. Discrete Fourier Transform (DFT) is a mathematical tool that converts n samples over time (with a sampling rate of f_s) into a n components in frequency (with a frequency step of f_s/n). The more samples are selected, the finer the component resolution is in frequency. We can always transform the whole sequence of data with DFT, achieving a high frequency resolution. However, we then lose information on signal evolution over time, or the time resolution. In order to solve this problem, Short-Time Fourier Transform (STFT) divides a longer time signal into shorter segments of equal length and computes DTF separately on each shorter segment. By losing a certain degree of frequency resolution, STFT helps us regain the time resolution to some extent. In choosing n, there arises a fundamental trade-off between the attainable time and frequency resolution, which is called the *uncertainty principle* [22]. For the purposes of learning to pedict a given output, the optimal trade-off point depends on the time and frequency granularity of the features that best determine the outputs we want to reproduce. The goal of STFNets is thus to learn frequency domain features that predict the output, while at the same time learn the best resolution trade-off point in which the relevant features exist.

The building component of an STFNet is an *STFNet block*, shown in Figure 1. An STFNet block is the layer-equivalent in our neural network. The larger network would normally be composed by stacking such layers. Within each block, STFNet circumvents the uncertainty principle by computing multiple STFT representations with different time-frequency resolutions. Collectively, these representations constitute what we call the *time-frequency hologram*. And we call an individual time-frequency signal representation, a hologram representation. They are then used to mutually enhance each other by filling-in missing frequency components in each.

Candidate frequency-domain features are then extracted from these enhanced representations via general spectral manipulations that come in two flavors; filtering and convolution. They represent global and local feature extraction operations, respectively. The filtering and convolution kernels are learnable, making each STFNet layer a building block for spectral manipulation and learnable frequency domain feature extraction. In addition, we also design a new mechanism, called pooling, for frequency domain dimensionality reduction in STFNets. Combinations of features extracted using the above manipulations then pass through activation functions and an inverse STFT transform to produce (filtered) outputs in the time domain. Stacking STFNet blocks has the effect of producing progressively sharper (i.e., higher order) filters to shape the frequency domain signal representation into more relevant and more fine-tuned features.

Figure 2 gives an example of an SFTNet block that accepts as input a two-dimensional time-series signal (e.g., 2D accelerometers data). Each dimension is then transformed to the frequency domain at four different resolutions using STFT, generating four different internal nodes, each representing the signal in the frequency domain at a different time-frequency resolution. Collectively, the four representations constitute the hologram. In the next step, mutual enhancements are done improving all representations. Each representation then undergoes a variety of alternative spectral manipulations (called “filters" in the figure). Two filters are shown in the figure for each dimension. The parameters of these filters are the weights multiplied by the frequency components of the filter input; a different weight per component. These parameters are what the network learns. Note that, a filter does not change the time-frequency

resolution of the corresponding input. Filter outputs of the same time-frequency resolution are then combined additively across all dimensions and passed through a non-linear activation function (as in a conventional convolutional neural network). An inverse STFT brings each such combined output back to the time domain, where it becomes an input to the next STFNet block. (Alternatively, the inverse STFT can be applied after dimension combination and before the activation function.) Hence, each output time-series is produced by applying spectral manipulation and fusion to one particular time-frequency resolution of all input time-series. Once converted to the time domain, however, the output time-series can be resampled in the next block at different time-frequency resolutions again. The goal of STFNet is to learn the weighting of different frequency components within each filter in each block such that features are produced that best predict final network outputs.

3.2 STFNet Block Fundamentals

In this subsection, we introduce the formulation of our design elements wihin each STFNet block. In the rest of this paper, all vectors are denoted by bold lower-case letters (e.g., $\mathbf{x}$ and $\mathbf{y}$), while matrices and tensors are represented by bold upper-case letters (e.g., $\mathbf{X}$ and $\mathbf{Y}$). For a vector $\mathbf{x}$, the j^{th} element is denoted by $\mathbf{x}_{[j]}$. For a tensor $\mathbf{X}$, the t^{th} matrix along the third axis is denoted by $\mathbf{X}_{[\cdot,\cdot,t]}$, and other slicing denotations are defined similarly. We use calligraphic letters to denote sets (e.g., $\mathcal{X}$ and $\mathcal{Y}$). For set $\mathcal{X}$, $|\mathcal{X}|$ denotes the cardinality.

We denote the input to the STFNet block as $\mathbf{X} \in \mathbb{R}^{T \times D}$, where we divide the input D-dimension time-series into windows of size T samples. We call T the signal length and D the signal dimension. Since we concentrate on sensing signals, we assume that all the raw and internal-manipulated sensing signals are real-valued in time domain.

As shown in Figure 1, the input signal $\mathbf{X}$ first goes through a multi-resolution short-time Fourier transform (Multi_STFT), which is a compound traditional short-time Fourier transform (STFT), to provide a time-frequency hologram of the signal. STFT breaks the original signal up into chunks with a sliding window, where sliding window $\mathbf{W}(t)$ with width τ only has non-zero values for $1 \le t \le \tau$. Then each chunk is Discrete-Fourier transformed,

$$\mathbf{STFT}^{(\tau,s)}(\mathbf{X})_{[m,k,d]} = \sum_{t=1}^{T} \mathbf{X}_{[t,d]} \cdot \mathbf{W}(t - s \cdot m) \cdot e^{-j\frac{2\pi k}{\tau}(t - s \cdot m)}, \quad (1)$$

where $\mathbf{STFT}^{(\tau,s)}(\mathbf{X}) \in \mathbb{C}^{M \times K \times D}$ denotes the short-time Fourier transform with width τ and sliding step s. M denotes the number of time chunks. K denotes the number of frequency components. Since input signal $\mathbf{X}$ is real-valued, its discrete Fourier transform is conjugate symmetric. Therefore, we only need the $\lfloor \tau/2 \rfloor + 1$ frequency components to represent the signal, *i.e.*, $K = \lfloor \tau/2 \rfloor + 1$. In this paper, we focus on sliding chunks with rectangular window and no overlaps to simplify the formulation, *i.e.*, $s = \tau$ and $M = T/\tau$. We therefore denote of short-time Fourier transform as $\mathbf{STFT}^{(\tau)}(\mathbf{X})$.

The Multi_STFT operation is composed of multiple short-time Fourier transform with different window widths $\mathcal{T} = \{\tau_i\}$. The window width, τ_i, determines the time-frequency resolution of STFT. Larger τ_i provides better frequency resolution, while smaller τ_i provides better time resolution. In this paper, we set the window widths to be powers of 2, *i.e.*, $\tau_i = 2^{p_i}\ \forall p_i \in \mathbb{Z}_0^+$, to simplify the design later.

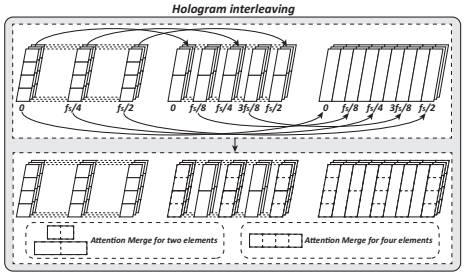


Figure 3: The design of hologram interleaving.

We can thus formulate Multi_STFT as:

$$\mathbf{Multi_STFT}^{(\mathcal{T})}\{\mathbf{X}\} = \left\{\mathbf{STFT}^{(\tau_i)}(\mathbf{X})\right\} \text{ for } 2^{p_i} \in \mathcal{T}. \quad (2)$$

Next, according to Figure 1, the multi-resolution representations go into the hologram interleaving component, which enables the representations to compensate and balance their time-frequency resolutions with each other. The technical details of the hologram interleaving component are introduced in Section 3.3.

The STFNet block then manipulates multiple hologram representations with the same set of spectral-compatible operation(s), including STFNet-filtering, STFNet-convolution, and STFNet-pooling. We will formulate these operations in Section 3.4, 3.5, and 3.6, respectively.

Finally, the STFNet block converts the manipulated frequency representations back into the time domain with the inverse short-time Fourier transform. The resulting representations from different views of the hologram are weighted and merged as the input "signal" for the next block. Since we merge the output representations from different views of the hologram, we reduce the output feature dimension of STFNet-filtering and convolution operations by the factor of $1/|\mathcal{T}|$ to prevent the dimension explosion.

3.3 STFNet Hologram Interleaving

In this subsection, we introduce the formulation of hologram interleaving. Due to the Fourier uncertainty principle, the representations in time-frequency hologram either have high time resolution or high frequency resolution. The hologram interleaving tries to use representations with high time resolution to instruct the representations with low time resolution to highlight the important components over time. This is done by two steps:

(1) Revealing the mathematical relationship of aligned time-frequency components among different representations in the time-frequency hologram.
(2) Updating the original relationship in a data-driven manner through neural-network attention components.

We start from the definition of time-frequency hologram, generated by Multi_STFT defined in (2). Note that, the window width set $\mathcal{T}$ is defined as $\{2^{p_i}\}$, $\forall p_i \in \mathbf{Z}_0^+$. Without loss of generality, an illustration of multi-resolution short-time Fourier transformed representations with input signal having length 16 and signal dimension 3 as well as $\mathcal{T} = \{4, 8, 16\}$ are illustrated in Figure 3.

In order to find out the relationship of aligned time-frequency components, we start with the frequency-component dimension. Since different representations only change the window width τ_i of STFT but not the sampling frequency f_s of input signal, these frequency components represent frequencies from 0 to $f_s/2$ (Nyquist frequency) with step f_s/τ_i. Then we can first obtain the relationship of frequency ranging steps among different representations,

$$\forall p_i > p_j\,,\ \frac{f_s/\tau_j}{f_s/\tau_i} = 2^{p_i-p_j} \in \mathbb{Z}_0^+. \tag{3}$$

Therefore, a low frequency-resolution representation (with window width 2^{p_j}) can find their frequency-equivalent counterparts for every $2^{p_i-p_j}$ frequency components in a high frequency-resolution representation (with window width 2^{p_i}). The upper part of Figure 3 provides a simple illustration of such relationship. In the following analysis, we will use the original index k and corresponding frequency $k \cdot f_s/\tau_i$ interchangeably to recall the frequency component from the time-frequency hologram $\mathbf{STFT}^{(\tau)}(\mathbf{X})_{[m,k,d]}$.

Next, we analyze the relationship over the time-chunk dimension, when two representations have frequency-equivalent components. Note that time chunks in $\mathbf{STFT}^{(\tau)}(\mathbf{X})$ are generated by sliding rectangular window without overlap. Based on (1), for representations having window widths $\tau_i = 2^{p_i}$ and $\tau_j = 2^{p_j}$ $(p_i > p_j)$,

$$\begin{aligned}
\mathbf{STFT}^{(\tau_i)}(\mathbf{X})_{[m,2^{p_i-p_j}k,d]} &= \sum_{t=2^{p_i}m+1}^{2^{p_i}(m+1)} \mathbf{X}_{[t,d]} \cdot e^{-j\frac{2\pi 2^{p_i-p_j}k}{2^{p_i}}(t-m\cdot 2^{p_i})}, \\
&= \sum_{m_j=2^{p_i-p_j}m}^{2^{p_i-p_j}(m+1)-1} \sum_{t=m_j+1}^{2^{p_j}(m_j+1)} \mathbf{X}_{[t,d]} \cdot e^{-j\frac{2\pi k}{2^{p_j}}(t-m\cdot 2^{p_j})}, \\
&= \sum_{m_j=2^{p_i-p_j}m}^{2^{p_i-p_j}(m+1)-1} \mathbf{STFT}^{(\tau_j)}(\mathbf{X})_{[m_j,k,d]}.
\end{aligned} \tag{4}$$

Therefore, given the equivalent frequency component, a time component in low time-resolution representation (with window width 2^{p_i}) is the sum of $2^{p_i-p_j}$ aligned time components of the high time-resolution representation (with window width 2^{p_j}). As a toy example in Figure 3, the first row of the middle tensor is equal to the sum of first two rows of the left tensor for frequencies 0, $f_s/4$, and $f_s/2$. The row of the right tensor is equal to the sum of four rows of the left tensor for frequencies 0, $f_s/4$, and $f_s/2$. The row of the right tensor is equal to the sum of two rows of the middle tensor for frequencies $f_s/8$ and $3f_s/8$, etc.

According to the analysis above, the high frequency-resolution representations lose their fine-grained time resolutions at certain frequencies by summing the corresponding frequency components up over a range of time. However, the high time-resolution representations preserve these information.

The idea of hologram interleaving is to replace the sum operation in high frequency-resolution representation with a weighted merge operation to highlight the important information over time. For a certain frequency component, the weight of merging is learnt through the most fine-grained information preserved in the time-frequency hologram. In this paper, we implement the weighted merge operation as a simple attention module. For a merging input $\mathbf{z} \in \mathbb{C}^{S\times 1}$, where S is the number of elements to be merged, the merge operation is formulated as:

$$\begin{aligned}
\mathbf{a} &= \mathrm{softmax}(|\mathbf{W}_m\mathbf{z}|), \\
y &= S \times \mathbf{a}^\top \mathbf{z},
\end{aligned} \tag{5}$$

where $|\cdot|$ is the piece-wise magnitude operation for complex-number vector; and $\mathbf{W}_m \in \mathbb{C}^{S\times S}$ is the learnable weight matrix. Notice that the final merged result is rescaled by the factor S to imitate the "sum" property of Fourier transform.

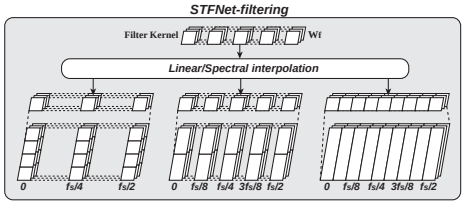


Figure 4: The STFNet-filtering operation.

3.4 STFNet-Filtering Operation

Starting from this subsection, we will introduce our three spectral-compatible operations in STFNet. In each subsection, the introduction includes two main parts: 1) the basic formulation of proposed spectral-compatible operation, and 2) extending a single operation to multi-resolution data.

Spectral filtering is a widely-used operation in time-frequency analysis. The STFNet-filtering operation replaces the traditional manually designed spectral filter with a learnable weight that can update during the training process. Although the spectral filtering is equivalent to the time-domain convolution according to convolution theorem [2], the filtering operation helps to handle the multi-resolution time-frequency analysis, and facilitates the parameterization and modelling. We denote the input tensor as $\mathbf{X} \in \mathbf{C}^{M\times K\times D}$, where M is the number of time chunk, K frequency component number, and D input feature dimension. The STFNet-filtering operation is formulated as:

$$\mathbf{Y}_{[m,k,\cdot]} = \mathbf{X}_{[m,k,\cdot]}\mathbf{W}_{f[k,\cdot,\cdot]}, \tag{6}$$

where $\mathbf{W}_f \in \mathbb{C}^{K\times D\times O}$ is the learnable weight matrix, O the output feature dimension, and $\mathbf{Y} \in \mathbb{C}^{M\times K\times O}$ the output representation.

The function of STFNet-filtering operation is providing a set of learnable global frequency template matchings over the time. However, it is not straightforward to extend the matching operation to the representations with different time-frequency resolutions. Although we can create multiple $\mathbf{W}_f$ with different frequency resolutions K, it can introduce unnecessary complexity and redundancy.

STFNet-filtering solves this problem by interpolating the frequency components in weight matrix. As we mentioned in Section 3.3, data in hologram with different frequency resolutions have the same frequency range (from 0 to $f_s/2$) but different frequency steps (f_s/τ). Therefore, STFNet-filtering operation only has one weight matrix $\mathbf{W}_f$ with $K = \lfloor \tau/2 \rfloor + 1$ frequency components. When the operation input has $K' = \lfloor \tau'/2 \rfloor + 1$ frequency components with $K' < K$, we can subsample the frequency components in $\mathbf{W}_f$. When $K' > K$, we interpolate the frequency components of $\mathbf{W}_f$. STFNet provides two kind of interpolation methods: 1) linear interpolation and 2) spectral interpolation.

The linear interpolation generates the missing frequency components in extended weight matrix $\mathbf{W}'_f \in \mathbb{C}^{K'\times D\times O}$ from the two neighbouring frequency components in $\mathbf{W}_f$:

$$\begin{aligned}
k_l &= \left\lfloor k'\frac{\tau}{\tau'} \right\rfloor \quad k_r = k_l + 1, \\
\mathbf{W}'_{f[k',\cdot,\cdot]} &= \mathbf{W}_{f[k_l,\cdot,\cdot]}\left(k_r - k'\frac{\tau}{\tau'}\right) + \mathbf{W}_{f[k_r,\cdot,\cdot]}\left(k'\frac{\tau}{\tau'} - k_l\right).
\end{aligned} \tag{7}$$

The spectral interpolation utilizes the relationship between discrete-time Fourier transform (DTFT) and discrete Fourier transform (DFT). For a time-limited signal (with length τ), DTFT regards

[2] https://en.wikipedia.org/wiki/Convolution_theorem

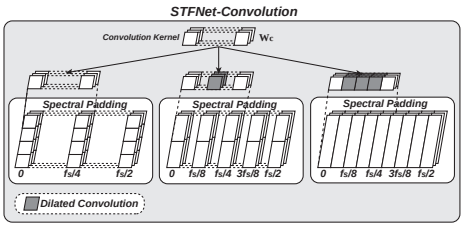


Figure 5: The STFNet-convolution operation with dilated configuration.

it as a infinite-length data with zeros outside the time-limited range, while DFT regards it as a τ-periodic data. As a result, DTFT generates a continuous function over the frequency domain, while DFT generates a discrete function. Therefore, DFT can be regarded as a sampling of DTFT with step f_s/τ. In order to increase the frequency resolution of $\mathbf{W}_f$, we can increase the sampling step from f_s/τ to f_s/τ', which is called spectral interpolation. Spectral interpolation can be done through zero padding in the time domain [22],

$$\mathbf{W}'_{f[\cdot,d,o]} = \mathbf{DFT}\Big(\mathbf{ZeroPad}_{\tau'-\tau}\mathbf{IDFT}\big(\mathbf{W}_{f[\cdot,d,o]}\big)\Big), \tag{8}$$

where $\mathbf{ZeroPad}_t$ denotes padding t zeros at the end of sequence, and $\mathbf{IDFT}(\cdot)$ denotes the inverse discrete Fourier transform. Please note that, if we pad infinite zeros to the IDFT result, then DFT turns into DTFT. An simple illustration of STFNet-filtering operation is shown in Figure 4.

3.5 STFNet-Convolution Operation

In this subsection, we introduce our design of STFNet-convolution operation. Other than filtering operation that handles global pattern matching, we still need the convolution operation to deal with local motifs in the frequency domain. We denote the input tensor as $\mathbf{X} \in \mathbf{C}^{M\times K\times D}$, where M is the number of time chunk, K number of frequency component, and D input feature dimension. The convolution operation involves two steps: 1) padding the input data, and 2) convolving with kernel weight matrix $\mathbf{W}_c \in \mathbb{C}^{1\times S\times D\times O}$, where S is the kernel size along the frequency axis and O is still the output feature dimension.

Without the padding step, the output of convolution operation will shrink the number of frequency components, which may break the underlying structure and information in the frequency domain. Therefore, we need to pad extra "frequency component" to keep the shape of output tensor unchanged compared to that of the input data. In the deep learning research, padding zeros is a common practice. Zero padding is reasonable for inputs such as images and signal in the time domain, meaning no additional information in the padding range. However, padding zero-valued frequency component introduces additional information in the frequency domain.

Therefore, STFNet-convolution operation proposes the spectral padding for time-frequency analysis. According to the definition of DFT, transformed data is periodic within the frequency domain. In addition, if the original signal is real-valued, then the transformed data is conjugate symmetric within each period. Previously, we cut the number of frequency components of a τ-length signal to $K = \lfloor \tau/2 \rfloor + 1$ for reducing the redundancy. In the spectral padding, we add these frequency components back according to the rule

$$\mathbf{X}_{[\cdot,\tau-k,\cdot]} = \mathbf{X}_{[\cdot,-k,\cdot]} = \mathbf{X}^*_{[\cdot,k,\cdot]}, \tag{9}$$

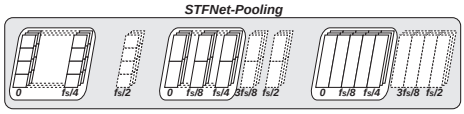


Figure 6: The low-pass STFNet-pooling operation.

where $\mathbf{X}^*$ denotes complex conjugation. In addition, the number of padding before and after the input tensor is same as the previous padding techniques.

Then we can define the basic convolution operation in STFNet

$$\mathbf{Y} = \mathbf{SpectralPad}(\mathbf{X}) \circledast \mathbf{W}_c, \tag{10}$$

where $\mathbf{SpectralPad}(\cdot)$ denotes our spectral padding operation, and $\circledast$ denotes the convolution operation.

Next, we discuss the way to share the kernel weight matrix $\mathbf{W}_c$ with multi-resolution data. Other than interpolating the kernel weight matrix as shown in (7) and (8), we propose another solution for the STFNet-convolution operation. The convolution operation concerns more about the pattern of relative positions on the frequency domain. Therefore, instead of providing additional kernel details on fine-grained frequency resolution, we can just ensure that the convolution kernel is applied with the same frequency spacing on representations with different frequency resolutions. Such idea can be implemented with the dilated convolution [33]. If $\mathbf{W}_c$ is applied to a input tensor with $K = \lfloor \tau/2 \rfloor + 1$ frequency components, for a input tensor with $K' = \lfloor \tau'/2 \rfloor + 1$ frequency components ($\tau' > \tau$), the dilated rate r is set to $\tau'/\tau - 1$. A simple illustration of STFNet-convolution with dilated configuration is shown in Figure 5.

3.6 STFNet-Pooling Operation

In order to provide a dimension reduction method for sensing series within STNet, we introduce the STFNet-pooling operation. STFNet-pooling truncates the spectral information over time with a predefined frequency pattern. As a widely-used processing technique, filtering zeroes unwanted frequency components in the signal. Various filtering techniques have been designed, including low-pass filtering, high-pass filtering, and band-pass filtering, which serve as templates for our STFNet-pooling. Instead of zeroing unwanted frequency components, STFNet-pooling removes unwanted components and then concatenates the left pieces. For applications with domain knowledge about signal-to-noise ratio over the frequency domain, specific pooling strategy can be designed. In this paper, we focus on low-pass STFNet-pooling as an illustrative example.

To extend the STFNet-pooling operation to multiple resolutions and preserving spectral information, we make sure that all representations have the same cut-off frequency according to their own frequency resolutions. A simple example of low-pass STFNet-pooling operation is shown in Figure 6. We can see that our three tensors are truncated according to the same cut-off frequency, $f_s/4$.

4 EVALUATION

In this section, we evaluate the STFNet with diverse sensing modularities. We focus on the device-based and device-free human activity recognitions with motion sensors (accelerometer and gyroscope), WiFi, ultrasound, and visible light. We first introduce the experimental setting, including data collection and baseline algorithms. Next, we show the performance metrics of leave-one-user-out evaluation of human activity recognition with different modularities. Finally, we analyze the effectiveness of STFNet through several ablation studies.

4.1 Experimental Settings

In this subsection, we first introduce detailed information of the dataset we used or collected for each evaluation task. Then we specify the way to test the performance of evaluation task.

Motion Sensor: In this experiment, we recognize human activity with motion sensors on smart devices. We use the dataset collected by Allan et al. [23]. This dataset contains readings from two motion sensors (accelerometer and gyroscope). Readings were recorded when users executed activities scripted in no specific order, while carrying smartwatches and smartphones. The dataset contains 9 volunteers, 6 activities (biking, sitting, standing, walking, climbStair-up, and climbStair-down). We align two sensor readings, linear interpolate two readings by 100Hz, and segment them into non-overlapping data samples with time interval 5.12s. Therefore, each data sample is a 512×6 matrix, where both accelerometer and gyroscope have readings on x, y, and z axis.

WiFi: In this experiment, we make use of Channel State Information (CSI) to analyze human activities. CSI refers to the known channel properties of a communication link, which can be affected by the presence of humans and their activities. We employ 11 volunteers (including both men and women) as the subjects and collect CSI data from 6 different rooms in two different buildings. In particular, we build a WiFi infrastructure, which includes a transmitter (a wireless router) and two receivers. We use the tool to report CSI values of 30 OFDM subcarriers [9]. The experiment contains 6 activities (wiping the whiteboard, walking, moving a suitcase, rotating the chair. sitting, as well as standing up and sitting down). We linearly interpolate the CSI data with a uniform sampling period, and down-sample the measurements into 100Hz. Then we segment the down-sampled CSI data into non-overlapping data samples with time interval 5.12s. Therefore, each data sample is a 512×30 matrix, where each CSI measurement has readings from 30 subcarriers.

Ultrasound: In this experiment, we conduct human activity recognition based on ultrasound. We employ 12 volunteers as the subjects to conduct the 6 different activities. The activity data are collected from 6 different rooms in two different buildings. The transmitter is an iPad on which an ultrasound generator app is installed, and it can emit an ultrasound signal of approximately 19 KHz. The receiver is a smartphone and we use the installed recorder app to collect the sound waves. We demodulate the received signal with carrier frequency 19KHz, and down-sample the measurement into 100Hz. Then we segment the down-sampled ultrasound data into non-overlapping data samples with time interval 5.12s. Therefore, each sample is a 512×1 matrix.

Visible light: In this experiment, we capture the human activity in the visible light system. We build an optical system using photoresistors to capture the in-air body gesture, which can detect the illuminance change (lux) caused by the body interaction. In the experiment, there are three light conditions (natural mode, warm mode, and cool mode) and 4 hand gestures (drawing an anticlockwise circle, drawing a clockwise circle, drawing a cross, and shaking hand side to side). We employ 6 volunteers as the subjects and each of them performs 20 trials of every gesture under a given lighting condition. We linearly interpolate and down-sample the measurements into 25Hz. Then we segment the data into non-overlapping data samples with time interval 5.12s. Therefore, each sample is a 128×6 matrix, where each measurement contains readings from 6 CdS cells.

Table 1: Illustration of models with two sensor inputs.

STFNet-Filter/Conv		DeepSense/ComplexNet	
Sensor Data 1	Sensor Data 2	Chunked Sensor Data 1	Chunked Sensor Data 2
STFNet1-1	STFNet1-2	Conv Layer1-1	Conv Layer1-2
STFNet2-1	STFNet2-2	Conv Layer2-1	Conv Layer2-2
STFNet3-1	STFNet3-2	Conv Layer3-1	Conv Layer3-2
STFNet-pooling		Max pooling	
STFNet4		Conv Layer4	
STFNet5		Conv Layer5	
STFNet6		Conv Layer6	
Averaging		GRU	
Softmax		Softmax	

Testing: In the whole evaluation, to illustrate the generalization ability of STFNet and other baseline models, we perform leave-one-user-out cross validation for every task. For each time, we choose the data from one user as testing data with the left as training data. We then compare the performance of models according to their accuracy and F1 score with 95% confidence interval.

4.2 Models in Comparison

In order to evaluate, when compared to conventional deep learning components (*i.e.*, convolutional and recurrent layers), whether our proposed STFNet component is better at decoding information and extracting features from sensing inputs, we substitute components in the state-of-the-art neural network structure for IoT applications with STFNet. In the whole evaluation, we choose DeepSense as the state-of-the-art structure, which has shown signifiant improvements on various sensing tasks [26]. The illustration of structures of five comparing models with two sensor inputs are shown in Table 1. Detailed information of our comparing models are listed as follows,

(1) ***STFNet-Filter:*** This model integrates the proposed STFNet component and the DeepSense structure. Within the STFNet component, we use the STFNet-filtering operation designed in Section 3.4. The intuition of DeepSense structure is to first perform local processing within each sensor and then perform global sensor fusion over multiple sensors. In this model, we replace all convolutional layers used in local/global sensor data processing with our time-frequency analyzing component, STFNet. Since our model has already incorporated time-domain analysis within the STFNet component through multi-resolution processing, we replace the Gated Recurrent Units (GRU) with simple feature averaging time at last.
(2) ***STFNet-Conv:*** This model is almost the same as the STFNet-Filter, except that we use the STFNet-convolution operation designed in Section 3.5.
(3) ***DeepSense-Freq:*** This model is the original DeepSense [26]. It divides the input sensing data into chunks, and processes each chunk with DFT. It treats the real and imagery parts of discrete Fourier transformed time chunks as the additional feature dimensions. This is the state-of-the-art deep learning model for sensing data modelling and IoT applications.
(4) ***DeepSense-Time:*** This model is almost the same as the DeepSense-Freq, except that it directly takes the chunked raw sensing data without DFT as input.
(5) ***ComplexNet:*** This model is a complex-value neural network [24] that can operate on complex-value inputs. Instead of using simple CNN and RNN structure as originally proposed [24], we cheat in their favor by using the DeepSense

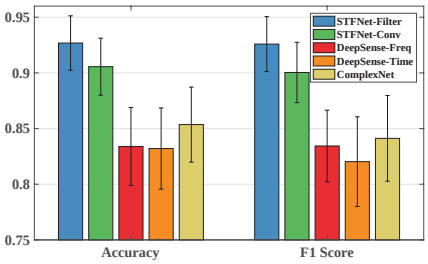


Figure 7: The accuracy and F1 score with 95% confidence interval for motion sensors.

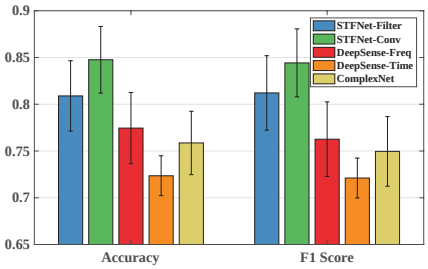


Figure 8: The accuracy and F1 score with 95% confidence interval for WiFi.

structure, which improves the performance in all tasks. The network inputs are chunked sensing data with DFT.

4.3 Effectiveness

In this section, we discuss about the effectiveness of our proposed STFNet based on extensive experiments and diverse sensing modalities, compared with other state-of-the-art deep learning models.

As we mentioned in Section 4.1, all models are evaluated through leave-one-user-out cross validation with accuracy and F1 score accompanied by the 95% confidence interval. STFNet-based models (STFNet-Filter and STFNet-Conv) take a sliding window set for multi-resolution short-time Fourier transform. We choose the set to be $\{16, 32, 64, 128\}$ for activity recognition with motion sensors, WiFi, and ultrasound; and choose set to be $\{8, 16, 32, 64\}$ for activity recognition with visible light. DeepSense-based models (DeepSense-Freq and DeepSense-Time) need a sliding window for chunking input signals. In the evaluation, we cheat in their favor by choosing the best-performing window size from $\{8, 16, 32, 64, 128\}$ according to the accuracy metric. In addition, we consistently configure STFNet-filtering operation with linear interpolation, and STFNet-convolution operation with spectral padding. We will show further evaluations on multi-resolution operations and the effects of diverse operation settings in Section 4.4.

4.3.1 Motion Sensors. For device-based activity recognition with motion sensors, there are 9 users. The accuracy and F1 score with the 95% confidence interval for leave-one-user-out cross validation are illustrated in Figure 7. STFNet based models, *i.e.*, STFNet-Filter and STFNet-Conv, outperform all other baseline models with a large margin. The confidence interval lower bound of STNet-Filter and STFNet-Conv is even better than the confidence interval upper bound of DeepSense-Freq and DeepSense-Time. STFNet-Filter performs better than STFNet-Conv in this experiment, indicating that different activities have distinct global profiling patterns with motion sensor readings in the frequency domain, even among different users.

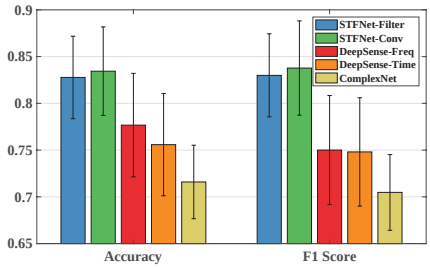


Figure 9: The accuracy and F1 score with 95% confidence interval for Ultrasound.

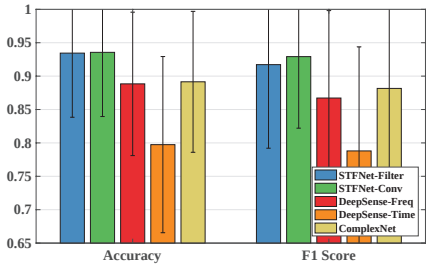


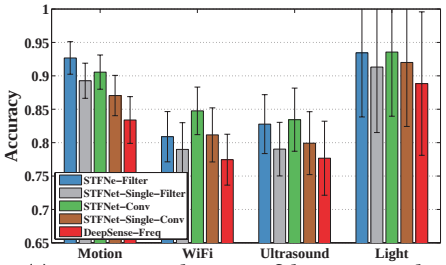
Figure 10: The accuracy and F1 score with 95% confidence interval for Visible light.

STFNet-Filter is able to learn the accurate global frequency profiling, which makes it the top-performance model in this task. In addition, compared to ComplexNet, STFNet based models show clear improvements. Therefore, using just complex-value neural network for sensing signal is far from enough. The multi-resolution processing and operations that are spectral-compatible are all crucial designs.

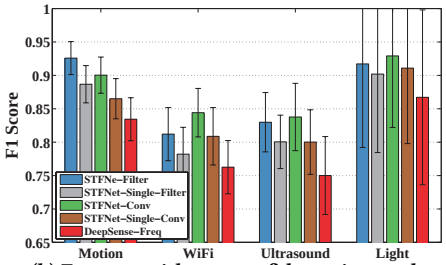
4.3.2 WiFi. For device-free activity recognition with WiFi signal, there are 11 users. The accuracy and F1 score with the 95% confidence interval for leave-one-user-out cross validation are illustrated in Figure 8. STFNet based models still outperform all others with a clear margin, illustrating the effectiveness of principled design of STFNet from time-frequency perspective. DeepSense-Freq outperforms DeepSense-Time in this experiment, which means that even having time-frequency transformation as pre-processing can help. The complex-value network, ComplexNet, performs worse than its real-value counterpart, DeepSense-Freq. This indicates that blindly processing time-frequency representations without preserving their physical meanings can even hurt the final performance. STNet-Conv performs better than STNet-Filter in the WiFi experiment, indicating that local shiftings in the frequency domain are more representative for diverse activities profiled with WiFi CSI.

4.3.3 Ultrasound. There are 12 users in device-free activity recognition with ultrasound experiment. The accuracy and F1 score with the 95% confidence interval for leave-one-user-out cross validation are illustrated in Figure 9. STFNet based models still significantly outperforms all other baselines. An interesting observation is that ComplexNet performs even worse than both DeepSense-Freq and DeepSene-Time, which again validates the importance of designing neural networks for sensing signal with multi-resolution processing as well as preserving the time and frequency information.

4.3.4 Visible Light. There are 6 users in the experiment of device-free activity recognition with visible light. The accuracy and F1 score with the 95% confidence interval are illustrated in Figure 10. Except

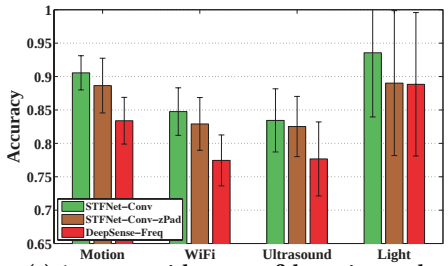


(a) Accuracy with 95% confidence interval.

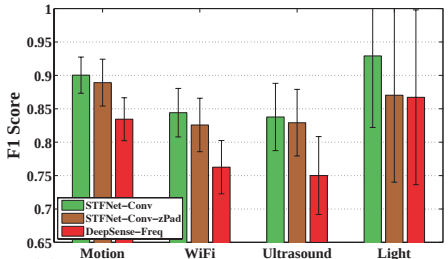


(b) F1 score with 95% confidence interval.

Figure 11: Multi-Resolution v.s. Single-Resolution



(a) Accuracy with 95% confidence interval.



(b) F1 score with 95% confidence interval.

Figure 12: Spectral Padding v.s. Zero Padding

for the DeepSense-Time, all other models can can achieve an accuracy of approximately 90% or higher. STFNet based models still do the best. There is no significant difference between STFNet-Filter and STFNet-Conv, which indicates that measured visible light readings have quite clean representations in the frequency domain.

4.4 Ablation Studies

In the previous section, we illustrate the performance of STFNet compared to other state-of-the-art baselines. In this section, we focus mainly on the STFNet design. We conduct several ablation studies by deleting one designing feature from STFNet at a time.

4.4.1 Multi-Resolution v.s. Single-Resolution. First, we validate the effectiveness of our design of multi-resolution processing in STFNet block. As shown in Figure 1, this includes multi-resolution STFT, hologram interleaving, and weights sharing techniques in STFNet-Filtering and STFNet-Convolution operations. In this experiment, we add two more baseline models, STFNet-Single-Filter and STFNet-Single-Conv, generated by deleting the multi-resolution processing in STFNet-Filter and STFNet-Conv respectively. These two models pick the best-performing window size from $\{8, 16, 32, 64, 128\}$ according to the accuracy metric. The results for all four tasks are illustrated in Figure 11, where DeepSense-Freq severs as a decent performance low-bound. The design of multi-resolution processing significantly impacts the performance of STFNet. STFNet-Single-Filter and STFNet-Single-Conv show clear performance degradation compared to their multi-resolution counterparts. In addition, STFNet-Single-Filter and STFNet-Single-Conv still consistently outperform DeepSense-Freq with a clear margin. This is because our other designed operations, including STFNet-Filtering, STFNet-Convolution, STFNet-Pooling still facilitate the learning in time-frequency domain.

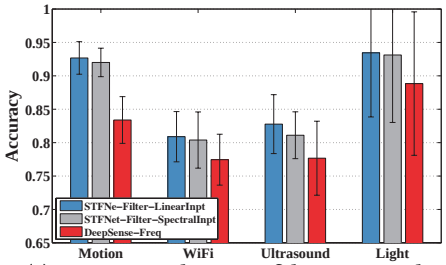
4.4.2 Spectral Padding v.s. Zero Padding. Next, we validate our design of spectral padding in the STFNet-Convolution operation as shown in Figure 5. In this experiment, we add a new baseline algorithm, STFNet-Conv-zPad, by replacing spectral padding with traditional zero padding in the STFNet-Conv. The accuracy and F1 score of all four tasks are shown in Figure 12. Here, DeepSense-Freq is still treated as a performance low-bound. By comparing STFNet-Conv-zPad and STFNet-Conv, we can see that spectral padding consistently helps improving the model performance. In most cases, the improvement is limited. However, in the case of visible light, spectral padding significantly improves both accuracy and F1 score. Therefore, designing neural network by preserving the time-frequency semantics of sensing signal is an important rule to follow.

4.4.3 Linear Interpolation v.s. Spectral Interpolation. Then, we compare our two designs of weight interpolation method in the STFNet-Filtering operation, linear interpolation and spectral interpolation, as shown in Figure 4. The STFNet-Filter defined in Section 4.2 uses linear interpolation, so we rename it as STFNet-Filter-LinearInpt in this experiment. We add a new baseline model called STFNet-Filter-SpectralInpt by using spectral interpolation instead of linear interpolation in STFNet-Filter. The results of all four tasks are illustrated in Figure 13. In general, the performance of two design choices are almost the same. At most of time, linear interpolation performs slightly better. In addition, we recommend using linear interpolation, since its implementation is easier,

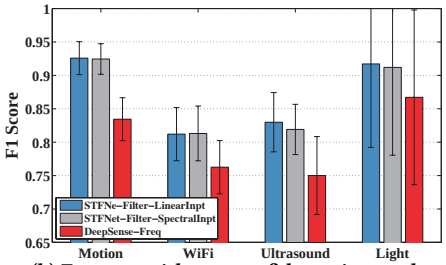
4.4.4 STFNet Pooling v.s. Mean/Max Pooling. Finally, we validate our design of STFNet-Pooling (low-pass deisgn) as shown in Figure 6. In this experiment, we add two new baseline algorithms, STFNet-Filter-mPad and STFNet-Conv-mPad, by replacing STFNet-Pooling in STFNet-Filter and STFNet-Conv with traditional max/mean pooling in the time domain (through choosing the one has better accuracy). The results are illustrated in Figure 14. In all settings, STFNet-Pooling shows better performance. In some cases, the improvement are significant. We believe that STFNet-Pooling can achieve even better performance if given the detailed signal-to-noise ratio over the frequency domain for each specific sensor. Then we can employ other pooling strategies instead of the low-pass design.

5 DISCUSSION

This paper provides a principled way of designing neural networks for sensing signals inspired by the fundamental nature of the underlying physical processes. STFNet, operates directly in the frequency

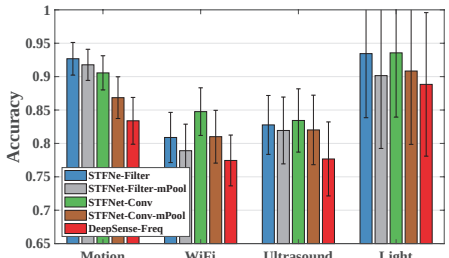


(a) Accuracy with 95% confidence interval.

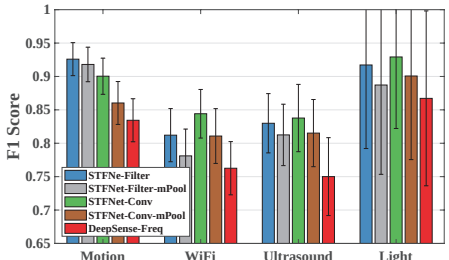


(b) F1 score with 95% confidence interval.

Figure 13: Linear Interpolation v.s. Spectral Interpolation



(a) Accuracy with 95% confidence interval.



(b) F1 score with 95% confidence interval.

Figure 14: STFNet Pooling v.s. Mean/Max Pooling

domain, in which the measured physical phenomena are best exposed. We propose three types of learnable frequency manipulations that are able to operate on multi-resolution representations, while preserving the underlying time-frequency information. Although extensive experiments have illustrated the superior performance of STFNet, further research is needed to better understand design choices for neural networks from the time-frequency perspective.

One challenge is to explore the possibility of integrating neural networks with other time-frequency transformations. In this paper, STFNet focuses on the short-time Fourier Transform. However, STFT is the most basic one. There are plenty of other transformation basis functions in traditional time-frequency analysis. How to naturally integrate them with neural network while keeping the underlying physical meaning within transformed representations? How to choose or design the most suitable transformation basis functions that meet the corresponding mathematical requirements? Answers to these questions can greatly impact the way researchers design neural networks for sensing signal processing. Another challenge is to empower the frequency manipulations to have heterogeneous behaviours over the time. In STFNet, all designed operations are learnable frequency manipulations, which perform identically over time. In order to fully exploit the potential of time-frequency analysis, further research is needed on designing time-varying time-frequency manipulations, that adapt to current temporal patterns.

Furthermore, a better experimental and theoretical understanding is needed of the basic settings of neural networks to support computation in time-frequency domain. For traditional real-value neural networks, researchers have good intuitions about the basic configurations of initialization, activation functions, dropout and normalization techniques, and optimization methods. However, for neural network in the time-frequency domain, our understanding is limited. Although the reseach community started to study the basic settings of neural networks with complex values [24], the current understanding remains preliminary. Time-frequency analysis can have operations in both the real and complex domains. At the same time, the underlying time-frequency information within the internal representations can make the related studies even more complicated. We believe that this understanding will greatly facilitate future design of deep learning systems for IoT.

In addition, outside the IoT context, there exists a large number of transformations and dimension reduction techniques, such as SVD and PCA, that have made great impact in revealing useful features of complex phenomena. Our study of deep learning with STFT suggests that integrating deep neural networks with other common transformations may facilitate learning in domains where such transformations reveal essential features of the input signal domain. Future work is needed to explore this conjecture.

6 CONCLUSION

In this paper, we introduced STFNet, a principled way of designing neural networks from the time-frequency perspective. STFNet endows time-frequency analysis with additional flexibility and capability. In addition to just parameterizing the frequency manipulations with deep neural networks, we bring two key insights into the design of STFNet. On one hand, STFNet leverages and preserves the frequency domain semantics that encode time and frequency information. On the other hand, STFNet circumvents the uncertainty principle through multi-resolution transform and processing. Evaluations show that STFNet consistently outperforms the state-of-the-art deep learning models with a clear margin under diverse sensing modalities, and our two designing insights significantly contribute to the improvement. The designs and evaluations of STFNet unveil the benefits of incorporating domain-specific modeling and transformation techniques into neural network design.

ACKNOWLEDGMENTS

Research reported in this paper was sponsored in part by NSF under grants CNS 16-18627 and CNS 13-20209 and in part by the Army Research Laboratory under Coop- erative Agreements W911NF-09-2-0053 and W911NF-17-2-0196. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory, NSF, or the U.S. Govern- ment. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Sourav Bhattacharya and Nicholas D Lane. 2016. Sparsification and separation of deep learning layers for constrained resource inference on wearables. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*. ACM, 176–189.

[3] Ke-Yu Chen, Daniel Ashbrook, Mayank Goel, Sung-Hyuck Lee, and Shwetak Patel. 2014. AirLink: sharing files between multiple devices using in-air gestures. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 565–569.

[4] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[5] Yarin Gal and Zoubin Ghahramani. 2015. Dropout as a Bayesian approximation. *arXiv preprint arXiv:1506.02157* (2015).

[6] Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*. 1019–1027.

[7] Rafael C Gonzalez, Richard E Woods, et al. 2002. Digital image processing.

[8] Sidhant Gupta, Daniel Morris, Shwetak Patel, and Desney Tan. 2012. Soundwave: using the doppler effect to sense gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1911–1914.

[9] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 2011. Tool release: Gathering 802.11 n traces with channel state information. *ACM SIGCOMM Computer Communication Review* 41, 1 (2011), 53–53.

[10] Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149* (2015).

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[12] Samuli Hemminki, Petteri Nurmi, and Sasu Tarkoma. 2013. Accelerometer-based transportation mode detection on smartphones. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*. ACM, 13.

[13] David H Hubel and Torsten N Wiesel. 1968. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology* 195, 1 (1968), 215–243.

[14] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[16] Nicholas D Lane, Petko Georgiev, and Lorena Qendro. 2015. DeepEar: robust smartphone audio sensing in unconstrained acoustic environments using deep learning. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 283–294.

[17] Tianxing Li, Qiang Liu, and Xia Zhou. 2016. Practical human sensing in the light. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 71–84.

[18] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[19] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2013. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th annual international conference on Mobile computing & networking*. ACM, 27–38.

[20] Oren Rippel, Jasper Snoek, and Ryan P Adams. 2015. Spectral representations for convolutional neural networks. In *Advances in neural information processing systems*. 2449–2457.

[21] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[22] Julius Orion Smith. 2007. *Mathematics of the discrete Fourier transform (DFT): with audio applications*. Julius Smith.

[23] Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. 2015. Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. ACM, 127–140.

[24] Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. 2017. Deep complex networks. *arXiv preprint arXiv:1705.09792* (2017).

[25] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and modeling of wifi signal based human activity recognition. In *Proceedings of the 21st annual international conference on mobile computing and networking*. ACM, 65–76.

[26] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. 2017. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 351–360.

[27] Shuochao Yao, Yiran Zhao, Huajie Shao, ShengZhong Liu, Dongxin Liu, Lu Su, and Tarek Abdelzaher. 2018. Fastdeepiot: Towards understanding and optimizing neural network execution time on mobile and embedded devices. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*. ACM, 278–291.

[28] Shuochao Yao, Yiran Zhao, Huajie Shao, Aston Zhang, Chao Zhang, Shen Li, and Tarek Abdelzaher. 2018. Rdeepsense: Reliable deep mobile computing models with uncertainty estimations. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (2018), 173.

[29] Shuochao Yao, Yiran Zhao, Huajie Shao, Chao Zhang, Aston Zhang, Shaohan Hu, Dongxin Liu, Shengzhong Liu, Lu Su, and Tarek Abdelzaher. 2018. Sensegan: Enabling deep learning for internet of things with a semi-supervised framework. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 3 (2018), 144.

[30] Shuochao Yao, Yiran Zhao, Huajie Shao, Chao Zhang, Aston Zhang, Dongxin Liu, Shengzhong Liu, Lu Su, and Tarek Abdelzaher. 2018. Apdeepsense: Deep learning uncertainty estimation without the pain for iot applications. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 334–343.

[31] Shuochao Yao, Yiran Zhao, Aston Zhang, Shaohan Hu, Huajie Shao, Chao Zhang, Lu Su, and Tarek Abdelzaher. 2018. Deep Learning for the Internet of Things. *Computer* 51, 5 (2018), 32–41.

[32] Shuochao Yao, Yiran Zhao, Aston Zhang, Lu Su, and Tarek Abdelzaher. 2017. Deepiot: Compressing deep neural network structures for sensing systems with a compressor-critic framework. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. ACM, 4.

[33] Fisher Yu and Vladlen Koltun. 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122* (2015).

CoaCor: Code Annotation for Code Retrieval with Reinforcement Learning

Ziyu Yao
The Ohio State University
yao.470@osu.edu

Jayavardhan Reddy Peddamail
The Ohio State University
peddamail.1@osu.edu

Huan Sun
The Ohio State University
sun.397@osu.edu

ABSTRACT

To accelerate software development, much research has been performed to help people understand and reuse the huge amount of available code resources. Two important tasks have been widely studied: *code retrieval*, which aims to retrieve code snippets relevant to a given natural language query from a code base, and *code annotation*, where the goal is to annotate a code snippet with a natural language description. Despite their advancement in recent years, the two tasks are mostly explored separately. In this work, we investigate a novel perspective of *Code annotation for Code retrieval* (hence called "CoaCor"), where a code annotation model is trained to generate a natural language annotation that can represent the semantic meaning of a given code snippet and can be leveraged by a code retrieval model to better distinguish relevant code snippets from others. To this end, we propose an effective framework based on reinforcement learning, which explicitly encourages the code annotation model to generate annotations that can be used for the retrieval task. Through extensive experiments, we show that code annotations generated by our framework are much more detailed and more useful for code retrieval, and they can further improve the performance of existing code retrieval models significantly.[1]

CCS CONCEPTS

• **Information systems** → **Novelty in information retrieval**; **Summarization**; • **Software and its engineering**; • **Computing methodologies** → *Reinforcement learning*; *Markov decision processes*; Neural networks;

KEYWORDS

Code Annotation; Code Retrieval; Reinforcement Learning

ACM Reference Format:
Ziyu Yao, Jayavardhan Reddy Peddamail, and Huan Sun. 2019. CoaCor: Code Annotation for Code Retrieval with Reinforcement Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313632

1 INTRODUCTION

Software engineering plays an important role in modern society. Almost every aspect of human life, including health care, education, transportation and web security, depends on reliable software [1]. Unfortunately, developing and maintaining large code bases are very costly. Understanding and reusing billions of lines of code in online open-source repositories can significantly speed up the software development process. Towards that, *code retrieval* (CR) and *code annotation* (CA) are two important tasks that have been widely studied in the past few years [1, 13, 19, 21, 58], where the former aims to retrieve relevant code snippets based on a natural language (NL) query while the latter is to generate natural language descriptions to describe what a given code snippet does.

Most existing work [2, 13, 19, 22, 58, 60] study either code annotation or code retrieval individually. Earlier approaches for code retrieval drew inspiration from the information retrieval field [14, 17, 23, 32] and suffered from surface form mismatches between natural language queries and code snippets [6, 34]. More recently, advanced deep learning approaches have been successfully applied to both code retrieval and code annotation [1, 2, 9, 13, 19–22, 31, 58, 60]. For example, the code retrieval model proposed by Gu et al. [13] utilized two deep neural networks to learn the vector representation of a natural language query and that of a code snippet respectively, and adopted cosine similarity to measure their matching degree. For code annotation, Iyer et al. [21] and Hu et al. [19] utilized encoder-decoder models with an attention mechanism to generate an NL annotation for a code snippet. They aim to generate annotations similar to the human-provided ones, and therefore trained the models using the standard maximum likelihood estimation (MLE) objective. For the same purpose, Wan et al. [58] trained the code annotation model in a reinforcement learning (RL) framework with reward being the BLEU score [41], which measures n-gram matching precision between the currently generated annotation and the human-provided one.

In this work, we explore a novel perspective - code annotation *for* code retrieval (CoaCor), which is to generate an NL annotation for a code snippet so that the generated annotation *can be used for code retrieval* (i.e., *can represent the semantic meaning of a code snippet and distinguish it from others w.r.t. a given NL query in the code retrieval task*). As exemplified by [56], such an annotation can be taken as the representation of the corresponding code snippet, based on which the aforementioned lexical mismatch issue in a naive keyword-based search engine can be alleviated. A similar idea of improving retrieval by adding extra annotations to items is also explored in document retrieval [47] and image search [61]. However, most of them rely on humans to provide the annotations. Intuitively, our perspective can be interpreted as one type of *machine-machine collaboration*: On the one hand, the NL annotation generated by the code annotation model can serve as a second view of a code snippet (in addition to its programming content) and can be utilized to match with an NL query in code retrieval. On the other hand, with

[1]Code available at https://github.com/LittleYUYU/CoaCor.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313632

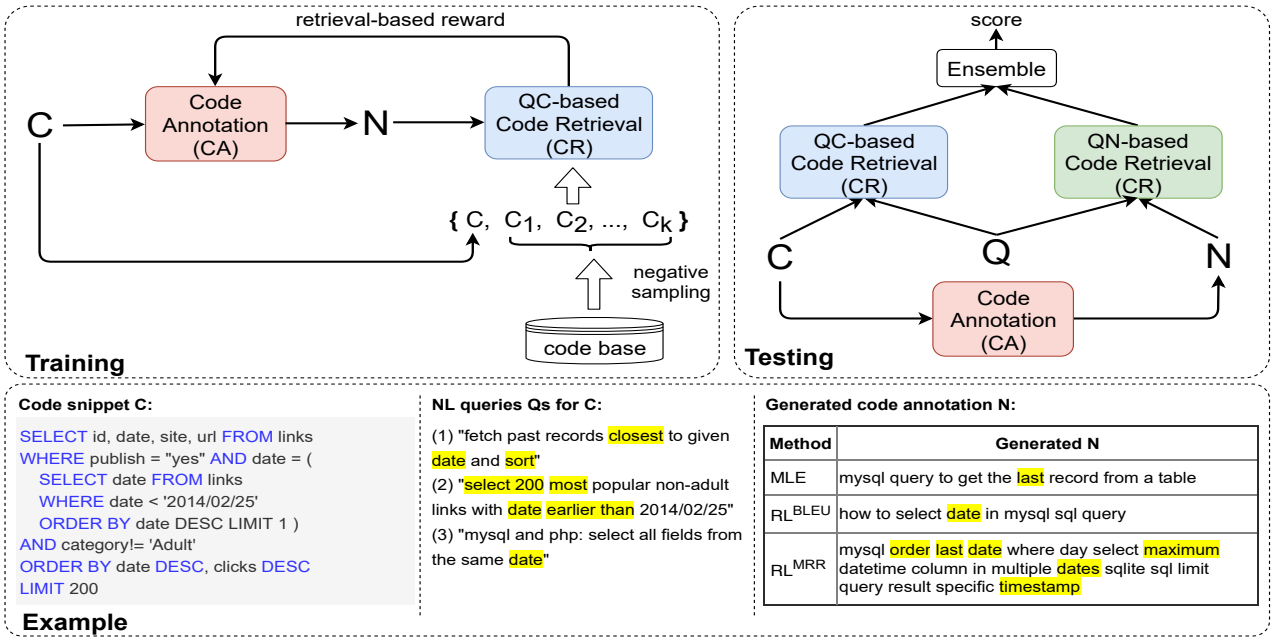


Figure 1: Our CoaCor framework: (1) *Training phase.* A code annotation model is trained via reinforcement learning to maximize retrieval-based rewards given by a QC-based code retrieval model (pre-trained using <NL query, code snippet> pairs). (2) *Testing phase.* Each code snippet is first annotated by the trained CA model. For the code retrieval task, given query Q, a code snippet gets two scores - one matching Q with its code content and the other matching Q with its code annotation N, and is ranked by a simple ensemble strategy[+]. (3) *Example.* We show an example of a code snippet and its associated multiple NL queries in our dataset. The code annotation generated by our framework (denoted as RL^{MRR}) is much more detailed with many keywords *semantically aligned* with Qs, when compared with CA models trained via MLE or RL with BLEU rewards (RL^{BLEU}). [+] We simply use a weighted combination of the two scores, and other ensemble strategies can also apply here.

a goal to facilitate code retrieval, the code annotation model can be stimulated to produce rich and detailed annotations. Unlike existing work [19–21, 58], our goal is *not* to generate an NL annotation as close as possible to a human-provided one; hence, the MLE objective or the BLEU score as rewards for code annotation will not fit our setting. Instead, we design a novel rewarding mechanism in an RL framework, which guides the code annotation model directly based on how effectively the currently generated code annotation distinguishes the code snippet from a candidate set.

Leveraging collaborations and interactions among machine learning models to improve the task performance has been explored in other scenarios. Goodfellow et al. [12] proposed the Generative Adversatial Nets (GANs), where a generative model produces difficult examples to fool a discriminative model and the latter is further trained to conquer the challenge. He et al. [15] proposed another framework called *dual learning*, which jointly learns two dual machine translation tasks (e.g., En -> Fr and Fr -> En). However, none of the existing frameworks are directly applicable to accomplish our goal (i.e., to train a code annotation model for generating annotations that can be utilized for code retrieval).

Figure 1 shows our reinforcement learning-based CoaCor framework. In the training phase, we first train a CR model based on <natural language query, code snippet> (QC) pairs (referred to as QC-based CR model). Then given a code snippet, the CA model generates a sequence of NL tokens as its annotation and receives a reward from the trained CR model, which measures how effectively the generated annotation can distinguish the code snippet from others. We formulate the annotation generation process as a Markov Decision Process [5] and train the CA model to maximize the received reward via an advanced reinforcement learning [53] framework called *Advantage Actor-Critic* or *A2C* [36]. Once the CA model is trained, we use it to generate an NL annotation N for each code snippet C in the code base. Therefore, for each QC pair we originally have, we can derive a QN pair. We utilize the generated annotation as a second view of the code snippet to match with a query and train another CR model based on the derived QN pairs (referred to as QN-based CR model). In the testing phase, given an NL query, we rank code snippets by combining their scores from both the QC-based as well as QN-based CR models, which utilize both the programming content as well as the NL annotation of a code snippet.

On a widely used benchmark dataset [21] and a recently collected large-scale dataset [63], we show that the automatically generated annotation can significantly improve the retrieval performance. More impressively, without looking at the code content, the QN-based CR model trained on our generated code annotations obtains a retrieval performance comparable to one of the state-of-the-art

QC-based CR models. It also surpasses other QN-based CR models trained using code annotations generated by existing CA models.

To summarize, our major contributions are as follows:

- First, we explored a novel perspective of generating useful code annotations *for* code retrieval. Unlike existing work [19, 21, 58], we do not emphasize the n-gram overlap between the generated annotation and the human-provided one. Instead, we examined the real usefulness of the generated annotations and developed a machine-machine collaboration paradigm, where a code annotation model is trained to generate annotations that can be used for code retrieval.
- Second, in order to accomplish our goal, we developed an effective RL-based framework with a novel rewarding mechanism, in which a code retrieval model is directly used to formulate rewards and guide the annotation generation.
- Last, we conducted extensive experiments by comparing our framework with various baselines including state-of-the-art models and variants of our framework. We showed significant improvements of code retrieval performance on both a widely used benchmark dataset and a recently collected large-scale dataset.

The rest of this paper is organized as follows. Section 2 introduces the background on code annotation and code retrieval tasks. Section 3 gives an overview of our proposed framework, with algorithm details followed in Section 4. Experiments are shown in Section 5. Finally, we discuss related work and conclude in Section 6 and 7.

2 BACKGROUND

We adopt the same definitions for code retrieval and code annotation as previous work [9, 21]. Given a natural language query Q and a set of code snippet candidates $\mathbb{C}$, *code retrieval* is to retrieve code snippets $C^* \in \mathbb{C}$ that can match with the query. On the other hand, given a code snippet C, *code annotation* is to generate a natural language (NL) annotation N^* which describes the code snippet appropriately. In this work, we use *code search* and *code retrieval* interchangeably (and same for *code annotation/summary/description*).

Formally, for a training corpus with <natural language query, code snippet> pairs, e.g., those collected from Stack Overflow [51] by [21, 63], we define the two tasks as:

Code Retrieval (CR): Given an NL Query Q, a model F_r will be learnt to retrieve the highest scoring code snippet $C^* \in \mathbb{C}$.

$$C^* = \underset{C \in \mathbb{C}}{\operatorname{argmax}} F_r(Q, C) \tag{1}$$

Code Annotation (CA): For a given code snippet C, the goal is to generate an NL annotation N^* that maximizes a scoring function F_a:

$$N^* = \underset{N}{\operatorname{argmax}} F_a(C, N) \tag{2}$$

Note that one can use the same scoring model for F_r and F_a as in [9, 21], but for most of the prior work [13, 19, 20, 58], which consider either code retrieval or code annotation, researchers usually develop their own models and objective functions for F_r or F_a. In our work, we choose two vanilla models as our base models for CR and CA, but explore a novel perspective of how to train F_a so that it can generate NL annotations that can be used for code retrieval. This perspective is inspired by various machine-machine collaboration mechanisms [15, 28, 55, 59] where one machine learning task can help improve another.

3 FRAMEWORK OVERVIEW

In this section, we first introduce our intuition and give an overview of the entire framework, before diving into more details.

3.1 Intuition behind CoaCor

To the best of our knowledge, previous code annotation work like [19–21, 58] focused on getting a large n-gram overlap between generated and human-provided annotations. However, it is still uncertain (and non-trivial to test) how helpful the generated annotations can be. Driven by this observation, we are the first to examine the real usefulness of the generated code annotations and how they can help a relevant task, of which we choose code retrieval as an example.

Intuitively, CoaCor can be interpreted as a *collaboration mechanism* between code annotation and code retrieval. On the one hand, the annotation produced by the CA model provides a second view of a code snippet (in addition to its programming content) to assist code retrieval. On the other hand, when the CA model is trained to be useful for the retrieval task, we expect it to produce richer and more detailed annotations, which we verify in experiments later.

3.2 Overview

The main challenge to realize the above intuition lies in how to train the CA model effectively. Our key idea to address the challenge is shown in Figure 1.

We first train a base CR model on <natural language query, code snippet> (QC) pairs. Intuitively, a QC-based CR model ranks a code snippet C by measuring how well it matches with the given query Q (in comparison with other code snippets in the code base). From another point of view, a well-trained QC-based CR model can work as a measurement on whether the query Q describes the code snippet C precisely or not. Drawing inspiration from this view, we propose using the trained QC-based CR model to determine whether an annotation describes its code snippet precisely or not and thereby, train the CA model to generate rich annotations to maximize the retrieval-based reward from the CR model. Specifically, given a code snippet C, the CA model generates a sequence of NL words as its annotation N. At the end of the sequence, we let the trained QC-based CR model use N to search for relevant code snippets from the code base. If C can be ranked at top places, the annotation N is treated as well-written and gets a high reward; otherwise, a low reward will be returned. We formulate this generation process as the Markov Decision Process [5] and train the CA model with reinforcement learning [53] (specifically, the *Advantage Actor-Critic* algorithm [36]) to maximize the retrieval-based rewards it can receive from the QC-based CR model. We elaborate the CR and CA model details as well as the RL algorithm for training the CA model in Section 4.1 ∼ 4.3.

Once the CA model is trained, in the testing phase, it generates an NL annotation N for each code snippet C in the code base. Now for each <NL query, code snippet> pair originally in the datasets,

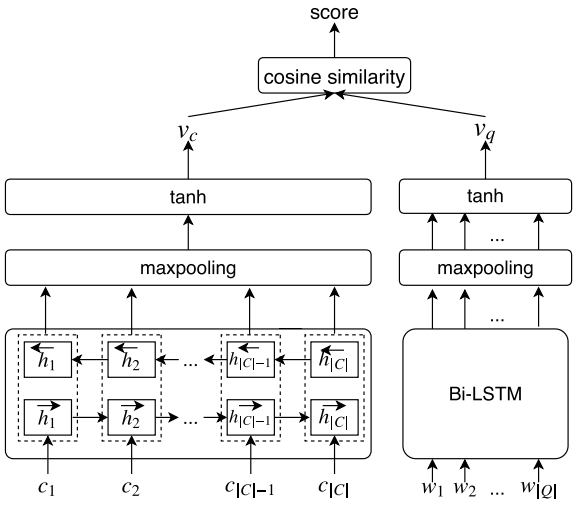


Figure 2: The base code retrieval model encodes the input code snippet C = $(c_1, c_2, ..., c_{|C|})$ and NL query Q = $(w_1, w_2, ..., w_{|Q|})$ into a vector space and outputs a similarity score.

we derive an <NL query, code annotation> (QN) pair and train another CR model based on such QN pairs. This QN-based CR model complements the QC-based CR model, as they respectively use the annotation and programming content of a code snippet to match with the query. We finally combine the matching scores from the two CR models to rank code snippets for a given query.

Note that we aim to outline a general paradigm to explore the perspective of code annotation for code retrieval, where the specific model structures for the two CR models and the CA model can be instantiated in various ways. In this work, we choose one of the simplest and most fundamental deep structures for each of them. Using more complicated model structures will make the training more challenging and we leave it as future work.

4 CoDE ANNOTATION FOR CoDE RETRIEVAL

Now we introduce model and algorithm details in our framework.

4.1 Code Retrieval Model

Both QC-based and QN-based code retrieval models adopt the same deep learning structure as the previous CR work [13]. Here for simplicity, we only illustrate the QC-based CR model in detail, and the QN-based model structure is the same except that we use the generated annotation on the code snippet side.

As shown in Figure 2, given an NL query $Q = w_{1..|Q|}$ and a code snippet $C = c_{1..|C|}$, we first embed the tokens of both code and NL query into vectors through a randomly initialized word embedding matrix, which will be learned during model training. We then use a bidirectional Long Short-Term Memory (LSTM)-based Recurrent Neural Network (RNN) [10, 11, 18, 35, 48, 64] to learn the token representation by summarizing the contextual information from both directions. The LSTM unit is composed of three multiplicative gates. At every time step t, it tracks the state of sequences by controlling how much information is updated into the new hidden state h_t and memory cell g_t from the previous state h_{t-1}, the previous memory cell g_{t-1} and the current input x_t. On the code side, x_t is the embedding vector for c_t, and on the NL query side, it is the embedding vector for w_t. At every time step t, the LSTM hidden state is updated as:

$$
\begin{aligned}
i &= \sigma(\mathbf{W}_i h_{t-1} + \mathbf{U}_i x_t + b_i) \\
f &= \sigma(\mathbf{W}_f h_{t-1} + \mathbf{U}_f x_t + b_f) \\
o &= \sigma(\mathbf{W}_o h_{t-1} + \mathbf{U}_o x_t + b_o) \\
g &= \tanh(\mathbf{W}_g h_{t-1} + \mathbf{U}_g x_t + b_g) \\
g_t &= f \odot g_{t-1} + i \odot g \\
h_t &= o \odot \tanh(g_t)
\end{aligned}
$$

where σ is the element-wise sigmoid function and $\odot$ is the element-wise product. $\mathbf{U}_i, \mathbf{U}_f, \mathbf{U}_g, \mathbf{U}_o$ denote the weight matrices of different gates for input x_t and $\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_g$, $\mathbf{W}_o$ are the weight matrices for hidden state h_t, while b_i, b_f, b_g, b_o denote the bias vectors. For simplicity, we denote the above calculation as below (the memory cell vector g_{t-1} is omitted):

$$h_t = \text{LSTM}(x_t, h_{t-1}) \tag{3}$$

The vanilla LSTM's hidden state h_t takes information from the past, knowing nothing about the future. Our CR model instead incorporates a bidirectional LSTM [48] (i.e., Bi-LSTM in Figure 2), which contains a forward LSTM reading a sequence X from start to end and a backward LSTM which reads from end to start. The basic idea of using two directions is to capture past and future information at each step. Then the two hidden states at each time step t are concatenated to form the final hidden state h_t.

$$
\begin{aligned}
\overrightarrow{h_t} &= \text{LSTM}(x_t, \overrightarrow{h_{t-1}}) \\
\overleftarrow{h_t} &= \text{LSTM}(x_t, \overleftarrow{h_{t+1}}) \\
h_t &= [\overrightarrow{h_t}, \overleftarrow{h_t}]
\end{aligned}
$$

Finally, we adopt the commonly used max pooling strategy [24] followed by a *tanh* layer to get the embedding vector for a sequence of length T.

$$v = \tanh(\text{maxpooling}([h_1, h_2, ..., h_T])) \tag{4}$$

By applying the above encoding algorithm, we encode the code snippet C and the NL query Q into v_c and v_q, respectively. Similar to Gu et al. [13], to measure the relevance between the code snippet and the query, we use cosine similarity denoted as $cos(v_q, v_c)$. The higher the similarity, the more related the code is to the query.

Training. The CR model is trained by minimizing a ranking loss similar to [13]. Specifically, for each query Q in the training corpus, we prepare a triple of $<Q, C, C^->$ as a training instance, where C is the correct code snippet that answers Q and C^- is a negative code snippet that does not answer Q (which is randomly sampled from the entire code base). The ranking loss is defined as:

$$\mathcal{L}(\theta) = \sum_{<Q,C,C^->} max(0, \epsilon - cos(v_q, v_c) + cos(v_q, v_{c^-})) \tag{5}$$

where θ denotes the model parameters, ϵ is a constant margin, and v_q, v_c and v_{c^-} are the encoded vectors of Q, C and C^- respectively.

Essentially, the ranking loss is a kind of hinge loss [46] that promotes the cosine similarity between Q and C to be greater than that between Q and C^- by at least a margin ϵ. The training leads the model to project relevant queries and code snippets to be close in the vector space.

4.2 Code Annotation Model

Formally, given a code snippet C, the CA model computes the probability of generating a sequence of NL tokens $n_{1..|N|} = (n_1, n_2, ..., n_{|N|})$ as its annotation N by:

$$P(N|C) = P(n_1|n_0, C) \prod_{t=2}^{|N|} P(n_t|n_{1..t-1}, C) \tag{6}$$

where n_0 is a special token "<START>" indicating the start of the annotation generation, $n_{1..t-1} = (n_1, ..., n_{t-1})$ is the partially generated annotation till time step t-1, and $P(n_t|n_{1..t-1}, C)$ is the probability of producing n_t as the next word given the code snippet C and the generated $n_{1..t-1}$. The generation stops once a special token "<EOS>" is observed.

In this work, we choose the popular sequence-to-sequence model [52] as our CA model structure, which is composed of an encoder and a decoder. We employ the aforementioned bidirectional LSTM-based RNN structure as the encoder for code snippet C, and use another LSTM-based RNN as the decoder to compute Eqn. (6):

$$h_t^{\text{dec}} = \text{LSTM}(n_{t-1}, h_{t-1}^{\text{dec}}), \forall t = 1, ..., |N|$$

where h_t^{dec} is the decoder hidden state at step t, and h_0^{dec} is initialized by concatenating the last hidden states of the code snippet encoder in both directions. In addition, a standard global attention layer [33] is applied in the decoder, in order to attend to important code tokens in C:

$$\tilde{h}_t^{\text{dec}} = \tanh(\mathbf{W}_\alpha[v_{\text{attn}}, h_t^{\text{dec}}])$$

$$v_{\text{attn}} = \sum_{t'=1}^{|C|} \alpha_{t'}\ h_{t'}^{\text{enc}}$$

$$\alpha_{t'} = \text{softmax}((h_{t'}^{\text{enc}})^T h_t^{\text{dec}})$$

where $\alpha_{t'}$ is the attention weight on the t'-th code token when generating the t-th word in the annotation, and $\mathbf{W}_\alpha$ is a learnable weight matrix. Finally, the t-th word is selected based on:

$$P(n_t|n_{0..t-1}, C) = \text{softmax}(\mathbf{W}\tilde{h}_t^{\text{dec}} + b) \tag{7}$$

where $\mathbf{W} \in R^{|V_n| \times d}$, $b \in R^{|V_n|}$ project the d-dim hidden state $\tilde{h}_t^{\text{dec}}$ to the NL vocabulary of size $|V_n|$.

4.3 Training Code Annotation via RL

4.3.1 Code Annotation as Markov Decision Process. Most previous work [9, 19, 21] trained the CA model by maximizing the log-likelihood of generating human annotations, which suffers from two drawbacks: (1) The *exposure bias* issue [4, 44, 45]. That is, during training, the model predicts the next word given the ground-truth annotation prefix, while at testing time, it generates the next word based on previous words generated by itself. This mismatch between training and testing may result in error accumulation in testing phase. (2) More importantly, maximizing the likelihood is not aligned with our goal to produce annotations that can be useful for code retrieval.

To address the above issues, we propose to formulate code annotation within the reinforcement learning (RL) framework [53]. Specifically, the annotation generation process is viewed as a Markov Decision Process (MDP) [5] consisting of four main components:

State. At step t during decoding, a state s_t maintains the source code snippet and the previously generated words $n_{1..t-1}$, i.e., $s_t = \{C, n_{1..t-1}\}$. In particular, the initial decoding state $s_0 = \{C\}$ only contains the given code snippet C. In this work, we take the hidden state vector $\tilde{h}_t^{dec}$ as the vector representation of state s_t, and the MDP is thus processing on a continuous and infinite state space.

Action. The CA model decides the next word (or *action*) $n_t \in V_n$, where V_n is the NL vocabulary. Thus, the action space in our formulation is the NL vocabulary.

Reward. As introduced in Section 3, to encourage it to generate words useful for the code retrieval task, the CA model is rewarded by a well-trained QC-based CR model based on whether the code snippet C can be ranked at the top positions if using the generated *complete* annotation N as a query. Therefore, we define the reward at each step t as:

$$r(s_t, n_t) = \begin{cases} \text{RetrievalReward}(C, n_{1..t}) & \text{if } n_t = \text{<EOS>} \\ 0 & \text{otherwise} \end{cases}$$

where we use the popular ranking metric Mean Reciprocal Rank [57] (defined in Section 5.2) as the RetrievalReward(C, N) value. Note that, in this work, we let the CR model give a valid reward only when the annotation generation stops, and assign a zero reward to intermediate steps. However, other designs such as giving rewards to a partial generation with the reward shaping technique [4] can be reasonable and explored in the future.

Policy. The policy function $P(n_t|s_t)$ takes as input the current state s_t and outputs the probability of generating n_t as the next word. Given our definition about state s_t and Eqn. (7),

$$P(n_t|s_t) = P(n_t|n_{1..t-1}, C) = \text{softmax}(\mathbf{W}\tilde{h}_t^{dec} + b) \tag{8}$$

Here, the policy function is stochastic in that the next word can be sampled according to the probability distribution, which allows action space exploration and can be optimized using policy gradient methods [53].

The objective of the CA model training is to find a policy function $P(N|C)$ that maximizes the expected accumulative future reward:

$$\max_\phi \mathcal{L}(\phi) = \max_\phi \mathbb{E}_{N \sim P(\cdot|C;\phi)}[R(C, N)] \tag{9}$$

where ϕ is the parameter of P and $R(C, N) = \sum_{t=1}^{|N|} r(s_t, n_t)$ is the accumulative future reward (called "return").

The gradient of the above objective is derived as below.

$$\begin{aligned} \nabla_\phi \mathcal{L}(\phi) &= \mathbb{E}_{N \sim P(\cdot|C;\phi)}[R(C, N) \nabla_\phi \log P(N|C;\phi)] \\ &= \mathbb{E}_{n_{1..|N|} \sim P(\cdot|C;\phi)}[\sum_{t=1}^{|N|} R_t(s_t, n_t) \nabla_\phi \log P(n_t|n_{1..t-1}, C;\phi)] \end{aligned} \tag{10}$$

where $R_t(s_t, n_t) = \sum_{t' \geq t} r(s_{t'}, n_{t'})$ is the return for generating word n_t given state s_t.

4.3.2 Advantage Actor-Critic for Code Annotation. Given the gradient in Eqn. (10), the objective in Eqn. (9) can be optimized by policy gradient approaches such as REINFORCE [62] and Q-value Actor-Critic algorithm [4, 54]. However, such methods may yield very high variance when the action space (i.e., the NL vocabulary) is large and suffer from biases when estimating the return of rarely taken actions [39], leading to unstable training. To tackle the challenge, we resort to the more advanced *Advantage Actor-Critic* or *A2C* algorithm [36], which has been adopted in other sequence generation tasks [39, 58]. Specifically, the gradient function in Eqn. (10) is replaced by an *advantage* function:

$$\nabla_\phi \mathcal{L}(\phi) = \mathbb{E}[\sum_{t=1}^{|N|} A_t \nabla_\phi \log P(n_t|n_{1..t-1}, C; \phi)] \tag{11}$$
$$A_t = R_t(s_t, n_t) - V(s_t)$$
$$V(s_t) = \mathbb{E}_{\hat{n}_t \sim P(\cdot|n_{1..t-1}, C)}[R_t(s_t, \hat{n}_t)]$$

where $V(s_t)$ is the state value function that estimates the future reward given the current state s_t. Intuitively, $V(s_t)$ works as a *baseline* function [62] to help the model assess its action n_t more precisely: When advantage A_t is greater than zero, it means that the return for taking action n_t is better than the "average" return over all possible actions, given state s_t; otherwise, action n_t performs worse than the average.

Following previous work [36, 39, 58], we approximate $V(s_t)$ by learning another model $V(s_t; \rho)$ parameterized by ρ, and the RL framework thus contains two components: the policy function $P(N|C; \phi)$ that generates the annotation (called "Actor"), and the state value function $V(s_t; \rho)$ that approximates the return under state s_t (called "Critic"). Similar to the actor model (i.e., the CA model in Section 4.2), we train a separate attention-based sequence-to-sequence model as the critic network. The critic value is finally computed by:

$$V(s_t; \rho) = \mathbf{w}_\rho^T (\tilde{h}_t^{dec})_{crt} + b_\rho \tag{12}$$

where $(\tilde{h}_t^{dec})_{crt} \in R^d$ is the critic decoder hidden state at step t, and $\mathbf{w}_\rho \in R^d$, $b_\rho \in R$ are trainable parameters that project the hidden state to a scalar value (i.e., the estimated state value).

The critic network is trained to minimize the Mean Square Error between its estimation and the true state value:

$$\min_\rho \mathcal{L}(\rho) = \min_\rho \mathbb{E}_{N \sim P(\cdot|C)}[\sum_{t=1}^{|N|} (V(s_t; \rho) - R(C, N))^2] \tag{13}$$

The entire training procedure of CoaCor is shown in Algorithm 1.

4.4 Generated Annotations for Code Retrieval

As previously shown in Figure 1, in the testing phase, we utilize the generated annotations to assist the code retrieval task. Now we detail the procedure in Algorithm 2. Specifically, for each <NL query, code snippet> pair (i.e., QC pair) in the dataset, we first derive an <NL query, code annotation> pair (i.e., QN pair) using the code annotation model. We then build another code retrieval (CR) model based on the QN pairs in our training corpus. In this work, for simplicity, we choose the same structure as the QC-based CR model (Section 4.1) to match QN pairs. However, more advanced methods for modeling the semantic similarity between two NL sentences (e.g., previous work on NL paraphase detection [16, 26]) are applicable and can be explored as future work.

Algorithm 1 : Training Procedure for CoaCor.

Input: <NL query, code snippet> (QC) pairs in training set, number of iterations E.

1: Train a base code retrieval model based on QC pairs, according to Eqn. (5).
2: Initialize a base code annotation model (ϕ) and pretrain it via MLE according to Eqn. (7), using Q as the desired N for C.
3: Pretrain a critic network (ρ) according to Eqn. (13).
4: **for** $iteration = 1$ to E **do**
5: Receive a code snippet C.
6: Sample an annotation $N \sim P(\cdot|C; \phi)$ according to Eqn. (8).
7: Receive the final reward $R(C, N)$.
8: Update the code annotation model (ϕ) using Eqn. (11).
9: Update the critic network (ρ) using Eqn. (13).
10: **end for**

Algorithm 2 : Generated Annotations for Code Retrieval.

Input: NL query Q, code snippet candidate C.
Output: The matching score, score(Q, C).

1: Receive score$_1(Q, C) = cos(v_q, v_c)$ from a QC-based code retrieval model.
2: Generate a code annotation $N \sim P(\cdot|C; \phi)$ via greedy search, according to Eqn. (8).
3: Receive score$_2(Q, C) = cos(v_q, v_n)$ from a QN-based code retrieval model.
4: Calculate score(Q, C) according to Eqn. (14).

The final matching score between query Q and code snippet C combines those from the QN-based and QC-based CR model:

$$\text{score}(Q, C) = \lambda * cos(v_q, v_n) + (1 - \lambda) * cos(v_q, v_c) \tag{14}$$

where v_q, v_c, v_n are the encoded vectors of Q, C, and the code annotation N respectively. $\lambda \in [0, 1]$ is a weighting factor for the two scores to be tuned on the validation set.

5 EXPERIMENTS

In this section, we conduct extensive experiments and compare our framework with various models to show its effectiveness.

5.1 Experimental Setup

Dataset. (1) We experimented with the **StaQC** dataset presented by Yao et al. [63]. The dataset contains 119,519 SQL <question title, code snippet> pairs mined from Stack Overflow [51], making itself the largest-to-date in SQL domain. In our code retrieval task, the question title is considered as the NL query Q, which is paired with the code snippet C to form the QC pair. We randomly selected 75% of the pairs for training, 10% for validation (containing 11,900 pairs), and the left 15% for testing (containing 17,850 pairs). As mentioned in [63], the dataset may contain multiple code snippets for the same NL query. We examine the dataset split to ensure that alternative relevant code snippets for the same query would not be sampled as negative code snippets when training the CR model. For pretraining the CA model, we consider the question title as a

code annotation N and form QN pairs accordingly. **(2)** Iyer et al. [21] collected two small sets of SQL code snippets (called "**DEV**" and "**EVAL**" respectively) from Stack Overflow for validation and testing. In addition to the originally paired question title, each code snippet is manually annotated by two different NL descriptions. Therefore, in total, each set contains around 100 code snippets with three NL descriptions (resulting in around 300 QC pairs). We use them as additional datasets for model comparison.[2] Following [21], QC pairs occuring in DEV and EVAL set or being used as negative code snippets by them are removed from the StaQC training set.

Data Preprocessing. We followed [21] to perform code tokenization, which replaced table/column names with placeholder tokens and numbered them to preserve their dependencies. For text tokenization, we utilized the "word_tokenize" tool in the NLTK toolkit [7]. All code tokens and NL tokens with a frequency of less than 2 were replaced with an <UNK> token, resulting in totally 7726 code tokens and 7775 word tokens in the vocabulary. The average lengths of the NL query and the code snippet are 9 and 60 respectively.

5.2 Evaluation

We evaluate a model's retrieval performance on four datasets: the validation (denoted as "StaQC-val") and test set (denoted as "StaQC-test") from StaQC [63], and the DEV and EVAL set from [21]. For each <NL query Q, code snippet C> pair (or QC pair) in a dataset, we take C as a positive code snippet and randomly sample K negative code snippets from all others except C in the dataset,[3] and calculate the rank of C among the $K+1$ candidates. We follow [9, 21, 63] to set $K = 49$. The retrieval performance of a model is then assessed by the Mean Reciprocal Rank (MRR) metric [57] over the entire set $\mathcal{D} = \{(Q_1, C_1), (Q_2, C_2), ..., (Q_{|\mathcal{D}|}, C_{|\mathcal{D}|})\}$:

$$MRR = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \frac{1}{Rank_i}$$

where $Rank_i$ is the rank of C_i for query Q_i. The higher the MRR value, the better the code retrieval performance.

5.3 Methods to Compare

In order to test the effectiveness of our CoaCor framework, we compare it with both existing baselines and our proposed variants.

Existing Baselines. We choose the following state-of-the-art code retrieval models, which are based on QC pairs, for comparison.

- Deep Code Search (DCS) [13]. The original DCS model [13] adopts a similar structure as Figure 2 for CR in Java domain. To learn the vector representations for code snippets, in addition to code tokens, it also considers features like function names and API sequences, all of which are combined into a fully connected layer. In our dataset, we do not have these features, and thus slightly modify their original model to be the same as our QC-based CR model (Figure 2).
- CODE-NN [21]. CODE-NN is one of the state-of-the-art models for both code retrieval and code annotation. Its core component is an LSTM-based RNN with an attention mechanism, which models the probability of generating an NL sentence conditioned on a given code snippet. For code retrieval, given an NL query, CODE-NN computes the likelihood of generating the query as an annotation for each code snippet and ranks code snippets based on the likelihood.

QN-based CR Variants. As discussed in Section 4.4, a trained CA model is used to annotate each code snippet C in our datasets with an annotation N. The resulting <NL query Q, code annotation N> pairs can be used to train a QN-based CR model. Depending on how we train the CA model, we have the following variants:

- QN-MLE. Similar to most previous work [9, 19, 21], we simply train the CA model in the standard MLE manner, i.e., by maximizing the likelihood of a human-provided annotation.
- QN-RL$^{\text{BLEU}}$. As introduced in Section 1, Wan et al. [58] proposed to train the CA model via reinforcement learning with BLEU scores [41] as rewards. We compare this variant with our rewarding mechanism.
- QN-RL$^{\text{MRR}}$. In our CoaCor framework, we propose to train the CA model using retrieval rewards from a QC-based CR model (see Section 3). Here we use the MRR score as the retrieval reward.

Since CODE-NN [21] can be used for code annotation as well, we also use its generated code annotations to train a QN-based CR model, denoted as "QN-CodeNN".[4]

Ensemble CR Variants. As introduced in Section 4.4, we tried ensembling the QC-based CR model and the QN-based CR model to improve the retrieval performance. We choose the DCS structure as the QC-based CR model as mentioned in Section 4.1. Since different QN-based CR models can be applied, we present the following 4 variants: (1) QN-MLE + DCS, (2) QN-RL$^{\text{BLEU}}$ + DCS, (3) QN-RL$^{\text{MRR}}$ + DCS, and (4) QN-CodeNN + DCS, where QN-MLE, QN-RL$^{\text{BLEU}}$, QN-RL$^{\text{MRR}}$, and QN-CodeNN have been introduced.

5.4 Implementation Details

Our implementation is based on Pytorch [42]. For CR models, we set the embedding size of words and code tokens to 200, and chose batch size in {128, 256, 512}, LSTM unit size in {100, 200, 400} and dropout rate [50] in {0.1, 0.35, 0.5}. A small, fixed ϵ value of 0.05 is used in all the experiments. Hyper-parameters for each model were chosen based on the DEV set. For CODE-NN baseline, we followed Yao et al. [63] to use the same model hyper-parameters as the original paper, except that the dropout rate is tuned in {0.5, 0.7}. The StaQC-val set was used to decay the learning rate and the best model parameters were decided based on the retrieval performance on the DEV set.

For CA models, the embedding size of words and code tokens and the LSTM unit size were selected from {256, 512}. The dropout rate is selected from {0.1, 0.3, 0.5} and the batch size is 64. We updated model parameters using the Adam optimizer [25] with learning rate 0.001 for MLE training and 0.0001 for RL training. The maximum length of the generated annotation is set to 20. For CodeNN, MLE-based and RL$^{\text{BLEU}}$-based CA models, the best model parameters

[2] Previous work [9, 21] used only one of the three descriptions while we utilize all of them to enrich and enlarge the datasets for a more reliable evaluation.
[3] For DEV and EVAL, we use the same negative examples as [21].
[4] There is another recent code annotation method named DeepCom [19]. We did not include it as baseline, since it achieved a similar performance as our MLE-based CA model (see Table 3) when evaluated with the standard BLEU script by [21].

Table 1: The main code retrieval results (MRR). * denotes significantly different from DCS [13] in one-tailed t-test (p < 0.01).

Model	DEV	EVAL	StaQC-val	StaQC-test
Existing (QC-based) CR Baselines				
DCS [13]	0.566	0.555	0.534	0.529
CODE-NN [21]	0.530	0.514	0.526	0.522
QN-based CR Variants				
QN-CodeNN	0.369	0.360	0.336	0.333
QN-MLE	0.429	0.411	0.427	0.424
QN-RL^{BLEU}	0.426	0.402	0.386	0.381
QN-RL^{MRR} (ours)	0.534	0.512	0.516	0.523
Ensemble CR Variants				
QN-CodeNN + DCS	0.566	0.555	0.534	0.529
QN-MLE + DCS	0.571	0.561	0.543	0.537
QN-RL^{BLEU} + DCS	0.570	0.559	0.541	0.534
QN-RL^{MRR} + DCS (ours)	0.582*	**0.572***	0.558*	0.559*
QN-RL^{MRR} + CODE-NN (ours)	**0.586***	0.571*	**0.575***	**0.576***

were picked based on the model's BLEU score on DEV, while for RL^{MRR}-based CA model, we chose the best model according to its MRR reward on StaQC-val. For RL models, after pretraining the actor network via MLE, we first pretrain the critic network for 10 epochs, then jointly train the two networks for 40 epochs. Finally, for ensemble variants, the ensemble weight λ in all variants is selected from $0.0 \sim 1.0$ based on its performance on DEV.

5.5 Results

To understand our CoaCor framework, we first show several concrete examples to understand the differences between annotations generated by our model and by baseline/variant models, and then focus on two research questions (RQs):

- **RQ1 (CR improves CA)**: Is the proposed retrieval reward-driven CA model capable of generating rich code annotations that can be used for code retrieval (i.e., can represent the code snippet and distinguish it from others)?
- **RQ2 (CA improves CR)**: Can the generated annotations further improve existing QC-based code retrieval models?

5.5.1 Qualitative Analysis. Table 2 presents two examples of annotations generated by each CA model. Note that we do not target at human language-like annotations; rather, we focus on annotations that can describe/capture the functionality of a code snippet. In comparison with baseline CA models, our proposed RL^{MRR}-based CA model can produce more concrete and precise descriptions for corresponding code snippets. As shown in Example 1, the annotation generated by RL^{MRR} covers more conceptual keywords semantically aligned with the three NL queries (e.g., "average", "difference", "group"), while the baseline CODE-NN and the variants generate short descriptions covering a very limited amount of conceptual keywords (e.g., without mentioning the concept "subtracting").

We also notice that our CA model can generate different forms of a stem word (e.g., "average", "avg" in Example 1), partly because the retrieval-based reward tends to make the generated annotation semantically aligned with the code snippet and these diverse forms of words can help strengthen such semantic alignment and benefit the code retrieval task when there are various ways to express user search intent.

5.5.2 Code Retrieval Performance Evaluation. Table 1 shows the code retrieval evaluation results, based on which we discuss **RQ1** and **RQ2** as below:

RQ1: To examine whether or not the code annotations generated by a CA model can represent the corresponding code snippet in the code retrieval task, we analyze its corresponding QN-based CR model, which retrieves relevant code snippets by matching the NL query Q with the code annotation N generated by this CA model. Across all of the four datasets, our proposed QN-RL^{MRR} model, which is based on a retrieval reward-driven CA model, achieves the best results and outperforms other QN-based CR models by a wide margin of around $0.1 \sim 0.2$ absolute MRR. More impressively, its performance is already on a par with the CODE-NN model, which is one of the state-of-the-art models for the code retrieval task, even though it understands a code snippet solely based on its annotation and without looking at the code content. This demonstrates that the code annotation generated by our proposed framework can reflect the semantic meaning of each code snippet more precisely.

To further understand whether or not the retrieval-based reward can serve as a better reward metric than BLEU (in terms of stimulating a CA model to generate useful annotations), we present the BLEU score of each CA model in Table 3.[5] When connecting this table with Table 1, we observe an inverse trend: For the RL^{BLEU} model which is *trained for* a higher BLEU score, although it can improve the MLE-based CA model by more than 2% absolute BLEU on three sets, it harms the latter's ability on producing useful code annotations (as revealed by the performance of QN-RL^{BLEU} in Table 1, which is worse than QN-MLE by around 0.04 absolute MRR on StaQC-val and StaQC-test). In contrast, our proposed RL^{MRR} model, despite getting the lowest BLEU score, is capable of generating annotations useful for the retrieval task. This is mainly because that

[5]BLEU is evaluated with the script provided by Iyer et al. [21]: https://github.com/sriniiyer/codenn/blob/master/src/utils/bleu.py.

Table 2: Two examples of code snippets and their annotations generated by different CA models. "Human-provided" refers to (multiple) human-provided NL annotations or queries. Words semantically aligned between the generated and the human-provided annotations are highlighted.

Model	Annotation
Example 1 from EVAL set	
SQL Code	SELECT col3, Format(Avg([col2]-[col1]),"hh:mm:ss") AS TimeDiff FROM Table1 GROUP BY col3;
Human-provided	(1) find the average time in hours , mins and seconds between 2 values and show them in groups of another column (2) group rows of a table and find average difference between them as a formatted date (3) ms access average after subtracting
CODE-NN	how do i get the average of a column in sql?
MLE	how to get average of the average of a column in sql
RL^{BLEU}	how to average in sql query
RL^{MRR}	average avg calculating difference day in access select distinct column value sql group by month mysql format date function?
Example 2 from StaQC-test set	
SQL Code	SELECT Group_concat(DISTINCT(p.products_id)) AS comma_separated, COUNT(DISTINCT p.products_id) AS product_count FROM ...
Human-provided	how to count how many comma separated values in a group_concat
CODE-NN	how do i get the count of distinct rows?
MLE	mysql query to get count of distinct values in a column
RL^{BLEU}	how to count in mysql sql query
RL^{MRR}	group_concat count concatenate distinct comma group mysql concat column in one row rows select multiple columns of same id result

BLEU score calculates surface form overlaps while the retrieval-based reward measures the semantically aligned correspondences.

These observations imply an interesting conclusion: *Compared with BLEU, a (task-oriented) semantic measuring reward, such as our retrieval-based MRR score, can better stimulate the model to produce detailed and useful generations.* This is in line with the recent discussions on whether the automatic BLEU score is an appropriate evaluation metric for generation tasks or not [30, 40]. In our work, we study the potential to use the performance of a relevant model to guide the learning of the target model, which can be generalized to many other scenarios, e.g., conversation generation [27], machine translation [4, 44], etc.

RQ2: We first inspect whether the generated code annotations can assist the base code retrieval model (i.e., DCS) or not by comparing several ensemble CR variants. It is shown that, by simply combining the matching scores from QN-RL^{MRR} and DCS with a weighting factor, our proposed model is able to significantly outperform the DCS model by 0.01 ~ 0.03 and the CODE-NN baseline by 0.03 ~ 0.06 consistently across all datasets, showing the advantage of utilizing code annotations for code retrieval. Particularly, the best performance is achieved when the ensemble weight $\lambda = 0.4$ (i.e., 0.4 weight on the QN-based CR score and 0.6 on the QC-based CR score), meaning that the model relies heavily on the code annotation to achieve better performance.

Table 3: The BLEU score of each code annotation model.

Model	DEV	EVAL	StaQC-val	StaQC-test
CODE-NN [21]	17.43	16.73	8.89	8.96
MLE	18.99	19.87	10.52	10.55
RL^{BLEU}	21.12	18.52	12.72	12.78
RL^{MRR}	8.09	8.52	5.56	5.60

In contrast, QN-CodeNN, QN-MLE and QN-RL^{BLEU} can hardly improve the base DCS model, and their best performances are all achieved when the ensemble weight $\lambda = 0.0 \sim 0.2$, indicating little help from annotations generated by CODE-NN, MLE-based and BLEU-rewarded CA. This is consistent with our conclusions to RQ1.

We also investigate the benefit of our generated annotations to other code retrieval models (besides DCS) by examining a baseline "QN-RL^{MRR} + CODE-NN", which combines QN-RL^{MRR} and CODE-NN (as a QC-based CR model) to score a code snippet candidate. As mentioned in Section 5.3, CODE-NN scores a code snippet by the likelihood of generating the given NL query when taking this code snippet as the input. Since the score is in a different range from the cosine similarity given by QN-RL^{MRR}, we first rescale it by taking its log value and dividing it by the largest absolute log score among all code candidates. The rescaled score is then combined with the cosine similarity score from QN-RL^{MRR} following Eqn. (14). The result is shown in the last row of Table 1. It is very impressive that, with the help of QN-RL^{MRR}, the CODE-NN model can be improved by ≥ 0.05 absolute MRR value across all test sets.

In summary, through extensive experiments, we show that our proposed framework can generate code annotations that are much more useful for building effective code retrieval models, in comparison with existing CA models or those trained by MLE or BLEU-based RL. Additionally, the generated code annotations can further improve the retrieval performance, when combined with existing CR models like DCS and CODE-NN.

6 DISCUSSION

In this work, we propose a novel perspective of using a relevant downstream task (i.e., code retrieval) to guide the learning of a target task (i.e., code annotation), illustrating a novel machine-machine collaboration paradigm. It is shown that the annotations generated by the RL^{MRR} CA model (trained with rewards from the DCS model) can boost the performance of the CODE-NN model, which was not involved in any stage of the training process. It is interesting to explore more about machine-machine collaboration mechanisms, where multiple models for either the same task or relevant tasks can be utilized in tandem to provide different views or effective rewards to improve the final performance.

In terms of training, we also experimented with directly using a QN-based CR model or an ensemble CR model for rewarding the

CA model. However, these approaches do not work well, since we do not have a rich set of QN pairs as training data in the beginning. Collecting paraphrases of queries to form QN pairs is non-trivial, which we leave to the future.

Finally, our CoaCor framework is applicable to other programming languages, such as Python and C#, extension to which is interesting to study as future work.

7 RELATED WORK

Code Retrieval. As introduced in Section 1, code retrieval has been studied widely with information retrieval methods [14, 17, 23, 32, 56] and recent deep learning models [3, 13, 21]. Particularly, Keivanloo et al. [23] extracted abstract programming patterns and their associated NL keywords from code snippets in a code base, with which a given NL query can be projected to a set of associated programming patterns facilitating code content-based search. Similarly, Vinayakarao et al. [56] built an entity discovery system to mine NL phrases and their associated syntactic patterns, based on which they annotated each line of code snippets with NL phrases. Such annotations were utilized to improve NL keyword-based search engines. Different from these work, we construct a neural network-based code annotation model to describe the functionality of an entire code snippet. Our code annotation model is explicitly trained to produce meaningful words that can be used for code search. In our framework, the code retrieval model adopts a similar deep structure as the Deep Code Search model proposed by Gu et al. [13], which projects a NL query and a code snippet into a vector space and measures the cosine similarity between them.

Code Annotation. Code annotation/summarization has drawn a lot of attention in recent years. Earlier works tackled the problem using template-based approaches [37, 49] and topic n-grams models [38] while the recent techniques [2, 19–22, 31] are mostly built upon deep neural networks. Specifically, Sridhara et al. [49] developed a software word usage model to identify action, theme and other arguments from a given code snippet, and generated code comments with templates. Allamanis et al. [2] employed convolution on the input tokens to detect local time-invariant and long-range topical attention features to summarize a code snippet into a short, descriptive function name-like summary. Most related to our work are [19] and [58], which utilized sequence-to-sequence networks with attention over code tokens to generate natural language annotations. They aimed to generate NL annotations as close as possible to human-provided annotations for human readability, and hence adopted the Maximum Likelihood Estimation (MLE) or BLEU score optimization as the objective. However, our goal is to generate code annotations *which can be used for code retrieval*, and therefore we design a retrieval-based reward to drive our training.

Deep Reinforcement Learning for Sequence Generation. Reinforcement learning (RL) [53] has shown great success in various tasks where an agent has to perform multiple actions before obtaining a reward or when the metric to optimize is not differentiable. The sequence generation tasks, such as machine translation [4, 39, 44], image captioning [45], dialogue generation [27] and text summarization [43], have all benefitted from RL to address the *exposure bias* issue [4, 44, 45] and to directly optimize the model towards a certain metric (e.g., BLEU). Particularly, Ranzato et al. [44] were among the first to successfully apply the REINFORCE algorithm [62] to train RNN models for several sequence generation tasks, indicating that directly optimizing the metric used at test time can lead to significantly better models than those trained via MLE. Bahdanau et al. [4] additionally learned a *critic* network to better estimate the return (i.e., future rewards) of taking a certain action under a specific state, and trained the entire generation model via the *Actor-Critic* algorithm [54]. We follow Nguyen et al. [39] and Wan et al. [58] to further introduce an *advantage* function and train the code annotation model via the *Advantage Actor-Critic* algorithm [36], which is helpful for reducing biases from rarely taken actions. However, unlike their work, the reward in our framework is based on the performance on a different yet relevant task (i.e., code retrieval), rather than the BLEU metric.

Machine-Machine Collaboration via Adversarial Training and Dual/Joint Learning. Various kinds of machine-machine collaboration mechanisms have been studied in many scenarios [12, 15, 28, 55, 59]. For example, Goodfellow et al. [12] proposed the Generative Adversarial Nets (GANs) framework, where a generative model generates images to fool a discriminative classifier, and the latter is further improved to distinguish the generated from the real ones. He et al. [15] proposed the dual learning framework and jointly optimized the machine translation from English to French and from French to English. Li et al. [29] trained a paraphrase generator by rewards from a paraphrase evaluator model. In the context of code retrieval and annotation, Chen and Zhou [9] and Iyer et al. [21] showed that their models can be used directly or with slight modification for both tasks, but their training objective only considered one of the two tasks. All these frameworks are not directly applicable to achieve our goal, i.e., training a code annotation model to generate rich NL annotations that can be used for code search.

8 CONCLUSION

This paper explored a novel perspective of generating code annotations *for* code retrieval. To this end, we proposed a reinforcement learning-based framework (named "CoaCor") to maximize a retrieval-based reward. Through comprehensive experiments, we demonstrated that the annotation generated by our framework is more detailed to represent the semantic meaning of a code snippet. Such annotations can also improve the existing code content-based retrieval models significantly. In the future, we will explore other usages of the generated code annotations, as well as generalizing our framework to other tasks such as machine translation.

ACKNOWLEDGMENTS

This research was sponsored in part by the Army Research Office under cooperative agreements W911NF-17-1-0412, NSF Grant IIS1815674, Fujitsu gift grant, and Ohio Supercomputer Center [8]. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notice herein.

REFERENCES

[1] Miltiadis Allamanis, Earl T Barr, Premkumar Devanbu, and Charles Sutton. 2018. A survey of machine learning for big code and naturalness. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 81.

[2] Miltiadis Allamanis, Hao Peng, and Charles Sutton. 2016. A convolutional attention network for extreme summarization of source code. In *International Conference on Machine Learning*. 2091–2100.

[3] Miltos Allamanis, Daniel Tarlow, Andrew Gordon, and Yi Wei. 2015. Bimodal modelling of source code and natural language. In *International Conference on Machine Learning*. 2123–2132.

[4] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086* (2016).

[5] Richard Bellman. 1957. A Markovian decision process. *Journal of Mathematics and Mechanics* (1957), 679–684.

[6] Ted J Biggerstaff, Bharat G Mitbander, and Dallas E Webster. 1994. Program understanding and the concept assignment problem. *Commun. ACM* 37, 5 (1994), 72–82.

[7] Steven Bird and Edward Loper. 2004. NLTK: the natural language toolkit. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*. Association for Computational Linguistics, 31.

[8] Ohio Supercomputer Center. 1987. Ohio Supercomputer Center. http://osc.edu/ark:/19495/f5s1ph73.

[9] Qingying Chen and Minghui Zhou. 2018. A neural framework for retrieval and summarization of source code. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*. ACM, 826–831.

[10] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 1999. Learning to forget: Continual prediction with LSTM. (1999).

[11] Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, Vol. 1. IEEE, 347–352.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[13] Xiaodong Gu, Hongyu Zhang, and Sunghun Kim. 2018. Deep code search. In *Proceedings of the 40th International Conference on Software Engineering*. ACM, 933–944.

[14] Sonia Haiduc, Gabriele Bavota, Andrian Marcus, Rocco Oliveto, Andrea De Lucia, and Tim Menzies. 2013. Automatic query reformulations for text retrieval in software engineering. In *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press, 842–851.

[15] Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tieyan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*. 820–828.

[16] Hua He and Jimmy Lin. 2016. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 937–948.

[17] Emily Hill, Manuel Roldan-Vega, Jerry Alan Fails, and Greg Mallet. 2014. NL-based query refinement and contextualized code search results: A user study. In *Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE), 2014 Software Evolution Week-IEEE Conference on*. IEEE, 34–43.

[18] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[19] Xing Hu, Ge Li, Xin Xia, David Lo, and Zhi Jin. 2018. Deep Code Comment Generation. In *Proceedings of the 2017 26th IEEE/ACM International Conference on Program Comprehension*. ACM.

[20] Xing Hu, Ge Li, Xin Xia, David Lo, Shuai Lu, and Zhi Jin. 2018. Summarizing Source Code with Transferred API Knowledge. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 2269–2275. https://doi.org/10.24963/ijcai.2018/314

[21] Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, and Luke Zettlemoyer. 2016. Summarizing source code using a neural attention model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 2073–2083.

[22] Siyuan Jiang, Ameer Armaly, and Collin McMillan. 2017. Automatically generating commit messages from diffs using neural machine translation. In *Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering*. IEEE Press, 135–146.

[23] Iman Keivanloo, Juergen Rilling, and Ying Zou. 2014. Spotting working code examples. In *Proceedings of the 36th International Conference on Software Engineering*. ACM, 664–675.

[24] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[25] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[26] Wuwei Lan and Wei Xu. 2018. Neural Network Models for Paraphrase Identification, Semantic Textual Similarity, Natural Language Inference, and Question Answering. *arXiv preprint arXiv:1806.04330* (2018).

[27] Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. 2016. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541* (2016).

[28] Yikang Li, Nan Duan, Bolei Zhou, Xiao Chu, Wanli Ouyang, Xiaogang Wang, and Ming Zhou. 2018. Visual question generation as dual task of visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6116–6124.

[29] Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018. Paraphrase Generation with Deep Reinforcement Learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 3865–3878.

[30] Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023* (2016).

[31] Pablo Loyola, Edison Marrese-Taylor, and Yutaka Matsuo. 2017. A neural architecture for generating natural language descriptions from source code changes. *arXiv preprint arXiv:1704.04856* (2017).

[32] Meili Lu, Xiaobing Sun, Shaowei Wang, David Lo, and Yucong Duan. 2015. Query expansion via wordnet for effective code search. In *Software Analysis, Evolution and Reengineering (SANER), 2015 IEEE 22nd International Conference on*. IEEE, 545–549.

[33] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[34] Collin McMillan, Mark Grechanik, Denys Poshyvanyk, Chen Fu, and Qing Xie. 2012. Exemplar: A source code search engine for finding highly relevant applications. *IEEE Transactions on Software Engineering* 38, 5 (2012), 1069–1087.

[35] Larry Medsker and Lakhmi C Jain. 1999. *Recurrent neural networks: design and applications*. CRC press.

[36] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*. 1928–1937.

[37] Laura Moreno, Jairo Aponte, Giriprasad Sridhara, Andrian Marcus, Lori Pollock, and K Vijay-Shanker. 2013. Automatic generation of natural language summaries for java classes. In *Program Comprehension (ICPC), 2013 IEEE 21st International Conference on*. IEEE, 23–32.

[38] Dana Movshovitz-Attias and William W Cohen. 2013. Natural language models for predicting programming comments. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 35–40.

[39] Khanh Nguyen, Hal Daumé III, and Jordan Boyd-Graber. 2017. Reinforcement Learning for Bandit Neural Machine Translation with Simulated Human Feedback. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1464–1474.

[40] Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for nlg. *arXiv preprint arXiv:1707.06875* (2017).

[41] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 311–318.

[42] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. (2017).

[43] Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304* (2017).

[44] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732* (2015).

[45] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *CVPR*, Vol. 1. 3.

[46] Lorenzo Rosasco, Ernesto De Vito, Andrea Caponnetto, Michele Piana, and Alessandro Verri. 2004. Are loss functions all the same? *Neural Computation* 16, 5 (2004), 1063–1076.

[47] Falk Scholer, Hugh E Williams, and Andrew Turpin. 2004. Query association surrogates for web search. *Journal of the American Society for Information Science and Technology* 55, 7 (2004), 637–650.

[48] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[49] Giriprasad Sridhara, Emily Hill, Divya Muppaneni, Lori Pollock, and K Vijay-Shanker. 2010. Towards automatically generating summary comments for java methods. In *Proceedings of the IEEE/ACM international conference on Automated software engineering*. ACM, 43–52.

[50] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[51] Stack Overflow. 2018. Stack Overflow. https://stackoverflow.com/.

[52] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[53] Richard S Sutton and Andrew G Barto. 1998. *Introduction to reinforcement learning*. Vol. 135. MIT press Cambridge.

[54] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*. 1057–1063.

[55] Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027* (2017).

[56] Venkatesh Vinayakarao, Anita Sarma, Rahul Purandare, Shuktika Jain, and Saumya Jain. 2017. ANNE: Improving Source Code Search Using Entity Retrieval Approach. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM '17)*. ACM, New York, NY, USA, 211–220. https://doi.org/10.1145/3018661.3018691

[57] Ellen M Voorhees et al. 1999. The TREC-8 Question Answering Track Report.. In *Trec*, Vol. 99. 77–82.

[58] Yao Wan, Zhou Zhao, Min Yang, Guandong Xu, Haochao Ying, Jian Wu, and Philip S Yu. 2018. Improving automatic source code summarization via deep reinforcement learning. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*. ACM, 397–407.

[59] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 515–524.

[60] Xiaoran Wang, Yifan Peng, and Benwen Zhang. 2018. Comment Generation for Source Code: State of the Art, Challenges and Opportunities. *arXiv preprint arXiv:1802.02971* (2018).

[61] Wikipedia contributors. 2019. Google Image Labeler — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Google_Image_Labeler&oldid=881738511 [Online; accessed 4-February-2019].

[62] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

[63] Ziyu Yao, Daniel S Weld, Wei-Peng Chen, and Huan Sun. 2018. StaQC: A Systematically Mined Question-Code Dataset from Stack Overflow. *arXiv preprint arXiv:1803.09371* (2018).

[64] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329* (2014).

Snapshot-based Loading Acceleration of Web Apps with Nondeterministic JavaScript Execution

JiHwan Yeo
Seoul National University
Seoul, Korea
jhyeo@altair.snu.ac.kr

ChangHyun Shin
Seoul National University
Seoul, Korea
schyun@altair.snu.ac.kr

Soo-Mook Moon
Seoul National University
Seoul, Korea
smoon@snu.ac.kr

ABSTRACT

JavaScript execution is heavily used during the loading of web apps, taking a substantial portion of the app loading time. To accelerate JavaScript execution, *snapshot-based app loading* has been proposed [5, 17]. We take a *snapshot* of the JavaScript objects in the heap at some point during app loading (which we call *snapshot point*) and save them in a file in advance. Then, we start app loading by copying the objects in the snapshot to the heap directly, skipping JavaScript execution to create those objects. One issue is that the JavaScript execution state at the snapshot point should be the same at every app loading. If JavaScript execution included in the snapshot is involved with some nondeterminism (e.g., use random function or current time/location), snapshot-based app loading might be inapplicable since the loaded state might differ each time.

In this paper, we perform an empirical study for the nondeterministic behavior of web apps during app loading. The result shows that nondeterminism is used frequently, but its impact is small from the user's perspectives. Based on this observation, we propose two techniques that can increase the applicability of snapshot. First, we propose two types of snapshot point, which allows saving as much execution state as possible in the snapshot, just before the nondeterministic execution affects the user's perception. Second, we develop a tool to identify those snapshot points by static and dynamic analysis, so that the developer can easily identify them. Our experimental results show that the techniques can accelerate the app loading by 1.81 times, by applying the snapshot that would otherwise be inapplicable, achieving a performance comparable to that of a snapshot that completely ignores the nondeterminism issues.

CCS CONCEPTS

• **Software and its engineering → Dynamic analysis**; *Runtime environments*; *Scripting languages*.

KEYWORDS

Web app; JavaScript engine; Snapshot; Nondeterminism

ACM Reference Format:
JiHwan Yeo, ChangHyun Shin, and Soo-Mook Moon. 2019. Snapshot-based Loading Acceleration of Web Apps with Nondeterministic JavaScript Execution. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313575

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313575

1 INTRODUCTION

Web apps are popular because they are extremely portable, running on any device with a web browser. They are standard apps for commercial web platforms such as Tizen [11] or WebOS [8], being employed by embedded devices such as smart TVs, smart watches, or IoT devices.

In the mobile environment, loading time is an important factor affecting the user experience and business revenues [3, 4, 25]. For example, it is found that every 100ms of latency costs 1% reduction of sales for Amazon [10]. Reducing the loading time is also important for web apps, especially in the embedded devices such as smart TVs, since their limited hardware resources might lead to a long loading time, sometimes unacceptable to the users.

One technique to accelerate app loading is *snapshot* [17]. Since app loading usually produces the same execution state, especially the same JavaScript state, we can save the loaded state in a file called the snapshot in advance and perform app loading by copying the state from the file, instead of executing the JavaScript code to produce the same state. Recently, Google's Chrome browser introduced a technique called the *custom snapshot* [5], extending its previous technique *serializer*, which does the same.

Snapshot has a subtle correctness issue. The loading process of a web app may include various *nondeterminisms*. For example, it can executes a JavaScript function *Math.random*, which returns a random value, or a function *Date*, which returns the current time. Also, it can execute Web APIs that retrieve the device status or the browser information. If the results of these nondeterministic executions were also saved in the snapshot, app loading using a snapshot created by a previous run might not produce a correct state for the current run. Existing snapshots include only those JavaScript executions that are deterministic [5, 17]. For this, the app developer must specify a proper snapshot point, up to which only deterministic JavaScript executions are made.

Not all nondeterministic JavaScript executions are unqualified for snapshot, though. Each nondeterminism has a different impact on app loading, and some nondeterminisms do not show any difference from the user's perspectives. For example, the *jQuery* web framework uses *Math.random* to generate a unique identifier and use it as a property name [6]. The property name will change at each loading, but it is used only inside the framework. So, changing the property name does not affect the DOM tree displayed on the screen or the behavior of the event handlers. It is completely safe to include these nondeterministic executions in the snapshot, thus increasing performance without affecting correctness.

Deciding the best snapshot point that excludes unsafe nondeterminism but includes as much JavaScript execution as possible would be a challenge. It is difficult for the app developer to analyze the app to find all variables affected by nondeterministic executions and all affected execution states. If an app uses a JavaScript framework, the developer should fully understand the framework code, which is not easy for most developers. In fact, even the snapshot of the commercial Chrome browser does not give any information on nondeterminism for JavaScript execution [5], so the developers have to analyze by themselves or go through a trial and error method to choose a proper snapshot point.

This paper made three contributions. First, we perform an empirical study of the nondeterministic behavior of web apps at loading time. The result shows that nondeterminism is frequent, but its impact is often little from the user's perspectives, as the unique identifier case of jQuery.

Secondly, we propose two nondeterminism-aware snapshot points that can lead to a higher performance than the snapshot with deterministic executions only. They allow the developers to choose between performance and correctness. The first snapshot point is *deterministic DOM*, which includes the JavaScript executions up to the point where the DOM tree is deterministic. This will include the JavaScript execution that does not affect the user's perception. The second snapshot point is *semi-randomized DOM*, which additionally saves the JavaScript executions that produce a nondeterministic DOM tree with *Math.random*. One example is a game app showing a random initial screen when loaded. If we make a snapshot including *Math.random*, the app will show the same screen at each loading. This does not affect the correctness, but only the user's perception (e.g., a minesweeper app shows the same position of mines at each loading). To handle this issue, we create multiple snapshots and use one of them randomly at loading time to mimic a randomized execution.

Finally, we create a tool to find the snapshot point for web apps using static and dynamic analysis to increase the applicability of the snapshot. We instrument HTML files and JavaScript code, and the browser runs the instrumented app to collect information such as JavaScript variable read/write and DOM tree changes. The tool creates a JavaScript dependency graph based on the information to find all JavaScript statements affected by nondeterminism. If such a JavaScript statement changes the DOM tree, it is regarded as a nondeterministic DOM change. Our tool identifies all script tags or event handlers that make nondeterministic DOM changes so as to help the app developer to decide a proper nondeterminism-aware snapshot point.

Previous studies on nondeterminism classified the source of nondeterminism in the web pages, and made the execution deterministic by saving the execution results of nondeterminism and using them in subsequent executions [2, 14]. Snapshot also saves the results of nondeterministic executions and reuse them for acceleration, but we identify nondeterminisms that affect little the user's perception and save only those executions in the snapshot.

Our experimental results show that the proposed techniques can accelerate the app loading by 1.81 times on average compared to a snapshot with deterministic executions only, by including safe, nondeterministic JavaScript executions in the snapshot that would otherwise be not includable. This achieves a performance comparable to that of a snapshot that completely ignores the nondeterminism issues.

The rest of the paper is as follows. Section 2 reviews snapshot-based app loading. Section 3 analyzes the nondeterminism sources in web apps. Section 4 proposes nondeterminism-aware snapshot points. Section 5 shows the nondeterminism analysis tool. Section 6 depicts the implementation and Section 7 shows the evaluation results. Related work is in Section 8 and conclusion is in Section 9.

2 SNAPSHOT-BASED ACCELERATION OF WEB APPS

The browser loads a web app by executing its HTML code, embedded with the script tags of JavaScript global code. It parses the HTML code to a DOM tree. When a script tag is met, the JavaScript code is executed after creating a *window* object in the heap. After HTML parsing completes, the browser fires *DOMContentLoaded* event and *onload* event, and the JavaScript event handlers registered in the script tags will be executed. This completes the app loading process, and then the app will work in an event-driven manner based on the user events.

Snapshot-based acceleration is based on an observation that app loading often repeats the same initializations. For example, most web apps employ JavaScript frameworks such as jQuery [6], which are initialized during app loading. Similarly, the app itself is initialized as well. These initializations are done by executing the same JavaScript code, which often produces the same objects in the JavaScript heap. So, instead of executing the JavaScript code to create those objects at each loading time, it is better to save the JavaScript objects right after these initializations to a file in advance, and to start an app by copying the JavaScript objects from the file to the heap directly [5, 17]. Even if we need to save or restore the objects one-by-one, starting from the JavaScript *window* object, snapshot-based acceleration is shown to be around four times faster than executing the JavaScript code [17].

Actually, the loading state of a web app includes the DOM state as well as the JavaScript state. Unlike the JavaScript state which can be saved as above, saving the DOM state is not straightforward. For the initial snapshot works [5, 17], the DOM tree is not saved in the snapshot, but completely re-built using the usual HTML parsing. One issue is that if some JavaScript code executed during app loading changes the DOM tree (e.g., add/delete some DOM nodes), such changes will be lost when employing snapshot-based app loading since the snapshot includes only the JavaScript objects. This constrains the snapshot point, such that only the JavaScript executions that do not change the DOM tree are eligible for inclusion in the snapshot [5, 17]. More precisely, since we can have only a single snapshot file, we can save a "consecutive" sequence of JavaScript executions until the first execution that changes the DOM tree is met, which limits the coverage.

More recent snapshots save some of the DOM state as well [16, 29]. One method records the log of all the DOM APIs that JavaScript executions invoke until the snapshot point [16]. Snapshot-based app loading re-builds the DOM tree using HTML parsing as previously, but restores the correct DOM tree state by replaying the DOM logs, while restoring the JavaScript state from the snapshot at the snapshot point. Another method is saving the whole DOM tree as

a serialized HTML string using the *innerHTML* API at the snapshot point [29]. Snapshot-based app loading re-builds the DOM tree, but at the snapshot point, it is overwritten by the de-serialized tree from the HTML string. For the implementation of nondeterminism-aware snapshot points in this paper, we employed the first implementation.

One advantage of saving the DOM state in the snapshot is that we can take a snapshot at a later point of app loading, even after the entire app loading process completes [16], thus increasing the applicability of snapshot. Generally, JavaScript executions during app loading comes from the framework script tags, the app script tags, and the event handlers, in this order. For those snapshots that cannot save the DOM state [5, 17], a snapshot can be taken just before the first DOM-changing JavaScript (actually, the first DOM-changing script tag since they cannot take a snapshot at the event handler). If we save the DOM state, snapshot-based app loading can skip the executions of all script tags and event handlers up to the snapshot point, irrespective of whether they change the DOM tree or not, then restore the JavaScript state from the snapshot, thus performing better.

One issue is that those JavaScript executions included in the snapshot should be deterministic, meaning that they should produce exactly the same execution state every time. If they are nondeterministic, the state stored in the snapshot might be different from the actual app loading state. Previous snapshots did not consider nondeterminism [5, 17], or even if they do, they let the app developer to decide the snapshot point manually [16, 29]. According to our study about nondeterminism in Section 3, however, there occur lots of nondeterminisms during app loading, so we need a better way to apply snapshot considering nondeterminism.

3 NONDETERMINISM IN LOADING OF WEB APPS

In this section, we examine nondeterminisms during app loading and list up the sources of nondeterminism. We also analyze how app loading exploits nondeterminism and examine the impact on the DOM tree.

3.1 Sources of Nondeterminism in App Loading

There are previous researches that analyze the source of nondeterminism that can occur in web apps [2, 14]. Unlike them, we are only interested in nondeterminism that may occur during app loading. For example, many events during app execution are involved with nondeterminism, but only the timer event is source of nondeterminism in the app loading process (see Section 3.1.3).

We have classified the sources of nondeterminism as *nondeterministic functions*, *web APIs*, and *interrupts*. Among the JavaScript functions, those declared in the HTML5 specification are categorized as *web APIs*, and the remaining functions are classified as *nondeterministic functions*. *Interrupts* mean the nondeterminism caused by events such as timer events. Details are discussed below.

3.1.1 Nondeterministic Functions. Among the functions in the JavaScript specification, there are two nondeterministic functions: *Math.random*, which returns a random value, and *Date*, which can be used to return the current time, thus being nondeterministic.

3.1.2 Web APIs. JavaScript supports a variety of APIs for browsers. The most commonly used APIs are related to the DOM tree, allowing the developers to change the display by adding or removing elements from the DOM tree. DOM is standardized and supported by all browsers, but latest APIs are supported by a few browsers. In this paper, we analyze only widely-used APIs, supported by the mainstream browsers. We have classified web APIs into three types.

File system is a set of APIs used to read and write persistent data in the browser, such as cookie, localStorage, or IndexedDB. We use the APIs to read preferences or data, thus being nondeterministic depending on their existence.

Device information is the APIs that fetch the status of the current device. They allow the developers to get device information from the web apps. Among many APIs, we consider geolocation, device orientation, and device motion APIs that are supported by most web browsers. The geolocation API uses GPS or IP address to determine the user's current location, and it is used by apps such as maps and weather to show information based on the location.

Browser Information is an API that retrieves the information of the browser running the app. Developers can get the size/position of the browser using those APIs. The userAgent API is widely used to get the type of the browser and its version. For snapshot implementation, the same browser is used, so we exclude userAgent from our analysis.

3.1.3 Interrupts. Previous researches have analyzed the events in detail [2, 14], but the events that can occur during the app loading process are limited. Generally, events can be triggered by a user interaction, by a JavaScript function call, or by the web browser. In the middle of app loading, the DOM element is not displayed on the screen because the rendering is not completed yet. Since it is difficult for the user to activate an event during loading, the event triggered by the user cannot be the source of nondeterminism.

JavaScript execution can fire an event directly by using the *dispatchEvent* function or by registering a timer event with the *setTimeout*, *setInterval* function. If *dispatchEvent* is used in the JavaScript code, the execution order is fixed because the event occurs immediately after the execution of the code, thus being deterministic. On the other hand, timer events can be nondeterministic. *setTimeout* and *setInterval* are used to register timer events which will be fired after a certain amount of time. Consider two consecutive script tags where the first tag registers a timer event that occurs 2.0 seconds later while the second tag registers a timer event that occurs 1.0 seconds later. If the first tag executes for 0.5 seconds, the first event will fire at 2.0 second while the second one will fire at 1.5 (0.5+1.0) second. If the first tag executes for 1.5 seconds due to some delay (e.g., page miss), however, the first event will still fire at 2.0 second, but the second one will fire at 2.5 (1.5+1.0) second. So, the execution order of the events can be nondeterministic.

Only two events are fired by the browser during app loading: *DOMContentLoaded* event and *onload* event. The *DOMContentLoaded* fires when the HTML parsing is done, and the *onload* event fires when the resources in HTML are completely downloaded. In most apps, the event handlers executed for each of these two events and their execution order are fixed, thus being deterministic.

Table 1: Number of nondeterminism occurrences during app loading

Type	Nondeterministic Functions		File System APIs		Browser Information	Interrupts		Total
	Random	Date	Cookie	localStorage	Size	setTimeout	setInterval	
Framework global code	0/6	0/12	0/0	0/0	0/0	0/0	0/0	0/18
App global code	251/251	1/1	0/0	0/0	9/9	0/1	0/0	261/262
Event handler	582/583	57/64	3/3	3/3	0/0	0/2	0/3	645/658

3.2 Nondeterminism Usage at Web App Loading

We used a dynamic analysis tool *Jalangi* [21] to record all nondeterminisms that occur during app loading. We observe every *function call* and *property access* during app loading, where nondeterministic result might occur (e.g., *Math.random()* or localStorage['setup']). We identify the location and the number of occurrences for nondeterminism defined in Section 3.1. To examine the impact of each nondeterminism on the user's display, we manually analyze the effect on the DOM tree, which is what the user see.

For this study, we selected 9 web apps that make extensive use of JavaScript. We selected these apps from the JavaScript projects on GitHub, and sample apps from the official framework site. Six of them use *jQuery* framework [6] and the rest use *enyo* framework [7]. jQuery is widely used by 73.3% of all websites and enyo is used on the commercial WebOS platform [26].

Each source of nondeterminism has a different impact on app loading. Some nondeterministic value is assigned to a local variable, thus not affecting the global state such as the DOM tree, while others can be used to affect the DOM tree. A previous study has classified nondeterminism based on whether it affects the global state [22]. For example, a nondeterministic JavaScript execution creating a unique identifier affects the global state, but it does not affect the user's display. We also categorized nondeterminism based on if it affects the DOM tree, thus changing the user's display and making the user feel the difference. For *nondeterministic functions* and *web APIs*, we analyzed if the DOM tree changes when the return value changes. For *interrupts*, we checked whether the DOM tree changes when the execution order of event handlers is changed. We also categorized nondeterminism based on where it occurs: *framework global code* in the framework script tag, *app global code* in the app script tag, or *event handlers* of the *DOMContentLoaded* and *onload* events.

Table 1 shows the total number of nondeterminism occurrences during the loading of 9 apps. The left value of each cell is the number of nondeterminisms that affects the DOM tree, while the right value is the total number of occurrences. Some nondeterminisms such as *IndexedDB* and *browser position* are not used, so they are not shown in the table. In the app loading process, nondeterminism occur 938 times in total, and the average number of usages per app is 104. If we categorized nondeterminism based on where it occurs, we can see that the framework global code, app global code, and event handlers use nondeterminism 18 times, 262 times, and 658 times, respectively. Framework global code creates objects and registers event handlers to initialize the framework, and those

(a) Unique identifier generation

```
jQuery.expando =
  "jQuery" + (version+Math.random()).replace(/\D/g, "");
Sizzle.expando = "sizzle" +-(new Date());
```

(b) Timestamp creation

```
jQuery.now = function() {
  return +( new Date() );
};
this.timeStamp = src && src.timeStamp || jQuery.now();
```

Figure 1: jQuery example code with nondeterminism sources

tasks are mostly deterministic. App global code and event handlers heavily use nondeterminism to change the DOM tree by reading the nondeterministic data such as the setting of the app or the current time. Since many jQuery apps uses event handlers to change the DOM tree [13], there are many occurrences of nondeterminism in the event handlers.

Among the sources of nondeterminism, the most frequent calls were *nondeterministic function*, among which *Math.random* is dominant. Since apps often use *Math.random* to determine the location and type of many DOM elements, so it is called frequently.

We examined the effect of nondeterminism on the DOM tree and found that none of the 34 nondeterminisms in the framework global codes affect the DOM tree. The reason is that the framework code uses nondeterminism in a way of not affecting the DOM tree. For example, the jQuery framework global code calls *Math.random* once and *Date* twice, as illustrated in Figure 1. Figure 1 (a) uses *Math.random* and *Date* to create a unique identifier, which is often used for avoiding conflicts between variable names when using multiple versions of jQuery in an app. Figure 1 (b) uses *Date* to record a timestamp, often for an event object to tell when the event is generated. This is useful for profiling the event performance, by measuring the time taken between two events generated as a result of the user interaction.

Since the unit of JavaScript execution included in the snapshot is a script tag or an event handler, we categorized nondeterminisms as a unit of script tags and event handlers. Table 2 shows the number of script tags and event handlers that have nondeterminism during the loading of 9 apps. In the framework script tag, 6 out of 11 are nondeterministic, but none of them affect the DOM tree. In the app script tag, only 4 out of 12 have nondeterminism, but all of them affect the DOM tree. Most of the event handlers (7 out of 9) have

Table 2: Number of script tags and event handlers

Type	Nondeterministic & affecting DOM	Nondeterministic	Total
Framework tag	0	6	11
App tag	4	4	12
Event handlers	6	7	9

nondeterminism, and most (6 out of 7) affect the DOM tree. We also found that nondeterminism is used extensively for some script tags and event handlers, but it is not used at all in others. For example, only two game apps (breakdom and piratepig) call *Math.random* 781 times. Those apps have more than a hundred DOM elements and use *Math.random* to specify the type of DOM elements or position of DOM elements randomly.

In summary, nondeterminism occurs frequently during app loading, but there are many scrip tags and some event handlers that are deterministic, or are nondeterministic yet not affecting the DOM tree. Based on this observation, we propose two nondeterminism-aware snapshot points that minimize the impact on the user's display while achieving a high performance.

4 NONDETERMINISM-AWARE SNAPSHOT POINTS

JavaScript execution order during app loading is fixed, composed of the executions of the framework script tags, app script tags, event handlers for the *DOMContentLoaded* event, and the event handlers for the *onload* event. We will decide a snapshot point at the end of some script in this sequence of scripts, where we create and restore snapshot.

If we consider nondeterminism when we choose a snapshot point, there can be two extremes. One is ignoring the effect of nondeterminism. Since we can save the DOM state in the snapshot [16, 29], this allows us to save all JavaScript executions in the snapshot, thus taking a snapshot point at the end of app loading. We define this snapshot point *nondeterministic JS*, which saves all JavaScript executions without considering nondeterminism.

The other extreme of snapshot is including deterministic JavaScript executions only, which we define *deterministic JS*. This method is consistent with the basic idea of a snapshot that stores and reuses the exactly same result of repeated JavaScript execution. However, as shown in Table 2, 17 out of 32 script tags and event handlers use nondeterminism, and more notably, 54% of the framework script tags have nondeterminism. Since the framework script tags tend to precede other script tags (i.e., the app global code uses the functions defined in the framework global code), if we cannot include some framework script in the snapshot due to nondeterminism, we cannot include any of other subsequent script tags and event handlers, either. This would take a snapshot point early in the app loading process, reducing the coverage of the snapshot. As such, the performance improvement using a snapshot of *deterministic JS* would be seriously limited.

To compromise between performance and correctness that the user perceives, we propose two more snapshot points considering nondeterminism. The first one is *deterministic DOM*, which saves all (even nondeterministic) JavaScript executions as long as they make the DOM tree deterministic. This includes creating unique identifiers using *Math.random* and logging the timestamps in Figure 1. From the user's perspectives, an execution is deemed correct if the result it produces is acceptable to the user [9]. In the web environment, the user observes only the DOM tree and the behavior of the event handlers, so the user will think that snapshot-based app loading is correct if those two states are deterministic. Usually, nondeterminism that affects the event handlers also changes the DOM tree, so we consider only the DOM tree in this paper (we discuss the event handler issue later in Section 5.3). Using a snapshot point of *deterministic DOM*, we can get additional performance by including more JavaScript executions than *deterministic JS* without affecting the user's display.

The second snapshot point is *semi-randomized DOM*, which extends *deterministic DOM* by saving additional JavaScript executions that may change the DOM tree using *Math.random*. Unlike *deterministic DOM*, the DOM tree of snapshot-based app loading can be different from that of regular app loading, but the correctness is not actually compromised. That is, if an app uses *Math.random*, any return value between 0 and 1 is correct, and the snapshot would include one of the possible, correct values (this is not true if a function retrieves the time or the location, since there should be only one correct value). For example, a game app is supposed to show a random, initial screen on each app loading (e.g., a minesweeper game with different mine locations generated by *Math.random*). If such a JavaScript execution is included in the snapshot, the initial screen would be fixed (e.g., the same location of mines), which might look odd, but still correct. *Semi-randomized DOM* exploits this randomness and save the executions that randomly change the DOM tree in the snapshot. Users might feel that the execution is incorrect if the state is fixed in one state, so we created multiple snapshot files generated by multiple app loadings, and use one of them randomly. If we store enough snapshots and use them, users might not feel any difference. This snapshot point is effective for our game apps that change the DOM tree each time of app loading using lots of randomness. For example, breakdom and piratepig app uses *Math.random* 212 and 560 times, respectively. *Semi-randomized DOM* saves those JavaScript executions in the snapshot and archives a substantial performance improvement. This approach leads to space overhead, but users can get a significant performance gain, as will be seen in the experiment.

5 NONDETERMINISM ANALYSIS TOOL

Previous section proposed two nondeterminism-aware snapshot points in the app loading process. Unfortunately, it is difficult for the developers to identify them since they must know every source of the nondeterminism in the app code as well as JavaScript variables and DOM tree elements affected by nondeterminism. In fact, most apps use one or more JavaScript frameworks and it is almost impossible for the developers to understand them completely since the frameworks are substantial with many features. To handle this problem, we propose an analysis tool, which informs the snapshot point automatically by analyzing nondeterminisms that occur during app loading using static and dynamic analysis.



Figure 2: Overview of our nondeterminism analysis tool

5.1 Overview of the Tool

Figure 2 overviews our nondeterminism analysis tool, which uses three steps to identify the snapshot points. First, the tool statically instruments the HTML codes and the JavaScript codes to extract the logs of read/writes for the JavaScript variables and of the DOM mutations in app loading. It also records the sources of nondeterminism. Secondly, we perform app loading with the instrumented app, dynamically collecting the logs, and our instrumented event handler invoked at the end of app loading uses those logs to generate a dependency graph of JavaScript and DOM. We actually run the instrumented app multiple times and get multiple dependency graphs. Since nondeterminism can be used at the conditional statement, the control flow of the app can change, producing a different dependency graph. So we run the app multiple times to deal with diverse control flows. Third, the tool uses the dependency graph to analyze if a DOM change during app loading is affected by a source of nondeterminism. Based on the analysis, it finds which script tags and event handlers can be included for the two proposed snapshot points. Then, the app developer will decide a proper snapshot point considering both their correctness policy and the expected performance benefit.

5.2 Nondeterminism Analysis

Our instrumentor is implemented by extending JS-Slicer [28], which instruments the HTML/JavaScript code to produce a runtime dependency graph for JavaScript and DOM. In the dependency graph, each node represents a statement, and an edge *(i, j)* represents a dependency relationship between node *i* and *j*. JS-Slicer uses static analysis to extract the control dependency and dynamic analysis to identify the JavaScript and DOM dependency.

We modified JS-Slicer to record the usages of nondeterminism based on the source of nondeterminism listed in Section 3.1. We check every *function call* and *property read* at app loading time to see if any source of nondeterminism is used. If nondeterminism is used, the *id* of the statement and the *type* of nondeterminism in the statement (e.g., random) is recorded in the *nondeterminism map*. We use the map information to find the snapshot points later.

After running the app, we analyze if a script (script tag or event handler) in the script sequence of app loading includes nondeterminism, and if so, analyze whether the nondeterminism affects the DOM tree or not. So, for each DOM mutation statement, we compute all statements whose execution result is "*reachable*" to it. If a nondeterministic statement is reachable to a DOM mutation statement, the script that includes the nondeterministic statement "*affects*" the DOM tree. It should be noted that a nondeterministic statement can affect a DOM mutation statement in a later, different script in the script sequence.

Figure 3 shows an example of nondeterminism analysis for a script tag. Figure 3 (a) is a JavaScript code that declares two variables and displays the values in the DOM tree. Figure 3 (b) shows the corresponding dependency graph and the nondeterminism map. We create a node of the dependency graph for each statement, and the number next to each node represents its *id*. Since the return value of node 1 is assigned to the variable *n1* in node 2, node 2 has a dependence on node 1. To analyze whether the DOM changes in the script tag are affected by nondeterminism, we find all the reachable nodes to node 4 which has a DOM tree mutation. In this case, node 1, 2, and 3 are reachable to node 4, so we check the nondeterminism map to see if there are nondeterminism among those three nodes. Since node 1 is in the nondeterminism map, and its type is random, we record that this script tag has nondeterminism that affects the DOM tree, and its nondeterminism type is random.

There can be an if-statement in a script whose execution flow depends on the result of a nondeterministic statement such as *nondeterministic functions*, *device info*, or *interrupts*. In this case, we make an edge from the nodes located in the true and false paths of the if-statement to the nondeterministic statement in the dependency graph, based on the control dependence result obtained during static analysis. This is for computing nondeterministic scripts that may affect the DOM tree conservatively.

Even with a conservative dependency graph, there might be a case that might miss a nondeterministic script that affects the DOM tree. For example, consider a DOM tree element defined before an if-statement that is affected by *Math.random*. If it is redefined only in the true path of the if-statement, *Math.random* would appear not affecting the DOM tree if the false path is taken during app loading with the instrumented app, although it is evident that the DOM tree is changed depending on *Math.random*. To handle this case, we run app loading multiple times to cover diverse control flows, hence diverse dependency graphs.

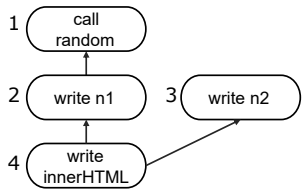
There is a more subtle case. Consider an app whose loading uses a file system API that checks if a file exist, and if so, updates the DOM tree with the file (so the API is nondeterministic). The problem is that the file can be created only by an event handler executed during an user interaction after app loading completes. For a *note* app, for example, if the user writes a new note after current app loading, and the app should show the saved note at the next app loading. Even when we run the instrumented app loading multiple times, if we do not fire the event and execute the event handler after app loading, the API will always return false, thus classified not affecting the DOM tree. So, if an app uses a file system API, we need to collect the information on registered event handlers and fire the user events a few times even after app loading completes.

Based on the above analysis, we identify the scripts to be included in each type of snapshot point. Algorithm 1 shows the algorithm to find a script where we create a snapshot for a given type of snapshot point. *checkCandidate* function analyzes each script

(a) JavaScript code

```
1  var n1 = Math.random();
2  var n2 = 10;
3  document.body.innerHTML = n1 + n2;
```

(b) Dependency graph & Nondeterminism map



Nondeterminism Map

ID	Type
1	random
...	...

Figure 3: Nondeterminism analysis for an example JavaScript code

and tells which candidate snapshot point it can possibly belong to. Since a *deterministic JS* is the strictest candidate, it can also be included in a more relaxed candidate such as *deterministic DOM*, *semi-randomized DOM*, and *nondeterministic JS* snapshot points. Similarly, a *deterministic DOM* candidate can also be included in *semi-randomized DOM* and *nondeterministic JS* snapshot points. Finally, a *semi-randomized DOM* candidate can be included in *nondeterministic JS* as well.

findSnapshotPoint function checks all the scripts to be executed during app loading and returns a script where we need to take a snapshot point for a given type. It checks sequentially for each script in the app loading process if it can be a candidate for a given type of snapshot point based on the result of *checkCandidate*. If so, the script can be a snapshot point but we continue to check if we can extend the snapshot point to the next script (by iterating the loop in Algorithm 1). Otherwise, the script is the first one that cannot be included in the snapshot, so we return the current snapshot point, which is the previous script.

5.3 Limitation

To decide a proper nondeterminism-aware snapshot point, we analyzed if nondeterministic scripts executed during app loading affect the DOM tree. However, they can also affect the behavior of event handlers executed after app loading for some apps. For the nondeterminism analysis tool to cover the behavior of these event handlers, the tool should record a dependency graph for them as well. Unlike the scripts executed during app loading, the execution order of these event handlers would be arbitrary, depending on user interaction. So, we should trigger every sequence of user events and check the event handler is affected by nondeterminism in app loading. It is challenging to test event sequence using dynamic analysis only, so using some static analysis approach for the event handler would be effective [12], which is left as a future work.

For the apps using *file system* APIs, we need to trigger the events for user interaction to explore more paths of app loading scripts, as mentioned in Section 5.2. We currently trigger user events up to a fixed depth of two. App testing time would dramatically increase if we use a higher depth for more precise analysis. In the field of automatic testing, there are researches to minimize the triggered events [1, 23], which we can use for efficient testing in the future.

Algorithm 1 Snapshot Point Detection

```
1:  function checkCandidate(script)
2:      if script has nondeterminism then
3:          if nondeterminism affect DOM tree then
4:              if only random affect DOM tree then
5:                  return "semi-randomized DOM candidate"
6:              else
7:                  return "nondeterministic JS candidate"
8:          else
9:              return "deterministic DOM candidate"
10:     else
11:         return "deterministic JS candidate"
12:
13: function findSnapshotPoint(snapshot_point_type)
14:     snapshot_point ← ∅
15:     for i = 1 to number of scripts do
16:         candidate ← checkCandidate(script_i)
17:         if snapshot_point_type ⊂ candidate then
18:             snapshot_point ← script_i
19:         else
20:             return snapshot_point
21:     return snapshot_point
```

6 IMPLEMENTATION

We extracted dependencies between JavaScript variables on the top of JS-Slicer [28], which performs dynamic slice using JavaScript. For nondeterminism analysis, we update the shadow execution rule of the existing implementation to extract information of nondeterminism. JS-Slicer calculates the dependency for the DOM tree, but only using the *appendChild* and *innerHTML* among the APIs that can modify the DOM tree. The *MutationObserver* API is a Web API that monitors every DOM tree changes [24]. We used the API to track the DOM tree changes after the execution of the *property write* and *function call* statements.

JS-Slicer correctly generated dependency graphs for most JavaScript implementations, but do not for some native functions. JS-Slicer assumes that all arguments have a data dependency on the return value. However, in a native function such as *Array.sort*, we must attach additional edges to the dependency graph. *Array.sort* function compares the elements in an array and sorts them in order. This function compares two values using the function at the first argument. If the function that compares the elements uses nondeterminism, the array is randomly sorted and there must be a dependency between the return value of the function argument and the base array. We analyzed 28 methods of *Array* object. 10 of them update the array after the function call, but JS-Slicer creates no dependency between the argument and the base array. 7 of them need dependency between the return value of the argument and the base array as *Array.sort*. *Array.sort* is included in both case and we add dependencies on those 16 methods.

For automated testing, we use selenium to run and trigger events [20]. We run the app without file system five times and trigger user events up to a fixed depth of two for others. Finally, we create five snapshot files for *semi-randomized DOM* to simulate randomized execution.

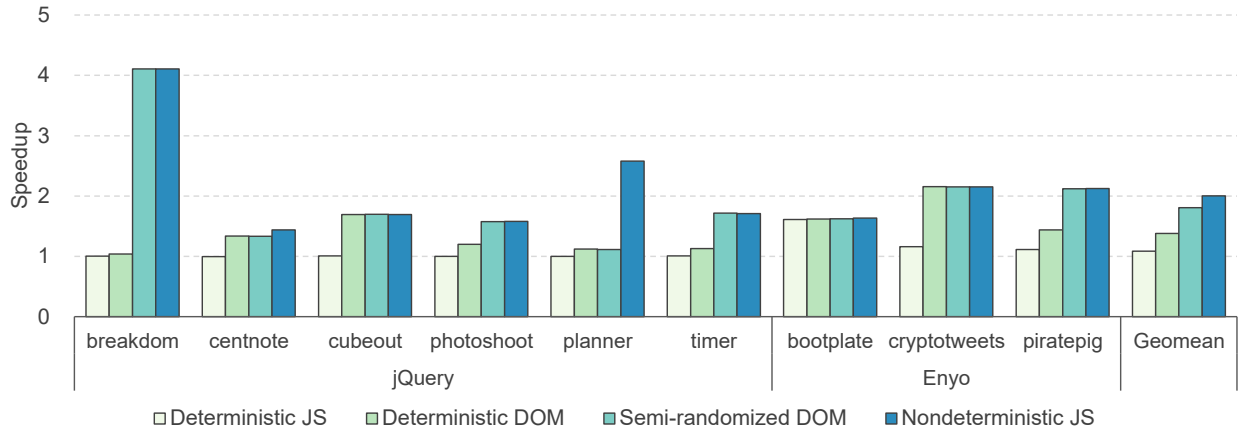


Figure 4: Snapshot-based app loading speedups of four snapshot points compared to original app loading

7 EVALUATION

We measured the performance of app loading based on the four snapshot points obtained from the nondeterminism analysis tool, and examined the effect of nondeterminism in the web app. Our snapshot is implemented based on the WebKit browser (rev-149728) and executed on a Pandaboard ES with 1.2GHz ARM Cortex-A9 CPU and 1GB of RAM. We experimented with 6 jQuery apps and 3 enyo apps used in the analysis in Section 3. We used a nondeterministic analysis tool to find *deterministic JS*, *deterministic DOM*, and *semi-randomized DOM* snapshot points. We also compare with the *nondeterministic JS* snapshot point, which saves the result of the whole JavaScript executions during app loading in the snapshot.

7.1 Snapshot Point Performance

Figure 4 shows the speedup of app loading time for the four snapshot points, compared to the app loading time without the snapshot. *Deterministic JS* accelerate app loading by 1.09 times on average. Web apps in our experiment use the JavaScript framework (jQuery or Enyo). jQuery framework calls the sources of nondeterminism during app loading, so we could not even save the execution state of the first script. On the other hand, Enyo framework does not call sources of nondeterminism source, so execution state of some global codes is included in the snapshot.

Deterministic DOM accelerate app loading by 1.38 times on average. It saves most execution results of framework and app script tags (global code) in the snapshot. Most enyo apps and some jQuery apps show the same performance improvement as *nondeterministic JS*, which ignores nondeterminism and saves the result of all event handlers.

Semi-randomized DOM accelerate app loading by 1.81 times on average. Although it still considers nondeterminism, it shows a performance improvement similar to *nondeterministic JS*, which accelerate app loading by 2.00 times. In most apps, *deterministic DOM* and *semi-randomized DOM* have the snapshot point at the same script, but the breakdom, photoshoot, and piratepig apps, which use lots of randomness, have a difference snapshot point. They use the *Math.random* function to calculate the location or the type of the DOM elements, which take a considerable amount of execution time. *Semi-randomized DOM* could achieve a substantial performance gain by including those executions in the snapshot.

7.2 Size of Nondeterministic Sets

We measured number of statements executed in the app loading. Table 3 shows the result. The first column shows the app names. The second column shows number of executed statements at app loading. The third and fourth columns show number of nondeterminism source statement and statements *reachable* from them. The fifth and sixth column show number of DOM change statement and number of DOM changes *affected* by nondeterminism. We measure LOC of jQuery framework and app code separately. Since the size of the enyo framework varies depending on the module used in the app, we analyzed minimal enyo framework code and both framework and app code together for enyo apps. We collected and averaged all execution results in the test runs.

The results show that many statements are reachable from nondeterminism at app loading. Although the number of nondeterminism sourzce takes less than 1% from executed statements, one source of nondeterminism propagates to average 379 statements. jQuery framework has nondeterminism but does not affect DOM tree with nondeterminism. We checked the statements executed during app loading to ensure that the tool correctly analyzes information on the statement and DOM tree mutations affected by nondeterminism, even in complex apps.

8 RELATED WORK

Wprof [27] classifies dependencies between browser activities and identifies bottleneck activities that contribute to the page load time using critical path analysis. They show that computation is a significant factor that constitute for 35% of the page load time on the critical path and synchronous JavaScript evaluation make a major contribution to page load time since it blocks parsing. Nejati et al. [15] analyzed web page load time for a mobile browser, which has limited hardward and poor network connection. On the mobile

Table 3: Number of executed statements during app loading

Web app	LOC	# exec stmt	# nondeter source	# nondeter affect	# DOM change	# nondeter DOM change
jQuery	10315	4736	4	4	7	0
breakdom	193	274064	217	253195	861	858
centnote	134	5457	5.4	625	7.6	0.6
cubeout	2180	77439	5	7	35	0
photoshoot	182	16384	25	14127	88	81
planner	584	2481	49	1081	113	7
timer	101	24851	7	15199	68	52
Enyo	4283	15153	0	0	0	0
bootplate	8730	36608	0	0	6	0
cryptotweets	8919	56296	29	253	28	0
piratepig	5068	80890	570	59392	14	2
Average	3037	58962	90.7	34388	122.1	100.1

browser, computation activities is the primary bottleneck which occupy more than 60% of the critical path.

There have been previous studies using snapshots to reduce app loading time. V8 JavaScript engine had a feature called *serialization*, which saves the JavaScript heap's built-in object in a file after the V8 initialization [5]. The V8 engine reads the serialized built-in object into the heap at startup, thus reducing the start-up time. Recently, using the APIs provided by V8, developers can also save the JavaScript execution results in the custom snapshot to accelerate the app loading. However, V8 can store only JavaScript state, and does not save the DOM state in the snapshot. There are earlier researches to accelerate app loading using a snapshot for the WebKit browser [16, 17]. Like V8, they traverse the JavaScript heap to save objects as snapshot files. It can also save the state of the DOM tree to extend the scope of the snapshot. The idea is recording all DOM change logs and restoring the DOM tree state by replaying it [16]. Recently, a different approach has been proposed where the DOM state is saved in serialized HTML [29]. It also provides a method of taking a snapshot point at the event handler, while the previous works take a snapshot at the script tag or at the end of app loading only. All these studies did not consider how to apply the snapshot when there is nondeterminism, so the developer had to analyze the app code and apply the snapshot on his own. This paper proposes a tool that can notify a proper snapshot point with two new nondeterminism-aware snapshot points.

Midas considered nondeterminism in middleware replication environments [22]. It has exploited application-level insight to classify nondeterminism as either affecting the application's persistent state or not. They used dynamic analysis to examine the effect of nondeterminism on the app and suggest ways to make the states the same when nondeterminism occurs in several devices. They classify nondeterminism based on the global state in the COBRA environment, and we categorize nondeterminism from the DOM tree in the web environment. We proposed a method to apply snapshot considering nondeterminism and made it easy to use snapshot through the tool.

Mugshot have developed record-and-replay for debugging for web environment and dealing with nondeterminism [14]. Mugshot classify the nondeterminism that can occur in the browser into five categories. *Tardis* also implemented record-and-replay in a managed runtime environment and considered nondeterminism [2]. They considered various nondeterminism such as network I/O, thread switches, etc. that can occur in the managed runtime, but focused on demonstrating the same execution result as the mugshot. We also classify nondeterminism that can occur in the web environment, but our research has focused on how to consider nondeterminism in environments using snapshots. In an environment where a snapshot is used, nondeterminism must be performed each time differently.

There have been various studies analyzing JavaScript. TypeDevil is research to find out when the type of variables changes dynamically through dynamic analysis [19]. They recorded all the information about how the type changed for each variable and analyzed it. ConflictJS is a study that tells you when a conflict occurs using the same variable between JavaScript libraries [18]. They analyzed the global variables that each JavaScript library writes to find conflicts between the libraries. Our study has analyzed nondeterminism occurring in a JavaScript environment differently from other studies. We also provide information about the scripts that can be included in the snapshot that developers can use based on the analysis.

9 CONCLUSION

In this paper, we analyzed JavaScript executions during app loading and found that nondeterminism occurs frequently but seldom affects the user's display. Based on these observations, we proposed two new snapshot points considering nondeterminism. We also developed a tool to inform the snapshot points automatically. We use the snapshot points obtained by the nondeterminism analysis tool for real apps and acclerate app loading by 1.81 times on average, compatitive to that of a snapshot point that completely ignores the nondeterminism issue.

ACKNOWLEDGMENTS

This work was supported by Basic Science Research Program through the National Research Foundation (NRF) of Korea funded by the Ministry of Science, ICT & Future Planning (NRF-2017R1A2B2005562).

REFERENCES

[1] Shay Artzi, Julian Dolby, Simon Holm Jensen, Anders Møller, and Frank Tip. 2011. A Framework for Automated Testing of JavaScript Web Applications. In *Proceeding of the 33rd international conference on Software engineering - ICSE '11*. ACM Press, New York, USA, 571–580. https://doi.org/10.1145/1985793.1985871

[2] Earl T Barr and Mark Marron. 2014. Tardis: Affordable Time-travel Debugging in Managed Runtimes. *ACM SIGPLAN Notices* 49, 10 (2014), 67–82. https://doi.org/10.1145/2660193.2660209

[3] Tammy Everts. 2018. New findings: For top ecommerce sites, mobile web performance is wildly inconsistent. https://blog.radware.com/applicationdelivery/wpo/2014/10/2014-mobile-ecommerce-page-speed-web-performance/

[4] Dennis F Galletta, Raymond M Henry, Scott Mccoy, and Peter Polak. 2002. Web site delays: How tolerant are users? *Information Systems Research* 17, December 2002 (2002), 20–37.

[5] Google. 2018. Custom startup snapshots. https://v8.dev/blog/custom-startup-snapshots

[6] JQuery. 2018. jQuery. https://jquery.com/

[7] LG. 2018. Enyo JavaScript Application Framework. http://enyojs.com/

[8] LG. 2018. webOS. http://webosose.org/

[9] Xuanhua Li and Donald Yeung. 2007. Application-level correctness and its impact on fault tolerance. In *Proceedings - International Symposium on High-Performance Computer Architecture*. 181–192. https://doi.org/10.1109/HPCA.2007.346196

[10] Jim Liddle. 2008. Amazon found every 100ms of latency cost them 1% in sales. https://blog.gigaspaces.com/amazon-found-every-100ms-of-latency-cost-them-1-in-sales/

[11] Linux Foundation. 2012. Tizen. https://www.tizen.org/

[12] Magnus Madsen, Frank Tip, and Ondřej Lhoták. 2015. Static analysis of event-driven Node.js JavaScript applications. *Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications - OOPSLA 2015* (2015), 505–519. https://doi.org/10.1145/2814270.2814272

[13] David Sawyer McFarland. 2011. *JavaScript & jQuery: The Missing Manual*. O'Reilly. 538 pages.

[14] James Mickens, Jeremy Elson, and Jon Howell. 2010. Mugshot : Deterministic Capture and Replay for JavaScript Applications. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*. 159–174.

[15] Javad Nejati and Aruna Balasubramanian. 2016. An In-depth Study of Mobile Browser Performance. In *Proceedings of the 25th International Conference on World Wide Web - WWW '16*. ACM Press, New York, USA, 1305–1315. https://doi.org/10.1145/2872427.2883014

[16] JinSeok Oh. 2016. *Exploiting Snapshot for Web Applications*. PhD diss. Seoul National University. http://s-space.snu.ac.kr/handle/10371/119192

[17] JinSeok Oh and Soo Mook Moon. 2015. Snapshot-based loading-time acceleration for web applications. In *Proceedings of the 2015 IEEE/ACM International Symposium on Code Generation and Optimization, CGO 2015*. IEEE, 179–189. https://doi.org/10.1109/CGO.2015.7054198

[18] Jibesh Patra, Pooja N Dixit, and Michael Pradel. 2018. ConflictJS: Finding and Understanding Conflicts Between JavaScript Libraries. In *Proceedings of the 40th International Conference on Software Engineering*. ACM, 741–751. https://doi.org/10.1145/3180155.3180184

[19] Michael Pradel, Parker Schuh, and Koushik Sen. 2015. TypeDevil: Dynamic type inconsistency analysis for javascript. In *Proceedings of the 37th International Conference on Software Engineering*, Vol. 1. IEEE Press, 314–324. https://doi.org/10.1109/ICSE.2015.51

[20] Selenium. 2018. Selenium, Browser Automation. https://www.seleniumhq.org/

[21] Koushik Sen, Swaroop Kalasapur, Tasneem Brutch, and Simon Gibbs. 2013. Jalangi: a selective record-replay and dynamic analysis framework for JavaScript. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering - ESEC/FSE 2013*. ACM, 488–498. https://doi.org/10.1145/2491411.2491447

[22] Joseph Slember and Priya Narasimhan. 2006. Living with Nondeterminism in Replicated Middleware Applications. In *Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware*. Springer-Verlag New York, Inc., 81–100. http://dx.doi.org/10.1007/11925071_5

[23] Chungha Sung, Markus Kusano, Nishant Sinha, and Chao Wang. 2016. Static DOM event dependency analysis for testing web applications. *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering - FSE 2016* (2016), 447–459. https://doi.org/10.1145/2950290.2950292

[24] Anne van Kesteren. 2015. MutationObserver. https://www.w3.org/TR/dom/#interface-mutationobserver

[25] Ted Vrountas. 2018. How Slow Mobile Page Speeds Are Ruining Your Conversion Rates. https://instapage.com/blog/optimizing-mobile-page-speed

[26] W3Techs. 2018. Usage of JavaScript libraries for websites. https://w3techs.com/technologies/overview/javascript_library/all

[27] Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall. 2013. Demystifying page load performance with WProf. In *Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation - NSDI'13*. 473–485.

[28] Jiabin Ye, Cheng Zhang, Lei Ma, Haibo Yu, and Jianjun Zhao. 2016. Efficient and Precise Dynamic Slicing for Client-Side JavaScript Programs. In *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*. 449–459. https://doi.org/10.1109/SANER.2016.96

[29] JiHwan Yeo, JinSeok Oh, and Soo-Mook Moon. 2019. Accelerating Web Application Loading with Snapshot of Event and DOM Handling. In *Proceedings of the 28th International Conference on Compiler Construction (CC '19)*.

Doppelgängers on the Dark Web: A Large-scale Assessment on Phishing Hidden Web Services

Changhoon Yoon
S2W LAB Inc.

Kwanwoo Kim
KAIST

Yongdae Kim
KAIST

Seungwon Shin
KAIST, S2W LAB Inc.

Sooel Son*
KAIST

ABSTRACT

Anonymous network services on the World Wide Web have emerged as a new web architecture, called the *Dark Web*. The Dark Web has been notorious for harboring cybercriminals abusing anonymity. At the same time, the Dark Web has been a last resort for people who seek freedom of the press as well as avoid censorship. This anonymous nature allows website operators to conceal their identity and thereby leads users to have difficulties in determining the authenticity of websites. Phishers abuse this perplexing authenticity to lure victims; however, only a little is known about the prevalence of phishing attacks on the Dark Web.

We conducted an in-depth measurement study to demystify the prevalent phishing websites on the Dark Web. We analyzed the text content of 28,928 HTTP Tor hidden services hosting 21 million dark webpages and confirmed 901 phishing domains. We also discovered a trend on the Dark Web in which service providers perceive dark web domains as their service brands. This trend exacerbates the risk of phishing for their service users who remember only a partial Tor hidden service address.

Our work facilitates a better understanding of the phishing risks on the Dark Web and encourages further research on establishing an authentic and reliable service on the Dark Web.

CCS CONCEPTS

• **Security and privacy** → **Phishing**; • **Information systems** → *Data extraction and integration*; • **Networks** → *Network privacy and anonymity*.

ACM Reference Format:
Changhoon Yoon, Kwanwoo Kim, Yongdae Kim, Seungwon Shin, and Sooel Son. 2019. Doppelgängers on the Dark Web: A Large-scale Assessment on Phishing Hidden Web Services . In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313551

1 INTRODUCTION

The *Web* is the most popular, worldwide, and accessible platform for sharing and disseminating information across the globe. However, there is not only a bright side of the Web, but there is also a dark side. A set of platforms that host websites whose owners and users remain anonymous is now referred to as the **Dark Web**, whereas the *Surface Web* hosts regular websites. The Dark Web hosts websites whose formats and appearances are the same as those of the Surface Web. However, the way to access the Dark Web is different from that of the Surface Web. It demands the use of an anonymity network service for Dark Web service providers and their visitors to hide their identities on the Web.

The definition of the Dark Web has not officially been established [60], but it is often referred by the popular press and security community to emphasize illicit activities that abuse anonymity networks [15]. In this paper, we use the term "Dark Web" to refer to the collection of hidden Web services built on anonymous networks.

The Dark Web has become a major distribution channel for delivering and advertising malicious content. Silkroad [29] and Hansa-Market [13] are well-known Dark Web marketplaces that sell drugs, illegal weapons, and even malware. In addition, researchers have revealed that the Dark Web contained a considerable amount of harmful content [27, 53], and their findings have been confirmed by government investigative agencies [6] as well.

Conversely, the Dark Web offers a last resort for people who want to avoid censorship, to abide freedom of the press, and even to minimize tracking risks for their privacy. For instance, Venezuela experiencing the recent financial turmoil have blocked accessing political and social content on the Web, thus leaving Tor as the only option to access the restricted content [22].

Motivation. Phishing is one of the most effective threats that harvest users' privacy-sensitive information [64]. There thus exist the previous investigative studies that emphasize the severity of phishing attacks and the prevalence of phishing campaigns on the Surface Web [44, 54, 59, 61, 64]. Thomas et al. assessed the severity of Web phishing campaigns. Their study showed that phishing websites on the Surface Web emulating Gmail, Yahoo, and Hotmail logins had managed to steal 1.4 million credentials [61].

Conversely, the phishing threats on the Dark Web are understudied. Relatively little is known about the prevalence of phishing websites across the Dark Web universe. This trend stems from the absence of the oracle telling whether a given dark website is a phishing site or not. There is sufficient information to obtain the identity of owners operating websites on the Surface Web, which include HTTPS certificates, WHOIS, and DNS records. On the other hand, most dark websites' owners seek to hide their identities, which naturally makes it improbable for users to distinguish an authentic dark website from its phishing websites on the Dark Web.

We argue that it is crucial to address these phishing risks. On the Dark Web, users have no practical way to check the authenticity of

*Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313551

Web services except leveraging out-of-band clues, thus exacerbating the severity of phishing threats. To the best of our knowledge, mo previous study has investigated the phishing risks on the Dark Web. Prior work focused on analyzing illegal contents or unexpected activities [26, 42, 43, 46, 51], uncovering illegal activities [24, 25, 29, 34, 58, 63], and analyzing the popularity of the content on the Dark Web [27, 28].

Contributions. We conducted an in-depth analysis to identify phishing websites on the Dark Web. To do this, we collected more than 21 million webpages from 100K Tor hidden services for seven months. Our dataset represents the most up-to-date and comprehensive characteristics of the Dark Web.

We start with identifying phishing candidates whose website contents are almost identical to other websites. In other words, we investigate how many domains with distinct content and their duplicates constituted the Dark Web. We employ a carefully designed content grouping algorithm that classifies onion domains into content-wise distinct website groups based on the text and the title in each Tor onion domain's homepage. We observe that only 5,718 website groups exhibiting distinct content are available on the Tor network. Interestingly, the content of the top two website groups is duplicated in over 200 domains, which calls into question the authenticity of these domains.

For each website group consisting of multiple domains, we analyze all of the identified duplicates and confirm the presence of abundant phishing websites among the duplicates. Specifically, we identify 791 phishing websites that target five major Dark Web services including dark marketplaces and Bitcoin mixing services. We further analyzed how often users would encounter such phishing domains by counting cross references from other domains. In general, an authentic dark website is more frequently referenced by other dark domains than its phishing websites. However, interestingly, the most referenced website of Dream Market, a popular black market, is a phishing domain, which demonstrates that the attacker diligently spreads phishing domains.

To find further phishing websites on the Dark Web, we leverage "gray website." *Gray website* refers to a website that provides their identical services on both of the Surface and Dark Web. By leveraging the same ownership of a gray website and its corresponding surface website, we find 297 phishing websites on the Dark Web that target gray websites including Facebook. Our study is the first large-scale investigative study that confirms prevalent phishing sites on the Dark Web. Biryukov et al. [27] briefly mentioned the presence of one phishing website that mirrored Silkroad.

We also analyze the common trend in onion domain addresses on the Dark Web. Each website on the Tor network is represented by an onion domain name, a cryptographic string computed from the owner's public key. Considering that even picking an arbitrary five character-long prefix of an onion domain is computationally expensive and the visitors should still type the entire domain, we expect no meaningful prefix on onion names. However, we observe that the majority of the Dark Web service providers intentionally generate them to have meaningful prefixes that are at least five characters long. We conclude that the onion domain name itself is perceived as a brand for the service providers on the Dark Web. This trend has led to dark websites being vulnerable to phishing attacks because victims will depend on a partial memorable prefix instead of its entire onion domains to determine the authenticity of a service.

In this paper, we systematically studied unique/duplicate textual contents based on the comprehensive Tor anonymity network dataset, which is larger than any previous research. We observe abundant duplicate websites that target phishing victims and manifest their characteristics. We also confirm close correlations between website content and onion domains, which exacerbate the phishing risks on the Dark Web. Our work facilitates a better understanding of the phishing websites on the Dark Web and invites further research.

2 RELATED WORK

The *Dark Web* is a privacy-centric web environment built on anonymity networks such as Tor [33]. In the name of anonymity, the Dark Web has been aiding and abetting illegal activities. According to Biryukov et al. [27], most dark websites hosted content devoted to adult films (17%), drugs (15%), counterfeit items (8%), and weapons (4%) in 6,579 HTTP(S) Web services. Moreover, another study [53] has shown that the most popular Tor hidden services were botnet command and control (C&C) servers, and a large proportion of the content on the Dark Web is of questionable legality.

The notoriety of the Dark Web has triggered national authorities to seize many websites that are directly involved in illegal activities or that indirectly help users commit crimes [6]. Therefore, there is a growing interest among the authorities as well as researchers to understand the Dark Web, which demands discovering, classifying, and even deanonymizing dark websites.

Unlike the Surface Web, where domain names are well indexed by search engines, the Dark Web domain names are unknown unless they are explicitly announced via public channels. Biryukov et al., however, have demonstrated that it was possible to reveal all the Tor hidden services domain names by exploiting a design flaw of Tor's hidden services protocol [28]. Specifically, they modified the Tor relays (Onion routers) to sniff out the domain names. Today, the Tor administrators are actively detecting and banning such misbehaving onion routers [38, 56]. In addition, a recent study [45] showed that it is also possible to gather some of the onion domain names using conventional search engines that crawl and index hidden services via Tor proxy services [37].

Deanonymization of Tor hidden services has also been studied in previous work. Øverlier and Syverson [51] were the first to demonstrate the deanonymization of Tor hidden services, and there have been other studies [26, 42, 43, 46, 50] until today. More recently, there have been attempts to automate the analysis of the Dark Web; CARONTE [46] is an automatic Tor hidden service deanonymization system, and ATOL [35] is an automated Tor hidden service categorization framework.

Christin [29] and Soska et al. [58] have performed an extensive analysis of anonymous marketplaces on the Dark Web. Several related studies [24, 34] have also assessed the volumes of these marketplaces. On the customer side of dark markets, Van Hout et al. [63] and Barratt et al. [25] have provided notable insights into the participants of drug marketplaces. Biryukov et al. [27, 28] and Owen et al. [53] have analyzed the content and popularity of the Tor hidden services in general. Recently, Sanchez-Rola et al. [57]

measured the structural connection between the Dark Web and the Surface Web, and they also assessed the usage of tracking scripts on the Dark Web.

Phishing has remained a critical security threat [31, 32, 36, 40, 41, 48, 61, 64]. M. Cova et al. [31] investigated the diverse phishing methods of prevalent phishing kits. X. Han et al. [36] proposed a defense sandboxing technique to protect victims from phishing kits. Thomas et al. [61] used a Google dataset to assess the severity of currently on-going phishing campaigns and found that 12.4 million people are potential phishing victims, and 1.9 billion credentials have been leaked due to previous data breaches.

We conducted a content-based analysis to identify phishing websites on the Dark Web. Due the anonymous nature of the Dark Web, identifying phishing websites itself imposes a unique challenge. To the best of our knowledge, our study is the first investigative study that systematically examines the presence of phishing websites on the Dark Web and their distinctive characteristics.

3 DATA COLLECTION

The Dark Web refers to websites accessible via all the anonymity networks, including Tor [33], I2P [39], and Freenet [30]. Our study targets *hidden Web services*, which are the hidden services accessible via HTTP(S) on the most prevalent anonymity network, Tor.

We initially collected 10K Tor hidden service addresses (or onion domain names) from the popular Tor hidden service search engines and directories: Ahmia [23] and FreshOnion [10]. One of the most effective onion domain collection strategies, which Biryukov et al. [28] introduced, would capture all the onion domains available on the Tor network. They introduced a method of deploying modified onion relays that sniff out hidden service descriptors from anonymity network traffic. However, because the Tor policy now bans such misbehaving onion routers [9], we did not employ this method for ethical and respectable research.

We implemented a Web crawler based on Scrapy [18] and Splash [20] to collect not only static content but also dynamically changing content. To bypass anti-crawler/bot mechanisms that dark websites may employ, we used the latest Tor browser user agent for the crawler and distributed HTTP(S) requests towards the same domain via different Tor circuits to avoid being blacklisted due to sending numerous requests.

We deployed 16 web crawler instances and began by exhaustively exploring each of the 10K seed onion domains to collect all of the Tor HTTP(S) hidden services. We configured the crawlers not to follow any hyperlink to the Surface Web, but only to traverse .onion links until there were no more links to follow. We regularly updated our seed onion addresses by visiting the sources above to maximize our data coverage. Because both Ahmia and FreshOnion provide a full list of their indexed hidden services, the collected seeds were not biased to our selections of search keywords. We only collected textual data to avoid downloading any illegal content.

During the data collection period (7 months), we collected 28,928 unique onion domains with 21 million webpages. Table 1 summarizes our dataset coverage. Notably, of the 28,928 onion domains, there were 13,326 distinct second-level domains (2LDs). Our crawlers were initially deployed with 10K seed onion addresses, and by the end of our data collection period, we found 100K distinct Tor hidden services (or 2LDs). Although we were able to collect 100K distinct 2LDs, in the end of our data collection period, our webpage dataset included the webpages collected from 13,326 2LDs. This discrepancy resulted from the fact that not all Tor hidden services were alive during the data collection period (7 months) and a partial fraction of the Tor hidden services provided their Web services.

Table 1: Dark Web data collection

Collection period	January 2017 ~July 2017
Number of domains	28,928
Number of 2LDs	13,326
Number of webpages	21,537,119

Note that previous research [27] covered 100% of the Tor hidden services (or 39,824 2LDs) in 2013, and they reported that *only 3,741 of them provided web (HTTP) services*. We emphasize that the number of domains and webpages of our dataset demonstrates extensive coverage of the Dark Web.

We use *domain* and *2LD* to refer to an onion domain and a second-level onion domain, respectively. The *2LD* of an onion domain is an 80-bit number in base32 encoding and generated from a public key, which represents a unique client who owns the corresponding private key. For a domain not on the Tor network, we explicitly use the term:*surface domain*. The *homepage* of a given domain is the final destination page when visiting the domain with a browser.

4 IDENTIFYING CONTENT DUPLICATES

A phishing adversary lures victims to her website of which content is almost identical to the one of a legitimate website, thus expecting victims to provide sensitive information. Since it is infeasible to vet the authenticity of websites on the Dark Web by its anonymous nature, we exploit this intrinsic characteristic to identify phishing website candidates.

To accurately compute phishing candidates, we cluster a large number of domains into one website group when their homepages showed near-identical text contents. We then analyze how many onion domains share the equivalent contents and how many distinct contents exist on a per-website-group basis. Section 4.1 elaborates on our grouping algorithms, and Section 4.1.3 shows the result.

We further investigate multiple domains that share near-identical content in each website group. We leverage external clues nudging authentic service domains and investigate domains not matching the clues in each website group. Section 4.2 describes our findings and the characteristics of observed phishing websites on the Dark Web.

Several surface websites have been hosting their services on Tor hidden services [16] for privacy-concerned users who refuse to be tracked by the service providers. Such websites that appear on both sides are not truly *dark* because the service providers of these websites are no longer anonymous. We refer to such websites as *gray websites* and analyze their unique characteristics. We then identity gray website groups, each shares an almost identical content. In each group, we excluded the gray website that matches a surface webpage and then investigated the remaining domains to manifest their phishing risks. Section 4.3 describes our algorithm for finding gray websites and discusses our findings.

4.1 Website Classification

Non-text data (e.g., videos or images) published on the Dark Web may contain illegal or abusive content [27, 53]. We refrained ourselves from analyzing visual content on the Dark Web, but focused on storing and analyzing textual data on the Dark Web. For this reason, we only scraped and stored HTTP(S) responses with textual content.

We group the domains whose homepages share near-identical text contents into one group. Our classification algorithm has two phases. Phase I groups domains based on visible texts and titles collected from homepages. For each dynamic homepage, which Phase I has overlooked, Phase II extracts a set of the words that appears on the homepage and finds the website group, of which representative word set best covers the word set from the homepage. Each website group represents a set of homepages that have different domains but share almost identical contents.

4.1.1 Classification Phase I. Phase I groups domains based on *visible texts* and *titles* on their homepages. It starts by grouping the domains that show identical texts on their homepages. In the next step, it merges the previously computed groups when any homepages from two different groups share the same title. We continue to merge groups until there is no overlapping titles. Then, each merged group is called a *website group* that represents a set of onion domains that provide almost identical textual contents on their homepages.

Our approach is a heuristic with limitations. A website group combined based on excessively common title or text does not provide meaningful insights into the group. For example, the Apache Web server default page (e.g., It Works!) could be a homepage of many onion domains. We analyzed all the titles and texts of each group and then identified 46 different common titles and texts. We deliberately ignored such texts and titles because they contain no meaningful content.

We manually analyzed dark website groups and verified whether the homepages in the same group indeed have the same content. We confirmed that the Phase I algorithm produced no error such that no website has text content different to its group's one. However, this analysis was incomplete because there were 4,816 domains left unclassified. Phase I only groups websites sharing an identical title and text content; hence, it cannot correctly group website homepages with dynamic contents (e.g., homepages with a visitor counter or a clock).

4.1.2 Classification Phase II. Phase II completes the grouping of homepages with dynamic contents. From each website group C from Phase I, a group of onion domains, we retrieved a static word set w_C that appear in common. Afterward, for each unclassified homepage, we compute its word set w_h and then find a group C s.t. $w_C \subset w_h$.

The suggested algorithm has a limitation. Phase II becomes ineffective when the representative word set for each group is quantitatively small. Therefore, we selectively compared each ungrouped homepage h with each website group C from Phase I. If the static word set size $|w_C|$ for group C is relatively smaller than the homepage word set size $|w_h|$, we excluded this grouping candidate pair, $\{C, h\}$.

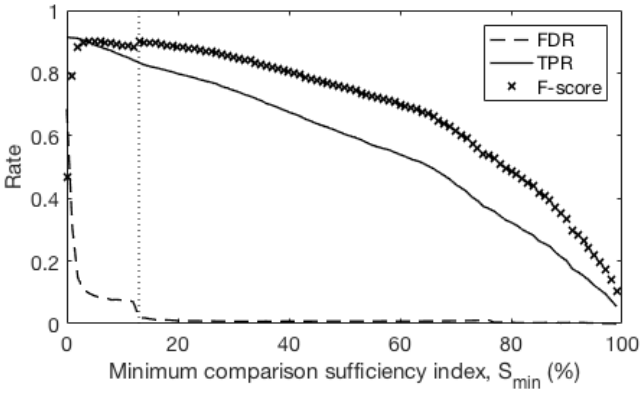


Figure 1: True positive rate (TPR), False discovery rate (FDR) and F-score of the Phase 2 classification by varying the minimum comparison sufficiency index S_{min}.

Quantitatively, for the pair $\{C, h\}$, if the comparison sufficiency index, $S_{C,h} = \frac{|w_C|}{|w_h|}$, is lower than the minimum comparison sufficiency index, S_{min}, we ignored grouping this pair. To determine the optimal S_{min}, we randomly sample 20% of the homepages from each group of onion domains from Phase I and reclassified them using different S_{min}. For each reclassification trial, we measured the true positive rate (TPR) [1] and false discovery rate (FDR) [2] by varying S_{min} as shown in Figure 1.

Our experiment shows that the F-score, the harmonic mean of TPR and 1−FDR, is highest when S_{min} was 13% (dotted line in Figure 1) and the FDR was only 1.9% based on our dataset (the TPR was 83%). We thus disregard any comparison pair whose $S_{C,h}$ is less than 13% in the final grouping phase. Phase II classified 3,397 domains into groups with only one domain member and the remaining 1,419 domains were assigned into the groups with multiple domains. For 200 sampled domains from those 1,419 domains, we manually checked their groups and confirmed that about 84% of websites had their correct grouping.

4.1.3 Classification Result. Table 2 summarizes our final grouping result. From the collected 28,928 onion domains, we found that there were 5,718 text-based distinct website groups. Of the 5,718 website groups, approximately 60% of them were single-domain website groups, and the rest (about 40%) of them were multi domain website groups. This means that 40% of the onion domains had at least one more domain whose homepage text contents were almost identical.

In the rest of this paper, a group of onion domains, which share almost identical texts on their homepages, is referred to as a *website group*. Moreover, we also counted the number of distinct 2LDs for each website group, and the result was not very different from the previous findings (approximately 40% of the website groups were associated with more than one 2LD).

[1] The proportion of websites truly in the group that are correctly classified, a.k.a. recall
[2] The proportion of websites classified to be in the group that are not originally in the group.

Table 2: Distinct website groups (WG) on the Dark Web

# of domains	1	2-50	51-100	101-200	201-300	Total
# of WG (%)	3,397 (59.31)	2,307 (40.35)	7 (.12)	5 (.09)	2 (.03)	5,718 (100)

# of 2LDs	1	2-50	51-100	101-200	201-300	Total
# of WG (%)	3,423 (59.86)	2,286 (39.98)	4 (.07)	4 (.07)	1 (.02)	5,718 (100)

4.2 Phishing Websites

A website group contains all websites that share almost identical texts. However, all websites in such a group are not necessarily the same website managed by one operator. If so, *why do onion domains in such a website group point to different websites although they share the identical content?* In this section, we analyze such domains and manifest 791 phishing onion domains that have targeted six popular dark websites.

There is no known effective method of identifying phishing websites on the Dark Web without an external clue. Dark website owners seek their anonymity and the nature of the Tor hidden network guarantees such anonymity at the protocol level. Therefore, it is improbable to decide whether two websites sharing the identical content belong to the same entity, thus hindering the inference of whether one of them is a phishing website. For the rest of this section, we use different kinds of *external clues* to identify 791 phishing domains. It means that the number of the identified phishing domains is a lower bound in 5,718 website groups, which may contain more phishing dark websites.

We identified the phishing domains that have targeted five (infamously) well-known dark websites (Table 3): Bitcoin Fog, Bitcoin Blender, Helix, AlphaBay Market and Dream Market. We specifically selected these sites, because they provide their authentic domains, which serve as clues for identifying phishing sites. However, there is a technical challenge. The five websites published their authentic domains on forums or their own websites on the Dark Web. It makes difficult for us to verify the authenticity of such information because anyone on the Dark Web can claim that his/her own website is an authentic domain for any of those five websites.

Table 3: Dark websites with known authentic domain names

Service type	Name	Authentic domains (#)	Phishing domains (#)
Bitcoin Mixers	Bitcoin Fog	1	276
	Bitcoin Blender	1	53
	Helix (Grams)	1	187
Dark Marketplaces	AlphaBay Market	16	165
	Dream Market	4	110
Total		24	791

4.2.1 Bitcoin mixers. Although all Bitcoin transactions are pseudonymous, they are public and traceable [49]. Hence, it is possible to link who has been involved in which transactions by analyzing the public blockchain [47, 55]. For this reason, Bitcoin mixing services that anonymize a user's Bitcoin transfers have gained significant popularity. Bitcoin Fog [3], Bitcoin Blender [1] and Helix [12] have been three of the most popular Bitcoin mixing services [62].

One challenge in this study is to identify the authentic onion domains of the three services, which are available exclusively on the Dark Web. Because the operators of these services conceal their identity, it is difficult for them to establish a reliable public channel to announce and promote their services. From our investigation, we learned that Bitcoin-related services are often promoted on a popular online forum called Bitcoin Forum [5], where the operators can anonymously promote their services and build up their reputations. Bitcoin Fog and Bitcoin Blender have first announced and released their authentic domain names on Bitcoin Forum, and both of these services are still currently active [2, 4]. In the case of Helix (or Grams[3]), they have done the same on Reddit [7]. Moreover, the authentic domain names of all three services have also been confirmed in a previous work [62].

By leveraging the authentic domain names of the Bitcoin mixing services and our website classification results, we were able to identify the phishing onion domains. The authentic domain of Bitcoin Fog was found in a website group with 277 different 2LDs, and we were able to conclude that the other 276 domains are phishing domains. To confirm this finding, we manually visited each domain in the Bitcoin Fog website group and investigated their content and behaviors.

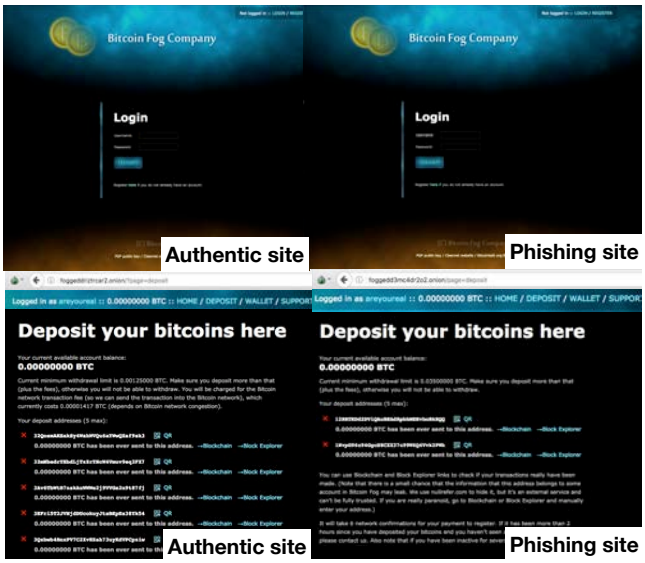


Figure 2: BitcoinFog: Authentic vs. phishing sites

The phishing websites of Bitcoin Fog were surprisingly sophisticated. As shown in Figure 2 (Top), the homepages of both the authentic and phishing sites were a login page and they were visually identical. The behaviors and functionalities of the websites

[3]Grams was a Dark Web marketplace search engine, and they additionally provided Bitcoin mixing services later on. Currently, Grams, including Helix, is discontinued. [12]

were also the same. Figure 2 (Bottom) shows that both the authentic and phishing sites allow users to add or remove Bitcoin deposit accounts. However, the authentic site and the phishing sites did not share the same backend user databases. As demonstrated in Figure 2 (Bottom), we were able to use the same username to create an account for both the authentic and phishing sites, but the Bitcoin deposit addresses linked to the accounts were different.

Bitcoin Blender's authentic domain belonged to a website group with 54 2LDs. Knowing that there is only one authentic 2LD, as shown in Table 3, we could infer that 53 2LDs in the group are phishing domains. We also manually verified each of the 53 phishing domains. Their websites sophisticatedly imitated the authentic website. The authentic website offered more functionalities, such as "Quick Mix" (Bitcoin mixing without an account) and password recovery; however, without knowing that the fact that the authentic website provides such features, it is highly likely that users will not be able to tell which one is the authentic site. Furthermore, the authentic site required users to solve CAPCHA, while the phishing sites did not.

In the case of Helix, its website group contained 188 2LDs, while only one of them was authentic. We also manually analyzed the websites linked to the domains, and we confirmed that the phishing domains led us to non-authentic websites.

4.2.2 Dark Marketplace. Another popular type of dark web services is a dark marketplace, which provides an e-commerce platform to anonymously trade illegal goods or services. Since Silk Road [29] has been terminated, there have appeared other dark marketplaces on the Dark Web. We analyzed two major dark marketplaces that have been active during our data collection period: AlphaBay and Dream Market. Compared to the other cases introduced above, it was even more difficult to identify the authentic domain names of AlphaBay and Dream Market because these sites have been using multiple number of domain names and even frequently changing them over time.

To discover the authentic domain names of AlphaBay, we took advantage of its takedown incident that happened in July 2017 [6]. According to FBI [6], "multiple computer servers used by the AlphaBay website were seized worldwide". Because the incident happened in the last month of our data collection period, we were able to capture the official seizure notice published on the seized websites. In our AlphaBay website group, there were 181 distinct 2LDs. 16 of the 2LDs have been shown the seizure notice since July 2018. Therefore, we inferred such 16 2LDs are authentic AlphaBay domain names, which makes the other 164 domains phishing.



Figure 3: Dream Market: Authentic vs. phishing sites

Dream Market is one of the largest dark marketplace that are currently active, and this market also has multiple mirrors as shown in Figure 3 (Left). Although Dream Market has published a list of the verified domain names to cope with its phishing sites, these phishing sites have also been doing the same by announcing their official domain names (Figure 3-Middle). In addition to actively promoting its authentic domain names, Dream Market provides a domain name verifier (Figure 3-Right), which allows users to check if a given onion domain name is one of the authentic Dream Market domain names or not. By using this verifier, we tested 114 2LDs and found that only four of them are authentic, which suggests that the other 110 domains are phishing domains.

4.2.3 Domain name popularity vs. authenticity. We used different types of clues to identify phishing domains for the five popular dark web services. For these identified phishing domains, we measured the popularity of such domain names on both the Dark and Surface Web. For the popularity, we counted the number of distinct dark webpages that mention each domain name, while excluding any self-references. We used 2LD names that are 16 characters long as keywords to search not only the hyperlinks but also the text that mention them from our dark webpage data collection. To measure the 2LD name popularity on the Surface Web, we counted the number of the search results returned by Google custom search API [11] for each 2LD name.

Figure 4 (a)-(e) shows the popularity of the authentic and phishing 2LD names on the Dark Web for each the service. Each point indicates the number of distinct dark webpages that mention each 2LD name, and two types of markers are used to show the authenticity of the domain names (X markers for phishing domains, O markers for authentic domains). Figure 4 (f)-(j) illustrates the popularity of the 2LD names on the Surface Web, and each point indicates the number of the search results returned from Google when searching for each 2LD name.

In the case of the three Bitcoin mixing services, each of which has the only one authentic domain, the authentic domains have been more popular than the phishing domains on both the Dark and Surface Web except for Bitcoin Blender. One of Bitcoin Blender's phishing domain names has been mentioned more than its authentic domain name on the Dark Web (Figure 4-b). On both of the web environments, the authentic domain names of the dark marketplaces have been referenced more than the phishing domain names; however, on the Dark Web, the most popular domain name of Dream Market was a phishing domain name (Figure 4-e). Furthermore, the phishing domain names have been more frequently mentioned on the Dark Web than on the Surface Web. This difference demonstrates that phishing attackers diligently spread fake information on the Dark Web by abusing their anonymity. In most cases, no result was returned by Google when searching for the phishing domain names of the five major services.

4.3 Gray Websites

A gray website is a dark website that satisfies two conditions: (1) it has the corresponding surface website providing and identical service, and (2) both websites are managed by the same entity. A gray website is usually the mirroring version of a surface website whose customers want to remain anonymous on the Web. We assumed that there are abundant phishing websites on the Dark Web that target such gray website visitors.

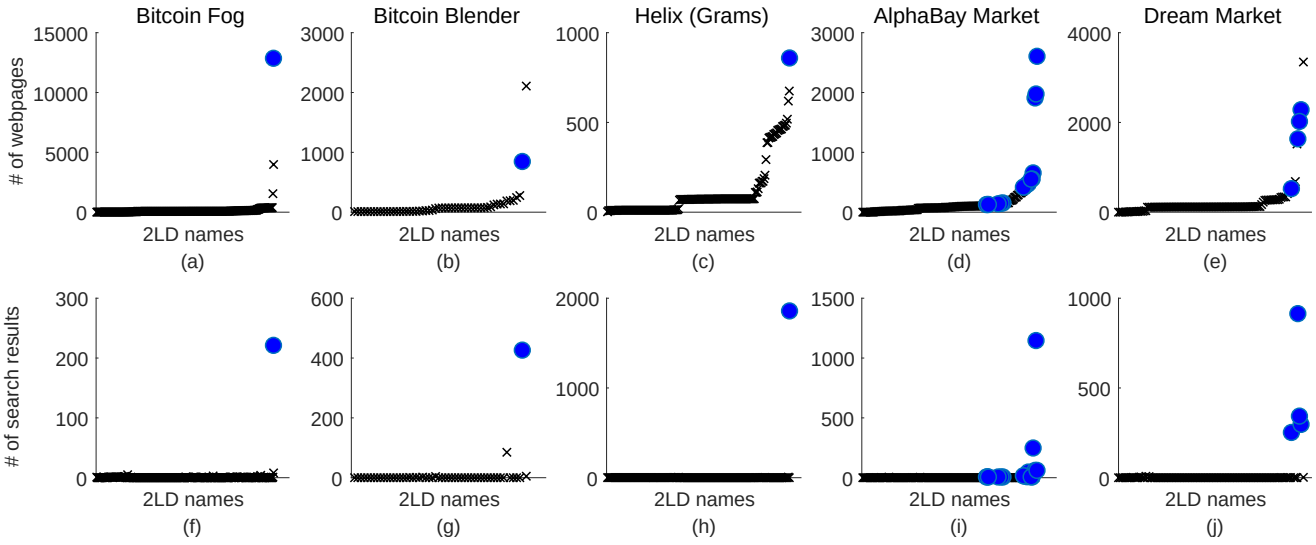


Figure 4: The popularity of five different Dark Web service domain names on the Dark (a-e) and Surface Web (f-j): The authentic domain names (in circular markers) have been mentioned more often than the phishing domain names (in x markers).

To identify such phishing websites, we first find authentic gray websites, each of which has a surface website with identical content. For each authentic gray website, we then compute its website group consisting of content duplicates. Finally, by leveraging information on each surface website paired with an authentic gray website, we confirmed phishing websites using content duplicates.

We started by identifying gray websites by partially applying the algorithm introduced by Matic et al. [46]. Our gray website identification algorithm extracts four types of clues from our dataset and then retrieves the same clues from surface websites. The extracted clues are dark website title, surface domain (e.g., non-onion domain names) in the title, Google Analytics tracking ID, and AdSense publisher ID.

We assumed that the four different clues are effective in deciding whether different websites are managed by the same entity. A homepage title is usually a representative string that shows the characteristics of the website. Because the homepages are already well-indexed by search engines, retrieving a title obtained from a dark homepage is an effective way of checking its existence on the Surface Web. Furthermore, if the title of a dark homepage includes a surface web domain, it is a strong indicator that the surface domain is the entry point of the same website. We also explored the methods of using both Google Analytics tracking and AdSense publisher IDs, which are assigned to each user. Usually, an owner of both dark and surface websites wants to maintain a unified view of his or her services. Therefore, such identifiers are compelling clues to check the same ownership of dark and surface websites.

We used a regular expression that matches surface domain names and matched 5,776 domains from the dark website titles. We then validated the matching surface domains by querying DNS servers, which confirmed 145 valid surface domains. We applied the same technique to extract the other clue types and obtained 276 Google Analytics IDs and 1,171 Google AdSense publisher IDs.

Table 4: Gray websites (GW) by each clue type

Clue type	# of search results	# of potential GW group	# of GW group
Domain in title	N/A	145	82
Homepage title	120,677	1,840	320
Analytics ID	1,241	1,049	80
AdSense ID	17,765	14,720	12

From the three clue types extracted from dark websites, we searched for surface websites using Microsoft's Bing Web Search API [17]. As Table 4 shows, we extracted surface domain candidates for each clue type. Each surface website candidate is then verified whether there exists a dark website whose body texts were similar to the contents of the surface website candidate. We used the same classification technique described in Section 4.1 to decide whether both dark and surface website candidates showed near-identical texts. As a result, we confirmed 383 gray website groups that were also available on the Surface Web.

In addition, we further investigated the contents of what had been identified as gray websites to understand their unique characteristics. We classified the website groups into three major categories: user-privacy-sensitive, provider-privacy-sensitive, and non-privacy-sensitive. A user-privacy-sensitive website is a website whose visitors would want anonymity. Such sites include pornography, file sharing, online gambling, forums, cryptocurrency services, etc. A provider-privacy-sensitive website indicates a website whose owners want to remain completely anonymous because of their illegal activities. Of the 383 gray website groups, about 35% of them were user-privacy-sensitive, about 10% of them were provider-privacy-sensitive, and the rest were non-privacy-sensitive as shown in Table 5.

Among 383 gray website groups, we further identified phishing websites. Note that each gray website group has one authentic

Table 5: Gray website categories

Privacy	Topic type	Percentage (%)	
User	adult	0.86	35.06
	political	2.01	
	forums	3.74	
	file sharing	5.17	
	cryptocurrency	6.9	
	gamble	7.18	
	communications	8.05	
	news	1.15	
Provider	illegal market	9.77	9.77
Irrelevant	academic	3.16	55.17
	software	6.9	
	security company	9.2	
	personal	17.53	
	others	18.39	

gray website with a corresponding surface webpage. We checked whether such a surface webpage explicitly mentions their authentic gray website onion domains. When a gray website group contains other domains that are not promoted by the surface website, we classify other domains as phishing websites. Table 6 summarizes the confirmed gray phishing websites. We confirmed 297 phishing domains from 383 gray website groups consisting of 734 domains. We classified 218 domains in 170 gray website groups as potential phishing websites because their surface websites did not specify their authentic onion domains.

Table 6: Phishing gray websites

Phishing	WG (#)	Authentic gray website domains (#)	Phishing domains (#)
Confirmed	213	219	297
Potential	170	N/A	N/A
Total	383	219	297

One of the phishing gray websites that we have identified targets Facebook. Since Facebook has announced the official onion domain name (facebookcorewwwi.onion) of their dark website [16], the other domains that belong to its website group were obviously phishing domains. As shown in Figure 5, the landing pages of the two domains are visually identical. One notable difference is that the Facebook's official dark website (Figure 5-Top) provides their HTTPS certificate, while the other site (Figure 5-Bottom) does not. We further analyzed the phishing site, and as shown in Figure 5 (Bottom), we confirmed that they were stealing the Facebook login credentials.

4.4 Discussion

By classifying the onion domains into content-wise identical website groups, we confirmed that about 40% of the website groups were associated with more than one onion domain. Then, *why would many website groups have more than one onion domain for their own service?*

Network balancing. One possible explanation is that many websites on the Dark Web could have been using multiple subdomains (lower than second-level) to provide identical content. However,

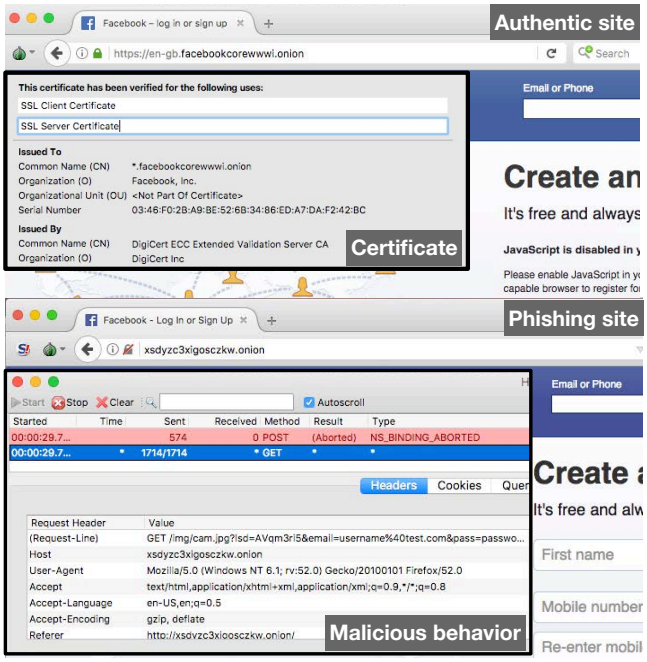


Figure 5: While the official Facebook site (Top) supports HTTPS with a valid certificate, the phishing site (Bottom) does not support HTTPS.

as shown in Table 2, the ratios of single-domain website groups to multi-domain website groups between domains and 2LDs remained almost the same. Hence, we excluded this explanation. Another explanation is that the dark websites could have deployed multiple numbers of Tor clients (or relays) to address the known scalability problems in the Tor hidden service architecture [14, 52].

One scalability problem arises when there are many concurrent requests to a 2LD. A 2LD is bounded only to a unique Tor client because it is computed from the public key assigned to the client. Consequently, one Tor client is solely responsible for the network traffic directed to and originated from its onion address. In such an architecture, the Tor clients that provide hidden services can easily become network bottlenecks as they lack load balancing options [14]. To avoid such a problem, the service providers operating a popular hidden service could have deployed multiple Tor clients, each of which has its own 2LD, to distribute the visitors.

The other scalability problem lies in the *introduction point*, as Tor announced [14], "The current hidden services architecture does not scale well due to introduction points hammered by many clients." An introduction point is a regular Tor relay, chosen by a Tor hidden service for setting up a *rendezvous point*. Thus, accessing such an introduction point is a preceding step to access the hidden service. When the service handles many concurrent requests, their service reliability depends on how many requests their introduction points can handle. This problem was addressed in a previous study [52]; however, Tor has not deployed the proposed solution.

Meanwhile, this problem is easily solvable by deploying multiple 2LDs (or Tor clients). Because multiple 2LDs have more introduction points than one 2LD, each introduction point becomes having less incoming requests of notifying rendezvous points. This simple trick

explains our observation that having multiple 2LDs is common across many dark websites to improve their service availability.

Prevalent phishing websites. We also confirm that phishing is another cause of abundant domains for each website group. We observed numerous phishing websites with content duplicates of authentic dark websites. In particular, we found that at least 791 onion domains have been hosting the phishing websites of the five well-known Dark Web services during our data collection period. After adding 297 gray phishing domains, we confirmed 901 unique phishing domains on the Dark Web.

We observed that many dark websites have been employing different strategies to cope with the phishing threat. One strategy is to leverage a popular social platform on the Surface Web. For instance, Bitcoin Fog and Bitcoin Blender have been building up their reputations on Bitcoin Forum which has more than 2M users. Bitcoin Forum serves as a reliable communication channel among their dark website visitors to establish the authenticity of their onion domains.

The other strategy is to provide an additional service telling whether a given domain is an authentic dark website service. Dream Market, for instance, has been actively announcing their official domain names, and they even provide an address verification service (Figure 3-Right) that tells whether a given domain name is one of their authentic domains or not. Interestingly, Dream Market offers a two-factor authentication (2FA) to protect their users from phishing attacks. Because their users also seek the anonymity as well, instead of requiring the email addresses of their users, they leverage the PGP public/private key pair of a user with 2FA. A user first registers her or his PGP public key. For the following authentications, the website requests the user to decrypt a cipher-text which is encrypted with her or his PGP public key. A phishing attacker is unable to access the Dream Market website even with the stolen credential of a victim unless the attacker has the victim's PGP private key.

We also measured the popularity of the authentic and phishing domain names of the six well-known Dark Web services. We counted the number of surface and dark websites that contain the authentic and phishing domains in their homepages. In most cases, the authentic domain names were relatively more popular than the phishing ones on both the Dark and Surface Web. It shows that the popularity of an onion domain name can be an indicator to determine its dark website authenticity.

Another interesting finding is that the phishing domains are rarely mentioned on the Surface Web while the authentic names are relatively popular in both of the environments. It demonstrates that the anonymous environment of the Dark Web is more susceptible to the fake information on the authentic domains than the Surface Web, which unfortunately has been abused by phishing attackers.

Gray websites. There have been a small number of popular surface websites that are also available as Tor hidden services. Such gray websites are mostly user-privacy-sensitive websites, and each aims to achieve a different goal by allowing users access to the site via the Tor network. For example, The New York Times has been available via nytimes3xbfgragh.onion [21], and as a news company, it aims to allow people around the world to avoid any Internet censorship and read its articles via the Tor network. Facebook has also launched facebookcorewwwi.onion [16] in a similar manner.

We observed that 213 surface websites released their authentic gray website domains. At the same time, we confirmed 297 phishing domains which have lured victims who wish to use gray websites. For instance, DeepDotWeb is a news site covering events and dark website reviews, accessible via www.deepdotweb.com and deepdot35wvmeyd5.onion. Our analysis discovered eight phishing domains cloning the homepage of DeepDotWeb. These phishing websites promoted phishing domains for prevailing dark marketplaces by showing forged user comments and ratings on these websites. Any visitors without knowing the authentic surface website of DeepDotWeb has no practical way of distinguishing its authentic gray website from the phishing websites.

5 DARK WEB DOMAIN NAMES

This section describes our findings regarding how much effort the service providers put into picking their choices of onion domain names. We then explain that this trend leads to many onion domain names containing keywords that reflect the contents of the dark websites. At the same time, this trend exacerbates phishing risks for users who leverage a partial word in a domain name to vet the authenticity of the current dark website.

5.1 Domain Naming Trends

A domain name is a valuable asset especially when it comes to Internet business. For e-commerce or social network services, their domains are often considered as the brands of their businesses. Likewise, a Dark Web service provider may want to obtain a desired domain name; however, onion domain names are fundamentally different from surface domains. An onion domain name must have ".onion" as its top-level domain (TLD), and its 2LD is an 80-bit number in base32 generated from a service provider's public key. Therefore, it is computationally expensive to obtain the exact domain name that a service provider asks for.

A service provider may conduct a brute-force search to find a public key that will generate the desired 2LD. However, finding the desired one (fully matching 16 characters) is computationally infeasible. The author of a brute-force onion address generation tool [19] reported that it would take approximately 2.6 million years to obtain an address that only matches the first 14 characters of the desired domain. Therefore, a service provider might prefer to use a shorter domain prefix, which drastically reduces the domain searching time. Using the same tool, if a service provider looks for a domain whose first five characters matches his/her chosen word, it would only require approximately one minute [19].

We conducted the study to understand how Dark Web service providers perceive their onion addresses. Figure 6 visualizes the domain name similarity heat map. For horizontal and vertical axes, we lexicographically sorted all the second-level onion domain names in our dataset. We then marked a cross point when two corresponding onion domains belonged to the same website group addressed in Section 4.1. We deliberately marked no point when two onion domains on both axes are the same. Finally, we clustered marked points within each 10 by 10 size cell into a color point, which depicts the number of marked cross points. That is, a color point in the figure represents the number of onion domain pairs that share the near-identical content among 100 pairs. Therefore, the existence of

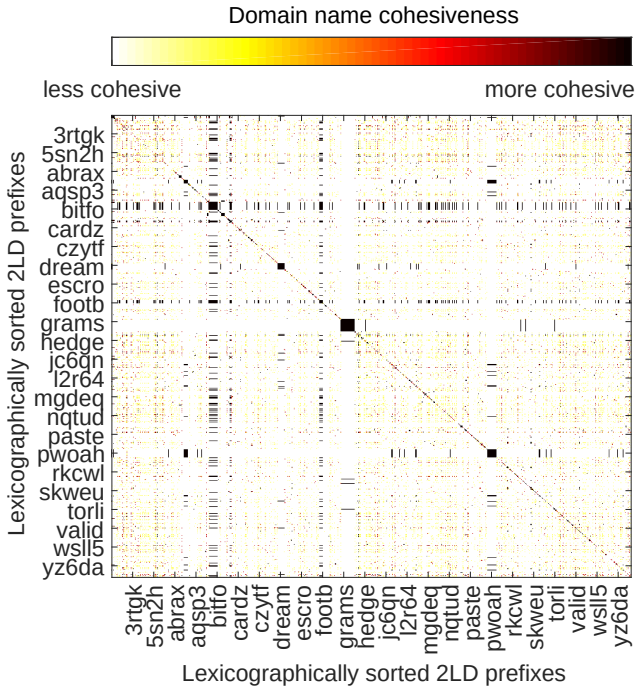


Figure 6: Name similarity heat map of 2LDs.

a clear diagonal line across the heat map of Figure 6 suggests that domain names with text-wise similar content share the same 2LD prefix.

Knowing that many similar 2LDs existed, we further examined whether such domain names were intentionally generated. We reused the website groups that were described in the previous section. We compared the 2LDs to one another for each website group and computed the longest common prefix (LCP) of each domain name combination pair. Therefore, for each website group, there are multiple LCPs out of all the possible domain name pairs. Considering that a 2LD is generated from a random public key, having a long LCP between two domain names is highly unlikely.

We found that at least 67% of the 5,718 website groups had LCPs of at least five characters long. This means that those website groups used brute-force to generate their onion domain names with their preferred prefixes. Interestingly, eight website groups selected domain names whose LCPs were 10 characters long. Generating such onion domains using Shallot [19] on a 1.5GHz processor, could take up to approximately 40 years. Based on this approximation, if a website owners uses 72 cores of Intel Xeon 2.3GHz CPU from Amazon cloud resources [8], it takes approximately 131 days and costs around $9,500.

We further analyzed the LCPs with at least five characters long. There were 184 common words (e.g., crypto, count), and 231 words were from the titles of onion domain homepages (e.g., grams7, alphabayww). Considering that these LCPs are computed from a website group with content duplicates, service providers choose a prefix of their 2LDs based on their content. In summary, a non-negligible portion of Dark Web service providers puts significant computational resources into assigning meaningful prefixes to their onion domain names.

5.2 Discussion

Each onion address is cryptographically bound to the Tor client who owns the private key corresponding to the onion address. Therefore, a correct onion domain that a user wants to visit leads to the webpage owner, who has the private key matching the domain. However, the onion domain itself is not memorable. We have observed that many service providers brute-force generate *partially memorable* onion domain names with human-readable prefixes to make their domains more intuitive.

These trends make phishing easier because people now check the authenticity of a dark website using its partial domain name. Consider two different onion domain names; bitcoinfoganfnad.onion and bitcoinfogansdse.onion. Because both of them share the common prefix "bitcoinfog", the visitors would presume that the addresses will lead them to the BitcoinFog website. However, neither of them is the authentic BitcoinFog website. To introduce one case of this type of abuse, our BitcoinFog website group has 277 domains, while one of them is the official BitcoinFog domain (foggeddriztrcar2.onion). In this website group, there were 109 onion domain names beginning with "bitfog," 40 domains with "fogged," and 19 domains with "btcfog". In the case of our AlphaBay website group (181 domains), where one of the authentic onion addresses is pwoah7foa6au2pul.onion, there were 98 domains with the common prefix "pwoah" and one domain had the longest common prefix "pwoah7foa." One light mitigation for such a phishing attack is to use a completely random onion domain so that its visitors give up on memorizing the domain and use a bookmark instead.

6 CONCLUSION

We conducted a comprehensive investigation of the phishing risks in the Tor anonymity network. We demonstrated that only 5,718 website groups with distinct contents exist among 28,928 onion domains and confirmed 901 phishing domains with content duplicates. We investigated how much effort Dark Web service providers put into selecting their onion domains and addressed that this trend brings an unfortunate side-effect of having many phishing sites with shared onion domain prefixes. We carefully designed the analysis algorithms to identity the phishing risks on the Dark Web and presented the severe phishing risk on the Dark Web, which urges further research to provide users with authentic dark Web services.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their concrete feedback. This research was supported by the Engineering Research Center Program through the National Research Foundation of Korea (NRF) funded by the Korean Government MSIT (NRF-2018R1A5A1059921).

REFERENCES

[1] Bitcoin Blender. http://bitblendervrfkzr.onion.
[2] Bitcoin Blender thread on Bitcoin Talk forum. https://bitcointalk.org/index.php?topic=436467.0.
[3] Bitcoin Fog. http://foggeddriztrcar2.onion.

[4] Bitcoin Fog thread on Bitcoin Talk forum. https://bitcointalk.org/index.php?topic=50037.0.

[5] Bitcoin Forum. https://bitcointalk.org.

[6] Darknet Takedown: Authorities Shutter Online Criminal Market AlphaBay. https://www.fbi.gov/news/stories/alphabay-takedown.

[7] DarkNetMarkets community on Reddit. https://www.reddit.com/r/DarkNetMarkets.

[8] EC2 Instance Pricing - Amazon Web Service (AWS). https://aws.amazon.com/ec2/pricing/on-demand/?nc1=h_ls.

[9] Ethical Tor Research: Guidelines. https://blog.torproject.org/ethical-tor-research-guidelines.

[10] FreshOnion: Tor Hidden Service Directory. http://zlal32teyptf4tvi.onion/.

[11] Google Custom Search JSON API. https://developers.google.com/custom-search/json-api/v1/overview.

[12] Grams, The Google Of The Dark Web Has Shuttered Operations. https://www.forbes.com/sites/zarastone/2017/12/16/grams-the-google-of-the-dark-web-has-shuttered-operations/#1ccbc5257624.

[13] Hansa Market. https://www.deepdotweb.com/marketplace-directory/listing/hansa-market/.

[14] Hidden Services need some love. https://blog.torproject.org/hidden-services-need-some-love.

[15] International Raids Target Sites Selling Contraband on the Dark Web. https://www.nytimes.com/2014/11/08/world/europe/dark-market-websites-operation-onymous.html.

[16] Making Connections to Facebook more Secure. https://www.facebook.com/notes/protect-the-graph/making-connections-to-facebook-more-secure/1526085754298237/.

[17] Microsoft Cognitive Services - Bing Web Search API. https://azure.microsoft.com/en-us/services/cognitive-services/bing-web-search-api/.

[18] Scrapy. https://scrapy.org.

[19] Shallot. https://github.com/katmagic/Shallot/.

[20] Splash - A javascript rendering service. https://splash.readthedocs.io/en/stable/.

[21] The New York Times is Now Available as a Tor Onion Service. https://open.nytimes.com/https-open-nytimes-com-the-new-york-times-as-a-tor-onion-service-e0d0b67b7482.

[22] Venezuela is blocking access to the Tor network. https://www.theverge.com/2018/6/25/17503680/venezuela-tor-blocked-web-censorship.

[23] Ahmia. Ahmia: Tor Hidden Service Search Engine. https://ahmia.fi/.

[24] J. Aldridge and D. Decary-Hetu. Not an'Ebay for Drugs': The Cryptomarket'Silk Road'as a Paradigm Shifting Criminal Innovation. 2014.

[25] M. J. Barratt, J. A. Ferris, and A. R. Winstock. Use of Silk Road, the online drug marketplace, in the United Kingdom, Australia and the United States. *Addiction*, 109(5):774–783, 2014.

[26] A. Biryukov and I. Pustogarov. Bitcoin over Tor isn't a good idea. In *Proceedings of the 2015 IEEE Symposium on Security and Privacy (SP)*, pages 122–134. IEEE, 2015.

[27] A. Biryukov, I. Pustogarov, F. Thill, and R.-P. Weinmann. Content and popularity analysis of Tor hidden services. In *Proceedings of the 2014 IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pages 188–193. IEEE, 2014.

[28] A. Biryukov, I. Pustogarov, and R.-P. Weinmann. Trawling for tor hidden services: Detection, measurement, deanonymization. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy (SP)*, pages 80–94. IEEE, 2013.

[29] N. Christin. Traveling the silk road: A measurement analysis of a large anonymous online marketplace. In *Proceedings of the 22nd international conference on World Wide Web*, pages 213–224. ACM, 2013.

[30] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Designing privacy enhancing technologies*, pages 46–66. Springer, 2001.

[31] M. Cova, C. Kruegel, and G. Vigna. There is no free phish: An analysis of "free" and live phishing kits. In *Proceedings of the Conference on USENIX Workshop on Offensive Technologies*. USENIX Association, 2008.

[32] R. Dhamija, J. Tygar, and M. Hearst. Why phishing works. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2006.

[33] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. Technical report, Naval Research Lab Washington DC, 2004.

[34] D. S. Dolliver. Evaluating drug trafficking on the Tor Network: Silk Road 2, the sequel. *International Journal of Drug Policy*, 26(11):1113–1123, 2015.

[35] S. Ghosh, P. Porras, V. Yegneswaran, D. Ken, and N. Ariyam. ATOL: A Framework for Automated Analysis and Categorization of the Darkweb Ecosystem. In *AAAI Workshop on Artificial Intelligence for Cyber Security (AICS)*. AAAI, 2017.

[36] X. Han, N. Kheir, and D. Balzarotti. Phisheye: Live monitoring of sandboxed phishing kits. In *ACM Conference on Computer and Communications Security*. ACM, 2016.

[37] HERMES Center for Transparency and Digital Human Rights. Tor2web: Browse the Tor Onion Services. https://tor2web.org/.

[38] The New Research from Northeastern University. https://blog.torproject.org/blog/new-research-northeastern-university.

[39] The Invisible Internet Project (I2P). https://geti2p.net/.

[40] T. Jagatic, N. Johnson, M. Jakobsson, and F. Menczer. Social phishing. *Commun. ACM*, 2007.

[41] M. Jakobsson and S. Myers. *Phishing and Countermeasures: Understanding the Increasing Problem of Electronic Identity Theft*. Wiley-Interscience.

[42] R. Jansen, F. Tschorsch, and A. Johnson. The sniper attack: Anonymously deanonymizing and disabling the Tor network. In *Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS)*. The Internet Society, 2014.

[43] A. Kwon, M. AlSabah, D. Lazar, M. Dacier, and S. Devadas. Circuit fingerprinting attacks: Passive deanonymization of tor hidden services. In *Proceedings of the 24th USENIX Security Symposium*, pages 287–302. USENIX Association, 2015.

[44] K. Levchenko, A. Pitsillidis, N. Chachra, B. Enright, M. Félegyházi, C. Grier, T. Halvorson, C. Kanich, C. Kreibich, H. Liu, et al. Click trajectories: End-to-end analysis of the spam value chain. In *Security and Privacy (S&P 2011)*, pages 431–446, 2011.

[45] K. Li, P. Liu, Q. Tan, J. Shi, Y. Gao, and X. Wang. Out-of-band discovery and evaluation for tor hidden services. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing (SAC)*, pages 2057–2062. ACM, 2016.

[46] S. Matic, P. Kotzias, and J. Caballero. Caronte: Detecting location leaks for deanonymizing tor hidden services. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1455–1466. ACM, 2015.

[47] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 127–140. ACM, 2013.

[48] T. Moore and R. Clayton. Discovering phishing dropboxes using email metadata. In *eCrime Researchers Summit*, 2012.

[49] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[50] R. Overdorf, M. Juarez, G. Acar, R. Greenstadt, and C. Diaz. How Unique is Your. onion? An Analysis of the Fingerprintability of Tor Onion Services. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017.

[51] L. Overlier and P. Syverson. Locating hidden servers. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy (SP)*, pages 100–114. IEEE, 2006.

[52] L. Overlier and P. Syverson. Valet services: Improving hidden servers with a personal touch. *Lecture Notes in Computer Science*, 4258:223–244, 2006.

[53] G. Owen and N. Savage. Empirical analysis of Tor hidden services. *IET Information Security*, 10(3):113–118, 2016.

[54] A. Ramachandran and N. Feamster. Understanding the network-level behavior of spammers. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. ACM, 2006.

[55] T. Ruffing, P. Moreno-Sanchez, and A. Kate. P2p mixing and unlinkable bitcoin transactions. In *NDSS*, 2017.

[56] A. Sanatinia and G. Noubir. Honey onions: a framework for characterizing and identifying misbehaving tor hsdirs. In *Proceedings of the 2016 IEEE Conference on Communications and Network Security (CNS)*, pages 127–135. IEEE, 2016.

[57] I. Sanchez-Rola, D. Balzarotti, and I. Santos. The Onions Have Eyes: A Comprehensive Structure and Privacy Analysis of Tor Hidden Services. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*, pages 1251–1260. ACM, 2017.

[58] K. Soska and N. Christin. Measuring the Longitudinal Evolution of the Online Anonymous Marketplace Ecosystem. In *Proceedings of the 24th USENIX Security Symposium*, pages 33–48. USENIX Association, 2015.

[59] B. Stone-Gross, T. Holz, G. Stringhini, and G. Vigna. The underground economy of spam: A botmaster's perspective of coordinating large-scale spam campaigns. In *Proceedings of the Conference on Large-scale Exploits and Emergent Threats*. USENIX Association, 2011.

[60] P. Syverson and G. Boyce. Bake in. onion for tear-free and stronger website authentication. *IEEE Security & Privacy*, 14(2):15–21, 2016.

[61] K. Thomas, F. Li, A. Zand, J. Barrett, J. Ranieri, L. Invernizzi, Y. Markov, O. Comanescu, V. Eranti, A. Moscicki, D. Margolis, V. Paxson, and E. Bursztein. Data breaches, phishing, or malware? understanding the risks of stolen credentials. In *ACM Conference on Computer and Communications Security*. ACM, 2017.

[62] M. Tran, L. Luu, M. S. Kang, I. Bentov, and P. Saxena. Obscuro: A bitcoin mixer using trusted execution environments. *IACR Cryptology ePrint Archive*, 2017:974, 2017.

[63] M. C. Van Hout and T. Bingham. âĂŸSilk RoadâĂŹ, the virtual drug marketplace: A single case study of user experiences. *International Journal of Drug Policy*, 24(5):385–391, 2013.

[64] S. Zawoad, A. Dutta, A. Sprague, R. Hasan, J. Britt, and G. Warner. Phish-net: Investigating phish clusters using drop email addresses. In *APWG eCrime Researchers Summit*, 2013.

Hierarchical Temporal Convolutional Networks for Dynamic Recommender Systems

Jiaxuan You
Stanford University
Pinterest
jiaxuan@cs.stanford.edu

Yichen Wang
Georgia Institute of Technology
Pinterest
yichen.wang@gatech.edu

Aditya Pal
Pinterest
apal@pinterest.com

Pong Eksombatchai
Pinterest
pong@pinterest.com

Chuck Rosenberg
Pinterest
crosenberg@pinterest.com

Jure Leskovec
Stanford University
Pinterest
jure@cs.stanford.edu

ABSTRACT

Recommender systems that can learn from cross-session data to dynamically predict the next item a user will choose are crucial for online platforms. However, existing approaches often use out-of-the-box sequence models which are limited by speed and memory consumption, are often infeasible for production environments, and usually do not incorporate cross-session information, which is crucial for effective recommendations. Here we propose Hierarchical Temporal Convolutional Networks (HierTCN), a hierarchical deep learning architecture that makes dynamic recommendations based on users' sequential multi-session interactions with items. HierTCN is designed for web-scale systems with billions of items and hundreds of millions of users. It consists of two levels of models: The high-level model uses Recurrent Neural Networks (RNN) to aggregate users' evolving long-term interests across different sessions, while the low-level model is implemented with Temporal Convolutional Networks (TCN), utilizing both the long-term interests and the short-term interactions within sessions to predict the next interaction. We conduct extensive experiments on a public XING dataset and a large-scale Pinterest dataset that contains 6 million users with 1.6 billion interactions. We show that HierTCN is 2.5x faster than RNN-based models and uses 90% less data memory compared to TCN-based models. We further develop an effective data caching scheme and a queue-based mini-batch generator, enabling our model to be trained within 24 hours on a single GPU. Our model consistently outperforms state-of-the-art dynamic recommendation methods, with up to 18% improvement in recall and 10% in mean reciprocal rank.

ACM Reference Format:
Jiaxuan You, Yichen Wang, Aditya Pal, Pong Eksombatchai, Chuck Rosenberg, Jure Leskovec. 2019. Hierarchical Temporal Convolutional Networks for Dynamic Recommender Systems. In *Proceedings of the 2019 World Wide Web Conf. (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, NY, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313747

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313747

1 INTRODUCTION

For many web applications, making item recommendations that match users' interests is of key importance. Effective recommendations greatly improve users experience and retention, which leads to long-term increase in engagement. In real-world scenarios, user's interests dynamically shift and evolve over time. While interests of a user across different sessions might depend on their long-term interests and are hence somewhat stable, the short-term in-session interests tend to evolve rapidly. Therefore, an ideal recommender system should be able to capture both levels of user's dynamic interests and update those interests in real-time based on user's interactions.

The most direct data sources to build a recommender system are users' past sequences of interactions, which are abundant and well structured. Currently, rule-based models are still widely used to make dynamic recommendations. Many works have shown that rule-based models, e.g., recommending the items with the largest number of interactions, is in fact a very strong baseline [12, 23]. Recent years have witnessed the power of sequence-based deep learning models in computer vision [22], natural language processing [19], and graph structured data [37]. In addition, recent works have applied sequence models for dynamic recommender systems [12, 31, 35] as well. These techniques usually involve using a specific sequence model, such as RNN or Convolutional Neural Networks (CNN), to encode users' past interactions into a latent feature space, which is then used for future predictions.

However, none of these approaches is suitable for recommender systems that scale to modern web-scale production environments with hundreds of millions of user, billions of items and tens of billions of interactions per day. In particular, rule-based models fail to perform well in complex large-scale tasks. Purely RNN-based or CNN-based approaches do not capture the hierarchical nature of in- and cross-session user interests. In addition, RNN-based approaches are slow and difficult to train on massive data due to issues with gradient backpropagation. CNN-based approaches have high memory consumption, and do not involve smooth and interpretable latent representations than can be reused for down-stream tasks.

Here we propose Hierarchical Temporal Convolutional Networks (HierTCN), a novel neural architecture for modeling users'

[1] Work done while at Pinterest.

sequential interactions, which enables real-time large-scale recommender systems. We design our model with modern web-scale recommender system production environment in mind; therefore, HierTCN is memory-efficient and fast to compute. HierTCN consists of two levels of models to capture hierarchical levels of user interests. The high-level model uses RNN to capture users' evolving long-term interests across sessions, while the low-level model is implemented with TCN, utilizing both a user's long-term interests and the short-term interactions within sessions to output a dynamic user embedding and make recommendations. We apply HierTCN to a public dataset as well as a large private dataset with 1.7 billion training examples. We show that HierTCN has the following benefits over existing approaches:

(1) HierTCN has a significant performance improvement over existing deep learning models by about 30% on a public XING dataset and 18% on a private large-scale Pinterest dataset.
(2) Compared with RNN-based approaches, HierTCN is 2.5 times faster in terms of training time and allows for much easier gradient backpropagation.
(3) Compared with CNN-based approaches, HierTCN requires roughly 10% data memory usage and allows for easy latent feature extraction.

Based on HierTCN, we build a dynamic recommender system that scales to millions of users and billions of interactions. In terms of scale, it is at least 100 times larger than existing dynamic deep learning based recommender systems. The HierTCN-based recommender system features in the following components:

(1) A framework for joint modeling the dynamics of millions of users and items, which is not possible by existing approaches.
(2) An efficient offline training pipeline for HierTCN, which consists of efficient data caching and mini-batch generator.
(3) An efficient online inference pipeline that enables real-time recommendations using HierTCN.

2 RELATED WORK

Sequence models. CNNs and RNNs are two important architectures for sequence modeling. The sequential nature of RNN has made it the default choice for sequence modeling, and Long short-term Memory (LSTM) [13] and GRU [4] are the two most popular RNN variants. CNN has a long history of its application to sequence modeling as well [18]. 1D CNN with dilated convolutions have been shown to be powerful for audio data [21]. The idea has been further developed by [2] and is summarized as Temporal Convolutional Network (TCN), demonstrating its capability of modeling sequences in general. Our work here builds on this line of work but extends it by making it feasible for modeling hierarchical user interests and large-scale production environments.

Static recommender systems. Recently, there has been a surge of interest in applying deep learning to recommendation systems. Several approaches treat user interactions as static information by ignoring the temporal dimension, and try to learn a static similarity matrix between users and items [7, 11, 14, 26]. In contrast, our work explicitly considers the dynamics of user-item interactions through sequence modeling.

Dynamic recommender systems. RNN is the most widely used architecture for dynamic recommender systems [6, 12, 16, 23, 34]. Concretely, [34] captures temporal aspects of user-item interactions and uses LSTM cell coupled with stationary factors to identify movie popularity fluctuations. [16] interpolates k-nearest-neighbor method with a session-based RNN [12] and demonstrates performance gains over static recommender systems. Recently, [6, 32] combine point process models with the RNN-style state update functions to capture the co-evolution of user and item embeddings. However, these works only use one RNN network and do not explicitly consider multi-session settings. [23] is the most relevant work with our paper, where a hierarchical Gated Recurrent Units (GRU) framework is proposed. Our work differs in the way we design the hierarchy, the application of the TCN model, and many practical techniques that allow for deployment on real-world large-scale problems. Experimental results also show significant performance improvement of our model in both accuracy and speed. Furthermore, TCN-based sequence models remain largely unexplored for recommender systems, except for a very recent preprint [38]. In comparison, our work significantly alleviates the memory consumption issue of TCN which restricts its large-scale application, and explicitly models different levels of user interests.

3 PROPOSED MODEL

3.1 Problem Setup

We consider the problem of building a dynamic recommender system that adapts to users' preferences in real-time. For each user, we observe a sequence of items $\mathbf{c} = (\mathbf{c}_1, \mathbf{c}_2, ...\mathbf{c}_n)$ that are impressed to a user, where $\mathbf{c}_t \in \mathbb{R}^{d \times m_t}$ are the embeddings of m_t impressions at time t. The user interacts with some subset of the impressed items, which are denoted as $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_n)$, where $\mathbf{x}_t \in \mathbb{R}^d$ is the embedding of the interacted item at time step t, with n being the total number of interactions for the user. In our context, an interaction $\mathbf{x}_t$ refers to *positive* actions such as clicking, saving, sharing, or buying an item, while impressions $\mathbf{c}_t$ refer to just viewing items. In addition, we assume that interactions can be segmented into sessions using a function $q(\cdot)$, such that $i = q(t)$ indicates that the time step t belongs to session i. Our goal is to learn a function that predicts $\mathbf{x}_t$ from $\mathbf{c}_{\leq t}$ and $\mathbf{x}_{<t}$.

3.2 High-level Sketch of Our Approach

Our solution is based on inferring user and item embeddings from the historical data $(\mathbf{x}, \mathbf{c})$. We note that typically users' interests vary significantly within sessions. Hence, it is crucial to update user embeddings in near real-time in order to model their short-term interests accurately. On the other-hand, item embeddings can be fairly stable and only require update after regular time-intervals and in-between these intervals they can be considered static for the sake of efficiency. In practice, we update item embeddings daily, whereas user embeddings are updated in near real-time. This choice is discussed in detail in Section 3.6.1.

The main goal of our recommender system is to predict the next interaction from a set of candidate items. We formulate the problem using a function $f_\theta(\cdot)$ that infers the *user embeddings* $\mathbf{u}_t$, then compute the similarity between $\mathbf{u}_t$ and the candidate item embeddings to produce an unnormalized conditional distribution

$p(\mathbf{x}_t|\mathbf{x}_{<t})$. Specifically,

$$p(\mathbf{x}_t|\mathbf{x}_{<t}) = S(\mathbf{x}_t, \mathbf{u}_t); \quad \mathbf{u}_t = f_\theta(\mathbf{x}_{<t}) \tag{1}$$

where $f_\theta(\cdot)$ is implemented as a deep neural network, and $S(\cdot)$ is a scoring function, e.g., a dot product, that measures the similarity between $\mathbf{x}_t$ and $\mathbf{u}_t$. To finally make a recommendation, we evaluate $p(\mathbf{x}_t|\mathbf{x}_{<t})$ over a pool of candidate items and rank the items in descending order. When the impression data $\mathbf{c}$ is known, the candidate pool consists of m_t impressions at time step t; otherwise, the candidate pool could be all the items, a random subset of all the items, or a selection of items via some simple heuristics. Next, we review the key ingredients that constitute $f_\theta(\cdot)$.

3.3 Single-level Sequence Models

We review the two recent state-of-the-art sequence models, which serve as the building blocks of our model.

3.3.1 Recurrent Neural Networks. RNNs are naturally designed for modeling sequences. RNN maintains a hidden state vector that is updated with new inputs using the following update function:

$$\mathbf{s}_t = \sigma(\mathbf{W}\mathbf{x}_t + \mathbf{U}\mathbf{s}_{t-1}) \tag{2}$$

where $\mathbf{x}_t$ is the input at time step t, $\mathbf{s}_t$ is the hidden state, and $\mathbf{W}, \mathbf{U}$ are trainable parameters. RNNs are known to suffer from vanishing gradient problem, and Gated Recurrent Unit (GRU) is a popular model that mitigates the issue [4]. The update function of a GRU can be written as

$$\begin{aligned}
\mathbf{g}_t &= \sigma(\mathbf{W}^g\mathbf{x}_t + \mathbf{U}^g\mathbf{s}_{t-1}) \\
\mathbf{r}_t &= \sigma(\mathbf{W}^r\mathbf{x}_t + \mathbf{U}^r\mathbf{s}_{t-1}) \\
\mathbf{h}_t &= \tanh(\mathbf{W}^h\mathbf{x}_t + \mathbf{U}^h(\mathbf{s}_{t-1} \odot \mathbf{r_t})) \\
\mathbf{s}_t &= (1 - \mathbf{g}_t) \odot \mathbf{h}_t + \mathbf{g}_t \odot \mathbf{s}_{t-1}
\end{aligned} \tag{3}$$

where $\mathbf{x}_t$ is the input at time step t, $\mathbf{s}_t$ is the hidden state, $\mathbf{g}_t$ is the update gate, $\mathbf{r}_t$ is the reset gate, $\mathbf{h}_t$ is the candidate activation, $\sigma(\cdot)$ is the sigmoid function, and $\mathbf{W}^g, \mathbf{W}^r, \mathbf{W}^h, \mathbf{U}^g, \mathbf{U}^r, \mathbf{U}^h$ are trainable parameters. A Multilayer Perceptron (MLP) with ReLU activation is then used to output prediction $\mathbf{u}_{t+1}$:

$$\mathbf{u}_{t+1} = \mathbf{W}^{(2)}\text{ReLU}(\mathbf{W}^{(1)}\mathbf{s}_t + \mathbf{b}^{(1)}) + \mathbf{b}^{(2)} \tag{4}$$

where $\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}$ are trainable parameters. Overall, GRU is a satisfying model for $f_\theta(\cdot)$, as $\mathbf{u}_{t+1}$ depends on all the previous interactions $\mathbf{x}_{\le t}$.

3.3.2 Temporal Convolutional Networks. TCN is a special type of 1D CNN, which is a natural way to encode information from a sequence [2]. A vanilla 1D convolutional layer can be written as

$$\begin{aligned}
F(\mathbf{x}_t) &= (\mathbf{x} * \mathbf{f})(t) = \sum_{j=0}^{k-1} \mathbf{f}_j^T \mathbf{x}_{t-j}, t \ge k \\
\mathbf{u} &= (F(\mathbf{x}_k), F(\mathbf{x}_{k+1}), ..., F(\mathbf{x}_n))
\end{aligned} \tag{5}$$

where $\mathbf{x}$ is the input sequence, $\mathbf{u}$ is the output sequence, and $\mathbf{f} \in \mathbb{R}^{k\times d}$ is a convolution filter with size k. A 1D CNN is then constructed by stacking several vanilla 1D convolutional layers. However, 1D CNN is restricted by its shrinking output size and limited receptive fields when being applied to model sequences, while TCN features with two techniques that solve these problems namely causal convolutions and dilated convolutions.

Causal convolutions. As is shown in Equation 5, a vanilla 1D convolutional layer takes as input a length n sequence and outputs a length $n-k+1$ sequence. The output can shrink further if more such layers are stacked together. This property can be problematic in our domain, as we want our model to make predictions at every time step and make updates in real-time. A causal convolutional layer solves the problem by concatenating a length $k-1$ zero padding at the beginning of the input sequence. Furthermore, it ensures that there is no information leakage from the future into the past, which is crucial when predicting future interactions. Concretely,

$$\begin{aligned}
F(\mathbf{x}_t) &= (\mathbf{x} * \mathbf{f})(t) = \sum_{j=0}^{k-1} \mathbf{f}_j^T \mathbf{x}_{t-j} \quad \mathbf{x}_{\le 0} := \mathbf{0} \\
\mathbf{u} &= (F(\mathbf{x}_1), F(\mathbf{x}_2), ..., F(\mathbf{x}_n))
\end{aligned} \tag{6}$$

This formulation ensures that the output sequence $\mathbf{u}$ is well-defined over each time step, and prediction $\mathbf{u}_t$ only depends on input $\mathbf{x}_{\le t}$.

Dilated convolutions. Another issue with vanilla 1D CNN is that it has a receptive field linear to the number of layers, which in our case is undesirable since we aim to model long-term dependencies. Dilated convolution is a technique that allows for receptive fields exponential to the number of layers. Specifically, when combining with causal convolution, the r^{th} level dilated convolutional layer can be written as

$$\begin{aligned}
F(\mathbf{x}_t) &= (\mathbf{x} *_{l_r} \mathbf{f})(t) = \sum_{j=0}^{k-1} \mathbf{f}_j^T \mathbf{x}_{t-l_r \cdot j} \quad \mathbf{x}_{\le 0} := \mathbf{0} \\
\mathbf{u} &= (F(\mathbf{x}_1), F(\mathbf{x}_2), ..., F(\mathbf{x}_n))
\end{aligned} \tag{7}$$

where l_r is the dilation factor which can be set as $(k-1)^{r-1}$ to achieve exponentially large receptive field. We refer to this formulation in Equation 7 as temporal convolutional layer.

A Temporal Convolutional Network (TCN) is then constructed by stacking multiple temporal convolutional layers. To facilitate training a deep TCN, a common practice is to organize temporal convolutional layers into blocks, and add residual connections [10] between blocks. By setting a proper size of filter and number of layers, $\mathbf{u}_{t+1}$ can depend on the full historical interactions $\mathbf{x}_{\le t}$.

3.4 Design Choices of Hierarchical Sequence Models

It is reasonable to consider using one of the above single-level sequence models to predict users' future interactions based on their past interactions. However, this formulation omits the session information which implies a hierarchy of user interests. Users typically have long-term interests that span across multiple sessions. These interests need to be updated after the end of each session, i.e. in the order of few hours to days. On the other hand, users' short-term interests are reflected via the interactions within each session, which must be updated in the order of a few seconds. In principle, we could include the session information as a feature to the single-level sequence models, enabling them to learn the short- and long-term interests. However, our experiments show that this approach does not work well in practice. The issue is that such an approach does not have the inductive bias over the hierarchy of user interests and thus fails to generalize to unseen cases.

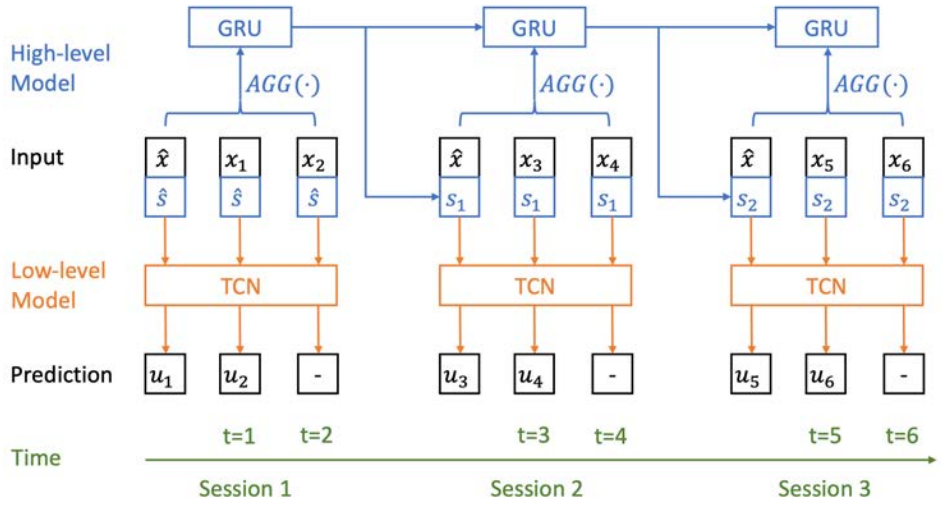


Figure 1: Visualization of HierTCN architecture. HierTCN generates predictions $\mathbf{u}_1, \mathbf{u}_2, ...\mathbf{u}_n$ based on a sequence of interactions $\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_n$. The high-level model (blue) is implemented with GRU which is updated by an aggregation of each session of interactions using function AGG(·). The low-level model (orange) uses TCN to predict user embeddings at each time step, based on a user's past interactions within the session and the hidden state $\mathbf{s}_i$ of the high-level model. $\hat{\mathbf{x}}$, $\hat{\mathbf{s}}$ are the default start tokens, which are used to produce the first output for the low-level model and the high-level model, respectively.

Rather than directly applying single-level sequence models, we impose the inductive biases over users' hierarchical interests via incorporating the hierarchical structure in the design of the neural network architecture. Specifically, our hierarchical model has a low-level component that learns from interactions within a session, and a high-level component that carries over information across sessions. The low-level component outputs a user embedding $\mathbf{u}_t$ using a neural network $f_\omega(\cdot)$, which can be written as:

$$\mathbf{u}_t = f_\omega((\mathbf{x}_i | \forall q(i) = q(t), i < t), \mathbf{s}_{q(t)}) \tag{8}$$

where $(\mathbf{x}_i | \forall q(i) = q(t), i < t)$ is the sequence of within-session item embeddings before time step t, and $s_{q(t)}$ represents a user's long-term interest which is modeled by a high-level component:

$$\mathbf{s}_{q(t)} = f_\phi((\mathbf{x}_i | \forall q(i) < q(t))) \tag{9}$$

where $f_\phi(\cdot)$ is the high-level model that summarizes the sequence of interactions before the session $q(t)$. In the remainder of this section, we will discuss different ways of designing hierarchical sequence models over multi-session data.

3.4.1 Choices of sequence models. The low-level model focuses on predicting the next interaction, based on both within-session interactions and the high-level information. The task is similar to making sequential prediction without considering the session structure, but over a much shorter sequence compared with taking all historical interactions of a user as input. Although RNN is generally much slower than TCN since the computation cannot be paralleled, both RNN and TCN can be chosen to represent $f_\omega(\cdot)$.

For the high-level model, the goal is to model a user's long-term interests. For this task, RNN has a natural advantage, because it maintains a hidden representation over different time steps, which can be naturally interpreted as a representation of the user's long-term interests. In contrast, TCN calculates the output directly through layers of convolution. Since there is no hidden state being maintained, the output is less smooth and interpretable, which is undesirable, especially when we would like to model a relatively stable long-term interests for all users. Moreover, TCN requires taking the raw input sequences as input. For the high-level model, this requirement corresponds to keeping track of users' entire history of interactions whenever inference is needed, resulting in high data memory usage, which is undesirable for large-scale implementation. Thus, we conclude that RNN is more suitable for representing $f_\phi(\cdot)$.

3.4.2 Choices of combining different levels of models. There are two problems to consider when trying to combine the low-level and the high-level model. The first problem is how to condition the low-level prediction on the high-level information, to which there are two approaches. The first approach is to use the high-level representation to initialize the low-level model; the second approach is to enforce the high-level representation propagated into the low-level model at each time step. These two approaches are first explored in [23], and we refer to them as *Init connection* and *Full connection*, respectively.

The second problem is to update the high-level model with low-level information. Suppose a user has n total interactions over m sessions, and function AGG(·) aggregates the embeddings within a session whose output is then used to update the high-level model. [23] explores using the low-level model as the aggregation function, where the final hidden state of a RNN is used to update the high-level model. However, this approach complicates the path of gradient backpropagation, as the longest backpropagation path has a length of n. In addition, the method couples the training of the low-level model and the high-level model; consequently, the biased low-level model at the start of training can negatively affect the training of the high-level model, resulting in a slow convergence. In contrast, we propose to use a simpler AGG(·) function, e.g., the MEAN(·) pooling function, by assuming that the ordering

of within-session interactions carries little information. Using a MEAN$(\cdot)$ pooling function only results in a length-m backpropagation path and decouples the training of the low-level model and the high-level model. We empirically find that this approach converges much faster and provides much better prediction performance.

3.5 HierTCN

Based on the discussions in Section 3.4, we design HierTCN, an efficient and scalable hierarchical sequence model.

3.5.1 Model design. HierTCN consists of a high-level model implemented with GRU and a low-level model implemented with TCN. The long-term user interests are represented via GRU's hidden state. The hidden state is updated by an aggregation over item embeddings within each session, using the MEAN$(\cdot)$ pooling function. Specifically, after all the interactions in a session $q(t)$ have been observed, user's long-term interests, represented as the GRU hidden state $\mathbf{s}_{q(t)}$, are updated using the following equation:

$$\mathbf{s}_{q(t)} = \text{GRU}(\mathbf{s}_{q(t)-1}, \text{MEAN}((\mathbf{x}_i | \forall q(i) = q(t)))) \tag{10}$$

where $q(t)$ refers to the session that contains time step t, $(\mathbf{x}_i | \forall q(i) = q(t))$ is the sequence of item embeddings in the session $q(t)$, and GRU$(\cdot)$ is a GRU described in Section 3.3.1.

The low-level TCN model represents short-term interests of the users and predicts their next interactions within the session. We use the *Full connection* (Section 3.4.2), i.e., include $\mathbf{s}_{q(t)}$ as the input to the TCN at each time step. TCN outputs a prediction $\mathbf{u}_t$, which can be interpreted as a dynamic user embedding. Concretely,

$$\mathbf{u}_t = \text{TCN}((\text{CONCAT}(\mathbf{x}_i, \mathbf{s}_{q(t)-1}) | \forall q(i) = q(t), i < t)) \tag{11}$$

where CONCAT$(\cdot)$ concatenates an input item embedding with the corresponding high-level hidden state, $(\mathbf{x}_i | \forall q(i) = q(t), i < t)$ is the sequence of within-session item embeddings before time step t, $\mathbf{s}_{q(t)}$ is passed from the high-level model and propagates its gradients into the high-level model when being trained, and TCN$(\cdot)$ is a TCN described in Section 3.3.2.

To output a final recommendation, HierTCN uses dot product between user and item embeddings as the scoring function to compute the unnormalized distribution $p(\mathbf{x}_t | \mathbf{x}_{<t})$

$$p(\mathbf{x}_t | \mathbf{x}_{<t}) = \mathbf{x}_t^T \mathbf{u}_t \tag{12}$$

We then compute $p(\mathbf{x}_t | \mathbf{x}_{<t})$ over the items in a candidate pool and rank the items in descending order, and output the top k items as the final recommendations. The overall architecture of HierTCN is visualized in Figure 1.

3.5.2 Objective function. In this section, we discuss the possible objective functions than can be used to train HierTCN when being applied to recommender systems. The performances of differ objective functions are compared in Section 4.4.1 When items can be represented as real-value vectors, the following objective functions are commonly used.

- **L2 loss** The simplest objective function would be minimizing the L2 distance between the user embedding and the interacted item embedding:

$$\min ||\mathbf{x}_t - \mathbf{u}_t||_2 \tag{13}$$

- **Noise Contrastive Estimation (NCE)** This objective is first proposed by [8], and is popularized by the Word2Vec paper [20]. This objective function makes use of the negative samples. Specifically, the objective function encourages the user embedding to be similar to the positive item embedding, while enforcing the user embedding to be different from the negative items' embeddings. NCE employs the following formulation:

$$\min -\log(\sigma(\mathbf{x}_t^T \mathbf{u}_t)) - \sum_i -\log(\sigma(-\mathbf{c}_{ti}^T \mathbf{u}_t)) \tag{14}$$

- **Bayesian personalized ranking (BPR)** [24] proposes using the following BPR objective function, which can be interpreted as maximizing the posterior estimator derived from a Bayesian analysis of the item recommendation problem:

$$\min -\sum_i \log(\sigma(\mathbf{c}_{ti}^T \mathbf{u}_t - \mathbf{x}_t^T \mathbf{u}_t)) \tag{15}$$

- **Hinge loss** Hinge loss is based on the idea of max margin learning [30], which has the following form:

$$\min \sum_i \max \left\{0, \delta + \mathbf{c}_{ti}^T \mathbf{u}_t - \mathbf{x}_t^T \mathbf{u}_t\right\} \tag{16}$$

When items do not come with features and thus are represented as one-hot vectors, the following cross entropy loss can be used:

- **Cross entropy loss** When $\mathbf{x}_t$ are one-hot vectors, cross entropy loss can be written as

$$\min -\mathbf{x}_t^T \log(\mathbf{u}_t) \tag{17}$$

3.5.3 Regularization and Normalization. There are many techniques to facilitate model convergence and prevent overfitting, and we adopt the dropout [29] and batch normalization [15] in the HierTCN model for both TCN and GRU modules. These techniques are not widely adopted for sequence models, and we provide a thorough discussion in this section.

Dropout. Dropout can be easily implemented for TCN by randomly dropping activations for the subsequent layers. For RNN, normally dropout is applied at the input, the output and the state transition function [1]. However, the benefits of applying dropout for state transition function are in fact doubtful because early state information can be wiped out by an exponential factor over the sequence length n. For recommender systems where users usually generate long interaction sequences, this approach is especially undesirable.

We use the following approach [27] to applying dropout for GRU transition. Specifically, dropout is only applied to the candidate activation $\mathbf{h}_t$, and only the last equation in Equation 3 is modified into $\mathbf{s}_t = (1 - \mathbf{g}_t) \odot D(\mathbf{h}_t) + \mathbf{g}_t \odot \mathbf{s}_{t-1}$, where $D(\cdot)$ is the dropout function. This technique does not wipe out information from the last time step, while still adding randomness to the state transition to prevent overfitting.

Batch normalization. When carrying out batch normalization, we find it necessary to keep track of the average statistics over each time step independently [5]. For dynamic recommender systems, since different users can have diverse sequence lengths, a common technique is to zero pad all the sequences to the same length. When updating the average statistics, it is important to mask out zero padded values in the batch normalization layer, in order to prevent it from using these meaningless values to update the statistics.

3.6 HierTCN for Real-world Recommender Systems

In this section, we discuss how to apply HierTCN to build a real-world dynamic recommender system. Specifically, we propose a general framework that allows for a joint update of user and item embeddings, and discuss some key techniques to ensure efficient model training and inference.

3.6.1 Joint modeling of user and item embeddings. When applying HierTCN for recommender systems, we propose two separate update mechanism: (1) User interests change rapidly and thus their embeddings get updated in real-time. (2) Item information changes more slowly and thus we update item embeddings on a daily schedule. This choice turns out to be very important in practice. Existing approaches that learn latent variables over all the interactions in a global chronological order between all the users and items [6] cannot scale to billions of interactions that are observed in a real-world recommender system. Such models become computationally prohibitive and infeasible for large datasets.

We side-step this problem by designing a two-phase update scheme. In the first phase, item embeddings are generated. The second phase considers the item embeddings fixed and only generates user embeddings.

Update of item embeddings. In the first phase, we first build an item graph based on user-item interactions, where each node represents an item and an edge is built if two items are interacted by the same user within a short period of time. Then, we compute the node embedding for each item, which integrates both visual and textual features as well as the structural features of the item graph using graph convolutional neural networks (GCN) based on localized graph convolutional layers [36], whose computation can be paralleled on a distributed system. Specifically, each node is initialized with the concatenated visual and textual embeddings of the item. We then apply 2 layers of localized graph convolutional layers to compute the node embedding for a node u, where the l^{th} layer can be written as

$$\begin{aligned} \mathbf{n}_u^{(l)} &= \text{AGG}(\text{ReLU}(\mathbf{Q}^{(l)}\mathbf{z}_v^{(l)} + \mathbf{q}^{(l)} | v \in N(u))) \\ \mathbf{z}_u^{(l+1)} &= \text{ReLU}(\mathbf{W}^{(l)}\text{CONCAT}(\mathbf{z}_u^{(l)}, \mathbf{n}_u^{(l)})) \end{aligned} \tag{18}$$

where $\mathbf{z}_u^{(l)}$ is the l^{th} level node embedding for node u, $N(u)$ is the local neighborhood of u, $\text{AGG}(\cdot)$ is an order invariant aggregation function such as $\text{MEAN}(\cdot)$ pooling, and $\mathbf{Q}^{(l)}, \mathbf{q}^{(l)}, \mathbf{W}^{(l)}$ are trainable parameters. After gathering new user-item interactions, we update the item graph and recompute the node embeddings for each item.

Update of user embeddings. In the second phase, we keep the item embeddings fixed and train the HierTCN model using the past interactions of users. Once the model is trained, any new within-session or across-session activity leads to a relatively fast update of the HierTCN model. Model can then be used in parallel to compute user embeddings in real time.

Training user and item models. The GCN model is trained by minimizing the following NCE-based objective function

$$-\log(\sigma(\mathbf{z}_u^T\mathbf{z}_v)) - C\mathbb{E}_{v_n \sim P_n(u)}[\log(\sigma(-\mathbf{z}_u^T\mathbf{z}_{v_n}))], \forall v \in N(u) \tag{19}$$

where $P_n(u)$ is a negative sampling distribution for node u and C is the number of negative samples, while HierTCN is trained with the hinge loss described in Section 3.5.2. GCN model is trained with about 7.5 billion samples and the HierTCN model is trained with about 1.7 billion samples, more details of training the GCN model are described in [36] and HierTCN training techniques are discussed in 3.6.2.

Overall, this two-phase approach enables the dynamic modeling of both user and item embeddings, which can be computationally cost-prohibitive otherwise. We conduct extensive offline evaluation on a subset of 6 million Pinterest users and present our results in Section 4.

Table 1: Statistics of the datasets (mean|std. deviation). Note that XING dataset does not have the impression data.

Dataset	XING	Pinterest
Users	65,347	5,923,659
Items	20,778	74,202,787
Interactions	1,450,300	56,050,857
Impressions	-	1,685,877,684
Sessions	535,747	23,354,523
Impressions per event	-	9.8|8.6
Impressions per session	-	25.8|51.1
Events per session	2.5|3.8	2.4|3.0
Sessions per user	8.2|7.0	3.9|4.7
Events per user	22.2|20.8	9.7|13.4
Impression per user	-	284.6 | 252.0

3.6.2 Efficient offline training of HierTCN. HierTCN is trained offline with about 6 million users and 1.7 billion training examples (Table 1). To conduct the training, we design an efficient offline training pipeline for HierTCN, which includes the following key components.

Item embedding cache. The item embeddings are represented as 512 dimensional float vectors, thus directly storing 1.7 billion interactions requires tens of terabytes of storage and even loading all the data to memory takes a day. To speed up the data pipeline, we first assign a universal ID to each of the 74 million items, and only keep item IDs when saving the user interaction data. Then we employ Linux huge page table [17] to cache all the item embeddings in a fixed chunk of memory. Finally, a sequence of item embeddings can be fetched by a direct look-up in the page table with a negligible query time. This approach enables us to load the item embeddings once; then, any subsequent models can share the embedding cache without the need to reload/rebuild the cache. In practice, this approach can reduce the data loading time from a day to a minute when launching a sequence model. Note that in the online deployment setting, the item embedding cache is rebuilt daily.

Queue-based mini-batch generator. To train a HierTCN, a mini-batch of training data should consist of the same number of sessions. In practice, the number of sessions that a user has can range from 1 to several hundreds; thus, the common approach of doing zero padding will cause huge amount of unnecessary data memory consumption. To address the problem, we design a queue-based mini-batch generator to significantly reduce memory consumption and speed up training, based on the data loading scheme proposed in [23]. The mini-batch generator first initializes B data queues where

B equals the batch size. Then an enqueue process loads a new user's sequence data and parses them by sessions, and enqueues the sessions of data along with the user ID to the queue with the least number of sessions. At the same time, a dequeue process takes a number of sessions from each queue, with each session zero-padded to the same length, then concatenates the sessions into batches and feeds a batch of data into HierTCN. During computation, HierTCN model constantly checks whether the user ID changes after a session; if so, the model will reset the hidden state of the high-level GRU to ensure each new user's state is properly initialized. The proposed mini-batch generator also naturally conducts the truncated backpropagation technique [33], a key technique that ensures stable training of RNN over long sequences.

3.6.3 Real-time inference. For real-time inference, the recommender system consists of a mini-batch generator that produces mini-batches and a trained HierTCN model that consumes the mini-batches. The mini-batch generator can be paralleled by creating multiple enqueue and dequeue processes, while the computation of HierTCN model can be paralleled by deploying multiple copies of the trained model, in order to achieve fast real-time computation. Two of the key techniques are discussed below.

User hidden state cache. A real-time dynamic recommender system requires efficient update of users' hidden states. Similar to item embedding cache, we cache all the users' hidden states into the memory, and maintain a dictionary that maps a user to the corresponding hidden state. Whenever a user starts a session, the system will read the hidden state of the user. After a session is closed, the hidden state of a user is updated with the high-level GRU, and is written to the original memory address.

Online queue-based mini-batch generator. We extend the offline mini-batch generator into the online settings. Specifically, after a user has interacted with an item, an enqueue process will fetch a session of data that contains the user's past interactions within the session. In addition, the enqueue process reads the cached hidden state of the user and concatenates the hidden state vector to all the interactions within the session. Finally, the process enqueues the session of data. At the same time, a dequeue process dequeues a batch of sessions with each session zero-padded to the same length and feeds them to HierTCN model. The process also retrieves the recommendation results and sends the results to the web API, while writing the updated hidden states to the cache.

4 EXPERIMENTS

4.1 Datasets

We use a public XING dataset and a large-scale private Pinterest dataset. The statistics of the datasets are summarized in Table 1. The aim of experimenting on a small-scale public dataset is to demonstrate that HierTCN can achieve transferable and robust performance, while rigorous and extensive evaluations are done over the large-scale Pinterest dataset.

The public XING dataset is extracted from XING Recsys Challenge 2016 dataset [3], with about 11 thousand users and 500 thousand interactions. The items do not come with features thus we represent each item as a one-hot vector. The dataset also does not have session information, thus we manually partition the interactions using a 30-minute idle threshold. Following the prior work [23], we remove interactions with type "delete" and do not consider the interaction types in the data. We remove items with less than 50 interactions and users with less than 10 or more than 1000 interactions.

The private Pinterest dataset is an internal dataset from Pinterest, with 6 million users, 56 million interactions and 1.7 billion impressions over 3 months. We clean the dataset by removing users with less than 10 or more than 1000 interactions. We represent items using a Pinterest internal item graph based on user-item interactions, with 3 billion nodes and 18 billion edges. Specifically, each item is represented as a node, and we connect two items if an user has interacted with both of them within a very short period of time. Each item is initialized with 4096 dimensional visual features, extracted from the 6-th fully connected layer of an image classification network using VGG-16 architecture [28], and 256 dimensional textual annotation features trained with Word2Vec [20]. A 2-layer GraphSage model [9] is then trained over the item graph using hinge loss function to differentiate positive and negative item samples. Finally, each item is represented by a 512 dimensional node embedding vector. More details about learning the item embeddings are discussed in [36].

4.2 Experimental Setup

4.2.1 Experimental setting. We consider two types of experiments. The first is *cold-start recommendations*. We split all the data by user, and select 80% of the users to train the model, 10% to tune the hyper-parameters of the model and test on the remaining 10% users. The second is *warm-start recommendations*. We select a fixed set of users, then train the model on the first two months of data and test on data in the following month. All the hyper-parameters remain the same with the cold-start recommendation settings.

All the deep learning based models are trained with Adam optimizer, with learning rate 0.001 and batch size 32. We stop training the model when the validation error plateaus. We find that weight normalization [25] does not help with the tasks we are experimenting, while batch normalization [15] and dropout [29] is helpful for the smaller XING dataset. For the large-scale Pinterest dataset, the vanilla models without regularization perform well.

4.2.2 Evaluation metrics. Our primary goal is to predict the user interaction in the next time step, which is evaluated by ranking the ground truth interaction against a pool of candidates at the given time step. For the XING dataset, the candidate pool is the set of all the items, while for the Pinterest dataset, the candidate pool is the impression data at the evaluated time step, which is given by a separate production system and cannot be altered by the model.

For models that can directly produce the ranking of an item (e.g., Maximum item similarity (MaxItem)), we directly use the predicted ranking for evaluation. For models that can output specific user embeddings, we compute the probability $p(\mathbf{x}_t|\mathbf{x}_{<t})$ over the candidate pool and rank the probability in descending order. We repeat the same process over all the interactions for all the users in the test data, and report various ranking statistics that are listed below.

Table 2: Performance of cold-start recommendation on the XING dataset and the large-scale Pinterest dataset.

Dataset	Metric	**HierTCN**	HierGRU	HRNN [23]	TCN	GRU	MV	MaxItem
XING	Recall@10 (Higher is better)	**0.139**	0.105	0.113	0.108	0.124	0.107	-
	MRR (Higher is better)	**0.071**	0.037	0.040	0.038	0.043	-	-
	MRP (Lower is better)	**0.121**	0.161	0.149	0.198	0.141	-	-
Pinterest	Recall@1	**0.206**	0.184	0.174	0.194	0.195	0.1658	0.1487
	Recall@5	**0.663**	0.639	0.619	0.653	0.655	0.6120	0.5843
	Recall@10	**0.855**	0.839	0.828	0.850	0.852	0.8173	0.7921
	MRR	**0.402**	0.380	0.366	0.391	0.392	0.3593	0.3400
	MRP	**0.304**	0.326	0.338	0.313	0.311	0.3484	0.3724

Table 3: Performance of warm-start recommendation on the large-scale Pinterest dataset.

Dataset	Metric	**HierTCN**	HRNN [23]	TCN	MV	MaxItem
Pinterest	Recall@1	**0.202**	0.188	0.183	0.174	0.155
	Recall@5	**0.667**	0.650	0.641	0.631	0.600
	Recall@10	**0.855**	0.848	0.838	0.830	0.801
	MRR	**0.399**	0.386	0.379	0.370	0.348
	MRP	**0.313**	0.324	0.336	0.346	0.374

- **Recall@K**. Recall@K reports the proportion of times that the ground truth interacted item is ranked within the top K list of recommendations. Higher Recall@K is better.
- **Mean Reciprocal Rank (MRR)**. This is a standard metric for evaluating recommender systems, which is the average reciprocal rank for the items that a user actually interacts with. Higher MRR is better.
- **Mean Rank Percentile (MRP)**. We divide the rank of the ground truth interaction by the size of the candidate pool, and average over all test cases. This is useful for Pinterest dataset where the size of candidate pool vary over different interactions. Lower MRP is better.

4.2.3 Baseline methods. We compare HierTCN with a variety of baseline methods, which are summarized below. To ensure a fair comparison, for all deep learning based model, we adjust the layer number and hidden units number such that all the models have very similar number of trainable parameters.

Rule-based models. Rule-based models maintain a pool $\mathbf{P}_t$ that consists of the past k interacted item embeddings t for each user.

- **Moving average (MV)**. The moving average model outputs the average of all the item embeddings in $\mathbf{P}_t$.
- **Maximum item similarity (MaxItem)** Rather than output a prediction, the MaxItem model can only evaluate a given candidate item embedding. An evaluation score is calculated as the highest dot product value between a candidate item embedding and embeddings in $\mathbf{P}_t$, which is then used for ranking the candidates.

Single-level sequence models. Single-level sequence models take sequences of user interactions without being partitioned into sessions. To make a fair comparison, we include a session indicator, marking the start of a session, as a input feature. Specifically, we consider the following two sequence models:

- **TCN** We implement a TCN model with 6 blocks of convolutional layers, where each block consists of 2 temporal convolutional layers that have 128 filters with size 5. The dilation factor of each block is set to grow exponentially with the number of the block, which is 1,2,4,8,16,32 in our scenario. We also add residual connections [10] between each block to facilitate training the model.
- **GRU** We construct a GRU model by stacking 4 layers of GRU cells described in Equation 3, each with 200 dimensional hidden state.

Hierarchical sequence models. We also compare HierTCN with the state-of-the-art hierarchical deep learning model, as well as a variant which we refer to as HierGRU.

- **HRNN** We implement a hierarchical GRU model following the paper [23]. Both high-level and low-level model are implemented with 4 layers of GRU cells, each with 128 dimensional hidden state. The high-level GRU's hidden state is used to initialized the low-level GRU, while the final hidden state of the low-level GRU is used to update the high-level GRU.
- **HierGRU** We implement a baseline version of HierTCN, where the only difference is that we change the low-level model from TCN to GRU. Both the high-level GRU and the low-level GRU has 4 layers of GRU cells, each with 128 dimensional hidden state.
- **HierTCN** This is our proposed model. The high-level model is a 4-layer GRU, each layer has 128 dimensional hidden state. The low-level model is a TCN with 4 blocks of convolutional layers, and the other settings are the same as a single-level TCN.

4.3 Experimental results

4.3.1 Results on Public XING dataset. Table 2 summarize the performance of all the models in XING dataset. Since items are represented as one-hot vectors, MV model cannot rank properly for unseen items thus only Recall@10 is reported. For the same reason, MaxItem is not implemented as the dot product between one-hot vectors is not meaningful. Our model significantly outperforms all the baseline methods by a large margin, and achieves 30.4% average performance improvement over the best baseline method (GRU). We observe that hierarchical GRU models converge very slowly,

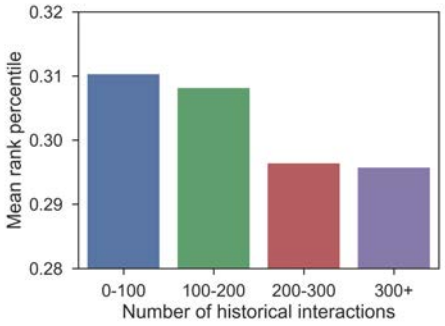


Figure 2: Summary of HierTCN's performance over users with different number of historical interactions. Lower is better.

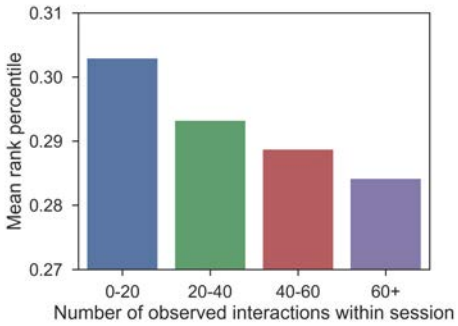


Figure 3: Summary of HierTCN's performance when observing different number of interactions within a session.

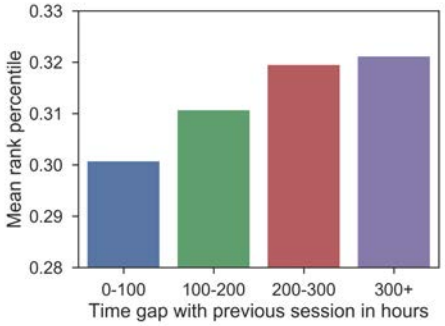


Figure 4: Summary of HierTCN's performance over sessions that have different time gap with previous sessions.

which may result from the complex gradient propagation path. In this task, all the models directly predict the probability of each item over all possible items, resulting an output dimension of over 10 thousand. We train the models with the simplest cross entropy loss, and people can use a more complex loss function including [23] to achieve better performance. The aim here is not to achieve the state-of-the-art performance on the XING dataset; rather, the results are used to show that the proposed HierTCN model can achieve more robust performance compared with baseline methods even in this simplified setting.

Table 4: The total time to finish one epoch training for the Pinterest dataset (in seconds).

Dataset	Model	Training
Pinterest	**HierTCN**	**295.3s**
	HierGRU	747.8s
	HRNN [23]	693.1s
	TCN	**254.0s**
	GRU	776.8s

Table 5: Cold-start performance of HierTCN with different objective functions. 10% of the training data are used for the experiment.

Dataset	Metric	L2	NCE	BPR	Hinge
Pinterest	Recall@1	0.167	0.192	0.183	**0.201**
	Recall@5	0.611	0.642	0.632	**0.657**
	Recall@10	0.817	0.838	0.832	**0.851**
	MRR	0.360	0.386	0.377	**0.396**
	MRP	0.349	0.322	0.331	**0.309**

4.3.2 Results on Large-scale Pinterest dataset. Table 2 and Table 3 summarize the performance of all the models in the large-scale Pinterest dataset. For cold-start evaluation, HierTCN outperforms the best hierarchical baseline by 18% in Recall@1, 10% in MRR. The gain over the best single-level sequence model is 6% in Recall@1, 3% in MRR. The gain over the best rule-based model is 24% in Recall@1, 13% in MRR. From the results, we can see that deep learning approaches perform much better than rule-based models, since users are possible to interact with hundreds of millions of items in Pinterest, and thus rule-based models are incapable of capturing this complex dynamics. Among deep learning approaches, it is interesting to see that hierarchical GRU models perform worse than single-level sequence models; empirically, we find that hierarchical GRU models converge slowly when we use the same learning rate as other models (0.001), while experiencing performance oscillation when we try to increase the learning rate. On the contrary, HierTCN consistently outperforms the single-level sequence models, and we do not observe issues for optimizing the model. This indicates that the hierarchical structure does capture more aspects of user interests. For warm-start evaluation, HierTCN also significantly outperforms the baseline models, with on average 12% improvement in Recall@1 and 5% improvement in MRR, and we can find similar observations from the results.

4.3.3 Running time. Table 4 summarizes the running time of different methods. We report the running time to finish one epoch training for all the deep learning models. We observe that models that make use of TCN are roughly 2.5 times faster than GRU-based models, which supports the argument that TCN is generally much faster than GRU. Moreover, adding the hierarchical structure in HierTCN only slightly affects the computation speed, owing to the fact that we only update the high-level model of HierTCN after each session and we use an efficient mini-batch generator. The efficient mini-batch generator makes the training of hierarchical GRU models even slightly faster than the single-level GRU model.

Table 6: Effects of adding different regularizations to HierTCN. BN and Drop stands for batch normalization and dropout.

Dataset	Regularization	None	Drop	BN	Drop + BN
XING	Recall@10	0.139	0.129	0.145	**0.147**
	MRR	0.071	0.066	0.073	**0.075**
	MRP	0.121	0.120	0.109	**0.107**

4.3.4 GPU memory consumption. GPU memory consumption consists of the memory to store model parameters, which is the same for all models in our experiments, and the memory to store the input data. TCN model takes a huge amount of memory to store the input data, because it has to keep the entire historical sequence to make a prediction. In contrast, HierTCN only needs to store the input sequence at each session. The specific reduction of memory consumption varies with the number of sessions that a user has, and we observe up to 90% reduction in the experiments.

4.4 Performance analysis

4.4.1 Choice of objective functions. Table 5 summarizes the results of using different loss functions for HierTCN on the Pinterest dataset. We conduct experiments in the cold-start setting and use only 10% of all the training data for this objective function comparison, thus the performance scores are worse than Table 2. We observe that using hinge loss provides about 5% improvement of Recall@1 and 3% improvement of MRR over the best competing loss function (NCE). In addition, using negative samples significantly improves the performance, and we observe 20% improvement of Recall@1 and 10% improvement of MRR over L2 loss.

4.4.2 Effects of dropout and batch normalization. To make fair comparison with baseline methods, we do not add dropout or batch normalization in the experiments. We conduct further experiments to examine the effects of adding dropout and batch normalization. From Table 6, we can see that adding dropout out alone does not improve the model performance, while adding batch normalization do help. When combining both techniques, there is further performance improvement. In addition, when adding batch normalization, we observe significant faster convergence; however, the model eventually overfits, and thus doing early stopping is necessary.

4.4.3 Model behavior analysis. We conduct further analyses to understand the performance of HierTCN under different scenarios. We select MRP as the metric and lower is better. To examine the overall performance of HierTCN, we summarize the performance of HierTCN over users with different historical interactions in Figure 2. It is clearly shown that HierTCN's performance increases as more interactions are observed. We analyze the performance of the low-level model by summarizing HierTCN's performance over different number of observed interactions within a session. Figure 3 shows that as more interactions are observed within a session, the performance of HierTCN gets better. Finally, we examine the performance of the high-level model by summarizing the performance of sessions that have different time gaps with previous sessions. Figure 3 shows that as the session's time gap increases,

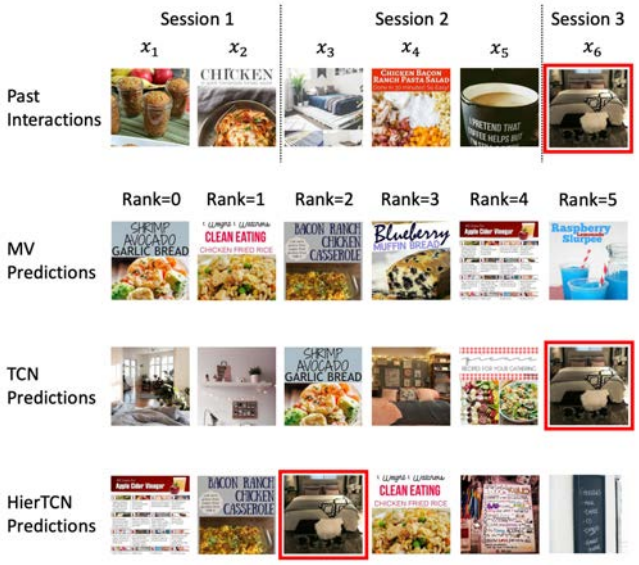


Figure 5: Recommendations given by different models. The first row shows the past interactions of a user, and the following 3 rows show the top-6 recommendations at time step 6 made by different models. The image marked with a red box is the ground-truth interaction.

a user's behavior becomes less predictable and the performance of HierTCN decreases. In summary, these analyses show that all the components of HierTCN perform reasonably under different scenarios.

4.4.4 Visualization of recommendations. Finally, we select a typical example to illustrate the superior performance of HierTCN. The first row in Figure 5 shows the past interactions of a user, which includes items related to food and bed. The following three rows illustrate the performance given by a rule-based model, a single-level sequence model and HierTCN. From the results we can see that rule-based model recommends all food-related items because they are prevalent in the past interactions, TCN exaggerates the existence of bed-related item and recommends too many items related to furniture, and HierTCN reaches a balance between recommending both types of items and achieves better performance.

5 CONCLUSION

In this paper, we have proposed Hierarchical Temporal Convolutional Networks for real-time large-scale recommender systems. By designing a novel hierarchical model using RNN and TCN, our model efficiently captures different levels of user interests and combines RNN's benefits of maintaining long-term hidden states and TCN's ability of conducting efficient and effective computation. We proposed a framework for large-scale dynamic recommender systems and applied HierTCN to a real-world dataset that contains millions of users and billions of activities. Compared with the state-of-the-art methods, HierTCN achieves superior performance and is much more scalable.

6 ACKNOWLEDGEMENT

The authors thank Kaifeng Chen, Ruining He, and Fei Liu for their help on idea discussions and data preparation.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16.

[2] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271* (2018).

[3] RecSys Challenge. 2016. RecSys Challenge 2016. In *Proceedings of the Recommender Systems Challenge*. ACM.

[4] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[5] Tim Cooijmans, Nicolas Ballas, César Laurent, Çağlar Gülçehre, and Aaron Courville. 2016. Recurrent batch normalization. *arXiv preprint arXiv:1603.09025* (2016).

[6] Hanjun Dai, Yichen Wang, Rakshit Trivedi, and Le Song. 2016. Deep coevolutionary network: Embedding user and item features for recommendation. *arXiv preprint arXiv:1609.03675* (2016).

[7] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for ctr prediction. *IJCAI* (2017).

[8] Michael U Gutmann and Aapo Hyvärinen. 2012. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research* 13, Feb (2012), 307–361.

[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee.

[12] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[14] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative metric learning. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee.

[15] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[16] Dietmar Jannach and Malte Ludewig. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM.

[17] Christoph Lameter. 2006. Local and remote memory: Memory in a Linux/NUMA system. In *Linux Symposium*.

[18] Yann LeCun, Yoshua Bengio, et al. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks* 3361, 10 (1995), 1995.

[19] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*.

[21] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499* (2016).

[22] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. 2016. Pixel recurrent neural networks. *International Conference on Machine Learning* (2016).

[23] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM.

[24] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[25] Tim Salimans and Diederik P Kingma. 2016. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*.

[26] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. AutoRec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*. ACM.

[27] Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. 2016. Recurrent dropout without memory loss. *arXiv preprint arXiv:1603.05118* (2016).

[28] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[29] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[30] Johan AK Suykens and Joos Vandewalle. 1999. Least squares support vector machine classifiers. *Neural processing letters* 9, 3 (1999), 293–300.

[31] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *11th ACM International Conference on Web Search and Data Mining*. ACM.

[32] Yichen Wang, Nan Du, Rakshit Trivedi, and Le Song. 2016. Coevolutionary latent feature processes for continuous-time user-item interactions. In *Advances in Neural Information Processing Systems*. 4547–4555.

[33] Ronald J Williams and Jing Peng. 1990. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural computation* 2, 4 (1990), 490–501.

[34] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent recommender networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM.

[35] Sai Wu, Weichao Ren, Chengchao Yu, Gang Chen, Dongxiang Zhang, and Jingbo Zhu. 2016. Personal recommendation using deep recurrent neural networks in NetEase. In *32nd International Conference on Data Engineering*. IEEE.

[36] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2018).

[37] Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. 2018. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International Conference on Machine Learning*. 5694–5703.

[38] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2018. A Simple but Hard-to-Beat Baseline for Session-based Recommendations. *arXiv preprint arXiv:1808.05163* (2018).

Spectrum-enhanced Pairwise Learning to Rank

Wenhui Yu
Tsinghua University
Beijing, China
yuwh16@mails.tsinghua.edu.cn

Zheng Qin*
Tsinghua University
Beijing, China
qingzh@mail.tsinghua.edu.cn

ABSTRACT

To enhance the performance of the recommender system, side information is extensively explored with various features (e.g., visual features and textual features). However, there are some demerits of side information: (1) the extra data is not always available in all recommendation tasks; (2) it is only for items, there is seldom high-level feature describing users. To address these gaps, we introduce the *spectral features* extracted from two hypergraph structures of the purchase records. Spectral features describe the *similarity* of users/items in the graph space, which is critical for recommendation. We leverage spectral features to model the users' preference and items' properties by incorporating them into a Matrix Factorization (MF) model.

In addition to modeling, we also use spectral features to optimize. Bayesian Personalized Ranking (BPR) is extensively leveraged to optimize models in *implicit feedback* data. However, in BPR, all missing values are regarded as negative samples equally while many of them are indeed unseen positive ones. We enrich the positive samples by calculating the similarity among users/items by the spectral features. The key ideas are: (1) similar users shall have similar preference on the same item; (2) a user shall have similar perception on similar items. Extensive experiments on two real-world datasets demonstrate the usefulness of the spectral features and the effectiveness of our spectrum-enhanced pairwise optimization. Our models outperform several state-of-the-art models significantly.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; **Social recommendation**; **Recommender systems**; • **Human-centered computing** → *Social recommendation.*

KEYWORDS

Collaborative filtering, spectral feature, pairwise learning to rank, latent community, latent category.

ACM Reference Format:
Wenhui Yu and Zheng Qin. 2019. Spectrum-enhanced Pairwise Learning to Rank. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313478

*The corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313478

1 INTRODUCTION

Recommender systems have been widely used in online services such as E-commerce and social media sites to predict users' preference based on their interaction histories. Modern recommender systems uncover the underlying latent factors that encode the preference of users and properties of items. In recent years, to strengthen the presentation ability of the models, various features are incorporated for additional information, such as visual features from product images [14, 44, 48], textual features from review data [8, 9], and auditory features from music [5]. However, these features are not generally applicable, for example, visual features can only be used in product recommendation, while not in music recommendation. Also, these features are unavailable in some recommendation tasks, such as point-of-interest recommendation [43, 45]. Even they are available, we need extra efforts to collect and process them. Moreover, we can only get features for items while there is seldom high-level feature for users. To address these gaps, we introduce spectral features extracted from the purchase records for both items and users in this paper. We then use the spectral features to (1) model users' preference and items' properties and to (2) optimize the proposed model.

Recommender systems predict the missing value by uncovering the similarity of users/items, thus the information of similarity is vital in recommendation tasks. For example, in user-/item-based collaborative filtering, we recommend by calculating the similarity among users/items [12, 39]; in model-based collaborative filtering, the low-rank form ensures the linear dependence of latent factors, i.e., the similarity among users/items [1, 22, 35]. Inspired by this, we propose a new feature that contains the similarity information. We define the front K eigenvectors of a Laplacian matrix as the spectral feature. Devised for spectral clustering [29], the spectral feature describes the distance among vertices in a graph space thus contains abundant information of similarity. The spectral feature extracts information from the purchase history rather than additional data, thus it is generally applicable and works in almost all situations. Also, we can extract spectral features for both users and items.

We define the set of users near in the graph space as a *latent community*, which means they have similar purchase behaviors. For an item preferred by a specific user, it may get a high score for her latent community members. Also, the set of items near in the graph space is defined as a *latent category*, items in the same latent category have similar properties. Users who like certain item may have interests in the latent category the item belongs to. Of special notice is that a latent category is not like a real category, items in it may be relevant items, similar items, etc., in a word, items strongly connected in the graph. In this paper, we incorporate the spectral features into a Matrix Factorization (MF) model [22, 37] to propose our **S**pectrum-enhanced **C**ollaborative **F**iltering (**SCF**) model. By

learning spectral dimensions, SCF uncovers users' preference for latent categories and items' fitness for latent communities.

Compared with other side information features, our spectral feature is more suitable for the *implicit feedback* data. Implicit feedback data is like "purchase" or "browse" in E-commerce sites, "like" in social media sites, "click" in advertisements, etc. In real-world applications, the data of user behaviours in "one-class" (implicit feedback) is easier to collect and more generally applicable than "multi-class" scores (*explicit feedback*). The spectral features, which are independent of extra data, are more suitable for the implicit feedback data, since they maintain the advantages of easy to collect and generality.

Besides providing information of similarity, we also use the spectral features to enhance the pairwise learning. When optimizing model on implicit feedback data, Bayesian Personalized Ranking (BPR) is widely used due to the outstanding performance [14, 37, 44]. It aims to maximize the likelihood of pairwise preference over positive samples and negative samples. However, there is a critical issue: All missing values are simply treated as negative samples in BPR. In fact, some of the missing values are indeed unlabelled positive samples, users may like them but just have not seen them yet. To deal with this issue, we cluster all items and users by spectral features to construct the latent categories and latent communities, respectively. We assume that a user shows stronger preference for items that are in the same latent category with what she purchased, or items that purchased by her neighbors in the same latent community, than other missing entries. Considering the enriched preference relationship, we propose an optimization method called **S**pectrum-enhanced **P**airwise **L**earning to **R**ank (**SPLR**), and optimize SCF with it. Finally, we validate the effectiveness of our proposed model by comparing it with several baselines on the *Amazon.Clothes* and *Amazon.Jewelry* datasets. Extensive experiments show that we improve the performance significantly by exploring spectral features.

Specifically, our main contributions are listed as follows:

- We leverage novel spectral features in recommendation tasks to capture the similarity information of users/items, and propose an SCF model by injecting these features to the MF model.
- We propose a spectral clustering-enhanced pairwise ranking method, SPLR, to optimize our model. We construct latent categories and latent communities to enrich the positive samples.
- We extend our model and propose a framework to use side information features for modeling and for optimization. We can explore any kinds of features in our framework, including spectral features and other conventional features. Multiple features can be utilized at a time.
- We devise comprehensive experiments on two real-world datasets to demonstrate the effectiveness of our proposed methods.

2 RELATED WORK

After Matrix Factorization (MF) is utilized to deal with recommendation tasks [1, 22, 37], modern recommender systems develop rapidly. MF models learn the latent factors of users and items by reconstructing the purchase records in a low-rank form. Latent factors represent the preference of users and properties of items. Many variants are proposed to promote the ability to model the complex preference that users exhibit toward items based on their past interactions. [15, 18] used deep structure to learn embedding. [7, 17] took time into account when making predictions. [19, 46] focused on fast algorithms for online recommendation.

2.1 Side Information Features

One important way to enhance the presentation ability is to leverage the side information. Theoretically, latent dimensions can capture all relevant factors, but they usually cannot in applications due to the sparsity of the datasets, thus extra information is desired. The visual features are widely used since users' decisions depend largely on products' appearance [14, 27, 44, 48]. [14, 27] predicted consumers' behavior with the CNN feature. Yu et al. [44] utilized the aesthetic feature to model users' aesthetic preference on clothes. Zhao et al. [48] leveraged several visual features to recommend movies. There are also many efforts exploring the textural feature to recommend [9, 26], McAuley and Leskovec [26] proposed models with the textural feature from the review data and Chen et al. [9] extracted the textural feature from time-synchronized comments for key-frame recommendation. In music recommendation, the auditory feature is generally used, Cao et al. [5] extracted the auditory feature by a pre-trained deep structure to recommend music.

Various features are used for different kinds of information. However, one feature, the spectral feature, has never been explored. In this paper, we extract the spectral features from the hypergraphs of users and items, which represent the similarity of vertices in the hypergraphs. Compared with features mentioned above, the spectral features are independent of additional information so they are suitable for more situations. Moreover, conventional features are all for items, while we extract spectral features for both users and items.

2.2 Graph-enhanced Recommendation

In recent years, hypergraph gains increasing attention in the recommendation domain [4, 23, 34]. Lierde and Chow [23] proposed a user-based collaborative filtering enhanced with the hypergraph: The similarity of the users is calculated with the hypergraph embedding. [4, 34] filtered the latent factors with hypergraph regularized term to smooth them. There are also some efforts promoting the recommendation performance with social networks [21, 42]. Walter et al. [42] calculated similarities with the social networks embedding for memory-based collaborative filtering. Jamali and Ester [21] calculated the latent factors of a user with its neighbors to give the prediction. Graph convolution networks are also widely used in recommendation tasks [2, 50]. Berg et al. [2] proposed a graph convolutional auto-encoder that learns the structural information of a graph for latent factors of users and items. Zheng et al. [50] constructed random walk laplacian matrix of the user-item bipartite graph and then introduced a deep spectral convolutional network to capture the user-item connectivity information.

In this paper, we introduce a new way to explore graph structures in recommendation tasks. We extract features from the Laplacian matrix of the hypergraphs of users and items, and make prediction

with the MF term jointly. We also utilize these features to optimize our proposed method.

2.3 Learning to Rank

As implicit feedback data is easier to collect, it is extensively used in real-world application. However, prediction on implicit feedback dataset is a challenging task because there are only positive samples and unobserved samples. We cannot discriminate negative samples and unlabeled positive samples from the unobserved ones. Rendle et al. [37] treated all unobserved samples as negative ones when sampling while some of them are indeed unlabelled positive samples. Users may like them but just have not seen them yet. To address this gap, many works improved the pairwise learning to rank method. It is assumed that all users are independent in BPR, Pan and Chen [30] tried to relax this constraint and proposed a method called group preference-based Bayesian personalized ranking (GBPR), which modeled the preference of user groups. Qiu et al. [33] constructed the preference chain of item groups for each user. Liu et al. [24] utilized collaborative information mined from the interactions between users and items. [36, 47] proposed dynamic negative sampling strategies to maximize the utility of a gradient step by choosing "difficult" negative samples. [10, 31] used view information to enrich positive samples. [6, 25] proposed listwise ranking methods instead of pairwise ones. Hwang et al. [20] utilized both implicit and explicit feedback data to improve the quality of negative sampling.

In existing efforts, only low-order connections are considered when measuring the similarity among vertices [24, 30, 33]. In this paper, we use the spectral features, which contains the information of high-order connections, to enhance the pairwise learning. We cluster all items/users by spectral features to construct latent categories/communities. For vertices (users/items) in the same cluster, they are strongly connected[1] in the hypergraph thus are very similar to each other. For each user, we regard items in the same latent category with her positive samples and items purchased by her latent community members as the potential samples, and assume that the user prefers them than other negative samples.

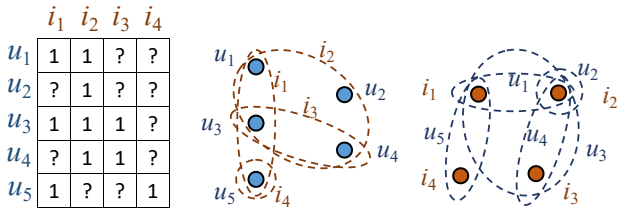
3 SPECTRUM-ENHANCED COLLABORATIVE FILTERING

In this section, we propose a novel recommendation model called Spectrum-enhanced Collaborative Filtering (SCF). We first introduce the spectral features and then inject them into an MF model. In this paper, bold uppercase letters refer to matrices. For example, $\mathbf{A}$ is a matrix, $\mathbf{A}_i$ is the i-th row of $\mathbf{A}$, $\mathbf{A}_{*,j}$ is the j-th column of $\mathbf{A}$, and $\mathbf{A}_{ij}$ is the value at the i-th row and the j-th column of $\mathbf{A}$. $\mathbf{A}^U$ is the matrix for user and $\mathbf{A}^I$ is the matrix for item, $\mathbf{A}^{(k)}$ is the k-th matrix.

3.1 Spectral Feature

In a recommendation task, we use matrix $\mathbf{R} \in \mathbb{R}^{N\times M}$ to denote the interactions between users and items (there are N users and M items in total). $\mathbf{R}_{ui} = 1$ if user u purchased items i and $\mathbf{R}_{ui} = 0$ otherwise. Our task is to predict the missing values (0 in $\mathbf{R}$) to recommend top-n items for each user.

The hypergraph is a generalization of the simple graph, where an hyperedge (edge in hypergraph) connects any number of vertices rather than just two. A hypergraph is usually represented with the incidence matrix $\mathbf{H} \in \mathbb{R}^{N\times M}$ (there are N vertices and M hyperedges). Each row in $\mathbf{H}$ is for a vertex and each column is for a hyperedge. $\mathbf{H}_{ij} = 1$ if vertex i is connected by hyperedge j and $\mathbf{H}_{ij} = 0$ otherwise. We can see that a hypergraph can represent the purchase records by nature. We define two hypergraph structures, a user hypergraph and an item hypergraph. In the user hypergraph, users are vertices and items are hyperedges while in the item hypergraph, items are vertices and users are hyperedges.



(a) Interaction Matrix (b) User Hypergraph (c) Item Hypergraph

Figure 1: An example for user hypergraph and item hypergraph. For the user hypergraph, the incidence matrix $\mathbf{H}^U = \mathbf{R}$ and for the item hypergraph, $\mathbf{H}^I = \mathbf{R}^\top$.

For a hypergraph represented with the incidence matrix $\mathbf{H}$, the Laplacian matrix $\mathbf{L}$ is defined as [51]:

$$\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{D} - \mathbf{H}\mathbf{W}\mathbf{\Delta}^{-1}\mathbf{H}^\top)\mathbf{D}^{-\frac{1}{2}}, \tag{1}$$

where the diagonal matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$ denotes the degrees of vertices, the diagonal matrix $\mathbf{W} \in \mathbb{R}^{M\times M}$ denotes the weights of hyperedges, and the diagonal matrix $\mathbf{\Delta} \in \mathbb{R}^{M\times M}$ denotes the degrees of hyperedges. Laplacian matrix is a difference operator of a hypergraph [28], for any signal $\mathbf{S} \in \mathbb{R}^{N\times L}$, it satisfies:

$$(\mathbf{LS})_i = \sum_{j\in\mathcal{N}_i}\sum_{k=1}^{M}\mathbf{H}_{ik}\frac{\mathbf{W}_{kk}}{\mathbf{\Delta}_{kk}}\mathbf{H}_{jk}\left(\frac{\mathbf{S}_i}{\sqrt{\mathbf{D}_{ii}}} - \frac{\mathbf{S}_j}{\sqrt{\mathbf{D}_{jj}}}\right),$$

where $\mathcal{N}_i$ is the set of vertices connected to vertex i. We can see the effect of the Laplacian matrix is to take the first-order difference in the neighborhood of vertex i. In many machine learning tasks, the parameter $\mathbf{S}$ is smoothed by minimizing the Laplacian regularization term $trace(\mathbf{S}^\top\mathbf{L}\mathbf{S})$ [11, 34].

To extract the spectral feature, we factorize the Laplacian matrix with eigen-decomposition: $\mathbf{L} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^\top$, where $\mathbf{\Lambda} = diag(\lambda_1, \lambda_2, \cdots, \lambda_N)$ is the eigenvalue matrix, all eigenvalues are in ascending order, i.e., $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_N$. We define the matrix formed by the first K eigenvectors as the spectral feature matrix: $\mathbf{F} = \mathbf{\Phi}_{*,1:K}$, and $\mathbf{F}_i$ is the spectral feature of vertex i. The spectral feature can be used for spectral clustering [23, 29] and graph Fourier transform [28]. It contains the information of the similarity among vertices thus we can measure the similarity of two vertices i and j with $\mathbf{F}_i\mathbf{F}_j^\top$. Though eigen-decomposition is computationally expensive (taking the user hypergraph as an example, with $O(N^3)$ time complexity, where N

[1] We use "strongly connected" to indicate that vertices are connected by many paths in the graph (including direct connections or high-order connections), which is different from the conception in directed graphs [40].

is the user number), however, $\mathbf{L}$ is highly sparse in recommendation tasks. Considering there are $O(N)$ non-zero elements in $\mathbf{L}$, the time complexity could be $O(K^2N)$ with Lanczos method [13, 32], which is much smaller than $O(N^3)$.

3.2 Hybrid Model

For the hybrid model (SCF), we incorporate the spectral features into a basic MF model, which is the state-of-the-art for rating prediction as well as modeling implicit feedback, to reconstruct the interaction matrix with the low-rank form:

$$\hat{\mathbf{R}} = \mathbf{U}\mathbf{V}^\mathsf{T} + \mathbf{P}\mathbf{F}^\mathsf{T} + \mathbf{E}\mathbf{Q}^\mathsf{T}, \tag{2}$$

where $\hat{\mathbf{R}} \in \mathbb{R}^{N\times M}$ is the reconstruction, $\mathbf{U} \in \mathbb{R}^{N\times K_0}$ and $\mathbf{V} \in \mathbb{R}^{M\times K_0}$ are the latent factors of users and items respectively. $\mathbf{U}_i$ is the preference of user i and $\mathbf{V}_j$ is the properties of item j. $\mathbf{E} \in \mathbb{R}^{N\times K_1}$ and $\mathbf{F} \in \mathbb{R}^{M\times K_2}$ are the spectral features of users and items respectively. $\mathbf{P} \in \mathbb{R}^{N\times K_2}$ is the preference matrix of users, $\mathbf{Q} \in \mathbb{R}^{M\times K_1}$ is the fitness matrix of items.

Considering that strongly connected users/items shall have similar preference/properties, $\mathbf{E}/\mathbf{F}$ contains the information of similarity among users/items. We use the similarity information to enhance the MF predictor. We call the three terms in Equation (2) as model-based, item-based, and user-based collaborative terms respectively. In a conventional item-based collaborative filtering model, we calculate the similarity between each pair of items. For a positive sample i and a missing sample j, if j is similar with i, we recommend j to current user. Similarly, in the second term of our predictor, for current user u and her purchased item i, $\mathbf{P}_u\mathbf{F}_i^\mathsf{T}$ is high. For a similar item j, $\mathbf{F}_i\mathbf{F}_j^\mathsf{T}$ is high, thus $\mathbf{P}_u\mathbf{F}_j^\mathsf{T}$ is high as well, and j can be recommended to u. We can see that this procedure is just the same as item-based collaborative filtering, so we call the second term of Equation (2) item-based collaborative term. And the third term, for the same reason, is called user-based collaborative term.

Now we briefly discuss the advantage of our item-based collaborative term over conventional item-based collaborative filtering. In item-based collaborative filtering, when calculating the similarity between a pair of items, only first-order connections are taken into account, while in our item-based collaborative term, high-order connections are also considered. Take Figure 1(c) as an example, i_3 and i_4 are not connected by certain hyperedge directly, hence the similarity is 0 in conventional item-based collaborative filtering. While in our model, connections i_4-i_1-i_3 and i_4-i_1-i_2-i_3 are also considered, therefore $sim(i_3, i_4) > 0$ (we will discuss the reason detailedly in the next section). In fact, in Equation (2), three models give the final prediction jointly: An MF model, and two hypergraph spectrum-enhanced memory-based collaborative filtering models.

4 SPECTRUM-ENHANCED PAIRWISE LEARNING

Besides providing information to model users' preference and items' property, spectral features are also used to optimize the model. Optimization on implicit feedback data usually takes the form of pairwise learning, which maximizes the likelihood of relative preference over a pair of positive and negative feedbacks:

$$\text{BPR_OPT} = \sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{I}\setminus\mathcal{I}_u^+} \ln\sigma\left(\hat{\mathbf{R}}_{uij}\right) - \frac{\lambda_r}{2}\|\mathbf{\Theta}\|_\mathrm{F}^2 .$$

where $\mathcal{U}$ and $\mathcal{I}$ are the sets of users and items. $\mathcal{I}_u^+$ is the set of positive items of u. $\sigma(\ )$ is the sigmoid function. $\hat{\mathbf{R}}$ is defined in the Equation (2) and $\hat{\mathbf{R}}_{uij} = \hat{\mathbf{R}}_{ui} - \hat{\mathbf{R}}_{uj}$. The last term is the regularization term to prevent overfitting, where λ_r is the regularization coefficient, $\|\ \|_\mathrm{F}$ is the Frobenius norm of the matrix, and $\mathbf{\Theta}$ represents the parameters of the model.

There is a critical issue: A user did not purchase an item may not because she has no interest in it, but just because she has never seen it yet. Our task is to uncover users' preference and recommend them unseen items they are interested in. However, in BPR, all missing entries are treated as negative samples nevertheless some of them are indeed unlabeled positive samples. To address this gap, we enrich the positive samples by the spectral clustering.

4.1 Objective Function

We cluster all vertices (items/users) by the normalized spectral features. Vertices in the same cluster are strongly connected in the hypergraph, though may not be connected directly.

Latent category: We define a cluster of items as a latent category, items in it are of the similar kind, or highly relevant, since they are purchased by the same user or users with similar preference. For certain item i, the latent category it belongs to is denoted as C_i. We argue that if a user likes i, she may like the items in C_i with a high probability.

Latent community: Similarly, we define a cluster of users as a latent community, users in it purchase the same items, similar items, or relevant items, thus they have similar preference. For certain user u, the latent community she belongs to is denoted as C_u. If a user likes i, her latent community members may like i as well.

To give the relative preference, we construct three sets for each user u:

$$\begin{cases} \mathcal{I}_u^+ = \{i|\mathbf{R}_{ui} = 1\}, & \text{Positive set} \\ \mathcal{P}_u = \mathcal{P}_u^U \bigcup \mathcal{P}_u^I, & \text{Potential set} \\ \mathcal{I}_u^- = \mathcal{I} - \mathcal{I}_u^+ - \mathcal{P}_u, & \text{Negative set} \end{cases}, \tag{3}$$

where $\mathcal{P}_u^U = \{i|i \in \mathcal{I}_v^+, v \in C_u \& i \notin \mathcal{I}_u^+\}$ is the user-based collaborative potential set, $\mathcal{P}_u^I = \{j|j \in C_i, i \in \mathcal{I}_u^+ \& j \notin \mathcal{I}_u^+\}$ is the item-based collaborative potential set. We have the preference relationship:

$$(u, \mathcal{I}_u^+) \succ (u, \mathcal{I}_u^-),\ (u, \mathcal{I}_u^+) \succ (u, \mathcal{P}_u),\ (u, \mathcal{P}_u) \succ (u, \mathcal{I}_u^-).$$

We can see that, $\mathcal{P}_u$ is the set of items that u may have interests in returned by the memory-based collaborative filtering.

BPR tries to optimize the standard AUC that is designed for binary classification [3], while we try to optimize a generalized AUC (GAUC) [24, 41],

$$\text{GAUC} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\Big[\text{AUC}(u, \mathcal{I}_u^+, \mathcal{I}_u^-) + \eta_1\text{AUC}(u, \mathcal{I}_u^+, \mathcal{P}_u) + \eta_2\text{AUC}(u, \mathcal{P}_u, \mathcal{I}_u^-)\Big],$$

where $\text{AUC}(u, \mathcal{A}, \mathcal{B}) = \frac{1}{|\mathcal{A}||\mathcal{B}|}\sum_{i\in\mathcal{A}}\sum_{j\in\mathcal{B}}\delta\left(\hat{\mathbf{R}}_{uij} > 0\right)$ is the standard AUC, η_1 and η_2 are confidence coefficients. We use differentiable loss $\delta(x > 0)$ which is identical to $\ln\sigma(x)$. We can see

that GAUC is the weighted combination of three standard AUC terms. To maximize it, we propose our **S**pectrum-enhanced **P**airwise **L**earning to **R**ank (**SPLR**) optimization:

$$\begin{aligned}\text{SPLR_OPT} = \sum_{u\in\mathcal{U}}\Bigg[&\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{I}_u^-}\ln\sigma\left(\hat{\mathbf{R}}_{uij}\right)\\ &+\eta_1\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{P}_u}\ln\sigma\left(\hat{\mathbf{R}}_{uij}\right)\\ &+\eta_2\sum_{i\in\mathcal{P}_u}\sum_{j\in\mathcal{I}_u^-}\ln\sigma\left(\hat{\mathbf{R}}_{uij}\right)\Bigg]-\frac{\lambda_r}{2}\|\mathbf{\Theta}\|_{\mathrm{F}}^2.\end{aligned}\tag{4}$$

4.2 Model Learning

To maximize the SPLR objective function, we take the first-order derivatives of Equation (4) with respect to each model parameter:

$$\begin{aligned}\nabla_{\mathbf{\Theta}}\text{SPLR_OPT} = \sum_{u\in\mathcal{U}}\Bigg[&\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{I}_u^-}\sigma\left(-\hat{\mathbf{R}}_{uij}\right)\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\mathbf{\Theta}}\\ &+\eta_1\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{P}_u}\sigma\left(-\hat{\mathbf{R}}_{uij}\right)\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\mathbf{\Theta}}\\ &+\eta_2\sum_{i\in\mathcal{P}_u}\sum_{j\in\mathcal{I}_u^-}\sigma\left(-\hat{\mathbf{R}}_{uij}\right)\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\mathbf{\Theta}}\Bigg]-\lambda_r\,\mathbf{\Theta}.\end{aligned}\tag{5}$$

We use $\boldsymbol{\theta}$ to denote certain row of $\mathbf{\Theta}$, the derivatives in Equation (5) are:

$$\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\boldsymbol{\theta}}=\begin{cases}\mathbf{V}_i-\mathbf{V}_j & \text{if}\quad \boldsymbol{\theta}=\mathbf{U}_u\\ \mathbf{U}_u/-\mathbf{U}_u & \text{if}\quad \boldsymbol{\theta}=\mathbf{V}_i/\mathbf{V}_j\\ \mathbf{F}_i-\mathbf{F}_j & \text{if}\quad \boldsymbol{\theta}=\mathbf{P}_u\\ \mathbf{E}_u/-\mathbf{E}_u & \text{if}\quad \boldsymbol{\theta}=\mathbf{Q}_i/\mathbf{Q}_j\end{cases}.\tag{6}$$

$\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\boldsymbol{\theta}}$ in Equation (6) is certain row of $\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\mathbf{\Theta}}$ in Equation (5), for example, the u-th row when $\boldsymbol{\theta}=\mathbf{U}_u$.

The detailed procedure is shown in Algorithm 1. We calculate the Laplacian matrices with the Equation (2) (line 1) and decompose them, the first K_1/K_2 eigenvectors of $\mathbf{L}^U/\mathbf{L}^I$ form $\mathbf{E}/\mathbf{F}$ (line 2). We then construct the three sets in Equation (3) for each user (line 3). Lines 5-14 show the process of model learning. For each record $\{u,i\}$, we choose m potential samples $\{j_1,\cdots,j_m\}$ from $\mathcal{P}_u$ (line 9) and m negative samples $\{k_1,\cdots,k_m\}$ from $\mathcal{I}_u^-$ (line 10), m is the sampling rate. The model is optimized with the pairs $\{u,i,j_1\},\cdots,\{u,i,j_m\},\{u,i,k_1\},\cdots,\{u,i,k_m\},\{u,j_1,k_1\},\cdots,\{u,j_m,k_m\}$. We finally calculate the derivatives given by Equation (5) with a batch of pairs (line 12) and update the parameters (line 13).

Like many existing works [30, 33, 49], we also use the connections of users and items to enhance the pairwise learning. However, in previous works, only one-order connections are utilized while in our SPLR model, we leverage connections with all orders. In a hypergraph, we have $\mathbf{L}^n=(\mathbf{I}-\mathbf{A})^n=\sum_{r=1}^{n}C_n^r(-\mathbf{A})^r$, where $\mathbf{A}=\mathbf{D}^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\mathbf{\Delta}^{-1}\mathbf{H}^{\mathsf{T}}\mathbf{D}^{-\frac{1}{2}}$ is the normalized adjacency matrix. $\mathbf{A}^r$ indicates the r-order connections, therefore $\mathbf{L}^n$ contains the information of all connections less than n-order in the hypergraph. For the i-th eigenvector $\mathbf{u}_i$ of $\mathbf{L}$, λ_i is the corresponding eigenvalue, we have $\mathbf{L}\mathbf{u}_i=\lambda_i\mathbf{u}_i$. Since $\mathbf{L}^n\mathbf{u}_i=\lambda_i^n\mathbf{u}_i$, $\mathbf{u}_i$ is also the eigenvector of $\mathbf{L}^n$, therefore $\mathbf{F}$ contains information of all-order connections in the hypergraph. Constructed with the spectral features, latent communities/categories also take the high-order information into consideration, that is the superiority of our model over existing learning to rank methods.

Algorithm 1: Mini-batch gradient descent algorithm

Input: Implicit feedback matrix $\mathbf{R}$, regularization coefficient λ_r, batch size b, learning rate η, weighting parameters η_1 and η_2, sampling rate m, maximum number of iterations $iter_max$.
Output: top-n predictions given by the completed matrix $\hat{\mathbf{R}}$

1 calculate the Laplacian matrices of user hypergraph $\mathbf{L}^U$ and item hypergraph $\mathbf{L}^I$ from $\mathbf{R}$;
2 decompose the Laplacian matrices to get the spectral features $\mathbf{E}$ and $\mathbf{F}$;
3 cluster vertices and calculate the potential set for each user;
4 initialize $\mathbf{\Theta}$ randomly;
5 **for** $iter = 1$ *to* $iter_max$ **do**
6 split all purchase records into b-size batches;
7 **for** *each batch* **do**
8 **for** *each record in current batch* **do**
9 select m potential items randomly from $\mathcal{P}_u$;
10 select m negative items randomly from $\mathcal{I}_u^-$;
11 add these items to the current batch;
12 calculate $\nabla_{\mathbf{\Theta}}$SPLR_OPT with current batch;
13 $\mathbf{\Theta}=\mathbf{\Theta}+\eta\nabla_{\mathbf{\Theta}}$SPLR_OPT;
14 calculate $\hat{\mathbf{R}}$ to predict the top-n items;
15 **return** *the top-n items*;

5 EXPERIMENTS

In this section, we design experiments on real-world datasets to validate the effectiveness of our models. We evaluate our proposed methods focusing on the following three key research questions:

RQ1: How is the performance of our entire spectrum-enhanced recommendation model (effectiveness of SCF_SPLR)?
RQ2: How is the performance of the spectral features to model users' preference and items' properties (effectiveness of SCF_BPR)?
RQ3: How is the performance of the SPLR optimization criterion (effectiveness of MF_SPLR)?

Table 1: Statistics of datasets

Dataset	Purchase	User	Item	Sparsity
Clothes	115841	32728	8777	99.9597%
Jewelry	37314	15924	3607	99.9350%

5.1 Datasets

In this paper, we perform experiments on the *Amazon* dataset [14], which is the consumption records from *Amazon.com*. We consider two large categories *Amazon.clothes* and *Amazon.jewelry*, which are filtered from the *Amazon* dataset. We take users' review histories as implicit feedback. Some details of the datasets are shown in Table 1.

Table 2: Recommendation performance (test set)

Datasets	Metrics (%)		MP	PMF	BPR	GBPR	VBPR	SPLR	Improvement BPR	Improvement GBPR
Jewelry	*F*-1@	2	0.808±0.048	3.851±0.144	3.684±0.194	3.811±0.178	3.998±0.165	**4.104**±0.185	11.40%	7.69%
		5	0.747±0.039	3.507±0.120	3.576±0.137	3.656±0.115	3.820±0.068	**3.905**±0.136	9.20%	6.81%
		10	0.608±0.012	3.064±0.082	3.159±0.099	3.192±0.102	**3.354**±0.109	3.339±0.068	5.70%	4.61%
		20	0.425±0.014	2.403±0.026	2.489±0.042	2.515±0.040	**2.653**±0.046	2.638±0.064	5.99%	4.89%
	NDCG@	2	0.618±0.047	3.762±0.128	3.722±0.184	3.845±0.185	3.960±0.192	**4.057**±0.179	9.00%	5.51%
		5	0.513±0.014	2.827±0.075	2.775±0.106	2.924±0.104	3.075±0.102	**3.099**±0.121	11.68%	5.98%
		10	0.402±0.012	2.229±0.047	2.227±0.039	2.301±0.047	2.445±0.043	**2.450**±0.048	10.01%	6.48%
		20	0.280±0.009	1.735±0.039	1.738±0.017	1.788±0.036	**1.892**±0.043	1.861±0.046	7.08%	4.08%
Clothes	*F*-1@	2	0.601±0.043	2.638±0.063	2.831±0.116	2.918±0.107	3.078±0.098	**3.125**±0.087	10.39%	7.09%
		5	0.563±0.032	2.481±0.140	2.689±0.041	2.803±0.045	2.945±0.048	**2.999**±0.070	11.53%	6.99%
		10	0.559±0.021	2.087±0.131	2.322±0.039	2.445±0.040	2.464±0.040	**2.531**±0.063	9.00%	3.52%
		20	0.462±0.027	1.626±0.065	1.828±0.056	1.914±0.054	1.957±0.053	**2.034**±0.022	11.27%	6.27%
	NDCG@	2	0.619±0.032	2.791±0.052	3.026±0.134	3.118±0.126	3.119±0.118	**3.331**±0.144	10.08%	6.83%
		5	0.452±0.039	2.021±0.109	2.185±0.068	2.293±0.054	2.410±0.040	**2.451**±0.028	12.17%	6.89%
		10	0.373±0.009	1.567±0.091	1.740±0.052	1.844±0.047	1.865±0.042	**1.907**±0.042	9.60%	3.42%
		20	0.300±0.007	1.197±0.054	1.330±0.069	1.402±0.045	1.442±0.022	**1.477**±0.020	11.05%	5.35%

5.2 Baselines

We adopt the following methods as baselines for performance comparison:

- **MP:** This **M**ost **P**opular method ranks items according to their popularity. It is a non-personalized method to benchmark the recommendation performances.
- **PMF:** This **P**robabilistic **M**atrix **F**actorization method, which was proposed by [38], is a frequently used state-of-the-art approach for rating-based optimization and prediction. We set the score of positive samples as 1 and missing values as 0.
- **BPR (MF_BPR):** This **B**ayesian **P**ersonalized **R**anking method is the most widely used ranking-based method for implicit feedback [37]. It regards all unobserved samples as negative samples and maximizes the likelihood of users' preference over a pair of positive sample and negative sample.
- **GBPR (MF_GBPR):** This **G**roup Preference-based **B**ayesian **P**ersonalized **R**anking method [30] is an extension of BPR, which tries to relax BPR's assumptions to a group pairwise preference assumption. We fix the number of grouped users to 3.
- **VBPR:** This **V**isual **B**ayesian **P**ersonalized **R**anking method is a stat-of-the-art visual-based recommendation method [14]. The visual features are extracted from the product images with a pre-trained deep convolutional neural network (CNN).

5.3 Experiment Settings

The eigen-decomposition of Laplacian matrices is implemented with `sparse.linalg()` function of `scipy` library in `python`. The *Amazon* dataset is filtered with 5-core (remove users and items with less than 5 purchase records) and records before 2010 is removed. We then filter *Jewelry* and *Clothes* datasets from *Amazon*. Each dataset is split into training set (80%), validation set(10%), and test set (10%) randomly. Cold items and users (items and users with no record in training set) in validation and test sets are removed. We adapt F_1-score and normalized discounted cumulative gain (*NDCG*) to evaluate the performance of the baselines and our proposed model. Mini-batch stochastic gradient descent (MSGD) is leveraged to optimize our model. The learning rate is determined by grid searching in the range of {0.005, 0.01, 0.02, 0.05, 0.1, 0.2} and the batch size is determined in the range of {1000, 2000, 3000, 4000, 5000}. We evaluate different number of latent factors K_0 in the range of {10, 20, 50, 100, 200}, the regularization coefficient λ_r in the range of $\{0, 0.1, 0.2, \cdots, 1.5\}$, and the weighted parameters η_1 and η_2 in the range of $\{0, 0.01, 0.1\}$. We conduct our experiments by predicting top-{2, 5, 10, 20} items to each user. The sampling rate m is set as 5 (select 5 negative/potential samples for each positive sample to construct pairs) in all pairwise learning to balance the accuracy and efficiency.

5.4 Performance of Our Entire Model (RQ1)

In this subsection, we report the performance of our entire model (SCF model optimized with SPLR, SCF_SPLR, abbreviated as SPLR) and baselines to give some analysis. We tune all models in the validation set and test the performance in the test set. The impact of some important hyperparameters, such as the number of latent factors, regularization coefficient, and weighted parameters, are shown. When training, we iterate the whole dataset 200 times to learn models and select 1000 samples randomly from the test/validation set to test all models in each iteration (except MP, we just test 200 times without training). We record the best performance of each model during this procedure as the evaluation of it. We execute all models 5 times and then report the mean and standard deviation. The learning rate is set as 0.05 and the batch size is set as 5000 for all models.

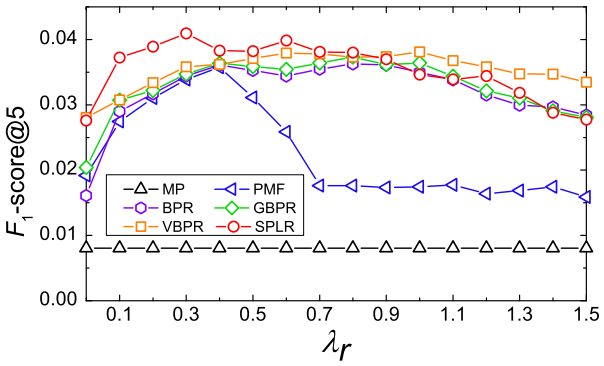
The performance of all models is represented in Table 2, we can see that all personalized models outperform MP significantly in all situations — about 3-6 times improvement. Comparison between BPR and PMF shows the superiority of pairwise learning over point-wise one. Considering the group behaviour, GBPR gets a further enhancement and outperforms BPR in all situations. As an enhanced learning to rank method, we compare our model with BPR and GBPR, and the improvement is shown in the last two columns in Table 2. We can see that our model dramatically outperforms these two models. In GBPR, only one-order user connections are utilized while in SPLR, the spectral features capture high-order connections for both users and items, thus provides high-level and comprehensive description of the similarity among vertices.

We also report the performance of VBPR, the state-of-the-art side information-based model. With the extra visual feature, it performs the best among all baselines. As we can see, our model can even outperform VBPR in most situations. Though both utilizing "side" information to enhance the accuracy, VBPR uses "outside" information, i.e., extra visual data, while SPLR uses "inside" information extracted from the purchase records. From the last two columns of Table 2, we can see that in *Jewelry* dataset, the relative improvement of our model decreases with the increasing of n (the number to recommend), while in *Clothes* dataset, the relative improvement is stable. That is not because the our model performs badly with a large n, it is the property of the dataset. We can see that in *Jewelry* set, the gap (relative improvement) between any two pairwise learning models decreases with the increasing of n.

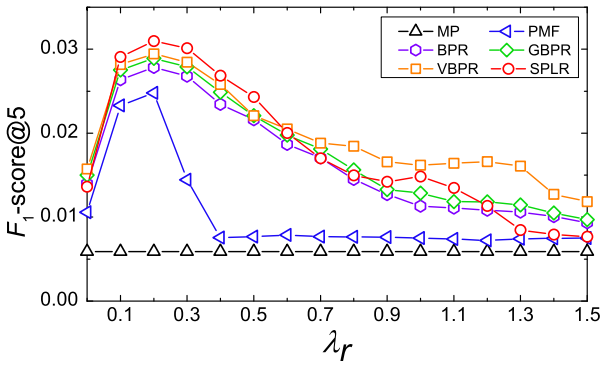
To analyze the sensitivity of our model, we report the performance (F_1-score@5) with different hyperparameters. The impact of regularization coefficient λ_r is represented in Figure 2. Though both filtered from *Amazon* dataset, these two datasets have pretty different properties. From Figure 2(a), we can see that all models perform quiet differently to each other in *Jewelry* dataset, PMF, BPR, GBPR, VBPR, and SPLR get the best performance when λ_r is set to 0.4, 0.8, 0.8, 1.0, and 0.3 respectively. While in Figure 2(b), all models get the best performance when λ_r is set to 0.2. With the increasing of λ_r, accuracy of the point-wise learning method decreases rapidly compared with pairwise learning methods.

We also report F_1-score@5 with different length of latent factors to get the best K_0. In both two datasets, performance of all models (except MP) increase with the increasing of dimensions. We can see that due to the smaller size, the purchase record matrix of *Jewelry* has a smaller rank than that of *Clothes*: The performance in *Jewelry* tends to be stable after K_0 is larger than 50 while the performance in *Clothes* keeps increasing obviously with K_0 changing from 10 to 200. To get the best performance, we set K_0 as 200 for all models.

The impact of weighting parameters is illustrated in Figure 4. Our model performs the best when $\eta_1 = 0.01$ and $\eta_2 = 0.01$ in these two datasets. When $\eta_1 = 0$ and $\eta_2 = 0$, SPLR (SCF_SPLR) degenerates into SCF_BPR. From the figures we can see that SPLR outperforms SCF_BPR 4.70% in *Jewelry* validation dataset and 4.03% in *Jewelry* validation dataset on F_1-score@5. We can see that the enhancement of SLPR is not so significant, this is because we have explored the similarity information by incorporating the spectral features into SCF_BPR model, which outperforms MF_BPR significantly (Table 3). We leverage SPLR optimization to explore the similarity information thoroughly for further accuracy improvement.

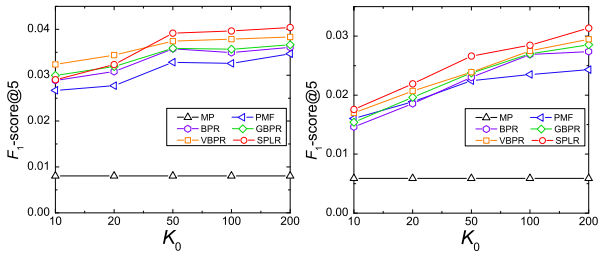


(a) *Jewelry*



(b) *Clothes*

Figure 2: Impact of regularization coefficient λ_r (validation set)



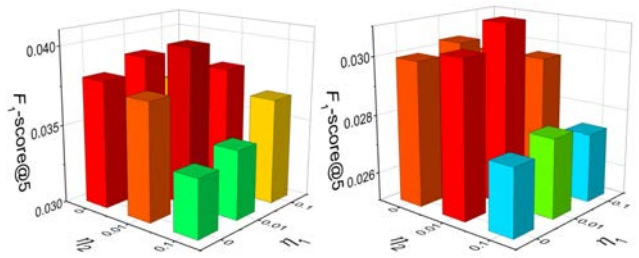
(a) *Jewelry* (b) *Clothes*

Figure 3: Impact of latent factor dimensions K_0 (validation set)

We demonstrated the effectiveness of our entire model in this subsection, and in next subsections, we illustrate the effectiveness of each part of our model — spectral feature-enhanced model (SCF_BPR) and spectral clustering-enhanced pairwise learning optimization strategy (MF_SPLR).

5.5 Effectiveness of SCF_BPR (RQ2)

In this subsection, we denote all models with the model names and optimization methods. We rename BPR as MF_BPR, which means



(a) *Jewelry* (b) *Clothes*

Figure 4: Impact of weighting parameters η_1 and η_2 (validation set)

MF model optimized with BPR algorithm. Similarly, SCF_BPR is SCF model optimized with BPR. We demonstrate the effectiveness of spectral features by comparing SCF_BPR with other baselines. We first report the sensitivity with the number of eigenvectors by grid searching in the range of {0,10,100,1000}. All experiments in this and next subsection are conducted in *Jewelry* dataset.

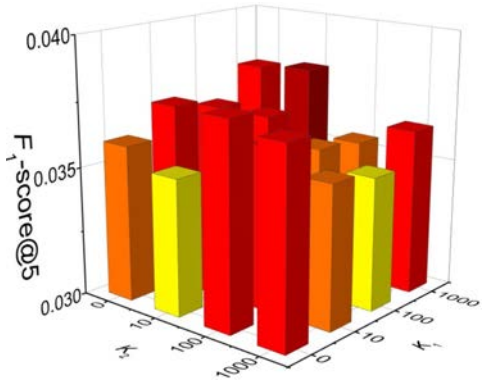


Figure 5: Performance of SCF_BPR with different length of spectral features (*Jewelry*, validation set)

The sensitivity of SCF_BPR with the length of spectral features is shown in Figure 5. When $K_1 = 1000$ and $K_2 = 10$, SCF_BPR performs the best. When $K_1 = 0$ and $K_2 = 0$, SCF_BPR degenerates into MF_BPR. SCF_BPR outperforms MF_BPR 7.2% on F_1-score@5 in *Jewelry* validation set. From Figure 5 we can see that user feature and item feature do not jointly work well since SCF_BPR does not perform well when K_1 and K_2 are both large. In real-world application, we can set $K_1 = 0$ and $K_2 = 100$ to balance the accuracy and the efficiency.

The performance of SCF_BPR and baselines in test set is shown in Table 3. To save space, only the performance of MF_BPR is reported, since it is the most important baseline. We can check Table 2 for the performance of other baselines. With the spectral features providing similarity information, SCF_BPR outperforms MF_BPR 7.27% and 6.38% on F_1-score and *NDCG* respectively in the best situation.

Table 3: Performance comparison for SCF_BPR (*Jewelry*, test set)

Metrics (%)		MF_BPR	SCF_BPR	Improvement
F-1@	2	3.684±0.194	**3.952**±0.221	7.27%
	5	3.576±0.137	**3.768**±0.098	5.37%
	10	3.159±0.099	**3.301**±0.087	4.50%
	20	2.489±0.042	**2.596**±0.029	4.30%
NDCG@	2	3.722±0.184	**3.953**±0.151	6.21%
	5	2.775±0.106	**2.952**±0.074	6.38%
	10	2.227±0.039	**2.366**±0.037	6.24%
	20	1.738±0.017	**1.835**±0.023	5.58%

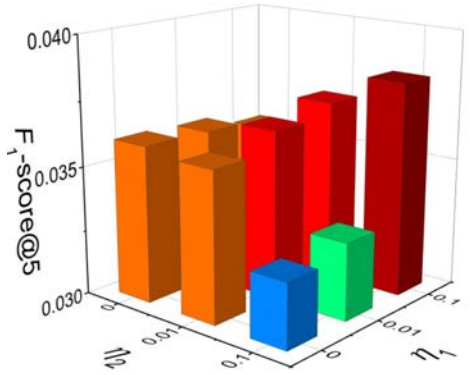


Figure 6: Performance of MF_SPLR with different weighting parameters (*Jewelry*, validation set)

5.6 Effectiveness of MF_SPLR (RQ3)

The sensitivity of MF_SPLR with the weighting parameters η_1 and η_2 are shown in Figure 6. We can see that MF_SPLR performs the best when $\eta_1 = 0.1$ and $\eta_2 = 0.1$. When $\eta_1 = 0$ and $\eta_2 = 0$, MF_SPLR degenerates to MF_BPR. MF_SPLR outperforms MF_BPR 6.13% F_1-score@5 in *Jewelry* validation set. Comparing Figure 4(a) and Figure 6 we can see that though both enhanced with SPLR optimization, MF_SPLR gains more improvement than SCF_SPLR, and the SPLR terms are weighted more in MF_SPLR ($\eta_1 = 0.1$, $\eta_2 = 0.1$ in MF_SPLR while $\eta_1 = 0.01$, $\eta_2 = 0.01$ in SCF_SPLR). It is because that the similarity information has been leveraged in SCF_SPLR by modeling with the spectral features. While in SCF_SPLR, SPLR is the only way to utilize the similarity information.

The performance of MF_SPLR and baselines in *Jewelry* test set is shown in Table 4. From the table we can see that SCF_BPR outperforms MF_BPR 8.20% and 5.42% on F_1-score and *NDCG* respectively in the best situation. Comparing Tables 2, 3, and 4, we can see that the improvement of the entire model is less than the sum of the improvement of each part. That may be because our model utilizes the similarity information by modeling (SCF) and optimizing (SPLR), thus these two parts provide the same information in different ways.

In Figure 7, we show some latent categories of the *Jewelry* set, which contains all watches and jewelries. There are six latent categories in Figure 7. Not like a real category, items in a latent category



Figure 7: Latent categories (*Jewelry* set)

Table 4: Performance comparison for MF_SPLR (*Jewelry*, test set)

Metrics (%)		MF_BPR	MF_SPLR	Improvement
F-1@	2	3.684±0.194	**3.986**±0.074	8.20%
	5	3.576±0.137	**3.789**±0.123	5.96%
	10	3.159±0.099	**3.314**±0.056	4.91%
	20	2.489±0.042	**2.578**±0.054	3.58%
NDCG@	2	3.722±0.184	**3.924**±0.151	5.43%
	5	2.775±0.106	**2.923**±0.074	5.33%
	10	2.227±0.039	**2.339**±0.037	5.03%
	20	1.738±0.017	**1.797**±0.023	3.39%

are relevant items or similar items (items strongly connected in the hypergraph). For example, in Figure 7(a), items are watches and some relevant commodities, such as watch bands and watch boxes[2]. Users who want to by watches may have interests in this category. Though items in the same latent categories may be different kinds of commodities, they are in the same style. For example, items in Figure 7(a) are designed for business men, they are luxurious, with metallic luster, look mature and steady. On the contrary, different categories may contain items of the same kind, but they are in different styles. For example, the category in Figure 7(b) is also for watches, however is in a totally different style from that in Figure 7(a). Most items in this category are plastic and cheap digital watches for sports and boys may have interests in them. Items in Figure 7(c) are watches and jewelries for young girls, they look simple and elegant. Jewelries in Figure 7(d) and Figure 7(e) are luxurious and exaggerative, which are full of diamonds and gemstones. Items in Figure 7(f) are in punk style and rebellious teenagers may prefer them.

[2]https://www.amazon.com/dp/B005IHDLYC/?tag=tc0f3f-20

Table 5: Performance with different features (*Jewelry*, test set)

Metrics (%) \ Features		None	Spectral features	CNN feature	Spectral features & CNN feature
F-1@	2	3.684±0.194	4.104±0.185	4.218±0.213	**4.414**±0.177
	5	3.576±0.137	3.905±0.136	3.974±0.106	**4.108**±0.165
	10	3.159±0.099	3.339±0.068	3.398±0.103	**3.567**±0.120
	20	2.489±0.042	2.638±0.064	2.631±0.067	**2.800**±0.103
NDCG@	2	3.722±0.184	4.057±0.179	4.184±0.165	**4.392**±0.278
	5	2.775±0.106	3.099±0.121	3.173±0.090	**3.315**±0.183
	10	2.227±0.039	2.450±0.048	2.542±0.078	**2.623**±0.137
	20	1.738±0.017	1.861±0.046	1.934±0.073	**2.011**±0.092

Figure 7 shows how the item spectral feature helps to recommend. The item spectral feature indicates the style of the item, circle of interest the item belongs to, the price level, target users, etc. Modeling with the item spectral feature (SCF) makes the items in the same latent category tend to have similar score. And optimizing with the item spectral feature (SPLR) makes the items in the positive category (latent category containing positive samples) tend to have a high score. For the user spectral feature, we can draw the same conclusion.

Both spectral features and latent factors are extracted from graph structures which are constructed from the purchase records (one bipartite graph and two hypergraphs). In fact, they both mine the similarity information while with different emphasis — user/item spectral features describe the similarity of users/items, while latent factors describe the similarity between a user and an item. Spectral features and latent factors complete each other in describing the similarity information and enhance the effectiveness of the recommendation model.

6 EXTENSION

In this paper, we proposed new spectral features and incorporated them into an MF model to predict users' preference. We then introduced new spectral clustering-based pairwise learning method to optimize our model. In this section, we extend our model and propose a framework of learning to rank models enhanced with side information features. Not only the spectral features, all features can be leveraged in our framework. Assuming there are n features for users $\mathbf{E}^{(1)}, \cdots, \mathbf{E}^{(n)}$ and m features for items $\mathbf{F}^{(1)}, \cdots, \mathbf{F}^{(m)}$, the prediction is given by:

$$\hat{\mathbf{R}} = \mathbf{U}\mathbf{V}^\top + \sum_{j=1}^{m} \mathbf{P}^{(j)}\mathbf{F}^{(j)\top} + \sum_{k=1}^{n} \mathbf{E}^{(k)}\mathbf{Q}^{(k)\top},$$

where $\mathbf{P}_u^{(j)}$ is the preference of user u based on the j-th item feature. These item features can be high-level features like the spectral feature, CNN feature, auditory feature, or low-level features like the color histogram or tags. $\mathbf{Q}_i^{(k)}$ is the fitness of item i based on the k-th user feature. These user features can be high-level features like the spectral feature, or the low-level information vector containing such as age, genders, addresses, etc.

For pairwise learning, we use the latent community $C_u = \bigcup_{j=1}^{n} C_u^{(j)}$ and latent category $C_i = \bigcup_{k=1}^{m} C_i^{(k)}$ to construct the potential set in Equation (3), where $C_u^{(j)}$ is the latent community of user u clustered by the j-th feature and $C_i^{(k)}$ is the latent category of item i clustered by the k-th feature.

Experimental results of our extended model are shown in Table 5. These four columns are performances of our extended models with no feature (i.e., MF_BPR), with the spectral features (i.e., SCF_SPLR), with the CNN feature (i.e., VBPR optimized with visual clustering-enhanced pairwise learning to rank), and with the spectral features plus the CNN feature. Enhanced with the spectral features, SCF_SPLR outperforms MF_BPR 9.20% on F_1-score@5. Leveraging appearance information, our extended model with the CNN feature outperforms MF_BPR 11.13% on F_1-score@5. Comparing Table 2 and Tale 5 we can see that it also outperforms VBPR 4.03% on F_1-score@5 due to the enhanced pairwise learning optimization. Our extended model with the CNN feature and spectral features outperforms MF_BPR 14.88% on F_1-score@5.

7 CONCLUSION AND FUTURE WORK

In this paper, we investigated the usefulness of the spectral features in recommendation tasks. We first introduced novel spectral features, which contain similarity information, and injected them into an MF structure to model users' preference and items' properties. We then clustered the spectral features to construct the latent communities and categories for users and items respectively, and used them to enhance the pairwise learning. We finally extended our model and proposed a framework for side information-enhanced pairwise learning. Experiments on challenging real-world datasets show that our proposed methods significantly outperform state-of-the-art models.

For future work, we will investigate the effectiveness of our proposed spectral features in the setting of explicit feedback. Also, we are interested in explaining the recommendation result to users with the information of latent communities and categories. Lastly, we will use neural networks [16, 18] to learn how to combine features, such as the item spectral feature and user spectral feature, or spectral features and other kinds of features.

REFERENCES

[1] James Bennett and Stan Lanning. 2007. The netflix prize. In *KDDCup*. New York, NY, USA, 35.

[2] Rianne Van Den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *Computing Research Repository - CoRR* (2017).

[3] Andrew P. Bradley. 1997. The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms. *Pattern Recogn.* (1997), 1145–1159.

[4] D. Cai, X. He, X. Wu, and J. Han. 2008. Non-negative Matrix Factorization on Manifold. In *2008 Eighth IEEE International Conference on Data Mining (ICDM '08)*. 63–72.

[5] Zhe Cao, Tao Qin, Tie Yan Liu, Ming Feng Tsai, Dawen Li, Hang Liang, Minshu Zhan, and Daniel P. W. Ellis. 2015. Content-Aware Collaborative Music Recommendation Using Pre-trained Neural Networks. In *International Society for Music Information Retrieval (ISMIR '15)*. 129–136.

[6] Zhe Cao, Tao Qin, Tie Yan Liu, Ming Feng Tsai, and Hang Li. 2007. Learning to rank:from pairwise approach to listwise approach. In *International Conference on Machine Learning (ICML '07)*. 129–136.

[7] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level Attention. In *International Conference on Research and Development in Information Retrieval (SIGIR '17)*. 335–344.

[8] Tao Chen, Xiangnan He, and Min-Yen Kan. 2016. Context-aware Image Tweet Modelling and Recommendation. In *ACM Multimedia (MM '16)*. 1018–1027.

[9] Xu Chen, Yongfeng Zhang, Qingyao Ai, Hongteng Xu, Junchi Yan, and Zheng Qin. 2017. Personalized Key Frame Recommendation. In *International Conference on Research and Development in Information Retrieval (SIGIR '17)*. 315–324.

[10] Jingtao Ding, Fuli Feng, Xiangnan He, Guanghui Yu, Yong Li, and Depeng Jin. 2018. An Improved Sampler for Bayesian Personalized Ranking by Leveraging View Data. In *International World Wide Web Conference (WWW '18)*. 13–14.

[11] Fuli Feng, Xiangnan He, Yiqun Liu, Liqiang Nie, and Tat Seng Chua. 2018. Learning on Partial-Order Hypergraphs. In *International World Wide Web Conference (WWW '18)*. 1523–1532.

[12] David Goldberg. 1992. Using collaborative filtering to weave an information tapestry. *Communications of the Acm* (1992), 61–70.

[13] Roger G. Grimes, John G. Lewis, and Horst D. Simon. 1994. A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. *Siam Journal on Matrix Analysis & Applications* (1994), 228–272.

[14] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *AAAI Conference on Artificial Intelligence (AAAI '16)*. 144–150.

[15] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *International Conference on Research and Development in Information Retrieval (SIGIR '17)*. 355–364.

[16] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer Product-based Neural Collaborative Filtering. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI '18)*. 2227–2233.

[17] Xiangnan He, Zhenkui He, Jingkuan Song, Zhenguang Liu, Yu Gang Jiang, and Tat Seng Chua. 2018. NAIS: Neural Attentive Item Similarity Model for Recommendation. *IEEE Transactions on Knowledge & Data Engineering* (2018), 1–1.

[18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *International World Wide Web Conference (WWW '17)*. 173–182.

[19] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *International Conference on Research and Development in Information Retrieval (SIGIR '16)*. 549–558.

[20] W. Hwang, J. Parc, S. Kim, J. Lee, and D. Lee. 2016. âĂIJTold you i didn't like itâĂİ: Exploiting uninteresting items for effective collaborative filtering. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE '16)*. 349–360.

[21] Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *ACM Conference on Recommender Systems (RecSys '10)*. 135–142.

[22] Yehuda Koren. 2009. The bellkor solution to the netflix grand prize. *Netflix prize documentation* (2009), 1–10.

[23] Hadrien Van Lierde and Tommy W. S. Chow. 2017. A Hypergraph Model for Incorporating Social Interactions in Collaborative Filtering. In *International Conference On Data Mining, Communications And Information Technology (DMCIT '17)*. 32.

[24] Hongzhi Liu, Zhonghai Wu, and Xing Zhang. 2018. CPLR: Collaborative Pairwise Learning to Rank for Personalized Recommendation. *Knowledge-Based Systems* (2018).

[25] Juntao Liu, Caihua Wu, Yi Xiong, and Wenyu Liu. 2014. List-wise probabilistic matrix factorization for recommendation. *Information Sciences* (2014), 434–447.

[26] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *ACM Conference on Recommender Systems (RecSys '13)*. 165–172.

[27] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *International Conference on Research and Development in Information Retrieval (SIGIR '15)*. 43–52.

[28] Narang, K Sunil, Ortega, Antonio, Vandergheynst, and Pierre. 2013. The Emerging Field of Signal Processing on Graphs. *IEEE Signal Processing Magazine* (2013), 83–98.

[29] Andrew Y Ng, Michael I Jordan, and Yair Weiss. 2002. On spectral clustering: Analysis and an algorithm. In *Annual Conference on Neural Information Processing Systems (NIPS '02)*. 849–856.

[30] Weike Pan and Li Chen. 2013. GBPR: group preference based Bayesian personalized ranking for one-class collaborative filtering. In *International Joint Conference on Artificial Intelligence (IJCAI '13)*. 2691–2697.

[31] Weike Pan, Hao Zhong, Congfu Xu, and Zhong Ming. 2015. Adaptive Bayesian Personalized Ranking for Heterogeneous Implicit Feedbacks. *Know.-Based Syst.* (2015), 173–180.

[32] Johan Paratte and Lionel Martin. 2016. Fast Eigenspace Approximation using Random Signals. *Computing Research Repository - CoRR* (2016).

[33] Shuang Qiu, Jian Cheng, Ting Yuan, Cong Leng, and Hanqing Lu. 2014. Item Group Based Pairwise Preference Learning for Personalized Ranking. In *International Conference on Research and Development in Information Retrieval (SIGIR '14)*. 1219–1222.

[34] Nikhil Rao, Hsiang-Fu Yu, Pradeep K Ravikumar, and Inderjit S Dhillon. 2015. Collaborative Filtering with Graph Information: Consistency and Scalable Methods. In *Annual Conference on Neural Information Processing Systems (NIPS '15)*. 2107–2115.

[35] Steffen Rendle. 2011. Factorization Machines. In *International Conference on Data Mining (ICDM '11)*. 995–1000.

[36] Steffen Rendle and Christoph Freudenthaler. 2014. Improving Pairwise Learning for Item Recommendation from Implicit Feedback. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM '14)*. 273–282.

[37] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Conference on Uncertainty in Artificial Intelligence (UAI '09)*. 452–461.

[38] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *Annual Conference on Neural Information Processing Systems (NIPS '07)*. 1257–1264.

[39] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *International World Wide Web Conference (WWW '01)*. 285–295.

[40] M. Sharir. 1981. A strong-connectivity algorithm and its applications in data flow analysis. *Computers & Mathematics with Applications* (1981), 67–72.

[41] Dongjin Song and David A Meyer. 2015. Recommending Positive Links in Signed Social Networks by Optimizing a Generalized AUC.. In *AAAI Conference on Artificial Intelligence (AAAI '15)*. 290–296.

[42] Frank Edward Walter, Stefano Battiston, and Frank Schweitzer. 2008. A Model of a Trust-based Recommendation System on a Social Network. *Autonomous Agents and Multi-Agent Systems* (2008), 57–74.

[43] Mao Ye, Peifeng Yin, Wang Chien Lee, and Dik Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *International Conference on Research and Development in Information Retrieval (SIGIR '11)*. 325–334.

[44] Wenhui Yu, Huidi Zhang, Xiangnan He, Xu Chen, Li Xiong, and Zheng Qin. 2018. Aesthetic-based Clothing Recommendation. In *International World Wide Web Conference (WWW '18)*. 649–658.

[45] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. 2013. Time-aware point-of-interest recommendation. In *International Conference on Research and Development in Information Retrieval (SIGIR '13)*. 363–372.

[46] Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat Seng Chua. 2016. Discrete Collaborative Filtering. In *International Conference on Research and Development in Information Retrieval (SIGIR '16)*. 325–334.

[47] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *International Conference on Research and Development in Information Retrieval (SIGIR '13)*. 785–788.

[48] Lili Zhao, Zhongqi Lu, Sinno Jialin Pan, and Qiang Yang. 2016. Matrix Factorization+ for Movie Recommendation. In *International Joint Conference on Artificial Intelligence (IJCAI '16)*. 3945–3951.

[49] Tong Zhao, Julian McAuley, and Irwin King. 2014. Leveraging Social Connections to Improve Personalized Ranking for Collaborative Filtering. In *ACM International Conference on Information and Knowledge Management (CIKM '14)*. 261–270.

[50] Lei Zheng, Chun Ta Lu, Fei Jiang, Jiawei Zhang, and Philip S. Yu. 2018. Spectral Collaborative Filtering. In *Proceedings of the 12th ACM Conference on Recommender Systems (WSDM '18)*. 311–319.

[51] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2006. Learning with Hypergraphs: Clustering, Classification, and Embedding. In *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems (NIPS '06)*. 1601–1608.

Beyond Shortest Paths: Route Recommendations for Ride-sharing

Chak Fai Yuen
Dept. of Computer Science and Engineering, Indian Institute of Technology
New Delhi, India
Chak.Fai.Yuen.mcs17@cse.iitd.ac.in

Abhishek Pratap Singh
Dept. of Computer Science and Engineering, Indian Institute of Technology
New Delhi, India
Abhishek.Pratap.Singh.cs513@cse.iitd.ac.in

Sagar Goyal
Dept. of Computer Science and Engineering, Indian Institute of Technology
New Delhi, India
Sagar.Goyal.cs115@cse.iitd.ac.in

Sayan Ranu
Dept. of Computer Science and Engineering, Indian Institute of Technology
New Delhi, India
sayanranu@cse.iitd.ac.in

Amitabha Bagchi
Dept. of Computer Science and Engineering, Indian Institute of Technology
New Delhi, India
bagchi@cse.iitd.ac.in

ABSTRACT

In taxi ride-sharing, multiple customers are allotted to the same taxi as long as they are *compatible*, i.e., if none of them suffers a detour beyond a permissible threshold. To attract customers to ride-sharing, taxi operators promise a reduced fare upfront. As a result, if the taxi fails to pair the initial customer with additional compatible passengers, the taxi operator incurs a financial loss. Hence, it is important to ensure that the taxi finds compatible customers once it has picked up the initial customer. In the current scenario, the appearance of subsequent compatible customers is left to luck: a taxi moves along the shortest (or quickest) path for the existing customer and hopes to find additional compatible customers on its way. In this paper, we ask: *Is the shortest path the optimal path for ride-sharing?* To investigate this question, we develop a route recommendation algorithm called SHARE, which predicts the route with the highest probability of finding compatible customers, while staying within the detour limits. Through extensive evaluations on real datasets, we show that SHARE reduces failure rate in ride-sharing by up to 40%, while also improving waiting time for customers by 20%. Technically, the route recommendation problem is NP-hard. We overcome this bottleneck, by reducing the search space of the entire road network into a smaller directed acyclic graph, on which we perform *dynamic programming* to identify the optimal route. This strategy allows us to answer queries within 0.2 seconds, and thereby making SHARE deployable on real road conditions.

ACM Reference Format:
Chak Fai Yuen, Abhishek Pratap Singh, Sagar Goyal, Sayan Ranu, and Amitabha Bagchi. 2019. Beyond Shortest Paths: Route Recommendations for Ride-sharing. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313465

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313465

1 INTRODUCTION

With well known brands such as Uber and Didi Chuxing promoting *ride-sharing*, the taxi-service industry has become a prominent part of the internet-based "Sharing economy" [11]. In contrast to traditional ride-hailing services where a taxi is assigned at most one *ride-order* (request for a taxi), in ride-sharing, multiple *compatible* ride-orders are allocated to the same taxi. Two ride-orders are compatible if both of them can be serviced by a single taxi without taking a significant detour in either of their travel routes.

Ridesharing offers several benefits. From the riders' perspectives, their travel costs are reduced since they share a vehicle. For the ride-hailing service providers, ride-sharing allows them to better utilize their limited vehicles by servicing more customers, which results in more profits [42]. This is particularly useful during peak hours. In addition, ride-sharing is more environment-friendly as it helps reduce fuel consumption, carbon footprint and vehicular traffic congestion in cities. Owing to the numerous benefits of ride-sharing, the problem has received significant interest both from academia [10, 15, 25, 41] and industry [24, 37, 42]. In this work, we study the problem of *route recommendation* in ride-sharing environments.

The context is this: In the current technology scenario, when a taxi is allocated to a ride-order opted for ride-sharing, it picks the ride-order from the source location and travels to its destination in the shortest (or quickest) route. While travelling along this shortest route, the taxi may get lucky and be allocated to another compatible ride-order from a nearby location by the taxi service provider, and the taxi is re-routed to pick up the new request. To entice customers into opting for ride-sharing, taxi service providers such as Uber and Didi show the exact ride-sharing fare upfront even before the trip is started. Typically, this fare is much lower than what the customer would pay if an exclusive access to the taxi is requested. Since the

fare would not hike even if the taxi is not allocated to another compatible ride-order, from the service providers' perspective, they risk incurring a loss with the ride-sharing model. This risk depends on the probability of being allocated to a compatible ride-order while travelling on the shortest route of the first ride-order. Therefore, an important question arises: *Can we move beyond shortest paths by recommending a route that is close in length to the shortest path and offers much higher chance of finding a compatible ride-order?* In this paper, we investigate this question and present substantial evidence that moving beyond shortest paths is indeed a promising approach.

Existing works on ride-sharing optimization are in two primary aspects: taxi assignment [10, 15, 24, 25, 32, 41] and route planning [37, 42]. Given the current status of all taxis and the ride-orders, the taxi assignment problem looks into the aspect of mapping each ride-order to a taxi such that a particular objective function is maximized. This objective function can be maximizing profit, minimizing detour, etc. The route planning problem identifies the sequence of customer pick-ups and customer drop-offs that maximizes an objective function. In both aspects of works, it is assumed that the taxi travels on the shortest path between two consecutive pick-up/drop-off points. Furthermore, there is no predictive element: all computations are deterministic in nature and based solely on the known ride-orders and taxi locations.

In our problem, we attack the *route recommendation* problem. Specifically we ask: Can we *predict* a slightly longer path that significantly enhances the chances of finding compatible customers? Our recommendation achieves three goals. First, by enhancing the likelihood of finding compatible customers, it reduces the risk associated with ride-sharing for taxi service providers. Second, by ensuring the length of recommended path is close to the shortest path, the customer experience is not compromised. Third, by anticipating where future ride-orders are likely to come from, the recommended route reduces the waiting time for customers and thereby improves their experience. The key contributions of our work are as follows:

- We formulate a novel problem of route recommendation for taxi in the ride-sharing environment.
- We show that the proposed problem is NP-hard. We overcome the computational bottleneck by developing an algorithm called SHARE:*Ride-Sharing with History-Aware Route Recommendations*. SHARE reduces the exponential search space of all possible paths in the entire road network to a directed acyclic graph, on which we perform dynamic programming to identify the optimal path.
- We perform extensive experiments on real taxi datasets from New York and Singapore across a range of metrics. We show that the routes recommended by the proposed model receives up to 40% reduction in orders without ride-sharing than the shortest path. In addition, the recommended routes reduce waiting time for customers by 20% while improving number of passengers per kilometer for the taxi operator.

2 PROBLEM STATEMENT

DEFINITION 1 (ROAD NETWORK). *A road network is a directed graph $G = (V, E, \delta, \lambda)$, where V is the set of nodes representing geographical sites, $E \subseteq V \times V$ is the set of edges representing road segments between sites, a distance function $\delta : E \to \mathbb{R}$ representing the length (weight) of each road segment, and a spatial mapping λ representing the geographic locations of sites in the form (latitude, longitude) tuples.*

We use the notation $e = (u, v)$ to denote a road segment (edge) from node u to v. The length $\delta(e)$ of an edge e is the Haversine distance from u to v. A *route* corresponds to a *simple path* in the network, i.e., path without cycles. In the rest of this paper, we use the word "path" to refer to a simple path. The length of a path $P = \{e_1, \cdots, e_k\}$ over k edges is $\delta(P) = \sum_{\forall e_i \in P} \delta(e_i)$. A path may also be represented as a sequence of connected nodes. We use the notation $e_i \in P$ and $v_i \in P$ to denote the fact that path P goes through edge e_i and node v_i respectively. Finally, we use the notation $SP(u, v)$ to denote the length of the *shortest path* from node u to v.

DEFINITION 2 (RIDE ORDER). *A ride-order o is represented as $\langle o^s, o^d, o^t \rangle$ where $o^s \in V$ denotes the pick-up point, $o^d \in V$ denotes the drop-off point, and o^t denotes the time at which the order request is made.*

A path P *covers* order o if there exists a *sub-path* $P' \subseteq P$, such that P' originates at o^s and ends at o^d. We use the notation $\delta_o(P) = \delta(P')$ to denote the distance travelled to fulfill order o through path P. The *detour ratio* $dr(o, P)$ of P with respect to order o is the proportion of extra distance travelled compared to the shortest path, i.e.,

$$dr(o, P) = \frac{\delta_o(P)}{SP(o^s, o^d)}. \tag{1}$$

When two ride-orders are allocated to the same taxi, they may incur a detour. To ensure that these detours are not too long, taxi service providers allocate two orders to the same taxi only if both their detours are at most α [37, 42], where $\alpha \geq 1$ is a *detour threshold*. We call such ride-orders *compatible*.

DEFINITION 3 (COMPATIBLE RIDE ORDERS). *Let $o_1 = \langle o_1^s, o_1^d, o_1^t \rangle$ be a ride-order being serviced by taxi C and $o = \langle o^s, o^d, o^t \rangle$ is a new ride-order request. Furthermore, let $P_1 = \left\{o_1^s, \cdots, n\right\}$ be the path taken by C so far where n is its current location at time o^t. We call o compatible with o_1 if there exists a path P_2 originating at n such that P_2 covers o, $P_C = P_1 + P_2$ is a path that covers o_1 and both $dr(o_1, P) \leq \alpha$, $dr(o, P_2) = dr(o, P_C) \leq \alpha$.*

To generalize the above definition for arbitrary number of passengers, we assume each taxi is associated with a capacity of X passengers. Furthermore, for some m s.t. $0 \leq m \leq X$, $\mathbb{O} = \{o_1, \cdots, o_m\}$ is the set of orders currently being serviced by the taxi and P_1 is the path it has taken till now. A new request o can be allotted to this taxi if $Compatible(\mathbb{O}, o) = 1$.

$$Compatible(\mathbb{O}, o) = \begin{cases} 0 & \text{if } m = X \\ 1 & \text{if } \exists \text{ path } P_2 \text{ from current node } n, \\ & \text{such that } P_2 \text{ covers } o, \\ & \forall o_i \in \mathbb{O},\ P_C = P_1 + P_2 \text{ covers } o_i, \\ & \forall o_i \in \{\mathbb{O} \cup o\},\ dr(o_i, P_C) \leq \alpha \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

EXAMPLE 1. *Consider the scenario shown in Fig. 1. A taxi is at v_1 servicing $\mathbb{O} = \{o\}$, where $o = \langle v_1, v_{10}, o^t \rangle$. There are two other*

orders, $o_1 = \langle v_5, v_8, o^t \rangle$ and $o_2 = \langle v_9, v_3, o^t \rangle$, that are yet to be allotted to a taxi. Now, we will calculate $Compatible(\mathbb{O}, o_1)$ and $Compatible(\mathbb{O}, o_2)$.

For o_1, there exists a path $P_C = \{v_1, v_3, v_5, v_6, v_8, v_{10}\}$ of distance 23. This path P_C covers both o and o_1. Assume $\alpha = 1.5$. A sub-path $P'_C = \{v_5, v_6, v_8\} \subseteq P_C$ is the shortest path for o_1 so the detour ratio is 1. Furthermore, $SP(v_1, v_{10}) = 20$, so the detour ratio for o is $23/20 = 1.15 \leq 1.5$. Hence, $Compatible(\mathbb{O}, o_1) = 1$.

For o_2, there does not exist a path that satisfies Eq. 2 without violating the detour constraint. Hence, $Compatible(\mathbb{O}, o_2) = 0$.

Let C be a taxi of capacity X servicing a set of ride-orders $\mathbb{O}$, $|\mathbb{O}| < X$. Taxi C has just picked up the order request $o_m = \langle o_m^s, o_m^d, o_m^t \rangle \in \mathbb{O}$. Without loss of generality (*WLOG*), we assume the taxi has taken path P_1 till now. We now need to recommend the *best feasible* route to the taxi. Formally, this problem is defined as follows.

Problem 1 (Route Recommendation in Ride-sharing). *A path P in the network is feasible if (i) P originates at o_m^s, and (ii) $P_1 + P$ does not exceed the detour ratio for any of the orders in $\mathbb{O}$. The best feasible path is the one that has the highest number of expected compatible orders with respect to $\mathbb{O}$ at time o^t. More specifically, find the path P^* where*

$$P^* = \arg\max_P \left\{ E\left(P|\mathbb{O}, o^t\right) \mid P \text{ is a feasible path} \right\} \tag{3}$$

$E\left(P|\mathbb{O}, o^t\right)$ denotes the expected number of compatible passengers in P at time o^t.

The next task therefore is to mathematically quantify $E\left(P|\mathbb{O}, o^t\right)$ for any given feasible path P.

2.1 Computing Expected Number of Passengers

To compute the expected number of compatible passengers, we use the historical data $\mathbb{H}$ containing past ride-orders. We use the notation $\mathbb{H}_D \subseteq \mathbb{H}$ to denote the set of ride-orders made on a particular date D. Given a node v and current time o^t, we extract the set of compatible orders originating at v from $\mathbb{H}_D$ for each date D covered in the history.

$$\mathbb{H}_{D,v,\mathbb{O},o^t} = \{o' | o' \in \mathbb{H}_D,\ o'^s = v, \\ \|o^t - o'^t\| \leq \Delta, Compatible(\mathbb{O}, o') = 1\} \tag{4}$$

Here, Δ is a smoothing parameter on the time dimension. It is reasonable to set Δ to any value between 5 to 15 minutes. Therefore, the *expected* number of ride-orders arising from node v at time o^t is the average count across all dates in $\mathbb{H}$. Specifically,

$$E(v|\mathbb{O}, o^t) = \frac{1}{Y} \sum_{\forall \mathbb{H}_{D_i} \in \mathbb{H}} \left| \mathbb{H}_{D_i,v,\mathbb{O},o^t} \right| \tag{5}$$

where Y is the total number of days spanning $\mathbb{H}$.

Let $P = \langle v_1, \cdots, v_n \rangle$ be a feasible path. The expected number of compatible passengers in P is therefore

$$E(P|\mathbb{O}, o^t) = \sum_{\forall v_i \in P} E(v_i|\mathbb{O}, o^t) \tag{6}$$

Eq. 6 completes the problem formulation.

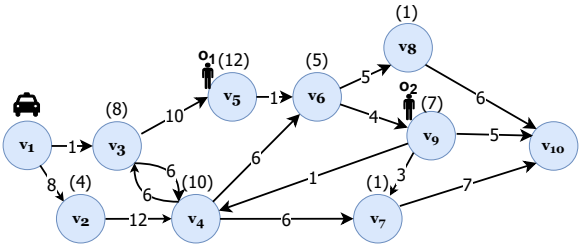


Figure 1: An illustration of a road network G. In this graph, the length of each edge is shown as edge weights. The numbers in brackets above each node denotes the expected number of compatible passengers, $E(v|\mathbb{O}, o^t)$, which we define in Sec. 2.1.

3 BASELINE: SHORTEST PATH

The baseline algorithm is to select the shortest path [37, 42]. There is no predictive element in this approach. Formally, let taxi C be servicing order set $\mathbb{O}$. We now introduce the concept of a *route plan.*

Definition 4 (Route Plan). *A* route plan, *RP, is a sequence of drop-off locations corresponding to each order in $\mathbb{O}$.*

A path P *satisfies* route plan RP if it visits the drop-off locations in the same sequence as in RP. Let $SP(RP)$ denote the shortest path satisfying route plan RP. The *shortest route plan* for $\mathbb{O}$, RP^*, is the route plan with the smallest shortest path. Mathematically,

$$RP^* = \arg\min_{RP} \{SP(RP) \mid RP \in \mathbb{RP}\} \tag{7}$$

where $\mathbb{RP}$ is the set of all possible route plans for $\mathbb{O}$. In the shortest path approach, the taxi therefore simply follows $SP(RP^*)$.

4 SHARE:RIDE-SHARING WITH HISTORY-AWARE ROUTE ReCOMMENDATIONS

The shortest path approach is blind to the need of finding compatible passengers on its way. Our formulation addresses this weakness.

4.1 Complexity Analysis

Definition 5 (Longest Path Problem). *Given a weighted graph $G = (V, E)$ with non-negative edge weights w_e, the Longest Path problem is to find the longest simple path from a source node (s) to a destination node (d).*

Definition 6 (Max-Weight Path Problem). *Given a graph $G = (V, E)$ with non-negative node weights w_v, the Max-Weight path problem is to find the maximum weight simple path from a source node (s) to a destination node (d).*

The route recommendation problem for ride-sharing is essentially the *Max-Weight Path* problem, where the weight of each node v is the expected number of compatible passengers at v (Eq. 5). The route with the highest number of expected compatible customers is therefore the Max-Weight path.
We also know that the Longest Path problem is NP-hard [6].

Theorem 1. *Finding the Max-Weight path is NP-Hard*

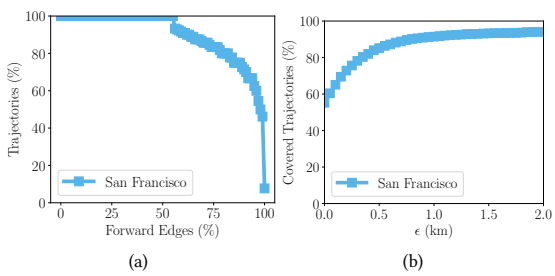


Figure 2: (a) Frequency distribution of trajectories against the percentage of forward edges. (b) Distribution of covered trajectories against the extra distance threshold ϵ.

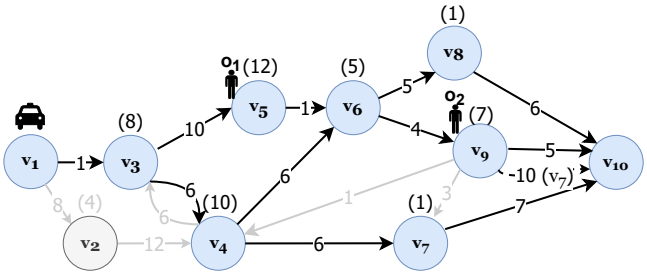


Figure 3: The subgraph D of the graph G in Fig. 1. The solid edges represent those that are part of E_D. The greyed-out edges and nodes represent those that are part of the original graph G but omitted from the D (See Ex. 2 for further details). The only dashed edge (v_9, v_{10}) represents an edge that is part of the extended DAG X (See Sec. 4.4) but not D.

Proof : Given an arbitrary instance of the Longest Path Problem, we reduce it to an instance of the Max-Weight Path problem through the following procedure. Consider the given graph $G = (V, E)$ in the Longest Path Problem, we design a graph $G' = (V', E')$ such that $\forall v \in V$, we have a node N_v in V' and $\forall e = (u, v) \in E$ we have a node N_e in V'. Furthermore, $\forall e = (u, v) \in E$ we have the edges $e_1 = (u, N_e)$ and the edge $e_2 = (N_e, v) \in E'$. We assign the weight w_e to all $N_e \in V'$, the weight 0 to all N_v and all the edges $e \in E'$ are un-weighted.

In G' the Max-Weight problem is to identify the path from a source s to a destination d with maximum weight. Since each node N_e selected in the path in G' corresponds to selecting the edge $e \in E$, if Max-Weight problem is solved, the resultant path is the longest path from s to d in graph G and hence, we would have solved the Longest Path Problem. □

Owing to the NP-hardness, an optimal algorithm is not feasible. This result motivates us to study real-world taxi trajectories and identify properties that may help reduce the search space.

4.2 An inspection of real-life trajectories

When a passenger is in a taxi, the taxi is expected to progressively get closer to the destination o^d through *forward* edges.

Definition 7 (Forward and Backward Edge). *Given a destination node o^d, we call an edge (u, v) to be a* forward edge *if $SP(u, o^d) > SP(v, o^d)$; otherwise, it is a* backward *edge.*

Taking a backward edge is a sub-optimal decision as it detours away from the destination, so we expect most real-life taxis move predominantly through forward edges. To verify this hypothesis, we analyze taxi trajectories in San Francisco [31] and compute the distribution of trajectories containing at least x% of forward edges. The details of the San Francisco dataset are provided in Sec. 5.1.1.

Fig. 2(a) presents the result. As can be seen, more than 55% of the trajectories have only forward edges. Even when a taxi travels on backward edges, the proportion of backward edges is generally small. For example, more than 80% of the trajectories have at least 80% forward edges. This result gives credence to our hypothesis that taxis progressively move closer to its destination in general.

4.3 Inducing a Directed Acyclic Graph on the Road Network

Motivated by the above result, we reduce the search space from all feasible paths to only those feasible paths that are comprised entirely of forward edges. To formalize this idea, let C be a taxi servicing order set $\mathbb{O}$ and have just picked up order $o \in \mathbb{O}$ from node o^s. Furthermore, let node o^{d*} be the next drop-off point according to the shortest route plan RP^* for $\mathbb{O}$. Then we extract the subgraph $D(V_D, E_D)$ from the entire road network $G(V, E)$, where

$$V_D = \left\{ v \in V \mid SP(v, o^{d*}) \leq SP(o^s, o^{d*}) \right\} \quad (8)$$

$$E_D = \{(u, v) \in E \mid u, v \in V_D, (u, v) \text{ is a forward edge}\} \quad (9)$$

Example 2. *Let us revisit graph G from Fig. 1. Fig. 3 presents the subgraph $D = (V_D, E_D)$ constructed from G corresponding to the order set $\mathbb{O} = \{o = \langle v_1, v_{10}, o^t \rangle\}$. Furthermore, only the solid edges in Fig. 3 are part of E_D. We illustrate the importance of the dashed edge (v_9, v_{10}) later in Sec. 4.4.*

In D, according to Eq. 8, $v_2 \notin V_D$ since $SP(v_2, v_{10}) > SP(v_1, v_{10})$. Moreover, according to Eq. 9, the edge (v_9, v_7) is omitted since $SP(v_7, v_{10}) = 7 > SP(v_9, v_{10}) = 5$.

Theorem 2. *$D(V_D, E_D)$ is a directed acyclic graph (DAG).*

Proof by contradiction. D is directed since G is directed. Let us assume that the set of edges $E_c = \{(v_1, v_2), (v_2, v_3) \cdots, (v_m, v_1)\}$ form a cycle in D. In D, an edge (u, v) from G is retained if and only if $SP(u, o^d) > SP(v, o^d)$. This leads to both $SP(v_1, o^d) > SP(v_m, o^d)$ and $SP(v_m, o^d) > SP(v_1, o^d)$, which is a contradiction. □

Thm. 2 has deep implication on our problem. Specifically, although the Max-Weight Path problem is NP-hard on a graph, on a DAG, it can be solved in polynomial time [33]. Specifically, let (d_1, o^{d*}) and (d_2, o^{d*}) be the two incoming edges to destination node o^{d*}. The longest path, $LP(o^s, o^{d*})$, from o^s to o^{d*} in a node-weighted graph is $LP(o^s, o^{d*}) = \max\{LP(o^s, d_1), LP(o^s, d_2)\} + weight(o^{d*})$. Since the Max-Weight Path problem in a DAG can be broken down into smaller problems, this can be solved in polynomial time using *dynamic programming (DP)*.

Unfortunately, this DP formulation cannot directly be applied to our problem. Specifically, we have two kinds of weights: the edge

weights denoting the road distance, and the node weights denoting the expected number of compatible customers. We need to find the Max-Weight path from o^s to o^{d*} in the domain of expected customers, such that the path, in terms of distance, does not violate the detour ratio (Eq. 1) of any of the orders in $\mathbb{O}$. We satisfy these dual needs through the following DP policy.

For each order $o_i \in \mathbb{O}$, let l_i be the length of the path they have travelled so far. The maximum amount of distance o_i can further travel in its path towards o^{d*} without violating the detour constraint is therefore $\overrightarrow{l_i} = \alpha \times SP(o_i^s, o_i^d) - l_i - SP(RP^*(o^{d*}, o_i^d))$, where $RP^*(o^{d*}, o_i^d)$ is the subsequence of the shortest route plan from o^{d*} to o_i^d. Therefore, any path P from o^s to o^{d*} is feasible if $\delta(P) \leq \overrightarrow{l_{min}}$, where $\overrightarrow{l_{min}} = \min\left\{\overrightarrow{l_i} \mid o_i \in \mathbb{O}\right\}$.

Next, we introduce the notation $c_{v,j}$ to denote the maximum expected number of compatible passengers among all paths of length j from o^s to some node v. Mathematically,

$$c_{v,j} = \arg\max_{\forall P \in \mathbb{P}_{v,j}} \left\{E(P|\mathbb{O}, o^t)\right\} \tag{10}$$

where $\mathbb{P}_{v,j} = \{P|\delta(P) = j, P \text{ is a path from } o^s \text{ to } v \text{ in } D\}$. We are now ready to define the DP. The initialization condition in the DP is

$$c_{v,j} = \begin{cases} 0 & \text{if } v = o^s, j = 0 \\ -\infty & \text{otherwise} \end{cases} \tag{11}$$

Now, the recurrence is defined as:

$$c_{v,j} = \max_{\forall (u,j') \in N_j(v)} \left\{c_{u,j'}\right\} + E\left(v \mid \mathbb{O}, o^t\right) \tag{12}$$

where $N_j(v) = \left\{(u, j') \mid (u, v) \in E_D,\ j = j' + \delta(u, v) \leq \overrightarrow{l_{min}}\right\}$ is the set of all incoming edge-length pairs that result in a path of length j without violating the detour ratio. The recurrence is computed using *memoization* for all nodes. The recommended path to o^{d*} corresponds to the path leading to $c^* = \max_{\forall j}\left\{c_{o^{d*},j}\right\}$.In addition, we also store the incoming neighbor from $N_j(v)$ that maximizes $c_{v,j}$. This is used to backtrack and identify the path providing the highest expected number of compatible passengers.

Example 3. *Table 1 presents the $c_{v,j}$ values for each node in the DAG D in Fig. 3. We assume source $o^s = v_1$ and destination $o^{d*} = v_{10}$. The shortest path from o^s to o^{d*} is of length 20. Assuming $\alpha = 1.5$, $l_{min} = 30$. Thus, $c_{v,j}$ matters only if $j \leq 30$.*

Consider calculating $c_{v_6,12}$, the possible (u, j') pairs are $N_{12}(v_6) = \{(v_5, 11), (v_4, 6)\}$. The maximum $c_{u,j'}$ is $c_{v_5,11} = 20$. Thus $c_{v_6,12} = c_{v_5,11} + E\left(v_6 \mid \mathbb{O}, o^t\right) = 20 + 5 = 25$.

At o^{d*}, the maximum $c_{v_{10},j}$ out of the feasible js is 32, which is highlighted in bold. The row corresponding to $\rightarrow$ in Table 1 indicates the incoming neighbor that maximizes the $c_{v,j}$ value. By backtracking using the incoming neighbors, we conclude that the optimal path P^* is $\{v_1, v_3, v_5, v_6, v_9, v_{10}\}$.

The computational complexity of the proposed DP remains exponential since, for each node v, the number of possible js can be as large as the number of possible ways to reach v from o^s, which is exponential. This implies an exponential number of columns in the table. The exponential cost can be avoided by moving from the continuous space to a discrete space. Particularly, each value

Table 1: The DP table corresponding to the DAG in Fig. 2(a) ($c_{v,j} = -\infty$ is omitted). Refer to *Ex.* 3 for further details.

v	v_1	v_3	v_4	v_5	v_6	v_6	v_7	v_8	v_8	v_9	v_9	v_{10}	v_{10}	v_{10}	v_{10}	v_{10}
j	0	1	7	11	12	13	13	17	18	16	17	20	**21**	22	23	24
$c_{v,j}$	0	8	18	20	25	23	19	26	24	32	30	19	**32**	30	26	24
$\rightarrow$	-	v_1	v_3	v_3	v_5	v_4	v_4	v_6	v_6	v_6	v_6	v_7	v_9	v_9	v_8	v_8

Algorithm 1 Path finding using Dynamic Programming in DAG

Input: Starting location o^s, destination o^{d*}, detour threshold α, number of bins Ω, Graph G
Output: Optimal path p^* in DAG
1: $D \leftarrow$ Construct DAG from G
2: $\forall v \in V_D$, compute $top-order(v)$
3: $c_{o^s,0} \leftarrow 0$
4: **for** $i \leftarrow 1$ **to** $top-order(o^{d*})$ **do**
5: $\quad V_i \leftarrow \{v \in V_D \mid top-order(v) = i\}$
6: $\quad$ **for each** $v \in V_i$ **do**
7: $\quad\quad \forall b,\ 1 \leq b \leq \Omega$, Compute $c_{v,b}$ (Eq. 12)
8: $c^* \leftarrow \max_{\forall b}\{c_{o^{d*},b}\}$
9: $P^* \leftarrow$ backtrack with c^* to get optimal path
10: **return** P^*

of j is assigned to bin $b = \left\lceil \frac{j}{\omega} \right\rceil$, where ω is the width of each bin. All aspects of the DP remains identical except (1) instead of computing $c_{v,j}$, we compute $c_{v,b}$; (2) instead of $N_j(v)$, we use $N_b(v) = \left\{(u, b') \mid (u, v) \in E_D,\ b = \left\lceil \frac{b' \times \omega + \delta(u,v)}{\omega} \right\rceil \leq \Omega\right\}$, where Ω is the total number of bins and the bin width ω is set to $\frac{\overrightarrow{l_{min}}}{\Omega}$. As we will see in our empirical evaluation (Sec. 5.4), $\Omega = 100$ provides a good balance between efficiency and quality.

Alg. 1 presents the pseudocode of the DP. After constructing the DAG D (line 1), we compute the *topological order* of each node in D (line 2). From the construction of D, it is guaranteed that the source o^s will have a topological order of 1 and o^{d*} will have the largest order. The nodes in D are processed according to their topological order (line 4) and $c_{v,b}$ for each node is computed (lines 5-7). By processing nodes in *topological order*, we ensure that all incoming neighbors of a node has already been processed. Finally, the path corresponding to the highest expected number of compatible passengers is returned (lines 8-10).

4.3.1 Properties: We next outline the key properties of Share.

Theorem 3. *The shortest path is one of the possible solutions.*

Proof. Consider the shortest path $P = \{v_1, \cdots, v_n\}$, where $v_1 = o^s$ and $v_n = o^{d*}$. Since it is the shortest path, $SP(v_i, o^{d*}) > SP(v_{i+1}, o^{d*})$ for $i < n$. Therefore, this path exists in D. □

Time Complexity: Topological sorting of all nodes in the DAG consumes $O(|V| + |E|)$ time. The number of states in the dynamic programming is $|V|\Omega$. To compute the value of each state, we consider all incoming edges. Hence, each edge is considered at most once per discrete distance leading to a complexity of $O((|V| + |E|) \cdot \Omega)$.

Space Complexity: Since we have $|V|\Omega$ states in the worst case, the space complexity is $O(|V|\Omega)$.

4.4 Improving the Search Space through Extended DAG

Our search space in the DAG is restricted to only paths formed by forward edges. Fig. 2(a) reveals that $\approx 55\%$ of the taxis conform to this assumption. Therefore, we ask: *Is it possible to incorporate more paths in the DAG by slightly relaxing the forward edge assumption?*

Whenever a taxi takes a backward edge from some node u, it moves further away from the destination, and therefore, incurs some *extra distance* over what would have been the shortest path from u. We hypothesize that the extra distance incurred by a taxi owing to a backward edge is small since no taxi would want to burn unnecessary fuel. *Extra distance* is formally defined as follows.

Definition 8 (Extra Distance). *Let (u, v) be a backward edge. Furthermore, let $P = \{u, v, \cdots, x\}$ be the sequence of nodes following the backward edge till the taxi reaches some node x such that $SP(x, o^{d*}) \leq SP(u, o^{d*})$. The extra distance, denoted as $eDist(u, v, o^{d*}) = \delta(P)$, is the distance travelled by the taxi to reach a node that is at least as close to the destination as u.*

To verify our hypothesis that backward edges are taken by a taxi only when the resulting extra distance is small, we revisit the San Francisco dataset and call a taxi trajectory T *covered* if $\forall (u, v) \in T$, either (1) (u, v) is a forward edge, or (2) $eDist\left(u, v, o^{d*}\right) \leq \epsilon$, where ϵ is a threshold and o^{d*} is the next drop-off point of T. ϵ controls when we call the extra distance owing to a backward edge *small*; when $\epsilon = 0$, only forward edges are allowed. Fig. 2(b) presents the results as ϵ is varied. As can be seen, at $\epsilon = 0.5km$, more than 85% trajectories are covered. We take inspiration from this result and propose the idea of an *extended* DAG $X = (V_X, E_X)$.

The nodes in the extended DAG is same as in the original DAG, i.e., $V_X = V_D$. The edge set $E_X = \{E_D \cup E_\epsilon\}$, where E_D is the set of forward edges (Eq. 9), and E_ϵ is the set of *extended* edges that capture paths through backward edges incurring an extra distance of at most ϵ. Formally, an extended edge is defined as follows.

Definition 9 (Extended Edge). *We add an* extended *edge between two nodes u, $x \in V_X$ if the following conditions hold:*

1. *$SP(x, o^{d*}) \leq SP(u, o^{d*})$, where o^{d*} is the next drop-off point according to RP^* for $\mathbb{O}$.*
2. *There exists a path $P = \{u, v, \cdots, x\}$ in the road network G such that (i) (u, v) is a backward edge, and (ii) $\delta(P) \leq \epsilon$. It is possible that multiple paths from u to x satisfy both conditions (i) and (ii). Consequently, multiple edges may exist between u and x.*

Since multiple edges may exist between two nodes in the extended DAG, we denote each edge with the tuple (u, x, P). For extended edges, P corresponds to the path satisfying condition (2) in Def. 9. On the other hand, in a forward edge (u, x, P), $P = \{u, x\}$. The *length* of any edge (u, x, P) is therefore $\delta(P)$. Condition (1) in Def. 9 ensures that the extended DAG remains acyclic. This enables us to extend Share from the forward edge assumption.

Example 4. *The extended DAG X of the network in Fig. 1 at $\epsilon = 10$ is shown in Fig. 3. The solid edges correspond to the forward edge, and the dashed edge from v_9 to v_{10} corresponds to an extended edge. This extended edge exists due to the path $P = \{v_9, v_7, v_{10}\}$. Now we can obtain a path $\{v_1, v_3, v_5, v_6, v_9, v_7, v_{10}\}$ with an optimal $c^* = 33$, which is higher than 32 obtained in plain DAG D.*

The size of the extended DAG is exponential in ϵ: All extended edges originating from node u can be identified by initiating a depth-first exploration from u, traversing a backward edge to v first, then exploring further edges until the total length exceeds ϵ or a node x is reached that is closer to the destination than u. Since the number of paths between any two nodes is potentially exponential, the worst case number of edges in E_ϵ is exponential in ϵ. Hence, constructing the extended DAG and solving the DP are both exponential in ϵ. However, by limiting ϵ to a small value such as $0.5km$, we can obtain a good balance between running time and quality. We further explore this aspect in Sec. 5.

Searching Algorithm: The dynamic programming (DP) applied on the DAG can also be applied on the extended DAG to identify the path having the largest number of expected compatible customers. However, there are two issues that need to be treated differently. First, each extended edge (u, x, P) corresponds to path P from the network. Thus, each extended edge (u, x, P) is replaced with P in the optimal path identified by the DP. Second, although any path in the extended DAG is simple, once extended edges are replaced with their corresponding paths, they may introduce repeated vertices. To give an example, consider two connected extended edges (u, x, P_1) and (x, z, P_2). It is possible that $\exists v \in V_X,\ v \in P_1$ and $v \in P_2$. To reject such paths, the recurrence is allowed to proceed only through incoming edges that do not introduce a cycle. To accommodate this additional constraint, we modify the definition of feasible incoming neighbors $N_j(x)$ of node x in Eq. 12 as follows.

$$N_j(x) = \{(u, j') \mid (u, x, P) \in E_X,\ j = j' + \delta(P) \leq \overrightarrow{l_{min}},\ P \cap path(c_{u,j}) = \emptyset\} \quad (13)$$

$path(c_{u,j})$ denotes the path that leads to $c_{u,j}$. Recall that this path can be identified through backtracking. Computing the intersection between two paths can be done in $O(|V|)$ time through hashing.

To summarize, the extended DAG includes paths with backward edges under the constraint that the backward edge does not cause an additional detour beyond ϵ than a path consisting of only forward edges. As shown in Fig. 2(b), more than 85% of real-life taxis do not take any backward edge that results in an extra distance of more than $0.5km$. The online recommendation pipeline is therefore as follows: every time a new order o is picked up by a taxi, we first compute the nearest drop-off location o^{d*} according to the shortest route plan RP^* (Recall Eq. 7). The extended DAG X is constructed with o^s and o^{d*} as the source and destination nodes and the feasible path with the highest number of expected customers is computed.

Time Complexity: As discussed earlier, the construction time of the extended DAG is exponential in ϵ and the time for executing the DP on this DAG is also similarly exponential in ϵ. Consequently, by limiting ϵ to a small value, we obtain a good balance between efficiency and quality.

Space Complexity: Since each backward edge corresponds to a path from the road network, and there could potentially be exponential number of paths, the number of extended edges is also exponential. However, in a practical scenario, under a small ϵ in the range $[0.1km, 0.7km]$, the space consumption is less than 0.4 GB on large metropolitan cities like New York and Singapore.

4.5 Infusing Additional Intelligence

The proposed prediction model is generic enough to accommodate further optimizations. We discuss some possibilities below. These optimizations are not part of SHARE, but may be added if preferred. We empirically evaluate their impact in Sec. 5.6

4.5.1 Non-linear weighting of nodes: The expected number of compatible customers in a path is simply the summation of the expected customers in each of the constituent nodes. This formulation does not incorporate the factor that finding a customer early in a path is much more profitable than in one of the later nodes. This aspect can be modeled by applying a non-linear weighting factor to each node. Specifically, we may modify Eq. 6 as:

$$E\left(P|\mathbb{O}, o^t\right) = \sum_{\forall v_i \in P} E\left(v_i|o^t\right) \cdot w_i \tag{14}$$

where w_i is the weight importance of node v_i. Under an exponential decay function, $w_i = e^{-\lambda \frac{dist_i}{\delta(P)}}$, where $dist_i$ is the distance travelled till node v_i in path P and λ is the decay parameter.

4.5.2 Regional Expectations: The expected number of compatible customers in a node v is derived from the historical pick-up data of v . In reality, a taxi can always deviate slightly from the path, pick up a customer, and then continue on its journey. To model the expected number of compatible customer from the region around a node instead of the node itself, we may modify Eq. 5 as:

$$E(v|\mathbb{O}, o^t) = \frac{1}{Y} \sum_{\forall \mathbb{H}_{D_i} \in \mathbb{H}} \sum_{\forall u \in R(v,\theta)} \left|\mathbb{H}_{D_i,u,\mathbb{O},o^t}\right| \tag{15}$$

where $R(v, \theta) = \{u \in V \mid SP(v, u) \leq \theta\}$ is the set of all nodes reachable within a distance of θ from v.

4.5.3 Incorporating Expected Time: If an order o is picked up by a taxi at time o^t, the expected number of customers at all nodes in a path $P = \{v_1, \cdots, v_n\}$ is computed conditioned on o^t. In reality, the taxi reaches any node $v_i \in P$ some time after o^t. To reflect the time component more accurately, we can compute the average commute time t_e for each edge e in the network. Next, one may modify Eq. 6 as follows to obtain better estimates.

$$E\left(P|\mathbb{O}, o^t\right) = \sum_{\forall v_i \in P} E\left(v_i \mid (o^t + time(P, v_i))\right) \tag{16}$$

where $time(P, v_i) = \sum_{j=1}^{i-1} t_{(v_j, v_{j+1})}$

5 EXPERIMENTS

In this section, we benchmark SHARE and establish that

- **Quality:** The path recommended by SHARE is better than the shortest path across a range of practical metrics.
- **Route Planning:** Using SHARE as the base approach, instead of shortest paths, reduces ride-sharing failure rates by up to 40% in the route planning problem for ride-sharing [42].
- **Efficiency:** SHARE consumes only ≈ 0.2 seconds to answer recommendation queries, and is therefore fast-enough for real-time recommendations on large metropolitan cities.

The code and datasets used can be downloaded from https://github.com/Ride-SHARE/SHARE.

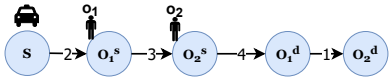


Figure 4: Illustrates the Passengers per km metric.

5.1 Experimental Setup

All experiments are implemented in Java and performed on a machine with Intel(R) Xeon(R) CPU 2.50GHz E5-2680 with 32GB RAM on Red Hat Enterprise Linux Server release 6.9.

5.1.1 Datasets. Table 2 summarizes the real taxi datasets used for our empirical evaluation. The road networks of all three cities have been obtained from OpenStreetMap [29].

•**New York [8]:** This dataset contains records of yellow and green taxis in the city of New York. Each record includes fields capturing pick-up and drop-off dates/times, locations and trip distances. From this dataset, we extract all pick-ups for yellow taxis in January, 2013 and construct our dataset.

•**Singapore [38]:** This dataset contains records of taxis from Singapore. Each record includes a series of tuples of the form $\langle Latitude, Longitude, TaxiID, DriverID, TaxiStatus\rangle$, which are recorded at 1-minute intervals. The $TaxiStatus$ field indicates whether the taxi is occupied or idling. We extract all pick-ups in January, 2012.

•**San Francisco [31]:** The dataset has been collected over a duration of one month and contains the trajectories and taxi pick-up and drop-off information from taxis in San Francisco.

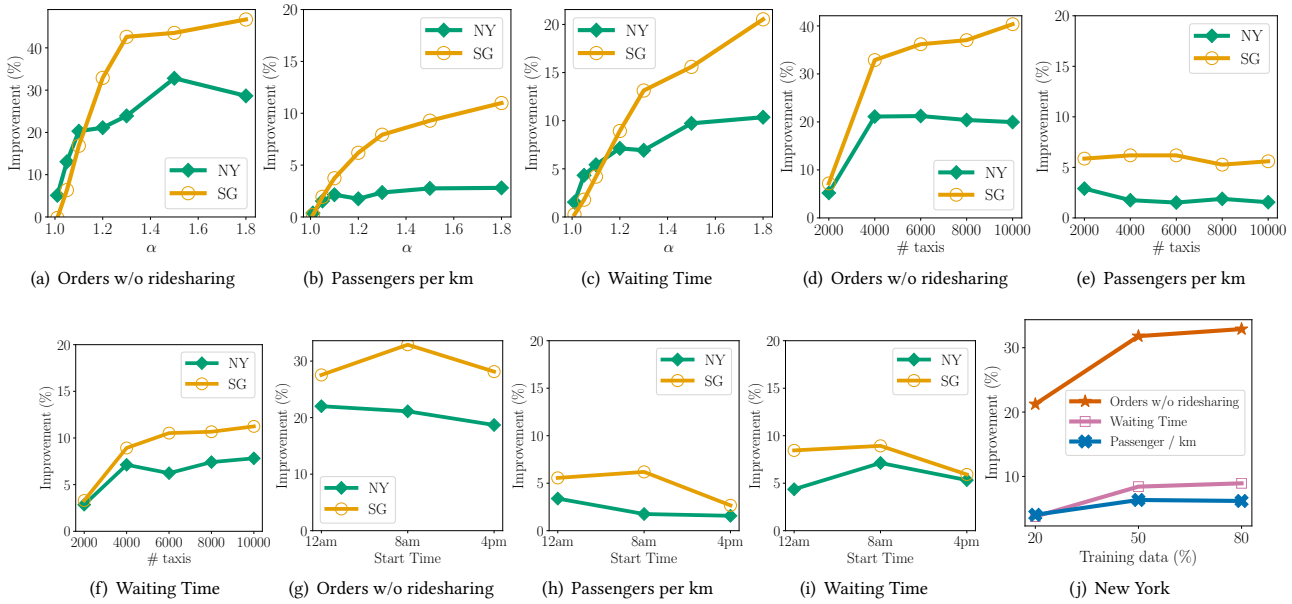
In all these datasets, the capacity of a taxi is not mentioned. We assume it to be 3 passengers.

5.1.2 Evaluation Framework: To evaluate the performance of SHARE, we partition each taxi pick-up dataset into two portions: the training set and the testing set. The recommendation model is learned from the training set and evaluated on the testing set. The training set contains a random 80% of all orders and remaining portion is assigned to test set. All taxis in the test set are initially randomly positioned in the road network. Next, each order $o = \langle o^s, o^d, o^t\rangle$ from the test set is picked sequentially according to o^t and added to a queue of pending orders. A pending order o is removed from the queue when a compatible taxi, i.e., a taxi servicing order set $\mathbb{O}$ such that $compatible(\mathbb{O}, o) = 1$ (Recall Eq. 2), $|\mathbb{O}| \geq 1$ passes through o^s. If it is detected that no compatible taxi can reach o^s within 5 minutes, then the closest empty taxi to o is assigned. To compute the expected time to reach a particular location, we use the mechanism described in Sec. 4.5.3.

For fast computation of shortest path distances, the road networks are indexed using the *two-hop* index structure[16].

Table 2: Summary of the datasets used

City	# nodes	# edges	# ride-orders
New York	61,298	141,373	14.77 million
Singapore	52,653	86,410	12.57 million
San Francisco	3,527	7,964	0.372 million



(a) Orders w/o ridesharing (b) Passengers per km (c) Waiting Time (d) Orders w/o ridesharing (e) Passengers per km

(f) Waiting Time (g) Orders w/o ridesharing (h) Passengers per km (i) Waiting Time (j) New York

Figure 5: Improvement of SHARE over Shortest Path (SP) against (a-c) α, (d-f) number of taxis and (g-i) time slot of the day (j) training dataset size.

5.1.3 Baselines: As discussed in Sec. 1, existing techniques for taxi ride-sharing primarily look into either route planning or taxi assignment. In both these problems, there is no element of prediction. Thus, a direct comparison is not possible. Therefore, We consider the following baselines.

•**Shortest Path:** Most major app-based taxi services such as Uber and Didi Chuxing move according to the shortest route plan [37, 42]. We therefore compare the performance of SHARE with shortest path.

•**Route Planning [42]:** Our goal in this comparison is to show that route planning algorithms can be improved by assuming the taxi moves along the path recommended by SHARE instead of the shortest path. Towards that end, we compute route based on SHARE and compare the performance with the state-of-the-art *Greedy* approach by Zheng et al. [42].

•**Optimal path:** Since the route recommendation problem is NP-hard, the extended DAG is used to approximate the optimal solution. To understand the approximation quality, we compare the path output by SHARE with the optimal path. We perform this evaluation only on the smaller San Francisco dataset since finding the optimal path has exponential computation cost.

5.1.4 Metrics: We use the following well-established metrics to quantify the performance of SHARE and other baselines.

Percentage of orders without ride-sharing [1]: Whenever a taxi is unable to find a compatible co-passenger for a ride-order, the taxi operator incurs a financial loss. Thus, we measure the percentage of ride-orders without any co-passenger. The lower the value, the better is the performance.

Passengers per kilometer (km) [9]: This metric measures the mean occupancy level per km. More specifically, let D_k be the total distance covered by all taxis with k passengers. Then the average passengers per km is $\frac{\Sigma_{\forall k} k D_k}{\Sigma_k D_k}$. A simple example is illustrated in Fig. 4. An empty taxi starts at S. Then, it picks up passengers at o_1^s and o_2^s after travelling 2km and 5km respectively, and finally, drops them off at o_1^d and o_2^d. In this scenario, the average number of passengers per km is $\frac{0\times2+1\times3+2\times4+1\times1}{2+3+4+1} = 1.2$. Clearly, the higher the value, the better is the performance.

Average waiting time [1]: Average waiting time measures the average number of minutes between an order being placed and a taxi reaching the pick-up location.

Rejection rate [42]: Rejection rate measures the number of ride-orders who remain in the queue of pending orders for more than a certain time threshold. Under a 15 minute time threshold, which is often used [42], we observe a rejection rate of less than 5% for both SHARE as well as the other baseline algorithms. Hence, we do not present results with respect to rejection rate.

5.1.5 Parameters: Share requires two key parameters, the extension threshold ϵ to construct the extended DAG, the number of bins Ω for discretization in the dynamic programming. Unless specifically mentioned, their default values are set to $\epsilon = 0.5km$ and $\Omega = 100$. Nonetheless, we also include experiments where we vary them explicitly and study their impact on the performance of SHARE.

5.2 Route Recommendations

First, we analyze the quality of the recommended routes. The performance of SHARE is compared with the shortest path in terms of the *Improvement (%)* over each metric in Sec. 5.1.4. Intuitively, *Improvement* measures the percentage by which SHARE improves upon shortest path. A higher *Improvement* signifies a better performance of SHARE. Mathematically, let $SHARE_M$ and SP_M be the

values obtained by Share and shortest path respectively in metric M. The *Improvement* in M is

$$Improvement\ (\%) = \frac{SP_M - SHARE_M}{SHARE_M} \times 100^1 \quad (17)$$

Figs. 5(a)-5(c) present the results as the detour parameter α is varied. Across all three metrics, Share improves upon shortest path. The improvement is most stark in the reduction of orders without ride-sharing (Fig. 5(a)); in New York (NY) Share is up to 30% better, while in Singapore (SG) the reduction is as high as 42%. This result highlights that Share is indeed able to learn from the historical data and recommend paths that lead to more ride-sharing. In contrast, in shortest paths, taxis solely depend on their luck since the route selection is done purely based on distance.

In its effort to maximize the number of expected customers, Share may recommend slightly longer routes. It is therefore important to ensure that this extra distance adds value to the taxi operators. The second metric, which compares the distance travelled per customer, evaluates this aspect. Fig. 5(b) reveals that Share leads to more customer density per km.

The third dimension that we measure is the amount time customers spend waiting for the taxis to arrive at their pick-up locations. We expect the waiting time to reduce in Share since we predict the locations where more requests are likely to originate from, and accordingly select routes that provides strategic advantage to taxis. This behavior is prominently visible in Fig. 5(c), where the reduction is as high as 20% in SG.

Two key patterns emerge from the results in Figs. 5(a)-5(c). First, the improvements are consistently higher in SG than in NY. A deeper look into the data reveals that the number of ride-orders is higher in NY and more importantly, highly concentrated in the smaller geographical area of Manhattan. In contrast, the orders are much more spread out in SG. When the density of ride-orders is very high in a particular region, like in Manhattan, any route has almost equal chance of finding compatible passengers. Thus, the improvement of Share in NY over shortest paths is not as prominent as in SG.

The second pattern that emerges from Figs. 5(a)-5(c) is that the improvement in general increases with α. Naturally, with higher α, the search space increases and Share has more scope to improve upon shortest paths.

Next, we benchmark the performance of Share as the number of taxis is explicitly varied. In this experiment, we randomly position C taxis in the city and initiate the simulation on the test set. Next, C is varied on the x-axis and *Improvement* is measured. We set $\alpha = 1.2$ in this experiment. As visible in Figs. 5(d)-5(f), the results follow a similar trend as earlier: Share is consistently better than shortest path and the improvement is higher in SG. As number of taxis increases, the improvement in orders without ridesharing and waiting time increases. When the number of taxis is small, the ratio of orders to taxis is high and hence there is a higher chance that a taxi would find a customer. But when the number of taxis increases, the taxis compete among each other for the same passenger. Hence route recommendation becomes important for a taxi to travel on a route which has a higher probability of picking up passengers. Thus, the improvement of Share over shortest paths increases.

In the next experiment, we analyze the performance at three different time slots of the day: 12AM-1:30AM (least busy), 8AM-9:30AM (office going traffic), and 4PM-5:30PM (home-bound). As expected, Share consistently outperforms shortest path. While the improvement of Share is relatively stable across different time slots of the day, a slight dip is observed in the last time slot of 4PM-5:30PM. This slot has the highest customer density and as we observed earlier, the competitive advantage of Share decreases when the ratio of orders to number of taxis is higher.

Finally, we analyze the performance of Share against the size of the training dataset in Fig. 5(j). As expected, the performance improvement begins to saturate as the training dataset size increases.

5.3 Taxi Assignment and Route Planning

The entire ride-sharing pipeline can be divided into three key stages: assigning taxis to orders, constructing the route plan that decides the sequence of order pick-ups and drop-offs, and finally, selecting the routes between each of the order pick-up and drop-off locations as decided in the route plan. While Share directly targets the third stage of the ride-sharing pipeline, we show that Share also makes the task of the other two stages easier.

In route planning, given the current positions of all taxis, and the pending order requests, the task is to dispatch the order into an optimal route plan of a taxi. The algorithm must ensure all constraints are met such as respecting the taxi capacity and detour ratio. While we allot the closest empty compatible taxi to a pending ride-order, in which only simple taxi assignment and no route planning is involved, more sophisticated algorithms have been proposed [42]. In this experiment, we compare the performance of Share with the state-of-the-art algorithm by Zheng et al. [42], where the assumption is that taxis always move along the shortest path.

Fig. 6(a) presents the results. As can be seen, Share has better performance across all three metrics. Note that the improvement is more prominently visible at lower values of α, which is the detour limit. When the detour limit is low, a taxi does not have much flexibility to re-route and pick up a passenger that is not already close to its current position. With Share, a taxi is more likely to be close to a new order since Share strategically chooses the routes based on prior historical patterns rather than travels through the shortest path. Consequently, the percentage of ride-sharing improves, waiting time decreases and passengers per km goes up, when compared to the shortest path route in conjunction with the route planning algorithm of Zheng et al. [42].

We also note that while the improvement of Share increases with α in Figs. 5(a)-5(c), in Fig. 6(a) the improvement decreases. This follows from the fact that in the route planning experiment, a taxi is allowed to re-route and pick up a pending order that is not directly on its path, but can be serviced without violating the detour limit of α. In the previous experiments, an order is serviced only by taxis that goes directly through the pick-up location. All in all, the results reveal that regardless of the model used, Share leads to substantially improved performance over shortest paths.

[1]For the Passengers per km metric, a higher value is better. Thus, the numerator is changed to $SHARE_M - SP_M$

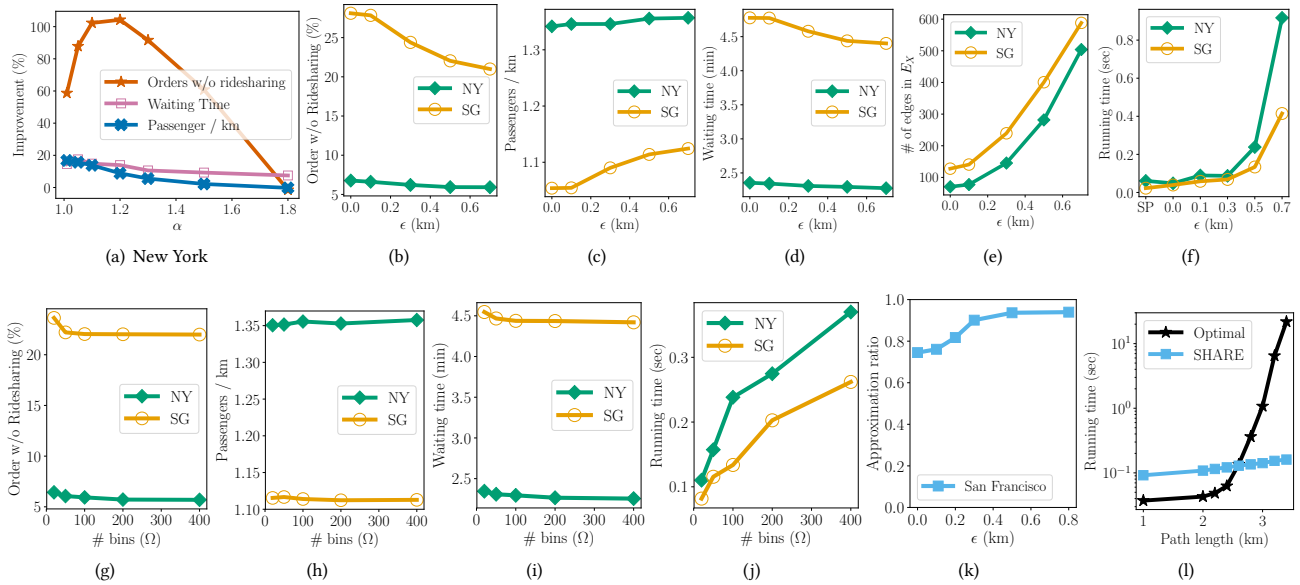


Figure 6: (a) Performance comparison of SHARE with the algorithm proposed by Zheng et al. [42]. Impact of extension threshold on (b-d) quality, (e) size of extended DAG, and (f) running time. Impact of number of bins on (g-i) quality and (j) running time. Comparison of SHARE with the optimal algorithm in term of (k) quality and (l) running time.

5.4 Impact of Input Parameters

Figs. 6(b)-6(d) present the impact of ϵ on the quality of SHARE as measured by the three metrics. We observe an improvement across all three metrics with higher values of ϵ. This is natural since the search space increases with increase in ϵ and this property is evident from Fig. 6(e), where we measure the number of edges in the extended DAG against ϵ. This improvement in quality, however, comes at the cost of higher running time since more paths need to be processed. Specifically, the number of edges increases exponentially with ϵ, which leads to an exponential increase in running time as well (Fig. 6(f)). Given all these factors, $\epsilon = 0.5km$ is suggested as the optimal choice since the quality saturates beyond this point, while the running time increases exponentially. Fig. 6(f) also shows that at $\epsilon = 0.5$, recommendations can be made within ≈ 0.2 seconds. This result substantiates our claim that SHARE is effective and fast enough to handle the load of any metropolitan city.

Next, we analyze the impact of number of bins, Ω, on the quality and running time. Figs. 6(g)-6(i) show that the quality remains stable against Ω. The running time, however, increases with number of bins, and thereby, justifying the choice of $\Omega = 100$.

5.5 Approximation Quality

SHARE approximates the longest path problem by searching only within the extended DAG. To diagnose the approximation quality, we plot the approximation ratio against the extension threshold ϵ. The approximation ratio is quantified as $\frac{w_{SHARE}}{w^*}$, where w^* and w_{SHARE} are the expected number of compatible passengers on the optimal path and the path recommended by SHARE respectively. Since computing w^* is NP-hard, we perform this analysis on the smaller San Francisco dataset. Fig. 6(k) shows that at $\epsilon = 0.5$, the approximation quality is near-optimal with a ratio larger than 0.95.

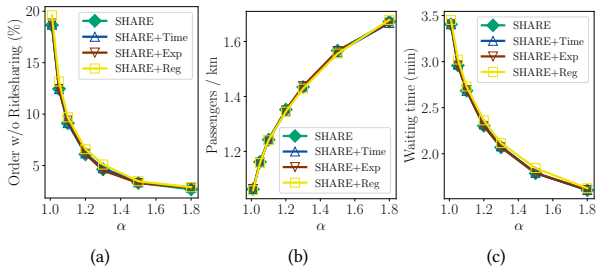


Figure 7: Impact of additional intelligence in New York (Time = Incorporate expected time, Exp = Exponential decay, Reg = Regional expectations).

The utility of the extended DAG is further highlighted by the running time analysis in Fig. 6(l). In this plot, we study the running times of the optimal algorithm and SHARE as a function of the distance between the source and destination nodes. As visible, SHARE is more than two orders of magnitude faster at $\epsilon = 0.5$, while providing near-optimal quality.

5.6 Impact of Additional Intelligence

In Sec. 4.5, we discussed some strategies to further improve the prediction model. Figs. 7(a)-7(c) present the impact of these various strategies. In these figures, SHARE+Time, SHARE+Exp, and

Share+Reg correspond to Sec 4.5.3, Sec. 4.5.1, and Sec. 4.5.2 respectively. In our experiments, we do not observe a noticeable improvement with any of these strategies except Share+Reg marginally improves in the metric of Passengers per km.

6 RELATED WORK

Historical trajectory data has been used for an array of smart-city applications such as anomaly detection[2], infrastructure improvement [20, 26–28] and community detection [34, 35]. In this work, we use historical data to recommend routes to partially occupied taxis such that their chances of finding ride-sharing passengers improve. Existing work on ride-sharing has focused on primarily two aspects: *taxi assignment* and *route planning*.

The taxi assignment approach works to optimize metrics such as profit, detour, etc. by assigning the right taxi with spare capacity to compatible orders [10, 15, 24, 25, 41]. Most recently, Ta et. al. work with a typical solution paradigm for this approach: they solve the assignment problem using an algorithm based on bipartite matching [36]. The focus on assignment makes the ride-sharing problem basically a generalization of the classical *dial-a-ride problem (DARP)* [5]One key difference of DARP with taxi ride-sharing is that in the ride-sharing partially occupied taxis can also be assigned to fulfill orders. The route planning approach assumes that a number of orders and taxis are available and it seeks to dispatch taxis to orders in a sequence computed such that the taxis can fulfill the orders while optimizing the appropriate metrics [37, 42]. This approach is a contemporary version of the classical vehicle routing problem [21] and typically involves solving a variant of the travelling salesman problem. Zheng et. al. apply a greedy method to overcoming this difficulty in [42], whereas Tong et. al. take a dynamic programming approach [37]. In the same line is the work by Cheng et. al. [4], which adds the dimension of rider utility and social connections. Some works take a hybrid approach of choosing an assignment and then repositioning empty taxis after order drop offs [1]. All these works, however, share one feature: An order once picked up is dropped off by following the shortest path between pick-up point and destination. And this is where our approach diverges.

Our method seeks to recommend good routes based on historical data. Historical data has been used to address various other problems related to taxis in the literature [13, 40, 41]. We refer the reader to den Hoedt's Masters thesis for a fuller discussion [7].

Our problem is a variant of the NP-hard longest path problem which cannot be approximated to with a factor of $\exp(\theta(\log^{1-\epsilon} n))$ for any $\epsilon > 0$ and a graph with n nodes [17]. So, we turn to heuristics without good approximation guarantees. Our basic optimization problem most resembles the orienteering problem which is a variant of the longest path that seeks to collect the maximum benefit on a graph using a path of given length [14]. Given the approximability constraints of the problem research on this problem has focused on meta-heuristics that work for small instances [3, 19, 25]. A recent work in this line also uses a dynamic programming approach like we do, but the formulation and approach is very different [25].

In the domain of route recommendations, it has been studied for both the cab service industry [12, 38] as well as itinerary recommendations [18, 22, 23, 30, 39]. These techniques, however, do not apply to our problem due to the recommendation setting being different.

7 CONCLUSION

In this paper, we established that it is indeed possible to learn from historical data and recommend routes for ride-sharing in taxis that significantly enhances the likelihood of finding more customers.The key weakness of the current taxi-sharing paradigm is that discounted fares are offered to promote sharing, which in turn, leads to a risk of revenue loss if sufficient number of orders are not allocated for ride-sharing. Till now, no effort has been made on the routing front to mitigate this risk, a shortcoming that we have endeavoured to address. Our route recommendation algorithm, Share, reduces failure rates in finding ride-sharing partners by as much as 40% when compared to the shortest paths used as default currently. In addition, it improves mean occupancy level and waiting times for customers by up to 20%. These excellent results are obtained by reducing the route recommendation problem to the classical problem of finding the *longest path* in a network. Share overcomes the scalability challenges posed by this NP-hard problem through an approximation algorithm based on *dynamic programming*. Extensive evaluation on real datasets demonstrate that Share provides near-optimal accuracy, while limiting the running time to within 0.2 seconds and thereby making it deployable on any major metropolitan city.

REFERENCES

[1] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus. 2017. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proc. Nat. Acad. Sci. United States* 114, 3 (2017), 462–467.

[2] Prithu Banerjee, Pranali Yawalkar, and Sayan Ranu. 2016. Mantra: a scalable approach to mining temporally anomalous sub-trajectories. In *ACM SIGKDD*. 1415–1424.

[3] I-Ming Chao, Bruce L. Golden, and Edward A. Wasil. 1996. A fast and effective heuristic for the orienteering problem. *European Journal of Operational Research* 88, 3 (1996), 475 – 489.

[4] P. Cheng, H. Xin, and L. Chen. 2017. Utility-Aware Ridesharing on Road Networks. In *Proc. SIGMOD '17*. 1197–1210.

[5] Jean-François Cordeau and Gilbert Laporte. 2007. The dial-a-ride problem (DARP): Models and algorithms. *Annals OR* 153 (06 2007), 29–46.

[6] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms, Third Edition* (3rd ed.). The MIT Press.

[7] Martijn den Hoedt. 2016. *Taxi ride scheduling and pricing using historical data*. Master's thesis. Delft University of Technology.

[8] Dan Donovan, Brian; Work. 2016. New York City Taxi Trip Data (2010-2013). https://doi.org/10.13012/J8PN93H8.

[9] P. M. d'Orey, R. Fernandes, and M. Ferreira. 2012. Empirical evaluation of a dynamic and distributed taxi-sharing system. In *15th Intl. IEEE Conf. Intelligent Transportation Systems*. 140–146.

[10] Esteban Feuerstein and Leen Stougie. 2001. On-line single-server dial-a-ride problems. *Theoretical Computer Science* 268, 1 (2001), 91 – 105. https://doi.org/10.1016/S0304-3975(00)00261-9 On-line Algorithms '98.

[11] Koen Frenken and Juliet Schor. 2017. Putting the sharing economy into perspective. *Environmental Innovation and Societal Transitions* 23 (2017), 3 – 10. Sustainability Perspectives on the Sharing Economy.

[12] Nandani Garg and Sayan Ranu. 2018. Route Recommendations for Idle Taxi Drivers: Find Me the Shortest Route to a Customer!. In *ACM SIGKDD*. ACM, 1425–1434.

[13] Yong Ge, Hui Xiong, Alexander Tuzhilin, Keli Xiao, Marco Gruteser, and Michael Pazzani. 2010. An Energy-efficient Mobile Recommender System. In *Proc. 16th ACM Intl. Conf. Knowledge Discovery and Data Mining (KDD '10)*. ACM, New York, NY, USA, 899–908.

[14] Aldy Gunawan, Hoong Chuin Lau, and Pieter Vansteenwegen. 2016. Orienteering Problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research* 255, 2 (2016), 315 – 332.

[15] Yan Huang, Favyen Bastani, Ruoming Jin, and Xiaoyang Sean Wang. 2014. Large Scale Real-time Ridesharing with Service Guarantee on Road Networks. *Proc.*

VLDB Endow. 7, 14 (Oct. 2014), 2017–2028.

[16] Minhao Jiang, Ada Wai-Chee Fu, Raymond Chi-Wing Wong, James Cheng, and Yanyan Xu. 2014. Hop Doubling Label Indexing for Point-to-Point Distance Querying on Scale-Free Networks. *CoRR* abs/1403.0779 (2014). arXiv:1403.0779 http://arxiv.org/abs/1403.0779

[17] D. Karger, R. Motwani, and G. D. S. Ramkumar. 1997. On approximating the longest path in a graph. *Algorithmica* 18 (1997), 82–98.

[18] Arzoo Katiyar, Arnab Bhattacharya, and Shubhadip Mitra. 2014. Efficient and Effective Route Planning in Road Networks with Probabilistic Data Using Skyline Paths. In *IKDD*. 3:1–3:10.

[19] Liangjun Ke, Claudia Archetti, and Zuren Feng. 2008. Ants Can Solve the Team Orienteering Problem. *Comput. Ind. Eng.* 54, 3 (April 2008), 648–665.

[20] Vinay Kolar, Sayan Ranu, Anand Prabhu Subramainan, Yedendra Shrinivasan, Aditya Telang, Ravi Kokku, and Sriram Raghavan. 2014. People In Motion: Spatio-temporal Analytics on Call Detail Records. In *COMSNETS*. 1–4.

[21] G. Laporte. 1992. The vehicle routing problem: An overview of exact and approximate algorithms. *Eur. J. Operations Research* 59, 3 (1992), 345–358.

[22] Wengen Li, Jiannong Cao, Jihong Guan, and Shuigeng Zhou. 2016. Retrieving Routes of Interest Over Road Networks. In *Web-Age Information Management*. 109–123.

[23] Yujiao Li, Weidong Yang, Wu Dan, and Zhipeng Xie. 2015. Keyword-Aware Dominant Route Search for Various User Preferences. In *Database Systems for Advanced Applications*. 207–222.

[24] S. Ma, Y. Zheng, and O. Wolfson. 2013. T-share: A large-scale dynamic taxi ridesharing service. In *29th Intl. IEEE Conf. Data Engineering (ICDE '13)*. 410–421.

[25] Zhibei Ma, Kai Yin, Lantao Liu, and Gaurav S. Sukhatme. 2017. A spatio-temporal representation for the orienteering problem with time-varying profits. In *Proc. IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (IROS '17)*. 6785–6792.

[26] Sourav Medya, Sayan Ranu, Jithin Vachery, and Ambuj Singh. 2018. Noticeable network delay minimization via node upgrades. *Proceedings of the VLDB Endowment* 11, 9 (2018), 988–1001.

[27] Shubhadip Mitra, Sayan Ranu, Vinay Kolar, Aditya Telang, Arnab Bhattacharya, Ravi Kokku, and Sriram Raghavan. 2015. Trajectory aware macro-cell planning for mobile users. In *INFOCOM*. 792–800.

[28] Shubhadip Mitra, Priya Saraf, Richa Sharma, Arnab Bhattacharya, Sayan Ranu, and Harsh Bhandari. 2017. NetClus: A scalable framework for locating top-k sites for placement of trajectory-aware services. In *ICDE*. 87–90.

[29] OpenStreetMap contributors. 2017. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org.

[30] Shiladitya Pande, Sayan Ranu, and Arnab Bhattacharya. 2017. SkyGraph: retrieving regions of interest using skyline subgraph queries. *Proceedings of the VLDB Endowment* 10, 11 (2017), 1382–1393.

[31] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. 2009. CRAWDAD dataset epfl/mobility (v. 2009-02-24). Downloaded from http://crawdad.org/epfl/mobility/20090224. https://doi.org/10.15783/C7J010

[32] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti. 2014. Quantifying the benefits of vehicle pooling with shareability networks. *Proc. Nat. Acad. Sci. US* 111, 37 (2014), 13290–13294.

[33] R. Sedgewick and K. D. Wayne. 2011. *Algorithms* (4th ed.). Addison Wesley Professional.

[34] Jieming Shi, Nikos Mamoulis, Dingming Wu, and David W Cheung. 2014. Density-based place clustering in geo-social networks. In *SIGMOD*. 99–110.

[35] Shivam Srivastava, Shiladitya Pande, and Sayan Ranu. 2015. Geo-social clustering of places from check-in data. In *ICDM*. 985–990.

[36] Na Ta, Guoliang Li, Tianyu Zhao, Jianhua Feng, Hanchao Ma, and Zhiguo Gong. 2018. An Efficient Ride-Sharing Framework for Maximizing Shared Route. *IEEE T. Knowl Data Eng* 30, 2 (February 2018), 219–233.

[37] Yongxin Tong, Yuxiang Zeng, Zimu Zhou, Lei Chen, Jieping Ye, and Ke Xu. 2018. A Unified Approach to Route Planning for Shared Mobility. *Proc. VLDB Endow.* 11, 11 (July 2018), 1633–1646.

[38] Tanvi Verma, Pradeep Varakantham, Sarit Kraus, and Hoong Chuin Lau. 2017. Augmenting Decisions of Taxi Drivers through Reinforcement Learning for Improving Revenues. In *ICAPS*, Vol. 27. 409–417.

[39] Pranali Yawalkar and Sayan Ranu. 2019. Route Recommendations on Road Networks for Arbitrary User Preference Functions. In *ICDE*.

[40] Nicholas Jing Yuan, Yu Zheng, Liuhang Zhang, and Xing Xie. 2012. T-Finder: A Recommender System for Finding Passengers and Vacant Taxis. *IEEE T. Knowl. Data Eng* 25, 10 (September 2012), 2390–2403.

[41] Mingyue Zhang, Jianxun Liu, Yizhi Liu, Zhenyang Hu, and Liang Yi. 2012. Recommending Pick-up Points for Taxi-drivers Based on Spatio-temporal Clustering. In *Proc. 2nd Intl. Conf. Cloud and Green Computing (CGC '12)*. IEEE Computer Society, Washington, DC, USA, 67–72.

[42] Libin Zheng, Lei Chen, and Jieping Ye. 2018. Order dispatch in price-aware ridesharing. *Proc. VLDB Endow.* 11, 8 (April 2018), 853–865.

Detecting Low Self-Esteem in Youths from Web Search Data

Anis Zaman
Department of Computer Science
University of Rochester
Rochester, NY
azaman2@cs.rochester.edu

Rupam Acharyya
Department of Computer Science
University of Rochester
Rochester, NY
racharyy@cs.rochester.edu

Henry Kautz
Department of Computer Science
University of Rochester
Rochester, NY
kautz@cs.rochester.edu

Vincent Silenzio
School of Medicine
University of Rochester
Rochester, NY
v.m.silenzio@rochester.edu

ABSTRACT

Online behavior leaves a digital footprint that can be analyzed to reveal our cognitive and psychological state through time. Recognizing these subtle cues can help identify different aspects of mental health, such as low self-esteem, depression, and anxiety. Google's web search engine, used daily by millions of people, logs every search query made by a user, which is accessible through a platform called Google Takeout. Previous researchers have made efforts to detect and predict behaviors associated with depression and anxiety from web data, but only at a population level. This paper fills in the gap of looking into signs of low self-esteem, a condition that work in a vicious cycle with depression and anxiety, at an individual level by looking into Google search history data. We target college students, a population prone to depression, anxiety, and low self-esteem, and ask to take mental health assessment survey along with their individual search history. Textual analysis show that search logs contain strong signals that can identify individuals with current low self-esteem. For example, participants with low self-esteem have fewer searches pertaining to family, friend, and money attributes; and we also observed differences in the search category distribution, over time, when compared with individuals with moderate to high self-esteem. Using these markers we were able to build a probabilistic classifier that can identify low self-esteem conditions, based on search history, with an average F1 score of 0.86.

KEYWORDS

low self-esteem, mental health, youths, college students, individual search logs

ACM Reference Format:
Anis Zaman, Rupam Acharyya, Henry Kautz, and Vincent Silenzio. 2019. Detecting Low Self-Esteem in Youths from Web Search Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313557

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313557

1 INTRODUCTION

Mental health can be viewed as a complex interaction between biological, genetic, social, and environmental factors. Due to the multifaceted nature and stigma of mental health, people are likely to rely on online services to investigate their problems in the World Wide Web [24, 50]. Google search is one such platform that is used by millions of people everyday and approximately 3.5 billion searches are made every day with 40,000 queries per second[1]. Hence, online searches may leave digital footprints that can be analyzed to reveal our cognitive and emotional state through time. Detecting these subtle cues can help therapists and others better recognize different components of mental health such as anxiety, stress, depression, and low self-esteem. Due to the complex nature of mental health and various factors associated to it, identifying individuals who are at risk or need help has been a challenge in the mental health research community [4, 8, 45]. For example, among college students, low self-esteem and stress are conditions that are more likely to correlate with suicidal ideation [51].

Our work brings significant new approaches and insights to the table: (i) Self-esteem has been rarely addressed in the AI literature before as part of the discussion of depression, and our approach provides a more expansive parsing of the conceptual frameworks in which constructs such as "depression," "self-esteem," and "suicidal ideation" are embedded [48]. (ii) We provide a scalable, lightweight solution that can be deployed at the point-of-care to use immediately available search history data to provide instantly actionable therapeutic targets for clinicians. (iii) Our work opens a way for clinicians to obtain access to patients' internal mental health states and traits, with high fidelity and accuracy far beyond the limits of traditional self-reported data: we are, in fact, now incorporating the approach into planned clinical trials.

Researchers have been using social media data to identify and predict different mental health conditions and associated factors [1, 6, 15, 15, 40]. Twitter, Facebook, Reddit, Instagram, and Weibo are some of the common data sources exploited by researchers to explore signs of depression, anxiety, suicidal ideation, and other related conditions.

With only a few exceptions [11, 12], these prior studies have not had access to ground-truth data about the subjects' mental health as assessed by a validated clinical instrument. While they make

[1] http://www.internetlivestats.com/google-search-statistics/

claims about depression at a population or individual level, they cannot justify those claims in a manner that would be actionable by a healthcare professional.

Another limitation of most prior work is that data is only gathered from users of particular social media sites. Because users self-select for particular sites, there may well be confounding factors that influence both the users' mental health and their online behavior. For example, people who routinely broadcast their thoughts on Twitter may have different ways of talking about depression than more private individuals. A further confounding factor is that users may choose to self-censor when posting to social media [22].

In this paper we investigate observable textual signatures related to current low self-esteem condition using both Google search histories and answers to a validated mental health assessment survey. For search, as opposed to use of social media, users do not self-select or face the same temptations to self-censor. Another advantage of search data is that the nature and volume of search queries give temporal behavioral information about a user. While some prior researchers have analyzed search logs for large aggregate populations using Google Trends [2, 55], applying their results to individuals may result in ecological errors. In addition, Google Trends data cannot be used to understand or predict *individual* behavior.

In summary, our dataset and analyses are unique in that they (i) include gold-standard individual psychological assessment data; (ii) use data from a form of online behavior (search) that engages all internet users — that is, 89% of the US adult population [46]; and (iii) employ a data-collection methodology (described below) that is scalable to hundreds or thousands of subjects while preserving users' privacy. Psycholinguistics analysis identified variables capable of differentiating people with low self-esteem. When compared with someone with moderate to high self-esteem, we observed noticeable differences in the type of searches made by individuals with low self-esteem condition. We demonstrated that these identifiers can be used to build statistical models that do better than chance for classifying users with low self-esteem condition using historical search data.

2 RELATED WORK

Researchers have used data from social media, public forums, blogs, and search engine logs to explore interaction of various aspects of mental health, including depression, anxiety, suicidal ideation, mood, and stress. De Choudhury provides an excellent overview of the role of social media in mental health research [11]. Her own previous work showed that lexicons in Tweets contain strong signals for predicting risk of major depressive disorder [14], and she recently identified linguistic markers to identify Twitter users likely to develop future suicidal ideation [15]. In 2018, Liu et al. examined statuses of Weibo users and demonstrated that users exposed to domestic violence showed increase in depressive symptoms, higher suicide risk, and decrease life satisfaction when compared against control group [30]. Other researchers have shown that Tweets contains meaningful clues for various mental health concerns including Adderall use [23], insomnia [26], and depression [10–14, 16].

Perhaps the dataset closest to ours which has been explored by researchers in the past is search logs from Google Trends [19]. Researchers have utilized Google Trends data to demonstrate deeper insight about population behavior and health related phenomena to build predictive models around mental and physical health assessment. For example, surveillance of influenza outbreaks [9, 18]; monitoring toxicological outbreaks [54]; identifying seasonality in seeking mental health [2]; monitoring suicide related terms to help predict rates of self-injury and death within the population [31]; and effects of geographic location on depression related searches [52] are some of the areas that researchers have explored. Nuti et al. provides a through review of the usage of Google Trends in the health care domain [33]. Sueki et al. investigated variations in the volume of Google searches about suicide and depression in Japan and found that the monthly search volumes for "suicide" and "suicide method" were not significantly correlated with the monthly suicide rate, but searches for "depression" were [49]. Yang et al. replicated this associations between the volume of Google search terms and the monthly suicide rate in Taipei City [53]. Gunn and Lester found that suicide rate for Americans was positively correlated with the search volume for search terms such as "commit suicide", "how to commit suicide", and "suicide prevention" [20]. They showed that monitoring internet searches for suicide related terms may provide a faster way to capture, in the moment, possible trends in mental health conditions such as depression, anxiety, and suicide. Song et al. showed that suicide and stress related searches are positively associated with suicide rates in Korea [47]. Hagihara et al. demonstrated that Google search volume for terms associated with hydrogen sulfide in Japan was correlated with the monthly suicide rate for people in their 20's and 30's [21]. Ayers et al. showed that there was a significant increase in suicide related terms in Google for the 19 days following the release of a popular Netflix show *13 Reasons Why*, which narrates the suicide of a fictional teen [3]. However, Page et al. found that internet search trends related to suicide terms are not necessarily straightforward indicators of the degree of suicide behavior in Australia [36]. It is debatable whether any query preceded an actual suicide attempt; but it is quite evident that search logs monitoring can be used to access understanding of different mental health conditions and capture any potential signals of low self-esteem, depression, anxiety, self harm risk, etc.

All of the work above was performed on search logs aggregated over a large population. By contrast, our work is the first to investigate search history data at an *individual level* for signs of low self-esteem.

3 DATA

Data for this study were collected through a 2-month on-campus recruitment to investigate how online services such as Google search can be used to detect early symptoms of suicidal ideation, depression, anxiety, low self-esteem, and lack of social supports. As described below, we developed a Human Subjects Review Board approved cloud-based application to acquire consent from subjects, download and anonymize their search histories, and link anonymized histories to corresponding mental health survey answers.

3.1 Participants

In total 108 individuals, 40 male and 68 female, volunteered to participate. Subjects were recruited through advertisements. To limit

any participation bias, the information on the advertisement was generic and had no mention of mental health related phenomena because some participants with certain personality traits can avoid recruitment.

A step in the data collection involves subjects completing an online written questionnaire about mental health, anxiety, suicidal ideation, and self-esteem. We call this the *Promote Health* survey. In addition to academic and demographic information, the survey incorporates questions from several standard instruments such as the depression module of the Patient Health Questionnaire (PHQ-9) [29] and the Rosenberg self-esteem scale [41, 43]. PHQ-9 is a common psychological instruments that has been used by previous researchers for depression related screening while mining social media data [27, 56]. Rosenberg self-esteem scale has been used to study usage of social media and its' association with low self-esteem among undergraduate students [39].

3.2 Recruitment Procedure

Interested participants were required to be at least 18 years old, able to read and write in English, and have an active Google services account. Qualified subjects were informed about the purpose, procedure, and goal of the study; then they reviewed and signed a statement of informed consent explaining the Human Subjects Committee approval. The study had 100% participation rate, meaning none of the 108 individuals decided to opt out of the study after the study was explained to them.

There are two parts to the data collection process. First, study recruiter asks participants to type their Google login credentials into a web-based app running on a tablet. Second, study participants were asked to take the *Promote Health* survey, a brief self assessment online questionnaire. There are 38 freshmen, 18 sophomore, 17 juniors, and 19 seniors in our dataset. In addition, there were 12 graduate students and 4 fifth year undergraduates. Participants' mean GPA was 3.51. Note that we did not make use of these additional data in the analyses presented in this paper, but that it may help explain the surprising level of low-self esteem described below.

3.3 Google Search History

Anyone with a Google account can freely access his or her entire personal data, at any time, through a platform called Google Takeout[2]. Using Google Oauth authentication protocol, participants authorize our search download application to access users Google related data. Although data about many forms of Google services can be accessed through this portal, this study uses only data from Google search. None of the other potential sources of data are accessed or stored at any time. Individuals who did not have Google search history tracking enabled were excluded from our analysis.

Once a user authorizes the application to obtain a copy of the search history, a one-time log in authorization is used to automatically transfer the search data from user's google drive to a secured server in the cloud. The oauth token that our application uses to access a user's data is valid only for 24 hours.

Before any data is transferred from "the cloud" and given to the research team to store as part of this study, the data is de-identified via Google's Cloud Data Loss Prevention (DLP) API. The API scans for personal identifiers such as names, addresses, phone numbers, etc., that can potentially identify individual study participants, and this information is automatically removed from the search history data prior to being saved as research data.

Once the search history data is successfully de-identified and transferred to the research team, the one-time log in authorization is permanently deleted. The research team is no longer able to use this authorization to access any participant's online data again. Fig. 1 presents the entire data collection process.

[2]https://takeout.google.com/settings/takeout

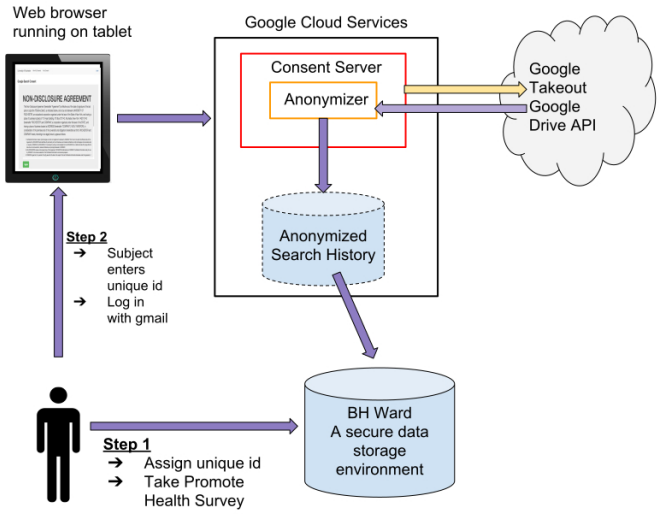


Figure 1: Data Collection Process.

Low Self-esteem (LS)	No Low Self-esteem (NLS)	
27	28	Analysis or Training set
21	19	Test set
5	8	No search data. Excluded from dataset
Total 108 subjects		

Table 1: Search dataset breakdown

3.4 Promote Health

Promote Health is a computer assisted survey instrument (CASI), designed at a fifth grade reading level and includes questions about age, academic concentration, gpa, race, gender, mental health (PHQ-9), Interpersonal violence (IPV), sexual health, self-esteem, etc. Users did not write their names on the survey at anytime, but instead entered a random ID number that was used to link their responses to their search history data. No records connecting subject names to ID numbers was maintained.

3.5 Data Description

A section in the survey asks series of questions for evaluating individual self-esteem. The answer to the self-esteem questions were used to compute the Rosenberg Self-Esteem Scale(RSES). 53 individuals had RSES score less than 12 (on a scale of 0-30, people with average to high self-esteem have score ≥ 15), indicating low self-esteem(LS). On the contrary, 55 individuals had score greater than or equal to 12 indicating no sign of low self-esteem(NLS). However, there were 13 individuals who participated but did not have any search data and were excluded from any analysis.

Before any analysis and modeling, we separate the dataset into two parts, namely **analysis set** and **test set**. This is illustrated in Table 1. For the self-esteem analysis, 27 participants had LS and 28 had NLS condition. The test set had 19 subjects with NLS and 21 with LS condition.

Through the *Promote Health*, we measured the current RSES score. Low self-esteem is one of the vulnerable factors that works in a vicious cycle with depression, anxiety, and suicide ideation [5]. However, low self-esteem state and any specific mental health condition may not necessarily occur at the same time but can have indirect effect on the quality of life.

In our survey responses, we observed that low self-esteem condition stood out strongly than depression in the responses. The average gpa of the participants is 3.51, which is quite high. A potential explanation can be "imposter syndrome" – where high performing students do not feel they are as smart as their colleagues. Young college undergraduates are more prone to self-criticism, social anxiety, achievement pressures, and depressive tendencies [28].

4 OBSERVATION ON GOOGLE SEARCH LOGS

Google search phrases tend to be short, recorded in the present and are good indicators of what may be going through an individual's mind at the moment. The context of a search phrase can be anything. We are interested in identifying the categories of the searches and uncover any linguistic signals in the search phrases that can help distinguish subjects with LS and NLS conditions. In the following sections, all the search logs analysis is carried out on the **analysis set**, see Fig 1.

4.1 Variance of Search Categories

Using the content classification feature of the Google Cloud NLP API[3] we were able to classify the search queries of every participating subjects. The API returns a hierarchical category label for every search query. We treat the broad category as the 'category' label for the query. For instance, for a query q, if the label from the API is '/Arts & Entertainment/Humor/Funny Pictures & Videos', we consider 'Arts & Entertainment' as the category for q.

In total, there are 27 such categories. We observed that, at any time of the day, individuals with LS condition consistently searched more on topics related to 'Sports', 'Health', 'Finance', 'News', 'Books & Literature', 'Reference', and 'Law & Government'. This is illustrated in Fig. 2. The 'Reference' category constitutes mostly of topics associated to humanities, language resources, geographic references, and general references such as concepts from Encyclopedias, Wikipedias, etc. One explanation for such high prevalence

[3]https://cloud.google.com/natural-language/docs/classifying-textt

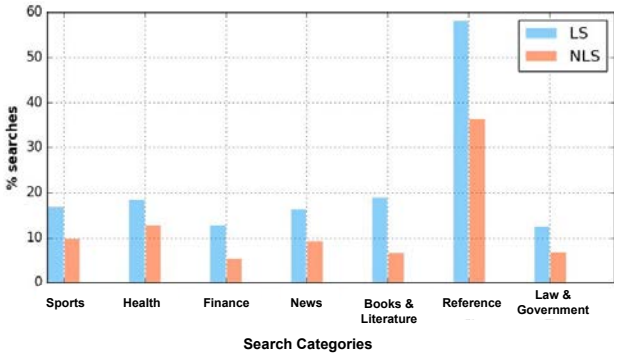


Figure 2: Search categories that were more frequent in LS population

of searches from this category can be that students with low self-esteem condition are more reliant on search engines to do their daily school works compared to the population that does not exhibit low self-esteem.

We also observed that individuals with current LS, on average, search more about certain topics than students with NLS condition during late hours (1AM - 5AM). For instance, students with LS condition made more searches pertaining to categories such as 'Reference', 'Jobs & Education', 'Books & Literature', and 'Health' than students with NLS condition. This increased late night activity disrupts normal sleep cycle which is reported to be one of the core symptoms of depression and can eventually lead to poor quality of mental health and life in general [34].

Context(i.e.category of a search) and frequency of searches on Google vary from person to person. Over the entire search history, we noticed observable differences in distribution of search categories between the LS and NLS population. For example, searches pertaining to the '/Arts & Entertainment' category were more frequent for both the groups. However, there are categories such as 'Health', 'Food & Drink', 'Sensitive Subjects', 'References', etc. for which the distribution demonstrate apparent differences.

4.2 Psycho linguistic attribute detection in search logs

At first, it may not be self-evident that query logs can be good indicators of what may be going through an individual's mind at the moment. Traditionally, researchers have done psycho linguistic analysis on social media data to uncover signals of mental health related phenomena. In 2017, Oexle et al. have shown that there is a strong stigma associated to expressing oneself related to mental illness and other related issues [35]. As a defensive mechanism, users may stop generating content on social media platforms [25]. However, when someone searches, there is no fear of any judgment; and what he/she is searching can potentially reveal his/her psychological state at that moment.

LIWC is a text analysis toolkit that outputs the proportion of words in a given text that fall into one or more of over 80 linguistic (such as prepositions, adverbs, first-person singular pronouns, conjunctions), psychological (happy, anger, achievement, etc.), and topical (leisure, money, etc.) categories [37]. Researchers have been

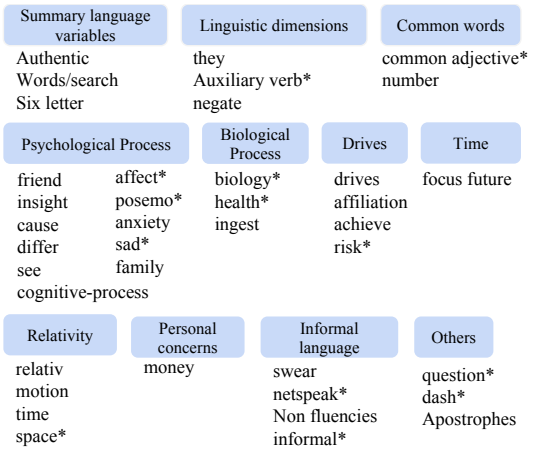


Figure 3: LIWC attributes that were statistically significantly different between the subjects with LS and NLS condition. Concepts with * were more prevalent among subjects with LS condition.

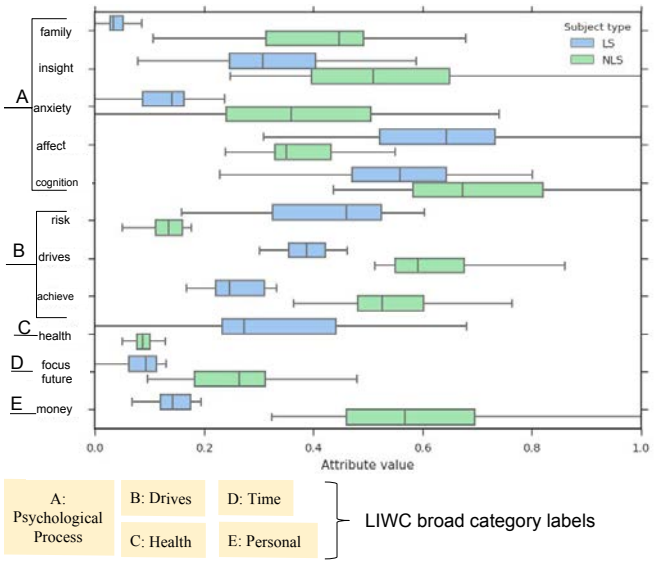


Figure 4: Distribution of some LIWC attributes score between participants with LS and NLS condition. These are some of the attributes from Fig. 3

using this tool to study the distribution of analytical thinking, authenticity, emotional tone, social and cognitive process present in a piece of text. Clinicians and social computer scientists do psychometric analysis using LIWC lexicons. Some popular text data analyzed using this lexicon includes social media, online forums, written notes, and self-introductions [42]. We are the first to demonstrate the presence of these linguistic attributes in individual level search data.

For the low self-esteem analysis, our data consists of 27 subjects with LS and 28 subjects with NLS condition from the **analysis set**(see Fig 1). For every subject with LS condition, all the Google search phrases are analyzed using LIWC and the output is a 92 dimension vector. The result can be treated as a table with 27 rows and 92 columns. A similar procedure is repeated for every participants with NLS condition, resulting in a table with 28 rows and 92 columns.

Next, we explore whether there are any statistically significant LIWC categories between the two populations (LS vs NLS). Our sample size is small, assuming normal distribution, we consider the LIWC variables from the LS and NLS groups and do student's t-tests. We found 40 LIWC attributes that were statistically significantly different(with p-value < 0.05) between participants with LS and NLS, as shown in Fig. 3.

In the low self-esteem analysis, the family variable was significantly lower for almost every participant with LS condition than participants with NLS condition. This supports previously established findings where researchers have shown that family support influences self-esteem, which impacts adolescent suicide risk behaviors [32, 44]. Fig. 4 presents a visual representation of the distribution of some of the LIWC attributes (from Fig 3) that are statistically significantly different between LS and NLS.

Search phrases are short and lack syntactic structure compared to other forms of data that are popularly analyzed using LIWC. Yet, we found strong signals in LIWC categories that can differentiate and identify individuals with low self-esteem. Our analysis of search data strengthens prior findings regarding the association of self esteem with other mental health phenomena [32] .

This indicates that there are some shared lexicons, even in Google search phrases, among individuals with LS condition. For instance Affect, Insight, Cogproc, Space, and Affiliation are some of the linguistic variables that are significantly to differentiate a population between LS and NLS.

5 MODELING LOW SELF-ESTEEM (LS) CONDITION

For the classification task, we ask the question: *Based on the search history, can we identify subjects with low self-esteem?*

We use the **analysis set** from Fig. 1 for selecting features and training our classifiers. Our training set consists of 27 individuals with LS and 28 with NLS condition.

The performance of the trained model is evaluated using a test set. We separated this set from the data prior to the feature selection and the training process. The held out test set consists of 19 subjects with NLS and 21 with LS.

5.1 Feature Selection from Analysis Set

Based on our observation from section 4.1 & 4.2, we consider two types of features for the classification task.

5.1.1 Features based on LIWC:. There were 40 linguistic attributes that were statistically significantly different between the LS and NLS population (see Fig. 3). We treat these LIWC attributes as features for training our models.

5.1.2 Features based on search category distribution: Searching behavior varies from person to person. Knowing the categories of

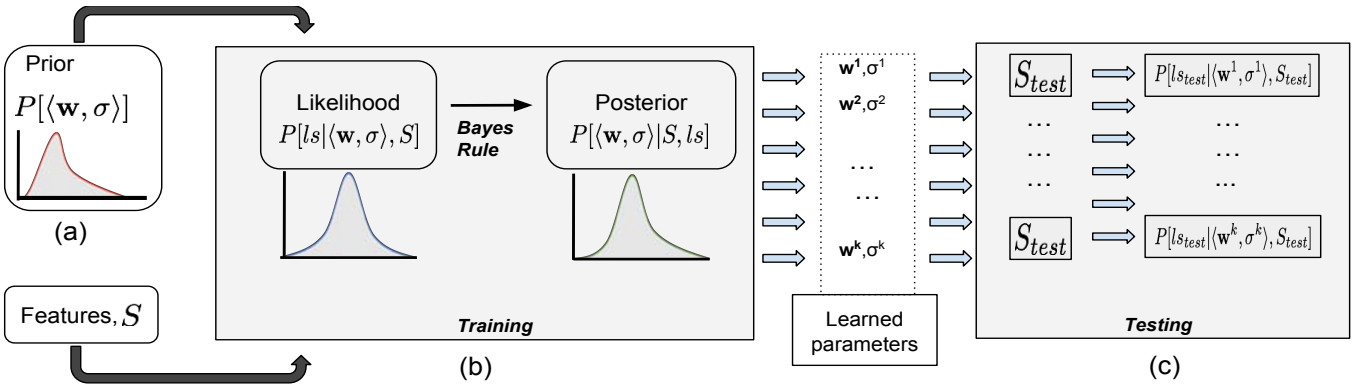


Figure 5: Process depicting learning and testing procedure of the HyBaR model. (a) Prior distribution of the parameters w and σ. Prior of σ is uniform distribution. In case of LIWC based features the prior of w is Laplace distribution defined in (4) and is Gaussian distribution with category similarity matrix as the covariance matrix for the search category based features. (b) Posterior is obtained from the corresponding prior and the likelihood using Bayes Rule. Using Metropolis-Hastings Algorithm several model parameters are sampled from the approximate posterior. (c) S_{test} represents the test data which has been held out prior to any analysis and training. Each of the learned parameters are used individually to classify S_{test}. This helps in model generalization.

searches can reveal insightful patterns in internet activities. Different individuals exhibit different searching behavior. For example, following Fig 3, we found that students with LS condition consistently searched more about certain topics than students with NLS condition. This motivated us to treat the search category distribution, over one's entire search history, as features for our models. Using the Google Cloud NLP API, we labelled search queries into 27 categories. We treat the distribution of these 27 search categories as features and train our classifiers.

5.2 Models

We propose a discriminative model whose parameters are learned using Bayesian approach. We call our model Hybrid Bayesian Regression (HyBaR) because it is a cross-breed between the Bayesian Linear Regression and the Bayesian Logistic Regression [7]. For details about the model, see section 5.2.1.

In addition, we trained a **Logistic Regression(LR)**, a **Support Vector Machine(SVM)** and compare their performance with HyBaR.

5.2.1 ***Hybrid Bayesian Regression (HyBaR):.*** HyBaR has advantage over Maximum Likelihood Estimation (MLE) due to the additional prior information. Compared to the maximum a posteriori (MAP) estimation, which outputs a parameter that maximizes the posterior probability for the given data, HyBaR estimates the entire posterior distribution which can avoid over-fitting and lead to a more generalized model.

Unlike LR and SVM, HyBaR incorporates domain knowledge in the form of prior. For the LIWC based features, we have used a sparse prior, and for the category distribution based features we employed the semantic distance between the categories as prior.

The Model: For $i \in [1, \ldots, n]$ we denote $\boldsymbol{s}^i = (s^i_1, \ldots, s^i_m)$ to be the vector that represents the features extracted from the search history of the person p_i, where n is the number of people and m is the number of search categories. For instance, when we consider the search category based features, s^i_j represents the percentage of searches of category j for i^{th} person. Also, let $ls_i \in \{0, 1\}$ be a binary variable denoting whether a person p_i exhibits low self-esteem or not. In particular, $ls_i = 1$ when the person p_i exhibits low self-esteem and 0 otherwise. Now we propose HyBaR, a probabilistic discriminative model with parameters $\boldsymbol{w}(\in \mathbb{R}^m)$ and $\sigma(\in \mathbb{R})$. For each $i \in [1, \ldots, n]$ we assume

$$ls_i \sim \text{Bern}\left(f(\mathbf{w}^T \mathbf{s}^i + \epsilon)\right) \tag{1}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is a latent variable and f is a function whose range is $[0, 1]$ (i.e.which outputs a valid probability). One choice for f can be **Sigmoid**. Also, Bern(p) denotes the bernouli distribution, i.e., $X \sim \text{Bern}(p)$ iff,

$$X = \begin{cases} 1 & \textit{with probability } p \\ 0 & \textit{with probability } (1-p) \end{cases}$$

Next, we describe the prior distribution of the parameters.

Remark: Let $\mathbf{x}$ be the data and y be its label. In case of linear regression, class label, $y(\in \mathbb{R})$ is modeled as,

$$y \sim \mathbf{w}^T \mathbf{x} + \epsilon \tag{2}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and $\mathbf{w}$ is the model parameter. However, for logistic regression, class label $y \in \{0, 1\}$ and is modeled as,

$$y \sim \text{Bern}\left(Sigmoid(\mathbf{w}^T \mathbf{x})\right) \tag{3}$$

Hence from (2) and (3) it follows that our model (1) is a simple hybrid of the two models.

Prior Selection: Domain knowledge is incorporated in the model through our priors. Hence, an important component during the model construction is prior selection. The prior of σ is chosen to be uniform in both types of features. However, we set the prior of $\boldsymbol{w}$ depending on the features we are selecting.

For the LIWC based features, the prior is set to be the Laplace distribution, i.e.,

$$P(w_i|\mu, b) = \frac{1}{2b} \exp\left(-\frac{|w_i - \mu|}{b}\right) \tag{4}$$

where μ and b are hyper-parameters. It is to note that Laplace prior is equivalent to sparsity constraint. In the context of LIWC based features, for any individual, these linguistic categories do not co-occur in the search history frequently. This motivated us to select a sparse prior.

For the search category distribution based features, we first construct a similarity matrix that represents the semantic distance between the search categories using GloVe [38]. We set the prior of $\boldsymbol{w}$ to be multivariate Gaussian with mean 0 and the category-similarity matrix as the covariance matrix, i.e.,

$$\boldsymbol{w} \sim \mathcal{N}(\vec{0}, \boldsymbol{Cov}) \tag{5}$$

where $\boldsymbol{Cov}_{ij}$ is the semantic similarity between category i and category j. We set this prior to enforce the similarities between the search categories.

Posterior Inference: Let $S = \left\{\boldsymbol{s}^i | i \in \{1, \ldots, n\}\right\}$ be the set of features for all the participants and $L = \left\{ls^i | i \in \{1, \ldots, n\}\right\}$ be the corresponding self esteem label for them. Now, we formalize the posterior for our model as follows:

$$\begin{aligned} P[\boldsymbol{w}, \sigma^2, \epsilon | L, S] &\propto P[L|\boldsymbol{w}, \sigma^2, \epsilon, S] \cdot P[\boldsymbol{w}, \sigma^2, \epsilon | S] \\ &= \underbrace{P[L|\boldsymbol{w}, \sigma^2, \epsilon, S] \cdot P[\epsilon|\sigma^2]}_{\text{Likelihood}} \cdot \underbrace{P[\boldsymbol{w}] \cdot P[\sigma^2]}_{\text{Prior}} \qquad (6) \\ &= \left(\prod_{i=1}^{n} P[ls_i|\boldsymbol{w}, \sigma^2, \epsilon, \boldsymbol{s}^i]\right) \cdot \left(P[\epsilon|\sigma^2] \cdot P[\boldsymbol{w}] \cdot P[\sigma^2]\right) \qquad (7) \end{aligned}$$

Here (6) follows from conditional independence of S & $\boldsymbol{w}$ and (7) follows from the i.i.d. properties of the data. From (1) we can see that,

$$P[ls_i|\boldsymbol{w}, \sigma^2, \boldsymbol{s}^i] = prob_i^{ls_i} \cdot (1 - prob_i)^{(1-ls_i)}, \tag{8}$$

where, $prob_i = Sigmoid(\mathbf{w}^T \mathbf{s}^i + \epsilon)$. Now, for LIWC based features $P[\boldsymbol{w}]$ in (7) is given by the distribution as in (4) and for the search category based features $P[\boldsymbol{w}]$ in (7) is the distribution as in (5).

In summary, we get the posterior of the parameters from the corresponding prior distribution and the training data likelihood by using Bayes Rule. Then our goal is to estimate the entire posterior distribution and sample model parameters from the posterior distribution. The complete process is depicted in Fig. 5 (b). However, it important to note that exact inference of the posterior given in (7) is intractable. Hence, we use Metropolis-Hastings Algorithm to get approximate sample from the posterior distribution.

Parameter Estimation: Our next step is to choose a parameter for the classification task. We sample several parameters from the posterior distribution as shown in Fig. 5. Let $\{\langle \boldsymbol{w}^1, \sigma^1 \rangle, \cdots, \langle \boldsymbol{w}^k, \sigma^k \rangle\}$ be the sampled parameters. Using each of these learned parameters, we classify the test data as follows,

$$ls^i_{test} \sim \text{Bern}\left(Sigmoid(\mathbf{w}^{\mathbf{i}^T} \mathbf{s}_{test} + \epsilon)\right),$$

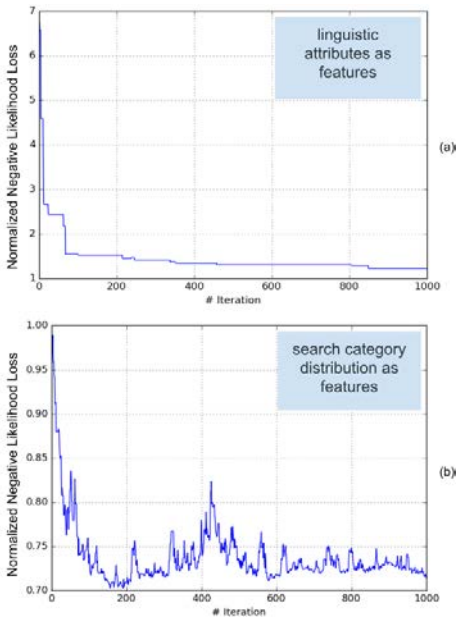


Figure 6: An instance of the convergence of negative log likelihood during training for HyBaR when trained using the two types of features separately.

where $\epsilon \sim \mathcal{N}(0, \sigma^{i^2})$. Hence we get k labels ($\{ls^1_{test}, \cdots, ls^k_{test}\}$) for each of the data point $\mathbf{s}_{test}$. We record the performance of the model for each $\langle \mathbf{w}^i, \sigma^i \rangle$ and report the average performance of the model on test set in Fig 8 (b) & (d).

Generalization Ability: Following the PAC-Bayesian framework in [17] we can set the normalized negative log likelihood as the loss function for the posterior inference problem. Fig 6 presents an instance of the convergence of negative log likelihood loss during training. Next, we ask the question, how well this model generalizes on a completely new data set? We test the generalization ability of HyBaR by running simulation on large synthetic data. Fig 7 shows the convergence of the expected empirical loss towards the expected true loss. A potential explanation for such behavior can be due to the Bayesian approach used in designing HyBaR.

Driving Search Categories: After training HyBaR on the 27 search categories (from 5.1.2), the top ten categories (ranked according to learned weights) that drive the classification tasks are: Internet & Telecom (2.17), Law & Government (1.91), People & Society (1.85), News (1.73), Real Estate (1.69), Jobs & Education (1.55), Online Communities (1.45), Finance (1.16),Science (1.15), and Health (1.00). High prevalence of these categories in a college student's search history may help identify conditions such as a low self-esteem. For example, searchers pertaining to People & Society categories: how to 'fit in' with communities in college settings, looking for ways to make friends etc. are practical situations that teenagers encounter when they move to college.

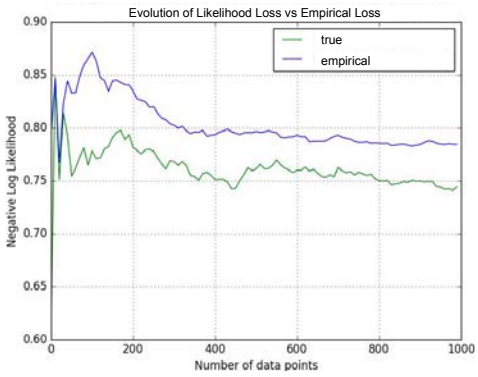


Figure 7: Convergence of the expected empirical loss towards expected true loss on the synthetic data.

6 RESULTS

The performance of our trained models is evaluated using the held out test set which consists of 19 individuals with NLS and 21 with LS condition. Fig. 8 presents the performance of our models. To evaluate the accuracy of the classification, we use F1 score to capture the balance between precision and recall and compare our models.

HyBaR samples multiple parameters from the posterior. We use each of these learned parameters individually to classify our test data and report the average/best performance on the test data (in terms of F1 score).

When LIWC's linguistic attributes are considered as features, on average, HyBaR performed better than both the SVM and the baseline model i.e., LR in classifying participants with low self-esteem. This is evident from Fig. 8 (a) & (b), where HyBaR has an average F1 score of 0.72 compared to SVM's 0.61 and LR's 0.46. We also report the best performance of HyBaR in Fig. 8(b) (numbers with * as superscript) and found that the best model has F1 score of 0.77.

When search category distributions were used as features, HyBaR performed significantly better than SVM and LR, compare Fig. 8 (c) & (d). On average, HyBaR has F1 score of 0.86 compared to SVM's 0.62 and LR's 0.47. Using the category distribution, HyBaR had the best performance with F1 score of 0.89.

Since our test set is small, we compared the performance of the models on the test set by analyzing the confusion matrix. Through Fig. 9, we compare the performance of our models (on the test set) when they are trained using the LIWC based features. The green and the yellow cell in the table represent number of misclassification and HyBaR made the fewest misclassifications compared to LR, our baseline, and SVM. We repeated our analysis on various test set with different ratios of the two categories and obtained similar results.

On average, HyBaR performed significantly better than LR and SVM when we trained it using features based on the search category distribution. The performance of HyBaR on the test set can be evaluated through the confusion matrix presented in Fig. 10. Out of 21 participants with LS condition, HyBaR only misclassified 2 subject (see yellow cell in Fig. 10) and misclassified 4 out of the 19 individuals with NLS condition.

7 DISCUSSION AND FUTURE WORK

Individual level Google search history data can be treated as a rich longitudinal data source that can capture quotidian online activities of individuals. This data has recently become readily accessible through the Google Takeout interface. We built a scalable human-subject review board-approved system for consenting, collecting, and anonymizing subjects' search history data, and created the first data set that links search history data to that from a clinically-used mental health survey instrument.

We showed that search history data provides identifiable signals for detecting low self-esteem. In addition to considering search category distribution, we were able to identify linguistic variables whose frequency of use was significantly different among subjects suffering from low self-esteem and those not exhibiting low self-esteem condition. Using these features, we have managed to build mental-health related predictors, for instance consider our low self-esteem model, that is superior to predicting by random chance. This motivates us to consider that search history data can be used to build models for capturing various mental health phenomena such as self-harm, suicide ideation, etc.

It will be interesting to explore whether both psycho linguistic attributes and search categories can be used simultaneously to build models. Due to small sample size, we could not use both sets of features at the same time. However, this limitation can be easily overcome as we collect more data through our future studies.

The mental health survey we employed red-flagged low self-esteem as the biggest mental health issue for the subject population of college students. Our next study will recruit a more general population and compare the findings. Recruitment sites will include mental-health inpatient and outpatient clinics, emergency rooms, and family court. Individuals from these sites are more likely to have relatively high incidence and prevalence of conditions such as major depression, anxiety, and stress, along with low self-esteem.

We will also consider features beyond linguistic and search categories. Other factors one can consider are time at which these searches are made, seasonality of certain types of searches. The time of a search is likely to be of particular value because searching in the middle of the night can be used as a proxy for sleeplessness, a symptom of many mental health problems. We will test the hypothesis that an increase in a user's volume of searches is a proxy for an increased use of technology due to withdrawal from social interactions.

The fact that search logs are time and date tagged means that they can be used for longitudinal studies on disease progression, as well as part of longitudinal treatment plans for individuals. We are currently beginning a study where subjects who are undergoing outpatient treatment for attempted suicide may consent to allow their history of PHQ-9 survey answers to be analyzed in conjunction with their search histories. The goal of the study will be to model the dynamics of health and online behavior in order to provide new tools for therapists to assess the efficacy of treatment strategies.

Finally, the accessibility of individual level search log data opens the door to studying signals for other mental health phenomena, such as anxiety and panic disorders, bipolar disorder, eating disorders, suicide ideation, and substance abuse and addiction. A feasible use case of our system can be in the university counselling services

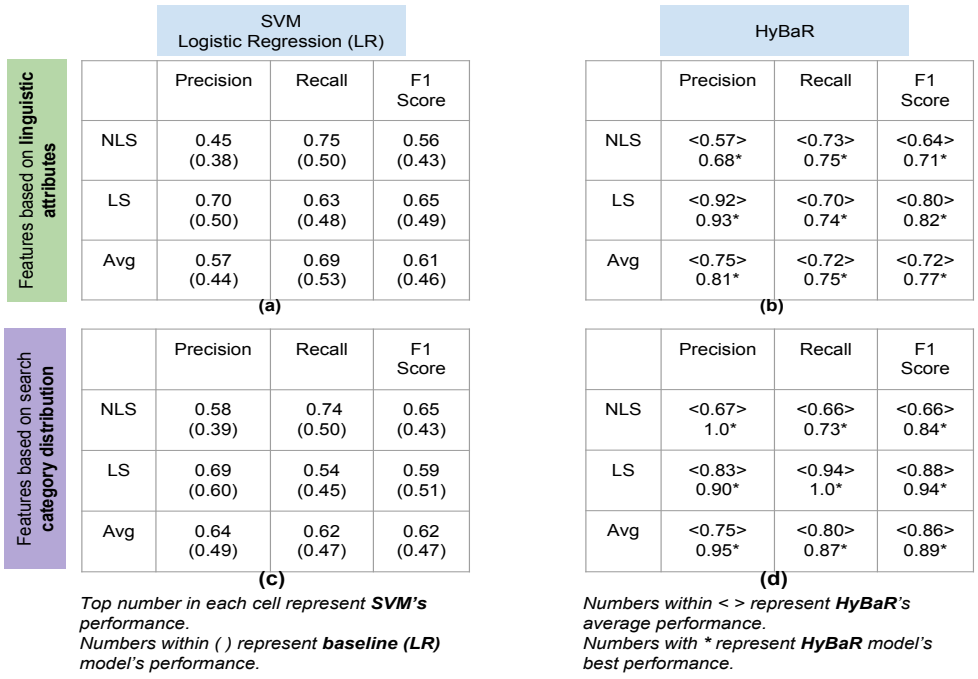


Figure 8: Classifiers performance on the Test Set. LR, SVM, and HyBaR are trained for the classification task. (a) & (b): Tables present the performance on the test set when LIWC based features are used. (c) & (d): Tables present the performance on the test set when search category based features are used. In case of HyBaR, the model learns multiple parameters (see From Fig 5), the test set is evaluated using each of the parameters. Numbers in ⟨⟩ represents the average performance over the learned parameters. Numbers with * represent the score of the best performing HyBaR parameter on the test set.

N = 40 (NLS: 19, LS: 21)	Predicted NLS	Predicted LS
True NLS	LR: 5 SVM: 14 HyBaR: 15	LR: 14 SVM: 5 **HyBaR: 4**
True LS	LR: 12 SVM: 8 **HyBaR: 5**	LR: 9 SVM: 13 HyBaR: 16

Figure 9: Confusion matrix to compare the performance of Logistic Regression(LR), Support Vector Machine (SVM), and HyBaR (average performance of HyBaR) on the test set when trained using linguistic features.

N = 40 (NLS: 19, LS: 21)	Predicted NLS	Predicted LS
True NLS	LR: 7 SVM: 14 HyBaR: 15	LR: 12 SVM: 5 **HyBaR: 4**
True LS	LR: 6 SVM: 10 **HyBaR: 2**	LR: 15 SVM: 11 HyBaR: 19

Figure 10: Confusion matrix to evaluate the performance of LR, SVM, and HyBaR on test set when trained on features based on search category distribution.

to access the morale of the student population which can eventually lead to better understanding of student life.

We are working with other researchers and clinicians at our university's medical center to include this work as a core component of a suicide prevention center that is currently being developed. For example, this is now a central feature of the proposed ALACRITY Research Center focused on suicide prevention among patients with Serious Mental Illness at our institution's medical center[4]. It will form a core data component for use in all NIH-funded sub-projects within the Center. This offers to expand the overall social impact by influencing clinical care in the future for those individuals who are among the absolute highest risk for suicide mortality.

ACKNOWLEDGMENTS

We thank Adam Sadilek and other collaborators at Google and Verily for discussions, advice, and the use of anonymization software. This work was supported by the Goergen Institute for Data Science and Google.

REFERENCES

[1] Nazanin Andalibi, Pinar Ozturk, and Andrea Forte. 2015. Depression-related imagery on instagram. In *Proceedings of the 18th ACM Conference Companion on Computer Supported Cooperative work & social computing*. ACM, 231–234.

[2] John W Ayers, Benjamin M Althouse, Jon-Patrick Allem, J Niels Rosenquist, and Daniel E Ford. 2013. Seasonality in seeking mental health information on Google. *American journal of preventive medicine* 44, 5 (2013), 520–525.

[4] https://grants.nih.gov/grants/guide/pa-files/PAR-16-354.html

[3] John W Ayers, Benjamin M Althouse, Eric C Leas, Mark Dredze, and Jon-Patrick Allem. 2017. Internet searches for suicide following the release of 13 Reasons Why. *JAMA internal medicine* 177, 10 (2017), 1527–1529.

[4] Roy F Baumeister. 1990. Suicide as escape from self. *Psychological review* 97, 1 (1990), 90.

[5] Sunil Bhar, Marjan Ghahramanlou-Holloway, Gregory Brown, and Aaron T Beck. 2008. Self-esteem and suicide ideation in psychiatric outpatients. *Suicide and Life-Threatening Behavior* 38, 5 (2008), 511–516.

[6] Jiang Bian, Umit Topaloglu, and Fan Yu. 2012. Towards large-scale twitter mining for drug-related adverse events. In *Proceedings of the 2012 international workshop on Smart health and wellbeing*. ACM, 25–32.

[7] Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

[8] Ronald L Bonner and Alexander Rich. 1988. Negative life stress, social problem-solving self-appraisal, and hopelessness: Implications for suicide research. *Cognitive Therapy and Research* 12, 6 (1988), 549–556.

[9] Herman Anthony Carneiro and Eleftherios Mylonakis. 2009. Google trends: a web-based tool for real-time surveillance of disease outbreaks. *Clinical infectious diseases* 49, 10 (2009), 1557–1564.

[10] Munmun De Choudhury, Scott Counts, and Eric Horvitz. 2013. Predicting postpartum changes in emotion and behavior via social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 3267–3276.

[11] Munmun De Choudhury, Scott Counts, and Eric Horvitz. 2013. Social media as a measurement tool of depression in populations. In *Proceedings of the 5th Annual ACM Web Science Conference*. ACM, 47–56.

[12] Munmun De Choudhury, Scott Counts, Eric J Horvitz, and Aaron Hoff. 2014. Characterizing and predicting postpartum depression from shared facebook data. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. ACM, 626–638.

[13] Munmun De Choudhury and Sushovan De. 2014. Mental Health Discourse on reddit: Self-Disclosure, Social Support, and Anonymity.. In *ICWSM*.

[14] Munmun De Choudhury, Michael Gamon, Scott Counts, and Eric Horvitz. 2013. Predicting Depression via Social Media.. In *ICWSM*. 2.

[15] Munmun De Choudhury, Emre Kiciman, Mark Dredze, Glen Coppersmith, and Mrinal Kumar. 2016. Discovering shifts to suicidal ideation from mental health content in social media. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. ACM, 2098–2110.

[16] Johannes C Eichstaedt, Robert J Smith, Raina M Merchant, Lyle H Ungar, Patrick Crutchley, Daniel Preoţiuc-Pietro, David A Asch, and H Andrew Schwartz. 2018. Facebook language predicts depression in medical records. *Proceedings of the National Academy of Sciences* (2018), 201802331.

[17] Pascal Germain, Francis Bach, Alexandre Lacoste, and Simon Lacoste-Julien. 2016. Pac-bayesian theory meets bayesian inference. In *Advances in Neural Information Processing Systems*. 1884–1892.

[18] Jeremy Ginsberg, Matthew H Mohebbi, Rajan S Patel, Lynnette Brammer, Mark S Smolinski, and Larry Brilliant. 2009. Detecting influenza epidemics using search engine query data. *Nature* 457, 7232 (2009), 1012.

[19] Google. 2017. Google Trends. https://trends.google.com/trends/

[20] John F Gunn III and David Lester. 2013. Using google searches on the internet to monitor suicidal behavior. *Journal of affective disorders* 148, 2 (2013), 411–412.

[21] Akihito Hagihara, Shogo Miyazaki, and Takeru Abe. 2012. Internet suicide searches and the incidence of suicide in young people in Japan. *European archives of psychiatry and clinical neuroscience* 262, 1 (2012), 39–46.

[22] Jeffrey A. Hall and Natalie Pennington. 2013. Self-monitoring, honesty, and cue use on Facebook: The relationship with user extraversion and conscientiousness. *Computers in Human Behavior* 29, 4 (2013), 1556 – 1564.

[23] Carl L Hanson, Scott H Burton, Christophe Giraud-Carrier, Josh H West, Michael D Barnes, and Bret Hansen. 2013. Tweaking and tweeting: exploring Twitter for nonmedical use of a psychostimulant drug (Adderall) among college students. *Journal of medical Internet research* 15, 4 (2013).

[24] Catherine Hoffman and Julia Paradise. 2008. Health insurance and access to health care in the United States. *Annals of the New York Academy of Sciences* 1136, 1 (2008), 149–160.

[25] Becky Inkster, David Stillwell, Michal Kosinski, and Peter Jones. 2016. A decade into Facebook: where is psychiatry in the digital age? *The Lancet Psychiatry* 3, 11 (2016), 1087–1090.

[26] Sue Jamison-Powell, Conor Linehan, Laura Daley, Andrew Garbett, and Shaun Lawson. 2012. I can't get no sleep: discussing# insomnia on twitter. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1501–1510.

[27] Lauren A Jelenchick, Jens C Eickhoff, and Megan A Moreno. 2013. âĂIJFacebook depression?âĂİ Social networking site use and depression in older adolescents. *Journal of Adolescent Health* 52, 1 (2013), 128–130.

[28] John Kolligian Jr and Robert J Sternberg. 1991. Perceived Fraudulence in Young Adults: Is There an'Imposter Syndrome'? *Journal of personality assessment* 56, 2 (1991), 308–326.

[29] Kurt Kroenke, Robert L Spitzer, and Janet BW Williams. 2001. The phq-9. *Journal of general internal medicine* 16, 9 (2001), 606–613.

[30] Mingming Liu, Jia Xue, Nan Zhao, Xuefei Wang, Dongdong Jiao, and Tingshao Zhu. 2018. Using social media to explore the consequences of domestic violence on mental health. *Journal of interpersonal violence* (2018), 0886260518757756.

[31] Michael J McCarthy. 2010. Internet monitoring of suicide risk in the population. *Journal of affective disorders* 122, 3 (2010), 277–279.

[32] Blace A Nalavany and Lena W Carawan. 2012. Perceived family support and self-esteem: The mediational role of emotional experience in Adults with Dyslexia. *Dyslexia* 18, 1 (2012), 58–74.

[33] Sudhakar V Nuti, Brian Wayda, Isuru Ranasinghe, Sisi Wang, Rachel P Dreyer, Serene I Chen, and Karthik Murugiah. 2014. The use of google trends in health care research: a systematic review. *PloS one* 9, 10 (2014), e109583.

[34] David Nutt, Sue Wilson, and Louise Paterson. 2008. Sleep disorders as core symptoms of depression. *Dialogues in clinical neuroscience* 10, 3 (2008), 329.

[35] Bridianne O'dea, Mark E Larsen, Philip J Batterham, Alison L Calear, and Helen Christensen. 2017. A linguistic analysis of suicide-related Twitter posts. *Crisis* (2017).

[36] Andrew Page, Shu-Sen Chang, and David Gunnell. 2011. Surveillance of Australian suicidal behaviour using the Internet?

[37] JW Pennebaker, RJ Booth, RL Boyd, and ME Francis. 2015. Linguistic inquiry and word count: LIWC2015. Austin, TX: Pennebaker Conglomerates.

[38] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[39] Carly R Pineiro. 2016. *Social media use and self-esteem in undergraduate students*. Rowan University.

[40] Wullianallur Raghupathi and Viju Raghupathi. 2014. Big data analytics in healthcare: promise and potential. *Health Information Science and Systems* 2, 1 (2014), 3.

[41] Richard W Robins, Holly M Hendin, and Kali H Trzesniewski. 2001. Measuring global self-esteem: Construct validation of a single-item measure and the Rosenberg Self-Esteem Scale. *Personality and social psychology bulletin* 27, 2 (2001), 151–161.

[42] Rebecca L Robinson, Reanelle Navea, and William Ickes. 2013. Predicting final course performance from studentsâĂŹ written self-introductions: A LIWC analysis. *Journal of Language and Social Psychology* 32, 4 (2013), 469–479.

[43] Morris Rosenberg. 1965. Rosenberg self-esteem scale (RSE). *Acceptance and commitment therapy. Measures package* 61 (1965), 52.

[44] Amira Y Sharaf, Elaine A Thompson, and Elaine Walsh. 2009. Protective effects of self-esteem and family support on suicide risk behaviors among at-risk adolescents. *Journal of Child and Adolescent Psychiatric Nursing* 22, 3 (2009), 160–168.

[45] Morton M Silverman and Ronald W Maris. 1995. The prevention of suicidal behaviors: An overview. *Suicide and Life-Threatening Behavior* 25, 1 (1995), 10–21.

[46] Aaron Smith and Monica Anderson. 2018. Social Media Use in 2018. http://www.pewinternet.org/2018/03/01/social-media-use-in-2018/. Accessed 7 June 2018.

[47] Tae Min Song, Juyoung Song, Ji-Young An, Laura L Hayman, and Jong-Min Woo. 2014. Psychological and social factors affecting Internet searches on suicide in Korea: a big data analysis of Google search trends. *Yonsei medical journal* 55, 1 (2014), 254–263.

[48] Julia Friederike Sowislo and Ulrich Orth. 2013. Does low self-esteem predict depression and anxiety? A meta-analysis of longitudinal studies. *Psychological Bulletin* 139, 1 (Jan. 2013), 213–240.

[49] Hajime Sueki. 2011. Does the volume of Internet searches using suicide-related search terms influence the suicide death rate: Data from 2004 to 2009 in Japan. *Psychiatry and clinical neurosciences* 65, 4 (2011), 392–394.

[50] Michael Van Ameringen, Catherine Mancini, William Simpson, and Beth Patterson. 2010. Potential use of Internet-based screening for anxiety disorders: a pilot study. *Depression and anxiety* 27, 11 (2010), 1006–1010.

[51] Victor R. Wilburn and Delores E. Smith. 2005. STRESS, SELF-ESTEEM, AND SUICIDAL IDEATION IN LATE ADOLESCENTS. *Adolescence* 40, 157 (Spring 2005), 33–45. https://search.proquest.com/docview/195943093?accountid=13567 Copyright - Copyright Libra Publishers Incorporated Spring 2005; Document feature - Tables; ; Last updated - 2017-10-31; CODEN - ADOLAO.

[52] Albert C Yang, Norden E Huang, Chung-Kang Peng, and Shih-Jen Tsai. 2010. Do seasons have an influence on the incidence of depression? The use of an internet search engine query data as a proxy of human affect. *PloS one* 5, 10 (2010), e13728.

[53] Albert C Yang, Shi-Jen Tsai, Norden E Huang, and Chung-Kang Peng. 2011. Association of Internet search trends with suicide death in Taipei City, Taiwan, 2004–2009. *Journal of affective disorders* 132, 1 (2011), 179–184.

[54] Shan Yin and Mona Ho. 2012. Monitoring a toxicological outbreak using Internet search query data. *Clinical toxicology* 50, 9 (2012), 818–822.

[55] Elad Yom-Tov and Shira H Fischer. 2017. The Werther Effect Revisited: Measuring the Effect of News Items on User Behavior. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 1561–1566.

[56] Soo Jeong Youn, Nhi-Ha Trinh, Irene Shyu, Trina Chang, Maurizio Fava, Joseph Kvedar, and Albert Yeung. 2013. Using online social media, Facebook, in screening for major depressive disorder among college students. *International Journal of Clinical and Health Psychology* 13, 1 (2013), 74–80.

The Matthew Effect in Computation Contests: High Difficulty May Lead to 51% Dominance*

Yulong Zeng
Nebulas
Beijing, China
yulong.zeng@nebulas.io

Song Zuo
Google Research
Beijing, China
szuo@google.com

ABSTRACT

We study the computation contests where players compete for searching a solution to a given problem with a winner-take-all reward. The search processes are independent across the players and the search speeds of players are proportional to their computational powers. One concrete application of this abstract model is the mining process of proof-of-work type blockchain systems, such as Bitcoin. Although one's winning probability is believed to be proportional to his computational power in previous studies on Bitcoin, we show that it is not the case in the strict sense.

Because of the gaps between the winning probabilities and the proportions of computational powers, the Matthew effect will emerge in the system, where the rich get richer and the poor get poorer. In addition, we show that allowing the players to pool with each other or reducing the number of solutions to the problem may aggravate the Matthew effect and vice versa.

CCS CONCEPTS

• **Theory of computation** → *Algorithmic game theory*; *Solution concepts in game theory*; Convergence and learning in games.

KEYWORDS

Computation contests; Matthew effect; blockchain; Bitcoin

ACM Reference Format:
Yulong Zeng and Song Zuo. 2019. The Matthew Effect in Computation Contests: High Difficulty May Lead to 51% Dominance. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3308558.3313593

1 INTRODUCTION

In a computation contest, a group of players compete with each other on solving a given computational problem and only the player who first finds a solution to this problem will receive a fixed reward. To solve the problem, the most efficient way is to execute a random brute force search across the candidate set, so the computational power is a critical resource for each player to increase the probability of winning this contest. Practical examples abound, ranging from tuning the parameters for a deep neural network, selecting random seeds for reinforcement learning to recovering the spatial structure of proteins and discovering new drugs for cancers [4, 9, 13, 16, 17].

The mining process of *Bitcoin* [14] is a concrete example that perfectly matches the computation contest in terms of mathematics. The Bitcoin system is maintained as a chain of blocks and it awards the builder of each block by a certain amount of BTC, which is about 12.5 BTC (≈ 79,400 USD) at the time this paper is written. To build (or mine) a block, one must find a block header (a fixed-length bit string) whose hash value meets a certain criteria and announce it before other competitors. Because it is hard to invert the secure hash function, the most efficient way is to enumerate (in an arbitrary order) and verify the candidate block headers. This is exactly the case as we described in the computation contest and the mining process is now an extremely fierce competition — the total computational power of the community is roughly 48E (4.8×10^{19}) hash per second.[1]

To participate such a vast contest, understanding the winning probability or the expected reward is undoubtedly important to the players. So far, the winning probability of each player is believed to be proportional to his computational power (or his speed of verifying one candidate) [7]. Their argument is briefly as follows: For each player, the wins follow a Poisson process with a strength proportional to his computational power and hence the expected reward is also proportional to his computational power.

However, in the strict sense, this conclusion is incorrect due to the nature of draws without replacement in the computation contests: Conditioned on no solution has been found yet, the more candidates one has already checked, the higher probability the next candidate will be a solution. To see this concretely, take the following extreme case as an example. There are only two players with computational powers being 2 and 1. When there is only one solution to the given problem, the winning probabilities of them are in fact 3/4 and 1/4, instead of 2/3 and 1/3. In Section 3, we will demonstrate the analysis in details. In general, our analysis shows that the gap between one's winning probability and the proportion of his computational power increases as the number of solutions of the problem decreases.

Such gaps, if not negligible, will lead to the Matthew effect in the system where the players repeatedly participate the computation contest. In practice, the reward received now often results in increments of computational powers in the future, for example, research groups can attract more researcher and funds by their results and Bitcoin miners can purchase more machines using BTCs. The gaps, as we will show in this paper, offer higher increments for those

*We thank the anonymous reviewers for their helpful comments.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313593

[1] See https://www.blockchain.com/en/charts/hash-rate.

with relatively high computational powers and makes *the rich (relatively) richer and the poor (relatively) poorer*. Eventually, the entire system could be dominated by a single player, which is undesirable in most applications. The converging speed could be accelerated if the players are allowed to form groups or *pools* where they can jointly search for the solution and split the rewards.

Our analysis of the computation contest then suggests that keeping the number of solutions large is critical to suppress the Matthew effect in the system.

1.1 Our contribution

In this paper, we make the following contributions, where all the results apply to general multiple-solution cases:

- We give the winning probabilities of the players in closed-form, both for the single-solution case (Theorem 1) and the multiple-solution case (Theorem 9).
- In general, the probabilities are *not* proportional to the computational powers of the players. In extreme cases, this may lead to *the Matthew effect* in terms of computational powers: "the rich get richer and the poor get poorer".
- We introduce a pool choosing game between the players, where each of them can either specify one pool to join or keep himself independent. We analyse the incentives of the players under this game (Section 4) and identify a non-trivial Nash equilibrium (Theorem 4).
  In particular, in the Nash equilibrium, there is always a pool (or a single player) who has more than half of the total computational power. It implies that the Matthew effect could become even stronger when pooling is allowed. For the application of Bitcoin, it also means the 51% attack[2] could emerge in this case.
- Last but not least, we show that increasing the number of solutions will mitigate the Matthew effect. This is because as the number of solutions grows, the winning probabilities will gradually get close to being proportional to computational powers, where the gap between the probabilities and the proportions can be roughly bounded by the order of n^{-1} (Theorem 11).

1.2 Related work

The blockchain and Bitcoin are first presented in 2008 under the pseudonym Nakamoto [14] and widely concerned in the past decade. Thousands of papers follow the work by Nakamoto [14] with research areas including cryptography, game theory, and distributed systems. The book "Mastering Bitcoin" [1] introduces the basic knowledge about Bitcoin for beginners and Tschorsch and Scheuermann [19] summarize some early studies and progresses of the Bitcoin techniques.

The game-theoretical analysis of the mining systems, where each miner performs as a self-interested player to maximize his reward, is one of the central topics of blockchain researches and most relevant to this work. Rosenfeld [15] introduces the attack called "pool-hopping", where the miner can work in one pool at the beginning of one round and hop to another pool when the current round is longer than a threshold, and analyzes the advantages and disadvantages under several reward mechanisms of mining pools. Eyal and Sirer [6] introduce the attack called "selfish mining". In such an attack, when a miner successfully mines a block, he can choose to not publish it and continue mining based on that block, called the private chain. Results show that a rational miner will follow the selfish mining strategy as long as the selfish miner controls at least 1/3 of the total computational power. Kiayias et al. [10] consider a mining game with similar attacks, where the model structure is a tree of blocks and a block becomes valid when its height is larger than a given threshold. Eyal [5] proposes an attack that a mining pool can send some "spies" to other pools and share the utilities of the spies obtained from these pools. Liu et al. [12] consider how a miner should choose a mining pool of which the required computational power is fixed. Fisch et al. [7] show that the geometric pay pool achieves the optimal social welfare based on the power utility function.

In the famous economic model, R&D race [3, 8], a similar setting is analyzed, where each agent can choose his effort rate to the winner-takes-all competition. However, the expected profits are still assumed to be linear to the agents' effort rates with additional costs. To the best of our knowledge, no literature drops the assumption about the proportional winning probability and this paper is the first to mathematically consider the winning probabilities in first principles.

2 PRELIMINARIES

Consider the following model of an abstract computation contest where the problem is solved via random trials:

- m players compete with each other in solving a problem;
- The problem is to find a solution of $h(v) = 0$ from a candidate set V, where $|V| = N$ is finite;
- There are $n \geq 1$ solutions to the problem, but only the player who first finds one solution receives a reward 1.

In particular, there is no trick for solving the problem in the sense that enumerating and verifying all candidates from V in an arbitrary order is the best algorithm. Formally, let h be randomly drawn from a ground set $\mathcal{H}$ according to some distribution, such that for any $v \in V$, the event $h(v) = 0$ is independent of the function values of any subset of $V \setminus \{v\}$. In addition, the probability of $h(v) = 0$ is the same for any $v \in V$ and the computational resource needed to determine whether $h(v) = 0$ or not is the also the same for any $v \in V$, both independent of the knowledge about the function values on $V \setminus \{v\}$.[3]

Throughout the paper, we refer each $v \in V$ as a *candidate* and the ones such that $h(v) = 0$ as *solutions*. In the contest, each player i has a type $x_i > 0$ called the *computational power*, i.e., the number of candidates he can verify in a unit of time. The player with higher computational power will be able to test more candidates and of course has higher chance to win. Here, we assume that each

[2]The 51% attack is a concept in PoW blockchain systems [19]. It states that if one player (or a pool) has computational power more than half of the total computational power in the system, then he/she is able to manipulate the blockchain.

[3]Although the assumption here can hardly be exactly met in real applications, our analysis still applies at some extend. For example, in the case of blockchain under PoW schemes, the hash function h is fixed and so are its solutions. However, as long as the hash function is still secure, the algorithmic advantage of solving the problem is negligible and every one is searching in a random order. As a result, the probability of the t-th candidate v_t being a solution is still a constant.

player is enumerating through the candidate set in a random order independent of others.

Blockchain. One concrete real-world example of this computation contest is the application of blockchains under Proof of Work (PoW) schemes. As its name would suggest, a *blockchain* is a chain of many blocks that are accepted by the peers, where each accepted block must have a valid *block header* pointing to the previous block.

Taking Bitcoin as the example: Each block header has a fixed length of 80 bytes (640 bits) which in general could be divided into two parts: (i) the fixed part (e.g., the hash value of the previous block header) and (ii) the adjustable part (e.g., the *nonce* and the *merkle root*). In other words, there is a fixed and finite set V containing all block header candidates.

A block header is valid if its hash value (a 256-bit integer) is less than a target threshold, i.e., $2^{208}(2^{16}-1)/D \approx 2^{224}/D$, where D is a positive number called *difficulty*. In this case, the probability of a randomly selected block header being valid is $2^{224}/D \cdot 2^{-256} = 2^{-32}D$. In the language of our computation contest model, the function value $h(v)$ of a candidate block header v is 0, if and only if v is a valid block header.

Since the hash function is deterministic, for any specific block, the number of valid block headers (solutions), n, is fixed (but unknown). Note that the fixed part of block header is randomly determined by the previous block and the system, therefore it can be seen as the random index of the function h, given that the hash function is ideal.

The main computational challenge of building the next block is to find one of the n valid block headers from the N possible candidates, or equivalently, solving $h(v) = 0$. In particular, if the hash function resists preimage attacks, the most efficient algorithm one can hope for is the brute force search (in arbitrary orders). In the context of blockchain, the procedure of searching a valid block header is referred as *mining* and the speed of computing the hash function is often referred as the *hash rate* or the computational power.

In the classic PoW scheme, such as Bitcoin, every miner has the equal right to mine and the one who first announces the next block with a valid block header will win the reward on that block. The reward is a constant with respect to the mining procedure. Therefore, the mining process of each block could be well-modeled by our computation contest, where each miner corresponds to a player, block headers correspond to candidates, and valid block headers correspond to solutions.

Pooling. In practice, the probability of winning the contest could be extremely low for a single player and hence the variance of the random reward is very high. To avoid the high variance, it is a common practice that individuals form a *pool* and jointly search for the solution. Once the pool wins, the reward will be distributed to the players in the pool according to a pre-specified policy. The most common policy, in the case of Bitcoin, is essentially a proportional allocation rule, i.e., the reward a player receives is proportional to the computational power he contributes to the pool.[4] In the same spirit, work by Bloch [2] analyzes sequential coalition game, splitting the whole party into subsets, and studies subgame perfect equilibrium.

[4]As recorded on https://investoon.com/mining_pools/btc, the most common reward schemes in the pools are PROP, PPS, PPLNS, all with expected reward approximately proportional to their hashrates.

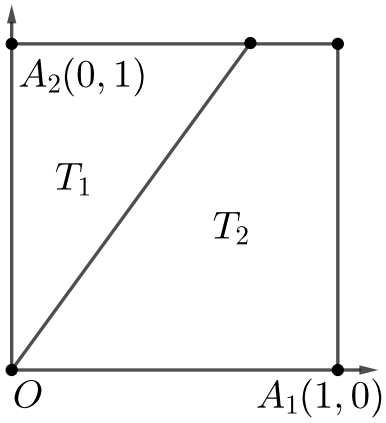


Figure 1: In the two-player case, player 1 wins if and only if the random coordinate t falls into T_1.

In this paper, we focus on two types of questions under this computation contest model:

- Given the environment and the computational powers, what are the winning probabilities of the players? Are the probabilities proportional to their computational powers?
- If not proportional to their computational powers, what is the optimal pooling strategy for the players?

3 THE SINGLE-SOLUTION CASE

In this section, we start with the basic case with only one solution, i.e., $n = 1$. Later in Section 5, we will generalize all results for single-solution cases to multiple-solution cases.

3.1 Warmup: two players and a single solution

As defined in the previous section, the player who find its first solution earliest wins. To compute the winning probabilities of the players, we use real numbers in $[0,1]$ to index the candidates in the order that they are verified by player i.[5] Since each candidate in general will have different indices in the enumerating sequences of different players, putting the indices together, a *coordinate* of each candidate is defined. In particular, let t_i be the index of the first solution in player i's sequence and then t_i/x_i is the time that player i first finds a solution. In this case, the player with minimum t_i/x_i wins. Note that the t_i's are random variables and also the source of the randomness of the outcome.

Let p_i be the winning probability of player i, we have

$$p_i = \Pr[i \text{ wins}] = \Pr\left[t_i/x_i \le \min_{j\neq i} t_j/x_j\right].$$

[5]Here we simplify the analysis by using real numbers to index the candidates. One can of course use the fractions k/N with integers $k < N$ to index the candidates for an even more rigorous analysis. In this case, our analysis still applies by replacing the integrations with summations. The only difference here is the break-ties at $t_i/x_i = t_j/x_j$. Their probabilities are zero under real number indexing but at order $1/N$ for fraction number indexing, which is also negligible for large N.

For the two-player single-solution case, since there is only one solution, t_1 and t_2 are independently and uniformly distributed over $[0,1]$. Denote $\boldsymbol{t} = (t_1, t_2)$ and let T_1 be the set of $\boldsymbol{t}$ where player 1 wins:

$$T_1 = \{\boldsymbol{t} \in [0,1]^2 : t_1/x_1 \le t_2/x_2\}.$$

Similarly, we can define T_2:

$$T_2 = \{\boldsymbol{t} \in [0,1]^2 : t_2/x_2 \le t_1/x_1\}.$$

Because $\boldsymbol{t}$ is uniformly distributed over $[0,1]^2$, $p_1 = \int_{T_1} \mathrm{d}\boldsymbol{t} = \mathrm{area}(T_1)$ equals to the area of set T_1 (see Figure 1). Suppose that $x_1 \le x_2$, then with some primary calculation, we know that the slope of the line separating T_1 and T_2 is x_2/x_1 and hence

$$p_1 = \mathrm{area}(T_1) = x_1/2x_2,\ p_2 = \mathrm{area}(T_2) = 1 - x_1/2x_2,$$

which are not proportional to the computational power x_1 and x_2 (unless $x_1 = x_2$).

3.2 m players and a single solution

For the general m-player case, the winning probability of each player is summarized by the following theorem, where $\pi(z)$ and $\pi_{-i}(z)$ are defined to simplify the notations:

$$\pi(z) = \prod_{i=1}^{m} \left(1 - \frac{x_i}{x_m} \cdot z\right), \quad \pi_{-i}(z) = \frac{\pi(z)}{1 - \frac{x_i}{x_m} \cdot z}.$$

Theorem 1. *Suppose that $x_1 \le x_2 \le \cdots \le x_m$, then the winning probability of player i is:*

$$p_i = \frac{x_i}{x_m} \int_0^1 \pi_{-i}(z)\mathrm{d}z.$$

Proof. Recall that

$$p_i = \Pr\left[t_i/x_i \le \min_{j \ne i} t_j/x_j\right] = \Pr\left[\bigwedge_{j \ne i} t_i/x_i \le t_j/x_j\right].$$

Let $f_i(\cdot)$ be the probability density function of the random variable t_i, then

$$p_i = \int_0^1 \Pr\left[\bigwedge_{j \ne i} t_j \ge \frac{t_i x_j}{x_i} \middle| t_i = t\right] f_i(t)\mathrm{d}t.$$

Since the orders of the players enumerating through the candidates are independent, we have that

$$\Pr\left[\bigwedge_{j \ne i} t_j \ge \frac{t_i x_j}{x_i} \middle| t_i = t\right] = \prod_{j \ne i} \Pr\left[t_j \ge \frac{t x_j}{x_i}\right].$$

In the meanwhile, note that $t_1, \ldots, t_m$ are uniformly distributed, hence $f_i(t) = 1$ and

$$\Pr\left[t_j \ge \frac{t x_j}{x_i}\right] = \max\left\{1 - \frac{t x_j}{x_i}, 0\right\}.$$

Thus the production is zero for $t \ge x_i/x_m$, which implies

$$p_i = \int_0^{x_i/x_m} \prod_{j \ne i} \left(1 - \frac{t x_j}{x_i}\right) \mathrm{d}t.$$

By substituting $t = x_i z/x_m$, we complete the proof

$$p_i = \frac{x_i}{x_m} \int_0^1 \prod_{j \ne i} \left(1 - \frac{x_j}{x_m} \cdot z\right) \mathrm{d}z = \frac{x_i}{x_m} \int_0^1 \pi_{-i}(z)\mathrm{d}z.$$

□

According to Theorem 1, the winning probabilities p_i are proportional to the computational powers x_i if and only if the integration $\int_0^1 \pi_{-i}(z)\mathrm{d}z$ is a constant for all $1 \le i \le m$, which in general is not true.

In particular, as we will show in Section 5, the observation above holds even when there are multiple solutions, i.e., $n > 1$. Nevertheless, as n goes to infinity, the winning probabilities will gradually get close to being proportional with respect to the computational powers.

However, the larger the player's computational power is, the higher efficiency (winning probability divided by computational power) it achieves.

3.2.1 The Matthew Effect. The next theorem in fact indicates the Matthew effect in such systems: the player with higher computational power gets higher return (or higher probability to win) for each unit of their computational power, i.e., $p_i/x_i \le p_j/x_j$.

Theorem 2 (The Matthew Effect). *If $x_i \le x_j$, then $p_i/x_i \le p_j/x_j$, where the equality is reached if and only if $x_i = x_j$.*

If both of them invest all the rewards (such as bitcoins) they get to increase their future computational power, the ratio of x_j/x_i will become larger and larger over time:

Δx_i and Δx_j are proportional to p_i and p_j, respectively, hence

$$\Delta x_i/x_i \le \Delta x_j/x_j, \quad \Longrightarrow \quad (x_i + \Delta x_i)/(x_j + \Delta x_j) \le x_i/x_j.$$

In other words, the player with the highest computational power in the system will gradually take a more and more fraction of the total computational power and eventually dominate the entire system.

Proof of Theorem 2. To simplify the notations, write

$$\pi_{-ij}(z) = \frac{\pi(z)}{\left(1 - \frac{x_i}{x_m} \cdot z\right)\left(1 - \frac{x_j}{x_m} \cdot z\right)}.$$

Therefore, by $x_i \le x_j \le x_m$ and $\pi_{ij}(z) \ge 0$, we have that $\forall z \in [0,1]$,

$$\left(1 - \frac{x_j}{x_m} \cdot z\right) \cdot \pi_{-ij}(z) \le \left(1 - \frac{x_i}{x_m} \cdot z\right) \cdot \pi_{-ij}(z).$$

Finally, using Theorem 1, we conclude that

$$\begin{aligned} \frac{p_i}{x_i} &= \frac{1}{x_m} \int_0^1 \left(1 - \frac{x_j}{x_m} \cdot z\right) \cdot \pi_{-ij}(z)\mathrm{d}z \\ &\le \frac{1}{x_m} \int_0^1 \left(1 - \frac{x_i}{x_m} \cdot z\right) \cdot \pi_{-ij}(z)\mathrm{d}z = \frac{p_j}{x_j}. \end{aligned}$$

In particular, the equality is reached if and only if $x_i = x_j$. □

4 POOLING INCENTIVES

In this section, we investigate the incentives of players to pool together assuming that the reward is split proportional to their computational powers. In particular, we identify a non-trivial Nash equilibrium of the following *pool choosing game*.

Definition 3 (Pool choosing game). *Let $\mathcal{P}_1, \ldots, \mathcal{P}_m$ be m pools. The actions of each player is either to choose one from these m pools (say $\mathcal{P}_i$) or to be independent (denoted as $\perp$). The utility of each*

player is the expected winning reward and if a pool wins, the reward is split proportional to its members' computational powers.

In the practice of Bitcoin, there are many such pools. The pools have their own websites, cooperatively mining tools, and reward splitting policies. Usually, the pools always welcome miners to join so that they can attract more computational resources. In contrast, each individual miner can of course work by himself.

As a quick example, the action profile $(\mathcal{P}_1, \ldots, \mathcal{P}_1)$ results in a giant pool including everyone ($\{1, \ldots, m\}$) and the action profile $(\perp, \ldots, \perp)$ results in everyone being independent. Although trivial, the latter is a Nash equilibrium of the game. Because, for example, if player i deviates, then the result is that only player i in a pool and all others being independent. No difference.

The following theorem then identifies a non-trivial Nash equilibrium for two cases.

Theorem 4 (Nash Equilibrium). *Suppose that $x_m \geq x_{m-1} \geq \cdots \geq x_1$. If $x_m \leq x_1 + \cdots + x_{m-1}$, then the action profile $(\mathcal{P}_1, \ldots, \mathcal{P}_1)$ is a Nash equilibrium, yielding a giant pool.*

Otherwise, we have $x_m \geq x_1 + \cdots + x_{m-1}$, and the action profile $(\mathcal{P}_1, \ldots, \mathcal{P}_1, \perp)$ is a Nash equilibrium, with player m being independent and the others in one pool.

To prove this theorem, we first need to understand the conditions that a player has incentive to join a pool (or another player). With the theorems and lemmas in place, we prove Theorem 4 at the end of this section.

We also emphasize that the results in this section will later be generalized to the multiple-solution case in Section 5.

4.1 Incentive Analysis

First of all, an important and immediate observation for pooling is that pooling is always beneficial when taking the pool as a whole. Formally,

Observation 5. *A pool's winning probability is no less than the sum of the winning probabilities of its members when they participate the computation contest separately.*

This is easy to get by considering a "pseudo-pool" of the members, where they enumerate through the candidates in the same order as they do when participate the contest separately. Clearly, the winning probability of the "pseudo-pool" equals to the sum of the winning probabilities of the members working separately. Since the players in the "pseudo-pool" may repeatedly verify some of the candidates, the winning probability could be even higher if they simply skip those already verified by other members in the pool. In particular, this "higher" winning probability equals to the winning probability of the real pool by symmetry.

Theorem 6. *For any player i, pooling with another player with higher computational power than him is always beneficial.*

Proof. Consider a player i pooling with another player j with $x_j \geq x_i$. We use p_{ij} to denote the winning probability of the pool. Then by Observation 5, we have $p_{ij} \geq p_i + p_j$ and hence the reward of player i from the pool is

$$p_i' = \frac{x_i}{x_i + x_j} \cdot p_{ij} = \frac{p_{ij}}{1 + x_j/x_i} \geq \frac{p_i + p_i}{1 + x_j/x_i}.$$

Note that by Theorem 2, we have

$$p_i/x_i \leq p_j/x_j \quad \Longleftrightarrow \quad x_j/x_i \leq p_j/p_i.$$

Therefore

$$p_i' \geq \frac{p_i + p_i}{1 + x_j/x_i} \geq \frac{p_i + p_j}{1 + p_j/p_i} = p_i.$$

In other words, player i gets more reward by pooling with player j. □

A direct consequence of this theorem is that a pool is stable against single deviation as long as the largest player's computational power is no more than half of the computational power of the pool.

Corollary 7. *Any individual member of a pool has no incentive to leave if its computational power is no more than half of the pool's computational power.*

Proof. Otherwise, once the player left the pool, he would have incentive to rejoin the pool by Theorem 6, a contradiction. □

In fact, the corollary asserts that any player (or pool) always has incentive to pool with a player (or pool) that has higher computational power than himself. In contrast, there would be a tradeoff for the player (or pool) being proposed:

- **Pros**: By pooling together, they form a larger pool, which by Theorem 2, has a higher efficiency in terms of computational power (i.e., higher ratio of reward divided by computational power).
- **Cons**: However, the player (or pool) with low computational power may actually takes more than he contributes. In this case, the total tradeoff for the player with high computational power could be negative and he would have incentive to be independent.

The following lemma then provides a sufficient condition where the tradeoff is positive.

Lemma 8. *If $x_i + x_j \leq x_m$, then both i and j will be better off in terms of the expected reward by pooling.*

Proof. Throughout the proof, we use the notations $\pi_{-ij}(z)$ (defined in the proof of Theorem 2) and p_{ij} (the winning probability of the pool $\{i, j\}$).

Without loss of generality, suppose that $x_i \leq x_j \leq x_m$. Then by Theorem 6, player i gets better off by pooling. In the rest of the proof, we show that player j gets better off as well.

On one hand, according to the proportional allocation rule, the reward of player j by pooling is $p_j' = \frac{x_j}{x_i + x_j} \cdot p_{ij}$. Thinking the pool $\{i, j\}$ as a player with computational power $x_i + x_j$, then by Theorem 1,

$$p_{ij} = \frac{x_i + x_j}{x_m} \int_0^1 \pi_{-ij}(z) \mathrm{d}z.$$

Hence

$$p_j' = \frac{x_j}{x_i + x_j} \cdot p_{ij} = \frac{x_j}{x_m} \int_0^1 \pi_{-ij}(z) \mathrm{d}z.$$

On the other hand, the reward of player j without pooling is p_i, which by Theorem 1 is,

$$p_j = \frac{x_j}{x_m} \int_0^1 \pi_{-j}(z)\mathrm{d}z.$$

Note that for any $z \in [0, 1)$, since $x_i < x_m$ and $\pi_{-ij}(z) > 0$,

$$\pi_{-ij}(z) > \left(1 - \frac{x_i}{x_m} \cdot z\right) \pi_{-ij}(z) = \pi_{-j}(z).$$

Therefore,

$$p'_j > p_j.$$

In other words, player j gets strictly higher expected reward by pooling with player i. □

4.2 Equilibrium Analysis

Now we are ready to prove Theorem 4.

Theorem (Theorem 4 restate). Suppose that $x_m \geq x_{m-1} \geq \cdots \geq x_1$. If $x_m \leq x_1 + \cdots + x_{m-1}$, then the action profile $(\mathcal{P}_1, \ldots, \mathcal{P}_1)$ is a Nash equilibrium, yielding a giant pool.

Otherwise, we have $x_m \geq x_1 + \cdots + x_{m-1}$, and the action profile $(\mathcal{P}_1, \ldots, \mathcal{P}_1, \perp)$ is a Nash equilibrium, with player m being independent and the others in one pool. □

Proof of Theorem 4. The first case is a direct consequence of Corollary 7. Because none of them has computational power more than half of the pool, no player has incentive to deviate (leave the pool).

For the second case, we first show that player m has no incentive to deviate (or equivalently, join pool $\mathcal{P}_1$).

We use x_{-m} and p_{-m} to denote the computational power and the winning probability of the pool $\{1, \ldots, m-1\}$, respectively. Since $x_m \geq x_{-m}$, by Theorem 2, $p_m/x_m \geq p_{-m}/x_{-m}$. Note that $p_m + p_{-m} = 1$, hence $p_{-m} = 1 - p_m$. Then we have

$$\frac{p_m}{x_m} \geq \frac{1 - p_m}{x_{-m}} \iff p_m \geq \frac{x_m}{x_{-m} + x_m},$$

where the right-hand-side is the reward of player m if he joins the pool. In other words, if player m deviates by joining pool $\mathcal{P}_1$, his winning probability will decreases.

Then we prove that any player $i \in \{1, \ldots, m-1\}$ has no incentive to deviate (or equivalently, leave pool $\mathcal{P}_1$).

To show this, we prove that player i has incentive to join pool $\mathcal{P}'_1$, when $\mathcal{P}'_1 = \{1, \ldots, i-1, i+1, \ldots, m-1\}$ and player m chooses $\perp$. Thinking the pool $\mathcal{P}'_1$ as a player with computational power $x' = x_1 + \cdots + x_{i-1} + x_{i+1} + \cdots + x_{m-1}$. Since $x_i + x' \leq x_m$, then by Lemma 8, player i has incentive to join pool $\mathcal{P}'_1$. In other words, player i gets (weakly) higher reward in pool $\mathcal{P}_1$ than being independent.

In summary, for the second case, no player has incentive to deviate, hence a Nash equilibrium. □

5 THE MULTIPLE-SOLUTION CASE

In this section, we first extend our results of single-solution cases to multiple-solution cases. Then we show that as the number of solutions goes to infinity, the winning probabilities become asymptotically proportional to the player computational powers. In particular, it means that the Matthew effect gets weaker as the number of solutions increases.

5.1 Generalization

We start with generalizing Theorem 1.

Theorem 9. *Suppose that $x_1 \leq \cdots \leq x_m$, then the winning probability for player i in the multiple-solution case is*

$$p_i = \frac{x_i}{x_m} \int_0^1 \pi_{-i}(z) \cdot \pi(z)^{n-1}\mathrm{d}z.$$

We omit the proof of this theorem but highlight the only two differences with the proof of Theorem 1. Both are because of the definition of t_i, the index of *the first solution* in player i's sequence. Therefore, the probability density function of t_i and the probability that $t_j \geq t x_j/x_i$ are different with those in the single-solution case:

$$f_i(t) = n(1-t)^{n-1}, \Pr\left[t_j \geq \frac{t x_j}{x_i}\right] = \max\left\{1 - \frac{t x_j}{x_i}, 0\right\}^n.$$

Then by substituting these two terms in the proof of Theorem 1, we get Theorem 9.

Based on Theorem 9, we claim that all previous results (except Theorem 1) also apply to multiple-solution cases:

Claim 10 (Generalization). *Theorem 2 and Lemma 8 can be easily reproved for multiple-solution cases with Theorem 9. Based on this, Theorem 4, Observation 5, Theorem 6, and Corollary 7 are automatically generalized because their proofs do not rely on the number of solutions.*

5.2 Asymptotically Proportional Winning Probabilities

Consider the probability that the first solution comes after $t_1, \ldots, t_m$ for each of the players. Since the distribution of each t_i is independent, we have:

$$\Pr[\boldsymbol{\tau} \geq \boldsymbol{t}] = \prod_{i=1}^m \Pr[\tau_i \geq t_i] = \prod_{i=1}^m (1 - t_i)^n,$$

hence its density function:

$$f(\boldsymbol{t}) = \prod_{i=1}^n f_i(t_i) = \prod_{i=1}^n \frac{\partial (1 - t_i)^n}{\partial t_i} = n^m \prod_{i=1}^m (1 - t_i)^{n-1}.$$

Let T_i be the set of $\boldsymbol{t}$ that player i wins and $T = T_1 \cup \cdots \cup T_m = [0, 1]^m$. Then the winning probability of player i is

$$p_i = \int_{\boldsymbol{t} \in T_i} f(\boldsymbol{t})\mathrm{d}\boldsymbol{t}.$$

Theorem 11 (Asymptotically Proportional). *For any fixed number of players, m, as the number of solutions, n, approaches to infinity,*

$$p_i \longrightarrow \frac{x_i}{x_1 + \cdots + x_m}.$$

In particular, the difference is vanishing in the order of $\widetilde{O}(n^{-1})$,[6] *i.e.,*

$$\left| p_i - \frac{x_i}{x_1 + \cdots + x_m} \right| = \widetilde{O}(n^{-1}).$$

The theorem could be directly derived based on the following Lemma 12 and Lemma 13.

We use $\rho_i(z)$ to denote the $(m-1)$-dimensional volume of set $\{\boldsymbol{t} \in T_i : t_1 + \cdots + t_m = z\}$ and $\rho(z) = \rho_1(z) + \cdots + \rho_m(z)$.

Then Lemma 12 essentially means that $\boldsymbol{p}_i$ is asymptotically proportional to $\rho_i(1)$.

Lemma 12. *For any constant* $m \ll n$,

$$p_i = \frac{\rho_i(1)}{\rho(1)} \cdot n^m \int_0^{n^{-1+\epsilon}} (1-z)^{n-1} \rho(z) \mathrm{d}z + \widetilde{O}(n^{-1}).$$

Furthermore, Lemma 13 indicates that $\rho_i(1)$ is proportional to the computational power of player i.

Lemma 13.

$$\frac{\rho_i(1)}{\rho(1)} = \frac{x_i}{x_1 + \cdots + x_m}.$$

Proof of Theorem 11. By Lemma 12 and Lemma 13. □

We dedicate the rest of this section to prove Lemma 12 and Lemma 13.

Proof idea of Lemma 12. The proof is done by a careful estimation of p_i using a technique called the *core-tail decomposition* [11]. To estimate the probability, consider the core-tail decomposition of the set T_i, i.e., let

$$\mathsf{core} = \{\boldsymbol{t} : \bar{t} \le \alpha\} \quad \text{and} \quad \mathsf{core}(T_i) = \mathsf{core} \cap T_i,$$

where $\bar{t} = t_1 + \cdots + t_m$ and $\alpha > 0$ is a parameter to be determined later. Similarly, define $\mathsf{tail} = [0,1]^m \setminus \mathsf{core}$ and $\mathsf{tail}(T_i) = \mathsf{tail} \cap T_i$. Thus,

$$p_i = \int_{\mathsf{core}(T_i)} f(\boldsymbol{t}) \mathrm{d}\boldsymbol{t} + \int_{\mathsf{tail}(T_i)} f(\boldsymbol{t}) \mathrm{d}\boldsymbol{t}. \tag{1}$$

The following two lemmas would be helpful on simplifying the analysis.

Lemma 14. *Let* $g(\boldsymbol{t}) = n^m (1-\bar{t})^{n-1}$. *If* $\alpha < n^{-2/3}$, *then*

$$\int_{\mathsf{core}(T_i)} f(\boldsymbol{t}) \mathrm{d}\boldsymbol{t} = \int_{\mathsf{core}(T_i)} g(\boldsymbol{t}) \mathrm{d}\boldsymbol{t} + O(n^{m+1} \alpha^{m+2}).$$

Lemma 15.

$$\int_{\mathsf{tail}(T_i)} f(\boldsymbol{t}) \mathrm{d}\boldsymbol{t} \le e^{-\alpha(n-1) + m \ln n}.$$

We postpone the proofs of these lemmas to Section 6, which are mainly elaborate treatments of the Taylor series.

Proof of Lemma 12. Plugging Lemma 14 and Lemma 15 into (1), and letting $\alpha = \frac{(m+1)\ln n}{n-1}$, we have that,

$$p_i - \int_{\mathsf{core}(T_i)} g(\boldsymbol{t}) \mathrm{d}\boldsymbol{t} = O\left(n^{-1} (\ln n)^{m+2}\right) = \widetilde{O}(n^{-1}).$$

[6] The $\widetilde{O}$ notation here hides a poly-log factor, meaning that the residue term is bounded by $O\left(n^{-1}(\ln n)^c\right)$, for some positive constant $c > 0$ when n is sufficiently large.

Hence we remain to show the following equation:

$$\int_{\mathsf{core}(T_i)} g(\boldsymbol{t}) \mathrm{d}\boldsymbol{t} = \frac{\rho_i(1)}{\rho(1)} \cdot n^m \int_0^{\alpha} (1-z)^{n-1} \rho(z) \mathrm{d}z,$$

or equivalently,

$$\int_{\mathsf{core}(T_i)} (1-\bar{t})^{n-1} \mathrm{d}\boldsymbol{t} = \frac{\rho_i(1)}{\rho(1)} \int_0^{\alpha} (1-z)^{n-1} \rho(z) \mathrm{d}z. \tag{$*$}$$

First of all, note that the left-hand-side of $(*)$ could be rewritten as a double integration over $\mathsf{core}_z(T_i) = \{\boldsymbol{t} \in \mathsf{core}(T_i) : \bar{t} = z\}$ and then for z from 0 to α:

$$\begin{aligned}\int_{\mathsf{core}(T_i)} (1-\bar{t})^{n-1} \mathrm{d}\boldsymbol{t} &= \int_0^{\alpha} \mathrm{d}z \int_L (1-z)^{n-1} \mathrm{dcore}_z(T_i) \\ &= \int_0^{\alpha} (1-z)^{n-1} \mathrm{d}z \int_L \mathrm{dcore}_z(T_i),\end{aligned}$$

where $\int_L \mathrm{d}S$ is the Lebesgue integration over a set S. In particular, $\int_L \mathrm{dcore}_z(T_i)$ is the $(m-1)$-dimensional volume of the set $\mathsf{core}_z(T_i)$, denoted as $\rho_i(z)$.

Then consider the ratio between $\rho_i(z)$ and the volume of set $\mathsf{core}_z(T)$, $\rho(z)$. Since both $\mathsf{core}(T_i)$ and $\mathsf{core}(T)$ are the intersections of the core and positive cones in $\mathbb{R}^m$, hence $\rho_i(z)$ and $\rho(z)$ must be proportional to z^{m-1} accordingly:

$$\rho_i(z) = z^{m-1} \rho_i(1), \quad \rho(z) = z^{m-1} \rho(1).$$

Finally, we conclude that

$$\int_L \mathrm{dcore}_z(T_i) = \rho_i(z) = \frac{\rho_i(1)}{\rho(1)} \cdot \rho(z),$$

which then directly implies $(*)$. □

Proof idea of Lemma 13. To smoothly proceed the analysis, we will start with the cases of $m = 2$ and $m = 3$.

For the two-player case (see Figure 2a), by definition,

$$T_1 = \{\boldsymbol{t} \in [0,1]^2 : t_1/x_1 \le t_2/x_2\},$$
$$T_2 = \{\boldsymbol{t} \in [0,1]^2 : t_2/x_2 \le t_1/x_1\}.$$

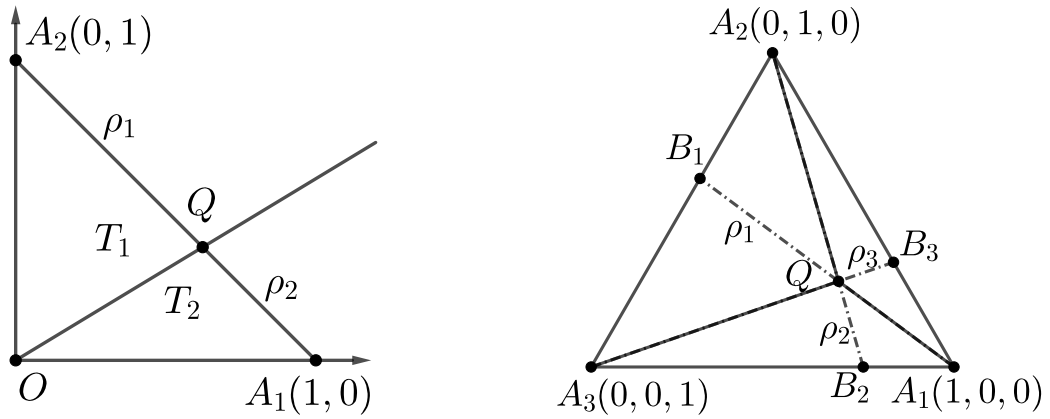
Therefore, the formula of the hyperplane (in fact a line in the 2-dimensional case) separating T_1 and T_2 is $t_1/x_1 = t_2/x_2$ and Q is the intersection of this line and the simplex A_1A_2. In particular, the 1-dimensional volume of the set $\{\boldsymbol{t} \in T_1 : \bar{t} = 1\}$ is exactly the length of the line segment QA_2, which equals to $\rho_1(1) = \sqrt{2}x_1/(x_1 + x_2)$. Similarly, we get $\rho_2(1) = \sqrt{2}x_2/(x_1 + x_2)$.

For the three-player case (see Figure 2b), by definition,

$$T_1 = \{\boldsymbol{t} \in [0,1]^3 : t_1/x_1 \le t_2/x_2, t_1/x_1 \le t_3/x_3\},$$
$$T_2 = \{\boldsymbol{t} \in [0,1]^3 : t_2/x_2 \le t_1/x_1, t_2/x_2 \le t_3/x_3\},$$
$$T_3 = \{\boldsymbol{t} \in [0,1]^3 : t_3/x_3 \le t_1/x_1, t_3/x_3 \le t_2/x_2\}.$$

In this case, there are three hyperplanes separating T_1, T_2, and T_3: $t_1/x_1 = t_2/x_2$, $t_2/x_2 = t_3/x_3$, and $t_3/x_3 = t_1/x_1$. Note that the intersection of the three hyperplanes is a line, $t_1/x_1 = t_2/x_2 = t_3/x_3$, which then intersects with the simplex $A_1A_2A_3 = \{\boldsymbol{t} : t_1 + t_2 + t_3 = 1\}$ at a point Q.

Therefore, the 2-dimensional volumes (or areas) $\rho_1(1)$, $\rho_2(1)$, and $\rho_3(1)$ are exactly the areas of triangles $\triangle QA_2A_3$, $\triangle A_1QA_3$, and $\triangle A_1A_2Q$, respectively. Note that $A_1A_2A_3$ is a regular 2-simplex,



(a) The two-player case. **(b) The three-player case.**

Figure 2: Simplexes for two-player and three-player cases.

hence $A_2A_3 = A_3A_1 = A_1A_2$. Thus the ratios between the areas of the triangles equal to the ratios between the heights of the triangles, because their bases (A_2A_3, A_3A_1, A_1A_2) are of the same length.

We conclude that

$$\frac{\rho_1(1)}{\rho_2(1)} = \frac{d(Q, A_2A_3)}{d(Q, A_3A_1)}, \frac{\rho_1(1)}{\rho_3(1)} = \frac{d(Q, A_2A_3)}{d(Q, A_1A_2)},$$

where d denotes the distance between a point and a line.

In the meanwhile, recall that the coordinate of point Q is:

$$\left(\frac{x_1}{x_1+x_2+x_3}, \frac{x_2}{x_1+x_2+x_3}, \frac{x_3}{x_1+x_2+x_3}\right),$$

and with some primary calculation, we know that

$$\frac{d(Q, A_2A_3)}{d(Q, A_3A_1)} = \frac{x_1}{x_2} \quad \text{and} \quad \frac{d(Q, A_2A_3)}{d(Q, A_1A_2)} = \frac{x_1}{x_3}.$$

In other words, $\rho_i(1)/\rho(1) = x_i/(x_1+x_2+x_3)$.

Finally, we generalize the observations above to get the proof.

Proof of Lemma 13. By the definition, $T_1, \ldots, T_m$ are separated by $\binom{m}{2}$ hyperplanes:

$$\forall i \neq j, t_i/x_i = t_j/x_j.$$

In particular, the intersection of them is a line and the line intersects the $(m-1)$-simplex $A_1 \cdots A_m$ at a unique point Q, whose coordinate is:

$$Q : \left(\frac{x_1}{x_1+\cdots+x_m}, \ldots, \frac{x_m}{x_1+\cdots+x_m}\right).$$

In the meanwhile, note that for each $i \in [m]$, point $A_j \in T_i$ for all $j \neq i$. Therefore, the set $\{\boldsymbol{t} \in T_i : \bar{t} = 1\}$ is a $(m-1)$-simplex, $A_1 \cdots A_{i-1} Q A_{i+1} \cdots A_m$ and its volume is proportional to the $(m-2)$-dimensional volume of its base $A_1 \cdots A_{i-1} A_{i+1} \cdots A_m$ times its height (the distance between point Q and the base).

Because $A_1 \cdots A_m$ is a regular $(m-1)$-simplex, the volumes of the bases for each i are the same. Therefore, $\rho_i(1)/\rho_j(1)$ equals to the ratios between the distances from Q to the corresponding bases. Based on the coordinate of Q, we conclude that the ratio equals to x_i/x_j. In other words,

$$\frac{\rho_i(1)}{\rho(1)} = \frac{\rho_i(1)}{\rho_1(1)+\cdots+\rho_m(1)} = \frac{x_i}{x_1+\cdots+x_m}.$$

□

6 MISSING PROOFS

The following technical lemma is repeatedly used in the proofs.

Lemma 16. *For all $x \in \mathbb{R}$, $e^x \geq 1+x$.*

Proof of Lemma 14. For the core part, because $\bar{t} < \alpha < n^{-2/3}$,

$$\prod_{i=1}^{m}(1-t_i) = 1 - \sum_i t_i + \sum_{i\neq j} t_i t_j + \cdots = 1 - \bar{t} + O(\bar{t}^2),$$

$$\prod_{i=1}^{m}(1-t_i)^{n-1} = (1-\bar{t})^{n-1}\left(1 + \frac{O(\bar{t}^2)}{1-\bar{t}}\right)^{n-1}.$$

Since $n\bar{t}^2 \leq n\alpha^2 < n^{-1/3}$, then by Lemma 16 and Taylor's expansion of e^x at 0,

$$\begin{aligned}\prod_{i=1}^{m}(1-t_i)^{n-1} &\leq (1-\bar{t})^{n-1} e^{\frac{O(\bar{t}^2)}{1-\bar{t}}\cdot(n-1)} \\ &= (1-\bar{t})^{n-1}\cdot\left(1 + \frac{O(\bar{t}^2)(n-1)}{1-\bar{t}} + O(n^2\bar{t}^4)\right) \\ &= (1-\bar{t})^{n-1} + O(n\bar{t}^2).\end{aligned}$$

Thus

$$\int_{\mathsf{core}(T_i)} f(\boldsymbol{t})\mathrm{d}\boldsymbol{t} \leq \int_{\mathsf{core}(T_i)} n^m\left((1-\bar{t})^{n-1} + O(n\bar{t}^2)\right)\mathrm{d}\boldsymbol{t},$$

where

$$\int_{\mathsf{core}(T_i)} n^m \cdot O(n\bar{t}^2)\mathrm{d}\boldsymbol{t} \leq n^{m+1}\int_{\mathsf{core}} O(\bar{t}^2)\mathrm{d}t.$$

Consider rewrite the right-hand-side as a double integration over $\mathsf{core}_z = \{\boldsymbol{t} \in \mathsf{core} : \bar{t} = z\}$ and then for z from 0 to α:

$$\int_{\mathsf{core}} O(\bar{t}^2)\mathrm{d}t = \int_0^\alpha O(z^2)\mathrm{d}z \int_L \mathrm{dcore}_z$$

Note that $\int_L \mathrm{dcore}_z$ is the Lebesgue integration over the set core_z and hence equals to its $(m-1)$-dimensional volume, which is $\rho(z) = \frac{\sqrt{m}}{(m-1)!} \cdot z^{m-1}$ [18]. Therefore,

$$\int_{\mathrm{core}} O(\bar{t}^2)\mathrm{d}t = \int_0^{\alpha} O(z^2) \cdot z^{m-1}\mathrm{d}z = O(\alpha^{m+2}).$$

That is

$$\int_{\mathrm{core}(T_i)} n^m \cdot O(n\bar{t}^2)\mathrm{d}\boldsymbol{t} \le O(n^{m+1}\alpha^{m+2}).$$

In summary,

$$\int_{\mathrm{core}(T_i)} f(\boldsymbol{t})\mathrm{d}\boldsymbol{t} = \int_{\mathrm{core}(T_i)} g(\boldsymbol{t})\mathrm{d}\boldsymbol{t} + O(n^{m+1}\alpha^{m+2}).$$

□

Proof of Lemma 15. For the tail part, according to the inequality of arithmetic and geometric means,

$$\prod_{i=1}^{m}(1-t_i) \le (1-\bar{t}/m)^m,$$

then by Lemma 16 and $\bar{t} \ge \alpha$,

$$\prod_{i=1}^{m}(1-t_i)^{n-1} \le (1-\bar{t}/m)^{m(n-1)} \le e^{-\bar{t}(n-1)} \le e^{-\alpha(n-1)}.$$

Therefore, $f(\boldsymbol{t}) \le n^m e^{-\alpha(n-1)} = e^{-\alpha(n-1)+m\ln n}$. Also note that $\int_{\mathrm{tail}} \mathrm{d}\boldsymbol{t}$ is the volume of tail, which is less than 1, hence

$$\int_{\mathrm{tail}(T_i)} f(\boldsymbol{t})\mathrm{d}\boldsymbol{t} \le e^{-\alpha(n-1)+m\ln n} \int_{\mathrm{tail}} \mathrm{d}\boldsymbol{t} \le e^{-\alpha(n-1)+m\ln n}.$$

□

REFERENCES

[1] Andreas M Antonopoulos. 2014. *Mastering Bitcoin: unlocking digital cryptocurrencies.* " O'Reilly Media, Inc.".

[2] Francis Bloch. 1996. Sequential formation of coalitions in games with externalities and fixed payoff division. *Games and Economic Behavior* 14, 1 (1996), 90–123.

[3] Pelin G Canbolat, Boaz Golany, Inbal Mund, and Uriel G Rothblum. 2012. A stochastic competitive R&D race where âĂIJwinner takes allâĂİ. *Operations Research* 60, 3 (2012), 700–715.

[4] Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popović, et al. 2010. Predicting protein structures with a multiplayer online game. *Nature* 466, 7307 (2010), 756.

[5] Ittay Eyal. 2015. The miner's dilemma. In *Security and Privacy (SP), 2015 IEEE Symposium on*. IEEE, 89–103.

[6] Ittay Eyal and Emin Gün Sirer. 2018. Majority is not enough: Bitcoin mining is vulnerable. *Commun. ACM* 61, 7 (2018), 95–102.

[7] Ben Fisch, Rafael Pass, and Abhi Shelat. 2017. Socially optimal mining pools. In *International Conference on Web and Internet Economics*. Springer, 205–218.

[8] Christopher Harris and John Vickers. 1987. Racing with uncertainty. *The Review of Economic Studies* 54, 1 (1987), 1–21.

[9] Scott Horowitz, Brian Koepnick, Raoul Martin, Agnes Tymieniecki, Amanda A Winburn, Seth Cooper, Jeff Flatten, David S Rogawski, Nicole M Koropatkin, Tsinatkeab T Hailu, et al. 2016. Determining crystal structures through crowdsourcing and coursework. *Nature communications* 7 (2016), 12549.

[10] Aggelos Kiayias, Elias Koutsoupias, Maria Kyropoulou, and Yiannis Tselekounis. 2016. Blockchain mining games. In *Proceedings of the 2016 ACM Conference on Economics and Computation*. ACM, 365–382.

[11] Xinye Li and Andrew Chi-Chih Yao. 2013. On revenue maximization for selling multiple independently distributed items. *Proceedings of the National Academy of Sciences* 110, 28 (2013), 11232–11237.

[12] Xiaojun Liu, Wenbo Wang, Dusit Niyato, Narisa Zhao, and Ping Wang. 2018. Evolutionary game for mining pool selection in blockchain networks. *IEEE Wireless Communications Letters* (2018).

[13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[14] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).

[15] Meni Rosenfeld. 2011. Analysis of bitcoin pooled mining reward systems. *arXiv preprint arXiv:1112.4980* (2011).

[16] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815* (2017).

[17] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of Go without human knowledge. *Nature* 550, 7676 (2017), 354.

[18] P Stein. 1966. A note on the volume of a simplex. *The American Mathematical Monthly* 73, 3 (1966), 299–301.

[19] Florian Tschorsch and Björn Scheuermann. 2016. Bitcoin and beyond: A technical survey on decentralized digital currencies. *IEEE Communications Surveys & Tutorials* 18, 3 (2016), 2084–2123.

Judging a Book by Its Cover: The Effect of Facial Perception on Centrality in Social Networks

Dongyu Zhang
Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116620, China

Teng Guo
Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116620, China

Hanxiao Pan
Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116620, China

Jie Hou
Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116620, China

Zhitao Feng
Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116620, China

Yang Liang
School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China

Hongfei Lin
School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China

Feng Xia
Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116620, China
f.xia@acm.org

ABSTRACT

Facial appearance matters in social networks. Individuals frequently make trait judgments from facial clues. Although these face-based impressions lack the evidence to determine validity, they are of vital importance, because they may relate to human network-based social behavior, such as seeking certain individuals for help, advice, dating, and cooperation, and thus they may relate to centrality in social networks. However, little to no work has investigated the apparent facial traits that influence network centrality, despite the large amount of research on attributions of the central position including personality and behavior. In this paper, we examine whether perceived traits based on facial appearance affect network centrality by exploring the initial stage of social network formation in a first-year college residential area. We took face photos of participants who are freshmen living in the same residential area, and we asked them to nominate community members linking to different networks. We then collected facial perception data by requiring other participants to rate facial images for three main attributions: dominance, trustworthiness, and attractiveness. Meanwhile, we proposed a framework to discover how facial appearance affects social networks. Our results revealed that perceived facial traits were correlated with the network centrality and that they were indicative to predict the centrality of people in different networks. Our findings provide psychological evidence regarding the interaction between faces and network centrality. Our findings also offer insights in to a combination of psychological and social network techniques, and they highlight the function of facial bias in cuing and signaling social traits. To the best of our knowledge, we are the first to explore the influence of facial perception on centrality in social networks.

CCS CONCEPTS

• **Applied computing** → **Psychology**; **Sociology**; • **Networks** → **Topology analysis and generation**.

KEYWORDS

Social Networks; Facial Perception; Bias; Centrality; Psychological Traits

ACM Reference Format:
Dongyu Zhang, Teng Guo, Hanxiao Pan, Jie Hou, Zhitao Feng, Yang Liang, Hongfei Lin, and Feng Xia. 2019. Judging a Book by Its Cover: The Effect of Facial Perception on Centrality in Social Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313527

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313527

1 INTRODUCTION

Many cultures share beliefs about the human face being a window to a personâĂŹs characters. Humans frequently make trait judgments from facial clues [29, 31, 33]. Studies have indicated that 100 milliseconds of exposure to unfamiliar faces is sufficient for

participants to form impressions about competence or trustworthiness [31]. Indeed, some have reported that people can make social judgments with exposures of less than 100 milliseconds, with any longer time being used to confirm judgments [2, 20, 29]. Moreover, there is a consensus concerning trait judgments based on faces among adults and even among 3- to 4-year-old children [9], despite the limited evidence of the validity of these judgments [23].

Although these facial impressions of traits may not be accurate, they are very important within social network environments, because they may influence social behaviors such as seeking certain individuals for help, advice, dating, and cooperation [24, 30]. For example, people invest more money with those who they perceive as more trustworthy in economic games [10, 22]. Evaluations of faces have impacted decisions in electoral politics [28], mate preferences [13], hypothetical crime verdicts [21], as well as approach or avoidance behaviors [12, 32] as well as approach or avoidance behaviours [27]. In particular, regarding network centrality, individualsâĂŹ roles and relationships with others are more likely to be influenced by facial appearance at the early stage of central networks construction without or with little interaction or information on others.

However, little to no research has explored the effect of facial perceptions on social networks. Although previous work has investigated the relationships between trait perceptions and social networks, those perceptions are based on assessments of personality, behavior, emotion, ability, etc. (e.g. [4, 8, 11, 14]) rather than appearances or facial perception. Moreover, recent work suggests that psychological traits (based on psychological scales) correlate with the centrality of social networks [16]. However, whether face-based perceived psychological traits relate to centrality of networks is scarcely known. We therefore aim to explore the relationship between facial appearance and social networks by exploring newly formed social networks in a first-year college residential area. We took face photos of participants and asked them to nominate community members associated with different networks. We then collected image evaluation data by asking other participants to rate face photos for three main attributions: dominance, trustworthiness, and attractiveness on a seven-point Likert-type scale. Our results revealed different social networks in different freshman residential areas. Moreover, we found that perceived facial dominance, trustworthiness, and attractiveness are effective at predicting the centrality of people on different networks. Our contributions are as follows.

- We explore the use of perceived facial features in a social network environment, and particularly for network centrality, to discover whether facial appearance affects social networks, which has scarcely been done before.
- We investigate the predictability of network centrality by rating face stimuli. The experimental results indicate that face-based traits predict membership of freshman community networks at the initial stage of formation.
- We propose a novel framework to conduct social network analysis in a psychological setting, and particularly for corroboration of facial bias. The results contribute to psychological knowledge of facial perception, because they are consistent with previous research that facial appearance affects network-based social behaviors.

This paper is organized as follows. In next section, we discuss the related work. In section 3, we introduce the details of data collection. In section 4, we describe methods used in this study. In section 5, we show the experiment results. In final section, we present conclusion of this paper.

2 RELATED WORK

2.1 Dimensions of Face Evaluation

Oosterhof and Todorov [19] provided empirical evidence contributing to our knowledge of face evaluation. To identify the dimensions on which individuals evaluate faces, they conducted a principal component analysis on various trait adjectives inferred from neutral faces, and they reduced them to two main axes; dominance and trustworthiness. They constructed computer models of three-dimensional face shapes that correlated with ratings of trustworthiness and dominance [19].They found that by moving from the negative to positive extremes (low and high ratings) of the trustworthiness dimension, the faces changed from expressing slight anger to slight happiness. Likewise, in the dominance dimension, the faces varied from expressing feminine and baby-faced traits to masculine and mature-faced traits. The trustworthiness dimension arguably signaled an individualâĂŹs harmful intent, whereas the dominance dimension signaled the capability to carry out the harmful intention [19]. Hence, they suggested that perhaps these evaluations of emotional valence and dominance are rooted in evolutionary mechanisms of threat detection [19].

Additionally, Sutherland et al [25]. described a third dimension of youthful attractiveness. In their study, they created average faces from photographs of people perceived as high or low in intelligence, trustworthiness, dominance, and confidence, and participants rated them on various traits. The stimuli of faces did not control for variables like hairstyle and piercings [25]. The stimuli of faces did not control for variables like hairstyle and piercings [25]. Their results supported the two dimensions proposed by Oosterhof and Todorov [19], and they indicated a third dimension of facial youthful attractiveness, which Sutherland et al. argued was important in making facial evaluations [25]. This additional dimension may be due to the wider age range of faces used in the study [25]. The aforementioned studies suggest that the main attributions made to faces are dominance, trustworthiness, and attractiveness; thus, we conducted studies based on these three dimensions.

Some studies have been done on how attractiveness, trustworthiness, and dominance impact network-related social behaviors. Alison et al. have studied how facial appearance can impact entrepreneurs to finance projects [5]. When entrepreneurs start a business, investors prefer pitches from attractive people, because they are much more persuasive. In strategic economic games, it is almost impossible for a player with an untrustworthy-looking face to be trusted by other players, even when he/she has a good

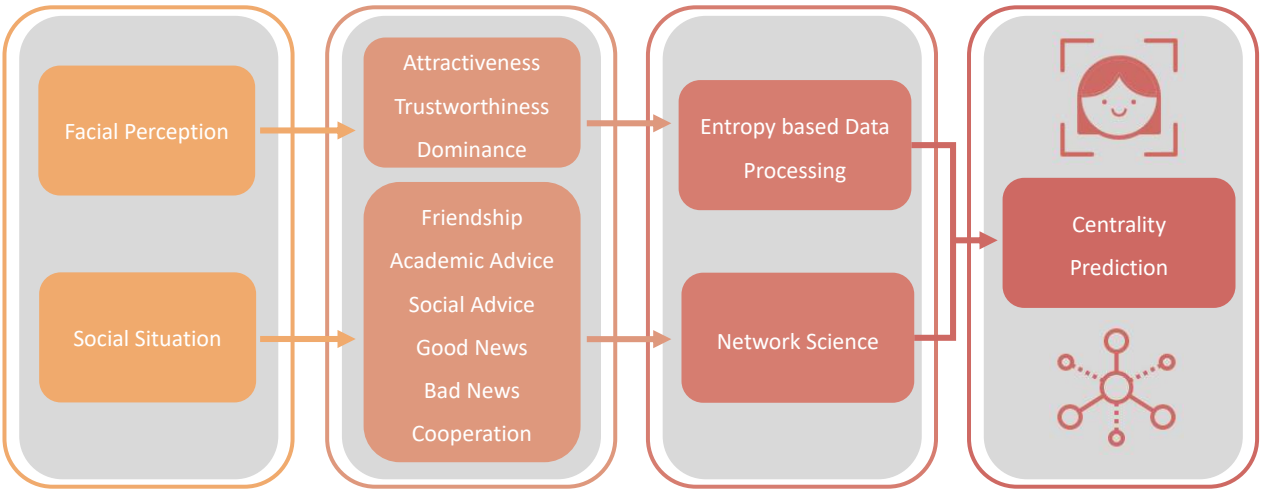


Figure 1: Methodology (PSP framework) used to analyze correlations between facial perception and social situation, and then to predict the centrality of different social networks. First, data of facial perception and social situation are collected from participants. Second, entropy-based data processing method and network science are used to process data. Finally, classification algorithm is used to predict the centrality in social networks according to facial perception features.

credit record [22, 26]. Moreover, Olivola et al. suggested that perceived dominance is correlated with professional success within the military and politics, because people believe that the leaders of these fields tend to look masculine, mature, and âĂIJcoldâĂİ [18]. Therefore, since perceived trustworthiness, attractiveness, and dominance relate to social behavior in social network settings, they may influence network centrality.

2.2 Positions in Social Networks

PeopleâĂŹs positions in social networks are affected by many factors, such as behavior, personality, and emotion. Previous research indicates that behavioral tactics are associated with certain structural positions, such as indegree centrality [4]. A person who is considered assertive and rational, and who has upward appeal, can also use behavioral tactics. These people are always considered dependable and reliable. Therefore, people tend to ask these persons to help them when they are in trouble, or they share good news with them. Thus, people like to follow those who have social power, and to give them central positions in social networks. A personâĂŹs behavior is the reflection of his/her personality. Psychological research suggests that peopleâĂŹs personalities have impacts on their positions in social networks. Feiler et al. [11] found that extraverted people have more friends than introverted people, and that two extraverts are more likely to become friends because of the homophily effect. Therefore, extraverted people are much more likely to be involved in social networks, which places them in central positions. Not only do different personalities have impacts on peopleâĂŹs positions in social networks, but also emotions can change peopleâĂŹs centrality in networks. PeopleâĂŹs emotions also relate to their personalities. Research indicates that people who have positive attitudes to life are attractive in social networks [8]. This is because people with high life satisfaction find it easy to attract their peersâĂŹ attention and alliances [15]. Those who have positive emotions like to experience positive emotions. Therefore, such persons tend to become extraverted. As we all know, outgoing people are much more active in social networks. They are likely to communicate with others and to make their positions more central.

3 DATA COLLECTION

3.1 Privacy Protection

Prior to participation, all participants provided a consent form and consented to take part in the study. They were debriefed after taking part in the experiments. Participants were informed that data gathered during this research would be coded and kept confidentially by the researcher, with only the researcher and supervisor having access; all information is stored with a code, and there is no immediately identifiable information. Part of the research involved taking photographic images. These images are kept secure and stored with no identifying factors, i.e., consent forms and questionnaires.

3.2 Facial Image and Social Network Data Collection

We used both online and offline campus advertisements to recruit college freshman participants who might be interested in this research in the universityâĂŹs freshman residential area. Student participants were required to be more than 18-year-old freshmen, who lived in several specific freshman residential buildings (next to each other) in the same area. A total of 185 participants (aged 18-20, mean = 19.07, SD = 0.18, males=102, female=84) were recruited (89% of the freshman in this area). They were called to the lab, they completed a social network questionnaire, and their photos were taken. This took place at the beginning of third month of freshman

registration of university entrance, and it lasted for one month. Monetary compensation was given for participation in the study.

When participants arrived at the lab, prior to participation, they provided a consent form and consented to take part in the study. Participants first reported their age, gender, country of birth, country of residence, and ethnicity via a questionnaire. They then listed up to eight people on six questions in random order with reference to additional name lists, which gave the names of all freshmen living in this area. Eight people should be listed with the priority from high to low: âĂIJ1âĂİ stands for the highest priority, and priority decreases in turn; âĂIJ8âĂİ stands for the lowest priority. The six questions are as follows: âĂIJPlease list your good friends.âĂİ âĂIJTo whom will you turn for help, when you have difficulties in study?âĂİ âĂIJTo whom will you turn for help, when you encounter some troubles in life?âĂİ âĂIJWhen you get a piece of good news, with whom will you share it?âĂİ âĂIJWhen you get a piece of bad news, with whom will you share it?âĂİ âĂIJWhom do you want to invite to join your team when you have a task that requires a team to accomplish it?âĂİ We also took face photographs of students after they completed the network questionnaire. Participants were instructed to look at the center of the camera lens with neutral expressions, hair pulled back, and no adornment. They were photographed under standardized lighting conditions with a fixed camera distance. They were debriefed after taking part in the experiments. A set of 185 images of 185 identities in six groups (30 or 31 images of each group) were thus available for the study.

3.3 Facial Perception Data Collection

A total of 180 participants (no freshmen) were recruited in the university with payment(aged 18-27, mean = 23.09, SD = 3.06, males=112, female=68). We used different participants from those freshmen to rate images because we focus on the effect of facial bias which frequently made by strangers. They were called to the lab for face rating by appointment. The set of stimuli for rating included 30 identity images (30 of 185 images in each block with no face identities repeated) with randomized allocation for different participants. Each image was rated by 30 participants. Each participant saw 30 images in a random order. Participants completed three ratings: trustworthiness, attractiveness, and dominance. Participants saw one image at a time. For trustworthiness (attractiveness or dominance) ratings, participants were instructed âĂIJYou will make judgements regarding trustworthiness (attractiveness or dominance). Please rate your impression from score 1 to 7 on how trustworthy (attractive or dominant) the face appearsâĂİ; they then rated the stimuli for perceived trustworthiness (attractiveness or dominance) on a seven-point Likert-type scale labelled (1) very negative and (7) very positive, with the question on the top of each facial image being âĂIJhow trustworthy (attractive or dominant) does this face appear?âĂİ For attractiveness rating, participants were given the instruction âĂIJYou will make judgements regarding attractiveness. Please rate your impression of how attractive this face is from 1 to 7.âĂİ Participants rated apparent attractiveness on this seven-point scale.

4 METHODS

To solve this kind of problem that explores the relation between facial perception and social network situation, we proposed a framework named PSP (Facial Perception and Social Situation Prediction Framework) (shown in Figure 1). In this framework, we proposed an entropy-based data processing method to remove the noise in questionnaire data. Each part included in the framework is introduced as follows.

4.1 Problem Formulation

We assumed that there is a photo of student i and this photo is evaluated by thirty people from three aspects: attractiveness, trustworthiness and dominance. In other word, the photo of student i will get three scores $[s_{i1}, s_{i2}, s_{i3}]$. In addition, we calculated the degree centrality of student i in six different social networks$[d_{i1}, d_{i2}, d_{i3}, d_{i4}, d_{i5}, d_{i6}]$. Then, we compute the correlation between photo scores and degree centrality of each network and explore the predictability of degree centrality using the photo scores.

4.2 Network Analysis

We carried out a detailed analysis of six social networks through several parameters as follows, in order to get a deep understanding to sample's social situation.

4.2.1 Degree correlation and assortativity coefficient. The degree distribution, which is the probability that there are nodes in a network with the degree of i, is defined as

$$P(i) = \frac{n(i)}{N}, \tag{1}$$

where $n(i)$ is the number of nodes with degree i and N is the total numbers of nodes in the network. The joint probability distribution of the network is defined as the proportion of edges between nodes with degree i and j:

$$P(i,j) = \frac{m(i,j)\mu(i,j)}{2M}, \tag{2}$$

where $m(i,j)$ is the number of edges between nodes with degree i and j and M represents the total number of edges. If $i = j$, then $\mu(i,j) = 2$, otherwise $\mu(i,j) = 1$. Then, the excess degree distribution is defined as

$$P_n(i) = \sum_{i=j_{min}}^{j_{max}} P(i,j), \tag{3}$$

where j_{min} and j_{max} are the minimum and maximum of nodes' degree in the network. The excess degree distribution indicates the probability that of choosing a node with degree i from the neighbors of a node chosen randomly.

Denote the degree distribution $P(i)$ as p_i, the joint probability distribution $P(i,j)$ as e_{ij}, and the excess degree distribution $P_n(i)$ as q_i, then the assortativity coefficient is defined as

$$r = \frac{1}{\sigma_q^2} \sum_{i,j} ij(e_{ij} - q_i q_j), \tag{4}$$

where $\sigma_q^2 = \sum_i i^2 q_i^2 - [\sum_i i q_i]^2$ is the variance of the excess degree distribution q_i. The assortativity coefficient r ranges from -1 to 1. If $r > 0$, then the network is assortative, which means that nodes with bigger degree prefer to connect with nodes with bigger degree,

and vice versa. The absolute value of r reflects the strength of assortativity or disassortativity of the network.

We can also use excess average degree to quantify the degree correlation of network approximately. The excess average degree neighbors is defined as

$$\langle k_{nn} \rangle_i = \frac{1}{k_i} \sum_{j=1}^{k_i} k_{i_j}, \tag{5}$$

where $k_i k_{i_j}$ is the degree of number i_jth neighbor of node i.

4.2.2 Clustering coefficient. The clustering coefficient quantifies the probability that the neighbors of node i are connected with each other. It is a frequently used metric for the community structure of a network. The clustering coefficient of a node i with degree of k_i is defined as

$$C_i = \frac{E_i}{(k_i(k_i-1))/2} = \frac{2E_i}{k_i(k_i-1)}, \tag{6}$$

where E_i is the numbers of edges existing among the k_i neighbors of node i.

4.3 Correlation Coefficient

PearsonâĂŹs correlation is used to explore the relation among facial perception features and spearman's correlation is used to explore the relation between facial perception feature and the centrality of six social networks because of the different scale.

4.3.1 SpearmanâĂŹs correlation. SpearmanâĂŹs correlation, also known as SpearmanâĂŹs rank correlation, measures the correlation of rank between two populations. Here, we use it in the analysis of the relationships between the scores of photos and their centralities in different social networks. The SpearmanâĂŹs rank correlation coefficient is defined as

$$r_S = 1 - \frac{6 \sum_{i=1}^{N} d_i^2}{N(N^2-1)}, \tag{7}$$

where N is the number of students in our experiment, and $d_i = r(S_i) - r(C_i)$, with $r(S_i)$ and $r(C_i)$ being the ranks of students' photo scores and centralities respectively.

4.3.2 PearsonâĂŹs correlation. PearsonâĂŹs correlation coefficient quantifies the linear correlation between two groups of continuous variables. The Pearson correlation coefficient is defined as

$$r_{X,Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}, \tag{8}$$

where μ and σ are the mean value and standard deviation of random variable X and Y, and $E(\cdot)$ is the expectation. Here, we use the Pearson correlation coefficient to analyze the linear relationships among three kinds of scores of studentsâĂŹ photos.

4.4 Prediction

First, we proposed a novel method that using information entropy to remove the noise in questionnaire data. Then, we redesigned our prediction task as binary classification task and used four classical classification algorithms to predict student centrality.

4.4.1 Feature Processing. As mentioned before, each photo was evaluated by a certain number of participants. Some participants evaluated all photos with same score or score with low separating capacity, because they didnâĂŹt take these questionnaires seriously. As the noise, these scores will impact experiment result. Here we borrowed the concept of entropy in information theory to remove these meaningless data. We proposed a participant entropy which is calculated according to the score they gave to each photo.

$$E = -\sum p(x) \log p(x) \tag{9}$$

where $p(x)$ represents the probability of occurrence of sample x.

4.4.2 Prediction Models. In the experiment, we used four classical classification methods to solve the prediction problem. First, we tried the k-nearest neighbor (KNN) classifier [6], which finds the top k nearest neighbors of a test sample in the train set on the feature space and makes predictions by vote. The idea of this method is very naive, namely that if samples are closer in the feature space, they are more likely to be in the same class.

Then, we observed two linear classification modelsâĂŹ performance, linear support vector classifier (linear SVC) [3, 17] and logistic regression (LR) [7]. Both identify hyperplanes in the feature space to split samples into different classes. However, these two algorithms obtain the best partition in different ways. The linear SVC looks for a hyperplane that maximizes the interval between two classes, that is,

$$\begin{cases} max_{\mathbf{w},b} & \frac{2}{\|\mathbf{w}\|} \\ s.t. & y_i(\mathbf{w}^T x_i) \geq 1, \quad i = 1, 2, 3 \cdots, \end{cases} \tag{10}$$

where y_i and x_i are labels and features of ith sample in the train set, and $\mathbf{w}$ is the normal vector of the hyperplane.

Logistic regression estimates the relative logarithm odds of a sample as a positive one, that is

$$\mathbf{w}^T \mathbf{x} + b = ln \frac{y}{1-y}, \tag{11}$$

where y is the odds of sample being a positive one, $\mathbf{x}$ is its feature vector, and $\mathbf{w}$ is its corresponding coefficient.

We also used a probability model, a Naive Bayesian classifier, which is based on the attribute conditional independence assumption,

$$P(c|\mathbf{x}) = \frac{P(c)P(\mathbf{x}|c)}{P(\mathbf{x})} = \frac{P(c)}{P(\mathbf{x})} \prod_{i=1}^{d} P(\mathbf{x}_i|c), \tag{12}$$

where $P(c|\mathbf{x})$ is the probability of a sample belonging to class c under the condition that its feature vector equals $\mathbf{x}$. The Naive Bayesian classifier involves a prior assumption of the sample distribution. Here we used the Gaussian distribution due to the distribution of data.

5 RESULTS

5.1 Facial Perception Analysis

To explore the relationship between the centrality of social networks and perception, photos of participants were measured in three dimensions: attractiveness, trustworthiness, and dominance, and the distributions of data of each dimension are shown in Figure 2. The average score for trustworthiness was higher than the scores

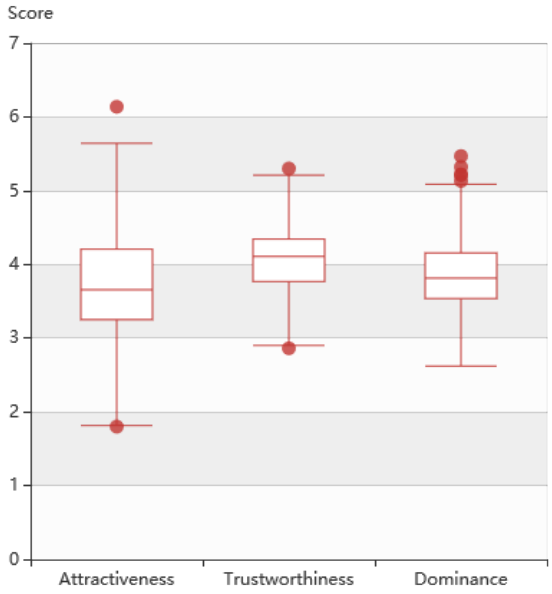


Figure 2: Box plot of attractiveness, trustworthiness, and dominance.

Table 1: Pearson's correlation coefficient

Score	Attractiveness	Trustworthiness	Dominance
Attractiveness	-	-	-
Trustworthiness	0.410**	-	-
Dominance	0.379**	0.179*	-

*p<0.05; **p<0.01

for other two aspects, and attractiveness scored lowest, which suggests that overall the freshmen prefer to trust strangers. For the dispersion level, the score for attractiveness was spread out further from its average value than others, as the span and interquartile range of attractiveness are much longer than the other two dimensions. For extreme outliers, compared with attractiveness and trustworthiness, participants give more extreme outliers on dominance. The distribution of data on these three dimensions has no significant skewness. In a word, Figure 2 suggests that students involved in our dataset covered almost all levels of facial appearance. Moreover, the correlations among attractiveness, trustworthiness, and dominance are shown in Table 1. Attractiveness has a relatively strong positive correlation with trustworthiness and dominance. The correlation between trustworthiness and dominance is also positive, but relatively weak. This means that people with high attractiveness can be trusted easily, and they can also be deemed to dominate.

5.2 Social Network Analysis

We built six directed networks defined by different types of relationships from our questionnaire. The nodes are all connected

Table 2: Correlation between three observed features and in-degree in six social networks

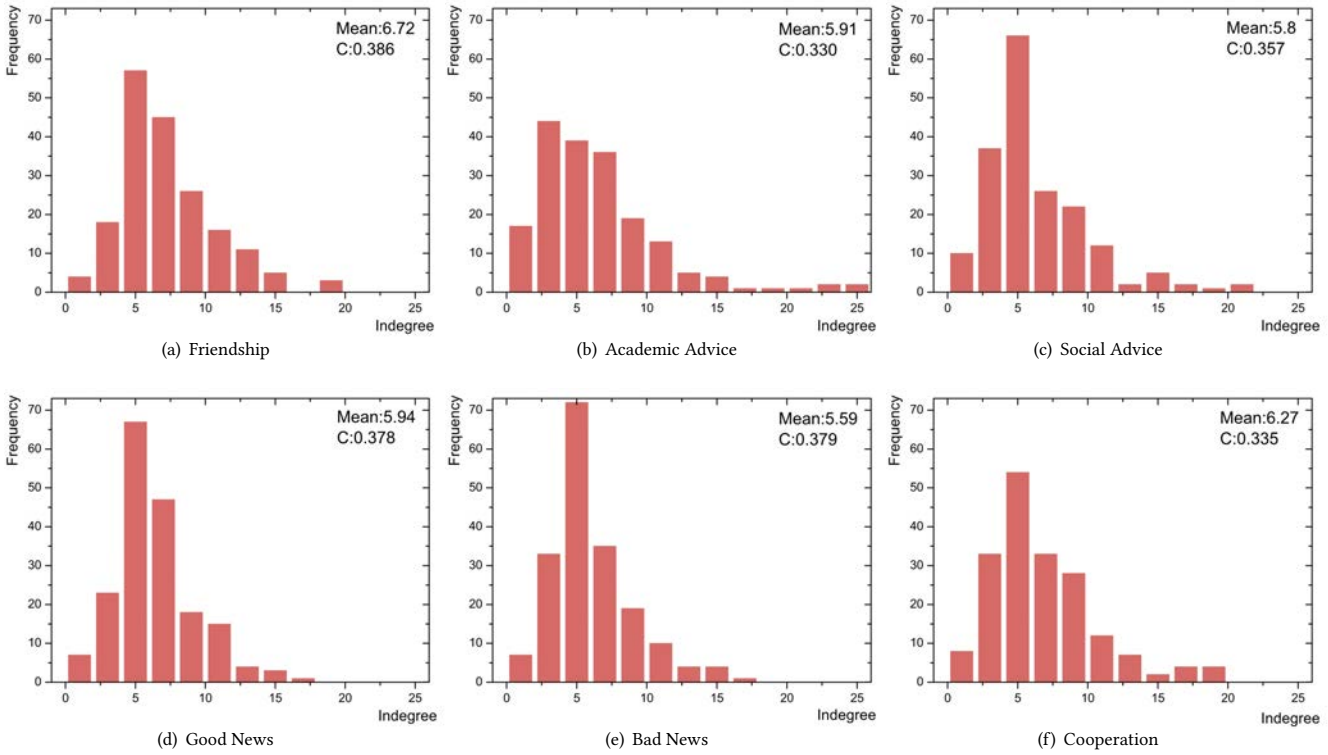
	Bad News	Friendship	Good News
Attractive	0.107	0.131	0.157*
Dominant	0.105	0.114	0.069
Trustworthy	0.152*	0.152*	0.143
	Social Advice	Academic Advice	Cooperation
Attractive	0.142	-0.055	0.084
Dominant	0.122	0.031	0.161*
Trustworthy	0.129	0.078	0.150*

* p<0.05

in these six directed networks. As we can see from Figure 3, the distributions of indegree for each of the six networks look like left-skewed bell-shaped curves. Due to the requirements of each questionnaire, most of the average indegree of the networks were about 6, except for the friendship network, with an average indegree of 6.72, which suggests that different friends meet different social requirement. For example, friends who are shopping together may not be suitable for starting a business together. However, in the bad news network, the average indegree is the lowest, and in most cases the indegree is around 5. This observation is consistent with general daily experience that bad news can only be shared with close friends. Moreover, about 75% of the indegrees are between 3 and 8, and the most frequent indegree is 5. Only a few individuals have a big indegree, which means that others prefer to choose them as listeners or helpers. In other words, few individuals occupy a center position in the social networks. This verifies the view that the Matthew Effect exists in social networks. Furthermore, four networksâĂŹ clustering coefficients are around 0.37, while they are about 0.33 in the cooperation and academic advice networks. Compared to some typical social networks, all these clustering coefficients are lower. This is probably because the individuals we researched are freshmen, and their social networks are still at an early stage. Their circles of friend are still narrow, so some friends may not know each other. As time goes by, more commonality will develop between students, and increasing links will appear, resulting in a higher clustering coefficient.

In Figure 4 we visualized the joint probability distribution of the indegree in all six networks. Note that nodes in all networks with a median indegree (with 8 to 10 nodes pointing to them) can have connections with higher probabilities (but no higher than 0.05, which is relatively very small), as there are more students in the social networks attracting 8 to 10 friends. We can also see this result from the indegree distribution and the excess degree distribution. Additionally, we computed the assortativity coefficient in these six networks, and we noticed that the values are all very low (less than 0.2). This result indicates that there are no obvious correlations between the connections of the indegree of node pairs.

Figure 5 shows the correlation between indegree and excess average degree, which is the average degree of their neighbors. There is an obvious negative correlation in all these six networks. This



(a) Friendship (b) Academic Advice (c) Social Advice

(d) Good News (e) Bad News (f) Cooperation

Figure 3: Histograms of indegree for the six social networks. In each figure, The 'Mean' and 'C' represent the average degree and clustering coefficient.

indicates that students prefer to choose people who are different than themselves.

5.3 Correlation Analysis

To explore the relations between facial appearance and social situation, we performed an analysis on the correlation between the three observed features and the indegrees in six social networks. All findings are shown in Table 2.

The results show that trustworthiness is related to the indegrees in the bad news and friends networks, as shown in Figure 6 (both with a correlation coefficient of 0.152 and $p < 0.05$). For the choice of people as good friend, trustworthiness is more important than attractiveness and dominance (shown in Figure 7), which confirms a saying by Epicurus: without confidence, there is no friendship. People prefer to share bad news with people who get high scores in trustworthiness, which is consistent with common sense in that people would far rather show their bad side to trustworthy people than those with beautiful facial appearances. This also suggests that part of the role of a friend is to share pain.

When it comes to the good news spreading network, on the contrary, studentsâĂŹ attractiveness shows a significant correlation with indegree (with a correlation coefficient of 0.157 and $p-value < 0.05$). This means that the people with whom one can share bad news are not the same people with whom one can share good news. The purpose of sharing bad news is to share the pain and to get comfort. However, the purpose of sharing good news is not just sharing happiness, but also showing oneâĂŹs good side to the person with good facial appearance, possibly to reduce the psychological gap caused by facial appearance. [1]

Compared with the scenarios of choosing friends, sharing good news and sharing bad news, the following three scenarios are relatively rational. In the cooperation network, both dominance and trustworthiness are significant, which indicates that while people want to build a team, the one who shows enough dominant force and looks more trustworthy is more favorable. It is the only time that participants take dominance into account. In psychology, dominance is a measure of a personâĂŹs characteristic feelings of control and influence over his/her life circumstances versus feelings of being controlled and influenced by others or events. If attractiveness represents external competitiveness, then dominance represents inherent competitiveness, giving people confidence. In such a rational sense with a clear goal to complete a task, it suggests than students prefer to choose someone with inherent competitiveness over someone with a beautiful appearance. For social advice and academic advice, facial appearance does not have any effect statistically. In these two scenarios, the situation is so complicated, including making decisions on various questions, that it is difficult to make a decision solely according to facial appearance.

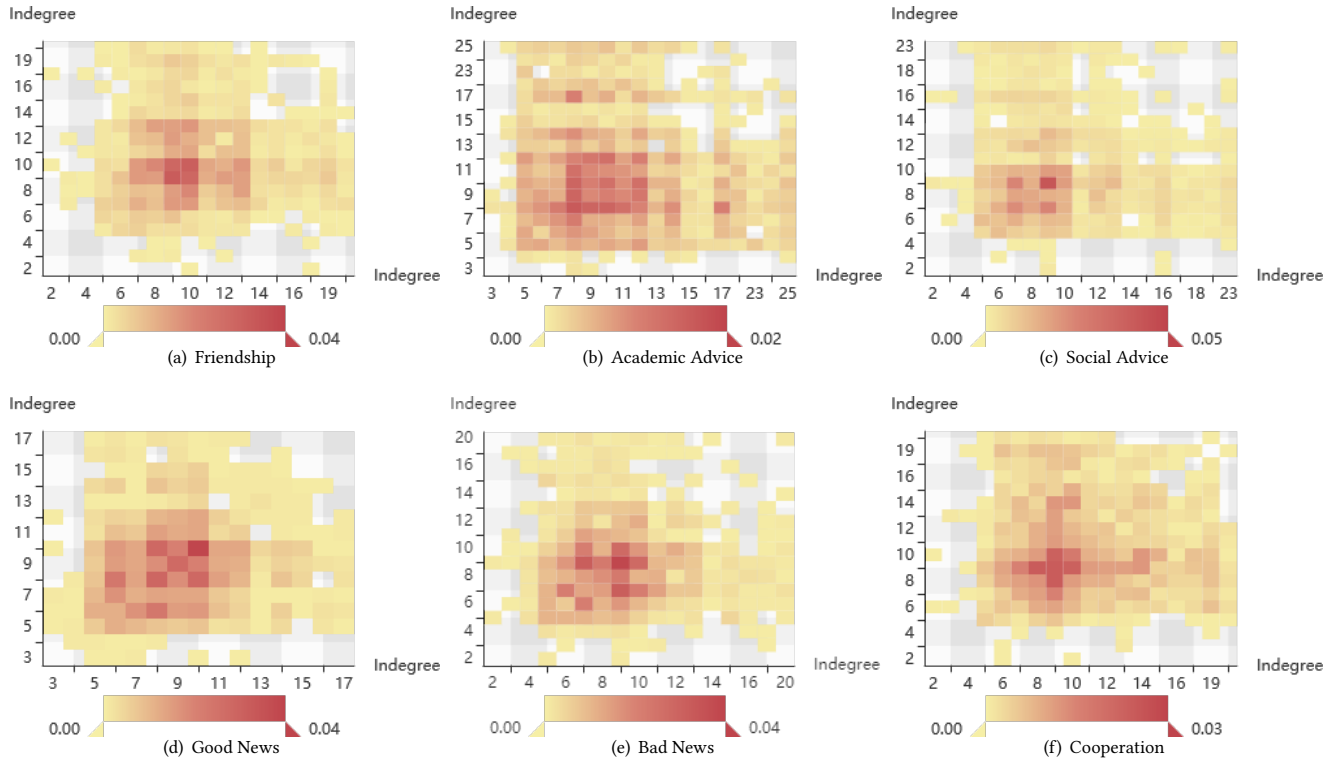


Figure 4: Joint probability distribution for the six social networks.

In addition, these findings also suggest that the main factors taken into account in various social scenarios differ. Finally, compared with attractive and dominance, trustworthiness is more important for peopleâĂŹs social lives, because three of these six social scenarios involve trust. In other words, being a trustworthy person will give you a good chance of being in a favorable social situation.

5.4 Prediction

The significant correlations imply that the three features extracted from facial appearance can predict studentsâĂŹ centrality in different social networks. To explore their predictive power, we carried out an experiment using the score of three aspects from studentâĂŹs facial appearance to predict their centrality in six different social networks. Excluding some incomplete or error data records, we got 185 samples in the experiment dataset. All three facial appearance quantification scores (attractiveness, dominance, and trustworthy) featured in the six observed networks. As every photo was quantified by 30 participants, we computed the average score from the final points. We performed our prediction task as a binary classification problem, which predicted whether a node could be high degree (with indegree higher than the median of population) or not.

We divided the dataset into two categories by degree of centrality: high and low. We used machine learning to predict the categories of the participants from their scores on the three dimensions. First, we removed 30 participants randomly from the dataset by stratified sampling to use as a test set, and we used the rest of the data as a train set. We performed a fivefold cross-validation on the train set to tune the super parameters of the predictive models, k-nearest neighbors, linear support vector classifier, naive Bayes classifier, and logistic regression, which are long-standing classical binary classification algorithms.

We calculated the participant entropy of each participant according to the score they gave to each photo. We assumed that there is a positive correlation between participant entropy and the validity of data given by participant. The higher the entropy gets, the more valid the data is. In this study, we removed 20%, 30% and 40% low entropy data respectively to remove the noise existed in data and the performances are shown in Figure 8.

Because the proportions of the positive and the negative are equal in the dataset, we used the accuracy value to evaluate the result of prediction. The accuracy value ranges from 0 to 1, with 0.5 being random chance in this condition; therefore, the predictive power is the extent to which the accuracy value exceeds 0.5.

Prediction results are shown in Figure 8. Obviously, it is notable that three features extracted from facial appearance are effective for predicting the centrality of four different networks (friendship network, bad news network, good news network and cooperation network) because all accuracy values exceed 0.5. The prediction result of removing 30% low-entropy data is the best. The result of removing 20% data is lower possibly because noise still exist in data, while the result of removing 40% data is lower possibly because

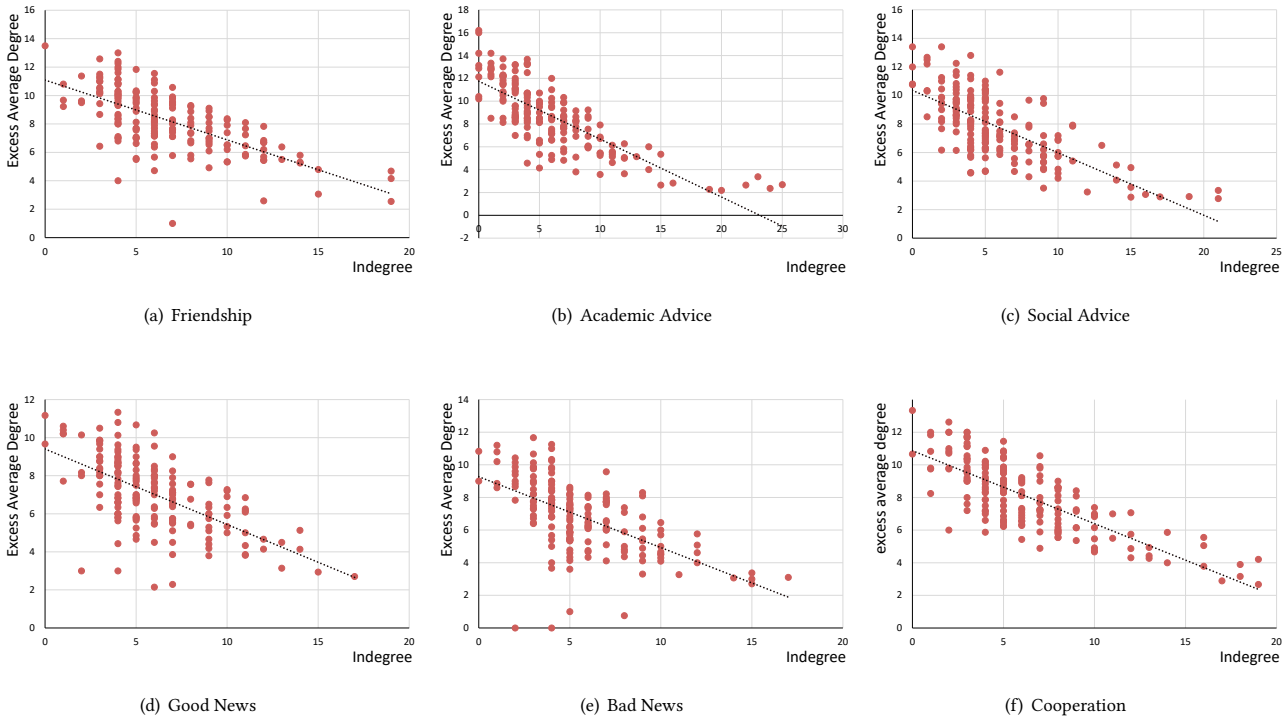


Figure 5: The correlation between indegree and excess average degree for each network.

some meaningful data is lost. In addition, accuracies of prediction in the social advice network and the academic advice network are relatively low, which is consistent with the previous findings about their correlation coefficients.

6 CONCLUSIONS

In this paper, we collected face perception data from freshmen in college to explore how appearance could impact the individualsâĂŹ

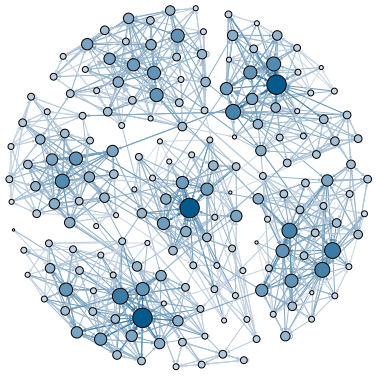


Figure 6: Sketch of the friendship network. The node represents student and the diameter of the nodes in the network indicates the trustworthiness score given by others.

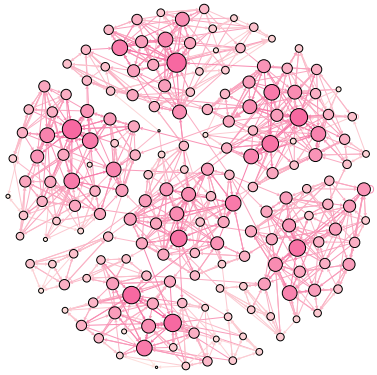
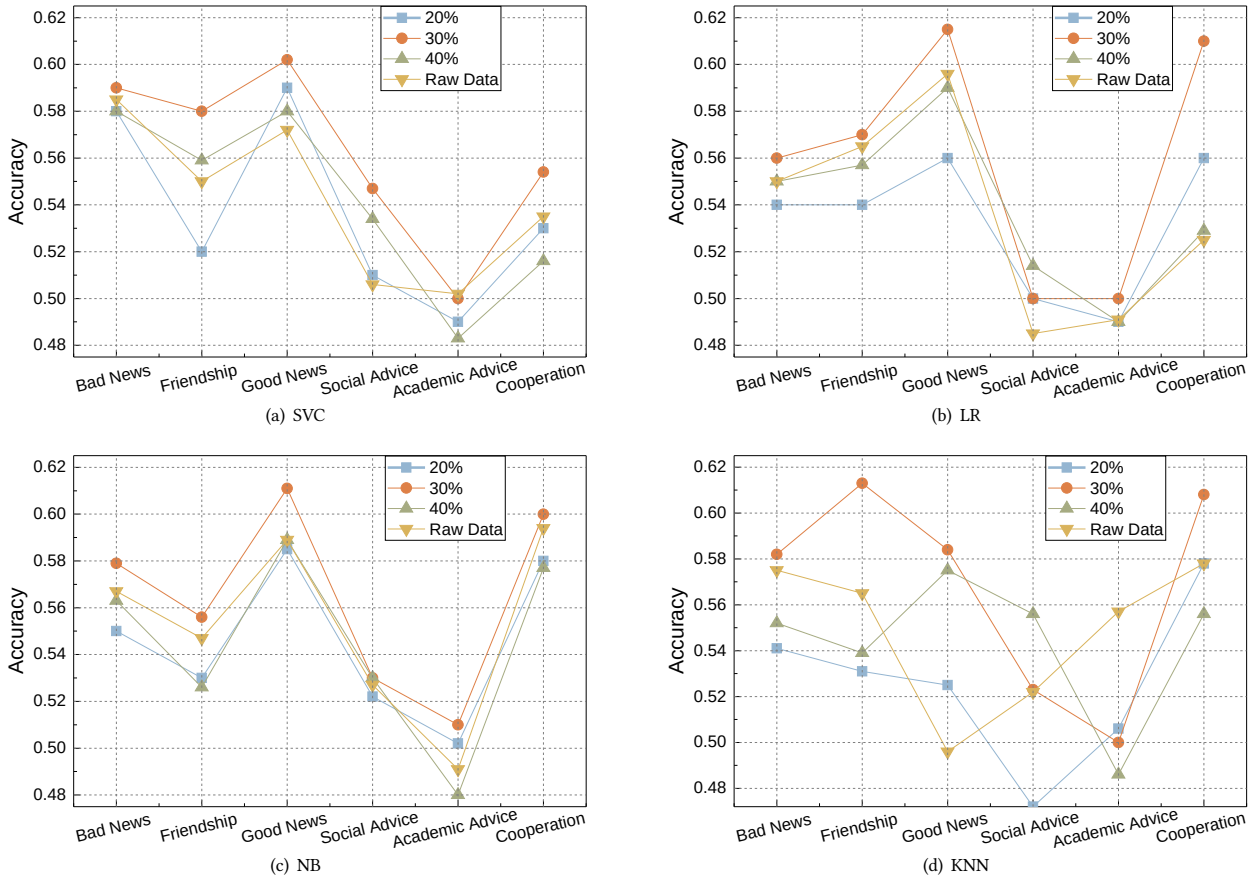


Figure 7: Sketch of the good news network. The node represents student and the diameter of the nodes in the network indicates the attractiveness score given by others.

centrality in different social networks. We selected six questions to capture their social relations in different scenarios, namely a friendship network, a good news network, a bad news network, a social advice network, an academic advice network, and a cooperation network. Experimental results show that perceived facial traits can predict the centrality of different networks. We proposed a PSP framework to discover the effect of facial perception on social situation and the experimental results verify its performance. Our



(a) SVC (b) LR (c) NB (d) KNN

Figure 8: Prediction performance of four classical classification algorithms. Lines of various colours represent the different percentage that we remove the low-entropy data from the raw data.

framework mechanism has a certain universality, which can be applied to the prediction of centrality in different social networks by reducing noise to improve the prediction accuracy. Our findings suggest that peopleâĂŹs appearance can impact their social situations at an early social stage in which people are not very familiar with each other. To the best of our knowledge, we are the first to explore the influence of facial appearance on centrality in social networks.

Our results also contribute to the psychological knowledge of facial perception, because they are consistent with previous research that facial appearance affects network-based social behaviors. Prior studies only examined the relationship between faces and behaviors in social network settings without using social network analysis and techniques, whose analysis might be limited and might lack depth. This paper sheds light on a novel approach by showing that apparent facial features can be evaluated and predicted by social network analysis and machine learning. In addition, our findings offer insights on a combination of psychological and social network techniques, and they highlight the function of facial bias in cueing and signaling social traits.

Our research is limited by the relatively small sample size, because obtaining a larger number of participants would be very challenging. The privacy and confidentiality of human participants was very important in this research, in that participantsâĂŹ photographs were taken and judged, and their social networks were reported. Accessing a large number of volunteers who are willing to participate such research is a challenge. However, a thorough analysis of social network from a novel, face-based perspective in this paper may remedy the limitations to some extent. Future work could obtaining a larger dataset to facilitate a more in-depth investigation.

ACKNOWLEDGMENTS

This work is partially supported by National Natural Science Foundation of China under Grant Nos. 61602079, 61632011, and 61872054, and by Ministry of Education under Grant No. 16YJCZH141.

REFERENCES

[1] Joel Birman and Cristina Cernat. 2018. Clinical figures of ugliness up against metapsychology. *Estudos em Teoria Psicanal* 21, 1 (2018), 1–11.
[2] Peter Borkenau, Steffi Brecke, Christine Möttig, and Marko Paelecke. 2009. Extraversion is accurately perceived after a 50-ms exposure to a face. *Journal of*

Research in Personality 43, 4 (2009), 703–706.

[3] Léon Bottou, Frank E Curtis, and Jorge Nocedal. 2018. Optimization methods for large-scale machine learning. *SIAM Rev.* 60, 2 (2018), 223–311.

[4] Daniel J Brass and Marlene E Burkhardt. 1993. Potential power and power use: An investigation of structure and behavior. *Academy of Management Journal* 36, 3 (1993), 441–470.

[5] Alison Wood Brooks, Laura Huang, Sarah Wood Kearney, and Fiona E Murray. 2014. Investors prefer entrepreneurial ventures pitched by attractive men. *Proceedings of the National Academy of Sciences* 111, 12 (2014), 4427–4431.

[6] Matias D Cattaneo, Max H Farrell, and Yingjie Feng. 2018. Large Sample Properties of Partitioning-Based Series Estimators. *arXiv preprint arXiv:1804.04916* (2018).

[7] Hua Chai, Yong Liang, Sai Wang, and Hai-wei Shen. 2018. A novel logistic regression model combining semi-supervised learning and active learning for disease classification. *Scientific reports* 8, 1 (2018), 13009.

[8] Chia-Yen Chad Chiu, Prasad Balkundi, and Frankie Jason Weinberg. 2017. When managers become leaders: The role of manager network centralities, social power, and followers' perception of leadership. *The Leadership Quarterly* 28, 2 (2017), 334–348.

[9] Emily J Cogsdill, Alexander T Todorov, Elizabeth S Spelke, and Mahzarin R Banaji. 2014. Inferring character from faces: A developmental study. *Psychological science* 25, 5 (2014), 1132–1139.

[10] Louise Ewing, Frances Caulfield, Ainsley Read, and Gillian Rhodes. 2015. Perceived trustworthiness of faces drives trust behaviour in children. *Developmental Science* 18, 2 (2015), 327–334.

[11] D.C. Feiler and A.M. Kleinbaum. 2015. Popularity, similarity, and the network extraversion bias. *Psychological Science* 26, 5 (2015), 593–603.

[12] Xiangjie Kong, Kai Ma, Shen Hou, Di Shang, and Feng Xia. 2019. Human Interactive Behavior: A Bibliographic Review. *IEEE Access* 7 (2019), 4611–4628.

[13] Anthony C Little, D Michael Burt, and David I Perrett. 2006. What is good is beautiful: Face preference reflects desired personality. *Personality and Individual Differences* 41, 6 (2006), 1107–1118.

[14] Jiaying Liu, Xiangjie Kong, Feng Xia, Xiaomei Bai, Lei Wang, Qing Qing, and Ivan Lee. 2018. Artificial intelligence in the 21st century. *IEEE Access* 6 (2018), 34403–34421.

[15] Sonja Lyubomirsky, Laura King, and Ed Diener. 2005. The benefits of frequent positive affect: Does happiness lead to success? *Psychological Bulletin* 131, 6 (2005), 803–855.

[16] Sylvia A Morelli, Desmond C Ong, Rucha Makati, Matthew O Jackson, and Jamil Zaki. 2017. Empathy and well-being correlate with centrality in different social networks. *Proceedings of the National Academy of Sciences* 114, 37 (2017), 9843–9847.

[17] Juan Antonio Morente-Molinera, József Mezei, Christer Carlsson, and Enrique Herrera-Viedma. 2017. Improving supervised learning classification methods using multi-granular linguistic modelling and fuzzy entropy. *IEEE transactions on fuzzy systems* 25, 5 (2017), 1078–1089.

[18] Christopher Y Olivola, Dawn L Eubanks, and Jeffrey B Lovelace. 2014. The many (distinctive) faces of leadership: Inferring leadership domain from facial appearance. *Leadership Quarterly* 25, 5 (2014), 817–834.

[19] Nikolaas N Oosterhof and Alexander Todorov. 2008. The functional basis of face evaluation. *Proceedings of the National Academy of Sciences* 105, 32 (2008), 11087–11092.

[20] Stephen Porter, Laura England, Marcus Juodis, Leanne ten Brinke, and Kevin Wilson. 2008. Is the face a window to the soul? Investigation of the accuracy of intuitive judgments of the trustworthiness of human faces. *Canadian Journal of Behavioural Science/Revue canadienne des sciences du comportement* 40, 3 (2008), 171.

[21] Stephen Porter, Leanne ten Brinke, and Chantal Gustaw. 2010. Dangerous decisions: The impact of first impressions of trustworthiness on the evaluation of legal evidence and defendant culpability. *Psychology, Crime & Law* 16, 6 (2010), 477–491.

[22] Constantin Rezlescu, Brad Duchaine, Christopher Y Olivola, and Nick Chater. 2012. Unfakeable facial configurations affect strategic choices in trust games with or without information about past behavior. *PloS one* 7, 3 (2012), e34293.

[23] Nicholas O Rule, Anne C Krendl, Zorana Ivcevic, and Nalini Ambady. 2013. Accuracy and consensus in judgments of trustworthiness from faces: Behavioral and neural correlates. *Journal of Personality and Social Psychology* 104, 3 (2013), 409.

[24] Michael Stirrat and David I Perrett. 2010. Valid facial cues to cooperation and trust: Male facial width and trustworthiness. *Psychological science* 21, 3 (2010), 349–354.

[25] Clare AM Sutherland, Julian A Oldmeadow, Isabel M Santos, John Towler, D Michael Burt, and Andrew W Young. 2013. Social inferences from faces: Ambient images generate a three-dimensional model. *Cognition* 127, 1 (2013), 105–118.

[26] Dustin Tingley. 2014. Face-off: Facial features and strategic choice. *Political Psychology* 35, 1 (2014), 35–55.

[27] Alexander Todorov. 2008. Evaluating faces on trustworthiness. *Annals of the New York Academy of Sciences* 1124, 1 (2008), 208–224.

[28] Alexander Todorov, Anesu N Mandisodza, Amir Goren, and Crystal C Hall. 2005. Inferences of competence from faces predict election outcomes. *Science* 308, 5728 (2005), 1623–1626.

[29] Alexander Todorov, Manish Pakrashi, and Nikolaas N Oosterhof. 2009. Evaluating faces on trustworthiness after minimal time exposure. *Social Cognition* 27, 6 (2009), 813–833.

[30] Jan Verplaetse, Sven Vanneste, and Johan Braeckman. 2007. You can judge a book by its cover: the sequel.: A kernel of truth in predictive cheating detection. *Evolution and Human Behavior* 28, 4 (2007), 260–271.

[31] Janine Willis and Alexander Todorov. 2006. First impressions: Making up your mind after a 100-ms exposure to a face. *Psychological science* 17, 7 (2006), 592–598.

[32] John Paul Wilson and Nicholas O Rule. 2015. Facial trustworthiness predicts extreme criminal-sentencing outcomes. *Psychological Science* 26, 8 (2015), 1325–1331.

[33] Leslie A Zebrowitz, Luminita Voinescu, and Mary Ann Collins. 1996. " Wide-Eyed" and" Crooked-Faced": Determinants of perceived and real honesty across the life span. *Personality and social psychology bulletin* 22, 12 (1996), 1258–1269.

Pruning based Distance Sketches with Provable Guarantees on Random Graphs

Hongyang Zhang
Stanford University
hongyang@cs.stanford.edu

Huacheng Yu
Harvard University
yuhch123@gmail.com

Ashish Goel
Stanford University
ashishg@stanford.edu

ABSTRACT

Measuring the distances between vertices on graphs is one of the most fundamental components in network analysis. Since finding shortest paths requires traversing the graph, it is challenging to obtain distance information on large graphs very quickly. In this work, we present a preprocessing algorithm that is able to create landmark based distance sketches efficiently, with strong theoretical guarantees. When evaluated on a diverse set of social and information networks, our algorithm significantly improves over existing approaches by reducing the number of landmarks stored, preprocessing time, or stretch of the estimated distances.

On Erdos-Renyi graphs and random power law graphs with degree distribution exponent $2 < \beta < 3$, our algorithm outputs an exact distance data structure with space between $\Theta(n^{5/4})$ and $\Theta(n^{3/2})$ depending on the value of β, where n is the number of vertices. We complement the algorithm with tight lower bounds for Erdos-Renyi graphs and the case when β is close to two.

ACM Reference Format:
Hongyang Zhang, Huacheng Yu, and Ashish Goel. 2019. Pruning based Distance Sketches with Provable Guarantees on Random Graphs. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313708

1 INTRODUCTION

Computing shortest path distances on large graphs is a fundamental problem in computer science and has been the subject of much study [27, 36, 37]. In many applications, it is important to compute the shortest path distance between two given nodes, i.e. to answer shortest path queries, in real time. Graph distances measure the closeness or similarity of vertices and are often used as one of the most basic metric in network analysis [29, 35, 39, 40]. In this paper, we will focus on efficient and practical implementations of shortest path queries in classes of graphs that are relevant to web search, social networks, and collaboration networks etc. For such graphs, one commonly used technique is that of *landmark-based labelings*: every node is assigned a set of landmarks, and the distance between two nodes in computed only via their common landmarks. If the set of landmarks can be easily computed, and is small, then we obtain both efficient pre-processing and small query time.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313708

Landmark based labelings (and their more general counterpart, *Distance Labelings*), have been studied extensively [9, 36]. In particular, a sequence of results culminating in the work of Thorup and Zwick [37] showed that labeling schemes can provide a multiplicative 3-approximation to the shortest path distance between any two nodes, while having an overhead of $O(\sqrt{n})$ storage per node on average in the graph (we use the standard notation that a graph has n nodes and m edges). In the worst case, there is no distance labeling scheme that always uses sub-quadratic amount of space and provides exact shortest paths. Even for graphs with maximum degree 3, it is known that any distance labeling scheme requires $\Omega(n^{3/2})$ total space [26]. In sharp contrast to these theoretical results, there is ample empirical evidence that very efficient distance labeling schemes exist in real world graphs that can achieve much better approximations. For example, Akiba et al. [3] and Delling et al. [20] show that current algorithms can find *landmark based labelings* that use only a few hundred landmarks per vertex to obtain *exact distance*, in a wide collection of social, Web, and computer networks with millions of vertices. In this paper, we make substantial progress towards closing the gap between theoretical and observed performance. We show that natural landmark based labeling schemes can give exact shortest path distances with a small number of landmarks for a popular model of (unweighted and undirected) web and social graphs, namely the heavy-tailed random graph model. We also formally show how further reduction in the number of landmarks can be obtained if we are willing to tolerate an additive error of one or two hops, in contrast to the multiplicative 3-approximation for general graphs. Finally, we present practical versions of our algorithms that result in substantial performance improvements on many real-world graphs.

In addition to being simple to implement, landmark based shortest path algorithms also offer a qualitative benefit, in that they can directly be used as the basis of a social search algorithm. In social search [8], we assume there is a collection of keywords associated with every node, and we need to answer queries of the following form: given node v and keyword w, find the node that is closest to v among all nodes that have the keyword w associated with them. This requires an index size that is $O(L)$ times the size of the total social search corpus and a query time of $O(L)$, where L is the number of landmarks per node in the underlying landmark based algorithm; the approximation guarantee for the social search problem is the same as that of the landmark based algorithm. Thus, our results lead to both provable and practical improvements to the social search problem.

Existing models for social and information networks build on random graphs with some specified degree distribution [15, 22, 38], and there is considerable evidence that real-world graphs have power-law degree distributions [16, 23]. We will use the Chung-Lu

model [14], which assumes that the degree sequence of our graph is given, and then draws a "uniform" sample from graphs that have the same or very similar degree sequences. In particular, we will study the following question: *Given a random graph from the Chung-Lu model with a power law degree distribution of exponent β, how much storage does a landmark-based scheme require overall, in order to answer distance queries with no distortion?*

In the rest of the paper, we use the term "random power law graph" to refer to a graph that is sampled from the Chung-Lu model, where the weight (equivalently, the expected degree) of each vertex is independently drawn from a power law distribution with exponent β. We are interested in the regime when $\beta > 2$ — this covers most of the empirical power law degree distributions that people have observed on social and information networks [16]. Admittedly, real-world graphs have additional structure in addition to having power-law degree distributions [34], and hence, we have also validated the effectiveness of our algorithm on real graphs.

1.1 Our Results

Our first result corresponds to the "easy regime", where the degree distribution has finite variance ($\beta > 3$). We show that a simple procedure for generating landmarks guarantees exact shortest paths, while only requiring each node to store $\tilde{O}(\sqrt{n})$ landmarks. The same conclusion also applies to Erdős-Renyi graphs $G(n, \frac{c}{n})$ when $c > 1$, or when $c = 2 \log n$.

We then study the case where $2 < \beta < 3$. This is the most emblematic regime for power-law graphs, since the degree distribution has infinite variance but finite expectation. We present an algorithm that generates at most $\tilde{O}(n^{(\beta-2)/(\beta-1)})$ landmarks per node when $\beta \geq 2.5$; and $\tilde{O}(n^{(3-\beta)/(4-\beta)})$ landmarks per node when $2 < \beta < 2.5$. We obtain additional improvements if we allow an additive error of 1 or 2. See Figure 1 for an illustration of our results.

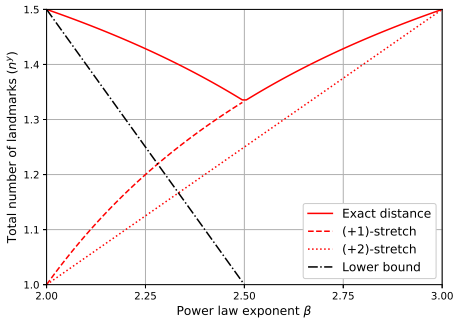


Figure 1: The x-axis is the exponent of the power law degree distribution and each value on the y-axis corresponds to a storage of $\tilde{O}(n^y)$. The lower bound is for exact distances.

While the dependence on β is complex, it is worth noting that in the entire range that we study ($\beta > 2$), the number of landmarks per node is at most $\tilde{O}(\sqrt{n})$ for *exact shortest paths*. This is in stark contrast to known impossibility results for general graphs, where no distance labeling with a multiplicative stretch less than 3 can use sub-linear space per node [26]. The query time of our algorithms is proportional to the number of landmarks per node, so we also get speed improvements.

Our algorithm is based on the pruned labeling algorithm of Akiba et al. [3], but differs in important ways. The pruned labeling algorithm initially posits that every node is a landmark for every other node, and then uses the BFS tree from each node to iteratively prune away unnecessary landmarks. In our approach, we apply a similar BFS with pruning procedure on a small subset of H (i.e. high degree vertices), but switch to lightweight local ball growing procedures up to radius l for all other vertices. As we show, the original pruned labeling algorithm requires storing $\tilde{\Omega}(n^2)$ landmarks on sparse Erdös-Rényi graphs. By growing local balls of size $\sqrt{n}$, our algorithm recovers exact distances with at most $\tilde{O}(n^{3/2})$ landmarks instead, for Erdös-Rényi graphs and random power law graphs with $\beta > 3$. Hence, our algorithm combines exploiting the existence of high-degree "central landmarks" with finding landmarks that are "locally important". Furthermore for $2 < \beta < 3$, by setting up the number of global landmarks H and the radius l suitably, we provably recover the upper bounds described in Figure 1. While the algorithmic idea is quite simple, the analysis is intricate.

We complement our algorithmic results with tight lower bounds for the regime when $\beta > 3$: the total length of any distance labeling schemes that answer distance queries exactly in this regime is almost surely $\tilde{\Omega}(n^{1.5})$. We also show that when $2 < \beta < 2.5$, any distance labeling scheme will generate labels of total size $\tilde{\Omega}(n^{3.5-\beta})$ almost surely. In particular, our algorithm achieves the optimal bound when β is close 2.

The parameter choice suggested by our theoretical analysis can be quite expensive to implement (as can earlier landmark based algorithms). We apply a simple but principled parameter tuning procedure to our algorithm that substantially improves the preprocessing time and generates a smaller set of landmarks at essentially no loss of accuracy. We conduct experiments on several real world graphs, both directed and undirected. First, compared to the pruned labeling algorithm, we find that our algorithm reduces the number of landmarks stored by 1.5-2.5x; the preprocessing time is reduced significantly as well. Next, we compare our algorithm to a variant of the distance oracle of Thorup and Zwick [37], which is believed to be theoretically optimal for worst-case graphs, as well as the distance sketch of Das Sarma et al [19] which has been found to be both efficient and useful in prior work [8]. For each graph, our algorithm substantially outperforms these two benchmarks. Details are in Section 5. It is important to note that the three algorithms we compare to also work much better on these real-world graphs than their theoretical guarantee, and we spend considerable effort tuning their parameters as well. Hence, the performance improvement given by our algorithm is particularly noteworthy.

It is worth mentioning that our technical tools only rely on bounding the growth rate of the breadth-first search. Hence we expect that our results can be extended to the related configuration model [22] as well. One limitation of our work is that the analysis does not apply directly to preferential attachment graphs, which correspond to another family of well known power law graphs. But we believe that similar results can be obtained there by adapting our analysis to that setting as well. This is left for future work.

Organizations: The rest of the paper is organized as follows. Section 2 introduces the basics of random graphs, reviews the pruned labeling algorithm and related work. Section 3 introduces our approach. We then present the analysis and experiments in Section 4 and Section 5. The lower bounds are presented in Section 6.

2 PRELIMINARIES AND RELATED WORK

2.1 Notations

Let $G = (V, E)$ be a directed graph with $n = |V|$ vertices and $m = |E|$ edges. For a vertex $x \in V$, denote by $d_{out}(x)$ the outdegree of x and $d_{in}(x)$ the indegree of x. Let $\mathcal{N}_{out}(x)$ denote the set of its out neighbors. Let $\text{dist}_G(x, y)$ denote the distance of x and y in G, or $\text{dist}(x, y)$ for simplicity. When G is undirected, then the outdegrees and indegrees are equal. Hence we simply denote by d_x the degree of every vertex $x \in V$. For an integer l and $x \in V$, denote by $\Gamma_l(x) = \{y : \text{dist}(x, y) = l\}$ the set of vertices at distance l from x. Denote by $N_l(x)$ the set of vertices at distance at most l from x.

We use notation $a \lesssim b$ to indicate that there exists an absolute constant $C > 0$ such that $a \leq Cb$. The notations $\tilde{O}(\cdot)$ and $\tilde{\Omega}(\cdot)$ hide poly-logarithmic factors.

2.2 Landmark based Labelings

In a landmark based labeling [21], each vertex x is assigned a set of forward landmarks $L_F(x)$ and backward landmarks $L_B(x)$. Each landmark set is a hash table, whose keys are vertices and values are distances. For example, if $y \in L_F(x)$, then the value associated with y would be $\text{dist}(x, y)$, which is the "forward" distance from x to y. Given the landmark sets $L_F(\cdot)$ and $L_B(\cdot)$, we estimate the distances as follows:

$$\min_{z \in L_F(x) \cap L_B(y)} \text{dist}(x, z) + \text{dist}(z, y), \forall x, y \in V. \tag{1}$$

If no common vertex is found between $L_F(x)$ and $L_B(y)$, then y is not reachable from x. In the worst case, computing set intersection takes $\Omega(\min(|L_F(x)|, |L_B(y)|))$ time.

Denote the output of equation (1) by $\hat{d}$. Clearly, we have $\hat{d} \geq \text{dist}(x, y)$ when y is reachable from x. The *additive stretch* of $\hat{d}$ is given by $\hat{d} - \text{dist}(x, y)$, and the *multiplicative stretch* is given by $\hat{d}/\text{dist}(x, y)$. When there are no errors for any pairs of vertices, such landmark sets are called *2-hop covers* [17].

There is a more general family of data structures known as labeling schemes [26], which associates a vector $\mathcal{L} : V \to \{0, 1\}^*$ for every vertex. When answering a query for a pair of vertices $x, y \in V$, only the two labels $\mathcal{L}(x)$ and $\mathcal{L}(y)$ are required without accessing the graph. The total length of $\mathcal{L}$ is given by $\sum_{x \in V} |\mathcal{L}(x)|$. It is clear from equation (1) that landmark sketches fall in the query model of labeling schemes.

2.3 The Pruned Labeling Algorithm

We review the pruned labeling algorithm [3] for readers who are not familiar. The algorithm starts with an ordering of the vertices, $\{x_1, x_2, \ldots, x_n\}$. First for x_1, a breadth first search (BFS) is performed over the entire graph. During the BFS, x_1 is added to the landmark set of every vertex.[1] Next for x_2, in addition to running BFS, a *pruning* step is performed before adding x_2 as a landmark. For example, suppose that a path of length l is found from x_2 to y. If x_1 lies on the shortest path fom x_2 to y, then by checking their landmark sets, we can find the common landmark x_1 to certify that $\text{dist}(x_2, y) = \text{dist}(x_2, x_1) + \text{dist}(x_1, y) \leq l$. In this case, x_2 is not added to y's landmark set, and the neighbors of y are pruned away. The above procedure is repeated on x_3, x_4, etc.

For completeness, we describe the pseudocode in Algorithm **??**. Note that the backward BFS procedure can be derived similarly. It has been shown that the pruned labeling algorithm is guaranteed to return exact distances [3].

Algorithm 1 PrunedLabeling (Akiba et al. [3])

Input: A directed graph $G = (V, E)$; An ordering of V, $\{x_1, x_2, \ldots, x_n\}$.

Let $O = \emptyset$, and $L_F(x) = L_B(x) = \emptyset$, for all $x \in V$
for $i = 1, \ldots, n$ **do**
    forwardBfs(x_i)
    backwardBfs(x_i)
    $O = O \cup \{x_i\}$
end for

procedure forwardBfs(x_i)
    Let Q be a priority queue and S be a hash set
    Set the priority of x_i to be zero
    while $Q \neq \emptyset$ **do**
        Let l be the minimum priority of Q
        Let u be the corresponding vertex
        $S = S \cup \{u\}$
        Let $\tilde{d} = \min_{y \in L_F(x_i) \cap L_B(u)} \text{dist}(x_i, y) + \text{dist}(y, u)$
        if $l < \tilde{d}$ **then** (otherwise u's neighbors are pruned)
            $L_B(u) = L_B(u) \cup \{x_i \to l\}$
            for $y \in \mathcal{N}_{out}(u)$ such that $y \notin O$ **do**
                Let q be the priority of y
                if $y \notin S$ and $l + 1 < q$ **then**
                    Decrease y's priority to $l + 1$
                end if
            end for
        end if
    end while
end procedure

2.4 Random Graphs

We review the basics of Erdös-Rényi random graphs. Let $G = G(n, p)$ be an undirected graph where every edge appears with probability p. It is well known that when $p \geq 2(\log n)/n$, G has only one connected component with high probability. Moreover, the neighborhood growth rate (i.e. $|\Gamma_{i+1}(x)| \,/\, |\Gamma_i(x)|$) is highly concentrated around its expectation, which is np. Formally, the following facts are well-known.

Fact 1 (Bollobás [10]). *Let $G = (V, E)$ be an undirected graph where every edge is sampled with probability $p = 2(\log n)/n$. Let $D = \lceil \frac{\log n}{\log(np)} \rceil$. Then the following are true with high probability:*

a) *The diameter of G is at most $D + 1$;*
b) *For any $x, y \in V$ and $l \leq D$, $\Pr(\text{dist}(x, y) \leq l) \leq \frac{(np)^{l+1}}{n(np-1)}$;*
c) *For any $x \in V$ and $l < D$, we have $\frac{1}{2} \leq \frac{|\Gamma_l(x)|}{(np)^l} \leq 2$.*

The Chung-Lu model: Let $p_x > 0$ denote a weight value for every vertex $x \in V$. Given the weight vector $\mathbf{p}$, the Chung-Lu model generalizes Erdös-Rényi graphs such that each edge is chosen independently with probability

$$\Pr[x \sim y] = \min\left\{\frac{p_x \cdot p_y}{\text{vol}(V)}, 1\right\}, \forall x, y \in V$$

[1] For directed graphs, there will be a forward BFS which looks at x_1's outgoing edges and its descendants, as well as a backward BFS which looks at x_1's incoming edges and its predecessors.

where $\text{vol}(V) = \sum_{x \in V} p_x$ denote the volume of V.

Random power law graphs: Let $f : [x_{\min}, \infty) \to \mathbb{R}$ denote the probability density function of a power law distribution with exponent $\beta > 1$, i.e. $f(x) = Zx^{-\beta}$, where $Z = (\beta - 1) \cdot x_{\min}^{\beta-1}$ [16]. The expectation of $f(\cdot)$ exists when $\beta > 2$. The second moment is finite when $\beta > 3$. When $\beta < 3$, the expectation is finite, but the empirical second moment grows polynomially in the number of samples with high probability. If $\beta < 2$, then even the expectation becomes unbounded as n grows.

In a random power law graph, the weight of each vertex is drawn independently from the power law distribution. Given the weight vector $\mathbf{p}$, a random graph is sampled according to the Chung-Lu model. If the average degree $\nu > 1$, then it is known that almost surely the graph has a unique giant component [15].

2.5 Related Work

Landmark based labelings: There is a rich history of study on how to preprocess a graph to answer shortest path queries [2, 4, 5, 11, 17, 37]. It is beyond our scope to give a comprehensive review of the literature and we refer the reader to survey [36] for references.

In general, it is NP-hard to compute the optimal landmark based labeling (or 2-hop cover). Based on an LP relaxation, a $\log n$ factor approximation can be obtained via a greedy algorithm [18]. See also the references [6, 7, 12, 21, 28] for a line of followup work. The current state of the art is achieved based on the pruned labeling algorithm [3, 20]. Apart from the basic version that we have already presented, bit-parallel optimizations have been used to speed up proprocessing [3]. Variants which can be executed when the graph does not fit in memory have also been studied [30]. It is conceivable that such techniques can be added on top of the algorithms that we study as well. For the purpose of this work, we will focus on the basic version of the pruned labeling algorithm. Compared to classical approaches such as distance oracles, the novelty of the pruned labeling algorithm is using the idea of *pruning* to reduce redundancy in the BFS tree.

Network models: Earlier work on random graphs focus on modeling the small world phenomenon [15], and show that the average distance grows logarithmically in the number of vertices. Recent work have enriched random graph models with more realistic features, e.g. community structures [31], shrinking diameters in temporal graphs [32].

Other existing mathematical models on special families of graphs related to distance queries include road networks [1], planar graphs and graphs with doubling dimension. However none of them can capture the expansion properties that have been observed on subnetworks of real-world social networks [34].

Previous work of Chen et al. [13] presented a 3-approximate labeling scheme requiring storage $\tilde{O}(n^{(\beta-2)/(2\beta-3)})$ per vertex, on random power law graphs with $2 < \beta < 3$. Our (+2)-stretch result improves upon this scheme in the amount of storage needed per vertex for $2 < \beta < 2.5$, with a much better stretch guarantee. Another related line of work considers compact routing schemes on random graphs. Enachescu et al. [24] presented a 2-approximate compact routing scheme using space $O(n^{1.75})$ on Erdös-Rényi random graphs, and Gavoille et al. [25] obtained a 5-approximate compact routing scheme on random power law graphs.

3 OUR APPROACH

In order to motivate the idea behind our approach, we begin with an analysis of the pruned labeling algorithm on Erdös-Rényi random graphs. While the structures of real world graphs are far from Erdös-Rényi graphs, the intuition obtained from the analysis will be useful. Below we describe a simple proposition which states that for sparse Erdös-Rényi graphs, the pruned labeling algorithm outputs $\tilde{\Omega}(n^2)$ landmarks.

Proposition 2. *Let $G = (V, E)$ be an undirected Erdös-Rényi graph where every edge appears with probability $p = 2(\log n)/n$. For any ordering of the vertices $V = \{x_1, x_2, \ldots, x_n\}$, with high probability over the randomness of G, the total number of landmarks produced by Algorithm **??** is at least $\tilde{\Omega}(n^2)$.*

Proof sketch. We first introduce a few notations. Let $r = np$ denote the growth rate of G. Let $\varepsilon = 1/\log n$. Consider a vertex x_i where $1 \le i \le \varepsilon n$. Denote by $X_{-i} = \{x_1, \ldots, x_{i-1}\}$. Consider any $u \in V$, if none of the shortest paths from x_i to u intersect with X_{-i}, then (x_i, u) is called a *bad* pair. Note that x_i must be added to u's landmark set by Algorithm **??**, because during the BFS from x_i, all estimates through X_{-i} will be strictly larger than $\text{dist}(x_i, u)$. Hence, to lower bound the total landmark sizes, it suffices to count the number of *bad* pairs. In the following, we show that in expectation for every x_i where $1 \le i \le \varepsilon n$, there are at least $n/(\log n)^3$ vertices u such that (x_i, u) are *bad*. It follows that Algorithm **??** requires at least $\varepsilon n^2/(\log n)^3 \ge \tilde{\Omega}(n^2)$ in expectation.

Let $D = \lfloor \log_r n - 2 \rfloor$. Consider $\Gamma_D(x_i)$, the set of vertices at distance equal to D from x_i. We count the number of *bad* vertices in $\Gamma_D(x_i)$ at follows. For each $1 \le k \le D$, consider the intersection $\Gamma_k(x_i) \cap X_{-i}$ and their subtree down to $\Gamma_D(x_i)$.

Starting from any $y \in \Gamma_k(x_i) \cap X_{-i}$, the subtree of y would result in *good* vertices in $\Gamma_D(x_i)$, whose distance from x_i can be correctly estimated (c.f. line 15-16 in Algorithm **??**). In expectation, the size of the intersection is $r^k\varepsilon$, because the probability that any two vertex has distance k on G is equal to r^k/n, and there are at most εn vertices in X_{-i}. Next, each y results in r^{D-k} vertices in its $(D-k)$-th level neighborhood. Combined together, the total number of *good* vertices which are covered by $\Gamma_k(x_i) \cap X_{-i}$ is at most $r^k\varepsilon \times r^{D-k} = \varepsilon r^D$. By summing over all $k \le D$, we obtain that the total number of *good* vertices in $\Gamma_D(x_i)$ is at most $D\varepsilon r^D$.

On the other hand, the size of $\Gamma_D(x_i)$ is r^D. Hence the total number of *bad* vertices is at least $(1 - D\varepsilon)r^D \ge n/\log^3 n$. To show that the the proposition holds with high probability, it suffices to apply concentration results on neighborhood growth in the arguments above. We omit the details.

□

The interesting point from the above analysis is that $\Theta(n)$ landmarks are added throughout the first εn vertices. The reason is that there are no high degree vertices in Erdös-Rényi graphs, hence the landmarks we have added in the beginning do not cover the shortest paths for many vertex pairs later.Secondly, a large number of distant vertices are added in the landmark sets, which do not lie on the shortest paths of many pairs of vertices.

Motivated by the observation, we introduce our approach as follows. We start with an ordering of the vertices. For the top H

vertices in the ordering, we perform the same BFS procedure with pruning. For the rest of the vertices, we simply grow a local ball up to a desired radius. Concretely, only the vertices from the local ball will be used as a landmark. Algorithm **??** describes our approach in full.[2] As a remark, when the input graph G is undirected, it suffices to run one of the forward or backward BFS procedures, and for each vertex, its forward and backward landmark sets can be combined to a single landmark set.

Algorithm 2 ApproximatePruning

Input: A directed graph $G = (V, E)$; An ordering of V $\{x_1, x_2, \ldots, x_n\}$; The number of global landmarks H; The set of radiuses $\{l_i\}_{i=H+1}^{n}$.

Let $O = \emptyset$, and $L_F(x) = L_B(x) = \emptyset$, for any $x \in V$.
for $i = 1, \ldots, H$ **do**
  forwardBfs(x_i)
  backwardBfs(x_i)
  $O = O \cup \{x_i\}$
end for
for $i = H + 1, \ldots, n$ **do**
  localForwardBfs(x_i, l_i)
  localBackwardBfs(x_i, l_i)
end for
procedure localForwardBfs(x_i, l_i)
  for y such that $\text{dist}(x_i, y) \le l_i - 1$ **do**
    $L_F(x_i) = L_F(x_i) \cup (y \to \text{dist}(x_i, y))$
  end for
  for y such that $\text{dist}(x_i, y) = l_i$ and $\exists z$ s.t. $\text{dist}(x, z) = l_i - 1$, $(z, y) \in E$, $d_{out}(z) \le d_{out}(y)$ **do**
    $L_F(x_i) = L_F(x_i) \cup (y \to \text{dist}(x_i, y))$
  end for
end procedure

Recall that the backward and forward BFS procedures do a breadth first search with a pruning step before enqueing a vertex (c.f. Algorithm **??**). For each x_i with $i > H$, the parameter l_i controls the depth of the local ball we grow from x_i. Furthermore, at the bottom layer, we only add vertices whose outdegree is higher than any of its predecessor to x_i's landmark set. The intuition is that vertices with higher outdegrees are more likely to be used as landmarks.

We begin by analyzing Algorithm **??** for Erdös-Rényi graphs, as a comparison to Proposition 2. The following proposition shows that without using global landmarks, local balls of suitable radius suffice to cover all the desired distances. The proof is by observing that for each vertex, if we add the closest $\sqrt{n}$ vertices to the landmark set of every vertex, then the landmark sets of every pair of vertices will intersect with high probability, i.e. we have obtained a 2-hop cover.

Proposition 3. *Let $G = (V, E)$ be an undirected random graph where each edge is sampled with probability $p = 2(\log n)/n$. By setting $H = 0$ and $l_i = l = \lceil \frac{\log n}{2 \log np} \rceil + 1$ for all $1 \le i \le n$, we have that Algorithm **??** outputs a 2-hop cover with at most $\tilde{O}(n^{3/2})$ landmarks with high probability.*

Proof. Denote by $L(x)$ the landmark set obtained by Algorithm **??**, for every $x \in V$. We will show that with high probability:

a) For all $x_i, x_j \in V$, $L(x_i) \cap L(x_j) \neq \emptyset$. This implies that $L(\cdot)$ is a 2-hop cover.
b) The size of $L(x_i)$ is less than $\tilde{O}(\sqrt{n})$, for all $x_i \in V$.

Claim a) follows because the diameter of G is at most $2l - 1$ with high probability by Fact 1. Note that $L(x_i)$ contains $N_{l-1}(x_i)$, the set of vertices with distance at most $l - 1$. If $\text{dist}(x_i, x_j) \le (l - 1) + (l - 1)$, $N_{l-1}(x_i)$ and $N_{l-1}(x_j)$ already intersect. Otherwise, since the diameter is at most $2l - 1$, these two neighborhoods must be connected by an edge e. Suppose between e's two endpoints, the one with a lower degree is on x_i's side, then the local BFS from x_i must add the other endpoint to $L(x_i)$, and vice versa. Therefore, $L(x_i)$ must intersect with $L(x_j)$.

Claim b) is because $L(x_i)$ is a subset of $N_l(x_i)$. By Fact 1, the size of $N_l(x_i)$ is at most $4(np)^l \lesssim \tilde{O}(\sqrt{n})$. Hence, the proof is complete. □

4 RANDOM POWER LAW GRAPHS

In this section we analyze our algorithm on random power law graphs. We begin with the simple case of $\beta > 3$, which generalizes the result on Erdös-Rényi graphs. Because the technical intuition is the same with Proposition 3, we describe the result below and omit the proof.

Proposition 4. *Let $G = (V, E)$ be a random power law graph with average degree $v > 1$ and power law exponent $\beta > 3$. For each $x_i \in V$, let l_i be the smallest integer such that the number of edges between $N_{l_i}(x_i)$ and $V \backslash N_{l_i}(x_i)$ is at least $\delta\sqrt{n}$, where $\delta = 5\sqrt{v \log n}$.*

*By setting $H = 0$ and $\{l_i\}_{i=1}^{n}$, Algorithm **??** outputs a 2-hop cover with high probability. Moreover, each vertex uses at most $O(\sqrt{n} \log^2 n)$ landmarks.*

Remark: The high level intuition behind our algorithmic result is that as long as the breadth-first search process of the graph grows neither too fast nor too slow, but rather at a proper rate, then an efficient distance labeling scheme can be obtained. Proposition 4 can be easily extended to configuration models with bounded degree variance. It would be interesting to see if our results extend to preferential attachment graphs and Kronecker graphs.

The case of $2 < \beta < 3$: Next we describe the more interesting case with power law exponent $2 < \beta < 3$. Here the graph contains a large number of high degree vertices. By utilizing the high degree vertices, we show how to obtain exact distance landmark schemes, (+1)-stretch schemes and (+2)-stretch schemes. The number of landmarks used varies depending on the value of β. We now state our main result as follows.

Theorem 5. *Let $G = (V, E)$ be a random power law graph with average degree $v > 1$ and exponent $2 < \beta < 3$. Let*

$$K = \begin{cases} \sqrt{n}, \text{for } 2.5 \le \beta \le 3 \\ n^{\frac{1}{(4-\beta)(\beta-1)}}, \text{for } 2 < \beta < 2.5. \end{cases}$$

Let H be the number of vertices whose degree is at least K in G. Let $\pi = \{x_i\}_{i=1}^{n}$ be any ordering of vertices V by their degrees in a non-increasing order. For each vertex $x_i \in V$, let l_i be the smallest integer such that the number of edges between $N_{l_i-1}(x_i)$ and $V \backslash N_{l_i-1}(x_i)$ is at least $\delta n^{(\beta-2)/(\beta-1)}$, where $\delta = 4v \cdot \log^2 n$.

*With ordering π, parameters H and $\{l_i\}_{i=H+1}^{n}$, Algorithm **??** outputs a 2-hop cover with high probability. Moreover, the maximum*

[2] Here we have omitted the details of the local backward BFS procedure, which can be derived similar to the local forward BFS procedure.

number of landmarks used by any vertex is at most

$$O\left(\max\left(n^{\frac{\beta-2}{\beta-1}}, n^{\frac{3-\beta}{4-\beta}}\right)\log^3 n\right).$$

The above theorem says that in Algorithm **??**, first we use vertices whose degrees are at least K as global landmarks. Then for the other vertices x_i, we grow a local ball of radius l_i, whose size is (right) above $n^{(\beta-2)/(\beta-1)}$. The two steps together lead to a 2-hop cover. We now build up the intuition for the proof.

Building up a +1-stretch scheme: First, it is not hard to show that G contains a heavy vertex whose degree is $n^{1/(\beta-1)}$, by analyzing the power law distribution. Note that $K \leq n^{1/(\beta-1)}$, hence we have added all such high degree vertices as global landmarks. This part, together with the local balls, already gives us a (+1)-stretch landmark scheme.

To see why, consider two vertices x_i, x_j. If their local balls (of size $n^{(\beta-2)/(\beta-1)}$) already intersect, then we can already compute their distances correctly from their landmark sets. Otherwise, since the bottom layers of x_i and x_j already have weight/degree $n^{(\beta-2)/(\beta-1)}$, they are at most two hops apart, by connecting to the heavy vertex with degree $n^{1/(\beta-1)}$. Recall that the heavy vertex is added to the landmark sets of every vertex. Hence, the estimated distance is at most off by one. As a remark, to get the (+1)-stretch landmark scheme, the number of landmarks needed per vertex is on the order of $n^{(\beta-2)/(\beta-1)}$. This is because we only need to use vertices whose degrees are at least $n^{1/(\beta-1)}$ as global landmarks (there are only $\log n$ of them), as opposed to H in Theorem 5.

Fixing the +1-stretch: To obtain exact distances, for each vertex on the boundary of radius $l_i - 1$, we add all of its neighbors with a higher degree to the landmark set (c.f. line 15-17 in Algorithm **??**). Whenever there is an edge connecting the two boundaries, the side with a lower degree will add the other endpoint as a landmark, which resolves the (+1)-stretch issue. For the size of landmark sets, it turns out that fixing the (+1)-stretch for the case $2 < \beta < 2.5$ significantly increases the number of landmarks needed. Specifically, the costs are $n^{(5-\beta)/(4-\beta)}$ landmarks per node.

Intuition for the +2-stretch scheme: As an additional remark, one can also obtain a (+2)-stretch landmark sketch by setting l_i in Algorithm **??** in a way such that every vertex stores the closest $\tilde{\Theta}(n^{(\beta/2)-1})$ vertices in its landmark set. This modification leads to a (+2)-stretch scheme, because for two vertices x, y, once the bottom layers of x, y have size at least $\tilde{\Theta}(n^{(\beta/2)-1})$, they are at most three hops away from each other. The reason is that with high probability, the bottom layer will connect to a vertex with weight $\Omega(\sqrt{n})$ in the next layer (which will all be connected), as it is not hard to verify that the volume of all vertices with weight $\sqrt{n}$ is $\Omega(n^{(4-\beta)/2})$. By a similar proof to Theorem 5, the maximum number of landmarks used per vertex is at most $\tilde{O}(n^{(\beta-2)/2})$.

We refer the reader to the full version for details of the full proof. The technical components involve carefully controlling the growth of the neighborhood sets by using concentration inequalities.

5 EXPERIMENTS

In this section, we substantiate our results with experiments on a diverse collection of network datasets. A summary of the findings are as follows. We first compare Algorithm **??** with the pruned labeling algorithm [3]. Recall that our approach differs from the pruned labeling algorithm by only performing a thorough BFS for a small set of vertices, while running a lightweight local ball growing procedure for most vertices. We found that this simple modification leads to 1.5-2.5x reduction in number of landmarks stored. The preprocessing time is reduced by 2-15x as well. While our algorithm does not always output the exact distance like the pruned labeling algorithm, we observe that the stretch is at most 1%, relative to the average distance.

Next we compare our approach to two approximate distance sketches with strong theoretical guarantees, Das Sarma et al. sketch [8, 19] and a variant of Thorup-Zwick's 3-approximate distance oracle [37], which uses high degree vertices as global landmarks [13]. We observe that our approach incurs lower stretch and requires less space compared to Das Sarma et al. sketch. The accuracy of Thorup-Zwick sketch is comparable to ours, but we require much fewer landmarks.

5.1 Experimental Setup

To ensure the robustness of our results, we measure performances on a diverse collection of directed and undirected graphs, with the datasets coming from different domains, as described by Table 1. Stanford, Google and BerkStan are all Web graphs in which edges are directed. DBLP (collaboration network) and Youtube (friendship network) are both undirected graphs where there is one connected component for the whole graph. Twitter is a directed social network graph with about 84% vertices inside the largest strongly connected component. All the datasets are downloaded from the Stanford Large Network Dataset Collection [33].

Table 1: Datasets used in experiments.

graph	# nodes	# edges	category	type
DBLP	317K	1.0M	Collaboration	Undirected
Twitter	81K	1.8M	Social	Directed
Stanford	282K	2.3M	Web	Directed
Youtube	1.1M	3.0M	Social	Undirected
Google	876K	5.1M	Web	Directed
BerkStan	685K	7.6M	Web	Directed

Implementation details: We implemented all four algorithms in Scala, based on a publicly available graph library.[3] The experiments are conducted on a 2.30GHz 64-core Intel(R) Xeon(R) CPU E5-2698 v3 processor, 40MB cache, 756 GB of RAM. Each experiment is run on a single core and loads the graph into memory before beginning any timings. The RAM used by the experiment is largely dominated by the storage needed for the landmark sets.

Parameters: In the comparison between the pruned labeling algorithm and our approach, we order the vertices in decreasing order by the indegree plus outdegree of each vertex.[4] Recall that there are

[3] https://github.com/teapot-co/tempest

[4] There are more sophisticated techniques such as ordering vertices using their betweenness centrality scores [20]. It is conceivable that our algorithm can be combined with such techniques.

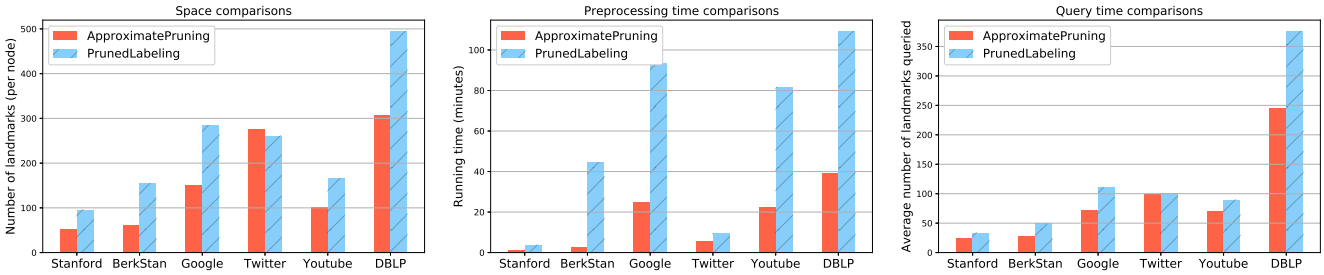


Figure 2: Comparing the efficiency of our approach to the pruned labeling algorithm.

Table 2: Measuring the accuracy of our approach.

	Stanford	BerkStan	Google	Twitter	Youtube	DBLP
Relative Average Stretch	0.37%	0.20%	0.51%	0.29%	0.33%	1.1%
Maximum Relative Stretch	21/10	10/7	8/5	4/3	4/3	7/5
Average Additive Stretch	0.046	0.030	0.060	0.014	0.018	0.075
Maximum Additive Stretch	11	3	3	1	2	2
Average Distance	12.3	14.6	11.7	4.9	5.3	6.8

two input parameters used in our approach, the number of global landmarks H and the radiuses of local balls $\{l_i\}_{i=H+1}^{n}$. To tune H, we start with 100, then keep doubling H to be 200, 400, etc. The radiuses $\{l_i\}_{i \geq H}$ are set to be 2 for all graphs.[5]

Benchmarks: For the Thorup-Zwick sketch, in the first step, $H = \sqrt{n}$ vertices are sampled uniformly at random as global landmarks. In the second step, every other vertex grows a local ball as its landmark set until it hits any of the $\sqrt{n}$ vertices. All vertices within the ball are used as landmarks. This method uses $O(n^{3/2})$ landmarks and achieves 3-stretch in worst case. In the follow up work of Chen et al. [13], the authors show a variant which uses high degree vertices as global landmarks and observe better performance. We implement Chen et al.'s variant in our experiment, and use the H vertices with the highest indegree plus outdegree as global landmarks. In the experiment, we start with H equal to $\sqrt{n}$. Then we report results for $\sqrt{n}$ multiplied by $\{2, 1/2, 1/4, 1/8\}$.

For the Das Sarma et al. sketch, first, $\log n$ sets S_i of different sizes are sampled uniformly from the set of vertices V, for $0 \leq i < \log n$, where the size of S_i is 2^i. Then a breadth first search is performed from S_i, so that every vertex $x \notin S_i$ finds its closest vertex inside S_i in graph distance. This closest vertex is then used as a landmark for x. The number of landmarks used in Dar Sarma's sketch is $n \log n$, and the worst case multiplicative stretch is $\log n$. If more accurate estimation is desired, one can repeat the same procedure multiple times and union the landmark sets together. We begin with 5 repetitions, then keep doubling it to be 10, 20 etc.

Our approach differs from the above two methods by using the idea of pruning while running BFS. This dramatically enhances performance in practice, as we shall see in our experiments.

Metrics: We measure the stretch of the estimated distances, and compute aggregated statistics over a large number of queries. For a query (x, y), if y is reachable from x, but the algorithm reports no common landmark between the landmark sets of x and y, then we count such a mistake as a "False disconnect error." On the other hand, if y is not reachable from x, then it is not hard to see that our algorithm always reports correctly that y is not reachable from x. In the experiments, we compute $\text{dist}(x, y)$ using Dijkstra's algorithm.

To measure space usage, we report the number of landmarks per node used in each algorithm as a proxy. Since the landmark sets are stored in Int to Float hash maps, the actual space usage would be eight bytes times the landmark sizes in runtime, with a constant factor overhead.

For the query time, recall that for each pair of vertices (x, y), we estimate their distance by looking at the intersection of $L_F(x)$ and $L_B(y)$ and compute the minimum interconnecting distance (c.f. equation 1). To find the minimum, we iterate through the smaller landmark set. Hence the running time is $\min(|L_F(x)|, |L_B(y)|)$ multiplied by the time for a hash map lookup, which is a small fixed value in runtime. A special case is when $y \in L_F(x)$ or $x \in L_B(y)$, where only one hash map lookup is needed. We will report the number of hash map lookups as a proxy for the query time.[6]

5.2 Comparisons to Exact Methods

We report the results comparing our approach to the pruned labeling algorithm. The pruned labeling algorithm is exact. To measure the accuracy of our approach, we randomly sample 2000 pairs of source and destination vertices. The number of global landmarks is set to be 400 for the Stanford dataset, 1600 for the DBLP dataset, and 800 for the rest of the datasets.

Figure 2 shows the preprocessing time, the number of landmarks and average query time used by both algorithms. We see that our approach reduces the number of landmarks used by 1.5-2.5x, except

[5]It follows from our theoretical analysis that the radiuses should be less than half of the average distance. As a rule of thumb, setting the radius as 2 works based on our experiments.

[6]It is conceivable that more sophisticated techniques may be devised to speedup set intersection. We leave the question for future work.

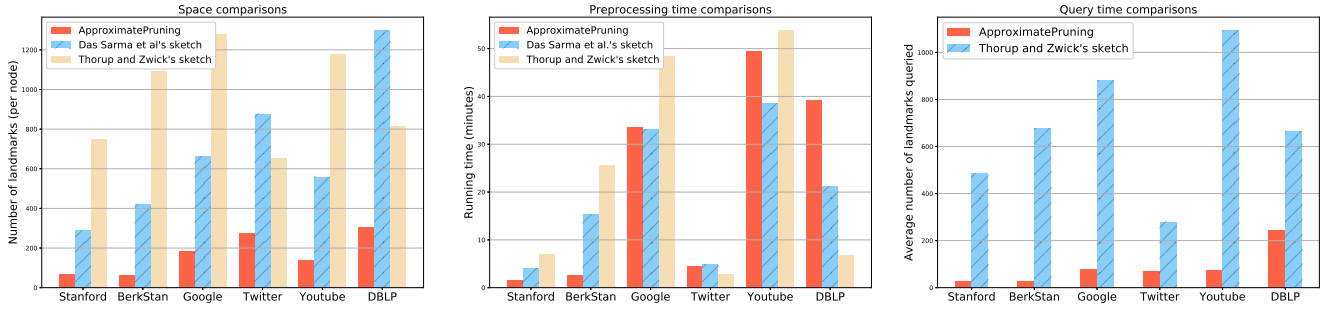


Figure 3: Comparing the efficiency of our approach to two well known distance sketches with strong theoretical guarantees.

Table 3: Measuring the stretch for all three methods.

	Stanford	BerkStan	Google	Twitter	Youtube	DBLP
DS et al. sketch	20.9%	17.2%	21.1%	11.1%	13.1%	5.4%
TZ sketch	0.30%	0.21%	0.36%	1.65%	0.03%	2.15%
Our approach	0.16%	0.20%	0.22%	0.29%	0.04%	1.1%

Table 4: Varying H in TZ sketch.

Youtube	$\sqrt{n}/2$	$\sqrt{n}/4$	$\sqrt{n}/8$	Ours
Stretch	0.04%	0.11%	0.07%	0.04%
# Landmarks	731	648	811	137
Preprocessing	37m	31m	35m	50m

on the Twitter dataset.[7] Our approach performs favorably in terms of preprocessing time and query time as well.

The accuracy of our computed estimate is shown in Table 2. We have also measured the median additive stretch, which turns out to be zero in all the experiments. To get a more concrete sense of the accuracy measures, consider the Google dataset as an example. Since the average additive stretch is 0.06 and there are 2000 pairs of vertices, the total additive stretch is at most 120 summing over all 2000 pairs! Specifically, there can be at most 120 queries with non-zero additive stretch and for all the other queries, our approach returns the exact answer. Meanwhile, among all the datasets, we observed only one "False disconnect error" in total. It appeared in the Stanford Web graph experiment, where the true distance is 80.

5.3 Comparisons to Approximate Methods

Next we compare our approach to Das Sarma et al.'s sketch (or DS et al. sketch in short) and the variant of Thorup and Zwick's sketch (or TZ sketch in short). Similar to the previous experiment, we sample 2000 source and destination vertices uniformly at random to measure the accuracy.

We start by setting the number of global landmarks to $\sqrt{n}$ in Thorup-Zwick sketch. To allow for a fair comparison, we tune our approach so that the relative average stretch is comparable or lower. Specifically, the Stanford, BerkStan and Twitter datasets use $H = 800$, the Google and DBLP datasets use $H = 1600$ and the Youtube dataset uses $H = 3200$.

Figure 3 shows the number of landmarks needed in each algorithm as well as the amount of preprocessing time consumed. Overall, our approach uses much fewer landmarks than the other two algorithms. In terms of preprocessing time, our approach is comparable or faster on all datasets, except on the DBLP network. We suspect that this may be because the degree distribution of the DBLP network is flatter than the others. Hence performing the pruning procedures on a small subset of high degree vertices are less effective in such a scenario.

We next report the relative average stretch for all three methods. As can be seen in Table 3, our approach is comparable to or slightly better than Thorup and Zwick's sketch, but much more accurate than Das Sarma et al's sketch. Note that the latter performed significantly worse than the other two approaches. We suspect that this may be because the sketch does not utilize the high degree vertices efficiently. Lastly, our approach performs favorably in the query time comparison as well. Note that the query time of Das Sarma et al.'s sketch are not reported because of the worse accuracy.

Effect of parameter choice: Note that in the above experiment, for Thorup and Zwick's sketch, we have set the number of global landmarks H to be $\sqrt{n}$. In the next experiment, we vary the value of H to $\sqrt{n}$ multiplied by $\{2, 1/2, 1/4, 1/8\}$.

First, we report a detailed comparison on the Google dataset in Figure 4. Note that when $H = 2\sqrt{n}$, the Thorup and Zwick's sketch requires over 2000 landmarks per node which is significantly larger than the other values. Hence, we dropped the data point from the plot. For our approach, we double H from 100 up to 1600. Overall, we can see that our approach requires fewer landmarks across different stretch levels.

Next, we report brief results on the Youtube dataset in Table 4 since the results are similar. The conclusions obtained from other datasets are qualitatively similar, and hence omitted.

[7] By setting the radiuses $\{l_i\}$ to be 1, we incur 0.72% relative additive stretch by using 173 landmarks per node, which improves over the pruned labeling algorithm by 1.5x.

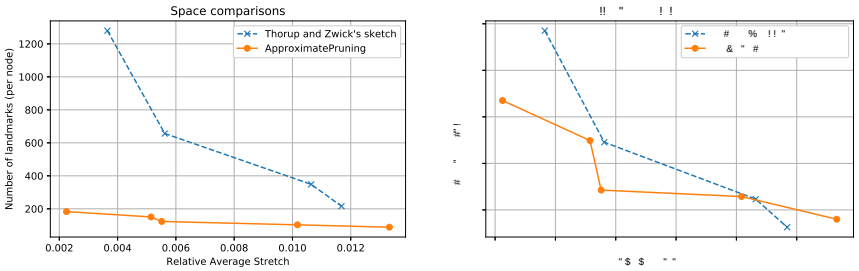


Figure 4: Varying H in TZ sketch and our approach, on the Google dataset.

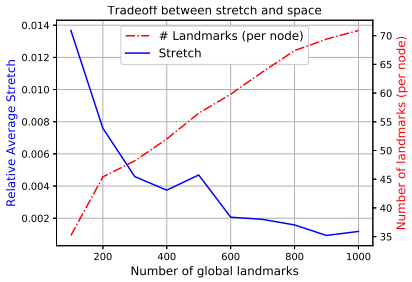


Figure 5: Tradeoff on the Stanford dataset.

5.4 More Experimental Observations

By varying the number of global landmarks used Algorithm **??**, it is possible to obtain a smooth tradeoff between stretch and number of landmarks used. As an example, we present the tradeoff curve for the Stanford Web dataset in Figure 5. Here we vary the number of global landmarks used from 100 to 1000. As one would expect, the relative average stretch decreases while the number of landmarks stored increases.

6 FUNDAMENTAL LIMITS OF LANDMARK SKETCHES

This section complements our algorithm with lower bounds. We begin by showing a matching lower bound for Erdös-Rényi graphs, saying that any 2-hop cover needs to store at least $\tilde{\Omega}(n^{3/2})$ landmarks. The results imply that the parameter dependence on n of our algorithm is tight for Erdös-Rényi graphs and random power law graphs with power law exponent $\beta > 3$. It is worth mentioning that the results not only apply to landmark sketches, but also work for the family of labeling schemes. Recall that labeling schemes associate a labeling vector for each vertex. To answer a query for a pair of vertices (x, y), only the labeling vectors of x, y are accessed. We first state the lower bound for Erdös-Rényi graphs.

Theorem 6. *Let $G = (V, E)$ be an Erdös-Rényi graph where every edge is sampled with probability $p = 2\log n/n$. With high probability over the randomness of G, any labelings which can recover all pairs distances exactly have total length at least $\Omega(n^{3/2}/\log^4 n)$.*

In particular, any 2-hop cover needs to store at least $\Omega(n^{3/2}/\log^4 n)$ many landmarks with high probability.

For a quick overview, we divide V into $\sqrt{n}$ sets of size $\sqrt{n}$ each. We will show that the total labeling length for each set of $\sqrt{n}$ vertices has to be at least $\tilde{\Omega}(n)$. By union bound over all the $\sqrt{n}$ sets, we obtain the desired conclusion. We now go into the proof details.

Proof. Denote by $r = np$. Let $d = \lfloor \frac{\log n}{2\log(np)} \rfloor - c$, where c is a fixed constant (e.g. $c = 2$ suffices). Divide V into groups of size $\sqrt{n}$. Clearly, there are $\sqrt{n}$ disjoint groups – let S be one of them. Denote by c_1 a fixed constant which will be defined later. We argue that

$$\Pr[\text{The total label length of } S \le c_1 \cdot r^{1-2c} n] \lesssim r^{1-2c}. \qquad (2)$$

Hence by Markov's inequality, with high probability except for $(\log n) r^{1-2c}\sqrt{n}$ groups, all the other groups will have label length at least $c_1 \cdot r^{1-2c} n \gtrsim \tilde{\Omega}(n)$, because $r \le 2\log n$. Hence we obtain the desired conclusion. For the rest of the proof, we focus on proving equation (2) for the group S.

Let $\{x_1, x_2, \ldots, x_{|S|}\}$ be an arbitrary ordering of S. We grow the neighborhood of each vertex in S one by one, up to level d. Denote by $G_1 = (V_1, E_1)$, where $V_1 = V$ and $E_1 = E$. For any $i \ge 1$, if $x_i \in V_i$, then we define define $T(x_i)$ to be the set of of vertices in V_i whose distance is at most d from x_i. Define $L(x_i) \subseteq T(x_i)$ to be the set of vertices in G_i whose distance is equal to d from x_i. On the other hand if $x_i \notin V_i$, then $T(x_i)$ and $L(x_i)$ are both empty. More formally,

$$T(x_i) := \begin{cases} \{y : \mathrm{dist}_{G_i}(x_i, y) \le d\}, & \text{if } x_i \in V_i \\ \varnothing, & \text{otherwise.} \end{cases}$$
$$L(x_i) := \{y \in T(x_i) : \mathrm{dist}_{G_i}(x_i, y) = d\}$$

We then define $F_i = \cup_{j=1}^{i} T(x_j)$. Denote by G_{i+1} to be the induced subgraph of G_i on the remaining vertices $V_{i+1} = V \backslash F_i$. We show that with high probability, a constant fraction of vertices $x_i \in S$ satisfy that $|L(x_i)| \ge \Omega((np)^d)$. □

Lemma 7 (Martingale inequality). *In the setting of Theorem 6, with high probability, at least $|S|/2$ vertices $x_i \in S$ satisfy that $|L(x_i)| \ge r^d/6$.*

Proof. For any $1 \le i \le |S|$, consider

$$X_i := \begin{cases} 1 & \text{if } x_i \notin V_i, \text{ or } |F_{i-1}| > |S| \cdot r^d \log n, \text{or } |L(x_i)| \ge r^d/6 \\ 0 & \text{otherwise.} \end{cases}$$

We claim that $\Pr[X_i = 1 \mid X_1, \ldots, X_{i-1}]$ with high probability. It suffices to consider the case $x_i \in V_i$ and $|F_{i-1}| \le |S| r^d \log n$. It is not hard to verify that $|F_{i-1}| \le n/\log n$ by our setting of d. Hence the size of V_i is at least $n(1 - 1/\log n)$. Note that the subgraph G_i is still an Erdös-Rényi random graph, and the number of vertices is at least $n(1 - 1/\log n)$. By Fact 1c), the size of $L(x_i)$ is at least

$$\frac{1}{2} r^d (1 - \log^{-1} n)^d \ge r^d/6,$$

since $d \le \log n$.

Thus by Azuma-Hoeffding inequality, $\sum_{i=1}^{|S|} X_i \ge 0.99\,|S|$ with high probability. We will show below that the contributions to $\sum_{i=1}^{|S|} X_i$ from $x_i \notin V_i$ and $|F_{i-1}| > |S| r^d \log n$ is less than $0.02\,|S|$. Hence by taking union bound, we obtain the desired conclusion.

First, we show that the number of x_i such that $x_i \notin V_i$ are at most $0.01|S|$ with high probability. Note that $x_i \notin V_i$ implies that

there exists some vertex x_j with $j < i$ such that $\text{dist}(x_i, x_j) \le d$. On the other hand, by Fact 1,

$$\Pr[\text{dist}(x, y) \le d] \le \frac{3r^d}{n}, \forall x, y \in S.$$

Hence, it is not hard to verify that the expected number of vertex pairs in S whose distance is at most d, is $O(|S|^2 r^{2d}/n) \lesssim |S| / \log n$, by the setting of d. By Markov's inequality, with probability $1 - 1/\log n$ only $0.01\,|S|$ vertex pairs have distance at most d in S. Hence there exists at most $0.01\,|S|$ i's such that $x_i \notin V_i$.

Secondly for all $1 \le i \le |S|$, the set of vertices T_i is a subset of $N_d(x_i)$, the set of vertices within distance d to x_i on G. By Fact 1c), the size of $N_d(x_i)$ is at most $2r^d$. Hence we have $|F_i| \le 2\,|S|\, r^d$ for all $1 \le i \le |S|$ with high probability. This proves the Lemma. □

Now we are ready to finish the proof. Given the labels of S, we can recover all pairwise distances in S. Let $\text{dist}_S : S \times S \to \mathbb{N}$ denote the distance function restricted to S. Consider the following:

a) $\exists\, |S|^2/9$ pairs (x_i, x_j) such that $\text{dist}_S(x_i, x_j) \le 2d + 1$. We know by Fact 1 that $\Pr[\text{dist}(x_i, x_j) \le 2d+1] \le 2r^{2d+1}/n$, for any $x_i, x_j \in S$. Hence the expected number of pairs with distance at most $2d+1$ in S, is at most $2\,|S|^2 \cdot r^{2d+1}/n \lesssim r^{1-2c}n$. By Markov's inequality, the probability that a random graph induces any such distance function is $r^{1-2c}n/(|S|^2/8) \lesssim r^{1-2c}$.

b) The number of pairs such that $\text{dist}_S(x_i, x_j) \le 2d + 1$ is at most $|S|^2/8$ in S. Let A denote

$$\left\{(x, y) \in S \times S \mid \text{dist}(x, y) > 2d + 1, \text{ and } |L(x)|, |L(y)| \ge r^d/6\right\}.$$

By Lemma 7 and our assumption for case b), the size of A is at least $\binom{|S|/2}{2} - |S|^2/8 \ge |S|^2/9$. For any $(x, y) \in A$, $L(x)$ and $L(y)$ are clearly disjoint. Note that the event whether there exists an edge between $L(x)$ and $L(y)$ is independent, conditional on revealing the subgraph for all $x \in S$ up to distance d. Hence

$$\begin{aligned}
&\Pr\left[\text{dist}_S(x, y) > 2d + 1,\ \forall (x, y) \in A\right] \\
&\le \prod_{(x,y)\in A} \Pr\left[\text{there is no edge between } L(x) \text{ and } L(y)\right] \\
&\le \prod_{(x,y)\in A} (1-p)^{|L(x)|\times|L(y)|} \\
&\le \exp\left(-p \times |A| \times r^{2d}/72\right) \qquad (\text{because } |L(x)|, |L(y)| \ge r^d/6) \\
&\le \exp(-c_1 r^{1-2c} n). \qquad (\text{because } |A| \ge n/9)
\end{aligned}$$

where $c_1 = 72 \times 9$ in the last line. Denote by $\kappa = c_1 \cdot r^{1-2c}n$. Note that the number of labelings of length (or number of bits) less than κ is at most 2^κ. For each labeling, the probability that it correctly gives all pairs distances is at most $\exp(-\kappa)$ by our argument above. Therefore by union bound, the probability that the total labeling length of $|S|$ is at most κ is at most $2^\kappa \cdot \exp(\kappa) \le r^{1-2c}$ for large enough n.

To recap, by combining case a) and b), we have shown that equation (2) is true. Hence the proof is complete.

Extensions to $\beta > 3$: It is worth mentioning that the lower bound on Erdös-Rényi graphs can be extended to random power law graphs with $\beta > 3$. The proof structure is similar because the degree distribution has finite variance, hence the number of high degree vertices is small. The difference corresponds to technical modifications which deal with the neighborhood growth of random graphs with constant average degree. We state the result below and leave the proof to the full version.

Theorem 8. *Let $G = (V, E)$ a random power law graph with average degree $\nu > 1$ and exponent $\beta > 3$. With high probability over the randomness of G, any labelings which can recover all pairs distances exactly have total length at least $\tilde{\Omega}(n^{3/2})$.*

In particular, any 2-hop cover needs to store at least $\tilde{\Omega}(n^{3/2})$ many landmarks with high probability.

Lower bounds for β close to 2: Next we show that the parameter dependence of our algorithm is tight when β is close to 2. Specifically, any 2-hop cover needs to store at least $\Omega(n^{3/2-\varepsilon})$ many landmarks when $\beta = 2 + \varepsilon$. Hence it is not possible to improve over our algorithm when β is close to 2. Furthermore, the lower bound holds for the general family of labeling schemes as well.

Theorem 9. *Let $G = (V, E)$ a random power law graph with average degree $\nu > 1$ and exponent $\beta = 2 + \varepsilon$ for $\varepsilon < 1/2$. With high probability over the randomness of G, any labelings which can recover all pairs distances exactly have total length at least $\Omega(n^{3/2-\varepsilon})$.*

In particular, any 2-hop cover needs to store at least $\Omega(n^{3/2-\varepsilon})$ many landmarks with high probability.

The proof is conceptually similar to Theorem 6, so we sketch the outline and leave the proof to the full version.

Let S_{high} be the set of vertices whose degrees are on the order of $\sqrt{n}$. Let S_{low} be a set of $\sqrt{n}$ vertices, where each vertex has weight between ν and 2ν. Such a set is guaranteed to exist because there are $\Theta(n)$ of them.

We first reveal all edges of G other than the ones between S_{high}. We show that at this stage, most vertices in S_{low} are more than 3 hops away from each other. If for some pair (x, y) in S_{low} whose distance is larger than three, and both x and y connect to exactly one (but different) vertex in S_{high}, then knowing whether $\text{dist}(x, y) = 3$ will reveal whether their neighbors in S_{high} are connected by an edge.

Based on the observation, we show that the total labeling length of S_{low} is at least $\tilde{\Omega}(n^{3-\beta})$. This is because the random bit between a vertex pair in S_{high} has entropy $\Omega(n^{2-\beta})$. Since there are $\Theta(n)$ pairs of vertices in S_{high}, the entropy of the labelings of S_{low} must be $\Omega(n^{3-\beta})$ (hence, its size must also be at least $\Omega(n^{3-\beta})$). Similar to Theorem 6, this argument is applied to $\sqrt{n}$ disjoint sets of "S_{low}", summing up to an overall lower bound of $\Omega(n^{7/2-\beta}) = \Omega(n^{3/2-\varepsilon})$.

7 CONCLUSIONS AND FUTURE WORK

In this work, we presented a pruning based landmark labeling algorithm. The algorithm is evaluated on a diverse collection of networks. It demonstrates improved performances compared to the baseline approaches. We also analyzed the algorithm on random power law graphs and Erdös-Rényi graphs. We showed upper and lower bounds on the number of landmarks used for Erdös-Rényi graphs and random power law graphs.

There are several possible directions for future work. One direction is to close the gap in our upper and lower bounds for random

power law graphs. We believe that any improved understanding can potentially lead to better algorithms for real world power law graphs as well. Another direction is to evaluate our approach on transportation networks, which correspond to another important domain in practice.

Acknowledgements. The authors would like to thank Fan Chung Graham, Tim Roughgarden, Amin Saberi and D. Sivakumar for useful feedback and suggestions at various stages of this work. Also, thanks to the anonymous referees for their constructive reviews. Hongyang Zhang is supported by NSF grant 1447697.

REFERENCES

[1] Ittai Abraham, Amos Fiat, Andrew V Goldberg, and Renato F Werneck. 2010. Highway dimension, shortest paths, and provably efficient algorithms. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 782–793.

[2] Ittai Abraham and Cyril Gavoille. 2011. On approximate distance labels and routing schemes with affine stretch. In *International Symposium on Distributed Computing*. Springer, 404–415.

[3] Takuya Akiba, Yoichi Iwata, and Yuichi Yoshida. 2013. Fast exact shortest-path distance queries on large networks by pruned landmark labeling. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. ACM, 349–360.

[4] Stephen Alstrup, Søren Dahlgaard, Mathias Bæk Tejs Knudsen, and Ely Porat. 2015. Sublinear distance labeling. *arXiv preprint arXiv:1507.02618* (2015).

[5] Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. 1993. On sparse spanners of weighted graphs. *Discrete & Computational Geometry* 9, 1 (1993), 81–100.

[6] Haris Angelidakis, Yury Makarychev, and Vsevolod Oparin. 2017. Algorithmic and hardness results for the hub labeling problem. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 1442–1461.

[7] Maxim Babenko, Andrew V Goldberg, Haim Kaplan, Ruslan Savchenko, and Mathias Weller. 2015. On the complexity of hub labeling. In *International Symposium on Mathematical Foundations of Computer Science*. Springer, 62–74.

[8] Bahman Bahmani and Ashish Goel. 2012. Partitioned multi-indexing: bringing order to social search. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 399–408.

[9] Hannah Bast, Daniel Delling, Andrew Goldberg, Matthias Müller-Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F Werneck. 2016. Route planning in transportation networks. In *Algorithm engineering*. Springer, 19–80.

[10] Béla Bollobás. 1998. Random graphs. In *Modern Graph Theory*. Springer, 215–252.

[11] Michele Borassi, Pierluigi Crescenzi, and Luca Trevisan. 2017. An Axiomatic and an Average-Case Analysis of Algorithms and Heuristics for Metric Properties of Graphs. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 920–939.

[12] Michele Borassi, Pierluigi Crescenzi, and Luca Trevisan. 2017. An axiomatic and an average-case analysis of algorithms and heuristics for metric properties of graphs. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 920–939.

[13] Wei Chen, Christian Sommer, Shang-Hua Teng, and Yajun Wang. 2009. Compact routing in power-law graphs. In *International Symposium on Distributed Computing*. Springer, 379–391.

[14] Fan Chung and Linyuan Lu. 2002. The average distances in random graphs with given expected degrees. *Proceedings of the National Academy of Sciences* 99, 25 (2002), 15879–15882.

[15] Fan RK Chung and Linyuan Lu. 2006. *Complex graphs and networks*. Vol. 107. American mathematical society Providence.

[16] Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. 2009. Power-law distributions in empirical data. *SIAM review* 51, 4 (2009), 661–703.

[17] Edith Cohen. 1997. Size-estimation framework with applications to transitive closure and reachability. *J. Comput. System Sci.* 55, 3 (1997), 441–453.

[18] Edith Cohen, Eran Halperin, Haim Kaplan, and Uri Zwick. 2003. Reachability and distance queries via 2-hop labels. *SIAM J. Comput.* 32, 5 (2003), 1338–1355.

[19] Atish Das Sarma, Sreenivas Gollapudi, Marc Najork, and Rina Panigrahy. 2010. A sketch-based distance oracle for web-scale graphs. In *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 401–410.

[20] Daniel Delling, Andrew V Goldberg, Thomas Pajor, and Renato F Werneck. 2014. Robust distance queries on massive networks. In *European Symposium on Algorithms*. Springer, 321–333.

[21] Daniel Delling, Andrew V Goldberg, Ruslan Savchenko, and Renato F Werneck. 2014. Hub labels: Theory and practice. In *International Symposium on Experimental Algorithms*. Springer, 259–270.

[22] Richard Durrett. 2007. *Random graph dynamics*. Vol. 200. Citeseer.

[23] Nicole Eikmeier and David F Gleich. 2017. Revisiting Power-law Distributions in Spectra of Real World Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 817–826.

[24] Mihaela Enachescu, Mei Wang, and Ashish Goel. 2008. Reducing maximum stretch in compact routing. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*. IEEE.

[25] Cyril Gavoille, Christian Glacet, Nicolas Hanusse, and David Ilcinkas. 2015. Brief Announcement: Routing the Internet with very few entries. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing*. ACM, 33–35.

[26] Cyril Gavoille, David Peleg, Stéphane Pérennes, and Ran Raz. 2001. Distance labeling in graphs. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 210–219.

[27] Andrew V Goldberg and Chris Harrelson. 2005. Computing the shortest path: A search meets graph theory. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 156–165.

[28] Andrew V Goldberg, Ilya Razenshteyn, and Ruslan Savchenko. 2013. Separating hierarchical and general hub labelings. In *International Symposium on Mathematical Foundations of Computer Science*. Springer, 469–479.

[29] Hao He, Haixun Wang, Jun Yang, and Philip S Yu. 2007. BLINKS: ranked keyword searches on graphs. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. ACM, 305–316.

[30] Minhao Jiang, Ada Wai-Chee Fu, Raymond Chi-Wing Wong, and Yanyan Xu. 2014. Hop doubling label indexing for point-to-point distance querying on scale-free networks. *Proceedings of the VLDB Endowment* 7, 12 (2014), 1203–1214.

[31] Tamara G Kolda, Ali Pinar, Todd Plantenga, and Comandur Seshadhri. 2014. A scalable generative graph model with community structure. *SIAM Journal on Scientific Computing* 36, 5 (2014), C424–C452.

[32] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. 2010. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research* 11, Feb (2010), 985–1042.

[33] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[34] Jure Leskovec, Kevin J Lang, Anirban Dasgupta, and Michael W Mahoney. 2008. Statistical properties of community structure in large social and information networks. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 695–704.

[35] Michalis Potamias, Francesco Bonchi, Carlos Castillo, and Aristides Gionis. 2009. Fast shortest path distance estimation in large networks. In *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 867–876.

[36] Christian Sommer. 2014. Shortest-path queries in static networks. *ACM Computing Surveys (CSUR)* 46, 4 (2014), 45.

[37] Mikkel Thorup and Uri Zwick. 2005. Approximate distance oracles. *Journal of the ACM (JACM)* 52, 1 (2005), 1–24.

[38] Remco Van Der Hofstad. 2009. Random graphs and complex networks. *Available on http://www. win. tue. nl/rhofstad/NotesRGCN.pdf* (2009), 11.

[39] Monique V Vieira, Bruno M Fonseca, Rodrigo Damazio, Paulo B Golgher, Davi de Castro Reis, and Berthier Ribeiro-Neto. 2007. Efficient search ranking in social networks. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM, 563–572.

[40] Sihem Amer Yahia, Michael Benedikt, Laks VS Lakshmanan, and Julia Stoyanovich. 2008. Efficient network aware search in collaborative tagging sites. *Proceedings of the VLDB Endowment* 1, 1 (2008), 710–721.

Large-Scale Talent Flow Forecast with Dynamic Latent Factor Model*

Le Zhang
Anhui Province Key Lab of Big Data Analysis and Application, University of Science and Technology of China, Baidu Talent Intelligence Center, Baidu Inc
laughing@mail.ustc.edu.cn

Chen Zhu
Baidu Talent Intelligence Center, Baidu Inc
zc3930155@gmail.com

Hengshu Zhu
Baidu Talent Intelligence Center, Baidu Inc
zhuhengshu@baidu.com

Tong Xu
Anhui Province Key Lab of Big Data Analysis and Application, University of Science and Technology of China
tongxu@ustc.edu.cn

Enhong Chen
Anhui Province Key Lab of Big Data Analysis and Application, University of Science and Technology of China
cheneh@ustc.edu.cn

Chuan Qin
Anhui Province Key Lab of Big Data Analysis and Application, University of Science and Technology of China
chuanqin0426@gmail.com

Hui Xiong
Anhui Province Key Lab of Big Data Analysis and Application, University of Science and Technology of China, Baidu Talent Intelligence Center, Baidu Inc, Business Intelligence Lab, Baidu Research National Engineering Laboratory of Deep Learning Technology an Application, China
xionghui@gmail.com

ABSTRACT

The understanding of talent flow is critical for sharpening company talent strategy to keep competitiveness in the current fast-evolving environment. Existing studies on talent flow analysis generally rely on subjective surveys. However, without large-scale quantitative studies, there are limits to deliver fine-grained predictive business insights for better talent management. To this end, in this paper, we aim to introduce a big data-driven approach for predictive talent flow analysis. Specifically, we first construct a time-aware job transition tensor by mining the large-scale job transition records of digital resumes from online professional networks (OPNs), where each entry refers to a fine-grained talent flow rate of a specific job position between two companies. Then, we design a dynamic latent factor based Evolving Tensor Factorization (ETF) model for predicting the future talent flows. In particular, a novel evolving feature by jointly considering the influence of previous talent flows and global market is introduced for modeling the evolving nature of each company. Furthermore, to improve the predictive performance, we also integrate several representative attributes of companies as side information for regulating the model inference. Finally, we conduct extensive experiments on large-scale real-world data for evaluating the model performances. The experimental results clearly validate the effectiveness of our approach compared with state-of-the-art baselines in terms of talent flow forecast. Meanwhile, the results also reveal some interesting findings on the regularity of talent flows, e.g. *Facebook becomes more and more attractive for the engineers from Google in 2016.*

CCS CONCEPTS

• **Information systems → Data mining**.

KEYWORDS

Talent Flow Forecast, Latent Factor Model, Tensor Factorization

ACM Reference Format:
Le Zhang, Chen Zhu, Hengshu Zhu, Tong Xu, Enhong Chen, Chuan Qin, and Hui Xiong. 2019. Large-Scale Talent Flow Forecast with Dynamic Latent Factor Model. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313525

1 INTRODUCTION

In the fast-evolving business environments, to ensure the competitive advantages, it becomes critical for companies to timely and consistently review their talent strategies. Therefore, talent flow

*Hui Xiong, Hengshu Zhu and Tong Xu are corresponding authors.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313525

analysis, the process of analyzing the phenomenon of job hopping among different regions or companies, always acts as a major component in market intelligence of modern companies. Indeed, with the help of talent flow analysis, many business insights and managerial initiatives can be achieved. For example, at the macro level, based on the talent flow among different countries, governments can monitor the brain drain, and carry out macroeconomic regulations for gaining and sustaining the global competitive advantage. At the micro level, companies can timely evaluate their employer brand, uncover the major competitors in the talent market, and conduct proactive talent management.

However, most of existing studies on talent flow analysis generally rely on subjective surveys, with a focus on qualitative causal inference from psychological, economic and cultural perspectives [2, 4]. Due to the lack of large-scale market data for quantitative analysis, there are limits for delivering fine-grained and predictive business insights through talent flow analysis [10, 20, 31]. Recently, the prevalence of online professional networks (OPNs) enables the accumulation of a large number of digital resumes, which contains rich information about the career path of talents. For example, as of the end of 2017, there are over 400 million resumes worldwide available at Linkedin. Indeed, these large-scale talent data provide unprecedented opportunities for conducting talent flow analysis.

To this end, in this paper, we aim to introduce a data-driven approach for fine-grained predictive talent flow analysis. Specifically, we first construct a time-aware job transition tensor by mining the large-scale job transition records of digital resumes, where each entry refers to a fine-grained talent flow rate of a specific job position between two companies, i.e., a tuple <Origin Company, Job Position, Destination Company, Time Slice>. Then, we design a dynamic latent factor based Evolving Tensor Factorization (ETF) model for predicting the future talent flows, where the origin company and the destination company are represented as the time-dependent vectors, the job position is represented as a time-independent low-dimension vector, and the time is used as the evolving signal. In particular, to model the evolving process of talent flows, we develop a job transition network based on the talent flows, where companies represent nodes, and the aggregated job transition between two companies represents the weight of edge. With this network, a novel evolving feature by jointly considering the influence of previous talent flows and global market is introduced for modeling the evolving principles of each company. Furthermore, to improve the predictive performance of ETF, we also integrate several representative attributes of companies as side information for regulating the model inference. Figure 1 illustrates the diagrammatic sketch of our approach. Finally, the extensive experiments conducted on a real-world dataset clearly validate the effectiveness of our approach compared with other state-of-the-art baselines in terms of talent flow forecast.

Specifically, the major contributions of this paper can be summarized as follows:

- We introduce a big data-driven approach for predictive talent flow analysis based on the large-scale online resume data. Our approach allows to identify fine-grained and predictive business insights.
- We propose a novel dynamic latent factor based evolving model for talent flow forecast, which consists of a novel evolving feature for jointly modeling the influence of previous talent flows and global market, and a company similarity based regularizer for enhancing model inference.
- We conduct extensive experiments on large-scale real-world data for evaluating the model performance. The experimental results not only validate the performance of our approach for talent flow forecast, but also reveal some interesting findings on the regularity of talent flows, such as *Facebook becomes more and more attractive for the engineers from Google.*

Overview. The rest of the paper is organized as follows. In Section 2, we describe the real-world data and formulate the problem. Section 3 shows the technical details of our dynamic latent factor model for talent flow forecast. Then, we introduce the experimental results in Section 4. After that, we summarize the important related works in Section 5. Finally, Section 6 concludes the work.

2 PRELIMINARIES

In this section, we first introduce the real-world dataset in our study, and then formulate the problem of fine-grained talent flow forecast.

2.1 Data Description

In this paper, we use a representative real-world dataset from one of the largest commercial OPNs, which contains more than 220 million professional resumes, to study the problem of talent flow forecast. Specifically, each resume contains a list of job experience records, where each record consists of the company name, job title with brief job description, and the working duration in a monthly granularity. An example is shown in Figure 2, more details of the dataset could be found in Section 4.

To extract the job transition records from the resumes, here we extend the methods introduced in [24] by allowing job overlaps since it is reasonable for an employee to start a new job a few months before the official resignation of her former job. Formally, a resume R contains a list of job experience records $[R_{c1}, R_{c2}, ..., R_{cn}]$, and each record R_{ci} contains a start date R^s_{ci} and an end date R^e_{ci}. Therefore, as long as the absolute difference between the end time of the former job and the start time of the successor job less than a predefined threshold θ, the transition from the former job to the successor job will be regarded as effective. Note that, in our settings, a former job may has multiple successor jobs. Furthermore, we formulate the effective job transition as a father-son relationship, and construct a career tree where each company represents a node. Intuitively, there may exist multiple paths from the root node to the leaf node, the length of which is set as the sum of corresponding work time. In particular, the longest path will be selected to represent the major career path of the employee, and the adjacent company node on the major path represents a job transition.

Specifically, we use the example shown in Figure 3 to illustrate how to extract job transition records from a resume. In the figure, we use line segments to represent work experiences and use letters to represent the corresponding company names. By checking the working durations of all the work experiences, we can get several job transitions records, including $a \rightarrow b$, $a \rightarrow c$ and $b \rightarrow c$. The corresponding career tree is shown in the right part of Figure 3.

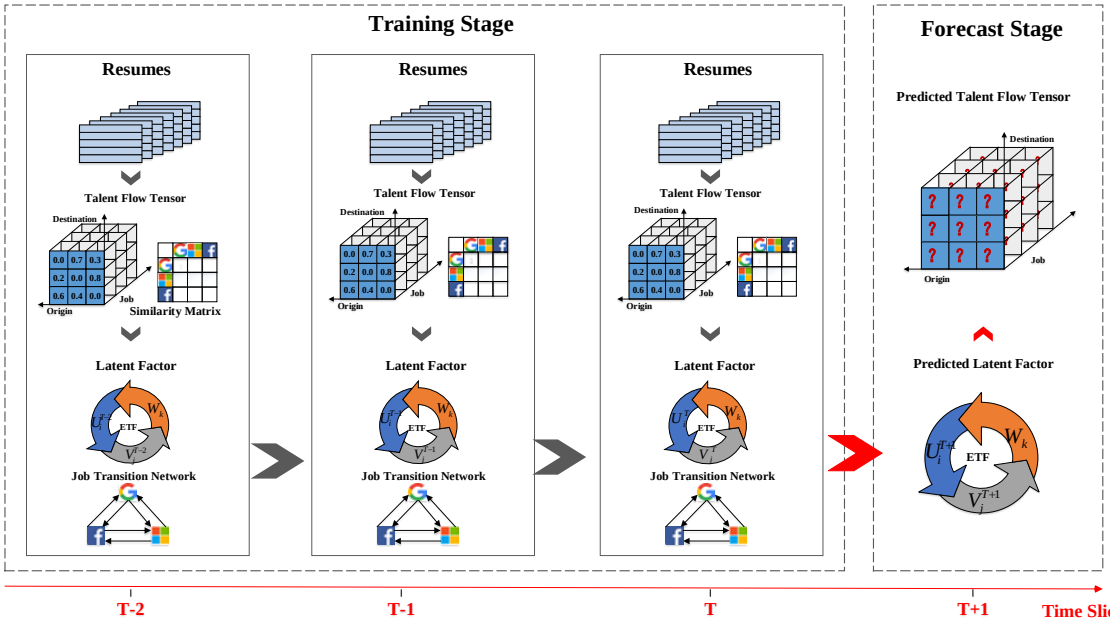


Figure 1: The diagrammatic sketch of our approach for talent flow forecast, where the dynamic latent factors are learned from the historical talent flows, and are used for forecasting the future talent flows.

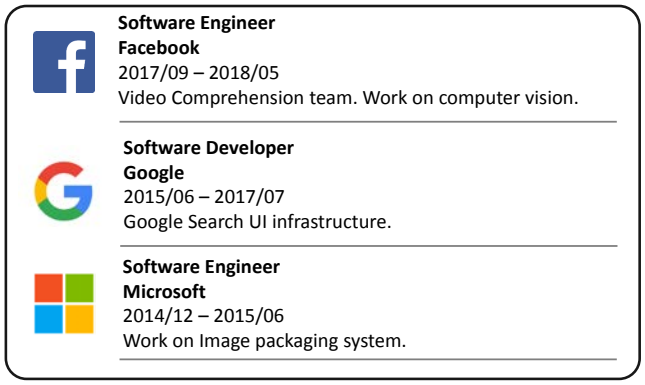


Figure 2: An example of the resume in our dataset.

Obviously, in this career tree there exist two paths, i.e., $\{a \rightarrow b \rightarrow e\}$ and $\{a \rightarrow c\}$. And because the first path is longer than the other one, we believe it is the major career path of this individual and just keep the job transitions in it, i.e., $[a \rightarrow b, b \rightarrow c]$. The job transition date is set as the start date of the successor and the job title is determined by the title of the predecessor.

2.2 Problem Formulation

Based on the extracted job transition records, here we introduce the problem formulation of talent flow forecast. For facilitating illustration, Table 1 lists some important mathematical notations used throughout this paper.

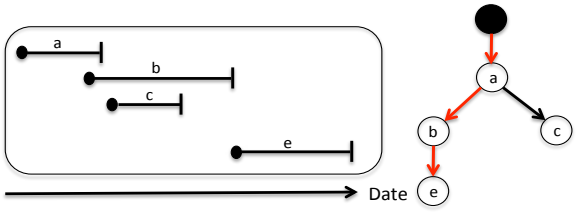


Figure 3: An example of transforming work experiences into job transition records.

Formally, we suppose there are N companies and M job positions. Based on the job transition records, we can construct talent flow tensor $\mathcal{R}^t \in \mathbb{R}^{N \times N \times M}$ for each time slice t, where each element $\mathcal{R}^t_{ijk}$ is defined as the normalized number of corresponding job transitions, i.e,

$$\mathcal{R}^t_{ijk} = \frac{Num^t_{i,j,k}}{\sum_{j=1}^{N} Num^t_{i,j,k}}, \tag{1}$$

where $Num^t_{i,j,k}$ denotes the aggregated job transitions from the job position k of company i to company j at time slice t. Along this line, the problem of talent flow forecast can be formulated as:

*Definition 2.1 (**Talent Flow Forecast**).* Given a set of talent flow tensors $\{\mathcal{R}^1, ..., \mathcal{R}^T\}$, which is constructed by the transition records as of time T, and the corresponding attributes of companies (e.g., industry, scale, and locations), the goal of talent flow forecast is to predict the value of $\mathcal{R}^{(T+1)}_{ijk}$.

Table 1: Mathematical notations.

Symbol	Description
$\mathcal{R}^t$	The talent flow tensor at time slice t;
U_i^t	The latent vector of origin company i at time slice t;
V_j^t	The latent vector of destination company j at time slice t;
W_k	The latent vector of job position k;
C	The set of companies;
S	The attribute similarity matrix of companies;
N	The number of companies;
M	The number of job position;
T	The number of time windows;
D	The dimension of latent vector.

3 TECHNICAL DETAILS

In this section, we first introduce a latent factor based model, named Basic Latent Factor based Evolving Tensor Factorization model (Basic-ETF), for talent flow forecast. Furthermore, we extend it by adding an attribute similarity based constraint for improving its performance.

3.1 Basic-ETF Model

Intuitively, given a talent flow at time slice t, i.e., $\mathcal{R}_{ijk}^t$, its value is influenced by three factors, namely talent outflow company i, talent inflow company j, and job position k. Here we represent them by latent vectors as follows:

- U_i^t represents the latent vector of the origin company i at time slice t.
- V_j^t represents the latent vector of the destination company j at time slice t.
- W_k^t represents the latent vector of the job position k at time slice t.

Without loss of generality, we assume the estimation of R_{ijk}^t equals to $f(U_i^t, V_j^t, W_k^t)$. Accordingly, the conditional distribution over the observed talent flow at time slice t is defined as

$$p(\mathcal{R}|U,V,W,\sigma) = \prod_{i=1}^{N}\prod_{j=1}^{N}\prod_{k=1}^{M}[\mathcal{N}(\mathcal{R}_{ijk}^t|f(U_i^t,V_j^t,W_k^t),\sigma^2)]^{I_{ijk}^t}, \tag{2}$$

where $\mathcal{N}(x|\mu,\sigma^2)$ is the probability density function of the Gaussian distribution with mean μ and variance σ^2, and I_{ijk}^t is the indicator function that equals to 1 if talent flow $\mathcal{R}_{ijk}^t > 0$ and equals to 0 otherwise. Specifically, we use interactions to model the relationships between talent flow R_{ijk}^t and latent vectors U_i^t, V_j^t, W_k^t. In other words, we have

$$f(U_i^t, V_j^t, W_k^t) = U_i^t V_j^t + U_i^t W_k^t + V_j^t W_k^t. \tag{3}$$

Next, we will introduce our assumptions about the time-aware relationships among U_i^t, V_j^t, and W_k^t.

As for W_k^t, in this paper, each job position represents a kind of jobs in all of time slices which play the similar function. Thus, we think it is generally time-independent and use W_k to denote the vector of job position k at all time slices. Here, we add zero-mean Gaussian priors on W, which is similar to the traditional matrix factorization [16]. Then, we have:

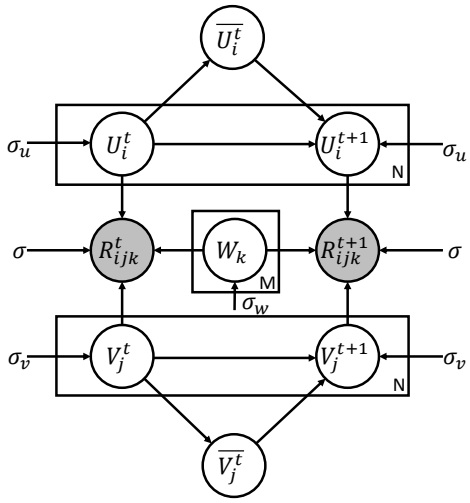


Figure 4: The graphical representation of Basic-ETF.

$$p(W|\sigma_w) = \prod_{k=1}^{M} p(W_k|\sigma_w) = \prod_{k=1}^{M} \mathcal{N}(W_k|0,\sigma_w^2 I), \tag{4}$$

where I is the D-by-D identity matrix.

As for the latent vectors of companies U_i^t and V_j^t, we think the talent preference of a company would evolve over time. Specifically, we assume there are two factors that influence companies' features in future. The first one is the **previous influence**. Intuitively, the features of company at time slice t are related to its features in the previous time slice $t-1$. In other words, the U_i^t (V_j^t) is related to $U_i^{t-1}(V_j^{t-1})$. The other one is the **market influence**, which means the company may evolve according to the general market tendency. Specifically, we assume those companies which have similar business construct the market together. And two companies are considered to have similar business when employees are very likely to move among them [3]. Therefore, here we define the business similarity of two companies based on the total number of talent transition between them. In particular, we construct a job transition network, where a node represents a company, a weighted edge represents the number of the corresponding job transitions. Then, for the target company i, at time slice t, we choose the neighbor companies that exist edges with the target company in the job transition network to represent its market, denoted by $\mathcal{M}_i^t$. And the impact of a neighbor node m to company i is denoted by P_{im}^t. In this paper, the P_{im}^t is set as follows:

$$P_{im}^t = \frac{E_{im}^t}{\sum_{m\in\mathcal{M}_i} E_{im}^t}, \tag{5}$$

where E_{im}^t denotes the total number of job transitions between company i and m. Now, we represent the market influence on the feature of origin company i by $\left(\sum_{m\in\mathcal{M}_i^{t-1}} P_{im}^{t-1} U_m^{t-1}\right)$ at time slice t, and for the destination company j by $\left(\sum_{m\in\mathcal{M}_j^{t-1}} P_{jm}^{t-1} V_m^{t-1}\right)$.

Intuitively, the companies may have their own decisions when balancing the two aspects. For example, some large companies prefer to follow their own historical state, while the small companies would like to follow the market. So we define two parameters for each company to balance the two aspects, namely α_i and β_j. Finally, the evolving process is modeled as:

$$\begin{aligned} &p(U_i^t) = \mathcal{N}(U_i^t|\bar{U}_i^t, \sigma_u^2 I), \quad p(V_j^t) = \mathcal{N}(V_j^t|\bar{V}_j^t, \sigma_v^2 I), \\ &where \quad \bar{U}_i^t = (1-\alpha_i)U_i^{t-1} + \alpha_i \sum_{m \in \mathcal{M}_i^{t-1}} P_{im}^{t-1} U_m^{t-1}, \\ &\qquad \bar{V}_j^t = (1-\beta_j)V_j^{t-1} + \beta_j \sum_{m \in \mathcal{M}_j^{t-1}} P_{jm}^{t-1} V_m^{t-1}, \\ &s.t.\ \forall i, j \in C,\ 0 \le \alpha_i \le 1,\ 0 \le \beta_j \le 1, \end{aligned} \tag{6}$$

where C denotes the company set. At the initial time $t = 1$, neither previous influence nor market influence is available. So we just assume zero-mean Gaussian distribution for the latent vectors of companies. To be specific,

$$p(U^1) = \prod_{i=1}^{N} \mathcal{N}(U_i^1|0, \sigma_u^2 I), \quad p(V^1) = \prod_{j=1}^{N} \mathcal{N}(V_j^1|0, \sigma_v^2 I). \tag{7}$$

Then, we summarize the prior over companies' latent feature matrix sequence as

$$\begin{aligned} p(U|\bar{U}, \sigma_u) &= \prod_{i=1}^{N} \mathcal{N}(U_i^1|0, \sigma_u^2 I) \prod_{t=2}^{T} \prod_{i=1}^{N} \mathcal{N}(U_i^t|\bar{U}_i^{t-1}, \sigma_u^2 I), \\ p(V|\bar{V}, \sigma_v) &= \prod_{j=1}^{N} \mathcal{N}(V_i^1|0, \sigma_v^2 I) \prod_{t=2}^{T} \prod_{j=1}^{N} \mathcal{N}(V_j^t|\bar{V}_j^{t-1}, \sigma_v^2 I). \end{aligned} \tag{8}$$

According to the graphical model shown in Figure 4, we can get the posterior distribution over U, V, W easily. To be specific,

$$\begin{aligned} &p(U, V, W|R, \sigma, \sigma_u, \sigma_v, \sigma_w) \propto \\ &p(U|\bar{U}, \sigma_u) p(V|\bar{V}, \sigma_v) p(W|\sigma_w) p(R|U, V, W, \sigma). \end{aligned} \tag{9}$$

We aim to find the value of U_i^t, V_j^t, W_k by maximizing the log-posterior of Equation 9, which is equivalent to minimizing the sum-of-squared-errors loss function:

$$\begin{aligned} \mathcal{L}_b = &\frac{1}{2} \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{M} I_{ijk}^t \left(\mathcal{R}_{ijk}^t - f(U_i^t, V_j^t, W_k) \right)^2 \\ &+ \frac{\lambda_u}{2} \sum_{t=2}^{T} \sum_{i=1}^{N} \left\| U_i^t - \bar{U}_i^{t-1} \right\|_F^2 + \frac{\lambda_v}{2} \sum_{t=2}^{T} \sum_{j=1}^{N} \left\| V_j^t - \bar{V}_j^{t-1} \right\|_F^2 \\ &+ \frac{\lambda_w}{2} \sum_{k=1}^{M} \|W_k\|_F^2 + \frac{\lambda_u}{2} \sum_{i=1}^{N} \left\| U_i^1 \right\|_F^2 + \frac{\lambda_v}{2} \sum_{j=1}^{N} \left\| V_j^1 \right\|_F^2, \\ &s.t.\ \forall i, j \in C, 0 \le \alpha_i \le 1, 0 \le \beta_j \le 1, \end{aligned} \tag{10}$$

where $\lambda_u = \sigma^2/\sigma_u^2$, $\lambda_v = \sigma^2/\sigma_v^2$, $\lambda_w = \sigma^2/\sigma_w^2$ and $\|\cdot\|_F^2$ denotes the Frobenius norm.

3.2 Company Attribute Constraint

The basic model only considers the numerical job transition information, here we integrate some extra attribute information of companies into the basic model to further refine the performance of talent flow forecast, and the complete model is called ETF.

Specifically, first we extract some representative attributes for each company, which contain company's industry, scale, location, specialties, type and age. Next, these attributes of a company are integrated as a vector by using one hot encoding. To estimate the attribute similarity among companies, we choose the Cosine similarity of the corresponding vectors. Finally, we use a constraint to model the similarity between companies. We formulate the **Company Similarity Regularizer** as

$$\begin{aligned} \mathcal{L}c &= \sum_{t=1}^{T} \left(\sum_{i=1}^{N} \sum_{i'=1}^{N} S(i,i') \left\| U_i^t - U_{i'}^t \right\|_F^2 + \sum_{j=1}^{N} \sum_{j'=1}^{N} S(j,j') \left\| V_j^t - V_{j'}^t \right\|_F^2 \right) \\ &= \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{i'=1}^{N} \sum_{d=1}^{D} S(i,i') \left(U_i^t(d)^2 - U_i^t(d) U_{i'}^t(d) \right) \\ &+ \sum_{t=1}^{T} \sum_{j=1}^{N} \sum_{j'=1}^{N} \sum_{d=1}^{D} S(j,j') \left(V_j^t(d)^2 - V_j^t(d) V_{j'}^t(d) \right) \\ &= \sum_{t=1}^{T} \sum_{d=1}^{D} \left(U^t(d)^\top (\bar{S} - S) U^t(d) + V^t(d)^\top (\bar{S} - S) V^t(d) \right) \\ &= \sum_{t=1}^{T} \left(tr\left((U^t)^\top (\bar{S} - S) U^t \right) + tr\left((V^t)^\top (\bar{S} - S) V^t \right) \right), \end{aligned} \tag{11}$$

where $tr(\cdot)$ represents the matrix trace, and $S(i,i')$ is the attribute similarity between the origin company i and i'. $\bar{S}$ is the degree matrix of S, which is defined as

$$\bar{S} = \begin{cases} \sum_{i=1}^{N} S(i,i'), & i = i', \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

Here, we use the company attribute similarity matrices S to regularize the learning process of company latent matrix U^t and V^t, which guarantees the components of U^t (or V^t) should be similar if their corresponding company attributes are similar. By integrating the extra attribute information of companies into the basic model, we can obtain the final optimization objective of our model. To be specific, we have

$$min\mathcal{L} = min(\mathcal{L}_b + \lambda_c \mathcal{L}_c), \tag{13}$$

where λ_c denotes the parameter of company similarity constraint.

3.3 Model Learning and Forecasting

The coupling between U, V, W and the balance parameters make the above loss function $\mathcal{L}$ not-convex. However, a local minimum could be achieved by performing gradient descent on each parameter iteratively. Here, we introduce how to use the gradient descent approach to learn our model. The partial derivative of the balancing

Algorithm 1 Parameter Learning of the ETF

Require: $\mathcal{R} = \{\mathcal{R}^1, \mathcal{R}^2, ..., \mathcal{R}^T\}$, attribute matrix F, the learning rate η, the hypeparemeters $\lambda_u, \lambda_v, \lambda_w, \lambda_c$
Ensure: U, V, W, α and β
1: Initialize U, V, W, α and β
2: Calculate the attribute similarity matrix S base on F
3: Calculate the corresponding degree matrix $\bar{S}$ of S
4: **while** Not Converged **do**
5: **for** each (i, j, k, t) in the $\mathcal{R}$ **do**
6: Update latent vector W_k with Equation 15
7: Update latent vector U_i^t with Equation 16
8: Update latent vector V_j^t with Equation 17
9: Update bias vector α_i and β_j according to Equation 18
10: **end for**
11: **end while**

coefficient α_i and β_j are:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \alpha_i} &= \lambda_u \sum_{t=2}^{T}\sum_{i=1}^{N}(U_i^t - \bar{U}_i^{t-1})\left(\sum_{m\in\mathcal{M}_i^{t-1}} P_{im}^{t-1}U_m^t - U_i^{t-1}\right),\\ \frac{\partial \mathcal{L}}{\partial \beta_j} &= \lambda_v \sum_{t=2}^{T}\sum_{j=1}^{N}(V_j^t - \bar{V}_j^{t-1})\left(\sum_{m\in\mathcal{M}_j^{t-1}} P_{jm}^{t-1}V_m^t - V_j^{t-1}\right).\end{aligned} \tag{14}$$

The derivatives of job position vector W_k, the origin company's latent vector U_i^t, and destination company's vector V_j^t are

$$\frac{\partial \mathcal{L}}{\partial W_k} = \sum_{i=1}^{N}\sum_{j=1}^{N} I_{ijk}^t\left(\mathcal{R}_{ijk}^t - f(U_i^t, V_j^t, W_k)\right)(U_i^t + V_j^t) + \lambda_w W_k, \tag{15}$$

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial U_i^t} = &\sum_{j=1}^{N}\sum_{k=1}^{M} I_{ijk}^t\left(\mathcal{R}_{ijk}^t - f(U_i^t, V_j^t, W_k)\right)(V_j^t + W_k)\\ &+ \mathcal{I}[t=1]\cdot\lambda_u U_i^1 + I[t\geq 2]\cdot\lambda_u(U_i^t - \bar{U}_i^t)\\ &+ \lambda_u(1-\alpha_i)(\bar{U}_i^{t+1} - U_i^{t+1}) + \lambda_c(\bar{S} - S)U_i^t\\ &+ \lambda_u \sum_{i\in\mathcal{M}_m^t}\alpha_m \cdot P_{mi}^t(\bar{U}_m^{t+1} - U_m^{t+1}),\end{aligned} \tag{16}$$

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial V_j^t} = &\sum_{i=1}^{N}\sum_{k=1}^{M} I_{ijk}^t\left(\mathcal{R}_{ijk}^t - f(U_i^t, V_j^t, W_k)\right)(U_i^t + W_k)\\ &+ \mathcal{I}[t=1]\cdot\lambda_v V_j^1 + I[t\geq 2]\cdot\lambda_v(V_i^t - \bar{V}_i^t)\\ &+ \lambda_v(1-\beta_j)(\bar{V}_j^{t+1} - V_j^{t+1}) + \lambda_c(\bar{S} - S)V_j^t\\ &+ \lambda_v \sum_{i\in\mathcal{M}_m^t}\beta_m \cdot P_{mj}^t(\bar{V}_m^{t+1} - V_m^{t+1}),\end{aligned} \tag{17}$$

where $\mathcal{I}[x]$ is an indicator function that equals 1 if x is true and 0 otherwise. Because there is no constraint on U, V, and W at the updating step, we can directly update them by the Stochastic Gradient Descent (SGD) [1] method. As for α_i and β_j, they must be bounded between 0 and 1. So we employee the Projected Gradient (PG) [15] to achieve it. Specifically, by denoting the learning rate as η, the PG method updates α_i and β_j in $l+1$-iteration by the following rules:

$$\begin{aligned}\alpha_i^{l+1} &\leftarrow \mathcal{P}\left(\alpha_i^l + \eta\lambda_u(U_i^t - \bar{U}_i^t)(\sum_{m\in\mathcal{M}_i^{t-1}} P_{im}^{t-1}U_m^t - U_i^{t-1})\right),\\ \beta_j^{l+1} &\leftarrow \mathcal{P}\left(\beta_j^l + \eta\lambda_v(V_j^t - \bar{V}_j^t)(\sum_{m\in\mathcal{M}_j^{t-1}} P_{jm}^{t-1}V_m^t - V_j^{t-1})\right),\end{aligned} \tag{18}$$

where the function $\mathcal{P}(\cdot)$ is defined as follows:

$$\mathcal{P}(x) = \begin{cases}0, & \text{if } x \leq 0,\\ \alpha_i, & \text{if } 0 \leq x \leq 1,\\ 1, & \text{if } x \geq 1.\end{cases} \tag{19}$$

In conclusion, Algorithm 1 describes the optimization process of the ETF model. After learning the related parameters U, V, W, α, and β, we can forecast the $\mathcal{R}_{ijk}^{T+1}$ by:

$$\begin{aligned}U_i^{T+1} &\approx U_i^T + \alpha_i \sum_{m\in\mathcal{M}_i^T} P_{im}^T U_m^T,\\ V_j^{T+1} &\approx V_j^T + \beta_j \sum_{m\in\mathcal{M}_j^T} P_{jm}^T V_m^T,\\ \mathcal{R}_{ijk}^{T+1} &\approx f(U_i^{T+1}, V_j^{T+1}, W_k).\end{aligned} \tag{20}$$

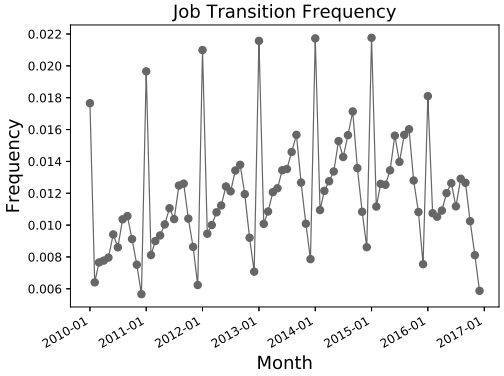


Figure 5: The normalized frequency of job transitions over different months.

4 EXPERIMENTS

In this section, we introduce the experimental results for validating the effectiveness of our approach.

4.1 Data Pre-processing

There are three main steps of data pre-processing, namely *data filtering*, *job title categorization* and *Data Spliting*.

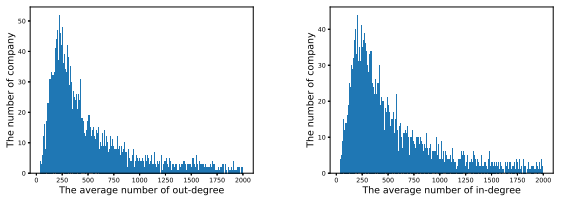
Data Filtering. In our real-world dataset, there are 670,260 unique companies. To avoid noise, we removed the companies that appeared in less than 1,000 resumes, and removed the resumes that contained only one company in the experience records (i.e., without job-hopping). After that, we finally retained 4,567 target

Table 2: Statistics of our dataset.

The number of resumes	2,176,157
The number of companies	4,567
The number of transition records	5,875,626

companies for experiments. Meanwhile, as shown in Figure 5, the frequency of job transition has a clear cycle of 12 months. Therefore, in the experiments, we set the length of time window as one year. Moreover, we also find that the frequency of job transition grows steadily before January 2016, while the frequency dropped significantly after then. It is because that, in most cases, the users in OPNs do not update their profiles immediately after changing the job. Therefore, the closer the date of job transition is to the time of data collection (e.g., January 2017 for our dataset), the more incomplete data is obtained. To guarantee the effectiveness of evaluation, we only selected the job transition records between January 2010 and January 2016 for experiments. The detailed statistics of our pre-processed dataset are shown in Table 2. In addition, Figure 6 shows the distribution of the number of companies with respect to different average number of out/in-flow of talents each year. From the results, we can find that both distributions are unbalanced, and most of the companies only have limited rate of talent flows, while only few companies have high rate of talent flows.

Job Title Categorization. In our dataset, there are total 448,309 unique job titles, which were filled in by users without specification. In our experiments, to normalize the job titles, we used an online API called MonkeyLearn [1] to categorize the original job titles into 26 classes according to the job descriptions written in the resumes. The detailed results of job title categorization in our dataset is shown in Table 3.



(a) Average Number of Out-degree (b) Average Number of In-degree

Figure 6: The distributions of the number of companies w.r.t the average number of in/out-flows of talents each year.

Data Spliting. After the above data-processing, we further standardized the date of job transitions into different time slices with year-based granularity. In the evaluation, we split four experimental datasets according to the time slice, the statistics of which are shown in Table 4. Specifically, we used the data as of time slice T for model training in the data splitting process, e.g., T=5 (T=3) in DATA10-15 (DATA12-15), and the data at the last time slice for test, e.g., T=6 (T=4) in DATA10-15 (DATA12-15). Finally, to initialize the talent flow tensor, we aggregated all transition records by the origin company, destination company, job category and time slice simultaneously.

[1]https://app.monkeylearn.com/main/classifiers

4.2 Evaluation on Talent Flow Forecast

In this subsection, we introduce the evaluation of our approach in talent flow forecast.

Baselines. To evaluate the performance of our model in talent flow forecast, we selected a number of state-of-art methods as baselines. Specifically, we first chose two popular tensor factorization based approaches, and several latent factor based evolutionary approaches. Then, to further validate the performance of each component in our model, we also designed some simplified variants of our model.

- **CP (CANDECOMP/PARAFAC)** [11], which decomposes the tensor into same rank of latent factors, i.e., an n-dimensional tensor is factorized into a sum of outer products of n vectors.
- **Tucker** [5], which considers the tensor factorization problem as the high order PCA, i.e., an n-dimensional tensor is factorized into a small core tensor and n matrices.
- **BPTF** [22], which extends PMF [16] to tensor factorization for modeling the temporal relational data by creating an additional latent feature vector for each time slice.
- **CKF** [19], which models the temporal dynamics by allowing users' factors to evolve through the linear state space model.
- **BTMF** [28], which extends the conventional matrix factorization by assuming the user preferences change over time, and models the dynamics by learning a transition matrix for user latent vectors between consecutive time windows.
- **Pure-ETF**, which is a simple version of ETF, by ignoring the influence of neighbor when modeling the evolution process and the attribute information of company. It equals to minimizing the loss function defined in Equation 10 with $\alpha_i = \beta_j = 0$, for each $i, j \in C$.
- **Basic-ETF**, which is a simplified version of ETF, by considering the evolutionary process with loss function defined in Equation 10.

Evaluation Metrics. In our experiments, we chose two metrics for evaluating the performance of talent forecast, namely Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). Specifically, the two metrics are defined as

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_i - \hat{y}_i)^2}, \tag{21}$$

$$MAE(y, \hat{y}) = \frac{1}{n}\sum_{t=1}^{n}|y_i - \hat{y}_i|, \tag{22}$$

where y_i is the actual value, while $\hat{y}_i$ is the estimated value, and n is the number of test instances. Generally speaking, the best prediction has a value of 0 of both RMSE and MAE. And the lower values of these metrics are, the better forecast is obtained. Both metrics were selected since they are widely adopted in evaluating forecasting models [9]. Meanwhile, the RMSE is a part of our loss function (i.e., defined in Equation 10), so that it is suitable to validate whether our model is over-fitting in the training stage.

Evaluation Setup. In our experiments, each method was evaluated at four datasets, as shown in Table 4. Specifically, 10% records

Table 3: Job title category and distribution

Category	# Title	# Transition	Category	# Title	# Transition
Accounting	50,543	565,151	Arts-and-design	25,931	310,469
Administrative	26,465	316,872	Business-development	44,158	488,703
Consulting	51,399	495,402	Community-and-social-services	10,902	128,140
Engineering	13,133	157,250	Education	43,518	491,046
Entrepreneurship	15,726	158,285	Finance	14,649	175,400
Healthcare-services	1,284	95,383	Human Resources	4,290	81,368
Information-technology	3,230	58,675	Legal	10,834	129,715
Military-and-protective-services	1,626	79,463	Marketing	6,646	79,576
Media-and-communication	8,819	105,592	Operations	8,815	105,545
Program-and-project-management	26,940	322,550	Purchasing	11,940	142,961
Product-management	27,127	336,772	Quality-assurance	9,348	111,934
Research	2,566	60,726	Real-estate	10,974	131,394
Sales	1,836	50,014	Support	15,803	189,215

Table 4: The Statistics of Experimental Dataset

DataSet	Density	# Training	# Test
DATA10-15	9.5787 ‰	3,116,686	514,582
DATA11-15	9.2922 ‰	2,519,555	514,582
DATA12-15	8.6874 ‰	1,884,447	514,582
DATA13-15	7.6677 ‰	1,247,452	514,582

of the training data were selected randomly as validation data for parameter tuning. The rest part of training data were used to train models. And the final performance was evaluated on the test set.

For the tensor factorization based approaches to forecast, such as CP and Tucker, we trained these models on the training tensors directly. After getting the representation of each time slice in the training data, the average representation of each time slice, such as $1, 2, \ldots, T$, was used to denote the representation of $T+1$ [7]. Finally, we could predict the value at time $T+1$ based on the specific decomposition strategy of each method. For the evolution based approaches, such as BPTF, CKF and BTMF, which only consider two dimensions (e.g., user- and item-dimension) at a specific time, we changed the form of our dataset. Specifically, we merged the `Original Company` and `Job Position` dimensions of our tensor at time t to construct an extended matrix. Then in our experiments, we treated the job position of a original company as "user" and treated the destination company as the "item", for executing each method. In particular, all the parameters of models were tuned in the validation set to ensure the best performance.

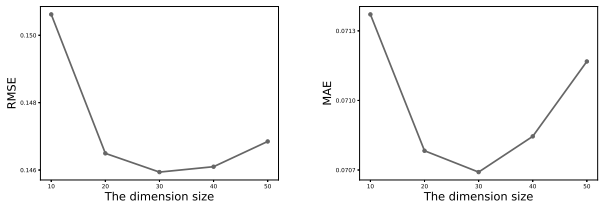
Overall Performance. The overall RMSE and MAE results of different approaches are shown in Table 5 and 6 respectively. Obviously, our ETF method achieves the better performance than all the baselines in all cases. And from the results, we can have the following observations. First, the tensor factorization based models always get the worst performance, which is because they treat time as a separate element and ignore the evolutionary relationship of latent factors. Second, the Pure-ETF model, which is the simplified version of ETF achieves the better performance than the latent factor based evolutionary models. i.e., BPTF, CKF and BTMF. This

Table 5: The RMSE performance of each model.

Model	DATA10-15	DATA11-15	DATA12-15	DATA13-15
CP	0.251807	0.251775	0.251941	0.252031
Tucker	0.251866	0.251922	0.252118	0.252291
BPTF	0.191331	0.195110	0.208030	0.232327
CKF	0.180299	0.185846	0.194905	0.209273
BTMF	0.180384	0.187388	0.197022	0.211710
Pure-ETF	0.159148	0.161018	0.163138	0.163533
Basic-ETF	0.146928	0.148899	0.150921	0.156031
ETF	**0.145037**	**0.146495**	**0.150856**	**0.148303**

Table 6: The MAE performance of each model.

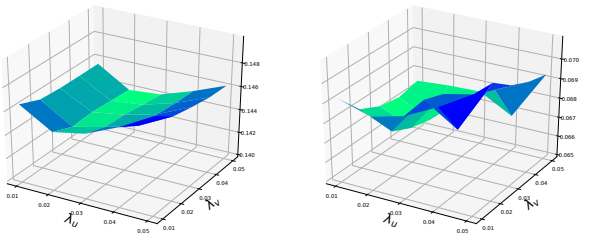
Model	DATA10-15	DATA11-15	DATA12-15	DATA13-15
CP	0.131997	0.131939	0.132112	0.132262
Tucker	0.132012	0.132075	0.132318	0.132572
BPTF	0.092110	0.098019	0.104780	0.118702
CKF	0.086739	0.090773	0.096204	0.104698
BTMF	0.087472	0.091340	0.096623	0.104959
Pure-ETF	0.079467	0.081352	0.083581	0.084650
Basic-ETF	0.072298	0.073935	0.077193	0.086700
ETF	**0.067981**	**0.069183**	**0.071247**	**0.073079**



(a) RMSE (b) MAE

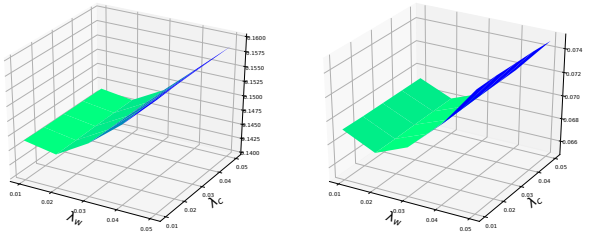
Figure 7: The evaluation on the impact of the dimension size.

means considering the interaction of three factors is more effective than only considering two factors. Third, Basic-ETF outperforms Pure-ETF, which means that considering the market influence in evolution process is effective. Meanwhile, we find the attribute similarity based constraints can improve the model by 1% at least



(a) RMSE (b) MAE

Figure 8: The evaluation on the impact of the λ_u and λ_v.



(a) RMSE (b) MAE

Figure 9: The evaluation on the impact of the λ_w and λ_c.

in RMSE metric and 5% at least in MAE. Finally, the prediction performance of methods which model the evolutionary process is related to the length of time slices. The longer the length of time slices is, the better performance is obtained. It may because that the longer time slices can help to capture the regular pattern of evolution. Meanwhile the performance of ETF drops slowly than all baselines as the time slices decrease, which validates the robustness of our model.

4.3 Evaluation on Parameter Sensitivity

Here, we turn to discuss the sensitivity of parameters in our approach. Specifically, several parameters will be discussed, namely the dimension of latent factor D, the coefficient of original company vector λ_u, the coefficient of destination company vector λ_v, the coefficient of job vector λ_w and the coefficient of company similarity constraints λ_c. All the experiments were conducted on DATA11-15, and we executed the model for three times and used the average of result to represent the final performance.

First, we discuss the sensitiveness of the dimension of latent factor D, which is summarized in Figure 7. By setting D from 10 to 50, we find that the generally the increasing D will improve the performance, which could be reasonable as more dimensions may probably keep more useful information. However, the performance of the model drops a little, which may because that the more parameters cause the model over-fitting at the training stage. Second, we discuss the sensitiveness of the evolving parameter λ_u and λ_v, which is shown in Figure 8. By setting parameters from 0.01 to 0.05, we find that the model gets the best performance when $\lambda_u = 0.02$ and $\lambda_v = 0.04$ on both metrics. Finally, we set λ_w and λ_c from 0.01 to 0.05. As the result shown in Figure 9, we find that the model gets the best performance when $\lambda_w = 0.02$ and $\lambda_c = 0.02$.

4.4 Case Study

Here we introduce a case study of talent flow forecast for further validating our approach. Specifically, we trained the ETF model based on the job transition records from January 2010 to December 2015, and forecasted the out-flow for engineers (i.e., employees with engineering related jobs) of Google in 2016 by Equation 20. The ground truth results in 2015/2016 and the forecast results in 2016 are all shown in Figure 10, where the size of words are proportional to the number of talent flows (i.e., more attractive). Based on the results, we can obtain two findings. First, comparing with the ground truth in 2016, the forecast results have very similar distributions, which clearly validates the performance of our approach. Second, comparing the ground truth in 2015 and 2016, we can find that the distributions of companies have obvious difference, which indicates the dynamic characteristics of talent flows. For example, Facebook becomes more and more attractive for the engineers from Google. Indeed, our approach can clearly capture such change of talent flows, which validates the effectiveness of using dynamic latent factors for modeling the evolving of talent flows.

5 RELATED WORK

In this section, we will summarize the related works in the following three categories, namely the talent flow analysis, time-dependent collaboration filtering and tensor factorization.

Talent Flow Analysis. As a crucial issue of human resource management, talent management has attracted wide attention in the past decade. On the one hand, the studies of talent flow analysis, especially those designed by researchers with sociology background, mainly discussed the potential causes of talent flows. For instance, at the national level, [2] and [20] proposed several factors that might influence the decision to migrate, such as economic, family and so on. Besides, a few studies investigated the national talent flow issues with the consideration of "brain drain" in several countries [2], [10], [31]. In addition to these macro-level talent flow analysis, [8] analyzed the factors for influencing individual job hopping, such as the salary level, job satisfaction and so on. On the other hand, with the prevalence of online professional networks (OPNs), a large number of digital resumes of talents can be collected. With these kind of data, there emerges a tend of conducting talent-related studies with machine learning methods. For instance, [21] and [23] predicted the individual job changes for understanding the professional careers of talents. [27] aimed to find the professional similarity between talents. [3] predicted the salary at company level by using talent transition data. [24] proposed the concept of job transition network (JTN) and extracted talent circle for talent recruitment. Recently, [25] extended the JTN with a dynamic setting and focused on the prediction of edge weights in dynamic JTN. Different from the above literatures, in this paper, we aim to introduce a big data-driven approach for predictive talent flow analysis, i.e., forecasting the future talent flows in a fine-grained manner.

Time-Dependent Collaborative Filtering. Latent factor model are widely used [29] [30], and the technical solution of this paper follows the idea of the model based collaborative filtering, which maps both preferences of users and items into the low-dimensional vector space, and conducts recommendations based on the learned latent factors [13]. However, since user preferences usually change



(a) Ground Truth-2015 (b) Ground Truth-2016. (c) Forecast-2016.

Figure 10: (a)-(b) The ground truth of the most attractive companies to engineers of Google in 2015 and 2016. (c) The forecast of the most attractive companies to engineers of Google in 2016 by our model.

over time, it is important to take the dynamics of preferences into account [6] [18] [26]. For instance, [12] proposed the TimeSVD++, which modeled the temporal dynamics of user latent vectors by summarizing three factors. [14] assumed the users' preferences would change smoothly over time, and the current preference of a user was the prior to future preference. [28] proposed the BTMF, which modeled the dynamics by learning the transition matrix for each user's latent vector between consecutive time windows. [19] modeled the dynamics of user's latent vector through a linear state space model. In fact, it is equal to learning a shared transition matrix for all users' latent factors evolving while each user has a specific bias evolving vector. In this paper, we also try to learn the vectors for three components, i.e., origin company, destination company and job position. Meanwhile, we assume the vectors of companies are time-dependent. Different from the above works, we model the evolving process of companies' vector by considering two kinds of influence, i.e., previous influence and market influence, which is more suitable for the scenario of talent flow forecast.

Tensor Factorization. There are two kinds of popular tensor factorization models, namely the CANDECOMP/PARAFAC (CP) model [11] and the Tucker model [5]. Specifically, the CP model factorizes a tensor into the sum of outer products of n vectors, and the Tucker model factorized an n-dimensional tensor into a small core tensor and n matrices. In the literatures, there are some works on integrating the temporal information for tensor factorization. For instance, [22] created an additional latent feature vector for each time slice and modeled each entry of the tensor as the inner product of latent factors of user, item and time slice. [17] defined a new measure of user preference dynamics to capture the shifting rate for each user. And it further modeled users' preference dynamics and users' side information in coupled tensor factorization. [7] considered the bipartite graphs that evolved over time and demonstrated that tensor based methods were effective for temporal data with varying periodic patterns. Different from these methods, in this paper, we propose a novel approach to model the evolution of talent flows, instead of learning the vector of time slice directly.

6 CONCLUSION

In this paper, we studied the problem of talent flow forecast from a data-driven perspective. Specifically, we first constructed a time-aware job transition tensor by mining the large-scale job transition records of digital resumes in online professional networks (OPNs). Then, we designed a dynamic latent factor based Evolving Tensor Factorization (ETF) model for predicting the future talent flows. In particular, a novel evolving feature by jointly considering the influence of previous talent flows and global market was introduced for modeling the evolving principles of each company. Furthermore, to improve the predictive performance of ETF, we also integrated the company similarity as side information for regulating the model inference. Finally, extensive experiments conducted on a large-scale real-world dataset clearly validate the effectiveness of our approach.

7 ACKNOWLEDGMENTS

This research was partially supported by grants from the National Natural Science Foundation of China (Grant No. 91746301, U1605251, 71531001 and 61703386), the Anhui Provincial Natural Science Foundation (Grant No. 1708085QF140) and K.C.Wong Education Foundation.

REFERENCES

[1] Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer, 177–186.
[2] Stuart C Carr, Kerr Inkson, and Kaye Thorn. 2005. From global careers to talent flow: Reinterpreting ?brain drain? *Journal of World Business* 40, 4 (2005), 386–398.
[3] Xi Chen, Yiqun Liu, Liang Zhang, and Krishnaram Kenthapadi. 2018. How LinkedIn Economic Graph Bonds Information and Product: Applications in LinkedIn Salary. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 120–129.
[4] Thomas H Davenport, Jeanne Harris, and Jeremy Shapiro. 2010. Competing on talent analytics. *Harvard business review* 88, 10 (2010), 52–58.
[5] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. 2000. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications* 21, 4 (2000), 1253–1278.
[6] Yi Ding and Xue Li. 2005. Time weight collaborative filtering. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 485–492.
[7] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. 2011. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 5, 2 (2011), 10.
[8] Daniel C Feldman and Thomas WH Ng. 2007. Careers: Mobility, embeddedness, and success. *Journal of management* 33, 3 (2007), 350–377.
[9] Tilmann Gneiting. 2011. Making and evaluating point forecasts. *J. Amer. Statist. Assoc.* 106, 494 (2011), 746–762.
[10] Kerr Inkson. 2005. Exploring the dynamics of New Zealand's talent flow. *New Zealand Journal of Psychology* 34, 2 (2005), 110–116.
[11] Tamara G Kolda and Brett W Bader. 2009. Tensor decompositions and applications. *SIAM review* 51, 3 (2009), 455–500.
[12] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 447–456.

[13] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[14] Ruijiang Li, Bin Li, Cheng Jin, Xiangyang Xue, and Xingquan Zhu. 2011. Tracking User-Preference Varying Speed in Collaborative Filtering.. In *AAAI*.

[15] Chih-Jen Lin. 2007. Projected gradient methods for nonnegative matrix factorization. *Neural computation* 19, 10 (2007), 2756–2779.

[16] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.

[17] Dimitrios Rafailidis and Alexandros Nanopoulos. 2014. Modeling the dynamics of user preferences in coupled tensor factorization. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 321–324.

[18] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 811–820.

[19] John Z Sun, Dhruv Parthasarathy, and Kush R Varshney. 2014. Collaborative Kalman Filtering for Dynamic Matrix Factorization. *IEEE Trans. Signal Processing* 62, 14 (2014), 3499–3509.

[20] Ibraiz Tarique and Randall S Schuler. 2010. Global talent management: Literature review, integrative framework, and suggestions for further research. *Journal of world business* 45, 2 (2010), 122–133.

[21] Jian Wang, Yi Zhang, Christian Posse, and Anmol Bhasin. 2013. Is it time for a career switch?. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1377–1388.

[22] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G Carbonell. 2010. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of the 2010 SIAM Internationxal Conference on Data Mining*. SIAM, 211–222.

[23] Huang Xu, Zhiwen Yu, Hui Xiong, Bin Guo, and Hengshu Zhu. 2015. Learning career mobility and human activity patterns for job change analysis. In *2015 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1057–1062.

[24] Huang Xu, Zhiwen Yu, Jingyuan Yang, Hui Xiong, and Hengshu Zhu. 2016. Talent circle detection in job transition networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 655–664.

[25] Huang Xu, Zhiwen Yu, Jingyuan Yang, Hui Xiong, and Hengshu Zhu. 2018. Dynamic Talent Flow Analysis with Deep Sequence Prediction Modeling. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[26] Tong Xu, Hengshu Zhu, Hao Zhong, Guannan Liu, Hui Xiong, and Enhong Chen. 2018. Exploiting the Dynamic Mutual Influence for Predicting Social Event Participation. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[27] Ye Xu, Zang Li, Abhishek Gupta, Ahmet Bugdayci, and Anmol Bhasin. 2014. Modeling professional similarity by mining professional career trajectories. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1945–1954.

[28] Chenyi Zhang, Ke Wang, Hongkun Yu, Jianling Sun, and Ee-Peng Lim. 2014. Latent factor transition for dynamic collaborative filtering. In *Proceedings of the 2014 SIAM International Conference on Data Mining*. SIAM, 452–460.

[29] Chen Zhu, Hengshu Zhu, Yong Ge, Enhong Chen, and Qi Liu. 2014. Tracking the evolution of social emotions: A time-aware topic modeling perspective. In *2014 IEEE International Conference on Data Mining*. IEEE, 697–706.

[30] Chen Zhu, Hengshu Zhu, Hui Xiong, Pengliang Ding, and Fang Xie. 2016. Recruitment market trend analysis with sequential latent variable models. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 383–392.

[31] David Zweig. 2006. Competing for talent: China's strategies to reverse the brain drain. *International labour review* 145, 1-2 (2006), 65–90.

From Stances' Imbalance to Their Hierarchical Representation and Detection

Qiang Zhang
University College London
London, United Kingdom
qiang.zhang.16@ucl.ac.uk

Shangsong Liang
Sun Yat-sen University
Guangzhou, China
liangshangsong@gmail.com

Aldo Lipani
University College London
London, United Kingdom
aldo.lipani@ucl.ac.uk

Zhaochun Ren
Shandong University
Qingdao, China
zhaochun.ren@sdu.edu.cn

Emine Yilmaz
University College London
London, United Kingdom
emine.yilmaz@ucl.ac.uk

ABSTRACT

Stance detection has gained increasing interest from the research community due to its importance for fake news detection. The goal of stance detection is to categorize an overall position of a subject towards an object into one of the four classes: *agree*, *disagree*, *discuss*, and *unrelated*. One of the major problems faced by current machine learning models used for stance detection is caused by a severe class imbalance among these classes. Hence, most models fail to correctly classify instances that fall into minority classes. In this paper, we address this problem by proposing a hierarchical representation of these classes, which combines the agree, disagree, and discuss classes under a new *related* class. Further, we propose a two-layer neural network that learns from this hierarchical representation and controls the error propagation between the two layers using the Maximum Mean Discrepancy regularizer. Compared with conventional four-way classifiers, this model has two advantages: (1) the hierarchical architecture mitigates the class imbalance problem; (2) the regularization makes the model to better discern between the related and unrelated stances. An extensive experimentation demonstrates state-of-the-art accuracy performance of the proposed model for stance detection.

CCS CONCEPTS

• **Information systems** → **Information extraction**; **Sentiment analysis**.

KEYWORDS

hierarchical classifier, maximum mean discrepancy

ACM Reference Format:
Qiang Zhang, Shangsong Liang, Aldo Lipani, Zhaochun Ren, and Emine Yilmaz. 2019. From Stances' Imbalance to Their Hierarchical Representation and Detection. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313724

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313724

1 INTRODUCTION

The quality of online news is usually less substantiated than that of traditional news services such as magazines or newspapers [1, 45, 47]. A large volume of fake news is being produced for political or economical purposes [8, 22, 41]. Fake news are those news articles that purport to be factual, but which contain misstatements of fact with intention to arouse passions, attract viewership, or deceive [25, 37, 44]. Verifying news content needs to retrieve *evidences* and determine their *stance* with respect to the *news claims*, which proposes new challenges for the conventional stance detection task [31, 36]. We specify evidence as text, e.g. web-pages and documents, that can be used to prove if news content is or is not true. Moreover, automatic stance detection has broad applications in information retrieval and text entailment [34, 42].

The task of stance detection is to identify the stance of an evidence towards a given news claim [12, 13]. Stances can be categorized into four classes: *agree*, *disagree*, *discuss* and *unrelated* [17]. Two characteristics make the stance detection task peculiar. On the one hand, news claims and evidences are often unrelated – generating a severe class imbalance problem; On the other hand, since the non-related classes are by definition related, intuitively, the identification of an evidence as related or unrelated to a news claim is semantically different from the identification of an evidence as belonging to one of the other three classes. These two characteristics suggests the natural presence of a hierarchical structure among stance classes.

Stance detection has been studied in areas of information extraction and natural language processing [11, 40]. However, previous methods tackle the task as a multiclass classification problem, neglecting the hierarchical structure in stance classes. Also, the commonly-used four-way classifiers are easily influenced by the class imbalance problem. In this paper, we address this issue by modeling the stance detection task as a two-layer neural network. The first layer aims at identifying the relatedness of the evidence, while the second layer aims at classifying, those evidences identified as related, into the other three classes: agree, disagree and discuss. Moreover, by studying various level of dependence assumptions between the two layers: (1) independent, when there is no error propagation between the two layers; (2) dependent, when the error propagation is left free, and; (3) learned, when the error propagation is controlled by Maximum Mean Discrepancy (MMD), we

show that when learned, the neural network (a) better separates the distributions of related and unrelated stances and (b) outperforms the state-of-the-art accuracy for the stance detection task.
The remainder of the paper is organized as follows: § 2 summarizes the related work; § 3 defines the stance detection task; § 4 details the proposed hierarchical classification model and the regularization term; § 5 describes the used datasets and experimental setup; § 6 is devoted to experimental results, and; § 7 concludes the paper.

2 RELATED WORK

Machine learning techniques are widely researched to tackle the stance detection task. Previous works focus on political or congressional floor debates [11, 40, 46] and online forums [2, 19, 27, 38, 39, 42]. Most of these works rely on content-based features, such as sentiment analysis and topic-specific features learned from labeled datasets for a closed set of topics.

Two methods only consider the agree, disagree and discuss classes: Bar-Haim et al. [7] split the stance detection task to three sub-tasks and propose a Contrast Classification Algorithm to distinguish agree and disagree classes; Augenstein et al. [4] build a neural network architecture based on bidirectional conditional encoding on a Tweeter dataset. A long-short term memory (LSTM) encodes the claim and another LSTM encodes the text with the encoded claim as initial states. These methods fail to consider the unrelated class.

Two other methods consider all the classes, but use two different models: Bourgonje et al. [10] use the lemmatized n-gram matching and a rule-based procedure to decide the evidence relatedness, and a three-way logistic regression classifier to distinguish among the relevant classes; Wang et al. [43] firstly develop a gradient boosted decision tree (GBDT) model [28] to determine the evidence relatedness, then another GBDT model is used to distinguish stances of the text towards the claim. These methods involve feature engineering in separate models and cannot be jointly optimized to achieve the best performance.

Other methods that also consider all the classes have been developed during the Fake News Challenge stage 1 (FNC-1) [18]. The winner team uses a 50%/50% weighted average between a GBDT model and a convolutional neural network (CNN) [5]. The second best performance is achieved by an ensemble of five multi-layer perceptrons (MLPs) where input features include bag-of-words, semantic analysis in addition to the baseline features developed by the challenge organizers [16]. Compared to the above two solutions, the third best team does not try ensemble methods. They use TF-IDF features and an MLP as a four-way classifier [33]. Zhang et al. [48] propose a ranking method to tackle the task and achieve empirical performance improvements. However, these methods all neglect the hierarchical structure among the four types of stances and suffer from class imbalance.

Deep learning-based methods have also been applied in the FNC-1. Bajaj [6] utilizes LSTM, CNN and their variants to detect stances. Bajaj finds that an attention-augmented CNN obtains the best performance. Rakholia and Bhargava [32] analyze the effectiveness of different ways of text coding, such as independent coding, bidirectional conditional encoding and attentive readers, and conclude that the attentive reader model is the most suitable for the task. Ma et al. [23] propose a multi-task learning algorithm that jointly detect rumours and stances. However, all these methods fail to achieve high accuracy for the agree and disagree classes.

There are three major defects in all the aforementioned methods: (a) they neglect the hierarchical relationships among the four stances; (b) they suffer from the class imbalance problem, and; (c) they fail to achieve acceptable detection performance for the agree and disagree classes.

3 STANCE DETECTION TASK

The stance detection task consists in classifying the stance of an evidence towards a claim as one of the four classes: agree, disagree, discuss and unrelated. Formal definitions of these four stances are:

agree – the evidence supports the claim;
disagree – the evidence denies the claim;
discuss – the evidence does not have a position about the claim;
unrelated – the evidence is not about the claim.

4 HIERARCHICAL CLASSIFICATION

In this section, we detail our proposed two-layer neural network for stance detection. § 4.1 outlines the model. In order to better differentiate between the related and unrelated classes, we design an MMD regularization term in § 4.2. This is then integrated into the two-layer neural network loss function in § 4.3. In Figure 1, we show the architecture of our model.

4.1 Two-Layer Neural Network

Let the input space be formed by m-dimensional real vectors in a neural network, denoted as $\mathbf{v} \in \mathbb{R}^m$. The four-class label can be transformed into a one-hot vector $\mathbf{y}$. The i-dimension of $\mathbf{y}$ (y_i) is 1 when the stance is the i-element in the label set $\{agree, disagree, discuss, unrelated\}$ and 0 otherwise. The hidden layer with parameters θ_u learns to map $\mathbf{v}$ to a k-dimensional hidden representation $\mathbf{u} \in \mathbb{R}^k$:

$$\mathbf{u} = f(\mathbf{v}; \theta_u). \tag{1}$$

For the two-layer classification, the first layer decides whether the evidence is related to a claim. Hence, the first classification layer is called *the relatedness layer*. This layer is parameterized by θ_r and learns to produce a 2-dimensional normalized vector $\hat{\mathbf{r}}$ as follows:

$$\hat{\mathbf{r}} = g(\mathbf{u}; \theta_r). \tag{2}$$

Note that the Softmax function is included in g to normalize the 2-dimensional vector, so each component of the vector $\hat{\mathbf{r}}$ denotes the probability that the neural network assigns $\mathbf{v}$ to the related and unrelated classes, i.e., $p(related)$ and $p(unrelated)$.

The second layer classifies the evidence into the related classes, i.e., agree, disagree, or discuss stances. Hence, the second classification layer is called *the stance layer*. The stance layer is parameterized by θ_s and learns to produce a 3-dimensional normalized vector $\hat{\mathbf{s}}$:

$$\hat{\mathbf{s}} = h(\hat{r} \cdot (1, 0); \theta_s), \tag{3}$$

where the vector multiplication $\hat{r} \cdot (1, 0)$ extracts the first element of $\hat{\mathbf{r}}$. Note that the Softmax function is also included in h to normalize the 3-dimensional vector, so that each component of the vector $\hat{\mathbf{s}}$ denotes the conditional probability that the neural network

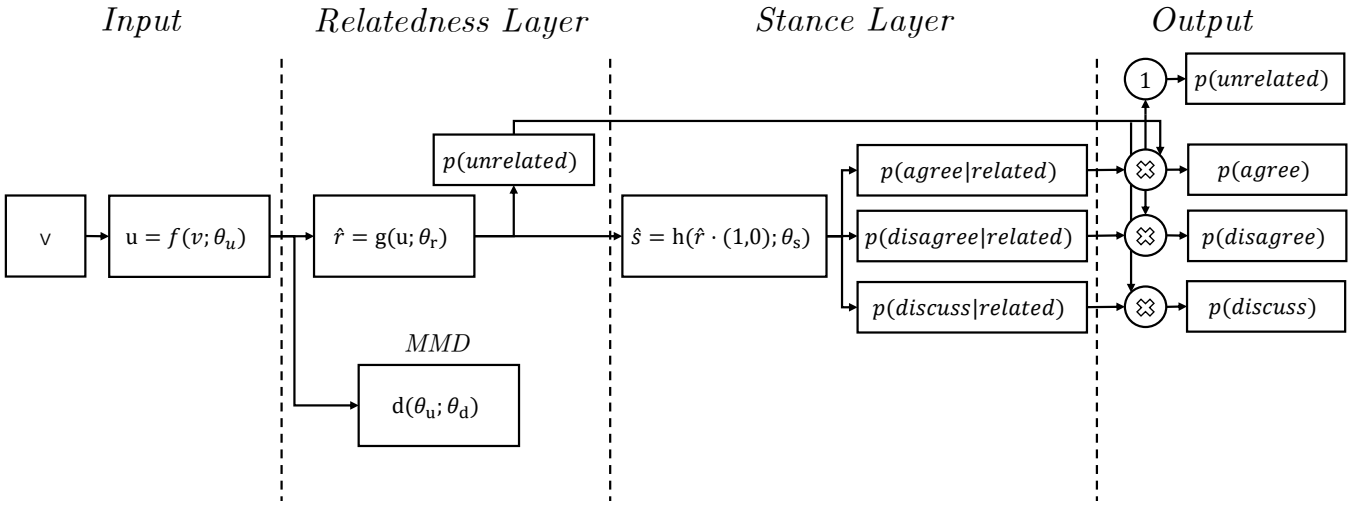


Figure 1: The architecture of our proposed two-layer neural network.

assigns $\mathbf{v}$ to agree, disagree and discuss given that $\mathbf{v}$ is related, i.e., $p(agree|related)$, $p(disagree|related)$, and $p(discuss|related)$.

We define the classification loss by the Kullback-Leibler (KL) divergence [21], which measures the difference between the network outputs and labels:

$$l^r(\theta_u, \theta_r) := \mathrm{KL}(\mathbf{r}||\hat{\mathbf{r}}), \tag{4}$$

where $\mathbf{r}$ is the ground-truth relatedness of the input data. $\mathbf{r}$ is computed from a label $\mathbf{y}$ as follows:

$$\mathbf{r} = (\mathbb{1}(\mathbf{y} \neq \mathbf{e}_4), \mathbb{1}(\mathbf{y} = \mathbf{e}_4)), \tag{5}$$

where $\mathbb{1}$ is the indicator function, $\mathbf{e}_4$ is a 4-dimensional one-hot vector with fourth element equal to 1. When $\mathbf{y} = \mathbf{e}_4$ is verified, it indicates that the label belongs to the unrelated class. Similarly, the stance classification loss can be defined as:

$$l^s(\theta_u, \theta_r, \theta_s) := \mathrm{KL}(\mathbf{s}||\hat{\mathbf{s}}), \tag{6}$$

where $\mathbf{s}$ is the ground-truth stance of the input data. $\mathbf{s}$ is computed from a label $\mathbf{y}$ as follows:

$$\mathbf{s} = (\mathbb{1}(\mathbf{y} = \mathbf{e}_1), \mathbb{1}(\mathbf{y} = \mathbf{e}_2), \mathbb{1}(\mathbf{y} = \mathbf{e}_3)), \tag{7}$$

where $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ are 4-dimensional one-hot vectors with first, second, and third elements equal to 1. When $\mathbf{y} = \mathbf{e}_1$ is verified, it indicates that the label belongs to the agree class, when $\mathbf{y} = \mathbf{e}_2$ is verified, it indicates that the label belongs to the disagree class, and when $\mathbf{y} = \mathbf{e}_3$ is verified, it indicates that the label belongs to the discuss class.

Finally, we now define the loss function for the two-layer neural network as the linear combination between the loss function of the relatedness layer (l^r) and the loss function of the stance layer (l^s):

$$l^c(\theta_u, \theta_r, \theta_s) := l^r(\theta_u, \theta_r) + \alpha \cdot l^s(\theta_u, \theta_r, \theta_s), \tag{8}$$

where α leverages the importance of the two classification layers.

4.2 Maximum Mean Discrepancy

The classification of related/unrelated stances is a different task from that of agree/disagree/discuss stances. Therefore, data representations from the relatedness layer and the stance layer can be seen as samples drawn from two different distributions. In order to measure distribution discrepancy between these two layers, we employ the Maximum Mean Discrepancy (MMD) [9] as a regularization term. The MMD does not involve density estimation and thus is a non-parametric way of measuring the difference between distributions. MMD has achieved success in face recognition and image annotation [15].

MMD is defined as follows:

Definition 4.1. Maximum Mean Discrepancy [9]: "Let p and q be two Borel probability distributions over a space $\mathcal{X}$ and let X and Z be sets with independent identically distributed samples drawn from p and q. The MMD is defined by a class Ψ of map functions ψ: $\mathcal{X} \to \mathcal{H}$ as:

$$MMD(p, q, \Psi) = \sup_{\psi \in \Psi} (\mathbf{E}_p[\psi(x)] - \mathbf{E}_q[\psi(z)]). \tag{9}$$

Here, x and z are samples from X and Z."

In other words, the MMD equation defines the largest possible distance between two expectations over the set of function Ψ. Moreover, "when $\mathcal{H}$ is the reproducing kernel Hilbert space (RKHS) [3], this means that for all $x \in \mathcal{X}$, the linear point evaluation function mapping $\psi \to \psi(x)$ exists and is continuous. When Ψ is the unit ball in a universal RKHS, it is guaranteed that $MMD(p, q, \Psi)$ will detect any discrepancy between p and q [9, 35]."

Let p denote the distribution for the first layer samples (unrelated hidden representations) in our model, with sample set $\mathcal{U}^1 = \{\mathbf{u}_1^1, \ldots, \mathbf{u}_{n_1}^1\}$ and according to Eq. (1) their generating set $\mathcal{V}^1 = \{\mathbf{v}_1^1, \ldots, \mathbf{v}_{n_1}^1\}$. And, q denotes the distribution for the second

layer samples (agree, disagree and discuss hidden representations), with sample set $\mathcal{U}^2 = \{\mathbf{u}_1^2, \ldots, \mathbf{u}_{n_2}^2\}$ and according to Eq. (1) their generating set $\mathcal{V}^2 = \{\mathbf{v}_1^2, \ldots, \mathbf{v}_{n_2}^2\}$. n_1 and n_2 are the number of samples in $\mathcal{U}^1$ and $\mathcal{U}^2$. Thus we have $\mathcal{X} = \mathcal{R}^k$ and $\mathcal{H} = \mathcal{R}^j$ with $\psi(x) = \theta_d x$, where θ_d is a $j \times k$ matrix in *the projection layer*. k and j are the space dimensions. According to Eq. (1), the hidden representation $\mathbf{u}$ is parameterized by θ_u, thus the empirical expression of MMD is parameterized by θ_u and θ_d:

$$d(\theta_u, \theta_d) = \frac{1}{n_1}\sum_{i=1}^{n_1}\theta_d \mathbf{u}_i^1 - \frac{1}{n_2}\sum_{i=n_1+1}^{n_1+n_2}\theta_d \mathbf{u}_i^2 \tag{10}$$

$$= \frac{1}{n_1}\sum_{i=1}^{n_1}\theta_d f(\mathbf{v}_i^1;\theta_u) - \frac{1}{n_2}\sum_{i=n_1+1}^{n_1+n_2}\theta_d f(\mathbf{v}_i^2;\theta_u). \tag{11}$$

By constantly changing *the projection layer* parameterized by θ_d, we find the maximum expectation difference between the representations of the two classification layers.

4.3 Optimization

The more different two distributions are, the larger the MMD is. Hence, in order to make the distributions easier to be distinguished, a larger MMD regularization term is preferred, and we treat the regularization term as an extra goal besides classification. We integrate the two-layer classification loss (see Eq. (8)) and the MMD regularization term (see Eq. (10)) into a single objective function (L). Specifically, we add these two sub-goals with a hyperparameter β as follows:

$$L(\theta_u, \theta_r, \theta_s, \theta_d) = l^c(\theta_u, \theta_r, \theta_s) - \beta \cdot d(\theta_u, \theta_d), \tag{12}$$

where β leverages the importance of the regularization. The larger the MMD regularization term is, the easier is for the classifier to distinguish between the related and unrelated stances. Thus, the sign of the regularization term is negative.

The optimization involves the minimization of the classification loss L with respect to θ_u, θ_r, θ_s, and θ_d as follows:

$$\min_{\theta_u, \theta_r, \theta_s, \theta_d} L(\theta_u, \theta_r, \theta_s, \theta_d). \tag{13}$$

Optimizing the model consists of two sub-goals. On the one hand, we want to maximize the distribution discrepancy between the two classification layers. On the other hand, we want to minimize the classification loss of both layers. Both of these two sub-goals involve the feature layer parameter θ_u update, but in opposite update directions. The optimization process will not stop until a saddle point (the feature layer parameters can be well applied in both sub-goals) is reached. Algorithm 1 shows the parameter update process, which is based on the mini-batch gradient descent algorithm.

4.4 Prediction

Given as input a feature vector $\mathbf{v}$, the classifier outputs the following probabilities: $p(unrelated)$, $p(agree|related)$, $p(disagree|related)$, and $p(discuss|related)$. However, these last 3 probabilities are not comparable with the first one. To make them comparable we derive

Algorithm 1: Parameter update process based on the mini-batch gradient descent algorithm.

input : Sample mini-batch $\{\mathbf{v}_i, \mathbf{r}_i, \mathbf{s}_i\}_{i=1}^n$, mini-batch size n, hyperparameters α, β, and μ
output: θ_u, θ_r, θ_s, θ_d

begin
  Initialize θ_u, θ_r, θ_s, θ_d;
  repeat
    /* forward propagation */
    $l^r, l^s \leftarrow 0$;
    for *i from* 1 *to n* **do**
      $\mathbf{u}_i \leftarrow f(\mathbf{v}_i;\theta_u)$;
      $\hat{\mathbf{r}}_i \leftarrow g(\mathbf{u}_i;\theta_r)$;
      $l_i^r \leftarrow \mathrm{KL}(\mathbf{r}_i \| \hat{\mathbf{r}}_i)$;
      $l^r \leftarrow l^r + l_i^r$;
      if $\mathbf{r}_i \cdot (1,0) = 1$ **then**
        /* classify related */
        $\hat{\mathbf{s}}_i \leftarrow h(\hat{\mathbf{r}}_i \cdot (1,0);\theta_s)$;
        $l_i^s \leftarrow \mathrm{KL}(\mathbf{s}_i \| \hat{\mathbf{s}}_i)$;
      else
        /* unrelated */
        $l_i^s = 0$;
      $l^s \leftarrow l^s + l_i^s$;
    $d = \mathrm{MMD}(\{\mathbf{u}_i, \mathbf{r}_i\}_{i=1}^n;\theta_d)$;
    /* backward propagation */
    $\theta_s \leftarrow \theta_s - \mu \cdot \alpha \cdot \frac{\partial l^s}{\partial \theta_s}$;
    $\theta_r \leftarrow \theta_r - \mu \cdot (\frac{\partial l^r}{\partial \theta_r} + \alpha \cdot \frac{\partial l^s}{\partial \theta_r})$;
    $\theta_d \leftarrow \theta_d + \mu \cdot \beta \cdot \frac{\partial d}{\partial \theta_d}$;
    $\theta_u \leftarrow \theta_u - \mu \cdot (\frac{\partial l^r}{\partial \theta_u} + \alpha \cdot \frac{\partial l^s}{\partial \theta_u} - \beta \cdot \frac{\partial d}{\partial \theta_u})$;
  until θ_u, θ_r, θ_s, θ_d *converge*;

$p(agree)$, $p(disagree)$ and $p(discuss)$. By observing that the class agree is assumed as related, thus $p(agree, related) = p(agree)$, we derive that:

$$\begin{aligned} p(agree) &= p(agree, related) \\ &= p(agree|related) \times p(related) \\ &= p(agree|related) \times (1 - p(unrelated)). \end{aligned} \tag{14}$$

Similarly, for the other two classes we derive that:

$$\begin{aligned} p(disagree) &= p(disagree|related) \times (1 - p(unrelated)), \\ p(discuss) &= p(discuss|related) \times (1 - p(unrelated)). \end{aligned} \tag{15}$$

Thereby, the model actual output $\hat{\mathbf{y}}$ is:

$$\hat{\mathbf{y}} = (p(agree), p(disagree), p(discuss), p(unrelated)), \tag{16}$$

where the class with the highest probability corresponds to the predicted stance.

5 EXPERIMENTAL SETUP

We start this section by presenting the datasets and evaluation measures relevant to the stance detection task. Then, we describe the features used by our model and the model parameterization. Finally, we present the baselines. The software used to run the experiments of this paper is available on the website of the first author.

5.1 Datasets

Experiments are conducted on two publicly available datasets: the *Emergent* dataset[1] [14] and the *FNC-1* dataset[2]. In these two datasets, a claim consists of a news article headline and an evidence of a news article content. These datasets are split into train and test subsets; see Table 1 for statistics about the splits.

The FNC-1 dataset consist of 75,385 instances. Each instance in the dataset is a pair claim-evidence labeled as one of the four stances: agree, disagree, discuss and unrelated. The ratio of training data over testing data in the FNC-1 dataset is ∼2:1. Every class accounts for a similar percentage in the train and test subsets. The unrelated stances are the majority (over 70%) in both subsets, while the disagree stances are less than 3%. The agree and discuss stances are less than 20% and 10%.

The Emergent dataset is similar to the FNC-1 dataset, however it contains only agree, disagree and discuss stances. Hence, it needs to be augmented with unrelated stances. Similarly to how the FNC-1 dataset unrelated stances have been labeled, we manually labeled unrelated stances by pairing a claim with an unrelated evidence, i.e., paired with another claim. Moreover, to make the class distributions less imbalanced, we make the ratio of related stances and unrelated ones ∼1:1. The augmented Emergent dataset contains 4,071 training labels and 1,024 testing labels with a ratio of ∼4:1. Class distributions between train and test subsets are similar.

Compared to the FNC-1 dataset, the class distributions of the augmented Emergent dataset is more balanced. The percentage of unrelated stances is about 50%, whereas the percentages of agree and disagree stances are about 24% and 8%. Both datasets have similar percentages of the discuss stances.

5.2 Evaluation Measure

In line with the FNC-1 challenge, the evaluation is based on a weighted two-level scoring system based on the *accuracy* measure. This evaluation measure, called *relative score*, evaluates a model by splitting the stance detection task into two sub-tasks, related/unrelated and agree/disagree/discuss classification sub-tasks. To the former sub-task is given a 25% weight. This is done because this sub-task is considered to be easier than the latter sub-task to which is given a 75% weight.

We report the evaluation measures: relative score, accuracy, and accuracy on a per class basis.

5.3 Feature Extraction

To represent claims and evidences we use a bag-of-words approach. For each claim and evidence we generate a TF-IDF vector, and for each pair claim-evidence we compute their cosine similarity. We also include the FNC-1 official features into the input feature vector.

The final set of features include:

- TF-IDF vectors of claims;
- TF-IDF vectors of evidences;
- Cosine similarity (CosSim) between the claim vector and the evidence;
- Ratio of word overlap (WordLap) between the claim and the evidence;
- An Indicator whether a claim has refuting words (RefWord);
- The polarity (Pol) of the claim and the evidence;
- The number of overlapping n-grams (NGrams) for $n \in \{2, 3, 4, 5, 6\}$ between the claim and the evidence.

For the TF-IDF vectors, we only use the top 2,000 most frequent terms except stop-words. All of these features are concatenated to form the input feature vector $\mathbf{v}$.

5.4 Experimental Setting

The following hyperparameters have been set via a five-cross validation on the train subsets:

- The dimension k of hidden representations is set to 100;
- The dimension j of the MMD is set to 10;
- The activation function used in the hidden layers is set to ReLu;
- The parameters α are set to 1.5 and 1.3 for the Emergent and FNC-1 datasets.
- The parameter β is set to 0.001;

We include a L2 regularization term [29] for the MLP weight parameters in the final loss function to mitigate overfitting. Dropout is also used to mitigate overfitting with rate set to 0.6. We train in mini-batches of size 64 over the entire train subset. Note that the gradient steps in Algorithm 1 can easily be alternated with a more powerful optimizer such as the Adam optimizer [20]. Early stopping is applied when the classification loss on the validation subset does not get smaller for three continuous iterations. The whole model is implemented with TensorFlow.

5.5 Baselines

We compare our model against the methods mentioned in Section 2. These methods are detailed in the following. Among them we distinguish between methods that use the same features as ours and methods that learn their representations. We start with the latter type, we call these *representation learning-based baselines*:

Bidirectional LSTM (BiLSTM). Augenstein et al. [4] build a neural network architecture based on bidirectional LSTM on a Tweeter dataset. A LSTM encodes the claim, and another LSTM encodes the evidence with the encoded claim set as initial states. The 100-d GloVe word embedding is used as input [30];

Attentive CNN (AtCNN). Bajaj [6] builds an attention-augmented CNN. The claim and the evidence are input to a convolutional neural network to obtain hidden representations, and the attention mechanism is employed to locate the most influential words or phases on the final results;

[1] https://github.com/willferreira/mscproject.
[2] https://github.com/FakeNewsChallenge/fnc-1.

Table 1: Statistics of the datasets.

Subset	Stance	Emergent		FNC-1	
		Number	Percentage	Number	Percentage
Training	agree	992	24.37	3,678	7.36
	disagree	303	7.44	840	1.68
	discuss	776	19.06	8,909	17.83
	unrelated	2,000	49.13	36,545	73.13
		4,071		49,972	
Testing	agree	246	24.02	1,903	7.49
	disagree	91	8.89	697	2.74
	discuss	776	19.06	4,464	17.57
	unrelated	500	48.83	18,349	72.20
		1,024		25,413	

Memory Network (MN). Mohtarami et al. [26] develop an end-to-end memory network for stance detection. The network operates at the paragraph level and integrates convolutional and recurrent neural networks, as well as a similarity matrix as part of the overall architecture;

Ranbking Model (RM). Zhang et al. [48] build a ranking method to tackle the stance detection and achieve empirical performance improvements. A ranking loss function is proposed to replace Softmax and maximize the representation difference between four classes of stance.

We now review the second type of baselines: those methods that use the same features as our method, we call these *feature engineering-based baselines*:

Official Baseline (OB). This is the FNC-1 official baseline that uses one gradient boosting decision trees model for four-way classification;

Logistic Regression (LR). Bourgonje et al. [10] use n-gram matching and a rule-based procedure to decide relatedness, and three-way logistic regression to distinguish among the related classes;

Gradient Boosted Decision Trees (GBDT). Wang et al. [43] develop two GBDT models, one to determine the relatedness of an evidence to a claim, and another to distinguish among the related classes;

Multi-Layer Perception (MLP). This model [33] achieved the third best performance in FNC-1. It extracts TF-IDF and cosine similarity between claims and evidences as input features, and uses a MLP as the four-class classifier.

6 RESULTS AND DISCUSSION

In this section, we start by analyzing the dependency assumption. Then, we compare and contrast our model against the baselines. Next, we provide a sensitivity analysis of the hyperparameters. We conclude with an impact analysis of the features used by the model.

6.1 Dependency Assumption

In Figure 2 we show the effect of the 3 dependency assumptions by visualizing the learned representations using a t-SNE projection [24]. We observe that when the classifiers are assumed independent, i.e., the classification is performed in cascade — no error is propagated from the second layer to the first during training — then the learned representation well separates the unrelated class from the unrelated ones. When the classifiers are assumed dependent, i.e., the two classifiers are trained together — the error is left free to propagate from the second layer to the first — then the learned representation is not very well separated. However, when the dependence assumption of the two classifiers is learned via the MMD regularization, i.e., the two classifiers are trained together with the error propagation controlled by the regularizer, then the learned representation is again well separated like in the first case. Well-separated representations suggest a greater discriminative power of the model — the unrelated and related classes are almost linearly separable.

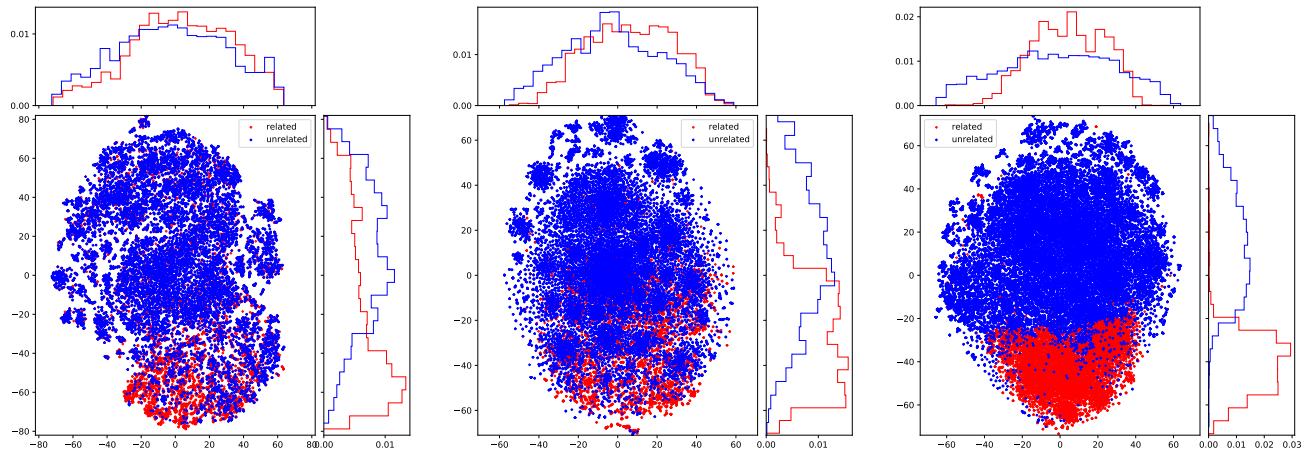
The last three rows of Tables 2 and 3 show the performance of our model on the two test subsets for each one of the three assumptions: independent, dependent, and learned. Looking at the accuracy of the unrelated class, we observe that the accuracy is greater when the learned representations are well-separated, as in the independent and learned cases. Furthermore, looking at all the other scores, we observe that the learned assumption outperforms both the independent and dependent assumptions in all other cases, demonstrating that learning together both, relatedness and stance of the evidences towards claims, is beneficial to the stance detection task.

6.2 Overall Performance

In Tables 2 and 3 we compare our model against the state-of-the-art models. Our model achieves the best stance detection performance for the relative score on both datasets. The model achieves 89.30% on the augmented Emergent test subset and 88.15% on the FNC-1 test subset.

By comparing with four-way classification baselines (OB, MLP, BiLSTM, AtCNN, MN and RM) we demonstrate the advantage of separating the relatedness detection from the stance detection. We observe that these classifiers perform poorly on the disagree class, which is caused by the large percentage difference between the minority disagree class and the majority unrelated class. Further, the more imbalanced the evaluation dataset becomes, the worse performance the four-way classifiers achieve on the minority disagree class.

By comparing with baselines that separate the relatedness detection from the stance detection (LR and GBDT) we demonstrate the



(a) Independent (b) Dependent (c) Learned

Figure 2: t-SNE visualization of the hidden representations on the training data. The hidden representations of model trained (a) with separated layers (b) together but without regularization, and (c) with MMD regularization.

Table 2: Performance comparison of our model against the State-of-the-Art models on the augmented Emergent dataset.

Model	Accuracy (%)				Relative Score (%)
	agree	disagree	discuss	unrelated	
Feature Engineering-Based Baselines					
OB	33.56	23.44	70.23	84.00	74.86
LR (Bourgonje et al.)	66.73	40.51	78.33	78.00	83.45
GBDT (Wang et al.)	80.62	50.42	83.52	88.00	87.53
MLP (Riedel et al.)	58.53	23.64	79.05	95.00	85.43
RM (Zhang et al.)	64.56	40.42	**85.45**	96.00	87.69
Representation Learning-Based Baselines					
BiLSTM (Augenstein et al.)	43.21	12.57	78.55	96.00	81.37
AtCNN (Bajaj)	44.78	14.60	72.44	**97.00**	83.56
MN (Mohtarami et al.)	54.64	40.05	72.10	89.00	85.92
Our Models					
Independent	74.54	45.32	82.59	95.49	86.33
Dependent	63.54	44.68	68.35	95.00	86.72
Learned	**82.52**	**69.05**	84.30	**97.00**	**89.30**

superiority of a single end-to-end model. LR and GBDT are better on the disagree class, although their overall performance is worse than our model.

In Figure 3 we show the confusion matrix of our model. Here we observe the detection performance on a per class basis. For the related/unrelated classification, we correctly classify 97.00% and 99.53% unrelated instances on the augmented Emergent and the FNC-1 test subsets. We can see that there is some misclassification between the agree and unrelated classes, and between the discuss and unrelated classes. The misclassification of the disagree class accounts for the largest error of the unrelated instances.

Our model achieves an accuracy of 69.05% and 72.35% for the disagree class on the Emergent and the FNC-1 test subsets. The classification accuracy is largely improved compared to the state-of-the-art. Some misclassification error exists between agree and disagree. However, our model can distinguish between the discuss and the disagree with few errors. While the number of discuss cases is the largest and the number of disagree instances is the smallest, our model does not mistake disagree instances as discuss ones, i.e., the model has learned the core representation difference between these two classes. Due to ambiguous expressions, misclassification between agree and discuss is the cause of most errors between these classes, which leads to a slightly worse accuracy for the discuss class on the Emergent (84.30%) and FNC-1 (77.49%) test subsets.

Two reasons account for the improved empirical performance observed on our model. On the one hand, the mitigation of the

Table 3: Performance comparison of our model against the State-of-the-Art models on the FNC-1 dataset.

Model	Accuracy (%)				Relative Score (%)
	agree	disagree	discuss	unrelated	
Feature Engineering-Based Baselines					
OB	10.51	1.00	79.66	97.98	75.20
LR (Bourgonje et al.)	67.42	31.61	75.23	95.36	80.63
GBDT (Wang et al.)	**82.93**	69.82	33.52	95.42	86.72
MLP (Riedel et al.)	44.04	6.60	81.38	97.90	81.72
RM (Zhang et al.)	64.90	27.26	**84.41**	99.12	86.66
Representation Learning-Based Baselines					
BiLSTM (Augenstein et al.)	35.96	0.94	80.33	98.54	78.70
AtCNN (Bajaj)	38.67	8.24	70.63	91.25	75.77
MN (Mohtarami et al.)	16.92	60.22	81.27	95.50	79.92
Our Models					
Independent	72.41	37.90	68.23	97.43	83.47
Dependent	61.34	42.93	59.38	99.05	85.32
Learned	80.61	**72.35**	77.49	**99.53**	**88.15**

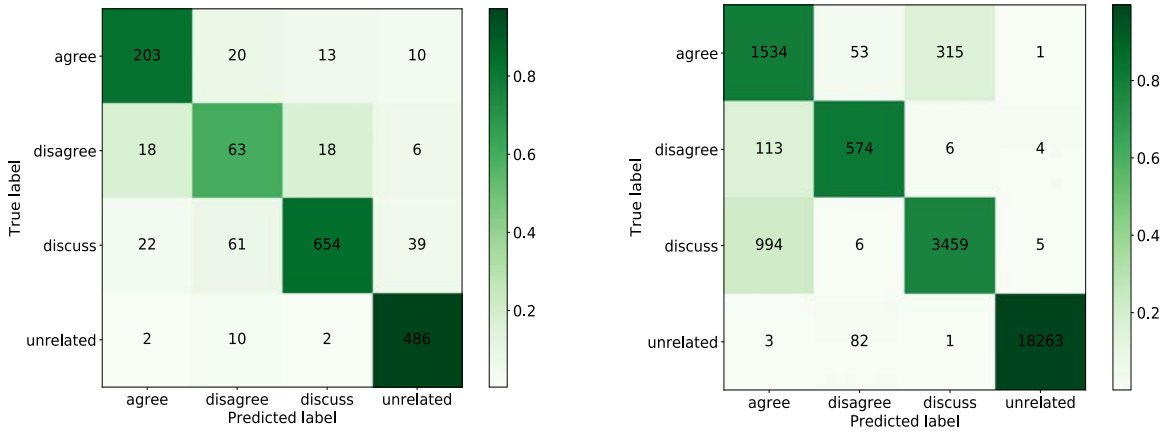


Figure 3: The confusion matrices of our model for the augmented Emergent (on the left) and FNC-1 (on the right) datasets.

class imbalance problem. Contrary to the four-way classifiers that directly compare the disagree and unrelated instances, the hierarchical model avoids the direct comparison of this minority disagree class (which is less than 2% in the FNC-1 dataset) with the majority unrelated one (which is more than 70% in the FNC-1 dataset). On the other hand, the MMD term that maximizes the discrepancy between the unrelated class and the aggregated related classes. Since the agree, disagree and discuss belong to the same class, the related class, the MMD regularization promotes the emergence of features that are useful to separate the class pairs: agree with unrelated, disagree with unrelated, and discuss with unrelated.

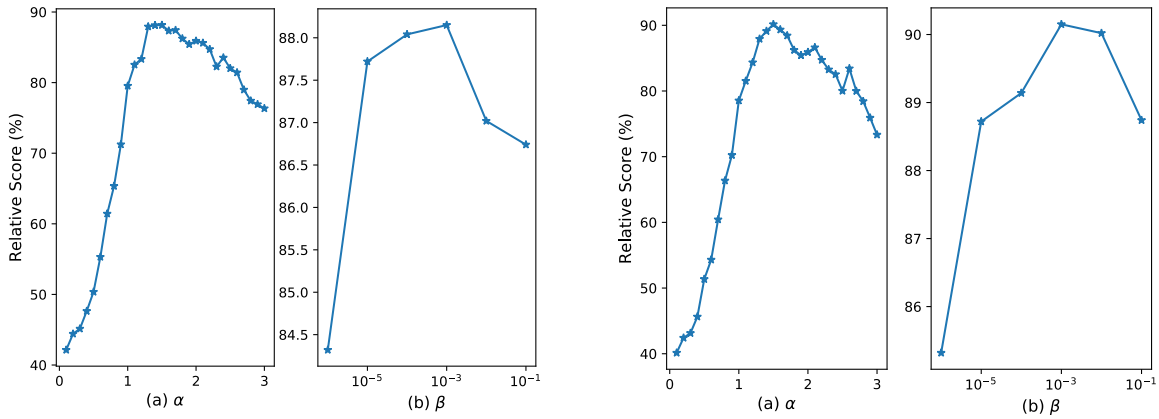
6.3 Hyperparameters Sensitivity

In this subsection we discuss the sensitivity to the hyperparameters of our model. The most influential hyperparameters for the proposed model are α and β. The former controls the relative importance of classification layers. The latter leverages the regularization.

In Figures 4(a) and 4(b) we show how the performance of the model changes when varying α and β for the augmented Emergent and FNC-1 test subsets. α is searched between 0.1 and 3.0 with steps of 0.1, and β is searched in $\{0, 0.1, 0.01, 0.001, 0.0001, 0.00001\}$. For α, we observe that the performance of the model improves quickly as α increases and peaks at 1.5 and 1.3 for the FNC-1 and augmented Emergent datasets, then the performance experiences a slight decrease when α is increased. We hypothesize that the optimal α is related to the class balance between the unrelated class and the related ones. The more unbalanced the dataset is towards the unrelated class, larger is the optimal α. For β, we observe that the performance is the highest when β is set to 0.001. This happens for both augmented Emergent and FNC-1 test subsets. These optimal values of α and β observed on the test subsets are equal to the one found when training the model.

6.4 Feature Analysis

In this subsection we evaluate and discuss the importance of each feature towards the final prediction. To examine the influence of each feature on the final performance, we do a leave-one feature set-out approach and record the classification accuracy on the stance



(a) Emergent (b) FNC-1

Figure 4: Sensitivity of the trained model when varying the parameters α and β on the test subset of the augmented Emergent (on the left) and FNC-1 (on the right) datasets.

Table 4: Performance of our model with different feature sets on the FNC-1 dataset. "/" denotes no feature set is removed.

Removed Feature Set	Accuracy (%)			
	agree	disagree	discuss	unrelated
CosSim	71.53	85.08	78.76	69.37
WordLap	67.43	80.49	77.31	77.89
RefWord	74.43	64.37	77.03	97.49
Pol	60.49	67.93	80.92	98.79
NGrams	74.27	75.73	87.82	84.52
/	80.61	82.35	77.49	99.53

detection task. The following analysis is only based on the FNC-1 dataset. Similar results are observed on the augmented Emergent dataset.

In Table 4 we show the results of this analysis. We observe that removing the CosSim feature leads to a large decrease in accuracy for the unrelated class. Similarly, the use of WordLap has a positive effect for the agree class, and it also contributes to the unrelated class. The RefWord and Pol features help for the classes agree and disagree, while removing the NGram feature leads to an increase on the discuss class, i.e., the NGram feature causes confusion between the discuss and the other classes.

7 CONCLUSION

In this paper, we studied the problem of stance detection: the classification of the stance of an evidence towards a claim into one of the four classes: agree, disagree, discuss and unrelated.

We proposed a hierarchical representation of the stance classes, where the classes agree, disagree and discuss are combined together into a class referred as the related class. The main idea here is to divide a concept into sub-concepts that are organized in a hierarchical structure, and design constraints between sub-concepts in order to make the model parameter optimization more sensible. The primary advantage of this hierarchical representation is that it is useful to overcome the class imbalance problem.

This hierarchical representation has inspired the proposed two-layer neural network to tackle the stance detection task. The first layer performs a related-unrelated classification, while the second layer performs a more fine-grained classification among the related classes. Furthermore, we have empirically demonstrated that (1) it is advantageous to learn these two classification tasks together, and (2) the dependency between these two layers can be learned through a MMD regularization term, which measures the representation discrepancy between the two layers. Experiments on two publicly available datasets have shown that our model is able to outperform the state-of-the-art stance detection methods.

As future work we consider the enriching of the proposed model as follows. First, integrating a credibility evaluation of information sources as features. Second, improving the explainability of the model by showing which words or phrases are the most influential in predicting the stance via attention mechanisms.

ACKNOWLEDGMENTS

This project was funded by the EPSRC Fellowship titled "Task Based Information Retrieval", grant reference number EP/P024289/1. We acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

REFERENCES

[1] Hunt Allcott and Matthew Gentzkow. 2017. *Social media and fake news in the 2016 election.* Technical Report. National Bureau of Economic Research.

[2] Pranav Anand, Marilyn Walker, Rob Abbott, Jean E Fox Tree, Robeson Bowmani, and Michael Minor. 2011. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of the 2nd workshop on computational approaches to subjectivity and sentiment analysis*. Association for Computational Linguistics, 1–9.

[3] Nachman Aronszajn. 1950. Theory of reproducing kernels. *Transactions of the American mathematical society* 68, 3 (1950), 337–404.

[4] Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance Detection with Bidirectional Conditional Encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 876–885. https://doi.org/10.18653/v1/D16-1084

[5] Sean Baird, Sibley Doug, and Yuxi Pan. 2017. Talos Targets Disinformation with Fake News Challenge Victory. (2017). https://blog.talosintelligence.com/2017/06/talos-fake-news-challenge.html

[6] Samir Bajaj. 2017. "The Pope Has a New Baby!" Fake News Detection Using Deep Learning. (2017).

[7] Roy Bar-Haim, Indrajit Bhattacharya, Francesco Dinuzzo, Amrita Saha, and Noam Slonim. 2017. Stance classification of context-dependent claims. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Vol. 1. 251–261.

[8] Adam J Berinsky. 2017. Rumors and health care reform: experiments in political misinformation. *British Journal of Political Science* 47, 2 (2017), 241–262.

[9] Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. 2006. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics* 22, 14 (2006), e49–e57.

[10] Peter Bourgonje, Julian Moreno Schneider, and Georg Rehm. 2017. From clickbait to fake news detection: an approach based on detecting the stance of headlines to articles. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*. 84–89.

[11] Clinton Burfoot, Steven Bird, and Timothy Baldwin. 2011. Collective classification of congressional floor-debate transcripts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 1506–1515.

[12] Sophie Chesney, Maria Liakata, Massimo Poesio, and Matthew Purver. 2017. Incongruent headlines: Yet another way to mislead your readers. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*. 56–61.

[13] Jiachen Du, Ruifeng Xu, Yulan He, and Lin Gui. 2017. Stance classification with target-specific neural attention networks. International Joint Conferences on Artificial Intelligence.

[14] William Ferreira and Andreas Vlachos. 2016. Emergent: a novel data-set for stance classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. ACL.

[15] Bo Geng, Dacheng Tao, and Chao Xu. 2011. DAML: Domain adaptation metric learning. *IEEE Transactions on Image Processing* 20, 10 (2011), 2980–2989.

[16] Andreas Hanselowski, PVS Avinesh, Benjamin Schiller, and Felix Caspelherr. 2017. *Description of the system developed by team athene in the FNC-1.* Technical Report. Technical report.

[17] Andreas Hanselowski, Avinesh PVS, Benjamin Schiller, Felix Caspelherr, Debanjan Chaudhuri, Christian M Meyer, and Iryna Gurevych. 2018. A Retrospective Analysis of the Fake News Challenge Stance Detection Task. *arXiv preprint arXiv:1806.05180* (2018).

[18] Andreas Hanselowski, Avinesh PVS, Benjamin Schiller, Felix Caspelherr, Debanjan Chaudhuri, Christian M. Meyer, and Iryna Gurevych. 2018. A Retrospective Analysis of the Fake News Challenge Stance-Detection Task. In *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 1859–1874. http://aclweb.org/anthology/C18-1158

[19] Kazi Saidul Hasan and Vincent Ng. 2013. Stance classification of ideological debates: Data, models, features, and constraints. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. 1348–1356.

[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[21] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics* 22, 1 (1951), 79–86.

[22] Srijan Kumar and Neil Shah. 2018. False information on web and social media: A survey. *arXiv preprint arXiv:1804.08559* (2018).

[23] Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Detect Rumor and Stance Jointly by Neural Multi-task Learning. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 585–593.

[24] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[25] Todor Mihaylov, Georgi Georgiev, and Preslav Nakov. 2015. Finding opinion manipulation trolls in news community forums. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*. 310–314.

[26] Mitra Mohtarami, Ramy Baly, James Glass, Preslav Nakov, Lluís Màrquez, and Alessandro Moschitti. 2018. Automatic Stance Detection Using End-to-End Memory Networks. *arXiv preprint arXiv:1804.07581* (2018).

[27] Akiko Murakami and Rudy Raymond. 2010. Support or oppose?: classifying positions in online debates from reply activities and opinion expressions. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. Association for Computational Linguistics, 869–875.

[28] Nasser M Nasrabadi. 2007. Pattern recognition and machine learning. *Journal of electronic imaging* 16, 4 (2007), 049901.

[29] Arnold Neumaier. 1998. Solving ill-conditioned and singular linear systems: A tutorial on regularization. *SIAM review* 40, 3 (1998), 636–666.

[30] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[31] Kashyap Popat, Subhabrata Mukherjee, Jannik Strötgen, and Gerhard Weikum. 2017. Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 1003–1012.

[32] Neel Rakholia and Shruti Bhargava. 2017. "Is it true?" – Deep Learning for Stance Detection in News. (2017).

[33] Benjamin Riedel, Isabelle Augenstein, Georgios P Spithourakis, and Sebastian Riedel. 2017. A simple but tough-to-beat baseline for the Fake News Challenge stance detection task. *arXiv preprint arXiv:1707.03264* (2017).

[34] Sebastian Ruder, John Glover, Afshin Mehrabani, and Parsa Ghaffari. 2018. 360 Stance Detection. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. 31–35.

[35] Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. 2004. *Kernel methods in computational biology*. MIT press.

[36] Prashant Shiralkar, Alessandro Flammini, Filippo Menczer, and Giovanni Luca Ciampaglia. 2017. Finding streams in knowledge graphs to support fact checking. In *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 859–864.

[37] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter* 19, 1 (2017), 22–36.

[38] Swapna Somasundaran and Janyce Wiebe. 2009. Recognizing stances in online debates. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*. Association for Computational Linguistics, 226–234.

[39] Swapna Somasundaran and Janyce Wiebe. 2010. Recognizing stances in ideological on-line debates. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*. Association for Computational Linguistics, 116–124.

[40] Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In *Proceedings of the 2006 conference on empirical methods in natural language processing*. Association for Computational Linguistics, 327–335.

[41] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science* 359, 6380 (2018), 1146–1151.

[42] Marilyn A Walker, Pranav Anand, Robert Abbott, and Ricky Grant. 2012. Stance classification using dialogic properties of persuasion. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*. Association for Computational Linguistics, 592–596.

[43] Xuezhi Wang, Cong Yu, Simon Baumgartner, and Flip Korn. 2018. Relevant Document Discovery for Fact-Checking Articles. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 525–533.

[44] Jen Weedon, William Nuland, and Alex Stamos. 2017. Information operations and Facebook. *version* 1 (2017), 27.

[45] Houping Xiao et al. 2018. *Multi-sourced Information Trustworthiness Analysis: Applications and Theory*. Ph.D. Dissertation. State University of New York at Buffalo.

[46] Ainur Yessenalina, Yisong Yue, and Claire Cardie. 2010. Multi-level structured models for document-level sentiment classification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1046–1056.

[47] Qiang Zhang, Aldo Lipani, Shangsong Liang, and Emine Yilmaz. 2019. Reply-Aided Detection of Misinformation via Bayesian Deep Learning. In *Companion Proceedings of the The Web Conference 2019*. ACM Press.

[48] Qiang Zhang, Emine Yilmaz, and Shangsong Liang. 2018. Ranking-based Method for News Stance Detection. In *Companion Proceedings of the The Web Conference 2018*. ACM Press.

Reply-Aided Detection of Misinformation via Bayesian Deep Learning

Qiang Zhang
University College London
London, United Kingdom
qiang.zhang.16@ucl.ac.uk

Aldo Lipani
University College London
London, United Kingdom
aldo.lipani@ucl.ac.uk

Shangsong Liang
Sun Yat-sen University
Guangzhou, China
liangshangsong@gmail.com

Emine Yilmaz
University College London
London, United Kingdom
emine.yilmaz@ucl.ac.uk

ABSTRACT

Social media platforms are a plethora of misinformation and its potential negative influence on the public is a growing concern. This concern has drawn the attention of the research community on developing mechanisms to detect misinformation. The task of misinformation detection consists of classifying whether a claim is *True* or *False*. Most research concentrates on developing machine learning models, such as neural networks, that outputs a single value in order to predict the veracity of a claim. One of the major problem faced by these models is the inability of representing the uncertainty of the prediction, which is due incomplete or finite available information about the claim being examined. We address this problem by proposing a Bayesian deep learning model. The Bayesian model outputs a distribution used to represent both the prediction and its uncertainty. In addition to the claim content, we also encode auxiliary information given by people's replies to the claim. First, the model encodes a claim to be verified, and generate a *prior belief* distribution from which we sample a latent variable. Second, the model encodes all the people's replies to the claim in a temporal order through a Long Short Term Memory network in order to summarize their content. This summary is then used to update the prior belief generating the *posterior belief*. Moreover, in order to train this model, we develop a Stochastic Gradient Variational Bayes algorithm to approximate the analytically intractable posterior distribution. Experiments conducted on two public datasets demonstrate that our model outperforms the state-of-the-art detection models.

CCS CONCEPTS

• **Information systems** → **Web mining**; • **Computing methodologies** → *Information extraction*.

KEYWORDS

misinformation detection, bayesian analysis, deep learning

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313718

ACM Reference Format:
Qiang Zhang, Aldo Lipani, Shangsong Liang, and Emine Yilmaz. 2019. Reply-Aided Detection of Misinformation via Bayesian Deep Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313718

1 INTRODUCTION

Although the digital news consumption has increased in the last decade, the increasing amount of misinformation and fake news has not certainly proven its quality. Different from traditional media where news are published by reputable organizations, online news on social media platforms such as Facebook and Twitter are shared by individuals and/or organizations without a careful checking or with malicious intents. In Figure 1 we show a false claim posted on Twitter about an alleged shooting in Ottawa. While some users showed surprise and asked for further clarifications in their replies, other users believed the claim and re-tweeted it as if it was true. This misinformation, when done on a large scale can influence the public by depicting a false picture of reality. Hence, detecting misinformation effectively has become one of the biggest challenges faced by social media platforms [17, 27].

A valuable attempt at rectifying this epidemic of false claims has been tackled by some news websites, such as: Snopes[1], Polifact[2], and Emergent[3], which have employed professional journalists to manually check and verify every potential false news. However, such manual approach is very expensive and way too slow to be able to check all the daily generated claims appearing on the web. Thus, making automatic tools is in great need to speed up this verification process.

In this paper, we tackle the automatic misinformation detection task, which consists in classifying whether a claim is *True* or *False*. Most existing models employ feature engineering or deep learning to extract features from claims' content and auxiliary information such as people's replies. However, these models generate deterministic mappings to capture the difference between true or false claims. A major limitation of these models is their inability to represent uncertainty caused by incomplete or finite available data about the claim being examined.

[1] https://www.snopes.com/
[2] https://www.politifact.com/
[3] http://www.emergent.info/



Figure 1: An example of a false claim and people's replies to it. From the replies, 231 users chose to trust the claim and re-tweeted it as if it was true, while only 4 users asked for further clarifications.

We address this problem by proposing a Bayesian deep learning model, which incorporates stochastic factors to capture complex relationships between the latent distribution and the observed variables. The proposed model makes use of the claim content and replies content. First, to represent the claim content we employ a neural model to extract textual features from claims. To deal with the ambiguity of the language used in claims and obtain salient credibility information, the model generates a latent distribution based on the extracted linguistic features. Since no auxiliary information has been used so far, we interpret this latent distribution as a *prior belief* of the claim being true. Second, to extract auxiliary information from people's replies content, we rank all the replies of the claim in temporal order, and summarize them using a Long Short Term Memory neural network (LSTM). Finally, after updating the *prior belief* with the aid of the LSTM output, the model computes the veracity prediction and its uncertainty. This updated prior belief distribution is interpreted as the *posterior belief*.

In order to train the proposed Bayesian deep learning model, due to the analytical intractability of the posterior distribution, we develop a Stochastic Gradient Variational Bayes (SGVB) algorithm. A tractable Evidence Lower BOund (ELBO) objective function of our model is derived to approximate the intractable distribution. The model is optimized along the direction of maximizing the ELBO objective function.

Our model inherit two advantages: first of all, the model incorporates a latent distribution, which enables to represent uncertainty and promote robustness; second, the Bayesian model formulates all of its prior knowledge about a claim being examined in the form of a prior, which can be updated by more added auxiliary information generating more accurate detection results. To sum up, the proposed model advances state-of-the-art methods in four aspects:

(1) An effective representation of uncertainty due to incomplete/finite available data;
(2) A temporal order-based approach to extract auxiliary information from people's replies;
(3) A SGVB algorithm to infer latent distributions;
(4) A systematic experimentation of our model on two real-world datasets.

The remainder of the paper is organized as follows: § 2 summarizes the related work; § 3 defines the misinformation detection task; § 4 details the proposed Bayesian deep learning model; § 5 derives the Stochastic Gradient Variational Bayes optimization algorithm; § 6 describes the used datasets and experimental setup; § 7 is devoted to experimental results, and; § 8 concludes the paper.

2 RELATED WORK

Misinformation has been existing for centuries in different forms of media, such as printed newspaper and television. Recently, online social media platforms are also suffering from the same issues. Recent work on misinformation detection have tried to understand the differences between true and false claims in various aspects: claim content, information source, multimedia such as affiliated images and videos, and other users' engagement.

2.1 Textual Content

The text of a claim can provide linguistic features to help predict its veracity. Since misinformation and false claims are created for financial or political purposes rather than to report an objective event, they often contain opinionated or inflammatory language [6]. In order to reveal linguistic differences between true and false claims, lexical and syntactic features at character, word, sentence and document level have been exploited [1, 11, 33, 36]. Wawer et al. [43] compute psycholinguistic features using a bag-of-words paradigm. Rashkin et al. [34] compare the language of true claims with that of satire, hoaxes, and propaganda to find linguistic characteristics of untrustworthy text. Kakol et al. [21] construct a content credibility corpus and examine a list of language factors that might affect web content credibility based on which a predictive model is developed. Bountouridis et al. [3] compare heterogeneous articles of the same story and reveal that pieces of information cross-referenced are more likely to be credible. Derczynski et al. [9] extract features from claim tweets including bag-of-words, presence of URLs, and presence of hashtags. A Support Vector Machine (SVM) is then used to distinguish between true and false claims. Guacho et al. [14] leverages a tensor decomposition to derive concise claim embeddings that capture contextual information from each claim; and uses these embeddings to create a claim-by-claim graph on which the labels propagate. Textual content has been empirically proven to be a strong indicator of claim veracity, and thus can be used as a prior probability.

2.2 Source Credibility Analysis

The credibility analysis of the sources of a claim is an important auxiliary information. As misinformation is usually published by unbelievable individuals or automatic bots, credibility plays a crucial role in message communication [18, 32]. Accurate and timely discrimination of such accounts inhibits the proliferation of misinformation at an early stage. Tseng and Fogg [40] identify two components of source credibility, namely trustworthiness and expertise. Trustworthiness is generally taken to mean truthful, unbiased and well intentioned. Expertise instead is understood as knowledgeable, experienced and competent. Thus, features that can reveal the trustworthiness and expertise of information sources are strong indicators of source credibility. With the aid of information source Thomson et al. [39] examine the credibility of tweets related to the Fukushima Daiichi nuclear disaster in Japan. They found that tweets from highly credible institutions and individuals are mostly correct. Useful account features can be derived from the account demographics, such as integrity of personal information, the number of followers and followees [5]. Besides, aggregating a group of account features are indicative, since spreaders of true and false claims might come from different communities [44], such as the percentage of verified user accounts [28] and the average number of followers [26]. However, account demographics can easily be altered to decrease the similarity between credible and incredible sources.

2.3 Multimedia Features

Multimedia features have been shown to be an important manipulator for propaganda based on misinformation [4]. As we have characterized, online misinformation exploits the individual vulnerabilities of people and thus often relies on sensational or even fake images to provoke anger or other emotional response of consumers. Visual-based features are extracted from images and videos to capture the different characteristics of misinformation. Faking images are identified based on various user-level and tweet-level hand-crafted features [15]. Recently, various visual and statistical features have been extracted for news verification [20]. Yang et al. [45] develop a convolutional neural network to extract text and visual features simultaneously. Visual features include clarity score, coherence score, diversity score, and clustering score. Statistical features include count, image ratio, multi-image ratio, hot image ratio, long image ratio, etc. This approach suffers from the problem that some misinformation on social media does not contain multimedia content.

2.4 Social Engagement

The news spreading process over time on social media involves user-driven engagement. Auxiliary information can also be derived from such engagement to improve the claim veracity detection. Ma et al. [29] propose to learn discriminative features by following non-sequential propagation structure of tweets. A top-down and a bottom-up recursive neural networks are proposed to predict claim veracity. Glenski et al. [12] seek to better understand how users react to trusted and deceptive news sources across two popular, and very different, social media platforms. Significant differences have been observed in the speed and the type of reactions between trusted and deceptive news sources on Twitter, but far smaller differences on Reddit. People react to a piece of claim by expressing their stances or emotions in social media posts. Stances can be categorized as supportive, opposing, and neutral, which can be used to infer claim veracity [19, 46, 47]. Kochkina et al. [25] propose a neural multi-task model that leverages the relationship between veracity detection and stance detection in a joint learning setup. Another common post feature is the topic distribution that indicates the central point of relevant affairs, which is derived by topic models [2]. Post features are expanded in two ways: via aggregation with relevant posts for a specific affair, and via temporal evolution of post features. The first way relies on the "wisdom of crowds" to locate potential misinformation [5], while the second way captures the periodic fluctuations of shock cycles [26] or temporal pattern of user activities, such as the number of engaged users and time intervals between engagements [37]. Yet, semantic coherence and temporal changes between users' replies are not fully explored by existing methods.

3 PROBLEM STATEMENT

The task of misinformation detection is to predict the news' veracity of claims, given their content and their people's replies.

Let $C = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_N\}$ be a set of N claims. The claim $\boldsymbol{c}_i$ is commented by a set of M user replies $\mathcal{D}_i = \{\boldsymbol{d}_{i,1}, \boldsymbol{d}_{i,2}, \ldots, \boldsymbol{d}_{i,M}\}$. We use y_i to denote the binary veracity label of the claim $\boldsymbol{c}_i$, which could be either $y_i = 1$ for true or $y_i = 0$ for false. The tuple of a claim and people's replies, i.e., $\{\boldsymbol{c}_i, \mathcal{D}_i\}$, forms a data instance to predict the claim veracity y_i. For the sake of clarity, in the following, we will omit the subscripts i when describing a single instance: $\{\boldsymbol{c}, \mathcal{D}, y\}$.

4 BAYESIAN DEEP LEARNING

In this section, we present our proposed Bayesian deep learning model that effectively integrates claim and people's replies. We will first introduce how to encode claim content with deep learning and generate a latent distribution that is interpreted as a prior belief of claim veracity. We then describe the temporal-ordered approach to encode people's replies, which captures semantic variation along the time line. Finally, we correct the prior belief with the aid of people's replies, the result of which process is interpreted as the posterior belief of claim veracity. Figure 2 describes the proposed model.

4.1 Encoding a Claim

As content are strong indicators of claim veracity [42], we apply deep learning to extract linguistic features from the claim $\boldsymbol{c}$. To avoid the ambiguity of claims and obtain salient credibility information, we generate a latent distribution based on the extracted linguistic features. The output of this claim encoder is the prior belief of the veracity of the claim.

Let each claim $\boldsymbol{c}$ be a sequence of discrete words or tokens, i.e., $\boldsymbol{c} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_l]$, where $\boldsymbol{w}_l \in \mathbb{R}^d$ is a d-dimensional word embedding vector. Based on the sequence of word embeddings, textual features are extracted via a Bidirectional Long Short Term Memory (BiLSTM) neural network [13]. The BiLSTM captures long and short semantic dependencies both from previous time steps and future time steps via forward and backward states.The BiLSTM

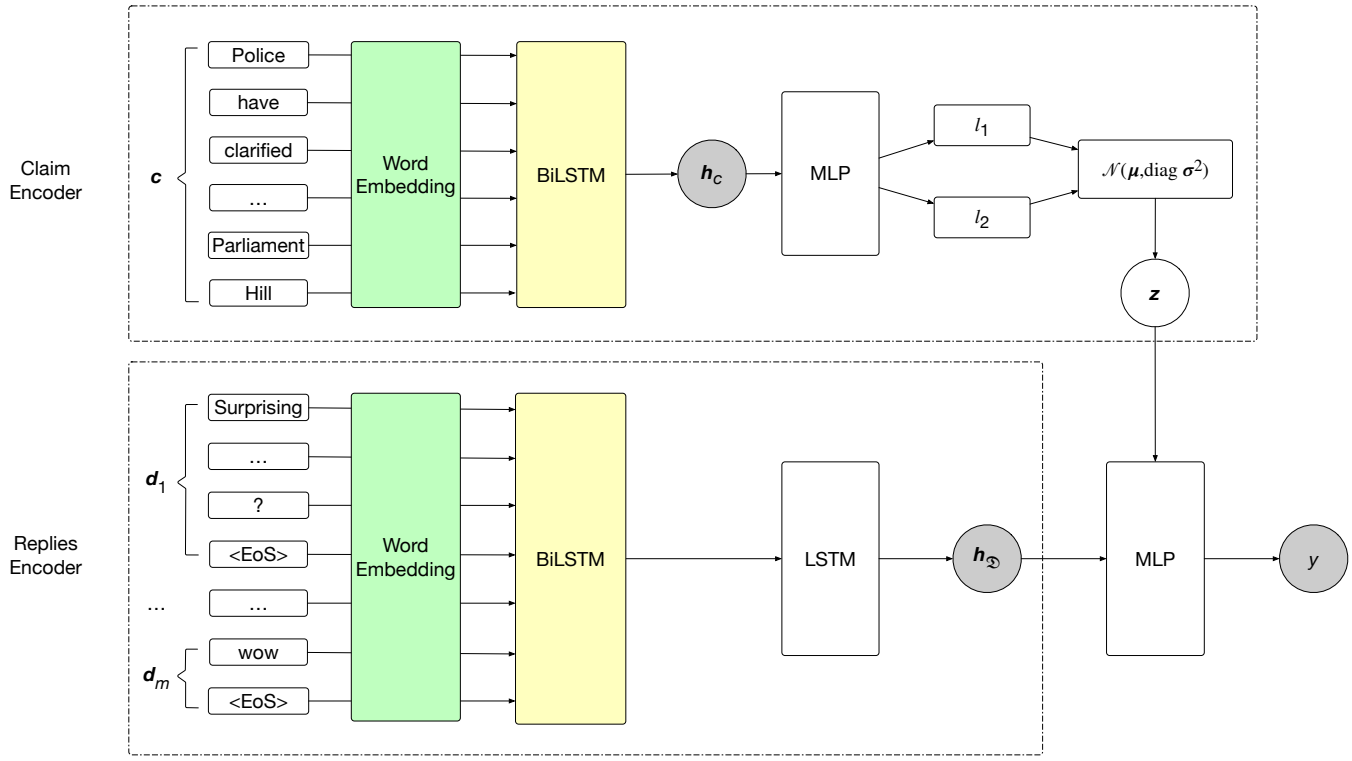


Figure 2: Framework of the Bayesian deep learning model. The framework consists of two parts, the claim encoder (§ 4.1) and the replies encoder (§ 4.2), the concatenation of which determines the *posterior belief* of claim veracity. Blocks and nodes represent computation modules and variables. Grey nodes are observed variables while blank nodes are latent variables (similarly with Figure 3). Note that blocks of the same color denote the same module.

takes as input $\boldsymbol{c}$, converts the sequence of word embeddings into a dense representation, and outputs the concatenation of two hidden states capturing past and future information:

$$\boldsymbol{h}_c = \text{BiLSTM}(\boldsymbol{c}), \tag{1}$$

where $\boldsymbol{h}_c$ denote the concatenated hidden states.

To avoid the ambiguity of claims, instead of a deterministic non-linear transformation, we generate a latent distribution, from which we sample a latent stochastic variable $\boldsymbol{z}$. To embed linguistic information into the latent variable, we set the latent variable to be conditional on $\boldsymbol{h}_c$:

$$\boldsymbol{z} \sim p_\theta(\boldsymbol{z}|\boldsymbol{h}_c), \tag{2}$$

where p is a latent distribution and θ denotes the non-linear transformation of $\boldsymbol{h}_c$ to generate the parameters of p. This non-linear transformation is essential to capture higher level representations of $\boldsymbol{h}_c$; we implement this non-linear transformation via a Multi-Layer Perceptron (MLP).

We assume that the latent variable $\boldsymbol{z}$ is continuous and follows a multivariate Gaussian distribution. The variable $\boldsymbol{z}$ is parameterized as follows:

$$p_\theta(\boldsymbol{z}|\boldsymbol{h}_c) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_\theta, \text{diag}(\boldsymbol{\sigma}_\theta^2)), \tag{3}$$

where $\boldsymbol{\mu}_\theta$ and $\text{diag}(\boldsymbol{\sigma}_\theta^2)$ are the mean and the covariance matrix of the multivariate Gaussian distribution. Since the variable $\boldsymbol{z}$ is conditional on the the claim hidden states $\boldsymbol{h}_c$, we derive these two parameters of the Gaussian distribution from $\boldsymbol{h}_c$ through a deep neural network:

$$\boldsymbol{\pi}_\theta = f_\theta(\boldsymbol{h}_c), \tag{4}$$

$$\boldsymbol{\mu}_\theta = l_1(\boldsymbol{\pi}_\theta) \quad \ln(\boldsymbol{\sigma}_\theta) = l_2(\boldsymbol{\pi}_\theta), \tag{5}$$

where f_θ denotes a MLP, l_1 and l_2 denote two Linear Transformations (LT). Since LT can generate negative values, to produce σ_θ we exponentiate the result of l_2.

In order to make μ_θ and σ_θ differentiable and backpropagate the loss through the latent distribution (p), the following reparameterization trick is used:

$$\boldsymbol{z} = \boldsymbol{\epsilon} \cdot \boldsymbol{\sigma}_\theta + \boldsymbol{\mu}_\theta \quad \text{where } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{6}$$

where $\mathbf{0}$ is a vector of zeros and $\mathbf{I}$ is the identity matrix. By making use of the latent variable ($\boldsymbol{z}$), our model is able to capture complex noisy patterns in the data.

4.2 Encoding People's Replies

We now present the people's replies encoder to obtain auxiliary information. This auxiliary information is claim-specific and is used to generate the posterior belief by correcting the prior belief of the claim veracity.

Replies on social media platforms are listed along the time line as shown in Figure 1, where the earliest reply appears at the top of the list. Truth about an event can be gradually manifest as more

evidence emerges, thus we assume that the latest replies tend to be more reliable and more important than the earlier replies. Based on this assumption, we design a two-layer recurrent neural network to encode replies: the first-layer applies a BiLSTM to summarize the semantic information of each reply and the second-layer applies a LSTM to capture the temporal semantic variation of the replies.

Given a claim $\boldsymbol{c}$ commented by a list of replies $\mathfrak{D} = \{\boldsymbol{d}_1, \ldots \ldots, \boldsymbol{d}_m, \ldots, \boldsymbol{d}_M\}$, these replies are ranked based on their temporal order. The content of a $\boldsymbol{d}_m$ consists of a sequence of words $\boldsymbol{d} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k]$. To project the claim and replies into the same semantic space, we use the same pre-trained word embeddings for both claims and replies. Hence, $\boldsymbol{w}_k \in \mathbb{R}^d$ is a d-dimensional vector such as the word embedding vector used to encoding the claim. For the sake of semantic coherence, we also employ the same BiLSTM to encode both the claim and its replies. Take the reply $\boldsymbol{d}_m \in \mathfrak{D}$ for example, the concatenation of hidden states from forward and backward directions is denoted as:

$$\boldsymbol{h}_{d_m} = \text{BiLSTM}(\boldsymbol{d}_m), \tag{7}$$

where $\boldsymbol{h}_{d_m}$ is the summary of the reply $\boldsymbol{d}_m$.

In order to capture the semantic information of all replies, we sequentially input the concatenated hidden states of each reply into a LSTM. We use the LSTM rather than a BiLSTM because the former gives high weights to recent input, which matches our assumption on the relative importance of the latest reply. Specifically, the LSTM takes the hidden states of each reply as input in a sequential way:

$$\boldsymbol{h}_{\mathfrak{D}} = \text{LSTM}(\boldsymbol{h}_d^M), \tag{8}$$

where $\boldsymbol{h}_d^M = \{\boldsymbol{h}_{d_1}, \ldots, \boldsymbol{h}_{d_m}, \ldots, \boldsymbol{h}_{d_M}\}$.

The last hidden state $\boldsymbol{h}_{\mathfrak{D}}$ contains both the linguistic information of each reply and the semantic changes between replies along the time line. This $\boldsymbol{h}_{\mathfrak{D}}$ is then used to generate the posterior belief by correcting the prior belief of the claim veracity.

4.3 Veracity Modeling

In § 4.1, we developed a prior belief of the claim veracity. In this section we show how to correct this prior belief by including its replies.

The posterior belief is generated by combining the claim and reply information via a MLP. The strong non-linearity of MLPs make them suitable to find complex relationships between the claim and its replies. Specifically, the MLP input is the latent claim variable $\boldsymbol{z}$ concatenated to the hidden state of replies $\boldsymbol{h}_{\mathfrak{D}}$:

$$y = \text{MLP}(\boldsymbol{z}, \boldsymbol{h}_{\mathfrak{D}}). \tag{9}$$

This is the final prediction of our Bayesian deep learning model for misinformation detection.

5 OPTIMIZATION

The stochastic variables of our model are non-linear and non-conjugate [41]. Hence, the posterior distribution cannot be derived analytically. To approximate the posterior distribution, we construct an inference model parameterized by ϕ to approximate the intractable true posterior $p_\theta(\boldsymbol{z}|\boldsymbol{h}_c)$; then we derive an objective function to measure how well $p_\theta(\boldsymbol{z}|\boldsymbol{h}_c)$ is approximated; finally we exploit the Stochastic Gradient Variational Bayes (SGVB) method [24, 35] to learn the inference model parameters ϕ together with the generative model parameters θ. Figure 3 shows the graphical representation of the generative model and the inference model.

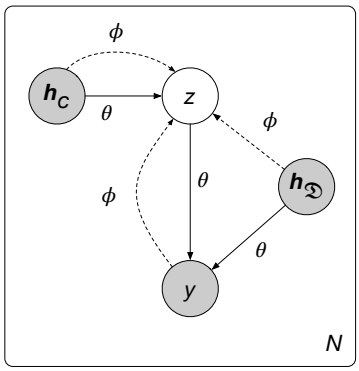


Figure 3: The directed graphical model. Grey nodes represent observed variables while blank nodes represent latent variables. Solid lines denote the generative model $p_\theta(z|\boldsymbol{h}_c)p(y|z, \boldsymbol{h}_{\mathfrak{D}})$, dashed lines denote the variational approximation $q_\phi(z|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}})$ to the intractable posterior $p_\theta(z|\boldsymbol{h}_c)$. The variational parameters are learned together with the generative model parameters.

5.1 Inference Model

Following the neural variational inference approach [24], we construct an inference model (as in Fig. 3) parameterized by ϕ to compute an approximated posterior distribution, called variational distribution. Given the observed variables, we define a variational distribution $q_\phi(\boldsymbol{z}|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}})$ to approximate the true posterior distribution $p_\theta(\boldsymbol{z}|\boldsymbol{h}_c)$. Like for the Variational Auto Encoder (VAE) [24], similarly to Eq. 3 for $p_\theta(\boldsymbol{z}|\boldsymbol{h}_c)$, the variational distribution is chosen to be a multivariate Gaussian distribution:

$$q_\phi(\boldsymbol{z}|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_\phi, \text{diag}(\boldsymbol{\sigma}_\phi^2)), \tag{10}$$

where $\boldsymbol{\mu}_\phi$ and $\text{diag}(\boldsymbol{\sigma}_\phi^2)$ are the mean and the covariance matrix of the multivariate Gaussian distribution. We use a deep neural network to derive these two parameters from the observed variables:

$$\boldsymbol{\pi}_\phi = f_\phi(y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}}), \tag{11}$$

$$\boldsymbol{\mu}_\phi = l_3(\boldsymbol{\pi}_\phi), \quad \ln(\boldsymbol{\sigma}_\phi) = l_4(\boldsymbol{\pi}_\phi), \tag{12}$$

where f_ϕ denotes a MLP, and l_3 and l_4 denote two LTs. Note that in the inference model to compute $\boldsymbol{\mu}$ and $\log \boldsymbol{\sigma}$ we use $y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}}$ and not only $\boldsymbol{h}_c$ as in the generative model.

5.2 Objective Function

In the following we derive the objective function of our Bayesian deep learning model following the variational principle. To maximize the log-likelihood $\ln(p(y|\boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}}))$, we derive an Evidence Lower Bound (ELBO) objective function, which ensures a correct approximation of the true posterior. To simplify the notation of the derivation of the objective function we make the following substitutions: $p_\theta(y) = p_\theta(y|\boldsymbol{z}, \boldsymbol{h}_{\mathfrak{D}})$, $p_\theta(\boldsymbol{z}) = p_\theta(\boldsymbol{z}|\boldsymbol{h}_c)$, $q_\phi(\boldsymbol{z}) =$

Algorithm 1: Optimization of the proposed model.

input : Claim hidden state $\boldsymbol{h}_c$, replies hidden state $\boldsymbol{h}_{\mathfrak{D}}$, veracity label y, the number claims N and the learning rate η.

begin
  $\theta, \phi \leftarrow$ Initialize parameters;
  repeat
    Randomly draw a minibatch of B claims;
    for *i=1,...,B* **do**
      $\tilde{z} \leftarrow$ randomly draw S samples from $\mathcal{N}(z|\boldsymbol{\mu}_\phi, \text{diag}(\boldsymbol{\sigma}^2_\phi))$;
      $\mathcal{L}(\theta, \phi|y_i, \boldsymbol{h}_{c_i}, \boldsymbol{h}_{\mathfrak{D}_i}) \leftarrow$ compute the loss for one claim according to Eq. 14;
    $\mathcal{L}(\theta, \phi|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}}) \leftarrow$ compute the loss for the full dataset according to Eq. 15;
    $\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}(\theta, \phi|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}})$;
    $\phi \leftarrow \phi - \eta \cdot \nabla_\phi \mathcal{L}(\theta, \phi|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}})$;
  until θ, ϕ *converge*;
  return θ, ϕ;

$q_\phi(z|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}})$. The objective function is derived as follows:

$$\begin{aligned}\ln\left(p(y|\boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}})\right) &= \ln\left(\int p_\theta(y)p_\theta(z)dz\right)\\ &\geq \int q_\phi(z)\ln\left(\frac{p_\theta(z)}{q_\phi(z)}p_\theta(y)\right)dz\\ &= \mathbb{E}_{q_\phi(z)}[\ln\left(p_\theta(y)\right)] - D_{KL}[q_\phi(z)\|p_\theta(z)]\\ &= \mathcal{L}(\theta, \phi|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}}),\end{aligned} \tag{13}$$

where $\mathcal{L}(\theta, \phi|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}})$ is the ELBO objective function. The second line of the derivation is possible by using the Jensen's inequality [16]. Since the ELBO objective function is a lower bound of the log-likelihood $\ln(p(y|\boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}}))$, its maximization maximizes the log-likelihood.

5.3 Gradient Estimation

Large-scale inference needs minibatch optimization. Thus, we derive a minibatch-based SGVB estimator to differentiate and optimize the ELBO objective function ($\mathcal{L}(\theta, \phi|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}})$) with respect to both the inference parameters (ϕ) and the generative parameters (θ).

Through Monte Carlo estimation we compute the expectation part of the ELBO objective function. Let the minibatch size to be B and, for each claim $\boldsymbol{c}_i$ with $i \in [1, B]$, S a sample drawn from the variational posterior distribution $\tilde{z} \sim q_\phi$. Given a subset of claims, we can construct an estimator of ELBO objective function for the full dataset based on mini-batches as follows:

$$\begin{aligned}\mathcal{L}(\theta, \phi|y_i, \boldsymbol{h}_{c_i}, \boldsymbol{h}_{\mathfrak{D}_i}) =& \frac{1}{S}\sum_{s=1}^{S}\left[\log p_\theta(y_i|\tilde{z}^{(s)}, \boldsymbol{h}_{\mathfrak{D}_i})\right]\\ &- D_{KL}[q_\phi(z)\|p_\theta(z)],\end{aligned} \tag{14}$$

$$\mathcal{L}(\theta, \phi|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}}) \approx \frac{N}{B}\sum_{i=1}^{B}\mathcal{L}(\theta, \phi|y_i, \boldsymbol{h}_{c_i}, \boldsymbol{h}_{\mathfrak{D}_i}), \tag{15}$$

where $\mathcal{L}(\theta, \phi|y_i, \boldsymbol{h}_{c_i}, \boldsymbol{h}_{\mathfrak{D}_i})$ denote the estimates based on the i claim and N is the total number of claims. Algorithm 1 shows the minibatch gradient descent optimization process for both the generative (θ) and inference (ϕ) parameters. Note that the gradient steps in Algorithm 1 can easily be alternated with a more powerful optimizer such as the Adam algorithm [23].

Although both $q_\phi(z|y, \boldsymbol{h}_c, \boldsymbol{h}_{\mathfrak{D}})$ and $p_\theta(z|\boldsymbol{h}_c)$ are modeled as parameterized Gaussian distributions. The former is an approximation of the latter that only functions during learning. The latter, instead, is the learned distribution from which samples are generated in order to classify claim veracity.

5.4 Prediction

After training, we compute the posterior distribution $p_\theta(z|\boldsymbol{h}_c)$ through the generative network. The actual prediction of a claim veracity is given by taking the expectation of S samples:

$$\overline{y} = \frac{1}{S}\sum_{s=1}^{S}\text{MLP}(z^s, \boldsymbol{h}_{\mathfrak{D}}), \tag{16}$$

where z^s denote the samples drawn from the true posterior distribution $p_\theta(z|\boldsymbol{h}_c)$.

6 EXPERIMENT SETUP

In this section we start by introducing 4 research questions. We then present the methodology used to answer them. The software used to run the experiments in this paper is available on the website of the first author.

6.1 Research Questions

We seek to answer the following four research questions, which will be guide the remainder of the paper:

RQ1 Does our model outperform the state-of-the-art misinformation detection baselines?
RQ2 Does the incorporation of the latent distribution outperforms a deterministic counterpart?
RQ3 Does the auxiliary information from people's replies produce a more accurate *posterior belief* of claim veracity?
RQ4 Is the temporal order better than random when encoding replies?
RQ5 Is it beneficial to incorporate a latent variable to encode replies?
RQ6 How does the dimension of the latent variable z affect the model's performance?

6.2 Datasets

In order to compare the performance of our proposed model against the baselines, we experimented with two real-world benchmark datasets, the *RumourEval* [9] and the *Pheme* [48] datasets. Both datasets contain Twitter conversation threads about news (like the example shown in Figure 1). A conversation thread consists of a tweet making a true or false claim, and branches of people's replies expressing their opinion about it. A summary of the datasets statistics is available in Table 1.

The RumourEval dataset has been developed for the SemEval-2017 Task 8 competition. This dataset consists of 325 source tweets

Table 1: Statistics of the datasets.

Subset	Veracity	RumourEval		Pheme	
		#Claims	#Replies	#Claims	#Replies
Training	True	83	1,949	861	24,438
	False	70	1,504	625	17,676
	Total	153	3,453	1,468	42,114
Validation	True	10	101	95	1,154
	False	12	141	115	1,611
	Total	22	242	210	2,765
Testing	True	9	412	198	3077
	False	12	437	219	3,265
	Total	21	849	417	6,342

and 5,568 user reply tweets. The veracity of each tweet can be true (45%), false (23%) or unverified (32%). Since we aim to only distinguish true and false claims, we filter out the unverified tweets. We divide the filtered dataset into a training subset, a validation subset and a testing subset. The training subset contains 153 claims with 3,453 replies, the validation subset contains 22 claims with to 242 replies, and the test subset contains 21 claims with 849 replies.

The *Pheme* dataset is constructed to help understand how users treat online rumour before and after the news is detected to be true or false. Like the *RumourEval* dataset, we divide the *Pheme* dataset into a training subset, a validation subset and a testing subset. Specifically, 70% of the claims are randomly selected as training instances, 10% as validation instances and the rest as testing instances. Users' replies are divided according to the claims.

6.3 Evaluation Measures

The misinformation detection task is a binary classification task. Such tasks are commonly evaluated by the following evaluation measures: Accuracy, F_1, Precision, and Recall.

Accuracy is a common evaluation measure for classification tasks. However, it is less reliable when datasets suffer from class imbalance. The evaluation measures Precision, Recall and F_1 complement Accuracy because not suffering from this problem.

6.4 Hyperparameters Setting

The activation function of the three LSTMs is tanh. The activation function of the MLPs is *ReLu*.

The hyperparameters tuned on the validation subset are:

- the dimension of the hidden layer of all three LSTMs is 30;
- the dimension of the latent variables is 10;
- the minibatch size is 32;
- the number of samples used in Monte Carlo estimates is 20.

State-of-the-art techniques have been employed to optimize the objective function: Dropout [38] is applied to improve neural networks training, L2-norm regularization is imposed on the weights of the neural networks, Adam optimizer [23] is exploited for fast convergence, and stepwise exponential learning rate decay is adopted to anneal the variations of convergence.

6.5 Baselines

We test our Bayesian deep learning model against six state-of-the-art models. In order to have a fair comparison, only those models using the claim content and users' replies have been selected.

Support Vector Machine (SVM). This model evaluates the performance of manually extracted features. The extracted features from claim content include: bag-of-words representation, presence of URLs, presence of hashtags, proportion of supporting and denying response [9]. These features are then input to a linear Support Vector Machine classifier. This classifier achieves the highest misinformation detection performance in the SemEval-2017 Task 8[4];

Convolutional Neural Networks (CNN). This model evaluates the performance of CNNs on the veracity detection task. Apart from the sequential approach such as BiLSTM, the convolutional model is another powerful neural architecture for natural language understanding [7, 8, 10, 22, 45]. CNN takes as input pre-trained word embeddings generated with Word2Vec [30] trained on the Google News dataset. To capture features similar to n-grams, we apply different convolutional window sizes. A max poling layer is applied to compress the output information of the convolutional layers [7];

Tensor Embeddings (TE). This model leverages tensor decomposition to derive concise claim embeddings, which are used to create a claim-by-claim graph on which labels propagate [14];

Evidence-Aware Deep Learning (DeClarE). This model retrieves evidences from replies using claims as a queries [31]. Then both claims and retrieved replies are input into a deep neural network with attention mechanism. Claim veracity is then computed by aggregating over the prediction generated by every claim-retrieved reply pair;

Multitask Learning (Multitask). This model leverages the relationship between two tasks of the veracity detection pipeline [25], stance detection and veracity prediction tasks. The model is trained on both jointly. We apply the hard parameter sharing mechanism, where different tasks share the same hidden LSTM layers. Task-specific layers takes the shared hidden information and generate per-task predictions;

Tree-structured RNN (TRNN). This model learns discriminative features from replies content by following their non-sequential propagation structure. Among the proposed two structures, we select the top-down structure for tweet representation learning because marginally better than the bottom-up structure [29].

7 RESULTS AND DISCUSSION

This section answers the research questions proposed in § 6.

7.1 Performance Comparison (RQ1)

Table 2 summarizes the classification performance of the baselines and our Bayesian deep learning framework on the RumourEval and Pheme datasets. We can observe that: (1) In terms of *Accuracy*

[4] http://alt.qcri.org/semeval2017/task8/

Table 2: Performance comparison of the proposed Bayesian deep learning framework against the baselines.

Dataset	Measure	SVM	CNN	TE	DeClarE	Multitask	TRNN	Ours
RumourEval	Accuracy (%)	71.42	61.90	66.67	66.67	66.67	76.19	**80.95**
	Precision (%)	66.67	54.54	60.00	58.33	57.14	70.00	**77.78**
	Recall (%)	66.67	66.67	66.67	77.78	**88.89**	77.78	77.78
	F_1 (%)	66.67	59.88	63.15	66.67	69.57	73.68	**77.78**
Pheme	Accuracy (%)	72.18	59.23	65.22	67.87	74.94	78.65	**80.33**
	Precision (%)	**78.80**	56.14	63.05	64.68	68.77	77.11	78.29
	Recall (%)	75.75	64.64	64.64	71.21	**87.87**	78.28	79.29
	F_1 (%)	72.10	60.09	63.83	67.89	77.15	77.69	**78.78**

and F_1, most deep learning-based models, such as ours, TRNN and Multitask, outperform the feature engineering-based models, i.e., SVM. This demonstrates that deep neural networks indeed help to learn better hidden representation of claims and replies. (2) Methods exploring relationships between a claim and its replies, such as ours, TRNN and Multitask, achieve better performance than claim content-based methods like TE and CNN. This demonstrates the significance of utilizing people's replies in the misinformation detection task. (3) Our model achieves state-of-the-art performance on both measures and both datasets demonstrating the effectiveness of our model in the misinformation detection task. This is true despite, our system is not the best for precision and recall alone. This because precision and recall alone do not offer a clear picture about the performance of a model since one measure can be increased at the expense of the other and *vice versa*.

Specifically, our model achieves the highest accuracy (80.95%), precision (77.78%) and F_1 (77.78%) on the RumourEval test subset, and the highest Accuracy (80.33%) and F_1 (78.78%) on the Pheme test subset. The TRNN is the strongest baseline achieving the second highest accuracy and F_1 on both RumourEval (76.19% and 73.68%) and Pheme (78.65% and 77.69%) test subsets.

7.2 Ablation of the Latent Distribution (RQ2)

In this subsection, we evaluate the impact of using a latent distribution into the claim encoder on the misinformation detection task. To evaluate the impact of the latent distribution p, we ablate p in our model and compare its classification performance against the full model. Specifically, the ablation is done by taking the output of the BiLSTM hidden states, i.e., $\boldsymbol{h}_c$ and give this as input to the output MLP. The rest of the model remains unchanged. Since no latent distribution is involved, the ablated model is optimized in accordance with the conventional Softmax loss minimization.

In Figure 4(a) and 4(b) we show the classification performance of the ablated model against the full model on the *RumourEval* and *Pheme* test subsets. We observe that the full model outperforms the ablated one by at least 7.77% on every evaluation measure. This demonstrates the better representation quality achieved by the use of the latent distribution.

7.3 Ablation of People's Replies (RQ3)

We now evaluate the contribution people's replies in the misinformation detection task. In order to examine its contribution we compare our full model with and without replies. Specifically, we ablate the input coming from the replies to the final MLP, which now is used only to perform a non-linear transformation of the latent variable $\boldsymbol{z}$.

In Figure 5(a) and 5(b) we show the classification performance of the ablated model against the full model on the RumourEval and the Pheme test subsets. Here, we observe that the auxiliary information extracted from people's replies has a large impact to the final performance our model. In fact, every evaluation measure is increased by at least 10.11%.

7.4 Random vs. Temporal Ordered Replies (RQ4)

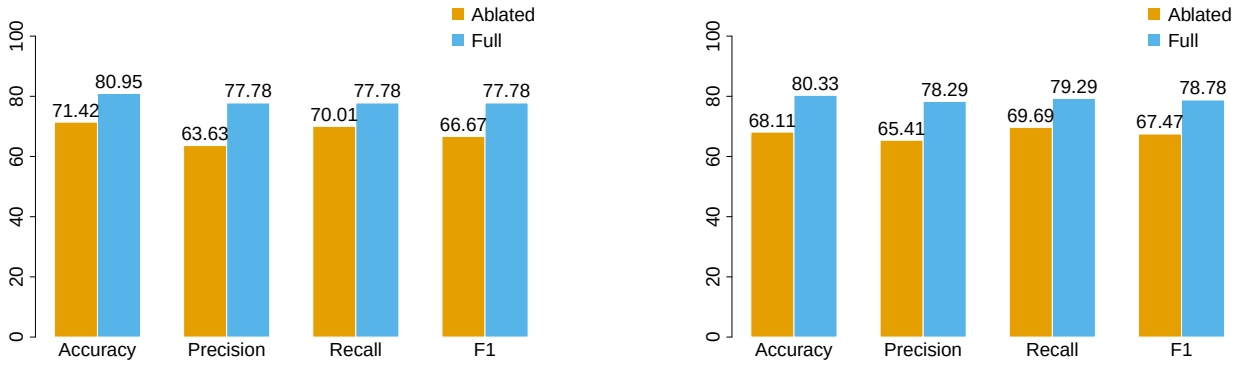
The proposed model rank people's replies based on the temporal order. In this subsection, we analyze the contribution of ranking the replies according to their temporal order. We compare this against a random order.Specifically, we randomize the $\boldsymbol{h}_d^M$ before it is input to the LSTM.

In Figure 6(a) and 6(b) we show the performance comparison of these two orders. We observe that the temporal ordered replies achieve better performance than the random ordered. Besides, the random ordered model is still worse TRNN yet better than Multitask. This is probably because TRNN takes the temporal structure of replies into the model while Multitask fail to involve temporal information.

7.5 A Latent Distribution for Replies (RQ5)

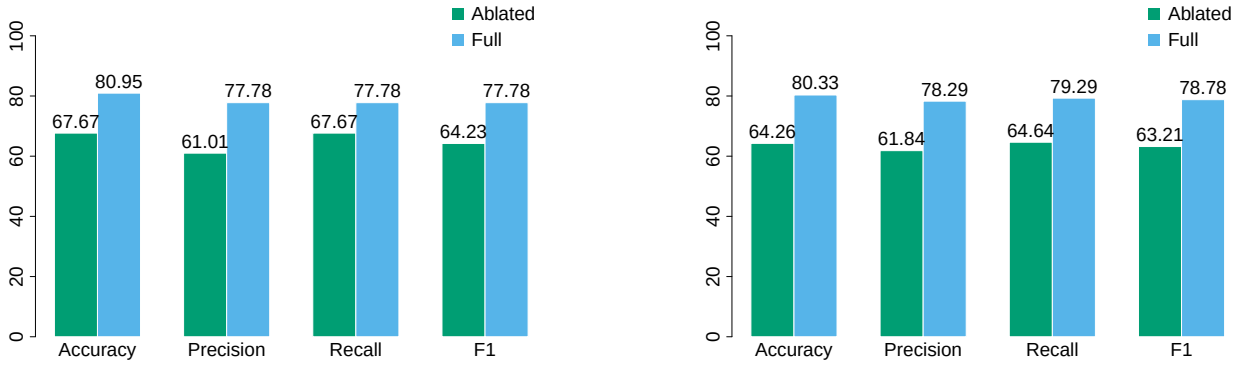
Considering the improved performance brought by the latent distribution for claims, in this subsection we answer whether it would be beneficial to incorporate a latent distribution also for replies. In order to answer this research question, we expand our model by adding a new latent distribution in the reply encoder. Similarly to what done for the claim encoder, the new latent distribution is designed as a multidimensional Gaussian distribution with mean and covariance matrix derived from the LSTM output $\boldsymbol{h}_{\mathcal{D}}$ (as in Eq. 3, 4 and 5). A new latent variable is sampled similarly as in Eq 6 and input to the MLP to predicting veracity of the claim being examined.

In Figure 8(a) and 8(b) we show the model performance comparison. We observe that the new latent distribution does not have an effect on the performance on the model for all the evaluation measures and dataset test subsets. Based on this analysis, we conclude that the incorporation of the additional latent distribution for replies does not provide any additional improvement in performance.



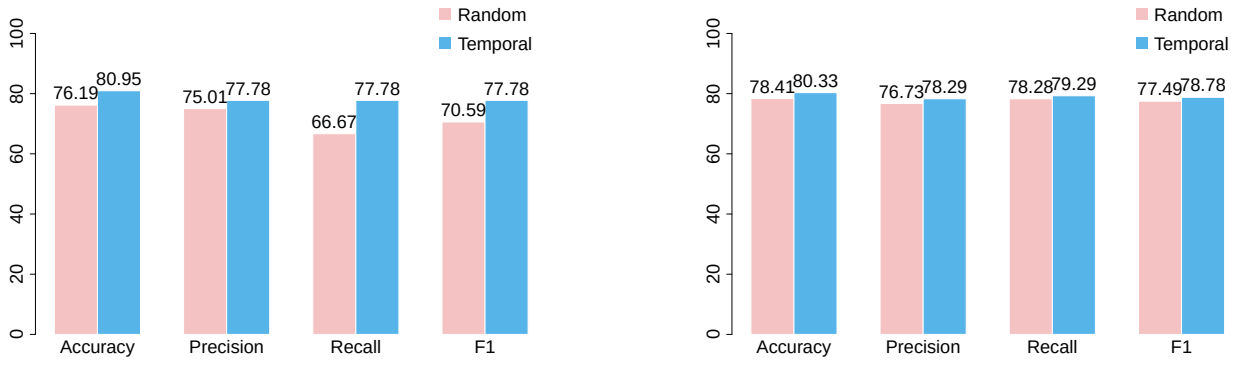
(a) RumourEval (b) Pheme

Figure 4: The impact of the latent distribution p on the model performance. In both figures we show the performance change on all the evaluation measures of the model with (Full) or without (Ablated) p. Figure (a) shows it for the RumourEval test subset and Figure (b) shows it for the Pheme test subset.



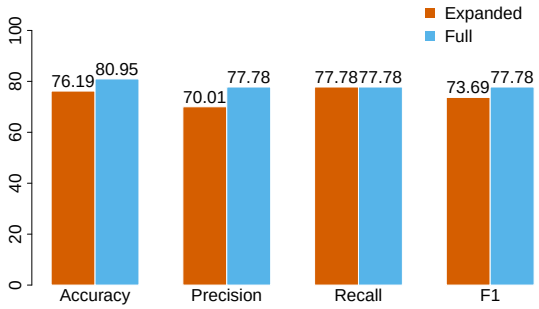
(a) RumourEval (b) Pheme

Figure 5: The impact of people's replies on the model performance. In both figures we show the performance change on all the evaluation measures of the model with (Full) or without (Ablated) replies information. Figure (a) shows it for the RumourEval test subset Figure (b) shows it for the Pheme test subset.

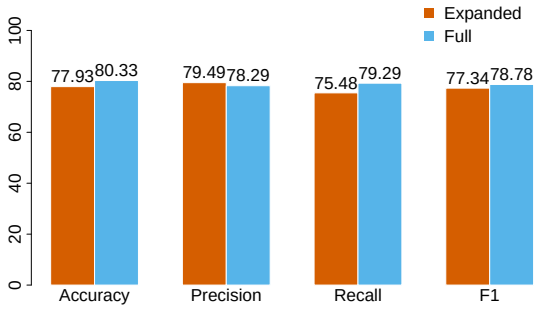


(a) RumourEval (b) Pheme

Figure 6: The effect of the temporal order of the reply encoder on model performance. In both figures we show the performance change on all the evaluation measures of the model with random and temporal ordered people's replies. Figure (a) shows it for the RumourEval test subset and Figure (b) shows it for the Pheme test subset.

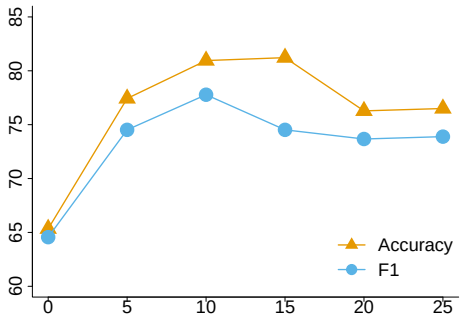


(a) RumourEval

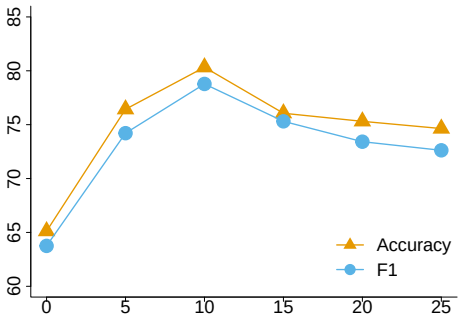


(b) Pheme

Figure 7: The effect of an additional latent distribution for people's replies on the model performance. In both figures we show the performance change on all the evaluation measures of the model with (Expanded) and without (Full) an additional latent variable for the people's replies. Figure (a) shows it for the RumourEval test subset and Figure (b) shows it for the Pheme test subset.



(a) RumourEval



(b) Pheme

Figure 8: The effect of the latent variable dimension on model performance. In both figures we show how the accuracy and F_1 scores change when varying the dimension of z. Figure (a) shows it for the RumourEval test subset and Figure (b) shows it for the Pheme test subset.

7.6 Sensitivity Analysis (RQ6)

In this subsection we evaluate the effect of the dimension of the latent variable z. To do this after setting a dimension for z we optimize the rest of the hyperparameters on the validation subset.

In Figure 8(a) and 8(b) we show the effect on performance of the dimension of z on both datasets. We observe that the results are similar for both evaluation measures, accuracy and F_1. Varying the dimension from 1 to 5 the model brings a larger performance improvement than when varying it from 5 to 25. When the dimension is 15 the model obtains the highest accuracy, 81.22%, on the RumourEval test subset while when the dimension is 10 the model obtains the highest F_1, 78.78%, on the RumourEval test subset and highest accuracy, 80.33% and F_1, 78.78%, on the Pheme test subset. These results also show that the increase in model capacity may not necessarily lead to an improvement in performance. The reason could be found on the limited size of the datasets, which might cause overfitting when the model is too complex.

8 CONCLUSIONS

In this paper, we study the problem of misinformation detection on social media platforms. One major problem faced by existing machine learning methods is the inability to represent uncertainty due to incomplete or finite available information. We address the problem by proposing a Bayesian deep learning model. When encoding claim content, we incorporate a latent distribution accounting for uncertainty and randomness caused by noisy patterns in the finite dataset. This latent distribution provides a prior belief of claim veracity. We also encode auxiliary information from people's replies in a temporal order through an LSTM. Such auxiliary information is then used to update the prior belief generating a posterior belief. In order to optimize the Bayes model, we derive a minibatch-based gradient estimation algorithm. Systematic experimentation has demonstrated the superiority of our approach against the state-of-the-art approaches in the misinformation detection task.

Despite encouraging experimental results, online misinformation detection is still a challenging problem with many open questions. In this paper, auxiliary information comes from people's replies alone, we argue that the proposed model can be enriched by utilizing other auxiliary information, such as source credibility. Also, the reply stances are a strong veracity indicator for a claim, since false claims are usually controversial and accompanied by opposite stances. We let for future work, the combination of features extract from credibility analysis and reply stances.

ACKNOWLEDGMENTS

This project was funded by the EPSRC Fellowship titled "Task Based Information Retrieval", grant reference number EP/P024289/1. We acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

REFERENCES

[1] Sadia Afroz, Michael Brennan, and Rachel Greenstadt. 2012. Detecting Hoaxes, Frauds, and Deception in Writing Style Online. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy (SP '12)*. IEEE Computer Society, Washington, DC, USA, 461–475. https://doi.org/10.1109/SP.2012.34

[2] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[3] Dimitrios Bountouridis, Mónica Marrero, Nava Tintarev, and Claudia Hauff. 2018. Explaining Credibility in News Articles using Cross-Referencing. In *SIGIR workshop on ExplainAble Recommendation and Search (EARS)*.

[4] Carlos Castillo, Mohammed El-Haddad, Jürgen Pfeffer, and Matt Stempeck. 2014. Characterizing the life cycle of online news stories using social media reactions. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. ACM, 211–223.

[5] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*. ACM, 675–684.

[6] Yimin Chen, Niall J Conroy, and Victoria L Rubin. 2015. Misleading online content: Recognizing clickbait as false news. In *Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection*. ACM, 15–19.

[7] Yi-Chin Chen, Zhao-Yang Liu, and Hung-Yu Kao. 2017. IKM at SemEval-2017 Task 8: Convolutional neural networks for stance detection and rumor verification. In *Proceedings of the 11th International Workshop on Semantic Evaluation*. 465–469.

[8] Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language Modeling with Gated Convolutional Networks. In *Proceedings of the 34th International Conference on Machine Learning*. 933–941.

[9] Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. 2017. SemEval-2017 Task 8: RumourEval: Determining rumour veracity and support for rumours. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, 69–76. https://doi.org/10.18653/v1/S17-2006

[10] Cicero dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. 69–78.

[11] Johannes Fürnkranz. 1998. A study using n-gram features for text categorization. *Austrian Research Institute for Artifical Intelligence* 3, 1998 (1998), 1–10.

[12] Maria Glenski, Tim Weninger, and Svitlana Volkova. 2018. Identifying and Understanding User Reactions to Deceptive and Trusted Social News Sources. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 176–181.

[13] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. 2005. Bidirectional LSTM networks for improved phoneme classification and recognition. In *International Conference on Artificial Neural Networks*. Springer, 799–804.

[14] Gisel Bastidas Guacho, Sara Abdali, Neil Shah, and Evangelos E. Papalexakis. 2018. Semi-supervised Content-Based Detection of Misinformation via Tensor Embeddings. In *IEEE/ACM 2018 International Conference on Advances in Social Networks Analysis and Mining*. 322–325. https://doi.org/10.1109/ASONAM.2018.8508241

[15] Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 729–736.

[16] Frank Hansen and Gert K Pedersen. 2003. Jensen's operator inequality. *Bulletin of the London Mathematical Society* 35, 4 (2003), 553–564.

[17] Del Harvey and Yoel Roth. 2018. An Update On Our Elections Integrity Work. https://blog.twitter.com/official/en{_}us/topics/company/2018/an-update-on-our-elections-integrity-work.html

[18] Carl I Hovland and Walter Weiss. 1951. The influence of source credibility on communication effectiveness. *Public opinion quarterly* 15, 4 (1951), 635–650.

[19] Zhiwei Jin, Juan Cao, Yongdong Zhang, and Jiebo Luo. 2016. News verification by exploiting conflicting social viewpoints in microblogs. In *Thirtieth AAAI Conference on Artificial Intelligence*.

[20] Zhiwei Jin, Juan Cao, Yongdong Zhang, Jianshe Zhou, and Qi Tian. 2017. Novel visual and statistical image features for microblogs news verification. *IEEE transactions on multimedia* 19, 3 (2017), 598–608.

[21] Michal Kakol, Radoslaw Nielek, and Adam Wierzbicki. 2017. Understanding and predicting Web content credibility using the Content Credibility Corpus. *Information Processing & Management* 53, 5 (2017), 1043–1061.

[22] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 1746–1751.

[23] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[24] Diederik P. Kingma and Max Welling. 2013. Auto-Encoding Variational Bayes. *CoRR* abs/1312.6114 (2013). arXiv:1312.6114

[25] Elena Kochkina, Maria Liakata, and Arkaitz Zubiaga. 2018. All-in-one: Multi-task Learning for Rumour Verification. In *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 3402–3413. http://aclweb.org/anthology/C18-1288

[26] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, et al. 2013. Prominent features of rumor propagation in online social media. In *International Conference on Data Mining*. IEEE.

[27] Tessa Lyons. 2018. Increasing Our Efforts to Fight False News | Facebook Newsroom. https://newsroom.fb.com/news/2018/06/increasing-our-efforts-to-fight-false-news/

[28] Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 2015. Detect rumors using time series of social context information on microblogging websites. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1751–1754.

[29] Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor Detection on Twitter with Tree-structured Recursive Neural Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1980–1989.

[30] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[31] Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2018. DeClarE: Debunking Fake News and False Claims using Evidence-Aware Deep Learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 22–32.

[32] Chanthika Pornpitakpan. 2004. The persuasiveness of source credibility: A critical review of five decades' evidence. *Journal of applied social psychology* 34, 2 (2004), 243–281.

[33] Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. 2018. A Stylometric Inquiry into Hyperpartisan and Fake News. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 231–240.

[34] Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2931–2937.

[35] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Eric P. Xing and Tony Jebara (Eds.), Vol. 32. PMLR, Bejing, China, 1278–1286.

[36] Victoria Rubin, Niall Conroy, Yimin Chen, and Sarah Cornwell. 2016. Fake news or truth? using satirical cues to detect potentially misleading news. In *Proceedings of the Second Workshop on Computational Approaches to Deception Detection*. 7–17.

[37] Natali Ruchansky, Sungyong Seo, and Yan Liu. 2017. Csi: A hybrid deep model for fake news detection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 797–806.

[38] Nitish Srivastava. 2013. Improving neural networks with dropout. *University of Toronto* 182 (2013), 566.

[39] Robert Thomson, Naoya Ito, Hinako Suda, Fangyu Lin, Yafei Liu, Ryo Hayasaka, Ryuzo Isochi, and Zian Wang. 2012. Trusting tweets: The Fukushima disaster and information source credibility on Twitter. In *Proceedings of the 9th International ISCRAM Conference*. Vancouver: Simon Fraser University, 1–10.

[40] Shawn Tseng and BJ Fogg. 1999. Credibility and computing technology. *Commun. ACM* 42, 5 (1999), 39–44.

[41] Chong Wang and David M. Blei. 2013. Variational Inference in Nonconjugate Models. *J. Mach. Learn. Res.* 14, 1 (April 2013), 1005–1031.

[42] William Yang Wang. 2017. "Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 422–426. https://doi.org/10.18653/v1/P17-2067

[43] Aleksander Wawer, Radoslaw Nielek, and Adam Wierzbicki. 2014. Predicting webpage credibility using linguistic features. In *Proceedings of the 23rd international conference on world wide web*. ACM, 1135–1140.

[44] Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. 2012. Automatic detection of rumor on Sina Weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*. ACM, 13.

[45] Yang Yang, Lei Zheng, Jiawei Zhang, Qingcai Cui, Zhoujun Li, and Philip S Yu. 2018. TI-CNN: Convolutional Neural Networks for Fake News Detection. *arXiv preprint arXiv:1806.00749* (2018).

[46] Qiang Zhang, Shangsong Liang, Aldo Lipani, Zhaochun Ren, and Emine Yilmaz. 2019. From Stances' Imbalance to Their Hierarchical Representation and Detection. In *Companion Proceedings of the The Web Conference 2019*. ACM Press.

[47] Qiang Zhang, Emine Yilmaz, and Shangsong Liang. 2018. Ranking-based Method for News Stance Detection. In *Companion Proceedings of the The Web Conference 2018*. ACM Press.

[48] Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS one* 11, 3 (2016), e0150989.

Multilevel Network Alignment

Si Zhang
Arizona State University
szhan172@asu.edu

Hanghang Tong
Arizona State University
hanghang.tong@asu.edu

Ross Maciejewski
Arizona State University
rmacieje@asu.edu

Tina Eliassi-Rad
Northeastern University
tina@eliassi.org

ABSTRACT

Network alignment, which aims to find the node correspondence across multiple networks, is a fundamental task in many areas, ranging from social network analysis to adversarial activity detection. The state-of-the-art in the data mining community often view the node correspondence as a probabilistic cross-network node similarity, and thus inevitably introduce an $\Omega(n^2)$ lower bound on the computational complexity. Moreover, they might ignore the rich patterns (e.g., clusters) accompanying the real networks. In this paper, we propose a multilevel network alignment algorithm (Moana) which consists of three key steps. It first efficiently *coarsens* the input networks into their structured representations, and then *aligns* the coarsest representations of the input networks, followed by the *interpolations* to obtain the alignment at multiple levels including the node level at the finest granularity. The proposed *coarsen-align-interpolate* method bears two key advantages. First, it overcomes the $\Omega(n^2)$ lower bound, achieving a *linear* complexity. Second, it helps reveal the alignment between rich patterns of the input networks at multiple levels (e.g., node, clusters, super-clusters, etc.). Extensive experimental evaluations demonstrate the efficacy of the proposed algorithm on both the node-level alignment and the alignment among rich patterns (e.g., clusters) at different granularities.

KEYWORDS

Network alignment; multilevel alignment; multiresolution

ACM Reference Format:
Si Zhang, Hanghang Tong, Ross Maciejewski, and Tina Eliassi-Rad. 2019. Multilevel Network Alignment. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313484

1 INTRODUCTION

Networks are often *multi-sourced* (i.e., variety) and *large-scale* (i.e., volume), such as the financial networks built from the massive size of transactions in different institutes, the social networks with a substantial number of users on various platforms and so on. Whether or not subtle patterns associated with different networks (e.g., a complex fraud schema) can be distilled, profoundly depends on the successful integration of the multi-sourced network data. Network alignment that aims to uncover the node correspondence across different networks has drawn a lot of attention in many applications, including social network analysis [37], bioinformatics [10], adversarial activity detection [31], etc. For instance, financial criminals (e.g., money launderers [38, 39]) often camouflage themselves by frequently conducting less suspicious activities in different financial transaction networks, making it hard to detect them if we only focus on a single network. By finding the correspondence among the suspects across different networks, network alignment could help unveil such complex fraud transaction patterns.

Generally speaking, network alignment can be categorized into *local* alignment and *global* alignment. Among others, local network alignment essentially intends to find the similar motifs across different networks [3]. Such local alignment methods might be too restrictive to find the node correspondence, or to align the larger complex structures across different networks. On the other hand, a prevalent choice for global alignment in the data mining community is the soft alignment that views the node correspondence as a probabilistic cross-network node similarity (e.g., *IsoRank* [27], *FINAL* [35]). However, it suffers from the following drawbacks. First, the soft alignment inevitably introduces an $\Omega(n^2)$ lower bound on the computational complexity due to the dense alignment matrix, where n is the number of nodes. Second, most soft alignment methods (e.g., *BigAlign* [16]) assume one network is a noisy permutation of the other and perform the alignment at a single node level. Consequently, these approaches might ignore the rich patterns accompanying networks, e.g., clusters, etc.

Some recent efforts attempt to go beyond this assumption and perform the alignment at two levels [2, 20]. In addition to the node-level alignment, these methods aim to find the alignment at the coarser level, for instance, among clusters. Compared to the node-level alignment, these methods not only break the $\Omega(n^2)$ floor of the computational complexity by viewing the node-level alignment as the interpolation of the cluster-level alignment, but also reveal the more complicated alignment across the input networks (e.g., which cluster in network $\mathcal{G}_1$ links to which cluster in network $\mathcal{G}_2$).

Nonetheless, these two-level alignment methods still bear some fundamental limitations. First (*problem setting*), the existing methods are restricted to only two levels, and therefore might overlook some richer patterns in the underlying networks, e.g., the hierarchical *cluster-within-clusters* structure [25]. Second (*alignment accuracy*), it is not clear how the coarser-level alignment would impact the accuracy of the finer-level alignment. This is crucial especially if we want to perform the multilevel alignment, as the error at one level could be propagated or even diverged through different levels. Third (*computational efficiency*), most of the existing

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313484

two-level approaches, with the only exception of [36], still requires a *sub-quadratic* computational complexity, e.g., [2].

In this paper, we aim to systematically address the above limitations and the main contributions are summarized as follows:

- **Problem Definition.** To our best knowledge, we are the first to address the multilevel network alignment problem.
- **Algorithm and Analysis.** We propose a multilevel network alignment algorithm (MOANA), which scales *linearly* w.r.t. the size of the input networks. In theory, we prove a condition on the perfect interpolation, which in principle allows us to perform the alignment at an arbitrary number of levels, without any interpolation error. In practice, we drive an error bound of alignment due to the coarsening step.
- **Experiments.** We perform extensive experiments, which demonstrate that (1) the proposed algorithm achieves a close node-level alignment accuracy to its single-level counterpart *FINAL* [35] with an up to 10× speedup, in the meanwhile outperforming all the other existing node-level alignment algorithms. Note that the proposed MOANA can be considered as an approximation of *FINAL* at the finest node level; (2) it scales *linearly* w.r.t. the size of the networks; and (3) it also reveals the alignment among the more complicated cluster structures at different granularities.

2 PROBLEM DEFINITION

Table 1 summarizes the main symbols and notations used throughout the paper. We use the bold uppercase letters to denote matrices (e.g., $\mathbf{A}$), bold lowercase letters for vectors (e.g., $\mathbf{s}$) and letters not in bold for scalars (e.g., α). We use $\mathbf{A}(i,j)$ to denote the entry at the intersection of the i-th row and j-th column of the matrix $\mathbf{A}$. We denote the transpose of a matrix by a superscript T (e.g., $\mathbf{A}^T$ as the transpose of $\mathbf{A}$). We use the subscripts to index the matrices at different levels. For example, $\mathbf{A}_1$ represents the symmetrically normalized matrix of the adjacency matrix $\bar{\mathbf{A}}_1$ of network $\mathcal{G}_1$ at the first level (i.e., $\mathbf{A}_1 = \mathbf{D}_1^{-\frac{1}{2}}\bar{\mathbf{A}}_1\mathbf{D}_1^{-\frac{1}{2}}$ where $\mathbf{D}_1$ is the diagonal degree matrix of $\bar{\mathbf{A}}_1$), and $\mathbf{A}_l$ is the corresponding coarsened adjacency matrix at the l-th level ($l \geq 2$). The identity matrix of size $n \times n$ is denoted by $\mathbf{I}_n$. Throughout this paper, we simplify $\mathbf{I}_n$ to $\mathbf{I}$ if the size is clear from the context. Without loss of generality, we assume that $\mathcal{G}_1$ and $\mathcal{G}_2$ share a comparable size, i.e., $O(n_1) = O(n_2) = O(n)$ and $O(m_1) = O(m_2) = O(m)$, to simplify the complexity analysis.

2.1 Multilevel Network Alignment Problem

Many large real-world networks often have the complex hierarchical *cluster-within-clusters* structures. For example, given the input network $\mathcal{G}_1$ in Figure 1 (a), nodes $\{1,3\}$ can be viewed as a two-node cluster and similarly nodes $\{2,12\}$ form another two-node cluster (in Figure 1 (b)). These two clusters are represented by node-1 and node-2 respectively at the second level, which further construct an even coarser two-node cluster (in Figure 1 (c)). This coarser cluster is represented by node-1 at the third level (in Figure 1 (d)). As one can see, each level reveals a unique pattern in the underlying network $\mathcal{G}_1$. This leads to the following question for the alignment task: *How can we align the clusters at different granularities across the input networks?* Such an alignment can not only uncover how the networks are aligned at the different cluster levels, but also might help the node-level alignment under the following hypothesis. If the cluster represented by node-1 in $\mathcal{G}_1$ is aligned with that represented by node-6′ in $\mathcal{G}_2$ at the coarsest level (in Figure 1 (e)), it might provide us with clues on how the two-node clusters represented by node-1 and node-2 are aligned with those represented by node-6′ and node-3′ at the next finer level (i.e., level-2). Furthermore, it might indicate how nodes $\{1,2,3,12\}$ are aligned with $\{6',3',11',13'\}$ at the original node level (in Figure 1 (f)). Formally, we define the multilevel network alignment problem.

Table 1: Symbols and Notations

Symbols	Definition
$\mathcal{G}_1$, $\mathcal{G}_2$	the input undirected networks
$\bar{\mathbf{A}}_1$, $\bar{\mathbf{B}}_1$	the adjacency matrices of $\mathcal{G}_1$ and $\mathcal{G}_2$
$\mathbf{A}_1$, $\mathbf{B}_1$	the symmetric normalization by degree matrix of $\bar{\mathbf{A}}_1$, $\bar{\mathbf{B}}_1$
$\mathbf{A}_l$, $\mathbf{B}_l$	the coarsened adjacency matrices at the l-th level
$\mathbf{P}_l$, $\mathbf{Q}_l$	the interpolation matrices at the l-th level
$\mathbf{H}_l$	the $n_2 \times n_1$ prior similarity matrix at the l-th level
$\mathbf{S}_l^*$	the output $n_2 \times n_1$ alignment matrix at the l-th level
$\mathcal{S}_{A_l}$, $\mathcal{S}_{B_l}$	the active indices of $\mathbf{A}_l$, $\mathbf{B}_l$ at the l-th level
n_1, n_2	# of nodes in $\mathcal{G}_1$, $\mathcal{G}_2$
m_1, m_2	# of edges in $\mathcal{G}_1$, $\mathcal{G}_2$
L	# of levels
$\mathrm{Tr}(\cdot)$, $\mathrm{nnz}(\cdot)$	the trace and the number of nonzero elements of a matrix
$\mathrm{vec}(\cdot)$, $\mathrm{mat}(\cdot)$	the vectorization and de-vectorization operators
$\otimes$	the Kronecker product operator

PROBLEM 1. *MULTILEVEL NETWORK ALIGNMENT PROBLEM.*

Given: *(1) adjacency matrices $\bar{\mathbf{A}}_1, \bar{\mathbf{B}}_1$ of two undirected (weighted) sparse networks $\mathcal{G}_1, \mathcal{G}_2$, (2) a sparse prior node similarity matrix $\mathbf{H}_1$ across two networks, and (3) the number of levels $L \geq 2$.*

Output: *a set of $n_2 \times n_1$ alignment matrices $\mathbf{S}_l^*$, $l = 1, \cdots, L$, where specifically $\mathbf{S}_1^*(x,a)$ represents the confidence that node-a in $\mathcal{G}_1$ is aligned with node-x in $\mathcal{G}_2$ at the input node level.*

2.2 Preliminaries #1: Node-Level Alignment

A classic node-level alignment algorithm is attributed to *IsoRank* [27] based on the *alignment consistency* by random walks on the product graph. With an additional normalization, it has the following optimization interpretation referred to as *FINAL* in [35].

$$\min_{\mathbf{s}_1} \ \alpha \mathbf{s}_1^T(\mathbf{I} - \mathbf{A}_1 \otimes \mathbf{B}_1)\mathbf{s}_1 + (1-\alpha)\|\mathbf{s}_1 - \mathbf{h}_1\|_2^2 \tag{1}$$

where $\mathbf{s}_1, \mathbf{h}_1$ are the vectorization of the alignment matrix $\mathbf{S}_1$ and the prior similarity matrix $\mathbf{H}_1$, respectively. $\mathbf{A}_1$ and $\mathbf{B}_1$ are the symmetrically normalized adjacency matrices by the degree matrices of $\bar{\mathbf{A}}_1$ and $\bar{\mathbf{B}}_1$ respectively. Besides, the prior node similarity $\mathbf{h}_1$ is to encode the prior alignment information (e.g., anchor links across two networks) and prevent trivial solutions (e.g., $\mathbf{s}_1 = \mathbf{0}$). When such anchor links are absent, we can construct it by some heuristics, e.g., degree similarity. Our proposed algorithm will be based on Eq. (1). The optimization in Eq. (1) is convex so that a global optimal solution can be obtained by the fixed-point algorithm:

$$\mathbf{S}_1 = \alpha \mathbf{B}_1 \mathbf{S}_1 \mathbf{A}_1 + (1-\alpha)\mathbf{H}_1 \tag{2}$$

with an $O(n^2)$ time complexity even with approximations [35].

2.3 Preliminaries #2: MMF

Although many multilevel methods (e.g., hierarchical clustering, algebraic multigrid) exist for mining a single network, it turns out that multiresolution matrix factorization (MMF) [15, 29] is particularly suited for network alignment task for the reason that we will explain in Section 3.2. Here, we give a short review of MMF.

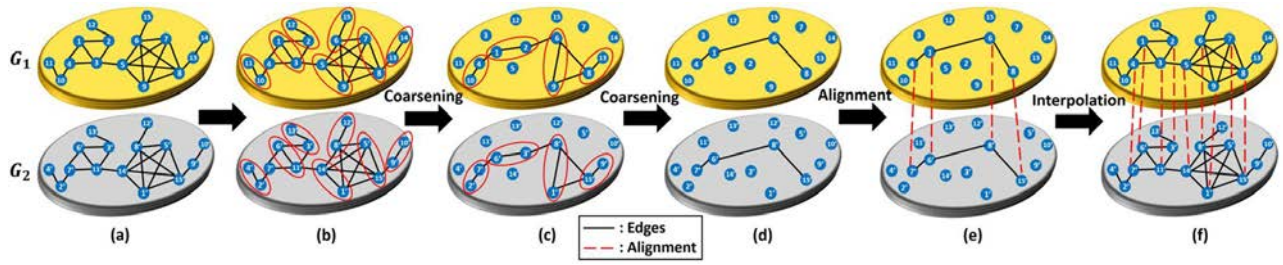


Figure 1: An illustrative example. (a) shows the input networks. (b)-(d) show the network coarsening process. (e) presents the alignment at the coarsest level and (f) shows parts of the node-level alignment across the input networks after interpolations.

Given a symmetric matrix $\mathbf{A}_1 \in R^{n_1 \times n_1}$, MMF explores the hierarchical cluster-within-clusters structure underlying the matrix $\mathbf{A}_1$ and factorizes $\mathbf{A}_1$ into several sparse orthogonal matrices and a core-diagonal matrix. This multilevel factorization is of the form

$$\mathbf{P}_{L-1} \cdots \mathbf{P}_1 \mathbf{A}_1 \mathbf{P}_1^T \cdots \mathbf{P}_{L-1}^T = \mathbf{A}_L \approx \tilde{\mathbf{A}}_L \tag{3}$$

where $\mathbf{P}_1, \cdots, \mathbf{P}_{L-1}$ are the sparse orthogonal matrices and the matrix $\tilde{\mathbf{A}}_L$ is the $\mathcal{S}_L$-core-diagonal matrix defined as below.

Definition 1. ***$\mathcal{S}_L$-core-diagonal matrix [15].*** *Denote a set of active indices as $\mathcal{S}_L = \{i_1, \cdots, i_k\} \subset \{1, \cdots, n_1\}$. Then a matrix $\tilde{\mathbf{A}}_L$ is defined as the $\mathcal{S}_L$-core-diagonal matrix under the condition that $\tilde{\mathbf{A}}_L(i,j) = 0$ unless $i = j$ or $i, j \in \mathcal{S}_L$.*

Throughout this paper, we refer to $\tilde{\mathbf{A}}_L(\mathcal{S}_L, \mathcal{S}_L)$ as the core matrix of $\tilde{\mathbf{A}}_L$. Note that $\tilde{\mathbf{A}}_L(\mathcal{S}_L, \mathcal{S}_L) = \mathbf{A}_L(\mathcal{S}_L, \mathcal{S}_L)$ according to how $\tilde{\mathbf{A}}_L$ is generated (see Appendix A). To compute these sparse orthogonal matrices, [15, 29] consider using the elementary rotation and compound rotation matrices. Among others, using the compound rotation matrices (see Appendix A) yields a more interpretable and compact hierarchical structure, which commonly resides in the underlying network. Therefore, in this paper, we use the parallel MMF with the compound rotations.

3 MULTILEVEL NETWORK ALIGNMENT

In this section, we present our solution to Problem 1. First, we present our multilevel network alignment framework from the optimization perspective. We analyze the theoretical condition on the perfect interpolation which allows no interpolation error across multiple levels. Then, we introduce the details of the proposed algorithm Moana, followed by further analysis of our algorithm.

3.1 Multilevel Optimization Formulation

Generally speaking, most of the existing multilevel approaches to graph mining problems (e.g., graph partitioning) follow similar strategies. They first hierarchically coarsen the network so that at the coarsest level, an approximation solution to the original problem can be obtained within an insignificant time. This approximation solution will then be sequentially projected to the finer levels by the *interpolation* matrices. Inspired by this generic strategy, our key idea is to use the interpolation matrices to efficiently estimate the alignment matrix at the finer level (e.g., $\mathbf{S}_l$ at the l-th level) from that at the next coarser level (e.g., $\mathbf{S}_{l+1}$). Since the alignment matrices hinge on two networks, different from the interpolations underlying a single network, the bilinear interpolations are required.

To derive the optimization formulation for our multilevel network alignment problem, without loss of generality, we focus on the first two levels for now. Denote two interpolation matrices $\mathbf{P}_1 \in R^{p_1 \times n_1}$ and $\mathbf{Q}_1 \in R^{q_1 \times n_2}$ where $p_1 \leq n_1, q_1 \leq n_2$ such that we can approximate the node-level alignment matrix $\mathbf{S}_1$ by $\mathbf{S}_1 = \mathbf{Q}_1^T \mathbf{S}_2 \mathbf{P}_1$ where $\mathbf{S}_2 \in R^{q_1 \times p_1}$ is the alignment matrix at the second level. By de-vectorization on $\mathbf{s}_1, \mathbf{h}_1$, Eq. (1) is equivalent to

$$\min_{\mathbf{S}_1} \ \alpha[\mathrm{Tr}(\mathbf{S}_1^T \mathbf{S}_1) - \mathrm{Tr}(\mathbf{S}_1^T \mathbf{B}_1 \mathbf{S}_1 \mathbf{A}_1)] + (1-\alpha)\|\mathbf{S}_1 - \mathbf{H}_1\|_F^2 \tag{4}$$

Plugging in $\mathbf{S}_1 = \mathbf{Q}_1^T \mathbf{S}_2 \mathbf{P}_1$, we have the objective function w.r.t. $\mathbf{S}_2$.

$$\begin{aligned} J(\mathbf{S}_2) =& \alpha[\mathrm{Tr}(\mathbf{P}_1^T \mathbf{S}_2^T \mathbf{Q}_1 \mathbf{Q}_1^T \mathbf{S}_2 \mathbf{P}_1) - \mathrm{Tr}(\mathbf{S}_2^T \mathbf{Q}_1 \mathbf{B}_1 \mathbf{Q}_1^T \mathbf{S}_2 \mathbf{P}_1 \mathbf{A}_1 \mathbf{P}_1^T)] \\ &+ (1-\alpha)\|\mathbf{Q}_1^T \mathbf{S}_2 \mathbf{P}_1 - \mathbf{H}_1\|_F^2 \end{aligned} \tag{5}$$

Notice that if the (semi-) orthogonality satisfies, i.e., $\mathbf{P}_1 \mathbf{P}_1^T = \mathbf{I}$ and $\mathbf{Q}_1 \mathbf{Q}_1^T = \mathbf{I}$, we can obtain the objective function at the second level which is of exactly the same form as Eq. (4), i.e.,

$$J(\mathbf{S}_2) = \alpha[\mathrm{Tr}(\mathbf{S}_2^T \mathbf{S}_2) - \mathrm{Tr}(\mathbf{S}_2^T \mathbf{B}_2 \mathbf{S}_2 \mathbf{A}_2)] + (1-\alpha)\|\mathbf{S}_2 - \mathbf{H}_2\|_F^2 \tag{6}$$

where $\mathbf{A}_2 = \mathbf{P}_1 \mathbf{A}_1 \mathbf{P}_1^T$, $\mathbf{B}_2 = \mathbf{Q}_1 \mathbf{B}_1 \mathbf{Q}_1^T$ and $\mathbf{H}_2 = \mathbf{Q}_1 \mathbf{H}_1 \mathbf{P}_1^T$. Equivalently, this can be viewed as coarsening $\mathbf{A}_1, \mathbf{B}_1$ into $\mathbf{A}_2, \mathbf{B}_2$ to be aligned at the second level by the interpolation matrices $\mathbf{P}_1$ and $\mathbf{Q}_1$, with the corresponding prior node similarity matrix $\mathbf{H}_2$.

In fact, the relationship between Eq. (4) and Eq. (6) applies to any two adjacent levels among all the L levels. This implies that as we coarsen the symmetrically normalized adjacency matrices of the input networks (i.e., $\mathbf{A}_1, \mathbf{B}_1$) into the coarsest level (i.e., $\mathbf{A}_L, \mathbf{B}_L$), we are always solving the same optimization problem but w.r.t. the different variables to find the alignment at different levels (e.g., $\mathbf{S}_L$). The benefits of this are three-fold. First, this ensures that the same theoretical properties and optimization algorithms at the finest node level (e.g., the convexity, the optimality, the convergence from [35]) will immediately apply to the coarse levels as well. Second, similar to most multilevel approaches, solving the alignment problem at the coarsest level is computationally more efficient. Third, provided that a set of 'good' (semi-) orthogonal interpolation matrices $\mathbf{P}_l, \mathbf{Q}_l$ can be found, we are able to obtain a set of well-represented adjacency matrices at the coarse levels that reveal the hierarchical cluster-within-clusters structure of the input networks.

3.2 Perfect Interpolation

Despite a large number of the proposed graph coarsening methods, to our best knowledge, few of them bear the (semi-) orthogonality of the interpolation matrices. For example, neither the heavy-edge matching [6, 13] nor the algebraic multigrid (AMG) coarsening [23] guarantees the semi-orthogonality. Moreover, since there are more variables at the finer levels than coarser levels, interpolation might introduce an interpolation error to the alignment matrix at the finer level. Even worse, in the multilevel setting, such an interpolation error could propagate or even diverge through different levels so that the node-level alignment might be sub-optimal or even misleading.

In this paper, instead of exploring the semi-orthogonal interpolation matrices, we seek to find a set of orthogonal matrices, i.e., $\mathbf{P}_l\mathbf{P}_l^T = \mathbf{I}$ and $\mathbf{P}_l^T\mathbf{P}_l = \mathbf{I}$. Indeed, by the following lemma, we show that the orthogonal interpolation matrices guarantee that the interpolation of the optimal alignment matrix from the coarser level is exactly the same as the optimal alignment matrix at the finer level.

Lemma 1. ***Perfect Interpolation.*** *The global optimal solution to the optimization problem at the finer level (e.g., Eq.* (4) *for level-1), denoted by* $\mathbf{S}_l$*, is exactly same as the interpolation of the optimal solution at the next coarser level (denoted by* $\mathbf{S}_{l+1}$*). That is,* $\mathbf{S}_l = \mathbf{Q}_l^T\mathbf{S}_{l+1}\mathbf{P}_l$ *if* $\mathbf{P}_l$ *and* $\mathbf{Q}_l$ *are orthogonal, where* $l = 1, \cdots, L-1$.

Proof. Without loss of generality, we prove $\mathbf{S}_1 = \mathbf{Q}_1^T\mathbf{S}_2\mathbf{P}_1$. For all the other levels, it is straightforward to show the same result. The optimal closed-form solution to Eq. (1) (and equivalently Eq. (4)) is $\mathbf{s}_1 = (1-\alpha)(\mathbf{I} - \alpha\mathbf{A}_1 \otimes \mathbf{B}_1)^{-1}\mathbf{h}_1$. Similarly, the alignment matrix between $\mathbf{A}_2, \mathbf{B}_2$ at the second level is computed by

$$\mathbf{s}_2 = (1-\alpha)[\mathbf{I} - \alpha(\mathbf{P}_1\mathbf{A}_1\mathbf{P}_1^T) \otimes (\mathbf{Q}_1\mathbf{B}_1\mathbf{Q}_1^T)]^{-1}\mathbf{h}_2$$

The difference between $\mathbf{S}_1$ and the interpolated alignment from $\mathbf{S}_2$ measured in the Frobenius norm is

$$\begin{aligned}
&\|\mathbf{Q}_1^T\mathbf{S}_2\mathbf{P}_1 - \mathbf{S}_1\|_F = \|(\mathbf{P}_1^T \otimes \mathbf{Q}_1^T)\mathbf{s}_2 - \mathbf{s}_1\|_2 \\
=&(1-\alpha)\|(\mathbf{P}_1^T \otimes \mathbf{Q}_1^T)[\mathbf{I} - \alpha(\mathbf{P}_1\mathbf{A}_1\mathbf{P}_1^T) \otimes (\mathbf{Q}_1\mathbf{B}_1\mathbf{Q}_1^T)]^{-1}\mathbf{h}_2 \\
&-(\mathbf{I} - \alpha\mathbf{A}_1 \otimes \mathbf{B}_1)^{-1}\mathbf{h}_1\|_2 \\
=&(1-\alpha)\Big\|\sum_{k=0}^{\infty}\alpha^k(\mathbf{P}_1^T \otimes \mathbf{Q}_1^T)[(\mathbf{P}_1\mathbf{A}_1\mathbf{P}_1^T)^k \otimes (\mathbf{Q}_1\mathbf{B}_1\mathbf{Q}_1^T)^k](\mathbf{P}_1 \otimes \mathbf{Q}_1)\mathbf{h}_1 \\
&-\sum_{k=0}^{\infty}\alpha^k(\mathbf{A}_1^k \otimes \mathbf{B}_1^k)\mathbf{h}_1\Big\|_2
\end{aligned}$$

where the third equation is by Neumann series due to the fact that (1) the eigenvalues of $\mathbf{A}_1, \mathbf{B}_1$ are in the range of $[-1, 1]$, and (2) $\mathbf{P}_1\mathbf{A}_1\mathbf{P}_1^T, \mathbf{Q}_1\mathbf{B}_1\mathbf{Q}_1^T$ share the same eigenvalues as $\mathbf{A}_1, \mathbf{B}_1$ respectively given that $\mathbf{P}_1$ and $\mathbf{Q}_1$ are orthogonal [24].

Due to the orthogonality of $\mathbf{P}_1, \mathbf{Q}_1$, the following equations hold.

$$\begin{aligned}
\mathbf{A}_1^k =&\mathbf{P}_1^T(\mathbf{P}_1\mathbf{A}_1\mathbf{P}_1^T)^k\mathbf{P}_1 \\
\mathbf{B}_1^k =&\mathbf{Q}_1^T(\mathbf{Q}_1\mathbf{B}_1\mathbf{Q}_1^T)^k\mathbf{Q}_1
\end{aligned}$$

Thus, we have that

$$\|\mathbf{S}_1 - \mathbf{Q}_1^T\mathbf{S}_2\mathbf{P}_1\|_F^2 = (1-\alpha)\Big\|\sum_{k=0}^{\infty}\alpha^k(\mathbf{A}_1^k \otimes \mathbf{B}_1^k - \mathbf{A}_1^k \otimes \mathbf{B}_1^k)\mathbf{h}_1\Big\|_2 = 0$$

which completes the proof. □

3.3 Multilevel Alignment Algorithm

Our multilevel alignment algorithm Moana contains three phases, including *coarsening* the input networks, *aligning* across the coarsest networks, and then *interpolating* the alignment matrix from the coarser levels to the finer levels.

A - Network coarsening. Based on Lemma 1, we want to find a set of orthogonal matrices to coarsen the input networks. However, even though the input adjacency matrix, say $\mathbf{A}_1$, is sparse, the computational complexity of $\mathbf{P}_1\mathbf{A}_1\mathbf{P}_1^T$ may still be $O(n^3)$ at worst if $\mathbf{P}_1$ is not sufficiently sparse. With an arbitrarily orthogonal matrix, the resulting matrix $\mathbf{A}_2$ at the next coarser level could be less informative and thus could fail to reveal the underlying structural information of $\mathbf{A}_1$. Thus, we seek to find a set of orthogonal interpolation matrices $\mathbf{P}_l$ and $\mathbf{Q}_l$ such that (1) they are sufficiently sparse, and (2) they are able to uncover the *hierarchical cluster-within-clusters* structure of the input networks. In this paper, we leverage a recently proposed multiresolution matrix factorization (MMF) algorithm that satisfies these requirements [15]. Specifically, for each input network, we use the parallel second order MMF algorithm (i.e., $k_1, \cdots, k_p \le 2$ in Eq. (25)) to find a set of rotation matrices $\mathbf{P}_l, \mathbf{Q}_l$ such that at the l-th level ($l \ge 2$),

$$\mathbf{A}_l = \mathbf{P}_{l-1}\cdots\mathbf{P}_1\mathbf{A}_1\mathbf{P}_1^T\cdots\mathbf{P}_{l-1}^T \tag{7}$$

$$\mathbf{B}_l = \mathbf{Q}_{l-1}\cdots\mathbf{Q}_1\mathbf{B}_1\mathbf{Q}_1^T\cdots\mathbf{Q}_{l-1}^T \tag{8}$$

where the active indices of $\mathbf{A}_l, \mathbf{B}_l$ are denoted as $\mathcal{S}_{A_l}$ of size λ_l and $\mathcal{S}_{B_l}$ of size μ_l, respectively. Specifically, at the coarsest level, the rotated matrices are denoted as $\mathbf{A}_L, \mathbf{B}_L$. Then we form the core-diagonal matrices $\tilde{\mathbf{A}}_L$ and $\tilde{\mathbf{B}}_L$ as detailed in Appendix A.

Remarks. Different from the traditional coarsening methods, such as the AMG-based methods, the MMF algorithm can be viewed as coarsening the network without reducing the size of the adjacency matrix. Due to its root in multiresolution analysis on networks, MMF offers a natural way to explore the *cluster-within-clusters* structure, which we elaborate as follows. Generally speaking, the multiresolution analysis on networks (e.g., diffusion wavelet [4]) aims to find a sequence of smoother spaces of the network. Each space captures the corresponding structured representation of the network (e.g., clusters) at a certain level by its basis (i.e., scaling functions) and the scaling coefficients. Specifically, in MMF, the scaling functions of $\mathbf{A}_1$ at the l-th level are formed by the rows of $\mathbf{P}_{l-1}\cdots\mathbf{P}_1$ indexed by $\mathcal{S}_{A_l}$, and $\mathbf{A}_l(\mathcal{S}_{A_l}, \mathcal{S}_{A_l})$ acts as the scaling coefficients. In fact, each row/column of $\mathbf{A}_l(\mathcal{S}_{A_l}, \mathcal{S}_{A_l})$ can be viewed as a node represented in the corresponding smooth space (or equivalently, clusters at level l) and these coefficients measure the strength of the interactions among the clusters at level l. Similarly, at the $(l+1)$-th level, the clusters at level l tend to be further clustered and hence MMF can explore the *cluster-within-clusters* structure of the networks. Thus, $\tilde{\mathbf{A}}_L, \tilde{\mathbf{B}}_L$ capture the coarsest cluster structures of the input networks.

B - Alignment across the coarsest networks. After the coarsening step (e.g., Eq. (3) on $\mathbf{A}_1$), the symmetrically normalized adjacency matrices of the input networks are transformed into the corresponding core-diagonal matrices, i.e.,

$$\tilde{\mathbf{A}}_L = \mathbf{\Pi}_A\begin{bmatrix}\tilde{\mathbf{A}}_{L_1} & \mathbf{0}\\ \mathbf{0} & \tilde{\mathbf{A}}_{L_2}\end{bmatrix}\mathbf{\Pi}_A^T,\ \tilde{\mathbf{B}}_L = \mathbf{\Pi}_B\begin{bmatrix}\tilde{\mathbf{B}}_{L_1} & \mathbf{0}\\ \mathbf{0} & \tilde{\mathbf{B}}_{L_2}\end{bmatrix}\mathbf{\Pi}_B^T \tag{9}$$

where $\tilde{\mathbf{A}}_{L_1} = \tilde{\mathbf{A}}_L(\mathcal{S}_{A_L}, \mathcal{S}_{A_L})$ and $\tilde{\mathbf{B}}_{L_1} = \tilde{\mathbf{B}}_L(\mathcal{S}_{B_L}, \mathcal{S}_{B_L})$ are the core matrices of $\tilde{\mathbf{A}}_L$ and $\tilde{\mathbf{B}}_L$ respectively. $\mathbf{\Pi}_A, \mathbf{\Pi}_B$ are the orthogonal permutation matrices to reorder the active indices of the matrices $\tilde{\mathbf{A}}_L, \tilde{\mathbf{B}}_L$ to be in the upper left part for the illustration purpose. Denote the inactive indices as $\overline{\mathcal{S}}_{A_L} = \{1, \cdots, n_1\} \setminus \mathcal{S}_{A_L}$ and $\overline{\mathcal{S}}_{B_L} = \{1, \cdots, n_2\} \setminus \mathcal{S}_{B_L}$. Accordingly, $\tilde{\mathbf{A}}_{L_2} = \tilde{\mathbf{A}}_L(\overline{\mathcal{S}}_{A_L}, \overline{\mathcal{S}}_{A_L})$ and $\tilde{\mathbf{B}}_{L_2} = \tilde{\mathbf{B}}_L(\overline{\mathcal{S}}_{B_L}, \overline{\mathcal{S}}_{B_L})$. Note that our algorithm does not need to explicitly compute such permutation matrices.

To solve the alignment between $\tilde{\mathbf{A}}_L$ and $\tilde{\mathbf{B}}_L$ at the coarsest level, we use the iterative fixed-point algorithm similar to Eq. (2).

$$\mathbf{S}_L = \alpha\tilde{\mathbf{B}}_L\mathbf{S}_L\tilde{\mathbf{A}}_L + (1-\alpha)\mathbf{H}_L \tag{10}$$

where $\mathbf{H}_L = \mathbf{Q}_{L-1}\cdots\mathbf{Q}_1\mathbf{H}_1\mathbf{P}_1^T\cdots\mathbf{P}_{L-1}^T$ is the corresponding prior similarity matrix at the coarsest level. By using the permutation matrices $\mathbf{\Pi}_A, \mathbf{\Pi}_B$, Eq. (10) can be rewritten as

$$\mathbf{\Pi}_B^T\mathbf{S}_L\mathbf{\Pi}_A = \alpha(\mathbf{\Pi}_B^T\tilde{\mathbf{B}}_L\mathbf{\Pi}_B)(\mathbf{\Pi}_B^T\mathbf{S}_L\mathbf{\Pi}_A)(\mathbf{\Pi}_A^T\tilde{\mathbf{A}}_L\mathbf{\Pi}_A^T)+(1-\alpha)\mathbf{\Pi}_B^T\mathbf{H}_L\mathbf{\Pi}_A$$

By denoting $\tilde{\mathbf{S}}_L = \mathbf{\Pi}_B^T\mathbf{S}_L\mathbf{\Pi}_A$ and $\tilde{\mathbf{H}}_L = \mathbf{\Pi}_B^T\mathbf{H}_L\mathbf{\Pi}_A$, the computation can be simplified to

$$\tilde{\mathbf{S}}_L = \alpha\begin{bmatrix}\tilde{\mathbf{B}}_{L_1} & \mathbf{0}\\ \mathbf{0} & \tilde{\mathbf{B}}_{L_2}\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{S}}_{L_1} & \tilde{\mathbf{S}}_{L_2}\\ \tilde{\mathbf{S}}_{L_3} & \tilde{\mathbf{S}}_{L_4}\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{A}}_{L_1} & \mathbf{0}\\ \mathbf{0} & \tilde{\mathbf{A}}_{L_2}\end{bmatrix}+(1-\alpha)\begin{bmatrix}\tilde{\mathbf{H}}_{L_1} & \tilde{\mathbf{H}}_{L_2}\\ \tilde{\mathbf{H}}_{L_3} & \tilde{\mathbf{H}}_{L_4}\end{bmatrix}$$

which allows the computation to be block-wise as follows:

$$\tilde{\mathbf{S}}_{L_1} = \alpha\tilde{\mathbf{B}}_{L_1}\tilde{\mathbf{S}}_{L_1}\tilde{\mathbf{A}}_{L_1} + (1-\alpha)\tilde{\mathbf{H}}_{L_1} \tag{11}$$

$$\tilde{\mathbf{S}}_{L_2} = \alpha\tilde{\mathbf{B}}_{L_1}\tilde{\mathbf{S}}_{L_2}\tilde{\mathbf{A}}_{L_2} + (1-\alpha)\tilde{\mathbf{H}}_{L_2} \tag{12}$$

$$\tilde{\mathbf{S}}_{L_3} = \alpha\tilde{\mathbf{B}}_{L_2}\tilde{\mathbf{S}}_{L_3}\tilde{\mathbf{A}}_{L_1} + (1-\alpha)\tilde{\mathbf{H}}_{L_3} \tag{13}$$

$$\tilde{\mathbf{S}}_{L_4} = \alpha\tilde{\mathbf{B}}_{L_2}\tilde{\mathbf{S}}_{L_4}\tilde{\mathbf{A}}_{L_2} + (1-\alpha)\tilde{\mathbf{H}}_{L_4} \tag{14}$$

Armed with the iterative fixed-point algorithm, the global optimal solutions to Eq. (11), Eq. (12) and Eq. (13) can be achieved and are denoted as $\tilde{\mathbf{S}}^*_{L_1}, \tilde{\mathbf{S}}^*_{L_2}, \tilde{\mathbf{S}}^*_{L_3}$ respectively. Furthermore, since both $\tilde{\mathbf{A}}_{L_2}$ and $\tilde{\mathbf{B}}_{L_2}$ are sparse diagonal matrices, the closed-form optimal solution of Eq. (14) can be easily computed by

$$\tilde{\mathbf{s}}^*_{L_4} = (1-\alpha)(\mathbf{I}-\alpha\tilde{\mathbf{A}}_{L_2}\otimes\tilde{\mathbf{B}}_{L_2})^{-1}\tilde{\mathbf{h}}_{L_4} \tag{15}$$

where $\tilde{\mathbf{s}}^*_{L_4} = \mathrm{vec}(\tilde{\mathbf{S}}^*_{L_4}), \tilde{\mathbf{h}}_{L_4} = \mathrm{vec}(\tilde{\mathbf{H}}_{L_4})$ and the operator $\otimes$ represents the Kronecker product. In this way, by the permutation matrices $\mathbf{\Pi}_A, \mathbf{\Pi}_B$, the optimal solution to the alignment problem at the coarsest level $\mathbf{S}^*_L$ is composed of $\mathbf{S}^*_L(\mathcal{S}_{B_L}, \mathcal{S}_{B_L})$, $\mathbf{S}^*_L(\mathcal{S}_{B_L}, \overline{\mathcal{S}}_{A_L})$, $\mathbf{S}^*_L(\overline{\mathcal{S}}_{B_L}, \mathcal{S}_{A_L})$ and $\mathbf{S}^*_L(\overline{\mathcal{S}}_{B_L}, \overline{\mathcal{S}}_{A_L})$ where

$$\begin{aligned}\mathbf{S}^*_L(\mathcal{S}_{B_L}, \mathcal{S}_{A_L}) &= \tilde{\mathbf{S}}^*_{L_1}, & \mathbf{S}^*_L(\mathcal{S}_{B_L}, \overline{\mathcal{S}}_{A_L}) &= \tilde{\mathbf{S}}^*_{L_2}\\ \mathbf{S}^*_L(\overline{\mathcal{S}}_{B_L}, \mathcal{S}_{A_L}) &= \tilde{\mathbf{S}}^*_{L_3}, & \mathbf{S}^*_L(\overline{\mathcal{S}}_{B_L}, \overline{\mathcal{S}}_{A_L}) &= \tilde{\mathbf{S}}^*_{L_4}\end{aligned} \tag{16}$$

Note that though the core matrices $\tilde{\mathbf{A}}_{L_1}, \tilde{\mathbf{B}}_{L_1}$ could be dense, their sizes $\lambda_L \times \lambda_L$ and $\mu_L \times \mu_L$ are very small (i.e., $\lambda_L \ll n_1, \mu_L \ll n_2$), which can be controlled by the parameter L. This is due to the fact of parallel MMF that the size of the core matrix decreases exponentially (e.g., $\lambda_L \approx 2^{-L}n_1$). Empirically, we find that λ_L, μ_L in the order of hundreds often achieves a good balance between effectiveness and efficiency, as we will show in Section 4.. Moreover, thanks to the fact that both $\tilde{\mathbf{A}}_{L_2}, \tilde{\mathbf{B}}_{L_2}$ are diagonal, the above computations are very efficient as we will show in the next subsection.

C - Interpolation. After the optimal alignment matrix $\mathbf{S}^*_L$ at the coarsest level is achieved, the alignment at each level $\mathbf{S}^*_l$ can be computed by the interpolation, i.e., $\mathbf{S}^*_l = \mathbf{Q}_l^T\mathbf{S}^*_{l+1}\mathbf{P}_l,\ l = 1,\cdots,L-1$.

As shown in the next subsection, the number of nonzero elements of $\mathbf{S}^*_L$ is $O(L^2m_H + nd_L)$, making the computations of interpolations possibly intense. To address this issue, we only preserve the top-K elements of each row/column based on their absolute values in $\mathbf{S}^*_L$. The entire algorithm is summarized in Algorithm 1. We use two criteria to terminate the iteration (line 7-9): (1) the change of $\tilde{\mathbf{S}}_{L_1}, \tilde{\mathbf{S}}_{L_2}, \tilde{\mathbf{S}}_{L_3}$ of two successive iterations is less than a threshold ξ, or (2) the maximum number of iterations $t_{\max}$ is reached.

Recall that the coarse-scale structures of the networks are captured by the core matrices (e.g., $\mathbf{A}_2(\mathcal{S}_{A_2}, \mathcal{S}_{A_2})$ and $\mathbf{B}_2(\mathcal{S}_{B_2}, \mathcal{S}_{B_2})$), the corresponding submatrix of the alignment matrix indicates how network clusters are aligned. For example, $\mathbf{S}^*_2(\mathcal{S}_{B_2}, \mathcal{S}_{A_2})$ indicates how clusters of network $\mathcal{G}_1$ and $\mathcal{G}_2$ are aligned at the second level.

Algorithm 1 Multilevel Network Alignment (Moana).

Input: (1) the adjacency matrices $\bar{\mathbf{A}}_1, \bar{\mathbf{B}}_1$ of two undirected networks $\mathcal{G}_1$, $\mathcal{G}_2$, (2) the sparse prior alignment preference $\mathbf{H}_1$, (3) the number of levels L, (4) the parameters α, K.

Output: the alignment matrices $\mathbf{S}^*_l,\ l = 1,\cdots,L$ between $\mathcal{G}_1, \mathcal{G}_2$.

1: Compute $\mathbf{A}_1, \mathbf{B}_1$ by symmetrically normalizing $\bar{\mathbf{A}}_1, \bar{\mathbf{B}}_1$;

▷ **Network coarsening.**

3: $\mathbf{P}_1,\cdots,\mathbf{P}_{L-1}$ and $\tilde{\mathbf{A}}_L \leftarrow \mathrm{MMF}(\mathbf{A}_1)$;

4: $\mathbf{Q}_1,\cdots,\mathbf{Q}_{L-1}$ and $\tilde{\mathbf{B}}_L \leftarrow \mathrm{MMF}(\mathbf{B}_1)$;

5: Compute the coarsest level $\mathbf{H}_L = \mathbf{Q}_{L-1}\cdots\mathbf{Q}_1\mathbf{H}_1\mathbf{P}_1^T\cdots\mathbf{P}_{L-1}^T$;

▷ **Alignment at the coarsest level.**

7: **while** not converged **do**

8: Update $\tilde{\mathbf{S}}_{L_1}, \tilde{\mathbf{S}}_{L_2}, \tilde{\mathbf{S}}_{L_3}$ by Eq. (11)-(13);

9: Compute $\tilde{\mathbf{s}}^*_{L_4}$ by Eq. (15) and $\tilde{\mathbf{S}}^*_{L_4} = \mathrm{mat}(\tilde{\mathbf{s}}^*_{L_4})$;

10: Compose $\mathbf{S}^*_L$ by Eq. (16);

▷ **Alignment interpolation.**

12: Preserve top-K elements in each row/column of $\mathbf{S}^*_L$;

13: **for** $l = L-1 \to 1$ **do**

14: Compute $\mathbf{S}^*_l = \mathbf{Q}_l^T\mathbf{S}^*_{l+1}\mathbf{P}_l$;

3.4 Algorithm Analysis

We analyze the effectiveness and efficiency of our proposed algorithm. The *perfect interpolation* (Lemma 1) indicates that in theory, by using orthogonal interpolation matrices, the alignment matrix at the finer level can be obtained from that at the coarser level without any interpolation error. Nonetheless, in practice, we might still introduce some alignment error at the coarsest level due to the fact that many matrices are not *fully multiresolution factorizable* (i.e., $\tilde{\mathbf{A}}_L \neq \mathbf{A}_L$) [15]. Having this in mind, we seek to quantify the alignment error due to the fact that $\mathbf{A}_L \neq \tilde{\mathbf{A}}_L, \mathbf{B}_L \neq \tilde{\mathbf{B}}_L$.

Theorem 1. ***Alignment error bound.*** *Denote $\delta_1 = \|\mathbf{A}_L - \tilde{\mathbf{A}}_L\|_F$, $\delta_2 = \|\mathbf{B}_L - \tilde{\mathbf{B}}_L\|_F$. Suppose the ranks of $\mathbf{A}_1, \mathbf{B}_1$ are r_1, r_2 whose values are often small due to the low-rank characteristics of many real-world networks. Then, the error of the output alignment matrix at level $l, l = 1,\cdots,L$ is bounded as*

$$\frac{\|\mathbf{S}^*_l - \mathbf{S}_l\|_F}{\|\mathbf{S}_l\|_F} \leq \frac{2\epsilon\kappa}{1-\epsilon\kappa} \tag{17}$$

where $\mathbf{S}^*_l$ is the alignment matrix at level l obtained by Algorithm 1, $\mathbf{S}_l$ is the ideally interpolated alignment from $\mathbf{S}_L$ which is computed by $\mathbf{A}_L$. The parameter κ is the condition number under Frobenius norm of the matrix $(1-\alpha)(\mathbf{I}-\alpha\mathbf{A}_1\otimes\mathbf{B}_1)$, and $\epsilon = \sqrt{\frac{\alpha}{2n}}(\delta_1 r_2 + \delta_2 r_1 + \delta_1\delta_2)$.

Proof. See Appendix B. □

For the efficiency analysis of Moana, we start with the Lemma 2 on the sparsity of the matrix $\tilde{\mathbf{H}}_{L_4}$ which paves the way to the complexity analysis.

Lemma 2. ***Sparsity of $\tilde{\mathbf{H}}_{L_4}$.*** *Suppose there are m_H nonzero elements in the sparse prior similarity matrix $\mathbf{H}_1$ and $m_H \ll n^2$. At the coarsest level, the number of nonzero elements of $\tilde{\mathbf{H}}_{L_4}$ is $O(L^2m_H)$.*

Proof. Denote the density of the nonzero elements in $\mathbf{H}_1$ as $\rho_1 = \frac{m_H}{n^2}$. Starting with the coarsening at the first level, the computation

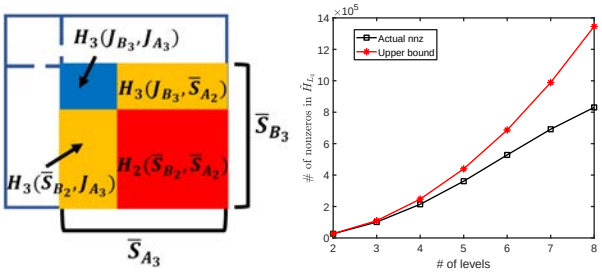


Figure 2: An illustration example of Lemma 2. (a) explains its proof where the indices have been reordered. (b) shows a case on Gr-Qc networks where $\rho_1 = 0.001, n = 5,241$.

of the matrix $\mathbf{H}_2$ is as follows.

$$\mathbf{H}_2(i,j) = (\mathbf{Q}_1\mathbf{H}_1\mathbf{P}_1^T)(i,j) = \sum_{p,q} \mathbf{Q}_1(i,q)\mathbf{H}_1(q,p)\mathbf{P}_1(j,p) \quad (18)$$

$$= \mathbf{Q}_1(i,i)\mathbf{H}_1(i,j)\mathbf{P}_1(j,j) + \mathbf{Q}_1(i,i)\mathbf{H}_1(i,p)\mathbf{P}_1(j,p) \quad (19)$$
$$+ \mathbf{Q}_1(i,q)\mathbf{H}_1(q,j)\mathbf{P}_1(j,j) + \mathbf{Q}_1(i,q)\mathbf{H}_1(q,p)\mathbf{P}_1(j,p)$$

The second equation is due to Eq. (25) where $k = 2$. That is, $\mathbf{Q}_1(i,q) \neq 0$ if $i_1^u = i$ and $i_2^u = q$ for some u, and similarly to $\mathbf{P}_1(j,p)$. Thus, $\mathbf{H}_2(i,j)$ can be nonzero if at least one of the $\mathbf{H}_1(i,j)$, $\mathbf{H}_1(i,p)$, $\mathbf{H}_1(q,j)$, $\mathbf{H}_1(q,p)$ are nonzero. In other words, each nonzero element in $\mathbf{H}_1$ can contribute to generating at most four nonzero elements in $\mathbf{H}_2$. Thus, the density of $\mathbf{H}_2$ is $\rho_2 \leq 4\rho_1$. Denote $\mathcal{S}_{A_l}, \mathcal{S}_{B_l}$ as the active indices of $\mathbf{A}_l, \mathbf{B}_l$ at the l-th level, and $\overline{\mathcal{S}}_{A_l}, \overline{\mathcal{S}}_{B_l}$ as the inactive indices. Since the second order MMF inactivates half of the current active indices [15], i.e., $|\overline{\mathcal{S}}_{A_2}| \approx |\overline{\mathcal{S}}_{B_2}| \approx \frac{n}{2}$, we have the number of nonzero elements $\text{nnz}(\mathbf{H}_2(\overline{\mathcal{S}}_{B_2}, \overline{\mathcal{S}}_{A_2})) \leq 4\rho_1(\frac{n}{2})^2 = \rho_1 n^2$. At the second level, thanks to the property of MMF that $\mathbf{P}_2(\overline{\mathcal{S}}_{A_2}, \overline{\mathcal{S}}_{A_2}) = \mathbf{I}$ and $\mathbf{Q}_2(\overline{\mathcal{S}}_{B_2}, \overline{\mathcal{S}}_{B_2}) = \mathbf{I}$, the computation of $\mathbf{H}_3 = \mathbf{Q}_2\mathbf{H}_2\mathbf{P}_2^T$ can be decomposed into

$$\mathbf{H}_3(\mathcal{S}_{B_2}, \mathcal{S}_{A_2}) = \mathbf{Q}_2(\mathcal{S}_{B_2}, \mathcal{S}_{B_2})\mathbf{H}_2(\mathcal{S}_{B_2}, \mathcal{S}_{A_2})\mathbf{P}_2^T(\mathcal{S}_{A_2}, \mathcal{S}_{A_2}) \quad (20)$$

$$\mathbf{H}_3(\mathcal{S}_{B_2}, \overline{\mathcal{S}}_{A_2}) = \mathbf{Q}_2(\mathcal{S}_{B_2}, \mathcal{S}_{B_2})\mathbf{H}_2(\mathcal{S}_{B_2}, \overline{\mathcal{S}}_{A_2}) \quad (21)$$

$$\mathbf{H}_3(\overline{\mathcal{S}}_{B_2}, \mathcal{S}_{A_2}) = \mathbf{H}_2(\overline{\mathcal{S}}_{B_2}, \mathcal{S}_{A_2})\mathbf{P}_2^T(\mathcal{S}_{A_2}, \mathcal{S}_{A_2}) \quad (22)$$

$$\mathbf{H}_3(\overline{\mathcal{S}}_{B_2}, \overline{\mathcal{S}}_{A_2}) = \mathbf{H}_2(\overline{\mathcal{S}}_{B_2}, \overline{\mathcal{S}}_{A_2}) \quad (23)$$

Similar to the analysis of Eq. (18), the density of the submatrix $\mathbf{H}_3(\mathcal{S}_{B_2}, \mathcal{S}_{A_2})$ is further increased by at most four times, and that of $\mathbf{H}_3(\mathcal{S}_{B_2}, \overline{\mathcal{S}}_{A_2}), \mathbf{H}_3(\overline{\mathcal{S}}_{B_2}, \mathcal{S}_{A_2})$ will be increased by at most twice. Note that the indices $\mathcal{J}_{A_3} = \mathcal{S}_{A_2} \setminus \mathcal{S}_{A_3}$ and $\mathcal{J}_{B_3} = \mathcal{S}_{B_2} \setminus \mathcal{S}_{B_3}$ are added into the inactive indices at the third level where $|\mathcal{J}_{A_3}| \approx |\mathcal{J}_{B_3}| \approx \frac{n}{4}$. And since the matrix $\mathbf{H}_3(\overline{\mathcal{S}}_{B_3}, \overline{\mathcal{S}}_{A_3})$ consists of four small blocks, including $\mathbf{H}_3(\mathcal{J}_{B_3}, \mathcal{J}_{A_3}), \mathbf{H}_3(\mathcal{J}_{B_3}, \overline{\mathcal{S}}_{A_2}), \mathbf{H}_3(\overline{\mathcal{S}}_{B_2}, \mathcal{J}_{A_3})$ and $\mathbf{H}_2(\overline{\mathcal{S}}_{B_2}, \overline{\mathcal{S}}_{A_2})$, the number of its nonzero elements is

$$\text{nnz}(\mathbf{H}_3(\overline{\mathcal{S}}_{B_3}, \overline{\mathcal{S}}_{A_3})) \leq 16\rho_1(\frac{n}{4})^2 + 2(\frac{n^2}{8})8\rho_1 + \rho_1 n^2 = 4\rho_1 n^2$$

By induction, at the L-th level, the total number of nonzero elements of $\mathbf{H}_L(\overline{\mathcal{S}}_{B_L}, \overline{\mathcal{S}}_{A_L})$ is

$$\text{nnz}(\mathbf{H}_L(\overline{\mathcal{S}}_{B_L}, \overline{\mathcal{S}}_{A_L})) \leq \sum_{l=2}^{L}(2l-3)\rho_1 n^2 \leq (L-1)^2 m_H \quad (24)$$

Thus, $\text{nnz}(\tilde{\mathbf{H}}_{L_4}) = \text{nnz}(\mathbf{H}_L(\overline{\mathcal{S}}_{B_L}, \overline{\mathcal{S}}_{A_L})) = O(L^2 m_H)$. □

We present an illustrative example of Lemma 2 in Figure 2, where Figure 2 (a) shows different parts of matrix $\mathbf{H}_L$ after some index reordering for the sake of illustration, and Figure 2 (b) gives a real case on the Gr-Qc networks where $\rho_1 = 0.001$, $n = 5,241$. We can observe that the actual number of nonzero elements in $\tilde{H}_{L_4}$ is much less than the analyzed upper bound, i.e., $(L-1)^2 m_H$, as L increases. Next, based on the Lemma 2, we give the complexity of the proposed Moana algorithm in Theorem 2, which states that the computational complexity of our proposed algorithm is *linear*.

Theorem 2. ***Complexity analysis.*** *The time complexity of Algorithm 1 is $O(mL + nd_L^2 t_{max} + L^2 m_H + LKn)$ and its space complexity is $O(L^2 m_H + L^2 Kn + nd_L)$. Here, m, n are the number of edges and nodes in the networks, $d_L = max(\lambda_L, \mu_L)$ is the size of core matrix. t_{max} is the number of iterations until convergence in the alignment phase and K is used for top-K preservation of $\mathbf{S}_L^*$. m_H is the number of nonzero elements in the matrix $\mathbf{H}_1$ and L is the number of levels.*

Proof. In Algorithm 1, the symmetric normalization of $\bar{\mathbf{A}}_1, \bar{\mathbf{B}}_1$ in Line 1 requires an $O(m)$ time complexity. In the network coarsening phase, the MMF on $\mathbf{A}_1, \mathbf{B}_1$ has an $O(mL)$ time and space complexity[1] where by a careful implementation, we use Graclus [6] for an extremely efficient clustering and avoid explicitly computing the Gram matrix in MMF. Note that we can compute $\mathbf{H}_l$ ($l = 2, \cdots, L$) in a same way as Eq. (20)-(23). Thus, by Lemma 2, the computation of the matrix $\mathbf{H}_L$ takes $O(Lm_H)$ as time complexity and $O(L^2 m_H + nd_L)$ as space complexity.

In the alignment phase, Line 8 has an $O(d_L^3)$ time complexity for Eq. (11), as well as $O(nd_L^2)$ for both Eq. (12) and Eq. (13). To compute Line 10, since there are $O(L^2 m_H)$ nonzero elements in $\tilde{h}_L$, the time complexity is $O(L^2 m_H)$. The space complexity to store the alignment matrix $\mathbf{S}_L^*$ is $O(L^2 m_H + nd_L)$ due to the nonzero elements.

In the interpolation phase, Line 13 costs $O(Kn)$ in both time and space. The analysis of Line 15 is analogous to that of Line 5 so Line 15 has an $O(LKn)$ time complexity and an $O(L^2 Kn)$ space complexity. Thus, the total time complexity is $O(m + nd_L^2 t_{\max} + L^2 m_H + LKn)$ and the space complexity is $O(L^2 m_H + L^2 Kn + nd_L)$. □

4 EXPERIMENTS

In this section, we present the experimental results of the proposed algorithm Moana. We evaluate it in the following aspects:

- *Effectiveness*: How accurate is our algorithm to align networks and how robust is our algorithm to the parameters?
- *Efficiency*: How fast and scalable is our algorithm?

4.1 Experimental Setup

Datasets. We evaluate the proposed algorithm on six real-world networks. The statistics of all datasets are summarized as follows.

- *Zachary's Karate Club:* This dataset contains 34 nodes and 78 edges. Each node is a member of the karate club and each edge represents the friendship between two members [33].
- *Gr-Qc network:* This collaboration network contains 5,241 nodes and 11,923 edges. Each node represents an author, and

[1] In [29], the complexity for a sparse matrix is given in the form of $O(\gamma c L n^2)$ where c is the cluster size, γ is the fraction of nonzero elements and hence $O(\gamma n^2) = O(m)$.

there exists an edge if two authors have coauthored together [17].

- *Google+ network:* This dataset contains 23,628 nodes and 39,194 edges. Nodes are the users of Google+ and an edge denotes that one user has the other user in his/her circles [18].
- *Amazon product co-purchasing network:* This network was collected from the Amazon website. Different nodes represent different products. If a product is frequently co-purchased with another product, there exists an edge between them. In total, there are 334,863 nodes and 925,872 edges in the network [32].
- *ACM coauthor network:* The ACM dataset is collected up to 2016 and it contains 2,381,688 papers. Each paper has a list of authors as well as the venue of the paper [28]. We construct a coauthorship network from the dataset where each node represents an author and each edge represents the coauthorship.
- *DBLP coauthor network:* The DBLP dataset is collected up to 2016 and it contains 3,272,991 papers [28]. Same as the ACM dataset, we construct an coauthorship network from the dataset.

Based on the above datasets, we construct the following five alignment scenarios for evaluations.

- *S1. Zachary vs. Zachary networks.* Given the adjacency matrix $\bar{\mathbf{A}}_1$ of the original network, we generate a random permutation matrix $\mathbf{P}$ and treat the permuted matrix $\bar{\mathbf{B}}_1 = \mathbf{P}\bar{\mathbf{A}}_1\mathbf{P}^T$ as the adjacency matrix of the second network. The permutation matrix $\mathbf{P}$ is used as the ground-truth alignment. We construct the prior similarity matrix $\mathbf{H}_1$ based on the node degree similarity.
- *S2. Gr-Qc vs. Gr-Qc networks.* The scenario is built same as *S1*.
- *S3. Google+ vs. Google+ networks.* This is built same as *S1*.
- *S4. Amazon co-purchasing vs. Amazon co-purchasing networks.* We randomly extract two subgraphs to be aligned from the entire network. The first subgraph has 74,596 nodes and 196,534 edges while the second subgraph has 66,951 nodes and 174,467 edges. There are 62,406 common nodes which are used as the ground-truth. We use node degree similarity as $\mathbf{H}_1$.
- *S5. ACM vs. DBLP coauthor networks.* We extract a subgraph from each coauthorship network. Specifically, the extracted ACM subgraph has 9,872 nodes and 39,561 edges, while the DBLP subgraph has 9,916 nodes and 44,808 edges. There are 6,325 common nodes as the ground-truth. We use the number of papers in different venues published by each author as the node attributes. We calculate $\mathbf{H}_1$ by the cosine similarity of the node attributes.

Implementation Details. As analyzed in Theorem 1, the zero-out operation in MMF implementation could introduce a certain alignment error at the coarsest level. To mitigate this issue, we run one iteration of the fixed-point update as the post-processing step: $\mathbf{S}_l^* = \alpha\mathbf{B}_l\mathbf{S}_l^*\mathbf{A}_l + (1-\alpha)\mathbf{H}_l$. This computation can be done in near-linear time with a locality sensitive hashing based implementation, thanks to the sparsity of the matrices. To evaluate the effectiveness of our proposed algorithm, we use the greedy match algorithm [27] as a post-processing to obtain the one-to-one mapping from the alignment matrix, followed by calculating the percentage of the ground-truth that can be correctly aligned as the accuracy.

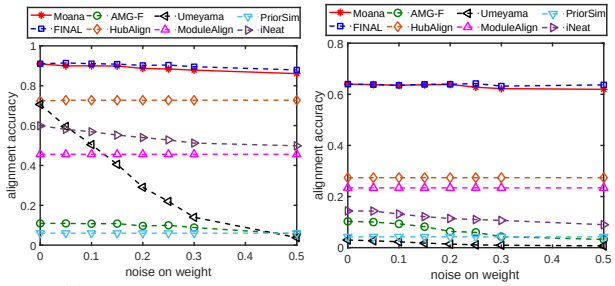
Comparison Methods. We compare our algorithm Moana with the following existing network alignment algorithms, including (1) *FINAL* that uses the fixed-point algorithm on Eq. (4) [35], (2) *AMG-F* that first uses *AMG* [23] to coarsen the input networks at L levels and then aligns based on the coarsest networks, followed by interpolating the alignment matrix with the interpolation matrices generated by *AMG*, (3) *iNeat* [36], (4) *HubAlign* [11], (5) *Umeyama* [30], (6) *ModuleAlign* [10] that leverages the hierarchical clustering on the networks and (7) *PriorSim* which aligns based on the prior similarity matrix $\mathbf{H}_1$. Note that *iNeat* was originally designed to learn the nonnegative matrix factorization of the incomplete adjacency matrices. In our experiments, we use the low-rank eigen-decomposition of the input adjacency matrices since there are no missing entries in our setting.

Machines and Repeatability. All experiments are performed with four 3.6GHz Intel Cores and 256G RAM. Note that a large size of memory is only needed for running some comparison methods on large datasets (e.g., *Umayama* and *ModuleAlign* on *S4*). Our proposed algorithm is programmed in MATLAB. We will release the source code and the datasets after the paper is published.

4.2 Effectiveness Results

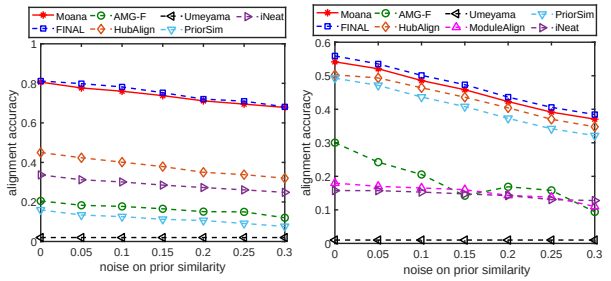
We first evaluate how the uniform noise on the edge weights influences the node-level alignment accuracy in scenarios *S2* and *S3*. The results are summarized in Figure 3. We have the following observations. First, our proposed algorithm Moana is very close to its single node-level alignment counterpart *FINAL*. Note that our algorithm can be viewed as an approximation of *FINAL* in terms of the node-level alignment. Specifically, our algorithm is only about 1.5% lower than *FINAL* in the worst case. In the meanwhile, the proposed Moana outperforms all the remaining methods that we compared against with respect to the node-level alignment accuracy. Second, our proposed algorithm, as well as *FINAL* are quite robust to the noise. This is due to the prior similarity matrix that works as the regularization and can mitigate the effects of noise on the networks. For *HubAlign* and *ModuleAlign*, since they are originally designed solely for unweighted networks, we replicate their results without any noise and therefore their curves are flat. On the other side, *Umeyama* is more sensitive to the noise. Third, our multilevel alignment algorithm achieves an accuracy improvement by up to 30%, compared with *HubAlign*, *ModuleAlign* and *iNeat* that are designed to leverage some special structural characteristics (e.g., hub nodes, hierarchical clustering, low rank) of the underlying networks. This improvement demonstrates the potential advantages in finding node-level alignment of our multilevel alignment method by exploring the richer patterns of networks.

Furthermore, we evaluate the impact on the alignment accuracy by the noise in the input prior similarity matrix $\mathbf{H}_1$ in the scenarios *S4* and *S5*. As one can see in the Figure 4, although most of the comparison methods have an accuracy drop due to the noise, our proposed algorithm still remains very close to *FINAL*. The results of *ModuleAlign* in the Amazon Product co-purchasing networks are not included because the algorithm cannot finish in a reasonable



(a) Cr-Qc networks. (b) Google+ networks.

Figure 3: (Higher is better.) Alignment accuracy vs. the edge weight noise. (a) L=5, K=500, α=0.5. (b) L=7, K=1500, α=0.5.



(a) Amazon co-purchasing networks (L=8, K=2000, α=0.5). (b) ACM vs. DBLP coauthor networks (L=5, K=1000, α=0.5).

Figure 4: (Higher is better.) Alignment accuracy vs. the noise in the prior similarity matrix.

time on large networks (i.e., about 70k nodes) given its high time complexity $O(n^3 \log n)$ [10].

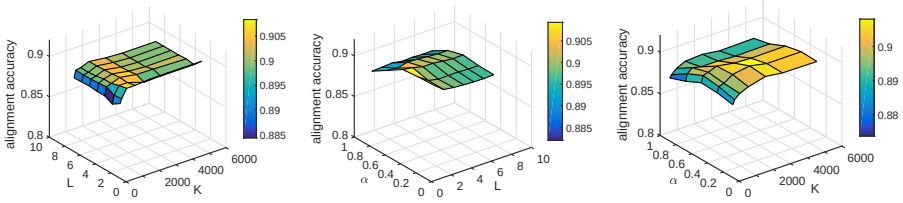
Parametric study on node-level alignment. There are three parameters K, L, α in the proposed algorithm. To be specific, α is the regularization parameter that controls the importance of the topology consistency, K is used to preserve top-K elements in $\mathbf{S}_L^*$, and L is the number of levels. We fix one of these parameters and study the impact of the other two parameters on the node-level alignment accuracy. The results are summarized in Figure 5. We have two main observations. First, the alignment accuracy is stable over a wide range of α, K and L. In particular, the performance first increases with respect to K, and it quickly becomes flat for $K \geq 500$. Notice that a small K can also accelerate the algorithm, and hence $K = 500$ is recommended in practice. On the other hand, recall that increasing L reduces the size of core matrices (i.e., λ_L, μ_L) and hence improves the efficiency of the alignment at the coarsest level, at a small expense in the alignment accuracy and the running time of MMF itself. We find that an L that leads to the size of core matrices (i.e., λ_L, μ_L) in the order of hundreds achieves a good balance between effectiveness and efficiency. Second, compared with the other two parameters (K, L), the regularization parameter α has a relatively larger impact on the alignment accuracy. This is because α balances the importance of the topology consistency and the prior alignment information in Eq. (1), and it directly influences the exact solution of its single node-level alignment.

The multilevel nature of Moana brings the potential to align clusters across different networks. To verify this, we further conduct an experiment to evaluate the cluster alignment accuracy on Gr-Qc networks. In particular, we view the nodes that are integrated into a single 'supernode' up to the l-th level (e.g., based on $\mathbf{P}_1, \cdots, \mathbf{P}_{l-1}$) as a cluster at level l. If a cluster at level l in network $\mathcal{G}_1$ share the most overlapped nodes with some cluster in $\mathcal{G}_2$, we treat the alignment between these two clusters as a ground-truth. With the output $\mathbf{S}_l^*(\mathcal{S}_{B_l}, \mathcal{S}_{A_l})$ matrices by Algorithm 1, we define the top-15% cluster alignment accuracy as follows. Given a ground-truth alignment between cluster i in $\mathcal{G}_1$ and cluster j in $\mathcal{G}_2$, if $\mathbf{S}_l^*(j, i)$ is among the highest top-15% entries within the j-th row and i-th column, we say there is a *hit*, i.e., the two clusters are correctly aligned. As Figure 6 shows, our algorithm can achieve a good cluster alignment accuracy at different levels.

A case study on the multilevel alignment. In addition to the quantitative comparisons, we also conduct a case study on the Zachary's Karate Club networks (*S1*) to further demonstrate our algorithm's capability to find the alignment among clusters at different granularities. We only show sample results in Figure 7 for clarity. Among others, Figure 7 (a) visualizes the input networks. Figure 7 (b) and (c) show the network structures at the two coarsest levels, respectively. To be specific, the nodes with a larger marker size represent the active indices of the coarsened matrices. Each of them represents a certain pattern (e.g., cluster) that is differentiated by the shape/color of the nodes. For example, in Figure 7 (b), the blue circle node with a larger size is one of the active nodes whereas the small blue nodes are inactivated during the MMF coarsening. As one can see, all these blue circle nodes form a cluster represented by the corresponding active node, and hence each network has three clusters in total in Figure 7 (b). The weighted interactions among the clusters are computed as the submatrix indexed by the active indices, and are represented by the thick solid lines. At the coarsest level (Figure 7 (c)), the clusters and their interactions are visualized similarly. In Figure 7 (d), the orange dashed lines show the alignment results among the active nodes at the coarsest level. In fact, since the active nodes represent different clusters, the correct alignment among the active nodes indicates that the clusters at the coarsest level are aligned correctly. Once the alignment at the coarsest level is done, we interpolate the results to the next finer level shown in Figure 7 (e). Likewise, we observe that the alignment after the first interpolation can unveil the alignment among the clusters at the finer level. Figure 7 (f) only shows parts of the interpolated node-level alignment for the sake of clarity. Overall, our algorithm can correctly align 32 out of 34 node pairs. This case study demonstrates that our algorithm can not only find the correct node-level alignment, but also unveil the meaningful alignment of the clusters in the networks at different granularities.

4.3 Efficiency Results

We first evaluate the efficiency of our algorithm (Moana) in terms of the balance between the running time and the node-level alignment accuracy. The results are summarized in Figure 8. As one can see, our algorithm obtains an up to 10× speedup compared with its single-level counterpart *FINAL* with a little loss in terms of the node-level alignment accuracy. For other alignment algorithms that explore the special characteristics of the networks (*HubAlign*, *ModuleAlign* and *iNeat*), our algorithm consistently outperforms them in terms of both efficiency and effectiveness, with the only exception of *iNeat* in Figure 8 (b) which runs faster at the cost of a much lower accuracy. Overall, we conclude from these results that



(a) Impact of K and L (fixing $\alpha = 0.5$). (b) Impact of α and L (fixing $K = 500$). (c) Impact of α and K (fixing $L = 5$).

Figure 5: Parametric studies on Cr-Qc networks.

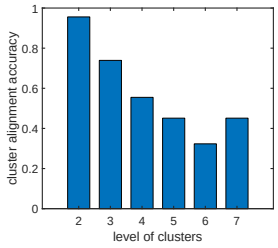


Figure 6: Cluster alignment accuracy at multiple levels.

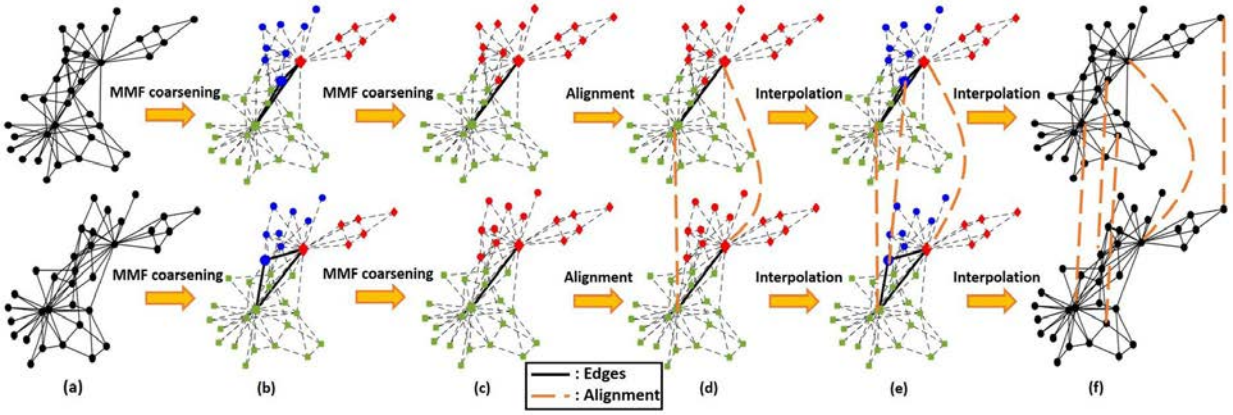
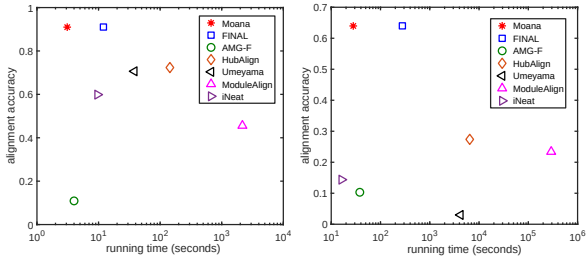
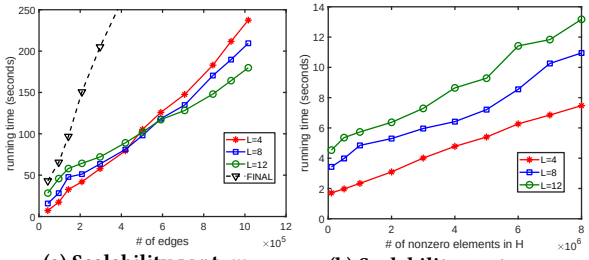


Figure 7: A case study on the Zachary's Karate Club networks (Best viewed in color).



(a) Gr-Qc networks. (b) Google+ networks.

Figure 8: Balance between the accuracy and running time.



(a) Scalability w.r.t. m. (b) Scalability w.r.t. m_H.

Figure 9: Scalability of Moana.

the proposed Moana algorithm achieves a good balance between the running time and the alignment accuracy.

Scalability. We evaluate the scalability of our algorithm (Moana) based on the Amazon product co-purchasing networks w.r.t. (1) the number of edges while fixing $m_H = 3 \times 10^5$, and (2) m_H while fixing the size of networks $n = 5,000$. The results are summarized in Figure 9. We observe that the algorithm scales *linearly* w.r.t. both the number of edges in the networks and the number of nonzero elements in the prior similarity matrix $\mathbf{H}_1$, which is consistent with Theorem 2. *FINAL* which can be viewed as a special case when $L = 1$ (i.e., without coarsening), on the other hand, scales quadratically and is slower than the proposed algorithm, especially when the networks are large. We also observe from Figure 9 (a) that large L is beneficial in terms of efficiency as the size of networks becomes large. This is because coarsening the networks at more levels can further reduce the size of the core diagonal matrices.

5 RELATED WORK

Network Alignment. Multiple networks naturally appear in many real-world applications, including network alignment [35], subgraph matching [8], multi-network ranking [12] and so on. Among others, network alignment has attracted extensive research interests. Network alignment can be generally categorized into local network alignment and global network alignment. Local network alignment aims to align the small regions (e.g., motifs, small subgraphs, etc.) across multiple networks. Some recent works in this category include [3, 22]. However, the local alignment methods might be too restrictive to find the one-to-one node mappings and the alignment among more complicated large patterns of the networks. On the other side, many existing global network alignment algorithms explicitly or implicitly assume the *topology consistency*. That is, if two nodes are aligned together, their corresponding neighbors are likely to be aligned. For example, an early work *IsoRank* conducts a random walk in the Kronecker product graph and propagates the node pair similarities to the neighboring node pairs so that the similarities are smoothed [27]. *NetAlign* formulates the network alignment problem as an optimization problem and maximizes the number of neighboring node pairs that are aligned [1]. *BigAlign*

(as well as its variant *UniAlign*) [16] and *UMA* [34] assumes one network is a noisy permutation of the other network. Moreover, *IONE* attempts to leverage the network embedding for alignment [19] in a semi-supervised manner, under the assumption that the aligned nodes are close to each other in the embedding space. To further incorporate the attribute information, *COSNET* formulates the local consistency among the node attributes and the global topological consistency into a joint optimization problem [37]. *FINAL* [35] considers both node and edge attributes to calibrate the topology-based alignment. [7] further improves the efficiency by accelerating solving the Sylvester equation.

More recently, several network alignment methods attempt to exploit some special characteristics of real-world networks. For example, *HubAlign* aligns the hub nodes first and then all the remaining nodes [11]. In a different thread, *ModuleAlign* [10] and *CAlign* [2] leverage the cluster/community structures in the networks. However, none of these methods allows to find the alignment at more than two levels. Thanks to the low rank characteristics of many real networks, a recent work *iNeat* exploits the low rank structure of the alignment matrix, leading to a provable linear algorithm [36]. The alignment matrices in *iNeat* [36] and another early work *Umeyama* [30] are represented by the multiplication of three matrices, which seems to resemble the interpolation formula in our method. However, neither *iNeat* [36] or *Umeyama* [30] is able to find the alignment at the coarse levels, since they are both essentially single-level alignment methods.

Multilevel Approaches. The multilevel approaches have shown a strong performance in many graph mining problems. For example, *METIS* [13] and *Graclus* [6] are two well-known multilevel approaches to solve the graph partitioning problem. These two methods use a heavy-edge matching based graph coarsening approach, making themselves extremely efficient. On the other side, the algebraic multigrid methods which are originally designed for solving the linear systems, have been widely used to many graph related problems, such as [21, 23]. Most of the multilevel approaches follow the *coarsen-solve-interpolate* strategy. Indeed, our proposed network alignment algorithm follows this generic strategy as well.

Another closely related concept is the multiresolution analysis on graph that has played an important role in many research areas. For example, it has been widely used to design the pooling operations in the graph convolutional neural networks [5], an emerging area in machine learning and data mining. Besides, it has been applied to design the multiscale graph kernel as well [14]. In graph signal processing, the multiresolution analysis is extensively studied by many different methods, such as the graph diffusion wavelet [4], multiscale pyramid transform [26]. Moreover, the multiresolution matrix factorization, as an alternate to low rank characteristics [40], is able to factorize the input matrix at different scales [15].

6 CONCLUSION

Large multi-sourced networks from many application domains have greatly galvanized network alignment research. Most of the existing single-level alignment methods might overlook the rich patterns (e.g., *hierarchical cluster-within-clusters* underlying the networks) and/or bear a super-linear computational complexity. In this paper, we study the multilevel network alignment problem. We first drive a theoretical condition on perfect interpolation (Lemma 1). Based on that, we propose a carefully designed *coarsening-alignment-interpolation* multilevel algorithm Moana, which has a *linear* complexity in both time and space. We also drive an error bound of the alignment due to the coarsening step. We perform extensive experiments that demonstrate the efficacy of our algorithm on the node-level alignment and its capability of finding the alignment of the rich patterns (e.g., clusters) across the input networks.

7 ACKNOWLEDGEMENT

This material is supported by the National Science Foundation under Grant No. IIS-1651203, IIS-1715385, IIS-1741197 and CNS-1639227, by DTRA under the grant number HDTRA1-16-0017, by the United States Air Force and DARPA under contract number FA8750-17-C-0153 [2], by Department of Homeland Security under Grant Award Number 2017-ST-061-QA0001, and by Army Research Office under the contract number W911NF-16-1-0168. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

A COMPOUND ROTATION MATRIX

The compound rotation matrix is formally defined as follows [15].

Definition 2. ***Compound rotation matrix.*** *The compound rotation matrix* $\mathbf{P}$ *of order* k *is a block diagonal orthogonal matrix as*

$$\mathbf{P} = \oplus_{(i_1^1,\cdots,i_{k_1}^1)}\mathbf{O}^1 \oplus_{(i_1^2,\cdots,i_{k_2}^2)} \mathbf{O}^2 \oplus \cdots \oplus_{(i_1^p,\cdots,i_{k_p}^p)} \mathbf{O}^p. \quad (25)$$

where $\{i_1^1,\cdots,i_{k_1}^1\},\cdots,\{i_1^p,\cdots,i_{k_p}^p\}$ are the subsets of $\{1,\cdots,n\}$ with $k_1,\cdots,k_p \le k$ and the matrices $\mathbf{O}^1,\cdots,\mathbf{O}^p$ are some orthogonal *Givens* rotation matrices. The operator $\oplus$ is defined to form the block diagonal matrix, i.e.,

$$\mathbf{P}(a,b) = \begin{cases} \mathbf{O}^u(q,r) & \text{if } i_q^u = a \text{ and } i_r^u = b \text{ for some } u,q,r \\ 0 & \text{otherwise} \end{cases}$$

By sequentially applying the compound rotation matrices, we can obtain a series of coarsened matrices as below

$$\mathbf{A}_1 \overset{\mathbf{P}_1}{\to} \mathbf{A}_2 \overset{\mathbf{P}_2}{\to} \cdots \overset{\mathbf{P}_{L-1}}{\to} \mathbf{A}_L \to \tilde{\mathbf{A}}_L$$

where $\tilde{\mathbf{A}}_L$ is formed by zeroing out all the nonzero elements that are neither on the diagonal of $\mathbf{A}_L$ nor inside the $\mathcal{S}_L$-core matrix.

B PROOF OF THEOREM 1

Denote $\Delta\mathbf{A} = \tilde{\mathbf{A}}_L - \mathbf{A}_L$ and $\Delta\mathbf{B} = \tilde{\mathbf{B}}_L - \mathbf{B}_L$. Consider two linear systems $\mathbf{W}_L\mathbf{s}_L = \mathbf{h}_L$ and $\tilde{\mathbf{W}}_L\mathbf{s}_L^* = \mathbf{h}_L$ where $\mathbf{W}_L = (1-\alpha)(\mathbf{I}-\alpha\mathbf{A}_L\otimes\mathbf{B}_L)$, $\tilde{\mathbf{W}}_L = (1-\alpha)(\mathbf{I}-\alpha\tilde{\mathbf{A}}_L\otimes\tilde{\mathbf{B}}_L)$ and $\mathbf{s}_L = \mathrm{vec}(\mathbf{S}_L)$, $\mathbf{s}_L^* = \mathrm{vec}(\mathbf{S}_L^*)$. Clearly, the solutions to these linear systems are the closed-form solutions of alignment matrices, e.g., $\mathbf{s}_L^*$ corresponding to $\mathbf{S}_L^*$ in line 11 of Algorithm 1. By denoting $\Delta\mathbf{W}_L = \tilde{\mathbf{W}}_L - \mathbf{W}_L$, we have

$$\begin{aligned}
\|\Delta\mathbf{W}_L\|_F &= \alpha(1-\alpha)\|\mathbf{A}_L\otimes\mathbf{B}_L - \tilde{\mathbf{A}}_L\otimes\tilde{\mathbf{B}}_L\|_F \\
&= \alpha(1-\alpha)\|\Delta\mathbf{A}\otimes\mathbf{B}_L + \mathbf{A}_L\otimes\Delta\mathbf{B} + \Delta\mathbf{A}\otimes\Delta\mathbf{B}\|_F \\
&\le \alpha(1-\alpha)(\|\Delta\mathbf{A}\|_F\|\mathbf{B}_L\|_F + \|\mathbf{A}_L\|_F\|\Delta\mathbf{B}\|_F + \|\Delta\mathbf{A}\|_F\|\Delta\mathbf{B}\|_F) \\
&\le \alpha(1-\alpha)(\delta_1 r_2\|\mathbf{B}_L\|_2 + \delta_2 r_1\|\mathbf{A}_L\|_2 + \delta_1\delta_2) \\
&= \alpha(1-\alpha)(\delta_1 r_2 + \delta_2 r_1 + \delta_1\delta_2)
\end{aligned}$$

[2] Distribution Statement "A" (Approved for Public Release, Distribution Unlimited)

where the fourth line is due to $\|\mathbf{A}\|_F \leq \text{rank}(\mathbf{A})\|\mathbf{A}\|_2$ [24] for any matrix and $\text{rank}(\mathbf{A}_L) = \text{rank}(\mathbf{A}_1)$. The last line is due to the fact that $\mathbf{A}_L, \mathbf{B}_L$ have the same eigenvalues of $\mathbf{A}_1, \mathbf{B}_1$ respectively and hence the largest eigenvalues are equal to 1. Besides, we have

$$\begin{aligned}\|\mathbf{W}_L\|_F &= (1-\alpha)\sqrt{n_1 n_2 + \alpha^2 \|\mathbf{A}_L \otimes \mathbf{B}_L\|_F^2} \\ &\geq (1-\alpha)\sqrt{n^2+\alpha^2} \geq (1-\alpha)\sqrt{2\alpha n}\end{aligned}$$

where $n = \min(n_1, n_2)$. Thus, by picking $\epsilon = \sqrt{\alpha}\frac{\delta_1 r_2 + \delta_2 r_1 + \delta_1\delta_2}{\sqrt{2n}}$, we can guarantee that $\|\Delta\mathbf{W}\|_F \leq \epsilon\|\mathbf{W}_L\|_F$.

According to the well-known sensitivity analysis of linear systems [9], we have

$$\frac{\|\mathbf{S}_L - \mathbf{S}_L^*\|_F}{\|\mathbf{S}_L\|_F} = \frac{\|\mathbf{s}_L - \mathbf{s}_L^*\|_2}{\|\mathbf{s}_L\|_2} \leq \frac{2\epsilon\kappa}{1-\epsilon\kappa} \tag{26}$$

where κ is the condition number under Frobenius norm of matrix $\mathbf{W}_L$, i.e., $\kappa = \|\mathbf{W}_L\|_F\|\mathbf{W}_L^{-1}\|_F$. By interpolating $\mathbf{S}_L$, at level l

$$\begin{aligned}\frac{\|\mathbf{S}_l^* - \mathbf{S}_l\|_F}{\|\mathbf{S}_l\|_F} &= \frac{\|\mathbf{Q}_l^T \cdots \mathbf{Q}_{L-1}^T(\mathbf{S}_l^* - \mathbf{S}_l)\mathbf{P}_{L-1}\cdots\mathbf{P}_l\|_F}{\|\mathbf{Q}_l^T \cdots \mathbf{Q}_{L-1}^T\mathbf{S}_l\mathbf{P}_{L-1}\cdots\mathbf{P}_l\|_F} \\ &= \frac{\|\mathbf{S}_L - \mathbf{S}_L^*\|_F}{\|\mathbf{S}_L\|_F} \leq \frac{2\epsilon\kappa}{1-\epsilon\kappa}\end{aligned} \tag{27}$$

REFERENCES

[1] Mohsen Bayati, Margot Gerritsen, David F Gleich, Amin Saberi, and Ying Wang. 2009. Algorithms for large, sparse network alignment problems. In *2009 Ninth IEEE International Conference on Data Mining*. IEEE, 705–710.

[2] Zheng Chen, Xinli Yu, Bo Song, Jianliang Gao, Xiaohua Hu, and Wei-Shih Yang. 2017. Community-based network alignment for large attributed network. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 587–596.

[3] Giovanni Ciriello, Marco Mina, Pietro H Guzzi, Mario Cannataro, and Concettina Guerra. 2012. AlignNemo: a local network alignment method to integrate homology and topology. *PloS one* 7, 6 (2012), e38107.

[4] Ronald R Coifman and Mauro Maggioni. 2006. Diffusion wavelets. *Applied and Computational Harmonic Analysis* 21, 1 (2006), 53–94.

[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*. 3844–3852.

[6] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. 2007. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE transactions on pattern analysis and machine intelligence* 29, 11 (2007), 1944–1957.

[7] Boxin Du and Hanghang Tong. 2018. FASTEN: Fast Sylvester Equation Solver for Graph Mining. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1339–1347.

[8] Boxin Du, Si Zhang, Nan Cao, and Hanghang Tong. 2017. First: Fast interactive attributed subgraph matching. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1447–1456.

[9] Gene H Golub and Charles F Van Loan. 2012. *Matrix computations*. Vol. 3. JHU press.

[10] Somaye Hashemifar, Jianzhu Ma, Hammad Naveed, Stefan Canzar, and Jinbo Xu. 2016. ModuleAlign: module-based global alignment of protein–protein interaction networks. *Bioinformatics* 32, 17 (2016), i658–i664.

[11] Somaye Hashemifar and Jinbo Xu. 2014. HubAlign: an accurate and efficient method for global alignment of protein–protein interaction networks. *Bioinformatics* 30, 17 (2014), i438–i444.

[12] Jian Kang, Scott Freitas, Haichao Yu, Yinglong Xia, Nan Cao, and Hanghang Tong. 2018. X-rank: Explainable ranking in complex multi-layered networks. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1959–1962.

[13] George Karypis and Vipin Kumar. 1998. Multilevelk-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed computing* 48, 1 (1998), 96–129.

[14] Risi Kondor and Horace Pan. 2016. The multiscale laplacian graph kernel. In *Advances in Neural Information Processing Systems*. 2990–2998.

[15] Risi Kondor, Nedelina Teneva, and Vikas Garg. 2014. Multiresolution matrix factorization. In *International Conference on Machine Learning*. 1620–1628.

[16] Danai Koutra, Hanghang Tong, and David Lubensky. 2013. Big-align: Fast bipartite graph alignment. In *2013 IEEE 13th International Conference on Data Mining*. IEEE, 389–398.

[17] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1, 1 (2007), 2.

[18] Jure Leskovec and Julian J Mcauley. 2012. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*. 539–547.

[19] Li Liu, William K Cheung, Xin Li, and Lejian Liao. 2016. Aligning Users across Social Networks Using Network Embedding. In *IJCAI*. 1774–1780.

[20] Rui Liu, Wei Cheng, Hanghang Tong, Wei Wang, and Xiang Zhang. 2015. Robust multi-network clustering via joint cross-domain cluster alignment. In *2015 IEEE International Conference on Data Mining*. IEEE, 291–300.

[21] Oren E Livne and Achi Brandt. 2012. Lean algebraic multigrid (LAMG): Fast graph Laplacian linear solver. *SIAM Journal on Scientific Computing* 34, 4 (2012), B499–B522.

[22] Hazel N Manners, Ahed Elmsallati, Pietro H Guzzi, Swarup Roy, and Jugal K Kalita. 2017. Performing local network alignment by ensembling global aligners. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 1316–1323.

[23] Yvan Notay. 2010. An aggregation-based algebraic multigrid method. *Electronic transactions on numerical analysis* 37, 6 (2010), 123–146.

[24] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. 2008. The matrix cookbook. *Technical University of Denmark* 7, 15 (2008), 510.

[25] Erzsébet Ravasz and Albert-László Barabási. 2003. Hierarchical organization in complex networks. *Physical review E* 67, 2 (2003), 026112.

[26] David I Shuman, Mohammad Javad Faraji, and Pierre Vandergheynst. 2016. A multiscale pyramid transform for graph signals. *IEEE Transactions on Signal Processing* 64, 8 (2016), 2119–2134.

[27] Rohit Singh, Jinbo Xu, and Bonnie Berger. 2008. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences* 105, 35 (2008), 12763–12768.

[28] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 990–998.

[29] Nedelina Teneva, Pramod Kaushik Mudrakarta, and Risi Kondor. 2016. Multiresolution matrix compression. In *Artificial Intelligence and Statistics*. 1441–1449.

[30] Shinji Umeyama. 1988. An eigendecomposition approach to weighted graph matching problems. *IEEE transactions on pattern analysis and machine intelligence* 10, 5 (1988), 695–703.

[31] Jiejun Xu, Hanghang Tong, Tsai-Ching Lu, Jingrui He, and Nadya Bliss. 2018. GTA 3 2018: Workshop on Graph Techniques for Adversarial Activity Analytics. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 803–803.

[32] Jaewon Yang and Jure Leskovec. 2015. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* 42, 1 (2015), 181–213.

[33] Wayne W Zachary. 1977. An information flow model for conflict and fission in small groups. *Journal of anthropological research* 33, 4 (1977), 452–473.

[34] Jiawei Zhang and S Yu Philip. 2015. Multiple anonymized social networks alignment. In *2015 IEEE International Conference on Data Mining*. IEEE, 599–608.

[35] Si Zhang and Hanghang Tong. 2016. Final: Fast attributed network alignment. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1345–1354.

[36] Si Zhang, Hanghang Tong, Jie Tang, Jiejun Xu, and Wei Fan. 2017. ineat: Incomplete network alignment. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1189–1194.

[37] Yutao Zhang, Jie Tang, Zhilin Yang, Jian Pei, and Philip S Yu. 2015. Cosnet: Connecting heterogeneous social networks with local and global consistency. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1485–1494.

[38] Dawei Zhou, Jingrui He, Hongxia Yang, and Wei Fan. 2018. Sparc: Self-paced network representation for few-shot rare category characterization. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2807–2816.

[39] Dawei Zhou, Si Zhang, Mehmet Yigit Yildirim, Scott Alcorn, Hanghang Tong, Hasan Davulcu, and Jingrui He. 2017. A local algorithm for structure-preserving graph cut. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 655–664.

[40] Yao Zhou and Jingrui He. 2016. Crowdsourcing via Tensor Augmentation and Completion.. In *IJCAI*. 2435–2441.

Automatic Generation of Pattern-controlled Product Description in E-commerce

Tao Zhang
Alibaba Group
Hangzhou, China
guyan.zt@aliibaba-inc.com

Jin Zhang
Alibaba Group
Hangzhou, China
zj146372@aliibaba-inc.com

Chengfu Huo
Alibaba Group
Hangzhou, China
chengfu.huocf@aliibaba-inc.com

Weijun Ren
Alibaba Group
Hangzhou, China
afei@aliibaba-inc.com

ABSTRACT

Nowadays, online shoppers have paid more and more attention to detailed product descriptions, since a well-written description is a huge factor in making online sales. However, for a website with billions of product data like Alibaba, the writing efficiency of human copywriters cannot match the growth rate of new products. To address this issue, we propose a novel pointer-generator neural network to generate product description. In particular, coordinate encoders and a pattern-controlled decoder are utilized to improve generation quality with an attention mechanism. The coordinate encoders equipped with a Transformer and a gated convolutional unit is introduced to learn the source input representations. In the decoding phase, a pattern controlled decoder is proposed to control the output description pattern (such as category, length, and style) to ensure the quality of the description. For evaluation, we build a substantial collection of real-world products along with human-written descriptions. An extensive set of experiments with both human annotated data demonstrate the advantage of the proposed method for generation qualities. Finally, an online deployment shows significant benefits of our model in a real online shopping scenario, as measured by the click-through rate.

KEYWORDS

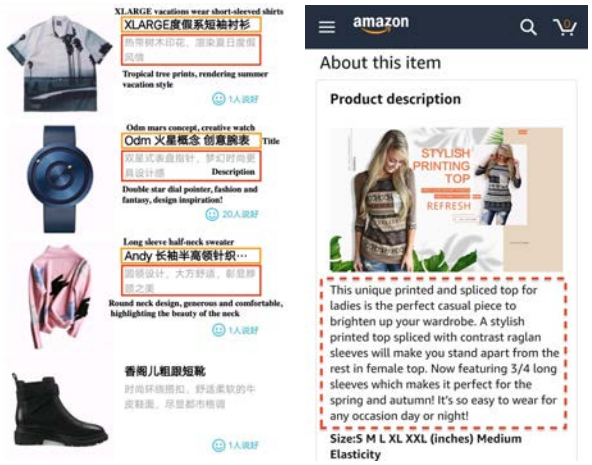
Natural Language Generation; Pattern-controlled; Product description;

ACM Reference Format:
Tao Zhang, Jin Zhang, Chengfu Huo, and Weijun Ren. 2019. Automatic Generation of Pattern-controlled Product Description in E-commerce. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313407

1 INTRODUCTION

The importance of content marketing for E-commerce has been significantly growing. Content marketing is one of the most effective ways for E-commerce websites to increase consumer engagement, namely because the customer chooses only to interact with the content that interests them, as opposed to traditional marketing. Major E-commerce sites, like Taobao, are gradually switching from trading platforms to content-based platforms since from users perspective the first is content consumption, followed by product consumption. On the Alibaba site, for example, there are many product oriented recommendations feeds flow, including product images, short videos, and recommendation reasons. However, the generation and maintenance of these content-based products require massive human costs. To reduce labor costs, product descriptions are expected to be automatically generated by the machine.

Generating product description is a difficult technical task of how to apply the state of art Natural Language Generation (NLG) technology to the field of E-commerce. Most human copywriters use product titles, attributes, and images, as well as the their own E-commerce industry experience to create product discriptions. Meanwhile, the writing techniques of copywriters can make a tremendous difference in the quality of generated product descriptions. Thus, it is a hugely difficult task to replace human being to



(a) Recommendation page on taobao (b) Detail page on Amazon

Figure 1: Product descriptions are widely used in E-commerce website Taobao and Amazon

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313407

generate product description for a machine. Product title and properties are unstructured datasets which are cluttered and disordered. The machine needs to analyze the product title to capture essential product information. Due to the diversity of E-commerce platform, it is necessary to control the pattern of description writing. For example, in the home page on mobile devices, the length of product description should be as short as possible without losing essential information owing to the limited display space on mobile phones. Another example is controlling style in the product description. In sales & promotion, we expect to add marketing-styled information into product description.

Generating product description can be viewed as a special kind of natural language generation. In NLG communities, machine translation and summarization are similar tasks to text generation. But the difference is that the length of the product description is not strictly limited to be approximately equal to or less than that of the source text, which is a controlled algorithm parameter in this task. Besides, compared with the traditional summarization task, this project has more restrains: **1)** Do not display any irrelevant product information. The objectivity and accuracy of product description shall be ensured; **2)** Try to focus on essential product information, like product categories and brands; **3)** Try to control the pattern of writing, including length and style. For consumers and sellers, wrong information in the product description is disallowed. An E-commerce study conducted by NNgroup[1] found that 20% of overall task failures when the user failed to complete a purchase could be attributed to incomplete or unclear product information. There are many successful abstract generation methods for news and wiki, like pointer-generator [36], but lots of wrong information can be found from generator mode. When generating long description based on the product title and property, we need to pay attention to the consistency between description and product. Moreover, the sequence to sequence (seq2seq) method prefers to generate safer text. When generating long text, seq2seq method will produce a large number of duplicated sub-sequences which seriously affects the quality of the final text. Thus, how to grasp essential product information more accurately and generate the corresponding product descriptions is particularly important. In addition, many similar products have similar key information at the Alibaba site. As a result, the description produced by this vanilla attention-based method are also similar, which is terrible for recommending similar products. For this reason, the diversity of product description needs to be considered as well.

This paper puts forward an attention based and pattern controlled neural network to solve the above three restrains. For the first restrain, we refer to the pointer-generator mechanism which used to solve the problem of out of vocabulary (OOV) words, like the brand names and commodity names in our task. Besides, we introduce coordinate encoders equipped with transformer[41] and Gated Convolutional Neural Network (Gated-CNN)[15] to learn the dependencies and representations among the words of product titles and properties on the input side which aims to improve the keywords capture ability when generating the text. The experiment proves that these two methods can improve the category and band accuracy of the generated text. Regarding the second problem, we use a name entity recognition tool called Term Weight to extract keywords in text, such as style, material, marketing information, and others. These keywords are fed into the decoding phase to control the output style. The third constraint is handled by adding length vector for decoding phase. The traditional method of length control is to limit the maximum search depth of decoding by truncation, which is so violent that severely impacts user's experience. In this paper, a length vector is introduced to restore the length information of each time step. We automatically label the length of descriptions in the training phase and made model learn length information. Then, the length of the generated text can be restrained by setting customized length vector when decoding.

To the best of our knowledge, this work is the first attempt to apply the neural language generation technology to product description generation. Our contributions are three-fold:

1. We put forward a product description generation technology by multi-task learning, which is far advanced than traditional generation method by seq2seq model;
2. We introduce a pointer-generator architecture with coordinate encoders (Transformer and Gated CNN) to extract the source features, which can capture the local correlations and ensure the accuracy and diversity of output description generation;
3. We intervene in the pattern of the text, such as the length and the style of the product description, to reduce model development period when switching the shopping scenarios and enhance the category accuracy.

2 BACKGROUND

Given the input sequence of product title, our goal is to generate the product description. We discuss this description generation problem in the sequence-to-sequence problem setting. Suppose that the input sequence $x = (x_0, x_1, x_2 \ldots x_M)$ and the output sequence $y = (y_0, y_1, y_2 \ldots y_N)$.

2.1 Sequence to Sequence Model

Recently, sequence-to-sequence (seq2seq) model has been widely used in machine translation and summarization system. The seq2seq model usually contains two Recurrent Neural Network (RNN) components, an encoder and a decoder. The encoder gets the context vector of the source sequence x and the decoder transforms the context vector c to the target sequence, given x.

$$\boldsymbol{h}_t = f(\boldsymbol{h}_t, \boldsymbol{w}_t)$$

$$\boldsymbol{c} = g(\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_N)$$

where h_t is the RNN hidden state at time t and w_t is the word embedding of word t. f is the activation function of RNN unit. In practice , gating mechanisms such as LSTM [19] and GRU [8] has been shown to exhibit better performance than vanilla RNN. The context vector C is also called the attention vector and is calculated using an attention mechanism [1] as a weighted sum of annotations of the encoder states.

$$C_j = \sum_{i=1}^{n} \alpha_{ji} h_i$$

[1] https://www.nngroup.com/reports/ecommerce-user-experience/

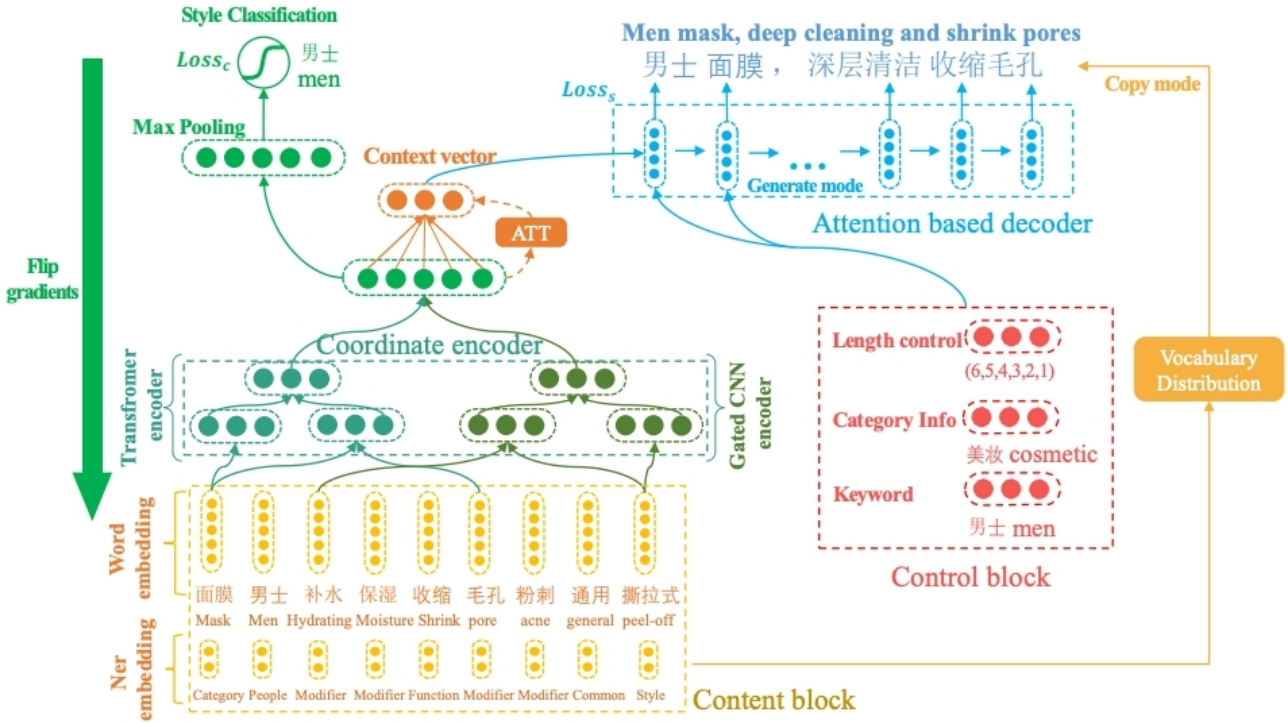


Figure 2: Architecture of Pattern Controlled Pointer-Generater Network.

Where a_{ji} are attention weights corresponding to each encoder hidden state output hi obtained as follows:

$$\alpha_{ji} = \frac{\exp(z_i)}{\sum_{k=1}^{n} \exp(z_k)}$$

Activations $z_k = a(d_{j-1}, h_k)$ are calculated by using a context function such as the dot product between the decoder state d_{j-1} and the encoder hidden states h_k. The decoder is also an RNN component, which is used to predict a target sequence,

$$\boldsymbol{d}_t = f(\boldsymbol{y}_{t-1}, \boldsymbol{d}_{t-1}, \boldsymbol{c})$$

$$p(y_t = w | y_{[0:t-1]}; x) = g(y_{t-1}, d_{t-1}, c)$$

where d_t is the decoder hidden state of time step t, y_t is the predicted output word symbol at t.

2.2 Copy mechanism model

The copy mechanism model [36] is a pointer-generator network, which is a hybrid between seq2seq network and pointer network [42]. This model allows both copying words via pointing and generating words from a fixed vocabulary which can be viewed as a balance between extractive and abstractive approaches. This model obeys the sequence-to-sequence architecture. Given the attention distribution a_t and the context vector $\boldsymbol{C}$, we can calculate the generation probability $p_{gen} \in [0, 1]$ as:

$$p_{gen} = f(C_t, s_t, y_t)$$

where f is a sigmoid function, s_t is decoder state and y_t is decoder input. In addition, p_{gen} is the soft switch to choose between generating a word from vocabulary or copying a word from source sentence according to the attention distribution a_t. The copy mechanism calculates the probability distribution over the extended vocabulary which consists of the vocabulary and all words from the source sentences:

$$P(w) = p_{gen} P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t$$

Besides, the $P_{vocab}(w)$ is zero if w is an out-of-vocabulary (OOV) words. Similarly, a_i^t is zero if w not appear in the source sentence. In this way, the copying mechanism has the ability to deal with the issues of OOV words by repeating themselves which improves accuracy.

3 PATTERN CONTROLLED P-G NETWORKS

Although pointer-generator network gets good performance in practice especially in the field of summarization, it still works not so well in product description generation especially when generating a long description. The copy mechanism prefers to repeat source words with higher value of attention distribution which results in higher repetition of generated descriptions. Also, copy mechanism can't recognize the key information such as brand names and categories.

To deal with this issue, we proposed a Pattern-Controled Pointer-Generator Network (PGPCN) as shown in Figure 2. Basically, we use

the seq2seq model as the main building block. However, unlike the traditional RNN-based sequence model, we instead use coordinate encoders with attention mechanism to generate the context vector. Also, a style classification is added to encourage the encoder to attend to style-decorrelated tokens by gradients reverse. Finally, we use the features in the control module to generate outputs in decoders.

3.1 Content Block

Given the product title and property, we use a specialized Named Entity Recognition (NER) tool[2] for E-commerce to label entities in product title. In total there are 36 types of entities, which are of common interest in E-commerce scenario, such as "色"(color), "格"(style) and "修" (modifier). For example, in segmented product title "包Nike 品牌的色" (Nike red sweatpants with free shipping), "包"(Free shipping) is labeled as Marketing Service, "Nike" is labeled as Brand, "色"(Red) is labeled as Color and "" (Sweatpants) is labeled as Category. then we look up an embedding matrix E to get the word embedding $E = (e_0, e_1, e_2 \ldots e_D)$ and word NER embedding Ner to get the named entity recognition(NER) embeddings $Ner = (ner_0, ner_1, ner_2 \ldots ner_D)$. D denotes the dimension of embeddings and N denotes the vocabulary size. The input word vector $Xi = E_i + Ner_i$. By combining the NER information into the word vector, the model can easily recognize the important source words while striving to preserve them in the outputs at decode time to aid reproduction of factual product details.

3.2 Coordinate Encoder

To enhance the accuracy and diversity of the generated description, we fed the source token vectors of the product title and product properties into coordinate encoders (Transformers [41] feature attention networks with multiple attention heads and convolutional layers equipped with gated linear units [15]). Different from the traditional RNN encoder, the Transformer encoder and Gated-CNN encoder are parallel structures which can speed up training time. In recent work, Chen et al.[40] find that hybrid architectures with a Transformer encoder and an RNN decoder can outperform a pure Transformer model. They speculate that the Transformer encoder is better at encoding or extracting features than the RNN encoder. Thus, the Transformer encoder can improve the output accuracy of our model. CNNs are hierarchical networks, in that convolutional layers capture local correlations. The n-gram features captured by CNN encoder can enhance the diversity of the output description. Besides, our model is equipped with Gated Linear Units and residual connections [10].

For Transformer encoder, we compute the hidden state h_i^l as:

$$\overrightarrow{h}_i^l = \overrightarrow{h}_i^{l-1} + f(\text{self} - \text{attention}(\overrightarrow{h}_i^{l-1}))$$

Scaled dot-product attention is adopted as the basic attention function in the Transformer, which describes as:

$$\text{attention}(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})$$

where f represents a feed forward network with ReLU as the activation function and layer normalization.

As for convolutional encoder, the local context size depends on the size of kernel and number of layers. To keep the output length same as the input, the CNN model adds padding at each layer. Also, we add residual connections from the input of each convolution to the output of the block.

$$\overleftarrow{h}_i^l = \overleftarrow{h}_i^{l-1} + f_{cnn}(W^l[\overleftarrow{h}_{i-\lfloor k/2 \rfloor}^{l-1}, \ldots, \overleftarrow{h}_{i+\lfloor k/2 \rfloor}^{l-1}] + b^l])$$

where a convolutional kernel $k \in \mathbb{R}^{d \times q \times d}$ is applied to each possible window of q words. ConvS2S choose a Gated Linear Units (GLU) [10] which can be viewed as a gated variation of ReLUs.

$$f([A, B]) = A \otimes \sigma(B)$$

where A, B are the outputs of two stacked convolutional networks, $\otimes$ is the point-wise multiplication. The gates $\sigma(B)$ control which inputs A of the current context is relevant. At last, the encoder outputs are the average of Transformer outputs and Gated CNN outputs $h^l = \text{avg}(\overrightarrow{h^l}, \overleftarrow{h^l})$.

3.3 Attention based Decoder

Repetition is a severe problem for language generation task, especially for generating long text. We use the coverage model to solve this problem. A coverage vector $\boldsymbol{c}$ is maintained to keep track of which source words have been translated('covered') in the past decoding time steps.

$$c^t = \sum_{t'=0}^{t-1} a^{t'}$$

c^0 is initialized as a zero vector denotes that no source words have been translated. Intuitively, coverage vector is used as an additional input to the attention mechanism, which reminds the attention mechanism about its previous decisions.

$$e_i^t = v^T \tanh(W_h \boldsymbol{h}_i + W_s \boldsymbol{s}_t + w_c \boldsymbol{c}_i^t + \boldsymbol{b}_{attn})$$

The attention distribution is calculated as:

$$a^t = \text{softmax}(e^t)$$

Next, we use the attention distribution to produce a context vector $\boldsymbol{C}$ which is a weighted sum of the encoder hidden states:

$$C_t = \sum_i a_i^t h_i$$

3.4 Control Block

Although the end-to-end network described above can works very well, it still generate wrong informations in practice. Alse, the pattern of generated descriptions are uncontrollable which is inflexible for online deployment. To address this issue, we introduced a control block to intervene in the pattern of description in decoding phase.

As shown in Figure 2, there are three components in the control module: style control, length control, and category control. We use the style keywords to represent the text style, described as $\boldsymbol{k}$ which share the embeddings with input words:

$$k_i = E_{k_i} + Ner_{k_i}$$

[2]https://www.alibabacloud.com/zh/product/machine-learning

Also, we look up an embedding matrix E_{cate} to get the category embeddings $E_{cate_i} \in \mathbb{R}^{d^c \times N^c}$ where d^c represents the dimension of category embeddings and N^c represents the maximum size of category. As for length control, a vector $\boldsymbol{L}$ is maintained to represents the distance between each word i to the end $\langle EOS \rangle$. Suppose the length of output text is n, $\boldsymbol{L}$ can be described as:

$$\boldsymbol{L} = (L_0, L_1, ..., L_n)$$

where $L_i = distance(i, \langle EOS \rangle)$ is the distance from position i to the end. At the decoding phase, we initialize L_0 with the value of target length. In addition, the L_i will decrease one at each time step before it fed into the model. When L_i equals 0, it means next word is the end of the sentence $\langle EOS \rangle$ and prediction will be terminated. The length parameter tells the model how much longer until the ending of the decoder. Besides, we look up an embedding matrix $E_L \in \mathbb{R}^{d^l \times N^l}$ to get the length embeddings before we feed it into the encoder. Here, d^l denotes the dimension of the length embeddings and N^l denotes the maximum length of target text.

We follow the architecture of pointer-generator network, so the generation probability p_{gen} for timestep t is calculated from the context vector C_t, the decoder states S_t , the decoder input x_t and the pattern control parameter E_{Li}, E_{Cate_i}, k_i:

$$p_{gen} = \sigma(w_i^T \boldsymbol{C}_t + w_s^T \boldsymbol{S}_t + w_x^T \boldsymbol{x}_t + w_L^T \boldsymbol{E}_{\boldsymbol{L}t} + w_{cate}^T \boldsymbol{E}_{\boldsymbol{cate}t} + w_k^T \boldsymbol{k}_t + \boldsymbol{b})$$

3.5 Classification Block

As shown in Figure 2, we feed the encoder vector into a style classification network. We draw from a line of research in the style of [5, 13, 20, 33]. This method involves passing gradients through a gradient reversal layer, which multiplies gradients by a negative constant, i.e. -1, as they propagate back through the network. Intuitively, this encourages parameters to update away from the optimization objective.

If $Loss_s$ and $Loss_c$ are the losses from two different prediction task of our network, where s represents description generation task, and c represents classification task. Thus, the loss function of our multi-task framework becomes $Loss = Loss_s + Loss_c$. The classification loss $Loss_c$ can be described as

$$Loss_c = \mathbb{E}_{h^i \sim \mathcal{H}} \left[-\log p_c(k_j \mid h^i) \right]$$

where k_j is the style keyword represents the style of product description i as discussed in 3.4. However, when backpropagating from each prediction network to the encoder, we reverse the gradients of the networks that are predicting style. This means that the prediction networks still learn to predict style, but the encoder is forced to learn style-invariant representations which are not useful to these downstream tasks. We hope that such representations encourage the encoder to attend to style-decorrelated tokens. In this way, we can easily control the style from a encoder vector without style features. Then we can use the style parameter in control block to generate the text in a specified style.

Table 1: The statistics of the pair dataset.

Train dataset size	3,516,805
Test dataset size	50,000
Avg. length of original titles	12
Avg. length of product properties	6
Avg. length of product description	25
Max. length of product description	50

Table 2: Parameter settings of our system.

Dim. of word embeddings	128
Dim. of ner embeddings	8
Num. of encoder/decoder hidden units	512
Adam optimizer	$lr = 1e-4$
Beam size	5
Minibatch size	128
Product category size	52
Vocabulary size	50000

4 EXPERIMENTS

4.1 DataSet

Our proposed method mainly requires two kinds of data: extended product description data and product title data[3]. These data are collected from **taobao**[4] a popular E-commerce website in China, since there is no public benchmark dataset for our task yet. We leverage the product description data from a product recommendation channel and product detail page. Displayed long descriptions are all written by experienced copywriters. We exclude the products whose descriptions are shorter than 8 Chinese characters and longer than 50 Chinese characters, since the product descriptions should be long enough to cover the essentials but short enough to keep it interesting. Moreover, product titles and property data are crawled from the detail page.

These two kinds of datasets are merged and organized as pairs, i.e.,(product title, product property and product description). Therefore, the dataset can be represented as $\langle T, D \rangle$, where T means the products' titles and the products' properties, and D means the products' descriptions. The details of our dataset are shown in Table 1.

4.2 Comparison Methods

To verify the effectiveness of our proposed model, we implement rich description generation methods. We compare our method with popular neural machine generation architectures, including:

- Vanilla sequence-to-sequence (**Seq2Seq**) is a basic encoder-decoder model based on LSTM unit and attention mechanism [1].
- Pointer-Generator (**Ptr-Gen**) [36] is a hybrid neural network model combing Seq2Seq-Gen with pointer network . Besides copying words from the input, it can also generate words from a predefined vocabulary.

[3]The dataset and code will be released soon.
[4]https://www.taobao.com

- Convolutional-based NMT (**ConvS2S**) [15] are hierarchical networks which is equipped with Gated Linear Units (GLU) and residual connections.
- Transformer-based NMT (**Trans**) [41] rely heavily on self-attention networks. Each token is connected to any other token in the same sentence directly via self- attention. Transformers feature attention networks with multiple attention heads which is more fine-grained, compared to conventional 1-head attention mechanisms.

4.3 Experiment Setup

For all experiments, we use the Xavier scheme [16] for parameter initialization, where weights are initialized using a normal distribution $\mathbf{W}_{i,j} \sim \mathcal{N}(0, \sigma)$, $\sigma = \sqrt{\frac{2}{n_{in}+n_{out}}}$; n_{in} and n_{out} are numbers of the input and output units of the network. We train all models using Adam with leaning rate $lr = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e-8$. We use 128-dimensional word embeddings, 8-dimensional word ner embeddings and 512-dimendional hidden states for LSTM units in both encoder and decoder. For bidirectional implementations, we linearly transform the forward hidden states and backward hidden states into 512-dimensional states.

We implement all the models in Tensorflow1.4[5]. For Ptr-Gen[6] we modify the I/O of the code to fit our dataset. For ConvS2S and Transformer[7], we use an open-source Python library OpenNMT. All the models are trained on a single Tesla P100 GPU with a batch size of 128. At train time, we use maximum encoding step 10 and maximum decoding step size 50. All training instances are sorted by the target sequence length and partitioned into mini-batches. The shorter sequences are padded to have the same length as the longest sequence in the batch. All batches are shuffled at the beginning of each epoch. At test time, we use beam search with beam size 5 and maximum decoding step size 20.

For the dataset preprocessing, all models are implemented based on Chinese characters. We use a Chinese word tokenizer and a pre-trained Ner tool for E-commerce to segment and label entities. Finally, our vocabulary contains 50k words according to the word frequency and 36 word entities. More detailed network parameters are presented in Table 2.

4.4 Automatic Evaluation

To evaluate the content prediction performance, we use some standard metrics (eg., BLEU [32], ROUGE [26], METEOR [2]). BLEU metrics is widely used for text generation tasks, such as machine translation, summarization. The metric analyzes co-occurrences lexical n-grams units between the candidate and references. For BLEU metric, we report BLEU-1, BLEU-2 here. The ROUGE metric measures the description quality by counting the overlapping units (e.g., n-grams) between the generated candidate and reference descriptions. Following the common practice, we report the F1 scores for ROUGE-1, ROUGE-2, and ROUGE-L. In addition, the METEOR metric was introduced to address several weaknesses in BLEU. It is calculated based on an explicit alignment between the unigrams in the candidate and references. In this work, the METEOR scores are evaluated in exact word match mode. We obtain the BLEU and METEOR scores using the nlg-eval11[8] packages, and ROUGE scores using python-rouge12[9] package.

The results of the last group indicate that the control block is helpful for description generation. Pattern-Controled Pointer Generator Network (PGPCN) achieves higher BLEU, ROUGE and METEOR scores on test dataset than other methods. PGPCN is a hybrid architecture with a Transformer encoder and an RNN decoder. Transformer encoder is proved to be better at encoding and extracting features than the RNN encoder. Also, we use the copy mechanism and control block to improve the quality of the output description. We did a comparative experiment that only used word embeddings as encoder input without NER information as shown in the last two group of Table 3. The results show that encoder with multi-data input including word embeddings and NER embeddings helps the model to get better performance. By using the NER information, the model can easily recognize the important source words while striving to preserve them in the outputs at decode time. This is consistent with our expectation as discussed in Section 3.1.

Also, we calculate the differences of metric score in different length, here we compare Chinese character size of length 10 with length 20. We use LENGTH-GAP represents the average retention rate by dividing the metric score of length 10 into score of length 20. The results show that our PGPCN method can reserve more information in short length than the other method. This is because our model takes the length as a model parameter during training step, it can memorizes the distance to the end $\langle EOS \rangle$, so it will compress key information in advance. However, the model without length control can only get fixed length by truncating which will result in loss of information.

4.5 Key Information Retention Test

For product description, it is particularly important to retain some key information (i.e., band names, commodity names) which is an essential requirement to attract people. This information helps users understand the product and accurate the descriptions of products. The evaluation metric for this task is the error rate of the brand names (B-ACC.) and commodity names (C-ACC.) in the generated product description. If the key information is missing or getting wrong in the product description, this prediction will be marked as an error. We also test the model on the sampled online data from a real-world E-commerce website.

The results of each model on test sets are shown in Table 4, seq2seq achieves the worst results in accuracy task due to the simpleness of the model architecture. Also, we find that models with pointer-generator mechanism can get higher accuracy than regular generator model, since the pointer mechanism can copy the relevant words to the output text directly. The conventional NMT model like ConvS2S and Transformer perform poor in Accuracy metric. Both of them are stuck in the source words, while some words in product title are ambiguous due to the similarity of category between E-commerce product.

[5]https://www.tensorflow.org
[6]https://github.com/abisee/pointer-generator
[7]https://github.com/OpenNMT/OpenNMT-py
[8]https://github.com/Maluuba/nlg-eval
[9]https://github.com/tagucci/pythonrouge

Table 3: BLEU, ROUGE (F1), and METEOR scores on the test set. Baselines on the top group are seq2seq. Bold scores are the best overall.

		BLEU-1	BLEU-2	ROUGE-1	ROUGE-2	ROUGE-L	METEOR	Info-Retention
Seq2Seq	Length 10	6.13	2.49	11.05	3.43	10.80	9.98	0.68
	Length 20	11.95	4.71	17.15	3.56	16.17	12.82	
Ptr-Gen	Length 10	7.29	3.68	12.94	5.68	12.91	13.70	0.72
	length 20	12.70	5.81	18.05	7.60	17.62	15.07	
ConvS2S	Length 10	7.54	3.71	13.98	7.78	13.17	14.81	0.71
	Length 20	13.41	6.22	20.62	8.34	20.21	17.61	
Trans	Length 10	7.97	3.73	13.39	7.76	13.36	13.95	0.71
	Length 20	13.71	6.58	20.09	8.35	19.91	16.59	
PGPCN$_{noNER}$	Length 10	7.43	3.12	12.70	7.17	12.43	13.15	0.75
	Length 20	12.09	5.55	17.73	7.22	17.32	14.87	
PGPCN	Length 10	**9.38**	**4.32**	**15.68**	**10.01**	**15.61**	**15.35**	**0.76**
	Length 20	**15.56**	**6.92**	**22.00**	**10.07**	**21.86**	**16.94**	

Table 4: Results of key info retention experiment. Bold scores are the best.

	Length	C-ACC.	B-ACC.	Repete RATE
Seq2Seq	10	3.72	4.98	7.76
	20	3.46	4.97	
Ptr-Gen	10	1.33	1.09	6.62
	20	1.15	1.03	
ConvS2S	10	2.58	2.57	**4.94**
	20	2.34	2.53	
Trans	10	1.57	2.96	6.94
	20	1.65	2.54	
PGPCN$_{noCB}$	10	1.27	0.90	5.17
	20	1.06	0.73	
PGPCN	10	**0.30**	**0.22**	5.36
	20	**0.28**	**0.21**	

Table 5: Manual evaluation results. Bold scores are the best. The improvements from baselines to PGPCN is statisti- cally signicant according to two-tailed t-test (p < 0.05).

Model	Completeness.	Consistency.	Readable.	Attract.
Seq2Seq	76.32%	62.90%	1.13	1.24
Ptr-Gen	93.13%	90.13%	3.52	2.92
ConvS2S	83.37%	80.23%	4.27	3.45
Trans	87.29%	82.95	4.53	3.67
PGPCN$_{nC}$	**94.51**%	91.07%	4.55	3.55
PGPCN	93.38%	**95.73**%	**4.60**	**3.96**

Moreover, the significant improvements on Accuracy metrics demonstrate that our method PGPCN can better recognize the commodity names and brand names after the intervention of the Control Block (CB). We also calculated the word repetition rate (Repete RATE.) of all description generated from each model. The results show that ConvS2S model performed best and got the lowest repetition rate. This is because the ConvS2S model has a robust capability to learn compressed expressions and address sentences with variable lengths. Also, the convolutional units can extract the N-gram features which is helpful to understand the text. As a result, our method has lower repetitive rate score than other non-convolutional methods.

4.6 Manual Evaluation

Like the previous work ([12], [39]), we also conduct manual evaluation on the generated product descriptions. As such, evaluation by human judges of the quality of generated text is the best measure of our methods' quality. We randomly sampled 500 products from our test datasets. Three experienced E-commerce website users were asked to measure the quality of the product descriptions. Three perspectives are considered during manual evaluation process: (1) **Completeness**, is the key information properly kept in the product description? (2) **Consistency**, is the generated description consistent with the given styles? (3)**Readability** (Readable), how fluent, grammatical the product description is? (4) **Attractiveness** (Attract.), how attractive the product description is?

Essential information retention is an extra and crucial requirement for product description generation. We use a very strict criterion for this property, the generated product description will be assessed as one only if it correctly retains both brand name and commodity name, otherwise 0. To make the annotation results more consistent and reliable, we excluded instances with divergent ratings (i.e., their variance is greater than the threshold). The readability and attractiveness are assessed with a score from 1 to 5 (i.e., 1 for worst and 5 for best).

The average results are shown in Table 5. The results indicate that our PGPCN method outperforms the conventional neural language generation method. The effects on Completeness show that all models built with copy mechanism can get a particularly good mark, since the copy mechanism can better retain the keywords from the inputs. The significant improvements to Attract metric demonstrate that our PGPCN can learn style information from the keywords parameter from the control block. Also, comparing with the PGPCN with no classification ($PGPCN_{nC}$), the PGPCN with classification can get a higher score on Consistency metrics. It mainly because the classification block can filter the irrelevant style by the gradient reverse. Considering all three metrics, our PGPCN produces more readable and more informative descriptions which show the advantages of classification and control block.

4.7 Online A/B Testing

This subsection presents the results of online evaluation in the product recommendation page scenario of an E-commerce mobile app with a standard A/B testing configuration. We deploy it in a real word product during E-commerce website sales day[10]. The A/B testing lasted for a half day, since millions of people visit our website during that sales day and we have got adequate users for testing. For online deployment, we generate fixed length description for about five hundred thousand sales product using PGPCN model. Moreover, we use different keywords to intervene in the description pattern, such as "新品" (new product) for new product scenario recommendation and "促" (promotion) for promotional products sales scenario. The baseline method uses several template description that only changing the product name. The A/B testing system split online users equally into two groups and direct them into two separate buckets respectively (about 1 million user views (UV) for each bucket). Then for users in the A buckets (i.e., the baseline group), the display product descriptions are generated by using a predefined template. While for users in B buckets (i.e., the experimental group), the display product descriptions are generated by the PGPCN method. We adopt the Click-Though Rate (CTR) to measure the performance of our description. In our recommendation scenario, product descriptions are displayed on the front page which is a crucial deciding factor in determining whether to click the product or skip to another. It can be obtained as:

$$CTR = \frac{product_click}{product_PV}$$

where product_click is the click times of the product, product_PV is the page view times of the product. We calculated the overall CTR for all products on sales day (from 20180920 12:00 to 20180920 22:00). We found that the performance in the experimental bucket is significantly better than that in the baseline bucket. PGPCN improved the CTR by 5.21% over the baseline. The results of the CTR improvement indicates that the users are more likely to click the products with the PGPCN generated descriptions.

[10]https://www.1688.com

4.8 Case Study

In this subsection, we show three realistic cases of the description generated by different methods. Generally, models without pointer-generator mechanism method achieve worse performance as shown in Table 6. The generator models are more likely to generate some irrelevant information such as Korea in the first example generate by Ptr-Gen. Also, we find that ConvS2S and Transformer prefer to generate some safe words, which may get a higher score at beam search step. However, this words can't help to improve the quality of the product description.

From the second case, we also found that the model without Control Block could not recognize the product category precisely. The Transformer model regard table lamp as robot toy which results in poor user experience. Consider the brand name of two examples, it is obvious that the model with pointer mechanism can recognize the brand information and copy it to the output. Besides, all method can produce fluent and grammatic descriptions due to the encoder-decoder sequential learning method.

We also give an example of how style keywords affect the output description as shown in Table 7. We use different keywords as the input of control parameter. Apparently, the results show that the keywords can control the description generation in different style. This is quite impressive, if we consider the user feature, we can generate personalized product description for different users according to their preference

5 RELATED WORK

Automatic text generation from structured data (Data-to-Text) is an import task in Natural Language Generation (NLG). Its primary goal is to produce readable text in natural language, from a non-linguistic data input. In NLG communities, machine translation[23, 38] and summarization [9, 44], which are two similar tasks to description generation. Machine translation aims for complete transduction of the sentence to be translated. Summarization condenses one long article to several sentences to highly summarize the article. Differently, this form of NLG is typically taken to require addressing (at least) two separate challenges: content selection, the selection of an appropriate subset of the input sequence [3, 4], and surface realization, producing text according to a given grammar.[21, 29]. Traditionally, these two challenges have been modularized and handled separately by generation systems.

Previous researches exploit NLG task in various domains. For example, Reiter et al. [34] generate weather forecasts from meteorological records and Chen and Mooney [6] generate sportscasting text from robot soccer events in RoboCup dataset. Christina Sauper et al. [35] investigate an approach for creating a multi-paragraph overview article. Jinpeng Wang et al. [43] present a statistical framework that generates product description from product attributes. A hierarchical semi-Markov method is proposed [25] which first associates the text sequences to corresponding records and then generates corresponding descriptions from these records. These works use a content selection way by selecting relevant material from source data and organizing them which is grammatically and syntactically. While this method is robust and portable across domains, output sequence often suffers from coverage and coherence problem.

Table 6: Examples of generated product description by different method. “␣” denotes visible whitespace in Chinese context.

Product Title	LIM’s␣蓝山␣风味␣咖啡豆 LIM’s Blue Mountain Flavored Coffee Beans
Seq2Seq	这␣款␣咖啡␣是␣以␣优质␣的␣咖啡豆␣为␣原料␣，␣经过␣深度␣烘焙␣制作␣而␣成␣，␣带␣有␣微微␣的␣玫红色␣，␣非常␣美妙␣。 This coffee is made from high-quality coffee beans and deeply baked. It has a slight rose red color and is very beautiful.
Ptr-Gen	LIM’s␣蓝山␣风味␣咖啡豆␣，␣韩国␣原装进口␣，␣选用␣优质␣咖啡豆␣为␣原料␣，␣浓浓␣的␣，␣口感␣公主风␣柔滑␣。 LIM’s Blue Mountain coffee beans, imported from South Korea, made from high-quality coffee beans, taste smooth ,like a princess.
ConvS2S	␣蓝山␣风味␣咖啡豆␣，␣精选␣优质␣咖啡豆␣，␣经过␣烘焙␣，␣烘焙␣出␣更多␣的␣咖啡␣。 Blue Mountain flavored coffee beans, selected high quality coffee beans, blew for more.
Trans	␣蓝山␣风味␣咖啡豆␣，␣由␣阿拉比␣卡␣咖啡豆␣为␣原料␣，␣经␣烘焙␣后␣精心␣烘焙␣而␣成␣，␣散␣发出␣浓郁␣的␣坚果␣香气 Blue Mountain flavored coffee beans, from Arabica, medium roast, giving off a rich nutty aroma.
PGPCN	LIM’s␣蓝山␣风味␣咖啡豆␣，␣精选␣优质␣咖啡豆␣，␣经过␣精心␣烘焙␣制作␣，␣口感␣清新␣，␣味道␣香浓 LIM’s Blue Mountain flavored coffee beans, carefully selected and fresh roasted, enjoy the rich, yet smooth and full-bodied taste.
Product Title	耐尔金␣创意␣时尚␣复古␣机器人␣卧室␣床头␣台灯 Nilkin Creative Fashion Vintage Robot Bedroom Table Lamp
Seq2Seq	这␣是␣一␣款␣创意␣十足␣的␣台灯␣，␣采用␣优质␣树脂␣材料␣精心␣制作␣而␣成␣，␣造型␣可爱␣，␣充满␣童趣␣。 This is a creative table lamp, crafted from high-quality resin materials, cute and full of childlikeness.
Ptr-Gen	这␣是␣一␣款␣非常␣有␣创意␣的␣台灯␣，␣阅读灯␣与␣清爽型␣的␣完美␣结合␣，␣简约␣而␣不␣简单␣。 This is a very creative table lamp, the perfect combination of reading light and refreshing style, more than just simple.
ConvS2S	树脂␣材质␣机器人␣，␣触摸␣可调光␣，␣光线␣易␣透过␣智能␣时尚␣。 Resin-based robot, touch dimmable, smart and stylish, light is easy to pass.
Trans	床头␣灯␣机器人␣玩具，␣采用␣优质␣铁艺␣材质␣，全新␣的␣设计␣，␣安全␣环保␣。 Bedlamp robot toy, made of high quality wrought iron material, new design, safe and environmentally.
PGPCN	耐尔金␣卧室␣台灯␣，␣机器人␣造型␣，␣创意␣时尚␣，␣个性␣有趣␣，␣装饰␣效果␣极␣佳␣。 Nelkin bedroom robotic table lamp, creative design, seems vivid and full of personality, adds a neat decorative touch to the room.

Table 7: Examples of generated product description from different style with length 9. “␣” denotes visible whitespace in Chinese context.

Product Title 羊毛呢␣毛领␣时尚␣女␣大衣␣大码␣厚␣连帽␣外套 Wool Collar Fashion Women’s Coat Large Size Thick Hooded Coat	
KeyWord	Product Description
爆款 Top-selling	大家␣都␣在␣买␣的␣爆款␣连帽␣外套␣。 Top-selling hooded coat people were purchasing.
折扣 discount	年末␣折扣␣大␣作战␣，␣毛领␣大衣␣来␣帮你 Year-end discount battle, fur collar coat help you.
简约 simple	连帽␣外套␣，␣简约␣不␣单调␣的␣穿␣搭 Hooded jacket, simple and not monotonous
小公主 little princess	大毛领␣外套␣，␣做␣个␣温暖␣系␣的␣小公主 Big fur collar coat, to be a warm little princess

Recently, sequence-to-sequence (Seq2Seq) neural network model based on encoder-decoder are widely used in existing NLG approached.Remi Lebret [24] introduces a neural model for concept-to-text generation. Chen [7] propose a general neural networks model based on attentional sequence-to-sequence model for text generation. Tianyu Liu et al. [28] proposed a novel structure-aware seq2seq architecture which consists of a field-gating encoder and description generator with dual attention. Mei, Bansal, and Walter [30] also proposed a seq2seq model with an aligner between weather records and weather broadcast. The model used one-hot encoding to represent the weather records as they are relatively simple and highly structured.

Moreover, pointer networks have been widely explored in abstract generation task. Pointer networks [42] allows to produce output sequence consisting of elements from the input-side words and have been used for task like extractive summarization [17, 37], machine translation [18] and dialogue generation [11]. In addition, it has also been shown to be helpful for geometric problems [42], question answering [22, 45, 46], code generation [27], and language modeling [31]. This model uses the soft attention distribution by Bahdanau et al.[1] which allows the model to learn alignments between modularity. Pointer-generator [36] network provided a viable new approach for abstractive text summarization by extending pointer network in a soft or hard way to decide whether to generate a token from the predefined vocabulary or the source sequence.

In this paper, derived from the general pointer-generator network, we propose coordinate encoders (Transformer [41] encoder and gated convolutional encoder [14, 15] equipped with self attention mechanism) to learn the representation of source input data and a pattern controlled decoder generates sentences using an attention mechanism. To the best of our knowledge, we make the first attempt to develop a neural data-driven solution for product description generation. Besides this study is launched on a dataset

with up to two million product description data, which is much higher than the data scale of previous studies and make the results more convincing.

6 CONCLUSION AND FUTURE WORK

Inspired by the neural language generation theory and the importance of content marketing of E-commerce, we proposed a neural model for product description generation. However, different from the conventional task of language generation, product description generation has more constraints like repetition and accuracy, since it is a divergent task. We address this problem with a pattern-controlled neural model by leveraging pointer-generator mechanism paradigm. The proposed model uses coordinate encoders combining the Transformer features with the Gated CNN features. In addition, we use a control block to intervene in the pattern (ie., length, category, style) of output description. Extensive experiments demonstrate the validity of the proposed method and the superiority of our approach over four language generation baselines. Finally, online A/B testing shows the significant business impact of our model in a mobile E-commerce app.

For further improvements, an interesting direction would be using user profile features to generate product descriptions. We find out one product may have different selling point in different user's perspectives, so it is possible for us to generate personalized descriptions for different users according to the user profile and preference. Another direction is how to combine the picture features into our model, and we can get more product information to enrich the generated description.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*. 65–72.

[3] Regina Barzilay and Mirella Lapata. 2005. Collective content selection for concept-to-text generation. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 331–338.

[4] Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. *arXiv preprint cs/0405039* (2004).

[5] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2007. Analysis of representations for domain adaptation. In *Advances in neural information processing systems*. 137–144.

[6] David L Chen and Raymond J Mooney. 2008. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*. ACM, 128–135.

[7] Shuang Chen. 2018. A General Model for Neural Text Generation from Structured Data. *E2E NLG Challenge System Descriptions* (2018).

[8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[9] Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 93–98.

[10] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2016. Language modeling with gated convolutional networks. *arXiv preprint arXiv:1612.08083* (2016).

[11] Mihail Eric and Christopher D Manning. 2017. A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. *arXiv preprint arXiv:1701.04024* (2017).

[12] Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1481–1491.

[13] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research* 17, 1 (2016), 2096–2030.

[14] Jonas Gehring, Michael Auli, David Grangier, and Yann N Dauphin. 2016. A convolutional encoder model for neural machine translation. *arXiv preprint arXiv:1611.02344* (2016).

[15] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122* (2017).

[16] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 249–256.

[17] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393* (2016).

[18] Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. *arXiv preprint arXiv:1603.08148* (2016).

[19] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[20] Fredrik Johansson, Uri Shalit, and David Sontag. 2016. Learning representations for counterfactual inference. In *International Conference on Machine Learning*. 3020–3029.

[21] Dan Jurafsky and James H Martin. 2014. *Speech and language processing*. Vol. 3. Pearson London.

[22] Rudolf Kadlec, Martin Schmid, Ondrej Bajgar, and Jan Kleindienst. 2016. Text understanding with the attention sum reader network. *arXiv preprint arXiv:1603.01547* (2016).

[23] Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1700–1709.

[24] Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. *arXiv preprint arXiv:1603.07771* (2016).

[25] Percy Liang, Michael I Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*. Association for Computational Linguistics, 91–99.

[26] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out* (2004).

[27] Wang Ling, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, Andrew Senior, Fumin Wang, and Phil Blunsom. 2016. Latent predictor networks for code generation. *arXiv preprint arXiv:1603.06744* (2016).

[28] Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2017. Table-to-text Generation by Structure-aware Seq2seq Learning. *arXiv preprint arXiv:1711.09724* (2017).

[29] Wei Lu and Hwee Tou Ng. 2011. A probabilistic forest-to-string model for language generation from typed lambda calculus expressions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1611–1622.

[30] Hongyuan Mei, Mohit Bansal, and Matthew R Walter. 2015. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. *arXiv preprint arXiv:1509.00838* (2015).

[31] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843* (2016).

[32] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 311–318.

[33] Reid Pryzant, Young-Joo Chung, and Dan Jurafsky. 2017. Predicting Sales from the Language of Product Descriptions. In *Proceedings of the SIGIR 2017 Workshop on eCommerce (ECOM 17)*.

[34] Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence* 167, 1-2 (2005), 137–169.

[35] Christina Sauper and Regina Barzilay. 2009. Automatically generating wikipedia articles: A structure-aware approach. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*. Association for Computational Linguistics, 208–216.

[36] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368* (2017).

[37] Fei Sun, Peng Jiang, Hanxiao Sun, Changhua Pei, Wenwu Ou, and Xiaobo Wang. 2018. Multi-Source Pointer Network for Product Title Summarization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 7–16.

[38] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[39] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1171–1181.

[40] Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. 2018. Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures. *arXiv preprint arXiv:1808.08946* (2018).

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[42] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*. 2692–2700.

[43] Jinpeng Wang, Yutai Hou, Jing Liu, Yunbo Cao, and Chin-Yew Lin. 2017. A Statistical Framework for Product Description Generation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 187–192.

[44] Jingang Wang, Junfeng Tian, Long Qiu, Sheng Li, Jun Lang, Luo Si, and Man Lan. 2018. A Multi-task Learning Approach for Improving Product Title Compression with User Search Log Data. *arXiv preprint arXiv:1801.01725* (2018).

[45] Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905* (2016).

[46] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 189–198.

Iteratively Learning Embeddings and Rules for Knowledge Graph Reasoning

Wen Zhang
College of Computer Science,
Zhejiang University
wenzhang2015@zju.edu.cn

Bibek Paudel*
Stanford University
& University of Zürich
paudel@ifi.uzh.ch

Liang Wang
College of Computer Science,
Zhejiang University
21621254@zju.edu.cn

Jiaoyan Chen
Department of Computer Science,
University of Oxford
jiaoyan.chen@cs.ox.ac.uk

Hai Zhu
Alibaba Group
China
marvin.zh@alibaba-inc.com

Wei Zhang
Alibaba Group & AZFT Joint Lab for
Knowledge Engine
lantu.zw@alibaba-inc.com

Abraham Bernstein
Department of Informatics,
University of Zürich
Switzerland
bernstein@ifi.uzh.ch

Huajun Chen†
Zhejiang University & AZFT Joint
Lab for Knowledge Engine
China
huajunsir@zju.edu.cn

ABSTRACT

Reasoning is essential for the development of large knowledge graphs, especially for completion, which aims to infer new triples based on existing ones. Both rules and embeddings can be used for knowledge graph reasoning and they have their own advantages and difficulties. Rule-based reasoning is accurate and explainable but rule learning with searching over the graph always suffers from efficiency due to huge search space. Embedding-based reasoning is more scalable and efficient as the reasoning is conducted via computation between embeddings, but it has difficulty learning good representations for sparse entities because a good embedding relies heavily on data richness. Based on this observation, in this paper we explore how embedding and rule learning can be combined together and complement each other's difficulties with their advantages. We propose a novel framework IterE iteratively learning embeddings and rules, in which rules are learned from embeddings with proper pruning strategy and embeddings are learned from existing triples and new triples inferred by rules. Evaluations on embedding qualities of IterE show that rules help improve the quality of sparse entity embeddings and their link prediction results. We also evaluate the efficiency of rule learning and quality of rules from IterE compared with AMIE+, showing that IterE is capable of generating high quality rules more efficiently. Experiments show that iteratively learning embeddings and rules benefit each other during learning and prediction.

*Work done while at UZH.
†Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313612

CCS CONCEPTS

• **Information systems → Web Ontology Language (OWL)**; • **Computing methodologies → Knowledge representation and reasoning**; **Reasoning about belief and knowledge**; **Logical and relational learning**; **Rule learning**; **Learning latent representations**.

KEYWORDS

knowledge graph; reasoning; embedding; rule learning

ACM Reference Format:
Wen Zhang, Bibek Paudel, Liang Wang, Jiaoyan Chen, Hai Zhu, Wei Zhang, Abraham Bernstein, and Huajun Chen. 2019. Iteratively Learning Embeddings and Rules for Knowledge Graph Reasoning. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313612

1 INTRODUCTION

Many Knowledge Graphs (KGs), such as Freebase [2] and YAGO [33], have been built in recent years and led to a broad range of applications, including question answering [4], relation extraction [36], and recommender system [49]. KGs store facts as triples in the form of *(subject entity, relation, object entity)*, abridged as (s, r, o). Some KGs also have an ontology with class and property expression axioms which place constraints on classes and types of relationships.

Knowledge graph reasoning (KGR) can infer new knowledge based on existing ones and check knowledge consistency. It is attracting research interest and is important for completing and cleaning up KGs. Two of the most common learning methods for KGR are embedding-based reasoning and rule-based reasoning [26].

One of the crucial tasks for embedding-based and rule-based reasoning is to learn embeddings and rules respectively. *Embedding learning methods* such as TransE [3], HolE [27] and ComplEx [35] learn latent representations of entities and relations in continuous vector spaces, called *embeddings*, so as to preserve the information

and semantics in KGs. Embedding-based reasoning is more efficient when there are a large number of relations or triples to reason over. *Rule learning methods* such as AMIE[10] aim to learn deductive and interpretable inference rules. Rule-based reasoning is precise and can provide insights for inference results.

With different advantages, both embedding and rule learning are important for KGR, while they still have their own difficulties and weaknesses.

Sparsity Problem for Embedding Learning. One of the main difficulties for embedding learning is the poor capability of encoding sparse entities. For example, Figure 1 shows correlation between entity frequency and entity link prediction results measured in mean reciprocal rank (MRR), where higher values means better results. In Figure 1, the blue line shows there are a large portion of entities having only a few triples, revealing the common existence of sparse entity. The yellow line shows that the prediction results of entities are highly related to their frequency, and the results of sparse entities are much worse than those of frequent ones.

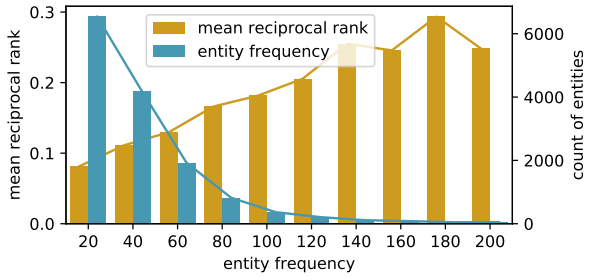


Figure 1: The numbers and mean reciprocal rank of different frequency entities based on ANALOGY results on FB15k-237.

Efficiency Problem for Rule Learning. The main difficulty in rule learning is the huge search space when determining rule structures and searching for support triples. For example, with a small KG containing 10 relations and 100 entities, the number of possible structures for a rule with 3 relations is 10^3 and the maximum number of supports for these rules is $100^{2*3} * 10^3 = 10^{15}$. Since the search space is exponential to the number of relations, it will be much larger for real KGs than this example.

With different advantages and difficulties, we argue that embedding learning and rule learning can benefit and complement each other. On the one hand, deductive rules can infer additional triples for sparse entities and help embedding learning methods encode them better. On the other hand, embeddings encoded with rich semantics can turn rule learning from discrete graph search into vector space calculation, so that reduce the search space significantly. Thus we raise the research question: **whether it is possible to learn embeddings and rules at the same time and make their advantages complement to each other's difficulties.**

In this paper, we propose a novel framework **IterE** that iteratively learns embeddings and rules, which can combine many embedding methods and different kinds of rules. Especially, we consider linear map assumption (Section 2.1) for embedding learning because it is inherently friendly for rule learning as there are special rule conclusions for relation embeddings in rules (Table 2). We also consider a particular collection of object property axioms defined in OWL2 (Table 1) for rule learning considering that semantics included in web ontology language are important for the development of knowledge graph.

IterE mainly includes three parts: (1) embedding learning, (2) axiom induction, and (3) axiom injection. Embedding learning learns embeddings for entities and relations, with input including triples existing in KG and those inferred by rules. Axiom induction first generates a pool of possible axioms with an effective pruning strategy proposed in this paper and then assigns a score to each axiom in the pool based on calculation between relation embeddings according to rule conclusions from linear map assumption. Axiom injection utilizes axioms' deductive capability to infer new triples about sparse entities according to axiom groundings and injects these new triples into KG to improve sparse entity embeddings. These three parts are conducted iteratively during training.

We evaluate IterE from three perspectives, 1) whether axiom improves sparse embeddings' quality and their predictions, 2) whether embedding helps improve rule learning efficiency and quality, and 3) how iterative training improves both embedding and rule learning during training. The experiment results show that IterE achieves both better link prediction performance and high quality rule learning results. These support our goal of making IterE complement the strengths of embedding and rule learning.

Contributions of our work are as follows:

- We propose an iterative framework that combines embedding learning and rule learning to explore the mutual benefits between them. Experiments show that it leads to better link prediction results using rules and embeddings together.
- We present a new method for *embedding learning with rules* based on axiom injection through t-norm based fuzzy logics. Experiments show that IterE can significantly improve embedding quality for the sparse part of a knowledge graph.
- We further identify a portfolio of ontology axioms for *rule learning with embedding* based on linear map assumption. Experiments show that IterE learns more high quality rules more efficiently than conventional rule learning systems.

2 PRELIMINARIES

2.1 Knowledge graph embedding

A KG $\mathcal{K} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$ contains a set of entities $\mathcal{E}$, a set of relations $\mathcal{R}$ and a set of triples $\mathcal{T} = \{(s, r, o)|s, o \in \mathcal{E}; r \in \mathcal{R}\}$. In a triple (s, r, o), the symbols s, r, and o denote subject entity, relation, and object entity respectively. An example of such triple is (Tokyo, *locatedIn*, Japan).

Knowledge graph embedding (KGE) aims to embed all entities and relations in a continuous vector space, usually as vectors or matrices called *embeddings*. Embeddings can be used to estimate the likelihood of a triple to be true via a score function $f : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow \mathbb{R}$. Concrete score functions are defined based on different vector space assumptions. We now describe two vector space assumptions commonly used in KGEs and their corresponding score functions.

(a) Translation-based assumption embeds entities and relations as vectors and assumes $\mathbf{v}_s + \mathbf{v}_r = \mathbf{v}_o$, in which $\mathbf{v}_s, \mathbf{v}_r$ and $\mathbf{v}_o$ are vector embeddings for s, r and o respectively. For a true triple, the relation-specific translation of subject embedding $(\mathbf{v_s} + \mathbf{v_r})$ is close to the object embedding $\mathbf{v_o}$ in embeddings' vector space.

Table 1: Conditions for object property expression axioms in OWL2 and translated rule formulation. OP refers to "ObjectProperty". OPE, with or without subscript, denotes Object Property Expression and x, y, z are entity variables. $\triangle_I$ is a nonempty set called object domain. $\cdot^{OP}$ is an object property interpretation function. When translating axioms into rule forms according to condition, we replace OPE in axioms with binary relation r in the context of knowledge graph and the number of OPE in EquivalentOP and the OPChain in SubOP is set to 2.

Object Property Axiom	Condition	Rule Form
ReflexiveOP(OPE)	$\forall x : x \in \triangle_I$ implies (x,x) $\in$ (OPE)^{OP}	(x, r, x)[1]
SymmetricOP(OPE)	$\forall x, y : (x, y) \in (OPE)^{OP}$ implies $(y, x) \in (OPE)^{OP}$	$(y, r, x) \leftarrow (x, r, y)$
TransitiveOP(OPE)	$\forall x, y, z : (x, y) \in (OPE)^{OP}$ and $(y, z) \in (OPE)^{OP}$ imply $(x, z) \in (OPE)^{OP}$	$(x, r, z) \leftarrow (x, r, y), (y, r, z)$
EquivalentOP(OPE_1 ... OPE_n)	$(OPE_j)^{OP} = (OPE_k)^{OP}$ for each $1 \leq j \leq n$ and each $1 \leq k \leq n$	$(x, r_2, y) \leftarrow (x, r_1, y)$
SubOP(OPE_1 OPE_2)	$(OPE_1)^{OP} \subseteq (OPE_2)^{OP}$	$(x, r_2, y) \leftarrow (x, r_1, y)$
InverseOP(OPE_1 OPE_2)	$(OPE_1)^{OP} = \{(x, y) \mid (y, x) \in (OPE_2)^{OP}\}$	$(x, r_1, y) \leftarrow (y, r_2, x)$
SubOP(OPChain(OPE_1 ... OPE_n) OPE)	$\forall y_0, \ldots, y_n : (y_0, y_1) \in (OPE)^{OP}$ and ... and $(y_{n-1}, y_n) \in (OPE_n)^{OP}$ imply $(y_0, y_n) \in (OPE)^{OP}$	$(y_0, r, y_2) \leftarrow (y_0, r_1, y_1), (y_1, r_2, y_2)$

(b) Linear map assumption embeds entities as vectors and relations as matrices. It assumes that the subject entity embedding $\mathbf{v}_s$ can be linearly mapped to object entity embedding $\mathbf{v}_o$ via relation embedding $\mathbf{M}_r$. In this case, for a true triple, the linear mapping of the subject embedding by the relation matrix ($\mathbf{v_s M_r}$) is close to the object embedding $\mathbf{v_o}$ in embeddings' vector space.

Thus the score function ϕ of two assumptions can be written as:

$$\phi_{translation} = sim(\mathbf{v}_s + \mathbf{v}_r, \mathbf{v}_o)$$
$$\phi_{linearmap} = sim(\mathbf{v}_s \mathbf{M}_r, \mathbf{v}_o) \quad (1)$$

where $sim(\mathbf{x}, \mathbf{y})$ calculates the similarity between vector $\mathbf{x}$ and $\mathbf{y}$.

From the assumption point of view, $\mathbf{v_s M_r} = \mathbf{v_o}$ (or $\mathbf{v_s} + \mathbf{v_r} = \mathbf{v_o}$) should exactly hold for true triples in the linear-map assumption (or translation-based assumption). From the modeling point of view, this is the optimization goal during learning, namely to be as close as possible to the equation in assumption. The learning process is done through either maximizing the objective or minimizing the error induced from assumptions given by their respective loss functions. Hence, the assumption equation usually does not exactly hold with learned embeddings, but their loss functions are designed to approach the assumption as much as possible.

In this paper, we adopt linear map assumption for embedding learning because many reasonable rule conclusions can be derived with relation embeddings based on this assumption (Table 2).

2.2 Rules Learning

Suppose $\mathcal{X}$ is a countable set of variables and C a countable set of constants. A rule is of the form $head \leftarrow body$, where $head$ is an atom over $\mathcal{R} \cup \mathcal{X} \cup C$ and $body$ is a conjunction of positive or negative atoms over $\mathcal{R} \cup \mathcal{X} \cup C$. An example of such rule can be:

$$(\mathrm{X}, hasMother, \mathrm{Y}) \leftarrow (\mathrm{X}, hasParent, \mathrm{Y}), (\mathrm{Y}, gender, \mathrm{Female}) \quad (2)$$

When replacing all variables in a rule with concrete entities in KG, we get a *grounding* of the rule. For example, one grounding of Rule (2) can be:

$$(\mathrm{Bob}, hasMother, \mathrm{Ann}) \leftarrow (\mathrm{Bob}, hasParent, \mathrm{Ann}), (\mathrm{Ann}, gender, \mathrm{Female}) \quad (3)$$

A grounding with all triples existing in knowledge graph is a *support* of this rule. For examples, if $(\mathrm{Bob}, hasMother, \mathrm{Ann}) \in \mathcal{K}$, $(\mathrm{Bob}, hasParent, \mathrm{Ann}) \in \mathcal{K}$ and $(\mathrm{Ann}, gender, \mathrm{Female}) \in \mathcal{K}$, then grounding (3) is a support for rule (2).

The results of rule learning is of the following form:

$$\alpha \; head \leftarrow body$$

in which $\alpha \in [0, 1]$ is a confidence score assigned to the rule by the learning method.

Incorporating logical rules into other learning system such as embedding learning is called *rule injection*. One way for rule injection is adding regularizer or other constraints to entity and relation representations by propositionalizing the rules. Another way is adding constraints to the constituent relations' representation in rules without direct effect on entity representations. As we want to get new information of sparse entities through rules, we chose propositionalization in this paper for rule injection.

2.3 OWL 2 Web Ontology Language Axioms

In this paper, instead of learning general Horn rules or closed-path rules as previous works[9][48], we are more interested in ontology axioms, the main components of knowledge graph ontologies, because they are important for enriching semantics in KGs.

OWL2 Web Ontology Language, informally OWL2[2], is an ontology language for Semantic Web with formally defined meaning and is a W3C recommendation. It defines multiple types of axioms, from which we select some of them as a guidance of rule structures. The selection is based on following principles: (i) the axioms are related with binary relations, the main components of rules in KG and (ii) they can infer new triples because rules are used to help add new information about sparse entities in this paper. Thus, we focus on object property expression axioms in OWL2 which are composed of binary relations in the context of KG. Finally, 7 types of object property expression axioms out of 14 are selected. The unselected axioms are mainly applied to help check the consistency in knowledge graph.

In OWL2, each axiom has its own condition revealing its semantics. The axiom is *satisfied* if its condition hold. We introduce the selected 7 types of object property expression axioms and their conditions in Table 1. We also translate the conditions of axioms into rule-form including a *head* and a *body* as introduced in the previous subsection. The translated rule forms are used to guide the structures of rules to be learned in this paper.

[1] the translated rule form of ReflexiveOP(OPE) only contain a *head*.
[2] https://www.w3.org/TR/owl2-primer/

3 METHOD

Given a knowledge graph $\mathcal{K} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$, our goal is to learn embeddings and rules at the same time and make their advantages complement each other's difficulties. As discussed in Section 1, embedding learning methods suffer from the problem of data sparsity and rule learning methods have a very large search space.

In this paper, we propose a general framework **IterE** which learns embeddings and rules in an iterative manner and can be applied to many KGEs that are based on linear map assumption. It includes three main parts:(i) embedding learning, (ii) axiom induction and (iii) axiom injection. Figure 2 shows the key idea of IterE with these three iterative parts.

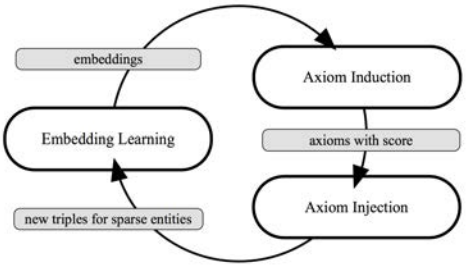


Figure 2: Overview of our IterE.

- **Embedding learning** learns entity embeddings **E** and relation embeddings **R** with a loss function $L_{embedding}$ to be minimized, calculated with input triples (s, r, o), each with a label related with its truth value. The inputs are of two types: triples existing in $\mathcal{K}$ and triples that do not exist in $\mathcal{K}$ but are inferred by axioms.
- **Axiom Induction** inducts a set of axioms $\mathcal{A}$ based on relation embeddings **R** from the embedding learning step, and assigns each axiom with a score s_{axiom}.
- **Axiom Injection** injects new triples about sparse entities in $\mathcal{K}$ to help improve their poor embeddings caused by insufficient training. The new triples are inferred from groundings of quality axioms with high scores in $\mathcal{A}$ from axiom induction. After axiom injection, with $\mathcal{K}$ updated, the process goes back to embedding learning again.

IterE is proposed based on the observation that embeddings learned with linear map assumption can fully support the axiom selected in this paper, while other assumptions such as translation-based assumption can't as pointed out in [50]. Inherently, for each type of axioms, a meaningful conclusion can be drawn with relation embeddings according to linear map assumption. For example, considering axiom `inverse`(*hasParent*, *hasChild*), if (*Mike*, *hasParent*, *John*) exists in knowledge graph, according to the condition and rule form of inverse axiom in Table 1, another triple (*John*, *hasChild*, *Mike*) can be inferred. Suppose embeddings of *Mike*, *John*, *hasParent* and *hasChild* are $\mathbf{v}_{Mike}$, $\mathbf{v}_{John}$, $\mathbf{M}_{hasChild}$ and $\mathbf{M}_{hasParent}$ respectively. According to the linear assumption for individual triples, we can get following two equations: $\mathbf{v}_{Mike}\mathbf{M}_{hasParent} = \mathbf{v}_{John}$ and $\mathbf{v}_{John}\mathbf{M}_{hasChild} = \mathbf{v}_{Mike}$. With these two equations, another equation can be deduced: $\mathbf{M}_{hasParent}\mathbf{M}_{hasChild} = \mathbf{I}$. Note that this conclusion equation is only related with two corresponding relation embeddings and is unrelated with concrete entities, thus it can be regarded as a general conclusion for axiom `inverse`(*hasParent*, *hasChild*). For other types of axioms, a general conclusion can be drawn in the same way. We list details of the conclusion for each type of axioms in Table 2. These conclusions about relation embeddings help guide axiom induction in this paper.

3.1 Embedding Learning

The input $\mathcal{I}$ of embedding learning is a set of triples with labels. Unlike previous KGEs, which only learn embeddings but not rules, positive input triple in IterE contains two parts: triples $(s, r, o) \in \mathcal{T}$ existing in original knowledge graph and triples $(s, r, o) \in \mathcal{T}_{axiom}$, where $\mathcal{T}_{axiom}$ is a set of triples inferred by axioms learned from embeddings. Negative input triples $(s', r', o') \in \mathcal{T}_{negative}$ are generated by randomly replacing s or o with $e \in \mathcal{E}$ or replacing r with $r' \in \mathcal{R}$ for $(s, r, o) \in \mathcal{T}$. Thus the input set $\mathcal{I}$ in IterE is as follows:

$$\mathcal{I} = \{((s, r, o), l_{sro}) | ((s, r, o)) \in \mathcal{T} \wedge \mathcal{T}_{axiom} \wedge \mathcal{T}_{negative}\} \quad (4)$$

where l_{sro} is the label for triple (s, r, o) to evaluate its truth value. $l_{sro} = 1$ for $(s, r, o) \in \mathcal{T}$ and $l_{sro} = 0$ for $(s, r, o) \in \mathcal{T}_{negative}$. For triples $(s, r, o) \in \mathcal{T}_{axiom}$ we assign $l_{sro} = \pi(s, r, o)$, where $\pi(s, r, o)$ is triple truth value predicted by axiom injection (Section 3.3).

With input $\mathcal{I}$, the loss function of embedding learning is calculated by mean of cross entropy loss among all n input triples and the training goal is to minimize the following loss function:

$$\min L_{embedding} = \frac{1}{n} \sum_{((s,r,o),l_{sro}) \in \mathcal{I}} [-l_{sro} \log(\phi(s, r, o)) - (1 - l_{sro}) \log(1 - \phi(s, r, o))] \quad (5)$$

As we adopt linear map assumption, the score function $\phi(s, r, o)$ for each triple (s, r, o) is defined as:

$$\phi(s, r, o) = sim(\mathbf{v}_s \mathbf{M}_r, \mathbf{v}_o) = \sigma(\mathbf{v}_s^\top \mathbf{M}_r \mathbf{v}_o) \quad (6)$$

in which $\mathbf{v}_s \in \mathbb{R}^{1\times d}, \mathbf{v}_o \in \mathbb{R}^{1\times d}$ are vector embeddings for subject and object entity. $\mathbf{M}_r \in \mathbb{R}^{d\times d}$ is matrix embedding for relation and σ denotes the sigmoid function. d is the embedding dimension. The similarity between two vectors is evaluated via dot product.

Our approach can be combined with many KGEs based on linear map assumption, such as DistMult[47] and ANALOGY[23]. In this paper, we adopt ANALOGY as it achieves state-of-the-art results on link prediction. ANALOGY is proposed to deal with an important kind of reasoning, analogy reasoning, in embedding learning. It imposes analogical structures among embeddings and requires linear maps associated with relations to form a commuting family of normal matrices. Specifically, the relation matrix embeddings are constrained as block-diagonal matrices with each diagonal block is either a real scalar, or a 2-dimensional real matrix in the form of $\begin{bmatrix} a & -b \\ b & a \end{bmatrix}$, where both a and b are real scalars.

After embedding learning, a collection of entity embeddings **E** and relation embeddings **R** are learned and **R** will be used in axiom induction.

3.2 Axiom Induction

Given relation embedding **R**, axiom induction aims to induce a set of axioms $\mathcal{A}$ and assign a confidence score to each axiom in $\mathcal{A}$. To achieve this, IterE firstly generates a pool of possible axioms $\mathcal{P}$

Table 2: Seven types of object property expression axioms selected from OWL2 ontology language. `OP` is the short for `ObjectProperty`. $\mathcal{K}$ denotes a KG and x, y, z are entity variables. v and M denote entity and relation embeddings respectively. I is identity matrix.

Object Property Axioms	Rule Form	According to Linear Map Assumption	Rule Conclusion
`ReflexiveOP`(r)	(x, r, x)	$\mathbf{v}_x\mathbf{M}_r = \mathbf{v}_x$	$\mathbf{M}_r = \mathbf{I}$
`SymmetricOP`(r)	$(y, r, x) \leftarrow (x, r, y)$	$\mathbf{v}_y\mathbf{M}_r = \mathbf{v}_x; \mathbf{v}_x\mathbf{M}_r = \mathbf{v}_y$	$\mathbf{M}_r\mathbf{M}_r = \mathbf{I}$
`TransitiveOP`(r)	$(x, r, z) \leftarrow (x, r, y), (y, r, z)$	$\mathbf{v}_x\mathbf{M}_r = \mathbf{v}_z; \mathbf{v}_x\mathbf{M}_r = \mathbf{v}_y,\ \mathbf{v}_y\mathbf{M}_r = \mathbf{v}_z,$	$\mathbf{M}_r\mathbf{M}_r = \mathbf{M}_r$
`EquivalentOP`(r_1, r_2)	$(x, r_2, y) \leftarrow (x, r_1, y)$	$\mathbf{v}_x\mathbf{M}_{r_2} = \mathbf{v}_y,\ \mathbf{v}_x\mathbf{M}_{r_1} = \mathbf{v}_y$	$\mathbf{M}_{r_1} = \mathbf{M}_{r_2}$
`subOP`(r_1, r_2)	$(x, r_2, y) \leftarrow (x, r_1, y)$	$\mathbf{v}_x\mathbf{M}_{r_2} = \mathbf{v}_y,\ \mathbf{v}_x\mathbf{M}_{r_1} = \mathbf{v}_y$	$\mathbf{M}_{r_1} = \mathbf{M}_{r_2}$
`inverseOP`(r_1, r_2)	$(x, r_1, y) \leftarrow (y, r_2, x)$	$\mathbf{v}_x\mathbf{M}_{r_1} = \mathbf{v}_y,\ \mathbf{v}_y\mathbf{M}_{r_2} = \mathbf{v}_x$	$\mathbf{M}_{r_1}\mathbf{M}_{r_2} = \mathbf{I}$
`subOP`(`OPChain`(r_1, r_2), r)	$(y_0, r, y_2) \leftarrow (y_0, r_1, y_1), (y_1, r_2, y_2)$	$\mathbf{v}_{y_0}\mathbf{M}_r = \mathbf{v}_{y_2},\ \mathbf{v}_{y_0}\mathbf{M}_{r_1} = \mathbf{v}_{y_1},\ \mathbf{v}_{y_1}\mathbf{M}_{r_2} = \mathbf{v}_{y_2}$	$\mathbf{M}_{r_1}\mathbf{M}_{r_2} = \mathbf{M}_r$

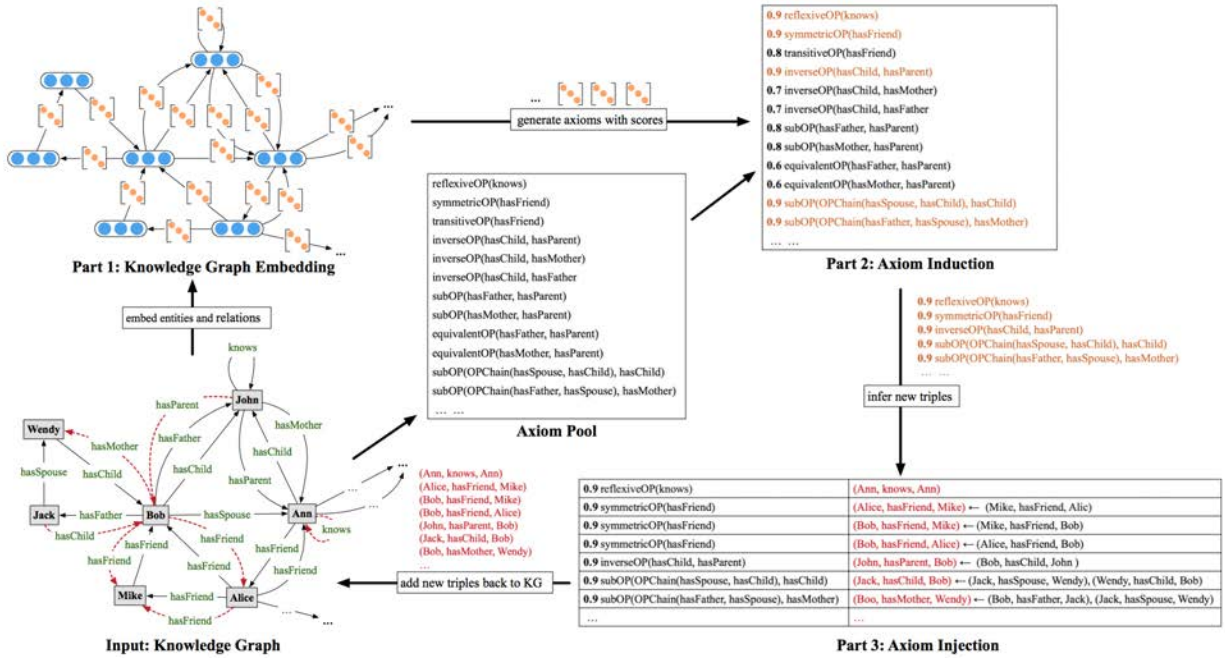


Figure 3: Detailed example for the process of IterE.

with an effective pruning strategy. Then it predicts a score of each axiom $a \in \mathcal{P}$ based on calculation with $\mathbf{R}$.

3.2.1 Axiom pool generation. Before calculating axiom scores with relation embedding $\mathbf{R}$, relation variables r, r_1, r_2, and r_3 in Table 2 should be replaced with concrete relations. Axiom pool generation generates a pool of possible axioms $\mathcal{P}$ by searching for possible axioms with the number of *support* greater than 1.

One intuitive way of searching for possible axioms is generating axioms by replacing all relation variable in each type of axioms with each relation and then check the number of support for them, but this suffers from a huge search space. Another way is to generate axioms via random walk on knowledge graph while this method can't ensure coverage of axioms. Therefore axiom pool generation is not an easy task because it has to achieve a good balance between search space and coverage of highly possible axioms.

In this paper, we propose a pruning strategy combining traversing and random selection. There are two steps for generating possible axioms for each relation $r \in \mathcal{R}$ in IterE.

- *step 1*: generate axioms or partial axioms: `ReflexiveOP`(r), `SymmetricOP`(r), `TransitiveOP`(r), `EquivalentOP`(r', r), `subOP`(r', r), `inverseOP`(r', r) and `subOP`(`OPChain`(r', r''), r), , in which $\{r', r''\}$ are relation variables to be replaced with relations in KG and r participants in the head of their rule forms.
- *step 2*: complete partial axioms via randomly selecting k triples $(e', r, e'') \in \mathcal{T}$ related with r. Replacing r' or r'' in partial axioms with relations that directly link to e' or e''.
- *step 3*: search for support of each axiom, and add those axioms with number of support larger than 1 into axiom pool $\mathcal{P}$.

The key point of whole process is choosing k, which is not trivial because a large k will lead to dramatic increase of search space while a small k will decrease coverage of axioms immediately. Thus a good choice of k should achieve followings: 1) the probability for $\mathcal{P}$ covering all highly possible axioms higher than t, named *including probability*. 2) k is as small as possible. We defined the highly possible axioms as axioms with existing probability larger than p in this paper and we name p as *minimum axiom probability*. We estimate the probability $p(r, a_x)$ that r replaces the relation variable in *head* of axiom a_x as:

$$p(r, a_x) = \frac{n}{N} \tag{7}$$

where N is the number of triples $(e', r, e'') \in \mathcal{T}$ and n is the number of support for axiom a_x in $\mathcal{K}$. Thus choosing k triples to satisfy two requirements mentioned above can be formulated as follows:

$$\min k \; s.t. \; 1 - \frac{C^k_{(1-p)N}}{C^k_N} > t \tag{8}$$

From above inequality in (8), following inequality can be reached:

$$k > N - N(1-t)^{\frac{1}{pN}} \tag{9}$$

Thus with fixed p, t, the best choice of k should be the upper bound v_u of equation $f(N) = N - N(1-t)^{\frac{1}{pN}}$. Fortunately, with $N \in [0, 10^{15}]$, this equation is monotone increasing and has small upper bound v_u. For example, when $p = 0.5, t = 0.95$, v_u is 6.

As the input of axiom pool generation is a fixed $\mathcal{K}$ during learning, axiom pool $\mathcal{P}$ only need to be generated once.

3.2.2 Axiom Predicting. Given current relation embedding $\mathbf{R}$ and axiom pool $\mathcal{P}$, axiom predicting predicts a score s_a for each axiom $a \in \mathcal{P}$ based on the rule conclusion for each type of axioms (column 4 in Table 2), in the form that $\mathbf{M}^a_1 = \mathbf{M}^a_2$, where $\mathbf{M}^a_1$ and $\mathbf{M}^a_2$ are matrices either from a single matrix or a product of two matrices. As rule conclusions are derived from ideal linear map assumption, $\mathbf{M}^a_1$ and $\mathbf{M}^a_1$ usually are not equal but similar during training. Thus, instead of matching $\mathbf{M}^a_1$ and $\mathbf{M}^a_2$, we estimate the truth of axiom a by similarity between $\mathbf{M}^a_1$ and $\mathbf{M}^a_2$ which is supposed to be related with Frobenius norm of their difference matrix:

$$s_a(F) = \|\mathbf{M}^a_1 - \mathbf{M}^a_2\|_F \tag{10}$$

$s_a(F)$ is then normalized as follows because the value of $s_a(F)$ for different type of axioms vary dramatically:

$$s_a = \frac{s_{max}(t) - s_a(F)}{s_{max}(t) - s_{min}(t)} \tag{11}$$

in which t is the type of axiom that a belongs to. $s_{max}(t)$ and $s_{min}(t)$ is the maximum and minimum Frobenius norm score among all type t axioms in $\mathcal{P}$. $s_a \in [0, 1]$ is the final score for axiom a and the higher s_a is the more confident that axiom a is.

3.3 Axiom Injection

Given knowledge graph $\mathcal{K}$ and possible axiom set $\mathcal{A}$, axiom injection utilizes axiom' deductive capability to infer a set of new triples $\mathcal{T}_{axiom}$ for sparse entities and predict their labels. $\mathcal{T}_{axiom}$ will be injected into embedding learning to reduce the sparsity of entities.

3.3.1 Sparse entities. We evaluate the sparsity of entities as follows:

$$sparsity(e) = 1 - \frac{freq(e) - freq_{min}}{freq_{max} - freq_{min}} \tag{12}$$

where $freq(e)$ is the frequency of entity e participating in a triple, as subject or object entity. $freq_{min}$ and $freq_{max}$ are the minimum and maximum frequency among all entities. $sparsity(e) \in [0, 1]$. $sparsity(e) = 1$ means e is the most sparse entity and $sparsity(e) = 0$ means the most frequent one. With $sparsity(e) > \theta_{sparsity}$, we regard entity e as a sparse entity, where $\theta_{sparsity}$ is a sparse threshold. We use $\mathcal{E}_{sparse}$ to denote the set of sparse entity in $\mathcal{K}$.

During axiom injection, triples related with sparse entities are injected into the input of embedding learning. In other words, in new inferred triples (s^a, r^a, o^a), either $s^a \in \mathcal{E}_{sparse}$ or $o^a \in \mathcal{E}_{sparse}$ or $\{s^a, o^a\} \in \mathcal{E}_{sparse}$. Thus after inferring all possible new triples, we filter those unrelated to sparse entities.

3.3.2 Predicting new triples and their labels. We utilize groundings to infer new triples, and the grounding for axioms considered in this paper can be generalized as the following form:

$$(s^a, r^a, o^a) \leftarrow (s_1, r_1, o_1), (s_2, r_2, o_2), ..., (s_n, r_n, o_n) \tag{13}$$

where the right side triples $(s_k, r_k, o_k) \in \mathcal{T}$ with $k \in [1, n]$ are generated from the body of axiom rule form and $(s^a, r^a, o^a) \notin \mathcal{T}$ is new inferred triples to be added into knowledge graph.

To predict label for (s^a, r^a, o^a), we first translate the grounding form in (13) in propositional logical expression:

$$(s_1, r_1, o_1) \wedge (s_2, r_2, o_2) \wedge ... \wedge (s_n, r_n, o_n) \Rightarrow (s^a, r^a, o^a) \tag{14}$$

then we model groundings through t-norm based fuzzy logics[17]. It regards the truth value of a propositional logical expression as a composition of constituent triples' truth value, through specific logical connectives (e.g. $\wedge$ and $\Rightarrow$). For example, the truth value of a propositional logical expression $(s_1, r_1, o_1) \Rightarrow (s_2, r_2, o_2)$ is determined by the true value of two component triples (s_1, r_1, o_1) and (s_2, r_2, o_2), via a composition defined by logical implication $\Rightarrow$. We follow the definition of composition associated with logical conjunction ($\wedge$), disconjunction ($\vee$) and negation ($\neg$) as [14]:

$$\pi(a \wedge b) = \pi(a) \cdot \pi(b) \tag{15}$$

$$\pi(a \vee b) = \pi(a) + \pi(b) - \pi(a) \cdot \pi(b) \tag{16}$$

$$\pi(\neg a) = 1 - \pi(a) \tag{17}$$

$$\pi(a \Rightarrow b) = \pi(\neg a \vee b) \tag{18}$$

in which a, b are logical expressions and $\pi(x)$ is the truth value of x. Given these compositions, the truth value of any propositional logical expression can be calculated recursively through Equation (15)-(18). For example, applying to propositional logical expression of grounding $g: \; (s_1, r_1, o_1) \wedge (s_2, r_2, o_2) \Rightarrow (s^a, r^a, o^a)$:

$$\pi(g) = 1 - \pi(s_1, r_1, o_1)\pi(s_2, r_2, o_2) + \pi(s_1, r_1, o_1)\pi(s_2, r_2, o_2)\pi(s^a, r^a, o^a)$$

To predict the truth value $\pi(s^a, r^a, o^a)$ for triples (s^a, r^a, o^a), inferred by axiom a according to grounding g_a, based on t-norm fuzzy logics, $\pi(s^a, r^a, o^a)$ can be calculated according to $\pi(g)$ and $\pi(s_1, r_1, o_1), \pi(s_2, r_2, o_2), ..., \pi(s_n, r_n, o_n) \in g_a$. In former example,

$$\pi(s^a, r^a, o^a) = \frac{\pi(g) - 1 + \pi(s_1, r_1, o_1)\pi(s_2, r_2, o_2)}{\pi(s_1, r_1, o_1)\pi(s_2, r_2, o_2)}$$

For the truth value of axiom a's grounding g, we evaluate it as the score of a generated by axiom induction, namely $\pi(g_a) = s_a$. For the truth value of triples existing in knowledge graph, we evaluate them with their training labels, thus $\pi(s, r, o) = 1$ for $(s, r, o) \in \mathcal{T}$, because existing triples are absolutely true. With these two types of truth value assignment, we can easily get the following result for triples (s^a, r^a, o^a) inferred by any type of axioms a:

$$\pi(s^a, r^a, o^a) = s_a \tag{19}$$

After axiom injection, a set of new triples $\mathcal{T}_{axiom} = \{(s^a, r^a, o^a) | s^a \in \mathcal{E}_{sparse} \text{ or } o^a \in \mathcal{E}_{sparse}\}$ are inferred via quality axioms and each new triple (s^a, r^a, o^a) is labeled with $l_{sro} = s_a$. Thus the input of embedding learning was updated, $\mathcal{I} = \{((s, r, o), l_{sro}) | (s, r, o) \in \mathcal{T} \wedge \mathcal{T}_{axiom}\}$. Then the process goes back to embedding learning.

4 EXPERIMENT

During experiments, we want to explore following questions: 1) whether axioms really help sparse entity embedding learning? To do this, we evaluate the quality of embeddings on link prediction task which is widely applied in previous knowledge graph embedding works; 2) whether embeddings really help rule learning overcome the huge search space and improve the quality of learned rules? To do this, we evaluate the efficiency of axiom learning with learning time and the quality with number and percentage of high quality axioms the method generate; 3) how does the iterative manner affect embedding learning and rule learning process? To this end, we show the changes of link prediction results, rule qualities and the number of triples injected along with different number of iterations.

4.1 Dataset

Four datasets are used in experiment, including WN18-sparse, WN18RR-sparse, FB15k-sparse, and FB15k-237-sparse. They are generated from WN18[3], WN18RR[7], FB15k[3] and FB15-237[34], four datasets that are commonly used in previous knowledge graph embedding work. WN18 and WN18RR are subsets of WordNet, a lexical knowledge graph for English. FB15k and FB15k-237 are subsets of a large collaborative knowledge base Freebase. The statistics of datasets are listed in Table 3.

We generate the sparse version of these datasets via changing the valid and test datasets which will be used for link prediction evaluation. In link prediction experiments, we want to explore whether axioms really contribute to sparse entity embeddings. Therefore, we only keep triples in test and valid dataset which involve at least one sparsity entity. In other words, for a test or valid triple (s, r, o), if either s or o is or both are sparse entity (Equation (12)), the triple will be kept in sparse dataset, otherwise, it will be filtered.

Table 3: Statistics of datasets.

Dataset	#E	#R	#Train	#Valid	#Test
WN18-sparse	40,943	18	141,442	3624(72.48%)	3590(71.8%)
WN18RR-sparse	40943	11	86835	1609(53.03)	1661(52.9%)
FB15k-sparse	14,951	1,345	483,142	18544(37.08%)	22013(37.26%)
FB15k-237-sparse	14541	237	272115	10671(60.8%)	12454(60.85%)

When deciding sparse entity hyperparameter θ, intuitively a principle is considered: the percentage of left triples in valid and test set should not be larger than 80% or less than 20% of original valid and test set for all datasets. Thus we choose $\theta = 0.995$ and regard entities e with $sparsity(e) > 0.995$ as sparse entity in this paper. Table 3 also shows the percent of triples left.

4.2 Training Details

For embedding learning, the number of negative samples is set to 6 and the number of scalars on the diagonal of each relation matrix is set to $\frac{d}{2}$ where d is embedding dimension. We initialize embeddings with uniform distribution $\mathbf{U}(-0.1, 0.1)$.

For axiom induction, we set the minimum axiom probability $p = 0.5$ and the including probability $t = 0.95$ for axiom pool generation. Based on these settings, the number of related triples selected for each relation is set as $k = 6$ according to Equation(9).

Details of axiom pools generated for all datasets are shown in Table 4, where we can see that the number of possible axioms in FB15k and FB15k-237 are much larger than WN18 and WN18RR because the number of axioms is highly related to the diversity of relations.

Table 4: Axiom pools details.

	ref.	sym.	tra.	equ.	inv.	sub.	sub.(Cha.)
WN18-sparse	0	3	1	2	19	2	72
WN18RR-sparse	0	3	1	1	3	1	22
FB15k-sparse	41	94	52	1872	2743	1872	59709
FB15k-237-sparse	5	30	27	197	192	197	5017

For axiom injection, considering that axioms with high scores are more reliable and less possible to introduce noise, we set a threshold θ for each dataset and regard axioms with scores $s_{axiom} > \theta$ as high quality axioms. We also set a maximum inferred triples m for axioms in each dataset, if one axiom will infer a lot of triples we ignore it because a huge number of triples inferred by one axiom will change the data distribution significantly and make the embedding training unstable.

During one iteration learning, we first train embedding learning for 10 epochs, and then conduct axiom induction and injection once. The maximum training iteration is set to 10 for WN18RR-sparse and FB15k-237-sparse and 50 for WN18-sparse and FB15k-sparse. We use Adam algorithm [19] during optimization with learning rate $lr = 0.001$. We apply grid search for the best hyperparameters based on the filter MRR on the validation dataset, with combinations from embedding dimension $d \in \{100, 200\}$ and l_1 regularizer weigh $\lambda \in \{10^{-4}, 10^{-5}, 10^{-6}\}$.

The final parameters are $d = 200, \lambda = 10^{-5}, \theta = 0.95, m = 20000$ for WN18-sparse, $d = 200, \lambda = 10^{-5}, \theta = 0.9, m = 10000$ for WN18RR-sparse, $d = 200, \lambda = 10^{-4}, \theta = 0.95, m = 100$ are for FB15k-sparse, and $d = 200, \lambda = 10^{-5}, \theta = 0.9, m = 1000$ are for FB15k-237-sparse.

4.3 Embedding Evaluation

We evaluate the quality of embeddings on link prediction tasks. Link prediction aims to predict the missing entity when given the other entity and relation in a triple, including subject entity prediction $(?, r, o)$ and object entity prediction $(s, r, ?)$.

4.3.1 Evaluation metrics. Given subject entity prediction task $(?, r, o)$ with right answer s, we first fit subject entity position with each entity $e \in \mathcal{E}$ and thus get a set of triples $\mathcal{T}_{subjectprediction} = \{(e, r, o) | e \in \mathcal{E}\}$. Then we calculate the score for each triple in $\mathcal{T}_{subjectprediction}$ according to Equation (6) and rank their scores in descending order. Thus the entity e in (e, r, o) with a higher rank is a more possible prediction. To evaluate how good the prediction is, we use the rank of (s, r, o) among all triples in $\mathcal{T}_{subjectprediction}$ as subject entity prediction evaluation result for (s, r, o), namely subject entity prediction rank $rank_s(?, r, o)$. The object entity prediction task is done in the same way and will get objection entity rank $rank_o(s, r, ?)$. The final prediction rank $rank(s, r, o)$ for (s, r, o) is the average of subject and object prediction rank:

$$rank(s, r, o) = \frac{1}{2}(rank_s(?, r, o) + rank_o(s, r, ?))$$

Table 5: Link prediction results with MRR and Hit@n on WN18RR-sparse and FB15k-237-sparse. Underlined scores are the better ones between ANALOGY and IterE(ANALOGY). Boldface scores are the best results among all methods.

	WN18-sparse					FB15k-sparse				
	MRR (filter)	MRR (raw)	Hit@1 (filter)	Hit@3 (filter)	Hit10 (filter)	MRR (filter)	MRR (raw)	Hit@1 (filter)	Hit@3 (filter)	Hit10 (filter)
TransE[3]	41.8	33.5	10.2	71.1	84.7	39.8	25.5	25.8	48.6	64.5
DistMult[47]	73.8	55.8	59.3	87.5	93.1	60.0	32.4	**61.8**	65.1	75.9
ComplEx[35]	91.1	67.7	89.0	93.3	94.4	61.6	32.7	54.0	65.7	76.1
ANALOGY[23]	**91.3**	67.5	89.0	93.4	94.4	62.0	33.1	54.3	66.1	76.3
IterE(ANALOGY)	90.1	67.5	87.0	93.1	**94.8**	61.3	35.9	52.9	66.2	76.7
IterE(ANALOGY) + **axioms**	**91.3**	**78.9**	**89.1**	**93.5**	**94.8**	**62.8**	**38.8**	55.1	**67.3**	**77.1**
	WN18RR-sparse					**FB15k-237-sparse**				
	MRR (filter)	MRR (raw)	Hit@1 (filter)	Hit@3 (filter)	Hit10 (filter)	MRR (filter)	MRR (raw)	Hit@1 (filter)	Hit@3 (filter)	Hit10 (filter)
TransE[3]	14.6	12.4	3.4	24.7	28.8	23.8	15.6	16.4	26.1	38.5
DistMult[47]	25.5	20.8	23.8	26.0	22.5	20.4	12.9	12.8	22.6	36.2
ComplEx[35]	25.9	21.4	24.6	26.2	28.6	19.7	13.3	12.0	21.7	35.4
ANALOGY[23]	19.8	13.3	24.6	27.5	28.7	19.8	13.9	12.3	21.4	34.9
IterE(ANALOGY)	27.2	22.7	25.0	**28.1**	**31.4**	20.7	14.0	13.1	22.8	36.2
IterE(ANALOGY) + **axioms**	**27.4**	**25.7**	**25.4**	**28.1**	**31.4**	**24.7**	**18.6**	**17.9**	**26.2**	**39.2**

Table 6: Rule evaluation results.

	WN18-sparse			WN18RR-sparse			FB15k-sparse			FB15k-237-sparse		
	time	HQr	%	time	HQr	%	time	HQr	%	time	HQr	%
AMIE+	4.98s	16	11.4%	3s	2	5.71%	428s	1820	4.4%	66s	470	1.9%
IterE	**1.63**s	**20**	**20.2**%	**0.75**s	**6**	**19.3**%	**26.49**s	**11375**	**17.6**%	**4.72**s	**653**	**11.8**%

Aggregating prediction rank for all test triples, we applied mean reciprocal predicting rank *(MRR)* and the percentage of predicting ranks within n *(Hit@n)* to evaluate the whole prediction result. These two metrics are widely used in previous work [35][23]. Generally, a higher *MRR* or *Hit@n* indicates a better prediction result.

We also apply $filter$ and raw setting. As mentioned in [41], when fitting subject or object entity position with other entities, we may generate triples existing in knowledge graph which are known to be true. It is not wrong if these triples are ranked higher than current test triple. With $filter$ setting, we filter the generated triples that exist in both train/valid/test dataset before ranking but not current test triple. raw means the setting of NO filtering.

4.3.2 Baselines. For baselines, one method from translation-based assumption TransE [3] and three methods based on linear map assumption, DistMult, ComplEx, and ANALOGY [4] are considered.

[3]the code for TransE is from https://github.com/thunlp/OpenKE. We train all dataset with learning rate 0.01, margin 1.0 and dimension 100 for maximum 3500 iterations.
[4]The code for DistMult, ComplEx and ANALOGY is from https://github.com/quark0/ANALOGY. We train them with the same parameter setting as [23] for maximum 500 iterations. For WN18-sparse, the parameters are dimension $d = 200$, regularizer weigh $\lambda = 10^{-1}$, negative weight $n = 6$ for DistMult and $d = 200$, $\lambda = 10^{-2}$, $n = 6$ for both ComplEx and ANALOGY. For WN18RR-sparse, the parameters are $d = 200$, $\lambda = 1-^{-1}$, $n = 3$ for DistMult, Complex and ANALOGY. For FB15k-sparse, the parameters are $d = 200$, $\lambda = 10^{-2}$, $n = 6$ for DistMult, Complex and ANALOGY. For FB15k-237-sparse, the parameters are dimension $d = 100$, $\lambda = 0.1$, $n = 6$ for ANALOGY and Complex and $d = 200$, $\lambda = 10^{-1}$, $n = 3$ for DistMult.

4.3.3 Results and analysis. To show how axiom helps sparse entity embeddings, we adopt two strategies: (1) Firstly, we evaluate how axioms directly improve entity embedding quality and compare link prediction results from our method, denoted as **IterE** in Table 5, with other baselines directly. (2) Secondly, we evaluate how axioms can help improve sparse entity link prediction utilizing its deductive ability. Thus we compare prediction results with embeddings and axioms, denoted as **IterE+axioms** in Table 5 with IterE and other methods. In IterE+axioms, if the test triple (s, r, o) are inferred by axioms during axiom injection, which means $(s, r, o) \in \mathcal{T}_{axiom}$, we regard it as correct and mark its prediction rank as 1.

The link prediction results are shown in Table 5. We analyze the results as follows: Firstly, the link prediction results of IterE competitive to ANALOGY, which means most of the triples injected into embedding learning are not noise, indicating that learning axioms from embeddings works well. Secondly, IterE outperforms baselines on WN18RR-sparse and FB15k-237-sparse, while are slightly improved on WN18-sparse and FB15k-sparse. These indicate that IterE helps sparse entities much more in WN18RR-sparse and FB15k-237-sparse than in WN18RR-sparse and FB15k-sparse. This is quite reasonable in our opinion, because among these four datasets, WN18RR-sparse and FB15k-237-sparse are more challenging and more sparse. They are created from WN18-sparse and FB15k-sparse via removing one relation of all inverse relation pair in the dataset and also their related triples because inverse relation pair is a significant pattern in these two datasets as first noted in [34]. Thirdly, the results of

IterE+axioms are improved compared with IterE in all datasets, especially on the most complex and sparse dataset FB15k-237-sparse. It indicates that the deductive capability of axioms can help the prediction of sparse entities further.

From the evaluation of link prediction, we can conclude that: (1) by injecting new triples for sparse entities, axioms help improve the quality of sparse entity embeddings and are more helpful in sparse KGs. (2) Combining axioms and embeddings together to predict missing links works better than using embeddings only. Both the deductive capability of axioms and the inductive capability of embeddings contribute to prediction and complement each other.

4.4 Rule Evaluation

We evaluate the learned rules/axioms from two perspectives: efficiency and quality. We compare our method with AMIE+[9][5] which is an improved rule mining system of widely used AMIE[10].

4.4.1 Evaluation metrics. The efficiency of rule learning is evaluated by learning time. The quality of rule learning is evaluated with the number of high quality rule (HQr) and their percentage. The quality of rules are evaluated by head coverage(HC) which is commonly used in pervious work, such as [10] and [29]. Head coverage for rule *rul* is defined as follows:

$$HC(rul) = \frac{\#(e, e') : support(rul) \wedge head(rul)(e, e')}{\#(e, e') : head(rul)(e, e')}$$

in which $head(rul)(e, e') = \{(e, r, e') \in \mathcal{T}\}$ if the head atom of *rul* is (X, r, Y). And $support(rul)$ is the supports for *rul*. We regard high quality rules as the rules with $HC > 0.7$ during test.

4.4.2 Results and analysis. Rule evaluation results are shown in Table 6. According to the time used to generate rules among all datasets, we can see that IterE learns rules more efficiently. For example, with FB15k-sparse and FB15k-237-sparse datasets, IterE costs 10 times less than AMIE+. This shows our pruning strategy works well. Among 4.72 seconds IterE cost for FB15k-237-sparse dataset, there is 4.55 seconds used for axiom pool generation and 0.17 seconds for axiom score calculation, namely axiom score calculation only cost 3.6% of the time. This means calculating axiom scores via embeddings is super efficient. We didn't include the time of embedding learning during this evaluation, as embedding learning is not devised mainly for rule learning, but for link prediction. The number of high quality rules shows that IterE generates more high quality rules than AMIE+ for each dataset and also achieves a higher percentage. This indicates that our axiom pool generation can filter meaningless axioms(rules) and achieve a good balance between small search space and coverage of highly possible axioms.

Further more, Figure 4 shows the changes of high quality axioms coverage and axiom percentage with different axiom score thresholds from IterE. For example, in 4(c), with axiom threshold 0.9, which means we select axioms with $s_a > 0.9$ from IterE for axiom injection, there are 23.4% axioms selected among all axioms and 46.4% high quality axioms included. And in 4(d), with axiom threshold 0.9, there are 53.9% axioms selected and 76.5% high quality axioms included. It illustrates that axiom scores calculated from embeddings are reliable because it is consistent with rule evaluation

[5]we run AMIE+ code from https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/amie/

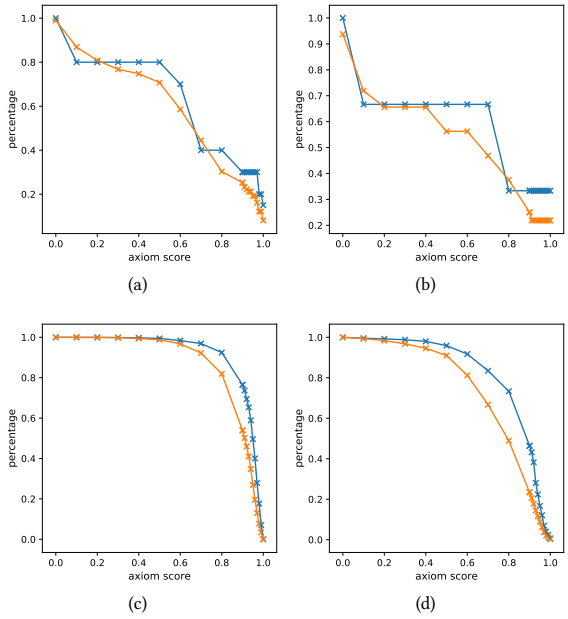


Figure 4: The coverage of high quality axioms (blue) and the axiom percentage (orange) with different axiom score thresholds. 4(a) is from WN18-sparse. 4(b) is from WN18RR-sparse. 4(c) is from FB15k-sparse. 4(d) is from FB15k-237-sparse.

results. The results in 4(a) and 4(b) for WN18-sparse and WN18RR-sparse are not obvious because a few relations are contained in these two datasets and the number of learned rules is limited.

From rule evaluation results, we can conclude that (1) embeddings together with axiom pool generation help rule learning overcome large search space problem and improve rule learning efficiency, and (2) they also improve rule learning qualities and rules' reliable scores generated based on calculation with embeddings.

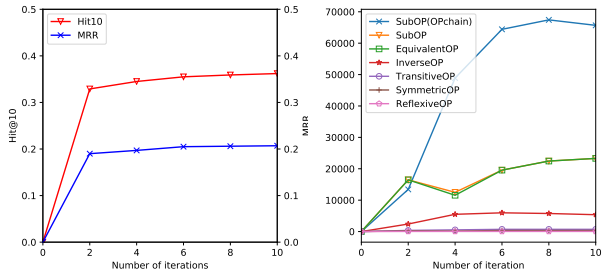
4.5 Iterative learning

To explore how iterative training improves embedding and rule learning during training, we show link prediction results on FB15k-237-sparse and number of injected triples at different iterations in Figure 5.

Figure 5 shows that the prediction results including Hit@10 and MRR become better as the training iteration increases and generally the number of injected triples increases during training and finally gets stable. The number of injected triples for `subOP` and `equivalentOP` in iteration 4 is less than iteration 3 which means axioms with high scores learned from previous iteration might get a lower score in the next iteration. This indicates that embedding learning effects rule learning. Thus from Figure 5, we can conclude that: (1) Iterative learning benefits embedding learning as the quality of embeddings gets better during training. (2) Iteratively learning benefits axiom learning because more axioms are learned and more triples are injected during training. (3) Axioms and embeddings influence and constrain each other during training.

Table 7: Examples from FB15k-237-sparse about the link prediction changes of test triples involving sparse entity. Following details are shown: the sparsity of subject and object entity in test triple, number within the square bracket beside the entity; the axiom that infers the test triple; the change of link prediction rank from ANALOGY to IterE; the change of link prediction rank from IterE to IterE+axiom.

triple	(Groundhog_Day[0.996]), /*film/currency*, United_States_Dollar[0.615])			
predicted by axiom	(X, /*film/currency*, Z) ← (X, *film_release_region*, Y), (Y, /*location/currency*, Z)			
rank change(ANALOGY → IterE)	**subject prediction rank:**	1409 → 356 (+1053)	**object prediction rank:**	1 → 1(+0)
rank change(IterE → IterE+axiom)	**subject prediction rank:**	356 → 1 (+355)	**object prediction rank:**	1 → 1(+0)
triple	(USA[0.0]), /*second_level_divisions*, Champaign_County[0.999])			
predicted by axiom	(X, /*second_level_divisions*, Z) ← (Y, /*bibs_location*, X), (Z, /*county_seat*, Y)			
rank change(ANALOGY → IterE)	**subject prediction rank:**	1 → 1 (+0)	**object prediction rank:**	416 → 214(+202)
rank change(IterE → IterE+axiom)	**subject prediction rank:**	1 → 1 (+0)	**object prediction rank:**	214 → 1(+213)
triple	(Jenny_McCarthy[0.997]), /*type_of_union*, Marriage[0.607])			
predicted by axiom	(X, /*type_of_union*, Z) ← (X, *spouse*, Y), (Y, /*type_of_union*, Z)			
rank change(ANALOGY → IterE)	**subject prediction rank:**	4199 → 33 (+4166)	**object prediction rank:**	2 → 2(+0)
rank change(IterE → IterE+axiom)	**subject prediction rank:**	33 → 1 (+32)	**object prediction rank:**	2 → 1(+1)



(a) (b)

Figure 5: 5(a) shows link prediction results, MRR and Hit@10, in different iterations. 5(b) shows the number of triples injected into embedding learning in different iterations.

4.6 Case study

In Table 7, we give a case study with 3 test triple examples which are inferred by axioms during training. Using the third one as an example, the test triple is (Jenny_McCarthy, /*type_of_union*, Marriage). Jenny_McCarthy is a person with only a few triples in KG and is a sparse entity with sparsity 0.997. Marriage is a type of union for people which has many links with individual person and its sparsity is 0.607. The subject prediction rank (?, /type_of_union, Marriage) in ANALOGY is 4199 and the object prediction (Jenny_McCarthy, /*type_of_union*, ?) is 2, from which we can see that the prediction for a sparse entity is much worse than a frequent entity. In IterE, this triple is predicted by rule (X, /*type_of_union*, Z) ← (X, *spouse*, Y), (Y, /*type_of_union*, Z) because there are (Jenny_McCarthy, *spouse*, Jim_Carrey) and (Jim_Carrey, /*type_of_union*, Marriage) in KG which can compose a grounding for previous rules. Thus during training, the test triple is inferred by IterE and injected into embedding learning. Thus the subject prediction is improved from 4199 to 33 compared with ANALOGY and the object prediction keeps the same. When we predict with both embeddings and axioms (IterE + axioms), the subject prediction is improved from 33 to 1 and the object prediction from 2 to 1 compared with IterE. Other two cases in Table 7 perform similar to the third one.

These examples show that: (1) adding triples that can be inferred by axioms back into training improves the prediction results of related sparse entities without hurting the results for non-sparse entities. (2) the deductive ability of axioms helps ensure truth value of triples with sparse entities, which, in our opinion, overcomes uncertainties and noises that effect embedding prediction.

5 RELATED WORK

In this paper, we focus on iteratively learning embeddings and rules from knowledge graphs. Thus the related work includes two parts: (1) embedding learning, (2) rule learning.

5.1 Embedding Learning

Knowledge graph embedding learns latent representations for entities and relations in continuous vector space. Normally entities are represented as vectors [3][27][35] and relations are represented as vectors [3][41][27] or matrices [47][28][23]. These embeddings are assumed to preserve the semantics in a knowledge graph such as the similarity between entities[30], the truth of triples[3]. For most embedding methods, the input are triples existing in knowledge graph and embeddings are trained based on different vector space assumptions, such as translation-based assumption in [3] [41] [22] [18], and linear map assumption in [28] [47] [35] [23]. Some methods do not follow specific vector space assumption for embeddings, and special neural networks are applied to train embeddings and assign reasonable scores for different triples. For example, neural tensor networks in [32], convolution neural network in [7] and shared memory neural network in [31]. Linear map assumption is adopted for embedding learning in this paper because of its simplicity and good property for rule learning, while other assumptions or neural network models don't have.

Besides triples in KGs, some embedding methods also utilize other information during learning, for examples, entity descriptions in [40][43][39][42], entity types in [12][45], entity images in [44], and paths in [11][21][25], which can be regarded as a kind of rules.

Rules, as useful information for reasoning, are also considered to assist embedding learning. Among these methods different types of rules are considered, for examples, [37] introduces one type of logical rule and three physical rules, [14] utilizes horn soft rules learned

from [9], [24] incorporates equivalence and inversion Axioms, and [13] considers two types of logical rules. To incorporate rules into embedding learning, some methods try to improve the embedding method's capability of modeling rules by various ways, such as encoding relations as convex regions in [15] and non-negativity constraints on entity representations in [8]. Some methods regard rules as guidance for embedding learning. For examples, [37] forms the whole learning process as a integer linear programming problem with conditions defined based on rules. [14] infers unlabeled triples according to input rules. [21] adds constrains for relation embeddings participating in path rules. [6] maps entity-tuple embedding into an approximately Boolean space and encourages a partial ordering over relation embedding based on implication rules. [8] adds approximate entailment constraints on relation representations. Some methods jointly embed rules and triples during learning, such as [13] which first learns rules based on embeddings from [3] and then retrains the embeddings incorporating with rules.

Aforementioned embedding methods with rules normally make rule learning detached from embedding learning and unrelated with embeddings. While our goal in this paper is not only an embedding learning system, but also a rule learning system in which rules are learned from embeddings and help improve embedding quality.

5.2 Rule learning

Rule is an important component for reasoning and has been studied in many previous works. Among rule learning works, different kinds of rules are adopted, for example, Horn rules in [9], closed path rules in [20] and general rules with both variables and constants and atoms occurring either positively or negatively in [16], frequent predicate cycles in [38], semantic association rules in [1]. These methods all consider the types of rules to learn from their structures, while in this paper, we consider rules based on their semantics because we think semantics are important for the development of knowledge graphs. Thus we adopt OWL2 object property expression axioms to form the types of rules to learn.

To calculate the confidence of rules, standard confidence or PCA confidence are usually used [9][16] which are based on searching for supports of rules in the whole knowledge graph. To reduce the search space, many rule learning systems have a pruning strategy. Apart from graph search, there are also some works trying to learn both structures and scores of rules based on deep neural models [48] or reinforcement learning [46][5].

Embeddings are also used to help rule learning and are used for different purposes. Some utilize embeddings to guide and prune the search for candidate rules[29]. Some make embeddings to complete the knowledge graph during rule learning[16]. Some use embeddings to assign scores for rules[47][13]. They all rely on calculations among embeddings and the way of calculations depends on specific embedding methods.

Different from these methods which mainly proposed to learning rules, we devote to learn embeddings and rules at the same time and make their advantages contribute to each other's learning.

6 CONCLUSION AND FUTURE WORK

In this paper, we discuss the advantages and disadvantages of two common knowledge graph reasoning methods, embedding-based method and rule-based method, and propose IterE that iteratively learns embeddings and rules in one model and enjoys the mutual benefit between them. In the future, we will continuously investigate combining inductive and deductive reasoning together and develop models that could unify different kinds of reasoning.

ACKNOWLEDGMENTS

This work is funded by NSFC91846204/61473260, national key research program YS2018YFB140004, Alibaba CangJingGe(Knowledge Engine) Research Plan and SNF Sino Swiss Science and Technology Cooperation Programme program under contract RiC 01-032014.

REFERENCES

[1] Molood Barati, Quan Bai, and Qing Liu. 2016. SWARM: An Approach for Mining Semantic Association Rules from Semantic Web Data. In *PRICAI (Lecture Notes in Computer Science)*, Vol. 9810. Springer, 30–43.
[2] Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. *Proceedings of SIGMOD* (2008), 1247–1250.
[3] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. *Proceedings of NIPS* (2013), 2787–2795.
[4] Wanyun Cui, Yanghua Xiao, Haixun Wang, Yangqiu Song, Seung-won Hwang, and Wei Wang. 2017. KBQA: Learning Question Answering over QA Corpora and Knowledge Bases. *PVLDB* 10, 5 (2017), 565–576.
[5] Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alexander J. Smola, and Andrew McCallum. 2017. Go for a Walk and Arrive at the Answer: Reasoning Over Paths in Knowledge Bases using Reinforcement Learning. *CoRR* abs/1711.05851 (2017).
[6] Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. 2016. Lifted Rule Injection for Relation Embeddings. In *EMNLP*. The Association for Computational Linguistics, 1389–1399.
[7] Tim Dettmers. 2018. Convolutional 2D Knowledge Graph Embeddings. *Proceedings of AAAI* (2018).
[8] Boyang Ding, Quan Wang, Bin Wang, and Li Guo. 2018. Improving Knowledge Graph Embedding Using Simple Constraints. In *ACL (1)*. Association for Computational Linguistics, 110–121.
[9] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2015. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB J.* 24, 6 (2015), 707–730.
[10] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*. 413–422.
[11] Alberto García-Durán, Antoine Bordes, and Nicolas Usunier. 2015. Composing Relationships with Translations. *Proceedings of EMNLP* (2015), 286–290.
[12] Shu Guo, Quan Wang, Bin Wang, Lihong Wang, and Li Guo. 2015. Semantically Smooth Knowledge Graph Embedding. *Proceedings of ACL* (2015), 84–94.
[13] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2016. Jointly Embedding Knowledge Graphs and Logical Rules. In *EMNLP*. The Association for Computational Linguistics, 192–202.
[14] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2018. Knowledge Graph Embedding With Iterative Guidance From Soft Rules. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.
[15] Víctor Gutiérrez-Basulto and Steven Schockaert. 2018. From Knowledge Graph Embedding to Ontology Embedding? An Analysis of the Compatibility between Vector Space Representations and Rules. In *KR*. AAAI Press, 379–388.
[16] Vinh Thinh Ho, Daria Stepanova, Mohamed H. Gad-Elrab, Evgeny Kharlamov, and Gerhard Weikum. 2018. Rule Learning from Knowledge Graphs Guided by Embedding Models. In *International Semantic Web Conference (1) (Lecture Notes in Computer Science)*, Vol. 11136. Springer, 72–90.
[17] Petr HÃajek. 1998. The metamathematics of fuzzy logic. (1998).
[18] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding via Dynamic Mapping Matrix. *Proceedings of ACL* (2015), 687–696.
[19] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).
[20] Ni Lao, Tom M. Mitchell, and William W. Cohen. 2011. Random Walk Inference and Learning in A Large Scale Knowledge Base. In *EMNLP*. ACL, 529–539.
[21] Yankai Lin, Zhiyuan Liu, Huan-Bo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling Relation Paths for Representation Learning of Knowledge Bases. *Proceedings of EMNLP* (2015), 705–714.

[22] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. *Proceedings of AAAI* (2015), 2181–2187.

[23] Hanxiao Liu, Yuexin Wu, and Yiming Yang. 2017. Analogical Inference for Multi-relational Embeddings. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. 2168–2178.

[24] Pasquale Minervini, Luca Costabello, Emir Muñoz, Vít Novácek, and Pierre-Yves Vandenbussche. 2017. Regularizing Knowledge Graph Embeddings via Equivalence and Inversion Axioms. In *ECML/PKDD (1) (Lecture Notes in Computer Science)*, Vol. 10534. Springer, 668–683.

[25] Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional Vector Space Models for Knowledge Base Completion. *Proceedings of ACL* (2015), 156–166.

[26] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016. A Review of Relational Machine Learning for Knowledge Graphs. *Proc. IEEE* 104, 1 (2016), 11–33.

[27] Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. 2016. Holographic Embeddings of Knowledge Graphs. *Proceedings of AAAI* (2016), 1955–1961.

[28] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. *Proceedings of ICML* (2011), 809–816.

[29] Pouya Ghiasnezhad Omran, Kewen Wang, and Zhe Wang. 2018. Scalable Rule Learning via Learning Representation. In *IJCAI*. ijcai.org, 2149–2155.

[30] Petar Ristoski and Heiko Paulheim. 2016. RDF2Vec: RDF Graph Embeddings for Data Mining. *Procedings of ISWC* (2016), 498–514.

[31] Yelong Shen, Po-Sen Huang, Ming-Wei Chang, and Jianfeng Gao. 2017. Modeling Large-Scale Structured Relationships with Shared Memory for Knowledge Base Completion. In *Rep4NLP@ACL*. Association for Computational Linguistics, 57–68.

[32] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. *Prodeddings of NIPS* (2013), 926–934.

[33] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. *Proceedings of WWW* (2007), 697–706.

[34] Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*. 57–66.

[35] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. *Proceedings ICML* (2016), 2071–2080.

[36] Guanying Wang, Wen Zhang, Ruoxu Wang, Yalin Zhou, Xi Chen, Wei Wang, Hai Zhu, and Huajun Chen. 2018. Label-Free Distant Supervision for Relation Extraction via Knowledge Graph Embedding. In *EMNLP*. The Association for Computational Linguistics.

[37] Quan Wang, Bin Wang, and Li Guo. 2015. Knowledge Base Completion Using Embeddings and Rules. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*. 1859–1866.

[38] Zhichun Wang and Juan-Zi Li. 2015. RDF2Rules: Learning Rules from RDF Knowledge Bases by Mining Frequent Predicate Cycles. *CoRR* abs/1512.07734 (2015).

[39] Zhigang Wang and Juan-Zi Li. 2016. Text-Enhanced Representation Learning for Knowledge Graph. *Proceedings of IJCAI* (2016), 1293–1299.

[40] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph and Text Jointly Embedding. *Proceedings of EMNLP* (2014), 1591–1601.

[41] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. *Proceedings of AAAI* (2014), 1112–1119.

[42] Han Xiao, Minlie Huang, Lian Meng, and Xiaoyan Zhu. 2017. SSP: Semantic Space Projection for Knowledge Graph Embedding with Text Descriptions. *Proceedings of AAAI* (2017), 3104–3110.

[43] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation Learning of Knowledge Graphs with Entity Descriptions. *Proceedings of AAAI* (2016), 2659–2665.

[44] Ruobing Xie, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2017. Image-embodied Knowledge Representation Learning. In *IJCAI*. ijcai.org, 3140–3146.

[45] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2016. Representation Learning of Knowledge Graphs with Hierarchical Types. *Proceedings of IJCAI* (2016), 2965–2971.

[46] Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *EMNLP*. Association for Computational Linguistics, 564–573.

[47] Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. *Proceedings of ICLR* (2015).

[48] Fan Yang, Zhilin Yang, and William W. Cohen. 2017. Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In *NIPS*. 2316–2325.

[49] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. In *KDD*. ACM, 353–362.

[50] Wen Zhang. 2017. Knowledge Graph Embedding with Diversity of Structures. *Proceedings WWW Companion* (2017), 747–753.

Language in Our Time: An Empirical Analysis of Hashtags

Yang Zhang
CISPA Helmholtz Center for Information Security
Saarland Informatics Campus
yang.zhang@cispa.saarland

ABSTRACT

Hashtags in online social networks have gained tremendous popularity during the past five years. The resulting large quantity of data has provided a new lens into modern society. Previously, researchers mainly rely on data collected from Twitter to study either a certain type of hashtags or a certain property of hashtags. In this paper, we perform the first large-scale empirical analysis of hashtags shared on Instagram, the major platform for hashtag-sharing. We study hashtags from three different dimensions including the temporal-spatial dimension, the semantic dimension, and the social dimension. Extensive experiments performed on three large-scale datasets with more than 7 million hashtags in total provide a series of interesting observations. First, we show that the temporal patterns of hashtags can be categorized into four different clusters, and people tend to share fewer hashtags at certain places and more hashtags at others. Second, we observe that a non-negligible proportion of hashtags exhibit large semantic displacement. We demonstrate hashtags that are more uniformly shared among users, as quantified by the proposed hashtag entropy, are less prone to semantic displacement. In the end, we propose a bipartite graph embedding model to summarize users' hashtag profiles, and rely on these profiles to perform friendship prediction. Evaluation results show that our approach achieves an effective prediction with AUC (area under the ROC curve) above 0.8 which demonstrates the strong social signals possessed in hashtags.

CCS CONCEPTS

• **Information systems** → **Data mining**; • **Human-centered computing** → **Social networks**; **Social tagging**; **Social media**; **Empirical studies in collaborative and social computing**.

KEYWORDS

Hashtag, online social networks, data analysis

ACM Reference Format:
Yang Zhang. 2019. Language in Our Time: An Empirical Analysis of Hashtags. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313480

1 INTRODUCTION

The last decade has witnessed the explosive development of online social networks (OSNs). Leading players in the business, such as Facebook,[1] Twitter,[2] and Instagram,[3] have become the major platform for people to share life moments, communicate with each other, and maintain social relations. Moreover, OSNs have introduced many new notions into our society, such as "like", "share", and "check-in". One particular interesting notion of this kind is *hashtag*.

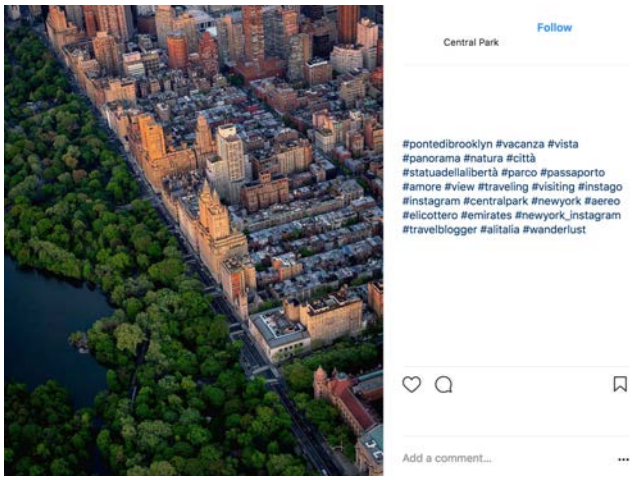


Figure 1: An Instagram post with multiple hashtags.

Created back in 2007, hashtags are designed to help users efficiently retrieve information on Twitter. With development, Instagram has become the major platform for hashtag-sharing. Nowadays, it is very common to see an Instagram post associated with multiple hashtags (see Figure 1 for an example). People also start to use hashtags for various purposes. For instance, many brands use hashtags to promote their products, such as #mycalvins and #shareacoke. Also, hashtags play a major role in various political movements, e.g., #blacklivesmatter and #notmypresident. More interestingly, hashtags have evolved themselves into a new-era language: People have created many hashtags the meanings of which do not exist in the natural language. For instance, #nomakeup attached to a photo indicates that the person in the photo did not wear any makeup; a user publishing #follow4follow means she will follow back others who follow her in OSNs. In another example, #tbt (throwback Thursday) indicates the corresponding photo was taken from old days.

The large quantity of hashtags has provided us with a new way to understand the modern society. Previously, researchers have studied hashtags from various angles [2, 12, 34, 37, 41, 42, 55, 58, 60, 65]. For

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313480

[1] https://www.facebook.com/
[2] https://twitter.com/
[3] https://www.instagram.com/

instance, Souza et al. have analyzed #selfie to understand the online self-portrait convention [58]. Mejova et al. use #foodporn to study people's dining preference on a global scale [37]. More recently, Zhang et al. investigate the (location) privacy risks stemming from sharing hashtags [65].

Most of these previous works have studied either a certain type of hashtags [34, 37, 41, 42, 58] or a certain property of hashtags [2, 55, 65]. Meanwhile, several general analyses concentrate on hashtags shared on Twitter [12, 60] which is a very different OSN from Instagram with respect to user group, popularity, and functionality [21, 35].

In this paper, we perform the first large-scale empirical study aiming at understanding hashtags shared on Instagram. Our analyses are centered around three research questions summarized from three different dimensions, i.e., the temporal-spatial dimension, the semantic dimension, and the social dimension.

1.1 Research Questions

Different hashtags exhibit different temporal patterns. Holiday-related hashtags, such as #newyear, may have periodic popularity, while the usage of some other hashtags may increase steadily over time. Besides, the information from the spatial dimension may also influence users' hashtag-sharing behavior: People are more (less) willing to share hashtags when they are at certain places. Therefore, we ask our first research question:

RQ1. *What are the temporal and spatial patterns of hashtags?*

As a new-era language, the semantics of hashtags can uncover many underlying patterns of the modern-style communication. Moreover, due to their inherent dynamic nature, some hashtags may change their meanings within a short time. We therefore ask:

RQ2. *Do hashtags exhibit semantic displacement?*

Following social homophily theory, we hypothesize that friends exhibit more similar hashtag-sharing behavior than strangers. In another way, hashtags possess strong signals for inferring users' social relations. To test this hypothesis, we ask:

RQ3. *Can hashtags be used to infer social relations?*

1.2 Contribution

We perform the first large-scale empirical analysis of hashtags shared on Instagram. We first sample in total 51,527 Instagram users from three major cities in the English-speaking world, including New York, Los Angeles, and London. We then collect all the posts that these users have shared from the end of 2010 to the end of 2015, and build three separate datasets. In total, our datasets contain more than 41 million Instagram posts shared together with 7 million hashtags.

To address **RQ1**, we first perform clustering to summarize hashtags' temporal patterns which results in four different clusters. In particular, one cluster of hashtags exhibits strong periodic popularity, some examples in this cluster are #snow, #bbq, and #superbowl. For the spatial dimension, we show that people in all the three datasets tend to share fewer hashtags at certain types of places, such as bars, while more at other types of places, e.g., parks.

For **RQ2**, we first adopt the skip-gram model [38, 39] to map hashtags to continuous vectors and demonstrate that these vectors can very well represent hashtags' semantics. Relying on the orthogonal Procrustes approach, we measure a hashtag's semantic displacement between two consecutive years as the distance between its two vectors trained at those years. Evaluation shows that a non-negligible proportion (more than 10%) of hashtags indeed shift their meanings to a large extent. We further define a notion, namely hashtag entropy, to quantify the uniformity of hashtags being shared among users, and observe that hashtags with low entropy are prone to semantic displacement: Correlation coefficients between semantic displacement and entropy in all the datasets are below -0.6.

To answer **RQ3**, we perform a friendship prediction task solely based on hashtags. We propose a bipartite graph embedding approach to learn each user's hashtag profile and conduct unsupervised friendship prediction based on two users' profiles' cosine distance. Extensive experiments show that our approach achieves an effective prediction with AUC (area under the ROC curve) above 0.8 in all the three datasets, and outperforms several baseline models by 20%. This indicates that hashtags indeed possess strong signals on social relations.

To the best of our knowledge, no previous works have studied hashtags' spatial patterns, semantic displacement, and social signals. We are the first to analyze hashtags from these angles.

We believe our analysis can benefit several parties. The conclusions drawn from answering **RQ1** and **RQ2** can help media campaigns to design more attractive hashtags to engage new customers. The semantic displacement result (from **RQ2**) can help researchers gain a deeper understanding of the OSN culture. The friendship prediction algorithm derived from answering **RQ3** shows that hashtags can also be used as a strong signal for friendship recommendation, which is essential for OSN operators.

1.3 Organization

The rest of the paper is organized as the following. We describe our dataset collection methodology with some initial analyses in Section 2. In Section 3, we investigate the temporal and spatial patterns of hashtags. Section 4 studies the semantic displacement of hashtags. In Section 5, we concentrate on using hashtags to infer social relations. Section 6 discusses the related work in the field. We conclude the paper in Section 7.

2 DATASETS AND INITIAL ANALYSES

In this section, we first describe our data collection methodology, then perform some initial analyses on hashtags.

2.1 Datasets Collection

We resort to Instagram to collect our datasets for experiments. Launched in October 2010, Instagram is a social network service concentrating on photo sharing. By now, it is the second most popular OSN with more than 1 billion monthly active users.[4] Instagram is the major social network for hashtag-sharing (Figure 1),

[4] https://instagram-press.com/our-story/

Table 1: Summary statistics of all the three datasets.

	New York	Los Angeles	London
No. posts	20,673,946	11,907,967	8,640,637
No. hashtags	4,095,575	3,071,158	1,702,675
No. check-ins	1,609,062	883,862	586,420
No. users	25,735	14,687	11,105
No. social links	82,964	36,434	12,900

many popular hashtags are strongly related to Instagram itself, e.g., #instagood and #instamood.

We collect our dataset relying on Instagram's public API on April, 2016. The first step is finding a sample of users [40]. In the literature, there exist multiple methods for this task. One is relying on the OSN's streaming API [46] which Instagram does not provide. Another way is generating random integers and query the Instagram API to see whether these numbers are valid user IDs [58]. However, this approach needs us to further investigate how Instagram users' IDs are distributed. In this paper, we instead sample users based on their locations following previous works [61, 65].

We concentrate on three major English-speaking cities including New York, Los Angeles, and London. In the first step, we query the API of Foursquare,[5] a location-based social network, to find all the Foursquare location IDs in the three cities. Then, we use Instagram's API to map all the Foursquare location IDs to the corresponding Instagram location IDs.[6] Next, we query all the obtained Instagram location IDs to get all users who have ever shared posts at those locations. As in previous works, we further perform some preprocessing to filter out the users matching any of the following criteria [3, 7, 44, 45, 57].

- users with less than 20 check-ins in each city
- users whose numbers of followers are above the 90th percentile (celebrities) or below the 10th percentile (bots)
- users not using human images in their profile photos[7]

In total, we obtain 51,527 Instagram user IDs. Then, we collect all these users' Instagram posts from the creation time of their accounts until December 31st, 2015. Each post is organized in the following format.

$$\langle \text{user ID, time, hashtags, location ID} \rangle$$

Note that a post is not necessarily associated with a set of hashtags or a certain location ID. We further query Instagram's API to extract users' social relationships. We consider two users to be friends if they follow each other [3, 7, 11].

We treat datasets collected from users in the three cities separately to ensure the robustness of our analyses. In total, the New York dataset (dataset collected from users sampled in New York) contains more than 20 million Instagram posts, the Los Angeles dataset contains 11 million posts, and the London dataset contains 8.6 million posts. Moreover, the three datasets contain more than 7 million hashtags. Table 1 presents some summary statistics of the datasets.

[5]https://foursquare.com/

[6]Instagram's API was connected with Foursquare's API until April 20th, 2016 (https://www.instagram.com/developer/changelog/).

[7]This is done with the help of Face++'s API.

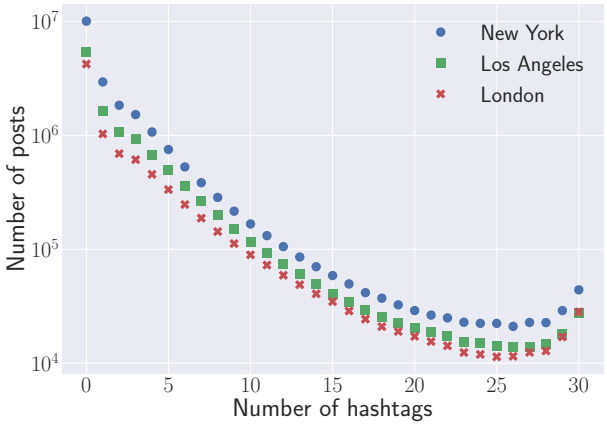


Figure 2: Distribution of the number of hashtags in each post in three datasets. The y-axis is in log scale. The proportions of posts with no hashtags are 48.69% in the New York dataset, 45.28% in the Los Angeles dataset, and 48.92% in the London dataset.

Table 2: Top 10 hashtags with the highest share times in three datasets.

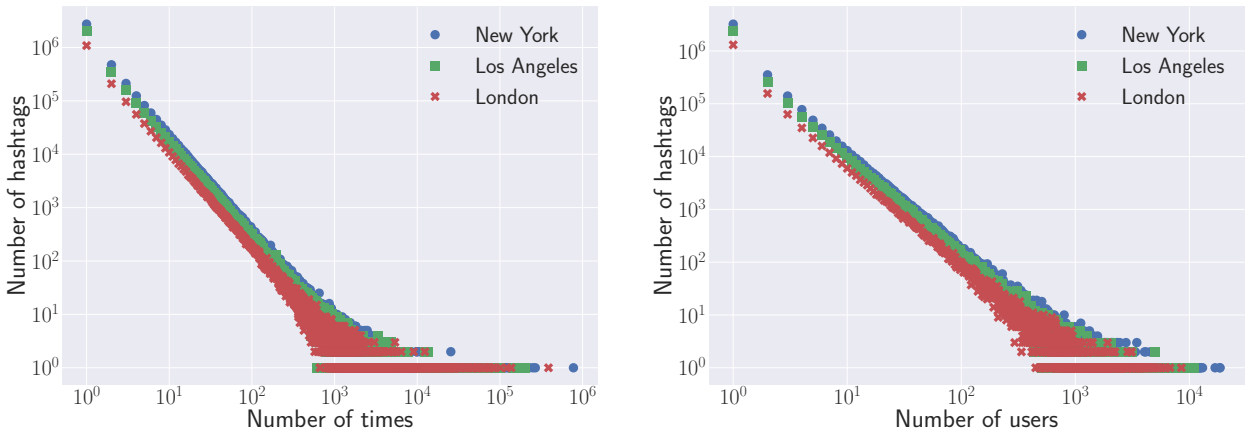
New York	Los Angeles	London
#nyc	#love	#london
#love	#losangeles	#love
#tbt	#tbt	#instagood
#nofilter	#california	#travel
#brooklyn	#instagood	#summer
#latergram	#nofilter	#photooftheday
#instagood	#foodporn	#food
#art	#family	#art
#travel	#art	#instadaily
#summer	#fun	#architecture

Ethical Considerations. Our data collection is done through Instagram's public API in 2016. All the datasets are stored in a central server with restricted access. We further anonymize the datasets by removing all users' screen names, and replacing their Instagram IDs with randomly generated numbers. Our experiments are conducted on these anonymized datasets.

2.2 Initial Analyses

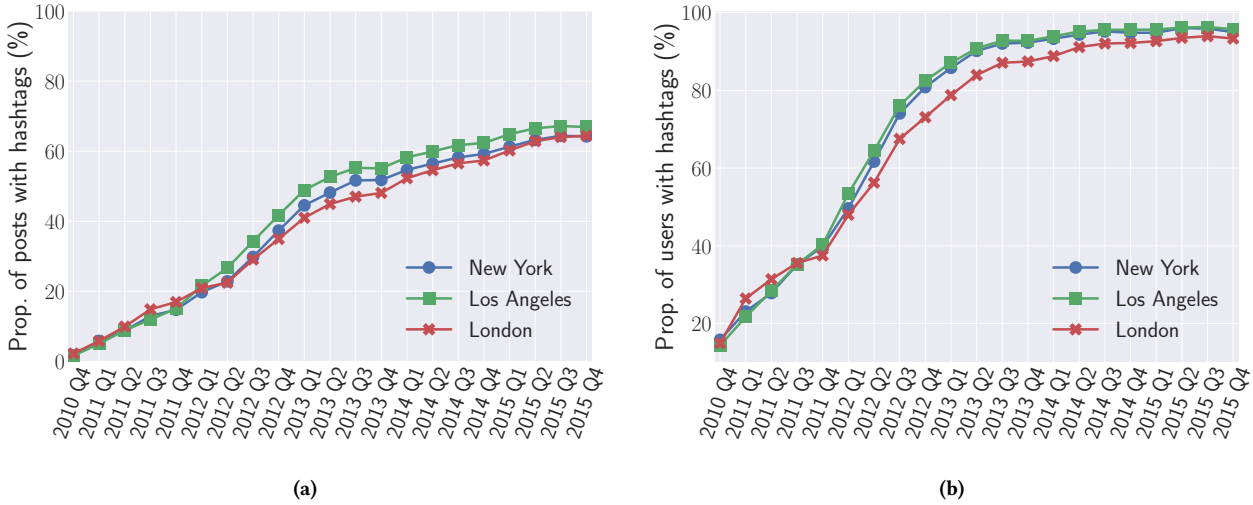
Figure 2 depicts the distribution of the number of hashtags in each post. We observe that more than half of the posts are associated with at least one hashtag in all the datasets. Moreover, there are around 30% posts associated with 1 to 3 hashtags. The small increase close to 30 is due to the fact that Instagram imposes an upper bound of 30 hashtags per post.[8]

[8]https://help.instagram.com/351460621611097



(a) (b)

Figure 3: (a) Distribution of the number of times a hashtag is shared in three datasets; (b) Distribution of the number of users a hashtag is used by in three datasets. Both x and y axes are in log scale.



(a) (b)

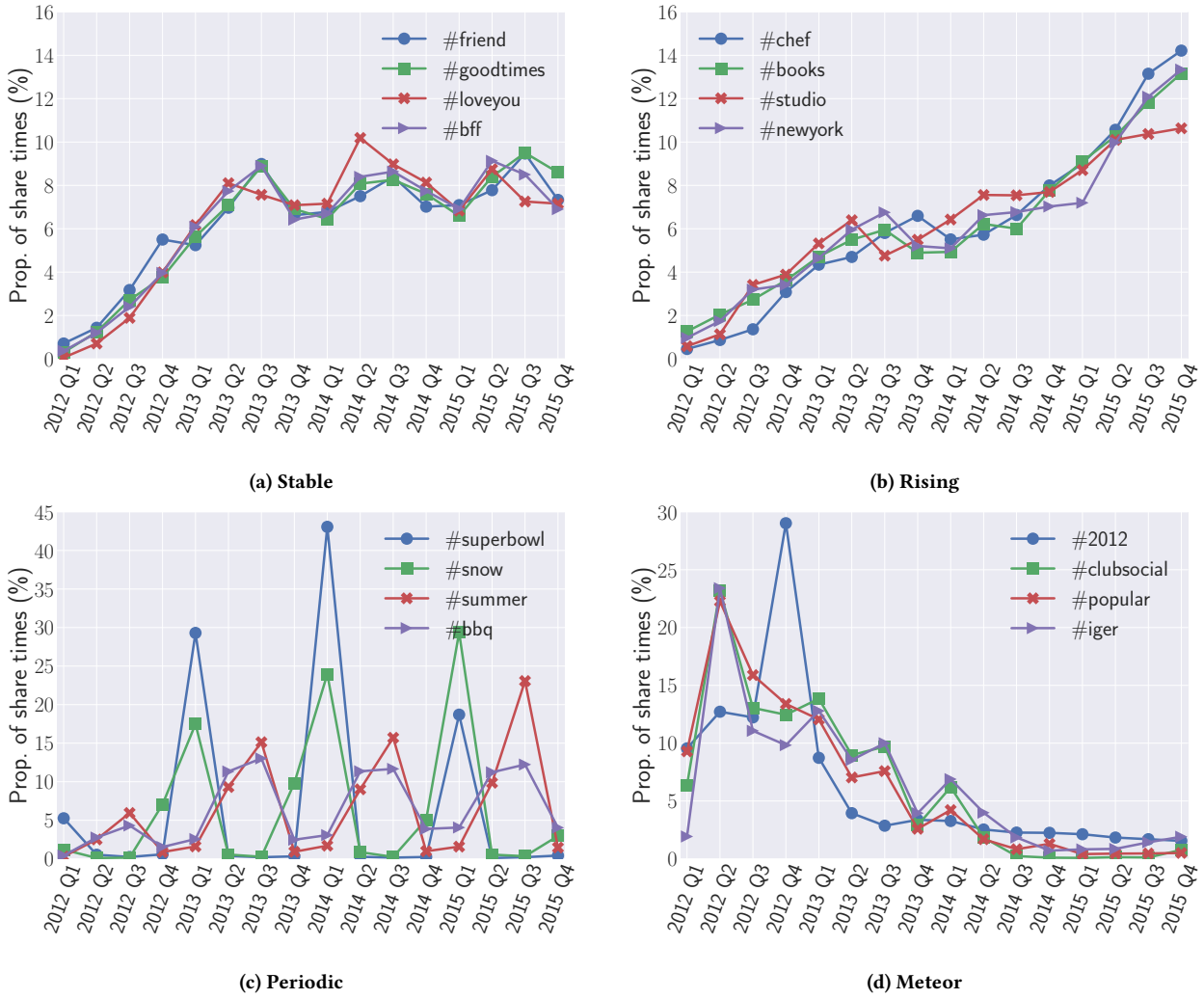
Figure 4: (a) Proportion of posts that are associated with hashtags from 2010 Q4 to 2015 Q4 in three datasets; (b) Proportion of users that use hashtags from 2010 Q4 to 2015 Q4 in three datasets. Q1 represents the first quarter of a year.

Figure 3a plots the distribution of the number of times each hashtag is shared, referred to as share times, while Figure 3b plots the distribution of the number of users each hashtag is used by. As expected, both distributions follow power law, i.e., most of the hashtags are shared only a few times and by a small number of users. Table 2 lists the hashtags with the highest share times. As our users are sampled by cities, many of these popular hashtags are related to city names, such as #nyc, #losangeles, and #london. However, general popular hashtags are captured as well, e.g., #love, #tbt, and #instagood. This indicates that our datasets are suitable for conducting the study.

Figure 4 depicts hashtags' general temporal patterns from 2010 until 2015. We see that in the fourth quarter of 2010 (2010 Q4) when Instagram was launched, there are less than 2% of the posts associated with hashtags, after 5 years (2015 Q4), the number becomes almost 70% (Figure 4a). Within the same time period, the proportion of users that ever use hashtags grow from 18% to 97% (Figure 4b). These results fully demonstrate the popularity of hashtags.

3 HASHTAGS IN THE TEMPORAL-SPATIAL DIMENSION

This section concentrates on our first research question: *What are the temporal and spatial patterns of hashtags?*. We start by investigating the temporal patterns of hashtags, then discuss the relation between hashtags and locations.



(a) Stable **(b) Rising**

(c) Periodic **(d) Meteor**

Figure 5: Temporal patterns of some example hashtags belonging to different temporal clusters in the New York dataset.

3.1 Temporal Patterns

We hypothesize that different hashtags exhibit different temporal patterns. Some hashtags should have periodic patterns, such as holiday-related ones, while others' share times may increase over time. Besides, there may exhibit other temporal patterns for hashtags. To perform a quantitative study, we resort to machine learning clustering to discover different temporal patterns.

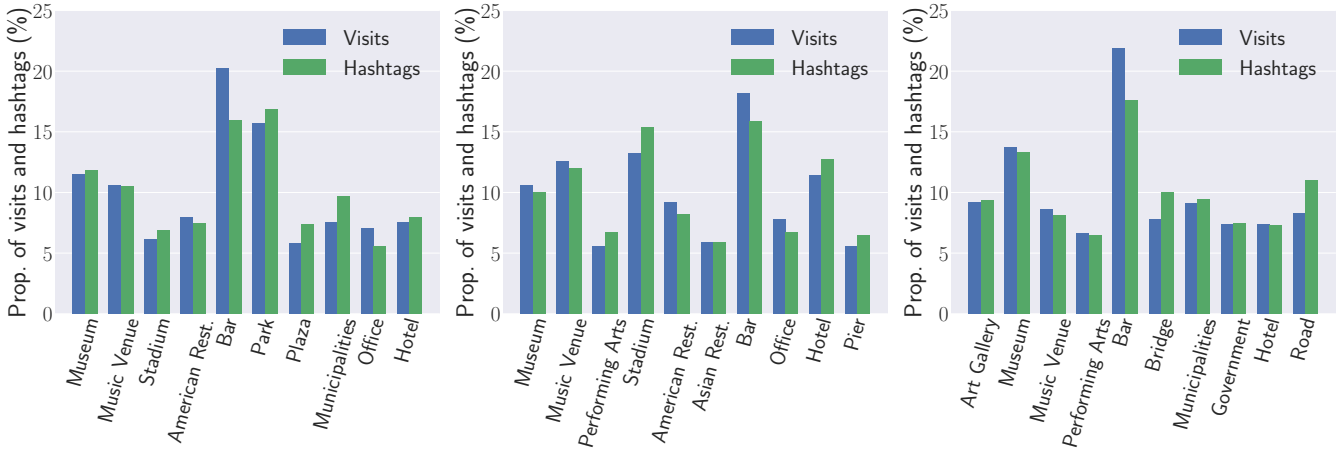
We aggregate each hashtag's proportion of share times to the granularity of quarters starting from the first quarter in 2012 to the last quarter of 2015.[9] This indicates that each hashtag's temporal pattern is organized as a 16-dimension vector (4 years × 4 quarters). k-means is adopted to perform clustering. We first try to fit each hashtag's temporal patterns, i.e., the 16-dimension vector, directly to k-means, however, the resulting clusters are not very promising. This is due to the fact that some hashtags may share similar temporal patterns, however, their patterns' peak happen at different time, e.g., #christmas and #halloween.

Instead, we manually define features over each hashtag's temporal patterns which are described in Table 3. To ensure the robustness of our results, we concentrate on the top 1,000 hashtags in terms of share times [19]. In order to select a suitable number of clusters for k-means, i.e., k, we adopt the Silhouette value. Experimental evaluation suggests that $k = 4$ leads to the highest Silhouette value (around 0.6) in all the three datasets.

The resulting four clusters are named *Stable*, *Rising*, *Meteor*, and *Periodic* based on their temporal patterns. Their detailed descriptions are as follows.

- Stable: Hashtags in the first cluster in the beginning exhibit an increase in their usage. After a certain time point, their share times become stable.

[9]We neglect the data in 2010 Q4 and 2011 due to the small quantity as shown in Figure 4).



(a) New York (b) Los Angeles (c) London

Figure 6: Proportions of visits and hashtags shared at the most popular location categories in three datasets. Rest. in (a) and (b) means Restaurant.

Table 3: Features defined over each hashtag h's temporal pattern $\mathcal{T}_h$.

Description
Standard deviation of $\mathcal{T}_h$
Largest 3 values in $\mathcal{T}_h$
Mean of the 3 largest values in $\mathcal{T}_h$
Standard deviation of the 3 largest values in $\mathcal{T}_h$
Standard deviation of the 3 largest values' indices in $\mathcal{T}_h$
Smallest 3 values in $\mathcal{T}_h$
Mean of the 3 smallest values in $\mathcal{T}_h$
Standard deviation of the 3 smallest values in $\mathcal{T}_h$
Standard deviation of the 3 smallest values' indices in $\mathcal{T}_h$

- Rising: The share times of hashtags in this cluster increase steadily over time.
- Meteor: In this cluster, hashtags at some temporal point are suddenly shared a large number of times.
- Periodic: Hashtags in this cluster exhibit periodic popularity.

The sizes of the four clusters are not uniform. The Rising cluster is the largest one containing around 60% of all hashtags, followed by the Stable cluster (around 25%). Meteor, on the other hand, is the smallest cluster with around 7% of all hashtags.

Figure 5 depicts the temporal patterns of some hashtags in different clusters in the New York dataset. We can make several interesting observations. For instance, the Periodic cluster contains not only hashtags about seasons or holidays, such as #summer, but also those about season-related activities, such as #bbq. Moreover, hashtags describing (annual) sports events belong to this cluster as well, e.g., #superbowl. As expected, hashtags that are specific to a certain point of time are categorized in the Meteor cluster, one example is #2012. Interestingly, some hashtags that are specific to Instagram also belong to the Meteor cluster. For instance, #iger in the second quarter of 2012 gains a large popularity, while in 2015, no one uses it anymore.

3.2 Spatial Patterns

Next, we investigate the spatial patterns of hashtags. In particular, we are interested in at which types of places people are more willing to share hashtags. To get the type/category of a location in an Instagram post, we again resort to Foursquare (see Section 2). Foursquare organizes all its location categories into a two-level tree,[10] we focus on the fine-grained second-level which contains more than 300 different location types.

We pick the top 10 location categories that users visit the most number of times in each dataset within the corresponding city, and calculate the proportion of hashtags shared among these categories. For comparison, we further calculate the proportion of users' visits to locations under these categories.

Figure 6 plots the results. First of all, places belonging to the Bar category are among the most popular locations for people to visit in all the datasets. However, the proportions of hashtags shared at bars drop in all the cases, e.g., the drop in the London dataset is around 5%. This means people are less likely to share hashtags at bars. Another type of locations exhibiting the same result is Office. On the other hand, people are more willing to share hashtags at outdoor places, e.g., Park in New York, Pier in Los Angeles, and Road in London. These results show that the spatial information indeed influences users' hashtag-sharing behavior.

4 SEMANTIC CHANGE

In this section, we address our second research question: *Do hashtags exhibit semantic displacement?*. We start by describing how to express each hashtag's semantics, then focus on semantic displacement.

[10] https://developer.foursquare.com/docs/resources/categories

Table 4: 10 hashtags that are most semantically similar to #family, #sushi, and #r2d2 in the New York dataset.

#family	#sushi	#r2d2
#familytime	#sashimi	#c3po
#cousins	#nigiri	#droids
#aunt	#sushiporn	#bb8
#father	#spicytuna	#artoo
#mom	#chirashi	#astromech
#sisters	#sushiroll	#carbonite
#grandparents	#unagi	#xwing
#nephews	#sushirolls	#obiwankenobi
#familylife	#shrimptempura	#lukeskywalker
#siblings	#californiaroll	#starwars

4.1 Semantics of Hashtags

As a new-era language, hashtags convey interesting meanings. To study this, our first step is finding a tool to represent each hashtag's semantics. Here, we adopt the skip-gram model with negative sampling [38, 39]. Skip-gram, designed following the distributional hypothesis in linguistics, is essentially a shallow neural network model, it maps each word into a continuous vector which preserves the information of the word's contexts words in a large corpus.

In our case, we treat each hashtag as a "word", and all hashtags in a post as one "sentence". Then, we perform skip-gram over all the "sentences" for each dataset. We set each hashtag's vector's dimension to 300 following previous works [19, 38, 39].

To show the effectiveness of skip-gram on capturing hashtags' semantics, we perform a qualitative study to find some hashtags' most semantically similar ones with respect to the shortest cosine distance between learned vectors. Table 4 presents the results for #family, #sushi, and #r2d2 in the New York dataset. As we can see, all the semantically similar hashtags found for these hashtags indeed express quite similar meanings, e.g., #r2d2 is similar to #c3po, #droids, and #bb8.

4.2 Semantic Displacement

With the cultural evolution, some words in natural language exhibit semantic displacement, e.g., the word "gay". We are interested in whether semantic displacement can be observed on hashtags as well. Moreover, due to hashtags' inherent dynamic nature, their semantic displacement should be much faster than that of words which normally takes decades, therefore, our datasets spanning over 5 years are sufficient for this study.

To measure semantic displacement of each hashtag, we first split our datasets by years staring from 2011 to 2015. Then, we perform skip-gram to map hashtags in each year into a vector. This means each hashtag has up to 5 vectors. A *single semantic displacement* of a hashtag is measured over two consecutive years as the cosine distance of the hashtag's two vectors at those years. The *overall semantic displacement* of a hashtag is the mean of all its single semantic displacements.

However, the stochastic nature of the skip-gram model drives vectors learned at different years to reside at different coordinate axes. Therefore, we need to align each hashtag's vectors at different years together before calculating their cosine distance [19, 25]. To this end, we apply the orthogonal Procrustes approach used by Hamilton et al. [19]. Concretely, we use $M^{(t)} \in \mathbb{R}^{300\times|\mathcal{H}|}$ to represent the matrix containing all the hashtags' vectors' at year t, and $\mathcal{H}$ to represent the set of all hashtags. Then, we solve the following optimization objective function

$$\arg\min_{X} ||XM^{(t)} - M^{(t+1)}||_F$$
$$\text{subject to } XX^T = I$$

by applying singular value decomposition (SVD). Here, $||\cdot||_F$ denotes the Frobenius norm. After obtaining the result, we align vectors of a hashtag in two consecutive years together.

Again, to ensure the robustness of our results, we concentrate on the top 1,000 hashtags that are shared the most number of times. Our experimental evaluation shows that some hashtags indeed exhibit a large semantic displacement. In our three datasets, there are at least 10% of the hashtags with more than 0.4 overall semantic displacement. Given the fact that cosine distance lies in the range of $[0, 2]$,[11] the semantic shift is pretty large. Table 5 and Table 6 list the hashtags with the highest and lowest overall semantic displacement. We see that hashtags that are specific to Instagram have quite large semantic displacement, such as #insta, #nofilter, and #ig. Many of

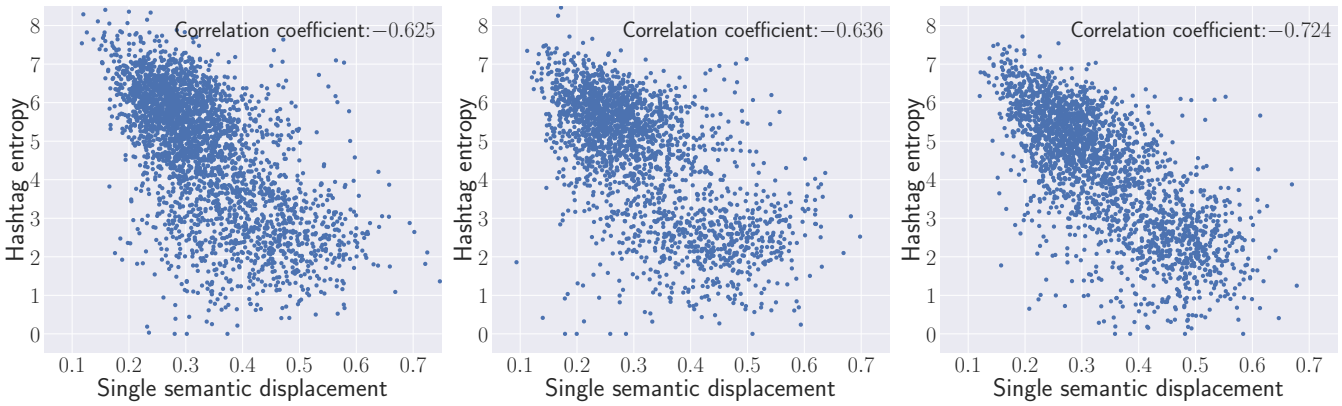
Table 5: 10 hashtags with the highest overall semantic displacement in three datasets.

New York	Los Angeles	London
#ontheroad	#hashtag	#iphoneography
#epic	#epic	#insta
#ilovemyjob	#hipstamatic	#world
#free	#winning	#today
#starwars	#nofilter	#popular
#winning	#insta	#potd
#pictureoftheday	#ig	#walking
#nofilter	#iphone	#ignation
#iphonography	#free	#instamoment
#yolo	#random	#studio

Table 6: 10 hashtags with the lowest overall semantic displacement in three datasets.

New York	Los Angeles	London
#nyc	#sweettooth	#pet
#beach	#dessert	#trees
#sky	#sweets	#beach
#sand	#trees	#tube
#eastriver	#palmtrees	#cat
#leaves	#soup	#kitty
#waves	#ocean	#skyscraper
#salad	#sunset	#thames
#sunset	#building	#smoothie
#pasta	#foodporn	#glutenfree

[11] Vectors learned by skip-gram are in $\mathbb{R}^{300}$, thus, the cosine distance is in $[0, 2]$.



(a) New York **(b) Los Angeles** **(c) London**

Figure 7: Relation between single semantic displacement and hashtag entropy in three datasets. Hashtag entropy is calculated on a yearly-base.

these hashtags are initially created to increase Instagram posts' popularity. We conjecture that when people are more used to these hashtags, they cannot attract as much attention as before, thus start to convey different meanings. On the other hand, we observe from Table 6 that nature-related hashtags are among those whose semantics stay rather stable, e.g., #sky, #trees, and #ocean. Also, food-related hashtags, such as #salad, #soup, and #smoothie, do not change their meanings that much.

Hamilton et al. show that high frequency words are less likely to change their meanings [19]. However, this is not the case for hashtags. We perform a correlation analysis between hashtags' frequency, i.e., share times, and single semantic displacement, and obtain a rather weak correlation (correlation coefficients are around -0.2 in all the datasets). Instead, we hypothesize that hashtags that are more uniformly shared among users are less likely to shift their meanings.

We propose *hashtag entropy* to measure each hashtag's sharing uniformity among users. Formally, a hashtag h, its entropy is defined as:

$$-\sum_{u \in \mathcal{U}} p(u,h) \log p(u,h)$$

where $\mathcal{U}$ denotes the set containing all the users and $p(u,h)$ represents the proportion of h shared by user u. It is worth noting that our hashtag entropy follows the same definition as the Shannon entropy: Higher entropy implies that the hashtag is more uniformly shared among users.

For each hashtag, we calculate multiple entropy at each year (from 2012 to 2015), then correlate each entropy with the corresponding single semantic displacement. Figure 7 depicts the results. In all the three datasets, we obtain strong negative correlation. In particular, the correlation coefficient in the London dataset is -0.724. These results show that if a hashtag is shared by many users in a similar frequency, i.e., high entropy, then it is less prone to semantic displacement.

5 HASHTAGS AND FRIENDSHIP

In this section, we address our last research question: *Can hashtags be used to infer social relations?*. We first describe our methodology on using hashtags for friendship prediction, then present the experimental evaluation.

5.1 Methodology

Social network is the major platform for people to share hashtags, this naturally leads to the question whether hashtags are related to people's social relations. To answer this question, we perform a friendship prediction task solely based on hashtags.

Friendship prediction is normally modeled as a binary classification task [1, 3, 4, 29, 31, 48, 57]. We can manually define features over two users' hashtags, and perform prediction either in a supervised manner (with the help of machine learning classifier) or an unsupervised manner. However, feature engineering is very time-consuming, and in many cases, the resulting features are not complete. Also, to define features over two users' hashtags, a natural approach is concentrating on their common hashtags. As most pairs of users share no common hashtags, this approach can be only applied to a small subset of user pairs, which cannot provide a complete picture on the relation between hashtags and social networks.

Instead, we choose to learn a hashtag profile, i.e., a feature vector, for each user, and compare two users' profiles to perform friendship prediction. As all users' hashtag profiles (feature vectors) are in the same dimension, this allows us to predict any pair of users' friendship.

Our hashtag profile learning follows the recent advancement of graph embedding [3, 13, 17, 20, 22, 32, 47, 50, 51, 53, 59, 62, 64]. In this setting, profiles are automatically learned following a general optimization objective function. Concretely, we organize users and hashtags into a weighted bipartite graph. For an edge connecting a user and a hashtag, its weight equals to the number of times the user shares the hashtag. We simulate a certain number of random walks starting from each user, referred to as *walk times*, on the

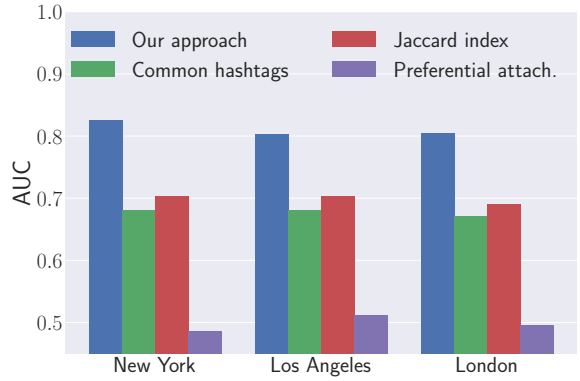


Figure 8: AUC for friendship prediction in three datasets. Preferential attach. means preferential attachment.

graph. The transition probability from each node to the next one follows the corresponding edges' weight. Each walk has a certain length, specified by the number of steps, which we refer to as *walk length*. This leaves us with a set of random walk traces. Then, we rely on the following optimization objective function to learn each user's hashtag profile.

$$\arg\max_{\theta} \prod_{v \in \mathcal{U} \cup \mathcal{H}} p(v|N(v);\theta)$$

Here, $N(v)$ denotes the neighborhood of node v[12] and $\theta(v)$ is the learned profile of node v. Moreover, $p(v|N(v);\theta)$ is modeled with a softmax function. Different from previous graph embedding approaches which define their objective functions following skip-gram [17, 47], our objective function is essentially the continuous bag-of-words (CBOW) model [38, 39]. We choose CBOW over skip-gram due to its better performance. In addition, we apply the negative sampling approach to speed up the learning process.

In the end, for any two users, we calculate their learned profiles' cosine distance, and predict them to be friends if the cosine distance is below a chosen threshold.[13] Note that our prediction does not need the knowledge of any existing friendships, therefore, it is unsupervised.

5.2 Evaluation

Experimental Setup. For each dataset, we randomly sample the same number of stranger pairs as the number of friend pairs to construct the negative cases. Then, we compute cosine distances for all friend and stranger pairs. AUC (area under the ROC curve) is adopted as the evaluation metric [3, 18, 56, 57]. There are mainly three hyperparameters in our model, i.e., walk length, walk times, and the dimension of the learned profile. Based on cross validation, we set them to 120, 80, and 512, respectively.

Baseline Models. We further establish three baseline models following the traditional link prediction approach [29], namely Jaccard index, common hashtags, and preferential attachment. Their formal definition is presented in Table 7.

[12]Following previous works [3, 17], $N(v)$ includes 10 nodes precedent and after v in all the random walk traces.

[13]We also learn a set of vectors for all hashtags, as we concentrate on friendship prediction, these hashtag vectors are simply neglected.

Table 7: Formal definition of baseline models for friendship prediction. $\mathcal{H}(u)$ is the set containing all the hashtags u has shared.

Baseline	Definition
Common hashtags	$\lvert\mathcal{H}(u) \cap \mathcal{H}(u')\rvert$
Jaccard index	$\frac{\lvert\mathcal{H}(u)\cap\mathcal{H}(u')\rvert}{\lvert\mathcal{H}(u)\cup\mathcal{H}(u')\rvert}$
Preferential attachment	$\lvert\mathcal{H}(u)\rvert \cdot \lvert\mathcal{H}(u')\rvert$

Results. Figure 8 depicts the AUCs for friendship prediction. In all the three datasets, we obtain more than 0.8 AUC which shows that hashtags can provide strong signals on inferring social relations. In particular, the prediction in the New York dataset achieves the best performance.

Our approach also outperforms all the three baseline models significantly. For the best baseline model, i.e., Jaccard index, we achieve around 20% performance gain in all the datasets. One of the major advantages of our approach is its ability to predict two users' friendship regardless of whether they share common hashtags. We observe that even for pairs of users sharing no common hashtags, our prediction still achieves a decent performance in all the datasets: 0.759 AUC in New York, 0.733 AUC in Los Angeles, and 0.728 AUC in London. This further demonstrates the effectiveness of our approach.

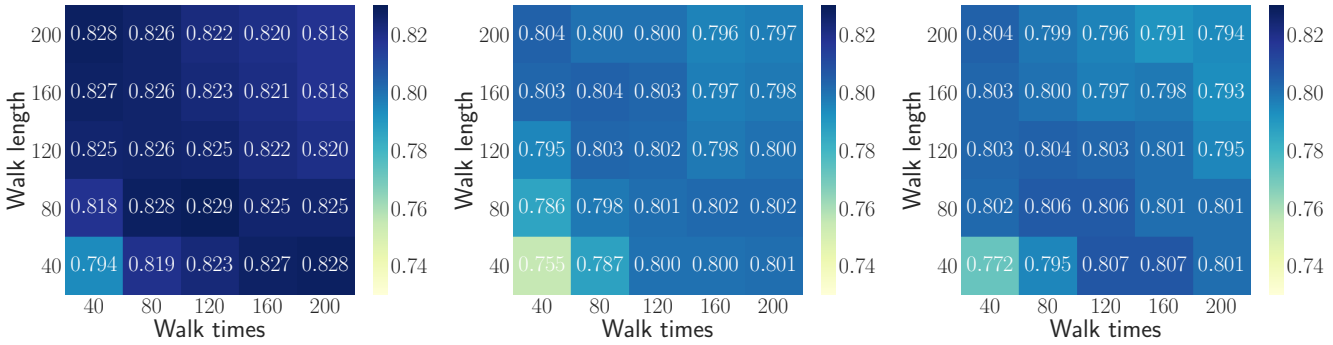
We also study the influence of hyperparameters on the prediction performance. Figure 9 depicts the heatmap when jointly tuning walk times and walk length. We see that when these two values are small, i.e., 40, the prediction results are relatively weak in all the three datasets. When increasing both, the performance gets better. However, the relation between the prediction result and the magnitude of the hyperparameters is not monotonic: When we set both walk times and walk length to 200, the prediction results drop in all cases. Note that we also perform the same study on the dimension of each learned profile, and observe that 512 leads to the best prediction.

6 RELATED WORK

Hashtags provide us with an unprecedented chance to understand the modern society. Researchers have studied hashtags from many perspectives [2, 5, 6, 8–10, 12, 14–16, 23, 24, 26–28, 30, 33, 36, 37, 41–43, 49, 52, 54, 55, 58, 60, 63, 65].

Souza et al. use #selfie to study the phenomenal self-portrait behavior in OSNs [58]. They collect a large dataset from Instagram, and show that the amount of posts associated with #selfie increases 900 times from 2012 to 2014. Moreover, their results suggest that posts with #selfie attract more likes and comments than others. The authors further show that there exist cultural variations of selfie behavior across countries.

Mejova et al. study food-related hashtags, such as #foodporn, on Instagram to understand people's dining behavior on a global scale [37]. They first show that desserts in social media are dominating over local cuisines. Then, they discover through hashtags



(a) New York **(b) Los Angeles** **(c) London**

Figure 9: Influence of walk length and walk times on the prediction performance in three datasets. Each value in the matrix represents the corresponding AUC.

that food can motivate people to engage in a healthier life style. Similar to #selfie, Mejova et al. show that posts associated with healthy-related hashtags attract more likes than others.

David et al. use hashtags together with the smiley face emoji to perform sentimental analysis on Twitter [10]. In this work, hashtags are used as the sentimental labels, and the authors summarize features over tweets. Evaluation results show that their classification achieves very effective performance.

Multiple works also use hashtags to study various political movements. Olteanu et al. analyze the demographics behind #blacklivesmatter, and conclude that African-Americans and young females engage more with the hashtag than others [42]. Manikonda et al. perform a comparative analysis of #metoo shared on Twitter and Reddit [34]. They observe that posts on Reddit concentrate on sharing personal stories while tweets express public support for the #metoo movement.

Zhang et al. study the privacy implications of sharing hashtags [65]. They first utilize a simple bag-of-words model and a random forest classifier to perform location prediction based on hashtags. Evaluation shows that their approach achieves more than 70% accuracy over fine-grained locations. Then, the authors propose a privacy-preserving system, namely Tagvisor, to mitigate the location privacy risks. Tagvisor implements three different obfuscation mechanisms including hiding, replacement, and (location category) generalization. Extensive experiments suggest that by obfuscating two hashtags, Tagvisor can successfully mislead the location prediction model, while maintaining a high-level utility with respect to hashtag semantics.

While the above works concentrate on either a certain type of hashtags or a certain property of hashtags, Ferragina et al. perform a general analysis of hashtags on Twitter [12]. In particular, they concentrate on the semantics of Twitter's hashtags. They first build a hashtag-entity graph over a large number of tweets. Then, the authors perform two natural language processing tasks, namely hashtag relatedness and hashtag classification, based on the features extracted from the constructed hashtag-entity graph. Experimental results show that this approach outperforms state-of-the-art solutions by a large extent. Besides targeting a different OSN than Ferragina et al. [12], we perform a much broader analysis on hashtags with several new angles, such as spatial pattern, semantic displacement, and social signal.

7 CONCLUSION

In this paper, we perform the first large-scale analysis on understanding hashtags shared on Instagram. Our study is centered around three research questions which aim at understanding hashtags from three different dimensions, i.e., the temporal-spatial dimension, the semantic dimension, and the social dimension. We collect three large datasets from Instagram containing more than 7 million hashtags shared over 5 years to perform our analyses.

We first show that hashtags can be categorized into four different clusters according to their temporal patterns, and people are more willing to share hashtags at certain places, such as parks. We then discover that some hashtags indeed exhibit large semantic displacement. Moreover, we propose a notion namely hashtag entropy and show the strong negative correlation between hashtag entropy and semantic displacement. In the end, we propose a bipartite graph embedding model to summarize users' hashtag profiles and rely on these profiles to perform friendship prediction. The effective prediction performance suggests that there is a strong connection between hashtags and social relations.

8 ACKNOWLEDGMENTS

This work was partially supported by the German Federal Ministry of Education and Research (BMBF) through funding for the Center for IT-Security, Privacy and Accountability (CISPA) (FKZ: 16KIS0656). The author would like to thank Dr. Sandra Strohbach and Dr. Mathias Humbert for their constructive feedback on the manuscript.

REFERENCES

[1] Luca Maria Aiello and Nicola Barbieri. 2017. Evolution of Ego-networks in Social Media with Link Recommendations. In *Proceedings of the 2017 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 111–120.
[2] Jisun An and Ingmar Weber. 2016. #greysanatomy versus #yankees: Demographics and Hashtag Use on Twitter. In *Proceedings of the 2016 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 523–526.
[3] Michael Backes, Mathias Humbert, Jun Pang, and Yang Zhang. 2017. walk2friends: Inferring Social Links from Mobility Profiles. In *Proceedings of the 2017 ACM*

SIGSAC Conference on Computer and Communications Security (CCS). ACM, 1943–1957.

[4] Lars Backstrom and Jure Leskovec. 2011. Supervised Random Walks: Predicting and Recommending Links in Social Networks. In *Proceedings of the 2011 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 635–644.

[5] Adam Badawy, Emilio Ferrara, and Kristina Lerman. 2018. Analyzing the Digital Traces of Political Manipulation: The 2016 Russian Interference Twitter Campaign. *CoRR abs/1802.04291* (2018).

[6] Yi Chang, Jiliang Tang, Dawei Yin, Makoto Yamada, and Yan Liu. 2016. Timeline Summarization from Social Media with Life Cycle Models. In *Proceedings of the 2016 International Joint Conferences on Artifical Intelligence (IJCAI)*. IJCAI, 3698–3704.

[7] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. 2011. Friendship and Mobility: User Movement in Location-based Social Networks. In *Proceedings of the 2011 ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 1082–1090.

[8] Michael Conover, Jacob Ratkiewicz, Matthew R Francisco, Bruno Goncalves, Alessandro Flammini, and Filippo Menczer. 2011. Political Polarization on Twitter. In *Proceedings of the 2011 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 89–96.

[9] Anqi Cui, Min Zhang, Yiqun Liu, Shaoping Ma, and Kuo Zhang. 2012. Discover Breaking Events with Popular Hashtags in Twitter. In *Proceedings of the 2012 ACM International Conference on Information and Knowledge Management (CIKM)*. ACM, 1794–1798.

[10] Dmitry David, Oren Tsur, and Ari Rappoport. 2010. Enhanced Sentiment Learning Using Twitter Hashtags and Smileys. In *Proceedings of the 2010 International Conference on Computational Linguistics (Coling)*. ACL, 241–249.

[11] Yuxiao Dong, Jie Tang, Sen Wu, Jilei Tian, Nittiest V Chawla, Jinghai Rao, and Huanhuan Cao. 2012. Link Prediction and Recommendation across Heterogeneous Social Networks. In *Proceedings of the 2012 International Conference on Data Mining (ICDM)*. IEEE, 181–190.

[12] Paolo Ferragina, Francesco Piccinno, and Roberto Santoro. 2015. On Analyzing Hashtags in Twitter. In *Proceedings of the 2015 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 110–119.

[13] Ming Gao, Leihui Chen, Xiangnan He, and Aoying Zhou. 2018. BiNE: Bipartite Network Embedding. In *Proceedings of the 2018 International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 715–724.

[14] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. 2016. Quantifying Controversy in Social Media. In *Proceedings of the 2016 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 33–42.

[15] Ruth Garcia Gavilanes, Neil O'Hare, Luca Maria Aiello, and Alejandro Jaimes. 2013. Follow My Friends This Friday! An Analysis of Human-Generated Friendship Recommendations. In *Proceedings of the 2013 International Conference on Social Informatics (SocInfo)*. Springer, 46–59.

[16] Eduardo Graells-Garrido, Mounia Lalmas, and Daniele Quercia. 2013. Data Portraits: Connecting People of Opposing Views. *CoRR abs/1311.4658* (2013).

[17] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 2016 ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 855–864.

[18] Inken Hagestedt, Yang Zhang, Mathias Humbert, Pascal Berrang, Haixu Tang, XiaoFeng Wang, and Michael Backes. 2019. MBeacon: Privacy-Preserving Beacons for DNA Methylation Data. In *Proceedings of the 2019 Network and Distributed System Security Symposium (NDSS)*. Internet Society.

[19] William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. In *Proceedings of the 2016 Annual Meeting of the Association for Computational Linguistics (ACL)*. ACL, 1489–1501.

[20] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Proceedings of the 2017 Annual Conference on Neural Information Processing Systems (NIPS)*. NIPS, 1025–1035.

[21] Yuheng Hu, Lydia Manikonda, and Subbarao Kambhampati. 2014. What We Instagram: A First Analysis of Instagram Photo Content and User Types. In *Proceedings of the 2014 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 595–598.

[22] Xiao Huang, Qingquan Song, Jundong Li, and Xia Hu. 2018. Exploring Expert Cognition for Attributed Network Embedding. In *Proceedings of the 2018 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 270–278.

[23] Jin Yea Jang, Kyungsik Han, Patrick C. Shih, and Dongwon Lee. 2015. Generation Like: Comparative Characteristics in Instagram. In *Proceedings of the 2015 Annual ACM Conference on Human Factors in Computing Systems (CHI)*. ACM, 4039–4042.

[24] Shoubin Kong, Qiaozhu Mei, Ling Feng, Fei Ye, and Zhe Zhao. 2014. Predicting Bursts and Popularity of Hashtags in Real-Time. In *Proceedings of the 2014 International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 927–930.

[25] Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2015. Statistically Significant Detection of Linguistic Change. In *Proceedings of the 2015 International Conference on World Wide Web (WWW)*. ACM, 625–635.

[26] Su Mon Kywe, Tuan-Anh Hoang, Ee-Peng Lim, and Feida Zhu. 2012. On Recommending Hashtags in Twitter Networks. In *Proceedings of the 2012 International Conference on Social Informatics (SocInfo)*. Springer, 337–350.

[27] Janette Lehmann, Bruno Goncalves, José J. Ramasco, and Ciro Cattuto. 2012. Dynamical Classes of Collective Attention in Twitter. In *Proceedings of the 2012 International Conference on World Wide Web (WWW)*. ACM, 251–260.

[28] Cheng-Te Li, Chien-Yuan Wang, Chien-Lin Tseng, and Shou-De Lin. 2011. MemeTube: A Sentiment-based Audiovisual System for Analyzing and Displaying Microblog Messages. In *Proceedings of the 2011 Annual Meeting of the Association for Computational Linguistics (ACL)*. ACL, 32–37.

[29] David Liben-Nowell and Jon Kleinberg. 2007. The Link-prediction Problem for Social Networks. *Journal of the American Society for Information Science and Technology* (2007).

[30] Yu-Ru Lin, Drew Margolin, Brian Keegan, Andrea Baronchelli, and David Lazer. 2013. #Bigbirds Never Die: Understanding Social Dynamics of Emergent Hashtags. In *Proceedings of the 2013 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 370–379.

[31] Qingyun Liu, Shiliang Tang, Xinyi Zhang, Xiaohan Zhao, Ben Y. Zhao, and Haitao Zheng. 2016. Network Growth and Link Prediction Through an Empirical Lens. In *Proceedings of the 2016 ACM Internet Measurement Conference (IMC)*. ACM, 1–15.

[32] Yao Ma, Zhaochun Ren, Ziheng Jiang, Jiliang Tang, and Dawei Yin. 2018. Multi-Dimensional Network Embedding with Hierarchical Structure. In *Proceedings of the 2018 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 387–395.

[33] Zongyang Ma, Aixin Sun, and Gao Cong. 2012. Will This #Hashtag Be Popular Tomorrow?. In *Proceedings of the 2012 International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 1173–1174.

[34] Lydia Manikonda, Ghazaleh Beigi, Huan Liu, and Subbarao Kambhampati. 2018. Twitter for Sparking a Movement, Reddit for Sharing the Moment: #metoo through the Lens of Social Media. *CoRR abs/1803.08022* (2018).

[35] Lydia Manikonda, Yuheng Hu, and Subbarao Kambhampati. 2014. Analyzing User Activities, Demographics, Social Network Structure and User-Generated Content on Instagram. *CoRR abs/1410.8099* (2014).

[36] Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. 2012. Adding Semantics to Microblog Posts. In *Proceedings of the 2012 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 563–572.

[37] Yelena Mejova, Sofiane Abbar, and Hamed Haddadi. 2016. Fetishizing Food in Digital Age: #foodporn Around the World. In *Proceedings of the 2016 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 250–258.

[38] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of the 2013 International Conference on Learning Representations (ICLR)*.

[39] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionally. In *Proceedings of the 2013 Annual Conference on Neural Information Processing Systems (NIPS)*. NIPS, 3111–3119.

[40] Fred Morstatter, Jurgen Pfeffer, Huan Liu, and Kathleen M. Carley. 2013. Is the Sample Good Enough? Comparing Data from Twitter's Streaming API with Twitter's Firehose. In *Proceedings of the 2013 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 400–408.

[41] Ferda Ofli, Yusuf Aytar, Ingmar Weber, Raggi al Hammouri, and Antonio Torralba. 2017. Is Saki #delicious?: The Food Perception Gap on Instagram and Its Relation to Health. In *Proceedings of the 2017 International Conference on World Wide Web (WWW)*. ACM, 509–518.

[42] Alexandra Olteanu, Ingmar Weber, and Daniel Gatica-Perez. 2016. Characterizing the Demographics Behind the #BlackLivesMatter Movement. *CoRR abs/1512.05671* (2016).

[43] Raphael Ottoni, Diego Las Casas, Joao Paulo Pesce, Wagner Meira Jr, Christo Wilson, Alan Mislove, and Virgilio Almeida. 2014. Of Pins and Tweets: Investigating How Users Behave Across Image- and Text-Based Social Networks. In *Proceedings of the 2014 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 386–395.

[44] Jun Pang and Yang Zhang. 2015. Location Prediction: Communities Speak Louder than Friends. In *Proceedings of the 2015 ACM Conference on Online Social Networks (COSN)*. ACM, 161–171.

[45] Jun Pang and Yang Zhang. 2017. Quantifying Location Sociality. In *Proceedings of the 2017 ACM Conference on Hypertext and Social Media (HT)*. ACM, 145–154.

[46] Beatrice Perez, Mirco Musolesi, and Gianluca Stringhini. 2018. You Are Your Metadata: Identification and Obfuscation of Social Media Users using Metadata Information. In *Proceedings of the 2018 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 241–250.

[47] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 2014 ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 701–710.

[48] Huy Pham, Cyrus Shahabi, and Yan Liu. 2013. EBM: An Entropy-Based Model to Infer Social Strength from Spatiotemporal Data. In *Proceedings of the 2013*

ACM SIGMOD International Conference on Management of Data (SIGMOD). ACM, 265–276.

[49] Lisa Posch, Claudia Wagner, Philipp Singer, and Markus Strohmaier. 2013. Meaning as Collective Use: Predicting Semantic Hashtag Categories on Twitter. In *Proceedings of the 2013 International Conference on World Wide Web (WWW)*. ACM, 621–628.

[50] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec. In *Proceedings of the 2018 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 459–467.

[51] Meng Qu, Jian Tang, and Jiawei Han. 2018. Curriculum Learning for Heterogeneous Star Network Embedding via Deep Reinforcement Learning. In *Proceedings of the 2018 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 468–476.

[52] Daniele Quercia, Rossano Schifanella, and Luca Maria Aiello. 2014. The Shortest Path to Happiness: Recommending Beautiful, Quiet, and Happy Routes in the City. In *Proceedings of the 2014 ACM Conference on Hypertext and Social Media (HT)*. ACM, 116–125.

[53] Suhas Ranganath, Ghazaleh Beigi, and Huan Liu. 2018. Curriculum Learning for Heterogeneous Star Network Embedding via Deep Reinforcement Learning. In *Proceedings of the 2018 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 468–476.

[54] Jacob Ratkiewicz, Michael D. Conover, Mark Meiss, Bruno Goncalves, Alessandro Flammini, and Filippo Menczer. 2011. Detecting and Tracking Political Abuse in Social Media. In *Proceedings of the 2011 International Conference on Weblogs and Social Media (ICWSM)*. AAAI, 297–304.

[55] Daniel M. Romero, Brendan Meeder, and Jon Kleinberg. 2011. Differences in the Mechanics of Information Diffusion Across Topics: Idioms, Political Hashtags, and Complex Contagion on Twitter. In *Proceedings of the 2011 International Conference on World Wide Web (WWW)*. ACM, 695–704.

[56] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. 2019. ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models. In *Proceedings of the 2019 Network and Distributed System Security Symposium (NDSS)*. Internet Society.

[57] Salvatore Scellato, Anastasios Noulas, and Cecilia Mascolo. 2011. Exploiting Place Features in Link Prediction on Location-based Social Networks. In *Proceedings of the 2011 ACM Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 1046–1054.

[58] Flavio Souza, Diego de Las Casas, Vinicius Flores, SunBum Youn, Meeyoung Cha, Daniele Quercia, and Virgilio Almeida. 2015. Dawn of the Selfie Era: The Whos, Wheres, and Hows of Selfies on Instagram. In *Proceedings of the 2015 ACM Conference on Online Social Networks (COSN)*. ACM, 221–231.

[59] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proceedings of the 2015 International Conference on World Wide Web (WWW)*. ACM, 1067–1077.

[60] Oren Tsur and Ari Rappoport. 2012. What's in a Hashtag? Content based Prediction of the Spread of Ideas in Microblogging Communities. In *Proceedings of the 2012 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 643–652.

[61] Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P. Gummadi. 2009. On the Evolution of User Interaction in Facebook. In *Proceedings of the 2009 ACM Workshop on Online social networks (WOSN)*. ACM, 37–42.

[62] Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. 2018. SHINE: Signed Heterogeneous Information Network Embedding for Sentiment Link Prediction. In *Proceedings of the 2018 ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 592–600.

[63] Lei Yang, Tao Sun, Ming Zhang, and Qiaozhu Mei. 2012. We Know What You #Tag: Does the Dual Role Affect Hashtag Adoption?. In *Proceedings of the 2012 International Conference on World Wide Web (WWW)*. ACM, 261–270.

[64] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. 2018. Hierarchical Graph Representation Learning with Differentiable Pooling. *CoRR abs/1806.08804* (2018).

[65] Yang Zhang, Mathias Humbert, Tahleen Rahman, Cheng-Te Li, Jun Pang, and Michael Backes. 2018. Tagvisor: A Privacy Advisor for Sharing Hashtags. In *Proceedings of the 2018 Web Conference (WWW)*. ACM, 287–296.

Neural IR Meets Graph Embedding: A Ranking Model for Product Search

Yuan Zhang
Department of Machine Intelligence
Key Laboratory of Machine Perception (MOE)
Peking University, China
yuan.z@pku.edu.cn

Dong Wang
Center for Data Science
Peking University, China
wangdongcis@pku.edu.cn

Yan Zhang
Department of Machine Intelligence
Key Laboratory of Machine Perception (MOE)
Peking University, China
zhy@cis.pku.edu.cn

ABSTRACT

Recently, neural models for information retrieval are becoming increasingly popular. They provide effective approaches for product search due to their competitive advantages in semantic matching. However, it is challenging to use graph-based features, though proved very useful in IR literature, in these neural approaches. In this paper, we leverage the recent advances in graph embedding techniques to enable neural retrieval models to exploit graph-structured data for automatic feature extraction. The proposed approach can not only help to overcome the long-tail problem of click-through data, but also incorporate external heterogeneous information to improve search results. Extensive experiments on a real-world e-commerce dataset demonstrate significant improvement achieved by our proposed approach over multiple strong baselines both as an individual retrieval model and as a feature used in learning-to-rank frameworks.

CCS CONCEPTS

• **Information systems → Retrieval models and ranking**; *Information retrieval query processing*.

KEYWORDS

Neural Information Retrieval, Graph Embedding, Product Search

ACM Reference Format:
Yuan Zhang, Dong Wang, and Yan Zhang. 2019. Neural IR Meets Graph Embedding: A Ranking Model for Product Search. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313468

1 INTRODUCTION

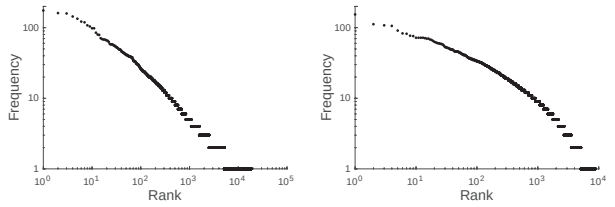
Online shopping has become increasingly popular over the past decade. Those e-commerce sites normally sell a vast number of products at the same time and provide plenty of choices under each category. For instance, there are hundreds of million products in Amazon and over one billion in eBay. It is almost impossible for consumers to find items that they are interested in without the help of an effective product search engine. On the other side, an improvement in product search accuracy could bring massive revenue to online retailers [1].

In a typical product search scenario, users formulate queries using characteristics of the product of interest, for example, the producer's name, brand, or terms which describe the category of the product [38]. However, the lexical gap between queries and product descriptions [42] makes it difficult to exploit such traditional information retrieval models as LSI [5], BM25 [37], which focus more on lexical matching, in this application scenario.

The recent neural network-based models (sometimes also known as deep learning models) [27] provide a good solution to this problem with their advantages in semantic matching. Yet, despite great success of deep learning in processing speech (1D), image (2D), and video (3D) data, neural models are rather limited in dealing with *high-dimensional* graph-structured data, which, on the other hand, has been extensively studied in the existing IR literature (e.g., the graph-based ranking features PageRank [31], HITS [21]; the graph-based similarity metric SimRank [17]; the click graph-based smoothing and propagation models [4, 11, 18, 23, 45]; the heterogeneous graph-based query intent learning [35]).

In fact, the neural approach and the graph-based approach are based on two different types of *inductive biases* that are complementary to each other. The inductive bias for the former is the smoothness with respect to input features (i.e., $f(x+\epsilon) \approx f(x)$ for small ϵ). For example, given queries "multifunction printer" and "all-in-one printer", neural nets should give similar predictions. On the contrary, the graph-based approach is mostly based on the smoothness regarding node proximities where node can be queries, items, sessions, etc. (i.e., $f(x) \approx f(y)$ if node x and y are "close" in a given graph). For example, queries q_1 and q_4 sharing common neighbors within a densely connected subgraph in Figure 2 are assumed to have similar outcomes and therefore the knowledge towards one query can be transferred to the other. Admittedly, in theory, neural nets can automatically learn the latter structure with sufficiently large amount of data. However, besides extensive computational burden, it is infeasible to obtain enough data for long-tailed queries/items in e-commerce search (Figure 1) and extremely hard for neural nets to learn high-order proximities in practice. Thus, it becomes an interesting research question whether we can combine these two complementary approaches for better retrieval performance.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313468



(a) query occurrence (b) product occurrence

Figure 1: The frequency-rank distribution (log-scale) of item and query occurrences in a sample of search logs from an e-commerce search engine, respectively.

In this paper, we propose to leverage the recent advances in *graph* [1] *embedding* techniques [12, 14] to integrate graph-structured data into a unified neural ranking framework. To the best of our knowledge, this is the first study on how to use the click-graph features in neural models for retrieval.

The benefits to allow neural retrieval models to incorporate graph-structured information are demonstrated in two-fold. Firstly, by integrating the click-graph features, our model can effectively overcome the query-item sparsity problem of the click-through data [11] and generalize better to unseen queries and long-tailed products. This is of great significance in the deep learning setting, where neural models not only require a large amount of data with sufficient coverage but also are prone to overfitting, which in turn causes poor retrieval performance on rare queries and reinforces the *rich-get-richer* effect [3]. Furthermore, we show that graphs can be used as convenient and efficient tools in our approach to fuse external heterogeneous information to improve search results, e.g., co-viewed items in a browsing session and co-purchased items in an order. Those features are, though, not very straightforward for existing neural ranking models to take advantage of.

From another perspective, our proposed approach can be seen as a way to tackle a side effect of neural networks, *unawareness of transitivity*, as their capacity (which can be loosely measured by the number of parameters) grows to fit non-smooth relations with limited data. For instance, as illustrated in Figure 2(a), query q_4 and q_5 share a common product click so that they tend to be queries with similar user intent and product p_3 (clicked through q_4) should also be considered somewhat relevant to q_5. However, neural network models themselves might not be aware of this transitive relation, because query-product pair (q_5, p_3) simply do not appear in the positive training examples and, when given sufficient capacity, neural networks do not require this relation to hold in order to preserve other positive relations either (see Section 5 for empirical evidence). While this issue can be implicitly mitigated to some extent with such regularization methods as L1/L2 penalties that merely "uninformatively" shrink the weights towards zero or with carefully designed neural architectures, it strongly motivates the direct use of graph-structured regularization towards utilizing the full potentials of neural networks for retrieval.

[1] In order to avoid confusion, we mostly use the word *graph* to refer to the collection of nodes and edges, and *network* to refer to the neural networks in this paper, although these two words are often used interchangeably in literature.

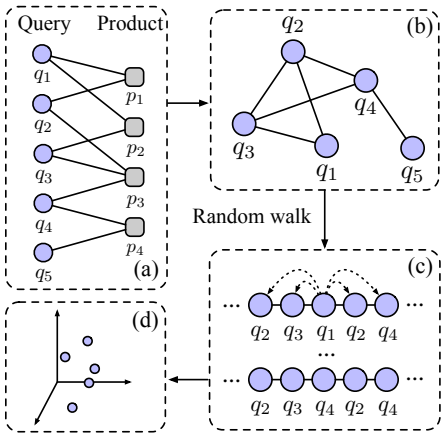


Figure 2: Graph embedding.

Finally, we evaluate our model on a real-world dataset of e-commerce search. Experimental results show that our model substantially outperforms state-of-the-art baselines both as an individual ranking model and as a feature in a learning-to-rank framework. Ablation study further demonstrates that the use of graph-embedding techniques leads to a significant improvement in retrieval performance.

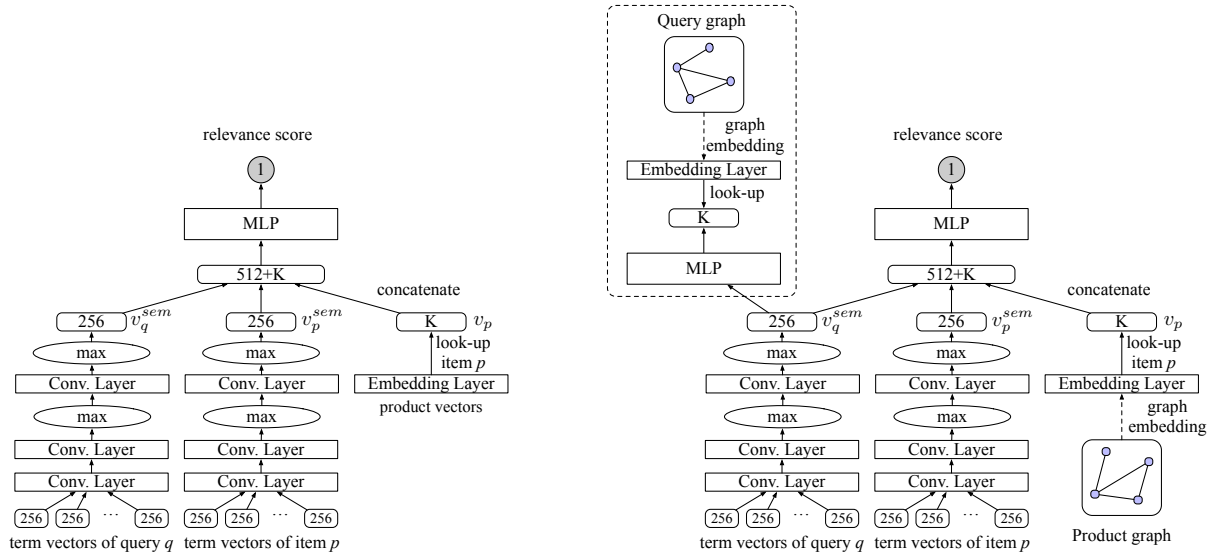
To conclude, the major contributions of this work are as follows:

- We propose a Graph Embedding-based ranking model for Product Search (GEPS), which is, to the best of our knowledge, the first to integrate click-graph features into a unified neural ranking framework.
- Our proposed graph embedding-based approach can effectively (a) deal with the long-tail sparsity problem and (b) combine heterogeneous external information into neural models to improve search results.
- Through extensive experimental study on a publicly available e-commerce dataset, we show that GEPS can achieve significant improvement in retrieval performance compared with the existing work.

2 BACKGROUND: GRAPH EMBEDDING

As mentioned in Section 1, there are many "hand-crafted" graph-based features in the existing IR literature. However, the lack of an effective representation makes it hard to apply deep learning models to graph-structured data for automatic feature extraction. Recently, the extensive study on the *graph embedding* provides a good solution to this problem. In this section, we introduce two popular state-of-the-art graph embedding methods, Deepwalk [33] and LINE [41], by taking the query graph induced by click-through data as an example.

As shown in Figure 2(a-b), we construct a query graph by establishing links between each pair of queries if they have once led to clicks on an identical item. (A product graph can also be constructed similarly.) This query graph contains rich signals for retrieval, for edges in the graph may indicate semantic similarities of queries. Intuitively, two queries that are nearby in the query graph tend to be similar to each other, and hence the knowledge we have for one



(a) Neural Base Model: NeuBase

(b) Proposed Model: GEPS

Figure 3: Model Architecture. We use dashed-line arrows to represent assignment of graph embedding vectors to the parameters/weights in embedding layers. K denotes the embedding dimension.

query can also be transferred to the other, which can be especially useful for a tail query that rarely appears in search logs. Motivated by this kind of intuition, the goal of graph embedding is to obtain vector representations for each node in a graph, as shown in Figure 2(d), so that the proximities between nodes can be well preserved.

Perozzi et al. [33] propose Deepwalk based on a distributional hypothesis on graphs, nodes with similar neighborhood tend to be similar to each other, which is also called the *second-order proximity*. They first generate random walk sequences and then use the node in the middle to predict the other nodes within a given window (Figure 2(c)), so that nodes occurring in similar contexts of random walk sequences will have close embeddings, as in Skip-gram [26] developed in language modeling. Thus, query q_1 and q_4 are supposed to have similar vector representations.

Further, Tang et al. [41] propose LINE to take the *first order proximity* into consideration, for example, query q_4 and q_5 are connected by a direct link and therefore should have close embeddings as well. Meanwhile, graph embedding is made scalable to graphs with over millions of nodes and billions of edges by an edge sampling technique proposed in their work. Finally, the embedding vectors preserving the two kinds of proximities are obtained respectively in LINE, and they can either be used separately or combined in a supervised manner.

3 MODEL FRAMEWORK

3.1 Preliminaries and Notations

Like other neural ranking models, we use click-through data to train our models. The click-through data used in our study consists of queries and products with their co-clicked information. We denote the set of queries as Q and each query is represented by a collection of terms $q = \{t_1^{(q)}, ..., t_M^{(q)}\}$. Similarly, the set of products is denoted by P and each product by $p = \{t_1^{(p)}, ..., t_N^{(p)}\}$. We also denote as w the word vector for each term t with the same subscripts and superscripts, for example, $w_1^{(q)}$ denotes the word vector of $t_1^{(q)}$. For each query $q \in Q$, the set of clicked items are represented by P_q, and items that are presented in the SERP (search engine results page) but not clicked as $\overline{P_q}$.

The query graph and the product graph are defined as $G_{query} = (Q, E_Q)$ and $G_{product} = (P, E_P)$, respectively. After graph embedding, we can obtain graph embedding vectors e_q for each query $q \in Q$ and e_p for each product $p \in P$. The graph embedding procedures then can be defined as mappings, $g : G(V, E) \mapsto \{e_v, \forall v \in V\}$.

While our proposed model framework is not limited to any specific graphs G_{query} and $G_{product}$ as long as they can provide meaningful interaction information between their nodes (i.e., queries and products) for search, the reader can assume that the graphs of queries and products are induced by the click-through data as depicted in Figure 2(a-b) until further discussion in Section 3.3.

3.2 Proposed Model: GEPS

Our goal is to combine the advantages of both graph-based methods and neural approaches. In this section, we first introduce a simple yet typical neural network architecture for product search as the "base model", and then show how graph embedding techniques can be plugged into such neural network models to incorporate graph-structured information for better retrieval performance.

As shown in Figure 3(a), we represent terms of queries and product descriptions by their term vectors pre-trained on the whole corpus. Then, we input these term vectors to convolutional layers for semantic feature extraction, where the filter window is set to

three. Max pooling layers are used to reduce feature dimensions and eventually transform feature maps into a feature vector. We also use an id embedding vector v_p for each product, which is commonly used in e-commerce search engines to capture its intrinsic features not reflected in the product description. The semantic feature vector v_q^{sem} for the query and v_p^{sem} for the product along with the product vector v_p are concatenated into a long vector. Finally, we use an MLP (multi-layer perceptron) to output the relevance score.

To integrate graph-based features for products, we use graph embeddings to take the place of product id embeddings v_p in the network architecture described above. This is analogous to using pre-trained word embeddings instead of learning word vectors from scratch, which has already been proved to be both effective and efficient in natural language processing. We find it rather helpful to fine-tune these graph embeddings during training due to the unsupervised nature of graph embedding methods at hand.

While it is straightforward to incorporate product graph embeddings in the neural framework, making use of the query graph embeddings is quite challenging, because our retrieval model inevitably has to handle lots of new queries that are absent in the click-through data for training and hence do not have corresponding graph embedding vectors. [2] A tempting approach would be training a regression model f (e.g., CNN or RNN) with query terms as input that "mimics" the query graph embedding vectors, that is,

$$arg\min_{f} \|f(t_1^{(q)}, ..., t_M^{(q)}) - e_q\|_2^2, \tag{1}$$

and then use f to re-create embedding vectors for queries [3]. However, we find it hard to fine-tune query graph embeddings with this approach to achieve best performance; fine-tuning either the re-created vectors themselves or the model f could easily cause overfitting. Besides, it is not ideal to consider graph-based features and semantic features separately to model textual queries (different from the item side).

In our model, we propose to use an encoder-based approach to encode query-graph features in their corresponding semantic feature vectors. More specifically, we use an MLP to transform the semantic feature vectors v_q^{sem} into the same vector space as query graph embeddings e_q and try to minimize the reconstruction error as shown in Figure 3(b), that is,

$$\begin{aligned} min. \quad & \mathcal{L}_{re}(q) = \|\widetilde{e_q} - e_q\|_2^2 \\ st. \quad & \widetilde{e_q} = MLP(v_q^{sem}). \end{aligned} \tag{2}$$

Different from the other approach mentioned above, we can jointly optimize this reconstruction loss with the final ranking loss in the model training stage (see Section 3.4). Also, this encoder-based approach has an additional effect: the word vectors are further trained to capture relevance-based semantics [47] more accurately in such a way that the semantic vectors $v_q^{sem} = CNN(w_1^{(q)}, ..., w_M^{(q)})$ can better reconstruct high-order relevance relations encoded in the query graph. This approach actually shares a similar idea to *multi-task learning*, where relevant tasks (analogous to the graph embedding reconstruction here) are trained simultaneously for mutual regularization and knowledge transfer.

[2] Although there could also exist products missing in the click-through data, the proportion is much smaller in practice. We initialize them with all-zero vectors meaning without any prior knowledge. Empirically, this trick works well enough.

[3] A similar approach [34] is used by Pinter et al. to obtain embedding vectors for the out-of-vocabulary (OOV) words with a character-level RNN.

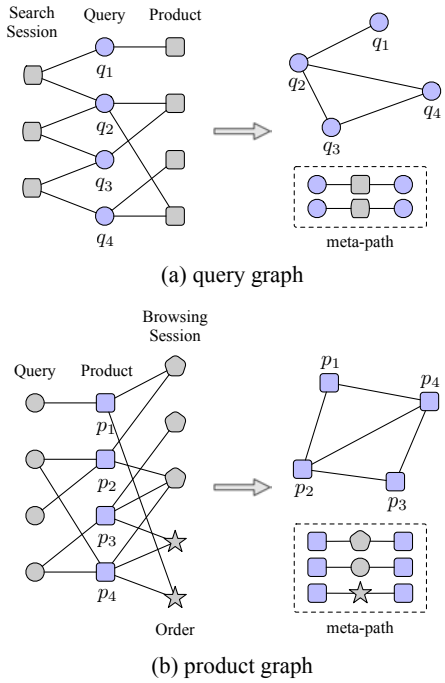


Figure 4: Fusing heterogeneous information with meta-paths.

Finally, it is worth mentioning that the proposed graph embedding-based approach can be used as a "plug-in" to boost the performance of almost any other neural retrieval model. Thus, the experimental results in this paper merely provide a lower bound for the performance of our proposed approach since we only use a simple base model for ease of demonstration.

3.3 Fusing Heterogeneous Information

Our proposed approach can utilize rich information on interactions between both queries and products contained in *graphs*, so that the knowledge we obtain from training data can be transferred to similar queries and correlated products for more accurate retrieval results. To fully take advantage of our graph-based approach, we can further incorporate fruitful external heterogeneous information, in addition to click-through data, into the graphs we use.

As shown in Figure 4(a), a search session may contain several queries; we can assume that the queries within a search session are more likely to share similar search intent. By regarding the heterogeneous relations among sessions, queries and products as a heterogeneous graph [35], we can use meta-paths [40] to construct the query graph. The semantics of graphs constructed by meta-paths are well interpretable, for example, the edge set E_{query} in Figure 4(a) contains all pairs of queries that are either issued in a search session or lead to clicks of an identical product.

We can also fuse more heterogeneous relations in the product graph as in Figure 4(b). In an e-commerce website, in addition to

direct search, customers also browse products through product category hierarchies, recommendations, etc.; products that are browsed at the same time can also be considered as somewhat related to each other. This can become especially useful in the cold-start scenarios where the products that are never clicked through any search queries. Besides, co-purchased items in the same order provide useful signals for search as well. Similarly, we make use of meta-paths to construct the product graph $G_{product}$.

Admittedly, more irrelevant factors and even data noises (e.g., unrelated queries in a session) are introduced into graphs as we fuse more heterogeneous information. Fortunately, our model is trained in a supervised manner so that the neural nets only extract useful features in terms of product search to produce the final results. Again, our proposed approach combines the advantages of both graph-based methods and deep learning approaches.

3.4 Model Training

In the training stage, we use a pairwise ranking loss to train our neural model. Given a search query q along with a positive product $p_+ \in P_q$ and a negative product $p_- \in \overline{P_q}$, the loss is defined as

$$\mathcal{L}_{rank}(q, p_+, p_-) = h(score(q, p_+) - score(q, p_-)), \tag{3}$$

where $score(q, p)$ is the relevance score given by our model for query-item pair (q, p) and h is the *smoothed hinge loss* [36],

$$h(x) = \begin{cases} 0 & x \geq 1, \\ 1/2 - x & x < 0, \\ (1-x)^2/2 & \text{otherwise.} \end{cases} \tag{4}$$

At the same time, we also minimize the reconstruction error as given in Eq. (2) for encoding graph embedding-based features in the query semantic vectors. To sum up, the overall loss function is

$$\mathcal{L} = \sum_{q \in Q} \sum_{p_+ \in P_q} \sum_{p_- \in \overline{P_q}} \mathcal{L}_{rank}(q, p_+, p_-) + \alpha \cdot \sum_{q \in Q} \mathcal{L}_{re}(q), \tag{5}$$

where α is a tunable parameter (we set $\alpha = 1$ in our experiments).

We apply the stochastic gradient descent (SGD) method with mini-batches to optimize the loss function given in Eq. (5). Meanwhile, the standard weight decay regularization and early stopping strategy are used to prevent overfitting. We also use gradient clipping in case gradient explosion happens. The batch size is set to 1024 and the weight decay is 10^{-4} in our experiments.

4 EXPERIMENTS

4.1 Research Questions

We are particularly interested in whether and, if so, how our proposed graph embedding-based approach can lead to performance gains in product search. We break down this principal research question into four individual ones to guide our experimental study:

RQ1. How does our proposed model GEPS compare to the state-of-the-art retrieval models?

RQ2. How does our graph embedding-based approach benefit neural models in dealing with the long-tail problem of click-through data?

RQ3. How does our graph embedding-based approach further benefit neural models by incorporating external heterogeneous information to improve search results?

Table 1: Dataset statistics.

Statistics	Value	
# products	184047	
# unique queries	26137	
Vocabulary size	181194	
Length of queries	2.66 ± 1.77	
Length of product descriptions	5.12 ± 2.04	
# items per order	1.34 ± 0.96	
	Original	Chronological
# queries (train/test)	35615/16218	28380/3011
# search sessions	26880	21505
# browsing sessions	349607	242852
# clicks	37705	30160
# views	1235380	857008
# orders	13506	9130

RQ4. How do graph embedding algorithms influence the performance of GEPS?

4.2 Experimental Setup

4.2.1 Dataset and Data Partition. In order to answer those research questions in Section 4.1, we evaluate our proposed model on the dataset from CIKM Cup 2016 Track 2 [4], which is also the only publicly available dataset of product search. The dataset contains five months of logs of an e-commerce search engine from 1st Jan to 1st Jun, 2016. We filter out the queries without textual information, for the search engine also allows "query-less" search.

In the **original data split**, the test set is obtained by taking the last queries of some sessions. However, with this data partitioning procedure, there could be larger overlaps between training topics and test topics than in real application scenarios, both lexically and semantically, for mainly two reasons: the queries in the same session but prior to a test query are very likely to share similar search intent; both the training and test sets are evenly distributed in the five-month period, not being significantly influenced by the topic shift in queries over time.

In order to simulate the real-world application scenario where models are trained using historical data and applied to unseen future queries, we also report experimental results with a **chronological data split**. We use the first four months' logs (until 1st May) for training and use the last month's test data for evaluation. Note that, we do not know those hidden search behaviors in the first four months (because they are not provided in the dataset), so the size of training data in this data split is only about 80 percent of that in the original data split and hence results on those two testbeds are not reasonably comparable. Also, the statistical significance of performance improvement could subsequently decline as the size of test data is reduced to 20 percent.

The detailed statistics about the dataset and data partitioning are listed in Table 1.

[4] https://competitions.codalab.org/competitions/11161

4.2.2 Baselines. We conduct comparison study with the following baseline methods.

- **LSI** (Latent Semantic Indexing) [5] is a classical vector space model in information retrieval. We set the dimension of vector spaces $K = 256$.
- **BM25** [37] is a very popular probabilistic model-based ranking function proposed by Robertson et al.
- **QLM** (Query-likelihood Model) with Jelinek-Mercer smoothing [48] is an effective entity search approach based on language models [42]. For each product p, we construct a statistical language model smoothed by the whole corpus,

$$P_\lambda(t|p) = (1-\lambda) \cdot P_{ml}(t|p) + \lambda \cdot P(t|C), \tag{6}$$

  where $P_{ml}(t|p)$ and $P(t|C)$ are simply given by relative counts of term t appearing in the description of product p and the whole corpus C, respectively. Then, we can obtain the relevance score for each product p given a query q according to Bayes' rule,

$$P(p|q) \propto P(p) \cdot P_\lambda(q|p) = P(p) \cdot \prod_{t \in C_q} P_\lambda(t|p), \tag{7}$$

  where $P(p)$ is the popularity of product p given by its click counts, and C_q is the collection of terms in query q. In our experiments, we set the smoothing parameter $\lambda = 0.01$ that turns out to be not very sensitive.
- **LSE** (Latent Semantic Entities) [42, 43] is a state-of-the-art latent vector space model for product search, which jointly learns vector representations of words and products from scratch. We choose 256 as the dimensions of both word vectors and product vectors.
- **VPCG** (Vector Propagation on Click Graphs) [18] is a state-of-the-art non-neural graph-based retrieval model that utilizes vector propagation on click graphs to overcome the above-mentioned long-tail problem and the lexical gap between queries and documents.
- **ARC-II** [15], **MatchPyramid** [32] are two state-of-the-art neural retrieval models . We use the implementations from the open-source toolkit MatchZoo[5] [9] specifically designed for neural IR. We set the embedding size to 256 and report the results achieved by the best of multiple model setups provided in MatchZoo.
- **NeuBase** (Figure 3(a)) is a variant of GEPS that does not use graph-based features (i.e., excluding the graph embedding-based components from GEPS) so as to investigate the performance improvement achieved by the proposed approach. In fact, this model is an extension of **C-DSSM** [39].

Since there exists little overlap between terms of queries and product descriptions (about 1.8%) [44] in our dataset, traditional retrieval models (LSI, BM25 and QLM) and LSE (originally based on product reviews instead of descriptions) from our baseline pools exhibit poor performances. For a more rigorous comparison, we also report the results of these models with augmented product descriptions by adding the terms of queries that led to clicks to the corresponding products.

[5] https://github.com/NTMC-Community/MatchZoo

Table 2: Retrieval performance. † indicates statistically significant improvement ($p < .01$) by the pairwise t-test over all the other baselines. We report performance gains in NDCG relatively to the original search results returned by the commercial search engine without re-ranking.

	Models	MRR	MAP	NDCG	Gain (%)
					Original data split
	LSI	.2223	.2117	.3536	+3.07
	BM25	.2250	.2143	.3545	+3.33
	QLM	.2422	.2305	.3670	+6.99
	LSE	.2313	.2162	.3577	+4.26
w/ augmented documents	LSI	.3787	.3575	.4618	+34.63
	BM25	.2897	.2652	.3869	+12.77
	QLM	.4743	.4449	.5173	+50.80
	LSE	.4357	.4075	.4958	+44.52
	VPCG	.4331	.4091	.4928	+43.65
	ARC-II	.4128	.3849	.4776	+39.23
	MatchPyramid	.4792	.4498	.5228	+52.40
	NeuBase	.4448	.4162	.4982	+45.22
	GEPS	**.4927**†	**.4677**†	**.5368**†	**+56.49**
					Chronological data split
	LSI	.2119	.2021	.3467	+1.37
	BM25	.2219	.2122	.3543	+3.62
	QLM	.2271	.2175	.3587	+4.89
	LSE	.2185	.2054	.3509	+2.62
w/ augmented documents	LSI	.3129	.2949	.4182	+22.30
	BM25	.2632	.2450	.3790	+10.84
	QLM	.3603	.3370	.4487	+31.22
	LSE	.3542	.3310	.4445	+29.98
	VPCG	.3571	.3391	.4505	+31.72
	ARC-II	.3388	.3165	.4350	+27.20
	MatchPyramid	.3864†	.3631†	.4692 †	+37.21
	NeuBase	.3475	.3263	.4444	+29.96
	GEPS	**.3907**†	**.3684**†	**.4753**†	**+39.00**

4.2.3 Evaluation Metrics. In our evaluation, we regard the items returned by the e-commerce search engine in the SERP for each test query as candidates, and evaluate ranking performance of different models on those candidates. The products shown in the SERP for each test query are automatically labeled based on user-specific actions by three grades of relevance, *irrelevant* (0), *somewhat relevant* (1), *very relevant* (2), which correspond to *not clicked*, *clicked*, and *clicked and purchased* products, respectively.

Three standard evaluation metrics, MRR (Mean Reciprocal Rank), MAP (Mean average precision) and NDCG (Normalized Discounted Cumulative Gain), are used for comparison. MRR measures the rank position of the first relevant (i.e., either *somewhat relevant* or *very relevant*) product which is important to attract users to keep browsing, while MAP measures the rank of all relevant products. In addition to the metrics that assume binary relevance, NDCG is used to capture multiple relevance levels, where product purchase is considered as the ultimate goal of the e-commerce search task.

4.3 As an Individual Retrieval Model

4.3.1 Retrieval Performance. To answer **RQ1** we list the results of baselines and the proposed model GEPS on the real-world testbed

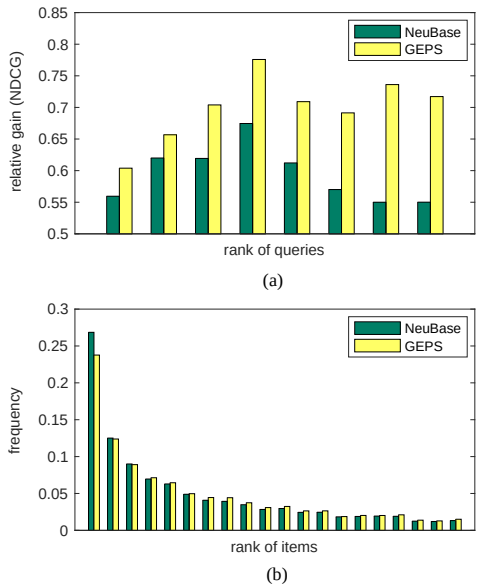


Figure 5: Retrieval performance regarding the long-tail problem of click-through data. (a) Relative gains in NDCG w.r.t. the popularity ranks of queries. (b) Frequencies of products appearing at the top-5 positions of ranking results returned by NeuBase and GEPS w.r.t. their popularity ranks.

in Table 2. As mentioned in Section 1, the traditional lexical-based models, LSI and BM25, do not perform well in the product search scenario even with augmented product descriptions by clicked queries. In contrast, neural retrieval models (i.e., ARC-II, MatchPyramid, NeuBase and GEPS) exhibit strong performances due to their advantages in semantic matching. Meanwhile, the click-graph based unsupervised model VPCG also provides a competitive baseline especially in the chronologically split dataset, which demonstrates the effectiveness of graph-based features in product search. By incorporating graph-structured data into the supervised neural retrieval model with the proposed graph embedding-based approach, GEPS can substantially outperform all the baseline methods in both data split settings, which proves the ability of GEPS to combine the advantages of graph-based models and neural network models.

To answer **RQ2**, we divide search queries into eight groups according to their occurrence frequency ranks in the click-through data as in Figure 1(a) and plot the averaged relative performance gains of NeuBase and GEPS in each group in Figure 5(a) [6]. In the last four groups where increasingly fewer training samples are provided, the performance of NeuBase starts to decline dramatically, while GEPS still remains robust. (Both of their relative gains increase in the first half of groups because the commercial search engine performs relatively better on popular queries.) This suggests that neural retrieval models indeed suffer from overfitting problems on those rare queries in the training data and our proposed graph embedding-based approach provides an effective solution.

[6]Due to limited space, we only report results in the original data split when conducting model analysis in the rest of this paper. Similar results can be obtained in the chronological data split.

Table 3: Ablation study in the original data split. *P_emb* and *Q_emb* denotes the graph embeddings of products and queries respectively, while *Heter* represents the integration of external heterogeneous information into models. We use +/− to denote cases with/without a certain feature in the model. All the improvements relative to their previous rows are statistically significant ($p < .01$) unless otherwise denoted by an asterisk.

P_emb	Q_emb	Heter	MRR	MAP	NDCG	Gain (%)
-	-	-	.4448	.4162	.4982	+45.22
+	-	-	.4529	.4275*	.5076	+47.97
+	-	+	.4640	.4387	.5160	+50.43
+	+	-	.4810	.4540	.5256	+53.22
+	+	+	.4955	.4704	.5378	+56.76

We also report the frequencies of products that appear at the top-k ($k = 5$) positions of the ranking results obtained by NeuBase and GEPS with regard to their frequencies of occurrence in the training data. Figure 5(b) shows that NeuBase tends to rank popular products higher and tail products lower than GEPS, which not only hurts search accuracy but also reinforces the *rich-get-richer* phenomenon [3] that might cause some long-term issues for e-commerce sales platforms. This further demonstrates that the proposed graph embedding-based approach can enable neural models to exploit graph-structured relations to make smoother predictions on the long-tailed products.

4.3.2 Ablation Study. To investigate **RQ3** and further answer **RQ2**, we conduct an ablation study of our proposed model. The results are shown in Table 3. We can conclude that incorporating graph embeddings of both products and queries can lead to significant performance improvement as a result of its successful treatment towards the long-tail problem of click-through data by comparing row 2 and row 4 to the baseline result row 1.

In addition, we can observe substantial performance gains when comparing row 2 and row 4 with their extended versions with external heterogeneous information in row 3 and row 5, respectively. This shows that the proposed approach is effective in taking advantage of heterogeneous information to improve search accuracy.

4.3.3 Analysis of Graph Embedding Methods. To answer **RQ4**, we analyze the performance of different graph embedding methods. As shown in Figure 6(a), Deepwalk performs slightly better than LINE with both embedding vectors preserving the first order proximity, the second order proximity, and their combination. This difference can be seen as a trade-off between efficacy and efficiency, for LINE runs relatively faster than Deepwalk as shown in Figure 6(b) thanks to its edge-sampling strategy (although they both have linear time complexity). Besides, it can lead to another marginal gain to concatenate embedding vectors obtained from Deepwalk with the *first order* embedding vectors obtained from LINE, because Deepwalk only considers the *second order* proximity to which the latter are complementary.

We also investigate how embedding dimensions influence the final performance. Overall, our proposed approach is quite robust to

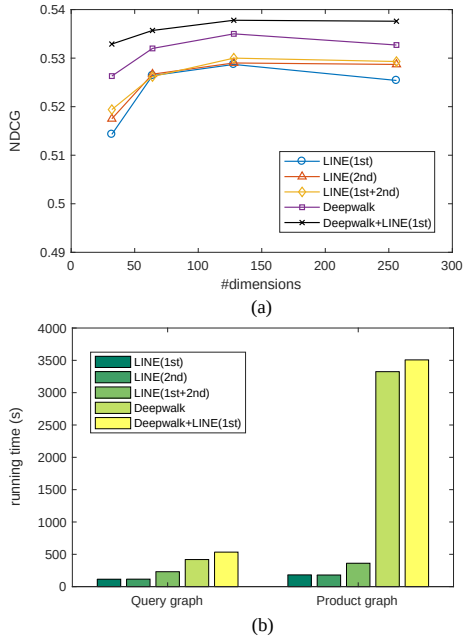


Figure 6: Performance of the graph embedding methods w.r.t. different number of dimensions $K \in [32, 64, 128, 256]$.

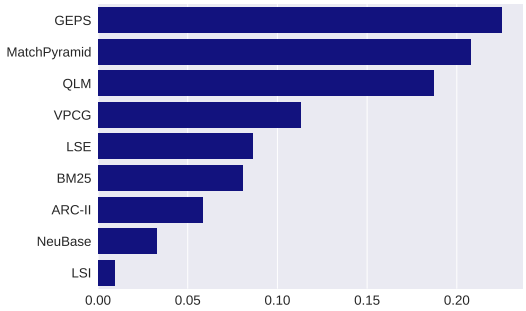


Figure 7: Importance scores of features used in GBDT.

different graph embedding methods with different embedding sizes (Figure 6(a)). Throughout this paper, we use "Deepwalk + LINE (1st)" with $K = 128$ as the default setting.

4.4 As a Feature for Learning-to-rank Models

We continue to answer **RQ1** by evaluating the performance of GEPS as a feature in a learning-to-rank framework implemented with GBDT [10]. We choose seven handcrafted features motivated by Wu et al. [44] as our basic feature pool *Base*: (0) original ranks (which might contain some unreleased information used by the commercial search engine), (1) total click counts, (2) total view counts, (3) total purchase counts, (4) click-through rates, (5) the word lengths of the description of each product, (6) the logarithms of product price.

We can draw the following conclusions from the results shown in Table 4. First, GEPS is such a powerful feature that we can observe

Table 4: Retrieval performance as a feature in the learning-to-rank framework.

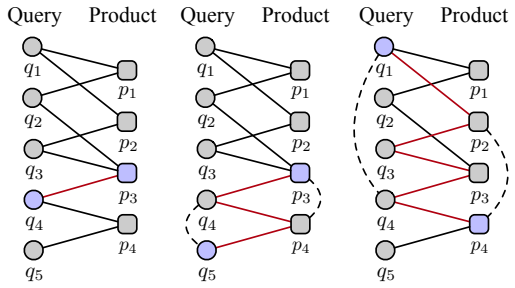
Models	MRR	MAP	NDCG	Gain (%)
			Original data split	
Base	.4249	.3973	.4876	+42.14
Base+QLM	.4878	.4568	.5266	+53.52
Base+LSE	.4450	.4160	.5000	+45.75
Base+VPCG	.4414	.4178	.4992	+45.50
Base+MatchPyramid	.4880	.4579	.5291	+54.24
Base+GEPS	**.4991**	**.4717**	**.5381**	**+56.87**
Base+QLM+LSE	.4828	.4525	.5227	+52.36
Base+LSE+MatchPyramid	.4976	.4673	.5359	+56.22
Base+QLM+MatchPyramid	.5113	.4791	.5434	+58.40
Base+LSE+GEPS	.5021	.4744	.5402	+57.46
Base+MatchPyramid+GEPS	.5120	.4827	.5459	+59.14
Base+QLM+GEPS	**.5207**	**.4913**	**.5520**	**+60.91**
Base+QLM+MatchPyramid+VPCG	.5124	.4819	.5454	+58.97
Base+QLM+LSE+MatchPyramid	.5162	.4843	.5475	+59.60
Base+QLM+LSE+GEPS	.5218	.4919	.5524	+61.03
Base+QLM+MatchPyramid+GEPS	**.5289**	**.4986**	**.5573**	**+62.45**

the largest performance improvement by adding GEPS to the feature pool. Second, GEPS is also complementary to other competitive retrieval approaches, especially the traditional language-modeling based approach *QLM* and the recently proposed neural architecture *MatchPyramid*; combining GEPS with QLM and MatchPyramid leads to a substantial improvement by 3.6% in NDCG over the best single model. However, it is interesting that we do not observe a significant improvement after combining GEPS with those features in the basic feature pool, possibly because most of the useful basic features can be automatically extracted from click-through data by GEPS. In addition, in Figure 7, we can see that GEPS is the most important feature when ensembling all the baselines with GBDT.

5 FURTHER DISCUSSION: RETRIEVAL AS LINK PREDICTION AND TRANSITIVITY

The goal of product search, especially from the perspective of neural network-based black box approaches, is to predict whether a given query-product pair (p, q) is relevant (and then rank all products according to the probability). In fact, this goal is exactly equivalent to predict whether a *link* exists in the bipartite click graph of queries and products as shown in Figure 8(a).

Nonetheless, the click graph used in the training stage can never be complete; there are inevitably many missing links partly because we can only access a limited sample of search logs and also because, given a query, some of the relevant products are not presented to customers by the search engine at all (though, this does not mean those products have no user impressions within other queries either). The good part is that we can obtain more information beyond those direct links thanks to the *transitivity* of relevance relations. To be more specific, products that are several hops from a given query (e.g., q_5 versus p_3 in Figure 8(b) and q_1 versus p_4 in Figure 8(c)) are more likely to be relevant than a totally random product. However, neural network models are not well aware of



(a) 1-hop (b) 3-hop (c) 5-hop

Figure 8: (a) Retrieval problems can be regarded as link prediction. (b-c) Products within several hops (e.g., 3 and 5 hops away) to a certain query tend to be relevant as well. In GEPS, such transitive relations are regularized by similar graph embeddings indicated by the dashed lines.

such transitivity although the one-hop relations can be perfectly maintained due to their strong expressive power.

In our proposed approach, we can effectively capture transitivity with graph embedding techniques, where correlations between queries and products are well preserved. For instance, in Figure 8(b), q_4 and q_5 as well as p_3 and p_4 have similar graph embeddings (i.e., $e_{q_4} \approx e_{q_5}$ and $e_{p_3} \approx e_{q_4}$) since they are close neighbors (i.e., the *first order proximity*) in the query graph and the product graph respectively, and therefore p_3 is considered as a relevant item to query q_5 because both features $[e_{q_4}, e_{p_3}]$ and $[e_{q_5}, e_{p_4}]$ indicate positive links in the click graph. It is the same case with 5-hop pair (q_1, p_4) in Figure 8(c) where q_1 and q_4 as well as p_2 and p_4 have similar neighbors (i.e., the *second order proximity*) and hence are also close to each other in the graph embedding space.

To empirically prove this point, we plot the distributions of relevance scores given by NeuBase, the proposed model GEPS and MatchPyramid for the pairs of queries and products that are *1 hop*, *3 hops* and *5 hops* away in the click graph along with *random* pairs in Figure 9. In the case of NeuBase, we can observe that the relevance scores for 3-hop and 5-hop query-product pairs drop quickly and their distributions are very similar to that of random pairs. On the contrary, GEPS can give smoothly decreasing scores as queries and products are farther away from each other in the click graph; the relevance score distribution for 3-hop query-product pairs is closer to that for 1-hop pairs, while the distribution for 5-hop pairs is somewhere between that for 3-hop pairs and that for random pairs. Table 5 supports our argument in a more rigorous way; without such proper regularization as graph structures, neural networks are likely to fit non-smooth relations without transitivity (e.g., 3-hop and 5-hop cases) as exactly appeared in the training data (i.e., 1-hop query-product pairs). To conclude, it is the ability to discriminate among those decreasingly relevant query-product pairs (3-hop, 5-hop and so on relatively to totally random) induced by transitivity that enables better ranking performance for GEPS.

Also shown in Figure 9 and Table 5, instead of stratifying query-product pairs of different distances in the click graph, MatchPyramid discriminates positive samples (i.e., 1-hop query-product pairs)

Table 5: AUC results of separating the pairs of queries and products within k hop(s) ($k \in [1, 3, 5]$) from farther pairs with relevance scores given by NeuBase and GEPS respectively .

Model	within 1 hop	within 3 hops	within 5 hops
NeuBase	.833	.616	.627
GEPS	.824	.744	.749
MatchPyramid	.726	.664	.647

in a more fine-grained manner. This suggests that MatchPyramid tackles the issues of neural networks from a different perspective from graph-based approaches (actually by focusing on the semantic interaction between query terms and document terms with a properly designed neural architecture). This is also consistent with the fact that MatchPyramid and GEPS can substantially boost each other's performance as features in the learning-to-rank model as shown in Table 4 and Figure 7 (Section 4.4).

Furthermore, such transitive relevance relations can also be induced from more complicated graphs than click graphs, for example, the ones shown in Figure 4. This is also why it can lead to significant improvement when incorporating external heterogeneous information with our graph-embedding approach.

6 RELATED WORK

We review the related work of product search, neural models for IR and graph-based models for IR along with a brief comment on neural models with graph-structured data in a more general context.

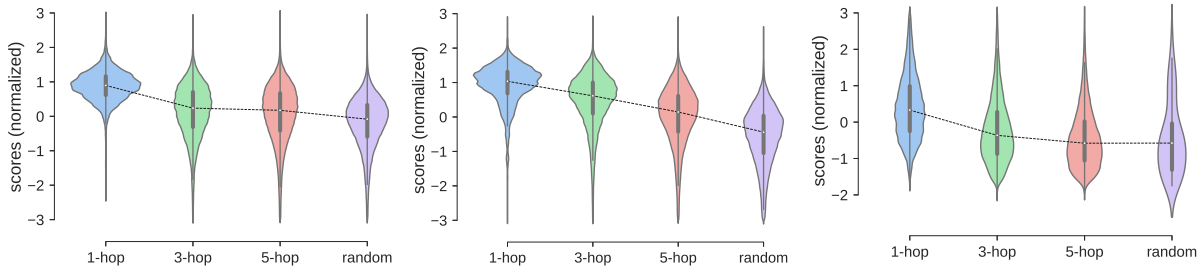
6.1 Product Search

Product search has been attracting extensive research efforts. Duan et al. (2013) [8] propose a probabilistic generative model to analyze e-commerce search log for useful knowledge. Duan et al. (2015) [7] study the problem of learning query intent representation for product search and recommendation with a proposed language model. Karmaker Santu et al. [19] discuss issues and strategies when applying learning-to-rank models to the product search scenario. Also, Li et al. [22] propose an economic theory-based product search model to complement those product search models mainly adapted from the information retrieval literature.

Recently, Van Gysel et al. [42] propose a latent vector space model (LSE) to jointly learn distributional representations of words and products for better product retrieval performance. Ai et al. [1] exploit a hierarchical embedding model (HEM) that jointly learns semantic representations for various entities, including terms, products, users and queries, to tackle the problem of personalized search. These two embedding models largely inspire our work although neither of them leverages graph embedding techniques to encode rich graph-structured relationships in their embeddings.

6.2 Neural Information Retrieval

Our work is in the context of recent advances in neural information retrieval. Due to limited space, only some of the recent work is introduced here, and we refer the reader to the paper [27] by Mitra and Craswell for a comprehensive review. Huang et al. [16]



(a) NeuBase (b) GEPS (c) MatchPyramid

Figure 9: The distributions of relevance scores given to the pairs of queries and products that are k hops way from each other and given to random query-product pairs.

propose a deep structured semantic model (DSSM) to learn semantic representations of queries and documents with large-scale click-through data for web search, while Guo et al. [13] propose an interaction-focused deep relevance matching model (DRMM) that uses histogram-based features to capture lexical relevance between queries and documents for better retrieval performance. Mitra et al. [28] then attempt to combine those two approaches with a hybrid deep model Duet composed by a local model for exact term matching and a distributed model for semantic matching. Recently, Liu et al. [24] introduces knowledge graphs to neural search systems, which, however, is different from the intrinsic problem of transitive relevance in click-through data we study in this paper. In fact, our contribution is orthogonal to this line of research and therefore can be combined with the above mentioned neural retrieval approaches.

6.3 Graph-based Information Retrieval

Graph-based features are widely used and studied in the IR literature [30], e.g., Pagerank [31], HITS [21] and Simrank [17]. Yet, our work is more related to the following. Craswell and Szummer [4] use random walks on the click graph to produce probabilistic rankings for image search. Gao et al. [11] propose a random walk-based smoothing approach to overcome the sparsity problem of click-through data in web search. Click graphs are also used to learn semantic representations of queries and documents (e.g., [18, 45]) and to learn query intent (e.g., [23, 25]). Ren et al. [35] propose to use a heterogeneous graph that consists of queries, web pages, and Wikipedia concepts to improve search performance.

Graph-based IR approaches can depend on global information to make local decisions and hence are rather robust to sparse, complex and incomplete data. However, most of them also suffer from disadvantages as unsupervised methods. While combining them into a learning-to-rank framework as features is one possible way to overcome this limitation, it also motivates us to directly tackle this problem at the model level to combine both the advantages of unsupervised graph-based models and supervised neural models.

6.4 Neural Models with Graph-structured Data

In a more general context, GCN (Graph Convolutional Network) [6, 20] provides an alternative solution to incorporate graph-structured data into neural networks by generalizing the traditional convolutional filters from images to graphs. Berg et al. recently propose a model called GC-MC (Graph Convolutional Matrix Completion) that attempts to apply GCN to recommender systems in an unpublished preprint [2]. Monti et al. [29] later extend GC-MC to multi-graph cases and obtain promising results.

The proposed approach and GCN can both be seen as ways to regularize neural models with graph structures. The former uses an auxiliary task of reconstructing graph embeddings to achieve such goal, whereas GCN directly conditions the output of neural network on the adjacency matrix (i.e., $f(\cdot|A)$). Due to the latter fact, each prediction depends on the whole graph (or, at least, a local subgraph as approximation [46]) causing GCN not very scalable to such large graphs as in real-world search scenarios. On the contrary, graph embeddings can serve as an "intermediate" which decouples neural nets and graphs *per se* to reduce computational burden while still allowing neural nets to utilize graph structure information.

7 CONCLUSION

This paper introduces a Graph Embedding-based ranking approach for Product Search (GEPS) which leverages recent advances in graph embedding techniques to incorporate graph-structured data into neural models in Section 3. Through extensive experiments in Section 4, we demonstrate that our proposed approach can successfully combine both the advantages of graph-based retrieval models and neural retrieval models. To be more specific, experimental results show that GEPS enables neural models to effectively deal with the long-tail sparsity problem and combine heterogeneous external information to improve search accuracy. Finally, we point out the equivalence between retrieval tasks and link prediction tasks, and then discuss the advantage of GEPS in capturing relevance transitive relations on graphs in Section 5.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments and helpful suggestions. This work is supported by NSFC under Grant No. 61532001, MOE-ChinaMobile Program under Grant No. MCM20170503, and 973 Program under Grant No. 2014CB340405.

REFERENCES

[1] Qingyao Ai, Yongfeng Zhang, Keping Bi, Xu Chen, and W Bruce Croft. 2017. Learning a hierarchical embedding model for personalized product search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 645–654.

[2] Rianne Van Den Berg, Thomas N Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *arXiv preprint arXiv:1706.02263* (2017).

[3] Junghoo Cho and Sourashis Roy. 2004. Impact of Search Engines on Page Popularity. In *Proceedings of the 13th International Conference on World Wide Web*. ACM, New York, NY, USA, 20–29.

[4] Nick Craswell and Martin Szummer. 2007. Random walks on the click graph. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 239–246.

[5] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science* 41, 6 (1990), 391.

[6] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Advances in Neural Information Processing Systems 29*. 3844–3852.

[7] Huizhong Duan and ChengXiang Zhai. 2015. Mining Coordinated Intent Representation for Entity Search and Recommendation. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, New York, NY, USA, 333–342.

[8] Huizhong Duan, ChengXiang Zhai, Jinxing Cheng, and Abhishek Gattani. 2013. A probabilistic mixture model for mining and analyzing product search log. In *Proceedings of the 22nd ACM international conference on Conference on information knowledge management*. ACM, New York, NY, USA, 2179–2188.

[9] Yixing Fan, Liang Pang, JianPeng Hou, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2017. MatchZoo: A Toolkit for Deep Text Matching. *arXiv preprint arXiv:1707.07270* (2017).

[10] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[11] Jianfeng Gao, Wei Yuan, Xiao Li, Kefeng Deng, and Jian-Yun Nie. 2009. Smoothing clickthrough data for web search ranking. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 355–362.

[12] Palash Goyal and Emilio Ferrara. 2017. Graph Embedding Techniques, Applications, and Performance: A Survey. *CoRR* abs/1705.02801 (2017). arXiv:1705.02801

[13] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 55–64.

[14] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation Learning on Graphs: Methods and Applications. *IEEE Data Engineering Bulletin* (2017).

[15] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*. 2042–2050.

[16] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information and knowledge management*. ACM, 2333–2338.

[17] Glen Jeh and Jennifer Widom. 2002. SimRank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 538–543.

[18] Shan Jiang, Yuening Hu, Changsung Kang, Tim Daly Jr, Dawei Yin, Yi Chang, and Chengxiang Zhai. 2016. Learning Query and Document Relevance from a Web-scale Click Graph. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 185–194.

[19] Shubhra Kanti Karmaker Santu, Parikshit Sondhi, and ChengXiang Zhai. 2017. On Application of Learning to Rank for E-Commerce Search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, 475–484.

[20] Thomas N Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks.

[21] Jon M Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)* 46, 5 (1999), 604–632.

[22] Beibei Li, Anindya Ghose, and Panagiotis G. Ipeirotis. 2011. Towards a Theory Model for Product Search. In *Proceedings of the 20th International Conference on World Wide Web*. ACM, New York, NY, USA, 327–336.

[23] Xiao Li, Ye-Yi Wang, and Alex Acero. 2008. Learning query intent from regularized click graphs. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 339–346.

[24] Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2018. Entity-Duet Neural Ranking: Understanding the Role of Knowledge Graph Semantics in Neural Information Retrieval. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2395–2405.

[25] Hao Ma, Haixuan Yang, Irwin King, and Michael R Lyu. 2008. Learning latent semantic relations from clickthrough data for query suggestion. In *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 709–718.

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[27] Bhaskar Mitra and Nick Craswell. 2017. Neural Models for Information Retrieval. *CoRR* abs/1705.01509 (2017). arXiv:1705.01509

[28] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to Match using Local and Distributed Representations of Text for Web Search. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1291–1299.

[29] Federico Monti, Michael Bronstein, and Xavier Bresson. 2017. Geometric matrix completion with recurrent multi-graph neural networks. In *Advances in Neural Information Processing Systems*. 3697–3707.

[30] R Mota, C Ramalho, J Monteiro, J Correia-Pinto, M Rodrigues, H GuimarÃčes, J Spratley, F Macedo, A Matias, and N Montenegro. 2011. *Graph-based natural language processing and information retrieval*. Cambridge University Press.

[31] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web*. Technical Report. Stanford.

[32] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition.. In *AAAI âĂŹ16*. 2793–2799.

[33] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[34] Yuval Pinter, Robert Guthrie, and Jacob Eisenstein. 2017. Mimicking Word Embeddings using Subword RNNs. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 102–112.

[35] Xiang Ren, Yujing Wang, Xiao Yu, Jun Yan, Zheng Chen, and Jiawei Han. 2014. Heterogeneous graph-based intent learning with queries, web pages and wikipedia concepts. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 23–32.

[36] Jason D. M. Rennie. 2005. Loss functions for preference levels: Regression with discrete ordered labels. In *Proceedings of the IJCAI Multidisciplinary Workshop on Advances in Preference Handling*. 180–186.

[37] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (April 2009), 333–389.

[38] Jennifer Rowley. 2000. Product search in e-shopping: a review and research propositions. *Journal of consumer marketing* 17, 1 (2000), 20–35.

[39] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 373–374.

[40] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. In *Proceedings of the VLDB Endowment, Vol. 4, No. 11*.

[41] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 1067–1077.

[42] Christophe Van Gysel, Maarten de Rijke, and Evangelos Kanoulas. 2016. Learning latent vector spaces for product search. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 165–174.

[43] Christophe Van Gysel, Maarten de Rijke, and Evangelos Kanoulas. 2017. Semantic Entity Retrieval Toolkit. In *SIGIR 2017 Workshop on Neural Information Retrieval (Neu-IR'17)*.

[44] Chen Wu, Ming Yan, and Luo Si. 2017. Ensemble Methods for Personalized E-Commerce Search Challenge at CIKM Cup 2016. *CoRR* abs/1708.04479 (2017).

[45] Wei Wu, Hang Li, and Jun Xu. 2013. Learning query and document similarities from click-through bipartite graph with metadata. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 687–696.

[46] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[47] Hamed Zamani and W Bruce Croft. 2017. Relevance-based word embedding. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 505–514.

[48] Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)* 22, 2 (2004), 179–214.

Neural Multimodal Belief Tracker with Adaptive Attention for Dialogue Systems

Zheng Zhang[†]
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems
Beijing National Research Center for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
zhangz.goal@gmail.com

Minlie Huang[*]
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems
Beijing National Research Center for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
aihuang@tsinghua.edu.cn

Lizi Liao[†]
National University of Singapore
liaolizi.llz@gmail.com

Xiaoyan Zhu
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems
Beijing National Research Center for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
zxy-dcs@tsinghua.edu.cn

Tat-Seng Chua
National University of Singapore
chuats@comp.nus.edu.sg

ABSTRACT

Multimodal dialogue systems are attracting increasing attention with a more natural and informative way for human-computer interaction. As one of its core components, the belief tracker estimates the user's goal at each step of the dialogue and provides a direct way to validate the ability of dialogue understanding. However, existing studies on belief trackers are largely limited to textual modality, which cannot be easily extended to capture the rich semantics in multimodal systems such as those with product images. For example, in fashion domain, the visual appearance of clothes play a crucial role in understanding the user's intention. In this case, the existing belief trackers may fail to generate accurate belief states for a multimodal dialogue system.

In this paper, we present the first neural multimodal belief tracker (**NMBT**) to demonstrate how multimodal evidence can facilitate semantic understanding and dialogue state tracking. Given the multimodal inputs, while applying a textual encoder to represent textual utterances, the model gives special consideration to the semantics revealed in visual modality. It learns concept level fashion semantics by delving deep into image sub-regions and integrating concept probabilities via multiple instance learning. Then in each turn, an adaptive attention mechanism learns to automatically emphasize on different evidence sources of both visual and textual modalities for more accurate dialogue state prediction. We perform extensive evaluation on a multi-turn task-oriented dialogue dataset in fashion domain and the results show that our method achieves superior performance as compared to a wide range of baselines.

CCS CONCEPTS

• **Computing methodologies** → **Discourse, dialogue and pragmatics**; • **Information systems** → *Information retrieval*; *Multimedia and multimodal retrieval*; Users and interactive retrieval.

ACM Reference Format:
Zheng Zhang[†], Minlie Huang, Lizi Liao, Xiaoyan Zhu, and Tat-Seng Chua. 2019. Neural Multimodal Belief Tracker with Adaptive Attention for Dialogue Systems. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313598

1 INTRODUCTION

By offering a natural and interactive way to satisfy user's information need, multimodal dialogue systems [34] have attracted more and more attention recently. Compared to traditional text-based systems, multimodal dialogue systems enable users to easily provide

[*]Corresponding author.
[†]These two authors contributed equally.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313598

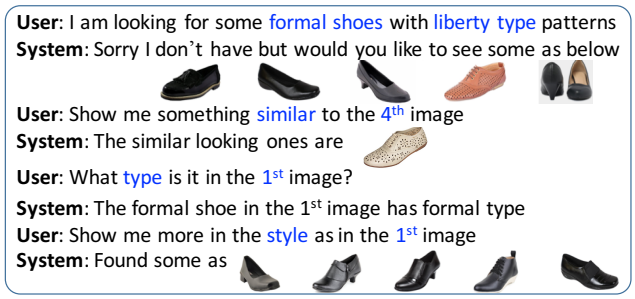


Figure 1: An illustrative example of multimodal dialogue for fashion retail, which demonstrates the importance of visual modality in understanding the inherent semantics. Keywords in blue correspond to images, where both modalities are crucial for dialogue state tracking.

an image sample instead of racking their minds for an appropriate text description, such as in search of fashion products. At the same time, it is more straight-forward for users to perceive information from system provided images rather than text based on supposition. In task oriented scenarios, multimodal dialogue systems can better help users achieve their goals such as finding specific fashion products or travel sights under the help of visual modality and has achieved superior performance [26].

Efficient operation of such dialogue systems requires a core component — belief tracker (or known as dialogue state tracking) that can track what has happened by modeling system outputs, user utterances, and context from previous turns *etc.* A belief tracker provides a direct way to validate the systems' understanding of user's goal at each step of the dialogue. At the same time, the output of the belief tracker also supports the downstream dialogue policy component to decide what action the system should take next. Such output offers an explicit way to evaluate whether the learned policy is reasonable. Therefore, the belief tracker component is essential for the final performance of a complete dialogue system.

However, since current dialogue systems are largely confined to textual modality, existing efforts on belief tracker are also limited primarily to text-based methods. Thus, it may fail to generate accurate belief states for multimodal systems, as the rich semantics inherent in visual modality are ignored. For example, in order to generate correct belief states for each turn of the dialogue as shown in Figure 1, the system needs to understand the visual features of the fourth image, the type and style of the first image, *etc.* Purely relying on text, the system would fail to obtain these useful semantics, and thus result in inaccurate belief states. Moreover, it may further affect the performance of other downstream dialogue system components. Therefore, the primary goal of our work is to build a multimodal belief tracker which is able to accurately understand multimodal evidence and adaptively integrate these modalities for dialogue state tracking.

Although we have shown that multimodal evidence is crucial for dialogue state tracking, it is non-trivial to correctly extract useful semantics and leverage them appropriately in dialogue state tracking. **First**, there exists the well-known semantic gap between the low-level visual cues and the high-level semantics (*e.g., style, fit*). Indeed, there are many studies focusing on the Vision-to-Language problems such as image captioning [27], visual storytelling [18] and visual question answering (VQA) [2], which have achieved good performance. Still, as pointed out in [10, 45], there exist strong language priors which lead to good superficial performance of these models without truly understanding the visual content. For example, for questions starting with "Do you see a ..." in VQA dataset, blindly answering "yes" can achieve 87% accuracy. **Second**, the heterogeneity between the visual and textual modalities makes it hard to incorporate them well to generate concrete belief states. Due to the difference in expressiveness, users tend to prefer image for illustrating certain concepts (*such as style, pattern*) while use text for describing others (*such as category, material*). Finding out these patterns of user behavior is essential for improving dialogue state tracking.

In this paper, we propose a neural multimodal belief tracker (**NMBT**) (see Figure 3) and apply it for semantic understanding across modalities and user intent tracking. First, to fill the semantic gap and avoid being misled by language priors, we develop a visual concept learner in a weakly-supervised manner by leveraging multiple instance learning [5] which reasons with sub-regions rather than the full image. The visual concept learner enables the tracker to understand fine-grained semantics of product images at the concept level. Second, to model user's behavior patterns regarding different modalities, the tracker employs an attention mechanism to adaptively attend to images and texts during dialogue state tracking at each turn. Conditioned on both textual contexts and visual contents observed so far, the model automatically learns to emphasize the evidence coming from user provided image, system provided image, or the text utterances for more accurate dialogue state tracking.

To the best of our knowledge, this is the first work for dialogue state tracking or belief tracking in multimodal dialogue systems. The main contributions for this work are as follows:

- We validate the importance of integrating multimodal evidence in dialogue state tracking and identify the critical challenges in understanding as well as leveraging such evidence.
- We delve into image sub-regions to learn concept level visual semantics and propose an adaptive attention mechanism for automatically deciding the evidence source for dialogue state tracking based on multimodal dialogue context.
- We conduct extensive experiments to evaluate the proposed method in various evaluation metrics and show superior performance over state-of-the-art methods.

2 RELATED WORK

2.1 Text-based Dialogue State Tracking

Since spoken interaction promises a natural, effective, and hands- and eyes-free method for human-computer interaction, together with the progress in natural language processing, dialogue systems have been mainly developed within textual modality. In this paper we focus on the dialogue state tracking (DST) task in task-oriented dialogue systems. Here, we summarize recent work on DST and discuss the major difference of our work.

Early dialogue systems used hand-crafted rules for DST, keeping track of a single top hypothesis for each slot of the belief state

[23, 46]. Such systems require no training data and allow developers to incorporate domain knowledge to boost performance. However, such methods fail to make use of the entire N-best hypothesis list, thus do not account for uncertainty in a principled way. In addition, uncertainty also arises from the inherent ambiguity of natural language.

Therefore, statistical DST methods are introduced to better solve the uncertainty problem. Generally speaking, the statistical methods can be categorized into two types, namely generative and discriminative approaches [12]. Generative approaches maintain a distribution over dialogue states, which is calculated by Bayes rules based on the history of observations (NLU results and previous system action) in each turn. Early generative approaches attempt to formalize the dialogue system as a Markov decision process (MDP) [24, 25, 44]. However, MDP models have an assumption that the state is observable, which cannot account for the uncertainty in either dialogue state or user act parsed by NLU. Therefore, [43] further proposed POMDP-based dialogue model, treating the dialogue state as hidden variable which can be inferred from system observations, and achieved better performance. However, since all possible states are enumerated in the above methods, it can be intractable when the state space is very large [12]. Also, as pointed out by [40], generative approaches must model all the correlations in the input features, so they cannot easily exploit arbitrary but potentially useful features.

Discriminative approches have the key benefits that they can incorporate a large number of features, and can be optimized directly for prediction accuracy. [3] proposed the first discriminative state tracking trained from data where features were taken from spoken language understanding (SLU) output and dialogue history. Subsequent work has explored numerous variations of this approach. For instance, [41] applied a ranking algorithm which has the ability to construct conjunctions of features. [15] applied a deep neural network as a classifier. More recently, [16] proposed an RNN-based DST to directly take automatic speech recognition results as input without extra semantic understanding and obtains better performance than those which only utilizes semantic features produced by external NLU. [30] proposed a CNN-based method to extract semantic features from raw ASR results and passed it to DST.

As observed in the dialogue state tracking challenges (DSTC), discriminative methods tend to dominate all other approaches [39]. Our work is also based on the discriminative statistical DST framework. However, existing approaches are constrained within textual modality while ignores the rich semantics inherent in visual images. In the emerging mutimodal dialogue systems, this information becomes essential for performance improvement. We thus take a step further towards understanding and adaptively using such multimodal evidence.

2.2 Multimodal Understanding

Another line of work related to ours lies in multimodal understanding, which focuses on recognizing inherent semantics of multimodal data [1] and exploits the relevance between different modalities [2, 7, 27]. In this work, we focus on understanding concept-level semantics within multimodal data, which relates to research of visual-semantic embedding, image captioning and visual question-answering (VQA).

With the aim of learning a mapping from images into a semantic space, visual-semantic embedding have shown to be effective in image-text ranking and zero-shot learning. There are some embedding models based on Canonical Correlation Analysis (CCA) [11] which learns a linear projection to maximize the correlation between two modalities [8, 9]. Kernel CCA [22] is further employed to extend to nolinear projection. Nevertheless, as point out in [38], scaling CCA to large amounts of data can be difficult. Another line of efforts train a joint embedding model with ranking loss. [7] learns linear transformation between visual and textual features with a single-directional ranking loss, which applies penalty to incorrect sentences ranked higher than correct ones. Bi-direction ranking loss is employed to boost the performance by further ensuring the correct image described by a sentence ranked higher than other images [19–21]. However, these works cannot be directly utilized by our work due to the difference in nature between our dialogue state tracking task and other tasks. In the above tasks, the semantics of image and text are to be aligned, while in the multimodal dialogue problem, the semantics of user intent come from either visual supplied evidence or textual inputs. Though certain semantics are presented by both modality, many of them only reside in one modality due to the concise nature of dialogues.

For image captioning, most existing methods adopt an encoder-decoder framework, which consists of a CNN visual encoder along with an RNN language decoder [27, 37]. The CNN encoder extracts visual feature from the image and feed it to the RNN decoder to generate a natural language text. Inspired by the advances in neural machine translation, some works further introduced the attention mechanism, which attends to different sub-regions of an image during decoding [42].

Compared to image captioning, VQA [2] shares a more similar task setting with multimodal dialogue, since it involves single-turn interaction through question and answer. VQA also utilized the encoder-decoder framework while the encoder side includes both image and question [2, 32]. The work of visual dialogue [4, 29] further handles multi-turn QA pairs as multimodal dialogue does. Even so, as pointed out in [29], these works should be categorized into image-grounded QA rather than multimodal dialogue. The reason lays on the way they utilize images. In VQA tasks, only one single image is involved and the conversation is centered on it. While in multimodal dialogue tasks, the model has to process multiple images which acts as supporting evidence.

However, as pointed out in [10, 45], most existing Vision-to-Language task performance should be attributed to the language prior and the models do not truly understand the images. While in our scenario, accurate understanding of visual image is rather important for intention inference. Therefore, we propose to extract explicit semantic concepts from image and feed to downstream dialogue tracker.

Recently, [33] contributed a large-scale benchmark dataset with 150K dialogue sessions and proposed two end-to-end models based on the hierarchical recurrent encoder decoder (HRED) framework [35] as baselines for response generation. We use their dataset while focus on the multimodal dialogue state tracking task.

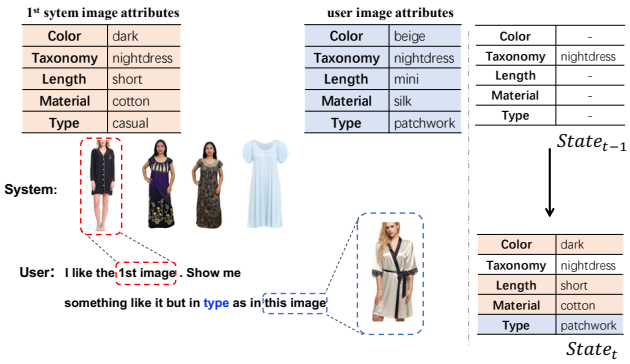


Figure 2: An example turn in multimodal dialogue. The model extracts semantics from both visual and textual input to update the dialogue state to $State_t$. Slots "Color", "Length" and "Material" are updated by understanding the system provided image. While "Type" is updated based on both textual and visual evidence from the user provided image.

3 MODEL

3.1 Task Definition and Model Overview

Our model is designed to tackle the problem of dialogue state tracking (DST) which maintains the belief state $\mathbf{S}_t$ in each turn during the dialogue flow. Different from traditional text-based task, we need to extract semantic concepts from both textual and visual modalities. An example is shown in Figure 2. Note that the slot values of a state are derived from not only the textual evidence, but also system provided and user provided visual evidences.

Suppose $T_t^{'}, V_t^{'}$ refer to the textual and visual parts of the system response in turn t, while T_t, V_t are those of the subsequent user posts, the task of multimodal dialogue state tracking can be stated as follows: in the t-th turn, given the dialogue context with messages from both user and agent sides $X_t = \{T_1^{'}, V_1^{'}, T_1, V_1, ..., T_t^{'}, V_t^{'}, T_t, V_t\}$, the model is supposed to give an estimation of the belief state $\mathbf{p}_t^k$ for each slot k:

$$\mathbf{p}_t^k = f^k(X_t), \tag{1}$$

where f^k represents the belief tracker for slot k, $\mathbf{p}_t^k$ is a $N_k + 1$ dimensional vector[1] which is a probability distribution over the values of slot k. The belief state $\mathbf{S}_t$ in turn t is defined as a collection of distributions over the values of all the N_s slots:

$$\mathbf{S}_t = [\mathbf{p}_t^1, \mathbf{p}_t^2, ..., \mathbf{p}_t^{N_s}]. \tag{2}$$

As described above, both the system and user messages may consist of a textual and a visual part where the textual part is a natural language utterance and the visual part refers to some images. It is worth noting that in our work, the visual part of system response $V_t^{'}$ usually contains zero or multiple images, while the user post V_t contains zero or only one image.

Specifically, our proposed model consists of three major parts:

- The basic textual network learns representations of the textual evidences by taking only the textual utterances from both system and user sides as inputs:

$$\mathbf{h}_t = f_{textual}(T_1^{'}, T_1, T_2^{'}, T_2, ..., T_t^{'}, T_t). \tag{3}$$

- The sub-region based visual concept learning part extracts concept level visual representation $\mathbf{v}$ for each image I. Given the image, it first embeds sub-regions and then maps to concepts under the multiple instance learning scenario:

$$\mathbf{v} = f_{visual}(I). \tag{4}$$

- The adaptive modality attention part predicts the belief state by integrating the textual and visual representations:

$$\mathbf{p}_t = f_{attn}(\mathbf{v}_t^{'}, \mathbf{v}_t, \mathbf{r}_t, \mathbf{h}_t), \tag{5}$$

where $\mathbf{v}_t^{'}$ and $\mathbf{v}_t$ denote the visual representation of system and user images respectively, $\mathbf{r}_t$ refers to the textual representation of user text T_t. The model learns to automatically emphasize on different sources of evidences for dialogue state tracking.

In the following part, we will explain the three parts in more detail. Since all the slots share a common model structure, for the convinience of description, we omit the slot subscript k in the following subsections and only describe the model architecture for a single slot.

3.2 Basic Textual Framework

We first introduce a basic textual framework upon which we will build our multimodal model. The framework is an hierarchical RNN model, which consists of a word-level encoder and an utterance-level encoder. In this framework, the textual inputs in the t-th turn are T_t and $T_t^{'}$. Suppose $\mathbf{T}_t$ and $\mathbf{T}_t^{'}$ are the embedding matrices obtained via one hot vectors of words in user post T_t and system response $T_t^{'}$ multiplying with the pre-trained word embedding matrix respectively. Under such processing, we actually represent each word in its semantic vector form, namely word vectors. We then employ a word-level RNN encoder, which takes as input a word vector at each time step, to learn the integrated semantic representations of T_t and $T_t^{'}$ respectively as follows:

$$\mathbf{r}_t = RNN_1(\mathbf{T}_t), \tag{6}$$

$$\mathbf{r}_t^{'} = RNN_1(\mathbf{T}_t^{'}), \tag{7}$$

where the word-level encoder is denoted as RNN_1. The concatenation of $\mathbf{r}_t$ and $\mathbf{r}_t^{'}$ is then fed into an utterance-level encoder, which is denoted as RNN_2. Therefore, we have

$$\mathbf{h}_t = RNN_2_CELL(\mathbf{h}_{t-1}, [\mathbf{r}_t, \mathbf{r}_t^{'}]), \tag{8}$$

where $\mathbf{h_t}$ is the output of RNN cell, and [,] denotes concatenation of vectors.

Note that $\mathbf{h_t}$ is the summary of information observed so far in turn t. In pure text-based systems, they usually directly fed $\mathbf{h_t}$ into a fully-connected layer followed by a softmax function to generate the probability distribution over the values of the slot:

$$\mathbf{g}_t = softmax(FC(\mathbf{h}_t)). \tag{9}$$

The dimension of the fully-connected layer is $d + 1$, where d is the number of possible slot values for this specific slot. The extra one dimension represents that this slot is not mentioned yet. As

[1] N_k is the number of values of slot k, while the extra one dimension represents that slot k is not mentioned in the dialogue yet.

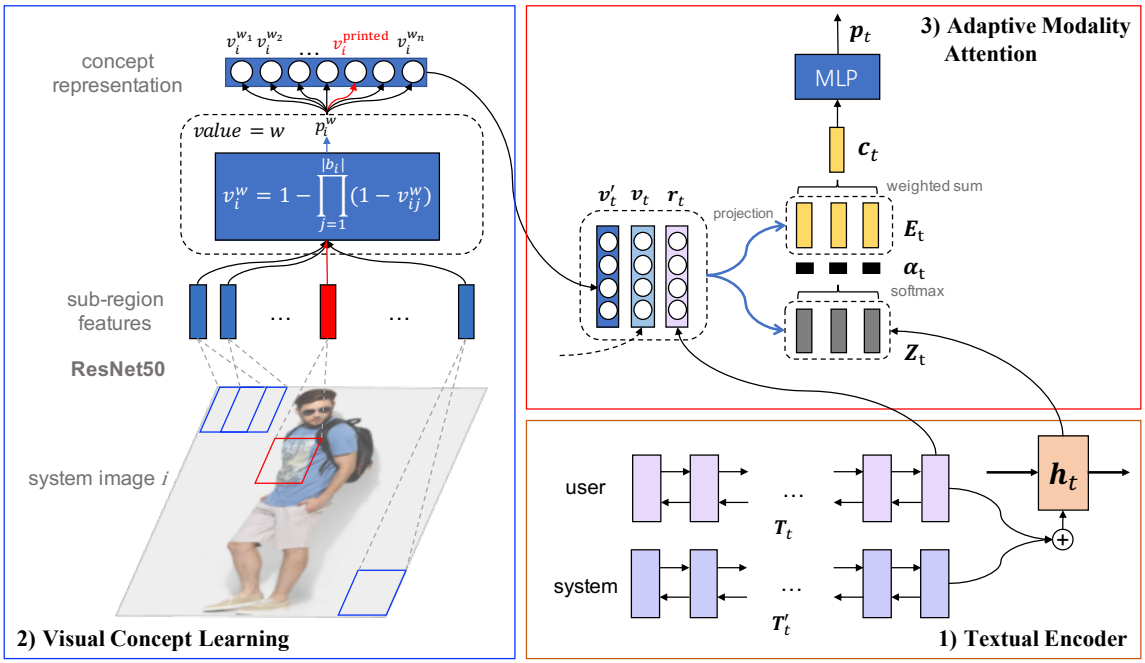


Figure 3: Overview of our proposed model. We first extract the features of user's and system's textual input evidences using two RNN networks and use them to update the textual base hidden state $\mathbf{h}_t$. In the visual concept learning stage, the model learns visual semantics by looking into image sub-regions and integrating probabilities under the multiple instance learning scenario. Finally, we apply the adaptive modality attention mechanism on different evidence sources to obtain the context vector $\mathbf{c}_t$, which is then fed into a MLP followed by a softmax activation to get the final belief state $\mathbf{P}_t$.

mentioned before, we have a totally of N_s slots. Thus, we maintain N_s such RNN pipelines for all the slots.

3.3 Sub-region based Visual Concept Learning

One of the major challenges in multimodal DST is to accurately extract visual concepts from images, which can be easily formulated as a multi-class or multi-label image classification problem with manual annotation. However, we notice that most visual concepts correspond to only a small sub-region of the image, such as the slots "Neck" and "Belt-loops". Studies like [6] have also shown that feeding the whole image into attribute classifiers leads to worse performance. Therefore, we delve into image sub-regions to harvest visual concepts.

Basically, for each individual image i, we define a bag b_i, which is a collection of the image's sub-regions (detailed in Section 4.3). An image is labeled positive for the concept w if there is at least one sub-region in the bag containing w, which is actually a multiple instance learning (MIL) problem. In this work, the sub-regions are squared areas which can overlap in case that some objects are cut up by the square boundary. Intuitively, suppose the probability of one sub-region j containing the concept w is v_{ij}^w, the probability of the whole image i containing w should be no less than the largest sub-region concept probability in the bag. At the same time, multiple sub-regions showing high probability of containing the concept w should result in increased probability but not over-exaggerated. A diminishing return characteristic is preferable for our case. Therefore, the probability of image i containing the concept w is defined as follows:

$$v_i^w = 1 - \prod_{j=1}^{|b_i|}(1 - v_{ij}^w), \tag{10}$$

where v_i^w is not larger than 1 and not less than the largest v_{ij}^w. When there are more v_{ij}^w being large, the incremental effect on v_i^w actually decreases.

We use the ResNet-50 without the last layer as the base network to learn the representation for image sub-region b_{ij} as $\mathbf{b}_{ij}$, and feed it into a fully-connected layer followed by a sigmoid activation to compute the sub-region level concept probability vector $\mathbf{v}_{ij}$:

$$\mathbf{v}_{ij} = \sigma(FC(\mathbf{b}_{ij}))), \tag{11}$$

where $\mathbf{v}_{ij}$ is a collection of Bernoulli distributions over each concept and v_{ij}^w is the probability of concept w.

At the whole image level, we obtain the image representation $\mathbf{v}_i$ for image i via

$$\mathbf{v}_i = 1 - \prod_{j=1}^{|b_i|}(1 - \mathbf{v}_{ij}). \tag{12}$$

3.4 Adaptive Modality Attention

When updating the dialogue state, each slot should emphasize on different evidence sources of the input. For example in Figure 2, the user said "... something like it but in type as in this image", which means that the value of "type" should be equal to that of the user given image, while the values of other primary slots should be the same as the chosen image ("the 1st image"). Another common

situation is that we can find an exact value in user text, such as "Show me more in purple colored type".

Generally speaking, there are three main sources of evidences for tracking dialogue states: the system provided images, user provided image [2] and user provided text. Note that the system provided text offers little useful clues about states, we thus only use it in context modeling as in Equation 8, and do not take it as an evidence source in a way similar to [30].

Essentially, we aim to get a context vector $\mathbf{c}_t$ as an attentive summarization of the three evidence sources: user provided image representation $\mathbf{v}_t$, system provided image representation $\mathbf{v}_t^{'}$, and user text representation $\mathbf{r}_t$. We then feed it into a fully-connected layer followed by a softmax activation to find the new distribution on values:

$$\mathbf{p}_t = softmax(FC(\mathbf{c}_t)). \tag{13}$$

The summarization vector $\mathbf{c}_t$ is obtained by combining the representations of the three evidence sources using a weight vector $\boldsymbol{\alpha}_t$, which is an probability distribution over the three input sources. How to decide the attention weights $\boldsymbol{\alpha}_t$ is essential to our model since emphasizing on the correct evidence source is critical for generating the right slot value. Therefore, we first use two projection matrices to map the three evidence vectors into a common space and obtain an concatenated evidence matrix $\mathbf{E}_t$:

$$\mathbf{c}_t = \sum_{m=1}^{3} \alpha_{t,m} \mathbf{E}_t(m), \tag{14}$$

$$\mathbf{E}_t = [\mathbf{W}_1 \mathbf{v}_t, \mathbf{W}_1 \mathbf{v}_t^{'}, \mathbf{W}_2 \mathbf{r}_t], \tag{15}$$

where $\mathbf{W}_1 \in \mathcal{R}^{h \times d}$ and $\mathbf{W}_2 \in \mathcal{R}^{h \times h}$ are projection parameters. m indicates the index of evidence source and $\mathbf{E}_t(m)$ is the m-th column vector of $\mathbf{E}_t$.

To get the attention distribution $\boldsymbol{\alpha}_t$, we first project the visual and textual evidences into a common space and then feed them along with the RNN output $\mathbf{h}_t$ into a neural network followed by a softmax activation:

$$\mathbf{Z}_t = [\mathbf{W}_3 \mathbf{v}_t, \mathbf{W}_3 \mathbf{v}_t^{'}, \mathbf{W}_4 \mathbf{r}_t], \tag{16}$$

$$k_{t,m} = score(\mathbf{Z}_t(m), \mathbf{h}_t), \tag{17}$$

$$\alpha_{t,m} = \frac{\exp(k_{t,m})}{\sum_{i=1}^{3} \exp(k_{t,m})}, \tag{18}$$

where $\mathbf{W}_3 \in \mathcal{R}^{h \times d}$, $\mathbf{W}_4 \in \mathcal{R}^{h \times b}$ are projection parameters mapping the textual and visual evidence $\mathbf{v}_t$, $\mathbf{v}_t^{'}$ and $\mathbf{r}_t$ into a common space. The *score* function yields a scalar measuring to what extent each evidence source is matched to the slot. It is based on the projected evidence source $\mathbf{Z}_t(m)$ and the textual encoder state $\mathbf{h}_t$. In our implementation, we parametrized it as a feed forward neural network which is jointly trained with all the other components.

$$score(\mathbf{Z}_t(m), \mathbf{h}_t) = \mathbf{w}_h^{\top} \tanh\left(\mathbf{Z}_t(m) + \mathbf{W}_g \mathbf{h}_t\right), \tag{19}$$

where $\mathbf{w}_h \in \mathcal{R}^h$, $\mathbf{W}_g \in \mathcal{R}^{h \times h}$ are the parameters to be learned, h, b are the model hyper-parameters in which b is the RNN hidden state size. Note that in our attention mechanism, we use different key and value matrices $\mathbf{Z}_t$ and $\mathbf{E}_t$ by projecting the evidence source into different spaces, since earlier works [28] have already suggested that the dual use of a single vector makes training the model difficult.

The intuition of this design is as follows: as defined in Section 3.2, the utterance-level hidden state $\mathbf{h}_t$ of the basic textual framework contains the long and short term information obtained from the textual side. More specifically, the long term information is an aggregation of previous dialogue evidences, such as whether certain slot has already been mentioned in history. While the short term information includes more recent evidences such as which slot is mentioned in the current user post. Both kinds of information can provide extensive clues on which modality should be attended to in the current turn. We thus feed the evidence source representations along with $\mathbf{h}_t$ through a single layer neural network (the *score* function) followed by a softmax activation to generate the attention distribution $\boldsymbol{\alpha}_t$ over different evidence sources.

We use cross entropy loss to measure the prediction results of belief tracker. Specifically, we have:

$$L = -\sum_{w} y_t^w \log p_t^w, \tag{20}$$

where w indicates a certain slot value, p_t^w is an element of $\mathbf{p}_t$ which is the probability that w is chosen as the new belief state, and y_t^w is the golden truth.

4 EXPERIMENT

In this section, we conducted extensive experiments to validate our NMBT's performance on the task of dialogue state tracking[3]. More specifically, we want to figure out:

- How much can the task of dialogue state tracking benefit by involving multimodal evidence ?
- Whether the image representations obtained by sub-region based visual concept learning perform better than the dense features extracted by the CNN models ?
- Has the adaptive attention mechanism really learned to pay attention to the correct modality ?

4.1 Data Preparation

The collection of training corpus is one of the bottlenecks for developing statistical dialogue system, due to the cost and concerns about privacy disclosure. Recently, [34] proposed a large-scale multimodal dialogue dataset which consists of about 150K conversation sessions between sale agents and shoppers. However, there is no dialogue state labels in the original dataset, we thus utilized the meta data provided by [34] to build the domain ontology[4] for dialogue act annotation. More specifically, we extracted slot-value pairs from the corpus using a two stage process. During the first stage, we defined a slot value dictionary and 81 natural language templates to extract slots using direct string matching techniques. During the second stage, we conducted manual correction. Finally, seven slots are considered and the number of their corresponding values are shown in Table 1. To train the sub-region based visual concept learning model, we also need to get a collection of images with labels in this domain. We extracted the labels of each image

[2] In the dataset, users only provide one image each time.

[3] The code is available at https://github.com/zhangzthu/NMBT

[4] The domain ontology means all the slots and their values to be involved in the study.

from the raw catalog which consists of text descriptions of each fashion item.

Table 1: The number of values of different slots.

Slot Name	# Value
material	114
style	17
color	47
type	56
fit	17
length	19
gender	3

In the dataset, the average turn number of dialogue sessions is 18.3, which is a lot larger than other text-based datasets [13, 14] and leads to greater difficulty.

4.2 Baselines

To evaluate the effectiveness of our proposed NMBT model, we compared it with the following baselines.

- **Seq2seq_DST** [17]: This model includes an encoder-decoder architecture with an attention mechanism to map an input utterance to a sequence of slot-value pairs. Note that Seq2seq_DST is different from RNN_DST in that the last hidden state of Seq2seq_DST's encoder is fed into an RNN decoder, rather than a MLP classifier.
- **CNN_DST**: A CNN based textual neural belief tracker[36], which utilizes a slot-specific filter to extract semantic features from raw inputs.
- **RNN_DST**: The textual only framework of our NMBT model. The hidden state of the utterance level encoder is fed into a MLP classifier to predict the belief state.
- **NBT**: A textual DST model which uses CNN to learn n-gram utterance feature from word embeddings. The utterance feature is then used for DST tracking [30].
- **M-RNN_DST**: We extended the RNN_DST model to multimodal scenario by feeding the image feature vectors of each turn as extra input of the untterance level encoder, which is similar to the Multimodal HRED model in [34]. The difference is that the last hidden state of the untterance level encoder is fed into a MLP with a softmax activation to predict the slot value rather than a natural language decoder.
- **M-NBT**: The multimodal version of NBT, in which we combined the ResNet-50 extracted image features with the text features obtained before the last layer of NBT to predict the dialogue state, similar as Multimodal HRED [34] does.

It is worth noting that Seq2seq_DST, CNN_DST, RNN_DST and NBT are representative textual-based models from the Dialogue State Tracking Challenge [13], while M-RNN_DST and M-NBT are two multimodal extensions based on strong textual baselines.

4.3 Experimental Setups

The training of our model is carried out in two stages. First, we pre-train the basic textual model and the visual concept detector respectively. Then we fuse them together to train the full NMBT model with adaptive modality attention. For the textual base component, both RNN encoders' hidden state sizes are set to 512, and the number of layers is 3. We use 300 as the dimension of word vectors, which are extracted by pre-trained GloVe model [31]. During training, we keep the extracted word embeddings fixed.

Table 2: The dialogue state tracking accuracy of different models.

Method	Overall	Slots			
		Style	Material	Color	Fit
Seq2seq_DST	51.1	80.0	84.7	47.7	51.2
CNN_DST	51.7	81.4	81.5	50.8	54.8
RNN_DST	51.2	80.4	85.1	49.5	51.5
NBT	52.0	84.9	66.1	49.3	47.9
M-RNN_DST	56.4	81.8	87.0	49.9	53.7
M-NBT	57.2	82.7	87.7	51.5	56.1
NMBT w/o SBVL	58.6	86.9	89.7	52.6	59.0
NMBT w/o Attn	57.5	81.3	88.0	54.3	56.8
NMBT	**59.8**	**87.9**	**92.3**	**55.6**	**60.6**

The sub-regions in visual concept learning are defined by sliding window. More specifically, we first resize the image to 468×468, and the size of image sub-region is 224×224 as in ResNet-50. With a stride size of 30, we finally get 81 (9×9) sub-regions for each image. The stride size is determined through mode validation, since a smaller size leads to computing complexity and a larger one may cut off a potential object. The size of image feature extracted by ResNet-50 is 2048, which is then fed into a MLP followed by a sigmoid activation to get $\mathbf{v}_{ij}$ (see Equation 10). The h in Section 3.4 is set to 100. Both MIL and adaptive modality attention parameters is trained using Adam optimizer with a learning rate of 0.001, and the momentum parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

4.4 Evaluation Protocols

We adopted the dialogue state tracking accuracy as the main metric to evaluate the performance of our model. In each turn, the model predicts a value for each slot k. For each slot, we can get a slot-wise accuracy. The overall accuracy is averaged over all slots. Each user utterance from the original corpus is annotated with a specific *state type*, which indicates its function, such as "*show orientations*" and "*show similar to*". To give a more elaborate comparison, we also provide the accuracy scores of each baseline model on different state types.

We also analysis the performance of visual concept detection. For each slot k, we picked out the value w with the highest probability v_{ij}^w as the predicted value for that slot. In this metric, we show the overall and slot-wise accuracy scores. To give a more intuitional understanding of how the visual concept extractor works, we visualize its v_{ij}^w results for some concept w in Figure 4. As described in Section 4.3, we get a 9×9 v_{ij}^w map for each image. In order to visualize these probabilities on the image, we simply resize the heat map to 464×464 by upsampling and apply a Gaussian filter. Note that the heat map here is not the "attention" heat map which is widely used in many attention-based visual models [42]. In our

model, v_{ij}^{w} is a binary classification probability, which indicates the probability that sub-region j contains concept w.

To validate the effectiveness of our adaptive modality attention mechanism, we conduct a case study with a visualization of the adaptive attention weights. Due to space limitation, we only visualize the attention weights of five representative slots.

4.5 Performance of Dialogue State Tracking

We first report the DST prediction performance on the overall accuracy score and accuracy scores of several representative slots. Besides the baseline methods described in Section 4.2, we also conduct two ablation studies on the sub-region based visual concept learning (**SBVL**) component and modality attention mechanism (**Attn**). The results are shown in Table 2. Note that we only report the result of some primary slots.

Some key observations are summarized as follows:

- First of all, as compared to textual-based methods, there is a significant improvement on both the overall and the slot-specific accuracy scores for those methods considering multimodal information. For example, the overall accuracy score of M-RNN_DST increases 5.2% as compared to that of its pure textual-based version RNN_DST. The overall accuracy score of M-NBT also shows similar improvement pattern comparing to that of NBT. It indicates that by involving images into dialogue state tracking, the model is able to extract more useful information for better tracking the user's intention. In fact, it is natural to use images in the fashion products shopping conversation scenario. As the example shown in Figure 1, the images carry detailed information about the user's requirements which can not be easily expressed using only textual utterances. Therefore, it is crucial to capture the visual evidence in dialogue state tracking under these multimodal scenarios.
- Secondly, by learning sub-region based concept level visual representations, our proposed model achieves better performance as compared to the multimodal models that leverage pre-trained model extracted visual features. We compared with several multimodal baselines and a variation of our model named NMBT w/o SBVL, in which we remove the sub-region based visual concept learning component while replace it with the pre-trained ResNet-50 to extract visual features. In Table 2, we observe that utilizing visual information by simply using representations learned by pre-trained ResNet-50 is not an efficient way. There exists a large performance gap between our method NMBT and M-RNN_DST, M-NBT. For instance, the overall accuracy score of NMBT is improved by 3.4% and 2.6% respectively as compare to that of M-RNN_DST and M-NBT. For our proposed model NMBT, when the sub-region based visual concept learning component is removed, the overall performance of resulting method NMBT w/o SBVL drops about 1.2%. These results validate the effectiveness of our proposed visual concept learning model. By using sub-regions of image rather than the whole image, the model manages to learn more accurate concepts with less background noise.
- Thirdly, by combining textual and visual evidences through adaptive modality attention mechanism, NMBT manages to learn a better integrated representation of multimodal evidences. In M-RNN_DST and M-NBT, the textual and visual representations are integrated by vector concatenation. In the ablation model NMBT w/o Attn, we removed the attention mechanism by redefining the context vector in Equation 13 as a concatenation of the column vectors in $\mathbf{E}_t$. Compared to these multimodal models, NMBT achieves better performance on both slot-specific and overall accuracy. Intuitively, it is a non-trivial task to fuse the visual and textual evidences together. The image provided by either system or user can be regarded as an attribute list, which acts as a candidate value set for belief state update. The textual utterances contain two important clues: the values of some slots which are explicitly expressed in the utterance, such as "looking for some **formal** shoes", and the control information which decides the source of each slot's update, such as "in the **style** as in the **1st** image". This mechanism can not be model by either simple feature concatenation as in M-RNN_DST, M-NBT, or the NMBT w/o Attn ablation which simply averages the three evidences. In contrast, our model learns to automatically emphasize on the different evidence sources for updating belief state based on the conversational context.

4.6 Fine-grained DST Performance Analysis

To give a more detailed analysis, we report the state type level accuracy on three representative types, as shown in Table 3. First we analyze the function of "*show similar to*" which means that the user requires the agent to show similar items to the currently selected product. The selected product can come from one of the three evidence sources, which favors our adaptive attention mechanism. Therefore, we observe that the performance of our method NMBT in this type is better than the averaged performance shown in Table 2, which means in this scenario where the modality choice appears frequently, the attention mechanism manages to emphasize the correct modality in most cases. Also, our NMBT method obtains higher accuracy compared to the baseline methods. Next we analyzed "*like show result*" which is similar to "*show similar to*" and thus we can get similar conclusion based on its results, which further verifies NMBT's performance on emphasizing different modalities. When coming to the "*like earlier show result*" type which means that the user requires items related to results shown several turns before, we observed that although NMBT still outperforms baseline methods, the margin is dwindled from 2.2% to 1.1%. The possible reason might be that in "*like earlier show result*", the agent needs to recall back to previous turns other than the current turn, which brings extra difficulty.

4.7 Evaluation of Visual Concept Learning Component

Learning good visual representations is crucial to the model's performance since it serves as the input to downstream modules of dialogue state tracking. We thus analyze the effectiveness of visual concept learning in terms of concept classification accuracy and report results in Table 4.

We found that the sub-region based visual concept learning (**SRVL**) can capture more accurate classification features by localizing the visual concept and reducing background noise. This statement can be concluded from Table 4, where the overall and several representative slot-wise concept prediction accuracy scores

Table 3: The comparison of DST accuracy of different methods over three state type, where "*show similar to*" and "*like show result*" indicate the user asks to recommend products similar to the current one, and "*like earlier show result*" means the user requires products similar to someone in previous turns. Several representative slot-specific accuracy scores and the average accuracy score are reported. Note that the accuracy scores here are calculated on state type level, which can not be compared with the results in Table 2.

Method	*show similar to*					*like show result*					*like earlier show result*				
	Style	Material	Color	Fit	Overall	Style	Material	Color	Fit	Overall	Style	Material	Color	Fit	Overall
Seq2seq_DST	81.5	88.6	49.4	52.1	52.6	86.2	88.0	47.9	55.7	53.6	81.7	83.6	48.6	46.2	51.6
CNN_DST	84.3	85.7	51.7	56.4	53.7	87.6	85.8	51.1	60.3	54.7	83.8	80.7	51.5	51.0	52.5
RNN_DST	83.3	87.4	49.9	55.3	52.6	88.0	87.3	48.0	55.1	54.0	81.9	86.0	48.7	46.6	52.4
NBT	84.7	72.5	45.3	50.7	52.0	85.5	71.9	54.6	47.3	53.1	84.0	67.6	49.0	45.2	52.1
M-RNN_DST	81.2	82.3	50.0	67.1	64.9	78.2	88.2	47.3	71.8	66.9	**87.5**	82.9	51.0	49.0	60.7
M-NBT	84.3	84.6	50.5	67.4	65.7	85.0	91.2	52.1	71.6	69.0	83.0	84.9	51.4	46.8	60.1
NMBT	**87.6**	**91.1**	**55.4**	**72.1**	**68.5**	**88.6**	**95.9**	**55.7**	**75.5**	**71.2**	85.6	**88.9**	**55.0**	**51.6**	**61.8**

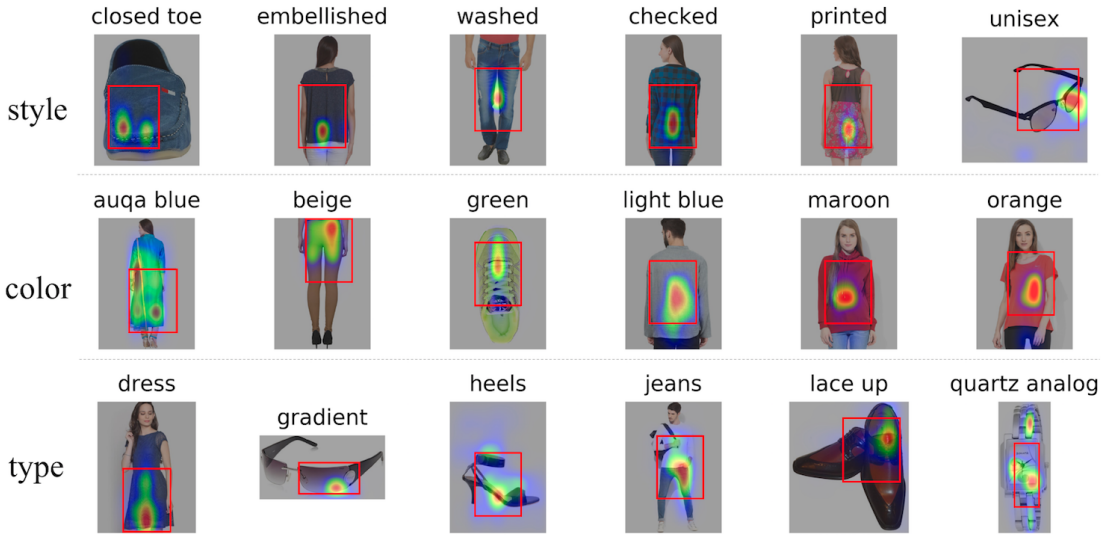


Figure 4: Sub-region based multiple instance learning for visual concept learning. We visualized the concept region probability of values such as *style*, *color* and *type*, where the three slots correspond to the three rows. The sub-region with the highest probability containing the concept is marked by a red rectangle.

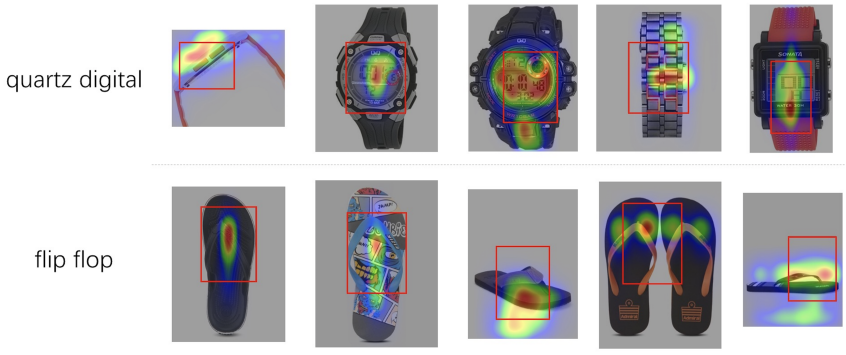


Figure 5: Visualization of the heat map of v_{ij}^{w} and the bounding box of *quartz digital* and *flip flop* concepts. We present multiple examples for the two concepts where the product images are taken in different orientations.

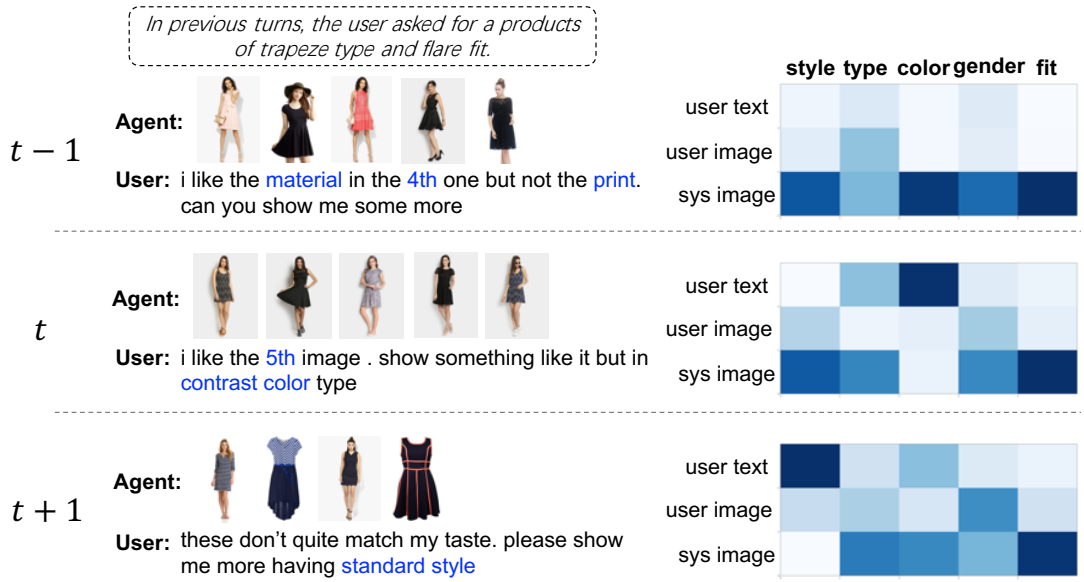


Figure 6: Visualization of modality attention weights. The attention weights of five major slots over the three evidence sources are plotted, where darker color indicates larger attention weight. Note that each column represents an attention distribution.

Table 4: The visual concept detection accuracy on different slots.

Method	Overall	Slots			
		Style	Material	Color	Fit
ResNet50	42.8	58.7	21.4	**42.8**	48.1
SRVL	**44.8**	**61.2**	**24.2**	42.0	**52.3**

are reported. We compare with the ResNet-50 model which directly uses the feature vector of the entire image to predict a concept. As shown in Table 4, by incorporating sub-region based visual concept learning, the accuracy score is improved by about 2.0%. Nevertheless, as it can be seen in the slot-specific result, the accuracy on "*color*" is dwindled by about 0.8%. This result is generally in line with our intuition, since the slot "*color*" is a more holistic concept which makes it more easily to be predicted with the overall image feature.

In order to validate whether our design of using image sub-regions is reasonable and whether our model can find those correct sub-regions for concepts, we visualize the spatial response map v_{ij}^{w} of some concepts as shown in Figure 4. The bounding box of the sub-region with the highest probability containing the concept is marked by a red rectangle. We can see from the visualization results that without bounding box annotations for training, our model is still able to locate and associate visual concepts with correct sub-regions. For example, in the "*closed toe*" picture, our model locates the sub-region on the toe part of a shoe, and in the "*washed*" example, our model focuses on the washed-styled pants rather than the T-shirts. For "*color*" prediction, the model focuses on the main body of each product which takes the most information about the product's color. These results indicate that the image representations given by sub-region based multiple instance learning indeed capture important visual classification concepts and can thus extract informative features for the task.

We further validate whether our proposed model can capture a concept presented in different orientations or poses. In fashion domain, the product images are often taken in different orientations, which requires the visual concept detector to identify a single concept in different modes. For example, as for the concept "*watch*", the model needs to identify it no matter its image is taken from the side or front, while the image features can be very different in this two cases. To validate whether our model fits this requirement, we conducted a more specific analysis by showing the heat map and bounding box of a single concept in images taken from different orientations. As demonstrated in Figure 5, we gave five images for two concepts, taken from at least three different orientations. For the concept of "*quartz digital*", our model can identify it by locating the feature of *knob*, *dial* and *band*, which are very different features. As for "*flip flop*", our model identifies its key characteristic, the *toe thong*, from different orientations. In the 4th image where there are two of it, the model is even able to find the *toe thong* feature for both identical objectives.

4.8 Case Study of Adaptive Modality Attention

As the adaptive modality attention is essential in our model, we conducted case studies by visualizing the modality attention weights to verify how it works in modeling multimodal conversations. As shown in Figure 6, we visualized the attention weights for our proposed model. Note that each column represents an attention distribution over the three sources where darker color indicates larger attention weight.

The result demonstrates that the adaptive modality attention mechanism is able to automatically emphasize the three sources based on both textual and visual contexts in multimodal dialogue,

even without specific supervision. For example in Figure 6, during the t-th turn, the user asked for products similar to the 5th one but with a different color, which indicates that the update of all slots except "*color*" should attend to system image. The corresponding attention weights fit this expectation, in which the colors of other slots is more shaded on "*sys image*", while that of "*color*" is more shaded on user text. In the $(t+1)$-th turn, the user asked for "*standard*" style products which is explicitly expressed in user text. Hence, the tracker of "*style*" pays more attention to user text. This case shows that the adaptive attention mechanism can capture the clues about which part is more important for each slot's update and thus boosts the overall performance.

5 CONCLUSION

In this work, we studied how visual evidence can be incorporated in the task of dialogue state tracking, and proposed a neural multimodal belief tracking model named NMBT, which seamlessly integrates and adaptively selects textual and visual information in multi-model dialogues. This model consists of a textual encoder which encodes textual utterances, a sub-region based visual concept detector which extracts concepts from image, and a multi-modality attention mechanism which adaptively attends to textual or visual evidence during conversations. Extensive experiments demonstrated that our model outperforms the state-of-the-art baselines. Results showed that dialogue state tracking in multimodal dialogues can significantly benefit from jointly considering multimodal evidences.

Multi-modal dialogue systems have many applications in real-world dialogue systems such as online shopping or virtual dialogue agent. We believe that this research direction is still in its infancy and our work may inspire many future studies.

6 ACKNOWLEDGMENTS

This work was supported by the National Key R&D Program of China (Grant No. 2018YFC0830200), the National Science Foundation of China (Grant No.61876096/61332007) and the National Research Foundation, Prime Minister's Office, Singapore under its IRC@SG Funding Initiative.

REFERENCES

[1] Xavier Alameda-Pineda, Miriam Redi, Mohammad Soleymani, Nicu Sebe, Shih-Fu Chang, and Samuel Gosling. 2017. MUSA2: First ACM Workshop on Multimodal Understanding of Social, Affective and Subjective Attributes. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 1974–1975.

[2] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*. 2425–2433.

[3] Dan Bohus and Alex Rudnicky. 2006. A k-hypotheses+ other belief updating model. In *Proc. of the AAAI Workshop on Statistical and Empirical Methods in Spoken Dialogue Systems*, Vol. 62.

[4] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. 2017. Visual dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2.

[5] Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence* 89, 1-2 (1997), 31–71.

[6] Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. 2015. From captions to visual concepts and back. In *CVPR*. 1473–1482.

[7] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. 2013. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*. 2121–2129.

[8] Yunchao Gong, Qifa Ke, Michael Isard, and Svetlana Lazebnik. 2014. A multi-view embedding space for modeling internet images, tags, and their semantics. *International journal of computer vision* 106, 2 (2014), 210–233.

[9] Yunchao Gong, Liwei Wang, Micah Hodosh, Julia Hockenmaier, and Svetlana Lazebnik. 2014. Improving image-sentence embeddings using large weakly annotated photo collections. In *European Conference on Computer Vision*. Springer, 529–545.

[10] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *CVPR*. 3.

[11] David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. 2004. Canonical correlation analysis: An overview with application to learning methods. *Neural computation* 16, 12 (2004), 2639–2664.

[12] Matthew Henderson. 2015. Machine learning for dialog state tracking: A review. In *Proc. of The First International Workshop on Machine Learning in Spoken Language Processing*.

[13] Matthew Henderson, Blaise Thomson, and Jason D Williams. 2014. The second dialog state tracking challenge. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*. 263–272.

[14] Matthew Henderson, Blaise Thomson, and Jason D Williams. 2014. The third dialog state tracking challenge. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, 324–329.

[15] Matthew Henderson, Blaise Thomson, and Steve Young. 2013. Deep neural network approach for the dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*. 467–471.

[16] Matthew Henderson, Blaise Thomson, and Steve Young. 2014. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*. 292–299.

[17] Takaaki Hori, Hai Wang, Chiori Hori, Shinji Watanabe, Bret Harsham, Jonathan Le Roux, John R Hershey, Yusuke Koji, Yi Jing, Zhaocheng Zhu, et al. 2016. Dialog state tracking with attention-based sequence-to-sequence learning. In *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 552–558.

[18] Ting-Hao Kenneth Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, et al. 2016. Visual storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1233–1239.

[19] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3128–3137.

[20] Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. 2014. Deep fragment embeddings for bidirectional image sentence mapping. In *Advances in neural information processing systems*. 1889–1897.

[21] Ryan Kiros, Ruslan Salakhutdinov, and Richard S. Zemel. 2014. Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models. *arXiv preprint arXiv:1411.2539* (2014).

[22] Pei Ling Lai and Colin Fyfe. 2000. Kernel and nonlinear canonical correlation analysis. *International Journal of Neural Systems* 10, 05 (2000), 365–377.

[23] Staffan Larsson and David R Traum. 2000. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural language engineering* 6, 3-4 (2000), 323–340.

[24] Esther Levin, Roberto Pieraccini, and Wieland Eckert. 1998. Using Markov decision process for learning dialogue strategies. In *ICAssP*, Vol. 98. 201–204.

[25] Esther Levin, Roberto Pieraccini, and Wieland Eckert. 2000. A stochastic model of human-machine interaction for learning dialog strategies. *IEEE Transactions on speech and audio processing* 8, 1 (2000), 11–23.

[26] Lizi Liao, Yunshan Ma, Xiangnan He, Richang Hong, and Tat-seng Chua. 2018. Knowledge-aware Multimodal Dialogue Systems. In *2018 ACM Multimedia Conference on Multimedia Conference*. ACM, 801–809.

[27] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan L Yuille. 2014. Explain images with multimodal recurrent neural networks. *arXiv preprint arXiv:1410.1090* (2014).

[28] Hideya Mino, Masao Utiyama, Eiichiro Sumita, and Takenobu Tokunaga. 2017. Key-value Attention Mechanism for Neural Machine Translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 290–295.

[29] Nasrin Mostafazadeh, Chris Brockett, Bill Dolan, Michel Galley, Jianfeng Gao, Georgios P Spithourakis, and Lucy Vanderwende. 2017. Image-grounded conversations: Multimodal context for natural question and response generation. *arXiv preprint arXiv:1701.08251* (2017).

[30] Nikola Mrkšić, Diarmuid O Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2016. Neural belief tracker: Data-driven dialogue state tracking. *arXiv preprint arXiv:1606.03777* (2016).

[31] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[32] Mengye Ren, Ryan Kiros, and Richard Zemel. 2015. Exploring models and data for image question answering. In *Advances in neural information processing systems*. 2953–2961.

[33] Amrita Saha, Mitesh Khapra, and Karthik Sankaranarayanan. 2017. Multimodal Dialogs (MMD): A large-scale dataset for studying multimodal domain-aware conversations. *arXiv preprint arXiv:1704.00200* (2017).

[34] Amrita Saha, Mitesh Khapra, and Karthik Sankaranarayanan. 2017. Towards Building Large Scale Multimodal Domain-Aware Conversation Systems. *arXiv preprint arXiv:1704.00200* (2017).

[35] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models.. In *AAAI*, Vol. 16. 3776–3784.

[36] Hongjie Shi, Takashi Ushio, Mitsuru Endo, Katsuyoshi Yamagami, and Noriaki Horii. 2017. Convolutional neural networks for multi-topic dialog state tracking. In *Dialogues with Social Robots*. Springer, 451–463.

[37] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3156–3164.

[38] Liwei Wang, Yin Li, and Svetlana Lazebnik. 2016. Learning deep structure-preserving image-text embeddings. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5005–5013.

[39] Jason Williams, Antoine Raux, and Matthew Henderson. 2016. The dialog state tracking challenge series: A review. *Dialogue & Discourse* 7, 3 (2016), 4–33.

[40] Jason D Williams. 2012. A critical analysis of two statistical spoken dialog systems in public use. In *Spoken Language Technology Workshop (SLT), 2012 IEEE*. 55–60.

[41] Jason D Williams. 2014. Web-style ranking and SLU combination for dialog state tracking. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*. 282–291.

[42] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. 2048–2057.

[43] Steve Young. 2006. Using POMDPs for dialog management. In *Spoken Language Technology Workshop, 2006. IEEE*. IEEE, 8–13.

[44] Steve J Young. 2000. Probabilistic methods in spoken–dialogue systems. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 358, 1769 (2000), 1389–1402.

[45] Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2016. Yin and yang: Balancing and answering binary visual questions. In *CVPR*. 5014–5022.

[46] Victor Zue, Stephanie Seneff, James R Glass, Joseph Polifroni, Christine Pao, Timothy J Hazen, and Lee Hetherington. 2000. JUPlTER: a telephone-based conversational interface for weather information. *IEEE Transactions on speech and audio processing* 8, 1 (2000), 85–96.

Auto-EM: End-to-end Fuzzy Entity-Matching using Pre-trained Deep Models and Transfer Learning

Chen Zhao*
University of Maryland, College Park
chenz@cs.umd.edu

Yeye He
Microsoft Research, Redmond
yeyehe@microsoft.com

ABSTRACT

Entity matching (EM), also known as entity resolution, fuzzy join, and record linkage, refers to the process of identifying records corresponding to the same real-world entities from different data sources. It is an important and long-standing problem in data integration and data mining. So far progresses have been made mainly in the form of model improvements, where models with better accuracy are developed when large amounts of training data is available. In real-world applications we find that advanced approaches can often require too many labeled examples that is expensive to obtain, which has become a key obstacle to wider adoption.

We in this work take a different tack, proposing a transfer-learning approach to EM, leveraging pre-trained EM models from large-scale, production knowledge bases (KB). Specifically, for each entity-type in KB, (e.g., location, organization, people, etc.), we use rich synonymous names of known entities in the KB as training data, to pre-train type-detection and EM models for each type, using a novel hierarchical neural network architecture we develop. Given a new EM task, with little or no training data, we can either fine-tune or directly leverage pre-trained EM models, to build end-to-end, high-quality EM systems. Experiments on a variety of real EM tasks suggest that the pre-trained approach is effective and outperforms existing EM methods.[1].

ACM Reference Format:
Chen Zhao and Yeye He. 2019. Auto-EM: End-to-end Fuzzy Entity-Matching using Pre-trained Deep Models and Transfer Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313578

1 INTRODUCTION

Entity matching (EM), also known as entity resolution, fuzzy join, and record linkage, has numerous important applications such as knowledge fusion [29, 51], customer record deduplication [22, 61], etc. EM has been a long-standing problem in the data mining and data integration community, where extensive work has resulted in a long and fruitful line of research (e.g., see surveys in [28, 30, 45]).

A typical EM task in a relational setting, is to predict which records from two tables correspond to the same real-world entities. (EM in a graph setting such as knowledge graphs can be cast in a similar manner using graph connections [51]). Figure 1 shows an example of EM task. Given two tables of customer records with information such as customer names and addresses, we need to match records likely corresponding to the same person across the two data sources.

In this example, we can intuitively tell that the first two pairs of records are likely matches despite their differences in string representations – "Joe White" and "Joseph White" likely refer to the same entity, so do "CA" and "California". However minor string differences are not sufficient to ensure matches. To the contrary, there are many record pairs that have minor differences but are clearly non-matches. For example, in the last two pairs of records, "Sam A. Miller" and "Sam B. Miller" are likely not the same person, so are "Mark Johnson" and "Mary Johnson".

As we can see, these match/non-match decisions can be subtle, and are non-trivial to predict accurately. Existing EM approaches such as ML-based ones [10, 15, 61, 65], often require a large amount of training data (labeled match/non-match pairs) *each new EM task*, before accurate EM predictions can be made. It is clearly expensive, and sometimes impossible, to produce a large number of labeled data for *each EM task*, and this has become major obstacles to wider adoption of advanced EM techniques.

EM in real-world business applications. This study is conducted in the context of a commercial CRM (customer relationship management) system, where a pressing need is for enterprises using the CRM system, to match their customer records across data silos in different business applications (CRM, ERP, billing, service, etc.), so that these enterprises can have a unified view of their customers. Customer records are often in relational formats as shown in Figure 1, although the schema of the tables are not always known beforehand (e.g., data can be in non-standard databases or CSV files). A key requirement of EM in such applications, is that matching of customers should ideally work accurately out-of-box with little or no training data from *each enterprise*. EM applications like this make it difficult to apply advanced EM methods from the literature, which typically requires a large amount of labeled data [66].

Transfer learning for EM using pre-trained models. Partly motivated by the real-world EM application, we in this work take a very different tack. We argue that one does not need to re-train EM models from scratch for each new EM task, and we propose a novel transfer-learning approach using pre-trained EM models.

Specifically, our insight is that while each EM task may be different in its own ways (e.g., tables may have different attributes, and attributes have different importance, etc.), the types of attributes involved are often drawn from a set of common attributes (e.g.,

*Work done at Microsoft Research.
[1]We plan to release the pre-trained EM models, and are working through required processes to make this happen. Once approved these models will be released on GitHub at https://github.com/henryzhao5852/AutoEM.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313578

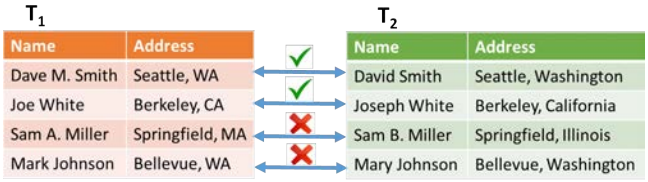


Figure 1: Example entity-matching between two tables.

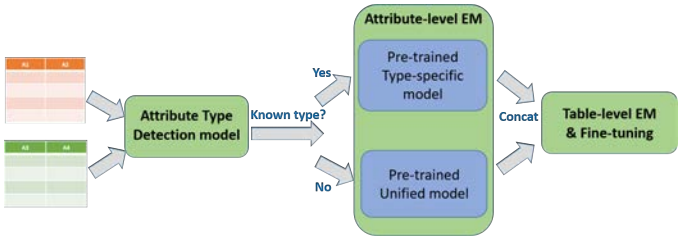


Figure 2: System architecture of end-to-end EM.

people-names, addresses, organizations, product-names, etc.). We observe that for each such attribute, the decision of match/non-match *at the attribute-level* can often be pre-trained and determined independent of the overall table-level EM task. For instance, in Figure 1, for the people-names domain, it is rather unambiguous that ("Joe White", "Joseph White") should match, while ("Sam A. Miller", "Sam B. Miller"), or ("Mark Johnson", "Mary Johnson") should not, irrespective of the EM tasks in question.

In addition, we observe that training data for these attribute-level match/non-match decisions are readily available in today's KBs, in the form of "synonymous names" for a large variety of entities (e.g., "Bill Gates" is also known as "William Gates" and "William H. Gates" in KBs). We leverage data harvested from KBs to pre-train accurate *attribute-level EM models* for a variety of domains, using a novel hierarchical deep model architecture we develop, which is shown to address shortcomings of existing methods.

Although the *attribute-level* EM decisions can be pre-trained and made separately, their contribution/importance to the overall *table-level* EM task can vary significantly. For instance, when matching two set of customer-records like in Figure 1, if the address-field of one table is "billing address" and the other is "mailing address", then they are not as important at the table-level (similarly, imagine if one phone-number field is "office-number" while the other is "cell-number"). In this work, we show that using pre-trained attribute-level EM models, and limited training data for each specific table-level EM task, we can quickly converge to accurate table-level EM predictions, through a fine-tuning step leveraging internal representations of these attribute-level EM models.

We would like to note that our pre-training approach to EM coincides with recent trends of employing pre-training models to NLP, including notable recent work such as BERT [26] that are shown to have great success.

Contributions. We make the following contributions.

- We propose a novel, end-to-end architecture for EM, that leverages large-scale and curated KB data, to pre-train models for both attribute type-detection, and attribute-level matches. For each new EM task, these attribute-level signals can be easily combined to produce an aggregate EM decision, using a small amount of training data.
- We develop a new hierarchical deep model to pre-train match/non-match for common types of attributes. This model leverages both character-level and word-level information and is designed to better handle complex attributes.
- We perform extensive experiments using KB data and real table data. Our results show that the proposed approach produces comparable or better results compared to state-of-the-art methods.

2 SYSTEM ARCHITECTURE

The EM problem we consider is simple to state: given two tables T_1 and T_2, with n and m records, respectively, determine for each record in T_1, if it matches with any record in T_2.

Figure 2 shows the end-to-end architecture of the proposed system. At a high level, the online EM prediction system has three main components: (1) Attribute-type detection; (2) Attribute-level EM; and (3) Table-level EM. We discuss each component in turn.

The first component is attribute-type detection, which takes a table as input, and predicts if each attribute/column in the table corresponds to a known KB type T. In the table of Figure 1, for instance, the first column is predicted as the KB type person, the second column as city, etc. These type-detection models are pre-trained offline using rich KB data from a commercial search engine. Specifically, KBs used by Google [5], Microsoft [32] and others have millions of entities for hundreds of common type such as person, city, organization, book, movie, etc. We leverage these (entity → type) data to train deep models to detect table column types. This component will be described in Section 4.

The second component is attribute-level EM models and is the central part of our system. It takes as input two entity values (e.g., "Dave M. Smith" and "David Smith" in Figure 1), and produces a score indicating the likelihood of match for the two input. We use two types of attribute-level EM models that are pre-trained offline:
(1) *Type-specific models*: For each known KB type T (e.g. person), we pre-train a separate model to predict match/non-match for values in T. We use synonymous entity names of type T in KB (e.g., "Bill Gates" is also known as "William Gates", "William Henry Gates" and "William H. Gates", etc.) as training data, and develop hierarchical deep models to learn name variations specific to each type T for accurate match/non-match decisions.
(2) *Unified model*: This is a single model that predicts match/non-match for values not in known KB types. While the model architecture is the same as the *type-specific models*, we use synonymous entity names taken from the *union* of many KB types, to pre-train a unified attribute-level EM that captures common name variations across different types (e.g., spelling variations). Such a model is reasonably accurate, and can be fine-tuned using limited training data to quickly fit a new type not known a priori.

As illustrated in Figure 2, at online prediction time the attribute-level EM can take two possible paths using the two types of models above, based on type-detection results. Specifically, if the type of a pair of attributes are detected to be a known KB type T, we apply the type-specific models for T (the upper path), otherwise we apply the general-purpose unified model (the lower path). We will describe this component in Section 3.

The final part of our system is the table-level EM. As discussed earlier, each table-level EM task can be different (e.g., different attributes, and different levels of importance for the same attributes). The table-level EM model starts from pre-trained attribute-level EM, and uses limited training data to quickly converge to aggregate EM decisions. This approach can also leverage pre-trained representations to fine-tune attribute-level EM for types that are not pre-trained, using limited (e.g., a few dozens) training data.

Terminology. Since in this work we will describe data coming from the contexts of relational tables and KBs, sometimes the same concept may be referred to using different names that are more natural in their respective contexts. For instance, "columns" or "attributes" that are more natural in tables, are better described as "entity-types" in KBs; similarly "attribute values" in tables are commonly described as "entity names" in KBs. While we try to keep the names consistent, we will use these names interchangeably in their corresponding contexts when appropriate.

3 ATTRIBUTE-LEVEL ENTITY MATCHING

We start by introducing our attribute-level EM models (in the middle of Figure 2), since they are the central part of the EM system, for which we develop novel hierarchical deep models. We defer the first component on type detection to Section 4, since we use simplified versions of the hierarchical models for type-detection.

Recall that attribute-level EM needs to take two attribute-values as input, and produce a score indicating their likelihood of match, which can be intuitively interpreted as "similarity".

3.1 Training data preparation

From Bing's knowledge graph [3, 32] (which is known as Satori and is similar to Google Knowledge Graph [5]), we select 40 head entity types that are deemed as common and useful for EM tasks (e.g., person, organization, city, book, etc.). In this KB (as well as other popular KBs [51, 56]), each entity e has an attribute called "alias", that lists all alternative/synonymous names of e. For example, the entity "Bill Gates" has alias "William Henry Gates", "William H. Gates", etc. These alternative names are clearly useful to train type-specific attribute-level EM models.

For positive examples, we take pairs of such alternative names, while filtering out pairs with no token overlap. The pairs that are listed as alternative names in KB but with no token overlap are likely semantic synonyms: e.g., "Lady Gaga" is also known as "Stefani Joanne Angelina Germanotta". Such semantic synonyms are too specific that are fine to memorize but difficult to generalize.

We would like to note that similar synonym data are also widely available in a similar manner from other KBs, such as the "also-known-as" relation in Wikidata [69], "alias" relation in Freebase [16], "foaf:nick" relation in DBpedia [12], "means" relation in YAGO [67], "alternateName" relation in Google Knowledge Graph [5]; as well as from standalone entity synonym data feeds [18, 21].

For negative examples, we use pairs of entities (e, e') in KB, whose names have some syntactic similarity. For example, we use "Bill Gates" and "Bill Clinton" as a pair of negative examples, as they resolve to different KB entities, but also share a common token in their names. The reason we require negative pairs to have syntactic similarity is that if the pair are completely different, it is trivial to determine that they should not match (e.g., "Bill Gates" and "Larry Page"). Such pairs would not be as helpful for models to learn. We generate "highly similar" pairs of names that are informative as negative examples as follows: for each entity e, we find top-100 entities in the same type, whose names are most similar to e (similarity to e is first decided based on the number of overlap tokens with e, and then based on Edit distance when there is a tie).

We note that for each canonical entity name from different types, on average we produce 2 to 5 positive examples (synonym names), and exactly 100 negative examples.

Our KB does not currently curates long-form physical mailing addresses (e.g., "206 South Cross Street, Little Rock, AR"), which however are common in EM tasks. In order to complement the KB for address data, we use query logs collected from the "Maps" vertical of the search engine, to obtain variations of addresses (in ways that users would type them), as well as their canonical addresses generated by the search engine. For example, a user query may be "206 South Cross Street, Little Rock, AR", and it is mapped to the canonical address "206 S Cross St, Little Rock, AR 72201" by the search engine. We collect such pairs of addresses as positive examples. And similar to KB types, negative examples are selected from high-similar address pairs that resolve to different canonical addresses. In total, we generate training data for addresses in 9 English-speaking locales (e.g. "en-us", "en-ca", "en-gb", "en-in", etc.), and use these as 9 additional types.

In total we pre-train EM models for these 49 attribute-types. We note that our approach of obtaining training data is general, and can be easily extended. For example, we could add types from KB, or use entity names in other languages from KB as additional types (most entities are curated to have names in many different languages). The same is true for addresses in other languages/locales.

3.2 Hierarchical Model for Attribute-level EM

We observe that positive examples of matching entity names exhibit complex structures and variations in different attribute-types. We make the following observations that motivate us to design a specific model architecture for attribute-level EM.

(1) First, we observe that sub-word/character-level matches are often important: for example, we have ("Dave Smith" = "David Smith"), and ("International Business Machine Corp" = "IBM Corp"), which requires character-level information to be modeled.

(2) In addition, word-level pairs are also an important source of information: for instance, we have ("Bill Gates" = "William Gates"), as well as ("William H. Gates" ≠ "William A. Gates") and ("Mary Miller" ≠ "Mark Miller"), etc. While character-level models are able to capture some of these, for long names with many tokens it can be difficult, such that explicit word-level models would be useful.

(3) Within one input, different words/characters may have different importance. For instance ("IBM Inc." = "IBM Corp."), since in the organization type words like "Inc." and "Corp." are not important; but ("IBM Corp." ≠ "IBS Corp."). The same is true for other types like person. This motivates us to introduce an intra-input, self-attention-like mechanism to learn character/word importance.

(4) Between two input, sometimes the word order may be different, e.g., ("Dave Smith" = "Smith, David"), which calls for an alignment-like, inter-input attention mechanism between the two input strings (reminiscent to attention used in machine-translation [13]).

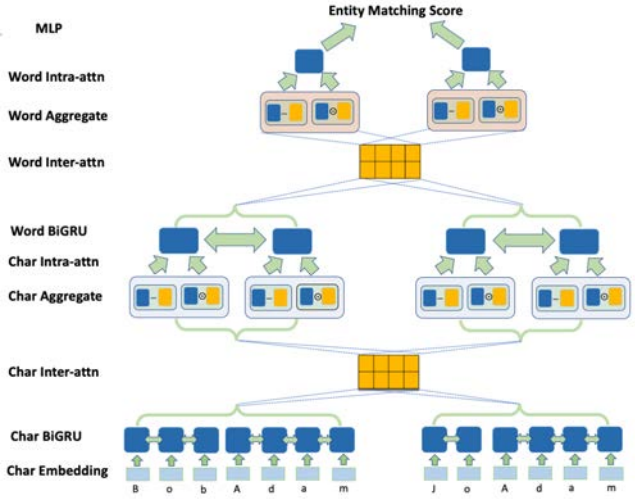


Figure 3: The hierarchical HI-EM model for attribute-level EM. The two input "Bob Adam" and "Jo Adam" at the bottom pass through a number of layers (GRU, attention, etc.), before producing a match score.

These observations motivate us to develop a hierarchical-EM model shown in Figure 3, which we will refer to as HI-EM for short. At a high level, the model has a hierarchical structure, which starts with character-level layers (the ones that start with "Char" in Figure 3), but also has upper-layers that explicitly capture word-level information (the layers that start with "Word"). With the hierarchical model, the character-level layers can not only capture fine-grained character-level variations, but also address the common out-of-vocabulary (OOV) issues. At the same time the word-level layers can explicitly leverage word-boundaries (separating characters in different words), which are especially beneficial for long input strings (e.g., data in types like address often have 5-10 words). We note that a similar idea of hierarchical models was recently explored in other contexts [74].

Additionally, for both the character-level and word-level, we introduce layers specifically designed for intra-input attention (within single input), and inter-input attention (between two input strings), which would help the model to learn character/word importance, as well as alignments between two input strings.

We now describe different layers of the model in turn in detail.

3.2.1 Character-level Layers for Word Representations.

We will first describe the 5 layers at the bottom of Figure 3. These are at the character-level to ultimately produce word-level representations, and their names all start with "Char". At a high level, we will first encode characters in the input, then look at the other input for alignments using attention, before aggregating to produce word representations for word-level layers.

Character Encoder. This part includes the first two layers: Char-Embedding, and Char-Bi-GRU. In the Char-Embedding layer, given a word $w_i, i \in [i, n]$, with its characters denoted as $c_{it}, t \in [1, l_i]$, we embed the characters to vectors through a character-embedding matrix W_e.

$$e_{it} = W_e * c_{it} \tag{1}$$

Then we pass the embedded vectors e_{it} to a recurrent neural network (RNN) block to obtain contextual information of the characters. In this work we use Bidirectional-Gated-Recurrent-Unit (BiGRU) [13] to capture both forward and backward information (similar to bidirectional LSTM [34]). The resulting character representation is denoted as hc_{it}.

$$hc_{it} = BiGRU(e_{it}) \tag{2}$$

Character Inter-input Attention. For each character representation hc_{it}, we adopt an inter-input attention layer to incorporate the character alignment $hc_j, j \in [1, l]$, where l refers to the length of whole character sequence from the other input. We use a bi-linear function with learned weight W_c to get the attention weights from the character sequence of the other input.

$$\alpha_j = hc_{it} * W_c * hc_j \tag{3}$$

For each character position it, the character information from the other attribute is summarized as

$$a_{it} = \sum_{j=1}^{l} \alpha_j hc_j \tag{4}$$

Character Aggregation and Intra-input Attention. For each character c_{it}, we produce a representation that is the concatenation of the element difference and multiplication between hc_{it} and a_{it}.

$$pc_{it} = [|hc_{it} - a_{it}|; hc_{it} \circ a_{it}] \tag{5}$$

We use the intra-attention layer to re-weight each combined character representation through a linear layer.

$$\beta_{it} = w_c * pc_{it} \tag{6}$$

The final representation for each word r_i is a weighted average of character representation.

$$r_i = \sum_{t=1}^{l_i} \beta_{it} * pc_{it} \tag{7}$$

We obtain word representation of each word from the two input strings, denoted as $r_i, i \in [1, n]$, and $r_j, j \in [1, m]$, where n, m are the total number words from the two input, respectively.

3.2.2 Word-level Layers for Attribute-value Representations.

On top of the character-level layers that produce word-level representations, we stack another set of word-level layers for overall attribute-value representations. These layers are designed similarly to include word encoding, inter-input attention, aggregation and finally intra-input attention, before producing a final representation for the full attribute value.

Word Encoder. We first use a BiGRU layer to contextualize each word representation $r_i, i \in [1, n]$.

$$hw_i = BiGRU(r_i) \tag{8}$$

And the same hw_j can be produced for $r_j, j \in [1, m]$.

Word Inter-input Attention. We have another inter-input attention layer to incorporate alignment information with the other input string $hw_j, j \in [1, m]$.

$$\alpha_j = hw_i * W_d * hw_j \tag{9}$$

$$a_i = \sum_{j=1}^{m} \alpha_j hw_j \tag{10}$$

Word Aggregation and Intra-input Attention. We again concatenate the element difference and multiplication of the word representation and the aligned word representation.

$$pw_i = [|hw_i - a_i|; hw_i \circ a_i] \tag{11}$$

Then we apply an intra-attention layer for final attribute-value representation z.

$$\beta_i = w_d * pw_i \tag{12}$$

$$z = \sum_{i=1}^{n} \beta_i * pw_i \tag{13}$$

We denote the final representations of the two input strings so computed as z_p and z_q, respectively.

3.2.3 Final Prediction.

The representation z_p, z_q for a pair of attribute values (P, Q) are concatenated and then pass through a multi-layer perceptron (MLP) layer to produce a final EM score.

$$score(P, Q) = MLP(z_p, z_q) \tag{14}$$

During training, we use logistic regression loss that averages over all N examples as the loss function.

$$loss = \frac{1}{N} \sum_{pos} log(\frac{1}{1 + e^{-score_{pos}}}) + \sum_{neg} log(\frac{1}{1 + e^{score_{neg}}}) \tag{15}$$

3.3 Transfer Learning for EM

For each attribute type T, we train a separate attribute-level EM model that captures the specific characteristics in T (e.g., synonymous tokens, token importance, etc.), which can then make highly accurate match/non-match decisions for data in type T.

However, even though we pre-train attribute-level EM for a large number of types, there will be attributes in EM tasks that are not in the known types. For those attributes we apply transfer-learning as follows, so that even for a new type not known a priori, we could quickly converge to a high-quality EM model.

We take the *union* of data in known attribute types, to build a general-purpose attribute-level EM model, which we will refer to as the *unified-model*. Such a model captures common variations general across many types (e.g., spell variations), and serves as a good starting point to train models for a new attribute-type. With limited training data for the new attribute-type, we take internal representations from the unified-model (right before the MLP layers in Figure 3), and add new MLP layers that can be fine-tuned using new training data to quickly converge to an EM model specific to the new type. Our experiments suggest that this transfer-learning approach produces high-quality results with limited training data.

Finally, for table level EM, we use a similar transfer-learning approach. Based on table attribute types, we use either type-specific attribute-level model or unified-model, to get internal representations every attribute pair (z in Equation (13)). We concatenate all representations, and add an MLP layer at the end for table-level EM. Such a model can be fine tuned end-to-end, using a small amount of table-level training data.

4 ATTRIBUTE TYPE DETECTION

In this section, we describe the first component of our system shown in Figure 2, which is for attribute type detection. Recall that for each value from input table column, we need to detect whether it belongs to known attribute-types T.

4.1 Training data preparation

Our training data used for attribute-type detection is similar to the data for attribute-level EM described in Section 3.1.

We use the same 40 common KB types, and 9 address types for type-detection. Note that each entity in the KB can be associated with one or more types. For example, entity "University of California" is of type "location", "organization", "educational institution", etc.; and entity "Harry Potter" is of type "written book", "film", "person", etc. Since the type hierarchy in the KB is such that types are not mutually exclusive but partially overlapping, and a string name can indeed belong to multiple types, we in this work formulate type-detection as a multi-hot classification problem – given an input string, predict all types it belongs to.

For each type T, we use names of entities in T, or $\{e \in T\}$ as positive examples for training. For negative examples, initially we use entities from $\{e' \notin T\}$. However, this turns out to be problematic, because KB types are often incomplete. For instance, while "University of California" has both types "organization" and "educational institution", another (smaller) university "Gonzaga University" only has type "educational institution" but not "organization" (which it a missing type)[2]. Note that because of the missing type, we may incorrectly use "Gonzaga University" as a negative example for "organization", which confuses the model.

To address this issue, we use a conservative approach to avoid selecting an entity $e \in T_1$ as a negative example for T_2, if its known type T_1 has positive correlation with T_2 (e.g., "organization" and "educational institution"). Specifically, for each pair of types T_1 and T_2, we compute their entity-instance-level point-wise mutual information (PMI) [46], defined as $\frac{|\{e|e \in T_1, e \in T_2\}| |\{e \in U\}|}{|\{e \in T_1\}| |\{e \in T_2\}|}$, where $\{e \in U\}$ is all the entities in the universe in the KB. If PMI> 0, then T_1 and T_2 are likely correlated and overlapping types. For instance, there are a substantial number of instances belonging to both "educational institution" and "organization", resulting in a positive PMI score. As such, we will not use *any* entity e of type "educational institution" as negative example of "organization", irrespective of whether e has type "organization". Formally, we use $\{e|e \notin T_1, \forall T_2 \ni e, PMI(T_1, T_2) < 0\}$ as the negative examples of T_1.

4.2 Type-detection Models

Figure 4 shows our Hierarchical entity-typing model (referred to as Hi-ET for short), for attribute type detection. We follow a similar hierarchical structure as the Hi-EM model.

We initially try to combine the type-detection task with the attribute-level EM task, using multi-task learning [23], given that they are similar intuitively. However our experiments show inferior quality in both tasks. We believe the reason lies in the fact that the "importance" of tokens in these two tasks are in fact opposite – to detect types for "University of California ", the token "University" is more important, but for EM (e.g., compare to "University of Colorado"), "California" is actually more important. Because of this reason we will design and train different models for type-detection.

[2]This problem of missing types can arise because types in KB are often generated from various sources (e.g., structured feeds, web data, etc.), which often do not cover Less popular entities as well as the head entities.

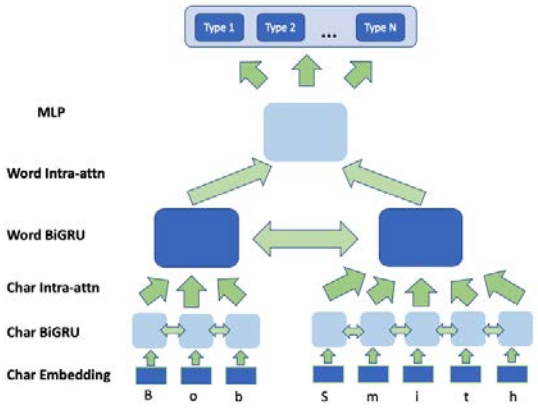


Figure 4: Our Hie-ET model to detect attribute types

Our type-detection model predicts 40 common KB types plus 9 address types. Since it can detect more than one type for each input value, we use one binary classifier for each target KB type, which is connected to the last layer of our model (Figure 4), for a total of 49 such classifiers. A benefit of this setup is that it can easily extend to new types, without needing to re-train existing models that have been tuned and tested.

The model in Figure 4 takes an input attribute value with n words $w_i, i \in [1, n]$, where each word contains l_i characters, written as $c_{it}, t \in [1, l_i]$. It produces C binary labels $(o_1, ...o_C) \in \{0, 1\}^C$ for the C pre-trained types. We describe each component of the model in Figure 4 below, using references to definitions in the attribute-level EM model from Section 3.

Word Level Representation. For each character $c_{it}, t \in [1, l_i]$, we first embed it into vectors e_{it} using Eq. 1 in Section 3. Then we use BiGRU layer (Eq. 2) to get contextual character representation hc_{it}. Using hc_{it}, we apply intra-attention (Eq. 6) to weight each contextual hidden state by importance. The final representation for each word r_i is a weighted average of representation hc_{it} (Eq. 7).

Attribute Level Representation. For each word representation $r_i, i \in [1, n]$, we first use another BiGRU layer (Eq. 8) to get contextual representation hw_i. Then we apply intra-attention (Eq. 12) to weight each contextual hidden state by importance. The final attribute representation z is weighted average of word contextual representation hw_i (Eq. 13).

Prediction layer. The representation z passes through an MLP layer. In our model, each binary output has it own MLP layer.

$$o_i = MLP_i(z), i \in [1, C] \tag{16}$$

Then the final output is the softmax of MLP output.

$$p_i = \text{softmax}(o_i) \tag{17}$$

Where $p_i = [p_{i0}, p_{i1}]$ indicates the probability of predicting the input value as the ith pre-trained type as true, and false, respectively.

During training, we use cross-entropy as our loss function, the final loss is the average of C classes over all examples.

$$L = -\frac{1}{N}\frac{1}{C}\sum_{ex}\sum_{i=1}^{C} y_{i0} log(p_{i0}) + y_{i1} log(p_{i1}) \tag{18}$$

Note that the model predicts types for one input value at a time. When predicting types for a column of k values, we simply compute an average score for the k values.

4.3 Transfer Learning for Type Detection

In our proposed EM system, we apply transfer-learning approach by directly using pre-trained type-detection models to detect if any table column/attribute corresponds to one of the known types.

Similar to transfer-learning in attribute-level EM (Section 3.3), we can also apply type-detection models to new types by first building a *unified-model*, which is trained using the union of data for all known types. For a new attribute type, we start from the representation of the unified-model, and use fine-tuning to produce an accurate model for a new type with little training data. Our experiments suggest that this is indeed the case – transfer-learning converges to high-quality type-detection models substantially faster than training from scratch.

5 EXPERIMENTS

In this section, we report experiments on different system components: Type Detection, Attribute-level EM and Table-level EM.

5.1 Type Detection Experiments

The goal of type detection is to accurately predict attribute types using pre-trained models. We report results on three experiments, entity-value type detection, table-column type detection, and transfer-learning for new types.

5.1.1 *Entity-value type-detection.*

Experimental setup. As discussed in Section 3.1, we use pre-trained models to detect 49 attribute-types, which include 40 common entity types from a KB, and 9 address types of different locales/markets ("en-us", "en-gb", etc.) from the "Maps" vertical of a search engine.

For each type T, we sub-sample at most 20K entities in T as positive examples (when a type has less than 20K entities we use all). For negative examples of T, we use entities not in T, filtered by the PMI procedure discussed in Section 4.1. This is to filter away negative examples that may be incorrect due to missing type labels (e.g., entity "Gonzaga University" has the type "educational institution" but is missing the type label "organization" in the KB. With PMI filtering we would not incorrectly use "Gonzaga University" as a negative example for "organization", since the two types are identified as overlapping/related). We randomly split the data into training (80%), development (10%) and test (10%).

We use PyTorch 0.4.1[55] to implement Hi-ET. We use random character embedding initialization with size 300. We use bidirectional GRU with 2 layers, and the size of each hidden layer is 300. For MLP, we use 2 linear layers of size 300 and ReLU as the non-linear activation function.

For training, we use batch size 32 and set dropout rate to 0.4 to help regularization. We use Adam [43] as the optimizer and use the default initial learning rate 0.001. We set gradient clipping threshold to 5.0 to increase stability. We finish the training after 5 epochs.

Experimental results. Figure 5 shows the precision-recall curves of entity-value type detection for the 49 types. In Figure 5(a) and 5(b), we can see that Hi-ET has high precision and recall for most of KB types, showing its ability to differentiate between entity values of different types. There are a few types (computer, architecture

venue, airline and sports facility) where the results are not as good. We found a main reason is the lack of positive training data – these types are small with less than 2000 entities in KB, which makes it difficult for deep models to learn. This is further exacerbated by the fact that entities from small types tend to be less popular and have more missing type information: if $e \in T_1$ but if the T_1 type is missing for e in the KB, we will incorrectly use e as a negative example of T_1, confusing the model. For small types and less popular entities, this tends to be more common, and is more difficult for our PMI-filtering approach to detect (Section 4.1).

Figure 5(c) shows precision/recall on 9 address types. It can be seen that despite the subtle differences of addresses from different markets (en-us, en-gb, en-ca, en-nz, etc.), where we intentionally remove obvious indicators such as all country tokens, our models still successfully differentiate addresses between different markets.

5.1.2 Table-column type-detection.

Experimental setup. For the table-column type-detection experiment, we use 1M Wikipedia tables as the test set, and evaluate precision/recall of two alternative methods: (1) our pre-trained Hi-ET models, which predict types using the average score of the first 10 values of each column; and (2) a keyword-based approach, which detects types based on keyword in Wikipedia table column-header (e.g., if a column-header contains the keyword "city" or "town", it is predicted to be of type city). Note that keyword is a strong baseline on Wikipedia, as Wikipedia tables are collaboratively edited by millions of editors [31], where column names are well-curated. In comparison, in enterprise CSV files and database tables, column headers are more likely to be cryptic or outright missing [24], which would make keyword search less effective.

We manually label 100 randomly selected columns, detected to be of type T. We report precision results from the 10 most common types in the interest of space.

Experimental results. Figure 7 shows that for most types (7/10), Hi-ET model has comparable or better results. However, quality results from Hi-ET can also be inferior to Keyword Search, notably for the entity type "food". Our analysis suggest that there is little sub-word pattern for this type (e.g., between apple, orange and banana), which is difficult for Hi-ET to generalize.

While Hi-ET is competitive for type-detection, we believe an ensemble of type-detection techniques that combine model-based, keyword-based, and even program-based [60, 72] approaches would be needed for best detection quality in practice.

5.1.3 Transfer-learning to new types.

Experimental setup. We also experiment whether our pre-trained type-detection models can be used in transfer-learning, to learn type-detection for other types faster and with less training examples. For this experiment, we use the same data from entity-value type detection (Section 5.1.1). We then select one type out of 49 types as the target-type for transfer-learning, and the remaining 48 types for pre-training. We compare two methods: (1) transfer-learning, using models fine-tuned from pre-trained type-detection models on 48 other types; and (2) learn-from-scratch, without using pre-trained models. For each method, we provide 200, 500 and 2000 examples from the target-type as training, and compare the resulting precision/recall curves.

Experimental results. Figure 6 compares the results with and without transfer-learning on 3 representative types. Similar results are observed in all other types (omitted here due to space constraints). We can see that with transfer-learning from pre-trained models, the model can learn a lot faster compared to learning-from-scratch, especially with little training data (e.g., 200 examples). We can see that results are better for types address and person, since these types have more regularity and are easier to learn. Data in type organization is more complex with more variations, which makes transfer-learning converge slower than other types.

Table 1: Statistics of types for attribute-level EM.

	Train	Dev	Test
Person	921230	3000	112312
Organization	271376	3000	31930
Movie	219574	3000	28832
Location	613792	3000	74062
Organism	5007	1629	1579
Local	85760	3000	10346
Book	21394	2745	2668
Software	2930	348	314

5.2 Attribute-Level Entity Matching

We conduct two experiments for attribute-level EM, pre-trained EM for known types, and transfer-learning for new types.

5.2.1 Experimental setup.

Data sets. As discussed in Section 3.1, for each entity e, we use synonymous names of e in KB (from the "alias" attribute, such as "Bill Gates" and "William Gates") as positive examples, and names of a different e' whose name is similar to e (by syntactic distance) as negative examples (e.g., "Bill Gates" and "Bill Clinton"). We split training pairs into train (80%) development (10%), and test (10%).

To evaluate pre-trained attribute-level EM of known types, we use 4 representative types: person, organization, location and movie, which show different types of name variations. The unified attribute-level model is trained using the union of these data.

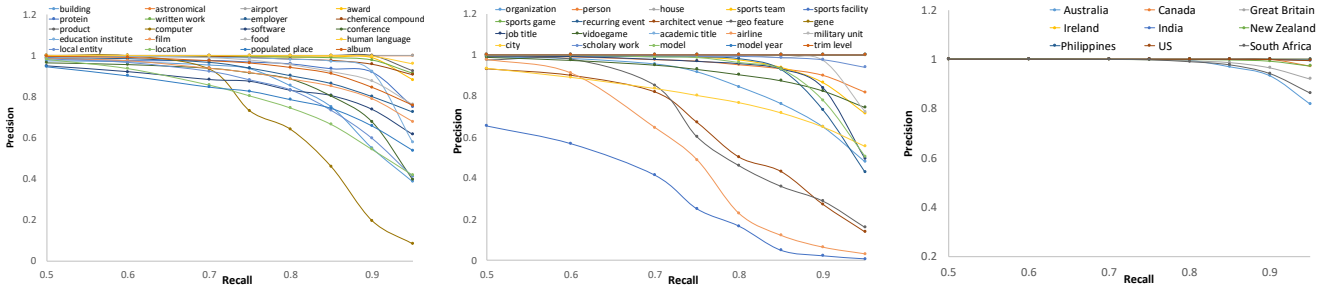
For transfer-learning, we start from the unified model, and report results on 4 different types: organism, local, book and software. We report results after fine-tuning using 200, 500 and 2000 labeled examples. Table 1 reports statistics of these types.

We evaluate the model quality using two metrics: Mean Reciprocal Rank (MRR) and precision/recall.

Methods compared. We compare the folowing methods:

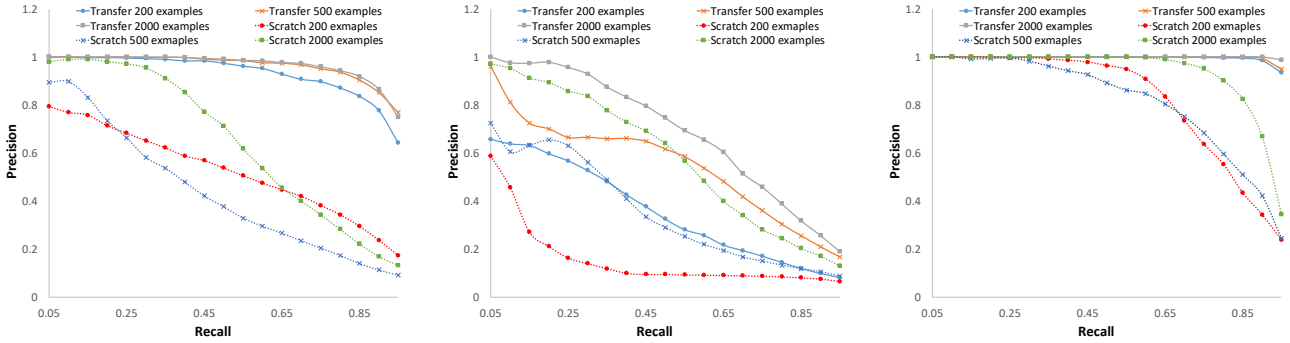
- **DSSM** [38] is one of the first deep models proposed for semantic similarity. DSSM uses DNN to represent each input in a continuous semantic space.
- **DeepER** [40] is proposed to use pre-trained word embedding for EM tasks.
- **DeepMatcher** [49] is also a deep model for EM problem with state-of-the-art results. We use its attribute matching component for attribute-level EM.
- **DeepMatcher (Unified)** is the same as DeepMatcher, but trained on unified data (union of data in different types).
- **Hi-EM** is the proposed EM model with hierarchical deep structure, trained using data for each type.
- **Hi-EM (Unified)** is Hi-EM trained on unified data.

Implementation details. To make fair comparison, we adopt same model settings for all methods. We set 300 as character/word



(a) First 20 Types from KB (b) 20 More Types from KB (c) 9 Types for Address in different locale

Figure 5: P/R curves of entity-value type-detection using Hi-ET model, for 40 KB entity types and 9 address types.



(a) Person (b) Organization (c) US Address

Figure 6: P/R curves for transfer learning using Hi-ET model, varying the amount of training data.

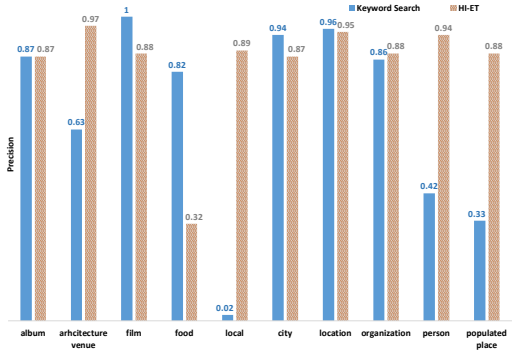


Figure 7: Comparision of Hi-ET and Keyword Search for Table-column type-detection across 10 most common types.

embedding size, 2 layers of size-300 for bidirectional GRU component, 2 layers of size 300 for MLP, and ReLU as nonlinear activation. For training, we use batch size of 32 and set dropout rate of 0.4. We use Adam as the optimizer with the default initial learning rate 0.001, and we set gradient clipping threshold at 5.0. We finish the training after 5 epochs (2 epochs for the unified data set). For each positive entity pair, we randomly select 5 negative examples for negative sampling.

For DSSM, we use character embedding, BiGRU for entity representations, and cosine similarity to compute match scores.

For DeepER, we use Glove [57] as the pre-trained word embedding, and fine-tune the embedding weights during training. For unknown words, we replace with 'UNK' token. We use BiGRU to get entity representations and MLP for final predictions.

For DeepMatcher, the authors present a design space of possible choices (with different neural network component). We report the configuration with the best performance in our experiments, which has character embedding, BiGRU component for attribute summarization, learnable distance (dot product) for attribute comparison and MLP for final predictions.

5.2.2 *Experimental Results.*

Pre-trained attribute-level EM for known types. Table 2 shows the MRR scores for attribute-level EM using pre-trained models on four different attribute types. The proposed Hi-EM produces better results than all other methods. DeepER does not perform well in this task, with lowest scores across all types. The reason is that DeepER uses word-based embedding with many OOV tokens. DSSM also has lower quality than DeepMatcher and Hi-EM, since it uses simple model architecture that does not consider interactions between two input strings. Finally, Hi-EM outperforms DeepMatcher in all types, showing the advantages of the hierarchical architecture, especially on complex attribute-types.

In the same table, we can see that Hi-EM (Unified) also achieves better quality than DeepMatcher (Unified). As expected, the unified models produce lower scores compared to the type-specific models.

Figure 8 reports the same experiments as above using precision/recall curves instead of MRR. The result is consistent with that

Table 2: MRR results for pre-trained attribute-level EM

	Person	Organization	Movie	Location
DSSM	0.888	0.850	0.844	0.853
Deep-ER	0.645	0.528	0.492	0.636
DeepMatcher	0.935	0.909	0.895	0.905
DeepMatcher (Unified)	0.924	0.893	0.894	0.896
Hi-EM	**0.943**	**0.925**	**0.924**	**0.911**
Hi-EM (Unified)	0.934	0.907	0.914	0.899

Table 3: MRR results for transfer-learning to new types

	Organism	Local	Book	Software
200 examples scratch	0.726	0.554	0.612	0.553
500 examples scratch	0.794	0.679	0.701	0.786
2000 examples scratch	0.828	0.804	0.790	0.810
0 example transfer	0.831	0.756	0.863	0.918
200 examples transfer	0.873	0.851	0.865	0.903
500 examples transfer	**0.884**	**0.853**	0.871	0.915
2000 examples transfer	0.881	0.849	**0.880**	**0.937**

of MRR, except in the movie type, where Hi-EM slightly underperforms DeepMatcher in some regions of the curve. An inspection of the errors suggest that the movie type has more synonymous name pairs that are semantic in nature – for example, the movie "Love Song" is also known as "Comrades: Almost a Love Story"; and "Star Crash" is also known as "Star Battle Encounters" or "Stella Star". These positive examples are all very specific and hard to generalize. At training time, Hi-EM overfits on these semantic examples, and produces high match scores for certain negative examples from the test data. Note that because the corresponding true-positive pairs have even higher match scores, in the MRR evaluation these high-scoring negative examples would not affect results. The P/R evaluation on the other hand, are more sensitive to the high-scoring negative examples, which affects precision.

We find other types of errors include name pairs that are inherently ambiguous (e.g., name pairs like "Rick Baker" and "Richard A. Baker" are marked as negative, but similar pairs like "Rick Barnes" and "Richard D. Barnes" would also be marked as positive), which makes it difficult to predict accurately. Finally abbreviation are also difficult to predict – Hi-EM is able to predict some pairs correctly (e.g., "IBM" and "International Business Machines Corp."), but get others wrong (e.g, "University of Geneva" and "UNIGE").

Transfer-learning for new types. Table 3 compares the MRR results between transfer-learning and train-from-scratch, with varying numbers of training data. We note that transfer-learning clearly helps, as there is a significant difference between k-example-transfer and k-example-scratch (for the same k). The difference is more pronounced when using fewer training examples. We also note that for many types (Organism, Book and Software), results from pre-trained unified-model (the line marked as "0 example transfer") already outperforms learn-from-scratch with 2000 examples, which is all the training examples we provide in this experiment.

Figure 9 reports precision/recall curves of the same experiment. We observe that these results are consistent with the MRR results.

5.3 Table-level Entity Matching

In this section, we evaluate our end-to-end EM on table data and compare with existing EM methods.

5.3.1 Experiment Setup.

Methods compared. We compare the following EM methods:

- **Magellan** [44] is a state-of-the-art feature-based EM system. We obtain it from GitHub[3] and use default settings.
- **DeepMatcher** [49] is a state-of-the-art deep EM model. We train the model from scratch, and use the same settings for attribute representations from attribute-level EM.
- **Hi-EM** is our hierarchical EM model trained from scratch, using the same settings from attribute-level EM.
- **Hi-EM (Unified)** is the same as Hi-EM, except that for all attributes we start with representations from the same unified attribute-level EM models that are pre-trained, and fine tune table-level EM based on table-level training data.
- **Hi-EM (Type)** is the same as Hi-EM, except that when attributes are detected as known types, we start with representations from type-specific attribute-level EM models, and fine tune table-level EM based on training data. This is the same as our end-to-end architecture outlined in Figure 2.

For both Hi-EM and DeepMatcher, we concatenate the representations of all attribute pairs and apply a 2 layer MLP of size 300 for final predictions.

Data sets. We use labeled data from a repository of EM tasks[4], which were also used in prior work [44]. There are a total of 24 EM tasks, each with a pair of tables, where some of the record pairs between the two tables were manually labeled as match/non-match. We exclude tasks whose corresponding data have quality issues (mainly due to formatting) and could not be run using the existing Magellan system, and ones that are very easy (e.g. matches are almost all exact, and all methods have over 0.95 $F1$). We use 8 remaining data sets that are more challenging EM tasks.

We evaluate model performance with varying number of training data. Specifically, for each data set, we randomly sample 5%, 10% and 20% labeled data as training, and use the remaining 80% as testing. We report $F1$ score on the test data. Note that the training data in most cases translate into just a few dozen labeled record pairs (there is one with only 13 training pairs), which is a small amount of training data.

To reduce randomness, we run deep models three times with different random seeds, and report an average $F1$. We keep the same random seed in each run between different deep models.

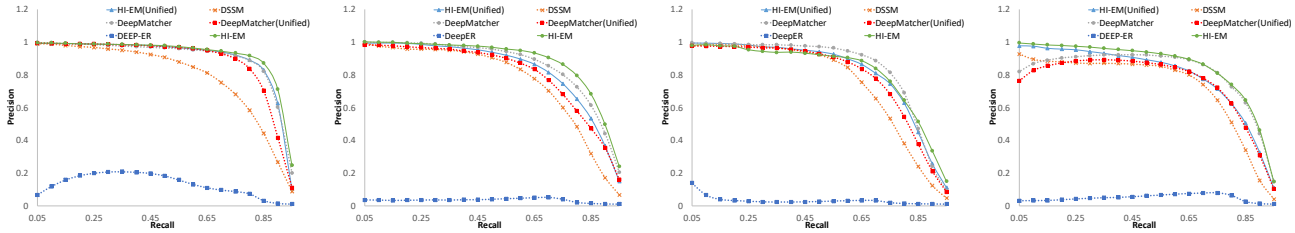
5.3.2 Experimental Results.

Table 4 shows $F1$ score across different data sets with varying amounts of training data. First, the Hi-EM (Type) achieves better quality than Hi-EM (Scratch) in 23 out of 24 settings, and it outperforms DeepMacher in 22 out of 24 settings, showing the benefit of a pre-training approach even with a small amount of training data.

Between Hi-EM (Type) and Hi-EM (Unified), Hi-EM (Type) produces better quality in 12/24 settings, and there are 6/24 settings for which the two methods are identical (since no columns are detected to be of known types). This is consistent with our finding in attribute-level EM that type-specific models are more accurate.

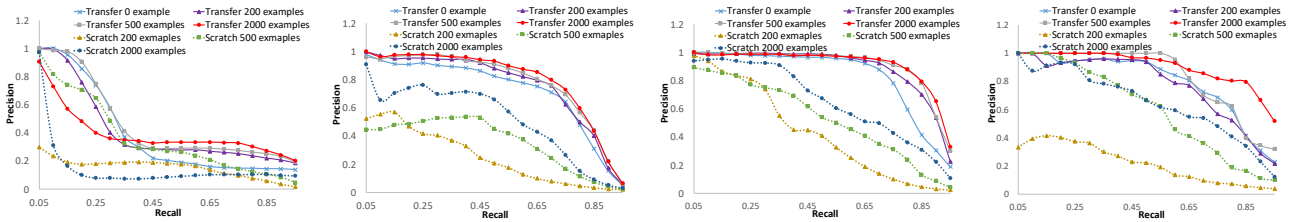
[3] https://github.com/anhaidgroup/py_stringmatching

[4] Available at https://sites.google.com/site/anhaidgroup/useful-stuff/data



(a) Person (b) Organization (c) Movie (d) Location

Figure 8: P/R curves of different models for attribute-level EM, on 4 types of attributes.



(a) Organism (b) Local (c) Book (d) Software

Figure 9: P/R curves of transfer-learning for new types using HI-EM, with varying training data for 4 different attribute-types.

Table 4: F1 for table-level EM with varying training data

	HI-EM Type	HI-EM Unified	HI-EM Scratch	Deep-Matcher	Magellan
Amz-BN 5%	0.510	0.450	0.460	0.200	**0.720**
Amz-BN 10%	**0.853**	0.810	0.657	0.350	0.820
Amz-BN 20%	**0.854**	0.790	0.646	0.650	0.800
Bk-Bw 5%	**0.820**	**0.820**	0.517	0.750	0.250
Bk-Bw 10%	**0.890**	**0.890**	0.820	0.850	0.400
Bk-Bw 20%	0.790	0.790	0.854	**0.880**	0.410
GR-BN 5%	**0.778**	0.720	0.612	0.680	0.710
GR-BN 10%	**0.809**	0.780	0.634	0.670	0.680
GR-BN 20%	**0.843**	0.830	0.661	0.650	0.650
YP-Yelp 5%	**0.955**	0.930	0.755	0.850	0.810
YP-Yelp 10%	**0.955**	0.940	0.721	0.850	0.810
YP-Yelp 20%	0.950	**0.970**	0.792	0.860	0.910
Amz-RT 5%	**0.852**	0.825	0.617	0.810	0.822
Amz-RT 10%	**0.868**	0.861	0.679	0.812	0.833
Amz-RT 20%	**0.853**	0.845	0.648	0.773	0.833
ANI-MAL 5%	**0.989**	**0.989**	0.927	0.956	0.633
ANI-MAL 10%	**0.989**	**0.989**	0.967	0.963	0.518
ANI-MAL 20%	**0.989**	**0.989**	0.967	0.964	0.481
BN-Half 5%	**0.932**	0.929	0.917	0.913	0.677
BN-Half 10%	0.936	**0.939**	0.919	0.917	0.744
BN-Half 20%	0.940	**0.953**	0.905	0.908	0.899
RE-IMDB 5%	0.803	0.811	0.772	0.803	**0.899**
RE-IMDB 10%	0.880	**0.885**	0.756	0.883	0.881
RE-IMDB 20%	0.863	0.904	0.842	0.860	**0.937**

Compared to the feature-based EM Magellan, HI-EM (Type) outperforms Magellan in 20/24 settings. On the other hand, DeepMatcher is comparable to Magellan, which is consistent with what is reported in [49].

6 RELATED WORKS

In this section, we describe existing work in three related areas: Entity Matching, Deep Learning in NLP, and Transfer Learning.

Entity Matching. Entity matching, also known as entity resolution, fuzzy join, record linkage, among other names, has been a long-standing problem in the literature of data mining and data integration [28, 30, 33, 45]. Various techniques have been proposed, including ML-based approaches [10, 15, 61, 65] (based on positive/negative pairs labeled by users), and constraint-based methods [11, 19, 63, 70] (based on rules programmed by experts). Recently, two deep EM models, DeepMatcher [40], and DeepER [49], have been proposed and are shown to achieve better result quality.

We observe that most existing EM approaches require a large amount of training data, which is a significant barrier to wider adoption. We in this work propose a hands-off Auto-EM architecture, which leverages type-detection and attribute-level EM models that are pre-trained on a large amount of data from KBs. It is shown to achieve high EM quality with little training data.

Entity type detection. Unlike the literature on type classification in NLP (e.g., [50, 75]), which typically relies on natural language contexts, our task of entity type detection is for database tables and columns, where natural language contexts are absent. Our approach leverages only characteristics of entities, in a setting similar to [72].

Deep model in NLP. Tremendous progress have been made in applying deep models to NLP. Text-classification and similarity problems are particularly relevant to our problem.

Classification. Text classification [9] is a fundamental problem in NLP, for which neural network models are developed, including Convolution Neural Network (CNN) [42] and Recurrent Neural Network (RNN)[52]. Recently self-attention [68] is used as additional layer to improve performance, and [74] adopts hierarechical structure with self-attention over RNN for document classification.

Similarity. Learning textual similarity between two input is important in NLP, with applications including Natural Language Inference (NLI) [17, 71], Answer Selection (AS) [73], etc. Early deep models [38] treat each input independently. Recently approaches leverages input pairs [20, 35, 54, 62], and solve these tasks with a

similar architecture of input embedding layer, context encoding layer, interaction and attention layer, and finally output classification layer [47].

Transfer learning. Transfer learning has a long history [53], where the idea is to transfer knowledge from a problem with abundant training data, to a target-problem that is related but with limited data. Transfer learning has been successfully applied to domains such as computer vision and NLP [14].

In NLP, transfer learning approaches include the well-known word embedding [41, 48]. Recent approaches propose pre-train models with language model objectives [25, 37], with fine-tuning for specific tasks, which has achieved great success [27, 59].

7 CONCLUSION AND FUTURE WORK

In this work we propose an end-to-end system for EM, that leverages models pre-trained using rich KB data. With the help of transfer-learning, we show that table-level EM tasks can be trained with little labeled data.

Our deep models are not currently good at detecting attributes involving numeric values (e.g., currency and measurements). Using programmatic methods to detect and featurize these attributes could complement the deep models for better overall EM results, and are interesting directions for future work.

REFERENCES

[1] [n. d.]. AWS Lake Formation ML Transforms: FindMatches/Deduplication. https://aws.amazon.com/lake-formation/faqs/.

[2] [n. d.]. Azure Machine Learning Data Prep SDK. https://docs.microsoft.com/en-us/python/api/overview/azure/dataprep/intro?view=azure-dataprep-py.

[3] [n. d.]. Bing Entity Search API. https://azure.microsoft.com/en-us/services/cognitive-services/bing-entity-search-api/.

[4] [n. d.]. Excel Fuzzy Lookup Addin. https://www.microsoft.com/en-us/download/details.aspx?id=15011.

[5] [n. d.]. Google Knowledge Graphs. https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html.

[6] [n. d.]. Microsoft Dynamics 365 for Customer Insights. https://dynamics.microsoft.com/en-us/ai/.

[7] [n. d.]. Salesforce Customer 360. https://www.salesforce.com/blog/2018/09/what-is-salesforce-customer-360.html.

[8] [n. d.]. Self-Service Data Preparation, Worldwide, 2016. https://www.gartner.com/doc/3204817/forecast-snapshot-selfservice-data-preparation.

[9] Charu C Aggarwal and ChengXiang Zhai. 2012. A survey of text classification algorithms. In *Mining text data*. Springer, 163–222.

[10] Arvind Arasu, Michaela Götz, and Raghav Kaushik. 2010. On active learning of record matching packages. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. ACM, 783–794.

[11] Arvind Arasu, Christopher Ré, and Dan Suciu. 2009. Large-scale deduplication with constraints using dedupalog. In *IEEE International Conference on Data Engineering*. IEEE, 952–963.

[12] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.

[13] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[14] Yoshua Bengio. 2012. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*. 17–36.

[15] Mikhail Bilenko and Raymond J Mooney. 2003. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 39–48.

[16] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM, 1247–1250.

[17] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 632–642.

[18] Kaushik Chakrabarti, Surajit Chaudhuri, Zhimin Chen, Kris Ganjam, Yeye He, and W Redmond. 2016. Data services leveraging Bing's data assets. *IEEE Data Eng. Bull.* 39, 3 (2016), 15–28.

[19] Surajit Chaudhuri, Anish Das Sarma, Venkatesh Ganti, and Raghav Kaushik. 2007. Leveraging aggregate constraints for deduplication. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. ACM, 437–448.

[20] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for Natural Language Inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1657–1668. https://doi.org/10.18653/v1/P17-1152

[21] Tao Cheng, Hady W Lauw, and Stelios Paparizos. 2012. Entity synonyms for structured web search. *IEEE transactions on knowledge and data engineering* 24, 10 (2012), 1862–1875.

[22] Peter Christen. 2009. Development and user experiences of an open source data cleaning, deduplication and record linkage system. *ACM SIGKDD Explorations Newsletter* 11, 1 (2009), 39–48.

[23] Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*. ACM, 160–167.

[24] Eli Cortez, Philip A Bernstein, Yeye He, and Lev Novik. 2015. Annotating database schemas to help enterprise search. *Proceedings of the VLDB Endowment* 8, 12 (2015), 1936–1939.

[25] Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in neural information processing systems*. 3079–3087.

[26] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* (2018).

[27] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* (2018).

[28] AnHai Doan and Alon Y Halevy. 2005. Semantic integration research in the database community: A brief survey. *AI magazine* 26, 1 (2005), 83.

[29] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 601–610.

[30] Ahmed K Elmagarmid, Panagiotis G Ipeirotis, and Vassilios S Verykios. 2007. Duplicate record detection: A survey. *IEEE Transactions on knowledge and data engineering* 19, 1 (2007), 1–16.

[31] Yérali Gandica, J Carvalho, and F Sampaio dos Aidos. 2015. Wikipedia editing dynamics. *Physical Review E* 91, 1 (2015), 012824.

[32] Yuqing Gao, Jisheng Liang, Benjamin Han, Mohamed Yakout, and Ahmed Mohamed. 2018. Building a Large-scale, Accurate and Fresh Knowledge Graph. In *KDD*.

[33] Lise Getoor and Ashwin Machanavajjhala. 2012. Entity resolution: theory, practice & open challenges. *Proceedings of the VLDB Endowment* 5, 12 (2012), 2018–2019.

[34] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. 2013. Hybrid speech recognition with deep bidirectional LSTM. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 273–278.

[35] Hua He and Jimmy Lin. 2016. Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 937–948. https://doi.org/10.18653/v1/N16-1108

[36] Yeye He, Xu Chu, Kris Ganjam, Yudian Zheng, Vivek Narasayya, and Surajit Chaudhuri. 2018. Transform-data-by-example (TDE): An Extensible Search Engine for Data Transformations. *Proc. VLDB Endow.* 11, 10 (June 2018), 1165–1177. https://doi.org/10.14778/3231751.3231766

[37] Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 328–339.

[38] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2333–2338.

[39] Zhipeng Huang and Yeye He. 2018. Auto-detect: Data-driven error detection in tables. In *Proceedings of the 2018 International Conference on Management of Data*. ACM, 1377–1392.

[40] Muhammad Ebraheem Saravanan Thirumuruganathan Shafiq Joty and Mourad Ouzzani Nan Tang. 2018. Distributed Representations of Tuples for Entity Resolution. *Proceedings of the VLDB Endowment* 11, 11 (2018).

[41] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* (2016).

[42] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language*

Processing (EMNLP). Association for Computational Linguistics, 1746–1751. https://doi.org/10.3115/v1/D14-1181

[43] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[44] Pradap Konda, Sanjib Das, Paul Suganthan GC, AnHai Doan, Adel Ardalan, Jeffrey R Ballard, Han Li, Fatemah Panahi, Haojun Zhang, Jeff Naughton, et al. 2016. Magellan: Toward building entity matching management systems. *Proceedings of the VLDB Endowment* 9, 12 (2016), 1197–1208.

[45] Hanna Köpcke and Erhard Rahm. 2010. Frameworks for entity matching: A comparison. *Data & Knowledge Engineering* 69, 2 (2010), 197–210.

[46] Solomon Kullback. 1997. *Information theory and statistics*. Courier Corporation.

[47] Wei Lan, Wuweiand Xu. 2018. Neural Network Models for Paraphrase Identification, Semantic Textual Similarity, Natural Language Inference, and Question Answering. In *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 3890–3902. http://aclweb.org/anthology/C18-1328

[48] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[49] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep Learning for Entity Matching: A Design Space Exploration. In *Proceedings of the 2018 International Conference on Management of Data*. ACM, 19–34.

[50] David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes* 30, 1 (2007), 3–26.

[51] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2016), 11–33.

[52] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* 24, 4 (2016), 694–707.

[53] Sinno Jialin Pan, Qiang Yang, et al. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[54] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A Decomposable Attention Model for Natural Language Inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2249–2255.

[55] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS-W*.

[56] Heiko Paulheim. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web* 8, 3 (2017), 489–508.

[57] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[58] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

[59] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/language-unsupervised/language_ understanding_paper. pdf* (2018).

[60] Vijayshankar Raman and Joseph M Hellerstein. 2001. Potter's wheel: An interactive data cleaning system. In *VLDB*, Vol. 1. 381–390.

[61] Sunita Sarawagi and Anuradha Bhamidipaty. 2002. Interactive deduplication using active learning. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 269–278.

[62] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603* (2016).

[63] Warren Shen, Xin Li, and AnHai Doan. 2005. Constraint-based entity matching. In *AAAI*. 862–867.

[64] Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering* 27, 2 (2015), 443–460.

[65] Parag Singla and Pedro Domingos. 2006. Entity resolution with markov logic. In *Data Mining, 2006. ICDM'06. Sixth International Conference on*. IEEE, 572–582.

[66] Michael Stonebraker and Ihab F Ilyas. 2018. Data Integration: The Current Status and the Way Forward. *IEEE Data Eng. Bull.* 41, 2 (2018), 3–9.

[67] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 697–706.

[68] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[69] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.

[70] Steven Euijong Whang, Omar Benjelloun, and Hector Garcia-Molina. 2009. Generic entity resolution with negative rules. *The VLDB Journal* 18, 6 (2009), 1261.

[71] Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 1112–1122. http://aclweb.org/anthology/N18-1101

[72] Cong Yan and Yeye He. 2018. Synthesizing type-detection logic for rich semantic data types using open-source code. In *Proceedings of the 2018 International Conference on Management of Data*. ACM, 35–50.

[73] Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2013–2018.

[74] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[75] Dani Yogatama, Daniel Gillick, and Nevena Lazic. 2015. Embedding methods for fine grained entity type classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 291–296.

[76] Erkang Zhu, Yeye He, and Surajit Chaudhuri. 2017. Auto-join: Joining tables by leveraging transformations. *Proceedings of the VLDB Endowment* 10, 10 (2017), 1034–1045.

Review Response Generation in E-Commerce Platforms with External Product Information

Lujun Zhao*
Fudan University
Shanghai, China
ljzhao16@fudan.edu.cn

Kaisong Song
Alibaba Group
Hangzhou, Zhejiang, China
kaisong.sks@alibaba-inc.com

Changlong Sun
Alibaba Group
Hangzhou, Zhejiang, China
changlong.scl@taobao.com

Qi Zhang
Fudan University
Shanghai, China
qz@fudan.edu.cn

Xuanjing Huang
Fudan University
Shanghai, China
xjhuang@fudan.edu.cn

Xiaozhong Liu†
Indiana University Bloomington
Bloomington, Indiana, USA
liu237@indiana.edu

ABSTRACT

"User reviews" are becoming an essential component of e-commerce. When buyers write a negative or doubting review, ideally, the sellers need to quickly give a response to minimize the potential impact. When the number of reviews is growing at a frightening speed, there is an urgent need to build a response writing assistant for customer service providers. In order to generate high-quality responses, the algorithm needs to consume and understand the information from both the original review and the target product. The classical sequence-to-sequence (Seq2Seq) methods can hardly satisfy this requirement. In this study, we propose a novel deep neural network model based on the Seq2Seq framework for the review response generation task in e-commerce platforms, which can incorporate product information by a gated multi-source attention mechanism and a copy mechanism. Moreover, we employ a reinforcement learning technique to reduce the exposure bias problem. To evaluate the proposed model, we constructed a large-scale dataset from a popular e-commerce website, which contains product information. Empirical studies on both automatic evaluation metrics and human annotations show that the proposed model can generate informative and diverse responses, significantly outperforming state-of-the-art text generation models.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Natural language processing**; **Natural language generation**; • **Information systems** → *Electronic commerce*.

KEYWORDS

Review Response Generation, Neural Network, Sequence to Sequence Model, Gated Multi-Source Attention Mechanism, Reinforcement Learning

*Work done as an intern at Alibaba.
†Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313581

ACM Reference Format:
Lujun Zhao, Kaisong Song, Changlong Sun, Qi Zhang, Xuanjing Huang, and Xiaozhong Liu. 2019. Review Response Generation in E-Commerce Platforms with External Product Information. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313581

1 INTRODUCTION

Over the past few years, online shopping has experienced unimaginable growth. In e-commerce platforms, such as *Amazon*, *eBay* and *Taobao*, massive reviews from users are becoming increasingly important. From the seller viewpoint, however, reviews can be a double-edged sword. On the one hand, the more reviews, the more buys, but on the other hand, sellers have little or no control over buyer opinion. Therefore, efficient and constructive response to the target review/buyer can be critically important. If the buyers, especially those who send negative or doubting reviews, do not get a quick reply, they and other potential customers may turn to alternatives, which is unfavorable to the sellers and platforms.

Statistics[1] from a world leading e-commerce platform shows that sellers who provide high quality review responses could achieve a higher selling volume than those who rarely provide responses. However, only 2.98% of the reviews and 5.68% of the negative reviews receive responses. Furthermore, a large number of responses are universal replies which are based on the pre-defined templates/messages. Ideally, sellers should provide high-quality responses to address the personal needs from the target reviews. However, such cost can be prohibitive for most small-businesses, which inspires us to build a response writing assistant. After the algorithm automatically generates some candidate responses, the customer services could choose one and polish it, which makes the procedure more friendly and efficient.

Recently, deep neural network based methods have widely used in many natural language generation (NLG) tasks and have achieved great success, such as machine translation [3, 4, 10, 26], dialogue generation [23, 24], and text summarization [17, 22, 30]. Most of these models belong to a family of sequence-to-sequence (Seq2Seq) framework, which is composed of two parts: an encoder and a decoder. The encoder reads and encodes a source sentence into a hidden vector and the decoder outputs a target sentence based on

[1]More detail can be found in Appendices A.1.

Review: The quality is not very good and the sleeves will pill. It looks like the fabric is bad.

Seq2Seq+Attention: Thank you for your support and feedback. We will continue to work hard to provide you with better service and look forward to your next visit!

Real Response: Dear customer, thank you for choosing NAISHITU flagship store. Due to the composition of polyester, slight pilling is a normal phenomenon. It is recommended that you use fabric shaver to deal with. Looking forward to your next visit.

Figure 1: Given a review, the response generated by "Seq2Seq +Attention" model is trivial and universal. In contrast, the real response written by human contains some product information, like the *brand* (NAISHITU) and *material* (polyester).

Field	Value
Brand	NAISHITU
Material	polyester 57% cotton 43%
Process	easy care finishing
Style	casual style
Color	black
Size	185 / 2XL
...	...

Figure 2: An example of product information collected as a table.

the hidden vector. The attention mechanism [3, 16], which selectively focuses on parts of the source sentence during generation, is integrated into the Seq2Seq framework to improve the quality of long sequence generation.

Despite its enormous success, for review response generation task, however, the standard Seq2Seq+Attention model can only employ the textual reviews when generating responses. Intuitively, each review also has an associated product, and the product information can be critical and necessary to generate high-quality responses. Figure 1 illustrates an example. In this example, the given review is a negative review. Without employing the product information, the Seq2Seq+Attention model tends to generate trivial and universal responses like "*thank you for your support and feedback*", or "*we will continue to work hard to provide you with better service*" due to the high frequency of these patterns in the training data. These universal responses will give users a very bad user experience. In contrast, in real world, an experienced seller gives a response according to the corresponding product information, like the *brand* (NAISHITU) and the *material* (polyester). Without taking such product information into consideration, the algorithm can hardly generate such high-quality responses. Hence, integrating product information into the generation process has become a crucial part of the task.

In this work, the product information is collected as a factual table which contains many field-value records and an example is given in Figure 2. We follow Lebret et al. [12] to encode this table. The proposed model utilizes product information via a gated multi-source attention mechanism and a copy mechanism. The novel attention mechanism includes two attentions: review attention and product attention. For generation, we first obtain the review context vector by review attention which follows the existing attention techniques. Then, the review context vector is used to calculate the product context vector by product attention, which simulates the human response process of finding useful information in a product according to the issue mentioned in review. Finally, we obtain the final context vector through a Gated Multimodal Unit (GMU) [1], which is used to learn fusion transformations from multiple sources of information. By incorporating the copy mechanism [9], the decoder could directly copy the sub-sequences of review or product information into the response. Moreover, we employ self-critical sequence training (SCST) [21], a policy gradient reinforcement learning technique, to train the model, which reduces the exposure bias problem [20]. To evaluate the proposed model, we constructed a large-scale dataset from Taobao, which contained 100K (review, product information, response) triples. Experimental results on this dataset show the effectiveness of our method.

In summary, the contributions of this paper include the following three aspects:

(1) We introduce the task of review response generation in e-commerce platforms, which is valuable for customer service providers, but has not yet been well-studied.
(2) In order to address this problem, we propose leveraging a gated multi-source attention mechanism and a copy mechanism to integrate product information and employ policy gradient reinforcement learning in the training procedure to reduce the exposure bias problem.
(3) Experimental results based on both automatic evaluation metrics and human annotations demonstrate that the proposed model can generate informative and diverse responses, significantly outperforming state-of-the-art NLG models. We also release the large-scale review response generation dataset to enable future investigations.

2 PRELIMINARIES

In this section, we first formulate the review response generation task formally, and then introduce the field representation which will be used by our model.

Symbol	Description
$\boldsymbol{X}$	a review
$\boldsymbol{x}_i$	i-th word in review $\boldsymbol{X}$
$\boldsymbol{Y}$	a response
$\boldsymbol{y}_i$	i-th word in response $\boldsymbol{Y}$
$\boldsymbol{T}$	a table representing product information
$\boldsymbol{t}_i$	i-th word in the sequence composed by all value
$\boldsymbol{f}_i$	the field name of i-th word $\boldsymbol{t}_i$
$\boldsymbol{p}_i^+, \boldsymbol{p}_i^-$	the position of i-th word $\boldsymbol{t}_i$
$\boldsymbol{z}_i$	the field representation of i-th word $\boldsymbol{t}_i$
$\boldsymbol{h}_i^X$	hidden state of the review encoder
$\boldsymbol{h}_i^T$	hidden state of the product information encoder
$\boldsymbol{h}_i^Y$	hidden state of the response decoder
$\boldsymbol{c}_i^X$	the review context vector
$\boldsymbol{c}_i^T$	the product information context vector
$\boldsymbol{c}_i$	the final context vector

Table 1: Commonly used notations in this paper.

Word	Field Representation
NAISHITU	(**Brand**, 1, 1)
polyester	(**Material**, 1, 4)
57%	(**Material**, 2, 3)
cotton	(**Material**, 3, 2)
...	...
casual	(**Style**, 1, 2)
style	(**Style**, 2, 1)
...	...

Figure 3: The table is represented as a sequence of words (the first column) and their corresponding field representation (the second column).

2.1 Task Definition

We first introduce our key notations used in this paper. Table 1 lists the main notation we use. Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ denote a review of length n, where $\boldsymbol{x}_i \in \mathbb{R}^{d_w}$ is the word embedding of i-th word. In the same way, let $\boldsymbol{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_m\}$ denote the corresponding response of length m. The product information is represented as a table $\boldsymbol{T}$, which is a combination of many field-value records. Figure 2 shows an example. In this example, we only show six fields: *brand*, *material*, *process*, *style*, *color* and *size*. Each value is a sequence of words and all values are concatenated as a long sequence $\{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_k\}$ (the first column in Figure 3) where $\boldsymbol{t}_i \in \mathbb{R}^{d_w}$ is the word embedding vector. To incorporate the field information, each word $\boldsymbol{t}_i$ also has a corresponding field representation $\boldsymbol{z}_i$ (the second column in Figure 3), which will be detailed in the next subsection. Given a review $\boldsymbol{X}$ and a table $\boldsymbol{T}$ representing the associated product information, the aim is to build a model that can generate a response $\boldsymbol{Y}$.

We formulate the response generation as a probabilistic model, which is trained to maximize the generation probability of $\boldsymbol{Y}$ conditioned on $\boldsymbol{X}$ and $\boldsymbol{T}$:

$$\begin{aligned} &\operatorname{argmax} P(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{T}) \\ &= \operatorname{argmax} P(\boldsymbol{y}_1|\boldsymbol{X}, \boldsymbol{T}) \prod_{i=2}^{m} P(\boldsymbol{y}_i|\boldsymbol{y}_{1:i-1}, \boldsymbol{X}, \boldsymbol{T}). \end{aligned} \tag{1}$$

2.2 Field Representation

As seen in Figure 2 and Figure 3, the value of each field in the product information is split into separate words and the entire table is transformed into a large sequence $\{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_k\}$. Besides the value information, the field information is also crucial, especially for the product attention. For example, an user may mention "material" in his review, but not mention "polyester". If we know that the corresponding field of "polyester" is "material", it will be easier for the attention mechanism to work well. Lebret, Grangier, and Auli [12] proposed representing value's field information by its corresponding field name and its position in the field. The field representation of a word $\boldsymbol{t}_i$ in the table is described by a {field, position} pairs:

$$\boldsymbol{z}_i = \{\boldsymbol{f}_i, \boldsymbol{p}_i\}. \tag{2}$$

The pair $\{\boldsymbol{f}_i, \boldsymbol{p}_i\}$ indicates that $\boldsymbol{t}_i$ occurs in field $\boldsymbol{f}_i$ at position $\boldsymbol{p}_i$. The position $\boldsymbol{p}_i$ can be further divided as $(\boldsymbol{p}_i^+, \boldsymbol{p}_i^-)$ which represents the positions of the word $\boldsymbol{t}_i$ counted from the beginning and the end of the field respectively:

$$\boldsymbol{z}_i = \{\boldsymbol{f}_i, \boldsymbol{p}_i^+, \boldsymbol{p}_i^-\}, \tag{3}$$

where $\boldsymbol{f}_i \in \mathbb{R}^{d_f}$ and $\boldsymbol{p}_i^+, \boldsymbol{p}_i^- \in \mathbb{R}^{d_p}$ are the embedding vector of field name and positions, respectively. d_f is the field name embedding size and d_p is the position embedding size. For example in Figure 2, the second field-value record is (**Material**, polyester 57% cotton 43%). The word "polyester" is the first word of field "**Material**", counted from the beginning, and the fourth word counted from the end. So the field representation of "polyester" is (**Material**, 1, 4), as shown in the second row of Figure 3.

3 THE PROPOSED MODEL

3.1 Network Architecture

In this section, we describe the network architecture used in this paper, which is illustrated in Figure 4. The proposed model mainly includes four parts: an encoder to read the review (blue), an encoder to read corresponding product information (green), a Gated Multimodal Unit (GMU) (yellow) to learn fusion transformations, and a decoder to generate the response (red).

Review Encoder

We first transform each word in a given review to a one-hot vector in the size of the vocabulary. Next, we use a simple embedding

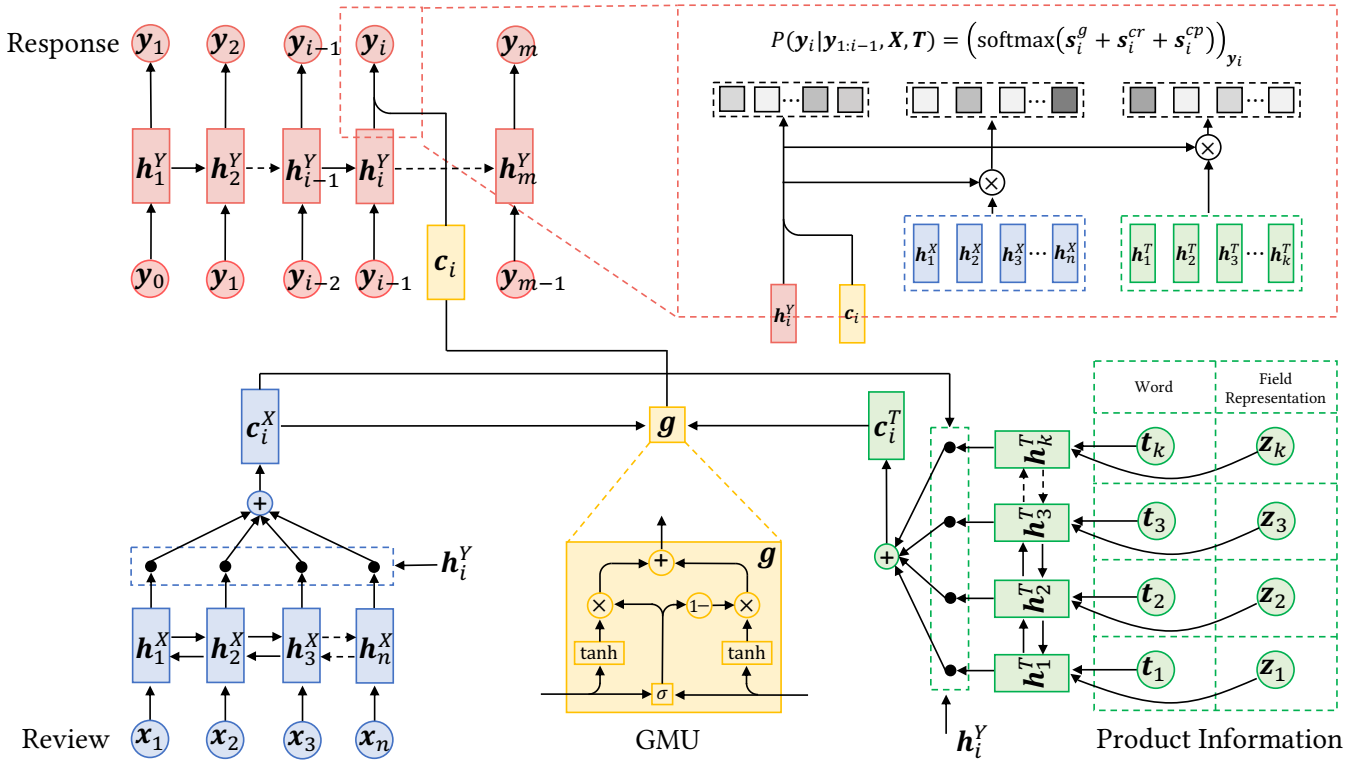


Figure 4: Architecture of our proposed model. It includes two encoder to read the review (blue) and corresponding product information (green) and a decoder to generate the response (red). A Gated Multimodal Unit (GMU) (yellow) is used to learn fusion transformations from multiple sources of information and a copy mechanism (top right corner) allows the decoder to directly copy from review or product information.

layer to encode each one-hot vector to a word vector $\boldsymbol{x}_i$. After we get the word-level review feature representation $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$, the review encoder aims to encode each word $\boldsymbol{x}_i$ in the review into the hidden state $\boldsymbol{h}_i^X$. In this work, we use two-layer bidirectional recurrent neural network (BiRNN) with gated recurrent units (GRU) [4] to read the review. The bidirectional RNN can not only capture the previous contextual information but also the future contextual information. At each time step, the bidirectional RNN unit takes the word embedding $\boldsymbol{x}_i$ as input, and outputs a hidden state $\boldsymbol{h}_i^X$. Concretely, the formulas at step i are as follows:

$$\begin{aligned} \overrightarrow{\boldsymbol{h}}_i^X, \overleftarrow{\boldsymbol{h}}_i^X &= \text{BiGRU}^X(\boldsymbol{x}_i, \overrightarrow{\boldsymbol{h}}_{i-1}^X, \overleftarrow{\boldsymbol{h}}_{i+1}^X; \boldsymbol{\theta}_X), \\ \boldsymbol{h}_i^X &= \overrightarrow{\boldsymbol{h}}_i^X \oplus \overleftarrow{\boldsymbol{h}}_i^X, \end{aligned} \tag{4}$$

where $\overrightarrow{\boldsymbol{h}}_i^X, \overleftarrow{\boldsymbol{h}}_i^X \in \mathbb{R}^{d_h}$ is the forward hidden state and backward hidden state of the BiGRU^X at step i, respectively; $\oplus$ is the concatenation operation and $\boldsymbol{h}_i^X$ is the final hidden state at step i. $\boldsymbol{\theta}_X$ represents all the parameters of the BiGRU^X.

Product Encoder

Besides the review encoder, we also need a product encoder to read the product information. As described above, the product information is represented as a sequence of words $\{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_k\}$ and their corresponding field representation $\{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_k\}$. The basic structure of the product encoder is the same as that of the review encoder, i.e., two-layer bidirectional recurrent neural network (BiRNN) with gated recurrent units (GRU). The difference is that we feed the concatenation of word embedding $\boldsymbol{t}_i$ and the corresponding field representation $\boldsymbol{z}_i$ to the bidirectional RNN. Concretely, the formulas at step i are as follows:

$$\begin{aligned} \overrightarrow{\boldsymbol{h}}_i^T, \overleftarrow{\boldsymbol{h}}_i^T &= \text{BiGRU}^T(\boldsymbol{t}_i \oplus \boldsymbol{z}_i, \overrightarrow{\boldsymbol{h}}_{i-1}^T, \overleftarrow{\boldsymbol{h}}_{i+1}^T; \boldsymbol{\theta}_T), \\ \boldsymbol{h}_i^T &= \overrightarrow{\boldsymbol{h}}_i^T \oplus \overleftarrow{\boldsymbol{h}}_i^T, \end{aligned} \tag{5}$$

where $\overrightarrow{\boldsymbol{h}}_i^T, \overleftarrow{\boldsymbol{h}}_i^T \in \mathbb{R}^{d_h}$ is the forward hidden state and backward hidden state of the BiGRU^T at step i, respectively; $\boldsymbol{h}_i^T$ is the final hidden state. $\boldsymbol{\theta}_T$ represents all the parameters of the BiGRU^T.

Response Decoder

Once the review and product information is encoded, a decoding two-layer RNN with GRU is used to generate the response. The decoding process starts as soon as the decoder receives a starting symbol "<s>" ($\boldsymbol{y}_0$ in Figure 4) and ends when the decoder generates an ending symbol "<e>". On each step i, the decoder receives the word embedding of the previous word and outputs the decoder hidden state $\boldsymbol{h}_i^Y$:

$$\boldsymbol{h}_i^Y = \text{GRU}^Y(\boldsymbol{y}_{i-1}, \boldsymbol{h}_{i-1}^Y; \boldsymbol{\theta}_Y), \tag{6}$$

where $\boldsymbol{y}_{i-1}$ is the word embedding of the previous word and $\boldsymbol{h}_i^Y$ is the hidden state of the GRU^Y at step i. $\boldsymbol{\theta}_Y$ represents all the parameters of the GRU^Y. Next, the hidden state $\boldsymbol{h}_i^Y$ is used to calculate the context vector $\boldsymbol{c}_i$ by an attention mechanism which will be detailed in the below. Finally, $\boldsymbol{h}_i^Y$ and $\boldsymbol{c}_i$ together is used to generate the word $\boldsymbol{y}_i$.

Gated Multi-Source Attention Mechanism

After the review encoder and product encoder, we get the review feature representation $\{\boldsymbol{h}_1^X, \boldsymbol{h}_2^X, \ldots, \boldsymbol{h}_n^X\}$ and the product feature representation $\{\boldsymbol{h}_1^T, \boldsymbol{h}_2^T, \ldots, \boldsymbol{h}_k^T\}$. In view of the fact that the word in response at step i is often only related to a small part of the review or product information, we utilize a gated multi-source attention mechanism to generate a high-level representation of the review part and product part. The proposed attention mechanism is introduced to make the decoder access useful review or product information in generation and it includes three parts: a review attention, a product attention, and a Gated Multimodal Unit (GMU). We also hope the generated response to cover all the issues in user's review, not just one. Therefore, we introduce the coverage vector $\boldsymbol{r}_i$ [27] in each time step, which maintains the sum of attention value over all previous steps and records what we have covered before step i. $\boldsymbol{r}_i$ is also used as an input to the review attention. Concretely, We first calculate the coverage vector $\boldsymbol{r}_i$ and then calculate the review context vector $\boldsymbol{c}_i^X$ by the review attention according to the following formulas:

$$\begin{aligned} \boldsymbol{r}_i &= \sum_{j=1}^{i-1} \boldsymbol{a}_j^X, \\ e_{ij}^X &= (\boldsymbol{v}_X)^\top \tanh(\boldsymbol{W}_X \boldsymbol{h}_j^X + \boldsymbol{W}_{XY} \boldsymbol{h}_i^Y + \boldsymbol{W}_{Xr} \boldsymbol{r}_i), \\ \boldsymbol{a}_i^X &= \mathrm{softmax}(\boldsymbol{e}_i^X), \\ \boldsymbol{c}_i^X &= \sum_{j=1}^{n} a_{ij}^X \boldsymbol{h}_j^X, \end{aligned} \tag{7}$$

where $\boldsymbol{v}_X, \boldsymbol{W}_X, \boldsymbol{W}_{XY}$ and $\boldsymbol{W}_{Xr}$ are learnable parameters and $\boldsymbol{a}_i^X$ is the attention weight at step i.

For the product attention, in addition to hidden state $\boldsymbol{h}_i^Y$, the review context vector $\boldsymbol{c}_i^X$ is also employed in the calculation of product attention, which allows the model to find relevant information in product according to the issue mentioned in review. Concretely, The calculation is made by the following equations:

$$\begin{aligned} e_{ij}^T &= (\boldsymbol{v}_T)^\top \tanh(\boldsymbol{W}_T \boldsymbol{h}_j^T + \boldsymbol{W}_{TX} \boldsymbol{c}_i^X + \boldsymbol{W}_{TY} \boldsymbol{h}_i^Y), \\ \boldsymbol{a}_i^T &= \mathrm{softmax}(\boldsymbol{e}_i^T), \\ \boldsymbol{c}_i^T &= \sum_{j=1}^{k} a_{ij}^T \boldsymbol{h}_j^T, \end{aligned} \tag{8}$$

where $\boldsymbol{v}_T, \boldsymbol{W}_T, \boldsymbol{W}_{TX}$ and $\boldsymbol{W}_{TY}$ are learnable parameters and $\boldsymbol{a}_i^T$ is the attention weight at step i.

To combine multiple sources of information, we use the Gated Multimodal Unit (GMU) [1] to learn fusion transformations and obtain the final context vector $\boldsymbol{c}_i$. The purpose of GMU is to find a feature representation based on a combination of data from different sources. The GMU learns to decide how different sources influence the activation of the unit using multiplicative gates. The equations governing the GMU are as follows:

$$\begin{aligned} \boldsymbol{g}_i^X &= \tanh(\boldsymbol{U}_X \cdot \boldsymbol{c}_i^X), \\ \boldsymbol{g}_i^T &= \tanh(\boldsymbol{U}_T \cdot \boldsymbol{c}_i^T), \\ \boldsymbol{z}_i &= \sigma(\boldsymbol{U}_z \cdot (\boldsymbol{c}_i^X \oplus \boldsymbol{c}_i^T)), \\ \boldsymbol{c}_i &= \boldsymbol{z}_i * \boldsymbol{g}_i^X + (1 - \boldsymbol{z}_i) * \boldsymbol{g}_i^T, \end{aligned} \tag{9}$$

where $\boldsymbol{U}_X, \boldsymbol{U}_T$ and $\boldsymbol{U}_z$ are learnable parameters and $\boldsymbol{z}_i$ is the gate that control the fusion transformations of $\boldsymbol{c}_i^X$ and $\boldsymbol{c}_i^T$.

Copy Mechanism

Through the above attention mechanism, we get the final context vector $\boldsymbol{c}_i$. In order to generate the next word, we need to define a score function. We use the generation score function defined by the following equation:

$$\boldsymbol{s}_i^g = \boldsymbol{W}_g(\boldsymbol{h}_i^Y \oplus \boldsymbol{c}_i) + \boldsymbol{b}_g. \tag{10}$$

where $\boldsymbol{W}_g$ and $\boldsymbol{b}_g$ are learnable parameters and $\boldsymbol{s}_i^g$ is the generation score.

To allow the decoder directly copies the words of review or product information, we also introduce two copying scores: the score of copying from review and the score of copying from product. That is,

$$\begin{aligned} s_{ij}^{cr} &= \tanh((\boldsymbol{h}_j^X)^\top \boldsymbol{W}_{cr}) \cdot (\boldsymbol{h}_i^Y \oplus \boldsymbol{c}_i), \\ \boldsymbol{s}_i^{cr} &= \sum_{j=1}^{n} s_{ij}^{cr} \cdot \mathbf{1}_{\boldsymbol{x}_i}, \end{aligned} \tag{11}$$

and

$$\begin{aligned} s_{ij}^{cp} &= \tanh((\boldsymbol{h}_j^T)^\top \boldsymbol{W}_{cp}) \cdot (\boldsymbol{h}_i^Y \oplus \boldsymbol{c}_i), \\ \boldsymbol{s}_i^{cp} &= \sum_{j=1}^{k} s_{ij}^{cp} \cdot \mathbf{1}_{\boldsymbol{t}_i}, \end{aligned} \tag{12}$$

where $\boldsymbol{W}_{cr}$ and $\boldsymbol{W}_{cp}$ are learnable parameters. $\mathbf{1}_{\boldsymbol{x}_i}$ is a one-hot vector of length V (V is the vocabulary size), among which the values are only with a single 1 in word $\boldsymbol{x}_i$ and all the others 0. The final score function is defined as

$$\boldsymbol{s}_i = \boldsymbol{s}_i^g + \boldsymbol{s}_i^{cr} + \boldsymbol{s}_i^{cp}, \tag{13}$$

and the probability of $\boldsymbol{y}_i$ is

$$P(\boldsymbol{y}_i | \boldsymbol{y}_{1:i-1}, \boldsymbol{X}, \boldsymbol{T}) = (\mathrm{softmax}(\boldsymbol{s}_i))_{\boldsymbol{y}_i}. \tag{14}$$

The maximum-likelihood loss at step t is given by:

$$\mathcal{L}_i^{ml} = -\log(P(\boldsymbol{y}_i | \boldsymbol{y}_{1:i-1}, \boldsymbol{X}, \boldsymbol{T})), \tag{15}$$

and the total loss is:

$$\mathcal{L}^{ml} = \sum_{i=1}^{m} \mathcal{L}_i^{ml}. \tag{16}$$

In training, we optimize the loss $\mathcal{L}^{ml}$ in the dataset.

3.2 Reinforcement Learning

During the training process, we feed the ground-truth output sequences to decoder and train the model by minimizing loss (16), while during inference decoder generates the next word given the previous predicted words. The errors will accumulate as decoder outputs the sequence, which is known as exposure bias problem [20]. Due to this problem, the maximum-likelihood training does not always produce the best results on evaluation metrics such as

	size	length of reviews	length of responses	number of records
Train	80000	39.96	72.67	15.22
Test	10000	39.76	72.64	15.27
Valid	10000	39.68	72.99	15.24

Table 2: Corpora statistics. We also give the average length of reviews and responses, and the average number of records of the product information.

ROUGE and BLEU and usually produces sub-optimal results with respect to these metrics.

In order to address this problem, we try to learn a policy that directly maximizes a specific evaluation metric, rather than minimizing the maximum-likelihood loss. Since the evaluation metric is non-differentiable, the traditional supervised learning methods cannot be used. Specifically, we employ self-critical sequence training (SCST) [21], a reinforcement learning algorithm, to achieve this optimization procedure. In SCST, we produce two output sequences: $\hat{Y}$, which is obtained by greedy decode, and Y^s, which is obtained by sampling from distribution at each decoding step. The reward function $r(Y)$ is defined as the specific metric, and we minimize the reinforcement loss:

$$\mathcal{L}^{rl} = (r(\hat{Y}) - r(Y^s)) \sum_{i=1}^{m} \log(P(\boldsymbol{y}_i^s | \boldsymbol{y}_{1:i-1}^s, \boldsymbol{X}, \boldsymbol{T})) \tag{17}$$

That is, if the sequence Y^s obtains a higher reward than baseline $\hat{Y}$, we will increase the probability of Y^s and decrease if it obtains a lower reward. Optimizing the specific metric encourages the model to focus more on global meanings, rather than on word-level discrepancy.

Therefore, the final loss is given by:

$$\mathcal{L} = \lambda_{rl} \cdot \mathcal{L}^{rl} + (1 - \lambda_{rl}) \cdot \mathcal{L}^{ml}, \tag{18}$$

where λ_{rl} ($0 \leq \lambda_{rl} \leq 1$) is a scaling factor that control the balance between $\mathcal{L}^{rl}$ and $\mathcal{L}^{ml}$.

4 EXPERIMENTS

4.1 The Taobao Dataset

We create the Taobao dataset from taobao.com, the largest online shopping website in China. There are 100K (review, product information, response) triples in the corpus which all belong to the category of Clothes. The product information is collected from the product page in the website and it is organized as a factual table. In the corpus, the average length of reviews is 39 words and the average length of response is 72 words. The average number of records in the table of product information is 15. The detailed corpora statistics is given in Table 2. We will release the dataset after the paper is published.

In our experiments, we use 80% of the data for training, 10% for validation and 10% for testing.

Hyper-parameter	
vocab size	15000
word embedding	128
field embedding	32
position embedding	8
hidden size	256
batch size	16
learning rate	0.02
dropout rate	0.15
decay rate	0.8
gradient clipping	5

Table 3: Hyper-parameters used in all experiments (baseline models and our model).

4.2 Evaluation Metrics

To evaluate our approach, we performed experiments on the Taobao dataset, and used the following evaluation metrics to assess the generation quality:

- **ROUGE**: ROUGE [14] is a widely used automatic evaluation metric in text summarization. The ROUGE score is obtained using the `pyrouge` package[2]. we report ROUGE- 1, ROUGE-2 and ROUGE-L in this paper.
- **METEOR**: METEOR[3] [6] has several features that are not found in other metrics, such as stemming and synonymy matching. It has proved to have good correlation with human judgement.
- **BLEU**: BLEU [18] is widely used in neural machine translation. It measures word overlap between the generated text and the ground-truth. The BLEU score is calculated using the `NLTK` package[4], in which the score is an average of BLEU-1∼4.
- **Distinct**[5]: Distinct metric [13] measures how informative and diverse the generated responses are. Distinct-i represents the ratio of distinct i-gram in responses. We report Distinct-1 and Distinct-2 in this paper.
- **Human Evaluation**: We randomly sampled 100 cases and invited five volunteers to evaluate the generated responses of different models. For each triple (review, product information,response), volunteers are asked to give a score from {0,1,2}, which represents the quality of response. 0 means the response is irrelevant or disfluent with grammatical errors; 1 represents the response is related but not informative enough; and 2 indicates the response is not only related and natural, but also informative and interesting.

[2]https://pypi.org/project/pyrouge/
[3]http://www.cs.cmu.edu/~alavie/METEOR/
[4]https://www.nltk.org
[5]Since this metric is affected by the dataset size (as the size increases, the metric will tend to zero.), we only choose the first 200 cases to compute this metric.

Model	ROUGE-1	ROUGE-2	ROUGE-L	METEOR	BLEU	Distinct-1	Distinct-2
Seq2Seq + Atte	29.93	7.05	21.98	11.95	4.22	5.73	14.49
Pointer-Generator	29.71	6.94	21.76	12.25	4.66	6.25	17.29
Copynet	30.57	7.17	22.37	12.39	4.46	4.89	12.24
Copynet + PI	33.91	11.99	26.35	14.44	10.75	8.66	25.15
Our Model	**35.06**	**13.66**	**27.72**	**15.06**	**12.30**	**8.70**	**25.90**

Table 4: Comparison with previous models using various automatic evaluation metrics. Note that the baseline "Copynet + PI" utilizes product information as well.

Model	2	1	0
Seq2Seq + Atte	68.2%	13.6%	18.2%
Pointer-Generator	68.8%	15.8%	15.4%
Copynet	69.2%	12.2%	18.6%
Copynet + PI	70.0%	16.4%	13.8%
Our Model	72.6%	15.4%	12.0%
Human Performance	82.4%	10.8%	6.8%

Table 5: Human evaluation of the five models. We show the percentage of each score. We also give the human performance in the last row.

4.3 Setups

In this work, we compare our model to the following four baseline models:

- **Seq2Seq + Atte**: "Seq2Seq + Atte" represents the standard Seq2Seq model with attention mechanism. The detailed description can be found in [4].
- **Pointer-Generator**: "Pointer-Generator" is proposed in [22], which augments "Seq2Seq + Atte" via pointing that can copy words from the source text and use coverage mechanism to eliminate repetition.
- **Copynet**: "Copynet" [9] integrates copying mechanism that is similar to the pointing in "Pointer-Generator", but the implementation is different.
- **Copynet + PI**[6]: "Copynet + PI" augments "Copynet" with the product information. The sequence of words $\{t_1, t_2, \ldots, t_k\}$ which represents the table of product information is appended to the input review.

For the four baseline models and our model, we used mini-batch SGD with momentum to train these networks. The momentum was set to 0.9 and the initial learning rate was set at 0.02. After 10 epochs, the learning rate decayed every 4000 steps and the decay rate was 0.8. We employed the dropout technique with 15% dropout rate to avoid overfitting. To cope with the exploding gradient problem, gradient clipping was performed and we clipped the norm of the gradients at 5. The word embeddings were initialized by pre-trained embedding based on word2vec and then fine-tuned during training, but the field embedding and position embedding was randomly initialized. These three embedding size was set to 128, 32, and 8, respectively. For all models, we used two-layers GRU, and the hidden state size was set to 256. We used Jieba[7] as Chinese word segmenter and used a vocabulary of 15K words which was shared by the reviews and responses. These hyper-parameters are summarized in Table 3.

Since RL training is hard to converge from random initialization, we first train our model with the standard maximum likelihood loss (λ_{rl} is set to 0). After 30 epochs, we switch to RL training with λ_{rl} set to 0.99 and learning rate set to 0.0001. We choose the BLEU metric as the reward function. Our code will be made publicly available for reproducibility.

4.4 Automatic Evaluation

We first discuss the experimental results based on automatic evaluation metrics. Table 4 lists the results of four baseline models and our model. According to the results in Table 4, "Pointer-Generator" achieves better performance than "Seq2Seq + Atte" on all metrics except ROUGE and "Copynet" outperforms "Seq2Seq + Atte" on all metrics except Distinct 1~2, demonstrating the effectiveness of incorporating copy or pointing mechanism. Compared with other three baselines, "Copynet + PI" achieves significant improvement on all metrics. For example, it improves ROUGE-L by 3.98, BLEU by 6.29 and Distinct-2 by 12.91, compared with "Copynet". The results further verify our claim that product information is crucial and necessary for generating high-quality response, and the results on Distinct metric show that the diversity is increased greatly by incorporating product information. The last row of Table 4 shows the result of our model, which outperforms all four baseline models. For example, compared with "Copynet + PI", our model further improves ROUGE-L by 1.37, BLEU by 1.55 and Distinct-2 by 0.75, achieving start-of-the-art performance.

4.5 Human Evaluation

Table 5 gives the results of human evaluation. We give the percentage of each score, which is calculated by combining all the annotations together. We also calculated the variance of five volunteers' score. The result is 0.25, which demonstrates the fair inter-human agreements. From Table 5, it is clear that the models of incorporating product information ("Copynet + PI" and our model) generate much more relevant and informative responses (labeled as "2") and

[6]Copynet is the best-performed baseline (on most metrics) of the first three models, so we employ this method for PI integration.

[7]https://github.com/fxsjy/jieba

Percent	ROUGE-1	ROUGE-2	ROUGE-L	METEOR	BLEU	Distinct-1	Distinct-2
20%	30.68	8.06	22.88	12.70	5.88	7.43	21.32
40%	31.95	9.41	24.12	13.25	7.53	7.69	21.99
60%	32.92	10.62	25.30	13.84	8.91	8.08	22.51
80%	33.62	11.68	26.16	14.31	10.25	8.51	24.58
100%	**35.06**	**13.66**	**27.72**	**15.06**	**12.30**	**8.70**	**25.90**

Table 6: The results (based on various automatic evaluation metrics) of the proposed model when using various sizes (20%, 40%, 60%, 80%, 100%) of product information.

Model	ROUGE-1	ROUGE-2	ROUGE-L	METEOR	BLEU	Distinct-1	Distinct-2
Our Model	**35.06**	**13.66**	**27.72**	**15.06**	**12.30**	**8.70**	**25.90**
− RL	34.59	13.39	27.27	14.50	12.23	8.23	24.61
− Gate	34.76	13.33	27.45	14.83	11.90	8.19	24.60
− Copy	34.82	13.44	27.64	14.93	11.95	7.92	23.47
− Field	34.72	13.35	27.40	15.02	12.19	8.30	24.80

Table 7: Ablation studies on different components. The results (based on various automatic evaluation metrics) of removing (−) some components are given.

less universal responses (labeled as "1") than other baseline models. Among them, our model achieves the best performance. 72.6% of responses generated by our model are labeled as score 2, improving "Copynet + PI" by 2.6%, and only 12.0% of responses are labeled as score 0. We also evaluated the performance of responses written by human, and the results are shown in the last row of Table 5. Human achieves 82.4% on score 2 (which outperforms our model by 9.8%), 10.8% on score 1, and only 6.8% on score 0. This results indicate that there is still a huge room for improvement and we will continue to improve our model in the future.

4.6 Analysis on Product Information

In the proposed framework, we employed the product information in the review response generation, which is an essential contribution of this study. In this section, we explore the impact of the product information in the model. To investigate the impact of these data, we performed experiments on using various sizes (20%, 40%, 60%, 80%, 100%) of product information. For instance, 20% means that we randomly sample 20% of the whole of product information. The results of the proposed model (based on various automatic evaluation metrics) when using various sizes of product information are shown in Table 6. We can find that, with an increase in the amount of product information, all evaluation metrics gradually grows as well, which indicating the usefulness and importance of the product information in review response generation.

4.7 Ablation Study

To analyze the effect of each component, we conducted ablation studies on components of the proposed model. Table 7 lists the results of the ablation studies based on various automatic evaluation metrics. The performance of full model is listed in the first row of Table 7. We also report the results of removing (−) some components, such as the RL training (−RL in Table 7), the GMU gate mechanism (−Gate), the copy mechanism (−Copy) and the field representation (−Field). The first thing we can observe is that the RL training helps improve the performance on all metrics. For example, ROUGE-L drops to 27.27 from 27.72 and METEOR drops to 14.50 from 15.06 when we removed the RL training. The GMU gate mechanism is also crucial: all evaluation metrics drops if it is removed and replaced with a concatenation operation of $\boldsymbol{c}_i^X$ and $\boldsymbol{c}_i^T$. To test the importance of copy mechanism, we performed an experiment that removed the copy mechanism and we observed a drop, indicating that directly copying from review of product information can be very useful. From the results, we found that the copy mechanism is especially useful in improving the Distinct metrics. Distinct-1 drops to 7.92 from 8.70 and Distinct-2 drops to 23.47 from 25.90 when removing the copy mechanism. Moreover, the using of field representation is necessary for improving performance, which can effectively help the model in incorporating product information.

4.8 Case Study

To compare the generated responses between different models, we give two examples on Figure 5. The case in the left of Figure 5 is a positive review and the case in the right is a negative review. Below the reviews, We show the generated responses of two best baselines ("Copynet" and "Copynet+PI") and our model. Since we used a Chinese dataset, we give the original Chinese text and the English

评论：质量不错，款式好。很显身材，也很上档次。
Review: Good quality and good style. The clothes are figure flattering and also very high grade.

Copynet: 亲，感谢您对我们的支持，您的满意是我们最大的动力，我们会继续努力，为您提供更优质的服务，期待您的再次光临！
Copynet: Dear, thank you for your support, your satisfaction is our greatest motivation, we will continue to work hard to provide you with better service. Looking forward to your next visit!

Copynet + PI: 亲，感谢您对我们的支持与信任，您的满意是我们的最高追求。后期您有任何需要我们帮助的地方，随时欢迎垂询，我们将一如既往地为您效劳！能为您服务是我们莫大的荣幸！伊芙心悦旗舰店，祝亲生活愉快！
Copynet + PI: Dear, thank you for your support and trust, your satisfaction is our highest pursuit. Please feel free to contact us if you need our help. We will continue to help you! It is our great honor to be able to serve you! EVE NY wishes you a happy life!

Our Model: 感谢您的欣赏和喜欢，每一寸面料的裁剪，都伴随着汗水的淋漓。每一件衣服的出品，我们都会倾之付出百分百的努力和用心，希望能给您带来更舒适的穿着体验，希望伊芙心悦的时尚优雅能成为您一生的知己，给予您贴心的呵护。伊芙心悦旗舰店期待您的下次光临！
Our Model: Thank you for your appreciation and love. The cut of every inch of fabric is accompanied by sweat. Every piece of clothing, we will pay 100% effort and care. We hope to bring you a more comfortable wearing experience, and hope that EVE NY's stylish elegance can become your life's confidant and give you intimate pampering. EVE NY looks forward to your next visit!

评论：描述尺码和实际尺码差太多了，害得我买了大号！退换太麻烦，我给别人穿吧！
Review: There is too much difference between the descriptive size and the actual size, which makes me buy a large size! It's too much trouble to return. I will give it to others!

Copynet: 亲，我们的尺码都是标准码的哦，您可以参考下尺码表的哦，您的满意是我们最大的动力，我们会继续努力，为您提供更优质的产品，更优秀的服务。您的满意是我们最大的追求，祝您生活愉快！
Copynet: Dear, our size is the standard size. You can refer to the size chart. Your satisfaction is our greatest motivation. We will continue to work hard to provide you with better products and better service. Your satisfaction is our greatest pursuit, I wish you a happy life!

Copynet + PI: 感谢您选购我们的产品。由于每个人体型不一样，另外款式设计的不同，每个人穿着感觉会不一样的哦。有空常来看看哦~ 祝您购物愉快! [cachecache 旗舰店]
Copynet + PI: Thank you for purchasing our products. Because each body type is different, and the style design is different, everyone will feel differently. Visit my shop when you are free ~ I wish you a happy shopping! [cachecache flagship store]

Our Model: 亲爱的顾客，非常感谢您对 cachecache 旗舰店的关注。因每款的设计理念和面料弹性不同，相对应的尺码上是可能稍有差异的呢，建议您先测量自身基本尺寸，再参考页面尺寸信息进行购买，您也可事先咨询客服，祝您购物愉快！
Our Model: Dear customer, thank you very much for your attention to the cachecache flagship store. Because each style design and fabric elasticity are different, the corresponding size may be slightly different. It is recommended that you first measure your basic size and then refer to the size information for purchase. You can also consult customer service in advance. I wish you a happy shopping!

Figure 5: Case study, comparing the two best baselines with our model. We show the responses of one positive review in the left and the responses of one negative review in the right. Red denotes some product information ("EVE NY" and "cachecache" are *brand* name.) and blue denotes the trivial and universal responses.

translation. From the results, we can see that Copynet without product information tends to generate universal responses whether the review is positive or negative, like "*your satisfaction is our greatest motivation*" and "*we will continue to work hard to provide you with better service*". These universal responses will give users a very bad user experience, especially when users give a negative review. Instead, the models with product information (Copynet + PI and our model) generate more diverse and informative responses. For different reviews and different products, they can give different responses. For example, these responses not only give an explanation, but also mention the shop name (EVE NY and cachecache flagship store), which carry more information and give users more friendly feeling.

5 RELATED WORK

Seq2Seq Framework There has been quite a bit of research on text generation based on sequence to sequence framework. Sutskever et al. [26] first proposed the Seq2Seq framework, which maps sequences to sequences. Then, the attention mechanism [3] is proposed in neural machine translation to enhance the ability of capturing long term dependency, which is also widely used in other text generation tasks and made great success. The copy mechanism (or Copynet) [9] is proposed to incorporate copying into the Seq2Seq learning framework, which enable the model to copy words from the source text. A similar phenomenon is observable in human language communication. The Pointer-Generator [22] is close to the Copynet model, with some small differences. For example, they

have an explicit switch probability to control the probability of generation or copy, while Copynet achieves the similar component through a shared softmax function. Different from Copynet and Pointer-Generator, our model achieves a copy mechanism that not only can copy text from review but also can copy from product information, which is very significant for the review response generation. Recently, some CNN-based [5, 8, 11, 28, 33] have been proposed to replace the recurrent structure in Seq2Seq and have achieved comparable performance while maintaining faster speed than RNN-based methods. For example, The Wavenet [28] and ByteNet [11] model used dilated convolutions to increase its receptive field in the encoder and decoder. In addition, the Transformer [29] model used multi-headed self-attention, which based solely on attention mechanisms, dispensing with recurrence and convolutions. We leave these as a future work.

Generation from Structured Data Generation text from structured data (e.g., a table) is also an important task in NLP. Lebret, Grangier, and Auli [12] proposed the representation of field using field name and position information. They generated text from Wikipedia infobox based a statistical n-gram model with local and global conditioning. Then, Liu et al. [15] proposed a structure-aware Seq2Seq method to model both content and structure of the table by local and global addressing, with a field-gating encoder and a dual attention mechanism. Sha et al. [25] presented an order-planning text generation model to capture the relationship between different fields and used such relationship to make the generated text more fluent and smooth. In our work, the product information is a structured data which is collected as a table. Different from them, the structured data is used as an auxiliary information and we need to model the interaction between the main text (review) and the structured text (product information). Wiseman et al. [32] found that even template based method exceed the performance of some neural models on some metrics in text generation conditioned on a small number of database records. We do not adopt template based method, since we find that it tend to generate universal responses. The work of Li et al. [7] also used product attributes in generation task. Different from us, they used attention-enhanced attribute-to-sequence model to generate product reviews, instead of review responses, for given attribute information. Our work also related to Xing et al. [34] and Wang et al. [30] which both incorporating auxiliary information in text generation. Xing et al. [34] proposed incorporating topic information into a Seq2Seq framework to generate informative and interesting responses for chatbots and Wang et al. [30] incorporated topic information into the convolutional Seq2Seq model in text summarization.

Generation with Reinforcement Learning Reinforcement learning trains an agent to maximize a reward function by interact with the external environment and has been used in many NLP tasks. Recently, many works have explored the use of reinforcement learning in text generation. When the metric that we want to optimize is not differentiable, the traditional supervised learning methods (such as optimizing the maximum-likelihood loss) cannot be used. Therefore, we can use an agent (in reinforcement learning algorithm) to perform discrete actions and obtain a reward. Ranzato et al. [20] first proposed a reinforcement learning method to train RNN-based sequence generation models, which based on the REINFORCE algorithm proposed by Williams [31]. Bahdanau et al. [2] used another reinforcement learning method, named actor-critic, to train neural networks to generate sequences. Rennie et al. [21] proposed self-critical sequence training to optimize the non-differentiable metric in image caption. Compared to previous supervised learning methods, their model leads to significant improvements. Then, Paulus, Xiong, and Socher [19] and Wang et al. [30] applied it to text summarization and demonstrate the superiority of these methods. In addition to being used in optimizing the non-differentiable metrics, reinforcement learning method can also be used in other ways. SeqGAN [35] addressed the differentiation problem in GAN by introducing the policy gradient methods when generating text, which bypasses the generator differentiation problem by directly performing gradient policy update. These works demonstrate the great potential of reinforcement learning method in NLP.

6 CONCLUSION

In this paper, we introduced the task of review response generation in e-commerce platforms, which is valuable for customer service providers, but has not yet been well-studied. To tackle this problem, we proposed a novel deep neural network model based on the Seq2Seq framework, which can incorporate external product information. A gated multi-source attention mechanism and a copy mechanism are utilized to leverage this product information. In addition, we used a reinforcement training technique to reduce the exposure bias problem. Experimental results based on both automatic evaluation metrics and human annotations demonstrated that the performance of the proposed model advances the state-of-the-art methods. A further analysis on each model component validates the effectiveness of our model and the case study demonstrates that our model can generate more informative and diverse responses than the baseline models.

A APPENDICES

A.1 More Detailed Statistics

We investigate the review data from taobao.com, the largest e-commerce website in China. Two product collections were randomly sampled, with 10K products for each. For the former one, the response rate of the negative reviews is above 80%, and the response rate for the latter one is below 20%. Statistics from these collections shows that the product selling volumes of these groups can be quite different, i.e., on average, 3545.79 items are sold per-product for high response rate group, and 3122.08 for low response rate group. In another experiment, we randomly sampled 500K products. For all the associated reviews, only 2.98% of the reviews and only 5.68% of the negative reviews have received the responses from sellers.

ACKNOWLEDGMENTS

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by China National Key R&D Program (No.2018YFC0831105), National Natural Science Foundation of China (No.61751201, 61532011), and STCSM (No.16JC1420401, 17JC1420200).

REFERENCES

[1] John Arevalo, Thamar Solorio, Manuel Montes-y Gómez, and Fabio A González. 2017. Gated Multimodal Units for Information Fusion. *arXiv preprint*

arXiv:1702.01992 (2017).

[2] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086* (2016).

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[4] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1724–1734.

[5] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2016. Language modeling with gated convolutional networks. *arXiv preprint arXiv:1612.08083* (2016).

[6] Michael Denkowski and Alon Lavie. 2014. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

[7] Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. 2017. Learning to generate product reviews from attributes. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Vol. 1. 623–632.

[8] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional Sequence to Sequence Learning. In *International Conference on Machine Learning*. 1243–1252.

[9] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1631–1640.

[10] Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1700–1709.

[11] Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. 2016. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099* (2016).

[12] Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 1203–1213.

[13] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 110–119.

[14] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out* (2004).

[15] Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. 2018. Table-to-Text Generation by Structure-Aware Seq2seq Learning. In *AAAI*. AAAI Press.

[16] Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1412–1421.

[17] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023* (2016).

[18] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 311–318.

[19] Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304* (2017).

[20] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732* (2015).

[21] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *CVPR*.

[22] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1073–1083.

[23] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models.. In *AAAI*, Vol. 16. 3776–3784.

[24] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues.. In *AAAI*. 3295–3301.

[25] Lei Sha, Lili Mou, Tianyu Liu, Pascal Poupart, Sujian Li, Baobao Chang, and Zhifang Sui. 2018. Order-Planning Neural Text Generation From Structured Data. In *AAAI*. AAAI Press.

[26] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[27] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling Coverage for Neural Machine Translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 76–85.

[28] Aäron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W Senior, and Koray Kavukcuoglu. 2016. WaveNet: A generative model for raw audio.. In *SSW*. 125.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[30] Li Wang, Junlin Yao, Yunzhe Tao, Li Zhong, Wei Liu, and Qiang Du. 2018. A Reinforced Topic-Aware Convolutional Sequence-to-Sequence Model for Abstractive Text Summarization. In *IJCAI*. ijcai.org, 4453–4460.

[31] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

[32] Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052* (2017).

[33] Felix Wu, Ni Lao, John Blitzer, Guandao Yang, and Kilian Weinberger. 2017. Fast Reading Comprehension with ConvNets. *arXiv preprint arXiv:1711.04352* (2017).

[34] Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic Aware Neural Response Generation.. In *AAAI*, Vol. 17. 3351–3357.

[35] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient.. In *AAAI*. 2852–2858.

CBHE: Corner-based Building Height Estimation for Complex Street Scene Images

Yunxiang Zhao
The University of Melbourne & NUDT
yunxiangz@student.unimelb.edu.au

Jianzhong Qi
The University of Melbourne
jianzhong.qi@unimelb.edu.au

Rui Zhang*
The University of Melbourne
rui.zhang@unimelb.edu.au

ABSTRACT

Building height estimation is important in many applications such as 3D city reconstruction, urban planning, and navigation. Recently, a new building height estimation method using street scene images and 2D maps was proposed. This method is more scalable than traditional methods that use high-resolution optical data, LiDAR data, or RADAR data which are expensive to obtain. The method needs to detect building rooflines and then compute building height via the pinhole camera model. We observe that this method has limitations in handling complex street scene images in which buildings overlap with each other and the rooflines are difficult to locate. We propose CBHE, a building height estimation algorithm considering both building corners and rooflines. CBHE first obtains building corner and roofline candidates in street scene images based on building footprints from 2D maps and the camera parameters. Then, we use a deep neural network named BuildingNet to classify and filter corner and roofline candidates. Based on the valid corners and rooflines from BuildingNet, CBHE computes building height via the pinhole camera model. Experimental results show that the proposed BuildingNet yields a higher accuracy on building corner and roofline candidate filtering compared with the state-of-the-art open set classifiers. Meanwhile, CBHE outperforms the baseline algorithm by over 10% in building height estimation accuracy.

CCS CONCEPTS

• **Information systems** → **Location based services**; • **Computing methodologies** → **Neural networks**; *Camera calibration*; *Image representations.*

KEYWORDS

Building Height Estimation; Camera Location Calibration; Open Set Classification

ACM Reference Format:
Yunxiang Zhao, Jianzhong Qi, and Rui Zhang. 2019. CBHE: Corner-based Building Height Estimation for Complex Street Scene Images. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313394

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313394



Figure 1: Building A's rooflines are hard to detect due to the overlapping with building B, while building A's corners can help figure out the true rooflines. Red arrows: building rooflines; Blue arrows: building corners.

1 INTRODUCTION

Building height plays an essential role in many applications, such as 3D city reconstruction [3, 30], urban planning [29], navigation [14, 32], and geographic knowledge bases [50]. For example, in navigation, knowing the height of buildings helps identify those standing out in a city block, which can then be used to facilitate navigation by generating instructions such as "turn left before an 18 meters high (five-story) building".

Previous studies for building height estimation are mainly based on high-resolution optical data [18, 48], synthetic aperture radar (SAR) images [7, 41], and Light Detection and Ranging (LiDAR) data [33, 44]. Such data, however, are expensive to obtain and hence the above approach is difficult to apply at a large scale, e.g., to all the buildings on earth. Moreover, such data is usually proprietary and not available to the public and research community. Recently, street scene images (or together with 2D maps) have been used for building height estimation [11, 46], which can be easily obtained at large scale (e.g., via open source mapping applications such as Google Street View [2] and OpenStreetMap [16]). Estimating building height via street scene images relies on accurate detection of building rooflines from the images, which then enables building height computation using camera projection. However, existing methods for roofline detection check the roofline segments only, which may be confused with overlapping buildings. As shown in Fig. 1, the roofline of building B may be detected as the roofline of building A because the rooflines of different buildings may be in parallel with each other, and the buildings have similar colors and positions in the street scene images.

In this paper, we present a novel algorithm named Corner-based Height Estimation (**CBHE**) to estimate building height for *complex street scene images* (with blocking problem from other buildings and trees). Our key idea to handle overlapping buildings is to detect not only the rooflines but also building corners. We obtain coordinates of building corners from building footprints in a 2D map (e.g., OpenStreetMap). We then map the corner coordinates into the

street scene image to detect building corners in the image. Corners of different buildings do not share the same coordinates, and it is easier to associate them with different buildings, as shown in Fig. 1.

CBHE works as follows. It starts with an upper-bound estimation of the height of a building, which is computed as the maximum height that can be captured by the camera. It then tries to locate a line (i.e., a roofline candidate) at this height and repeats this process by iteratively reducing the height estimation. Following a similar procedure, CBHE also locates a set of building corner candidates. Next, CBHE filters the roofline candidates with the help of the corner candidates (i.e., a roofline candidate needs to be connected to a corner of the same building to be a true roofline). When the true roofline is identified, CBHE uses the pinhole camera model to compute the building height.

In the process above, when locating the roofline and corner candidates, we fetch two sets of image segments that may contain building rooflines or corners, respectively. To filter each set of objects and identify the true roofline and corner images, we propose a deep neural network model named **BuildingNet**. The key idea of BuildingNet is as follows. Building corner has a limited number of patterns (e.g., " ", " ", " ", " "), while non-corner images may have any pattern. The same applies to the building rooflines. Thus, we model building corner (roofline) identification as an *open set* image classification problem, where each corner (roofline) pattern forms a class, while non-corner (non-roofline) images should be differentiated from them when building the classifier. To do so, BuildingNet learns embeddings of the input images, which minimize the intra-class distance and maximize the inter-class distance, and then differentiates different classes using a Support Vector Classification (SVC) model on the learned embeddings. When a new image comes, the trained SVC model will tell whether it falls into any corner (roofline) classes. If the image does not fall into any corner (roofline) classes, it is a non-corner image and can be safely discarded.

When estimating building height via the pinhole camera model, the result highly relies on the accuracy of the camera location due to GPS errors. Therefore, CBHE calibrates the camera location before roofline detection. To calibrate the camera location, CBHE detects candidate regions of all building corners in street scene images by matching buildings in street scene images with their footprints in 2D maps based on the imprecise camera location from GPS. Then, it uses BuildingNet to classify the corner candidates and remove those images classified as non-corner. From the remaining corner after the classification through BuildingNet, CBHE selects two corners with the highest score (detailed in Section 4.2) to calibrate the camera location via the pinhole camera model. We summarize our contributions as follows:

- We model building corner and roofline detection as an open set classification problem and propose a novel deep neural network named BuildingNet to solve it. BuildingNet learns embeddings of the input images, which minimize the intra-class distance and maximize the inter-class distance. Experimental results show that BuildingNet achieves higher accuracy on building corner and roofline identification compared with the state-of-the-art open set classifiers.
- We propose a corner-based building height estimation algorithm named CBHE, which uses an entropy-based algorithm to select the roofline among all candidates from BuildingNet. The entropy-based algorithm considers both building corner and roofline features and yields higher robustness for the overlapping problem in complex street scene images. Experiments on real-world datasets show that CBHE outperforms the baseline method by over 10% regarding the estimation error within two meters.
- We propose a camera location calibration method with an analytical solution when given the locations of two building corners in a 2D map, which means highly accurate result can be guaranteed with the valid building corners from BuildingNet.

We organize the rest of this paper as follows. We review related work in Section 2 and give an overview of CBHE in Section 3. The BuildingNet and entropy-based ranking algorithm are presented in Section 4, and the building height estimation method is detailed in Section 5. We report experimental results in Section 6 and conclude the paper in Section 7.

2 RELATED WORK

In this section, we review studies on camera location calibration and building heigh estimation. We also detail our baseline method [46].

2.1 Camera Location calibration

Camera location calibration aims to refine the camera location of the taken images, given rough camera position information from GPS devices or image localization [1, 25].

Existing work uses 2.5D maps (2D maps with height information) to calibrate camera locations. Arth et al. [4] present a mobile device localization method that calibrates the camera location by matching building facades in street scene images with their footprints in 2.5D maps. Armagan et al. [3] train a *convolutional neural network* (CNN) to predict the camera location based on a semantic segmentation of the building facades in input images. Their method iteratively applies CNN to compute the camera's position and orientation until it converges to a location that yields the best match between building facades and 2.5D maps. Camera location calibration using 2.5D maps can produce good results. The hurdle is the requirement of building height information for generating 2.5D maps, which may not be available for every building. Chu et al. [9] extract the position features of *building corner lines* (the vertical line of a corner) and then find the camera location and orientation by matching the extracted position features with building footprints in 2D maps. However, their method cannot handle buildings overlapping with each other or having non-uniform patterns on their facades.

2.2 Building Height Estimation

Building height estimation has been studied using geographical data such as high-resolution images, *synthetic aperture radar* (SAR) images, and *Light Detection and Ranging* (LiDAR) data.

Studies [18, 23, 31, 39, 48] based on high-resolution images (such as satellite or optical stereo images) estimate building height via methods such as elevation comparison and shadow detection, which may be impacted by lighting and weather condition when the images are taken. Similarly, methods based on height estimation is synthetic aperture radar (SAR) images [7, 37, 41] are mainly based

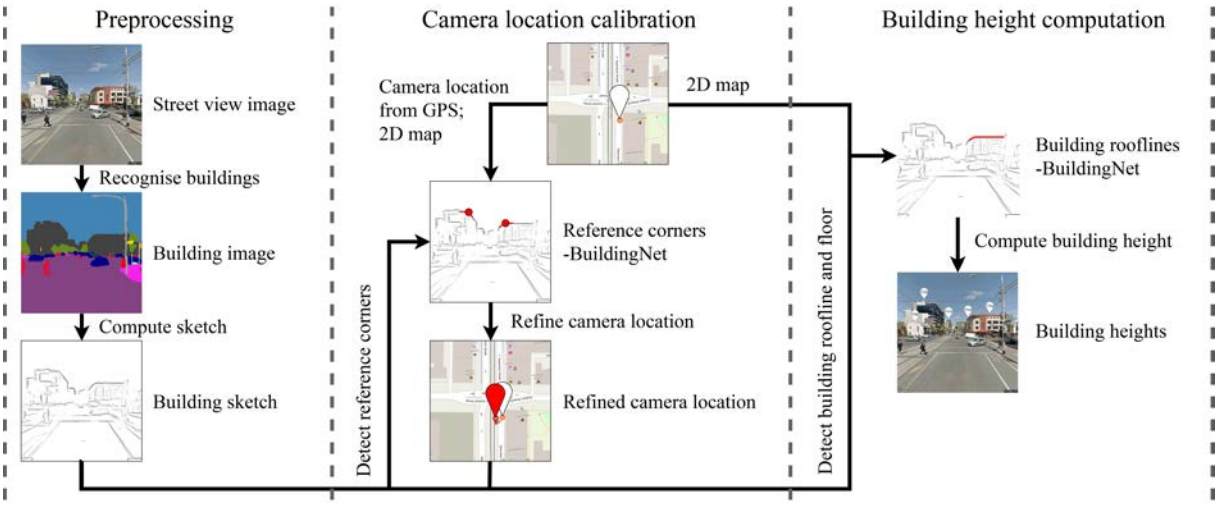


Figure 2: Solution overview

on shadow or layover analysis. Methods based on aerial images and aerial LiDAR data [33, 36] usually segment, cluster and then reconstruct building rooftop planar patches according to predefined geometric structures or shapes [48]. LiDAR data is expensive to analysis and has a limited operating altitude because the pulses are only effective between 500 and 2,000 meters [44]. A common limitation shared by the methods above is that the data that they use are expensive to collect, which significantly constraints the scalability of these methods.

Method based on street scene image and 2D map. Yuan and Cheriyadat propose a method for building height estimation uses street scene images facilitated by 2D maps [46]. Street scene images are widely available from Google Street View [2], Bing Street-Side [21] and Baidu Map [5], which makes building height estimation based on such data easier to scale. Yuan and Cheriyadat's method has four main steps: (i) Match buildings in a street scene image with their footprints in a 2D map via camera projection based on the camera location that comes with the image. Here, the camera location may be imprecise due to GPS error [15, 47]. (ii) Calibrate the camera location via camera projection with the extracted building corner lines from street scene images. (iii) Re-match buildings from a 2D map with those in the street scene image based on the calibrated camera location, and then detect building rooflines through edge detection methods. (iv) Compute building height via camera projection with camera parameters, calibrated camera location, the height of building rooflines in the street scene image, and the building footprint in the 2D map.

Our proposed CBHE differs from Yuan and Cheriyadat's method in the following two aspects: (A) In Step (ii) of their method, they calibrate camera location by matching building corner lines in the street scene image with building footprints in the 2D map. Such a method cannot handle images in urban areas where the corner lines of different buildings are too close to be differentiated, or the buildings have non-uniform patterns/colors on their facades which makes corner lines difficult to recognize. CBHE uses building corners instead of corner lines, which puts more restriction on the references for camera location calibration, and thus yields more accurate results. (B) In Step (iv) of their method, they use a local spectral histogram representation [26] as the edge indicator to capture building rooflines, which can be ineffective when buildings overlap with each other. CBHE uses the proposed deep neural network named BuildingNet to learn a latent representation of building rooflines, which has been shown to be more accurate in the experiments.

3 OVERVIEW OF CBHE

We present the overall procedure of our proposed CBHE in this section. We also briefly present the process of camera projection, which forms the theoretical foundation of building height estimation using street scene images.

3.1 Solution Overview

We assume a given street scene image of buildings that comes with geo-coordinates and angle of the camera by which the image is taken. Here, the geo-coordinates may be imprecise due to GPS errors. Google Street View images are examples of such images, and we aim to compute the height of each building in the image. As illustrated in Fig. 2, CBHE contains three stages:

- **Stage 1 – Preprocessing:** In the first stage, we pre-process the input image by recognizing the buildings and computing their sketches. There are many methods for these purposes. We use two existing models RefineNet [24] and Structured Forest [12] to identify the buildings and compute their sketches, respectively. After this step, the input image will be converted into a grayscale image with each pixel valued from 0 to 255 that contains building sketches, which enables identifying rooflines and computing the height of the building via camera projection.
- **Stage 2 – Camera location calibration:** Before computing building height by camera projection, in the second stage, we calibrate the camera location. This is necessary because a precise camera location is required in the camera projection, while the geo-coordinates that come with street scene images are imprecise due to GPS errors. To calibrate the camera location, we first detect building corner candidates in street scene images according

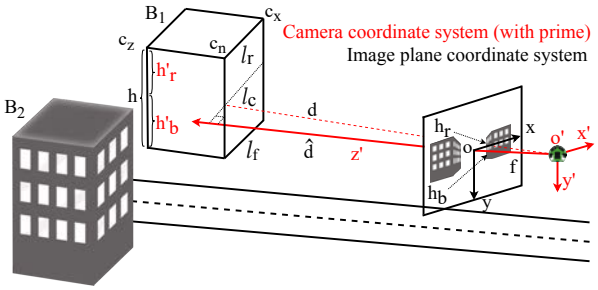


Figure 3: Geometric variables in the camera and the image coordinate systems (best view in color).

to their footprints in 2D maps and their relative position to the camera. Then, by comparing the locations and the projected positions of building corners (two corners), we calibrate the camera location via camera projection. In this stage, we propose a deep neural network named BuildingNet to determine whether an image segment contains a valid building corner. The BuildingNet model and the process of selecting two building corners for the calibration are detailed in Section 4.

- **Stage 3 – Building height computation:** In this stage, we obtain the roofline candidates of each building via weighted Hough transform and filter out those invalid roofline candidates via BuildingNet. Then we rank all valid rooflines by an entropy-based ranking algorithm considering both corner and roofline features and select the best one for computing building height via camera projection. The detailed process is provided in Section 5.

Since Stage 1 is relatively straightforward, we focus on Stages 2 and 3 in the following Sections 4 and 5, respectively. Before diving into these two stages, we briefly discuss the idea of camera projection and present the frequently used symbols.

3.2 Camera projection

We use Fig. 3 to illustrate the idea of camera projection and the corresponding symbols. In this figure, there are two coordinate systems, i.e., the camera coordinate system and the image plane coordinate system. Specifically, $\{o', x', y', z'\}$ represent the camera coordinate system, where origin o' represents the location of the camera. The camera is set horizontal to the sea level, which means that plane $x'z'$ is vertical to the building facades while the y'-axis is horizontal to the building facades. We use $\{o, x, y\}$ to represent the image plane coordinate system, where origin o is the center of the image, and plane xy is parallel to plane $x'y'$.

In Fig. 3, there are two buildings B_1 and B_2 that have been projected onto the image. For each building, we use l_r, l_f, and l_c to represent the roofline, the floor, and the line on the building projected to the x-axis (center line) of the image plane xy. Corners c_n, c_x, and c_z are the corner nearest to the camera, the corner farthest to the y-axis of the image plane when projected to the image plane (along the x-axis), and the corner closest to the y-axis of the image plane when projected to the image plane (along the z-axis), respectively. The height h of the building is the sum of the distance between l_r and l_c and the distance between l_c and l_b. These two distances are denoted as h'_r and h'_b, and the projected length of h'_r in the image plane xy is denoted by h_r. Since the camera is set horizontal to the sea level, the height of h'_b is the same as the height of the car or human beings who captured the street scene image, which can be regarded as a constant.

Table 1: Frequently used symbols

Notation	Description
h_r	the height of a building above images' center line
h_b	the height of a building below images' center line
d	the distance from the camera to c_n of a building
$\hat{d}$	the projected length of d onto the z-axis
f	the focal length of the camera
l_r	a building roofline
c_n	the corner nearest to the camera
c_x	the corner farthest to o in the image plane
c_z	the corner closest to o in the image plane

Let d be the distance from the camera o' to corner c_n, $\hat{d}$ be the projected length of d onto the z'-axis, and f be the focal length of the camera (i.e., the distance between the image center o and the camera center o'). Based on the *pinhole camera projection*, the height of a building can be computed as follows:

$$h = h'_r + h'_b = h_r \cdot \hat{d}/f + h'_b \tag{1}$$

In this equation, the focal length f comes with the input image as its metadata. The distance $\hat{d}$ is computed based on the geo-coordinates of the building and the camera, as well as the orientation of the camera. The geo-coordinates of the building are obtained from an open-sourced digital map, OpenStreetMap, while the geo-coordinates and orientation of the camera come with the input image from Google Street View. Due to GPS errors, we describe how to calibrate the location of the camera in Section 4. The height h_r is computed based on the position of the roofline l_r which is discussed in Section 5. Table 1 summarizes the symbols that are frequently used in the rest of the discussion.

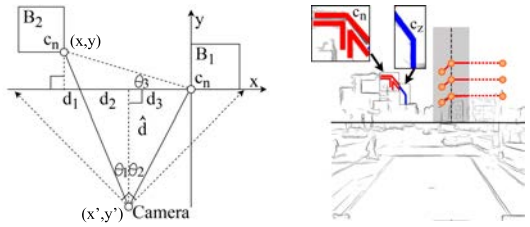
4 CAMERA LOCATION CALIBRATION

When applying camera projection for building height estimation, we need the distance $\hat{d}$ between the building and the camera. Computing this distance is based on the locations of both the building and the camera. Due to the error of GPS, we calibrate the camera location in this section.

4.1 Key Idea

We use two building corners in the street scene image with known real-world locations for camera location calibration. To illustrate the process, we project Fig. 3 to a 2D plane, as shown in Fig. 4a, and assume that corner c_n of building B_1 and building B_2 are two reference corners.

We consider a coordinate system with corner c_n of building B_1 as the origin, and the camera orientation as the y-axis. Let θ_1 and θ_2 be the angles of corner c_n of B_1 and corner c_n of B_2 from the orientation of the camera, respectively. Then the ratio of d_2/d_3 is determined by the position of these two reference corners in the image. θ_3 represents the angle between the line connecting corner c_n of B_1 and corner c_n of B_2 and x-axis, and it can be computed



(a) (b)

Figure 4: (a) geometric variables of Fig. 3 in plan view. (b) the left building shows the formation of c_n and c_z, while the right building illustrates how to find corner candidates.

according to the camera's orientation and the relative locations of the two reference corners in 2D maps. Therefore, we can compute the coordinates (x, y) of corner c_n of building B_2 in the coordinate system. With θ_1, θ_2, and the coordinate (x, y), we compute the y coordinate of the camera as follows:

$$y' = \frac{x - y \cdot tan\theta_1}{tan\theta_1 + tan\theta_2} \tag{2}$$

Since the x coordinate of the camera is equals to $y' \cdot tan\theta_2$, we obtain the relative position of the camera to the corner c_n of building B_1. Thus, camera location calibration becomes the problem of matching two building corners with their positions in the image.

The real-world location of the building corners can be obtained from 2D maps, and we need to locate their corresponding positions in the street scene image based on the (inaccurate) geo-coordinates of the camera. For a pinhole camera, matching a 3D point in the real world to a 2D point in the image is determined by a 3×4 camera projection matrix as follows:

$$\alpha \cdot p = [I|0_3] \begin{bmatrix} R & t \\ 0_3^T & 1 \end{bmatrix} \begin{bmatrix} p' \\ 1 \end{bmatrix}, \; I = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

where a real-world point $p' = (x', y', z')^T$ can be projected to its position $p = (x, y, 1)^T$ in the image plane; α is the parameter that transfers pixel scale to millimeter scale [28]; $[I|0_3]$ is the camera matrix determined by focal length f; R is the camera rotation matrix, while t is a 3-dimensional translation vector that describes the transformation from the real-world coordinates to the camera coordinates.

Since the image geo-coordinates may be inaccurate, we can only compute rough locations of the building corners. Based on the rough position of each corner, we then iteratively assume a height h_r for each building to obtain the gradient of its rooflines, as shown in Fig. 4b. We use 120×120 sub-images with the horizontal position and the assumed height of the corner as their center for building corner detection. A building corner consists of two rooflines or a roofline with a building corner line, as shown in Fig. 4b. For each building, we only consider their corners c_n and c_z. There are three types of formation for corner c_n as illustrated by the red lines on the left-hand side building in Fig. 4b, and there is one type of formation for corner c_z as illustrated by the blue lines. Based on the detected building corner candidates, we use BuildingNet described Section 4.2 to filter out non-corner image segments, and then select the two reference corners which is discussed in Section 4.2.

We assume the camera location error from Google Street View API to be less than three meters due to its camera location optimization [20]. If the camera location we compute is more than three meters away from the one provided by Google Street View API, we use the camera location from Google Street View API directly. We further improve the estimation accuracy by a multi-sampling strategy, which uses the median height among results from different images of the same building taken at different distances.

4.2 BuildingNet

We formulate building corner detection as an object classification problem, which first detects candidate corner regions for a specific building by a heuristic method, and then classifies them into different types of corners or non-corners.

We classify images that may contain building corners into five classes. The first four classes correspond to images containing one of the four types of building corners, i.e., corner c_n, c_z of the left-hand side buildings, and corner c_n, c_z of the right-hand side buildings (" ", " ", " ", " "). The last class corresponds to non-corner images which may contain any pattern except the above four types of corners (e.g., they could contain trees, lamps or power lines), and should be filtered out. Such a classification problem is an *open set* problem in the sense that the non-corner images do not have a unified pattern and will encounter unseen patterns. To solve this classification problem, we build a classifier that only requires samples of the first four classes in the training stage (can also take advantage of non-corner images), while can handle all five classes in the testing stage. To enable such a classifier, we first propose the *BuildingNet* model based on LeNet 5 [22] and triplet loss functions, which learns embeddings that map potential corner region image segments to a Euclidean space where the embeddings have small intra-class distances and large inter-class distances.

4.2.1 Triplet Relative Loss Function. As shown in Fig 5, an input of BuildingNet contains three images. Two of them (x^p and x^t) contain the same type of corner, and we name them the target (x^t) and positive (x^p), respectively. The other image x^n contains another type of corners (or a non-corner image if available), and we name it negative. BuildingNet trains its inputs to d-dimensional embeddings based on a *triplet relative loss* function inspired by Triplet-Center Loss and FaceNet [17, 35, 40, 43], which minimizes the distances within the same type of corners, and maximizes the distances between different types of corners as follows:

$$l = \sum_{i=1}^{N} \alpha \cdot ||f(x_i^t) - f(x_i^p)||_2^2 + (1 - \alpha) \cdot \frac{||f(x_i^t) - f(x_i^p)||_2^2}{||f(x_i^t) - f(x_i^n)||_2^2} \tag{4}$$

where $\alpha \in [0, 1]$ is the weight of intra-class distance in the d-dimensional Euclidean space; $(1 - \alpha)$ is the weight of the ratio between intra-classes distance and inter-class distance, which aims to separate different classes in the d-dimensional Euclidean space; N is the cardinality of all input triplets. Function f computes the d-dimensional embedding of an input image, and we normalize it to $||f(x)||_2^2 = 1$. Different from existing loss function based on triplet selection [35, 42], *triplet relative loss* function can minimise the intra-class distance and maximize the inter-class distance by means of their relative distance.

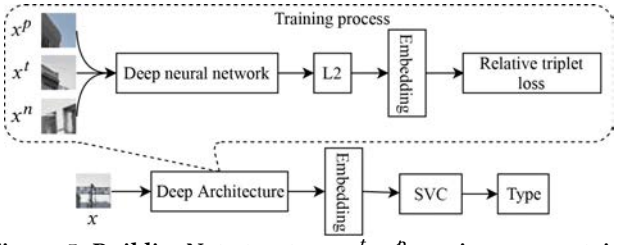


Figure 5: BuildingNet structure. x^t, x^p are images containing the same corner type. x^n is an image containing another corner type or non-corner, x is a testing image.

4.2.2 Hard Triplet Selection. Generating all possible image triplets for each batch during the training process will result in a large amount of unnecessary training data (e.g., x^t and x^p are too similar, while x^n is way different). It is crucial to select triplets that contribute more to the training phase. In BuildingNet, we accelerate convergence by assigning a higher selection probability to triplets that may contribute more to the training process. The probability of selecting a negative image x_i^n to form a training triplet is:

$$p(x_i^n) = \frac{e^{||f(x_i^n)-f(x^t)||_2^2-m}}{\sum_{i=1}^{k} e^{||f(x_i^n)-f(x^t)||_2^2-m}},\ i=[1,k]$$
$$m = min(||f(x_i^n)-f(x^t)||_2^2 - ||f(x^t)-f(x^p)||_2^2),\ i=[1,k] \tag{5}$$

Here, k is the total number of negative images in a batch. After randomly choosing x^t and x^p for a triplet, we compute the Euclidean distance between x^t and x^p, as well as the distances between x^t and the k negative images x^n in the batch. Let m be the minimum Euclidean distance between x^t and any x^n, which can be positive or negative. Then, the negative image x_i^n similar to x^t will have a higher probability to be selected.

After the training process, we obtain a d-dimensional embedding for each input image. We then learn a support vector classifier [8] based on these embeddings for corner region image classification.

4.3 Entropy-based Ranking

BuildingNet can filter out non-corner images. Among the remaining corner candidates, we select the two corners with the highest score as the reference corners. Reference corner selection relies on multiple factors: the length and edgeness (detailed in Section 5) of the lines forming the corner, the number of other corner candidates (c_n, c_x, and c_z) of the same building with the same assumed height, and the position of the corner in the image. We take the position of the corner into consideration because, empirically, corners close to one quarter or three-quarters (horizontally) of the image yield more accurate matching between their positions in the image and their footprints in 2D maps. We also consider their real-world locations because a corner close to the camera will be clearer and has higher accuracy when matching them to their footprints in 2D maps. Therefore, we define the score of each corner candidate as:

$$[s_{c_1},...,s_{c_k}]' = \begin{bmatrix} \lambda,\omega,\tau,\rho,d|c_1 \\ \vdots \\ \lambda,\omega,\tau,\rho,d|c_k \end{bmatrix} \cdot \begin{bmatrix} w_\lambda \\ \vdots \\ w_d \end{bmatrix} \tag{6}$$

where k is the number of corner candidates from all buildings; c_i is the ith corner candidate; s_{c_i} is the score of the ith corner candidate; λ is the detected length of the two lines that form a corner, while ω is the sum of the edgeness of the two lines; τ is the number of other corner candidates of the same building with the same assumed height; ρ is the minimum distance from the corner to a quarter or three-quarters of the image, and d is the distance from the corner to the camera; w_λ, w_ω, w_τ, w_ρ, w_d are the weight of these parameters. Parameters λ, ω, τ and ρ correlate with the score positively, while parameter d correlates with the score negatively.

We use an entropy-based ranking method to compute the weights of parameters $(w_\lambda, ..., w_d)^T$. Shannon entropy is a commonly used measurement of uncertainty in information theory [38]. The main idea of the entropy-based ranking algorithm is to compute the objective weights of different parameters according to their data distribution. If the samples of a parameter vary greatly, the parameter should be considered as a more important feature and thus should be given a larger weight.

For building corner classification, there are $n = 5$ parameters and $m = k$ samples. We denote the decision matrix as r, where r_{ij} is the value of the ith sample under the jth parameter. Before applying the entropy-based ranking algorithm, we pre-process these parameters by Min-max scaling as follows:

$$r_{ij} = \begin{cases} (r_{ij} - \min_j(r_{ij}))/(\max_j(r_{ij}) - \min_j(r_{ij})), & \text{iff positive} \\ (r_{ij} - \min_j(r_{ij}))/(\max_j(r_{ij}) - \min_j(r_{ij})) + 1, & \text{iff negative} \end{cases} \tag{7}$$

where positive and negative mean that the jth parameter is positively/negatively correlated with the value of r. After Min-max scaling, the entropy of each parameter based on the normalized decision matrix r' is defined as:

$$e_j = -ln(m)^{-1} \cdot \sum_{j=1}^{m} r'_{ij} \cdot ln(r'_{ij}),\ r'_{ij} = r_{ij} / \sum_{j=1}^{m} r_{ij} \tag{8}$$

where r' is the standardized r. Based on the entropy of each parameter, the weight of each parameter is computed by:

$$w_j = (1 - e_j)/(n - \sum_{j=1}^{m} e_j),\ j=[1,n] \tag{9}$$

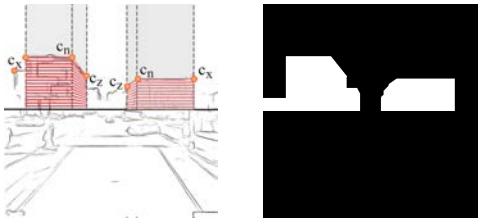
After computing the weight of each parameter, we apply them to all corner candidates and rank all the candidates by their scores to obtain the best two as the reference corners.

5 ROOFLINE DETECTION

Building height estimation requires detecting the roofline of each building. In this section, we present our method for roofline candidate detection in Section 5.1, and our method for the true roofline selection in Section 5.2. We further present a strategy for handling tall building (over 100 meters) in Section 5.3.

5.1 Roofline Candidate Detection

We consider the rooflines from corner c_n to the corner next to c_n, along the positive direction of the x'-axis in the camera coordinate system, and the one from corner c_z to the corner next to corner c_z



(a) (b)

Figure 6: (a) the heuristic method for roofline candidate detection. (b) the mask of detected buildings.

along the negative direction of the z'-axis in the camera coordinate system. The corner between corner c_z and corner c_x is corner c_n if they are adjacent to each other, as shown in Fig. 3, and we take this situation to simplify the explanation.

Similar to corner candidate detection, as shown in Fig. 6a, we find all roofline candidates of each building by a heuristic method, which projects the rooflines of each building according to its relative location to the camera in the real world, together with the camera's parameters. To do so, we first assume h_r of a building to be the maximum height that can be captured, which means that at least a roof corner (c_n, c_x, and c_z) is visible in the image. If corner c_n is visible, the maximum height computed via camera projection is:

$$h_r = \hat{d} \cdot (h_I / 2f) \tag{10}$$

where h_I is the height of the street scene image; $\hat{d}$ is the distance from corner c_n to the camera projected to the z'-axis of the camera coordinate system. If corner c_n is invisible, we use c_z as the reference corner when computing the maximum height of a building in the same way. With the maximum height of the building, we compute the position of corner c_n, c_x, and c_z in the image. We then apply Hough transform to the input edge map in Fig. 2 to detect roofline candidates, and the roofline candidates from c_n to c_x need to match the computed position of c_n and c_x. Similarly, the roofline candidates from c_n to c_z need to match the computed position of c_n and c_z. Instead of using the typical Hough transform for line detection, which takes binarized images as the input, we sum the value of all pixels valued from 0 to 255 within a line as its weight, and name the summed value as the **edgeness** of a roofline candidate, which reflects the visibility of a line in the edge map.

We iteratively reduce the assumed height with a step length of 0.5 meters until $h_r = 0$ and collect all candidate rooflines. Similar to reference corner detection, we formulate the true building roofline detection as an open set classification and ranking problem.

5.2 Roofline Classification and Ranking

There are three types of rooflines: (i) Roofline from c_n to c_x; (ii) Roofline from c_n to c_z of the left hand side buildings; (iii) Roofline from c_n to c_z of the right hand side buildings, as shown in Fig. 4c. We use BuildingNet to filter these candidates and find the true roofline, which is similar to the corner candidate validation process in Section 4.2. Based on the valid roofline candidates from BuildingNet, we weight each roofline candidate l_r by its detected length λ, edgeness ω, and the number of corners τ with the same assumed

Algorithm 1: Roofline pre-processing

Inputs: buildings B, tree area T, edge map E;
Output: updated buildings B;
M = null;
forall *building b in B* **do**
  // buildings are ordered by whether they have a detectable corner, and then their distance to the camera;
  forall *roofline* l_r *in* $b.L_r$ **do**
    // traverse all roofline candidates of building b;
    forall *pixel* $p \in l_r$ **do**
      if $M(p)$ **then**
        l_r.delete(p);
    $l_{r(ini)} = l_r$;
    forall *pixel* p *in* l_r *and* $p \notin l_r$ *and* $!M(p)$ **do**
      if $connected(p, l_r)$ *and* $T(p)$ **then**
        l_r.add(p);
    // update the length and edgeness of l_r;
    $\lambda_{l_r} = len(l_r)$;
    $\omega_{l_r} = E(l_{r(ini)}) * len(l_r)/len(l_{r(ini)})$;

height of the same building. We rank all roofline candidates via the entropy-based ranking algorithm in Section 4.3, as follows:

$$[s_{l_{r_1}}, ..., s_{l_{r_k}}]' = \begin{bmatrix} \lambda, \omega, \tau | l_{r_1} \\ \vdots \\ \lambda, \omega, \tau | l_{r_k} \end{bmatrix} \cdot \begin{bmatrix} w_\lambda | l_r \\ \vdots \\ w_\tau | l_r \end{bmatrix} \tag{11}$$

where k is the number of roofline candidates for a specific roofline; l_{r_i} is the ith roofline candidate; $s_{l_{r_1}}$ is the score of the ith roofline candidate. w_λ, w_ω and w_τ are the weight of these parameters based on all candidates of a specific roofline of a building, and all three parameters are positively correlated with the score s. The value of τ depends on the number of corners (c_n, c_x, and c_z) with the same assumed height as the roofline candidate, and its value is {0, 1, 2, 3}. We discussed how to detect references corners in Section 4.2, and the difference in detecting the corners of a specific building is that we do not consider ρ and d in Equation 6 and all corner candidates here are those of a specific building corner.

Different from building corners, which can only be visible or invisible, rooflines can also be partially blocked by other objects (trees in particular). Therefore, before we apply the ranking algorithm, we pre-process the length λ and edgeness ω which are affected by the blocking via Algorithm 1 as follows:

When estimating building height, we first separate buildings into two classes: (i) with at least one valid corner; (ii) without any valid corner. Then, we process buildings in class (i) according to their distance to the camera. After all the buildings in class (i) have been processed, we process the buildings in class (ii) according to their distance to the camera. After we obtain the height of a building, we mark the scope of the building in the street scene image, as shown in Fig. 6b (i.e., height has been obtained).

After detecting the roofline candidates of a building, we refine the λ_{l_r} of each roofline candidate using the following equation:



(a) (b)

Figure 7: (a) an image with an upward-looking view. (b) the corresponding image with a horizontal view.

$$\lambda_{l_r} = \lambda_{l_r(ini)} - \sum_{p \in l_r} M(p) + \sum_{p\ in\ l_r} T(p) \tag{12}$$

where $\lambda_{l_r(ini)}$ is the detected length of a roofline candidate l_r. $M(p)$ checks whether a pixel p within a roofline belongs to a building's scope in the street scene image that has been processed and closer to the camera, or within another building's roofline that has been processed but farther to the camera. We remove pixel p from a roofline if $M(p)$ is true. $T(p)$ checks whether a pixel p, which is in the extended line of l_r but within the projected scope of the roofline, has been blocked by trees. If there do exist these pixels and they connect to the detected roofline segment, we add them to the roofline. Accordingly, we update the edgeness of a roofline as:

$$\omega_{l_r} = (1 + \lambda_{l_r(ini)}/\lambda_{l_r}) \cdot \sum_{p \in l_r} E(p) \cdot (1 - M(p)) \tag{13}$$

where E is the input edge map of the original image, $\lambda_{l_r(ini)}$ and λ_{l_r} are the initial and prolonged length of the roofline, respectively.

5.3 Tall Building Preprocessing

Height estimation for tall buildings (over 100 meters) requires the camera to be placed far away from the buildings with an upward-looking view to capture the building roof. For images with an upward-looking view, all building corner lines will become slanted. Typically, we take the upward-looking view as 25 degrees as an example to show the strategy that we use for handling tall buildings.

We first compute a plane-to-plane homography [51], which maps an image with an upward-looking view to the corresponding image with a horizontal view. Here, we use the homogeneous estimation method [10], which solves a 3×3 homogeneous matrix h that matches a point in an upward-looking image (Fig. 7a) to a horizontal-view image (Fig. 7b) using the Equation 14:

$$A \cdot h = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -y_1Y_1 & X1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1X_1 & -y_1Y_1 & X1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nX_n & -y_nY_n & Xn \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_nX_n & -y_nY_n & Xn \end{bmatrix} \cdot h = 0 \tag{14}$$

where the homogeneous matrix $h(|h| = 1)$ is represented in the vector form as $h = (h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}, h_{33})^T$; n is the number of point pairs, which should be no less than four to validate the homogeneous equation; (X_i, Y_i) represents a point in the upward-looking image and (x_i, y_i) represents the corresponding point in the resultant image with a horizontal view. Vector h minimizes the algebraic residuals, and $A \cdot h$ is a standard result of linear algebra. Subject to $h = 1$, the least eigenvalue of $A \cdot A^T$ is given by the eigenvector, and this eigenvector can be obtained from the singular value decomposition (SVD) of A.

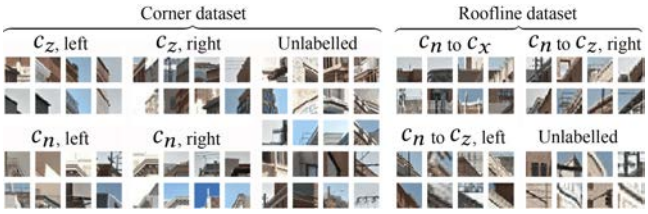


Figure 8: Examples of four types of corners, three types of rooflines, and the corresponding unlabelled images.

6 EXPERIMENTS

In this section, we first evaluate our proposed BuildingNet model for building corner and roofline classification and then evaluate our proposed CBHE algorithm for building height estimation.

6.1 Datasets

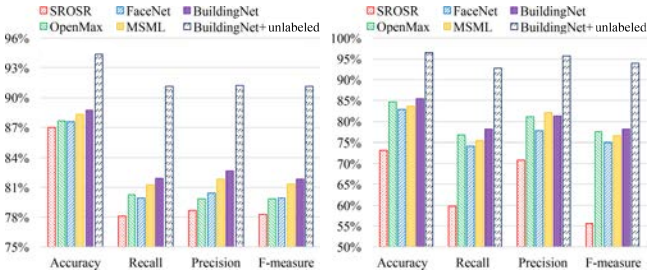
In our experiments, we obtain building footprints (geo-coordinates) from OpenStreetMap and building images from Google Street View, respectively. For the experiments on building height estimation, we use two datasets:

(i) **City Blocks**, which contains 128 buildings in San Francisco. We collect all Google Street View images (640×640 pixels) with camera orientation along the street. We set the view of the camera is 90 degrees, and the focal length can be derived via the camera parameters provided by Google. We do not need to consider the camera rotation matrix R and the translation vector t due to the image preprocessing of Google Street View. We obtain the building height ground truth from high-resolution aerial maps (e.g., NearMap [13]).

(ii) **Tall Buildings**, which contains 37 buildings taller than 100 meters in San Francisco, Melbourne, and Sydney collected by us via Google Street View API. We set the camera with an upward-looking view (25 degrees) to capture their rooflines. The building hight ground truth comes from Wikipedia pages of these buildings or derived from NearMap [13].

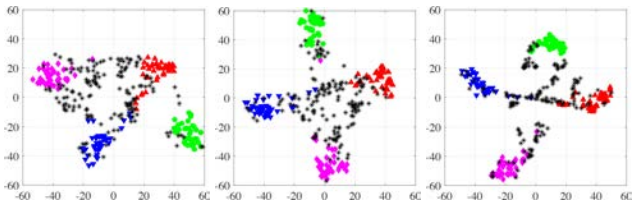
For building corner classification, we crop images from City Blocks dataset. We generate the corner dataset semi-automatically, where we crop 28×28 pixels image segments from street scene images, and then manually label whether an image segment contains a building corner (and the type of corner). The training dataset that we collected contains 10,400 images, including 1,300 images of each type of building corner (i.e., a total of 5,200 building corner images) and 5,200 non-corner images. The testing dataset contains 1,280 images, including 160 images for each type of building corners and 640 non-corner images. The training data and testing data come from different city blocks.

Following a similar approach, we collect a roofline dataset. For each roofline candidate, we extend the upper and lower 10 pixels of the roofline to obtain a $21 \times W$ image segments, where W is the length of the roofline, and we further resize (rotate if the roofline is not a horizontal line) the image to 28×28 to generate same-size inputs for BuildingNet. The training dataset includes 7,800 images, including 1,300 images for each type of rooflines (i.e., a total of



(a) (b)

Figure 9: Effectiveness of BuildingNet on (a) corner classification and (b) roofline classification (best view in color).



(a) (b) (c)

Figure 10: t-SNE [27] 2D embeddings of four types of corners and the unlabeled data after 100 epochs, learned by the loss functions in (a) FaceNet, (b) MSML, and (c) the proposed triplet relative loss (best view in color).

3,900 building roofline images) and 3,900 non-roofline images. The testing dataset contains 960 images, including 160 images for each kind of roofline and 480 non-roofline images.

6.2 Effectiveness of BuildingNet

Building corner and roofline classification is an open set classification problem where the invalid corner or roofline candidates do not have consistent features. To test the effectiveness of BuildingNet, we use two open set classifiers as the baselines: *SROSR* [49] and *OpenMax* [6]. SROSR uses the reconstruction errors for classification. It simplifies the open set classification problem into testing and analyzing a set of hypothesis based on the matched and no-matched error distributions. OpenMax handles the open set classification problem by estimating the probability of whether an input comes from unknown classes based on the last fully connected layer of a neural network. Further, we use two loss functions based on triplet selection to illustrate the effectiveness of our proposed triplet relative loss function. The loss function in *FaceNet* [35] makes the intra-class distance smaller than inter-class distance by adding a margin, and the one in *MSML* [45] optimizes the triplet selection process towards selecting hard triplets in each in training.

Hyperparameters. For the OpenMax, we use the LeNet 5 model to train on the building corner and roofline dataset for 10K iterations with the default setting in Caffe [19]. We then apply the last fully connected layer to OpenMax for classification. For our BuildingNet, we pre-train LeNet 5 with the MNIST dataset and fine-tune it with our collected building corner and roofline images. Further, since BuildingNet can also take advantage of unlabeled data (known unknown [34]) during training, we also pre-train a LeNet 5 model based on MNIST dataset (0 to 4 as the labeled data and 5 to 9 as the unlabeled data) and fine-tune it with our data. We set the learning rate as 0.1 with the decay rate of 0.95 after each 1K iterations (50K iterations in total). The batch size is 30 images for each class, the embeddings that BuildingNet learns are 128-dimensional, and the α in the triplet relative loss function is 0.5. We perform 10-fold cross-validation on the models tested, and then compute the accuracy, precision, recall, and F_1 score of different models, which are summarized in Figure 9.

On the corner dataset, BuildingNet achieves an accuracy of 94.34%, and its recall, precision, and F_1 score are all over 91% when using both labeled and unlabeled data for training. Compared with SROSR and OpenMax, BuildingNet improves the accuracy and F_1 score by more than 6% and 10%, respectively. When trained with labeled data only, BuildingNet still has the highest accuracy and F_1 score (i.e., 88.72% and 81.8%), which are 1.1% and 2% higher than OpenMax, respectively. Compared with the two loss functions in MSML and FaceNet which are also based on triplet selection, our proposed loss function can improve the accuracy and F_1 score by more than 0.4% and 0.5%, respectively. For the roofline dataset, the proposed BuildingNet again outperforms the baseline models consistently. These confirm the effectiveness of BuildingNet.

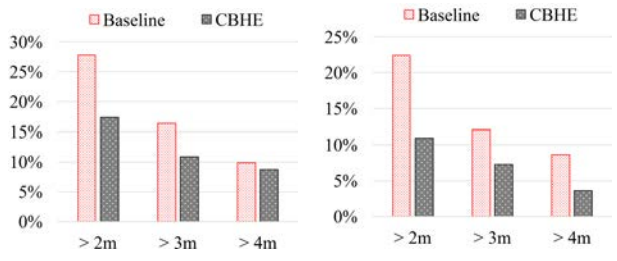
To further illustrate the effectiveness of BuildingNet, we visualize the embeddings generated by three triplet based loss functions on the corner dataset, as shown in Fig. 10. Compared with random triplet selection with margin (FaceNet) and hard triplet selection with margin (MSML), our triplet relative loss function obtains better classification result with smaller average intra-class distance and larger average inter-class distance after the same number of epochs.

6.3 Effectiveness of CBHE

We evaluate the performance of CBHE on City Blocks and Tall Buildings in this subsection.

6.3.1 Building height estimation on City Blocks. Figure 11 shows the building high estimation errors of the baseline method [46] and CBHE over the City Blocks dataset. It shows the percentage of buildings where the height estimation is greater than 2, 3, and 4 meters, respectively. In both city blocks, CBHE achieves a smaller percentage of buildings than that of the baseline [46].

In particular, in the first city block (Fig. 11a, which has been used in [46]), CBHE has 10.4%, 5.5%, and 1.2% fewer buildings than those of the baseline with height estimation errors greater than 2, 3, and 4 meters, respectively. Note that the results of the baseline method are obtained from their paper [46] since we are unable to obtain



(a) City block 1 (b) City block 2

Figure 11: The errors of the baseline method [46] and CBHE on City Blocks.



(a) Images used by the baseline (b) Images used by CBHE

Figure 12: City block street scene images at the same spots.

their source code. Also, even though CBHE is run on the same city block as the baseline in this set of experiments, the images that we used are more challenging to handle as the trees in the street scenes have grown larger which block the buildings (cf. Fig. 12).

Fig 11b shows the result in a second city block (which was not used in [46]). As we are unable to obtain the source code of the baseline method, the result is based on our implementation of their method. CBHE again outperforms the baseline. It has 11.5%, 4.8%, and 5% fewer buildings than those of the baseline with height estimation errors greater than 2, 3, and 4 meters, respectively.

6.3.2 Building height estimation on Tall Buildings. For tall buildings, the camera needs to be placed far away with an upward-look view to capture the building roofline. We capture the building images 250 meters away from the buildings via Google Street View API. Fig. 13 presents examples of the street scene images for tall building height estimation. For each street scene image, we first rotate it to the horizontal view according to Equation 14, and then compute the height of the buildings according to Section 5.



Figure 13: Tall building examples (best view in color).

The baseline method [46] cannot be applied to tall buildings, and here we only show the result of CBHE. As shown in Table 2, more than 53% of the tall buildings have a height estimation error of less than five meters and 73.33% of the tall buildings have an error of less than 10 meters.

Table 2: CBHE for tall building height estimation.

Absolute error	Percentage	Relative Error	Percentage
>5m	45.9%	>5%	40.5%
>10m	27.0%	>10%	13.5%

The errors of tall buildings may seem larger due to the camera projection (i.e., the errors are multiplied by a larger multiplier for tall buildings). However, we would like to emphasize that the relative errors are still quite low, e.g., since the tall buildings are taller than 100 meters, even a 10-meter error is less than 10% and is barely notable in reality.



(a) (b) (c) (d)

Figure 14: Challenging examples (best view in color).

6.4 Error Analysis

We summarize the challenging cases for CBHE in this section. These challenging scenarios will be explored in future work.

For those buildings whose rooflines are entirely blocked by other objects such as trees, CBHE will ignore them, or output a wrong estimation. Take Fig. 14a as an example, the trees on the left-hand side of the image block the roof of the green colored building heavily, resulting in a line below the roof to be identified as the roofline. Additionally, if the corners of a building are not detectable, lines from other buildings behind this building may also impact the result. As illustrated in Fig. 14b, the roofline of a building behind the blue colored building was detected as its roofline.

In dense city areas, the buildings may overlap with each other, and it is difficult to match all buildings with their boundaries in a 2D map accurately. Take Fig. 14c as an example, building ② is blocked by building ① and building ②'s corners have a similar horizontal position to building ⑤. Therefore, CBHE regards the rooflines of building ⑤ as the rooflines of building ②, which results in the estimated height of building ② being 77.41m, although its real height is 24.53m. Moreover, the height of building ⑤ is also wrong because the incorrect rooflines of building ② block the rooflines of building ⑤. In Fig. 14d, the blue shaded building mask on the right-hand side is wrongly assigned to the building ② (between building ① and building ③) because it is closer to the camera with the similar position to building ③.

7 CONCLUSIONS

We proposed a corner-based algorithm named CBHE to learn building height from complex street scene images. CBHE consists of camera location calibration and building roofline detection as its two main steps. To calibrate camera location, CBHE performs camera projection by matching two building corners in street scene images with their physical locations obtained from a 2D map. To identify building rooflines, CBHE first detects roofline candidates according to the building footprints in 2D maps and the calibrated camera location. Then, it uses a deep neural network named BuildingNet that we proposed to check whether a roofline candidate indeed is a building roofline. Finally, CBHE ranks the valid rooflines based on an entropy-based ranking algorithm, which also involves building corner information as an essential indicator, and then computes the building height through camera projection. Experimental results show that the proposed BuildingNet model outperforms two state-of-the-art classifiers SROSR and OpenMax consistently, and CBHE outperforms the baseline algorithm by over 10% in building height estimation accuracy.

8 ACKNOWLEDGMENTS

We thank the anonymous reviewers for their feedback. We appreciate the valuable discussion with Bayu Distiawan Trsedya, Weihao Chen, and Jungmin Son. Yunxiang Zhao is supported by the Chinese Scholarship Council (CSC). This work is supported by Australian Research Council (ARC) Discovery Project DP180102050, Google Faculty Research Award, and the National Science Foundation of China (Project No. 61402155).

REFERENCES

[1] Pratik Agarwal, Wolfram Burgard, and Luciano Spinello. 2015. Metric Localization using Google Street View. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 3111–3118.

[2] Dragomir Anguelov, Carole Dulong, Daniel Filip, Christian Frueh, Stéphane Lafon, Richard Lyon, Abhijit Ogale, Luc Vincent, and Josh Weaver. 2010. Google Street View: Capturing the World at Street Level. *Computer* 43, 6 (2010), 32–38.

[3] Anil Armagan, Martin Hirzer, Peter M. Roth, and Vincent Lepetit. 2017. Learning to Align Semantic Segmentation and 2.5D Maps for Geolocalization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4590–4597.

[4] Clemens Arth, Christian Pirchheim, Jonathan Ventura, Dieter Schmalstieg, and Vincent Lepetit. 2015. Instant Outdoor Localization and SLAM Initialization from 2.5D Maps. *IEEE Transactions on Visualization and Computer Graphics* 21, 11 (2015), 1309–1318.

[5] Baidu. 2018. Baidu Map. Retrieved Oct 18, 2018 from https://map.baidu.com/#

[6] Abhijit Bendale and Terrance E. Boult. 2016. Towards Open Set Deep Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1563–1572.

[7] Dominik Brunner, Guido Lemoine, Lorenzo Bruzzone, and Harm Greidanus. 2010. Building Height Retrieval from VHR SAR Imagery based on an Iterative Simulation and Matching Technique. *IEEE Transactions on Geoscience and Remote Sensing* 48, 3 (2010), 1487–1504.

[8] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology (TIST)* 2, 3 (2011), 27.

[9] Hang Chu, Andrew Gallagher, and Tsuhan Chen. 2014. GPS Refinement and Camera Orientation Estimation from a Single Image and a 2D Map. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 171–178.

[10] Antonio Criminisi. 1997. Computing the Plane to Plane Homography.

[11] Elkin Díaz and Henry Arguello. 2016. An Algorithm to Estimate Building Heights from Google Street-view Imagery using Single View Metrology across a Representational State Transfer System. In *Dimensional Optical Metrology and Inspection for Practical Applications V*, Vol. 9868. 98680A.

[12] Piotr Dollár and C. Lawrence Zitnick. 2013. Structured Forests for Fast Edge Detection. In *IEEE International Conference on Computer Vision (ICCV)*. 1841–1848.

[13] Google. 2018. NearMap. Retrieved Nov 4, 2018 from http://maps.au.nearmap.com/

[14] Floraine Grabler, Maneesh Agrawala, Robert W. Sumner, and Mark Pauly. 2008. Automatic Generation of Tourist Maps. *ACM Transactions on Graphics (TOG)* 27, 3 (2008), 100:1–100:11.

[15] Andreas Grammenos, Cecilia Mascolo, and Jon Crowcroft. 2018. You Are Sensing, but Are You Biased?: A User Unaided Sensor Calibration Approach for Mobile Sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (2018), 11.

[16] Mordechai Haklay and Patrick Weber. 2008. OpenStreetMap: User-Generated Street Maps. *IEEE Pervasive Computing* 7, 4 (2008), 12–18.

[17] Xinwei He, Yang Zhou, Zhichao Zhou, Song Bai, and Xiang Bai. 2018. Triplet-Center Loss for Multi-View 3D Object Retrieval. *arXiv preprint arXiv:1803.06189* (2018).

[18] Mohammad Izadi and Parvaneh Saeedi. 2012. Three-Dimensional Polygonal Building Model Estimation from Single Satellite Images. *IEEE Transactions on Geoscience and Remote Sensing* 50, 6 (2012), 2254–2272.

[19] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*. 675–678.

[20] Bryan Klingner, David Martin, and James Roseborough. 2013. Street View Motion-from-Structure-from-Motion. In *IEEE International Conference on Computer Vision (ICCV)*. 953–960.

[21] Johannes Kopf, Billy Chen, Richard Szeliski, and Michael Cohen. 2010. Street Slide: Browsing Street Level Imagery. In *ACM Transactions on Graphics (TOG)*, Vol. 29. 96.

[22] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-Based Learning Applied to Document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[23] Gregoris Liasis and Stavros Stavrou. 2016. Satellite Images Analysis for Shadow Detection and Building Height Estimation. *ISPRS Journal of Photogrammetry and Remote Sensing* 119 (2016), 437–450.

[24] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian D. Reid. 2017. RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5168–5177.

[25] Liu Liu, Hongdong Li, and Yuchao Dai. 2017. Efficient Global 2D-3D Matching for Camera Localization in a Large-Scale 3D Map. In *IEEE International Conference on Computer Vision (ICCV)*. 2391–2400.

[26] Xiuwen Liu and DeLiang Wang. 2002. A Spectral Histogram Model for Texton Modeling and Texture Discrimination. *Vision Research* 42, 23 (2002), 2617–2634.

[27] Laurens Van Der Maaten and Geoffrey Hinton. 2008. Visualizing Data Using T-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.

[28] Trish Meyer and Chris Meyer. 2010. *Creating Motion Graphics with After Effects*. Taylor & Francis.

[29] Edward Ng. 2009. Policies and Technical Guidelines for Urban Planning of High-Density Cities–Air Ventilation Assessment (AVA) of Hong Kong. *Building and Environment* 44, 7 (2009), 1478–1488.

[30] Jiyan Pan, Martial Hebert, and Takeo Kanade. 2015. Inferring 3D Layout of Building Facades from a Single Image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2918–2926.

[31] Feng Qi, John Z Zhai, and Gaihong Dang. 2016. Building Height Estimation using Google Earth. *Energy and Buildings* 118 (2016), 123–132.

[32] Adam Rousell and Alexander Zipf. 2017. Towards a Landmark-Based Pedestrian Navigation Service using OSM Data. *ISPRS International Journal of Geo-Information* 6, 3 (2017), 64.

[33] Aparajithan Sampath and Jie Shan. 2010. Segmentation and Reconstruction of Polyhedral Building Roofs from Aerial Lidar Point Clouds. *IEEE Transactions on Geoscience and Remote Sensing* 48, 3 (2010), 1554–1567.

[34] Walter J Scheirer, Lalit P Jain, and Terrance E Boult. 2014. Probability Models for Open Set Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 36, 11 (2014), 2317–2324.

[35] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A Unified Embedding for Face Recognition and Clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 815–823.

[36] Gunho Sohn, Xianfeng Huang, and Vincent Tao. 2008. Using a Binary Space Partitioning Tree for Reconstructing Polyhedral Building Models from Airborne Lidar Data. *Photogrammetric Engineering & Remote Sensing* 74, 11 (2008), 1425–1438.

[37] Hélène Sportouche, Florence Tupin, and Léonard Denise. 2011. Extraction and Three-Dimensional Reconstruction of Isolated Buildings in Rrban Scenes from High-Resolution Optical and SAR Spaceborne Images. *IEEE Transactions on Geoscience and Remote Sensing* 49, 10 (2011), 3932–3946.

[38] Li-yan Sun, Cheng-lin Miao, and Li Yang. 2017. Ecological-Economic Efficiency Evaluation of Green Technology Innovation in Strategic Emerging Industries based on Entropy Weighted TOPSIS Method. *Ecological Indicators* 73 (2017), 554–558.

[39] Frederik Tack, Gurcan Buyuksalih, and Rudi Goossens. 2012. 3D Building Reconstruction based on Given Ground Plan Information and Surface Models Extracted from Spaceborne Imagery. *ISPRS Journal of Photogrammetry and Remote Sensing* 67 (2012), 52–64.

[40] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2018. KDGAN: Knowledge Distillation with Generative Adversarial Networks. In *Advances in Neural Information Processing Systems (NIPS)*. 783–794.

[41] Zhuang Wang, Libing Jiang, Lei Lin, and Wenxian Yu. 2015. Building Height Estimation from High Resolution SAR Imagery via Model-Based Geometrical Structure Prediction. *Progress In Electromagnetics Research* 41 (2015), 11–24.

[42] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. 2006. Distance Metric Learning for Large Margin Nearest Neighbor Classification. In *Advances in Neural Information Processing Systems (NIPS)*. 1473–1480.

[43] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. 2016. A Discriminative Feature Learning Approach for Deep Face Recognition. In *European Conference on Computer Vision (ECCV)*. 499–515.

[44] WordPress and HitMag. 2018. LIDAR and RADAR Information. Retrieved Aug 9, 2018 from http://lidarradar.com/category/info

[45] Qiqi Xiao, Hao Luo, and Chi Zhang. 2017. Margin Sample Mining Loss: A Deep Learning Based Method for Person Re-identification. *arXiv preprint arXiv:1710.00478* (2017).

[46] Jiangye Yuan and Anil M. Cheriyadat. 2016. Combining Maps and Street Level Images for Building Height and Facade Estimation. In *ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics*. 8:1–8:8.

[47] Paul A. Zandbergen and Sean J. Barbeau. 2011. Positional Accuracy of Assisted GPS Data from High-Sensitivity GPS-Enabled Mobile Phones. *The Journal of Navigation* 64, 3 (2011), 381–399.

[48] Chuiqing Zeng, Jinfei Wang, Wenfeng Zhan, Peijun Shi, and Autumn Gambles. 2014. An Elevation Difference Model for Building Height Extraction from Stereo-Image-Derived DSMs. *International Journal of Remote Sensing* 35, 22 (2014),

7614–7630.
[49] He Zhang and Vishal M. Patel. 2017. Sparse Representation-Based Open Set Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 39, 8 (2017), 1690–1696.
[50] Rui Zhang. 2017. Geographic Knowledge Base (2017): http://www.ruizhang.info/GKB/gkb.htm.
[51] Zhengyou Zhang. 2000. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 22, 11 (2000), 1330–1334.

Domain-Constrained Advertising Keyword Generation

Hao Zhou
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems
Beijing National Research Center for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
tuxchow@gmail.com

Minlie Huang*
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems
Beijing National Research Center for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
aihuang@tsinghua.edu.cn

Yishun Mao
Sogou Inc., Beijing, China
ps-adwr@sogou-inc.com

Changlei Zhu
Sogou Inc., Beijing, China
zhuchanglei@sogou-inc.com

Peng Shu
Sogou Inc., Beijing, China
shupeng203672@sogou-inc.com

Xiaoyan Zhu
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems
Beijing National Research Center for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
zxy-dcs@tsinghua.edu.cn

ABSTRACT

Advertising (ad for short) keyword suggestion is important for sponsored search to improve online advertising and increase search revenue. There are two common challenges in this task. First, **the keyword bidding problem**: hot ad keywords are very expensive for most of the advertisers because more advertisers are bidding on more popular keywords, while unpopular keywords are difficult to discover. As a result, most ads have few chances to be presented to the users. Second, **the inefficient ad impression issue**: a large proportion of search queries, which are unpopular yet relevant to many ad keywords, have no ads presented on their search result pages. Existing retrieval-based or matching-based methods either deteriorate the bidding competition or are unable to suggest novel keywords to cover more queries, which leads to inefficient ad impressions.

To address the above issues, this work investigates to use generative neural networks for keyword generation in sponsored search. Given a purchased keyword (a word sequence) as input, our model can generate a set of keywords that are not only relevant to the input but also satisfy the domain constraint which enforces that the domain category of a generated keyword is as expected. Furthermore, a reinforcement learning algorithm is proposed to adaptively utilize domain-specific information in keyword generation. Offline evaluation shows that the proposed model can generate keywords that are diverse, novel, relevant to the source keyword, and accordant with the domain constraint. Online evaluation shows that generative models can improve coverage (COV), click-through rate (CTR), and revenue per mille (RPM) substantially in sponsored search.

CCS CONCEPTS

• **Information systems → Sponsored search advertising**; *Query log analysis*; Query suggestion.

KEYWORDS

ad keyword generation, domain constraint, reinforcement learning

ACM Reference Format:
Hao Zhou, Minlie Huang, Yishun Mao, Changlei Zhu, Peng Shu, and Xiaoyan Zhu. 2019. Domain-Constrained Advertising Keyword Generation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313570

1 INTRODUCTION

Advertising (ad for short) keyword suggestion is an important task for sponsored search which is one of the major types of online advertising and the major source of revenue for search companies. In sponsored search, a search engine first retrieves a set of advertisements whose keywords match a user issued query. It then ranks these advertising candidates according to an auction process by considering both the ad quality and the bid price of each ad [3]. Finally, the chosen advertisement is presented in a search result

*Corresponding author: Minlie Huang, aihuang@tsinghua.edu.cn.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313570

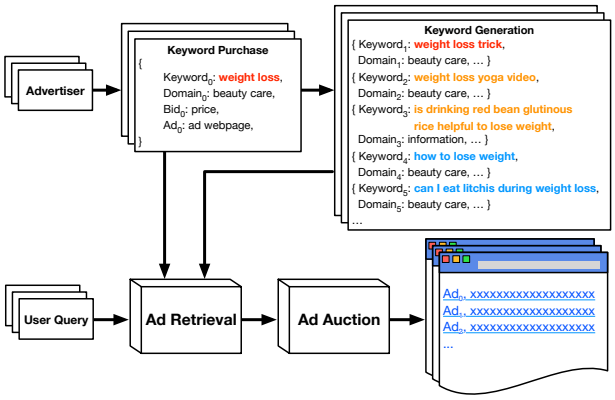


Figure 1: The sponsored search with keywords generated by our model. The red/orange/blue keywords are the keywords which have more than 3/less than 3/no ads on the impression webpage respectively.

page. Therefore, ad keywords are vital for ads to gain access to the auction process and have the chance to be displayed on a search result page.

However, there are two common challenges that should be addressed in sponsored search. The first one is the **keyword bidding problem**: due to the Matthew effect [29], the hot ad keywords become too expensive for most of the advertisers, because too many advertisers bid on such keywords. As a result, many advertisers cannot survive in the auction process to get their desired ad impressions. As reported in [43], 55.3% advertisers have no ad impression, and 92.3% advertisers have no ad click at all, which is mainly caused by low bid price or improper keywords that they bid. The second issue is **inefficient ad impressions**: a substantial proportion of search queries, which are unpopular yet relevant to many ad keywords, have less competitive ads (46.6% search queries) even no ads (41.0% search queries) on their search result pages as reported in [43]. Because of the two reasons, the expectation of advertisers is not satisfied and the revenue of search engines is also not optimized.

To address these problems, several prior studies have been conducted in keyword generation or suggestion [2, 8, 13, 23, 30]. Most of these studies adopt matching methods based on the word co-occurrence between ad keywords [8] and queries [13]. However, these methods tend to suggest popular keywords to advertisers, which will deteriorate the bidding competition. In addition, these approaches cannot suggest novel ad keywords which do not appear in the corpus.

Recently, deep learning technologies have been applied in many natural language tasks, such as machine translation [39], ad keyword suggestion [15], and query rewriting [18]. However, it's not trivial to adapt these neural networks to the ad keyword generation task, due to two major challenges. **First**, the generated ad keywords should be diversified and relevant to the original keywords to cover more user queries, which is not supported by existing neural models applied in keyword and query generation tasks. **Second**, the generated ad keywords should satisfy many constraints in sponsored search. For instance, to provide relevant yet unexplored ads for users, it is necessary to satisfy the domain constraint which means that the generated keywords should belong to the domain of the source keyword or several appropriate domains. For instance, a keyword in the *health care* domain should only match the keywords from the same domain to ensure the ad quality, while a keyword from the *information* domain could match those from various domains, such as *entertainment* and *shopping*, to cover diverse user queries.

In this paper, we investigate to use generative neural networks in the task of ad keyword generation. Given the purchased keyword as input, our generative model can suggest a set of keywords based on the semantics of the input keyword, as shown in Figure 1. The generated keywords are diverse and even completely novel (the blue keywords) from those in the dataset. This generative approach can address the aforementioned problems in two ways. **First**, our model is able to generate diverse, novel keywords, instead of merely suggesting the existing popular keywords in the dataset, which can recommend keywords for advertisers to alleviate the keyword bidding problem and retrieve ads by keyword reformulation for sponsored search engines to address the inefficient ad impression issue. **Second**, to improve the quality of the generated keywords, we incorporate the domain constraint in our model, which is a key factor considered in sponsored search to display ads. Through capturing the domain constraint, our model learns both semantic information and domain-specific information of ad keywords during training, and is consequently able to predict the proper domain category and generate the ad keyword based on the predicted category. In addition, our model uses reinforcement learning to strengthen the domain constraint in the generation process, which further improve the domain correlation and the keyword quality.

To summarize, this paper makes the following contributions:

- This work investigates to use generative neural networks for keyword generation in sponsored search, which addresses the issues of keyword bidding and inefficient ad impressions.
- We present a novel model that incorporates the domain constraint in ad keyword generation. The model is able to predict a suitable domain category and generate an ad keyword correspondingly. A reinforcement learning algorithm is devised to adaptively utilize domain-specific information in keyword generation, which further improves the domain consistency and the keyword quality.
- We perform offline and online evaluation with the proposed model, and extensive results demonstrate that our model can generate diverse, novel, relevant, and domain-consistent keywords, and also improves the performance of sponsored search.

2 RELATED WORK

2.1 Keyword Generation

A variety of methods has been proposed for generating and suggesting the keywords for advertisements, as ad keywords play a critical role in sponsored search. Joshi and Motwani [23] collected text-snippets from search engine given the keyword as input, and

constructed them as a graph model to generate relevant keywords based on the similarity score. Abhishek and Hosanagar [2] further improved the graph model, which computes the similarity score based on the retrieved documents. Chen et al. [8] applied concept hierarchy to keyword generation, which suggests new keywords according to the concept information rather than the co-occurrence of the keywords itself. Fuxman et al. [13] made use of the query-click graph to compute the keyword similarity for recommendation based on a random walk with absorbing states. Ravi et al. [32] introduced a generative approach, a monolingual statistical translation model, to generate bid phrases given the landing page, which performs significantly better than extraction-based methods. Recently due to the advances of deep learning, various neural network models have been applied to ad keyword suggestion. Grbovic et al. [15] proposed several neural language models to learn low-dimensional, distributed representations of search queries based on context and content of the ad queries within a search session. Zhai et al. [41] applied an attention network which is stacked on top of a recurrent neural network (RNN) and learns to assign attention scores to words within a sequence (either a query or an ad).

2.2 Query Generation

Another related research topic is query generation, which is widely studied and applied in organic search. It improves user experience by either expanding (reformulating) a user's query to improve retrieval performance, or providing suggestions through guessing the user intention, according to the user's behaviour pattern (query suggestion). Some previous studies adopt query logs to generate queries by handcrafted features such as click-through data [11, 25, 35, 42], session based co-occurrence [17, 20, 22] or query similarity [4, 6, 12]. Recently, artificial neural networks have been applied in query processing. A hierarchical recurrent encoder-decoder model [38] is introduced to query suggestion. He et al. [18] proposed a learning to rewrite framework consisting of a candidate generating phase and a candidate ranking phase for query rewriting. Song et al. [37] use an RNN encoder-decoder to translate a natural language query into a keyword query. An attention based hierarchical neural query suggestion model that combines a session-level neural network and a user-level neural network to model the short- and long-term search history of a user is proposed by Chen et al. [7]

2.3 Generative Neural Network

Recently, generative neural networks have been applied in many natural language tasks, such as machine translation [39], dialogue generation [34], and query rewriting [18]. Sutskever et al. [39] apply an end-to-end approach, a sequence to sequence (Seq2Seq) model, on machine translation tasks. Shang et al. [34] further introduce the Seq2Seq model to dialogue generation tasks with novel attention mechanisms. Although the Seq2Seq model is capable of generating a sequence with a certain meaning, it isn't suitable for diversified sequence generation as argued in [27]. Therefore, latent variable based models are proposed to address the diversity and uncertainty problem. Serban et al. [33] introduce latent variables to a hierarchical encoder-decoder neural network to explicitly model generative processes that possess multiple levels of variability. Zhou and Wang [45] propose several conditional variational autoencoders to use emojis to control the emotion of the generated text. Zhao et al. [44] use latent variables to learn a distribution over potential conversational intents based on conditional variational autoencoders, that is able to generate diverse responses using only greedy decoders.

3 MODEL

3.1 Background: Encoder-Attention-Decoder Framework

We first introduce a general encoder-attention-decoder framework based on sequence-to-sequence (Seq2Seq) learning [39], which is a widely used generative neural network. The encoder and decoder of the Seq2Seq model [39] are implemented with GRU [9, 10].

The encoder represents an input sequence $X = x_1 x_2 \cdots x_n$ with hidden representations $\boldsymbol{H} = \boldsymbol{h}_1 \boldsymbol{h}_2 \cdots \boldsymbol{h}_n$[1] , which is briefly defined as below:

$$\boldsymbol{h}_t = \mathbf{GRU}(\boldsymbol{h}_{t-1}, \boldsymbol{e}(x_t)), \tag{1}$$

where $\boldsymbol{e}(x_t)$ is the embedding of the word x_t, and GRU is gated recurrent unit [9].

The decoder takes as input a context vector $\boldsymbol{c}_t$ and the embedding of a previously decoded word $\boldsymbol{e}(y_{t-1})$, and updates its state $\boldsymbol{s}_t$ using another GRU:

$$\boldsymbol{s}_t = \mathbf{GRU}(\boldsymbol{s}_{t-1}, [\boldsymbol{c}_{t-1}; \boldsymbol{e}(y_{t-1})]), \tag{2}$$

where $[\boldsymbol{c}_{t-1}; \boldsymbol{e}(y_{t-1})]$ is the concatenation of the two vectors, serving as input to the GRU network. The context vector $\boldsymbol{c}_t$ is designed to attend to the key information of the input sequence during decoding, which is a weighted sum of the encoder's hidden states as $\boldsymbol{c}_{t-1} = \sum_{k=1}^{n} \alpha_k^{t-1} \boldsymbol{h}_k$, and α_k^{t-1} measures the relevance between state $\boldsymbol{s}_{t-1}$ and hidden state $\boldsymbol{h}_k$. Refer to [5] for more details.

Once the state vector $\boldsymbol{s}_t$ is obtained, the decoder generates a token by sampling from the generation distribution $\boldsymbol{o}_t$ computed from the decoder's state $\boldsymbol{s}_t$ as follows:

$$y_t \sim \boldsymbol{o}_t = P(y_t \mid y_1, y_2, \cdots, y_{t-1}, \boldsymbol{c}_t), \tag{3}$$

$$= \mathrm{softmax}(\mathbf{W_o} \boldsymbol{s}_t). \tag{4}$$

3.2 Task Definition and Overview

Our problem is formulated as follows: Given a purchased ad keyword $\boldsymbol{X} = (x_1, x_2, \cdots, x_n)$ (a word sequence)[2] and the corresponding domain category d_x obtained from a domain classifier, the goal is to predict a suitable target domain category d_y and generate a target keyword $\boldsymbol{Y} = (y_1, y_2, \cdots, y_m)$ (a word sequence) that is coherent with the domain category d_y. Essentially, the model estimates the probability: $P(\boldsymbol{Y}, d_y | \boldsymbol{X}, d_x) = P(d_y | \boldsymbol{X}, d_x) \prod_{t=1}^{m} P(y_t | y_{<t}, d_y, \boldsymbol{X}, d_x)$. The domain categories are adopted from the sponsored search engine, which consists of k domain categories, such as *beauty care*, *shopping*, and *entertainment*.

Building upon the encoder-attention-decoder framework, we propose the Domain-Constrained Keyword Generator (DCKG) to generate diversified keywords with domain constraints using three mechanisms. **First**, DCKG incorporates a latent variable sampled from a multivariate Gaussian distribution to generate diversified

[1] Throughout the paper, a bold character (e.g., $\boldsymbol{h}$) denotes the vector representation of a variable (h).

[2] Throughout the paper, a keyword refers to a word sequence, but not a single word.

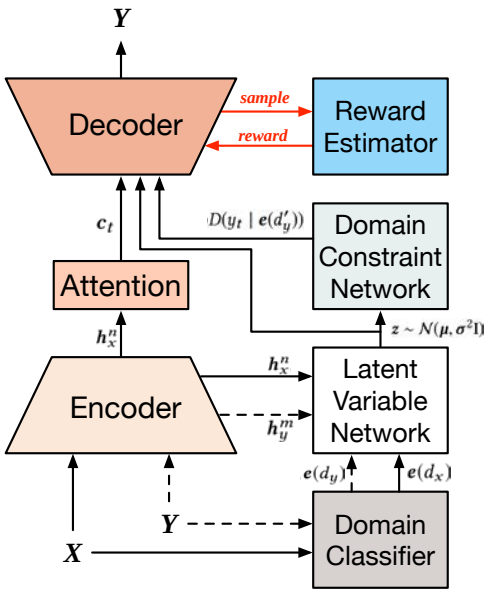


Figure 2: Overview of DCKG. The dashed arrow is used only in supervised training. The red arrow is used only in reinforcement learning. During reinforcement learning, DCKG infers a set of keyword samples and get their rewards from the reward estimator, which are used to further train our model to improve the quality of generated keywords.

keywords. **Second**, a domain constraint network is proposed to facilitate generating domain-consistent keywords, which imposes more probability bias to domain-specific words. **Third**, DCKG further optimizes the decoder to adjust the word generation distribution with reinforcement learning.

An overview of DCKG is presented in Figure 2, which illustrates the dataflow of DCKG in supervised learning, reinforcement learning, and inference processes. In supervised learning, the source keyword $X = (x_1, x_2, \cdots, x_n)$ and the target keyword $Y = (y_1, y_2, \cdots, y_m)$ are fed to the encoder to generate the hidden representations $\boldsymbol{h}_x^n$ and $\boldsymbol{h}_y^m$, meanwhile, they are fed to the domain classifier to obtain their domain categories d_x and d_y respectively. The domain categories are further converted to the domain embeddings $\boldsymbol{e}(d_x)$ and $\boldsymbol{e}(d_y)$ to encode the domain-specific information in DCKG. Then, the latent variable $\boldsymbol{z}$ is sampled from a multivariate Gaussian distribution, which is determined by a recognition network that takes the hidden representations and the domain embeddings as input. Given the latent variable $\boldsymbol{z}$, the hidden representation of source keyword $\boldsymbol{h}_x^n$, and the domain embedding $\boldsymbol{e}(d_x)$, DCKG predicts the target keyword category d'_y and generate the domain-specific word score $D(y_t \mid \boldsymbol{e}(d'_y))$. Finally, the decoder takes as input the context vector $\boldsymbol{c}_t$ generated by attention mechanism, the latent variable $\boldsymbol{z}$, and the domain-conditioned word score $D(y_t \mid \boldsymbol{e}(d'_y))$ to generate the target keyword Y.

During the inference process, DCKG has the input of only the source keyword $X = (x_1, x_2, \cdots, x_n)$ to generate a target keyword, conditioned on the latent variable sampled from a prior network which is approximated by the recognition network during supervised training.

During the reinforcement learning process, DCKG first infers a set of keyword samples. Then, these samples are fed to the reward estimator to get their rewards, considering both domain-specific and semantic information. Finally, these rewards are applied to train our model using reinforcement learning to further improve the quality of generated keywords.

3.3 Supervised Learning

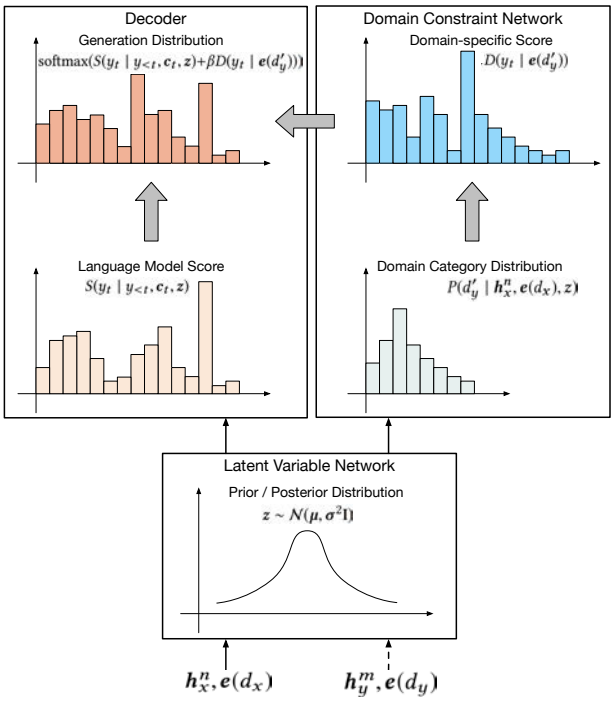


Figure 3: Computation flow of DCKG. The dashed arrow is used only in supervised learning to model the posterior distribution. During the inference process, the latent variable z is sampled from the prior distribution, and then fed to the domain constraint network and to the decoder. In the domain constraint network, the latent variable is used to predict the domain category distribution to obtain the target domain category d'_y, which is then used to compute the domain-specific score. In the decoder, the latent variable is used to compute the language model score. Finally, the language model score and the domain-specific score are combined to estimate the distribution for word generation.

We will concentrate on introducing the latent variable network, the domain constraint network, and the decoder network, as shown in Figure 3. The domain classifier is adopted from the sponsored search engine with parameters fixed during training. It generates a one-hot domain category distribution d_x for a keyword. A category is converted to a domain category embedding as follows:

$$\boldsymbol{e}(d_x) = \mathbf{V_d} d_x, \tag{5}$$

where $\mathbf{V_d}$ is a random initialized domain category embedding matrix which will be learned automatically.

3.3.1 Latent Variable Network. Building upon the encoder-attention-decoder framework, our model introduces a latent variable to project ad keywords to a latent space. By this means, the model can generate diversified keywords conditioned on different latent variable sampled from the latent space. Specifically, we adopt the conditional variational autoencoder (CVAE) [36] as the latent variable network, which is successfully applied in language generation tasks [44, 45].

DCKG assumes the latent variable $\boldsymbol{z}$ follows a multivariate Gaussian distribution, $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2\mathbf{I})$. The latent variable network consists of a prior network and a recognition network to model the semantic and domain-specific information of ad keyword in the latent space.

In the training process, the recognition network takes as input the semantic representations $\boldsymbol{h}_x^n, \boldsymbol{h}_y^m$ and the domain embeddings $\boldsymbol{e}(d_x), \boldsymbol{e}(d_y)$ of the source keyword $\boldsymbol{X}$ and the target keyword $\boldsymbol{Y}$, to approximate the true posterior distribution, as $q_\phi(\boldsymbol{z} \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{h}_y^m, \boldsymbol{e}(d_y)) \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2\mathbf{I})$.

In the inference process, the prior network takes only the semantic representation $\boldsymbol{h}_x^n$ and the domain embedding $\boldsymbol{e}(d_x)$ of the source keyword $\boldsymbol{X}$ as input, to sample latent variables from the prior distribution, $p_\theta(\boldsymbol{z} \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x)) \sim \mathcal{N}(\boldsymbol{\mu}', \sigma'^2\mathbf{I})$. The prior/posterior distribution can be parameterized by neural networks such as a multilayer perceptron (MLP) as follows:

$$\left[\boldsymbol{\mu}, \sigma^2\right] = \mathbf{MLP}(\boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{h}_y^m, \boldsymbol{e}(d_y)), \tag{6}$$

$$\left[\boldsymbol{\mu}', \sigma'^2\right] = \mathbf{MLP}(\boldsymbol{h}_x^n, \boldsymbol{e}(d_x)), \tag{7}$$

To alleviate the inconsistency between the prior distribution and the posterior distribution, we add the KL divergence term to the loss function as follows:

$$\mathcal{L}_1 = KL(q_\phi(\boldsymbol{z} \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{h}_y^m, \boldsymbol{e}(d_y)) \,\|\, p_\theta(\boldsymbol{z} \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x))) \tag{8}$$

3.3.2 Domain Constraint Network. The domain constraint network is designed to model the domain-specific information of ad keyword. The output of the network is further incorporated into the process of keyword generation to improve the quality of generated keywords in sponsored search. It plays two roles in our model: first, predicting an appropriate domain category given a latent variable; second, endowing the generated keyword with the target domain features.

Essentially, it predicts a domain category distribution conditioned on the latent variable sampled from a multivariate Gaussian distribution. Once the domain category distribution is determined, we can sample a target domain category by an argmax operation. However, argmax operation is non-differentiable and the training signal cannot be backpropagated. Inspired by the Gumbel-Max trick [16, 28], we adopt Gumbel-Softmax [19] as a differentiable substitute to generate a sample from the domain category distribution, which is defined as follows:

$$\boldsymbol{\epsilon} \sim \mathbf{U}(0, 1), \tag{9}$$

$$\boldsymbol{g} = -\log(-\log(\boldsymbol{\epsilon})), \tag{10}$$

$$\boldsymbol{o}_d = \mathbf{U_d}\,\mathbf{MLP}(\boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{z}), \tag{11}$$

$$P_{real}(d'_y \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{z}) = \mathrm{softmax}(\boldsymbol{o}_d), \tag{12}$$

$$P_{sample}(d'_y \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{z}) = \mathrm{softmax}((\boldsymbol{o}_d + \boldsymbol{g})/\tau) \tag{13}$$

where $\boldsymbol{\epsilon}$ is a sample from the uniform distribution $\mathbf{U}(0, 1)$, $\boldsymbol{g}$ is a sample from the Gumbel distribution $\mathbf{Gumbel}(0, 1)$, $\boldsymbol{o}_d$ is the logits computed by a MLP and a projection matrix $\mathbf{U_d} \in \mathbb{R}^{k \times n}$; $P_{real}(d'_y \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{z})$ is the real distribution of the predicted domain category used in the inference process, $P_{sample}(d'_y \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{z})$ is the sample distribution used in the training process, and τ is a temperature used to adjust the shape of the sample distribution, which is annealed during training.

In supervised learning, we use the ground-truth domain category d_y of a keyword as the supervision signal in the loss function, such that the domain constraint network can predict the target domain category as expected, which is defined as follows:

$$\mathcal{L}_2 = -\mathbf{E}_{q_\phi(\boldsymbol{z} \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{h}_y^m, \boldsymbol{e}(d_y))}[\log P_{real}(d'_y \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{z})] \tag{14}$$

Another task of the domain constraint network is to compute the domain-specific score of a generated word from the domain category distribution. The domain-specific score is added to the word generation distribution in the decoder to endow a generated keyword with desirable domain-specific features. When the target domain category distribution is obtained, the target domain embedding can be computed as follows:

$$\boldsymbol{e}(d'_y) = \mathbf{V_d} P(d'_y \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{z}), \tag{15}$$

where $\mathbf{V_d}$ is the domain category embedding matrix as introduced in Eq. 5. Subsequently, taking the target domain embedding as input, the domain word score is generated as follows:

$$D(y_t \mid \boldsymbol{e}(d'_y)) = \mathbf{W_d}\,\mathbf{MLP}(\boldsymbol{e}(d'_y)), \tag{16}$$

where $D(y_t \mid \boldsymbol{e}(d'_y))$ is the domain-specific score of a generated word, which models the domain-specific features of a target keyword, $\mathbf{W_d}$ is the domain word embedding matrix.

3.3.3 Decoder Network. The decoder of DCKG incorporates the latent variable and the domain word score to generate an ad keyword. Taking as input the latent variable $\boldsymbol{z}$ and the context vector $\boldsymbol{c}_t$, the language model score of a generated word, which captures the semantic information in a keyword, is generated as follows:

$$\boldsymbol{s}_t = \mathbf{GRU}(\boldsymbol{s}_{t-1}, [\boldsymbol{c}_{t-1}; \boldsymbol{z}; \boldsymbol{e}(y_{t-1})]), \tag{17}$$

$$S(y_t \mid y_{<t}, \boldsymbol{c}_t, \boldsymbol{z}) = \mathbf{W_s}\boldsymbol{s}_t, \tag{18}$$

where $S(y_t \mid y_{<t}, \boldsymbol{c}_t, \boldsymbol{z})$ is the language model score and $\mathbf{W_s}$ is the semantic word embedding matrix.

Finally, the decoder combines the language model score and the domain-specific score with a factor β and then normalizes the result to the word generation distribution, which is defined as follows:

$$P(y_t \mid y_{<t}, \boldsymbol{c}_t, \boldsymbol{z}, \boldsymbol{e}(d'_y)) = \mathrm{softmax}(S(y_t|y_{<t}, \boldsymbol{c}_t, \boldsymbol{z}) + \beta D(y_t|\boldsymbol{e}(d'_y))), \tag{19}$$

where β is a domain constraint factor that controls the influence of the domain-specific information in the final word generation distribution, and is fixed to 1.0 during supervised training. The generation loss of the decoder is given as below:

$$\mathcal{L}_3 = -\mathbf{E}_{q_\phi(\boldsymbol{z} \mid \boldsymbol{h}_x^n, \boldsymbol{e}(d_x), \boldsymbol{h}_y^m, \boldsymbol{e}(d_y))}[\sum_{t=1}^{m} \log P(y_t \mid y_{<t}, \boldsymbol{c}_t, \boldsymbol{z}, \boldsymbol{e}(d'_y))]. \tag{20}$$

3.3.4 Loss Function. The final loss to be minimized in supervised learning is the combination of the KL divergence term $\mathcal{L}_1$, the domain prediction loss $\mathcal{L}_2$, and the generation loss $\mathcal{L}_3$:

$$\mathcal{L} = \max(\delta, \mathcal{L}_1) + \mathcal{L}_2 + \mathcal{L}_3, \tag{21}$$

where the max operation and the factor δ are used to balance the KL divergence term and other loss function for better optimization, which is known as the free bits method in [24].

3.4 Reinforcement Learning

One major disadvantage of DCKG described above is the domain constraint factor β is fixed for all the keywords in any domain. However, the optimal factor should be determined by the semantic and domain-specific information of a keyword dynamically. A lower β value leads to keywords containing less domain-specific features, while a higher value results in keywords that are less fluent or relevant but contain more domain-specific features, as shown in Section 4.4. Therefore, we propose a reinforcement learning algorithm that is able to learn different β values for different keywords.

3.4.1 Policy Network. To explore suitable β values for different keywords, we first define a value space $\mathcal{B}$ which contains feasible β values. The main idea is to choose the best β value by a policy network to achieve the maximum reward with respect to the evaluation metrics, which can be implemented in three steps: first, generate a set of keywords with different β values sampled from $\mathcal{B}$, given the same source keyword and latent variable; second, obtain the reward of each keyword using the reward estimator; third, update the policy network to choose the β value that leads to a maximum reward. The policy network $\pi_\psi(\beta \mid \boldsymbol{X}, \boldsymbol{z})$, parameterized by ψ, is formally given below:

$$\boldsymbol{o}_b = [\boldsymbol{h}_x^n; \boldsymbol{e}(d_x); \boldsymbol{z}; \boldsymbol{e}(d'_y)], \tag{22}$$

$$\pi_\psi(\beta \mid \boldsymbol{X}, \boldsymbol{z}) = \text{softmax}(\mathbf{W_b}\,\mathbf{MLP}(\boldsymbol{o}_b)), \tag{23}$$

where $\mathbf{W_b} \in \mathbb{R}^{b \times n}$ is a matrix projecting the input vector $\boldsymbol{o}_b$ to the action space $\mathcal{B}$.

We use the REINFORCE algorithm [40], a policy gradient method, to optimize the parameters by maximizing the expected reward of a generated keyword as follows:

$$\mathcal{J}(\psi) = \mathbb{E}_{\beta \sim \pi_\psi(\beta|\boldsymbol{X},\boldsymbol{z})}[R(\boldsymbol{X}, \boldsymbol{z}, \beta)], \tag{24}$$

where $R(\boldsymbol{X}, \boldsymbol{z}, \beta)$ is the normalized reward of a generated keyword. It is noteworthy that the policy network cannot be optimized through supervised learning, as the ground-truth β value is not observable.

3.4.2 Reward Estimator. The reward estimator is designed to provide a reward balancing the domain-specific information and the semantic information for a generated keyword. To estimate the reward, given the source keyword $\boldsymbol{X}$, the latent variable $\boldsymbol{z}$, and the target domain category d'_y as the same input, we first sample a set of β values $\{\beta_1, \beta_2, \cdots, \beta_k\}$, and infer a set of keyword samples $\{\boldsymbol{Y}_1, \boldsymbol{Y}_2, \cdots, \boldsymbol{Y}_k\}$ based on different β samples using the proposed model. Then, we use the domain classifier to predict the domain category d_{y_i} of each keyword $\boldsymbol{Y}_i$. The agreement γ between d'_y and d_{y_i} is treated as an evaluation metric of the domain-specific information, which is defined as follows:

$$\gamma_i = \begin{cases} 1, & \text{if } d'_y = d_{y_i} \\ 0, & \text{if } d'_y \neq d_{y_i} \end{cases} \tag{25}$$

Subsequently, we use the generation probabilities computed by a language model and DCKG as an evaluation metric of the semantic information. Finally, the min-max normalization is applied to rescale the rewards $\{r_1, r_2, \cdots, r_k\}$ for each β_i and $\boldsymbol{Y}_i$. This process is defined as follows:

$$r_i = \gamma_i(\lambda P_{LM}(\boldsymbol{Y}_i) + (1-\lambda)P_{DCKG}(\boldsymbol{Y}_i)), \tag{26}$$

$$R_i(\boldsymbol{X}, \boldsymbol{z}, \beta_i) = \frac{r_i - \min(\{r_i\})}{\max(\{r_i\}) - \min(\{r_i\})}, \tag{27}$$

where r_i is the reward for each β_i and $\boldsymbol{Y}_i$, γ_i is the domain category agreement, P_{LM} and P_{DCKG} are the generation probabilities modeled by a language model and our DCKG respectively, $\lambda \in (0, 1)$ is a weight to balance these two generation probabilities.

And the gradient is approximated using the likelihood ratio trick [14] as:

$$\nabla\mathcal{J}(\psi) = R(\boldsymbol{X}, \boldsymbol{z}, \beta)\nabla\log\pi_\psi(\beta \mid \boldsymbol{X}, \boldsymbol{z}), \tag{28}$$

This RL component encourages the model to choose for each keyword an optimal factor β that will lead to both higher domain category agreement and generation quality. Through reinforcement learning, we can further improve the quality of keywords generated by DCKG, resulting in more relevant, fluent, and domain-specific keywords.

4 EXPERIMENTS

4.1 Dataset

Table 1: Statistics of the dataset.

Pairs		Details	Keywords	Queries
Training	43,756,585	Number	4,419,555	19,203,972
		Length	3.96	5.40
Validation	10,000	Number	9,504	9,997
		Length	3.73	5.21
Test	10,000	Number	9,474	9,996
		Length	3.74	5.21

We sampled about 40 million query logs from Sogou.com, and each sample consists of a <*ad keyword, user query*> pair. When the search engine retrieves ads based on their owners' purchased keywords, the ad keyword must be relevant to the user query. Therefore, we treat user queries as target keywords to train our model. In other words, the input to our model is a purchased ad keyword ($\boldsymbol{X}$), and the expected output is a user query ($\boldsymbol{Y}$). We randomly sampled 10,000 pairs for validation and test. The statistics are presented in Table 1.

4.2 Implementation Details

Our model was implemented with Tensorflow [1]. The encoder and decoder do not share parameters and each has 4-layer GRU

networks with 1024 hidden neurons in each layer. The word embedding size is set to 1,024. The vocabulary size is limited to 40,000. The MLP is implemented as a one-layer linear transformation with a tanh activation function. The parameters of the MLP and the embedding matrix share the same hidden size $n = 1,024$. All the parameters and embeddings are randomly initialized and tuned during end-to-end training.

We adopted top $k = 25$ frequent domain categories produced by the domain classifier, which is a key component used in the sponsored search engine. The domain classifier is a support vector machine [21] trained on a large human-labeled corpus. The accuracy of the 25-class classification is 92.65%. The temperature τ is set to 3.0 at the beginning, and is annealed to 0.1 during supervised learning. The factor δ is set to 5.0. The value space $\mathcal{B}$ consists of $b = 21$ values in the range $[0, 5]$ with an interval of 0.25. The generation probability factor λ is set to 0.9.

Our model is trained in two stages: supervised learning is first used to model the semantic information and the domain-specific information of keywords; reinforcement learning is then used to adaptively utilize domain-specific information in the generation process. We used the Adam optimizer with a mini-batch size of 100. The learning rate is 0.0001 for supervised learning and 0.00001 for reinforcement learning. The models were run at most 10 epochs for supervised learning and 2 epochs for reinforcement learning. The training process of each model took about a week on a Tesla V100 GPU machine.

4.3 Baselines

As aforementioned, this paper is the first work to address the domain constraint factor in ad keyword generation. We did not find closely-related baselines in the literature.

Although retrieval-based approaches proposed in previous studies are able to recommend relevant keywords from the dataset, these methods cannot generate novel keywords and thus deteriorate the bidding competition problem. Furthermore, as the dataset consists of <*ad keyword, user query*> pairs, retrieval-based methods can only suggest keywords which are already covered by the search engine. Consequently, the coverage and revenue of the search engine cannot be improved more than those methods which can generate novel keywords. Due to these reasons, we didn't consider retrieval-based methods as our baselines.

Nevertheless, we chose two suitable generative models as our baselines:

- A modified Seq2Seq model [39]. First, the target domain category is predicted by the domain constraint network and is embedded into a category vector. Then, the vector serves as input to each decoding position.
- A modified CVAE model [36] considering the domain constraint information, which is implemented similarly to DCKG but without reinforcement learning.

4.4 Automatic Evaluation

We evaluated the quality of keywords generated by different models using several automatic metrics. To evaluate the performance in metrics except for perplexity[3], we generated 10 keywords for each purchased ad keyword using different models (for DCKG and CVAE, keywords were generated conditioned on 10 sampled latent variables using greedy search; for Seq2Seq, keywords were inferred using beam search with $beamsize = 10$). Due to the uncertainty caused by the latent variable in our model and the CVAE baseline, we repeatedly tested all the models for 10 times. The averaged performance, the standard deviation, and the significance test results were reported for these models [4]. The bold number indicates the best performing model, which outperforms all other models significantly (2-tailed t-test, $p - value < 0.01$).

4.4.1 *Metrics.*

- Perplexity: We adopted perplexity to evaluate the generation quality with respect to grammar and fluency. The perplexity here is computed by the generation distribution in the models themselves.
- Perplexity$_{LM}$: We trained a language model on our dataset, and used this language model to calculate the perplexity of keywords generated by different models.
- Accuracy: To evaluate whether the model can generate domain-constrained keywords, we adopted the domain accuracy as the agreement between the expected domain category (as predicted by the model, see Eq. 12) and the domain category of a generated keyword predicted by the domain classifier.
- Distinct-n: We calculated the proportion of distinct n-grams to all the n-grams in generated keywords to evaluate the diversity.
- Novelty$_{ALL/AD}$-n: To evaluate the novelty of the generated keywords, we counted the distinct n-grams that don't appear in all the corpus (ALL) or not in the purchased ad keyword set (AD). Novelty$_{ALL}$-n means the percentage of totally novel n-grams that have not been observed in the corpus.

4.4.2 *Results.*

The results are shown in Table 2 and Table 3, from which we made the following observations:

First, DCKG obtains the best performance in domain accuracy, distinct, and novelty. The improvement in domain accuracy shows that our model can enhance the domain consistency of the generated keywords, thanks to the reward we applied in reinforcement learning. DCKG can generate more diverse and novel keywords than baselines, which can be attributed to the latent variable network we adopted and the domain constraint we considered. Our model has better perplexity and perplexity$_{LM}$ than Seq2Seq but slightly worse than CVAE, because DCKG is tuned to satisfy the domain constraint more by reinforcement learning than fit to the language generation probability. Nevertheless, the disadvantage in perplexity and perplexity$_{LM}$ harms the grammaticality of the generated keywords but less influences the overall performance, as shown in the following experiments.

Second, the novelty scores indicate that our model can generate more novel n-grams than the baselines. The generative models can generate novel keywords which have not been purchased yet so

[3] We generated the ground-truth target keyword using greedy search for calculating the perplexity.

[4] For the deterministic Seq2Seq model, we tested 10 models which achieve the top 10 performance in the validation set. These models differ in different training steps.

Table 2: Automatic evaluation.

Model	Perplexity		Perplexity$_{LM}$		Accuracy		Distinct-2		Distinct-3		Distinct-4	
	avg.	std.	avg.	std.	avg.(%)	std.(%)	avg.(%)	std.(%)	avg.(%)	std.(%)	avg.(%)	std.(%)
Seq2Seq	18.82	0.033	12.26	0.022	76.84	0.054	39.09	0.028	46.78	0.033	53.02	0.041
CVAE	**10.76**	0.021	**8.24**	0.023	77.40	0.057	69.58	0.051	83.04	0.060	90.32	0.065
DCKG	11.21	0.025	8.94	0.034	**84.61**	0.080	**71.07**	0.077	**84.26**	0.065	**91.26**	0.051

Table 3: Novelty evaluation.

Model	Novelty$_{ALL}$-2		Novelty$_{ALL}$-3		Novelty$_{ALL}$-4		Novelty$_{AD}$-2		Novelty$_{AD}$-3		Novelty$_{AD}$-4	
	avg.	std.	avg.	std.	avg.(%)	std.(%)	avg.(%)	std.(%)	avg.(%)	std.(%)	avg.(%)	std.(%)
Seq2Seq	0.54	0.005	5.33	0.015	15.53	0.020	2.65	0.010	12.40	0.026	24.73	0.019
CVAE	2.01	0.016	16.03	0.058	38.38	0.119	8.18	0.024	33.49	0.085	57.64	0.099
DCKG	**2.60**	0.021	**18.23**	0.056	**41.48**	0.114	**9.34**	0.035	**36.01**	0.095	**60.40**	0.106

that the bidding competition problem can be alleviated as aforementioned. For retrieval-based methods, the Novelty$_{ALL}$ scores, which count n-grams that are never observed in the corpus, will be zero since they can only suggest existing keywords from the corpus.

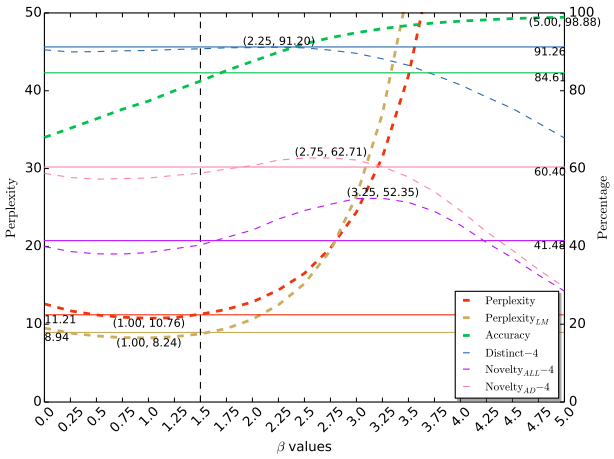


Figure 4: Performance curve with varying β. The dashed curve is the performance of DCKG with manually fixed β in different metrics. The bold curves (perplexity, perplexity$_{LM}$, accuracy) show the change of rewards which are used reinforcement learning (see Eq. 26). The horizontal line is the performance of DCKG with dynamic β tuned by RL. The dashed vertical line is an auxiliary line to compare DCKG with fixed $\beta = 1.5$ to DCKG with dynamic β optimized by RL. Left axis: Perplexity and Perplexity$_{LM}$. Right axis: Accuracy, Distinct-4, Novelty$_{ALL}$-4, and Novelty$_{AD}$-4.

4.4.3 Effect of β. In order to investigate the influence of the domain constraint factor β in DCKG, we tested the performance of DCKG with different fixed $\beta \in \mathcal{B}$. The results are shown in Figure 4. As we can see, the domain accuracy of DCKG improves continuously with the increase of β (the green curve), indicating that β is critical to satisfying the domain constraint in keyword generation because larger β applies more domain-specific information in keyword generation. DCKG achieves the best performance in perplexity and perplexity$_{LM}$ with $\beta = 1$ since this value is the default β value during supervised learning. However, when β becomes larger, DCKG performs worse in the perplexity metrics. It is thus important to balance the domain constraint and the language generation quality. Therefore, we apply reinforcement learning to find an optimal β for different instances dynamically.

It is noteworthy that DCKG with the dynamically determined β outperforms DCKG with the manually adjusted β. For example, although the performance of DCKG in perplexity and perplexity$_{LM}$ is approximately equal to DCKG with the fixed $\beta = 1.5$, the domain accuracy of DCKG outperforms DCKG with the fixed $\beta = 1.5$ (**84.61** vs. 82.49). Moreover, the novelty score and the diversity score of DCKG are also better than DCKG with the fixed $\beta = 1.5$, for instance, the Distinct-4 of DCKG is better than DCKG with any fixed $\beta \in \mathcal{B}$ (**91.26** vs. 91.20), indicating that the domain constraint can further improve the quality of generated keywords.

4.5 Manual Evaluation

In manual evaluation, we tested whether the generated keywords are relevant to the source keyword and consistent with the expected domains, such that they are suitable to be suggested to advertisers. We manually evaluated the relevance and the grammaticality of keywords generated by different models. We randomly sampled 100 purchased ad keywords from the test set, and generated 10 keywords for each purchased ad keyword using different models as in automatic evaluation. Five experienced annotators were recruited to judge the relevance (*Relevant/Irrelevant*) of 3000 pairs of generated and purchased keywords. The grammaticality of a generated keyword is also considered during annotation, as bad grammaticality leads to the ambiguity in the meaning and thus results in irrelevance rating. We used majority voting to obtain the final label of annotation. Moreover, as domain accuracy is critical to measuring how well the models satisfy domain constraint, we also reported the automatic measure in these tables. The bold number indicates the best performing model, which outperforms all other models significantly (2-tailed sign test, $p - value < 0.05$).

4.5.1 Metrics.

- Precision (P): Precision is defined as the ratio of the number of relevant keywords generated to the number of total keywords generated.
- Recall (R): Recall is the proportion of relevant keywords generated by a model to all the relevant keywords pooled from all models' results. Similar to [23], the relevant keywords from all the models are pooled together and are treated as the entire relevant set. This metric is useful to compare the ability to cover the potential relevant keywords.
- Accuracy (A): Similar to the preceding section, this metric is the agreement between the expected domain category, predicted by the model, and the domain category of a generated keyword, predicted by the domain classifier (SVM).
- F-measures: We adopted four F-measures to quantify the overall performance including F(PR), F(PA), F(RA), and F(PRA). Each F-measure is defined as the harmonic mean of all factors, where P/R/A indicate precision/recall/accuracy respectively.

Table 4: Manual evaluation.

Model	Seq2Seq	CVAE	DCKG
Precision	0.948	0.954	**0.970**
Recall	0.325	0.332	**0.343**
Accuracy	0.761	0.774	**0.847**
F(PR)	0.483	0.491	**0.503**
F(PA)	0.808	0.832	**0.892**
F(RA)	0.440	0.454	**0.480**
F(PRA)	0.534	0.548	**0.574**

4.5.2 Annotation Statistics. We calculated the statistics to measure inter-rater agreement. For relevance rating, the percentage of the pairs that all 5 judges gave the same label is 85.1%, the percentage of the pairs that 4 judges gave the same label (4/5 agreement) amounts to 8.8%, and the percentage for 3/5 agreement is 6.1%. We further computed the free-marginal kappa [31] to measure inter-rater consistency. The free-marginal kappa is 0.856, indicating adequate inter-rater agreement.

4.5.3 Results. The results are shown in Table 4. As it can be seen, DCKG obtains the best performance in all the metrics, indicating that our model can generate relevant and grammatical keywords. The best precision shows that more relevant keywords are generated by DCKG. Our model also achieves the best recall, indicating that DCKG is able to cover more potential user queries than other models. In addition, DCKG outperforms other models in domain accuracy, indicating that incorporating the domain constraint appropriately can enhance the quality of generated keywords. Moreover, DCKG ranks highest for all the four F-measures, showing that our model is superior considering all these factors.

4.6 Online Evaluation

In online evaluation, we examined whether the sponsored search engine can retrieve and present more relevant ads using the keywords generated by our model. We applied the generative models in the keyword reformulation method to facilitate the ad retrieval process of sponsored search (see Figure 1). Specifically, we collected 5 million purchased ad keywords, and then used each model to generate 10 keywords for each purchased keyword. We used these generated keywords as the reformulated keywords to retrieve original ads. The original keyword reformulation method, based on handcrafted rules and templates, is one of the retrieval methods in the ad retrieval process of the sponsored search engine, which retrieves about 2% unique ads. We added the keywords generated by our models to the keyword reformulation method of the sponsored search engine, which is called the enhanced keyword reformulation method. To make a fair comparison, the enhanced keyword reformulation methods of all models run ten days in our A/B online test system on Sogou.com, where 10% user queries are selected into the test system. We compared the performance of the enhanced keyword reformulation method with the original keyword reformulation method, and reported the relative gain in percentage for all the metrics as the final result. The bold number indicates the best performing model, which outperforms all other models significantly (2-tailed t-test, $p - value < 0.05$).

4.6.1 Metrics.

- Coverage (COV): Coverage is defined as the percentage of web pages that contain at least one ad.
- Click-through rate (CTR): Click-through rate is the ratio of page views that lead to a click to the total number of page views.
- Revenue per mille (RPM): Revenue per mille is the revenue of the search engine per one thousand page views.

4.6.2 Results. The results are shown in Table 5. As it can be seen, DCKG obtains the best performance in all the metrics. Results show that keywords generated by DCKG can cover more user queries than those by other models. The click-through rate scores indicate that the new ads which are matched with ad keywords from DCKG lead to more user clicks, because the generated keywords (user queries) are of higher quality and more relevant. The revenue is improved because the click-through rate increases, which verifies that our model can contribute more ad impressions in sponsored search.

Although the Seq2Seq model can also retrieve more ads, the improvements in all metrics are the lowest in the three models, as the diversity and the quality of the generated keywords are lower than ours. CVAE outperforms Seq2Seq because CVAE is able to model diversity with latent variables. However, as CVAE incorporate the domain constraint in a fixed way, it cannot balance the semantic information and the domain-specific information in the generation process. DCKG further improves the diversity and the quality of generated keywords by tuning the β factor dynamically with reinforcement learning. The results of online evaluation agree with those of automatic evaluation, indicating that DCKG can generate the most diverse, novel and domain consistent keywords among the three models.

4.7 Case Study

Several sample keywords generated by different models are shown in Table 6. We can see that keywords generated by Seq2Seq have

Table 5: Online evaluation. The number reported in each metric is the increased proportion compared to the original keyword reformulation method of the sponsored search engine, which retrieves about 2% unique ads.

Day	Seq2Seq			CVAE			DCKG		
	COV(%)	CTR(%)	RPM(%)	COV(%)	CTR(%)	RPM(%)	COV(%)	CTR(%)	RPM(%)
1	39.51	44.92	21.96	53.64	59.14	24.86	58.52	63.26	30.52
2	44.30	49.99	24.99	58.22	60.38	28.49	55.16	60.88	33.49
3	46.38	50.02	22.52	51.53	54.04	30.30	58.07	58.78	28.24
4	47.32	46.22	16.78	51.12	57.54	34.64	58.24	61.68	28.23
5	46.41	50.25	20.56	49.82	52.58	21.72	59.40	60.85	33.78
6	45.27	47.06	17.36	51.61	51.53	24.95	55.20	58.17	21.68
7	41.09	45.37	26.00	51.54	55.39	24.12	54.36	59.58	24.81
8	36.90	49.98	17.42	51.53	55.96	23.15	53.38	58.29	27.41
9	35.89	42.85	19.25	51.92	61.53	27.00	50.60	63.24	37.75
10	39.46	49.36	25.07	56.17	60.54	28.06	53.86	57.61	22.06
avg.	42.25	47.60	21.19	52.71	56.86	26.73	**55.68**	**60.23**	**28.80**

Table 6: Sample keywords generated by different models with domain accuracy (Acc.) and relevance (Rel.).

Model	Domain		Keyword		Acc.	Rel.
	Original	Translated	Original	Translated		
Source	美容保健	beauty care	减肥减肥	weight loss weight loss	-	-
Seq2Seq	美容保健	beauty care	抖音上的减肥的舞蹈视频	dance video on weight loss	0	1
	美容保健	beauty care	抖音上的减肥的舞蹈视频教程	dance video tutorial on weight loss	1	1
	美容保健	beauty care	减肥的人可以吃什么水果不容易	what fruit is not easy for people who want to lose weight	1	0
	美容保健	beauty care	减肥的人可以吃什么水果不容易胖	what fruit is not easy to fat for people who want to lose weight	1	1
CVAE	信息	information	减肥的时候吃的热量低的食物	low calorie food when losing weight	1	1
	信息	information	减肥吃什么肉会胖	what meat will lead to fat if I eat during weight loss	0	1
	美容保健	beauty care	减肥期间吃什么主食最好	what is the best staple food during weight loss	1	1
	电脑网络	Internet	一天中什么时候减肥最好	when is the best time to lose weight in a day	0	1
DCKG	美容保健	beauty care	减肥瑜伽视频	weight loss yoga video	1	1
	美容保健	beauty care	减肥小妙招	weight loss trick	1	1
	美容保健	beauty care	减肥能吃荔枝吗	can I eat litchis during weight loss	1	1
	信息	information	喝红豆薏米减肥吗	is drinking red bean glutinous rice helpful to lose weight	1	1

lower diversity than other models since it tends to generate keywords with the same prefix in similar meanings using beam search [26], such as the 3rd and 4th keywords generated by Seq2Seq. Moreover, it can only predict one target domain category since it is a deterministic model, which cannot handle the task of generating different keywords from multi-domain categories.

In comparison, CVAE and DCKG are able to predict multiple domain categories as well as generate diverse keywords according to these domain categories. However, in CVAE, the domain constraint is not addressed properly in some cases, as it takes a fixed way to combine the domain-specific information with the semantic information. For example, although the 6th generated keyword, "what meat will lead to fat if I eat during weight loss", is relevant to the source keyword, it is inconsistent with its domain category "information" since the domain-specific information is less considered than the semantic information in this case. By contrast, DCKG is able to choose a proper domain constraint factor to balance the domain-specific information and the semantic information for each instance. Hence, the generated keywords are both relevant to the source keyword and consistent with their domain categories.

In order to validate the influence of the domain constraint factor β in the generated keywords, we presented several sample keywords generated by DCKG with different β values in Table 7. Take the 6th keyword in Table 6 as an example. Given the source keywords "weight loss weight loss" and its domain category "beauty care", our model predicts a target domain category "information", and then generates keywords with different β values while other parameters are fixed.

Table 7: Sample keywords conditioned on different β with domain accuracy (Acc.) and relevance (Rel.).

β	Keyword		Acc.	Rel.
0.00	Original	减肥瘦身应该吃什么	0	1
	Translated	what should I eat to lose weight and slim		
1.00	Original	减肥吃什么肉会胖	0	1
	Translated	what meat will lead to fat if I eat during weight loss		
2.00	Original	减肥吃牛肉有什么好处	1	1
	Translated	the benefits of eating beef during weight loss		
3.00	Original	减肥吃牛肉有什么好处	1	1
	Translated	the benefits of eating beef during weight loss		
4.00	Original	减肥公司调查报告范文	1	0
	Translated	weight loss company survey report essay		
5.00	Original	减肥公司调查调查信息	1	0
	Translated	weight loss company survey survey information		

As shown in Table 6, when β equals to 0.00, the target keyword belongs to the source domain category "beauty care" but violates the target domain category "information" since the keyword is generated regardless of the domain-specific information. With the increase of β, the domain accuracy improves since more domain-specific information is involved in the model, but the relevance decreases in some instances because the model faces with the compatibility of controlling semantic and domain-specific information. For example, the keywords generated when $\beta = 2.00/3.00$, are both consistent to the target domain category and relevant to the source keyword. However, when β increases to 4.00/5.00, the generated keywords are not relevant to the source keyword anymore, as a high β value will harm the quality of generated keywords severely.

5 CONCLUSION

In this paper, we investigate the use of generative neural networks for ad keyword generation. We propose a Domain-Constrained Keyword Generator (DCKG) to generate diversified and domain-consistent keywords. A latent variable network is designed to model the diversity and a domain constraint network to satisfy the domain constraint in supervised learning. To further leverage the domain-specific information in keyword generation, we propose a reinforcement learning algorithm to adaptively utilize domain-specific information in keyword generation. Offline evaluation shows that the proposed model can generate keywords that are diverse, novel, relevant to the source keyword, and accordant with the domain constraint. Online evaluation shows that generative models can improve coverage (COV), click-through rate (CTR), and revenue per mille (RPM) substantially in sponsored search.

ACKNOWLEDGMENTS

This work was jointly supported by the National Science Foundation of China (Grant No.61876096/61332007), and the National Key R&D Program of China (Grant No. 2018YFC0830200).

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *CoRR* abs/1603.04467 (2016).
[2] Vibhanshu Abhishek and Kartik Hosanagar. 2007. Keyword generation for search engine advertising using semantic similarity between terms. In *Proceedings of the ninth international conference on Electronic commerce*. ACM, 89–94.
[3] Gagan Aggarwal, Ashish Goel, and Rajeev Motwani. 2006. Truthful auctions for pricing search keywords. In *Proceedings of the 7th ACM conference on Electronic commerce*. ACM, 1–7.
[4] Ioannis Antonellis, Hector Garcia Molina, and Chi Chao Chang. 2008. Simrank++: query rewriting through link analysis of the click graph. *Proceedings of the VLDB Endowment* 1, 1 (2008), 408–421.
[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR* abs/1409.0473 (2014).
[6] Lidong Bing, Wai Lam, Tak-Lam Wong, and Shoaib Jameel. 2015. Web query reformulation via joint modeling of latent topic dependency and term context. *ACM Transactions on Information Systems (TOIS)* 33, 2 (2015), 6.
[7] Wanyu Chen, Fei Cai, Honghui Chen, and Maarten de Rijke. 2018. Attention-based Hierarchical Neural Query Suggestion. *arXiv preprint arXiv:1805.02816* (2018).
[8] Yifan Chen, Gui-Rong Xue, and Yong Yu. 2008. Advertising keyword suggestion based on concept hierarchy. In *Proceedings of the 2008 international conference on web search and data mining*. ACM, 251–260.
[9] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*. 1724–1734.
[10] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR* abs/1412.3555 (2014).
[11] Hang Cui, Ji-Rong Wen, Jian-Yun Nie, and Wei-Ying Ma. 2002. Probabilistic query expansion using query logs. In *Proceedings of the 11th international conference on World Wide Web*. ACM, 325–332.
[12] Bruno M Fonseca, Paulo Golgher, Bruno Pôssas, Berthier Ribeiro-Neto, and Nivio Ziviani. 2005. Concept-based interactive query expansion. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 696–703.
[13] Ariel Fuxman, Panayiotis Tsaparas, Kannan Achan, and Rakesh Agrawal. 2008. Using the wisdom of the crowds for keyword generation. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 61–70.
[14] Peter W Glynn. 1990. Likelihood ratio gradient estimation for stochastic systems. *Commun. ACM* 33, 10 (1990), 75–84.
[15] Mihajlo Grbovic, Nemanja Djuric, Vladan Radosavljevic, Fabrizio Silvestri, and Narayan Bhamidipati. 2015. Context-and content-aware embeddings for query rewriting in sponsored search. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. ACM, 383–392.
[16] Emil Julius Gumbel. 1954. Statistical theory of extreme values and some practical applications. *NBS Applied Mathematics Series* 33 (1954).
[17] Qi He, Daxin Jiang, Zhen Liao, Steven CH Hoi, Kuiyu Chang, Ee-Peng Lim, and Hang Li. 2009. Web query recommendation via sequential query prediction. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*. IEEE, 1443–1454.
[18] Yunlong He, Jiliang Tang, Hua Ouyang, Changsung Kang, Dawei Yin, and Yi Chang. 2016. Learning to rewrite queries. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1443–1452.
[19] Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016).
[20] Jyun-Yu Jiang, Yen-Yu Ke, Pao-Yu Chien, and Pu-Jen Cheng. 2014. Learning user reformulation behavior for query auto-completion. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 445–454.
[21] Thorsten Joachims. 1998. *Making large-scale SVM learning practical.* Technical Report. Technical Report, SFB 475: Komplexitätsreduktion in Multivariaten Datenstrukturen, Universität Dortmund.
[22] Rosie Jones, Benjamin Rey, Omid Madani, and Wiley Greiner. 2006. Generating query substitutions. In *Proceedings of the 15th international conference on World Wide Web*. ACM, 387–396.
[23] Amruta Joshi and Rajeev Motwani. 2006. Keyword generation for search engine advertising. In *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*. IEEE, 490–496.
[24] Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. 2016. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*. 4743–4751.

[25] Kenneth Wai-Ting Leung, Wilfred Ng, and Dik Lun Lee. 2008. Personalized concept-based clustering of search engine queries. *IEEE transactions on knowledge and data engineering* 20, 11 (2008), 1505–1518.

[26] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *NAACL*. 110–119.

[27] Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. A Simple, Fast Diverse Decoding Algorithm for Neural Generation. *CoRR* abs/1611.08562 (2016).

[28] Chris J Maddison, Daniel Tarlow, and Tom Minka. 2014. A* sampling. In *Advances in Neural Information Processing Systems*. 3086–3094.

[29] Robert K Merton. 1968. The Matthew effect in science: The reward and communication systems of science are considered. *Science* 159, 3810 (1968), 56–63.

[30] Dandan Qiao and Jin Zhang. 2015. A Novel Keyword Suggestion Method to Achieve Competitive Advertising on Search Engines.. In *PACIS*. 142.

[31] Justus J Randolph. 2005. Free-Marginal Multirater Kappa (multirater K [free]): An Alternative to Fleiss' Fixed-Marginal Multirater Kappa. *Online submission* (2005).

[32] Sujith Ravi, Andrei Broder, Evgeniy Gabrilovich, Vanja Josifovski, Sandeep Pandey, and Bo Pang. 2010. Automatic generation of bid phrases for online advertising. In *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 341–350.

[33] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues.. In *AAAI*. 3295–3301.

[34] Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. In *ACL*. 1577–1586.

[35] Marc Sloan, Hui Yang, and Jun Wang. 2015. A term-based methodology for query reformulation understanding. *Information Retrieval Journal* 18, 2 (2015), 145–165.

[36] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*. 3483–3491.

[37] Hyun-Je Song, A Kim, Seong-Bae Park, et al. 2017. Translation of Natural Language Query Into Keyword Query Using a RNN Encoder-Decoder. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 965–968.

[38] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 553–562.

[39] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*. 3104–3112.

[40] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

[41] Shuangfei Zhai, Keng-hao Chang, Ruofei Zhang, and Zhongfei Mark Zhang. 2016. Deepintent: Learning attentions for online advertising with recurrent neural networks. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 1295–1304.

[42] Wei Vivian Zhang and Rosie Jones. 2007. Comparing click logs and editorial labels for training query rewriting. In *WWW 2007 Workshop on Query Log Analysis: Social And Technological Challenges*.

[43] Ying Zhang, Weinan Zhang, Bin Gao, Xiaojie Yuan, and Tie-Yan Liu. 2014. Bid keyword suggestion in sponsored search based on competitiveness and relevance. *Information Processing & Management* 50, 4 (2014), 508–523.

[44] Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 654–664.

[45] Xianda Zhou and William Yang Wang. 2017. MojiTalk: Generating Emotional Responses at Scale. *arXiv preprint arXiv:1711.04090* (2017).

Predicting ConceptNet Path Quality Using Crowdsourced Assessments of Naturalness

Yilun Zhou
MIT CSAIL
yilun@mit.edu

Steven Schockaert
Cardiff University
SchockaertS1@cardiff.ac.uk

Julie A. Shah
MIT CSAIL
julie_a_shah@csail.mit.edu

ABSTRACT

In many applications, it is important to characterize the way in which two concepts are semantically related. Knowledge graphs such as ConceptNet provide a rich source of information for such characterizations by encoding relations between concepts as edges in a graph. When two concepts are not directly connected by an edge, their relationship can still be described in terms of the paths that connect them. Unfortunately, many of these paths are uninformative and noisy, which means that the success of applications that use such path features crucially relies on their ability to select high-quality paths. In existing applications, this path selection process is based on relatively simple heuristics. In this paper we instead propose to learn to predict path quality from crowdsourced human assessments. Since we are interested in a generic task-independent notion of quality, we simply ask human participants to rank paths according to their subjective assessment of the paths' *naturalness*, without attempting to define naturalness or steering the participants towards particular indicators of quality. We show that a neural network model trained on these assessments is able to predict human judgments on unseen paths with near optimal performance. Most notably, we find that the resulting path selection method is substantially better than the current heuristic approaches at identifying meaningful paths.

CCS CONCEPTS

• **Information systems** **Crowdsourcing**; *Answer ranking*; • **Computing methodologies** **Knowledge representation and reasoning**; *Natural language processing*.

KEYWORDS

Crowdsourcing, Knowledge Graph, Feature Selection, Commonsense Knowledge, ConceptNet

ACM Reference Format:
Yilun Zhou, Steven Schockaert, and Julie A. Shah. 2019. Predicting ConceptNet Path Quality Using Crowdsourced Assessments of Naturalness. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313486

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313486

1 INTRODUCTION

Many applications require information about the semantic relation between two (or more) words, concepts, or entities. For example, a recommender system needs to recommend items related to a user's browsing history, a task allocation agent needs to match people's experiences and skills to a pool of problems to maximize problem-solving efficiency, and a household robot given the instruction to "wash the plates" needs to infer that a dishwasher could be used. Open-domain knowledge graphs such as ConceptNet (27) allow us to characterize such semantic relationships in the form of relational paths. Compared to the use of word embeddings (21) for characterizing relatedness, knowledge graphs have the potential advantage of producing easier-to-understand characterizations, and they can capture relationships that go beyond what is encoded in standard word embeddings (33).

Typical knowledge graphs (KGs), such as DBpedia and WikiData, are concerned with named entities and their relations (e.g. "Abraham Lincoln" is one of "United States Presidents"). In this paper, however, we are concerned with capturing semantic relationships between common nouns or concepts, which requires a commonsense KG such as ConceptNet. Despite its indisputable value, however, effectively using ConceptNet in applications comes with a unique set of challenges. The knowledge captured in ConceptNet is, by design, often informal, subjective and vague. Due to the fact that ConceptNet partly consists of unverified crowdsourced assertions, it is also noisier than many other KGs. Furthermore, many commonsense assertions are true only under some circumstances and to some extent. For example, ConceptNet encodes that "popcorn" is required for "watching a movie", which captures the useful commonsense knowledge that eating popcorn is associated with watching a movie, but the statement that popcorn is *required* is nonetheless false. In addition, ConceptNet only disambiguates concepts to a coarse part-of-speech level (e.g. noun meaning vs verb meaning of the word "watch"). Finally, a lot of concepts are linked by the generic "RelatedTo" relation, which covers relationships as diverse as collocations ("Tire $\xleftrightarrow{\text{RelatedTo}}$ Spare"), hypernyms ("Tire $\xleftrightarrow{\text{RelatedTo}}$ All Seasons Tire"), co-meronyms ("Tire $\xleftrightarrow{\text{RelatedTo}}$ Exhaust"), homonyms ("Tire $\xleftrightarrow{\text{RelatedTo}}$ Tier"), and very loosely related terms ("Tire $\xleftrightarrow{\text{RelatedTo}}$ Clothes").

Because of those challenges, few authors use relational paths from ConceptNet in applications. In particular, while several authors have found the knowledge encoded in ConceptNet to be highly valuable, they typically restrict themselves to using relationships that are directly encoded as an edge. For example, Speer et al. (27) showed that ConceptNet can be used to improve word embeddings by incorporating the intuition that words which are linked by an edge in ConceptNet should be represented by similar vectors.

We believe that this common restriction to direct edges is due to the lack of sufficiently accurate methods for filtering nonsensical paths, or conversely, for identifying the most natural paths. This path selection problem is the focus of our paper.

In existing work (11, 19), the problem of selecting high-quality ConceptNet paths has been addressed in a heuristic way (see Section 4). While some intuitions about high-quality paths can easily be formulated (e.g. shorter paths tend to be more informative than longer paths, the nodes occurring in natural paths tend to have similar word vector representations), such heuristic methods still fail to filter out many nonsensical paths, and conversely sometimes erroneously filter out highly valuable paths. For example, the best path between "lead" and "poison" found by one of the standard heuristics (pairwise baseline, see Section 4.3) is "Lead $\xleftrightarrow{\text{Synonym}}$ Take $\xleftrightarrow{\text{DistinctFrom}}$ Give $\xleftrightarrow{\text{RelatedTo}}$ Poison", which uses the verb meaning of "lead", despite the fact that the noun meaning (i.e. the poisonous chemical element Pb) is more relevant in this case.

Rather than trying to construct increasingly intricate heuristics, we propose to learn to predict path quality using a neural network model. To this end, we rely on crowdsourced assessments about the *naturalness* of ConceptNet paths. Specifically, to collect training data, human annotators were asked to choose the more natural path among a pair of paths, without any guidance on how to interpret naturalness. This notion of naturalness was chosen because it is intuitively easy to understand for crowdsourcers (as opposed to e.g. terms such as "semantic coherence" or "predictive value"). The term is also deliberately vague, as we do not want to steer annotators towards particular types of features. The resulting pairwise judgments are then used to train a neural network to predict a latent naturalness score, which can be used for path ranking or path selection.

The main contribution of our work is to show that people's intuitive understanding of naturalness is sufficiently coherent to be used as training data for a data-driven path selection method. To this end, we train a simple neural network model to predict latent naturalness scores that are predictive of human pairwise judgments. We find that our model can predict such judgments with a performance that is close to the optimum suggested by the inter-annotator agreement (Sec 4.2). For example, the best path found by our model for the above example is "Lead $\xrightarrow{\text{HasProperty}}$ Toxic $\xleftrightarrow{\text{RelatedTo}}$ Lethal $\xleftrightarrow{\text{RelatedTo}}$ Poison". We also show, for a number of different evaluation tasks, that our method allows us to select semantically more meaningful paths than the previously proposed heuristics.

2 RELATED WORK

Knowledge graphs: KGs explicitly encode relationships between different entities as subject-predicate-object triples. Such triples can be seen as defining a graph, where the subject and object entities refer to nodes of the graph and the predicates correspond to edge labels. KGs are typically constructed by a domain expert, via crowdsourcing, or by extracting assertions from a natural language corpus (6). In this work, we used ConceptNet (20), one of the most comprehensive commonsense knowledge graphs with 46 relation types, nearly 1 million concepts (words and phrases), and nearly 3 million edges (i.e. triples). ConceptNet partly obtained through crowdsourcing, but also incorporates external sources such as OpenCyc (18), WordNet (22), Verbosity (29), DBPedia (2), and Wiktionary[1].

Knowledge base construction by crowdsourcing: While the use of crowdsourcing for constructing knowledge bases has already been well studied, most existing approaches focus on knowledge acquisition. This can involve, for instance, direct collaborative editing of a knowledge base (3, 31) or indirect construction of a knowledge base through the use of an interactive game (29). In contrast to these works, our focus is on using crowdsourcing for learning how to detect noisy paths within an existing knowledge graph.

An important challenge with crowdsourcing is the fact that there will inevitably be disagreements within the collected data. One framework (1) suggests that this can be due to (i) the inclusion of low-quality participants, (ii) an ambiguous interpretation of the input data and (iii) an ambiguous definition of the labels that crowdsourcers are required to provide. In our work, we mitigated the first issue with a quality control mechanism (Section 4.1). To address the remaining two issues, we will rely on a probabilistic generative model to interpret the provided ratings.

Word embeddings: Word embedding methods (21, 24) represent each word in a relatively low-dimensional space of typically around 300 dimensions, which is estimated from word co-occurrence statistics. With most training procedures, vector differences represent abstract relations in the resulting embedding space: for example, $\mathbf{v}_{\text{king}} - \mathbf{v}_{\text{queen}} \approx \mathbf{v}_{\text{man}} - \mathbf{v}_{\text{woman}}$, in which $\mathbf{v}_{\text{word}}$ represents the embedding of a given word.

Both KGs and word embeddings capture aspects of meaning, and can thus to some extent be seen as alternatives. One advantage of word embeddings is that they capture types of knowledge which are difficult to encode symbolically. For example, the cosine similarity between word vectors tends to correlate very strongly with human perceptions of word similarity. On the other hand, although related words are often close to each other in the embedding space, it is difficult to "reverse-engineer" and describe the specific relation from the vector difference (5). Moreover, when multiple relations exist between two concepts (e.g. "France $\xleftrightarrow{\text{BorderWith}}$ Germany" and "France $\xleftrightarrow{\text{AllyWith}}$ Germany"), the vector difference can only reflect an aggregate. Apart from issues which arise because of the use of vectors, there are also some problems that are more generally related to the use of co-occurrence statistics for representing meaning. As a simple illustration of such problems, a system (16) incorrectly assumes that garlic has wings, because of the prevalence of phrases such as "garlic chicken wings".

Given the complementary nature of the KGs and word embeddings, it is perhaps not surprising that several studies have found it to be beneficial to integrate these two types of resources. Some studies (e.g., (10, 27, 33)) reported that knowledge graphs can be used to improve embeddings to achieve better results on various benchmark tasks such as synonym selection and analogy question solving. Conversely, embeddings can also be incorporated into methods that rely on paths in knowledge graphs, for example to

[1] https://www.wiktionary.org/

cluster such paths (11), or as features for a path selection method, as in our work.

Path Features in Applications: Several prior works have described systems that incorporated knowledge graph paths as features. For example, Boteanu and Chernova (4) used ConceptNet to solve analogy questions (e.g., "dog is to animal as banana is to fruit") by comparing similarities between paths. Lin et al. (19) proposed models for learning the embeddings of paths and showed that they performed better on problems such as relation prediction compared to embeddings computed by previous methods. Guu et al. (12) proposed to answer compositional queries (e.g., "Q: What is the population of the capital of Russia?" "A: Russia $\xrightarrow{\text{Capital}}$ Moscow $\xrightarrow{\text{Population}}$ 12.2 million") by traversing the knowledge graph in *vector space*, representing traversal as a series of matrix-vector multiplications. Das et al. (8) proposed a recurrent neural network (RNN) model with attention mechanism to reason over entities and relations in a knowledge graph, and observed improved performance in path query answering upon that by Guu et al. (12).

Among many applications, knowledge graph completion (i.e., inferring missing relations from existing paths) has been a popular target. Lao and Cohen (17) used path features for knowledge graph completion in the path ranking algorithm (PRA). Gardner et al. (11) incorporated embeddings of the *relations*, allowing the system to recognize semantically similar relations, such as "(river) runs through (city)" and "(river) flows through (city)." Neelakantan et al. (23) built upon the same PRA idea, but used an RNN to model paths. Toutanova et al. (28) proposed a dynamic programming algorithm that can incorporate both edge and vertex features and reported better performance on knowledge graph completion. While all of the above-mentioned knowledge graph completion works use some version of PRA to generate paths, Xiong et al. (32) trained a reinforcement learning agent for path generation that rewards accuracy, efficiency and diversity, and demonstrated better performance than PRA-generated paths.

Path Selection: The number of paths between two nodes in a knowledge graph tends to grow exponentially with path length. In addition, while each type of path can be viewed as encoding a kind of semantic relationship, for many paths this relationship is difficult to interpret. Thus, all of the works described above mitigate the scalability and/or quality problem via heuristics. To limit the number of paths, Boteanu and Chernova (4) stopped the path search after a pre-defined number of nodes are explored. Lao and Cohen (17) and Guu et al. (12) limited the maximum path length. To obtain natural paths, Gardner et al. (11) favored paths that contain words with higher embedding similarity. Lin et al. (19) calculated the quality of paths using a heuristic based on vertex degree and network flow.

We note that while ConceptNet has been proven useful, most works only use direct edges, with one work (4) being a notable exception. For other types of knowledge graphs, it was already shown that incorporating longer relational paths can be highly beneficial, and there is no reason to assume that this situation would be any different for ConceptNet. For instance, while there is an intuitively obvious semantic relationship between the words "beach" and "sun", there is no direct edge connecting these words in ConceptNet. However, this relationship can be uncovered by observing that ConceptNet contains the path "Beach $\xleftrightarrow{\text{RelatedTo}}$ Sunbathing $\xleftrightarrow{\text{RelatedTo}}$ Sun". Nevertheless, to enable more effective use of such relational paths from ConceptNet, we believe that more work is needed on how to avoid nonsensical paths, which is the focus of this paper.

3 METHOD

3.1 Problem Formulation

Our goal in this work is to train a classifier for predicting path quality based on crowdsourced information. Specifically, we ask human annotators to assess the *naturalness* of ConceptNet paths. Rather than trying to provide guidelines on how naturalness should be understood, we simply rely on annotators' intuitive understanding of this notion. This has several advantages, including the fact that we do not steer annotators towards particular indicators and the fact that this makes the annotation task much easier. Moreover, given that we are interested in a task-independent form of path quality (i.e., our focus is on eliminating non-sensical and uninformative paths, rather than on selecting the best paths for a particular application), it is not clear what further guidance would be meaningful for annotators. As we will discuss in more detail in Section 4.2, despite the lack of guidance on how naturalness should be understood, we observed a high inter-annotator agreement in practice.

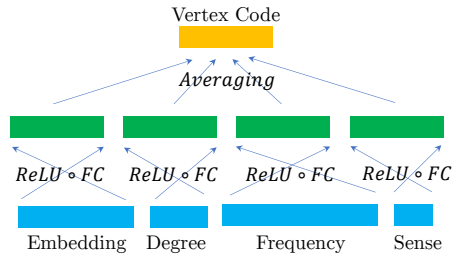
Clearly, however, asking participants to provide ratings on an absolute scale is problematic: people have varying thresholds for naturalness, and these thresholds may also change over time. After observing a large number of unnatural paths, people may lower their thresholds to be more willing to accept a path as natural, and vice versa. Therefore, we instead asked participants about pairwise comparisons: determining which of two given paths is more natural. In this setting, if one path is more natural than the other, two raters would provide the same answer even if they maintained different absolute naturalness thresholds, or if they shifted their underlying absolute threshold unconsciously over time.

If the two paths are equally (un)natural, the selected answer would be more or less arbitrary, and even the same annotator might not consistently give the same answer when presented with the same pair more than once. Therefore, in our model, we assume that answers to these pairwise comparison questions are probabilistically generated from a latent naturalness score. Specifically, each path is assumed to have a latent naturalness score m, such that for a pair paths with scores m_1 and m_2, the observed answer is drawn from a Bernoulli distribution, with probability $e^{m_1}/(e^{m_1} + e^{m_2})$ that the first path is selected. The problem we consider is to learn a model that can predict the naturalness score m of a given path, using crowdsourced answers of pairwise comparisons as supervision signal.

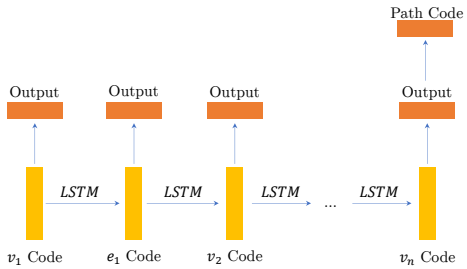
3.2 Model

In the big picture, our model consists of an encoder and a pairwise predictor. The encoder is a long short-term memory (LSTM) network (13) that transforms a path into a code vector. The predictor then takes the path codes and computes the probability that one path is more natural than the other. Each path is represented by an alternating sequence of vertex and edge representations, $(v_1, e_1, v_2,$

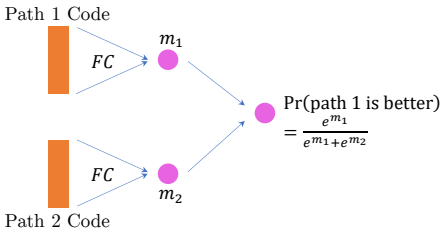
$e_2, ..., e_{n-1}, v_n$). Each vertex representation is a list of n_v features, $v_i = (\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, ..., \mathbf{v}_{i,n_v})$, with j-th feature being a d_j^v-dimensional vector. The edge representations are structured similarly. The features which we use are summarized in Section 3.4.



(a) The encoding of a vertex.



(b) The encoding of the path.



(c) The predictor model.

Figure 1: LSTM model architecture. Blue (bottom) cells represent raw features (of variable lengths). Green (middle) cells represent transformed features (of length l_f). Orange (top) cells represent the final code for the vertex (of length l_f). The encoding of an edge is computed in a similar fashion. The codes (yellow) for vertices and edges are successively processed by an LSTM network. The last state $\mathbf{h}_{2n-1}$ (orange) is used as the code for the entire path. The path codes (orange) first pass separately through fully-connected layers (with shared weight) with an output dimension of 1. A softmax layer then calculates the probability that path 1 is more natural than path 2.

A standard LSTM architecture can only process sequential data of the same dimension. Thus, vertex and edge representations must first pass through an encoder. Figure 1a depicts how a neural network encodes a vertex into a vertex code of pre-defined length l_f. Each feature (such as the word embedding and absolute frequency in some large corpus) is transformed through a rectified linear unit (ReLU)-activated fully-connected (FC) layer to a vector of length l_f. The overall code is the average of the vectors. The encoding of an edge to the same length l_f is conducted in a similar fashion. Figure 1b depicts the way in which the path code is calculated. After the codes for each vertex and edge are computed, they are sequentially aggregated by an LSTM network. The last state $\mathbf{h}_{2n-1}$ is interpreted as the code of the path. As shown in Figure 1c, the predictor then outputs the pairwise comparison result. It first transforms each path code to a score (m_1 and m_2) using linear layers with shared weight. The softmax function is then applied to the two scores to calculate a probability. While the network predicts pairwise comparisons, it is the score m_i produced by the network – or, at least, the ordering induced by the scores – that matters for applications.

The problem we consider falls under the broad umbrella of learning to rank, for which many algorithms have already been proposed. We use an LSTM encoder because it elegantly deals with variable length input, while most other methods can only take fixed length input representations.

3.3 Training

We used the negative-log-likelihood (NLL) loss as our objective function, along with the Adam optimization algorithm (14). Although the neural network model seems like a natural choice for the data generation process described in Section 3.1, here we show that the predicted scores m indeed converge to the latent scores from which the pairwise comparison results were generated, up to a constant difference.

Consider two paths with scores m_1 and m_2. According to our generative model, the first path is selected with probability $p_{m_1 m_2} = e^{m_1}/(e^{m_1}+e^{m_2})$. Note that from $e^{m_1}/(e^{m_1}+e^{m_2}) = e^{m_1'}/(e^{m_1'}+e^{m_2'})$ it follows that $m_1 - m_1' = m_2 - m_2'$ by simple algebra. This means that if our system can correctly predict the probabilities $p_{m_1 m_2}$, for any pair of paths, then it must be the case that there is a fixed constant c such that for each path with naturalness score m, the predicted naturalness score m' is such that $m = m' + c$.

It remains to be shown that the NLL objective will indeed lead our neural network model to predict the correct probabilities. Specifically, assume that observations $\{y_i \sim \text{Bernoulli}(q_\theta(\mathbf{x}_i))\}_{i=1}^n$ are given, where $\mathbf{x}_i$ is the feature representation of a given pair of paths and $q_\theta(\mathbf{x}_i)$ is the corresponding output of our neural network model when the parameters are set as θ. We show that minimizing the NLL objective will result in parameters ϕ such that $q_\theta = q_\phi$. For any fixed $\mathbf{x}_i$, given a set of observations $\{(\mathbf{x}_i, y_{i,1}), ..., (\mathbf{x}_i, y_{i,n_i})\}$, we define the expected data likelihood with respect to q_i to be as follows:

$$\mathbb{E}_{q_i}\left[\Pr(y_{i,1:n_i}|\mathbf{x}_i)\right] = \mathbb{E}\left[\prod_{j=1}^{n_i}\left[y_{i,j}q_i + (1-y_{i,j})(1-q_i)\right]\right].$$

When $y_{i,1:n_i}$ is generated by Bernoulli $(q_\theta(\mathbf{x}_i))$, it can be shown that $\arg\max_{q_i} \mathbb{E}_{q_i}\left[\Pr(y_{i,1:n_i}|\mathbf{x})\right] = q_\theta(\mathbf{x}_i)$. For the entire dataset $\{(\mathbf{x}_1, y_1), .., (\mathbf{x}_n, y_n)\}$, since the y_i's are independent of each other

given $\mathbf{x}_i$, the expected joint likelihood for the entire dataset

$$\mathbb{E}_{q_\phi}\left[\Pr(y_{1:n}|\mathbf{x}_{1:n})\right] = \mathbb{E}\left[\prod_{j=1}^{n}\left[y_j q_\phi(\mathbf{x}_j) + (1-y_j)(1-q_\phi(\mathbf{x}_j))\right]\right]$$

is then maximized for $q_\phi(\mathbf{x}) = q_\theta(\mathbf{x})$ for all $\mathbf{x}$. Therefore, we can simply minimize NLL (i.e. $-\log\left[\Pr(y_{1:n}|\mathbf{x}_{1:n})\right]$).

Note in particular that the network will converge to optimal predictions even with only one answer for each pair, which we can exploit to maximize the diversity of the crowdsourced annotations: by only collecting a single annotation for each pair of paths, a broader range of paths can be assessed (with a fixed budget) than with multiple annotations per pair. While each individual answer will to some extent be arbitrary (e.g., the fact that path 1 was selected could mean that path 1 is more natural or that both paths are approximately equally natural and that path 1 was selected by chance), because the network is trained on many different pairs, it will still learn to differentiate between clear-cut cases and borderline cases.

3.4 Features

In this section, we describe the features we used to encode vertices and edges. The number in parenthesis after each feature name indicates the dimension of that feature.

Vertex embedding (300) This feature is taken directly from the 300-dimensional GloVe (24) embedding, pre-trained on the Common Crawl[2] dataset with 840 billion tokens. In some experiments, we used principal component analysis (PCA) to first reduce the dimensionality of this feature before inputting it into the neural network.

Vertex frequency (1) This feature is a scalar representing the frequency of the given word. We estimated the (unnormalized) frequency using Zipf's law (35) from word occurrence ranks, which can be derived from the pre-trained GloVe embedding since it orders words by frequency. For example, the word "science" is ranked 717th, and is thus assigned a frequency of 1/717=1.39e-3.

Vertex degree (1) This feature is the number of neighbors (in both directions) of the vertex in the graph, representing how well connected the word is within ConceptNet.

Vertex sense score (1) This feature is a number between 0 and 1 representing how consistent is the overall meaning of the vertex compared to its neighbors on the left and right. For example, the sense score for the word "book" in the path "Knowledge $\xleftarrow{\text{HasA}}$ Book $\xleftrightarrow{\text{RelatedTo}}$ Restaurant" would be quite low because it is used in two different senses . On the other hand, the sense score for "book" in "Knowledge $\xleftarrow{\text{HasA}}$ Book $\xleftrightarrow{\text{RelatedTo}}$ Paper" would be high. We believe that a lower sense score would result in the path being considered less natural. In calculating this feature, we borrow and extend the idea of "word sense profile" proposed by (7). The details of the sense score calculation can be found in the appendix of the extended version[3].

Edge ends similarity (1) For an edge connecting two vertices (s, t), this feature is the cosine similarity of the embeddings of s and t.

Edge direction (3) This feature is a one-hot vector that represents if the edge is forward ("Dog $\xrightarrow{\text{IsA}}$ Animal"), backward ("Wheel $\xleftarrow{\text{HasA}}$ Car"), or bidirectional ("Big $\xleftrightarrow{\text{Antonym}}$ Small").

Edge relation (46) This feature is a one-hot vector that represents the relation type of the edge.

Edge provenance (6) In ConceptNet, each edge is derived from at least one source. In the case of multiple sources, the weight is the sum of the weights across all sources. However, weights across sources are not directly comparable; for example, nearly all edges from WordNet have a weight of 2.0, while those from Wiktionary have a weight of 1.0. This does not necessarily mean that WordNet is more reliable than Wiktionary. Thus, we encode the provenance of an edge using a 6-dimensional vector, with one component for each possible source: WordNet, DBpedia, Verbosity, Wiktionary, OpenCyc, and Open Mind Common Sense (26). When an edge has only one source, its component in the vector has a value equal to the weight, and all other entries are 0. When an edge has multiple associated sources, we split the weight across the sources using the most common weight of each source (e.g., 2.0 for WordNet).

Edge sense score (1) Similar to the vertex sense score, this feature is also a number between 0 and 1 characterizing the consistency of meaning, but at an edge level. Again, the details are included in the extended version[3].

4 EXPERIMENTS

In this section, we first describe our data collection procedure. Then we present an evaluation of the collected data and learned model.[4]

4.1 Data Collection

For our experiments, we used Amazon Mechanical Turk to collect pairwise human rating data. Each questionnaire consisted of 73 pairs, including 60 genuine pairs and 13 quality-control pairs. The quality-control pairs consisted of an obviously good path and an obviously bad path. The good paths were manually specified and meant to be very natural and straightforward (e.g., Email $\xrightarrow{\text{UsedFor}}$ Communication $\xleftarrow{\text{UsedFor}}$ Telephone). To generate bad paths, words and relations were sampled independently at random (e.g., "Beautiful $\xrightarrow{\text{IsA}}$ Other $\xleftrightarrow{\text{RelatedTo}}$ Them $\xleftarrow{\text{MotivatedBy}}$ Rug"). We only used answers from questionnaires for which all 13 quality-control questions were answered correctly. The genuine pairs consisted of randomly sampled paths from ConceptNet. We used a number of different strategies to sample these paths, as outlined below. Overall, around 1,500 valid responses were collected, giving us around 90,000 pairs of paths to use in different settings of the experiment.

4.2 Inter-Annotator Agreement

The human annotators were provided with a generic question: "which of the two paths is more natural?" Different participants may interpret this question slightly differently, focusing on different aspects of naturalness. Thus, one important question is how much humans agree with each other about naturalness. Establishing the inter-annotator agreement level is also useful because it serves as an intrinsic, model-agnostic performance upper bound.

[2] http://commoncrawl.org/

[3] https://arxiv.org/pdf/1902.07831.pdf

[4] The code and data for the experiments can be found at https://github.com/YilunZhou/path-naturalness-prediction

Our default data collection strategy is to obtain only one answer for each pair of paths. This strategy maximizes the coverage of our dataset, while still allowing the model to make probabilistic predictions, as discussed in Section 3.3. However, to evaluate the inter-annotator agreement, we also collected a multi-response dataset, with 59 questions each answered by 16 different participants.

Table 1 summarizes the results of an analysis of this dataset. The first column lists the different possible opinion splits for a given question; for example, 13/3 corresponds to a question for which 13 people preferred one of the paths, and 3 people preferred the other. The second column shows how many of the 59 questions have a given opinion split.

The last two columns of Table 1 refer to results obtained by our model trained in the 1st setting described below in Section 4.3. The third column shows the number of questions predicted correctly, i.e. the number of questions for which the prediction of our model coincides with the majority opinion of the human participants. The last column is the model's average confidence in the majority consensus. The 8/8 row does not include prediction statistics because there is no majority consensus for these questions.

A number of conclusions can be drawn from these results. First, given the disagreement levels shown in the second column, the theoretically best performing model, which always predicts the majority preference, would achieve 70.1% accuracy on this dataset. This shows that while people do not agree with each other on all of the questions, for the majority of pairs, human annotators do have a clear preference. Note that we should not expect a perfect agreement, since in some cases both of the paths may be equally natural or equally unnatural. This is illustrated in Table 2, which shows examples of pairs with different opinion splits. As can be seen from these examples, the fact that there is no majority for a given pair does not necessarily mean that the paths involved are of low quality. It simply means that the two paths are approximately equally natural.

As can be seen from the third column in Table 1, when there is a clear human consensus about which of the two paths is most natural, our model can predict this with a very high accuracy; e.g. for all questions where the opinion split was 14/2 or better, the majority view was predicted correctly. Furthermore, there is a strong positive correlation between the confidence scores predicted by our model, shown in the last column, and the amount of disagreement among human annotators. Our model is thus able to distinguish between cases where the difference in naturalness is clear-cut and cases where human annotators would be undecided.

Opinion Split	# Q	# Correct	Avg Conf
8/8	7	NA	NA
9/7	11	5	52.2%
10/6	6	5	60.1%
11/5	8	7	64.4%
12/4	11	7	56.6%
13/3	5	4	70.0%
14/2	5	5	72.5%
15/1	4	4	76.2%
16/0	2	2	89.8%

Table 1: Results for the multi-response dataset.

OS	Paths (The one on top is favored by the majority)
8/8	Forest $\xleftarrow{\text{AtLocation}}$ Country $\xleftrightarrow{\text{RelatedTo}}$ Geography $\xleftrightarrow{\text{RelatedTo}}$ Land Kingdom $\xleftrightarrow{\text{Synonym}}$ Land $\xleftarrow{\text{MadeOf}}$ Mountain $\xleftrightarrow{\text{RelatedTo}}$ Hill
	Dream $\xleftrightarrow{\text{RelatedTo}}$ State $\xleftarrow{\text{AtLocation}}$ City $\xleftrightarrow{\text{RelatedTo}}$ Population Desk $\xleftrightarrow{\text{RelatedTo}}$ Office $\xleftrightarrow{\text{RelatedTo}}$ Area $\xleftrightarrow{\text{RelatedTo}}$ Science
11/5	Blood $\xrightarrow{\text{AtLocation}}$ Person $\xleftrightarrow{\text{RelatedTo}}$ Home $\xleftrightarrow{\text{Antonym}}$ Street Range $\xleftrightarrow{\text{RelatedTo}}$ Food $\xrightarrow{\text{AtLocation}}$ Home $\xleftrightarrow{\text{Antonym}}$ Office
12/4	Kingdom $\xleftrightarrow{\text{RelatedTo}}$ Country $\xleftarrow{\text{AtLocation}}$ City $\xleftarrow{\text{AtLocation}}$ Office Sun $\xleftrightarrow{\text{RelatedTo}}$ Source $\xleftrightarrow{\text{RelatedTo}}$ Blood $\xleftrightarrow{\text{RelatedTo}}$ Tissue
13/3	Person $\xleftrightarrow{\text{RelatedTo}}$ Woman $\xleftrightarrow{\text{DistinctFrom}}$ Men $\xleftrightarrow{\text{RelatedTo}}$ People Type $\xleftrightarrow{\text{RelatedTo}}$ Unit $\xleftrightarrow{\text{RelatedTo}}$ Card $\xleftrightarrow{\text{RelatedTo}}$ Holiday
16/0	School $\xrightarrow{\text{AtLocation}}$ City $\xleftarrow{\text{IsA}}$ Town $\xrightarrow{\text{AtLocation}}$ Country Point $\xleftrightarrow{\text{RelatedTo}}$ Mountain $\xleftrightarrow{\text{RelatedTo}}$ Wave $\xrightarrow{\text{IsA}}$ Woman
	People $\xleftrightarrow{\text{RelatedTo}}$ Life $\xleftarrow{\text{PartOf}}$ Fun $\xleftarrow{\text{IsA}}$ Soccer Plane $\xleftrightarrow{\text{Antonym}}$ Point $\xleftrightarrow{\text{RelatedTo}}$ Hand $\xleftrightarrow{\text{RelatedTo}}$ Instrument

Table 2: Pairs of paths with different opinion splits (OS). In each cell the most favored path is shown on top.

4.3 Prediction Accuracy

4.3.1 Experimental Setup. In this set of experiments we studied how well our model can predict human judgements in three settings.

- In the 1st setting, we first determined a set of 100 words. To this end, starting from the center word "science", we performed a random walk on ConceptNet, considering only elementary-school-level nouns[5], until 100 different words had been sampled. We then sampled pairs of paths with 4 or less nodes in the subgraph induced by these 100 words. This dataset included about 10,000 paths. The training set contains 40,000 pairs sampled from 8,000 paths, and the test set contains 1,000 pairs put together from the remaining 2,000 paths. This evaluation allows us to assess the performance of our system in a domain-specific setting, since the model is evaluated on paths involving the same words as those in the training paths (but nonetheless different paths). This performance is recorded in Table 3.
- In the 2nd setting, we collected another set of 100 words using the same method as above, but using "money" as the center word, while additionally ensuring that there is no overlap between this set of 100 words and those from the "science" dataset. We then collected 1,000 paths among those 100 words, and put them into 500 pairs, which we use to evaluate models trained on the "science" dataset. In this way,

[5]http://www.k12reader.com/

we can assess the transfer learning performance of our model. This performance is recorded in Table 4.

- In the 3rd setting, we selected the nouns, verbs and adjectives from the 5,000 most frequent English words in the Corpus of Contemporary American English[6], for a total of 3,887 words. We then sampled 80,000 paths of up to 5 nodes, which we split into 20,000 training pairs and 20,000 testing pairs (ensuring that there are no overlapping paths between training and testing sets). Compared with the 1st setting, this represents much wider coverage of the set of words (i.e., embedding space), but much sparser coverage of paths, allowing us to assess the performance of our model in an open-domain setting. This performance is recorded in Table 5.

To put our performance into context, we also considered the following four heuristic baselines, all of which have similar mechanics: computing a score for each path (as does our model), and then selecting the path with the higher score.

Source-Target Baseline (ST-B) scores paths using the cosine similarity between the embeddings of the source and target words (note that the two paths in a pairwise comparison do not typically start and end with the same words);

Pairwise Baseline (Pair-B) scores paths using the average cosine similarity of all word pairs connected with an edge in the path;

Resource Flow Baseline (Flow-B) scores paths using the "path reliability" method proposed by Lin et al. (19) with the idea that a path is better if there are less branches out of the path;

Length Baseline (Path-B) scores paths solely by their length, and favors shorter paths to longer ones.

We performed a grid search of two hyper-parameters: vertex embedding dimension in $\{2, 10, 50, 100, 300\}$ and path code length in $\{1, 2, 5, 10, 20\}$. For the first hyper-parameter, we used principal component analysis (PCA) for dimensionality reduction. We also tested one-hot encodings of words for the 1st setting. For the other settings, this one-hot encoding cannot be used, because it cannot generalize to unseen words (2nd setting) or scale to a large number of words (3rd setting). In addition, Length-B performance is only available for the 3rd setting, because for the first two settings we did not explicitly control for path length. As a result, the vast majority of paths are of the cutoff length of 4, and path length is not predictive.

4.3.2 Quantitative Analysis. For the first setting (Table 3), we can see that performance was relatively insensitive to embedding dimension and code length, as long as both were sufficiently high. However, performance on the 2nd setting (Table 4) dropped significantly with higher embedding dimensions, suggesting an overfitting problem, with an embedding dimension of 2 achieving best generalization performance. This suggests that a model that was trained on one domain can indeed successfully be applied to another domain, as long as the embedding dimension is small enough to allow for sufficient generalization. In the 3rd setting (Table 5), we found that a higher embedding dimension is necessary, most likely because the range of meanings is more diverse for this dataset, which includes not only nouns, but also verbs and adjectives.

[6]https://www.wordfrequency.info/

		Path Code Length				
		1	2	5	10	20
Emb. Dim.	2	62.1	63.1	65.8	65.3	65.6
	10	63.6	65.5	65.6	67.5	66.4
	50	63.7	65.5	66.9	67.4	67.7
	100	65.4	65.9	66.1	**68.2**	67.6
	300	66.2	66.6	67.6	67.6	67.7
One-Hot		67.0	66.4	67.0	67.5	67.9
ST-B		53.4				
Pair-B		58.0				
Flow-B		51.6				

Table 3: Test accuracy (in percentage) in the domain-specific setting of "science" related words.

		Path Code Length				
		1	2	5	10	20
Emb. Dim.	2	60.5	64.6	63.9	65.1	**65.5**
	10	60.1	61.7	64.8	64.9	64.8
	50	62.5	62.8	63.1	63.3	64.0
	100	60.0	61.8	63.1	64.2	64.8
	300	58.8	60.5	61.1	62.0	62.1
ST-B		53.9				
Pair-B		55.9				
Flow-B		52.5				

Table 4: Test accuracy (in percentage) in the transfer learning setting, where the model was trained on the "science" dataset and evaluated on the "money" dataset.

		Path Code Length				
		1	2	5	10	20
Emb. Dim.	2	61.4	60.5	63.1	62.9	63.3
	10	61.2	62.6	65.0	64.9	64.2
	50	61.6	64.5	64.8	65.4	65.8
	100	62.0	64.3	65.0	**67.9**	65.1
	300	61.9	63.0	65.7	65.4	65.4
ST-B		52.3				
Pair-B		57.5				
Flow-B		46.8				
Length-B		55.8				

Table 5: Test accuracy (in percentage) in the open-domain setting of the most frequent English words.

For all three settings, considering the level of agreement found in Section 4.2, we found that the performance of our model approaches the expected performance upper bound. Moreover, our

model performs substantially and consistently better than all of the baselines. In fact, in Table 5 we can see that only one of the baselines is able to outperform the simple path length heuristic (Length-B), and only in a minimal way.

4.3.3 Qualitative Analysis. To qualitatively compare how our model prediction differs from the baselines, Table 6 presents the best paths between the words "health" and "computer" (of up to 4 nodes) selected by our method and the two strongest baselines.

We can see that the top paths found by the pairwise baseline lack variety, most likely due to the high embedding similarity between "health" and "care". In addition, the length baseline does not perform satisfactorily because the shortest paths are not easily understandable without more explanation, with nearly all relations being the generic "$\xleftrightarrow{\text{RelatedTo}}$". By contrast, our model (trained on the 3rd open-domain setting) selects paths that overcome both drawbacks. It covers a wider variety of concepts such as "virus" and "expensive" and favors longer paths with smoother transition of node semantics.

In another evaluation, we inspect paths for which our model and the pairwise baseline significantly disagree. We randomly sampled 120,000 paths from the entire ConceptNet graph and use the two methods to predict their quality. Table 7 lists paths that are predicted to be among the top 10% most natural by our model but are in the bottom 10% according to the pairwise baseline. Table 8 conversely shows paths that are among the 10% least natural according to our model but the 10% most natural according to the baselines. Paths in both tables were selected at random.

As we can see, the paths in Table 7 are intuitively quite meaningful, with all paths using some rather uncommon but relevant words, which is perhaps clearest in the last example (Hechsher refers to a certification given to a food product that complies with Jewish religious law and Kashrut refers to the set of Jewish dietary laws). By comparison, the paths in Table 8 mostly use very common but uninformative words. This analysis suggests that the pairwise baseline suffers from the so-called hubness problem, i.e. the problem that in high-dimensional vector space embeddings, there are typically a few central objects which are highly similar to many of the other objects (25), an issue which is known to affect word embeddings (9). The pairwise baseline favors paths with very common words, even if they are not semantically related, because these words act as hubs in the word embedding.

4.4 Ablation Study on Feature Importance

To discern the contribution of each feature to the prediction performance, we trained variants of our model with subsets of the full feature set. For this analysis, we used the 3rd evaluation setting with the best performing hyper-parameters (embedding dimension of 100 and path code length of 10). The results are summarized in Table 9.

We can see that leaving out the edge ends similarity feature hurts performance most. This is also in accordance with the fact that pairwise baseline in Table 5, which only uses this feature, performs best among the four baseline methods. Furthermore, the second-highest reduction is found when leaving out the vertex embedding feature. Finally, we see that neither vertex features nor edge features alone can achieve optimal prediction performance.

4.5 Naturalness as a Path Selection Criterion

Next we discuss three experiments which are aimed at assessing how well naturalness indicates path quality. We start with an intrinsic evaluation of the semantic coherence of natural paths, after which we discuss two extrinsic tasks: information retrieval and analogy inference.

4.5.1 Semantic Coherence. Let us define the *type* of a path as the sequence of relations appearing in that path, e.g. the type of the path "A $\xrightarrow{\text{IsA}}$ B $\xrightarrow{\text{HasA}}$ C" is ($\xrightarrow{\text{IsA}}$, $\xrightarrow{\text{HasA}}$). In this experiment, we consider pairs of nodes in ConceptNet that are connected with a relational path of a given type, as well as a one-step relation (e.g., "A $\xrightarrow{\text{HasA}}$ C"). In this situation, we call the latter relation a *path-summarizing (PS) relation* for the considered path type. Many path types uniquely determine the PS relation, or at least narrow down the set of candidate PS relations to a small number. For instance, for a path of type ($\xrightarrow{\text{IsA}}$, $\xrightarrow{\text{HasA}}$) we would expect to only see $\xrightarrow{\text{HasA}}$ as the PS relation. However, due to the presence of noise in ConceptNet, some relations of this type will actually have a different PS relation. Motivated by this view, as an intrinsic evaluation task, we propose to assess the extent to which a path selection method is able to select *semantically coherent* paths in terms of the variability of the PS relations among the selected paths.

In particular, for a set of paths, we first determine a mapping from path types to counters of PS relations: $\{p_1 \rightarrow [(r_{11}, c_{11}), (r_{12}, c_{12}), ..., (r_{1n}, c_{1n})], ..., p_m \rightarrow [(r_{m1}, c_{m1}), (r_{m2}, c_{m2}), ..., (r_{mn}, c_{mn})]\}$, in which each p_i is a path type, each r_{ij} is one of 46 relation types from ConceptNet, and each c_{ij} is the number of times the relation r_{ij} was found as a PS relation for path type p_i. The average entropy of the PS relations is then defined as:

$$\frac{-\sum_{i=1}^{m}\left[C_i \cdot \sum_{j=1}^{n} p_{ij}\ln(p_{ij})\right]}{C},$$

where $C_i = \sum_{j=1}^{n} c_{ij}, C = \sum_{i=1}^{m} C_i, p_{ij} = c_{ij}/C_i$. A higher average entropy suggest a higher proportion of spurious PS relations and thus less semantically coherent paths.

Using the 3,887 words from the 3rd setting, we generated 126,600 length-4 paths whose source and target nodes are also directly connected by a relation (i.e. the PS relation). Our model was applied to rank the paths by naturalness. Based on this ranking, we calculate the average entropy of top N% natural paths, where N varies from 10 to 100, and plot the result in Figure 2. We compared our model to Pair-B, the best-performing baseline for predicting naturalness. The entropy for a random shuffle of the dataset is also depicted for comparison. We see that using our model to select the most natural paths consistently leads to the lowest entropy. Note that all three methods will converge to the same entropy at the 100% mark, at which point all paths are used, regardless of the path selection method.

4.5.2 Information Retrieval. When using a search engine, users often provide under-specified queries, e.g. because they assume that the search engine is "smart" enough to infer their true intention or because they do not actually have a clear idea on how to formulate a better query. Moreover, even if a query is unambiguous, it may use different terms than those appearing in the most relevant documents. To deal with such issues, many information retrieval

Pairwise	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xrightarrow{\text{IsA}}$ Work $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xleftrightarrow{\text{RelatedTo}}$ Help $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xleftrightarrow{\text{RelatedTo}}$ Do $\xleftarrow{\text{CapableOf}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xleftarrow{\text{Desires}}$ Person $\xrightarrow{\text{Desires}}$ Computer
Length	Health $\xleftrightarrow{\text{RelatedTo}}$ System $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Level $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Apple $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Well $\xleftrightarrow{\text{RelatedTo}}$ Screw $\xrightarrow{\text{AtLocation}}$ Computer
Our Model	Health $\xleftarrow{\text{Antonym}}$ Disease $\xleftarrow{\text{Causes}}$ Virus $\xrightarrow{\text{AtLocation}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xrightarrow{\text{IsA}}$ Work $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Insurance $\xrightarrow{\text{HasProperty}}$ Expensive $\xleftarrow{\text{HasProperty}}$ Computer
	Health $\rightarrow$ Sickness $\rightarrow$ Virus $\xrightarrow{\text{AtLocation}}$ Computer

Table 6: Best paths of 4 or less nodes between "health" and "computer" ranked by baselines and our model.

Lead $\xrightarrow{\text{HasProperty}}$ Toxic $\xleftrightarrow{\text{RelatedTo}}$ Toxicant $\xleftrightarrow{\text{Synonym}}$ Poison
Wave $\xrightarrow{\text{IsA}}$ Fluctuation $\xleftrightarrow{\text{RelatedTo}}$ Brainwave $\xrightarrow{\text{DerivedFrom}}$ Brain
Space $\xleftarrow{\text{MadeOf}}$ Passageway $\xrightarrow{\text{AtLocation}}$ Building $\xleftrightarrow{\text{RelatedTo}}$ Station
Food $\xleftrightarrow{\text{RelatedTo}}$ Hechsher $\xleftrightarrow{\text{RelatedTo}}$ Kashrut $\xleftrightarrow{\text{RelatedTo}}$ Law

Table 7: Paths predicted to be among 10% most natural by our model, but 10% least natural by the pairwise baseline.

Bee $\xleftrightarrow{\text{RelatedTo}}$ A $\xleftrightarrow{\text{RelatedTo}}$ After $\xleftrightarrow{\text{RelatedTo}}$ Attack
Space $\xleftrightarrow{\text{RelatedTo}}$ Very $\xleftrightarrow{\text{RelatedTo}}$ Little $\xleftrightarrow{\text{RelatedTo}}$ Moon
Lead $\xleftrightarrow{\text{Synonym}}$ Run $\xleftrightarrow{\text{RelatedTo}}$ Very $\xleftrightarrow{\text{RelatedTo}}$ Poison
Adult $\xleftrightarrow{\text{RelatedTo}}$ A $\xleftrightarrow{\text{RelatedTo}}$ The $\xleftrightarrow{\text{RelatedTo}}$ English

Table 8: Paths predicted to be among 10% most natural by the pairwise baseline, but 10% least natural by our model.

All Features	67.9%
No vertex embedding	65.4%
No vertex frequency	67.3%
No vertex degree	67.1%
No edge ends similarity	63.5%
No edge direction	67.8%
No edge relation	67.3%
No edge provenance	67.2%
No vertex and edge sense scores	66.9%
Vertex features only	61.6%
Edge features only	63.7%

Table 9: Ablation study on feature importance.

systems rely on some form of query expansion (34), a strategy for automatically adding semantically related keywords to the user's query. In this experiment, we study the performance of a query expansion method based on natural ConceptNet paths.

We use the TREC 2004 Robust Retrieval Track dataset (30), commonly known as ROBUST04, consisting of 250 queries and around 260,000 documents. Each query from this dataset typically consists of 2 or 3 words, and is associated on average with 1,000 annotated documents (marked as either positive or negative). We use a simple TF-IDF retrieval model, in which the TF-IDF weighting scheme is used to vectorize both the documents and the queries and the cosine similarity is used as the ranking metric. Following the work by Kotov and Zhai (15), we focus on hard queries, which are defined as those with Precision at 10 (P@10) performance of 0 when only using the original query terms. We then evaluate how the performance can be improved by expanding the query with terms that are selected using a given strategy. There are 44 examples of such hard queries among the original 250 queries.

To expand queries based on ConceptNet, we first find all paths, consisting of up to 4 nodes, which connect any two of the query terms. For example, if the query is "drug crime organization", we find paths between the pairs (drug, crime), (drug, organization), and (crime, organization). We then add the words from the most natural paths to the original query. In particular, we add a fixed number of words by processing the paths in the order defined by the chosen path ranking method. To this end, we have considered the following strategies for ranking paths:

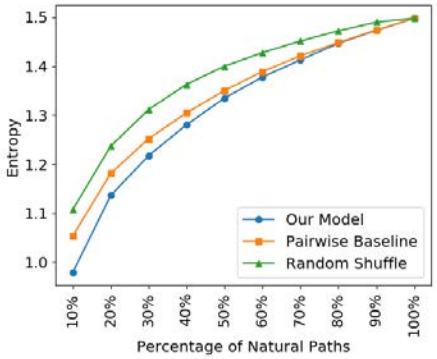


Figure 2: Average entropy of path-summarizing relations for different proportions of natural paths.

(1) naturalness score, which is calculated from our model trained on the open-domain (3rd) setting;
(2) pairwise score, which is the pairwise baseline;
(3) length score, which is the length baseline (i.e. shorter paths are better), with random tie-breaking;
(4) random score, which assigns a random score to each path;
(5) naturalness+length score, which first favors shorter paths, but uses our naturalness score to rank paths with same length.

To put our results into context, we also compared this method with a query expansion strategy based on the GloVe word embedding. Specifically, in this case we add the words that are closest to the query words, ordered by their cosine similarity in the vector space. Note that each of the considered strategies will yield an ordered list of possible words to add to the query. We study the retrieval performance of using different cut-offs in this list.

The P@10 and MAP performances are shown in Figure 3. We can see that for both evaluation metrics, the naturalness+length score achieves the highest performance, suggesting that while nodes in shorter paths are generally more relevant to original queries, considerable gains can be made by using naturalness to rank paths of the same length. The word embedding model also performs well, but it cannot reach the best overall performance of the two naturalness based methods.

4.5.3 Analogy Inference. Boteanu and Chernova (4) used relation paths to solve analogy questions of the following form:

```
Question: Dog:Animal::
       A: Table:Chair
       B: Apple:Fruit
       C: Fast:Slow
       D: Ice:Cold
```

The goal is to select the word pair which has the same relation as the query pair (e.g. the correct answer in the question above is *B* because dog is a type of animal, and apple is a type of fruit). In their proposed method, the quality of the analogy $a : b :: c : d$ (meaning a is to b as c is to d) is calculated using the proportion of overlapping relations on paths connecting (a, b) and (c, d) respectively. Boteanu

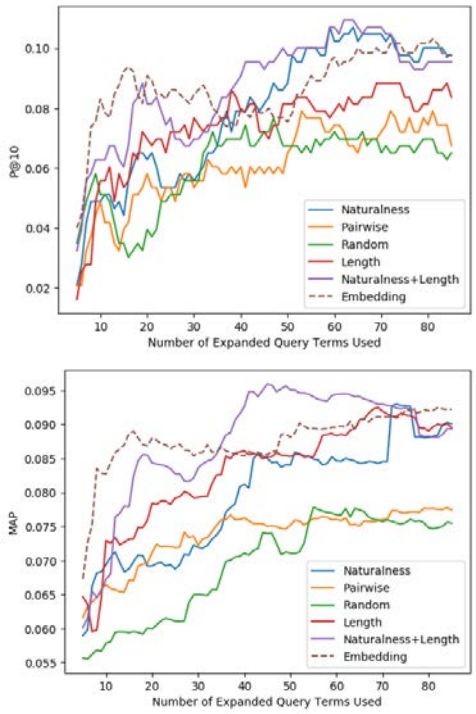


Figure 3: Performance of query expansion for hard queries in terms of Precision at 10 (P@10, top) and Mean Average Precision (MAP, bottom).

and Chernova (4) considered paths of two (i.e. direct connection) and three (i.e. with an intermediate hop) nodes.

We found that when direct connections exist between words in the query pair, using only these direct connection (and not including 3-node paths) achieves the best performance. However, not all query pairs have direct connections, in which case we are required to use 3-node paths. For our dataset of 85 questions, 74 have associated direct edge connections. Thus, we study the effect of only considering the most natural paths, rather than all of them, on the remaining 11 questions. Figure 4 shows the performance of the prediction as a function of the proportion of 3-node paths used. Using our model to filter 3-node paths, the best performance is obtained when only the 30% most natural paths are considered. As a comparison, the pairwise baseline is not able to outperform a random selection.

Across all three evaluations, we consistently find that including all (or too many) paths leads to sub-optimal performance, due to the inclusion of noisy paths, and that our proposed method outperforms existing heuristics based on word embedding and path length.

5 CONCLUSIONS

Our work is motivated by the observation that commonsense knowledge graphs such as ConceptNet are noisy and informal, and that many of the relational paths in such knowledge graphs are therefore non-sensical. This means that the only way in which applications can successfully make use of relational paths from such resources is by relying on a method to filter out these non-sensical paths. While

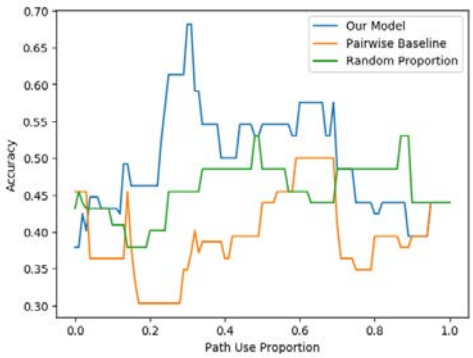


Figure 4: Performance on the 11 analogy questions for which direct connections are not available, using various proportions of natural paths.

this problem has already been studied by others, previous solutions relied on relatively simple heuristics to select higher-quality paths, which we found to perform poorly in practice. In fact, in the extrinsic evaluation tasks, we found that some heuristics were not even able to outperform random selection (Figures 3 and 4).

Rather than engineering more sophisticated heuristics, the solution we proposed in this paper is to take a data-driven approach and learn the concept of path *naturalness* based on crowdsourced judgments. The main insights we obtained are as follows:

(1) although we intentionally left the interpretation of naturalness open to crowdsourcers, they do largely share a common perception of this concept;
(2) this concept can be effectively learned by an LSTM model, with accuracy close to an empirical upper bound derived from the inter-annotator agreement level; and
(3) our learned model outperforms previously proposed heuristics as a path selection method in several intrinsic and extrinsic evaluation tasks.

ACKNOWLEDGMENT

Steven Schockaert was supported by ERC Starting Grant 637277.

REFERENCES

[1] Lora Aroyo and Chris Welty. 2014. The Three Sides of CrowdTruth. *Human Computation* 1 (2014), 31–34.
[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. *The Semantic Web* (2007), 722–735.
[3] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. ACM, 1247–1250.
[4] Adrian Boteanu and Sonia Chernova. 2015. Solving and Explaining Analogy Questions Using Semantic Networks. In *AAAI Conference on Artificial Intelligence*. AAAI, 1460–1466.
[5] Zied Bouraoui, Shoaib Jameel, and Steven Schockaert. 2018. Relation Induction in Word Embeddings Revisited. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*. 1627–1637.
[6] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an Architecture for Never-Ending Language Learning. In *AAAI Conference on Artificial Intelligence*. 1306–1313.
[7] Junpeng Chen and Juan Liu. 2011. Combining ConceptNet and WordNet for Word Sense Disambiguation. In *International Joint Conference on Natural Language Processing (IJCNLP)*. 686–694.
[8] Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017. Chains of Reasoning over Entities, Relations, and Text Using Recurrent Neural Networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Vol. 1. 132–141.
[9] Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. 2015. Improving Zero-Shot Learning by Mitigating the Hubness Problem. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
[10] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting Word Vectors to Semantic Lexicons. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL:HLT)*. Association for Computational Linguistics, 1606–1615.
[11] Matt Gardner, Partha Talukdar, Jayant Krishnamurthy, and Tom Mitchell. 2014. Incorporating Vector Space Similarity in Random Walk Inference over Knowledge Bases. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 397–406.
[12] Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing Knowledge Graphs in Vector Space. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.
[14] Diederik Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the International Conference Learning Representations (ICLR)*.
[15] Alexander Kotov and Chengiang Zhai. 2012. Tapping into Knowledge Base for Concept Feedback: Leveraging ConceptNet to Improve Search Results for Difficult Queries. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 403–412.
[16] Alicia Krebs, Alessandro Lenci, and Denis Paperno. 2018. SemEval-2018 Task 10: Capturing Discriminative Attributes. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval)*. 732–740.
[17] Ni Lao and William W Cohen. 2010. Relational Retrieval Using a Combination of Path-Constrained Random Walks. *Machine Learning* 81, 1 (2010), 53–67.
[18] Douglas B Lenat. 1995. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM (CACM)* 38, 11 (1995), 33–38.
[19] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling Relation Paths for Representation Learning of Knowledge Bases. In *Proceedings of the Conference Empirical Methods in Natural Language Processing (EMNLP)*. 705–714.
[20] Hugo Liu and Push Singh. 2004. ConceptNet—a Practical Commonsense Reasoning Tool-Kit. *BT Technology Journal* 22, 4 (2004), 211–226.
[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Advances in Neural Information Processing Systems (NIPS)*. 3111–3119.
[22] George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. 1990. Introduction to WordNet: An On-line Lexical Database. *International Journal Lexicography* 3, 4 (1990), 235–244.
[23] Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional Vector Space Models for Knowledge Base Completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL & IJCNLP)*. Association for Computational Linguistics, 156–166.
[24] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.
[25] Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. 2010. Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data. *Journal of Machine Learning Research (JMLR)* 11 (2010), 2487–2531.
[26] Push Singh, Thomas Lin, Erik T Mueller, Grace Lim, Travell Perkins, and Wan Li Zhu. 2002. Open Mind Common Sense: Knowledge Acquisition from the General Public. In *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*. Springer, 1223–1237.
[27] Robert Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *AAAI Conference on Artificial Intelligence*. AAAI, 4444–4451.
[28] Kristina Toutanova, Victoria Lin, Wen-tau Yih, Hoifung Poon, and Chris Quirk. 2016. Compositional Learning of Embeddings for Relation Paths in Knowledge Base and Text. In *Procedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, Vol. 1. 1434–1444.
[29] Luis Von Ahn, Mihir Kedia, and Manuel Blum. 2006. Verbosity: A Game for Collecting Common-Sense Facts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. ACM, 75–78.
[30] Ellen M Voorhees. 2004. Overview of the TREC 2004 Robust Retrieval Track. In *TREC*.
[31] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Communications of the ACM (CACM)* 57, 10 (2014), 78–85.
[32] Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *Proceedings*

of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP). 564–573.

[33] Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 2014. RC-NET: A General Framework for Incorporating Knowledge into Word Representations. In *Procedings of the 23rd ACM International Conference Information and Knowledge Management (CIKM)*. ACM, 1219–1228.

[34] Jinxi Xu and W. Bruce Croft. 1996. Query Expansion Using Local and Global Document Analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, 4–11.

[35] George Kingsley Zipf. 1935. The Psycho-Biology of Language. (1935).

A Hierarchical Attention Retrieval Model for Healthcare Question Answering

Ming Zhu
Dept. of Computer Science
Virginia Tech, Arlington, VA
mingzhu@vt.edu

Aman Ahuja
Dept. of Computer Science
Virginia Tech, Arlington, VA
aahuja@vt.edu

Wei Wei
Google AI
Mountain View, CA
wewei@google.com

Chandan K. Reddy
Dept. of Computer Science
Virginia Tech, Arlington, VA
reddy@cs.vt.edu

ABSTRACT

The growth of the Web in recent years has resulted in the development of various online platforms that provide healthcare information services. These platforms contain an enormous amount of information, which could be beneficial for a large number of people. However, navigating through such knowledgebases to answer specific queries of healthcare consumers is a challenging task. A majority of such queries might be non-factoid in nature, and hence, traditional keyword-based retrieval models do not work well for such cases. Furthermore, in many scenarios, it might be desirable to get a short answer that sufficiently answers the query, instead of a long document with only a small amount of useful information. In this paper, we propose a neural network model for ranking documents for question answering in the healthcare domain. The proposed model uses a deep attention mechanism at word, sentence, and document levels, for efficient retrieval for both factoid and non-factoid queries, on documents of varied lengths. Specifically, the word-level cross-attention allows the model to identify words that might be most relevant for a query, and the hierarchical attention at sentence and document levels allows it to do effective retrieval on both long and short documents. We also construct a new large-scale healthcare question-answering dataset, which we use to evaluate our model. Experimental evaluation results against several state-of-the-art baselines show that our model outperforms the existing retrieval techniques.

CCS CONCEPTS

• **Information systems** → **Language models**; **Learning to rank**; **Question answering**.

KEYWORDS

Neural Networks, Information Retrieval, Consumer Healthcare, Question Answering

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313699

ACM Reference Format:
Ming Zhu, Aman Ahuja, Wei Wei, and Chandan K. Reddy. 2019. A Hierarchical Attention Retrieval Model for Healthcare Question Answering. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313699

1 INTRODUCTION

With the growth of the Web in recent years, a vast amount of health-related information is now publicly available on the Internet. Many people use online health information platforms such as WebMD[1] and Patient[2] to search for information regarding the symptoms, diseases, or any other health-related information they are interested in. In addition to consumers, often doctors and healthcare professionals need to look into knowledgebases that contain detailed healthcare information about diseases, diagnosis, and procedures [8, 31]. Despite the abundance of available information, it might be difficult for healthcare consumers to navigate through these documents to get the required healthcare information. Hence, effective retrieval techniques are required to allow consumers to efficiently use such platforms. Since healthcare documents usually include several details about the disease such as it's symptoms, preventive measures, and common treatments, they are usually more elaborate, compared to other factual documents, which describe well-known facts (e.g., population of a town, capital of a city, or any other entity), and are very specific in nature. Hence, in such cases, it might be desirable to provide the consumers with a short piece of text that succinctly answers their queries. Furthermore, many questions that users have about health-related topics are very abstract and open-ended in nature, and hence traditional search methods do not work well in such cases.

Prompted by the success of deep neural networks in language modeling, researchers have proposed several techniques that apply neural networks for effective information retrieval [9, 21] and question answering [33, 38]. This has been facilitated primarily due to the development of large training datasets such as TREC [35] and SQuAD [25]. However, both these datasets are primarily composed of factoid queries / questions, and the answers are generally short in length. Hence, systems trained on such datasets cannot perform

[1] https://www.webmd.com/
[2] https://patient.info/

What would happen if I didn't take antithyroid medicines?
It is usually advisable to treat an overactive thyroid gland (hyperthyroidism). Untreated hyperthyroidism can cause significant problems with your heart and other organs. It may also increase your risk of complications should you become pregnant. However, in many cases there are other treatment options. That is, radioactive iodine or surgery may be suitable options. See the separate leaflet called Overactive Thyroid Gland (Hyperthyroidism) for details of these other treatment options .

Figure 1: An example of a healthcare question, and it's corresponding answer. The question and answer do not have any overlapping words. The highlighted text corresponds to the most relevant answer snippet from the document.

well in a setting where a large proportion of the queries are non-factoid and open-ended, and the documents are relatively longer in length. Figure 1 shows an example of a typical question that a consumer would have regarding antithyroid medicines, and it's corresponding answer paragraph, selected from the website Patient. This problem and the domain provides some unique challenges which require us to build a more comprehensive retrieval system.

- **Minimal overlap between question and answer words:** There is minimal or no word overlap between the question and answer text. As there are no matching terms, traditional keyword-based search mechanisms will not work for answering such questions.
- **Length of question and answer:** The question is longer than a typical search engine query. The answer is also typically longer than a sentence. Although, for illustration purposes, we show a short paragraph, in many cases, the answer, as well as the document containing it, might be even longer. Hence, neural semantic matching algorithms will not be effective in such cases, as they are ideally designed for short sentences. Therefore, an effective retrieval system would require a mechanism to deal with documents of varied lengths.
- **Non-factoid nature:** The question is very open-ended in nature, and does not ask for any specific factual details. As such, a majority of the machine comprehension models are trained on datasets like SQuAD, which are comprised of factoid QA pairs. Such systems do not work well in a setting where the desired answer is more elaborate.

To overcome these problems, we propose **HAR**, a **H**ierarchical **A**ttention **R**etrieval model for retrieving documents for healthcare related queries. The proposed model uses a cross-attention mechanism between the query and document words to discover the most important words that are required to sufficiently answer the query. It then uses a hierarchical inner attention, first over different words in a sentence, and then over different sentences in a document, to successively select the document features that might be most relevant for answering the query. Finally, it computes a similarity score of a document with the query, that could be used to rank different documents in the corpus, given a query. The use of hierarchical attention also enables it to find the most important sentences and words, that could be important to answer a query, without the need of using an explicit machine comprehension module. To evaluate the performance of our model, we construct a large scale healthcare question answering dataset, using knowledge articles collected from the popular health services website Patient. Although we use this model in the healthcare domain, where the questions are usually non-factoid in nature, and the documents are longer due to the presence of detailed description about different medical procedures, our model is more generic, and can be used in any domain where the questions are open-ended, and the documents are longer.

The rest of this paper is organized as follows: Section 2 gives an overview of the existing techniques related to our work. In Section 3, we describe our proposed neural retrieval model called HAR, and provide the details about its architecture and the training procedure, including the optimization for the HAR model. The details about the data collection and annotation have been described in Section 4. In Section 5, we give details about our experimental evaluation, and the metrics and baseline techniques used in the evaluation process. Finally, Section 6 concludes the paper, with possible directions for future research.

2 RELATED WORK

2.1 Document Ranking

Document retrieval and ranking is a classical problem in the information retrieval community, which has attracted significant interest from researchers for many years. Early methods in informational retrieval were largely based on keyword-based query-document matching [27, 29, 30]. With the advancement of machine learning algorithms, better retrieval mechanisms have been proposed. Logistic Inference [7] used logistic regression probabilities to determine the relevance between queries and documents. In [14], the authors used Support Vector Machine (SVM) based approach for retrieval, which allows the retrieval system to be trained using the search engine click logs. Other traditional techniques in information retrieval include boosting-based methods [6, 41]. TF-IDF based similarity [26] and Okapi BM25 [28] are the most popularly used term-based techniques for document search and ranking. However, such techniques usually do not perform well, when the documents are longer [18], or have minimal exact word overlap with the query.

2.2 Neural Information Retrieval

With the success of deep neural networks in learning feature representation of text data, several neural ranking architectures have been proposed for text document search. Deep Structured Semantic Model (DSSM) [13] uses a simple feed-forward network to learn the semantic representation of queries and documents. It then computes the similarity between their semantic representations using cosine similarity. Convolutional Deep Structured Semantic Model (CDSSM) [34] uses convolutional layers on word trigram features, while the model proposed in [22] uses the last state outputs of LSTM encoders as the query and document features. Both these models then use cosine similarity between query and document representations, to compute their relevance. In [12], the authors

propose convolutional neural network models for semantic matching of documents. The Architecture-I (ARC-I) model proposed in this work also uses a convolutional architecture to create document-level representation of query and document, and then uses a feed-forward network to compute their relevance. The InferSent Ranker [11] proposed recently also uses a feed forward network to compute the relevance between query and documents, by summing up their sentence embeddings. However, all these methods use the document-level semantic representation of queries and documents, which is basically a pooled representation of the words in the document. However, in majority of the cases in document retrieval, it is observed that the relevant text for a query is very short piece of text from the document. Hence, matching the pooled representation of the entire document with that of the query does not give very good results, as the representation also contains features from other irrelevant parts of the document.

To overcome the problems of document-level semantic-matching based IR models, several interaction-based IR models have been proposed recently. In [9], the authors propose Deep Relevance Matching Model (DRMM), that uses word count based interaction features between query and document words, while the Architecture-II (ARC-II) proposed in [12] uses convolution operation to compute the interaction features. These features are then fed to a deep feed-forward network for computing the relevance score. The models proposed in [4, 40] use kernel pooling on interaction features to compute similarity scores, while MatchPyramid [23] uses the dot product between query and document word vectors as their interaction features, followed by convolutional layers to compute the relevance score. Other methods that use word-level interaction features are attention-based Neural Matching Model (aNMM) [42], that uses attention over word embeddings, and [36], that uses cosine or bilinear operation over Bi-LSTM features, to compute the interaction features. The Duet model proposed in [21] combines both word-level interaction features, as well as document-level semantic features, in a deep CNN architecture, to compute the relevance. One common limitation of all these models is that they do not utilize the inherent paragraph and sentence level hierarchy in documents, and hence, they do not perform well in case of longer documents. By using a powerful cross attention mechanism between query and document words, our model can effectively determine the most relevant document words for a query. It then uses hierarchical inner attention over these features, which enables it to effectively deal with long documents. This is especially helpful in cases where the relevant information in the document is a very small piece of text.

2.3 Information Retrieval for Healthcare

Early works in the domain of information retrieval for medicine and healthcare used tradition search methods such as TF-IDF [19] and BM25 [17]. MedQA [45] uses hierarchical clustering along with TF-IDF for answering definitional questions asked by physicians. Th question-answering system proposed in [5] aimed at helping clinicians to search for treatments for any disease. However, their system is tailored to answer one specific type of questions, and cannot be used to answer open-ended questions. As discussed in [15], physicians also often need to use such systems, and they have limited time to browse through every returned document. Hence, medical retrieval needs to be accurate, and should precisely serve the requirements of the users. Retrieval in this domain is complex, attributed to the non-factoid nature of queries, and longer documents. Hence, traditional IR techniques or semantic matching algorithms do not work well on such datasets.

3 THE PROPOSED MODEL

In this section, we introduce our proposed Hierarchical Attention Retrieval (HAR) model, which uses deep attention mechanism for effective retrieval. The detailed architecture of our model is shown in Fig. 2. HAR is a novel neural network model that uses two powerful attention mechanisms to overcome the shortcomings of existing document retrieval models. Given a query q, the model computes a relevance score r_i with each candidate document d_i in the document knowledgebase $\mathcal{D}$. The different components of our model are described in detail below.

3.1 Word Embeddings

The input layer in our model is an embedding lookup function which converts the query and document words into fixed K-dimensional word vectors using a lookup matrix $\boldsymbol{E} \in \mathbb{R}^{V \times K}$ of V pre-trained word embeddings such as GloVe [24] or Word2Vec [20]. Let $\{w_t^q\}_{t=1}^m$ be the words in q. Let l be the number of sentences in document d, and $\{w_t^{id}\}_{t=1}^n$ be the words in sentence i in d. This layer converts each of the words in q and d into the word vectors $\{e_t^q\}_{t=1}^m$ and $\{e_t^{id}\}_{t=1}^n$, respectively. Here, m and n are the number of words in query and each of the document sentences, respectively.

3.2 Encoder

We use two bidirectional RNN (Bi-RNN) [32] encoders to encode the inter-document temporal dependencies within query and documents words, respectively. This layer consists of two RNN layers in different directions, whose output is concatenated to get the H-dimensional contextual representation of each word. We choose GRU [2] over vanilla-RNN or LSTM [10] because of its high performance and computational efficiency. Since we split the documents into short sentences, GRU performs equally well as LSTM, because the encoder does not need to deal with very long sequences.

3.2.1 Query Encoder. The query encoder contains a simple Bi-GRU layer, which takes the query word embeddings $\{e_t^q\}_{t=1}^m$ as the input, and outputs the contextual representation $U^q = \{u_t^q\}_{t=1}^m \in \mathbb{R}^{m \times H}$.

$$u_t^q = BiGRU_Q(u_{t-1}^q, e_t^q) \tag{1}$$

3.2.2 Document Encoder. Since documents are usually longer than queries, we encode each sentence in the document separately, using a sentence-level Bi-GRU encoder. Given a sentence i, this layer takes the sentence word embeddings $\{e_t^{id}\}_{t=1}^n$ as the input, and returns the contextual word embeddings $U^{id} = \{u_t^{id}\}_{t=1}^n \in \mathbb{R}^{n \times H}$. After encoding each of the l sentences in the document through this encoder, the new document representation is $\{U^{1d}, .., U^{ld}\} \in \mathbb{R}^{l \times n \times H}$.

$$u_t^{id} = BiGRU_D(u_{t-1}^{id}, e_t^{id}) \tag{2}$$

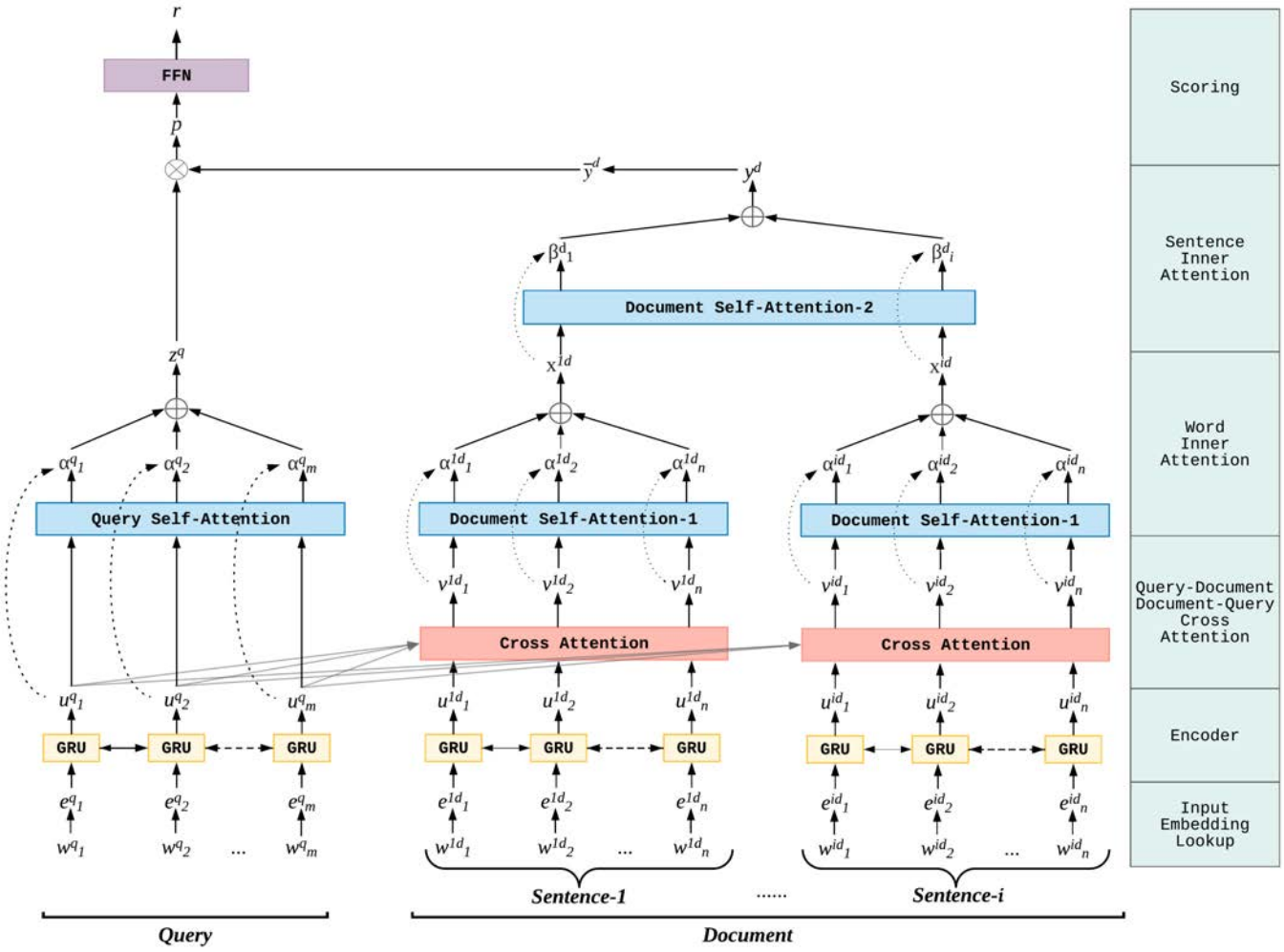


Figure 2: Architecture of the proposed HAR model.

3.3 Cross Attention between Query and Document

This layer is used to fuse the information from query words to the document words. It computes the relevance of each query word with respect to each word in the document. As we use a hierarchical modeling for documents, this layer can compute the attended embeddings of each word in sentence i in d with each word in the query q. We use the cross attention mechanism proposed in [39, 44], as this method has been proven to show superior performance in state-of-the-art reading comprehension systems. The attention layer computes the relevance between each pair of query and document words, using their contextual embeddings generated by the respective encoders. To calculate the relevance of query words with respect to document words, and vice-versa, we use bi-directional attention mechanism [33], which is composed of document-to-query attention $D2Q$ and query-to-document attention $Q2D$. This is done by first computing a similarity matrix $S \in \mathbb{R}^{n \times m}$, which is then normalized over each row and column, using a normalization operation such as softmax. This generates normalized similarity matrices $\overline{S}_{D2Q} \in \mathbb{R}^{n \times m}$ and $\overline{S}_{Q2D} \in \mathbb{R}^{n \times m}$, respectively. Finally, the attention matrices $A_{D2Q} \in \mathbb{R}^{n \times H}$ and $A_{Q2D} \in \mathbb{R}^{n \times H}$ can be computed as described below.

Let $s_{xy} \in \mathbb{R}$ be an element of the similarity matrix S from row x and column y. Given $U^{id} \in \{U^{1d}, .., U^{ld}\}$ and U^q as inputs, the final output $V^{id} = \{v_t^{id}\}_{t=1}^n \in \{V^{1d}, .., V^{ld}\}$ of the cross attention layer can be computed as follows:

$$s_{xy} = w_c^T \cdot [u_x^{id}; u_y^q; u_x^{id} \odot u_y^q] \tag{3}$$

$$\overline{S}_{D2Q} = \text{softmax}_{\text{row}}(S) \tag{4}$$

$$\overline{S}_{Q2D} = \text{softmax}_{\text{col}}(S) \tag{5}$$

$$A_{D2Q} = \overline{S}_{D2Q} \cdot U^q \tag{6}$$

$$A_{Q2D} = \overline{S}_{D2Q} \cdot \overline{S}_{Q2D}{}^T \cdot U^{id} \tag{7}$$

$$V^{id} = [U^{id}; A_{D2Q}; U^{id} \odot A_{D2Q}; U^{id} \odot A_{Q2D}] \in \mathbb{R}^{n \times 4H} \tag{8}$$

In the above equations, ; is the concatenation operation, $\odot$ is element-wise multiplication, $\cdot$ is matrix multiplication, and $w_c \in \mathbb{R}^{3H}$ is a trainable weight vector.

3.4 Query Inner Attention

To encode variable length queries into a fixed size embedding, we use the self attention mechanism proposed in [16]. The importance of different words varies from document to document, and is dependent on the context in which they are used. This layer allows the model to give higher priority to more important words while creating a pooled representation of the query. This ensures that the query representation contains features from more significant words. Let A be the dimension of the pooled representation. Given query features $U^q = \{u_t^q\}_{t=1}^m$ as the input, this layer generates a pooled representation of $z^q \in \mathbb{R}^H$ as follows:

$$c_t^q = w_q^T(\tanh(W_q u_t^q)) \tag{9}$$

$$\alpha_t^q = \frac{\exp(c_t^q)}{\sum_{j=1}^m \exp(c_j^q)} \tag{10}$$

$$z^q = \sum_{t=1}^m \alpha_t^q u_t^q \tag{11}$$

3.5 Document Hierarchical Inner Attention

Since documents are longer in length, it is not necessary that the entire document is relevant to a query. In fact, in most cases, it is observed that part of the document that is relevant to a query is just a few sentences. Even inside each sentence, different words might have varying relevance to the query. Furthermore, because of the varied lengths of documents, a mechanism is required to get a fixed-dimensional representation of the document. Hence, we use a two-level hierarchical inner attention (as proposed in [43]), to get the document embedding.

3.5.1 Level-1: Attention over words in a sentence. The first level in our hierarchical attention encodes each sentence independently from other sentences at word-level, resulting in a fixed-dimensional representation of each sentence. This layer computes the importance of each word within the sentence, and then creates a pooled representation of each sentence weighted by the attention weights. For each sentence i in the document d, this layer takes the output vectors $V^{id} = \{v_t^{id}\}_{t=1}^n \in \mathbb{R}^{n \times 4H}$ from the cross-attention layer as the input, and returns a sentence vector $x^{id} \in \mathbb{R}^{4H}$.

$$c_t^{id} = w_{d1}^T(\tanh(W_{d1} v_t^{id})) \tag{12}$$

$$\alpha_t^{id} = \frac{\exp(c_t^{id})}{\sum_{j=1}^n \exp(c_j^{id})} \tag{13}$$

$$x^{id} = \sum_{t=1}^n \alpha_t^{id} v_t^{id} \tag{14}$$

3.5.2 Level-2: Attention over sentences in a document. To ensure that the sentences more relevant to the query are given higher importance while computing the similarity score, we use a second inner attention, to compute the document representation. This layer takes the sentence embeddings $\{x_i^d\}_{i=1}^l$ as the input, and returns a document vector $y^d \in \mathbb{R}^{4H}$ as the output.

$$b_i^d = w_{d2}^T(\tanh(W_{d2} x^{id})) \tag{15}$$

$$\beta_i^d = \frac{\exp(b_i^d)}{\sum_{j=1}^l \exp(b_j^d)} \tag{16}$$

$$y^d = \sum_{j=1}^l \beta_j^d x^{jd} \tag{17}$$

3.6 Score Computation

The final layer in our model computes the score between the query representation z^q and document representation y^d. Since the dimension of y^d is 4 times the dimension of z^q, we first pass y^d through a feed-forward layer to compute $\overline{y}^d \in \mathbb{R}^H$. After this, we compute the similarity vector $p \in \mathbb{R}^H$ by performing element-wise multiplication of z^q and $\overline{y}^d$. Finally, we pass p through a feed-forward network to compute the final relevance score $r \in \mathbb{R}$.

$$\overline{y}^d = w_{d3}^T y^d + b_{d3} \tag{18}$$

$$p = z^q \odot \overline{y}^d \tag{19}$$

$$r = w_f^T p + b_f \tag{20}$$

3.7 Optimization

3.7.1 Negative sampling. Many retrieval datasets, such as the ones created using user click-logs, only have the query-document pairs, which serve as the positive data for the model. However, for the model to have sufficient discriminative power to give a score to every document proportional to their relevance with the query, the model also needs negative query-document pairs during the training process. Hence, we do negative sampling to generate negative data samples of query-document pairs for our model. For each query, the negative samples are composed of the following:

- *Irrelevant negative samples:* For the model to have a sufficient discriminative power that is needed to distinguish documents at a high level, we sample negative documents that have very low relevance to the query.
- *Partially relevant negative samples:* We define partially relevant negative documents as those that might have some relevance to the query, either due to some overlapping words, or because they are from the same topic, but do not contain the correct answer for the query. As suggested in [37], having such samples in the training dataset gives a higher discriminative power to the model,

as compared to a model trained with randomly sampled negative pairs.

3.7.2 Loss function. We use pairwise maximum margin loss [14] as the objective function to be minimized. Given a query q, positive document d^{pos}, and k negatively sampled documents $\{d_1^{neg}, .., d_k^{neg}\}$, the loss is given by:

$$\mathcal{L} = \sum_{i=1}^{k} max(0, M - score(q, d^{pos}) + score(q, d_i^{neg})) \quad (21)$$

Here, M is the margin, by which we want the score of positive query-document pair to exceed that of a negative query-document pair.

4 HEALTHQA DATASET

We will now introduce the dataset created in this work to train and evaluate the proposed HAR model. We call this dataset *HealthQA*. It consists of question and document pairs from the healthcare domain. The details of this dataset are described below:

4.1 Knowledge Articles

To create HealthQA dataset, we collected healthcare articles from the popular health-services website Patient. We scraped all the articles from the *Health Topics* section of Patient. The website contains articles from a diverse set of healthcare domains such as child health, mental health, sexual health, details about treatments and medications, and several other healthcare domains. The articles on this website are much more detailed, as compared to other healthcare knowledgebases like MedlinePlus[3]. In total, we collected 1,235 health articles, with each article having an average of 6 sections. As the sections themselves are very long in these articles, we use each section as one document.

Figure 3 shows the distribution of percentage of documents with respect to the length of the documents (number of words). We can see that the proportion of documents with less than 50 words is very less (5%). The dataset contains a large number of documents with 100-200 words, and a high proportion of documents containing more than 200 words.

4.2 Question-Answer Pair Generation

To create healthcare-related questions, we employed human workers from diverse age groups, and from different countries. For the dataset to have a diverse set of questions and answers that different people might have about healthcare, we hired six annotators, consisting of a combination of freelancers, graduate and undergraduate students. To ensure high quality of the dataset, and low error rates, we ensured that all the annotators had good English skills. For each document, workers were instructed to create 1 to 3 questions that can be asked using the information given in the document. They were encouraged to use simple language in the queries, so that the questions follow the style of those asked by a common person, without any domain expertise in healthcare. All the generated questions also underwent an additional round of cross validation by the authors, and any query-document pairs with errors or insufficient context were either corrected or discarded.

[3]https://medlineplus.gov/

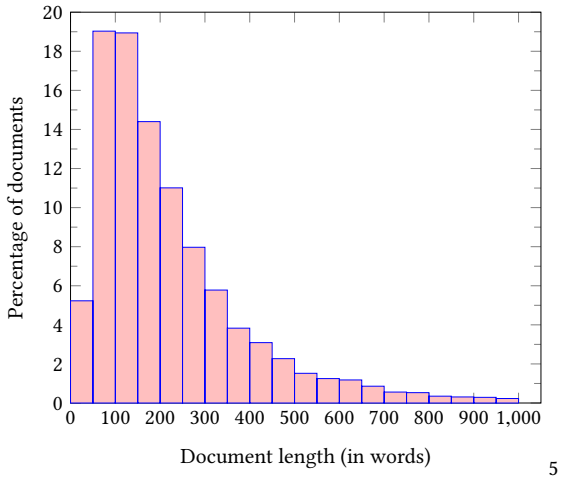


Figure 3: Percentage of documents vs. number of words.

It was found that many articles on Patient have several subtitles, roughly one subtitle per paragraph, and in most cases, these subtitles could be rephrased into valid questions. Some of the titles have incomplete context, which can be made into a valid question by rephrasing them. Hence, workers were also allowed to use the subtitles as questions, by rephrasing them into valid questions.

Figure 4 shows the percentage of different types of questions in our dataset. The questions with the type “How" and “Why" are mostly non-factoid in nature. Such questions are open-ended, and require detailed answers. Although the questions with the type “What", that are generally factoid, have a large proportion in our dataset, after manual analysis, we found that a large proportion of such questions are also non-factoid. Examples of such questions include *“What is the outlook of gaming disorder?"*.

Table 1: Statistics of the HealthQA dataset.

Number of articles	1,235
Number of documents (article sections)	7,355
Number of questions	7,517
Average length of questions (in words)	8.04
Average length of documents (in words)	233.4
Average number of sentences in documents	13.54
Average length of sentences (in words)	17.24

5 EXPERIMENTAL RESULTS

5.1 Evaluation Metrics

We compare the performance of HAR with various baseline techniques using the following evaluation metrics:

- **Recall@K:** Recall@K for a query is defined as the ratio of the number of relevant documents in top-K retrieved documents, with respect to the total number of relevant documents for that query. This is averaged over all the queries in the dataset. Since, in our case, each query has only one relevant document, Recall@K

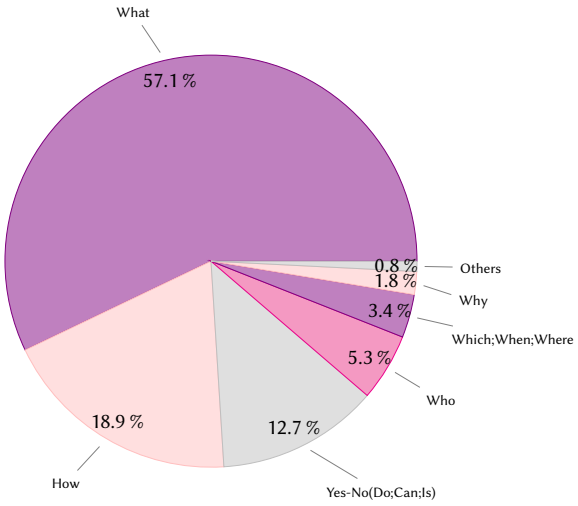


Figure 4: Percentage of questions by type.

denotes the percentage of queries whose correct document was present in the top-K retrieved documents.

$$Recall@K = \frac{1}{Q_{test}} \sum_{i=1}^{Q_{test}} \frac{\#\text{ relevant documents in top-K for query } i}{\#\text{ total relevant documents for query } i} \tag{22}$$

- **Mean Reciprocal Rank (MRR):** The reciprocal rank is defined as the inverse of the rank of the first correct document d^{pos} for a given query. MRR is the mean of the reciprocal rank for all the queries in the test set. Since we have only one correct document for every query, MRR is well suited for our evaluation. As compared to Recall@K, MRR also takes into account the ranking order in the evaluation.

$$MRR = \frac{1}{Q_{test}} \sum_{i=1}^{Q_{test}} \frac{1}{rank(d^{pos}) \text{ of query } i} \tag{23}$$

5.2 Baseline Methods

For performance comparison, we use the following state-of-the-art retrieval models as the baselines:

- **TF-IDF:** This is the standard baseline for retrieval tasks, that uses TF-IDF representation for both the query and document, and cosine similarity as the similarity function.
- **CDSSM [34]:** It is an extension of DSSM, that uses a CNN-based model on letter trigrams from query and document as input features, and then computes the cosine similarity to calculate the relevance between query and document representation.
- **ARC-II [12]:** This model first computes the interaction feature vector between query and document using CNN layers. It then computes the score for the query-document interaction vector using feed-forward network.
- **MV-LSTM [36]:** It is a neural semantic matching model that was proposed to find semantic similarity between a pair of sentences. The model uses the word embeddings obtained by passing the sentences through a Bi-LSTM, and then computes an interaction vector using cosine similarity, or a bilinear operation. It finally passes the interaction vector through a feed-forward network to compute the similarity score. In our implementation, we use cosine similarity to compute the interaction vector.
- **DRMM [9]:** It is a state-of-the-art neural ranking model that uses cosine similarity between query and document word vectors to compute their similarity, and then computes a histogram-like interaction vector by binning the cosine similarity scores into pre-defined intervals. It then passes these features through a feed-forward network to compute the scores.
- **KNRM [40]:** It is a recently proposed neural ranking model that first computes cosine similarity between each query word with each of the document words. It then performs kernel pooling, followed by a feed-forward network to compute the relevance score.
- **aNMM [42]:** This model first computes an interaction matrix by computing the cosine similarity between each of the query and document words. Similar to DRMM, this model also performs binning to compute a fixed-dimensional interaction vector. However, instead of using the counts of word-pairs that fall into a bin as its features, this model uses the total sum of the similarity between those word pairs as the bin features. It also uses an attention mechanism over the query word vectors, which is then combined with the interaction vector to compute the final relevance scores.
- **Duet [21]:** It is a recently proposed hybrid neural matching model that uses the word-level interaction, and document-level similarity, in a deep CNN architecture, to compute similarity between two documents.
- **MatchPyramid [23]:** This model computes pair-wise dot product between query and document word vectors to compute an interaction matrix. It then passes this matrix through CNN layers with dynamic pooling to compute the similarity score.

5.3 Implementation Details

We implemented our model in Keras [3], with TensorFlow [1] as the backend. The model was trained using Adadelta optimizer [46], with an initial learning rate of 2.0. We used one Bi-GRU layer (one forward and one backward), and each GRU layer had an output dimension of 150 units. The maximum number of words in query, and each document sentence was set to 15, and sentences greater than 15 words were split into multiple sentences. The maximum number of sentences in each document was set to 20. All the attention layers had a dimension of 300. We used a feed-forward network with 3 layers to compute the final score from the similarity vector. Additionally, we used dropout of 0.2 after each layer. For each query, we had one positive document, 3 partially relevant negative documents, and 6 irrelevant negative documents. For all the neural baselines, we used an open-source implementation MatchZoo[4]. The TF-IDF baseline was implemented using the `scikit-learn`[5] package.

We used GloVe [24] pre-trained word vectors with 300 dimensions. Since healthcare documents contain some medical words which cannot be found in GloVe, we used randomly initialized embeddings for such out-of-vocabulary words. We experimented with

[4]https://github.com/NTMC-Community/MatchZoo
[5]http://scikit-learn.org/stable/index.html

training GloVe word vectors on our document corpus, but it was found that the original pre-trained GloVe outperformed our medical word embeddings. We hypothesize that it was due to the fact that our corpus contained far lesser documents than those used in the original GloVe. Also, it is observed that people usually do not use very complex medical words in their queries, and hence, most of the queries were composed of words that were present in the pre-trained GloVe.

The dataset was split into three parts: train, validation, and test. The number of queries in each split were 5,274, 1,109, and 1,134 respectively.

5.4 Results

In this section, we present the results obtained in the experimental evaluation of the proposed HAR model.

5.4.1 Quantitative comparison against state-of-the-art models. Table 2 shows the performance of HAR against the baseline retrieval models on the HealthQA dataset. For computing Recall@K, we set K as 1, 3, and 5. As we can see, the results for HAR are consistently better than all other baselines, across all the metrics, which can be attributed to its strong performance.

- **Effect of long documents on model performance:** We can observe from Table 2 that TF-IDF has very low performance on our dataset. With the exception of ARC-II, the performance of TF-IDF is consistently lower than all the other models. This can be attributed to the non-factoid nature of our dataset, as well as the long length of documents in the corpus. Hence, a keyword-based method cannot perform well on such dataset, since its performance largely depends on the word overlap between the query and the document. In such cases, embedding-based methods yield better performance in general, as they can correlate queries and documents based on their semantic representation.
- **Document-representation based semantic similarity methods:** One key observation from our results is that neural models that match query with documents solely based on their document-level representation do not work well for retrieval problems. CDSSM computes the representation of query and document separately, and then computes the similarity between their vector representations. Although using semantic representations can help in dealing with the problems that arise where the queries do not have matching words with the documents, this concept only works in problems such as sentence or paraphrase matching, where the length of both the documents being matched is similar. In case of retrieval, queries are typically much shorter compared to the documents. Moreover, the actual part in the document that is relevant to the query is only a few words or sentences. Hence, the vector representation of the document contains features from other parts of the documents that are irrelevant to the query. This leads to the poor performance of such models for retrieval tasks.
- **Effect of word interactions:** With the exception of ARC-II, other baselines such as MV-LSTM, DRMM, KNRM, aNMM, Duet, and MatchPyramid use some form of embedding-based pair-wise keyword interaction in the feature representation of the query-document pair. This results in their better performance compared to other baselines. Using word-level interaction features based on their vector representation allows the models to deal with the problems faced by traditional word matching methods such as TF-IDF. However, by computing interaction in the early stages of the model, these models do not have any mechanism to incorporate the underlying structure of the query and document in the interaction feature generation or scoring process. This leads to their poor performance in a setting where the documents are longer in length.

By using a cross attention mechanism between the query and document, HAR is able to model the interaction features between the query and document words, while retaining the overall semantic meaning of the document sentences. The self attention mechanism then facilitates focussing on sentences and words in the document that are most relevant to the query. This mechanism helps HAR to achieve the highest performance compared to all other baseline methods. Most importantly, HAR achieves a considerably high MRR and Recall@1 against all other methods. This implies that, in most of the cases, our model is able to rank the correct document with the highest score, demonstrating a higher reliability of HAR.

In Figure 5, we show the performance of HAR and other baseline methods on each of the question types given in Figure 4. Although questions of the type *"what"* are relatively easier to answer (as they contain many factoid questions), our model outperforms other baselines on these questions. We can also see that HAR gives high performance on question types *"how"* and *"why"*, which are non-factoid in nature, and difficult for a retrieval system. The performance of HAR is consistently higher than the baselines across all other question categories as well.

5.4.2 Qualitative results. For qualitative evaluation of the performance of HAR, we show an example of a question, and the retrieved document, obtained by MatchPyramid and HAR, in Figure 6. We compare with MatchPyramid, since it is the strongest baseline among all the other methods. The question shown here is about *"the effect of wisdom tooth removal on brushing"*. Although the document returned by MatchPyramid is relevant to the topic, which is *wisdom teeth removal*, it is not the one that can correctly answer the query. MatchPyramid computes interaction features at an early stage in the model. It does not retain the original query and document, and computes scores solely based on the interaction features. Due to this, the main intent of the question can sometimes be lost, as shown in the example here. By using a powerful attention mechanism, HAR has the ability to discriminate between two similar documents, based on the intent of the query. Hence, HAR is able to retrieve the correct document for the question.

5.4.3 Using HAR for answer extraction. As mentioned earlier, an added advantage of using the hierarchical attention mechanism is that it allows the model to discover the most probable answer snippet from the long document. This can be done by comparing the attention weights of different sentences in level-1 of hierarchical inner attention over documents. Since sentences with high attention weights have more contribution towards generating the document representation, they are likely to be more relevant to the query, as compared to sentences with low attention weights.

In Figure 7, we illustrate how the attention weights can be used to extract the most probable answer from the document. We show

Table 2: Comparison of HAR model with other baseline models on HealthQA dataset.

Model name	MRR	Recall@1	Recall@3	Recall@5
TF-IDF	60.597	36.576	81.740	96.909
CDSSM	64.461	44.274	81.335	93.418
ARC-II	50.366	29.847	61.858	78.539
MV-LSTM	75.145	58.882	90.262	97.385
DRMM	74.082	57.078	90.081	99.008
KNRM	70.965	54.914	84.040	94.139
aNMM	74.717	58.251	90.352	98.287
Duet	69.659	53.291	84.310	93.868
MatchPyramid	81.816	69.432	93.688	98.918
HAR	**87.877**	**78.900**	**96.844**	**99.639**

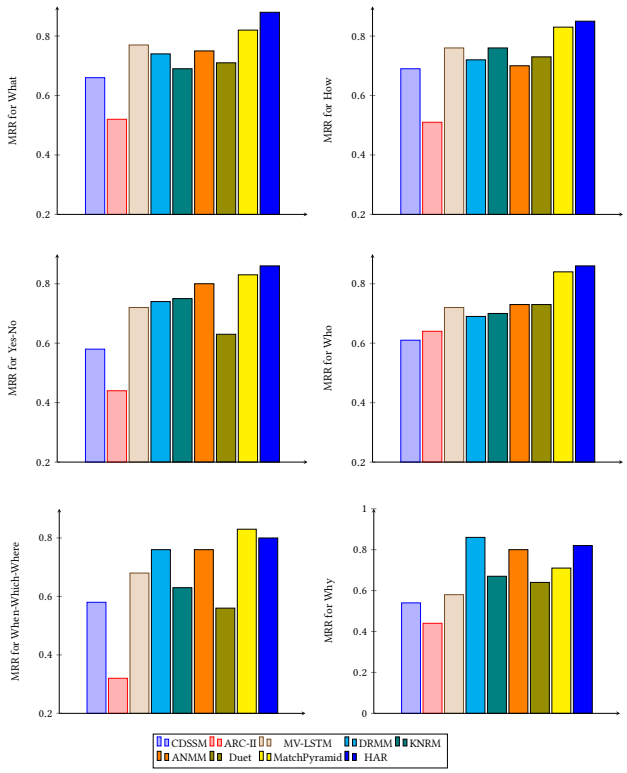


Figure 5: Performance of HAR and baseline methods on different types of questions.

one question, and it's corresponding highest-ranked document. The self-attention weights over the query words are highlighted in blue, while the self-attention weights at sentence level (level-1) of the document hierarchical attention are shown in red.

The question shown in Figure 7 is about *Alopecia Areata*, which is a condition that leads to hair loss in many people, mainly because the immune system of a person starts attacking the hair follicles. The question asks about the diagnosis procedure for this condition, demonstrated by the use of the word *test*. Since diagnosis is the main intent of this question, the feature representation uses highest

Will removal of wisdom teeth affect my brushing ?	
MatchPyramid answer (incorrect): Most people need two or three days off work to recover from the effects of the anaesthetic after they have had wisdom teeth removed. Those with a more physical job may need an extra day or two. If your job involves driving you should be fine to return after 24 - 48 hours after having had anaesthetic, but you need to use your own judgement and seek advice from your dentist if you are unsure.	**HAR answer (correct):** You may find it uncomfortable to brush shortly after having wisdom teeth removed, especially around any wounds. For this reason, your dentist might recommend that you use a mouthwash to help keep the area clean.

Figure 6: An example of a question and its answer retrieved by MatchPyramid (left) and HAR (right).

attention weight for the word *test*, followed by other words that are useful in the question. The document here has the highest attention weight for the first sentence, that contains the answer to this question. Other highlighted sentences in the document are those which can provide additional information supporting the answer for this question.

Do I need any tests, for Alopecia Areata ?

Usually not . the diagnosis is usually based on the typical appearance of the bald patches . If there is doubt about the cause of the hair loss , sometimes some blood tests or a skin scraping from a bald patch may be done to rule out other causes . A small sample of skin (skin biopsy) is sometimes taken to look at under the microscope . Occasionally you may need some tests to check for other autoimmune diseases . For example , if you have certain other symptoms , you might need to have blood tests to check your blood count and thyroid function . Usually blood tests come back entirely normal with alopecia areata .

Figure 7: An example of a question and its answer document with highlightings based on the attention weights. The attention weights for the question are obtained from the query self attention, and those for document are obtained from level-2 self attention.

5.5 Performance Analysis

5.5.1 Effect of attention mechanism. To quantitatively evaluate the effect of various components used in our HAR on the model performance, we compare the performance of HAR against its two variants. These are described below:

- **HAR without cross-attention:** To evaluate the effect of using cross-attention mechanism on model performance, we evaluate the performance of a variant of HAR that does not use cross-attention between query and document words. This model uses an inner attention over query words, and a hierarchical inner attention over document words and sentences. By removing the cross attention, the model is not able to use the interaction features between query and document word vectors in the scoring process. We refer to this model as *HAR-WCA*.
- **HAR without cross and hierarchical attention:** This is an even simpler version of HAR that neither uses cross-attention between query and document words, nor the hierarchical inner attention in the document. This model uses similar encoder as HAR for query and document, and then uses only one level of inner attention to get the query and document representations. It then computes the scores between these vectors similar to the scoring process used in HAR. It does not incorporate the underlying document structure in the scoring process due to the removal of hierarchical attention. We refer to this model as *HAR-simple*.

Model name	MRR	Recall@1	Recall@3	Recall@5
HAR-simple	82.139	69.883	94.770	99.008
HAR-WCA	83.667	72.047	95.942	99.369
HAR	**87.877**	**78.900**	**96.844**	**99.639**

Table 3: Performance comparison of HAR and its variants.

Table 3 shows the performance comparison of HAR with two of its variants. The performance of both *HAR-WCA* and *HAR-simple* is worse than the full model. We believe that since *HAR-simple* uses a single long encoder for documents, it is not able to embed the contextual dependencies in the encoded embeddings. Also, it does not use cross-attention, thereby ignoring the keyword-interaction features. The performance of *HAR-WCA* is slightly better than that of *HAR-simple*. Since each sentence sequence is much smaller than the full document, the encoded representation is able to embed the context of the sentence in each word.

5.5.2 Hyperparameter sensitivity and model convergence. As mentioned earlier, by splitting the documents into short sentences and using the hierarchical attention mechanism, HAR does not need to deal with long sequences. This allows the model to be achieve high performance using computationally efficient GRU, which can effectively model short sequences. We also find that by using GRU, the model is able to converge quickly, as compared to a model that uses LSTM. We also evaluated the performance of HAR by varying different parameters of our model, such as the number of GRU hidden units and the number of feed-forward layers. We find that these parameters only have a marginal impact (~1% MAP reduction) on the performance of our model. In Figure 8, we show the convergence of HAR and other baselines over training epochs. We show the MRR of different models on test queries, as the number of training epochs increase. We can see that HAR converges faster as compared to other baselines, demonstrating a better learning ability of our model.

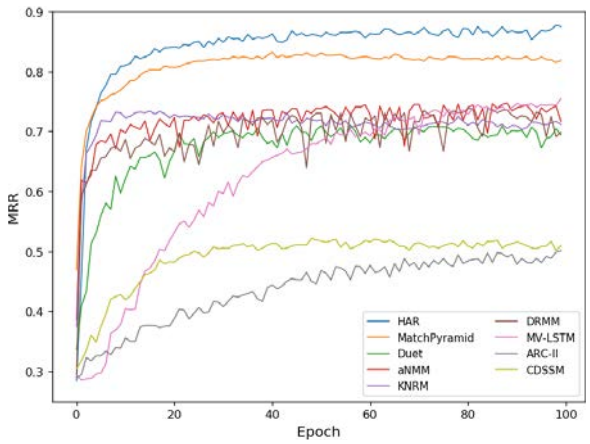


Figure 8: MRR convergence with training epochs.

6 CONCLUSION

In this paper, we proposed a novel deep neural network architecture to rank documents for healthcare related queries. The model uses a combination of powerful attention mechanisms to develop a robust retrieval system. This attention mechanism also enables the model to discover highly probable answer snippets from the documents, without the need for using a computationally expensive machine comprehension module. The model has been carefully designed by considering the special characteristics of question-answering in healthcare domain, such as the open-ended nature of queries, and longer document length.

To evaluate the proposed HAR model, we constructed a novel consumer-oriented healthcare question answering dataset, HealthQA. This dataset is comprised of questions that consumers typically ask about health-related topics. We evaluated our proposed model on this dataset, against several state-of-the-art baseline techniques. Our experimental results show that our model outperforms these techniques by a wide margin. We also show how our model can be used to extract the most probable answer snippets from the highly-ranked documents. We hope that our proposed model will be useful for both healthcare and information retrieval communities, to make healthcare information more accessible to the people.

ACKNOWLEDGEMENTS

This work was supported in part by the US National Science Foundation grants IIS-1619028, IIS-1707498, and IIS-1838730, and NVIDIA Corporation.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. TensorFlow: a system for large-scale machine learning. In *Proceedings of*

the 12th USENIX conference on Operating Systems Design and Implementation. USENIX Association, 265–283.

[2] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1724–1734.

[3] François Chollet et al. 2015. Keras.

[4] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 126–134.

[5] Dina Demner-Fushman and Jimmy Lin. 2006. Answer extraction, semantic clustering, and extractive summarization for clinical question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 841–848.

[6] Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *Journal of machine learning research* 4, Nov (2003), 933–969.

[7] Fredric C Gey. 1994. Inferring probability of relevance using the method of logistic regression. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*. Springer-Verlag New York, Inc., 222–231.

[8] Paul N Gorman, Joan Ash, and Leslie Wykoff. 1994. Can primary care physicians' questions be answered using the medical journal literature? *Bulletin of the Medical Library Association* 82, 2 (1994), 140.

[9] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 55–64.

[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[11] Phu Mon Htut, Samuel Bowman, and Kyunghyun Cho. 2018. Training a Ranking Function for Open-Domain Question Answering. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*. 120–127.

[12] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*. 2042–2050.

[13] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2333–2338.

[14] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 133–142.

[15] Minsuk Lee, James Cimino, Hai Ran Zhu, Carl Sable, Vijay Shanker, John Ely, and Hong Yu. 2006. Beyond information retrieval : medical question answering. In *AMIA annual symposium proceedings*, Vol. 2006. American Medical Informatics Association, 469.

[16] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *International Conference on Learning Representations (ICLR)* (2017).

[17] Gang Luo, Chunqiang Tang, Hao Yang, and Xing Wei. 2008. MedSearch: a specialized search engine for medical information retrieval. In *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 143–152.

[18] Yuanhua Lv and ChengXiang Zhai. 2011. When documents are very long, BM25 fails!. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 1103–1104.

[19] Wenlei Mao and Wesley W Chu. 2002. Free-text medical document retrieval via phrase-based vector space model.. In *Proceedings of the AMIA Symposium*. American Medical Informatics Association, 489.

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[21] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to match using local and distributed representations of text for web search. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1291–1299.

[22] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* 24, 4 (2016), 694–707.

[23] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI Press, 2793–2799.

[24] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[25] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 2383–2392.

[26] Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, Vol. 242. 133–142.

[27] Stephen E Robertson and K Sparck Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information science* 27, 3 (1976), 129–146.

[28] Stephen E Robertson, Steve Walker, Susan Jones, et al. 1995. Okapi at TREC-3. (1995).

[29] Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management* 24, 5 (1988), 513–523.

[30] G. Salton, A. Wong, and C. S. Yang. 1975. A Vector Space Model for Automatic Indexing. *Commun. ACM* 18, 11 (Nov. 1975), 613–620. https://doi.org/10.1145/361219.361220

[31] Rudolf Schneider, Sebastian Arnold, Tom Oberhauser, Tobias Klatt, Thomas Steffek, and Alexander Löser. 2018. Smart-MD: Neural Paragraph Retrieval of Medical Topics. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 203–206.

[32] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[33] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. *International Conference on Learning Representations (ICLR)* (2017).

[34] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 373–374.

[35] Ellen M. Voorhees and Dawn M. Tice. 2000. Building a Question Answering Test Collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '00)*. ACM, New York, NY, USA, 200–207. https://doi.org/10.1145/345508.345577

[36] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations.. In *AAAI*, Vol. 16. 2835–2841.

[37] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. 2014. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1386–1393.

[38] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 189–198.

[39] Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making Neural QA as Simple as Possible but not Simpler. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. 271–280.

[40] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 55–64.

[41] Jun Xu and Hang Li. 2007. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 391–398.

[42] Liu Yang, Qingyao Ai, Jiafeng Guo, and W Bruce Croft. 2016. aNMM: Ranking short answer texts with attention-based neural matching model. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 287–296.

[43] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[44] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. *International Conference on Learning Representations (ICLR)* (2018).

[45] Hong Yu, Minsuk Lee, David Kaufman, John Ely, Jerome A Osheroff, George Hripcsak, and James Cimino. 2007. Development, implementation, and a cognitive evaluation of a definitional question answering system for physicians. *Journal of biomedical informatics* 40, 3 (2007), 236–251.

[46] Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* (2012).

ShadowBlock: A Lightweight and Stealthy Adblocking Browser

Shitong Zhu
University of California, Riverside
shitong.zhu@email.ucr.edu

Umar Iqbal
The University of Iowa
umar-iqbal@uiowa.edu

Zhongjie Wang
University of California, Riverside
zwang048@ucr.edu

Zhiyun Qian
University of California, Riverside
zhiyunq@cs.ucr.edu

Zubair Shafiq
The University of Iowa
zubair-shafiq@uiowa.edu

Weiteng Chen
University of California, Riverside
wchen130@ucr.edu

ABSTRACT

As the popularity of adblocking has soared over the last few years, publishers are increasingly deploying anti-adblocking paywalls that ask users to either disable their adblockers or pay to access content. In this work we propose ShadowBlock, a new Chromium-based adblocking browser that can hide traces of adblocking activities from anti-adblockers as it removes ads from web pages. To bypass anti-adblocking paywalls, ShadowBlock takes advantage of existing filter lists used by adblockers and hides all ad elements stealthily in such a way that anti-adblocking scripts cannot detect any tampering of the ads (e.g., absence of ad elements). Specifically, ShadowBlock introduces lightweight hooks in Chromium to ensure that DOM states queried by anti-adblocking scripts are exactly as if adblocking is not employed. We implement a fully working prototype by modifying Chromium which shows great promise in terms of adblocking effectiveness and anti-adblocking circumvention but also more efficient than the state-of-the-art adblocking browser extensions. Our evaluation on Alexa top-1K websites shows that ShadowBlock successfully blocks 98.3% of all visible ads while only causing minor breakage on less than 0.6% of the websites. Most importantly, ShadowBlock is able to bypass anti-adblocking paywalls on more than 200 websites that deploy visible anti-adblocking paywalls with a 100% success rate. Our performance evaluation further shows that ShadowBlock loads pages as fast as the state-of-the-art adblocking browser extension on average.

CCS CONCEPTS

• **Information systems** → **Browsers**; **Online advertising**; • **Security and privacy** → *Usability in security and privacy*.

KEYWORDS

Adblocking; Anti-adblocking; Browser Modification

ACM Reference Format:
Shitong Zhu, Umar Iqbal, Zhongjie Wang, Zhiyun Qian, Zubair Shafiq, and Weiteng Chen. 2019. ShadowBlock: A Lightweight and Stealthy Adblocking Browser. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313558

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313558

1 INTRODUCTION

The deployment of adblocking technology has been steadily increasing over the past few years. PageFair reports that more than 600 million devices globally use adblockers as of December 2016 [1]. Many reasons contribute to the popularity of adblocking. First, lots of websites show flashy and intrusive online ads that negatively impact user experience. Second, the pervasiveness of targeted or personalized ads has incentivized a global ecosystem of online trackers and data brokers, which in turn raises concerns for user privacy. Third, the inclusion of numerous advertising and tracking scripts causes excessive website bloat resulting in slower page loads. The rise of adblocking has jeopardized the ad-powered business model of many online publishers. For example, U.K. publishers lose nearly 3 billion GBP in revenue annually due to adblocking [2].

In response to adblocking, many publishers have deployed JavaScript-based, client-side anti-adblockers to detect and circumvent adblockers. An anti-adblocker typically consists of two components: detection and reaction. For adblocker detection, common practices include checking the absence of ad elements and proactively injecting bait ad elements [33]. Both practices exploit the fact that adblockers make observable changes to the DOM by either blocking relevant requests or hiding DOM elements directly [37]. As a result, these DOM changes can be perceived by the detection part of anti-adblockers through invocation of JavaScript APIs such as `getElementById()`. After adblocker detection, the reaction component can perform different subsequent operations. It can be aggressive paywalls that prevent users from accessing the content or even switching ad sources.

Adblockers have addressed anti-adblockers in one of the following three ways: (i) blocking the JavaScript code of anti-adblockers using filter lists [29], (ii) disrupting anti-adblocker code based on program analysis [39], and (iii) hiding the trace of adblocking to fool anti-adblockers [37]. The first countermeasure is currently adopted by the adblocking community using filter lists such as Anti-Adblock Killer and Adblock Warning Removal [29]. However, the coverage and accuracy of these filter list is lacking. Our manual evaluation on 207 websites using anti-adblockers with visible reactions (i.e. warning message or paywall), only less than 30% of them are correctly identified by Adblock Warning Removal [4] or Anti-Adblock Killer [5]. This is likely due to the manual nature of filter list curation and maintenance which is cumbersome and error-prone. The second countermeasure of rewriting JavaScript to deactivate anti-adblockers is prone to false positives causing site breakage with unacceptable user experience degradation [39]. The third countermeasure of hiding the trace of adblocking, as

implemented in prior work [37], is not stealthy because it injects new JavaScript which is easily detectable by anti-adblockers.

In this paper, we aim to completely hide the traces of adblocking in a stealthy manner by going deep into the browser core. This is analogous to the rootkits in the OS kernel where user applications are unable to detect the presence of a malicious process [36]. Since the browser core is at a lower level (more privileged), it is in theory capable of hiding the states of adblockers from the anti-adblockers while presenting an ad-free view to the user. Specifically, since anti-adblocker is implemented as client-side JavaScript, it can only access web-page states through a number of predefined Web APIs including the ones used to probe the the presence/absence of ad elements. These APIs are standardized by W3C and implemented eventually by web browsers. ShadowBlock hooks any JavaScript API that can potentially distinguish the difference before and after hiding ad elements, and assures no information about the element hiding can be leaked through such API.

We tackle three major challenges in designing and implementing ShadowBlock. First, existing adblockers' model of blocking ad-related URLs (e.g., scripts, iframes, images) does not fit well in our requirement of presenting the same exact DOM view as if no adblocker is employed. For example, if a DOM element is not even retrieved, ShadowBlock would have no way to fake its size, dimension, and other properties. Second, given that the Web APIs suite and the rendering process implemented in modern web browsers are highly complex and intertwined, there may exist unexpected channels that leak information about adblocking deployment. To achieve stealthy adblocking, we need to ensure that no such channel discloses differentiable information about our element hiding action in a conclusive way. Third, modern web browsers make significant efforts in improving their page loading and rendering performance. As we develop ShadowBlock on open-sourced Chromium, we need to minimize the overhead it incurs during the page load process.

Contributions. We summarize our key contributions as follows.

(1) We design a well-reasoned solution where we present two different views to anti-adblockers and users. On one hand, ad elements are never directly blocked (so they remain visible to anti-adblockers); on the other hand, these ads are stealthily hidden from users.
(2) We reuse the rules from filter lists used by adblocking browser extensions to element hiding decisions. On top of existing lists that are community-backed and have been widely adopted, we replicate 98.3% of their ad coverage according to manual inspection over Alexa Top 1K websites, with less than 0.6% breakage rate.
(3) We design and implement a fully functional prototype which is open-sourced at the time of publication. We evaluate the effectiveness of ShadowBlock prototype on 207 websites with visible anti-adblockers. We pick real anti-adblockers with different trigger mechanisms and of different complexity. All of them are successfully evaded by our new adblocker design.
(4) Our performance evaluation of ShadowBlock shows that it loads pages comparably fast as Adblock Plus on average, in terms of Page Load Time and SpeedIndex.

2 BACKGROUND AND RELATED WORK

2.1 Adblockers And Filter Lists

Mechanisms. Adblockers rely on manually curated filter lists to identify ads on web pages. EasyList and EasyPrivacy are two most widely used filter lists to block online ads and trackers, with about 71000 and 15000 rules [8], respectively. These lists consist of two types of rules which are basically regular expressions. One of them is HTTP rules that block HTTP requests to URL addresses that are known to serve ads. For example, the first filter rule below blocks all third-party HTTP requests to amazon-adsystem.com, preventing any resource on this domain from being downloaded. The other type is HTML rules, which generally hides HTML elements that are identified as ads. For example, the second filter below hides all HTML elements with ID `promo_container` on wsj.com.

```
||amazon-adsystem.com^$third-party
wsj.com###promo_container
```

It is noteworthy that HTML rules are mostly only introduced to complement HTTP rules while dealing with first-party text ads. This is because these text ads are directly embedded into the HTML itself with no associated additional resource loads, making it inevitable to be included while the browser downloads the web page. Otherwise, HTTP rules are preferred as they prevent ad resources from being loaded in the first place, saving unnecessary network traffic and avoiding execution of ad-related scripts to speed up page loading and rendering.

Limitations. A group of volunteers that maintain filter lists carry out the manual process to add new rules, correct and remove erroneous or redundant rules all based on informal feedback from users [7]. Due to its crowd-sourced nature, this laborious effort faces challenges from both the completeness and soundness. In the context of adblocking, the former results in missed ads and the latter often translates to site breakage or malfunction [23]. At the same time, as adblockers gain their popularity rapidly (11% of global Internet population is blocking ads as of December 2016 [1]), online publishers are also fast adopting countermeasures against adblockers that we summarize below.

2.2 Countermeasures Against Adblockers

Concealing ad signatures. First, online advertisers can bypass rules on filter lists by concealing the signatures these lists use to identify ads. At a high level, this line of countermeasures attempts include first-party advertising and rotation of ad serving domains. First-party advertising exploits the fact that many rules on filter lists are designed to block ads from being loaded from third-party servers. Instead, ads are served from the same domain of the web page hosting them and their nature as ads is concealed as normal content [11]. However, adblockers can easily hide any HTML element on a web page by applying HTML rules that are crafted to target elements based on any combination of their CSS/HTML properties. In other words, any CSS selectors used to create and/or locate ads elements, can also let adblockers identify and hide these elements in turn.

Domain name rotation is another tactic for obfuscating advertising content. It relies on ad rotation networks that serve ads from

frequently-changing, or even automatically generated [38] domain names, which overwhelms volunteers that maintain the filter lists so they are hard to keep pace with the rule updates. This can result in, however, the indifferent blocking of all third-party resources on websites that show such ads [9]. By only whitelisting legitimate scripts that support core functionalities, any possible ads or tracking JavaScript are prevented from running, leaving no chance for loading domain rotating ads. Moreover, AdBlock Plus recently launched its Anti-Circumvention Filter List [3] that specifically counter "circumvention ads", including ads adapting the two countermeasures above.

Deploying anti-adblockers. Second, many publishers choose to deploy anti-adblocker JavaScript code to battle the rise of adblocking. Specifically, such client-side scripts consist of two main components, *trigger* that detects the presence of adblockers by checking whether ads or bait elements are still present, and *reaction* that can display warning messages and/or simply report the results to a remote server [33, 39]. Prior work [29] showed that 686 out of Alexa top-100K websites detect and visibly react to adblockers on their homepages. Even worse, these visible ones only account for less than 10% of all anti-adblockers [39]. Zhu et al. [39] showed that among Alexa top-10k websites, 30.5% are countering adblockers in some form, with most of them being silent reporting. In summary, because of their flexibility and ease of deployment, anti-adblockers are considered the most widely used countermeasure against adblockers adopted by online publishers.

2.3 Countermeasures Against Anti-adblockers

Dedicated anti-adblocking filter lists. As a response, adblockers attempt to circumvent anti-adblockers by blocking their JavaScript code snippets, whitelisting bait scripts/elements, or hiding warning notifications. To this end, adblockers once again rely on manually curated filter lists such as Anti-Adblock Killer [5] and Adblock Warning Removal [4]. These lists either trick anti-adblockers' trigger so they cannot detect adblockers, or mute their reaction component to prevent responses after successful detection.

```
/kill-adblock/js/function.js$script
@@||removeadblock.com/js/show_ads.js$script
ilix.in,urlink.at,priva.us###blockMsg
```

For example, the first HTTP rule above blocks the code snippets containing implementation of an anti-adblocking library `KillAdBlock`, and the second whitelists a bait script file named `show_ads.js` that is used to detect adblockers. The third HTML rule hides the warning message with ID `blockMsg` issued by the associated anti-adblocker. However, our manual evaluation (§4) shows that these lists targeting anti-adblockers are generally ineffective. Only less than 30% of the anti-adblocking warning messages can be removed by the state-of-the-art filter lists. This is again partly because of the crowdsourcing nature of these lists, and also the rising popularity of third-party anti-adblocking services that deploy sophisticated techniques dedicated for detecting/circumventing adblockers [33].

Disrupting anti-adblocker code. Other than the filter lists that have been officially adopted by adblockers, there are also research efforts for detecting and evading anti-adblockers. One solution to measure the anti-adblockers is to perform program analysis techniques that automatically determine if a script functions for anti-adblocking purposes. Such analysis can be static that is based on syntactic and structural features extracted from JavaScript code, and utilizes machine learning approaches to classify the code from ground-truth-labeled training data [29]. It can also be dynamic that captures JavaScript behavior at runtime by collecting and analyzing differential execution trace with the adblocker turned on and off [39]. After successfully pinpointing the critical conditions that are used by anti-adblockers to assert/react against the presence of adblockers, one can choose to rewrite these conditions to prevent the anti-adblockers from functioning. This approach is generally intrusive (patching Javascript can be tricky and cause breakage) and easy to evade. Indeed, the overall success rate of this strategy is shown to be only less than 80%.

Hiding adblockers. Besides disrupting the functionalities of anti-adblockers, researchers also have proposed a way to hide the trace of using adblockers, or known as *stealthy adblocking*. In [37], Storey et al. created a shadow copy of the DOM that anti-adblockers operate on before any adblocking actions take place, and then redirects all JavaScript APIs (e.g. `getElementById()`) that can be used to detect the presence of ad elements to the copy instead of the original DOM. However, this so-called rootkit-style stealthy adblocker has inherent drawbacks. First, unless it lives in browser core and with significant engineering efforts, the underlying DOM mirroring and propagation are difficult to be complete in all cases. This is especially problematic in the context of web browsing as any site breakage causes unacceptable user experience degradation. Second, even with a perfect implementation, maintaining a live copy of complicated data structures such as DOM poses a prohibitively high overhead onto the rendering performance of modern web browsers. Given that modern browsers place significant emphasis on performance, heavy operations like such at runtime are generally not acceptable.

3 SHADOWBLOCK

In this section, we first provide an overview of ShadowBlock's architecture. We then discuss ShadowBlock's two building blocks: (1) the identification of ad elements by translating filter list rules to per-element hiding decisions and (2) the concealment of our hiding actions. Finally, we summarize the modifications we make in the relevant modules of Chromium.

3.1 ShadowBlock Overview

Figure 1 illustrates ShadowBlock's architecture. It consists of two sub-systems: one translates rules from filter lists and use them for identifying ad elements in DOM to hide; the other hooks necessary points in Chromium to ensure that the hiding actions are transparent to the trigger/detection component of anti-adblockers. Recall from Section 2.1 that filter lists contain tens of thousands of rules that either block HTTP requests to fetch ad resources or hide HTML ad elements. To prevent exposing adblocking actions to anti-adblockers, we need to hide the changes in DOM or other states (e.g. resource loads) introduced by adblocking because these changes can be detected by anti-adblockers through JavaScript APIs such as `getElementById()`. In order to do so, we first allow all HTTP requests to proceed, then mark any element that results in ads, and subsequently hide the marked elements. It is important that we

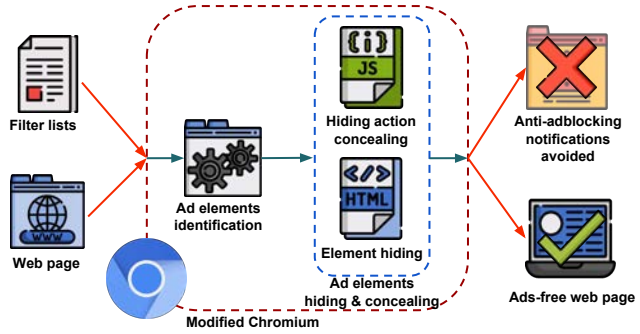


Figure 1: Architectural overview of ShadowBlock

allow these elements to be retrieved so when the anti-adblocking script queries the state of the element, ShadowBlock will be able to generate valid responses (e.g., dimension of the element).

3.2 Identifying Ad Elements

Next, we explain our approaches for marking different types of ad elements. In general, there are two types of elements: (1) those that are statically embedded in the HTML, and (2) those that are dynamically created by JavaScript. First, some ad elements are statically embedded in the HTML, including ad images, iframes, media files (i.e. video and audio). Such ad elements can be typically identified by matching against the HTML rules on filter lists. Second, ad scripts usually *create* new ad elements that display advertising content. These dynamically created elements should be identified, marked, and hidden.

Ad elements loaded statically. ShadowBlock does not need to do anything special for such elements. Ad filter lists already contain extensive rules that cover them. For example, the rule `||googlesyndication.com/safeframe/` in EasyList blocks the HTML file from https://tpc.googlesyndication.com/safeframe/1-0-23/html/container.html on site cnbc.com, which prevents a container frame used by Google Ads from being loaded. To hide such an ad element, ShadowBlock needs to first identify the iframe element with the blocked URL as its source attribute and then hide it.

Alternatively, Easylist rules may hide ads based on element properties (e.g., element id). We simply reuse these rules to match elements in the page, and mark them accordingly.

Ad elements generated dynamically by ad scripts. There are generally two cases. The easy and the hard. For the easy case, the dynamically generated ads may contain URLs or ids that already show up on Easylist. This allows us to directly mark them as ads using very much the same strategy as mentioned above.

For the hard case, the dynamically generated elements are not on Easylist. This is because it is assumed that ad scripts are blocked upfront and therefore there is no need to block the elements generated by them. In ShadowBlock, in contrast, we need to allow ad scripts to load and execute, which mandates us to track such elements.

To identify elements dynamically created by ad scripts, we need to attribute each element to the script that created it. More formally, we can define this process of attribution as tracking the *data provenance* of HTML elements using taint analysis. Note that there are in general two types of element creation that can be initiated by a script: (1) control-flow-based creation, in which the script directly invokes JavaScript API (e.g. `createElement(tagName)`) and only propagates data from itself into the new element; and (2) data-flow-based creation, in which the script uses data from sources other than itself into the new element (e.g. `createElement(fetchTagNameFromServer())`). Dynamic taint analysis [31] can accurately track the data provenance through taint propagation for both types. Simply put, taint analysis involves *taint source* (where data comes from), *taint sink* (where data ends), and *propagation policies* that define how that tainted data are propagated through the program execution. In our case, all data derived from an identified ad script as "tainted" (i.e., data downloaded through an ad URL, or retrieved/generated by an ad script), then such data can be tracked standard taint analysis propagation policies for JavaScript (e.g., [27]). Finally we will hide any tainted HTML elements (i.e. taint sinks).

Unfortunately, such dynamic taint analysis at runtime usually incurs significant overhead for web browsing [31]. In addition, we argue that the cases where taint analysis will be required are limited. Specifically, even if ad elements take dynamically fetched data, e.g., `createElement(fetchTagNameFromServer())`, they are most likely created by ad scripts. This is sufficient for us to mark the element and hide it (irrespective of what data are actually fetched from the server). The reasoning is that if we consider extension-based adblockers as our baseline, the dynamically created element wouldn't even exist in the first place (as the script as a whole would have been blocked). Based on the above, we devise a simple technique which we call *execution projection* using the call stack information extensively (which are used for various other purposes as well [30, 32]). At a high level, we maintain an "execution stack" that keeps track of what scripts are being executed at any given time point, and mark an element as ad if there is *any* ad script in the stack when the element is being created. For example, consider a simple ad script `ad_loader.js` in Code 1.

```
1 var ad_img = document.createElement("img");
2 ad_img.src = "https://some_ad_publisher.com/ad.jpg";
3 document.body.appendChild(ad_img);
```

Code 1: Example ad script `ad_loader.js`

In this example, at the time of the image element creation, `ad_loader.js` would be at the top of the execution stack because it is being executed.

After attributing elements to ad scripts, which are identified using filter lists, we can mark all ad elements and hide them accordingly. We illustrate this projection/marking process in Figure 2. In more complex cases, there can be many scripts in the executing stack, because code in one script can invoke functions in other scripts, and so on. More importantly, ad scripts can invoke non-ad libraries (e.g. jQuery) to create ad elements so the top script in stack at the time of element creation is not necessarily ad script. To tackle this challenge, we need to scan the entire execution stack. If there is any ad script in the execution/invocation chain, ShadowBlock should mark the element being created as ad. This is because any script (ad-related or not) invoked by a known ad script should

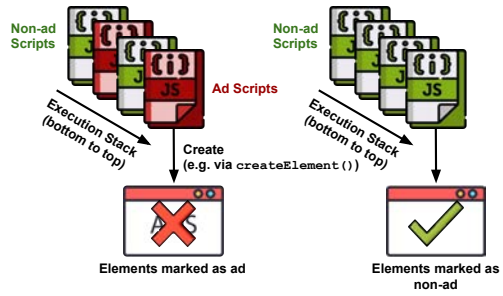


Figure 2: Execution projection for marking script-created ad elements

never have been executed in the first place, given that adblocking extensions simply block the whole ad script altogether.

Note that our approximate solution does not handle a special case when an element is created via a JavaScript API that gets overridden (hooked) by an ad scripts . Since client-side JavaScript is allowed to override (i.e. injecting code containing a callback to its own) arbitrary JavaScript API functions (e.g. `createElement()`) with its own version, we would see ad script in the stack when an element is being created via the API overridden by an ad script. In this case, we may mistakenly mark a non-ad element as ad when the overriding ad script does not propagate any data into the newly created element. To tackle this issue, at the time of element creation, we would need to further check whether the code injected by the overriding ad script alters the element. If it alters the element then we mark the element as ad, and non-ad otherwise.

```
var original = document.createElement;
document.createElement = function (tag) {
  new_elem = original.call(document, tag);
  report_statistic_to_server(new_elem);
  return new_elem;
};
```

Code 2: API overriding with the element intact

```
var original = document.createElement;
document.createElement = function (tag) {
  new_elem = original.call(document, tag);
  ad_elem = change_to_ad_elem(new_elem);
  return ad_elem;
};
```

Code 3: API overriding with the element altered

3.3 Stealthily Hiding Ad Elements

After identifying ad elements, ShadowBlock needs to stealthily hide them so as to not leaving its trace to anti-adblockers. Next, we discuss how we realize such stealthiness through CSS property access API hooking.

Choice of hiding mechanism. We first need to decide how to hide ad elements within Chromium. Given the complexity of modern web browsers and API standards, we can hide an HTML element in several different ways. To better understand different hiding mechanisms, we illustrate Blink's rendering process in Figure 3 [16]. Blink's rendering path, from parsing an HTML file to the pixel display on user's screen, can be summarized in the following phases.

(1) **Parse** flat HTML and CSS in plain-text to DOM and CSS Object Model (CSSOM) in tree structure.
(2) **Combine** DOM and CSSOM to Render Tree, which captures all the visible DOM content and all the CSSOM style information for each node.
(3) **Paint** the rendered pixels to user's display according to Render Tree.

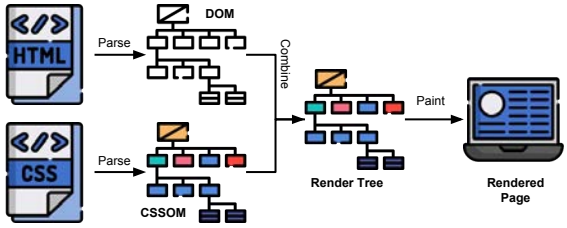


Figure 3: Rendering Path for Blink

In theory, each of these 3 phases contain APIs/modules that we can leverage for hiding an HTML element (or in other words, preventing it from being rendered). We outline different possible strategies to hide HTML elements below:

(1) **DOM/CSS layer**: (i) remove the element from DOM; (ii) set the element's style to `display:none`, `visibility:hidden` or `opacity:0`
(2) **Render Tree layer**: remove the `LayoutObject` (i.e. a styled node) from Render Tree
(3) **Paint layer**: prevent the region in pixels associated with the element from being painted

Note that generally, the higher-layer (i.e. closer to DOM/CSS layer) we tweak around in the hierarchy, more engineering effort is required for ensuring its stealthiness against anti-adblockers, while narrower gap we have to bridge for translating the identified ad HTML elements to the data structure corresponding to that layer (e.g. CSS properties for DOM/CSS layer, `PaintBlock` for Paint layer etc.). In contrary, the lower-layer (i.e. closer to Paint layer) our modifications reside, fewer unexpected channels there are that can potentially leak the hiding activities, but at the same time more engineering efforts are required for translating the identified ad elements to that layer's data structure.

After investigating different possibilities, we find the CSS property `visibility:hidden` achieves the most suitable trade-off for our objective. It persists in phase (1), functions in phase (2) of the rendering path, and eventually affects both Render Tree and painted/rendered page in phase (2) and (3). On one hand, `visibility` is a property associated with every HTML element so we can easily identify after marking its effective element as ad, without the need of further tracing. On the other hand, unlike `display:none`, it by design preserves the space taken up by the hidden element so it causes no side-effect to the layout of the document, minimizing the impacted points that need to be hooked for covering the hiding action. Figure 4 shows the visual difference between their respective effects. Compared to `opacity:0` that also preserves the occupied space, `visibility` is a categorical CSS property instead of numerical as `opacity` so its implementation in Chromium is considerably less complex, simplifying our hooking logic as well.

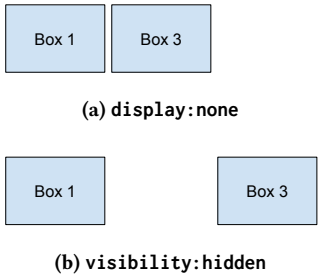


Figure 4: Comparison of toggling different CSS properties (Box 2 is hidden)

Hooking for concealing hiding actions. After hiding the target ad element by setting its `visibility` CSS property value [21] to `hidden`, we next need to cover any traces that result from this change of value and can be detected by anti-adblockers. Since anti-adblocking scripts are client-side JavaScript code, which can only access the change of states happened in the page through JavaScript Web APIs [22], we search through the source code of Chromium, analyze relevant modules and locate the following three categories of APIs that are impacted by the `visibility` CSS property:

- **CSS/Style-related**: changing a CSS property value immediately affects its own return value to JavaScript APIs via `getComputedStyle()`. We hook this value to `visible` to fool anti-adblockers. Fortunately, for our case, setting `visibility:hidden` preserves the space the element occupies, so it does not collapse the page layout nor affect other relevant CSS properties such as `offsetHeight/offsetWidth`.
- **Event-related**: flipping the `visibility` property prevents an element from receiving some DOM events, such as `onfocus`. Since anti-adblockers can leverage these events as a side-channel to infer the real visibility of an element, we hook relevant modules in Chromium so hidden elements can receive these events just like visible elements.
- **Hit-testing-related**: another effect of setting `visibility:hidden` is Blink treats elements with it as inapplicable to Hit Testing, which checks if an element is clickable by users in their viewpoints. This removes the hidden element from return values to APIs like `elementFromPoint()`, which can potentially be used by anti-adblockers to differentiate hidden elements from visible ones. We hook relevant modules in Blink to cover it.

Since all JavaScript APIs, to our best knowledge, directly or indirectly rely on the modules above to determine an element's visibility, we ensure the completeness of our hooking against potential information leakage to client-side anti-adblockers. It is worth noting that to avoid affecting existing `visibility:hidden` CSS property value, in Chromium we create a new `visibility:fake-visible` value that completely mimics what `visibility:hidden` behaves, except for the points that are intentionally hooked. Moreover, since `visibility` is a CSS property that inherits from parent node to child nodes, we can make the identified ad elements invisible to user's display even though we only identify the top-most element as ad according to filter lists.

3.4 Chromium Modification

Next, we describe our modifications to Chromium for implementing ShadowBlock. We start with a brief introduction of Chromium's architecture, then move to the instrumentation we use for identifying ad elements, and lastly discuss the modules we leverage for hiding identified ad elements and hook for eliminating the traces resulted from the hiding action. We implement the prototype of ShadowBlock with 1307 LOC (1265 LOC addition and 42 LOC deletion) in C++ on top of Chromium [1]. Note that we re-use `SubresourceFilter` [18] and `libadblockplus` [10] for parsing filter lists with production-level robustness.

Chromium Architecture. Chromium's rendering engine is called Blink and its JavaScript engine is called V8. Blink [6] is responsible for fulfilling the rendering path shown in Figure 3, in which its core module renders all HTML elements and handles their `visibility` CSS properties we use for hiding ad elements. V8 [19] handles the compilation and execution of all JavaScript code, including the ad scripts ShadowBlock needs to identify and anti-adblocker scripts that intend to detect our hiding action over ad elements. Blink has a bindings module to handle interactions between rendering and JavaScript execution. Rendering related script tasks are passed through bindings module. For example, JavaScript API calls such as `getComputedStyle()` are handled through the bindings module.

Instrumentation for identifying ad elements. As discussed in Section 3.2, there are three types of ad elements ShadowBlock needs to identify.

First, for ad elements generated by ad scripts, we rely on execution projection. In Chromium, we leverage `blink::SourceLocation::Capture` to capture the full `v8::v8_inspector::V8StackTrace` that includes the entire JavaScript call stack[2] at any given time point. It serves as the underlying stack tracing mechanism for V8's debugger/inspector, and has therefore been optimized with low overhead [20].

Second, we instrument the constructor of `blink::Element` class in Blink, which captures the earliest point of creation event for all HTML elements. Because of V8's single-threading nature, we can safely associate the current stack trace to the element creation event, and scan the stack to match the scripts in it against filter lists. If it is a match, we then mark the element as ad. Additionally, we instrument the `DispatchWillSendRequest` event in both `blink::FrameFetchContext` and `blink::WorkerFetchContext` to intercept the point when an ad script loads another script, and mark the loaded script as ad script. By adding such loaded ad scripts to a set, we match the stack trace against them as well at element creation, ensuring we can mark all ad elements.

Third, for elements loaded with resources that match rules in the filter lists, we intercept the `AttributeChanged` event in `blink::`

[1]We open source our implementation at https://github.com/seclab-ucr/ShadowBlock to allow reproducibility as well as help future extensions by the research community.
[2]We admit that there are cases where asynchronous tasks are not correctly captured by the default V8 call stack trace. However, we argue this incompleteness only translates to very limited number of missing ads according to our manual evaluation in Section 4.2.

Element and match the URL against filter lists, if it is a match we mark this element as ad. For element hiding rules, we adopt libadblockplus [10], a C++ wrapper library around the core functionality of Adblock Plus to parse filter lists and generate the CSS selectors for matching ad elements for a particular domain. Then, we mark the ad elements that match the generated CSS selectors by calling ContainerNode::QuerySelectorAll().

Since many web pages are dynamic due to JavaScript execution over time, we also need to monitor attribute changes of each element. For this purpose, we instrument the AttributeChanged event and match any element with newly changed attributes against CSS selectors from HTML rules. We mark an element as ad if it is a match, or un-mark the element if this element has been marked but it is not matched this time. Note that in order for minimizing the number of matches needed to perform, we conduct the first batch match (via QuerySelectorAll()) after the load event of DOM is fired, and then match elements upon their attribute changes. This design choice leaves a short period of time (few milliseconds) between page navigation and load DOM event in which ads are displayed. We make this trade off to reduce the overhead incurred by QuerySelectorAll(). In comparison, adblocking extensions such as Adblock Plus inject CSS rules when document.readyState turns interactive [26], which happens before the load event. However, it is important to note that most ads in current web ecosystem are loaded in an asynchronous manner and are unlikely to appear before the load event in first place.

Stealthy modifications for hiding ad elements. As mentioned earlier, we leverage visibility CSS property to hide identified ad elements by creating a new fake-visible enumerate and visually hide elements with this enumerate, as if it behaves as hidden. In the meantime, we hook relevant modules in both Blink and its bindings with V8 to ensure the stealthiness of our hiding action. More specifically, for eliminating traces accessible by CSS/Style-related APIs, we hook CSSComputedStyleDeclaration::GetPropertyCSSValue in Blink and force return visible to queries about hidden elements. For event-related APIs, we hook Element::IsFocusableStyle() and other conditions that determine if an element can receive events. Lastly, we hook ComputedStyle::VisibleToHitTesting() so ad elements are still regarded "visible" from the viewpoint perspective of Blink. In principle, our hooking guarantees that the identified ad elements are invisible to user's display as pixels on screen but appear as visible to APIs accessible to client-side JavaScript.

4 EVALUATION

We evaluate ShadowBlock in terms of its (1) stealthiness against anti-adblockers, (2) ad coverage, and (3) performance as as compared to adblocking extensions.

4.1 Stealthiness Analysis

Takeaway: ShadowBlock has 100% success rate against anti-adblockers whereas state-of-the-art anti-adblocking filter lists have only 29% success rate.

Experimental Setup. To evaluate the stealthiness of ShadowBlock, we use previously reported [39] 682 websites with visual anti-adblockers. We manually analyze these websites and find that 207 of them still use visible anti-adblockers. For each website, we perform stealthiness comparison as follows.

(1) Open a website with four Chromium instances simultaneously. Each instance has a different profile configuration: (i) no modification or extension; (ii) Adblock Plus extension with EasyList only; (iii) Adblock Plus extension with EasyList, Anti-Adblock Killer list, and Adblock Warning Removal list; and (iv) ShadowBlock using EasyList.
(2) Scroll the page down to the bottom and wait for 30 seconds after the load event has fired to ensure complete page load.
(3) Capture the full-page screenshots (including content after scrolldown) for all browser instances.
(4) Manually inspect the screenshots: compare (i) and (ii) to determine if the page has visual anti-adblocker. If so, further compare (ii) and (iii) to check whether anti-adblocking filter lists evade the anti-adblocker, compare (iii) and (iv) to check whether ShadowBlock achieves the evasion.

Table 1: Breakdown of stealthiness analysis

Tool	Notification	Ad switching	Crypto-mining
Total	201	5	1
ShadowBlock	201 (100%)	5 (100%)	1 (100%)
Filter lists	59 (29%)	1 (20%)	0 (0%)

Results. In addition to visible anti-adblocking notifications, we also consider ad switching and crypto-mining reactions from websites. Table 1 compares ShadowBlock with anti-adblocking filter lists for each of these anti-adblocking reactions. "Notification" refers to websites that show anti-adblocking notifications such as paywalls. "Ad switching" refers to websites that switch their ad sources upon detection of adblockers. "Crypto-mining" refers to the websites that load crypto-mining scripts to mine crypto-currencies on detection of adblockers [28]. We note that ShadowBlock has 100% success rate as compared to 29% success rate of anti-adblocking filter lists.

Case Studies. Below we discuss a few interesting examples of anti-adblockers that ShadowBlock successfully handles but filter lists do not. Note that besides visible anti-adblocking notifications, we also include one example discovered in the wild that uses non-visual countermeasure against adblocker users.

Ad source switching. On detecting adblockers, some websites switch their ad sources to sources that are currently not blocked by filter lists. Figure 5a and 5b show an example from golem.de. Since ShadowBlock stealthily hides the original ads, the ad source switching script is never triggered. Therefore, unlike what Figure 5b shows in which adblocking extensions fail to remove the replaced ad, ShadowBlock successfully hides it.

Silent reporting. Besides visible reaction, anti-adblockers can also choose to silently report the adblocking status to back-end servers to collect adblocking statistics. For example, varmatin.com uses Code 4 to place a bait with keywords on EasyList to track adblocking users and report the status to back-end server. ShadowBlock handles such cases and these statistics are never reported.



(a) Original ad



(b) Replaced ad

Figure 5: Ad switching behavior on golem.de

```
function checkAds() {
  if ($(document.getElementById('adsense')).css('display'
      ) !== 'none' && $('#myadsblock').length === 1) {
    dataLayer.push({
      'DimAdBlock': 'Unblocked'
    });
    window.adblockdetected = false;
  } else {
    dataLayer.push({
      'DimAdBlock': 'Blocked'
    });
    window.adblockdetected = true;
  }
}
```

Code 4: Code snippet on varmatin.com **of silent anti-adblocker**

Crypto-currency mining. Some websites have started to employ cryptojacking as a response to adblocking [28]. To this end, websites use anti-adblockers to detect use of adblockers and load scripts to mine a crypto-currency on user's browser. Mining crypto-currencies consumes processing power on user's machine. For example, knowlet3389.blogspot.com blocks organic content on detection of adblockers and asks users of allow crypto-currency mining for monetization instead. Code 5 shows the crypto-currency mining script on knowlet3389.blogspot.com.

```
setInterval(function() {
  try {
    var a3 = document.getElementById('AdSense3');
    if (a3.offsetHeight < 33 || a3.clientHeight < 33) {
      throw "Fuck U AdBlock!";
    }
  } catch (err) {
    miner.start(CoinHive.IF_EXCLUSIVE_TAB);
  }
}, 5487);
```

Code 5: Code snippet on knowlet3389.blogspot.com **for mining crypto-currency**

4.2 Ad Coverage Analysis

Takeaway: ShadowBlock achieves 97.7% accuracy, with 98.2% recall and 99.5% precision in blocking ads on Alex top-1K websites.

Experimental Setup. For ad coverage analysis, we use ShadowBlock on Alexa top-1K sites and measure its accuracy in terms of



(a) Adblock Plus



(b) ShadowBlock

Figure 6: Minor visual breakage caused by ShadowBlock

true positive (TP), false negative (FN), true negative (TN), and false positive (FP). We define TP, FN, TN, and FP as:

TP: All ad elements on a page are correctly hidden.
FN: At least one ad element on a page is not hidden.
TN: No non-ad element on a page is incorrectly hidden.
FP: At least one non-ad element on a page is incorrectly hidden.

For each website, we perform ad coverage comparison as follows.

(1) Open a website with three Chromium instances simultaneously. Each profile has a different profile configuration: (i) no modification or extension; (ii) Adblock Plus extension with EasyList; and (iii) ShadowBlock using EasyList.
(2) Scroll the page down to the bottom and wait for 30 seconds after the `load` event has fired to ensure complete page load.
(3) Capture the full-page screenshots (including content after scrolldown) for all browser instances.
(4) Manually inspect the screenshots: compare (i), (ii) and (iii) to determine if the website has any FPs or FNs.

Results. Table 2 shows the breakdown of our manual analysis. We evaluate results in terms of TPs, FNs, TNs, and FPs. Note that we are able to perform our analysis on 943 out of Alexa top-1000 websites. The remaining websites failed to properly load primarily due to server-side errors (e.g., 404).

False Positive Analysis. From Table 2, we note that ShadowBlock has only 0.5% false positive rate. However, they are still critical as they might lead to user experience degradation. We further investigate false positives to diagnose their root cause.

theatlantic.com is an example of false positive. Figure 6 shows that ShadowBlock incorrectly hides organic content at the bottom of the page. On further investigation, we find that an ad script `ads.min.js` loads another script `script.js` that hooks JavaScript API methods. In this case, even when a non-ad element is being processed it would go through same hooked JavaScript API methods. Since our ad marking heuristics check for the presence of ad scripts on execution stack it will incorrectly mark such elements as ads. In comparison, extension-based adblockers block the request for downloading `ads.min.js` in the first place, so the hooking script never gets executed. As discussed in Section 3.2, we can deal with

Table 2: Breakdown of ad coverage analysis

Event	TP	FN	TN	FP
Count	926 (98.2%)	17 (2.8%)	938 (99.5%)	5 (0.5%)

this issue by checking whether or not the overridden API alters the elements and hiding the elements altered by ad scripts.

False Negative Analysis. From Table 2, it can be seen that ShadowBlock has only 2.8% FNs. We further investigate FNs and identify that they are again caused by corner cases not covered by ShadowBlock and that they can be handled by performing taint analysis.

sohu.com is an example of false negative. On further investigation, we find that sohu.com uses a non-ad script (not on Easylist) to load both ads and non-ad content on the page. Since ShadowBlock only attributes elements created by ad scripts as ads, it misses dual-purpose scripts. It's noteworthy that this should be a rare case, as it is contrary to the common practice of using dedicated third-party scripts to create and load ad elements that most ad publishers exercise today. These publishers normally deploy third-party ad scripts because they have a complex bidding system and prefer dominant control over their ad modules

```
"resource": {
  "type": "text",
  "text": "Guangzhou, Audi TT 82.2K RMB off",
  "md5": "",
  "click": "http://dealer.auto.sohu.com/882054/promotion/
      article?id=7360579",
  "imp": [],
  "clkm": [],
  "adcode": "Guangzhou, Audi TT 82.2K RMB off",
  "itemspaceid": "15770"
}
```

Code 6: JSON snippet on sohu.com **for loading ads (translated from Chinese)**

We can tackle this issue by implementing the taint analysis approach discussed in Section 3.2. Specifically, Code 6 shows the snippet of a JSON file on sohu.com containing parameters required to create ad elements. In this case, we will need to first mark the JSON object as ad-related, or tainted, and whenever any piece of the data derived from it propagates to any element field (e.g. the URL in JSON's `click` field is used to set an element's `src` attribute), we mark the element as ad. In comparison, extension-based adblockers intercept the network request to load such ad JSON based on its URL in the first place, which effectively prevents the resulting ad HTML element from being created.

Similarly, we observe FNs on youtube.com where ShadowBlock is unable to hide all video ads. Our manual analysis shows that youtube.com leverages the Media Source Extensions (MSE) API [25] to load video segments through AJAX requests as byte streams. Unlike the standard HTML `video` tag that loads videos as HTTP requests, youtube.com loads ad videos in `Blob` objects [24] which are downloaded by JavaScript on the fly. ShadowBlock cannot identify video ads loaded as `Blob` objects, because both ad and non-ad objects are generated by the same non-ad script and assigned to a single HTML `video` element. Unlike our strategy that relies on differentiating ad scripts, extension-based adblockers block the AJAX requests to fetch ad video segments based on their URLs, which achieves the goal of ad removal. As discussed earlier, taint tracking can be used to address this challenge.

Even through we show that tainting is the ultimate solution to the FN cases encountered during our evaluation, we argue that it a comprehensive taint engine poses prohibitively high runtime overhead in the context of web browsing [27, 31]. More importantly, our evaluations have shown the sufficient accuracy of ShadowBlock with the lightweight stack-based execution approximation, as discussed in Section 3.2.

4.3 Performance

Takeaway: we use two web performance metrics: Page Load Time (PLT) and SpeedIndex. ShadowBlock speeds up page loads by 5.96% in terms of median PTL and 6.37% in terms of median SpeedIndex, on Alexa top-1000 websites.

Page Load Time (PLT). PLT has been the de-facto standard metric for measuring web performance. PLT can be computed by timing the difference between certain browser events using the Navigation Timing API [12]. In order to minimize variations introduced by the initial network setup (e.g., establishing TCP connection with server), we measure the time between `responseStart` [15] and `loadEventStart` [14] events.

SpeedIndex. PLT does not capture a real user's visual perception of webpage rendering process. For example, two pages A and B can have exactly the same PLTs, but page A can have 95% of its visual content rendered by a certain time point while page B has only rendered 30%. From the user perception perspective, page A outperforms page B but they are equally good in terms of PLT. To address this issue, SpeedIndex [17] was proposed to capture the visual progress of *above-the-fold* content, i.e., content in the viewport without scrolling. Unlike PLT, SpeedIndex measures how visually complete a webpage looks at different points during its loading process. Specifically, the page loading process is recorded as a video and each frame is compared to the final frame, for measuring completeness. SpeedIndex is computed using the following formula:

$$SpeedIndex = \int_{t_{begin}}^{t_{end}} 1 - \frac{VisualCompleteness}{100},$$

where t_{begin} and t_{end} represent the time points of the start (i.e. `responseStart` event in our case) and end (i.e. `loadEventStart` event in our case) of video recording, respectively. $VisualCompleteness$ measures the difference of the color histogram for each frame in the video versus the histogram at frame t_{begin}, and compares it to the baseline (difference of histogram at t_{begin} and t_{end}) to determine how "complete" that video frame is.

We emulate DSL network condition by throttling Chromium [13] to 4 Mbps downlink bandwidth and 5ms RTT latency for all responses to best mitigate measurement volatility across different browser instances. [3] For each site, we first load the webpage to generate its resource cache, then we re-load the webpage 10 times and average the measured PLT and SpeedIndex for each page load. Note that our warm-up strategy ensures most of the static non-ad resources are cached, while ad resources dynamically generated by JavaScript execution are not. This is intended, because we want to minimize the variability introduced by irrelevant factors such as processing non-ad network traffic.

[3] We also run another configuration with 750 Kbps downlink bandwidth and 100ms RTT latency to emulate a regular 3G condition [13] and observe similar median trends for both PLT (DSL -5.96% vs 3G +0.30%) and SpeedIndex (DSL -6.37% vs 3G -7.07%) with respect to Adblock Plus.

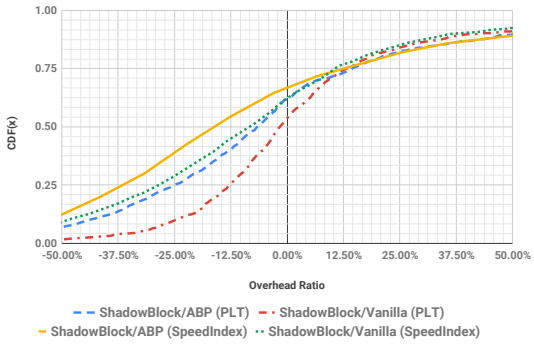


Figure 7: CDF for performance metrics

We compute relative ratio of PLT/SpeedIndex across two different configuration pairs (A) ShadowBlock and vanilla Chromium; and (B) ShadowBlock and Adblock Plus (EasyList + Anti-blocking lists), which are denoted as $Conf_A$ and $Conf_B$, respectively.

$$\frac{PLT_{Group_A} - PLT_{Group_B}}{PLT_{Group_B}}$$

$$\frac{SpeedIndex_{Group_A} - SpeedIndex_{Group_B}}{SpeedIndex_{Group_B}}$$

Overall, both PLT and SpeedIndex show that ShadowBlock speeds up page load time as compared to not only Adblock Plus but also vanilla Chromium. In comparison to Adblock Plus, ShadowBlock speeds up page loads by 5.96% in terms of median PTL and 6.37% in terms of median SpeedIndex. In comparison to vanilla Chromium, ShadowBlock speeds up page loads by 1.03% in terms of median PTL and 5.22% in terms of median SpeedIndex.

The distributions of PLT and SpeedIndex in Figure 7, also confirm this trend. We surmise that ShadowBlock's speed up with respect to Adblock Plus is because ShadowBlock's in-browser modifications, versus Adblock Plus's JavaScript-level implementation, inherently incur less overhead without the necessity of communications between browser core and extension code. For the speed up with respect to vanilla Chromium, it can be explained because ShadowBlock avoids the rendering and painting work for hidden ad elements.

5 DISCUSSIONS AND LIMITATIONS

Hiding instead of blocking. As discussed in Section 3.3, ShadowBlock is designed to visually hide ad elements. Compared to extension-based adblockers that prevent ad resources from loading, ShadowBlock's hiding strategy is inherently limited in two ways. First, ShadowBlock loads the ads and then hides them, thus does not save any network bandwidth. Second, it allows ad resources to load and ad-related scripts to execute, thus exposes users to online tracking. However, we argue that ShadowBlock can complement other tracker blocking approaches that obfuscate and anonymize user-identifiable data [34, 35] which do not require blocking ad requests or stopping script execution.

Completeness of implementation. As discussed in Section 3.2, a sufficiently complete yet lightweight taint analysis engine is required to tackle all the FN cases we encounter during ad element identification. However, given the adequate accuracy and practical runtime overhead level, we consider our execution projection technique a sufficient and necessary simplification of taint tracking conceptually.

Adversary from publishers. Modern websites enforce strong isolation among different scripts running in the same document. Violating such policies would normally raise substantial awareness to owners of other scripts or the website itself. This isolation also helps establish the assumption that ad scripts should never interact with non-ad elements in the same page. However, As soon as the publishers become aware of our approach, they might attack ShadowBlock by pro-actively breaking this assumption to cause collateral damage. For example, an ad script can intentionally modify an attribute of a non-ad element without changing its semantics (e.g. by changing the text encoding). In this case, if we blindly follow the taint tracking and mark the element with taint as ads, we might end up hiding benign elements. To address this challenge, we will need an equivalence test on the semantics of the cases with and without tainting.

Alternatively, an adversary may attempt to detect ShadowBlock. Even though we have closed all normal channels (JavaScript APIs) from leaking information about the presence of ShadowBlock. The adversary may still use more extreme means such as side channels. For instance, if we conduct taint analysis, we slow down the JavaScript execution and therefore they can potentially detect ShadowBlock by timing. However, we argue that this will be extremely challenging if not impossible, because there exist many browsers with different versions of JavaScript engines. There are simply too many possibilities if an adversary observes that the execution is slightly slower (it can even be just a slow machine).

6 CONCLUSIONS

In this paper, we propose ShadowBlock— a Chromium based stealthy adblocking browser. In addition to blocking ads it hides the traces of adblocking, making it insusceptible to anti-adblocking. Compared to the current state-of-the-art adblocking extensions, that block resources, ShadowBlock allows resources to load and keeps track of them. Later it hides the loaded resources and fakes their states to JavaScript APIs used by anti-adblockers. Through manual evaluation, we find that ShadowBlock (i) achieves 100% success rate in evading visual anti-adblockers; (ii) replicates 98.2% of ads coverage achieved by adblocking extensions; and (iii) causes minor visual breakage on less than 0.6% of the tested websites. In addition, we evaluate ShadowBlock's performance and find that it loads web pages as fast as adblocking extensions in terms of SpeedIndex and Page Load Time, on average. In summary, ShadowBlock constitutes a substantial advancement for building adblockers invisible to anti-adblockers and presents an important advancement in the rapidly escalating adblocking arms race.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under grant numbers 1719147, 1715152, and 1815131.

REFERENCES

[1] The state of the blocked web 2017 Global Adblock Report. PageFair. https://pagefair.com/downloads/2017/01/PageFair-2017-Adblock-Report.pdf, 2017.

[2] Uk publishers lose nearly 3bn in revenue annually due to adblocking, 2017.

[3] Abp anti-circumvention filter list. https://github.com/abp-filters/abp-filters-anti-cv, 2018.

[4] Adblock warning removal list. https://easylist-downloads.adblockplus.org/antiadblockfilters.txt, 2018.

[5] Anti-adblock killer: Don't touch my adblocker! https://reek.github.io/anti-adblock-killer/, 2018.

[6] Blink - the chromium projects. https://www.chromium.org/blink, 2018.

[7] Easylist forum. https://forums.lanik.us/, 2018.

[8] Easylist: Overview. https://easylist.to/, 2018.

[9] Issues with yavli advertising. https://easylist.to/2015/08/19/issues-with-yavli-advertising.html, 2018.

[10] libadblockplus: A c++ library offering the core functionality of adblock plus. https://github.com/adblockplus/libadblockplus, 2018.

[11] Native advertising: A guide for businesses. https://www.ftc.gov/tips-advice/business-center/guidance/native-advertising-guide-businesses, 2018.

[12] Navigation timing api - web apis | mdn. https://developer.mozilla.org/en-US/docs/Web/API/Navigation_timing_API, 2018.

[13] Optimize performance under varying network conditions | tools for web developers | google developers. https://developers.google.com/web/tools/chrome-devtools/network-performance/network-conditions, 2018.

[14] Performancetiming.loadeventstart - web apis | mdn. https://developer.mozilla.org/en-US/docs/Web/API/PerformanceTiming/loadEventStart, 2018.

[15] Performancetiming.responsestart - web apis | mdn. https://developer.mozilla.org/en-US/docs/Web/API/PerformanceTiming/responseStart, 2018.

[16] Render-tree construction, layout, and paint. https://developers.google.com/web/fundamentals/performance/critical-rendering-path/render-tree-construction, 2018.

[17] Speed index - webpagetest documentation. https://sites.google.com/a/webpagetest.org/docs/using-webpagetest/metrics/speed-index, 2018.

[18] Subresourcefilter in chromium source code. https://cs.chromium.org/chromium/src/components/subresource_filter/, 2018.

[19] V8 javascript engine. https://v8.dev/, 2018.

[20] V8stacktraceimpl in chromium source code. https://cs.chromium.org/chromium/src/v8/src/inspector/v8-stack-trace-impl.h, 2018.

[21] visibility - css: Cascading style sheets | mdn. https://developer.mozilla.org/en-US/docs/Web/CSS/visibility, 2018.

[22] Web apis | mdn. https://developer.mozilla.org/en-US/docs/Web/API, 2018.

[23] Yavli filters issues - easylist forum. https://forums.lanik.us/viewtopic.php?f=64&t=36091, 2018.

[24] Blob - web apis | mdn. https://developer.mozilla.org/en-US/docs/Web/API/Blob, 2019.

[25] Media source extensions api - web apis | mdn. https://developer.mozilla.org/en-US/docs/Web/API/Media_Source_Extensions_API, 2019.

[26] preload.js in adblock plus extension that injects css selectors into web pages. https://github.com/adblockplus/adblockpluschrome/blob/c742bcc37b459c03bd564aea941ef6f05834e7fd/include.preload.js#L259, 2019.

[27] Q. Chen and A. Kapravelos. Mystique: Uncovering information leakage from browser extensions. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1687–1700. ACM, 2018.

[28] G. Hong, Z. Yang, S. Yang, L. Zhang, Y. Nan, Z. Zhang, M. Yang, Y. Zhang, Z. Qian, and H. Duan. How you get shot in the back: A systematical study about cryptojacking in the real world. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1701–1713. ACM, 2018.

[29] U. Iqbal, Z. Shafiq, and Z. Qian. The ad wars: retrospective measurement and analysis of anti-adblock filter lists. In *Proceedings of the 2017 Internet Measurement Conference*, pages 171–183. ACM, 2017.

[30] U. Iqbal, Z. Shafiq, P. Snyder, S. Zhu, Z. Qian, and B. Livshits. Adgraph: A machine learning approach to automatic and effective adblocking. *arXiv preprint arXiv:1805.09155*, 2018.

[31] R. Karim, F. Tip, A. Sochurkova, and K. Sen. Platform-independent dynamic taint analysis for javascript. *IEEE Transactions on Software Engineering*, 2018.

[32] B. Li, P. Vadrevu, K. H. Lee, and R. Perdisci. Jsgraph: Enabling reconstruction of web attacks via efficient tracking of live in-browser javascript executions. In *25th Annual Network and Distributed System Security Symposium*, 2018.

[33] M. H. Mughees, Z. Qian, and Z. Shafiq. Detecting anti ad-blockers in the wild. *Proceedings on Privacy Enhancing Technologies*, 2017(3):130–146, 2017.

[34] N. Nikiforakis, W. Joosen, and B. Livshits. Privaricator: Deceiving fingerprinters with little white lies. In *Proceedings of the 24th International Conference on World Wide Web*, pages 820–830. International World Wide Web Conferences Steering Committee, 2015.

[35] X. Pan, Y. Cao, and Y. Chen. I do not know what you visited last summer: Protecting users from third-party web tracking with trackingfree browser. In *Proceedings of the 2015 Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA*, 2015.

[36] E. Rudd, A. Rozsa, M. Gunther, and T. Boult. A survey of stealth malware: Attacks, mitigation measures, and steps toward autonomous open world solutions. *IEEE Communications Surveys & Tutorials*, 19(2):1145–1172, 2017.

[37] G. Storey, D. Reisman, J. Mayer, and A. Narayanan. The future of ad blocking: An analytical framework and new techniques. *arXiv preprint arXiv:1705.08568*, 2017.

[38] W. Wang, Y. Zheng, X. Xing, Y. Kwon, X. Zhang, and P. Eugster. Webranz: web page randomization for better advertisement delivery and web-bot prevention. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 205–216. ACM, 2016.

[39] S. Zhu, X. Hu, Z. Qian, Z. Shafiq, and H. Yin. Measuring and disrupting anti-adblockers using differential execution analysis. NDSS, 2018.

GraphVite: A High-Performance CPU-GPU Hybrid System for Node Embedding

Zhaocheng Zhu
Mila - Québec AI Institute
Université de Montréal
zhaocheng.zhu@umontreal.ca

Shizhen Xu
Tsinghua University
xsz12@mails.tsinghua.edu.cn

Meng Qu
Mila - Québec AI Institute
Université de Montréal
meng.qu@umontreal.ca

Jian Tang
Mila - Québec AI Institute
HEC Montréal
CIFAR AI Research Chair
jian.tang@hec.ca

ABSTRACT

Learning continuous representations of nodes is attracting growing interest in both academia and industry recently, due to their simplicity and effectiveness in a variety of applications. Most of existing node embedding algorithms and systems are capable of processing networks with hundreds of thousands or a few millions of nodes. However, how to scale them to networks that have tens of millions or even hundreds of millions of nodes remains a challenging problem. In this paper, we propose *GraphVite*, a high-performance CPU-GPU hybrid system for training node embeddings, by co-optimizing the algorithm and the system. On the CPU end, augmented edge samples are parallelly generated by random walks in an online fashion on the network, and serve as the training data. On the GPU end, a novel parallel negative sampling is proposed to leverage multiple GPUs to train node embeddings simultaneously, without much data transfer and synchronization. Moreover, an efficient collaboration strategy is proposed to further reduce the synchronization cost between CPUs and GPUs. Experiments on multiple real-world networks show that *GraphVite* is super efficient. It takes only about one minute for a network with 1 million nodes and 5 million edges on a single machine with 4 GPUs, and takes around 20 hours for a network with 66 million nodes and 1.8 billion edges. Compared to the current fastest system, *GraphVite* is about 50 times faster without any sacrifice on performance.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; *Parallel algorithms.*

KEYWORDS

Unsupervised node embedding, parallel processing, scalability, graphics processing unit

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313508

ACM Reference Format:
Zhaocheng Zhu, Shizhen Xu, Meng Qu, and Jian Tang. 2019. GraphVite: A High-Performance CPU-GPU Hybrid System for Node Embedding. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313508

1 INTRODUCTION

Networks are ubiquitous in the real world. Examples like social networks [19], citation networks [27], protein-protein interaction networks [30] and many more cover a wide range of applications. In network analysis, it is critical to have effective representations for nodes, as these representations largely determine the performance of many downstream tasks. Recently, there is a growing interest in unsupervised learning of continuous node representations, which is aimed at preserving the structure of networks in a low-dimensional space. This kind of approaches has been proven successful in various applications, such as node classification [23], link prediction [14], and network visualization [31].

Many works have been proposed on this stream, including DeepWalk [23], LINE [32], and node2vec [8]. These methods learn effective node embeddings by predicting the neighbors of each node and can be efficiently optimized by asynchronous stochastic gradient descent (ASGD) [25]. On a single machine with multi-core CPUs, they are capable of processing networks with one or a few millions of nodes. Given that real-world networks easily go to tens of millions nodes and nearly billions of edges, how to adapt node embedding methods to networks of such large scales remains very challenging. One may think of exploiting computer clusters for training large-scale networks. However, it is a non-trivial task to extend existing methods to distributed settings. Even if distributed algorithms are available, the cost of large CPU clusters is still prohibitive for many users. Therefore, we are wondering whether it is possible to scale node embedding methods to very large networks on a single machine, which should be particularly valuable for common users.

Inspired by the recent success of training deep neural networks with GPUs [5, 13], we would like to utilize such highly parallel hardware to accelerate the training of node embeddings. However, directly adopting GPUs for node embedding could be inefficient, since the sampling procedure in node embedding requires excessive

random memory access on the network structure, which is at the disadvantage of GPUs. Compared to GPUs, CPUs are much more capable of performing random memory access. Therefore, it would be wise to use both CPUs and GPUs for training node embeddings. Along this direction, a straightforward solution is to follow the mini-batch stochastic gradient descent (mini-batch SGD) paradigm utilized in existing deep learning frameworks (e.g. TensorFlow [1] and PyTorch [22]). Different from deep neural networks, the training of node embeddings involves much more memory access per computation. As a result, mini-batch SGD would suffer from severe memory latency on the bus before it benefits from fast GPU computation. Therefore, other than mini-batch SGD, we need to design a system that leverages distinct advantages of CPUs and GPUs and uses them collaboratively to train node embeddings efficiently.

Overall, the main challenges of building an efficient node embedding system with GPUs are:

(1) **Limited GPU Memory** The parameter matrices of node embeddings are quite large while the memory of a single GPU is very small. Modern GPUs usually have a capacity of 12GB or 16GB.
(2) **Limited Bus Bandwidth** The bandwidth of the bus is much slower than the computation speed of GPUs. There will be severe latency if GPUs exchange data with the main memory frequently.
(3) **Large Synchronization Cost** A lot of data are transferred between CPUs and GPUs. Both the CPU-GPU or inter-GPU synchronizations are very costly.

In this paper, we propose a high-performance CPU-GPU hybrid system called *GraphVite* for training node embeddings on large-scale networks. *GraphVite* takes full advantages of CPUs and GPUs by co-optimizing the node embedding algorithm and the system. Specifically, we observe that existing node embedding methods typically consist of two stages, i.e. network augmentation and embedding training. In the first stage, an augmented network is constructed by random walks on the original network, and positive edges are sampled on the augmented network. In the second stage, node embeddings are trained according to the samples generated in the previous stage. Since the augmentation stage involves excessive random access, we resort to CPUs for this part. The training stage is assigned to GPUs as it is mainly composed of matrix computation.

In *GraphVite*, the above challenges are addressed by three components, namely parallel online augmentation, parallel negative sampling and collaboration strategy. In parallel online augmentation, CPUs augment the network with random walks and generate edge samples in an online fashion. In parallel negative sampling, the edge samples are organized into a grid sample pool, where each block corresponds to a subset of the network. Then GPUs iteratively fetch orthogonal blocks and their corresponding embeddings in each episode. Because GPUs do not share any embeddings, multiple GPUs can perform gradient updates with negative sampling in its own subset simultaneously. With such a design, the problem of limited GPU memory is solved as each GPU only stores the subset of node embeddings corresponding to the current sample block. The problem of limited bus bandwidth is mitigated since model parameters are transferred only when GPUs change their blocks. No inter-GPU synchronization is needed and CPU-GPU synchronization is only needed at the end of each episode. The collaboration strategy further reduces the synchronization cost between CPUs and GPUs on the sample pool.

We evaluate *GraphVite* on 4 real-world networks of different scales. On a single machine with 4 Tesla P100 GPUs, our system only takes one minute to train a network with 1 million nodes and 5 million edges. Compared to the current fastest system [32], *GraphVite* is 51 times faster and does not sacrifice any performance. On a network with 66 million nodes and 1.8 billion edges, *GraphVite* takes only around 20 hours to finish training. We also investigate the speed of *GraphVite* under different hardware configurations. Even on economic GPUs like GeForce GTX 1080, *GraphVite* is able to achieve a speedup of 29 times compared to the current fastest system.

Organization Section 2 reviews existing state-of-the-art node embedding methods and points out the challenges of extending these methods to GPUs. Section 3 introduces our proposed system in details. We present our experiments in Section 4, followed by extensive ablation studies in Section 5. Section 6 summarizes the related work, and we conclude this paper in Section 7.

2 PRELIMINARIES

In this section, some preliminary knowledge is introduced. We first review existing state-of-the-art node embedding methods, followed by a discussion on the main challenges of extending these methods to GPUs.

2.1 Node Embedding Methods Review

Given a network $G = (V, E)$, the goal of node embedding is to learn a low-dimensional representation for each node. The learned embeddings are expected to capture the structure of the network. Towards this goal, most existing methods train node embeddings to distinguish the edges in E (i.e. positive edges) from some randomly sampled node pairs (i.e. negative edges). In other words, edges are essentially utilized as training data. Since many real-world networks are extremely sparse, most existing embedding methods conduct random walks on the original network to introduce more connectivity. Specifically, they connect nodes within a specified distance on a random walk path as additional positive edges. For example, LINE [32] uses a breadth-first search strategy on low-degree nodes, while DeepWalk [23] uses a depth-first search strategy for all nodes. Node2vec [8] developed a mixture of the above two strategies.

Once the network is augmented, node embeddings are trained on samples from the augmented network. Typically, the embeddings are encoded in two sets, namely **vertex** embedding matrix and **context** embedding matrix. For an edge sample (u, v), the dot product of $\mathbf{vertex}[u]$ and $\mathbf{context}[v]$ is computed to predict whether the sample is a positive edge. This encourages neighbor nodes to have close embeddings, whereas distant nodes will have very different embeddings.

Overall, the computation procedures of these node embedding methods can be divided into two stages: **network augmentation** and **embedding training**. Algorithm 1 summarizes the general framework of existing node embedding methods. Note that the first

stage can be easily parallelized, and the second stage can be parallelized via asynchronous SGD. In most existing node embedding systems, these two stages are executed in a sequential order, with each stage parallelized by a bunch of CPU threads.

Algorithm 1 General framework of node embedding

$E' \leftarrow E$
for $v \in V$ **do** ▷ parallelizable
  for $u \in \text{WALK}(v)$ **do**
    $E' \leftarrow E' \cup \{(v, u, \text{WEIGHT}(v, u))\}$
  end for
end for

for each iteration **do** ▷ parallelizable
  $v, u \leftarrow \text{EDGESAMPLING}(E')$
  $\text{TRAIN}(\textbf{vertex}[v], \textbf{context}[u], label = 1)$
  for $u' \in \text{NEGATIVESAMPLING}(V)$ **do**
    $\text{TRAIN}(\textbf{vertex}[v], \textbf{context}[u'], label = 0)$
  end for
end for

2.2 Challenges for Hybrid Node Embedding System

Inspired by the recent success of training neural networks with GPUs, we are interested in building a node embedding system by leveraging the power of GPUs, which can benefit the embedding training stage. Since the first stage of network augmentation involves extensive memory random access, CPUs are more suitable for this stage. As a result, we desire to develop a hybrid CPU-GPU system for training node embeddings. A common approach for a hybrid machine learning system is the mini-batch SGD paradigm, which is widely adopted in existing deep learning frameworks such as TensorFlow [1] and PyTorch [22]. In mini-batch SGD, model parameters are stored on GPUs and training data is iteratively passed to GPUs in batches.

However, mini-batch SGD cannot be applied directly to node embedding on large networks. Take a scale-free network with 50 million nodes and 1 billion edges as an example. (1) The size of the augmented network goes to 373 GB large, which may overwhelm the memory of most servers. We need to figure out a way to generate the augmented network and edge samples on the fly. (2) The embedding matrices are much larger than the parameter matrices in deep neural networks. Either **vertex** or **context** matrix consumes 23.8 GB memory, which is beyond the memory limit of any single GPU. As a result, both the **vertex** and **context** embedding matrices have to be stored in the main memory, and transferred to the GPUs in small parts during training. See Table 1 for a detailed analysis of the memory cost.

For the second problem, while transferring parameters from CPUs and GPUs sounds feasible, it will become a bottleneck if we consider the bus bandwidth between CPUs and GPUs. Consider training d-dimensional embeddings with n edge samples. There is $O(nd)$ computation workload and also $O(nd)$ memory access if the samples are not overlapped with each other. Since the computation speed of GPUs is way faster than the speed of bus transfer, the entire system will be bounded by the speed of parameter transfer from CPUs to GPUs severely. Indeed, such a system is even worse than its CPU parallel counterpart, which is verified in our experiments (see Table 3).

	Size	Example	Memory cost
nodes	$\|V\|$	$5 * 10^7$	191 MB
edges	$\|E\|$	$1 * 10^9$	7.45 GB
augmented edges	$\|E'\|$	$5 * 10^{10}$	373 GB
vertex	$\|V\| \times d$	$5 * 10^7 \times 128$	23.8 GB
context	$\|V\| \times d$	$5 * 10^7 \times 128$	23.8 GB

Table 1: Memory cost of node embedding on a scale-free network with 50 million nodes and 1 billion edges.

Another challenge in a hybrid system is the large synchronization cost. Since the system is distributed on multiple CPUs and GPUs, there is necessary data (e.g. parameters and edge samples) shared across sub tasks. A trivial but safe solution is to synchronize the shared data frequently, which will result in huge synchronization cost. To achieve high speed performance, the system should reduce shared data as much as possible, and use a collaboration strategy to minimize synchronization cost between devices.

3 *GRAPHVITE*: A HYBRID CPU-GPU NODE EMBEDDING SYSTEM

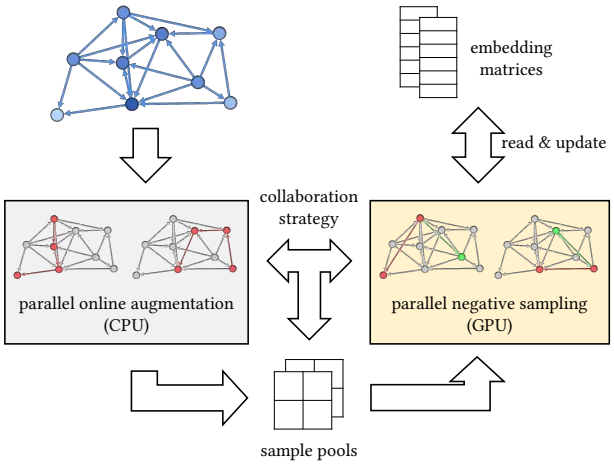


Figure 1: Overview of our hybrid system. The gray and yellow boxes correspond to the stages of network augmentation and embedding training respectively. The former is performed by parallel online augmentation on CPUs, while the latter is performed by parallel negative sampling on GPUs. The two stages are executed asynchronously with our collaboration strategy.

In this section, we introduce a high-performance hybrid CPU-GPU system called *GraphVite* for training node embeddings. Our

system leverages distinct advantages of CPUs and GPUs and addresses the above three challenges. Specifically, we propose a parallel online augmentation for efficient network augmentation on CPUs. We introduce a parallel negative sampling to cooperate multiple GPUs for embedding training. A collaboration strategy is also proposed to reduce the synchronization cost between CPUs and GPUs.

3.1 Parallel Online Augmentation

As discussed in Section 2.1, the first stage of node embedding methods is to augment the original network with random walks. Since the augmented network is usually one or two magnitude larger than the original one, it is impossible to load it into the main memory if the original network is already very large. Therefore, we introduce a parallel online augmentation, which generates augmented edge samples on the fly without explicit network augmentation. Our method can be viewed as an online extension of the augmentation and edge sampling method used in LINE[32]. First, we draw a departure node with the probability proportional to the degree of each node. Then we perform a random walk from the departure node, and pick node pairs within a specific augmentation distance s as edge samples. Note that edge samples generated in the same random walk are correlated and may degrade the performance of optimization. Inspired by the experience replay technique widely used in reinforcement learning [15, 20], we collect edge samples into a sample pool, and shuffle the sample pool before transferring it to GPUs for embedding training. The proposed edge sampling method can be parallelized when each thread is allocated with an independent sample pool in advance. Algorithm 2 gives the process of parallel online augmentation in details.

Algorithm 2 Parallel Online Augmentation

```
1:  function ParallelOnlineAugmentation(num_CPU)
2:      for i ← 0 to num_CPU − 1 do                    ▷ paralleled
3:          pool[i] ← ∅
4:          while pool is not full do
5:              x ← DepartureSampling(G)
6:              for u, v ∈ RandomWalkSampling(x) do
7:                  if Distance(u, v) <= s then
8:                      pool.append((u, v))
9:                  end if
10:             end for
11:         end while
12:         pool[i] ← Shuffle(pool[i])
13:     end for
14:     return Concatenate(pool[·])
15: end function
```

Pseudo Shuffle While shuffling the sample pool is important to optimization, it slows down the network augmentation stage (see Table 7). The reason is that a general shuffle consists of lots of random memory access and cannot be accelerated by the CPU cache. The loss in speed will be even worse if the server has more than one CPU socket. To mitigate this issue, we propose a pseudo shuffle technique that shuffles correlated samples in a much more cache-friendly way and improves the speed of the system significantly. Note that most correlation comes from edge samples that share the source node or the target node in the same random walk. As such correlation occurs in a group of s samples for an augmentation distance s, we divide the sample pool into s continuous blocks, and scatter correlated samples into different blocks. For each block, we always append samples sequentially at the end, which can benefit a lot from CPU cache. The s blocks are concatenated to form the final sample pool.

3.2 Parallel Negative Sampling

In the embedding training stage, we divide the training task into small fragments and distribute them to multiple GPUs. The sub tasks are necessarily designed with little shared data to minimize the synchronization cost among GPUs. To see how model parameters can be distributed to multiple GPUs without overlap, we first introduce a definition of *ϵ-gradient exchangeable*.

Definition 1. ***ϵ-gradient exchangeable.*** *A loss function $L(X;\theta)$ is ϵ-gradient exchangeable on two sets of training data X_1, X_2 if for small $\epsilon \geq 0$, $\forall\theta_0 \in \Theta$ and $\forall\alpha \in \mathbb{R}^+$, exchanging the order of two gradient descent steps results in a vector difference with norm no more than ϵ.*

$$\begin{cases} \theta_1 \leftarrow \theta_0 - \alpha\nabla L(X_1;\theta_0) \\ \theta_2 \leftarrow \theta_1 - \alpha\nabla L(X_2;\theta_1) \end{cases} \tag{1}$$

$$\begin{cases} \theta_1' \leftarrow \theta_0 - \alpha\nabla L(X_2;\theta_0) \\ \theta_2' \leftarrow \theta_1' - \alpha\nabla L(X_1;\theta_1') \end{cases} \tag{2}$$

i.e. $\|\theta_2 - \theta_2'\| \leq \epsilon$ is true for the above equations.

Particularly, we abbreviate *0-gradient exchangeable* to *gradient exchangeable*. Due to the sparse nature of node embedding training, there are many sets that form *gradient exchangeable* pairs in the network. For example, for two edge sample sets $X_1, X_2 \subseteq E$, if they do not share any source nodes or target nodes, X_1 and X_2 are *gradient exchangeable*. Even if X_1 and X_2 share some nodes, they can still be *ϵ-gradient exchangeable* if the learning rate α and the number of iterations are bounded.

Based on the gradient exchangeability observed in node embedding, we propose a parallel negative sampling algorithm for the embedding training stage. For n GPUs, we partition rows of **vertex** and **context** into n partitions respectively (see the top-left corner of Figure 2). This results in an $n \times n$ partition grid for the sample pool, where each edge belongs to one of the blocks. In this way, any pair of blocks that does not share row or column is *gradient exchangeable*. Blocks in the same row or column are *ϵ-gradient exchangeable*, as long as we restrict the number of iterations on each block.

We define *episode* as the block-level step used in parallel negative sampling. During each episode, we send n orthogonal blocks and their corresponding **vertex** and **context** partitions to n GPUs respectively. Each GPU then updates its own embedding partitions with ASGD. Because these blocks are mutually *gradient exchangeable* and do not share any row in the parameter matrices, multiple GPUs can perform ASGD concurrently without any synchronization. At the end of each episode, we gather the updated parameters from all GPUs and assign another n orthogonal blocks. Here *ϵ-gradient exchangeable* is controlled by the number of total samples

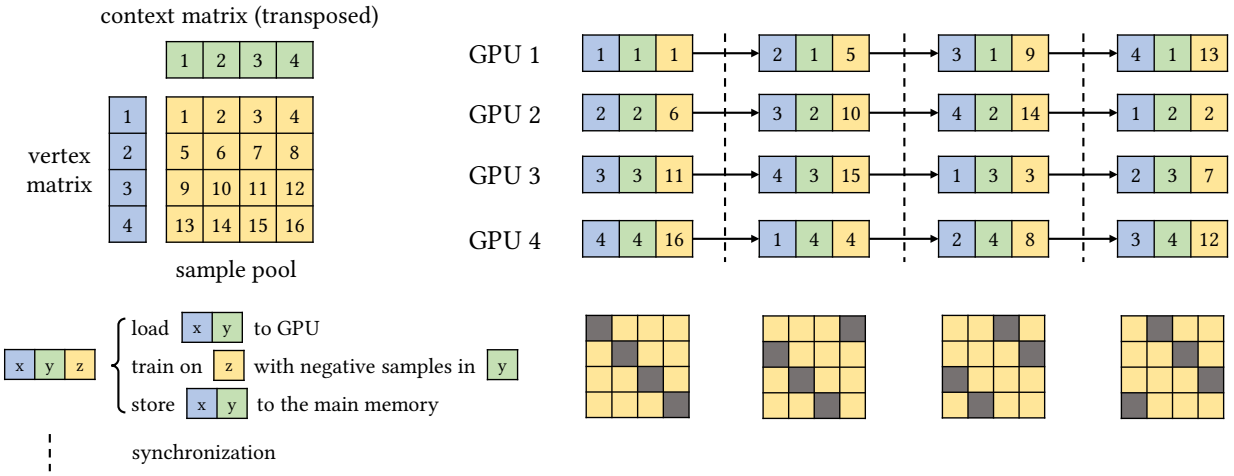


Figure 2: Illustration of parallel negative sampling on 4 GPUs. During each episode, GPUs take orthogonal blocks from the sample pool. Each GPU trains embeddings with negative samples drawn from its own context nodes. Synchronization is only needed between episodes.

in n orthogonal blocks, which we define as *episode size*. The smaller episode size, the better *ϵ-gradient exchangeable* we will have for embedding training. However, smaller episode size will also induce more frequent synchronization. Hence the episode size is tuned so that there is a good trade off between the speed and *ϵ-gradient exchangeable* (see Section 5.3). Figure 2 gives an example of parallel negative sampling with 4 partitions.

Typically, node embedding methods sample negative edges from all possible nodes. However, it could be very time-consuming if GPUs have to communicate with each other to get the embeddings of their negative samples. To avoid this cost, we restrict that negative samples can only be drawn from the **context** rows on the current GPU. Though this seems a little problematic, we find it works well in practice. An intuitive explanation is that with parallel online augmentation, every node is likely to have positive samples with nodes from all context partitions. As a result, every node can potentially form negative samples with all possible nodes.

Note that although we demonstrate with the number of partitions equal to n, the parallel negative sampling can be easily generalized to cases with any number of partitions greater than n, simply by processing the orthogonal blocks in subgroups of n during each episode. Algorithm 3 illustrates the hybrid system for multiple GPUs.

Algorithm 3 Parallel Negative Sampling

```
function ParallelNegativeSampling(num_GPU)
    vertex_partitions ← Partition(vertex)
    context_partitions ← Partition(context)
    while not converge do
        pool ← ParallelOnlineAugmentation(num_CPU)
        block[·][·] ← Redistribute(pool)
        for offset ← 0 to num_GPU − 1 do
            for i ← 0 to num_GPU − 1 do                ▷ paralleled
                vid ← i
                cid ← (i + offset)  mod num_GPU
                send vertex_partitions[vid] to GPU i
                send context_partitions[cid] to GPU i
                train block[vid][cid] on GPU i
                receive vertex_partitions[vid] from GPU i
                receive context_partitions[cid] from GPU i
            end for
        end for
    end while
end function
```

3.3 Collaboration Strategy

Our parallel negative sampling enables different GPUs to train node embeddings concurrently, with only synchronization required between episodes. However, it should be noticed that the sample pool is also shared between CPUs and GPUs. If they synchronize on the sample pool, then only workers of the same stage can access the pool at the same time, which means hardware is idle for half of the time. To eliminate this problem, we propose a collaboration strategy to reduce the synchronization cost. We allocate two sample pools in the main memory, and let CPUs and GPUs always work on different pools. CPUs first fill up a sample pool and pass it to GPUs. After that, parallel online augmentation and parallel negative sampling are performed concurrently on CPUs and GPUs respectively. The two pools are swapped when CPUs fill up a new pool. Figure 1 illustrates this procedure. With the collaboration strategy, the synchronization cost between CPUs and GPUs is reduced and the speed of our hybrid system is almost doubled.

3.4 Discussion

Here we further discuss some practical details of our hybrid system.

Batched Transfer In parallel negative sampling, the sample pool is assigned to GPUs by block, which is sometimes very large for the

Dataset	YOUTUBE	FRIENDSTER-SMALL	HYPERLINK-PLD	FRIENDSTER
$\|V\|$	1,138,499	7,944,949	39,497,204	65,608,376
$\|E\|$	4,945,382	447,219,610	623,056,313	1,806,067,142
Evaluation Task	47-class node classification	100-class node classification	link prediction	100-class node classification

Table 2: Statistics of the datasets used in experiments

memory of a GPU. Instead of copying the whole sample block to a GPU, we transfer the sample block by a small granularity. In this way, the memory cost of edge samples on GPUs becomes negligible.

Bus Usage Optimization When the number of partitions equals the number of GPUs, we can further optimize the bus usage by fixing the context partition for each GPU. In this way, we save the transfer of *context* matrix and further reduce the synchronization cost between CPUs and GPUs.

Single GPU Case Although parallel negative sampling is proposed for multiple GPUs, our hybrid system is compatible with a single GPU. Typically a GPU can hold at most 12 million node embeddings. So a single GPU is sufficient for training node embeddings on networks that contain no more than 12 million nodes.

4 EXPERIMENTS

In this section, we verify the effectiveness and efficiency of *GraphVite*. We first evaluate our system on YOUTUBE, which is a large network widely used in the literature of node embeddings. Then we evaluate *GraphVite* on three larger datasets.

4.1 Datasets

We use the following datasets in our experiments. Statistics of these networks are summarized in Table 2.

- YOUTUBE [19] is a large-scale social network in the Youtube website. It contains 1 million nodes and 5 million edges. For some of the nodes, they have labels that represent the type of videos users enjoy.
- FRIENDSTER-SMALL [37] is a sub-graph induced by all the labeled nodes in FRIENDSTER. It has 8 million nodes and 447 million edges. The node labels in this network are the same as those in FRIENDSTER.
- HYPERLINK-PLD [16] is a hyperlink network extracted from the Web corpus [1]. We use the pay-level-domain aggregated version of the network. It has 43 million nodes and 623 million edges. This dataset does not contain any label.
- FRIENDSTER [37] is a very large social network in an online gaming site. It has 65 million nodes and 1.8 billion edges. Some nodes have labels that represent the group users join.

4.2 Compared Systems

We compare *GraphVite* with the following node embedding systems.

- LINE [32] [2] is a CPU parallel system based on C++. We parallel its network augmentation stage for fair comparison with other methods.
- DeepWalk [23] [3] is a CPU parallel system based on Python and gensim [26].
- node2vec [8] [4] is another CPU parallel system based on Python and gensim [26].
- LINE in OpenNE [2] [5] is a GPU system based on the Python and TensorFlow [1].

Although DeepWalk and node2vec are implemented in Python, their computation is fully paralleled by Cython code without GIL, which is comparable to C++ implementations.

4.3 Implementation Details

Our implementation generally follows the open source codes of LINE [2] and DeepWalk [3]. We adopt the asynchronous SGD [25] in GPU training, and leverage the on-chip shared memory of GPU for fast forward and backward propagation. We also utilize the alias table trick [8, 32] to boost parallel online augmentation and parallel negative sampling.

Our hyperparameters are set according to the settings in LINE [32] and DeepWalk [23]. We treat networks as undirected graphs. During the network augmentation stage, we sample random walks with a length of 40 edges. We use a degree-guided strategy to partition both **vertex** and **context** matrices. More specifically, we first sort nodes by their degrees and then assign them into different partitions in a zig-zag fashion, as illustrated in Figure 3. We tune the episode size to maximize the speed of our hybrid system. During the embedding training stage, negative samples are sampled with a probability proportional to the 3/4 power of the node degrees. For each positive sample, we draw 1 negative sample and scale the gradient of the negative sample by 5 to match the gradient scale in LINE. We follow the initial learning rate of 0.025 and the linear learning rate decay mechanism in LINE and DeepWalk. We only adopt the *O3* optimization in *g++* and *nvcc*. We do not use any non-standard optimizations or low precision training [17, 39], though they may further improve the speed of our system.

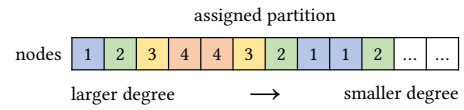


Figure 3: Degree-guided node and context partition strategy in the case of 4 partitions.

We define a training epoch as training $|E|$ positive edge samples. For YOUTUBE, the length of random walk is set to 5, and the total

[1] http://commoncrawl.org/
[2] https://github.com/tangjianpku/LINE
[3] https://github.com/phanein/deepwalk
[4] https://github.com/aditya-grover/node2vec
[5] https://github.com/thunlp/OpenNE

Method	CPU threads	GPU	Training time	Preprocessing time
LINE [32]	20	-	1.24 hrs	17.4 mins
DeepWalk [23]	20	-	1.56 hrs	14.2 mins
node2vec [8]	20	-	47.7 mins	25.9 hrs
LINE in OpenNE [2]	1	1	> 1 day	2.14 mins
GraphVite	6	1	**3.98 mins**(18.7×)	**7.37 s**
GraphVite	24	4	**1.46 mins**(50.9×)	**16.0 s**

Table 3: Results of time of different systems on YOUTUBE. The preprocessing time refers to all the overhead before training, including network input and offline network augmentation. Note the preprocessing time of OpenNE is not comparable since it does not have the network augmentation stage. The speedup ratio of *GraphVite* is computed with regard to LINE, which is the current fastest system.

	% Labeled Nodes	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1(%)	LINE[32]	32.98	36.70	38.93	40.26	41.08	41.79	42.28	42.70	43.04	43.34
	LINE[32]+augmentation	36.78	40.37	42.10	43.25	43.90	44.44	44.83	45.18	45.50	45.67
	DeepWalk[32]	**39.68**	41.78	42.78	43.55	43.96	44.31	44.61	44.89	45.06	45.23
	GraphVite	39.19	**41.89**	**43.06**	**43.96**	**44.53**	**44.93**	**45.26**	**45.54**	**45.70**	**45.86**
Macro-F1(%)	LINE[32]	17.06	21.73	25.28	27.36	28.50	29.59	30.43	31.14	31.81	32.32
	LINE[32]+augmentation	22.18	27.25	29.87	31.88	32.86	33.73	34.50	35.15	35.76	36.19
	DeepWalk[32]	**28.39**	**30.96**	**32.28**	**33.43**	**33.92**	34.32	34.83	35.27	35.54	35.86
	GraphVite	25.61	29.46	31.32	32.70	33.81	**34.59**	**35.27**	**35.82**	**36.14**	**36.49**

Table 4: Results of node classification on YOUTUBE

number of training epochs is set to 4,000. For the other 3 datasets, the length of random walks is set to 2, and the total number of training epochs is set to 2,000 since they are denser. The dimension of node embeddings is set to 128 except on FRIENDSTER, where we use 96. For other hyperparameters, we follow their default values in previous works. For fair comparison, we report the training time of all methods with the same number of training epochs. We parallel the network augmentation in LINE. For DeepWalk, we store the random walks in memory, which is the fastest setting.

4.4 Results on YOUTUBE

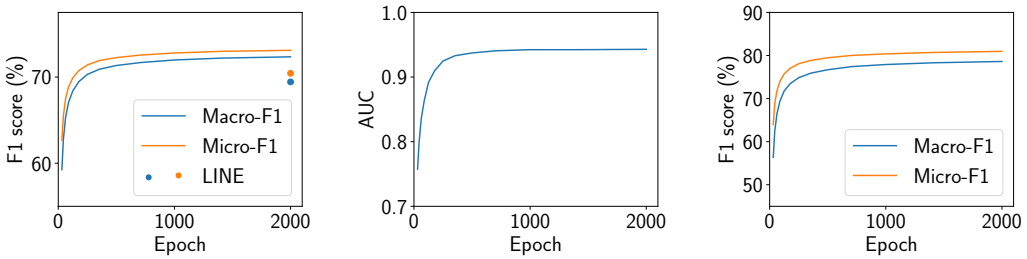
We first evaluate our hybrid system on the widely-used YOUTUBE dataset. We compare the speed and performance of *GraphVite* with existing systems of node embedding. For existing systems, we replicate their parallel implementations and report their training time under the same number of training epochs. Table 3 presents the speed of different systems. Among all existing systems, LINE [32] takes the minimal total time to run. However, the GPU implementation of LINE in OpenNE is even worse than its CPU counterpart, possibly due to the mini-batch SGD paradigm it uses. Compared to the current fastest system, LINE, *GraphVite* is much more efficient. With 4 GPUs, our system finishes training node emebdding on a million-scale network in only one and a half minutes. Even on a single GPU, *GraphVite* takes no more than 4 minutes and is still 19 times faster than LINE.

One may be curious about the performance of node embeddings learned by *GraphVite*. Therefore, we compare the performance of *GraphVite* with existing systems on the standard task of multi-label node classification. Note that normalizing the embeddings or not yields different trade off between Micro-F1 and Macro-F1 metrics. For fair comparison, we follow the practice in [32] and train one-vs-rest linear classifiers over the normalized node embeddings. Table 4 summarizes the performance over different percentages of training data. It is observed that *GraphVite* achieves the best or competitive results in most settings, showing that *GraphVite* does not sacrifice any performance. In some small percentage cases, *GraphVite* falls a little behind DeepWalk. This is because *GraphVite* uses negative sampling for optimization, while DeepWalk uses both hierarchical softmax and negative sampling, which could be more robust to few labeled data.

4.5 Results on Larger Datasets

	FRIENDSTER-SMALL	HYPERLINK-PLD	FRIENDSTER
1 GPU	8.78 hrs	-	-
4 GPU	2.79 hrs	5.36 hrs	20.3 hrs

Table 5: Results of time on larger datasets. We only evaluate HYPERLINK-PLD and FRIENDSTER with 4 GPUs since their embedding matrices cannot fit into the memory of a single GPU.



(a) Friendster-small (b) Hyperlink-PLD (c) Friendster

Figure 4: Performance curves of *GraphVite* on larger datasets. For Friendster, we plot the results of LINE for reference. The other systems cannot solve any of these datasets within a week.

	Parallel Online Augmentation	Parallel Negative Sampling (4 GPUs)	Collaboration Strategy	Micro-F1	Macro-F1	Training time
Single GPU baseline				35.26	20.38	8.61 mins
	✓			41.48	29.80	6.35 mins
		✓		34.38	19.81	2.66 mins
	✓	✓		41.75	29.30	2.24 mins
GraphVite	✓	✓	✓	41.89	29.46	**1.46 mins**

Table 6: Ablation of main components in *GraphVite*. Note that the baseline has the same GPU implementation with *GraphVite* and parallel edge sampling on CPU. The baseline should be regarded as a very strong one.

To demonstrate the scalability of *GraphVite*, we further test *GraphVite* on three larger networks. We learn the node embeddings of Friendster-small with 1 GPU and 4 GPUs. For Hyperlink-PLD and Friendster, since their embedding matrices cannot fit into the memory of a single GPU, we only evaluate them with 4 GPUs. Table 5 gives the training time of *GraphVite* on these datasets. The training time of baseline systems is not reported here, as all existing systems cannot solve such large networks in a week, except LINE [32] on Friendster-small. Compared to them, *GraphVite* takes less than 1 day to train node embeddings on the largest dataset Friendster with 1.8 billion edges, showing that *GraphVite* can be an efficient tool for analyzing billion-scale networks.

We also evaluate the performance of the node embeddings on these datasets. For Friendster-small and Friendster, we test their node embeddings on multi-label node classification. The test set is built on the top-100 communities of Friendster and has a total of 39,679 nodes. We do not normalize the learned embeddings during evaluation, and report Macro-F1 and Micro-F1 based on 2% labeled data. For Hyperlink-PLD, we adopt link prediction as the evaluation task since node labels are not available. We randomly exclude 0.01% edges from the training set, and combine them with the same number of uniformly sampled negative edges to form a test set. Each edge sample is scored by the cosine similarity of two node embeddings. We report the AUC metric for link prediction. Figure 4 presents the performance of *GraphVite* over different training epochs on these datasets. On Friendster-small, we also plot the performance of LINE for reference. Due to the long training time, we only report the performance of LINE by the end of all training epochs. It is observed that *GraphVite* converges on all these datasets. On the Friendster-small dataset, *GraphVite* significantly outperforms LINE. On the Hyperlink-PLD, we get an AUC of 0.943. On Friendster, the Micro-F1 reaches about 81.0%. All the above observations verify the performance of our system.

5 ABLATION STUDY

To have a more comprehensive understanding of different components in *GraphVite*, we conduct several ablation experiments. For intuitive comparison, we evaluate these experiments on the standard Youtube dataset. We only report performance results based on 2% labeled data due to space limitation. All the speedup ratios are computed with respect to LINE [32].

5.1 What is the contribution of each main component?

In the *GraphVite*, parallel online augmentation, parallel negative sampling, and the collaboration strategy are the main components in the sytem. Here we study how these components contribute to the performance of our system. We compare *GraphVite* with a strong baseline system with single GPU. Specifically, the baseline has the same GPU implementation as *GraphVite*, while it uses the standard parallel edge sampling instead of parallel online augmentation, and executes two stages sequentially.

Table 6 shows the results of this ablation. Compared to the baseline, we notice that parallel online augmentation helps improve the quality of node embeddings, since it introduces more connectivity

to the sparse network. Besides, parallel online augmentation also accelerates the system a little, as it reuses nodes and reduces the amortized cost of each sample. With parallel negative sampling, we are able to employ multiple GPUs for training, and the speed is boosted by about 3 times. Moreover, the collaboration strategy even improves the speed and does not impact the performance.

5.2 Is it necessary to perform pseudo shuffle?

In parallel online augmentation, *GraphVite* performs pseudo shuffle to decorrelate the augmented edge samples, while some existing systems [8, 23] do not shuffle their samples. We compare the proposed pseudo shuffle with three baselines, including no shuffle, a full random shuffle and an index mapping algorithm. The index mapping algorithm preprocesses a random mapping on the indexes of samples and saves the time of computing random variables.

Table 7 gives the results of different shuffle algorithms on a single GPU. It is observed that all shuffle algorithms are about 1 percent better than the no shuffle baseline. However, different shuffle algorithms vary largely in their speed. Compared to the no shuffle baseline, the random shuffle and index mapping algorithms slow down the system by several times, while our pseudo shuffle has only a little overhead. Therefore, we conclude that pseudo shuffle is the best practice considering both speed and performance.

Shuffle algorithm	Micro-F1(%)	Training time
None	40.41	3.60 mins
Random shuffle	41.61	17.1 mins
Index mapping	41.21	12.1 mins
Pseudo shuffle	**41.52**	**3.98 mins**

Table 7: Results of performance and speed by different shuffle algorithms. The proposed pseudo shuffle algorithm achieves the best trade off between performance and speed.

5.3 What is a practical choice for episode size?

In parallel negative sampling, *GraphVite* relies on the property of gradient exchangeability to ensure its approximation to standard SGD. While the smaller episode size provides better exchangebility, it will increase the frequency of synchronization over rows of the embedding matrices, and thus slows down embedding training. To quantify such influence in speed and performance, we examine our system on 4 GPUs with different episode sizes.

Figure 5 plots the curves of speed and performance with respect to different episode sizes. On the performance side, we notice that the performance of *GraphVite* is insensitive to the choice of the episode size. Compared to the single GPU baseline, parallel negative sampling achieves competitive or slightly better results, probably due to the regularization effect introduced by partition. On the speed side, larger episode size achieves more speedup since it reduces the amortized burden of the bus. The speed drops at very large episode size, as there becomes only a few episodes in training. Therefore, we choose an episode size of $2 * 10^8$ edge samples for YOUTUBE. Generally, the best episode size is proportional to $|V|$, so one can set the episode size for other networks accordingly.

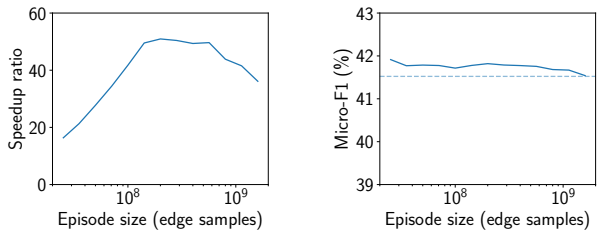


Figure 5: Speed and performance of *GraphVite* with respect to different episode sizes. The dashed line represents the single GPU baseline without parallel negative sampling.

5.4 What is the speedup w.r.t the numebr of CPUs and GPUs?

In *GraphVite*, both online augmentation and negative sampling can be parallelized on multiple CPUs or GPUs, and synchronization is only required between episodes. Therefore, our system should have great scalability. To verify that point, we investigate our system with different number of CPU and GPU. We change the number of GPU from 1 to 4, and vary the number of sampler per GPU from 1 to 5. The effective number of CPU threads is $\#GPU * (\#sampler\ per\ GPU + 1)$ as there is one scheduler thread for each GPU.

Figure 6 plots the speedup ratio with respect to different number of CPUs and GPUs. The speedup ratio almost forms a plane over both variables, showing that our system scales almost linearly to the hardware. Quantitatively, *GraphVite* achieves a relative speedup of 11× when the hardware is scaled to 20×. The speedup is about half of its theoretical maximum. We believe this is mainly due to the increased synchronization cost, as well as increased load on shared main memory and bus when we use more CPUs and GPUs.

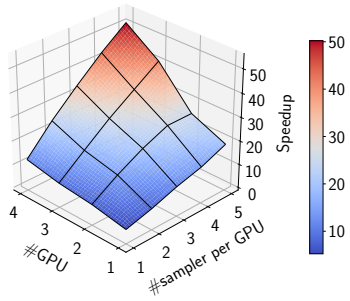


Figure 6: Results of speedup under different number of hardware. It is observed that the speedup is almost linear to the number of CPUs and GPUs.

5.5 Does the hardware configuration matter?

Up to now, all experiments are conducted on a server with Xeon E5 CPUs and Tesla P100 GPUs. One might wonder whether such a high performance depends on the specific hardware configuration. Therefore, we further test our system on an economic server with Core i7 CPUs and GTX 1080 GPUs.

Table 8 compares the results from two configurations. Different hardware does have difference in speed, but the gap is marginal. The time only increases to 1.6× when we move to the economic server. Note that this two configurations are almost the best and the worst in current machine learning servers, so one could expect a running time between these two configurations on his own hardware.

Hardware	CPU threads	GPU	Training time
Tesla P100 server	6	1	3.98 mins
	24	4	1.46 mins
GTX 1080 server	3	1	6.28 mins
	12	4	2.48 mins

Table 8: Training time of *GraphVite* under different hardware configurations. Generally *GraphVite* may take a time between these two configurations on most hardware.

6 RELATED WORK

Node embedding has been proven effective in a wide range of applications, such as node classification [10], link prediction [14], and network visualization [31]. Many different methods [4, 8, 23, 24, 32, 33, 35, 36] have been proposed to learn node embeddings that preserve the structure of networks from different aspects. Among them, DeepWalk [23], LINE [32] node2vec [8] and VERSE [33] are built on either edge or path samples of networks, which makes them the most scalable methods of all. Our work follows this stream and is related to these methods.

Node Embedding Algorithm Generally, node embedding algorithms consist of two stages, namely network augmentation and embedding training. The network augmentation stage is widely adopted in existing methods [4, 8, 23, 24, 32] to improve the performance of learned embeddings on sparse networks. DeepWalk [23] and node2vec [8] augment networks by generating random paths according to different distributions. The edge samples derived by path are correlated in those methods. LINE [32] directly adds edges to networks and generates independent edge samples. GraRep [4] and NetMF [24] take different powers of the adjacency matrix as augmentation. Our parallel online augmentation generates decorrelated edge samples using pseudo shuffle, and thus is close to the augmentation in LINE. However, our augmentation does not need to store the whole augmented network, which saves a lot of disk and memory usage compared to existing methods.

In the embedding training stage, most existing node embedding algorithms [8, 23, 32] train node embedding with standard negative sampling [18] in a shared memory space. While there is a parallel word embedding algorithm [29] that restricts negative sampling within the context partition of each worker, it still needs to transfer rows of embedding matrices between each worker for positive samples. By contrast, our parallel negative sampling trains on orthogonal sample blocks and does not need any transfer between worker during an episode. The most related method is the distributed SGD used in large-scale matrix factorization algorithms [3, 6, 38, 40]. These methods divide the input matrix into $n \times n$ blocks and factorize orthogonal blocks simultaneously. Different from these methods, our system mainly focuses on the negative sampling technique and is designed for the node embedding task.

Node Embedding System From the perspective of system, our work belongs to the parallel implementation of node embedding. There are many CPU parallel systems, including DeepWalk [3], LINE [2], node2vec [4] and VERSE [6]. These systems use asynchronous SGD [25] in embedding training and exploit multiple CPU threads for acceleration. Due to the limited computation speed of CPUs, such systems cannot scale to ten-million-scale networks without a large CPU cluster. Recently, there are some GPU parallel systems [2] built on deep learning frameworks like TensorFlow [1] or PyTorch [22]. Since existing frameworks are based on mini-batch SGD paradigm, these systems severely suffer the problem of limited bus bandwidth, and are even worse than their CPU counterparts. Compared to them, *GraphVite* is a hybrid CPU-GPU system that leverages distinct advantages of CPUs and GPUs, and uses them collaboratively to train node embedding, which makes it much faster than either pure CPU or mini-batch-SGD based systems.

In addition, parallel word embedding systems [9, 11, 18, 28] are also very related to our work, since they share similar embedding training and negative sampling steps with node embedding. Among these methods, Wombat [28] and BlazingText [9] accelerate training with GPUs. Wombat only supports single GPU. BlazingText can scale to multiple GPUs, but it simply makes a copy of the parameter matrices on each GPU. Our system is more efficient than BlazingText in two aspects. First, our system partition the parameter matrices and consumes less memory on each GPU. Second, our system requires less synchronization cost, as GPUs do not share any rows in the parameter matrices.

7 CONCLUSION

In this paper, we present a high-performance CPU-GPU hybrid system for node embedding. Our system extends existing node embedding methods to GPUs and significantly accelerates training node embeddings on a single machine. With parallel online augmentation, *GraphVite* efficiently utilizes CPU threads to generate augmented edge samples for node embedding training. With parallel negative sampling, *GraphVite* enables training node embeddings on multiple GPUs without much synchronization. A collaboration strategy is also developed to reduce the synchronization cost between CPUs and GPUs. Experiments on 4 large networks prove that *GraphVite* significantly outperforms existing systems in speed without sacrifice on performance. In the future, we plan to generalize our system to semi-supervised settings and graph neural networks, such as graph convolutional networks [12], graph attention networks [34], and neural message passing networks [7].

ACKNOWLEDGEMENTS

We would like to thank Compute Canada [7] for supporting GPU servers. Jian Tang is supported by the Natural Sciences and Engineering Research Council of Canada and the Canada CIFAR AI Chair Program. We specially thank Wenbin Hou for useful discussions on C++ and GPU programming techniques, and Sahith Dambekodi for proofreading this paper.

[6] https://github.com/xgfs/verse
[7] https://www.computecanada.ca

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

[2] Natural Language Processing Lab at Tsinghua University. 2017. OpenNE: An open source toolkit for Network Embedding. https://github.com/thunlp/OpenNE.

[3] Prasad G Bhavana and Vineet C Nair. 2019. BMF: Block matrix approach to factorization of large scale data. *arXiv preprint arXiv:1901.00444* (2019).

[4] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 891–900.

[5] Dan C Ciresan, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and Jürgen Schmidhuber. 2011. Flexible, high performance convolutional neural networks for image classification. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, Vol. 22. Barcelona, Spain, 1237.

[6] Rainer Gemulla, Erik Nijkamp, Peter J Haas, and Yannis Sismanis. 2011. Large-scale matrix factorization with distributed stochastic gradient descent. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 69–77.

[7] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212* (2017).

[8] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[9] Saurabh Gupta and Vineet Khare. 2017. BlazingText: Scaling and Accelerating Word2Vec using Multiple GPUs. In *Proceedings of the Machine Learning on HPC Environments*. ACM, 6.

[10] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584* (2017).

[11] Shihao Ji, Nadathur Satish, Sheng Li, and Pradeep Dubey. 2016. Parallelizing word2vec in multi-core and many-core architectures. *arXiv preprint arXiv:1611.06172* (2016).

[12] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[14] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58, 7 (2007), 1019–1031.

[15] Long-Ji Lin. 1993. *Reinforcement learning for robots using neural networks*. Technical Report. Carnegie-Mellon Univ Pittsburgh PA School of Computer Science.

[16] Robert Meusel, Sebastiano Vigna, Oliver Lehmberg, and Christian Bizer. 2015. The graph structure in the web: Analyzed on different aggregation levels. *The Journal of Web Science* 1, 1 (2015), 33–47.

[17] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaev, Ganesh Venkatesh, et al. 2017. Mixed precision training. *arXiv preprint arXiv:1710.03740* (2017).

[18] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[19] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. 2007. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*. ACM, 29–42.

[20] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[21] David Newman, Padhraic Smyth, Max Welling, and Arthur U Asuncion. 2008. Distributed inference for latent dirichlet allocation. In *Advances in neural information processing systems*. 1081–1088.

[22] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. (2017).

[23] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[24] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 459–467.

[25] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. 2011. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in neural information processing systems*. 693–701.

[26] Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer.

[27] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–93.

[28] Trevor M Simonton and Gita Alaghband. 2017. Efficient and accurate Word2Vec implementations in GPU and shared-memory multicore architectures. In *High Performance Extreme Computing Conference (HPEC), 2017 IEEE*. IEEE, 1–7.

[29] Stergios Stergiou, Zygimantas Straznickas, Rolina Wu, and Kostas Tsioutsiouliklis. 2017. Distributed Negative Sampling for Word Embeddings.. In *AAAI*. 2569–2575.

[30] Damian Szklarczyk, John H Morris, Helen Cook, Michael Kuhn, Stefan Wyder, Milan Simonovic, Alberto Santos, Nadezhda T Doncheva, Alexander Roth, Peer Bork, et al. 2016. The STRING database in 2017: quality-controlled protein–protein association networks, made broadly accessible. *Nucleic acids research* (2016), gkw937.

[31] Jian Tang, Jingzhou Liu, Ming Zhang, and Qiaozhu Mei. 2016. Visualizing large-scale and high-dimensional data. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 287–297.

[32] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1067–1077.

[33] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2018. VERSE: Versatile Graph Embeddings from Similarity Measures. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 539–548.

[34] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* 1, 2 (2017).

[35] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2018. Deep graph infomax. *arXiv preprint arXiv:1809.10341* (2018).

[36] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1225–1234.

[37] Jaewon Yang and Jure Leskovec. 2015. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* 42, 1 (2015), 181–213.

[38] Hyokun Yun, Hsiang-Fu Yu, Cho-Jui Hsieh, SVN Vishwanathan, and Inderjit Dhillon. 2014. NOMAD: Non-locking, stOchastic Multi-machine algorithm for Asynchronous and Decentralized matrix completion. *Proceedings of the VLDB Endowment* 7, 11 (2014), 975–986.

[39] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. 2016. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160* (2016).

[40] Yong Zhuang, Wei-Sheng Chin, Yu-Chin Juan, and Chih-Jen Lin. 2013. A fast parallel SGD for matrix factorization in shared memory systems. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 249–256.

Transfer Learning for Unsupervised Influenza-like Illness Models from Online Search Data

Bin Zou
Department of Computer Science
University College London
United Kingdom
bin.zou.14@ucl.ac.uk

Vasileios Lampos
Department of Computer Science
University College London
United Kingdom
v.lampos@ucl.ac.uk

Ingemar J. Cox*
Department of Computer Science
University College London
United Kingdom
i.cox@ucl.ac.uk

ABSTRACT

A considerable body of research has demonstrated that online search data can be used to complement current syndromic surveillance systems. The vast majority of previous work proposes solutions that are based on supervised learning paradigms, in which historical disease rates are required for training a model. However, for many geographical regions this information is either sparse or not available due to a poor health infrastructure. It is these regions that have the most to benefit from inferring population health statistics from online user search activity. To address this issue, we propose a statistical framework in which we first learn a supervised model for a region with adequate historical disease rates, and then transfer it to a target region, where no syndromic surveillance data exists. This transfer learning solution consists of three steps: (i) learn a regularized regression model for a source country, (ii) map the source queries to target ones using semantic and temporal similarity metrics, and (iii) re-adjust the weights of the target queries. It is evaluated on the task of estimating influenza-like illness (ILI) rates. We learn a source model for the United States, and subsequently transfer it to three other countries, namely France, Spain and Australia. Overall, the transferred (unsupervised) models achieve strong performance in terms of Pearson correlation with the ground truth (> .92 on average), and their mean absolute error does not deviate greatly from a fully supervised baseline.

ACM Reference Format:
Bin Zou, Vasileios Lampos, and Ingemar J. Cox. 2019. Transfer Learning for Unsupervised Influenza-like Illness Models from Online Search Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313477

1 INTRODUCTION

Syndromic surveillance systems aim to provide timely estimates about the prevalence of a disease in a population. Their main source of information is based on doctor assessments about the probable health status of patients given a set of symptoms. For example, to monitor the rate of influenza, syndromic surveillance relies on a network of doctors who report on a daily or weekly basis the number of patients exhibiting related symptoms, such as fever, cough or a sore throat. Recent research efforts have shown that this traditional approach can be complemented by alternative methods trained on data from online user activity, e.g. social media or online search behavior [43]. Applications vary from modelling dengue fever [24] to depression [17], but particular research focus has been drawn to influenza, an infectious disease that is responsible for 290-650,000 deaths worldwide on an annual basis.[1] Data from the microblogging platform of Twitter [15, 32, 50] as well as from search engines [21, 35, 53, 67] combined with statistical natural language processing methods have produced promising outcomes, which in some occasions have been incorporated into national influenza surveillance schemes [9, 63]. The main advantages of these complementary methods are timeliness, and sampling from a larger segment of the population, including people who may not visit a doctor while being ill. It is also commonly cited that such approaches may be very useful in regions where health infrastructure is poor or absent. However, this is often impractical as the proposed machine learning solutions rely on training data which apart from the user-generated inputs, need to contain confirmed disease rates at the target location, broadly referred to as "ground truth". This data is typically provided by existing syndromic surveillance systems. Hence, for locations where ground truth is not available, user-data driven approaches are not realistically applicable.

In this paper, we propose a statistical framework to circumvent problems associated with no training data in some geographic regions. Our approach is based on the broad notion of transfer learning, where we aim to transfer parts of the knowledge gained while solving a certain task to better solve a different, but related one [49]. In particular, our goal is to transfer a well-performing disease rate inference model from a source location, where supervised learning is possible, to a target location, where supervision is not possible, given the lack of ground truth. We focus our experiments on influenza (flu) and utilize Google search query statistics as our descriptive variable for aggregate, population-level, online user activity. For example, the US Centers for Disease Control and Prevention (CDC) monitor and report influenza-like illness (ILI) rates on a weekly basis, providing sufficient ground truth to learn a function that maps online search query frequencies to these rates. In our experiments we show that we can adapt this function to derive estimates of ILI rates at different locations (outside the US). Language may or may not differ between the source and target locations. Online search statistics can be obtained for these target locations, but we assume that there is no ground truth data.

*Also affiliated with the University of Copenhagen, Denmark.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313477

[1] World Health Organization, who.int/mediacentre/news/statements/2017/flu/

The proposed approach comprises 3 steps. After learning a source regression model (step 1), we seek ways to map the selected source search queries to sets of queries in the target location. To derive this mapping we deploy a hybrid metric, which combines a semantic similarity with a time series correlation component (step 2). Semantic similarities are estimated using cross-lingual or monolingual word embeddings and correlations are computed using query frequencies. Finally, query weights from the source model are transferred to the identified target queries (step 3). This framework is evaluated on three transfer learning tasks, where the source model is always based in the US, and the target countries are France, Spain and Australia. While ground truth is available for all the target countries, we only use it to evaluate the performance of the transferred models. Transferred models, assessed on four flu seasons (2012 to 2016), can accurately estimate the peak of each flu season, achieving on average Pearson correlations greater than .92 and root mean squared errors comparable to the ones obtained by the corresponding fully supervised models ($\leq$ 21.6% increase in errors). Therefore, they can be considered as practical solutions for locations that lack historical ground truth data.

Main contributions. A novel, end-to-end transfer learning framework is proposed for mapping a disease model trained on online search data from a location, where ground truth is available, to a location, where ground truth is not available. Variations of this model are investigated, exploring different query mapping functions using semantic or temporal similarities or combinations of the two. In addition, we empirically show that our approach works in three case studies, two of which require a transfer to a different language (English to French or Spanish), and one that maintains the same language (English), but demands a model transfer to a different hemisphere (US to Australia).

2 DATA SETS

We use two sources of data, namely Google search query frequency statistics and ILI rates from established health organizations.

Google search query frequency statistics. Time series of weekly search query frequencies were retrieved through Google Correlate. A frequency represents the weekly search activity of a query (number of times issued) within a geographical region. It is normalized by dividing by the total number of search queries issued during that week. This normalization controls for variations in the number of searches issued each week which can be due to a variety of causes, including summer vacations, responses to news events, and a longer-term trend of increased web usage [45]. Normalized query frequencies are subsequently standardized, such that their time series have a zero mean and a standard deviation of one. This results in expressing query frequencies under the same units for different geographical regions with potentially varying population sizes and search usage patterns. We obtained weekly frequencies of search queries from September 1, 2007 to August 31, 2016 inclusive (470 weeks) for US, France, Spain, and Australia. Given that an exhaustive list of user search queries was not available to us, we extracted them by first using a set of 12 flu-related queries per country as a seed to Google Correlate and then iterating through this process (using correlated queries as new seeds). This process extracted 34,121, 29,996, 15,673 and 8,764 queries for US, France, Spain and Australia, respectively. Queries were not limited to the topic of flu, given that various other spurious queries may also correlate with the seeds.

ILI rates. We obtained weekly ILI rates for the US, France, Spain and Australia from their established syndromic surveillance systems, namely the Centers for Disease Control and Prevention (CDC), GPs Sentinelles Network (SN), Spanish Influenza Sentinel Surveillance System (SISSS), and Australian Sentinel Practices Research Network (ASPREN), respectively.[2] ILI rates represent fraction of the population that has been diagnosed with influenza-like symptoms.[3] The data spans from September 1, 2007 to August 31, 2016 inclusive, which covers approximately 9 consecutive influenza seasons. Note that for Spain, we only have ILI rates from week 40 in a year to week 20 in the following year. The prevalence of influenza outside this period is typically very low.[4] We denote the ILI rates from each syndromic surveillance system using the corresponding country code (US, FR, ES, and AU).

In our experiments we are transferring a flu model trained on US data to one of the other three countries. To provide some insight about the difficulty of the task, we have plotted the historical ILI rates for all countries in Fig. 1. ILI rates may correlate between countries, e.g. the Pearson correlation between the US and FR rates is equal to .6 ($p \approx 3 \cdot 10^{-54}$), but peaks and troughs are occurring at different times and with very different intensity. The US and AU ILI rates are negatively correlated ($-.4$, $p \approx 8 \cdot 10^{-17}$), as expected, since these countries are situated in different hemispheres and influenza is strongly seasonal. The optimal correlation we can obtain by shifting the ILI rate time series is equal to .68 (US-ES). Notably, the metric for ILI may differ in the countries we considered in this paper. Therefore, in our experiments we are working with a standardized representation of ILI (z-score).

3 METHODS

Disease rate estimation from online search data is commonly formulated as a regression task [21, 35]. The aim is to learn a function $f\colon \mathbf{X} \rightarrow \mathbf{y}$ that maps the input space of search query frequencies, $\mathbf{X} \in \mathbb{R}^{n \times s}$, to the target variable, $\mathbf{y} \in \mathbb{R}^{n}$, representing disease rates; n denotes the number of samples and s is the size of the feature space, i.e. the number of unique search queries we are considering. More specifically, $\mathbf{X}$ contains the time series of search query frequencies, and $\mathbf{y}$ represents a rate of disease diagnoses in a population (as reported by a health agency) at corresponding times. The time interval for computing the frequency of queries is often set to one week to match the frequency of syndromic surveillance reports.

Regression approaches require observations of the target variable $\mathbf{y}$ (ground truth) for training a machine learning model. This restricts the application of such techniques to areas where historical disease rates are available. We attempt to address this limitation by proposing a transfer learning methodology, that maps an existing disease model, $f\colon \mathbf{X} \rightarrow \mathbf{y}$, from a source location, where disease rates are available, to another location, where disease rates are not possible to obtain. We define the source domain as $\mathcal{D}_S = \{(\mathbf{x}_i, y_i)\}$,

[2] Links: **CDC** (US), `cdc.gov`; **SN** (FR), `websenti.u707.jussieu.fr/sentiweb`; **SISSS** (ES), `eng.isciii.es/ISCIII`; **ASPREN** (AU), `aspren.dmac.adelaide.edu.au`

[3] ILI is defined as the presence of high fever plus cough or sore throat [11, 46].

[4] In Figs. 1 and 2, we have set the missing ILI rates for ES to zero for visualization purposes. However, we are not using these rates to train or evaluate models for ES.

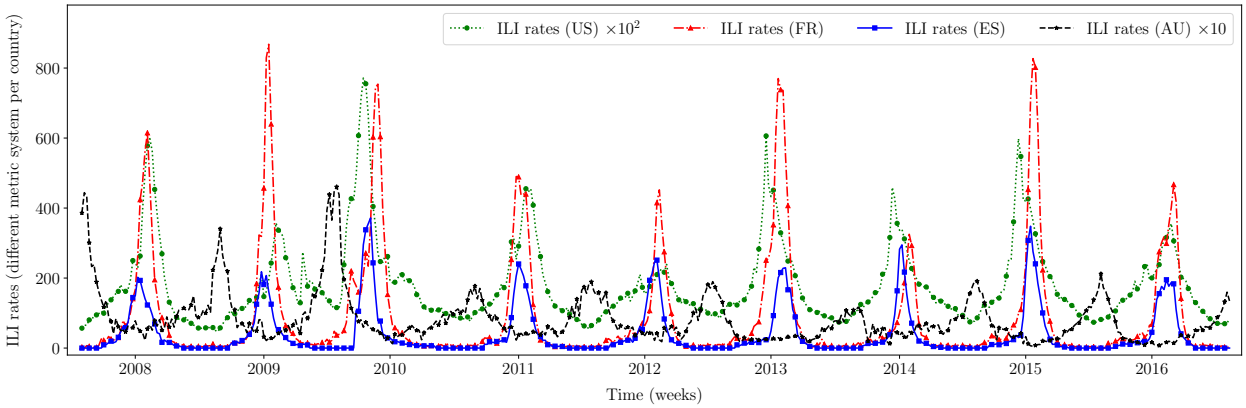


Figure 1: ILI rates for the United States (US), France (FR), Spain (ES) and Australia (AU).

$i \in \{1, \ldots, n\}$, where $\mathbf{x}_i$ is an s-dimensional vector holding the frequencies of the s queries for the time interval i, y_i is the corresponding disease rate, and n is the number of observations. The target domain is denoted by $\mathcal{D}_\mathrm{T} = \left\{\mathbf{x}'_i\right\}$, $i \in \{1, \ldots, m\}$, where $\mathbf{x}'_i$ is a t-dimensional vector of the frequencies of the t queries in the target domain that are going to be associated with the s queries in the source domain. No ground truth is available for the target domain. Note that t need not equal s, thus allowing one-to-many query mappings. In theory, the m time intervals may precede or overlap the n time intervals in the source region. In our experiments, we the m target intervals are always after the n source intervals.

3.1 User search behavior in different countries

As the transfer learning framework is detailed in the next paragraphs, it will become apparent that it is grounded on a fundamental assumption, which is that online user search behavior will be similar in the source and the target countries. Narrowing this assumption down to our specific task, this implies that the conditional probability of issuing a query q under a certain health status h (with or without experiencing disease symptoms), $P(q|h)$, will be similar for the populations of the source and the target countries. Relevant literature offers some evidence about this with regards to user search behavior for various health-related themes [1, 3, 25, 68]. In addition, we also provide some empirical evidence using our data. Table 1 shows the average query frequency over the corresponding ILI rate ratio for three basic queries in the US and AU. It also shows these ratios for translations of these queries in FR and ES (e.g. flu → grippe (FR) → gripe (ES)). The main observation is that these ratios do not vary much over the time span of our data, which is almost a decade. Although this is a limited observation, in that it does not involve many different search queries, it serves as a strong indication that user search behavior, at least for this specific area of interest, has similarities among different countries. The transfer learning framework, described in the following paragraphs, tries to exploit these similarities.

3.2 Transfer learning framework

The proposed transfer learning framework consists of three steps which are described in detail in the following sections.

3.2.1 *Step 1 — Learning a regression function in the source domain.*

Regularized regression has been successfully applied to various text regression tasks, including the estimation of disease rates from social media or online search data [32, 35]. In this paper, we use elastic net [74] as our regression function, similarly to previous work on the topic [35, 37]. Elastic net combines ℓ_1-norm regularization, commonly known as the *lasso* [58], with ℓ_2-norm, or *ridge* [26], regularization. In addition to the sparsity encouraged by the ℓ_1-norm regularization, the ℓ_2-norm regularizer attempts to address model consistency problems that arise when collinear predictors exist in the input space [69], which is common in text regression tasks [34, 36, 54]. Given $\mathbf{X} \in \mathbb{R}^{n \times s}$ and $\mathbf{y} \in \mathbb{R}^n$ from the source domain $\mathcal{D}_\mathrm{S}$, we apply a constrained version of elastic net which solves the following optimization problem:

$$\underset{\mathbf{w}, \beta}{\operatorname{argmin}} \left(\|\mathbf{y} - \mathbf{X}\mathbf{w} - \beta\|_2^2 + \lambda_1 \|\mathbf{w}\|_2^2 + \lambda_2 \|\mathbf{w}\|_1 \right) \text{ subject to } \mathbf{w} \geq 0, \quad (1)$$

where $\lambda_1 > 0$, $\lambda_2 > 0$ are respectively the ℓ_1-norm and ℓ_2-norm regularization parameters, and β denotes the intercept term. The non-negativity constraint for $\mathbf{w}$ may result in a worse performing model for the source country, but, at the same time, makes the weight transfer from a source to a target country more comprehensible (positive weights are easier to interpret) and eventually more accurate in terms of performance (see Section 4.2).

Due to the seasonal nature of influenza, our dataset of candidate queries contains a significant number of confounders, i.e. queries with frequencies that are correlated to ILI rates, but have no link to flu, such as '*college basketball*' or '*spring break*'. To remove these unrelated queries we applied a semantic filter based on word embedding representations, similar to the one proposed in [38, 72, 73]. Word embeddings were trained on the English Wikipedia corpus using the `fastText` method [12]. A topic about flu, $\mathcal{T}$, was defined as a simple set of two flu-related terms, $\mathcal{T} = \{$'flu', 'fever'$\}$. For each of the source queries, we calculate a similarity score defined as the product of the cosine similarities between the embeddings of the terms in $\mathcal{T}$ and $\mathbf{e}_\mathrm{q}$, i.e.

$$g(\mathbf{q}, \mathcal{T}) = \cos\left(\mathbf{e}_\mathrm{q}, \mathbf{e}_{\mathcal{T}_1}\right) \times \cos\left(\mathbf{e}_\mathrm{q}, \mathbf{e}_{\mathcal{T}_2}\right), \quad (2)$$

where each cosine similarity component is mapped to $[0, 1]$ via $(\cos(\cdot, \cdot) + 1)/2$.[5] Queries from the source domain with $g \leq .5$ are

[5] This resolves misleading similarity scores based on different sign combinations.

Table 1: Mean ratio of query frequency over ILI rate (and standard deviation of the mean) in four countries.

Search queries	US	FR	ES	AU
flu (US/AU), grippe (FR), gripe (ES)	.036 (.010)	.033 (.012)	.032 (.011)	.031 (.016)
symptoms of flu (US/AU), symptômes de la grippe (FR), síntomas de gripe (ES)	.030 (.009)	.031 (.012)	.029 (.009)	.027 (.014)
flu in children (US/AU), grippe chez le bébé (FR), gripe en el bebé (ES)	.017 (.007)	.020 (.008)	.019 (.009)	.022 (.010)

filtered out and are not considered in our experiments. The remaining queries are used to train an elastic net. This operation further reduces the selected queries to a subset Q_S, i.e. the ones that have been allocated a nonzero weight.

3.2.2 Step 2 — Mapping source to target queries. The identified and weighted set of search queries in the source domain (Q_S) should be mapped to a set of queries in the target domain from a potential pool of target query candidates ($\mathcal{P}_T$). Queries about the same topic may vary in their textual formulation, especially when they are issued by users located in different countries. Even in cases, where countries share the same language, cultural and socioeconomic differences may result into different querying preferences. Thus, simple approaches, where search queries from the source country are translated or directly mapped to queries in the target country, are not effective.[6] In our approach, we utilize word embeddings (mono- or cross-lingual) to map source to target queries based on their broad semantic relationship. We consider both one-to-one and one-to-many query mappings from the source to the target domain. In addition, the weight associated with each source query reflects on how correlated the query is with the modeled disease rate. Therefore, another desired property is to map source queries to target ones based on their pairwise temporal correlation as this may enhance the statistical relevance of the mapping. Consequently, there is a trade-off between mapping based on semantic similarity and based on the similarity in temporal correlation. To capture both, we define a combined similarity metric, Θ, that is the weighted sum of a semantic similarity Θ_s and a correlation similarity, Θ_c, i.e.

$$\Theta = \gamma\Theta_s + (1-\gamma)\Theta_c\,, \tag{3}$$

where $\gamma \in [0,1]$ controls the relative weighting of each. When $\gamma = 1$ the mapping is based only on semantic similarity. Conversely, when $\gamma = 0$ the mapping is based only on the correlation similarity.

Semantic similarity (Θ_s)**.** If the source and target domains have different languages, a translation module is required. For this purpose, we deploy cross-lingual word embeddings. Cross-lingual embeddings are trained using corpora from multiple languages, and can be used to compute word similarities in different languages [57, 60, 61]. Empirical evidence indicates that they can also facilitate better knowledge transfer between languages [2, 44, 47]. The majority of cross-lingual word embedding models are trained by exploiting sources of monolingual text alongside a smaller cross-lingual corpus of aligned text [56]. The alignment can be made at word [2, 5, 18, 41, 57, 60], sentence [39, 75], and document level [44, 62]. In this paper, we utilize a method for learning bilingual word embeddings proposed by Smith *et al.* [57].

First, for each of the source and target languages, we respectively learn a word embedding space based on monolingual text. For all languages considered in our experiments (English, French and Spanish) we obtained word embeddings by applying `fastText` on corresponding Wikipedia corpora [12].[7] The dimensionality of the word embeddings was set to $d = 300$. Then, we used a core selection of exact translation pairs ($\sigma \rightarrow \tau$) from the source to the target domain language to generate bilingual embeddings. Given the embedding matrices of this alignment dictionary, $\mathbf{E}_\sigma$ and $\mathbf{E}_\tau$ both $\in \mathbb{R}^{m\times d}$, where m, d denote the number of translation pairs and the dimensionality of the word embedding respectively, we learn a transformation matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$ such that $\mathbf{E}_\tau \approx \mathbf{E}_\sigma\mathbf{W}$. $\mathbf{W}$ is an orthogonal matrix learned by minimizing the squared Euclidean distance between $\mathbf{E}_\sigma$ and $\mathbf{E}_\tau$, i.e.

$$\underset{\mathbf{W}}{\operatorname{argmin}} \|\mathbf{E}_\sigma\mathbf{W} - \mathbf{E}_\tau\|_2^2\,, \text{ subject to } \mathbf{W}^\top\mathbf{W} = \mathbf{I}\,. \tag{4}$$

The orthogonality constraint ensures that the transformation works both ways, that is $\mathbf{E}_\tau \approx \mathbf{E}_\sigma\mathbf{W}$, $\mathbf{E}_\sigma \approx \mathbf{E}_\tau\mathbf{W}^\top$, and $\mathbf{E}_\tau \approx \mathbf{E}_\tau\mathbf{W}^\top\mathbf{W}$ [57]. In addition, Artexte *et al.* have empirically shown that it also improves the performance of machine translation [4]. The exact solution of Eq. 4 is given by $\mathbf{W} = \mathbf{V}\mathbf{U}^\top$, where $\mathbf{E}_\tau^\top\mathbf{E}_\sigma = \mathbf{U}\Sigma\mathbf{V}^\top$ is the singular value decomposition of $\mathbf{E}_\tau^\top\mathbf{E}_\sigma$ [4, 23].

A query's embedding is defined as the average of the embeddings of its tokens, an effective practice for short texts [8, 42, 66, 72]. We denote with $\mathbf{v}_{S_i}, \mathbf{v}_{T_j}$ both $\in \mathbb{R}^{1\times d}$, the embeddings of a source query (from Q_S) and of a target query from $\mathcal{P}_T$, respectively. Then, an element ω_{ij} from the cosine similarity matrix $\Omega \in \mathbb{R}^{s\times|\mathcal{P}_T|}$ between the embeddings of source and valid target queries is given by $\omega_{ij} = \left(\mathbf{v}_{S_i}\mathbf{W}\,\mathbf{v}_{T_j}^\top\right) / \left(\|\mathbf{v}_{S_i}\mathbf{W}\|_2\|\mathbf{v}_{T_j}\|_2\right)$. Note that the cosine similarities are computed after projecting the embeddings of the source domain to the target domain using the transformation matrix $\mathbf{W}$.

In theory, we can directly use ω_{ij} to determine the k most similar target queries to the source query, thus providing a one-to-many mapping. However, in practice when conducting translations based on cross-lingual word embeddings, this may result in the presence of "hubs", i.e. target words or queries that are similar to unrealistically many different source words, a development that reduces the performance of translation [18, 57]. Smith *et al.* mitigate this effect by using an inverted softmax ranking, described next [57].

Given q_i in the source language, its translation is determined by finding candidate target queries q'_j that maximize the probability defined by

$$P_{j\rightarrow i} = \frac{\exp\left(\eta\,\omega_{ij}\right)}{\alpha_j \sum_{z=1}^{s}\exp\left(\eta\,\omega_{iz}\right)}\,, \tag{5}$$

where α_j is a normalization factor that ensures $P_{j\rightarrow i}$ is a probability, and s is the number of source queries in the vocabulary. The inverted softmax estimates the probability $P_{j\rightarrow i}$ that a candidate target query translates back to the source query, rather than

[6] We have empirical evidence about this, obtained during the first stages of this work.

[7] The embeddings were obtained from `github.com/facebookresearch/fastText`

the other way around, $P_{i\to j}$ [18, 57]. If a target query is a hub, then the denominator in Eq. 5 will be large, preventing this target query from being selected. The parameter η is learned by maximizing the log probability over the alignment dictionary ($\sigma \to \tau$), i.e., $\text{argmax}_\eta \sum_{\text{pairs } ij} \ln\left(P_{j\to i}\right)$. The top-$k$ queries from $\mathcal{P}_\text{T}$ with the highest pairing probability ($P_{j\to i}$) are then selected as possible translations of the source query q_i. Finally, we compute the semantic (cosine) similarity score Θ_s between the source query q_i and the target query q_j using $\Theta_\text{s}(q_i, q_j) = \left(\mathbf{e}_{q_i}\mathbf{W}\,\mathbf{e}_{q_j}^\top\right) / \left(\|\mathbf{e}_{q_i}\mathbf{W}\|_2 \|\mathbf{e}_{q_j}\|_2\right)$, where $\mathbf{e}_{q_i}$, $\mathbf{e}_{q_j}$ are the embeddings of q_i, q_j, respectively. Our experiments report results for a variety of values of k.

If the language in the source and the target domain is the same, the previously described approach is not applicable. Given potential differences in querying preferences across different countries, some of the source queries, Q_S, may not be present in the pool of candidate target queries, $\mathcal{P}_\text{T}$. Therefore, we use cosine similarity to map each source query to the k most similar target ones using the common word embedding space for the shared language.

Temporal correlation similarity (Θ_c)**.** We compute the Pearson correlation between the frequency time series of the source and target queries over a fixed period (set to 5 years in our experiments). Since the flu season may be offset in the target domain with respect to the source domain, we computed the maximum correlation between these two frequency time series using a shifting window of $\pm\xi$ weeks. The range of possible values for ξ is determined based on the seasonal offset between the source and target countries (see Section 4). Given a source query, q_i, and a target query, q_j which is a member of a mapping set $\mathcal{T}_i$ (consisting of $k \geq 1$ queries from $\mathcal{P}_\text{T}$), and their associated daily search frequencies, $\mathbf{x}_i(t)$ and $\mathbf{x}_j(t)$, respectively, the temporal correlation similarity, Θ_c, is given by

$$\Theta_\text{c}(q_i, q_j) = \rho\left(\mathbf{x}_i(t), \mathbf{x}_j(t + l_{ij})\right), \tag{6}$$

where $\rho(x_i(t), x_j(t + l_{ij}))$ denotes the optimal Pearson correlation coefficient between $\mathbf{x}_i, \mathbf{x}_j$ within the shifting window. Note that the optimal window is independently computed for each target query in $\mathcal{T}_i$, and thus optimal shifts may vary.

3.2.3 Step 3 — Weighting target queries. In the previous steps, we have established that a source query q_i, which has received a regression weight w_i, is mapped to a set, $\mathcal{T}_i$, of $k \geq 1$ queries in the target domain. If $k = 1$, then we can directly assign w_i to the single target query. If $k > 1$, then the source query's weight, w_i, should be distributed across these k mapping target queries. To perform this, we have considered two alternatives:

- **Uniform.** We divide the source query weight, w_i, by the number of queries q'_j in $\mathcal{T}_i$, and assign each query in $\mathcal{T}_i$ a weight equal to $w'_j = w_i/k$.
- **Non-uniform.** The k target query weights are determined based on each target query's similarity score $\Theta_{ij}, j \in \{2, \ldots, k\}$, with the source query (see Eq. 3). More specifically, a target weight w'_j is defined as $w'_j = w_i\Theta_{ij'} / \sum_{q'_j \in \mathcal{T}_i} \Theta_{ij'}$.

To obtain a baseline performance estimate, we randomly shuffle the established query mappings in Step 2, and then transfer the source weights to k target queries using the uniform approach. We repeat this process multiple times and report the mean performance of these randomized transfer learning models.

4 EXPERIMENTS

We deploy the proposed transfer learning framework to estimate ILI rates in three target countries without using any ground truth from these countries to supervise modeling. US is always set as the source country, while the target countries are FR, ES and AU. We assess the performance of the proposed model, comparing it to various baselines, and also provide a qualitative analysis, aiming to interpret some of the intrinsic properties of our approach.

Settings. After applying the semantic filter (Eq. 2) to the pool of 34,121 US queries, 1,403 queries were retained. The applied evaluation protocol is as follows. We train a source model (US) using the first 5 flu seasons (2007-12). A flu season is conventionally defined as the 1-year long period from the first week in September to the last week of August in the next year.[8] Prior to applying elastic net, we maintain search queries that have a $\geq .3$ Pearson correlation with the US ILI rates (these queries may vary per training fold). We then transfer the model to FR, ES, and AU and test it in the following flu season (2012-13). Then, we move our training data window to include the 2012-13 flu season and remove the first flu season (2007-08), and test in the following season (2013-14), so that we still have 5 flu seasons to train. We repeat this process until we have tested on the last flu season in our data set (2015-16), evaluating performance 4 times in total. The window size (ξ) used for identifying optimal correlations between the frequency time series of the source and target queries (see Section 3) is set to ±6 weeks for FR and ES. The window is the same for AU, although prior to applying it, the query frequency time series are shifted by 6 months to account for the seasonal difference in the northern and southern hemispheres. For the one-to-k mapping from a source to a set of target queries, we explore sizes up to $k = 5$ (values > 5 did not yield any different insights). We measure the performance of transferred models by comparing our estimates with their national public health estimates, using Pearson correlation (r), mean absolute error (MAE), and root mean squared error (RMSE). Regression errors are computed after reverting inferences back to their corresponding non standardized values.

Baseline models. To demonstrate the effectiveness of our transfer learning framework, we compare it with four baseline models:

- **Random**. After determining the mapping between source and target queries, the pairs (one-to-k) are randomly permuted. The source query weight is uniformly distributed across the mapped k target queries. We repeat this process 2,000 times and report the average inference performance. This random assignment of query weights provides a possibly worst case baseline.
- **Transfer component analysis (TCA)**. TCA is a transfer learning approach that aims to learn transfer components across source and target domains in a reproducing kernel Hilbert space using maximum mean discrepancy [48]. After we map source to target queries, TCA is applied to source and target query frequencies.
- **Unsupervised query selection based on semantic similarity**. We apply a semantic filter (described in Eq. 2) to remove queries that are irrelevant to the flu topic. The term pairs {'grippe',

[8] Note that for AU this may result into including the end of a flu season and the beginning of the next in training and test folds.

Table 2: Performance estimates for the US→FR transfer learning task. Different values of γ determine how queries are mapped from the source to the target domain (γ=1: semantic similarity only, γ=0: temporal correlation only, $\gamma \in (0,1)$: joint similarity score). Numbers in parentheses represent the standard deviation of the error. The best performance among all transfer learning models is denoted in bold. The best performance among models under a common γ value is underlined. Only the best random mapping performance (R) is enumerated per choice of γ. The last two rows show the performance of the baseline models.

Mapping	k	w	09/2012 – 09/2013			09/2013 – 09/2014			09/2014 – 09/2015			09/2015 – 09/2016			Average		
			r	MAE	RMSE	r	MAE	RMSE	r	MAE	RMSE	r	MAE	RMSE	r	MAE	RMSE
$\gamma = 0$	1	–	.797	78.905	136.098	.789	59.584	93.752	.900	56.107	92.324	.855	51.533	78.073	.835 (.045)	61.532 (10.429)	100.062 (21.690)
	2	U	.803	80.044	137.247	.794	59.961	94.853	.890	58.372	96.282	.843	55.532	84.438	.833 (.038)	63.477 (9.696)	103.205 (20.179)
	3	U	.802	79.010	135.905	.796	59.750	95.350	.896	57.241	94.451	.844	57.306	86.588	.834 (.040)	63.327 (9.111)	103.073 (19.260)
	4	U	.798	79.077	135.892	.795	59.529	95.295	.895	58.380	95.852	.834	59.729	90.180	.830 (.040)	64.179 (8.617)	104.305 (18.370)
	5	U	.799	78.881	135.743	.794	58.508	95.036	.893	58.439	96.988	.829	60.075	91.182	.829 (.040)	63.976 (8.630)	104.737 (18.023)
	2	NU	.803	80.012	137.180	.794	59.971	94.869	.891	58.360	96.268	.843	55.502	84.399	.833 (.038)	63.461 (9.689)	103.179 (20.159)
	3	NU	.802	78.999	135.881	.796	59.763	95.360	.896	57.244	94.453	.844	57.271	86.538	.834 (.040)	63.319 (9.110)	103.058 (19.259)
	4	NU	.799	79.068	135.875	.795	59.519	95.278	.895	58.367	95.834	.834	59.676	90.106	.830 (.040)	64.157 (8.623)	104.273 (18.381)
	5	NU	.799	78.868	135.725	.794	58.499	95.015	.893	58.434	96.972	.829	60.029	91.110	.829 (.040)	63.957 (8.632)	104.706 (18.033)
	1	R	.771	125.422	152.275	.731	93.122	105.769	.807	138.579	158.000	.825	102.972	113.607	.783 (.036)	115.024 (17.943)	132.413 (22.982)
$\gamma = 1$	1	–	.964	51.885	77.728	.928	24.373	35.801	.974	51.623	69.254	.917	75.416	92.946	.946 (.024)	50.824 (18.071)	68.932 (20.927)
	2	U	.967	41.298	68.164	.939	22.993	33.287	.973	62.869	81.119	.924	84.469	102.422	.951 (.020)	52.907 (23.049)	71.248 (25.099)
	3	U	.967	39.789	67.336	.947	21.219	30.446	.972	58.654	79.471	.933	76.235	93.338	.955 (.016)	48.974 (20.564)	67.648 (23.366)
	4	U	.965	40.120	65.882	.947	24.037	33.095	.970	63.290	85.390	.939	77.601	93.301	.955 (.013)	51.262 (20.638)	69.417 (23.224)
	5	U	.965	37.632	61.217	.952	26.136	35.651	.972	66.825	90.248	.943	78.479	93.855	.958 (.011)	52.268 (21.190)	70.243 (23.642)
	2	NU	.968	41.272	68.016	.939	22.925	33.213	.973	61.971	80.280	.924	83.058	101.160	.951 (.020)	52.306 (22.495)	70.667 (24.658)
	3	NU	.967	39.665	66.933	.948	21.189	30.378	.973	58.568	79.476	.933	75.661	92.917	.955 (.016)	48.770 (20.388)	67.426 (23.280)
	4	NU	.966	39.754	65.480	.948	23.794	32.767	.971	62.957	85.275	.939	76.868	92.866	.956 (.013)	50.843 (20.486)	69.097 (23.236)
	5	NU	.966	37.295	60.749	.952	25.925	35.383	.972	66.890	90.583	.943	77.969	93.647	.958 (.012)	52.020 (21.167)	70.091 (23.805)
	3	R	.891	83.535	113.537	.890	79.396	86.904	.949	116.532	124.478	.922	109.746	119.219	.913 (.024)	97.302 (16.084)	111.034 (14.459)
	2	C	.968	39.972	65.695	.941	21.639	31.190	.974	59.103	77.964	.926	78.798	97.444	.952 (.019)	49.878 (21.313)	68.073 (24.117)
	3	C	.967	38.062	64.349	.949	20.408	29.002	.973	56.188	77.822	.933	72.492	90.289	.956 (.016)	46.788 (19.501)	65.365 (22.911)
	4	C	.965	38.225	63.063	.949	22.869	31.161	.971	60.623	83.764	.938	73.644	90.367	.956 (.013)	48.840 (19.629)	67.089 (23.059)
	5	C	.966	35.827	58.820	.953	24.940	33.619	.973	63.562	87.764	.942	74.547	90.793	.958 (.012)	49.719 (20.094)	67.749 (23.325)
$\gamma_{\text{opt}} = .5$	1	–	.968	33.475	53.775	.951	22.615	34.416	.973	34.793	58.007	.944	45.324	62.417	**.959 (.012)**	**34.052 (8.043)**	**52.153 (10.687)**
	2	U	.959	37.461	60.529	.939	24.885	38.056	.967	43.197	69.883	.930	54.504	74.766	.949 (.015)	40.012 (10.671)	60.809 (14.097)
	3	U	.954	38.786	63.909	.939	26.390	39.771	.968	44.241	71.312	.931	61.182	81.592	.948 (.014)	42.650 (12.503)	64.146 (15.410)
	4	U	.948	41.150	69.125	.934	29.553	43.996	.966	47.021	74.662	.932	62.330	82.811	.945 (.014)	45.014 (11.810)	67.649 (14.498)
	5	U	.945	41.936	71.322	.925	30.387	46.164	.963	46.108	75.703	.931	61.750	82.670	.941 (.015)	45.045 (11.233)	68.965 (13.772)
	2	NU	.959	37.414	60.456	.939	24.881	38.036	.967	43.118	69.763	.930	54.329	74.599	.949 (.015)	39.936 (10.610)	60.714 (14.045)
	3	NU	.954	38.675	63.792	.940	26.423	39.789	.968	44.452	71.495	.931	61.147	81.601	.948 (.014)	42.674 (12.495)	64.169 (15.428)
	4	NU	.948	40.867	68.727	.935	29.381	43.748	.966	47.093	74.691	.932	62.323	82.804	.945 (.014)	44.916 (11.890)	67.492 (14.591)
	5	NU	.946	41.610	70.892	.926	30.201	45.863	.963	46.192	75.685	.931	61.788	82.685	.942 (.015)	44.948 (11.333)	68.781 (13.881)
	1	R	.913	86.752	110.096	.846	72.130	83.158	.943	94.681	109.176	.942	97.352	104.952	.911 (.039)	87.729 (9.813)	101.845 (10.962)
Unsupervised	–	–	.936	–	–	.870	–	–	.947	–	–	.910	–	–	.916 (.030)	–	–
Supervised	–	–	.977	27.331	50.643	.979	23.665	33.994	.992	34.345	62.803	.987	15.011	21.956	.984 (.006)	25.088 (6.970)	42.349 (15.595)

k: number of target queries (1-to-k mapping), **w**: weighting approach, **U**: uniform, **NU**: non-uniform, **C**: correlation, **R**: random

'fièvre'}, {'gripe', 'fiebre'} and {'flu', 'fever'} are used to define this semantic filter in FR, ES and AU, respectively. Queries with $g \leq .5$ are filtered out and are not considered in our experiments. The mean weekly frequency of the retained queries is regarded as a proxy of the estimated ILI rates. These estimates are in different scale with the true ILI rates, thus we only report their Pearson correlation (r).

- **Supervised learning**. We first apply a semantic filter (see point above) to the queries of each target country. We then train an elastic net, after maintaining only queries that have a moderate correlation with the ground truth ($r \geq .3$ with the target values in the training data). This is inline with previously proposed, state-of-the-art supervised models for the task [38] and is considered as the top performance we could obtain, if we had access to ground truth in the target countries.

4.1 Quantitative analysis

Performance estimates are enumerated in Tables 2, 3, and 4 for each transfer learning task (US→FR, US→ES, US→AU). We first explored the extreme cases of $\gamma = 0$ and $\gamma = 1$ (Eq. 3) that result to using only temporal correlation or semantic similarity, respectively.

For $\gamma = 0$, spurious queries could be included in the target domain's mappings. This is a result of the way the pool of target queries, $\mathcal{P}_T$, was originally formed (see Section 2). Seasonal search queries, correlating with the occurrence of flu incidents in a population, are very likely to be selected as mappings, e.g. "symptoms flu" was mapped to "ski serre chevalier" in the US→FR task. Seasonal activities or expressions may change in time, and thus such queries are very unstable predictors. In fact, the best average performance we can obtain for $\gamma = 0$ is considerably worse (MAEs of 61.532, 25.977 and 42.348 for FR, ES, and AU) than for alternative values. Setting $k = 1$ provides the best results on average. In general, performance is not affected much by different choices of weighting (uniform, non-uniform) or the number of queries in a mapping (k).

For $\gamma = 1$, we obtain on average more accurate estimates than for $\gamma = 0$. As a precursor to the joint similarity, we also introduce a correlation-based weighting scheme (denoted by "C"), which uses the optimal correlation between source and target queries (after

Table 3: Performance estimates for US→ES transfer learning task. Please refer to Table 2's caption for further information.

Mapping	k	w	09/2012 – 09/2013			09/2013 – 09/2014			09/2014 – 09/2015			09/2015 – 09/2016			Average		
			r	MAE	RMSE	r	MAE	RMSE	r	MAE	RMSE	r	MAE	RMSE	r	MAE	RMSE
$\gamma = 0$	1	–	.808	25.068	41.104	.807	25.789	42.137	.843	29.221	47.360	.791	25.134	38.497	.812 (.019)	26.303 (1.708)	42.275 (3.222)
	2	U	.799	25.589	42.631	.843	23.850	39.092	.844	30.069	48.120	.821	24.470	36.902	.827 (.019)	25.994 (2.434)	41.686 (4.240)
	3	U	.795	25.756	42.883	.840	23.669	38.934	.843	29.509	48.189	.813	24.989	37.713	.823 (.020)	25.981 (2.169)	41.930 (4.088)
	4	U	.783	26.504	43.662	.835	23.671	39.207	.844	29.850	48.335	.809	25.715	38.745	.818 (.024)	26.435 (2.226)	42.487 (3.884)
	5	U	.783	26.579	43.391	.840	23.605	38.861	.842	30.336	48.800	.806	26.586	39.843	.818 (.025)	26.776 (2.388)	42.724 (3.892)
	2	NU	.799	25.579	42.610	.843	23.852	39.095	.844	30.060	48.111	.821	24.472	36.907	.827 (.019)	25.991 (2.430)	41.681 (4.233)
	3	NU	.795	25.748	42.867	.840	23.670	38.936	.843	29.503	48.176	.813	24.989	37.712	.823 (.020)	25.977 (2.167)	41.922 (4.082)
	4	NU	.784	26.491	43.643	.835	23.671	39.209	.844	29.842	48.325	.809	25.708	38.734	.818 (.023)	26.428 (2.223)	42.478 (3.881)
	5	NU	.783	26.567	43.380	.840	23.605	38.866	.842	30.324	48.785	.806	26.575	39.826	.818 (.025)	26.768 (2.384)	42.714 (3.887)
	3	R	.830	40.548	46.584	.903	36.241	40.718	.846	53.929	61.098	.813	45.762	49.637	.848 (.034)	44.120 (6.591)	49.509 (7.419)
$\gamma = 1$	1	–	.954	28.614	34.944	.976	27.777	30.129	.919	44.638	50.082	.899	43.761	46.590	.937 (.030)	36.197 (8.013)	40.436 (8.175)
	2	U	.955	27.342	33.979	.976	27.118	29.294	.923	44.723	50.213	.925	44.518	49.547	**.945 (.022)**	35.926 (8.696)	40.758 (9.274)
	3	U	.958	25.523	31.885	.971	28.293	32.055	.916	47.603	53.909	.917	48.513	54.053	.941 (.024)	37.483 (10.625)	42.975 (11.006)
	4	U	.960	25.316	31.623	.973	27.998	31.797	.918	46.862	53.458	.918	47.443	52.823	.942 (.025)	36.905 (10.294)	42.425 (10.718)
	5	U	.957	24.445	30.821	.975	27.169	30.959	.917	45.775	52.620	.914	45.854	51.505	.941 (.026)	35.811 (10.050)	41.476 (10.594)
	2	NU	.955	26.336	32.978	.977	26.069	28.232	.923	43.543	49.056	.925	43.389	48.409	**.945 (.022)**	34.834 (8.632)	39.669 (9.221)
	3	NU	.958	25.532	31.879	.971	28.327	32.076	.917	47.471	53.737	.917	48.356	53.908	.941 (.024)	37.422 (10.543)	42.900 (10.923)
	4	NU	.960	25.324	31.587	.973	28.020	31.814	.919	46.770	53.334	.917	47.391	52.769	.942 (.025)	36.876 (10.251)	42.376 (10.678)
	5	NU	.958	24.432	30.759	.975	27.197	30.990	.917	45.778	52.576	.915	45.951	51.574	.941 (.026)	35.839 (10.073)	41.475 (10.607)
	2	R	.731	47.277	53.345	.804	44.924	52.394	.795	60.370	70.934	.719	48.986	56.506	.762 (.038)	50.389 (5.940)	58.295 (7.454)
	2	C	.954	25.520	33.516	.976	24.408	26.693	.923	41.827	47.782	.924	41.142	46.278	.944 (.022)	33.224 (8.273)	38.567 (8.816)
	3	C	.957	23.642	31.398	.970	25.353	29.090	.916	44.358	50.846	.916	45.405	51.174	.940 (.024)	34.690 (10.217)	40.627 (10.416)
	4	C	.960	23.339	30.912	.973	24.900	28.709	.919	43.431	50.236	.918	44.297	49.873	.942 (.025)	33.992 (9.892)	39.933 (10.153)
	5	C	.957	24.137	30.513	.974	26.555	30.466	.917	44.598	51.464	.915	45.662	51.359	.941 (.026)	35.238 (9.936)	40.950 (10.461)
$\gamma_{\text{opt}} = .2$	1	–	.931	21.419	30.004	.948	15.403	23.900	.907	27.050	39.864	.888	26.762	35.420	.918 (.023)	**22.658 (4.751)**	**32.297 (5.974)**
	2	U	.926	21.433	29.944	.941	17.334	25.525	.899	30.166	43.243	.877	30.662	40.272	.911 (.025)	24.899 (5.705)	34.746 (7.260)
	3	U	.936	21.249	28.841	.961	18.189	24.028	.908	31.608	42.568	.900	35.661	43.995	.926 (.024)	26.677 (7.186)	34.858 (8.608)
	4	U	.945	21.016	28.161	.965	18.720	23.647	.917	32.235	41.483	.910	37.141	44.448	.934 (.022)	27.278 (7.654)	34.435 (8.742)
	5	U	.946	20.977	28.041	.967	18.727	23.321	.910	33.330	43.018	.903	36.846	44.547	.932 (.026)	27.470 (7.760)	34.732 (9.219)
	2	NU	.926	21.427	29.932	.941	17.321	25.510	.899	30.135	43.214	.877	30.626	40.233	.911 (.025)	24.877 (5.694)	34.723 (7.251)
	3	NU	.936	21.254	28.845	.961	18.186	24.037	.908	31.583	42.554	.900	35.629	43.969	.926 (.024)	26.663 (7.171)	34.851 (8.595)
	4	NU	.945	21.023	28.158	.965	18.739	23.682	.917	32.241	41.509	.910	37.128	44.438	.934 (.022)	27.283 (7.643)	34.447 (8.734)
	5	NU	.946	20.983	28.033	.967	18.747	23.364	.910	33.337	43.028	.903	36.872	44.568	.932 (.026)	27.485 (7.762)	34.748 (9.215)
	1	R	.865	32.859	40.942	.931	33.323	38.687	.878	49.262	57.762	.814	45.799	51.424	.872 (.042)	40.311 (7.325)	47.204 (7.763)
$\gamma = .5$	1	–	.945	20.016	28.161	.965	17.720	23.647	.917	31.235	41.483	.910	36.141	44.448	.934 (.022)	26.278 (7.654)	34.435 (8.742)
Unsupervised	–	–	.936	–	–	.976	–	–	.910	–	–	.878	–	–	.925 (.036)	–	–
Supervised	–	–	.968	19.788	24.487	.993	30.642	41.059	.972	24.779	35.861	.954	13.271	20.992	.971 (.014)	22.120 (6.392)	30.600 (8.166)

k: number of target queries (1-to-k mapping), **w**: weighting approach, **U**: uniform, **NU**: non-uniform, **C**: correlation, **R**: random

deploying a shifting window) to determine the proportion of the source weight that will be allocated to the k mapped queries. In countries that deploy a translation module based on bilingual word embeddings, the "C" scheme ($k = 2$ or 3) outperforms the other two (uniform, non-uniform). For the US→AU task, where high semantic similarity often means that very similar queries are being mapped to each other (given the common language), the optimal model is obtained for $k = 1$, and thus, no further distribution of the weights is required. With or without the "C" weighting scheme, better performance is achieved compared to setting $\gamma = 0$ (MAEs of 46.788/48.77, 33.224/34.834 and 34.509/30.275 for FR, ES, and AU).

The joint similarity scheme attempts to combine the positive attributes of semantic and correlation based similarities. To assess its potential contribution, we performed a grid search using 9 values of γ (from .1 to .9), and presented the results for the best performing one (γ_{opt}). For completeness, we also show results for the default choices of $\gamma = .5$ and $k = 1$. Firstly, the application of the joint similarity leads to significant performance improvements in all tasks (MAEs of 34.052, 22.658 and 22.043 for FR, ES, and AU). Secondly, the best performing model consistently occurs for $k = 1$, i.e. for one-to-one query mappings, where no weight redistribution is required. Finally, although results do not deviate much from the default settings of $\gamma = .5$ and $k = 1$, there are discrepancies between the optimal γ value for each task ($\gamma_{\text{opt}} = .5$, .2 and .9 for FR, ES, and AU). One possible explanation may be that this is an artefact of the intrinsic characteristics (size, semantic/temporal similarities) of the pool of candidate target queries used for each task (see Section 4.2).

Better performance is always obtained (in terms of MAE and RMSE) compared to the random mapping allocation baseline ("R"), the best performance estimates of which per γ value are provided. The same holds for TCA, which performs even worse than random (results are omitted). One explanation for this is that TCA fails to capture the time series structure of this particular data set, an essential property for producing a meaningful solution. Furthermore, the optimal models (joint similarity) outperform the unsupervised baseline in terms of correlation, the only metric which is relevant in this case. Finally, compared to the fully supervised elastic net, the transfer learning unsupervised approach reaches to a comparable performance, which is worse by 23.15%, 5.55%, and 17.5% (in terms of RMSE), for FR, ES, and AU, respectively.

Fig. 2 plots the time series of a selection of these estimates, including the ones of the best performing models, in comparison to the ground truth, for each target country. We can see how estimates become significantly better when the joint similarity is

Table 4: Performance estimates for the US→AU transfer learning task. Please refer to Table 2's caption for further information.

Mapping	k	w	09/2012 – 09/2013			09/2013 – 09/2014			09/2014 – 09/2015			09/2015 – 09/2016			Average		
			r	MAE	RMSE	r	MAE	RMSE	r	MAE	RMSE	r	MAE	RMSE	r	MAE	RMSE
$\gamma = 0$	1	–	.704	38.804	50.140	.677	39.151	48.508	.630	51.412	65.215	.787	40.025	57.421	.700 (.056)	42.348 (5.359)	55.321 (6.830)
	2	U	.622	41.824	55.943	.663	41.708	50.752	.633	52.017	66.448	.763	40.557	59.312	.670 (.055)	44.027 (4.734)	58.114 (5.873)
	3	U	.621	42.263	56.819	.669	42.900	51.487	.631	53.041	67.754	.769	41.330	59.468	.672 (.058)	44.883 (4.840)	58.882 (6.055)
	4	U	.607	42.040	56.755	.669	42.501	51.008	.634	51.868	66.404	.759	40.287	58.660	.667 (.056)	44.174 (4.611)	58.207 (5.678)
	5	U	.600	41.900	56.618	.671	41.950	49.692	.647	50.958	64.744	.761	40.899	58.979	.670 (.058)	43.927 (4.164)	57.508 (5.561)
	2	NU	.623	41.886	55.947	.663	41.642	50.818	.633	52.068	66.590	.763	40.617	59.384	.670 (.055)	44.053 (4.747)	58.185 (5.908)
	3	NU	.620	42.263	56.812	.668	42.857	51.533	.631	53.062	67.852	.769	41.373	59.540	.672 (.058)	44.889 (4.845)	58.934 (6.081)
	4	NU	.607	42.031	56.745	.669	42.466	51.039	.634	51.909	66.504	.759	40.343	58.732	.667 (.056)	44.187 (4.621)	58.255 (5.708)
	5	NU	.600	41.885	56.601	.671	41.928	49.723	.647	51.011	64.844	.761	40.935	59.032	.670 (.058)	43.940 (4.186)	57.550 (5.589)
	1	R	.653	60.835	71.392	.710	52.090	62.045	.628	67.895	78.856	.738	69.695	75.320	.683 (.043)	62.629 (7.069)	71.903 (6.468)
$\gamma = 1$	1	–	.916	23.447	26.436	.871	13.994	18.129	.902	35.315	42.126	.971	48.344	50.617	.915 (.035)	30.275 (13.143)	34.327 (13.150)
	2	U	.900	28.828	33.029	.880	18.583	22.656	.925	39.274	45.149	.989	59.174	60.026	.923 (.040)	36.465 (15.320)	40.215 (14.366)
	3	U	.896	30.804	35.148	.881	19.492	23.743	.938	36.748	42.294	.990	57.829	58.516	.926 (.041)	36.218 (14.216)	39.925 (12.999)
	4	U	.889	30.876	35.549	.872	21.475	26.089	.935	37.484	42.966	.994	57.871	58.397	.922 (.047)	36.926 (13.636)	40.750 (12.180)
	5	U	.882	31.248	35.738	.868	21.320	25.883	.936	37.615	43.059	.992	58.773	59.318	.919 (.047)	37.239 (14.002)	41.000 (12.584)
	2	NU	.902	28.789	32.947	.880	18.497	22.565	.925	39.278	45.150	.989	59.007	59.861	.924 (.039)	36.393 (15.287)	40.131 (14.347)
	3	NU	.897	30.805	35.137	.882	19.510	23.775	.938	36.973	42.482	.990	57.779	58.462	.927 (.041)	36.267 (14.193)	39.964 (12.978)
	4	NU	.890	30.839	35.484	.873	21.367	25.986	.936	37.554	42.999	.994	57.825	58.354	.923 (.046)	36.896 (13.655)	40.706 (12.205)
	5	NU	.884	31.217	35.678	.870	21.261	25.830	.936	37.609	43.019	.992	58.770	59.309	.920 (.047)	37.214 (14.022)	40.959 (12.603)
	1	R	.825	58.539	60.310	.793	42.200	46.818	.890	55.940	61.462	.963	65.023	66.924	.868 (.064)	55.426 (8.491)	58.878 (7.627)
	2	C	.905	27.444	31.356	.881	17.547	21.520	.925	37.373	43.387	.989	58.318	59.229	.925 (.040)	35.171 (15.399)	38.873 (14.510)
	3	C	.900	28.802	32.701	.882	18.039	22.091	.939	34.534	40.310	.990	56.660	57.516	**.928 (.041)**	34.509 (14.381)	38.154 (13.316)
	4	C	.894	28.643	32.867	.874	19.505	23.747	.938	34.613	40.360	.994	56.309	57.011	.925 (.045)	34.768 (13.828)	38.496 (12.579)
	5	C	.888	29.149	33.118	.870	19.259	23.507	.939	34.622	40.252	.992	57.220	57.962	.922 (.047)	35.063 (14.211)	38.710 (12.993)
$\gamma_{\text{opt}} = .9$	1	–	.922	11.997	14.986	.879	15.084	18.011	.898	24.898	31.110	.985	36.191	38.271	.921 (.039)	**22.043 (9.649)**	**25.594 (9.796)**
	2	U	.892	16.642	19.922	.881	15.719	19.009	.923	23.858	30.280	.988	39.919	41.175	.921 (.041)	24.034 (9.895)	27.596 (9.282)
	3	U	.890	18.641	22.543	.876	18.391	21.453	.930	23.965	29.934	.989	41.232	42.249	.921 (.043)	25.557 (9.510)	29.045 (8.549)
	4	U	.883	19.078	23.494	.866	19.766	22.757	.928	23.691	29.686	.991	40.159	41.138	.917 (.047)	25.673 (8.721)	29.269 (7.590)
	5	U	.875	20.091	24.960	.862	18.791	21.614	.933	23.474	29.474	.991	41.433	42.483	.915 (.050)	25.947 (9.288)	29.633 (8.171)
	2	NU	.894	16.565	19.826	.882	15.679	18.961	.923	23.830	30.226	.988	39.809	41.071	.922 (.040)	23.971 (9.873)	27.521 (9.270)
	3	NU	.892	18.588	22.457	.877	18.312	21.353	.930	23.995	29.967	.989	41.230	42.245	.922 (.042)	25.531 (9.534)	29.005 (8.589)
	4	NU	.885	19.043	23.410	.867	19.639	22.621	.929	23.690	29.673	.991	40.229	41.204	.918 (.047)	25.650 (8.781)	29.227 (7.665)
	5	NU	.877	19.983	24.795	.864	18.716	21.530	.933	23.414	29.390	.991	41.416	42.462	.916 (.049)	25.882 (9.318)	29.544 (8.210)
	1	R	.844	47.859	50.120	.817	37.727	40.926	.900	54.008	59.263	.940	55.980	59.071	.875 (.047)	48.893 (7.254)	52.345 (7.791)
$\gamma = .5$	1	–	.871	18.642	23.367	.848	17.735	20.735	.873	27.140	32.733	.930	39.651	43.484	.880 (.298)	25.792 (8.982)	30.080 (9.208)
Unsupervised	–	–	.815	–	–	.810	–	–	.881	–	–	.942	–	–	.862 (.054)	–	–
Supervised	–	–	.891	19.353	25.297	.865	22.048	25.200	.939	18.658	22.473	.971	11.255	14.159	.916 (.041)	17.829 (4.001)	21.782 (4.545)

k: number of target queries (1-to-k mapping), **w**: weighting approach, **U**: uniform, **NU**: non-uniform, **C**: correlation, **R**: random

used versus its extremes. The transferred models can very often estimate the peak of the flu season accurately. This includes the time of occurrence as well as its intensity. Notably, ILI rates in the target countries differ in terms of scale compared to ones of the source, but the proposed models are capable of capturing different scales effortlessly, providing further evidence about the user search behavior similarities among different countries (Section 3.1). At the same time, most models show some inaccuracies, especially during the time periods with very moderate flu circulation (e.g. summer).

4.2 Qualitative analysis

A fair criticism for the proposed framework is that in a practical scenario the optimal values for γ and k cannot be validated. However, we have already demonstrated that the default settings of $\gamma = .5$ and $k = 1$ provide very satisfactory performance in all our case studies. Fig. 3 looks further into this, depicting performance estimates (MAE) for different values of γ. As discussed previously, optimal γ values differ per target country. Interestingly, all error trends are monotonically decreasing (as γ increases) until they reach a minimum, and then begin to monotonically increase. We argue that γ_{opt} reflects on the actual pool of candidate target queries ($\mathcal{P}_{\text{T}}$), although we have a small sample size to be able to empirically prove this. In our data, the average correlation over the average semantic similarity ratio between all source-target query pairs is equal to 1.143, .982 and 2.261, for the FR, ES, and AU tasks respectively. These ratios depend on characteristics of the target queries which we are not controlling for in our approach. They do correlate with the respective optimal γ values (.5, .2, and .9), an insight that can be used to make a more informed choice of γ in future applications of the proposed framework.

Table 5 lists the top-5 query mappings that were the most impactful in the ILI estimates on average during the 10 weeks with the lowest and greatest MAEs (for the optimal transfer models). Impact is determined by the percentage of an estimated ILI rate that is contributed by a query (frequency × weight / estimated ILI rate). The identified pairs during the weeks with the lowest errors are topically coherent (about flu) and in many occasions are accurate translations from the source to the target language. On the other hand, pairs responsible for the largest errors include inaccurate translations that sometimes lead to an off-topic target query selection. For example, "24 hour flu" is mapped to "grippe intestinale" (impact: 13.2%),[9] "child fever" to "sinusitis" (7.7%), and "child temperature" to "warmer" (9.8%). Nevertheless, it is encouraging

[9] "Grippe intestinale" translates to "stomach flu" (formally "viral gastroenteritis").

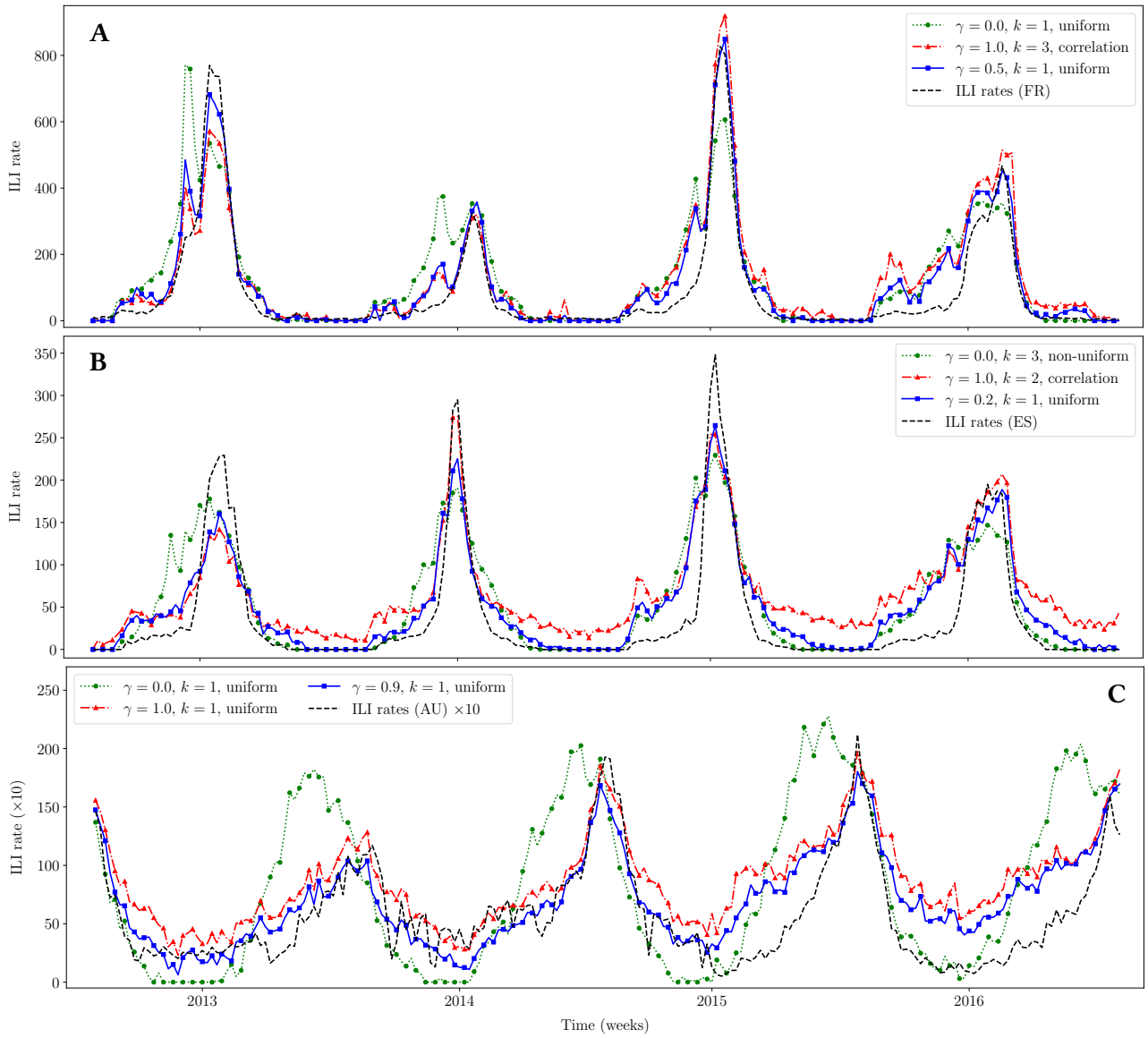


Figure 2: Comparison of transfer learning models for estimating ILI rates in France (A), Spain (B) and Australia (C) with the corresponding actual ILI rates obtained by health agencies in these countries.

that some of these mappings may have been avoided by carefully preprocessing the target query candidates to avoid spurious queries.

The optimal joint similarity transfer models do not improve by increasing the number of target queries ($k > 1$). An interpretation for that might be drawn by the fact that for $k = 1$ at most 77.9% of the selected target queries are unique (at least 22.1% are repetitive selections). Hence, the method seems to be converging to a subset of queries already for $k = 1$. As k increases, the error increases monotonically. This might be due to the existence of various spurious queries in the feature space which are being introduced as additional mappings.

Finally, the choice of adding a non-negativity constraint to the regularized regression function for the source domain (Eq. 1), was also empirically justified. When it is removed, we can learn a more accurate source model for the US, but the MAE on the target countries increases on average by 20.6%, 21.6%, and 20.5% for FR, ES, and AU respectively. This confirms our original assumption that transferring negative weights is a harder task, and thus, error-prone.

5 RELATED WORK

The fundamental properties of transfer learning have been thoroughly discussed in relevant literature [6, 7, 40, 49, 59, 65]. In contrast to traditional machine learning methods, which assume that the training and test data belong to the same domain, i.e. they are drawn from the same feature space and distribution, transfer learning aims to improve the learning function in a target domain by

Table 5: Top-5 target queries (with source mappings) in terms of mean ILI estimate impact (%) in the 10 weeks with the lowest and greatest MAE (all test periods), for all target countries (TC), based on their respective optimal transfer learning models.

TC	Mappings during accurate estimates	Mappings during inaccurate estimates
FR	flu incubation period → grippe durée (10.9), cough fever → la toux (6.3), how to treat flu → comment soigner une grippe (6), fever flu → fièvre de la grippe (5.47), flu treatment → traitement de la grippe (4.95)	24 hour flu → grippe intestinale (13.24), influenza a treatment → grippe traitement (8.07), remedies for colds → rhume de cerveau (6.75), child temperature → température du corps (6.37), child fever → fièvre adulte (6.04)
ES	symptoms of flu → symptômes grippe (9.04), fever flu → con gripe (7.49), cough fever → la tos (6.34), flu incubation period → cuanto dura una gripe (5.19), how to treat a fever → para bajar la fiebre (5.03)	mucinez for kids → tratmiento de la grippe (20.76), child fever → sinusitis (7.76), influenza a treatment → con gripe (7.02), symptoms pneumonia → bronquitis (6.04), child temperature → temperatura corporal (5.62)
AU	treatment for the flu → flu treatment (9.85), cough fever → cough and fever (8.05), flu type → influenza type (5.37), symptoms of flu → symptoms of flu (5.11), flu incubation period → flu incubation period (5.03)	24 hour flu → flu duration (11.51), child temperature → warmer (9.77), how to treat a fever → have a fever (6.94), tamiflu and breastfeeding → flu while pregnant (6.81), robitussin cf → colds (5.18)

transferring knowledge from a related, source domain. This concept has been successfully applied to various tasks, including text classification [14, 16, 22, 48], part of speech tagging [10, 28], machine translation [20, 29], and image classification [19, 30, 71].

In this work, we present a statistical framework for transferring a disease surveillance model from a source country, where supervised learning is applicable, to a target country, where no ground truth is available. We formulate it as a cross-lingual transductive regression task [49], which poses the following challenges: (a) ground truth is not available in the target domain, and (b) features (queries) may not belong in the same feature space due to linguistic or cultural differences. Due to (a), multi-task learning models, such as this solution for ILI [72], cannot be used because they still require partial ground truth from the target domain to capture the relationship between the different tasks [13]. To solve (b), a few studies have attempted to learn a mapping of both source and target languages to the same space [27, 55, 57, 64]. For example, Prettenhofer and Stein used unlabeled documents along with a word translation oracle to automatically induce task-specific, cross-lingual correspondences for cross-lingual text classification [55]. In this paper, we used cross-lingual word embeddings to align different languages [57].

Methods have also been proposed for reducing the distance between the source and target features [48, 70]. For example, Pan *et al.* proposed TCA to learn transfer components across source and target domains in a reproducing kernel Hilbert space using maximum mean discrepancy [48]. Zhou *et al.* constructed a sparse feature transformation matrix based on compressive sensing theory to map the weight vector of classifiers learned from the source domain to the target domain [70]. However, their tasks are very different from the regression task studied in this paper. These models were not able to capture efficiently the time series structure in our data.

Finally, the topic of disease modelling, and in particular of ILI, from online user-generated content has been extensively studied in the literature. The vast majority of methods proposed supervised solutions, using social media or search engine data together with disease rates from an established health authority [15, 21, 32, 33, 35, 38, 50, 52, 53, 67]. A few unsupervised methods have also been attempted, but they showcased moderate accuracy in terms of correlation [31, 51]. Our approach is able to provide accurate estimates without using any ground truth in the target locations.

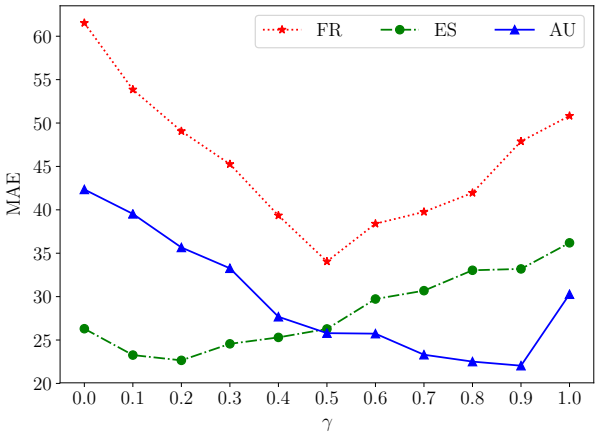


Figure 3: MAE under different γ values for the transfer learning models for FR, ES, and AU ($k = 1$).

6 CONCLUSIONS

Prior work on estimating disease rates from online user-generated content relies heavily on supervised learning models. Such models require ground truth data which is usually provided by public health organizations. Syndromic surveillance data, however, is either sparse or absent from locations with a poor healthcare infrastructure. This is somewhat ironic as it is often stated that web-based approaches hold considerable promise for regions that lack an established health surveillance system. This paper proposes a transfer learning framework as a potential solution to this problem. We leverage semantic and temporal relationships to map a supervised model from a source to a target location. We show that we can obtain a satisfactory performance ($r > .92$ on average) that does not deviate much from a fully supervised model ($\leq 21.6\%$ increase in RMSE), without using any ground truth from the target domain.

There is a number of avenues for future work. It is highly desirable to perform a study where the target country is from a low or middle income region. However, such a study is complicated, since the lack of ground truth data does not allow the performance to be quantified. Nevertheless, a qualitative study that demonstrated ILI estimates that followed an expected seasonal pattern would be of value. Our experiments on regions with ground truth data allowed us to investigate parameters k and γ, i.e. the choice for the one-to-k mapping and the relative weight assigned to the semantic and temporal similarities. Our analysis indicated that a one-to-one ($k = 1$) mapping performed best on average, and that the optimal γ differed per target country. Although we attempted to justify both outcomes, further experiments on other regions are needed to understand the effect of these parameters better.

ACKNOWLEDGMENTS

This work has been supported by the EPSRC grants EP/K031953/1 and EP/R00529X/1. The authors would like to acknowledge RCGP, PHE and SISSS for providing anonymized and aggregated syndromic surveillance data. Special thanks to Amparo Larrauri (SISSS) and Richard Pebody (PHE). We also thank Simon Moura and Peter Hayes for providing constructive feedback.

REFERENCES

[1] C. Alicino, N. L. Bragazzi, V. Faccio, D. Amicizia, D. Panatto, R. Gasparini, G. Icardi, and A. Orsi. 2015. Assessing Ebola-related Web Search Behaviour: Insights and Implications from an Analytical Study of Google Trends-based Query Volumes. *Infectious Diseases of Poverty* 4, 54 (2015), 1–13.

[2] W. Ammar, G. Mulcaire, Y. Tsvetkov, G. Lample, C. Dyer, and N. A. Smith. 2016. Massively Multilingual Word Embeddings. *arXiv Preprint* (2016), arXiv:1602.01925.

[3] H. K. Andreassen, M. M. Bujnowska-Fedak, C. E. Chronaki, R. C. Dumitru, I. Pudule, S. Santana, H. Voss, and R. Wynn. 2007. European Citizens' Use of E-health Services: A Study of Seven Countries. *BMC Public Health* 7, 53 (2007).

[4] M. Artetxe, G. Labaka, and E. Agirre. 2016. Learning Principled Bilingual Mappings of Word Embeddings while Preserving Monolingual Invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2289–2294.

[5] M. Artetxe, G. Labaka, E. Agirre, and K. Cho. 2018. Unsupervised Neural Machine Translation. In *Proceedings of the 6th International Conference on Learning Representations*.

[6] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. 2010. A Theory of Learning from Different Domains. *Machine Learning* 79, 1-2 (2010), 151–175.

[7] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. 2007. Analysis of Representations for Domain Adaptation. In *Proceedings of Advances in Neural Information Processing Systems 19*. 137–144.

[8] A. Benton, R. Arora, and M. Dredze. 2016. Learning Multiview Embeddings of Twitter Users. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Vol. 2. 14–19.

[9] M. Biggerstaff, M. Johansson, D. Alper, L. C. Brooks, P. Chakraborty, D. C. Farrow, S. Hyun, S. Kandula, C. McGowan, N. Ramakrishnan, et al. 2018. Results from the Second Year of A Collaborative Effort to Forecast Influenza Seasons in the United States. *Epidemics* 24 (2018), 26–33.

[10] J. Blitzer, R. McDonald, and F. Pereira. 2006. Domain Adaptation with Structural Correspondence Learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. 120–128.

[11] G. Boivin, I. Hardy, G. Tellier, and J. Maziade. 2000. Predicting Influenza Infections during Epidemics with Use of a Clinical Case Definition. *Clinical Infectious Diseases* 31, 5 (2000), 1166–1169.

[12] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association of Computational Linguistics* 5, 1 (2017), 135–146.

[13] R. Caruana. 1998. Multitask Learning. In *Learning to Learn*. Springer, 95–133.

[14] M. Chen, K. Q. Weinberger, and J. Blitzer. 2011. Co-Training for Domain Adaptation. In *Proceedings of Advances in Neural Information Processing Systems 24*. 2456–2464.

[15] A. Culotta. 2010. Towards Detecting Influenza Epidemics by Analyzing Twitter Messages. In *Proceedings of the 1st Workshop on Social Media Analytics*. 115–122.

[16] W. Dai, G. Xue, Q. Yang, and Y. Yu. 2007. Transferring Naive Bayes Classifiers for Text Classification. In *Proceedings of the 22nd International Conference on Artificial Intelligence*. 540–545.

[17] M. De Choudhury, M. Gamon, S. Counts, and E. Horvitz. 2013. Predicting Depression via Social Media. In *Proceedings of the 7th International AAAI Conference on Weblogs and Social Media*. 128–137.

[18] G. Dinu, A. Lazaridou, and M. Baroni. 2014. Improving Zero-shot Learning by Mitigating the Hubness Problem. *arXiv Preprint* (2014), arXiv:1412.6568.

[19] L. Duan, D. Xu, and I. W. Tsang. 2012. Learning with Augmented Features for Heterogeneous Domain Adaptation. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*. 667–674.

[20] G. Foster, C. Goutte, and R. Kuhn. 2010. Discriminative Instance Weighting for Domain Adaptation in Statistical Machine Translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. 451–459.

[21] J. Ginsberg, M. H. Mohebbi, R. S. Patel, L. Brammer, M. S. Smolinski, and L. Brilliant. 2009. Detecting Influenza Epidemics using Search Engine Query Data. *Nature* 457, 7232 (2009), 1012–1014.

[22] X. Glorot, A. Bordes, and Y. Bengio. 2011. Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach. In *Proceedings of the 28th International Conference on Machine Learning*. 513–520.

[23] G. H. Golub and C. Reinsch. 1970. Singular Value Decomposition and Least Squares Solutions. *Numerische Mathematik* 14, 5 (1970), 403–420.

[24] J. Gomide, A. Veloso, W. M. Jr., V. Almeida, F. Benevenuto, F. Ferraz, and M. Teixeira. 2011. Dengue Surveillance Based on a Computational Model of Spatio-temporal Locality of Twitter. In *Proceedings of the 3rd International Web Science Conference*. Article 3, 3:1–3:8 pages.

[25] O. Higgins, J. Sixsmith, M. M. Barry, and C. Domegan. 2011. *A Literature Review on Health Information Seeking Behaviour on the Web: A Health Consumer and Health Professional Perspective*. Technical Report.

[26] A. E. Hoerl and R. W. Kennard. 1970. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics* 12 (1970), 55–67.

[27] J.-T. Huang, J. Li, D. Yu, L. Deng, and Y. Gong. 2013. Cross-Language Knowledge Transfer using Multilingual Deep Neural Network with Shared Hidden Layers. In *Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. 7304–7308.

[28] J. Jiang and C. Zhai. 2007. Instance Weighting for Domain Adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. 264–271.

[29] P. Koehn and J. Schroeder. 2007. Experiments in Domain Adaptation for Statistical Machine Translation. In *Proceedings of the 2nd Workshop on Statistical Machine Translation*. 224–227.

[30] B. Kulis, K. Saenko, and T. Darrell. 2011. What you Saw is not What you Get: Domain Adaptation using Asymmetric Kernel Transforms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1785–1792.

[31] A. Lamb, M. J. Paul, and M. Dredze. 2013. Separating Fact from Fear: Tracking Flu Infections on Twitter. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 789–795.

[32] V. Lampos and N. Cristianini. 2010. Tracking the flu pandemic by monitoring the social Web. In *Proceedings of the 2nd International Workshop on Cognitive Information Processing*. IEEE Press, 411–416.

[33] V. Lampos and N. Cristianini. 2012. Nowcasting Events from the Social Web with Statistical Learning. *ACM Transactions on Intelligent Systems and Technology* 3, 4 (2012), 1–22.

[34] V. Lampos, T. De Bie, and N. Cristianini. 2010. Flu Detector – Tracking Epidemics on Twitter. In *Proceedings of the 2010 European Conference on Machine Learning and Knowledge Discovery in Databases: Part III (ECML PKDD'10)*. Springer, 599–602.

[35] V. Lampos, A. C. Miller, S. Crossan, and C. Stefansen. 2015. Advances in nowcasting influenza-like illness rates using search query logs. *Scientific Reports* 5, 12760 (2015).

[36] V. Lampos, D. Preoţiuc-Pietro, and T. Cohn. 2013. A user-centric model of voting intention from Social Media. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 993–1003.

[37] V. Lampos, E. Yom-Tov, R. Pebody, and I. J. Cox. 2015. Assessing the Impact of a Health Intervention via User-Generated Internet Content. *Data Mining and Knowledge Discovery* 29, 5 (2015), 1434–1457.

[38] V. Lampos, B. Zou, and I. J. Cox. 2017. Enhancing Feature Selection Using Word Embeddings: The Case of Flu Surveillance. In *Proceedings of the 26th International Conference on World Wide Web*. 695–704.

[39] O. Levy, A. Søgaard, and Y. Goldberg. 2017. A Strong Baseline for Learning Cross-Lingual Word Embeddings from Sentence Alignments. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, Vol. 1. 765–774.

[40] Y. Mansour, M. Mohri, and A. Rostamizadeh. 2009. Domain Adaptation: Learning Bounds and Algorithms. *arXiv Preprint* (2009), arXiv:0902.3430.

[41] T. Mikolov, Q. V. Le, and I. Sutskever. 2013. Exploiting Similarities among Languages for Machine Translation. *arXiv Preprint* (2013), arXiv:1309.4168.

[42] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of Advances in Neural Information Processing Systems 26*. 3111–3119.

[43] G. J Milinovich, G. M Williams, A. C. A. Clements, and W. Hu. 2014. Internet-based Surveillance Systems for Monitoring Emerging Infectious Diseases. *The Lancet Infectious Diseases* 14, 2 (2014), 160–168.

[44] A. Mogadala and A. Rettinger. 2016. Bilingual Word Embeddings from Parallel and Non-parallel Corpora for Cross-language Text Classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 692–702.

[45] M. Mohebbi, D. Vanderkam, J. Kodysh, R. Schonberger, H. Choi, and S. Kumar. 2011. Google Correlate Whitepaper. https://www.google.com/trends/correlate/whitepaper.pdf.

[46] A. S. Monto, S. Gravenstein, M. Elliott, M. Colopy, and J. Schweinle. 2000. Clinical Signs and Symptoms Predicting Influenza Infection. *Archives of Internet Medicine* 160, 21 (2000), 3243–3247.

[47] N. Mrkšić, I. Vulić, D. Ó Séaghdha, I. Leviant, R. Reichart, M. Gašić, A. Korhonen, and S. Young. 2017. Semantic Specialisation of Distributional Word Vector Spaces using Monolingual and Cross-lingual Constraints. *arXiv Preprint* (2017),

arXiv:1706.00374.

[48] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. 2009. Domain Adaptation via Transfer Component Analysis. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*. 1187–1192.

[49] S. J. Pan and Q. Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359.

[50] M. J. Paul and M. Dredze. 2011. You Are What You Tweet: Analysing Twitter for Public Health. In *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media*. 265–272.

[51] Michael J. Paul and Mark Dredze. 2014. Discovering Health Topics in Social Media Using Topic Models. *PLOS One* 9, 8 (2014), e103408.

[52] M. J. Paul, M. Dredze, and D. Broniatowski. 2014. Twitter Improves Influenza Forecasting. *PLOS Currents* 6 (2014).

[53] P. M. Polgreen, Y. Chen, D. M. Pennock, F. D. Nelson, and R. A. Weinstein. 2008. Using Internet Searches for Influenza Surveillance. *Clinical Infectious Diseases* 47, 11 (2008), 1443–1448.

[54] D. Preoţiuc-Pietro, S. Volkova, V. Lampos, Y. Bachrach, and N. Aletras. 2015. Studying User Income through Language, Behaviour and Affect in Social Media. *PLOS ONE* 10, 9 (2015), e0138717.

[55] P. Prettenhofer and B. Stein. 2010. Cross-language Text Classification using Structural Correspondence Learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. 1118–1127.

[56] S. Ruder. 2017. A Survey of Cross-Lingual Embedding Models. *arXiv Preprint* (2017), arXiv:1706.04902.

[57] S. L. Smith, D. H. P. Turban, S. Hamblin, and N. Y. Hammerla. 2016. Offline Bilingual Word Vectors, Orthogonal Transformations and the Inverted Softmax. *arXiv Preprint* (2016), arXiv:1702.03859.

[58] R. Tibshirani. 1996. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 58, 1 (1996), 267–288.

[59] L. Torrey and J. Shavlik. 2009. Transfer Learning. *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques: Algorithms, Methods, and Techniques* (2009), 242.

[60] I. Vulić and M.-F. Moens. 2015. Bilingual Word Embeddings from Non-Parallel Document-Aligned Data Applied to Bilingual Lexicon Induction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 2. 719–725.

[61] I. Vulić and M.-F. Moens. 2015. Monolingual and Cross-lingual Information Retrieval Models based on (Bilingual) Word Embeddings. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 363–372.

[62] I. Vulić and M.-F. Moens. 2016. Bilingual Distributed Word Representations from Document-Aligned Comparable Data. *Journal of Artificial Intelligence Research* 55 (2016), 953–994.

[63] M. Wagner, V. Lampos, I. J. Cox, and R. Pebody. 2018. The Added Value of Online User-generated Content in Traditional Methods for Influenza Surveillance. *Scientific Reports* 8, 13963 (2018).

[64] X. Wan. 2009. Co-training for Cross-lingual Sentiment Classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association of Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*. 235–243.

[65] K. Weiss, T. M. Khoshgoftaar, and D. Wang. 2016. A Survey of Transfer Learning. *Journal of Big Data* 3, 1 (2016).

[66] J. Xu, P. Wang, G. Tian, B. Xu, J. Zhao, F. Wang, and H. Hao. 2015. Short Text Clustering via Convolutional Neural Networks.. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 62–69.

[67] S. Yang, M. Santillana, and S. C. Kou. 2015. Accurate Estimation of Influenza Epidemics using Google Search Data via ARGO. *Proceedings of the National Academy of Sciences* 112, 47 (2015), 14473–14478.

[68] M. Ybarra and M. Suman. 2008. Reasons, Assessments and Actions Taken: Sex and Age Differences in Uses of Internet Health Information. *Health Education Research* 23, 3 (2008), 512–521.

[69] P. Zhao and B. Yu. 2006. On Model Selection Consistency of Lasso. *Journal of Machine Learning Research* 7 (2006), 2541–2563.

[70] J. T. Zhou, I. W. Tsang, S. J. Pan, and M. Tan. 2014. Heterogeneous Domain Adaptation for Multiple Classes. In *Artificial Intelligence and Statistics*. 1095–1103.

[71] Y. Zhu, Y. Chen, Z. Lu, S. J. Pan, G.-R. Xue, Y. Yu, and Q. Yang. 2011. Heterogeneous Transfer Learning for Image Classification. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*. 1304–1309.

[72] B. Zou, V. Lampos, and I. J. Cox. 2018. Multi-Task Learning Improves Disease Models from Web Search. In *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 87–96.

[73] B. Zou, V. Lampos, R. Gorton, and I. J. Cox. 2016. On Infectious Intestinal Disease Surveillance using Social Media Content. In *Proceedings of the 6th International Conference on Digital Health*. ACM, 157–161.

[74] H. Zou and T. Hastie. 2005. Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67, 2 (2005), 301–320.

[75] W. Y. Zou, R. Socher, D. Cer, and C. D. Manning. 2013. Bilingual Word Embeddings for Phrase-based Machine Translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1393–1398.

Tortoise or Hare? Quantifying the Effects of Performance on Mobile App Retention

Agustin Zuniga, Huber Flores
Eemil Lagerspetz, Sasu Tarkoma
University of Helsinki, Finland
first.last@helsinki.fi

Jukka Manner
Aalto University, Finland
jukka.manner@aalto.fi

Pan Hui
University of Helsinki, Finland
HKUST, Hong Kong
panhui@cse.ust.hk

Petteri Nurmi
University of Helsinki, Finland
Lancaster University, United Kingdom
petteri.nurmi@cs.helsinki.fi

ABSTRACT

We contribute by quantifying the effect of network latency and battery consumption on mobile app performance and retention, i.e., user's decisions to continue or stop using apps. We perform our analysis by fusing two large-scale *crowdsensed* datasets collected by piggybacking on information captured by mobile apps. We find that app performance has an impact in its retention rate. Our results demonstrate that high energy consumption and high latency decrease the likelihood of retaining an app. Conversely, we show that reducing latency or energy consumption does not guarantee higher likelihood of retention as long as they are within reasonable standards of performance. However, we also demonstrate that what is considered reasonable depends on what users have been accustomed to, with device and network characteristics, and app category playing a role. As our second contribution, we develop a model for predicting retention based on performance metrics. We demonstrate the benefits of our model through empirical benchmarks which show that our model not only predicts retention accurately, but generalizes well across application categories, locations and other factors moderating the effect of performance.

CCS CONCEPTS

• **General and reference** → **Cross-computing tools and techniques**; • **Networks** → *Network performance evaluation*; • **Computing methodologies** → Modeling and simulation.

KEYWORDS

performance evaluation, crowdsensing, data fusion, mobile computing, apps retention, mobile networks, energy consumption.

ACM Reference Format:
Agustin Zuniga, Huber Flores, Eemil Lagerspetz, Sasu Tarkoma, Jukka Manner, Pan Hui, and Petteri Nurmi. 2019. Tortoise or Hare? Quantifying the Effects of Performance on Mobile App Retention. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313428

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313428

1 INTRODUCTION

Nowadays there is an app for almost everything with Apple AppStore, Google Play and other major marketplaces offering millions of apps to users [48]. While the total number of apps on the marketplaces is high, a large fraction of them vanish without ever attracting a significant user base, and the majority of the rest struggle to maintain their user base over time. Specifically, studies on mobile app usage suggest that over a quarter of installed apps are only used once [25], and even apps used for more than a day are unlikely to stay relevant longer than a fortnight [46].

While low retention of apps is well known [9, 10], surprisingly little is known about the conditions that drive users to abandon apps. Indeed, existing research has mostly focused on identifying factors that result in poor user perceptions without quantifying their effect on user behaviour. For example, performance related characteristics and technical problems have been shown to be a leading factor for abandoning apps [15], a major source of frustration, and a common complaint in app reviews [27, 28]. However, *the point at which they result in users abandoning apps* is currently not known. Improving our understanding of the relationship between user behaviour and these factors would be of significant academic and commercial interest as, among others, it would deepen our understanding of mobile interactions and how they are influenced by context [4, 7, 14, 36], provide marketers information about which factors most contributed to the success or failure of an app [12, 23], and allow developers to better understand how to improve their app [3, 40].

Traditionally, mobile app performance has been analyzed by capturing performance metrics either on the network level or using active monitoring on the user's mobile device [1, 5, 24]. Unfortunately both approaches are limited in their capability to capture and quantify performance and its effect on mobile app usage. Network level measurements only capture performance factors related to networking, ignoring other factors such as energy completely. Additionally, relating network measurements to specific applications is difficult [1]. Active client monitoring captures richer set of performance factors and allows relating them with specific applications, but suffers from limited generality due to performance being sensitive to the current context of the user. For example, ambient temperature and mobility affect an app's energy use [37]

while latency is influenced by mobility, network operator, communication technology and available network infrastructure [20]. Characterizing performance would thus require collecting a large number of samples to ensure all relevant contexts are captured – something that would take prohibitively long or require battery heavy sampling when performed on individual device [20, 22].

In this paper, we contribute by quantifying the relationship between mobile application performance and retention, i.e., whether users are willing to continue using an app. We perform our analysis by fusing two large-scale datasets of *crowdsensed* measurements collected by piggybacking on information collected by mobile apps. Crowdsensing allows reaching a larger pool of users and usage contexts by taking advantage of existing application distribution channels, such as mobile app marketplaces. By piggybacking on mobile app data, the overhead of data collection is minimized and separate instrumentation of the devices is avoided, unlike in intrusive client-side monitoring. The first dataset we consider consists of detailed information about network latency [47] and the second contains information about energy consumption and mobile app usage [32]. To analyze the combined effect of these performance factors, we build on statistical survey analysis methods (hot deck multiple imputation [2, 41]) to fuse the datasets together. Our analysis focuses on energy consumption and latency as they have been shown to be key components in shaping user's perceptions of apps [23, 27]. Indeed, energy drain is known to be a major source of frustration and cognitive burden as users are forced to actively take countermeasures to prolong their battery lifetime [6, 38, 39]. Importance of energy is also reflected by the high popularity of energy saving apps on mobile marketplaces [32]. Latency, in turn, affects user engagement and results in shorter interactions and higher degree of frustration [11, 18]. While importance of these performance parameters has been established beyond doubt, their effect on continued mobile application usage is currently not understood.

Our analysis reveals that both high energy consumption and high degree of latency decrease the likelihood of retaining an app. Conversely, we show that decreasing latency or battery consumption does not guarantee higher likelihood of retention - as long as they are within reasonable standards of performance. However, what is considered reasonable depends on what the users have been accustomed to, with the app category, device characteristics, and network characteristics having an influence. For example, we find the level of expected latency to be lower in Finland than in USA, thanks to faster network infrastructure and consequently differing user expectations. To provide further insights into app retention and abandonment, as our second contribution, we develop a model that *predicts the extent that performance affects retention*. We demonstrate the benefits of our model through empirical benchmarks which show that our model not only predicts retention accuracy, but generalizes well across application categories, locations and other factors moderating the effect of performance.

Summary of Contributions:

- We demonstrate that latency and energy, two key performance metrics for mobile apps, have a significant adverse effect on mobile app retention and abandonment and that this effect is stable across most application categories. However, we also conversely demonstrate that improving latency and energy drain does not guarantee improvements in retention as long as the performance metrics are within reasonable levels.
- We derive *critical points* for popular applications and application categories, demonstrating that the point where performance has a significant effect varies depending on the level of performance people are accustomed to, as well as the functionality and category of the application.
- We further explain our findings through a model that predicts the extent that performance affects retention. Our model achieves an overall error of 1.4 percentage points (measured using Mean Absolute Error MAE) of retention across all data.

Table 1: Summary statistics of application usage and network connectivity datasets.

Dataset	Samples	Users	Apps	Time
NetRadar [47]	875,907	-	-	Jan 1 - Dec 31, 2016
Carat [32]	19,608,938	25,402	48,770	Jul 1 - Dec 31, 2016
Combined	1,000,058 (Latency) 2,819,748 (Energy)	1,241	243	Jul 1 - Dec 31, 2016

2 DATASETS AND PREPARATION

We quantify the effect of performance-related variables on long-term application usage by analyzing two large-scale crowdsensed datasets. Our first dataset, NetRadar [47], contains information about network performance at different locations, whereas the second dataset, Carat [32], contains information about mobile application usage and energy drain of applications.[1] To analyze overall impact of performance, we fuse the two datasets using coarse-grained location and timezone information. While instrumenting a single application to monitor both energy and latency is technically feasible, in practice the functionalities for sampling network or energy differ significantly, which would make attracting sufficiently large user base difficult. Indeed, network profilers are typically used sporadically instead of continuous data collection. Continually collecting network performance measures, on the other hand, would significantly increase energy overhead from sampling and thus be contrary to the original purpose of energy-awareness apps.

After combining the datasets, we select those countries with highest amounts of data for our analysis. In the intersection of the two datasets, 91% of data is from Finland and USA, and 93% of the USA data is from Eastern USA. This is mainly due to demographics of the user populations of the mobile apps which were used to collect measurements. As a result, we focus our analysis in Finland and USA (EST - Eastern Standard Time). In the remainder of the paper we use EST-USA to refer to the measurements collected from USA that are included in our analysis. The datasets considered in our study are summarized in Table 1.

2.1 Network Latency

Latency directly influences the response time of applications and consequently has a major impact on the observed performance. Several application categories, such as on-line gaming and web conferencing [49], require low latency while for many others it can

[1]The datasets are available through separate license agreements. More information can be found at http://carat.cs.helsinki.fi and http://www.netradar.com/.

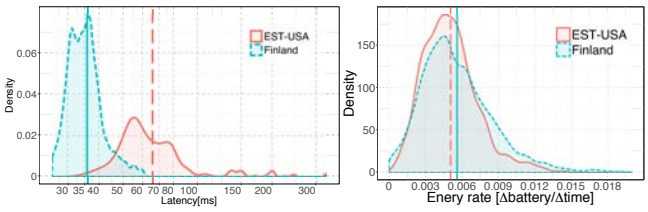
cause frustration even if it is not critical for running the app [11, 18]. For these reasons, we consider latency as a performance - related parameter in our experiments. As source of latency data we consider data collected through NetRadar, a mobile app designed for end-users to help diagnose network conditions [47]. The data considered in our analysis consists of samples collected during cellular network connectivity and contain timestamp, average RTT latency calculated during a 5 second window interval, mobile network information in the form of mobile country code (MCC) and mobile network code (MNC), and GPS location. We restrict our analysis on cellular network connectivity as Wi-Fi has higher bandwidth than cellular technologies and as its performance has less variation overall [16, 17]. Another reason to limit on cellular networks is that this guarantees our analysis to capture a broad range of usage contexts and higher spectrum of mobility patterns.

Figure 1a compares the latency distributions of Finland and EST-USA in the NetRadar dataset. The two locations have distinct latency distributions with Finland having lower expected latency than EST-USA (median 36ms vs. 66ms). The overall variation within Finland is several orders of magnitude smaller with the majority of values being within 45ms. For EST-USA, latency is mostly in the range between 40 to 100ms, but we can observe smaller peaks at around 140ms and even at 200ms. While the values of the distributions differ, the shape of the two distributions is similar with both being long-tailed and skewed towards lower values. To put the values into context, most latencies for Finland are below 70ms which is within LTE network range. For EST-USA, the majority of values is within LTE range, but we can also observe values over 100ms which are likely to correspond to 3G connectivity - or even 2G at the end of the tail. This would suggest there are differences in network infrastructure, or mobile subscriptions, within the two locations. These differences in latency distributions, and characteristics of the underlying network infrastructure motivate us to consider the two locations separately in our analysis.

2.2 Energy Consumption

Our second performance factor, energy, has been shown to be an active source of frustration and a cognitive burden as users actively seek to prolong their battery lifetime [6, 38, 39]. As source of energy consumption measurements we consider Carat [32], a popular mobile energy-awareness application. Carat samples the device whenever battery level changes. Each sample contains current battery level, timestamp, list of running applications and additional attributes, such as device uptime and battery state.

As unit of analysis we consider *energy rates* which correspond to the relative change in battery in a given time interval. Formally, let Δb denote change in battery between successive samples, and let Δt denote the difference in timestamps. Energy rate is then defined as the mean change in battery over the interval, i.e., $e = \Delta b/\Delta t$. We only consider samples where rate is positive (negative rate indicates charging), battery state is not charging (AC or USB), and device uptime has increased from last sample (device has not been turned off in between). We restrict our analysis to samples collected from Android devices as information about running applications cannot be accessed on other platforms and because the sampling granularity on Android devices is better than on iOS devices.



(a) Latency (b) Energy rate

Figure 1: Comparison of latency and energy rate distributions between Finland and Eastern USA.

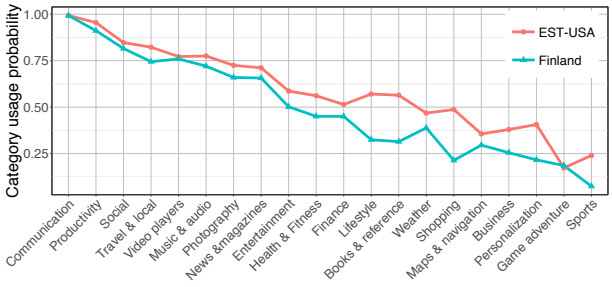


Figure 2: App category use profile differs between locations.

To study energy consumption of application a, we take all rates e_a containing a in the list of currently running apps and calculate the mean rate $\overline{e_a}$ for the application. Since energy consumption fluctuates due to environmental variables (e.g., Wi-Fi and strength of cellular signal) and specific system settings (e.g., screen brightness or use of location tracking), and other currently running applications, we represent energy consumption using the 95% standard error of the mean (SEM) confidence interval [32] given by

$$e_{a,range} = \overline{e_a} \pm h \cdot \frac{\sigma_a}{\sqrt{n_a}}, \quad (1)$$

where $h = 1.96$ is the confidence interval coefficient, σ_a is the standard deviation, and n_a is the number of samples containing a.

Figure 1b compares energy consumption distributions of Finland and EST-USA. Compared to latency, the distributions are closer to each other (mean rate 0.0056 for Finland cf. 0.0051 for EST-USA) and have similar variance, with 44% and 45% of apps exceeding the average in Finland and EST-USA, respectively. To put these values into context, the means correspond to around 5 battery life while most apps are in the range of 5-9 hours. Note that Carat flags applications with very heavy energy drain as hogs and recommends the user to remove them [32]. For this reason applications with very low expected battery life are rare in the dataset.

While the two locations have similar energy consumption distributions, they differ in terms of application usage patterns. This is illustrated in Figure 2, which shows the usage frequency of each category for both locations. EST-USA has higher overall application usage, and the two locations differ in terms of the relative importance of application categories. In our analysis, we separately consider the moderating effect of application category. Motivated by these differences in application category use, we analyze also energy separately across the two locations.

2.3 Retention Rate

The main focus of our work is on analyzing and quantifying how performance related factors affect long-term user behavior. As measure of user behavior we consider n day retention rate, which is the fraction of users continuing to use an app n days since first use. Retention is widely used to measure the success of apps as higher retention corresponds to higher adoption and level of engagement [46]. As source of retention information we use the list of running applications collected by Carat. The list of applications contains all applications running on the device, including those related with the operating system and those pre-installed by the manufacturer. To ensure our analysis focuses on actual usage, we filter the applications by (i) removing all apps for which no category information is found on Google Play; and (ii) removing pre-loaded applications that have not been seen on the foreground, such as Google Exchange Services (com.google.android.exchange).

To estimate retention, let d_a^u denote the number of days between first and last use of an application a by user u in the Carat data. To ensure retention is not influenced by the data collection period, we only consider cases where we have measurements for 7 days before first use, and 7 days after last use. Similarly, to ensure the estimated retention patterns are sufficiently robust, we only consider users that have at least 14 days of data, and apps that have at least 10 users that have used them for more than a day. The n day retention rate of a, denoted r_n^a, is then given by the fraction of users whose retention time d_a^u is higher than n, i.e.,

$$r_n^a = \frac{\#U_{a,n}}{\#U_a} \cdot 100 \tag{2}$$

where U_a is the set of users to use a, and $U_{a,n} \subseteq U_a$ is the subset of users for whom $d_a^u \geq n$. Note that retention rate is cumulative so a user with $d_a = 3$ days also contributes to day 1 and 2 retention.

Table 2 compares the average retention over the first 7 days across the two locations for the 5 most popular applications. From the table we can observe the two locations to have distinct retention patterns with Finland having higher mean retention than EST-USA, but also much higher variation. To highlight differences in usage across the locations, as part of the table, we have also included the number of users, number of samples and summary statistics of the performance variables for the apps. While the number of users tends to be lower in EST-USA than in Finland, we can observe that the number of samples to characterize each performance variable is enough for both locations. In terms of retention behavior, one of the applications, Dropbox, has lower latency but higher energy drain in Finland than in EST-USA, which suggests that it is used in differing situations. These differences in retention and usage patterns further serve to illustrate the need to consider the measurements separately.

2.4 Combining Datasets

Different performance factors are typically closely linked with each other. For example, latency results in energy drain, and networking technology has a major impact on energy use [26, 37, 47]. To analyze and quantify the combined effect of performance factors, we thus need to combine measurements in the two datasets. We perform the combination using hot deck multiple imputation, a widely used method for aligning two datasets that overlap only partially [2, 30, 41]. The idea in hot deck imputation is to fill in missing values (in the combined set) with items that are similar (in the individual data sets). In our analysis we consider the combined dataset as our primary source of analysis as this ensures the usage contexts where performance is captured are comparable across the datasets.

Data Fusion: We combine the datasets using a combination of timestamp, and coarse grained location information given by Mobile Country Code (MCC), Mobile Network Code (MNC) and reverse geocoding of the GPS (time zone from the cellular coverage). The sampling periods of the datasets differ and hence we first need to align them temporally. We perform the alignment by creating hourly bins and mapping each sample in NetRadar and Carat to the closest bin. Next, we match the (MCC, MNC, Time zone) tuples across the two datasets in each bin and calculate hourly latency values for a given location as medians of all matching measurements. The measurements in the combined dataset are summarized in Table 1. In total, the combined dataset comprises 243 applications and 1,241 users from July to December, 2016. This is translated in terms of samples to $1,000,058$ measurements for analyzing latency, and $2,819,748$ measurements for analyzing energy. The reason for differing sample counts for energy and latency is that we perform the matching separately for each application and category considered in our analysis. As the energy dataset is originally larger, this results in a higher total sample count of energy.

Validity: We demonstrate the validity of the combined dataset by comparing statistical characteristics extracted from the combined data against those extracted from the individual datasets. First, we compare mean latency and energy of the individual datasets to those of the combined set. The mean and standard deviation values for both latency (Finland, mean = 38.27, SD = 12.31; EST-USA, mean = 87.45, SD = 61.34; combined, Finland mean = 36.7, SD = 6.6; EST-USA, mean = 84.4, SD = 53.4) and energy (Finland, mean = 0.0053, SD = 0.012; EST-USA, mean = 0.0059, SD = 0.016; combined, Finland mean = 0.0056, SD = 0.003; EST-USA, mean = 0.0051, SD = 0.0023) are closely aligned, suggesting that the statistical characteristics of the individual datasets are preserved in the fusion. Second, we compare the sample distributions between the combined and individual datasets using Kolmogorov-Smirnov distribution tests. No statistically significant differences were found (latency: Finland $KS = 0.104$; EST-US $KS = 0.096$, $p > 0.05$; energy: Finland $KS = 0.04$; EST-US $KS = 0.05$, $p > 0.05$).

Representativeness: We assess whether the energy distribution of Eastern USA is representative of USA as a whole by comparing energy distributions of all samples from USA against those matched to Eastern USA based on timezone information. Again, no significant differences were found ($KS = 0.06$, $p > 0.05$).

2.5 Privacy

We consider only aggregate-level data derived from anonymous user records. We consider country level granularity due to privacy reasons and the need for additional application permissions for collecting finer grained location information of individual users. For Carat, the privacy protection mechanisms are detailed in [32] and for NetRadar in [47]. Users of both apps are informed about the collected data and consent from their devices.

Table 2: Retention rate variation, energy and latency, and influence of app performance in retention for top 5 categories (a) and applications (b) together with statistical analysis results of the importance of performance on retention. Darker colors reflect statistical significance ($p \leq 0.05$).

(a)		Retention(r)			Energy			Latency		Significance: Energy			Significance: Latency		
Category	Location	Mean	St.dev.	Users	Mean	St.dev.	Samples	Median	Samples	Day 1	Day 7	Day 15	Day 1	Day 7	Day 15
Communication (C)	Finland	94.1	2.3	993	0.0049	0.0014	3,184,554	34	1,197,260	0.028	0.005	0.006	0.003	0.685	0.921
	EST-USA	78.1	7.5	83	0.0052	0.0010	474,340	79	4,044	0.019	0.026	0.244	0.313	0.001	0.053
Productivity (P)	Finland	90.1	3.8	719	0.0049	0.0014	1,726,808	37	622,071	0.654	0.600	0.378	0.033	0.001	0.002
	EST-USA	78.1	7.5	63	0.0048	0.0018	328,787	79	1,654	0.263	0.636	0.756	3.0E-04	1.0E-04	0.584
Tools (T)	Finland	87.5	4.8	697	0.0059	9.0E-04	1,614,639	36	606,531	3.0E-04	0.001	1.0E-04	2.0E-04	1.0E-05	0.007
	EST-USA	76.8	8.1	73	0.0050	0.0015	376,269	89	2,585	0.059	0.005	5.0E-04	0.499	0.029	0.016
Social (S)	Finland	88.7	4.3	684	0.0056	0.0016	1,367,938	34	53,2110	0.223	0.284	0.027	0.010	0.246	0.0669
	EST-USA	76.4	8	68	0.0051	0.0011	294,461	68	2,779	2.0E-04	4.0E-05	0.003	0.0612	0.022	0.099
Music (M)	Finland	79.7	7.2	552	0.0051	0.0020	292,685	35	118,019	0.004	0.050	0.304	0.803	0.288	5.0E-04
	EST-USA	64.4	12.3	42	0.0037	0.0011	118,454	101	885	0.027	0.389	0.908	0.244	0.007	0.013
(b)		Retention(r)			Energy			Latency		Significance: Energy			Significance: Latency		
App	Location	Mean	St.dev.	Users	Mean	St.dev.	Samples	Median	Samples	Day 1	Day 7	Day 15	Day 1	Day 7	Day 15
Whatsapp (C)	Finland	98.3	0.8	815	0.0047	0.0014	980,097	35	405,203	0.028	0.040	0.011	0.007	0.066	0.109
	EST-USA	66.5	13.8	25	0.0050	3.0E-04	45,281	79	588	0.315	0.194	0.724	0.460	0.293	0.125
Facebook (C) messenger	Finland	98	0.9	617	0.0039	0.0018	1,109,947	32	389,834	0.027	0.107	0.007	0.203	0.638	0.381
	EST-USA	95	2.1	55	0.0052	0.0016	239,841	108	2,178	0.050	0.017	0.011	0.186	0.096	0.158
Facebook (S) app.	Finland	96.9	1.5	565	0.0036	0.0021	955,718	37	345,073	0.239	0.431	0.022	0.009	0.001	0.002
	EST-USA	86.7	5.4	59	0.0060	1.0E-04	225,832	77	1897	0.013	0.004	0.009	0.035	0.050	0.010
Dropbox (P)	Finland	94	2.6	428	0.0056	0.0019	606,958	40	204,430	0.665	0.478	0.792	0.039	0.004	1.0E-04
	EST-USA	78.1	8.8	24	0.0038	1.0E-04	102,060	79	316	0.377	0.216	0.134	0.105	0.074	0.313
Twitter (N)	Finland	92.7	3.2	323	0.0042	0.0017	192,232	37	71,067	0.089	0.040	0.147	0.231	0.065	0.232
	EST-USA	67.8	13.8	26	0.0045	0.0012	38,192	70	314	0.471	0.033	0.077	0.030	0.041	0.198

3 QUANTIFYING EFFECT OF PERFORMANCE ON RETENTION

In this section, we *quantify*, for the first time, the impact of performance related factors on long-term user behavior, as reflected by retention. We focus on latency and energy as the main performance related variables and analyze them using the Netradar and Carat datasets described in the previous section (see Sections 2.1 and 2.2). We first consider the impact of latency and energy individually, showing they indeed have a significant influence on retention. We proceed to quantify the point where the effect of performance becomes significant, demonstrating that we can identify a *critical* point beyond with performance has a clear effect. We also demonstrate that the effects of energy and latency are moderated by user expectations and by application functionality. We end the section by analyzing the combined effect of latency and energy, showing them to have a complex relationship where neither variable alone is capable of explaining retention.

3.1 Performance Influences Retention

We begin our analysis by demonstrating and quantifying the overall influence of latency and energy as *individual* performance-related variables on retention, analyzing their combined effect in Sec. 3.6. Both latency and energy have been shown to affect user experience [27], and hence to have an indirect effect on long term user behavior. However, whether they have a direct effect on retention has not been previously established. We assess overall effect using Kruskal-Wallis test and considering the five most popular application categories (Communication, Productivity, Tools, Music & Audio, and Social) and applications (Dropbox, Facebook Messenger, Whatsapp, Facebook, Twitter). For each day of the retention period ($1-15$ days), we compare the mean performance of those that stop using the application and those that retain it.

Table 2 shows results of statistical tests at both category-level and app-level together with corresponding retention and performance values. From the results, we can observe that performance indeed affects retention, but the effect is moderated by application category and popularity of the app. We can also observe the effects to reflect differing interaction patterns across application categories. For example, messaging apps (Facebook messenger and Whatsapp) that require users to wait for response are not influenced by latency but energy drain has significant effect on their retention. On the other hand, productivity apps, which tend to be used for shorter periods of time, demonstrate significant effect for latency but not for energy. Comparing effects across number of days, we can observe users to have different levels of tolerance for poor performance depending on application category. For example, Music apps show no effect on latency at day 1, and even at day 7 they only show an effect in Eastern USA where latency is higher than in Finland. However, at day 15 latency has a significant effect for both locations. Similarly, effects of energy are higher for later days for both Facebook and Twitter, suggesting users are willing to tolerate more performance issues with them – potentially because other factors, such as user experience, are more important during the first few days.

Table 3: Retention rate difference of $high$ and low groups proportions for app categories and apps: L: Latency, E: Energy; O: Overall, C: Communications, P: Productivity, T: Tools, S: Social, M: Music, w: Whatsapp, fm: Facebook Msg., fa: Facebook, d: Dropbox, t: Twitter.

Area, Factor	O	C	P	T	S	M	w	fm	fa	d	t
Finland, L	63	67	58	26	65	67	10	84	30	57	45
Finland, E	69	63	78	50	67	63	50	43	69	78	51
EST-USA, L	24	32	10	18	25	10	10	10	10	10	31
EST-USA, E	52	10	64	25	11	29	10	10	10	10	10

3.2 Level of "Critical Point" in Performance

Previous section demonstrated that both latency and energy have an overall effect on retention. We next analyze the relationship

between performance related variables and retention in more detail, showing that we can identify clear points where decrease in performance results in lower retention. We refer to these points as *critical points* of performance. Conversely, we demonstrate that improving performance beyond this point has no influence on retention.

We carry out the analysis by splitting users into two performance groups using a threshold v on the two performance factor, and compare the retention in the two groups using a test of proportions (i.e., a two-tailed z-test). We iterate over different values of v considering values between the 10^{th} and 90^{th} percentile identifying the range of values where retention is significantly different. We omit the lowest and highest 10 percentiles as these resulted in the smaller group having insufficient data for assessing statistical significance. In the following we refer to the two user groups as *high* and *low* depending on which side of v the average performance of users in the corresponding group is.

Table 3 shows the percentile of latency and energy after which the test of proportions indicates difference in retention rate between the low and high groups to be statistically significant ($p < 0.05$). We show percentiles instead of exact performance values due to the fact that the values vary across categories and applications depending on which samples include the category or application. From the results we can see clear differences in the points where performance start to influence retention. Mirroring the results of previous section, we can see that both the category and application moderate the results Besides the category and application, we can also observe *location* to heavily moderate the level where performance starts to influence retention. In Finland, significant differences start to occur only at higher percentiles, whereas in Eastern USA significances start to appear earlier. As an example, latency higher than 60^{th} percentile has significant effect of retention across all application categories in Finland, whereas in USA the effect is significant already from 30^{th} percentile onwards. Similarly, energy starts to have an effect at much earlier percentile in Eastern USA than in Finland. For latency, this difference can be partially explained by differences in network infrastructure, with users in Finland having lower latency and less variability than users in Eastern USA. However, for energy this is not the case with the distributions being similar across the two locations. Consequently, this suggests that users at different locations either assign different importance to energy or have different levels of tolerance.

3.3 Difference in the Effect of Performance

In the previous section we showed there to be a critical point beyond which performance starts to have a significant effect on retention. We next analyze the extent at which individual performance factors start to have a significant effect on retention during app usage.

To perform this analysis, we first calculate the difference in retention percentage between the high and low groups. Figure 3 shows the results. We identify in the figure the percentile where the changes first become significant and can start to be quantified (critical point depicted as vertical line). Points where the difference is negative correspond to cases where retention drops in line with performance degradation. From the figure we can observe the *strength* of significance to vary considerably across locations, categories, and applications. We also can see that, depending on the category, the percentile at which performance differences become significant varies between energy and latency, with one factor typically having a significant effect much earlier than the other. We also can observe that for a few cases, the performance factor does not seem to influence the retention (gray background). Most of these cases correspond to communications apps, such as Whatsapp and Facebook messenger, which are commonly used within social circles and whose usage is moderated by level of social activity [44]. Even if the performance of these apps would be suboptimal, replacing them would require the user's entire social circle to migrate to a new service, which is unlikely to happen rapidly.

Table 4: Critical Point and ΔP for top 5 categories and apps.

	Critical Point, EST-US			Critical Point, Fin		
Category	Energ.	Lat.	ΔP	Energ.	Lat.	ΔP
Communication	10	27	-17	64	67	-3
Productivity	N/A	10	N/A	79	58	21
Tools	24	28	-4	50	23	27
Social	11	16	-5	67	58	93
Music	29	10	19	64	67	-3
App	Energ.	Lat.	ΔP	Energ.	Lat.	ΔP
Whatsapp	N/A	10	N/A	74	11	63
Facebook Msg.	10	10	0	84	N/A	N/A
Facebook App.	19	10	9	87	30	57
Dropbox	46	10	36	80	57	23
Twitter	70	31	39	N/A	45	N/A

To further analyze these differences in significance, Figure 4 compares the difference in critical points (CP) between energy and latency, represented as difference in percentiles ΔP. In the Figure we consider the collection of all applications across all categories for both, Finland and Eastern USA. Interestingly, when the effect of latency is perceived first, the area of ΔP covers a wider percentile range than in the opposite case. Indeed, when latency is the first to have an effect, energy becomes significant only at much later percentages. This relation is explored in Table 4 for all categories and apps considered in our analysis. We analyze this relation by calculating a (Kendall) correlation between CPs for both factors. We observe a positive correlation between CPs for both categories (0.51, $p = 0.04$) and apps (0.62, $p = 0.05$). From the table, we see indeed greater difference between energy and latency when latency is perceived first to decrease performance. We can observe that this relation is three-times as significant for apps than for categories. Intuitively, energy consumption of a particular application may take a long time for the user to discover, while network conditions can change rapidly within seconds and minutes. Therefore bad latency can be discovered much quicker than high energy consumption. Higher latency may also affect the energy consumption of the device, which can result in retention decreasing faster. Because latency is a shorter-term phenomenon than battery life, the decreased retention is easily attributed to latency instead of both energy consumption and latency.

3.4 Effects on Highly-Rated Apps

Besides performance, app functionality and user interface design can influence user satisfaction and ultimately retention. To demonstrate that effects of performance on retention are robust across

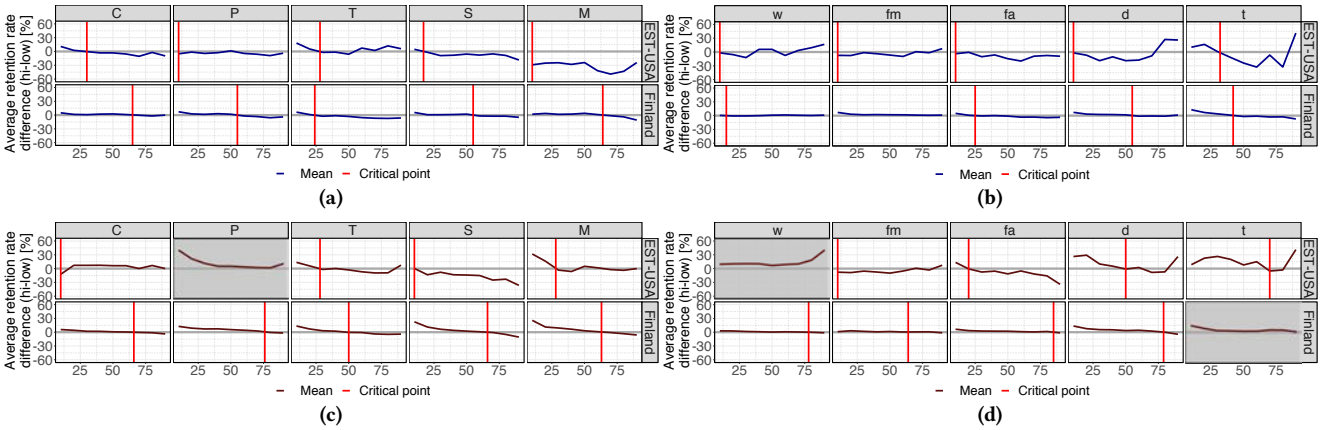


Figure 3: Average retention difference for *high* and *low* groups. (a), (b) Latency, (c), (d) Energy. Categories: O: Overall, C: Communications, P: Productivity, T: Tools, S: Social, M: Music; Apps: o: overall, w: Whatsapp, fm: Facebook Msg., fa: Facebook App., d: Dropbox, t: Twitter

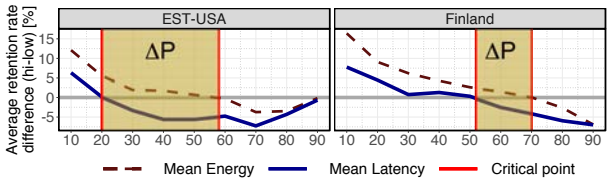


Figure 4: Overall average retention difference for *high* and *low* groups combining latency and energy "critical point" thresholds.

other factors, we analyze 10 applications that are not among the top apps of their categories but have high user satisfaction. As proxy for user satisfaction, we consider the star rating of the apps on Google Play store. As these applications have predominantly received high ratings, users are likely satisfied with the functionalities and user interface design of the apps. We omit most popular applications to avoid possible popularity biases influencing the retention of these apps. For our selection of apps, we also ensured that any negative ratings would not be caused by differences in functionality between commercial and free versions of the app, e.g., due to high amount of advertisement or restricted functionality.

Table 5 shows results together with the applications and their ratings. We applied the same method to calculate the critical point(CP) and difference (*DeltaP*) (see Sec. 3.2). Latency has higher impact than energy for apps that highly depend on displaying on-line content, such as Viaplay. Retention of apps used for personalization, such as Zedge, is more influenced by energy variations. The critical points behave similarly for apps with similar functionality. For example, for both eBay and Aliexpress the critical point on latency is low whereas users are more tolerant to energy. As both applications focus on online shopping are used only intermittently, the importance of latency is understandable. However, for utility apps, such as AVG and Avast, no clear patterns can be identified. Indeed, Avast is more sensitive to latency whereas AVG is more sensitive to energy. In summary, our results show that even for applications with high user ratings, clear critical points can be identified, suggesting that performance indeed has a major influence on their retention. Our analysis also suggests that the relative importance of latency and energy is dependent on the functionality of the app.

Table 5: Effect of performance in case study apps. r: retention, CP: Critical Point ,E: Energy, L: Latency; App: ag: AVG antivirus, aa: Avast antivirus, ac: Avast cleanup, f: Firefox, ae: Aliexpress, e: Ebay, v: Viaplay, h: Here WeGo, s: Sports tracker, z: Zedge; Categories: T: Tools, C: Communication, SH: Shopping, E: Entertainment, MA: Maps, H: Health, PE: Personalization.

Categ	App	Google ⋆	r	CP,E	CP,L	Δ*P*	Δ*P*, graph
T	ag	4.5	90.2	10	22	-12	
T	aa	4.5	95.2	33	14	19	
T	ac	4.5	90	21	12	9	
C	f	4.4	84.1	37	90	-53	
SH	ae	4.6	88.2	71	31	40	
SH	e	4.4	79.2	49	10	39	
E	v	4.0	84.3	58	17	41	
MA	h	4.4	71.2	50	28	22	
H	s	4.5	70.6	56	21	35	
PE	z	4.6	92	10	73	-63	

3.5 Other Factors

To further demonstrate the robustness of our findings, we next demonstrate that the effect of performance is robust across other factors influencing user perceptions and retention.

Data Freshness: We repeated our analysis for effect of energy considering a snapshot of Carat data collected between May 2018 and October 2018. We consider the same five categories and applications as in Sec. 3.1. The mean retention of all categories and apps decreased compared to 2016 with particularly Tools and Dropbox witnessing steep decline. The critical point shifted to a higher percentile for communication (9 percentiles) and music and audio (5 percentiles), but decreased for the other three categories (percentile differences: 11 for productivity, 5 for social, 40 for tools). For the individual apps, the critical point shifted to a lower percentile for four

apps. The sole exception was Twitter, for which we were unable to obtain a clear critical point earlier. In the 2018 data, the critical point is at 66^{th} percentile, which is in line with the other apps. The results thus suggest that the effect of energy on retention is robust over time. In fact our results suggest it has increased in importance since most critical points have shifted to lower percentiles.

Robustness against Energy Saving Mechanisms: To rule out potential biases resulting from the fact that our application data was collected using an energy-awareness app, we repeated our analysis on all applications with energy rates lower than the average energy rate across all apps. For these apps neither Carat nor energy-efficiency mechanisms of the operating system trigger warnings, suggesting the effects of performance are direct result of user perceptions while using the app. The overall effect of energy persists. However, since their overall energy usage is low compared to other apps, critical points shift on average to a 17 percentile higher point.

3.6 Combined Effect of Latency and Energy

We next assess the combined effect of energy and latency by performing a cost-benefit analysis that looks at the combined effect on retention when the importance of individual factors is varied. To perform the analysis, we define a linear cost function that determines the overall effect of the two performance variables as a weighted combination of their individual effect. We consider different relative weightings to see how the importance of individual factors affects retention. Formally, let r_l and r_e denote the differences in retention between the *high* and *low* groups (See Section 3.2), and lw and ew the weights of latency and energy, respectively. Given energy e and l, we estimate retention for a given performance level, denoted $R(e, l)$, using

$$R(e, l) = \frac{r_e \cdot ew + r_l \cdot lw}{ew + lw}. \quad (3)$$

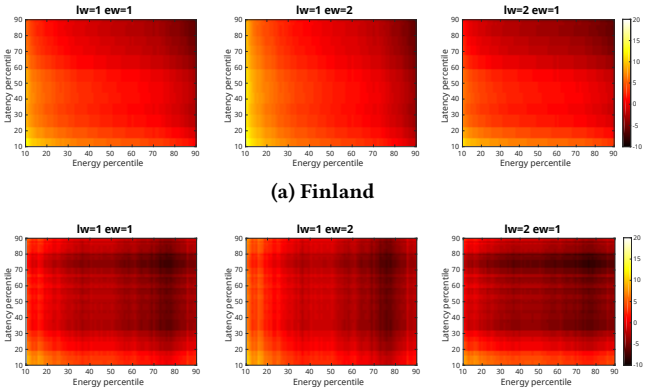
Figure 5 shows the results of our analysis as series of heatmaps. Each heatmap shows the combined effect of latency (y-axis) and energy (x-axis) on retention for different percentiles (10-90) and different weights lw and ew. In the figure, lighter colors reflect retention improvement and darker ones worsened retention. The scale is in percentage units of retention.

From the figure we can observe the effect of performance factors to be non-linear with neither variable clearly dominating the other. When latency is twice as important as energy, the effect on retention is slightly higher than in the opposite case. However, even in this case there is a lot of variation and a highly complex relationship between the two performance variables.

4 MODELING RETENTION

Having *quantified* the effect of performance on retention, we next develop a mode for *predicting the degree to which performance affects retention*. The model is important to estimate how users will respond to apps during their evolving life span, e.g., different releases with extra processing and network functionality.

Model specification: In our general model, app retention is influenced by M factors $F_i, |1 <= i <= M$. Each factor F_i has a performance threshold ϵ_i, whose changes affect the overall retention. Thus, ϵ_i depicts the starting point to quantify how incremental poor performance of a factor impacts app retention. By analyzing



(a) Finland

(b) EST-USA

Figure 5: Retention behaviour for performance-related factor given different weights of latency lw and energy ew.

changes in performance relative to ϵ_i, it is possible to estimate the amount of influence that a performance level has on retention. We estimate retention using a step function as depicted in equation 4, where x is the performance value for a factor, e.g., 30 ms for latency; and $g_i(x)$ is an exponential probability function that approximates the retention rate of the factor given expected performance.

$$R_i(x) = \begin{cases} 0, & x \leq \epsilon_i \\ g_i(x), & x > \epsilon_i \end{cases} \quad (4)$$

We then quantify the overall impact on retention by aggregating the influence of each individual factor. The overall impact of app performance on retention, R, is determined by the factor $F_i \in M$ whose influence on retention is highest, i.e. $R = max(R_i)$. The expected retention rate is then calculated from the uninfluenced retention rate curve $r(x)$ by division as follows $r'(x) = r(x)/R$

Experimental Setup: We assess performance of our model by performing a 80/20 split for each country (Baseline). We also perform a cross-country validation between Finland and EST-USA subsets by training our model with data from Finland and predicting EST-USA retention based on expected performance, and vice versa. Following same approach, we also analyze the effect of mixing data from Finland and EST-USA (Mixed) into a single subset to predict retention. We then compare the performance of our model when predicting retention based on combined factors.

Individual factor prediction: The top part of Table 6 shows the results of the baseline. We can observe that our model indeed is able to predict retention values with lower error rate, specially for Finland as it is the country that has most of the data for our analysis. We then explore Cross-country validation. The *Cross-country* sections of Table 6 shows the results for both latency and energy. From the result, we can observe that the error rate increases slightly compared to the country baseline. Our model has an average overall prediction MAE of 2.25, which depicts an overhead of 46% when compared with the baseline. However, we can observe a small error window in retention based expected latency for (EST-USA → Finland), in this case, the error is reduced in 5%. The slightly overhead is due to the fact that critical points in which performance starts to be perceived by users is different in each country. For

Table 6: MAE for categories (model data → predicted), O: Overall, C: Communications, M: Music, P: Productivity, S: Social, T: Tools

Retention estimation	O	C	M	P	S	T
(Baseline) Latency						
Finland (80%) → (20%)	0.91	0.54	1.67	0.98	0.83	1.72
EST-USA (80%)→(20%)	1.90	3.33	10.03	2.86	3.15	2.13
(Baseline) Battery						
Finland (80%) → (20%)	0.63	0.30	0.75	0.73	1.12	1.08
EST-USA (80%)→(20%)	1.51	2.49	3.87	1.69	3.70	2.77
(Cross-country) Latency						
Finland → EST-USA	3.58	5.00	31.99	3.76	7.13	2.29
EST-USA → Finland	1.66	6.49	37.06	3.39	8.15	2.01
(Cross-country) Battery						
Finland → EST-USA	2.29	5.84	4.98	2.21	15.49	4.61
EST-USA → Finland	1.53	1.30	5.67	2.18	21.29	4.18
(Mixed Finland + EST-USA)						
Latency (80%)→(20%)	0.82	0.50	1.10	0.85	0.61	1.49
Battery (80%)→(20%)	0.28	0.27	0.83	0.56	1.11	1.04
(Mixed Finland + EST-USA)						
Combined Latency+						
Battery (80%)→(20%)	0.29	0.45	0.97	0.35	0.47	0.51

instance, when training our model with latency data from Finland to predict EST-USA, we can observe that the observed latency in Finland is between 30 − 45 ms while the ground truth of EST-USA consists of values around 51 − 147 ms. Clearly, the accuracy prediction of our model is reduced due to this issue.

On the other hand, when analyzing a specific application category, we observe a higher error rate, particularly when trying to generalize the data from EST-USA to Finland. The errors for categories are influenced by the number of samples and applications included in the subset that is used to train the model (see Table 2). Additionally, the dominant applications of each category can differ in the two locations. For instance, in the case of the Music category, most of the collected data for Finland is from the *Spotify* app, while in the case of EST-USA, most of the data is from the *Pandora Music* app (not available for installation in Finland), which explains the higher rate of error for the Music category. Similarly, the Tools category contains significant variation across the locations which explains worse predictive performance. However, for categories with similar usage patterns the results are well aligned across the two countries, like in the case of the Productivity category.

We then proceed to analyze the performance of our model when mixing the data of Finland and EST-USA to predict retention. The bottom of Table 6 shows the results. We can observe that our model improves significantly when mixing data from both countries. Indeed, we can observe very accurate predictions with marginal errors up to 1.49 for all the categories and for both factors.

Combined factor prediction: We analyze the effect on retention when multiple performance factors are taken into consideration. Since the accuracy of the prediction clearly is improved when mixing the data of both countries, we perform a 80/20 validation using both mixed datasets for both energy and latency. In the combined effect, the value of retention is constrained by the factor that influences the performance the most. In other words, the factor that is perceived first by users. Table 6 also shows the results for the combined prediction. Compared to the results of individual factors (Baseline, Cross-country and Mixed), we can observe an improvement in overall retention prediction. We can observe a maximum marginal error of 0.51 for all the categories, which depicts around a 50% reduction in error when compared with our mixed model that performs the best in the individual factor analysis.

5 DISCUSSION

On data validity: The Carat application used as source of energy measurements records samples whenever the battery level changes, and may not be able to record data when the phone is in sleep mode, depending on the operating system version. Therefore, the data we see here is biased towards active use, and the resulting battery life values represent the remaining time for actively using the device with a given application running 100% of the time. In the dataset, the most common e (around 0.005) represents an active battery life of 5 hours. To mitigate these biases, we ensure selecting the location and apps with the most samples to foster better characterization of performance factors. Similar considerations apply to the NetRadar dataset used as source of latency data, which is predominantly collecting data whenever users explicitly request network performance assessment or periodically at user configurable intervals (between 1 and 120 minutes).

Data Quantity: Our validation of the retention prediction model suggests that number of training samples is critical for ensuring high quality predictions. In the case of Finland, data from several hundreds of users was obtained while for US only few tens of users were retained after data fusion. Our data was collected from two mobile applications that have been in long-term usage worldwide, suggesting that crowdsensing is indeed essential for capturing sufficient quantity of measurements. However, our results also highlight the difficulties when *multiple* crowdsensing datasets need to be combined in that their intersection might be small, limiting the power of statistical analyses carried on it.

Fusion of Large-Scale Passive Data: We combined passive measurements from NetRadar and Carat datasets in our analysis. While we ensured that statistically the dataset combination is representative by analyzing and estimating similarity metrics of each dataset individually, we experienced a high reduction of available samples in the fused dataset, mainly due to limited coverage of USA in the NetRadar dataset. However, data fusion is necessary to ensure the quality of contexts that we study. For example, the location, time, operators and communication technologies, etc, must be matched between the records of the two datasets. In other words, there is a trade-off between data data size and data quality. In addition, the individual nature of each dataset (NetRadar - infrastructure, Carat - App usage) also acted as a filter in the combination process, as extra manipulation was required to match attributes in both datasets, e.g., reverse geocoding in the GPS of NetRadar to match the timezones of Carat data records. We were able to model the combined relation that energy and latency have on retention by merging the two datasets. Our methodology also provided insights about the relationship between performance factors that was initially hidden, but revealed when the different sources were combined, similar to recent observations [13] . Specifically, we observe that when latency starts affecting retention, large variations in energy efficiency are possible before retention is affected further. However,

when battery life issues cause lower retention, latency can vary less before retention degrades further.

Energy efficiency models can influence retention: Extending battery life for smartphones is a primary research topic in academy and industry. Mobile devices are equipped with awareness mechanisms that monitor energy consumption based on applications usage and resources utilization. Smartphones then can decide based on this information whether to stop, outsource, or moderate the execution of tasks to save energy. While these mechanisms indeed induce gains in energy as the computation of tasks are reduced, they can foster a collateral damage in the perception that users have towards apps. For instance, iOS devices implement a low power mode mechanism that reduces computation of applications in the background to save energy [43] . This suggests that app performance is reduced and augmented dynamically based on application usage. This can potentially affect retention as the responsiveness of applications is degraded. By using our model in conjunction with energy efficiency models, it is possible to equip smartphones with a smarter mechanism that can save energy without degrading performance to an extent in which it is not tolerable to users anymore. As shown in Sec. 3.5, both the overall effect and existence of critical points are robust against effects of these mechanisms.

Influence of performance depends on usage patterns: Application usage patterns can be very different. For example, Dropbox is a productivity application which mostly runs on the background synchronizing photos. When the user interacts with it, they do so to find or share a file, using it infrequently and for a short period of time, which results in smaller influence of performance degradation. On the other hand, Facebook and Twitter apps provide a continuous feed of updates enabling users to spend hours reading, watching, and interacting with content. Not only does this presenting a larger window of opportunity for performance issues to manifest, but this highlights how different usage patterns are likely to influence the importance of different performance factors.

6 RELATED WORK

Mobile App Quality: Previous research on mobile app quality has focused on exploring user perception of mobile apps. Common techniques include usability studies [42], contextual inquiries [19], sensor data logging [33], interviews [27], and text mining on user reviews [23]. These studies generally focus on the user's perception instead of factors that affect it. Ickin et al. [27] list bugs, performance issues, and poor match with user needs as factors that influence quality perceptions while Chen et al. [12] show that app ratings are key driver for app downloads. While these factors affect app perceptions, studies do not explain how these perceptions translate into changes in behaviour. Our research addresses this gap with existing studies, focusing specifically on the effect of app performance.

Mobile App Performance: There has also been research looking at modeling application performance through the steps involved by the user's task [40] and the response time perceived by the user. Most relevant aspects causing bottlenecks in app performance have been shown to be network communication and processing costs. There have been research efforts to study network performance metrics and relate them with user satisfaction [1, 24]. The general idea is to capture features, such as bitrate, jitter and delay metrics, and to use machine learning to predict user response times [5, 8, 31]. Approaches for dynamic resource augmentation have been proposed to alleviate bottlenecks, relying solely on the device's resources [45] or remote infrastructure [21]. While performance has been modeled and improved, the level at which it starts to influence user perceptions has not been quantified previously.

Effect of Energy: Human interface studies have shown that 80% of mobile users will take steps to improve their battery life [39]. Past work have termed unnecessarily high energy consumption in applications as *energy bugs* [34] and identified their possible causes, such as the environment or settings of the smartphone [37] and programming problems [35]. Regardless of the cause, an app that reduces battery life may end up getting uninstalled by users [3], and possibly replaced by a different app. Many applications try to improve energy consumption by controlling processes on the device, or helping users identify energy hungry applications [32] and raising the level of energy awareness among users [3].

Effect of Network Quality: Network quality has been studied mainly through the influence of latency on user experience in desktop contexts and within specific application categories, e.g. online gaming [11, 49], education and video streaming [29, 50]. Studies on the former have shown that latency reduces session times, and that users actively seek countermeasures to reduce latency. In education, latency has been a barrier for real-time interaction and collaboration, e.g., in Second Life. These findings, however, do not directly translate to mobile apps, since network quality can change abruptly depending on the technology (WiFi, 3G, LTE) and traffic conditions. Moreover, network quality also influences battery life, raising the impact of network quality on users.

7 SUMMARY AND CONCLUSION

The present paper contributed by quantifying the influence of two performance factors, latency and energy, on app retention. We find poor performance to increase likelihood of app abandonment. However, the point at which the effect becomes significant depends on what the user is accustomed to, with app category, device characteristics and network characteristics having an influence. For example, Finland has lower expected latency than USA thanks to faster network infrastructure, and consequently user expectations differ between the two countries. Conversely, we find that improving app performance does not have any positive influence on retention as long as performance is within reasonable performance standards. Our results also indicate that latency and energy have a combined effect on app retention, but in most cases the effect of latency is perceived before energy. Lastly, we built a model that estimates retention based on the expected performance of an application. Performance validation considering cross-country performance demonstrated good performance across all application categories. However, this was moderated by application categories with those categories with limited overlap suffering in performance.

8 ACKNOWLEDGEMENTS

This research has been supported by Academy of Finland grants 296139, 297741, 303825 and 317875, and by the Jorma Ollila Grant 201920082.

REFERENCES

[1] Vaneet Aggarwal, Emir Halepovic, Jeffrey Pang, Shobha Venkataraman, and He Yan. 2014. Prometheus: Toward Quality-of-Experience Estimation for Mobile Apps from Passive Network Measurements. In *Proceedings of the 15th ACM Workshop on Mobile Computing Systems and Applications (HotMobile 2014)*. ACM, 18:1–18:6.

[2] Rebecca R Andridge and Roderick JA Little. 2010. A Review of Hot Deck Imputation for Survey Non-Response. *International Statistical Review* 78, 1 (2010), 40–64.

[3] Kumaribaba Athukorala, Eemil Lagerspetz, Maria von Kügelgen, Antti Jylhä, Adam J. Oliner, Giulio Jacucci, and Sasu Tarkoma. 2014. How Carat Affects User Behavior: Implications for Mobile Battery Awareness Applications. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2014)*. ACM, 1029–1038.

[4] Ricardo A. Baeza-Yates, Di Jiang, Fabrizio Silvestri, and Beverly Harrison. 2015. Predicting The Next App That You Are Going To Use. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining (WSDM 2015)*. ACM, 285–294.

[5] Athula Balachandran, Vyas Sekar, Aditya Akella, Srinivasan Seshan, Ion Stoica, and Hui Zhang. 2013. Developing a Predictive Model of Quality of Experience for Internet Video. In *Proceedings of the Annual ACM International Conference of the Special Interest Group on Data Communication (SIGCOMM 2013)*. ACM, 339–350.

[6] Abhijeet Banerjee, Lee Kee Chong, Sudipta Chattopadhyay, and Abhik Roychoudhury. 2014. Detecting Energy Bugs and Hotspots in Mobile Apps. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2014)*. ACM, 588–598.

[7] Matthias Böhmer, Brent Hecht, Johannes Schöning, Antonio Krüger, and Gernot Bauer. 2011. Falling Asleep with Angry Birds, Facebook and Kindle: A Large Scale Study on Mobile Application Usage. In *Proceedings of the 13th ACM International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI 2011)*. ACM, 47–56.

[8] Peter Brooks and Bjørn Hestnes. 2010. User Measures of Quality of Experience: Why Being Objective and Quantitative is Important. *IEEE Network* 24, 2 (2010), 8–13.

[9] Maura Canavan. 2017. The 8 Mobile App Metrics That Matter [Checklist]. Retrieved February 14, 2019 from http://info.localytics.com/blog/the-8-mobile-app-metrics-that-matter

[10] Andrew Chen. 2018. Number of Apps Available in Leading App Stores as of 3rd Quarter 2018. Retrieved February 14, 2019 from https://andrewchen.co/new-data-shows-why-losing-80-of-your-mobile-users-is-normal-and-that-the-best-apps-do-much-better/

[11] Kuan-Ta Chen, Polly Huang, and Chin-Laung Lei. 2006. How Sensitive are Online Gamers to Network Quality? *Commun. ACM* 49, 11 (2006), 34–38.

[12] Ning Chen, Jialiu Lin, Steven CH Hoi, Xiaokui Xiao, and Boshen Zhang. 2014. AR-miner: Mining Informative Reviews for Developers from Mobile App Marketplace. In *Proceedings of the 36th ACM International Conference on Software Engineering (ICSE 2014)*. ACM, 767–778.

[13] Jamie Condliffe. 2018. Fitness App Data is Revealing Military Bases to Enemy Fighters. Retrieved February 14, 2019 from https://www.technologyreview.com/the-download/610086/fitness-app-data-is-revealing-military-bases-to-enemy-fighters/

[14] Mark de Reuver, Harry Bouwman, Nico Heerschap, and Hannu Verkasalo. 2012. Smartphone Measurement: Do People Use Mobile Applications as They Say They Do?. In *Proceedings of the 11th AIS International Conference on Mobile Business, (ICMB 2012)*. AIS, 2.

[15] Prathap Dendi. 2018. App Attention Index: 80% of Users Delete Apps Due to Poor Performance. Retrieved February 14, 2019 from https://blog.appdynamics.com/news/deletetheapp-digital-leaders-raise-consumer-expectations-new-heights/

[16] Shuo Deng, Ravi Netravali, Anirudh Sivaraman, and Hari Balakrishnan. 2014. WiFi, LTE, or Both?: Measuring Multi-Homed Wireless Internet Performance. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC 2014)*. ACM, 181–194.

[17] Pralhad Deshpande, Xiaoxiao Hou, and Samir R Das. 2010. Performance Comparison of 3G and Metro-Scale WiFi for Vehicular Network Access. In *Proceedings of the 10th SIGCOMM Conference on Internet measurement (IMC 2010)*. ACM, 301–307.

[18] Florin Dobrian, Vyas Sekar, Asad Awan, Ion Stoica, Dilip Joseph, Aditya Ganjam, Jibin Zhan, and Hui Zhang. 2011. Understanding the Impact of Video Quality on User Engagement. In *Proceedings of the ACM International Conference of the Special Interest Group on Data Communication (SIGCOMM 2011)*. ACM, 362–373.

[19] Denzil Ferreira, Jorge Goncalves, Vassilis Kostakos, Louise Barkhuus, and Anind K. Dey. 2014. Contextual Experience Sampling of Mobile Application Micro-Usage. In *Proceedings of the 16th ACM International Conference on Human-Computer Interaction with Mobile & Services (MobileHCI 2014)*. ACM, 91–100.

[20] Huber Flores, Pan Hui, Petteri Nurmi, Eemil Lagerspetz, Sasu Tarkoma, Jukka Manner, Vassilis Kostakos, Yong Li, and Xiang Su. 2018. Evidence-Aware Mobile Computational Offloading. *IEEE Transactions on Mobile Computing* 17, 8 (2018), 1834–1850.

[21] Huber Flores, Xiang Su, Vassilis Kostakos, Jukka Riekki, Eemil Lagerspetz, Sasu Tarkoma, Pan Hui, Yong Li, and Jukka Manner. 2017. Modeling Mobile Code Acceleration in the Cloud. In *Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS 2017)*. IEEE, 480–491.

[22] Huber Flores, Sasu Tarkoma, Petteri Nurmi, and Pan Hui. 2018. MobileCloudSim: A Context-Aware Simulation Toolkit for Mobile Computational Offloading. In *Proceedings of the ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers (UbiComp 2018)*. ACM, 38–41.

[23] Bin Fu, Jialiu Lin, Lei Li, Christos Faloutsos, Jason Hong, and Norman Sadeh. 2013. Why People Hate Your App: Making Sense of User Feedback in a Mobile App Store. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2013)*. ACM, 1276–1284.

[24] Vijay Gabale and Dilip Krishnaswamy. 2015. Mobinsight: On Improving the Performance of Mobile Apps in Cellular Networks. In *Proceedings of the 24th International Conference on World Wide Web (WWW 2015)*. International World Wide Web Conferences Steering Committee, 355–365.

[25] Dave Hoch. 2017. App User Retention Improves in the U.S., but Declines Internationally. Retrieved February 14, 2019 from http://info.localytics.com/blog/app-user-retention-improves-in-the-us

[26] Junxian Huang, Feng Qian, Alexandre Gerber, Z. Morley Mao, Subhabrata Sen, and Oliver Spatscheck. 2012. A Close Examination of Performance and Power Characteristics of 4G LTE Networks. In *Proceedings of the 10th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys 2012)*. ACM, 225–238.

[27] Selim Ickin, Katarzyna Wac, Markus Fiedler, Lucjan Janowski, Jin-Hyuk Hong, and Anind K Dey. 2012. Factors Influencing Quality of Experience of Commonly Used Mobile applications. *IEEE Communications Magazine* 50, 4 (2012), 48–56.

[28] Hammad Khalid, Emad Shihab, Meiyappan Nagappan, and Ahmed E Hassan. 2015. What do Mobile App Users Complain About? *IEEE Software* 32, 3 (2015), 70–77.

[29] Ricky KP Mok, Edmond WW Chan, and Rocky KC Chang. 2011. Measuring the Quality of Experience of HTTP Video Streaming. In *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM 2011)*. IEEE, 485–492.

[30] Teresa A Myers. 2011. Goodbye, Listwise Deletion: Presenting Hot Deck Imputation as an Easy and Effective Tool for Handling Missing Data. *Communication Methods and Measures* 5, 4 (2011), 297–310.

[31] Dushyanth Narayanan, Jason Flinn, and Mahadev Satyanarayanan. 2000. Using History to Improve Mobile Application Adaptation. In *Proceedings of the Third IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2000)*. IEEE, 31–40.

[32] Adam J Oliner, Anand P Iyer, Ion Stoica, Eemil Lagerspetz, and Sasu Tarkoma. 2013. Carat: Collaborative Energy Diagnosis for Mobile Devices. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys 2013)*. ACM, 10.

[33] Antti Oulasvirta, Tye Rattenbury, Lingyi Ma, and Eeva Raita. 2012. Habits Make Smartphone Use More Pervasive. *Personal and Ubiquitous Computing* 16, 1 (2012), 105–114.

[34] Abhinav Pathak, Y. Charlie Hu, Ming Zhang, Paramvir Bahl, and Yi-Min Wang. 2011. Fine-Grained Power Modeling for Smartphones Using System Call Tracing. In *Proceedings of the Sixth ACM International Conference on Computer Systems (EuroSys 2011)*. ACM, 153–168.

[35] Abhinav Pathak, Abhilash Jindal, Y. Charlie Hu, and Sam Midkiff. 2012. What is Keeping My Phone Awake? Characterizing and Detecting No-Sleep Energy Bugs in Smartphone Apps. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys 2012)*. ACM, 267–280.

[36] Ella Peltonen, Eemil Lagerspetz, Jonatan Hamberg, Abhinav Mehrotra, Mirco Musolesi, Petteri Nurmi, and Sasu Tarkoma. 2018. The Hidden Image of Mobile Apps: Geographic, Demographic, and Cultural Factors in Mobile Usage. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '18)*. ACM, 10:1–10:12.

[37] Ella Peltonen, Eemil Lagerspetz, Petteri Nurmi, and Sasu Tarkoma. 2015. Energy Modeling of System Settings: A Crowdsourced Approach. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom 2015)*. IEEE, 37–45.

[38] Ella Peltonen, Eemil Lagerspetz, Petteri Nurmi, and Sasu Tarkoma. 2016. Constella: Crowdsourced System Setting Recommendations for Mobile Devices. *Pervasive and Mobile Computing* 26 (2016), 71–90.

[39] Ahmad Rahmati, Angela Qian, and Lin Zhong. 2007. Understanding Human-battery Interaction on Mobile Phones. In *Proceedings of the 9th ACM International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI 2007)*. ACM, 265–272.

[40] Lenin Ravindranath, Jitendra Padhye, Sharad Agarwal, Ratul Mahajan, Ian Obermiller, and Shahin Shayandeh. 2012. AppInsight: Mobile App Performance Monitoring in the Wild. In *Proceedings of the 10th Symposium on Operating Systems Design and Implementation (OSDI 2012)*. USENIX, 107–120.

[41] Marie Reilly. 1993. Data Analysis Using Hot Deck Multiple Imputation. *Journal of the Royal Statistical Society: Series D (The Statistician)* 42, 3 (1993), 307–313.

[42] Yvonne Rogers, Kay Connelly, Lenore Tedesco, William Hazlewood, Andrew Kurtz, Robert E. Hall, Josh Hursey, and Tammy Toscos. 2007. Why It's Worth the Hassle: The Value of In-Situ Studies When Designing Ubicomp. In *Proceedings of the 9th International Conference on Ubiquitous Computing (UbiComp 2007)*. Springer, 336–353.

[43] Tom Rolfe. 2017. Low Power Mode: Automatic Energy Saving | IOS 11 Guide. Retrieved February 14, 2019 from http://www.tapsmart.com/tips-and-tricks/guide-understanding-low-power-mode-on-iphone-ios-10/

[44] Mohammad Salehan and Arash Negahban. 2013. Social Networking on Smartphones: When Mobile Phones Become Addictive. *Computers in Human Behavior* 29, 6 (2013), 2632–2639.

[45] Mahadev Satyanarayanan and Dushyanth Narayanan. 2001. Multi-Fidelity Algorithms for Interactive Mobile Applications. *Wireless Networks* 7, 6 (2001), 601–607.

[46] Stephan Sigg, Ella Peltonen, Eemil Lagerspetz, Petteri Nurmi, and Sasu Tarkoma. 2019. Exploiting Usage to Predict Instantaneous App Popularity: Trend Filters and Retention Rates. *ACM Transactions on the WEB* (2019). Accepted for publication.

[47] Sebastian Sonntag, Jukka Manner, and Lennart Schulte. 2013. Netradar-Measuring the Wireless World. In *Proceedings of the 11th IEEE International Symposium on Modeling & Optimization in Mobile, Ad Hoc & Wireless Networks (WiOpt, 2013)*. IEEE, 29–34.

[48] Statista. 2018. Number of Apps Available in Leading App Stores as of 3rd Quarter 2018. Retrieved February 14, 2019 from https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

[49] Po-Han Tseng, Nai-Ching Wang, Ruei-Min Lin, and Kuan-Ta Chen. 2011. On the Battle Between Lag and Online Gamers. In *Proceedings of the IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR 2011)*. IEEE, 1–6.

[50] Steven Warburton. 2009. Second Life in Higher Education: Assessing the Potential for and the Barriers to Deploying Virtual Worlds in Learning and Teaching. *British Journal of Educational Technology* 40, 3 (2009), 414–426.

BotCamp: Bot-driven Interactions in Social Campaigns

Noor Abu-El-Rub
University of New Mexico
nabuelrub@unm.edu

Abdullah Mueen
University of New Mexico
mueen@unm.edu

ABSTRACT

Bots (i.e. automated accounts) involve in social campaigns typically for two obvious reasons: to inorganically sway public opinion and to build social capital exploiting the organic popularity of social campaigns. In the process, bots interact with each other and engage in human activities (e.g. likes, retweets, and following).

In this work, we detect a large number of bots interested in politics. We perform multi-aspect (i.e. temporal, textual, and topographical) clustering of bots, and ensemble the clusters to identify campaigns of bots. We observe similarity among the bots in a campaign in various aspects such as temporal correlation, sentimental alignment, and topical grouping. However, we also discover bots compete in gaining attention from humans and occasionally engage in arguments. We classify such bot interactions in two primary groups: agreeing (i.e. positive) and disagreeing (i.e. negative) interactions and develop an automatic interaction classifier to discover novel interactions among bots participating in social campaigns.

CCS CONCEPTS

• **Information systems → Online advertising**; **Web mining**; *Clustering*; Social networking sites.

KEYWORDS

Bots Interaction; Campaign Detection; Influence; Cluster Ensemble; Graph Mining

ACM Reference Format:
Noor Abu-El-Rub and Abdullah Mueen. 2019. BotCamp: Bot-driven Interactions in Social Campaigns. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313420

1 INTRODUCTION

Social networking sites bring people closer to each other and facilitate fast and convenient information flow. However, modern social media sites suffer from user accounts that work towards fast and automated building of social capital and exploiting the social influence to sway public opinion. Such user accounts (commonly named as bots) perform scheduled posting [26], near-automated registrations [27], and chronological deletions [11] among many other unsocial and non-human behavior.

To multiply the effect, instead of creating super smart bots, botmasters employ a large number of naive bot accounts to attain their objectives. Not surprisingly, humans tend to believe repeatedly encountered information from diverse sources [22]. Thus, a swarm of bots can potentially run successful advertising campaigns to promote products, election campaigns to win races, and organizational campaigns to recruit for ideological groups. To understand the fullest potential of a swarm of bots, in this paper, we perform an empirical study on bot activities in social campaigns and develop a technique to detect and classify bot-driven interactions in social campaigns. Quantifying bot-driven interactions in social campaigns can be useful for political parties, advertising agencies, charitable organizations among many others. Early detection and characterization of bot participation in campaigns will help campaigns flourish organically.



Figure 1: An example of bot interactions. Politically motivated bots are discussing trend manipulation.

An example of bot behavior in Twitter at the time of U.S. Presidential Election in 2016 is given in Figure 1. The user account `@JaredWyand` was an active supporter of Trump campaign. The account has been detected by both DeBot [9] and Botometer [12] systems due to its high frequency of tweets and content similarity. The account is currently suspended by Twitter. The tweet shown here has been retweeted 1.2K times. The two other users copied the tweet shortly after that [6]. These users are also detected as bots by DeBot [1] *and* Botometer[2], however, they are not suspended by Twitter at the time of writing. The content of the tweets shows that bot accounts are promoting a specific topic in Twitter's ranking system by frequently tagging relevant hashtags. The content shows that bot accounts are tracking progress of competing political campaign.

[1] www.cs.unm.edu/~chavoshi/debot/
[2] https://botometer.iuni.iu.edu

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313420

Note that every bot account has a human owner who can post in natural language in between scheduled posts.

The example demonstrates that bot accounts collaborate towards an objective i.e. making a topic trending. It also demonstrates that bots exhibit negative sentiments towards competing campaigns.

In this work, we develop a system to detect bot-driven interaction in campaigns categorized by general topics. For example, we have detected five major campaigns interacting under the "U.S. Election 2016 topic". Three of the bot-driven campaigns are taking sides of the candidates. The objective of the two of the remaining bot-driven campaigns is to gain human attention by adopting popular topics such as U.S. Election. Our system, named **BotCamp**, continuously collects bots for a given topic and detects bots using the DeBot system [9]. BotCamp identifies bots that are posting similar content on the campaign topic, and accumulates such bots over a long time to create graph structures on various aspects such as retweets, mentions, shared media and hashtags. We develop a heuristic cluster ensembling approach to combine communities detected from these graphs, which leads to discovering bot-driven interactions.

In this work, we have collected bot activities related to social campaigns in U.S. election. We analyze the campaigns to understand their information flow and status after campaigns are over. All data and code are made public [2].

The rest of the paper contains a discussion section in related work and background (2), an overview section describing the framework (3), an experimental section showing ensembling and interaction classifier evaluation (4), and the last section concludes the work (5).

Disclaimer: We do not address the question, "who" create and operate bot accounts. We define bots as the accounts that show signs of automation. We collect empirical evidence of "how" bots are involved in social campaigns and reason about "why" bots are involved. To the best of our knowledge, this work is one of the very first to generalize bot interactions on social media.

2 RELATED WORK AND BACKGROUND

2.1 Related Work

Our work combines two independent streams of research on social media: campaign detection and bot detection. Campaign detection works mostly focus on finding campaigns based on *one* specific aspect of messaging, such as message similarity [19][18][23], URL bursts [20], retweet structure [17]. We combine several other messaging aspects such as mentions, hashtags, and media sharing. All of these works detect clusters/communities in some messaging graphs. Such communities may include both bots and humans, hence existing work cannot separate the bot-driven part of the campaign.

DARPA Bot Challenge suggests an estimated 15% of Twitter accounts are bots [25]. Existing bot detection techniques are either supervised [12][14] and unsupervised [9]. Since our goal is to identify campaign specific bots, we opt for an unsupervised technique, DeBot [9].

Bot activities related to campaigns have been studied previously that associated bot activities with political entities [16][3][24]. Our goal is to explore beyond politics to sports, entertainment, marketing, etc., at a much larger scale of thousands of bot accounts.

Bots have been categorized based on their roles as individual users, independent of campaigns they take part in [21]. In contrast, we categorize bots based on their type of interactions in social campaigns.

2.2 Social Campaigns

Definition of campaign has been diverse in the literature, mostly attributed as unethical and illegal cases of social campaign. For example, coordinated campaigns [19], spam campaigns [13], promoted campaigns [15], fraud campaigns [8], and incentivized campaigns [4, 5] are some of the characterizations of campaigns.

In general, we define a social campaign as a group of concepts aligned to an objective that a group of people want to achieve. For example, `#antivax` and `#autism` are concepts supported by people who want to abolish vaccination. Another example is a fund-raising campaign started by Peter Dunn (`@PeterThePlanner`) in Indiana to support homeless people immediately before a blizzard hit Indianapolis. $41K was raised organically from various organizations and individuals for `@WheelerMission`. Therefore, a social campaign should not be perceived as purely inorganic or organic. Instead, considering that both humans and bots are involved together, we propose to quantify the level of bot and human participation in social campaigns. Quantifying organic participation in campaigns can be useful for political parties, advertising agencies, charitable organizations among many others.

2.3 Bot Detection

Automated accounts, a.k.a *bots*, are tweeting/re-tweeting always. Bots are controlled by computer programs. There may exist automated accounts which are not harmful such as `@countforever`, but most bots pretend to be human, entice people to follow them, and/or share ideas. DeBot is a parameter-free unsupervised system [9], that constantly collects data from Twitter and detects bots based on their synchronicity at intervals of 180 minutes. Number of bots DeBot detects in an interval depends on the topic, time of day, bot presence and sampling rate. Note that, Twitter streaming API provides a 1% of sample. In a successful interval, DeBot detects few bot-clusters containing tens of bots.

DeBot is a near real-time system that exploits highly unusual activity correlation across users as an indicator of bot behavior. The authors show that even if millions of active users interact at a time instance, human users are not likely to have more than tens of synchronous postings at random [10]. Although we use DeBot as an integral part of the detection system, we can replace DeBot by any other topic-specific near real-time system.

3 BOT INTERACTIONS IN CAMPAIGNS

3.1 The Framework

Figure 2 shows the BotCamp framework. There are three components of the system: Keyword Generator, Campaign Detector and Interaction Detector. We describe each of the components below.

Keyword Generator: BotCamp continuously collects trending hashtags to maintain related keywords to a seed set of keywords. The motivation behind such a keyword generator is to adapt with

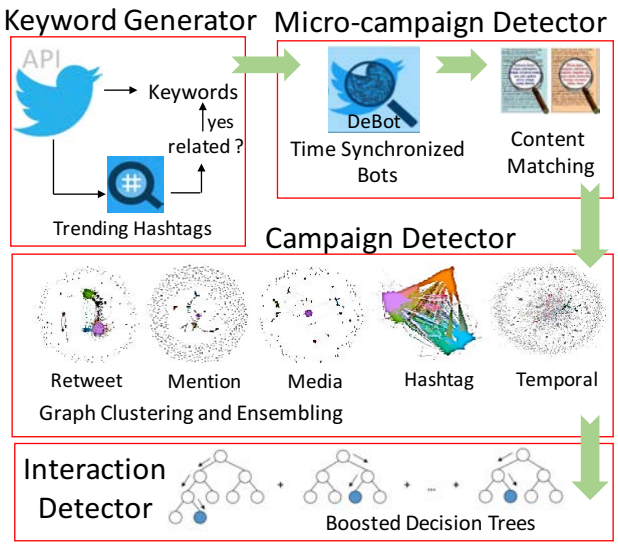


Figure 2: BotCamp framework

changing campaign dynamics. Trending keywords related to a campaign can be changed frequently. For example, to monitor the U.S. election, we started with a seed of twenty keywords including general topics such as `election`, `Trump`, `Clinton`, etc. After the U.S. election, the seed set grew to 231 keywords including `MAGA` ("short form of Make America Great Again"), `PodestaEmails`, and `CrookedHillary`. We collect the top 50 trends in twitter in every three-hour interval. If more than 50% of the tweets containing a trend also contains a seed keyword, we add the trend to seed set. In short campaigns, the seed set remains almost identical for the lack of dynamics in the campaigns. In long campaigns (i.e. election campaigns), keywords can be weighted based on their recency. The keywords can be both in support or in favor of parties involved in a campaign. We have labeled the sentiment associated with the keywords manually for the U.S. dataset.

Campaign Detector: We use the keywords in an instance of DeBot system that detects synchronized bots within an interval of three hours. We use the recommended threshold of 0.99 correlation to detect bots. DeBot outputs clusters of bots that we further analyze to detect clusters of bots that are both temporally synchronous and textually similar. BotCamp accumulates bots for a duration that is sufficient for the campaign to reach a stable state. We have accumulated at least one week of bots for all of our experiments. After bots are collected, we produce five graphs capturing various aspects of campaigns (e.g. retweet graph, hashtag graph, etc.). BotCamp detects communities in these graphs based on modularity optimization algorithm [7]. We develop a cluster ensembling technique that combines the communities from different aspects into consensus communities representing campaigns.

Interaction Detector: BotCamp consists of a classifier that categorizes the interaction between pairs of campaigns. The classifier is trained on a manually labeled set of interactions. We consider two types of interactions: *agreeing* and *disagreeing* interactions. We produce a set of 94 novel features that are indicative to various interaction types. The classifier is AdaBoost ensemble classifier, we use the classifier to categorize all possible pairs of interacting campaigns, and quantify bot participation in a campaign. Figure 3 shows examples of such interactions.

In the next two sections, we elaborate on the the campaign and interaction detectors.

3.2 Campaign Detector

Our campaign detection system is a two step process: Content matching and Graph clustering.

3.2.1 **Content Matching.** DeBot produces a set of unusually synchronous bots. Although a group of unlikely synchronous bots typically works towards a campaign, there can be spurious synchronous groups that are just naively periodic. In this step, we detect bots that are posting not only at close time instances, but also similar content. We consider each synchronous cluster detected by DeBot, and calculate the text and hashtag similarity among the bots in the cluster. *Text similarity* between two users u and v is defined by the Jaccard similarity of their set of unigrams. More precisely, if $G(u)$ is the set of unigrams extracted from the tweets made by u, excluding the stop words, the text similarity between u and v is:

$$SimText(u,v) = \frac{G(u) \cap G(v)}{G(u) \cup G(v)}$$

The similarity within a cluster C is

$$SimText(c) = \frac{\sum_{\forall u,v \in C} SimText(u,v)}{|C| * (|C|-1)/2}$$

Hashtag similarity between two users u and v is defined by the Jaccard similarity of their set of hashtags. More precisely, if $H(u)$ is the set of unigrams extracted from the hashtags made by u, the hashtags similarity between u and v is:

$$SimHashtag(u,v) = \frac{H(u) \cap H(v)}{H(u) \cup H(v)}$$

The hashtag similarity within a cluster C is

$$SimHashtag(c) = \frac{\sum_{\forall u,v \in C} SimHashtag(u,v)}{|C| * (|C|-1)/2}$$

We define a micro-campaign as a cluster of temporally synchronous bots, C, having either $SimText(C) \geq 0.5$ or $SimHashtags(C) \geq 0.5$. Note that such micro-campaigns are formed based on three hours of information.

3.2.2 **Graph Construction.** Once BotCamp accumulates micro-campaigns detected in three hour batches for over the duration of the campaign, the system creates five graphs namely: *retweet*, *media*, *hashtag*, *mention* and *temporal* graphs. The objective is to study the underlying interaction among micro-campaigns on various aspects over the duration of the campaign. Since the graphs are based on three hour long captures, the graphs are crude approximations of the graphs that we could produce if we had all data available. We describe each of these graphs below.

Retweet Graph

Retweets usually mean endorsement. Hence, we create a undirected retweet graph where nodes are bots, and we add an edge between two bots when they retweet from each other at least once, encoding



Figure 3: (left) Agreement interaction among bots. (middle and right) Disagreement interactions.

their mutual endorsements. In contrast, we can create a directed retweet graph by adding edges from the retweeting node to the original author node.

Mention Graph

In public conversations, bots mention (i.e. adding @ before an account name) other accounts in tweets. Mentions are typically used to draw attention of the person being mentioned. Thus, mentions are useful to express agreement, disagreement, endorsement, promotion, etc. We create a mention graph by adding an edge between two bots if they mentioned each other.

Media Graph

Bots in the same campaign proliferate the same information. Memes, photos, and videos are typically more expensive to create compared to tweets, however, such media are more attractive. Determined campaigns spend resources to create media and employ automated accounts to share the media. We create the media graph on bots by connecting two bots that share the exact same URL media.

Hashtag Graph

Hashtag is a powerful way to organize content for better searching. Information seekers often use hashtags to learn discussion items about a topic. Competing campaigns fight for strong position on common discussion topics (e.g. `#Oscars`). Campaigns also want to make hashtags trending (See Figure 1). Thus, tagging the same hashtag may mean either agreement or disagreement; at weaker level than mentions. If two bots have more than 50% of their hashtags common (i.e. $SimHashtags(u, v) = \frac{H(u) \cap H(v)}{H(u) \cup H(v)} \geq 0.5$), we add an edge between them to create the hashtag graph.

Temporal Correlation Graph

Synchronous bot activities indicate that bots using the same scheduler (e.g. a random posting interval generator or a human leader). We add an edge between two bots if they have been correlated at least once in their campaign lifetime for three hours interval regardless of their content similarity. Since we are using Debot we know that all bots are temporally correlated at least once with other bots, however, this graph exposes further correlations that could happen along the campaign life duration.

3.2.3 **Graph Clustering and Ensembling.** We consider building larger campaigns from the micro-campaigns by clustering the individual graphs mentioned in the previous section and ensembling the clusters across various aspects.

To cluster the graphs, we use a state-of-the-art technique called Louvain Modularity to cluster bots [7]. The algorithm uses greedy modularity optimization method and has linear complexity, thus it run fast on large dataset. We run the algorithm on the five graphs respectively. For each graph, we produce clusters of bots, therefore, each bot will belong to five clusters of various aspects.

Ensembling clusters from the five graphs enable detection of interesting patterns that independent aspect alone cannot reveal. We propose an ensembling method to detect campaign. First, we define a dissimilarity matrix A between bots participating in a campaign, where we compute the pairwise distance between two bots as:

$$A_{i,j} = 1 - \frac{\|Community(i) \cap Community(j)\|}{\|Community(i))\|}$$

Where $Community(i)$ refers to the set of communities that user i belongs to. The resulting distance matrix A contains normalized values range between 0 to 1. Where 0 indicates that the two bots appeared in the same cluster in all graphs, and 1 means the two bots did not appear in any common cluster. Then, we use average linkage hierarchical clustering to cluster bots and choose a restrictive threshold to stop unnecessary cluster merging. The merging starts with the most similar bots and stop when threshold is *0.8*. The selected threshold is chosen because it ensemble bots in one community if all bots share one common community on average with all other bots within the campaign. For verification, we conducted a small experiment on a sample of labeled bots in the U.S. Election, where labels are either Trump or Clinton supporter, we used different threshold and reported the Normalized Mutual Information (NMI) with the labeled data. the largest NMI was reported at 0.8.

3.3 Interaction Detector

Once we find a set of campaigns, we are interested to study the interactions among them. The simplest starting point is to consider pair-wise interactions. We consider developing a machine learned

classifier to automatically classify interactions in agreeing and disagreeing categories.

We label interactions between a pair of campaigns by manually checking the tweets, replies and retweets where bots from both campaigns participated. Such interactions can be largely categorized in two classes: agreeing and disagreeing. The example in the Introduction can be treated as a disagreeing interaction between the Trump and Clinton supporting bots. One may consider creating a full scale of classes between -3 and 3, 0 being the neutral class, instead of a two-class problem. However, the cost associated with labeling hundreds of pairs of campaigns is significant. In contrast, any neutral interaction can also be thought of as weak agreement, and thus, a two-class formulation is chosen. We manually labeled 80 campaign interactions where 57.5% are disagreement interaction and 42.5% are agreement.

3.3.1 **Feature Generation and Selection.** We start with a set of 94 features. The features are from four categories: time-based, sentiment-based, user-based features and network-based features. We describe a subset of features from each category below.

(1) *Time-based Features:* Temporal features help revealing bots that are collaboratively working toward the same objective or operated by the same software. Features such as the number of temporally correlated bots can be useful to understand the relation between a pair of campaign. Similarly, average interval time between mentions and number of bots involved in conversational interaction can indicate the interaction type. Usually, conversations with small intervals between mentions can be an indication for argument and disagreement.
(2) *Sentiment-based Features:* While retweet interaction almost always indicates agreement, mention interaction is controversial in nature. Bots and cyborgs could engage in arguments to support or attack a certain topic. To understand the nature of these conversations, we investigate entities sentiment within each conversation using IBM Watson Natural Language Understanding API [1]. For each conversation, we create various features describing the number of sentiment disagreement, difference of average sentiment over all entities to understand bots opinion polarity towards topics.
(3) *Content-based Features:* Usually, campaigns that share common objective tend to have more agreement than disagreement towards specific topics, and vice versa. We create features that characterize the relationship between two interacting campaigns. Examples include number of common topics, hashtags and media between two campaigns.
(4) *Network-based Features:* In addition to these three categories, we summarize bots and campaign connectivity using features that describe network topology such as ratio of friends to followers for a user.

Although indirect interactions are possible, we consider only direct interactions in the forms of retweets and mentions between campaigns. We obtain 94 features from four categories. We perform feature selection to identify the best features based on their importance weights in a decision tree model using Gini importance. After feature selection, the set is reduced to 15 features. Most informative features are content-based, temporal-based and sentiment-based features. The complete list of features is available in the supporting webpage [2].

3.3.2 **Training the Classifier.** We employ an AdaBoost model trained on decision tree classifier (CART) with ten weak learners. Information Gain is used to measure the quality of splits, then predictions from different learners are combined using weighted majority vote to produce the final prediction. Our choice for Adaboost is a result of lack of labeled data. Quantifying the sentiment of an interaction needs significant effort because of short length of tweets (The character limit for tweets is 280) and many alternative usages (emoji, abbreviation, etc.). Boosting the decision tree helps tackle these challenges.

4 EXPERIMENTAL EVALUATION

4.1 BotCamp by Numbers

We describe the BotCamp framework in numbers for the U.S. Election campaigns. First, we start with 20 seed keywords, the keyword generator component expands the set to 231 keywords in 60 days. Using the campaign detector, we collect a set of 75 million tweets from 6 million users talking about the election. The number of bots detected is 120K. We exclude clusters that are not matching in content and identify 29K bots from different micro-campaigns. We construct five graphs: retweet (7162 bots with 30811 edges), mention (1137 bots with 785 edges), hashtags (4122 bots with 731687 edges), media (954 bots with 10385 edges) and temporal (29840 with 73623). Graph clustering and ensembling are performed to obtain clusters of 29K bots and ensemble them into 231 campaigns. From the interaction classifier, we identify 87 disagreement interactions and 2700 agreement interactions between bot campaigns.

The above set of numbers are reproducible using the dataset provided in our supporting webpage [2]. However, U.S. Election 2016 has already happened, which limits comparison to alternative methods. To facilitate experimental comparison, we made our code public in the supporting webpage, it only requires a set of keywords to run for days to weeks, and produce interacting campaigns.

4.2 Evaluation of Interaction Classifier

We evaluate the boosted decision tree classifier using a 10-fold cross-validation technique. The average and standard deviation of classification performance is described in the Table 2. The results strongly suggest that the feature set can capture the manually labeled training data. The low standard deviation suggests consistency across random samples.

We consider applying the classifier to the unlabeled pairs of campaigns that have some form of interactions (i.e. retweet, mentions, etc.) between them. The results are shown in the Table 3

We have identified 87 disagreeing pairs of campaigns during U.S. election that include disagreement over debate results, email controversy, etc. The result suggests that while some campaign domains are non-competitive in nature, others are controversial leading to disagreement interactions.

4.3 Example Campaigns

This project has identified several small to large campaigns with bot participation in Twitter. Are they meaningful campaigns? - is

Table 1: Example campaigns in the U.S. Election. The bot accounts may be suspended currently.

Campaign	Bots Examples	Number of Bots
Trump Supporters	RWB4Trump, usfortrump Vegans4Trump, TXChiks4Trump MyVoteIs4Trump, Hyperslw4Trump	480
Clinton Supporters	Hillary2016MN IL4Hillary, voteforourlives Liberalibrarian, Hope012015	304
Sanders Supporters	1Birdie4Sanders BernieEvents, BernItUpTV i_AM_theChange, drJimWas4Bernie	100

Table 2: Model Performance

	Accuracy	Precision	Recall
Average	87.5%	98%	83.6%
Variance	0.025	0.003	0.049

Table 3: Interaction Summary

Number of Interactions	Interactions Agreement	Interactions Disagreement
2,790	6.88%	3.11%

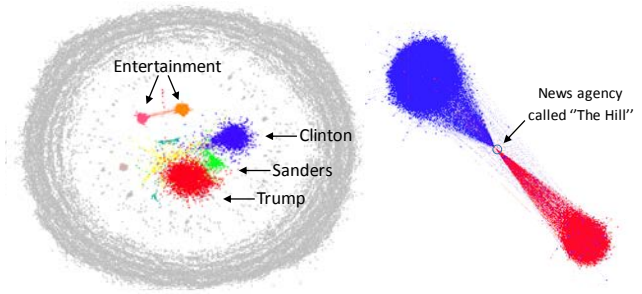


Figure 4: (left) Detected campaigns are shown on an undirected retweet graph (red for Trump supporters, blue for Clinton supporters, green for Sanders supporters). (right) Campaigns found by considering a directed retweet graph. Node in the middle is a news agency called The Hill. Colors indicate strong sentiment polarity towards different candidates based on hashtags.

the natural follow up question. We have investigated the campaigns manually to identify their objectives. Tagging all accounts in all campaigns is labor intensive. We take a 10% random sample of bot accounts to identify the objective, by navigating through their profile and rendering the last 15 tweets. In this section, we first show examples of a few campaigns (see Table 1). We name the campaigns based on the objectives we identified.

(1) **Trump Supporters**: In this campaign, all bots supported candidate Trump in U.S. presidential election in 2016. Their names, profile pictures and tweets show that they mostly care about politics. The bot accounts show strong retweet interactions among them.
(2) **Clinton Supporters**: All bots in this campaign are supporting Clinton. All their tweets are mostly political tweets. Number of Clinton supporting bots is less than that of Trump supporting bots.
(3) **Sanders Supporters**: All bots in this campaign supported candidate Sanders in the U.S. election 2016.

To provide a comprehensive picture, we show all of the campaigns detected by BotCamp in U.S. election dataset on an undirected retweet graph in the Figure 4. In addition to the supporters of three prominent candidates, several other small campaigns exist. The two loosely connected campaigns that we labeled as *Entertainment* campaigns are consisted of bots interested in variety of topics including politics, but mostly celebrity news. We explain the weak communication between the campaigns as an artifact of partially complete dataset. Note that Twitter API provides 1% of tweets.

The Figure 4(left) shows that politically motivated bots rarely retweet mutually across parties. This is not surprising, however, the directed retweet graph in the Figure 4(right) shows that the campaigns retweet from a common news source named *The Hill*.

5 CONCLUSION

Online social media is tremendously important for the future of democratic governance. Automated activities on social media create opportunities for manipulation, misinformation and distrust. This paper demonstrates that social campaigns can be corrupted by inorganic interactions among bots and develops a technique to classify inorganic interactions among and within campaigns. We show empirical evidence of various interactions among campaigns. However, this work is merely one step towards better monitored social media, significant effort must be made to protect human users from inorganic interruptions.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1527127.

REFERENCES

[1] 2016. Natural Language Understanding. https://www.ibm.com/watson/services/natural-language-understanding

[2] 2019. Supporting web page containing data, code, results for BotCamp. http://cs.unm.edu/~nabuelrub/BotCamp.

[3] Norah Abokhodair, Daisy Yoo, and David W. McDonald. 2015. Dissecting a Social Botnet: Growth, Content and Influence in Twitter. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing - CSCW '15*. ACM Press, New York, New York, USA, 839–851. https://doi.org/10.1145/2675133.2675208

[4] Noor Abu-El-Rub, Amanda Minnich, and Abdullah Mueen. 2017. Anomalous Reviews Owing to Referral Incentive. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*. ACM, 313–316.

[5] Noor Abu-El-Rub, Amanda Minnich, and Abdullah Mueen. 2017. Impact of referral incentives on mobile app reviews. In *International Conference on Web Engineering*. Springer, 351–359.

[6] ADORABLEDEPLORABLE. 2016. ChristiChat: RT JaredWyand: Twitter pushing ... https://twitter.com/GoldStarMomTX55/status/795287405804879872

[7] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[8] Zhu Sun Chang Xu Jie Zhang. 2017. Online Reputation Fraud Campaign Detection in User Ratings. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, {IJCAI-17}*. 3873–3879. https://doi.org/10.24963/ijcai.2017/541

[9] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. 2016. DeBot: Twitter Bot Detection via Warped Correlation.. In *ICDM*. 817–822.

[10] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. 2016. Identifying correlated bots in twitter. In *International Conference on Social Informatics*. Springer, 14–21.

[11] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. 2017. On-Demand Bot Detection and Archival System. *Proceedings of the 26th International Conference on World Wide Web Companion* (2017), 183–187. https://doi.org/10.1145/3041021.3054733

[12] Clayton Allen Davis, Onur Varol, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2016. Botornot: A system to evaluate social bots. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 273–274.

[13] Son Dinh, Taher Azeb, Francis Fortin, Djedjiga Mouheb, and Mourad Debbabi. 2015. Spam campaign detection, analysis, and investigation. *Digital Investigation* 12, Supplement 1 (2015), S12 – S21. https://doi.org/10.1016/j.diin.2015.01.006

[14] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2014. The Rise of Social Bots. *CoRR* abs/1407.5 (2014). http://arxiv.org/abs/1407.5225

[15] Emilio Ferrara, Onur Varol, Filippo Menczer, and Alessandro Flammini. 2016. Detection of Promoted Social Media Campaigns.

[16] Michelle C Forelle, Philip N. Howard, Andres Monroy-Hernandez, and Saiph Savage. 2015. Political Bots and the Manipulation of Public Opinion in Venezuela. *SSRN Electronic Journal* (2015). https://doi.org/10.2139/ssrn.2635800

[17] Kiran Garimella, Ingmar Weber, and Munmun De Choudhury. 2016. Quote RTs on Twitter: usage of the new feature for political discourse. In *Proceedings of the 8th ACM Conference on Web Science*. ACM, 200–204.

[18] Kyumin Lee, James Caverlee, Zhiyuan Cheng, and Daniel Z. Sui. 2011. Content-driven detection of campaigns in social media. In *Proceedings of the 20th ACM international conference on Information and knowledge management - CIKM '11*. ACM Press, New York, New York, USA, 551. https://doi.org/10.1145/2063576.2063658

[19] Kyumin Lee, James Caverlee, Zhiyuan Cheng, and Daniel Z. Sui. 2013. Campaign extraction from social media. *ACM Transactions on Intelligent Systems and Technology* 5, 1 (12 2013), 1–28. https://doi.org/10.1145/2542182.2542191

[20] Huayi Li, A Mukherjee, Bing Liu, R Kornfield, and S Emery. 2014. Detecting Campaign Promoters on Twitter Using Markov Random Fields. In *Data Mining (ICDM), 2014 IEEE International Conference on*. 290–299.

[21] Richard J. Oentaryo, Arinto Murdopo, Philips K. Prasetyo, and Ee-Peng Lim. 2016. On Profiling Bots in Social Media. Springer, Cham, 92–109. https://doi.org/10.1007/978-3-319-47880-7{_}6

[22] Gordon Pennycook, Tyrone D Cannon, and David G Rand. 2017. Prior exposure increases perceived accuracy of fake news. (2017).

[23] Eduardo J Ruiz, Vagelis Hristidis, Carlos Castillo, and Aristides Gionis. 2013. Measuring and Summarizing Movement in Microblog Postings.. In *ICWSM*.

[24] Pablo Suárez-Serrato, Margaret E Roberts, Clayton Davis, and Filippo Menczer. 2016. On the influence of social bots in online protests. In *International Conference on Social Informatics*. Springer, 269–278.

[25] V. S. Subrahmanian, Amos Azaria, Skylar Durst, Vadim Kagan, Aram Galstyan, Kristina Lerman, Linhong Zhu, Emilio Ferrara, Alessandro Flammini, Filippo Menczer, Rand Waltzman, Andrew Stevens, Alexander Dekhtyar, Shuyang Gao, Tad Hogg, Farshad Kooti, Yan Liu, Onur Varol, Prashant Shiralkar, Vinod Vydiswaran, Qiaozhu Mei, and Tim Huang. 2016. The DARPA Twitter Bot Challenge. (Jan. 2016). arXiv:1601.05140 http://arxiv.org/abs/1601.05140

[26] A Nicole Sump-Crethar. 2012. Making the most of Twitter. *The reference librarian* 53, 4 (2012), 349–354.

[27] Kurt Thomas, Vern Paxson, Damon Mccoy, and Chris Grier. 2013. Trafficking Fraudulent Accounts : The Role of the Underground Market in Twitter Spam and Abuse Trafficking Fraudulent Accounts :. In *USENIX Security Symposium (SEC'13)*. 195–210.

City-Wide Signal Strength Maps: Prediction with Random Forests

Emmanouil Alimpertis
UC Irvine
ealimper@uci.edu

Athina Markopoulou
UC Irvine
athina@uci.edu

Carter T. Butts
UC Irvine
buttsc@uci.edu

Konstantinos Psounis
USC
kpsounis@usc.edu

ABSTRACT

Signal strength maps are of great importance to cellular providers for network planning and operation, however they are expensive to obtain and possibly limited or inaccurate in some locations. In this paper, we develop a prediction framework based on random forests to improve signal strength maps from limited measurements. First, we propose a random forests (RFs)-based predictor, with a rich set of features including location as well as time, cell ID, device hardware and other features. We show that our RFs-based predictor can significantly improve the tradeoff between prediction error and number of measurements needed compared to state-of-the-art data-driven predictors, *i.e.,* requiring 80% less measurements for the same prediction accuracy, or reduces the relative error by 17% for the same number of measurements. Second, we leverage two types of real-world LTE RSRP datasets to evaluate into the performance of different prediction methods: (i) a small but dense Campus dataset, collected on a university campus and (ii) several large but sparser NYC and LA datasets, provided by a mobile data analytics company.

CCS CONCEPTS

• **Networks** → **Network measurement**; **Network performance analysis**.

KEYWORDS

Signal Strength Maps; LTE; RSRP; RSS; Random Forests; Prediction

ACM Reference Format:
Emmanouil Alimpertis, Athina Markopoulou, Carter T. Butts, and Konstantinos Psounis. 2019. City-Wide Signal Strength Maps: Prediction with Random Forests. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313726

1 INTRODUCTION

Cellular providers rely on key performance indicators (*a.k.a.* KPIs) to understand the performance and coverage of their network, as well as that of their competitors. KPIs usually include wireless channel measurements (the most important of which for LTE is arguably the reference signal received power, *a.k.a.* RSRP) as well as other performance metrics [17] (*e.g.,* throughput, delay, frequency band, location, time) associated with the measurement. Signal maps consist of a large number of measurements of KPIs and are of crucial importance to cellular operators, for network management, maintenance, upgrades, operations and troubleshooting [19].

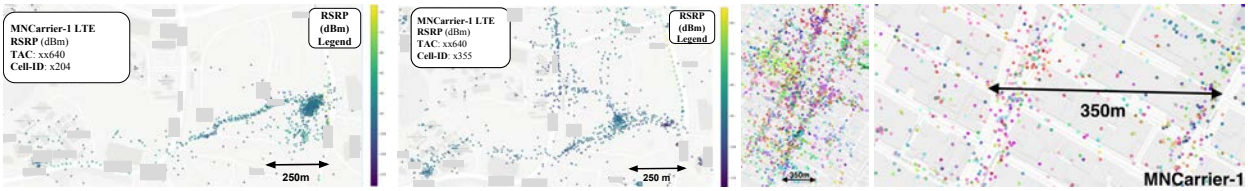
Although cellular providers can collect measurements on the network edge themselves (*e.g.,* via wardriving [31]), they increasingly choose to outsource the data collection to third parties for a variety of reasons, including: cost, liability related to privacy concerns of collecting data on end-user devices, and lack of access to competitor networks. Mobile analytics companies (*e.g.,* OpenSignal [22], RootMetrics [28], Tutela [30] *etc.*) crowdsource measurements directly from end-user devices, via standalone mobile apps [22], or measurement SDKs [30] integrated into partnering apps. Thus large scale signal maps collection is achieved, but the measurements can be sparse in space (depending on end-user location) and time (measurements are collected infrequently so as to not drain user resources, such as battery or cellular data). Either way, signal strength maps are expensive for both carriers (paying millions dollars to third parties) and crowdsourcing companies (most of which use cloud services, thus collecting more measurements increases their operational cost). Moreover, trends, such as (i) 5G dense deployment of small cells and (ii) smart city and IoT deployments, will only increase the need for accurate performance measurements [4, 9, 14, 16], while data may be sparse, unavailable, or expensive to obtain.

Our goal in this paper is to improve the tradeoff between cost (number of measurements) and quality (*i.e.,* coverage and accuracy) of signal maps via signal strength prediction from limited measurements. In general, there are two approaches in RSRP prediction: propagation models and data-driven approaches. Our approach falls in the second category and we employ a powerful machine learning framework that naturally incorporates multiple features. More specifically, we make the following contributions.

1. RFs RSRP prediction framework. We develop a powerful machine learning framework based on random-forests (RFs), considering a rich set of features including, but not limited to, location, time, cell ID, device hardware, distance from the tower, frequency band, and outdoors/indoors location of the receiver, which all of them affect the wireless properties. To the best of our knowledge, this is the first time that location, time, device and network information are considered jointly for the problem of signal strength prediction. We assess the feature importance and we find cell ID, location, time and device type to be the most important. Moreover, this is the first time that RFs have been applied to the signal maps estimation problem. Prior work on data-driven prediction for signal maps was primarily based on geospatial interpolation

Acknowledgments: This work was supported by NSF Awards 1649372, 1526736, 1444060; as well as by Networked Systems and CPCC at UC Irvine and the Broadcom Foundation Fellowship program.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313726



(a) Campus example cell x204: high density (0.66), low dispersion (325). (b) Campus: example cell x355: small density (0.12) more dispersed data (573). (c) NYC: Manhattan LTE TA (d) NYC: zooming in Manhattan Midtown (Time Square) for some of the available cells.

Figure 1: LTE RSRP Map Examples from our datasets. (a)-(b): Campus dataset**. Color indicates RSRP value. (c)-(d):** NYC dataset**. Data for a group of LTE cells in the Manhattan Midtown area. Different colors indicate different cell IDs.**

techniques [8, 20, 24], which do not naturally extend beyond location features. We show that our RFs-based predictors can significantly improve the tradeoff between prediction error and number of measurements needed, compared to state-of-the-art data-driven predictors. They can achieve the lowest error of these baselines with 80% less measurements; or they can reduce the *RMSE* (root mean square error) by 17% for the same number of measurements.

2. Real-world datasets. Our study leverages two types of real-world datasets: (i) a small but dense Campus dataset collected on a university campus; and (ii) several large but sparser NYC and LA datasets, provided by a mobile data analytics company. Examples are depicted in Fig. 1 and information about the datasets is provided in Table 2. We use these datasets to evaluate and contrast different methods and gain insights into tuning our framework. For example, cell ID is an important feature in areas with high cell density, which is encountered in urban areas such as Manhattan Midtown; in contrast, cell ID should be used to train cell-specific RFs in suburban areas. Furthermore, time features are important in cells with less dispersed measurements, *i.e.,* concentrated in fewer locations. To the best of our knowledge, the NYC and LA datasets are the largest used to date for RSRP (or other signal strength) prediction, in terms of any metric (number of measurements, geographical scale, number of cells *etc.*). They contain 10.9 million LTE data points in areas of $300km^2$ and $1600km^2$ for NYC and LA respectively, instead of at most tens of km^2 and tens of thousands of measurements in [13] or smaller scale in [8, 15, 20, 24].

The structure of the rest of the paper is as follows. Section 2 presents our random forests-based prediction approach as well as baselines and prior work for comparisons. Section 3 presents the available signal map datasets. Section 4 provides evaluation results. Section 5 concludes the paper.

2 RSRP PREDICTION

2.1 Problem Statement

RSRP Definition. Although there are many KPIs related to received signal strength (RSS), including RSRP, RSRQ (reference signal received quality), RSSI (RSS Indicator), in this paper we focus specifically on reference signal received power (RSRP), defined by 3GPP in [12]. This choice is both because RSRP is widely used for various operations in LTE (*e.g.,* cell selection, handover decisions [10], network quality assessment *etc.*) and as a case study that can potentially be applied to prediction of other RSS metrics. Typically, RSRP is reported in dBm by UEs (user equipment) as the average power over several, narrow-band, control channels.

LTE Cells vs. Tracking Areas. A serving LTE cell is uniquely identified by the CGI (cell global identifier) which is the concatenation of the following identifiers: the MCC (mobile country code), MNC (mobile network code), TAC (tracking area code) and the cell ID. We abbreviate and refer to CGI as cell ID or *cID*. LTE also defines Tracking Areas (which we will refer to as LTE TA) by the concatenation of MCC, MNC and TAC, to describe a group of neighboring cells, under common LTE management for a specific area.

The RSRP Prediction Problem. Our goal is to predict the RSRP value at a given location, time, and potentially considering additional contextual information (Section 2.3), based on available measurements either in the same cell *cID* or in the same LTE TA. Table 1 summarizes our prediction methods and related work.

2.2 Model-Based Prediction: LDPL

As a representative baseline from the family of model-based predictors, we consider the Log Distance Path Loss (LDPL) propagation model [25], which is simple yet widely adopted in the literature (*e.g.,* [2], [8]). LDPL predicts the power (in dBm) at location $\vec{l}_j$ at distance $||\vec{l}_{\text{BS}} - \vec{l}_j||_2$ from the transmitting basestation (BS *a.k.a.* cell tower), as a log-normal random variable (*i.e.,* normal in dBm) [2]:

$$P_{cID}^{(t)}\left(\vec{l}_j\right) = P_0^{(t)} - 10n_j \log_{10}\left(||\vec{l}_{\text{BS}} - \vec{l}_j||_2/d_0\right) + w_j^{(t)}. \quad (1)$$

[1]We consider two cases regarding path loss exponent (PLE) n_j.

Homogeneous LDPL: Much of the literature assumes that the PLE n_j is the same across all locations. We can estimate it from Eq. (1) from the training data points.

Heterogeneous LDPL: Recent work (*e.g.,* [2, 8]) considers different PLE across locations. We considered several ways to partition the area into regions with different PLEs, and we present *knn* regression, where we estimate $\widehat{n_j}$ from its k nearest neighbors, weighted according to their Euclidean distance, which we refer to as "LDPL-knn".

2.3 Proposed Data-Driven Prediction: RFs

We apply Random Forests (RFs) regression. RFs are an ensemble of multiple decision trees [5], which provides a good trade-off between

[1] $P_0^{(t)}$ is the received power at reference distance d_0 (typically $1m$), calculated by the free-space path loss (Friis) transmission equation for the corresponding downlink frequency, gain and antenna directionality, and $\vec{l}_{\text{BS}}$ the location of the transmitting antenna. n_j, *i.e.,* PLE, has typical values between 2 and 6. The log-normal shadowing is modeled by $w_j^{(t)} \sim \mathcal{N}(0, \sigma_j^2(t))$ (in dB), with variance $\sigma_j^2(t)$ assumed independent across different locations. The cell, cID, affects several parameters in Eq. (1), including P_0, w_j, the locations of transmitting ($\vec{l}_{\text{BS}}$) and receiving ($\vec{l}_j$) antennas.

Table 1: Overview of RSRP Prediction Methodologies evaluated in this paper. Methods proposed in this paper are marked in bold. Methods in regular font are prior art, evaluated as baselines for comparison. Methods in light gray font are reviewed but not implemented in this paper.

(1) Model Based (Radio Frequency Propagation Model)		1(a) $LDPL$ Eq. (1) (Log Distance Path Loss)	1-(b) **LDPL – knn** (heterogeneous PLE)	1(c) WINNER I/II [6], COST 231 [27], Ray Tracing [32], Hata Model [29] *etc.*	Limitations: Requires environment's info *e.g.,* topology, street width, BSs height, 3D map.
Data Driven	(2) Geospatial Interpolation[8]	2(a) OK: Ordinary Kriging	2-(b) OKD:OK Detrending (hybrid of model and data)	2-(c) OKP: OK partitioning (spatial heterogeneous)	(4) Other Data-Driven: 4(a) Bayesian Compressive Sensing [15]. Limitation: no arbitrary features. 4(b) Deep Neural Nets [11] Limitation: Needs 3D Map/LiDAR Data.
	(3) Random Forests (RFs)	3(a) $RFs_{x,y}$ [26] Spatial Features: $\mathbf{x} = (l^x, l^y)$	**3-(b)** $RFs_{x,y,t}$ **Spatiotemporal:** $\mathbf{x} = (l^x, l^y, d, h)$	**3(c)** RFs_{all} **Full Feats: (and in some scenarios:** cid**)** $\mathbf{x} = (l^x, l^y, d, h, dev, \|\|\vec{l}_{BS} - \vec{l}_j\|\|_2, freq_{dl}, out)$	

bias and variance by exploiting the idea of bagging [5]. An RSRP value P can be modeled as follows given a set of features vector $\mathbf{x}$.

$$P|\mathbf{x} \sim \mathcal{N}(RFs_{\mu}(\mathbf{x}), \sigma^2_{\mathbf{x}}) \quad (2)$$

where $RFs_{\mu}(\mathbf{x})$, $RFs_{\sigma}(\mathbf{x})$ are the mean and standard deviation respectively of the RSRP predictor, $\sigma^2_{\mathbf{x}} = RFs_{\sigma}(\mathbf{x}) + \sigma^2_{RFs}$ and σ^2_{RFs} is the MSE of the predictor. The prediction $\widehat{P} = RFs_{\mu}(\mathbf{x})$ is the MLE (maximum likelihood estimation) value minimizing the MSE.

For each measurement j in our data, we consider the following full set of features, available via the Android API:

$$\mathbf{x}^{\mathbf{full}}_{\mathbf{j}} = (l^x_j, l^y_j, d, h, cID, dev, out, ||\vec{l}_{BS} - \vec{l}_j||_2, freq_{dl})$$

• *Location* $\vec{l}_j = (l^x_j, l^y_j)$. These are the spatial coordinates and the only ones considered by previous work on data-driven RSS prediction [8, 20] or in the context of localization [19, 26].

• *Time* features $\mathbf{t_j} = (d, h)$, where d denotes the weekday and h the hour of the day that the measurement was collected. Using h as a feature implies stationarity in hour-timescales, which is reasonable for signal strength statistics.

• *The cell ID, cID*. This is a natural feature since RSRP is defined per serving cell, *i.e.,* the CGI defined in Section 2.1.

• *Device hardware* type, dev. This refers to the device model (*e.g.,* Galaxy9 or iPhone X) and *not* to device identifiers. This feature captures (i) the different noise figures (NF), *i.e.,* electronic interference, and reception characteristics across different devices and (ii) the RSRP calculation may differ across devices and manufacturers [12].

• *The downlink carrier frequency*, $freq_{dl}$. Radio propagation and signal attenuation heavily depend on frequency calculated by *EARFCN* (E-UTRA Absolute Radio Frequency Channel Number).

• $out \in \{0, 1\}$ is an approximate indicator of outdoors or indoors location, inferred from Android's GPS velocity sensor.

• *Euclidean distance* $||\vec{l}_{BS} - \vec{l}_j||_2$, of the receiver at location $\vec{l}_j$ from the transmitting antenna BS (base station or cell tower).

Among the above features, the cell ID cID is particularly important, as it will be shown in Section 4.2.1. When there is a large number of measurements with the same cID, it is advantageous to train a separate RFs model per cID, using the remaining features:

$$\mathbf{x}^{\mathbf{-cID}}_{\mathbf{j}} = (l^x_j, l^y_j, d, h, dev, out, ||\vec{l}_{BS} - \vec{l}_j||_2, freq_{dl}).$$

When there are a few measurements per cID, then we can treat cID as one of the features in $\mathbf{x}^{\mathbf{full}}_{\mathbf{j}}$. We denote as $\text{RFs}_{x,y}$, $\text{RFs}_{x,y,t}$, RFs_{all} the RFs predictors with only spatial (l^x, l^y), spatial (l^x, l^y) and temporal (d, h), and all features, respectively. In Section 4, we assess feature importance in different datasets.

Why RFs for Data-Driven Prediction? First, RFs can naturally incorporate all aforementioned features, since geospatial interpolation [8, 24] does not naturally extend to arbitrary features. Second, RFs, partition the feature space with axis-parallel splits [21].

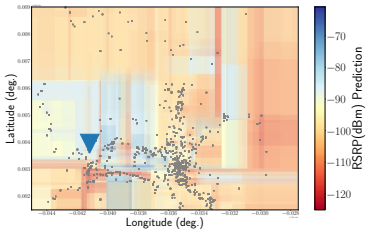


Figure 2: Example of decision boundaries chosen by $\text{RFs}_{x,y}$ for Campus cell x306. We can see that RFs can naturally identify spatially correlated measurements, *i.e.,* regions with similar wireless propagation characteristics.

An example of decision boundaries produced by $\text{RFs}_{x,y}$ is depicted in Fig. 2. One can see the splits according to the spatial coordinates (lat, lng) and the produced areas agree with our knowledge of the placement and direction of antennas on campus. Automatically identifying these regions with spatially (and temporal) correlated RSRP comes for free to RFs and is particularly important in RSRP prediction because wireless propagation has different properties across neighborhoods [26]. In contrast, prior art (OKP, [8]) requires additional preprocessing for addressing this spatial heterogeneity.

2.4 Baseline: Geospatial Interpolation

State-of-the-art approaches in data-driven RSS prediction [8, 13, 20, 24] have primarily relied on geospatial interpolation, which however is inherently limited to only spatial features (l^x, l^y). The best representative of this family of predictors is ordinary kriging (OK) [20] and its variants [8], which are used as baselines for comparison in this paper.

Ordinary Kriging (OK): It predicts RSS at the testing location $\vec{l}_j = (l^x_j, l^y_j)$ as a weighted average of the K nearest measurements in the training set: $P_j = \sum_{i=1}^{K} w_i P_i$. The weights w_i are computed by solving a system of linear equations that correlate the test with the training data via the semivariogram function $\gamma(h)$ [8].

Ordinary Kriging Partitioning (OKP) [8]: Voronoi-based partitioning is used to identify regions with the same PLE and apply a distinct OK model in each region. It is comparable to the heterogeneous LDPL, yet is impractical for city wide signal maps.

Ordinary Kriging Detrending (OKD) [8, 24]: OK assumes spatial stationarity, which does not hold for RSRP. OKD incorporates a version of LDPL in the prediction in order to address this issue [8]. This can be thought as a hybrid approach of geospatial and LDPL and serves as our baseline for comparison in this paper.

3 DATA SETS

Table 2 summarizes the two datasets used in this paper. The first is a campus dataset and the second consists of two city-wide datasets from NYC and LA. Fig. 1 depicts some examples and Table 3 summarizes representative LTA TAs examples used in our evaluation.

Table 2: Overview of Signal Maps Datasets used in this study.

Dataset	Period	Areas	Type of Measurements	Characteristics	Source
Campus	02/10/17 - 06/18/17	Univ. Campus Area $\approxeq 3km^2$	LTE KPIs: RSRP, [RSRQ]. Context: GPS Location, timestamp, dev, cid. Features: $\mathbf{x} = \left(l_j^x, l_j^y, d, h, dev, out, ||\vec{l}_{\text{BS}} - \vec{l}_j||_2\right)$	No. Cells = 25, No. Meas $\approxeq$ 180K Per Cell Density: 0.01 - 0.66 (Table 3) Overall Density: 0.06	Ourselves [3]
NYC & LA	09/01/17-11/30/17	NYC Metropolitan Area $\approxeq 300km^2$ / LA metropolitan Area $\approxeq 1600km^2$	LTE KPIs: RSRP, [RSRQ, CQI]. Context: GPS Location, timestamp, dev, cid. EARFCN. Features:$\mathbf{x} = \left(l_j^x, l_j^y, d, h, cid, dev, out, ||\vec{l}_{\text{BS}} - \vec{l}_j||_2, freq_{dl}\right)$	No. Meas NYC $\approxeq$ 4.2M, No. Cells NYC $\approxeq 88k$ Density NYC-all $\approxeq 0.014\ N/m^2$ / No. Meas LA $\approxeq$ 6.7M, No. Cells LA $\approxeq$ 111K Density LA-all $\approxeq 0.0042\ N/m^2$	Mobile Analytics Company

3.1 Campus dataset

Dataset Overview. We collected the first dataset on UC Irvine campus. This Campus dataset is relatively small: 180,000 data points, collected by seven different users, using 2 cellular providers. In terms of geographical area, it covers approximately $3km^2$, as the devices move between locations (*e.g.,* housing, office *etc.*) on campus. Some examples are depicted in Fig. 1(a)-(b). Although small, the cells in this dataset exhibit a range of characteristics (reported in Table 4) regarding the (i) number of measurements, (ii) mean and variance of RSRP and (iii) dispersion and density (*e.g.,* multiple measurements over time on the same or nearby locations).

Data Collection. We developed a user-space app [1, 3] that uses the Android APIs to obtain radio layer and other information needed for RSRP prediction. Although the design of the monitoring system itself is challenging, we defer to [3] due to space limitations. No personally identifiable information is collected.

3.2 NYC and LA datasets

Dataset Overview. The second type consists of much larger datasets: 10.9M measurements in total, covering approx. $300km^2$ and $1600km^2$ in the metropolitan areas of NYC and LA, respectively, for a period of 3 months (Sep'17 - Nov'17). Key characteristics are summarized in Table 2. An example of the NYC Midtown Manhattan neighborhood is depicted in Fig. 1(c)-(d). While these are large datasets, they are also relatively sparse in space and heterogeneous. For example, the density is now approx. 300 measurements per cell on average and up to the order of 20 thousands max. We only consider cells with more than 100 measurements. There is also sparsity in time: unlike the Campus dataset, there are no longer multiple measurements at different times for the same location.

Data Collection. This dataset was collected by a major mobile analytics crowdsourcing company and shared with us. They collect measurements from a large user base infrequently so as to not burden each end-users, which explains the smaller overall density of the dataset, as shown in Table 2. Each location data point is accompanied by rich network and contextual information, except for device or other personal identifiers.

3.3 Common Description of Datasets

Data Format. For the purposes of RSRP prediction, we use the same subset of information from all datasets, *i.e.,* RSRP values and the corresponding features defined in Section 2.1. We utilize GeoJSON format to represent and process our data, which offers several practical advantages.

Properties of Datasets. For each dataset, the following metrics describe characteristics that affect RSRP prediction.

• *Data Density:* Number of measurements per unit area (N/m^2).

Table 3: NYC and LA datasets**: LTE TAs Examples.**

	NYC (MNC-1) Manhattan Midtown	NYC (MNC-1) E. Brooklyn	LA (MNC-2) Southern
No. Measurements	$\approxeq$ 63K	$\approxeq$ 104K	$\approxeq$ 20K
Area km^2	$1.8km^2$ (Fig. 1 (c-d))	44.8 km^2	220 km^2
Data Density N/m^2	$\approxeq 0.035$	$\approxeq 0.002$	$\approxeq 0.0001$
No. Cells $\|C\|$	429	721	353
Cell Density $\|C\|/km^2$	238.3	16.1	1.6

• *Cells Density:* Number of unique cells (*cid*s) per unit area, *i.e.,* $|C|/km^2$. The higher it is, the more *cID* helps as a feature.

• *Dispersion:* In order to capture how concentrated or dispersed are the measurements in an area, we use the Spatial Distance Deviation (*SDD*) metric [18], defined as the standard deviation of the distance of measurement points from the center.

OpenCellID. Both the LDPL and in the RFs predictors need the distance between the transmitting antenna and the receiver's location (where RSRP is measured or predicted), $||\vec{l}_{\text{BS}} - \vec{l}_j||_2$. To that end, we lookup the location of the BS, $\vec{l}_{\text{BS}}$, using the public APIs of a popular online crowdsourced database opencellid.org.

4 PERFORMANCE EVALUATION

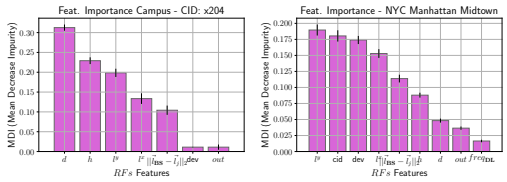
4.1 Setup

4.1.1 RFs *Setup.* The most important hyper-parameters for RFs are the number of decision trees, *i.e.,* n_{trees}, and the maximum depth of each tree, *i.e.,* $\max_{depth}$. We used a grid search over the parameter values of the RFs estimator [23] in a small hold-out part of the data to select the best values. For the Campus dataset, we select $n_{trees} = 20$ and $\max_{depth} = 20$ via 5-Fold Cross-Validation (*CV*); larger $\max_{depth}$ values could overfit RFs. For the NYC and LA datasets, we select $n_{trees} = 1000$ and $\max_{depth} = 30$; more and deeper trees are needed for larger datasets. One important design choice is what granularity we choose to build our RFs models: per *cID* or per LTE TA (defined in Section 2.1).

•*Training per cID:* We can train a separate RFs model per cell using all features except *cID* ($\mathbf{x_j^{-cID}}$), since RSRP is measured per serving cell, but requires many measurements per cell. This is the case in Campus dataset but not in NYC and LA datasets.

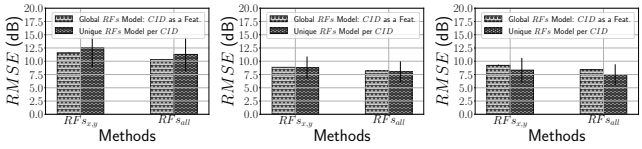
•*Training per LTE TA:* Another option is to train one RFs model per LTE TA and use *cID* as one of the features in $\mathbf{x_j^{full}}$. This is particularly useful in the NYC dataset, where there are less measurements per cell unit area, insufficient to train a model per *cID*. In urban areas, there is very high cells density in a region and data points from different cells in the same LTE TA can be useful.

4.1.2 Baselines' Setup. For LDPL-knn: we select empirically $k = 100$ neighbors for the Campus dataset and $k = 10\%$ of the training data points in each cell for the NYC and LA datasets. For *Geospatial Predictors,* the number of neighbors was empirically set to $k = 10$; a larger k did not show significant improvement. An exponential fitting function of the semivariogram function $\gamma(h)$



(a) Cell x204 (b) Manhattan Midtown TA

Figure 3: For Campus dataset (a) : Feature Importance for cell x204 (RFs built per distinct cell). Cells' data are depicted in Fig. 1. For NYC dataset, (b) shows the MDI score for one LTE TA for MNC-1.



(a) MNC-1, NYC Manhattan Midtown (urban) (b) MNC-1, East Brooklyn (suburban) (c) MNC-2, Southern LA (suburban)

Figure 4: RMSE in NYC and LA datasets. This figure makes multiple comparisons: (1) urban vs suburban LTE TAs; (2) cID as feature vs. training a different RFs model per cID; (3) providers MNC-1 vs. MNC-2.

was selected [8]; the maximum lag (h) was set to 200m, as in [8], for the Campus and NYC environments, while it was set to 600m for the LA suburban environment. The empirical semivariogram $\widehat{\gamma(h)}$ was calculated per $10m$ [8].

4.1.3 Splitting Data into Training and Testing. We select randomly 70% of the data as the training set $\{\mathbf{X}_{train}, \mathbf{P}_{train}\}$ and 30% of the data as the testing set $\{\mathbf{X}_{test}, \mathbf{P}_{test}\}$ for the problem of predicting missing RSRP values. The results are averaged over $S = 5$ independent random splits. These default choices are used unless otherwise stated. An exception is Fig. 5, where we vary the size of training set and we show that our RFs-based predictors degrade slower than baselines with decreasing training size.

4.1.4 Evaluation Metrics. We evaluate the performance of the predictors in terms of absolute error (RMSE) and Absolute Relative Improvement (ARI) as well as feature importance in RFs.

Root Mean Square Error (RMSE): If $\widehat{P}$ is an estimator for P, then $RMSE(\widehat{P}) = \sqrt{MSE(\widehat{P})} = \sqrt{E((P-\widehat{P})^2)}$, in dB. We report $RMSE$ for each predictor at different levels of granularity, namely: (i) per cID (ii), per LTE TA (in NYC and LA), (iii) over all data (Campus).

Absolute Relative Improvement (ARI): This captures the improvement of each predictor over the variance in the data: $ARI = 1 - (1/|\mathrm{C}|)\sum_{i\in\mathrm{C}}(MSE_i/Var_i)$, where |C| is the number of the different cells in the dataset, and Var_i is cell i's variance.

Mean Decrease Impurity (MDI), a.k.a. Gini Importance: It captures how often a feature is used to perform splits in RFs. It is defined as the total decrease in node impurity, weighted by the probability of reaching that node (approx. by the proportion of samples reaching that node), averaged over all trees in the ensemble [23].

4.2 Results

4.2.1 Feature Importance.

a. Campus dataset*:* We train one RFs model per cID for the set of features $\mathbf{x} = (l_j^x, l_j^y, d, h, ||\vec{l}_{\mathrm{BS}} - \vec{l}_j||_2, out, dev)$. Results w.r.t. MDI are shown on Fig. 3. We observe that, in cells with high data density and low dispersion, the most important are the time features (d, h) w.r.t. to MDI. An example of such a cell is x204 (depicted in Fig. 1(a)), which has $SDD = 325$, density=0.66 points/m^2 and its MDI is shown in Fig. 3(a). Feature importance for dev and out are close to zero, which is expected because of the small number of devices in the Campus dataset. For more dispersed and less dense cells, such as cell x355 ($SDD = 573, 0.116N/m^2$, map in Fig. 1(b)), the location features (l_j^x, l_j^y) have higher MDI; results are omitted.

b. NYC and LA datasets*:* In this case, $freq_{dl}$ is available and the datasets contain thousands of cells. We start with a RFs model per LTE TA. As a representative example, we report the feature importance, in Fig. 3(b), for the LTE TA of a major mobile network carrier (MNCarrier-1) located in NYC Midtown Manhattan and already depicted in Fig. 1(c)-1(d). The most important features turn out to be the spatial features (l_j^x, l_j^y) as well as the cell cID and dev. This is because the data are sparser and the whole LTE TA is served by geographically adjacent or overlapping cells.

We also investigated whether we should train a separate RFs per cID, or cID should be used as one of the features in a single RFs. For a representative urban LTE TA (Manhattan Midtown), in Fig. 4(a) we calculate the $RMSE$ for two cases: (i) when cID is used as a feature in a single RFs per LTE TA and (ii) when a separate RFs model is produced per cell. Interestingly, the prediction is better when cID is utilized as a feature. Given the sparsity of the data and the high overlap of the cells, RFs benefit from the features of the additional measurements. Manhattan Midtown has a cells density of 238 per km^2 (see Table 3): the cell size does not exceed the size of a few blocks or sometimes there are multiple cells within a skyscraper. On the contrary, for the suburban LA dataset, where the cells are not so densely deployed, a unique RFs model per cell performs better than RFs per LTE TA, as shown in Fig. 4(c). In the Campus dataset (lower density than NYC), the RFs model per cID did better than using as a feature in a single RFs model for the entire LTE TA. Similar results and findings were observed for the rest of cells and TAs, but are omitted due to space constraints. In general, RFs trained per cID is usually a better option, but cID should be used as a feature in urban areas with high cells density.

Table 4: Campus dataset: Comparing Predictors per cell

	Cell Characteristics					$RMSE$ (dB)						
cID	N	$\frac{N}{\text{sq } m^2}$	SDD	$\mathbb{E}[P]$	σ^2	LDPL hom	LDPL kNN	OK	OKD	RFs x,y	RFs x,y,t	RFs all
x312	10140	0.015	941	-120.6	12.0	17.5	1.63	1.70	1.37	1.58	0.93	**0.92**
x914	3215	0.007	791	-94.5	96.3	13.3	3.47	3.59	2.28	3.43	1.71	**1.67**
x034	1564	0.010	441	-101.2	337.5	19.5	7.82	7.44	5.12	7.56	3.82	**3.84**
x901	16051	0.162	355	-107.9	82.3	8.9	4.60	4.72	3.04	4.54	1.73	**1.66**
x204	55566	0.666	325	-96.0	23.9	6.9	3.84	3.85	2.99	3.83	2.30	**2.27**
x922	3996	0.107	218	-102.7	29.5	5.6	3.1	3.16	2.01	3.10	1.92	**1.82**
x902	34193	0.187	481	-111.5	8.1	21.0	2.60	2.47	1.64	2.50	1.37	**1.37**
x470	7699	0.034	533	-107.3	16.9	24.8	3.64	2.73	1.87	2.78	1.26	**1.26**
x915	4733	0.042	376	-110.6	203.9	14.3	7.54	7.39	4.25	7.31	3.29	**3.15**
x808	12153	0.035	666	-105.1	7.7	4.40	2.41	2.42	1.60	2.34	1.75	**1.59**
x460	4077	0.040	361	-88.0	32.8	11.2	2.35	2.28	1.56	2.31	1.84	**1.84**
x306	4076	0.011	701	-99.2	133.3	18.3	4.85	4.30	2.80	3.94	3.1	**3.06**
x355	30084	0.116	573	-94.3	42.6	9.3	2.42	2.31	1.85	2.26	1.79	**1.79**

4.2.2 Comparing RSRP Predictors. We compare against baselines, both geospatial interpolation (OK and OKD) and model-driven (LDPL-knn and LDPL-hom).

a. Campus dataset*:* Table 4 reports the $RMSE$ for all predictors for each cell in the Campus dataset, for the default 70-30% split.

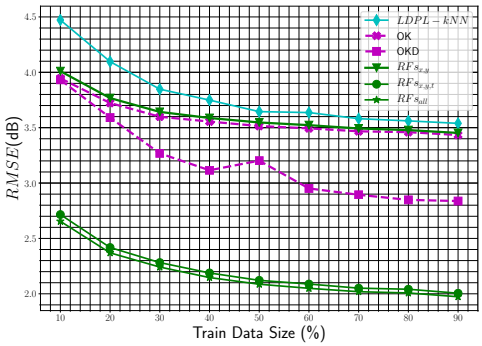


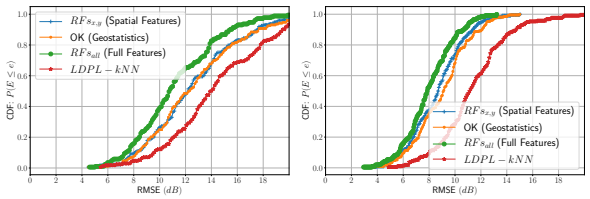
Figure 5: Campus dataset: *RMSE* **vs. Training Size. Our methodology (RFs with more than spatial features, *i.e.,* $RFs_{x,y,t}$, RFs_{all}) significantly improves the RMSE-cost tradeoff: it can reduce** *RMSE* **by 17% for the same number of measurements compared to state-of-the-art data-driven predictors(OKD);** ***or*** **it can achieve the lowest error possible by OKD (≃ 2.8dB) with 10% instead of 90% (and 80% reduction) of the measurements.**

Fig. 7 compares all methods, calculating *RMSE* over the entire Campus dataset, instead of per cell. We can see that our RFs-based predictors ($RFs_{x,y,t}$, RFs_{all}) outperform model (LDPL) and other data-driven (OK, OKD) predictors.

Fig. 5 shows the *RMSE* as a function of the training size (as % of all measurements in the dataset). First, the performance of OK and $RFs_{x,y}$ is almost identical, as it can be seen for *RMSE* over all measurements (Fig. 5 and Fig. 7) and *RMSE* per cell (Table 4). This result can be explained by the fact that both predictors are essentially a weighted average of their nearby measurements, although they achieve that in a different way: OK finds the weights by solving an optimization problem while $RFs_{x,y}$ uses multiple decision trees and data splits. Second and more importantly, considering additional features can significantly reduce the error. For the Campus dataset, when time features $\mathbf{t} = (d, h)$ are added, $RFs_{x,y,t}$ significantly outperforms OKD: it decreases *RMSE* from 0.7 up to 1.2 dB. Alternatively, in terms of training size, $RFs_{x,y,t}$ needs only 10% of the data for training, in order to achieve OKD's lowest error (≃ 2.8dB) with 90% of the measurement data for training. Our methodology achieves the lowest error of state-of-the-art geospatial predictors with 80% less measurements. The absolute relative improvement of $RFs_{x,y,t}$ compared to OKD is 17%, shown in Fig. 7(b).

b. NYC and LA datasets*:* Fig. 6 shows the error for two different LTE TAs, namely for NYC Manhattan Midtown (urban) and for southern LA (suburban), where RFs have been trained per *cID*. CDFs of the error per *cID* of the same LTE TA are plotted for different predictors. Again, OK performance is very close to $RFs_{x,y}$, because they both exploit spatial features. However, RFs_{all} with the rich set of features improves by approx. 2dB over the baselines for the 90th percentile, in both LTE TAs. Interestingly, the feature *dev* is important, which is expected since this data, has heterogeneous devices reporting RSRP.

There are multiple reasons why RFs_{all} outperform geospatial interpolation predictors. The mean and variance of RSRP depend on time and location and the complex propagation environment. RFs can easily capture these dimensions instead of modeling a priori every single aspect. For example, $RFs_{x,y,t}$ predicts a time-varying value for the measurements at the same location in Fig. 1(a), while $RFs_{x,y}$ or OK/OKD produce just a flat line over time. OK also



(a) NYC Manhattan Midtown (b) LA Southern Suburb

Figure 6: NYC and LA datasets**: CDFs for** *RMSE* **per** *cID* **for two different LTE TAs, for the same major MNCarrier-1.** RFs_{all} **offer 2dB gain over the baselines for the 90th percentile.**

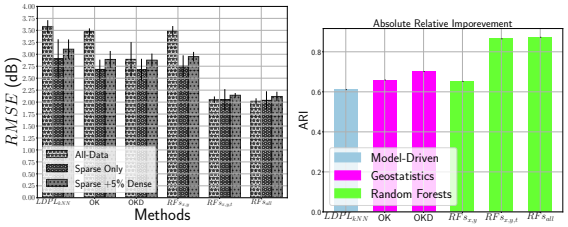


Figure 7: Comparison of all predictors over the entire Campus dataset**. Left (a)** $RMSE(dB)$ **under various scenarios, Right (b) ARI over all data points. Our Approaches ($RFs_{x,y,t}$, RFs_{all}) outperform prior art in all scenarios.**

relies on some assumptions (same mean over space, semivariogram depending only on the distance between two locations), which do not hold for RSRP. Even hybrid geospatial techniques (OKD) cannot naturally incorporate additional features (*e.g.,* time, device type, *etc.*). Finally, RFs significantly outperform propagation models, except for a few areas with limited number of measurements.

4.2.3 Location density and overfitting. In the Campus dataset, we observed that a significant fraction of the data comes from a few locations, *i.e.,* from grad students' home and work, which begs the question whether this leads to overfitting. We investigated this question and found that our RFs predictors neither get a performance boost nor overfit. To that end, we utilize HDBSCAN [7], a state-of-the-art clustering algorithm, to identify very dense (spatially) clusters of measurements (cluster size 5% of the cell's data). We refer to data from those locations as "dense"; we remove them and we refer to the remaining ones as "sparse-only" data. Fig. 7(a) reports the *RMSE* of different methods when training and testing is based on (i) all-data, (ii) sparse-only data and (iii) sparse-only data with a 5% randomly sampled from the dense data. It can be clearly seen that our $RFs_{x,y,t}$ and RFs_{all} have similar performance in all scenarios and consistently outperform baselines (similarly in cell-by-cell basis; results omitted). Please note that OK and LDPL-knn's errors slightly decrease for "sparse-only"; OK cannot handle repeated locations and LDPL-knn may overfit.

5 CONCLUSION

We developed a machine learning framework for cellular signal strength prediction, which is important for creating signal maps in a cost-efficient way, crucial for future 5G and IoT deployments. We used the powerful tool of random forests, which we adapted in this context by evaluating different features. The datasets under study are the largest used in this context and provide unique insight into city-wide signal map prediction. Future work includes a hybrid ML-propagation model to harvest the diversity of both worlds.

REFERENCES

[1] A. Shubaa, A. Le, E. Alimpertis, M. Gjoka, A. Markopoulou. 2016. AntMonitor: System and Applications. *arXiv:1611.04268* (Nov. 2016).

[2] E. Alimpertis, N. Fasarakis-Hilliard, and A. Bletsas. 2014. Community RF sensing for source localization. *IEEE Wireless Comm. Letters* 3, 4 (2014), 393–396.

[3] E. Alimpertis and A. Markopoulou. 2017. A system for crowdsourcing passive mobile network measurements. In *14th USENIX NSDI'17, Posters Sessions*. USENIX Association, Boston, Massachusetts, USA.

[4] T. Bilen, B. Canberk, and K. R. Chowdhury. 2017. Handover Management in Software-Defined Ultra-Dense 5G Networks. *IEEE Network* 31, 4 (2017), 49–55.

[5] L. Breiman. 2001. Random Forests". *Machine Learning* 45, 1 (Oct. 2001), 5–32.

[6] Y. J. Bultitude and T. Rautiainen. 2007. *IST-4-027756 WINNER II D1. 1.2 V1. 2 WINNER II Channel Models.* Technical Report.

[7] R. JGB Campello, D. Moulavi, and J. Sander. 2013. Density-based clustering based on hierarchical density estimates. In *PAKDD '13*. Springer, 160–172.

[8] A. Chakraborty, Md S. Rahman, H. Gupta, and S. R Das. [n. d.]. SpecSense: Crowdsensing for Efficient Querying of Spectrum Occupancy. In *Proc. of the IEEE INFOCOM '17*. 1–9.

[9] B. Cici, E. Alimpertis, A. Ihler, and A. Markopoulou. 2016. Cell-to-cell activity prediction for smart cities. In *Proc. of the IEEE INFOCOM WKSHPS '16*. IEEE, 903–908.

[10] H. Deng, C. Peng, A. Fida, J. Meng, and Y C. Hu. 2018. Mobility Support in Cellular Networks: A Measurement Study on Its Configurations and Implications. In *Proc. of the ACM IMC '18*. ACM, 147–160.

[11] R. Enami, D. Rajan, and J. Camp. 2018. RAIK: Regional analysis with geodata and crowdsourcing to infer key performance indicators. In *Proc. of the IEEE WCNC '18*. 1–6.

[12] ETSI. 2015. LTE, Evolved Universal Terrestrial Radio Access (E-UTRA), Physical layer Measurements (3GPP TS 36.214 version 12.2.0 Release 12). `http://www.etsi.org/`.

[13] M. R. Fida, A. Lutu, M. K. Marina, and O. Alay. 2017. ZipWeave: Towards efficient and reliable measurement based mobile coverage maps. In *Proc. of the IEEE INFOCOM '17*. IEEE.

[14] A. Gomez-Andrades, R. Barco, P. Munoz, and I. Serrano. 2017. Data analytics for diagnosing the RF condition in self-organizing networks. *IEEE Transactions on Mobile Computing* 16, 6 (2017), 1587–1600.

[15] S. He and K. G. Shin. 2018. Steering Crowdsourced Signal Map Construction via Bayesian Compressive Sensing. In *Proc. of the IEEE INFOCOM '18*. IEEE, 1016–1024.

[16] A. Imran, A. Zoha, and A. Abu-Dayya. 2014. Challenges in 5G: How to empower SON with big data for enabling 5G. *IEEE network* 28, 6 (2014), 27–33.

[17] J. Johansson, W. A Hapsari, Sean Kelley, and G. Bodog. 2012. Minimization of drive tests in 3GPP release 11. *IEEE Comm. Magazine* 50, 11 (2012).

[18] Z. Li, A. Nika, X. Zhang, Y. Zhu, Y. Yao, B. Y Zhao, and H. Zheng. 2017. Identifying value in crowdsourced wireless signal measurements. In *Proc. of the ACM WWW '17*. ACM, 607–616.

[19] R. Margolies, Richard Becker, S. Byers, S. Deb, R. Jana, S. Urbanek, and C. Volinsky. 2017. Can you find me now? Evaluation of network-based localization in a 4G LTE network. In *Proc. of the IEEE INFOCOM '17*. 1–9.

[20] M. Molinari, M R. Fida, M. K Marina, and A. Pescape. 2015. Spatial Interpolation Based Cellular Coverage Prediction with Crowdsourced Measurements. In *Proc. of the ACM SIGCOMM Workshop on Crowdsourcing and Crowdsharing of Big Internet Data (C2BID)*. ACM, 33–38.

[21] K. P. Murphy. 2012. *Machine Learning: A Probabilistic Prespective.* The MIT Press, Cambridge, Massachusetts.

[22] Open Signal Inc. 2011. 3G and 4G LTE Cell Coverage Map. `http://www.opensignal.com`.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12, Oct. (2011), 2825–2830.

[24] C. Phillips, M. Ton, D. Sicker, and D. Grunwald. 2012. Practical radio environment mapping with geostatistics. *Proc. of the IEEE DYSPAN '12* (Oct. 2012), 422–433.

[25] T. Rappaport. 2001. *Wireless Communications: Principles and Practice* (2nd ed.). Prentice Hall PTR, Upper Saddle River, NJ, USA.

[26] A. Ray, S. Deb, and P. Monogioudis. 2016. Localization of LTE measurement records with missing information. In *Proc. of the IEEE INFOCOM '16*.

[27] COST 231 Final Report. 1999. *Digital mobile radio towards future generation systems.* Technical Report.

[28] Root Metrics Inc. 2018. Metro RootScore Reports. `http://www.rootmetrics.com/us`.

[29] Bernard Sklar. 1997. Rayleigh fading channels in mobile digital communication systems. I. Characterization. *IEEE Comm. Magazine* 35, 7 (1997), 90–100.

[30] Tutela Inc. 2011. Crowdsourced mobile data. `http://www.tutela.com`.

[31] J. Yang, A. Varshavsky, H. Liu, Y. Chen, and M. Gruteser. 2010. Accuracy characterization of cell tower localization. In *Proc. of the ACM UbiComp '10*. ACM, 223–226.

[32] Z. Yun and M. F. Iskander. 2015. Ray tracing for radio propagation modeling: Principles and applications. *IEEE Access* 3 (2015), 1089–1100.

Is Yelp Actually Cleaning Up the Restaurant Industry? A Re-Analysis on the Relative Usefulness of Consumer Reviews

Kristen M. Altenburger
Stanford University
kaltenb@stanford.edu

Daniel E. Ho
Stanford University
dho@law.stanford.edu

ABSTRACT

Social media provides the government with novel methods to improve regulation. One leading case has been the use of Yelp reviews to target food safety inspections. While previous research on data from Seattle finds that Yelp reviews can predict unhygienic establishments, we provide a more cautionary perspective. First, we show that prior results are sensitive to what we call "Extreme Imbalanced Sampling": *extreme* because the dataset was restricted from roughly 13k inspections to a sample of only 612 inspections with only extremely high or low inspection scores, and *imbalanced* by not accounting for class imbalance in the population. We show that extreme imbalanced sampling is responsible for claims about the power of Yelp information in the original classification setup. Second, a re-analysis that utilizes the full dataset of 13k inspections and models the full inspection score (regression instead of classification) shows that (a) Yelp information has lower predictive power than prior inspection history and (b) Yelp reviews do not significantly improve predictions, given existing information about restaurants and inspection history. Contrary to prior claims, Yelp reviews do not appear to aid regulatory targeting. Third, this case study highlights critical issues when using social media for predictive models in governance and corroborates recent calls for greater transparency and reproducibility in machine learning.

KEYWORDS

consumer reviews; Yelp; food safety; regulation; replication

ACM Reference Format:
Kristen M. Altenburger and Daniel E. Ho. 2019. Is Yelp Actually Cleaning Up the Restaurant Industry? A Re-Analysis on the Relative Usefulness of Consumer Reviews. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313683

1 INTRODUCTION

Governments are increasingly exploring predictive approaches with large-scale datasets to improve regulation. Data mining approaches have witnessed great success across domains such as inferring wealth in a country using mobile phone records [5] or satellite images [31] or inferring migration stocks using both social media data like Facebook and survey data [53]. Social media datasets like Twitter have the potential to provide an additional signal of future, offline outcomes of interest as illustrated by predicting electoral outcomes [16], health outcomes [12], and many others [40].

Increasingly, government usage has expanded to leverage social media data for predictive governance [34]. Police departments have mined Facebook and Twitter for investigation and predictive policing [48]. Municipal service departments have used social media to forecast the need for municipal services, such as filling potholes [1]. Governments have used traffic complaints on Twitter to shape transportation planning [51]. One of the leading examples has been the case of food safety [14, 19, 23, 33, 44, 46, 47]. Health departments from major cities, including New York, Las Vegas, Boston, and Chicago, have, for instance, used Yelp and Twitter terms, such as "sick" and "vomit," to allocate enforcement and investigatory resources.

While governments have experimented with such novel data uses, researchers have begun to examine the reliability of such approaches. A prominent example is that of Google Flu Trends. Google Flu Trends was initially advertised as using search terms to forecast influenza epidemics, beating early detection of the Centers for Disease Control and Prevention (CDC) [18]. But researchers showed that the algorithm was prone to substantial overfitting [36]. As put in *Science*, big data "does not mean that one can ignore measurement and construct validity and reliability" [36].

Our article makes three contributions. First, we revisit the seminal case of food safety. Earlier work showed that the use of social media may import private biases into public enforcement [2, 11]. We show here instead that the leading case of social media data use for food safety [33] has substantial methodological shortcomings: (a) changing the prediction problem to predict only extremely hygienic and extremely unhygienic establishments and (b) failing to assess the impact of class imbalance in the population. We call this a case of "extreme imbalanced sampling." We show that a single, seemingly minor correction – drawing a random sample of hygienic establishments (as opposed to using the nonrandom order in the original dataset, which overrepresents the same establishments) – causes the originally reported results to disappear. This analysis follows other social media studies documenting classification results that are overoptimistic [9] or limited in their data sampling approaches [52].

Second, we conduct a comprehensive analysis, by using the full dataset of about 13k inspections, rather than the subset of approximately 612 inspections originally analyzed. We find that (a) Yelp reviews provide substantially less predictive power than inspection history already possessed by the health department, and (b) Yelp reviews do not significantly improve predictions, given existing information about restaurants and inspection history.

Third, we discuss some of the implications both for machine learning and government use of social media. Most importantly,

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313683

our paper points to emerging recognition of a replication crisis in machine learning [27, 49]. Methodological flubs can easily occur inadvertently with complex methods, and we highlight that the implications are severe when it comes to public policy. News outlets, including *NPR*, the *New York Times*, *Forbes*, and *Newsweek*, quickly touted the idea of food safety targeting based on social media. And private industry may have an incentive to promote use cases before thorough vetting. Yelp's CEO, for instance, proclaimed that Yelp could "beat gold standard healthcare measures" [50]. Our evidence shows that much more thought needs to be given for ethical and reliable use by government of social media [21].

Our article proceeds as follows. In Section 2, we describe the Seattle restaurant dataset that led to provocative claims that Yelp could "clean up the restaurant industry" [3]. In Section 3, we assess the original analysis that purported to show that Yelp reviews could be used to predict unhygienic establishments. We highlight how a single, minor modification – random rather than non-random sampling of the majority class – invalidates the claim that Yelp terms are more informative than inspection history. In Section 4, we conduct a more comprehensive analysis that (a) makes use of all of the data (13k as opposed to 612 inspections), (b) uses all of the information in inspection outcomes (regression vs. classification), and (c) uses word embeddings to exploit review terms [42]. In Section 5, we discuss the more general implications for replication in machine learning and government use of social media data.

2 THE SEATTLE RESTAURANT DATASET

The prediction problem considered throughout this work is to infer the inspection penalty status of a restaurant (either in terms of a restaurant's raw inspection score or classifying restaurants as unhygienic if above a certain score threshold [33]) given features created based on the restaurant, online Yelp reviews, or past inspection history. This prediction problem can be framed as either a classification problem if one sets a threshold for defining "unhygienic" restaurants or a regression problem if one aims to predict the actual prediction score. We will use the terms penalty score and inspection score interchangeably throughout this work.

Data: We utilize the Yelp and inspection dataset compiled in earlier work [33], which we summarize here. In Seattle, food safety inspectors conduct routine inspections one to two times a year per establishment, resulting in an inspection score based on health code violations. The data contain about 13k inspections covering 1,756 Seattle restaurants with 152k Yelp reviews from 2006 to 2013. Yelp reviews are collected for the preceding period of time between inspections (or six months earlier when no prior inspection existed). Review data is concatenated as a text string.[1]

Features: We evaluate three categories of features for predicting a restaurant's sanitation level: restaurant-fixed features (cuisine and location), inspection history features, and features derived from Yelp for the preceding inspection period (e.g., average rating, review text). We list these features in Table 1. Inspection history includes information derived from an establishment's past inspection record (e.g., average past inspection history or normalized metrics).[2] The principal question is whether Yelp reviews help predict sanitation.

Table 1: Features for prediction. Inspection history includes an establishment's previous penalty score and an average measure across previous inspections. Cuisine indicates type of food offered, and ZIP code denotes restaurant's location. Yelp features we consider include ratings, review counts, and then the review text itself using a TF-IDF or word embedding representation. The negative review count feature is referred to as non-positive review count in [33].

Feature Name	Feature Category
inspection history	Inspection
cuisine	Restaurant
zip code	Restaurant
avrg. rating	Yelp
negative review count	Yelp
review count	Yelp
review text	Yelp

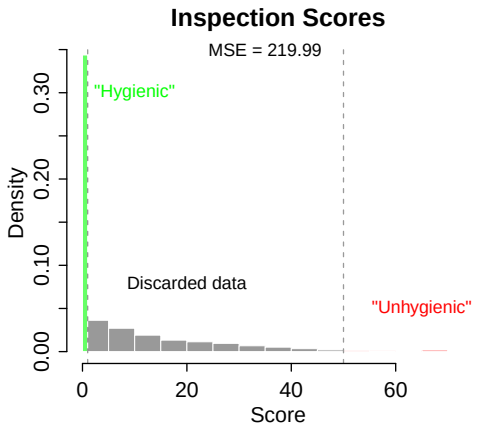


Figure 1: Distribution of inspection scores, with vertical lines indicating cutoffs for "unhygienic" restaurants (red) with scores above 50 and "hygienic" restaurants (green) with scores of 0. This figure shows how the prior analysis sampled from the extremes of the distribution.

Prior Analysis: Prior analysis of this data attempted to classify "hygienic" and "unhygienic" restaurants. First, all features were calculated on the full dataset of approximately 13k observations and outcomes were labeled as "unhygienic" if the inspection score was greater than 50.[3] Figure 1 displays the outcome distribution, showing highly skewed scores. Only 306 restaurants, 2.3% of the sample, are labeled as "unhygienic," resulting in an extremely imbalanced setting for classification. Second, because of this class imbalance, 306 "hygienic" restaurants with no violation history

[1] See https://www3.cs.stonybrook.edu/~junkang/hygiene/. We are not able to observe different reviews or reviewers. While there are other data quality issues, such as duplicated review texts, we do not address these issues here for comparability.

[2] While cuisine information is technically derived from Yelp data, we denote this as a restaurant category because this information is not derived from user-input. It is common for health agencies to retain information about cuisine.

[3] While the original paper describes the labels as a non-strict inequality, the implementation code uses a strict inequality (see line 720 of train_model.py). The original analysis also assessed performance at other thresholds, but mainly focused on the 50 point threshold as that with highest accuracy (see Figure 3 and Table 1 in [33]).

were (non-randomly) selected from the majority class.[4] We discuss below the effects of this non-random selection. Figure 1 shows that 34% of restaurants receive scores of 0 and the prior analysis discards 63% of restaurants receiving scores between 0 and 50.

Third, the prior analysis trained ℓ_1-regularized support vector machines (SVMs), with a constant C parameter, for each feature separately, evaluating results via 10-fold cross-validation on the dataset of 612 observations. By only evaluating predictive performance on this dataset of 612 observations, predictive performance may be overstated compared to performance on the full population of 13k inspections. The top left panel of Figure 3 presents accuracy statistics as reported in the original paper.[5] Because unigrams and bigrams from Yelp reviews outperformed inspection history, the study concluded that Yelp reviews can aid in targeting food safety inspections. We obtained the original code and successfully replicate the results in the top right panel of Figure 3. Minor differences likely stem from different train/test splits, as random seeds were not originally set.

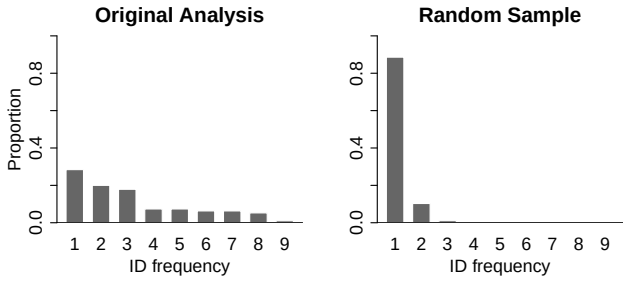


Figure 2: Histogram of counts of occurrences for a unique hygienic restaurant (i.e. defined by inspection scores of 0) in the original analysis (left panel) and under one instance of a random sample of 306 hygienic restaurants (right panel). This panel shows that by selecting observations in the non-random order of the dataset (sorted by restaurant ID), the hygienic sample overrepresents specific establishments compared to a sample one would expect under random sampling.

3 EXTREME IMBALANCED SAMPLING

We now explain the methodological challenges in the prior analysis. First, rather than predicting hygienic and non-hygienic restaurants with a particular cutoff for classification, the prior analysis dramatically simplified the prediction problem. Hygienic restaurants were those with inspection scores of 0 violations[6] (green in Figure 1) and unhygienic restaurants were those with over 50 violation points (red in Figure 1). In other words, the analysis sampled extreme end points, making the prediction substantially easier. Consider an educational analogy: instead of predicting the full grade range for students, it is akin to classifying straight-A students versus drop-out students, discarding the majority of students in the mid-range.

Moreover, hygienic restaurants were not randomly selected from the 4,571 hygienic observations. Instead, they were selected based on (non-random) order in the dataset, which was sorted by restaurant and cuisine type. As a result, hygienic establishments in the training data overrepresent the *same* restaurants and cuisines, as illustrated in Figure 2. As can be seen from that Figure, 72% of training observations represent multiple inspections for the same establishment (left panel), which would be expected for only 12% of training observations under random sampling[7] (right panel illustrates one random sample instance). This is problematic because terms unique to a restaurant (e.g., "café", "Chipotle") may result in overfitting. To continue the educational analogy, it would be akin to predicting grades for each school year, but rather than randomly sampling training data, repeatedly selecting the same students observed over multiple school years.

Second, instead of implementing methods to account for class imbalance using undersampling [15] or oversampling [8], the study reported accuracy only on the extreme balanced sample of 612 observations. By not reporting results on the full held-out dataset, results were likely overly optimistic, particularly in light of non-random sampling of hygienic restaurants noted above. Jointly, we refer to this problem as one of "extreme imbalanced sampling," and consider two minor modifications to illustrate the consequences of these methodological choices.

Random Sample of Hygienic Establishments: First, we consider the consequence of over-representing the same restaurants in the hygienic training sample. We randomly sample 306 hygienic restaurants (i.e. restaurants with scores of 0) and fit the same SVM models. We report mean accuracy results across 5 different datasets created with a random sample of hygienic restaurants. Contrary to the prior analysis, we find that inspection history has *higher* predictive performance than Yelp review features as shown in the bottom left panel of Figure 3. Indeed, this simple modification reduces the predictive power of Yelp review substantially: unigrams and bigrams drop from 83% reported accuracy to 69% reported accuracy, compared to 71% accuracy of inspection history, suggesting that the original results were due to overfitting.

Full Imbalanced Dataset: Next, we consider the consequence of not taking into consideration the imbalanced nature of this dataset by only reporting results on the sample of 612 inspections. Instead, we evaluate predictive performance on the full population of 13k inspections (via 10-fold cross-validation), accounting for class imbalance [30]. We implement undersampling on the training data, and report class averaged accuracy results on the test data.[8] The lower right panel of Figure 3 illustrates results. Inspection history alone achieves higher relative performance (67% class-averaged accuracy) than all other features. The (class-averaged) accuracy for unigrams and bigrams drops to 62%. Moreover, the overall predictive performance is substantially lower than what previous results suggested.

[4] As described in the original paper [33]: "For restaurants with "hygienic" labels, we only consider those without violation, as there are enough number of such restaurants to keep balanced distribution between two classes".
[5] See Table 1 in [33].
[6] There is one observation with a score of -1 that is treated as an inspection score of 0.
[7] We simulate random draws of hygienic restaurants and compute the mean number of duplicates across 10k draws.
[8] The class averaged accuracy represents a class-specific measure of accuracy calculated as $:= \frac{1}{2} \cdot \left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP} \right)$, where TP denotes true-positives, FN denotes false-negatives, TN denotes true-negatives, and FP denotes false-positives. Note that in settings with exact (or approximately) class balance that accuracy is exactly (or approximately) the same as class averaged accuracy.

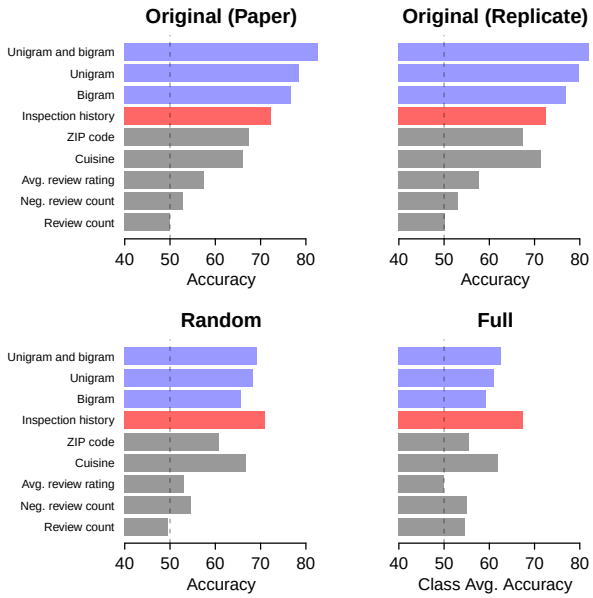


Figure 3: Predictive performance of Yelp review words (blue) and inspection history (red) in prior analysis (top left), replication (top right), random sample of hygienic restaurants (bottom left), and the full imbalanced dataset (bottom right). The figure shows that the relative performance of Yelp reviews depends entirely on a non-random and unrepresentative sample. The vertical dashed line represents baseline accuracy of 50%.

The fact that these two minor modifications reverse the core claim of the original paper is sobering. Higher levels of accuracy with predictive targeting could be achieved exclusively with data already possessed by the health department. The bottom panels of Figure 3 also show that predictive performance degrades when the hygienic class is chosen in random order and/or the full dataset is evaluated.

4 A RE-ANALYSIS

We now re-analyze this dataset from a regression perspective, to more closely align with the actual prediction problem of modeling inspection scores (without discarding data), and offer an alternative interpretation of the relative usefulness of inspection features versus online Yelp features. We view this re-analysis as a step forward in understanding the ability to predict an establishment's future inspection performance.

4.1 Data, Features, and Model

First, we reduce the dataset to routine (unannounced) inspections, by matching the Seattle Restaurant Dataset to data from the public health department. This removes (a) return inspections, which are triggered by poor performance in routine inspections, and typically occur a month after the routine inspection, and (b) educational inspections, which are unscored.[9] This leaves us with 12,613 observations.

Second, for comparability, we use largely the same feature set. Minor modifications are that we (a) construct cuisine features that indicate if a restaurant is denoted by a particular cuisine category, (b) do not normalize inspection scores,[10] (c) use the raw average rating score as opposed to further processing this score, and (d) use the raw previous penalty score instead of discretizing scores. The more substantial change is that we address sparseness of Yelp review text by leveraging word embedding `doc2vec` [37], instead of raw unigrams and bigrams. We apply a standard text cleaning process[11] that converts text to lower case and tokenizes all words. Results for embeddings created using the `doc2vec` parameters `vector_size=100`, `window=5`, and `min_count=3` are reported for the remainder of this paper.[12]

Third, rather than discarding information in inspection scores by discretization at an arbitrary cutoff, we evaluate each feature's prediction performance in a regression set-up to predict the full inspection score. We evaluate a Random Forest (RF) model [7], which is suitable for capturing higher-order interactions.[13] We report results across a 10-fold cross-validation set-up, where we select hyperparameters via 3-fold cross-validation on each training data portion.[14] Model parameters are selected via a randomized search for quicker run-time with textual features,[15] but we note comparable model performances between random versus grid search implementation on all the other features.[16]

4.2 Evaluation of Inspection, Restaurant, and Yelp-derived Features

Consistent with the earlier study, Figure 4 presents the predictive performance for each of the features individually. The baseline prediction is the mean inspection score from training observations used to predict on testing observations. The dashed vertical line represents the mean performance across the 10-folds for this baseline classifier. For each of the features, we plot densities of mean-squared-error (MSE) across the 10 folds in gray, with the vertical dash representing mean MSE. Figure 4 shows that inspection history achieves the lowest MSE (i.e., the highest predictive performance). Yelp features outside of the review text (review count, non-positive review count, average review reading) perform poorly.

[9] There are 2 observations in the original dataset that do not have corresponding inspection IDs in our dataset, which we drop. We also set one observation with an inspection score of -1 to 0.

[10] Inspection history features here will include the following fields provided in the original dataset: `inspection_average_prev_penalty_scores` and `inspection_prev_penalty_score`.

[11] In the `gensim` library in Python, we use `gensim.utils.simple_preprocess()`.

[12] We vary the hyperparameter settings when training a `doc2vec` model and observe comparable prediction performance across the range of hyperparameters considered. See [35] for more technical details on hyperparameter parameters in `doc2vec`.

[13] Technically inspection penalty scores are integers but RF computes averages in subsamples of the model resulting in fractional numbers. We evaluate the raw predictions from RF and avoid rounding.

[14] We consider the following search range for each hyperparameter parameter: `max_features = [None, 0.25, 0.5, 0.75]`, `max_depth = [None, 5, 10]`, `min_samples_leaf = [0.0005, 0.01, 0.05, 0.1]`, `min_samples_split = [2, 5, 10]`, and `n_estimators = [100, 200, 500]`

[15] http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

[16] http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html



Figure 4: Predictive performance of restaurant-, inspection- or Yelp-derived features as measured by mean squared error (MSE) in regression problem. Results are based on 10-fold cross-validation set-up (presented as densities, with means in vertical lines), fitting a RF regression model with hyperparameter tuning. Yelp review (blue) indicates word embedding representation of review content.

We observe only modest performance of Yelp reviews, providing less predictive power than inspection history and about the same as ZIP code.

4.3 Marginal Value of Yelp Features

While Figure 4 considered features individually, we now compare the performance of a "full" model consisting of inspection-, restaurant-, and Yelp-derived features, versus a "non-Yelp" model to evaluate the marginal value of Yelp. A full model with all features considered in Figure 4 yields a mean MSE of 185.70, while a non-Yelp model (excluding Yelp review, ratings, and review counts) yields a comparable mean MSE of 183.61, with a non-significant p-value of 0.984 from a paired Wilcoxon signed rank one-sided test for whether the additional Yelp features improve the fit. These results indicate that Yelp features do not significantly improve predictions, given existing information available about restaurants and inspection history. Table 2 illustrates the marginal value of Yelp reviews given different feature sets. Yelp adds predictive value only in instances where the features are extremely limited (e.g., only ZIP code). The bottom four rows show that with any realistic feature set (e.g., inspection history and cuisine), Yelp reviews do not statistically (and substantively) improve predictive performance.

4.4 Limitations

We note several limitations to our analysis. First, as in the prior analysis, our review information is aggregated. It may be important to treat each user's review as a separate block of text instead of a concatenated text field across the inspection period. That said, review information is already exceedingly sparse, so accounting for specific reviews may be challenging. Second, it is possible that more refined feature extraction from text (e.g., sentence encoding) would improve the predictive performance of Yelp reviews or that some alternative feature representation of the review text would be useful.[17] The prior analysis used raw unigrams and bigrams, which we also found lacking in predictive power. Third, while we have followed the original setup of evaluating performance by cross-validation, a more appropriate evaluation should take into account that the use case involves forecasting *future* sanitation [32]. Understanding how algorithms generalize temporally has been raised in other contexts such as inferring economic conditions in a country [6]. In our setting, a more appropriate model may involve early detection prediction to trigger an early inspection [55]. Fourth, alternative datasets or features, such as the length of time a restaurant has been open, might provide a better signal of a restaurant's future hygienic conditions.

Table 2: This table evaluates the additional predictive value of the Yelp review text given different subsets of other features such as ZIP code, cuisine, or inspection history. We observe that Yelp's additional predictive value is minimal. We report p-values from a paired Wilcoxon signed rank test, testing whether the difference in MSE without versus with the Yelp text features across the same test folds is greater than 0. */** indicate statistical significance at 0.05 and 0.01.

Features included			Mean Squared Error		
ZIP	Cuisine	Inspection history	Without Yelp review text	With Yelp review text	p-value
			222.13	209.93	0.003**
✓			211.30	203.27	0.003**
	✓		198.50	204.13	0.996
		✓	192.00	189.12	0.004**
✓	✓		188.44	199.02	0.998
✓		✓	188.65	187.49	0.042*
	✓	✓	187.91	188.65	0.762
✓	✓	✓	183.61	185.61	0.988

5 DISCUSSION

While web and social media data offer tantalizing possibilities for governance, we have shown that one of the most celebrated cases for "reinventing food safety" overstates the utility of Yelp reviews. While this may be no fault of the authors – indeed the original authors generously offered replication code, data, and comments to help assess the methodology – the popular susceptibility to "big data hubris" has considerable ethical and policy implications. As mentioned above, health departments in Boston, Las Vegas, Chicago, and New York City each adopted techniques for targeting based on Yelp or Twitter, and much more serious work needs to be conducted to validate and assess these methods for policy adoption. We here highlight three particular implications.

5.1 Replication and Transparency

Our study corroborates that the replication crisis that has affected the social sciences and biomedicine [10, 29] may also affect the machine learning community [15, 39]. Only 6% of papers at top artificial intelligence conferences made code available [22]. As data and methods become more complex, unarticulated researcher "degrees of freedom" expand. Here, such discretionary choices included: the

[17] For observations on the consequences of pre-processing for topic modeling, see [13].

feature set, feature representation (e.g., TF-IDF weighting of word terms), data cleaning, software version, hyperparameter tuning (the kernel, C, and γ in the original SVM setup), the penalty threshold, and selection of majority class from an ordered dataset. The original authors deserve much credit to providing the replication data. But even with the exact data in hand, we were unable to replicate the findings after three months of efforts. Only after the code was generously shared did it become clear that classification was limited to ≈600 restaurants with extreme imbalanced and non-random sampling. Moreover, while we were able to perform a "direct replication," this uncovered substantial methodological problems.

Our evidence hence underscores recent calls for more general reproducibility (or "conceptual replication") in machine learning [15, 20, 38]. More generally, our findings highlight the importance of recent projects in machine learning to promote transparency and replication , such as including code with conference submissions and details on the trained model [43], automating hyperparameter searching [4], developing open source software and computational infrastructure for reproducibility [41, 49], and providing datasheets for detailed dataset information [17].

We also note that another form of transparency is relevant in the law and policy context, namely surrounding the role of private industry sponsorship. When private data is used for public purposes, a greater obligation of transparency may attach. Yelp, for instance, sponsored a predictive tournament for health scores in Boston [19], posing a potential conflict of interest. Recall Yelp's CEO statement that Yelp could "beat gold standard healthcare measures." Disclosure both of sponsorship, data, and methodology would help ensure that these findings can be externally validated when private data enters public enforcement.

5.2 Machine and Policy Translation

Our results also show that much work remains to be done for machine learning to be translated into policy. First, notwithstanding wide media coverage of how Yelp could solve food safety, we show that the prediction is much more difficult than previously appreciated. As Table 2 shows, predictive power of *any* features is not very high. By definition, health authorities already retain rich administrative data on inspection history and restaurant meta-information, and Table 2 confirms that with any reasonable baseline set of features, Yelp reviews add no statistically significant predictive power.

Second, machine learning scholars need to understand the substantive policy setting of a prediction problem. For instance, when the prediction problem is defined as a classification problem, cutoff points should be policy-relevant. The 50-point cutoff, predicting only the most extreme 2.3% of the sample, does not accord to any meaningful quantity in Seattle. As a policy matter, Seattle (and King County) mandate (i) a return inspection when more than 35 points for critical violations are assessed and (ii) a shutdown of the restaurant when more than 90 critical violation points are assessed. The prior analysis neither distinguished between critical and non-critical violations nor paid attention to these substantive thresholds. Converting a regression to classification problem must be done with care and attention to the substantive setting [28], particularly when discretization can discard meaningful information.

Third, applying this technique in policy has to grapple with the fact that Yelp reviews reflect the digital divide. In Seattle and King County, for instance, Yelp has a much higher penetration in areas with lower minority presence, higher educational attainment, and lower unemployment rates [26]. When reviews themselves are of an unrepresentative sample of restaurants, there are considerable ethical concerns about the distributive implications for adopting food safety enforcement based on such data. For a similar example, Boston's StreetBump application, which used smartphone accelerometer and GPS data to trigger service requests for potholes, initially deployed more public resources to wealthier areas where smartphone penetration was higher [11]. None of these concerns are explored in work proposing to use social media to target inspections and thus, these concerns warrant much greater attention.

Last, scholars may also need to contemplate what "predictive accuracy" means in a given context. One of the predominant substantive concerns in food safety enforcement has been that inspection stringency may be largely driven by the differences across inspectors [24, 25]. If so, targeting based on predictive scores may contravene rather than support policy goals: more inspection resources would simply be deployed to where inspections are already the most stringent [26].

5.3 Class Imbalance

For classification problems, class imbalance is a common concern [45, 54]. The prior study exacerbated this concern by engaging in extreme imbalanced sampling. But the fact that accuracy statistics were only presented on the sample of ≈600 restaurants may illustrate a more pervasive challenge in machine learning: while it may be desirable to train a classifier on a balanced sample (e.g., via undersampling or oversampling), class imbalance in the population must be accounted for when assessing model performance. This can clearly be seen in the bottom two panels of Figure 3, where class-averaged accuracy from the population appears much worse than the originally reported 82% accuracy statistic. This concern is particularly acute in the policy setting. Food safety predictions, for instance, necessarily must be conducted on the population of restaurants.

6 CONCLUSION

We have shown in this work that one of the most widely cited examples of machine learning with social media for public policy suffers from considerable concerns about replicability. We close by noting that the original authors are not to be blamed: they advanced a novel hypothesis and amassed new data to test it. Given the large number of researcher degrees of freedom, any researcher may inadvertently introduce methodological problems in one of hundreds of lines of code. Yet the case also illustrates the profound challenges of the replication crisis given researcher degrees of freedom in machine learning. We believe that the ethical concerns with such applications are compounded when applied to law and public policy, heightening the importance of recent calls for improving the replication, transparency, and validation of methods in the machine learning community.

Code Availability Statement: For access to the original code for replicating the original results in [33], please contact Jun Kang at junkang@cs.stonybrook.edu. All other code for the analysis presented in this paper is available at: `https://github.com/kaltenburger/YelpReplicate`

Acknowledgements: Altenburger was supported in part by the National Defense Science and Engineering Graduate Fellowship program. We thank Kang et al. for sharing their original code, and Yejin Choi, Alex Chin, Tristan Cossio, Dirk Hovy, Eun Seo Jo, Philip Kegelmeyer, Erin LeDell, Michael Luca, Anne McDonough, Oluchi Mbonu, Stephen Ragain, Arjun Seshadri, Sarah Shugars, George Pakapol Supaniratisai, Angela Teuscher, Johan Ugander, and anonymous reviewers for helpful comments.

REFERENCES

[1] Laura Adler. 2016. Learning from Location. *Data-Smart City Solutions* (2016).
[2] Kristen M Altenburger and Daniel E Ho. 2018. When Algorithms Import Private Bias into Public Enforcement: The Promise and Limitations of Statistical Debiasing Solutions. *Journal of Institutional and Theoretical Economics* (2018).
[3] Emily Badger. 2013. How Yelp Might Clean Up the Restaurant Industry. *Atlantic* (2013).
[4] James Bergstra, Daniel Yamins, and David Daniel Cox. 2013. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. (2013).
[5] Joshua Blumenstock, Gabriel Cadamuro, and Robert On. 2015. Predicting poverty and wealth from mobile phone metadata. *Science* 350, 6264 (2015), 1073–1076.
[6] Joshua E Blumenstock. 2018. Estimating Economic Characteristics with Phone Data. In *AEA Papers and Proceedings*, Vol. 108. 72–76.
[7] Leo Breiman. 2001. Random forests. *Machine Learning* 45, 1 (2001), 5–32.
[8] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* 16 (2002), 321–357.
[9] Raviv Cohen and Derek Ruths. 2013. Classifying political orientation on Twitter: It's not easy!. In *ICWSM*.
[10] Open Science Collaboration et al. 2015. Estimating the reproducibility of psychological science. *Science* 349, 6251 (2015), aac4716.
[11] Kate Crawford. 2013. The hidden biases in big data. *Harvard Business Review* 1 (2013).
[12] Munmun De Choudhury, Michael Gamon, Scott Counts, and Eric Horvitz. 2013. Predicting depression via social media. *ICWSM* 13 (2013), 1–10.
[13] Matthew J Denny and Arthur Spirling. 2018. Text preprocessing for unsupervised learning: why it matters, when it misleads, and what to do about it. *Political Analysis* 26, 2 (2018), 168–189.
[14] Katelynn Devinney, Adile Bekbay, Thomas Effland, Luis Gravano, David Howell, Daniel Hsu, Daniel O'Hallorhan, Vasudha Reddy, Faina Stavinsky, HaeNa Waechter, et al. 2018. Evaluating Twitter for Foodborne Illness Outbreak Detection in New York City. *Online Journal of Public Health Informatics* 10, 1 (2018).
[15] Chris Drummond, Robert C Holte, et al. 2003. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on Learning from Imbalanced Datasets II*, Vol. 11. Citeseer, 1–8.
[16] Daniel Gayo-Avello. 2013. A meta-analysis of state-of-the-art electoral prediction from Twitter data. *Social Science Computer Review* 31, 6 (2013), 649–679.
[17] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumeé III, and Kate Crawford. 2018. Datasheets for Datasets. *arXiv preprint arXiv:1803.09010* (2018).
[18] Jeremy Ginsberg, Matthew H Mohebbi, Rajan S Patel, Lynnette Brammer, Mark S Smolinski, and Larry Brilliant. 2009. Detecting influenza epidemics using search engine query data. *Nature* 457, 7232 (2009), 1012.
[19] Edward L. Glaeser, Andrew Hillis, Scott Duke Kominers, and Michael Luca. 2016. Crowdsourcing City Government: Using Tournaments to Improve Inspection Accuracy. *American Economic Review* 106, 5 (May 2016), 114–18.
[20] Steven N Goodman, Daniele Fanelli, and John PA Ioannidis. 2016. What does research reproducibility mean? *Science Translational Medicine* 8, 341 (2016), 341ps12–341ps12.
[21] Verena Grubmüller, Katharina Götsch, and Bernhard Krieger. 2013. Social media analytics for future oriented policy making. *European Journal of Futures Research* 1, 1 (26 Sep 2013), 1–20.
[22] Odd Erik Gundersen and Sigbjørn Kjensmo. 2017. State of the art: Reproducibility in artificial intelligence. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence and the Twenty-Eighth Innovative Applications of Artificial Intelligence Conference*.
[23] Cassandra Harrison, Mohip Jorder, Henri Stern, Faina Stavinsky, Vasudha Reddy, Heather Hanson, H Waechter, Luther Lowe, Luis Gravano, Sharon Balter, et al. 2014. Using online reviews by restaurant patrons to identify unreported cases of foodborne illness – New York City, 2012–2013. *Morbidity and Mortality Weekly Report* 63, 20 (2014), 441–445.
[24] Daniel E. Ho. 2012. Fudging the Nudge: Information Disclosure and Restaurant Grading. *Yale Law Journal* 122, 3 (2012), 574–688.
[25] Daniel E. Ho. 2017. Does Peer Review Work: An Experiment of Experimentalism. *Stanford Law Review* 69 (2017), 1–119.
[26] Daniel E Ho. 2017. Equity in the Bureaucracy. *Irvine Law Review* 7 (2017), 401–451.
[27] Matthew Hutson. 2018. Artificial intelligence faces reproducibility crisis.
[28] Nitin Indurkhya and Sholom M Weiss. 2001. Solving regression problems with rule-based ensemble classifiers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 287–292.
[29] John PA Ioannidis. 2005. Why most published research findings are false. *PLoS Medicine* 2, 8 (2005), e124.
[30] Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intelligent Data Analysis* 6, 5 (2002), 429–449.
[31] Neal Jean, Marshall Burke, Michael Xie, W Matthew Davis, David B Lobell, and Stefano Ermon. 2016. Combining satellite imagery and machine learning to predict poverty. *Science* 353, 6301 (2016), 790–794.
[32] Sham Kakade, Percy Liang, Vatsal Sharan, and Gregory Valiant. 2016. Prediction with a short memory. *arXiv preprint arXiv:1612.02526* (2016).
[33] Jun Seok Kang, Polina Kuznetsova, Michael Luca, and Yejin Choi. 2013. Where not to eat? improving public policy by predicting hygiene inspections using online reviews. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1443–1448.
[34] Jon Kleinberg, Jens Ludwig, Sendhil Mullainathan, and Ziad Obermeyer. 2015. Prediction policy problems. *American Economic Review* 105, 5 (2015), 491–95.
[35] Jey Han Lau and Timothy Baldwin. 2016. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368* (2016).
[36] David Lazer, Ryan Kennedy, Gary King, and Alessandro Vespignani. 2014. The parable of Google Flu: traps in big data analysis. *Science* 343, 6176 (2014), 1203–1205.
[37] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*. 1188–1196.
[38] Jeffrey T Leek and Roger D Peng. 2015. Opinion: Reproducible research can still be wrong: Adopting a prevention approach. *Proceedings of the National Academy of Sciences* 112, 6 (2015), 1645–1646.
[39] Zachary C Lipton and Jacob Steinhardt. 2018. Troubling trends in machine learning scholarship. *arXiv preprint arXiv:1807.03341* (2018).
[40] Yelena Mejova, Ingmar Weber, and Michael W Macy. 2015. *Twitter: a digital socioscope*. Cambridge University Press.
[41] Jill P Mesirov. 2010. Accessible reproducible research. *Science* 327, 5964 (2010), 415–416.
[42] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[43] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model Cards for Model Reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. ACM, 220–229.
[44] Elaine O Nsoesie, Sheryl A Kluberg, and John S Brownstein. 2014. Online reports of foodborne illness capture foods implicated in official foodborne outbreak reports. *Preventive Medicine* 67 (2014), 264–269.
[45] Rachel A Oldroyd, Michelle A Morris, and Mark Birkin. 2018. Identifying Methods for Monitoring Foodborne Illness: Review of Existing Public Health Surveillance Techniques. *JMIR Public Health and Surveillance* 4, 2 (2018), e57.
[46] Adam Sadilek, Sean Brennan, Henry Kautz, and Vincent Silenzio. 2013. nEmesis: which restaurants should you avoid today?. In *First AAAI Conference on Human Computation and Crowdsourcing*.
[47] John P. Schomberg, Oliver L. Haimson, Gillian R. Hayes, and Hoda Anton-Culver. 2016. Supplementing Public Health Inspection via Social Media. *PLOS ONE* 11, 3 (03 2016), 1–21.
[48] Somini Sengupta. 2013. In Hot Pursuit of Numbers to Ward Off Crime. *New York Times* (2013).
[49] Sören Sonnenburg, Mikio L Braun, Cheng Soon Ong, Samy Bengio, Leon Bottou, Geoffrey Holmes, Yann LeCun, Fernando Pereira, and Carl Edward Rasmussen. 2007. The Need for Open Source Software in Machine Learning. *Journal of Machine Learning Research* 8 (2007), 2443–2466.
[50] Jeremy Stoppelman. 2016. Yelp CEO says online reviews could beat "gold standard" healthcare measures. *Modern Healthcare* (2016).
[51] Tim Thompson. 2015. How Our Cities Are Using Social Data. *IBM Big Data & Analytics Hub* (2015).
[52] Zeynep Tufekci. 2014. Big Questions for Social Media Big Data: Representativeness, Validity and Other Methodological Pitfalls. *ICWSM* 14 (2014), 505–514.
[53] Emilio Zagheni, Kivan Polimis, Monica Alexander, Ingmar Weber, and Francesco C Billari. 2018. Combining Social Media Data and Traditional Surveys to Nowcast Migration Stocks. (2018).

[54] Jiaming Zeng, Berk Ustun, and Cynthia Rudin. 2017. Interpretable classification models for recidivism prediction. *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 180, 3 (2017), 689–722.

[55] Justine Zhang, Jonathan P Chang, Cristian Danescu-Niculescu-Mizil, Lucas Dixon, Yiqing Hua, Nithum Thain, and Dario Taraborelli. 2018. Conversations Gone Awry: Detecting Early Signs of Conversational Failure. *arXiv preprint arXiv:1805.05345* (2018).

Bi-LSTM-CRF Sequence Labeling for Keyphrase Extraction from Scholarly Documents

Rabah A. Al-Zaidy
Pennsylvania State University
alzaidy@psu.edu

Cornelia Caragea
University of Illinois at Chicago
cornelia@uic.edu

C. Lee Giles
Pennsylvania State University
giles@ist.psu.edu

ABSTRACT

In this paper, we address the keyphrase extraction problem as sequence labeling and propose a model that jointly exploits the complementary strengths of Conditional Random Fields that capture label dependencies through a transition parameter matrix consisting of the transition probabilities from one label to the neighboring label, and Bidirectional Long Short Term Memory networks that capture hidden semantics in text through the long distance dependencies. Our results on three datasets of scholarly documents show that the proposed model substantially outperforms strong baselines and previous approaches for keyphrase extraction.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Information extraction**.

KEYWORDS

Keyphrase extraction, sequence labeling, deep learning.

ACM Reference Format:
Rabah A. Al-Zaidy, Cornelia Caragea, and C. Lee Giles. 2019. Bi-LSTM-CRF Sequence Labeling for Keyphrase Extraction from Scholarly Documents. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313642

1 INTRODUCTION

Keyphrase extraction is a key natural language processing task aimed at automatically extracting descriptive phrases or words from a document [17]. Keyphrases (also referred as keywords) provide a brief summary of the content of a document. The importance of keyphrases has been widely recognized in many downstream applications such as query formulation, document clustering, classification, recommendation, indexing, and summarization [16, 21, 38].

Most of the existing works on automatic keyphrase extraction focus on supervised and unsupervised approaches. The unsupervised approaches use ranking techniques to rank phrases based on the aggregated "informativeness" scores of the individual words comprising a phrase. Graph-based ranking algorithms that are applied to the word graph representation of a document are the most prevalent in this category. To construct the graph, each candidate word in a document (i.e., a word with certain part-of-speech tags) is mapped to a node and connecting edges represent the association patterns among the candidate words. The scores of individual words are estimated using various graph centrality measures such as PageRank [10, 12, 27, 34, 41].

The supervised approaches use binary classification to label candidate phrases positively (as keyphrases) or negatively (as non-keyphrases), based on a set of linguistic and statistical features such as *tf-idf*, part of speech (POS) tags, and the position of phrases in documents. Supervised keyphrase extraction allows for expressive feature design and is reported to often outperform unsupervised methods [6, 23]. Two major limitations of these supervised approaches are: (1) they classify the labels of each candidate phrase independently, while completely ignoring the dependencies that could potentially exist between neighboring labels; and (2) they do not incorporate the hidden semantics in the input text.

More recently, Gollapalli et al. [13] formulated keyphrase extraction as sequence labeling and showed that using linear-chain Conditional Random Fields can improve the performance over baseline models for this task. However, the approach in [13] does not explicitly take into account the long-term dependencies and semantic relationships hidden in text. Figure 1 shows examples of long-term dependency patterns and semantic relationships hidden in text from a research paper published in the ACM KDD conference, e.g., the phrase "we describe" is followed by (and indicative of) keyphrases, whereas the terms "Conditional Random Field" and "discriminatively-trained model" are semantically related. We posit that a deep understanding of the text is required in order to correctly identify keyphrases for a document.

To this end, we address keyphrase extraction as a sequence labeling problem, using *research papers* as a case study, and precisely aim to capture both the semantics of document contexts as well as the dependencies among the labels of neighboring words in order to overcome the limitations in previous approaches. Specifically, we explore a neural learning model, called Bi-LSTM-CRF, that combines a bi-directional Long Short-Term Memory (Bi-LSTM) layer to model the sequential text data with a Conditional Random Field (CRF) layer to model dependencies in the output [19, 30]. The result of this extraction task will aid indexing of documents in scholarly document collections, and hence, will lead to improved organization, search, retrieval, and recommendation of scientific documents. In summary, our contributions are as follows:

- We explore a neural learning model for keyphrase extraction from scholarly documents that combines the complementary strengths of Bi-LSTM and CRF. In the combined model, the input and output layers are not directly connected as in CRF, but instead a Bi-LSTM layer is inserted between them to exploit the long term dependencies in the text.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313642

A Unified Approach for **Schema Matching**, **Coreference** and **Canonicalization**
by Michael L. Wick, Khashayar Rohanimanesh, Karl Schultz and Andrew McCallum

The automatic consolidation of database records from many heterogeneous sources into a single repository requires solving several *information integration* tasks. Although tasks such as **coreference**, **schema matching**, and **canonicalization** are closely related, they are most commonly studied in isolation. Systems that do tackle *multiple integration problems* traditionally solve each independently, allowing errors to propagate from one task to another. *In this paper, we describe* a *discriminatively-trained model* that reasons about **schema matching**, **coreference**, and **canonicalization** jointly. [...]

Author-annotated keyphrases: *Data Integration, Coreference, Schema Matching, Canonicalization, Conditional Random Field, Weighted Logic*

Figure 1: The title and abstract of a paper by Wick et al. (2008) and the author-input keyphrases for the paper. Cyan bold phrases represent the gold-standard (author-annotated) keyphrases for the document. Red italic phrases represent long-term dependency patterns or semantic relationships with the gold-standard keyphrases.

- We conduct a thorough evaluation to examine the role of each layer in the model. To our knowledge, there is no study that demonstrates an ablation experiment that compares, with this clarity, the role of input dependencies and label dependencies in keyphrase extraction from research papers.
- We show empirically on three datasets of research papers that the Bi-LSTM-CRF model outperforms strong baselines and previous works on the keyphrase extraction task.
- We investigate the performance of Bi-LSTM-CRF at document and sentence level and show that a document level model that captures a broader context is more accurate than a sentence level model. To our knowledge, this has not been addressed before, especially, not on a large scale dataset as we do in this paper.

In the next section, we describe related work on keyphrase extraction. We then introduce the neural learning model in Section 3, followed by Section 4 that presents our evaluation setup. Finally, we discuss our experimental results in Section 5 before we conclude the paper and touch on future directions in Section 6.

2 RELATED WORK

Keyphrase extraction has been the focus of many studies. These studies generally adopt a two phase approach. In the first phase, candidate words or phrases are extracted from the text using heuristics such as POS patterns for words or *n*-grams [20]. In the second phase, the candidate phrases are predicted as keyphrases or non-keyphrases, using both supervised and unsupervised approaches. In the supervised approaches, the prediction is done based on a selection of features, e.g., POS tags, *tf-idf* scores, and position information, used in conjunction with machine learning classifiers [11, 20, 36, 39]. Traditional features were also combined with features extracted from external sources such as WordNet and Wikipedia [29, 32] or from various neighborhoods, e.g., a document's citation network or a webpage's hyper-link network [5, 6, 22].

Unsupervised approaches include phrase scoring methods based on measures such as *tf-idf* and topic proportions [3, 28, 46], graph-based ranking using centrality measures [15, 34, 41], and keyphrase selection from topics detected using topic modeling [25, 40]. In this context, several extensions of PageRank and personalized PageRank have been proposed that make use of a document's citation network [12] or that bias the random walk based on the words' positions in text [10] or the words' topic distribution [27]. In order to add semantic relatedness between the words in a word graph, Martinez-Romo et al. [31] used information from WordNet.

The best performing SemEval 2010 system used term frequency thresholds to filter out phrases that are unlikely to be keyphrases, where the thresholds were estimated from the data [9]. The candidate phrases were ranked using the *tf-idf* model in conjunction with a boosting factor which aims at reducing the bias towards single word terms. Danesh et al. [8] computed an initial weight for each phrase based on a combination of the *tf-idf* score and the first position of a phrase in a document. Phrases and their initial weights were then incorporated into a graph-based algorithm, which produces the final ranking of keyphrases. Adar and Datta [1] extracted keyphrases by mining abbreviations from scientific literature and built a semantic hierarchical keyphrase database. Many of the above approaches, both supervised and unsupervised, are compared and analyzed in a survey by Hasan and Ng [17].

Neural networks have started to be incorporated into models for keyphrase extraction. For example, Wang et al. [42] investigate word embeddings to measure the relatedness between words in graph-based models. A Recurrent Neural Network (RNN) based approach is proposed by Zhang et al. [45] to identify keyphrases in Twitter data. The model addresses the problem as sequence labeling for very short text, where a joint-layer RNN is used to capture the semantic dependencies in the input sequence, but does not address the dependencies in the labels. In our work, we capture the flow of scientific writing and the dependencies between keyphrases and the other words in the text that may not necessarily exist between hashtags and the text of the tweets. Augenstein and Søgaard [2] treated keyphrase extraction as a multi-task learning problem and applied RNNs to classify keyphrase boundaries. Inspired from work in machine translation, Meng et al. [33] focused on keyphrase generation (rather than keyphrase extraction) and addressed it as a sequence to sequence learning problem with a copying mechanism, where the sequence of words in a document is used to generate a sequence of keyphrases. An Encoder-Decoder RNN, originally proposed by Cho et al. [7], was used to generate the keyphrase sequences. A variation of the model, which performs better than the Encoder-Decoder RNN, includes a copying mechanism to identify keyphrases that occur rarely in the text. Unlike Meng et al. [33], we focus on keyphrase extraction, i.e., extracting only words that are present in text, and not keyphrase generation, which outputs words that may or may not be present in text. We use the Encoder-Decoder RNN with the copying mechanism as one of our baselines.

Sequence labeling models for keyphrase extraction have shown promising results in recent studies [4, 13, 44]. For example, Gollapalli et al. [13] trained Conditional Random Fields (CRFs) to extract keyphrases from scholarly documents, using features such as *tf-idf*

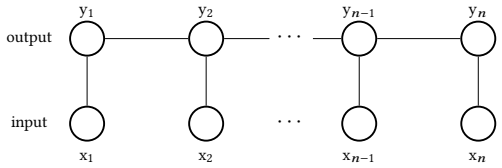


Figure 2: Layers in a CRF network.

and POS tags to predict a label for each token position in a document as being a keyphrase token (**KP**) or not (**Non-KP**). This CRF model is able to capture the dependencies in previous and future tags in the label sequence, however, the semantic dependencies in the input sequence, i.e., the text, are not incorporated.

Recently, a sequence labeling framework that takes into consideration both types of dependencies has been explored on a variety of NLP tasks such as part-of-speech tagging, noun phrase chunking, and named entity recognition [19, 26, 30]. This framework combines a bidirectional LSTM (Bi-LSTM) network as the first layer to capture sequential text dependencies with a second CRF layer to capture label dependencies. In our work, we identify the limitations of each independent model (Bi-LSTM and CRF) and explore their combination on keyphrase extraction from scholarly documents. To our knowledge, in the context of keyphrase extraction from scholarly documents, we are the first to simultaneously capture the semantics of document contexts and long-range dependencies in text by modeling the sequential text data as well as the dependencies among the labels of neighboring words.

3 METHODOLOGY

In this section, we formulate keyphrase extraction as a sequence labeling task and present the details of the Bi-LSTM-CRF model and its constituent components, CRF and Bi-LSTM.

3.1 Problem Formulation

Keyphrase extraction can be formulated as a sequence labeling task as follows: Given an input sequence $\mathbf{x} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$, where each $\mathbf{x}_i$ represents the input vector of the i^{th} word, predict a sequence of labels $\mathbf{y} = \{y_1, \cdots, y_n\}$, one label for each word in the input, where each label y_i is **KP** (keyphrase word) or **Non-KP** (not keyphrase word). The sequence labeling formulation of our task takes into account the correlations between neighboring labels and allows to *jointly* decode the best sequence of labels for the input sequence, rather than decoding each label independently.

3.2 Conditional Random Fields

Conditional Random Field (CRF) introduced by Lafferty et al. [24] has been successfully used in many sequence labeling tasks and we use it here to jointly model the sequence of labels for our keyphrase extraction task. The conditional probability distribution over the label sequence $\mathbf{y}$ given $\mathbf{x}$ defined by CRF has the form:

$$p(\mathbf{y}|\mathbf{x}; \mathbf{W}, \mathbf{b}) \propto \exp\left(\sum_{i=1}^{n} \mathbf{W}^T_{y_{i-1}, y_i} \mathbf{x}_i + \mathbf{b}_{y_{i-1}, y_i}\right)$$

where $\mathbf{W}_{y_{i-1}, y_i}$ and $\mathbf{b}_{y_{i-1}, y_i}$ are model parameters (weight vector and bias) corresponding to the neighboring labels (y_{i-1}, y_i).

For training a CRF model, we estimate model parameters $\mathbf{W}$ and $\mathbf{b}$ from a training dataset $\mathcal{D} = \{(\mathbf{x}^{(j)}, \mathbf{y}^{(j)})\}_{j=1}^{N}$ by maximizing the log-likelihood given by:

$$L(\mathbf{W}, \mathbf{b}) = \sum_{j=1}^{N} \log p(\mathbf{y}^{(j)}|\mathbf{x}^{(j)}; \mathbf{W}, \mathbf{b})$$

To find the best sequence path during decoding, the optimal sequence $\mathbf{y}$ that maximizes the likelihood, is computed using the Viterbi decoding:

$$\mathbf{y}^* = \text{argmax}_{y \in \mathcal{Y}(\mathbf{x})} p(\mathbf{y}|\mathbf{x}; \mathbf{W}, \mathbf{b})$$

Figure 2 depicts the layout of a simple CRF network. As shown in the figure, nodes in the output layer are connected, enabling the model to capture dependencies in the sequence of labels.

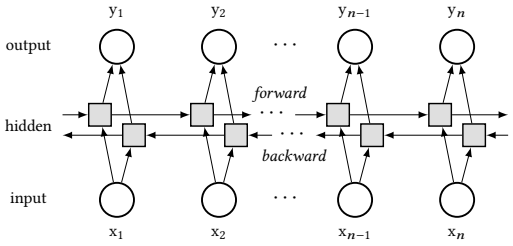


Figure 3: Layers in a Bi-LSTM network.

3.3 Bi-LSTM

Long Short Term Memory networks (LSTMs) are a special type of Recurrent Neural Networks (RNNs) and are designed to overcome the gradient vanishing problem of RNNs. In particular, LSTMs have additional memory cells, which store memory from long distance terms [18]. Because LSTMs are capable of preserving information over previous inputs of a sequence into the current input state, they have been a natural choice for applications involving temporal and sequence data such as speech recognition, language modeling and translation [14, 35]. The structure of an LSTM includes an input layer, a hidden layer and an output layer. An LSTM unit at time t consists of sub-unit-inputs ($\mathbf{i}_t$), output ($\mathbf{o}_t$), forget gates ($\mathbf{f}_t$) and memory cell ($\mathbf{c}_t$). The updates at time t are then:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}^{(i)}\mathbf{h}_{t-1} + \mathbf{U}^{(i)}\mathbf{x}_t + \mathbf{b}^{(i)}) \\
\mathbf{f}_t &= \sigma(\mathbf{W}^{(f)}\mathbf{h}_{t-1} + \mathbf{U}^{(f)}\mathbf{x}_t + \mathbf{b}^{(f)}) \\
\mathbf{o}_t &= \sigma(\mathbf{W}^{(o)}\mathbf{h}_{t-1} + \mathbf{U}^{(o)}\mathbf{x}_t + \mathbf{b}^{(o)}) \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}^{(c)}\mathbf{h}_{t-1} + \mathbf{U}^{(c)}\mathbf{x}_t + \mathbf{b}^{(c)}) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{(t-1)} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned}$$

Here, σ is the element-wise logistic sigmoid function and $\odot$ is the Hadamard product; $\mathbf{x}_t$ is the input vector at time t, i.e., the word embedding in our case, and $\mathbf{h}_t$ is the hidden state that stores sequential information up to time t. $\mathbf{W}$, $\mathbf{U}$, and $\mathbf{b}$ are model parameters to be estimated during training.

In a forward LSTM network, the hidden state $\mathbf{h}_t$ stores information only from the past. To capture the flow of information both ways, we use a bi-directional LSTM network, which consists of a forward hidden layer and a backward hidden layer. Figure 3 shows the structure of a Bi-LSTM network. Here, the nodes in the hidden layer are connected which is how long distance information is maintained in the matrix weights.

Table 1: Dataset statistics.

Statistics	Training	Validation	Testing		
	kp527k	**kp20k-v**	**kp20k**	**WWW**	**KDD**
Number of documents	527,830	20,000	20,000	1,330	755
Number of sentences	4,686,986	176,930	177,278	12,288	7,768
Number of tokens in total	78,441,075	2,948,609	2,971,668	200,704	132,728
Number of tokens in keyphrases	5,458,743	205,586	207,073	12,181	6,119
Number of keyphrases	2,806,381	106,181	105,523	6,405	3,093

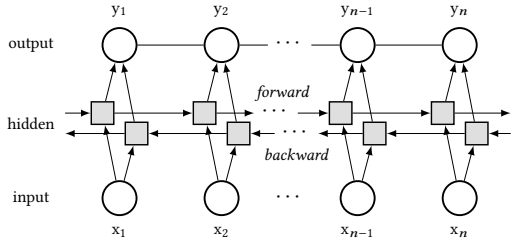


Figure 4: Layers in a Bi-LSTM-CRF network.

3.4 Bi-LSTM-CRF

In order to build a sequence labeling model that incorporates long distance information over a sequence of input as well as information on the output sequence, we combine a Bi-LSTM network with a CRF network. The network architecture is shown in Figure 4. As shown in the figure, the first layer of the model is a Bi-LSTM network with the purpose of capturing the semantics of the input text sequence. The output of the Bi-LSTM layer is passed to a CRF layer that produces a probability distribution over the tag sequence using the dependencies among labels of the entire sequence. In order to find the best sequence of labels for an input sequence, the Viterbi algorithm is used.

4 EVALUATION SETTING

To evaluate the performance of our proposed model, we conduct a wide range of experiments. In this section, we describe the datasets used for training and evaluation, discuss hyper-parameters, baselines and the evaluation measures.

4.1 Datasets

We use three different datasets of scientific documents for our evaluation purpose. The first dataset was made available by Meng et al. [33] and was collected by crawling the metadata of papers from several online digital libraries such as ACM, WebofScience, and Wiley. The dataset contains metadata for 567, 830 papers with a clear split as train, validation, and test sets provided by the authors, as follows: 527, 830 were used for model training, 20, 000 were used for parameter tuning, and the remaining 20, 000 were used for model evaluation. We refer to these sets as **kp527k**, **kp20k-validation** (or **kp20k-v**), and **kp20k-test** (or simply **kp20k**), respectively.

The second and third datasets were made available by Gollapalli and Caragea [12] and were compiled from the CiteSeerX digital library. These datasets contain metadata for research papers from the proceedings of two top-tier data mining and machine learning conferences, i.e., the ACM Conference on Knowledge Discovery and Data Mining (referred as **KDD**) and the World Wide Web Conference (referred as **WWW**).

The metadata of each paper from all three datasets above consist of titles, abstracts, and author-assigned keyphrases. The title and abstract of each paper are used to extract keyphrases, whereas the author-input keyphrases are used as gold-standard for evaluation. In experiments, for all models, we use **kp527k** for model training, **kp20k-validation** for parameter tuning, and **kp20k**, **KDD**, and **WWW** as three independent test sets for model evaluation. Note that we removed from **KDD** and **WWW** the entries that occur also in **kp527k-train** and **kp20k-validation** to avoid the overlap between train, validation, and test sets. Table **??** shows the statistics of these datasets.

4.2 Implementation Details

Since we use a sequence labeling formulation of the keyphrase extraction problem, the abstract/keyphrases data pairs are converted such that each document is a sequence of word tokens, each with a positive (**KP**) label if it occurs in a keyphrase, or with a negative (**Non-KP**) label, otherwise (consistent with [13]). We train four network models: Bi-LSTM-CRF, CRF, Bi-LSTM, and LSTM. Each sentence from a document is passed to a network as the input sequence. To evaluate performance and runtime differences, we also train Bi-LSTM-CRF models with the entire document as the input sequence (instead of each sentence at a time).

We use word embeddings as input to the above four models. The word embeddings are initialized with 100-dimension Glove pre-trained embedding vectors [37]. For all models, we use a single 100-dimension hidden layer. The LSTM, Bi-LSTM, and Bi-LSTM-CRF are optimized during training using stochastic gradient descent with learning rate $\eta_t = \frac{\eta_0}{1+\rho_t}$, where initial learning rate $\eta_0 = 10^{-2}$ and decay ratio $\rho_t = 0.5$. Gradient clipping of 5.0 is used to prevent the gradient from overflows during back-propagation. In addition, we use dropout to avoid over-fitting. We select the model with the best F1 score on the validation set, **kp20k-validation**.

Our implementation of the network models is based on a modified version of the implementation developed by [26].[1] Publicly available implementations were used for the other baselines [33].[2]

4.3 Baselines and Evaluation Measures

We contrast the Bi-LSTM-CRF network with three baselines: CRF, forward LSTM, and Bi-LSTM, and several previous models: copy-RNN [33], KEA [11], *Tf-Idf*, TextRank [34] and SingleRank [41].

[1] https://github.com/LiyuanLucasLiu/LM-LSTM-CRF
[2] https://github.com/memray/seq2seq-keyphrase

Table 2: Results (Pr, Re, F1) of Bi-LSTM-CRF in an ablation experiment on the kp20k, WWW, and KDD datasets.

Method	kp20k			WWW			KDD		
	Pr%	Re%	F1%	Pr%	Re%	F1%	Pr%	Re%	F1%
Bi-LSTM-CRF	64.19	24.66	**35.63**	64.33	28.43	**39.43**	**57.83**	31.85	**41.08**
CRF	**66.67**	10.04	17.46	**64.89**	22.11	32.98	55.76	18.69	27.99
Bi-LSTM	9.41	76.24	16.75	9.53	**76.76**	16.96	8.15	**75.93**	14.71
LSTM	9.41	**78.43**	16.81	9.51	75.71	16.90	8.13	75.38	14.68

Consistent with previous works, we evaluate the predictions of each model against the author-input keyphrases, which are available with each document, i.e., the gold standard, and report Precision, Recall, and F1-score. For model comparison in the next section, we focus the discussion of our results in terms of the F1-score, which is the harmonic mean of Precision and Recall.

5 RESULTS AND OBSERVATIONS

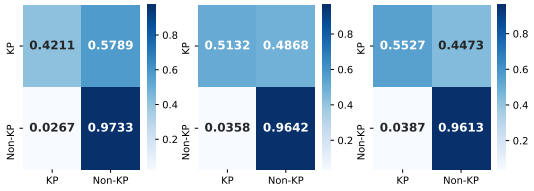
We now describe the evaluation performance of the Bi-LSTM-CRF model for extracting keyphrases from research papers.

5.1 Performance of Bi-LSTM-CRF for Keyphrase Extraction

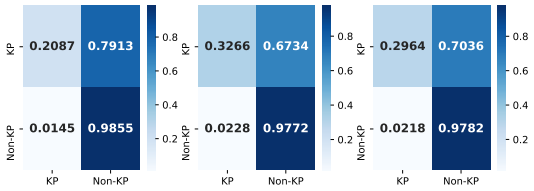
First, we evaluate the performance of Bi-LSTM-CRF in an ablation experiment to determine the role played by each component in extracting keyphrases from research papers. Specifically, we compare the Bi-LSTM-CRF model with a bidirectional LSTM, a forward LSTM, and a CRF model (by removing one component at a time from the full Bi-LSTM-CRF model).

Table 2 shows the results of this comparison. As can be seen from the table, Bi-LSTM-CRF consistently achieves the best results on all three datasets in terms of the F1-score. For example, on the **kp20k** dataset, Bi-LSTM-CRF achieves an F1-score of 35.63%, whereas CRF and Bi-LSTM achieve an F1-score of 17.46% and 16.75%, respectively. Interestingly, removing the Bi-LSTM component and using only a CRF model, we notice a steep drop in recall on all three datasets. For example, on **kp20k**, Bi-LSTM-CRF achieves a recall of 24.66%, whereas CRF alone achieves a recall of only 10.04%. This result demonstrates that leveraging the long distance semantic dependencies from text through the Bi-LSTM component of the full model is beneficial for correctly extracting a larger fraction of gold keyphrases. The CRF model alone generally achieves a similar or small increase in precision compared with the Bi-LSTM-CRF model. For example, on **WWW**, the precision values of Bi-LSTM-CRF and CRF are 64.33% and 64.89%, respectively, whereas on the **kp20k** dataset, these values are 64.19% and 66.67%, respectively.

On the other hand, removing the CRF component and using only an LSTM model (either bidirectional or forward LSTM) yields a consistent and substantial improvement in recall on all datasets over both CRF and Bi-LSTM-CRF at the expense of a dramatic drop in precision, consistently across all datasets (see Table 2). That is, on **kp20k**, the precision drops from 64.19% (achieved by Bi-LSTM-CRF) to an unacceptable value of 9.41% (obtained by LSTM alone), whereas the recall increases from 24.66% to 78.43%, obtained by Bi-LSTM-CRF and LSTM, respectively. This result indicates that LSTM is powerful in capturing the deep semantics of the text, and is



(a) Bi-LSTM-CRF



(b) CRF

Figure 5: Keyphrase extraction confusion matrices on kp20k (left), WWW (middle), and KDD (right) (results reported at word-level). The darker the blue on the main diagonal, the more accurate the model is.

able to achieve a high recall. This result also indicates that the CRF component of the full Bi-LSTM-CRF model successfully captures the label dependencies among the output labels for identifying keyphrases and contributes the most towards the full model's precision. Thus, exploiting dependencies in both the textual content of a document and the sequence of labels through the combination of an LSTM model with a CRF model yields improved results over CRF and LSTM models alone, quantified by a much larger F1-score.

Table 2 shows also that the performance of bidirectional-LSTM is very similar to that of forward LSTM. Intuitively, this makes sense since, in scientific papers, often dependencies occur in a forward fashion, e.g., patterns such as "we propose/study/explore/describe" precede keyphrases.

Moreover, from Table 2, we can also see that the performance on the **WWW** and **KDD** datasets is higher than that on the **kp20k** dataset. The lower performance on **kp20k** could be due to a larger spectrum of venues, author writing styles and author keyphrase annotations, whereas **WWW** and **KDD** are specialized datasets of papers from the data mining and machine learning communities.

Figures 5a and 5b show the confusion matrices of Bi-LSTM-CRF and CRF, on all three datasets. Each matrix is represented as a heat map, i.e., the darker the color, the higher the value at that position. An accuracy of 100% will be represented by a matrix with dark blue blocks on the main diagonal and white blocks off diagonal. As can be seen from the figures, the Bi-LSTM-CRF model that incorporates

Table 3: Comparison of Bi-LSTM-CRF with previous approaches on kp20k, WWW, and KDD datasets.

Method	kp20k			WWW			KDD		
	Pr%	Re%	F1%	Pr%	Re%	F1%	Pr%	Re%	F1%
Bi-LSTM-CRF	**64.19**	24.66	**35.63**	**64.33**	**28.43**	**39.43**	**57.83**	**31.85**	**41.08**
copyRNN @5	27.71	**41.79**	33.29	11.47	14.72	12.89	8.59	11.8	9.94
Tf-Idf @5	8.97	13.49	10.77	8.90	10.00	9.40	8.30	10.20	9.20
TextRank @5	15.29	23.01	18.37	5.80	7.10	6.20	5.10	6.50	5.60
SingleRank @5	8.42	12.70	10.14	8.80	10.90	9.50	7.70	10.30	8.60
KEA	15.14	22.78	18.19	13.57	15.25	13.86	11.39	14.50	12.42

dependencies in both the input and output, is more accurate (and thus, has a lower percentage of mis-labeled keyphrases) showed by darker cells corresponding to the **KP-KP** entry in all matrices.

5.2 Baseline Comparisons

Second, we compare the performance of Bi-LSTM-CRF with existing state-of-the-art models including supervised, unsupervised as well as deep learning models. The unsupervised models are Tf-Idf, TextRank [34], and SingleRank [41]. The supervised model is KEA[3] [43], and the deep learning model is the recently proposed copyRNN[4] [33], which is a sequence to sequence learning model based on an RNN Encoder-Decoder framework [7], combined with a copying mechanism. The Bi-LSTM-CRF, copyRNN, and KEA models are all trained on the **kp527k** dataset. For the unsupervised models and the sequence to sequence learning model, we report the performance at top 5 predicted keyphrases since top-5 showed highest performance in previous works for these models.

Table 3 shows the results of this comparison. The Bi-LSTM-CRF model outperforms all baselines in terms of the F1-score, on all three datasets. More notably, Bi-LSTM-CRF outperforms the copyRNN model in precision on all datasets, yet is slightly worse in the recall score on the **kp20k** dataset. For example, on the **kp20k**, Bi-LSTM-CRF achieves an F1-score of 35.63% as compared with 33.29% achieved by copyRNN. The precision increases from 27.71% (obtained by copyRNN) to 64.19% (obtained by Bi-LSTM-CRF) at the expense of a drop in recall from 41.79% (obtained by copyRNN) to 24.66% (obtained by Bi-LSTM-CRF).

This result is consistent with our findings in the previous section regarding the role of CRF in improving precision. Although both models use variants of RNNs to capture the semantics of the input sequence, by integrating the learning of phrasal structure into the model itself, via the CRF layer, we get higher performance than applying beam search to a decoded sequence after the learning phase, as is done in the copyRNN model.

5.3 Sentence vs. Document Level Input

Third, we conduct an experiment to examine the effect of the type of input sequence used to the models. Specifically, we compare the model where the input sequence consists of the entire document with the model where the input sequence consists of each sentence.

[3]http://www.nzdl.org/Kea/Download/Kea-4.0.zip
[4]https://github.com/memray/seq2seq-keyphrase

Table 4: Comparison results for document ("doc") vs. sentence ("sent") level for Bi-LSTM-CRF on the kp20k dataset.

Method		Pr%	Re%	F1%
Bi-LSTM-CRF	doc	**67.30**	**30.32**	**41.81**
	sent	64.19	24.66	35.63

Table 4 shows the results of this comparison for document vs. sentence level Bi-LSTM-CRF, on the **kp20k** dataset. As can be seen from the table, the performance of the Bi-LSTM-CRF that takes as input the entire content (all sentences at once) is higher than that of the Bi-LSTM-CRF that takes as input each sentence at a time. Precisely, precision, recall, and F1 values for the document-level Bi-LSTM-CRF show improvement by up to 6% over the sentence-level Bi-LSTM-CRF. This is likely due to the increased information captured by the LSTM layer (through the larger context) when using the entire content as opposed to each sentence.

A common drawback with increasing input sequence length for a recurrent neural network is that computing the gradients during back-propagation takes longer, which in turn increases the time it takes to train the model. For example, in our setting, training the Bi-LSTM-CRF model on the **kp527k** dataset at the document level took on average 22.1 hours for only five epochs, where the average input sequence size is 147 tokens, whereas when training Bi-LSTM-CRF using the sentence-level, it took only 2.7 hours, where the average number of tokens in a sentence is 16.

6 CONCLUSIONS

In this paper, we formulated the keyphrase extraction task as sequence labeling and proposed to use a Bi-LSTM-CRF model that incorporates dependencies among both the input and output sequences. To our knowledge, this is the first work to propose a neural network based sequence labeling model for keyphrase extraction from scholarly documents. The model takes advantage of the ability of Bi-LSTM to capture long distance semantic information from the input sequence and the ability of CRF to capture dependencies from the label sequence. Experimental results on three datasets showed that the proposed Bi-LSTM-CRF model that takes into account both of these dependencies play an important role in the keyphrase extraction performance.

As future work, it would be interesting to integrate the relationships between documents such as those available from a citation network. Another interesting direction would be to extend keyphrase extraction approaches to other scholarly domains.

ACKNOWLEDGEMENTS

We thank NSF for support from grants IIS-1802358, IIS-1813571, CNS-1853919, and CNS-1823288, which supported this research.

REFERENCES

[1] Eytan Adar and Srayan Datta. 2015. Building a Scientific Concept Hierarchy Database (SCHBase). In *Proceedings of the Association for Computational Linguistics*. 606–615.

[2] Isabelle Augenstein and Anders Søgaard. 2017. Multi-Task Learning of Keyphrase Boundary Classification. In *Proceedings of ACL (Volume 2: Short Papers)*, Vol. 2. 341–346.

[3] Ken Barker and Nadia Cornacchia. 2000. Using noun se heads to extract document keyphrases. (2000), 40–52.

[4] Pinaki Bhaskar, Kishorjit Nongmeikapam, and Sivaji Bandyopadhyay. 2012. Keyphrase Extraction in Scientific Articles: A Supervised Approach. In *Proceedings of COLING 2012: Demonstration Papers*. Mumbai, India, 17–24.

[5] Florin Adrian Bulgarov and Cornelia Caragea. 2015. A Comparison of Supervised Keyphrase Extraction Models. In *Proceedings of the 24th International Conference on World Wide Web Companion, WWW 2015, Florence, Italy, May 18-22, 2015 - Companion Volume*. 13–14.

[6] Cornelia Caragea, Florin Bulgarov, Andreea Godea, and Sujatha Das Gollapalli. 2014. Citation-Enhanced Keyphrase Extraction from Research Papers: A Supervised Approach. In *Proceedings of EMNLP*. 1435–1446.

[7] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[8] Soheil Danesh, Tamara Sumner, and James H Martin. 2015. SGRank: Combining Statistical and Graphical Methods to Improve the State of the Art in Unsupervised Keyphrase Extraction. *Lexical and Computational Semantics* (2015), 117.

[9] Samhaa R El-Beltagy and Ahmed Rafea. 2010. Kp-miner: Participation in semeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*. Association for Computational Linguistics, 190–193.

[10] Corina Florescu and Cornelia Caragea. 2017. PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Vancouver, Canada, 1105–1115.

[11] Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. 1999. Domain-Specific Keyphrase Extraction. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*. 668–673.

[12] Sujatha Das Gollapalli and Cornelia Caragea. 2014. Extracting keyphrases from research papers using citation networks. In *Proceedings of the 28th AAAI*. 1629–1635.

[13] Sujatha Das Gollapalli, Xiaoli Li, and Peng Yang. 2017. Incorporating Expert Knowledge into Keyphrase Extraction. In *Proceedings of the Thirty-First AAAI Conference, San Francisco, California, USA*. 3180–3187.

[14] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 6645–6649.

[15] Maria Grineva, Maxim Grinev, and Dmitry Lizorkin. 2009. Extracting key terms from noisy and multitheme documents. In *Proceedings of the 18th International Conference on World Wide Web*. 661–670.

[16] Khaled M Hammouda, Diego N Matute, and Mohamed S Kamel. 2005. Corephrase: Keyphrase extraction for document clustering. In *Machine Learning and Data Mining in Pattern Recognition*. Springer, 265–274.

[17] Kazi Saidul Hasan and Vincent Ng. 2014. Automatic Keyphrase Extraction: A Survey of the State of the Art. In *Proceedings of the 52nd ACL*. Baltimore, Maryland, 1262–1273.

[18] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[19] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).

[20] Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 216–223.

[21] Steve Jones and Mark S. Staveley. 1999. Phrasier: A System for Interactive Document Retrieval Using Keyphrases. In *Proc. of the 22nd Annual International ACM SIGIR*. 160–167.

[22] Daniel Kelleher and Saturnino Luz. 2005. Automatic hypertext keyphrase detection. In *IJCAI*, Vol. 5. 1608–1609.

[23] Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2013. Automatic keyphrase extraction from scientific articles. *Language resources and evaluation* 47, 3 (2013), 723–742.

[24] John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. (2001).

[25] Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. 2009. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of ACL*. 620–628.

[26] Liyuan Liu, Jingbo Shang, Frank Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower Sequence Labeling with Task-Aware Neural Language Model. In *Proceedings of AAAI*.

[27] Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic keyphrase extraction via topic decomposition. In *Proceedings of EMNLP*. 366–376.

[28] Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of EMNLP*. 257–266.

[29] Patrice Lopez and Laurent Romary. 2010. HUMB: Automatic key term extraction from scientific articles in GROBID. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 248–251.

[30] Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany, 1064–1074. http://www.aclweb.org/anthology/P16-1101

[31] Juan Martinez-Romo, Lourdes Araujo, and Andres Duque Fernandez. 2016. SemGraph: Extracting keyphrases following a novel semantic graph-based approach. *Journal of the Association for Information Science and Technology* 67, 1 (2016), 71–82.

[32] Olena Medelyan, Eibe Frank, and Ian H Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. ACL, 1318–1327.

[33] Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep Keyphrase Generation. In *Proceedings of the ACL*. 582–592.

[34] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. 404–411.

[35] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

[36] Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *Asian Digital Libraries*. Springer, 317–326.

[37] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*. 1532–1543.

[38] Vahed Qazvinian, Dragomir R. Radev, and Arzucan Özgür. 2010. Citation Summarization Through Keyphrase Extraction. In *Proceedings of the 23rd COLING (COLING '10)*. 895–903.

[39] Lucas Sterckx, Cornelia Caragea, Thomas Demeester, and Chris Develder. 2016. Supervised Keyphrase Extraction as Positive Unlabeled Learning. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. 1924–1929.

[40] Nedelina Teneva and Weiwei Cheng. 2017. Salience Rank: Efficient Keyphrase Extraction with Topic Modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vol. 2. 530–535.

[41] Xiaojun Wan and Jianguo Xiao. 2008. Single Document Keyphrase Extraction Using Neighborhood Knowledge. In *Proceedings of the 2008 AAAI*. 855–860.

[42] Rui Wang, Wei Liu, and Chris McDonald. 2014. Corpus-independent Generic Keyphrase Extraction Using Word Embedding Vectors. In *Software Engineering Research Conference*. 39.

[43] Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. 1999. KEA: Practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries*. ACM, 254–255.

[44] Chengzhi Zhang, Huilin Wang, Yao Liu, Dan Wu, Yi Liao, and Bo Wang. 2008. Automatic Keyword Extraction from Documents Using Conditional Random Fields. *Journal of Computational Information Systems* 4(3) (2008), 1169–1180.

[45] Qi Zhang, Yang Wang, Yeyun Gong, and Xuanjing Huang. 2016. Keyphrase extraction using deep recurrent neural networks on Twitter. In *EMNLP*. 836–845.

[46] Yongzheng Zhang, Evangelos Milios, and Nur Zincir-Heywood. 2007. A comparative study on key phrase extraction methods in automatic web site summarization. *Journal of Digital Information Management* 5, 5 (2007), 323.

LAVA: Longitudinal Adversarial Attack on Electronic Health Records Data

Sungtae An
Georgia Institute of Technology
Atlanta, Georgia
stan84@gatech.edu

Cao Xiao
IQVIA
Cambridge, Massachusetts
cao.xiao@iqvia.com

Walter F. Stewart*
HINT Consultants
Walnut Creek, California
wfs502000@yahoo.com

Jimeng Sun
Georgia Institute of Technology
Atlanta, Georgia
jsun@cc.gatech.edu

ABSTRACT

Although deep learning models trained on electronic health records (EHR) data have shown state-of-the-art performance in many predictive clinical tasks, the discovery of adversarial examples (i.e., input data that are engineered to cause misclassification) has exposed vulnerabilities with lab and imaging data. We specifically consider adversarial examples with longitudinal EHR data, an area that has not been previously examined because of the challenges with temporal high-dimensional and sparse features. We propose Longitudinal AdVersarial Attack (LAVA), a saliency score based adversarial example using a method that requires a minimal number of perturbations and that automatically minimizes the likelihood of detection. Features are selected and modified by jointly modeling a saliency map and attention mechanism. Experimental results with longitudinal EHR data show that LAVA can substantially reduce model performance for attention-based target models (from AUPR = 0.5 to AUPR = 0.08).

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by classification**; **Neural networks**; • **Applied computing** → **Health informatics**.

KEYWORDS

Health analytics; predictive model; adversarial examples; attention mechanism; neural networks

ACM Reference Format:
Sungtae An, Cao Xiao, Walter F. Stewart, and Jimeng Sun. 2019. LAVA: Longitudinal Adversarial Attack on Electronic Health Records Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313528

*Work done at Sutter Health.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313528

1 INTRODUCTION

Electronic health record (EHR) data from millions of patients are now routinely collected across diverse healthcare systems and physician practices, that can be represented as a longitudinal patient-level history of diagnoses, laboratory test results, and medication prescriptions orders, procedure orders, among other events and accompanied by demographic and other features. Deep learning models trained on these data are yielding high levels of performance for many clinical tasks such as diagnostic classification [15], disease detection [5], risk prediction [24]. In some instances, the performance of these models exceed the capabilities of experienced physicians in head-to-head comparisons [2]. However, the discovery of "adversarial examples" has exposed vulnerabilities in deep learning models across various application domains [9]. The connected EHR systems are under risk of adversarial attacks for a number of reasons including manipulations of pharmaceutical and device approvals, medical reimbursement decisions, etc [7]. As a major means of adversarial attacks, the "adversarial examples" are crafted to cause diagnostic errors via perturbation of EHR data, and further trigger significant consequences in treatment or other patient outcomes. Therefore, the risks from adversarial attacks need to be mitigated. Ultimately, a good understanding of such adversarial examples on longitudinal EHR can help patients and EHR administrators to proactively prevent the high-risk attacks to their records.

Adversarial examples have received much attention in computer vision [13, 19, 21] and natural language processing [12, 29], but relatively little work has been done with a focus on EHR data. Finlayson et al. [7] demonstrated vulnerability of convolutional neural network models trained with medical imaging datasets, replicating prior work on imaging from other industries. Sun et al. [26] proposed a recurrent neural network (RNN) based time-preferential minimum attack model to identify susceptible locations in clinical time series data focused on real-valued continuous variables such as vital sign and lab measurements. Despite all these efforts, adversarial attacks on longitudinal EHR data on neural network models is still fairly open. More generally, challenges to devising adversarial examples for longitudinal EHR data include:

- *High-dimensional, discrete and sparse feature space*: Unlike most other domains such as image and audio waveform, EHR data are characterized by high-dimensional, discrete, and extremely sparse feature space. As many as 80% to 90% of the 10,000 or more

features (i.e., diagnosis, medication, and procedure codes) have non-null values less than 1% of the time. Thus, existing crafting algorithms for continuous or dense data cannot be directly applied because they assume perturbations were broadly spread over the entire feature space.

- *Feature significance and relevance*: Each medical code differs in clinical significance and relevance to prediction tasks. For example, diagnostic codes for fever and heart failure inform the different level of risks in terms of morbidity and mortality risks. Perturbation of critical features such as heart failure could be readily detected even though they are the most effective attacks.
- *Temporal significance*: Patients' health conditions evolve over time. The temporality encoded in time-stamped medical events reveals important information on impending patient health conditions, with differential significance on prediction outcomes. For example, more recent events are likely to have a stronger influence on the prediction result. Thus adversarial attacks on more recent events are more likely to be detected.

To address these challenges, we propose LAVA (**L**ongitudinal **A**d**V**ersarial **A**ttack), a saliency score based method for discrete and sequential EHR data. LAVA introduces a minimal number of perturbations while also avoiding the selection of features and clinical encounters whose modifications are highly likely to be detected. This is achieved by jointly modeling the saliency map and attention mechanism. We specifically propose LAVA with the following technical contributions.

- *Efficient crafting with a bidirectional saliency map*: We proposed a new bidirectional saliency map that searches candidate features from both (+/−) altering directions simultaneously.
- *Low detectability with an attention mechanism*: LAVA automatically avoids choosing features that have high relevance to the prediction targets. This is achieved via a dual-level attention mechanism that can highlight and avoid important visits and important features within those visits.

In experimental modeling with real-world data show that LAVA substantially reduces performance (i.e., area under the precision-recall curve or AUPR) for the white-box[1] type of attention-based target models from 0.5 to 0.08, and achieves comparable performance reductions for other gray-box[2] target models, while providing minimal to non-detectable perturbations.

2 RELATED WORK

Adversarial examples can be crafted in different ways and categorized by the norm they use. In general, L_p-norm, $\|\cdot\|_p$ measures the distance between an original input $\mathbf{x}$ and the corresponding adversarial example $\tilde{\mathbf{x}}$. The choice of L_p (e.g., L_0, L_1, L_2, or L_∞) is related to the problem domain. Goodfellow et al. [9] proposed the Fast Gradient Sign Method (FGSM), a widely used method that generates untargeted adversarial examples optimized under the L_∞ norm. FGSM works by adding perturbations of size ϵ to the direction of increasing loss function $J(\theta, \mathbf{x}, y)$ used to train the model F with the model parameters θ, the input $\mathbf{x}$, and the target y. Kurakin et al. [14] proposed an iterative variant of FGSM called Projected Gradient Descent (PGD), which shows better generalization and performance for most cases. PGD is a universal adversary among first-order approaches [17]. Moosavi Dezfooli et al. [20] proposed an approach that uses a directional derivative to find a distance from the input data point in the feature space to the decision boundary, crafting adversarial examples that minimize perturbations.Papernot et al. [22] proposed Jacobian-based Saliency Map Approach (JSMA) that finds a feature that has the highest overall impact on leading a classifier to an aimed wrong class label at each iteration under the L_0 (or L_1 according to the termination condition) norm constraint.

L_∞ norm constrained approaches (e.g., FGSM and PGD) add perturbations to each dimension of the entire feature space; thus, it is difficult to apply these methods directly to EHR data given the sparseness of the feature space, where non-zero values are rare. We show the perturbation distribution generated by PGD from EHR data in Section 5. For both the discrete and sparse feature space we selected L_0 or L_1 as the options for the target domain, where the proposed method LAVA is optimized under L1 objective.

3 BACKGROUND

3.1 Jacobian-based Saliency Map

The Jacobian-based Saliency Map Approach (JSMA) [22] finds a feature that has the highest overall impact on leading a classifier to a targeted wrong class label at each iteration. Given $F : \mathbb{R}^d \mapsto \mathbb{R}^s$, a neural network that maps an input vector $\mathbf{x} \in \mathbb{R}^d$ to probabilities over s classes, x_k is the k-th element of the input vector $\mathbf{x}$, and F_r is the model output probability for the class r w.r.t. the input $\mathbf{x}$, the Jacobian matrix of the model is computed as the forward derivative of the neural network, where the element at the k-th column and the r-th row is denoted as $\mathbf{J}_{rk}$, is given by $\mathbf{J}_{rk} = \frac{\partial F_r}{\partial x_k}$. Assuming that t is the target class that an adversary wants the neural network model incorrectly classify the input $\mathbf{x}$ into, a *target class saliency* A_k and an *overall non-target classes saliency* B_k for feature k are given by

$$A_k = \mathbf{J}_{tk} = \frac{\partial F_t}{\partial x_k}, \quad B_k = \sum\nolimits_{r \neq t} \mathbf{J}_{rk} = \sum\nolimits_{r \neq t} \frac{\partial F_r}{\partial x_k}. \tag{1}$$

We can then construct a saliency map $S(\mathbf{x}, t)$ based on A_k and B_k to search candidate features. The JSMA is applicable to craft adversarial example from EHR data since it perturbs one feature at each iteration. However, it only searches along one direction (i.e., either increasing or decreasing) and thus is not adequately efficient given the very high feature space in EHR data.

3.2 Neural Attention Mechanism

The attention mechanism is increasingly used to understand what part of spatial or temporal information contributes most in predicting target values. It is widely used applications for computer vision [18, 28], natural language processing [1, 11], and speech recognition [6]. When introduced to EHR modeling, attention weights indicate the degree to which past clinical events predict disease onsets or future events [3, 16]. Choi et al. [4] proposed a model architecture with a dual-level attention mechanism, RETAIN, which calculates attention weights at the visit and feature (embedding) levels to compose the context vector for the final predictions. In this current

[1]White-box means both training data and the model architecture are available to the attackers.
[2]Gray-box refers to the setting where model architectures are unknown but training data are available to the attackers.

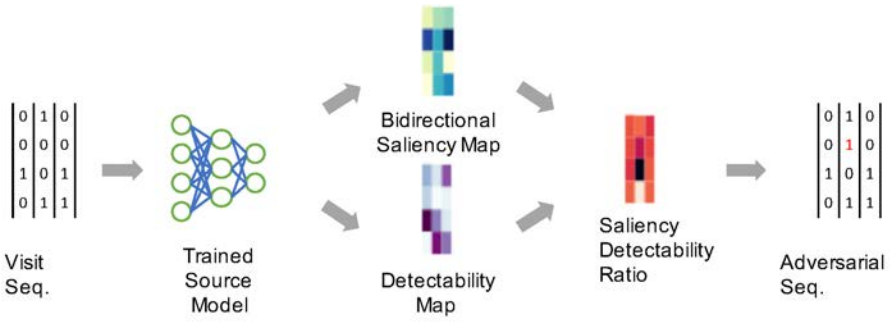


Figure 1: Crafting iteration of LAVA. LAVA balances the effectiveness of the attack through the bidirectional saliency and the detectability of the attack over longitudinal EHR data. Given a trained model and an input visit sequence of a patient, LAVA calculate both the bidirectional saliency score and the detectability cost for each feature at each visit. LAVA increases or decreases the binary feature value that has the highest saliency-to-detectability ratio at a specific visit. This procedure is iteratively performed until it satisfies stopping criteria.

work, we have adopted the RETAIN architecture to capture both visit and feature level significance and use this information to estimate the likelihood of perturbations to be discovered by a target model or defender.

4 LAVA: LONGITUDINAL ADVERSARIAL ATTACK

Given longitudinal EHR data from N patients, denote $\mathbf{x}^{(n)}$ as the clinical trajectory of patient n, which is characterized by a sequence of T^n hospital visits. Then $\mathbf{x}^{(n)}$ can be formulated as given by Eq. 2.

$$\mathbf{x}^{(n)} = \left\{\mathbf{x}_1^{(n)}, \mathbf{x}_2^{(n)}, \dots \mathbf{x}_{T^n}^{(n)}\right\} \tag{2}$$

where $\mathbf{x}_i^{(n)} \in \mathbb{R}^d$ denotes the concatenation of medical codes (often binary encoded) of the i-th visit made by patient n. In the following, we omit the superscript (n) to reduce clutter (e.g., represent $\mathbf{x}^{(n)}$ as $\mathbf{x}$, and $\mathbf{x}_i^{(n)}$ as $\mathbf{x}_i$). We also denote $x_{i,k}$ as the k-th dimension of the i-th visit $\mathbf{x}_i$ for a patient. Then, for a given patient records $\mathbf{x}$, a trained model F, and an original label $y = F(\mathbf{x})$, crafting adversarial example from longitudinal EHR requires finding a perturbation $\delta_\mathbf{x}$:

$$\underset{\delta_\mathbf{x}}{\operatorname{argmin}} \|\delta_\mathbf{x}\|_p \quad \text{s.t. } F(\mathbf{x} + \delta_\mathbf{x}) = \tilde{y} \neq y \tag{3}$$

The resulting adversarial example $\tilde{\mathbf{x}} = \mathbf{x} + \delta_\mathbf{x} = \{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots \tilde{\mathbf{x}}_T\}$ can have $\tilde{\mathbf{x}}_i \neq \mathbf{x}_i$ and $\tilde{x}_{i,k} \neq x_{i,k}$ for any k-th feature in any i-th visit.

4.1 Overview

We propose LAVA as an adversarial examples crafting method for discrete and sequential EHR data. LAVA iteratively searches features from selected clinical visits that are suitable as adversarial examples as defined by a higher ratio of the sensitivity to fool a given model over the likelihood of the attack being detected. LAVA executes the following sequential procedure.

- Computes the **bidirectional saliency score** of each feature at each visit for perturbation to measure suitability as an adversarial example.
- Computes the **detectability cost**, which is the likelihood that modification to a feature will be detected.
- Calculates the **saliency-to-detectability ratio** to determine which feature at a specific visit is optimal to be perturbed.

This procedure is iteratively performed until it satisfies stopping criteria. Figure 1 depicts the overall process pipeline for an iteration of LAVA, and the details of each step are described in the following sections.

4.2 Bidirectional Saliency Score

To find the most prominent feature that contributes to the current classification result, we propose the *bidirectional saliency score* that rearranges and combines the computation of increasing and decreasing saliency map [22] with a minimal additional computational cost. We first derive a bidirectional saliency map for non-sequential data first, then extend it to sequential data.

Non-sequential Data. Given (1), the bidirectional saliency map $S(\mathbf{x}, t)$ for the input $\mathbf{x}$ and the target label t is given by (4).

$$S(\mathbf{x}, t)[k] = \operatorname{sgn}(B_k) \min(0, A_k B_k) \tag{4}$$

Here a positive valued $A_k B_k$ indicates two possible cases: both A_k and B_k are either positive or negative simultaneously, such that feature x_k has the impact on the probabilities of target class and non-target classes toward the same direction (e.g., increase or decrease). To avoid this ambiguity, we limit $A_k B_k$ to negative values only where the feature x_k has opposing directional effect on the target class and on the non-target classes. When this condition arises, a saliency score has a sign that is opposite to the sign of B_k but same as the sign of A_k since the resulting value of $\min(0, A_k B_k)$ is always negative. Consequently, this single bidirectional saliency map $S[k]$, omitting $\mathbf{x}$ and t, indicates which direction of perturbation on the k-th dimension of the input feature leads to a higher probability that the input is classified as the target class t with its relative strength given by the value of $S[k]$.

Sequential Data. For sequential data such as the longitudinal EHR sequence in our study, we firstly represent it as Eq. 2, then we adopt a RETAIN architecture [4], an RNN-based attention model,

to be the source model for crafting adversarial examples, and extend the bidirectional saliency method to handle sequential data. In particular, we extend the forward derivative, the Jacobian matrix whose element is given by $\mathbf{J}_{rk}$ in Section 3, to a sequential input for a recurrent neural network. Computing the forward derivative through a recurrent neural network is straightforward; any recurrent connections can be unrolled and transformed into a directed acyclic graph (DAG), and the partial derivatives of output with respect to input at each visit on the unrolled computational graph can easily be taken [8]. Thus, a Jacobian matrix can be computed for each visit, and they can be represented together as a tensor form with an additional time axis. Consequently, each element of the forward derivative and the corresponding target and non-target saliency are formulated as (5)–(7).

$$\mathbf{J}_{rik} = \frac{\partial F_r}{\partial x_{i,k}} \tag{5}$$

$$A_{ik} = \mathbf{J}_{tik} = \frac{\partial F_t}{\partial x_{i,k}} \tag{6}$$

$$B_{ik} = \sum\nolimits_{r \neq t} \mathbf{J}_{rik} = \sum\nolimits_{r \neq t} \frac{\partial F_r}{\partial x_{i,k}} \tag{7}$$

where $x_{i,k}$ is the k-th coordinate of the input vector at time i. As a result, a bidirectional saliency score map where each element is a score for each k-th feature at each i-th input $\mathbf{x}_i$ of a sequential input $\mathbf{x}$ can be obtained simply by (8).

$$S[i,k] = \operatorname{sgn}(B_{ik}) \min(0, A_{ik} B_{ik}) \tag{8}$$

Assuming that each visit vector $\mathbf{x}_i$ is a binary vector where each $x_{i,k}$ can have only either 1 or 0, we apply a restriction that a feature valued at 0 will not decrease and valued at 1 will not increase. Thus the bidirectional saliency score for sequential data is given by (9).

$$S[i,k] = \begin{cases} \max(0, \operatorname{sgn}(B_{ik}) \min(0, A_{ik} B_{ik})), & \text{if } x_{i,k} = 0 \\ \min(0, \operatorname{sgn}(B_{ik}) \min(0, A_{ik} B_{ik})), & \text{if } x_{i,k} = 1 \end{cases}. \tag{9}$$

4.3 Detectability Cost

We formulate the *detectability cost* using the attention mechanism to measure the imperceptibility of perturbations where the lower cost indicates the lower chance that a perturbation will be detected. The attention mechanism is a widely used technique to pay more 'attention' to features influential on model decisions; significant changes in the amount of 'attention' on each event reflect either a critical event has been newly added or an existing important event has been removed. In particular, the impact of each $x_{i,k}$ on the final classification result is measured from the two-level attention weights by (10) following the derivation from [4]:

$$w_{ik} = \alpha_i \mathbf{W}(\boldsymbol{\beta}_i \odot \mathbf{W}_{\text{emb}}[:,k]) \tag{10}$$

where α_i is the attention weight on the i-th visit, $\boldsymbol{\beta}_i$ is an attention weight vector for all features $x_{i,k}$ of the i-th visit, $\mathbf{W}$ is the output weight matrix, $\mathbf{W}_{\text{emb}}$ is the weight matrix at the embedding layer, and $\odot$ denotes element-wise multiplication. Here we assume that an input vector $\mathbf{x}_i$ at time i is a binary vector. Similar to the computation of the saliency score in the previous section, the forward derivative of each w_{jl} with respect to each input feature $x_{i,k}$ can be obtained from the unrolled computational graph. Then, we define the detectability cost by

$$D[i,k] = \sum_{j,l} \frac{\partial w_{jl}}{\partial x_{i,k}} \tag{11}$$

which captures the overall amount of change of the impact by all features and all visits when the feature $x_{i,k}$ is perturbed.

4.4 Saliency-to-Detectability Ratio

Last, we calculate *Saliency-to-Detectability Ratio* (*SDR*) at each iteration, which is defined as a ratio between the normalized saliency score and the normalized detectability cost as given by (12)–(13).

$$SDR[i,k] = \frac{\hat{S}_{ik}}{\hat{D}_{ik}} \tag{12}$$

where

$$\hat{S}_{ik} = \frac{S_{ik}}{\max_{j,l} \left|S_{jl}\right|}, \quad \hat{D}_{ik} = \frac{D_{ik}}{\max_{j,l} \left|D_{jl}\right|}. \tag{13}$$

Consequently, the feature $x_{i,k}$ that has the maximum value of $SDR[i,k]$ is perturbed in the direction of the sign of $SDR[i,k]$ at each iteration. To conclude, we summarize the pseudo code of LAVA in Algorithm 1. The source code of LAVA is publicly available at https://github.com/ast0414/lava

Algorithm 1 LAVA (Longitudinal AdVersarial Attack)

F is a trained model, $\mathbf{x}$ is an input sequence, $\tilde{\mathbf{x}}$ is a crafted adversarial sequence, t is a target class/label, and M is the maximum number of distortions

procedure LAVA($F, \mathbf{x}, t, M$)
  $\tilde{\mathbf{x}} \leftarrow \mathbf{x}$
  count $\leftarrow 0$
  $y, w \leftarrow F(\tilde{\mathbf{x}})$ ▷ y is a model output and w is a contribution matrix
  while count $< M$ and $y \neq t$ **do**
    Initialize S and D ▷ a bidirectional saliency map S and a detectability map D
    for all i-th visit and k-th feature **do**
      $A_{ik} \leftarrow \frac{\partial F_t}{\partial \tilde{x}_{i,k}}$
      $B_{ik} \leftarrow \sum_{r \neq t} \frac{\partial F_r}{\partial \tilde{x}_{i,k}}$
      $S[i,k] \leftarrow \operatorname{sgn}(B_{ik}) \min(0, A_{ik} B_{ik})$
      if $x_{i,k} == 0$ **then**
        $S[i,k] \leftarrow \max(0, S[i,k])$
      else
        $S[i,k] \leftarrow \min(0, S[i,k])$
      $D[i,k] \leftarrow \sum_{j,l} \frac{\partial w_{jl}}{\partial x_{i,k}}$
    $\hat{S}_{ik} \leftarrow S_{ik} / \max_{j,l} \left|S_{jl}\right|$
    $\hat{D}_{ik} \leftarrow D_{ik} / \max_{j,l} \left|D_{jl}\right|$
    for all i-th visit and k-th feature **do**
      $SDR[i,k] \leftarrow \hat{S}_{ik} / \hat{D}_{ik}$
    $i^*, k^* \leftarrow \operatorname{argmax}_{i,k} SDR[i,k]$
    $\tilde{x}_{i^*,k^*} \leftarrow \tilde{x}_{i^*,k^*} + 1 \cdot \operatorname{sgn}(SDR[i^*,k^*])$
    $y, w \leftarrow F(\tilde{\mathbf{x}})$
    count $\leftarrow$ count + 1
  return $\tilde{\mathbf{x}}$

Table 1: Descriptive statistics of EHR dataset

# total patients	30,742	Avg. # visits per patient	18.68
# case patients	3,409	Max. # visits per patient	246
# total visits	574,108	Avg. # codes per visit	3.27
# unique codes	17,081	Max. # codes per visit	47

Table 2: Model performances on the clean test set

Model	AUROC	AUPR	F1 Score	Accuracy
RETAIN-WHITE	0.851	0.495	0.465	0.821
RETAIN-GRAY	0.849	0.493	0.461	0.817
RNN	0.837	0.456	0.444	0.809
MLP	0.821	0.388	0.426	0.797

5 EXPERIMENTS

In this section, we evaluated the sequences with adversarial examples using different types of neural network models.

5.1 Setup

Data. We conducted all experiments using EHR data provided by Sutter Health. The dataset was originally constructed for a heart failure (HF) onset prediction study that consists of 30K patients (age 50-89) including 3K HF case patients chosen by a set of criteria described by Gurwitz et al. [10], Vijayakrishnan et al. [27] and 27K control patients matched by a set of criteria described by Choi et al. [5]. Patients data over an 18 month period were used in this study. The summary statistics of the data is described in Table 1.

Source model. To craft adversarial examples, we used the RETAIN [4] architecture with two independent GRU layers, each of which includes 128 hidden units and embedding layer of 128 dimensions.

Target models. To evaluate multiple aspects of the crafted adversarial examples, we considered the following white-box and gray-box target models. The white-box target model is identical to the source model that crafts adversarial examples. The gray-box models are ones with different architectures and/or different hyperparameters while they are trained using the same train data as the source model.

- *RETAIN-WHITE*: Pure white-box attack was evaluated on the exact same architecture and trained parameters as the source model.
- *RETAIN-GRAY*: A RETAIN model with a different set of hyperparameters was trained and evaluated on attacks. It has 256 hidden units for each GRU layer for visit and feature attention, and an embedding layer of 256 dimensions was used.
- *RNN*: An RNN model without attention mechanism. It includes two GRU layers, each consisting of 128 hidden units, and an embedding layer of 128 dimensions.
- *MLP*: A multi-layer perceptron (MLP) model that was trained on aggregated feature vectors where a patient is represented by a single vector without temporal information. Then, the crafted sequential adversarial examples were also aggregated for each patient and passed into the trained MLP. The MLP model consists of three hidden layers, each of which consists of 128 hidden units.

Baseline methods. We compare LAVA with the following crafting algorithms.

- LAVA *w/o detectability*: We used non-penalized saliency scores only to find a candidate feature at each iteration. In other words, this algorithm is equivalent to LAVA where the detectability costs for all feature over the visits are equal to one.
- *PGD*: Adversarial examples were crafted by Projected Gradient Descent [13], where we set the step size as 1 and limit feature space as binary.
- *Random*: At each iteration, a visit was randomly selected from the sequence of visits, and a feature, a medical code, was chosen also randomly to be perturbed.

Implementation details. All models were implemented using PyTorch 0.3 [23] and trained with a system equipped with Intel Xeon E5-2683v4 CPU, 756GB RAM, and Nvidia Tesla P100 GPU. We used Stochastic Gradient Descent (SGD) with momentum [25] for optimization. The momentum value was set to 0.9. The initial learning rate was set to 0.01 and was reduced by a factor of 10 once the learning stagnated.

5.2 Attack Evaluation

As a contrast, we first show how these target models perform on unperturbed data. Table 2 shows that all models achieved accurate predictions of a future HF diagnosis when using unperturbed test data. Figure 2 compares the performance of different craft algorithms by the extent to which they reduce prediction performance as defined by the Area Under Precision-Recall curve (AUPR).

- For attention based target models such as RETAIN-WHITE and RETAIN-GRAY, LAVA achieved the best performance drop and was able to beat the state-of-the-art gradient-based method, PGD, after 12 iterations.
- For some models (e.g., RNN and MLP) and during early iterations of attention models, LAVA caused less AUPR drop than LAVA w/o detectability, the non-penalized saliency score method. This is due to the fact that saliency score methods often perturbed the most critical feature, while LAVA avoids such perturbation. During interactive attack and defense, perturbing most critical features will again be easily caught. While LAVA address this concern, it still achieved good AUPR drops.
- Although PGD showed good performances in some settings, it perturbs a large number of features (1,650 per visit in our experiment) even in the first iteration. While LAVA is cost-sensitive, it only perturbs one feature at a time.
- Moreover, the random perturbations barely have any influence on the performance for all settings.

5.3 Perturbation Analysis

We evaluated the perturbations made by each algorithm to determine effectiveness and the ease of detection (i.e., from the EHR system perspective that is a target of an adversarial attack). Figure 3 shows how the perturbations in the adversarial examples crafted by each algorithm are distributed over the visit and feature

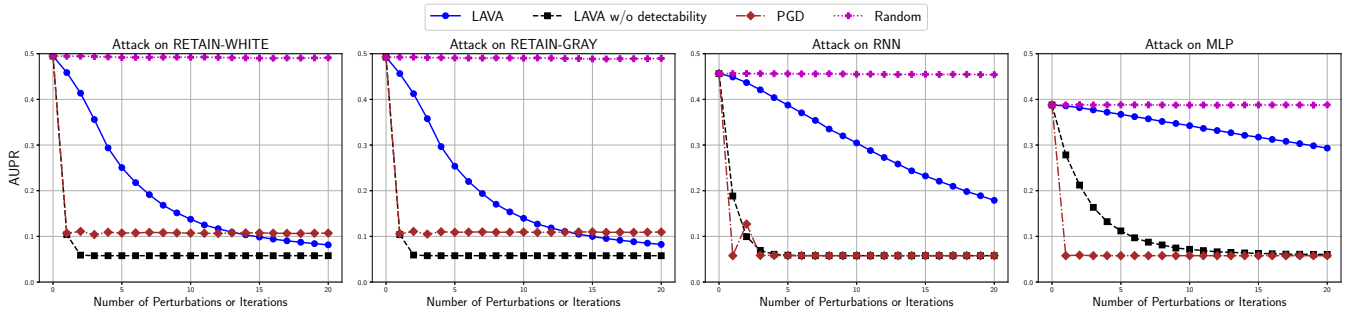


Figure 2: Comparison of the different types of attack on each target model.

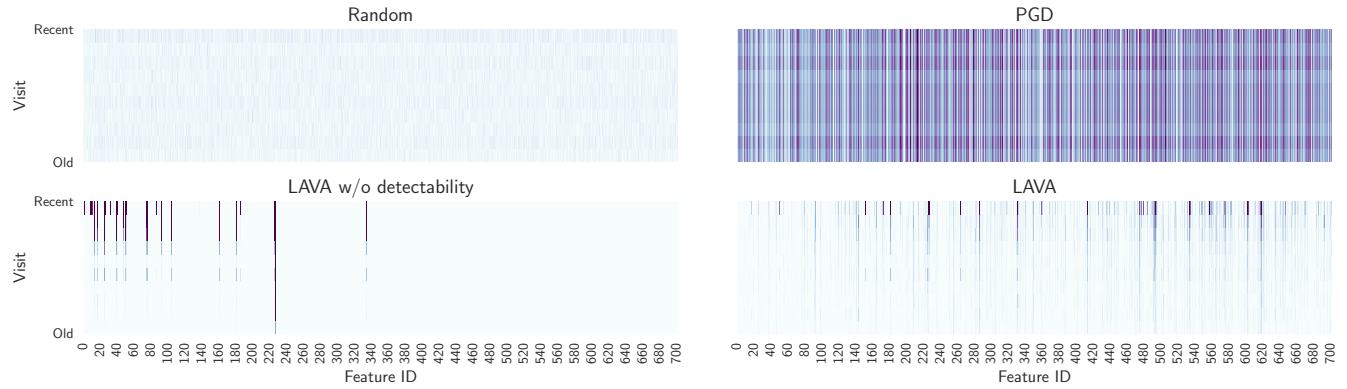


Figure 3: Comparison of perturbations by the different types of attack. The x-axis represents features ordered by feature ID. The y-axis represents visits in chronological order from the bottom to the top. Each heatmap depicts how the adversarial perturbations by each algorithm are distributed over the visits and features in the sequential EHR data. The white color means no perturbation and the darker color represents the more perturbations.

dimension of the test sequential EHR data. Random attacks spread perturbations over the entire visit and feature space, but each perturbation holds no meaningful attraction for the models since it was crafted in a purely random manner. PGD attacks also crafted perturbations over the entire space and are densely distributed, influencing almost every feature in every visit. PGD attacks would be easily discovered by a defender (e.g., an administrator of the EHR system that might be human or machine). Moreover, PGD attacks may be treated as a system malfunction since it may seem the data themselves are scrambled. On the contrary, the perturbations in the adversarial examples crafted using non-penalized saliency scores are concentrated on a few certain features in relatively recent visits. Thus, the administrator of a machine learning based EHR system can monitor and pay attention to only those particular features to make provisions against potential adversarial attacks by non-penalized saliency methods. Adversarial examples crafted by LAVA have perturbations spread wider than those made by the algorithm without detectability costs although the recent visits are still highlighted slightly more than the others. Hence, the administrator may need to monitor the system more carefully to figure out the existence of adversarial attacks; otherwise, it might be misunderstood for random noises that are naturally inherent in the data. In conclusion, it is shown that LAVA crafted the adversarial examples for sequential EHR data, where it is difficult to detect their existence in terms of the distribution of the perturbations over both visit and feature spaces.

6 CONCLUSION

In this work, we propose LAVA to craft adversarial examples on longitudinal EHR data. We compared the attack result by LAVA with other attacks. It is shown that the adversarial examples crafted by LAVA perform effective attacks toward white-box and gray-box target models. Moreover, it is also shown that the adversarial examples crafted by LAVA are relatively well spread over the visits and the feature space; it would be more difficult to detect such attacks generated by LAVA than other attacks. As future work, we plan to design and compare different defense mechanisms to provide better guidance on how to alleviate the adversarial attacks on longitudinal EHR data.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation, award IIS-1418511 IIS-1838042 and CCF-1533768, the National Institute of Health award 1R01MD011682-01 and R56HL138415.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[2] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*. 301–318.

[3] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. 2017. GRAM: Graph-based attention model for healthcare representation learning. In *SIGKDD*.

[4] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. 2016. RETAIN: An Interpretable Predictive Model for Healthcare using Reverse Time Attention Mechanism. In *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (Eds.). Curran Associates, Inc., 3504–3512.

[5] Edward Choi, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Using recurrent neural network models for early detection of heart failure onset. *Journal of the American Medical Informatics Association* 24, 2 (2016), 361–370.

[6] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Advances in neural information processing systems*. 577–585.

[7] Samuel G Finlayson, Isaac S Kohane, and Andrew L Beam. 2018. Adversarial Attacks Against Medical Deep Learning Systems. *arXiv preprint arXiv:1804.05296* (2018).

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[10] Jerry H Gurwitz, David J Magid, David H Smith, Robert J Goldberg, David D McManus, Larry A Allen, Jane S Saczynski, Micah L Thorp, Grace Hsu, Sue Hee Sung, et al. 2013. Contemporary prevalence and correlates of incident heart failure with preserved ejection fraction. *The American journal of medicine* 126, 5 (2013), 393–400.

[11] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. 1693–1701.

[12] Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328* (2017).

[13] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533* (2016).

[14] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236* (2016).

[15] Zachary C Lipton, David C Kale, Charles Elkan, and Randall Wetzell. 2016. Learning to diagnose with LSTM recurrent neural networks. In *ICLR*.

[16] Tengfei Ma, Cao Xiao, and Fei Wang. 2018. Health-ATM: A Deep Architecture for Multifaceted Patient Health Record Representation and Risk Prediction. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 261–269.

[17] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).

[18] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. 2014. Recurrent models of visual attention. In *Advances in neural information processing systems*. 2204–2212.

[19] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. 2017. Universal adversarial perturbations. *arXiv preprint* (2017).

[20] Seyed Mohsen Moosavi Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[21] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 506–519.

[22] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*. IEEE, 372–387.

[23] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS-W*.

[24] T. Pham, T. Tran, D. Phung, and S. Venkatesh. 2017. Predicting healthcare trajectories from medical records: A deep learning approach. *J. Biomed. Inform.* (2017).

[25] Ning Qian. 1999. On the momentum term in gradient descent learning algorithms. *Neural networks* 12, 1 (1999), 145–151.

[26] Mengying Sun, Fengyi Tang, Jinfeng Yi, Fei Wang, and Jiayu Zhou. 2018. Identify Susceptible Locations in Medical Records via Adversarial Attacks on Deep Predictive Models. *arXiv preprint arXiv:1802.04822* (2018).

[27] Rajakrishnan Vijayakrishnan, Steven R Steinhubl, Kenney Ng, Jimeng Sun, Roy J Byrd, Zahra Daar, Brent A Williams, Shahram Ebadollahi, Walter F Stewart, et al. 2014. Prevalence of heart failure signs and symptoms in a large primary care population identified through the use of text and data mining of the electronic health record. *J. Card. Fail.* 20, 7 (2014), 459–464.

[28] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*. 2048–2057.

[29] Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2017. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342* (2017).

A Graph is Worth a Thousand Words: Telling Event Stories using Timeline Summarization Graphs

Jeffery Ansah*
University of South Australia
jeffery.ansah@mymail.unisa.edu.au

Lin Liu
University of South Australia
lin.liu@unisa.edu.au

Wei Kang
University of South Australia
wei.kang@unisa.edu.au

Selasi Kwashie
University of South Australia
selasi.kwashie@mymail.unisa.edu.au

Jixue Liu
University of South Australia
jixue.liu@unisa.edu.au

Jiuyong Li
University of South Australia
jiuyong.li@unisa.edu.au

ABSTRACT

Story timeline summarization is widely used by analysts, law enforcement agencies, and policymakers for content presentation, story-telling, and other data-driven decision-making applications. Recent advancements in web technologies have rendered social media sites such as Twitter and Facebook as a viable platform for discovering evolving stories and trending events for story timeline summarization. However, a timeline summarization structure that models complex evolving stories by tracking event evolution to identify different themes of a story and generate a coherent structure that is easy for users to understand is yet to be explored. In this paper, we propose StoryGraph, a novel graph timeline summarization structure that is capable of identifying the different themes of a story. By using high penalty metrics that leverage user network communities, temporal proximity, and the semantic context of the events, we construct coherent paths and generate structural timeline summaries to tell the story of how events evolve over time. We performed experiments on real-world datasets to show the prowess of StoryGraph. StoryGraph outperforms existing models and produces accurate timeline summarizations. As a key finding, we discover that user network communities increase coherence leading to the generation of consistent summary structures.

CCS CONCEPTS

• **Information systems** → **Social networks**; **Summarization**;

KEYWORDS

StoryGraph, Twitter, event evolution, story timeline summarization

ACM Reference Format:
Jeffery Ansah, Lin Liu, Wei Kang, Selasi Kwashie, Jixue Liu, and Jiuyong Li. 2019. A Graph is Worth a Thousand Words: Telling Event Stories using Timeline Summarization Graphs. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313396

*Corresponding author. We acknowledge D2DCRC, Cooperative Research Centres Programme, and the University of South Australia for funding this research. The work is partially supported by ARC Discovery project DP170101306.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313396

1 INTRODUCTION

With the increasing explosion of Web 2.0 technologies, social media sites such as Twitter, Facebook and Weibo have provided a vast continuous supply of diverse information content [2, 4, 14, 16]. These sites provide functionalities for users to share and broadcast information on fashion trends, political debates, breaking news and other issues of interests [8, 17, 18, 28]. However, with the overwhelming amount of information generated from social media, it has become difficult for news analysts, law enforcement agencies and policymakers to obtain a summarative perspective of the large amount of information on trending topics, news stories and events.

A prevailing solution is to organize and present stories in a summarization timeline for easy comprehension [1, 14, 16, 23, 28]. This provides a better understanding of how events evolve throughout the life cycle of a story [8, 16, 23, 28], as well as reveal the flow and evolution of the various themes that make up a story. Such summaries help users to understand the general information of the story in a more concise and logical manner [14, 16].

Motivating Example. Figure 1 is a graph illustrating a story of the popular *"Black Lives Matter"* protest from Aug. to Nov. 2014 in the US. Each distinct colour represents a theme in the story graph. For example: all the green nodes represent related events that evolved in *Ferguson Missouri*; all orange coloured nodes are events related to the *shooting of Michael Brown*. Each node in this graph (i.e. the StoryGraph proposed in this paper) represents an event σ, and a connection between two events represents an event evolution in the story. For example, the path $\sigma_2 \rightarrow \sigma_{16}$ tells the story of the events that emerged from the *Ferguson Missouri protests* leading to the *formation of the Ferguson Committee*. As shown in Fig.1, it is of no doubt that modeling the evolutionary connections between events into a summarization timeline structure can help users to learn the main information quickly. However, developing such comprehensive summarization from the massive volume of social media posts inundated with spam and diverse range of events is like *"finding a needle in a haystack"* [9, 14, 28, 29].

In the attempts to solve timeline summarization problems, Traditional Document Summarization (TDS) [12, 21, 25] has been extensively studied with increasing popularity in many domains [6, 12]. In these lines of research [16, 21, 25, 30], the objective is to provide a curative summary of documents, news reports and articles over a period of time to tell the story of how events evolve. Despite the successes in developing novel TDS techniques to generate timeline summaries, TDS approaches are however not effective in

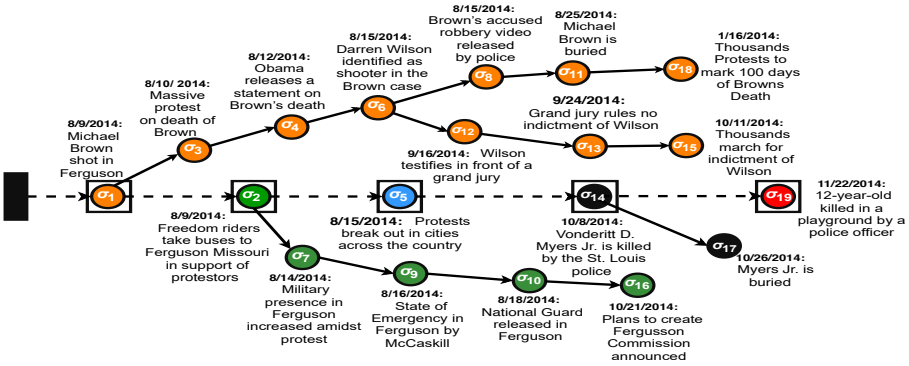


Figure 1: The StoryGraph of the *Black Lives Matter* protests from August to November 2014

handling data from social media sites such as Twitter for timeline summarization due to a number of reasons.

First and foremost, traditional documents such as news reports are well written, proof read by professionals and usually have a writing structure (i.e headlines, paragraphs, body, etc) which provides important cues for summarization [11, 22]. Tweets, on the other hand are highly unstructured. Secondly, tweets are very short, and often do not provide sufficient statistical information and context for robust similarity measures [10]. Furthermore, with the daily massive volume of tweets swamped with spam, it has become increasingly difficult to sort through the hundreds of breaking news events on Twitter and track their evolution to generate timeline summaries [11, 22]. These properties of Twitter data do not only make it difficult to apply TDS techniques, but also manifest a new research challenge for event storyline summarization.

Recent works [9, 14, 28, 29] have attempted to solve the timeline summarization problem using social media. However these models suffer from two main limitations. 1) Their output structures are usually flat timeline structures which are unable to capture the different themes in a larger story. For example, stories like the *"Black Lives Matter"* protest in Fig.1 have different themes that make up the bigger story. A flat timeline structure will connect node σ_6 to σ_7. However, these two event evolutions are totally different and have no logical connection even though they are part of the *"Black Lives Matter"* protest story. 2) The flat timeline models are incapable of presenting the complex evolution relationships between events thus resulting in incomprehensible story summaries devoid of any logical connection between the events.

Developing models to resolve these limitations will aid in the generation of comprehensible story timeline structures. Addressing these problems is non-trivial as it requires the construction of summarization timeline structures with coherent paths, while maintaining logical consistency. This will help users to understand the story in a more concise and logical manner.

To achieve this goal, we propose a novel summarization framework called StoryGraph. Given a set of tweets from the Twitter stream, StoryGraph uses an information propagation based algorithm [4, 5] to detect and extract events with high coherence from online communities in Twitter. Once these events are extracted, we carefully design high penalty metrics using temporal proximity of events, semantic contents, and user community membership. Using Fig.1 as an example, our solution identifies nodes (events) $\sigma_1, \sigma_2, \sigma_5, \sigma_{14}$ and σ_{19} as the starting events of the main themes in the story. StoryGraph further uses coherence and topic drifting metrics to build coherent timeline paths in the summary structure such that each path provides a comprehensive evolution of events related to a specific theme in the story.

While storyline summarization using social media has been attempted in previous works, to the best our knowledge, this is the first work that proposes a comprehensible summarization timeline structure that leverages: user communities, textual content and temporal metrics to model the event evolution relationships and present event summary as a graph. StoryGraph is able to identify the different aspects of a story as the story evolves.

Our main contributions can be summarized as:

- We formally define and formulate a novel graph timeline summarization structure that is capable of identifying and representing the different themes of a story.
- We design high penalty metrics to capture evolving events in communities to construct coherent paths in our summarization structure.
- We develop StoryGraph, a comprehensive summarization framework to generate summaries to tell the story of how events evolve over time. We perform extensive experiments on real-world datasets to show the prowess of our methods.

2 NOTATION AND PROBLEM DEFINITION

In this paper, we focus on timeline summarization of evolving stories using information spreading on Twitter. It is however worth mentioning that the models and formulations can be extended to other social networks such as Sina Weibo and Facebook.

We model the Twitter follower network as a graph $G_N = \langle V_N, \mathbf{E}_N \rangle$, where $V_N = \{u_1, u_2, ..., u_n\}$ represents a set of n Twitter users, and $\mathbf{E}_N \subseteq V_N \times V_N$ is the set of directed edges, and an edge $u_i \rightarrow u_j$, denotes that user u_j is a follower of user u_i on Twitter. As users post tweets, information propagates from one user to another in the network. We represent the tweet stream containing tweets as $\mathcal{D} = \{d_1, d_2, ..., d_m\}$, from which we seek to extract event set $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_k\}$ in a given time period $T_s - T_f$. Each event σ_i can be represented as $(\mathbb{W}_i, \tau_i)$, where $T_s \leq \tau_i \leq T_f$ is the timestamp of σ_i and $\mathbb{W}_i = \{w_1, w_2, ..., w_{|\mathbb{W}_i|}\}$ is the event lexis comprising of a set of words. We aim to construct a story graph to represent the development of the events in Σ.

To clearly present our problem, we firstly define some key concepts that are crucial to the proposed StoryGraph framework.

Story: A story consists of a set of events that revolve around a particular subject or concept during a specific period of time. An example of a story is the *"US presidential election"*, *"Black Lives Matter"* protest, etc. A story usually has different topic themes.
Topic Theme: A topic theme is a specific aspect/subject of a story. For example in Fig. 1, conversations about *Michael Brown* (i.e., $\sigma_1 \rightarrow \sigma_{15}, \sigma_{18}$) is one topic theme in the story graph.
Event: An event is an incident that occurs at a specific time [30], represented as $\sigma_i = (\mathbb{W}_i, \tau_i)$.

Thus, we have the hierarchy: Story →Topic Theme → Event.

More formally, we present the following definitions. Let $Sim(\sigma_i, \sigma_j)$ be a metric that returns the similarity of two events σ_i, σ_j.

*Definition 2.1 (**Related Events**).* Given a user-specified threshold $\lambda \in [0, 1]$ two events σ_i, σ_j are *related* iff. $Sim(\sigma_i, \sigma_j) \geq \lambda$. □

In our story graph, a group of related events form a topic theme. A topic theme can be thought of as the subject of a cluster of related events in the story graph.

*Definition 2.2 (**Topic Theme**).* A topic theme ϑ is a distribution of words generated by a set of *related* events. □

Intuitively, a set of related events $\Sigma' = \{\sigma_1, \sigma_2, ..., \sigma_{k'}\}$ belong to the same topic theme, and share similar words distribution.

*Definition 2.3 (**Topic Theme Node**).* Given $\Sigma' = \{\sigma_1, \sigma_2, ..., \sigma_{k'}\}$ of temporally ordered related events, the *topic theme node* is the first event σ_1 in Σ'. □

A topic theme node initiates a theme in the story graph. From Fig.1, *shooting of Michael Brown* (σ_1), is the topic theme node of the topic theme *Michael Brown*. Also, the other topic theme nodes in the story graph are $\sigma_2, \sigma_5, \sigma_{14}$ and σ_{19}.

*Definition 2.4 (**Topic Theme Path** ψ).* Given a set of related events $\Sigma' = \{\sigma_1, \sigma_2, ..., \sigma_{k'}\}$, a topic theme path is a sequence of directed edges from the topic theme node σ_1 to all events in Σ'. □

Figure 1 shows the three topic theme paths that can be found in the story graph. A topic theme path is said to be *coherent* if every pair of consecutive events are related.

*Definition 2.5 (**Coherent Score** Γ).* The coherence score of a topic theme path is a measure of the theme consistency [25] along its path of related events $\Sigma' = \{\sigma_1, \sigma_2, ..., \sigma_{k'}\}$, and is given by:

$$\Gamma(\sigma_1, \sigma_2, ..., \sigma_{k'}) = \frac{1}{|k' - 1|} \sum_{i=1}^{|k'-1|} Sim(\sigma_i, \sigma_{i+1}). \quad \square$$

*Definition 2.6 (**Timeline Path**).* A timeline path is a temporally ordered sequence of topic theme nodes in a story graph. □

A timeline path connects different topic themes in a story graph. Thus, given any evolving story, a timeline path shows the various distinct topic themes that make up the story. For example, in Fig. 1, the timeline path is: $\sigma_1 \rightarrow \sigma_2 \rightarrow \sigma_5 \rightarrow \sigma_{14} \rightarrow \sigma_{19}$.

We proceed to define a story graph in Definition 2.7, following the definition of all relevant concepts.

*Definition 2.7 (**Story Graph**).* A story graph is a directed graph $\mathbf{G} = (\Sigma^G, E)$, with a set of nodes $\Sigma^G \subseteq \Sigma = \{\sigma_1, \sigma_2, ..., \sigma_k\}$ and edge set E. The node set Σ^G of a story graph is such that $\Sigma^G = \Sigma^G_T \cup \Sigma^G_E$, and $\Sigma^G_T \cap \Sigma^G_E = \emptyset$, where Σ^G_T represents the set of topic theme nodes and Σ^G_E represents the set of event evolutions. A directed edge $\sigma_i \rightarrow \sigma_j$ from node σ_i to σ_j indicates a temporal and/or a semantic relationship between two events s.t $\forall\, \sigma_i \rightarrow \sigma_j \in E, \tau_i < \tau_j$. The edge set $E \in \mathbf{G}$ is such that $E \subseteq \Sigma^G \times \Sigma^G \setminus \left(\Sigma^G_E \times \Sigma^G_T\right)$. □

Our goal is to develop a comprehensive and coherence structure that can capture the event evolution of the various themes of a bigger story. Stories contain different topic themes that evolve as new events emerge [14, 16]. For any given story, we expect our solution to identify the main topic themes and capture how the events on each topic theme evolve. For example, given the *"Black Lives Matter"* protest (Refer to Fig.1), we expect our timeline summarization structure to identify the Fergusson protest (node σ_2) as a topic theme node and capture the series of aftermath related events (nodes $\sigma_7, \sigma_9, \sigma_{10}, \sigma_{16}$). The StoryGaph proposed in this paper presents a structural timeline summarization of events evolution on a general story as shown in Fig.1

*Definition 2.8 (**Problem Statement**).* Given a tweet stream $\mathcal{D} = \{d_1, d_2, ..., d_m\}$ containing tweets published by users in the Twitter follower network $G_N = \langle V_N, \mathbf{E}_N \rangle$ over a period $T_s - T_f$, the goal of this paper is to construct a StoryGraph $\mathbf{G} = (\Sigma^G, E)$ using the event set $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_k\}$ extracted from the Tweets such that all the topic theme paths are coherent. □

3 STORY GRAPH CONSTRUCTION

In Fig. 2 is an overview of our StoryGraph framework. The framework accepts tweets over a period and the follower network relationships as input. Recall, that structural timeline summarization involves two main steps [16, 30]: 1) extracting candidate events, and 2) using the events to build a comprehensive summarization structure. To achieve this, StoryGraph constructs propagation trees and leverages bursty tree detection [5] to extract events with high coherent word-distribution from online communities. These events are used as conduits for generating our structural timeline summaries. The next subsections present details of the steps.

3.1 Event Extraction from Communities

The goal here is to extract events from the online Twitter communities. To achieve this goal, we adopt and extend SensorTree [5], an event detection method capable of extracting coherent events using burst from online communities.

Given a set of tweets over a time period as input, we construct propagation trees [4] which effectively capture information diffusion among users in a community. The trees represent communities of online social media users who are discussing events of interest. The nodes of the trees are users in the community and the edges represent the adoption of information among users in a community.

We incorporate user community membership as a feature to increase the confidence of finding topic themes of related events for our StoryGraph construction. This enriches our feature set and helps to overcome limitations posed by relying solely on text (tweet content) [5, 26].

We design a feature vector to help us get our desired output.

*Definition 3.1 (**Event Feature Vector**).* Given an extracted event σ, its event feature vector is a tuple, $\overrightarrow{\sigma} = (U_l, \mathbb{W}, \tau)$, where:

- U_l is a list of the community membership in which the event was detected;
- $\mathbb{W}$ is a word distribution describing the event;
- τ is the timestamp indicating when the event was detected.

□

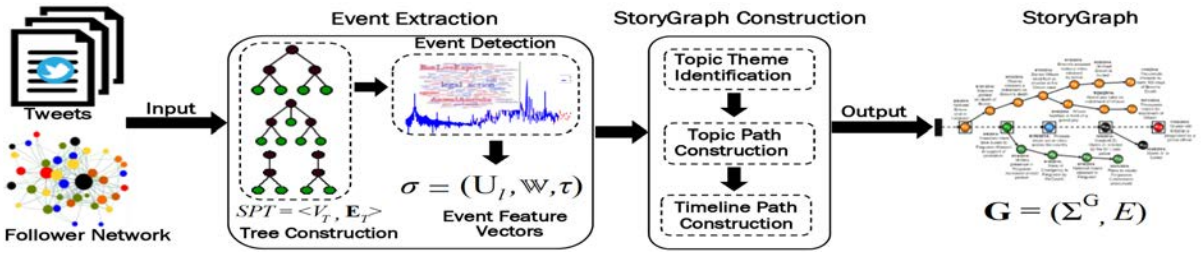


Figure 2: Framework Overview of StoryGraph

The event feature vector $\vec{\sigma}$ provides a feature rich vector that covers user community relationships, textual and temporal information on extracted events.

Given a set of tweets $\mathcal{D} = \{d_1, d_2, ..., d_m\}$ published during a given time period $T_s - T_f$, we construct semantic propagation trees [5], $SPT = \langle V_T, \mathbf{E}_T \rangle$, where $V_T = \{(u_i, [(c_i, t_i)]) \,|\, u_i \in V_T, [(c_i, t_i)] = [(c_{i_1}, t_{i_1}), (c_{i_2}, t_{i_2}) \cdots (c_{i_Q}, t_{i_Q})]\}$ represents users u_i, the contents c_i and the timestamps t_i of their posts, and the edge set is $\mathbf{E}_T = \{(u_i, [(c_i, t_i)]) \rightarrow (u_j, [(c_j, t_j)]) \,|\, u_i \rightarrow u_j \in \mathbf{E}, t_{i_1} \leq t_{j_1}\}$. We set the time window $\top = 60min$, once the burst threshold ϕ is obtained in any tree, an event description model using tensor decomposition [3] outputs word distribution and parses it into $\vec{\sigma}$. We query the node list of the community in which the event was detected to output the event feature vector $\vec{\sigma} = (U_l, \mathbb{W}, \tau)$ for that event.

The **EventExtract** block (Algorithm 1, lines 1-10) shows the pseudocode for extracting events and obtaining the event feature vector. Line 9 outputs a candidate event set $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_k\}$ which is represented by their corresponding event feature set vector $\vec{\sum} = \{\vec{\sigma_1}, \vec{\sigma_2}, ..., \vec{\sigma_k}\}$ to build a story graph. The event feature set vector $\vec{\sum}$ holds the features vectors of all the events extracted.

Algorithm 1 StoryGraph Algorithm

Input: tweets stream $\mathcal{D} = \{d_1, d_2, \ldots, d_m\}$ over $T_s - T_f$; G_N

Output: StoryGraph, $\mathbf{G} = (\Sigma^G, E)$.

1: **EventExtract:** Initialize $\mathcal{D} = \{d_1, d_2, \ldots, d_m\}$
2: → Sort $\mathcal{D}$ in temporal order
3: → Construct $SPT = \langle \boldsymbol{\mathcal{V}}_T, \mathbf{E}_T \rangle$ using extended [5]
4: → Vary $\top$ to optimum threshold and compute burst score
5: → Extract detected event set $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$
6: **for** $\sigma_i \in \Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ **do**
7: *Generate tensorized feature word-distribution* $\mathbb{W}$ [3]
8: *Extract* $\vec{\sigma} = (U_l, \mathbb{W}, \tau)$
9: *Query* $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$, $\vec{\sum} = \{\vec{\sigma_1}, \vec{\sigma_2}, \ldots, \vec{\sigma_k}\}$
10: **end for**
11: **SGConstruct** : Load $\vec{\sum} = \{\vec{\sigma_1}, \vec{\sigma_2}, \ldots, \vec{\sigma_k}\}$
12: → Sort $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ in temporal order s.t. $\tau_i \leq \tau_j$
13: → Set $\sigma_1 \in \Sigma$ as first topic theme node $\vartheta \in \Sigma_T^G$
14: **for** $\vec{\sigma} \in \vec{\sum} = \{\vec{\sigma_1}, \vec{\sigma_2}, \ldots, \vec{\sigma_k}\}$ **do**
15: *Compute* $F_{Rel}(\vec{\vartheta}, \vec{\sigma})$
16: **if** $F_{Rel}(\vec{\vartheta}, \vec{\sigma}) > \lambda$
→ Append and update $\vec{\sigma}$ with $\vec{\vartheta}$
→ *Check* **Node Merge Criterion** *to merge* (Definition 3.2)
→ Compute *PC* (Eqn. 3) *to append new child node* and update Σ_E^G
17: **else**
Set σ *as new topic theme node* and update Σ_T^G
18: **end for**
19: → *Align topic theme nodes* Σ_T^G *on time line path in chronological order*
20: **return** $\mathbf{G} = (\Sigma^G, E)$.

3.2 StoryGraph Construction

The input for our story graph construction phase **SGConstruct** (i.e, Algorithm 1, lines 11-20) is $\vec{\sum}$. To construct the story graph, we sort the candidate events feature vectors $\vec{\sigma_1}, \vec{\sigma_2}, ..., \vec{\sigma_k}$ using increasing temporal ordering such that $\tau_i \leq \tau_j$. The first event in $\vec{\sum}$ is chosen as the first topic theme node. For every incoming node, we apply a dual-component metric to detect its topic theme.

Topic Theme Identification: As described earlier, every candidate event has its own event feature vector $\vec{\sigma}$. To quantify the relatedness between an incoming event and the existing topic theme, we compute the feature relatedness F_{Rel}, between the topic theme feature vector $\vec{\vartheta}$ and the feature vector of the incoming event $\vec{\sigma}$. The feature relatedness F_{Rel} is given by:

$$F_{Rel}\left(\vec{\vartheta}, \vec{\sigma}\right) = \frac{1}{2}\left[D_{KL}(\mathbb{W}^{\vec{\vartheta}}||A) + D_{KL}(\mathbb{W}^{\vec{\sigma}}||A)\right] \times \frac{2U_l^c}{U_l^{\vec{\vartheta}} + U_l^{\vec{\sigma}}} \quad (1)$$

where $A = \frac{1}{2}(\mathbb{W}^{\vec{\sigma}} + \mathbb{W}^{\vec{\vartheta}})$ is the average of the two word distributions, and D_{KL} :

$$D_{KL}(\mathbb{W}||A) = \sum_{w \in |\mathbb{W}|} p(w|\mathbb{W}) \log \frac{p(w|\mathbb{W})}{p(w|A)} \quad (2)$$

is the *Kullback-Leibler* divergence [13] which measures distribution divergence of A from $\mathbb{W}$. U_l^c is the number of the common users in the community user list of $U_l^{\vec{\vartheta}}$ and $U_l^{\vec{\sigma}}$.

Equation 1 has two parts. The first part measures the text semantic similarity to determine if an event belongs to a particular topic theme. We utilized the *Jenson-Shannon divergence (JSD)* [19] to measure relatedness directly in $F_{Rel}(\vec{\vartheta}, \vec{\sigma})$ instead of $D_{KL}(\vec{\vartheta}||\vec{\sigma})$, because the *JSD* is a symmetrical and smoothed. The second part of the equation uses the *Sørensen-Dice coefficient* [27] to measure the community similarity between the incoming event and the topic theme community. We apply this dual-component criteria because the use of only textual content is not always reliable [7, 26]. Using community membership increases the confidence of obtaining coherence. The intuition here is that users in the same community have similar interest and are likely to have discussions on the same topic theme.

Equation 1 (F_{Rel}) is used to determine whether an incoming candidate event should be added to an existing topic theme node or not. If the relatedness $F_{Rel} \geq \lambda$, we can add the event to the related topic theme. Once we determine which topic theme an incoming event belongs to, we update the topic theme feature vector $\vec{\vartheta}$. Consequently, a topic theme vector becomes the summation of all the related event feature vectors. For example, given an existing topic theme with

topic theme feature vector $\overrightarrow{\vartheta_i} = \{(U_l^1, \mathbb{W}_1, \tau_1), ..., (U_l^k, \mathbb{W}_k, \tau_k)\}$. Using F_{Rel}, if an incoming event with feature vector $\overrightarrow{\sigma_j} = (U_l^j, \mathbb{W}_j, \tau_j)$ is found to be related to $\overrightarrow{\vartheta_i}$, then we append and update $\overrightarrow{\vartheta_i}$ with $\overrightarrow{\sigma_j}$. If the $F_{Rel} \geq \lambda$ criterion is not met, we create a new topic theme vector $\overrightarrow{\vartheta_j}$ with $\overrightarrow{\sigma_j}$ as the topic theme node.

Constructing the Topic Theme Path: After a candidate event has been identified to be related to a topic theme, the next step is to connect the candidate event to an existing node in the topic theme path of the related topic theme. One of our objectives is to reduce redundancy or duplication of events presented in the story graph. Thus we seek to merge nodes that discuss the same event. If two events are the same we merge their event feature vector as one event using the following criterion.

*Definition 3.2 (**Node Merge Criterion**).* Given a topic path ψ with a set of nodes $\sigma_1, \sigma_2, ..., \sigma_{k''}$, the merge operation combines an incoming node with an existing event in the topic theme path if: i) $\cos(\overrightarrow{\sigma}_i, \overrightarrow{\sigma}_j) = \frac{\mathbb{W}^i \cdot \mathbb{W}^j}{|\mathbb{W}^i| \cdot |\mathbb{W}^j|} \geq \beta$ (i.e., refers to the same event), where $\cos(\overrightarrow{\sigma}_i, \overrightarrow{\sigma}_j)$ is the similarity between the two word distributions, ii) and the two events occur within $\tau_j - \tau_i < \theta$. □

We set $\theta = 24hrs$. If a candidate event does not overlap with an existing node in the topic theme path, we find its parent node by computing the parent-child propensity score, *PC* in Equation 3. We design a matrix that stores the *PC* score. The *PC* score is the pairwise comparison of the incoming candidate event (potential child node) σ_i and the existing nodes (potential parent nodes) in the topic theme path. Given an incoming node σ_i that is related to $\overrightarrow{\vartheta_i}$ (i.e. has passed the F_{Rel} threshold test), and a potential parent node σ_j in the topic theme path, The *PC* score is given by:

$$PC(\sigma_i, \sigma_j) = ComSim \cdot t_d \cdot \Gamma \tag{3}$$

where $ComSim = \frac{|U_l^c|}{|U_l^{\overrightarrow{\sigma_j}}| + |U_l^{\overrightarrow{\sigma_i}}| - |U_l^c|}$ is the community similarity between the event child node and the potential parent node, the temporal distance score t_d:

$t_d(\tau_i, \tau_j) = \begin{cases} \frac{1}{(\tau_j - \tau_i)}, & \forall \tau_j - \tau_i > 0 \\ 0, & otherwise \end{cases}$ measures the time proximity between the two nodes. A bigger t_d score increases the possibility that a candidate event is connected to a parent. The third component of the *PC* score is the topic path coherence $\Gamma(\sigma_1, \sigma_2, ..., \sigma_{k'}) = \frac{1}{|k'|} \sum_{i=1}^{|k'|-1} \cos(\overrightarrow{\sigma}_i, \overrightarrow{\sigma}_j)$ measuring the theme consistency [29] along the theme path. For the candidate pair with the highest score in *PC* score matrix, we create an edge from the parent node to the incoming node in the topic theme path. The topic nodes are then aligned in chronological order in the story graph.

4 EXPERIMENTS

We present a detailed description of how the experiments were conducted, evaluated as well as the results obtained in comparison with existing state-of-the-art models.

4.1 Datasets and System Settings

We use four different and geographically diverse real world Twitter data sets for our experiments. Table 1 shows the information on our datasets[1]. With the exception of the FP-Datasets that was provided by the D2DCRC[2], all other datasets used in the work as well as the user follower network were crawled through the Twitter API.

We implemented and conducted all experiments using Python 3.2 and 3.6 on a Windows machine with 8GB Memory and a 64-bit Linux virtual machine with 6GB memory all running on an Intel(R) Core(TM) i5-4310m CPU 2.70Ghz processor.

Table 1: Dataset Description

Data sets	No. of Tweets	Network Size (No. of Nodes)
FP	200,756	12,243,427
USGH	100,000	854,626
Live Exports	3,200,000	8,202,053
USJM	5,200,000	13,202,724

4.2 Gold Standard Records (GSR)

To construct a GSR for evaluation, we search through popular news portals every day and record reported civil unrest events as ground truth. Each GSR entry represents a reported protest event with attributes such as protest event date, headline description, event summary, and news source. An event is recorded if more than three major news outlets cover a story on it. Table 2 shows sample entries in the GSR. As a preprocessing step, we remove stopwords and manually select keywords tokens to obtain a GSR keywords distribution which summarizes the real-word event. For every GSR summary, we expect the summarization models to produce some word token that matches the GSR summary.

4.3 Baseline and Comparison Methods

To evaluate the performance of our methods, we compare our solution to the following models:

1. TwitInfo [20]: explores and tracks events by aggregating tweets about the event using a peak detection algorithm.

2. ConnectingDots [25]: presents a chain structure that finds hidden connections to link coherent news articles together.

3. EventGraph [30]: uses a similarity threshold to calculate connection strength for event pairs to generate an event graph.

4. LDA+Temporal Ordering: We build a single LDA [6] topic model over the datasets and temporally order the events into a timeline chain as a baseline model. This method exemplifies the naive approach to solving the timeline structure summarization problem.

It is worth mentioning that ConnectingDots [25] and EventGraph [30] were originally designed to handle news articles. To level the playground for a fair comparison we build topical tweet clusters using the methods in [24]. These clusters are used as representative news articles. We then follow strictly the authors implementation in their published work and selected the parameter settings that yield the best results. Comparing our methods with ConnectingDots [25] and EventGraph [30] will help us further assess how models that are designed to handle news articles for summarization perform with Twitter data.

4.4 Performance Evaluation

In the first part, we evaluate the quality of the abstractive summaries generated using Recall-Oriented Understudy for Gisting Evaluation (ROGUE) metric [15] which has been demonstrated to be the most consistent with human judgement and effective for evaluating such

[1] data description available at https://github.com/vanslerry/dataDescription

[2] www.d2dcrc.com.au

Table 2: Sample coded GSR event entries by news analysts

Event ID	Date	Headline Description Summary	News Sources
FP-548662	2/05/2017	Freeport to reduce Indonesian Mining Activities	Kompass, Republica, Jarkarta Post
USGM-204201	28/03/2018	Thousands fill the streets to protest new Ghana-US defence deal	Joy News, TV3, GhanaWeb
Live Exports	18/06/2018	Protesters in Adelaide rally against live animals exports	ABC, News, Advertiser

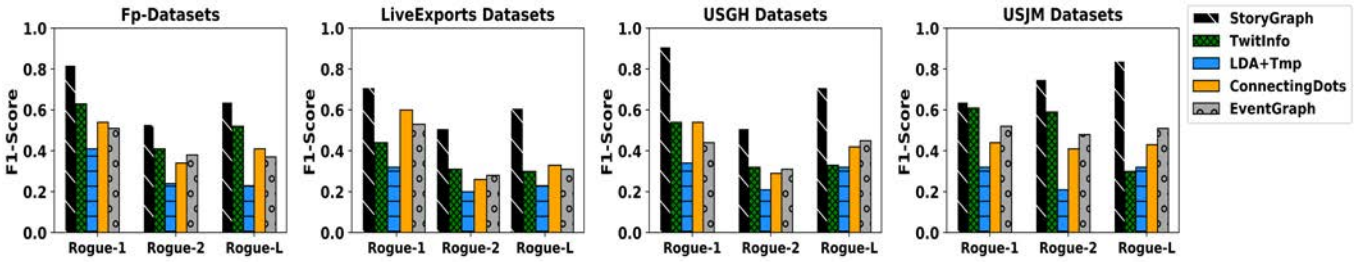


Figure 3: F1-Score on all Datasets

summaries. ROGUE measures the degree of overlap of the model's output with the ground truth. We use ROGUE-1, ROGUE-2 and ROGUE-L. For a fair comparison, we vary the vocabulary $\mathbb{W}$ size of output of the individual models, using $\mathbb{W} = 10, 20, 30, 40, 50$, to reduce bias on models that rely on word frequencies.

In the second part, we evaluate the output of the summarization structures generated by measuring the coherence between nodes in a topic path. This is achieved by computing the distance between the word vectors of the node-pairs which is given by: $C_T(\hat{c_i}, \hat{c_j}) = 1 - \frac{\hat{c_i} \cdot \hat{c_j}}{|\hat{c_i}|^2 + |\hat{c_j}|^2 - \hat{c_i} \cdot \hat{c_j}}$, where are c_i and c_j are the word vector pairs whose coherence is to be determined. This metric also indirectly measures the validity of an edge between two nodes in a path.

5 RESULTS AND DISCUSSIONS

Summary Quality: We compare the ROUGE-1, ROGUE-2 and ROGUE-L metric results of StoryGraph with 4 other comparison models. Due to space limitation, we present the averaged ROGUE metrics performance of all the model across the four datasets in Table 3. StoryGraph outperforms the comparison models in almost all cases followed by TwitInfo. LDA+Temporal is seen to be trailing in all cases. This observation is reasonable since building a single topic model over the tweet datasets will produce words distributions which are not focused on a specific subject. Even though ConnectingDots [25] and EventGraph [30] outperforms the naive approach, these models do not perform well in most cases.

Table 3: Averaged Performance comparison on all datasets

Model	ROGUE- 1		ROGUE-2		ROGUE-L	
	Precision	Recall	Precision	Recall	Precision	Recall
ConnectingDots	0.443	0.592	0.142	0.154	0.372	0.354
LDA+Temporal	0.332	0.425	0.021	0.023	0.121	0.263
EventGraph	0.522	0.53	0.258	0.253	0.254	0.248
TwitInfo	0.551	0.621	0.292	0.281	**0.513**	0.462
StoryGraph	**0.781**	**0.872**	**0.432**	**0.392**	0.503	**0.572**

This observation further affirms the claims that traditional document summarization (TDS) techniques do not perform well with Twitter data due to the unique properties of Twitter data [8, 11, 22]. We further show the F1-scores across the 4 datasets in Fig.3. StoryGraph achieves the best performance across the datasets. Recall that, StoryGraph incorporates user communities in extracting event summaries. The use of community structure contributes to this good performance. From earlier work in [4, 7], using community structure helps to extract event summaries that are coherent with fewer word intrusions. The reason is that conversations in user communities are usually focused on specific subjects of interests. Hence, there are fewer word intrusions from conversations that are not related to the topic of discussion in the community.

Structure Coherence: We evaluate three different timeline and story generation structures. A Complex Graph Structure (CGS) [30], Flat Timeline (FT) [25] and our Storygraph (SG) structure. The goal is to find out which structure presents the general information of the story in a more coherent and logical manner. We utilize the FP datasets since we have the complete evolution of events in the GSR. We present the results on the evaluation metrics in Fig.4. StoryGraph (SG) achieves the best structure coherence, followed by flat timeline (FT). Complex graph (CGS) is observed to be trailing behind in this section. While this is surprising, the possible explanation that Eventgraph [30] presents a graph structure which models the complex relationships between events in the graph. Thus, if all the nodes in the graph share some similar keywords, edges are created between them. This results in complex structures with low theme inconsistency among node pairs. A good summary structure should be simple while at the same time having the capability to model complex relationships between events, which is exactly what StoryGraph provides.

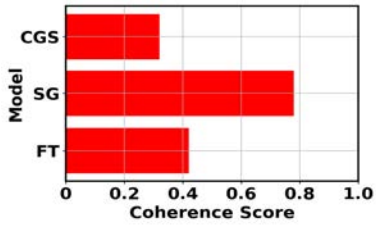


Figure 4: Coherence Comparison in Structural Summaries

6 CONCLUSION

In this paper, we have presented StoryGaph, a timeline graph summarization framework for telling the story of events that evolve over time. We formulate a novel graph timeline summarization structure capable of identifying the different themes of a story using high penalty metrics. StoryGraph constructs coherent paths to track event evolution and generates comprehensive story summaries to

tell the story of how events evolve over time. Experimental results show that StoryGraph outperforms existing models.

REFERENCES

[1] Hamed Abdelhaq, Christian Sengstock, and Michael Gertz. 2013. EvenTweet: Online Localized Event Detection from Twitter. *PVLDB* 6, 12 (2013), 1326–1329.

[2] Dhekar Abhik and Durga Toshniwal. 2013. Sub-event detection during natural hazards using features of social media data. In *Proceedings of the 22nd International Conference on World Wide Web*. ACM, 783–788.

[3] Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. 2014. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research* 15, 1 (2014), 2773–2832.

[4] Jeffery Ansah, Wei Kang, Lin Liu, Jixue Liu, and Jiuyong Li. 2018. Information Propagation Trees for Protest Event Prediction. In *PAKDD 2018*. Springer, 777–789.

[5] Jeffery Ansah, Wei Kang, Lin Liu, Jixue Liu, and Jiuyong Li. 2018. SensorTree: Bursty Propagation Trees as Sensors for Protest Event Detection. In *Web Information Systems Engineering - WISE 2018 - 19th International Conference, Dubai, United Arab Emirates, November 12-15, 2018, Proceedings, Part I*. 281–296. https://doi.org/10.1007/978-3-030-02922-7_19

[6] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[7] Jose Cadena, Gizem Korkmaz, Chris J Kuhlman, Achla Marathe, Naren Ramakrishnan, and Anil Vullikanti. 2015. Forecasting social unrest using activity cascades. *PloS one* 10, 6 (2015), e0128879.

[8] Deepayan Chakrabarti and Kunal Punera. 2011. Event Summarization Using Tweets. *ICWSM* 11 (2011), 66–73.

[9] Yi Chang, Jiliang Tang, Dawei Yin, Makoto Yamada, and Yan Liu. 2016. Timeline Summarization from Social Media with Life Cycle Models.. In *IJCAI*. 3698–3704.

[10] Ruifang He, Yang Liu, Guangchuan Yu, Jiliang Tang, Qinghua Hu, and Jianwu Dang. 2017. Twitter summarization with social-temporal context. *World Wide Web* 20, 2 (2017), 267–290.

[11] Karen Spärck Jones. 2007. Automatic summarising: The state of the art. *Information Processing & Management* 43, 6 (2007), 1449–1481.

[12] April Kontostathis, Leon M Galitsky, William M Pottenger, Soma Roy, and Daniel J Phelps. 2004. A survey of emerging trend detection in textual data mining. In *Survey of text mining*. Springer, 185–224.

[13] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics* 22, 1 (1951), 79–86.

[14] Chen Lin, Chun Lin, Jingxuan Li, Dingding Wang, Yang Chen, and Tao Li. 2012. Generating event storylines from microblogs. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 175–184.

[15] Chin-Yew Lin and Eduard H. Hovy. 2003. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In *HLT-NAACL*.

[16] Bang Liu, Di Niu, Kunfeng Lai, Linglong Kong, and Yu Xu. 2017. Growing Story Forest Online from Massive Breaking News. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 777–785.

[17] Quanchao Liu, Heyan Huang, and Chong Feng. 2013. Micro-blog post topic drift detection based on LDA model. In *Behavior and Social Computing*. Springer, 106–118.

[18] Deepa Mallela, Dirk Ahlers, and Maria Soledad Pera. 2017. Mining Twitter features for event summarization and rating. In *Proceedings of the International Conference on Web Intelligence*. ACM, 615–622.

[19] Christopher D Manning, Christopher D Manning, and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*. MIT press.

[20] Adam Marcus, Michael S Bernstein, Osama Badar, David R Karger, Samuel Madden, and Robert C Miller. 2011. Twitinfo: aggregating and visualizing microblogs for event exploration. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 227–236.

[21] Ramesh Nallapati, Ao Feng, Fuchun Peng, and James Allan. 2004. Event threading within news topics. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. ACM, 446–453.

[22] Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining text data*. Springer, 43–76.

[23] Jeffrey Nichols, Jalal Mahmud, and Clemens Drews. 2012. Summarizing sporting events using twitter. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*. ACM, 189–198.

[24] Kevin Dela Rosa, Rushin Shah, Bo Lin, Anatole Gershman, and Robert Frederking. 2011. Topical clustering of tweets. *Proceedings of the ACM SIGIR: SWSM* (2011).

[25] Dafna Shahaf and Carlos Guestrin. 2010. Connecting the dots between news articles. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 623–632.

[26] Imane Tamimi, El Khadir Lamrani, and Mohamed El Kamili. 2017. Community Detection Through Topic Modeling in Social Networks. In *International Symposium on Ubiquitous Networking*. Springer, 70–80.

[27] Vikas Thada and Vivek Jaglan. 2013. Comparison of jaccard, dice, cosine similarity coefficient to find best fitness value for web retrieved documents using genetic algorithm. *International Journal of Innovations in Engineering and Technology* 2, 4 (2013), 202–205.

[28] Han Wang and Jia-Ling Koh. 2017. Timeline Summarization for Event-Related Discussions on a Chinese Social Media Platform. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer, 579–594.

[29] Shize Xu, Shanshan Wang, and Yan Zhang. 2013. Summarizing complex events: a cross-modal solution of storylines extraction and reconstruction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1281–1291.

[30] Christopher C Yang, Xiaodong Shi, and Chih-Ping Wei. 2009. Discovering event evolution graphs from news corpora. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 39, 4 (2009), 850–863.

Firsthand Opiates Abuse on Social Media: Monitoring Geospatial Patterns of Interest Through a Digital Cohort

Duilio Balsamo
University of Turin & ISI Foundation
Turin, Italy
duilio.balsamo@unito.it

Paolo Bajardi
ISI Foundation
Turin, Italy
paolo.bajardi@isi.it

André Panisson
ISI Foundation
Turin, Italy
andre.panisson@isi.it

ABSTRACT

In the last decade drug overdose deaths reached staggering proportions in the US. Besides the raw yearly deaths count that is worrisome *per se*, an alarming picture comes from the steep acceleration of such rate that increased by 21% from 2015 to 2016. While traditional public health surveillance suffers from its own biases and limitations, digital epidemiology offers a new lens to extract signals from Web and Social Media that might be complementary to official statistics. In this paper we present a computational approach to identify a digital cohort that might provide an updated and complementary view on the opioid crisis. We introduce an information retrieval algorithm suitable to identify relevant subspaces of discussion on social media, for mining data from users showing explicit interest in discussions about opioid consumption in Reddit. Moreover, despite the pseudonymous nature of the user base, almost 1.5 million users were geolocated at the US state level, resembling the census population distribution with a good agreement. A measure of prevalence of interest in opiate consumption has been estimated at the state level, producing a novel indicator with information that is not entirely encoded in the standard surveillance. Finally, we further provide a domain specific vocabulary containing informal lexicon and street nomenclature extracted by user-generated content that can be used by researchers and practitioners to implement novel digital public health surveillance methodologies for supporting policy makers in fighting the opioid epidemic.

CCS CONCEPTS

• **Information systems → Information retrieval**; • **Applied computing → Health informatics**; • **Human-centered computing → Social media**.

KEYWORDS

Online disease monitoring, Opioid epidemic, Reddit

ACM Reference Format:
Duilio Balsamo, Paolo Bajardi, and André Panisson. 2019. Firsthand Opiates Abuse on Social Media: Monitoring Geospatial Patterns of Interest Through a Digital Cohort. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313634

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313634

1 INTRODUCTION

The opioid crisis emerged in the US from the interplay of several determinants and evolved over time through three major phases: from the abuse of drug prescriptions of late 80's [26], to heroin injecting users soaring in 2010 [31] and the very recent rise of synthetic opioids flooding the drug market [23]. The rate of overdose deaths involving opioids shows heterogeneous geographical patterns ranging from 4.9 per 100,000 inhabitants in Texas to 43.4 in West Virginia in 2016 [45]. Unfortunately, despite the effort of national agencies in monitoring the phenomenon, state-level estimates suffer biases due to unequal coverage of the surveillance system as well as intrinsic biases in counting overdose deaths that lead to low specificity of drugs involved [17, 25, 41]. Besides drug overdose maps, opioid prescription maps are provided on yearly basis by the Centers for Disease Control and Prevention (CDC), showing heterogeneous patterns also due to different prescribing policies implemented by individual states. In this perspective, the gold standard represented by official surveillance statistics has to be carefully considered in the light of the known biases of the reported numbers and estimates. A complementary way to investigate these phenomena is by field studies, through ethnographic approaches and structured interviews: sadly these methods require an exceptional effort in order to gather insights from firsthand users [28] or subjects under opioid substitution maintenance treatment [47], and are often limited to small sample sizes ranging from tens to hundreds of individuals. In this context, a digital epidemiology approach [42] aimed at integrating and complementing existing knowledge about the opioid crisis gathering information with a bottom-up unsolicited approach might be extremely valuable.

In recent years, Social Media changed the way drug users share information about opiates, give online warnings and avoid potentially toxic drug batches [21], pointing at "Reddit: the front page of Internet" [39] as a promising digital source of information. Reddit is a social content aggregation website, the 5th most popular website in the US (`Alexa.com`, Fall 2018), on which users can post, comment and vote content. It is structured in cross-referenced yet independent sub-communities (i.e. *subreddits*), centered around a variety of topics ranging from science to drugs and porn [30]. Reddit is constantly growing, with a total of almost 800 million comments in the year 2016 and almost a billion comments in the year 2017. Given the ease to register with a "throwaway" account, Reddit is often used to discuss topics otherwise considered socially unacceptable or unsuitable for the mainstream; users can actively engage with others, talking uninhibitedly about personal experiences [27], receiving back social support and even life-saving tips from sub-communities of peers [21]. However, navigating such massive platforms and finding areas of specific interest is usually cumbersome since topics

are self-organized bottom-up through users' interactions, leaving the users to find relevant topics by word of mouth or using a basic search feature.

The main contributions of this work are summarized as follows:

- Design a general purpose information retrieval algorithm able to identify regions of interest when conducting epidemiological surveillance and monitoring on social media
- Provide an open domain specific vocabulary related to opiates discussions
- Demonstrate how information disclosed by Reddit users can be used to estimate their geographical location
- Identify a novel digital cohort of individuals addressing in a pseudonymous fashion health related topics
- Provide prevalence maps of interest in opiates consumption

2 RELATED WORK

Digital epidemiology [42], also referred to as *digital disease detection* [4] and *infodemiology* [13], broadly includes the use of the Internet and digital technologies for collecting and processing health information, both in aggregate form for public health surveillance or from individuals for personal health monitoring. Data from a so called *digital cohort* might be collected with active participation of individuals, as in the case of participatory epidemiology [14], but might also be collected passively, e.g. from Social Media, as a byproduct of platforms designed for different purposes. Participatory systems have been implemented through the use of Web platforms [35] and signals collected from such systems have been shown to be useful for epidemic forecasting [49]. Data collected from the Web and Social Media have also shown to be useful for monitoring different infectious diseases [3, 5, 9, 22, 46].

Reddit has already proven to be suitable for a variety of research purposes, ranging from the study of user engagement and interactions between highly related communities [18, 48] to post-election political analyses [1]. Also, it has been useful to study the impact of linguistic differences in news titles [20] and to explore recent web-related issues such as hate speech [43] or cyberbullying [38] as well as health related issues like mental illness [8], also providing insights about the opioid epidemics [36]. It is worth to notice that since the platform is collectively generated by the users as in most Social Media, there is not a blueprint of Reddit to perimeter the area under study, and only few attempts tried to overcome this issue extracting meaningful maps or suitable embedding spaces [29, 34].

We approach the problem of selecting subreddits that are relevant for a specific topic as an information retrieval (IR) problem, where it is possible to retrieve topic-specific documents by expressing a limited set of known keywords. Language models [10, 37, 44] tackle this problem using a probabilistic approach with the idea that words that are relevant for a given topic would be more likely to appear in a relevant document. While language modeling is not a common approach for ranking documents collected from Social Media due to the inherent sparsity of the documents – e.g. for Twitter, more elaborated IR approaches are needed to resolve the sparsity of short texts for tweets [33] – subreddits, on the other hand, can be seen as very rich documents from which topic-specific word distributions can be built.

The set of keywords expressed by the user may not include some yet unknown terms that are relevant to the topic but very specific to the language models of the documents, pointing the necessity of query expansion techniques. Relevance feedback [40] and pseudo-relevance feedback [6] are common approaches for query expansion; many of these approaches use human judgment to manually select a subset of the top retrieved documents, and use them to expand the query. More recently, word embeddings [24] have been used to expand the query with terms that are semantically similar to the query terms. Techniques that are based on language modeling might incorporate term proximity information [12] to address the automatic query expansion problem, or use an approach based on Information Theory, exploiting the *Kullback-Leibler distance* for query reweighing [7] and for training local word embeddings [11].

3 DATASET AND DATA PREPARATION

A dataset [1] containing the list of all submissions and comments published in Reddit since 2007 is publicly available online [2] and is maintained monthly by adding recent entries. The dataset is not 100% complete and it contains some gaps [15], but these are very small for the years 2016 and 2017 (around 1% missing data per month). We use the union of all submissions and comments from years 2016 and 2017, for a total of 1,980,497,553 entries. Only subreddits with at least 100 entries were selected, resulting in a set of 1,973,863,886 entries with 74,810 distinct subreddits and 15,747,502 distinct users. The text of each entry is parsed and tagged using the spaCy NLP library [2] v1.9.0. For the part-of-speech tagging, a greedy averaged perceptron model was used [19]. Finally, lemmatization is applied to each POS tag; the English lemmatization data is taken from WordNet [32] and lookup tables are taken from Lexiconista [3]. After all terms are lemmatized, we select those that appear at least 100 times in the corpus, resulting in a vocabulary of 762,746 lemmas.

4 IDENTIFYING RELEVANT DOCUMENTS VIA DOCUMENT RANKING AND QUERY EXPANSION

As discussed in Section 2, the wealth of information contained in Reddit data is not readily available and has to be thoroughly mined. In this section we describe an iterative methodology of semi-automatic retrieval of documents in heterogeneous corpora, in which human intervention is as little as possible. It is worth to stress that the approach is general and fully unsupervised. However, we added a human-in-the-loop to include domain expert knowledge in the process and reach better results. On the other hand, a domain expert alone without the aid of the algorithmic pipeline for document ranking and query expansion would have been hopeless in navigating the Reddit world by hand. The steps are summarized in Algorithm 1. We start with a small set of keywords Q provided by the user, or *query* in the following. At each iteration, we select documents which are both relevant to the query and informative, and enrich the *query* terms set until we arrive to a stable list of documents and query terms. While this methodology works well on Reddit in the domain of topics related to the opioid epidemics

[1] https://files.pushshift.io/reddit/
[2] https://spacy.io/
[3] http://www.lexiconista.com/datasets/lemmatization/

(see Section 4.1), it is also sufficiently general to be used for other information retrieval tasks and might be valuable for different epidemiological research questions.

Algorithm 1: IR steps for document ranking

Input: Corpus C, query Q
Parameters: n, m, α
Initialize the vocabulary $\mathbf{V}$;
foreach *word* $w \in \boldsymbol{V}$ **do**
  calculate $p_C(w)$;
  foreach *document* $d \in C$ **do**
    calculate $p_d(w)$;
$Q_{new} \leftarrow Q$;
$K_{new} \leftarrow \emptyset$;
repeat
  $Q \leftarrow Q_{new}$;
  $K \leftarrow K_{new}$;
  $R_d \leftarrow$ Rank documents using score$(d \mid Q, C)$ (Eq. 2);
  $K_{new} \leftarrow$ top n documents in R_d;
  $R_w \leftarrow$ Rank terms using score$(w \mid K_{new})$ (Eq. 3);
  $Q_{candidate} \leftarrow$ top m terms from R_w;
  $Q_{new} \leftarrow$ manual selection of terms in $Q \cup Q_{candidate}$;
until $Q = Q_{new}$ *and* $K = K_{new}$;
Output: R_d, R_w

First, we create a general vocabulary $\mathbf{V}$ by collecting terms from the entire corpus in a bag-of-words fashion. We compute the probability of occurrence of a term w in the entire corpus C as the ratio $p_C(w) = f_C(w)/\sum_w f_C(w)$ between its raw count $f_C(w)$ in the corpus and the total number of words in the corpus. Let us also define the regularized marginal probability of occurrence of term w in a document d as

$$p_d(w) = \frac{f_d(w)}{\sum_w f_d(w) + \alpha} + p_C(w). \tag{1}$$

where $f_d(w)$ is the count of w in d. In case of corpora with very heterogeneous document sizes, the regularization term α is added to control "the size" of the language model of the documents (small documents will result in small marginal probabilities). In our experiments, we use $\alpha = 10^4$, so documents with total number of words lower than 10^4 have "flattened" probabilities in their language model. Adding $p_C(w)$ to the marginal probability of w reduces the impact of words that are rare or not present in a document, and only words that are more likely to appear in the document will impact the document ranking. The use of these two regularization terms (α and $p_C(w)$) are effective in low-count scenarios and have a small impact for words with high probability in the document or in case of documents with large language model.

With the intuition that a document will result to be relevant in the context of the query if it contains query terms more likely to appear in the document than in the general corpus, we evaluate

$$\text{score}(d \mid Q, C) = KLD_Q(p_d, p_C) = \sum_{w \in Q} p_d(w) log \frac{p_d(w)}{p_C(w)} \tag{2}$$

which is the total contribution of the query terms in the *Kullback-Leibler divergence* between the document and the whole corpus C. We consider the top n documents ranked by relevance as measured by Eq. (2) as the *set of relevant documents* K. Once n is chosen (and thus K is obtained), in order to enrich the query terms Q we assign a score to each term based on the logarithm of the likelihood ratio

$$\text{score}(w \mid K) = \sum_{k \in K} log \frac{p_k(w) + p_C(w)}{p_C(w)}. \tag{3}$$

Whenever the the maximum likelihood estimate $p_k(w)$ of a term in a relevant document is higher than its estimate in the general context $p_C(w)$ – i.e. the term is more informative in the document than in the general context – the contribution to the score will be high and positive. Conversely, whenever a term is less likely to appear in the document than in general context, its contribution to the score will be smaller, highlighting that the term is common or not relevant to the context. Considering the top m terms ranked by Eq. (3), a subset of previously unknown relevant terms is added to Q with the supervision of an expert. With the newly enriched set of query terms, the entire pipeline can be iteratively evaluated until convergence, reached if no new documents are added to the top n documents and no new relevant terms among the top m can be added to Q. These steps are summarized in Algorithm 1, and the document ranking resulting from the last iteration of the algorithm is used to select the most relevant documents.

4.1 Opioid related subreddits

Our assumption is that authors who post content in a subreddit related to a particular topic are *interested* in that topic. Therefore we consider all authors participating in threads on subreddits related to opioid consumption as those *interested* in the topic, and we discover such subreddits by applying the algorithm described in Section 4.1. After data preparation steps described in Section 3, starting from a list of opiates related keywords q = `[fentanyl, oxycodone, suboxone, morphine]` we extract the top $n = 10$ subreddits: `[suboxone, fentanyl, Opiatewithdrawal, TarkovTrading, heroinaddiction, ChronicPainPlayhouse, OpiatesRecovery, opiates, Methadone, PoppyTea]`. We then proceed with the iterative procedure of query enrichment and document ranking, considering as relevant terms only opioid drug names, i.e. chemical names (e.g. *oxycodone*), brand names (e.g. *Percocet*) and street slang (e.g. *percs*), disregarding drugs that might be abused together with opiates (like benzodiazepines) but are not in the opiates domain. The final set of *opioid related subreddits* used in this paper is K = `[fentanyl, suboxone, Opiatewithdrawal, Methadone, opiates, pillhead, lean, ChronicPainPlayhouse, heroinaddiction, OpiatesRecovery]`. All the 37,009 users who posted on such subreddits in 2016 or 2017 are then considered as interested in opiates. Figure 1 (top) shows the final ranking score for the subreddits, plotted against their size in terms of number of words.

4.2 Opioid specific vocabulary

When applied to Reddit, the query expansion approach is particularly useful in revealing how a topic is discussed on subreddits. Given the large user base of Reddit and the tendency of the users in employing slang and street names alongside proper drug names,

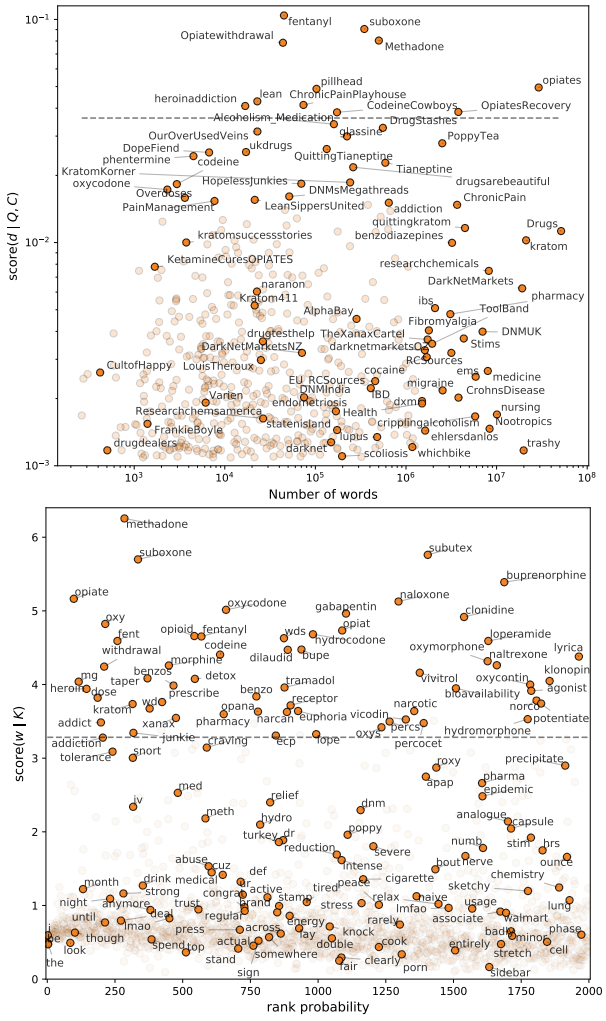


Figure 1: *Opiates subreddits* (top): individual subreddits are shown in a coordinate space of the number of words in the subreddit and the final ranking score. The subreddits above the dashed line are those selected as K opiates related subreddits. *Opiates vocabulary* (bottom): Top 2,000 terms sorted by rank probability of term in the set K. Terms above the dashed line were selected as query term candidates in the last step of the subreddit retrieval algorithm.

this method is very helpful in acknowledging alternative names of drugs, like *sub* for *suboxone* and *bth* for *black tar heroin*.

As a byproduct of the proposed methodology applied to the opiates domain, we extracted a topic-specific vocabulary by weighting each term of the vocabulary with Eq. 3. Very specific opioid-related terms (i.e. with high probability in opiates subreddits and low probability in the whole corpus) have large positive values of $score(w \mid K)$, as shown in Figure 1 (bottom), while stop-words and common terms have small score values. A total of 2,616 terms out of the original 762,746 (0.3%) have score higher than 1. The full list of vocabulary terms ranked by score is available for research purposes [4].

5 GEOLOCATING USERS ON REDDIT

Reddit does not provide any explicit information about users' location, therefore we apply three methodologies to assign a location to users. Finally, we merge the mappings in a single user-location matching.

1. Self reporting via regular expression: Reddit users often declare geographical information about themselves in submission or comment texts. We selected all texts containing the expression `'I live in'` (3,337,850 instances in 2016 and 2017) and extracted candidate expressions from the text that follows, to identify which ones represented US states and cities. We started with a set of US cities from the GeoNames database [5] with population higher than 20k, and selected only the candidate expressions that included both the city name and the state (e.g. 'Newark, California' or 'Newark, CA') to avoid confusion with cities with same name (e.g. 'Newark, New Jersey'). Once the US state for these expressions were assigned and removed from the candidate expressions, we proceeded to all US cities with population higher than 200k, selecting expressions with the name of the city and their variants (e.g. 'New York', 'Big Apple'). After assigning the corresponding US state for these expressions and removing them from the candidates, we proceeded to select the expressions with a state name on it (e.g. 'Alabama', 'California'). Among the initial set of candidate expressions, 886,919 (27%) had a state associated to it. By removing inconsistent self reporting (13,374 users who reported more than one US state) we geolocated 378,898 distinct users.

2. Self reporting via *user flairs*: In Reddit, *user flairs* are attributes (usually selected by the users) that are attached to their submissions or comments in a specific subreddit. In some subreddits flairs might be limited to a set of geographical locations (countries, states, cities and city neighborhoods), meaning that users should identify themselves with one of these locations. A user selecting a location flair is therefore considered equivalent to a user self reporting its location. We mapped the users participating in subreddits with location flairs referring to US states to their *flaired* positions. Using this approach, we mapped 206,125 users to the 51 US states (including District of Columbia) by selecting the most common among the position flairs expressed by a user.

3. Posting on subreddits specific to locations: Reddit includes subreddits discussing topics specific to geographical locations (e.g. `r/Alabama` or `r/Seattle`). The subreddit `r/LocationReddits` keeps a curated list of these local subreddits. We collected from the page corresponding to North America[6] the mappings of 2,844 subreddits to 51 US states. By assuming that a user who posts comments in one of these subreddits is likely to live that location, we estimated the position of 1,198,096 authors.

After retrieving US states positions using the three methods above, we found that about 12% of mapped users expressed multiple locations. In order to uniquely map authors and states, for location flairs and LocationReddits sources we assigned each author a unique

[4] https://github.com/ISIFoundation/WWW19OpiatesAbuseSocialMedia
[5] http://www.geonames.org/
[6] https://www.reddit.com/r/LocationReddits/wiki/faq/northamerica

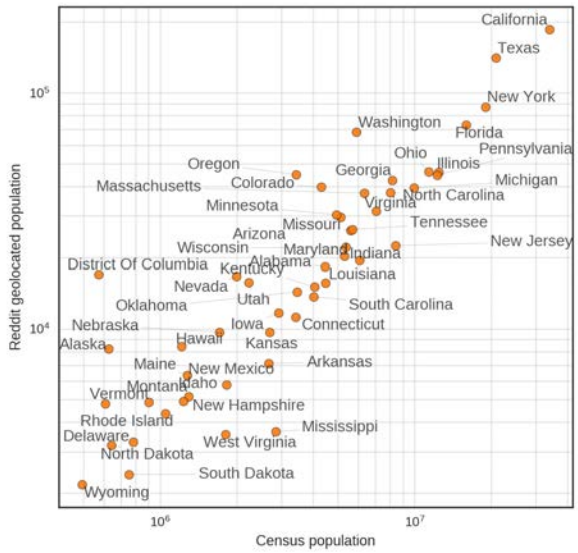


Figure 2: *Reddit geolocated population*: scatter plot of the number of geolocated Reddit users and census population.

location, it being the most frequent among the ones expressed by the author. We discarded authors whose most frequent location was not unique. This resulted in 194,008 authors retrieved via location flairs (5.9% loss) and 1,077,516 via LocationReddits (1.4% loss).

We evaluated Pearson's r correlation between the log of the 2000 US Census population and the log of the population assigned to the same US states using the three methodologies. Results are in good agreement for all sources, with r= 0.85, 0.91 and 0.86 for respectively User flairs, Regular expression and LocationReddits, and all p-values below 1e-12.

Finally, we merged the information from all three sources in a unique location for each author. We considered the regular expression technique to be the most reliable due to its unambiguous self reporting nature, resulting in the highest correlation with census data. We proceeded in the merging process by first assigning the authors their regular expression location, if present. If missing, we assigned them their position from the joint information of location flairs and LocationReddits by summing the occurrences of locations expressed in the two sources and verifying the uniqueness of the most frequent location. Although some approaches have been proposed to geolocate users using language models [16], we rely on a conservative approach with the aim of reducing misclassification, considering only explicit geographical information directly provided by the authors. The full set of users geolocated using the above methodology consists of 1,408,388 users, with state representativeness in the order of 5.5 Reddit users per thousand U.S. residents (median value among all U.S. states). Although we acknowledge a potential bias due to heterogeneities in Reddit population coverage and users demographics, the number of Reddit users has good linear correlation of r = 0.89 and p-value below 1e-12 with census population (Figure 2).

Region	Division	Opiates authors	Reddit authors	Interest prevalence
Northeast	Middle Atlantic	1,186	154,418	768.05
Northeast	New England	455	69,132	658.16
Midwest	East North Central	1,082	172,902	625.79
Midwest	West North Central	424	92,931	456.25
South	East South Central	457	63,269	722.31
South	South Atlantic	1,656	242,470	682.97
South	West South Central	1,079	177,856	606.67
West	Mountain	793	119,742	662.26
West	Pacific	1,894	315,668	599.10

Table 1: *Interest prevalence by US regional division*: number of opiates authors, total number of authors, and interest prevalence per 100,000 individuals measured on Reddit.

6 PREVALENCE OF INTEREST IN OPIATES

Conversations in opiates related subreddits branch off in many topics, mostly regarding opiates usage, dosages, interactions with other substances, safe practices and withdrawal, usually with a personal perspective. Users share their health and addiction status and provide support among each other. In general, they share a common and firsthand interest in opiates experiences. Thus, the number of authors participating in the conversations in opiates related subreddits (as identified in Section 4) is not to be considered as a crude number of opiates users and addicts but it is rather to be considered as a proxy of users personally interested in opiates in the broadest sense.

Using the geographical information of Reddit population estimated in Section 5 and having identified the opiates authors in Section 4.1, we were able to evaluate the *opiates interest prevalence* at the US state level as the fraction of geolocated users engaged in opiates subreddits and the total estimated population.

We mapped to US states the 9,026 geolocated users who posted in opioid related subreddits, equivalent to 24% of the opiates authors, for a mean interest prevalence of 636 per 100,000 Reddit users (CDC data for 2016 reported an age-adjusted rate of overdose deaths of 19.8 per 100,000 , and opioid prescription rate of 66.5 prescriptions dispensed per 100 persons). The areas of greater interest according to our estimates is the South Region (Table 1), with high prevalence for Mississippi, Arkansas, Louisiana and the highest measured value of 1,180 interested users per 100,000 population in West Virginia (Figure 3, green map). Middle Atlantic and New England states like Pennsylvania, New Jersey and Rhode Island are also largely involved, showing high interest rates ranging between 850 to 900 individuals per 100,000. In line with official statistics about drugs overdose deaths, West North Central states are those with the lowest interest rate measured on Reddit, ranging from 341 per 100,000 in Nebraska to 510 per 100,000 in Minnesota.

We confronted the estimated interest prevalence with official statistics from the Centers for Disease Control and Prevention [7] grasping different angles of the opioid crisis. In particular we focused

[7] https://www.cdc.gov/drugoverdose/data/index.html

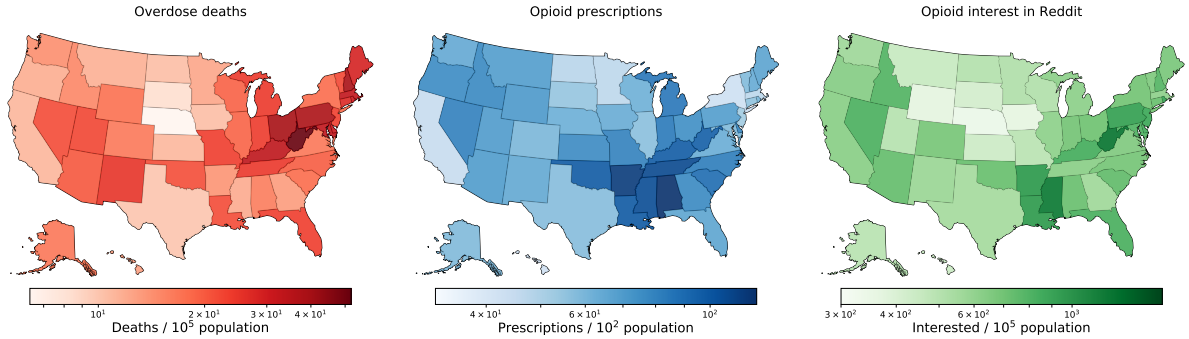


Figure 3: *US states distribution maps:* choropleth maps representing the overdose deaths rate for 2016 (red), the opioid prescription rate for 2016 (blue), the opioid interest rate in Reddit for 2016 and 2017 (green).

on opioid drug overdose deaths rates and retail opioid prescribing rates, both regarding 2016 (the most recent available dataset at the time of writing), shown in Figure 3 in red and blue respectively. These two phenomena seem fairly uncorrelated, with a Pearson's correlation of 0.068 (p-value of 0.637). It is worth to stress that those "gold-standard" data are the only ones provided by CDC that allows for comparisons between different states: although the counting of drugs overdose deaths includes every drug and is not broken down by drug type, state-level estimates of opiate-related overdose deaths are affected by heterogeneities in the surveillance system. On the other hand, official statistics about prescribing rates that includes both appropriate prescriptions and drug abuse are affected by different prescription policies in place at different states.

The interest prevalence shows fairly high positive linear correlations with CDC rates, respectively r = 0.45 (p-value = 8.4e-04) with the opioid overdose deaths rate and r = 0.506 (p-value = 1.6e-04) with the retail opioid prescribing rate. These correlations suggest that the signal of interest in opiates measured on Reddit partially explains the observed phenomena around opioid epidemics measured by the standard surveillance system. Moreover, we trained a linear regression model to fit the estimated prevalence using drug overdose death rates and prescriptions rates as features and predicted new values of interest prevalence, resulting in an higher correlation of r = 0.655 (p-value = 1.7e-07) with the estimated interest prevalence. This result confirms that we are sensing a broad signal, tied to drug overdose deaths and opioid prescriptions but probably accounting for more complex aspects of the phenomenon.

Leveraging the geolocated cohort, we also evaluated the temporal variation of interest prevalence between 2016 and 2017 broken down by state. According to Figure 4, the interest prevalence decreased only in 8 states while in general we observe that in areas with a good coverage of opiates-related users (namely, California, Texas, New York, Florida), the interest prevalence increased by 10% to 20%. It is worth to stress that, at the time of writing, no official data about 2017 drug overdose deaths and associated trends are available, highlighting the tremendous potential of a digital epidemiology approach to gather timely insights about hard-to-reach information of health-related topics at the population level.

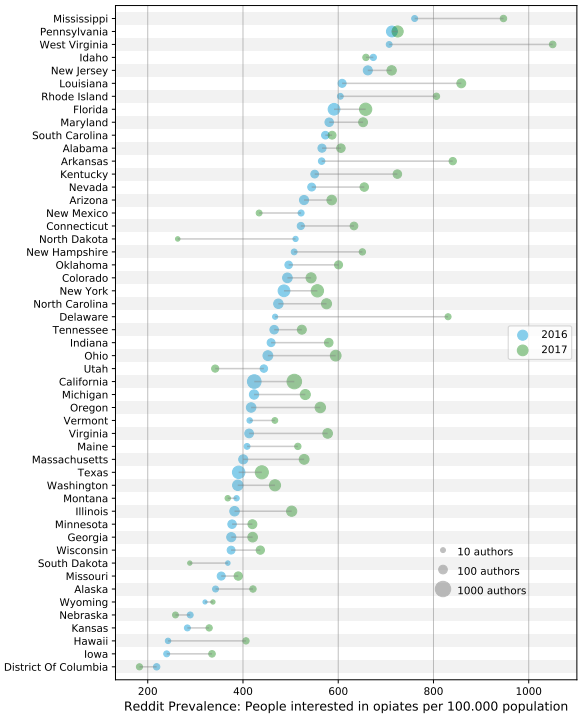


Figure 4: *Opioids interest prevalence:* number of Reddit authors per 100,000 Reddit population. Prevalence values for years 2016 and 2017 are reported on the x-axis. The size of the bubbles is proportional to the number of Reddit authors.

7 CONCLUSION

This study provides an analysis of Reddit content related to personal opiates abuse in the period of 2016 and 2017. Starting from almost 2 billion posts over 74k distinct subreddits, we applied a general information retrieval algorithm to identify specific subreddits of interest thus selecting 37,009 users that show an explicit interest in the topic. 1.5 million pseudonymous Reddit users were geolocated at US state level by looking at the content they generated in the platform. The number of mapped users for each state are in good agreement with census data, with some differences in terms of coverage. Such cohort might represent a biased yet valuable digital

observatory on several social, political and health-related topics. The prevalence of opiates interest extracted with the presented approach shows a complementary perspective to official surveillance, and its geographical heterogeneity partially encodes signals from opioid prescribing rates and drug overdose deaths.

ACKNOWLEDGMENTS

DB acknowledges support from the Lagrange Project and CRT Foundation (http://isi.it/en/lagrange-project/project).

REFERENCES

[1] Michael Barthel. 2016. How the 2016 presidential campaign is being discussed on Reddit. Retrieved November 5, 2018 from http://www.pewresearch.org/fact-tank/2016/05/26/how-the-2016-presidential-campaign-is-being-discussed

[2] Jason Baumgartner. 2015. I have every publicly available Reddit comment for research. 1.7 billion comments @ 250 GB compressed. Any interest in this? – r/datasets. Retrieved November 5, 2018 from https://redd.it/3bxlg7

[3] John S Brownstein, Shuyu Chu, Achla Marathe, Madhav V Marathe, Andre T Nguyen, Daniela Paolotti, Nicola Perra, Daniela Perrotta, Mauricio Santillana, Samarth Swarup, Michele Tizzoni, Alessandro Vespignani, Anil Kumar S Vullikanti, Mandy L Wilson, and Qian Zhang. 2017. Combining participatory influenza surveillance with modeling and forecasting: three alternative approaches. *JMIR public health and surveillance* 3, 4 (2017).

[4] John S Brownstein, Clark C Freifeld, and Lawrence C Madoff. 2009. Digital disease detection - harnessing the Web for public health surveillance. *New England Journal of Medicine* 360, 21 (2009), 2153–2157.

[5] John S Brownstein, Clark C Freifeld, and Lawrence C Madoff. 2009. Influenza A (H1N1) virus, 2009 –online monitoring. *New England Journal of Medicine* 360, 21 (2009), 2156–2156.

[6] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. 1995. Automatic query expansion using SMART: TREC 3. *Proceedings of The third Text REtrieval Conference (TREC-3)* (1995), 69–69.

[7] Claudio Carpineto, Renato De Mori, Giovanni Romano, and Brigitte Bigi. 2001. An information-theoretic approach to automatic query expansion. *ACM Transactions on Information Systems (TOIS)* 19, 1 (2001), 1–27.

[8] Munmun De Choudhury and Sushovan De. 2014. Mental Health Discourse on Reddit: Self-Disclosure, Social Support, and Anonymity. In *Proceedings of the Eighth Intl Conf. on Weblogs and Social Media, ICWSM 2014*. The AAAI Press.

[9] Rumi Chunara, Jason R Andrews, and John S Brownstein. 2012. Social and news media enable estimation of epidemiological patterns early in the 2010 Haitian cholera outbreak. *The American journal of tropical medicine and hygiene* 86, 1 (2012), 39–45.

[10] W Bruce Croft and John Lafferty. 2013. *Language modeling for information retrieval*. Vol. 13. Springer Science & Business Media.

[11] Fernando Diaz, Bhaskar Mitra, and Nick Craswell. 2016. Query expansion with locally-trained word embeddings. *arXiv preprint arXiv:1605.07891* (2016).

[12] Liana Ermakova, Josiane Mothe, and Elena Nikitina. 2016. Proximity relevance model for query expansion. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. ACM, 1054–1059.

[13] Gunther Eysenbach. 2009. Infodemiology and infoveillance: framework for an emerging set of public health informatics methods to analyze search, communication and publication behavior on the Internet. *Journal of medical Internet research* 11, 1 (2009).

[14] Clark C Freifeld, Rumi Chunara, Sumiko R Mekaru, Emily H Chan, Taha Kass-Hout, Anahi Ayala Iacucci, and John S Brownstein. 2010. Participatory epidemiology: use of mobile phones for community-based health reporting. *PLoS medicine* 7, 12 (2010), e1000376.

[15] Devin Gaffney and J Nathan Matias. 2018. Caveat Emptor, Computational Social Science: Large-Scale Missing Data in a Widely-Published Reddit Corpus. *arXiv preprint arXiv:1803.05046* (2018).

[16] Bo Han, Paul Cook, and Timothy Baldwin. 2014. Text-based twitter user geolocation prediction. *Journal of Artificial Intelligence Research* 49 (2014), 451–500.

[17] H. Hedegaard, M. Warner, L. Paulozzi, and R. Johnson. 2014. Issues to Consider When Analyzing ICD-10 Coded Data on Drug Poisoning (Overdose) Deaths. National Center for Health Statistics and the National Center for Injury Prevention and Control.

[18] Jack Hessel, Chenhao Tan, and Lillian Lee. 2016. Science, AskScience, and BadScience: On the Coexistence of Highly Related Communities.. In *Proc. of the 10th Intl AAAI Conf. on Web and Social Media, ICWSM 2016*. The AAAI Press, 171–180.

[19] Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1373–1378.

[20] Benjamin D Horne and Sibel Adali. 2017. The impact of crowds on news engagement: A Reddit case study. (2017). arXiv:1703.10570v2

[21] Fernando Alfonso III. 2017. How a Reddit forum has become a lifeline to opioid addicts in the US. https://www.theguardian.com/society/2017/jul/19/opioid-addiction-reddit-fentanyl-appalachia.

[22] Taha A Kass-Hout and Hend Alhinnawi. 2013. Social media in public health. *Br Med Bull* 108, 1 (2013), 5–24.

[23] Andrew Kolodny, David T Courtwright, Catherine S Hwang, Peter Kreiner, John L Eadie, Thomas W Clark, and G Caleb Alexander. 2015. The prescription opioid and heroin crisis: a public health approach to an epidemic of addiction. *Annual review of public health* 36 (2015), 559–574.

[24] Saar Kuzi, Anna Shtok, and Oren Kurland. 2016. Query expansion using word embeddings. In *Proceedings of the 25th ACM international on conference on information and knowledge management*. ACM, 1929–1932.

[25] Michael G Landen, Stuart Castle, Kurt B Nolte, Mary Gonzales, Luis G Escobedo, Barbara F Chatterjee, Karen Johnson, and C Mack Sewell. 2003. Methodological issues in the surveillance of poisoning, illicit drug overdose, and heroin overdose deaths in New Mexico. *American journal of epidemiology* 157, 3 (2003), 273–278.

[26] Pamela TM Leung, Erin M Macdonald, Matthew B Stanbrook, Irfan A Dhalla, and David N Juurlink. 2017. A 1980 letter on the risk of opioid addiction. *New England Journal of Medicine* 376, 22 (2017), 2194–2195.

[27] Lydia Manikonda, Ghazaleh Beigi, Huan Liu, and Subbarao Kambhampati. 2018. Twitter for Sparking a Movement, Reddit for Sharing the Moment:# metoo through the Lens of Social Media. *arXiv preprint arXiv:1803.08022* (2018).

[28] Sarah G Mars, Philippe Bourgois, George Karandinos, Fernando Montero, and Daniel Ciccarone. 2014. "Every 'never' I ever said came true": Transitions from opioid pills to heroin injecting. *Int 'l Journal of Drug Policy* 25, 2 (2014), 257–266.

[29] Trevor Martin. 2017. community2vec: Vector representations of online communities encode semantic relationships. In *Proceedings of the Second Workshop on NLP and Computational Social Science*. 27–31.

[30] Alexey N Medvedev, Renaud Lambiotte, and Jean-Charles Delvenne. 2018. The anatomy of Reddit: An overview of academic research. *arXiv preprint arXiv:1810.10881* (2018).

[31] Thomas Michel and Joseph Loscalzo. 2015. Shifting patterns of prescription opioid and heroin abuse in the United States. *Mars* 372 (2015), 241–8.

[32] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[33] Nasir Naveed, Thomas Gottron, Jérôme Kunegis, and Arifah Che Alhadi. 2011. Searching microblogs: coping with sparsity and document quality. In *Proc. of the 20th ACM int'l conf. on Information and knowledge management*. ACM, 183–188.

[34] Randal S Olson and Zachary P Neal. 2015. Navigating the massive world of reddit: Using backbone networks to map user interests in social media. *PeerJ Computer Science* 1 (2015), e4.

[35] Daniela Paolotti, Annasara Carnahan, Vittoria Colizza, Ken Eames, John C. Edmunds, Gabriel Gomes, Carl E Koppeschaar, Moa Rehn, Ronald Smallenburg, Clément Turbelin, Sander P. van Noort, and Alessandro Vespignani. 2014. Web-based participatory surveillance of infectious diseases: the Influenzanet participatory surveillance experience. *Clinical Microbiology and Infection* 20, 1 (2014), 17–21.

[36] Albert Park and Mike Conway. 2017. Towards Tracking Opium Related Discussions in Social Media. *Online journal of public health informatics* 9, 1 (2017).

[37] Jay M Ponte and W Bruce Croft. 1998. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 275–281.

[38] Tazeek Bin Abdur Rakib and Lay-Ki Soon. 2018. Using the Reddit Corpus for Cyberbully Detection. In *Asian Conference on Intelligent Information and Database Systems*. Springer, 180–189.

[39] Reddit. 2018. Reddit: The front page of the internet. https://www.reddit.com/.

[40] Joseph John Rocchio. 1971. Relevance feedback in information retrieval. In *The Smart retrieval system: experiments in automatic document processing*. Englewood Cliffs, NJ: Prentice-Hall, 313–323.

[41] Christopher J Ruhm. 2017. Geographic variation in opioid and heroin involved drug poisoning mortality rates. *American journal of preventive medicine* 53, 6 (2017), 745–753.

[42] Marcel Salathé, Linus Bengtsson, Todd J. Bodnar, Devon D. Brewer, John S. Brownstein, Caroline Buckee, Ellsworth M. Campbell, Ciro Cattuto, Shashank Khandelwal, Patricia L. Mabry, and Alessandro Vespignani. 2012. Digital Epidemiology. *PLOS Computational Biology* 8, 7 (07 2012), e1002616.

[43] Haji Mohammad Saleem, Kelly P Dillon, Susan Benesch, and Derek Ruths. 2017. A web of hate: Tackling hateful speech in online social spaces. *arXiv preprint arXiv:1709.10159* (2017).

[44] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*. Vol. 39. Cambridge University Press.

[45] Puja Seth, Lawrence Scholl, Rose A Rudd, and Sarah Bacon. 2018. Overdose deaths involving opioids, cocaine, and psychostimulants -United States, 2015–2016. *American Journal of Transplantation* 18, 6 (2018), 1556–1568.

[46] Megha Sharma, Kapil Yadav, Nitika Yadav, and Keith C Ferdinand. 2017. Zika virus pandemic - analysis of Facebook as a social media health information platform. *American journal of infection control* 45, 3 (2017), 301–302.

[47] Luis Sordo, Gregorio Barrio, Maria J Bravo, B Iciar Indave, Louisa Degenhardt, Lucas Wiessing, Marica Ferri, and Roberto Pastor-Barriuso. 2017. Mortality

risk during and after opioid substitution treatment: systematic review and meta-analysis of cohort studies. *bmj* 357 (2017), j1550.

[48] Chenhao Tan and Lillian Lee. 2015. All who wander: On the prevalence and characteristics of multi-community engagement. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1056–1066.

[49] Qian Zhang, Nicola Perra, Daniela Perrotta, Michele Tizzoni, Daniela Paolotti, and Alessandro Vespignani. 2017. Forecasting seasonal influenza fusing digital indicators and a mechanistic disease model. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. 311–319.

Learn2Clean: Optimizing the Sequence of Tasks for Web Data Preparation

Laure Berti-Equille [1,2]
[1]ESPACE-DEV/IRD, UMR 228, IRD/UM/UG/UR, Montpellier, France
[2]Aix Marseille Université, Université de Toulon, CNRS, LIS, DIAMS, Marseille, France
laure.berti@ird.fr

ABSTRACT

Data cleaning and preparation has been a long-standing challenge in data science to avoid incorrect results and misleading conclusions obtained from dirty data. For a given dataset and a given machine learning-based task, a plethora of data preprocessing techniques and alternative data curation strategies may lead to dramatically different outputs with unequal quality performance. Most current work on data cleaning and automated machine learning, however, focus on developing either cleaning algorithms or user-guided systems or

argue to rely on a principled method to select the sequence of data preprocessing steps that can lead to the optimal quality performance of. In this paper, we propose Learn2Clean, a method based on Q-Learning, a model-free reinforcement learning technique that selects, for a given dataset, a ML model, and a quality performance metric, the optimal sequence of tasks for preprocessing the data such that the quality of the ML model result is maximized. As a preliminary validation of our approach in the context of Web data analytics, we present some promising results on data preparation for clustering, regression, and classification on real-world data.

CCS CONCEPTS

• **Information systems → Data extraction and integration**; • **Computing methodologies → Reinforcement learning**.

KEYWORDS

Principled data preprocessing; Data cleaning; Q-Learning; Reinforcement learning

ACM Reference Format:
Laure Berti-Equille [1,2]. 2019. Learn2Clean: Optimizing the Sequence of Tasks for Web Data Preparation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313602

1 INTRODUCTION

Through the use of data mining, statistics, and machine learning, data scientists and researchers can discover relevant patterns and gain crucial and actionable knowledge from data. In the context of Web data analytics, it is necessary to extract, transform, and structure the data and also adapt the data volume, data format, and distributional characteristics in order to better suit the underlying assumptions and constraints of statistical machine learning (ML) models so that they can be effectively applied to the preprocessed input data. It is essential to detect glitches (aka data quality problems) [2] because erroneous data limit the performance of statistical methods and cause misleading results [3, 11].

Web Data can be big, noisy, unreliable, highly imbalanced, heterogeneous, and evolve over time [6]. In practice, Web data analytics can suffer from a wide range of data anomalies such as missing values, inconsistencies, outlying data, duplicates, etc. The results of classification, clustering or predictive models obtained from the data that were preprocessed inappropriately are misleading and can cause inadequate decision-making. To mitigate the impact of data anomalies and thus provide high quality analytical results, a first necessary step is to preprocess data appropriately.

Data preprocessing has been acknowledged as a primary sequence of tasks to correct the negative effects and bias that data anomalies may produce on output results as demonstrated in various contexts [9, 12]. Common data preprocessing tasks mainly include: data extraction and formatting, selection of features, data normalization, treatment of missing values and outliers, deduplication, inconsistency detection (wrt some rules and constraints), data replacement, and pattern enforcement. Although there is a vast number of data preprocessing methods available to achieve these tasks in the context of Web mining [12–16], the selection and orchestration of the sequence of tasks often remains *ad hoc*, difficult, and with no guarantee that it is optimal. An important question is: How do we actually measure the quality of data preprocessing for a given analytics task? This is a challenging problem with combinatorial issues if we consider the infinite "curation space" with hundreds of possible techniques, each with various parameter settings, and multiple possibilities in the combination and the ordering of the cleaning tasks, and even worse when we consider multiple iterations of certain tasks.

For example, in the context of Web usage data mining, data characterizing the interaction of Internet-users with Web sites are collected and analyzed in order to provide personalized Web-pages or enhance search engines. To prepare such data before analysis, one may define a preprocessing strategy that starts with a distance-based deduplication for Web user and session identification, then the removal of outliers, and ends with a regression-based imputation of missing values; another expert may use a totally different sequence of methods with another ordering and add the re-iteration of outlier detection and removal. Ultimately, the two preprocessed datasets may be so different that the conclusions from classification or clustering results may also differ radically. Similarly in the context of Web content mining, the information presented on Web

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313602

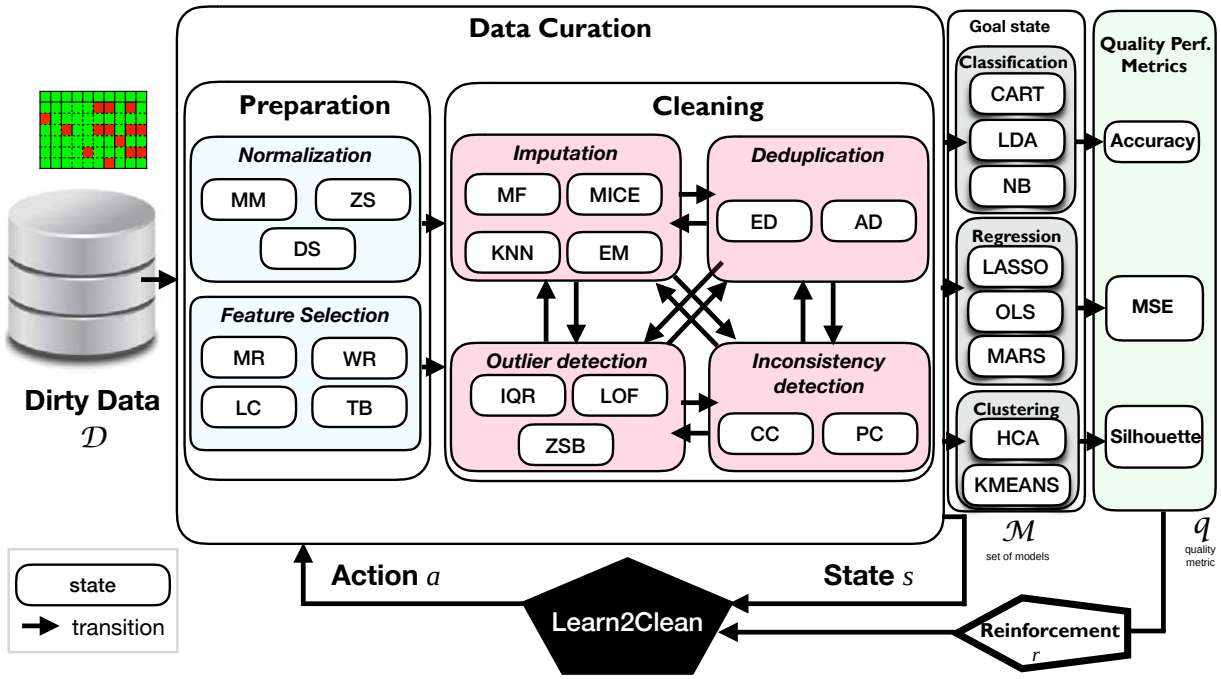


Figure 1: Learn2Clean Architecture

pages combining text and multimedia content is used for page classification, page summarization, entity extraction, and semantic processing. The data need to be filtered because many objects present on Web pages are not relevant with the stated objectives (e.g., digital marketing or advertising). Text content by itself is noisy: for example, sentences contain many words that have a poor influence on the semantic content and various strategies for text preprocessing, Web content cleaning and transformation into feature vectors may lead to very different mining results.

There are several methods that optimize single data cleaning operations based on downstream accuracy metrics (see [18] for a survey) and also recent approaches such as [17], data transformation learning [22], and AutoML line of work [19, 20] in the area of automated machine learning. However, the problem of selecting the optimal data preprocessing pipeline has been understudied.

The focus of our work is first to study the impact of various sequences of data preprocessing methods on the output results of some representative ML models; second, we propose a novel method that learns by reinforcement the sequence of tasks that can optimize a preselected quality performance metric for the considered ML model. In our study, we consider the following tasks of (1) selecting features, (2) normalizing data, dealing with (3) missing values and (4) outliers, (5) deduplicating, and (6) consistency checking, since these methods are the most frequently used for preparing data.

2 OVERVIEW OF LEARN2CLEAN

Our overall approach can be summarized in Figure 1. We consider a ML pipeline with a particular goal such as classification, regression or clustering. The ML pipeline includes: (i) the data $\mathcal{D}$; (ii) the feature set $\mathcal{X}$; (iii) the learning algorithm $\mathcal{M}$, along with the objective function $\mathcal{L}$ that maximize a quality performance metric q; and, possibly, (iv) its (trained) parameters $\mathbf{w}$. The ML pipeline can thus be formalized in terms of a space $\Theta = (\mathcal{D}, \mathcal{X}, \mathcal{M}, \mathbf{w})$. Input data preprocessing is strongly dependent on the given practical scenario. For example, additional constraints while preprocessing the data have to be considered to conform the data with the ML model requirements (e.g., normalization, discretization, deduplication, missing value imputation, extreme values replacement, etc.). Typically, these constraints can be nevertheless accounted for in the definition of the optimal curation strategy. In particular, we characterize them by assuming that $\mathcal{D}'$ is the dataset resulting from a sequence of data modifications in the space of all possible data cleaning transformations of $\mathcal{D}$ such as $D' \in \Phi(\mathcal{D})$ with $\Phi(\mathcal{D}) = \{\phi(\mathcal{D}) | \forall \phi\}$. The problem can be formalized as follows:

Given a ML pipeline $\theta \in \Theta$ and the data sets resulting from all preprocessing alternatives $\Phi(\mathcal{D})$, the goal of preprocessing can be characterized in terms of an objective function $\mathscr{P}_q \in \mathbb{R}$ which evaluates how effective the preprocessing alternative D' is with respect to a given quality metric q. The result of the optimal preprocessing strategy can be thus given as:

$$D^\star = \underset{D' \in \Phi(\mathcal{D})}{\operatorname{argmax}} \mathscr{P}_q(D', \mathcal{X}, \mathcal{M}, \mathbf{w}). \quad (1)$$

More intuitively, the problem we address is the following: *Given a dataset as input $\mathcal{D}$, a ML pipeline θ to apply to the input dataset, a quality performance metric q, and the space of all possible data preparation and cleaning strategies $\Phi(\mathcal{D})$: Find the dataset D' in $\Phi(\mathcal{D})$ that maximizes the quality metric q.*

To solve this problem, we propose Learn2Clean a system based on Q-learning that learns in an on-line fashion with interleaving learning and execution. The intuition is the following. In the early

stages of execution, when little is known, it is important to explore and try some data curation actions when their consequences on the output quality performance of the given ML model are still unknown. Rewards are given when the quality performance improves. Later on, the system may want to almost always choose the action that gets highest rewards; when there is little information to be gained, the value of learning can be negligible. This is modeled with a Q-learning function that assigns a probability of being chosen for each possible action in a given state such that it should tend to choose actions with higher Q-values, but should sometimes select lower Q-value actions.

3 Q-LEARNING

Q-learning [10] is a form of model-free reinforcement learning, where the system only needs to know what state exists next and what actions are possible in each state. An estimated value Q-value is assigned to each state. When the system visits a state and receives a reward, the Q-value of that state is updated accordingly such as:

$$Q(s,a) = R(s,a) + \gamma max_{a'} Q(s',a') \tag{2}$$

where the value of taking action a in state s is the immediate reward denoted $R(s,a)$ for the state-action pair (s,a), plus the value of the best possible state-action pair for the successor state. The γ parameter has a range of 0 to 1 ($0 \leq \gamma < 1$). If γ is closer to zero, the system will tend to consider only immediate rewards. If it is closer to 1, the system will consider future rewards with greater weight, willing to delay the reward. We can use Eq. (2) to update the Q-value of a state-action pair as a reward r' is observed:

$$Q_{t+1}(s,a) = r' + \gamma max_{a'} Q(s',a'). \tag{3}$$

r' is the reinforcement received when the system selected the a action in the state s to move to the state s'. In our context, we represent data preprocessing and curation strategy as a graph: each state is a node representing a preparation or cleaning task that generates a "cleaner" version of the original dirty dataset. A sequence of tasks is represented by a link between two nodes (i.e., between a cleaning task and the next cleaning task). In the architecture presented in Figure 1, we can start with any normalization method as a task (e.g., ZS, MM, SM or DS) or feature selection method (e.g., CFS, WR, LC)[1], and move to the next task choosing one method amongst the ones proposed either for deduplication, inconsistency checking, outlier detection or imputation, until the system reaches the goal state. We define the goal state and a particular quality metric we want to maximize denoted $G(q)$. For example, as shown in Figure 2, the goal state for a LASSO regression applied to the input dataset is the state of preprocessing that minimizes the mean squared error, MSE. To reach this node as a goal, we associate a reward value to each link between nodes. The nodes that lead immediately to the goal node have an instant reward of r_{max}. There may be no link between nodes (encoded as -1 in the Reward R-matrix represented in Figure 2). In this example, particular cleaning paths can be avoided or task precedence can be enforced: e.g., after normalization, there is feature selection, then imputation (either using MICE or KNN), and then outlier detection (either with LOF or IQR), and again imputation. This figure represents a particular graph instantiation of Learn2Clean architecture presented in Figure 1. The final goal

[1] Acronyms of the methods are described in Section 3.1.

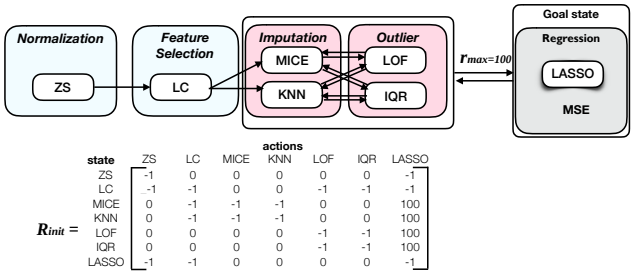


Figure 2: Simplified example of state-action graph with initial reward matrix.

state is ML model-dependent since the dataset should be prepared and curated with respect to the specific requirements of the ML model considered for the analysis. The goal state in the example of Figure 2 is reached after a maximum number of 100 iterations when LASSO regression is executed after either imputation or outlier detection such that MSE (the considered quality performance metric) is minimal.

Reinforcement via reward. The reinforcement function provides, for each new state s' reached from state s, a signal r' that can be a reward or punishment, this signal takes the value $\{-1\}$ when the state s is not linked to the new state s' or a value in $[0,1]$, which is used by the update function in Eq. (3) to adjust the Q-value function associated with the state-action pair. The reward r' is computed as follows:

$$r' = \beta(Norm(s,q_m) - Norm(s',q'_m)) \tag{4}$$

where $Norm(s,q_m)$, $Norm(s',q'_m)$ are the normalized quality metric values q and q' in [0,1] for the considered ML model m and

$$\beta = \begin{cases} -r_{max} & if\, Norm(s,q_m) - Norm(s',q'_m) < 0 \\ r_{max} & if\, Norm(s,q_m) - Norm(s',q'_m) > 0 \end{cases}$$

with r_{max}, the user-defined maximum reward value. This penalizes the next data curation task when it provides a lower quality performance metric compared to the ones obtained after the current cleaning task.

Selection of the next curation tasks. The probability of selecting the highest Q-value action should increase over time and this can be defined by the Boltzmann distribution as follows:

$$P(a|s) = \frac{e^{Q(s,a)/k}}{\sum_j e^{Q(s,a_j)/k}}. \tag{5}$$

The k parameter (often referred to as temperature) controls the probability of selecting non-optimal actions. If k is large, all actions will be selected fairly uniformly. If k is close to zero, the best action will always be chosen. We begin with k large and gradually decrease it over time. Each edge contains an instant reward value, as shown in the initial reward matrix R_{init} given as example in Figure 2.

Learn2Clean will learn through experience in an unsupervised way. It will explore from state to state until it reaches the goal. Each exploration consists of the system moving from the initial state to the goal state. Each time, it arrives at the goal state, the program goes to the next exploration. Learn2CLean algorithm is given in Algorithm 1 where the system learns from experience. Each exploration is equivalent to one training session. In each training

session, the system explores the curation graph (represented by matrix R), receives the reward (if any) until it reaches the goal state (lines 6–8). The purpose of the training is to enhance the decision of the system, represented by matrix Q to get more training results in a more optimized matrix Q. In this case, if the matrix Q has been enhanced, instead of exploring around, and going back and forth to the same tasks, the system will find the fastest route to the goal state.

Algorithm 1 Learn2CLean Algorithm

```
1. INPUT γ and reward matrix R
2.        ML model m, quality metric q
3. Initialize Q matrix
4. FOR each exploration
5.     Select the initial state
..     Apply the corresponding curation method to generate
..     a "cleaner" dataset, and apply m to it
6.     WHILE {goal G(m,q) has not been reached}
7.         Select one among all possible actions with prob.
..             in Eq.(5) for the current state
8.         Using this possible action, move to the next state
9.         Get max Q-value for this next state based on all
..             possible actions
10.        Compute Eq.(2)
11.        Set the next state as the current state.
12.    END_WHILE
13. RETURN the list of actions with highest Q-values and
..         optimal quality metric and the final curated dataset
```

The algorithm finds the actions with the highest reward values recorded in matrix Q (lines 9–10) and returns the sequence of states from the initial state to the goal state where the quality metric is optimal (either minimized for regression MSE or maximized for classification accuracy or clustering silhouette) and the ultimately cleaned dataset to be used for the considered ML model (line 13).

Why Q-learning? In model-based reinforcement methods, the model learns the transition probability $T(s_1|(s_0, a))$ from the pair of current state s_0 and action a to the next state s_1. However, model-based algorithms become impractical as the state space and action space grows, which is typically the case with the space of data cleaning strategies. Moreover, the transition probability and the dynamics of the system is not given a priori. These are the reasons why model-free algorithms are more adequate for our problem. We choose Q-Learning instead of other model-free reinforcement learning methods (like Monte-Carlo or SARSA methods) because Q-learning will not wait until the end of the episode (i.e., cleaning task) to update the expected future reward estimation, it will only wait until the next time step to update the value estimates and may not use the same policy to choose the next action. This makes it agile and performant compared to other model-free reinforcement learning methods.

3.1 Data preprocessing methods

As shown in Figure 1, various types of representative preprocessing techniques were used in our study with the long-term goal to provide an extensible library for data cleaning and preparation.

- **Normalization**. Three methods were applied to numerical data: min-max (MM), Z-score (ZS), and decimal scale normalization (DS);
- **Feature selection**. Four methods were tested to select an optimal set of features, namely: based on a user-defined acceptable ratio of missing values (MR), removing collinear features (LC), using a wrapper subset evaluator (WR) [8], and a tree-based classifier for feature selection (TB);
- **Imputation**. Four imputation methods were used to impute missing values: two distance-based imputation methods: *Expectation-Maximization* (EM), *K-NN* [1], Multiple Imputation by Chained Equations (*MICE*) [5], and replacement by the most frequent value (MF);
- **Outlier detection**. Outliers were detected using three methods including: a statistic-based approach, Inter Quartile Range (IQR), the Z-score-based method (ZSB) [7], and a density-based approach (Local Outlier Factor, LOF) [4]. Outliers were replaced using one of the previous imputation methods;
- **Deduplication**. Exact duplicate (ED) and approximate duplicate (AD) detection methods were used. In the later case, various comparison functions were used to measure the similarity distance between pairs of records to detect duplicates based on the user-defined threshold (e.g., Jaro-Winkler or Levenshtein distance);
- **Consistency checking**. Two methods based on constraint discovery and checking (CC) and pattern checking (PC) were used to identify inconsistencies.

3.2 ML Models and Quality Performance

Our goal is to study the impact of data preprocessing pipeline selection and how it can affect the output results of statistical ML methods. In particular for Web usage data analysis, we considered the following ML models and corresponding quality performance metrics.

- **Clustering**. We used two clustering methods: HCA (Hierarchical Clustering) and K-means with silhouette metric to evaluate the quality of the clustering as $silhouette = (b(i) - a(i))/max\{a(i), b(i)\}$ with $a(i)$ the average distance between i and all other data within the same cluster and $b(i)$ the smallest average distance of i to all points in any other cluster, of which i is not a member;
- **Regression**. We used three regression methods: LASSO (Least Absolute Shrinkage and Selection Operator), OLS (Ordinary Least Squares Regression), and MARS (Multivariate Adaptive Regression Splines) and as quality metrics, we used MSE (Mean Squared Error) to measure the robustness of regression as $MSE = \sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2/N$; and
- **Classification**. We used three classification methods: LDA (Linear Discriminant Analysis), NB (Naive Bayes), and CART (Classification and Regression Trees) with accuracy metric as $accuracy = (TP + TN)/(TP + TN + FP + FN)$ with TN and FN true and false negatives and TP and FP true and false positives respectively in the classification result.

Our approach is not limitative and other ML models, quality metrics or preprocessing methods could be added and used by Learn2CLean.

4 EXPERIMENTS

We have performed preliminary experiments to compare the preprocessing alternatives (described in Section 3.1) for the classification, regression, and clustering methods (described in Section 3.2) using real-world datasets from Kaggle[2] repository as shown in Table 1.

All methods and Learn2CLean are implemented in Python (Jupyter Notebook 5.5, Python 3.6.5). Source code and datasets used in our study are available at: https://github.com/LaureBerti/Learn2Clean.

Table 1: Real-world datasets

Datasets	[#]Att.	[#]Rows	Clustering	Regression	Classification
Google Play Store Apps	13	10.8k	X		X
Google Play Store Users	5	64.3k	X		
House Prices	81	1.46k	X	X	X

4.1 Compared Methods

We have compared the results of data preprocessing strategy returned by Learn2CLean to four types of data cleaning: (1) random selection (*RAND*) of a sequence of one (or none) curation task from each block of possible actions for normalization, feature selection, imputation, outlier, duplicate and inconsistency detection and removal (as depicted in Figure 1) with no particular order; (2) manual cleaning by a human expert in Data Science (*DS_EXP*); (3) one automated ML approach (*AUTO*) obtained by MLBox[3] including cleaning and preprocessing; and (4) *NO-PRE* when no data preparation/curation is achieved. In particular, we have compared the quality metrics of the ML models applied to the preprocessed datasets obtained from these various preparation strategies, respectively MSE for regression, accuracy for classification, and silhouette for clustering[4]. In the case of random preprocessing, we averaged the results in terms of quality and time over the total number of execution runs.

We used default parameter settings for all the ML models with no particular hyper-parameter optimization and future work will be to integrate this important aspect and provide an in-depth experimental analysis and a fair comparison with other AutoML approaches [20] (such as AutoSklearn[5]). In these approaches, the user has to predefine the pipeline of default preprocessing methods to be used for feature selection (e.g., drift thresholding for MLBox or variance thresholding for AutoSklearn) and imputation techniques (e.g., One-Hot encoding for AutoSklearn). They do not include approximate duplicate or inconsistency detection/correction. Feature preprocessing is a single transformer which implements, for example, feature selection or transformation of features into a different space (i.e., PCA). The AutoML preprocessors do not test and select the optimal preprocessing method for a given dataset and their core added-value is on hyper-parameter optimization of the ML models.

In average, it takes 1.55 seconds for Learn2CLean to enumerate all possible cleaning strategies given the 19x19 Q-matrix for the 18 possible preprocessing methods. Then, each strategy as a sequence of curation tasks is executed with a duration depending on the dataset size, the number of glitches to detect and correct, and the time complexity of each task. Ultimately, the strategy with best quality metric is selected by Learn2CLean. In the last columns, Table 2 presents the pipeline of curation selected by Learn2CLean and the overall execution time.

4.2 Clustering

Clustering analysis was performed using K-Means (KMEANS) and Hierarchical clustering (HCA) methods. The number of clusters that maximizes the silhouette for each clustering method was automatically selected to perform the clustering. Table 2 presents the silhouette measures obtained by KMEANS and HCA applied to the Learn2CLean-preprocessed vs no-preprocessing, random, manually, and AutoML curated datasets. We can observe that Learn2CLean and *AUTO* show the best silhouette values for the three datasets, close or better to what an expert would obtain in reasonable time: less than one minute for Google Play Store Apps dataset and from 15 to 40 seconds for the other datasets.

4.3 Classification

For classification, the observations were split into a training and test set where 70% of data was used for training and 30% for testing. The model was fit to the training set, and the fitted model was used to predict the responses for the observations in the testing set. Classification methods cannot be executed and accuracy cannot be computed in presence of missing values, which is the case for House Prices dataset where 3 variables have from 80% to 99% missing values (MiscFeature, Fence, PoolQC). The classification methods require first feature selection based on the ratio of missing values for example, then imputation for other missing values, and another feature selection to avoid collinearity. Consequently, *RAND* and *NO-PRE* are not applicable (NA) and *DS_EXP* curation is the best strategy for this dataset because the preprocessing pipeline is quite complex with reiteration of feature selection methods before and after handling missing values. For this set of experiments, we initialized the Q-matrix of Learn2CLean to avoid re-iteration of methods from the same block. In other words, feature selection methods are mutually exclusive in the same pipeline; similarly only one imputation method can be applied. This explains the relatively low accuracy of Learn2CLean in this case and the need of expertise. However, for Google Play Store Apps classification, *AUTO* and Learn2CLean the best accuracy values for NB and and LDA respectively; not surprisingly, *NO-PRE* and *RAND* give the worst accuracy values.

[2]https://www.kaggle.com/datasets
[3]https://mlbox.readthedocs.io/
[4]For technical details on the quality metrics: https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics
[5]https://github.com/automl/auto-sklearn/tree/master/autosklearn/pipeline/components/data_preprocessing

Table 2: Experimental results for clustering, classification, and regression applied to three real-world datasets

Dataset	Model (metric)	*NO_PRE*	*RAND*	*DS_EXP*	*AUTO*	Learn2CLean	Learn2CLean Best strategy	Time (seconds)
Google Play Store Apps	HCA (Silhouette)	**0.965**	0.775	**0.963**	**0.965**	**0.968**	LC→MF→HCA	55.6
	KMEANS (Silhouette)	0.668	0.798	0.652	**0.808**	**0. 801**	LOF→AD→KMEANS	46.1
	LDA (Accuracy)	0.071	0.085	0.734	0.766	**0.796**	MICE→CART	15.5
	NB (Accuracy)	0.175	0.158	0.679	**0.786**	0.722	WR→IQR→MICE→NB	13.5
Google Play Store Users	HCA (Silhouette)	**0.965**	0.775	0.878	0.804	**0.965**	HCA	14.8
	KMEANS (Silhouette)	0.417	0.429	**0.676**	0.587	**0.655**	MR→AD→KMEANS	39.5
House Prices	HCA (Silhouette)	0.793	0.714	**0.846**	**0.844**	**0.844**	MM→ED→HCA	32.2
	KMEANS (Silhouette)	0.644	0.609	0.632	**0.674**	**0.678**	KMEANS	28.3
	CART (Accuracy)	NA	NA	**0.611**	0.345	0.449	MM→AD→CART	58.7
	NB (Accuracy)	NA	NA	**0.663**	0.457	0.518	LC→MF→NB	20.1
	LASSO (MSE)	> 1	> 1	0.165	0.138	**1.44E-12**	KNN→ZSB→LASSO	24.2
	MARS (MSE)	0.175	0.192	**0.163**	NA	0.231	IQR→MICE→MARS	134.5
	OLS (MSE)	> 1	> 1	0.176	0.137	**2.20E-25**	KNN→ZSB→OLS	45.1

4.4 Regression

For LASSO and OLS regression, we can observe that Learn2CLean has the lowest MSE for House Prices dataset and it outperfoms the other curation approaches. For *RAND* and *NO-PRE*, MSE values are not reliable (>1) because of high collinearity of the dataset variables.

In conclusion, the results of our limited set of experiments is promising and has opened several directions for research and improvements of Learn2CLean algorithm and the experimental set-up. In particular, we plan further experiments with semi-synthetic data where noise injection is carefully controlled in order to study more complex and intricate distributions of various glitch types and patterns [3] (combining duplicates, inconsistencies, missing and outlying values) and their impact on the robustness of our approach. We also plan to combine AutoML hyper-optimization with Learn2CLean extended library of preprocessing methods and Q-learning based selection of the best preparation strategy.

5 CONCLUSIONS

We proposed an approach based on Q-learning that identifies the optimal data preprocessing strategies for representative ML models applied to various real-world datasets. To the best of our knowledge, Learn2CLean is the first approach that leverages reinforcement learning for data preparation. Although there is no universally adequate data preprocessing procedure, we were able to learn hidden features for selecting the optimal data preprocessing for a given dataset and a given ML task. Our preliminary results show that an instantiation of the proposed framework based on Q-Learning is efficient and a promising direction for automating data curation. As future work, we plan to explore a wider spectrum of datasets from diverse applications of Web content extraction and mining, and extend the set of data preprocessing and ML methods to test Learn2CLean in a wider range of settings.

ACKNOWLEDGMENTS

The authors would also like to thank the anonymous referees for their valuable comments and helpful suggestions. This research is supported by the French National Agency ANR under Grant No.: 18-CE23-0002 QualiHealth.

REFERENCES

[1] Batista G., Monard M.C., et al. (2002). A Study of K-Nearest Neighbour as an Imputation Method, HIS, vol. 87, pp.48.
[2] Berti-Equille L., Loh J.M., Dasu T. (2015). A masking index for quantifying hidden glitches. Knowledge and Information Systems, 44, pp.253–277.
[3] Berti-Equille L., Dasu T., Srivastava D. (2011). Discovery of complex glitch patterns : A novel approach to Quantitative Data Cleaning. Proc. ICDE, pp.733–744.
[4] Breunig M.M., Kriegel H.-P., Ng R.T., Sander J. (2000). LOF: identifying density-based local outliers, ACM SIGMOD Record, vol. 29(2), pp.93–104.
[5] Buuren S., Groothuis-Oudshoorn K. (2011). MICE: Multivariate imputation by chained equations in R, J. of Statistical Software, vol. 45, num. 3.
[6] Berti-Equille L., Scannapieco M. (2016). Quality of Web Data (Chapter). In the 2nd Edition of the book Data Quality: Concepts, Methodologies and Techniques, Springer, 2016
[7] Filzmoser P., Garrett R.G., Reimann D. (2005). Multivariate outlier detection in exploration geochemistry, Computers & Geosciences, vol. 31, pp.579–587.
[8] Kohavi R., John G. H. (1997). Wrappers for feature subset selection, Artificial intelligence, vol. 97, pp.273–324.
[9] Serrano Balderas E.C., Berti-Equille L., Armienta Hernandez M.A., Grac C., (2017). Principled Data Preprocessing: Application to Biological Aquatic Indicators of Water Pollution. Proc. of the DEXA-BIOKDD'17 workshop.
[10] Watkins J.C.H. (1989). Learning from Delayed Rewards, PhD Thesis, University of Cambridge, England.
[11] Zaveri A., Maurino A., Berti-Equille L. (2014). Web Data Quality: Current State and New Challenges, Int. J. Semant. Web Inf. Syst.,10(2):1552-6283, IGI Global.
[12] Suresh R.M., Padmajavalli R. (2006). An Overview Of Data Preprocessing In Data and Web Usage Mining, Proc. of the 1st International Conference on Digital Information Management, pp.193–198.
[13] Vellingiri J., Chenthur Pandian S. (2011). A Novel Technique for Web Log Mining with Better Data Cleaning and Transaction Identification, Journal of Computer Science, pp.683–689.
[14] Chang-bin J., Li C. (2010). Web Log Data Preprocessing Based On Collaborative Filtering, Proc. of IEEE 2nd International Workshop On Education Technology and Computer Science, pp.118–121.
[15] Zheng L., Hui Gui H., Li F. (2010). Optimized Data Preprocessing Technology For Web Log MiningâĂİ, Proc. of IEEE International Conference On Computer Design and Applications (ICCDA), pp. VI-19-VI-21.
[16] Sudheer Reddy K., Kantha Reddy M., Sitaramulu V. (2013). An effective data preprocessing method for Web Usage Mining, Proc. of the 2013 International Conference on Information Communication and Embedded Systems (ICICES).
[17] Bilalli B., Abelló A. (2018). PRESISTANT: Learning based assistant for data preprocessing, Arxiv CoRR abs/1803.01024.
[18] Chu X., Ilyas I. F., Krishnan S., Wang J. (2016). Data Cleaning: Overview and Emerging Challenges. Proc. of the 2016 International Conference on Management of Data (SIGMOD'16), pp.2201–2206.

[19] Feurer M., Hutter F. (2018). Towards Further Automation in AutoML, Proc. of ICML 2018 AutoML Workshop.
[20] Feurer M., Klein A., Eggensperger K., Springenberg J., Blum M., Hutter F. (2015). Efficient and Robust Automated Machine Learning, Advances in Neural Information Processing Systems 28, December, pp.2962–2970.
[21] Vanschoren J., van Rijn J. N., Bischl B., Torgo L. (2014). OpenML: Networked science in machine learning, ACM SIGKDD Explorations Newsletter, 15(2):49âĂŞ-60.
[22] Wu B., Knoblock C. A. (2016). Maximizing Correctness with Minimal User Effort to Learn Data Transformations. IUI 2016, pp.375-384.

Global Vectors for Node Representations

Robin Brochier*
Université de Lyon, Lyon 2, ERIC EA3083
Digital Scientific Research Technology
robin.brochier@univ-lyon2.fr

Adrien Guille*
Université de Lyon, Lyon 2, ERIC EA3083
adrien.guille@univ-lyon2.fr

Julien Velcin
Université de Lyon, Lyon 2, ERIC EA3083
julien.velcin@univ-lyon2.fr

ABSTRACT

Most network embedding algorithms consist in measuring co-occurrences of nodes via random walks then learning the embeddings using Skip-Gram with Negative Sampling. While it has proven to be a relevant choice, there are alternatives, such as GloVe, which has not been investigated yet for network embedding. Even though SGNS better handles non co-occurrence than GloVe, it has a worse time-complexity. In this paper, we propose a matrix factorization approach for network embedding, inspired by GloVe, that better handles non co-occurrence with a competitive time-complexity. We also show how to extend this model to deal with networks where nodes are documents, by simultaneously learning word, node and document representations. Quantitative evaluations show that our model achieves state-of-the-art performance, while not being so sensitive to the choice of hyper-parameters. Qualitatively speaking, we show how our model helps exploring a network of documents by generating complementary network-oriented and content-oriented keywords.

CCS CONCEPTS

• **Computing methodologies** → **Learning latent representations**; • **Information systems** → *Information retrieval.*

KEYWORDS

Representation learning; network embedding; matrix factorization

ACM Reference Format:
Robin Brochier, Adrien Guille, and Julien Velcin. 2019. Global Vectors for Node Representations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313595

1 INTRODUCTION

Networks are ubiquitous. The Web is a large-scale network of resources, social media help developing broad online social networks [6], the scientific literature forms a vast network of documents, from which one can derive a network of co-authors [16], *etc.* Understanding and exploring these networks involves solving tasks like node classification or link prediction. Efficiently solving these tasks via machine learning requires meaningful representations of the nodes.

The usual approach is to learn node representations using techniques originally devised for word embedding, based on the distributional hypothesis [13]. The analogy between word embedding and network embedding makes sense, because of the similarities in some of the statistical properties of networks and language [11]. DeepWalk [11], arguably the most popular network embedding algorithm, consists in extracting sequences of nodes, akin to sentences, via truncated random walks and then learning node representations based on the Skip-Gram model [9], using the hierarchical softmax approximation. Node2Vec [5] builds on DeepWalk and suggests another strategy for extracting node sequences via biased truncated random walks. It learns the node representations based on the Skip-Gram model, using the negative sampling approximation (SGNS). Metapath2vec [3] adapts DeepWalk to heterogeneous networks. Other variants based on SGNS are generalized into a common frame in [12].

Dealing with large-scale networks requires low-complexity algorithms. An issue with these algorithms is that SGNS scales linearly in the size of the corpus of node sequences. GloVe [10], the main alternative of SGNS, hasn't been investigated yet for network embedding, even though it scales sub-linearly in the size of the corpus. Still, GloVe is limited in the sense that it ignores non co-occurrence, as opposed to SGNS, which could result in less relevant node representations.

In this paper, we address these two issues and propose a matrix factorization approach for network embedding, inspired by GloVe. Our contributions are the following:

- we present a general model for network embedding, that consists in factorizing a thresholded co-occurrence matrix. By formulating a regression problem on all positive entries and randomly sampled zero entries, this model can take both co-occurrence and non co-occurrence into account, while preserving a competitive time complexity;
- we show how to extend this general model to networks of documents, by jointly learning word, document and node representations;
- we quantitatively assess the performance of the general model on well-known networks, against recent baselines. Not only we show that it outperforms GloVe by a very large margin on several datasets, but we also show that it outperforms on-par with recent network embedding algorithms;
- we quantitatively and qualitatively show that our model brings interesting innovation to deal with network of short

*These authors contributed equally to the work.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313595

documents. We show how to leverage the extension of our model to explore such networks by suggesting keywords.

The rest of the paper is organized as follows. In Section 2 we survey related work. We present in details our general model, discuss its relationship to other models, and show how to deal with networks of short documents by incorporating text into the model in Section 3. Next, in Section 4, we present a thorough experimental study, where we assess the performance of our model following the usual evaluation protocol on a node classification task for well-known networks. We also discuss hyper-parameter sensitivity, and evaluate, quantitatively speaking, the extended model for networks of documents. In Section 5, we present a case study that illustrates the recommendation of keywords with the extended model. Lastly, we conclude this paper and provide future directions in Section 6.

The code for both our model and the evaluation procedure are made publicly available[1].

2 RELATED WORK

The quality and informativeness of data representation greatly influence the performance of machine learning algorithms. For this reason, a lot of efforts are devoted to devising new ways of learning representations [1]. Word embedding, *i.e.*, the task of learning representations of words, is tightly connected to the task of learning representations of nodes, *i.e.* network embedding. In this section, we first cover important works related to word embedding and then survey recent developments in network embedding.

2.1 Word Embedding

The distributional hypothesis is the basis for word embedding. It states that distributional similarity and meaning similarity are correlated, which allows learning representation of words based on the contexts in which they occur, the context of a word being co-occurring words [13]. Co-occurrences are observed by sliding a window over a large corpus.

The Skip-Gram [9] model learns word representations by maximizing the log-likelihood of a multiset of co-occurring word pairs, C:

$$\sum_{(w_i, w_j) \in C} \log p(w_j|w_i). \tag{1}$$

The conditional probability is given by the softmax function, parameterized by the word vectors:

$$p(w_j|w_i) = \frac{e^{u_i \cdot v_j}}{\sum_W e^{u_i \cdot v_w}}. \tag{2}$$

This formulation is impractical because of the cost of computing the denominator. For this reason, two variants are introduced in [9]. Skip-Gram with Hierarchical Softmax (SGHS) uses a binary tree to approximate $p(w_j|w_i)$ and speed-up learning. Skip-gram with Negative Sampling (SGNS) redefines $p(w_j|w_i)$ to make it easier to compute:

$$p(w_j|w_i) = \frac{1}{1 + e^{-u_i \cdot v_j}} = \sigma(u_j \cdot v_j). \tag{3}$$

[1] https://github.com/brochier/gvnr

The objective becomes maximizing the following log-likelihood:

$$\sum_{(w_i, w_j) \in C} \Big(\log \sigma(w_j \cdot w_i) + \sum_{k=1}^{K} \mathbb{E}_{w_k \sim q(w_k)} \big[\log \sigma(-w_k \cdot w_i) \big] \Big). \tag{4}$$

It boils down to a classification task that consists in distinguishing the pairs of co-occurring words in C from random pairs of words, *i.e.* the negative samples. For each $(w_i, w_j) \in C$, K negative samples (w_i, w_k) are drawn, with $q(w_k) \propto \text{frequency}(w_k)^{\frac{3}{4}}$.

The GloVe [10] model learns word representations by factorizing the word-word co-occurrence matrix. Its objective is minimizing the reconstruction error, only for positive entries of X:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} f(x_{ij}) \big(u_i \cdot v_j + b_i^U + b_j^V - \log(x_{ij}) \big)^2, \tag{5}$$

where $f(x_{ij})$ is the following weighting function, that notably reduces the importance of rare co-occurrences and filter-out zero entries:

$$f(x_{ij}) = \begin{cases} \big(x/x_{\max}\big)^{\frac{3}{4}} & \text{if } x < x_{\max}, \\ 1 & \text{otherwise.} \end{cases} \tag{6}$$

The authors show that the distribution of x_{ij} follows a power-law and that the time complexity of GloVe is $O(|C|^{\frac{1}{\alpha}})$, α being the exponent of the power-law. Because α is usually larger than 1 for text, $\frac{1}{\alpha}$ becomes smaller than 1. Hence, this model has a better time complexity than Skip-gram with Negative Sampling, which runs in $O(|C|)$.

In [8], Levy and Goldberg note that the introduction of a bias for each target/context word adds an extra degree of freedom to the GloVe model as compared to the Skip-gram model.

2.2 Network Embedding

Even though the distributional hypothesis originated in linguistics and is naturally leveraged for word embedding, Perozzi *et al.* establish the connection with network embedding. To do so, they show that the frequency at which nodes appear in short random walks follows a power-law distribution, like the frequency of words in language [11].

They propose DeepWalk, that consists in applying skip-gram with hierarchical softmax on a corpus of node sequences, deemed equivalent to sentences, generated with truncated random walks [11]. For some specific tasks, the representations learned with DeepWalk offer large performance improvements. Thus, many subsequent works focus on modifying or extending DeepWalk. Node2vec replaces random walks with biased random walks, in order to better balance the exploration-exploitation trade-off, arguing that the added flexibility in exploring neighborhoods helps learning richer representations [5]. Dong *et al.* address the heterogeneous network representation learning problem. They propose Metapath2vec [3], a modification of DeepWalk based on meta-path-based random walks to generate sequences of heterogeneous nodes. They also suggest learning the representation with negative sampling instead of hierarchical softmax. In [18], Yang *et al.* prove that skip-gram with hierarchical softmax can be equivalently formulated as a matrix factorization problem. They then propose Text-Associated DeepWalk (TADW), to deal with networks of documents. TADW consist

Table 1: Notations.

Notation	Definition
n	Number of nodes.
d	Embedding dimension.
C	Corpus of co-occurring nodes.
$U \in \mathbb{R}^{n \times d}$	Target node embeddings.
$V \in \mathbb{R}^{n \times d}$	Context node embeddings.
$l \in \mathbb{N}_*^+$	Window size for observing co-occurrence.
$X \in \mathbb{R}^{n \times n}$	Co-occurrence matrix.
n_i	Number of nodes that co-occur with node i.

in constraining the factorization problem, with a pre-computed representation of documents via LSA [2].

Qiu *et al.* [12] provide the theoretical connections between Skip-Gram based network embedding algorithms and the theory of graph Laplacian. This allows them to unify DeepWalk, LINE [15] (which they prove to be a special case of DeepWalk), PTE and Node2Vec, all with with negative sampling, into the matrix factorization framework.

3 MODEL FORMULATION

In this section, we present our model, *GVNR* (Global Vectors for Node Representation), to learn node representations, taking into account both co-occurrence and non co-occurrence. We formulate a factorization problem on the thresholded co-occurrence matrix. More specifically, we formulate this problem so that the reconstruction error is measured on all the positive entries and some randomly sampled zero entries. We begin by listing the set of notations we use in Table 1, next we describe the matrix to factorize and then formulate the factorization problem. Eventually, we discuss the relationship to other models.

3.1 Description of the Matrix to Factorize

We observed node co-occurrences in truncated random walks [11]. Then, for each node j visited within q steps, with $q \leq l$, from a node i, we increase X_{ij} by $\frac{1}{q}$ [10]. Thus, we construct a weighted co-occurrence matrix, so that distant co-occurrences are increasingly downweighted. The matrix obtained with this procedure is approximately proportional to the weighted sum of the l first of the adjacency matrix: $\sum_{i=1}^{l} \frac{1}{i} A^i$. However, this sum is likely to give a denser co-occurrence matrix because unlikely and distant co-occurrences will lead to coefficients close to zero. On the contrary, the truncated random walk is likely to estimate these coefficients as exactly zero. Because we consider coefficients close to zero as noise, we're not interested in calculating them, and can computationally benefit from a sparser matrix. For the same reason, we zero-out coefficients that are less than a threshold $x_{\min}$, assuming they are irrelevant.

3.2 Formulation of the Factorization Problem

We formulate a factorization problem on X, measuring the error only for positive coefficients and a fraction of randomly sampled zero coefficients. Note that we measure the error w.r.t the logarithm, which help compress the range of values in X [10]. Because the matrix is already thresholded, we assume all the remaining positive entries have the same importance, thus we don't weight the least-square objective:

$$\underset{U,V,b^U,b^V}{\operatorname{argmin}} \sum_{i=1}^{n} \sum_{j=1}^{n} s(x_{ij}) \Big(u_i \cdot v_j + b_i^U + b_j^V - \log(c + x_{ij}) \Big)^2. \quad (7)$$

The constant $c \in]0; 1]$ allows for smoothing X while making the logarithm negative when $x_{ij} = 0$. The function s effectively selects the coefficients considered for measuring the reconstruction error:

$$s(x_{ij}) = \begin{cases} 1 & \text{if } x_{ij} > 0, \\ m_i & \text{else, with } m_i \sim \text{Bernoulli}(\alpha_i). \end{cases} \quad (8)$$

It takes the value 1 for all positive coefficients of X, while for zero coefficients, its value is given by a Bernoulli random variable, m_i, with a node-specific parameter α_i. Denoting the proportion of positive coefficient on the i$^{\text{th}}$ row of X by p_i, α_i is calculated in terms of the odd-ratio:

$$\alpha_i = \begin{cases} k \times \frac{p_i}{1-p_i} & \text{if } p_i \leq (k+1)^{-1}, \\ 1 & \text{else} \end{cases} \quad (9)$$

where $k > 0$ is an hyper-parameter that controls the proportion of zero coefficients incorporated into the calculation of the reconstruction error, akin to the number of negative samples in SGNS. The larger k, the more importance is given to pushing away vectors of non co-occurring nodes.

3.3 Relationship to Other Models

This objective function bears a resemblance to the objective of GloVe, still the two are quite different. As stated by the equation 5, GloVe performs a weighted least-square regression on the raw co-occurrence matrix. Although the authors of GloVe claim that rare co-occurrences are noisy and carry less information than the more frequent ones, this objective function still takes them into account (with a proportionally smaller weight, according to the equation 6). Our model has a stronger interpretation of this claim, by zeroing out rare co-occurrences, and weighting equally all the others. In addition, while all zero coefficients are ignored in GloVe due to the definition of f in equation 6, we incorporate a fraction of them, proportional to the quantity of positive coefficients. That constitutes an additional set of constraints we think should lead to better representations. Lastly, thresholding X helps us eliminating noise while drastically sparsifying it, since x_{ij} follows a power law [10, 11].

The complexity of Skip-Gram based algorithms, implemented with the procedure described in [9], is linear in the size of the multiset of pairs of co-occurring nodes, *i.e.* $O(|C|)$. Based on the proof given in [10], the complexity of our model is $O(|C|^{\frac{1}{\alpha}})$, where α is the exponent of the power law that models x_{ij}. For the networks studied in this paper, we observe a mean value for $\frac{1}{\alpha}$ of 0.79.

3.4 Extension to Networks of Documents

Lastly, we show how to extend the general model under the name *GVNR-t*, to deal with networks where nodes are text documents.

Assuming word order is negligible for documents [2], we can model a text as a bag of words and thus represent it by a vector $\delta \in \mathbb{N}^{+m}$, m being the size of the vocabulary. We can further

Table 2: General properties of the studied networks.

	$\|V\|$	$\|E\|$	# labels	weighted	multi-label
Citation 1	2,708	10,556	7	no	no
Citation 2	3,312	9,226	6	no	no
Co-authorship	5,021	29,856	5	yes	no
Protein	3,890	76,584	50	no	yes
Language	4,777	184,812	40	yes	yes

assume that the meaning of a text can be captured by averaging the representations of its words [7]. Therefore, with $W \in \mathbb{R}^{m \times d}$ a word embedding matrix, rather than learning the the context-vector of a node as explained previously, we define it as the average of the representations of the words it contains:

$$\underset{U,V,b^U,b^V}{\operatorname{argmin}} \sum_{i=1}^{n}\sum_{j=1}^{n} s(x_{ij})\Big(u_i \cdot \frac{\delta_j\, W}{|\delta_j|_1} + b_i^U + b_j^V - \log(c + x_{ij})\Big)^2, \quad (10)$$

where δ_j is the bag of words representation of text associated to the node j, and $|\delta_j|_1$ is the number of words in it. Thus, the model jointly learns node and word representations, that in turn allows representing documents.

4 QUANTITATIVE EVALUATION

A common task in network analysis is node classification. Following the experimental designs in recent works [12, 18], we assess the quality of the representations learned with *GVNR* by using them as input of a linear classifier to solve multi-class and multi-label classification tasks.

4.1 Networks

To show the versatility of *GVNR*, we consider five networks of various nature:

- Two citations networks: Citation (1) and Citation (2), extracted respectively from Cora and Citeseer[2]; each node is an article and is labelled with a conference.
- A co-authorship network extracted from DBLP[3]; each node is an author labelled with a domain of expertise[4] and each edge is weighted according to the number of common publications.
- A protein-protein interaction (PPI) network [14] which is a subgraph of the PPI network for Homo Sapiens. Each node is associated to several labels that represent biological states.
- A language network that describes collocated words observed in the first 10^8 bytes of the English Wikipedia[5] (as of March, 2006); each node is associated to multiple labels, which are the potential part-of-speech tags identified with the Stanford POS tagger [17].

The general properties of these five networks are reported in Table 2.

[2]https://linqs.soe.ucsc.edu/data
[3]https://dblp.uni-trier.de/
[4]https://static.aminer.org/lab-datasets/expertfinding/
[5]http://mattmahoney.net/dc/text.html

4.2 Tasks and Evaluation Metrics

For each network, we consider a classification task and evaluate the performance of a linear classifier, namely a logistic regression, using node representations as input. We use the LIBLINEAR implementation [4], without regularization, for a fair comparison between different representations. For each network, we train and evaluate the classifier by cross-validation, with varying split ratios.

More precisely, we consider a multi-class classification problem for the citation and co-authorship networks and measure the overall accuracy of the one-vs-all logistic regression. We consider a multi-label classification problem for the protein-protein and the language networks and measure the F_1 score of the one-vs-all logistic regression, following the procedure described in [12].

4.3 Compared Representations

We run $\gamma = 80$ random walks per node of length $t = 40$ and apply a sliding window of size $l = 5$ to generate a multiset of co-occurring nodes and then learn representation with $d = 80$ with four algorithms:

- DeepWalk: we report the results with the hierarchical softmax approximation.
- GloVe: we report the results with $x_{\max} = 10$.
- NetMF: we report the results with the small-scale implementation provided by the authors, with $k = 10$ negative samples.
- *GVNR*: we report the results with $c = 1$, $x_{\min} = 1$ and $k = 1$ and discuss their impact in the next sub-section.

4.4 Result Analysis

Tables 3 to 7 detail the accuracy measures. The classifier performs well with the representations learned by *GVNR*, achieving similar or better results w.r.t the representations learned with DeepWalk and NetMF. Still, there is one exception concerning the language network, where the classifier performs best with the representation learned with GloVe. We sum up our findings in two points:

(1) *GVNR* always achieves a good performance, while there is more variance in the performance of the baselines. For instance, NetMF is largely outperformed by the others on the language network, GloVe is also significantly outperformed on the co-authorship network, while DeepWalk struggles to learn good representations from the citation (2) network.
(2) Thresholding always improves the performance of *GVNR*. As an example, its leads to an average gain of 9.4 accuracy on the co-authorship network.

Interestingly, GloVe produces the best results on the language network. The average number of neighbors is 38.7 for this network which makes it the denser network.

4.5 Impact of the Hyper-Parameters

Figure 1 shows the sensitivity, in terms of accuracy, of *GVNR* to its hyper-parameters, namely, the threshold $x_{\min}$, the shifting constant c, the window length l, and the sampling proportion k. We only report the results for the citation (1) network since their are similar across the considered networks. We see that c has little impact on the accuracy. In practice, we found *GVNR* performs best across the

Table 3: Accuracy on the citation (1) network.

	% of training data				
	10%	20%	30%	40%	50%
GloVe	57.7	62.4	69.5	72.8	73.8
GVNR ($x_{\min} = 0$)	58.5	62.5	70.7	73.4	75.0
NetMF	65.7	**72.9**	**76.4**	**78.6**	79.4
DeepWalk	67.8	71.6	74.5	75.8	79.2
GVNR ($x_{\min} = 1$)	**69.5**	72.6	75.9	78.1	**80.2**

Table 4: Accuracy on the citation (2) network.

	% of training data				
	10%	20%	30%	40%	50%
GloVe	42.8	53.5	55.3	56.2	56.8
GVNR ($x_{\min} = 0$)	38.7	46.8	49.1	50.4	50.9
NetMF	**51.2**	54.8	55.1	55.0	54.8
DeepWalk	41.3	52.5	54.5	55.5	56.0
GVNR ($x_{\min} = 1$)	45.6	**55.6**	**57.3**	**58.7**	**59.0**

Table 5: Accuracy on the co-authorship network.

	% of training data				
	10%	20%	30%	40%	50%
GloVe	41.0	42.1	43.7	46.4	51.2
GVNR ($x_{\min} = 0$)	60.3	64.6	67.4	67.1	68.2
NetMF	60.7	66.2	70.1	72.1	72.8
DeepWalk	**75.4**	**77.2**	**77.3**	**75.9**	**79.3**
GVNR ($x_{\min} = 1$)	74.7	75.3	76.3	73.8	74.6

Table 6: F_1 score on the language network.

	% of training data				
	10%	20%	30%	40%	50%
GloVe	**34.0**	**44.1**	**46.7**	**47.7**	**48.6**
GVNR ($x_{\min} = 0$)	31.7	40.7	43.2	44.7	45.1
NetMF	27.5	33.5	36.2	37.7	38.7
DeepWalk	33.6	43.6	46.2	47.6	48.2
GVNR ($x_{\min} = 1$)	32.2	41.7	44.0	45.2	46.1

datasets with $c = 1$. It seems that the model performs best with a threshold value between 1 and 5. Setting $x_{\min} > 0$ clearly improves the performance. In general, we observed that $k = 1$ constantly brings an improvements over $k = 0$, but higher values only brings a boost for high training ratios. Finally, the accuracy increases along with the window size l, with few improvements above $l = 5$.

4.6 Additional Results with Text

We now report additional results when taking into account the text information, associated to the nodes of the citation networks, *i.e.* titles and abstracts. We consider three ways of learning text-aware representations: TADW [18], the representations learned with DeepWalk concatenated with the LSA representation of the texts, and *GVNR-t*. We report the results for TADW with 20 iterations and 4 iterations for *GVNR-t*.

Table 7: F_1 score on the protein-protein network.

	% of training data				
	10%	20%	30%	40%	50%
GloVe	11.8	13.8	15.6	17.3	19.1
GVNR ($x_{\min} = 0$)	10.7	13.3	15.0	16.7	18.1
NetMF	10.2	11.7	13.5	14.7	16.1
DeepWalk	**12.2**	**13.9**	**16.2**	**17.8**	**19.6**
GVNR ($x_{\min} = 1$)	11.7	13.3	15.8	17.6	19.2

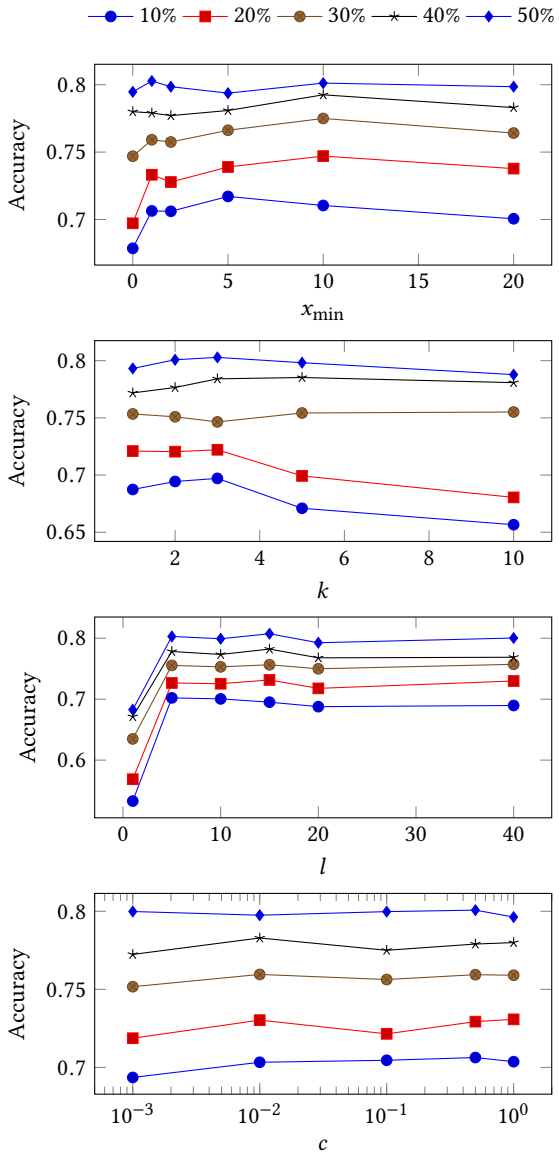


Figure 1: Sensitivity of *GVNR* to the hyper-parameters $x_{\min}$, k, l and c on the citation (1) network.

Tables 8 and 9 report the accuracies. *GVNR-t* shows interesting improvements over the simple concatenation of text and graph features and the reference baseline in the field, TADW. This motivates the study of the word representations learned by *GVNR-t* and the interplay between the node and document representations.

Table 8: Accuracy on the citation (1) network, considering the text features.

	% of training data				
	10%	20%	30%	40%	50%
LSA	54.7	61.0	62.4	63.0	62.8
DeepWalk+LSA	73.8	77.9	78.4	78.1	78.1
TADW	77.1	78.8	78.2	78.8	78.6
GVNR-t	**79.3**	**80.7**	**80.8**	**81.4**	**81.1**

Table 9: Accuracy on the citation (2) network, considering the text features.

	% of training data				
	10%	20%	30%	40%	50%
LSA	52.0	54.7	54.7	58.4	65.7
DeepWalk+LSA	58.3	60.7	61.1	60.0	61.2
TADW	60.6	60.1	60.1	66.2	69.3
GVNR-t	**63.3**	**62.5**	**64.9**	**68.6**	**70.4**

5 CASE STUDY

To showcase the usefulness of learning jointly word, node and document representation, we apply *GVNR-t* to full DBLP network that consists in 1,397,240 documents and 3,021,489 citation relationships. After computing X, we threshold it with $x_{\min} = 20$, which divides the density of $X_{(ij)}$ by 50. For the learning phase, we keep the same settings as before, that is $d = 80$, $k = 1$, $c = 1$, $l = 5$. Note that we apply a standard pre-processing, that consists in merging recurrent phrases, as suggested in [9]. Computing X and estimating U and W, the node and word representations takes about 8 hours on a single machine with 32 cores and 192 GB of RAM.

Our aim is to show that we can measure the similarity, on the one hand, between word and node representations, and, on the other hand, between word and document representations, to extract sets of complementary keywords.

Tables 10 and 11 show 2 randomly selected papers for which we computed the 5 closest word embeddings w_k to, respectively, (i) the node representation u and (ii) its content representation v. First, we note that the keywords are all relevant, even though none of them actually appear in the documents. Then, we see that words close to the node representation are more general, giving a broad view of the studied topic, whereas words close to the document representation are more specialized and provide a more fine-grained perception. We suspect that the network topology, mostly influencing U, helps locating a node in its broader context, while the content of documents, mostly influencing W, helps in selecting some very specific keywords. In future work, we would like to investigate further the quality of the learned representations for a wider range of recommendation tasks.

Table 10: Keyword recommendation by selecting the closest word embeddings w_k to both embeddings u (node) and v (content) of an input document (1).

Document	**A brief survey of computational approaches in social computing** Web 2.0 technologies have brought new ways of connecting people in social networks for collaboration in various on-line communities. Social Computing is a novel and emerging computing paradigm...
Closest words to u (node)	*cold start problem, storylines, document titles, movielens data, computational humor*
Closest words to v (content)	*social, social network, enron email corpus, social networks, extremely large datasets, sites blogs*

Table 11: Keyword recommendation by selecting the closest word embeddings w_k to both embeddings u (node) and v (content) of an input document (2).

Document	**Discovering company revenue relations from news** A network approach Large volumes of online business news provide an opportunity to explore various aspects of companies. A news story pertaining to a company often cites other companies. Using such company citations we...
Closest words to u (node)	*datenbanksystemen, denizens, want hear, technological infrastructures, asynchronous discussions*
Closest words to v (content)	*company, consumer brand, today highly competitive, data cleaning, consumer heterogeneity*

6 CONCLUSION

In this paper, we presented *GVNR*, a matrix factorization based method for network embedding, that better handles non co-occurrence than GloVe. We further extended this model to incorporate textual content associated with the nodes to learn meaningful representations of words and documents. We showed that *GVNR* performs state-of-the-art results on a wide range of networks and can provide recommendations based on the distance between words, documents and graph embeddings. In future works we would like to explore further the relations between the word embeddings learned with *GVNR-t* and traditional word embeddings learned with GloVe or Skip-Gram. More particularly, we would like to study whether *GVNR-t* could help learning better word embeddings from small structured corpora.

REFERENCES

[1] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.

[2] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE* 41, 6 (1990), 391–407.

[3] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 135–144.

[4] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of machine learning research* 9, Aug (2008), 1871–1874.

[5] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[6] Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A. Zighed. 2013. Information Diffusion in Online Social Networks: A Survey. *SIGMOD Record* 42, 2 (2013), 17–28.

[7] Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*. 1188–1196.

[8] Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics* 3 (2015), 211–225.

[9] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[10] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[11] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[12] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 459–467.

[13] Magnus Sahlgren. 2008. The distributional hypothesis. *Italian journal of linguistics* (2008), 23–53.

[14] Chris Stark, Bobby-Joe Breitkreutz, Andrew Chatr-Aryamontri, Lorrie Boucher, Rose Oughtred, Michael S Livstone, Julie Nixon, Kimberly Van Auken, Xiaodong Wang, Xiaoqi Shi, et al. 2010. The BioGRID interaction database: 2011 update. *Nucleic acids research* 39, suppl_1 (2010), D698–D704.

[15] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. 1067–1077.

[16] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 990–998.

[17] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 173–180.

[18] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y Chang. 2015. Network representation learning with rich text information.. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 2111–2117.

The Music Streaming Sessions Dataset

Brian Brost
Spotify Research, London
brianbrost@spotify.com

Rishabh Mehrotra
Spotify Research, London
rishabhm@spotify.com

Tristan Jehan
Spotify Research, NY
tjehan@spotify.com

ABSTRACT

At the core of many important machine learning problems faced by online streaming services is a need to model how users interact with the content they are served. Unfortunately, there are no public datasets currently available that enable researchers to explore this topic. In order to spur that research, we release the Music Streaming Sessions Dataset (MSSD), which consists of 160 million listening sessions and associated user actions. Furthermore, we provide audio features and metadata for the approximately 3.7 million unique tracks referred to in the logs. This is the largest collection of such track metadata currently available to the public. This dataset enables research on important problems including how to model user listening and interaction behaviour in streaming, as well as Music Information Retrieval (MIR), and session-based sequential recommendations. Additionally, a subset of sessions were collected using a uniformly random recommendation setting, enabling their use for counterfactual evaluation of such sequential recommendations. Finally, we provide an analysis of user behavior and suggest further research problems which can be addressed using the dataset.

KEYWORDS

music streaming; user sessions; dataset; user interactions

ACM Reference Format:
Brian Brost, Rishabh Mehrotra, and Tristan Jehan. 2019. The Music Streaming Sessions Dataset. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313641

1 INTRODUCTION

A long-standing and central challenge for online services is to understand and model how users behave [18]. For web search and online advertising, this has led to a large body of work on click modeling, most of which would not have been possible without access to publicly available click logs [8]. Modeling user behaviour is similarly important to streaming services [17, 19], but to the best of our knowledge there are no user interaction datasets currently available to the public. This is particularly limiting when it comes to designing recommender systems, where the use of implicit feedback is often critical [14].

Motivated by the paucity of user interaction logs in streaming, we release the Music Streaming Sessions Dataset (MSSD), which consists of over 160 million listening sessions with associated user interaction information. In addition, we provide audio features and metadata for the approximately 3.7 million unique tracks referred to in the logs, making this the largest collection of such track metadata currently available to the public. A useful characteristic of the dataset is that a subset of the sessions contained in the log were obtained using a uniformly random shuffle, enabling research on counterfactual methods. Lastly, we provide snapshots of the playlists from which this subset of sessions was streamed.

By providing the sequences of track plays, the smaller pools of tracks which were available for recommendation during the session, and the probabilities each of those tracks had of being played next, we allow this dataset to be used as a test-bed for sequential recommendation research, and in particular with counterfactual approaches to the problem. The rise of counterfactual methods [4] dramatically improved the speed of experimentation for online services, however there is still very little data available to academic researchers in that area, and no data at all for the specific problem of sequential recommendation.

One of the main ambitions of this dataset release is to enable public research on two central challenges facing music streaming services, namely understanding when a user will 1) skip a track, and 2) move from one listening context to another. Related to this dataset we therefore also organized a skip prediction machine learning challenge for the 2019 WSDM Cup. More generally, we expect that this dataset will spur further research in the area of Music Information Retrieval (MIR), where it will function as a partial expansion of the Million Song Dataset (MSD) [3]. For the first time we provide a dataset that links track audio features with user listening behaviour. Finally, sequentially recommending items for users is of particular importance to music streaming services but also to other types of services, such as news or e-commerce, for example in next-basket recommendation. We believe the underlying solutions for these cases would be comparable to those for music.

Our contributions are therefore threefold: (1) we provide the only dataset of streaming logs and interactions currently available to the public in Section 3; (2) we provide an analysis of the logs in Section 4; and (3) we identify important research questions that can be addressed using this dataset in Section 5.

2 RELATED WORK

Click log releases have played a key role in allowing the development of sophisticated click models for web search and advertising applications, with companies including Microsoft, Yahoo, Yandex, and Criteo all providing such logs to the academic research community. An overview of the click modeling literature, and available datasets is provided in [8].

In the context of recommender systems there are fewer such interaction logs available. Instead, recommender systems datasets have tended to contain explicit ratings [2, 9, 11]. Some streaming logs containing user interactions have been released, for example the XING dataset which was part of the 2016 Recsys Challenge [1].

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313641

However, this dataset was in the field of job recommendations and is no longer publicly available.

There are several music listening histories datasets, most of which were crawled from from social media such as last.fm or twitter, however none of these datasets contain user interactions beyond what songs were listened to [5, 12, 15–17]. Similarly, the kkbox dataset from the 2018 WSDM Cup challenge provides listening logs [7], but these logs are not timestamped, nor do they provide user interactions beyond what tracks were listened to. Thus, although there are some music listening logs, crucially none of these contain information about how users interacted with the tracks they listened to, showing instead only what tracks users were exposed to.

An important feature of our dataset is that a subset of the sessions contained in the logs were collected from users experiencing a uniformly random shuffle function. This allows our dataset to be used in connection with so-called counterfactual methods. These have the potential to allow offline evaluations of new algorithms while only requiring some randomization of the deployed algorithm's output [4]. If the cost of randomizing the deployed algorithm's output is not too great, this can provide a practical way of obtaining results that are potentially as reliable as those obtained from A/B testing, but without requiring the variants being tested to be production ready [10]. This can allow tests of new algorithms to be carried out more quickly and efficiently than what is currently possible. To the best of our knowledge there are no other public datasets except the counterfactual test-bed from [13]. Our dataset distinguishes itself since it is focused on the novel problem of applying counterfactual methods to sequential recommendations. Counterfactual evaluation of sequenced recommendations provides an interesting challenge since a naive approach would suffer from a growth in the propensities exponential in the length of the sequence.

3 THE MUSIC STREAMING SESSIONS DATASET

The MSSD consists of 160 million streaming sessions with associated user interactions, audio features and metadata describing the tracks streamed during the sessions, and snapshots of the playlists listened to during the sessions. The dataset is hosted at http://research.spotify.com/datasets/music-streaming-sessions together with a set of tools for working with the dataset.

The streaming sessions are stored in a log, where each row of the log contains a session id, a timestamp, contextual information about the stream, the track and context id's, and the timing and type of user interactions within the stream. A schema for the log is provided in Table 1. Each session is defined to be a period of listening with no more than 60 seconds of inactivity between consecutive tracks. Additionally for this dataset we set a cut off of at most 20 tracks per session as part of our privacy strategy. Sessions included in the dataset are sampled uniformly at random from eligible listening sessions on the contexts included in the dataset over an 8 week period. We exclude sessions that include tracks which did not meet a minimum popularity threshold. The sessions in this dataset are sampled from radio, personalized recommendation mixes, the user's own collections, and 100 of the most popular playlists on a major music streaming service. The logs therefore contain a mix of listening sessions based on user's personally curated collections; expertly curated playlists; contextual, but non-personalized recommendations; and finally, personalized recommendations.

For each track contained in the sessions, we provide audio features and metadata describing the track. This part of our dataset can be regarded as a partial update and expansion of the Million Song Dataset (MSD) [3]. We provide approximately 3.7 million tracks, and many new features not included in the Million Song Dataset. These include features like acousticness, a measure of confidence that the track is acoustic; downbeats, estimated timestamps for the downbeats in a track; and valence, a measure of how positive a track sounds. The schema for the track features is provided in Table 2, features which did not exist in the Million Song Dataset are bolded. More detailed descriptions of the features are available in [3] and on our dataset website.

3.1 Uniform Random shuffle subset

As noted earlier, counterfactual methods have the potential to allow offline evaluations of new algorithms if the deployed algorithm's output is randomized and the propensities are known [4]. The standard shuffle function used during streaming on our service is not uniformly random, however we have collected and labelled a subset of the sessions contained in this log using a uniformly random shuffle.

With this dataset we focus on the novel problem of applying counterfactual methods to sequential recommendations. Shuffled listening sessions provide a nice means of collecting randomized observations, without seriously harming the user's experience since the user already expects a somewhat random experience. However good and bad sequences of tracks can still be distinguished by a variety of session level and track play level metrics such as session length and skip rate.

For each session using the uniformly random shuffle function we include snapshots of the tracks contained in the playlist and their positions in the playlists. This allows us to use the logs and playlist snapshots for counterfactual evaluations of sequential recommendations as outlined in Section 3.1.

4 DATASET ANALYSIS

Here we provide summary statistics and figures for the dataset, highlight some of its properties, and explore implications for future user modeling efforts. The logs detail how users behave in terms of two particularly important forms of implicit feedback: user skip behavior, and their context switch behavior, i.e. when users change from one listening context or playlist to another.

4.1 Summary Statistics

The dataset consists of 160 million sessions with lengths varying between 10 to 20 interactions. Recall that a session is defined as a period of listening interrupted by no more than 60 seconds between consecutive playbacks. Sessions shorter than this limit are excluded to try to increase the amount of information available early in sessions to predict a user's interactions later in the session. Recall that longer sessions are excluded as part of our privacy strategy. Figure 1 shows that the proportion of sessions of a given length decreases steadily with increasing length, although sessions of

Table 1: Schema for the interaction log

Column name	Column description	Example value
session id	unique session identifier	57 _55129e3f-29bf-4ef6-aa72-d140333eac9c
session position	position of track within session	18
session length	length of session	20
track id	unique track identifier	t _aae12819-de17-4dd3-97b0-cad4dd7b9a56
skip 1	whether the track was only played very briefly	false
skip 2	whether the track was only played briefly	false
skip 3	whether most of the track was played	true
not skipped	whether the track was played in its entirety	false
context switch	whether the user changed context between the previous row and the current row	true
no pause	whether there was no pause between playback of the previous track and current track	false
short pause	whether there was a short pause between playback of the previous track and current track	true
long pause	whether there was a long pause between playback of the previous track and current track	true
num seekfwd	the number of times the user scrubbed forward during playback	0
num seekbk	the number of times the user scrubbed backward during playback	3
shuffle	whether the track was played with shuffle mode activated	false
hour of day	hour of day (integers between 0 and 23)	18
date	date in YYYY-MM-DD format	2018-09-10
premium	whether the user was on premium or not	true
context type	what type of context the playback occurred within	catalog
reason start	cause of this track play starting	forward button
reason end	cause of this track play ending	track done
uniform random	whether shuffle would be uniformly random for this session	false

Table 2: Features for the track metadata

track id	duration	release year
popularity	**acousticness**	**beat strength**
bounciness	danceability	**dyn range mean**
energy	**flatness**	**instrumentalness**
key	**liveness**	loudness
mechanism	mode	**organism**
speechiness	tempo	time signature
valence	**acoustic vector**	

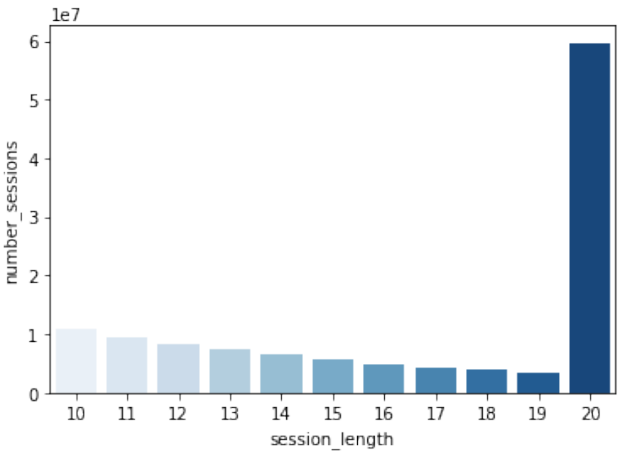


Figure 1: Distribution of session lengths

length 20 are much more common since sessions that were longer are capped at this length.

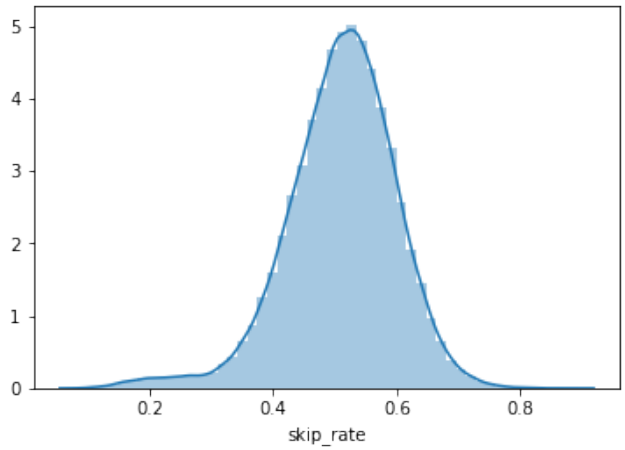


Figure 2: Distribution of skip rate across tracks

4.2 User Interactions

We begin by exploring user skip behavior. In this dataset we provide a variety of different skip definitions, for this analysis we use the skip _1 threshold. The distribution of skip rates across tracks is given in Figure 3.

In general, skip rates are higher for longer sessions, as shown in Figure 3. However, whereas skip rates are higher for longer sessions, skip rates are not substantially higher earlier in sessions, remaining relatively constant at a non-skip rate between 34% and 35% for all session positions.

Whereas it might be expected that a user explicitly selecting a song would lead to a low probability of that song being skipped, we see from Figure 4 that the reason for a track starting that is most

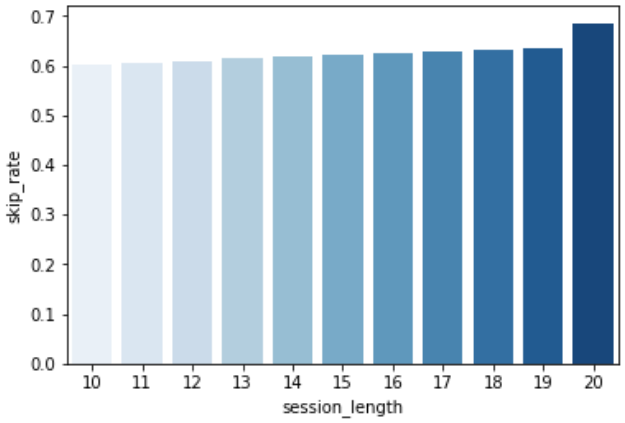


Figure 3: Session length against skip rate

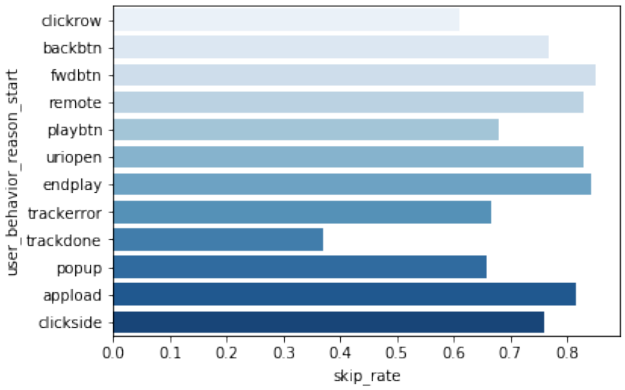


Figure 4: Skip rate against reason for start of playback

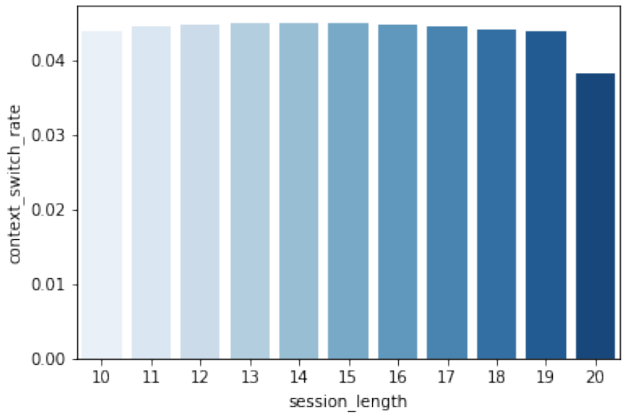


Figure 5: Session length against context switch rate

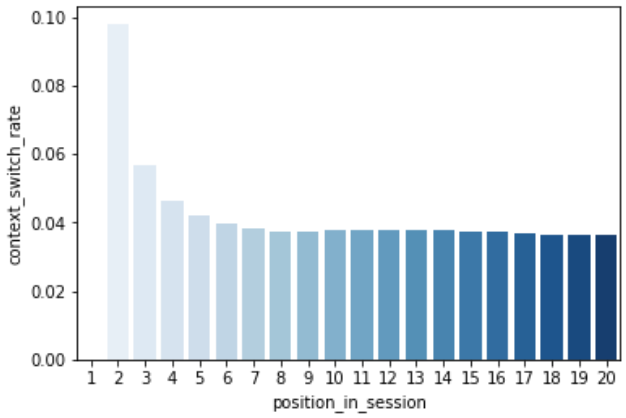


Figure 6: Session position against context switch rate

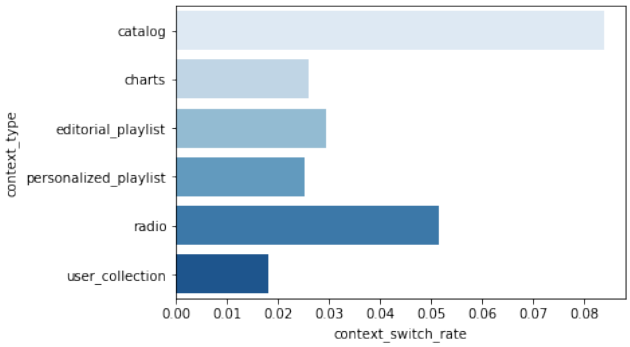


Figure 7: Context switch rate against source context

closely associated with a track not being skipped is that the previous track was listened to completion. This may be partly explained by the fact that although a user who explicitly selected a song wanted to listen to that track, a user who is in a so-called lean-back listening mode, is even less likely to skip. A deeper analysis of such listening modes is an interesting challenge for MIR researchers.

Whereas skip rates are highly correlated with session length, this is less obviously the case for the context switch rate, as shown in Figure 5, with the exception of sessions of length 20, which due to the cutoff include potentially longer sessions. Here we see a substantially lower context switch rate, possibly reflecting extreme lean-back sessions, such as sleep playlists. Despite this, we see in Figure 6 that context switches are much more likely at the beginning of a session. Finally, we see in Figure 7 that the likelihood of a user switching context depends heavily on the source context. Understanding the causes and directions of users context switches is an important challenge for music streaming services.

4.3 Track Features

Figure 8 shows the distributions of feature values across tracks for selected audio features. The distributions for the remaining audio features are included on the dataset website. Features display dramatically different distributions, with a feature like instrumentalness bimodal and highly skewed, with most tracks having a very low value indicating an extremely low probability estimate

of the track being an instrumental track, and relatively few tracks displaying a high estimated probability of being instrumental.

Such a large number of audio features and the variations in their distribution could inspire future research on understanding how user consumption is linked with content level features. While most user engagement studies in the past have looked at patterns across recommended items, jointly performing such analysis on content features would make for an interesting insightful discussion.

5 FURTHER RESEARCH PROBLEMS

While we have highlighted issues around user interactions with a sequential recommender, specifically predicting user interactions (skips) and context switches, the released dataset allows us to go beyond these problems, and can serve as a benchmark dataset for researchers interested in various other research areas. Here we list a few key potential research areas.

1. Session based Sequential Recommendations
While most machine learning techniques focus on either item recommendation or even set recommendations aimed at increasing instantaneous user satisfaction, we believe that this dataset will enable researchers to propose novel techniques aimed at optimizing user sessions, and more long term satisfaction.

2. User Intervention in Automated Systems
For each session, the dataset provides user interaction with each track, which opens up interesting problems around human interventions with automated systems. Indeed, the user might begin by relying on the recommender system to suggest content, but at some point decide to intervene, either slightly (e.g. only skip a track) or to a larger extent (e.g. stop using the recommender system and move to their own playlist). As we develop advanced automated systems, we need to understand user reactions to them and comprehend, predict and leverage user interventions, in an attempt to not only reduce the need for further interventions but also to optimize the system more efficiently.

3. Offline Evaluation of Recommender Systems
Metrics and reward models of machine learning systems relies heavily on user feedback. The detailed per item meta-data allows researchers to develop advanced evaluation metrics and reward functions for systems to learn from.

4. User Journeys
The session level data enables research on carefully constructing user journeys not only in the general space of recommendation items, but also in the space of user moods. Indeed, research on music therapy has highlighted the impact of positive music [6], and this dataset enables researchers to develop carefully crafted user journeys to affect user moods.

5. Proactive Recommendations
Anticipating user actions and interventions allows machine learning systems to be proactive in recommendations. So far, research on proactive recommendations has been conducted in the industry by researchers having access to large scale user interaction logs. We hope the dataset enables academic researchers to conduct research on proactive systems.

6. Counterfactual Evaluation
Most major web companies rely on large scale A/B tests for model development and iteration. Counterfactual estimation of metrics has recently enabled researchers to make offline predictions of online metrics. We advocate the use of counterfactual estimation techniques for unbiased offline evaluation of systems using our dataset. Compared to A/B tests, offline evaluation allows multiple models to be evaluated on the same log, without the need to be run online. Effectively, counterfactual estimation techniques make it possible to run many A/B tests simultaneously, leading to substantial increase in experimentation agility. Developing good ounterfactual estimators for sequential recommendations is an open research problem for which this dataset can function as a test bed.

Owing to the heterogeneous data released as part of this dataset (user sessions, audio featres, user contexts, etc), we envision myriad future use-cases of this dataset, beyond the six broader areas highlighted above.

6 CONCLUSION

The problems of understanding, modeling and predicting how users interact with content on streaming services has until now been understudied, mainly because of a lack of access to data. With the paper, we provide the only dataset of streaming logs and user interactions currently available to the research community.

With this dataset we address two key problems facing streaming services, namely how to predict user skips and context switches. We provide an analysis of the dataset in terms of these problems. In particular, we illustrated that skip behavior in a session is correlated with prior skip behavior within that session. We believe that a thorough investigation of these forms of user behavior is an important future research challenge.

Finally, we identify other important research questions that can be addressed using the dataset. We believe that the task of providing session-based sequential recommendations is of particular importance, due to the need for fast learning from limited user-based information when onboarding new users.

While the dataset enables new types research directions, compromises were made for privacy and commercial reasons. An interesting future extension of this dataset would be to provide more precision on the skip information than the currently bucketed times. That level of information could allow to predict moments in the track where skips are most likely to occur, which could be of great value for generative and interactive music models.

REFERENCES

[1] Fabian Abel, András Benczúr, Daniel Kohlsdorf, Martha Larson, and Róbert Pálovics. Recsys challenge 2016: Job recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 425–426. ACM, 2016.

[2] Robert M Bell and Yehuda Koren. Lessons from the netflix prize challenge. *Acm Sigkdd Explorations Newsletter*, 9(2):75–79, 2007.

[3] Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Ismir*, volume 2, page 10, 2011.

[4] Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. Counterfactual reasoning and learning systems: The example of computational advertising. *The Journal of Machine Learning Research*, 14(1):3207–3260, 2013.

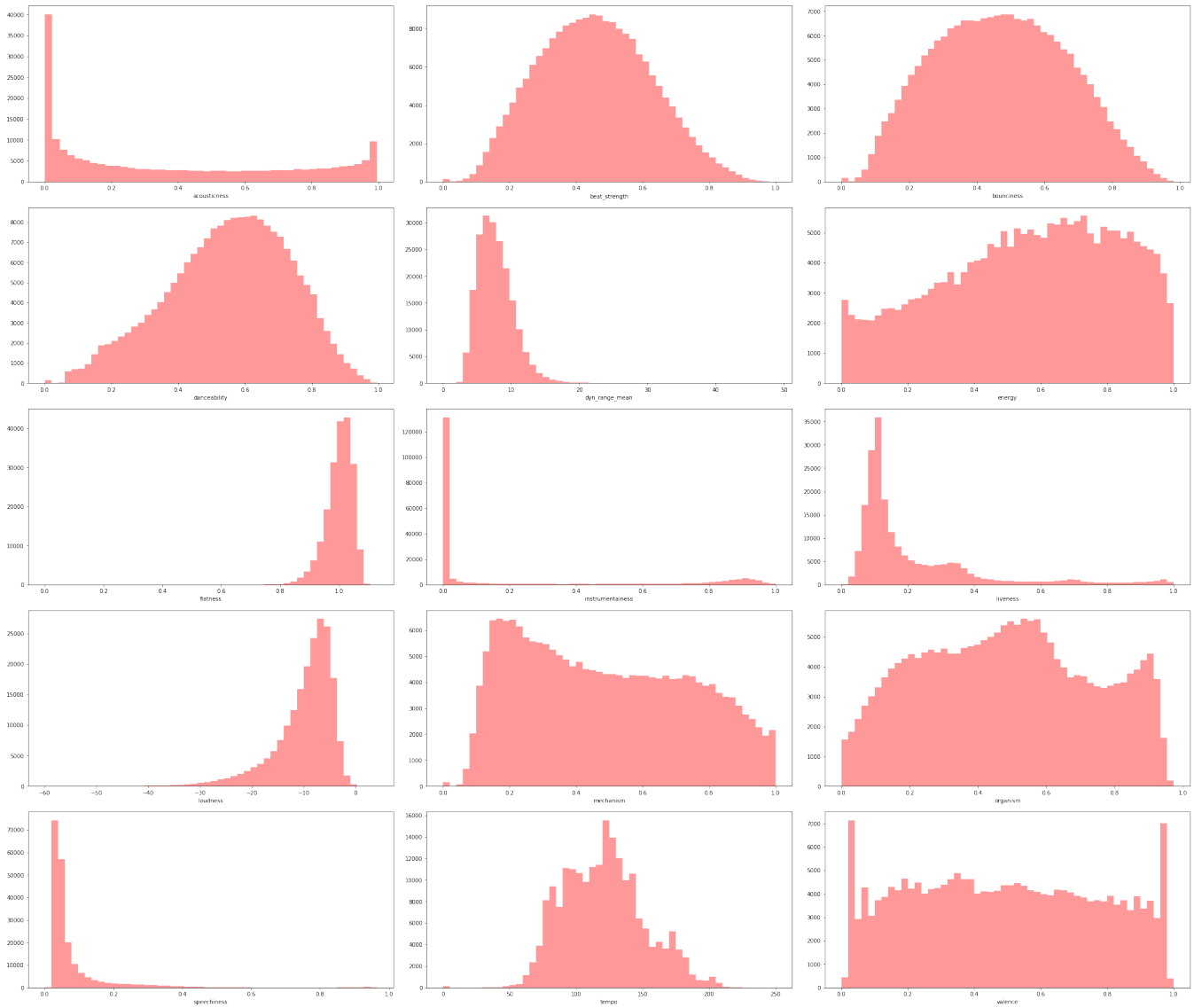


Figure 8: Distributions of feature values across tracks for selected audio features

[5] O. Celma. *Music Recommendation and Discovery in the Long Tail.* Springer, 2010.

[6] Early Intervention Center, Cardiac Rehabilitation Center, and Physical Fitness Center. Music therapy. 2005.

[7] Yian Chen, Xing Xie, Shou-De Lin, and Arden Chiu. Wsdm cup 2018: Music recommendation and churn prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 8–9. ACM, 2018.

[8] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. Click models for web search. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 7(3):1–115, 2015.

[9] Gideon Dror, Noam Koenigstein, Yehuda Koren, and Markus Weimer. The yahoo! music dataset and kdd-cupâĂŹ11. In *Proceedings of KDD Cup 2011*, pages 3–18, 2012.

[10] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. Offline a/b testing for recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 198–206. ACM, 2018.

[11] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19, 2016.

[12] David Hauger, Markus Schedl, Andrej Košir, and Marko Tkalcic. The million musical tweets dataset: what can we learn from microblogs. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIRâĂŹ2013)*, pages 189–194, 2013.

[13] Damien Lefortier, Adith Swaminathan, Xiaotao Gu, Thorsten Joachims, and Maarten de Rijke. Large-scale validation of counterfactual learning methods: A test-bed. *arXiv preprint arXiv:1612.00367*, 2016.

[14] Douglas W Oard, Jinmook Kim, et al. Implicit feedback for recommender systems. In *Proceedings of the AAAI workshop on recommender systems*, volume 83. WoUongong, 1998.

[15] Markus Schedl. Leveraging microblogs for spatiotemporal music information retrieval. In *European Conference on Information Retrieval*, pages 796–799. Springer, 2013.

[16] Markus Schedl. The lfm-1b dataset for music retrieval and recommendation. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, pages 103–110. ACM, 2016.

[17] Gabriel Viglensoni and Ichiro Fujinaga. The music listening histories dataset. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIRâĂŹ2017)*, 2017.

[18] Geoffrey I Webb, Michael J Pazzani, and Daniel Billsus. Machine learning for user modeling. *User modeling and user-adapted interaction*, 11(1-2):19–29, 2001.

[19] Hongliang Yu, Dongdong Zheng, Ben Y Zhao, and Weimin Zheng. Understanding user behavior in large-scale video-on-demand systems. In *ACM SIGOPS Operating Systems Review*, volume 40, pages 333–344. ACM, 2006.

Rethinking the Detection of Child Sexual Abuse Imagery on the Internet

Elie Bursztein
Google

Travis Bright
Thorn

Einat Clarke*
Google

Michelle DeLaune
National Center for Missing and Exploited Children (NCMEC)

David M. Eliff
National Center for Missing and Exploited Children (NCMEC)

Nick Hsu
National Center for Missing and Exploited Children (NCMEC)

Lindsey Olson
National Center for Missing and Exploited Children (NCMEC)

John Shehan
National Center for Missing and Exploited Children (NCMEC)

Madhukar Thakur
Google

Kurt Thomas
Google

ABSTRACT

Over the last decade, the illegal distribution of child sexual abuse imagery (CSAI) has transformed alongside the rise of online sharing platforms. In this paper, we present the first longitudinal measurement study of CSAI distribution online and the threat it poses to society's ability to combat child sexual abuse. Our results illustrate that CSAI has grown exponentially—to nearly 1 million detected events per month—exceeding the capabilities of independent clearinghouses and law enforcement to take action. In order to scale CSAI protections moving forward, we discuss techniques for automating detection and response by using recent advancements in machine learning.

KEYWORDS

child sexual abuse; child exploitation; online abuse

ACM Reference Format:
Elie Bursztein, Travis Bright, Einat Clarke, Michelle DeLaune, David M. Elifff, Nick Hsu, Lindsey Olson, John Shehan, Madhukar Thakur, Kurt Thomas. 2019. Rethinking the Detection of Child Sexual Abuse Imagery on the Internet. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313482

1 INTRODUCTION

Child sexual abuse is a horrific crime affecting an estimated 9–19.7% of girls and 3–7.9% of boys [2, 31, 38]. These children endure indecent exposure, forced intercourse, and sex trafficking [33]. Even after escaping abusers, victims face immediate and lasting consequences such as a heightened risk of depression, substance abuse, and suicide [5, 23].

Compounding the enormity of this situation, in the last decade, child sexual abuse imagery (CSAI) has evolved to leverage online sharing platforms as a tool for distributing abusive videos and images. The children depicted are re-victimized every time the content is accessed. According to the Internet Watch Foundation, which reviews reports of CSAI in the United Kingdom, 53% of CSAI images depict the abuse of children under the age of ten and 28% of CSAI images involve rape and sexual torture [20].

Platform operators including Google, Microsoft, Facebook, and Twitter have responded by proactively blocking attempts to access, share, or distribute CSAI [1, 9, 28, 32]. This involves scanning user-generated content using fingerprints of known abusive images [29] and reporting all incidents, as required by law, to independent clearinghouses that then manually process and action reports. Likewise, law enforcement actively disrupts criminal communities and web forums that foster child sexual abuse [16]. For example, law enforcement agencies dismantled the W0nderland Club, an online network of 300 participants who shared over 750,000 photos of child sexual abuse taken of 1,236 children [4].

Despite intense scrutiny into CSAI on the part of researchers and law enforcement, the broad scale of the problem and the effectiveness of current solutions for protecting online sharing platforms remains unknown. We address this gap by conducting the first longitudinal measurement study of CSAI distribution across the Internet. In collaboration with the National Center for Missing and Exploited Children (NCMEC)—the United States clearinghouse for all CSAI content detected by the public and online sharing platforms in the region—we examine metadata associated with 23.4M incidents of CSAI from 1998–2017. Our study captures the exponential growth of detected CSAI content, the expanding international scope of abuse, and the evolution of the technologies and mediums used to create and distribute CSAI content online.

Overall, our analysis reveals that online sharing platforms have accelerated the pace of CSAI content creation and distribution to a breaking point where NCMEC's manual review capabilities and law enforcement investigations no longer scale. To cope with the current volume of nearly one million CSAI reports per month, we argue there is a pressing need for these institutions and processes to

*Author Clarke has since joined Facebook.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313482

leverage technological advancements to automate CSAI detection and response. We outline promising directions, such as extensive use of deep-learning, that address pain points highlighted in our measurement findings.

In summary, the key insights of this study are:

CSAI reports are growing exponentially. Of 23.4M reports of CSAI, 9.6M (40%) occurred in 2017 alone. The ability of clearing houses to manually investigate each incident cannot scale to this volume of reports absent automated prioritization, labeling, and clustering.

Protecting against CSAI requires coordinated, global action. Ten years ago, 70% of CSAI reports reflected abuse in the Americas. Today, 68% of reports relate to abuse in Asia, 19% the Americas, 6% Europe, and 7% Africa.

New CSAI content is constantly emerging. 84% of detected CSAI images and 91% of videos are reported only once. The specter of potential false negatives due to unknown abusive imagery requires moving from the existing, decades old blacklist-based approaches to algorithms that recognize the nature of CSAI content.

Criminals readily adopt new technologies. Over the years, distribution vectors have changed from email and FTP, to peer-to-peer, and now Tor and resurgent chat and messaging apps. Protections must keep pace with advancements in both distribution channels and digital mediums

2 CHILD SAFETY LAWS & REPORTING

Before diving into our study, we briefly outline the laws in the United States surrounding child sexual abuse imagery (CSAI) and the process that the public and companies use to report abuse to the National Center for Missing and Exploited Children (NCMEC) [30]. These reports form the basis of our study.

2.1 Reporting requirements

CSAI refers to any visual depiction of sexually explicit content involving a minor under the age of 18. United States law prohibits the distribution, production, receipt, and possession of any such material (see generally 18 U.S. Code §2251, 2252). Furthermore, any United States-based electronic service provider (ESP) must report any apparent CSAI to the CyberTipline operated by NCMEC [30].

We note that similar laws exist in 82 countries around the world such as the 1978 Protection of Children Act [13] and the 1988 Criminal Justice Act [14] in UK. Many countries also have CSAI clearing houses—including Canada with the Canadian Centre for Child Protection [12] and United Kingdom with the Internet Watch Foundation [18].

Our study focuses on NCMEC reports as NCMEC is by far the largest clearing house in the world: In 2016 NCMEC received 8.2 million CSAI reports, compared to 39,714 for the Centre for Child Protection [6] and roughly 105,000 for the Internet Watch Foundation during the same period [20]. The large volume gap between NCMEC and the other entities is due to the fact that many large US companies report all material detected on their platforms to NCMEC, with NCMEC routing reports to the correct law enforcement agency.

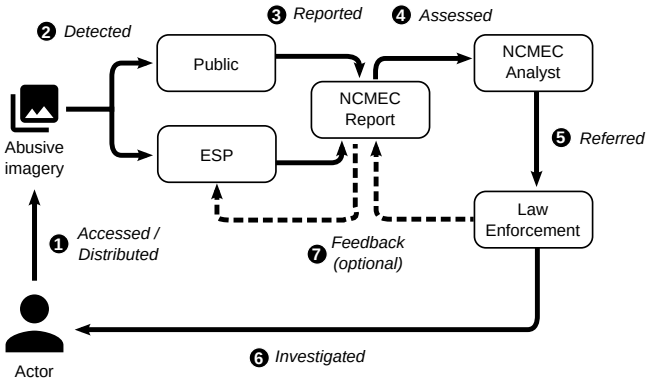


Figure 1: Life cycle of a CSAI report. Upon detection of an actor distributing or accessing abusive imagery, the public or an ESP report the activity to NCMEC. A NCMEC analyst reviews the report and refers actionable leads to law enforcement.

2.2 Life cycle of a NCMEC report

NCMEC's CyberTipline was first launched in 1998 and has since served as a clearing house that coordinates between the public, ESPs, and law enforcement. We provide a rough breakdown of the life cycle of a NCMEC report in Figure 1. Reports first stem from a public entity observing an abusive actor accessing or distributing CSAI material, or by an ESP detecting such activities on their service. Automated detection tools currently include, among others, Microsoft's PhotoDNA for images [29] and Google's CSAI Match for videos [42]. Both tool leverage a variation of the local-sensitive hashing algorithm [10] to not only detect known bad content but also slightly modified versions that abusers produce in an attempt to avoid detection. These CSAI detection technologies are deployed by many companies including Google, Microsoft, Facebook, and Twitter [1, 9, 28, 32].

Once NCMEC receives a report of suspected CSAI, an analyst reviews, augments, and deconflicts the report. As part of this, the analyst manually characterizes the type of incident and the distribution channel involved. The analyst also infers the likely location of the abusive actor and victim, using among multiple factors, the IP address embedded in each abuse report. Ultimately, the analyst assesses the priority of the report for law enforcement referral.

NCMEC refers all US cases to United States state or federal law enforcement. For those reports related to abusive actors or victims outside the United States, NCMEC works with Interpol, Europol, and national police forces. These agencies then investigate reports, take action when appropriate, and may provide optional feedback to NCMEC (and through NCMEC potentially to ESPs) on the outcome. During our study we found out that fewer than 3% of all NCMEC reports have such optional feedback. This precludes any detailed analysis on the actionability of reports.

3 DATASET

Our study builds on a snapshot of anonymized metadata associated with the 23,494,983 NCMEC reports related to suspected CSAI that NCMEC received from March 1, 1998 when the CyberTipline was

created till September 30, 2017. As mentioned earlier these reports come from the US public and many US ESPs. In this section, we provide an overview of the data used through the study and discuss inherent limitations that might bias our study.

3.1 Metadata per Report

Every report consists of at least a timestamp for when NCMEC received the report, the content reported as well as optional metadata such as the IP address of the uploader. As reports have evolved since 1998, more recent reports contain more details. Here are the key metadata fields we use throughout this study:

Source. Reports originate from one of two sources: the public, or electronic service providers (ESPs). For privacy reason, our metadata is limited to these coarse groupings; researchers involved in the study never obtained access to detailed information on reporting entities.

File hashes. Starting in 2013, report metadata contains an MD5 hash of every image, video, and other file attached to the reports. We note that more recent reports include a PhotoDNA hash (discussed in Section 2) and other local sensitive hash, but we did not use these due to the lack of historical coverage.

File format. Starting in 2013, NCMEC's staff began tracking and annotating CSAI content present in each report into one of three file formats: images, videos, and other (e.g., PDFs, HTML). In total, reports reference 20,568,240 abusive images, 2,335,536 abusive videos, and 337,575 other files.

Distribution vector. When possible, NCMEC's staff will annotate each report with a distribution vector. A majority of CSAI distribution vectors are either URLs (90.5%) or unidentifiable due to insufficient report details (8.5%). Other vectors in the remaining 1% of reports include IRC channels, email messages, chat applications, P2P websites, and Tor. We discuss these in further detail in Section 4.

Country. When possible, NCMEC's staff will annotate each report with a country representing either the victim or the abusive actor accessing or distributing content. We note this location information is overloaded: it can reflect the server where CSAI is hosted, the location of the person accessing the CSAI, or even the suspected location of a victim. As such, we are cautious when drawing conclusions about global CSAI trends.

3.2 Limitations

Our dataset is skewed towards US ESPs that proactively scan content using automated tools. As such, our findings likely underestimate the volume of CSAI content that actors access or distribute online. Furthermore, our data is skewed towards the market share of reporting ESPs; for example, China is entirely absent from our dataset. For confidentiality reasons, we are unable to share the identities of any of the ESPs involved or the relative volumes of CSAI reports they contribute. Despite these limitations, we believe that our large-scale dataset provides the first portrait of the evolution of CSAI online and the challenges faced by clearing houses and law enforcement. We discuss these challenges later in Section 5.

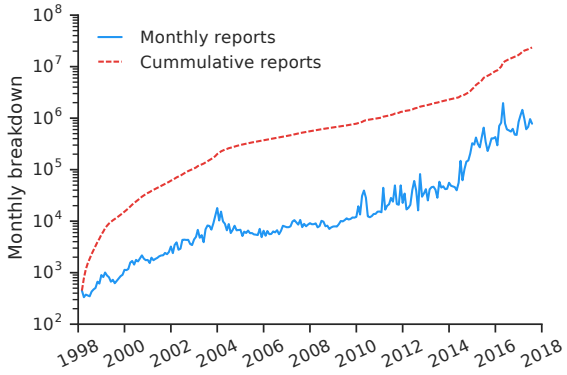


Figure 2: Monthly volume of CSAI reports (log scale) received by NCMEC since the creation of the CyberTipline in 1998. An exponential increase in report volume has lead to 9.6M reports in the last year alone.

3.3 Ethics

Working with CSAI reports presents significant ethical hurdles. Due to the sensitivity and legal protections surrounding reports, the researchers involved in this work never had direct access to reports, or even the metadata of reports. We constructed all measurements based of a report's schema, which were then executed by NCMEC's staff on their infrastructure after vetting by NCMEC's leadership. Additionally, to avoid re-victimizing the children involved in cases reported to NCMEC, and to avoid adding additional burdens to NCMEC's staff, we avoided any reprocessing of old reports. This also precludes analyzing images to understand the nature of abuse, the age of victims, or the uniqueness of the content beyond the hashes present in a report's metadata.

4 ANALYSIS

Our longitudinal measurement study surfaces how reports of CSAI have grown exponentially in parallel with online sharing platforms. In particular, we highlight the global nature of abuse, the adaptation of abusers to new technologies and mediums, and the frequent appearance of new content. Our measurements underscore the challenges of scaling manual review and law enforcement as a response to this threat.

4.1 Exponential Report Growth

Since the founding of the CyberTipline, NCMEC's report volume has grown a median of 51% year-over-year as shown in Figure 2. In aggregate, the public and ESPs have reported over 23.4M suspected incidents of CSAI. Over 9.6M reports (40%) occurred in the year 2017 alone—nearly 1 million per month. For contrast, NCMEC received only 565,000 reports (2.4%) in its first ten years of operation.

A major contributor to the observed exponential growth is the rise of proactive, automated detection efforts by ESPs, as shown in Figure 3. In July, 2009, for the first time the number of reports generated by ESPs exceeded that of the number of public reports. This coincides with the release of Microsoft's PhotoDNA tool (discussed in Section 2). Since then, reporting by ESPs increased an

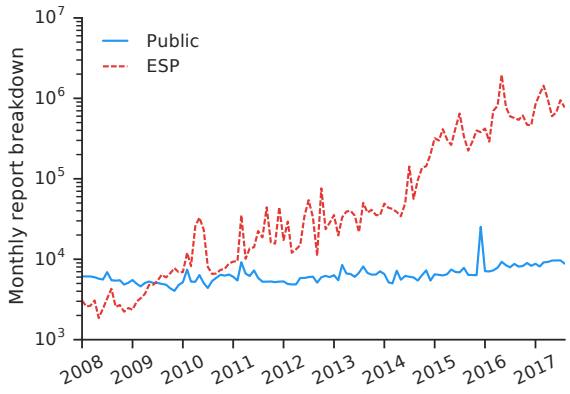


Figure 3: Breakdown of CSAI reports by the public and ESPs. Since 2009, an exponential growth in proactive, automated reporting has overtaken the public as the primary source of reports.

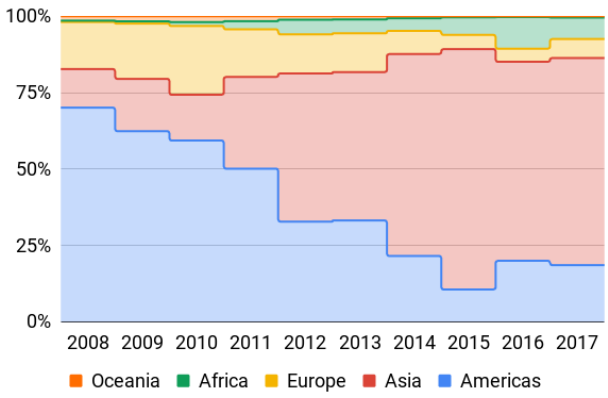


Figure 4: Annual breakdown of CSAI reports by geographic region. NCMEC has transformed from a regional reporting body to a global clearing house of CSAI reports.

average of 101% year-over-year, likely due to increasing user bases and an influx of user-generated content. While automated detection solutions help ESPs scale their protections, law enforcement and NCMEC analysts currently contend with the deluge of reports in a non-automated fashion as they are required to manually reviews the reports.

4.2 Global Threat

Over the last ten years, NCMEC has transformed from a regional reporting body—with 70% of reports related to victims or abusive entities in the Americas and 54% from the United States—to a worldwide clearing house with 68% of its reports relating to Asia, 19% to Americas, 6% to Europe, 7% to Africa, and 0.04% Oceania (Figure 4). Consequently, combating CSAI increasingly requires broad international cooperation with law enforcement.

Table 1: Top 10 countries flagged in reports. To account for population, we also consider the volume of reports per 1000 estimated Internet users for each country.

Country	Reports	Frac. reports	Normalized per 1K capita
India	3,880,235	24.1%	11.9
Indonesia	1,743,362	10.8%	31.0
Thailand	1,706,055	10.6%	63.8
Mexico	1,243,028	7.7%	17.8
Bangladesh	985,347	6.1%	40.5
United States	792,957	4.9%	3.3
Brazil	719,704	4.5%	6.0
Vietnam	701,315	4.3%	14.1
Algeria	631,190	3.9%	41.8
Pakistan	567,850	3.5%	15.8

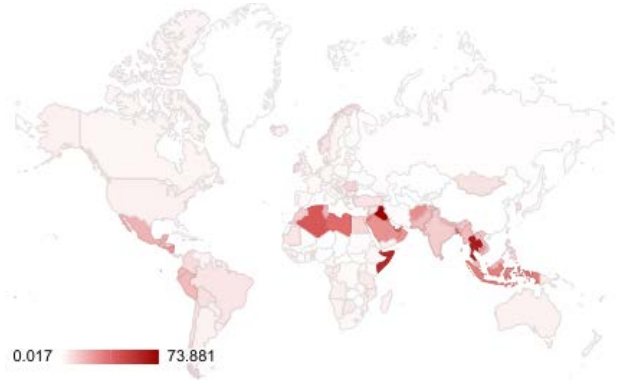


Figure 5: Estimated volume of CSAI reports per one thousand Internet users. South-East Asia, the Middle East, and North Africa have the highest incident rates.

We provide a breakdown of reports by the top 10 locations of victims and abusive entities (when discernible by analysts) in Table 1. As this information is often based on the IP address for abusive entities, we caution it may include VPNs or proxies that skew geographic distributions. We find that India, Indonesia, and Thailand account for roughly 37% of all reports. Both Indonesia and Thailand lack a legal definition for "child pornography" and do not require ISP reporting [18], potentially contributing to CSAI in the region.

As the populations of the top 10 countries differ by an order of magnitude, we also estimate the volume of reports per 1,000 Internet users (based on the International Telecommunication Union Internet penetration estimates for 2015 [19]). After this normalization, the top three countries involved in CSAI include Iraq, Thailand, and Somalia. Both Iraq and Somalia lack any laws related to child sexual exploitation [18]. We visualize these normalized rates across the globe in Figure 5, excluding countries with fewer than 500 reports. We observe higher incident rates of CSAI involvement in South-East Asia, the Middle East, and North Africa. A multitude of

	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	Sample size
FTP	9%	10%	22%	25%	9%	5%	3%	6%	1%	1%	1%	1%	0%	1%	1%	0%	1%	0%	2%	0%	274
Email	0%	1%	1%	2%	9%	16%	18%	8%	3%	4%	6%	4%	3%	4%	3%	4%	5%	4%	3%	2%	86,601
P2P	0%	0%	0%	2%	9%	10%	7%	9%	11%	11%	6%	6%	5%	2%	1%	1%	2%	7%	1%	9%	8,900
Chatroom/IRC	1%	1%	1%	1%	2%	2%	3%	3%	7%	10%	6%	3%	2%	1%	2%	2%	2%	5%	22%	23%	36,086
Instant Messanger	1%	1%	1%	2%	3%	3%	3%	3%	10%	17%	7%	4%	2%	2%	4%	2%	3%	1%	13%	19%	13,733
Forum	2%	4%	3%	3%	4%	4%	4%	5%	5%	7%	3%	6%	7%	24%	2%	2%	2%	2%	7%	4%	30,743
Gaming	0%	0%	0%	0%	0%	0%	0%	0%	0%	12%	18%	5%	3%	3%	7%	8%	9%	9%	15%	10%	3,838
SMS	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	13%	12%	10%	18%	12%	13%	8%	4%	7%	1%	511
Cell phone	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	2%	2%	1%	4%	10%	11%	17%	27%	25%	38,711
Tor	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	2%	3%	5%	26%	42%	22%	4,427
URL	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1%	1%	2%	2%	5%	20%	39%	26%	21,431,212

Figure 6: Evolution of technologies involved in the distribution of CSAI. For example, 25% of all CSAI found on FTP servers was reported in 2001. Conversely, 42% of CSAI content distributed via Tor was reported in 2016. Our findings illustrate how technologies other than URL hosting fall in and out of favor over time.

factors likely play a role in this, including the lack of criminalization [18], the presence of sexual exploitation in conflict zones [8], and entrenched commercial production [22].

4.3 Proliferation of Contexts

As technologies for communicating and sharing over the Internet have shifted, CSAI distribution has evolved in kind (Figure 6). While over 91% of reports since 1998 reference a URL, the minority of other reports provide a lens into how distribution channels fall in and out of favor over time. For example, prior to 2004, abusive entities relied on FTP and email attachments before a rise in peer-to-peer technology. In recent years, we find some sophisticated abusers have adopted Tor (likely to avoid law enforcement authorities), while less sophisticated abusers have favored the convenience of chatroom and messaging apps. One other notable trend is the rise of gaming communities in the last 2–3 years (where children are likely to be participants), both as an enticement platform and as a distribution mechanism. While the overall volume of reports in these mediums is small (possibly due to a lack of ESP monitoring), our data illustrates the broad cooperation necessary among Internet services to tackle CSAI and the adaptability of abusers to new platforms.

4.4 Rise of Videos

In conjunction with video-capable smartphones and the availability of higher bandwidth, we find that CSAI video sharing has dramatically increased from under 1,000 reports per month in 2013 to over 2 million reports per month (Figure 7). In the last year alone, NCMEC observed a 379% year-over-year growth in reported CSAI videos, compared to an 18% increase in CSAI images. In contrast, other reports, such as HTML or PDFs, have remained largely stable over the years at 6,000 reports per month.

This exponential growth of video sharing highlights the necessity for online sharing platforms to develop robust and scalable detection. In particular there is an urgent need for platforms to agree on a standard for video fingerprinting, with Microsoft's PhotoDNA being designed for individual images. At present, ESPs use in-house technologies to flag CSAI videos. Additionally, the size and volume of videos poses a challenge for clearing houses to store and index content. With the advent of virtual and augmented reality, scaling will only increase as a future challenge.

4.5 Uniqueness & Lifetime of Imagery

CSAI content is constantly being transformed and created. Using each reported content's (non-PhotoDNA) hash as an indicator of uniqueness, we find that the public and ESPs report 84% of images and 91% of videos only a single time. Outliers exist, with reports flagging a single image that resurfaced 491,200 times and a single video that resurfaced 5,876 times. Our results illustrate the challenge of identifying new CSAI purely from approximate histograms of previously flagged images; algorithms instead need a deeper understanding of the abusive nature of CSAI content.

Re-distributed content offers an estimate of the lifetime of CSAI. Here, we calculate the elapsed time between the first and last report of CSAI for all hashes reported more than once (Figure 8). We caution this is strictly an underestimate. We find the median lifetime of an image was 257 days, while 10% of images resurfaced over the course of three years. For videos, which just recently emerged as a popular CSAI medium, the median lifetime was 210 days while 10% of videos resurfaced over the course of 1.5 years. Other media is skewed towards long-lived file bundles, where 70% of files resurfaced over the course of 3 years. These findings suggest that once CSAI content is uploaded to the Internet, it continues to resurface, with multiple months elapsing between attempts at distribution.

5 DISCUSSION

Our measurements highlight how the rapid adoption of the latest advancements in Internet platforms and digital mediums allows abusers to create and distribute CSAI across the Internet at an exponential rate. This exponential growth currently threatens our society's ability to combat online child abuse as the rate at which abusive content emerges vastly outpaces the processing capabilities of clearing houses and law enforcement agencies.

At its core this imbalance between abusers and defenders stems from the fact that the development of detection and processing technologies substantially lags behind the development of content creation and sharing capabilities. This section discuss the key technological directions that need to be pursued to reverse this trend.

Scaling the review process. For NCMEC and other clearing houses, a significant amount of time and human labor is spent processing, aggregating, and de-duplicating reports. Recent advancements in deep learning offer the opportunity to scale the review process by

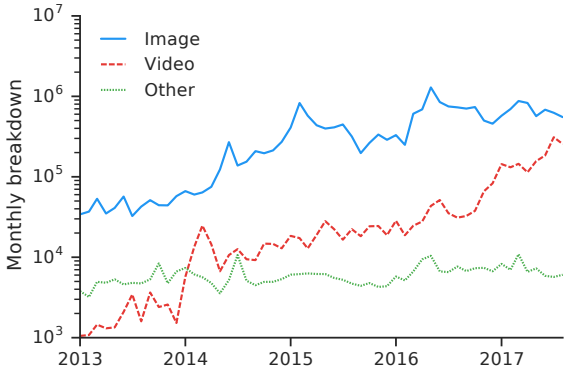


Figure 7: Volume of images, videos, and other file types reported each month since annotations appeared in our dataset in 2013. Simplified video sharing and recording has lead to a 379% increase in CSAI videos in 2017 alone.

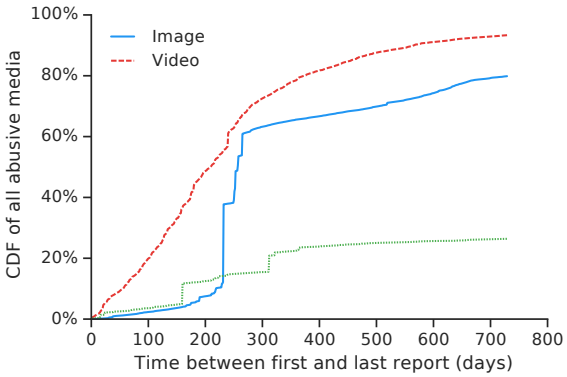


Figure 8: Time elapsed between the first and last report of a unique CSAI hash. We find that a median image has an observed lifetime of 257 days, while videos have a lifetime of 210 days.

automatically grouping reports that have content in common. In particular, two types of clustering could help:

- **Scene clustering:** Content retrieval techniques [11, 27, 41] can be used to group reports coming form various ESP by looking for reports that have similar images/videos that are potentially from the same scene.
- **Facial clustering:** Another possibility involves leveraging facial recognition [7, 15, 37] to find reports that contains the same child or perpetrator. A key challenge here is that traditionally models, including the latest ones based on triplet-loss, have struggled with children's faces because due to less defined facial traits. This represents a challenge both for facial recognition and identifying key attributes such as gender or approximate age.

Scaling the investigation process. A key challenge for law enforcement is prioritizing the reports which are most likely to lead to an arrest. Deep learning techniques can potentially help with prioritization by automatically detecting which content is the most "actionable" from an investigation standpoint. In particular:

- **Facial detection:** Recent progress in facial detection [21, 26] and gender or age prediction [25, 34] lend themselves to automatically reporting the number of victims or perpetrators found in an image. Additionally, it may be possible to extend these technologies to also report if an image contains blurred faces or if identifying features such as hair and eye color are present.
- **Scene information:** Image classifiers can automatically report objects that are present in a scene [39, 40], an image's environment [35, 45], and potentially identify landmarks that will help locate the region where an abusive image originated [43].

Detecting new content. Microsoft's PhotoDNA, originally developed in 2009, is a specialized version of minhashing that fingerprints images based on the histogram of grid-segmented images [29]. This technique is resilient to image transformations related to scale, color, or angle, but is incapable of detecting new CSAI images that have yet to be flagged during manual review. Likewise, the technique does not currently scale to videos. A critical direction here is to develop new algorithms that automatically identify CSAI material based on an image's subject matter. Any such algorithm should work seamlessly across digital mediums, including images and videos.

Reducing reviewer burden. Human analysts currently play a critical role in reviewing CSAI images and videos, both to identify victims, but also to confirm the abusive nature of content. Finding ways to reduce the burden and emotional toll of this task—even through non-technical means—is more pressing than ever given the exponential increase in the amount of content undergoing review.

6 RELATED WORK

Prior investigations of the distribution of CSAI online have largely focused on criminal sharing in peer-to-peer networks. Bissias *et al.* studied roughly 300,000 CSAI torrents and found that, despite law enforcement actions against criminals sharing CSAI, torrents had a survival rate of over 60% even after four years [3]. Our metrics also show a long lifetime of CSAI content: images and videos resurfaced over the course of nearly a year. Other research has tried to estimate the number of criminals accessing CSAI based off of unique IP addresses tied to torrent requests [44], though such metrics are susceptible to DHCP churn from residential machines. In terms of detection and disruption, research has focused either on flagging queries for CSAI content based on known keywords [24, 36] or on targeting major seeders to prevent further distribution [17]. Given the frequency of content shared outside of torrents—our dataset of 23.4M reports included only 8,900 torrents—solutions beyond peer-to-peer networks are critical.

7 CONCLUSION

In this paper, we showed how CSAI on the Internet has outpaced the capabilities of independent clearinghouse analysts and law enforcement to respond. In 2017 alone, NCMEC received 9.6M reports of CSAI, compared to roughly 10,000 reports a year when NCMEC first started in 1998. This exponential growth and the frequency of unique images requires re-imagining CSAI defenses away from

fingerprint-based detection and manual review. Instead, we argued that researchers need to develop algorithms that automatically detect CSAI content, cluster images and videos of victims, and ultimately surface identifying features to help law enforcement. The cost of human review also requires both technical and non-technical solutions that reduce the emotional toll of examining CSAI content. Absent new protections, society will be unable to adequately protect victims of child sexual abuse.

REFERENCES

[1] Charles Arthur. Twitter to introduce photodna system to block child abuse images. https://www.theguardian.com/technology/2013/jul/22/twitter-photodna-child-abuse, 2013.

[2] Jürgen Barth, Lilian Bermetz, Eva Heim, Sven Trelle, and Thomai Tonia. The current prevalence of child sexual abuse worldwide: a systematic review and meta-analysis. *International journal of public health*, 2013.

[3] George Bissias, Brian Levine, Marc Liberatore, Brian Lynn, Juston Moore, Hanna Wallach, and Janis Wolak. Characterization of contact offenders and child exploitation material trafficking on five peer-to-peer networks. *Child Abuse & Neglect*, 2016.

[4] Martin Bright and Tracy McVeigh. This club had its own chairman and treasurer. its business was child abuse. https://www.theguardian.com/uk/2001/feb/11/tracymcveigh.martinbright, 2001.

[5] Angela Browne and David Finkelhor. Impact of child sexual abuse: A review of the research. *Psychological bulletin*, 1986.

[6] Canadian Center for Child Protection. Cybertip.ca 15-year anniversary report. https://www.cybertip.ca/pdfs/CTIP_15thAnniversaryReport_en.pdf, 2017.

[7] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: a deep quadruplet network for person re-identification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2017.

[8] Clar Ni Chonghaile. Millions of iraqi children repeatedly and relentlessly targeted, says un. https://goo.gl/vZ33bg, 2016.

[9] Facebook. Meet the safety team. https://www.facebook.com/notes/facebook-safety/meet-the-safety-team/248332788520844, 2011.

[10] Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al. Similarity search in high dimensions via hashing. In *Vldb*, volume 99, pages 518–529, 1999.

[11] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. End-to-end learning of deep visual representations for image retrieval. *International Journal of Computer Vision*, 124(2):237–254, 2017.

[12] Canada government. Canadian centre for child protection, 1985.

[13] UK government. Protection of children act, 1978.

[14] UK government. Criminal justice act, 1988.

[15] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[16] Thomas J Holt, Kristie R Blevins, and Natasha Burkert. Considering the pedophile subculture online. *Sexual Abuse*, 2010.

[17] Ryan Hurley, Swagatika Prusty, Hamed Soroush, Robert J Walls, Jeannie Albrecht, Emmanuel Cecchet, Brian Neil Levine, Marc Liberatore, Brian Lynn, and Janis Wolak. Measurement and analysis of child pornography trafficking on p2p networks. In *Proceedings of the International Conference on World Wide Web*, 2013.

[18] International Center for Missing and Exploited Children. Child pornography: model legislation and global review. http://www.icmec.org/wp-content/uploads/2016/02/Child-Pornography-Model-Law-8th-Ed-Final-linked.pdf, 2016.

[19] International Telecommunication Union. Percentage of individuals using the Internet. https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx, 2017.

[20] Internet Watch Foundation. IWF annual report 2016. https://www.iwf.org.uk/sites/default/files/reports/2017-04/iwf_report_2016.pdf, 2017.

[21] Huaizu Jiang and Erik Learned-Miller. Face detection with the faster r-cnn. In *Automatic Face & Gesture Recognition (FG 2017), 2017 12th IEEE International Conference on*, pages 650–657. IEEE, 2017.

[22] Christine Joffres, Edward Mills, Michel Joffres, Tinku Khanna, Harleen Walia, and Darrin Grund. Sexual slavery without borders: trafficking for commercial sexual exploitation in india. *International Journal for Equity in Health*, 2008.

[23] Kathleen A Kendall-Tackett, Linda M Williams, and David Finkelhor. Impact of sexual abuse on children: a review and synthesis of recent empirical studies. *Psychological bulletin*, 1993.

[24] Matthieu Latapy, Clémence Magnien, and Raphaël Fournier. Quantifying paedophile queries in a large p2p system. In *Proceedings of INFOCOM*, 2011.

[25] Gil Levi and Tal Hassner. Age and gender classification using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 34–42, 2015.

[26] Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua. A convolutional neural network cascade for face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5325–5334, 2015.

[27] Kevin Lin, Huei-Fang Yang, Jen-Hao Hsiao, and Chu-Song Chen. Deep learning of binary hash codes for fast image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 27–35, 2015.

[28] Microsoft. Microsoft and National Center for Missing and Exploited Children push for action to fight child pornography. https://goo.gl/7oxEtW, 2009.

[29] Microsoft. PhotoDNA cloud service. https://www.microsoft.com/en-us/photodna, 2018.

[30] NCMEC. National center for missing and exploited children, 1984.

[31] Noemí Pereda, Georgina Guilera, Maria Forns, and Juana Gómez-Benito. The prevalence of child sexual abuse in community and student samples: A meta-analysis. *Clinical psychology review*, 2009.

[32] Sarah Perez. Why the gmail scan that led to a man's arrest for child porn was not a privacy violation. https://tcrn.ch/2NlbBrW, 2014.

[33] Frank W Putnam. Ten-year research update review: Child sexual abuse. *Journal of the American Academy of Child & Adolescent Psychiatry*, 2003.

[34] Pau Rodríguez, Guillem Cucurull, Josep M Gonfaus, F Xavier Roca, and Jordi Gonzalez. Age and gender recognition in the wild with deep attention. *Pattern Recognition*, 72:563–571, 2017.

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[36] Moshe Rutgaizer, Yuval Shavitt, Omer Vertman, and Noa Zilberman. Detecting pedophile activity in bittorrent networks. In *Proceedings of the International Conference on Passive and Active Network Measurement*, 2012.

[37] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[38] Marije Stoltenborgh, Marinus H Van Ijzendoorn, Eveline M Euser, and Marian J Bakermans-Kranenburg. A global perspective on child sexual abuse: meta-analysis of prevalence around the world. *Child maltreatment*, 2011.

[39] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.

[40] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[41] Ji Wan, Dayong Wang, Steven Chu Hong Hoi, Pengcheng Wu, Jianke Zhu, Yongdong Zhang, and Jintao Li. Deep learning for content-based image retrieval: A comprehensive study. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 157–166. ACM, 2014.

[42] Nicholas Watt and Juliette Garside. Google to tackle images of child sexual abuse with search and youtube changes. https://www.theguardian.com/technology/2013/nov/18/uk-us-dark-web-online-child-abuse-internet, 2013.

[43] Tobias Weyand, Ilya Kostrikov, and James Philbin. Planet - photo geolocation with convolutional neural networks. *Lecture Notes in Computer Science*, page 37âĂŞ55, 2016.

[44] Janis Wolak, Marc Liberatore, and Brian Neil Levine. Measuring a year of child pornography trafficking by us computers on a peer-to-peer network. *Child Abuse & Neglect*, 2014.

[45] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.

The Illusion of Change: Correcting for Biases in Change Inference for Sparse, Societal-Scale Data

Gabriel Cadamuro*
University of Washington
gabca@cs.washington.edu

Ramya Korlakai Vinayak
University of Washington
ramya@cs.washington.edu

Joshua Blumenstock
University of California Berkeley
jblumenstock@berkeley.edu

Sham Kakade
University of Washington
sham@cs.washington.edu

Jacob N. Shapiro
Princeton University
jns@princeton.edu

ABSTRACT

Societal-scale data is playing an increasingly prominent role in social science research; examples from research on geopolitical events include questions on how emergency events impact the diffusion of information or how new policies change patterns of social interaction. Such research often draws critical inferences from observing how an exogenous event changes meaningful metrics like network degree or network entropy. However, as we show in this work, standard estimation methodologies make systematically incorrect inferences when the event also changes the sparsity of the data.

To address this issue, we provide a general framework for inferring changes in social metrics when dealing with non-stationary sparsity. We propose a plug-in correction that can be applied to any estimator, including several recently proposed procedures. Using both simulated and real data, we demonstrate that the correction significantly improves the accuracy of the estimated change under a variety of plausible data generating processes. In particular, using a large dataset of calls from Afghanistan, we show that whereas traditional methods substantially overestimate the impact of a violent event on social diversity, the plug-in correction reveals the true response to be much more modest.

KEYWORDS

Computational Social Science, Change Detection, Entropy Estimation, Call Detail Records

ACM Reference Format:
Gabriel Cadamuro*, Ramya Korlakai Vinayak, Joshua Blumenstock, Sham Kakade, and Jacob N. Shapiro . 2019. The Illusion of Change: Correcting for Biases in Change Inference for Sparse, Societal-Scale Data. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313722

1 INTRODUCTION

Over the past decade, the increasing availability of societal-scale data has led to new approaches to social science research[5, 11, 24, 38]. In this literature, one common strain of analysis studies the human response to important geo-political events, using digital trace data as a lens into that response. For instance, [36] shows how to rapidly detect an earthquake from Twitter behaviour, [3] uses mobile phone data to study collective response to several different types of emergencies, and [39] studies rumors on social media following an oil spill, to cite just a few examples.

A common methodological challenge in such research is the issue of *sampling sparsity*: where the likelihood of observing any given edge in the social graph during a given period may be low and lead to inaccurate estimates of an individual-level properties. This problem is well-known and there is a rich body of work[19, 32, 35, 41, 43, 46] in both theory and application considering how to better estimate in the presence of sparsity. However, additional and previously unconsidered issues arise when this sparsity may vary over time: we call this property *dynamic sampling sparsity*.

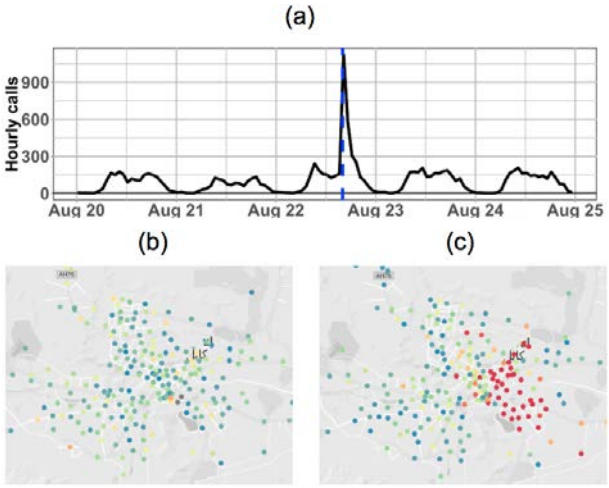


Figure 1: Illustrating variations in sparsity through analysis of call records during a bomb attack in a major city. Graph (a) shows how the hourly call volume of one of the impacted cell towers experiences a very noticeable surge during the emergency. This is also apparent when mapping the tower-level call volume (b) one hour before and (c) one hour after the bombing. The color of each tower represents how abnormally high the call volume is: with red representing call volumes over 5 standard deviations from the mean.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW'19, May 2019, San Francisco, California USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313722

While dynamic sampling sparsity appears in many scenarios, analyzing the impact of emergency events provides a particularly illustrative example. Almost without fail, emergencies produce an immediate spike in transaction log activity (indeed, this spike often serves as the basis for emergency event detection and prediction[8, 18, 22, 36, 50]). However, this means that the sparsity of the social networks decreases at precisely the most confounding time: in the immediate aftermath of the event. An example of the abrupt change in sparsity conditions, derived from anonymized mobile phone data from Afghanistan, in the wake of a serious emergency can be seen in Figure 1. Understanding how important metrics of mobility and social diversity are impacted by such an emergency event, without being misled by the increased volume of communication, now becomes a serious challenge.

Our contribution: This paper shows how dynamic sampling sparsity of digital trace data can systematically bias downstream statistical inferences, and proposes a plug-in correction (namely, a fix that can be applied as a pre-processing step for any existing estimator) to address this problem. In particular:

- We develop a general framework to show why existing methods will systematically produce spurious discoveries. We use this framework to derive a simple statistical correction.
- We benchmark against several state of the art estimators using both real-world and simulated data, under a range of dynamic sparsity conditions. We show that our correction reliably outperforms or matches these methods under all conditions.

The rest of the paper is organized as follows. Section 2 provides necessary technical background and discusses related work. Section 3 introduces a general framework to model the problem and proves that existing methods are biased. We construct a simple plug-in correction to existing estimators that is unbiased. This correction is then put to the test in section 4, where we test its performance on both real-life and synthetic examples and then examine how this alters the conclusions of a sociological analysis. Finally, in Section 5 we discuss the pertinence of our investigation to the broader computational social science community, noting that this problem extends to many scenarios outside of emergency event analysis, and suggest further questions of both practical and statistical relevance.

2 BACKGROUND AND RELATED WORK

2.1 Measuring social phenomena with societal-scale digital trace data

A common approach to current computational social science research involves the analysis of summary statistics that are derived from societal-scale digital trace data. Though the methods we propose apply to a great many such statistics, we begin by introducing a few common metrics which will serve as a running example in the analysis that follows.

The first set of metrics summarize network structure, a generic class of metrics equally applicable to the Twitter re-tweet graph, the DBLP citation network, or a mobile phone network. Specifically, we consider **network degree** (which captures the number of unique connections of each node in the network, also called degree centrality) and **network entropy** (a measure of the dispersion of each individual's network). For any graph, let the number of interactions between node i and node j during a given time period t be $c_{ij}(t)$, and the total volume of i's interactions $c_i(t) = \sum_j c_{ij}(t)$. Degree $D_i(t)$ and network entropy $H_i(t)$ of node i during period t are defined as,

$$D_i(t) = \left|\{j \mid c_{ij}(t) > 0\}\right| \text{ and } H_i(t) = -\sum_j \frac{c_{ij(t)}}{c_i(t)} \log \frac{c_{ij}(t)}{c_i(t)} \quad (1)$$

A second set of metrics, most relevant in networks with geomarkers, capture the characteristic travel distance or diversity of locations visited. Common examples of metrics here include **location entropy** [51] (defined similarly to the network entropy, but over the distribution of locations visited rather than individuals called) for diversity and **radius of gyration** [16] for travel distance.

These network- and location-related metrics have been used in hundreds of papers on dozens of different datasets. For instance, entropy and degree have proven informative in inference tasks ranging from estimating regional unemployment from Twitter usage[25] to predicting wealth from cell-phone records [4, 10]. Related papers show similar results for mobility metrics [6, 15, 30]. In addition to proving useful on this range of societal-scale social networks, several forms of entropy have shown usefulness in aiding visualization of the DBLP citation network[37].

2.2 Estimators and Bias

As the sheer scale of data available increases, it is important to note the growing problem of sparsity. Metrics that require large number of samples from the distribution may be confounded when the number of samples (for instance, the volume of communication) is much smaller than the support size of the distribution (e.g., the number of individuals in the true distribution). This necessitates the use of **estimator** functions that approximate the true underlying metric. Since we are interested in the predictive accuracy of the estimator, we focus mainly on its bias and variance (Equation (2)), the former of which underlies the problem discussed in this paper.

Definition 2.1. Let $\hat{\theta}(Y)$ be an estimator of true parameter θ^* using the data Y. The *bias* and *variance* of $\hat{\theta}$ is defined as,

$$bias(\hat{\theta}) = E[\hat{\theta}] - \theta^*, \quad var(\hat{\theta}) = E[(\hat{\theta} - E[\hat{\theta}])^2]. \quad (2)$$

Note that the expectation E[.] is over the randomness of data.

It is important to note that many key social metrics, including all of those defined in Section 2.1, have serious issues with bias when being estimated. Network degree and any entropy based metric have no unbiased estimator[29]. Obtaining an unbiased estimator for the radius of gyration, since it is related to the standard deviation of locations visited, is known to be a hard problem[47].

2.3 Inferring changes

Detecting and quantifying the impact of an exogenous geopolitical event on a social metric of interest (either over time or across locations) can provide important insight into how such events impact the behavior of larger populations. Examples in the literature include using anomalies detected in social media [20, 23] and mobile phone data [22, 50] to infer the severity and location of damage from natural disasters, or the impact of employment shocks [40]. Many of these difference detection techniques transfer smoothly across data-sets: techniques first applied to social media and communications data can be adapted to a data set as dissimilar as credit card transactions[9].

Non-parametric paired tests are used to detect if there is a systematic change in the mean of a metric of interest, say X, over the same population before and after an event or a treatment. For example, the Wilcoxon signed-rank test takes the paired difference of X before and after an event, ranks them in the order of magnitude and then uses the rank and the sign of the difference (discarding the actual magnitude of difference to avoid the effect of heavy-tail noise) to determine whether a change occurred. However, such tests (implicitly) assume that the bias in measuring in X is the same before and after the event. The proportion of times a paired test detects a change when there is no actual change (null hypothesis) is called the *type I error rate* (α) and when the underlying value did indeed change this proportion is called the *power* (β).

Why bias matters: In contrast to the assumptions of the paired tests, the bias in estimating quantities like entropy depend on the sparsity of the observed distribution. Since the sparsity levels can differ widely before and after an event, the bias in the measurement of entropy will also be different. Therefore, when we take a paired difference, we are not only measuring the change, but also an additional *unknown bias* term that is difficult to isolate. Even when there is no change in entropy, a systematic bias due to dynamic sampling sparsity can lead to a consistently increased rate of type I errors. This is discussed more formally in Section 3. The implications of dynamic sparsity on bias and consequently on the outcome of change detection is discussed Section 3.1. We systematically study this effect for state-of-the-art entropy and support size estimators in Section 4.

2.4 Related work

Estimating the support size, entropy and general symmetric functions[1] of discrete distributions when the number of observations is much smaller than the support size of the distribution is a fundamental problem that has been very well-studied [2, 13, 14, 17]. It is still an active area of research in statistics, information theory as well as computer science [1, 21, 26–28, 31, 34, 41–43, 45, 46, 48, 49]. While this research has improved state-of-the-art estimators, the primary focus has been on estimating a function on a single distribution, rather than the issue of varying sparsity across the network and over time — which are crucial in the applications of interest.

To give a concrete example, the optimal number of samples needed to estimate the entropy of a discrete distribution within ϵ-error is $\Theta\left(\frac{S}{\log S}\frac{1}{\epsilon}\right)$, where S is the support size of the distribution[2] [21]. In practice, we do not have the luxury of soliciting more samples to meet this bound and consequently the estimation of entropy per individual in a network will incur some non-negligible bias. As we will see in Section 4.1, this can lead to systematic inference errors in a way that falls outside of this body of statistical literature.

In contrast to the situation in the statistical literature, the issue of dynamic sparsity arises when analyzing social graphs. This has been an issue in particular when looking at mobility since key metrics like radius of gyration and location entropy have issues with estimator bias. Using a more densely sampled signal that is normally not available, one work[51] showed a systematic underestimation of mobility metrics using call networks that was greater for individuals making few calls. This has also been previously seen in [33] which found that while key locations were generally well inferred, functions like location entropy or radius of gyration likewise had similarly unbalanced issues with bias. However neither of these works offered general solutions to the problem. Heuristics such as dividing a biased metric by the number of communications[7] have no guarantees in improving accuracy: whether they mitigate or aggravate the problem is entirely dependent on the underlying distributions, functions and sample sparsities. A recent work[44] has analyzed the specific bias induced on location metrics by the location-varying tower density and provided correction specially designed for their specific application setting.

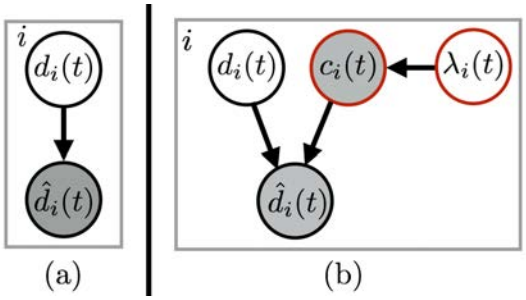


Figure 2: Generative model for the data for a single period of time (a) when sparsity is stationary, and (b) when sparsity is non-stationary.

3 THEORY: DYNAMIC SAMPLING SPARSITY

In the case where sparsity is stationary, the number of samples observed before and after an event is the same on average. The generative model for the observed data is as shown in Figure 2(a), where $d_i(t)$ denotes the true distribution and $\hat{d}_i(t)$ denotes the *observed distribution* for an individual i. However, as motivated in the introduction, the sampling rate or *sparsity is not stationary* (Figure 1). In this section, we describe a general framework to capture the observation model in the setting of dynamic sparsity.

Let $c \sim \text{Poi}(\lambda)$ denote a random variable drawn according to Poisson distribution with rate parameter λ. At time t, let $\lambda_i(t)$ be the rate of sampling for individual i and $c_i(t) \sim \text{Poi}(\lambda_i(t))$ denote the number of samples observed for an individual i. So, we get to observe the empirical distribution $\hat{d}_i(t)$, which is obtained by drawing $c_i(t)$ samples from the true distribution $d_i(t)$. This generative model is illustrated in Figure 2(b).

Let f be the functional (e.g: entropy) we are interested in computing on the distribution d. Let $\hat{f}_i := g\left(\hat{d}_i(t), c_i(t)\right)$ denote the estimator of f for individual i. We note that this estimator is not only dependent on the true underlying distribution $d_i(t)$, but also the sampling rate $\lambda_i(t)$. Therefore, the bias in estimating f_i at time t is also affected by the sampling sparsity, that is,

$$\text{bias}(\hat{f}_i(t)) = \text{E}(\hat{f}_i(t)) - f_i(t) =: B(d_i(t), \lambda_i(t)). \tag{3}$$

Since the sparsity is not stationary, that is, $\lambda_i(\text{after}) \neq \lambda_i(\text{before})$, the *bias itself is not stationary*. Even when $d_i(a) = d_i(b)$, the change in sparsity leads to a systematically increased type-I error rate in

[1] A function over a discrete distribution is said to by a *symmetric function* if it remains invariant to relabeling of domain symbols.
[2] The notation $h(n) = \Theta(g(n))$ means that h is bounded both above and below by g asymptotically.

classical tests like Wilcoxon signed-rank test for detecting change in f as we discuss next.

3.1 Analysis of existing methods

We are interested in detecting and quantifying the difference in f of the distribution $d(t)$ before and after an event. For each individual i, let the difference be denoted by, $\delta_i := f(d_i(a)) - f(d_i(b))$, where a and b stand for *after* and *before* respectively. However, we do not get see the distribution d_i itself, and instead we get to observe $\hat{d}_i$ which has c_i samples from the true distribution d_i. Given an estimator $\hat{f}_i$, the intuitive way to use it to find the difference δ_i is to estimate on the before and after distributions seperately and then take the difference. This gives us the estimator for the change $\hat{\delta}_i := \hat{f}_i(\hat{d}_i(a)) - \hat{f}_i(\hat{d}_i(b))$, where $\hat{f}_i(\hat{d}_i(t))$ denotes the estimate of f on the observed distribution $\hat{d}_i(t)$. Using Equation (3), the expected difference can then be written as,

$$\mathrm{E}(\hat{\delta}_i) = \delta_i + B(d_i(a), \lambda_i(a)) - B(d_i(b), \lambda_i(b)). \tag{4}$$

Under the null hypothesis, the underlying distributions remain the same before and after, i.e, $d_i(a) = d_i(b)$. Therefore, under the null, $\delta_i = 0$. When we test for change, we want to control α (the chance of declaring a change when the null is true). If sparsity was stationary, i.e $\lambda_i(b) = \lambda_i(a)$, the mean of the difference would be zero since bias would cancel when we take paired differences (Equation (4)). However, since the observed distribution also depends on the non-stationary rate parameter $\lambda_i(t)$, the mean of paired difference is not zero under the null. For $\mathrm{E}(\hat{\delta}_i)$ to be zero under the null hypothesis, we need the following to hold, for all $d_i(a), \lambda_i(a)$ and $\lambda_i(b)$,

$$B(d_i(a), \lambda_i(a)) = B(d_i(a), \lambda_i(b)). \tag{5}$$

For functions like entropy, which do not have unbiased estimators [29], such a condition would never hold for any non-trivial distribution d_i and estimator $\hat{f}_i$. This leads to a systematically increased type-I error rate under classic tests like Wilcoxon signed-rank test as illustrated in Figure 3.

3.2 Correction by downsampling

We propose downsampling the observed distributions to same number of samples before estimating f as a plug-in solution to avoid this problem in change detection tests.

Let $c_i^{\min} := \min\{c_i(a), c_i(b)\}$, and $\widetilde{d}_i(a, l)$ and $\widetilde{d}_i(b, l)$ be obtained by drawing $l \le c_i^{\min}$ samples from $\hat{d}_i(a)$ and $\hat{d}_i(b)$ respectively. The downsampling-corrected version of estimator $\hat{f}_i$ for difference is then defined as follows:

$$\tilde{\delta}_i := \mathrm{E}\left(\hat{f}(\widetilde{d}_i(a, l))\right) - \mathrm{E}\left(\hat{f}(\widetilde{d}_i(b, l))\right), \tag{6}$$

where the expectation is over the randomness of drawing $\widetilde{d}_i \sim \hat{d}_i$. In practice this can be approximated by averaging over a few random re-samplings. Downsampling ensures that under the null hypothesis, the bias in estimating f is same for before and after and hence it cancels out when we take paired difference. The situation where the null hypothesis is false is significantly harder to analyze but the performance of the proposed correction in this case is explored empirically in Section 4.2.

4 EMPIRICAL ANALYSIS

To verify the pertinence of this problem in real-life analysis we perform a number of empirical studies. In all of these we focus on

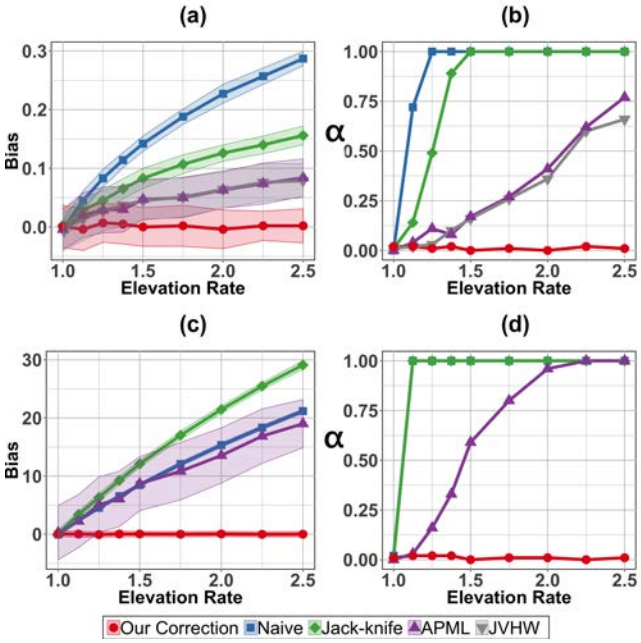


Figure 3: Comparison of how the (a) bias and (b) type-I error rate (α) for estimating difference in entropy increases with more variation in sparsity. (c) and (d) show the same for network degree. The bands in (a) and (c) show the variance in estimates of the average difference.

inferring the change in two metrics: social entropy and network degree. We pick these two since they are both socially informative as well as ubiquitously available over many different types of social graphs. We are interested in how estimates in the change of these metrics are impacted by the variation in sparsity, which we quantify as the *elevation rate* r.

$$\text{Elevation Rate, } r := \frac{\lambda(\text{after})}{\lambda(\text{before})}.$$

We compare the performance of the following four estimators:

(1) **Naive-Estimator**: This simply computes the metric by treating the empirical distribution as the true distribution.
(2) **Jackknifed naive**[12]: This is the naive estimator with a jackknife heuristic that averages the naive estimate over all distribution generated by removing one sample from the empirical distribution.
(3) **JVHW**[21]: This estimator combines an unbiased estimator for the best polynomial approximation of the function being estimated in the non-smooth region with a bias-corrected estimate on the smooth region.
(4) **APML**[31]: Approximate Profile Maximum Likelihood Estimator is a computationally efficient approximation of the profile maximum likelihood [1] which maximizes the probability of the observed profile (multiplicities of the symbols observed ignoring the label).

Note that JVHW is only applicable to entropy and not network degree. We compare these estimators to their *corrected* variants, where we downsample the data (as described in Section 3.2) before running these estimators. We found the results broadly similar across the corrected version of these four methods. In the interest

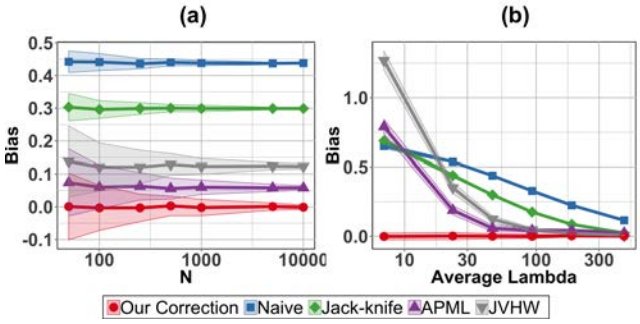


Figure 4: Analysis of how (a) varying the number of individuals N and (b) average sparsity impacts entropy bias. Both experiments are run on synthetic data generated from the Dirichlet distribution with elevation rate $r = 2$. The former only improves the variance while the latter decreases bias as the average sparsity drops (as calling rate increases).

of clarity we only show one corrected estimator per graph: the JVHW-correction for entropy and the jack-knifed correction for network degree.

In all of these experiments we ask two questions. Firstly, what is the bias in the estimated difference for each estimator under different values of elevation rate r? Secondly, how does this translate into type-I and type II errors? The first question is simply done by computing the average predicted change and comparing it to the actual average change. The second question is studied by applying a Wilcoxon signed-rank test to the estimated differences with a desired α of 0.01.

4.1 Natural experiments with real-world data

In the first set of experiment we use a country-wide CDR dataset collected over 6 months in Afghanistan. This data comprises data for millions of callers and since our interest concerns changes in specific groups of individuals we restricted this to calls from a set of $N = 1000$ individuals determined to be living near a specific tower in a major city. We take the empirical distribution generated by 6 months of data (with a median of 500 calls per individual) as being sufficiently well sampled to approximate the true social distributions $\{d_i\}$'s and call rates $\{\lambda'_i\}$'s.We take the empirical call rates for six months and scale them down to the equivalent rate for a week $\lambda_i = \frac{7}{180}\lambda'_i$. We then assign before and after distributions to be identically $d_i(a) = d_i(b) = d_i$ and $\lambda_i(a) = \lambda_i$ but we multiply the second calling rate by the elevation rate: $\lambda_i(b) = r\lambda_i$. We repeat 100 trials where we sample using these distributions and λ's as in Figure 2(b) and compare the estimated difference between the metric average of sets a and b. We run the Wilcoxon signed-rank test and check if it detects a change. Since the distributions are the same, ideally we would like to estimate that there is no difference. Figure 3 shows the results for social entropy and network degree: though the same trend is present in both. We clearly see that all the methods that do not correct for varying sparsity, including cutting-edge estimation techniques like JVHW and APML, reveal substantial issues with bias at even modest elevation rates which get progressively worse as the rate increases. In contrast, our corrected method consistently returns the correct result no matter the level of imbalance in sparsity.

4.2 Synthetic tests

While experiments on real data are essential to proving the practical concerns around the sampling problem they only provide a fixed set of conditions to experiment with. For this reason we created a synthetic test suite that would allow us to compare our methods against baselines on a variety of distributions and at a significantly more granular level. This allows us to directly set $d_i(a)$ and $d_i(b)$ to both explore different distributions and also be significantly different. As such we can compute the bias of estimators when $E[\delta_i] \neq 0$ as well as for the null case where $E[\delta_i] = 0$.

The experiment then proceeds similarly to section 4.2: with the exception that λ_i and $d_i(a)$ are drawn randomly from a prior distribution. λ_i is consistently distributed by a log-normal with mean of 50 while we perform separate experiments where the $d_i(a)$ are drawn from a distribution of either Dirichlet (with Dirichlet parameter $\alpha_D = 1.0$), geometric (with average probability of success $p = 0.9$) or uniform distributions. For the case where we wish $E[\delta_i] \neq 0$, we additionally alter the parameter of $d_i(a)$ by some fixed amount to generate $d_i(b)$.

We note that the existing methods will have substantial bias in the null case no matter how large the population N is, as shown in Figure 4(a). The variance decreases as a function N, but not the bias. We set $N = 1000$ for the remainder of our synthetic experiments. Figure 4(b) shows that decreasing the sparsity, or equivalently increasing the observation rate λ, of course helps all methods: though as previously noted this is rarely possible in practice.

We investigate how the estimators perform in the case of both no change and some change: a subset of our results are shown in Figure 5. Our results for the null case reinforce our conclusions in Section 4.1: there is considerable variance between the different distributions and uncorrected metrics but our correction consistently return an accurate estimate (Figure 5(a—d)). This illustrates the difficulty of the problem when not accounting for variable sparsity: a non-corrected method that seems to work on one distribution may entirely fail on another. We also record how often the Wilcoxon singed-rank test records a true positive as a function of the actual average difference. We see that the elevation rate has induced an asymmetric change in non-corrected methods and hence worse discovery rates when the true change in entropy is negative. On the other hand, the corrected method is reliable through-out (Figure 5(e—h)). Even in a situation where a given uncorrected method perform well (notably, the APML method is fairly robust in the uniform scenario for both network degree and entropy), the corrected method has comparable or better sensitivity while outperforming it in all other situations. This provides strong evidence that the plug-in correction is an improvement also in the case where there is a difference.

4.3 Analysis of sociological events

In this section, we highlight the relevance of this problem to computation social science by demonstrating how it can alter the conclusion of a real analysis. Recalling the call dataset described in Section 4.1, we cross-referenced calls made in that set with the time and location of a serious bomb attack and generated a set of 220 individuals who appear to live in the vicinity of this attack. Our goal is now to analyze how the average network entropy changes

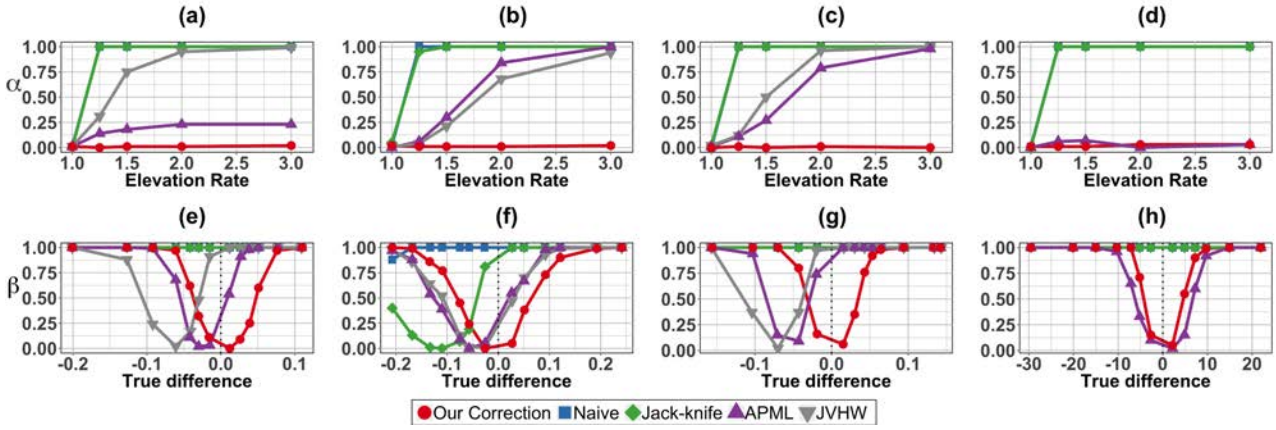


Figure 5: Panels (a)-(c): Experiments showing type-I error rates (α) for entropy change for the uniform, geometric and dirichlet scenarios respectively. Panel (d): Type-I error rate for network degree under the uniform scenario. Panels (e)-(g): Power (β) for entropy change detection at an elevation rate of 3 for the uniform, geometric and dirichlet scenarios respectively (h): Power for network degree under the uniform scenario and elevation rate of 3.

in the immediate aftermath of this emergency. For each 24-hour period in our date range we take the difference with respect to the same period one week before. For example the 24-hour period starting on August 22nd 10am is paired with the 24-hour period starting on August 15th 10am, the one starting at August 22nd 11am is paired with that starting on August 15th 11am etc. We then compare how different methods infer changes based on these differences: our results are shown in Figure 6. While both the basic methods and our correction to JVHW method detect an increase during the emergency period, the uncorrected methods detect anywhere from twice to three times as much of a change. Moreover, the corrected method finds only one 24-hour period to be statistically significantly different: while the other methods declare almost the entire period to show a significant increase in network-entropy.

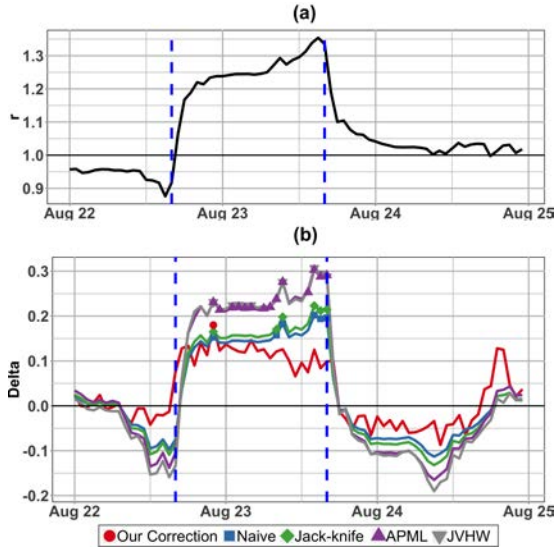


Figure 6: Analysis of (b) how different methods infer the change in network entropy in (a) the presence of varying sampling sparsity caused by a violent event. The period between the dotted blue lines indicate when the sliding window contains the bomb blast period. Marked points in (b) indicate a statistically significant difference between this 24-hour period and the same 24-hour period one week prior.

5 CONCLUSION

This paper explains and formalizes the concept of *dynamic sampling sparsity*, and highlights why it is such an important problem for estimation and change detection. Our statistical framework shows that failing to account for varying sparsity in the data frequently leads to systematic errors in the downstream statistical analysis. We demonstrate the severity of this issue through experiments on both real social graph datasets and comprehensive synthetic tests.

While we motivated this problem by considering the real-world problem of understanding the impact of emergency events, we note that this problem of varying sparsity is significantly broader. Indeed the issue would likely arise when comparing average values of social metrics (whose bias gets influenced by sampling sparsity) between two different populations with different sampling sparsity rates. Examples in the literature include comparing the structure of social networks in urban locations with that of provincial villages [10], or wealthy provinces to a poorer ones [11, 25]. Our empirical results show that it is very hard to determine ahead of time how much a specific scenario will be affected: the impact is a complex function of the different sparsity rates, the underlying distributions and the estimators themselves. The correction we develop can help avoid such errors in arbitrary environments.

A broader implication of the results in this paper is that great care is needed when performing empirical analysis on societal-scale datasets with non-stationary sampling sparsity. Many common distributional tests fail when two distributions are generated from different sparsity regimes. Rather than applying one-off fixes to each such biased metric, more research is needed into optimal statistical detection, estimation and inference tools for large-scale heterogeneous and sparse datasets.

ACKNOWLEDGMENTS

This research was supported by the National Science Foundation Grant under award CCF - 1637360 (Algorithms in the Field) and by the Office of Naval Research (Minerva Initiative) under award N00014-17-1-2313.

REFERENCES

[1] Jayadev Acharya, Hirakendu Das, Alon Orlitsky, and Ananda Theertha Suresh. 2017. A unified maximum likelihood approach for estimating symmetric properties of discrete distributions. In *International Conference on Machine Learning*. 11–21.

[2] John Aldrich et al. 1997. RA Fisher and the making of maximum likelihood 1912-1922. *Statistical science* 12, 3 (1997), 162–176.

[3] James P Bagrow, Dashun Wang, and Albert-Laszlo Barabasi. 2011. Collective response of human populations to large-scale emergencies. *PloS one* 6, 3 (2011), e17680.

[4] Joshua E Blumenstock. 2015. Calling for better measurement: Estimating an individual's wealth and well-being from mobile phone transaction records. (2015).

[5] Ray M Chang, Robert J Kauffman, and YoungOk Kwon. 2014. Understanding the paradigm shift to computational social science in the presence of big data. *Decision Support Systems* 63 (2014), 67–80.

[6] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1082–1090.

[7] Yves-Alexandre de Montjoye, Luc Rocher, Alex Sandy Pentland, et al. 2016. bandicoot: A python toolbox for mobile phone metadata. *J Machine Learn Res* 17 (2016), 1–5.

[8] Adrian Dobra, Nathalie E Williams, and Nathan Eagle. 2015. Spatiotemporal detection of unusual human population behavior using mobile phone data. *PloS one* 10, 3 (2015), e0120449.

[9] Xiaowen Dong, Joachim Meyer, Erez Shmueli, Burçin Bozkaya, and Alex Pentland. 2018. Methods for quantifying effects of social unrest using credit card transaction data. *EPJ Data Science* 7, 1 (2018), 8.

[10] Nathan Eagle, Yves-Alexandre de Montjoye, and Luís MA Bettencourt. 2009. Community computing: Comparisons between rural and urban societies using mobile phone data. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, Vol. 4. IEEE, 144–150.

[11] Nathan Eagle, Michael Macy, and Rob Claxton. 2010. Network diversity and economic development. *Science* 328, 5981 (2010), 1029–1031.

[12] Bradley Efron and Charles Stein. 1981. The jackknife estimate of variance. *The Annals of Statistics* (1981), 586–596.

[13] Bradley Efron and Ronald Thisted. 1976. Estimating the number of unseen species: How many words did Shakespeare know? *Biometrika* 63, 3 (1976), 435–447.

[14] Ronald A Fisher, A Steven Corbet, and Carrington B Williams. 1943. The relation between the number of species and the number of individuals in a random sample of an animal population. *The Journal of Animal Ecology* (1943), 42–58.

[15] Vanessa Frias-Martinez and Jesus Virseda. 2012. On the relationship between socio-economic factors and cell phone usage. In *Proceedings of the fifth international conference on information and communication technologies and development*. ACM, 76–84.

[16] Marta C Gonzalez, Cesar A Hidalgo, and Albert-Laszlo Barabasi. 2008. Understanding individual human mobility patterns. *nature* 453, 7196 (2008), 779.

[17] IJ Good and GH Toulmin. 1956. The number of new species, and the increase in population coverage, when a sample is increased. *Biometrika* 43, 1-2 (1956), 45–63.

[18] Didem Gundogdu, Ozlem D Incel, Albert A Salah, and Bruno Lepri. 2016. Countrywide arrhythmia: emergency event detection using mobile phone data. *EPJ Data Science* 5, 1 (2016), 25.

[19] Sahar Hoteit, Guangshuo Chen, Aline Viana, and Marco Fiore. 2016. Filling the gaps: On the completion of sparse call detail records for mobility analysis. In *Proceedings of the Eleventh ACM Workshop on Challenged Networks*. ACM, 45–50.

[20] Muhammad Imran, Carlos Castillo, Fernando Diaz, and Sarah Vieweg. 2015. Processing social media messages in mass emergency: A survey. *ACM Computing Surveys (CSUR)* 47, 4 (2015), 67.

[21] Jiantao Jiao, Kartik Venkat, Yanjun Han, and Tsachy Weissman. 2015. Minimax estimation of functionals of discrete distributions. *IEEE Transactions on Information Theory* 61, 5 (2015), 2835–2885.

[22] Ashish Kapoor, Nathan Eagle, and Eric Horvitz. 2010. People, Quakes, and Communications: Inferences from Call Dynamics about a Seismic Event and its Influences on a Population.. In *AAAI spring symposium: artificial intelligence for development*.

[23] Sejeong Kwon, Meeyoung Cha, and Kyomin Jung. 2017. Rumor detection over varying time windows. *PloS one* 12, 1 (2017), e0168344.

[24] David Lazer, Alex Sandy Pentland, Lada Adamic, Sinan Aral, Albert Laszlo Barabasi, Devon Brewer, Nicholas Christakis, Noshir Contractor, James Fowler, Myron Gutmann, et al. 2009. Life in the network: the coming age of computational social science. *Science (New York, NY)* 323, 5915 (2009), 721.

[25] Alejandro Llorente, Manuel Garcia-Herranz, Manuel Cebrian, and Esteban Moro. 2015. Social media fingerprints of unemployment. *PloS one* 10, 5 (2015), e0128692.

[26] Alon Orlitsky, NP Santhanam, Krishnamurthy Viswanathan, and Junan Zhang. 2005. Convergence of profile based estimators. In *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*. IEEE, 1843–1847.

[27] Alon Orlitsky, Narayana P Santhanam, Krishnamurthy Viswanathan, and Junan Zhang. 2004. On modeling profiles instead of values. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*. AUAI Press, 426–435.

[28] Alon Orlitsky, Ananda Theertha Suresh, and Yihong Wu. 2016. Optimal prediction of the number of unseen species. *Proceedings of the National Academy of Sciences* 113, 47 (2016), 13283–13288.

[29] Liam Paninski. 2003. Estimation of entropy and mutual information. *Neural computation* 15, 6 (2003), 1191–1253.

[30] Luca Pappalardo, Dino Pedreschi, Zbigniew Smoreda, and Fosca Giannotti. 2015. Using big data to study the link between human mobility and socio-economic development. In *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE, 871–878.

[31] Dmitri S Pavlichin, Jiantao Jiao, and Tsachy Weissman. 2017. Approximate profile maximum likelihood. *arXiv preprint arXiv:1712.07177* (2017).

[32] Aditi Raghunathan, Greg Valiant, and James Zou. 2017. Estimating the unseen from multiple populations. *arXiv preprint arXiv:1707.03854* (2017).

[33] Gyan Ranjan, Hui Zang, Zhi-Li Zhang, and Jean Bolot. 2012. Are call detail records biased for sampling human mobility? *ACM SIGMOBILE Mobile Computing and Communications Review* 16, 3 (2012), 33–44.

[34] Sofya Raskhodnikova, Dana Ron, Amir Shpilka, and Adam Smith. 2009. Strong lower bounds for approximating distribution support size and the distinct elements problem. *SIAM J. Comput.* 39, 3 (2009), 813–842.

[35] Hassan Saif, Miriam Fernández, Yulan He, and Harith Alani. 2014. On stopwords, filtering and data sparsity for sentiment analysis of twitter. (2014).

[36] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*. ACM, 851–860.

[37] Ekrem Serin and Selim Balcisoy. 2012. Entropy based sensitivity analysis and visualization of social networks. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*. IEEE Computer Society, 1099–1104.

[38] Emma S Spiro. 2016. Research opportunities at the intersection of social media and survey data. *Current Opinion in Psychology* 9 (2016), 67–71.

[39] Emma S Spiro, Sean Fitzhugh, Jeannette Sutton, Nicole Pierski, Matt Greczek, and Carter T Butts. 2012. Rumoring during extreme events: A case study of Deepwater Horizon 2010. In *Proceedings of the 4th Annual ACM Web Science Conference*. ACM, 275–283.

[40] Jameson L Toole, Yu-Ru Lin, Erich Muehlegger, Daniel Shoag, Marta C González, and David Lazer. 2015. Tracking employment shocks using mobile phone data. *Journal of The Royal Society Interface* 12, 107 (2015), 20150185.

[41] Gregory Valiant and Paul Valiant. 2011. Estimating the unseen: an n/log (n)-sample estimator for entropy and support size, shown optimal via new CLTs. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*. ACM, 685–694.

[42] Gregory Valiant and Paul Valiant. 2011. The power of linear estimators. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*. IEEE, 403–412.

[43] Paul Valiant and Gregory Valiant. 2013. Estimating the unseen: improved estimators for entropy and other properties. In *Advances in Neural Information Processing Systems*. 2157–2165.

[44] Maarten Vanhoof, Willem Schoors, Anton Van Rompaey, Thomas Ploetz, and Zbigniew Smoreda. 2018. Comparing Regional Patterns of Individual Movement Using Corrected Mobility Entropy. *Journal of Urban Technology* (2018), 1–35.

[45] Shashank Vatedka and Pascal O Vontobel. 2016. Pattern maximum likelihood estimation of finite-state discrete-time Markov chains. In *Information Theory (ISIT), 2016 IEEE International Symposium on*. IEEE, 2094–2098.

[46] Pascal O Vontobel. 2012. The Bethe approximation of the pattern maximum likelihood distribution. In *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*. IEEE.

[47] Wikipedia contributors. 2018. Unbiased estimation of standard deviation — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Unbiased_estimation_of_standard_deviation&oldid=854784987 [Online; accessed 5-November-2018].

[48] Yihong Wu and Pengkun Yang. 2015. Chebyshev polynomials, moment matching, and optimal estimation of the unseen. *arXiv preprint arXiv:1504.01227* (2015).

[49] Yihong Wu and Pengkun Yang. 2016. Minimax rates of entropy estimation on large alphabets via best polynomial approximation. *IEEE Transactions on Information Theory* 62, 6 (2016), 3702–3720.

[50] William Chad Young, Joshua E Blumenstock, Emily B Fox, and Tyler H McCormick. 2014. Detecting and classifying anomalous behavior in spatiotemporal network data. In *Proceedings of the 2014 KDD workshop on learning about emergencies from social information (KDD-LESI 2014)*. 29–33.

[51] Ziliang Zhao, Shih-Lung Shaw, Yang Xu, Feng Lu, Jie Chen, and Ling Yin. 2016. Understanding the bias of call detail records in human mobility research. *International Journal of Geographical Information Science* 30, 9 (2016), 1738–1762.

Rating Augmentation with Generative Adversarial Networks towards Accurate Collaborative Filtering

Dong-Kyu Chae
Hanyang University
Seoul, Korea
kyu899@hanyang.ac.kr

Jin-Soo Kang
Hanyang University
Seoul, Korea
jensoo7023@hanyang.ac.kr

Sang-Wook Kim*
Hanyang University
Seoul, Korea
wook@hanyang.ac.kr

Jaeho Choi
NAVER Corporation
SeongNam, Korea
choi.jaeho@navercorp.com

ABSTRACT

Generative Adversarial Networks (GAN) have not only achieved a big success in various generation tasks such as images, but also boosted the accuracy of classification tasks by generating additional labeled data, which is called *data augmentation*. In this paper, we propose a Rating Augmentation framework with GAN, named RAGAN, aiming to alleviate the *data sparsity problem* in *collaborative filtering* (CF), eventually improving recommendation accuracy significantly. We identify a unique challenge that arises when applying GAN to CF for rating augmentation: naive RAGAN tends to generate values *biased towards high ratings*. Then, we propose a refined version of RAGAN, named RAGAN^{BT}, which addresses this challenge successfully. Via our extensive experiments, we validate that our RAGAN^{BT} is really effective to solve the data sparsity problem, thereby providing existing CF models with great improvement in accuracy under various situations such as basic top-N recommendation, long-tail item recommendation, and recommendation to cold-start users.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Collaborative filtering, generative adversarial networks, data sparsity, data augmentation, top-N recommendation

ACM Reference Format:
Dong-Kyu Chae, Jin-Soo Kang, Sang-Wook Kim, and Jaeho Choi. 2019. Rating Augmentation with Generative Adversarial Networks towards Accurate Collaborative Filtering. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313413

*Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313413

1 INTRODUCTION

In recommender systems, *collaborative filtering* (CF) is one of the most widely used frameworks. The aim of CF is to generate a personalized item recommendation to a user based on her prior preference. Users' past preferences are typically presented in the form of a user-item *rating matrix*, where each cell value contain a rating given by a user for an item, mostly within a specific range (e.g., [1, 5]). However, the majority of entries in a rating matrix are missing: this *data sparsity problem* [13] is a universal obstacle to most CF methods, making them difficult to capture true relationships between users and items, thereby causing unsatisfactory accuracy in recommendation. To mitigate such a problem, most studies try to acquire additional information from social/trust networks among users [24] and auxiliary content of items [32]. However, their methods are applicable only when such additional data is available. The goal of this work is to investigate a generic solution to alleviate the sparsity issue in the context of pure CFs, i.e., a recommendation task given with only the user-item interaction data.

Recently, Goodfellow et al. [9] proposed the *Generative Adversarial Networks* (GAN), which learn to generate samples that mimic the distribution of the training data. Through an adversarial process between a *generative model* G and a *discriminator model* D, G learns to capture the data distribution in order to generate *synthetic but realistic* data. GAN has been very successful in image generation [25], and has gained popularity in performing other tasks, such as natural language generation [35] and music generation [7]. In addition, GAN's impressive ability to generate synthetic but realistic data has been used to improve the accuracy of a classification task, where the synthetic data is used as additional training data, which is called *data augmentation* [1, 2]. For example, [8] synthesized images labeled with Liver Lesion using GAN, resulting in 7% increase in terms of accuracy in Liver Lesion classification.

Motivated by such a success of data augmentation with GAN, this paper explores the *use of GAN for relieving the data sparsity problem in CF* under the context of data augmentation. We expect that using GAN to augment plausible ratings to a rating matrix would help CF models to provide more accurate recommendation by learning the augmented rating matrix, rather than the original one. Along this line, we propose a novel Rating Augmentation framework based on GAN, named RAGAN. RAGAN first trains GAN by using the observed ratings in a rating matrix as a training data, and

then makes its G generate plausible ratings for unobserved user-item interactions, which are to be augmented in the original rating matrix. Finally, conventional CF models can be trained by using the augmented rating matrix to produce final recommendations.

When applying this idea of data augmentation by GAN to the field of CF, we face a unique challenge that comes from the inherent *selection bias* of the rating data. We observed that only a small fraction (i.e., 10−17%) of ratings are low values (i.e., 1 or 2) and the rest are high values (i.e., 3, 4, or 5) from four real-life datasets used in our experiments, which has been also noticed in other studies [6, 12]. This might be because people have a tendency to select (and thus to rate) the items likely to give high satisfaction to them, and to disregard those items likely to result in low satisfaction (i.e., selection bias [6]). If GAN learns from the original rating matrix skewed to such high ratings, it would also generate *high ratings for most unobserved entries* in the matrix. However, this augmentation result seems not satisfactory since the majority of unobserved entries are likely to indicate users' negative preferences; in other words, users would not give a rating to those items that they are not interested in. If CF models are trained with such a *biased* augmented rating matrix, they would show poor recommendation accuracy since they fail to capture users' true preferences. We will demonstrate this phenomenon clearly through our experiments.

Towards this challenge, we propose RAGAN$^{\text{BT}}$, a refined version of RAGAN, which takes users' *negative items* (i.e., the items having users' negative preferences) additionally into account when training GAN, in order to avoid G from being biased towards high ratings in rating augmentation. Here, we formulate the problem of identifying such negative items as the well-known *one-class collaborative filtering* (OCCF) [23, 34] that infers the interestingness (or uninterestingness) of every user-item pair. By training GAN with the identified negative items as an additional source together with the original rating matrix, RAGAN$^{\text{BT}}$ successfully avoids the bias towards high ratings, and thus could generate the plausible ratings that better reflect the characteristics of unrated items. As a result, by augmenting the plausible ratings to the rating matrix, RAGAN$^{\text{BT}}$ solves the data sparsity problem, thereby contributing towards significant improvements of accuracy in CF.

In order to validate the effectiveness of our RAGAN$^{\text{BT}}$, we conduct extensive experiments using four real-world datasets. Notably, the results demonstrate that the key idea behind RAGAN$^{\text{BT}}$ (i.e., involving negative items in training GAN) is quite effective in generating more realistic ratings, leading to higher recommendation accuracy. The results also demonstrate that our RAGAN$^{\text{BT}}$ consistently and universally outperforms the state-of-the-art CF models not only under basic top-N recommendations, but also under more-difficult situations: (1) long-tail item recommendations and (2) recommendations to cold-start users. We believe our RAGAN$^{\text{BT}}$ sheds a light on the practical adoption of GAN for alleviating the data sparsity problem, which has been a universal obstacle to accuracy improvement in recommender systems.

2 PRELIMINARIES

Let two sets be denoted as $U = \{u_1, u_2, .., u_m\}$ and $I = \{i_1, i_2 ..., i_n\}$, where U consists of m users and I contains a set of n items. I_u is defined as a set of items that a user u has rated. A corresponding m-by-n rating matrix is $\mathbb{R}$, each entry r_{ui} is a rating on item i given by user u, and the ratings are within a specific range (e.g., [1, 5]). We denote $\mathbf{r_u}$ and $\mathbf{r_i}$ as rating vectors of user u and item i, respectively.

Given complicated real-world data, the *Generative Adversarial Network* (GAN) allows for a novel training method for generative models. GAN is made up of two models (neural networks in general), where one is a generative model, G, and the other is a discriminative model, D. The training process is a minimax game between G and D (i.e., $\min_G \max_D \mathcal{L}_{GAN}$): G tries to increase the error rate of D by generating seemingly authentic data, while D tries to accurately distinguish whether the data has come from the ground truth (i.e., real data) or from G. Formally, the objective function is:

$$\mathcal{L}_{GAN} = \mathbb{E}_{\mathbf{x}\sim p_{data}}[\log D(\mathbf{x})] - \mathbb{E}_{\hat{\mathbf{x}}\sim p_G}[\log(D(\hat{\mathbf{x}}))], \qquad (1)$$

where $\mathbf{x}$ is a ground-truth data from the data distribution p_{data} while $\hat{\mathbf{x}}$ is a synthetic data from the model distribution p_G, typically derived by $\hat{\mathbf{x}} = \text{G}(\mathbf{z})$ where $\mathbf{z}$ is a random noise vector. D$(\cdot)$ indicates the estimated probability of its input being a ground truth [4].

3 RAGAN

The main idea of this paper is to train GAN with the rating data, generate plausible ratings, and augment the generated values in the rating matrix. This section illustrates a basic solution to this objective, called RAGAN (rating augmentation framework based on GAN). As a first step, we need to train GAN by using the rating data. In this context, there exist several recently proposed GANs that could be proper candidates for our framework, such as IRGAN [33], GraphGAN [31], and CFGAN [3], all of which focus on recommendation tasks. Among them, we note that both IRGAN and GraphGAN can only learn from users' implicit feedback data, and expanding these methods to also learn from explicit ratings (i.e., the degree of satisfaction) and generate such data is non-trivial. In contrast, CFGAN can be easily extended to learn from and generate the rating data, even though it was also proposed to perform CF based on implicit feedback dataset. For this reason, our GAN for rating data is based on CFGAN.

Formally, $\mathcal{L}_{GAN}$ in our work is defined as:

$$\begin{aligned}\mathcal{L}_{GAN} &= \mathbb{E}_{\mathbf{x}\sim P_{data}}[lnD(\mathbf{x}|c)] - \mathbb{E}_{\hat{\mathbf{x}}\sim P_G}[lnD(\hat{\mathbf{x}}|c)] \\ &= \frac{1}{m}\Big(\Sigma_u lnD(\mathbf{r_u}|\mathbf{r_u}) - \Sigma_u lnD(\hat{\mathbf{r}}_\mathbf{u} \odot \mathbf{e_u}|\mathbf{r_u})\Big), \qquad (2)\end{aligned}$$

where $\hat{\mathbf{r}}_\mathbf{u}$ is an n-dimensional rating vector generated by G corresponding to a user u and $\mathbf{r_u}$ is her real rating vector (i.e., ground truth). $\hat{\mathbf{r}}_\mathbf{u}$ is derived by $\hat{\mathbf{r}}_\mathbf{u} = G(\{\mathbf{z}, \mathbf{r_u}\})$ where {} indicates the concatenation of two vectors inside. From the viewpoint of *conditional GAN*[22], $\mathbf{r_u}$ corresponds to a user u's specific condition; our G generates output conditioned by $\mathbf{r_u}$. Likewise, conditioned by $\mathbf{r_u}$, D outputs a single scalar value representing the probability that its input came from the real rating data, rather than from G. Both G and D are represented by neural networks. $\mathbf{e_u}$ is an n-dimensional indicator vector specifying whether user u has given a rating to item i (e_{u_i}=1) or not (e_{u_i}=0), and $\odot$ stands for element-wise multiplication. $\mathbf{e_u}$ is used to mask G's output $\hat{\mathbf{r}}_\mathbf{u}$ by multiplying it with $\mathbf{e_u}$, and thus the model parameters are learned based solely on the observed ratings and will be used later to generate potential (yet unobserved) user-item ratings.

We employ the *stochastic gradient descent* (*and ascent*) with mini-batch and back-propagation to train our G (and D). We update the model parameters of G and D alternately, keeping one fixed when

Table 1: Accuracy with/without RAGAN (P@5)

	ML100K	Ciao	Watcha	ML1M
AutoRec	.0696	.0191	.0272	.0727
RAGAN+AutoRec	.0543	.0150	.0210	.0805
SVD	.0976	.0128	.0384	.0419
RAGAN+SVD	.1016	.0125	.0321	.0444
ItemKNN	.0326	.0079	.0071	.0362
RAGAN+ItemKNN	.0339	.0079	.0049	.0297

the other is being trained. After training of GAN is completed, we feed $\mathbf{z}$[1] and $\mathbf{r_u}$ for G to generate a dense rating vector $\hat{\mathbf{r}}_\mathbf{u}$, which contains predicted ratings on all the items for user u. Among all the predicted ratings, we collect element j in $\hat{\mathbf{r}}_\mathbf{u}$ where $j \in I \setminus I_u$, i.e., the predicted ratings only for the unrated items, and augment the original rating matrix $\mathbb{R}$ with those ratings.

4 RAGAN^{BT}

4.1 Motivation

We first point out a unique challenge of treating biases in RAGAN. We note that the ratings in our datasets are skewed towards high ratings (i.e., 3, 4, or 5 that implies *positive preferences*), coinciding with the results in other studies [6, 12]. This is mainly because people tend to choose (thus, to give ratings to) only the items that they are likely to love, and never to give ratings to those items that do not attract their interest. We claim that this *selection bias* is the main reason to make GAN trained to generate the values for unobserved entries *biased towards high ratings*. As a result, the CF models learning the rating matrix augmented with such biased ratings would fail to capture users' true preferences.

In order to make the above challenge clearer, we conducted preliminary experiments with four real-world datasets. We compare two *precisions* of top-5 recommendations performed by CF methods: one obtained by learning from the original rating matrix and the other from the rating matrix augmented with the synthetic ratings generated by RAGAN. As CF methods, we employ ItemKNN [27], SVD [16] and AutoRec [28], which are very popular ones among memory-based methods, MF-based models, and DNN-based models, respectively.

As shown in Table 1, the effect of the rating augmentation does not seem that positive: RAGAN did not help improve the accuracy in all cases and even worsened it in some cases. We analyzed the results and found the reason as shown Figure 1. First, in the original rating matrix, we observe severe *imbalance* between low (i.e., 1 or 2) and high (i.e., 3, 4, or 5) ratings from all the datasets: only a small fraction (i.e., 10–17%) of ratings are 1 and 2 which indicate negative preferences, and the rest are 3, 4, and 5 which indicate positive preferences. Since RAGAN is trained with the data biased to those high ratings, it generates mostly high ratings as in Figure 1. We note, however, this result is quite far from the reality: the majority of unobserved entries would imply users' negative preferences since users would not give a rating to those items that they are not interested in [6, 13]. Consequently, the CF methods learning the augmented matrix having unrealistic ratings could not provide accuracy improvement as shown in Table 1.

[1]In practice, however, we did not use the random noise vector $\mathbf{z}$ as done in [3].

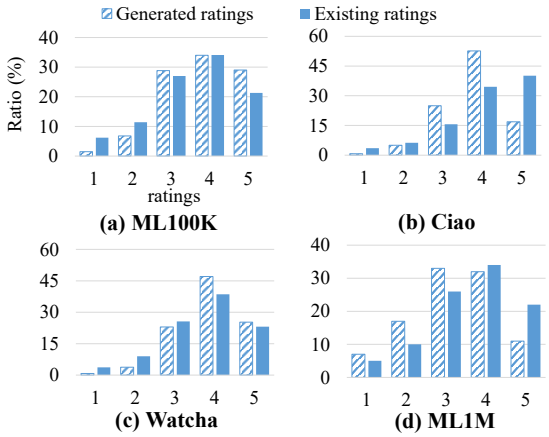


Figure 1: Rating distributions.

4.2 Overview of RAGAN^{BT}

A challenge then arises: how can we prevent GAN from being biased towards high ratings, thus generating realistic ratings for accurate CF? As a solution to this challenge, we now propose RAGAN^{BT}: a refined RAGAN with *bias treatment*. Figure 2 shows an overview of our RAGAN^{BT}, composed of the following three steps: first, we identify the *negative items* for users in a rating matrix (Figure 2(a)); we add low ratings to the negative items and use the newly added ratings along with the original ratings to train GAN (Figure 2(b)); finally, G in the trained GAN generates future ratings, which are to be augmented in the rating matrix (Figure 2(c)). We will elaborate the details of each step in the following sections.

4.3 Finding Negative Items

If user u's rating on item i is missing in a rating matrix, it will be due to one of the following reasons [11, 21, 29]:

- **lack of awareness**: u might prefer i, but was not aware of the existence of i. So, u could not rate i.
- **lack of interest**: u was aware of the existence of i, but was not interested in i because u has a low preference on i. So, u did not buy i, thus not rating i.

Obviously, a user's negative items would be her items unrated due to the latter reason. Along this line, our goal is to capture a user's interest on her unrated items and to select the items on which she has low interest. To this end, we employ the *one-class collaborative filtering* (OCCF) framework [23] following several recent work [13, 17, 19, 20]: The *implicit feedback* of a user does not always indicate her eventual preference on items represented by ratings, but naturally implies her interest on items *before using them* [10, 13, 23]; therefore, OCCF can be seen as a task of analyzing a user's interest on her unrated items and then finding top-N items that are likely to be of interest to her. Note that we have the goal *opposite* to that of OCCF: while the goal in OCCF is to find the items *most interesting* to a user, our goal is to find the items *least interesting* to her.

There have been numerous OCCF methods proposed; for example, BPR [26], FISM [14], and CDAE [34]. Among them, we employ Collaborative Denoising Auto-Encoder (CDAE), which is a DNN-based latent factor model and has shown a state-of-the-art accuracy

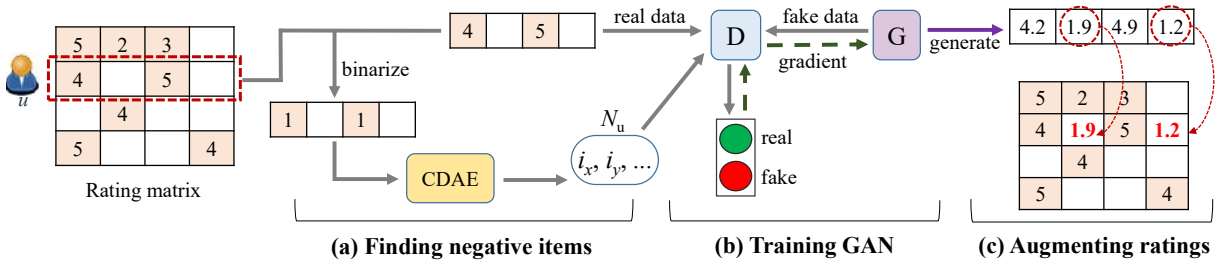


Figure 2: Overview of our proposed RAGAN^{BT}.

under OCCF setting. CDAE's input is a sparse implicit feedback vector of a user u, $\mathbf{r}_\mathbf{u}^\mathbf{b}$ (i.e., u's binarized ratings). This vector is first corrupted by masking noise: formally, the corrupted input vector, $\tilde{\mathbf{r}}_\mathbf{u}^\mathbf{b}$, is drawn by making some fraction (chosen at random for each user) of observed entries of $\mathbf{r}_\mathbf{u}^\mathbf{b}$ 0. Then, fed with $\tilde{\mathbf{r}}_\mathbf{u}^\mathbf{b}$, CDAE maps it to the hidden layer as $\mathbf{h} = \sigma(W_1^T \tilde{\mathbf{r}}_\mathbf{u}^\mathbf{b} + \mathbf{V_u} + \mathbf{b_1})$ where $\mathbf{V_u}$ is user u's latent feature vector. Finally, it tries to reconstruct the initial input (i.e., not corrupted one) by $\hat{\mathbf{r}}_\mathbf{u}^\mathbf{b} = \sigma(W_2\mathbf{h} + \mathbf{b_2})$. The model parameters of CDAE (i.e., W_1, b_1, W_2, b_2, and V) are learned by optimizing the objective function in the following:

$$\arg\min_{\psi} \frac{1}{m} \sum_{u=1}^{m} \mathbb{E}_{p(\tilde{\mathbf{r}}_\mathbf{u}^\mathbf{b} | \mathbf{r}_\mathbf{u}^\mathbf{b})} [\ell(\mathbf{r}_\mathbf{u}^\mathbf{b}, \hat{\mathbf{r}}_\mathbf{u}^\mathbf{b})] + \Omega(\psi), \tag{3}$$

where $\ell(\cdot)$ is a loss function and $\Omega(\cdot)$ is a term for so-called L2 regularization. While $\ell(\cdot)$ can be either square loss or logistic loss, we choose square loss that provides better empirical results in our work. We use ψ to denote a collection of all edge weights (W_1 and W_2) and biases ($\mathbf{b_1}$ and $\mathbf{b_2}$) in the network.

After ψ is learned, CDAE is fed with $\mathbf{r}_\mathbf{u}^\mathbf{b}$, and then reconstructs a *dense vector* that contains predicted scores of interestingness on all *unrated* items for user u. We now define a user u's *negative items*, N_u, as the bottom S percent of unrated items whose predicted scores of interestingness are the lowest [13].

4.4 Training GAN with Negative Items

We now present how we leverage those negative items to GAN's learning process. Our goal is to prevent G from being biased towards high ratings. To this end, we add a *low rating* for the negative items to $\mathbf{r_u}$ which is fed to D, as a new ground truth vector that G tries to mimic. By doing so, G would try to generate not only high ratings on the observed entries in $\mathbf{r_u}$ but also low ratings on the identified negative items in order to deceive D. Thus, we expect that G will no longer be biased towards high ratings, being able to learn users' positive and negative preferences *together in a balanced way*.

Let δ denote a low rating to be added. There are three choices for δ assigned to $\mathbf{r_u}$: 0, 1, and 2. Assigning ratings of 0 (i.e., $\delta = 0$) poses strong implication that a user does not like the negative items at all, while assigning ratings of 2 (i.e., $\delta = 2$) indicates a less severe uninterestingness towards the negative items. For each user u given, we add ratings of δ for N_u to u's real rating vector $\mathbf{r_u}$ (i.e., $r_{uj} \leftarrow \delta$, $\forall j \in N_u$), and then the imputed $\mathbf{r_u}$ is fed with D as a ground truth vector that G tries to mimic.

4.5 Recommendation

Once the training of GAN is complete, we feed G with $\mathbf{z}$ and $\mathbf{c_u}$ to generate plausible ratings. Then, we fill all missing entries in $\mathbb{R}$ with those generated ratings. We denote this augmented rating matrix as $\mathbb{R}'$. Finally, any CF models such as ItemKNN, SVD, and AutoRec can be trained to predict the preferences on the unobserved entries in $\mathbb{R}$ in their own way by using $\mathbb{R}'$ as training data. Finally, for each user, we pick the top-N items having the highest predicted preferences for recommendation.

5 EVALUATION

5.1 Experimental Settings

For evaluation, we used four real-world datasets of Watcha[2], Ciao [30], Movielens 100K, and Movielens 1M[3] (shortly, ML100K and ML1M). Table 2 summarizes their detailed statistics. For each dataset, we randomly select 80% of ratings for training data and the rest 20% for testing. In addition, we again set aside 20% of the training data for validation in order to use it for tuning hyper-parameters, such as the number of hidden layers and hidden nodes in G and D. Among the ratings in the test and validation sets, we only consider the ratings of 5 as our ground truth [5, 18]. Also, we employed four popular accuracy metrics: *precision* (P@N), *recall* (R@N), *normalized discounted cumulative gain* (G@N), and *mean reciprocal rank* (M@N). We set N as 5 and 20[4].

Table 2: Statistics of datasets

Datasets	# users	# items	# ratings	Sparsity
Ciao	996	1,927	18,648	98.72%
Watcha	1,391	1,927	101,073	96.98%
ML100K	943	1,682	100,000	93.69%
ML1M	6,039	3,883	1,000,209	95.72%

5.2 Experimental Results and Analyses

5.2.1 Q1: Effectiveness of our idea of bias treatment. Our key contribution is to address a unique challenge arises when data augmentation by GAN is applied to CF, and to suggest RAGAN^{BT}, a solution to this challenge. To show the effectiveness of this solution in rating augmentation, we visualize the accuracies of ItemKNN, SVD, and AutoRec that learn from two different rating matrices: (1) $\mathbb{R}'$ augmented by RAGAN and (2) $\mathbb{R}'$ augmented by RAGAN^{BT}.

Figure 3 reports the learning curves provided by each method *A+B* where *A* indicates a rating augmentation method and *B* does a CF model. We observe that RAGAN^{BT} makes great improvements in accuracy for all three CF models: for instance, RAGAN^{BT}+AutoRec

[2]http://watcha.net
[3]https://grouplens.org/datasets/movielens/
[4]Throughout the paper, however, we report the result of P@5 due to space limitations. We observed that the results of the other metrics exhibited very similar tendency.

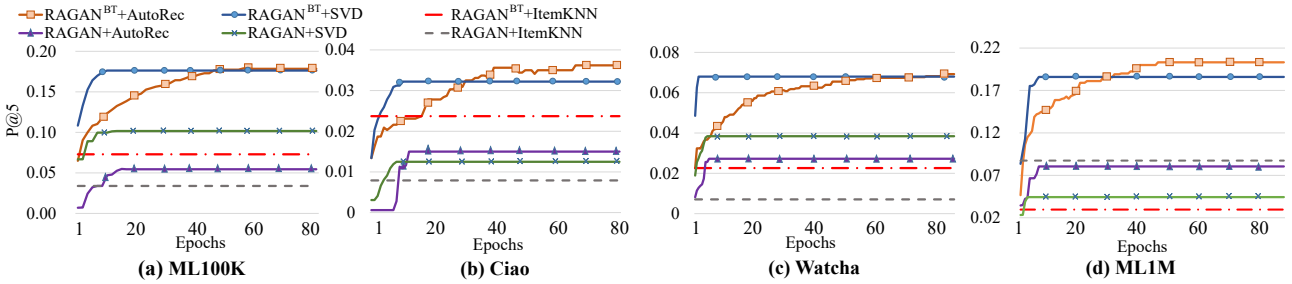


Figure 3: Learning curves (P@5).

provides up to 141.3% improvement, compared to RAGAN+AutoRec on Ciao and up to 228.5% on ML100K. This great improvement is mainly because RAGAN^{BT} generates plausible ratings by treating successfully the bias towards high ratings, thereby alleviating the data sparsity problem by augmenting those ratings.

Figure 4 reports the distributions of ratings generated by RAGAN^{BT}. Compared to the distribution of ratings generated by naive RAGAN, which is shown in Figure 1, most of generated ratings for unrated items are shown lower than or equal to 2. This result coincides with the observation that people tend to have low preferences on most of their unrated items [6, 13]. As a result, RAGAN^{BT} helps CF models to better understand users' true preferences, thereby improving the accuracy of top-N recommendation significantly.

5.2.2 Q2: Accuracy of RAGAN^{BT} according to varying hyper-parameters. This subsection investigates the impact of the following key hyper-parameters, which are unique in our RAGAN^{BT} framework: (1) the portion of negative items, S, for bias treatment and (2) which value to use as δ. Here, we employ SVD as our CF model. The results on Ciao and ML1M are shown only due to limited space.

Figure 5 reports the experimental results, where the three graphs in each figure represent the P@5 value over varying S when δ is given to negative items with 0, 1, and 2, respectively. We observe that the proposed framework works well when δ is 0 and 1. We also observe that when S reaches sufficiently large number (e.g., 25%), the accuracy of RAGAN^{BT} converges and does not change much after that point. These results show that setting S as a value moderately larger than 25% ensures CF models to understand users' positive and negative preferences in a balanced way, owing to our bias treatment. Thus, in the following experiments, we fix $\delta = 0$ and $S = 25$, and then compare it with the state-of-the-art methods.

5.2.3 Q3: Comparisons with state-of-the-arts. We now compare the accuracy of RAGAN^{BT} with those of the state-of-the-art top-N recommenders. Notably, we performed the comparisons not only under basic top-N recommendation, but also under the following two more-difficult settings: (1) long-tail items (i.e., items having a few ratings) recommendation and (2) recommendation to cold-start users (i.e., users giving a few ratings) [13]. To set up the first situation, we divided all items, by their popularity, into *top-head* items[5] that involve 30% of all ratings and *long-tail* items that are the rest. Then, we recommended only long-tail items to users. To create the second setting, we randomly selected 200 users as cold-start users. Then, we also randomly selected five ratings for each

[5] 140, 107, 101, and 202 items are included as the *top-head* items in Ciao, Watcha, ML100K, and ML1M, respectively.

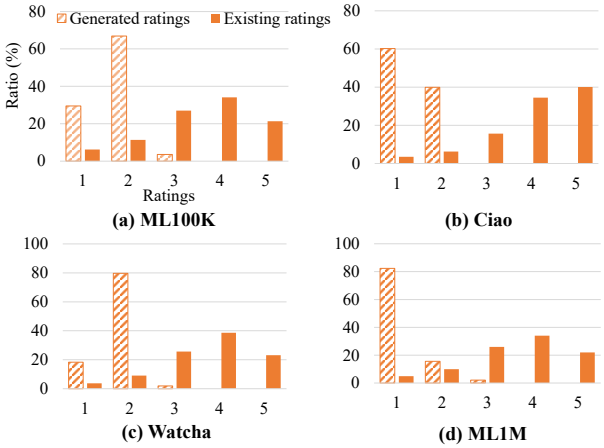


Figure 4: Rating distributions.

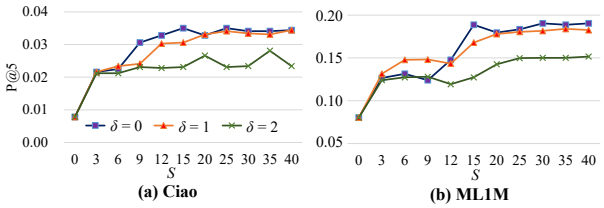


Figure 5: Impact of key hyper-parameters (P@5).

cold-start user and discarded the rest of her ratings. We finally recommended items to the cold-start users only.

Next, we introduce our competitors: (1) ItemPop, the simplest non-personalized method that ranks items based on the descending order of popularity; (2) ItemKNN, a popular neighborhood based method [27]; (3) SVD, a standard MF-based model along with user and item bias terms [16]; (4) SVD++, a combined idea of an MF and a neighborhood based method by approximating similarities among items via MF model [15]; (5) AutoRec, an autoencoder based, non-linear latent factor model [28]. Furthermore, we include two models that employ the *zero rating imputation*: PureSVD [5] and Zero-Injection [13]. Notably, Zero-Injection is the most similar to ours: both Zero-Injection and ours find a set of negative items, but after then Zero-Injection runs SVD to directly produce recommendations while our RAGAN^{BT} runs GAN to generate ratings to be imputed. PureSVD is also similar to ours: it regards *all unrated items* as negative ones and imputes zeros to them before running SVD.

Table 3: Accuracy comparisons with the state-of-the-art CF methods (P@5)

Datasets	Ciao			ML100K			Watcha			ML1M		
Situations	Basic	Long-tail	Cold-start	Basic	Long-tail	Cold-start	Basic	Long-tail	Cold-start	Basic	Long-tail	Cold-start
ItemPop	.0209	.0050	.0185	.0685	.0210	.0390	.0256	.0088	.0244	.1071	.0322	.0498
ItemKNN	.0079	.0075	.0064	.0326	.0262	.0159	.0087	.0071	.0099	.0362	.0199	.0284
SVD	.0128	.0072	.0164	.0976	.0312	.0431	.0384	.0204	.0221	.0419	.0155	.0390
SVD++	.0197	.0063	.0174	.0910	.0297	.0455	.0324	.0207	.0240	.0755	.0256	.0520
PureSVD	.0278	.0131	.0185	.1466	.0753	.0572	.0573	.0377	.0290	.1483	.0923	.0620
AutoRec	.0191	.0078	.0174	.0696	.0324	.0440	.0272	.0212	.0242	.0727	.0231	.0511
Zero-Injection	.0347	.0197	.0256	.1429	.0617	.0601	.0672	.0389	.0255	.1856	.0926	.0600
RAGAN^{BT}+ItemKNN	.0237	.0206	.0231	.0729	.0637	.0573	.0369	.0288	.0346	.1035	.0346	.0610
RAGAN^{BT}+SVD	**.0366**	.0197	.0256	.1762	.0880	.0660	.0680	.0423	**.0350**	.1860	**.0997**	**.0780**
RAGAN^{BT}+AutoRec	.0362	**.0267**	**.0277**	**.1784**	**.0884**	**.0670**	**.0701**	**.0426**	.0330	**.2032**	.0957	.0680

Table 3 reports the comparison results. Overall, we see that AutoRec and SVD along with our RAGAN^{BT} significantly outperform all the other competitors. In particular, we beat PureSVD and Zero-Injection, which have shown the best accuracy among all CF models, by a wide margin: the average of relative improvements of our RAGAN^{BT}+AutoRec over Zero-Injection is 15.8%. We note that PureSVD and Zero-Injection can be considered as simplified versions of our work: they assign 0 ratings to all the unrated items (PureSVD) or to carefully chosen negative items (Zero-Injection), and then run SVD to produce recommendation without GAN for further imputation. However, their important drawback is that they do not take users' *personalized preferences* on items into account, just adding the *same zero* ratings to all (PurSVD) and some (Zero-Injection) unrated items. In contrast, we exploit GAN so as to generate more realistic and personalized ratings carefully. Our experimental results indeed verify the effectiveness of our RAGAN^{BT} over the two competitors: it successfully relieves the data sparsity problem by carefully generating plausible ratings, rather than simply imputing zero ratings as PureSVD and Zero-Injection did.

Another notable point is that RAGAN^{BT} is more successful under the long-tail and cold-start settings, which are more-difficult situations than basic top-N recommendation. More specifically, RAGAN^{BT}+AutoRec performs well under the long-tail setting on Ciao (35.5% improved, compared to the best performer among our competitors) and ML100K (17.3% improved), and RAGAN^{BT}+SVD performs well under the cold-start setting on Watcha (20.7% improved) and ML1M (25.8% improved). It can be a very promising result since those difficult situations are very common in most real-world recommender systems. Therefore, our RAGAN^{BT} could be a good choice for those situations suffering from lots of cold-start users and long-tail items.

5.2.4 Q4: Execution time analysis. Lastly, we analyzed our proposed RAGAN^{BT} in terms of the execution time. Specifically, we measured each time consumed by (1) finding negative items with CDAE, (2) augmenting ratings with GAN, and (3) recommending items with AutoRec, which has achieved higher accuracy than SVD and ItemKNN. Here, we used a single machine equipped with an i9 7700K Intel CPU, 64GB RAM, and NVIDIA TITAN XP GPU. The results are summarized in Table 4. For the largest dataset (i.e., ML1M), total time required is about 6,891 seconds: each epoch of CDAE takes around 8 seconds, and needs 150 epochs to sample appropriate negative items for each user; each epoch of GAN takes approximately 12.5 seconds, and needs 450 epochs to generate plausible ratings; finally, AutoRec needs 100 epochs to get satisfactory accuracy where each epoch takes about 1 second. Note that the total execution time consumed by SVD++, which is one of our baselines, is around 5,078 seconds. In summary, we see that our framework augments plausible ratings and provides high-quality recommendations within a reasonable time. This is because, even though our RAGAN^{BT} involves three procedures of model training, all of them are based on neural networks whose training can be easily parallelized by fast GPU.

Table 4: Execution time (s)

	Total	Finding negative items (**CDAE**)	Augmenting ratings (**GAN**)	Recommending items (**AutoRec**)
Ciao	1,062.76	123.49	927.11	6.08
Watcha	2,060.70	222.69	1,827.21	10.80
ML100K	2,500.42	382.08	2,102.03	16.31
ML1M	6,891.66	1,164.88	5,623.58	103.20

6 CONCLUSIONS

In this paper, we have presented RAGAN, a novel rating augmentation framework based on GAN, to relieve the data sparsity problem in CF. Then, we have identified a challenge of treating the biases to high ratings that GAN faces when learning from the rating data. We have proposed RAGAN^{BT}, which successfully addresses the challenge by involving negative items when training GAN. From our comprehensive experiments, we have demonstrated that RAGAN^{BT} significantly improves the accuracies of existing CF methods, making them outperform the existing state-of-the-art CF methods, not only under basic top-N recommendations but also under more-difficult situations of long-tail item recommendations and recommendations to cold-start users.

ACKNOWLEDGMENTS

This work was supported by (1) the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT: Ministry of Science and ICT) (No. NRF-2017R1A2B3004581) and (2) Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (No. NRF-2017M3C4A7083678). Also, we thank the Naver Corporation for their support including computing environment and data, which helped us greatly in performing this research successfully.

REFERENCES

[1] Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2017. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340* (2017).

[2] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. 2017. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision and pattern recognition*, Vol. 1. 7.

[3] Dong-Kyu Chae, Jin-Soo Kang, Sang-Wook Kim, and Jung-Tae Lee. 2018. CFGAN: A generic collaborative filtering framework based on generative adversarial networks. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 137–146.

[4] Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F Stewart, and Jimeng Sun. 2017. Generating multi-label discrete electronic health records using generative adversarial networks. *arXiv preprint arXiv:1703.06490* (2017).

[5] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the 4th ACM conference on recommender systems*. 39–46.

[6] Nilesh N Dalvi, Ravi Kumar, and Bo Pang. 2013. Para'normal'activity: on the distribution of average ratings.. In *Proceedings of the 7th international AAAI conference on weblogs and social media*. 110–119.

[7] Chris Donahue, Julian McAuley, and Miller Puckette. 2018. Synthesizing audio with generative adversarial networks. *arXiv preprint arXiv:1802.04208* (2018).

[8] Maayan Frid-Adar, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. 2018. GAN-based synthetic medical image augmentation for increased CNN performance in Liver Lesion classification. *arXiv preprint arXiv:1803.01229* (2018).

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[10] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on World Wide Web*. 173–182.

[11] José Miguel Hernández-Lobato, Neil Houlsby, and Zoubin Ghahramani. 2014. Probabilistic matrix factorization with non-random missing data. In *Proceedings of the 31st international conference on machine learning*. 1512–1520.

[12] Nan Hu, Jie Zhang, and Paul A Pavlou. 2009. Overcoming the J-shaped distribution of product reviews. *Commun. ACM* 52, 10 (2009), 144–147.

[13] Won-Seok Hwang, Juan Parc, Sang-Wook Kim, Jongwuk Lee, and Dongwon Lee. 2016. "Told you i didn't like it": Exploiting uninteresting items for effective collaborative filtering. In *Proceedings of the IEEE 32nd international conference on data engineering*. 349–360.

[14] Santosh Kabbur, Xia Ning, and George Karypis. 2013. Fism: Factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining*. 659–667.

[15] Yehuda Koren. 2008. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining*. 426–434.

[16] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).

[17] Jongwuk Lee, Won-Seok Hwang, Juan Parc, Youngnam Lee, Sang-Wook Kim, and Dongwon Lee. 2019. l-Injection: Toward effective collaborative filtering using uninteresting items. *IEEE transactions on knowledge and data engineering* 31, 1 (2019), 3–16.

[18] Sang-Chul Lee, Sang-Wook Kim, Sunju Park, and Dong-Kyu Chae. 2018. An approach to effective recommendation considering user preference and diversity simultaneously. *IEICE transactions on information and systems* 101, 1 (2018), 244–248.

[19] Youngnam Lee, Sang-Wook Kim, Sunju Park, and Xing Xie. 2018. How to impute missing ratings?: Claims, solution, and its application to collaborative filtering. In *Proceedings of the 27th international conference on World Wide Web*. 783–792.

[20] Yeon-Chang Lee, Sang-Wook Kim, and Dongwon Lee. 2018. gOCCF: Graph-theoretic one-class collaborative filtering based on uninteresting items. In *Proceedings of the 2018 AAAI international conference on artificial intelligence*.

[21] Dawen Liang, Laurent Charlin, James McInerney, and David M Blei. 2016. Modeling user exposure in recommendation. In *Proceedings of the 25th international conference on World Wide Web*. 951–961.

[22] Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784* (2014).

[23] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Proceedings of the 2008 IEEE 8th international conference on data mining*. 502–511.

[24] Chanyoung Park, Donghyun Kim, Jinoh Oh, and Hwanjo Yu. 2016. Improving top-k recommendation with truster and trustee relationship in user trust network. *Information Sciences* 374 (2016), 100–114.

[25] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[26] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th international conference on uncertainty in artificial intelligence*. 452–461.

[27] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[28] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th international conference on World Wide Web*. 111–112.

[29] Harald Steck. 2010. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th international conference on knowledge discovery and data mining*. 713–722.

[30] Jiliang Tang, Huiji Gao, and Huan Liu. 2012. mTrust: Discerning multi-faceted trust in a connected world. In *Proceedings of the 5th ACM international conference on Web search and data mining*. 93–102.

[31] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2017. GraphGAN: Graph Representation Learning with Generative Adversarial Nets. *arXiv preprint arXiv:1711.08267* (2017).

[32] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 1235–1244.

[33] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*. 515–524.

[34] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the 9th ACM international conference on Web search and data mining*. 153–162.

[35] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence generative adversarial nets with policy gradient.. In *Proceedings of the 2017 AAAI international conference on artificial intelligence*. 2852–2858.

On the Impact of Choice Architectures on Inequality in Online Donation Platforms

Abhijnan Chakraborty*
MPI for Software Systems

Nuno Mota*†
EPFL

Asia J. Biega
MPI for Informatics

Krishna P. Gummadi
MPI for Software Systems

Hoda Heidari
ETH Zürich

ABSTRACT

Online donation platforms, such as DonorsChoose, GlobalGiving, or CrowdFunder, enable donors to financially support entities in need. In a typical scenario, after a fundraiser submits a request specifying her need, donors contribute financially to help raise the target amount within a pre-specified timeframe. While the goal of such platforms is to counterbalance societal inequalities, biased donation trends might exacerbate the unfair distribution of resources to those in need. Prior research has looked at the impact of biased data, models, or human behavior on inequality in different socio-technical systems, while largely ignoring the *choice architecture*, in which the funding decisions are made.

In this paper, we focus on (i) quantifying inequality in the project funding in online donation platforms, and (ii) understanding the impact of platform design on donors' behavior in magnifying those inequalities. To this end, we borrow decomposable measures of income inequality from economics, and apply it to identify candidate factors contributing to inequality on the DonorsChoose website. Analyzing longitudinal changes in the website design, we show how the platform design impacts the relative contribution of the different factors. Our work motivates the need for a careful investigation of the impact of choice architectures on user decisions, in donation platforms in particular, and in online platforms more generally.

CCS CONCEPTS

• **Human-centered computing → Empirical studies in HCI**.

ACM Reference Format:
Abhijnan Chakraborty, Nuno Mota, Asia J. Biega, Krishna P. Gummadi, and Hoda Heidari. 2019. On the Impact of Choice Architectures on Inequality in Online Donation Platforms. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313663

*Both authors contributed equally to this work.
†This work was done while the author was interning at MPI for Software Systems.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313663

1 INTRODUCTION

Online donation platforms, such as DonorsChoose, Chuffed, GlobalGiving, CrowdFunder or Facebook fundraising, provide effective platforms for connecting donors to entities in need of financial support. In a typical scenario, after a recipient submits a request specifying her need, donors can contribute financially to help raise the target amount within a pre-specified timeframe. The core mission of such platforms is to counter existing social inequalities. Biased donor behavior, however, may skew the distribution of donations towards or against certain recipients, exacerbating some of these inequalities as a result. Consider DonorsChoose.org, a platform where US-based school teachers can create funding requests to support the educational needs of their classrooms. Since public schools in the US are primarily supported by local taxes [24], schools located in poor neighborhoods are often severely underfunded. If the donors have a preference to donate to schools in their own region, the funding inequality combined with biased donation behavior could lead to a *rich get richer* effect, where wealthy schools easily receive additional funding in the form of online donations, while schools in poor neighborhoods remain impoverished [25].

At first glance, such biased outcomes may seem to be the sole artifact of donors' preferences rather than that of the platform mediating donations. After all, donors are rational beings, and the donation platforms offer them full freedom of choice. However, drawing on a rich body of work in behavioral economics, we argue that the platform can indeed play an important role in shaping donors' behavior without limiting their choices: Donation websites present requests within a *choice architecture* [33], influencing donors' decision-making and *nudging* them towards choosing certain alternatives. Examples of nudges [32] on DonorsChoose include the *default* ranking strategy used in project listings, *decision making aids* such as search filters, or the *selection of project attributes* disclosed to the donors. The main conceptual contribution of our work is to draw attention to the power of nudges in socio-technical systems, while our empirical contributions focus particularly on online donation platforms.

Prior research works have looked at how different factors impact long-term retention of donors [1] or the project financial success [14, 16, 34]. The likelihood of funding has also been shown to be influenced by the language used in project descriptions [22], the gender of teachers [26], the social media activity associated with the project [20], or the amount of early donations [29]. However, our work highlights that the way a platform presents its users with choices will have significant impact on their donation decisions.

We, therefore, advocate for a careful investigation of choice architectures and the ethics of their impact on guiding user decisions. What we focus on in this paper more specifically, is how the choice architecture of the website might nudge donors towards choosing projects with certain attributes, and as a result, contribute to inequality in donations along the dimension of these attributes.

As our first contribution, we quantify the inequality in project funding and attribute portions of the inequality to different project characteristics (Note that in highly empirical applications, it is difficult to reason about causal effects without making strong assumptions about the data generating process, which limits the practicality of causal estimation approaches). Our approach to this task emerges from its similarity to the problem of quantifying income inequality, specifically to measure the contribution of characteristics such as gender or race to wage inequality. This problem has been studied in economics using decomposable inequality indices [4, 23], whose *additive decomposability property* helps factor out the influence of individual differences. We borrow one of such measures called 'Generalized Entropy' to find the contribution of different project characteristics to the overall inequality in funding.

Our second contribution revolves around *understanding how platform design choices affect donor behavior*. This idea is related to the existing literature on nudges [27, 32] and choice architectures [19, 28, 33] in behavioral economics. Research in these areas investigates how people's choices depend on how information is presented to them. This framing applies to our scenario: we are interested in learning whether the way a donation platform presents project information influences how donors choose projects with specific attributes. We face two challenges when attempting this goal.
First, once we pick a design element whose influence we want to measure, we have to make sure all other elements of the platform environment are fixed, including other website elements as well as donor characteristics. Second, while it is feasible to imagine running active experiments (i.e., A/B testing) by the platform, such an approach raises a lot of ethical concerns in a setting where people's decisions are nudged, and the outcomes influence future real-world financial success of others [3]. To overcome these issues, we analyze longitudinal changes in DonorsChoose platform design, and we estimate the average impact of a design change by comparing the observations right before and after it was first introduced. This way, we are able to quantify the influence of individual website design elements on project selection rates from historical donation data, while avoiding manipulation of future donations.

We run the inequality analysis on the data released by DonorsChoose in 2016, which provides over ten years worth of project and donation information, revealing two interesting findings. First, school location (zip code) is a factor largely contributing to funding inequality. Further analysis confirms that this is because (i) the majority of money is donated within-state (i.e., donors tend to give to their local projects), and (ii) the bulk of donations come from a handful of wealthier states, such as California or New York. Second, we find that the poverty level of the school does not contribute much to funding inequality – projects from high-poverty facilities have equal success rates as those from low-poverty ones, despite research showing that people consciously prefer charitable giving to be need-based [10].

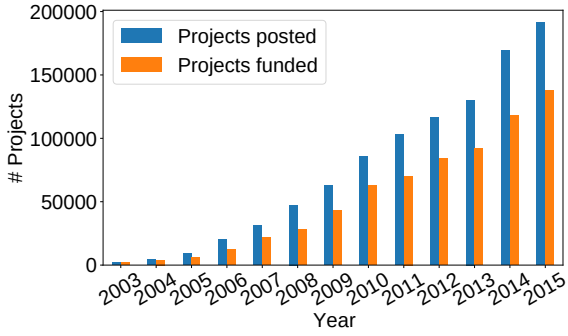


Figure 1: Number of projects posted per year (along with the number of projects successfully funded).

Our longitudinal analysis shows that it is possible to enhance the impact of school location or poverty level in influencing the project selection rates, via changes in the platform design. In the process, we highlight that even though donation platforms are perceived as neutral facilitators of interactions between the donors and the requestors, they do have an active role to play. We envision our work to pave the way for future studies aiming to employ nudges effectively to improve platform-specific goals, such as to maximize total amount of donations, or to ensure the social and ethical requirements of online altruism, such as bringing in fairness in the distribution of donations across projects.

2 DONORSCHOOSE.ORG DATASET

`DonorsChoose.org` is a popular donation platform where teachers from schools in the United States can request donations for their classroom projects. Such requests range from buying books or tools for specialized experiments, funding of excursions, all the way to replacing wore-down basic classroom equipment. Teachers do not receive money directly; DonorsChoose purchases all the requested resources and ships them directly to the schools [13]. Teachers on average request $561.57, with the median request being $413.67[1].

When choosing the projects, donors can select projects based on attributes such as the school location (zip code, city, state), main focus subject of the project (e.g., Math and Science, Health and Sports, Literacy and Language), or requested resources (e.g., Books, Computers and Tablets, Food, Clothing and Hygiene etc.). However, if a project does not raise its target amount by a certain deadline, it receives no funding, and the donors need to reallocate the donated money to other projects. While implementing this 'all-or-nothing' rule has been shown to have certain positive effects on donations, like increasing the overall amount of donations [35], there are projects that fail to receive funding as a result.

To identify the magnitude of such unsuccessful projects, in this work, we analyze the dataset publicly released by DonorsChoose in 2016 [12], which contains information about all projects posted on the platform starting from 2002 until the dataset's release date. During this period, DonorsChoose helped raise over $500M from over 5M donors. There were over 1M projects created by around 500K teachers from over 70K different schools. The dataset contains many project attributes, such as details about the teacher, requested

[1]The median amount is not round because of factors such as sales tax.

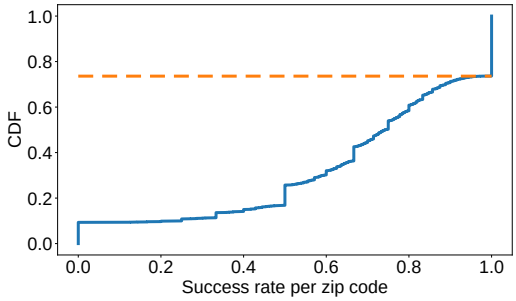


Figure 2: CDF of project success rates across different zip codes. The distribution is highly unequal. Around 26% zip codes have 100% success rate (shown as the dashed line), whereas the success rate is much lower in other zip codes.

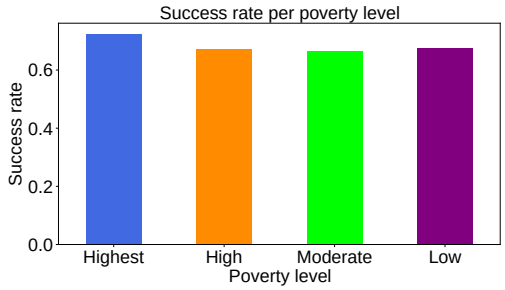


Figure 3: Distribution of project success rates across 4 poverty levels. The distribution is relatively equal.

resources, and the textual description of the request. The dataset also contains donation data, including donation date, amount donated, and location of the donor (state and city; while the zip codes of donors are anonymized for privacy reasons). Since the dataset covers only a part of 2002 and 2016, we remove these two years from our analysis and focus on projects posted from 2003 to 2015.

Figure 1 shows that the number of projects on the platform has been steadily growing, yet a significant number of projects fail to get funded each year. The data reveals that on average 29.79% of projects fail every year. These failure rates are unlikely to be the result of some projects being low-quality or fraudulent since all the projects posted on the website are vetted and verified by DonorsChoose. The platform also decided to fund all active projects after getting Ripple's generous $29M donation in early 2018 [6]. We can thus conclude that all projects on DonorsChoose are worthy of getting funded, and it is important to understand the factors contributing to the inequality of success rates across projects, to ensure that the failure rates do not adversely affect certain types of projects or schools. This problem is the focus of the next section.

3 MEASURING INEQUALITY IN DONATIONS

In this section, we propose a methodology based on the well-studied concept of inequality measurement in economics, to identify whether a given feature (or attribute) contributes to inequality in projects' chances of success. The proposed method allows us to quantify the inequality contributions *without the need for making any assumptions about the data generating process.*

For a fixed choice architecture, let features $X_1, \cdots, X_k$ characterize a project, and Y denote its success (with $Y = 1$ meaning that the project is successfully funded). We have access to observational data of the form $D = \{\langle (x_1^i, \cdots, x_K^i), y_i \rangle\}_{i=1}^N$ sampled i.i.d. from a joint distribution F. Then, we can utilize D to decide whether feature X_k contributes to the inequality across Y values. Note that we use upper case letters to denote random variables and lower case letters to denote in-sample realizations.

3.1 Decomposable Measures of Inequality

We propose to measure inequality using an *additively decomposable index of inequality* (namely, *generalized entropy*) for the vector $(y_1, \cdots, y_N)$. We then estimate the impact of a feature X_k by decomposing the total inequality according to X_k feature values.

We begin with an overview of inequality measurement. Measures of inequality quantify the degree of inequality in a distribution of resources or benefits over a population. More precisely, given a distribution/vector $\mathbf{y} = (y_1, \cdots, y_N) \in \mathbb{R}^n_{\geq 0}$, an inequality measure, $I : \bigcup_{n=1}^{\infty} \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$, maps any distribution/vector $\mathbf{y}$ to a non-negative real number $I(\mathbf{y})$. A distribution $\mathbf{y}$ is considered more equal than $\mathbf{y}'$ if and only if $I(\mathbf{y}) < I(\mathbf{y}')$. A long established line of work in economics offers an axiomatic characterization for inequality indices. Virtually all commonly-used inequality indices satisfy axioms such as Anonymity, Population invariance, Transfer principle, Zero-Normalization [30].

Additive decomposability is a structural property of some inequality measures requiring that for *any* partition G of the population into groups, the measure, $I(\mathbf{y})$, can be expressed as the sum of a *"within-group component"* $I^G_\omega(\mathbf{y})$ (a weighted sum of subgroup inequality levels) and a *"between-group component"* $I^G_\beta(\mathbf{y})$ (computed by assigning to each element in a subgroup $g \in G$ the subgroup's mean benefit μ_g): [2]

$$I(\mathbf{y}) = I_\omega(\mathbf{y}) + I_\beta(\mathbf{y}).$$

Following [30], we focus on a family of inequality indices called *generalized entropy indices* that satisfy additive decomposability. For a constant $\alpha \neq 0, 1$, the generalized entropy is defined as follows:

$$\mathcal{E}^\alpha(y_1, y_2, \cdots, y_n) = \frac{1}{n\alpha(\alpha-1)} \sum_{i=1}^{n} \left[\left(\frac{y_i}{\mu} \right)^\alpha - 1 \right]. \quad (1)$$

One can interpret generalized entropy as a measure of information theoretic *redundancy* in data. Given a partition G of the population, generalized entropy can be additively decomposed: Suppose subgroup $g \in G$ consists of n_g individuals with the benefit vector $\mathbf{y}^g = (y_1^g, \cdots, y_{n_g}^g)$ and mean benefit μ_g. Generalized Entropy can be decomposed as follows:

$$\mathcal{E}^\alpha(y_1, y_2, \cdots, y_n) = \sum_{g=1}^{|G|} \frac{n_g}{n} \left(\frac{\mu_g}{\mu} \right)^\alpha \mathcal{E}^\alpha(\mathbf{y}^g) + \sum_{g=1}^{|G|} \frac{n_g}{n\alpha(\alpha-1)} \left[\left(\frac{\mu_g}{\mu} \right)^\alpha - 1 \right]$$

[2] When the partition G in reference to is clear from the context, we drop the superscript G to simplify notation.

Feature	School Location	Amount Requested	Matching Donations	Resource Type	School Type	Primary Focus Subj.	Secondary Focus Subject	Secondary Focus Area	Primary Focus Area	Teacher Prefix	Poverty Level	Teacher Cred.	Grade Level
Normalized Inequality Contribution	0.1372	0.07198	0.02048	0.01399	0.0034	0.00307	0.00291	0.00208	0.0016	0.00108	0.00093	0.00087	0.00058

Table 1: The contribution of different features to inequality in donations. School location has the highest contribution, whereas poverty level has low contribution.

3.2 Measuring Feature Contribution through Inequality Decomposition

We propose estimating the impact of a feature X_k on Y through the following procedure:

(1) Partition the population (i.e., the set of all projects) into groups G, according to the value of the feature X_k. For instance, suppose X_k represents a teacher's gender, taking two different values. G partitions the projects into two groups, those with male teachers and those with female teachers.
(2) Compute and return $I_\beta^G(\mathbf{y})/I(\mathbf{y})$ as the impact of X_k on Y.

Note that just because there is a higher between-group component in decomposition according to feature X_k when compared to the decomposition according to X_j does not necessarily mean that feature X_k has more *causal* impact on Y when compared to X_j. Why are we focusing on inequality decomposition then?

3.3 Relationship with Causal Inference

Traditionally, in order to understand the impact of a set of explanatory variables (in our case, $X_1, \cdots, X_K$) on a target variable of interest (in our case, Y), econometricians begin by specifying an "explanatory model" (e.g., a simple regression model). Such a model carefully specifies a counterfactual in order to examine the influence of each supposedly causal factor on the target. In the process of developing such a model, econometricians must make several important choices, including which explanatory variables belong in the model, what functional form should the regression function take (i.e. logarithms, quadratic, etc.), and what the error distribution looks like. These choices are usually guided by domain expertise. For instance, consider a simple regression (which is one of the most commonly used methods in empirical studies): We assume $Y = U + \beta_0 + \sum_{k=1}^{K} \beta_k X_k$ where U is the error/noise term. Provided that this functional form is correctly specified, and that standard assumptions such as exogenous covariates hold, we can use OLS methods to estimate the coefficients. These estimates then serve as the measure of the causal impact of each attribute on the target variable: the parameter β_k measures the increase in Y attributable to one unit of change in X_k, keeping all the other covariates fixed.

Unfortunately, the above approach is restrictive in many empirical applications. First and foremost, this is because the modeling step can be cumbersome, and conclusions made through explanatory models are often sensitive to model specification [9]. Given these issues, here we base our analysis on an alternative tool, namely, measurement of inequality. [9] have indeed established a strong connection between the two methods under certain conditions. Assuming that the model is correctly specified (e.g., the data is indeed generated through the assumed linear model), they show that the impacts estimated by OLS and those derived by inequality decomposition are closely related. For the precise expressions, see equation (22) in [9]. In summary, our decomposition approach allows us to measure feature influence without the need for strong assumptions on the data generating process, and is better suited for discrete variables.

4 IMPACT OF CHOICE ARCHITECTURE ON DONATION INEQUALITY

Using the proposed methodology, in this section, we utilize observational data from DonorsChoose to identify the features contributing to inequality in projects' chances of success.

4.1 What's Behind Inequality in Donations?

Table 1 illustrates our proposal on the DonorsChoose data corresponding to the year 2015[3]. Note that the contributions of school location (in terms of zip codes), amount requested, resource type (the type of resource that was requested), and matching donations (whether the amount donated by individual donors to a project are going to be matched by larger corporate donation) are deemed high, whereas the contribution of poverty level is low. Throughout the rest of this section, we narrow down our attention to two of these features: school location and poverty level.

Figure 2 shows the CDF of project success rates across different zip codes. As evident from the figure, zip code has a high contribution to inequality. For around 26% of the zip codes, all the projects were funded, whereas for other zip codes, the percentage of funded projects varied anywhere between 50% to 90%. Figure 3 shows the distribution of project success across different poverty levels (schools in DonorsChoose are classified into four poverty categories: Highest, High, Moderate, and Low, depending on how many students come from low-income families). As evident from the figure, the project success rates are similar across different poverty levels, explaining the low value of inequality contribution for this feature.

4.2 Behavioral Mechanisms Underlying the Inequality Contributions

We suspect that the high inequality contribution of school location is in part due to the donors' preference to donate locally, that is, to donate to schools in their own zip code, city or state. The dataset does not contain donors' zipcode (due to privacy considerations), but our analysis at the city and state level indeed reveals a strong

[3] As mentioned earlier, the DonorsChoose dataset consists of project funding information of multiple years. To exclude external factors such as the amount of traffic to the website, large fundraising activities in specific years, or varied amounts of matching donations from our analysis, we focus on the projects posted on a particular year.

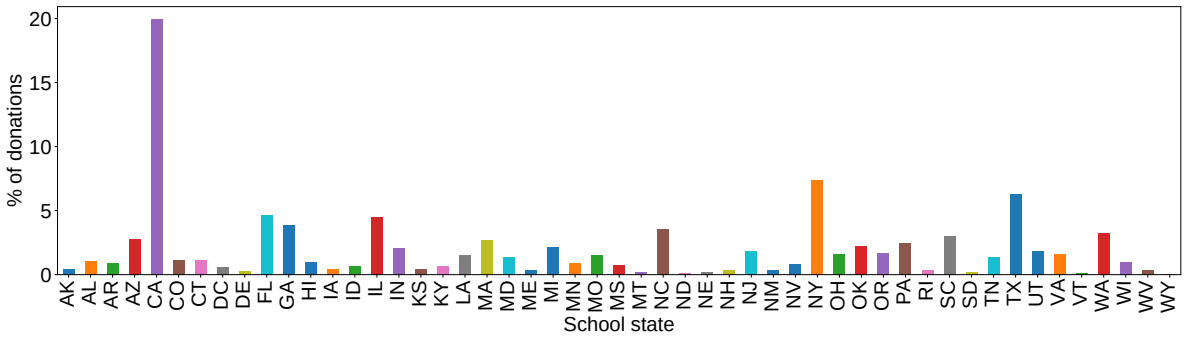


Figure 4: Percentage of donations going to different school states. There is huge disparity in the donation distribution.

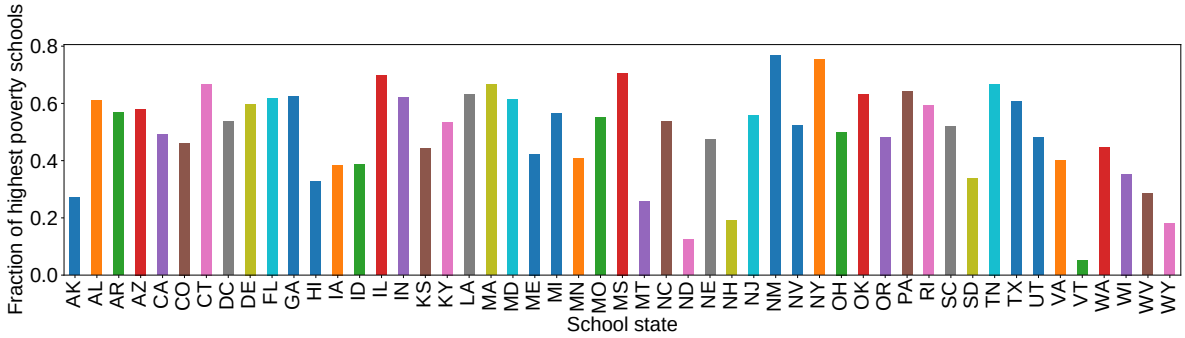


Figure 5: Fraction of schools belonging to highest poverty level in different states. Donation distribution in Figure 4 does not correlate with the school poverty level distribution.

tendency of donors to fund local projects: almost 80% of the total amount of funding is donated within the same state, and around 20% of the total amount is donated within the same city as the donor's. Only 20% of the total donations are made to schools outside donors' states. This tendency, combined with the fact that the amount of total donation varies widely across states (see Figure 4), explains the high inequality contribution of the school location. As we will discuss in Section 4.3, the locality bias might be further encouraged by the presence of certain nudges on the website (e.g., the option to search projects by location). This is despite the fact that the donation distribution in presence of the location bias does not correlate with the distribution of highest poverty schools in different states (see Figure 5), which are in higher need of economic support.

From a moral standpoint and to alleviate current social injustice, one may argue that the schools with higher economic need are more deserving of receiving donations, and the website should nudge the donations in their favor. To put it more formally, it may be desirable (ethically and socially) to reduce the contribution of location, but increase the contribution of school's poverty level to how donations are distributed. Next, we show evidence that choice architecture and nudges indeed impact contribution of these factors.

4.3 Evidence on the Impact of Nudges

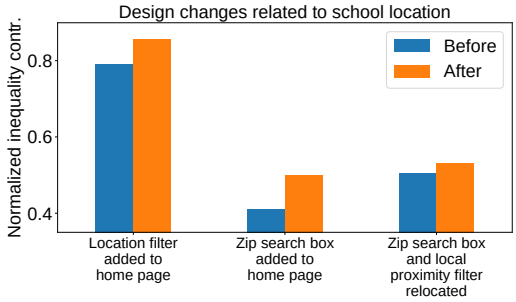
To gain insights on the impact of choice architecture and nudges on inequality in different projects' chances of success, we identify the time points, throughout the years, when a new website design was introduced in DonorsChoose, using snapshots from 'The Internet Archive' (`web.archive.org`). Then, utilizing the longitudinal donation data, we analyze the effect of these interventions.

To this end, prior works have proposed *Regression Discontinuity Designs* (RDDs) [2, 11, 17, 21]—a quasi-experimental design that reasons about the causal effects of interventions by identifying a threshold above or below which an intervention is assigned. We follow this methodology by comparing the projects *before* and *after* a new design feature is introduced. (The intuition is that other conditions (e.g., the project and donor pool) are likely to be very similar right before and after each intervention.) More specifically, let $(x_1, ..., x_k)$ specify the features that impact a project's success (e.g., *school location* or *poverty level*) and t_0 the timestamp when a new design feature is introduced. Let y specify whether a project is successful or not. We fit the following model to the data consisting of projects introduced right before and right after the design change of interest:

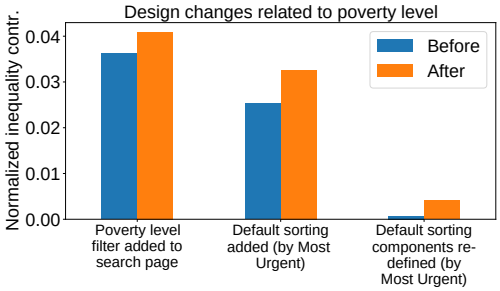
$$y = a_0 + a_1 x_1 + ... + a_k x_k + I(t > t_0)(b_0 + b_1 x_1 + ... + b_k x_k). \quad (2)$$

The estimated coefficients $b_1, ..., b_k$ specify if the intervention has impacted the importance of feature $x_1, .., x_k$ on the success of a project. Unfortunately for our data set, linear regression is not a good fit—making the conclusions from the RDD design unreliable.

To circumvent the issue, we measure the impact of each design change on donors' behavior via another quasi-experimental design: we estimate the average impact of the change by comparing the



(a)



(b)

Figure 6: Changes in the inequality contribution of school location and poverty level before and after certain design changes were introduced in DonorsChoose website.

inequality contribution of different features right before and after it was first introduced. Again, the intuition is that other conditions are likely to be very similar right before and after each intervention. Therefore, comparing the inequality contribution after the intervention (treatment group) to that before the treatment (control group) will deliver the local treatment effect.

Next, we look at several nudges directly related to the school location, introduced in DonorsChoose website at different times:

- **2004:** Location filter was added to the homepage and the project search page. We expect this to increase the contribution of school location. Figure 6a confirms this.
- **2012:** Added 'projects near me' filter to the homepage. We expect this to increase the contribution of school location. Figure 6a further confirms this.
- **2015:** Location search box was added to both home and search pages. We expect this to increase the contribution of school location, and this is indeed consistent with what we observe in Figure 6a.

We now look at several nudges directly related to the school poverty level:

- **2005:** Filter "High level of poverty" was added. We expect this to increase the contribution of poverty level to inequality (i.e., poor schools now have more chance of getting funds). This is what we observe in Figure 6b.
- **2008:** "Most Urgent" criteria became the default ranking of search results (urgency = highest poverty + lowest cost + fewest days left). We expect this to increase the contribution of poverty level. This is consistent with our observation in Figure 6b.
- **2012:** "Most Urgent" criteria was redefined (to highest poverty + closest to finish line). We expect this to increase the contribution of poverty level. This is consistent with what we observe in Figure 6b.

These results show the potential of choice architecture to impact donors' behavior, and we believe that by carefully designing the nudges, the platform can influence the selection rates of projects of different poverty levels and locations in a socially desirable way.

5 CONCLUSION AND FUTURE DIRECTIONS

Recently, there has been a lot of focus on biases in human behavior [7, 22, 26] as well as in algorithmic models [5, 8, 15], and their impacts on different socio-technical systems. Our work sheds light on the power of choice architecture and nudges in shaping the user behavior in online platforms, thus contributing to the observed biases. We conducted an in-depth analysis of the donation platform – DonorsChoose.org, by utilizing longitudinal empirical data. In this process, our work takes an important first step towards adapting nudge frameworks to the social and ethical requirements of online altruism. Many interesting directions remain open for future work:

How can donations be fairly distributed across projects?
To answer this question, we need to first carefully specify a measure of *need* for different projects, which may depend on factors such as the poverty level of the school district, the funding target, project urgency, and others. Given an inter-project-comparable measure of need, one can define fairness as follows: projects with the same level of need should have a similar likelihood of success. Our findings illustrate that the choice architecture can influence project selection rates, and the platform can potentially explore different ways to achieve a fair distribution of donations using nudges.

Ethics of nudges
There has already been much debate about whether and when the use of nudges is ethical [18]. Some refer to nudges as a form of paternalism and restriction of choice and agency, whereas others refer to it as libertarian paternalism – a way of positively influencing people without limiting their freedom of choice [31]. We believe such a discussion should be informed by the context in which nudges are employed. While we did not attempt to tackle these issues in this paper, there are deep ethical concerns that arise when discussing nudges in the context of online charity platforms: For instance, does the goal of promoting more deserving (but less attractive) projects justify the use of the nudges? To what extent? We leave these and many other questions for future work.

Acknowledgments: This research was supported in part by a European Research Council (ERC) Advanced Grant for the project "Foundations for Fair Social Computing", funded under the European Union's Horizon 2020 Framework Programme (grant agreement no. 789373).

REFERENCES

[1] Tim Althoff and Jure Leskovec. 2015. Donor retention in online crowdfunding communities: A case study of donorschoose. org. In *WWW*. 34–44.
[2] Pablo Aragón, Vicenç Gómez, and Andreas Kaltenbrunner. 2017. Detecting platform effects in online discussions. *Policy & Internet* 9, 4 (2017), 420–443.

[3] Sarah Bird, Solon Barocas, Kate Crawford, Fernando Diaz, and Hanna Wallach. 2016. Exploring or exploiting? social and ethical implications of autonomous experimentation in ai. (2016).

[4] Alan S Blinder. 1973. Wage discrimination: reduced form and structural estimates. *Journal of Human resources* (1973), 436–455.

[5] Engin Bozdag. 2013. Bias in algorithmic filtering and personalization. *Ethics and information technology* 15, 3 (2013), 209–227.

[6] Stephen Burke. 2018. #BestSchoolDay 2018: Every Project Funded! https://www.donorschoose.org/blog/best-school-day-2018.

[7] Abhijnan Chakraborty, Saptarshi Ghosh, Niloy Ganguly, and Krishna P Gummadi. 2016. Dissemination biases of social media channels: On the topical coverage of socially shared news. In *ICWSM*. AAAI.

[8] Abhijnan Chakraborty, Johnnatan Messias, Fabricio Benevenuto, Saptarshi Ghosh, Niloy Ganguly, and Krishna P Gummadi. 2017. Who makes trends? understanding demographic biases in crowdsourced recommendations. In *ICWSM*. AAAI.

[9] Frank A Cowell and Carlo V Fiorio. 2011. Inequality decompositions – a reconciliation. *The Journal of Economic Inequality* 9, 4 (2011), 509–528.

[10] Cynthia Cryder, Simona Botti, and Yvetta Simonyan. 2017. The Charity Beauty Premium: Satisfying Donors 'Want' Versus 'Should' Desires. *Journal of Marketing Research* 54, 4 (2017), 605–618.

[11] Sayamindu Dasgupta and Benjamin Mako Hill. 2018. How "wide walls" can increase engagement: evidence from a natural experiment in Scratch. In *CHI*. ACM, 361.

[12] Donorschoose.org Data. 2016. Research for Education. https://research.donorschoose.org/t/download-opendata/33.

[13] Donorschoose.org. 2019. About US. https://www.donorschoose.org/about.

[14] Vincent Etter, Matthias Grossglauser, and Patrick Thiran. 2013. Launch hard or go home!: predicting the success of kickstarter campaigns. In *COSN*. ACM, 177–182.

[15] Megan Garcia. 2016. Racist in the machine: The disturbing implications of algorithmic bias. *World Policy Journal* 33, 4 (2016), 111–117.

[16] Michael D Greenberg, Bryan Pardo, Karthic Hariharan, and Elizabeth Gerber. 2013. Crowdfunding support tools: predicting success & failure. In *CHI Extended Abstracts*. ACM, 1815–1820.

[17] Scott A Hale, Peter John, Helen Margetts, and Taha Yasseri. 2018. How digital design shapes political participation: A natural experiment with social information. *PloS one* 13, 4 (2018), e0196068.

[18] Daniel M Hausman and Brynn Welch. 2010. Debate: To nudge or not to nudge. *Journal of Political Philosophy* 18, 1 (2010).

[19] Eric J Johnson, Suzanne B Shu, Benedict GC Dellaert, Craig Fox, Daniel G Goldstein, Gerald Häubl, Richard P Larrick, John W Payne, Ellen Peters, David Schkade, et al. 2012. Beyond nudges: Tools of a choice architecture. *Marketing Letters* 23, 2 (2012), 487–504.

[20] Chun-Ta Lu, Sihong Xie, Xiangnan Kong, and Philip S Yu. 2014. Inferring the impacts of social media on crowdfunding. In *WSDM*. ACM, 573–582.

[21] Momin M Malik and Jürgen Pfeffer. 2016. Identifying platform effects in social media data. In *ICWSM*. AAAI.

[22] Tanushree Mitra and Eric Gilbert. 2014. The language that gets people to give: Phrases that predict success on kickstarter. In *CSCW*. ACM, 49–61.

[23] Ronald Oaxaca. 1973. Male-female wage differentials in urban labor markets. *International economic review* (1973), 693–709.

[24] U.S. Department of Education. 2017. The Federal Role in Education. https://www2.ed.gov/about/overview/fed/role.html.

[25] Chankyung Pak and Rick Wash. 2017. The Rich Get Richer? Limited Learning in Charitable Giving on donorschoose. org.. In *ICWSM*. AAAI, 172–181.

[26] J Radford. 2016. The emergence of gender inequality in a crowdfunding market: an experimental test of gender system theory. (2016).

[27] Christoph Schneider, Markus Weinmann, and Jan vom Brocke. 2018. Digital nudging: guiding online user choices through interface design. *Commun. ACM* 61, 7 (2018), 67–73.

[28] Itamar Simonson. 1989. Choice based on reasons: The case of attraction and compromise effects. *Journal of consumer research* 16, 2 (1989), 158–174.

[29] Jacob Solomon, Wenjuan Ma, and Rick Wash. 2015. Don't wait!: How timing affects coordination of crowdfunding donations. In *CSCW*. ACM, 547–556.

[30] Till Speicher, Hoda Heidari, Nina Grgic-Hlaca, Krishna P. Gummadi, Adish Singla, Adrian Weller, and Muhammad Bilal Zafar. 2018. A Unified Approach to Quantifying Algorithmic Unfairness: Measuring Individual and Group Unfairness via Inequality Indices. In *KDD*. ACM.

[31] Cass R Sunstein and Richard H Thaler. 2003. Libertarian paternalism is not an oxymoron. *The University of Chicago Law Review* (2003), 1159–1202.

[32] Richard H Thaler and Cass R Sunstein. 2008. *Nudge: Improving decisions about health, wealth, and happiness*. Yale University Press.

[33] Richard H Thaler, Cass R Sunstein, and John P Balz. 2014. Choice architecture. (2014).

[34] Thanh Tran, Madhavi R Dontham, Jinwook Chung, and Kyumin Lee. 2016. How to succeed in crowdfunding: a long-term study in kickstarter. *arXiv preprint arXiv:1607.06839* (2016).

[35] Rick Wash and Jacob Solomon. 2014. Coordinating donors on crowdfunding websites. In *CSCW*. ACM, 38–48.

Multi-Domain Gated CNN for Review Helpfulness Prediction

Cen Chen
Ant Financial Services Group
Hangzhou, Zhejiang, China
chencen.cc@antfin.com

Minghui Qiu
Alibaba Group & Zhejiang University
Hangzhou, Zhejiang, China
minghui.qmh@alibaba-inc.com

Yinfei Yang
Mountain View, CA, United States
yangyin7@gmail.com

Jun Zhou
Ant Financial Services Group
Hangzhou, Zhejiang, China
jun.zhouju@antfin.com

Xiaolong Li
Ant Financial Services Group
Hangzhou, Zhejiang, China
xl.li@antfin.com

Jun Huang
Alibaba Group
Hangzhou, Zhejiang, China
huangjun.hj@alibaba-inc.com

Forrest Sheng Bao
Iowa State University
Ames, IA, United States
forrest.bao@gmail.com

ABSTRACT

Consumers today face too many reviews to read when shopping online. Presenting the most helpful reviews, instead of all, to them will greatly ease purchase decision making. Most of the existing studies on review helpfulness prediction focused on domains with rich labels, not suitable for domains with insufficient labels. In response, we explore a multi-domain approach that learns domain relationships to help the task by transferring knowledge from data-rich domains to data-deficient domains. To better model domain differences, our approach gates multi-granularity embeddings in a Neural Network (NN) based transfer learning framework to reflect the domain-variant importance of words. Extensive experiments empirically demonstrate that our model outperforms the state-of-the-art baselines and NN-based methods without gating on this task. Our approach facilitates more effective knowledge transfer between domains, especially when the target domain dataset is small. Meanwhile, the domain relationship and domain-specific embedding gating are insightful and interpretable.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Transfer learning**.

KEYWORDS

Review helpfulness prediction; transfer learning

ACM Reference Format:
Cen Chen, Minghui Qiu, Yinfei Yang, Jun Zhou, Xiaolong Li, Jun Huang, and Forrest Sheng Bao. 2019. Multi-Domain Gated CNN for Review Helpfulness Prediction. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313587

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313587

1 INTRODUCTION

Product reviews, primarily text, provide important information for consumers to make purchase decisions. Hence, it makes great business sense to quantify the quality of reviews and present consumers useful reviews in an informative manner. In fact, major E-commerce websites like Amazon.com encourage users to rate whether a review is helpful or not. However, most reviews do not have helpfulness scores or votes. Fewer than 20% of the reviews in Amazon Review Dataset [20] have 5+ votes while only 0.44% have 100+ votes. In addition, the review voting itself may be biased [5, 8]. It is thus interesting yet important to automatically infer the helpfulness score of any review.

Recently, there are growing interests from both the academia and the industry on such *review helpfulness prediction task* [17, 19, 30, 31]. Pioneering work hypothesizes that helpfulness is an underlying property of the text, and uses handcrafted linguistic feature to study it [17, 19, 30, 31]. Unfortunately, most of them focus on domains with rich labels, and may not be practical for domains with insufficient data. For example, the "Electronics" domain from the Amazon Review Dataset has more than 354k labeled reviews, while "Watches" has less than 10k [30]. To alleviate, we study Transfer Learning (TL) for review helpfulness prediction. Specifically, we propose a multi-domain approach that transfers knowledge from multiple data-rich domains to data-deficient domains. Inspired by the remarkable performance and efficiency of Convolutional Neural Networks (CNNs) on many tasks in natural language processing [6, 13], we adopt CNNs to prove the concept of our TL framework for the task. Our proposed framework is a multi-domain gated NN model based on two important observations below (arranged in the order of the data processing flow.)

First, while embeddings of words are equally used downstream in previous applications of CNNs in NLP, actually different words have different and domain-dependent impacts to review helpfulness. For example, the top aspect extracted by the model in [30] for "Watches" domain is "Warranty" whereas "Portablity" for "Electronics" domain, implying that words related to "Warranty" are

important to "Watches," while words related to "Portability" are important to "Electronics." Hence, we introduce the *word-level gating* mechanism to weight embeddedings[1].

Although gating mechanisms have been used in RNNs [7, 26, 29], our gating method is complimentary for being word- and domain-specific, i.e., one gate weight for one word in one domain. Our gating mechanism can help selectively memorize the input representations of the words and scores the relevance/importance of such representations in different domains, to provide insightful word-level interpretations for the TL prediction results. Meanwhile, to the best of our knowledge, our gating mechanism is the first to study cross-domain word-level importance.

Second, domain difference is indeed a pending problem in review analysis, as agreed by previous work [6, 17, 30]. When transferring knowledge, not all domains are helpful to a target domain. We propose multi-domain relationship learning to exploit cross domain relationships to properly transfer knowledge from *related* domains with sufficient labeled data to domains with limited reviews. It is worth noting that, existing studies on this task only focus on a single product category or largely ignore the inner correlations between different domains [6]. Previous work show some features are domain-specific while others can be shared. For example, image quality features are only useful for cameras [30], while semantic features and argument-based features are applicable for all domains [17, 31]. While there are some common practices to transfer knowledge between domains [21, 32], such as using a shared NN. We urge *domain correlations* to be established before the knowledge can be transferred properly for review helpfulness prediction, otherwise, transferring the knowledge from a wrong source domain may backfire. We thus provide a holistic solution to both domain correlation learning and knowledge transfer by incorporating domain relationship learning in our framework.

Furthermore, we propose to enrich the word representation in a vanilla CNN by considering cross-domain *character-* and *aspect-level* representations. Character-level representations are notably beneficial for alleviating the out-of-vocabulary problem [2, 14], while aspect distribution provides another semantic view on words [30].

In all, our model explores cross-domain word representations, multi-domain word gating, and multi-domain relationship learning to help the task. As shown in Figure 1, our network has four types of layers: (1) shared embedding layers, (2) gating layers, (3) convolutional layers, and (4) output layers with relationship learning. Experimental results show that our final model can correctly tap into domain correlations and facilitate the knowledge transfer between correlated domains. In all, we summarize our contributions following the network architecture (left to right in Figure 1) below:

- **Rich word representations:** We study CNN for review helpfulness prediction task, and further enrich our word representations by incorporating multi-granularity information, i.e., character-, word-, and aspect- based embeddings;
- **Domain specific gating layer:** We proposed a novel domain-specific gating mechanism to learn/interpret the important/non-important words in reviews and to boost model performance.
- **Cross-domain relationship learning:** We are the first to study cross-domain relationship learning for the task and show that the cross-domain correlations learned are insightful and beneficial for the task;
- **Evaluation:** Extensive experiments show that our model significantly outperforms the state of the art and can transfer knowledge to help target domain with limited data.

[1]The gates are applied over all three types of representations (i.e., character-, word-, and aspect-based) for all words and weights are domain-specific.

2 RELATED WORK

Review helpfulness prediction is a task close to but *different from sentiment analysis* [16]. The goal of sentiment analysis is to detect the emotion behind an opinion, whereas review helpfulness aims to identify helpful information like reasoning etc., regardless of the emotion nor the opinion [17, 31].

Recent studies on review helpfulness prediction extract handcrafted features from the review text. For example, semantic features used in [31] and [19], aspect features in [30] and argument-based features in [17]. These methods require prior knowledge and human effort in feature engineering and may not be robust for new domains. CNNs with the ability to extract deep features, have demonstrated remarkable performance over many NLP tasks, for its high-efficiency and comparable performance to Recurrent Neural Networks (RNNs) [13]. Our initial study shows that vanilla RNN [27] does not outperform our CNN models. It echoes the finding that RNNs are more robust but do not guarantee to have better results than CNNs [33]. Thus, we employ CNNs for our task. Since character-level representations are notably beneficial for alleviating the out-of-vocabulary [14] problem, while aspect distribution provides another semantic view on words [30], we further enrich our word representation by adding character- and aspect-based representations.

Unlike a single domain model, our work focuses on leveraging reviews from multiple domains. As a word may play different importance on different domains, we consider to weight word representations by adding word-level gates. Note that some gating mechanisms have been used in RNNs [7, 10]. The work in [15] explored contextual word attentions in different tasks. Different from these studies, our word-level gates are domain-specific and help further differentiate important and non-important words in different domains.

It is common that some domains have rich user reviews while other domains may not. To help domains with limited data, we study cross-domain learning, i.e., transfer learning (TL) [22] or multi-task learning (MTL) [36] for this task. There are generally two lines of studies for TL. The first is supervised domain adaptation, which assumes enough labeled data from source domains and also a little labeled data from a target domain [9]. The second assumes having labeled data from source domains only and some unlabeled data from a target domain [3, 35]. Our study belongs to the first line of work. Typical method in this line is to find a shared feature space which can reduce the divergence between the distributions of the source and target domains [1, 28]. In this work, we adapt CNN model to learn shared feature representations.

With the popularity of deep learning, a great amount of Neural Network (NN) based methods are proposed for TL [6, 11, 12, 34]. A typical framework is to use a shared NN to learn shared features for

both source and target domains [21, 32]. Another approach is to use both a shared NN and domain-specific NNs to derive shared and domain-specific features [18]. A multi-domain relationship learning method is introduced in [37], to uncover the relationship between domains. Inspired by this, we adopt the relationship learning in our gated multi-domain framework to help model the correlation between different domains. Experiments show that the multi-domain relationship learning method indeed helps capture the correlations between different domains and can further boost our model's performance by better leveraging knowledge from correlated domains. We have also visualized the learned domain correlations and find them insightful.

3 MODEL

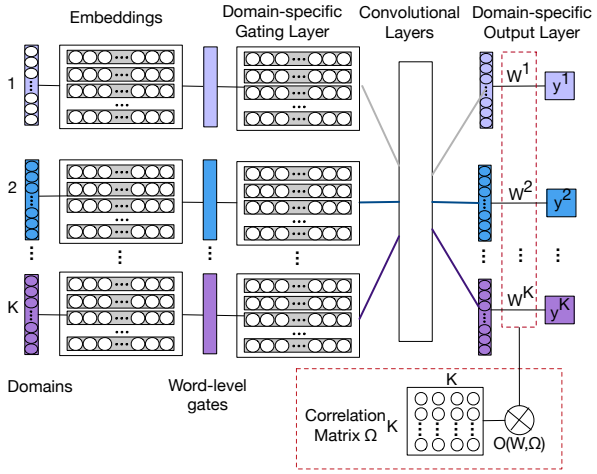


Figure 1: Proposed multi-domain gated CNN model. Note that the embedding and convolutional layers are shared across domains, while the gating and output layers are domain-specific. U and $W^{(\cdot)}$ are shared and domain-specific output layers, respectively. The parameter Ω models domain correlations.

Review helpfulness prediction is defined as a regression task to predict the helpfulness score given a text review [31]. The ground truth of helpfulness is determined using the "a of b approach": a of b users think the review is helpful.

In this work, we consider a *cross-domain review helpfulness prediction task* given labeled reviews from multiple source domains and a target domain. We seek to transfer knowledge from data-rich source domains to a target domain. For a review $\mathbf{X}^k$, our goal is to predict its helpfulness score y^k, where $k \in [0..K]$ is the domain label indicating which domain the data instance is from.

As shown in Fig. 1, Our proposed multi-domain gated CNN has four components: embedding layers, gating layers, convolutional layers, and output layers with relationship learning. Embedding and convolutional layers are shared across domains, while other layers are domain-specific.

3.1 Shared embedding layers

Our word embeddings contain word-level, character-level, and aspect-level representations.

3.1.1 Word and Character Representations. A review $\mathbf{X}$ consists of a sequence of words, i.e., $\mathbf{X} = [x_1, x_2, \ldots, x_m]$. Following the model in [13], for words in a review $\mathbf{X}$, we first lookup their embeddings $(\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_m)$ from an embedding matrix $E \in \mathbb{R}^{|V| \times D}$ where $|V|$ is vocabulary size and D is embedding dimension, and $\mathbf{e}_i \in \mathbb{R}^{D \times 1}$. This word embedding matrix is then fed into a convolutional neural network to obtain an output representation. This is a typical *word embedding* based model.

In many applications, such as text classification [4] and machine reading comprehension [25], it is beneficial to enrich word embeddings with subword information. Inspired by that, we consider to use a character embedding layer to obtain *character embeddings* to enrich word representations. Specifically, the characters of the i-th word x_i are embedded into vectors and then fed into another convolutional neural network to obtain a fixed-sized vector $\texttt{CharEmb}(x_i)$.

3.1.2 Aspect Representations. A recent work in [30] shows that extracting the aspect/topic distribution from the raw textual contents does help the task of review helpfulness prediction. The reason is that many helpful reviews tend to mention certain aspects, like "brand," "functionality," or "price," of a product. Inspired by this, we enrich our word representations by aspect distributions. We adopt the generative model in [30] to learn aspect-word distributions $\Phi \in \mathbb{R}^{|V| \times A}$, where A is aspect size and $|V|$ is the size of vocabulary. Note, aspect representations learned are shared across domains.

A word-aspect representation Φ' is obtained by row-wise normalization of the matrix Φ. Then for each word x_i in input review $\mathbf{X}$, we obtain aspect representation by looking up the matrix Φ' to get $\Phi'_i \in \mathbb{R}^{A \times 1}$.

3.1.3 Putting all three embeddings together. In all, for an input review $\mathbf{X}$, we obtain its representation as:

$$\mathbf{e}_X = [\mathbf{e}'_1, \mathbf{e}'_2, \ldots, \mathbf{e}'_m], \tag{1}$$

$$\mathbf{e}'_i = \mathbf{e}_i \oplus \texttt{CharEmb}(x_i) \oplus \Phi'_i, \quad \forall i \in [1..m], \tag{2}$$

where $\mathbf{e}_i$, $\texttt{CharEmb}(x_i)$, and Φ'_i are word-, character-, and aspect-representations respectively, and $\oplus$ is a stacking operator. Note that m (=100 in this paper) is the sentence length limit. Sentences shorter than m words will be padded with zeros, while those longer than m words will be truncated.

3.2 Domain-specific gating layers

It is intuitive that given a domain, some words play more important roles to the helpfulness of reviews. For example, "warranty" words are more important for "Watches" domain than other domains, and descriptive or semantic words (such as "great battery life" or "versatile function") are more informative than general background words like "phone" in "Cellphone" domain [30]. Hence, we propose to weight the input word embeddings by a multi-domain gating mechanism. Specifically, the input word representations are first transformed with a weighting matrix $\mathbf{V}_g$ and concatenate with a domain-specific embedding $\mathbf{V}_k$ for domain k. The final output is then fed into a fully connected layer to obtain final word-level gates.

More specifically, for each input word x_i, we learn its weight g_i through the gating mechanism. And for input $\mathbf{X} = [x_1, \ldots, x_m]$, we obtain its representation $\mathbf{e'}_X$ as:

$$\mathbf{e}_X = [\mathbf{e}'_1, \mathbf{e}'_2, \ldots, e'_m], \tag{3}$$

$$g_i = \texttt{FC}(\mathbf{V}_g^\top \mathbf{e}'_i \oplus \mathbf{V}_k), \quad \forall i \in [1..m], \tag{4}$$

$$\mathbf{e'}_X = [g_1\mathbf{e}'_1, g_2\mathbf{e}'_2, \ldots, g_m\mathbf{e}'_m], \tag{5}$$

where $\mathbf{V}_g$ is a transform matrix, $\mathbf{V}_k$ is domain-specific weights, and FC represents fully connected layers with sigmoid activations. The same weight g_i is applied to all dimensions of the embedding $\mathbf{e}'_i$ for word x_i.

3.3 Shared convolutional layers

Next, we stack a 2-D convolutional layer and a 2-D max-pooling layer on the matrix $\mathbf{e}'_X$ to obtain the hidden representation h_X. Multiple filters are used, where for each, we obtain a hidden representation h_f. All these representations are concatenated to form the final representation h_X.

$$h_f = \texttt{CNN}\Big(\mathbf{e}'_X, \text{filterSize} = \big(f, D, C\big)\Big),$$
$$h_X = h_2 \oplus h_3 \oplus h_4 \oplus h_5. \tag{6}$$

where $f \in \{2, 3, 4, 5\}$ is window size, D is embedding dimension, C is channel size, and $\texttt{CNN}(\cdot)$ represents a convolution layer followed by a max-pooling layer [2]. We refer our base model as Embedding-Gated CNN (EG-CNN).

3.4 Domain-specific output layers

If all the domains are homogeneous, we can build an unified model for our task to optimize the following objective:

$$l = \sum_k \sum_{\mathbf{X}^k \in \mathcal{D}^k} \left(\mathbf{U}^\top \texttt{EG-CNN}\left(\mathbf{X}^k\right) - y^k\right)^2 + l_{reg}, \tag{7}$$

where $\mathbf{U}$ is the output layer weights, $\mathbf{X}^k$ is a review in the set $\mathcal{D}^k$ containing all reviews from domain k, y^k is the corresponding label, l_{reg} is a regularization term.

However, *domains are not homogeneous.* The formulation in Eqn. (7) is limited because it does not take domain differences into consideration. To utilize cross-domain knowledge, we convert the method above to a multi-domain setting where besides the shared output layer $\mathbf{U} \in \mathbb{R}^{P\times 1}$, we add a domain-specific output layer $\mathbf{W}_k$ for each domain k. Assuming $\mathbf{W}_k \in \mathbb{R}^{P\times 1}$, where P is output layer size, we have $\mathbf{W} = [\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_K]$. Unlike a unified model to learn universal feature representation, our *cross-domain relationship learning* approach has two output layers $\mathbf{U}$ and $\mathbf{W}$ to model domain *commonalities* and *differences* respectively.

Following matrix-variate distribution setting in [37], we assume:

$$\mathbf{W} = q(W) \prod_{i=1}^{K} \mathcal{N}\left(W_i \mid \mathbf{0}_P, \epsilon_i^2 \mathbf{I}_P\right), \tag{8}$$

$$q(W) = \mathcal{MN}_{P\times K}\Big(W \mid \mathbf{0}_{P\times K}, \mathbf{I}_P \times \Omega\Big), \tag{9}$$

where $\mathcal{N}(\cdot \mid A, B)$ and $\mathcal{MN}(\cdot \mid A, B)$ are normal distribution and matrix-variate normal distribution with A as mean and B as variance/covariance matrix respectively, $\mathbf{0}$. is a zero vector/matrix, $\mathbf{I}_P$ is identity matrix. The covariance matrix $\Omega \in \mathbb{R}^{K\times K}$ models domain correlations, where $\Omega_{i,j}$ is the correlation between domains i and j.

As shown in [37], the above formulas are resolved to optimize the trace of the matrix product $\text{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^\top)$. Hence, our objective can be defined as follows:

$$\begin{aligned} l \quad = \quad & \sum_k \sum_{X^k \in \mathcal{D}^k} ((\mathbf{U} + \mathbf{W_k})^\top \texttt{EG-CNN}(\mathbf{X}^k) - y^k)^2 + \\ & \lambda_1 \text{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^\top) + \lambda_2 l_{reg}, \\ \text{s.t.} \quad & \Omega \succeq 0, \quad \text{tr}(\Omega) = 1. \end{aligned} \tag{10}$$

where $\text{tr}(\cdot)$ gets the trace of a matrix, l_{reg} is a regularization term, λ_1 and λ_2 are weight coefficients. To minimize the trace term $\text{tr}(\mathbf{W}\Omega^{-1}\mathbf{W}^\top)$, when domain i and domain j are close, i.e., $\mathbf{W}_i$ is close to $\mathbf{W}_j$, the model tends to learn a large $\Omega_{i,j}$. And when domain i and j are distant, the model tends to learn a small $\Omega_{i,j}$.

3.5 The final model

In all, our final model has four types of layers: embedding layers, gating layers, convolutional layers, and output layers. In the output layers, we consider learning domain correlations in the multi-domain setting. Note that, if we set the correlation matrix Ω as an identity matrix (no domain correlation) and $\mathbf{U} = \mathbf{0}$ (no shared output layer), the multi-domain setting is degenerated to a fully-shared setting in [21]. The limitation of the fully-shared setting is that it ignores domain relationships. However, in practice, we may think that reviews from some domains (e.g., the domain "Electronics") are helpful to the task on other domains (e.g., "home" and "cellphones") but not all. With our model, we seek to automatically capture such domain relationships and use that information to help boost model performance.

4 INFERENCE

Let $\boldsymbol{\theta}$ be the model parameters in our model. It is not easy to optimize the objective function in Eqn. (10) with respect to Ω, $\mathbf{W}$, $\mathbf{U}$, and $\boldsymbol{\theta}$. Alternatively, we use a stochastic alternating method [37] to first optimize $\boldsymbol{\theta}$, $\mathbf{W}$, and $\mathbf{U}$ through fixing Ω, and then optimize Ω by fixing $\boldsymbol{\theta}$, $\mathbf{W}$, and $\mathbf{U}$. The procedure is repeated within a given iteration or until the convergence criterion is met.

Optimizing $\mathbf{W}$, $\mathbf{U}$ *and* $\boldsymbol{\theta}$. While Ω is fixed, the objective function is convex. We can use gradient descent method to optimize $\mathbf{W}$, $\mathbf{U}$, and $\boldsymbol{\theta}$ jointly. The gradients can be computed by back propagation method with details in [24].

Optimizing Ω. With the fixed $\mathbf{W}$, $\mathbf{U}$, and $\boldsymbol{\theta}$, our objective is to optimize $\mathbf{min_\Omega\ tr(W\Omega^{-1}W^\top)}$ with regard to $\Omega \succeq \mathbf{0}$ and $\mathbf{tr(\Omega) = 1}$.

According to [37], the convex problem above has a closed-form solution: $\Omega = \frac{(\mathbf{W}^\top\mathbf{W})^{\frac{1}{2}}}{\text{tr}\left((\mathbf{W}^\top\mathbf{W})^{\frac{1}{2}}\right)}$. If $\mathbf{W_i}$ and $\mathbf{W_j}$ are close, $\mathbf{\Omega_{i,j}}$ will be be large. $\mathbf{W^\top W}$ is a symmetric matrix, whose square root can be computed by singular-value decomposition (SVD).

[2] Our gated TL framework can work with any sentence representation model, where CNN layer here is chosen as an example to prove the concept for the task.

5 EXPERIMENTS

We use reviews from 5 different categories/domains from the public Amazon Review Dataset [20] in the experiments. The 5 categories are: watches, Phones, Outdoor, Home, and Electronics, each with 9k, 18k, 73k, 219k, 354k data instances respectively. The 5 categories are chosen in accordance with the practices of previous work on this topic [6, 30, 31].

In our model, we initialize the lookup table $\mathbf{E}$ with pre-train word embeddings from *GloVe* [23] by setting $\boldsymbol{D = 100}$. For aspect representation, we adopt the settings from [30] to set the aspect size to 100. For the convolutional layer, the activation function is ReLU, the channel size is set to 128. The output layer size of $\mathbf{U}$ and $\mathbf{W}_{\boldsymbol{k}}$ are 128, i.e., $\boldsymbol{P = 128}$. We use data from five domains, hence $\boldsymbol{K = 5}$. AdaGrad is used with an initial learning rate of 0.08.

Following the previous work on helpfulness prediction, all experiment results are evaluated in the *Pearson correlation coefficient* between the predicted helpfulness score and the ground truth score. The ground truth scores are computed by "a of b approach", indicating the percentage of consumers thinking a review as useful.

5.1 EG-CNN vs. the state of the art and vanilla CNNs without gating

We first compare EG-CNN, i.e., our full base model with rich word representations and domain-specific gates, against those based on traditional linguistic features. Following [31], five baselines use handcrafted features "STR", "UGR", "LIWC", "INQ" and aspect-based features "ASP" [30]", respectively, are established. We further consider two fusional baselines with ensemble linguistic features: Fusion_1 with "STR", "UGR", "LIWC", and "INQ" features [31] and Fusion_2 with additional "ASP" features [30]. Different vanilla CNN-based models without gating are also compared to examine the effectiveness of different embeddings and domain-specific gates: CNN - the vanilla CNN model in [13], $\text{CNN}_{\boldsymbol{c}}$ - the vanilla CNN model with additional character-based representation used in [6].

Table 1: Performance of EG-CNN variants and linguistic feature baselines in Pearson correlation coefficient. † and ‡ mean the method outperforms the best baseline with $\boldsymbol{p < 0.05}$ and $\boldsymbol{p < 0.1}$ by the Student's paired t-test in terms of prediction errors respectively.

		Watches	Phones	Outdoor	Home	Elec.
Baselines	STR	0.276	0.349	0.277	0.222	0.338
	UGR	0.425	0.466	0.412	0.309	0.355
	LIWC	0.378	0.464	0.382	0.331	0.400
	INQ.	0.403	0.506	0.419	0.366	0.405
	ASP	0.406	0.437	0.385	0.283	0.406
	Fusion_1	0.488	0.539	0.497	0.432	0.484
	Fusion_2	0.493	0.550	0.501	0.436	0.491
CNNs without gating	CNN	0.480	0.562	0.501	0.459	0.524
	$\text{CNN}_{\boldsymbol{c}}$	0.495	0.566	0.511	0.464	0.521
Our method	EG-CNN	$\mathbf{0.515}^\dagger$	$\mathbf{0.585}^\dagger$	$\mathbf{0.555}^\dagger$	$\mathbf{0.541}^\dagger$	$\mathbf{0.544}^\ddagger$

Table 1 shows several interesting observations that validate our motives behind this work. First, even the vanilla CNN outperforms the baselines, indicating the expressive superiority of NN-extracted features over handcrafted features. Second, increasing the level of embedding yields better performance as vanilla CNN is outperformed by $\text{CNN}_{\boldsymbol{c}}$. In all cases, EG-CNN outperforms all the CNN variants, which shows the importance of adding aspect-based representations and the gating mechanism of our model. Last, EG-CNN consistently outperforms all the baselines. This further shows the advantage of our base model, i.e., EG-CNN, for the task.

5.2 Our full model with domain relationships

To evaluate the effectiveness of our domain relationship learning, we compare our proposed full model against the following baselines:

- Source-only (Src-only): EG-CNN model that does not consider target data. We use the largest domain 'Elec.' as source.
- Target-only (Tgt-only): EG-CNN model that does not consider source data, it is equivalent to train our model on each domain individually by setting $\mathbf{W = 0}$.
- Fully-Shared transfer learning model (FS): a variant of our full model by setting $\mathbf{U = 0}$, which is identical to the fully-shared TL model in [21] except that our base model is EG-CNN with rich word representations;
- Transfer Learning with Adversarial loss (TL_{adv}): a recently proposed TL model for the task with adversarial loss [6], which is extended from [18]. For fair comparison, the base model used in TL_{adv} is also replaced with EG-CNN.
- Our full model (Ours): our final model with both $\mathbf{U}$ and $\mathbf{W}$ in a domain relationship learning setting.

Table 2: Comparisons between our model and TL models

		Watches	Phones	Outdoor	Home	Elec.
Baselines	Src-only	0.481	0.502	0.501	0.445	0.544
	Tgt-only	0.515	0.585	0.555	0.541	0.544
	FS	0.522	0.580	0.551	0.518	0.534
	TL_{adv}	0.525	0.588	0.556	0.541	0.545
	Ours	$\mathbf{0.535}^\dagger$	$\mathbf{0.592}^\ddagger$	$\mathbf{0.561}^\dagger$	**0.544**	**0.548**

According to results in Table 2, Tgt-only outperforms Src-only, which is intuitive as source domains are close to but different from the target domain. On all product domains, our model consistently achieves better results than Src-only, Tgt-only, FS, and $\text{TL}_{\boldsymbol{adv}}$, supporting the effectiveness and benefit of multi-domain relationship learning. The improvement is greater on domains with fewer labeled data, e.g., the "Watches" domain which has the least number of reviews and where our model shows the most improvement.

Interestingly, the FS model performs much worse than the Tgt-only model in the "Home" domain. This might be justified by the potential domain shift, under which the FS model may not perform better than the Tgt-only model. Because some domains are more related while some others are more different, incorporating data from those less related can hardly help, especially when the target domain (such as "Home") has sufficient data for the Tgt-only model to perform well enough.

The domain correlations presented in Figure 2 provides further evidence into this. The "Home" domain is positively correlated with the two smallest domains, "Watches" and "Phones," and negatively correlated with two other data-rich domains, "Outdoor" and "Electronics." When relevant domains have much less data than

uncorrelated domains, the FS model will cause non-ideal transfer of knowledge, just like what we have seen on the "Home" domain. On a separate note, our model is more robust that it outperforms all the baselines, thanks to our model's ability to learn domain correlations and leverage knowledge from the right domains.

5.3 Is more data equal to more knowledge?

We study whether more data equals to more knowledge by varying the amount of source or target data available. A transfer learning setting from the largest domain "Electronics" to a target domain "Phones" is used as a case study.

Table 3: Performances w.r.t. different amounts of target data.

% of Phone Data	25%	50%	75%	100%
# of Reviews	19.7k	39.4k	59.2k	78.9k
Target-only	0.453	0.532	0.564	0.585
Our full model	0.481	0.544	0.575	0.594
Improvement	6.2%	2.3%	2.0%	1.5%

We study how performance changes along with different amounts (1/4, 1/2, 3/4, and 100% of all, data pre-scrambled) of reviews from the target domain used in the case study. The performances of both our full model and the Tgt-only model are presented in Table 3. Our model has the most advantage (6.2%) over Tgt-only model when only 25% of target domain data is used. The more target data involved in training, the smaller the advantage is. When all of the target domain data is used, the advantage shrinks to 1.5%. It suggests that our model helps more when the target domain is more data-deficient. The reason behind is that the task on target domain benefits little from the knowledge transferred from other domain when target domain data itself is enough to train a good model.

5.4 Domain relationships

We lastly examine the *domain correlation* Ω learned by our model. The correlation matrix is a 5×5 matrix where each value indicating the correlation between two domains. As in Figure 2, domain correlation learned is reasonable. For example, domain "Phone" is close to "Home" and "Elec.," but not so close to "Watch" and "Outdoor."

	Watch	Phone	Outdoor	Home	Elec.
Watch	1	-0.096	-0.41	0.128	-0.008
Phone	-0.096	1	-0.089	0.166	0.206
Outdoor	-0.41	-0.089	1	-0.2	-0.014
Home	0.128	0.166	-0.2	1	-0.279
Elec.	-0.008	0.206	-0.014	-0.279	1

Figure 2: An illustration on domain correlation. A redder cell represents a closer domain relationship between the two domains.

5.5 Interpretability of embedding gates

To examine the word-level gates in our proposed model, we visualize the gate weights learned in Figure 3. We randomly choose one sentence from the review text from the 'Watch' domain and plot a gray-scale heat map according to the gate weights learned for the words. The weights, ranging from 0 (unimportant) to 1 (important), are placed under each word. The redder a cell is, the closer a word's weight is to 1.

i	be	really	disappoint	in	this	product	i
0.2517	0.2405	0.4392	0.5086	0.4541	0.2902	0.3993	0.2517
have	a	watch	that	i	could	not	get
0.4271	0.3001	0.3395	0.336	0.2517	0.4097	0.3348	0.4114
the	pin	all	the	way	back	in	purchase
0.3385	0.3949	0.3814	0.3385	0.4131	0.4975	0.4541	0.4106
this	hope	to	resolve	the	problem	the	pin
0.2902	0.5286	0.3529	0.6678	0.3385	0.5587	0.3385	0.3949
bit	bent	and	could	not	handle	the	task
0.5564	0.5617	0.3673	0.4097	0.3348	0.5445	0.3385	0.5777

Figure 3: An visualization of word-level gate weights, ranging from 0 (unimportant) to 1 (important) from a review sample in the 'Watch' domain. The redder a cell is, the closer a word's weight is to 1.

The example in Figure 3 is very interpretable. Stop words, e.g., "i", "be"[3], and "the", and general background words, such as "product" and "watch" in the "Watches" domain, have small weights, whereas sentiment words and adjectives are assigned with large weights, such as "disappoint," "careful," and "error." In summary, we find that the word-level gate weights learned are helpful for improving model performance and provide insightful word-level interpretations for the prediction.

Furthermore, we visualize the top important and non-important words indicated by the word-level gates learned by our model in all the five domains in Table 4. First, important words and non-important words are easily distinguishable. For example in the "Watch" domain, important words include words related to opinions like "angry" and "disappointment," or aspects like "warranty." And non-important words contain stop words like "i," "be," and some words that do not carry specific meanings like "le" and "de." Similarly in other domains, we find the top important words are far more meaningful than the unimportant words.

Table 4: Top important/unimportant words in five domains.

Domain	Top Important Words
Watch	cut-out,flimsier,unsuccesfully,angry,warranty,disappointment
Phone	unplayable,retransmit,equipped,trouble-free,screen,well-protected
Outdoor	bizzare,wired,niggles,slumberland,stuggle,double-a,right-angle
Home	unsuccesfully,bizzare,button-down,stone-age,creasing,wierd,home
Elec.	units,superiores,unsuccesfully,know-nothing,exterior,checkable
Domain	**Top Unimportant Words**
Watch	le,de,como,i,bonito,be,monte,hoy,ser,in,the,alta,-,um,mil,monte
Phone	para,le,por,la,eso,todo,mudo,como,hace,gran,the,-,musica,casa
Outdoor	ha,hoy,de,le,an,como,the,um,despues,ecuador,6,-,vinal,mas,this,km
Home	mi,por,ser,ha,lo,como,eso,nah,the,quart,doubly,para,alta,estas,fue
Elec.	high,optional,fiber,play,an,this,meter,is,playing,inch,-,mb,the,ha

Second, in general, each domain has its own set of important words. For example,"warranty" in "Watches," and "trouble-free" and "screen" in "Phones." And the top 10 important words in "Watches" domain differ from those in "Phones." Interestingly, some words are important across different domains. For example, "unsuccessfully"

[3] All the words in our model are lemmatized.

is important to "Watches," "Home,", and "Electronics" domains, and "bizzare" is important to "Outdoor" and "Home" domains.

Last but not least, almost all of the top 10 unimportant words are hard to make sense of, e.g. words like "de," "le," and "lo." We find some background words like "fiber," "meter," and "play" are regarded as unimportant for "Electronics". This is also intuitive as the high occurrences of these words in "Electronics" domain makes them less important.

To summarize, the multi-domain word-level gates are interpretable and provide us an insight into the important and unimportant words in different domains. And such word-level gates can help boost model performance.

6 CONCLUSION

In this work, we tackle the review helpfulness prediction in a cross domain transfer learning framework. We built our base model on EG-CNN with word-, character- and aspect-based representations. On top of this model, domain-specific gates and domain relationships were learned to better transfer knowledge across domains. Extensive experiments showed that our model significantly outperforms the state of the art, and the domain relationships and embedding gates learned are insightful.

REFERENCES

[1] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. 2007. Multi-task feature learning. In *NIPS*. 41–48.

[2] Miguel Ballesteros, Chris Dyer, and Noah A Smith. 2015. Improved transition-based parsing by modeling characters instead of words with LSTMs. *arXiv:1508.00657* (2015).

[3] John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *EMNLP*. 120–128.

[4] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *CoRR* abs/1607.04606 (2016). http://arxiv.org/abs/1607.04606

[5] Qing Cao, Wenjing Duan, and Qiwei Gan. 2011. Exploring determinants of voting for the "helpfulness" of online user reviews: A text mining approach. *Decision Support Systems* 50, 2 (2011).

[6] Cen Chen, Yinfei Yang, Jun Zhou, Forrest Sheng Bao, and Xiaolong Li. 2018. Cross-Domain Review Helpfulness Prediction Based on Convolutional Neural Networks with Auxiliary Domain Discriminators. In *NAACL*.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555* (2014).

[8] Cristian Danescu-Niculescu-Mizil, Gueorgi Kossinets, Jon Kleinberg, and Lillian Lee. 2009. How Opinions Are Received by Online Communities: A Case Study on Amazon.Com Helpfulness Votes. In *WWW*. 141–150.

[9] Hal Daume III. 2007. Frustratingly Easy Domain Adaptation. In *ACL*. 256–263.

[10] Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. 2016. Gated-attention readers for text comprehension. *arXiv:1606.01549* (2016).

[11] Xin Huang, Yuxin Peng, and Mingkuan Yuan. 2017. Cross-modal Common Representation Learning by Hybrid Transfer Network. *CoRR* abs/1706.00153 (2017).

[12] Xin Huang, Yuxin Peng, and Mingkuan Yuan. 2017. MHTN: Modal-adversarial Hybrid Transfer Network for Cross-modal Retrieval. *CoRR* abs/1708.04308 (2017).

[13] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *EMNLP*, 1746–1751.

[14] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-Aware Neural Language Models.. In *AAAI*. 41–49.

[15] Zheng Li, Ying Wei, Yu Zhang, and Qiang Yang. 2018. Hierarchical Attention Transfer Network for Cross-Domain Sentiment Classification. In *AAAI*.

[16] Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies* 5, 1 (2012), 1–167.

[17] Haijing Liu, Yang Gao, Pin Lv, Mengxue Li, Shiqiang Geng, Minglan Li, and Hao Wang. 2017. Using Argument-based Features to Predict and Analyse Review Helpfulness. In *EMNLP*. Copenhagen, Denmark, 1358–1363.

[18] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial Multi-task Learning for Text Classification. In *ACL*. 1–10.

[19] Lionel Martin and Pearl Pu. 2014. Prediction of Helpful Reviews Using Emotions Extraction. In *AAAI*. 1551–1557.

[20] J McAuley and J Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*. 165–172.

[21] Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2016. How Transferable are Neural Networks in NLP Applications?. In *EMNLP*. 479–489.

[22] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[23] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*.

[24] D.E. Rumelhart, G.E. Hintont, and R.J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323 (1986).

[25] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional Attention Flow for Machine Comprehension. *CoRR* abs/1611.01603 (2016). http://arxiv.org/abs/1611.01603

[26] Chuanqi Tan, Furu Wei, Qingyu Zhou, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. 2018. Context-Aware Answer Sentence Selection With Hierarchical Gated Recurrent Neural Networks. *TASLP* 26, 3 (2018), 540–549.

[27] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *EMNLP*. 1422–1432.

[28] Chang Wang and Sridhar Mahadevan. 2008. Manifold alignment using procrustes analysis. In *ICML*. 1120–1127.

[29] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated Self-Matching Networks for Reading Comprehension and Question Answering. In *ACL*. 189–198.

[30] Yinfei Yang, Cen Chen, and Forrest Sheng Bao. 2016. Aspect-Based Helpfulness Prediction for Online Product Reviews. In *ICTAI*.

[31] Yinfei Yang, Yaowei Yan, Minghui Qiu, and Forrest Bao. 2015. Semantic Analysis and Helpfulness Prediction of Text for Online Product Reviews. In *ACL-IJCNLP*. 38–44.

[32] Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. In *ICLR*.

[33] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. 2017. Comparative Study of CNN and RNN for Natural Language Processing. *CoRR* abs/1702.01923 (2017). arXiv:1702.01923 http://arxiv.org/abs/1702.01923

[34] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks?. In *NIPS*. 3320–3328.

[35] Jianfei Yu and Jing Jiang. 2016. Learning Sentence Embeddings with Auxiliary Tasks for Cross-Domain Sentiment Classification. In *EMNLP*. 236–246.

[36] Yu Zhang and Qiang Yang. 2017. A survey on multi-task learning. *arXiv:1707.08114* (2017).

[37] Yu Zhang and Dit-Yan Yeung. 2010. A convex formulation for learning task relationships in multi-task learning. In *UAI*.

Collaborative Similarity Embedding for Recommender Systems

Chih-Ming Chen*
National Chengchi University
Taipei, Taiwan
104761501@nccu.edu.tw

Chuan-Ju Wang
Academia Sinica
Taipei, Taiwan
cjwang@citi.sinica.edu.tw

Ming-Feng Tsai†
National Chengchi University
Taipei, Taiwan
mftsai@nccu.edu.tw

Yi-Hsuan Yang
Academia Sinica
Taipei, Taiwan
yang@citi.sinica.edu.tw

ABSTRACT

We present collaborative similarity embedding (CSE), a unified framework that exploits comprehensive collaborative relations available in a user-item bipartite graph for representation learning and recommendation. In the proposed framework, we differentiate two types of proximity relations: direct proximity and k-th order neighborhood proximity. While learning from the former exploits direct user-item associations observable from the graph, learning from the latter makes use of implicit associations such as user-user similarities and item-item similarities, which can provide valuable information especially when the graph is sparse. Moreover, for improving scalability and flexibility, we propose a sampling technique that is specifically designed to capture the two types of proximity relations. Extensive experiments on eight benchmark datasets show that CSE yields significantly better performance than state-of-the-art recommendation methods.

ACM Reference Format:
Chih-Ming Chen, Chuan-Ju Wang, Ming-Feng Tsai, and Yi-Hsuan Yang. 2019. Collaborative Similarity Embedding for Recommender Systems. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313493

1 INTRODUCTION

Collaborative filtering (CF) is a common yet powerful approach for generating user recommendations that takes the advantage of the collective wisdom from all users [10]. Many CF-based recommendation algorithms have been shown to work well across various domains and been used in many real-world applications [21]. The core idea of model-based CF algorithms is to learn low-dimensional representations of users and items from either explicit user-item associations such as user-item ratings or implicit feedback such as playcounts and dwell time. This can be done by training a rating-based model with matrix completion to learn from observed user-item associations (either explicit or implicit feedback) to predict associations that are unobserved [1, 4, 8, 9, 11–15, 20, 28]. In addition to this rating-based approach, ranking-based methods have been proposed based on optimizing ranking loss; the ranking-based methods [15, 19, 25–27] have been found more suitable for implicit feedback. However, many existing model-based CF algorithms leverage only the user-item associations available in a given user-item bipartite graph. Thus, when the available user-item associations are sparse, these algorithms may not work well.

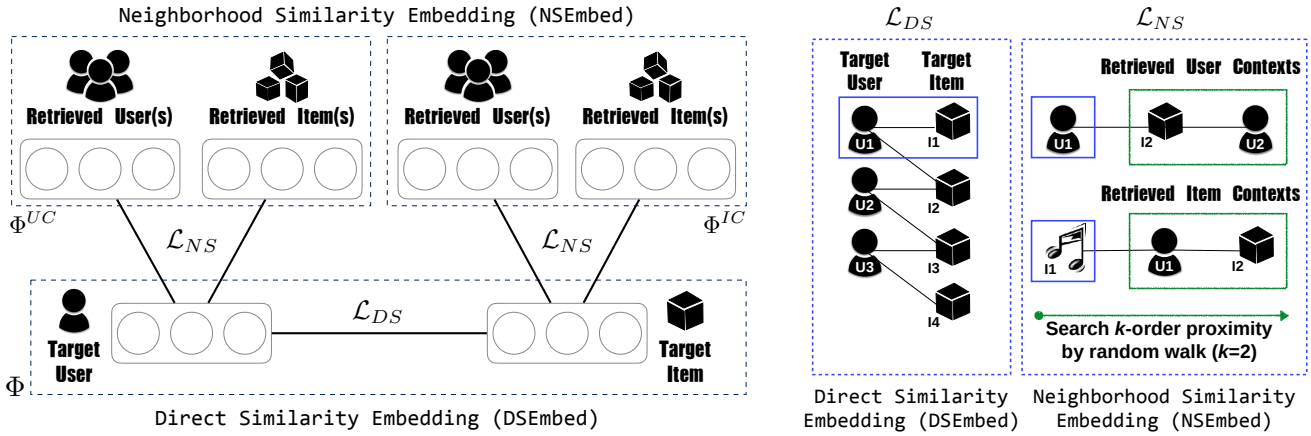
It has been noted that it is possible to mine from a user-item bipartite graph other types of collaborative relations, such as user-user and item-item similarities, since users and items can be indirectly connected in the graph. By taking random walks on the graph, it also exploits higher-order proximity among users and items. Using item-item similarities in the learning process has been firstly studied by Liang *et al.* [13], who propose to jointly decompose the user-item interaction matrix and the item-item co-occurrence matrix with shared item latent factors. Hsieh *et al.* [8] propose to learn a joint metric space to encode both user preferences and user-user and item-item similarities. A recent work presented by Yu *et al.* [27] shows that jointly modeling user-item, user-user, and item-item relations outperforms competing methods that consider only user-item relations. In [6, 16], the higher-order proximity has been shown useful in graph embedding methods. In general, exploiting additional collaborative relations shows promise in learning better representations of vertexes in an information graph.

We note that these prior arts [6, 8, 13, 27] share the same core idea: using some specific methods to sample auxiliary information from a graph to augment the data for representation learning. However, there is a lack of a unified and efficient model that generalizes the underlying computation and aims at recommendation problems. For example, Liang *et al.* [13] consider only the item-item similarities but no other collaborative relations; Yu *et al.* [27] consider only ranking-based loss functions but not rating-based ones. Higher-order proximity is exploited in [5, 6, 16], which however deal with the general graph embedding problem not the recommendation one. Moreover, the model presented by [8] fails to manage large-scale user-item associations [23].

To address this discrepancy, we present *collaborative similarity embedding* (CSE), a unified representation learning framework for collaborative recommender systems with the aim of modeling

*Social Networks and Human-Centered Computing, Taiwan International Graduate Program, Institute of Information Science, Academia Sinica, Taiwan
†MOST Joint Research Center for AI Technology and All Vista Healthcare, Taiwan

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313493



(a) The bottom part depicts the direct similarity embedding module for user-item associations, whereas the upper left (right) part corresponds to modeling user-user (item-item, respectively) similarity with the neighborhood similarity embedding module. An optimization step includes a sampled pair of a user and an item in DSEmbed and multiple high-order relation pairs in NSEmbed.

(b) The left part shows that a target user-item pair (U1-I1) can be directly sampled from the observed edges for DSEmbed; the right part shows that for U1 (or I1), a 2-step random walk is applied to obtain the contexts used in the NSEmbed module.

Figure 1: An overview of the proposed CSE framework

the direct and in-direct edges of user-item interactions in a simple and effective way. CSE involves a *direct similarity* embedding module for modeling user-item associations as well as a *neighborhood similarity* embedding module for modeling user-user and item-item similarities. The former module provides the flexibility to implement various types of modeling techniques for user-item associations, whereas the later module models user-user and item-item relations via k-order neighborhood proximity. To simultaneously manage the two modules, we introduce *triplet embedding* into the proposed framework to ideally model user-user, item-item clustering and user-item relations in a single and joint-learning model, while most prior arts use only one or two embedding mappings in their methods. Moreover, the two sub-modules are fused by a carefully designed sampling technique for scalability and flexibility. The space complexity and time of convergence are both only linear with respect to the number of observed user-item associations. In addition, with the proposed sampling techniques, CSE provides the flexibility to shape different relation distributions in its optimization.

Extensive experiments were conducted on eight recommendation datasets. We compare the performance of CSE with classic methods such as matrix factorization (MF) and Bayesian personalized ranking (BPR) [19], recent methods that incorporate user-user and/or item-item relations [8, 13, 27], as well as several general graph embedding methods [5, 16]. The evaluation shows that CSE outperforms the competing methods for seven out of the eight datasets. Our framework advances the state-of-the-art recommendation algorithms along the following four dimensions.

(1) The CSE serves as a generalized framework that models comprehensive pairwise relations among users and items with a unified function in a simple and effective manner.
(2) The proposed sampling technique enables the suitability of CSE for large-scale user-item recommendations.
(3) We report extensive experiments over eight recommendation datasets covering different user-item interaction types, levels of data sparsity, and data sizes, demonstrating the robustness, efficiency, and effectiveness of our framework.
(4) For reproducibility, we share the source code of CSE online at a GitHub repo,[1] by which the learning process can be done within an hour for each dataset performed in this work.

2 PROPOSED CSE FRAMEWORK

Problem Formulation. A recommender system provides a list of ranked items to users based on their historical interactions with items. Let U and I denote the sets of users and items, respectively. User-item associations can be presented as a bipartite graph $G = (V, E)$, where $V = \{v_1, \ldots, v_{|V|}\} = U \cup I$, and E represents the set of observed user-item associations. Note that for explicit rating data, the weights of the user-item preference edges can be positive real numbers, whereas for implicit interactions, the bipartite graph becomes a binary graph. The goal of the CSE framework is to obtain an embedding matrix $\Phi \in \mathbb{R}^{|V| \times d}$ that maps each user and item into a d-dimensional embedding vector for item recommendation; that is, with the learned embedding matrix Φ, for a user $v_i \in U$, the proposed framework generates the top-N recommended items via computing the similarity between the embedding vector of the

[1] https://github.com/cnclabs/proNet-core

user, i.e., Φ_{v_i}, and those of all items, i.e., Φ_{v_j} for all $v_j \in I$, where Φ_{v_x} denotes the row vector for vertex $v_x \in V$ from matrix Φ.

Framework Overview. Figure 1 provides an overview of CSE. In the figure, CSE consists of two similarity embedding modules: a direct similarity embedding (DSEmbed) module to model user-item associations, and a neighborhood similarity embedding (NSEmbed) module to model user-user and item-item similarities. The DSEmbed model provides the flexibility to implement two mainstream types of modeling techniques: rating-based and ranking-based models to preserve direct proximity of user-item associations; NSEmbed, in turn, models user-user and item-item relations using the contexts within a k-step random walk, as shown in Fig. 1(b), to preserve k-order neighborhood proximity between users and items. To minimize the sum of the losses from DSEmbed and NSEmbed modules, which are denoted as $\mathcal{L}_{DS}$ and $\mathcal{L}_{NS}$ respectively, the objective function of the proposed framework is designed as

$$\mathcal{L} = \mathcal{L}_{DS} + \lambda \mathcal{L}_{NS},$$

where λ controls the balance between the two losses. The rationale behind this design is that $\mathcal{L}_{DS}$ controls the optimization of the embedding vectors towards preserving direct user-item associations, and $\mathcal{L}_{NS}$ encourages users/items sharing similar neighbors to be close to one another in the learned embedding space.

2.1 Direct Similarity Embedding (DSEmbed) Module

Definition 2.1. **(Direct Proximity)** Given a bipartite graph $G = (V, E)$, the *direct proximity* between a user $v_i \in U$ and an item $v_j \in I$ is represented by the presence of an edge $(v_i, v_j) \in E$ between these two vertices. If there is no edge between user v_i and item v_j, then their direct proximity is defined as 0.

The DSEmbed module is designed to model the direct proximity of the user-item associations defined in Definition 2.1. For a rating-based approach, the objective is to find the embedding matrix Φ that maximizes the log-likelihood function of observed user-item pairs:

$$\begin{aligned} &\arg\max_{\Phi} \sum_{(v_i, v_j) \in E} \log p(v_i, v_j|\Phi) \\ &= \arg\min_{\Phi} \sum_{(v_i, v_j) \in E} -\log p(v_i, v_j|\Phi). \end{aligned} \tag{1}$$

In contrast, a ranking-based approach cares more about whether we can predict stronger association between a 'positive' user-item pair $(v_i, v_j) \in E$ than a 'negative' user-item pair $(v_i, v_k) \in \bar{E}$ [19], where $\bar{E}$ denotes the set of edges for all the unobserved user-item associations. This can be approached by maximizing the log-likelihood function of observed user-item pairs over unobserved user-item pairs for each user:

$$\begin{aligned} &\arg\max_{\Phi} \sum_{(v_i, v_j, v_k)} \log p(v_j >_i v_k|\Phi) \\ &= \arg\min_{\Phi} \sum_{(v_i, v_j, v_k)} -\log p(v_j >_i v_k|\Phi), \end{aligned} \tag{2}$$

where $v_i \in U$ and $v_j, v_k \in I$, and $>_i$ indicates that user v_i prefers item v_j over item v_k. In the above two equations, $p(v_i, v_j|\Phi)$ and $p(v_j >_i v_k|\Phi)$ is calculated by

$$p(v_i, v_j|\Phi) = \sigma\left(\Phi_{v_i} \cdot \Phi_{v_j}\right),$$

and

$$p(v_j >_i v_k|\Phi) = \sigma\left(\Phi_{v_i} \cdot \Phi_{v_j} - \Phi_{v_i} \cdot \Phi_{v_k}\right),$$

respectively, and $\sigma(\cdot)$ denotes the sigmoid function.

2.2 Neighborhood Similarity Embedding (NSEmbed) Module

Definition 2.2. **(k-Order Neighborhood Proximity)** Given a bipartite graph $G = (V, E)$ representing the observed user-item associations of the set of users and items in $V = U \cup I$, the *k-order neighborhood proximity* of a pair of users (or items) is defined as the similarity between their neighborhood network structures retrieved by k-step random walks. Mathematically speaking, given the k-order neighborhood structures of a pair of users (or items), $v_i, v_j \in U$ (or $v_i, v_j \in I$, respectively), which are denoted as two sets of neighbor nodes N_{v_i} and N_{v_j}, with $|N_{v_i}| = |N_{v_j}| = k$, the k-order neighborhood proximity between v_i and v_j is decided by the similarity between these two sets N_{v_i} and N_{v_j}. If there are no shared neighbors between v_i and v_j, the neighborhood proximity between them is 0.

The NSEmbed module is designed to model k-order neighborhood proximity for capturing user-user and item-item similarities. Given a set of neighborhood relations for users (or items) $S_U = \{(v_i, v_j)|\ \forall v_i \in U, v_j \in N_{v_i}\}$ (or $S_I = \{(v_i, v_j)|\ \forall v_i \in I, v_j \in N_{v_i}\}$, respectively), the NSEmbed module seeks a set of embedding matrices $\Phi, \Phi^{UC}, \Phi^{IC} \in \mathbb{R}^{|V| \times d}$ that maximizes the likelihood of all pairs in S_U (or S_I, respectively), where Φ is a *vertex mapping matrix* akin to that used in the DSEmbed module, and Φ^{UC} and Φ^{IC} are two *context mapping matrices*. Note that each vertex (representing a user or an item) plays two roles for modeling the neighborhood proximity: 1) the vertex itself and 2) the context of other vertices [2, 6, 16, 22, 29]. With this design, the embedding vectors of vertices that share similar contexts are thus closely located in the learned vector space. Therefore, the maximization of the likelihood function can be defined as

$$\begin{aligned} \arg\max_{\Phi, \Phi^{UC}, \Phi^{IC}} \quad & \textstyle\prod_{(v_i, v_j) \in S_U} p(v_j|v_i; \Phi; \Phi^{UC}) \\ & + \textstyle\prod_{(v_i, v_j) \in S_I} p(v_j|v_i; \Phi; \Phi^{IC}). \end{aligned}$$

Similar to Eqs. (1) and (2), the above objective function becomes

$$\begin{aligned} \arg\min_{\Phi, \Phi^{UC}, \Phi^{IC}} \quad & \textstyle\sum_{(v_i, v_j) \in S_U} -\log p(v_j|v_i; \Phi; \Phi^{UC}) \\ & + \textstyle\sum_{(v_i, v_j) \in S_I} -\log p(v_j|v_i; \Phi; \Phi^{IC}), \end{aligned} \tag{3}$$

where

$$p(v_j|v_i; \Theta) = \begin{cases} \sigma\left(\Phi_{v_i} \cdot \Phi^{UC}_{v_j}\right) & \text{if } v_i \in U, \\ \sigma\left(\Phi_{v_i} \cdot \Phi^{IC}_{v_j}\right) & \text{if } v_i \in I. \end{cases} \tag{4}$$

It is worth mentioning that most prior arts use only one or two embedding mappings; while the former approach fails to consider high-order neighbors (e.g., [3, 7, 12, 22, 24]), the later one cannot model user-user, item-item, and user-item relations simultaneously (e.g., [1, 2, 6, 16, 17, 29]). Our designed triplet embedding solution

(i.e., $\Theta = \{\Phi, \Phi^{UC}, \Phi^{UC}\}$) can ideally model user-user, item-item clustering and user-item relations in a joint-learning model.

2.3 Sampling-based Expectation Loss

In order to minimize the above objective functions, we need to go through all the pairs in E for Eq. (1), E and $\bar{E}$ for Eq. (2), and S_U and S_I for Eq. (3), to compute all the pairwise losses. This is not feasible in real-world recommendation scenarios as the complexity is $O(|V| \times |V|)$. To address this, we propose a sampling technique to work in tandem with the above two modules to enhance CSE's scalability and flexibility in learning user and item representations from large-scale datasets.

In CSE, the DSEmbed and NSEmbed modules are fused with the shareable data sampling technique described below. For each parameter update, we first sample an observed user-item pair $(v_i, v_j) \in E$, as shown as U1 and I1 in Fig. 1(b), where $v_i \in U$ and $v_j \in I$. Then, we search for the k-order neighborhood structures of user v_i and item v_j via the k-step random walks. To improve computational efficiency, we use negative sampling [22]. Consequently, for a rating-based approach (see Eq. (1)), the expected sampled loss of the DSEmbed module can be re-written as

$$\mathcal{L}_{DS} = \mathbb{E}_{(v_i,v_j)\sim E}\left[-\log p(v_i, v_j|\Phi)\right] + \sum_{M} \mathbb{E}_{(v_k,v_h)\sim \bar{E}}\left[\log p(v_k, v_h|\Phi)\right], \quad (5)$$

where M denotes the number of negative pairs adopted. For a ranking-based approach (see Eq. (2)), the DSEmbed module can be re-written as

$$\mathcal{L}_{DS} = \mathbb{E}_{(v_i,v_k)\sim \bar{E}}\left[\mathbb{E}_{(v_i,v_j)\sim E}\left[-\log p(v_j >_i v_k|\Phi)\right] | v_i\right]. \quad (6)$$

Note that for the ranking-based approach, there is no need to explicitly include M negative sample pairs as this kind of method naturally involves negative pairs from $\bar{E}$. Similarly, given a user or an item vertex v_i, its k-order neighborhood structure N_{v_i} is composed of nodes in the k-step random walks surfing on G, $\mathcal{W}_{v_i} = (\mathcal{W}^0_{v_i}, \mathcal{W}^1_{v_i}, \mathcal{W}^2_{v_i}, \ldots, \mathcal{W}^k_{v_i})$, where the vertex for $\mathcal{W}^j_{v_i}$ is randomly chosen from the neighbors of the vertex v given $\mathcal{W}^{j-1}_{v_i} = v$ and $\mathcal{W}^0_{v_i} = v_i$. The expected sampled loss of the NSEmbed module can be re-written as

$$\begin{aligned}\mathcal{L}_{NS} = {} & \mathbb{E}_{(v_i,v_j)\sim S_U}\left[-\log p(v_j|v_i;\Phi;\Phi^{UC})\right] + \sum_{M}\mathbb{E}_{(v_i,v_j)\sim \bar{E}}\left[\log p(v_j|v_i;\Phi;\Phi^{UC})\right] + \\ & \mathbb{E}_{(v_i,v_j)\sim S_I}\left[-\log p(v_j|v_i;\Phi;\Phi^{IC})\right] + \sum_{M}\mathbb{E}_{(v_i,v_j)\sim \bar{E}}\left[\log p(v_j|v_i;\Phi;\Phi^{IC})\right]. \end{aligned} \quad (7)$$

Since $\mathcal{L}_{DS}$ and $\mathcal{L}_{NS}$ are described in a sampling-based expectation form, CSE provides the flexibility for accommodating arbitrary distributions of positive and negative data. In the following experiments, we produce the positive data according to primitive edges distribution of given user-item graph. As to negative sampling, we propose to directly sample the negative data from whole data collection instead of unobserved data collection.

Dataset	#Users	#Items	#Edges	Edge type
Frappe	957	4,028	96,202	click count
CiteULike	5,551	16,980	210,504	like/dislike
Netflix	65,533	17,759	25,120,129	5-star
MovieLens-Latest	259,137	40,110	24,404,096	5-star
Last.fm-360K	359,347	294,015	17,559,530	play count
Amazon-Book	603,668	367,982	8,898,041	5-star
Epinions-Extend	755,760	120,492	13,668,319	5-star
Echonest	1,019,318	384,546	48,373,586	play count

Table 1: Statistics of the datasets considered in our experiments.

2.4 Optimization

In the optimization stage, we use asynchronous stochastic gradient descent (ASGD) [18] to efficiently update the parameters in parallel. The model parameters are composed of the three embedding matrices $\Theta = \{\Phi, \Phi^{UC}, \Phi^{IC}\}$, each having the size $O(|V|d)$. They are updated with learning rate α according to

$$\Theta \leftarrow \Theta - \alpha\left(\frac{\partial \mathcal{L}_{DS}}{\partial \Theta} + \lambda\left(\frac{\partial \mathcal{L}_{NS}}{\partial \Theta}\right) - \lambda_V \|\Phi\|\right), \quad (8)$$

where λ_V is a hyper-parameter for reducing the risk of overfitting.

3 EXPERIMENT

3.1 Settings

3.1.1 Datasets and Preprocessing. To examine the capability and scalability of the CSE framework, we conducted experiments on eight publicly available real-world datasets that vary in terms of domain, size, and density, as shown in Table 1. For each of the datasets, we discarded the users who have less than ten associated interactions with items. In addition, we converted each data into implicit feedback:[2] 1) for 5-star rating datasets, we transformed ratings higher than or equal to 3.5 to 1 and the rest to 0; 2) for count-based datasets, we transformed counts higher than or equal to 3 to 1 and the rest to 0; 3) for the CiteULike dataset, no transformation was conducted as it is already a binary preference dataset.

3.1.2 Baseline Algorithms. We compare the performance of our model with the following eight baseline methods: 1) POP, a naive popularity model that ranks the items by their degrees, 2) DeepWalk [16], a classic algorithm of network embedding, 3) WALS [9], a weighted rating-based factorization model, 4) ranking-based factorization models: BPR [19], WARP [25], and K-OS [26], 5) BiNE [5], a network embedding model specialized for bipartite networks, and 6) recent advanced models considering user-user/item-item relations: coFactor [13], CML [8] and WalkRanker [27]. Note that except for POP, the embedding vectors for users and items learned by these competitors as well as by our method can be directly used for item recommendations. Additionally, while CML adopts Euclidean distance as the scoring function, all other methods including ours utilize the dot product to calculate the score of a pair of user-item embedding vectors. The experiments for WALS and

[2]Note that in real-world scenarios, most feedback is not explicit but implicit [19]; we here converted the datasets into implicit feedback as most of the recent developed methods focus on dealing with such type of data. However, our method is not limited to binary preference since the presented sampling technique has the flexibility to manage arbitrary weighted edge distributions and rating estimation is also allowed in the proposed RATE-CSE.

	Frappe		CiteULike		Netflix		MovieLens-Latest	
	Recall@10	mAP@10	Recall@10	mAP@10	Recall@10	mAP@10	Recall@10	mAP@10
Pop	0.1750	0.0708	0.0270	0.0114	0.0861	0.0359	0.0882	0.0289
DeepWalk [16]	0.0430	0.0256	0.0875	0.0458	0.0235	0.0112	0.0207	0.0061
WALS [9]	0.1632	0.1117	0.1851	0.0915	0.1214	0.0471	0.2350	†0.1682
BPR [19]	0.2785	0.1550	0.0861	0.0426	0.1496	0.0757	0.2163	0.1130
WARP [25]	0.3012	0.1796	0.1468	0.0813	†0.1887	†0.1004	†0.2712	0.1651
K-OS [26]	0.3018	0.1914	0.1356	0.0756	0.1783	0.0868	0.2522	0.1641
BiNE [5]	0.2159	0.1201	0.0422	0.0201	-	-	-	-
coFactor [13]	0.2110	0.1309	0.1323	0.0721	-	-	-	-
CML [8]	†0.3311	0.1958	0.1740	0.1008	0.1035	0.0444	0.1109	0.0957
WalkRanker [27]	0.3286	†**0.2099**	†0.2059	†0.1192	0.1090	0.0483	0.1307	0.0351
RATE-CSE	**0.3347**	0.2047	*__0.2362__	*__0.1452__	*0.2014	*0.1039	*__0.3225__	*__0.1990__
Improv. (%)	+1.0%	−2.4%	+14.7%	+21.9%	+6.7%	+3.5%	+18.9%	+18.3%
RANK-CSE	0.3155	0.2005	0.1993	*0.1228	*__0.2156__	*__0.1202__	*0.3094	*0.1902
Improv. (%)	−4.7%	−4.4%	−3.2%	+3.0%	+14.2%	+19.7%	+14.1%	+13.1%

	Last.fm-360K		Amazon-Book		Epinions-Extend		Echonest	
	Recall@10	mAP@10	Recall@10	mAP@10	Recall@10	mAP@10	Recall@10	mAP@10
Pop	0.0309	0.0133	0.0053	0.0015	0.0450	0.0246	0.0257	0.0104
WALS [9]	0.1621	0.0857	†0.0540	†0.0227	0.1479	0.0634	0.1287	†0.0638
BPR [19]	0.1120	0.0545	0.0248	0.0119	0.1126	0.0579	0.0499	0.0210
WARP [25]	0.1556	0.0832	0.0457	0.0199	†0.1509	†0.0775	0.1001	0.0447
K-OS [26]	†0.1641	†0.0888	0.0511	0.0215	0.1493	0.0766	†0.1249	0.0597
CML [8]	0.0496	0.0199	0.0129	0.0052	0.1171	0.0629	0.0357	0.0195
WalkRanker [27]	0.0233	0.0088	0.0080	0.0036	0.0560	0.0289	0.0309	0.0133
RATE-CSE	*0.1687	*0.0909	0.0540	0.0240	*0.1659	0.0788	0.1260	0.0605
Improv. (%)	+2.8%	+2.3%	+0.0%	+5.7%	+9.9%	+1.7%	+0.8%	−0.5%
RANK-CSE	*__0.1762__	*__0.0970__	*__0.0625__	*__0.0274__	*__0.1767__	*__0.0921__	*__0.1358__	*__0.0679__
Improv. (%)	+8.2%	+14.4%	+15.7%	+20.7%	+17.0%	+20.2%	+8.7%	+6.4%

Table 2: Recommendation performance. The † symbol indicates the best performing method among all the baseline methods; '*' and '%Improv.' denote statistical significance at $p < 0.01$ with a paired t-test and the percentage improvement of the proposed method, respectively, with respect to the best performing baseline.

BPR were conducted using the matrix factorization library QMF,[3] and those for WARP and K-OS were conducted using LightFM;[4] for coFactor, CML, and WalkRanker, we used the code provided by the respective authors.

3.1.3 Experimental Setup. For all the experiments, the dimension of embedding vectors was fixed to 100; the values of the hyper-parameters for the compared method were decided via implementing a grid search over different settings, and the combination that leads to the best performance was picked.

For our model, the learning rate α was set to 0.1, λ_V was set to 0.025; the hyper-parameter λ was set to 0.05 and 0.1 for rating-based CSE and ranking-based CSE, respectively, and k was set to 2 as the default value. The sensitivity of CSE parameters are additionally reported. For each dataset, the sample time for convergence depends on the number of non-zero user-item interaction edges and is set to $80 \times |E|$. Sensitivity analysis for k and λ and convergence analysis are later provided in the section for convergence analyses.

[3]https://github.com/quora/qmf
[4]https://github.com/lyst/lightfm

3.2 Results

3.2.1 Recommendation Performance Comparison. The results for the ten baseline methods along with the proposed method are listed in Table 2, where RATE-CSE and RANK-CSE denote two versions of our method that employ respectively rating-based and ranking-based loss functions for user-item associations. Note that the best results are always indicated by the bold font, and for coFactor and BiNE we report only part of the experimental results on Frappe and CiteULike because of resource limitations.[5] DeepWalk is not suitable for user-item recommendation as it make the users apart from items in the embedding space. In addition, observe that BiNE does not perform well in our experiments; such a result is due to the fact that BiNE is a general network embedding model and thus does not incorporate the regularizer in their objective function, which is however an important factor for the robustness of recommendation performance. Comparing the performance of the other baseline methods, we observe that the performance of WALS, WARP

[5]While the memory usage of coFactor implementation is $O(|V|^2)$, BiNE's requires extensive computational time, e.g., more than 24 hours to learn the embedding for the large dataset, Movielens-Latest.

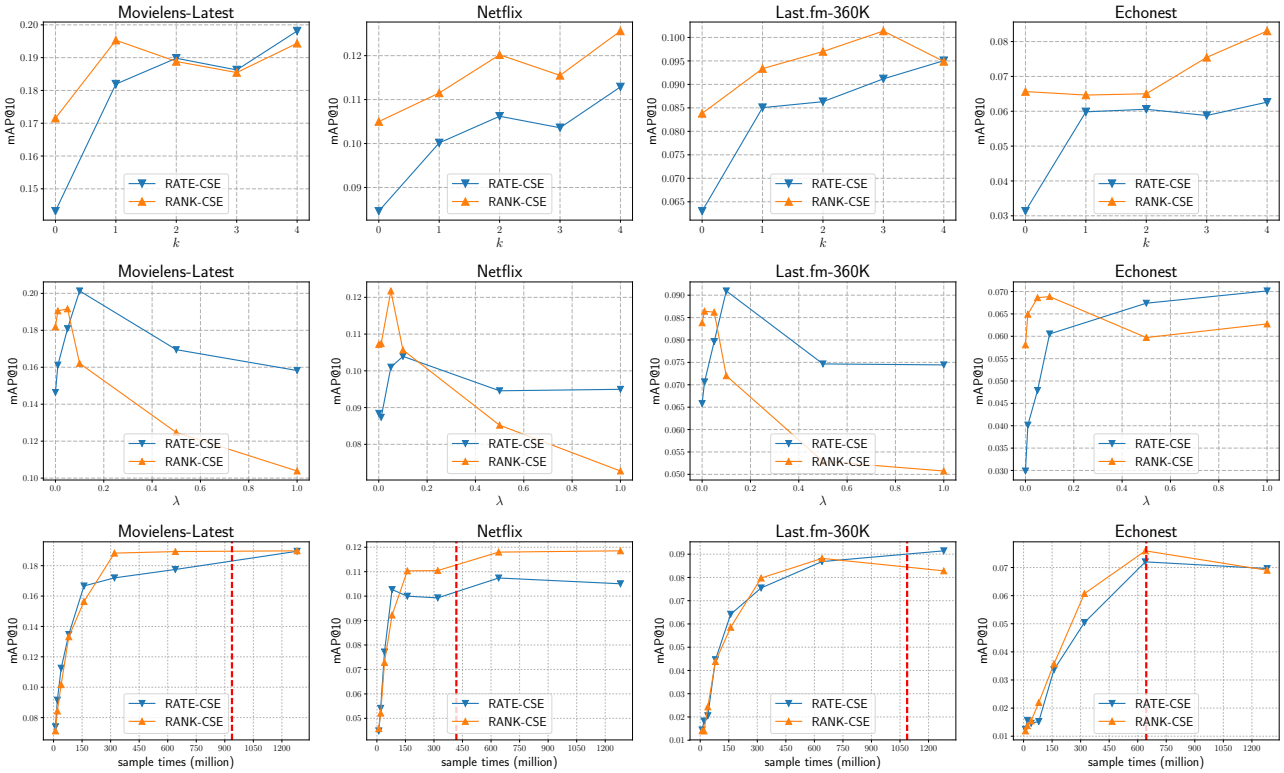


Figure 2: Sensitivity and convergence analyses

and K-OS is very competitive. That is, these methods achieve the top performance among all the baselines on several datasets. The performance of WalkRanker and CML, on the other hand, seems satisfactory only on two rather small datasets.

We observe that our method achieves the best results in terms of both Recall@10 and mAP@10 for most datasets. Moreover, RANK-CSE generally outperforms RATE-CSE in the experiments, reconfirming that using a ranking-based loss is indeed better for datasets with binary implicit feedbacks [19, 25, 26]. Specifically, except for Frappe, RATE-CSE or RANK-CSE achieves significantly much better performance than the best performing baseline methods.

3.2.2 Parameter Sensitivity and Convergence Analyses. Figure 2 shows the sensitivity tests on two hyper-parameters k and λ in the first and second rows, respectively, and those of the convergence analysis based on sample times in the third row.[6] We first observe that increasing the order k of modeling neighborhood proximity between users or items improves the performance in general. The optimal value of k is data dependent and has to be empirically tuned considering the trade-off between accuracy and time/space complexity. In general, a larger k leads to better result, and from our experience, the result would reach a plateau when k is sufficiently large (e.g., when $k > 3$). The second row of Figure 2 shows how the balancing parameter λ affects performance: RANK-CSE obtains better performance with a value around 0.05, while RATE-CSE performs well with a value around 0.1. Finally, we empirically show that the required total sample times for convergence is linear with respect to $|E|$ as illustrated in the third row, where the vertical dash line indicates the boundary of $|E| \times 80$ as we applied to the previous recommendation experiment. As the training time depends linearly on the sample times, it can be said that both RATE-CSE and RANK-CSE converge with less than a constant multiple of $|E|$ sample times. This demonstrates the nice scalability of CSE.

[6]Note that due to space limits, we report the results for the four largest datasets only.

4 CONCLUSION

We present CSE, a unified representation learning framework that exploits comprehensive collaborative relations available in a user-item bipartite graph for recommender systems. Two types of proximity relations are modeled by the proposed DSEmbed and NSEmbed modules. Moreover, we propose a sampling technique to enhance the scalability and flexibility of the model. Experimental results show that CSE yields superior recommendation performance over a wide aange of datasets with different sizes, densities, and types than many state-of-the-art recommendation methods.

ACKNOWLEDGMENTS

This research was partially supported by the Ministry of Science and Technology of Taiwan under the grant MOST 107-2218-E-002-061.

REFERENCES

[1] Oren Barkan and Noam Koenigstein. 2016. Item2Vec: Neural item embedding for collaborative filtering. In *Workshop IEEE MLSP*.

[2] Chih-Ming Chen, Ming-Feng Tsai, Yu-Ching Lin, and Yi-Hsuan Yang. 2016. Query-based Music Recommendations via Preference Embedding. In *Proc. ACM RecSys.*

[3] Wei-Sheng Chin, Bo-Wen Yuan, Meng-Yuan Yang, Yong Zhuang, Yu-Chin Juan, and Chih-Jen Lin. 2016. LIBMF: A Library for Parallel Matrix Factorization in Shared-memory Systems. *Journal of Machine Learning Research* (2016).

[4] Evangelia Christakopoulou and George Karypis. 2016. Local Item-Item Models For Top-N Recommendation. In *Proc. ACM RecSys.*

[5] Ming Gao, Leihui Chen, Xiangnan He, and Aoying Zhou. 2018. BiNE: Bipartite Network Embedding. In *Proc. ACM SIGIR.*

[6] Aditya Grover and Jure Leskovec. 2016. Node2Vec: Scalable Feature Learning for Networks. In *Proc. ACM SIGKDD.*

[7] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. 2015. TrustSVD: Collaborative Filtering with Both the Explicit and Implicit Influence of User Trust and of Item Ratings. In *Proc. AAAI.*

[8] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative Metric Learning. In *Proc. WWW.*

[9] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proc. IEEE ICDM.*

[10] Dietmar Jannach, Paul Resnick, Alexander Tuzhilin, and Markus Zanker. 2016. Recommender systemsâĂŤbeyond matrix completion. *Commun. ACM* (2016).

[11] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *Proc. ACM KDD.*

[12] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. (2009).

[13] Dawen Liang, Jaan Altosaar, Laurent Charlin, and David M. Blei. 2016. Factorization Meets the Item Embedding: Regularizing Matrix Factorization with Item Co-occurrence. In *Proc. ACM RecSys.*

[14] Xia Ning and George Karypis. 2011. SLIM: Sparse Linear Methods for Top-N Recommender Systems. In *Proc. IEEE ICDM.*

[15] Enrico Palumbo, Giuseppe Rizzo, and Raphaël Troncy. 2017. entity2rec: Learning User-Item Relatedness from Knowledge Graphs for Top-N Item Recommendation. In *Proc. ACM RecSys.*

[16] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proc. ACM SIGKDD.*

[17] Bryan Perozzi, Vivek Kulkarni, and Steven Skiena. 2016. Walklets: Multiscale Graph Embeddings for Interpretable Network Classification. *CoRR* (2016).

[18] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. 2011. HOGWILD!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent. In *Proc. NIPS.*

[19] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback]. In *Proc. UAI.*

[20] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *Proc. NIPS.*

[21] Xiaoyuan Su and Taghi M Khoshgoftaar. [n. d.]. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 ([n. d.]).

[22] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proc. WWW.*

[23] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent Relational Metric Learning via Memory-based Attention for Collaborative Ranking. In *Proc. WWW.*

[24] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In *Proc. ACM KDD.*

[25] Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. WSABIE: Scaling Up to Large Vocabulary Image Annotation. In *Proc. IJCAI.*

[26] Jason Weston, Hector Yee, and Ron J. Weiss. 2013. Learning to Rank Recommendations with the K-order Statistic Loss. In *Proc. ACM RecSys.*

[27] Lu Yu, Chuxu Zhang, Shichao Pei, Guolei Sun, and Xiangliang Zhang. 2018. WalkRanker: A Unified Pairwise Ranking Model with Multiple Relations for Item. In *Proc. AAAI.*

[28] Wayne Xin Zhao, Jin Huang, and Ji-Rong Wen. 2016. Learning Distributed Representations for Recommender Systems with a Network Embedding Approach.. In *AIRS.*

[29] Chang Zhou, Yuqiong Liu, Xiaofei Liu, Zhongyi Liu, and Jun Gao. 2017. Scalable Graph Embedding for Asymmetric Proximity. In *Proc. AAAI.*

Sampled in Pairs and Driven by Text: A New Graph Embedding Framework

Liheng Chen
Shanghai Jiao Tong University
lhchen@apex.sjtu.edu.cn

Yanru Qu
Shanghai Jiao Tong University
kevinqu@apex.sjtu.edu.cn

Zhenghui Wang
Shanghai Jiao Tong University
felixwzh@apex.sjtu.edu.cn

Weinan Zhang
Shanghai Jiao Tong University
wnzhang@apex.sjtu.edu.cn

Ken Chen
Synyi LLC.
chen.ken@synyi.com

Shaodian Zhang
Synyi LLC.
shaodian@apex.sjtu.edu.cn

Yong Yu
Shanghai Jiao Tong University
yyu@apex.sjtu.edu.cn

ABSTRACT

In graphs with rich texts, incorporating textual information with structural information would benefit constructing expressive graph embeddings. Among various graph embedding models, random walk (RW)-based is one of the most popular and successful groups. However, it is challenged by two issues when applied on graphs with rich texts: (i) *sampling efficiency*: deriving from the training objective of RW-based models (e.g., DeepWalk and node2vec), we show that RW-based models are likely to generate large amounts of redundant training samples due to three main drawbacks. (ii) *text utilization*: these models have difficulty in dealing with zero-shot scenarios where graph embedding models have to infer graph structures directly from texts. To solve these problems, we propose a novel framework, namely Text-driven Graph Embedding with Pairs Sampling (TGE-PS). TGE-PS uses Pairs Sampling (PS) to improve the sampling strategy of RW, being able to reduce ∼99% training samples while preserving competitive performance. TGE-PS uses Text-driven Graph Embedding (TGE), an inductive graph embedding approach, to generate node embeddings from texts. Since each node contains rich texts, TGE is able to generate high-quality embeddings and provide reasonable predictions on existence of links to unseen nodes. We evaluate TGE-PS on several real-world datasets, and experiment results demonstrate that TGE-PS produces state-of-the-art results on both traditional and zero-shot link prediction tasks.

CCS CONCEPTS

• **Computing methodologies** → *Learning latent representations*;
• **Information systems** → *Information retrieval.*

KEYWORDS

Graph Embedding, Data Mining, Link Prediction, Zero-shot

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313520

ACM Reference Format:
Liheng Chen, Yanru Qu, Zhenghui Wang, Weinan Zhang, Ken Chen, Shaodian Zhang, and Yong Yu. 2019. Sampled in Pairs and Driven by Text: A New Graph Embedding Framework. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313520

1 INTRODUCTION

Graph provides a fundamental tool to represent interconnected entities (e.g., articles, diseases) and their attributes (e.g., entity description). Graphs with rich text information are ubiquitous in many fields [18], and there is often a strong dependency between graph structure and text structure in these graphs. Textural information may also expose structural information [56]. Hence, it is promising to better utilize textual information in graphs.

Graph embedding is famous for its efficient representations for entities in graphs [16, 38]. A series of models are proposed to maximize edge reconstruction probability with different proximities [5], e.g., LINE [45], DeepWalk [42] and node2vec [17]. Although these models are widely used in mapping nodes to low-dimensional dense vectors, they are not well designed for graphs with rich text information. To solve this problem, models like DDRW [26], GENE [8], PPNE [25] and Tri-DNR [40] focus on preserving vertex labels. Especially, TADW [53], CANE [47] and Paper2Vec [14] are proposed to utilize textural information with effectiveness in many scenarios.

However, these graph embedding models still need to resolve two issues. The first issue is *sampling efficiency*. The optimization goals of these models can be summarized as maximizing pairwise node similarity, thus the number of training node pairs is critical to training time and can be used as a metric of sampling efficiency. These models usually use edges as training node pairs directly like LINE, or sample training sequences using random walk (RW) and extract node pairs within a specific shortest distance like DeepWalk and node2vec. RW generalizes the idea of directly sampling edges and shows better experimental performance under similar settings. However, our theoretical and empirical analyses show that RW samples redundant node pairs which severely lags efficiency.

The second issue is *text utilization*. Text utilization models like TADW and CANE mainly rely on training node embeddings (NE)

and text embeddings (TE) together (denoted as NE+TE). In zero-shot settings, where link existence with previously unseen nodes is to be predicted and the only reliable information is texts attributed to nodes , humans can make inference by assuming textual connections even if he has little prior knowledge about this field. However, NE+TE is incapable of dealing with this scenario due to two drawbacks: (i) it requires both NE and TE to generate high-quality representations, but NE is missing for unseen nodes, and (ii) even if it uses only TE part to predict, empirical results show that it may be no better than simple text matching. This reveals that NE+TE does not utilize text information sufficiently.

In this paper, we propose a novel Text-driven Graph Embedding with Pairs Sampling (TGE-PS) framework. TGE-PS uses Pairs Sampling (PS) to efficiently generate training samples, and uses Text-driven Graph Embedding (TGE) to produce final node representations. We theoretically analyze the redundant sample phenomenon of RW from three perspectives , and propose PS to improve sampling efficiency. PS samples center-neighbor node pairs directly from central node's neighborhood. Our experiment results on 6 datasets show that PS produces competitive or even better results in link prediction task with much fewer training samples (saving ~99% samples) compared with RW. In embedding stage, we propose inductive TGE method. TGE incorporates text information with structure information and encodes character- and word-level embeddings into node embeddings while following graph structure. Since node embeddings are generated from text embedding in TGE, they can be applied to zero-shot scenarios. The comparison between TGE-PS and other strong baseline models shows that TGE-PS produces remarkably good results in traditional and zero-shot link prediction tasks.

2 RELATED WORKS

Recent years have witnessed various graph embedding models with applications in link prediction[2, 27, 30], node classification[21, 44, 46, 54], clustering[3, 12], recommendation [50, 51, 55], knowledge graph [4], etc. These graph embedding models can be categorized into three classes, factorization-based, random walk (RW)-based, and deep learning-based[5, 16]. Factorization-based models focus on the connections among nodes and using matrix factorization (MF) to learn the low-rank representations of nodes [1, 6, 39, 53]. TADW [53] uses MF to decompose the transition matrix with textual information incorporated. RW-based models explore the neighborhood of each node through sampling paths, and thus can maintain local structural information in node embeddings. Among RW-based models, DeepWalk [42] and node2vec [17] are the most representative, and DeepWalk can be regarded as a special case of node2vec. Deep learning-based models mainly use deep representation learning techniques to improve the quality of node embeddings [7, 20, 47, 48].

There are also models that do not belong to these three classes. LINE [45] proposes to train graph embeddings through first-order and second-order proximities and is efficient in large-scale networks. Besides, LINE claims to preserve both local and global network structures. GraphGAN [49] adopts GAN [15] framework to learn the underlying true connectivity distribution implicitly. The generator produces "fake edges" while the discriminator tries to tell generated node pairs from ground truth. It also proposes Graph Softmax to boost its efficiency in training.

Among graph embedding models, RW-based models present robust and remarkable performance on various datasets. The training of RW-based models have two phases, the sampling phase and the optimization phase. Different sampling policies are proposed to generate high-quality node sequences to explore the neighborhoods of certain nodes. Especially, DeepWalk adopts the simplest policy where each node is generated only depending on its predecessor, while node2vec adopts a biased policy to trade off between Breadth-first Sampling (BFS) and Depth-first Sampling (DFS). After obtaining sufficient node sequences, RW-based models use Skip-Gram [33, 35] model to maximize the log-probability of observing a network neighborhood for a node conditioned on its feature representation. However, we concern the sampling strategies of RW-based models have intrinsic drawbacks, which we discuss in Section 3.2.2.

For graphs with rich texts, many models are proposed to incorporate textual information with structural information. TADW [53] incorporates text features under the framework of matrix factorization. CANE [47] uses convolutional neural networks and mutual attention mechanism to learn text embeddings, which are interacted with node embeddings via vector inner product. These models encode structural information and textual information into two separate embedding spaces and generate final node representations from the interactions between these two spaces. Paper2Vec [14] pre-trains node embeddings with text embeddings in Skip-gram model, and then the node embeddings are trained with node2vec. STNE [29] "self-translates" sequences of text embeddings into sequences of node embeddings. All the above models rely on known connections to generate graph embeddings, thus are not applicable to zero-shot scenarios. Although some previous works [13, 22, 52] discuss graph representation learning in zero-shot setting, they do not solve the same problem as we do.

When processing texts in zero-shot scenarios, it is common to encounter out-of-vocabulary words. In NLP field, character-level embeddings have proven success like Neural Machine Translation [9, 23, 28] and Named Entity Recognition [31, 43]. These models are based on a hybrid of character- and word-level embeddings, thus can automatically capture patterns in the sub-word level and present advantages in dealing with unseen words.

3 TEXT-DRIVEN GRAPH EMBEDDING WITH PAIRS SAMPLING FRAMEWORK

In this section, we firstly give definitions of notions, and then introduce PS and TGE separately. We show the architecture of TGE-PS in Fig. 1, with fully-connected and lookup layers omitted.

3.1 Definitions

Let $G = (V, E)$ be the given graph and $f : V \to \mathbb{R}^n$ be the mapping function from the node set to the n-dimensional embedding space. We denote the central and context embeddings of node v_i as $\mathbf{e}_i = f(v_i)$ and $\mathbf{e}'_i = f'(v_i)$, and the trainable parameters as θ. We define the distance $\mathcal{L}(u, v)$ between u and v as the length of the shortest path between them. And we define node v_i's o-neighborhood $\mathcal{N}_i^o$ as a set of nodes within a given distance o from v_i, $\mathcal{N}^o(v_i) = \{v_j \in$

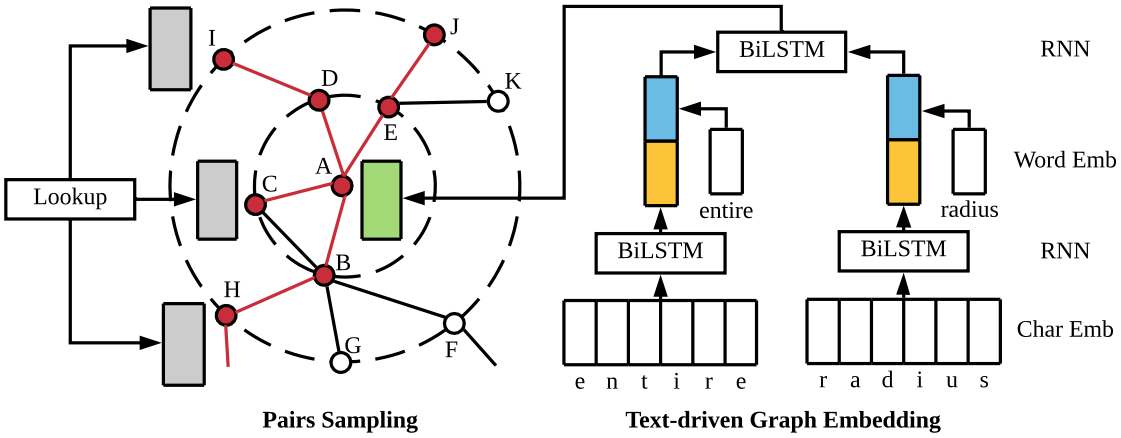


Figure 1: Model architecture of TGE-PS. *Note:* Circles represent nodes and colored boxes represent embeddings. Blue boxes represent word-level embeddings, yellow boxes represent character-based word embeddings, green boxes represent central embeddings and gray boxes represent neighbor embeddings. The sampled nodes and paths are colored as red.

$V|\text{dist}(v_i, v_j) \leq o\}$. For every node v_i, we call one node v_j^o as an o-th order neighbor of v_i when the distance between v_i and v_j is o.

3.2 Pairs Sampling Method

3.2.1 Revisit Random Walk. Since DeepWalk can be regarded as a special case of node2vec [17, 42], our discussion mainly focuses on node2vec. The objective of node2vec is to maximize the log-probability of observing the o-neighborhood $\mathcal{N}_i^o$ of a node v_i as in Eq. (1), and following conditional independence assumption of SkipGram, node2vec changes its objective to maximize the log-probability of all center-neighbor node pairs as in Eq. (2). Following symmetry in feature space assumption, the conditional probability of a node pair is defined as in Eq. (4), where $\tau_{i,j}$ is the abbreviation of the score function $\tau(v_i, v_j)$. It is worth noting that, the score function $\tau_{i,j}$ is asymmetric since the input v_j is conditioned on the other input v_i. With Eq. (2) and Eq. (4), the objective becomes Eq. (3), where Z_i denotes the normalizing term, and each window has size $2k$. Z_i is usually approximated by hierarchical softmax or negative sampling in training. And we mainly focus on the scoring terms $\sum_i \sum_j \tau_{i,j}$.

$$\max_\theta \sum_{v_i \in V} \log Pr(\mathcal{N}_i^o | v_i) \tag{1}$$

$$= \max_\theta \sum_{v_i \in V} \sum_{v_j \in \mathcal{N}_i^k} \log Pr(v_j | v_i) \tag{2}$$

$$= \max_\theta \sum_{v_i \in V} \Big(\sum_{v_j \in \mathcal{N}_i^k} \tau_{i,j} \Big) - |\mathcal{N}_i^k| \log Z_i \tag{3}$$

$$Pr(v_j | v_i) = \frac{\exp(\tau_{i,j})}{\sum_{v_k \in V} \exp(\tau_{i,k})} \tag{4}$$

From the training perspective, node2vec defines the inner product of a central embedding and a context embedding to represent the score function, i.e., $\mathbf{e}_j'^\top \mathbf{e}_i$. The training objective of the scoring terms becomes Eq. (5), where s denote a node sequence, S is the set of all sequences, ω_i^s is the abbreviation of the window function $\omega(v_i, s)$ which denotes the nodes within the window of v_i when v_i appears in s and j' denotes the position of v_j in s. We transform Eq. (5) to Eq. (6) and Eq. (7), where T denotes the sampling time of RW starting from v_i, T_i denotes the amount of sequences where v_i is not the starting point, $T_{j|i}$ denotes the amount of v_j appearing in N_i^k when v_i appears in sequence s. It is worth noting that, each node serves as the starting point in T sequences, and each node also appears in the sequences starting from the other nodes. Thus, there are $T + T_i$ windows centering at v_i to be optimized in the sequences. α_i is the ratio of v_i being more "important" than the other nodes, e.g., bridge nodes are likely to be sampled more frequently, and these nodes have larger α. $\beta_{j|i}$ is the probability of v_j sampled in the neighborhood $\mathcal{N}_i^o$ of v_i. Since there are $T + T_i$ windows centering at v_i, there are $2k(T + T_i)$ center-neighbor node pairs. Thus $2k(T + T_i) = \sum_{v_j \in N_i^k} T_{j|i}$, and $\beta_{\cdot|i}$ reflects the distribution of the neighbor nodes appearing in training samples. Till now, we get the ideal training objective of RW regardless of the sampling strategies. In another word, any RW policies will converge to Eq. (7) when the amount of sampled sequences approaches infinity.

$$\max_\theta \sum_{v_i \in V} \sum_{s \in S} \sum_{v_{j'} \in \omega_i^s} \mathbf{e}_{j'}'^\top \mathbf{e}_i \tag{5}$$

$$= \max_\theta \sum_{v_i \in V} (T + T_i) \Big(\sum_{v_j \in \mathcal{N}_i^k} \frac{T_{j|i}}{2k(T + T_i)} \mathbf{e}_{j'}'^\top \mathbf{e}_i \Big) \tag{6}$$

$$= \max_\theta T \sum_{v_i \in V} (1 + \alpha_i) \Big(\sum_{v_j \in \mathcal{N}_i^k} \beta_{j|i} \mathbf{e}_{j'}'^\top \mathbf{e}_i \Big) \tag{7}$$

3.2.2 Inefficiency of Random Walk. We introduce the sampling strategy of node2vec at first. For every node v_i in the graph, node2vec simulates a random walk with length L. During the sampling process, given previous node t and current node v, the next node x is sampled from the following distribution:

$$Pr(x|v) = \begin{cases} \frac{p_{vx}}{Z} & \text{if } (v, x) \in E \\ 0 & \text{otherwise} \end{cases}$$

where p_{vx} is the unnormalized transition probability from nodes v to x, and Z is the normalizing term. The unnormalized transition probability p_{vx} is set to be $\alpha(t, x; p, q) \cdot w_{vx}$, where $\alpha(t, x; p, q)$ is a function of $\text{dist}(t, x)$, w_{vx} denotes the edge weight of (v, x),

$\text{dist}(t, x)$ denotes the shortest path length between t and x, p and q control the searching strategy of random walks. Without loss of generality, we take $w_{vx} = 1$.

DeepWalk can be regarded as a special case of node2vec when $p = q = 1$. TADW has proved that DeepWalk is reconstructing the transition matrix M. Induced from this conclusion, we view node2vec as reconstructing a biased transition matrix M'. Thus in DeepWalk, $\beta_{j|i}$ in Eq. (7) are actually the non-zero elements of the i-th row of M in DeepWalk, or the i-th row of M' in node2vec. The sampling strategy of node2vec is an exploration of neighborhoods with second-order Markov Property. We analyze the inefficiency of RW in 3 aspects.

Biased Objective. Comparing Eq. (3) and (7), we find node2vec is biased from its log-probability objective since the neighborhoods N_i^k are weighted by different α_i. α_i can be regarded as an "importance ratio", given the previously mentioned bridge nodes example. Hence node2vec introduces a prior distribution implicitly, making its training objective biased from its proposal. If we follow Eq. (1) strictly, α should be all zero and we should only sample each neighborhood T time, which can reduce a lot of training samples.

Interconnected Neighbors. The target of node embedding is to encode structural information in an embedding space. It is a free lunch to assume when optimize a central node v_i, its neighbor nodes $v_j \in N_i^k$ have already been trained and the context embedding $\mathbf{e}'_j$ have already been encoded with the structural information of v_j respectively after training RW for some time, otherwise, the target of node embedding becomes ill-posed and can never be achieved. Thus center-neighbor connections are much more important than neighbor-neighbor connections, and the neighbor-neighbor connections within one neighborhood will become the center-neighbor connections of other neighborhoods. With these observations, we concern RW is less efficient since it results in lots of neighbor-neighbor connections when training central nodes. Therefore, adopting a stronger assumption that alleviates variance of distribution $\beta_{\cdot|i}$ is still possible to produce good results while decreases sampling complexity.

Revisiting Nodes. DeepWalk has the first-order Markov property since it only remembers the current node. node2vec is second-order Markov since it remembers the current node and the previous node. When the window size is larger than the Markov order, RW cannot prevent from revisiting a previously visited neighbor node. The worst case is revisiting the central node itself, because that introduces self-loops and doubles the training time of the central node. In ideal cases, revisiting is equivalent to approximate the transition matrix, where $\beta_{j|i} = M_{i,j}$, $\tau_{i,j} = \beta_{j|i}\mathbf{e}'^{\top}_j \mathbf{e}_i$. Since $\beta_{j|i}$ is a constant determined by the graph structure, the node embeddings $\mathbf{e}_i$ and $\mathbf{e}'_j$ are approximating $\tau_{i,j}/\beta_{j|i}$, where $\tau_{i,j}$ is the unormalized probability. The ideal sample complexity of a neighborhood N_i^k is determined by the minimal sampling probability $\beta_{j*|i}$, by assuming every node in the neighborhood N_i^k gets sufficient training samples. For simplicity, we say the sample complexity of N_i^k is $O(1/\beta_{j*|i})$. And this complexity must be greater than $O(|N_i^k|)$ since $\beta_{j*|i} \le 1/|N_i^k|$. This means any distribution β over N_i^k results in a higher sample complexity than uniform distribution $\beta_{\cdot|i} = 1/|N_i^k|$.

3.2.3 Method Introduction. The intuition of Pairs Sampling (PS) has three points: as for the **biased objective** problem, it is desirable to sample each neighborhood with same time; as for the **interconnected neighbors** problem, it is desirable to sample center-neighbor pairs directly from a neighborhood; as for the **revisiting nodes** problem, it is promising to introduce higher-order Markov property.

To obtain the training node pairs set $\mathcal{P}$, we sample the neighbor nodes in the neighborhood of v_i in the following process:

1. Add all first-order neighbor nodes of v_i into N_i^O.
2. If $O > 1$, for every v_j^o $(1 \le o \le O-1)$, sample a next-order neighbor node v_k^{o+1} from the following distribution:

$$Pr(v_k^{o+1}|v_j^o) = \begin{cases} \frac{1}{Z_j} & \text{if } (v_j^o, v_k^{o+1}) \in E \text{ and } \mathcal{L}(v_i, v_k^{o+1}) = o+1 \\ 0 & \text{otherwise.} \end{cases}$$

   where Z_j is the number of $o+1$-order neighbors connected to v_j^o. The sampled v_k^{o+1} is added into N_i^O.
3. For each neighbor node v_j^o, add node pair (v_i, v_j^o) into pairs set $\mathcal{P}$.
4. Repeat for N times.

We take the graph in Fig. 1 as an example to illustrate how PS works. In this graph, A is the central node, with dashed circles indicating the neighbors of the same order. We color sampled neighbors as red circles and leave the others as white ones. Following the aforementioned process, first-order neighbors $\{B, C, D, E\}$ are all sampled. H is sampled from $\{F, G, H\}$ as the successor of B. I is sampled as the successor of D. J is sampled from $\{J, K\}$ as the successor of E. Unsampled nodes are ignored in the next iteration of sampling, only sampled nodes in this order can be used to generate the next-order samples, e.g., H continues searching while F stops. By now PS samples a set of node pairs $\{(X, A)\}|X \in \{B, C, D, E, H, I, J\}\}$.

By restricting the max order of neighbors and the number of successors for each node to be at most 1, we successfully set an upper bound for the total number of node pairs. In fact, under the same training objective, pairs sampled by different sampling strategies will converge to different distributions of α and β. Theoretical optimal distributions are so far too complex to compute, but we show empirically that pairs sampled by Pairs Sampling can yield competitive results as RW-based does.

3.3 Text-driven Embedding Model

3.3.1 Intuition. Previous graph embedding models focus on generating graph embeddings from only structural information (denoted as NE) or incorporating text attributes with structural information (denoted as NE+TE). NE+TE models can be regarded as encoding structural information into an NE space and encoding textual information into a TE space. Even though complicated interactions are explored between these two spaces, we concern the textual information has not been fully utilized, and propose to generate graph embeddings from text embeddings (denoted as TE2NE). TE2NE is more suitable for large graphs with strong text dependency. A large-scale graph with rich texts may contain million- to billion-level nodes, where the number of nodes is much larger than the number of words. Besides, TE2NE can also apply to zero-shot scenarios, since no explicit NE is required in inference. Therefore,

we propose the Text-driven Graph Embedding (TGE) method that makes the most of textual information by projecting textual information into the NE space. To model the text, we adopt bidirectional LSTM (BiLSTM) [31] for its success in Nature Language Processing field [34, 36]. To deal with out-of-vocabulary words in unseen nodes, we adopt character-level embeddings. Character-level embeddings have proven success in NLP tasks [9, 31]. The advantages of character-level embeddings over word-level embeddings are summarized by [9]. Hence, we adopt character-level embeddings in addition to word-level embeddings.

3.3.2 Generating Embeddings. We denote the set of words as $\mathcal{D}^w$, the set of characters as $\mathcal{D}^c$ and text of node v_i to be $t_i = \{w_{ij}, 1 \le j \le |t_i|\}$, where $w_{ij} \in \mathcal{D}^w$ and $|t_i|$ is the length of t_i. Each word w_{ij} contains characters as $w_{ij} = \{c_{ijk}, 1 \le k \le |w_{ij}|\}$, where $c_{ijk} \in \mathcal{D}^c$ and $|w_{ij}|$ is the length of w_{ij}. We denote character- and word-level embedding vectors as $\mathbf{e}^c$ and $\mathbf{e}^w$ respectively.

We start by generating embedding of node v_i. We feed the sequence of character embeddings $\mathbf{e}^c_{ij,1:|w_{ij}|}$ into character-level BiLSTM and obtain a character-based word embedding $\mathbf{e}^{w'}_{ij}$ as in Eq. (8). The character-based word embedding is concatenated with the corresponding word embedding $\mathbf{e}^w_{ij}$ and fed into the word-level BiLSTM layer as in Eq. (9).

$$\mathbf{e}^{w'}_{ij} = \text{BiLSTM}^c(\mathbf{e}^c_{ij,1:|w_{ij}|}) \quad (8)$$

$$\mathbf{e}_i = \tanh(W\text{BiLSTM}^w([\mathbf{e}^w_{i,1:|T_i|}; \mathbf{e}^{w'}_{i,1:|T_i|}]) + b) \quad (9)$$

Now we obtain the text-based node embedding $\mathbf{e}_i$ of node v_i. We also set up a lookup layer which embeds v_i into *dim*-dimensional structure-based vector $\mathbf{e}'_i$ that is used to help train $\mathbf{e}_i$, and outputs $\mathbf{e}_i$ as the embedding vector of v_i.

3.3.3 Training. The training objective is Eq. (1). To reduce computation complexity, we define the loss function to be

$$\mathcal{L}_{\text{sim}}(v_i, v_j) = -\log(\sigma(\mathbf{e}'^{\top}_j \mathbf{e}_i)) - \sum_{v_k \sim P(v)}^{N_{\text{neg}}} \log(\sigma(-\mathbf{e}'^{\top}_k \mathbf{e}_i))$$

where $\mathbf{e}_i$, $\mathbf{e}'_j$ and $\mathbf{e}'_k$ denotes embeddings of the central node, the neighbor node in a pair, and the randomly sampled negative node, N_{neg} denotes the number of negative samples [35], σ denotes the sigmoid function and $P(v) \propto d_v^{3/4}$ denotes the distribution of nodes when sampling negative samples, where d_v is the degree of v.

We also apply L_2-regularization on parameters and embeddings We use AdaGrad optimizer [11] to minimize the loss.

4 EXPERIMENTS

4.1 Datasets

To verify the effectiveness and efficiency of PS, we conduct a series of experiments over 6 datasets. We list their details in Tab. 1, where $|V|$ and $|E|$ refer to number of nodes and edges, respectively. In practice, we keep all nodes and $p\%$ edges of the dataset *Data* for training and denote this setting as *Data@p%*. Before conducting our experiments, we pre-process texts including lowering all characters, removing stop words and discarding punctuations.

Table 1: Details of Datasets

Datasets	$\|V\|$	$\|E\|$	Type
Cora [32]	2,211	5,214	Citation Graph
Facebook [24]	4,039	88,234	Social Network
Zhihu [47]	10,000	43,894	Q&A Datasets
AstroPh [24]	18,772	198,110	Co-work Network
HepTh [24]	27,400	352,542	Citation Graph
SNOMED [10, 37]	391,892	2,047,749	Health Terminology

4.2 Baselines

We evaluate our model against several graph embedding models: **LINE** [45], **DeepWalk** [42] **node2vec** [17], **CANE** [47], **Paper2Vec** [14] and **STNE** [29]. Since all listed models are not capable of fitting in zero-shot scenarios. Therefore, we design a rule-based embedding method, **Text Matching**, which represents each node as the average vector of pre-trained word embeddings from Glove [41] for every word in the text. The performance of Text Matching reflects the dependency between graph and text structure.

4.3 Evaluation Metrics

In evaluation, we sample an unobserved link (v_c, v_p) and a non-exist link (v_c, v_n) as the positive and negative samples for each node v_c, respectively. we adopt AUC (Area Under Curve) [19] as the metric in two different ways: we use the a portion of pairs to train a logistic regression classifier, use the trained classifier to infer the connectivity of another set of pairs and compute the final AUC score as $\mathbf{AUC_{LR}}$; we directly compute the times of $\mathbf{e}_c^{\top}\mathbf{e}_p > \mathbf{e}_c^{\top}\mathbf{e}_n$ as $\mathbf{AUC_{pair}}$, similar to that in [30]. We report the results of all experiments in two columns, which represent AUC_{LR} and AUC_{pair} respectively.

4.4 Experiments of Pairs Sampling

4.4.1 Theoretical Analysis. Firstly, we compute the sample complexity of each method. Given the number of nodes $|V|$, average degree $\bar{d}$, walk length L, window size W and walk time T in RW, and max order O and sampling time N in PS, the number of sample pairs under RW and PS are $(2L - W - 1)WT|V|$ and $NO\bar{d}|V|$, and the ratio of them r is $\frac{(2L-W-1)WT}{NO\bar{d}}$. We compute the ratio of node pairs with fine-tuned parameters in Tab. 2. Note that real ratios will be larger for that the number of pairs sampled by PS is actually no greater than $NO\bar{d}|V|$. From results in Tab. 2, we can see that PS can significantly reduce the training samples (reducing ~99% samples) compared with RW. Note that r is often much larger in sparse networks like Cora, Zhihu and SNOMED. In graphs with small average degree, RW often conducts DFS-like walks and encounter leaf nodes, which results in frequent revisiting behaviors and hence more redundant samples.

Table 2: Ratios of Different Datasets

Datasets	$\|V\|$	$\|E\|$	$\bar{d}$	r
Cora@50%	2,211	2,607	2.33	183.60
Cora@100%	2,211	5,214	4.42	140.46
Facebook@50%	4,039	44,117	21.84	36.17
Zhihu@50%	10,000	21,947	4.28	369.16
AstroPh@50%	18,772	99,054	10.56	55.40
HepTh@50%	27,400	176,271	12.86	99.79
SNOMED@20%	391,892	409,550	2.09	409.36

4.4.2 Link Prediction. We evaluate PS against RW on 6 datasets listed in Section 4.1. We use PS and RW to generate training sets, and train node embeddings respectively. Results in Tab. 3 show that PS outperforms RW on almost all datasets. This indicate that even if PS adopts stronger assumptions for efficiency consideration, it still proves to be a competitive alternative to RW.

Table 3: Link Prediction Results of RW and PS

Datasets	Random Walk		Pairs Sampling	
	AUC_{LR}	AUC_{pair}	AUC_{LR}	AUC_{pair}
Cora@50%	0.9200	0.9293	**0.9272**	**0.9394**
Facebook@50%	0.9921	0.9892	**0.9922**	**0.9911**
Zhihu@50%	0.8659	**0.9144**	**0.8673**	0.9136
AstroPh@50%	0.9788	0.9768	**0.9795**	**0.9789**
HepTh@50%	0.9741	0.9648	**0.9743**	**0.9730**
SNOMED@20%	0.9350	0.9396	**0.9359**	**0.9402**

4.5 Experiments of TGE-PS

4.5.1 Link Prediction. We evaluate TGE-PS against all baseline models on SNOMED and HepTh respectively. The results are shown in Tab. 4, where "-" refers to results of failed experiments. From Tab. 4, we have following observations:

- TGE-PS outperforms all baseline models on both datasets. Incorporating textual information with structural information helps constructing expressive graph embeddings. Improvements of TGE-PS over PS and Paper2Vec over node2vec strongly support this point. Only preserving structural information like LINE or textual information like Text Matching both presents limitations.
- The improvements of TGE-PS are quite different on the two graphs. Recall that we use the text description of concepts in SNOMED and the whole abstract in HepTh, we owe this to that SNOMED has stronger text dependency than HepTh. The performance of Text Matching in SNOMED supports this point. Similar difference of improvements also appears between Paper2Vec and node2vec.

Table 4: Link Prediction Results of Different Models

Model	**SNOMED**@20%		**HepTh**@50%	
	AUC_{LR}	AUC_{pair}	AUC_{LR}	AUC_{pair}
LINE	0.6461	0.6985	0.7883	0.7520
DeepWalk	0.9164	0.9258	0.9711	0.9573
node2vec	0.9350	0.9396	0.9741	0.9648
Text Matching	0.8954	0.8717	0.7057	0.5700
TADW	-	-	0.8866	0.8977
CANE	0.9613	0.9544	0.9785	0.9388
Paper2Vec	0.9581	0.9604	0.9745	0.9748
STNE	-	-	0.9651	0.9572
PS	0.9359	0.9402	0.9758	0.9716
TGE-PS	**0.9721**	**0.9621**	**0.9793**	**0.9752**

It is also worth noting that not all models are capable of dealing with large graphs: TADW requires pre-loading the whole adjacent matrix with $O(|V|^2)$ complexity; the training speed of STNE is extremely slow on astronomical number of walks generated in SNOMED. We also try to conduct experiments on GraphGAN in link prediction task, but they fail on both datasets due to OOM. On the contrary, our TGE-PS performs well in handling large graphs.

4.5.2 Zero-Shot Experiments. Unlike common link prediction scenarios, where each node embedding is trained at least once before inference, zero-shot scenarios require prediction on link existence with unseen nodes. Zero-shot scenarios are common in real-world applications. For example, when updating medical terminology graphs like SNOMED, terms of newly found diseases will be added into existing databases, and it is possible for a human to infer link existence by text information with high accuracy even if he is not an expert in this field. Therefore, zero-shot scenario in fact reflects whether the model incorporates text information by capturing important message and potential connections or simply matching literal similarities.

We conduct zero-shot experiments on SNOMED and HepTh where 0.5% of the nodes and related edges are removed from the graph. Embeddings of these nodes will be generated by trained models directly and used for evaluation. As we stated in previous sections, zero-shot scenarios are not widely studied and most current models are incapable of generating embeddings for unseen nodes. However, to better study this problem, we manage to conduct experiments on a variant of CANE with only averaged TE part , which we denote as CANE (TE): We also conduct ablation experiments on zero-shot scenarios to analyze the influence of character- and word-level embeddings. In ablation experiments, we either replace $[\mathbf{e}^w; \mathbf{e}^{w'}]$ with $\mathbf{e}^{w'}$ (w/o word), or replace $[\mathbf{e}^w; \mathbf{e}^{w'}]$ with $\mathbf{e}^w$ (w/o char). High AUC scores in Tab. 5 indicate TGE-PS is practical and reliable in zero-shot scenarios. Besides, we have following observations:

- Word-level embeddings are essential in embedding nodes. This is not surprising given that there are much more words (100,471 in SNOMED and 72,083 in HepTh) than characters (88 in SNOMED and 59 in HepTh, mostly English characters and numbers). Besides, when words are often composed of sub-words as in SNOMED, the improvements of character-level embeddings are apparently higher.
- Using Text Matching as a baseline, TGE-PS presents larger promotion on HepTh than SNOMED. Unlike terms in SNOMED, texts in HepTh are abstracts of papers, which have longer average length. In this case, BiLSTM shows its advantages over naive matching methods in processing long texts.

Table 5: Zero-Shot Experiment Results

Model	SNOMED		HepTh	
	AUC_{LR}	AUC_{pair}	AUC_{LR}	AUC_{pair}
Text Matching	0.9059	0.8813	0.5934	0.6250
CANE (TE)	0.5271	0.5344	0.6036	0.5288
TGE-PS (w/o word)	0.5000	0.5003	0.5000	0.5037
TGE-PS (w/o char)	0.9701	0.9786	0.8979	**0.9485**
TGE-PS	**0.9760**	**0.9811**	**0.8990**	**0.9485**

ACKNOWLEDGMENTS

The work is supported by National Natural Science Foundation of China (61702327, 61772333, 81771937), Shanghai Sailing Program (17YF1428200). Weinan Zhang and Yong Yu are the corresponding authors.

REFERENCES

[1] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J Smola. 2013. Distributed large-scale natural graph factorization. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 37–48.

[2] Lars Backstrom and Jure Leskovec. 2011. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 635–644.

[3] Mikhail Belkin and Partha Niyogi. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*. 585–591.

[4] Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. 2011. Learning structured embeddings of knowledge bases. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*.

[5] Hongyun Cai, Vincent W Zheng, and Kevin Chang. 2018. A comprehensive survey of graph embedding: problems, techniques and applications. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[6] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 891–900.

[7] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep neural networks for learning graph representations. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI Press, 1145–1152.

[8] Jifan Chen, Qi Zhang, and Xuanjing Huang. 2016. Incorporate group information to enhance network embedding. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1901–1904.

[9] Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. 2016. A Character-level Decoder without Explicit Segmentation for Neural Machine Translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1693–1703.

[10] Kevin Donnelly. 2006. SNOMED-CT: The advanced terminology and coding system for eHealth. *Studies in health technology and informatics* 121 (2006), 279.

[11] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[12] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* 486, 3-5 (2010), 75–174.

[13] Yanwei Fu, Timothy M Hospedales, Tao Xiang, and Shaogang Gong. 2015. Transductive multi-view zero-shot learning. *IEEE transactions on pattern analysis and machine intelligence* 37, 11 (2015), 2332–2345.

[14] Soumyajit Ganguly and Vikram Pudi. 2017. Paper2vec: Combining graph and text information for scientific paper representation. In *European Conference on Information Retrieval*. Springer, 383–395.

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[16] Palash Goyal and Emilio Ferrara. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems* 151 (2018), 78–94.

[17] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[18] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584* (2017).

[19] James A Hanley and Barbara J McNeil. 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology* 143, 1 (1982), 29–36.

[20] Yuting Jia, Qinqin Zhang, Weinan Zhang, and Xinbing Wang. 2019. CommunityGAN: Community Detection with Generative Adversarial Nets. *arXiv preprint arXiv:1901.06631* (2019).

[21] Przemyslaw Kazienko and Tomasz Kajdanowicz. 2012. Label-dependent node classification in the network. *Neurocomputing* 75, 1 (2012), 199–209.

[22] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[23] Jason Lee, Kyunghyun Cho, and Thomas Hofmann. 2017. Fully character-level neural machine translation without explicit segmentation. *Transactions of the Association for Computational Linguistics* 5 (2017), 365–378.

[24] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[25] Chaozhuo Li, Senzhang Wang, Dejian Yang, Zhoujun Li, Yang Yang, Xiaoming Zhang, and Jianshe Zhou. 2017. PPNE: property preserving network embedding. In *International Conference on Database Systems for Advanced Applications*. Springer, 163–179.

[26] Juzheng Li, Jun Zhu, and Bo Zhang. 2016. Discriminative deep random walk for network classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1004–1013.

[27] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58, 7 (2007), 1019–1031.

[28] Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. 2015. Character-based neural machine translation. *arXiv preprint arXiv:1511.04586* (2015).

[29] Jie Liu, Zhicheng He, Lai Wei, and Yalou Huang. 2018. Content to node: Self-translation network embedding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1794–1802.

[30] Linyuan Lü and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications* 390, 6 (2011), 1150–1170.

[31] Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1064–1074.

[32] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 2000. Automating the construction of internet portals with machine learning. *Information Retrieval* 3, 2 (2000), 127–163.

[33] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[34] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

[35] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[36] Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 234–239.

[37] Jane Millar. 2016. The Need for a Global Language-SNOMED CT Introduction. *Studies in health technology and informatics* 225 (2016), 683–685.

[38] SS Nishana and Subu Surendran. 2013. Graph embedding and dimensionality reduction-a survey. *International Journal of Computer Science & Engineering Technology (IJCSET)* 4, 1 (2013), 29–34.

[39] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1105–1114.

[40] S Pan, J Wu, X Zhu, C Zhang, and Y Wang. 2016. Tri-party deep network representation. In *IJCAI International Joint Conference on Artificial Intelligence*.

[41] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[42] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[43] Cicero Nogueira dos Santos and Victor Guimaraes. 2015. Boosting named entity recognition with neural character embeddings. *arXiv preprint arXiv:1505.05008* (2015).

[44] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–93.

[45] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1067–1077.

[46] Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)* 3, 3 (2007), 1–13.

[47] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. Cane: Context-aware network embedding for relation modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1722–1731.

[48] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1225–1234.

[49] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. GraphGAN: graph representation learning with generative adversarial nets. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[50] Pengyang Wang, Jiawei Zhang, Guannan Liu, Yanjie Fu, and Charu Aggarwal. 2018. Ensemble-Spotting: Ranking Urban Vibrancy via POI Embedding with Multi-view Spatial Graphs. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 351–359.

[51] Min Xie, Hongzhi Yin, Hao Wang, Fanjiang Xu, Weitong Chen, and Sen Wang. 2016. Learning graph-based POI embedding for location-based recommendation. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 15–24.

[52] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *Thirtieth*

AAAI Conference on Artificial Intelligence.

[53] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Chang. 2015. Network representation learning with rich text information. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[54] Shuang-Hong Yang, Bo Long, Alex Smola, Narayanan Sadagopan, Zhaohui Zheng, and Hongyuan Zha. 2011. Like like alike: joint friendship and interest propagation in social networks. In *Proceedings of the 20th international conference on World wide web*. ACM, 537–546.

[55] Chao Zhang, Keyang Zhang, Quan Yuan, Haoruo Peng, Yu Zheng, Tim Hanratty, Shaowen Wang, and Jiawei Han. 2017. Regions, periods, activities: Uncovering urban dynamics via cross-modal representation learning. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 361–370.

[56] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. 2018. Network representation learning: A survey. *IEEE transactions on Big Data* (2018).

BoFGAN: Towards A New Structure of Backward-or-Forward Generative Adversarial Nets

M.K. Sophie Chen†
Institute of Computer Science and Technology, Peking University, Beijing, China
Department of Computer Science and Engineering, UC San Diego, La Jolla, USA

Xinyi Lin†
Institute of Computer Science and Technology, Peking University, Beijing, China

Chen Wei
Turing Robot, Beijing, China

Rui Yan*
Institute of Computer Science and Technology, Peking University, Beijing, China
ruiyan@pku.edu.cn

ABSTRACT

Natural Language Generation (NLG), as an important part of Natural Language Processing (NLP), has begun to take full advantage of recent advances in language models. Based on recurrent neural networks (RNNs), NLG has made ground breaking improvement and is widely applied in many tasks. RNNs typically learn a joint probability of words, and the additional information is usually fed to RNN's hidden layer using implicit vector representations. Still, there exists some problem unsolved. Standard RNN is not applicable when we need to impose *hard constraints* on the language generation tasks: for example, standard RNNs cannot guarantee designated word(s) to appear in a target sentence to generate. In this paper, we propose a Backward-or-Forward Generative Adversarial Nets model (BoFGAN) to address this problem. Starting from a particular given word, a generative model at every time step generates a new *preceding* or *subsequent* word conditioned on the generated sequence so far until both sides reach an end. To train the generator, we first model it as a stochastic policy using Reinforcement Learning; then we employ a discriminator to evaluate the quality of a complete sequence as the end reward; and lastly, we apply Monte Carlo (MC) search to estimate the long-term return and update the generator via policy gradient. Experimental results demonstrate the effectiveness and rationality of our proposed BoFGAN model.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Natural language generation**; • **Information systems** → *Web applications.*

*Corresponding author: Rui Yan (ruiyan@pku.edu.cn).
† indicates equal contributions.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313734

KEYWORDS

BoFGAN; constrained generation; reinforcement learning

ACM Reference Format:
M.K. Sophie Chen, Xinyi Lin, Chen Wei, and Rui Yan. 2019. BoFGAN: Towards A New Structure of Backward-or-Forward Generative Adversarial Nets. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313734

1 INTRODUCTION

Natural Language Generation (NLG) is the process of producing high-quality texts from some underlying representation of key information. NLG is an important task in Natural Language Processing, and it is also a critical step in many applications, such as machine translation [1, 33], dialogue systems [36, 38], stylistic writing [6, 17, 35], storytelling [16, 37], and abstractive summarization [7, 28], etc. The task can be roughly divided into 2 steps: content determination and sentence structuring. Content determination involves choices of what semantic information to convey, which is closely related to the specific application and/or scenario. Although contents may vary, the way for sentence structuring of natural languages is similar even given different applications/scenarios. Thus, it is likely to propose a general framework to generate texts from given contents. There are three dominant approaches for sentence generation architectures: human-crafted templates [19, 27], grammar-based systems [3], statistical data-driven approaches [2, 20, 21, 26].

Template-based method is widely used in industry systems to generate articles by filling the templates with correct terms. It allows full control over the quality of the output, but it is quite labor-intensive, and rigid. It is only applicable when the application domains are small and variation is expected to be restricted. *Grammar*-based system is more flexible than template-based ones. However, the complexity and difficulty to design and to use these systems make them less popular. Recently, *statistical* approaches gain more attention in research work powered by deep learning. In particular, deep neural networks (DNNs) lead the way since they can learn dense, low-dimensional, and distributed representations at a more abstract level for sentence generation.

It is well-suited to get rid of hand-crafted templates and rules by capturing grammatical and semantic generalisations in a fixed size vector [18, 22, 25]. Thereinto, recurrent neural networks (RNNs) [21], a prevailing class of language models, has achieved notable successes in sequential modeling. It projects histories into a low-dimensional space and propagates the projection through the sequence to build up the implicit relations among the sequence. Compared to the standard *n*-gram model, RNNs can represent sequences of varying lengths, while avoiding both data sparseness and an explosion of parameter numbers.

There is a prominent drawback for RNNs: once the model is learned, the model generates autonomously. Standard RNN is not directly applicable when we need to impose *hard constraints* on the language generation tasks. In other words, for example, standard RNNs cannot guarantee designated word(s) to "explicitly appear" in the generated sentence. The designated word is the hard-constraint for sentence generation. The need for constrained generation is real.

Templates and grammar based systems are inflexible since different scenarios require different templates and grammars: the given word may not match any slots/templates. Existing language models, such as RNNs, can learn the joint probability of a sentence with an additional information. With a given word information fed into the hidden layer, RNN can generate a sentence semantically correlated with the word, but there is no policy to guarantee the word will appear.

Mou *et al.* [23] proposed a backward and forward language model (B/F LM) to tackle the problem. Given a specific word, the model generates previous words and subsequent words simultaneously or asynchronously. That is to say, once the generation direction is determined, the generation process will be executed until the end of the direction. A more natural way is that given the constraint word, we generate sentences incrementally, i.e., from words to word groups (e.g., phrases), from word groups to larger phrases, and from phrases to sentences. Such a process is to some extent analogous to human cognitions about natural language generation [8].

Here is a motivating case. Given a word "*convolutional*" and the target sequence is "*word order for text categorization with convolutional neural networks*", the B/F language model firstly generate backward words (*"with", "categorization", "text", "for", "order", "word"*) and then (*"neural", "networks"*). However, the entire sequence consists of two parts using *"with"* between *"word order for text categorization"* and *"convolutional neural networks"*. It is difficult to derive the first part from the single word *"convolutional"* since the word itself contains little information while it is more feasible to do so from the phrase *"convolutional neural networks"*.

In this paper, we propose a novel framework to tackle the problem of imposing hard constraints on sentence generation. Rather than *pre-defined* backward and then forward direction to generate words, we leave the choice to the model itself. The generative model is treated as an agent of Reinforcement Learning (RL): the state is the generated tokens so far and the action is the combination of what the next token is and which direction (backward, or forward) to generate it. We consider the generation procedure as a sequential decision making process: at each time, it decides to generate either a preceding or a subsequent word conditioned on generated tokens. One side will stop "growing" when it has generated the $\langle eos \rangle$ token. The reward is given by a discriminator, which evaluates the generated sequences and feeds the evaluation back to guide the learning of the generative model. Since the gradients can not pass back to the generator when the output is discrete, we follow SeqGAN [39] and regard the generative model as a stochastic parametrized policy. We employ Monte Carlo (MC) search to approximate the state-action value and directly train the policy via policy gradient. Theoretically, our model guarantees a sentence with a designated constraint word. The model does not plan a whole sentence with merely a single word in the first place; itself decides backward or forward to generate words, it has the chance to generate words into phrases, and to generate phrases into sentences.

Our contributions are as follows: we propose a novel *backward-or-forward* model with gradual information enrichment to generate a sentence given hard-constraints. The model generates text pieces to eliminate uncertainty and finally organize them into a complete sentence. We formulate the model under a Generative Adversarial Network framework and a backward/forward sequential decision making process. The model constructs sentences in a hierarchial generation process, from words to "phrases" and to sentences.

2 RELATED WORK

2.1 RNN and Sequence-to-Sequence

An RNN is widely used in language modeling, which is capable of capturing sequential dependency. It consists of an input layer, a hidden layer and an output layer. At each time t, a word is mapped to a fixed dimensional vector as an input x_t, which is called word embedding. The hidden state vector h_t depends on its previous hidden state h_{t-1} and the input vector x_t. The output layer predicts the generated word y_t from a vocabulary using a $softmax$ function. The formulas are as follows:

$$h_t = \text{RNN}(x_t, h_{t-1})$$

$$y_t = \text{softmax}(W_{out} h_t)$$

The vanilla RNN has the gradient vanishing or explosion problem. To solve this problem, Graves and Schmidhuber [10] proposed long short term memory (LSTM) unit, which contains a memory cell and three kinds of gates controlling what information is retained or forgotten.

Character-level RNN language model [32] has been applied to various generation tasks at early times. Later, a text-to-text NLG framework named sequence-to-sequence (Seq2Seq) [33] plays an important role in language generation. It combines the subtasks (i.e., content determination and sentence structuring) together through an encoder-decoder framework. The encoder encodes the source sentence into a fixed-size vector, acting as the content decision. The decoder decodes the vector to the target sentence, acting as the sentence realisation. The vector passed from encoder to decoder is an underlying representation of the content information of the generated sentence. LSTM-based RNN can be used as the encoder and the decoder.

2.2 B/F Language Model

The standard RNNs or LSTMs are able to formulate language models but they cannot deal with language models with constraints.

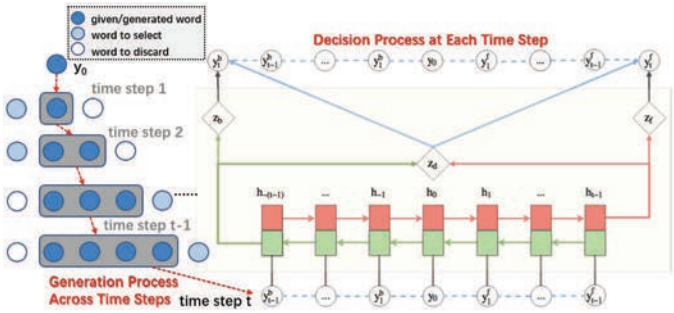


Figure 1: BoFGAN sequence generation at time step t: G_θ would generate the next two tokens conditioned on the previously generated sequence $Y_{0:t-1}$. Note that the red flow is forward h_f propagation, and the green flow is the backward h_b propagation. The diamonds are softmax layers: z_d represents the confidence for direction, and z_f, z_b choose the next token. Across time steps, the model repeats the process incrementally, starting from the constraint word, until the whole sentence is completed by $\langle eos \rangle$'s.

Under the scenarios that constraints exist, the B/F language model, introduced by [23], is a new language model towards the generation of constrained natural language. Given a specific word, the model generates preceding words and subsequent words simultaneously or asynchronously. In the simultaneous setting (syn-B/F model), there is a shared RNN sequence which emits words in the backward and forward directions at the same time. The backward language model and the forward language model share the same input layer and hidden layer. In the asynchronous setting (asyn-B/F model), the model generates the backward sequence firstly, which is then fed to another RNN to generate the forward sequence.

For both settings, the B/F language model pre-defines the directions to generate the sequence while we are seeking for a more flexible model that decides which direction to generate according to the current status and changes the generation strategies from time to time, dynamically.

2.3 GAN and SeqGAN

Generative adversarial networks (GAN) proposed by [9] is a state-of-the-art generative model, which has been successfully applied in generating picture samples [5, 11]. GAN includes a discriminator D classifying the given data as real or fake, and a generator G generating almost real data to confuse D. G and D finally reach a Nash equilibrium, which can be regarded as a win-win situation. However, problems arise when applying GAN in sequential language generation.

Originally, GAN is defined in real number domain for continuous conditions. G is updated via the gradient loss from D. In general, it makes sense only when we slightly change the synthetic data which is continuous, such as pixel values. When GAN generates discrete tokens, the slightly changed data may not have corresponding tokens from the vocabulary. Besides, we can only evaluate the quality of a sentence when it is completely generated. To this end, Yu *et al.* [39] proposed a model named SeqGAN considering the generation procedure as a sequential decision making process, where the generator is an agent of Reinforcement Learning (RL) and the discriminator is used to score the generated sequence and feedback the reward. They employ Monte Carol (MC) search to approximate the state-action value and train the policy via the policy gradient.

However, the GAN and SeqGAN models cannot deal with constrained information so that we can not guarantee the designated word will appear in the generated sentence.

3 BOFGAN

In this part, we introduce our proposed *Backward-or-Forward GAN* (BoFGAN) model in details. The model consists of two parts: a generator and a discriminator. The generator is an RL agent to generate a sentence from a given token. Starting from a single word, each time it needs to make two choices: where to put the new word and what the new word is. The generation process will reach an end when both sides have met an $\langle eos \rangle$ token or the sequence length exceeds limits. In other words, the state is the tokens generated so far, and the actions consist of 1) the position and 2) selection of the new word. Since it is hard to evaluate an incomplete sequence with an appropriate reward for each step, we also employ MC search to approximate the action value function and evaluate the complete sequence using a discriminator. We treat the agent's behaviour as a stochastic parameterized policy and train it using policy gradient method. The discriminator is trained to evaluate how likely a sequence is from real data or not by learning from real-world data and synthetic data from the generator. The generator and discriminator are trained to improve each other iteratively in an adversarial way.

3.1 Generator

G_θ is the θ-parameterized generative model. Γ is the vocabulary of candidate words plus the empty token ε. y_0 is the **hard-constraint** word, i.e., the start token given to the generator in the beginning and is required to exist in the generated sentence.

At timestep t, the state s_{t-1} is the previously generated sequence $Y_{0:t-1} = (y^b_{t-1}, y^b_{t-2}, ..., y^b_1, \mathbf{y_0}, y^f_1, ..., y^f_{t-2}, y^f_{t-1})$, where $y^f_i, y^b_i \in \Gamma$. The action a_t is to separately append new tokens (y^b_t, y^f_t) to two ends of $Y_{0:t-1}$, i.e., the next state s_t means $Y_{0:t} = (y^b_t, y^b_{t-1}, y^b_{t-2}, \ldots, y^b_1, y_0, y^f_1, \ldots, y^f_{t-2}, y^f_{t-1}, y^f_t)$. For the *backward or forward* language modeling, at least one of (y^b_t, y^f_t) is ε because the G_θ is allowed to append one valid word to the sequence at each timestep of generation.

To generate two tokens at a time, we use the bi-directional RNN [29] to model the policy $G_\theta(y^b_t, y^f_t | Y_{0:t-1})$. The bi-RNN maps $X_{0:t-1} = (x_{-t+1}, x_{-t+2}, ..., x_{-1}, x_0, x_1, ..., x_{t-1})$, the input embedding representation of $Y_{0:t-1}$, into two sequences of hidden state, by using the update function g recursively:

$$h_{0:t-1} = (\{h^b_{-t+1}, h^f_{-t+1}\}, \ldots, \{h^b_0, h^f_0\}, \ldots, \{h^b_{t-1}, h^f_{t-1}\})$$

$$\begin{aligned} h^f_i &= g_f(h^f_{i-1}, x_i) \\ h^b_i &= g_b(h^b_{i+1}, x_i) \end{aligned} \tag{1}$$

where h^f_{-t} and h^b_t, the initial vectors to feed into the hidden layer, are both zero vectors. Note that we allow the subscripts be negative for convenience and each subscript stands for the relative position to y_0. g_f and g_b are implemented by LSTM cells, since LSTM cells can faster converge and eliminate the problem of vanishing or exploding gradients in sequence via self-regularization [10].

Generator's action actually consists of two parts: 1) the position and 2) the selection of the new token, thus, we use three output units to parameterize the policy.

$$\begin{aligned} z_d &= \text{softmax}\left(b_d + W_d h\right) \\ z_f &= \text{softmax}\left(b_f + W_f h^f_{t-1}\right) \\ z_b &= \text{softmax}\left(b_b + W_b h^b_{-t+1}\right) \end{aligned} \tag{2}$$

where W_d, W_f, W_b are weight matrics and b_d, b_f, b_b are bias vectors. h is the concatenation of h^f_{t-1} and h^b_{-t+1}. z_d is a two-dimension vector to control the preference for which direction to append newly generated tokens, and z_f, z_b are used to represent the output token distribution separately:

$$\begin{aligned} p\left(y^f_t | Y_{0:t-1}\right) &= z_{d0} \cdot z_f \\ p\left(y^b_t | Y_{0:t-1}\right) &= z_{d1} \cdot z_b \end{aligned} \tag{3}$$

Note that

$$\sum_{y^f_t \in \Gamma} p\left(y^f_t | Y_{0:t-1}\right) + \sum_{y^b_t \in \Gamma} p\left(y^b_t | Y_{0:t-1}\right) = 1.$$

At timestamp t, the generator would only sample one word from $p\left(y | Y_{0:t-1}\right)$. If one side has obtained $\langle eos \rangle$ token, the corresponding direction of z_d would be forced to be 0 and the other to be 1. When both sides reach an end or the total length goes beyond the limit, the generation is terminated. The whole generation process is shown in Figure 1.

3.2 Discriminator

D_ϕ is the ϕ-parameterized discriminative model. It is a binary classifier that takes a sequence as the input and outputs a probability indicating how likely the input is generated by humans. Deep learning methods have been widely used in the task of text classification and achieves good performance, especially using the convolutional neural network (CNN)-based methods [12, 13, 15]. CNN is similar to the n-gram model to some degree as the filter window can be seen as the n-gram method. The use of convolution layer and pooling layer can largely reduce the number of parameters and extracts high-level information of text at the same time. We follow SeqGAN [39] and choose the same architectures as our discriminator.

The overall classification framework is based on CNN. The input sequence is represented as:

$$x_{1:T} = x_1 \oplus x_2 \oplus \cdots \oplus x_T$$

where $\oplus$ is the concatenation operator and $x_t \in \mathbb{R}^k$ is the k-dimensional word embedding corresponding to the t-th word in the sequence. The convolutional operation is then applied to a window size of m words with a kernel $w \in \mathbb{R}^{m \times k}$. We can use multiple kernels to extract different features and the window size varies when needed. Each feature map can be denoted as:

$$\mathbf{c} = [c_1, c_2, \cdots, c_{T-m+1}]$$

where

$$c_i = f(w \cdot x_{i:i+m-1} + b)$$

Here f is a non-linear function and $b \in \mathbb{R}$ is a bias term. Then, a max-over-time operation is applied to the feature map getting the most important feature $\tilde{c} = max\{\mathbf{c}\}$

We add highway networks [31] after the pooling operation. At last, we apply a fully connected layer to distinguish whether the input sequence is from human or not. The loss function is the cross entropy between the ground truth label and the classifier output.

3.3 BoFGAN via Policy Gradient

To guide the learning of the generator, we establish an adversarial reinforcement architecture. The key idea of the system is to encourage the G_θ to generate a semantically coherent and grammatically correct sentence with a constraint word. In other words, our objectives are that 1) the generated sequence includes the designated constraint word, and 2) is indistinguishable from human generated sentences.

We construct a bi-directionally growing generator. On one hand, the model guarantees that the constraint word will appear; on the other hand, it splits the whole generation task into sub-tasks, i.e., from words to word groups (e.g., phrases), from text pieces to sentences, making each sub-task easier than the task to generate a sentence as a whole. The process is quite similar to human cognitions [8].

We employ a discriminator to assess the quality of the generated sentence in the end. If the synthetic sentence gets a high probability from D_ϕ, that means the sentence is good enough to fool the discriminator and shows the effectiveness of the generator. However, it exists a problem applying traditional GAN to our model: it has difficulty to pass back the gradient to G_θ due to the discrete output. Thus, we adopt the policy gradient method to train the generator, and we use MC search to estimate the action-value function and the discriminator to provide the reward.

To improve the authority of the discriminator, we periodically re-train the discriminator by feeding real-world data and synthetic data generated by the upgraded Generator G_θ. The objective of the discriminative model is to minimize the cross entropy of the classification:

$$\min_\phi -\mathbb{E}_{Y \sim p_{\text{data}}}\left[\log D_\phi(Y)\right] - \mathbb{E}_{Y \sim G_\theta}\left[\log\left(1 - D_\phi(Y)\right)\right] \tag{4}$$

Each time when we obtain an upgraded discriminator D_ϕ, we continue the improvement of Generator G_θ. The objectives of G_θ is to maximize the state value of state s_0.

$$\begin{aligned} J(\theta) &= V_\theta(s_0) \\ &= \mathbb{E}\left[Q_\theta(s_0, a)\right] \\ &= \sum_{y^b_1, y^f_1 \in \Gamma} G_\theta\left(y^b_1, y^f_1 | s_0\right) \cdot Q^{G_\theta}_{D_\phi}\left(s_0, y^b_1, y^f_1\right) \end{aligned} \tag{5}$$

$V_\theta(s_0)$ is the state value of s_0, $Q_\theta(s_0, a)$ is the action value function, and the $Q^{G_\theta}_{D_\phi}\left(s_0, y^b_1, y^f_1\right)$ is the approximation of $Q_\theta\left(s_0, y^b_1, y^f_1\right)$

obtained from MC Search. Since there is no intermediate reward, we utilize the discriminator to provide the reward for the complete sequence, apply MC search with a roll-out policy G_β to sample the rest of the sequence, and estimate action value function with the empirical mean return. Note that, to improve the accuracy, we set β the same as θ.

To reduce variance, we run the roll-out policy for N times to get a batch of samples. Starting from sequence $Y_{0:t}$, the sampled sequences are represented as

$$\{Y_{0:T}^1, ..., Y_{0:T}^N\} = \mathrm{MC}^{G_\beta}(Y_{0:t}; N)$$

Thus, the $Q_{D_\phi}^{G_\theta}(s_{t-1}, y_t^b, y_t^f)$ can be written as:

$$Q_{D_\phi}^{G_\theta}(s_{t-1}, y_t^b, y_t^f) = \begin{cases} \frac{1}{N}\sum_{n=1}^{N} D_\phi(Y_{0:T}^n),\ Y_{0:T}^n \in \mathrm{MC}^{G_\beta}(Y_{0:t}; N) & \text{for } t < T \\ D_\phi(Y_{0:T}) & \text{for } t = T \end{cases} \tag{6}$$

We use policy gradient method to update G_θ. Following [34], the gradient of $J(\theta)$ can be derived as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{s_{t-1}\sim G_\theta}\Big[\sum_{y_t^b, y_t^f \in \Gamma} \nabla_\theta G_\theta(y_t^b, y_t^f | s_{t-1}) \cdot Q_{D_\phi}^{G_\theta}(s_{t-1}, y_t^b, y_t^f)\Big] \tag{7}$$

Thus, the equation above can be unbiasedly estimated using likelihood ratios as:

$$\begin{aligned} \nabla_\theta J(\theta) &\simeq \frac{1}{T}\sum_{t=1}^{T}\sum_{y_t^b, y_t^f \in \Gamma} \nabla_\theta G_\theta(y_t^b, y_t^f | s_{t-1}) \cdot Q_{D_\phi}^{G_\theta}(s_{t-1}, y_t^b, y_t^f) \\ &= \frac{1}{T}\sum_{t=1}^{T}\sum_{y_t^b, y_t^f \in \Gamma} G_\theta(y_t^b, y_t^f | s_{t-1}) \nabla_\theta \log G_\theta(y_t^b, y_t^f | s_{t-1}) \cdot Q_{D_\phi}^{G_\theta}(s_{t-1}, y_t^b, y_t^f) \\ &= \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{(y_t^b, y_t^f)\sim G_\theta(y_t^b, y_t^f | s_{t-1})}\Big[\nabla_\theta \log G_\theta(y_t^b, y_t^f | s_{t-1}) \cdot Q_{D_\phi}^{G_\theta}(s_{t-1}, y_t^b, y_t^f)\Big] \end{aligned} \tag{8}$$

For brevity, we omit several details of derivations. We can approximate the expected value with sampled means, thus, we can update the parameters θ as:

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta),$$

where α is learning rate. We can adopt gradient algorithms such as Adam [14] here.

To sum up, we first pretrain the generator and the discriminator. Then the generator and discriminator are trained alternately. For the discriminator, positive samples are from human generated sequences and negative samples are synthetic sequences generated by the generator. We need to periodically re-train the discriminator for several times as we get synthetic data of higher quality from the improved generator. In turn, the generator needs to be re-trained after we obtain an upgraded discriminator. In the end, the discriminator and the generator can both get improved.

4 EXPERIMENT

We conduct experiments to evaluate the effectiveness and to understand the inherent mechanism of BoFGAN.

4.1 Experimental Setups

Data and Pre-processing. We implement our experiments over the the Westbury Lab Wikipedia Corpus [30]. We carry out the data pre-processing, including data selection and data cleaning. First, we split the paragraphs into sentences and replace some entities with designated symbols, such as numbers, name mentions, temperature and so on. Since extremely short sentences are less meaningful, we extract one million sentences of length between 5 and 10 tokens as dataset $Data_1$, and one million sentences of length between 10 and 20 tokens as dataset $Data_2$. Both vocabulary sizes are set as 30,000, and the words out of vocabulary ($\leq$ 25 occurrences) were grouped as a single token, ⟨UNK⟩. Then we remove those sentences with more than three ⟨UNK⟩'s. Both $Data_1$ and $Data_2$ have 900,000 sentences, including 800,000 samples for training, and 100,000 for testing.

Baselines. We compare our model with traditional RNN language models as one of our baselines. Since standard RNNs can only generate sequence in one direction, it cannot handle our task scenario if a constraint word occurs in the middle of the target sentence. Therefore, for RNNs, we test the performance of **Forward-RNN** (F-RNN), starting from the constraint word, as well as the performance of **Backward-RNN** (B-RNN), ending by the constraint word. RNN is capable to deal with sequential generation in only one direction. We also include the B/F language models with RNNs. As mentioned above, B/F language model is designed for the same scenario as ours: to generate a sentence with the hard constraint word. The **asyn-B/F** language model is the best method as reported in [23].

In the pre-training process, asyn-BF language model is splitted into two part: the forward one and the backward one, each of which is trained as normal RNN language models. We feed the first word of each sentence as the start token, and the forward sequence reconstructs the original sentence. For the backward one, we reverse the sequence and train in the same way. The pretraining loss of asyn-BF is as:

$$-\frac{1}{M}\sum_{i=1}^{M}\left[\log p_f(Y_{1:T}|y_0) + \log p_b(Y_{T-1:0}|y_T)\right] \tag{9}$$

where p_f and p_b are the probability of forward generation and backward generation, and M is the batch size.

Implementation. Since BoFGAN is implemented based on the bi-directional architecture, thus, we use the same pretraining method mentioned above for the generator G_θ. At each time step, it actually acts like a bi-directional RNN if we ignore the direction decision layer z_d. After pretraining, the generator is preliminarily able to generate the start or end token for a sequence. As for the softmax layer of z_d, we leave it for next training stage.

We use LSTM architectures in both our generator and asyn-B/F language model. Actually, most of the RNN variants, such as the GRU [4] and soft attention mechanism [1], can also be applied. We adopt Adam gradient algorithms [14] for all the stochastic optimization, including the training of generator and discriminator. Adam is computationally efficient, has low memory cost, requires little tuning, and is invariant to diagonal rescaling of the gradients.

Evaluation Metric. Since our model is not learned by joint probability of word estimation during generation, it is infeasible to

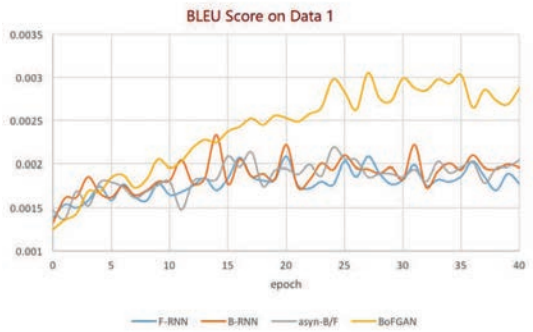


Figure 2: BLEU score curves on dataset $Data_1$

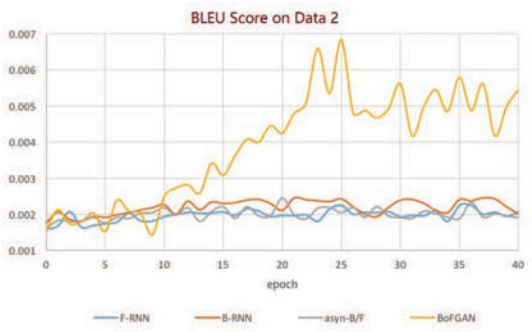


Figure 3: BLEU score curves on dataset $Data_2$

Table 1: Evaluation scores of different groups, measured in terms of BLEU evaluations.

Methods	BLEU of $Data_1$	BLEU of $Data_2$
F-RNN	1.82	2.04
B-RNN	1.96	2.37
asyn-B/F	1.84	2.02
BoFGAN	**2.83**	**4.73**

evaluate with *perplexity* as other language model studies do. *BLEU* is an algorithm to evaluate the quality of generation by comparing the overlap between the generated texts and the target references from humans [24]. BLEU is calculated by taking the geometric mean of the N-gram precision scores and then multiply the result by an exponential brevity penalty factor. Readers may refer to [24] for details. In this work, we regard all target sentences with the constraint word as references for BLEU calculation. N is set to 4 combining BLEU-1 to BLEU-4.

4.2 Experimental Results

The BLEU performance of all methods is provided in Table 1, measuring the overlap between the generated sentences and the targeted sentences as references. For each method, we use the best model in training stage to generate candidate samples and calculate their BLEU score. We can see that BoFGAN shows apparent advantages over other methods. BoFGAN outperforms baselines both on dataset $Data_1$ and $Data_2$, and it shows much larger gap in $Data_2$, which is trained to generate longer sentences. As mentioned above, when a sentence is longer, it is more likely to contain complex structures. This situation means that it would be more favorable for BoFGAN to perform its stronger capability. There is not so much difference among three baselines, since they are all based on the same RNN language model. Note that Forward RNN and Backward RNN has a "prior" knowledge because they are aware of the position of the constraint word (at the beginning or the end). Besides, they can only generate sequence in one way. Thus, if they are fed with a constraint word at a random position of the target sentence, they would perform much worse than BoFGAN and asyn-BF model.

In Figures 2-3, we show the learning curves, which strongly illustrate the effectiveness of BoFGAN. In the pretraining period—from epoch 0 to 10—BoFGAN is even slightly weaker than baselines since parameters of BoFGAN are not fully trained. In the training stage, i.e., from epoch 11 to 40, BoFGAN quickly outperforms baselines and enlarges the gap gradually, which demonstrates the effectiveness.

5 CONCLUSION

In this paper, we proposed a BoFGAN model for hard-constrained natural language generation. The model consists of two parts: a generator treated as a stochastic policy in reinforcement learning, and a discriminator to give a reward. Given the constraint word, we can generate a complete sentence containing it. The generation process is innovative. Starting from the given word, 1) which direction to generate and 2) what is the next token to generate are conditioned on previously generated words so far. Thus, we can form a word group (e.g., a phrase) from individual words and form a sentence from the word groups (or phrases).

Experimental results show that BoFGAN outperforms other baselines using BLEU scores. The generation process enlightens us another perspective of natural language generation paradigm: not only sequential, but also *backward-or-forward*. We will investigate how to organize multiple inconsecutive constraint words with our proposed model.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Program of China (No. 2017YFC0804001), the National Science Foundation of China (NSFC No. 61672058; NSFC No. 61876196), and Tencent Open Research Fund.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473* (2014).
[2] Srinivas Bangalore and Owen Rambow. 2000. Corpus-based lexical choice in natural language generation. In *ACL*. 464–471.
[3] John A. Bateman. 1997. Enabling technology for multilingual natural language generation: the KPML development environment. *Natural Language Engineering* 3, 1 (1997), 15–55.
[4] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv:1409.1259* (2014).
[5] Emily Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. 2015. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. In *NIPS*. 1486–1494.
[6] Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
[7] Shen Gao, Xiuying Chen, Piji Li, Zhaochun Ren, Lidong Bing, Dongyan Zhao, and Rui Yan. 2019. Abstractive Text Summarization by Incorporating Reader Comments. In *AAAI'19*.

[8] Merrill F Garrett. 1980. Levels of processing in sentence production. *Language production I* (1980), 177–220.

[9] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. In *NIPS*. 2672–2680.

[10] Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18, 5 (2005), 602–610.

[11] Daniel Jiwoong Im, Chris Dongjoo Kim, Hui Jiang, and Roland Memisevic. 2016. Generating images with recurrent adversarial networks. *arXiv:1602.05110* (2016).

[12] Rie Johnson and Tong Zhang. 2014. Effective Use of Word Order for Text Categorization with Convolutional Neural Networks. *arXiv:1412.1058* (2014).

[13] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *arXiv:1408.5882* (2014).

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980* (2014).

[15] Ji Young Lee and Franck Dernoncourt. 2016. Sequential Short-Text Classification with Recurrent and Convolutional Neural Networks. In *HLT-NAACL*. 515–520.

[16] Juntao Li, Lidong Bing, Lisong Qiu, Dongmin Chen, Dongyan Zhao, and Rui Yan. 2019. Learning to Write Creative Stories with Thematic Consistency. In *AAAI'19*.

[17] Juntao Li, Yan Song, Haisong Zhang, Dongmin Chen, Shuming Shi, Dongyan Zhao, and Rui Yan. 2018. Generating Classical Chinese Poems via Conditional Variational Autoencoder and Adversarial Training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 3890–3900.

[18] Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. In *CoNLL*. 104–113.

[19] S. W Mcroy. 2003. An augmented template-based approach to text realization. *Natural Language Engineering* 9, 4 (2003), 381–420.

[20] Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. CGMH: Constrained Sentence Generation by Metropolis-Hastings Sampling. In *AAAI'19*.

[21] Tomas Mikolov, Martin KarafiÃąt, Lukas Burget, Jan CernockÃ¡, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*. 1045–1048.

[22] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. 3111–3119.

[23] Lili Mou, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Backward and Forward Language Modeling for Constrained Sentence Generation. *Computer Science* 4, 6 (2016), 473–482.

[24] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *ACL*. 311–318.

[25] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. 1532–1543.

[26] Adwait Ratnaparkhi. 2000. Trainable Methods for Surface Natural Language Generation. *arXiv:cs.CL/0006028* (2000).

[27] Ehud Reiter, Chris Mellish, and Jon Levine. 1995. Automatic Generation of Technical Documentation. *Journal of Applied Artificial Intelligence* (1995).

[28] Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization.. In *EMNLP*. 379–389.

[29] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* (1997).

[30] C. Shaoul and Westbury C. 2010. The Westbury Lab Wikipedia Corpus, Edmonton, AB: University of Alberta.

[31] Rupesh Kumar Srivastava, Klaus Greff, and JÄijrgen Schmidhuber. 2015. Highway Networks. *arXiv:1505.00387* (2015).

[32] Ilya Sutskever, James Martens, and Geoffrey E. Hinton. 2011. Generating Text with Recurrent Neural Networks. In *ICML*. 1017–1024.

[33] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*. 3104–3112.

[34] R. Sutton, D. McAllester, S. Singh, and Y. Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*. 1057–1063.

[35] Rui Yan. 2016. i, Poet: Automatic Poetry Composition through Recurrent Neural Networks with Iterative Polishing Schema.. In *IJCAI*. 2238–2244.

[36] Rui Yan. 2018. "Chitty-Chitty-Chat Bot": Deep Learning for Conversational AI.. In *IJCAI'18*. 5520–5526.

[37] Lili Yao, Nanyun Peng, Weischedel Ralph, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-And-Write: Towards Better Automatic Storytelling. In *AAAI'19*.

[38] Lili Yao, Yaoyuan Zhang, Yansong Feng, Dongyan Zhao, and Rui Yan. 2017. Towards implicit content-introducing for generative short-text conversation systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2190–2199.

[39] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *AAAI*. 2852–2858.

Outage Prediction and Diagnosis for Cloud Service Systems

Yujun Chen*
chenjohn@buaa.edu.cn
Beihang University, Microsoft Research, Beijing, China

Hongyu Zhang
hongyu.zhang@newcastle.edu.au
The University of Newcastle, Callaghan, Australia

Xian Yang, Qingwei Lin, Dongmei Zhang,
Hang Dong, Yong Xu, Hao Li, Yu Kang
{xian.yang,qlin,domgmeiz,v-hadon,Yong.Xu,v-lihao,
kay}@microsoft.com
Microsoft Research, Beijing, China

Feng Gao, Zhangwei Xu, Yingnong Dang
{fgao,zhangxu,Dang.Yingnong}@microsoft.com
Microsoft Azure, Redmond, USA

ABSTRACT

With the rapid growth of cloud service systems and their increasing complexity, service failures become unavoidable. Outages, which are critical service failures, could dramatically degrade system availability and impact user experience. To minimize service downtime and ensure high system availability, we develop an intelligent outage management approach, called AirAlert, which can forecast the occurrence of outages before they actually happen and diagnose the root cause after they indeed occur. AirAlert works as a global watcher for the entire cloud system, which collects all alerting signals, detects dependency among signals and proactively predicts outages that may happen anywhere in the whole cloud system. We analyze the relationships between outages and alerting signals by leveraging Bayesian network and predict outages using a robust gradient boosting tree based classification method. The proposed outage management approach is evaluated using the outage dataset collected from a Microsoft cloud system and the results confirm the effectiveness of the proposed approach.

CCS CONCEPTS

• **Software and its engineering → Software maintenance tools**; *Maintaining software*; *System administration.*

KEYWORDS

Outage prediction, outage diagnosis, cloud system, system of systems, service availability

ACM Reference Format:
Yujun Chen, Xian Yang, Qingwei Lin, Honyu Zhang, Feng Gao, Zhangwei Xu, Yingnong Dang, Domgmei Zhang, Hang Dong, Yong Xu, Hao Li, Yu Kang. 2019. Outage Prediction and Diagnosis for Cloud Service Systems. In Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313501

*Work done during internship at Microsoft Research.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313501

1 INTRODUCTION

A typical cloud system is a system of systems, providing different services such as networking, storage, computation, security and management. Cloud providers, such as Microsoft, Amazon, Google, and IBM, aim at fast delivery of computing resources in a dynamically scalable and virtualized manner [3, 25]. Many cloud services could be hosted in a cloud system and each service itself is a large and complex system that consists of many components. Failures would happen in a complicated system due to frequent updates of components, changes in operation environment, online repairs, and mobility of devices, [10, 12, 27]. Failures could dramatically degrade system availability and lead to bad user experience. To manage failures, various system monitors and alerting tools are deployed at different places of a cloud service system to detect if a service performs well or not.

In cloud systems, outages are critical system failures that can lead to system unavailability. When an outage is detected, the management tool is expected to automatically notify, mitigate, and diagnose the outage. In literature, there is a large amount of work on predicting and diagnosing failures in a large and complex system such as a data center, grid system, and defense system [14, 15, 20, 30]. Such an ability can help prevent potential disasters and minimize damages caused by system unavailability. However, they only consider a single system and ignore the related systems that could have an impact on the prediction results. In this work, we are interested in predicting and diagnosing outages of a cloud system. A typical cloud system contains many sub-systems (i.e., services), each of which consists of many interconnected components. Each component has its own monitors that regularly check the runtime status of the component. Signals from components reflect different aspects of system health status, such as individual cloud resource, node/data center traffic volume, response latency, temperature, and power consumption. While component-level alerting signals are useful, it is important to have a global watcher for the entire cloud system which understands the topology, resiliency models and dependencies of the system. After connecting all the signals, the global watcher can proactively monitor and correlate service health issues across the whole system without manually defined rules.

In our work, we propose an intelligent outage management tool, called AirAlert, as a global watcher of the whole cloud system. AirAlert collects all alerting signals across the whole cloud system, and utilizes them to diagnose and predict outages. The outages we

try to predict come from two levels: component-level and service-level. These two concepts are hierarchical, where service-level outage consists of various corresponding component-level outages. The service-level outage prediction can help locate the suspicious behavior in the general system, also, locating which component is responsible for the outage can better alleviate the cost of diagnosing and debugging. To diagnose where outages come from, we adopt a Bayesian network approach and investigate dependency relationships between signals and outages. We also construct predictive models to achieve robust outage prediction.

In our work, we are investigating a large-scale Microsoft cloud system. As there are a large number of altering signals, it is hard to know beforehand the dependencies among the signals and their relationships with outages. Therefore, a comprehensive global watcher, specially designed for diagnosing and alerting failures, is needed. In our approach, Bayesian network is used to build the signal dependencies from historical failure statistics. For outage prediction, many algorithms can be used for constructing a prediction model, such as time-series forecasting method (e.g., auto-regressive moving average), rule-based methods (e.g., frequent event set mining [6, 24]), and supervised machine learning methods [8, 17]. However, traditional methods are not capable of handling outage prediction in a cloud service system as they ignore interactions among different system components. There are several challenges in constructing an effective predictive model. The foremost one is that we have very imbalanced datasets due to the limited number of outage cases. Therefore, we apply sampling techniques to preprocess the data and then use the gradient boosting tree based classification method for robust model construction.

Our approach has been evaluated using a set of outage data collected from the Microsoft cloud system over an one year period. We compare the AirAlert approach against other methods and obtain good performance in outage prediction. Also, we show that AirAlert can help outage diagnosis and present several real-world cases.

The contributions of the paper are as follows:

- We propose an outage prediction and diagnosis approach for a cloud service system. Our model takes into consideration multiple related services and components in the cloud system.
- We can infer causal relationships among outages and alerting signals, which can help us understand where the outage is from and which components work cooperatively.
- We can predict whether a component or service will have an outage in the near future, which would potentially help engineering teams alleviate the influence of outages as quickly as possible.
- We have evaluated the proposed approach using an one-year outage dataset collected from a Microsoft cloud system, and the results confirm the effectiveness of the proposed approach.

The remaining sections of this paper are organized as follows. Section 2 describes the related work of the study. Section 3 presents the proposed outage prediction and diagnosis approach and Section 4 describes the experiments. Section 5 concludes our work.

2 RELATED WORK

Failures occurred in online systems could degrade system performance and impact user experience. They are often defined as 'an event that occurs when the delivered service deviates from correct service' [1]. Traditional work on failure management for online systems deals with failures after they have happened. Recently, there are increasing efforts on proactively predicting failures to prevent potential disasters or minimize damages [20]. For internet service provider networks, there are intensive work on failure predictions (e.g., [11, 13, 21, 22]). For example, in [30], switch failures in data center networks are predicted using signals reflecting switch system current status and data containing curated historical hardware failure cases. Input signals of the predictive model are mostly system event logs, such as changes in configuration, interface, device working mode and operational maintenance. Similarly, in [9], network failures in data centers are predicted using data recording network errors related to device and link failures. There are also many studies on investigating the failure characteristics [19, 23, 29] of high performance computing systems (HPCs). In [18], failures occurred in a 350-node cluster system is predicted using various system reliability, availability and service ability (RAS) logs containing health related events. In [7], a failure prediction framework for HPCs is designed by exploring correlations among failures and forecasting the time-between-failure. In [26], a Bayesian network based fault diagnosis tool is developed for managing a defense system-of-systems (SoS), which collects signals from a sensor, network, command and control systems. Bayesian network is used to find the root causes of system failures by using the network topology of the whole SoS. In our work, we use the Bayesian network to find the relationships among alerting signals and outages of cloud systems and then we use XGBoost model for prediction.

The work in [10] studies proactive failure management for cloud systems. It uses Bayesian models and decision trees to proactively predict failure dynamics. Failures are regarded as abnormal signals detected by monitors and failure prediction is tackled from an anomaly detection point of view. Different from their works which focus on predicting failures based on raw signals, our work uses the failure signals from monitors (which we call, *alerting signals*) to predict critical failures (which we call, *outages*). In current real-time system, outage prediction is an important issue that bothers many systems. The reason lies in several aspects: outages are critical system failures that could lead to severe consequences; outages occur without a significant alerting signals pattern, which makes it harder to predict; and the scope of impacting signals is a complex process to define.

3 PROPOSED APPROACH

In this section, we introduce our proposed approach, which is called AirAlert. We first give definitions for our task. Then, we introduce the Bayesian network for generating relationships between alerting signals and outages as well as the gradient boosting tree classification method for outage prediction.

3.1 Definitions

Alerting Signal A single alerting signal A^i can be represented by $A^i = [A^i_1, A^i_2, ..., A^i_T]$, where each component A^i_t indicates the

strength of this signal at time $t \in (1, T)$. For all alerting signals, we denote a multivariate time series of length T as $A = [A_1, A_2, ..., A_T]' \in \mathbb{R}^{D \times T}$ for all signals, where for each $t \in \{1, 2, ..., T\}$, and $A_t = [A_t^1, A_t^2, ..., A_t^D] \in \mathbb{R}^D$ represents the observations of all the alerting signals at time t.

Outage Sequence The outage sequence is a binary time series of length T as $O = (O_1, O_2, ..., O_T) \in \mathbb{R}^T$, where for each $t \in \{1, 2, ..., T\}$, $O_t \in \{1, 0\}$ indicating whether an outage happens at time t or not.

Outage Prediction. Outage prediction at time t is a classification task. We want to maximize the probability $Pr(O_t|\boldsymbol{\mathcal{A}}_t)$, which is predicting accuracy of O_t given the input feature $\boldsymbol{\mathcal{A}}_t$. The input feature $\boldsymbol{\mathcal{A}}_t = [A_t^1, A_t^2, ..., A_t^m]$ are the m alerting signals used for prediction, where $m \leq D$.

3.2 Bayesian network for outage diagnosis

A cloud system is an online system of systems, where the occurrence of an outage is associated with a combination of alerting signals. For example, if a particular web application encounters an outage, it is normally resulted from multiple failures occurred in different services such as networking, hardware and DNS server, rather than from one single failure of the web application component . To be more specific, when several alerting signals are observed, the chance of having an outage depends on all alerting signals. However, it is hard to enumerate all possible combinations of alerting signals and determine which set of signals is related to an outage. To better model the problem, we used a Bayesian network inference method to detect the relationship between alerting signals and outages. Here, we use the FCI-algorithm [5] to infer our Bayesian network. The fundamental idea of FCI algorithm is to build a directed acyclic graph (DAG), where each node X_i represents an alerting signal or an outage based on the causal Markov assumption. The FCI algorithm can be used for the connectivity inference and orientation determination. In this paper, we use it to generate the connectivity between the alerting signals and outages rather than infer the direction.

In our method, the conditional dependence between the alerting signal and outage given a set of other alerting signals are obtained by calculating the Pearson correlation. The influence of the conditional signals needs to be regressed out first. Then, the Fisher's z-transform is performed as in [5]. For example, given the time series sequence of an alerting signal A_i and an outage O_i, and the conditional set only contains the alerting signal A_{i_2}, the correlation between A_i and O_i given A_{i_2} is:

$$\begin{aligned} r &= \frac{cov(A_i|A_{i_2}, O_i|A_{i_2})}{\sigma_{A_i|A_{i_2}}\sigma_{O_i|A_{i_2}}} \\ &= \frac{\sum_{t=1}^{T}(A_{i|i2} - \bar{A}_{i|i2})(O_{i|i2} - \bar{O}_{i|i2})}{\sqrt{\sum_{t=1}^{T}(A_{i|i2} - \bar{A}_{i|i2})^2}\sqrt{\sum_{t=1}^{T}(O_{i|i2} - \bar{O}_{i|i2})^2}}. \end{aligned} \tag{1}$$

Then, Fisher-z transform is used as follows to generate the score for testing the significance of the correlation value:

$$z = \frac{1}{2}\ln(\frac{1+r}{1-r}). \tag{2}$$

z is the correlation score for the alerting signal A_i and outage O_i give A_{i_2}. The significant test of z is then used to check whether the independence assumption can be accepted that A_i is independent of O_i given A_{i_2}.

The FCI-algorithm is an approximation method to obtain the network structure by calculating conditional dependence. It adopts a recursive inference process and uses the Fisher-z conditional independence test to capture all the independence possibilities among all alerting signals and outages. A bootstrap method for stable result generation is used here [16]. After using FCI-algorithm, the skeleton of the causal network among alerting signals and outages can be obtained. In our work, the Bayesian network mainly works as a diagnositic tool to infer the relationship between the alerting signals and the outage. We can also use it to select the most relevant features and feed them as the inputs of the outage prediction model.

3.3 Gradient boosting tree for outage prediction

In practical online systems, conventionally outage prediction is performed using rule-based methods [20]. Despite the complex pattern of outage, each component or service only focuses on their own alerting signals. For each component or service, engineers will simply set up a monitor to predict outage by examining whether the strength of the alerting signal exceeds a certain threshold θ:

$$P(t) = \begin{cases} 1 & \text{if} A_t > \theta \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where A_t is the observation of alerting signals reported from a certain component at time t, and the value of θ for each team is set based on human knowledge. Such rule-based outage prediction method is normally effective for in-system outages, as this type of outages only caused by malfunction of a component .

In cloud systems, outages are often caused by cross-service malfunctions simultaneously. Thus, using a single signal of a component or service will not be very effective. As a result, we use the alerting signals A_t at time t as the input feature for outage prediction. At a certain time slot t, several alerting signals $\boldsymbol{\mathcal{A}}_t$ will be collected to predict whether the outage will happen at this time. To construct a robust predictor, we use the gradient boosting tree based model (XGBoost) [4]. XGBoost is fundamentally a regression tree that has the same decision rule as the decision tree model. But the model ensembles a set of classification and regression trees (CART). In the CART tree, each node is one of the alerting signals. The prediction result is the sum of scores predicted by K decision trees, as:

$$\hat{y}_t = \sum_{k=1}^{K} f_k(\boldsymbol{\mathcal{A}}_t), f_k \in F \tag{4}$$

where $f_k(\cdot)$ is the score from the kth tree, F is the function space containing all regression trees, and $\hat{y}_t$ contains prediction results at time t. The XGBoost is optimized to achieve the best prediction results by minimizing the following loss:

$$\mathcal{L} = \sum_{t=1}^{T} l(y_t, \hat{y}_t) + \lambda \sum_{k=1}^{K} \Omega(||f_k||) \tag{5}$$

where the first term $\sum_{t=1}^{T} l(y_t, \hat{y}_t)$ is the summation of cross-entropy loss measuring whether the classification model and features can

well perform for the prediction task across all timestamp. The second term $\lambda \sum_{k=1}^{K} \Omega(||f_k||)$ penalizes the tree boosting parameters f_k in the model as described in [28].

4 EXPERIMENTS

In this section, we carry out extensive experiments using data collected from six representative services across tens of data centers over one year period from the Microsoft cloud system. Due to the privacy policy of Microsoft, we deliberately masked sensitive data throughout this paper. We first show the results of predicting outages at component and service levels, where AirAlert has two prediction modes: AirAlert Related and AirAlert Full. In the AirAlert Related mode, only outage dependent signals found in the Bayesian network are used for prediction, while in the AirAlert Full mode all signals are used for prediction. Then, we give case studies to show how our work can help to diagnose outages and find which team to handle the outage.

4.1 Outage prediction

In this subsection we investigate the performance of predicting the component-level outages and also the service-level outages. These experiments aim at investigating the performance of our method for predicting outages of different complex levels. The component-level outage is the outage coming from a specific component . As a service would contain multiple components, the service-level outage is more complicated and heterogeneous.

4.1.1 Experiment Setups. The investigated component-level outages come from three most representative components: the Storage Location, Physical Networking and Storage Streaming component. The service-level outages come from three most representative services: the Web Application, Cloud Network and the Microsoft Cloud System Operation Service. We evaluate the method by collecting outages over one year period from tens of data centers. We acquired data at the time step of one hour and obtained over 8,000 samples in total (24hrs*365days). The period that the data covered only has a very small number of outages. For evaluation, we choose the service and component outages that occurred frequently in the last year (that is why we choose three most representative services/components). For other services/components, which rarely have any outages happened, there is no sufficient data to construct a predictive model. As we only have a limited number of outages , classification methods would suffer from the imbalanced data problem. To better deal with the imbalanced data, the **SMOTE** [2] over-sampling strategy is used for generating the training data from system database so that the positive and negative samples in the training phase can be balanced.

Given alerting signals across the cloud system, we predict whether a component or service outage will happen. We use precision, recall and F1 as the evaluation metrics. Five different outage prediction methods are compared, which are:

- Simple Spike: This is a rule-based outage prediction method as described in Equation 3, where the threshold θ is predefined by domain knowledge and could vary across different components or services . If the strength of the alerting signal is larger than θ, an outage will happen.
- Support Vector Machine (SVM): SVM is commonly used in classification problems. In this experiment, all the alerting signals are served as input features for the SVM classifier with linear kernel.
- Penalized Logistic Regression (PLR): PLR is commonly used in feature selection and prediction with the introduction of sparse constraints. In this experiment, all alerting signals are used as input features for the PLR model.
- AirAlert Related: It first applies the Bayesian network method to find the most relevant alerting signals for outage prediction, which are signals in the Bayesian network directly connected to the outage. Then, we use these selected signals for outage prediction using the Xgboost classification method.
- AirAlert Full: It is based on the XGBoost classifier. Different from 'AirAlert Related', this approach uses all alerting signals as input features.

AirAlert Related and AirAlert Full are two modes of our method. Users can choose which mode they want to use based on their needs. When they want to use all signals for prediction, they can choose the full mode. However, for the dataset with limited number of samples but large number of signals, full mode would easily return an overfitted model. Therefore, we provide the AirAlert Related mode, where only the signals selected by Bayesian network are fed into the gradient boosting tree model.

4.1.2 Results. The results for the component-level and service-level prediction are shown in Table 1 and 2 respectively.

Table 1 shows the outage prediction results at the component-level. Here, we select three most representative and frequently occurred component-level outages for performance evaluation, which are related to Storage Location, Physical Networking and Storage Streaming. All five different prediction methods are compared in terms of precision, recall and F1. We can see that their performance results are quite similar. Simple spike, which is a straightforward rule-based outage prediction method, can achieve 100% recall in some cases for component-level outage prediction. We will later show its performance in predicting service-level outages, which are more complex and involving many heterogeneous signals. Unlike other methods using the whole feature set for prediction, AirAlert Related only uses the Bayesian network selected features. With reduced feature set size, the performance can be still maintained. This observation shows that Bayesian network can help us find the most representative and dependent alerting signals. Other signals which are not found to be highly related to outages in the network would not significantly improve the performance.

Table 2 shows the outage prediction results at the service level. Again, three most representative and frequently happened outages are selected, which are from Website Application, Cloud network and Microsoft Cloud System Operation. Different from the results in Table 1, we can see that the performance of different prediction methods varies greatly. Among them, Simple Spike obtains very low precision and recall. This is because the service-level outage is more complex than the component-level outage and simple rule-based method cannot work for complicated cases. The outage from a single service can be heterogeneous. For example, outages coming

Table 1: Comparison of different methods for component-level outage prediction.

	Outage (Storage Location)			Outage (Physical Networking)			Outage (Storage Streaming)		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Simple Spike	61.65	100.00	76.28	73.71	67.71	70.58	61.52	100.00	76.18
PLR	70.02	92.71	79,78	67.72	83.33	74.72	63.23	91.67	74.84
SVM	65.65	95.83	77.92	63.13	88.54	73.71	58.62	88.64	70.57
AirAlert Related	65.31	100.00	79.01	63.33	98.95	77.25	62.34	100.00	76.80
AirAlert Full	71.11	100.00	83.17	69.07	100.00	81.71	63.75	98.99	77.86

Table 2: Comparison of different methods for service-level outage prediction.

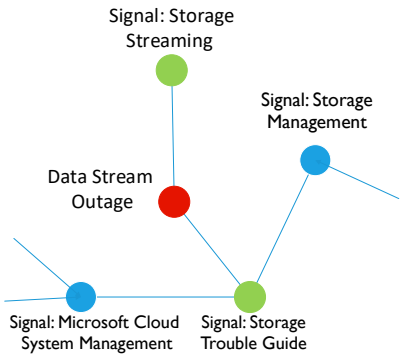
	Outage (Website Application)			Outage (Cloud Network)			Outage (Microsoft Cloud System Operation)		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Simple Spike	5.73	11.83	7.72	4.47	67.74	8.39	7.27	29.03	11.63
PLR	61.18	54.17	57.46	26.27	60.52	36.64	20.36	35.17	25.79
SVM	66.41	88.54	75.89	6.89	88.42	12.78	26.90	22.50	24.50
AirAlert Related	92.18	85.63	88.78	62.08	47.65	53.92	72.40	77.96	75.08
AirAlert Full	82.75	76.74	79.63	75.93	67.07	71.22	72.59	50.15	59.32

from the "website application" service may result from network or hardware failures.

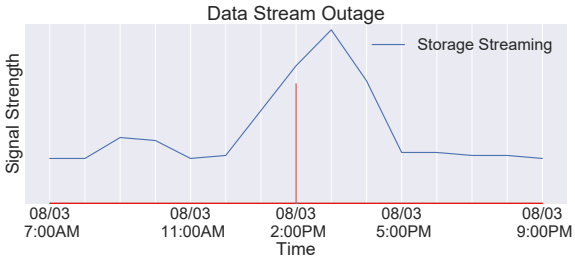
From Table 2 we can further observe that different from Simple Spike, PLR and SVM, AirAlert Related and AirAlert Full have more stable results across different outages. This observation tells us that XGBoost is more robust than others. Consistent with Table 1, the performance of AirAlert Related using Bayesian network selected features has similar performance as AirAlet Full, which uses the whole dataset. More specifically, for the first and the third outage, using related alerting signals about 10% gain in both precision and recall can be achieved. With the help of the Bayesian network, not only the most relevant alerting signals can be identified but also the most predictive features can be detected. The most relevant signals are the ones which would directly lead to outages, while the most predictive features are the ones which contribute to the predictive model the most.

4.2 Outage diagnosis

4.2.1 Cases for outage diagnosis. As mentioned in the previous section, Bayesian network can specify the root cause of outages by calculating the conditional dependence. Here, we would like to give two examples of Bayesian network results. Fig. 1 and Fig. 2 show component-level and service-level outage diagnosis results, respectively. In Fig. 1, subplot (a) contains the Bayesian network for diagnosing relevant alerting signals for the outage called 'Data Stream Outage'. In the graph, the studied outage is in red; its directly linked nodes are in green; and the two-hop neighbours are in blue. This figure shows that the 'Storage Streaming component signal' and the 'Storage Trouble Guide component signal' are most relevant to the occurrence of the 'Data Stream Outage'. When we look up records saved in the Microsoft cloud system, we find that engineer team manually diagnosed the causes of the Data Stream outage

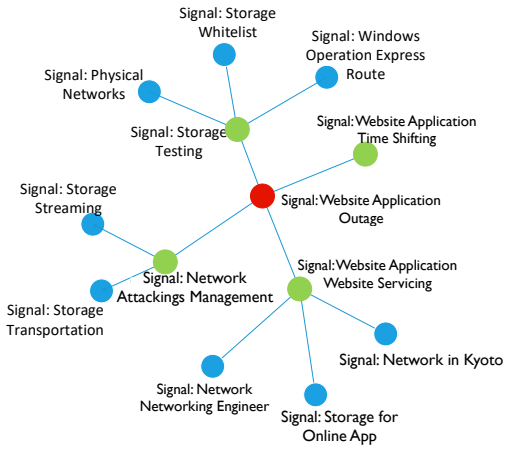


(a) Bayesian network for 'Data Stream Outage'.

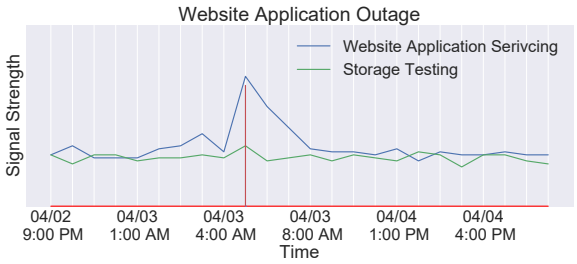


(b) Time series of the 'Storage Streaming' Signal.

Figure 1: Diagnosis results for a component-level outage named 'Data Stream Outage'.



(a) Bayesian network for 'Web Application Outage'.



(b) Time series of the 'Web Application Services' and 'Storage Testing' Signals.

Figure 2: Diagnosis results for a service-level outage named 'Web Application Outage'.

based on their knowledge. They found that the Storage Streaming signal is the top first cause, which is consistent with our findings from Bayesian network. Fig. 1 (b) presents the time-series change for alerting signals, where there was one 'Data Stream Outage' happened at 2:00 PM Aug 3rd. At that time, the related 'Storage Streaming signal' rose significantly, showing the findings in the Bayesian network are quite reasonable.

In Fig. 2, we show an example of diagnosing a service-level outage, called 'Web Application Service Outage'. Fig. 2 (a) tells us how the alerting signals are related to this service-level outage. Comparing Fig. 2 (a) with Fig. 1 (a), we can see that Fig. 2 (a) has more interconnected nodes, which indicate that the service-level outage is more complex than component-level outages and thus has more relevant alerting signals across the system. The signals that directly linked to the 'Website Application Service Outage' are 'Storage testing signal', 'Web Application Time Rotation signal', 'Network attacking Management signal' and 'Web Application Website Servicing signal'. From the records of the Microsoft cloud system, The engineering team has manually identified a large number of Web Application Service outages that are related to 'Web Application Website Servicing', which matches our results well. In Figure 2 (b), we also show the changes of some alerting signals over time, where one 'Web Application Service Outage' happened at 5:00 AM April 3rd. At that time, both 'Web Application Website Servicing signal' and 'Storage testing signal' had a significant jump, which is consistent with the observation in Figure 2 (a). These two examples show that our work on Bayesian network can be a useful outage diagnosis tool that helps the engineering team locate the root cause of outages.

4.2.2 Cases for outage assignment. In a real cloud system, determining which team to handle each outage is a non-trivial task. If we correctly predict the outage to be the 'TeamA Outage', the system will automatically assign the *TeamA* team to handle this outage. Here, we would like to give two examples to show the potential of using our method to assign outages. In these examples, the cloud system first wrongly assigns an outage to a team and then reassigns the outage to the correct team sometime later. However, AirAlert correctly predicted which part of the cloud would have the outage and indicated the correct responsible team who should handle it. There are two instances of the 'Location Servicing Outage' happened on June 7th at 16:00 and July 19th at 16:50, respectively. When we look at the operation record in the Microsoft cloud system, this outage was first assigned to the 'Windows Cloud System Live Site' team and then transferred to the 'Location Servicing' team. If we use our prediction and diagnosis approach, we find that we can correctly predict in advance that an outage would happen in 'Location Servicing Outage'. From the Bayesian network diagnosis results, we can find alerting signals related to 'Location Servicing Outage'. By investigating the time series changes of those related signals, we can see that they increased obviously before these two instances of 'Location Servicing Outage'.

5 CONCLUSION

In this paper, we studied the outage diagnosis and prediction problem in cloud systems. We proposed an intelligent outage management method called AirAlert. In contrast to the previous work focusing on dealing with failures in a single service, we work on outages that would come from many parts of the whole cloud system, which is a system of systems. AirAlert works as a global watcher of the cloud system by collecting alerting signals across the whole system. The dependence relationships among the alerting signals and outages are discovered based on the Bayesian network method. AirAlert can diagnose where the outages come from to help us identify the root causes of outages. For outage prediction, we use the gradient boosting tree based model and compare the results from both component-level and service-level outage predictions. To the best of our knowledge, this is the first work that uses alerting signals across the whole cloud system to diagnose and predict outages coming from different levels.

Acknowledgement

We would like to thank the anonymous reviewers for their helpful and constructive comments. The work was supported by the National Natural Science Foundation of China (Project No. 61702107 and 61828201).

REFERENCES

[1] Algirdas Avizienis, J-C Laprie, Brian Randell, and Carl Landwehr. 2004. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions*

on Dependable and Secure Computing 1, 1 (2004), 11–33.

[2] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* 16 (2002), 321–357.

[3] Junjie Chen, Xiaoting He, Qingwei Lin, Yong Xu, Hongyu Zhang, Dan Hao, Feng Gao, Zhangwei Xu, Yingnong Dang, and Dongmei Zhang. 2019. An Empirical Investigation of Incident Triage for Online Service Systems. In *Proceedings of the 41st ACM/IEEE International Conference on Software Engineering*. to appear.

[4] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 785–794.

[5] Diego Colombo, Marloes H Maathuis, Markus Kalisch, and Thomas S Richardson. 2012. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics* (2012), 294–321.

[6] Carlotta Domeniconi, Chang-Shing Perng, Ricardo Vilalta, and Sheng Ma. 2002. A classification approach for prediction of target events in temporal sequences. In *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 125–137.

[7] Song Fu and Cheng-Zhong Xu. 2007. Exploring event correlation for failure prediction in coalitions of clusters. In *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing*. 41.

[8] Song Fu and Cheng-Zhong Xu. 2010. Quantifying event correlations for proactive failure management in networked computing systems. *J. Parallel and Distrib. Comput.* 70, 11 (2010), 1100–1109.

[9] Phillipa Gill, Navendu Jain, and Nachiappan Nagappan. 2011. Understanding network failures in data centers: measurement, analysis, and implications. In *ACM SIGCOMM Computer Communication Review*, Vol. 41. ACM, 350–361.

[10] Qiang Guan, Ziming Zhang, and Song Fu. 2012. Ensemble of bayesian predictors and decision trees for proactive failure management in cloud computing systems. *Journal of Communications* 7, 1 (2012), 52–61.

[11] Guenther Hoffman and Miroslaw Malek. 2006. Call availability prediction in a telecommunication system: A data driven empirical approach. In *2006 25th IEEE Symposium on Reliable Distributed Systems (SRDS'06)*. IEEE, 83–95.

[12] Qingwei Lin, Ken Hsieh, Yingnong Dang, Hongyu Zhang, Kaixin Sui, Yong Xu, Jian-Guang Lou, Chenggang Li, Youjiang Wu, Randolph Yao, et al. 2018. Predicting Node failure in cloud service systems. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM, 480–490.

[13] Qingwei Lin, Jian-Guang Lou, Hongyu Zhang, and Dongmei Zhang. 2016. iDice: problem identification for emerging issues. In *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*. IEEE, 214–224.

[14] Qingwei Lin, Hongyu Zhang, Jian-Guang Lou, Yu Zhang, and Xuewei Chen. 2016. Log clustering based problem identification for online service systems. In *Proceedings of the 38th International Conference on Software Engineering Companion*. ACM, 102–111.

[15] Debra Greenhalgh Lubas. 2017. Department of defense system of systems reliability challenges. In *Reliability and Maintainability Symposium (RAMS), 2017 Annual*. 1–6.

[16] Sara Magliacane, Tom Claassen, and Joris M Mooij. 2016. Ancestral causal inference. In *Advances in Neural Information Processing Systems*. 4466–4474.

[17] James W Mickens and Brian D Noble. 2006. Exploiting Availability Prediction in Distributed Systems. In *Symposium on Networked Systems Design and Implementation (NSDI)*, Vol. 6. 73–86.

[18] Ramendra K Sahoo, Adam J Oliner, Irina Rish, Manish Gupta, José E Moreira, Sheng Ma, Ricardo Vilalta, and Anand Sivasubramaniam. 2003. Critical event prediction for proactive management in large-scale computer clusters. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 426–435.

[19] Ramendra K Sahoo, Mark S Squillante, Anand Sivasubramaniam, and Yanyong Zhang. 2004. Failure data analysis of a large-scale heterogeneous server environment. In *Dependable Systems and Networks, 2004 International Conference on*. IEEE, 772–781.

[20] Felix Salfner, Maren Lenk, and Miroslaw Malek. 2010. A survey of online failure prediction methods. *ACM Computing Surveys (CSUR)* 42, 3 (2010), 10.

[21] Felix Salfner and Miroslaw Malek. 2007. Using hidden semi-Markov models for effective online failure prediction. In *Reliable Distributed Systems, 26th IEEE International Symposium on*. 161–174.

[22] Felix Salfner and Steffen Tschirpke. 2008. Error Log Processing for Accurate Failure Prediction. In *First USENIX Workshop on the Analysis of System Logs, WASL 2008*. USENIX.

[23] Bianca Schroeder and Garth Gibson. 2010. A large-scale study of failures in high-performance computing systems. *IEEE Transactions on Dependable and Secure Computing* 7, 4 (2010), 337–350.

[24] Ricardo Vilalta and Sheng Ma. 2002. Predicting rare events in temporal domains. In *2002 IEEE International Conference on Data Mining*. IEEE, 474–481.

[25] Asher J Vitek and MN Morris. 2012. Service oriented cloud computing architectures. In *UMM CSci Senior Seminar Conference*. Technical Report.

[26] Hongbing Wang, Lei Wang, Qi Yu, Zibin Zheng, Athman Bouguettaya, and Michael R Lyu. 2017. Online reliability prediction via motifs-based dynamic bayesian networks for service-oriented systems. *IEEE Transactions on Software Engineering* 43, 6 (2017), 556–579.

[27] Yong Xu, Kaixin Sui, Randolph Yao, Hongyu Zhang, Qingwei Lin, Yingnong Dang, Peng Li, Keceng Jiang, Wenchi Zhang, Jian-Guang Lou, et al. 2018. Improving service availability of cloud systems by predicting disk error. In *2018 USENIX Annual Technical Conference (USENIX ATC 18)*. 481–494.

[28] Zhixiang Xu, Gao Huang, Kilian Q Weinberger, and Alice X Zheng. 2014. Gradient boosted feature selection. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 522–531.

[29] Praveen Yalagandula, Suman Nath, Haifeng Yu, Phillip B Gibbons, and Srinivasan Seshan. 2004. Beyond Availability: Towards a Deeper Understanding of Machine Failure Characteristics in Large Distributed Systems. In *First USENIX Workshop on Real, Large Distributed Systems, WORLDS'04*. USENIX.

[30] Shenglin Zhang, Ying Liu, Weibin Meng, Zhiling Luo, Jiahao Bu, Sen Yang, Peixian Liang, Dan Pei, Jun Xu, Yuzhi Zhang, et al. 2018. PreFix: Switch Failure Prediction in Datacenter Networks. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 2, 1 (2018), 2.

What Makes a Good Team? A Large-scale Study on the Effect of Team Composition in Honor of Kings

Ziqiang Cheng
Zhejiang University

Yang Yang*
Zhejiang University

Chenhao Tan
University of Colorado Boulder

Denny Cheng
Tencent Corporation

Alex Cheng
Tencent Corporation

Yueting Zhuang
Zhejiang University

ABSTRACT

Team composition is a central factor in determining the effectiveness of a team. In this paper, we present a large-scale study on the effect of team composition on multiple measures of team effectiveness. We use a dataset from the largest multiplayer online battle arena (MOBA) game, Honor of Kings, with 96 million matches involving 100 million players. We measure team effectiveness based on team performance (whether a team is going to win), team tenacity (whether a team is going to surrender), and team rapport (whether a team uses abusive language). Our results confirm the importance of team diversity with respect to player roles, and show that diversity has varying effects on team effectiveness: although diverse teams perform well and show tenacity in adversity, they are more likely to abuse when losing than less diverse teams. Our study also contributes to the situation vs. personality debate and show that abusive players tend to choose the leading role and players do not become more abusive when taking such roles.

CCS CONCEPTS

• **Applied computing → Law, social and behavioral sciences**.

KEYWORDS

team composition, team performance, tenacity, toxic behavior, MOBA

ACM Reference Format:
Ziqiang Cheng, Yang Yang*, Chenhao Tan, Denny Cheng, Alex Cheng, and Yueting Zhuang. 2019. What Makes a Good Team? A Large-scale Study on the Effect of Team Composition in Honor of Kings. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313530

1 INTRODUCTION

The increasing complexity and scale of tasks in modern society require individuals to work together as a team [13, 40]. Crucial to the effectiveness of a team are the individuals in the team, i.e., team composition. Extensive prior research has studied team roles to analyze team composition [6, 7, 30, 31, 36, 38]. The most influential work is Belbin's team role framework, also known as Belbin Team Inventory. A central hypothesis in Belbin's framework is that balance in team roles is associated with team performance. Belbin [7] defines nine roles[1], and finds that teams with certain role combinations result in poor performance, even if they are formed by members with the sharpest mind and the most experience. However, team roles are usually implicit in real life and it is difficult to identify such nine roles in most contexts.

Another challenge in studying the effect of team composition on team effectiveness arises from the definition of *effectiveness*. Team performance is the most straightforward definition and has been studied in a battery of studies [10, 13, 15, 35]. Example measures of team performance include the impact of published papers from a team [40], winning a sports game [4, 24], etc. However, the effectiveness of a team can also be reflected by the tenacity in face of adversity, and the rapport between team members [9]. The effect of team composition likely depends on the definition of team effectiveness. Therefore, it is important to understand the effect of team composition on the effectiveness of teams in multiple measures.

To address these two challenges, we identify multiplayer online battle arena (MOBA) as an ideal testbed and provide a large-scale study on the effect of team compositions on multiple measures of team effectiveness. We use a dataset from Honor of Kings, the largest MOBA game in the world. This platform is ideal for studying the effect of team composition for three reasons. First, there are defined roles in the game and players choose their roles to form a team. Given that there are five players in each team, we are able to enumerate all possible team compositions. We also have information about the characteristics of each role. Second, the popularity of Honor of Kings leads to digital traces of hundreds of millions of players in hundreds of millions of games. These detailed game records allow us to explore the notion of team "effectiveness" beyond the simple measure of team performance, winning or losing. For instance, we study the effect of team composition on abusive language use, which reflects the rapport in a team. This research question naturally connects to the literature on toxic behavior in online communities and gaming [11, 12, 26]. Finally, we have access to the past history of players for capturing their background and experience. Historical information enables us to investigate questions in the "situation vs. personality" debate [16, 22]: how much of our behavior is determined by fixed personality traits or by the specific situation at hand? We will explore this question in the context of toxic behavior in online gaming.

Organization and paper highlights. In this paper, we conduct a large-scale study on the effect of team composition in the context of online gaming, particularly, in Honor of Kings. Our dataset comes

*Corresponding author: Yang Yang, yangya@zju.edu.cn.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313530

[1] See https://en.wikipedia.org/wiki/Team_Role_Inventories for a quick explanation.

Statistics	Number(fraction/SEM)
Overall statistics	
#ranked games	54, 434, 817
#team compositions	126
#surrender games	5, 807, 005 (10.7%)
#abusing games	30, 668, 577 (56.3%)
#early games($t \leq 11$ min)	5, 049, 934 (9.3%)
#middle games($11 < t \leq 16$ min)	23, 089, 993 (42.4%)
#late games($t > 16$ min)	26, 294, 890 (48.3%)
Top three frequent combinations	
– – – –	16, 186, 003 (14.9%)
– – – –	14, 516, 443 (13.3%)
– – – –	12, 630, 805 (11.6%)
Personal statistics	
#players	1, 306, 754
#players who played ranked games	1, 173, 372
#ranked games each player has	48.4 (0.05)
average winrate	52.4% (1.0×10^{-4})
average abusing probability	9.3% (1.0×10^{-4})
average surrender probability	5.0% (6.9×10^{-5})
#target players who have played at least 20 ranked games	758, 494 (58.0%)

Table 1: Statistics of the dataset.

from this popular MOBA game in China, and includes 96 million games and 100 million players in total. We provide details of the dataset and character roles in §2.

We quantitatively study the effect of team composition on three distinct definitions of team effectiveness: (§3) team performance — whether a team is going to win — is directly based on the result/goal in a game and is the most common measure in prior studies; (§4) team tenacity — whether a team is going to surrender — indicates a team's resilience to adversity and is understudied in existing literature as most studies only look at individual tenacity or perseverance [5, 19]; (§5) toxic behavior, i.e., abusive language use, reflects the rapport in a team and is extracted from the activities during a game.

Our results on team effectiveness confirm the theory that balance in team roles is important for effective teams. We find that diverse teams are more likely to win. Furthermore, diverse teams are less likely to surrender, indicating a higher level of tenacity. Note that the likelihood to surrender does not correlate with winning rate for commonly used team compositions. However, we observe diverging effects on abusive language use: diverse teams are more likely to abuse when losing, but are less likely to abuse when winning.

In addition to exploring the effect of team composition, we investigate individual-level abusive language use to understand how teamwork influences individuals and shed light on the "situation vs. personality" debate [16, 22]. We find that players who choose assassins (the leading role) are more likely to abuse. We further demonstrate that the reason is that abusive players are more likely to choose assassins, instead of the alternative explanation that a player becomes more abusive when playing assassins. Finally, we present related work in §6 and offer our concluding thoughts in §7. A detailed version with additional analysis and prediction experiments can be found on arXiv: https://arxiv.org/abs/1902.06432.

Role	Description	Freq.(%)
warrior	Warriors have a large pool of health and considerable damage; they often undertake the stress from enemies in side lanes.	27.1
mage	Mages have heavy spell damage, but a small pool of health; they often take the middle lane and prepare to assist teammates.	25.8
marksman	Marksmen have a small pool of heath but can cause heavy damage; they need protection and may need time to grow.	25.7
assassin	Assassins are explosive and control the pace, but the health pool is often small; they are usually the leading role in a team.	11.5
support	Supports always aid teammates; they absorb damage while disrupting opponents by stunning and displacing them.	9.9

Table 2: Description and frequency of each role.

2 DATASET

Our dataset is provided by Tencent[2] and comes from a multiplayer online battle arena (MOBA) game: Honor of Kings, the most profitable game in the world. Similar to many other MOBA games such as Dota2 and League of Legends, Honor of Kings involves two teams to battle with each other. Each team consists of five players and the success of a team depends on the chemistry and collaboration between the five team members. The main motivation of our study is that players are expected to take certain roles when they select characters. These roles are designed to complement each other and "good" teams usually require particular compositions of roles. In this paper, we aim to explore the effect of team composition on different measures of team effectiveness.

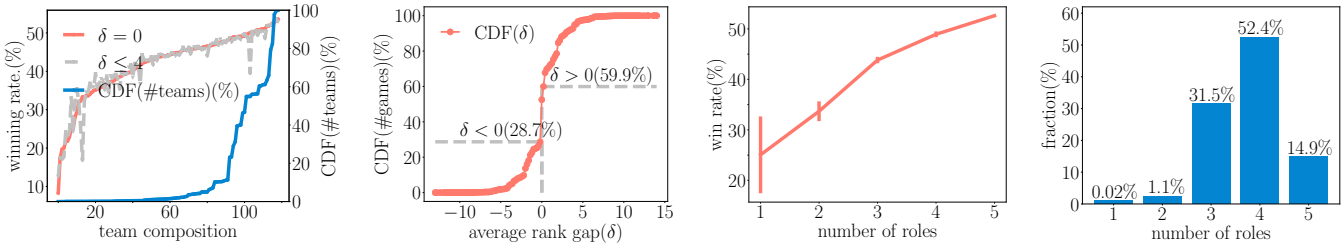
There are multiple types of games in Honor of Kings. *Ranked games* are used to estimate a player's rank level in the game; while the other games are for practice or for fun. Since players care much less about unranked games than ranked games, all of our analyses are based on ranked games. Unranked games are only used to analyze players' role preferences.

Dataset description. Our dataset is derived from daily logs spanning three weeks in August, 2017. Specifically, we first randomly sample 20K games in each day of the third week, thereby obtaining 140K games and 1.3M players participating in these games. Among them, there are 78K ranked games. We then retrieve all the games involving any of these 1.3M players in the first two weeks to obtain player information. In total, we obtain 95.6M games and 100.2M players. Among these games, there are 54.4M *ranked games* on which most analyses in §3, §4, and §5 are based. In those analytical settings, we do not need historical information of every player in a team. There are several exceptions where we need historical statistics of every team member, so we use games in the third week in those cases and will point out in the text.

The log of each game includes its start time, end time, result (which team wins), gaming stats recorded when the game ends, and text messages in the chat room with abusive labels. See overall statistics of the dataset in Table 1.

Team roles. There are five roles in Honor of Kings: warrior ( ), mage ( ), marksman ( ), assassin ( ), and support ( ). A team is

[2]Honor of Kings is developed and published by Tencent, which is one of the largest Internet companies and also the largest game service provider in China.



(a) Sorted winning rate of different team compositions.

(b) Distribution of rank gap (δ) between two teams.

(c) Winning rate vs. team diversity (rank gap $\delta = 0$).

(d) Frequency of teams vs. team diversity.

Figure 1: Team composition and winning rates. In Fig. 1a, x-value corresponds to the index of a team composition, and is sorted by winning rate. The CDF of #teams shows the cumulative distribution in #teams according to the order in the x-axis. Fig. 1b shows the distribution of the rank gap between two teams. Fig. 1c shows that a diverse team with more role categories is more likely to win (error bars represent standard errors). Fig. 1d shows that most teams are rationale and cover at least 3 roles.

composed of different roles depending on players' choices. Henceforth, we use five icons to represent a team composition, e.g., (🛡, 🗡, 🗡, 🏹, 🏹) stands for a team with one warrior, two mages, two assassins, and no marksman or support. In total, there exist 126 team compositions. Table 2 presents the characteristics and frequency of each role. Table 3 shows several role combinations with its winning rate and frequency. We can clearly see that team compositions are not used equally by players: single roles and role combinations both vary greatly in frequency.

3 TEAM COMPOSITION AND WINNING

We first investigate the effect of team composition on whether a team wins or not, a direct measure of team performance. Although the matching system in Honor of Kings is designed to balance the ability of two teams, we observe that the winning rate of most team compositions is significantly lower than 50%, indicating that some role combinations cannot effectively work with each other.

Although two teams are with similar "skills", there exist winning and losing team compositions (Fig. 1a). Since we have 126 different team compositions in total, it is straightforward to estimate the *winning rate* of each team composition using the fraction of winning games, $\frac{\#\text{winning games}}{\#\text{games}}$. Fig. 1a shows that the winning rate of different team compositions spans a wide range, from 8.3% to 53.6%. It seems that some team compositions are doomed to lose. In addition, the figure also shows the cumulative distribution of #teams for each team composition, suggesting that most losing teams do not occur frequently. Notice that a similar plot will also be made for surrendering and abusing and we will see that the CDFs present very different shapes.

The fluctuation of winning rate between different team compositions among all games may not due to the effect of team compositions; another possible explanation is the ability of individual players: only relatively bad players choose certain bad team compositions. Although a matching system is designed to balance the ability of the players in two teams, it is difficult to always make sure that two teams exactly have the same ability. Fig. 1b shows the distribution of the rank level gap (δ) between two teams [3]. To calculate δ, we sum up the individual rank level in a team, and subtract the result of losing team from that of winning team. By

[3]Rank level indicates a player's gaming skill and ranges from 0 to 26.

Team compositions	Win rate	Used frequency(%)
team compositions with the highest winning rate		
🗡-✙ – ✙ – ✙ – ✙	53.6%	0.3
🏹 – 🗡 – ✙ – ✙ – ✙	53.2%	2.0
🏹 – 🗡-✙ – ✙ – 🛡	52.6%	14.9
team compositions with the lowest winning rate		
🗡 – 🗡 – ✙ – ✙ – ✙	19.7%	3.1×10^{-3}
🏹 – 🏹 – 🏹 – 🏹 – 🏹	17.4%	2.2×10^{-2}
🗡 – ✙ – ✙ – ✙ – ✙	8.3%	7.0×10^{-4}
team compositions that are most used		
🏹 – 🗡 – ✙ – ✙ – 🛡	52.6%	14.9
🗡 – ✙ – ✙ – 🛡 – 🛡	52.0%	13.3
🏹 – 🗡 – ✙ – 🛡 – 🛡	48.5%	11.6

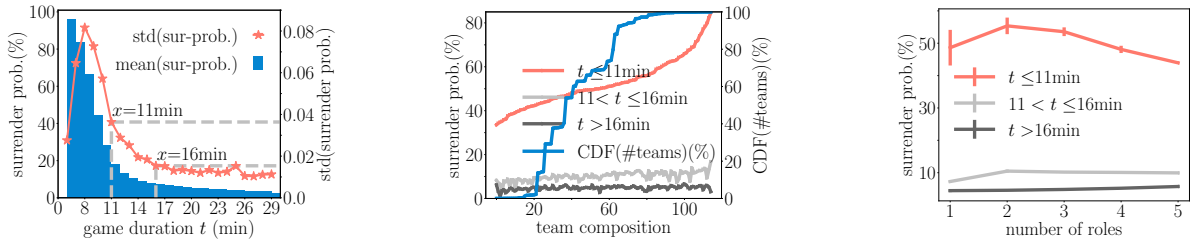
Table 3: Example team compositions and winning rates.

the effect of the matching system, the rank gaps in over 90% games are relatively small (within ±5). These minor gaps, however, are still associated with a significant difference in winning rate: only 28.7% games end up with the winning of the lower ranked team. Therefore, to demonstrate the influence of team compositions on winning rates and to exclude the ability factor, we control the rank gap between teams in Fig. 1a: we focus on games with $\delta = 0$ (red plots) and $\delta \leq 4$ (gray plots) respectively.

To further illustrate winning and losing team compositions, we present team compositions with the highest (lowest) winning rate in Table 3 (controlling the rank gap $\delta = 0$). Upon a careful examination, these losing teams do not often occur, reflected also by the cumulative distribution curve in Fig. 1a. In total, the bottom 90 losing team compositions only take 11.1% of all teams. This suggests that most players choose a reasonable team composition. However, the last three rows in Table 3 show that the most common team compositions still vary in winning rates, from 48.5% to 52.6%.

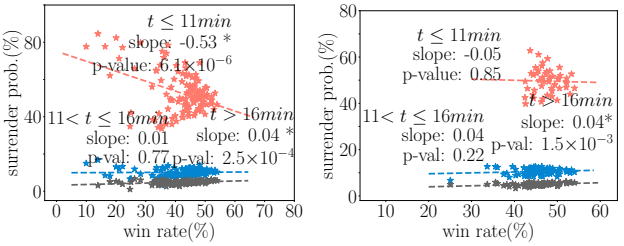
Diverse teams tend to perform well (Fig. 1c, 1d). An observation that stands out in Table 3 is that teams with the highest winning rate consist of more role categories than teams with the lowest winning rate. For instance, support (✙) or assassin (🏹) dominates the two compositions with the lowest winning rate, while all the top winning teams have at least four roles.

We further explore this observation by examining how winning rate changes as the number of roles in a team grows (Fig. 1c). We



(a) Surrender probability vs. game duration. (b) Surrender prob. vs. team composition. (c) Surrender prob. vs. team diversity.

Figure 2: Team composition and surrender probability. Fig. 2a shows the mean and standard deviation of surrender probability across team compositions conditioned on game duration. Fig. 2b shows that team compositions have varying surrender probabilities in games that end within 11 minutes, but have similar ones in longer games. Fig. 2c shows that diverse teams are less likely to surrender in early games (error bars represent standard errors).



(a) All team compositions. (b) Frequent team compositions.

Figure 3: Surrender probability vs. winning rate. In both figures, each dot denotes a team composition with its x-value for winning rate and y-value for surrender probability. Fig. 3a shows results for all team compositions. Fig. 3b shows the results after removing infrequent compositions, where surrender is not correlated with winning rate.

find that diverse team compositions are much more likely to win. One explanation is that these roles are designed to complement each other and teams with too few roles have weaknesses to be exploited. For instance, a team with no mage (🔧) or marksman (🎯) cannot cause sufficient damage in a game. Fig. 1d shows that most teams are employing at least three role categories, which is consistent with the observation in Fig. 1a that the losing team compositions on the left are not usually used.

4 TEAM COMPOSITION AND SURRENDER

As nothing worth achieving is going to be easy, tenacity is an important characteristic of an effective team. In the context of Honor of Kings, team tenacity can be reflected by not surrendering easily. The surrendering procedure is as follows: a player can propose to surrender and if at least four of five team members agree, the team will surrender as a whole and lose immediately. In this section, we examine the effect of team composition on team tenacity.

Surrender probability varies across team compositions in early games (Fig. 2a, 2b). Similar to winning, surrender happens at the end of a game. But surrender requires extra careful analysis because surrender may not always reflect team tenacity in Honor of Kings. For example, when the opponents are destroying the team base and all team members are killed, it is time to move on and surrender is simply a sign of "game over". Therefore, it is necessary to distinguish surrender in different game stages.

We examine the mean and the standard deviation of surrender probability across team compositions conditioned on game duration in Fig. 2a. We make two observations. First, the surrender probability monotonically decreases as game duration increases, partly because a team is only allowed to surrender starting from the 6th minute and games end in 6-8 minutes mostly because a team surrenders. Second, the standard deviation across team compositions first increases and then decreases as game duration increases, indicating that in late games every team surrenders with similar probability to signal game over. Based on Fig. 2a, we define three game stages: *early games* ($t \leq 11$), where the surrender probability has great mean and great std; *late games* ($t > 16$), where the surrender probability has small mean and small std; *middle games* ($11 < t \leq 16$), the middle part.

Having established the three types of games, we hypothesize that surrender in early games is the most indicative of team tenacity and varies across team compositions, while surrender in middle and late games is more of a formality and should not depend on team compositions. To study that, we sort team compositions by surrender probability in early games in Fig. 2b. According to the same order, we show the surrender probability in middle games and late games, as well as the cumulative distribution function of #teams. Consistent with our hypothesis, the surrender probability of different team compositions span a wide range from 33.6% to 84.6% in early games, but is pretty stable in middle games and late games. The CDF of #teams presents a different shape from that in Fig. 1a: commonly used team compositions are neither the most tenacious nor the least tenacious.

Diverse teams tend to be tenacious (Fig. 2c). Role diversity influences surrender probability in early games, especially when the number of roles is large. It seems that diverse teams with five roles are the most tenacious in early games. In comparison, the influence on middle and late games is minimal.

A weak team can still be tenacious (Fig. 3). We have shown that diverse teams tend to both perform well and show tenacity in adversity. One concern is that these two measures are correlated and tenacity is simply a side effect of team strength (winning). This hypothesis suggests that winning team compositions should be less likely to surrender. To further understand this issue, we study the correlation between surrender probability and winning rate for different team compositions. Fig. 3a shows that there is indeed a negative correlation between winning rate and surrender

probability in early games, while little correlation in middle and late games. However, it seems that the correlation between winning rate and surrender probability in early games is dominated by a few outliers with very low winning rates. Given that we know that these teams only take a small fraction of all games in §3, we filter infrequent team compositions that appear less than 10,000 times and find that in the remaining team compositions (98.7% of all teams), surrender probability is not correlated with winning rate in all game stages (Fig. 3b), suggesting that team tenacity is almost independent of team strength/performance.

5 TEAM COMPOSITION AND ABUSIVE LANGUAGE USE

Our final measure of team effectiveness is concerned with the rapport in a team during the game. We use abusive language use to capture the rapport. Toxic behavior in online communities and gaming has received significant interests from our research community recently [11, 12, 26]. Here we provide the first systematic study on the effect of team composition on team-level abusive language use.

5.1 Team-level Abusive Language Use

Team compositions vary in abusing probability (Fig. 4a). In our dataset, we have a label of whether a message uses abusive language for all text messages based on a dictionary-based method officially used by Tencent. A team abuses if any player in that team abuses. For each team composition, we define *abusing probability* as the fraction of games that this team composition abuses.

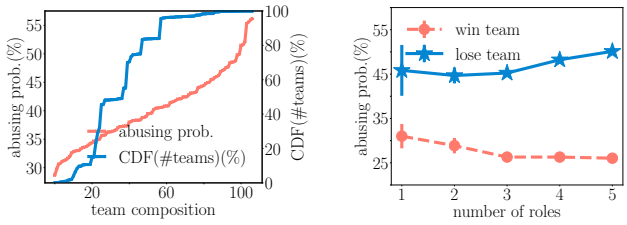
We find that similar to winning and surrendering, team compositions vary in abusing probability, ranging from 28.7% to 56.2%. The cumulative distribution function of #teams looks much more similar to surrendering in Fig. 2b than winning in Fig. 1a: most commonly used team compositions are neither the most nor the least abusive. We will investigate further the interaction of individual-level abusing and team-level abusing in §5.2.

Losing teams are more likely to abuse. We hypothesize that losing teams are more likely to abuse because winning usually brings positive team morale, while losing leads to frustration and dissatisfaction [26]. This is indeed the case, as we examine the correlation between abusing probability and winning rate by the same way in Fig. 3: the slope of abusing probability with respect to winning rate is -0.41 (p-value is 5.2×10^{-4}, and we omit the figure due to the space limitation). Therefore, it is important to distinguish abusing probability between winning teams and losing teams.

Diverse teams tend to abuse more when losing and abuse less when winning (Fig. 4b). We further explore the effect of role diversity on team-level abusing. We observe different trends in winning teams and losing teams. When a team wins, team diversity is associated with low abusing probability; but it becomes the other way around if a team loses.

5.2 Individual-level Abusive Language Use

Different from winning and surrendering, abusive language use is an individual behavior. It provides a great opportunity to understand the effect of team composition on individual behavior and shed light on the "situation vs. personality" debate [16, 22].



(a) Sorted abusing probability. **(b) Abusing probability. vs. team diversity.**

Figure 4: Team composition and abusive language use. Fig. 4a presents sorted abusing probability for different team compositions and its corresponding cumulative distribution function in #teams. In Fig. 4b, x-axis represents #roles in a team, and y-axis represents team-level abusing probability.

Note that we conduct this part of experiments on players who have played at least 20 games to ensure sufficient samples for statistics.

A team does not equal the sum of individuals (Fig. 5a). We define individual abusing probability based on the fraction of games that a player abuses in. Because of linearity of expectation, the total number of expected abusing players in a team is simply the sum of each player's abusing probability:

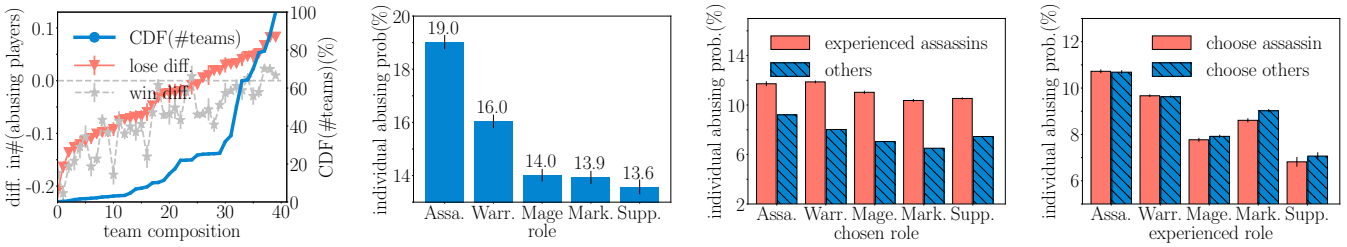
$$\mathbb{E}(\#\text{abusing players in a team}) = \sum_{v \in \texttt{team}} P_{\texttt{abuse}}(v) \qquad (1)$$

where $P_{\texttt{abuse}}$ is estimated separately when a player wins or loses because abusing is associated with losing. This expectation is only valid if all players act independently in a team, but studies on teamwork have shown that a team does not equal the sum of individuals [7]. Therefore, we examine the discrepancy between observed and expected values to understand how team composition influences individual players. This analysis is done on games in the third week since it requires historical information of every team member.

Figure 5a shows the difference between the observed and the expected values for winning and losing teams, where *win/lose diff.* refers to the disparity between the observed and the expected number of abusing players($\mathbb{O}$(#abusing players) − $\mathbb{E}$(#abusing players)). For most team compositions, the difference is not zero, indicating that individuals abuse differently depending on team compositions. In particular, in the most commonly used teams (when the CDF grows quickly on the right of the plot), individuals are more likely to abuse than expected when losing.

Players who prefer leading roles are more abusive (Fig. 5b, 5c, 5d). There always exists a leader or a major contributor in a team. In Honor of Kings, assassins usually take this role in a team because of their explosiveness: assassins can carry the team and control game pace. A failed assassin may lead the team to lose.

Fig. 5b shows that assassins are more likely to abuse than other roles. We call this the "abusive assassin" phenomenon. One natural question arises: are assassins more abusive because abusive players tend to choose assassins, or players become more abusive when choosing assassins? To answer this question, we compare the abusing probability of the same roles chosen by different players and the same player choosing different roles. We define the *experienced role* for a player if that player chooses a role frequently, i.e., playing in more than 50% of games. This procedure identifies experienced



(a) Observed #abusing players – expected values. **(b) Abusing probability vs. roles.** **(c) Abusing probability of the same roles by different players.** **(d) Abusing probability of the same player with different roles.**

Figure 5: Fig. 5a shows the difference between the observed number of abusing players and the expected value based on individual abusing probability. The right three figures investigate the "abusive" assassin phenomenon. Fig. 5b shows individual abusive probability grouped by roles. Fig. 5c compares the abusing probability of players who usually play assassins with those who usually play other roles, grouped by their chosen role. Fig. 5d compares the abusing probability when a player choose assassin with when the same player chooses other roles, grouped by his experienced role. Error bars represent standard errors.

assassin players, experienced warrior players, etc. Fig. 5c presents the abusing probability of experienced assassin players when they choose each role (the same player choosing different roles). We observe that these experienced assassin players are much more likely to abuse than other players no matter what role they choose.

Fig. 5d examines the alternative hypothesis: players become more abusive when choosing assassins. We compare the same player's abusing probability when they choose assassins with choosing other roles, grouped by their experienced roles. We find that players do not become more abusive when choosing assassins. If anything, experienced mages and experienced marksmen are actually more abusive when they choose roles other than assassins.

Overall, our results in Honor of Kings support the hypothesis that abusive players tend to choose assassins, the leading role in a team, and players do not become abusive when choosing assassins.

6 RELATED WORK

Team formation. Researchers from the data mining community have formulated the problem of team formulation as a constrained optimization problem based on each individual's skills and their social networks [2, 3, 27, 28]. For instance, Li et al. [28] study a family of problems in team enhancement including team member replacement, team expansion, and team shrinkage.

Social/team roles. Digitalization of human traces have increasingly made implicit social/team roles explicit and enabled large-scale study on theories of social/team roles [6–8, 14, 41], e.g., Yang et al. [42] study how social roles influence the process of online information diffusion. In the context of gaming, Stetina et al. [37] examined problematic gaming behavior and depressive tendencies among people who play different types of online games.

Online gaming. Many researchers have recognized that online gaming can serve as a platform for studying individual/team behavior and communities [18, 24], where players not only develop individual skills but also coordinate and communicate with others [17, 20, 21, 29, 39]. Among our three measures of effectiveness, team performance is the most heavily studied topic, e.g., Pobiedina et al. [33, 34] explore factors that influence player's performance in Dota 2. and Agarwala and Pearce [1] demonstrate that different role combinations may affect team performance. Furthermore, gamification can also play an important role in education and scientific discovery [23, 32]. In addition to these exciting opportunities, online gaming leads to issues such as addiction and depression [25, 37].

7 CONCLUSION

In this paper, we study the effect of team composition in the largest MOBA game: Honor of Kings. We quantitatively show the varying effects of team composition on team performance (winning), tenacity (surrender), and rapport (abusive language use): although diverse teams tend to perform well and show tenacity in adversity, they are more likely to abuse when losing. The double-edged influence of team diversity suggests the importance of balancing team composition. We further examine how team composition influences individual behavior in abusive language use. In addition to showing that a team is not the sum of independent individuals, we contribute to the "situation vs. personality" debate and show that assassins abuse more because abusive players tend to choose assassins instead of players becoming abusive when choosing assassins. Our work suggests that the gaming environment may be improved by adjusting team matching and preventing players from using team compositions that correlate with increased abusive language use.

Limitations. Although we control for roles in our analyses, our study is observational and can be strengthened with experimental studies. However, experimental studies on team compositions are non-trivial because of the great number of possible combinations. Our notion of diversity is limited by our lack of access to users' demographic information. Despite that Honor of Kings is the largest MOBA game, the selection bias in our data and specifics of the game design may limit the generalizability of our findings.

Future directions. Several promising directions arise from our work. It remains challenging to develop a holistic model that captures the interaction between individual members, learns novel representations for role combinations, and makes accurate predictions for effectiveness metrics to validate and deeply understand our findings in this paper. Furthermore, although MOBA games provide an ideal environment for understanding the effect of team composition, it is important to generalize our findings in other scenarios, e.g., with implicit roles beyond gaming.

Acknowledgments. The work is supported by NSFC (61702447), the Fundamental Research Funds for the Central Universities, and a research funding from HIKVISION Inc.

REFERENCES

[1] Atish Agarwala and Michael Pearce. 2014. *Learning Dota 2 team compositions.* Technical Report. Technical report, Stanford University.

[2] Aris Anagnostopoulos, Luca Becchetti, Carlos Castillo, Aristides Gionis, and Stefano Leonardi. 2010. Power in unity: forming teams in large-scale community systems. In *CIKM*.

[3] Aris Anagnostopoulos, Luca Becchetti, Carlos Castillo, Aristides Gionis, and Stefano Leonardi. 2012. Online team formation in social networks. In *WWW*.

[4] Rick Audas, Stephen Dobson, and John Goddard. 2002. The impact of managerial change on team performance in professional sports. *Journal of Economics and Business* 54, 6 (2002), 633–650.

[5] J Robert Baum and Edwin A Locke. 2004. The relationship of entrepreneurial traits, skill, and motivation to subsequent venture growth. *Journal of applied psychology* 89, 4 (2004), 587.

[6] R Meredith Belbin. 2011. Management teams: Why they succeed or fail. *Human Resource Management International Digest* 19, 3 (2011).

[7] R Meredith Belbin. 2012. *Team roles at work.* Routledge.

[8] Bruce J Biddle. 1986. Recent developments in role theory. *Annual review of sociology* 12, 1 (1986), 67–92.

[9] Erin Bradner, Gloria Mark, and Tammie D Hertel. 2005. Team size and technology fit: Participation, awareness, and rapport in distributed teams. *IEEE Transactions on Professional Communication* 48, 1 (2005), 68–77.

[10] Michael T. Brannick, Regina M. Roach, and Eduardo Salas. 1993. Understanding Team Performance: A Multimethod Study. *Human Performance* 6, 4 (1993), 287–308.

[11] Justin Cheng, Michael Bernstein, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2017. Anyone can become a troll: Causes of trolling behavior in online discussions. In *CSCW*.

[12] Justin Cheng, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2015. Antisocial Behavior in Online Discussion Communities. In *ICWSM*.

[13] Susan G. Cohen and Diane E. Bailey. 1997. What Makes Teams Work: Group Effectiveness Research from the Shop Floor to the Executive Suite. *Journal of Management* 23, 3 (1997), 239–290.

[14] Cristian Danescu-Niculescu-Mizil, Lillian Lee, Bo Pang, and Jon Kleinberg. 2012. Echoes of Power: Language Effects and Power Differences in Social Interaction. In *WWW*.

[15] Kurt T Dirks. 1999. The effects of interpersonal trust on work group performance. *Journal of applied psychology* 84, 3 (1999), 445.

[16] M. Brent Donellan, Richard E. Lucas, and William Fleeson (Eds.). 2009. *Personality and Assessment at Age 40: Reflections on the Past Person-Situation Debate and Emerging Directions of Future Person-Situation Integration [Special Issue].*

[17] Nicolas Ducheneaut, Nicholas Yee, Eric Nickell, and Robert J Moore. 2006. Alone together?: exploring the social dynamics of massively multiplayer online games. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*. ACM, 407–416.

[18] Nicolas Ducheneaut, Nicholas Yee, Eric Nickell, and Robert J. Moore. 2007. The Life and Death of Online Gaming Communities. In *CHI*.

[19] Angela L Duckworth, Christopher Peterson, Michael D Matthews, and Dennis R Kelly. 2007. Grit: perseverance and passion for long-term goals. *Journal of personality and social psychology* 92, 6 (2007), 1087.

[20] Guo Freeman. 2018. *Multiplayer Online Games: Origins, Players, and Social Dynamics.* AK Peters/CRC Press.

[21] Matthew Hudson and Paul Cairns. 2014. Measuring social presence in team-based digital games. *Interacting with Presence: HCI and the Sense of Presence in Computer-mediated Environments* (2014), 83.

[22] Douglas T Kenrick and David C Funder. 1988. Profiting from controversy: Lessons from the person-situation debate. *American Psychologist* 43, 1 (1988), 23.

[23] Firas Khatib, Seth Cooper, Michael D Tyka, Kefan Xu, Ilya Makedon, Zoran Popović, and David Baker. 2011. Algorithm discovery by protein folding game players. *the National Academy of Sciences* 108, 47 (2011), 18949–18953.

[24] Young Ji Kim, David Engel, Anita Williams Woolley, Jeffrey Yu-Ting Lin, Naomi McArthur, and Thomas W. Malone. 2017. What Makes a Strong Team?: Using Collective Intelligence to Predict Team Performance in League of Legends. In *CSCW*.

[25] Daria J Kuss, Jorik Louws, and Reinout W Wiers. 2012. Online gaming addiction? Motives predict addictive play behavior in massively multiplayer online role-playing games. *Cyberpsychology, Behavior, and Social Networking* 15, 9 (2012), 480–485.

[26] Haewoon Kwak, Jeremy Blackburn, and Seungyeop Han. 2015. Exploring Cyberbullying and Other Toxic Behavior in Team Competition Online Games. In *CHI*.

[27] Theodoros Lappas, Kun Liu, and Evimaria Terzi. 2009. Finding a team of experts in social networks. In *KDD*.

[28] Liangyue Li, Hanghang Tong, Nan Cao, Kate Ehrlich, Yu-Ru Lin, and Norbou Buchler. 2017. Enhancing Team Composition in Professional Networks: Problem Definitions and Fast Solutions. *IEEE Trans Knowl Data Eng* 29, 3 (2017), 613–626.

[29] Alexandru Losup, Ruud Van De Bovenkamp, Siqi Shen, Adele Lu Jia, and Fernando Kuipers. 2014. Analyzing implicit social networks in multiplayer online games. *IEEE Internet Computing* 3 (2014), 36–44.

[30] Glenn M Parker. 1990. *Team players and teamwork.* Jossey-Bass San Francisco, CA.

[31] David Partington and Hilary Harris. 1999. Team role balance and team performance: an empirical study. *Journal of Management Development* 18, 8 (1999), 694–705.

[32] Stacie Petter. 2017. More Than Child's Play: Embracing the Study of Online Gaming in Information Systems Research. *SIGMIS Database* 48, 4 (Nov. 2017), 9–13.

[33] Nataliia Pobiedina, Julia Neidhardt, Maria del Carmen Calatrava Moreno, and Hannes Werthner. 2013. Ranking factors of team success. In *Proceedings of the 22nd International Conference on World Wide Web*. ACM, 1185–1194.

[34] Nataliia Pobiedina, Julia Neidhardt, Maria del Carmen Calatrava Moreno, Laszlo Grad-Gyenge, and Hannes Werthner. 2013. On successful team formation: Statistical analysis of a multiplayer online game. In *Business Informatics (CBI), 2013 IEEE 15th Conference on*. IEEE, 55–62.

[35] Eduardo Salas, Nancy J Cooke, and Michael A Rosen. 2008. On teams, teamwork, and team performance: Discoveries and developments. *Human factors* 50, 3 (2008), 540–547.

[36] Barbara Senior. 1997. Team roles and team performance: Is there 'really' a link? *Journal of Occupational and Organizational Psychology* 70, 3 (1997), 241–258.

[37] Birgit U. Stetina, Oswald D. Kothgassner, Mario Lehenbauer, and Ilse Kryspin-Exner. 2011. Beyond the fascination of online-games: Probing addictive behavior and depression in the world of online-gaming. *Computers in Human Behavior* 27, 1 (2011), 473 – 479. Current Research Topics in Cognitive Load Theory.

[38] Greg L. Stewart, Ingrid S. Fulmer, and Murray R. Barrick. 2005. An exploration of member roles as a multilevel linking mechanism for individual traits and team outcomes. *Personnel Psychology* 58, 2 (2005), 343–365.

[39] Michael Szell and Stefan Thurner. 2010. Measuring social dynamics in a massive multiplayer online game. *Social networks* 32, 4 (2010), 313–329.

[40] Stefan Wuchty, Benjamin F Jones, and Brian Uzzi. 2007. The Increasing Dominance of Teams in Production of Knowledge. *Science* 316, 5827 (2007), 1036–1039.

[41] Diyi Yang, Aaron Halfaker, Robert Kraut, and Eduard Hovy. 2016. Who Did What: Editor Role Identification in Wikipedia. In *ICWSM*.

[42] Yang Yang, Jie Tang, Cane Wing-ki Leung, Yizhou Sun, Qicong Chen, Juanzi Li, and Qiang Yang. 2015. RAIN: Social Role-Aware Information Diffusion. In *AAAI'15*. 367–373.

TiFi: Taxonomy Induction for Fictional Domains*

Cuong Xuan Chu
Max Planck Institute for Informatics
Saarbrücken, Germany
cxchu@mpi-inf.mpg.de

Simon Razniewski
Max Planck Institute for Informatics
Saarbrücken, Germany
srazniew@mpi-inf.mpg.de

Gerhard Weikum
Max Planck Institute for Informatics
Saarbrücken, Germany
weikum@mpi-inf.mpg.de

ABSTRACT

Taxonomies are important building blocks of structured knowledge bases, and their construction from text sources and Wikipedia has received much attention. In this paper we focus on the construction of taxonomies for fictional domains, using noisy category systems from fan wikis or text extraction as input. Such fictional domains are archetypes of entity universes that are poorly covered by Wikipedia, such as also enterprise-specific knowledge bases or highly specialized verticals. Our fiction-targeted approach, called TiFi, consists of three phases: (i) category cleaning, by identifying candidate categories that truly represent classes in the domain of interest, (ii) edge cleaning, by selecting subcategory relationships that correspond to class subsumption, and (iii) top-level construction, by mapping classes onto a subset of high-level WordNet categories. A comprehensive evaluation shows that TiFi is able to construct taxonomies for a diverse range of fictional domains such as Lord of the Rings, The Simpsons or Greek Mythology with very high precision and that it outperforms state-of-the-art baselines for taxonomy induction by a substantial margin.

ACM Reference Format:
Cuong Xuan Chu, Simon Razniewski, and Gerhard Weikum. 2019. TiFi: Taxonomy Induction for Fictional Domains. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313519

1 INTRODUCTION

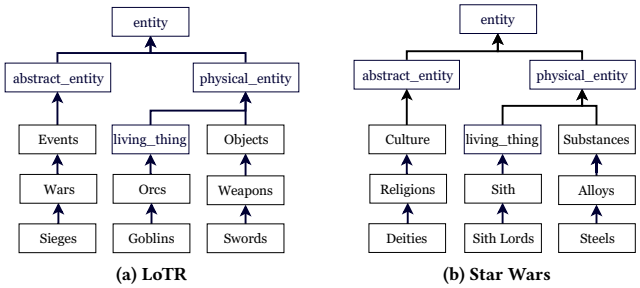
1.1 Motivation and Problem

Taxonomy Induction: Taxonomies, also known as type systems or class subsumption hierarchies, are an important resource for a variety of tasks related to text comprehension, such as information extraction, entity search or question answering. They represent structured knowledge about the subsumption of classes, for instance, that `electric guitar players` are `rock musicians` and that `state governors` are `politicans`. Taxonomies are a core piece of large knowledge graphs (KGs) such as DBpedia, Wikidata, Yago and industrial KGs at Google, Microsoft Bing, Amazon, etc. When search engines receive user queries about classes of entities, they can often find answers by combining instances of taxonomic classes. For example, a query about "left-handed electric guitar players" can be answered by intersecting the classes `left-handed people`, `guitar players` and `rock musicians`; a query about "actors who became politicans" can include instances from the intersection of `state governors` and `movie stars` such as Schwarzenegger. Also, taxonomic class systems are very useful for type-checking answer candidates for semantic search and question answering [27]. Taxonomies can be hand-crafted, examples being WordNet [13], SUMO [31] or MeSH and UMLS [4], or automatically constructed by *taxonomy induction* from textual or semi-structured cues about type instances and subtype relations. Methods for the latter include text mining using Hearst patterns [20] or bootstrapped with Hearst patterns (e.g., [47]), harvesting and learning from Wikipedia categories as a noisy seed network (e.g., [8, 14, 16, 35–37, 44, 46]), and inducing type hierarchies from query-and-click logs (e.g., [18, 32, 34]).

The Case for Fictional Domains: Fiction and fantasy are a core part of human culture, spanning from traditional literature to movies, TV series and video games. Well known fictional domains are, for instance, the Greek mythology, the Mahabharata, Tolkien's Middle-earth, the world of Harry Potter, or the Simpsons. These universes contain many hundreds or even thousands of entities and types, and are subject of search-engine queries – by fans as well as cultural analysts. For example, fans may query about Muggles who are students of the House of Gryffindor (within the Harry Potter universe). Analysts may be interested in understanding character relationships [3, 24, 43], learning story patterns [5, 6] or investigating gender bias in different cultures [1]. Thus, organizing entities and classes from fictional domains into clean taxonomies (see example in Fig. 1) is of great value.

Challenges: While taxonomy construction for encyclopedic knowledge about the real world has received considerable attention already, taxonomy construction for fictional domains is a new problem that comes with specific challenges:

1. State-of-the-art methods for taxonomy induction make assumptions on entity-class and subclass relations that are often invalid for fictional domains. For example, they assume that certain classes are disjoint (e.g., living beings and abstract entities, the oracle of Delphi being a counterexample). Also, assumptions



(a) LoTR (b) Star Wars

Figure 1: Excerpts of LoTR and Star Wars taxonomies.

*Extended version available at https://arxiv.org/abs/1901.10263.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313519

about the surface forms of entity names (e.g., on person names: with or without first name, starting with Mr., Mrs., Dr., etc.) and typical phrases for classes (e.g., noun phrases in plural form) do not apply to fictional domains.

2. Prior methods for taxonomy induction intensively leveraged Wikipedia categories, either as a content source or for distant supervision. However, the coverage of fiction and fantasy in Wikipedia is very limited, and their categories are fairly ad-hoc. For example, Lord Voldemort is in categories like `Fictional cult leaders` (i.e., people), `J.K. Rowling characters` (i.e., a meta-category) and `Narcissism in fiction` (i.e., an abstraction). And whereas Harry Potter is reasonably covered in Wikipedia, fan websites feature many more characters and domains such as House of Cards (a TV series) or Hyperion Cantos (a 4-volume science fiction book) that are hardly captured in Wikipedia.
3. Both Wikipedia and other content sources like fan-community forums cover an ad-hoc mixture of in-domain and out-of-domain entities and types. For example, they discuss both the fictional characters (e.g., Voldemort) and the actors of movies (e.g., Ralph Fiennes) and other aspects of the film-making or book-writing.

The same difficulties arise also when constructing enterprise-specific taxonomies from highly heterogeneous and noisy contents, or when organizing types for highly specialized verticals such as medieval history, the Maya culture, neurodegenerative diseases, or nano-technology material science. Methodology for tackling such domains is badly missing. We believe that our approach to fictional domains has great potential for being carried over to such real-life settings. This paper focuses on fiction and fantasy, though, where raw content sources are publicly available.

1.2 Approach and Contribution

In this paper we develop the first taxonomy construction method specifically geared for fictional domains. We refer to our method as the **TiFi** system, for **T**axonomy **i**nduction for **Fi**ction. We address Challenge 1 by developing a classifier for categories and subcategory relationships that combines rule-based lexical and numerical contextual features. This technique is able to deal with difficult cases arising from non-standard entity names and class names. Challenge 2 is addressed by tapping into fan community Wikis (e.g., `harrypotter.wikia.com`). This allows us to overcome the limitations of Wikipedia. Finally, Challenge 3 is addressed by constructing a supervised classifier for distinguishing in-domain vs. out-of-domain types, using a feature model specifically designed for fictional domains.

Moreover, we integrate our taxonomies with an upper-level taxonomy provided by WordNet, for generalizations and abstract classes. This adds value for searching by entities and classes. Our method outperforms the state-of-the-art taxonomy induction system for the first two steps, HEAD [16], by 21-23% and 6-8% percentage points in F1-score, respectively. An extrinsic evaluation based on entity search shows the value that can be derived from our taxonomies, where, for different queries, our taxonomies return answers with 24% higher precision than the input category systems. TiFi datasets and code are available at https://www.mpi-inf.mpg.de/yago-naga/tifi.

2 RELATED WORK

Text Analysis and Fiction. Analysis and interpretation of fictional texts are an important part of cultural and language research, both for the intrinsic interest in understanding themes and creativity [5, 6], and for extrinsic reasons such as predicting human behaviour [12] or measuring discrimination [1]. Other recurrent topics are, for instance, to discover character relationships [3, 24, 43], to model social networks [3, 9], or to describe personalities and emotions [10, 26].

Taxonomy Induction from Text. A seminal contribution towards their automated construction was the discovery of Hearst patterns [20], simple syntactic patterns like *"X is a Y"* that achieve remarkable precision, and are conceptually still part of many advanced approaches. Subsequent works aim to automate the process of discovering useful patterns [38, 42], devise probabilistic models and graph-flow algorithms [15], utilize distributional representations of types [30, 39, 45, 49], or aim to infer pairwise or hierarchical hypernymy [30, 49].

Taxonomy Construction using Wikipedia. A popular structured source for taxonomy construction is the Wikipedia category network (WCN) for taxonomy induction. One project, WikiTaxonomy [36, 37] aims to classify subcategory relations in the WCN as *subclass* and *not-subclass* relations. They investigate heuristics based on lexical matching between categories, lexico-syntactic patterns and the structure of the category network for that purpose. YAGO [22, 44] uses a very simple criterion to decide whether a category represents a class, namely to check whether it is in plural form. It also provides linking to WordNet [13] categories, choosing in case of ambiguity simply the meaning appearing topmost in WordNet. WiBi (Wikipedia Bitaxonomy) [14] proceeds in two steps: It first builds a taxonomy from Wikipedia pages by extracting and disambiguating lemmas from the first sentence of pages, then combines the page taxonomy and the original Wikipedia category network to induce the final taxonomy. A recent system, HEAD [16], exploits multiple lexical and structural rules towards classifying subcategory relations.

Domain-specific Taxonomies. TAXIFY is an unsupervised approach to domain-specific taxonomy construction from text [2]. Relying on distributional semantics, TAXIFY creates subclass candidates, which in a second step are filtered based on a custom graph algorithm. Similarly, Liu et al. [28] construct domain-specific taxonomies from keyword phrases augmented with relative knowledge and contexts. Compared with taxonomy construction from structured resources, these text-based approaches usually deliver comparably flat taxonomies.

Fan Wikis. Fans are organizing content on fictional universes on a multitude of webspaces. Particularly relevant for our problem are fan Wikis, i.e., community-built web content constructed using generic Wiki frameworks. Some notable examples of such Wikis are *tolkiengateway.net/wiki*, with 12k articles, *www.mariowiki.com* with 21k articles, or *en.brickimedia.org* with 29k articles. Particularly relevant are also Wiki farms, like Wikia[1] and Gamepedia[2], which host Wikis for 380k and 2k different fictional universes, and have Alexa rank 49 and 340, respectively.

[1] www.wikia.com/fandom
[2] www.gamepedia.com

These Wikis, like on Wikipedia, come with support for categories, the *The Lord of the Rings* Wiki, for instance, having over 900 categories and over 1000 subcategory relationships, the *Star Wars* Wiki having 11k and 14k of each, respectively. Similarly as on Wikipedia, these category networks do not represent clean taxonomies in the ontological sense, containing for instance meta categories such as `1980 films`, or relations such as `Death in Battle` being a subcategory of `Character`.

3 DESIGN RATIONALE AND OVERVIEW

3.1 Design Space and Choices

Input: The input of TiFi is a noisy category/tag tree or graph for a given set of entities (with textual description), from a domain of interest. Entities are easily available in many forums incl. Wikipedia, wikis of fan communities or scholarly collaborations, and other online media. Tags and categories, including some form of category hierarchy, are available in various kinds of wikis – typically in very noisy form, though, with a fair amount of uninformative and misleading connections. When such sites merely provide tags for entities, we can harness subsumptions between tags (e.g., simple association rules) to derive a *folksonomy* (see, e.g., [11, 23, 25]) and use this as an initial category system. When only text is available, we can use Hearst patterns and other text-based techniques [7, 20, 40] to generate categories and construct a subsumption-based tree.

Output: The goal of TiFi is to construct a clean taxonomy that preserves the valid and appropriate classes and their instance-of and subclass-of relationships but removes all invalid or misleading categories and connections. Formally, the output of TiFi is a directed acyclic graph (DAG) $G = (V, E)$ with vertices V and edges E such that (i) non-leaf vertices are semantic classes relevant for the domain, (ii) leaf vertices are entities, (iii) edges between leaves and their parents denote which entities belong to which classes, (iv) edges among non-leaf vertices denote subclass-of relationships.

There is a wealth of prior literature on taxonomy induction methods, and the design space for going about fictitious and other non-standard domains has many options. Our design decisions are driven by three overarching considerations:

- We leverage *whatever input information is available*, even if it comes with a high degree of noise.
- For the output taxonomy, we *prioritize precision over recall*. So our methods mostly focus on removing invalid vertices and edges. Moreover, to make classes for fictitious domains more interpretable and support cross-domain comparisons (e.g., for search), we aim to align the domain-specific classes with appropriate upper-level classes from a general-purpose ontology, using WordNet [13].
- It may seem tempting to cast the problem into an end-to-end machine-learning task. However, this would require sufficient training data in the form of pairs of input datasets and gold-standard output taxonomies. Such training data is not available, and would be hard and expensive to acquire. Instead, we break the overall task down into focused steps at the granularity of individual vertices and individual edges of category graphs. At this level, it is much easier to acquire labeled training data, and to devise features that capture specific context.

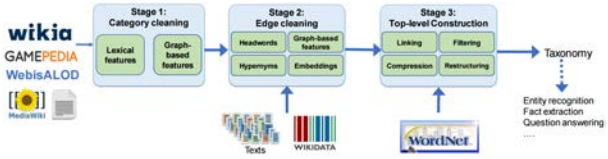


Figure 2: Architecture of TiFi.

3.2 TiFi Architecture

Based on the above considerations, we approach taxonomy induction in three steps, (1) category cleaning, (2) edge cleaning, (3) top-level construction. The architecture of TiFi is depicted in Fig. 2.

The first step, *category cleaning* (Section 4), aims to clean the original set of categories V by identifying categories that truly represent classes within the domain of interest, and by removing categories that represent, for instance, meta-categories used for community or Wikia coordination, or concern topics outside of the fictional domain, like movie or video game adaptions, award wins, and similar. While previous work has tackled this step via syntactic and lexical rules [33, 36, 44], for generalizability, we opt for a supervised classification approach that combines rules from above with additional graph-based features.

The second step, *edge cleaning* (Section 5), identifies the edges from the original category network E that truly represent subcategory relationships. We combine ideas from previous rule-based [17, 36] and embedding-based approaches [30] with various adapted semantic features and novel graph-based features.

For the third step, *top-level construction* (Section 6), we choose to reuse the existing WordNet top-level classes. This comes with the additional advantage of establishing a shared vocabulary across domains, allowing to query, for instance, for *animal species appearing both in LoTR and GoT* (with answers such as dragons).

4 CATEGORY CLEANING

In the first step, we aim to select the categories from the input that actually represent classes in the domain of interest. There are several reasons why a category would not satisfy this criterion, including the following: (i) *Meta-categories:* Wiki platforms typically introduce metacategories related to administration and technical setup, e.g., `Meta` or `Administration`. (ii) *Contextual categories:* Community Wikis usually contain also information about the production of the universes (e.g., inspirations or actors), about the reception (e.g., awards), and about remakes and adaptions, which do not related to the real content of the universes. (iii) *Instances:* Editors frequently create categories that are actually instances, e.g., Arda or Mordor in *The Lord of The Rings*). (iv) *Extensions:* Wikis sometimes also contains fan-made extensions of universes that are not universally agreed upon.

Previous works on Wikipedia remove either only meta-categories or instances by using crafted lexical rules [33, 36, 37]. As our setting has to deal with a wider range of noise, we instead choose the use of supervised classification. We use a logistic regression classifier with binary (0/1) lexical and integer graph-based features, as detailed next.

A. Lexical Features.

Meta-categories: True if a categories' name contains one of 22 manually selected strings, such as `wiki, template, user, portal, disambiguation, articles, administration, file, pages`, etc.
Plural categories: True if the headword of a category is in plural form. We use shallow parsing to extract headwords, for instance, identifying the plural term `Servants` in `Servants of Morgoth`, a strong indicator for a class.
Capitalization: True if a category starts with a capital letter. We introduced this feature as we observed that in fiction, lowercase categories frequently represent non-classes.

B. Graph-based Features.

Instance count: The number of direct instances of a category.
Supercategory/subcategory count: The number of super/subcategories of a category. Categories with more instances, superclasses or subclasses have potentially more relevance.
Average depth: Average upward path length from a category. Categories with short paths above are potentially more likely not relevant.
Connected subgraph size: The maximal size of connected subgraphs which a given category belongs to. Each connected subgraph is extracted by using depth first search on each root of the input category network. Meta-categories are sometimes disconnected from the core classes of a universe.

While the first two are established features, all other features have been newly designed to especially meet the characteristics of fiction. As we show in Section 7, this varied feature set allows to identify in-domain classes with 83%-85% precision.

5 EDGE CLEANING

Once the categories that represent classes in the domain of interest have been identified, the next task is to identify which subcategory relationships also represent subclass relationships. While most previous works rely on rules [8, 14, 16, 36], these are again too inflexible for the diversity of fictional universes. We thus tackle the task using supervised learning, relying on a combination of syntactic, semantic and graph-based features for a regression model.

A. Syntatic Features.

Head Word Matching. Head word matching is arguably the most popular feature for taxonomy induction. Categories sharing the same headword, for instance `Realms` and `Dwarven Realms` are natural candidates for hypernym relationships.

We use a shallow parsing to extract, for a category c, its headword $head(c)$, its prefix $pre(c)$, and its suffix (postfix) $pos(c)$, that is, $c = pre(c) + head(c) + pos(c)$. Consider a subcategory pair (c_1, c_2):

1. If $head(c_1) = head(c_2)$, $head(c_1) + pos(c_1) = head(c_2) + pos(c_2)$ and $pre(c_2) \subseteq pre(c_1)$ then c_2 is a superclass of c_1.
2. If $head(c_1) = head(c_2), pre(c_1)+head(c_1) = pre(c_2)+head(c_2)$ and $pos(c_2) \subseteq pos(c_1)$ then c_2 is a superclass of c_1.
3. If $head(c_1) \neq head(c_2)$ and $head(c_2) \subseteq pre(c_1)$ or $head(c_2) \subseteq pos(c_1)$ then there is no subclass relation between c_1 and c_2.

Case (1) covers the example of `Realms` and `Dwarven Realms`, while case (2) allows to infer, for instance, that `Elves` is a superclass of `Elves of Gondolin`. Case (3) allows to infer that certain categories are not superclasses of each other, e.g., `Gondor` and `Lords of Gondor`. Each of subclass and no-subclass inference are implemented as binary 0/1 features.
Only Plural Parent. True if for a subcategory pair (c_1, c_2), c_1 has no other parent categories, and c_2 is in plural form [16].

B. Semantic Features.

WordNet Hypernym Matching. WordNet is a carefully handcrafted lexical database that contains semantic relations between words and word senses (synsets), including hypo/hypernym relations. To leverage this resource, we map categories to WordNet synsets, using context-based similarity to identify the right word sense in the case of ambiguities. To compute the context vectors of categories, we extract their definitions, that is, the first sentence from the Wiki pages of the categories (if existing), and their parent and child class names. As context for WordNet synsets we use the definition (gloss) of each sense. We then compute cosine similarities over the resulting bags-of-words, and link each category with the position-adjusted most similar WordNet synset.Then, given categories c_1 and c_2 with linked WordNet synset s_1 and s_2, respectively, this feature is true if s_2 is a WordNet hypernym of s_1.
Wikidata Hypernym Matching. Similarly to WordNet, Wikidata also contains relations between entities. For example, Wikidata knows that `Maiar` is an instance (P31) of `Middle-earth races` in the *The Lord of the Rings*. While Wikidata's coverage is generally lower than that of Wordnet, its content is sometimes complementary, as WordNet does not know certain concepts, e.g., `Maiar`.
Page Type Matching. One interesting contribution of the WiBi system [14] was to use the first sentence of Wikipedia pages to extract hypernyms. First sentences frequently define concepts, e.g., *"The Haradrim, known in Westron as the Southrons and once as the "Swertings" by Hobbits, were a race of Men from Harad in the region of Middle-earth directly south of Gondor"*. For categories having matching articles in the Wikis, we rely on the first sentence from these. We use the Stanford Parser [29] on the definition of the category to get a dependency tree. By extracting `nsubj, compound` and `conj` dependencies, we get a list of hypernyms for the category. For example, for `Haradrim` we can extract the relation `nsubj(race-13, Haradrim-2)`, hence `race` is a hypernym of `Haradrim`. After getting hypernyms for a category, we link these hypernyms to classes in the taxonomies by using head word matching, and set this feature to true for any pair of categories linked this way.
WordNet Synset Description Type Matching. Similar to page type matching, we also extract superclass candidates from the description of the WordNet synset. For instance, given the WordNet description for `Werewolves`: *"a monster able to change appearance from human to wolf and back again"*, we can identify `Monster` as superclass.
Distributional Similarity. The distributional hypothesis states that similar words share similar contexts [19], and despite the subclass relation being asymmetric, symmetric similarity measures have been found to be useful for taxonomy construction [41]. In this work, we utilize two distributional similarity measures, a symmetric one based on the structure of WordNet, and an asymmetric one based on word embeddings. The first is the symmetric Wu-Palmer score compares the depth of two synsets (the headwords of the categories) with the depth of their least common subsumer,

while the second is the [48]. HyperVec score [30], which not only shows the similarity between a category and its hypernym, but is also directional.

While WordNet only captures similarity between general concepts, embedding-based measures can cover both conceptual and non-conceptual categories, as often needed in the fantasy domain (e.g. similarity between `Valar` and `Maiar`).

C. Graph-based Features.

Common Children Support. Absolute number of common children (categories and instances) of two given categories. Presumably, the more common children two categories have, the more related to each other they are.

Children Depth Ratio. The ratio between the number of child categories of the parent of the edge, and its average depth in the taxonomy. This captures the generality of the parent candidate.

The features for edge cleaning combine existing state-of-the-art features (Head word matching, Page type matching, HyperVec) with adaptations specific to our domain (Wikidata hypernym matching, WordNet synset matching), and new graph-based features. Section 7 shows that this feature set allows to surpass the state-of-the-art in edge cleaning by 6-8% F1-score.

6 TOP-LEVEL CONSTRUCTION

Category systems from Wiki sources often rather resemble forests than trees, i.e., do not reach towards very general classes, and miss useful generalizations such as `man-made structures` or `geographical features` for `fortresses` and `rivers`. While works geared towards Wikipedia typically conclude with having identified classes and subclasses [8, 14, 16, 36, 37], we aim to include generalizations and abstract classes consistently across universes. For this purpose, TiFi employs as third step the integration of selected abstract WordNet classes. The integration proceeds in three steps:

(1) Given the taxonomy constructed so far, nodes are linked to WordNet synsets based on context similarity. Where the linking is successful, WordNet hypernyms are then added as superclasses. For example, the category `Birds` is linked to the WordNet synset `bird%1:05:00::`, whose superclasses are `wn_vertebrate` → `wn_chordate` → `wn_animal` → `wn_organism` → `wn_living_thing` → `wn_whole` → `wn_object` → `wn_physical_entity` → `wn_entity`.
(2) The added classes are then compressed by removing those that have only a single parent and a single child, for instance, `abstract_entity` and `physical_entity` in Fig. **??** (right) would be removed, if they really had only one child.
(3) We correct a few WordNet links that are not suited for the fictional domain, and use a self-built dictionary to remove 125 top-level WordNet synsets that are too abstract to add value, for instance, `whole`, `sphere` and `imagination`.

Note that the present step can add subclass relationships between existing classes. For instance, after edge filtering, there is no relation between `Birds` and `Animals`, while after linking to WordNet, the subclass relation between `Birds` and `Animals` is added, making the resulting taxonomy more dense and useful.

Table 1: Input categories from Wikia/Gamepedia.

Universe	# Categories	# Edges
Lord of the Rings (LoTR)	973	1118
Game of Thrones (GoT)	672	1027
Star Wars	11012	14092
Simpsons	2275	4027
World of Warcraft	8249	11403
Greek Mythology	601	411

7 EVALUATION

In this section we evaluate the performance of the individual steps of the TiFi approach, and the ability of the end-to-end system to build high-quality taxonomies. We use 6 universes that cover fantasy (LoTR, GoT), science fiction (Star Wars), animated sitcom (Simpsons), video games (World of Warcraft) and mythology (Greek Mythology). For each of these, we extract their category networks from dump files of Wikia or Gamepedia. The sizes of the respective category networks, the input to TiFi, are shown in Table 1.

Step 1: Category Cleaning. Evaluation data for the first step was created using crowdsourcing, which was used to label all categories in LoTR, GoT, and random 50 from each of the other universes.

As baselines we employ a rule-based approach by Ponzetto & Strube [37], to the best of our knowledge the best performing method for general category cleaning, and recent work by Marius Pasca [33] that targets the aspect of separating classes from instances. Furthermore, we combine both methods into a joint filter. The results of training and testing on LoTR/GoT, respectively, each under 10-fold crossvalidation, are shown in Table 2. TiFi achieves both superior precision (+40%) and F1-score (+22%/+23%), while observing a smaller drop in recall (-18%/-15%). On both fully annotated universes the improvement of TiFi over the combined baseline in terms of F1-score is statistically significant (p-value 2.2^{-16} and 1.9^{-13}, respectively). The considerable difference in precision is explained largely by the limited coverage of the rule-based baseline.

As our approach requires labeled training data, a question is to which extent labeled data from one domain can be used for cleaning categories of another domain. We thus next evaluate the performance when applying models trained on LoTR on the other 5 universes, and the model trained on GoT on LoTR. The results are shown in Table 3, where for universes other than LoTR and GoT, having annotated only 50 samples. As one can see, F1-scores

Table 2: Step 1 - In-domain category cleaning.

Method	Universe	Precision	Recall	F1-score
Pasca 2018 [33]	LoTR	0.33	0.75	0.46
	GoT	0.57	0.85	0.68
Ponzetto & Strube 2011 [37]	LoTR	0.44	**1.0**	0.61
	GoT	0.45	**1.0**	0.62
Pasca + Ponzetto & Strube	LoTR	0.41	0.75	0.53
	GoT	0.64	0.85	0.73
TiFi	LoTR	**0.84**	0.82	**0.83**
	GoT	**0.85**	0.85	**0.85**

drop by only 9%/2% compared with same-domain training, and the F1-score is above 70% even for quite different domains.

To explore the contribution of each feature, we performed an ablation test using recursive feature elimination. The most important feature group were lexical features (30%/10% F1-score drop if removed in LoTR/GoT), with plural form checking being the single most important feature. In contrast, removing the graph-based features lead only to a 10%/0% drop, respectively.

Step 2: Edge Cleaning. We used crowdsourcing to label all edges that remained after cleaning noisy categories from LoTR, GoT, and random 100 edges in each of the other universes.

We compare TiFi with two state-of-the-art systems: (1) HEAD [16], the most recent system for Wikipedia category relationship cleaning, and (2) HyperVec [30], a recent embedding-based hypernym relationship learning system. The results for in-domain evaluation using 10-fold crossvalidation are shown in Table 4. As one can see, TiFi achieves a comparable precision (-2%/+2%), and a superior recall (+15%/+13%), resulting in a gain in F1-score of 6%/8%. Again, the F1-score improvement of TiFi over HyperVec and HEAD on the two fully annotated universes is statistically significant (p-values 7.1^{-9}, 0.01, 5.8^{-5} and 6.5^{-5}, respectively).

To explore the scalability of TiFi, we again perform cross-domain experiments using 100 labeled edges per universe. The results are shown in Table 5. In all universes but *World of Warcraft*, TiFi achieves more than 80% F1-score, and the performance is further highlighted by mean average precision (MAP) scores above 89%, meaning TiFi can effectively separate correct from incorrect edges.

In the ablation test, all three groups of features showed importance, each leading to a 1-4% drop in F1-score when being removed. The individually most important features were *Only Plural Parent, Headword Matching, Common Children Support* and *Page Type Matching*.

Step 3: Top-level Construction. The key step in top-level construction is the linking of categories to WordNet synsets (i.e. category disambiguation), hence we only evaluate this step. For this purpose, in each universe, we randomly selected 50 such links and evaluated their correctness, finding precisions between 84% and 92%. Overall, this step is able to link 30-72% of top-level classes from Step 2, and adds between 22 to 373 WordNet classes and 76 to 3387 subclass relationships to our universes.

Final Taxonomies. Table 6 summarizes the taxonomies constructed for our 6 universes, with the bottom 4 universes built using the models for GoT. Reported precisions refer to the weighted average of the precision of subclass edges from Step 2, and the precision of WordNet linking. The taxonomies are available at https://www.mpi-inf.mpg.de/yago-naga/tifi.

Other Inputs. We also evaluated TiFi on the real-world Wikipedia dataset, and on the noisy WebIsALOD [21] hypernymy dataset. We find that TiFi performs comparable to a state-of-the-art Wikipedia baseline [37] on Wikipedia, and that it significantly outperforms [37] and [33] on the noisier WebIsALOD data. More details are in the extended version.

Table 3: Step 1 - Cross-domain category cleaning.

Train	Test	Precision	Recall	F1-score
LoTR	GoT	0.81	0.85	0.83
GoT	LoTR	0.64	0.88	0.74
LoTR	Star Wars	0.63	0.94	0.75
LoTR	Simpsons	0.91	0.63	0.74
LoTR	World of Warcraft	0.95	0.63	0.75
LoTR	Greek Mythology	0.86	0.6	0.71

Table 4: Step 2 - In-domain edge cleaning.

Method	Universe	Precision	Recall	F1-score
HyperVec [30]	LoTR	0.82	0.8	0.81
	GoT	**0.83**	0.81	0.82
HEAD [16]	LoTR	**0.85**	0.83	0.84
	GoT	0.81	0.78	0.79
TiFi	LoTR	0.83	**0.98**	**0.90**
	GoT	**0.83**	**0.91**	**0.87**

Table 5: Step 2 - Cross-domain edge cleaning.

Train	Test	Precision	Recall	F1-score	MAP
LoTR	GoT	0.81	0.79	0.80	0.92
GoT	LoTR	0.89	0.87	0.88	0.89
GoT	Star Wars	0.92	0.92	0.92	0.91
GoT	Simpsons	0.86	0.86	0.86	0.92
GoT	Word of Warcraft	0.72	0.71	0.72	0.76
GoT	Greek Mythology	0.92	0.92	0.92	0.92

Table 6: Taxonomies produced by TiFi.

Universe	# Types	# Edges	Precision
LoTR	353	648	0.88
Game of Thrones	292	497	0.83
Star Wars	7352	12282	0.90
Simpsons	1029	2171	0.88
World of Warcraft	4063	7882	0.76
Greek Mythology	139	313	0.91

8 USE CASE: ENTITY SEARCH

To highlight the usefulness of our taxonomies, we provide an extrinsic evaluation based on the use case of entity search. Entity search is a standard problem in information retrieval, where often, textual queries shall return lists of matching entities. In the following, we focus on the retrieval of correct entities only, and disregard the ranking aspect. The number of answers and their precision@10 is shown in Table 7. As one can see, entity search using TiFi yields significantly higher precision than Google and Wikia text search.

Table 7: Avg. #Answers and precision of entity search.

	Text		Structured Sources	
Query	Google	Wikia	Wikia-categories	TiFi
t	2 (52%)	7 (65%)	10 (62%)	8 (87%)
$t_1 \cap t_2$	1 (23%)	2 (11%)	8 (40%)	3 (70%)
$t_1 \setminus t_2$	1 (20%)	4 (36%)	8 (63%)	6 (79%)
Average	1 (32%)	4 (37%)	9 (55%)	6 (79%)

9 CONCLUSION

In this paper we have introduced TiFi, a system for taxonomy induction for fictional domains. TiFi uses a three-step architecture with category cleaning, edge cleaning, and top-level construction, thus building holistic domain specific taxonomies that are consistently of higher quality than what the Wikipedia-oriented state-of-the-art could produce.

REFERENCES

[1] Apoorv Agarwal, Jiehan Zheng, Shruti Kamath, Sriramkumar Balasubramanian, and Shirin Ann Dey. 2015. Key Female Characters in Film have more to talk about besides men: Automating the Bechdel Test. In *NAACL*. 830–840.

[2] Daniele Alfarone and Jesse Davis. 2015. Unsupervised Learning of an is-a Taxonomy from a Limited Domain-specific Corpus. In *IJCAI*. 1434–1441.

[3] David Bamman, Brendan O'Connor, and Noah A Smith. 2014. Learning Latent Personas of Film Characters. In *ACL*. 352.

[4] Olivier Bodenreider. 2004. The Unified Medical Language System (UMLS): Integrating Biomedical Terminology. *Nucleic acids research* (2004), D267–D270.

[5] Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised Learning of Narrative Schemas and their Participants. In *ACL/IJCNLP*. 602–610.

[6] Snigdha Chaturvedi, Mohit Iyyer, and Hal Daumé III. 2017. Unsupervised Learning of Evolving Relationships Between Literary Characters. In *AAAI*. 3159–3165.

[7] Philipp Cimiano, Andreas Hotho, and Steffen Staab. 2005. Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis. *J. Artif. Intell. Res.* 24 (2005), 305–339.

[8] Gerard de Melo and Gerhard Weikum. 2010. MENTA: Inducing Multilingual Taxonomies from Wikipedia. In *CIKM*. 1099–1108.

[9] Vinodh Krishnan Elangovan and Jacob Eisenstein. 2015. "You're Mr. Lebowski, I'm the Dude": Inducing Address Term Formality in Signed Social Networks. In *NAACL*. 1616–1626.

[10] David K Elson, Nicholas Dames, and Kathleen R McKeown. 2010. Extracting Social Networks from Literary Fiction. In *ACL*. 138–147.

[11] Quan Fang, Changsheng Xu, Jitao Sang, M. Shamim Hossain, and Ahmed Ghoneim. 2016. Folksonomy-Based Visual Ontology Construction and Its Applications. *IEEE Trans. Multimedia* 18, 4 (2016), 702–713.

[12] Ethan Fast, William McGrath, Pranav Rajpurkar, and Michael S Bernstein. 2016. Augur: Mining Human Behaviors from Fiction to Power Interactive Systems. In *CHI*. 237–247.

[13] Christiane Fellbaum and George Miller. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

[14] Tiziano Flati, Daniele Vannella, Tommaso Pasini, and Roberto Navigli. 2014. Two Is Bigger (and Better) Than One: the Wikipedia Bitaxonomy Project.. In *ACL*. 945–955.

[15] Amit Gupta, Rémi Lebret, Hamza Harkous, and Karl Aberer. 2017. Taxonomy Induction using Hypernym Subsequences. In *CIKM*. 1329–1338.

[16] Amit Gupta, Francesco Piccinno, Mikhail Kozhevnikov, Marius Pasca, and Daniele Pighin. 2016. Revisiting Taxonomy Induction over Wikipedia. In *COLING*. 2300–2309.

[17] Amit Gupta, Francesco Piccinno, Mikhail Kozhevnikov, Marius Pasca, and Daniele Pighin. 2016. Revisiting Taxonomy Induction over Wikipedia. In *COLING 2016*. 2300–2309.

[18] Rahul Gupta, Alon Y. Halevy, Xuezhi Wang, Steven Euijong Whang, and Fei Wu. 2014. Biperpedia: An Ontology for Search Applications. *PVLDB* (2014), 505–516.

[19] Zellig S Harris. 1954. Distributional Structure. *Word* (1954), 146–162.

[20] Marti A Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *COLING*. 539–545.

[21] Sven Hertling and Heiko Paulheim. 2017. WebIsALOD: providing hypernymy relations extracted from the web as linked open data. In *ISWC*.

[22] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia. *Artificial Intelligence* (2013), 28–61.

[23] Andreas Hotho, Robert Jäschke, Christoph Schmitz, and Gerd Stumme. 2006. Information Retrieval in Folksonomies: Search and Ranking. In *ESWC 2006*. 411–426.

[24] Mohit Iyyer, Anupam Guha, Snigdha Chaturvedi, Jordan Boyd-Graber, and Hal Daumé III. 2016. Feuding Families and Former Friends: Unsupervised Learning for Dynamic Fictional Relationships. In *NAACL*. 1534–1544.

[25] Robert Jäschke, Leandro Balby Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. 2007. Tag Recommendations in Folksonomies. In *PKDD 2007*. 506–514.

[26] Harshita Jhavar and Paramita Mirza. 2018. EMOFIEL: Mapping Emotions of Relationships in a Story. In *The Web Conference*. 243–246.

[27] Aditya Kalyanpur, J. William Murdock, James Fan, and Christopher A. Welty. 2011. Leveraging Community-Built Knowledge for Type Coercion in Question Answering. In *ISWC 2011*. 144–156.

[28] Xueqing Liu, Yangqiu Song, Shixia Liu, and Haixun Wang. 2012. Automatic Taxonomy Construction from Keywords. In *KDD*. 1433–1441.

[29] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *ACL*. 55–60.

[30] Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017. Hierarchical Embeddings for Hypernymy Detection and Directionality. *arXiv preprint arXiv:1707.07273* (2017).

[31] Ian Niles and Adam Pease. 2001. Towards a Standard Upper Ontology. In *FOIS*. 2–9.

[32] Marius Pasca. 2013. Open-Domain Fine-Grained Class Extraction from Web Search Queries. In *EMNLP 2013*. 403–414.

[33] Marius Pasca. 2018. Finding Needles in an Encyclopedic Haystack: Detecting Classes Among Wikipedia Articles. In *WWW*. 1267–1276.

[34] Marius Pasca and Benjamin Van Durme. 2007. What You Seek Is What You Get: Extraction of Class Attributes from Query Logs. In *IJCAI 2007*. 2832–2837.

[35] Simone Paolo Ponzetto and Roberto Navigli. 2009. Large-Scale Taxonomy Mapping for Restructuring and Integrating Wikipedia. In *IJCAI*. 2083–2088.

[36] Simone Paolo Ponzetto and Michael Strube. 2007. Deriving a Large Scale Taxonomy from Wikipedia. In *AAAI*. 1440–1445.

[37] Simone Paolo Ponzetto and Michael Strube. 2011. Taxonomy Induction Based on a Collaboratively Built Knowledge Repository. *Artificial Intelligence* (2011), 1737–1756.

[38] Stephen Roller and Katrin Erk. 2016. Relations such as Hypernymy: Identifying and Exploiting Hearst Patterns in Distributional Vectors for Lexical Entailment. *arXiv preprint arXiv:1605.05433* (2016).

[39] Stephen Roller, Katrin Erk, and Gemma Boleda. 2014. Inclusive yet Selective: Supervised Distributional Hypernymy Detection. In *COLING*. 1025–1036.

[40] Mark Sanderson and W. Bruce Croft. 1999. Deriving Concept Hierarchies from Text. In *SIGIR 1999*. 206–213.

[41] Vered Shwartz, Enrico Santus, and Dominik Schlechtweg. 2016. Hypernyms under Siege: Linguistically-motivated Artillery for Hypernymy Detection. *arXiv preprint arXiv:1612.04460* (2016).

[42] Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2005. Learning Syntactic Patterns for Automatic Hypernym Discovery. In *NIPS*. 1297–1304.

[43] Shashank Srivastava, Snigdha Chaturvedi, and Tom M Mitchell. 2016. Inferring Interpersonal Relations in Narrative Summaries. In *AAAI*. 2807–2813.

[44] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. YAGO: A Core of Semantic Knowledge. In *WWW*. 697–706.

[45] Tu Vu and Vered Shwartz. 2018. Integrating Multiplicative Features into Supervised Distributional Methods for Lexical Entailment. *arXiv preprint arXiv:1804.08845* (2018).

[46] Fei Wu, Raphael Hoffmann, and Daniel S. Weld. 2008. Information Extraction from Wikipedia: Moving Down the Long Tail. In *KDD 2008*. 731–739.

[47] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Qili Zhu. 2012. Probase: A Probabilistic Taxonomy for Text Understanding. In *SIGMOD 2012*. 481–492.

[48] Zhibiao Wu and Martha Palmer. 1994. Verb Semantics and Lexical Selection. In *ACL*. 133–138.

[49] Zheng Yu, Haixun Wang, Xuemin Lin, and Min Wang. 2015. Learning Term Embeddings for Hypernymy Identification. In *IJCAI*. 1390–1397.

Deriving User- and Content-specific Rewards for Contextual Bandits

Paolo Dragone*†
University of Trento
TIM (SKIL)
paolo.dragone@unitn.it

Rishabh Mehrotra
Spotify
rishabhm@spotify.com

Mounia Lalmas
Spotify
mounia@acm.org

ABSTRACT

Bandit algorithms have gained increased attention in recommender systems, as they provide effective and scalable recommendations. These algorithms use *reward functions*, usually based on a numeric variable such as click-through rates, as the basis for optimization. On a popular music streaming service, a contextual bandit algorithm is used to decide which content to recommend to users, where the reward function is a binarization of a numeric variable that defines success based on a static threshold of user streaming time: 1 if the user streamed for at least 30 seconds and 0 otherwise. We explore alternative methods to provide a more informed reward function, based on the assumptions that streaming time distribution heavily depends on the type of user and the type of content being streamed. To automatically extract user and content groups from streaming data, we employ "co-clustering", an unsupervised learning technique to simultaneously extract clusters of rows and columns from a co-occurrence matrix. The streaming distributions within the co-clusters are then used to define rewards specific to each co-cluster. Our proposed co-clustered based reward functions lead to improvement of over 25% in expected stream rate, compared to the standard binarized rewards.

ACM Reference Format:
Paolo Dragone, Rishabh Mehrotra, and Mounia Lalmas. 2019. Deriving User- and Content-specific Rewards for Contextual Bandits. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313592

1 INTRODUCTION

Given the overwhelming choices faced by users on what to watch, read and listen to online, recommender systems play a pivotal role in helping users navigate the myriad of choices. Most modern recommender systems are powered by interactive machine learning algorithms such as bandits [14, 17, 23], which learn to adapt their recommendations by estimating a model of the user satisfaction metric from the users feedback.

A key component of such bandit based recommender systems is the choice of *reward function*. A proper reward function should correlate well with user satisfaction, to allow the model to learn to serve good recommendations. Most prior work on quantifying satisfaction has relied on leveraging implicit signals derived from user interactions, mostly clicks [12, 19, 21, 25]. Often, click based signals are not informative enough and fail to differentiate between satisfying and dissatisfying experiences of users [11, 28]. Consequently, there has been a pressing need to move beyond clicks and investigate *post-click* behavior of users. These focus on the engagement of users with the served recommendations and have shown promising results in other areas [1, 11, 15].

Considering the specific use case of music streaming, quantifying such a notion of satisfaction from implicit signals involves understanding the diverse needs of users and their expectations of what is a successful streaming session. Such needs often include how users feel, and the expectations that music recommended to them align with their mood or their intent of the moment [8]. To fulfill these, music streaming services provide users with curated playlists, ranging from "sleep" to "run". In addition, and to account for the diverse user interests and plethora of musics, genre playlists have been made available to users, ranging from rap, pop, jazz, to niche ones. As a results, millions of playlists are available to users to listen to based on their intent and needs. Given such a heterogeneity in user needs, and the different intents of content (playlists), it becomes important to consider both user and content behavior to formalize the notion of satisfaction, and in turn design the appropriate reward models to capture these.

A major portion of work done on specifying reward functions rely on manually crafted functions, including rewards based on click-through behavior of users and other positive and negative feedback signals [29]. Some efforts have looked at strategies to improve the reward estimation in dynamic recommendation environments [2]. Overall, most bandit models powering recommendation system employ a rather simple threshold based reward function, mostly determined through click-through behavior of the user: *if user clicks on X, a payout of 1 is incurred and 0 otherwise* [17].

In this work, we go beyond such simple threshold-based formulation. We consider the case of a music streaming service, Spotify, powered by a contextual bandit model [20], and aim at defining appropriate reward functions to optimize the bandit model. We revisit how success is quantified, based on how users *consume* playlists and how playlists *are consumed* by users. Our approach moves from a simple user- and content-agnostic, binary reward model to a more sophisticated reward model, which is aware of the distribution of listening behavior. We highlight the need for jointly

*This work was carried out while Paolo Dragone was an intern at Spotify.
†Paolo Dragone is a fellow of TIM-SKIL Trento and is supported by a TIM scholarship.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313592

considering user and content interaction to define rewards, and leverage insights from co-clustering of users and contents together to define rewards. We obtain a substantial improvement of over 25% in expected stream rate with our proposed co-clustering-based calculation of the reward functions.

2 BACKGROUND AND MOTIVATIONS

Our work aims to improve on a current bandit algorithm used to select which playlists to display to users on the home of Spotify, a popular streaming service [20].

Rewards for playlist recommendation. Playlist recommendation in music streaming services is often cast as a contextual bandit problem. Given a set of playlists $\mathcal{Y}$, a contextual bandit algorithm has to decide which playlists to recommend to a user $u \in \mathcal{U}$ at a certain time iteration $t \in 1, \ldots, \infty$. At each iteration t, the algorithm has access to a "context", i.e. a feature vector $x_{t,y}$ for each playlist $y \in \mathcal{Y}$. For each recommended playlist y_t, the algorithm receives a reward $\hat{r}_t$ based on the metric being optimized and the behavior of the user. In playlist recommendation, the most frequently employed metric is the *stream rate*, i.e. a binary reward indicating whether the user has streamed the playlist y_t or not. When optimizing for stream rate, the reward signal with which the algorithm is trained is a function of the streaming time the user has listened a recommended playlist for, $\hat{r}_t = \hat{r}(s_t)$. This is a binary reward function that assigns a value of 1 to recommendations y_t for which the streaming time s_t exceeds a given threshold of 30 seconds:[1]

$$\hat{r}(s) = \begin{cases} 0 & \text{if } s \leq 30 \text{ seconds} \\ 1 & \text{if } s \geq 30 \text{ seconds} \end{cases} \tag{1}$$

As the users preferences, and thus their behavior, are unknown and non-deterministic, the contextual bandit algorithm has to trade-off *exploration* for gathering information about the user preferences over the playlist domain and *exploitation* for maximizing the expected reward. The expected reward $\mathbb{E}[r_{t,y}]$, in our case the expected stream rate, of a playlist y in context $x_{t,y}$ is unknown and has to be estimated from data collected in the recommendation process. Thus, the expected reward is modeled as a parametrized function of the type:

$$\mathbb{E}[r_{t,y}] = h(x_{t,y}; \theta_t) \tag{2}$$

The above function can then be learned from some hypothesis class $h \in \mathcal{H}$ (e.g. as a linear model or a neural network), and the previously recommended playlists and the resulting users behaviors. New recommendations are selected at each iteration by sampling according to some policy, i.e. a probability distribution that accounts for some degree of exploration versus exploitation. Commonly used policies encompass an exploitation component that selects the best playlists according to the current model $y_t = \text{argmax}_{y \in \mathcal{Y}} \mathbb{E}[r_{t,y}]$, and an exploration component that selects novel recommendations according to some criteria (e.g. uniform random sampling). As this paper focuses on the definition of novel reward functions, we refer to the literature for details on the algorithmic framework [17].

The reward signal defined in Equation 1 is, effectively, a translation of the click-through rate, used in many other domains, to music recommendation. In this paper we explore the use of the streaming time as a metric of its own and not just as a static threshold. Next we discuss the limitations of the standard stream rate metric.

Stream rate. Using the stream rate as above defined as a reward signal, the algorithm learn an expected reward that models the probability of a user to stream a given content y for an undetermined amount of time (more than a fixed threshold):

$$\mathbb{E}[r_{t,y}] = p(\text{user streams } y) \tag{3}$$

This, however, fails to take into account how the user actually engaged with the recommended content *after* a successful recommendation. In a music recommendation system, we can use the streaming time s_{t,y_t} as a proxy for the user engagement: *the longer a user streams a playlist, the higher the user engagement.*

Even when a recommended playlist is streamed for more than 30 seconds, the user behavior, and thus the streaming time, can still vary substantially, and as such may imply different things. If a user clicks on and streams a playlist once, it does not necessarily mean he or she will do it again in the future. Many streams are "explorative", i.e. the user streams a playlist for a short period to evaluate its content. Other unsuccessful recommendations may also be one-time streams or even mistaken clicks [26].

By employing a stream rate reward signal, the algorithm will not learn to distinguish from genuinely successful recommendations, with which the user engages and is likely to engage in the future, and other kinds of uninformative feedback. This is due to the fact that the algorithm gets rewarded the same amount for any recommended playlist that gets streamed, despite eventual large differences in streaming time. Our objective is, instead, to maximize the engagement of the user with respect to the recommended content. To do so, we take into account the distribution of *streaming time* and define a success metric that rewards the algorithm only when it makes recommendations with which the users *truly* engage. This can be achieved by learning as expected reward modeling the probability of a user being engaged with the recommended playlist, assuming the user streams the playlist:

$$\mathbb{E}[r_{t,y}] = p(\text{user engages with } y \mid \text{user streams } y) \tag{4}$$

By learning this expected reward model, the algorithm is able to estimate which content is more likely to keep the user engaged and, hopefully, increase the chances for the user to stream (and engage with) the content in the future. We propose different reward signals that can be used to learn the above expected reward model.

Streaming time. While taking into account the streaming time distribution is a step forward in defining an engagement-aware reward signal for playlist recommendation, the reward is independent on the user and the content that generated it. This implies that the algorithm will learn to judge streams from any kind of users and playlists with the same criteria. This might be suboptimal because different users exhibit different listening behavior and different playlists get listened in vastly different ways.

Indeed, a study comparing streaming time distributions associated with specific groups of playlists and users to the global distribution (for playlists and users, respectively) highlighted these differences. For instance, the average streaming time across all playlists is 23.75 minutes, compared to 43.28 minutes for "sleep"

[1] The 30 seconds threshold is somewhat arbitrary, and it is often determined by unrelated factors.

playlists, i.e. playlists people listen to fall asleep. This is almost double the global average, demonstrating that this type of playlists is listened to in a drastically different way. Now focusing on user types, users who like Jazz music tend to listen to recommended playlists (all playlists, not just Jazz playlists) for longer than other users, with an average of 27.16 minutes, which is significantly higher than the global average of 20.43 minutes.

This data highlights differences between listening habits of users, as well as different attitudes towards certain types of playlists. This suggests that a reward signal based on a single aka "static" threshold is essentially averaging-out these peculiarities that are key to better generalization over a vastly diverse set of users and contents. For this reason, we also propose an approach to take into account user- and content-specific patterns.

User and content patterns. Ideally, we would have a reward function for each pair of user and content, defined on the streaming behaviour of the user with respect to that content. This method, however, would only work in an ideal situation in which we had plenty of data from each user and each playlist. In practice, we have extremely sparse data, in which each user listens to only a handful of the available playlists. Applying such a method on our data would result in poor generalization to similar users and contents. We propose, instead, to group together users and playlists, on the basis of their streaming time distribution, and to define reward functions for each pair of user group and playlist group.

To extract these groups of users and playlists automatically, we need to jointly cluster users on the basis of their streaming behavior and playlists on the basis of how they get streamed. This can be achieved using *co-clustering*, a technique that simultaneously extract clusters from two different variables based on a joint distribution. We then define reward functions specific to each pair of user cluster and playlist cluster, based on the streaming time distribution of the users and playlists of each co-cluster.

3 DISTRIBUTION-AWARE REWARDS

We motivated the need for rewarding the contextual bandit algorithm with a signal based on the engagement of the user with streamed content, as opposed to a standard reward based on stream rate only. Our approach consists in using the streaming time distribution of our data as a basis for defining reward functions for user engagement, with the assumption that the longer a user streams a playlist, the higher the chances of him or her being engaged with the content.

We present three different reward functions aiming to give a sensible notion of user engagement in the playlist recommendation setting.

Mean-based reward. The first reward function consists in a binary reward that determines whether a user is engaged or not. We define a user being engaged with a playlist when he or she streams it for at least as long as global mean $\hat{\mu}$ of streaming times in our dataset. We reward the algorithm with different values based on whether the user is engaged or not. In formula (where $0 \leq \lambda_1 \leq \lambda_2 \leq 1$):

$$\hat{r}_{\hat{\mu}}(s) = \begin{cases} \lambda_1 & \text{if } s \in [0, \hat{\mu}) \\ \lambda_2 & \text{if } s \geq \hat{\mu} \end{cases} \tag{5}$$

Additive reward. A more fine-grained solution is to define "engagement levels", each associated with a specific interval of streaming time. Given the streaming time s the user spent on listening to a recommended playlist, the algorithm will observe the reward amount associated with the proper interval. We define the engagement levels based on how different the streaming time is from the mean $\hat{\mu}$. The variability of the streaming time can be quantified with the standard deviation $\hat{\sigma}$ of the streaming times, which we can use to define engagement levels in the following way:

$$\hat{r}_{\hat{\mu},\hat{\sigma}}(s) = \begin{cases} \lambda_1 & \text{if } s \in [0, \hat{\mu} - \hat{\sigma}) \\ \lambda_2 & \text{if } s \in [\hat{\mu} - \hat{\sigma}, \hat{\mu}) \\ \lambda_3 & \text{if } s \in [\hat{\mu}, \hat{\mu} + \hat{\sigma}) \\ \lambda_4 & \text{if } s \geq \hat{\mu} + \hat{\sigma} \end{cases} \tag{6}$$

Here $0 \leq \lambda_j \leq \lambda_{j+1} \leq 1$, for $1 \leq j \leq 3$, are the reward amounts associated with each level. With the same approach one could define more engagement levels at any given granularity, by adding and subtracting multiples and submultiples of the standard deviation. In our experiments, however, we considered only these four intervals.

Cumulative reward. We can define *infinite* engagement levels as a *continuous* function of the streaming time, based on the underlying distribution. This function needs to be bounded between 0 and 1 and increases with the streaming time. We obtain this "cumulative" function by first fitting a model to the relative frequency distribution, and then using the associated cumulative distribution function to define the reward. As the streaming time distribution in our data follow a clear exponentially decreasing pattern, we fit the following exponential model to our data:

$$f(s; \gamma) = \gamma e^{-\gamma s} \tag{7}$$

The cumulative distribution function associated with the above is:

$$F(s; \gamma) = 1 - e^{-\gamma s} \tag{8}$$

Once fitted, wee obtain the best $\hat{\gamma}$ value to fit our data, which can be used to parametrize the cumulative distribution function. The cumulative reward function is then defined as follows:

$$\hat{r}_{\hat{\gamma}}(s) = \max\{F(s; \hat{\gamma}), \lambda\} \tag{9}$$

4 USER- AND CONTENT-AWARE REWARDS

We defined rewards that take into account the distribution of the streaming time of the users, thereby optimizing the probability of picking recommendations that would result in high streaming times when streamed. Streaming patterns across different groups users and different group of playlists are, however, very different from each other. In this section we introduce a technique that allows us to subdivide our data into smaller sets associated with closely related groups of users and playlists, on the basis of their streaming behavior. These subsets then help us defining reward functions that are specific to the different user and playlist groups.

Co-clustering. We want to subdivide the distribution into smaller sets based on similar streaming patterns across users and playlists. As the streaming time distribution is jointly dependent on both users and clusters, we should take into account these dependency in the clustering process. This can be achieved using *co-clustering*, a technique that simultaneously clusters two (or more) different

variables based on a joint underlying distribution [6, 7, 24, 27]. Co-clustering has been successfully employed in combination with several recommendation techniques [4, 9, 16]. Co-clustering has also been used to improve recommendation based contextual bandits algorithms [18]. The difference with our work is that [18] learns a co-clustering from the given reward, e.g. click-through rate, whereas we use co-clustering to define specific metrics to optimize for each user and playlist group.

We employ the information-theoretic co-clustering technique introduced by Dhillon et al. [7]. Given two random variables, in our case one variable $U \in \mathcal{U}$ representing the user and a variable $Y \in \mathcal{Y}$ representing the playlist, and an underlying joint probability distribution $p(U, Y)$, co-clustering will output two sets of clusters $\hat{\mathcal{U}}$ and $\hat{\mathcal{Y}}$, as well as two mappings $C_U : \mathcal{U} \to \hat{\mathcal{U}}$ and $C_Y : \mathcal{Y} \to \hat{\mathcal{Y}}$ from elements onto clusters. The number of k of row clusters and ℓ of column clusters, i.e. elements of $\hat{\mathcal{U}}$ and $\hat{\mathcal{Y}}$ respectively, is fixed a priori. The aforementioned technique computes a co-clustering by minimizing the loss in mutual information [5] between the original variables and the clustered variables. This in turn equates to finding an approximate distribution $q(U, Y)$ minimizing the KL-divergence with the original distribution:

$$I(U;Y) - I(\hat{U};\hat{Y}) = D\left(p(U,Y) \| q(U,Y)\right) \tag{10}$$

where $q(U=u, Y=y) = p(\hat{u},\hat{y})p(u|\hat{u})p(y|\hat{y})$, $C_U(u)=\hat{u}$, $C_Y(y)=\hat{y}$.

The algorithm minimizes the objective in a coordinate descent fashion: at each iteration the algorithm re-computes the row and column clusters separately by minimizing the marginal objectives:

$$C_U(u) = \operatorname*{argmin}_{\hat{u}} D(p(Y|u) \| q(Y|\hat{u})) \tag{11}$$

$$C_Y(y) = \operatorname*{argmin}_{\hat{y}} D(p(U|y) \| q(U|\hat{y})) \tag{12}$$

for each row u (users) and column y (playlists). The algorithm recomputes the distributions $q(\hat{U},\hat{Y})$, $q(U|\hat{U})$ and $q(Y|\hat{Y})$ with the new found clusters. Each iteration diminishes the joint objective $D(p(U,Y)\|q(U,Y))$. The algorithm keeps iterating through the data until the maximum number of iteration is met or the change in the objective between iterations is lower than a fixed threshold.

We chose this co-clustering technique because it scales better then others with the amount of data we were dealing with. In particular, given that our data is very sparse, this techniques scales as $O(nz \cdot \tau \cdot (k+l))$, where nz is the number of non-zero elements and τ is the number of iterations, which was fixed to 20 as recommended by [7]. We considered a maximum number of 15 clusters for both rows and columns, and leave for future work how to determine the optimal number of co-clusters.

Reward functions based on co-clusters. After co-clustering the users and the playlists, we are left with $k \times l$ co-clusters, each associated with a pair $(\hat{u},\hat{y}) \in [1,k] \times [1,\ell]$ of user cluster and playlist cluster. Each co-cluster is associated with a set of data points for all users belonging to cluster $\hat{u}$ who listened to playlists belonging to cluster $\hat{y}$. These data points define a co-cluster specific streaming time distribution, which is used to define co-cluster specific reward signals. We use the distribution-aware reward functions defined in Section 3, but with respect to each co-cluster distribution. For a given user u and playlist y belonging, respectively, to cluster $\hat{u}$ and $\hat{y}$, we define the co-cluster specific binary engagement reward:

$$\hat{r}_{\hat{\mu}_{\hat{u},\hat{y}}}(s_i) = \begin{cases} \lambda_1 & \text{if } s_i \in [0, \hat{\mu}_{\hat{u},\hat{y}}) \\ \lambda_2 & \text{if } s_i \geq \hat{\mu} \end{cases} \tag{13}$$

where $\hat{\mu}_{\hat{u},\hat{y}}$ is the average streaming time computed only on the data point contained in the co-cluster $(\hat{u},\hat{y})$. Similarly, we can also define the additive reward:

$$\hat{r}_{\hat{\mu}_{\hat{u},\hat{y}},\hat{\sigma}_{\hat{u},\hat{y}}}(s_i) = \begin{cases} \lambda_1 & \text{if } s_i \in [0, \hat{\mu}_{\hat{u},\hat{y}} - \hat{\sigma}_{\hat{u},\hat{y}}) \\ \lambda_2 & \text{if } s_i \in [\hat{\mu}_{\hat{u},\hat{y}} - \hat{\sigma}_{\hat{u},\hat{y}}, \hat{\mu}_{\hat{u},\hat{y}}) \\ \lambda_3 & \text{if } s_i \in [\hat{\mu}_{\hat{u},\hat{y}}, \hat{\mu}_{\hat{u},\hat{y}} + \hat{\sigma}_{\hat{u},\hat{y}}) \\ \lambda_4 & \text{if } s_i \geq \hat{\mu}_{\hat{u},\hat{y}} + \hat{\sigma}_{\hat{u},\hat{y}} \end{cases} \tag{14}$$

with $\hat{\sigma}_{\hat{u},\hat{y}}$ being the cluster specific standard deviation. For the cumulative reward, we can fit different exponential models to each co-cluster, thereby defining the co-cluster based reward as:

$$\hat{r}_{\hat{\gamma}_{\hat{u},\hat{y}}}(s_i) = \max\left\{F(s_i;\hat{\gamma}_{\hat{u},\hat{y}}), \lambda\right\} \tag{15}$$

where $\hat{\gamma}_{\hat{u},\hat{y}}$ is the parameter found by fitting the exponential model on the co-cluster data.

The procedure for getting the co-cluster specific rewards from the data is straightforward. First compute a co-clustering over the dataset and group examples (u_i, x_i, y_i, s_i) by their associated co-cluster $(\hat{u}_i, \hat{y}_i)$. For each co-cluster $(\hat{u},\hat{y})$ compute the parameters $\hat{\mu}_{\hat{u},\hat{y}}$, $\hat{\sigma}_{\hat{u},\hat{y}}$ and $\hat{\gamma}_{\hat{u},\hat{y}}$. Finally assign the co-cluster specific rewards to each data point employing the respective parameters.

5 EXPERIMENTS

Dataset. The dataset consists of a sample of logged feedback data from user interaction with the recommendation system of Spotify, a popular music streaming service. The datasets contains about 9M data points, each corresponding to an "impression". Each impression is associated with a user u_i, a recommended playlist y_i, a contextual feature vector x_i, and the streaming time s_i (equal to 0 if the playlist was not selected). The data included feedback from more than 800K users about over 900K playlists.

The feature vectors used in the contextual bandit algorithm are composed of more than 150K features, including: (i) features of the user, such as age range, gender, location, affinity to genres; (ii) features of the playlist such as its artist, its (micro and macro) genres, diversity of songs, popularity; (iii) affinity between the user and the playlist, taking into account past interactions, such as streams, skips, likes, and saves; and (iv) other contextual information, such as the day of the week and the time of day.

The data was collected over a period of four weeks from a small percentage of the live traffic of the production system. The first three weeks worth of data were used for the training set ($\approx$ 90%) and the last one for the test set ($\approx$ 10%).

Evaluation Setup. The trained model used in the production system is an ensemble of boosted regression trees, trained by XGBoost [3]. We replicate the same training setup as the production system, including the same hyper-parameters of the learning algorithm. We train a model using the *stream rate* (akin to click-through rates) as reward signal (Equation 1) and use it as a baseline. We also use a *random threshold* approach as a second baseline, which is a binary reward function akin to that in Equation 1, but whose

Table 1: Precision, recall and F_1 results (percentage over the baseline) for the different rewards over the global distribution.

Reward for global distribution	Prec@1	Prec@5	Rec@1	Rec@5	F_1@1	F_1@5
Mean-based reward	+19.30 %	+3.27 %	+22.68 %	+2.40 %	+20.94 %	+3.13 %
Additive reward	+15.33 %	-0.62 %	+15.81 %	-0.53 %	+15.57 %	-0.60 %
Cumulative reward	+6.91 %	-0.38 %	+8.92 %	-1.04 %	+7.89 %	-0.48 %

threshold was set to a randomly chosen value in the range [0, 10] minutes, instead of being fixed to 30 seconds.

We compare the two baselines against models trained with the proposed reward functions on the *global* distribution, i.e. using the same reward function (akin to post-click). We then compare the models trained on reward defined with the use of *co-clustering*, i.e. using the reward specific for its co-cluster (akin to per co-cluster post-click). The hyper-parameters of the reward functions were determined empirically. In addition, as there are no principled way to determine a proper number of clusters, a typical situation when employing unsupervised clustering techniques, the number of row (user) and column (playlist) clusters were also determined empirically. In Tables 1 and 2, we report standard precision, recall and F_1 measures, as improvement percentage over the first baseline (stream rate based reward), for global- and co-clustering-based rewards, respectively.

Evaluating a trained offline model on logged data from an online recommender systems is know to lead to biases in the estimation of the user satisfaction metric. This is because user feedback is collected only on the recommended items selected by the given collection policy, and no feedback is observed on every item that was not recommended. To avoid this problem and perform an *unbiased* evaluation of the models performance, we used a counterfactual evaluation methodology [22], which constitutes a reliable and much less expensive alternative to online A/B testing [10].

We follow the protocol laid out by [17]. We first collect testing data using a random shuffle policy, which assigns a uniform probability score to all recommendations presented to users. Upon evaluation of a new policy, we "re-run" the history of recommendations and keep only the recommendations that happen to be selected again by the policy being evaluated. We then compute the average evaluation metric (expected stream rate) on the retained data points. The evaluation policy used was a multinomial distribution defined on the learned expected reward function. Results comparing various reward strategies are shown (the difference in expected stream rate) in Figures 1 and 2.

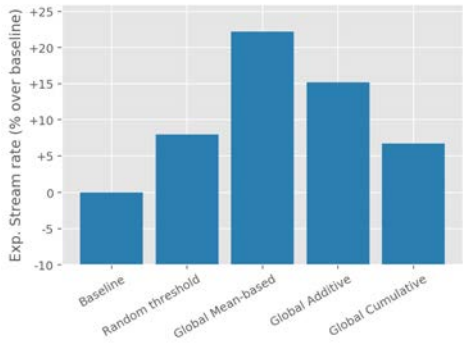


Figure 1: Expected stream rate improvement (in percentage over normalized baseline) of the random threshold method and all reward functions over the global distribution.

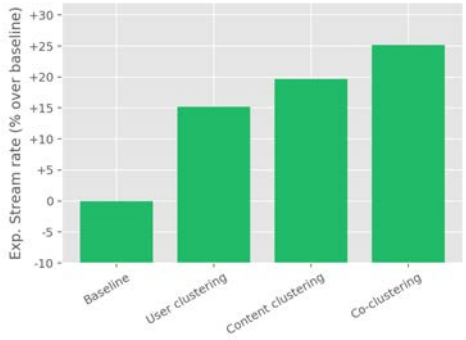


Figure 2: Expected stream rate improvement (in percentage over normalized baseline) of all mean-based reward functions over the distributions obtained by clustering users, clustering playlists and co-clustering.

6 RESULTS

Comparison of distribution aware rewards. Table 1 presents the precision and recall results comparing distribution aware reward functions with the static binary reward model. Overall, we observe that distribution aware models perform better than considering a fixed threshold. The mean based reward performs better than additive and cumulative reward, with over 19% gain in precision at the top and 22% gain in recall at the top. These results highlight that considering the distribution of user engagement is informative even without considering any user or content specific insights. Furthermore, upon randomly setting the threshold (see Figure 1), we observe an improvement in performance (expected stream rate), which further demonstrates the incompetence of the fixed threshold baseline strategy based on click-through rates.

Using co-clustering. Table 2 presents precision and recall results for the co-clustering method for all three distribution aware rewards. The co-clustering based reward model outperforms all other approaches, i.e. binary reward and global distribution aware reward. We observe a substantial improvement of over 25% in the unbiased

Table 2: Precision, recall and F_1 results (percentage over the baseline) for the different rewards over the co-cluster distributions. We experimented with several row x column numbers and we report those leading to the best performance.

Reward for co-cluster distribution	Row clusters	Column clusters	Prec@1	Prec@5	Rec@1	Rec@5	F_1@1	F_1@5
Mean-based reward	6	13	+24.74 %	+6.11 %	+26.40 %	+5.62 %	+25.56 %	+6.04 %
Additive reward	2	10	+13.34 %	+0.00 %	+12.83 %	+0.31 %	+13.09 %	+0.05 %
Cumulative reward	8	8	+13.34 %	+2.76 %	+14.76 %	+2.58 %	+14.04 %	+2.74 %

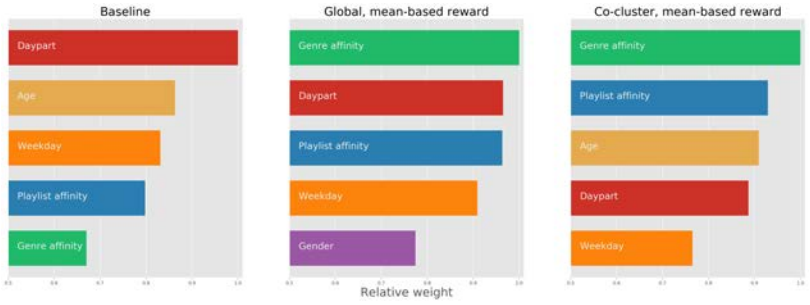


Figure 3: Relative weights of the model's features (by feature type) across the different trained models.

expected stream rate metric (see Figure 2). Similar to previously, the mean based function performs better than additive and cumulative even in the case of co-clustering of users and content.

While co-clustering of users and content allows us to jointly capture insights from both when defining the reward function, it may be hard to investigate the potential benefit offered by each. To disentangle the effect of user clusters from content clusters, we consider a special case of clustering, wherein we set the number of content clusters to 1, to give us a user-cluster, and vice-versa for content cluster. As shown in Figure 2, we observe that both the user- and content- aware reward models perform better than the static binary reward formulation, but both underperform the co-clustering based reward functions.

Comparison across metrics. The counterfactual method results indicate a consistent trend in performance across the unbiased counterfactual and offline evaluation metric, as we observe improvements in co-clustering based reward across all metrics. Furthermore, it is important to note that our estimate of expected stream rate metric is de-biased and corrected from position based bias, and as a result, provides a more reliable indicator of the benefit offered by the joint user- and content- aware reward model.

Quantifying the impact of contributing factors. We analyze how the trained models are different in terms of the features they give importance to. Figure 3 highlights the top 5 feature groups for the baseline binary reward model, the global distribution aware (mean) reward and the co-clustering reward model. As is expected, for the most important feature, the models give different weights to the different feature groups, with the baseline binary model method emphasizing the part of the day features more over others, whereas the distribution aware and user/content aware method focus on genre affinity feature group most. Indeed, the binary reward model is agnostic to user and content types, while the reward function in the co-clustering model is taking into account how users interact with content.

Looking beyond the most important feature group, among the top 5 most important features, user and content co-clustering values features that quantify user- and content- interactions: genre affinity and 7 day card affinity, whereas the baseline binary reward model values part of day and age group of users. Long term interactions like the 30 day affinity of users with content is weighted more in co-clustering than in others, which highlight the fact that training via co-clustering based reward signal allows the model to capture longer term interest and usage patterns.

Finally, we observe that as the reward signal becomes more sophisticated (binary → global mean → co-clustering), the focus moves from user-agnostic features (e.g. part of day), to user-only and content-only (e.g. user age and gender), to more sophisticated interaction based features (e.g. 30 day card affinity, genre affinity). These results highlight the importance of carefully defining reward signals that acknowledge the interaction between users and contents in our context (listening to music) by learning to weight the features appropriately.

7 CONCLUSIONS

We motivated this work by posting the hypothesis that distribution aware as well as user- and content-aware reward functions –capturing the "post-click" experience– are better than binary rewards –capturing only clicks. Extensive large-scale experiments on real world user data demonstrated the benefit of *personalizing* the definition of reward function according to how users listen to playlists **and** to how playlists are listened to.

Reward definitions lie at the forefront of systems that adopt reinforcement learning based methods to serve recommendations to users. As can be seen from our results, the reward function governs how the model is trained and has a major impact on the recommendations served. We contend that these results highlight the benefit of considering distribution and user/content aware functions to quantify the rewards, rather than creating hand-crafted reward functions. Setting fixed thresholds is very common. Indeed, a major portion of industrial search systems consider 30 seconds as the fixed threshold on dwell time to gauge user satisfaction with a clicked document [13].

Our findings reinforce the insight that when we optimize for a metric of user satisfaction, it is important to go beyond clicks. By focusing on the post-click experience, we can capture the "*intent*" of the content, as shown with "*genre*"-affinity being the most significant factor in impacting the bandit algorithm. We are also able to capture that users are different regarding how they consume content, as we see that "*playlist affinity*" is often the second most significant factor. Using clicks alone cannot surface these important components when measuring satisfaction.

REFERENCES

[1] Nicola Barbieri, Fabrizio Silvestri, and Mounia Lalmas. 2016. Improving post-click user engagement on native ads via survival analysis. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 761–770.

[2] Shi-Yong Chen, Yang Yu, Qing Da, Jun Tan, Hai-Kuan Huang, and Hai-Hong Tang. 2018. Stabilizing Reinforcement Learning in Dynamic Environment with Application to Online Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. ACM, New York, NY, USA, 1187–1196. https://doi.org/10.1145/3219819.3220122

[3] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM, 785–794.

[4] Sungwoon Choi, Heonseok Ha, Uiwon Hwang, Chanju Kim, Jung-Woo Ha, and Sungroh Yoon. 2018. Reinforcement Learning based Recommender System using Biclustering Technique. In *WSDM Workshop on Multi-dimensional Information Fusion for User Modeling and Personalization*.

[5] Thomas M Cover and Joy A Thomas. 2012. *Elements of information theory*. John Wiley & Sons.

[6] Inderjit S Dhillon. 2001. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 269–274.

[7] Inderjit S Dhillon, Subramanyam Mallela, and Dharmendra S Modha. 2003. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 89–98.

[8] Jean Garcia-Gathright, Brian St Thomas, Christine Hosey, Zahra Nazari, and Fernando Diaz. 2018. Understanding and Evaluating User Satisfaction with Music Discovery. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 55–64.

[9] Thomas George and Srujana Merugu. 2005. A scalable collaborative filtering framework based on co-clustering. In *Data Mining, Fifth IEEE international conference on*. IEEE, 4–pp.

[10] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline A/B testing for Recommender Systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 198–206.

[11] Qi Guo and Eugene Agichtein. 2012. Beyond dwell time: estimating document relevance from cursor movements and other post-click searcher behavior. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 569–578.

[12] Ahmed Hassan, Xiaolin Shi, Nick Craswell, and Bill Ramsey. [n. d.]. Beyond clicks: query reformulation as a predictor of search satisfaction. In *CIKM 2013*.

[13] Youngho Kim, Ahmed Hassan, Ryen W White, and Imed Zitouni. 2014. Comparing client and server dwell time estimates for click-level satisfaction prediction. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 895–898.

[14] Branislav Kveton, Zheng Wen, Azin Ashkan, Hoda Eydgahi, and Brian Eriksson. 2914. Matroid Bandits: Fast Combinatorial Optimization with Learning. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence (UAI'14)*. AUAI Press, Arlington, Virginia, United States, 420–429. http://dl.acm.org/citation.cfm?id=3020751.3020795

[15] Dmitry Lagun and Mounia Lalmas. 2016. Understanding user attention and engagement in online news reading. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 113–122.

[16] Kenneth Wai-Ting Leung, Dik Lun Lee, and Wang-Chien Lee. 2011. CLR: a collaborative location recommendation framework based on co-clustering. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 305–314.

[17] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 297–306.

[18] Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. 2016. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 539–548.

[19] Yiqun Liu, Ye Chen, Jinhui Tang, Jiashen Sun, Min Zhang, Shaoping Ma, and Xuan Zhu. 2015. Different users, different opinions: Predicting search satisfaction with mouse movement information. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 493–502.

[20] James McInerney, Benjamin Lacker, Samantha Hansen, Karl Higley, Hugues Bouchard, Alois Gruson, and Rishabh Mehrotra. 2018. Explore, exploit, and explain: personalizing explainable recommendations with bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*. 31–39.

[21] Rishabh Mehrotra, Ahmed Hassan Awadallah, Milad Shokouhi, Emine Yilmaz, Imed Zitouni, Ahmed El Kholy, and Madian Khabsa. 2017. Deep Sequential Models for Task Satisfaction Prediction. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 737–746.

[22] Thomas Nedelec, Nicolas Le Roux, and Vianney Perchet. 2017. A comparative study of counterfactual estimators. *arXiv preprint arXiv:1704.00773* (2017).

[23] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. 2008. Learning Diverse Rankings with Multi-armed Bandits. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. ACM, New York, NY, USA, 784–791. https://doi.org/10.1145/1390156.1390255

[24] Hanhuai Shan and Arindam Banerjee. 2008. Bayesian co-clustering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 530–539.

[25] Ning Su, Jiyin He, Yiqun Liu, Min Zhang, and Shaoping Ma. 2018. User Intent, Behaviour, and Perceived Satisfaction in Product Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 547–555. https://doi.org/10.1145/3159652.3159714

[26] Gabriele Tolomei, Mounia Lalmas, Ayman Farahat, and Andrew Haines. 2018. You must have clicked on this ad by mistake! Data-driven identification of accidental clicks on mobile ads with applications to advertiser cost discounting and click-through rate prediction. *International Journal of Data Science and Analytics* (2018).

[27] Pu Wang, Carlotta Domeniconi, and Kathryn Blackmond Laskey. 2009. Latent dirichlet bayesian co-clustering. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 522–537.

[28] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond clicks: dwell time for personalization. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 113–120.

[29] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendations with Negative Feedback via Pairwise Deep Reinforcement Learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. ACM, New York, NY, USA, 1040–1048. https://doi.org/10.1145/3219819.3219886

Pcard: Personalized Restaurants Recommendation from Card Payment Transaction Records

Min Du
University of Utah

Robert Christensen, Wei Zhang
Visa Research

Feifei Li
University of Utah

ABSTRACT

Personalized Point of Interest (POI) recommendation that incorporates users' personal preferences is an important subject of research. However, challenges exist such as dealing with sparse rating data and spatial location factors. As one of the biggest card payment organizations in the United States, our company holds abundant card payment transaction records with numerous features.

In this paper, using restaurant recommendation as a demonstrating example, we present a *personalized* POI recommendation system (Pcard) that learns user preferences based on user transaction history and restaurants' locations. With a novel embedding approach that captures user embeddings and restaurant embeddings, we model pairwise restaurant preferences with respect to each user based on their locations and dining histories. Finally, a ranking list of restaurants within a spatial region is presented to the user. The evaluation results show that the proposed approach is able to achieve high accuracy and present effective recommendations.

ACM Reference Format:
Min Du, Robert Christensen, Wei Zhang, and Feifei Li. 2019. Pcard: Personalized Restaurants Recommendation from Card Payment Transaction Records. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313494

1 INTRODUCTION

Personalized recommendation has greatly facilitated our daily life. For example, YouTube recommends a personalized video watch list based on each user's watching history as well as the reviews and ratings from the crowd [4, 5]. It is appealing to have such a personalized recommendation system for restaurant recommendations.

To recommend users with items particularly interesting to them, traditional personalized recommendation systems typically use collaborative filtering [20]. To provide recommendations to a user u, collaborative filtering first finds a group S of users that have similar tastes with u (e.g., they have rated similar items with a similar score for each), and then recommends user u with items that are liked by S but haven't been explored by u. An essential requirement of such an approach is the availability of user rating scores, such as Yelp ratings and reviews. However, such information is sparse and often hard to obtain. Furthermore, if a user never gives a rating or review, which is quite common, it is almost impossible to learn this user's preference to provide personalized recommendation. To make the matter worse, reviews and ratings are not always reliable as there are increasingly more artificially generated fake reviews.

To tackle this problem, implicit ratings have been proposed to replace explicit scores. For example, the YouTube [4] recommendation system considers a feedback as positive if a user watches a video until the end rather than aborting early. Yet similar features are not available for restaurant recommendations: knowing a person has visited a restaurant is relatively easy, but it is hard if not impossible to know whether this restaurant is being liked. Also, unexplored restaurants may be due to omission rather than being disliked. As such, simply assigning a visited restaurant with positive rating while unvisited ones with negative as in [12, 28] could make the results highly biased. A naïve idea to model user preferences is to use transaction frequencies made for different merchants as implicit ratings. However, for restaurant recommendation, this approach may not be effective due to the challenge introduced by location factor. User activities are greatly restricted by distance, especially when it comes to *where* to dine.

Our contributions. Our company holds one of the world's largest payments processing networks and handles hundreds of millions transactions every day. Most people in the US have transaction history in our network. Such scales of transaction data are invaluable towards learning users' purchasing behaviors for various recommendation purposes. This work shares our experience in mining the abundant payment card transaction history records to learn user preferences, with the particular focus of personalized restaurant recommendations as an example. Note that all user identity related attributes are processed as double hashing values, thus it is not able to identify a user based on a single transaction record. Further investigation on protecting user privacy while dealing with transaction data with advanced privacy-preserving techniques are also in place, which are not the focus of this work.

To the best of our knowledge, this is the first work that explicitly learns user preferences *based only on* card transaction records, without any rating or review data nor detailed metadata about the restaurants. Our contributions are summarized as follows.

(1) We adopt a novel embedding scheme to represent each user and each restaurant, where similar users/restaurants are represented by vector embeddings close in distance.
(2) We propose a novel approach to model user preferences over restaurants, which considers location effect while generating data samples to learn, but removes the influence of location information while modeling, for better generalization.
(3) We utilize user preferences learned from historical payment card transaction data to provide an effective ranked list of restaurants given any region.
(4) We evaluate our method on large-scale real transaction data to demonstrate its effectiveness, and a web demo was created to showcase the proposed approach.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313494

2 PRELIMINARY

2.1 Dataset

For each card payment through our network, a *transaction record* is logged which contains numerous features: user id, merchant id, merchant location, transaction time, transaction amount, and etc..

2.2 Embeddings

To model the preference of any user to any restaurant, a major challenge is to find a meaningful and effective representation of each user and each restaurant. The number of users we have is comparable to the population in the United States, which makes it unrealistic to represent each user using a naïve method, say, one-hot encoding. Dimension reduction is necessary, while a question raised by which is how to find meaningful representations in low-dimensional space for hundreds of millions of users.

Specifically, for the time window between January 2016 and April 2016, we have around 30 billion transactions available in our database, among which there are around 220 million active cardholders (users) and 78 million active merchants (e.g., online shopping websites, local supermarkets, and restaurants). For each transaction, we extract a pair of user ids (i.e., cardholder account) and merchant ids, denoted as (uid_i, mid_i), which indicates that user uid_i has made a payment to merchant mid_i.

User embeddings. All (uid, mid) pairs are grouped by distinct merchants, i.e., by mids. Thus, each merchant group contains a sequence of all users who have made payments to this merchant, represented by mid_i: $[uid_{i,1}, \ldots, uid_{i,n_1}]$. Finally, each uid list is treated as a sentence and "sentences" from all merchants are treated as an article, and word2vec [16] is used to generate an embedding for each user uid.

Merchant embeddings. All (uid, mid) pairs are firstly grouped by distinct uids. A list of all merchants that have been attended by a user is treated as a sentence and all such "sentences" are treated as an article. Word2vec is used again to generate an embedding for each merchant mid.

Each user embedding is a 200 dimension float vector while each merchant embedding has 400 dimensions. The number of dimensions for each vector space are chosen to be both efficient and effective for different tasks.

All user embeddings and merchant embeddings are stored in two tables: *global_user_embeddings* and *global_merchant_embeddings*. Standard similarity measures such as k-nearest neighbors with euclidean distance or cosine similarity demonstrate that this embedding approach, as a representation method, is able to model intrinsic features of different entities. These embeddings have been shown effective in our other applications, e.g. fraud detection.

The embeddings are created in a way such that *similar* users have closer distance in the embedding space, e.g., if they live/work in the same area, or have similar shopping behaviors. Likewise, *similar* merchants also have closer distance in the embedding space if they are in the same category, in the same area or they accommodate similar users. Note that this property is essential towards building effective recommendation systems. Recall that the core idea of collaborative filtering is to provide a user with recommendations that are liked by other users who share *similar* tastes. Using this embedding approach, we are able to provide a user with effective restaurant recommendations even if this user has never been to a restaurant within our modeled time period. If this user has ever made any card payment through our network, e.g., shopping at a local grocery store, we could recommend to this user restaurants that are liked by other users who have similar purchasing behaviors.

3 METHODOLOGY

3.1 Overview

In this section, we will describe how Pcard works using the embeddings generated for each user and each restaurant. The objective is to provide a ranked list of restaurants for a given user u in a selected geographical region RG.

Pcard proceeds in two stages: 1) in an *offline learning stage*, a machine learning model is trained to learn different users' preferences over different restaurants, which we refer to as a *preference model*; 2) in an *online recommendation stage*, given a user u and a region RG, we rank all restaurants within RG based on u's preference over each restaurant in RG, provided by our preference model.

An intuition that we leverage is that a user's preference over a restaurant is typically a *relative ranking* compared to another restaurant, instead of an *absolute ranking*. Based on this observation, we learn user preferences by comparing pairs of restaurants for each user. Specifically, we extract data samples of $[u, r_1, r_2]$ from all transaction records within our learning time period, where r_1 and r_2 stand for two restaurants u has attended. The embeddings of u, r_1, r_2 are concatenated as the input vector for our machine learning model, and a label of 1 or 0 is given to indicate whether r_1 or r_2 is more preferred (by comparing the frequencies that u visited r_1 and r_2 respectively). Finally, given a user u and a region RG, we compare all pairs of restaurants within RG for u and output a ranked list, similar to ranking in a chess tournament.

However, for preference model learning, challenges exist for extracting data samples $[u, r_i, r_j]$ to learn. The major challenges are brought by location factors. First of all, location could be the major (or only) concern while considering *where* to dine, e.g., for work lunch, but it is not fair to say a company cafeteria is preferred over a fine restaurant that is far away. Thus, location information should be considered when generating data samples to learn. Secondly, a recommendation system that is built upon data from one location, should be generalized to provide recommendations for other locations. Therefore, location information should be removed while constructing the learning model. We address the two challenges at once by comparing restaurants that are close, and use visit frequencies as the preference measurements. For one, imagine if we are surrounded by multiple restaurants which are similar in distance, choosing which restaurant reveals our preference. For another, a pair of restaurants that are close have similar "location information" in their embedding space, which are automatically omitted in preference modeling since they do not provide a difference.

Another challenge is for a pair of restaurants (r_i, r_j) nearby, it is not certain to claim that a user u prefers r_i over r_j if, say, r_i was visited for 5 times but r_j for 4 times. As a result, we only assume that a restaurant r_i is preferred over r_j if r_i is visited many times but r_j few. We do not consider any restaurant that u has not visited during the learning period, since that could be simply due to that a user is unaware of this restaurant rather than he/she doesn't like the restaurant. Our embedding technique guarantees similar

users/restaurants would have similar embeddings, which enables the effective learning of user preferences on a subset of data.

To summarize, in the offline learning stage, from all transaction records within the learning period, we extract a data sample $[u,r_i,r_j]$ if: 1) r_i and r_j are within a distance threshold; 2) u attends r_i greater than or equal to f_u times while r_j less than or equal to f_ℓ times.

Finally, after the offline preference model is successfully trained, we could serve this model in the online recommendation stage for any user u and any region RG. The Pcard recommendation system, as shown in Figure 1, works as follows: 1) find all restaurants within region RG, denoted as *restSet*; 2) compare each pair of restaurants in *restSet* for u using our preference model; 3) compute a ranked list of all restaurants in RG based on the pairwise comparison results.

Next, we will explain each of the two stages in detail: 1) train a model that learns any user's preference on any pair of restaurants; 2) design a ranking method that is able to calculate an effective ranked list based on the pairwise comparison results.

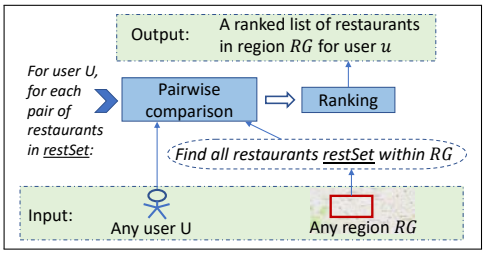


Figure 1: Recommendation system architecture

3.2 Learning a preference model

In this section, we train a preference model where given any user u and any pair of restaurants (r_i, r_j), it could output *how much* u prefers r_i compared to r_j. The resulting model serves as *Preference model* component in Figure 1.

Recall that each user is represented with an embedding such that users that are similar (e.g. have similar shopping preferences) have a smaller distance than users who are dissimilar. Such embeddings also exist for restaurants. Hence, although we're not able to cover all restaurants and all users for learning, the hope is if we have learned enough data, we are able to output meaningful results for similar users and restaurants that do not exist in training data.

To train a preference model, we take the following steps.

Step 1. Data extraction. In this step, we select all restaurant transaction records within the learning time period, and then group by user ids. For each user, we count the number of visits (i.e., visiting frequency f) to each distinct restaurant. The resulting data structure is: $\{u_1 : \{r_1: f_1, r_2: f_2, \ldots\}, u_2 : \{\ldots\}, \ldots\}$, which represents the frequency of visits to each restaurant by each user.

Step 2. Preference pair generation. With the data from step 1, we generate user preference pairs as data samples. Specifically, for all the restaurants a user u has visited, if restaurant r_i and restaurant r_j are within distance threshold D_{max}, and r_i's visiting frequency is higher than or equal to an upper threshold f_u, while r_j's visiting frequency is less than or equal to a lower threshold f_ℓ, we say u prefers r_i over r_j, and generate a preference sample $[u,r_i,r_j]$.

Step 3. Preference model training. After step 2, we have generated many preference pairs $[u,r_i,r_j]$ to learn. Since the goal is to learn a model which could output a preference over two restaurants for any user (i.e., whether u prefers the first restaurant or the second one), it is straightforward to use a binary classifier to learn this property and output "u prefers r_i" (label 1) or "u prefers r_j" (label 0). To generate training data for such model, for each preference pair $[u,r_i,r_j]$, we create two training samples: $[u,r_i,r_j]$ with label 1 to say u prefers the first restaurant, and $[u,r_j,r_i]$ with label 0 to indicate u prefers the second restaurant. For this task, we leverage a deep neural network (DNN) classifier provided by TensorFlow. As in Figure 2, for each data sample $[u,r_i,r_j]$, we first find embeddings for u, r_i and r_j in the embedding datasets and concatenate them to construct the input vector: [u embedding]+[r_i embedding]+[r_j embedding]. Optionally, we could also concatenate relevant statistics to the input vector to learn together. The output is a probability distribution of all possible classes, indicating which restaurant is more preferred. For example, ground truth label "1" is denoted with probability distribution of [1,0], indicating the first restaurant is more preferred. The process of training a DNN classifier is thus to adjust the parameters, in order to minimize the error between the actual output and the ground truth output.

After training, the preference model could be served in online recommendation stage to find a preference, for any given user u and any two restaurants. To do that, a concatenation of their embeddings is fed into the DNN classifier, the output of which is a pair of preference probabilities $[p_{j,i}, p_{i,j}]$, indicating u prefers r_i by a probability of $p_{j,i}$, and prefers r_j by $p_{i,j}$, where $p_{j,i} + p_{i,j} = 1$.

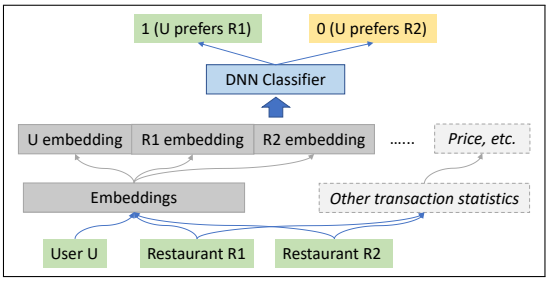


Figure 2: Preference model

3.3 Ranking from pairwise comparison

In online recommendation stage, for a given user u and a region RG, we find all pairs of restaurants within distance D_{max} and calculate their pairwise comparison results. With that, the next step is to output a meaningful ranking list for all distinct restaurants in RG. As discussed in Section 5, there are many previous works investigating the problem of ranking from pairwise comparisons.

3.3.1 Our approach. Recall that in Section 3.2, for a user u and a pair of restaurants (r_i, r_j) to compare, our preference model outputs a preference probability pair $[p_{j,i}, p_{i,j}]$. For all restaurants within region RG, i.e., $\{r_1, r_2, \ldots, r_n\}$, there are a total of $\binom{n}{2} = \frac{n(n-1)}{2}$ pairs of restaurants. For now, suppose the diameter of region RG is less than D_{max}, i.e., all pairs are comparable. Next, we need to find a meaningful way to rank all n restaurants from a total of $\binom{n}{2}$ pairwise comparison results. We use $\Pi(j)$ to denote the final ranking score for restaurant r_j, i.e., r_i ranks higher than r_j if and only if $\Pi(i) > \Pi(j)$. Since all the probability pairs are normalized in pairwise comparison results, it is reasonable to sum up all probabilities of each distinct restaurant, as the final ranking score Π for that restaurant. For instance, if the pairwise comparison results for 3

restaurants $\{r_1, r_2, r_3\}$ are $[p_{2,1}, p_{1,2}]$ (for (r_1, r_2)), $[p_{3,2}, p_{2,3}]$ (for (r_2, r_3)), $[p_{3,1}, p_{1,3}]$ (for (r_1, r_3)) respectively, there are $\Pi(1)=p_{2,1}+p_{3,1}$, $\Pi(2)=p_{1,2}+p_{3,2}$ and $\Pi(3)=p_{2,3}+p_{1,3}$. Finally, we could rank (r_1, r_2, r_3) by comparing $\Pi(1)$, $\Pi(2)$ and $\Pi(3)$. The extension to a larger region *RG* (i.e., not all pairs of restaurants within *RG* are comparable) is discussed later in Section 3.3.3. In evaluation we'll show this seemingly naïve method performs very good in practice, e.g., it is able to reduce the amount of noisy pairwise comparison results. What's more, it also has nice theoretical guarantees.

3.3.2 Theoretical property. Balanced Rank Estimation (BRE) is a ranking method proposed by Wauthier [24] to calculate the ranking score of each item from pairwise comparison results. Specifically, for the j-th item of a total of n items, its ranking score is:

$$\Pi(j) = \frac{\sum_{i \neq j} s_{i,j}(2\bar{c}_{i,j} - 1)}{2m(n)} \tag{1}$$

where $m(n)/n$ is the probability for each pairwise comparison being measured, $s_{i,j} = 1$ if a pair (i, j) is measured, and $\bar{c}_{i,j}$ is the (possibly noisy) measurement that is made.

In our case, each pairwise comparison is measured with a probability of 1, which has: 1) $m(n)/n = 1$, and hence $m(n) = n$; 2) $s_{i,j} = 1$; and 3) $\bar{c}_{i,j} = p_{i,j}$. Therefore, equation 1 reduces to: $\Pi(j) = \frac{\sum_{i \neq j}(2p_{i,j}-1)}{2n} \propto \sum_{i \neq j} p_{i,j}$. This means that BRE method is proportional to our proposed approach to calculate ranking score, which is, for each distinct restaurant, the sum of corresponding probabilities in all pairwise comparison results. Therefore, the theoretical properties proved for BRE method all proportionally hold for our approach. These include upper bound on number of inverts (cases where $p_{i,j} > p_{j,i}$ while $\Pi(j) < \Pi(i)$), and sample complexity (number of pairwise results needed to compute the final ranking), details of which could be found in [24].

3.3.3 Extension. The approach in Section 3.3.1 is based on pairwise comparison results for all pairs of restaurants in region *RG*. However, our initial assumption is that, two restaurants are only comparable if they are close (i.e., within a distance threshold D_{max}). Therefore, if two restaurants in region *RG* are further than D_{max}, it's neither valid nor accurate to compare them using our preference model, since there might be location information inscribed in their embeddings that may obfuscate the results. Hence, if *RG* is too large, we could only get incomplete pairwise comparison results, i.e., a set of pairwise results for pairs of restaurants that are within distance D_{max}, among a total of $\binom{n}{2}$ possibilities. Ranking from incomplete pairwise results is an important research topic in theory community [24]. However, the equations are not ready to use because they assume each pairwise comparison is *independently measured with the same probability*, which is often not the case in reality. Combined with the results from [24] (Equation 1), and our ranking approach in 3.3.1, we adopt another simple yet meaningful approach for ranking from incomplete pairwise results. In this approach, the ranking score of a restaurant is the sum of corresponding probabilities in all pairwise comparison results, normalized by the count of pairwise comparisons being made on that restaurant. For example, For three restaurants (r_1, r_2, r_3), if pairs (r_1, r_2) and (r_2, r_3) are close while r_1 and r_3 are far away, we have pairwise comparison results of $(p_{2,1}, p_{1,2})$, $(p_{3,2}, p_{2,3})$, then the ranking score for each restaurant is: $\Pi(1) = p_{2,1}$, $\Pi(2) = \frac{p_{1,2}+p_{3,2}}{2}$, $\Pi(3) = p_{2,3}$, based on which we could rank all three restaurants. Note that, by normalization we make sure each restaurant's measurements have the same weights in final ranking score (i.e., each restaurant is normalized to be measured once). Suppose restaurant r_j is measured by t_j times, the probability it is measured is $t_j/\binom{n}{2}$, i.e., $m_j(n)/n = t_j/\binom{n}{2}$ in Equation 1, which gives $m_j(n) = 2t_j/(n-1)$. Finally, Equation 1 becomes:

$$\Pi(j) = \frac{\sum_{i \neq j} s_{i,j}(2\bar{c}_{i,j} - 1)}{4t_j/(n-1)} \propto \frac{\sum_{i \neq j} p_{i,j}}{t_j} \tag{2}$$

which is exactly what we propose, to normalize the sum of corresponding probabilities for each restaurant by the number of times that restaurant is measured.

4 EVALUATION

This section evaluates the performance of Pcard. For all experiments we use the time period of January to February 2016 for training and validation, and March to April 2016 for testing. Note that *none* of the previous work has explored personalized restaurant recommendation in our setting, that is, without any explicit/implicit rating, and for a humongous volume of data. As a result, for baseline comparison, we simply adopt crowdsourcing approaches which are the only practical alternatives that could be evaluated on our data.

4.1 Pairwise comparison evaluation

4.1.1 Dataset. The evaluation dataset is extracted from transaction records stored in our database. For pairwise comparison evaluation, we first train a preference model on data samples generated from transaction records of a selected spatial area (e.g., Bay Area). After that, the accuracy of the preference model is tested not only on data samples of the same spatial region, but also for a different region, to test *the influence of location factors*.

Given an area, we first select all restaurant transaction records within this area. The rest of dataset generation follows the procedure described in Section 3.2. Recall that for each user u and each pair of restaurants (r_i, r_j), we generate a data sample $[u,r_i,r_j]$ if r_i, r_j are within distance threshold D_{max} and u has visited r_i more than or equal to f_u times while r_j fewer than or equal to f_ℓ times. The parameters used for all experiments are shown in Table 1a.

To validate our approach, we extract transaction records from areas ranging from smaller ones such as Bay Area, New York City, to larger ones such as California (CA) and the contiguous United States as our geographical regions to learn user preferences. Note that, if a preference pair $[u,r_i,r_j]$ appears in both training and testing datasets, we would eliminate that from testing data, to make sure that our model in fact learns preferences from embeddings instead of just memorizing training results. The number of data samples $[u, r_i, r_j]$ for each dataset is listed in Table 1b.

distance threshold	D_{max}	500 meters
visiting frequencies threshold	f_u	5
	f_ℓ	1

(a) Parameters used for all experiments

(BA: Bay Area; NYC: New York City; CA: California)

region	BA	NYC	BA & NYC	CA	US
#samples	1,951,413	2,720,745	4,679,721	24,955,210	211,519,990

(b) Number of preference pairs generated

Table 1: Datasets

4.1.2 Baseline. Our baseline mimics the default setting in current recommendation services (e.g., Yelp), which simply recommends a restaurant preferred by most users. Specifically, for each data sample $[u, r_i, r_j]$ in testing dataset, this baseline outputs a preference for r_i

if r_i has a higher visiting frequency by *all card holders* in training period, otherwise r_j.

4.1.3 Evaluation results. As discussed in Section 3.2 and Figure 2, for each extracted data sample [u,r_i,r_j], we create a concatenation of all their embeddings to train a DNN classifier using TensorFlow. The DNN classifier learns to output a pair of probabilities showing the preferences over two restaurants. For example, for an embedding concatenation [u embedding]+[r_i embedding]+[r_j embedding], the output of the classifier would be [$p_{j,i}$,$p_{i,j}$] where $p_{j,i} + p_{i,j} = 1$, showing the probabilities of liking r_i, r_j respectively. Note that, for a sample [u,r_i,r_j], there are two possible input embedding concatenations, one is [u embedding]+[r_i embedding]+[r_j embedding], and the other is [u embedding]+[r_j embedding]+[r_i embedding], which outputs [$p'_{i,j}$, $p'_{j,i}$]. In an ideal case, $p_{j,i} = p'_{j,i}$ and $p_{i,j} = p'_{i,j}$ should be true. However, due to the high dimension of input embeddings and the un-interpretable DNN classifier with millions of weights, it is not even true that [$p_{j,i}$, $p_{i,j}$] and [$p'_{i,j}$, $p'_{j,i}$] always have the same order (i.e., $p_{j,i} > p_{i,j}$ and $p'_{j,i} > p'_{i,j}$ or $p_{i,j} > p_{j,i}$ and $p'_{i,j} > p'_{j,i}$). To correct this, we test a sample [u,r_i,r_j] with the two different embedding concatenations as two separate inputs to the DNN classifier. Then, we output whether r_i or r_j is more preferred by comparing $p_{j,i} + p'_{j,i}$ and $p_{i,j} + p'_{i,j}$. The idea is that, although the order of [$p_{j,i}$, $p_{i,j}$] and [$p'_{j,i}$, $p'_{i,j}$] might not be the same, the case where the probability difference is higher means that the prediction label is more certain, where the other case might be because some part of the weights are not sufficiently trained. In our evaluation we found that this type of "two-way comparison" is more accurate than just using one type of embedding.

The prediction accuracy for each spatial area is shown in Table 2. Our method outperforms the baseline in all areas. We observe test accuracy decreases slightly when selected region size becomes larger, which could be caused by increased data varieties. Nevertheless, in the entire California state, the accuracy is still as high as 76.70%, and this accuracy does not drop further when the spatial area extends to the whole US region. Furthermore, to evaluate the influence of location information encoded in our embeddings, we use a classifier trained on New York City dataset to test preference accuracy in Bay Area, and the result is 80.77%, which is almost the same with the accuracy (81.33%) of using a classifier trained on Bay Area itself. It means that although restaurant embeddings may contain location information, by learning the preferences over pairs of restaurants within a region, DNN classifier is able to "ignore" that information which is similar between each pair of restaurants, and use other encoding information to learn a preference instead.

	BA	NYC	BA & NYC	CA	US
Pcard	81.33%	83.65%	80.64%	76.70%	76.66%
	Train on NYC and test on BA:		80.77%		
Baseline	64.27%	64.78%	64.53%	66.49%	68.38%

Table 2: Pairwise comparison evaluation results

4.2 Ranking from pairwise comparison results

In this section, we evaluate our ranking algorithm, which outputs a ranked list from pairwise preference probabilities provided by the preference model (see Figure 1 for detail).

To evaluate the correctness of ranking algorithm, previous works [19, 24] typically count the number of inverts. Given a set of n objects and binary pairwise comparisons between pairs of objects (e.g., object i > object j), an invert happens if the original pairwise order of object (i, j) is not preserved in the final ranked order. In previous work, the goal of ranking algorithm is to minimize the number of inverts.

However, in our work, there are two major differences. First, our pairwise comparison results are not the ground truth. To check final accuracy of ranking algorithm, we need to check with ground truth results from actual transaction history, instead of pairwise comparison probabilities output by the preference model. Second, it is very rare that a user u would have visited all restaurants within a region *RG*, which means that for ground truth data, we do not have all pairwise visiting frequency comparisons between any two restaurants within a region *RG* for a particular user u. Thus, we are not able to count the number of inverts in final ranking order compared to ground truth pairwise comparisons.

Instead, based on the data we have, we design a meaningful evaluation approach that could reveal the performance of our ranking algorithm. Specifically, in the testing dataset generated in Section 4.1, each sample [u,r_i,r_j] has a ground truth label of 1 or 0, meaning which restaurant between r_i and r_j is preferred. Using this dataset, the following procedures are applied to each data sample [u,r_i,r_j]: 1) find the smallest region RG_{ij} that could enclose both restaurants; 2) find all restaurants within region RG_{ij}; 3) compute pairwise preference probabilities for each pair of restaurants in region RG_{ij}; 4) rank all restaurants within region RG_{ij} based on pairwise probability results, through summing up all corresponding probabilities of each restaurant as its score; 5) check the ranking order of restaurants r_i and r_j, and see if it's the same with the ground truth label of data sample [u,r_i,r_j], i.e., whether the summed score of r_i in final ranking results is higher or lower than r_j, and whether the ground truth label of [u,r_i,r_j] is 1 or 0.

After above procedures, for all data samples in testing dataset, we get a *ranking order accuracy*, i.e., the percentage of the test samples that are correctly ordered in the *ranking results*. Then we compare this *ranking order accuracy* with the *pairwise comparison accuracy* in Section 4.1. If the *ranking order accuracy* is higher, it means that our ranking algorithm not only does not exacerbate previous step, but is even able to reduce noise in the pairwise comparison results.

The evaluation results show that the ranking algorithm has noticeably increased pairwise accuracy, from 79.95% (pairwise comparison accuracy) to 82.00% (ranking order accuracy) for Bay Area, which expresses the effectiveness of our ranking algorithm.

4.3 Overall recommendation performance

To test if our recommended restaurants are actually meaningful, for a given user, we use the whole end-to-end model shown in Figure 1 to output a ranked list of restaurants within a given region, and validate our results, by checking if this user actually attended the restaurants we recommended during the test period. We conducted this experiment for all users in Palo Alto area, and found that for over 80% of the cases, a restaurant the given user attended in test dataset is within top 30% of our ranked list. This means that if we recommend a list of 10 restaurants to a user, with large probability this user will go to one of our top 3 recommendations.

Baseline. For baseline, we adopt the crowdsourcing setting used by most services, where the recommendation list is sorted by visiting frequencies of all card holders. We found that, for over 80%

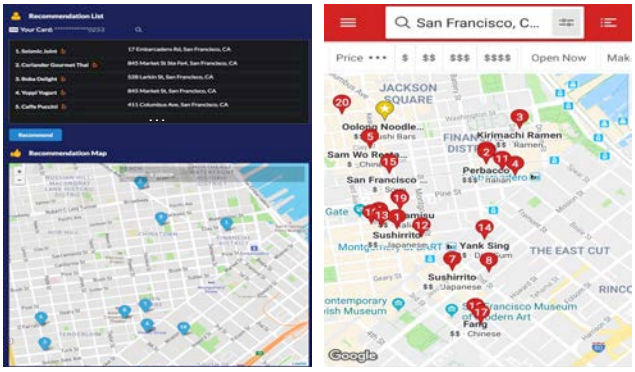


Figure 3: Screenshots of Pcard **(left) and Yelp (right) showing restaurant recommendations in downtown San Francisco**

of the cases, an attended restaurant in test dataset is only at top 46.15% of the recommendation list; while by only 65.12% of the cases could the attended restaurant appear within top 30% of the list. Apparently, Pcard provides more effective recommendations.

4.4 Interactive Demo

An interactive Pcard demo has been deployed at our company front desk to collect user feedbacks which continuously help to improve the system. A user interacts with the demo by sliding a payment card and selecting a location (range) for recommendations. The front-end submits a request for the restaurants in the area to be ranked by this user's preference. On the back-end, the system will look up all relevant embeddings, score all restaurants as described in Section 3, and finally return top k (e.g., k=10) restaurants.

A screenshot of the demo is shown on Figure 3 left. All restaurants' locations are shown with a link to each restaurant's Yelp page for more information. Compared with the Yelp recommendation result shown in Figure 3 right, which lists an overwhelming number of recommendations to be filtered further, Pcard saves user efforts to input additional information such as cuisine preference, which is already encoded in embeddings. Providing useful and interesting recommendations without using manual data labels is an important advantage of Pcard. The feedback from our demo is positive and is used to continue to improve the performance of Pcard.

4.5 Remarks

Pcard in its current state has certain limitations. First, two restaurants must be sufficiently close in order to be compared. The distance threshold of 500 meters in Table 1a is selected to be conservative. An alternative is to use restaurant density together with distance, in order to decide whether two restaurants are comparable. Second, Pcard may emphasize less expensive restaurants because of higher transaction frequency. To accommodate for this limitation, we could leverage the price range of restaurants and aggregated visiting frequencies to help with the preference model learning.

Despite the above limitations which we could address further, Pcard outperforms current POI recommendation systems in many aspects. First of all, with Pcard and our demo app as shown in Figure 3, users never have to explicitly label what they prefer or dislike in order to get useful recommendations. What's more, by only learning information from transaction records, Pcard addresses the cold start problem since we do not need any review or rating data to begin with. Also, our deployed model only contains information about double-hashed user ids and restaurant visiting frequencies, which protects user privacy. Last but not the least, Pcard could be further improved with user review data. A user could provide feedback through our app, after he/she tries a recommended restaurant.

5 RELATED WORK

Recommendation systems have been an important research topic for decades. Collaborative filtering, used by most traditional recommender systems [5, 13], recommends a user u with unexplored items that are liked by other users having similar tastes [6]. To achieve this, it needs sufficient rating scores to find *similar* users, e.g., Amazon rating scores [13], Yelp reviews [15], and Netflix's movie ratings [2]. However, explicit ratings or reviews are often not available, and could be highly biased. Implicit ratings [18] are proposed for the aforementioned problem. For instance, YouTube recommendation system treats a video as being liked by a user if the user finishes that video [4]. Similarly, if a webpage is being scrolled to the end, then it is likely that webpage is being liked [11].

However, everything is more complicated when it comes to point-of-interest (POI) recommendation [26, 27], especially restaurant recommendation, partly due to the complexity introduced by geographical location factors [1, 9, 14, 21, 26]. For restaurant recommendation, GeoMF [12] models geographical preferences of users using check-in data on location-based social networks (LBSNs), but they treat locations that users have been to as what users are likely to prefer. Zhang etc. [28] integrate implicit check-in data and explicit review data for more accurate recommendation, while analyzing user preferences based on their demographics information and restaurants attributes. In contrast, we aim to remove the influence of location factor while modeling user preferences, without any need of review data.

Deep learning has been gradually replacing many traditional algorithms in various tasks including recommendation systems [4, 8, 22, 23, 29]. Despite of the successful results, these methods still need either explicit ratings or implicit ones to begin with.

In our approach, we propose a ranking method that is able to output a ranking list of restaurants based on pairwise comparison results. Ranking from pairwise comparison results is heavily used in tournaments like chess player games, e.g., ELO rating system [25]. Research efforts in this field include to rank more efficiently [10, 17, 19], rank from incomplete pairwise results [24], and to only select top-k candidates [3, 7, 19]. Our proposed method is easy to implement, and has the same theoretical bounds with [24].

6 CONCLUSION

In this paper, we address the problem of restaurant recommendation using payment card transaction records, without any review or rating data. The experiences learnt shed light on addressing the *cold start* problem in general recommendation systems, especially for POI recommendations. For future work, we plan to utilize extra information such as restaurant statistics mined from transaction records and data from other websites to improve the accuracy, and to learn time series visiting patterns of each user for better recommendation performance.

ACKNOWLEDGMENTS

Feifei Li is supported in part by NSF grants 1816149 and 1619287. Feifei Li is also supported in part by NSFC grant 61729202.

REFERENCES

[1] Jie Bao, Yu Zheng, David Wilkie, and Mohamed Mokbel. 2015. Recommendations in location-based social networks: a survey. *GeoInformatica* (2015).

[2] James Bennett, Stan Lanning, et al. 2007. The netflix prize. In *Proceedings of KDD cup and workshop*.

[3] Róbert Busa-Fekete, Balazs Szorenyi, Weiwei Cheng, Paul Weng, and Eyke Hüllermeier. 2013. Top-k selection based on adaptive sampling of noisy preferences. In *ICML*.

[4] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *ACM RecSys*.

[5] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. 2010. The YouTube video recommendation system. In *ACM RecSys*.

[6] Pierre Dillenbourg. 1999. *Collaborative learning: Cognitive and computational approaches. advances in learning and instruction series*. ERIC.

[7] Brian Eriksson. 2013. Learning to top-k search using pairwise comparisons. In *AISTATS*.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*.

[9] Tzvetan Horozov, Nitya Narasimhan, and Venu Vasudevan. 2006. Using location for personalized POI recommendations in mobile environments. In *IEEE SAINT*.

[10] Claire Kenyon-Mathieu and Warren Schudy. 2007. How to rank with few errors. In *ACM STOC*.

[11] Hyoung-rae Kim and Philip K Chan. 2005. *Implicit indicators for interesting web pages*. Technical Report.

[12] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *ACM SIGKDD*.

[13] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* (2003).

[14] Bin Liu, Yanjie Fu, Zijun Yao, and Hui Xiong. 2013. Learning geographical preferences for point-of-interest recommendation. In *ACM SIGKDD*.

[15] Michael Luca. 2016. Reviews, reputation, and revenue: The case of Yelp. com. (2016).

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[17] Sahand Negahban, Sewoong Oh, and Devavrat Shah. 2012. Iterative ranking from pair-wise comparisons. In *NIPS*.

[18] Douglas W Oard, Jinmook Kim, et al. 1998. Implicit feedback for recommender systems. In *AAAI workshop on recommender systems*.

[19] Nihar B Shah and Martin J Wainwright. 2015. Simple, robust and optimal ranking from pairwise comparisons. *arXiv preprint arXiv:1512.08949* (2015).

[20] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* (2009).

[21] Hao Wang, Manolis Terrovitis, and Nikos Mamoulis. 2013. Location recommendation in location-based social networks using user check-in data. In *ACM SIGSPATIAL*.

[22] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *ACM SIGKDD*.

[23] Hao Wang, SHI Xingjian, and Dit-Yan Yeung. 2016. Collaborative recurrent autoencoder: Recommend while learning to fill in the blanks. In *NIPS*.

[24] Fabian Wauthier, Michael Jordan, and Nebojsa Jojic. 2013. Efficient ranking from pairwise comparisons. In *ICML*.

[25] Wikipedia. 2017. Elo rating system — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Elo_rating_system&oldid=792554580

[26] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *ACM SIGIR*.

[27] Yonghong Yu and Xingguo Chen. 2015. A survey of point-of-interest recommendation in location-based social networks. In *AAAI Workshop*.

[28] Fuzheng Zhang, Nicholas Jing Yuan, Kai Zheng, Defu Lian, Xing Xie, and Yong Rui. 2016. Exploiting dining preference for restaurant recommendation. In *WWW*.

[29] Yin Zheng, Bangsheng Tang, Wenkui Ding, and Hanning Zhou. 2016. A neural autoregressive approach to collaborative filtering. *arXiv preprint arXiv:1605.09477* (2016).

Improving Multiclass Classification in Crowdsourcing by Using Hierarchical Schemes

Xiaoni Duan
Kyoto University
Kyoto, Japan
duan@dl.soc.i.kyoto-u.ac.jp

Keishi Tajima
Kyoto University
Kyoto, Japan
tajima@i.kyoto-u.ac.jp

ABSTRACT

In this paper, we propose a method of improving accuracy of multiclass classification tasks in crowdsourcing. In crowdsourcing, it is important to assign appropriate workers to appropriate tasks. In multiclass classification, different workers are good at different subcategories. In our method, we reorganize a given flat classification task into a hierarchical classification task consisting of several subtasks, and assign each worker to an appropriate subtask. In this approach, it is important to choose a good hierarchy. In our method, we first post a flat classification task with a part of data and collect statistics on each worker's ability. Based on the obtained statistics, we simulate all candidate hierarchical schemes, estimate their expected accuracy, choose the best scheme, and post it with the rest of data. In our method, it is also important to allocate workers to appropriate subtasks. We designed several greedy worker allocation algorithms. The results of our experiments show that our method improves the accuracy of multiclass classification tasks.

KEYWORDS

task design; worker assignment; task assignment; annotation

ACM Reference Format:
Xiaoni Duan and Keishi Tajima. 2019. Improving Multiclass Classification in Crowdsourcing by Using Hierarchical Schemes. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313749

1 INTRODUCTION

Classification task, such as image categorization, is one of the most common types of tasks in crowdsourcing. Classification into multiple categories are called multiclass classification. For multiclass classification, flat classification schemes, which show all choices of categories to workers at once, is commonly used. Many crowdsourcing platforms, such as Amazon Mechanical Turk (http://www.mturk.com), provide templates of flat classification scheme.

In crowdsourcing, it is important to assign workers to tasks that they are good at in order to improve the quality of the task outputs. In multiclass classification, some workers are good at some subcategories while other workers are good at other subcategories. In the ordinary flat classification scheme, however, all the workers are assigned to the same task including all subcategories.

We propose a new approach to multiclass classification tasks. Given a flat multiclass classification task, we reorganize it into a hierarchical classification task consisting of several subtasks. We can then assign each worker to a subtask that she is good at.

For example, a flat classification task of classifying images of Canis animals into 7 categories (Siberian Husky, Alaskan Malamute, Samoyed, German Shepherd, wolf, coyote and dhole) can be reorganized into a two-level hierarchical classification task shown in Figure 1. This task consists of the following three subtasks: (1) a root task of classifying images into two top-level subcategories: (Husky, Malamute, Samoyed, Shepherd) and (wolf, coyote and dhole), which intuitively correspond to subcategories "dogs" and "wild species", respectively, (2) a task of classifying images that have been classified into the dog category further into the four breeds, and (3) a task of classifying images that have been classified into the wild species category further into the three species.

In this hierarchical task, we can assign different workers to different subtasks. If we have workers who are good at distinguishing these four dog breeds but not familiar with the wild species, we can assign them to the subtask of classifying dogs. Similarly, if we have workers who are familiar with wild animals but not familiar with dog breeds, we can assign them to the subtask of classifying wild species. Some workers are assigned to the root task of distinguishing dogs and wild species, which is relatively easy. We expect this assignment would improve the overall classification accuracy.

Another advantage of hierarchical schemes is simplicity of tasks. We expect that each subtask with a fewer choices requires less cognitive load and less execution time than the original flat task.

On the other hand, a disadvantage of hierarchical schemes is its cost. In a two-level hierarchical classification scheme, each data goes through two subtasks, thus the total number of the required task instances are double of that of the flat scheme. However, when we have enough budget and enough time, a method that can improve the accuracy of the output by fully utilizing these resources is useful.

Another standard method of improving the quality of crowdsourcing tasks by using more resources is the majority voting: we assign multiple workers to one task, and take majority vote. Improvement by the use of votes by two workers can be bigger than the improvement by the use of two-level hierarchical scheme. However, the majority voting and hierarchical reorganization are not exclusive with each other. For example, we can assign five workers to each subtask in a two-level hierarchical scheme, and take their majority vote. It requires ten workers for each data, so we could instead assign ten workers in a flat classification scheme. We should compare the accuracy of these two schemes. In other words, we

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313749

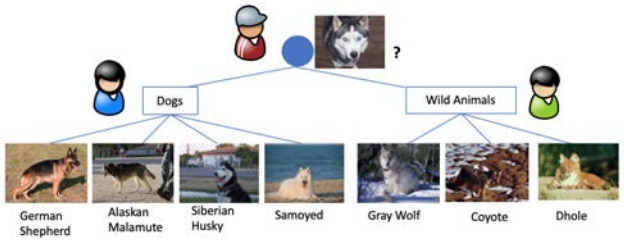


Figure 1: Example of a hierarchical classification scheme

should compare the benefit of two-level schemes with the benefit of doubling the number of workers in the majority voting scheme.

In the majority voting, we can assign as many workers to one task as we want, but the improvement is likely to stop after we reach some number of workers. For example, increasing the number of workers from five to ten is not likely to improve the accuracy significantly. We therefore expect that the benefit of the two-level schemes is bigger than the benefit of doubling the number of workers when we have already assigned enough number of workers.

In our hierarchical reorganization approach, there are many ways to reorganize a given flat classification task into a hierarchical one, and we need to choose an appropriate hierarchy. In this paper, we propose a method of selecting an appropriate hierarchy. We first post a flat classification task with a part of data as a worker qualification task, and collect statistics on each worker's ability. Based on the obtained statistics, we simulate all candidate hierarchical schemes and estimate their expected accuracy. We then choose the best scheme, and post it with the rest of data.

In our method, it is also important to allocate workers to appropriate subtasks in the simulation step, and when posting a hierarchical task. We designed several greedy worker allocation algorithms.

In order to evaluate our method, we ran several classification tasks on Amazon Mechanical Turk. The result shows that hierarchical schemes can improve the accuracy of multiclass classification if we choose appropriate hierarchies.

The contribution of this paper is summarized as follows:

- we propose a hierarchical reorganization approach for improving the accuracy of multiclass classification tasks,
- we propose a method of selecting appropriate hierarchy and a method of assigning workers to appropriate subtasks, and
- we experimentally confirmed that our method can improve the accuracy of multiclass classification tasks.

2 RELATED WORK

Hierarchical classification has been studied much in machine learning [4, 10, 14], but only in a few studies in crowdsourcing.

Vempaty et al. [15] proposed a method of transforming a flat multiclass classification (e.g., into multiple dog breeds) into a combination of many binary classifications (e.g., large/small dogs, and long/short tail dogs). They represent the results of the binary classifications of a data by a sequence of 0/1 and decode it into a class (e.g., a breed) based on coding theories. Their method uses a combination of binary classifications, not hierarchical one. In addition, appropriate binary classifications need to be specified manually.

Weng et al. [16] proposed a method of reorganizing a task of selecting data satisfying multiple conditions. Instead of asking all conditions at once, they reorganize a task into a hierarchical one where questions are asked one by one. In this scheme, they can reduce the number of tasks by narrowing down the candidates in earlier steps. Their goal is to reduce the workload in selection tasks while our goal is to improve the accuracy in classification tasks.

Otani et al. [12] proposed a method of aggregating multiple answers in flat multiclass classification tasks where classes have hierarchical relationships. For example, suppose two workers classify a given book into Society.Sociology category, two workers classify it into Science.Medical, and one worker classify it into Science.Others. In this case, Society.Sociology and Science.Medical have two votes, but they choose the latter because three of the five workers voted to Science category. The task shown to workers in their method is a flat multiclass classification task, and discovery of hierarchical relationship is not discussed.

Chilton et al. [2] and Bragg et al. [1] proposed a workflow for creating hierarchical taxonomy by crowdsourcing. In their methods, each worker classifies a small fraction of the data, and their methods create a globally consistent taxonomy based on the answers. Their purpose is to create a semantically meaningful hierarchy while our purpose is to select a hierarchy that best improve the accuracy of a classification task. As shown later in our experiment, the best hierarchy is not necessarily a semantically meaningful hierarchy.

There have also been many proposals of methods of assigning workers to appropriate tasks. The Hungarian algorithm [9] is a classical algorithm for the optimal task-worker assignment. However, we cannot simply use the Hungarian algorithm for assigning workers to hierarchical subtasks because the accuracy of subtasks in hierarchical classification tasks have inter-dependency: if the results of the upper subtasks are wrong, it cannot be corrected in the lower subtasks. In addition, the Hungarian algorithm is too slow when we have many workers. We develop new greedy algorithms for assigning workers to hierarchical classification subtasks.

Mavridis et al. [11] proposed a worker allocation method based on the hierarchical relationship between required skills. Their skill hierarchy, however, does not correspond to classification hierarchy, and we cannot use it for worker allocation in our method.

Our hierarchical schemes require more workers, and such resources can instead be used for repeated labeling as explained before. Sheng et al. [13] showed that repeated labeling does not always improve the output quality. For deciding whether we should assign more workers or not, Gao et al. [5] designed two models to estimate the profit of additional tasks. Kamar and Horvitz [7] also proposed a method of deciding whether to hire more workers or not. Karger et al. [8] has shown a theoretical result on the trade-off between the quality and the number of workers for each task in multiclass classification tasks. We can use these methods to determine whether we should introduce hierarchical schemes or should increase the number of workers for one data instead.

3 OUR METHOD

In this section, we explain the details of the four steps of our method: (1) posting qualification tasks, (2) generating candidate hierarchical schemes, (3) estimating ability of workers at subtasks in the

candidate schemes based on the statistics obtained in the qualification tasks, and (4) allocating workers to subtasks in the candidate schemes for estimating the expected accuracy of the schemes.

3.1 Qualification Task

As explained in Section 1, we first post an ordinary flat classification task with a part of data as a qualification task. Qualification tasks are popularly used in crowdsourcing for eliminating spammers. We use it also for calculating the accuracy of each worker for each category, and creating a confusion matrix of each worker.

For these purposes, we need the ground truth for the data used in the qualification task. In our experiment, we used data with the ground truth. When it is not available, we can estimate it by majority voting or more recent alternatives [3, 6, 17].

In our experiment, 20 to 30 items from each category was enough to obtain stable statistics. We use the same data set for all users.

3.2 Hierarchical Scheme Generation

We then generate candidate hierarchical schemes, and run simulations of them. There are, however, too many possible hierarchies to run simulation of. In this paper, we only consider two-level schemes because the number of levels corresponds to the number of subtasks each data goes through, and the cost of the whole classification task is proportional to the number of levels. The scheme in Figure 1 is an example of a two-level scheme with one root task, two top-level categories, two leaf tasks for them, and seven final leaf classes.

Because the number of two-level schemes is still large, we only consider schemes with at most $\lfloor n/2 \rfloor$ top-level categories when we have n leaf classes. Schemes with more top-level categories have at least one top-level category with only one leaf class, and are probably inefficient. With this restriction, the number of candidate schemes for n final classes is $\sum_{k=2}^{\lfloor n/2 \rfloor} S(n,k)$, where $S(n,k)$ is the Stirling numbers of the second kind.

3.3 Estimation of Worker Ability

Given a candidate scheme, we estimate two types of accuracy of each worker based on the results of the qualification task: (1) accuracy in the root task, and (2) accuracy in each leaf task.

When we estimate the root task accuracy of a worker, if the worker has classified an item of one class into another class within the same top-level category in the qualification task, it is regarded as a correct answer. For example, suppose we estimate the accuracy of a worker at the root subtask in the scheme in Figure 1. If she has classified a Husky image into the Samoyed class in the qualification task, it is regarded as correct. We calculate the ratio of the correct answers to the all answers in the qualification task.

On the other hand, when we calculate the accuracy of a worker at a leaf task corresponding to a top-level category A, we only consider items in the qualification task whose correct classes belong to A. In the qualification task, the worker sometimes classifies an item in A into some wrong leaf class c which does not belong to A. When the worker is really assigned to the leaf task for A, she cannot classify the item to c because c is not in the choices. In such cases, we determine which class she would choose based on the probability distribution of her answer in the qualification task.

For example, suppose we estimate the accuracy of a worker at the leaf task for the top-level category "dog" in the scheme in Figure 1. In the qualification task, the worker classified 6 images of Husky: 3 into the Husky class, 2 into Malamute, and 1 into wolf. When the worker is assigned to the leaf task for the dog category, she cannot classify an image into the wolf class, which is not in the choices. When calculating her accuracy in this subtask, we assume that she would classify the image she misclassified into wolf, either into Husky or Malamute in the probability 3/5 and 2/5, respectively.

We also collect statistics on task speed of workers. In order to minimize the response time of a task, the number of tasks assigned to workers should be proportional to their task speed.

3.4 Worker Allocation Algorithms

We have calculated the accuracy of workers at subtasks in the candidate scheme. In order to estimate the expected overall accuracy of the scheme, we also need to decide worker allocation to subtasks.

Notice that we cannot simply allocate each worker to a subtask for which she has her best accuracy. If we did it, we would have no worker for difficult subtasks. We need to assign appropriate number of workers to each subtask for minimizing response time.

We can assign a worker to more than one subtasks, but even if we do it, the worker would only work on a subtask with the best reward/time ratio until all the data at the subtask is processed. In this paper, we assume that each subtask has far more data than one worker can process. We, therefore, only consider simple worker allocation where each worker is allocated only to one subtask.

There are too many ways to allocate workers to subtasks, and we cannot calculate expected accuracy for all of them. Even with only three subtasks and 20 workers, we have $3^{20} > 3$ billion ways to allocate workers. To decide worker allocation without an exhaustive search, we designed four greedy worker allocation algorithms.

Algorithm 1 gives priority to workers whose accuracy largely changes depending on tasks. It is important to assign such workers to appropriate tasks. On the contrary, we give the least priority to workers who have the same accuracy for all tasks.

After calculating accuracy of workers for subtasks in the given scheme, we create a worker list l_r, which is sorted by the accuracy of workers at the root subtask in descending order. We also create worker lists $l_1, \ldots, l_k$, each of which is sorted by the accuracy of workers at the corresponding leaf subtask. Let $r_r^u, r_1^u, \ldots, r_k^u$ be u's ranks in $l_r, l_1, \ldots, l_k$. We calculate the variance of $r_r^u, r_1^u, \ldots, r_k^u$ for each worker. Here we measure the variance of u's ability by using her ranks instead of the accuracy values because the accuracy values for different subtasks are incomparable. We then produce a list of workers, W, sorted by the variance in descending order.

We also maintain variables $j_r, j_1, \ldots, j_k$ storing how many jobs are remaining at the root task and each leaf task. Suppose we have m data items, n leaf classes, and a hierarchical scheme with k top-level categories, which includes $n_1, \ldots, n_k$ leaf classes, respectively. We may not know the ratio of data in each leaf class, so we assume that each leaf class has m/n items. We initialize j_r to m and initialize each j_i $(i = 1, \ldots, k)$ to either $m * n_i/n$ if $n_i > 1$, or 0 if $n_i = 1$. We are also given w_u for each u, the relative task speed of u.

We scan W starting from its top, and for each u in W, we find the largest value among $r_r^u, r_1^u, \ldots, r_k^u$, i.e., the best subtask for u.

If it has remaining jobs, we assign u to that task and subtract the expected number of jobs the worker u will do, i.e., $2m * w_u/\Sigma_j w_j$, from the number of its remaining jobs. If it has no remaining job, we assign u to the next best task for u.

Algorithm 2 is same as Algorithm 1 except that we use the accuracy values instead of ranks. Because the accuracy values in different tasks are incomparable, we normalize them to the standard scores. Let a_i^u be the accuracy of worker u for the task i. For each task i ($i = r, 1, \ldots, k$), we calculate the mean of accuracy $\bar{a}_i = \sum_u a_i^u/|W|$ and the standard deviation $\sigma_i = \sqrt{\sum_u (a_i^u - \bar{a}_i)^2/|W|}$, where $|W|$ is the number of workers. We then calculate s_i^u, the standard score of the worker ability, by $s_i^u = (a_i^u - \bar{a}_i)/\sigma_i$. We use the variance of s_i^u instead of the variance of r_i^u when we sort W and when we select the best subtask for u.

Algorithm 3 is the opposite of Algorithm 1 and 2. It gives priority to tasks whose accuracy largely changes depending on the allocated workers. In this case, we do not need to use rankings or standard scores. For each task i, we calculate the average accuracy $\bar{a}_i = \sum_u a_i^u/|W|$ and the variance $\sigma_i^2 = \sum_u (a_i^u - \bar{a}_i)^2/|W|$. We then give priority to a task i with the largest σ_i^2, and assign it a remaining worker with the highest accuracy for i.

Algorithm 4 gives priority to a task with the largest ratio of remaining jobs. It repeatedly selects a task corresponding to the largest value among $j_r/j_r^0, j_1/j_1^0, \ldots, j_k/j_k^0$, where $j_r^0, j_1^0, \ldots, j_k^0$ are the initial values of $j_r, j_1, \ldots, j_k$ explained before, and allocates a remaining worker with the highest accuracy for that task. We repeat it until all the workers are assigned.

For all combinations of candidate schemes and our four allocation algorithms, we compute worker allocation, and compute the expected overall accuracy of the task under that worker allocation. We then choose the combination that achieved the highest accuracy, post the classification task using that hierarchical scheme, and assign workers to subtasks by using that worker allocation.

4 EXPERIMENTS

We ran experiments on two data sets on Amazon Mechanical Turk (AMT). The results are explained in this section.

4.1 Experiment of classifying Canis Animals

In the first experiment, we compare the accuracy of flat and hierarchical classification tasks classifying images of 7 kinds of Canis animals explained in Section 1. In this experiment, we only consider hierarchical schemes with two top-level categories, which we call A and B. We have $2^7 - 1 = 63$ such schemes (excluding one scheme that is equivalent to the flat classification).

We collected over 100 images of each of these seven kinds of Canis animals, and obtained 800 images in total. We posted a flat classification task with randomly chosen 200 images on AMT as a qualification task. The remaining 600 images are used in the main task later. One task instance consists of 20 images. We obtained information on 153 workers, and removed 11 workers as spammers.

We then estimated overall accuracy of 63 hierarchical schemes by allocating workers with Algorithm 1 and 4. For all 63 schemes, both Algorithm 1 and 4 achieved better expected accuracy than the accuracy observed in the flat classification task posted as the qualification task, which was 0.746. In addition, for all 63 schemes, Algorithm 1 achieved better accuracy than Algorithm 4. Table 1 lists schemes that achieved the three best and the three worst accuracies with Algorithm 1. The columns A and B show the leaf classes allocated to the two top-level categories.

Table 1: Best/Worst Hierarchical Schemes with Algorithm 1

rank	accuracy	A	B
1	0.866	(Samoyed, wolf, coyote)	others
2	0.862	(Malamute, Husky, dhole)	others
3	0.862	(Shepherd, Malamute, Husky)	others
⋮	⋮	⋮	⋮
61	0.807	(Shepherd)	others
62	0.800	(Malamute, wolf, dhole)	others
63	0.776	(Malamute, coyote)	others
flat	0.746		

Table 2: Confusion Matrix for All Users (bold fonts: > 0.1)

	Malamute	coyote	dhole	wolf	Shepherd	Samoyed	Husky
Malamute	0.611	0.002	0.009	0.016	0.024	0.038	**0.300**
coyote	0.018	0.618	**0.134**	**0.182**	0.016	0.018	0.015
dhole	0.010	**0.209**	0.698	0.030	0.012	0.018	0.015
wolf	0.053	**0.118**	0.026	0.674	0.022	0.029	0.077
Shepherd	0.009	0.009	0.022	0.008	0.917	0.018	0.017
Samoyed	**0.104**	0.006	0.045	0.020	0.015	0.755	0.055
Husky	**0.218**	0.010	0.008	0.040	0.028	0.023	0.674

Table 3: Accuracy of Real Tasks on AMT

Scheme	Total	Root	Sub-A	Sub-B
best scheme in Table 1 (3 to 4)	0.875	0.970	0.853	0.850
A: (wolf) B: others (1 to 6)	0.765	0.900	0.880	0.749
worst scheme in Table 1 (2 to 5)	0.590	0.610	0.444	0.644
flat (majority voting)	0.833	-	-	-
flat (weighted majority voting)	0.767	-	-	-

Table 2 shows the confusion matrix calculated from the answers by the workers in the qualification task. For example, it shows that 0.2% of Malamute images were classified as coyote. Misclassification with a probability higher than 0.1 is shown in bold fonts. This table shows that Malamute and Husky are most difficult to distinguish, coyote and dhole are the next, and coyote and wolf follow.

In all the top three schemes shown in Table 1, the most difficult pair, Malamute and Husky, are in the same top-level category, while they are in different top-level categories in the two worst schemes. It suggests that we can improve overall accuracy by allocating these two kinds to the same top-level category, and assigning workers who are good at distinguishing them to the leaf task corresponding to that top-level category. Another confusing pair, wolf and coyote, are also in the same top-level category in all the top three schemes.

We then posted three hierarchical classification tasks on AMT to know their real accuracy (i.e., not the estimation by our simulation). We chose the best scheme and the worst scheme in Table 1, and also the best 1-to-6 scheme (i.e., one class in A and 6 classes in B) as a medium-level scheme. Because the best scheme is 3-to-4 scheme and the worst scheme is 2-to-5 scheme, these three schemes include all n-to-m cases. Table 3 lists the three schemes and their results.

We have 600 images excluding 200 images used in the qualification task. We randomly divide them into three sets of 200 images and allocate them to the three hierarchical tasks. All 142 workers are invited to the three tasks, which were posted at the same time. We also posted a flat classification task with the same 600 images.

In all the tasks, five workers are assigned to each image. We used two methods to aggregate the five answers: majority voting and the standard EM-based weighted voting method [3]. When we have a tie in majority voting (e.g., 2 votes to wolf, 2 to coyote, and 1 to dhole), we randomly choose one of the classes involved in the tie.

We show the result in Table 3. Because the accuracy of the flat classification with the weighted voting is lower than that with the simple majority voting, we only used the simple majority voting in the hierarchical tasks. The hierarchical scheme which was best in the simulation achieved higher accuracy than the flat scheme. It confirms that we can improve the accuracy of a multiclass classification task by reorganizing it into a hierarchical task.

However, the accuracy of the other two hierarchical schemes are worse than that of the flat scheme although their expected accuracy was higher than it. This result suggests that the choice of hierarchical schemes significantly influence workers' performance, and if you choose an inappropriate scheme, it can result in the accuracy that is worse than the expected accuracy estimated from the workers' performance in the flat classification task.

Also note that none of the best three schemes in Table 1 is the semantically meaningful one shown in Figure 1. It shows that it is not easy to manually determine the good hierarchy, and we should compute it based on the ability of currently available workers.

To confirm that good hierarchies depend on the workers available at the time, we run the experiment of classifying Canis animals twice. In the previous run, the most confusing pair, Malamute and Husky, are allocated to one top-level category, and another confusing pair, wolf and coyote, are allocated to the other top-level category, in the three best schemes. In the second run, these two pairs are assigned to the same top-level category in the two best schemes. This happened because the set of workers who are good at distinguishing Malamute and Husky, and the set of workers who are good at distinguishing wolf and coyote, were relatively disjoint in the previous run, but have more overlaps in the second run. As a result, allocating both pairs to the same top-level category becomes a better strategy. As shown in this result, the best hierarchical schemes depends on the set of workers currently available.

4.2 Experiment with Reptiles and Amphibians

Next, we compare flat and hierarchical tasks classifying images of the following 10 kinds of reptiles and amphibians: Komodo dragon, frilled agama, chameleon, gecko, iguana, skink, giant salamander, tuatara, and basilisk. In the previous experiment, we only generated hierarchical schemes with two top-level categories, but in this experiment, we use 2 to 5 top-level categories.

We collected over 100 images for each class and obtained over 1000 images in total. We then published a qualification task on AMT with 200 images, collected answers from 307 workers, and we eliminated 44 workers as spammers, which result in 263 workers.

We computed the confusion matrix from their answers. We omit the details, but the most frequent misclassification was 21.7% of tuatara images classified into the iguana class, which look similar and better known. Compared with Canis animals, the distinction between confusing pairs and the others are not clear, and the confusion matrix is less symmetric. This is because the 10 classes in this task include both well-known classes and less-known classes.

We then generated all the hierarchical schemes with 2 to 5 top-level categories, and ran Algorithm 1 to 4. We obtain the best accuracy with Algorithm 2, which uses the standard score of the accuracy of workers. Table 4 shows some results obtained with Algorithm 2. It lists the best scheme, which has four top-level categories, the best one among those with five top-level categories (202nd), the best one among those with three top-level categories (6422nd), and the worst scheme. The worst scheme has five top-level categories, four of which include only one leaf class. The best to the 201th schemes all have four top-level categories.

In the best scheme shown in Table 4, basilisk is allocated to the same top-level category as frilled agama and chameleon, to which basilisk images are often misclassified. In this scheme, newt and giant salamander, which are often misclassified to each other, are also allocated to the same top-level category. Newt and giant salamander are allocated to the same top-level categories in most schemes with high accuracy (although it is not shown in Table 4). This is similar to what happened in the previous experiment.

We posted hierarchical classification tasks on AMT with the four schemes shown in Table 4, and assigned 263 qualified workers to the subtasks by Algorithm 2. We divide 800 images that were not used in the qualification task to four tasks so that each of them has 200 images. In each subtask, each image is classified by five workers, and we take the majority votes.

In this experiment, we calculate the accuracy of the flat scheme by using the answers in the qualification task. In the hierarchical tasks, we take majority votes by 5 workers, but in the flat task, we take majority votes by 10 workers because our two-level schemes require twice as many workers as the flat scheme. We randomly choose 10 users who participated in the hierarchical classification tasks, take the majority votes of their answers in the flat qualification task, and calculate the overall accuracy. We repeat the random selection of 10 users five times and took the average of the accuracy. We only choose users who also participated in the hierarchical tasks in order to be fair. If we include workers who did not participate in the hierarchical tasks, the accuracy of the flat scheme was lower.

Table 5 shows the results. It also shows the result we would have if we take majority votes by 3 workers for hierarchical schemes and 6 workers for the flat scheme. When we calculate the accuracy in these cases, we randomly choose 3 (or 6) workers from 5 (or 10) workers and take their majority votes. We repeat it five times and calculate the average of the accuracy.

When we take majority votes by 5 and 10 workers, our top and the 202th schemes achieved higher accuracy than the flat scheme. The 202th scheme achieved better accuracy than the top scheme. However, the 6422th scheme shows lower accuracy than the flat scheme, so it is again important to choose a good hierarchy.

On the other hand, when we take majority votes by 3 and 6 workers, the flat scheme achieved higher accuracy. The results in these two settings shows that the improvement we obtain by increasing the number of workers from 6 to 10 in the flat scheme is not as big as the improvement we obtain by increasing it from 3 to

Table 4: Part of Hierarchical Schemes and their Expected Accuracy Obtained with Algorithm 2

rank	accuracy	*A*	*B*	*C*	*D*	*E*
1	0.918	skink, tuatara	Komodo dragon, gecko	agama, chameleon, basilisk	others	-
202	0.903	basilisk	Komodo dragon, gecko	skink, agama, iguana	newt	others
6422	0.878	iguana, gecko, tuatara	skink, Komodo dragon, newt, basilisk	agama, chameleon, giant salamander	-	-
last	0.626	Komodo dragon	chameleon	skink	iguana	others

Table 5: Result with Majority Votes by 5 Workers (10 for Flat Scheme) and 3 Workers (6 for Flat Scheme)

Scheme	by 5 (10)	by 3 (6)
1st	0.885	0.855
202th	**0.890**	0.845
6422th	0.830	0.820
last	0.860	0.755
flat (majority voting)	0.863	**0.860**
flat (weighted majority voting)	0.730	-

Table 6: Average Task Execution Time (in second) for Reptile and Amphibian Images

	flat task			hierarchical task	
	all u	u at root	u at leaf	root	leaf
best	916.798	2372.66	578.958	2311.78	278.880

5 in the hierarchical schemes, and as a result, the accuracy of some hierarchical schemes become higher than that of the flat scheme when we have 5 (or 10) workers. This result suggests that our hierarchical scheme is useful when we have a budget to hire more workers, but we have already allocated enough number of workers to each task, and the benefit of doubling the number of workers is likely to be smaller than the benefit of hierarchical schemes.

4.3 Temporal Factors

When we take majority voting by the half number of workers in a hierarchical task, the number of required task instances is the same as in the flat task. We should also compare the workload of one task instance. Subtasks in hierarchical tasks have a fewer choices, so it may require lighter cognitive load than the tasks in the flat scheme.

To confirm it, we measured the average task execution time for a task instance including 20 reptile and amphibian images in the following cases: the average over flat tasks by all workers, the average over flat tasks done by workers who are allocated to the root task when they worked in the best hierarchical scheme, the average over flat tasks done by workers who are allocated to the leaf tasks when they worked in the best hierarchical scheme, the average over the root task in the best scheme, and the average over the leaf tasks in the best scheme. Table 6 shows the results.

The average execution time of flat tasks is 916.798, but the average over flat tasks done by workers who were allocated to the root task in the best hierarchical scheme is 2372.66. It means that our algorithm tend to allocate slow workers to the root task in this scheme. The average execution time of the root task in the best hierarchical scheme is almost the same. This is reasonable because both of them show workers all the leaf classes as the choices. The average execution time of the leaf tasks in the best hierarchical scheme is shorter than that of the flat task done by the same workers. This is also reasonable because the leaf tasks of the hierarchical scheme shows fewer choices to workers. The sum of the average execution time of the root tasks and the leaf tasks is bigger than the double of that of the flat task. Therefore, when we have a time constraint, hierarchical schemes have a disadvantage.

We also measured the time required by the worker allocation algorithms. In the experiment with images of Canis animals, it processed 63 schemes in a few minutes. In the experiment with images of reptiles and amphibians, we have a huge number of schemes because we generate schemes with 2 to 5 top-level categories, but our algorithms processed all of them within a day. This cost is not impractical when we classify a large number of data in crowdsourcing because such a task usually takes far longer. If we have more classes, and want to generate schemes including a larger number of top-categories, the time required in this step can be prohibitive. In that case, we should limit the number of top-categories. On the other hand, the time complexity of the Hungarian algorithm is in $O(n^3)$, and it is also infeasible when n is large.

5 CONCLUSION AND FUTURE WORK

In this paper, we propose a new approach to multiclass classification tasks in crowdsourcing. By reorganizing a task into a hierarchical classification task consisting of multiple subtasks, we can allocate workers with different expertise more appropriately to subtasks.

In our method, we generate all reasonable schemes, and estimate their expected accuracy by virtually assigning workers to its subtasks. For this step and also for the real execution of hierarchical tasks, we developed four greedy worker allocation algorithms. They give priority either to workers whose ability largely changes depending on tasks, to tasks whose accuracy largely changes depending on workers, or to tasks with higher remaining job ratio.

We posted hierarchical tasks on AMT with two data sets. We confirmed that we can achieve higher accuracy than the flat tasks if we choose appropriate hierarchical schemes, and also confirmed that our method can choose appropriate schemes. Because the best algorithm among our four algorithms depends on the data set, the best strategy is to use all of them to run simulation for all candidate schemes, and choose the one that achieved the best accuracy.

Our worker allocation algorithms have large room for improvement. We also need a method for efficiently enumerating promising candidate schemes because we cannot enumerate all schemes when we have many classes. They are remaining issues for future work.

6 ACKNOWLEDGMENTS

This work was supported by JST CREST Grant Number JPMJCR16E3, Japan.

REFERENCES

[1] Jonathan Bragg, Mausam, and Daniel S. Weld. 2013. Crowdsourcing Multi-Label Classification for Taxonomy Creation. In *Proc. of HCOMP*. 25–33.

[2] Lydia B. Chilton, Greg Little, Darren Edge, Daniel S. Weld, and James A. Landay. 2013. Cascade: Crowdsourcing Taxonomy Creation. In *Proc. of SIGCHI*. 1999–2008.

[3] Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied statistics* 28, 1 (1979), 20–28.

[4] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. 2013. Learning hierarchical features for scene labeling. *IEEE TPAMI* 35, 8 (2013), 1915–1929.

[5] Jinyang Gao, Xuan Liu, Beng Chin Ooi, Haixun Wang, and Gang Chen. 2013. An online cost sensitive decision-making method in crowdsourcing systems. In *Proc. ACM SIGMOD*. 217–228.

[6] Panagiotis G Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality management on amazon mechanical turk. In *Proc. of ACM SIGKDD workshop on human computation*. 64–67.

[7] Ece Kamar and Eric Horvitz. 2015. Planning for Crowdsourcing Hierarchical Tasks. In *Proc. of AAMAS*. 1191–1199.

[8] David R Karger, Sewoong Oh, and Devavrat Shah. 2013. Efficient crowdsourcing for multi-class labeling. *ACM SIGMETRICS Performance Evaluation Review* 41, 1 (2013), 81–92.

[9] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics (NRL)* 2, 1-2 (1955), 83–97.

[10] Shailesh Kumar, Joydeep Ghosh, and Melba M Crawford. 2002. Hierarchical fusion of multiple classifiers for hyperspectral data analysis. *Pattern Analysis & Applications* 5, 2 (2002), 210–220.

[11] Panagiotis Mavridis, David Gross-Amblard, and Zoltán Miklós. 2016. Using hierarchical skills for optimized task assignment in knowledge-intensive crowdsourcing. In *Proc. of WWW Conf.* 843–853.

[12] Naoki Otani, Yukino Baba, and Hisashi Kashima. 2015. Quality Control for Crowdsourced Hierarchical Classification. In *Proc. of IEEE ICDM*. 937–942.

[13] Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. 2008. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proc. ACM SIGKDD*. 614–622.

[14] Carlos N Silla and Alex A Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery* 22, 1-2 (2011), 31–72.

[15] Aditya Vempaty, Lav R Varshney, and Pramod K Varshney. 2014. Reliable crowdsourcing for multi-class labeling using coding theory. *IEEE Journal of Selected Topics in Signal Processing* 8, 4 (2014), 667–679.

[16] Xueping Weng, Guoliang Li, Huiqi Hu, and Jianhua Feng. 2017. Crowdsourced Selection on Multi-Attribute Data. In *Proc. ACM CIKM*. 307–316.

[17] Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier R. Movellan. 2009. Whose Vote Should Count More: Optimal Integration of Labels from Labelers of Unknown Expertise. In *Proc. of NIPS*. 2035–2043.

Modeling the Factors of User Success in Online Debate

Esin Durmus
Cornell University
ed459@cornell.edu

Claire Cardie
Cornell University
cardie@cs.cornell.edu

Abstract

Debate is a process that gives individuals the opportunity to express, and to be exposed to, diverging viewpoints on controversial issues; and the existence of online debating platforms makes it easier for individuals to participate in debates and obtain feedback on their debating skills. But understanding the factors that contribute to a user's success in debate is complicated: while success depends, in part, on the characteristics of the language they employ, it is also important to account for the degree to which their beliefs and personal traits are compatible with that of the audience. Friendships and previous interactions among users on the platform may further influence success.

In this work, we aim to better understand the mechanisms behind success in online debates. In particular, we study the relative effects of debaters' language, their prior beliefs and personal traits, and their social interactions with other users. We find, perhaps surprisingly, that characteristics of users' social interactions play the most important role in determining their success in debates although the best predictive performance is achieved by combining social interaction features with features that encode information on language use during the debate.

CCS Concepts

• **Information systems** → *Web applications*; *Social networks*; • **Social and professional topics** → *User characteristics.*

Keywords

online debate; social interactions; personal traits; language; persuasion

ACM Reference Format:
Esin Durmus and Claire Cardie. 2019. Modeling the Factors of User Success in Online Debate. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313676

1 Introduction

Previous work from Natural Language Processing (NLP) and Computational Social Science (CSS) that studies argumentative text and its persuasive effects has mainly focused on identifying linguistic features that are indicative of effective argumentation strategies [13, 16, 17, 33, 37, 41, 43]. For example [43] has shown that language characteristics such as conversational flow, are predictive of the success of a debater in persuading an audience member to change their stance *during a debate.* [10] has further shown that the prior beliefs of the audience with respect to the topic of discussion and the stance/beliefs of the debaters are at least as important as the linguistic characteristics of the debate in predicting the successful debater. At the same time, it has been shown that there is a strong relationship between users' social interactions and their influence on social media. For example, [35] and [7] and have shown that individuals with more activity and personal engagement are more influential on Twitter. We **hypothesize** that success in persuasion might also depend on an individual's social interactions and engagement with other users (on the debate platform) **over time**. For example, being more engaged with others over time may expose an individual to more diverse ideas and people, which in turn could foster argumentation skills that are more applicable to convincing a more diverse audience. Focusing on only individual debates and discussion threads, prior work has not investigated the relative effect of individuals' social interactions, personal traits and language use on their success in persuasion. To understand the relative effect of these factors on users' success in persuasion, we focus on online debates and study success over a user's lifetime by looking at interactions and engagement with the community over time, rather than focusing on success in individual debates.

Our study employs the DDO (debate.org) dataset [10]. Its extensive user information and multiple well-structured debates/interactions per user provides a unique opportunity to study users' success over time while accounting for the effect of individuals' social interactions, personal traits and language use. Users provide demographic information as well as their stance on controversial topics. They interact with one another in many ways: 1) debating 2) evaluating the performance of other debaters, 3) commenting on debates, 4) asking/answering opinion questions, 5) voting in polls, 6) creating polls 7) becoming friends. Ultimately, we find that the characteristics of individuals' social interactions (e.g. their friendships and voter network) are an important component in predicting their success in debates. The best predictive performance is achieved by combining social interaction features with language use features (e.g. vocabulary diversity and the extent to which their language use matches that of their opponent). **Contributions.** To the best of our knowledge, this is the first work to study computationally 1) the factors of an individual's overall success in online debating **over time** 2) by considering the combined effect of **social interactions**, personal traits and language use on success[1].

2 Dataset

Available User Information. The DDO (debate.org) dataset contains personal trait information for 45, 348 users including demographics such as gender and ethnicity as well as users' beliefs, such as their political ideology, religious ideology and stance on

[1]The dataset is publicly available at http://www.cs.cornell.edu/ esindurmus/.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313676

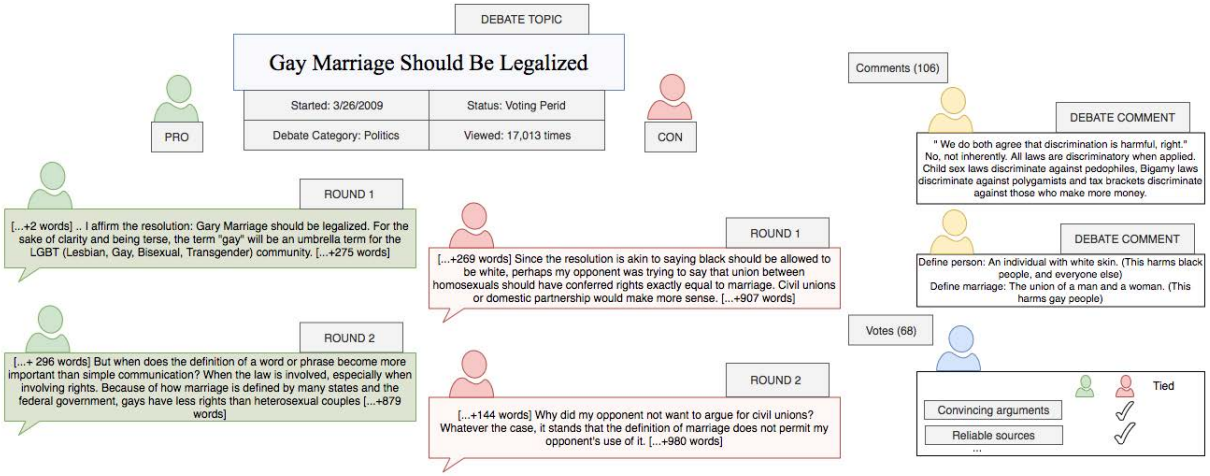


Figure 1: Example debate for the claim "*Gay Marriage Should Be Legalized*". In each round, opposing sides (PRO and CON) present their argument. Users on the platform give feedback and evaluate each debater's performance by providing comments and votes.

controversial issues. For this work, we augment DDO with social interaction data associated with each user: the debates that they participated in, voted on, and commented on; opinion questions asked, opinion arguments, poll votes and poll topics of the users[2]. Users can interact with others in more ways than simply participating in debates on DDO. For example, they can post poll and opinion questions/arguments. This allows users to exchange opinions, and to be exposed to new perspectives without engaging in a formal, structured argument. Opinion questions and arguments are more open-ended whereas polls require users to pick from a set of predefined options. An example opinion question and opinion argument are provided below:

Example Opinion Question. "Does God exist?"[3]

Example Opinion Argument. "He probably does not exist. I don't think that it's possible to say yes or no either way. We can only conclude that there is more logical evidence to say that a God probably does not exist, ..."

Debate structure. Each debate consists of multiple rounds, in which each of the two opposing debaters provides their argument. Debaters have a single opportunity to present their argument in any given round. The majority of debates have three or more rounds. There are 23 distinct debate categories. The most common debate categories are *Politics*, *Religion*, and *Society*. Figure 1 shows an example debate on GAY MARRIAGE as well as some of the comments and votes provided for this debate by other users. We see from the example that unlike monological argumentation, the debaters' arguments are shaped not only by their own opinions, but also by the way their opponents' express their opinions: debaters tend to refer to points made by their opponents throughout the debate. Voters evaluate debaters according to the quality and persuasiveness of their arguments and also provide an overall score for each debater. The winner of the debate is determined according to the total score from the voters.

3 METHODOLOGY

In this section we describe the methods used to investigate the underlying dynamics of success in online debate.

3.1 User Success

We compute the overall *success* in debate for user u as:

$$success_u = \frac{\#\text{ of debates } u \text{ won}}{\#\text{ of total debates } u \text{ participated in as a debater}}$$

We treat users with $success_u \geq 70\%$ as *successful*, $success_u \leq 30\%$ as *unsuccessful* and $30\% < success_u < 70\%$ as *mediocre*.

3.2 Prediction Task

To understand the relative effect of users' personal traits, social interactions and language on their *success*, we study the following prediction task: **given a pair of debaters where one of them is *successful* and other is *unsuccessful* over the second and third stage of their lifetime, predict the *successful* one**. Note that while determining our label for success, we consider only the debates in the second and third stage of the a user's lifetime, to be able to study the relative effect of *success* in their first life stage (*success prior*) vs. other factors in a controlled way. We experiment with two settings where we control for the effect of *debate experience* and *success prior* respectively.

SETTING 1. To **control the effect of debate experience** in *success*, we create the pairs by **matching the users according to the number of debates** that they participated in (i.e. users within a pair have the same number of debates[4].

[2]The dataset includes 77,655 debates, 592,390 comments, 197,231 debate votes, 262,222 poll votes, 20,300 poll topics, 10,045 opinion questions, and 61,183 opinion arguments.
[3]Full discussion on the topic can be found at https://www.debate.org/opinions/does-god-exist.
[4]There are 2154 such pairs in our dataset.

Aspect	Features
Personal traits	1) **match of the personal traits (e.g. gender, political ideology, religious ideology and ethnicity) with friends and voters**. 2) **opinion similarity with friends and voters**.
Social Interactions	1) **participation features** : # of comments, # of votes, # of friends, # of opinion questions and arguments, # of voted debates, # of poll votes and topics. 2) **friendship network features** : degree, degree centrality, page rank scores. 3) **voter network features**: in-degree, out-degree, in-degree centrality, out-degree centrality, page rank, hub and authority scores.
Language	1) **features of debaters' own language** : # of words, # of definite articles, # of indefinite articles, # of person pronouns, # of positive words, # of negative words, # of hedges, # of swear words, # of punctuation, # of links, average sentiment, type-token ratio, # of quotes, distribution of POS tags, distribution of named entities, BOW. 2) **features to encode the interplay** : exact content word match, exact stop word match, content word match with synonyms.

Table 1: Features

SETTING 2. Given that we're interested in understanding the factors that correlate with *success*, we **control for the *success prior*** in a very specific way – we only consider users that were *unsuccessful* in their initial life stage (**success prior**$\leq 30\%$[5]). This allows us to directly study the factors that are correlated with users that were initially *unsuccessful*, but later went on to become *successful* debaters.

3.2.1 Personal Traits It has been shown that characteristics of the audience's personality [22], and the degree to which the debaters' beliefs match with that of the audience [10], are important to consider in persuasion studies. We further investigate this effect in debaters' *success* over their lifetime. We also extend this study by considering additional personal traits, such as the degree to which debaters' demographics (e.g. gender and ethnicity) matches with those of their friends and the voters participating in the debates. We extract features to encode the similarity for a user's opinion, political ideology, religious ideology, gender, and ethnicity with that of her friends and voters. To compute opinion similarity, we used the information about users' opinions on 48 different controversial topics which they share on their user profiles and we measure the match of the opinions for these controversial topics[6].

3.2.2 Social Interactions The users interact with each other on the platform in following ways: 1) debating 2) evaluating the performance of other debaters, 3) commenting on debates, 4) asking/answering opinion questions, 5) voting in polls, 6) creating polls, 7) becoming friends.

We hypothesize that modeling these interactions is important to understand the differences between how *successful* and *unsuccessful* users interact on this platform, and whether or not these are important factors for success. The ability to interact with others in a myriad of different ways, provides users with ample opportunity to learn interesting new strategies and improve their skills over time, as they are exposed to a diverse set of perspectives.

[5]There are 957 such pairs in our dataset.

[6]We consider issues where users identified their side as either PRO or CON and measure the similarity of their opinion for these issues with their friends and voters.

Friendship network. We represent the friendship network as an **undirected** graph $G = (V, E)$ where V represents the set of users, and E represents the set of edges where $(x, y) \in E$ if $x \in V$ and $y \in V$ are friends.

Voter network. We represent the voter network as a weighted **directed** graph $G = (V, E)$ where V represents the set of users, and E represents the set of edges where $(x, y) \in E$ if $x \in V$ voted in a debate in which $y \in V$ participated as a debater. The weight of the graph represents how many times x voted in debates y was a debater. Note having (x, y) edge in the graph **does not** imply that x voted for y in a debate.

Using HITS algorithm [19], we compute hub and authority scores for each node (user) in the voter network graph. We find that *successful* users have, on average, a significantly higher hub score than *unsuccessful* users ($p < 0.001$).

3.2.3 Language To capture the linguistic style of the debaters' language and its relationship to their *success*, we use textual features that encode 1) users' own language and 2) the interplay between users' and their opponents' language.

Modeling users' own language. We extract features from the text of users' debates, opinion questions, opinion arguments, poll votes and poll topics. These features includes # of words, word category features (e.g. # of personal pronouns, # of positive and negative words), structural features (e.g. distribution of POS tags and named entities) and features to encode the characteristics of the entire language (e.g. type-token ratio)

Modeling the interplay between the debaters' and opponents'. We measure the interplay between debaters and their opponents by measuring how similar a debater's language is to the previous statement made by her opponent. To measure the similarity of a debater's language (D) to that of the opponent's (O) in a round, we look at # of content words that are in both D and O, # of stop words that are in both D and O and # of content words that are in D and have synonyms in O.

The *content word match with synonyms* feature aims to capture the cases where the opponent refers to similar concepts but doesn't necessarily use the exact same words as the debater.

		SETTING 1			SETTING 2		
	Feature	Precision(%)	Recall(%)	F1(%)	Precision(%)	Recall(%)	F1(%)
	(1) Majority	$26.47_{\pm 1.11}$	$51.44_{\pm 1.08}$	$34.95_{\pm 1.22}$	$26.67_{\pm 1.61}$	$51.62_{\pm 1.56}$	$35.16_{\pm 1.76}$
	(2) Debating experience	$52.70_{\pm 2.91}$	$52.04_{\pm 1.77}$	$41.76_{\pm 2.06}$	$46.00_{\pm 0.89}$	$50.16_{\pm 1.02}$	$38.98_{\pm 3.92}$
	(3) Success prior	$65.20_{\pm 0.77}$	$64.39_{\pm 0.65}$	$63.63_{\pm 0.50}$	$55.60_{\pm 0.93}$	$55.07_{\pm 0.39}$	$52.10_{\pm 0.47}$
Personal Traits	(4) Overall similarity with voters	$61.93_{\pm 1.60}$	$60.86_{\pm 1.70}$	$59.44_{\pm 1.67}$	$56.55_{\pm 2.43}$	$55.69_{\pm 1.31}$	$52.68_{\pm 1.47}$
	(5) Overall similarity with friends	$62.70_{\pm 0.86}$	$59.98_{\pm 1.05}$	$56.94_{\pm 1.14}$	$55.87_{\pm 3.43}$	$54.23_{\pm 2.35}$	$47.52_{\pm 3.18}$
Social Interactions	(6) Participation features	$67.78_{\pm 1.66}$	$66.02_{\pm 2.33}$	$64.82_{\pm 2.70}$	$59.39_{\pm 4.09}$	$57.68_{\pm 2.34}$	$55.08_{\pm 3.16}$
	(7) Friendship network features	$64.23_{\pm 1.40}$	$63.60_{\pm 1.40}$	$62.92_{\pm 1.35}$	$57.94_{\pm 1.87}$	$57.16_{\pm 1.50}$	$55.41_{\pm 1.67}$
	(8) voter network features	$72.39_{\pm 0.19}$	$70.75_{\pm 0.34}$	$70.20_{\pm 0.70}$	$70.54_{\pm 1.78}$	$69.91_{\pm 1.79}$	$69.65_{\pm 1.76}$
	(6) + (7) + (8)	$72.67_{\pm 0.73}$	$72.29_{\pm 0.93}$	$72.12_{\pm 1.03}$	$71.66_{\pm 0.71}$	$71.47_{\pm 0.51}$	$71.38_{\pm 0.51}$
Language	(9) # of words	$70.37_{\pm 1.41}$	$70.15_{\pm 1.55}$	$69.97_{\pm 1.59}$	$65.78_{\pm 0.85}$	$64.99_{\pm 1.03}$	$64.41_{\pm 1.16}$
	(10) Features of debaters' interplay	$62.11_{\pm 1.09}$	$62.07_{\pm 1.03}$	$61.92_{\pm 1.01}$	$57.47_{\pm 1.42}$	$57.16_{\pm 1.31}$	$56.41_{\pm 1.29}$
	(11) Features of debaters' own language	$72.65_{\pm 2.45}$	$72.66_{\pm 2.45}$	$72.64_{\pm 2.44}$	$64.48_{\pm 0.74}$	$64.37_{\pm 0.90}$	$64.24_{\pm 0.97}$
Combinations	(6) + (7) + (8) + (11)	$78.49_{\pm 1.29}$	$78.46_{\pm 1.32}$	$78.45_{\pm 1.32}$	$75.44_{\pm 0.90}$	$75.44_{\pm 0.90}$	$75.43_{\pm 0.89}$
	(6) + (7) + (8) + (10) + (11)	$\mathbf{81.63_{\pm 1.63}}$	$\mathbf{81.62_{\pm 1.65}}$	$\mathbf{81.61_{\pm 1.65}}$	$\mathbf{78.06_{\pm 0.88}}$	$\mathbf{78.05_{\pm 0.89}}$	$\mathbf{78.05_{\pm 0.88}}$

Table 2: Prediction Task Results for SETTING 1 and SETTING 2. For both SETTING 1 and SETTING 2, voter network features are the most predictive social interaction features. Combining interaction and language features achieves the best predictive performance.

The full list of features modeling the aspects of personal traits, social interactions and language features is shown in Table 1.

3.3 Prediction Results

We use weighted logistic regression and choose the amount and type of regularization ($\ell 1$ or $\ell 2$) by grid search over 5 cross-validation folds. We compute **weighted** precision, recall and F1 scores.

In SETTING 1, we create user pairs (u_1,u_2) where:

- u_1 and u_2 have an equal number of debates they participated in as debaters.
- One of u_1 or u_2 is *successful* and the other one is *unsuccessful* over the second and third stage of their lifetime [7].

In SETTING 2, in addition to the requirements of SETTING 1, we also require u_1 and u_2 to both have *success prior* ≤ 0.3.

Task. For both SETTING 1 and SETTING 2, we aim to predict whether u_1 or u_2 is *successful* over the second and third stage of her lifetime.

In SETTING 2, by only studying user pairs with low *success priors*, we aim to understand the factors that are important for a user to improve as a debater over time.

3.3.1 Results for SETTING 1 Table 2 shows the results for SETTING 1 and SETTING 2. We compare our model with three simple baselines – majority, debating experience, and success prior. For the majority baseline, for each example, we predict the most common label in the training data. For debating experience baseline, we use # of debates as the only feature to predict the *successful* debater. For success prior baseline, we pick the user with the higher *success prior* as successful.

In SETTING 1, since we do not control for the *success* in the first life stage, we see that the *success prior* information alone can achieve 63.63% F1 score. This implies that there is a correlation between users' *success* in their early life stage and later life stages. This factor may be related to users' prior debating skills. We observe that the features that encode debaters' overall similarity with voters and friends achieves significantly better F1 score than majority and debating experience baselines. However, these features do not have as high a predictive power as the *success prior*. We perform an ablation study for participation features, friendship network features and voter network features. We find that voter network features are significantly more predictive than the baselines, personal trait features and other social interaction features. We also perform an ablation study for the language features and find that # of words is a very predictive feature of *success*. When we combine the language features with the interaction features, we get the best predictive performance (81.61% F1 score) for this task which is significantly better than the baselines. This indicates that it is important to account for both social interaction and language factors to determine the *successful* debater since these two components encode different kinds of information about the users.

3.3.2 Results for SETTING 2 In this task, by controlling for *prior success*, we aim to understand the factors that are correlated with *success* by reducing the effect of prior debating skills of the users. As shown in Table 2, the F1 score for the *success prior* baseline is not as quite as high as in SETTING 1, since we control for this aspect by ensuring both users in the pair are *unsuccessful* in their initial life stage. However, this does not necessarily mean that the two paired users will have the exact same success prior, which explains why success prior still performs better than the other baselines. We do not observe any significant difference between the performance of the features encoding personal traits, participation, and the baseline. However, consistent with the SETTING 1, we see that features of the voter network are significantly better (69.65%) in predicting *success*. Although language features achieve significantly better F1 score than the baseline, they perform significantly worse than

[7] We consider success only over the second and third stage of users' lifetime in our prediction task, in order to study the effect of *success prior* vs. the other aspects. We use the success in the first life stage as *success prior*.

the voter network features. Similar to SETTING 1, combining these language features with the social interaction features improves the performance significantly (78.05% F1 score).

3.3.3 Feature Analysis To understand the important social interaction and language features, we 1) compute the correlation coefficients for the feature values and the labels, 2) analyze the coefficients of the logistic regression classifier and 3) apply recursive feature elimination method [15] to rank the features according to their importance. In this section, we present the features that are consistently important with respect to each of these methods.

Analysis of Social Interaction Features. We find that the **most important** social interaction features for SETTING 1 are authority score, hub score, in and out-degree centrality and the page rank of the voter network. Note that all these important features are **positively correlated** with *success*. Although participation and friendship network features (e.g. # of voted debates, degree of the user node in friendship network) are also positively correlated with *success*, the correlation values for these are not as high as the ones of the voter network features. We also find high correlation between some of the user activities. For example, users with more # of comments are more active in making friends, voting, providing poll votes and they have higher centrality value in the friendship network. Perhaps surprisingly, we do not observe any correlation between # of voted debates and hub/authority scores in voter network. However, we see a highly positive correlation between hub scores, authority scores, in-degree centrality, out-degree centrality and page rank values of voter network. This implies that *success* is not only about the quantity of voted debates but also about the characteristics of the debaters that are involved in these debates, because the hub score of a user is influenced by the authority scores of the debaters they vote for. Similarly, the authority score of a user is influenced by the hub scores of the voters that participate in her debates. Therefore, besides the frequency of interactions, type of interactions and characteristics of users involved in the interactions are important to take into account. Consistent with SETTING 1, in SETTING 2, the most important features (positively correlated with *success*) are authority score, hub score, in and out-degree centrality and the page rank of the voter network. We observe the same patterns of user activities and authority and hub scores as in SETTING 1.

Analysis of Language Features. We find that # of words is positively correlated with *success*. It may be the case that longer text may convey more information and explain the points more explicitly [29, 30]. Bag of words features are not as predictive as the # of words feature. For both SETTING 1 and SETTING 2, we observe that the value of average sentiment is negatively correlated with *success*. The reason for this may be that negative information is more attention grabbing than positive information [9, 18, 34] since people are more used to seeing arguments that are phrased in a more positive way [25]. We also find that *type-token ratio* (diversity of language) is negatively correlated with *success* for both settings. It may be the case that people who talk about a smaller set of topics gain expertise on these topics over time; therefore, they may be more *successful*. We observe that other textual features are positively correlated with *success* for both of these settings. However, the degree of correlation is not as high as it is for type-token ratio and sentiment.

4 UNDERSTANDING THE LOSS OF *SUCCESS*

In the previous section, we show that social interaction and language features are important to predict *successful* debaters. Our findings are consistent for the case when 1) we only control for users' debating experience and 2) we also control for users' *success prior*. Users' participation, the types of interactions they have on the platform, and the characteristics of the users they interact with are predictive of their *success*, regardless of a their prior expertise in debating (encoded by the *success prior*).

In SETTING 1, since we did not control for the *success prior*, we studied the factors that are important for a user to become *successful* in their second and third life stages, regardless of their *success* in the beginning. In SETTING 2, we studied the factors that are important for an *unsuccessful* users to improve their performance and become *successful* over time. As a natural follow-up, we would also like to understand what factors are correlated with users who are initially *successful*, but later become *unsuccessful* in their lifetime. To do that, in SETTING 3, in addition to the requirements of SETTING 1, we have an additional criterion for all user pairs (u_1,u_2):

- u_1 and u_2 both have *success prior* ≥ 0.7 [8].

4.1 Results

As shown in Table 3, features of personality traits, social interactions and language perform significantly better than the baselines. For this task, *success prior* baseline performs relatively worse than in the previous two settings. Upon closer examination, we observed that the variance of success priors for this task is an order of magnitude smaller than in SETTING 2, and therefore, as a possible explanation, the *success priors* may not be as predictive for this task.

In social interaction features, similarity with friends is the most predictive feature. However, participation features perform significantly better than the features of personal traits. For this task, contrary to SETTING 1 and SETTING 2, we see that participation features are the most predictive in the set of social interaction features. This implies users' participation is important for them to remain *successful*. Lower participation could be a contributing factor for these users to eventually become unsuccessful. Although friendship and voter network features are still significantly more predictive than the baselines, they are not as highly predictive as the participation features. For users with high *success priors*, continued participation may be the most important aspect of their social interaction. We observe that language features alone achieves a similar performance as the social interaction features. Consistent with the SETTING 1 and SETTING 2, combining social interaction and language features gives the best predictive performance (73.43% F1 score).

Analysis of Social Interaction Features. The **most important** social interaction features include # of voted debates, degree of the user node in the friendship network, in-degree, out-degree, in-degree centrality and out of the user node in the voter network. All these features are indicative of a higher participation on the platform and they are positively correlated with staying *successful*.

[8] We have 700 user pairs with these criteria.

	Feature	Precision(%)	Recall(%)	F1(%)
	(1) Majority	$26.97_{\pm 2.69}$	$51.86_{\pm 2.62}$	$35.46_{\pm 2.95}$
	(2) Debating experience	$53.77_{\pm 2.95}$	$52.43_{\pm 2.91}$	$43.02_{\pm 6.19}$
	(3) Success prior	$39.94_{\pm 7.63}$	$51.00_{\pm 2.23}$	$36.04_{\pm 2.35}$
Personal Traits	(4) Overall similarity with voters	$55.17_{\pm 1.58}$	$55.00_{\pm 2.36}$	$53.94_{\pm 2.99}$
	(5) Overall similarity with friends	$66.38_{\pm 4.11}$	$63.43_{\pm 2.77}$	$60.87_{\pm 3.33}$
Social Interactions	(6) Participation features	$68.88_{\pm 3.57}$	$68.00_{\pm 2.86}$	$67.88_{\pm 2.96}$
	(7) Friendship network features	$65.60_{\pm 4.83}$	$64.00_{\pm 3.81}$	$62.81_{\pm 3.73}$
	(8) voter network features	$64.36_{\pm 1.57}$	$62.72_{\pm 2.37}$	$61.44_{\pm 2.87}$
	(6) + (7) + (8)	$67.80_{\pm 1.86}$	$67.14_{\pm 1.43}$	$66.97_{\pm 1.42}$
Language	(9) # of words	$67.63_{\pm 3.90}$	$66.57_{\pm 2.70}$	$66.29_{\pm 2.39}$
	(10) Features of debaters' interplay	$58.76_{\pm 2.03}$	$57.43_{\pm 0.86}$	$56.60_{\pm 0.93}$
	(11) Features of debaters' own language	$68.47_{\pm 0.21}$	$68.14_{\pm 0.14}$	$68.10_{\pm 0.17}$
Combinations	(6) + (7) + (8) + (11)	$69.32_{\pm 2.48}$	$69.00_{\pm 2.42}$	$69.00_{\pm 2.41}$
	(6) + (7) + (8) + (10) + (11)	$\mathbf{73.60_{\pm 0.80}}$	$\mathbf{73.43_{\pm 0.70}}$	$\mathbf{73.43_{\pm 0.72}}$

Table 3: Prediction Task Results for loss of *success*. Participation features are the most important social interaction features. Combining the social interaction features with the language features gives the best prediction performance.

Although the other social interaction features, such as authority and hub scores of the voter network are also positively correlated with *success*, the value of correlation for these is not as high as the previously mentioned features. For users who are initially *unsuccessful*, participation alone may not be enough for them to become *successful* debaters – the type of interactions and the characteristics of people with whom they interact are crucially important for their *success*. On the other hand, users who are initially *successful* may already be experienced debaters, and staying active and participating may be sufficient for them to remain *successful*.

Analysis of Language Features. As in SETTING 1 and SETTING 2, # of words is positively correlated with staying *successful*. We find that the # of first person pronouns is the language feature with the highest positive correlation with staying *successful*. We observe that users who refer to their personal experiences and opinions use first person pronouns more often. It may be the case that debaters may try to appeal to logos by citing personal experience [8]. Consistent with SETTING 1 and SETTING 2, the value of average sentiment is negatively correlated with staying *successful*.

5 Related Work

5.1 Modeling Social Interactions

There has been a tremendous amount of research on understanding user interactions and behaviour on social media [2, 4, 5, 14, 20, 21, 23, 24, 27, 42]. For example, [42] analyze the interaction graphs of Facebook user traces and show that interaction activity on Facebook is significantly skewed towards a small portion of each user's social links. [21] investigates how people interact in multiple online social networks. Although there is a lot of work on understanding user behavior on social media sites such Facebook and Twitter, the work on understanding user behaviour on online debate platforms has been limited. [35] is the most similar to our work, in that the authors study the effect of interaction dynamics, such as participant entry order and degree of back-and-forth exchange in the discussion, on success in changing an opinion holder's stance in a thread. Note that unlike our study, this work does not consider the effect of social interaction features (such as friendship network or voter network) on users' *success*. Moreover, in our work, we study the overall *success* of users over their lifetime, rather than a single debate or discussion thread.

5.2 Language in Persuasion

Previous work in psychology and linguistics has studied the effect of language on people's perception of the arguments and persuasion [3, 12, 25, 39]. These studies explore the effects of *framing*, – whether presenting different formulations of the same problem (e.g. presenting something as a gain or loss) would result in different effects on people. Besides the effects of framing, researchers investigate the effects of structural and lexical characteristics of language on success. For example, [36] found that using concrete words yields more positive outcomes than using abstract words, in terms of convincing customers to have a positive attitude towards a product.

NLP (Natural Language Processing) studies in argumentation mining focus on 1) identifying the structure of the persuasive aspects of text and relationships between them [1, 26, 28, 31, 32, 38, 40] and 2) the characteristics of persuasive text [6, 13, 16, 17, 33, 37, 41, 43]. There is very limited work in NLP that considers the factors of the individuals who are involved in the argumentation [11, 22]. [22] analyzes the effect of people's personality traits on their perceptions of the monologic arguments. However, in their work, they do not have any information about the source of the arguments. Moreover, in their setting, there is no notion of social interactions. [10] considers the effect of prior beliefs of the audience members and debaters on success in an individual debate. However, unlike this study, it does not consider the effect of social interactions on a user's success in online debate over time.

References

[1] Khalid Al Khatib, Henning Wachsmuth, Johannes Kiesel, Matthias Hagen, and Benno Stein. 2016. A News Editorial Corpus for Mining Argumentation Strategies.

In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. The COLING 2016 Organizing Committee, 3433–3443. http://aclweb.org/anthology/C16-1324

[2] Lars Backstrom, Eytan Bakshy, Jon M Kleinberg, Thomas M Lento, and Itamar Rosenn. 2011. Center of attention: How facebook users allocate attention across friends. (2011).

[3] Sara M Banks, Peter Salovey, Susan Greener, Alexander J Rothman, Anne Moyer, John Beauvais, and Elissa Epel. 1995. The effects of message framing on mammography utilization. *Health psychology* 14, 2 (1995), 178.

[4] Fabrício Benevenuto, Tiago Rodrigues, Meeyoung Cha, and Virgílio Almeida. 2009. Characterizing User Behavior in Online Social Networks. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement (IMC '09)*. ACM, New York, NY, USA, 49–62. https://doi.org/10.1145/1644893.1644900

[5] Moira Burke, Cameron Marlow, and Thomas Lento. 2009. Feed me: motivating newcomer contribution in social network sites. In *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 945–954.

[6] Amparo Elizabeth Cano-Basave and Yulan He. 2016. A Study of the Impact of Persuasive Argumentation in Political Debates. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 1405–1413. https://doi.org/10.18653/v1/N16-1166

[7] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, P Krishna Gummadi, et al. 2010. Measuring user influence in twitter: The million follower fallacy. (2010).

[8] M. Cooper and W.L. Nothstine. 1992. *Power Persuasion: Moving an Ancient Art Into the Media Age*. Educational Video Group. https://books.google.com/books?id=mkEqAQAAMAAJ

[9] Peter Ditto and David F. Lopez. 1992. Motivated Skepticism: Use of Differential Decision Criteria for Preferred and Nonpreferred Conclusions. *Journal of Personality and Social Psychology* 63 (10 1992), 568–584. https://doi.org/10.1037/0022-3514.63.4.568

[10] Esin Durmus and Claire Cardie. 2018. Exploring the Role of Prior Beliefs for Argument Persuasion. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, 1035–1045. https://doi.org/10.18653/v1/N18-1094

[11] Esin Durmus and Claire Cardie. 2018. Understanding the Effect of Gender and Stance in Opinion Expression in Debates on "Abortion". In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*. Association for Computational Linguistics, New Orleans, Louisiana, USA, 69–75. https://doi.org/10.18653/v1/W18-1110

[12] Adrian Edwards, Glyn Elwyn, Judith Covey, Elaine Matthews, and Rolsin Pill. 2001. Presenting risk information a review of the effects of framing and other manipulations on patient outcomes. *Journal of health communication* 6, 1 (2001), 61–82.

[13] Vanessa Wei Feng and Graeme Hirst. 2011. Classifying arguments by scheme. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 987–996.

[14] Scott A Golder, Dennis M Wilkinson, and Bernardo A Huberman. 2007. Rhythms of social interaction: Messaging within a massive online network. In *Communities and technologies 2007*. Springer, 41–66.

[15] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. 2002. Gene selection for cancer classification using support vector machines. *Machine learning* 46, 1-3 (2002), 389–422.

[16] Ivan Habernal and Iryna Gurevych. 2016. What makes a convincing argument? Empirical analysis and detecting attributes of convincingness in Web argumentation.. In *EMNLP*. 1214–1223.

[17] Christopher Hidey, Elena Musi, Alyssa Hwang, Smaranda Muresan, and Kathy McKeown. 2017. Analyzing the Semantic Types of Claims and Premises in an Online Persuasive Forum. In *Proceedings of the 4th Workshop on Argument Mining*. Association for Computational Linguistics, Copenhagen, Denmark, 11–21. http://www.aclweb.org/anthology/W17-5102

[18] Pamela M. Homer and Sun-Gil Yoon. 1992. Message Framing and the Interrelationships among Ad-Based Feelings, Affect, and Cognition. *Journal of Advertising* 21, 1 (1992), 19–33. https://doi.org/10.1080/00913367.1992.10673357 arXiv:https://doi.org/10.1080/00913367.1992.10673357

[19] Jon M Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)* 46, 5 (1999), 604–632.

[20] Shamanth Kumar, Reza Zafarani, and Huan Liu. 2011. Understanding User Migration Patterns in Social Media.

[21] Bang Hui Lim, Dongyuan Lu, Tao Chen, and Min-Yen Kan. 2015. # mytweet via instagram: Exploring user behaviour across multiple social networks. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*. ACM, 113–120.

[22] Stephanie Lukin, Pranav Anand, Marilyn Walker, and Steve Whittaker. 2017. Argument Strength is in the Eye of the Beholder: Audience Effects in Persuasion. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, 742–753. http://aclweb.org/anthology/E17-1070

[23] Sofus A Macskassy and Matthew Michelson. 2011. Why do people retweet? anti-homophily wins the day!

[24] Marcelo Maia, Jussara Almeida, and Virgílio Almeida. 2008. Identifying User Behavior in Online Social Networks. In *Proceedings of the 1st Workshop on Social Network Systems (SocialNets '08)*. ACM, New York, NY, USA, 1–6. https://doi.org/10.1145/1435497.1435498

[25] Beth E Meyerowitz and Shelly Chaiken. 1987. The effect of message framing on breast self-examination attitudes, intentions, and behavior. *Journal of personality and social psychology* 52, 3 (1987), 500.

[26] Raquel Mochales and Marie-Francine Moens. 2011. Argumentation Mining. *Artif. Intell. Law* 19, 1 (March 2011), 1–22. https://doi.org/10.1007/s10506-010-9104-x

[27] Meenakshi Nagarajan, Hemant Purohit, and Amit P Sheth. 2010. A Qualitative Examination of Topical Tweet and Retweet Practices. (2010).

[28] Vlad Niculae, Joonsuk Park, and Claire Cardie. 2017. Argument Mining with Structured SVMs and RNNs. In *Proceedings of ACL*.

[29] Daniel J. O'Keefe. 1997. Standpoint Explicitness and Persuasive Effect: A Meta-Analytic Review of the Effects of Varying Conclusion Articulation in Persuasive Messages. *Argumentation and Advocacy* 34, 1 (1997), 1–12. https://doi.org/10.1080/00028533.1997.11978023 arXiv:https://doi.org/10.1080/00028533.1997.11978023

[30] Daniel J. O'Keefe. 1998. Justification Explicitness and Persuasive Effect: A Meta-Analytic Review of the Effects of Varying Support Articulation in Persuasive Messages. *Argumentation and Advocacy* 35, 2 (1998), 61–75. https://doi.org/10.1080/00028533.1998.11951621 arXiv:https://doi.org/10.1080/00028533.1998.11951621

[31] Andreas Peldszus. 2014. Towards segment-based recognition of argumentation structure in short texts. In *Proceedings of the First Workshop on Argumentation Mining*. Association for Computational Linguistics, 88–97. https://doi.org/10.3115/v1/W14-2112

[32] Andreas Peldszus and Manfred Stede. 2016. Rhetorical structure and argumentation structure in monologue text. In *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*. Association for Computational Linguistics, 103–112. https://doi.org/10.18653/v1/W16-2812

[33] Peter Potash and Anna Rumshisky. 2017. Towards Debate Automation: a Recurrent Model for Predicting Debate Winners. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2455–2465.

[34] Felicia Pratto and Oliver P. John. 1991. Automatic Vigilance: The Attention-Grabbing Power of Negative Social Information. *Journal of personality and social psychology* 61 (10 1991), 380–91. https://doi.org/10.1037//0022-3514.61.3.380

[35] Daniel M Romero, Wojciech Galuba, Sitaram Asur, and Bernardo A Huberman. 2011. Influence and passivity in social media. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 18–33.

[36] John R Rossiter and Larry Percy. 1987. *Advertising and promotion management*. McGraw-Hill Book Company.

[37] Christian Stab and Iryna Gurevych. 2014. Identifying Argumentative Discourse Structures in Persuasive Essays. In *EMNLP*.

[38] Christian Stab and Iryna Gurevych. 2017. Parsing Argumentation Structures in Persuasive Essays. *Computational Linguistics* 43, 3 (2017), 619–659. https://doi.org/10.1162/COLI_a_00295

[39] Alessandra Tasso and Strada Cappuccini. 2005. Frame Effects in Persuasive Messages Against Smoking. (01 2005).

[40] Henning Wachsmuth, Martin Potthast, Khalid Al Khatib, Yamen Ajjour, Jana Puschmann, Jiani Qu, Jonas Dorsch, Viorel Morari, Janek Bevendorff, and Benno Stein. 2017. Building an Argument Search Engine for the Web. In *Proceedings of the 4th Workshop on Argument Mining*. Association for Computational Linguistics, 49–59. https://doi.org/10.18653/v1/W17-5106

[41] Lu Wang, Nick Beauchamp, Sarah Shugars, and Kechen Qin. 2017. Winning on the Merits: The Joint Effects of Content and Style on Debate Outcomes. *Transactions of the Association for Computational Linguistics* 5 (2017), 219–232. http://aclweb.org/anthology/Q17-1016

[42] Christo Wilson, Bryce Boe, Alessandra Sala, Krishna PN Puttaswamy, and Ben Y Zhao. 2009. User interactions in social networks and their implications. In *Proceedings of the 4th ACM European conference on Computer systems*. Acm, 205–218.

[43] Justine Zhang, Ravi Kumar, Sujith Ravi, and Cristian Danescu-Niculescu-Mizil. 2016. Conversational Flow in Oxford-style Debates. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, 136–141. https://doi.org/10.18653/v1/N16-1017

HopRank: How Semantic Structure Influences Teleportation in PageRank (A Case Study on BioPortal)

Lisette Espín-Noboa
GESIS & Uni. Koblenz-Landau
lisette.espin@gesis.org

Florian Lemmerich
RWTH Aachen University
florian.lemmerich@cssh.rwth-aachen.de

Simon Walk
Detego GmbH
s.walk@detego.com

Markus Strohmaier
RWTH Aachen University & GESIS
markus.strohmaier@cssh.rwth-aachen.de

Mark Musen
BMIR-Stanford
musen@stanford.edu

ABSTRACT

This paper introduces HopRank, an algorithm for modeling human navigation on semantic networks. HopRank leverages the assumption that users know or can see the whole structure of the network. Therefore, besides following links, they also follow nodes at *certain distances* (i.e., k-hop neighborhoods), and not at random as suggested by PageRank, which assumes only links are known or visible. We observe such preference towards k-hop neighborhoods on BioPortal, one of the leading repositories of biomedical ontologies on the Web. In general, users navigate within the vicinity of a concept. But they also "jump" to distant concepts less frequently. We fit our model on 11 ontologies using the transition matrix of clickstreams, and show that semantic structure can influence teleportation in PageRank. This suggests that users—to some extent—utilize knowledge about the underlying structure of ontologies, and leverage it to reach certain pieces of information. Our results help the development and improvement of user interfaces for ontology exploration.

CCS CONCEPTS

• **Information systems** → **Content ranking**; Browsers; • **Mathematics of computing** → *Exploratory data analysis.*

KEYWORDS

Biased random walker; PageRank; k-hop neighborhood; BioPortal

ACM Reference Format:
Lisette Espín-Noboa, Florian Lemmerich, Simon Walk, Markus Strohmaier, and Mark Musen. 2019. HopRank: How Semantic Structure Influences Teleportation in PageRank (A Case Study on BioPortal). In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313487

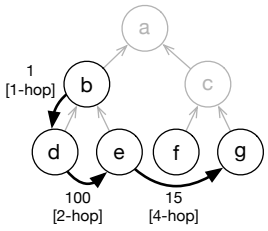
1 INTRODUCTION

Ontology Engineering and Ontology Learning are two branches of the Semantic Web whose aim is to accurately build and curate ontologies. The former studies new techniques to improve collaboration among humans while editing ontologies [26, 29], and the latter introduces new methodologies and algorithms to automatically create ontologies by crawling the Web [5, 25]. These efforts represent significant advances in the development of knowledge bases, which represent facts about the real world (e.g., people, diseases). However, there is little knowledge about how users consume such ontologies on the Web. To this end, Walk et al. studied how users browse BioPortal [28]. Their findings suggest that some ontologies influence the way users interact with the website. However, how users navigate through the ontology structure (i.e., from one concept to another) remains unclear.

Problem Statement: In this paper, we study the influence of semantic structure on *teleportation* (i.e., jumping to any node chosen at random) in PageRank. For example, consider the ontology shown in Figure 1(a), where nodes represent *classes* (a.k.a. concepts) and edges *isASubClassOf* relationships. On BioPortal, ontologies are shown vertically as hierarchical trees, and concepts can be explored using the *expand-on-demand* principle. This means that only top level concepts are shown first, and then users are able to expand and collapse as many concepts as they need at any level of the ontology. In other words, users can use and therefore are potentially aware of a *virtually fully connected network* in all stages of navigation. Previous studies [21, 31] have modeled user navigation using PageRank. However, these assume that navigation paths are constrained by links and random teleportation. In our scenario, where the whole structure of an ontology can be visualized at any time, we believe that teleportation is not fully random, but rather biased towards k-hop neighborhoods.

Approach: Motivated by previous studies on information foraging [8–10, 22], decentralized search [16, 18], and PageRank [4, 14, 15, 21, 31], we propose HopRank, a method for modeling transitions across k-hop neighborhoods on semantic networks. The key idea of this work relies on the HopPortation vector $\vec{\beta}$, which defines the probabilities of transitioning to each k-hop neighborhood. From the PageRank point of view, we can say that teleportation is not fully random, and the probability of following the structure of a page is not based only on one parameter (i.e., probability of following links), but on k parameters, representing all k-hop neighborhoods *reached from the current page*. Technically, we pass the HopPortation vector to a random walker to make biased decisions on which neighborhood to go next. Once this decision is made, the random walker uniformly chooses a concept within that neighborhood.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313487



(a) Transitions

	β_0	β_1	β_2	β_3	β_4
Transitions	0	1	100	0	15
Smoothing	1	2	101	1	16
Normalized	0.008	0.017	0.835	0.008	0.132

(b) HopPortation Vector

	a	**b**	**c**	**d**	**e**	**f**	**g**
a	0.001	0.011	0.011	0.244	0.244	0.244	0.244
b	0.008	0.001	0.963	0.008	0.008	0.006	0.006
c	0.008	0.963	0.001	0.006	0.006	0.008	0.008
d	0.419	0.018	0.009	0.001	0.419	0.067	0.067
e	0.419	0.018	0.009	0.419	0.001	0.067	0.067
f	0.419	0.009	0.018	0.067	0.067	0.001	0.419
g	0.419	0.009	0.018	0.067	0.067	0.419	0.001

(c) Transition Probabilities

Figure 1: HopRank on semantic networks. This example illustrates an instance of navigation on an ontology. (a) Shows the underlying network composed by seven concepts (a-g) and six *isASubClassOf* relationships (straight-thin grey arrows). Transitions (curved-thick black arrows) are labeled by the actual number of transitions between concepts, as well as the [k-hop] distance (i.e., shortest path) between them. (b) Illustrates how the HopPortation vector $\vec{\beta}$ is built using transition counts per k-hop. (c) Shows the transition probabilities inferred by HopRank, see Equation (1).

Contributions: The contributions of this paper are:

(1) We empirically show how users leverage the structure of the ontologies on BioPortal by quantifying the proportion of transitions per k-hop neighborhood.
(2) We propose HopRank, an algorithm for modeling human navigation on semantic networks.
(3) We demonstrate that HopRank outperforms traditional navigation and popularity-based models on BioPortal, especially when users browse ontologies directly without search.
(4) We make an implementation of this approach openly available on the Web [11].

2 RELATED WORK

BioPortal provides users with a *tree-like explorer* and a *local search* engine to navigate ontologies. In addition, concepts can be *expanded on demand* to see their children nodes. Although these functionalities are exploited differently across ontologies [28], it is unclear how users navigate through the ontology structure. Thus, this section covers previous work on search and navigation on networks.

Search. *Information Foraging* [22] assumes that people, when possible, modify their strategies or the structure of the environment to maximize their gain of valuable information. These patterns are also found in the way humans recall information from memory [17]. Similarly, *berrypicking* [6], a model of online searching, states that queries are not static, but rather evolve, and users commonly gather information in pieces instead of in one large set.

Navigation. *PageRank* [21] is the most popular method to measure the importance of web pages based on their incoming and outgoing links. It relies on an imaginary surfer who is randomly *clicking on links*, and eventually *jumps* to any node in the network. The probability of following links is given by a *damping factor*. Multiple variations have been proposed for improving information retrieval systems, e.g., a biased PageRank [15] to capture the importance of a page more accurately by taken topics into account or a weighted PageRank [31] to assign larger rank values to more popular pages (i.e., preferential attachment) instead of distributing the rank value of a page uniformly to all outgoing pages. Geigl et al. suggest that the behavior of a *random surfer* is almost similar to real users, as long as they do not use search engines [13]. They also find that classical navigation structures, such as navigation hierarchies or breadcrumbs, only exercise limited influence on navigation. Experiments in [24] reveal that memory-less *Markov chains* represent a quite practical model for human navigation on a page level. However, this assumption is violated when the analysis is expanded to a topical level. Helic et al. identify certain configurations of *decentralized search* that are capable of modeling human navigation in information networks [16]. Their findings suggest that navigation on such networks is a two phase process combined with the *exploitation of the known* (i.e., goal-seeking) and the *exploration of the unknown* (i.e., orientation).

User Interfaces. Human navigation has also been studied for enhancing interfaces. For instance, [12] explores *fisheye* views to display large information structures such as programs and databases. The intuition behind this paradigm is that users often explore their neighborhood, and distant major landmarks in more detail. Similarly, Van Ham and Perer studied the *search, show context, expand on demand* browsing model in [27], and proposed techniques to design better graph visualization tools.

We propose HopRank—a biased random walker—to model navigation on semantic networks. HopRank builds upon insights from information foraging [17, 22], decentralized search [16, 18] and PageRank [21]. More precisely, we replace the *damping factor* by a HopPortation vector to encode the probabilities of visiting each k-hop neighborhood. The intuition here is that users browse semantically close terms more often than semantically distant ones.

3 BIOPORTAL

There exist a large number of ontologies in the biomedical domain. They are highly specialized and therefore expensive to develop. To enable ontology adoption and reuse, effective support for browsing and exploring existing ontologies is required. Towards that goal, the National Center for Biomedical Ontology (NCBO) [3, 19] features BioPortal [1, 20, 30]—one of the leading repositories of biomedical ontologies on the Web—containing currently more than 700 ontologies with more than 9 million ontology classes. On BioPortal, practitioners and experts can access ontologies via Web services and Web browsers. The latter allows users to navigate ontologies

by searching specific classes, or by directly browsing their concept hierarchies within a tree-like explorer [28].

Ontologies. We propose to model human navigation on semantic networks using the structure of the underlying ontology. On BioPortal, ontologies are defined as directed networks, where nodes represent *concepts* and edges *isASubClassOf* relationships. Since such edges are usually non-cyclic and have a common root, these ontologies often form trees. Table 1 shows 11 of the most visited ontologies in 2015[1]. For instance, LOINC the largest ontology with $175K$ nodes, $153K$ edges, and $74K$ connected components.

Transitions. We analyzed all HTTP requests made in 2015 and extracted $336K$ valid sessions (i.e., after filtering out sessions with less than 2 requests, and requests to ontologies or concepts which do not exist). Each session contains transitions (i.e., a sequence of visited concept pages) triggered by a single user (i.e., IP address) without breaks (i.e., pauses of at least 60 minutes). For simplicity, we only consider transitions within the largest connected component (LCC) of each ontology, and discard ontologies with less than 1000 transitions[2]. Overall, we found 11 ontologies and $133K$ transitions between their concepts[3], see Table 1 for some key properties.

Navigation Types. Based on the HTTP request headers, we inferred 7 navigation types: Details (DE), Direct Click (DC), Direct URL (DU), Expand (EX), External Link (EL), External Search (ES), and Local Search (LS). **DE**: are all clicks made within the *Details* tab of a selected concept. **DC**: are all clicks made on concepts within the tree-like explorer. **DU**: refers to all concept requests without HTTP referrer (e.g., direct URL in the browser). **EX**: considers all clicks on the (+) symbol of a concept, which triggers the expansion of the concept to show all its children nodes. Notice that this request is called only once, even if the symbol is clicked multiple times. The opposite behavior (collapse) is not considered[4]. **EL**: captures all requests coming from external websites that are not search engines. **ES**: are all requests coming from the top 10 most popular external search engines such as Google and Yahoo. **LS**: are all requests made via the local search functionality of each ontology. Notice that this search is a 3-step process. First users type a keyword, then the system shows auto-suggestions and finally users click on one of the concepts shown in the auto-suggestion list. We only consider the final step a local search transition. **ALL**: includes all the above-mentioned types. Figure 2 shows the distribution of transitions across navigation types for each ontology. In general, most traffic comes from expanding a concept (EX, 44%), followed by local search (LS, 17%), direct URL (DU, 16%) and details (DE, 14%). Surprisingly, direct clicks on concepts (DC) only represent 7% of all transitions. This suggests that users spend substantial time expanding concepts before they find a concept of interest.

[1]As ontologies can be edited over time, we work with their latest snapshots from 2015.
[2]Transitions within the LCCs of these ontologies represent 80% of all transitions.
[3]We left out the popular SNOMEDCT ontology due to computational limitations.

Table 1: Datasets. This table illustrates network properties of 11 of the most popular ontologies on BioPortal in 2015. Ontologies represent networks whose nodes refer to *concepts* and edges *isASubClassOf* relationships. Original number of nodes, edges, and connected components of ontologies are shown under N, E and cc, respectively. Properties of the largest connected component (LCC) of each ontology are shown under N', E', d' and T', where d' refers to the diameter and T' to the number of transitions.

#	Ontology	N	E	cc	N'	E'	d'	T'
1	CPT	13219	13235	3	13092	13110	15	44651
2	MEDDRA	66506	31863	43493	22889	31738	8	42746
3	NDFRT	35019	34504	522	32074	32080	24	22452
4	LOINC	174513	152683	73518	100871	152558	13	6349
5	ICD9CM	22534	22531	3	22407	22406	12	4434
6	WHO-ART	1852	2997	3	1725	2872	4	2811
7	MESH	165166	24182	145652	16947	21596	31	2623
8	ICD10	12446	11256	1190	11132	11131	10	2288
9	CHMO	2966	3071	3	2964	3071	22	1423
10	HL7	10319	10600	1049	9146	10475	19	1374
11	OMIM	81821	39359	44110	37587	39234	6	1291

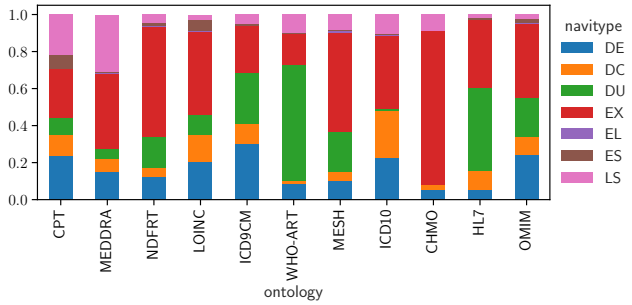


Figure 2: Navigation Types. Each bar shows the fraction of transitions within the LCC of each ontology. Stacked bars differentiate types of navigation: details (DE, blue), direct click (DC, orange), direct URL (DU, green), expand (EX, red), external link (EL, purple), external search (ES, brown) and local search (LS, pink). Most ontologies are mainly navigated by *expanding* nodes within the tree-like explorer.

4 HOPRANK: A BIASED RANDOM WALKER

HopRank models human navigation on semantic networks. Imagine a random walker whose decisions on where to go next are biased towards specific k-hop neighborhoods. This bias is what we call *HopPortation*, which encodes the probabilities of transitioning to each k-hop neighborhood. In our model, navigation on networks can be explained as a 2-step process. First, a k-hop neighborhood of the current node i is drawn from a categorical distribution. Second, a node j is randomly chosen within that k-hop neighborhood. Note that this process holds only if the walker is fully or partially aware of the structure of the network (i.e., knows or can see it). Without this prerequisite, and if links are not preferred, then random jumps to random pages will be more plausible. In comparison to the classic random walker with teleportation (e.g., PageRank [21]), where its movements are constrained by the damping factor α (i.e., probability of following links), HopRank is constrained by a vector $\vec{\beta}$ containing

[4]Collapse is a client-side functionality, and thus, it is not recorded in the log files.

k different factors, which define the probabilities of going to each k-hop neighborhood from the current location.

Visited k-hop Neighborhoods on BioPortal. We aggregate ALL transitions by the shortest distance between two sequentially visited nodes. This distance is referred to as k-hop neighborhood. In Figure 3(a) we see that target nodes at large distances are less likely to be visited next. This is expected, since—to some extent—larger distances enclose more branches, therefore more target candidates. Note that ontologies are sorted by diameter in descendant order from MESH to WHO-ART. Interestingly, users tend to hop as far as the ontology's diameter, for $d' \leq 12$. For instance, OMIM's diameter is 6 (see Table 1), and 6 is the maximum hop done by users. Otherwise, users (roughly) hop up to two-thirds of the ontology's diameter, for $d' > 12$. For example, MESH's diameter is 31, and the largest hop reached is 19.

Transitions per k-hop Neighborhood on BioPortal. Figure 3(b) shows the average percentage of transitions across k-hop neighborhoods per navigation type. We see that users on average (ALL, grey) prefer to navigate through 2-hop (41%) and 1-hop (23%) neighbors. In particular, when navigation is triggered by direct clicks (DC, orange) and expand (EX, red). Notice their fast decay when $khop > 8$. Other types of navigation such as external link (EL, purple), and direct URL (DU, green)—which do not leverage the tree-like explorer—tend to reach concepts at larger distances more frequently. Notice their peaks at $khop = \{5, 11\}$, respectively. Interestingly, when users opt for external search (ES, brown), they often click on 2-hop concepts, but also on 12-hop and 15-hop neighbors. Intuitively, the details tab (DE, blue) helps users to click on nearby concepts at $khop \leq 2$, more often than local search (LS, pink), which is more likely to reach concepts at $khop \geq 2$.

5 MODELS OF HUMAN TRANSITIONS

In this section, we formally introduce our HopRank model, and recap popular navigation models for comparison. We denote the transition probabilities, and # of parameters according to HopRank and 7 other models that we will use later on for model selection.

We formally represent an ontology[5] as a graph $G = (V, E)$, with $V = (v_1, \ldots v_n)$ being a set of N nodes, and $E = \{(v_i, v_j)\} \in V \times V$ a set of undirected edges[6]. The ontology structure is captured by the adjacency matrix $A_{N\times N} = a_{ij}$, where a_{ij} is 1 if the link exists, 0 otherwise. Transitions are represented by the transition matrix $T_{N\times N} = t_{ij}$, where t_{ij} represents the number of transitions between source node i and target node j.

HopRank (HR). Given the HopPortation vector $\vec{\beta}$, the probability of reaching a k-hop neighborhood is denoted by factor $\beta_k \in \vec{\beta}$. The stochastic k-hop matrix M_k encodes all nodes j with a shortest distance k from i. HopRank uniformly distributes β_k across all nodes j at distance k. The limits of k-hop neighborhoods go from 1 (direct edges), to d', the diameter of the ontology G. Noise $\beta_0 = 1 - \sum_{k=1}^{d'} \beta_k$ is added to allow for random jumps and self-loops. Figure 1(b) illustrates how the HopPortation vector is computed from the transition counts. *Number of model parameters:* $d' + 1$.

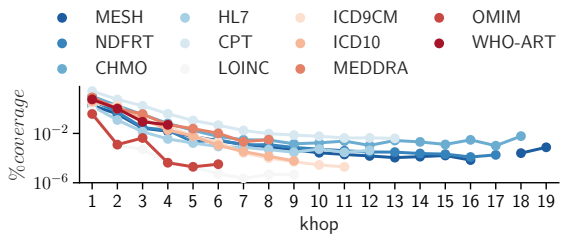
$$P_{HR} = \beta_1 \boldsymbol{M}_1 + \beta_2 \boldsymbol{M}_2 + \cdots + \beta_k \boldsymbol{M}_k + \frac{\beta_0}{N} \tag{1}$$

Preferential Attachment (PA). Given the degree matrix $D_{N\times N} = d_{ij} = d_j$, where d_j represents the degree of the target node j. The probability of moving from i to j is proportional to the degree of j. *Number of model parameters:* 0.

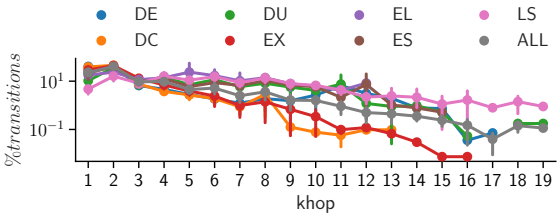
$$P_{PA} = \boldsymbol{D} \tag{2}$$

Gravitational (Gr). Given the matrix $S_{N\times N} = (sp(i,j) + \epsilon)^2$, where $sp(i,j)$ denotes the shortest path between nodes i and j. The probability of navigating from i to j is proportional to the degree of node j and inversely proportional to the square distance between i and j. We add a smoothing factor ϵ to avoid overflows when dyads are disconnected. In such cases, we set ϵ to the diameter d' of G plus 1, to consider these jumps with a very low probability. Similarly, we set the diagonal (i.e., self-loops) to $\epsilon = d' + 2$. *Number of model parameters:* 0.

$$P_{Gr} = \frac{\boldsymbol{D}}{S} \tag{3}$$



(a) % of dyads traversed per ontology



(b) Mean % of transitions per navigation type

Figure 3: Popularity of k-hops. (a) Shows the percentage of dyads that are traversed per k-hop neighborhood. Lines represent ontologies and are sorted by their LCC diameter: In descendent order from MESH (dark blue) to WHO-ART (dark red). (b) Shows the distribution of transitions across k-hop neighborhoods per navigation type. Percentages are averages across ontologies, and error bars the respective standard deviation. While several k-hop distances are being traversed non-uniformly, most transitions happen across nearby nodes, especially when browsing (DE, DC, EX, ES) 2-hop neighbors. In contrast, non-browsing types (EL, LS, DU) tend to reach more distant nodes more frequently.

[5]We focus on its largest connected component (LCC)
[6]Directionality of edges is omitted to calculate shortest paths between all pair of nodes.

Random Walker (RW). Given the damping factor α (i.e., probability of following links), the probability of visiting a node j is proportional to α divided by the degree of the source node i, plus a random choice equally distributed among all nodes. Depending on the α value, a random walker can model four different behaviors: **(i)** $\alpha = 0.0$: random jumps only, **(ii)** $\alpha \approx 1.0$: navigation over links only, **(iii)** $\alpha = 0.85$: PageRank using the commonly used damping factor for navigating the Web [7], and **(iv)** the empirical PageRank which learns the parameter α from the transitions data. *Number of model parameters:* 1 *if empirical,* 0 *otherwise.*

$$P_{RW} = \alpha \boldsymbol{A} + \frac{(1-\alpha)}{N} \tag{4}$$

Markov Chain (MC). We assume that moving to the next node follows a Markov process. Therefore, the probability of moving to a node j only depends on the current node i. These probabilities represent the maximum likelihood, learned from the transition matrix T. Thus, the probability of visiting node j from node i is proportional to the number of transitions t_{ij}. *Number of model parameters:* $N \times (N-2)$.

$$P_{MC} = \boldsymbol{T} \tag{5}$$

Note that $\boldsymbol{M}$, $\boldsymbol{A}$, and all P_* from Equations (1) to (5) are right stochastic matrices (i.e., each row must sum to 1).

6 EXPERIMENTS

In this section, we compare the performance of HopRank to the baselines on synthetic and real-world networks.

6.1 Model Selection

For comparing the models, we employ the Bayesian Information Criterion (BIC) [23] to select the best, i.e., lowest BIC score. BIC evaluates *log-likelihoods LL* (i.e., how likely our transitions are for a given model) and takes into account the *number of model parameters* and *observations* (i.e., # of transitions) to avoid over-fitting.

$$BIC = -2 \cdot LL + nparams \cdot log(nobservations), \tag{6}$$

$$LL = \sum_{i=1}^{N} \sum_{j=1}^{N} t_{ij} \cdot log(p_{ij}), \tag{7}$$

where t_{ij} represents the actual number of transitions from node i to node j, and p_{ij} the probability of transitioning from node i to node j for a given model.

6.2 Synthetic Network

Setup. The underlying network (structure) is a binary tree composed by $N = 7$ nodes and $|E| = 6$ edges as shown in Figure 1(a). Transitions (curved-thick edges) are biased towards 2-hop and 4-hop neighborhoods. These biases are reflected in the HopPortation vector shown in Figure 1(b).

Results. Probabilities inferred using Equation (1) are depicted in Figure 1(c). Figure 4 (left) shows the number of parameters inferred by each model. While the Markov chain model (MC) requires 35 parameters, HopRank only needs 5. The empirical PageRank (RW E.) learned a damping factor of $\alpha = 0.01$. This means that users are 1% likely to follow links. In Figure 4 (right) we see the comparison of models using BIC scores. In this synthetic network, transitions

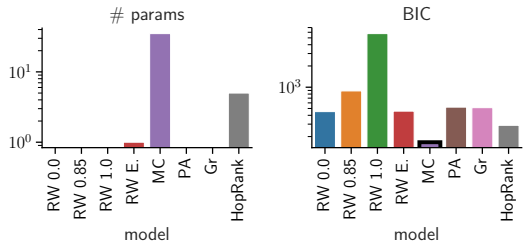


Figure 4: Results on Synthetic Network from Figure 1. X-axis maps the models at interest. (Left) Number of parameters inferred by each model. (Right) BIC scores: the lower the better at explaining the data. In this example, navigation is best described by Markov chain followed by HopRank.

are best described by the Markov chain model because model parameters (i.e., maximum likelihood) are proportional to the actual transition counts per dyad, and the data structure is very small[7]. In spite of that, HopRank is the second best model and describes navigation better than random (RW 0.0).

6.3 Medical Dictionary for Regulatory Activities Terminology (MEDDRA)

Setup. MEDDRA[2] is one of the the largest ontologies in our dataset (see Table 1). After pre-processing, its largest connected component (LCC) consists of $23K$ nodes and $43K$ transitions.

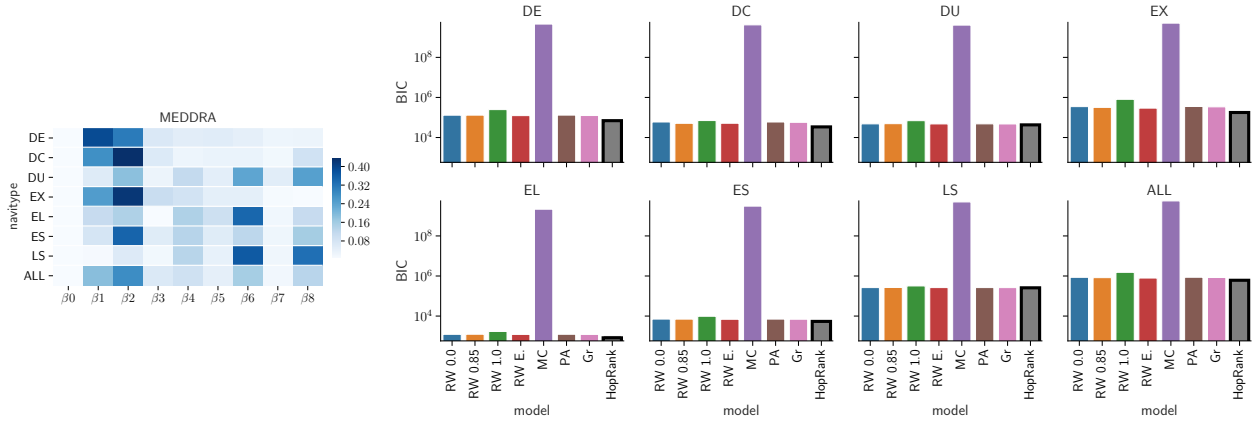
Results. Figure 5(a) shows the HopPortation vectors learned for each type of navigation in MEDDRA. We see that users mainly navigate through 1, 2, 6, and 8-hop neighbors. For instance, transitions through direct clicks—on a concept (DC), its details (DE) or expand (EX)—mainly follow 1-hop and 2-hop neighbors. However, when transitions are triggered by direct URLs (DU), local search (LS) or external links (EL), users tend to reach distant target nodes (i.e., 6-hop and 8-hop neighbors). Figure 5(b) shows the ranking of models according to BIC scores (lower is better). We see that in MEDDRA all types of navigation are best explained by HopRank.

6.4 Top11 Ontologies in BioPortal

Setup. We fit HopRank and the baseline models to all transitions by ontology and navigation type. These represent $133K$ transitions coming from the 11 ontologies described in Table 1.

Results. In Figure 6 we highlight the model that explains the number of transitions per ontology and navigation type best (i.e., the model with lowest BIC score). Ontologies are sorted by their number of transitions from CPT (largest) to OMIM (smallest). HopRank outperforms the other models 89% of the time, especially when users browse directly—regardless of the ontology—the tree-like explorer via clicks (DC), details (DE) and expand (EX). When there are not enough observations (i.e., the number of transitions is small), the other models tend to outperform HopRank due to the fact that the other models require fewer parameters and/or it is less likely to find transitions across different k-hop neighborhoods. This is the case for 6 ontologies in certain navigation types. For instance, we found

[7]Therefore, number of parameters does not play a very important role in BIC.



(a) HopPortation vectors

(b) Model selection

Figure 5: Results on MEDDRA. (a) This heatmap shows the HopPortation vectors learned from the transitions in MEDDRA. Cells represent the probabilities of visiting a certain k-hop neighborhood (column) by a given navigation type (row). In general, 2-hop and 1-hop neighborhoods are more likely to be visited next, regardless of navigation type (ALL). However, distant hops are preferred through *direct URLs* (DU), *external links* (EL), and *local search* (LS). (b) This figure shows the comparison of models across navigation types using BIC scores. We see that HopRank outperforms all baseline models.

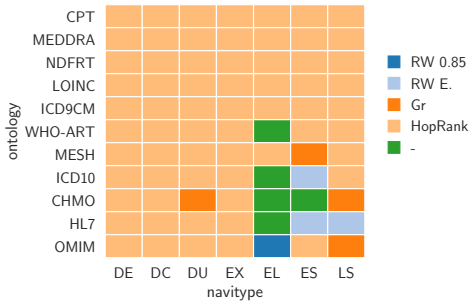


Figure 6: Model Selection on BioPortal. This heatmap highlights the model—with lowest BIC score—that best describes the # of transitions per ontology and navigation type. HopRank outperforms the other models 89% of the time, especially when browsing concepts via details (DE), direct click (DC) and expand (EX). When transitions are scarce (i.e., the other 11%), BIC penalizes HopRank since it has more parameters than the other models (except Markov chain).

5 external search (ES) transitions in MESH which are best described by the Gravitational model (Gr). Even though HopRank was a better candidate (i.e., higher log-likelihood), BIC penalized it for having more parameters ($nparams_{HopRank} = 32 > nparams_{Gr} = 0$). Notice that we model navigation in ontologies with at least 2 transitions. Ontologies that do not fulfil this condition per navigation type are marked as green cells "-".

7 DISCUSSION AND FUTURE WORK

In this section, we discuss decisions made for data processing, and future directions that can be pursued to improve our results.

Largest Connected Component (LCC). Surprisingly, ontologies on BioPortal may have multiple connected components. In those cases, only the branch connected to the root *owl:Thing* is shown at first in the tree-like explorer. Disconnected (and hidden) nodes or branches need to be accessed from external pages or local search. For simplicity, we opted to work with the LCC of each ontology with the cost of removing 20% of all transitions. Future work should consider the whole network to study the tradeoffs between number of transitions and random teleportation.

HopRank Extensions. More extensions based on network properties or similarity measures between nodes could improve our results. For instance, considering ontologies as directed graphs, and assuming that navigation is not only constrained by distance but also directionality: top-down or bottom-up.

Other Types of Networks. Even though this paper targets semantic networks, we believe that HopRank can be utilized to model human navigation in other networks, such as the Web or cities. The only assumption required is that users must have background knowledge of the underlying network they are surfing/traveling in.

8 CONCLUSIONS

In this paper, we introduced the concept of *HopPortation* which states that users—navigating a known or visible network—are biased towards certain k-hop neighborhoods. This is a variation of PageRank, where we assume that teleportation is not fully random but rather distributed non-uniformly across different neighborhoods. We proposed *HopRank*—a biased random walker—to model navigation on semantic networks. Our findings on BioPortal suggest that semantic structure (i.e., shortest path) influences navigation on networks. In particular, users tend to be biased towards certain k-hop neighborhoods depending on the type of navigation. For instance, when manually browsing the tree-like explorer, users tend to hop to nearby concepts, whereas far-away concepts are more

likely to be reached by non-browsing types such as external links. These results advance our understanding of how ontologies are actually navigated and consumed, and help to develop and improve user interfaces for ontology exploration.

Acknowledgements. We would like to thank Tania Tudorache, John Graybeal, Matthew Horridge, Clement Jonquet, Maulik Kamdar, Alex Skrenchuk, Marcos Oliveira, Fabian Flöck, Reinhard Munz, Dimitar Dimitrov, Indira Sen, Mattia Samory and the three anonymous reviewers for their time and suggestions to improve the quality of the paper. This work was funded by DFG German Science Fund research projects "KonSKOE" and "PoSTs II".

REFERENCES

[1] 2011. BioPortal. https://bioportal.bioontology.org/ Accessed: 2019-02-21.
[2] 2011. MEDDRA. https://bioportal.bioontology.org/ontologies/MEDDRA Accessed: 2019-02-21.
[3] 2011. National Center for Biomedical Ontology (NCBO). https://www.bioontology.org/ Accessed: 2019-02-21.
[4] Joshua T Abbott, Joseph L Austerweil, and Thomas L Griffiths. 2015. Random walks on semantic networks can resemble optimal foraging. (2015).
[5] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. *The semantic web* (2007), 722–735.
[6] Marcia J Bates. 1989. The design of browsing and berrypicking techniques for the online search interface. *Online review* 13, 5 (1989), 407–424.
[7] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems* 30, 1-7 (1998), 107–117.
[8] Stuart K Card, Peter Pirolli, Mija Van Der Wege, Julie B Morrison, Robert W Reeder, Pamela K Schraedley, and Jenea Boshart. 2001. Information scent as a driver of Web behavior graphs: results of a protocol analysis method for Web usability. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 498–505.
[9] Ed H Chi, Peter Pirolli, Kim Chen, and James Pitkow. 2001. Using information scent to model user information needs and actions and the Web. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 490–497.
[10] Ed H Chi, Peter Pirolli, and James Pitkow. 2000. The scent of a site: A system for analyzing and predicting information scent, usage, and usability of a web site. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, 161–168.
[11] Lisette Espín-Noboa. 2018. HopRank. https://github.com/lisette-espin/HopRank Accessed: 2019-01-23.
[12] George W Furnas. 1986. *Generalized fisheye views*. Vol. 17. ACM.
[13] Florian Geigl, Daniel Lamprecht, Rainer Hofmann-Wellenhof, Simon Walk, Markus Strohmaier, and Denis Helic. 2015. Random surfers on a web encyclopedia. In *Proceedings of the 15th International Conference on Knowledge Technologies and Data-driven Business*. ACM, 5.
[14] Michael Gorman. 2004. Google and God's mind. *Los Angeles Times* 17 (2004).
[15] Taher H Haveliwala. 2003. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE transactions on knowledge and data engineering* 15, 4 (2003), 784–796.
[16] Denis Helic, Markus Strohmaier, Michael Granitzer, and Reinhold Scherer. 2013. Models of human navigation in information networks based on decentralized search. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*. ACM, 89–98.
[17] Thomas T Hills, Michael N Jones, and Peter M Todd. 2012. Optimal foraging in semantic memory. *Psychological review* 119, 2 (2012), 431.
[18] Jon M Kleinberg. 2002. Small-world phenomena and the dynamics of information. In *Advances in neural information processing systems*. 431–438.
[19] Mark A Musen, Natalya F Noy, Nigam H Shah, Patricia L Whetzel, Christopher G Chute, Margaret-Anne Story, Barry Smith, and NCBO team. 2011. The national center for biomedical ontology. *Journal of the American Medical Informatics Association* 19, 2 (2011), 190–195.
[20] Natalya F Noy, Nigam H Shah, Patricia L Whetzel, Benjamin Dai, Michael Dorf, Nicholas Griffith, Clement Jonquet, Daniel L Rubin, Margaret-Anne Storey, Christopher G Chute, et al. 2009. BioPortal: ontologies and integrated data resources at the click of a mouse. *Nucleic acids research* (2009), gkp440.
[21] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.
[22] Peter Pirolli and Stuart Card. 1999. Information foraging. *Psychological review* 106, 4 (1999), 643.
[23] Gideon Schwarz et al. 1978. Estimating the dimension of a model. *The annals of statistics* 6, 2 (1978), 461–464.
[24] Philipp Singer, Denis Helic, Behnam Taraghi, and Markus Strohmaier. 2014. Detecting memory and structure in human navigation patterns using markov chain models of varying order. *PloS one* 9, 7 (2014), e102070.
[25] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 697–706.
[26] Tania Tudorache, Jennifer Vendetti, and Natalya Fridman Noy. 2008. Web-Protege: A Lightweight OWL Ontology Editor for the Web.. In *OWLED*, Vol. 432.
[27] Frank Van Ham and Adam Perer. 2009. "Search, show context, expand on demand": supporting large graph exploration with degree-of-interest. *IEEE Transactions on Visualization and Computer Graphics* 15, 6 (2009).
[28] Simon Walk, Lisette Espín-Noboa, Denis Helic, Markus Strohmaier, and Mark A Musen. 2017. How Users Explore Ontologies on the Web: A Study of NCBO's BioPortal Usage Logs. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 775–784.
[29] Simon Walk, Philipp Singer, Lisette Espín-Noboa, Tania Tudorache, Mark A Musen, and Markus Strohmaier. 2015. Understanding how users edit ontologies: Comparing hypotheses about four real-world projects. In *International Semantic Web Conference*. Springer, 551–568.
[30] Patricia L Whetzel, Natalya F Noy, Nigam H Shah, Paul R Alexander, Csongor Nyulas, Tania Tudorache, and Mark A Musen. 2011. BioPortal: enhanced functionality via new Web services from the National Center for Biomedical Ontology to access and use ontologies in software applications. *Nucleic acids research* 39, suppl_2 (2011), W541–W545.
[31] Wenpu Xing and Ali Ghorbani. 2004. Weighted pagerank algorithm. In *Proceedings of the Second Annual Conference on Communication Networks and Services Research, 2004.* IEEE, 305–314.

Product-Aware Helpfulness Prediction of Online Reviews

Miao Fan, Chao Feng, Lin Guo, Mingming Sun, Ping Li
Cognitive Computing Lab (CCL), Baidu Research
{fanmiao,v_fegchao,v_guolin02,sunmingming01,liping11}@baidu.com

ABSTRACT

Helpful reviews are essential for e-commerce and review websites, as they can help customers make quick purchase decisions and merchants to increase profits. Due to a great number of online reviews with unknown helpfulness, it recently leads to promising research on building automatic mechanisms to assess review helpfulness. The mainstream methods generally extract various linguistic and embedding features solely from the text of a review as the evidence for helpfulness prediction. We, however, consider that the helpfulness of a review should be fully aware of the metadata (such as the title, the brand, the category, and the description) of its target product, besides the textual content of the review itself. Hence, in this paper we propose an end-to-end deep neural architecture directly fed by both the metadata of a product and the raw text of its reviews to acquire product-aware review representations for helpfulness prediction. The learned representations do not require tedious labor on feature engineering and are expected to be more informative as the *target-aware* evidence to assess the helpfulness of online reviews. We also construct two large-scale datasets which are a portion of the real-world web data in Amazon and Yelp, respectively, to train and test our approach. Experiments are conducted on two different tasks: helpfulness identification and regression of online reviews, and results demonstrate that our approach can achieve state-of-the-art performance with substantial improvements.

KEYWORDS

E-commerce; online reviews; helpfulness prediction; neural networks; benchmark datasets

ACM Reference Format:
Miao Fan, Chao Feng, Lin Guo, Mingming Sun, Ping Li. 2019. Product-Aware Helpfulness Prediction of Online Reviews. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313523

1 INTRODUCTION

The e-commerce and review websites dramatically facilitate our daily lives as they provide a massive number of products and businesses available online and allow us to access to the comments made by experienced consumers. In order to find a desirable product, we prefer to browsing the online reviews of a product in addition to its descriptions, as we believe that the online reviews can provide more subjective and informative opinions from various perspectives on the product besides the objective descriptions given by its merchant. With the explosive growth of the transaction volume of the e-commerce market in recent years, it is common for an online product commented and rated by thousands of purchasers as illustrated by Figure 1. As a result, it is quite time-consuming for potential consumers to sift through all the online reviews with uneven qualities to make purchase decisions.

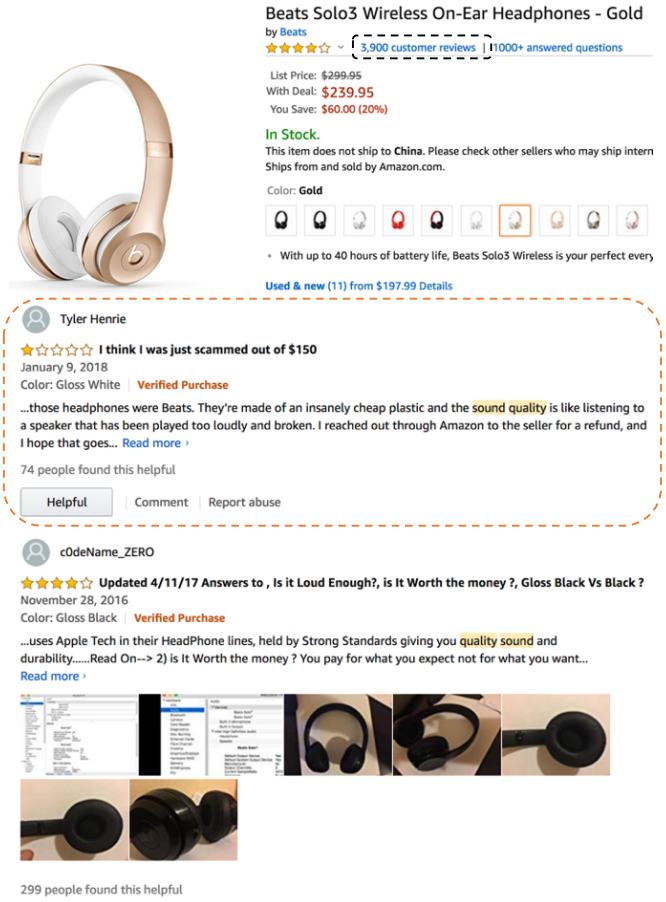


Figure 1: A screenshot of an online product sold on Amazon.com: *Beats Solo3 Wireless On-Ear Headphones - Gold* which receives 3,900 customer reviews. However, we have no ideas on the helpfulness of most reviews.

Therefore, helpful reviews are essential for e-commerce services, as they are able to bridge the gap between customers and merchants in a *win-win* manner, where customers tend to make quick purchase decisions once catching sight of helpful reviews and merchants can increase profits by surfacing the helpful reviews. To discover the helpful reviews, some platforms such as Amazon and Yelp have launched a module (see the "Helpful" buttons beneath each customer review shown by Figure 1) which allows users to

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313523

give feedbacks on the helpfulness of online reviews. This featured module proved by a recent study[1] increases the revenue of Amazon with an estimated 27 billion U.S. dollars annually. Although the crowd-sourcing module could help find a fraction of helpful reviews, roughly 60% online reviews in Amazon.com and Yelp.com did NOT receiving any vote of helpfulness or unhelpfulness. This phenomenon on unknown helpfulness is even more common in low-traffic items including those less popular and new arrival products.

It leads to a promising research direction on building an automatic helpfulness prediction system for online reviews, which we believe could be as useful as a product recommendation engine in e-commerce. So far as we know, a series of work on review helpfulness prediction has been proposed from two perspectives: 1) some work leverages domain-specific knowledge to extract a wide range of hand-crafted features (including structural, lexical, syntactic, emotional, semantic and argument features) from the text of reviews as the evidence for off-the-shelf learning tools; and 2) recent studies modify the convolutional neural network [9, 10] to acquire low-dimensional features from the raw text of online reviews. Generally speaking, these mainstream approaches extract various linguistic and embedding features solely from the text of a review as the evidence for helpfulness prediction.

We, however, suggest that the helpfulness of a review should be fully aware of the meta-data (e.g., title, brand, category, description) of the target product besides the textual content of the review itself. Take the online customer review (circled by the orange dashed box) shown by Figure 1 as an example. The effective features to indicate that it is a helpful review on the product: *Beats Solo3* are probably the phrase "*sound quality*" and word "*headphone*". But the same textual features are hardly considered to be *helpful* when appearing in a comment on a *Nikon Digital Camera*.

In this paper, we propose an end-to-end deep neural architecture to capture the intrinsic relationship between the meta-data of a product and its numerous comments that could be beneficial to discover the helpful reviews. Our model is directly fed by both the title of a product and the raw text of its reviews, and then acquire product-aware review representations from the supervision of helpfulness scores given by the crowd-sourcing module. The learned neural representations do not require tedious labor on feature engineering and are expected to be more informative as the product-aware evidence to assess the helpfulness of online reviews. Given the drawbacks that prior systems have been evaluated by different datasets and not built on the successes of each other, we also construct two large-scale datasets, i.e. *Amazon-9* and *Yelp-5*, which are a portion of the real-world web data in Amazon.com and Yelp.com, respectively, for the successive assessment on the helpfulness prediction of online reviews. The mainstream approaches mentioned in this paper are re-implemented in addition to our model, for fair comparison. Extensive experiments are conducted on two different application scenarios: helpfulness identification and regression of online reviews using the two benchmark datasets. Experimental results demonstrate that our model can achieve state-of-the-art performance on the two tasks with significant absolute improvements (**4.25%** AUROC in the identification of helpful reviews and **4.40%** R^2-score in the regression of helpfulness voting).

[1] https://articles.uie.com/magicbehindamazon/

2 RELATED WORK

An up-to-date and comprehensive survey on various approaches on helpfulness prediction of online reviews was recently conducted by Ocampo Diaz and Ng [17]. According to their survey, these mainstream methods generally extract various linguistic and embedding features solely from the text of an online review as the evidence for helpfulness prediction. We go further and categorize those approaches into two classes in terms of the way of acquiring supportive features for helpfulness prediction, i.e., learning with hand-crafted features (see Section 2.1) and from deep neural networks (see Section 2.2), respectively.

2.1 Learning with Hand-crafted Features

So far as we know, a series of conventional approaches on review helpfulness prediction leverage domain-specific knowledge to extract a wide range of hand-crafted features from the text of customer reviews as the evidence fed into off-the-shelf classifiers or regressors such as SVM [1, 5] or Random Forest [12, 23]. According to [13] and [26], these hand-crafted features involve:

- *Structural features* (STR) [16, 25]: The structural evidence refers to the number of tokens, the number of sentences, the average length of sentences, and even the star rating of an online review. These important features indicate the attitude of the buyers when they write down their comments.
- *Lexical features* (LEX) [8, 24]: Inspired by the idea of text classification, the bag-of-words (BOW) features are essential to helpfulness prediction of online reviews. Explicitly, we usually remove the stop words and non-frequent words, extract unigrams (UGR) and bigrams (BGR) and weight these terms by the measurement of *tf-idf* as the lexical features.
- *Syntactic features* [8]: We can also obtain the part-of-speech (POS) tag of each token in a review. The syntactic features are composed of the percentages of tokens that are nouns, verbs, adjectives, and adverbs, respectively.
- *Emotional features*: Martin and Pu [14] used the Geneva Affect Label Coder (GALC) dictionary [20] to define 36 emotion states of a review. The emotional features include the number of occurrences of each emotional state plus one additional dimension for the number of non-emotional words.
- *Semantic features*: Yang et al. [26] leveraged the General Inquirer (INQUIRER) dictionary [22] to map each word in a review into a semantic tag. This is similar to the way of obtaining emotional features. The semantic features are formulated by a vector where each entry records the number of the occurrences of each semantic tag.
- *Argument features*: Liu et al. [13] explored more intricate linguistic features such as evidence-conclusion discourse relations, also known as arguments, to study the helpfulness of an online review. To be exact, they adopted different granularities of argument features, e.g., the number of arguments, the number of words in arguments, etc.

2.2 Learning from Deep Neural Nets

The emergence of Deep Learning [11] brings in a good insight that we do not have to manually design heuristic rules to extract domain-specific features for learning tasks. To avoid the tedious

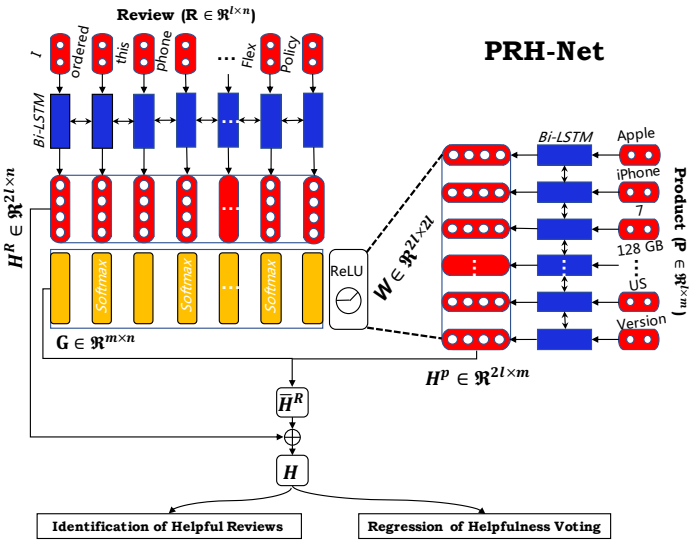


Figure 2: Framework of our neural architecture for product-aware review helpfulness prediction: PRH-Net. $\mathbf{H}^P \in \mathbb{R}^{2l\times m}$ and $\mathbf{H}^R \in \mathbb{R}^{2l\times n}$ are the contextual embeddings of the words in the review R and the product P, respectively. $\mathbf{G} \in \mathbb{R}^{m\times n}$ is the semantic matching matrix between R and P. $\overline{\mathbf{H}}^R \in \mathbb{R}^{2l\times n}$ is the product-aware review repesentation attended by G.

labor on feature engineering for helpfulness prediction of online reviews, the research community recently made several attempts:

- *Embedding-gated CNN* (EG-CNN) [2, 3]: This work adopts a convolutional neural network model [9] that is able to extract multi-granularity text features from reviews. As different words may contribute to the meaning of a review diversely, Chen et al. [2] suggest using word-level embedding-gates to control the word embeddings fed into the CNN model.
- *Multi-task Neural Learning* (MTNL) [4]: Fan et al. [4] introduce a multi-task neural learning (MTNL) paradigm for identifying helpful reviews. The main task, i.e. identifying helpful reviews, leverages the convoluted neural representations from the textual content of reviews. In order to make more accurate predictions on review helpfulness, the convoluted embeddings are also used to fit the star ratings of reviews as an auxiliary task.

Though MTNL [4] and EG-CNN [2, 3] are two newly proposed neural approaches which perform higher on review helpfulness prediction than the hand-crafted methods, these mainstream approaches generally acquire various embedding features solely from the textual content of a review as the evidence without considering the corresponding product, which we believe is inadequate for building a feasible system to discover helpful online reviews.

3 PROPOSED MODEL

In this section, we elaborate our neural model, abbr. PRH-Net, for product-aware review helpfulness prediction. PRH-Net is devised by the motivation that the helpfulness of an online review should be fully aware of the title of its target product besides the textual content of the review itself. As shown by Figure 2, it is composed of two components: 1) the local contextual embeddings of a review and 2) the product-aware distributed representations of the review. We will then explain how to model the two components.

Suppose that we have a product title P and one of its online reviews R. We use m and n to denote the number of tokens/words in the product title P and the review R, respectively. Firstly, we align each token with the embedding dictionary acquired by the word embedding approaches such as Word2Vec [15], Glove [18] or Elmo [19] to initialize the distributed representations of the product title $\mathbf{P} \in \mathbb{R}^{l\times m}$ and the review $\mathbf{R} \in \mathbb{R}^{l\times n}$.

To achieve the local contextual embeddings of the review R, we use a Bi-LSTM network [21] which takes the word embeddings of the review $\mathbf{R}$ as input:

$$\mathbf{H}^R = \text{Bi-LSTM}(\mathbf{R}). \tag{1}$$

$\mathbf{H}^R \in \mathbb{R}^{2l\times n}$ stands for the contextual embeddings where each word can obtain two hidden units with the length of $2l$ encoding both the backward and the forward contextual information of the review locally.

Similarly, we can re-fine the word embeddings of the product title $\mathbf{P}$ via another Bi-LSTM network:

$$\mathbf{H}^P = \text{Bi-LSTM}(\mathbf{P}), \tag{2}$$

and achieve the contextual embeddings of the product title $\mathbf{H}^P \in \mathbb{R}^{2l\times m}$. To make the contextual embeddings of the review fully aware of the product title, we devise a word-level matching mechanism as follows,

$$\mathbf{Q} = \text{ReLU}(\mathbf{W}^P\mathbf{H}^P + \mathbf{b}^P \otimes \mathbf{e}_P)^{\mathrm{T}}\mathbf{H}^R \tag{3}$$

where $\mathbf{W}^P \in \mathbb{R}^{2l\times 2l}$ is the weight matrix and $\mathbf{b}^P \in \mathbb{R}^{2l}$ is the bias vector for the Rectifier Linear Unit (ReLU). The outer product $\otimes$ copys the bias vector $\mathbf{b}^P$ m times to generate a $2l \times m$ matrix. Then $\mathbf{Q} \in \mathbb{R}^{m\times n}$ is the sparse matrix that holds the word-level matching information between the product title P and the review R. If we further apply the softmax function to each column of $\mathbf{Q}$, we will obtain $\mathbf{G} \in \mathbb{R}^{m\times n}$, the i-th column of which represents the normalized attention weights over all the words in product title P for the i-th word in the review R:

$$\mathbf{G} = \text{softmax}(\mathbf{Q}). \tag{4}$$

Then we can use the attention matrix $\mathbf{G} \in \mathbb{R}^{m\times n}$ and the contextual embeddings of the product $\mathbf{H}^P \in \mathbb{R}^{2l\times m}$ to re-form the product-aware review representation $\overline{\mathbf{H}}^R \in \mathbb{R}^{2l\times n}$:

$$\overline{\mathbf{H}}^R = \mathbf{H}^P\mathbf{G}. \tag{5}$$

Driven by original motivation, we need to join the local contextual embeddings of the review ($\mathbf{H}^R$) and the product-aware distributed representations of the review ($\overline{\mathbf{H}}^R$) together for predicting its helpfulness with the feature matrix $\mathbf{H} \in \mathbb{R}^{2l\times n}$:

$$\mathbf{H} = \mathbf{H}^R + \overline{\mathbf{H}}^R. \tag{6}$$

$\mathbf{H}$ can also benefit from the idea of ResNet [6] that efficiently acquires the residual between $\mathbf{H}^R$ and $\overline{\mathbf{H}}^R$, and provides a highway to update $\mathbf{H}^R$ if the residual is tiny.

Table 1: The statistics of the Amazon-9 dataset for helpfulness prediction of online reviews in Amazon.com. #(P.): the number of products; #(R.): the number of reviews; #(R. ≥ 1v.): the number of the reviews, each receiving at least 1 vote regardless of helpfulness/unhelpfulness; #(R. ≥ 0.75h.r.): the number of the reviews, each regarded as helpful by at least 75% votes.

Category (Amazon-9)	Training Set				Test Set			
	# (P.)	# (R.)	# (R. ≥ 1v.)	# (R. ≥ 0.75h.r.)	# (P.)	# (R.)	# (R. ≥ 1v.)	# (R. ≥ 0.75h.r.)
Books	1,157,801	16,240,648	8,884,688	5,538,695	288,901	4,066,140	2,223,022	1,378,901
Clothing, Shoes & Jewelry	381,980	4,599,396	1,480,371	1,076,792	96,018	1,149,157	371,251	269,701
Electronics	223,058	6,220,960	2,597,464	1,660,149	56,117	1,577,881	658,894	422,155
Grocery & Gourmet Food	77,019	1,044,446	441,009	294,932	19,301	250,526	105,264	70,896
Health & Personal Care	118,097	2,395,311	1,094,046	686,932	29,552	581,376	267,255	166,135
Home & Kitchen	176,288	3,406,157	1,479,561	1,090,746	44,415	839,357	361,686	267,284
Movies & TV	126,719	3,614,284	2,078,390	996,024	31,686	867,957	499,574	243,851
Pet Supplies	44,926	997,365	362,499	270,017	11,232	236,222	85,979	63,946
Tools & Home Improvement	107,504	1,538,851	637,665	450,244	26,997	386,130	162,769	114,990
TOTAL	2,413,392	40,057,418	19,055,639	12,064,531	604,219	9,954,746	4,735,694	2,997,859

Table 2: The statistics of the Yelp-5 dataset for helpfulness prediction of online reviews in Yelp.com. #(P.): the number of products; #(R.): the number of reviews; #(R. ≥ 1v.): the number of the reviews, each receiving at least 1 vote regardless of helpfulness/unhelpfulness; #(R. ≥ 0.75h.r.): the number of the reviews, each regarded as helpful by at least 75% votes.

Category (Yelp-5)	Training Set				Test Set			
	# (P.)	# (R.)	# (R. ≥ 1v.)	# (R. ≥ 0.75h.r.)	# (P.)	# (R.)	# (R. ≥ 1v.)	# (R. ≥ 0.75h.r.)
Beauty & Spas	14,066	301,822	162,108	90,003	3,565	76,864	41,555	23,006
Health & Medical	12,034	173,480	103,158	66,616	3,009	43,764	25,979	16,791
Home Services	13,233	201,234	116,011	76,709	3,346	48,085	27,980	18,846
Restaurants	44,597	2,948,004	1,472,688	590,588	11,152	706,793	352,268	142,722
Shopping	22,929	355,366	212,954	100,373	5,725	87,312	51,571	24,105
TOTAL	106,859	3,979,9061	2,066,919	924,289	26,794	962,818	499,353	225,470

4 EXPERIMENTS

4.1 Benchmark Datasets

We find two well-formatted JSON resources online which contain plenty of meta-data (including titles, brands, categories, and descriptions) of products and numerous customer reviews. One is the data collection[2] of Amazon.com crawled by He and McAuley [7] up to July 2014. The other one is the dump file[3] directly provided by Yelp.com for academic purposes.

We use the product ids ("*asin*" in Amazon and "*business_id*" in Yelp) as the foreign keys to align the meta-data of products with corresponding online reviews. 80% products with reviews are randomly picked as the training set, leaving the rest as the test set. In this way, two benchmark datasets, i.e. *Amazon-9* and *Yelp-5*, are prepared, and the statistics of the two datasets are shown by Table 1 and Table 2, respectively.

In line with Table 1, *Amazon-9* covers more than 3 million products spreading over nine different categories in Amazon.com. About 50 million online reviews are included, but less than 48% (roughly 24 million reviews) of them receive at least 1 vote regardless of helpfulness/unhelpfulness by crowd-sourcing. As for *Yelp-5* shown by Table 2, it contains about 130 thousand businesses which fall into five categories in Yelp.com. The proportion of voted reviews in *Yelp-5* is also lower, i.e., roughly 52% (about 2.5 million reviews).

In this work, we regard the reviews which receive at least 1 vote, i.e. the column named after # (R. ≥ 1v.) in Table 1 and Table 2, as the experimental samples. In Amazon.com, the crowd-sourcing module for voting helpful reviews provides an "X of Y" score of helpfulness where "Y" stands for the total number of users who participate in voting and "X" denotes the number of users who think the review is helpful. Yelp.com offers more options: *useful*: X,*cool*: Y, and *funny*: Z, to the users who are willing to give feedbacks. Regardless of the difference, we generally regard the reviews which receive at least 0.75 ratio of helpfulness/usefulness, i.e. # (R. ≥ 0.75h.r.), as positive samples, leaving the others as the negative samples.

4.2 Comparison Methods

We compare our model (PRH-Net) with a wide range of prior arts mentioned in Section 2. Specifically, we re-implement the methods learning with hand-crafted features and from deep neural networks. The up-to-date neural approaches involve the embedding-gated CNN (EG-CNN) [2, 3] and the multi-task neural learning (MTNL) architecture [4] for review helpfulness prediction. The hand-crafted

[2]The data collection is available at http://jmcauley.ucsd.edu/data/amazon/links.html
[3]The dump file can be downloaded from https://www.yelp.com/dataset

Table 3: Comparison of the performance (AUROC) of mainstream approaches on identifying helpful reviews evaluated by the test sets of Amazon-9. (*italic fonts: the best performance among the baseline approaches; bold fonts: the state-of-the-art performance of all the approaches)**

Category (Amazon-9)	Area under Receiver Operating Characteristic (AUROC)								
	STR	LEX	GALC	INQUIRER	FUSION (SVM)	FUSION (R.F.)	EG-CNN	MTNL	PRH-Net
Books	0.595	0.572	0.610	0.620	0.594	0.601	0.625	*0.629**	**0.652** (+0.023)
Clothing, Shoes & Jewelry	0.559	0.538	0.565	*0.608**	0.587	0.557	0.590	0.592	**0.614** (+0.006)
Electronics	0.590	0.555	0.593	*0.627**	0.584	0.588	0.615	0.618	**0.644** (+0.017)
Grocery & Gourmet Food	0.540	0.526	0.566	0.618	0.537	0.556	0.613	*0.638**	**0.715** (+0.077)
Health & Personal Care	0.560	0.533	0.569	0.617	0.599	0.565	0.617	*0.624**	**0.672** (+0.048)
Home & Kitchen	0.572	0.545	0.576	0.609	0.579	0.573	0.605	*0.611**	**0.630** (+0.019)
Movies & TV	0.613	0.562	0.624	0.637	0.605	0.617	0.648	*0.652**	**0.675** (+0.023)
Pet Supplies	0.560	0.542	0.585	0.603	0.548	0.558	0.580	*0.619**	**0.679** (+0.060)
Tools & Home Improvement	0.584	0.548	0.580	0.592	0.565	0.586	0.607	*0.621**	**0.644** (+0.023)
MACRO AVERAGE	0.575	0.547	0.585	0.615	0.578	0.578	0.611	*0.623**	**0.658** (+0.035)
MICRO AVERAGE (Primary)	0.587	0.559	0.598	0.620	0.589	0.591	0.620	*0.625**	**0.651 (+0.026)**

Table 4: Comparison of the performance (AUROC) of mainstream approaches on identifying helpful reviews evaluated by the test sets of Yelp-5. (*italic fonts: the best performance among the baseline approaches; bold fonts: the state-of-the-art performance of all the approaches)**

Category (Yelp-5)	Area under Receiver Operating Characteristic (AUROC)								
	STR	LEX	GALC	INQUIRER	FUSION (SVM)	FUSION (R.F.)	EG-CNN	MTNL	PRH-Net
Beauty & Spas	0.512	0.500	0.527	0.570	0.521	0.541	0.571	*0.581**	**0.642** (+0.061)
Health & Medical	0.525	0.517	0.538	0.576	0.539	0.531	0.580	*0.596**	**0.665** (+0.069)
Home Services	0.528	0.528	0.562	0.584	0.535	0.538	0.563	*0.603**	**0.732** (+0.129)
Restaurants	0.559	0.516	0.552	0.582	0.569	0.554	0.581	*0.605**	**0.658** (+0.053)
Shopping	0.528	0.518	0.560	0.609	0.542	0.555	0.572	*0.619**	**0.674** (+0.055)
MACRO AVERAGE	0.530	0.516	0.548	0.584	0.541	0.544	0.573	*0.601**	**0.674** (+0.073)
MICRO AVERAGE (Primary)	0.548	0.516	0.551	0.584	0.559	0.551	0.578	*0.604**	**0.663 (+0.059)**

features include the structural features (STR) [16, 25], the lexical features (LEX) [8, 24], the emotional features (GALC) [14] and the semantic features (INQUIRER) [26]. We also add two more experiments on integrating all the hand-crafted features via the Support Vector Machines (SVM) and the Random Forest (R.F.) model for review helpfulness prediction.

4.3 Application Scenarios

Most previous studies just reported their performance on either the task of review helpfulness identification or regression. In this part, we assess the performance of PRH-Net by comparing our model with all the other approaches on both tasks.

4.3.1 Identification of Helpful Reviews. We use the data shown by the column named #(R. $\geq$ 1v.) in Table 1 and Table 2 to conduct this task. Within the data for binary classification, the reviews belonging to the column #(R. $\geq$ 0.75h.r.) are regarded as positive samples. As both the training and test sets are imbalanced, we adopt the Area under Receiver Operating Characteristic (AUROC) as the metric to evaluate the performance of all the approaches on helpful review identification. As shown by Table 3 and Table 4, MTNL [4] achieves the up-to-date performance on this classification task among the baseline approaches as it shows the best performance on 12 of 14 categories in *Amazon-9* and *Yelp-5* datasets. Our model (PRH-Net) surpasses MTNL on both datasets and obtains state-of-the-art (micro-averaged) results of 65.1% AUROC (*Amazon-9*) and 66.3% AUROC (*Yelp-5*) with absolute improvements of **2.6%** AUROC and **5.9%** AUROC, respectively.

4.3.2 Regression of Helpfulness Voting. In this task, all the approaches are required to predict the fraction of helpful votes that each review receives. We still use the data in the column named of #(R. $\geq$ 1v.) in Table 1 and Table 2 as the training and test sets. The Squared Correlation Coefficient (R^2-score) is adopted as the metric to evaluate the performance of all the approaches on helpfulness score regression. Table 5 and Table 6 show that MTNL [4] achieves the up-to-date performance on this regression task among the baselines. Our model (PRH-Net) outperforms MTNL on both datasets and obtains state-of-the-art (micro-averaged) results of 55.2% R^2-score (*Amazon-9*) and 58.2% R^2-score (*Yelp-5*) with absolute improvements of **3.5%** R^2-score and **5.3%** R^2-score, respectively.

Table 5: Comparison of the performance (R^2-score) of mainstream approaches on helpfulness voting regression evaluated by the test sets of Amazon-9. (*italic fonts: the best performance among the baseline approaches; bold fonts: the state-of-the-art performance of all the approaches)**

Category (Amazon-9)	Squared Correlation Coefficient (R^2-score)								
	STR	LEX	GALC	INQUIRER	FUSION (SVM)	FUSION (R.F.)	EG-CNN	MTNL	PRH-Net
Books	0.250	0.234	0.312	0.458	0.401	0.441	0.506	*0.549**	**0.586** (+0.037)
Clothing, Shoes & Jewelry	0.245	0.307	0.368	0.428	0.360	0.376	0.491	*0.510**	**0.567** (+0.057)
Electronics	0.223	0.291	0.333	0.450	0.416	0.357	0.394	*0.489**	**0.516** (+0.027)
Grocery & Gourmet Food	0.242	0.319	0.419	0.450	0.426	0.425	0.469	*0.506**	**0.569** (+0.063)
Health & Personal Care	0.233	0.332	0.376	0.433	0.420	0.399	0.506	*0.509**	**0.537** (+0.028)
Home & Kitchen	0.236	0.298	0.330	0.464	0.339	0.361	0.402	*0.498**	**0.513** (+0.015)
Movies & TV	0.253	0.228	0.312	0.387	0.360	0.352	0.393	*0.453**	**0.495** (+0.042)
Pet Supplies	0.237	0.237	0.285	0.420	0.301	0.339	0.400	*0.473**	**0.523** (+0.050)
Tools & Home Improvement	0.234	0.201	0.287	0.437	0.247	0.302	0.398	*0.481**	**0.503** (+0.022)
MACRO AVERAGE	0.239	0.272	0.336	0.436	0.363	0.372	0.440	*0.496**	**0.534** (+0.038)
MICRO AVERAGE (Primary)	0.243	0.258	0.325	0.445	0.385	0.399	0.463	*0.517**	**0.552 (+0.035)**

Table 6: Comparison of the performance (R^2-score) of mainstream approaches on helpfulness voting regression evaluated by the test sets of Yelp-5. (*italic fonts: the best performance among the baseline approaches; bold fonts: the state-of-the-art performance of all the approaches)**

Category (Yelp-5)	Squared Correlation Coefficient (R^2-score)								
	STR	LEX	GALC	INQUIRER	FUSION (SVM)	FUSION (R.F.)	EG-CNN	MTNL	PRH-Net
Beauty & Spas	0.283	0.384	0.511	0.537	0.349	0.418	0.550	*0.552**	**0.624** (+0.072)
Health & Medical	0.253	0.454	0.433	0.557	0.478	0.459	0.572	*0.573**	**0.635** (+0.062)
Home Services	0.264	0.452	0.416	0.554	0.492	0.481	0.570	*0.575**	**0.645** (+0.070)
Restaurants	0.256	0.338	0.383	0.501	0.356	0.407	0.510	*0.518**	**0.564** (+0.046)
Shopping	0.306	0.347	0.417	0.523	0.334	0.400	0.537	*0.540**	**0.606** (+0.066)
MACRO AVERAGE	0.272	0.395	0.432	0.534	0.402	0.433	0.548	*0.552**	**0.615** (+0.063)
MICRO AVERAGE (Primary)	0.264	0.355	0.402	0.512	0.367	0.414	0.523	*0.529**	**0.582 (+0.053)**

5 CONCLUSION AND FUTURE WORK

This paper engages in the emerging research on helpfulness prediction of online reviews. Our idea is driven by the motivations that 1) the helpfulness of an online review should be fully aware of the meta-data of its target product besides the textual content of the review itself; 2) the hand-crafted features requiring tedious labor are domain-specific and prone to high generation error; and 3) there is no widely-used benchmark dataset for constantly improving intelligent systems to precisely assess the helpfulness of online reviews.

To address the problems above, we contribute an end-to-end neural architecture which can automatically acquire product-aware review representations besides the textual embeddings of reviews as more informative evidence for review helpfulness prediction. We also construct two large-scale and real-world benchmark datasets, i.e. *Amazon-9* and *Yelp-5*, for the sake of 1) fairly conducting the performance comparison of all the approaches on review helpfulness prediction, and 2) leaving the datasets available for successive studies. Specifically, we run extensive experiments on our newly constructed datasets under the application scenarios of helpfulness identification and regression. Experimental results demonstrate that our model surpasses all the mainstream approaches and achieves state-of-the-art performance with substantial improvements.

For future work, we might consider to study the following topics related to helpfulness prediction of online reviews:

- User-specific and explainable recommendation of helpful reviews: As different users may concern about various aspects of the products online, helpful review recommendation needs to be more user-specific and self-explainable.
- Cross-domain helpfulness prediction of online reviews [3]: Given that it costs a lot on manually annotating plenty of helpful reviews in a new domain, we should explore effective approaches on transferring useful knowledge from limited labeled samples in another domain.
- Enhancing the prediction of helpful reviews with unlabeled data: As a small proportion of reviews could be heuristically regarded as helpful or unhelpful, it, therefore, becomes a promising study to automatically predict the helpfulness of online reviews based on the small amount of labeled data and a huge amount of unlabeled data.

REFERENCES

[1] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology* 2, 3 (2011), 27.

[2] Cen Chen, Minghui Qiu, Yinfei Yang, Jun Zhou, Jun Huang, Xiaolong Li, and Forrest Bao. 2018. Review Helpfulness Prediction with Embedding-Gated CNN.

[3] Cen Chen, Yinfei Yang, Jun Zhou, Xiaolong Li, and Forrest Sheng Bao. 2018. Cross-Domain Review Helpfulness Prediction Based on Convolutional Neural Networks with Auxiliary Domain Discriminators. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL), Volume 2 (Short Papers)*. Association for Computational Linguistics, 602–607.

[4] Miao Fan, Yue Feng, Mingming Sun, Ping Li, Haifeng Wang, and Jianmin Wang. 2018. Multi-Task Neural Learning Architecture for End-to-End Identification of Helpful Reviews. In *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. 343–350.

[5] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research (JMLR)* 9 (2008), 1871–1874.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. 770–778.

[7] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*. International World Wide Web Conferences Steering Committee, 507–517.

[8] Soo-Min Kim, Patrick Pantel, Tim Chklovski, and Marco Pennacchiotti. 2006. Automatically Assessing Review Helpfulness. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 423–430.

[9] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 1746–1751.

[10] Yann LeCun and Yoshua Bengio. 1998. The Handbook of Brain Theory and Neural Networks. (1998), 255–258.

[11] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

[12] Andy Liaw, Matthew Wiener, et al. 2002. Classification and regression by randomForest. *R news* 2, 3 (2002), 18–22.

[13] Haijing Liu, Yang Gao, Pin Lv, Mengxue Li, Shiqiang Geng, Minglan Li, and Hao Wang. 2017. Using Argument-based Features to Predict and Analyse Review Helpfulness. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 1369–1374.

[14] Lionel Martin and Pearl Pu. 2014. Prediction of Helpful Reviews Using Emotions Extraction. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*. AAAI Press, 1551–1557.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems (NIPS)*. 3111–3119.

[16] Susan M. Mudambi and David Schuff. 2010. What Makes a Helpful Online Review? A Study of Customer Reviews on Amazon.Com. *MIS Quarterly* 34, 1 (March 2010), 185–200.

[17] Gerardo Ocampo Diaz and Vincent Ng. 2018. Modeling and Prediction of Online Product Review Helpfulness: A Survey. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, 698–708.

[18] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[19] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*. Association for Computational Linguistics, 2227–2237.

[20] Klaus R Scherer. 2005. What are emotions? And how can they be measured? *Social science information* 44, 4 (2005), 695–729.

[21] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing (TSP)* 45, 11 (1997), 2673–2681.

[22] Philip J Stone, Robert F Bales, J Zvi Namenwirth, and Daniel M Ogilvie. 1962. The general inquirer: A computer system for content analysis and retrieval based on the sentence as a unit of information. *Behavioral Science* 7, 4 (1962), 484–498.

[23] Vladimir Svetnik, Andy Liaw, Christopher Tong, J Christopher Culberson, Robert P Sheridan, and Bradley P Feuston. 2003. Random forest: a classification and regression tool for compound classification and QSAR modeling. *Journal of chemical information and computer sciences* 43, 6 (2003), 1947–1958.

[24] Wenting Xiong and Diane Litman. 2011. Automatically Predicting Peer-Review Helpfulness. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL)*. Association for Computational Linguistics, 502–507.

[25] Wenting Xiong and Diane J Litman. 2014. Empirical analysis of exploiting review helpfulness for extractive summarization of online reviews.. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING)*. Association for Computational Linguistics, 1985–1995.

[26] Yinfei Yang, Yaowei Yan, Minghui Qiu, and Forrest Bao. 2015. Semantic Analysis and Helpfulness Prediction of Text for Online Product Reviews. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, 38–44.

The World Wants Mangoes and Kangaroos: A Study of New Emoji Requests Based on Thirty Million Tweets

Yunhe Feng
University of Tennessee, Knoxville
yfeng14@utk.edu

Wenjun Zhou
University of Tennessee, Knoxville
wzhou4@utk.edu

Zheng Lu
University of Tennessee, Knoxville
zlu12@utk.edu

Zhibo Wang
Wuhan University
zbwang@whu.edu.cn

Qing Cao
University of Tennessee, Knoxville
cao@utk.edu

ABSTRACT

As emojis become prevalent in personal communications, people are always looking for new, interesting emojis to express emotions, show attitudes, or simply visualize texts. In this study, we collected more than thirty million tweets mentioning the word "emoji" in a one-year period to study emoji requests on Twitter. First, we filtered out bot-generated tweets and extracted emoji requests from the raw tweets using a comprehensive list of linguistic patterns. Then, we examined patterns of new emoji requests by exploring their time, locations, and context. Finally, we summarized users' advocacy behaviors and identified expressions of equity, diversity, and fairness issues due to unreleased but expected emojis, and concluded the significance of new emojis on society. To the best of our knowledge, this paper is the first to conduct a systematic, large-scale study on new emoji requests.

CCS CONCEPTS

• **Information systems** → **Web mining**; **Social networks**; **Information retrieval**; • **Human-centered computing** → *Interactive systems and tools.*

KEYWORDS

emoji analysis; emoji mining; emoji petition; relatedness, fairness and equality in emojis; emoji categorization; emoji profiling

ACM Reference Format:
Yunhe Feng, Wenjun Zhou, Zheng Lu, Zhibo Wang, and Qing Cao. 2019. The World Wants Mangoes and Kangaroos: A Study of New Emoji Requests Based on Thirty Million Tweets. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313728

1 INTRODUCTION

The word emoji comes from the Japanese words e ("picture") and moji ("character") and has a history of nearly 30 years since it originated on Japanese mobile phones in the late 1990s. In 2009, a set of 722 emojis were first officially added into Unicode Standard 5.2 [30]. After Apple introduced the iOS emoji keyboard in 2011, the use of emojis grew rapidly [44]. By 2018, more than 2700 emojis had been added into Unicode Standard 11.0. According to a recent survey [44], almost everyone online (92% of the online population) was using emojis. With the popularity of social networks, nowadays, emojis are used extensively on various social networking platforms, such as Twitter, Facebook, WhatsApp, and Instagram. In particular, nearly half of comments and captions on Instagram have emojis [11].

As the usage of emojis (and social media in general) evolves, new emojis are being continuously requested. The Unicode Consortium[1] updates the official list of Unicode emojis by judging and accepting proposals for new emojis annually. For each candidate emoji, its evidence of frequency from Google Search, Bing Search, Youtube Search and Google Trends must be submitted, and evidence from NGram Viewer and Wikipedia Search are optional. Besides substantial efforts to collect such evidence, this method has several additional drawbacks. First, not all objects with a higher frequency in search engines are more likely to be emojilized. For example, although the "mascot" is heavily searched, it is unlikely to be an emoji because there exists no specific image representing all mascots for different teams, events, and organizations. Second, it completely ignores emoji petitions directly generated by users who have first-hand information regarding the valuable usage context.

However, a systematic study on which new emojis are wanted, when, where and why these emojis are requested, and how to call for these emojis still remains unexplored. Few studies attempted to offer even partial answers to these questions. The emoji satisfaction survey [45] reported mobile message app users always desired more emoji choices, but provided no further detailed answers to the above specific emoji questions. Thomas Dimson [11] showed the emoji usage trend on Instagram from 2010 to 2015 during which a large number of new emojis were proposed, but those new emojis were not studied. Yonatan Zunger [52] analyzed the animal emoji requests of one day (August 3, 2017) using Twitter's Search APIs, and demonstrated the world wanted raccoon and lobster emojis. However, it only studied emojis in a single category on a small-scale in terms of both the number of tweets and the tweeting time span.

To more comprehensively study the emoji requests, this paper investigated more than thirty million tweets mentioning the word "emoji," and proposed a new framework to answer the above questions. Specifically, the framework consisted of the following analyses on the thirty million emoji-mentioned tweets. First, we extracted

[1] https://unicode.org/emoji/

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313728

the requested emoji descriptions and calculated their corresponding frequencies. After filtering out emojis that already existed (i.e., extant emoji requests) and explaining why they were still being requested, we proposed a WordNet [33] based emoji classifier to cluster requested emojis. Then, we studied spatiotemporal patterns of these requests and explored possible reasons why new emojis were requested. Moreover, we presented the common characteristics of requested emojis and advocacy behaviors. Finally, we illustrated the existing relatedness, fairness, and equality problems reflected through emojis, and discussed the positive impacts of new emojis on society.

As the first step to conduct a systematic, large-scale study on new emoji requests, our contributions can be summarized as follows:

- We revealed new and strong evidence of frequency, i.e., the explicit and accurate evidence like "why is there no *foo* emoji", for the Unicode emoji community to evaluate emoji petitions.
- Our study explained why extant emojis were still requested by users, and provided multiple suggestions to enable the timely availability of newly released emojis to users.
- A comprehensive understanding of new emoji requests was offered by profiling spatiotemporal distributions, summarizing advocacy behaviors, and exploring factors that inspire requests.
- We discussed the equality, fairness and diversity in emojis, and presented the potential significance of new emojis in many aspects like business promotion and violence control.

2 RELATED WORK

There is a great deal of work that investigates how people use emojis to facilitate communication and social interactions. Expressing and strengthening emotions [24, 43, 47], and conveying humor [10, 22] and sarcasm [16, 48] are major functions of emojis in interpersonal communication. Emojis are also used to manage conversations, such as maintaining a conversational connection, and ending a thread [9, 51]. In addition, emojis show more diverse functions in close relationships. Kelly et al. [27] reported that emojis encouraged playful interactions, and created shared and secret uniqueness between people in mediated close personal relationships. Similarly, Wiseman et al. [50] examined the repurposing emojis for personalized communication between close partners, friends and family members. An example is using the shared love of the pizza 🍕 to represent romantic love between partners.

Emojis are also having the significant influence on relevant life domains of business, politics, religion, entertainment and food, etc. Various companies used emojis to enrich their promotions, create awareness and attract attention from consumers [21, 28, 32]. Political leaders from the United States [3], Australia [42], and Argentina [20] used emojis in official speech, during interviews, or on social networks. For religions, some users embedded emojis like the folded hands (the prayer hands) emoji 🙏 into their usernames. Another example is the recycling symbol emoji ♻️, which was taking over on Twitter due to its extensive usage by Arabic speakers to represent a shared Islamic *Dua* (supplication or invocation) [40]. In the entertainment field, *The Emoji Movie*, an animated film based on emoji graphics, was released in 2017. Recently, many researchers [18, 25, 26, 47] associated foods with emojis, and suggested emojis to be an easy and non-verbal way to measure food-related emotions, especially for children [18].

As emojis are impacting various aspects of real life, problems of equity, diversity, and fairness caused by emojis are drawing more attention. To promote gender equality through emojis, researchers from Google proposed a set of emojis reflecting a wide range of professions for women (as well as men) with a goal of highlighting the diversity of women's careers [4]. The *person* emoji 👱, an adult with no gender specified, had become available as a gender-inclusive alternative to the man 👱‍♂️ or the woman 👱‍♀️ since Unicode Standard 10.0 in 2017. A recent study revealed that 0.13% of all emojis send by Americans were either a rainbow flag 🏳️‍🌈 (commonly known as the lesbian, gay, bisexual and transgender pride flag), men holding hands 👬 or women holding hands 👭 emojis [23]. Since *"people all over the world want to have emoji that reflect more human diversity, especially for skin tone"* [7], the Unicode Consortium released five different skin tone modifiers, which was based on the six tones of the Fitzpatrick scale [17, 49], to enrich human diversity in Unicode Standard 8.0 in 2015. When a human emoji is immediately followed by one skin tone modifier character, the person(s) or body part will be rendered using the specified skin tone. For example, 👶 + {🏻,🏼,🏽,🏾,🏿} → {👶🏻,👶🏼,👶🏽,👶🏾,👶🏿}. To depict diverse hair colors and styles, Unicode Standard 11.0 introduced hair components including red-haired, curly-haired, white-haired, and bald components in 2018 [8]. Besides, some emojis enable multi-person groupings, which enhances family-related emojis diversity significantly, such as single-parent families 👩‍👦 👨‍👧 , and homoparental families 👨‍👨‍👦 👩‍👩‍👧.

3 IDENTIFYING NEW EMOJI REQUESTS

3.1 Data Curation

We used Twitter's Streaming APIs, which enable developers to filter and collect real-time tweets, to crawl all the English tweets containing the word "emoji". In our study, more than thirty million tweets of interest were crawled in total from Oct. 2017 to Oct. 2018. The collected tweets were formatted in JavaScript Object Notation (JSON) files with named attributes and associated values [46].

To eliminate the side effects of bots on Twitter, we followed approaches proposed by Ljubesic et al. [29] to filter out those bot-generated tweets. More specifically, we removed eleven users who produced on average more than 10 "emoji" tagged tweets per day. For each of users having more than 100 collected postings, we calculated the time (in minutes) between her/his two successive tweets and removed those users whose three most frequent time spans between postings covered more than 90% of their overall production. This method deleted overall 131 users and 43,461 tweets.

We then extracted information of interest such as user profiles, tweet contents, timestamps, and geo tags, from JSON files, and built the *complete dataset* and the *unique dataset* for emoji analysis subtasks in Section 4 and Section 5 respectively. The *complete dataset* consisted of four types of tweets including general tweets, retweets, quoted tweets and replies. The *unique dataset* only contained the original tweets and excluded duplicated tweets like retweets. As tweets are usually composed of incomplete, noisy and poorly structured sentences due to the frequent presence of abbreviations, irregular expressions, ill-formed words and non-dictionary terms, a series of preprocessing steps were applied to reduce noise in tweets.

For example, we removed URLs and non-ASCII characters except Unicode characters reserved for emojis.

3.2 Emoji Extraction Using Linguistic Patterns

Note that not all collected tweets were petitions for new emojis, e.g., the tweet like "I love this emoji!" was crawled as well since it contained the keyword "emoji". Therefore, we needed to identify emoji-requested tweets and extracted wanted emojis. However, Twitter users could choose different words and sentence patterns to express their expectations of new emojis, which made the emoji extraction challenging. Yonatan Zunger [52] assumed that mentions of phrases like "*foo* emoji" were positive statements about desiring such an emoji. Although this hypothesis was claimed to hold true when validating with spot-checks of the matching tweets, it suffered from false positives, e.g., a tweet like "I hate a *foo* emoji!" was incorrectly recognized as desiring the *foo* emoji.

Inspired by [52], we proposed fine-tuned linguistic patterns to detect desired emojis more precisely. Based on our observations, we summarized 49 frequent linguistic patterns and their 2620 variations to match emoji-requested tweets and extract emojis. Ten linguistic patterns are illustrated as follows, and the whole linguistic pattern list and corresponding tweet screenshots are available through http://yunhefeng.me/linguistic_patterns.html.

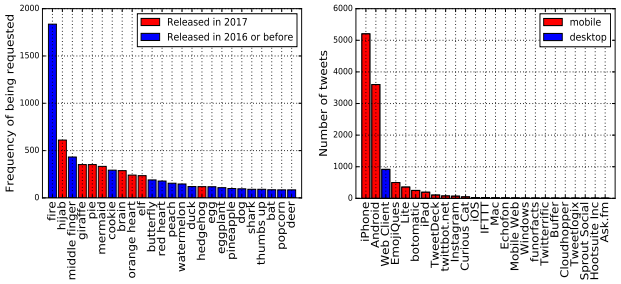
- why is there no *foo* emoji
- where is the *foo* emoji
- need a *foo* emoji
- look for a *foo* emoji
- demand a *foo* emoji
- have no *foo* emoji
- invent a *foo* emoji
- a *foo* emoji is overdue
- still no *foo* emoji
- give us a *foo* emoji

To broaden the matching scopes, we first adopted natural language processing techniques, including part-of-speech (POS) tagging, stemming and lemmatization, before checking tweet contents. For example, "look for a *foo* emoji" would match "looked for an *foo* emoji", "looks for the *foo* emoji", etc. We also considered characteristics of casual English on social networks [6] and fixed common problems, such as the punctuation omission/error (e.g., theres → there's), the wordplay (e.g., neeeeeed → need), and the censor avoidance (e.g., shlt, fck, f***). In addition, we took possible variations on sentence structures of linguistic patterns into account. For example, our linguistic patterns covered not only "need a *foo* emoji" but also "need an emoji of/with/for *foo*".

3.3 Requesting Extant Emojis

When examining extracted emojis, to our surprise, we found hundreds of emojis that had already been released by the Unicode Consortium were still requested extensively. Figure 1(a) demonstrates the top 25 most requested extant emojis. Seven out of the top 10 extant emojis came from the recent Emoji Version 5.0, which was released in May 2017 [13], while we started to crawl the data in Oct. 2017. It is also interesting to note that the percentage of Twitter users on mobile was about 80% [35], but they contributed more than 91.8% extant emoji requests, as shown in Figure 1(b).

We compared Twitter's post-a-tweet interfaces on different platforms to explore why users could not find extant emojis, especially on mobile. On Twitter's desktop site, the post-a-tweet interface offers an emoji picker which contains all latest official emojis. By contrast, on mobile devices, post-a-tweet interfaces of both the



(a) Extant emoji requests (b) Platform distribution

Figure 1: Requesting extant emojis

mobile site and mobile apps have no such emoji pickers. Instead, users have to rely on on-screen keyboards to type emojis, which may cause potential poor user experiences. First, keyboards may not incorporate the latest emojis promptly so that new emojis are unavailable for users. Second, users may not update keyboards to the latest versions to access the recently added emojis, or their mobile operating systems are too out-of-date to be compatible with the latest versions of keyboards. Third, bad emoji keyboard layouts make it difficult for users to find and type intended emojis even if these emojis have been included.

Accordingly, we have the following suggestions and recommendations to improve the user experience of inputting emojis. Mobile users should update installed keyboard software frequently and select the high-quality keyboard with proper emoji arrangements. It is keyboard developers' responsibility to merge newly released emojis into their products as soon as possible and highlight it in "What's New" descriptions of their keyboard apps to remind users that new emojis are available. For app developers, they can add the emoji picker or the search bar to enable users to input emojis without entirely relying on third-party keyboards.

3.4 Emojis Categorization

The emoji categorization plays a big role in facilitating emoji inputs for both mobile and desktop users. Almost all mobile emoji keyboards arrange emojis into categories to alleviate the problems of large lists. Most emoji pickers on social networks also group emojis to help users select wanted emojis quickly and effortlessly. When new emojis come, knowing the number of emojis in each category guides emoji input interface designers to adjust the emoji arrangement, such as increasing the number of emojis per screen. Especially when a large number of new emojis are requested, an automatic emoji classifier is necessary and helpful.

The Unicode Consortium officially categorizes emojis into eight groups, i.e., Smileys & People, Animals & Nature, Food & Drink, Activity, Travel & Places, Objects, Symbols, and Flags. The category of Flags is the easiest one to be detected, since each emoji belonging to this category contains the keyword of "flag". Therefore, we could simply search this keyword in descriptions of each requested emoji to determine whether it should be classified into the flag group.

However, for the rest categories, the method of searching keywords is obviously ineffective because of the difficulty in summarizing a set of keywords representing a particular category. We instead trained a semantic classifier based on WordNet [33], which is a widely-used lexical database for English. A critical concept in WordNet is the synset, namely a set of synonyms sharing a common meaning. In our study, we focused on noun synsets to which words in emoji descriptions belonged. For example, the similarity between two emojis e_1 and e_2 was calculated as the highest similarity score between the two noun synsets containing words in e_1 and e_2. For the similarity score between two synsets, we took the *path_similarity* score, a similarity metric based on the shortest path that connected the senses in the is-a taxonomy, to denote how similar two word senses were. For a testing emoji e_{te}, we calculated its *path_similarity* with each training emoji e_{tr} in each category c. Then, for each category c, we sorted its similarity score list and summed up the top k similarity scores to represent the similarity of the unlabeled emoji e_{te} and the category c. Finally, we set the category with the largest summed similarity as the label for e_{te}.

We performed the 5-fold cross validation on the category dataset collected from Emojipedia[2] for 50 times and achieved an average accuracy of 71.1% with the top k set as 9. One may argue that the accuracy is too low. However, even the official category labels of some emojis are indeed ambiguous. For example, the monkey face (U+1F435) is categorized as animals & nature, but the see-no-evil monkey (U+1F648), hear-no-evil monkey (U+1F649), and speak-no-evil monkey (U+1F64A) are classified as smileys & people. In addition, prior research studies [34, 37] revealed that ambiguities in emoji categorization were common. So, we think the achieved accuracy is acceptable with the messy data.

4 PATTERNS OF NEW EMOJI REQUESTS

4.1 Requested Emojis by Category

We used the proposed keyword matching method and the WordNet-based classifier to categorize requested emojis. Considering requested emojis were too diverse, we only counted emoji requests greater than 10 times. As shown in Table 1, more than 31.8% of wanted emojis were from the Smileys & People category, which might indicate people's great passions for new emojis to express emotions. The public also desired many emojis, including kangaroos and mangoes, from the Animals & Nature and Food & Drink categories. Surprisingly, the number of tweets requesting symbol emojis was very large. After digging into related tweets, we found one tweet petitioning the anarchy symbol emoji, had been retweeted for over 6,000 times, which accounted for more than 27% of the total tweets in the Symbols category.

It is reasonable that categories of the Activity, Travel & Places, and Flags had relative fewer requests, since most emojis in these categories have been released. In addition, it takes a long time to evolve a new activity like a sports game, a new place like electric vehicle charging stations, or flags for new-born countries or influential social movements. The different demands of emojis per category may inspire emoji input interface designers to optimize emoji layouts, such as reserving spaces for new coming emojis, and displaying more emojis per screen. They can even regroup emojis, as suggested by Na'aman et al [34], to enhance user experience.

[2]https://emojipedia.org/

Table 1: Emoji requests by category

Category	# emojis	# tweets	examples
Smileys & People	385	36,790	redhead, ass shaking
Animals & Nature	185	18,059	kangaroo, flamingo
Food & Drink	164	12,067	mango, waffle
Activity	42	2,421	slide, softball
Travel & Places	56	1,946	compass, brick
Objects	161	12,229	broom, red carpet
Symbols	170	21,443	anarchy, infinity
Flags	44	3,156	trans flag, Texas flag

4.2 Temporal Distributions

We aggregated tweets petitioning the same emoji together by month. Figure 2 demonstrates emojis that were requested more than 1000 times throughout one year (Oct. 2017 - Oct. 2018). The circle diameter represents the number of requests made. Although the overall requested number was not very large, emojis of brooms, flamingos and kangaroos appeared consistently in all months. In contrast, heavily requested emojis like the lookout and the red carpet mainly appeared in one or two months. The fact that the broom, flamingo and kangaroo emojis were selected as part of Unicode 11.0 in 2018 or Unicode 12.0 in 2019 implied those emojis that were requested continuously and by multiple users were more likely to be approved by the Unicode Consortium as they reflected the real needs of the majority of online users. We also found the extensive but relatively concentrated emoji requests were usually triggered by celebrities or their followers. For example, petitions of the red carpet emoji were retweeted more than one thousand times by fans of BTS, a South Korean boy band, within 24 hours. After that, there was nearly no petition for the red carpet emoji any more.

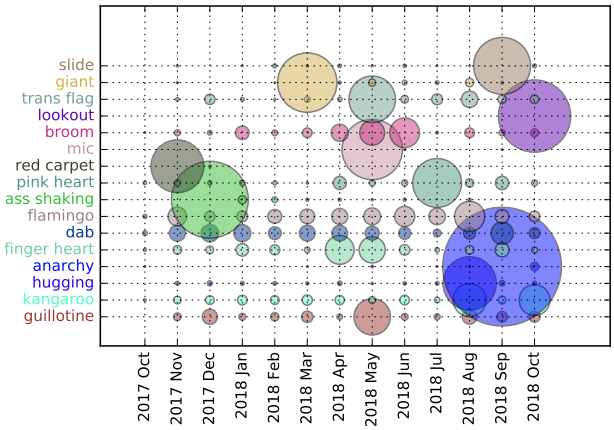


Figure 2: The number of requests per emoji per month. The circle diameter represents the number of requests.

4.3 Geographic Distributions

We utilized geotagged tweets (2.8% of tweets in the *complete dataset*) to profile geographic distributions of new emoji requests at both worldwide and national levels. We observed that people in as many as 110 different countries petitioned for new emojis. As we collected tweets written in English, English-speaking countries, such as United States (73.6%), United Kingdom (10.9%), and Canada (3.2%), contributed the most of emoji requests. It is interesting that non-native English-speaking countries, such as China, Japan, Brazil and Mexico, also expressed their desire for new emojis even in English, which might be one evidence of the world's passion for emojis.

Since most requests were made in the United States, we then focused on the United States to explore the geographic distribution of emoji requests at the national level. As expected, states like California, Texas, and New York made a large number of requests, whereas those states lying at the heartland had low requesting percentages. We think this uneven distribution is caused mainly by the different populations in these regions. After normalizing by state population [39], the geographical distribution was relatively smooth and even across the country, which indicated that people in different states had a similar level of desire for new emojis.

5 BEHAVIORS OF NEW EMOJI REQUESTS

5.1 Context of Emoji Requests

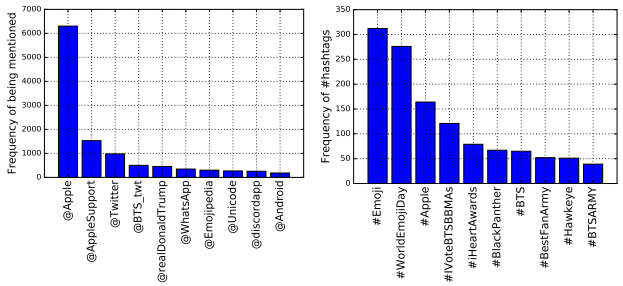
5.1.1 Time-Related Events & Activities. During holidays and festivals, people requested time-sensitive and content-related emojis very frequently, such as the candy cane emoji on Candy Holidays, the carnation emoji on Mother's Day, and the waffle emoji on National Waffle Day. We also found emoji requests were related to popular entertainment products or events. Especially when trends of these popular elements emerged, related emojis were requested extensively by many participants such as movie audiences, music enthusiasts and game players. For example, shortly after Black Panther, a superhero film, was released in early 2018, hundreds of panther emoji requests occurred on Twitter. In addition, periodic reoccurring events like sports games might promote the expectation of new emojis. For example, during 2018 Winter Olympics, many users asked for the Olympic Rings. The yellow card (a serious warning sign in soccer) and red card (a sending-off sign) were petitioned widely in 2018 FIFA World Cup.

5.1.2 Place-Related Interests. Places of interests at different levels, like a single landmark, tourist attractions and even regions or countries, might encourage users to seek new place-related emojis. Many Twitter users visiting Paris claimed for an Eiffel Tower emoji, like "Paris first though!! why's there no Eiffel Tower emoji?!". Similarly, the Mickey and Minnie emoji was requested at Walt Disney World (WDW) Resort like "Guess where I am?!!! WDW (why is there no Mickey and Minnie emoji?!)". Note that the Unicode Consortium does not adopt emojis covered by trademarked logos or copyrighted designs [15, 19]. Residents in Hawaii and Texas looked for their state flag emojis respectively.

5.1.3 Twitter Influencer-Related Behaviors. Emoji requests made by prominent people on Twitter might trigger a widespread discussion of the requested emojis through a massive number of followers. In other words, people were more likely to interact with tweets created by Twitter influencers than those tweeted by unknown Twitter accounts. For example, Enya Umanzor, a famous YouTuber with over 800,000 subscribers to her makeup channel, tweeted "why is there no ass shaking emoji" and garnered 13,000 likes, 2,600 retweets and 34 replies. However, four non-prominent people tweeting for the same ass shaking emoji before Enya Umanzor only got three retweets, no like or reply in total.

5.2 Advocacy Behaviors

When wanted emojis were unavailable, nearly one in three Twitter users would use the symbol of '@' to mention some people or organizations for their attention. Figure 3(a) shows the top 10 Twitter accounts being mentioned by users, where eight of them (except BTS_twt and realDonaldTrump) are specific apps or mobile operating system related Twitter accounts. It is reasonable for ordinary users to seek help from the apps or operating systems, because users thought it was these service providers' responsibility for the nonexistent emojis. The fact that people switched to Twitter to petition new emojis for other apps, e.g., WhatsApp and Discord, could be viewed as a justification for choosing tweet data to study the emoji request.



(a) '@' Mentioning (b) #hashtags Mentioning

Figure 3: Frequency of '@' accounts and #hashtags

In addition to '@' other Twitter accounts, more than 12% of users inserted #hashtags in their tweets when wanted emojis were inaccessible. As we can see from Figure 3(b), the #Emoji is the most frequently created by users, which indicates that the primary concern of these tweets is about emojis. There are also some hashtagged words similar to those '@' mentioned words, such as the #Apple. According to Twitter, the #hashtags are mainly used for indexing keywords or topics, and popular hashtagged words are often trending topics. In our case, users attempted to advocate their desire for more emoji options with the aid of #hashtags.

5.3 Relatedness, Fairness, and Equality

An interesting scenario for the emoji request rose when users complained that there existed an emoji for A but no emoji for B in the same tweets. They thought it was unfair or unreasonable because A and B were usually very similar or related to each other. As emojis are ubiquitous in our lives, such concerns appear in diverse domains as shown in Table 2.

The gender, color, similar function and similar looking can cause a sense of unfairness and inequality. Gender equality and diversity

in emojis are expected by both women and men. Women claimed for the female skier and the woman in tuxedo emojis, while men wanted male-holding-baby and pregnant man emojis. Also, the transgender flag was requested widely. The color is another factor leading to emoji inequalities. Since the blond hair, purple grape, and red ribbon emojis were released, people thought the red hair, green grape, and pink ribbon emojis should be available as well. The similar function can also be an excuse to request new emojis, like mobile phone emojis versus iPad/tablet emojis, guitar emojis versus ukulele emojis, alembic emojis versus test tube emojis, and trophy emojis versus Oscar emojis. Besides, the similarity in looks between two distinct objects can cause unfairness if one of them is unavailable. For example, people reluctant to use the tortoise emoji to represent the turtle believed it was unfair to the turtle.

Emerging technologies, recent social movements, and the equality of political symbols motivate people to petition for new emojis. The bitcoin sign was approved in 2017 as a Unicode character, but not as an emoji. Twitter users wanted an emoji version of the bitcoin to be added. When #MeToo movement reached 1.7 million, Twitter gave it a custom emoji (three raising hands of different skin shades). However, this #MeToo emoji has not been officially supported by the Unicode Consortium and cannot be displayed across multiple platforms. In politics, the equality of both symbols and flags was considered. For example, as there existed the elephant emoji which could be used to represent GOP (the Republican Party), a donkey emoji representing the Democratic Party was requested.

6 SIGNIFICANCE OF NEW EMOJIS

New emojis contain both the unreleased emojis and the emojis needed to be re-designed by tech vendors, like Apple, Google, and Twitter. Identifying and introducing new emojis benefits the society a lot from many aspects, which explains why the Unicode Consortium and vendors update emojis continuously. The newly added hijab (woman with headscarf) emoji through the Hijab Emoji Project campaign led by 15-year-old Saudi Rayouf Alhumedhi promotes inclusivity for about 550 million Muslim women on this earth [2, 12]. Researchers from the Johns Hopkins Bloomberg School of Public Health and the Bill & Melinda Gates Foundation proposed a mosquito emoji to better explain mosquito-borne illnesses like malaria, Zika, dengue and yellow fever in 2017 [31]. Prior work also suggested creating a set of nursing emojis might facilitate health communications for patients and allow them to better understand their health data [41]. As branded emojis helped improve the amount of ads receive by almost 10% [38], brands like furniture company IKEA and fast food restaurant Tim Hortons have released app-specific branded emoji to iconize their products [1].

New appearances of emojis are always desired along with fixing design flaws, considering social influence, etc. When people found the original official lobster emoji and the one designed by Twitter were missing a set of legs, a new anatomically accurate lobster emoji was requested strongly and the four-legged lobster emoji was available soon. The misplaced cheese in Google's burger emoji sparked wide controversy online and was addressed soon by putting the cheese in its correct place [36]. A more recent example is the Apple's bagel emoji released in iOS 12.1 beta 2. Its lackluster appearance caused overwhelming complaints from bagel lovers and birthed the #SadBagel movement for a more appetizing design on Twitter. Apple added cream cheese to its forthcoming bagel emoji after the social media outcry [5]. To curb visual representations of gun violence, all major vendors switched the realistic-looking pistol emoji to a toy water gun in 2018, e.g., Apple (→), Google (→), Microsoft (→), Facebook (→) and Twitter (→) [1, 14].

Table 2: Requesting related emojis in the same tweets

Domain	Available Emoji (A)	Unavailable Emoji (B)
Human Diversity	breast-feeding	male-holding-baby
	man in tuxedo	woman in tuxedo
	blond hair	red hair
Life	pancakes	waffle
	bed	pillow
	wine glass	white wine
Science & Tech	antenna bars	Wi-Fi
	microscope	DNA
	mobile phone	ipad/tablet
Nature	honeybee	fly
	tortoise	turtle
	crab	lobster
Business	Unicode (U+20BF)	Bitcoin
	TOP arrow	bottom arrow
	bar chart	pie chart
Society	#MeToo hashtag	#MeToo in Unicode
	Greenland flag	transgender flag
	water pistol	real gun (AR15)
Politics	elephant for GOP	donkey for Dems.
	Guyana/Ghana flag	pan African flag
	United States flag	Confederate flag
Entmt. & Arts	trophy	Oscar
	videocassette/DVD	cassette tape
	guitar	ukulele

7 CONCLUDING REMARKS

In this paper, we proposed a framework for crawling and analyzing emoji requests on Twitter. We collected more than thirty million English tweets containing the keyword "emoji" throughout a year from Oct. 2017 to Oct. 2018. After filtering out bot-generated tweets, we extracted emoji descriptions using fine-tuned linguistic patterns. Surprisingly, some extant emojis were still frequently requested by many users, which were probably caused by out-of-date emoji keyboards or poor emoji keyboard layouts. For non-existing requested emojis, we categorized them into eight groups using a combination of keyword matching and WordNet-based classifiers. We then profiled temporal and geographic distributions of new emojis at different scales. Emojis requested consistently in every month and by multiple users were more likely to be approved by the Unicode

Consortium. We next summarized three typical contexts of emoji requests, i.e., time-related events & activities, place-related interests, and Twitter influencer-related behaviors. Finally, we presented the equity, diversity, and fairness issues due to unreleased but expected emojis, and discussed the significance of new emojis on society. To the best of our knowledge, this paper is the first to conduct a systematic, large-scale study on new emoji requests.

ACKNOWLEDGMENT

The work reported in this paper was supported in part by the grant USDA NIFA 2017-67007-26150 and the Roy & Audry Fancher Summer Research Award.

REFERENCES

[1] Crystal Abidin and Joel Gn. 2018. Between art and application: Special issue on emoji epistemology. *First Monday* 23, 9 (2018).
[2] Rayouf Alhumedhi, Aphee Messer, Alexis Ohanian, and Jennifer 8. Lee. 2016. *The Hijab Emoji Project.* http://www.hijabemoji.org/
[3] Matt Alt. 2015. *How Emoji Got to the White House.* https://www.newyorker.com/culture/culture-desk/how-emoji-got-to-the-white-house
[4] Rachel Been, Nicole Bleuel, Agustin Fonts, and Mark Davis. 2016. *Expanding Emoji Professions: Reducing Gender Inequality.* http://unicode.org/L2/L2016/16160-emoji-professions.pdf
[5] Jeremy Burge. 2018. *Apple Fixes Bagel Emoji.* https://blog.emojipedia.org/apple-fixes-bagel-emoji/
[6] Eleanor Clark and Kenji Araki. 2011. Text normalization in social media: progress, problems and applications for a pre-processing system of casual English. *Procedia-Social and Behavioral Sciences* 27 (2011), 2–11.
[7] The Unicode Consortium. 2018. *Unicode Technical Standard #51 - UNICODE EMOJI.* http://unicode.org/reports/tr51/#Emoji_Modifiers_Table
[8] The Unicode Consortium. 2018. *Unicode Technical Standard #51 - UNICODE EMOJI.* http://unicode.org/reports/tr51/#hair_components
[9] Henriette Cramer, Paloma de Juan, and Joel Tetreault. 2016. Sender-intended functions of emojis in US messaging. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services.* ACM, 504–509.
[10] Marcel Danesi. 2016. *The semiotics of emoji: The rise of visual language in the age of the internet.* Bloomsbury Publishing.
[11] Thomas Dimson. 2015. Emojineering Part 1: Machine Learning for Emoji Trends. Instagram Engineering Blog. https://instagram-engineering.com/emojineering-part-1-machine-learning-for-emoji-trendsmachine-learning-for-emoji-trends-7f5f9cb979ad.
[12] Melissa Eddy. 2016. *Muslim Teenager Proposes Emoji of Woman Wearing a Head Scarf.* https://www.nytimes.com/2016/09/15/world/europe/muslim-teenager-proposes-emoji-of-woman-wearing-a-head-scarf.html
[13] Emojipedia. 2017. *Emoji Version 5.0 List.* https://emojipedia.org/emoji-5.0/
[14] Emojipedia. 2018. *The Pistol Emoji.* https://emojipedia.org/pistol/
[15] Gabriella E. Ziccarelli Eric Goldman. 2018. *Emojis and intellectual property law.* https://www.wipo.int/wipo_magazine/en/2018/03/article_0006.html
[16] Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *arXiv preprint arXiv:1708.00524* (2017).
[17] Thomas B Fitzpatrick. 1975. Soleil et peau. *J Med Esthet* 2 (1975), 33–34.
[18] Katherine E Gallo, Marianne Swaney-Stueve, and Delores H Chambers. 2017. A focus group approach to understanding food-related emotions with children using words and emojis. *Journal of Sensory Studies* 32, 3 (2017), e12264.
[19] Eric Goldman. 2018. Emojis and the Law. *Wash. L. Rev.* 93 (2018), 1227.
[20] Kevin Gray. 2015. *Argentina's 'Facebook president' announces his cabinet with emojis.* https://splinternews.com/argentinas-facebook-president-announces-his-cabinet-wit-1793853317
[21] Robert D. Hof. 2016. *Picture This: Marketers Let Emojis Do the Talking.* http://www.nytimes.com/2016/03/07/business/media/picture-this-marketers-let-emojis-do-the-talking.html?_r=1#
[22] Tianran Hu, Han Guo, Hao Sun, Thuy-vy Thi Nguyen, and Jiebo Luo. 2017. Spice up your chat: The intentions and sentiment effects of using emoji. *arXiv preprint arXiv:1703.02860* (2017).
[23] SwiftKey Inc. 2015. *SwiftKey Emoji Report 2015.* https://www.aargauerzeitung.ch/asset_document/i/129067827/download
[24] Sara R Jaeger and Gastón Ares. 2017. Dominant meanings of facial emoji: Insights from Chinese consumers and comparison with meanings from internet resources. *Food Quality and Preference* 62 (2017), 275–283.
[25] Sara R Jaeger, Soh Min Lee, Kwang-Ok Kim, Sok L Chheang, David Jin, and Gastón Ares. 2017. Measurement of product emotions using emoji surveys: Case studies with tasted foods and beverages. *Food quality and preference* 62 (2017), 46–59.
[26] Sara R Jaeger, Leticia Vidal, Karrie Kam, and Gastón Ares. 2017. Can emoji be used as a direct method to measure emotional associations to food names? Preliminary investigations with consumers in USA and China. *Food quality and preference* 56 (2017), 38–48.
[27] Ryan Kelly and Leon Watts. 2015. Characterising the inventive appropriation of emoji as relationally meaningful in mediated close personal relationships. *Experiences of Technology Appropriation: Unanticipated Users, Usage, Circumstances, and Design* (2015).
[28] Chi Hong Leung and Winslet Ting Yan Chan. 2017. Using emoji effectively in marketing: An empirical study. *Journal of Digital & Social Media Marketing* 5, 1 (2017), 76–95.
[29] Nikola Ljubešić and Darja Fišer. 2016. A global analysis of emoji usage. In *Proceedings of the 10th Web as Corpus Workshop.* 82–89.
[30] Peter Edberg Mark Davis. 2017. *Unicode Emoji.*
[31] Jeff Chertack Marla Shaivitz. 2017. *Proposal for Mosquito Emoji.* http://www.unicode.org/L2/L2017/17268-mosquito-emoji.pdf
[32] Stanley Mathews and Seung-eun Lee. 2018. Use of Emoji as a Marketing Tool: An Exploratory Content Analysis. *Fashion, Industry and Education* 16, 1 (2018), 46–55.
[33] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.
[34] Noa Na'aman, Hannah Provenza, and Orion Montoya. 2017. Varying linguistic purposes of emoji in (twitter) context. In *Proceedings of ACL 2017, Student Research Workshop.* 136–141.
[35] Christina Newberry. 2018. *28 Twitter Statistics All Marketers Need to Know in 2018.* https://blog.hootsuite.com/twitter-statistics/
[36] Sarah Perez. 2017. *WHEW, Google fixed the burger emoji in Android 8.1.* https://techcrunch.com/2017/11/28/whew-google-fixed-the-burger-emoji-in-android-8-1
[37] Henning Pohl, Christian Domin, and Michael Rohs. 2017. Beyond Just Text: Semantic Emoji Similarity Modeling to Support Expressive Communication. *ACM Transactions on Computer-Human Interaction (TOCHI)* 24, 1 (2017), 6.
[38] EyeSee Research. 2016. *Branded emojis increase adsâĂŹ attention by almost 10%!* http://eyesee-research.com/blog/branded-emojis-increase-ads-attention/
[39] World Population Review. 2018. *United States Population 2018.* http://worldpopulationreview.com/countries/united-states-population/
[40] Matthew Rothenberg. 2017. *Why the Emoji Recycling Symbol is Taking Over Twitter.* https://medium.com/@mroth/why-the-emoji-recycling-symbol-is-taking-over-twitter-65ad4b18b04b
[41] Diane J Skiba. 2016. Face with tears of joy is word of the year: are emoji a sign of things to come in health care? *Nursing education perspectives* 37, 1 (2016), 56–57.
[42] Mark Di Stefano. 2015. *Julie Bishop Describes Serious Diplomatic Relationships With Emoji.* https://www.buzzfeed.com/markdistefano/emoji-plomacy
[43] Channary Tauch and Eiman Kanjo. 2016. The roles of emojis in mobile phone notifications. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct.* ACM, 1560–1565.
[44] Emogi Research Team. 2015. *Emoji Report.*
[45] Emogi Research Team. 2016. *Emoji Report.*
[46] Twitter. 2018. *Introduction to Tweet JSON.* https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json
[47] Leticia Vidal, Gastón Ares, and Sara R Jaeger. 2016. Use of emoticon and emoji in tweets for food-related emotional expression. *Food Quality and Preference* 49 (2016), 119–128.
[48] Benjamin Weissman and Darren Tanner. 2018. A strong wink between verbal and emoji-based irony: How the brain processes ironic emojis during language comprehension. *PloS one* 13, 8 (2018), e0201727.
[49] Wikipedia. 2018. *Fitzpatrick scale.* https://en.wikipedia.org/wiki/Fitzpatrick_scale
[50] Sarah Wiseman and Sandy JJ Gould. 2018. Repurposing Emoji for Personalised Communication: Why pizza slice emoji means "I love you". In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems.* ACM, 152.
[51] Rui Zhou, Jasmine Hentschel, and Neha Kumar. 2017. Goodbye text, hello emoji: mobile communication on wechat in China. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems.* ACM, 748–759.
[52] Yonatan Zunger. 2017. *Animal Emoji Requests on Twitter (The world wants raccoons and lobsters).* https://medium.com/@yonatanzunger/animal-emoji-requests-on-twitter-d514d5c729ec

Online Learning for Measuring Incentive Compatibility in Ad Auctions*

Zhe Feng†
Harvard University
Cambridge, MA, USA
zhe_feng@g.harvard.edu

Okke Schrijvers
Facebook Research
Menlo Park, CA, USA
okke@fb.com

Eric Sodomka
Facebook Research
Menlo Park, CA, USA
sodomka@fb.com

ABSTRACT

In this paper we investigate the problem of measuring end-to-end Incentive Compatibility (IC) regret given black-box access to an auction mechanism. Our goal is to 1) compute an estimate for IC regret in an auction, 2) provide a measure of certainty around the estimate of IC regret, and 3) minimize the time it takes to arrive at an accurate estimate. We consider two main problems, with different informational assumptions: In the *advertiser problem* the goal is to measure IC regret for some known valuation v, while in the more general *demand-side platform (DSP) problem* we wish to determine the worst-case IC regret over all possible valuations. The problems are naturally phrased in an online learning model and we design Regret-UCB algorithms for both problems. We give an online learning algorithm where for the advertiser problem the error of determining IC shrinks as $O\left(\frac{|B|}{T}\cdot\left(\frac{\ln T}{n}+\sqrt{\frac{\ln T}{n}}\right)\right)$ (where B is the finite set of bids, T is the number of time steps, and n is number of auctions per time step), and for the DSP problem it shrinks as $O\left(\frac{|B|}{T}\cdot\left(\frac{|B|\ln T}{n}+\sqrt{\frac{|B|\ln T}{n}}\right)\right)$. For the DSP problem, we also consider stronger IC regret estimation and extend our Regret-UCB algorithm to achieve better IC regret error. We validate the theoretical results using simulations with Generalized Second Price (GSP) auctions, which are known to not be incentive compatible and thus have strictly positive IC regret.

CCS CONCEPTS

• **Theory of computation → Online learning theory**; **Algorithmic game theory**.

KEYWORDS

Ad Auctions; Incentive Compatibility; Multi-armed Bandits

ACM Reference Format:
Zhe Feng, Okke Schrijvers, and Eric Sodomka. 2019. Online Learning for Measuring Incentive Compatibility in Ad Auctions. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313674

*The full version of this paper, including the proof details, is available on ArXiv.
†This work was done while the first author was an intern in Core Data Science, Facebook Research.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313674

1 INTRODUCTION

Online advertising has grown into a massive industry, both in terms of ads volume as well as the complexity of the ecosystem. Most inventory is sold through auctions, and demand-side platforms (DSPs)—such as AppNexus, Quantcast, Google's DoubleClick and Facebook's Audience Network—offer advertisers the opportunity buy inventory from many different publishers. In turn, publishers themselves may run their own auctions to determine which ad to show and what price to charge, resulting in a complicated sequence of auction systems. While auction theory has influenced the design of online ad auctions, in practice not all auctions are incentive compatible. Practices such as sequential auctions (where the winners of an auction on a DSP compete in further auctions on the publisher website) and "dynamic reserve prices" (where publishers use a reserve price of e.g. 50% of the highest bid in the auction) both violate incentive compatibility, even when individual auctions are otherwise designed to be incentive compatible.

A lack of incentive compatibility in the ecosystem is problematic both for advertisers and DSPs. Advertisers have to spend time and energy in figuring out the optimal bidding strategy for different placements, a process which can be quite costly especially when incentives differ across different publishers. For DSPs this is problematic because common budget management techniques, such as multiplicative pacing, are only optimal when the individual auctions are incentive compatible [10].

Since (the lack of) incentive compatibility impacts bidding strategies for advertisers and the quality of the product that a DSP offers, it's important to understand the end-to-end incentive compatibility of an ad system. But how should we think of quantifying incentive compatibility? IC is a binary property: either a buyer maximizes their utility by bidding truthfully or she doesn't. However, this fails to capture that a first-price auction is "worse," in some sense, than a second-price auction with a dynamic reserve price of 50% of the highest bid. To capture these differences, we focus on dynamically measuring *IC regret*:

$$\text{IC regret}(v_i) = \max_{b_i} \mathbb{E}_{b_{-i}}\left[u_i(b_i, b_{-i}) - u_i(v_i, b_{-i})\right], \qquad (1)$$

where v_i is the true value of advertiser i, b_i the bid of i, b_{-i} the bids of other advertisers, and $u_i(\cdot)$ the (expected) utility of i. IC regret captures the difference in utility between bidding truthfully, and the maximum utility achievable. By definition, incentive compatible mechanisms have IC regret 0, while higher IC regret indicates a stronger incentive to misreport.

In this paper we focus on measuring IC regret assuming only black-box access to the auction mechanism. That is, for a given advertiser bid b_i we observe whether this yielded an impression x_i, and what price p_i was paid if any. While it seems restrictive to only focus on black box access, this setting allows one to deal with the fact that auction logic is spread out over different participants in the ecosystem (e.g. a DSP and publisher may each run an auction using proprietary code), and it even allows an advertiser who has no access to any auction code to verify the end-to-end incentives in the system.

Given black-box access to the auction mechanism, our goal is to 1) compute an estimate for IC regret in an auction, 2) provide a measure of certainty around the estimate of IC regret, and 3) minimize the time it takes to arrive at an accurate estimate.

We approach this problem using tools from the combinatorial (semi-)bandit literature: to measure the IC regret in an auction system, a bidder may enter different bids in different auctions[1]. By judiciously choosing these bids, we give bounds on the difference between the measured IC regret, and the true IC regret of the system as a function of the number of auctions the bidder participated in. This provides a trade-off between the accuracy of the IC regret measurement, and the the amount of time (and thus money) spend on obtaining the insight.

We consider the problem from two perspectives. The first, which we call the "Advertiser Problem," consists of determining the IC regret in the system for some *known* true value v. This is the problem the advertiser faces when they are participating in an online ad auction. In Section 3 we give an algorithm to select bids that determine IC regret with an error that shrinks as $O\left(\frac{|B|}{T}\cdot\left(\frac{\ln T}{n}+\sqrt{\frac{\ln T}{n}}\right)\right)$, where B is the (finite) bid space, n is the number of auctions at each time step and T is the total number of time steps. The second problem, which we call the "DSP Problem" consists of determining IC regret for the worst-case valuation v, i.e. the value v that yields the *highest* IC regret. This is the problem that a DSP faces if they want to ensure that IC regret is low for *all* advertisers that may use their services. In Section 4 we give an algorithm to select bids and (hypothetical) values that determine worst-case IC regret over all possible values where the error term shrinks as $O\left(\frac{|B|}{T}\cdot\left(\frac{|B|\ln T}{n}+\sqrt{\frac{|B|\ln T}{n}}\right)\right)$ as the number of time steps T grows.

1.1 Related Work

Structurally, designing incentive compatible (IC) mechanisms is well-understood: if the corresponding optimization problem can be solved exactly then the VCG mechanism [9, 13, 19] is IC. In cases where computing the optimal allocation is prohibitively expensive, and some approximation algorithm is used, VCG is no longer incentive compatible [17]. At the same time, the structure of IC mechanisms beyond optimal allocation is known, and tied to some monotonicity property of the allocation function for single-parameter settings [16] and more generally [18]. A weakness in these approaches is that determining if some given mechanism M is IC, one needs to know both the allocation *function* and the pricing *function*. Without access to the code that computes the allocation and pricing, the best an advertiser (or a DSP who depends on a third party auction that's run by a publisher) can hope to have access to is samples from the allocation and pricing functions, not the functions themselves.

Lahaie et al. [15] are the first to study the problem of testing incentive compatibility assuming only black box access. They design an A/B experiment to determine whether an auction is incentive compatible in both single-shot and dynamic settings (the latter concerns cases where an advertiser's bid is used to set their reserve price in later auctions). The work of Lahaie et al. [15] provides a valuable tool for validation of incentive compatibility, but leaves open the question of how to design an experiment that minimizes the time required to get results with high confidence. The present paper complements [15] by giving an algorithm to select alternative bids that minimize the error in the estimate for IC regret.

Our work is also related with *No regret learning in Game Theory and Mechanism Design* [6]. However, in Game Theory and Mechanism Design, most of the existing literature focuses on maximizing revenue for the auctioneer without knowing a priori the valuations of the bidders, e.g. [1, 2, 4], as well as optimizing the bidding strategy from the bidder side, e.g. [3, 11, 20]. To the best of our knowledge, this is the first paper on testing IC through online learning approach.

1.2 Contributions

There are three main contributions of this paper.

(1) We build an online learning model to measure IC regret with only black-box access to the auction mechanism. We present algorithms that find IC regret with an error that shrinks as $O\left(\frac{|B|}{T}\cdot\left(\frac{\ln T}{n}+\sqrt{\frac{\ln T}{n}}\right)\right)$ for a known valuation v, and an error that shrinks as $O\left(\frac{|B|}{T}\cdot\left(\frac{|B|\ln T}{n}+\sqrt{\frac{|B|\ln T}{n}}\right)\right)$ for all possible valuations v. In the above, B is the (finite) bid space, n is the number of auctions at each time step and T is the total number of time steps. We also extend our Regret-UCB algorithm to handle stronger IC regret estimation and shrink the error as $O\left(\frac{|B|^2(\ln T)^{2/3}}{n^{2/3}T}\right)$.

(2) We present a combinatorial semi-bandit algorithm for the setting where the "observed reward" is the max over several arms selected at each time while the benchmark is the standard optimal arm with highest expected reward. To understand this mismatch between the benchmark and observed reward, we analyze the *Pseudo-Regret* of the algorithm and uncover its trade-off with the number of arms selected at each time. This analysis may be of independent interest to the combinatorial (Semi-)Bandits literature.

(3) Simulations suggest that, (1) there is a trade-off between *Pseudo-Regret* and the number of blocks to partition auctions, i.e. we would like to choose the number of blocks to be neither very small nor very large, (2) the *Pseudo-Regret* (near linearly) decays when the number of auctions accessed at each time grows, and (3) our designed algorithm performs better than naive Random-Bids algorithm and ϵ-Greedy algorithm.

[1] In this paper, we consider the model that a bidder participates in many auctions and partitions them into several blocks. For every auction in each block, the bidder enters a bid and only observes the allocation and payment of the bid he enters from each block. For more formal definition, see Section 2.2.

2 MODEL AND PRELIMINARIES

In this section, we formally define the model we considered in this paper and present some important preliminaries. For simplifying description, we introduce "test bidder" to model both the advertiser and DSP.

2.1 Auctions for a Test Bidder

Consider a test bidder i who is eligible for n auctions every day. In each auction, they submit a bid b_i, the auction is run and the outcome is determined by the allocation rule $g_i : b_i \to [0, 1]$ and the payment rule $p_i : b \to \mathbb{R}_{\geq 0}$ (where g_i and p_i are conditioned on the competition in the auction). We assume a single-parameter quasilinear utility model $u_i(v_i, b_i) = g_i(b_i) \cdot v_i - p_i(b_i)$ for some true value $v_i \in \mathbb{R}_{\geq 0}$. Let U be a bound on the utility of a buyer, i.e. the utility function is bounded by $[-U, U]$.

Since there is a large number of auctions and other bidders in the ecosystem, we model the randomness in the system using a stationarity assumption where the conditional allocation and pricing functions are drawn from the same underlying distribution for each auction k:

Assumption 2.1 (Stochastic Assumption). *For each auction k, the allocation rule g_i and payment rule p_i of the test bidder i are drawn from an unknown distribution $\mathcal{F}$, i.e. $(g_i, p_i)_k \overset{i.i.d.}{\sim} \mathcal{F}$.*

2.2 Online Learning Model

At every time step $t \in [T]$, the test bidder participates n auctions and randomly partitions them into $m + 1$ blocks of equal size.[2] For $j \in [m + 1]$, let $\mathcal{A}_j$ be the set of auctions in block j.[3] For every auction in each block j, the test bidder submits a bid $b_t^j \in B$, where B is a finite set of bids.[4]

At the end of time t, in each block $j \in [m]$, the test bidder observes the average allocation probability $\widetilde{g}_t^j(b_t^j)$ and average payment $\widetilde{p}_t^j(b_t^j)$ over all auctions in block j. Let $\vec{b}_t = \{b_t^1, b_t^2, \cdots, b_t^m\}$ be the bids of the test bidder at time t. For block $m + 1$, the test bidder bids their true value v and observes the average allocation $\widetilde{g}_t^{m+1}(v)$ and payment $\widetilde{p}_t^{m+1}(v)$. Without loss of generality, we also assume $v \in B$. The average utility of the test bidder in each block $j \in [m + 1]$ is $\widetilde{u}_t^j(v, b_t^j) \equiv \widetilde{g}_t^j(b_t^j) \cdot v - \widetilde{p}_t^j(b_t^j)$.

With the setting defined, we can instantiate an empirical version of the IC regret from Equation (1) for a given true value v and bids $\vec{b}_t$:

Definition 2.2 (empirical IC regret).

$$\widetilde{rgt}_t(v, \vec{b}_t) = \max_{j \in [m]} \left\{ \widetilde{u}_t^j(v, b_t^j) - \widetilde{u}_t^{m+1}(v, v) \right\}. \tag{2}$$

For notational simplicity, we restate $g^* \triangleq \mathbb{E}[\widetilde{g}_t^j]$ and $p^* \triangleq \mathbb{E}[\widetilde{p}_t^j]$ for each block j at time t.[5] Similarly, define $u^*(v, b) \triangleq \mathbb{E}[\widetilde{u}_t^j(v, b)] \equiv g^*(b) \cdot v - p^*(b)$. Given these definitions, *IC-regret* for a particular true value v and bid b is:

Definition 2.3 (IC-regret). $rgt(v, b) = u^*(v, b) - u^*(v, v)$.

Thus far, in Definitions 2.2 and 2.3 we've considered the regret that a buyer has for bidding their true value v compared to a particular alternative bid b. The quantity that we're really interested in (cf. Equation (1)) is the worst-case IC regret with respect to different possible bids. For the Advertiser Problem (which we treat in Section 3) this is with respect to some known true value v, whereas for the DSP Problem (which we treat in Section 4) we consider the worst-case IC regret over all possible true values v. To summarize, the learning task of the test bidder is to design an efficient learning algorithm to generate v_t and b_t in order to minimize the following *Pseudo-Regret*, the difference between cumulative empirical regret and benchmark.[6]

Definition 2.4 (Pseudo-Regret [5]).

$$\mathbb{E}[R(T)] = \max_{v, b \in B} \sum_{t=1}^{T} rgt(v, b) - \mathbb{E}\left[\sum_{t=1}^{T} \widetilde{rgt}_t(v_t, \vec{b}_t) \right] \tag{3}$$

Given the above *Pseudo-Regret* definition, we can define the error of determining IC regret, which is to measure the distance between the optimal IC regret and the average empirical IC regret over time.

Definition 2.5 (IC Regret Error). $\mathcal{E}(T) = \frac{\mathbb{E}[R(T)]}{T}$

2.3 More Related work

The online learning model in our work is also related to Combinatorial (Semi)-bandits literature [8, 12, 14]. Chen et al. [8] first proposed the general framework and Combinatorial Multi-Armed Bandit problem, and our model lies in this general framework (i.e. the bids are generated from a super arm at each time step). However, in our model the test bidder can observe feedback from multiple blocks at each time step, which is similar to the combinatorial semi-bandits feedback model. Gai et al. [12] analyze the combinatorial semi-bandits problem with linear rewards, and Kveton et al. [14] provides the tight regret bound for the same setting. The Regret-UCB algorithm in our paper is similar and inspired by the algorithm in [12, 14]. However, in our model, the reward function is not linear and has a "max", which needs more work to address it. Chen et al. [7] firstly consider the general reward function in Combinatorial Bandits problem and can handle "max" in reward function, however, there is a mismatch between the benchmark and observed reward in our model (see more in "Contribution").

3 THE ADVERTISER PROBLEM

In this section, we focus on the special case of measuring the IC regret for a known true value v. The learning problem in this setting is to select, for each timestep t, a bid profile $\vec{b}_t$ of m bids to be used in the m auction blocks at that time. The bids $\vec{b}_t$ should be selected to minimize the *Pseudo-Regret* defined in Equation 3.

We propose the Regret-UCB algorithm, given in Algorithm 1, which is inspired by the CombUCB1 algorithm [12, 14]. Regret-UCB works as follows: at each timestep t, the algorithm performs three operations. First, it computes the *upper confidence bound* (UCB) on

[2] The number of blocks m is a variable that will be fixed later. The choice of which m to use will trade off the number of different bids we get information on and the accuracy of the outputs for each bid. Throughout the paper, we assume $m < |B|$.

[3] We let $[n] = \{1, 2, 3, \ldots, n\}$ be the set of positive integers up to and including n.

[4] While in theory the bid space is continuous, in practice outcomes for different bids are relatively smooth, see e.g. the plots in [15], so discretizing the bid space is a reasonable simplification.

[5] Throughout this paper, the expectation $\mathbb{E}$ is over all the randomness (e.g., randomization over payment and allocation distribution and randomization over algorithm).

[6] Where in the case of the Advertiser Problem, we define Pseudo-Regret with respect to a given value v analogously.

the expected utility of each bid $b \in B$ given v,

$$\text{UCB}_t^{\text{u}}(v,b) = \widehat{g}_{t-1}(b)\cdot v - \widehat{p}_{t-1}(b) + 2U\sqrt{\frac{2(m+1)\ln t}{n_{t-1}(b)\cdot n}}, \quad (4)$$

where $\widehat{g}_s(b)$ is the average allocation probability of bid b up to time s, $n_s(b)$ is the number of times that bid b has been submitted in s steps and U is the bound of utility (i.e. utility is bounded by $[-U, U]$). Second, the algorithm chooses the m bids that correspond to the largest UCB_t^{u} values; call this bid vector $\vec{b}_t$. The algorithm uses the m bids $\vec{b}_t$ in the first m blocks of auctions, and in the final block of auctions, it uses the bidder's true value v. Finally, for each of the blocks, the algorithm observes the average allocation probability and payment, and it uses that to update the estimates of allocation $\widehat{g}$ and payment $\widehat{p}$.

Algorithm 1 Regret-UCB Algorithm for a known valuation v.

Input: A finite set of bids B, parameter m, n. $\forall b \in B, n_0(b) = 1$
Initialize: Run Init(B, m, n) algorithm to get $\widehat{g}_0$ and $\widehat{p}_0$.
for $t = 1, \cdots, T$ **do**
  Update UCB^u terms (Eq. (4)) of each bid b.
  Generate a sequence of different m bids $\vec{b}_t \in B^m$ to maximize

$$\sum_{b\in\vec{b}_t} \text{UCB}_t^{\text{u}}(v,b) \quad (5)$$

  for $j = 1, \cdots, m$ **do**
    $n_t(b_t^j) \leftarrow n_{t-1}(b_t^j) + 1$
    Observe $\tilde{g}_t^j(b_t^j)$ and $\tilde{p}_t^j(b_t^j)$
    $\widehat{g}_t(b_t^j) \leftarrow \left[\widehat{g}_{t-1}(b_t^j)\cdot n_{t-1}(b_t^j) + \tilde{g}_t^j(b_t^j)\right] \big/ n_t(b_t^j)$
    $\widehat{p}_t(b_t^j) \leftarrow \left[\widehat{p}_{t-1}(b_t^j)\cdot n_{t-1}(b_t^j) + \tilde{p}_t^j(b_t^j)\right] \big/ n_t(b_t^j)$
  for $b \notin \vec{b}_t$ **do**
    $n_t(b) \leftarrow n_{t-1}(b), \widehat{g}_t(b) \leftarrow \widehat{g}_{t-1}(b), \widehat{p}_t(b) \leftarrow \widehat{p}_{t-1}(b)$

Algorithm 2 Init algorithm to get $\widehat{g}_0$ and $\widehat{p}_0$.

Input: A finite set of bids B, parameter m and n
for b in B **do**
  Randomly participate in $\frac{n}{m+1}$ auctions and submit bid b for each auction.
  Observe the average allocation $\widetilde{g}(b)$ and payment $\widetilde{p}(b)$.
  $\widehat{g}_0(b) \leftarrow \widetilde{g}(b), \widehat{p}_0(b) \leftarrow \widetilde{p}(b)$.

Given a fixed valuation v, define b^* to be the best-response bid:

$$b^* \triangleq \arg\max_{b\in B} rgt(v,b) = \arg\max_{b\in B} u^*(v,b).$$

and denote $\Delta(b) \triangleq u^*(v,b^*) - u^*(v,b)$. The following theorem bounds the worst case *Pseudo-Regret* of the Regret-UCB algorithm for known valuation v.

Theorem 3.1. *Regret-UCB achieves pseudo-regret at most*

$$\sum_{b\in B: u^*(v,b)<u^*(v,b^*)} \frac{32(m+1)U^2\ln T}{n\Delta(b)} + \frac{\pi^2}{3}\cdot\frac{\Delta(b)}{m}$$

To prove the theorem, we rely on the following lemma, which is widely used in stochastic bandit literature. We refer the reader to check [5] for more material.

Lemma 3.2 ([5]). *Fix a valuation v and an iteration $t-1$ (where $t \geq 2$). If a bid $b \neq b^*$ (i.e. $u^*(v,b) < u^*(v,b^*)$) has been observed $n_{t-1}(b) \geq \frac{8(m+1)(2U)^2\ln t}{n\Delta(b)^2}$ times, then with probability of at least $1-\frac{2}{t^2}$, $\text{UCB}_t^u(v,b) \leq \text{UCB}_t^u(v,b^*)$.*

3.1 Discussion

The Regret-UCB algorithm can also be used to implement a low-regret bidding agent: Consider an advertiser who knows the valuation v and wants to maximize the expected utility $u^*(v,b)$ by seeking for a best response bid b. Indeed, the advertiser can adopt the exact same algorithm – Regret-UCB to maximize the utility. The analysis for regret bound in Theorem 3.1 also works when we change the reward function from IC regret to utility.

The term m appears in both terms in Theorem 3.1, hence we can pick it to minimize the asymptotic *Pseudo-Regret*:

Corollary 3.3. *Let $\overline{\Delta} \triangleq \max_{b\in B}\Delta(b)$ and $\underline{\Delta} \triangleq \min_{b\in B}\Delta(b)$. Choosing $m = \frac{\pi}{4U}\sqrt{\frac{n\overline{\Delta}\underline{\Delta}}{6\ln T}}$, the error of determining IC regret achieved by Regret-UCB algorithm for known valuation setting is upper bounded by*

$$\mathcal{E}(T) \leq O\left(\frac{|B|}{T}\cdot\left(\frac{\ln T}{n} + \sqrt{\frac{\ln T}{n}}\right)\right).$$

The big-Oh notation assumes $\overline{\Delta}$ and $\underline{\Delta}$ to be constants.

4 THE DSP PROBLEM

In this section, we consider the problem of determining the worst-case IC regret over all possible valuations v. Let v^* and b^* be the value and bid combination that yields the highest *IC regret*,[7] i.e. $(v^*, b^*) = \arg\max_{v,b\in B} rgt(v,b)$, We modify our previous Regret-UCB algorithm in this setting and show the pseudocode in Algorithm 3.

At each time t, the algorithm first computes the UCBs on the expected *IC regret* of each valuation and bid pair (v,b).

$$\text{UCB}_t^{\text{rgt}}(v,b) = \widehat{rgt}_t(v,b) + 4U\sqrt{\frac{3(m+1)\ln t}{n\cdot\left(n_{t-1}(v)\wedge n_{t-1}(b)\right)}} \quad (6)$$

where $\widehat{rgt}_t(v,b) = \widehat{g}_t(b)\cdot v - \widehat{p}_t(b) - \left(\widehat{g}_t(v)\cdot v - \widehat{p}_t(v)\right)$, "$\wedge$" is the min function, and the other notations $\widehat{g}_t$, $\widehat{p}_t$, $n_{t-1}(b)$, $n_{t-1}(v)$, and U are identical to Section 3. Then the algorithm selects (v_t, b_t^1) to maximize the $\text{UCB}_t^{\text{rgt}}$ term. Given the valuation v_t, the algorithm chooses other $m-1$ bids to achieve the $(m-1)-$largest $\text{UCB}_t^{\text{u}}(v_t,\cdot)$ terms defined in Equation (4). The rest of update steps in the algorithm are exactly the same as in Algorithm 1.

We denote $\Delta(v,b) \triangleq rgt(v^*,b^*) - rgt(v,b)$ and start with the following lemma that gives the concentration property of the UCB^{rgt} terms.

[7] We assume, without loss of generality, that this optimal combination is unique throughout the paper.
[8] We choose the remaining $m-1$ bids be different with each other and b_t^1.

Algorithm 3 REGRET-UCB Algorithm for unknown valuation.

Input: A finite set of bids B, parameter m, n. $\forall b \in B, n_0(b) = 1$
Initialize: Run INIT(B, m, n) algorithm to get $\widehat{g}_0$ and $\widehat{p}_0$.
for $t = 1, \cdots, T$ **do**
  Update UCB^{rgt} terms (Eq. (6)) of every (v, b) pair
  Choose $(v_t, b_t^1) \in B \times B$ to maximize $\text{UCB}_t^{\text{rgt}}(v, b)$
  if $m \geq 2$ **then**
    Choose remaining $m-1$ bids $\{b_t^2, \cdots, b_t^m\}$ to maximize[8]

$$\sum_{b \in \vec{b}_t \setminus b_t^1} \text{UCB}^{\text{u}}(v_t, b)$$

  for $b \in \vec{b}_t \cup v_t$ **do**
    $n_t(b) \leftarrow n_{t-1}(b) + 1$
    Observe $\tilde{g}_t(b)$ and $\tilde{p}_t(b)$ (omit subscript j for each block),

$$\widehat{g}_t(b) \leftarrow \left[\widehat{g}_{t-1}(b) \cdot n_{t-1}(b) + \widetilde{g}_t^j(b)\right] \Big/ n_t(b)$$

$$\widehat{p}_t(b) \leftarrow \left[\widehat{p}_{t-1}(b) \cdot n_{t-1}(b) + \widetilde{p}_t^j(b)\right] \Big/ n_t(b)$$

  for $b \notin \vec{b}_t$ **do**
    $n_t(b) \leftarrow n_{t-1}(b)$

LEMMA 4.1. *At iteration $t-1, (t \geq 2)$, for a (value, bid) pair $(v, b) \neq (v^*, b^*)$, where v and b are both observed at least $\frac{48(m+1)(2U)^2 \ln t}{n\Delta(v,b)^2}$ times, then with probability at least $1 - \frac{4}{t^2}$, $\text{UCB}_t^{rgt}(v, b) \leq \text{UCB}_t^{rgt}(v^*, b^*)$.*

Utilizing Lemma 4.1, we show the worst case *Pseudo-Regret* bound for IC-testing in non-fixed valuation setting in Theorem 4.2.

THEOREM 4.2. *REGRET-UCB algorithm for unknown valuation setting (DSP problem) achieves pseudo-regret at most*

$$\sum_{\substack{v,b \in B, \\ (v,b) \neq (v^*, b^*)}} \frac{384(m+1)U^2 \ln T}{n\Delta(v,b)} + \frac{2\pi^2 \Delta(v,b)}{3} \cdot \left(\mathbb{1}\left\{v \neq v^*\right\} + \frac{\mathbb{1}\left\{v = v^*\right\}}{m} \right)$$

Following the same argument in Section 3, if we choose m appropriately, we will get the following asymptotic bound.

COROLLARY 4.3. *Let $\overline{\Delta} \triangleq \max_{v,b \in B} \Delta(v, b)$ and $\underline{\Delta} \triangleq \min_{v,b \in B} \Delta(v, b)$. Choosing $m = \frac{\pi}{24U} \sqrt{\frac{n\overline{\Delta}\underline{\Delta}}{|B| \ln T}}$, the asymptotic error of determining IC regret achieved by REGRET-UCB algorithm is upper bounded by*

$$\mathcal{E}(T) \leq O\left(\frac{|B|}{T} \cdot \left(\sqrt{\frac{|B| \ln T}{n}} + \frac{|B| \ln T}{n} \right) \right).$$

The big-Oh notation assumes $\overline{\Delta}$ and $\underline{\Delta}$ to be constants.

Note in the DSP problem, since we consider determining the worst-case IC regret over all possible valuations v, our asymptotic error bound quadratically grows as $|B|$, while for the advertiser problem the error linearly grows as $|B|$ since the benchmark we consider there is weaker than it in the DSP problem.

5 EXTENSION: STRONGER IC REGRET

In Section 4, we assume the test bidder (DSP) only chooses one valuation v_t at every time step t to empirically estimate IC regret. However, in practice, for each (value, bid) pair, the test bidder can switch the roles of bid and value to better estimate the IC regret.

Since there is no conceptual difference between bid and value in this setting, at time step t, the test bidder also submits a "bid" b_t^{m+1} for every auction in block $m+1$. We still denote the bids submitted at time t be $\vec{b}_t = \left\{b_t^1, b_t^2, \cdots, b_t^{m+1}\right\}$. The only difference between this case and our standard model is from the definition of *empirical IC regret*, we define the modified *empirical IC regret* in this setting as below,

Definition 5.1 (modified empirical IC regret).

$$\widetilde{rgt}_t(\vec{b}_t) = \max_{i,j \in [m+1], i \neq j} \left\{ \widetilde{u}_t^j(b_t^i, b_t^j) - \widetilde{u}_t^i(b_t^i, b_t^i) \right\}$$

The learning task in this setting is to design an efficient algorithm to generate $m+1$ bids at every time step to minimize *Pseudo-Regret*,

$$\mathbb{E}\left[R(T)\right] \triangleq \max_{v,b \in B} \sum_{t=1}^{T} rgt(v, b) - \mathbb{E}\left[\sum_{t=1}^{T} \widetilde{rgt}_t(\vec{b}_t) \right]$$

We extend our REGRET-UCB algorithm for unknown valuation (Algorithm 3) for this setting.

At each time step t, we update the UCB^{rgt} terms defined in Equation (6) for every (b_1, b_2) bids pair. Given these UCB^{rgt} terms, we run GENERATE-BIDS algorithm proposed in Algorithm 4 to generate $m+1$ bids and update $\hat{p}_t(b)$ and $\hat{g}_t(b)$ for any chosen b at time t.

Algorithm 4 GENERATE-BIDS Algorithm for generating bids given UCB^{rgt} terms.

Input: A finite set of bids B, UCB^{rgt} terms for every (b_1, b_2) bids pair, where $b_1, b_2 \in B$. Initialize a set $B_0 = \{\}$.
while $|B_0| < m+1$ **do**
  Choose the largest $\text{UCB}^{\text{rgt}}(b_1, b_2)$ such at $b_1 \notin B_0$ or $b_2 \notin B_0$.
  Update $B_0 \leftarrow B_0 \cup \{b_1, b_2\}$. [9]

We call the above algorithm for stronger IC regret estimation REGRET-UCB*, of which the *Pseudo-Regret* is shown in Theorem 5.2.

THEOREM 5.2. *REGRET-UCB* algorithm for DSP problem using stronger IC regret estimation achieves pseudo-regret at most*

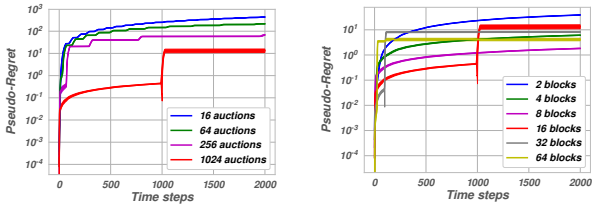
$$\sum_{\substack{b_1, b_2 \in B, \\ (b_1, b_2) \neq (v^*, b^*)}} \frac{192(m+1)U^2 \ln T}{n\Delta(b_1, b_2)} + \frac{2\pi^2 \Delta(b_1, b_2)}{3(m+1)^2}$$

Note the *Pseudo-Regret* bound achieved by REGRET-UCB* algorithm is always better than it achieved by Algorithm 3 for any m. Then we can always get a better asymptotic error bound by REGRET-UCB* algorithm for the DSP problem.

COROLLARY 5.3. *Choosing $m = O\left(\frac{n}{\ln T}\right)^{1/3}$ the asymptotic error of determining IC regret achieved by REGRET-UCB algorithm is upper bounded by*[10]

$$\mathcal{E}(T) \leq O\left(\frac{|B|^2 (\ln T)^{2/3}}{n^{2/3} T} \right)$$

[9] If the two bids chosen in the last round are both not in B_0, we randomly choose one to add it in B_0.
[10] This asymptotic error bound can be achieved if the optimal m can be chosen to satisfy the constraint that $m < |B|$.



(a) (b)

Figure 1: *Pseudo-Regret* of Regret-UCB algorithm for known valuation $v = 9.5$ (Algorithm 1).

6 SIMULATIONS

In this section, we run some simulations with Generalized Second Price (GSP) auctions to validate our theoretical results. Every experiment is repeated for 10 times and the dash area represents the 95% confidence intervals of *Pseudo-Regret* at each time step.

6.1 Settings

At each time t, the test bidder randomly participates n GSP auctions. Each auction has 5 slots and 20 bidders, each bidder's bid is i.i.d generated from $U[0, 10]$. The *click-through-rate* sequence of each auction is generated from descending ordered $Beta(2, 5)$ distribution[11]. We consider a finite bids space $B \triangleq \{0.01, 0.02, \cdots, 10\}$.

6.2 The advertiser problem

We test the performance of Regret-UCB algorithm for known valuation $v = 9.5$. First, we fix the number of blocks be 16 (i.e. $m = 15$) and plot the *Pseudo-Regret* achieved by Regret-UCB with different $n = 16, 64, 256, 1024$ (Figure 1(a)). We observe that the *Pseudo-Regret* decays linearly with n which is consistent with our analysis. Second, we fix $n = 1024$ and test the performance of the algorithm with different number of blocks, such as $m = 1, 3, 7, 15, 31, 63$ (Figure 1(a)). We find $m = 7$ (8 blocks) achieves the lowest *Pseudo-Regret*. When m is too large, the *Pseudo-Regret* curve incurs some shocks and suffers high variance because of the noisy observed information at each time.

6.3 The DSP problem

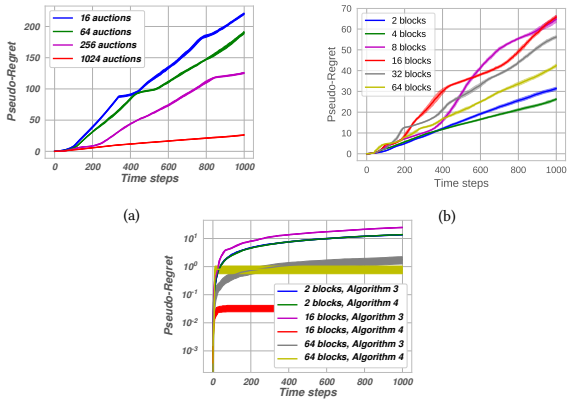
We test the performance of Regret-UCB for unknown valuation case (i.e., the DSP problem). Similarly, we show the *Pseudo-Regret* curve for different n given $m = 15$ and different m given $n = 1024$ in Figure 2. Figure 2(a) validates that *Pseudo-Regret* decays when n grows and we observe $m = 3$ (4 blocks) gives the best *Pseudo-Regret*. This corresponds to our theory that the optimal m in the DSP problem should be smaller than the optimal m in the advertiser problem for the same auction setting when $|B|$ is large (see Corollary (3.3) and (4.3)).

6.4 Efficiency of Regret-UCB algorithm

We first introduce two standard baselines, Random-Bids algorithm and ϵ-Greedy algorithm.

Random-Bids. At each time step t, the advertiser uniformly randomly choose m bids b_t, while the DSP uniformly randomly choose m bids b_t and a valuation v_t from B.

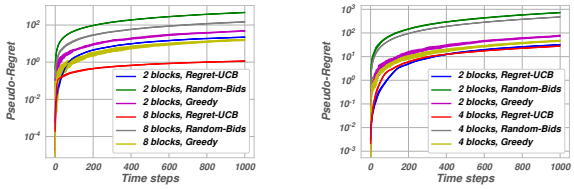
[11] *Click-through-rate* of each slot is first i.i.d generated from $Beta(2, 5)$ and then arranged by descending order.



(a) (b)

(c)

Figure 2: *Pseudo-Regret* plot of Regret-UCB algorithm for unknown valuation. (a): Algorithm 3 for different n, (b): Algorithm 3 for different m and (c): a semi-logarithmic *Pseudo-Regret* to compare Algorithm 3 and Regret-UCB*.



(a) (b)

Figure 3: *Pseudo-Regret* with Random-Bids, ϵ-Greedy and Regret-UCB for the advertiser problem (a) and DSP problem (b).

ϵ-Greedy. For the advertiser problem with known valuation v, at each time step, the advertiser uniformly randomly chooses m bids from B with probability ϵ, otherwise, chooses the m bids that correspond to the largest average utility $\hat{u}(v, \cdot)$ terms.[12] For the DSP problem, at each time step t, the DSP uniformly randomly choose m bids b_t and a valuation v_t with probability ϵ. Otherwise, the DSP chooses the (v_t, b_t^1) pair corresponds to the largest $\widehat{rgt}$ terms and the rest $m - 1$ bids associated with the largest $\hat{u}(v_t, \cdot)$ terms.

We compare the performance of our Regret-UCB algorithm with the above two baselines for the advertiser problem and the DSP problem. For the both settings, Regret-UCB algorithm performs better than two baselines for different number of blocks.

6.5 Regret-UCB for stronger IC regret

Moreover, we compare the performance of Algorithm 3 and Regret-UCB* algorithm for different m given $n = 1024$ in Figure 2(c) to validate the theory that allowing switching value and bid leads to lower Pseudo-Regret.[13] In addition, we also observe the tradeoff between Pseudo-Regret and m of Regret-UCB* algorithm. Among $m = 1, 15, 63$, the optimal m is 15 (16 blocks) for Regret-UCB*

[12] In the experiments, we fix $\epsilon = 0.1$

[13] Based on our observation in the experiments, sometimes switching value and bid may "over-estimate" the IC regret, i.e. the empirical estimation of IC regret is sometimes larger than worst case IC regret.

algorithm. If we choose m too large, like $m = 63$, Pseudo-Regret is worse than the $m = 1$ case and incurs high variance.

REFERENCES

[1] Kareem Amin, Rachel Cummings, Lili Dworkin, Michael Kearns, and Aaron Roth. 2015. Online Learning and Profit Maximization from Revealed Preferences. In *AAAI*. 770–776.

[2] Kareem Amin, Afshin Rostamizadeh, and Umar Syed. 2014. Repeated contextual auctions with strategic buyers. In *Advances in Neural Information Processing Systems*. 622–630.

[3] Santiago Balseiro and Yonatan Gur. 2017. Learning in Repeated Auctions with Budgets: Regret Minimization and Equilibrium. (2017).

[4] Avrim Blum, Vijay Kumar, Atri Rudra, and Felix Wu. 2004. Online learning in online auctions. *Theoretical Computer Science* 324, 2-3 (2004), 137–146.

[5] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning* 5, 1 (2012), 1–122.

[6] Shuchi Chawla, Jason D. Hartline, and Denis Nekipelov. 2014. Mechanism design for data science. In *ACM Conference on Economics and Computation, EC '14, Stanford , CA, USA, June 8-12, 2014*. 711–712.

[7] Wei Chen, Wei Hu, Fu Li, Jian Li, Yu Liu, and Pinyan Lu. 2016. Combinatorial multi-armed bandit with general reward functions. In *Advances in Neural Information Processing Systems*. 1659–1667.

[8] Wei Chen, Yajun Wang, and Yang Yuan. 2013. Combinatorial multi-armed bandit: General framework and applications. In *International Conference on Machine Learning*. 151–159.

[9] Edward H Clarke. 1971. Multipart pricing of public goods. *Public choice* 11, 1 (1971), 17–33.

[10] Vincent Conitzer, Christian Kroer, Eric Sodomka, and Nicolás E. Stier Moses. 2017. Multiplicative Pacing Equilibria in Auction Markets. *CoRR* abs/1706.07151 (2017). arXiv:1706.07151 http://arxiv.org/abs/1706.07151

[11] Zhe Feng, Chara Podimata, and Vasilis Syrgkanis. 2018. Learning to Bid Without Knowing Your Value. In *Proceedings of the 2018 ACM Conference on Economics and Computation (EC '18)*. ACM, New York, NY, USA, 505–522.

[12] Y. Gai, B. Krishnamachari, and R. Jain. 2012. Combinatorial Network Optimization With Unknown Variables: Multi-Armed Bandits With Linear Rewards and Individual Observations. *IEEE/ACM Transactions on Networking* 20, 5 (Oct 2012), 1466–1478.

[13] Theodore Groves. 1973. Incentives in teams. *Econometrica: Journal of the Econometric Society* (1973), 617–631.

[14] Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvari. 2015. Tight regret bounds for stochastic combinatorial semi-bandits. In *Artificial Intelligence and Statistics*. 535–543.

[15] Sébastien Lahaie, Andrés Munoz Medina, Balasubramanian Sivan, and Sergei Vassilvitskii. 2018. Testing Incentive Compatibility in Display Ad Auctions. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1419–1428.

[16] Roger B Myerson. 1981. Optimal auction design. *Mathematics of operations research* 6, 1 (1981), 58–73.

[17] Noam Nisan and Amir Ronen. 2007. Computationally feasible VCG mechanisms. *Journal of Artificial Intelligence Research* 29 (2007), 19–47.

[18] Jean-Charles Rochet. 1987. A necessary and sufficient condition for rationalizability in a quasi-linear context. *Journal of mathematical Economics* 16, 2 (1987), 191–200.

[19] William Vickrey. 1961. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance* 16, 1 (1961), 8–37.

[20] Jonathan Weed, Vianney Perchet, and Philippe Rigollet. 2016. Online learning in repeated auctions. In *Conference on Learning Theory*. 1562–1583.

TableNet: An Approach for Determining Fine-grained Relations for Wikipedia Tables

Besnik Fetahu
L3S Research Center, Leibniz University of Hannover, Hannover, Germany
fetahu@L3S.de

Avishek Anand
L3S Research Center, Leibniz University of Hannover, Hannover, Germany
anand@L3S.de

Maria Koutraki
L3S Research Center, Leibniz University of Hannover, Hannover, Germany
FIZ-Karlsruhe Leibniz Institute, Karlsruhe Institute of Technology, Karlsruhe, Germany
koutraki@L3S.de

ABSTRACT

We focus on the problem of interlinking Wikipedia tables with fine-grained table relations: `equivalent` and `subPartOf`. Such relations allow us to harness semantically related information by accessing related tables or facts therein. Determining the type of a relation is not trivial. Relations are dependent on the schemas, the cell-values, and the semantic overlap of the cell values in tables.

We propose *TableNet*, an approach for interlinking tables with `subPartOf` and `equivalent` relations. *TableNet* consists of two main steps: (i) for any *source* table we provide an *efficient* algorithm to find *candidate* related tables with *high coverage*, and (ii) a neural based approach that based on the table schemas and data, determines with *high accuracy* the fine-grained relation.

Based on an extensive evaluation with more than 3.2M tables, we show that TableNet retains more than 88% of relevant tables pairs, and assigns table relations with an accuracy of 90%.

ACM Reference Format:
Besnik Fetahu, Avishek Anand, and Maria Koutraki. 2019. TableNet: An Approach for Determining Fine-grained Relations for Wikipedia Tables. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313629

1 INTRODUCTION

One of the most notable uses of Wikipedia is on knowledge base construction like DBpedia [2] or YAGO [14], built almost exclusively built with information coming from Wikipedia's infoboxes. Infoboxes have several advantages as they adhere to pre-defined templates and contain factual information (e.g. *bornIn* facts). However, they are sparse and the information they cover is very narrow. In most use cases of Wikipedia, availability of factual information is a fundamental requirement.

Wikipedia tables on the other hand are in abundance, and are rich in factual information for a wide range of topics. Wikipedia contains more than 3.23M tables from more than 520k articles. Thus far, their use has been limited, despite them covering a broad domain of information that can be used to answer complex queries, e.g., *"Award winning movies of horror Genre?"* the answer can be found from facts in *multiple tables*. However, question answering systems [1] built upon knowledge base facts, in most cases will provide *incomplete* answers or fail altogether. The sparsity or lack of factual information from infoboxes can be easily remedied by additionally considering facts from tables.

A rough estimate shows that from the 3.23M tables we can generate hundreds of millions of additional facts that can be converted into knowledge base triples [10]. This amount in reality is much higher, when tables are aligned. That is, currently, tables are seen in isolation, and *semantically* related tables are not interlinked. Table alignments allow us to access related tables as *supersets or subsets* (i.e. `subPartOf` relations) or `equivalent` tables, which in turn we can use to infer new knowledge and facts that can be used in QA systems or other use cases.

Determining the fine-grained table relations is not a trivial task. The relations are dependent on the semantics of the columns (e.g. a column with values of type `Country`), the context in which the column appears (e.g. *"Name"* is ambiguous and it can only be disambiguated through other columns in a table schema), cell values etc. Furthermore, not all columns are important for determining the table relation [13]. Finally, to establish relations amongst all relevant table pairs, we need to ensure the efficiency by avoiding exhaustive computations between all table pairs that can be cumbersome given the extent of tables in Wikipedia.

We propose *TableNet*, an approach with the goal of aligning tables with `equivalent` and `subPartOf` fine-grained relations. Our goal is to ensure that for any table, we can find with high coverage candidate tables for alignment, and accurately determine the relation type for a given pair. We distinguish between two main steps: (i) efficient and high coverage table candidate generation for alignment, and (ii) relation type prediction by leveraging table schemas and values therein. An extensive evaluation of *TableNet* over the entire English Wikipedia with more than 3.2 million tables, shows that we are able to retain table pairs with a high coverage of 88%, and predict the fine-grained relation with an accuracy of 90%.

2 RELATED WORK

Google Fusion Tables [5, 6, 13, 15] are the most significant efforts in providing additional semantics over tables, and to the best of our knowledge, only some of the works carried in this project are most related to our work, against which we provide fair comparisons [13].

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313629

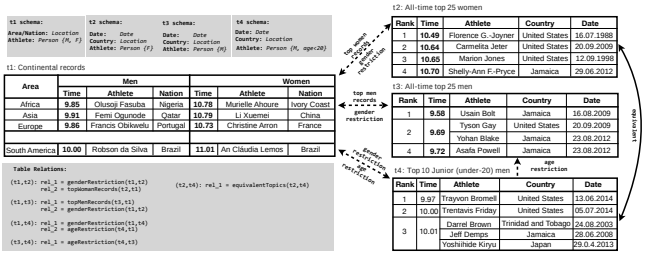


Figure 1: Table alignment example with `subPartOf` (dashed line), and `equivalent` relations (full line).

Carafella et al. [5] propose an approach for table extraction from Web pages and also provide a ranking mechanism for table retrieval. An additional aspect they consider is the schema auto-completion, specifically column recommendation for some input column to generate a "complete" schema. Our goals are different, while, we aim at providing more fine-grained representations of columns in a table schema, our goal is to use this information for table alignment.

Das Sarma et al. [13] propose an approach for finding related tables for two scenarios: (i) *entity complement* and (ii) *schema complement*. For (i), the task is to align tables with similar table schemas, however, with complementary instances. In (ii), the columns of a target table are used to complement the schema of a source table, with the precondition that the instances (from *subject* columns) are the same in both tables.

Our work is related to the first case. In the entity complement case [13] the *schema similarity* between two tables is computed in order to decide if a table can be considered for complementing the instances in another table. The similarity of the schemas is considered as a max-weight bipartite matching approach. The edges are weighted and are established between the columns in the disparate schemas. The weights correspond to the *string similarity* between column names and jaccard similarity between the column types (established from the values in a column through the `WebIsA` database). Despite the fact that this approach is unsupervised, we adapt it such that we find the best threshold of the max-weight matching, and consider tables to be either *aligned* or *not-aligned*.

3 PRELIMINARIES AND OVERVIEW

3.1 Terminology

We consider Wikipedia articles $A = \{a_1, \ldots, a_n\}$; each article a is associated with a set of Wikipedia categories $\Psi_a = \{\psi_1, \ldots, \psi_n\}$. From all the categories we induce the category graph Ψ, which consists of *parent* and *child* relations between categories ψ_i `childOf` ψ^p. The parent/child relations allow us to establish a hierarchical graph in Ψ. The level of a category is denoted by λ_ψ.

Next, we define the tables from an article a as $T_a = \{t_1, \ldots, t_n\}$. A table t has a *table schema* consisting of columns $C(t) = \{c_1, \ldots c_n\}$, where each column consists of a *textual description* and the set of all values $c_i = \langle desc, \{v_i^1, \ldots, v_i^n\}\rangle$ assigned to the corresponding column cells in the table rows $t_i(r) = \{r_i^1, \ldots, r_i^n\}$. More specifically, the cell value is indicated by v_i^k, where k is the row r^k and i is the column c_i. Cell values can point to existing articles in Wikipedia, that is $v_i^k = \langle a_k \rangle$, which we will refer to as *instance values*, or *primitive values* in cases of text, numbers, dates etc.

For the tables $T = \{t_1, \ldots, t_n\}$ from A, we define two fine-grained types of relations for a table pair $\langle t_i, t_j \rangle$: (i) $t_i \vDash t_j$ where t_j is considered to be *semantically* a `subPartOf` of t_i, and (ii) $t_i \equiv t_j$ where t_i and t_j are *semantically* `equivalent`. We indicate a relation with $r(t_i, t_j) \neq \emptyset$, and define in the next section the table relations.

3.2 Table Alignment Task Definition

Table Alignment. Our task is to determine the *fine-grained relation* for a table pair $r(t_i, t_j)$ from the articles $\langle a_i, a_j \rangle$. The relations can be either `subPartOf`, `equivalent` or `none` (in case $r(t_i, t_j) = \emptyset$).

Definition 1 (subPartOf). *For $r(t_i, t_j) = \{$`subPartOf`$\}$ holds if the schema $C(t_i)$ can subsume either at the **data value** (i.e. **cell-value**) or **semantically** the columns from $C(t_j)$, i.e., $C(t_i) \supseteq C(t_j)$.*

Definition 2 (equivalent). *For a pair $r(t_i, t_j) = \{$`equivalent`$\}$ holds if both table schemas have **semantically similar** column representation, that is, $C(t_i) \approx C(t_j)$.*

3.3 TableNet Overview

In TableNet, for any *source* article $a_i \in A$, we first generate candidate pairs $\langle a_i, a_j \rangle$, such that the likelihood of the corresponding tables to have a relation is high. We describe the two main steps: (1) article candidate pair generation, and (2) table alignment.

4 ARTICLE CANDIDATE PAIR GENERATION

Table 1: Article candidate pair features.

	feature	*description*	*group*
f_1	*tfidf*	*tfidf* similarity between abstracts	*abstract*
f_2	*d2v*	*doc2vec* similarity between abstracts	
f_3	*w2v*	*avg.* word2vec abstract vectors similarity	
f_4	$sim(\Psi_{a_i}, \Psi_{a_j})$	similarity in embedding space between Ψ_a and Ψ_a^p categories for the article pair	*Ψ & KB*
f_5	$\bigcap_{a \in \langle a_i, a_j \rangle} \Psi_a$	direct and parent categories overlap	
f_6	$sim(a_i, a_j)$	embedding similarity of the article pair	
f_7	*type*	type overlap	
f_8	$sim(\psi_i, \psi_j)$	column title similarity (f_8^l) and column distance (f_8^d) between the schemas in a table pair	*tables*
f_9	$\lVert\gamma(\psi_i) - \gamma(\psi_j)\rVert$	category representation similarity γ	

In this step, we address the problem of determining article pairs, whose tables will result in a relation. Here, we are interested in two main properties: (1) minimize the amount of *irrelevant article pairs*, whose tables do not have a relation, and (2) filtering out of article pairs should not affect the coverage of relevant pairs (i.e. whose tables in a table relation.

With these properties in mind, we define features and use them in two ways: (i) filter out irrelevant article pairs, and (ii) employ the features in a supervised manner to further filter out such pairs.

4.1 Features

Table 1 shows all the features. We omit the description of features that are obvious from the table.

Article Abstract. Abstracts contain a summary of the most important information of an article. This feature ensures *topical/semantic* similarities between a pair of articles, through contextual similarities like *doc2vec* [9], word based similarity in the embedding space [11], or through *tf-idf* which captures the salience of tokens.

Categories & Knowledge Bases. Wikipedia categories are indicators that two articles are semantically similar. Categories are associated *manually* to articles, and thus are prone to noise. We circumvent this problem by computing *graph embeddings* based on the category graph Ψ, and use node2vec embedding [7] to elevate the category comparisons from the link based structure in Ψ into the embedding space. The features represent the cosine similarity in the embedding space for an article pair, specifically between the *directly associated* and *parent* categories, the articles themselves.

Tables. Article pair features capture only coarse grained similarity. Thus, we compute light-weight features between the columns of a table pair. The features correspond to the maximal similarity w.r.t the *average word embedding* of the column descriptions, and additionally we compute the *positional index difference* between the matching columns in the respective tables.

Column-Representation. For a column c_i with instance-values, we compute a representation based on the attributes associated with the instances $\{v_i\}$ in a target KB, e.g. v_i^1 = *"George Lucas'* ***bornIn*** *"Modesto, California, U.S."*. Since usually there are multiple instances $|v_i| > 1$ for c_i, we find the *lowest common ancestor* ψ_L category from $\{v_i\}$ in the category graph Ψ, which can be seen as a the common *type* of instances in $\{v_i\}$. The representations allow us to compute the semantic similarity between columns.

The representation of c_i is computed as in Equation 1. We weigh the importance of *attributes* based on how discriminative they are for ψ_L, e.g. an attribute associated with articles directly belonging to category ψ_L are exclusive for ψ_L, and thus are weighted high. For an attribute p, the weight for ψ_L is computed as following:

$$\gamma(p, \psi_L) = \frac{\lambda_{\psi_L}}{\max \lambda_{\psi}} * \left(-\log \frac{|\bigcup o| : \forall \langle a, p, o \rangle \wedge a \in \psi_L}{|o| : \forall \langle a, p, o \rangle \wedge a \in \psi_L} \right) \quad (1)$$

where, the first part of the fraction weighs p by taking into account the level of λ_{ψ_L} and the deepest category where p is present in a target KB. $|\bigcup o|$ represents the number of distinct values assigned to attribute p in ψ_L, whereas $|o|$ is the total number of assignments.

Through $\gamma(\psi_L)$ we capture the most *important* and *descriptive* attributes for a column. For columns whose representations yield a high cosine similarity is an indicator of high semantic similarity.

4.2 Filtering & Classification

We use the computed features in two ways: (i) filter out irrelevant article pairs (i.e. *unlikely* to have a table relation), and (ii) train a supervised model to classify article pairs as *relevant* or *irrelevant*.

Filtering. We consider a *conjunction* of filtering criteria based on empirically evaluated thresholds for the individual features. Our main goal is to retain a high coverage of *relevant article pairs*, and at the same time filter out drastically *irrelevant pairs*. For thresholds we consider the *mean* value of a particular feature. This is an indicator that the pair is unlikely to yield any table relation based on a given feature. Section 7 shows that we are able to drastically reduce the number of pairs by simply applying such thresholds.

Classification. From the remaining pairs we train a classification model and classify pairs as either *relevant* or *irrelevant*. We consider as positive instances all table pairs from the article pair $\langle a_i, a_j \rangle$ which have *at least one* table relation, i.e, $r(t_i, t_j) \neq \texttt{none}$, where $\exists(t_i \in a_i \wedge t_j \in a_j)$.

We use Random Forests (RF) [4] for classification, as they allow to set *minimal amount of samples* that are allowed for a node in the tree to be split. This directly impacts the accuracy of the classifier, however, it allows us to retain high coverage. Setting this number high makes the leafs of the different trees to be impure containing relevant and irrelevant article pairs. Our classifier will classify such impure leafs as relevant. Section 7 shows that we can maintain high coverage of relevant pairs and at the same drastically reduce the amount of irrelevant pairs.

5 TABLE ALIGNMENT

TableNet is a bidirectional recurrent neural network (RNN) with LSTM cells, which classifies a table pair $r(t_i, t_j)$ into one of the relations `equivalent`, `subPartOf`, or `none`. For a model to accurately align table, the *order* of columns in their schemas needs to be taken into account. Additionally, the matching columns in the two schemas need to fulfill two main criteria: (i) the *context* in which the columns occur needs to be semantically similar, and (ii) the *positions* of the columns needs to be comparably similar [13].

5.1 Table Representation

How we represent columns is key towards an accurate alignment model. A column in a table schema consists of the following information $c_i = \langle desc, \{v_i^1, \ldots v_1^n\} \rangle$ (see Section 3).

Column Description. The column description is a strong indicator of the cell-values v_i. We represent the description based on GloVe pre-trained word embeddings [11]. In the case of multiple tokens, we average the word embeddings. One disadvantage of this representation is that the description can be ambiguous, e.g., *"Title"* can refer to various different value types, e.g. `Movies, Books` etc.

Instance–Values. In case c_i contains *instance values*, we represent c_i through the average embeddings of the values v_i based on pre-computed graph embeddings. We use *node2vec* embeddings [7] trained on the Wikipedia *anchor graph*[1]. The combination of *description* and *instance values* improves the representation of a column and reduces its ambiguity.

Column–Type. Representing the columns based solely on instance values poses a risk of biasing the column representation towards articles that are often linked together in the Wikipedia anchor graph, and thus it may ignore the topic information that is present in such articles based on their category associations.

For columns that contain instance values, we additionally represent it through its *type* or *category*. From the instances v_i of c_i, we extract their LCA category from Ψ (a similar idea was employed by Das Sarma et al. [13]). We represent the LCA category through graph embeddings. In cases we have multiple LCA categories, then we average their corresponding representations.

[1]The anchor graph consists of nodes (Wikipedia articles and categories), while the edges correspond to the anchor text and the category-article associations.

5.2 Table Alignment Architecture

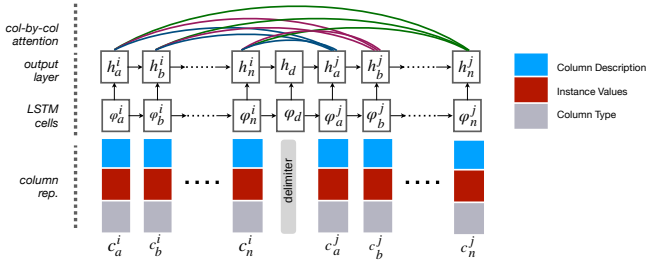


Figure 2: TableNet uses *BiLSTMs* to encode the tables as a sequence of columns. The *column-by-column* captures alignments between columns.

TableNet adopts the model from [12], and for a table pair $r(t_i, t_j)$ it predicts the relation as `equivalent`, `subPartOf`, or `none`. The alignment model is a bidirectional LSTM [8]. We compute a *column-by-column* attention, which helps us generate soft-alignments between columns in the table schemas, and thus further improve the alignment accuracy. Below we describe the encoding of table columns, and the intuition behind the attention mechanism.

Table Encoding. Since we have two separate tables, a precondition for accurate alignment is the encoding of the sequence of columns. Our model provides a *conditional encoding*, in that it first reads the columns from $C(t_i)$, then the cell state c_d (the table separator) is initialized with the last state of t_i (in this case c_n^i) and is used to conditionally encode the sequence of columns in $C(t_j)$.

The advantage of the conditional encoding is that by encoding table t_j with initial cell state that corresponds to the last column cell state of t_i, we bias the model to learn encodings that are optimal for the task of table alignment. That is instead of trying to encode all columns, it will learn to encode the columns of t_j such that it can best predict the relation for the table pair. Since we have a bidirectional LSTM, we encode in a similar fashion the table t_i by conditioning it on the last state of t_j.

Attention Mechanism. For a table pair to be aligned with either `equivalent` or `subPartOf` relation, we expect that the most important columns in each of the tables to have their corresponding matches. This is inline with the intuition that not all columns in a table are equally important [13].

The classification of the table pair $r(t_i, t_j)$ through a standard RNN model, is done by using the last cell state of the encoded table pair. This additionally forces the model to consider columns as being equally important. A common workaround is to use attention mechanism [3] or *global attention*, which is able to capture the importance of certain sequences. However, global attention is geared for language generation tasks, and as such is not suitable for the classification of $r(t_i, t_j)$.

Column-by-Column Attention. In TableNet, we employ the *column-by-column* attention mechanism, which works as following. After having encoded the last column from t_i, we process the columns in t_j individually and generate the attention weights w.r.t the columns in t_i. As a consequence, for each column in t_j we generate soft alignments to highest matching columns in t_j. After having processed all the columns in t_j and computing the corresponding attention weights (the upper part in Figure 2), for classification of the table pair $r(t_i, t_j)$ we will use a non-linear combination of the weighted representation of the last column c_n^j in t_j. We use *softmax* classification function for determining the label for $r(t_i, t_j)$.

6 EXPERIMENTAL SETUP

Here we describe the experimental setup. The entire evaluation setup and code of TableNet are available for download.[2]

6.1 Wikpedida Tables Dataset

We extract tables from the HTML content of Wikipedia articles. From the entire snapshot of Wikipedia, only 529,170 Wikipedia articles contain tables, resulting in a total of 3,238,201 tables. On average there are **6 tables per article** with an average of **6.6 columns**, and with an average of **10 rows** per table.

Additionally, more than 20% of columns in total are instance-values based (see Section 3), and additionally more than 85% of tables contain such columns. This shows that in the vast majority of cases, we can represent tables, specifically the columns with highly rich semantic representations.

6.2 Table Alignment Ground-Truth

Our ground-truth consists of a sample of 50 source Wikipedia articles from which we construct article candidate pairs. Since the *naive* approach would generate 26.5M pairs, we apply a set of *filtering keywords* to filter out irrelevant pairs. We filter articles by checking if a keyword appears *anywhere* in the article's content[3].

We manually inspect a random sample of pairs that are filtered out, and assess if we remove pairs that should be considered relevant, and consequentially refine the keywords. For article pairs that remain after filtering, we check if they can be seen as *false positives* and similarly refine our filtering keywords to remove such cases. We iteratively apply the refine and filtering steps, until we are left with an initial set of article pairs that we deploy for evaluation through crowdsourcing.

For the 50 source articles, we are left with 3.7k pairs after three rounds of filtering refinements, which result in a set of 17k table pairs that we evaluate by means of crowdsourcing.

6.2.1 Ground-Truth Statistics. From **17,047 table pairs**, after labeling our ground-truth consists of 52% table pairs marked with `noalignment` relation, 24% marked with as having `equivalent` alignment, and the remaining 23% with `subPartOf` relation. The 47% portion of table pairs with a relation, result from **876 article pairs**. The average agreement rate amongst crowdworkers is 0.91.

6.3 Baselines and TableNet setup

We compare *TableNet* in two main aspects: (i) efficiency in candidate pair generation, and (ii) table alignment.

6.3.1 ***Candidate Generation Baselines.***

Greedy – G. For each article we consider all other articles as pairs. This has maximal coverage, however, the amount of irrelevant pairs is extremely high.

[2] https://github.com/bfetahu/wiki_tables/

[3] A detailed description on the ground-truth generation is provided in the paper URL.

Direct Categories – C1. We consider as pairs articles that are associated with the same *directly connected* categories. Due to the noisy article-category associations, there is no guarantee that we will have maximal coverage of relevant pairs.
Deepest Category – C2. We consider as pairs all those articles that belong to the same *deepest associated category* in Ψ. Articles are associated with multiple categories across different levels in Ψ.
Parent Categories – PC. To increase the coverage of relevant pairs, we consider as pairs, articles that have the same *parent categories* based on the directly associated categories.

6.3.2 *Table Alignment Baselines.*

Google Fusion. The work in [13] finds related tables for a given table by computing a set relatedness scores against all possible table candidates. Two tables are related if their schemas are related based on max-weight bipartite graph matching score (see Section 2 for a detailed discussion). Google Fusion is unsupervised, thus, we use a threshold τ (we fine tune τ s.t we find the best F1 score) to classify table pairs as either having a relation or not.

BiLSTM. We train a BiLSTMs as a baseline for table alignment and use the different column representations from Section 5.

Setup: BiLSTM & TableNet. We use 100 dimensions for the *hidden layer* for the LSTM cells, and train for 50 epochs. We split the data with 60% for training, 10% for validation, and the remaining 30% for testing.

We represent columns based on the three representations (see Section 5). The simplest representation is the column description, TableNet^{d}, and then incrementally add the more complex representations, TableNet^{+v} and TableNet^{+t}, respectively.

6.4 Evaluation Metrics

Candidate Generation. The aim is to *minimize* the amount of *irrelevant article pairs* $\langle a_i, a_j \rangle = \emptyset$, and at the same time retain pairs whose tables have an alignment. We compute Δ to measure the amount of reduction we achieve w.r.t the *greedy* approach in generating article pairs:

$$\Delta = 1 - \frac{\langle a_i, a_j \rangle}{k * |A|} \text{ where } a_i \neq a_j \wedge a_i, a_j \in A \quad (2)$$

where, k is the number of source articles.

Coverage we measure through *micro* and *macro* recall indicated with R_μ and R, respectively. R_μ represents the recall in terms of all table pairs from all the source articles, whereas macro recall measures the average recall scores from all source articles.

Table Alignment. We measure the accuracy of the models for determining the table relation based on standard evaluation metrics, such as *precision* (P), *recall* (R), and *F1 score* (F1).

7 EVALUATION RESULTS

7.1 Candidate Generation Results

Table 2 shows the efficiency and coverage results for the baseline strategies. From the baselines we notice that the use of the Wikipedia category graph Ψ reduces the amount of irrelevant pairs drastically. In terms of recall, baseline **PC** maintains high recall with $R = 0.83$, and at the same time reduces by $\Delta = 87\%$ the amount of irrelevant pairs when compared to *greedy* approach. **C2** achieve the highest reduction rate Δ. Despite, the high reduction rates, we still face the issue of either having a highly imbalanced ratio of relevant and irrelevant pairs, or in some cases like **C2** where the reduction rate is the highest, the recall is low $R = 0.49$.

Table 2: Reduction rate for the baseline strategies.

	$\lvert\langle a_i, a_j \rangle\rvert$	Δ	*rel. pairs*	R
G	26,500,000	-	876	1.0
PC	3,486,031	0.87	724	0.83
C1	792,701	0.97	571	0.65
C2	6,738	0.99	440	0.50

TableNet: Filtering. The filtering step uses the features in Table 1 to remove irrelevant article pairs. Figure 3 shows the impact of the different features in reducing the amount of article pairs w.r.t *greedy* approach. For instance, the f_2 feature, which computes the similarity of article abstracts based on their *doc2vec* representation, provides a high reduction with $\Delta = 0.91$. This feature follows our intuition on generating the article pairs for the ground-truth (see Section 6), where the *topic* and other semantic similarities for an article pair can be extracted from the article's content.

Since different features capture different notions of similarity, we apply them in *conjunction*, resulting in very high reduction rate of article pairs with $\Delta > 0.99$, and at the same time retaining a relatively high coverage with $R = 0.68$. The reduction compared to the greedy approach is more than ▼255 times less pairs.

We believe that this high reduction rate and at the same time the relatively high recall of relevant pairs, when compared to the baseline approaches can be attributed to the fact that we consider the similarities of articles, and their corresponding categories and articles' content in the embedding space. This allows us to capture implicit semantics that cannot be captured through the simple link-based structure in the category graph Ψ.

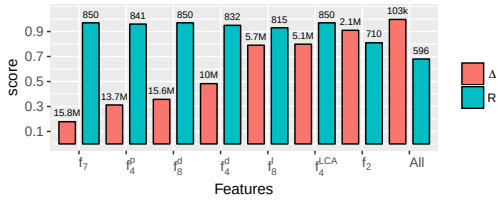


Figure 3: Feature impact in terms of reducing the amount of irrelevant pairs and the coverage of $\langle a_i, a_j \rangle_r$ pairs.

TableNet: Classification. Determining which article pairs, specifically, their tables will have a table relation is a difficult.

Despite the high reduction of irrelevant pairs from 26M pairs to 103k pairs, the amount of irrelevant pairs is still too high for any supervised approach to be able to learn models that predict with great accuracy the table relations. Thus, based on the configured RF model for high coverage (see Section 6), we train it on the feature set in Table 1 to further classify irrelevant pairs and filter them out.

Figure 4 shows the evaluation results for varying thresholds τ for the RF model. The increase of τ is directly proportional with the decrease in the amount of relevant pairs. This is intuitive as

from the 103k pairs, only 876 pairs are actually relevant. Based on the configuration of the RF (see Section 6), we retain relevant pairs with high coverage. We choose $\tau = 0.5$, as it shows the best trade-off between the coverage of relevant pairs, and the amount of irrelevant pairs that are passed onto the table alignment step. We achieve a high reduction rate of $\Delta = 0.982$ leaving us with only 1.8k pairs, and with a recall of $R_\mu = 0.81$.

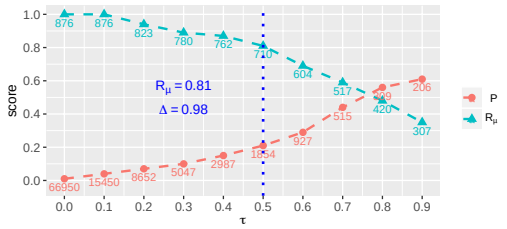


Figure 4: τ (x-axis) shows the scores for R_μ and P, and the corresponding amount of relevant pairs we retain.

Dependent on the scenario, higher τ can be used to have a higher ratio of relevant pairs, making the alignment task more efficient.

7.2 Table Alignment Results

Performance. Table 3 shows the table alignment evaluation results. In the case of *Google Fusion*, we consider a table pair to be *aligned* if their matching score is above some threshold that we determine empirically for which we achieve the highest F1 score.

Google Fusion. This baseline has a reasonably high accuracy in determining whether a $r(t_i, t_j) \neq \emptyset$. Here we cannot distinguish between the different classes as the approach is unsupervised. In terms of recall it has the lowest score. This is due to the fact that the matching is performed by considering only the *column type* and the *column title*similarity. Additionally, the *bipartite* matching algorithm cannot retain the order of the columns, which is highly important for determining the alignment relation.

BiLSTM. One key motivation in this work is the hypothesis that through sequence based models, we can retain the order of columns in their respective schemas, an important aspect in determining the table alignment relation. The BiLSTM approach represent a very competitive baseline. An additional advantage which addresses a deficiency in the standard supervised models, is that we jointly encode the different representations of a column for the classification task. Representing the columns as a combination of their description through word embeddings, and the type and instance values through graph embeddings, we can capture complex relationship between the column description and the underlying cell-values.

For `equivalent` relations, BiLSTM^{+d} achieves a very high F1 score with $F1 = 0.887$, which is close to the best result we achieve with TableNet^{+v}. For `subPartOf` relations, BiLSTM^{+v} the representation based on the column description and instance values achieves the highest F1 scores. The introduction of the column type in BiLSTM^{+t} provides a further boost in the accuracy of determining `subPartOf` relations. One conclusion we draw from the comparison between the two relations and two models, is that for `subPartOf` relations the *column type* provides additional power on determining the table alignment relation, whereas for `equivalent` it does not provide an additional advantage. These findings are inline with [13], where column type can provide important information in finding related tables. Comparing BiLSTM against *Google Fusion* we have a 64% improvement for `equivalent` relation.

TableNet. In our approach, we addressed several deficiencies from the related work. Through our *column-by-column* attention mechanism, we can compute soft alignments between columns in the respective table schemas and thus take into account the position of the matching columns in the corresponding schemas. Additionally, the column representations allow us to capture the similarity between columns and the schema context in which they appear, and additionally the representation context based on their description, type and its instance values.

The evaluation results reflect this intuition. Comparing our best performing setup, TableNet^{+t} achieves an overall $F1 = 0.840$ across all three classes, which presents also the highest F1 score across all competitors. We achieve a relative improvement of 64% when comparing F1 scores for the `equivalent` class against *Google Fusion*, or 56% if we compare the average F1 score for both alignment relations (`equivalent` and `subPartOf`). Furthermore, TableNet^{+t} outperforms the BiLSTM approach on average F1 score across all classes. For the individual classes, we note a variations amongst the different configurations of TableNet that perform best (marked in bold). This confirms the usefulness of the attention mechanism for the alignment task, where we achieve an overall better performance in terms of F1 score.

8 CONCLUSIONS AND FUTURE WORK

We presented TableNet, an approach for table alignment that ensures *high coverage* of retained table relations, and provides highly accurate *fine-grained* table alignments.

We constructed an exhaustive ground truth for a random sample of 50 Wikipedia articles for which we labeled all possible table pairs, providing a dataset against which we can measure the coverage and accuracy of approaches in determining table relations for more than 17k table pairs in our ground-truth.

In terms of *efficiency*, we show that from a naive approach which produces 26.5M pairs, we can guarantee a high coverage of retaining relevant article pairs with more than 68%, while, at the same time reducing the amount of irrelevant pairs by ▼255 times. In terms of *table alignment*, we show that we can improve over strong baselines and provide high improvement over strong baselines and other competitors like *Google Fusion*.

ACKNOWLEDGMENT:

This work is funded by the ERC Advanced Grant ALEXANDRIA (grant no. 339233), and BMBF Simple-ML project (grant no. 01IS18054A).

REFERENCES

[1] Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2018. Never-Ending Learning for Open-Domain Question Answering over Knowledge Bases. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018.* 1053–1062. https://doi.org/10.1145/3178876.3186004

[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.* 722–735. https://doi.org/10.1007/978-3-540-76298-0_52

Table 3: Evaluation results for the tasks of table alignment for the different competitors and TableNet. The evaluation results correspond to our manually constructed ground-truth dataset.

	equivalent			subPartOf			noalignment					
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**Acc**	**R**	**F1**
Google Fusion	0.809	0.405	0.540	-	-	-	-	-	-			
BiLSTM^{d}	0.883	0.891	0.887	0.684	0.960	0.799	0.918	0.752	0.827	0.828	0.868	0.838
BiLSTM^{+v}	0.877	0.871	0.874	0.684	0.975	0.804	0.915	0.747	0.823	0.826	0.864	0.834
BiLSTM^{+t}	0.854	0.908	0.880	0.690	0.957	0.802	0.925	0.741	0.823	0.823	0.869	0.835
TableNet^{d}	**0.888**	0.884	0.886	0.686	0.947	0.796	0.909	0.759	0.827	0.828	0.863	0.836
TableNet^{+v}	0.856	**0.926**	**0.890**	0.675	**0.993**	0.804	**0.952**	0.719	0.819	0.828	**0.880**	0.838
TableNet^{+t}	0.872	0.903	0.887	0.692	0.961	**0.805**	0.925	0.752	**0.829**	**0.830**	0.872	**0.840**

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[4] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[5] Michael J. Cafarella, Alon Y. Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. WebTables: exploring the power of tables on the web. *PVLDB* 1, 1 (2008), 538–549. http://www.vldb.org/pvldb/1/1453916.pdf

[6] Hector Gonzalez, Alon Y. Halevy, Christian S. Jensen, Anno Langen, Jayant Madhavan, Rebecca Shapley, Warren Shen, and Jonathan Goldberg-Kidon. 2010. Google fusion tables: web-centered data management and collaboration. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, Indianapolis, Indiana, USA, June 6-10, 2010*. 1061–1066. https://doi.org/10.1145/1807167.1807286

[7] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 855–864. https://doi.org/10.1145/2939672.2939754

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[9] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*. 1188–1196. http://jmlr.org/proceedings/papers/v32/le14.html

[10] Emir Muñoz, Aidan Hogan, and Alessandra Mileo. 2014. Using linked data to mine RDF from wikipedia's tables. In *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*. 533–542. https://doi.org/10.1145/2556195.2556266

[11] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. 1532–1543. http://aclweb.org/anthology/D/D14/D14-1162.pdf

[12] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomas Kocisky, and Phil Blunsom. 2016. Reasoning about Entailment with Neural Attention. In *International Conference on Learning Representations (ICLR)*.

[13] Anish Das Sarma, Lujun Fang, Nitin Gupta, Alon Y. Halevy, Hongrae Lee, Fei Wu, Reynold Xin, and Cong Yu. 2012. Finding related tables. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*. 817–828. https://doi.org/10.1145/2213836.2213962

[14] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. 697–706. https://doi.org/10.1145/1242572.1242667

[15] Petros Venetis, Alon Y. Halevy, Jayant Madhavan, Marius Pasca, Warren Shen, Fei Wu, Gengxin Miao, and Chung Wu. 2011. Recovering Semantics of Tables on the Web. *PVLDB* 4, 9 (2011), 528–538. http://www.vldb.org/pvldb/vol4/p528-venetis.pdf

Learning Graph Pooling and Hybrid Convolutional Operations for Text Representations

Hongyang Gao
Texas A&M University
College Station, TX
hongyang.gao@tamu.edu

Yongjun Chen
Washington State University
Pullman, WA
yongjun.chen@wsu.edu

Shuiwang Ji
Texas A&M University
College Station, TX
sji@tamu.edu

ABSTRACT

With the development of graph convolutional networks (GCN), deep learning methods have started to be used on graph data. In additional to convolutional layers, pooling layers are another important components of deep learning. However, no effective pooling methods have been developed for graphs currently. In this work, we propose the graph pooling (gPool) layer, which employs a trainable projection vector to measure the importance of nodes in graphs. By selecting the k-most important nodes to form the new graph, gPool achieves the same objective as regular max pooling layers operation on images and texts. Another limitation of GCN when used on graph-based text representation tasks is that, GCNs do not consider the order information of nodes in graph. To address this limitation, we propose the hybrid convolutional (hConv) layer that combines GCN and regular convolutional operations. The hConv layer is capable of increasing receptive fields quickly and computing features automatically. Based on the proposed gPool and hConv layers, we develop new deep networks for text categorization tasks. Our experimental results show that the networks based on gPool and hConv layers achieves new state-of-the-art performance as compared to baseline methods.

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**; **Neural networks**; *Artificial intelligence*; *Structured outputs.*

KEYWORDS

Graph; Pooling; Text Classification

ACM Reference Format:
Hongyang Gao, Yongjun Chen, and Shuiwang Ji. 2019. Learning Graph Pooling and Hybrid Convolutional Operations for Text Representations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313395

1 INTRODUCTION

Convolutional neural networks (CNNs) [11] have shown great capability of solving challenging tasks in various fields such as computer vision and natural language processing (NLP). A variety of CNN networks have been proposed to continuously set new performance records [4, 6, 10, 28]. In addition to image-related tasks, CNNs are also successfully applied to NLP tasks such as text classification [38] and neural machine translation [1, 31]. The power of CNNs lies in trainable local filters for automatic feature extraction. The networks can decide which kind of features to extract with the help of these trainable local filters, thereby avoiding hand-crafted feature extractors [33].

One common characteristic behind the above tasks is that both images and texts can be represented in grid-like structures, thereby enabling the application of convolutional operations. However, some data in real-world can be naturally represented as graphs such as social networks. There are primarily two types of tasks on graph data; those are, node classification [9, 32] and graph classification [21, 34]. Graph convolutional networks (GCNs) [9] and graph attention networks (GATs) [32] have been proposed for node classification tasks under both transductive and inductive learning settings. However, GCNs lack the capability of automatic high-level feature extraction, since no trainable local filters are used.

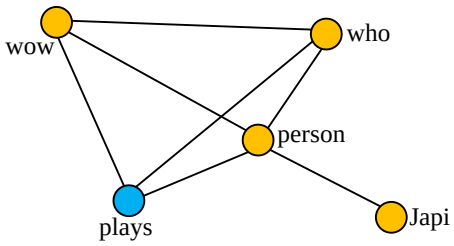
In addition to convolutional layers, pooling layers are another important components in CNNs by helping to enlarge receptive fields and reduce the risk of over-fitting. However, there are still no effective pooling operations that can operate on graph data and extract sub-graphs. Meanwhile, GCNs have been applied on text data by converting texts into graphs [15, 27]. Although GCNs can operate on graphs converted from texts, they ignore the ordering information between nodes, which correspond to words in texts.

In this work, we propose a novel pooling layer, known as the graph pooling (gPool) layer, that acts on graph data. Our method employs a trainable projection vector to measure the importance of nodes in a graph. Based on measurement scores, we rank and select k-largest nodes to form a new sub-graph, thereby achieving pooling operation on graph data. When working on graphs converted from text data, the words are treated as nodes in the graphs. By maintaining the order information in nodes' feature matrices, we can apply convolutional operations to feature matrices. Based on this observation, we develop a new graph convolutional layer, known as the hybrid convolutional (hConv) layer. Based on gPool and hConv layers, we develop a shallow but effective architecture for text modeling tasks [16]. Results on text classification tasks demonstrate the effectiveness of our proposed methods as compared to previous CNN models.

2 RELATED WORK

Before applying graph-based methods on text data, we need to convert texts to graphs. In this section, we discuss related literatures on converting texts to graphs and use of GCNs on text data.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313395



"Japi is a person who plays wow."

Figure 1: Example of converting text to a graph using the graph-of-words method. For this text, we use noun, adjective, and verb as terms for node selection. The words of "Japi", "person", "who", "plays", and "wow" are selected as nodes in the graph. We employ a sliding window size of 4 for edge building. For instance, there is an undirected edge between "Japi" and "person", since they can be covered in the same sliding window in the original text.

2.1 Text to Graph Conversion

Many graph representations of texts have been explored to capture the inherent topology and dependence information between words. Bronselaer and Pasi [3] employed a rule-based classifier to map each tag onto graph nodes and edges. The tags are acquired by the part-of-speech (POS) tagging techniques. In [15] a concept interaction graph representation is proposed for capturing complex interactions among sentences and concepts in documents. The graph-of-word representation (GoW) [19] attempts to capture co-occurrence relationships between words known as terms. It was initially applied to text ranking task and has been widely used in many NLP tasks such as information retrieval [25], text classification [17, 24], keyword extraction [26, 30] and document similarity measurement [22].

Before applying graph-based text modeling methods, we need to convert texts to graphs. In this work, we employ the graph-of-words [19] method for its effectiveness and simplicity. The conversion starts with the phase preprocessing such as tokenization and text cleaning. After preprocessing, each text is encoded into an unweighted and undirected graph in which nodes represent selected terms and edges represent co-occurrences between terms within a fixed sliding window. A term is a group of words clustered based on their part-of-speech tags such as noun and adjective. The choice of sliding window size depends on the average lengths of processed texts. Figure 1 provides an example of this method.

2.2 GCN and its Applications on Text Modeling

Recently, many studies [9, 32] have attempted to apply convolutional operations on graph data. Graph convolutional networks (GCNs) [9] were proposed and have achieved the state-of-the-art performance on inductive and transductive node classification tasks. The spectral graph convolution operation is defined such that a convolution-like operation can be applied on graph data. Basically, each GCN layer updates the feature representation of each node by aggregating the features of its neighboring nodes. To be specific, the layer-wise propagation rule of GCNs can be defined as $\text{gcn}(\hat{A}, X_\ell) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}X_\ell W_\ell)$, where X_ℓ and $X_{\ell+1}$ are input and output matrices of layer ℓ, respectively. The numbers of rows of two matrices are the same, which indicates that the number of nodes in graph does not change in GCN layers. We use $\hat{A} = A + I$ to include self-loop edges to the graph. The diagonal node degree matrix $\hat{D}$ is generated to normalize $\hat{A}$ such that the scale of feature vectors remains the same after aggregation. W^ℓ is the trainable weight matrix, which plays the role of linear transformation of each node's feature vector. Finally, $\sigma(\cdot)$ denotes the nonlinear activation function such as ReLU [20]. Later, graph attention networks (GAT) [32] are proposed to solve node classification tasks by using the attention mechanism [35].

In addition to graph data, some studies attempted to apply graph-based methods to grid-like data such as texts. Compared to traditional recurrent neural networks such as LSTM [5], GCNs have the advantage of considering long-term dependencies by edges in graphs. Marcheggiani and Titov [18] applied a variant of GCNs to the task of sentence encoding and achieved better performance than LSTM. GCNs have also been used in neural machine translation tasks [2]. Although graph convolutional operations have been extensively developed and explored, pooling operations on graphs are not well studied currently.

3 GRAPH POOLING AND HYBRID CONVOLUTIONAL OPERATIONS

In this section, we describe our proposed graph pooling (gPool) and hybrid convolutional (hConv) operations. Based on these two new layers, we develop FCN-like graph convolutional networks for text modeling tasks.

3.1 Graph Pooling Layer

Pooling layers are very important for CNNs on grid-like data such as images, as they can help quickly enlarge receptive fields and reduce feature map size, thereby resulting in better generalization and performance [36]. In pooling layers, input feature maps are partitioned into a set of non-overlapping rectangles on which non-linear down-sampling functions, such as maximum, are applied. Obviously, the partition scheme depends on the local adjacency on grid-like data like images. For instance in a max-pooling layer with a kernel size of 2×2, 4 spatially adjacent units form a partition. However, such kind of spatial adjacency information is not available for nodes on graphs. We cannot directly apply regular pooling layers to graph data.

To enable pooling operations on graph data, we propose the graph pooling layer (gPool), which adaptively selects a subset of important nodes to form a smaller new graph. Suppose a graph has N nodes and each node contains C features. We can represent the graph with two matrices; those are the adjacency matrix $A^\ell \in \mathbb{R}^{N\times N}$ and the feature matrix $X^\ell \in \mathbb{R}^{N\times C}$. Each row in X^ℓ corresponds to a node in the graph. We propose the layer-wise propagation rule of gPool to be defined as

$$\begin{aligned} \mathbf{y} &= |X^\ell \mathbf{p}^\ell|, & \text{idx} &= \text{rank}(\mathbf{y}, k), \\ A^{\ell+1} &= A^\ell(\text{idx}, \text{idx}), & \tilde{X}^\ell &= X^\ell(\text{idx}, :), \\ \tilde{\mathbf{y}} &= \tanh(\mathbf{y}(\text{idx})), & X^{\ell+1} &= \tilde{X}^\ell \odot \left(\tilde{\mathbf{y}}\mathbf{1}_C^T\right), \end{aligned} \tag{1}$$

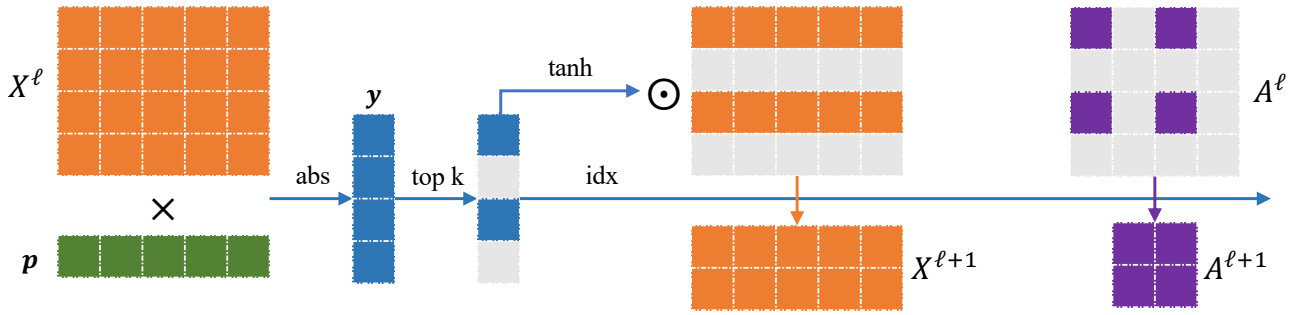


Figure 2: An illustration of the proposed graph pooling layer which samples $k = 2$ nodes. $\times$ and $\odot$ represent matrix and element-wise multiplication, respectively. This example graph has four nodes, each of which contains 5 features. We have the adjacency matrix $A^\ell \in \mathbb{R}^{4\times4}$ and the input feature matrix $X^\ell \in \mathbb{R}^{4\times5}$ of layer ℓ representing this graph. $\mathbf{p} \in \mathbb{R}^5$ is a trainable projection vector in this layer. By matrix multiplication and abs$(\cdot)$, we have scores $\mathbf{y}$ which estimate the closeness of each node to the projection vector. Using $k = 2$, we select two nodes with the highest scores and record their indices in idx, which represents indices of selected nodes. With indices idx, we extract corresponding nodes to form the new graph, which results in a new pooled feature map $\tilde{X}^\ell$ and adjacency matrix $A^{\ell+1}$. To control information flow, we create a gate vector by applying element-wise tanh$(\cdot)$ **to score vector $\mathbf{y}$. By element-wise multiplication between gate vector and $\tilde{X}^\ell$, we obtain the $X^{\ell+1}$. Outputs of this graph pooling layer are $A^{\ell+1}$ and $X^{\ell+1}$.**

where k is the number of nodes to be selected from the graph, $\mathbf{p}^\ell \in \mathbb{R}^C$ is a trainable projection vector of layer ℓ, rank$(\mathbf{y})$ is an operation that returns the indices corresponding to the k-largest values in $\mathbf{y}$, A^ℓ(idx, idx) extracts the rows and columns with indices idx, X^ℓ(idx, :) selects the rows with indices idx using all column values, $\mathbf{y}$(idx) returns the corresponding values in $\mathbf{y}$ with indices idx, $\mathbf{1}_C \in \mathbb{R}^C$ is a vector of size C with all components being 1, and $\odot$ denotes the element-wise matrix multiplication operation.

Suppose X^ℓ is a matrix with row vectors $\mathbf{x}_1^\ell, \mathbf{x}_2^\ell, \cdots, \mathbf{x}_N^\ell$. We first perform a matrix multiplication between X^ℓ and $\mathbf{p}^\ell$ followed by an element-wise abs$(\cdot)$ operation, resulting in $\mathbf{y} = [y_1, y_2, \cdots, y_N]^T$ with each y_i measuring the closeness between the node feature vector $\mathbf{x}_i^\ell$ and the projection vector $\mathbf{p}^\ell$.

The rank$(\cdot)$ operation ranks the N values in $\mathbf{y}$ and returns the indices of the k-largest values. Suppose the indices of the k selected values are $i_1, i_2, \cdots, i_k$ with $i_m < i_n$ if $1 \le m < n \le k$. These indices correspond to nodes in the graph. Note that the selection process retains the order information of selected nodes in the original feature matrix. With indices, the k-node selection is conducted on the feature matrix X^l, the adjacency matrix A^ℓ, and the score vector $\mathbf{y}$. We concatenate the corresponding feature vectors $\mathbf{x}_{i_1}^\ell, \mathbf{x}_{i_2}^\ell, \cdots, \mathbf{x}_{i_k}^\ell$ and output a matrix $\tilde{X}^\ell \in \mathbb{R}^{k\times C}$. Similarly, we use scores $\tanh(y_{i_1}), \tanh(y_{i_2}), \cdots, \tanh(y_{i_k})$ as elements of the vector $\tilde{\mathbf{y}} \in \mathbb{R}^k$. For the adjacency matrix A^ℓ, we extract rows and columns based on the selected indices and output the adjacency matrix $A^{\ell+1} \in \mathbb{R}^{k\times k}$ for the new graph.

Finally, we perform a gate operation to control the information flow in this layer. Using element-wise product of $\tilde{X}^\ell$ and $\tilde{\mathbf{y}}\mathbf{1}_C^T$, features of selected nodes are filtered through their corresponding scores and form the output $X^{\ell+1} \in \mathbb{R}^{k\times C}$. The ith row vector in $X^{\ell+1}$ is the product of the corresponding row vector in $\tilde{X}^\ell$ and the i^{th} scalar number in $\tilde{\mathbf{y}}$. In addition to information control, the gate operation also makes the projection vector $\mathbf{p}^\ell$ trainable with back-propagation [12]. Without the gate operation, the projection vector $\mathbf{p}^\ell$ only contributes discrete indices to outputs and thus is not trainable. Figure 2 provides an illustration of the gPool layer.

Compared to regular pooling layers used on grid-like data, our gPool layer involves extra parameters in the trainable projection vector $\mathbf{p}^\ell$. In Section 4.6, we show that the number of extra parameters is negligible and would not increase the risk of over-fitting.

3.2 Hybrid Convolutional Layer

It follows from the analysis in Section 2.2 that GCN layers only perform convolutional operations on each node. There is no trainable spatial filters as in regular convolution layers. GCNs do not have the power of automatic feature extraction as achieved by CNNs. This limits the capability of GCNs, especially in the field of graph modeling. In traditional graph data, there is no ordering information among nodes. In addition, the different numbers of neighbors for each node in the graph prohibit convolutional operations with a kernel size larger than 1. Although we attempt to modeling texts as graph data, they are essentially grid-like data with order information among nodes, thereby enabling the application of regular convolutional operations.

To take advantage of convolutional operations with trainable filters, we propose the hybrid convolutional layer (hConv), which combines GCN operations and regular 1-D convolutional operations to achieve the capability of automatic feature extraction. Formally, we propose the hConv layer to be defined as

$$\begin{aligned} X_1^{\ell+1} &= \text{conv}(X^\ell), \qquad X_2^{\ell+1} = \text{gcn}(A^\ell, X^\ell), \\ X^{\ell+1} &= \left[X_1^{\ell+1}, X_2^{\ell+1}\right], \end{aligned} \tag{2}$$

where conv$(\cdot)$ denotes a regular 1-D convolutional operation, and the gcn$(\cdot,\cdot)$ operation is defined in Eq. **??**. For the feature matrix X^ℓ, we treat the column dimension as the channel dimension, such that the 1-D convolutional operation can be applied along the row dimension. Using the 1-D convolutional operation and the GCN operation, we obtain two intermediate outputs; those are $X_1^{\ell+1}$ and

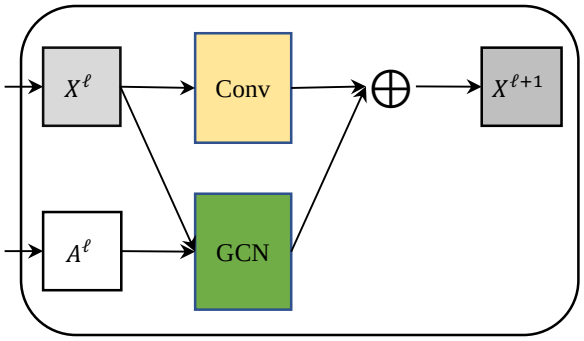


Figure 3: An illustration of the hybrid convolutional layer. $\oplus$ denotes matrix concatenation along the row dimension. In this layer, A^ℓ and X^ℓ are the adjacency matrix and the node feature matrix, respectively. A regular 1-D convolutional operation is used to extract high-level features from sentence texts. The GCN operation is applied at the graph level for feature extraction. The two intermediate outputs are concatenated together to form the final output $X^{\ell+1}$.

$X_2^{\ell+1}$. These two matrices are concatenated together as the layer output $X^{\ell+1}$. Figure 3 illustrates an example of the hConv layer.

We argue that the integration of GCN operations and 1-D convolutional operations in the hConv layer is especially applicable to graph data obtained from texts. By representing texts as an adjacency matrix A^ℓ and a node feature matrix X^ℓ of layer ℓ, each node in the graph is essentially a word in the text. We retain the order information of nodes from their original relative positions in texts. This indicates that the feature matrix X^ℓ is organized as traditional grid-like data with order information retained. From this point, we can apply an 1-D convolutional operation with kernel sizes larger than 1 on the feature matrix X^ℓ for high-level feature extraction.

The combination of the GCN operation and the convolutional operation in the hConv layer can take the advantages of both of them and overcome their respective limitations. In convolutional layers, the receptive fields of units on feature maps increase very slow since small kernel sizes are usually used to avoid massive number of parameters. In contrast, GCN operations can help to increase the receptive fields quickly by means of edges between terms in sentences corresponding to nodes in graphs. At the same time, GCN operations are not able to automatically extract high-level features as they do not have trainable spatial filters as used in convolutional operations. From this point, the hConv layer is especially useful when working on text-based graph data such as sentences and documents.

3.3 Network Architectures

Based on our proposed gPool and hConv layers, we design four network architectures, including our baseline with a FCN-like [16] architecture. FCN has been shown to be very effective for image semantic segmentation. It allows final linear classifiers to make use of features from different layers. Here, we design four architectures based on our proposed gPool and hConv layers.

Datasets	#Training	#Testing	#Classes	#Words
AG's News	120,000	7,600	4	45
DBPedia	560,000	70,000	14	55
Yelp Polarity	560,000	38,000	2	153
Yelp Full	650,000	50,000	5	155

Table 1: Summary of datasets used in our experiments. The #words denotes the average number of words of the data samples for each dataset. These numbers help the selection of the sliding window size used in converting texts to graphs.

- **GCN-Net:** We establish a baseline method by using GCN layers to build a network without any hConv or gPool layers. In this network, we stack 4 standard GCN layers as feature extractors. Starting from the second layer, a global max-pooling layer [14] is applied to each layer's output. The outputs of these pooling layers are concatenated together and fed into a fully-connected layer for final predictions. This network serves as a baseline model in this work for experimental studies.
- **GCN-gPool-Net:** In this network, we add our proposed gPool layers to GCN-Net. Starting from the second layer, we add a gPool layer after each GCN layer except for the last one. In each gPool layer, we select the hyper-parameter k to reduce the number of nodes in the graph by a factor of two. All other parts of the network remain the same as those of GCN-Net.
- **hConv-Net:** For this network, we replace all GCN layers in GCN-Net by our proposed hConv layers. To ensure the fairness of comparison among these networks, the hConv layers output the same number of feature maps as the corresponding GCN layers. Suppose the original ith GCN layer outputs n_{out} feature maps. In the corresponding hConv layer, both the GCN operation and the convolutional operation output $n_{out}/2$ feature maps. By concatenating those intermediate outputs, the ith hConv layer also outputs n_{out} feature maps. The remaining parts of the network remain the same as those in GCN-Net.
- **hConv-gPool-Net:** Based on the hConv-Net network, we add gPool layers after each hConv layer except for the first and the last layers. We employ the same principle for the selection of hyper-parameter k as that in GCN-gPool-Net. The remaining parts of the network remain the same. Note that gPool layers maintain the order information of nodes in the new graph, thus enabling the application of 1-D convolutional operations in hConv layers afterwards. Figure 4 provides an illustration of the hConv-gPool-Net network.

4 EXPERIMENTAL STUDY

In this section, we evaluate our gPool layer and hConv layer based on the four networks proposed in Section 3.3. We compare the performances of our networks with that of previous state-of-the-art models. The experimental results show that our methods yield improved performance in terms of classification accuracy. We also perform some ablation studies to examine the contributions of the gPool layer and the hConv layer to the performance. The number of extra parameters in gPool layers is shown to be negligible and will not increase the risk of over-fitting.

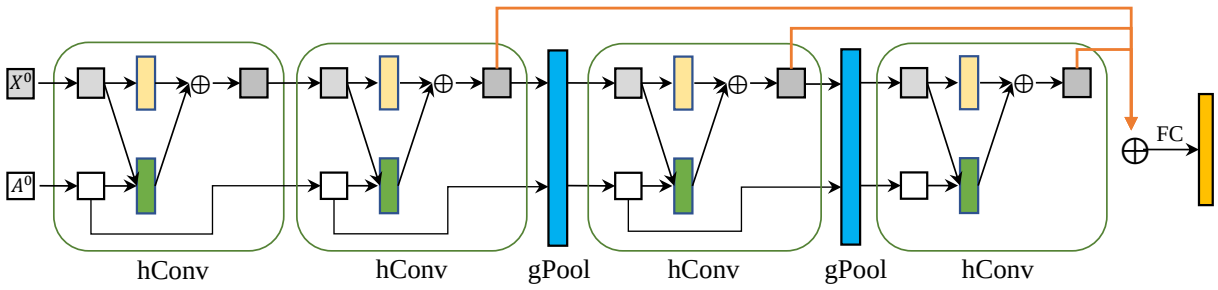


Figure 4: The architecture of the hConv-gPool-Net. $\oplus$ denotes the concatenation operation of feature maps. The inputs of the network are an adjacency matrix A^0 and a feature matrix X^0. We stack four hConv layers for feature extraction. In the second and the third hConv layers, we employ the gPool layers to reduce the number of nodes in graphs by half. Starting from the second hConv layer, a global max-pooling layer is applied to the output feature maps of each hConv layer. The outputs of these pooling layers are concatenated together. Finally, we employ a fully-connected layer for predictions. To obtain the other three networks discussed in Section 3.3, we can simply replace the hConv layers with GCN layers or remove gPool layers based on this network architecture.

Models	AG's News	DBPedia	Yelp Polarity	Yelp Full
Word-level CNN [38]	8.55%	1.37%	4.60%	39.58%
Char-level CNN [38]	9.51%	1.55%	4.88%	37.95%
GCN-Net	8.64%	1.69%	7.74%	42.60%
GCN-gPool-Net	8.09%	1.44%	5.82%	41.83%
hConv-Net	7.49%	1.02%	4.45%	37.81%
hConv-gPool-Net	**7.09%**	**0.92%**	**4.37%**	**36.27%**

Table 2: Results of text classification experiments in terms of classification error rate on the AG's News, DBPedia, Yelp Review Polarity, and Yelp Review Full datasets. The first two methods are the state-of-the-art models without using any unsupervised data. The last four networks are proposed in this work.

4.1 Datasets

In this work, we evaluate our methods on four datasets, including the AG's News, Dbpedia, Yelp Polarity, and Yelp Full [38] datasets. **AG's News** is a news dataset containing four topics: World, Sports, Business and Sci/Tech. The task is to classify each news into one of the topics. **Dbpedia** ontology dataset contains 14 ontology classes. It is constructed by choosing 14 non-overlapping classes from the DBPedia 2014 dataset [13]. Each sample contains a title and an abstract corresponding to a Wikipedia article. **Yelp Polarity** dataset is obtained from the Yelp Dataset Challenge in 2015 [38]. Each sample is a piece of review text with a binary label (negative or positive). **Yelp Full** dataset is obtained from the Yelp Dataset Challenge in 2015, which is for sentiment classification [38]. It contains five classes corresponding to the movie review star ranging from 1 to 5. The summary of these datasets are provided in Table 1. For all datasets, we tokenize the textual document and convert words to lower case. We remove stop-words and all punctuation in texts. Based on cleaned texts, we build the graph-of-word representations for texts.

4.2 Text to Graph Conversion

We use the graph-of-words method to convert texts into graph representations that include an adjacency matrix and a feature matrix. We select nouns, adjective, and verb as terms, meaning a word appears in the graph if it belongs to one of the above categories. We use a sliding window to decide if two terms have an edge between them. If the distance between two terms is less than the window size, an undirected edge between these two terms is added. In the generated graph, nodes are the terms appear in texts, and edges are added using the sliding window. We use a window size of 4 for the AG's News and DBpedia datasets and 10 for the other two datasets, depending on their average words in training samples. The maximum numbers of nodes in graphs for the AG's News, DBPedia, Yelp Polarity, Yelp Full datasets are 100, 100, 300, and 256, respectively.

To produce the feature matrix, we use word embedding and position embedding features. For word embedding features, the pre-trained fastText word embedding vectors [7] are used, and it contains more than 2 million pre-trained words vectors. Compared to other pre-trained word embedding vectors such as GloVe [23], using the fastText helps us to avoid massive unknown words. On the AG's News dataset, the number of unknown words with the fastText is only several hundred, which is significantly smaller than the number using GloVe. In addition to word embedding features, we also employ position embedding method proposed in Zeng et al. [37]. We encode the positions of words in texts into one-hot vectors and concatenate them with word embedding vectors. We obtain the feature matrix by stacking word vectors of nodes in the row dimension.

Models	Depth	Error Rate
Word-level CNN	9	8.55%
Character-level CNN	9	9.51%
GCN-Net	5	8.64%
GCN-gPool-Net	5	8.09%
hConv-Net	5	7.49%
hConv-gPool-Net	5	**7.09%**

Table 3: Comparison in terms of the network depth and the text classification error rate on the AG's News dataset. The depth listed here is calculated by counting the number of convolutional and fully-connected layers in networks.

Models	Error rate	# Params	Ratio of increase
GCN-Net	8.64%	1,554,820	0.00%
GCN-gPool-Net	**8.09%**	1,555,719	0.06%

Table 4: Comparison between the GCN-Net and GCN-gPool-Net in term of parameter numbers and text classification error rates on the AG's News dataset.

4.3 Experimental Setup

For our proposed networks, we employ the same settings with minor adjustments to accommodate the different datasets. As discussed in Section 3.3, we stack four GCN or hConv layers for GCN-based networks or hConv-based networks. For the networks using gPool layers, we add gPool layers after the second and the third GCN or hConv layers. Four GCN or hConv layers output 1024, 1024, 512, and 256 feature maps, respectively. We use this decreasing number of feature maps, since GCNs help to enlarge the receptive fields very quickly. We do not need more high-level features in deeper layers. The kernel sizes used by convolutional operations in hConv layers are all 3×1. For all layers, we use the ReLU [20] for nonlinearity. For all experiments, the following settings are shared. For training, the Adam optimizer [8] is used for 60 epochs. The learning rate starts at 0.001 and decays by 0.1 at the $30^t h$ and the $50^t h$ epoch. We employ the dropout with a keep rate of 0.55 [29] and batch size of 256. These hyper-parameters are tuned on the AG's News dataset, and then ported to other datasets.

4.4 Performance Study

We compare our proposed methods with other state-of-the-art models, and the experimental results are summarized in Table 2. We can see from the results that our hConv-gPool-Net outperforms both word-level CNN and character-level CNN by at least a margin of 1.46%, 0.45%, 0.23%, and 3.31% on the AG's News, DBPedia, Yelp Polarity, and Yelp Full datasets, respectively. The performance of GCN-Net with only GCN layers cannot compete with that of word-level CNN and char-level CNN primarily due to the lack of automatic high-level feature extraction. By replacing GCN layers using our proposed hConv layers, hConv-Net achieves better performance than the two CNN models across four datasets. This demonstrates the promising performance of our hConv layer by employing regular convolutional operations for automatic feature extraction. By comparing the GCN-Net with GCN-gPool-Net, and hConv-Net with hConv-gPool-Net, we observe that our proposed gPool layers promote both models' performance by at least a margin of 0.4%, 0.1%, 0.08%, and 1.54% on the AG's News, DBPedia, Yelp Polarity, and Yelp Full datasets. The margins tend to be larger on harder tasks. This observation demonstrates that our gPool layer helps to enlarge receptive fields and reduce spatial dimensions of graphs, resulting in better generalization and performance.

4.5 Network Depth Study

In addition to performance study, we also conduct experiments to evaluate the relationship between performance and network depth in terms of the number of convolutional and fully-connected layers in models. The results are summarized in Table 3. We can observe from the results that our models only require 5 layers, including 4 convolutional layers and 1 fully-connected layer. Both word-level CNN and character-level CNN models need 9 layers in their networks, which are much deeper than ours. Our hConv-gPool-Net achieves the new state-of-the-art performance with fewer layers, demonstrating the effectiveness of gPool and hConv layers. Since GCN and gPool layers enlarge receptive fields quickly, advanced features are learned in shallow layers, leading to shallow networks but better performance.

4.6 Parameter Study of gPool Layer

Since gPool layers involve extra trainable parameters in projection vectors, we study the number of parameters in gPool layers in the GCN-gPool-Net that contains two gPool layers. The results are summarized in Table 4. We can see from the results that gPool layers only needs 0.06% additional parameters compared to GCN-Net. We believe that this negligible increase of parameters will not increase the risk of over-fitting. With negligible additional parameters, gPool layers can yield a performance improvement of 0.54%.

5 CONCLUSION

In this work, we propose the gPool and hConv layers in FCN-like graph convolutional networks for text modeling. The gPool layer achieves the effect of regular pooling operations on graph data to extract important nodes in graphs. By learning a projection vector, all nodes are measured through cosine similarity with the projection vector. The nodes with the k-largest scores are extracted to form a new graph. The scores are then applied to the feature matrix for information control, leading to the additional benefit of making the projection vector trainable. Since graphs are extracted from texts, we maintain the node orders as in the original texts. We propose the hConv layer that combines GCN and regular convolutional operations to enable automatic high-level feature extraction. Based on our gPool and hConv layers, we propose four networks for the task of text categorization. Our results show that the model based on gPool and hConv layers achieves new state-of-the-art performance compared to CNN-based models. gPool layers involve negligible number of parameters but bring significant performance boosts, demonstrating its contributions to model performance.

ACKNOWLEDGMENTS

This work was supported in part by National Science Foundation grant IIS-1908166.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations* (2015).

[2] Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. Graph convolutional encoders for syntax-aware neural machine translation. *arXiv preprint arXiv:1704.04675* (2017).

[3] Antoon Bronselaer and Gabriella Pasi. 2013. An approach to graph-based analysis of textual documents. In *8th European Society for Fuzzy Logic and Technology (EUSFLAT-2013)*. Atlantis Press, 634–641.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

[5] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[6] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Vol. 1. 3.

[7] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, Vol. 2. 427–431.

[8] Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *The International Conference on Learning Representations* (2015).

[9] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations* (2017).

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[11] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[12] Yann LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. 2012. Efficient backprop. In *Neural networks: Tricks of the trade*. Springer, 9–48.

[13] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[14] Min Lin, Qiang Chen, and Shuicheng Yan. 2013. Network in network. *arXiv preprint arXiv:1312.4400* (2013).

[15] Bang Liu, Ting Zhang, Di Niu, Jinghong Lin, Kunfeng Lai, and Yu Xu. 2018. Matching Long Text Documents via Graph Convolutional Networks. *arXiv preprint arXiv:1802.07459* (2018).

[16] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3431–3440.

[17] Fragkiskos D Malliaros and Konstantinos Skianis. 2015. Graph-based term weighting for text categorization. In *Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on*. IEEE, 1473–1479.

[18] Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826* (2017).

[19] Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

[20] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 807–814.

[21] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. 2016. Learning convolutional neural networks for graphs. In *International Conference on Machine Learning*. 2014–2023.

[22] Giannis Nikolentzos, Polykarpos Meladianos, François Rousseau, Yannis Stavrakas, and Michalis Vazirgiannis. 2017. Shortest-Path Graph Kernels for Document Similarity. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1890–1900.

[23] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[24] François Rousseau, Emmanouil Kiagias, and Michalis Vazirgiannis. 2015. Text categorization as a graph classification problem. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 1702–1712.

[25] François Rousseau and Michalis Vazirgiannis. 2013. Graph-of-word and TW-IDF: new approach to ad hoc IR. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM, 59–68.

[26] François Rousseau and Michalis Vazirgiannis. 2015. Main core retention on graph-of-words for single-document keyword extraction. In *European Conference on Information Retrieval*. Springer, 382–393.

[27] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2017. Modeling Relational Data with Graph Convolutional Networks. *arXiv preprint arXiv:1703.06103* (2017).

[28] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. *Proceedings of the International Conference on Learning Representations* (2015).

[29] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[30] Antoine Tixier, Fragkiskos Malliaros, and Michalis Vazirgiannis. 2016. A graph degeneracy-based approach to keyword extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 1860–1870.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 6000–6010.

[32] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph Attention Networks. *arXiv preprint arXiv:1710.10903* (2017).

[33] Tao Wang, David J Wu, Adam Coates, and Andrew Y Ng. 2012. End-to-end text recognition with convolutional neural networks. In *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 3304–3308.

[34] Jia Wu, Shirui Pan, Xingquan Zhu, Chengqi Zhang, and S Yu Philip. 2017. Multiple structure-view learning for graph classification. *IEEE transactions on neural networks and learning systems* (2017).

[35] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. 2048–2057.

[36] Fisher Yu and Vladlen Koltun. 2016. Multi-Scale Context Aggregation by Dilated Convolutions. In *Proceedings of the International Conference on Learning Representations*.

[37] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. 2335–2344.

[38] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*. 649–657.

Predicting Human Mobility via Variational Attention

Qiang Gao
University of Electronic Science and Technology of China, Chengdu, China
qianggao@std.uestc.edu.cn

Fan Zhou*
University of Electronic Science and Technology of China, Chengdu, China
fan.zhou@uestc.edu.cn

Goce Trajcevski
Iowa State University, Ames
gocet25@iastate.edu

Kunpeng Zhang
University of Maryland, College park
kpzhang@umd.edu

Ting Zhong
University of Electronic Science and Technology of China, Chengdu, China
zhongting@uestc.edu.cn

Fengli Zhang
University of Electronic Science and Technology of China, Chengdu, China
fzhang@uestc.edu.cn

ABSTRACT

An important task in Location based Social Network applications is to predict mobility – specifically, user's next point-of-interest (POI) – challenging due to the implicit feedback of footprints, sparsity of generated check-ins, and the joint impact of historical periodicity and recent check-ins. Motivated by recent success of deep variational inference, we propose VANext (Variational Attention based Next) POI prediction: a latent variable model for inferring user's next footprint, with historical mobility attention. The variational encoding captures latent features of recent mobility, followed by searching the similar historical trajectories for periodical patterns. A trajectory convolutional network is then used to learn historical mobility, significantly improving the efficiency over often used recurrent networks. A novel variational attention mechanism is proposed to exploit the periodicity of historical mobility patterns, combined with recent check-in preference to predict next POIs. We also implement a semi-supervised variant – VANext-S, which relies on variational encoding for pre-training all current trajectories in an unsupervised manner, and uses the latent variables to initialize the current trajectory learning. Experiments conducted on real-world datasets demonstrate that VANext and VANext-S outperform the state-of-the-art human mobility prediction models.

CCS CONCEPTS

• **Information systems** → **Location based services**; **Geographic information systems**.

KEYWORDS

human mobility, variational attention, convolutional neural network

ACM Reference Format:
Qiang Gao, Fan Zhou, Goce Trajcevski, Kunpeng Zhang, Ting Zhong, and Fengli Zhang. 2019. Predicting Human Mobility via Variational Attention. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313610

*Corresponding author: Fan Zhou (fan.zhou@uestc.edu.cn)

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313610

1 INTRODUCTION

Many users of Location-Based Social Networks (LBSNs) such as Foursquare and Twitter leave and/or share their footprints with friends. This has enabled new research in learning human mobility and using it in various settings – e.g., social relation inference [1], friendship prediction [24], trajectory-user linking [11, 34], etc. A critical task in understanding human mobility patterns is the next Point-of-Interest (POI) prediction – extensively studied in recent years [6, 7, 17, 27] and becomes of interest in many application domains – e.g., location sharing or recommendation for users of enterprises in social media [16]; route planing for taxi driver to maximize potential passenger pickups [27]; forecasting criminals/terrorists next location for public safety [11]; etc.

Many methods exploiting human recent mobility and their historical visits have been proposed for next POI prediction, including Matrix Factorization (MF) [16], Markov Chain (MC) [9], Factorizing Personalized Markov Chains (FPMC) [8] and Tensor Factorization (TF) [32], all incorporating human visit preferences and investigating sequential patterns. Deep learning techniques have also been applied in learning users' mobility – ST-RNN [17], POI2Vec [7], and DeepMove [6] – employing recurrent neural networks (RNN – e.g., LSTM [21] and GRU [4]) to capture the sequential patterns of users' mobility preferences. Given historical trajectories, these works train a RNN module to predict the next POI, together with corresponding contextual (e.g., spatial and temporal) features.

However, there are certain challenges for the existing works: (1) *Sparsity of recent mobility* – within a given period, a user may generate only a few check-ins (different from GPS data that is passively/continuously sampled), making the prediction difficult. (2) *Density of historical mobility* – LBSNs often collect user's check-ins for months and even years. In comparison with recent mobility, it is more difficult to represent and exploit the whole historical trajectory effectively and efficiently in a training model. (3) *Complex data* – user's historical moving patterns affect the subsequent footprints [3, 17], whereas individual check-in behaviors are complex and personalized.

To address the above challenges, we propose a novel model – Variational Attention based Next (VANext) POI prediction, leveraging the recent advances in variational Bayesian techniques [14] to encode recent mobility into a latent variable and use it to query

one's historical mobility. We also introduce a historical trajectory-based learning module implemented by a convolutional neural network (CNN) kernel – capturing individual's long-term moving patterns in a more efficient way, compared to widely used RNNs. Specifically, a variational attention on historical trajectory is combined with the latent representation learned from a recent trajectory to capture the similar check-in sequence (not necessarily continuous) which reflects user long-term visit intention. Finally, we concatenate representations of recent mobility and historical trajectory features to predict user's next POI. We also propose VANext-S – a semi-supervised variant, to better capture the latent variables by pre-training all current trajectories in an unsupervised manner within the framework of Variational Auto-Encoder (VAE) [14]. It samples from latent variables to generate and reconstruct each current trajectory, and utilizes the encoder to initialize VANext. In summary, our main contributions are:

• We frame the mobility prediction task using a generative probabilistic model together with neural networks, and propose VANext that can simultaneously learn implicit relationships between users and POIs and capture sequential patterns of user check-in behavior for next POI prediction.

• We introduce a novel variational attention mechanism to learn the attention on the historical trajectories, from which the latent representation of a recent trajectory is leveraged to match user's most similar moving patterns in the past.

• We conduct experiments on real-world datasets consisting of users' check-in behavior in four cities, demonstrating that VANext outperforms existing models in terms of both prediction accuracy and training efficiency.

To our knowledge, VANext is the first attempt to incorporate CNN to capture long term and structural dependency among user check-ins, achieving comparable learning ability with the state-of-the-art RNN based methods, while significantly improving the learning efficiency. For the rest of this paper, we first review the related work, then formalize the problem and present our main results, followed by experimental evaluations and concluding remarks.

2 RELATED WORK

POI recommendation and prediction are two different but related and extensively studied topics in LBSN: the former usually learns users' preferences over POIs while the latter is more interested in mobility pattern recognition. Collaborative filtering (CF)-based models, such as Matrix Factorization (MF) [2, 15, 16] and tensor factorization (TF) [32], are widely used in POI recommendation for learning users latent preferences. Geographical information [15, 16, 31], as well as time-aware influence and content-aware influence [10, 26, 30] are most effective features incorporated in enhancing recommendation performance [18]. However, these model-based methods are not suitable to predict user's next behavior/visits, due to the lack of taking sequential and periodical patterns into account. Markov Chains (MC)-based methods are widely used for modeling sequential influence. For example, FPMC [20] and FPMC-LR [3] aim to predict the user's next visit based on factorization of the probability transition matrix. In addition, embedding has also been applied for addressing next POI prediction [7, 8] – e.g., a Personalized Ranking Metric Embedding (PRME) [8], attempting to capture user preference with the latent space. Inspired by the word2vec [19] technique in NLP, POI2Vec [7] was proposed for next POI prediction with geographical influence included. However, both MC and embedding based methods are not capable of learning long term dependency of POIs and the periodicity of individual's historical moving.

Recently, deep learning techniques – especially recurrent neural networks (RNNs) such as LSTM [21] and GRU [4] – have been widely used to capture the long term sequential influence and mobility patterns. Spatial Temporal Recurrent Neural Networks (ST-RNN) [17] extend the RNN model by incorporating temporal and spatial context in each time unit for predicting next POIs. A unified RNN-based framework jointly learning the embeddings of multiple factors (e.g., user identity, location and time, etc.), was presented in [29]. However, these methods do not explicitly model user's historical visit patterns and personal preferences, but greatly focusing on current locations and short term dependencies among POIs. More recently, [6] propose an attentional recurrent network for mobility prediction from lengthy and sparse trajectories, where two RNN models – learning the current and the historical trajectory, respectively – together exploit user's mobility and location preference with attention on multi-level periodicity of historical trajectories. However, it is complicated to train due to the relative high density of historical trajectories.

CNNs have been widely applied in image classification and video tagging, but are less common for sequence embedding. A CNN-based architecture successfully improving the machine translation performance was recently proposed in [12]. Compared to RNN-based model, CNNs can easily capture hierarchical representations of the underlying context, and, most importantly, are faster to be trained due to parallelization with GPU. Variational Auto-Encoder (VAE) [14] has been applied in many NLP tasks, such as text classification [25] and dialogue generation [22], and has shown promising performance in exploiting underlying patterns of trajectories [34]. In addition, these models neither consider the contextual information associated with POIs, nor capture periodical mobility patterns, thereby cannot be directly employed for POI prediction.

Our proposed model VANext differs from these VAE-based methods in terms of its periodical pattern learning and the variational attention mechanism.

3 PRELIMINARIES

We now proceed with introducing the basic terminology and formalizing the problem, followed by the details of the proposed model – including causal POI embedding, variational current trajectory learning, historical trajectory attention module and the classifier.

A POI is defined as a location of relevance obtained, for example, as a GPS point; a centroid of a region; an address of an object, etc. – which can be uniquely identified in a suitable coordinate system. It is represented as a triplet $l = <id,lo,la>$, where id, lo and la denote the POI id, longitude and latitude. T^u denotes a trajectory generated by a user u, represented as a sequence $\mathrm{T}^u = (l^u_{t_1}, l^u_{t_2}, \cdots, l^u_{t_n})$, where $l^u_{t_i}$ is i-th POI visited by user u at time t_i. When there is no ambiguity, we will omit the superscript ^{u}. A trajectory T can be segmented as $\mathrm{T} = (\mathrm{T}^1, \mathrm{T}^2, \cdots, \mathrm{T}^m)$, meaning that there are m

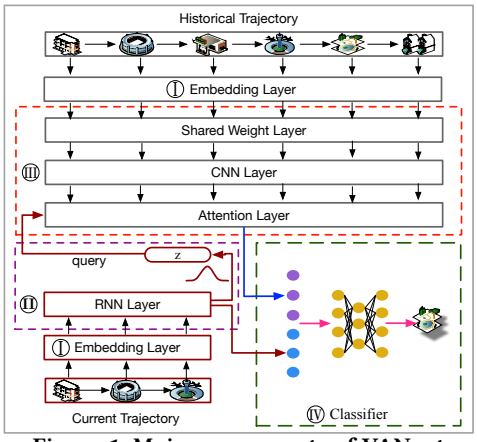


Figure 1: Main components of VANext

sub-trajectories within the time interval $[t_1, t_n]$, ordered along the temporal dimension. The trajectories are separated from each other by application dependent thresholds δ_{T}^{j} $(1 \le j \le m)$, which could indicate, for example: – the time span of each T^j is no more than δ_{T}^{j}; – the last time-stamp of T^j is at least δ_{T}^{j} time units smaller than the time-stamp of the first location in T^{j+1}; etc.

Let $\mathrm{T}_h = (\mathrm{T}^1, \cdots, \mathrm{T}^m)$ denote the entire historical trajectory of the user u (note that its time span may vary, from weeks, to months or even years for different users) and let $\mathrm{T}_c = \mathrm{T}^{m+1} = (l_{t_{n+1}}, l_{t_{n+2}}, \cdots, l_{t_{n+k-1}})$ denote the current/most recent sub-trajectory.

Problem formulation: We consider next POI prediction as a multi-classification problem, which is formulated as: given a user u with the historical trajectory T_h, and the recently visited sequence of POIs $\mathrm{T}_c = (l_{t_{n+1}}, l_{t_{n+2}}, \cdots, l_{t_{n+k-1}})$, train a model M to predict the next POI $\gamma = l_{t_{n+k}}$ for user u – i.e., $M(\mathrm{T}_h, \mathrm{T}_c) \mapsto \gamma$.

VAE [14] is a generative model that learns the probability $p(X|\mathbf{z})$ of the data X (e.g., a trajectory) given a latent variable $\mathbf{z}$, and a recognition model $q(\mathbf{z}|X)$ simultaneously. The lower bound of the true marginal log likelihood is:

$$\log p_\theta(X) \ge \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|X)}[\log p_\theta(X|\mathbf{z})] - \mathcal{KL}(q_\phi(\mathbf{z}|X)||p(\mathbf{z}))$$

where both $q_\phi(\cdot)$ and $p_\theta(\cdot)$ are learned with deep neural networks, parameterized by ϕ and θ, respectively. $\mathcal{KL}$ is the KL-divergence.

4 METHODOLOGY

4.1 Basic Framework

Figure 1 shows the four main modules of our proposed model VANext:

(I) *Causal POI embedding*: To better present the semantic relationships among different POIs, we first embed POIs, from both current trajectory and historical trajectory, into a low dimensional representation using a causal embedding method.

(II) *Current Trajectory Learning Module*: We apply a recurrent neural network (RNN) – in this paper, we choose the GRU cell – to encode the current trajectory T_c, and use the last hidden state to represent it , and further learns the distribution of a latent variable $\mathbf{z}$ using variational encoding as $q_\phi(\mathbf{z}|\mathrm{T}_c)$.

(III) *Historic Trajectory Learning Module*: VANext utilizes a shared weight matrix and a convolutional layer to encode the historical trajectory T_h. The attention mechanism with latent variable $\mathbf{z}$ is considered as a query to exploit users' (historical) mobility patterns. It seeks the most similar mobility to the current trajectory T_c by searching the historical trajectory T_h, and aggregates its hidden state from T_h to form the representation. The formalized representation of aggregated result is $\tilde{\mathrm{T}}_h$. In other words, attention module will force the historical trajectory representation $\tilde{\mathrm{T}}_h$ to automatically match T_c.

(IV) *Classifier*: This is the final component which unifies the last hidden state from current trajectory and the aggregated attention from historical trajectory into a feature representation, and predicts the next POI γ.

4.2 Trajectory Generative Model

Before presenting the details of each component, we first adapt our VANext into a trajectory generative model. The primary task of VANext is to predict γ from T_c. Hence, the likelihood can be specified as:

$$\begin{aligned}\log p_\theta(\gamma|\mathrm{T}_c) &= \log \int_{\mathbf{z}} p_\theta(\gamma|\mathrm{T}_c, \mathbf{z}) p(\mathbf{z}) d\mathbf{z} \\ &= \log \int_{\mathbf{z}} q_\phi(\mathbf{z}|\mathrm{T}_c, \gamma) \frac{p_\theta(\gamma|\mathrm{T}_c, \mathbf{z})}{q_\phi(\mathbf{z}|\mathrm{T}_c, \gamma)} p(\mathbf{z}) d\mathbf{z},\end{aligned}$$

With Jensen's inequality, we have

$$\begin{aligned}&\log p_\theta(\gamma|\mathrm{T}_c) \ge \mathbb{E}_{q_\phi(\mathbf{z})}[\log p_\theta(\gamma|\mathrm{T}_c, \mathbf{z}) + \log \frac{p(\mathbf{z})}{q_\phi(\mathbf{z}|\gamma, \mathrm{T}_c)}] \\ &= \mathbb{E}_{q_\phi(\mathbf{z})} p_\theta(\gamma|\mathrm{T}_c, \mathbf{z}) - \mathcal{KL}(q_\phi(\mathbf{z}|\gamma, \mathrm{T}_c)||p(\mathbf{z})). \quad (1)\end{aligned}$$

Since γ is a function of input T_c, we can absorb γ into T_c following [33] and therefore obtain the following variant of Eq.(1):

$$\log p_\theta(\gamma|\mathrm{T}_c) \ge \mathbb{E}_{q_\phi(\mathbf{z})} p_\theta(\gamma|\mathrm{T}_c, \mathbf{z}) - \mathcal{KL}(q_\phi(\mathbf{z}|\mathrm{T}_c)||p(\mathbf{z})). \quad (2)$$

In VaNext, we propose a variational attention mechanism to obtain an attentional representation $\tilde{\mathrm{T}}_h$. In detail, we use the latent variable $\mathbf{z}$ as a query vector to find the most similar mobility $\tilde{\mathrm{T}}_h$ from T_h. Since $\mathbf{z}$ determines $\tilde{\mathrm{T}}_h$, hence we obtain our training objective that is to maximize the lower bound of likelihood $L(\theta, \phi)$:

$$L(\theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z})}[\log p_\theta(\gamma|\mathrm{T}_c, \tilde{\mathrm{T}}_h)] - \lambda \mathcal{KL}[q_\phi(\mathbf{z}|\mathrm{T}_c)||p(\mathbf{z})], \quad (3)$$

where λ is harmonious factor.

4.3 Causal POI Embeddings

Inspired by word2vec [19], previous works [6, 11, 34] embed POIs with the context information using CBOW or Skip-Gram models. However, for the next POI prediction problem, we are more interested in embedding the proceeding part of a trajectory instead of the surrounding context – the rationale behind is that the current POI is determined by its previous footprint rather than the ones behind it, which is very similar to the idea of high-order Markov Process. That is, there can be no leakage from the future into the past when embedding the POIs. This can be accomplished with a *causal* embedding where the probability of check-in l_τ is maximized only with the given elements from earlier footprints $l_{\tau-\omega} : l_{\tau-1}$, where ω is the fixed size of the sliding window.

Specifically, the check-in representations $\mathbf{v} \in \mathbb{R}^{|\mathcal{L}| \times d}$ – $|\mathcal{L}|$ is the number of check-ins in the dataset and d is the dimensionality – are obtained by predicting each current location l_τ in both T_h

and T_c given the proceeding context locations $l_{\tau-\omega} : l_{\tau-1}$. The probability $p(l_\tau|C(l_\tau))$ is defined by the *softmax* function as:

$$p(l_\tau|C(l_\tau)) = \prod_{l' \in C(l_\tau)} \frac{\exp(\mathbf{v}(l_\tau) \cdot \mathbf{v}(l'))}{\sum_{l'' \in \mathcal{L}} \exp(\mathbf{v}(l'') \cdot \mathbf{v}(l'))}$$

where we adopt the *Negative Sampling* technique to avoid the enumeration of all check-ins $l'' \in \mathcal{L}$ for the efficiency purpose.

4.4 Current Trajectory Learning Module

This module processes the current trajectory T_c with a recurrent network and meanwhile learns a distribution of a latent variable $\mathbf{z}$.

4.4.1 – Current trajectory with GRU:. To capture users' current sequential check-in pattern, we use a vanilla GRU to exploit the mobility of T_c. Note that user's recent trajectories are usually shorter sequence compared to the extremely long historical trajectory, i.e., $|T_c| \ll |T_h|$. Therefore, we use a recurrent network instead of convolutional operation that would be employed in the historical module introduced in next section. Moreover, to make a well-targeted comparison with previous work [6], we also select GRU kernel as the neural network unit:

$$h_\tau = (1 - g_\tau)h_{\tau-1} + g_\tau \tilde{h}_\tau \tag{4}$$

where g_τ is the update gate in time τ which decides how much the unit updates its activation by

$$g_\tau = \sigma(W_g \mathbf{v}(l_\tau) + U_g h_{\tau-1}), \tag{5}$$

where σ denotes the sigmoid activation function. The candidate state $\tilde{h}_t$ is computed similarly to traditional RNN unit

$$\tilde{h}_\tau = \tanh(W_{\tilde{h}} \mathbf{v}(l_\tau) + U_{\tilde{h}}(s_\tau \odot h_{\tau-1})) \tag{6}$$

where $\odot$ is element-pair product, s_τ is a set of reset gates and is computed similarly to update the gate

$$s_\tau = \sigma(W_s \mathbf{v}(l_\tau) + U_s h_{\tau-1}) \tag{7}$$

where W_* and U_* are both parameterized matrices.

4.4.2 – Latent variable learning: Variational auto-encoding provides an efficient way to approximate the posterior distribution of latent variables. In our trajectory generative model, we sample the empirical posterior $q_\phi(\mathbf{z}|T_c)$ of latent variables $\mathbf{z}$ from the last hidden state of current trajectories. The objective is to minimize the KL-divergence between posterior and prior as shown in Eq.(3). Typically, we assume the prior distribution of latent variable $\mathbf{z}$ following a standard Gaussian: $p(\mathbf{z}) \sim N(0, I)$.

4.5 Historical Trajectory Learning Module

This module involves the convolutions on historical trajectories and the variational attention layer.

4.5.1 – Historical trajectory convolution: Before applying the convolution operation over the historical trajectories, we first use a shared weight layer (e.g, a MLP) to incorporate each POI representation vector into a hidden state:

$$c_{t_i} = \text{ReLU}(W_h(\mathbf{v}(l_{t_i})) + b_h), i \in [1, 2, \cdots, n] \tag{8}$$

where ReLU is the activation function. Then, we obtain all hidden states of embedded POIs as: $c_t = (c_{t_1}, c_{t_2}, \cdots, c_{t_n})$.

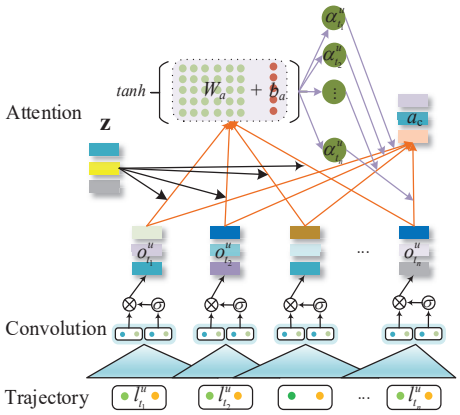


Figure 2: Attention mechanism in Historical Trajectory Module.

Subsequently, a 1×D convolution with non-linearity neural network is involved to exploit the hidden state patterns. The convolution kernel is parameterized with matrix $W_c \in \mathbb{R}^{2d \times nd}$ and bias $b_c \in \mathbb{R}^{2d}$, where n is the number of the hidden states. Therefore, each input hidden state $c_{t_i} \in \mathbb{R}^d$ will generate an output $O_i \in \mathbb{R}^{2d}$ by a CNN. By splitting each element O_i into O_{i_1} and O_{i_2} (both $\in \mathbb{R}^d$), the final output o_{t_i} will be obtained by activating O_{i_1} and O_{i_2} to a non-linearity with gated linear units, as follows:

$$o_{t_i} = O_{i_1} \odot \sigma(O_{i_2}) \tag{9}$$

Thus, the convolutional state $o_t = (o_{t_1}, o_{t_2}, \cdots, o_{t_n})$ is obtained as the input of the next attention layer.

4.5.2 – Variational attention. As the Figure 2 shows, we present a variational attention method to learn the weight of convolutional states in historical trajectories. We first use a MLP to obtain the query vector $\mathbf{q}$ from $\mathbf{z}$:

$$\mathbf{q} = \text{ReLU}(\mathbf{z}W_z + b_z) \tag{10}$$

which is then used as a query to calculate the similarity score between $\mathbf{q}$ and o_t as

$$u_{t_i} = \tanh(\mathbf{q}(W_u o_{t_i} + b_u)) \tag{11}$$

We next get a normalized importance weight α_i with a softmax function.

$$\alpha_i = \frac{\exp(u_{t_i}^\top u_w)}{\sum_{i=1}^n \exp(u_{t_i}^\top u_w)} \tag{12}$$

where u_w is a vector variable learned during the training process.

Now we obtain the representation of historical trajectory as a weighted sum of the POIs.

$$\tilde{a} = \sum \alpha_i o_{t_i} \tag{13}$$

Note that above attention method is different from traditional attention methods in that the learned mobility pattern representation not only depends on the check-ins of historical trajectory but also considers the variational query variable $\mathbf{z}$ from current trajectory. This variational attention technique allows us to sample from a continuous distribution $q_\phi(\mathbf{z}|T_c)$ rather than from discrete/sparse current trajectory T_c, and thus is able to better match the historical trajectory. As we will show in experiments, this novel variational attention method introduces significant performance

improvement compared to the traditional vanilla attention used in previous work [6].

Next POI prediction To predict the next POI of T_c, we first concatenate the last hidden state of current trajectory and the attentional historical trajectory to obtain Φ as follows:

$$\Phi = W_e * \begin{bmatrix} h_{n+k-1} \\ \tilde{a} \end{bmatrix} + b_e \tag{14}$$

Then, we feed Φ into a softmax function to calculate the probability of each POI $p(l)$ in the dataset. During the training of VANext, we are interested in minimizing the empirical risk of predicting the next POI γ.

Semi-supervised VANext: VANext-S We propose a variant of VANext, called VANext-S, which utilizes VAE to pretrain all current trajectories in an unsupervised manner, and uses the encoder and latent variables to initiate the current trajectory learning.

5 EXPERIMENTS

In this section, we conduct experiments on two real-world datasets to compare the performance of the proposed models against several baselines.

Datasets. To ease reproducing the results, we conduct all experiments on the publicly available datasets including Foursquare and Gowalla [7, 28, 30]. We select two cities (New York, Singapore) from Foursquare, and two cities (Houston, California) from Gowalla. For each dataset, we remove the POIs which have been visited by fewer than 5 users. For each user, we concatenate all check-in locations to form a single trajectory – subsequently, we divide it into sub-trajectories with the time interval of 6 hours each, as it was done in previous related works [7, 11, 17]. Further, we filter out the users who have fewer than 5 sub-trajectories. For all datasets, we choose each user's first 80% sub-trajectories as the training set, and the remaining 20% as testing data. Table 1 summarizes the statistics of the datasets after pre-processing.

Table 1: Descriptive statistics of Datasets.

City	Users	POIs	Check-ins	Sub-trajectories
New York	1,083	9,815	120,007	36,182
Singapore	2,321	5,596	194,108	34,713
Houston	4,627	15,234	362,783	18,501
California	3,987	21,354	239,493	66,612

Table 2: Experimental settings of VANext and VANext-S.

Parameter	Values	Model Setting	Chooses
window size	5	embedding size	256
HS/NS	NS	hidden size	300
learning rate	1e-3	attention size	300
dropout rate	0.5	$\|\mathbf{z}\|$ size	128
λ	0.01	batch size	16

Settings. Table 2 characterizes the specific settings and hyper-parameters when implementing our VANext and VANext-S models. In particular, we use negative sampling (NS) instead of Hierarchical Softmax (HS) for our causal POI embedding. We implement two models, as well as all deep neural network-based models, on the Tensorflow platform and speed up using a GTX1080ti GPU.

Metrics. To make fair comparisons, we use the standard evaluation performance metrics, such as **Top@k**, Area under the ROC curve (**AUC**) and Mean Average Precision (**MAP**) used in [6, 17].

Baselines. We compare our models with several classical methods and the most recently proposed methods regarding next POI prediction, including:

• **Markov chain** [9] – Markov chain is one of the classical methods which predict the POIs using the estimated transition probability.

• **PRME** [8] – Personalized ranking metric embedding is a pairwise metric embedding method embedding every POI into a latent Euclidean space and computing the location transition in a Markov chain model.

• **ST-RNN** [17] – It is a deep learning model that incorporates spatial and temporal context and predicts user's next visit within the framework of RNN.

• **Bi-LSTM** [26] – It is a method extending ST-RNN with a more sophisticated Bidirectional-LSTM model that has been used in [26] for learning individual trajectory patterns and predicting POIs.

• **POI2vec**[7] – It is an embedding method which incorporates the geographical influence for next POI prediction based on word2vec technique.

• **DeepMove**[6] – The first historical attention method for learning human mobility, consists of a sequential encoding module with RNN for learning motion patterns from both recent and historical trajectories.

Experimental Results

Overall performance comparison. Table 3 shows the **Top@k** performance comparison on four citywide datasets. As we can see, overall, the proposed VANext and VANext-S yield the best performance among all methods in terms of prediction precision. Specifically, VANext and VANext-S outperform the second-best method DeepMove by 31.52%, 36.81% and 32.68%, in terms of *Top@1*, *Top@5* and *Top@10* respectively on Singapore dataset.

As for the performance of baselines, deep neural network-based methods (i.e., ST-RNN, Bi-LSTM, POI2vec and DeepMove) exhibit higher prediction accuracy than traditional feature-based embedding method and Markov-based methods. Furthermore, DeepMove generally gives better performance than other baselines due to its ability of exploiting historical trajectories while others can only learn the sequential patterns in (short) current trajectories during training. Although all of these methods have employed recurrent networks in order to capture long term POI dependency, even LSTM and GRU cannot work well on very long historical trajectories due to the gradient vanishing problem inherent in recurrent networks. This result is in accordance with the limitations of applying RNN in natural language processing [5, 13]. Therefore, the main advantage of DeepMove lies in its attention on historical trajectories, which also confirms the superiority of attention mechanism in processing sequential data, e.g., attention mechanism along achieves comparable results on machine translation with CNN or RNN based seq2seq methods [23].

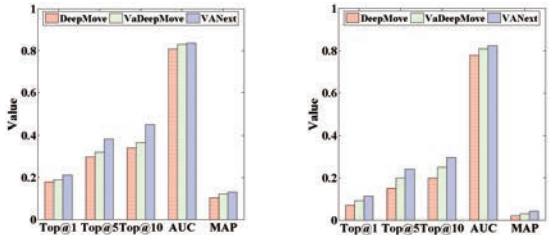
On the other hand, our methods outperform DeepMove mainly because the introduced trajectory generative model which not only captures short-term human mobility patterns more effectively but also yields better historical trajectory attention with the novel variational attention method.

Additionally, VANext-S outperforms VANext because of its pre-training on the current trajectories in an unsupervised manner. Although autoencoder based pretraining is a well-known training

Table 3: Comparison of overall prediction accuracy on four datasets. The best method is shown in bold, and the second best is shown as underlined

Method	New York			Singapore			Houston			California		
	Top@1	Top@5	Top@10	Top@1	Top@5	Top@10	Top@1	Top@5	Top@10	Top@1	Top@5	Top@10
Markov	18.29%	28.01%	29.59%	6.57%	13.36%	15.81%	6.32%	11.57%	13.12%	5.35%	10.10%	11.71%
PRME	12.83%	24.38%	19.66%	2.36%	6.27%	8.29%	2.36%	6.27%	8.29%	1.57%	4.10%	5.91%
ST-RNN	18.64%	28.01%	30.23%	6.64%	15.01%	18.34%	6.32%	11.38%	13.37%	5.23%	10.30%	11.94%
Bi-LSTM	18.42%	28.85%	31.27%	6.19%	13.67%	17.58%	5.65%	11.14%	13.67%	5.01%	10.15%	11.68%
POI2Vec	18.79%	28.85%	30.81%	6.89%	14.87%	18.27%	6.32%	11.21%	13.52%	4.64%	8.77%	10.23%
DeepMove	19.29%	35.37%	38.35%	8.82%	17.93%	22.80%	6.65%	13.43%	15.87%	6.62%	13.79%	17.15%
VANext	21.09%	38.22%	44.88%	11.60%	24.53%	30.25%	7.29%	15.78%	20.21%	7.34%	14.71%	17.78%
VANext-S	**23.21%**	**41.03%**	**47.45%**	**12.13%**	**24.78%**	**30.69%**	**7.69%**	**16.98%**	**21.36%**	**8.35%**	**17.16%**	**20.31%**

trick in NLP, we are the first to adapt it in a variational autoencoder framework for learning human mobility. Since this is not core part of this paper, we focus on the VANext in following experiments for further explanations.



(a) New York. (b) Singapore.

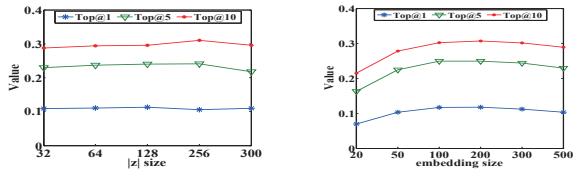
Figure 3: Module performance comparison.

Individual Module Performance. To investigate the effectiveness of various components in VANext, we conduct experiments by manipulating modules in different ways. Specifically, we replace the attention mechanism of *DeepMove* with our proposed variational attention module, yielding a new method *VaDeepMove*. Thus, VaDeepMove is similar to VANext only except the historical trajectory learning – VANext uses CNN while GRU units are employed in VaDeepMove. We compare our VANext with VaDeepMove and DeepMove upon New York and Singapore datasets, as illustrated in Figure 3(a) and 3(b). The result clearly indicates the superiority of CNN and variational attention used in VANext. Further, it proves that CNN exhibits better structural learning than RNN in the case of extremely long history trajectories which encodes more structural and periodical patterns of human mobility. The results of VaDeepMove and DeepMove reflect the advantage of variational attention which is also their difference.

Efficiency. We now investigate the efficiency of the proposed methods. Table 4 shows the time consumed when training VANext (CNN units) and VaDeepMove (RNN units). Note that both VANext and (Va)DeepMove use RNN for training current trajectory T_c. However, it is negligible due to few POIs in most current trajectory. Obviously, CNN based historical trajectory learning significantly reduces the training time. Combined with above results on learning structural/periodical patterns of historical trajectories, we conclude that, arguably, CNN is the most suitable method for learning human mobility, at least for long term motion patterns.

Table 4: Comparison of time consumption during the model training. $|T_c|$: average length of the current trajectories; $|T_h|$: average length of the historical trajectories.

City in Dataset	$\|T_c\|$	$\|T_h\|$	Cell	Time Cost(min)
New York	3.23	207.70	RNN CNN	≈487 ≈277
Singapore	2.92	119.80	RNN CNN	≈267 ≈117
Houston	9.53	1027.94	RNN CNN	≈1810 ≈463
California	3.64	129.66	RNN CNN	≈790 ≈247



(a) Sensitivity of $|\mathbf{z}|$ size. (b) Sensitivity of embedding.

Figure 4: Sensitivity of parameters in Singapore data.

Parameter sensitivity. Finally, Figure 4(a) evaluates impact of latent variable size $|\mathbf{z}|$ and causal POI embedding size d on the performance of VANext. Simply speaking, VANext demonstrates robust performance to the two parameters and achieves stable results when both are small. We observe similar robust results for other parameters and omit to report due to the space limitation.

6 CONCLUSIONS AND FUTURE WORK

We propose a novel model VANext and its variant VANext-S to learn human mobility patterns for next POI prediction. It introduces an attention mechanism affected by the latent variable learned from current mobility to capture the moving pattern and hierarchical semantics of historical trajectories. We also apply convolutional operations on trajectory to greatly improve efficiency over the existing RNN-based methods. Part of our future work will focus on incorporating other contextual POIs attributes (e.g., "museum"; "restaurant") to further improve VANext effectiveness. Another possible extension is to include transportation modes (e.g., "walk"; "bus"; "taxi") of users trajectories for performance boosting.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China (Grant No.61602097 and No.61472064), NSF grants III 1213038 and CNS 1646107, and ONR grant N00014-14-10215.

REFERENCES

[1] Michael Backes, Mathias Humbert, Jun Pang, and Yang Zhang. 2017. walk2friends: Inferring Social Links from Mobility Profiles. In *SIGSAC*.
[2] Chen Cheng, Haiqin Yang, Irwin King, and Michael R Lyu. 2012. Fused Matrix Factorization with Geographical and Social Influence in Location-Based Social Networks.. In *AAAI*.
[3] Chen Cheng, Haiqin Yang, Michael R Lyu, and Irwin King. 2013. Where You Like to Go Next: Successive Point-of-Interest Recommendation.. In *IJCAI*.
[4] Junyoung Chung, Caglar Gulcehre, Kyung Hyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *Eprint Arxiv* (2014).
[5] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. 2015. A Recurrent Latent Variable Model for Sequential Data. In *NIPS*.
[6] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. DeepMove: Predicting Human Mobility with Attentional Recurrent Networks. In *WWW*.
[7] Shanshan Feng, Gao Cong, Bo An, and Yeow Meng Chee. 2017. POI2Vec: Geographical Latent Representation for Predicting Future Visitors.. In *AAAI*.
[8] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized Ranking Metric Embedding for Next New POI Recommendation.. In *IJCAI*.
[9] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. 2012. Next Place Prediction Using Mobility Markov Chains. In *MPM*.
[10] Huiji Gao, Jiliang Tang, Xia Hu, and Huan Liu. 2015. Content-aware Point of Interest Recommendation on Location-based Social Networks. In *AAAI*.
[11] Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Xucheng Luo, and Fengli Zhang. 2017. Identifying human mobility via trajectory embeddings. In *IJCAI*.
[12] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *ICML*.
[13] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward Controlled Generation of Text. In *ICML*.
[14] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).
[15] Xutao Li, Gao Cong, Xiao-Li Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. 2015. Rank-GeoFM: A Ranking Based Geographical Factorization Method for Point of Interest Recommendation. In *SIGIR*.
[16] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. GeoMF: Joint Geographical Modeling and Matrix Factorization for Point-of-interest Recommendation. In *KDD*.
[17] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the Next Location : A Recurrent Model with Spatial and Temporal Contexts. In *AAAI*.
[18] Yiding Liu, Tuan-Anh Pham, Gao Cong, and Quan Yuan. 2017. An Experimental Evaluation of Point-of-interest Recommendation in Location-based Social Networks. *PVLDB* (2017).
[19] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[20] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing Personalized Markov Chains for Next-basket Recommendation. In *WWW*.
[21] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.
[22] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *AAAI*.
[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*.
[24] Hongjian Wang, Zhenhui Li, and Wang Chien Lee. 2015. PGT: Measuring Mobility Relationship Using Personal, Global and Temporal Factors. In *ICDM*.
[25] Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. 2017. Variational Autoencoder for Semi-Supervised Text Classification. In *AAAI*.
[26] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. 2017. Bridging Collaborative Filtering and Semi-Supervised Learning: A Neural Approach for POI Recommendation. In *KDD*.
[27] Cheng Yang, Maosong Sun, Wayne Xin Zhao, Zhiyuan Liu, and Edward Y. Chan. 2017. A Neural Network Approach to Jointly Modeling Social Networks and Mobile Trajectories. *ACM Trans. Inf. Syst.* 35, 4 (2017), 36:1–36:28.
[28] Dingqi Yang, Daqing Zhang, Vincent W Zheng, and Zhiyong Yu. 2015. Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2015), 129–142.
[29] Di Yao, Chao Zhang, Jianhui Huang, and Jingping Bi. 2017. SERM: A Recurrent Model for Next Location Prediction in Semantic Trajectories. In *CIKM*.
[30] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. 2013. Time-aware point-of-interest recommendation. In *SIGIR*.
[31] Shenglin Zhao, Tong Zhao, Irwin King, and Michael R Lyu. 2017. Geo-Teaser: Geo-Temporal Sequential Embedding Rank for Point-of-interest Recommendation. In *WWW*.
[32] Vincent Wenchen Zheng, Bin Cao, Yu Zheng, Xing Xie, and Qiang Yang. 2010. Collaborative Filtering Meets Mobile Recommendation: A User-centered Approach. In *AAAI*.
[33] Chunting Zhou and Graham Neubig. 2017. Morphological Inflection Generation with Multi-space Variational Encoder-Decoders. In *CoNLL*.
[34] Fan Zhou, Qiang Gao, Kunpeng Zhang, Goce Trajcevski, Zhong Ting, and Fengli Zhang. 2018. Trajectory-User Linking via Variational AutoEncoder. In *IJCAI*.

Maximizing Marginal Utility per Dollar for Economic Recommendation

Yingqiang Ge
Rutgers University
yingqiang.ge@rutgers.edu

Shuyuan Xu
Rutgers University
shuyuan.xu@rutgers.edu

Shuchang Liu
Rutgers University
shuchang.liu@rutgers.edu

Shijie Geng
Rutgers University
shijie.geng@rutgers.edu

Zuohui Fu
Rutgers University
zuohui.fu@rutgers.edu

Yongfeng Zhang
Rutgers University
yongfeng.zhang@rutgers.edu

ABSTRACT

Understanding the economic nature of consumer decisions in e-Commerce is important to personalized recommendation systems. Established economic theories claim that informed consumers always attempt to maximize their utility by choosing the items of the largest *marginal utility per dollar* (MUD) within their budgets. For example, gaining 5 dollars of extra benefit by spending 10 dollars makes a consumer much more satisfied than having the same amount of extra benefit by spending 20 dollars, although the second product may have higher absolute utility value. Meanwhile, making purchases online may be risky decisions that could cause dissatisfaction. For example, people may give low ratings towards purchased items that they thought they would like when placing the order. Therefore, the design of recommender systems should also take users' risk attitudes into consideration to better learn consumer behaviors.

Motivated by the first consideration, in this paper, we propose a learning algorithm to maximize marginal utility per dollar for recommendations. With the second, economic theory shows that rational people can be arbitrarily close to risk neutral when stakes are arbitrarily small, and this is generally applicable to consumer online purchase behaviors because most people spend a small portion of their total wealth for a single purchase. To integrate this theory with machine learning, we propose to augment MUD optimization with approximate risk-neural constraint to generate personalized recommendations. Experiments on real-world e-Commerce datasets show that our approach is able to achieve better performance than many classical recommendation methods, in terms of both traditional recommendation measures such as precision and recall, as well as economic measures such as MUD.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Applied computing** → **Economics**; • **Computing methodologies** → *Intelligent agents.*

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313725

KEYWORDS

Recommendation Systems; Personalization; Risk Attitude; Marginal Utility per Dollar; Computational Economics

ACM Reference Format:
Yingqiang Ge, Shuyuan Xu, Shuchang Liu, Shijie Geng, Zuohui Fu, and Yongfeng Zhang. 2019. Maximizing Marginal Utility per Dollar for Economic Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313725

1 INTRODUCTION

Due to the significant information overload in the Web environment, designing personalized recommendation systems is important to help users find relevant items efficiently. On one hand, it helps to save the exploration time for consumers, and on the other hand, it helps to improve the revenue of various online economic platforms. Traditional recommendation algorithms mostly focus on optimizing rating- or ranking-oriented measures. For example, rating prediction algorithms such as matrix factorization [10, 12, 23] or neural approaches [27] devise models to optimize for the prediction accuracy in terms of RMSE, while top-N recommendation algorithms such as pair-wise learning to rank [7, 20, 28] propose models to optimize for ranking performance in terms of Precision, Recall, F1 and NDCG, etc.

However, previous recommendation algorithms seldom consider users' economic incentives when modeling user behaviors and generating recommendations. Actually, consumers' economic incentives play an important role when making decisions in online economic systems such as e-Commerce. By integrating well-established behavioral economic principles and machine learning algorithms, it is possible to develop economics-driven recommender systems so as to make informed recommendations. Therefore, in this paper, we propose to maximize the marginal utility per dollar for economic recommendation. According to behavioral economic theories, rational consumers would always attempt to maximize their utility out of the purchased products by choosing the items that have the largest Marginal Utility per Dollar within their budget. The underlying intuition is that consumers would like to spend their money in an efficient way so that every spent dollar should bring the maximum marginal utility. Meanwhile, making purchasing decisions online involves potential risks of dissatisfaction. A common observation in practical systems is that people may spend money to purchase an item that they think will match their preference when placing the order, but they may eventually find themselves unsatisfied with

the decision when the item is received, and will thus make lower ratings towards the item. As a result, an informed recommendation system should be able to incorporate risk attitude into the model. Fortunately, when integrated with machine learning, established economic principles can help to model the risk attitudes of user decisions based on large-scale user transaction logs.

In this work, we introduce users' risk attitudes into the economic modeling of recommendation systems. In particular, we simulate the risk distribution of each item based on its rating distribution, and by jointly optimizing the marginal utility per dollar under approximate risk neutral constraint, our model learns to predict the consumer decisions with risk-awareness. Experiments based on real-world e-Commerce datasets verify that our approach not only achieves better recommendation performance in terms of precision and recall than both shallow and deep baselines, but also achieves better economic performance in terms of the marginal utility per dollar, which means that our recommendation results would help consumers to spend their money in a more efficient way.

The key contributions of the paper can be summarized as follows:

- We propose to conduct marginal utility per dollar maximization for economic recommendation, which is consistent with established behavioral economic principles, and helps users to spend their money more efficiently.
- We take consumers' risk-attitude into consideration for economic recommendation, which better simulates the real-world online economic environment, where users have to make decisions under potential risks of dissatisfaction.
- We design a joint learning framework to maximize the utility per dollar under risk constraints. Experimental results on several real-world e-commerce datasets show that our approach not only achieves better performance than both shallow and neural recommendation baselines, but also improves the economic measures in terms of money efficiency.

The following parts of the paper will be organized as follows: we review the related work in section 2, and then introduce the necessary economics preliminary knowledge in section 3 to help readers better understand the background. The proposed model and recommendation strategy are introduced in section 4, followed by the experimental results in section 5. We finally conclude the work with possible future research directions in section 6.

2 RELATED WORK

2.1 Collaborative Filtering

Collaborative Filtering (CF) has been an important approach to recommender systems. Early approaches to CF consider the user-item rating matrix and conduct rating prediction with user-based [9, 21] or item-based [14, 24] collaborative filtering methods. With the development of dimension reduction methods, latent factor models such as matrix factorization are later widely adopted in recommender systems, such as singular value decomposition [11], non-negative matrix factorization [13], probabilistic matrix factorization [16], localized matrix factorization [29], etc. In these approaches, each user and item is learned as a latent factor representation to calculate the matching score of the user-item pairs.

Recently, the development of deep learning and neural networks has further extended collaborative filtering methods for recommendation. The relevant methods can be broadly classified into two sub-categories: similarity learning approach, and representation learning approach. The similarity learning approach adopts simple user/item representations (such as one-hot) and learns a complex prediction network as the similarity function to calculate user-item matching scores [8], while the representation learning approach learns rich user/item representations and adopts a simple similarity function (e.g., inner product) for matching score calculation [28].

Another important research direction is learning to rank for recommendation, which learns the relative ordering of items instead of the absolute preference scores. The most representative method on this direction is perhaps Bayesian personalized ranking [20], which is a pair-wise learning to rank method. It is also further generalized to take other information sources such as images [7].

2.2 Economic Recommendation

For a long time, recommendation system research has been working on the above mentioned rating- or ranking-related tasks such as rating prediction and top-N recommendation. However, the related methods seldom consider the economic value that a recommendation list brings to the user or the system, although this is one of the most important goals for real-world recommendation systems. Some recent research on economic recommendation has begun to take care of the economic value of personalized recommendation. For example, [26] studied user's sense of value in terms of utility in recommender systems, and [31] conducted large-scale experiments with real-world users to validate the consumer sense of utility for personalized promotion. [30] further bridged economic principles and machine learning to maximize the social surplus for recommendation, [32] proposed to learn the substitutive and complementary relations between products for utility maximization, and [18] proposed value-aware recommendation for profit maximization based on reinforcement learning. Although current economic recommendation approaches may improve the economic value, their basic motivation is to maximize a total utility function for each user to generate recommendations. However, established behavioral economic principles show that consumers tend to rely on the marginal utility per dollar to make purchasing decisions so as to improve their money efficiency [4, 22], which motivates us to estimate and maximize the marginal utility per dollar for economics-driven recommender systems.

3 PRELIMINARIES

3.1 Utility and Marginal Utility

Utility is an economic measure to quantify consumer's pleasure or satisfaction towards some items. Basically, it is widely used to analyze human behavior in rational choice theory [3]. Utility is usually a function of the consumption quantity Q, which measures the total utility of consuming Q services of a certain product. Among the many different forms of utility functions, the most fundamental and frequently used one is the King-Plosser-Rebelo (KPR) Utility:

$$U_{KPR}(x) = a_x \ln(1 + Q_x), \tag{1}$$

where Q_x is the consumption quantity of item x, and a_x is a scale

parameter that could be different for diffident users, which means that people may have different utility even for the same amount of the same product, because of their different personalized preference.

Marginal Utility (MU) is the increment of utility gained from purchasing one extra unit of a product. It can be defined as the first derivative of the utility function, which is

$$MU_{KPR}(x) = \frac{a_x}{1 + Q_x}. \tag{2}$$

3.2 Expected Utility and Utility of Expectation

Expected utility, also known as the Von Neumann-Morgenstern (VNM) utility, is a basic concept in economics, game theory and decision theory to measure user preference in uncertain circumstances. When an individual has to make a decision under uncertainty, it is rational to make a choice with the highest expected utility [5, 25].

Assuming that U is the utility function for random variable X, then the expected utility (EU) is $EU(X) = \sum_{x \in X} U(x)P(x)$. Meanwhile, the utility of expectation is the utility of the expected value of random variable X, which is $U(EX) = U\Big(\sum_{x \in X} xP(x)\Big)$. In this paper, we use the rating distribution of an item to estimate the probability distribution $P(x)$ for the item.

3.3 Marginal Utility per Dollar (MUD)

In rational choice theory, the condition where a consumer's total utility reaches its maximum is called the *consumer equilibrium*, which means that the consumer would not purchase either more or less products. The equilibrium condition can be defined as follows,

$$\Big\{ p_1 Q_1 + p_2 Q_2 + \cdots + p_n Q_n = I \tag{3}$$

$$\begin{cases} \frac{MU_1}{p_1} = \frac{MU_2}{p_2} = \cdots = \frac{MU_n}{p_n} = \lambda \end{cases} \tag{4}$$

where Q_i is the purchasing quantity of product i; p_i is the price of i; I is the budget (or 'income' in economics); MU_i is the marginal utility of i; λ is called the *marginal utility of money*, which is a constant value at the equilibrium [17, 22]. Based on the above KPR utility function, we can also get the functional form of marginal utility per dollar (MUD) as follows,

$$MUD_{KPR}(x) = \frac{a_x}{(1 + Q_x)p_x} \tag{5}$$

Intuitively, rational choice and consumer equilibrium mean that when deciding which item to buy, users will always choose the one that makes the best use of their money. However, after an item is purchased, its marginal utility (and thus MUD) will diminish, which means that the next optimal choice could be another item. This process will iterate until equilibrium, where MUDs of all items are at the same level that equals the marginal utility of money λ. More intuitively, at the consumer equilibrium condition, the marginal utility of purchasing any product with the remaining money will be less than the marginal utility of the remaining money itself, so the consumer will stop buying any product. In this work, we use sigmoid-normalized price $\sigma(p_x)$ in Eq.(5) for easier optimization.

3.4 Risk Attitude

Risk usually exists when people make decisions under uncertainty, for example, in recommender systems, consumers never know for sure if they will satisfy with a product or not. Different people have different attitudes toward risk. Generally, risk attitudes can be divided into three categories, which are risk aversion, risk appetite, and risk neutral. Take buying lottery as an instance, let $p : Z \to [0, 1]$ describe a lottery that has uncertain incomes, and let lottery X be a random variable, whose expected value is $EX = \sum_{x \in Z} xp(x)$ (or $EX = \int xp(x)dx$ for continuous cases), then,

Risk Aversion implies that, for a given lottery, its expected utility is less than its utility of expectation, as in Figure 1(a), $EU(X) < U(EX)$. A risk aversion person always tries to avoid risks.

Risk Appetite means that, for a given lottery, its expected utility is greater than its utility of expectation, as in Figure 1(b), $EU(X) > U(EX)$. A risk appetite person prefers to maximize the profits even with considerable risks.

Risk Neutral implies that, for a given lottery, its expected utility is equal to its utility of expectation, as in Figure 1(c), which means that a person keeps neutral in face of risks, $EU(X) = U(EX)$.

For example, suppose there are two choices in a gamble, one is to win either \$100 or \$0 with 50% probability each, and the other choice is to get \$50 with 100% probability. Though the expected utility of both choices is the same (\$50), the user's utility of expectation can be different because of their different risk preferences.

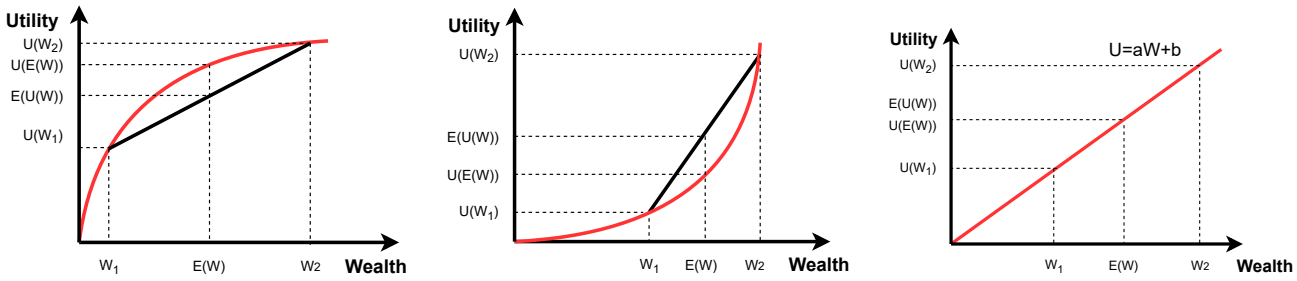
In this paper, our model is based on an economic assumption that the risk attitudes of all the consumers are approximately risk-neutral, which means that the optimization algorithm attempts to minimize the difference between $EU(X)$ and $U(EX)$. The theoretical basis of this assumption comes from classical economic theorem on expected utility [1, 2, 19], which shows that a rational person with a differentiable utility function will always prefer to take a sufficiently small stake in any positive-expected-value bet. In other words, rational persons can be (almost everywhere) arbitrarily close to risk neutral when stakes are sufficiently small. Since most people only spend a small part of their assets purchasing an item online, this theorem is suitable for e-Commerce recommendation scenarios.

4 THE FRAMEWORK

4.1 The Personalized MUD Function

Different users may have different marginal utility functions, which represent their personalized preferences towards different items. In order to leverage MUD optimization for personalized recommendation, we need to adapt the (non-personalized) MUD function in Eq.(5) to personalized settings. First, the parameter a_x in Eq.(5) only depends on the product x instead of a particular user-item pair. To solve the problem, we substitute a_x with a_{ij}, where i indicates the i-th user, and j indicates the j-th item. Besides, a piece of important information in recommendation systems is the user-item rating, which contains rich signals about the consumer preference on targeted items. Usually, higher rating scores for a user-item pair imply higher consumer satisfaction, thus brings higher MUD. As a result, we introduce the rating scores into the MUD function. Finally, the personalized MUD function is defined as:

$$\mathcal{M}_{ij}(\hat{r}_{ij}, Q_{ij}) = \frac{\tanh(\hat{r}_{ij})a_{ij}}{(1 + Q_{ij})\sigma(p_j)} \tag{6}$$



(a) Risk Aversion (b) Risk Appetite (c) Risk Neutral

Figure 1: Three different risk attitudes based on the relationship between the expected utility and the utility of expectation. Here, x-axis represents the uncertain income, which depends on the specific scenario, e.g. it can be the reward money in a lottery. y-axis is the utility of the corresponding value. Intuitively, W can be considered as the objective (monetary) utility of an item or action, and U can be considered as the consumer's subjective utility (feeling) over the item. The relative relationship between U and W reflects the risk attitude of the consumer.

where Q_{ij} is the consumption quantity of item j by consumer i; $\sigma(p_j)$ is the sigmoid price of item j, which can be seen as a normalization of item prices; a_{ij} is the personalized shape parameter that determines the scale of MUD; and $\hat{r}_{ij}$ is the predicted rating score of item j by consumer i. For easy optimization, we adopt hyperbolic tangent $\tanh(\cdot)$ to map the ratings in to the range of (0,1).

There are many different ways to predict the user-item rating scores $\hat{r}_{ij}$, including user/item-based collaborative filtering, matrix factorization, or deep neural networks. For simplicity, we take the most commonly used matrix factorization (MF) approach [11] for rating prediction.

4.2 Multinomial Logistic Modeling (MLM)

Discrete choice problem describes a situation when a consumer chooses an option between two or more discrete alternatives. There are many forms of different discrete choice models and they share some similarities. More formally, suppose that at time point t, consumer i chooses item j over a set of some other alternative products $\Omega_{it}(j)$. We define the total choice set as $\Pi_{it} = \{j, \Omega_{it}(j)\}$ and its k-th element is Π^k_{it} ($\Pi^1_{it} = j$). The probability that consumer i chooses alternative j is expressed as P_{ij}.

Researchers in economics have utilized Random Utility Models (RUM) to deal with this problem [32]. Different from traditional RUMs, in this paper, we adapt the idea of choosing the alternative item that provides the highest utility into choosing the alternative item with the highest MUD, so as to maximize the utility without violating the budget constraint. In this way, we have $\hat{\mathcal{M}}_i(\Pi^k_{it}) = \mathcal{M}_i(\Pi^k_{it}) + \epsilon_k$, where $\mathcal{M}_i(\Pi^k_{it})$ represents the true MUD of a product combination, and $\hat{\mathcal{M}}_i(\Pi^k_{it})$ represent the observed MUD. ϵ_k is a random variable capturing the impact of all unknown factors. Thus, the probability a customer chooses Π^1_{it} (item j) over other alternatives is:

$$P_{ij}(\hat{\mathcal{M}}_i(\Pi^1_{it}) > \hat{\mathcal{M}}_i(\Pi^k_{it})) = P_{ij}(\epsilon_k - \epsilon_1 < \mathcal{M}_i(\Pi^1_{it}) - \mathcal{M}_i(\Pi^k_{it})) \quad (7)$$

where $k = 2, \dots, |\Pi_{it}|$. If ϵ_1 and ϵ_k obey an i.i.d. extreme value distribution, then we have,

$$P_{ij}(y_{it} = 1) = P_{ij}(\hat{\mathcal{M}}_i(\Pi^1_{it}) > \hat{\mathcal{M}}_i(\Pi^k_{it})) = \frac{exp(\mathcal{M}_i(\Pi^1_{it}))}{\sum_{k=1}^{|\Pi_{it}|} exp(\mathcal{M}_i(\Pi^k_{it}))} \quad (8)$$

where y_{it} is an indication function that,

$$y_{it} = \begin{cases} 1 & \mathcal{M}_i(\Pi^1_{it}) > \mathcal{M}_i(\Pi^k_{it}) \quad \forall k \neq 1 \\ 0 & Otherwise \end{cases} \quad (9)$$

4.3 Estimated Risk Distribution

Shopping online is risky because consumers may be unsatisfactory with the purchased items. In order to better understand consumer behaviors, we take risk into consideration based on classical expected utility theory. To calculate the expected utility of a product, we have to know the probability distribution function $P(x)$ of its utility. In this work, we simulate the distribution function for each specific user-item pair as $\mathcal{N}(\hat{r}_{ij}, \sigma^2_{MF})$, where $\hat{r}_{ij}$ is the predicted rating score from matrix factorization (MF), and $\sigma_{MF} = \sqrt{\frac{1}{N-1}\sum(r_{ij} - \hat{r}_{ij})^2}$ is the unbiased estimation of prediction RMSE. Since the objective function of MF minimizes RMSE, this distribution naturally applies a higher probability to those frequent ratings. As a result, the intuition of the setting is that if the predicted rating $\hat{r}_{ij}$ is close to the true rating r_{ij} (i.e., easy to predict), then we should have a higher probability to observe this rating.

4.4 Risk-based Optimization of MUD (ROM)

We introduce our optimization framework based on approximate risk neutral constraint as follows:

$$\begin{cases} \max \sum_{(i,j)\in\mathcal{R}} log(P(y_{ij} = 1)) - \lambda\|\Phi\|^2 \end{cases} \quad (10)$$

$$\begin{cases} s.t. \ \min \sum_{(i,j)\in\mathcal{R}} (\bar{U}_{ij} - U_{ij}(\bar{r}))^2 + \lambda\|\Phi\|^2 \end{cases} \quad (11)$$

where $\mathcal{R}$ is the set of all the observed user-to-item rating pairs; $P(y_{it} = 1)$ is the multinomial logistic model described in Eq.(8); $\bar{U}_{ij}$ is the expected utility, and $U_{ij}(\bar{r})$ is the utility of expectation, which will be introduced in detail in the next section; and finally, Φ is the parameter set to be learned in the corresponding loss function, which will also be crystallized in the next section. The model involves two optimization functions Eq.(10) and Eq.(11), showing that

Table 1: Parameters setting for each datasets and method where K is the latent factor size, lr is learning rate, #i is the number of iterations and λ is the regularization parameter.

Dataset	Baby				Electronics				Games			
Methods	K	lr	λ	#i	K	lr	λ	#i	K	lr	λ	#i
CF	50	1e-4	1.5	3	100	1e-4	0.5	2	100	1e-5	1	3
BPR	100	1e-4	1	3	50	1e-4	1	3	100	1e-4	1	3
NCF	100	1e-4	1	5	70	1e-4	1	5	100	1e-4	1	5
MPUM	100	1e-4	1	3	70	5e-4	1	3	100	5e-4	1.5	3
ROM	100	5e-4	1	3	70	1e-4	0.5	3	100	1e-4	1.5	3

we maximize the log-likelihood of a multinomial logistic model, so as to optimize the probability of purchasing positive items based on marginal utility per dollar under risk neutral constraint.

4.5 Model Specification

We provide details of the above framework to explain how to learn the parameters. In particular, ratings are predicted based on MF:

$$\hat{r}_{ij} = \alpha^r + \beta_i^r + \gamma_j^r + (\mathbf{p}_i^r)^T \mathbf{q}_j^r \tag{12}$$

where α^r is the global bias, β_i^r is the user bias, γ_j^r is the item bias, $\mathbf{p}_i^r$ and $\mathbf{q}_j^r$ are K-dimensional latent factors of user i and item j. Then we use $\hat{r}_{ij}$ as an input to calculate MUD, as shown in Eq.(6). To reduce the parameter space, a_{ij} is also re-parameterized based on the idea of latent factor modeling,

$$a_{ij} = \alpha^a + \beta_i^a + \gamma_i^a + (\mathbf{p}_i^a)^T \mathbf{q}_j^a \tag{13}$$

In this way, the optimization problem in Eq.(10) and Eq.(11) can be specialized as follow:

$$\max \sum_{(i,j)\in\mathcal{R}} \log\left(\frac{exp(\mathcal{M}_i(\Pi_{it}^1))}{\sum_{k=1}^{|\Pi_{it}|} exp(\mathcal{M}_i(\Pi_{it}^k))}\right) - \lambda\|\Phi\|^2, \tag{14}$$

$$\min \sum_{(i,j)\in\mathcal{R}} \left\{\sum_{n=1}^{5} P_n \tanh(n) - \tanh\left(\sum_{n=1}^{5} nP_n\right)\right\}^2 a_{ij}^2 \ln^2(1+Q_{ij}) + \lambda\|\Phi\|^2, \tag{15}$$

where $P_n = Pr(r_{ij} = n) = \int_{n-0.5}^{n+0.5} \mathcal{N}(\hat{r}_{ij}, \sigma_{MF}^2)\mathrm{d}r$ is the probability that the rating score of item j is n (n = 1, 2, 3, 4 or 5, representing five different rating scores, for $n = 1$, integral lower bound is $-\infty$, and for $n = 5$, upper bound is $+\infty$). We use SGD optimization algorithm to learn the model parameters.

4.6 Top-K Recommendation

Once we learned the model parameters a_{ij} and $\hat{r}_{ij}$ according to ROM, we can then calculate the marginal utility per dollar $\mathcal{M}_{ij}$ for each user-item pair. We rank all products for a user according to their MUD values, and select the top items whose price is within the user's budget to generate the top-K recommendation list, wherein this work, a consumer's budget is considered as the highest price that the consumer had ever spent in his/her purchasing history.

5 EXPERIMENTS

5.1 Dataset Description

We use the consumer transaction data from Amazon[1] [6, 15] in our experiments. The dataset includes user transaction (user id, item id, rating, etc.) and item metadata (item id, price, related item, etc.) on 24 product categories lasting from May 1996 to July 2014. We take three categories (Baby, Electronics and Video Games) that have different size and data sparsity for experiments.

Table 2: Basic statistics of the experimental data sets.

Dataset	#users	#items	#interactions	sparsity
Baby	23,894	39,767	200,170	0.0211%
Electronics	58,248	45,777	477,074	0.0179%
Video Games	24,735	23,669	236,530	0.0404%

Basic statistics of the experimental datasets as shown in Table 2. For each dataset, we sort the transactions of each consumer according to the purchase timestamp, and then split the records into training, validation and testing sets chronologically by 3:1:1 [2].

5.2 Experimental Setup

We compare our model with the following baselines, including both economic and non-economic methods. For non-economic methods, we involve both shallow and deep learning baselines.

CF: Collaborative Filtering based on matrix factorization is a representative method for rating prediction. In this experiment, we use CF based on latent factor modeling [11].

BPR: Bayesian Personalized Ranking [20] is one of the most representative ranking-based methods for the top-N recommendation.

NCF: Neural Collaborative Filtering is a state-of-the-art recommendation algorithm based on deep neural networks. We choose Neural Matrix Factorization to conduct the experiments, which fuses Generalized Matrix Factorization and Multiple Layer Perceptron under the NCF framework.

MPUM: Multi-Product Utility Maximization for recommendation [32], which is an economic recommendation approach that maximizes the utility of product combinations for recommendation.

For each dataset, we use the validation set to find the best parameters of each method, which are shown in Table 1.

5.3 Evaluation Measure

To compare the performance of our model with the baselines, we use both traditional measures such as Precision, Recall, and F_1, and the economic measure MUD. Assuming Γ_i is the recommendation list for consumer i, and N is the total number of consumers, the average marginal utility per dollar over all users is,

$$\mathcal{M}@|\Gamma_i| = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\in\Gamma_i} \frac{a_{ij}\tanh(r_{ij})}{(1+Q_{ij})\sigma(p_j)} \tag{16}$$

[1] http://jmcauley.ucsd.edu/data/amazon/

[2] code of the paper is released at https://github.com/TobyGE/Maximizing-Marginal-Utility-per-Dollar-for-Economic-Recommendation

Table 3: Summary of the performance. We evaluate for ranking (P, R, F_1, NDCG) and economic value (MUD), and K is the length of recommendation list. When ROM is the best, its improvements against the best baseline are significant at p=0.01.

Dataset	Baby														
Measures	Precision(%)			Recall (%)			F_1 Measure (%)			NDCG			MUD ($\times 10^{-2}$)		
K	1	5	10	1	5	10	1	5	10	1	5	10	1	5	10
CF	3.687	2.658	1.929	2.613	9.421	13.669	3.059	4.147	3.380	0.0084	0.0283	0.0599	0.29448	1.3698	2.6969
NCF	3.829	3.151	2.522	2.714	8.935	17.878	3.177	4.659	4.421	0.0187	0.0503	0.0868	0.28657	1.3699	2.7224
BPR	4.185	3.041	2.483	2.966	10.777	17.600	3.472	4.743	4.352	0.0162	0.0472	0.0837	0.26216	1.3164	2.6720
MPUM	6.177	2.978	2.262	4.378	10.554	16.036	5.125	4.645	3.966	0.0338	0.0718	0.0915	0.26664	1.3482	2.7099
ROM	**7.014**	**3.421**	**2.553**	**4.972**	**12.124**	**18.098**	**5.819**	**5.336**	**4.475**	**0.0380**	**0.0804**	**0.1021**	**33.149**	**73.634**	**106.523**
Dataset	Electronics														
Measures	Precision(%)			Recall(%)			F_1 Measure(%)			NDCG			MUD ($\times 10^{-2}$)		
K	1	5	10	1	5	10	1	5	10	1	5	10	1	5	10
CF	5.983	3.199	2.262	4.316	11.538	16.319	5.014	5.009	3.974	0.0106	0.0336	0.0726	0.01793	0.06293	0.12605
NCF	6.852	4.089	**2.944**	4.942	14.746	**21.233**	5.742	6.402	**5.170**	0.0142	0.0456	0.0970	0.01426	0.05981	0.11355
BPR	7.847	4.011	2.648	5.660	14.468	19.099	6.577	6.281	4.651	0.0087	0.0313	0.0811	0.01149	0.07350	0.12193
MPUM	9.866	3.452	2.333	7.117	12.452	16.827	8.269	5.406	4.098	0.0652	0.1006	0.1168	0.00788	0.05266	0.11403
ROM	**11.262**	**4.159**	2.766	**8.124**	**15.000**	19.952	**9.439**	**6.512**	4.859	**0.0721**	**0.1167**	**0.1346**	**14.551**	**29.770**	**41.179**
Dataset	Video Games														
Measures	Precision(%)			Recall(%)			F_1 Measure(%)			NDCG			MUD ($\times 10^{-2}$)		
K	1	5	10	1	5	10	1	5	10	1	5	10	1	5	10
CF	3.558	2.421	1.891	2.159	7.345	11.473	2.687	3.641	3.247	0.0121	0.0313	0.0568	0.12979	0.64827	1.2875
NCF	3.869	2.907	2.426	2.348	8.819	14.719	2.922	4.372	4.165	0.0170	0.0431	0.0745	0.13018	0.64710	1.2867
BPR	4.253	2.917	2.359	2.581	8.851	14.316	3.212	4.388	4.051	0.0152	0.0404	0.0715	0.12956	0.64319	1.2845
MPUM	6.642	3.007	2.201	4.030	9.123	13.357	5.017	4.523	3.078	0.0389	0.0690	0.0846	0.12303	0.63397	1.2765
ROM	**7.026**	**3.297**	**2.516**	**4.263**	**10.001**	**15.268**	**5.307**	**4.959**	**4.320**	**0.0400**	**0.0737**	**0.0920**	**9.0342**	**20.267**	**29.874**

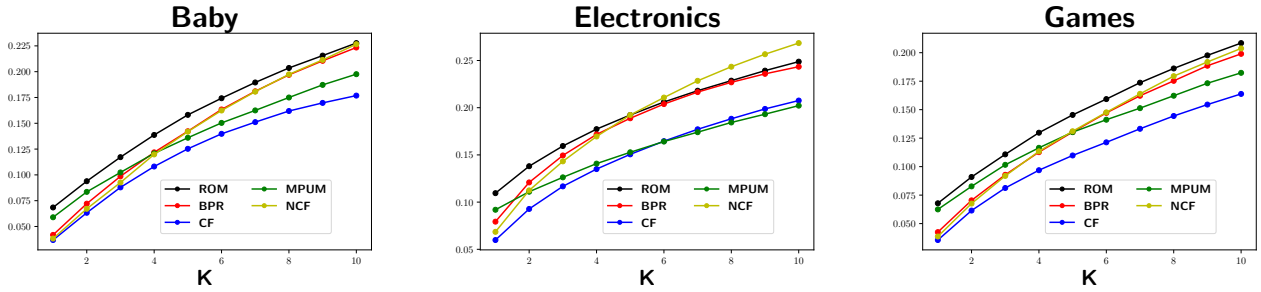


Figure 2: Conversion rate on three datasets. x-axis is the length of the recommendation list and y-axis is the conversion rate.

5.4 Experimental Results

Key experimental results are shown in Table 3, and we also plot the conversion rate in Figure 2 under different recommendation length K (from 1 to 10). We analyze and discuss the results in terms of the following three perspectives.

5.4.1 **Recommendation Performance:** According to Table 3, among the shallow baseline models (CF, BPR, and MPUM), all pair-wise learning methods (BPR and MPUM) are better than the point-wise CF method, which shows the superiority of pair-wise methods on top-K ranking tasks. By learning deep structures for user-item matching, NCF achieves the best baseline performance in many cases. Furthermore, our ROM approach achieves the best top-K performance against all baselines in most cases. For example, when averaged across all datasets and recommendation lengths, we get 27.97% improvement than NCF. In particular, the improvement is 83.18% for Precision@1 on the *Baby* dataset against NCF. For NDCG, we get 112.54% improvement in average than NCF baseline, especially the largest improvement (407.75%) is achieved on the *Electronics* when $K = 1$. For conversion rate, we get 15.66% improvement than the BPR baseline when averaged across K on the *Baby* dataset, and the largest improvement (63.5%) is achieved when $K = 1$. Compared with NCF, we got 19.01% improvement for top 1 to 10 conversion rate on *Baby* and the largest improvement (78.69%) is achieved when $K = 1$.

The observations imply that by modeling user behaviors based on established economic principles, our model is able to better capture the user preferences for the top-K recommendation.

5.4.2 **Economic Performance:** It is not surprising that our method achieves significantly higher MUD scores than non-economic methods, because we try to optimize for economic values directly. Therefore, we only compare ROM with another economic baseline (i.e., MPUM). We see that our model gets much higher MUD than MPUM, which means that higher utility values do not necessarily imply higher marginal utility per dollar. The higher MUD values produced by our model mean that our recommendations could help users to spend their money more efficiently (i.e., more utility gain per dollar), and thus help to provide more informed recommendations.

5.4.3 **Shallow vs. Deep Models:** According to Table 3 and Figure 2, our model is better than NCF in most cases. Nevertheless, NCF got 6.43% improvement from ROM when $K = 10$ on *Electronics*, and according to Fig.2, NCF grows faster when the length of the recommendation list increases. Neural network is good at learning complex user-item interactions, so it is not surprising that deep models are able to achieve better performance when K is sufficiently large. However, in practical recommender systems, the top-5 or 10 results are much more important than later items. Our model is better than NCF from top-1 to top-10 in most cases, and the improvement is especially significant on top-1, which is also reasonable because marginal utility measures the incremental benefit for each extra unit of an item.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we bridge economic principles and machine learning for personalized recommendation together. In particular, we propose to maximize marginal utility per dollar under approximate risk-neutral constraint for recommendation. Experimental results verified the effectiveness of our model on top-K recommendation and economic values. In the future, we will relax the risk-neural constraint to other risk attitude functions for more diverse user modelings. We will also consider other economic principles and/or learning methods to benefit recommendation systems both effectively and economically.

REFERENCES

[1] Kenneth J Arrow. 1970. *Essays in the theory of risk-bearing*. Technical Report.

[2] Shlomo Benartzi and Richard H Thaler. 1995. Myopic loss aversion and the equity premium puzzle. *The quarterly journal of Economics* 110, 1 (1995), 73–92.

[3] James S. Coleman and Thomas J. Fararo. 1992. Rational Choice Theory. *Nueva York: Sage* (1992).

[4] Daniel Friedman and József Sákovics. 2015. Tractable consumer choice. *Theory and Decision* 79, 2 (2015), 333–358.

[5] John C Harsanyi. 1987. Von Neumann-Morgenstern utilities, risk taking, and welfare. In *Arrow and the ascent of modern economic theory*. Springer, 545–558.

[6] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on World Wide Web*. 507–517.

[7] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *AAAI*.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*. 173–182.

[9] Joseph A Konstan, Bradley N Miller, David Maltz, Jonathan L Herlocker, Lee R Gordon, and John Riedl. 1997. GroupLens: applying collaborative filtering to Usenet news. *Commun. ACM* 40, 3 (1997), 77–87.

[10] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).

[11] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[12] D. D. Lee and H. S. Seung. 2001. Algorithms for Non-negative Matrix Factorization. *NIPS* (2001), 556–562.

[13] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*. 556–562.

[14] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.Com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing* 7, 1 (Jan. 2003), 76–80. https://doi.org/10.1109/MIC.2003.1167344

[15] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-based recommendations on styles and substitutes. In *SIGIR*. ACM.

[16] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.

[17] James N Morgan. 1945. Can we measure the marginal utility of money? *Econometrica: Journal of the Econometric Society* (1945), 129–152.

[18] Changhua Pei, Xinru Yang, Qing Cui, Xiao Lin, Fei Sun, Peng Jiang, Wenwu Ou, and Yongfeng Zhang. 2019. Value-aware Recommendation based on Reinforcement Profit Maximization. *WWW* (2019).

[19] Matihew Rabin. 2013. Risk aversion and expected-utility theory: A calibration theorem. In *Handbook of the Fundamentals of Financial Decision Making: Part I*. World Scientific, 241–252.

[20] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[21] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. ACM, 175–186.

[22] Jozsef Sakovics, Daniel Friedman, et al. 2011. *The marginal utility of money: A modern Marshallian approach to consumer choice*. Technical Report. Edinburgh School of Economics, University of Edinburgh.

[23] R. Salakhutdinov and A. Mnih. 2008. Bayesian Probabilistic Matrix Factorization using Markov Chain Monte Carlo. *ICML* (2008), 880–887.

[24] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[25] Peter Wakker and Daniel Deneffe. 1996. Eliciting von Neumann-Morgenstern utilities when probabilities are distorted or unknown. *Management science* 42, 8 (1996), 1131–1150.

[26] Jian Wang and Yi Zhang. 2011. Utilizing marginal net utility for recommendation in e-commerce. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 1003–1012.

[27] Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep learning based recommender system: A survey and new perspectives. *arXiv preprint arXiv:1707.07435* (2017).

[28] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W Bruce Croft. 2017. Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1449–1458.

[29] Yongfeng Zhang, Min Zhang, Yiqun Liu, Shaoping Ma, and Shi Feng. 2013. Localized matrix factorization for recommendation based on matrix block diagonal forms. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1511–1520.

[30] Yongfeng Zhang, Qi Zhao, Yi Zhang, Daniel Friedman, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Economic Recommendation with Surplus Maximization. In *WWW*.

[31] Qi Zhao, Yi Zhang, Daniel Friedman, and Fangfang Tan. 2015. E-commerce recommendation with personalized promotion. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 219–226.

[32] Qi Zhao, Yongfeng Zhang, Yi Zhang, and Daniel Friedman. 2017. Multi-product utility maximization for economic recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 435–443.

The few-get-richer: a surprising consequence of popularity-based rankings*

Fabrizio Germano
U. Pompeu Fabra, BGSE
fabrizio.germano@upf.edu

Vicenç Gómez
U. Pompeu Fabra
vicen.gomez@upf.edu

Gaël Le Mens
U. Pompeu Fabra, BGSE, SDU
gael.le-mens@upf.edu

ABSTRACT

Ranking algorithms play a crucial role in online platforms ranging from search engines to recommender systems. In this paper, we identify a surprising consequence of popularity-based rankings: the fewer the items reporting a given signal, the higher the share of the overall traffic they collectively attract. This few-get-richer effect emerges in settings where there are few distinct classes of items (e.g., left-leaning news sources versus right-leaning news sources), and items are ranked based on their popularity. We demonstrate analytically that the few-get-richer effect emerges when people tend to click on top-ranked items and have heterogeneous preferences for the classes of items. Using simulations, we analyze how the strength of the effect changes with assumptions about the setting and human behavior. We also test our predictions experimentally in an online experiment with human participants. Our findings have important implications to understand the spread of misinformation.

ACM Reference Format:
Fabrizio Germano, Vicenç Gómez, and Gaël Le Mens. 2019. The few-get-richer: a surprising consequence of popularity-based rankings. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313693

1 INTRODUCTION

Ranking systems are at the core of many online services, including search engines, recommender systems, or news feeds in social media. Recent research suggests that the underlying ranking algorithms may impact society, playing an active role in the spread of misinformation [29], political polarization [11], or trustworthiness [14]. They might also reinforce existing judgment biases [2].

Rankings systematically affect the information people access about products, services, events, or ideas, because users are more likely to click on top-ranked items [19, 24, 30]. When items are ranked based on popularity, this leads to a self-reinforcing dynamics according to which popular items become increasingly more popular [28].

In this paper, we identify a surprising effect of popularity-based rankings. Consider a setting with two distinct classes of news sources that differ in their political orientations, e.g., left-leaning or right-leaning. We show that, under a fairly broad set of conditions, *the total share* of web traffic (proportion of clicks) attracted by a given class of news sources *decreases* with the number of news sources in that class. We call this phenomenon the 'few-get-richer' effect. For example, if there are 20 news sources, the total number of clicks on left-leaning sources will be larger when there are just 3 of these sources than when there are 17 of them.

Intuition suggests that popular items should be more relevant and trustworthy than unpopular ones. Yet, extensive research indicates that popularity is often not very informative about quality, especially in settings characterized by 'rich-get-richer' dynamics (sometimes called the 'Matthew effect') [13, 22, 25, 28], or information cascades [3, 5]. In these settings, the randomness inherent to the dynamics of the system implies that items that become the most popular are not always those with the best quality. The 'few-get-richer' effect adds to research on the 'rich-get-richer' dynamics by showing that popularity-based rankings do not only create 'noise' in the ranking, but can also lead to a systematic ranking bias: when there are two distinct classes of items, items from the smaller class become better ranked than similar items from the larger class.

The few-get-richer effect emerges in settings characterized by two design features. The first feature consists in the ranking of items in terms of popularity (i.e., items with more clicks are higher ranked). The second feature is a partition of the available items in two (or more) distinct classes.

We make two reasonable behavioral assumptions. The first assumption is users' tendency to click on top-ranked items. The second assumption is that users have heterogeneous preferences for the item classes. Some users have a preference for items of a particular class, while others have a preference for items of other classes. Still other users are indifferent to the item class.

Returning to our news search example, suppose there are few left-leaning and many right-leaning news sources. We assume there are three types of users: left-leaning, right-leaning, and indifferent. The heterogeneous preference assumption means that left-leaning users are more likely to click on left-leaning news sources, right-leaning users are more likely to click on right-leaning news sources, and indifferent users click exclusively based on rank. Even if the left-leaning news sources are unpopular, left-leaning individuals will seek them out. Because there are few such news sources, the clicks of these left-leaning individuals will be concentrated on a few news sources, and these sources will tend to 'shoot up to the top'.

*F. Germano acknowledges financial support from Grant ECO2017-89240-P (AEI/FEDER, UE) from the Spanish MINECO. G. Le Mens benefited from financial support from Ramon y Cajal Fellowship RYC-2014-15035 and Grant #AEI/FEDER UE-PSI2016-75353 from the Spanish MINECO, and ERC Consolidator Grant #772268 from the European Commission. V. Gómez is funded by the Ramon y Cajal program RYC-2015-18878 (AEI/MINECO/FSE,UE) and the María de Maeztu Units of Excellence Programme (MDM-2015-0502). The research leading to these results has received funding from "La Caixa" Banking Foundation. Data from the experiment are available at https://osf.io/nwjyf/.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313693

Once a news source has gotten close to the top, it will attract not only the clicks from the left-leaning individuals, but also the clicks of indifferent users, simply because of the rich-get-richer dynamics. This is the few-get-richer effect.

2 RELATED WORK

Our results contribute to the understanding of the limitations of recommender systems [8, 10, 18, 21, 23, 24], with direct applications to the design of fair, transparent and efficient ranking systems [4, 6, 32], as well as methods to reduce the spread of misinformation [20, 29] or uncivil behavior [9, 26].

An extensive literature on modeling (click) user behavior is weakly related to our work [7, 15, 27, 33]. Closer to our work, several papers have proposed models of the dynamics of interactions between individual searches and ranking algorithms, e.g., for understanding the feedback loop between ranking system and user queries [12], explaining the observed mitigation of search engines' popularity bias [16], or the competition of memes using limited attention [31]. The paper closest to ours is [17], which also obtains a few-get-richer effect in a model where individuals get multiple signals and where (news) items are ranked via a probabilistic popularity-based ranking. Besides being simpler, our model works with a discrete and deterministic ranking of the websites rather than a continuous and probabilistic one. Among other things, this allows for a tighter connection with the experiment of Section 5.

3 THE MODEL

We present a stylized model of a search environment where individuals use a search engine to look for information on a binary issue. At the center of the model is the ranking algorithm, which ranks and directs individuals to the different websites, based on the popularity of individuals' choices.

3.1 Model of the Search Environment

There are M items, each of which belongs to exactly one of two classes. For example, the items can be news articles and the class of an article can be whether the source is known to be left-leaning or right-leaning. A different, visual, example can comprise images about animals, some of which are cats while the others are dogs. In general, each of the M items is characterized by a binary signal $\{0, 1\}$ that defines its class; let M_k denote the set and number of items of class k, $k \in \{0, 1\}$.

There is a single popularity-based ranking algorithm which, starting with a given initial ranking (r_1), ranks all the items in M according to the number of clicks received. Let $r_n \in \{1, \dots, M\}^M$ denote the ranking seen by individual n, where $r_{n,m} \in \{1, \dots, M\}$ denotes the rank of item m observed by individual n.

There are N individuals, each of which is characterized by a parameter γ_n, which can be of one of three types. The three types are denoted by Type 0, Type 1 and Type 2, and their proportions in the population are p_0, p_1 and $p_2 = 1 - p_0 - p_1$, respectively. Type 0 (resp. Type 1) individuals have a preference for clicking on items of class 0 (resp. class 1). Type 2 individuals are indifferent between clicking on items of class 0 or of class 1 *absent ranking*. We represent the set of possible types by $\Gamma = \{\Gamma_0, \Gamma_1, \Gamma_2\}$, where Γ_i denotes the probability that an individual of Type i (i.e., with $\gamma_n = \Gamma_i$) clicks on an item of class 0 *absent ranking*. Our assumptions about the preferences of the individuals in the three types imply $\Gamma_0 > \frac{1}{2}$, $\Gamma_1 < \frac{1}{2}$ and $\Gamma_2 = \frac{1}{2}$.

We summarize these preferences in terms of propensities $\varphi_{n,m}$ with which individual n with $\gamma_n \in \Gamma$ clicks on item m, defined by:

$$\varphi_{n,m} = \begin{cases} \frac{\gamma_n}{M_0} & \text{if } m \in M_0 \\ \frac{1-\gamma_n}{M_1} & \text{if } m \in M_1. \end{cases} \tag{1}$$

For simplicity, in this section, we consider the extreme and symmetric case, where $\Gamma_0 = 1$, $\Gamma_1 = 0$ and $p_0 = p_1 = p > 0$.

3.2 Model of Stochastic Choices

Individuals enter one after the other and observe the ranked list of items, and based on the ranking and on their preference for the signals given by $\varphi_{n,m}$, they click on one of the M items according to a probabilistic choice function obtained as follows. We use the function $\beta^{(M-r_{n,m})}$ to weigh the propensities $\varphi_{n,m}$, whereby $\beta > 1$ calibrates an individual's search cost or attention bias, so that an item ranked exactly one position higher has β times as much probability of being clicked. The probability individual n clicks on website m is given by:

$$\rho_{n,m} = \frac{\beta^{(M-r_{n,m})}\varphi_{n,m}}{\sum_{m' \in M} \beta^{(M-r_{n,m'})}\varphi_{n,m'}}. \tag{2}$$

Thus, individual n observes $r_{n,m}$ and clicks on a website according to $\rho_{n,m}$. His click gets recorded by the ranking algorithm, which updates its ranking to r_{n+1}. This affects the ranking of the websites observed by the next individual, who clicks on a website according to $\rho_{n+1,m}$ and so on. We denote the total clicking probability on items of class k (M_k) by $\rho_{n,M_k} = \sum_{m \in M_k} \rho_{n,m}$.

Overall, this defines a search environment $\mathcal{E}$ with parameters $(M, M_1); (N, \beta, \Gamma, p); r_1$. We assume $\beta > 1$ and $0 < p < \frac{1}{2}$ so that individuals are subject to an attention bias and are heterogeneous in the sense that there are nonzero shares of Types 0, 1 and 2.

3.3 The Few-Get-Richer Effect

The following result shows how few items of a given class can attract more traffic than many more items of the same class taken together. In particular, it implies that if there is just one item of a given class, then it will attract more traffic by itself than $M - 1$ such items taken together in a corresponding environment where there are $M - 1$ such items.

Proposition 1. *Fix two popularity-based search environments $\mathcal{E}$ and $\mathcal{E}'$ that differ only in the number of items of class 1 (M_1 and M_1' respectively). Suppose $M_1 < \frac{M}{1+\beta} < \frac{\beta M}{1+\beta} < M_1'$, then there exists $\overline{N}$ such that, for any $N \geq \overline{N}$, the total clicking probability (ρ_{N,M_1}) by individual N on an item in M_1 in environment $\mathcal{E}$ is strictly greater than the total clicking probability ($\rho_{N,M_1'}$) by individual N on an item in M_1' in environment $\mathcal{E}'$, provided $p > 0$ is sufficiently small.*

The proof is in three steps. First, we characterize a limit ranking (r_∞) of the process ρ_n defined by Eq. (2) and show it constitutes a (stable) limit. Second, we show it is the unique such limit ranking. Finally, we compute total traffic on all items in M_1 at the limit and

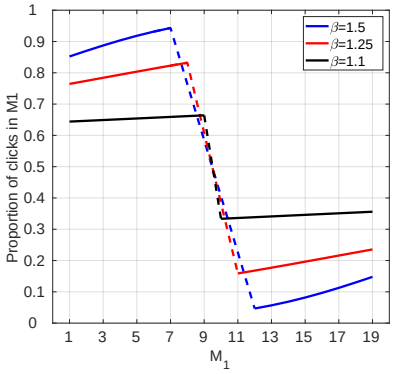


Figure 1: Total traffic on all M_1 items of class 1 as a function of M_1 for the limit distribution of the process ρ_n of Eq. (2) for different values of β, with $M = 20$, $\Gamma = \{0.8, 0.2, 0.5\}$ and $p = p_0 = p_1 = 0.4$.

show it is over half of total traffic when $M_1 < \frac{M}{1+\beta}$, and hence greater than total traffic on all items in M_1' for $M_1' > \frac{\beta M}{1+\beta}$.

Step 1. Consider M items and a ranking $r_\infty \in \{1, \ldots, M\}$ defined by $r_{\infty,k} = k$, for $k = 1, \ldots, M$. Then, by popularity ranking, r_∞ is a (stable) limit ranking of the process ρ_n if and only if at r_∞:

$$\rho_{n,1}(r_\infty) > \rho_{n,2}(r_\infty) > \ldots > \rho_{n,M}(r_\infty) \tag{3}$$

holds for the expected individual n. Suppose r_∞ is such that the M items are ranked in two blocks of items of the same class where, if $M_1 < \frac{M}{1+\beta}$, the first ranked M_1 items are all in M_1 and the remaining ones (bottom ranked) are all in M_0; and symmetrically if $M_1 > \frac{\beta M}{1+\beta}$, the bottom ranked M_1 items are all in M_1 and the remaining ones (top ranked) are all in M_0. To see that these constitute limit rankings, suppose $M_1 < \frac{M}{1+\beta}$, and consider the ranking r_∞, where the first M_1 items are ranked on top. In this case, the corresponding clicking probabilities for $k \in M_1$ (i.e., $k \in \{1, \ldots, M_1\}$) are given by:

$$\rho_{n,k}(r_\infty) = \frac{\beta^{(M-k)}(1-\gamma_n)/M_1}{\sum_{k=1}^{M_1} \beta^{(M-k)}(1-\gamma_n)/M_1 + \sum_{k=M_1+1}^{M} \beta^{(M-k)}\gamma_n/M_0},$$

and for $k \in M_0$ (i.e., $k \in \{M_1 + 1, \ldots, M\}$) satisfy:

$$\rho_{n,k}(r_\infty) = \frac{\beta^{(M-k)}\gamma_n/M_0}{\sum_{k=1}^{M_1} \beta^{(M-k)}(1-\gamma_n)/M_1 + \sum_{k=M_1+1}^{M} \beta^{(M-k)}\gamma_n/M_0}.$$

Clearly, $\rho_{n,k}(r_\infty) > \rho_{n,k+1}(r_\infty)$ holds within the classes M_1 and M_0, that is, for $k = 1, \ldots, M_1 - 1$ and for $k = M_1 + 1, \ldots, M - 1$ Hence it suffices to show that $\rho_{n,k=M_1}(r_\infty) > \rho_{n,k=M_1+1}(r_\infty)$. This is easily checked for the expected individual n (whose value of γ_n is drawn from $\Gamma = \{1, 0, \frac{1}{2}\}$ according to the probabilities, respectively, $p_0 = p, p_1 = p < \frac{1}{2}$ and $p_2 = 1 - 2p > 0$).

Step 2: To see that the above ranking constitutes a unique limit, we show that no other ranking satisfies Eq. (3) and that, for any other ranking, whenever an item is in the less numerous class (say M_1 when $M_1 < \frac{M}{1+\beta}$), it will always get strictly more clicks in expectation than the item of the other more numerous class ranked just above. Assume $M_1 < \frac{M}{1+\beta}$, then it can be checked that the two strongest constraints to be satisfied are the ones comparing the clicking probability on the lowest-ranked item of class M_1 when (i) it is ranked in the Mth position while all remaining $M_1 - 1$ items are ranked in the first $M_1 - 1$ positions, and when (ii) it is ranked in position $M_1 + 2$ while all remaining $M_1 - 1$ items are ranked in the first $M_1 - 1$ positions. Both are easily seen to be satisfied whenever $p = 0$ and $M_1 < \frac{M}{1+\beta}$. By continuity they continue to hold for sufficiently small $p > 0$. The case $M_1 > \frac{\beta M}{1+\beta}$, which implies $M_0 < \frac{M}{1+\beta}$ and which has the M_0 items ranked on top holds by symmetry.

Step 3. It suffices to show that whenever $M_1 < \frac{M}{1+\beta}$, then the first ranked items in M_1 always get strictly more that half the share of the total clicks. Since the share of traffic on the individual items is given by the probabilities $\rho_{n,k}(r_\infty)$ defined above, it suffices to show that for any $M_1 < \frac{M}{1+\beta}$, $\rho_{n,M_1}(r_\infty) = \sum_{k=1}^{M_1} \rho_{n,k}(r_\infty) > \frac{1}{2}$. Given our assumptions on p and Γ, this is easily checked for the expected individual n (using the fact that, for any $1 \le K < K' \le M$, $\sum_{k=1}^{K} \beta^{M-K}/K > \sum_{k=1}^{K'} \beta^{M-K'}/K'$). It also implies that when $M_1' > \frac{\beta M}{1+\beta}$ all items in M_1' will (be bottom-ranked) and will obtain less than half the total traffic. □

When the share of items in M_1 is close to one half ($\frac{M}{1+\beta} < M_1 < \frac{M}{2}$), then there may be multiple limit rankings and the above proof no longer applies. We believe that the few-get-richer result may still go through in these cases, but it is necessary to evaluate the likelihood of the different limit rankings and guarantee that limit rankings are more likely to give classes with fewer items a higher probability of being higher ranked. This proof goes beyond the scope of this paper.

Similarly, for interior types ($\Gamma_0 < 1, \Gamma_1 > 0$). As Figure 1 shows for different values of β (see also the simulations in the next section), the effect continues to hold: when M_1 has few items (minority case $M_1 < \frac{M}{1+\beta}$) it obtains a larger proportion of clicks than when M_1 has many items (majority case $M_1 > \frac{\beta M}{1+\beta}$).

4 SIMULATIONS OF THE MODEL

We now analyze through simulations the presence of the few-get-richer effect in different settings. Our focus is on analyzing the click-through rate (CTR), defined as the the ratio of the probability of clicking on an item in M_1 to the total number of clicks N (which we fix at $N = 100$ users), as a function of M_1 for different settings. We mainly consider stochastic choices with $\Gamma = \{0.9, 0.1, 0.5\}$, which means that non-indifferent users do not exclusively click on one of the two classes. We assume that the ranking r_n is proportional to the number of clicks that the item received at time n. As before, we consider a ranking of $M = 20$ items with the M_1 items initially at the bottom. In this case, instead of the limit ranking, we characterize the dynamic transient during which the minority class may reach the top of the ranking. This makes our analysis dependent on the initial conditions. For an item m initially at position $r_{1,m}$, we assume a *uniform* initialization, with all items having one click.

4.1 Dependence on β

We first consider the dependence on the ranking effect, parametrized by β. Larger values of β correspond to a stronger relative ranking

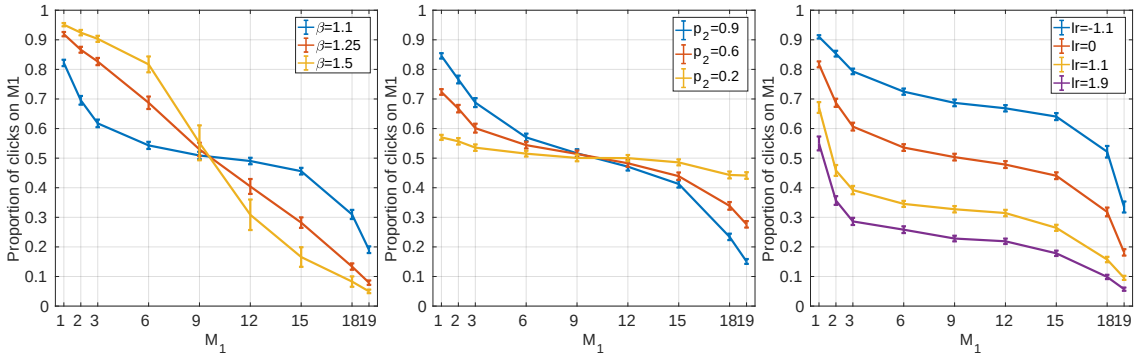


Figure 2: Proportion of clicks on all M_1 items of class 1 as a function of M_1 for runs of $N = 100$, and assuming $M = 20$, $\beta = 1.1, \Gamma = \{0.9, 0.1, 0.5\}, p_0 = p_1 = 0.4$ and uniform initialization, as baseline, and varying β (left plot), $p_2 = 1 - p_0 - p_1$ (middle plot), and the log-ratio $\log \frac{p_0}{p_1}$ (right plot). Error bars indicate confidence intervals for 100 different random realizations.

effect compared to the propensities. Here, we consider a symmetric case with $p_0 = p_1 = p = 0.4$.

Figure 2 (left) shows the CTR for different values of β. The resulting CTR is almost symmetric. We observe a monotonic decrease as a function of M_1 in all cases, showing, in particular, that the minority class *always* receives more clicks than the majority one.

We can differentiate between three cases, corresponding to small, intermediate and large values of β, respectively. In general, larger values of β lead to a relatively larger CTR for the minority class, due to relatively larger probability of clicking on top ranked items. For small values of β, the CTR is the smallest, but even here the effect as a function of M_1 is particularly pronounced for small (and large) M_1. For intermediate values of β, the flat region for intermediate values of M_1 disappears and the CTR decreases monotonically with constant slope, indicating a decrease independent of M_1. Finally, for larger values of β, the CTR is largest, and the effect as a function of M_1 is smallest for small (and large) M_1. In this extreme case, for $M_1 = 1$, the single item at the bottom quickly reaches the top and attracts almost all the traffic, leaving almost no traffic to the remaining $M - 1$ items of the other class.

From these simulations, we can conclude that the few-get-richer effect is robust to varying β.

4.2 Dependence on p_2

We now analyze the CTR as a function of the proportion of indifferent users $p_2 = 1 - p_0 - p_1$. For this, we choose $\beta = 1.1$ and vary the proportion of indifferent users while keeping $p_0 = p_1$. To better analyze this dependence, we consider extreme preferences, so that only indifferent users can really click on both items, that is, we assume $\Gamma = \{1, 0, 0.5\}$.

Figure 2(middle) shows the results. We observe that, in this case, the few-get-richer effect is more pronounced as p_2 increases. Having a larger proportion of indifferent users results in relatively more clicks per item, and hence a larger amplification of ranking effect, which is a key ingredient for the few-get-richer effect to emerge. With less extreme preferences, e.g., $\Gamma = \{0.9, 0.1, 0.5\}$, the effect in p_2 is still present, but less pronounced (data not shown).

We conclude that the indifferent users play a key role in amplifying the effect of the ranking, and that in general, having a larger proportion of them contributes importantly towards the few-get-richer effect.

4.3 Dependence on the ratio $\frac{p_0}{p_1}$

So far we have considered cases where the distribution of user types was symmetric, $p_0 = p_1 = p$. In practice this need not be the case since, e.g., the minority class might also be preferred by a minority of users. We analyze the effect as a function of $lr = \log \frac{p_0}{p_1}$.

Figure 2(right) shows the results for different values of lr. As expected, the few-get-richer effect is more pronounced when there is a larger relative proportion of users that prefer the minority item. To see this, compare the blue line, where $p_1 = 0.6 > p_0 = 0.2$ (there are three times more users who prefer the 'minority' item), and the purple line, where $p_1 = 0.1 < p_0 = 0.7$. We see that the effect is still present for this choice of parameters, even when there are seven times more users who prefer the majority item. Here, the proportion of indifferent users is set to $p_2 = 0.2$. Consistent with the results of the previous subsection, if we increase p_2 and keep the same ratios, the effect becomes more pronounced. We conclude that the effect is also robust to different ratios of proportions lr.

4.4 Ranking evolution

Finally, we illustrate the typical behavior of the ranking evolution for different values of M_1, assuming our usual baseline parameter values for β, Γ, and p and uniform initialization. Figure 3 shows the results, which confirm the idea that minority items tend to move towards the top.

5 ONLINE EXPERIMENT WITH HUMAN PARTICIPANTS

To test the predictions of the model, we executed an online experiment in which participants clicked on one out of 20 possible options (pictures).[1] The options belonged to two classes, M_0 and M_1, just as in the model. As in the simulations reported above, we measure the effect of the popularity-based ranking of options as the total number of clicks attracted by items of type M_1. To this end,

[1]Data from the experiment are available at https://osf.io/nwjyf/.

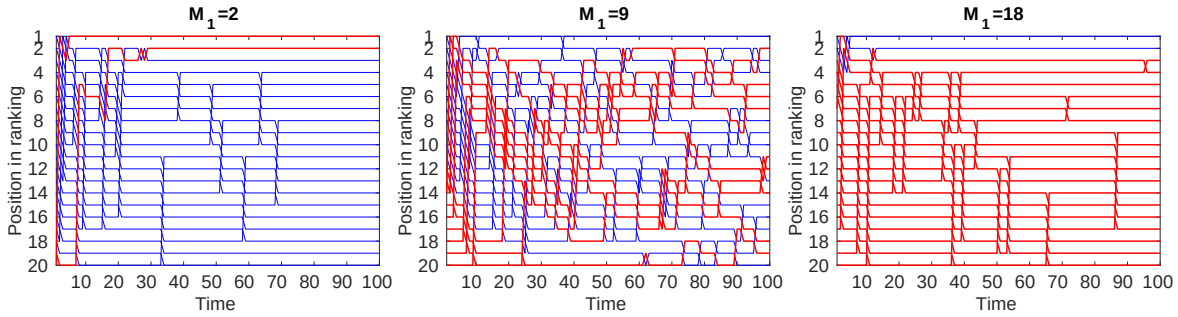


Figure 3: Examples of ranking evolution for different values of M_1. The items in M_1 always start at the bottom of the ranking. With $M_1 = 2$ (left plot), the two items quickly move to the top. With $M_1 = 9$ (middle plot), the items in M_1 are spread through the different ranking positions. Finally, with $M_1 = 18$ (or $M_0 = 2$) (right plot), the items M_1 eventually stay at the bottom.

we created 8 independent conditions. In conditions D1 to D4, the ordering of the options changed dynamically as a function of the number of clicks received by each option, with the most popular option (in terms of cumulative number of clicks) at the top of the screen and the least popular option at the bottom of the screen. The conditions differed in terms of the number of options in M_0 and M_1, as summarized in Table 1. The number of options in M_1 was 17, 12, 8, or 3 in conditions D1, D2, D3 and D4, respectively.

These four 'dynamic' conditions were matched to four 'static' conditions with the same sets of options M_0 and M_1. In the static conditions, the ranking of options did not change over time and was set to the initial ranking in the matching dynamic condition. We denote these static conditions by S1, S2, S3, and S4.

Comparisons of matched pairs of conditions allows us to establish the causal effect of the ranking algorithm on the total traffic attracted by each option. Comparisons of the 4 dynamic conditions allows us to test the prediction that the total traffic attracted by options in M_1 decreases with the number of options in M_1.

To keep the experimental setup as simple as possible, the options in our experiment were not news sources, but pictures of dogs and cats. In order to activate their preferences for cat or dog pictures, participants were first asked whether they were a 'cat person', a 'dog person', or 'neither a cat person nor a dog person.' Answers to this identity question allowed us to compute the proportion of participants in each of the 3 types discussed in the model section. Then participants were shown a set of 20 pictures in a vertical list. In this setup, M_0 is the set of cat pictures (initially at the top of the screen) and M_1 is the set of dog pictures (initially at the bottom). The initial popularity of all pictures was set to 1, consistent with the uniform initialization of the previous section.

5.1 Methodological Details

We recruited 786 participants on Amazon Mechanical Turk. It was administered via a Qualtrics survey embedded in the Amazon Mechanical Turk webpage as an iframe. After signing up for the task, participants read the informed-consent form. Then they were randomly assigned to one of 8 conditions. They first answered a question about their type: "Are you more of a cat person or a dog person?" with three possible choices "I am a cat person" / "I am neither a cat person nor a dog person" / "I am a dog person." On the next screen they were shown 20 buttons with "Please click on a photo from the following list of photos of cats and dogs and rate it according to your liking." The buttons were displayed in a vertical list. Participants could initially see 3 to 4 buttons and had to scroll down to access the other buttons. After clicking a button, participants were shown the corresponding picture and gave it a rating of 1 to 5 stars. The rating task was presented as a reason to ask participants to select an item according to their preferences. The collected ratings are not discussed because our interest is only in the clicking behavior of the participant. Participants were paid $0.15 for their time.

5.2 Results

5.2.1 Participant Types. 30% indicated they were 'Cat persons' ($p_0 = 0.30$), 55% 'Dog persons' ($p_1 = 0.55$), 15% neither ($p_2 = 0.15$).

5.2.2 Total Traffic Attracted by M_1 (Dog pictures). Dog pictures were initially ranked at the bottom the screen. First we discuss the effect of popularity-based ranking of options on the total traffic attracted by Dog pictures. In the 4 dynamic conditions, Dog pictures attracted substantially more traffic than in the corresponding static condition (compare the two rows 'Experiment' in the top and bottom panels of Table 1). In other words, ordering the options in terms of popularity had a systematic effect on the share of traffic attracted by options that started at the bottom of the choice screen.

Unsurprisingly, in all static conditions, the total traffic attracted by Dog pictures was lower than 50%. The set of Dog pictures attracted more traffic when there were relatively more Dog pictures. For example, while 17 Dog pictures attracted 44% of the traffic, 3 Dog pictures attracted 27%.

The pattern is completely different in the dynamic conditions. First, in all dynamic conditions, the total traffic attracted by Dog pictures was higher than 50%. The most important finding is that the total traffic attracted by Dog pictures was *larger* with just 3 Dog pictures (and 17 Cat pictures) than with 17 Dog pictures (and 3 Cat pictures)! Similarly, the total traffic attracted by Dog pictures was *larger* with 8 Dog pictures (and 12 Cat pictures) than with 12 Dog pictures (and 8 Cat pictures). These results are consistent with the predictions of our model. The total traffic attracted by Dog pictures did not decrease monotonically with the size of the set of Dog pictures (it is larger with 8 Dog pictures than with 3 Dog pictures). This pattern is seemingly inconsistent with the predictions of the

Table 1: Experiment results. The 'Sim1' rows report the average traffic attracted by Dog pictures (M_1) over 1000 simulations of the choice model with a setting matching the exact number of participants of each identity type in each condition. The 'Sim2' rows report the average traffic attracted by Dog pictures (M_1) over 1000 simulations of the choice model with 100 users in each condition where the numbers of users who are a 'dog person', 'neither a dog person nor a cat person' and a 'cat person' are 55, 15 and 30, respectively (same frequencies for all conditions).

# Cats (M_0)		3	8	12	17
# Dogs (M_1)		17	12	8	3
Dynamic Button Ordering					
Condition		D1	D2	D3	D4
# participants		96	102	99	101
# participants in each type	Cat person	34	30	24	29
	Neither	9	21	11	16
	Dog person	53	51	64	56
Dog traffic share	Experiment	.53	.69	.76	.71
	Sim1	.46	.56	.73	.76
	Sim2	.47	.60	.67	.75
Static Button Ordering					
Condition		S1	S2	S3	S4
# participants		96	101	95	96
# participants in each type	Cat person	34	30	25	33
	Neither	13	19	9	15
	Dog person	49	52	61	48
Dog traffic share	Experiment	.44	.37	.40	.27
	Sim1	.41	.37	.39	.28
	Sim2	.44	.39	.35	.30

simulations (Fig. 2). Yet, this lack of monotonicity can be explained by the fact that the conditions were not perfectly balanced with respect to the distribution of identity types of the participants. In condition D3 (8 Dog pictures) there were substantially more Dog lovers (64/99) than in condition D4 (3 Dog pictures, 56/101). This is a chance event resulting from the fact that we elicited the identity type of the participants after the random assignment into conditions, thus we could not balance types within conditions.

5.2.3 Estimated Parameters of the Choice Model. We estimated the parameters of the stochastic choice model, Eq. (2), using maximum likelihood on the whole dataset. Our parameter estimates are $\beta = 1.22$, $\Gamma_C = .74$ for Cat persons (Γ_0 in the model), and $\Gamma_D = .08$ for Dog persons (Γ_1 in the model). Cat and Dog persons had strong tendencies to choose pictures from the class consistent with their identity type.

We used these estimated parameters to simulate choices and the dynamics of picture ranks in the 8 experimental conditions. Simulations reported in rows 'Sim1' in Table 1 indicate a close match between the simulated data and the actual traffic proportions attracted by Dog pictures (both in static and dynamics conditions).

We also used the estimated parameters to simulate what would happen if there were 100 participants in each condition and if the distributions of identity types were the same in all conditions ('Sim2' rows). We find a decreasing monotonous relation between the number of Dog pictures and the share of traffic attracted by Dog pictures. This pattern is consistent with the qualitative prediction of our model.

5.2.4 Summary. Overall, the results indicate that the experimental setting falls within the boundary condition of application of our theory. They provide a proof-of-concept that popularity-based ordering of options can lead to the counter-intuitive phenomenon that when there are fewer options of one class, the total share of traffic attracted by this class of options becomes larger.

6 DISCUSSION & CONCLUSION

The few-get-richer effect can have both positive and negative effects on the quality of the information people obtain from search results. On the positive side, when there are few relevant items, the few-get-richer effect may help them become top-ranked, making them more accessible. At the same time, the few-get-richer effect can contribute to the spread of misinformation. It may help few irrelevant or 'fake news' items become top-ranked, especially if there is a strong preference for such items and only few websites report them.

Our analyses highlight a potential unintended effect of regulations that 'ban' particular 'alternative' news sources known for spreading 'fake news'. When a sizeable proportion of users have a preference for identifiable 'alternative' news sources, the removal of some of these news sources might lead to an increase in the total traffic attracted by the remaining 'alternative' news sources. This could result in more overall exposure to 'fake news'!

To neutralize the few-get-richer effect, our theory suggests that it may be advisable to keep track of the number of items in each class when incorporating clicks in the search engine algorithm. Ideally, the ranking algorithm should use the popularity of the different items in a way that is neutral to the number of items in each class.

The few-get-richer effect also has implications for the design of recommender systems. The learning efficiency of these systems is impeded by the *presentation bias* problem: items shown to the user can get clicks whereas items not shown get no clicks. The recommender system thus cannot learn about the relevance of the latter items. A popular solution to this challenge is the *explore-and-exploit* approach, in which some items from a minority class are randomly inserted in the search results [1]. This purportedly increases the amount of exploration (clicks on the minority class), and thus increases the learning opportunities of the system at the cost of slightly hurting the user experience [18, 23]. The few-get-richer effect suggests precisely the opposite. Adding more of those items might reduce, rather than increase, the total amount of exploration.

REFERENCES

[1] Deepak Agarwal, Bee-Chung Chen, and Pradheep Elango. 2009. Explore/Exploit Schemes for Web Content Optimization. In *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining (ICDM '09)*. IEEE Computer Society, Washington, DC, USA, 1–10.

[2] Ricardo Baeza-Yates. 2018. Bias on the Web. *Commun. ACM* 61, 6 (May 2018), 54–61.

[3] Abhijit V Banerjee. 1992. A simple model of herd behavior. *The quarterly journal of economics* 107, 3 (1992), 797–817.

[4] Asia J. Biega, Krishna P. Gummadi, and Gerhard Weikum. 2018. Equity of Attention: Amortizing Individual Fairness in Rankings. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, 405–414.

[5] Sushil Bikhchandani, David Hirshleifer, and Ivo Welch. 1992. A theory of fads, fashion, custom, and cultural change as informational cascades. *Journal of political Economy* 100, 5 (1992), 992–1026.

[6] Carlos Castillo. 2019. Fairness and Transparency in Ranking. *SIGIR Forum* 52, 2 (Jan. 2019), 64–71.

[7] Olivier Chapelle and Ya Zhang. 2009. A Dynamic Bayesian Network Click Model for Web Search Ranking. In *Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. ACM, New York, NY, USA, 1–10.

[8] Yeon-Koo Che and Johannes Hörner. 2018. Recommender Systems as Mechanisms for Social Learning. *Quarterly Journal of Economics* 133, 2 (2018), 875–921.

[9] Justin Cheng, Michael Bernstein, Cristian Danescu-Niculescu-Mizil, and Jure Leskovec. 2017. Anyone Can Become a Troll: Causes of Trolling Behavior in Online Discussions. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17)*. ACM, New York, NY, USA, 1217–1230.

[10] Giovanni Luca Ciampaglia, Azadeh Nematzadeh, Filippo Menczer, and Alessandro Flammini. 2018. How algorithmic popularity bias hinders or promotes quality. *Scientific Reports* 8, 1 (2018), 15951.

[11] Pranav Dandekar, Ashish Goel, and David T. Lee. 2013. Biased assimilation, homophily, and the dynamics of polarization. *Proceedings of the National Academy of Sciences* 110, 15 (2013), 5791–5796.

[12] Gabrielle Demange. 2014. Collective attention and ranking methods. *Journal of Dynamics & Games* 1, 2164-6066 (2014), 17.

[13] Jerker Denrell and Gaël Le Mens. 2017. Information Sampling, Belief Synchronization, and Collective Illusions. *Management Science* 63, 2 (2017), 528–547.

[14] Alejandro Diaz. 2008. Through the Google Goggles: Sociopolitical Bias in Search Engine Design. In *Web Search: Multidisciplinary Perspectives*. Springer Berlin Heidelberg, Berlin, Heidelberg, 11–34.

[15] Georges E. Dupret and Benjamin Piwowarski. 2008. A User Browsing Model to Predict Search Engine Click Data from Past Observations.. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '08)*. ACM, New York, NY, USA, 331–338.

[16] Santo Fortunato, Alessandro Flammini, Filippo Menczer, and Alessandro Vespignani. 2006. Topical interests and the mitigation of search engine bias. *Proceedings of the National Academy of Sciences* 103, 34 (2006), 12684–12689.

[17] Fabrizio Germano and Francesco Sobbrio. 2018. Opinion Dynamics via Search Engines (and other Algorithmic Gatekeepers). *ArXiv e-prints* (Oct. 2018). arXiv:1810.06973

[18] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)* 22, 1 (2004), 5–53.

[19] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2017. Accurately Interpreting Clickthrough Data As Implicit Feedback. In *SIGIR Forum*, Vol. 51. ACM, New York, NY, USA, 4–11.

[20] Jooyeon Kim, Behzad Tabibian, Alice Oh, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. 2018. Leveraging the Crowd to Detect and Reduce the Spread of Fake News and Misinformation. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 324–332.

[21] Ilan Kremer, Yishay Mansour, and Motty Perry. 2014. Implementing the "Wisdom of the Crowd". *Journal of Political Economy* 122, 5 (2014), 988–1012.

[22] Gaël Le Mens, Jerker Denrell, Balázs Kovács, and Hülya Karaman. 2018. Information Sampling, Judgment, and the Environment: Application to the Effect of Popularity on Evaluations. *Topics in Cognitive Science* (2018). https://doi.org/10.1111/tops.12387

[23] Sean M. McNee, John Riedl, and Joseph A. Konstan. 2006. Being Accurate is Not Enough: How Accuracy Metrics Have Hurt Recommender Systems. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems (CHI EA '06)*. ACM, New York, NY, USA, 1097–1101.

[24] Gaël Le Mens, Balázs Kovács, Judith Avrahami, and Yaakov Kareev. 2018. How Endogenous Crowd Formation Undermines the Wisdom of the Crowd in Online Ratings. *Psychological Science* 29, 9 (2018), 1475–1490.

[25] Robert K Merton. 1968. The Matthew effect in science: The reward and communication systems of science are considered. *Science* 159, 3810 (1968), 56–63.

[26] Ashley Muddiman and Natalie Jomini Stroud. 2017. News values, cognitive biases, and partisan incivility in comment sections. *Journal of communication* 67, 4 (2017), 586–609.

[27] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting Clicks: Estimating the Click-through Rate for New Ads. In *Proceedings of the 16th International Conference on World Wide Web (WWW '07)*. ACM, New York, NY, USA, 521–530.

[28] Matthew J Salganik, Peter Sheridan Dodds, and Duncan J Watts. 2006. Experimental study of inequality and unpredictability in an artificial cultural market. *science* 311, 5762 (2006), 854–856.

[29] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science* 359, 6380 (2018), 1146–1151.

[30] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position Bias Estimation for Unbiased Learning to Rank in Personal Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 610–618.

[31] Lillian Weng, Alessandro Flammini, Alessandro Vespignani, and Fillipo Menczer. 2012. Competition among memes in a world with limited attention. *Scientific reports* 2 (2012), 335.

[32] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. 2017. FA*IR: A Fair Top-k Ranking Algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. ACM, New York, NY, USA, 1569–1578.

[33] Yuchen Zhang, Weizhu Chen, Dong Wang, and Qiang Yang. 2011. User-click Modeling for Understanding and Predicting Search-behavior. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*. ACM, New York, NY, USA, 1388–1396.

Context-Sensitive Malicious Spelling Error Correction

Hongyu Gong, Yuchen Li, Suma Bhat, Pramod Viswanath
University of Illinois at Urbana-Champaign
{hgong6,li215,spbhat2,pramodv}@illinois.edu

ABSTRACT

Misspelled words of the malicious kind work by changing specific keywords and are intended to thwart existing automated applications for cyber-environment control such as harassing content detection on the Internet and email spam detection. In this paper, we focus on malicious spelling correction, which requires an approach that relies on the context and the surface forms of targeted keywords. In the context of two applications–profanity detection and email spam detection–we show that malicious misspellings seriously degrade their performance. We then propose a context-sensitive approach for malicious spelling correction using word embeddings and demonstrate its superior performance compared to state-of-the-art spell checkers.

CCS CONCEPTS

• **Computing methodologies** → *Lexical semantics*; *Unsupervised learning*.

KEYWORDS

Malicious spelling correction; cyberbullying; machine learning

ACM Reference Format:
Hongyu Gong, Yuchen Li, Suma Bhat, Pramod Viswanath. 2019. Context-Sensitive Malicious Spelling Error Correction. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313431

1 INTRODUCTION

Automatic spelling correction has been an important natural language processing component of any input text interface of today [17]. While spelling errors in common writing could be regarded as innocuous omissions, affecting opinions about the writer's competence at worst, those of the malicious kind can potentially be highly offensive, since they are intended to deceive an automated mechanism–be it a spam filter for emails or a profanity detector for social media. For those applications that rely on automatic detection of specific keywords to enable real-time control of the cyber-environment, detecting and correcting spelling errors is of primary importance.

Perspective [11], is one such application that helps reduce abusive content online by detecting profane language. It has been pointed out that Perspective's otherwise good performance in detecting toxic comments is not robust to spelling errors [15]. Consider the sentence, "My thoughts are that people should stop being stupid." which was assigned a perceived toxicity score of 86% by Perspective. When the word "stupid" was misspelled as "stup*d" in the same sentence, its toxicity score reduces to 8%. This marked reduction induced by the deceptive spelling error on the keyword 'stupid' reflects the importance of accurate spelling correction for optimal toxicity detection. Had the spelling error been detected and corrected *before* scoring for toxicity, the score would not have lowered.

Likewise, deliberate typographic errors are common in phishing scams and spam emails. For instance, "Please pya any fees you may owe to our company" where *pya* is clearly a spelling error, included to deceive spam filters that are sensitive to keywords. In both these instances, spelling correction as a preprocessing step of the messages are critical for a robust performance of the target system. Spelling correction in the preprocessing stage in malicious settings constitutes the focus of this study.

In this paper, we empirically demonstrate the effect of spelling errors in a malicious setting by adding synthetic misspellings to sensitive words in the context of two applications–profanity detection and spam detection. We propose an *unsupervised* embedding-based algorithm to correct the targeted misspelled words. Earlier approaches to spelling correction primarily depend on edit distances to find words morphologically similar to corrections. More recently, improved spell checkers with the addition of contextual information (e.g., n-gram modeling in [37]), often with intensive computation and memory requirements (to obtain and store N-gram statistics). A recently studied neural network-based spell checker learns the misspelling pattern from annotated train data in a supervised way, and was found to be sensitive to data domains and dependent on human annotations [9]. Our approach to make the correction procedure context-aware involves harnessing the geometric properties of word embeddings. It is light-weight in comparison, unsupervised, and can be adapted to a variety of data domains including Twitter data and spam data, since domain information can be well captured by tuning embeddings on domain-specific data [33].

We first demonstrate the effect of spelling errors in a malicious setting in two applications–profanity detection and spam detection. Then we propose an unsupervised algorithm to correct the targeted spelling errors. Besides performing well on the correction of synthetic spelling errors, the algorithm also performs well on spell checking of real data, and shows that it can improve hate speech detection using Twitter data.

2 RELATED WORK

Spelling correction. Automatic spelling correction in non-malicious settings has had a long research track as an important component of text processing and normalization. Early works have used edit

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313431

distance to find morphologically similar corrections [29], noisy channel model for misspellings [17], and iterative search to improve corrections of distant spelling errors [12]. Word contexts have been shown to be improve the robustness of spell checkers with n-gram language model as one approach to incorporate contextual information [8, 14]. Other ways of incorporating contextual information include n-gram statistics capturing the cohesiveness of a candidate word with the given context [37].

Effect of malicious misspellings. Misspellings are commonly seen in online social platforms such as Twitter and Facebook, and recent studies have drawn attention to the fact that online users deliberately introduce spelling errors to thwart bullying detection [27] or to challenge to moderators of online communities [26]. This is because simple ways of filtering terms of profanity are rendered inadequate in the presence of spelling errors. Likewise, email spam filters are often obfuscated by misspelled sensitive words [38] because word-based spam filters tend to make false decisions in the presence of misspellings [32], and so are word-based cyberbullying detection systems [3]. Misspelling is also seen in web attacks where a phishing site has a domain name as a misspelling of a legitimate website [16].

Approaches to correct malicious misspellings. Given the malicious intent of misspellings in the context of the Internet, recent studies have proposed correction strategies specifically for malicious misspellings. Levenshtein edit distance is commonly used in spell checking of detection systems. Spam filters [38] and cyberbullying detection systems [36] rely on the idea of edit distance to recognize camouflaged words by capturing the pattern of misspellings. For example, [30] studied an edit distance function designed to reveal instances where spammers had interspersed black-listed words with non-alphabetic symbols. Lexical and context statistics, such as word frequency, have also been used to make corrections in social texts [5, 18]. Existing approaches have the problem of domain specificity, since their lexical statistics are obtained from a certain domain which might differ from the domain of target applications.

Our study considers the spelling correction in a malicious setting where errors are not random, but are carefully introduced. Our context-aware approach to spelling correction relies on the geometry of word embeddings, which has the advantage of efficient domain adaptation. As we will show in Section 4.2, the embedding can be easily tuned on a small corpus from the target domain to capture domain knowledge.

3 METHODS

Spelling errors include non-word errors and real-word errors [28]. Non-word misspelling is not a valid word in standard dictionaries, while real-word misspelling is. For example, *piecce* is a non-word error of *piece*, and *peace* is its real-word misspelling. In this work, we focus on non-word misspellings. We study malicious spelling correction for three target applications–toxicity detection using the Perspective API, email spam detection, and hate speech detection on Twitter data. We work with synthetic spelling errors in the first two applications, and study the real misspellings in the third.

For the toxicity detection task, we use the Perspective dataset [11], which provides a set of comments collected from the Internet with human annotated scores of toxicity. Using the Perspective API we obtained the toxicity score for 2767 comments in the dataset.

The spam dataset consists of 960 emails from the Ling-Spam dataset [1]. We randomly split the data into 700 train emails and 260 test emails. Both the train and test sets were balanced with respect to the positive and negative classes. The most frequent 2500 words in the training data were selected, and we counted the occurrence of each word in the emails. The number of occurrences of these frequent words were used as a 2500-dimension feature vector for each email. We first used these features to train a Naive Bayes classifier. It achieved an accuracy of 98% in spam detection.

As for the Twitter data, a total of 16K user posts were collected from Twitter, a social communication platform and used in used in [34]. Of these tweets, 1937 were labeled as being racist, 3117 were labeled as being sexist, and the rest were neither racist nor sexist. The task of hate speech detection is to classify these tweets into one of the three categories *racist, sexist, neither*. We randomly split the tweets into train and test data, and trained a neural-network based hate speech detection system on the training data (described later in Section 5.3).

3.1 Characterization

Here we summarize our assumptions on the characteristics of the malicious spelling errors.

(1) Malicious errors are usually made on the sensitive keywords in order to obfuscate the true intentions and deceive detection systems that rely on keywords [35].

(2) The error yields a word that is similar to the original word in surface form. The misspelled words would be words that humans can easily understand (thus the erroneous words still convey the intended meaning, while being in disguise). Their similarity is reflected in the small edit distance between the erroneous and the correct word [24]. The errors often involve character-level operations as shown in Table 1 [21].

(3) With edit distance as the similarity criterion, it is often the case that the word with an error is similar to multiple valid words, making correction a challenge in real applications. For example, both *stud* and *stupid* can be thought of as the correct form of the erroneous word *stupd*. In such cases, spelling correction relies on the context in which the word occurs.

3.2 Effect of malicious spelling errors

We quantify the effect of spelling errors by showing that a simple mechanism of injecting malicious spelling errors greatly degrades the performance of toxicity detection and spam detection. Toward this, we first describe the general mechanism to generate synthetic errors and then study their impact on toxicity and spam detection. We point out that owing to the absence of a dataset with intended malicious errors, we had to generate them synthetically.

Spelling error generation. We first choose the "sensitive" words that the detection algorithm relies (the mechanism will be discussed separately for each application we consider). We then replace these words with their erroneous forms (which may not be real words). Given the characteristics of malicious errors mentioned in Section 3.1, our assumption is that as a result of these errors the altered

[1] http://csmining.org/index.php/ling-spam-datasets.html

words will appear similar to the original ones. As illustrated in Table 1, we consider four basic character-level operations to change the sensitive words: insertion, permutation, replacement, and removal. In our experiments we perform these operations on randomly picked characters of the sensitive word. For permutation, we exchange this character with the next one. Each operation in Table 1 increases the edit distance by 1, and we generate malicious misspellings of the sensitive word that are at most 2 edit distance from the correct word.

Table 1: Common Spelling Errors

Type	insertion	permutation	replacement	removal
Error	idio.t	moeny	chanse	stupd
Correction	idiot	money	chance	stupid

Finally, we obtain revised sentences by replacing the sensitive words in the original sentences with their erroneous counterparts. Next, we introduce our mechanism for selecting the "sensitive" words for each application.

Perspective toxicity detection. We select 2767 clean comments from a total of 2969 comments in the Perspective data, with the selection criteria given below.

- The most toxic word in a comment should contain more than two characters. A word that is too short is not likely to be a meaningful toxic word, and can have too many candidate corrections in the dictionary.
- The most toxic word in a comment should appear at least 100 times in the dataset, since rare words tend to be misspellings.
- The most toxic word in a comment should be a content word, i.e., it should not belong in the list of function words such as "must", "ought", "shall" and"should". Function words are not toxic and so were excluded from our experiments.

For each comment, its predicted toxicity score lies in the range of $[0, 1]$. The higher the score, the more toxic the sentence is. For each sentence, we chose the most toxic word to be that word, without which the toxicity score reduced the most. We then added malicious errors, as described above, to this word. We note that for 2380 toxic comments with toxicity scores higher than 0.5, spelling errors brought down their predicted toxicity by 32%. Section 5 provides the details of the degradation in toxicity detection.

Spam detection. A Naive Bayes spam detection model provides the likelihood of each word given the spam and non-spam classes. For a given word, the difference between these probabilities reflects how important that word is in spam detection. We sorted all the words in decreasing order of this difference in probabilities, and picked the most spam-indicative words. We then added malicious errors to these words in test spam emails. For a spam filtering system which relies heavily on the counts of spam-indicative words, the errors can easily disguise spams as non-spams. This is seen in the test accuracy dropping from 98% on the original test data to 72% on the revised test data.

We note that for the Perspective data, we added errors to only one word per sentence, but did not have such limits for the spam data. Also, we do not limit the number of corrections during the spell check process.

4 SPELLING CORRECTION

We have shown that malicious errors degrade the performance of toxicity and spam detection. We next introduce our unsupervised algorithm on non-word error correction based on the relevant context. Our correction method proceeds in two stages. In the first stage of **candidate enumeration**, the algorithm detects the spelling error and proposes candidates that are valid words and similar in surface form to the misspelled word. In the next stage, **correction via context**, the best candidate is chosen based on its context.

4.1 Candidate Enumeration

As in the case of non-word spelling correction [17], we check whether an erroneous word is valid or not using a vocabulary of valid words. For toxicity detection and spam detection, the vocabulary consists of words that occur more than 100 times in English Wikipedia corpus [2] For hate speech detection, the vocabulary consists of standard words from Wikipedia and the list of internet slang words as detailed in Section 5.3, since slang words frequently occur in tweets. For an invalid word, we find candidate words with the smallest Damerau-Levenshtein edit distance [7, 20]. This distance between two strings is defined as the minimum number of unit operations required to transform a string into another, where the unit operations include character addition, deletion, transposition and replacement. We note that because there are potentially multiple candidates having the smallest distance to the misspelled word, the output of this stage is a set of candidate corrected words.

4.2 Correction via Context

After enumerating all possible candidates, we choose the one that fits the best in the context. We propose a word-embedding-based method to match the context with the candidate.

Pretrained embedding. We use the word2vec CBOW model to train word embeddings on the training data of the three applications [22]. The Perspective and Twitter data have an informal style, while email data consists of relatively formal expressions. Word embeddings are known to be domain-specific [25] and naturally domain-specific corpora are used to train word embeddings for use in a given domain. We note that in the absence of a large enough representative corpus to train domain-specific high-quality embeddings for this study, we reconcile with word embeddings trained on a large WikiCorpus [4] to capture the general lexical semantics and further tuned on a domain-specific corpus, such as that of the Perspective dataset, the spam emails data and the Twitter data. This step allows us to combine domain information in trained embeddings.

Error Correction. Word vectors permit us to decide the fit of a word in a given context by considering the geometry of the context words in relation to the given word. Firstly, the embedding of the compositional semantics of phrases or sentences can be approximated by a linear combination of the embeddings of their constituent words [31]. Let v_w be the embedding of the token w. Take the phrase "hate group" as an example, it holds that

$$v_{\text{hate_group}} \approx \alpha v_{\text{hate}} + \beta v_{\text{group}},$$

where α and β are real-valued coefficients.

[2] available at: http://www.cs.upc.edu/~nlp/wikicorpus/

Table 2: Correction Accuracy on Perspective and Spam

Dataset	Our system	PyEnchant	Ekphrasis	Google
Perspective	**0.840**	0.476	0.713	0.323
Spam	**0.786**	0.618	0.639	0.526

This enables us to represent the context as a linear space of the component words. Another property is that semantically relevant words lie close in the embedding space. If a word fits the the context, it should be close to the context space, i.e., the normalized Euclidean distance from the word embedding to the context space should be small [10]. We quantify the inconsistency between the word and its context by the distance between the word embedding and the span of the context words.

For a misspelled word in a sentence, all words occurring within a window of size p are considered to be its context words. Let the context T_p be the set of words within distance p from w_0:

$$T_p = \{w_{-p}, w_{-p+1}, \ldots, w_0, \ldots, w_{p-1}, w_p\},$$

where w_0 is the misspelled word. Let C be the set of candidate replacements of w_0. Let v_i be the word embedding of context word w_i. We denote by $\text{dist}(c, T_p)$, the distance between a candidate c and the linear span of the words in its context T_p, termed as the candidate-context distance of a candidate, defined as the normalized distance between the word embedding of the candidate, v_c and its linear approximation obtained by the context vectors:

$$\text{dist}(c, T_p) = \min_{\{a_i\}} \frac{1}{\|v_c\|_2} \| \sum_{\substack{i=-p, \\ i\neq 0}}^{p} a_i v_i - v_c \|_2^2. \quad (1)$$

This is a quadratic minimization problem for which we can find a closed-form solution.

Instead of fixing one context window size, we consider multiple window sizes and weigh the distances obtained using different window sizes. The context words that are closer to the misspelled word are more informative in candidate selection. In the sentence "the stu*pid and stubborn administrators", the word *stubborn* suggests that the misspelled word should be a negative personality adjective, and the word *administrator* that it should be an adjective for people. Thus, the closest context word *stubborn* provides more relevant information than the distant word *administrator*.

We thus weigh the distance by the inverse of the context window size, i.e., the weight $\frac{1}{p}$ for the window size p. Suppose that T is the full context, and the candidate-context distance is defined below:

$$\text{dist}(c, T) = \sum_{p=1}^{P} \frac{1}{p} \cdot \text{dist}(c, T_p), \quad (2)$$

where $P = 4$ in our experiments.

Given the context T, the suggested correction c^* is the candidate with the smallest distance to the linear space of context words, i.e.,

$$c^* = \arg\min_{c \in C} \text{dist}(c, T). \quad (3)$$

5 EXPERIMENTS

We evaluated our spelling correction approach in three settings: toxicity and spam detection with synthetic misspellings, and hate speech detection with real errors. Even though some recent works using neural networks [9] are available, they require ground truth corrections for supervised training. For our unsupervised approach, we compare its performance with that of three strong unsupervised baselines below, which are generic off-the-shelf tools used in many NLP systems such as AbiWord [1] and Google search engine [2].
(1) Pyenchant: a generic spell checking library of multiple correction algorithms [19]. We use the MySpell library in this work.
(2) Ekphrasis: a spelling correction and text normalization tool for texts from social networks [5].
(3) Google spell checker: a Google search engine-based spell check [6]. We chose it for its ability to undertake context-sensitive correction on the basis of user search history.

Table 3: Correction on Perspective Data

Original	**stupid** and stubborn administrators	anti American **hate** groups
Revised	**stu*pid** and stubborn administrators	anti American **ahte** groups
Our result	**stupid** and stubborn administrators	anti American **hate** groups
PyEnchant	**sch*pid** and stubborn administrators	anti American **hate** groups
Ekphrasis	**stupid** and stubborn administrators	anti American **ate** groups
Google	**stupid** and stubborn administrators	anti American **ahte** groups

5.1 Toxicity Detection

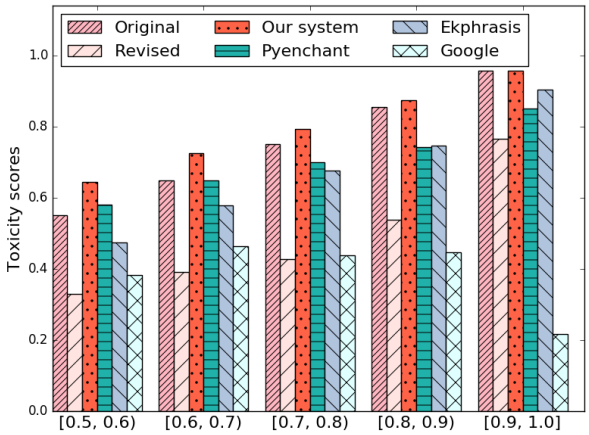


Figure 1: Toxicity scores by different approaches.

We took 2380 sentences from Perspective dataset whose toxicity scores were higher than 0.5 and added synthetic errors maliciously to these sentences as described in Section 3. Some example Perspective sentences, revised sentences and corrections given by different algorithms are shown in Table 3. The correction accuracy achieved by different approaches is reported in Table 2.

We divided toxic sentences into 5 bins according to their original toxicities. In Fig. 1, we report the average toxicity of the original, revised and corrected sentences in each bin. We see that the malicious errors drastically reduce the sentences' toxicities. The original average toxicity in the first bin is 0.55, whereas the revised toxicity is only 0.33 (a drop by 40%). This shows that toxicity detection is very sensitive to spelling errors.

Table 4: Correction Examples on Spam Data

Original sentence	it really be a **great** oppotunity to make relatively easy **money** , with little **cost** to you .	we have quit our **jobs** , and will soon buy a home on the beach and live off the interest on our **money** .
Revised sentence	it really be a **grfat** opportunity to make relatively easy **fmoney**, with little **cosgt** to you.	we have quit our **tobs**, and will soon buy a home on the beach and live off the interest on our **jmoney**.
Our correction	it really be a **great** opportunity to make relatively easy **money**, with little **cost** to you.	we have quit our **jobs**, and will soon buy a home on the beach and live off the interest on our **money**.
PyEnchant	it really be a **graft** opportunity to make relatively easy **fmoney**, with little **cost** to you.	we have quit our **bots**, and will soon buy a home on the beach and live off the interest on our **money**.
Ekphrasis	it really be a **great** opportunity to make relatively easy **money**, with little **cost** to you.	we have quit our **tobs**, and will soon buy a home on the beach and live off the interest on our **money**.
Google	it really be a **great** opportunity to make relatively easy **fmoney**, with little **cosgt** to you.	we have quit our **jobs**, and will soon buy a home on the beach and live off the interest on our **jmoney**.

From the same figure we see that our proposed error correction results in toxicity scores closer to the original ones when compared with the baselines, validating the effectiveness of our approach. We note that in some cases our approach might result in a slightly higher toxicity score than the original one, because of the pre-existing misspellings in the original sentences. For example, original sentences "you are a *fagget*", "*fukkin* goof's track record" and "I suspect that closedmouth is *g*y*" have crude misspellings, which are used as Internet slang. Our approach corrects these pre-existing misspellings in addition to the errors we add later, resulting in higher toxicity scores of corrected sentences. We perform corrections of all spelling errors, recovering more toxic words than we added maliciously.

5.2 Spam Detection

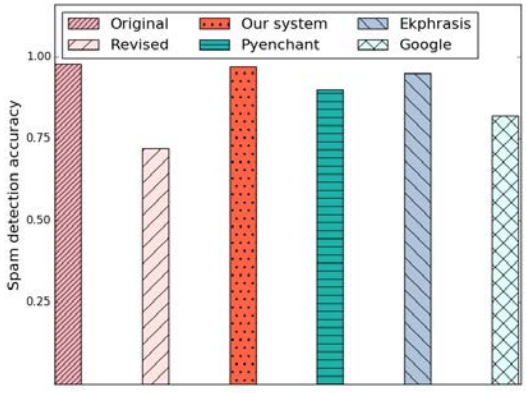


Figure 2: Spam detection accuracy on test emails.

We next evaluate our spelling correction on the spam data. Synthetic malicious errors were added to spam-indicative words in spam mails based on our misspelling generation mechanism.

Some example spams are shown in Table 4. Sensitive words such as "money" which are indicative of spams are highlighted, and the corrections given by different approaches are shown. Table 2 shows the spelling correction accuracy achieved by our algorithm and the baselines for the spam data.

Fig. 2 shows the spam detection accuracy on the original, revised and corrected test emails. Again we see that malicious errors resulted in a large accuracy drop, the accuracy was restored after spelling correction, and the increase in accuracy using our approach was the maximal among those compared.

5.3 Real Misspellings in Tweet Hate Speech Detection

We have shown that synthetic misspellings are able to deceive the toxicity detector and the spam detector, and reported the performance of spell checkers on the synthetic data. We also experimented with data containing real spelling errors collected from Twitter, where instances of hate speech contain user-generated spelling errors. The correction performance of spell checkers are now compared on the real misspellings.

Table 5: Example of Tweets and Hate Speech Categories

Category	Example sentence
Racism	#isis #islam pc puzzle: converting to a religion of peace leading to violence? http://t.co/tbjusaemuh
Sexism	Real question is do feminist liberal bigots understand that different rules for men/women is sexism
Neither	The mother and son team are sooooo nice !!!

Tweet normalization. Some example tweets of racist and sexist nature are shown in Table 5. Tweets are notoriously noisy and unstructured given the frequent occurrences of hashtags, URLs, reserved words and emojis. These non-standard tokens greatly increase the vocabulary size of tweets while also injecting noise to classification tasks. We use Tweet Preprocessor, a tweet preprocessing tool which can replace aforementioned tokens with a special set of tokens. For example, one tweet is "#isis #islam pc puzzle: converting to a religion of peace leading to violence ?,http://t.co/tbjusaemuh", which becomes "$HASHTAG$ $HASHTAG$ pc puzzle: converting to a religion of peace leading to violence? URL" after preprocessing. This preprocessing stage can clean texts without dealing with misspellings. Tweets are preprocessed right before they are fed into the neural network.

Hate speech detection. The state-of-the-art system for hate speech detection is a Bidirectional Long Short Term Memory (BLSTM) network with attention mechanism [3]. The model takes a sequence of words in a tweet, obtains word embeddings in the embedding layer, and passes them to bidirectional recurrent layers to generate

Table 6: Hate Speech Detection Results with Different Spelling Corrections

Category	Metric	Original	Our system	PyEnchant	Ekphrasis	Google
racist	Precision	0.630	**0.640**	0.630	0.630	0.569
	Recall	0.617	0.681	0.617	0.617	**0.702**
	F1 score	0.623	**0.660**	0.623	0.623	0.628
sexist	Precision	0.641	0.630	0.629	0.629	**0.649**
	Recall	0.775	0.767	**0.790**	0.790	0.759
	F1 score	0.701	0.692	**0.701**	**0.701**	0.700
Macro average over all categories	Precision	0.721	**0.721**	0.718	0.718	0.703
	Recall	0.741	0.757	0.743	0.743	**0.759**
	F1 score	0.727	**0.737**	0.727	0.727	0.727

a dense representation of the input tweet. The feedforward layer takes the tweet vector and predicts its probability distribution over all three classes. The class with the highest likelihood is chosen as the category of the input tweet. In our experiment, we use a BLSTM model for the hate speech detection task.

Vocabulary construction. Firstly we build a vocabulary list using both a standard dictionary and the frequent words (frequency higher than 5) in the training data. This list will serve as a reference for spelling correction; words outside the list will be taken as spelling errors and replaced with legal words from the vocabulary. The reason for collecting words from the training data is to include the Internet slangs in tweets which may not exist in the standard dictionary. Some examples are "tmr" (for *tomorrow*), "fwd" (for *forward*) and "lol" (for *laughing out loudly*). Some previous works proposed to replace these slang terms with their standard forms [13, 23], which require either expert knowledge or human annotations. We argue that because these Internet slangs change rapidly we should understand them in a data-driven manner instead of standardizing them based on human knowledge. Since the representation of these slangs and their use in hate speech will be learned by the neural network from the train data, we add these slangs to our vocabulary as legal words.

Spelling correction. Misspellings are another source of noise which cannot be handled in the tweet normalization stage. Some misspellings are user-created to deceive the online detection system. For example, the swear word "fucking" has a lot of variants in tweets such as "fckn", "f*ckin", "f**king", "fckin", and "fuckin". When a new variant of a swear word arises in the test data, the model takes it as an out-of-vocabulary word, and is unable to match it with any learned pattern. The purpose of spelling correction is to map these new variants to words that are known to the model. As discussed in Section 4, we enumerate the candidates of a misspelled word, and choose the candidate which best fits with the context as its correction.

Results. We do a random 80:20 train-test split of the Twitter dataset. A detection system was trained on the normalized train data (without spelling correction) using the state-of-the-art BLSTM model. There were 3218 test tweets, 571 of which had misspellings. We applied the different spelling correction approaches to these 571 tweets, and the corrected tweets were then cleaned as described in the tweet normalization stage. Processed tweets were input to the trained detection system. We compared the hate speech detection results on the 571 test tweets to evaluate the effect of spelling correction on this downstream application. As shown in Table 6, We report results on the original test data, and also on the test data which are corrected by our approach, PyEnchant, Ekphrasis, and google spell checker. We report precision, recall and F1-score of racist and sexist classification respectively and the macro-averages (to evaluate the overall performance).

The best performance of each metric is highlighted in the table. Compared with the classification performance on the test data with spelling errors for the racist category, our approach improves the precision by 1%, the recall by 6.4%, and the F1 score by 3.7% absolute points. Both PyEnchant and Ekphrasis spell checkers improve the recall of sexist category, but decrease the precision, so their corrected forms achieve F1 scores similar to the original test data. Google spell checker also gives similar F1 score on both racist and sexist classes. Our approach outperforms the other baselines in terms of F1 score for the racist class and the macro F1 score.

For sexism-related tweets, our approach does not improve the results compared to the original test data. Taking a closer look at these tweets we notice that they often contain some abbreviations which might be taken as misspellings, and that their contextual information is insufficient to decide the appropriate corrections. For example, in the sexist tweet "I'm sorry but if you watch women ufc fights kys". Our approach replaces *kys* with *keys*, and the trained neural network misclassified the corrected tweet. Another instance is "these nsw promo girls think way too highly of themselves", where *nsw* is incorrectly replaced with *new* by our approach.

6 CONCLUSION

In this study, we showed how malicious spelling errors can deceive profanity- and spam detectors. To deal with these malicious misspellings, we proposed a context-sensitive spelling corrector based on word embeddings. Our spell checker is light-weight, unsupervised and can be easily incorporated into downstream applications. It achieved a favorable spelling correction performance when compared with general purpose spell-checking tools such as PyEnchant, Ekphrasis and Google spell checkers on both synthetic and real misspellings from different datasets.

ACKNOWLEDGMENTS

This work is supported in part by the IBM-ILLINOIS Center for Cognitive Computing Systems Research (C3SR) - a research collaboration as part of the IBM AI Horizons Network.

REFERENCES

[1] 2018. AbiWord. Available at: https://www.abisource.com.

[2] 2018. Google Search Engine. Available at: https://www.google.com.

[3] Sweta Agrawal and Amit Awekar. 2018. Deep Learning for Detecting Cyberbullying Across Multiple Social Media Platforms. In *European Conference on Information Retrieval*. Springer, 141–153.

[4] Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed Word Representations for Multilingual NLP. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Sofia, Bulgaria, 183–192. http://www.aclweb.org/anthology/W13-3520

[5] Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, 747–754.

[6] Noah Coad. 2018. Google Spell Check. Available at: https://github.com/noahcoad/google-spell-check.

[7] Fred J Damerau. 1964. A technique for computer detection and correction of spelling errors. *Commun. ACM* 7, 3 (1964), 171–176.

[8] Noura Farra, Nadi Tomeh, Alla Rozovskaya, and Nizar Habash. 2014. Generalized Character-Level Spelling Error Correction.. In *ACL (2)*. 161–167.

[9] Shaona Ghosh and Per Ola Kristensson. 2017. Neural Networks for Text Correction and Completion in Keyboard Decoding. *arXiv preprint arXiv:1709.06429* (2017).

[10] Hongyu Gong, Suma Bhat, and Pramod Viswanath. 2017. Geometry of Compositionality.

[11] Jigsaw Google. 2017. Perspective. Available at: https://www.perspectiveapi.com/.

[12] Sergey Gubanov, Irina Galinskaya, and Alexey Baytin. 2014. Improved Iterative Correction for Distant Spelling Errors.. In *ACL (2)*. 168–173.

[13] Itisha Gupta and Nisheeth Joshi. 2017. Tweet normalization: A knowledge based approach. In *Infocom Technologies and Unmanned Systems (Trends and Future Directions)(ICTUS), 2017 International Conference on*. IEEE, 157–162.

[14] Hany Hassan and Arul Menezes. 2013. Social Text Normalization using Contextual Graph Random Walks.. In *ACL (1)*. 1577–1586.

[15] Hossein Hosseini, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. 2017. Deceiving Google's Perspective API Built for Detecting Toxic Comments. *arXiv preprint arXiv:1702.08138* (2017).

[16] Fraser Howard. 2008. Web Attacks: Modern web attacks. *Network Security* 2008, 4 (2008), 13–15.

[17] Dan Jurafsky and James H Martin. 2014. *Speech and language processing*. Vol. 3. Pearson London.

[18] Daniel Jurafsky and James H. Martin. 2017. Distributed Representations of Words and Phrases and their Compositionality. In *Speech and Language Processing*. Chapter 5, 1–12.

[19] Ryan Kelly. 2015. PyEnchant. Available at: https://github.com/rfk/pyenchant.

[20] Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, Vol. 10. 707–710.

[21] Fei Liu, Fuliang Weng, Bingqing Wang, and Yang Liu. 2011. Insertion, deletion, or substitution?: normalizing text messages without pre-categorization nor supervision. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, 71–76.

[22] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3111–3119.

[23] Abiodun Modupe, Turgay Celik, Vukosi Marivate, and Melvin Diale. 2017. Semi-supervised probabilistics approach for normalising informal short text messages. In *Information Communication Technology and Society (ICTAS), Conference on*. IEEE, 1–8.

[24] Howard L. Morgan. 1970. Spelling correction in systems programs. In *Communications of the ACM*. 90–94.

[25] Thien Huu Nguyen and Ralph Grishman. 2014. Employing Word Representations and Regularization for Domain Adaptation of Relation Extraction.. In *ACL (2)*. 68–74.

[26] Etienne Papegnies, Vincent Labatut, Richard Dufour, and Georges Linares. 2017. Impact Of Content Features For Automatic Online Abuse Detection. In *International Conference on Computational Linguistics and Intelligent Text Processing*.

[27] Aurelia Power, Anthony Keane, Brian Nolan, and Brian O'Neill. 2018. Detecting Discourse-Independent Negated Forms of Public Textual Cyberbullying. *Journal of Computer-Assisted Linguistic Research* 2, 1 (2018), 1–20.

[28] Luz Rello, Miguel Ballesteros, and Jeffrey P Bigham. 2015. A spellchecker for dyslexia. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility*. ACM, 39–47.

[29] Eric Sven Ristad and Peter N Yianilos. 1998. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20, 5 (1998), 522–532.

[30] Sergio A Rojas-Galeano. 2013. Revealing non-alphabetical guises of spam-trigger vocables. *Dyna* 80, 182 (2013), 15–24.

[31] Bahar Salehi, Paul Cook, and Timothy Baldwin. 2015. A Word Embedding Approach to Predicting the Compositionality of Multiword Expressions.. In *HLT-NAACL*. 977–983.

[32] Manish Saxena and PM Khan. 2015. Spamizer: An approach to handle web form Spam. In *Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on*. IEEE, 1095–1100.

[33] Kaveh Taghipour and Hwee Tou Ng. 2015. Semi-supervised word sense disambiguation using word embeddings in general and specific domains. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 314–323.

[34] Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*. 88–93.

[35] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. 2009. Feature hashing for large scale multitask learning. In *Proceedings of the 26th annual international conference on machine learning*. ACM, 1113–1120.

[36] Zar Zar Wint, Theo Ducros, and Masayoshi Aritsugi. 2017. Spell corrector to social media datasets in message filtering systems. In *Digital Information Management (ICDIM), 2017 Twelfth International Conference on*. IEEE, 209–215.

[37] Zar Zar Wint, Théo Ducros, and Masayoshi Aritsugi. 2018. Non-words Spell Corrector of Social Media Data in Message Filtering Systems. *Journal of Digital Information Management* 16, 2 (2018).

[38] Xinwang Zhong. 2014. Deobfuscation based on edit distance algorithm for spam filitering. In *Machine Learning and Cybernetics (ICMLC), 2014 International Conference on*, Vol. 1. IEEE, 109–114.

With a Little Help from My Friends (and Their Friends): Influence Neighborhoods for Social Recommendations

Avni Gulati
Computer Engineering Department
San Jose State University
San Jose, CA, USA
avni.gulati@sjsu.edu

Magdalini Eirinaki
Computer Engineering Department
San Jose State University
San Jose, CA, USA
magdalini.eirinaki@sjsu.edu

ABSTRACT

Social recommendations have been a very intriguing domain for researchers in the past decade. The main premise is that the social network of a user can be leveraged to enhance the rating-based recommendation process. This has been achieved in various ways, and under different assumptions about the network characteristics, structure, and availability of other information (such as trust, content, etc.) In this work, we create neighborhoods of influence leveraging only the social graph structure. These are in turn introduced in the recommendation process both as a pre-processing step and as a social regularization factor of the matrix factorization algorithm. Our experimental evaluation using real-life datasets demonstrates the effectiveness of the proposed technique.

CCS CONCEPTS

• **Information systems** → **Social recommendation**; **Social networks**; • **Human-centered computing** → **Collaborative filtering**; **Social recommendation**; • **Theory of computation** → Social networks.

KEYWORDS

social recommender systems, influence propagation, recommendation systems

ACM Reference Format:
Avni Gulati and Magdalini Eirinaki. 2019. With a Little Help from My Friends (and Their Friends): Influence Neighborhoods for Social Recommendations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313745

1 INTRODUCTION

Humans form opinions and make multiple decisions, small or big, on any given day. For many of those, such as deciding which movie to watch, or which new restaurant to try, or even which graduate school to attend, we tend to rely not only on our own personal judgment, knowledge and intution, but also that of others, especially those whose opinion we value and trust. This type of influence is what social recommender systems attempt to simulate. Social recommender systems leverage social relations to improve the rating-based recommendation process [1], based on the assumption that a user's preferences are likely to be similar to, or influenced by these of their friends [11].

Most of existing related work is making the assumption that the users are mostly influenced by their direct neighbors. However, studies have shown that humans tend to form opinions not only through their direct connections, but also via acquaintances [8, 9]. This concept aligns with the study of influence propagation in social networks. This line of work is drawing inspiration from social correlation theories such as homophily and social influence [18, 19]. Most of the existing approaches regard this as a long-term, network diffusion process, and follow a graph-theoretic approach to solve the problem of identification of influentials in a social network [2, 6, 13].

In this work, we introduce a social recommendation algorithm that leverages the influence propagation beyond direct neighbors in a social graph. In particular, we explore the integration of the social graph as input to the recommendation process. We employ our threshold-bounded influence propagation algorithm [10] to generate social graph-based neighborhoods for each user. This neighborhood of influence is used as a pre-processing step. The outcome of this algorithm is also integrated as a social regularization factor in the matrix factorization process. Our experimental results using real-life datasets demonstrate the effectiveness of such an approach. The proposed high-level methodology is depicted in Figure 1.

The rest of the paper is organized as follows: we review the related work in Section 2. We then present the influence propagation algorithm in Section 3. The proposed recommendation algorithm with social regularization is presented in Section 4. Finally, in Section 5 we discuss the results of our experimental evaluation and conclude with our observations and our plans for future work in Section 6.

2 RELATED WORK

Social recommender systems have gained a lot of attention from the research in an effort to leverage social relationships to improve the recommendation process. Rooted in the sociology concepts of homophily and social influence [18], this line of work is based on the assumption that users' preferences are influenced more by these of their connected friends than these of unknown users [26]. Tang et al. [22] give a narrow definition of social recommendation as "any recommendation with online social relations as an additional

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313745

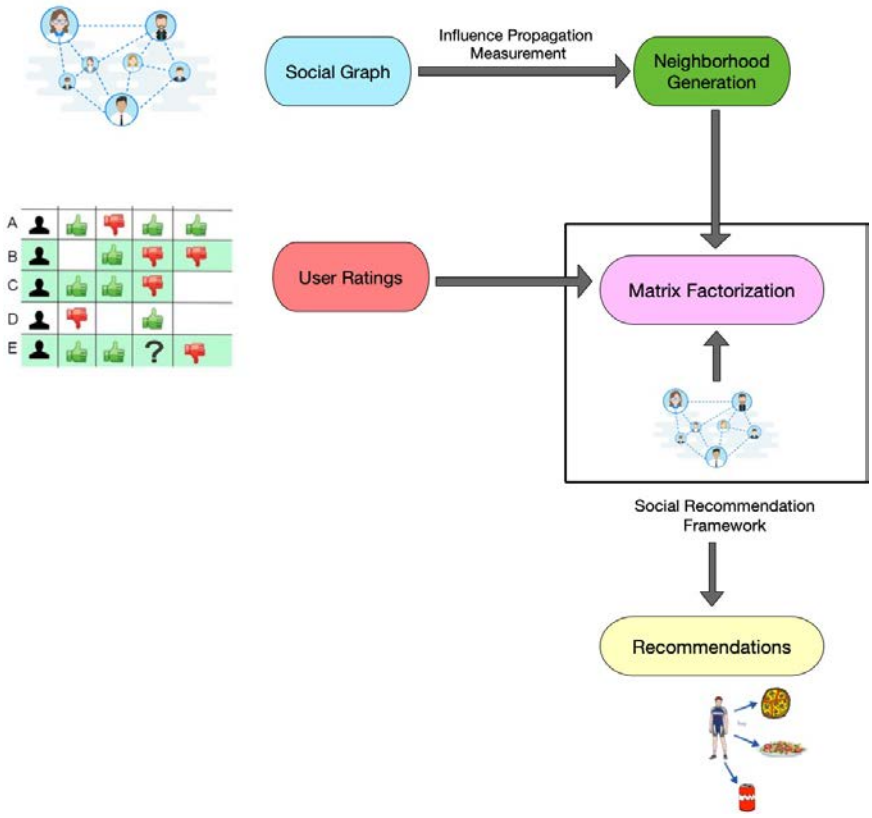


Figure 1: Social Recommendation Framework incorporating influence propagation approach for neighborhood generation in Matrix Factorization for generating effective recommendations for users.

input, i.e., augmenting an existing recommendation engine with additional social signals" (a broader definition, not applicable to this work, refers to recommender systems targeting social media domains [11]).

The various proposed approaches can be categorized depending on the type of social relationship (trust, friendship etc.), the type of the underlying recommendation algorithm (model-based, memory-based, etc.), and the level of integration of the social information in the recommendation process. A common approach is to enhance model-based recommender systems with social connections, again most often expressed as trust. This can be done through co-factorization, where the assumption is that the users share the same preference vector in both the rating and the social spaces (e.g. [20]), ensemble methods, where the resulting recommendation is derived by the linear combination of two systems (e.g. [15, 21]), or regularization, where priority is given to the social-based ratings (e.g. [12, 16]).

An alternative line of work involves ways to enhance the memory-based collaborative filtering process by forming the user's neighborhood using similarities deriving from the users' ratings and/or their social relationships, focusing on trust [5, 7, 17, 23–25]. One of the most recent works [4] introduces a Deep Learning Matrix Factorization (DLMF) approach to address the issues related to the initialization of latent feature vectors in Matrix Factorization (MF). The authors use the user's trust on other users who belong to his/her clique as the basis of neighborhood formation, and integrate it with Matrix Factorization as a trust regularization factor to predict ratings.

Ma et al. [16] introduced social regularization constraints in the recommendation framework. They used the social information to effectively predict the missing user-item matrix. Experimental results on real world data showed that their algorithm outperformed the traditional Matrix Factorization methodology. In this work we follow a similar approach, however we focus on influence as derived from the social graph connections, rather than metadata related to the users. We leverage our proposed influence propagation algorithm [10] and create social graph-based personalized neighborhoods that are subsequently used as input to the recommendation process. Moreover, we combine the social information of the users gained from influence propagation algorithm to Matrix Factorization forming a unified social recommendation framework.

3 NEIGHBORHOOD FORMATION USING INFLUENCE PROPAGATION

In this section we briefly present our Threshold-Bounded Influence Propagation algorithm (TB-IP) [10]. While this algorithm was introduced to solve the *min-max* problem of maximizing influence coverage on a social network with the minimum number of "seed" nodes, in this work we utilize it as a means to integrate social graph data to the recommendation process[1].

A social network can be modeled as a weighted-edge graph $G = (V, E, W)$, where V is the set of all the vertices in the graph (i.e. people), E is the set of edges (i.e. their connections), and W is the set of edge weights. An edge from node u to node v ($u \rightarrow v$) signifies that user u "follows" v in the social network, and the edge weight $w(u_v)$ represents the influence of v on u. Our objective is to identify the influential nodes $v \in M, M \subset V$ that influence the remaining nodes $u \in D, D \subset V$, $D \cap M = \emptyset$ such that $|M|$ is minimized and $|D|$ is maximized. The *Threshold-Bounded Influence Propagation in Digraph (TB-IP)* algorithm takes as input a graph G and outputs sets M and D.

The algorithm accepts the following as parameters: a) a threshold thr that is defined according to the selected threshold condition and is used to determine whether a node u is influenced by v, b) a maximum number of *"hops"* (i.e. the maximum allowed depth of influence propagation), c) a *decay factor* for the influence propagation, and d) a ranking strategy for initializing the nodes (e.g. PageRank, Out-degree centrality, Upper-bound PageRank etc.). We consider three alternatives as threshold conditions:

- *Condition I: No Threshold (NoThr):* The first condition is considered for two-hops in the graph considering no threshold for influence propagation.
- *Condition II: Average Threshold (AvgThr):* The second condition takes threshold as the average of edge-weights of the entire network. This would mean that the threshold is constant for all the nodes, depending on the characteristics of the network as a whole.
- *Condition III: Edge-Weight dependent Threshold (EWThr):* The third condition determines the threshold by taking average of edge-weights of all the outgoing-edges from the node in the graphically represented social network. Thus, this threshold condition is vertex-specific but constant for every node.

The algorithm begins by sorting all nodes in descending order based on their assigned rank $r(v)$. Then, beginning with the highest ranked node, it examines whether its direct connections should be added to its neighborhood and therefore be considered influenced or not. This is determined by examining whether the edge-weight $w(u_v)$ of a connected node u is above the set threshold thr or not. If at least one connected node qualifies, node v is being added to the "influencer" set M and the qualifying nodes are being added to the "influenced" set D. If the algorithm is set to examine nodes that are indirectly connected to v (depth is defined by the *maxhop* parameter), each of the nodes that were added to $N_G(v)$ in the previous step are used to find their directly connected nodes. However, in this case, the respective edge-weights are updated by the *hopping factor* before being evaluated against the threshold thr as follows:

$$w(p_u) * (1 + hop * decay) \geq threshold \quad (1)$$

where *hop* is an integer value 0 for immediate neighbors (i.e. adjacent nodes) and it increments by 1 for each subsequent hop, and *decay* is a constant equal to 0.1[2].

When a node satisfies the threshold condition and has not been previously added to the "influenced" set D, then it is being added to both this set, and the neighborhood of v, $N_G(v)$. The algorithm stops this loop when either the maximum depth (i.e. number of hops) has been reached, or no nodes are qualifying as "influenced" in the current level. This process is being repeated for each of the nodes, as selected from the ranked list, and as long as they have not already been added in the "influenced" set D. The above process is described in detail in Algorithm 1.

Algorithm 1 Threshold-Bounded Influence Propagation in Digraph (TB-IP)

Require: A weighted and directed social network $G = (V, E, W)$
Ensure: Influenced Vertices $D \subset V$ and Influential Vertices $M \subset V$a

```
Initialize: thr, maxhop, visit = 0, hop = 0, decay = 0.1, D = ∅, M = ∅
∀v ∈ V, r(v) = compute_rank(v)
Add all (v_i, v_j) ∈ V to ordered set S s.t. if r(i) > r(j) then i < j
for each v ∈ S do
    if w(u_v) > thr and ∃ u ∈ V \ D then
        N_G(v) = N_G(v) ∪ u
        D = D ∪ u
        M = M ∪ v
        visit = 1
    end if
    while hop ≤ maxhop and visit = 1 do
        hop = hop + 1
        visit = 0
        for each u ∈ N_G(v) do
            w(p_u) = w(p_u) * (1 + hop * decay)
            if w(p_u) > thr and ∃ p ∈ V \ D then
                N_G(v) = N_G(v) ∪ p
                D = D ∪ p
                visit = 1
            end if
        end for
    end while
end for
output D, M
```

4 SOCIAL REGULARIZATION IN MATRIX FACTORIZATION

A lot of techniques have been used in the past to generate effective recommendations for users. Perhaps the most popular methodology in terms of both effectiveness and efficiency is matrix factorization [14]. This collaborative filtering technique takes as input the

[1] More details as well as the experimental evaluation of this algorithm can be found in [10].

[2] Value set after experimentation.

matrix containing user ratings for items and applies dimensionality reduction to identify latent features of correlated user-item interactions. Mathematically, the sparse user-item (or *utility*) matrix ($R \in \mathbb{R}^{m \times n}$) can be decomposed into two matrices, representing the users ($P \in \mathbb{R}^{k \times m}$) and items ($Q \in \mathbb{R}^{k \times n}$). The low-ranked matrix factorization approach approximates the utility matrix R by multiplication of k-rank factors (where $k < min(m, n)$):

$$\mathrm{R} \approx \mathrm{P}^{\mathsf{T}} \times \mathrm{Q} \tag{2}$$

Singular Value Decomposition (SVD) is traditionally used to factorize the observed ratings from the utility matrix to obtain the latent factors by minimizing the following objective function:

$$\min_{P,Q} \frac{1}{2} \sum_{u=1}^{m} \sum_{i=1}^{n} I_{ui}(R_{ui} - P_u^T \cdot Q_i)^2 + \lambda_1(\|P\|^2) + \lambda_2(\|Q\|^2), \tag{3}$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are the regularization factors added to avoid overfitting. I_{ui} is an indicator function which is 1 when user u has rated item i and it is equal to 0 if the user has not rated the item. Several optimization approaches can be used to find local minimum of Equation 3, such as Alternating Least Squares (ALS) or Stochastic Gradient Descent (SGD).

In this work, we integrate the social graph data in the recommendation process following a similar approach to that of Ma et al. [16], who introduced a social regularization term in the objective function, incorporating the preferences of a user's friends, i.e. the directly connected nodes of a node in the social graph. We employ the TB-IP algorithm discussed in section 3 in two ways, as shown in Figure 1.

First, we apply the algorithm as a pre-processing step to define a social graph-based neighborhood of influence in the graph. This process removes some of the nodes that do not belong to the circle of influence of any other users, and thus are expected to contribute less to the rating prediction. Secondly and most importantly, we define a new social similarity function that integrates the edge-weights derived from the TB-IP algorithm as social relation weights between users.

The problem of estimating the utility matrix R is therefore reduced to minimizing the following objective function and regularization terms:

$$\begin{aligned} \min_{P,Q} \mathcal{L}_2(R, P, Q) = \frac{1}{2} \sum_{u=1}^{m} \sum_{i=1}^{n} I_{ui}(R_{ui} - P_u^T \cdot Q_i) + \\ \frac{\beta}{2} \sum_{v \in N_G^{+(u)}} SocSim(u, v)(\|P_u - P_v\|^2) + \\ \lambda_1(\|P\|^2) + \lambda_2(\|Q\|^2), \end{aligned} \tag{4}$$

where $\beta > 0$, and ${N_G}^{+(u)}$ denotes u's out-link connections belonging to its neighborhood N_G^u as derived by the TB-IP algorithm.

We further augment this social regularization term by introducing a social similarity to weigh how much each neighbor's preference affects the user's predictions. We define the social similarity $SocSim(u, v)$ between two users u and v as follows:

$$SocSim\,(u, v) = \frac{1}{2}(Sim\,(u, v) + w(u_v)). \tag{5}$$

$Sim(u, v)$ stands for any vector similarity metric that can be employed to calculate how similar the two user vectors u and v are (in terms of similar ratings). In this work, we use the Pearson correlation coefficient [3] formulated as:

$$Pearson\,(u, v) = \frac{\sum_{i \in P_u \bigcap P_v} (R_{ui} - \bar{R}_u)(R_{vi} - \bar{R}_v)}{\sqrt{\sum_{i \in P_u \bigcap P_v} (R_{ui} - \bar{R}_u)^2} \sqrt{\sum_{i \in P_u \bigcap P_v} (R_{vi} - \bar{R}_v)^2}}. \tag{6}$$

Here, i is the subset of total items user u and user v have both rated. R_{ui} is the rating given by user u to item i. $\bar{R}_u$ and $\bar{R}_v$ are the average ratings of users u or v respectively and R_{ui}, R_{vi} are the ratings of users u and v respectively for item i.

The second term, $w(u_v)$, is the edge weight derived by Algorithm 1. This weight measures the extend to which user v influences user u, and is based on the social graph's structure. Therefore, this social similarity metric leverages both the social graph and the rating preferences of the users.

A local minimum of Equation 4 can be obtained by applying SGD on the P_u and Q_i feature vectors, solving the partial derivatives on both the feature vectors as shown in Equations 7 and 8:

$$\begin{aligned} \frac{\partial \mathcal{L}_2}{\partial P_u} = \sum_{i=1}^{n} I_{ui}(P_u^T \cdot Q_i - R_{ui})Q_i + \lambda_1 P_u \\ +\beta \sum_{v \in N_G^{+(u)}} SocSim(u, v)(P_u - P_v) \end{aligned} \tag{7}$$

$$\frac{\partial \mathcal{L}_2}{\partial Q_i} = \sum_{u=1}^{m} I_{ui}(P_u^T \cdot Q_i - R_{ui})P_u + \lambda_2 Q_i \tag{8}$$

5 EXPERIMENTAL EVALUATION

The need for datasets that combine both user-item-rating triplets and social graph data for the users limits our choices to a big extend. One of the most known and broadly used datasets that combines both sources of information is the Yelp Challenge dataset[3]. We also employ the Epinions dataset[4]. Epinions is a trust-based rating dataset - in essence users have denoted whether they trust other users in their reviews and ratings or not. For our work we have translated "trust" endorsements to directed edges. While the resulting graph is not representing social relationships (of friendship), it encapsulates some social characteristics (users trusting other users in some context), and in the absence of other publicly available datasets, we follow the example of related research approaches and use this as a second real-life data source.

For the Yelp dataset we select the Las Vegas subset, including only the users who had rated at least one business in the city of Las Vegas. This is because the Yelp social subset (i.e. friends's network) is very sparse, with the Las Vegas dataset containing the most dense social graph subset. Hence, we only considered those connections who rate restaurants and reside in the same area. The details of the social graph of each dataset are included in Table 1, while the user-item-rating characteristics are included in the first row of Tables 2 and 3.

[3] https://www.yelp.com/dataset/challenge
[4] www.epinions.com

For each dataset, we generate recommendations using two baselines and several variations of our proposed methodology. We employ the optimal parameters for the TB-IP algorithm, as experimentally identified in [10], running the algorithm using the *EWThr* threshold selection strategy for two *hops* (named *TwoD EWThr*).

Table 1: Social graph dataset characteristics

Dataset	Yelp (Las Vegas)	Epinions (Trust)
# Nodes	247,111	18,089
# Edges	5,340,568	355,217
Avg. Degree	21.612	19.6372
Type	Directed	Directed

Table 2: Yelp input datasets

Dataset	# Users	# Businesses	# Ratings
BaseAll	247,111	26,304	1,104,768
TwoD EWThr	96,900	24,632	733,408

Table 3: Epinions input datasets

Dataset	# Users	# Items	# Ratings
BaseAll	22,164	296,277	922,267
TwoD EWThr	7,615	222,281	539,963

We evaluate the effect of the neighborhood formation in two levels: i) as the input (i.e. as a pre-processing step) to the recommendation process, and ii) as an integral part of the recommendation algorithm.

To evaluate the first objective, we apply TB-IP to the entire dataset. We therefore create two different input sets, as shown in Tables 2 and 3:

Dataset 1: BaseAll. This dataset has the entire user-item-rating data.

Dataset 2: TwoD EWThr. This dataset has the rating data only for users obtained after applying the TB-IP Algorithm with *EWThr* threshold condition. We use the *Outdeg* ranking methodology for initial ranking of nodes.

We consider two baseline methodologies to evaluate our proposed Social Recommendation Framework. The first baseline evaluates the traditional Matrix Factorization (*SimpleMF*) explained in equation 3. As the second baseline we consider the Social Regularization in Matrix Factorization (*Simple SoReg*) developed by Ma et al. [16]. In that work, the similarity function does not incorporate the social graph weight (in essence reflects a normalized Pearson correlation coefficient). We compare these baselines against several variations of our method, employing the TB-IP algorithm and the new similarity metric function (*SocSim*) that we defined in Equation 5. The different setups are explained in what follows.

Experiment 1: SimpleMF (First Baseline). Using as input the entire dataset (*BaseAll*) and implemented *Basic MF*, i.e. traditional Matrix Factorization [14].

Experiment 2: TB-IP SimpleMF. In this experiment, we considered the social-graph induced dataset (*TwoD EWThr*) and generated recommendations using *Basic MF*.

Experiment 3: Simple SoReg (Second Baseline). Here, we evaluated *SoReg* methodology with *Sim* for *BaseAll* dataset, as discussed in [16].

Experiment 4: TB-IP SoReg. We use as input the social-graph induced dataset *TwoD EWThr* and generate recommendations using *SoReg* technique using the normalized Pearson similarity metric.

Experiment 5: SocSim Simple SoReg. Here, we apply our proposed *SoReg* algorithm incorporating the *SocSim* similarity metric over the entire (*BaseAll*) dataset.

Experiment 6: SocSim TB-IP SoReg. In this experiment, the preprocessing step is applied to create the input dataset *TwoD EWThr* and the recommendations are generated by using the *SoReg* technique with *SocSim* similarity metric.

We performed 5-fold cross validation for all the experimental setups.[5] We evaluated all combinations of the following Matrix Factorization parameters: *rank (f)* = $\{10, 20\}$, *regularization* (λ) = $\{0.1, 0.01, 0.001\}$, *maximum iterations (max_iter)* = $\{10, 20\}$. For the Social Regularization technique we also used another metric, called *social regularization* (β) = $\{0.1, 0.01\}$.

We use RMSE and MAE evaluation metrics. The results, along with the optimal parameters for each setup, are presented with the corresponding experiment in Tables 4 and 6 (RMSE) and Tables 5 and 7 (MAE) for the Epinions and Yelp dataset respectively.

Table 4: RMSE and optimal parameter settings for different experiments performed for Epinions

Experiment	RMSE	f	λ	β	max iter
SimpleMF	1.063	20	0.001	0	10
TB-IP SimpleMF	1.046	20	0.001	0	10
Simple SoReg	1.046	20	0.01	0.1	20
TB-IP SoReg	**1.030**	20	0.01	0.1	20
SocSim Simple SoReg	1.061	20	0.01	0.1	10
SocSim TB-IP SoReg	1.043	20	0.01	0.1	10

Table 5: MAE and optimal parameter settings for different experiments performed for Epinions

Experiment	MAE	f	λ	β	max iter
SimpleMF	0.812	20	0.001	0	10
TB-IP SimpleMF	0.800	20	0.001	0	10
Simple SoReg	0.807	20	0.01	0.01	10
TB-IP SoReg	**0.794**	20	0.01	0.01	20
SocSim Simple SoReg	0.813	20	0.01	0.1	10
SocSim TB-IP SoReg	0.799	20	0.01	0.01	10

We observe that the *TB-IP* prefixed approaches, i.e. those that include the pre-processing step of identifying the

[5]We used ReQ python library from https://github.com/Coder-Yu/RecQ to perform our experiments on the baselines and updated the code to reflect our methodology.

Table 6: RMSE and optimal parameter settings for different experiments performed for Yelp

Experiment	RMSE	f	λ	β	max iter
SimpleMF	1.252	20	0.001	0	10
TB-IP SimpleMF	1.175	20	0.01	0	10
Simple SoReg	1.222	20	0.01	0.01	10
TB-IP SoReg	1.152	20	0.01	0.01	10
SocSim Simple SoReg	1.212	20	0.01	0.1	10
SocSim TB-IP SoReg	**1.145**	20	0.01	0.1	10

Table 7: MAE and optimal parameter settings for different experiments performed for Yelp

Experiment	MAE	f	λ	β	max iter
SimpleMF	0.970	20	0.001	0	10
TB-IP SimpleMF	0.909	20	0.01	0	10
Simple SoReg	0.951	20	0.01	0.01	10
TB-IP SoReg	**0.895**	20	0.01	0.01	10
SocSim Simple SoReg	0.953	20	0.01	0.1	10
SocSim TB-IP SoReg	0.899	20	0.01	0.01	10

influential neighborhoods within the original dataset, outperform the others in both datasets. For the Epinions dataset we observe that the *TB-IP SoReg* approach outperforms the others, giving the best results both in terms of RMSE and MAE, with second best our *SocSim TB-IP SoReg* approach. On the other hand, we observe that for the much larger Yelp dataset, our approach (*SocSim TB-IP SoReg*) gives better results than the other approaches in terms of RMSE while having very comparable MAE results with the *TB-IP SoReg*. In fact, we observe a big improvement over the baselines, ranging from 6.3% to 8.6% for RMSE and 5.8% to 7.7% for MAE.

As previously mentioned, the Yelp social graph represents a real social network (in that the edges reflect friendships between users), compared to the one of Epinions that signifies trust relationships ("I trust someone's opinions/reviews") but not necessarily real-life friendships or social relationships. Therefore we can conclude that the difference in our algorithm's effectiveness lies in the fact that it works better when real-life relationships are represented in the social graph and therefore better reflected in the social weights $w(u_v)$ and the derived *SocSim* similarity.

Tables 8 and 9 show the running time for neighborhood formation (where applicable) and recommendation for the Epinions dataset. The times reported reflect the optimal parameters for each setup. As expected, the proposed approaches introduce an overhead for the neighborhood formation phase, resulting in higher total times. However, this pre-processing step is performed offline and is not expected to affect real-time recommendations. On the contrary, we observe that both approaches integrating our social regularization factor with the *SocSim* similarity are faster than their simple Pearson similarity counterparts during recommendation time. This is due to the fact that the algorithm took less iterations to converge in the former case. We hypothesize that this is due to the fact that the social graph weights $w(u_v)$ introduced in the user similarity (*SocSim*) act as a form of better initialization for the matrix factorization process, helping the algorithm to converge in fewer steps.

Overall our results confirm our claim that combining neighborhood formation with influence propagation and social regularization produces better recommendations than the traditional social recommendation approaches.

Table 8: Epinions time analysis for least RMSE (in min)

Experiment	Neigh	Rec	Total
SimpleMF	0	9	9
TB-IP SimpleMF	26	4	30
Simple SoReg	0	53	53
TB-IP SoReg	26	19	45
SocSim Simple SoReg	0	10	10
SocSim TB-IP SoReg	26	13	39

Table 9: Epinions time analysis for least MAE (in min)

Experiment	Neigh	Rec	Total
SimpleMF	0	9	9
TB-IP SimpleMF	26	4	30
Simple SoReg	0	28	28
TB-IP SoReg	26	19	45
SocSim Simple SoReg	0	10	10
SocSim TB-IP SoReg	26	7	33

6 CONCLUSIONS

In this work, the problem of generating social recommendations is explored in the context of influence propagation. We employ an influence propagation algorithm to create social graph-based neighborhoods of influence. This neighborhood is then used as input to the recommendation algorithm, both as a pre-processing step and as an integral part of the recommendation algorithm in the form of a similarity metric. Our experiments show that the integration of social graph-derived information improves the traditional rating-based recommendation process. Moreover, we observe that in the case of real-life relationships reflected in the input social graph, our proposed similarity metric greatly improves the results and accelerates the recommendation algorithm.

Part of our ongoing work includes improving further on the social regularization component and social similarity metric. We are also exploring employing deep learning to initialize the latent vectors, and want to study how other influence propagation algorithms and centrality metrics can be employed as a means to create user influence neighborhoods. Finally, we believe that the introduction of a social component in the recommendation process not only improves the accuracy of recommendations, but can also be leveraged to generate explainable recommendations. This is an interesting extension of this work that we plan to explore in the near future.

REFERENCES

[1] C.C. Aggarwal. 2016. *Recommender Systems: The Textbook* (1st ed.). Springer Publishing Company, Incorporated.

[2] Christian Borgs, Michael Brautbar, Jennifer T. Chayes, and Brendan Lucier. 2014. Maximizing Social Influence in Nearly Optimal Time. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*. 946–957.

[3] John S Breese, David Heckerman, and Carl Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 43–52.

[4] ShuiGuang Deng, Longtao Huang, Guandong Xu, Xindong Wu, and Zhaohui Wu. 2017. On Deep Learning for Trust-Aware Recommendations in Social Networks. *IEEE Trans. Neural Netw. Learning Syst.* 28, 5 (2017), 1164–1177.

[5] Magdalini Eirinaki, Malamati D. Louta, and Iraklis Varlamis. 2014. A Trust-Aware System for Personalized User Recommendations in Social Networks. *IEEE Trans. Systems, Man, and Cybernetics: Systems* 44, 4 (2014), 409–421.

[6] Magdalini Eirinaki, Nuno Moniz, and Katerina Potika. 2016. Threshold-Bounded Influence Dominating Sets for Recommendations in Social Networks. In *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom), BDCloud-SocialCom-SustainCom 2016, Atlanta, GA, USA, October 8-10, 2016*. 408–415.

[7] Jennifer Golbeck. 2006. Trust on the world wide web: a survey. *Found. Trends Web Sci.* 1, 2 (2006), 131–197.

[8] Mark Granovetter. 2018. *Getting a job: A study of contacts and careers*. University of Chicago press.

[9] Mark S Granovetter. 1977. The strength of weak ties. In *Social networks*. Elsevier, 347–367.

[10] Avni Gulati and Magdalini Eirinaki. 2018. Influence Propagation for Social Graph-based Recommendations. In *IEEE International Conference on Big Data, Big Data 2018, Seattle, WA, USA, December 10-13, 2018*. 2180–2189.

[11] Ido Guy. 2015. *Social Recommender Systems*. 511–543.

[12] Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*. 135–142.

[13] David Kempe, Jon Kleinberg, and Éva Tardos. 2015. Maximizing the Spread of Influence through a Social Network. *Theory of Computing* 11, 4 (2015), 105–147.

[14] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37.

[15] Hao Ma, Irwin King, and Michael R. Lyu. 2009. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*. 203–210.

[16] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011*. 287–296.

[17] Paolo Massa and Paolo Avesani. 2007. Trust-aware recommender systems. In *Proceedings of the 2007 ACM Conference on Recommender Systems, RecSys 2007, Minneapolis, MN, USA, October 19-20, 2007*. 17–24.

[18] James M. Cook Miller McPherson, Lynn Smith-Lovin. 2001. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology* 27 (2001), 415–444.

[19] Cosma Rohilla Shalizi and Andrew C. Thomas. 2011. Homophily and Contagion Are Generically Confounded in Observational Social Network Studies. *Sociological Methods Research* 40, 2 (5 2011), 211–239.

[20] Jiliang Tang, Huiji Gao, Xia Hu, and Huan Liu. 2013. Exploiting homophily effect for trust prediction. In *Sixth ACM International Conference on Web Search and Data Mining, WSDM 2013, Rome, Italy, February 4-8, 2013*. 53–62.

[21] Jiliang Tang, Huiji Gao, and Huan Liu. 2012. mTrust: discerning multi-faceted trust in a connected world. In *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012*. 93–102.

[22] Jiliang Tang, Xia Hu, and Huan Liu. 2013. Social recommendation: a review. *Social Netw. Analys. Mining* 3, 4 (2013), 1113–1133.

[23] Iraklis Varlamis, Magdalini Eirinaki, and Malamati D. Louta. 2010. A Study on Social Network Metrics and Their Application in Trust Networks. In *International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2010, Odense, Denmark, August 9-11, 2010*. 168–175.

[24] Iraklis Varlamis, Magdalini Eirinaki, and Malamati D. Louta. 2013. Application of Social Network Metrics to a Trust-Aware Collaborative Model for Generating Personalized User Recommendations. In *The Influence of Technology on Social Network Analysis and Mining*. Springer Vienna, 49–74.

[25] Patricia Victor, Martine De Cock, and Chris Cornelis. 2010. Trust and Recommendations. In *Handbook of Recommender Systems*. springer, 645–675.

[26] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. 2010. TwitterRank: finding topic-sensitive influential twitterers. In *Proceedings of the Third International Conference on Web Search and Web Data Mining, WSDM 2010, New York, NY, USA, February 4-6, 2010*. 261–270.

Personalized Online Spell Correction for Personal Search

Jai Gupta*, Zhen Qin*, Michael Bendersky, Donald Metzler
Google Inc.
Mountain View, CA
{jaigupta,zhenqin,bemike,metzler}@google.com

ABSTRACT

Spell correction is a must-have feature for any modern search engine in applications such as web or e-commerce search. Typical spell correction solutions used in production systems consist of large indexed lookup tables based on a global model trained across many users over a large scale web corpus or a query log.

For search over personal corpora, such as email, this global solution is not sufficient, as it ignores the user's personal lexicon. Without personalization, global spelling fails to correct tail queries drawn from a user's own, often idiosyncratic, lexicon. Personalization using existing algorithms is difficult due to resource constraints and unavailability of sufficient data to build per-user models.

In this work, we propose a simple and effective personalized spell correction solution that augments existing global solutions for search over private corpora. Our event driven spell correction candidate generation method is specifically designed with personalization as the key construct. Our novel spell correction and query completion algorithms do not require complex model training and is highly efficient. The proposed solution has shown over 30% click-through rate gain on affected queries when evaluated against a range of strong commercial personal search baselines - Google's Gmail, Drive, and Calendar search production systems.

CCS CONCEPTS

• **Information systems → Information retrieval**.

KEYWORDS

Spell correction, personalization, personal search

ACM Reference Format:
Jai Gupta*, Zhen Qin*, Michael Bendersky, Donald Metzler. 2019. Personalized Online Spell Correction for Personal Search. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313706

1 INTRODUCTION

Misspelling is common in real-world information retrieval systems. More than 10% of search engine queries are misspelled [8]. This rate is even higher for tail queries, for which there is more than a 20% misspelling rate [3]. Misspellings are not just limited to unintentional typing errors, but are often a result of the challenges imposed by the spelling itself. For example, due to factors such as ambiguous word breaks, phonetically similar words, introduction of new words (e.g. project names), and inconsistent spelling rules [1], remembering the correct spelling can be surprisingly difficult even for native speakers.

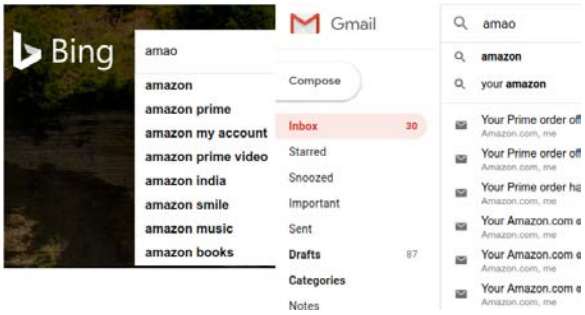


Figure 1: Online spell correction and query completion while the user types on-the-fly (Left) provides suggestions only, (Right) provides as suggestions as well as results.

For these reasons, spelling correction is essential in modern information retrieval systems today. By providing high quality spell correction capabilities, search engines can significantly reduce the amount of effort required for users to find their desired results.

The traditional spell correction approach, referred to as *offline spell correction*, is provided only after the entire query is typed and passed to the search engine (e.g. after a user presses "Enter"). Such a system assumes that any query sent to the search engine has been completely typed by the user. The modern approach, referred to as *online spell correction* [10], provides spell correction and query completion as the user is typing. It is enabled for almost all popular web search services nowadays. In this case, most of the queries under consideration are incomplete. See Fig. 1 for an illustration.

The algorithm presented in this paper is particularly designed for online spell correction systems, and specifically for personal search scenarios. Providing online spell corrections for personal search is remarkably complicated primarily due to the large and diverse vocabulary of private corpora. Furthermore, the stringent latency requirements for online spell correction systems adds to the complexity and renders most of the alternative algorithms impractical for large-scale use.

Personal search, including email search [2, 5, 24], desktop search [11], and most recently on-device search [18], has recently attracted a considerable amount of attention from the information retrieval community. The key difference between personal and public search (e.g. web search) is that users, in the personal search scenario, have access only to their own private document corpora (e.g. emails,

*Equal contribution.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313706

files, or mobile application data). This poses several new challenges for existing spell correction approaches: (1) A global lexicon may not work well or even hurt performance as each user may have a very unique private corpus. (2) Traditional machine learning-based models are difficult to train for personalization due to relatively small-sized personal corpus, sparse user interactions, and resource constraints. In many real-world cases, training a model directly from private data may not even be allowed due to privacy restrictions. (3) Tail-queries, which are difficult to handle using traditional spell correction systems, become even more problematic, since for personal search many frequently used private terms for a particular user are tail-queries from a global perspective. (4) Due to the lack of quality control, there could be more misspellings in private data itself (e.g. a typo in an email title). This is contrary to the rationale behind existing spell correction systems which build models over correctly spelled words.

This paper introduces an effective and efficient training-free spell correction solution for personal search. It can either be used independently or to complement existing global spell correction systems. We describe the spell correction candidate generation algorithm that provides a personalized lexicon created from a user's recent activities. We provide some details on how to realize such a system under real-world constraints (e.g. privacy and computing resources). We then describe a novel and simple algorithm augmenting the Levenshtein distance based approach for unified spell correction and query completion that allows serving millions of users in a real-time online spell correction system. We build and test our system using real-world online experiments conducted on millions of users for three of the world's largest commercial private corpora (Google's GMail, Drive, and Calendar), and show unanimously positive performance for each corpus across several languages, compared against strong production baselines with spell corrections learned using public web search data.

The remainder of the paper is organized as follows. We start with related works in Section 2. An overview of our system is presented in Section 3. Section 4 outlines candidate generation setup followed by detailed discussion on the spell correction and query completion algorithms in Section 5. Finally, evaluations from our experiments are presented in Section 6.

2 RELATED WORK

Most recent work on spell correction generally focuses on public search queries. Related methods usually produce a global model for all users. Also, a large amount of historical user interactions are normally needed for learning-based approaches. Such approaches typically cannot be applied to personal search settings, as the private data (e.g. search queries, corpora) available is often many orders of magnitude smaller.

For traditional offline spell correction (e.g. "did you mean"), Cucerzan and Brill [8] proposed an iterative approach that transforms unlikely queries into more likely variants based on web query logs. Chen *et al.*[7] explored web search results (queries and their top-ranked candidates) to improve existing spell correction models. Some methods [15] depend on user feedback using an existing spell correction system, which can have cold-start problems and bias concerns (e.g. position bias by the existing system). Sun *et al.*[23] learned phrase-based spell correction models from clickthrough data. Hasan *et al.*[16] collected similar user queries using user session and time information in which users "correct themselves," and uses Levenshtein distance [19] as the similarity criterion. Recently, approaches based on deep learning have become popular, such as those using sequence-to-sequence learning [13]. All of these approaches, especially deep models, typically require large amounts of user interactions or labeled data which does not scale to fully personalized spell correction and personal search scenarios.

A special less data intensive case is word embedding [21] based methods where pre-trained embeddings can be used. However, existing methods primarily use embeddings trained on a large public corpus in a user agnostic manner (e.g. Word2Vec [20] is trained over news articles). Some methods are designed in an *ad hoc* manner for specific language pairs [13, 14]. Our simple approach works well across several languages.

Online spell correction has drawn more interest in the research community [6, 9, 10] and real-world applications (e.g. Google and Bing web search engines). These systems may provide search results, but the incoming queries are often incomplete. Hence, spell correction and query completion are essential components. Chaudhuri and Kaushik [6] made use of edit distance for spell correction followed by a fuzzy search over database records to find completions. Our work unifies spell correction and query completion in an efficient augmented edit distance method. Huizhong and Bo-June [10] trained a Markov n-gram transformation model and adapted A^* search with pruning for efficiency. However, their method is specific to data-intense public corpora since it can only be applied to previously observed queries and requires a large amount of training data. Our approach does not require historical queries (which could be sparse) and complex model training.

More broadly, personalized search, including personalized query completion, has also been extensively explored [4, 22] on public corpora. Existing approaches are typically segment-based (e.g. using gender as a feature) while our work is specifically focusing on individual-level private corpora search. Some works utilize user's past search activities (e.g. queries) as signals, which could be sparse for many users. We are not aware of any prior work that focuses on individual-level spell correction in personal search, where personal lexicon is ubiquitous and user search activities are sparse, both of which pose challenges to existing personalized search methods.

3 OVERVIEW

A high level overview of our online spell correction system is present in Fig. 2, depicting the *candidate generation* and the *spell correction and query completion* components. Each user has their own personal lexicon constructed only from their set of private corpora. The lexicon can be updated periodically or in real-time (e.g. when a new document is created). The private user lexicon is small and highly relevant, and does not need any sophisticated data structure, language model or error model trained for that user.

4 CANDIDATE GENERATION

The candidate generation component is responsible for creating a personalized lexicon for every user represented as a set of contextually relevant n-grams. Generating these n-grams consists of

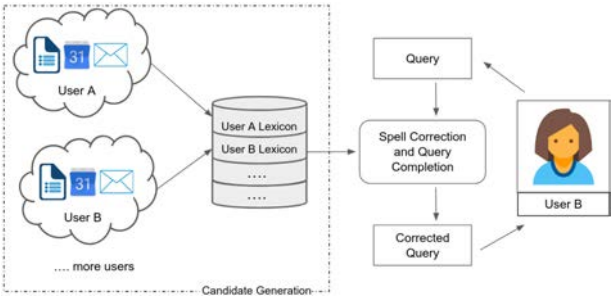


Figure 2: System Overview: Candidates from the personalized lexicon generator passed to spell correction and query completion component.

the following steps: 1) data ingestion, 2) candidate extraction, and 3) candidate update under constraints. Below, we discuss each of these steps in more detail providing general guidelines that can be useful for other applications.

4.1 Data Ingestion

We limit the input data for users to their private corpus. There are a couple of reasons for this design choice: 1) Users will only have access to their private data in personal search, so information from other users decreases the signal-to-noise ratio. 2) Privacy is critical in real-world systems, and non-aggregated data from other users can leak information. 3) By limiting candidates to the user's own set of corpora, the candidate space of n-grams for evaluation is trimmed to a significantly smaller (but highly relevant) set. Note that the user does not need to be the owner of a document, shared documents are taken into account as well.

4.2 Candidate Extraction

The lexicon of a user is composed of n-grams. We use three sources of data that are combined to form a single lexicon per user: 1) Email subjects and the email addresses of senders and recipients, 2) Drive document titles and the email addresses of the owners, and 3) Calendar event titles and the email addresses of the event participants. We found that titles (ignoring bodies) are simpler to manage and have high signal-to-noise ratio for spelling. Note that users can still search document bodies. Also, we keep email addresses as they are frequent query terms.

The snippets collected in the previous step are decomposed into candidate n-grams by standard techniques including tokenization and stopword removal. Additionally, we apply normalization techniques including lowercasing and lemmatization. Note that if documents are retrieved for a spell corrected query (see Fig. 1 right), these normalizations must be consistent with the normalization used when indexing the documents.

4.3 Candidate Update Under Constraints

Maintaining a personal lexicon for every user has many real-world constraints. For example, due to data retention policies, the system may be required to delete any candidate derived from deleted documents. Also, the system is bounded by space constraints as it stores a personal lexicon for every user. This is especially critical for real-world search engines that serve many millions of users. Additionally, a small but high-quality lexicon allows fast online spell correction by iterating over candidates without designing sophisticated index-based data structures for each user. We propose to use an event driven model that addresses these constraints in a unified manner.

The high-level idea of candidates update is to construct a small lexicon that is contextually relevant to the user when they issue a query. For this work, the model ranks snippets generated from the user's recent events based on the recency. The set of events used in our experiments are:

- GMail: Search interactions with emails.
- Drive: Interactions with documents including creation, view, edit, and comment events that are relevant to the user.
- Calendar: Ongoing and upcoming calendar events.

For example, when a user edits a document, n-grams from the document title are prioritized high in the lexicon. When space constraints are reached, the lowest ranked candidates are removed from the lexicon to accommodate new candidates. Prioritization of n-grams may not be necessary in some applications. For instance, in mobile apps search, it might be feasible to retain all of the currently installed app names.

4.4 Properties

There are several interesting properties of our event driven candidate generation method.

- Previous *search* activity and clicks are helpful but not required, unlike in most prior work. For example, n-grams from Drive documents can be from an edit event, or even initiated by another user (e.g. comment events where this user is tagged, share events, etc).
- Prioritization of n-grams based on recency provides a simple approach to address privacy and resource constraints.
- n-grams from multiple corpora (GMail, Drive, Calendar) are merged into a single lexicon and are used in search for each corpus. This is intuitively useful for our scenario. If someone has been working on a `TheWebConference` paper, it's possible that she gets emails and calendar invites discussing `TheWebConference` and has related documents about the `TheWebConference` paper. We advocate that applications should take into account such a unified representation whenever sensible, since (1) shared knowledge across corpora mitigates the data sparsity problem and (2) a single lexicon for each user is much easier to manage in practice.

5 EFFICIENT SPELL CORRECTION AND QUERY COMPLETION

We describe an efficient spell correction and query completion algorithm using a modified Levenshtein distance based approach. Given each candidate from a user's lexicon and an input query (which may be incomplete and have spelling mistakes), the algorithm calculates a modified edit distance between them, which will be used for spelling correction candidate selection or ranking. We also describe several improvements to further speed up the calculation in practice.

5.1 Levenshtein Distance

Edit distance is defined in terms of the minimum number of operations (insertions, deletions, or substitutions) required to convert the user input string to a candidate spell correction. Levenshtein Distance is a special case of edit distance with three allowed operation: insertion, deletion, and substitution with unit cost for each of them. See an example in Fig. 3. The algorithm is $O(m * n)$ where m is the length of the input string and n is the length of the candidate string.

		E	L	E	P	H	A	N	T
	0	1	2	3	4	5	6	7	8
R	1	1	2	3	4	5	6	7	8
E	2	1	2	2	3	4	5	6	7
L	3	2	1	2	3	4	5	6	7
E	4	3	2	1	2	3	4	5	6
V	5	4	3	2	2	3	4	5	6
A	6	5	4	3	3	3	3	4	5
N	7	6	5	4	4	4	4	3	4
T	8	7	6	5	5	5	5	4	3

Figure 3: Levenshtein Distance algorithm example between user input string RELEVANT and candidate string ELEPHANT.

5.2 Extending Levenshtein Distance For Query Completion

The main novelty of our algorithm lies in extending the Levenshtein distance algorithm for low-cost query completions that makes it feasible for use in online spell correction scenarios. Fig. 4 shows the edit distance matrix M for the evaluation of Levenshtein distance.

The entry *M[i][j]* denotes the minimum cost to transform i characters of the input string to j characters of output string. For example, *M[3][5]* in Fig. 4 denotes the cost of transforming ELE to ELIZA.

The most important row for extending the algorithm for query completion is the last row. If *input_length* denotes the length of the input string, the *jth* column of this row, denoted by *M[input_length][j]*, represents the cost of transforming the complete input string into the first j characters of the output string. In other words, this row represents the cost of transforming the input string to each prefix of the output string. Hence, in Fig. 4, each column in the last row represents the cost of transforming ELEZA into the corresponding prefix of ELIZABETH.

In order to support query completions, we define a new parameter *prefix_completion_cost*. In the edit distance algorithm above, each of the substitution, insertion, and deletion operations have unit cost. This new parameter allows insertion of characters at a different cost, which should be much lower than the unit cost. We take the entries in the last row and count the number of additional characters required to convert the corresponding prefix string to the complete candidate string. For the prefix corresponding to column j, we need to insert $output_length - j$ characters, where *output_length* is the length of the output string. Let *R'* be a new row denoting the overall cost after allowing prefix completion. Hence, *R'[j]* denotes the cost after transforming the input string to a prefix of length j of the output string, followed by using low cost prefix completion operations to complete it to the output string.

$$R'[j] = M[input_length][j] + prefix_completion_cost * (output_length - j). \quad (1)$$

In Fig. 4, the input string is ELEZA and the candidate string is ELIZABETH. The edit distance produced by Levenshtein distance is 5. But since *prefix_completion_cost* is 0.2, one possible sequence of operations is to replace the second "E" in the input string with an "I", which has unit cost, following by completing the obtained prefix by adding BETH at the end. This will incur a cost of $0.2 * 4 = 0.8$ resulting in a total cost of 1.8 as the minimum cost.

5.3 Further Speed-up techniques

Take the case of an input string ELEPHANT and the candidate string RELEVANT. The Levenshtein Distance between the two strings is 3 but we had to calculate all the entries in the matrix.

Suppose that the max distance for candidates that we are interested in is *max_allowed_distance* (i.e., any candidate with a higher edit distance from the query will be filtered out). For Levenshtein Distance, it can be proved that if we calculate a strip of $2 * max_allowed_distance + 1$ along the diagonal, any element outside this strip must be greater than *max_allowed_distance*. For example, for $max_allowed_distance = 3$, Fig. 3 highlights the strip of interest along the diagonal.

This reduces the complexity of the algorithm from $O(m * n)$ to $O(m * max_allowed_distance)$, which is linear in the length of the input string. In practice, we found a *max_allowed_distance* of 2 is sufficient for our use cases.

Another speed-up technique we use is to reject a candidate if its first or second characters do not match the first character of the input string. The first character of an input query is rarely found to contain any error [12]. But it was observed that due to lag in web browsers, the first character is often missing in the input query. Therefore, we relaxed our check to also allow candidates with the second character same as the first character of the input string.

6 EXPERIMENTS

Due to privacy concerns and counterfactual effects (e.g. enabling our algorithm will generate very different search results that are not available in search logs), it is difficult to perform thorough offline analysis except for limited qualitative check (e.g. Fig. 5). We thus directly conduct real-world online experiments against three of the world's largest private corpora search systems: Google's GMail, Drive, and Calendar. These systems use the sophisticated spell correction and query completion mechanisms of Google's web search, thereby providing strong baselines [17].

We demonstrate consistent and significant improvements in effectiveness over these strong production baselines. Since these are online experiments within live systems used by millions of users, we were required to meet the rigorous latency requirements of these systems. The algorithm proposed here easily achieves these requirements. More details about the performance of the proposed algorithm are described in section 6.4.

		E	L	I	Z	A	B	E	T	H
	0	1	2	3	4	5	6	7	8	9
E	1	0	1	2	3	4	5	6	7	8
L	2	1	0	1	2	3	4	5	6	7
E	3	2	1	1	2	3	4	4	5	6
Z	4	3	2	2	1	2	3	4	5	6
A	5	4	3	3	2	1	2	3	4	5
perfix completion cost = (0.2 * (9 - column_index))	1.8	1.6	1.4	1.2	1	0.8	0.6	0.4	0.2	0
R' = original cost + prefix completion cost	6.8	5.6	4.4	4.2	3	1.8	2.6	3.4	4.2	5

Figure 4: Illustration of our augmented Levenshtein Distance algorithm that unifies spell correction and query completion. Column *j* for row *R'* represents the cost of prefix completion after converting input string (ELEZA) to a prefix of length *j* of candidate string (ELIZABETH).

query_length	1	2	3	4+
Prefix Completion	X	✓	✓	✓
Spell Correction(dist = 1)	X	X	✓	✓
Spell Correction(dist = 2)	X	X	X	✓

Table 1: Desirable behavior of prefix completion and spell correction for different length of queries.

6.1 Setup

Our application scenario is "instant" search, which performs "search as you type" (e.g. see Fig. 1 right). This type of search has stringent latency requirements to ensure a high quality user experience. The spell corrected and prefix completed query is used to retrieve the final list of relevance ranked search results.

We conduct three A/B live experiments, one on each of the above mentioned corpora (GMail, Google Drive, Google Calendar) that augments their existing approaches to spell correction that use global models trained on web data. These experiments are run on millions of users for a month, resulting in many millions of queries being included in the experiment (the exact number being proprietary).

6.2 Hyperparameters

Real-world experiments that affect millions of users forbids trying many hyperparameter combinations. Thus we first define the "desirable behavior" in Tbl. 1 and tune the hyperparameters offline.

We proposed this setting because it has clear and intuitive semantics. For very short queries, neither spell correction nor prefix completion is enabled. But as the query length increases, prefix completion gets enabled followed by both spell correction and prefix completion getting enabled.

In order to achieve the above behavior, we introduce a new parameter *max_cost* which can be used to control *max_allowed_distance* based on the *query_length* as follows:

$$max_allowed_distance = max_cost - \frac{\alpha}{query_length^2} \quad (2)$$

Since Levenshtein distance has unit cost for every operation, the number of spelling errors allowed are decided by the integer part of *max_allowed_distance*. We set *max_cost* to 2.7, α to 7, and *prefix_completion_cost* to 0.08. These parameters allow us to realize the desirable behavior (compare Tbl. 1 and Tbl. 2).

query_length	1	2	3	4+
$max_allowed_distance$ $(max_cost - \frac{\alpha}{query_length^2})$	-4.3	0.95	1.92	2.26
$\lfloor max_allowed_distance \rfloor$	-5	0	1	2
Prefix Completion $(max_allowed_distance > 0)$	X	✓	✓	✓
Spell correction(dist = 1) $(\lfloor max_allowed_distance \rfloor \geq 1)$	X	X	✓	✓
Spell correction(dist = 2) $(\lfloor max_allowed_distance \rfloor \geq 2)$	X	X	X	✓

Table 2: Behavior of the algorithm after setting *max_cost* to *2.7* , α to 7 and *prefix_completion_cost* to *0.08.*

6.3 Metrics

Through the above mentioned algorithms, we measure the impact on the following search quality metrics:

- **Click-Through Rate (CTR)**: represents the fraction of queries for which a result was clicked.
- **Mean Reciprocal Rank (MRR)**: The reciprocal rank is the multiplicative inverse of the rank of the clicked result: 1 for first place, $\frac{1}{2}$ for second place and so on, and 0 for queries without click. MRR is the mean of reciprocal ranks of all query responses.
- **Has Result Rate**: represents the fraction of queries for which we have at least one result.
- **Number of Results**: represents the mean of the number of results that were shown to the user for a given query. GMail, Drive, and Calendar search engines show a maximum of 6, 5, and 4 results respectively.

For all metrics, larger numbers are better. *CTR* and *MRR* are traditional search quality metrics. *Has result rate* and *Number of Results* are particularly interesting for spell correction, because misspelled queries tend to produce either a low number of results, or no result at all. These metrics signify improved recall while CTR and MRR signify precision improvements.

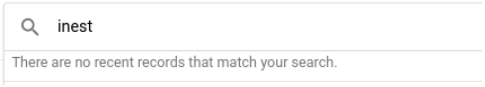
6.4 Runtime

Our per-user lexicon has at most 50 n-grams and 25 email addresses. We consider up to 7-grams. The average number of characters in the input query and candidate unigrams (from lexicon n-grams) was observed to be 5 and 8 respectively. The average number of characters in email addresses was observed to be 23.

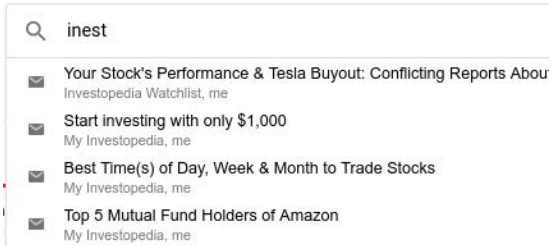
Under this configuration, the algorithm takes 0.2ms on single-thread modern hardware to produce corrections. Hence, the time taken by the spell correction component is insignificant and is well-suited for our latency-sensitive system.

6.5 Illustrative Example

We show an illustrative example in Fig. 5 to demonstrate the quality improvements. As shown, our personalized online spell correction solution is able to successfully correct the query and provide relevant search results, while the baseline system without personalized spell correction produces bad quality results.



(a) Performance without our algorithm.



(b) Performance with our algorithm enabled.

Figure 5: A qualitative illustration of our solution in GMail. "investopedia", with its prefix misspelled as "inest", is a word only relevant to certain users. (a) The production system with web spelling is unable to find any relevant email. (b) Since the user interacts frequently with documents containing the word "investopedia", it is used to spell correct "inest" and show relevant emails.

6.6 Online Experiment Results

We report search quality metrics for affected queries for each of the three corpora under consideration. Affected queries in the experiment group are those rewritten by our component (i.e. a spell correction and query completion is applied, and the corrected word is used for search). Affected queries in the control group are those that would be affected if our component were deployed (i.e. we only detect that a spell correction is possible). Our current setup results in 3% of affected queries across all corpora, which (in absolute terms) is a non-trivial amount of search queries. We summarize the results to show relative improvements in Tbl. 3.

We observe consistently positive improvements in search quality metrics across each of the three corpora. These improvements are significant and highly meaningful in the production setting.

Metrics	Email	Drive	Calendar
CTR	+31.55%	+16.04%	+38.27%
MRR	+26.00%	+10.17%	+30.41%
Has Result Rate	+15.55%	+8.51%	+43.34%
Number of Results	+31.13%	+14.53%	+76.61%

Table 3: Search quality improvements on affected queries in our online experiment. All numbers in the table are statistically significant at the $p < 0.05$ level.

Language	Email CTR	Drive CTR	Calendar CTR
English	+28.78%	+14.04%	+38.57%
Spanish	+44.58%	+11.24%	+21.51%
French	+21.02%	+9.60%	+220.93%
Japanese	+12.48%	+1.72%	+32.33%
Portuguese	+25.44%	+65.88%	+50.29%
Italian	+27.19%	+33.04%	+68.05%
Chinese	+41.86%	+28.94%	NA

Table 4: CTR improvements on affected queries in different languages across corpora.

Furthermore, these metrics are quite intuitive. In personal search, a misspelled query without good spell correction is unlikely to locate any meaningful result due to the small personal corpora, leading to low CTR and result count based metrics such as Has Result Rate.

6.7 Cross-Language Performance

We further show that our simple approach works well across different languages in Tbl. 4 in terms of CTR (we omit other metrics with similar trends due to space constraints). These languages are the top languages in terms of query count and cover more than 90% of all queries in our experiments.

Overall we get positive CTR (and other metrics) on a diverse set of languages including languages without natural delimiters like Chinese, Japanese, and Korean. There is variance in terms of quality improvements and further adapting our approach for different languages (e.g. through simple parameter tuning) could yield additional improvements as part of future work.

7 CONCLUSION

In this paper, we introduced a practical solution for personalized spell correction in personal search. We discussed our approach to generate a personal lexicon for users that ensures high quality by extracting them from the user's context derived from recent activities. We then used it in our novel spell correction and query completion algorithm that performs well under the resource constraints and latency requirements.

We further evaluated it on some of the world's largest private corpora (GMail, Google Drive, and Calendar) on hundreds of millions of users and saw highly significant improvements in search quality metrics for each corpus. Even though more sophisticated approaches are needed for global models, restricting our corrections to a small highly relevant lexicon helped us achieve great results with a simple and elegant approach. While many spell correction algorithms need sophisticated mechanisms to support different languages, our improvements are agnostic of the language.

REFERENCES

[1] 2018. I before E except after C. Retrieved 2018-10-30 from https://en.wikipedia.org/wiki/I_before_E_except_after_C

[2] Qingyao Ai, Susan T. Dumais, Nick Craswell, and Dan Liebling. 2017. Characterizing Email Search Using Large-scale Behavioral Logs and Surveys. In *WWW*. 1511–1520.

[3] Andrei Broder, Peter Ciccolo, Evgeniy Gabrilovich, Vanja Josifovski, Donald Metzler, Lance Riedel, and Jeffrey Yuan. 2009. Online Expansion of Rare Queries for Sponsored Search. In *WWW*. 511–520.

[4] Fei Cai, Shangsong Liang, and Maarten de Rijke. 2014. Time-sensitive Personalized Query Auto-Completion. In *CIKM*. 1599–1608.

[5] David Carmel, Guy Halawi, Liane Lewin-Eytan, Yoelle Maarek, and Ariel Raviv. 2015. Rank by Time or by Relevance?: Revisiting Email Search. In *CIKM*. 283–292.

[6] Surajit Chaudhuri and Raghav Kaushik. 2009. Extending Autocompletion to Tolerate Errors. In *ACM SIGMOD*. 707–718.

[7] Qing Chen, Mu Li, and Ming Zhou. 2007. Improving Query Spelling Correction Using Web Search Results. In *EMNLP-CoNLL*. 181–189.

[8] Silviu Cucerzan and Eric Brill. 2004. Spelling Correction as an Iterative Process that Exploits the Collective Knowledge of Web Users.. In *EMNLP*. 293–300.

[9] Dong Deng, Guoliang Li, He Wen, H. V. Jagadish, and Jianhua Feng. 2016. META: An Efficient Matching-based Method for Error-tolerant Autocompletion. In *VLDB*. 828–839.

[10] Huizhong Duan and Bo-June Hsu. 2011. Online Spelling Correction for Query Completion. In *WWW*. 117–126.

[11] Susan Dumais, Edward Cutrell, JJ Cadiz, Gavin Jancke, Raman Sarin, and Daniel C. Robbins. 2003. Stuff I'Ve Seen: A System for Personal Information Retrieval and Re-use. In *SIGIR*. 72–79.

[12] Mohammad Ali Elmi and Martha Evens. 1998. Spelling correction using context. In *ACL*. 360–364.

[13] Pravallika Etoori, Manoj Chinnakotla, and Radhika Mamidi. 2018. Automatic Spelling Correction for Resource-Scarce Languages using Deep Learning. In *ACL, Student Research Workshop*. 146–152.

[14] Pieter Fivez, Simon Šuster, and Walter Daelemans. 2017. Unsupervised Context-Sensitive Spelling Correction of English and Dutch Clinical Free-Text with Word and Character N-Gram Embeddings. *arXiv preprint arXiv:1710.07045* (2017).

[15] Jianfeng Gao, Xiaolong Li, Daniel Micol, Chris Quirk, and Xu Sun. 2010. A Large Scale Ranker-based System for Search Query Spelling Correction. In *COLING*. 358–366.

[16] Sasa Hasan, Carmen Heger, and Saab Mansour. 2015. Spelling Correction of User Search Queries through Statistical Machine Translation. In *EMNLP*. 451–460.

[17] Michael Herscovici, Dan Guez, and Hyung-Jin Kim. 2017 granted. Autocompletion using previously submitted query data. In *US9740780B1*.

[18] Maryam Kamvar, Melanie Kellar, Rajan Patel, and Ya Xu. 2009. Computers and Iphones and Mobile Phones, Oh My!: A Logs-based Comparison of Search Users on Different Devices. In *WWW*. 801–810.

[19] Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*. 707–710.

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. 3111–3119.

[21] Harshit Pande. 2017. Effective search space reduction for spell correction using character neural embeddings. In *EAACL*. 170–174.

[22] Milad Shokouhi. 2013. Learning to Personalize Query Auto-completion. In *SIGIR*. 103–112.

[23] Xu Sun, Jianfeng Gao, Daniel Micol, and Chris Quirk. 2010. Learning Phrase-based Spelling Error Models from Clickthrough Data. In *ACL*. 266–274.

[24] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *SIGIR*. 115–124.

Inferring Search Queries from Web Documents via a Graph-Augmented Sequence to Attention Network

Fred X. Han
University of Alberta
Edmonton, Canada
xuefei1@ualberta.ca

Di Niu
University of Alberta
Edmonton, Canada
dniu@ualberta.ca

Kunfeng Lai
Tencent
ShenZhen, China
calvinlai@tencent.com

Weidong Guo
Tencent
BeiJing, China
weidongguo@tencent.com

Yancheng He
Tencent
ShenZhen, China
collinhe@tencent.com

Yu Xu
Tencent
ShenZhen, China
henrysxu@tencent.com

ABSTRACT

We study the problem of search query inference from web documents, where a short, comprehensive natural language query is inferred from a long article. Search query generation or inference is of great value to search engines and recommenders in terms of locating potential target users and ranking content. Despite being closely related to other NLP tasks like abstract generation and keyword extraction, we point out that search query inference is, in fact, a new problem, in that the generated natural language query, which consists of a few words, is expected to be comprehensive enough to lead to the click-through of the corresponding document. Therefore, query generation requires an accurate inference of query words, as well as a deeper level of understanding on document semantic structures. Toward this end, we propose a novel generative model called the Graph-augmented Sequence to Attention (G-S2A) network. Adopting an Encoder-Decoder architecture, G-S2A incorporates a sentence-level Graph Convolutional Network (GCN), a keyword-level GCN, as well as a hierarchical recurrent neural network (RNN) into the encoder to generate structural document representations. An attentional Transformer decoder is then applied to combine different types of encoded features to generate a target query. On a query-document dataset from a real-world search engine, our model outperforms several neural generative models on a wide range of metrics.

CCS CONCEPTS

• **Information systems → Information retrieval**; **Query suggestion**;

KEYWORDS

natural language processing; information retrieval; search query generation; query suggestion

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313746

ACM Reference Format:
Fred X. Han, Di Niu, Kunfeng Lai, Weidong Guo, Yancheng He, and Yu Xu. 2019. Inferring Search Queries from Web Documents via a Graph-Augmented Sequence to Attention Network. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313746

1 INTRODUCTION

Search engines handle a massive amount of natural language queries every second. Although a search query is commonly made up of only a few words, it can reflect the user's intent as well as the theme of the retrieved documents on a conceptual level. In this paper, we consider a reverse problem of web search, which is to infer the natural language query that leads to the click-through of a web document. Automatically reverse-engineering search queries from documents is of great values to personalization in search and news feed apps for a number of reasons. First, a search query usually reveals the conceptual theme of the article retrieved by the query and thus can assist information retrieval and personalized recommendation. Second, we can discover trending topics broadly discussed on the Internet by looking at search queries. Natural language search queries, such as "best SUVs under 50k" and "jobs with least competitions", are more fine-grained than keywords and topics. As a result, concepts represented by these queries provide a unique angle to reveal user interests, as compared to pre-defined topics or key phrases. In the meantime, a query is still less specific than a headline or abstract of the document—a query covers a number of matching articles. Thus, generated queries also serve as excellent labels for fine-grained text classification and clustering.

A natural, conventional method to tag documents with search queries is to check query-document pairs in the click-through logs of a search engine. The first document a user clicks and spends some time reading can be viewed as a positive match to the query, while the remaining documents retrieved by the query may or may not be a match, depending on the search engine. As a result, the positive query-document pairs generated this way would only cover a small portion of all available documents. This fact motivates the need to reverse-engineer the best query for a given document via a generative machine learning model, such that documents can be automatically tagged with the matching search queries, which can in turn enhance personalized search experience and recommender performance.

In this work, we first formally define the problem of search query generation from web documents and show that it is unique as compared to other similar Natural Language Processing (NLP) tasks. We propose a deep learning model called **G**raph-augmented **S**equence to **A**ttention network (G-S2A) to solve this problem. To better capture the most important concepts and semantic attentions within a document, in G-S2A, we extend the existing hierarchical Recurrent Neural Network (RNN) based encoder [38] by novelly incorporating two Graph Convolutional Networks (GCN) [16], namely, a sentence-level GCN that produces a condensed graphical representation of a document, and a keyword-level GCN, which emphasizes on the interactions among critical keywords in a document. Subsequently, an attentional Transformer-based [44] decoder, which is more efficient to train, is adopted to attend over different types of the document representations to generate a natural language query.

We evaluate the proposed generative model on a real-world query-document dataset collected from the Tencent news engine in the QQ mobile browser. Extensive evaluation results suggest that by adopting a number of innovations in the encoding and decoding processes, G-S2A outperforms a number of generative baselines, including the widely popular Seq2Seq model with copying and attention mechanisms and a hierarchical RNN-based generative model [38], on all variants of ROUGE [18] and BLEU [33] scores as well as the Exact Match (EM) ratio.

Content: Heihe City, located in the northwestern part of Heilongjiang Province, has the reputation of the doors of Europe and Asia, the window of China and Russia, and is the first batch of open cities along the border. The only border city in the Heihe Sino-Russian # km border that corresponds to the Russian capital. It is also the largest and highest-profile border city. Famous attractions include Wudalianchi, Sino-Russian National Customs Park, Shaoguan East Film and Television Base, Jinhe Grand Canyon, and Ancient City. Wudalianchi Scenic Area, located in Wudalianchi City, Heihe City, Heilongjiang Province, is composed of the Wushan Lake area of Helong Lake, Heilongjiang Lake, Bailong Lake, Heilongjiang Lake, and is a national AAAAA scenic spot global geological park.

Topics: Heihe city, attractions, parks

Keywords: Wudalianchi, city, world, reputation, Jinhe, national, lake, park, attractions

1-Sentence Summary: Famous attractions include Wudalianchi, Sino-Russian National Customs Park, Shaoguan East Film and Television Base, Jinhe Grand Canyon, and Ancient City.

Query: Tourist attractions worth visiting in Heihe

Figure 1: An example illustrating the differences between topics, keywords, summaries and queries. Note that the keyword "attractions" appears in the content and all outputs, whereas the rest of the query words are inferred.

2 PROBLEM DEFINITION

The goal of search query generation is to reverse-engineer *appropriate* queries from a given piece of text. The term *appropriate* means that the generated query should be similar to what a human user would type into a search engine, and then he/she would click on the input text. By doing this, the user implicitly creates an input-query pair. In this paper, the input is a long document. We also limit the scope of this work to generate a single best query only and leave multi-query generation as future work. We model the search query generation task as a sequence-to-sequence learning problem [40], where given an input text sequence T with n tokens $\{t_1, t_2, ..., t_n\}$, the objective is to maximize the conditional probability of generating query sequence Q with m tokens $\{q_1, q_2, ..., q_m\}$.

We argue that the problem of search query generation is different from existing NLP tasks such as summarization, keyword extraction and topic detection. Fig. 1 is an example from our training data that best illustrates the differences. We use TextRank [29] to extract the keywords and the 1-sentence summary. We manually determine the topics and translate the article content from Chinese to English. We discuss the differences from other tasks separately.

Summarization: We believe search query generation and automatic summarization share the same starting point—both tasks attempt to gain a deeper semantic understanding on the document context. The key difference between them lies in the output length. Summarization consolidates a document into a shorter summary of one or several sentences. Each summary is unique to the document itself and captures its most salient information. In contrast, a query crafted in our problem is much more concise, usually consisting of only a few words. A query does not need to summarize a document accurately, but rather, should match the document on a higher conceptual level. Consider the summary and query in Fig. 1. Here the query does not contain any details about tourist attractions in Heihe, but the document is still a good match to this query because they matched on the concept of "Heihe tourist attractions".

Keyword & key-phrase extraction: The key distinguishing factor here is that query generation relies on semantic understanding while keyword extraction does not. If a word in a search query also appears in the matching document, then it is likely to be a keyword. However, the reverse is not true, as shown in Fig. 1. A document could have many keywords or key-phrases. Yet, simply assembling some of them would not produce an appropriate query. The reason is that a generated query needs to properly and semantically "grasp" the central concept discussed in a document, whereas in keyword & key-phrase extraction we only need to determine whether a word or a phrase is important.

Topic detection: The boundary between topic detection and search query generation lies in their granularities. A traditionally perceived topic is usually predefined, such as "signs of pregnancy", "baby foods", "in-door plants", while a good fraction of queries are very specific, such as "early pregnancy signs", "recipes for three-year-old babies", "best in-door plants for lazy people", which automatically discovers finer-grained topics that people pay more attention to.

In essence, the problem of search query generation is different from yet is related to all previously compared tasks. Therefore, a good model for our problem should first develop a strong semantic understanding of the article context. Then, it should be able to accurately extract critical query terms from the content as well as infer the corresponding higher-level conceptual terms to produce a concise natural language query. In Sec. 4, we design our generative query inference model taking these challenges into consideration.

3 RELATED WORK

In this section, we review the literature in closely related tasks including text summarization, keyword extraction, query-document matching, and topic models.

Summarization: Traditional approaches [7, 29] are extractive based, where salient summary sentences are discovered from a document. Deep extractive models like [4, 30, 34] are able to learn more useful latent semantic features and easily outperform traditional methods. Generative summarization models are attracting increased attention due to their potential in generating diverse summaries. [31] is the first to apply an RNN-based Encoder-Decoder setup with an attention mechanism. [37] further refines this framework by adding a pointer network to permit direct copy from the input. More recently, [41] introduces a hierarchical Encoder-Decoder with a graph attention mechanism to better capture multi-granularity features in a document. Other studies [9, 22, 23] improve summary quality from the perspectives of semantic relations, bottoms-up key-phrase selection and multi-document clustering.

Keyword & Key-phrase Extraction: As another classic NLP task, unsupervised keyword & key-phrase extraction methods like [20, 24, 29, 36] rank all the words in a document and select the top candidates as keywords. Supervised approaches such as [8, 42] initially model this task as a binary classification problem, where key-phrases are distinguished from non-key-phrases. Then, statistical machine learning methods like Naive Bayes [43], Conditional Random Fields (CRFs) [46] and Support Vector Machines (SVMs) [47] have been applied to learn from more useful input features like TF-IDF scores, Part-of-Speech (POS) tags and syntactical functions.

We would also like to point out that many extraction techniques [6, 20, 29] propose that a document should be represented as a graph, in order to discover key dependencies among words. In the query generation problem, if a query word appears in the document, then it is likely to be a keyword of the document. This motivates us to also consider graphical representations in our model.

Query-Doc Matching & Query Generation: Another relevant field is query-document matching, where the best document for a query is selected. Trivial approaches compute an aggregated similarity score between query and document word embeddings. Metrics like cosine and TF-IDF similarities, or Okapi BM25 [35] are often considered. Many deep matching models have also been proposed [14, 15, 21, 25, 32, 39], which are either representation-based that focus on semantic feature extraction, or interaction-based that emphasize pair-wise matching. The MGAN [48] is a more recent framework for query to long document matching. It models a long document as a keyword distance graph, where each vertex is a keyword and edge weights are the inverses of distances between keyword pairs. The only related work on search query generation, to the best of our knowledge, is [45]. It proposes a multi-tasking sequence-to-sequence model that simultaneously perform title compression and query generation on E-commerce product titles.

Topic Models: Conventional methods for topic modelling take either probabilistic, Singular Value Decomposition (SVD) based, or matrix factorization based approaches [1, 2, 13]. Deep learning based methods have also been proposed, [27] applies Deep Belief Nets (DBN) to discover topics from digital media content. [17] applies deep neural networks to bag-of-words document representations and achieve state-of-the-art topic modelling performance. [49] further extends the model to multimodal data. A common characteristic of these approaches is that the generated topic is always a single word or phrase.

4 MODEL

In this section, we present a detailed description of the proposed G-S2A network. Fig. 2 is a complete illustration of the model.

4.1 Hierarchical Bi-directional RNN

To reveal the sequential structure of a document, we first utilize a hierarchical RNN-based encoder [38, 41] to model the document, where the lower-level RNN encodes each sentence word-by-word into a single vector, while the higher-level RNN aggregates all sentence vectors to create a paragraph representation. We believe the hierarchical RNN is an effective approach for learning crucial sequential dependencies among the sentences. In our work, we generate a single vector representation for each sentence S_i using a Bi-directional Gated Recurrent Unit (GRU) [5] network. Next, we join sentence vectors on the first axis to form a 2-D document representation, which is then fed into another sentence-level Bi-GRU to yield sequential sentence embeddings.

4.2 Sentence-Level GCN

In order to capture more complex semantic structures beyond sequential dependencies, we further represent a document as a graph $G_s = \{V_s, E_s\}$, where the vertices V_s correspond to sentences and the edges E_s represent some connections between sentences. Our sentence-level Graph Convolutional Network (GCN) [16] is partially established on top of the word-level Bi-GRU from Sec. 4.1. Specifically, the feature of a vertex input to the GCN is the sentence vector produced by the word-level Bi-GRU.

Each edge weight in our sentence-level document graph reflects the percentage of word overlaps between a pair of sentences, which is determined by dividing the number of unique overlapping words against the total number of unique words in both sentences. The intuition here is that similar sentences have more overlapping words and thus a higher edge weight between them. At this point, we have created a new feature map that captures the pair-wise sentence similarities within a document. Next, a multi-layer GCN learns from such sentence interactions by iteratively convolving on the features of a node and its neighbors. This operation is defined by the following update rule:

$$H^{l+1} = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^l W^l), \tag{1}$$

where H^l is the output of the previous GCN layer. For the first layer, H^0 is simply the input vertex features, i.e., the sentence vectors produced by the word-level Bi-GRU. $\hat{A}$ is the adjacency matrix constructed based on edge weights, which is symmetrical. $\hat{D}$ is a diagonal degree matrix where $D_{ii} = \sum_j A_{ij}$. σ is the sigmoid activation function. W^l is the set of trainable weights in layer l. We refer interested readers to [16] for more GCN-related details.

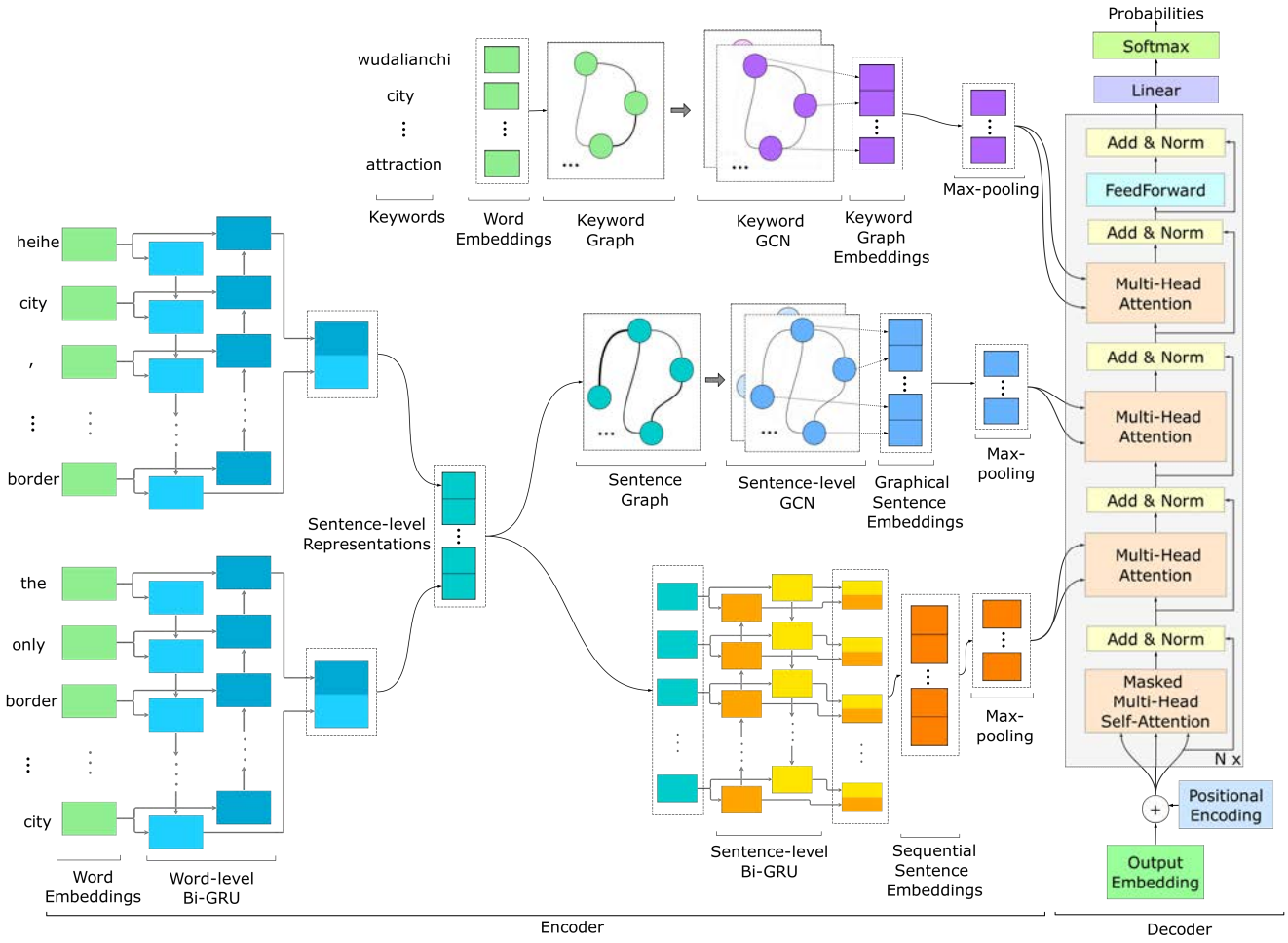


Figure 2: Full view of our proposed model.

4.3 Keyword-Level GCN

As has been mentioned in Sec. 2, if a query word appears in a document, it is usually also a keyword for that document. Therefore, in addition to the sentence-level abstractions, we should permit the decoder to learn from keyword-related information directly. To this end, we further augment our model with a document keyword distance graph similar to [48]. In the keyword distance graph $G_{kw} = \{V_{kw}, E_{kw}\}$, each vertex feature is the embedding vector of a keyword, and the edge weight between two vertices is the inverse average distance among the keywords. Specifically, for unique keywords k_i and k_j, we first locate all of their occurrences in the document. Next, we calculate the pair-wise distance between each pair of occurrences by counting the number of words between them. In other words, if there is no word between two occurrences, the distance is 1; if there is one word in-between, then the distance is 2, and so forth. We take the inverse of the average pair-wise distance. The intuition here is that keywords that are closer to each other are more closely related. Thus, their edge weight is higher in the keyword distance graph. Mathematically, this is expressed as

$$w_{ij} = \frac{1}{d_{ij}^{\text{avg}}} = \frac{m}{\sum_{j=1}^{m} d_{ij}}, \tag{2}$$

where w_{ij} is the edge weight between keywords k_i and k_j, and m is the total number of pair-wise keyword occurrences in the document. After the graph construction, we feed the node and edge features to a keyword-level GCN. The output is an enhanced keyword-level representation, which embodies the graphical keyword interactions within a document.

4.4 Transformer Decoder

Our decoder is a uni-directional Transformer [44] with two additional Multi-Head Attention modules for the GCN outputs. We choose the Transformer decoder over RNN decoders mainly because the Transformer decoder offers a more modularized internal structure, which permits us to readily incorporate more features into the decoding process. Also, the Multi-Head Attention mechanism is more efficient to train due to its easily parallelizable design.

The Multi-Head Attention module is the core component of the Transformer and it is made up of parallel layers of the Scaled Dot-Product Attention. On 2-D feature maps Q, K, and V, the Scaled Dot-Product Attention defines the operation of query Q attending on value V through keys K as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V, \tag{3}$$

where d_k is the feature dimension of K. For the masked self-attention block in Fig. 2, Q, K, V all correspond to the partially generated search query vector. For the remaining attention modules, Q is the partially generated search query vector, while K and V are the output representations from the encoder. The Multi-Head Attention module equally divides the feature dimension of Q, K and V into h heads, producing a set of $\{Q_i, K_i, V_i\}$ with i ranging from 1 to h. Then, a feed-forward layer is added before Q_i, K_i and V_i, which allows the network to learn and adjust these inputs. Finally, the output from each head is concatenated to restore the original shape and fed through another feed-forward network. Interested readers are referred to [44] for more details on the Transformer.

4.5 The Complete Model

We connect the components as depicted by Fig. 2. The word-level Bi-GRU first encodes each sentence in a document. Then, the aggregated sentence vectors propagate to the sentence-level Bi-GRU and sentence-level GCN for further learning. We believe this hierarchical encoding scheme combined with the additional sentence-level graphical embedding allow our model to better understand the semantic interactions within a document. On the other hand, the graphical embedding produced by the keyword-level GCN lets the decoder directly attend on potential query words while maintaining awareness of the document context at the same time, which aids the query term selection and generation process.

In addition to the aforementioned components, we also apply a single *Max-Pooling* layer onto all encoder outputs, since empirically, we find that the Multi-Head Attention mechanism in the decoder performs poorly when attending on long target sequences. Although we have already condensed document representations at the sentence and keyword levels, in reality, a query can be inferred from an even more compact representation—Max-Pooling serves as another stage of significance filtration before decoding.

5 EXPERIMENTATION

In this section, we present the experimental setup and compare results of our model on a real-world Chinese query document dataset against a number of generative baselines.

5.1 Dataset

A suitable dataset for our problem should contain the content of documents and the matching queries. Unfortunately, we cannot locate a publicly available dataset that meets this criteria. Summarization datasets like CNN or DailyMail [11] are not suitable because their summaries are too long to be considered queries. Query-document matching datasets like Ohsumed [12] contains too few queries, whereas datasets with enough unique user queries, such as the NFCorpus [3], have too few unique documents. The lack of a suitable dataset is another evidence that the search query generation problem is new and is less studied.

Therefore, we work with our industry partners at Tencent to create a Chinese query-document dataset, which includes real search queries collected anonymously from the QQ mobile browser. The documents in this dataset mainly consist of news articles or blog posts provided by the Tencent news engine. We construct the query-document pairs using the passive method from Sec. 1, where the first article the user clicks and spends some time reading is considered a matching document. Train, development, and test sets are randomly divided. Table 1 lists the important statistical information about our dataset.

Table 1: Dataset information

Train size	120773
Dev size	10000
Test size	10000
Average document length	286.8
Number of unique queries	10420
Min query length	2
Max query length	11
Average query length	3.2
Document vocabulary	479K
Query vocabulary	9.2K
Average doc-query word overlap	1.85

5.2 Experimental Setup

We adopt the word-overlap based ROUGE [19] and BLEU [33] metrics, which are widely used in summarization and machine translation tasks. In our work, we report the macro-averaged ROUGE-1, 2, L, BLEU-1 to 4 scores. Each metric variant cumulatively considers n-gram (1 to 4) or longest common sub-sequence (L) overlaps. ROUGE-1 is also the metric used to select the best performing model on the development set. ROUGE and BLEU scores reflect how well a generated sequence captures the truth sequence, with higher scores indicating better performance. Additionally, we report the Exact Match (EM) percentages. Because the search queries are a lot more concise, we are interested in how many generated queries match the truth exactly.

We tokenize documents and queries with the Stanford CoreNLP tool [28]. We limit the document word length at 300, All characters are lower-cased. All numbers are replaced with a # symbol. We construct the vocabulary by first combining the document and query vocabularies, then take the top 40000 most frequent words. In every experiment, we train a word embedding layer from scratch, and share it among the encoder and decoder.

For our model, we set the hidden layers of GCNs, the hidden layers of Bi-GRUs, as well as the attention dimension of the decoder to the same value. The position-wise feed-forward dimension is set to be 4 times this value. We also set a global dropout probability for all components. We fix the number of GCN and Transformer layers to 2, and the number of Bi-GRU layers to 1. The word embedding dimension is set to 300. The Max-Pooling window is set to 2. The number of Transformer attention heads is set to 8. We tune the global hidden size and dropout probability on the development set.

Table 2: Performance of various models on the test set

Models	#Weights	ROUGE-1	ROUGE-2	ROUGE-L	BLEU-1	BLEU-2	BLEU-3	BLEU-4	EM
TextRank-kw	-	12.94	0.49	7.97	8.33	2.75	1.92	1.60	0.0
TextRank-kw-cheat	-	15.38	1.11	14.75	15.38	6.94	5.17	4.43	0.0186
TextRank-kp	-	11.85	3.73	11.26	10.73	6.63	4.18	3.32	0.0276
Seq2Seq-Attn	35.7M	72.82	58.15	72.41	71.78	63.17	37.60	25.27	0.6220
Seq2Seq-Attn-Copy	35.7M	72.38	57.24	71.97	71.30	62.46	37.15	24.96	0.6147
HRED	23.7M	73.83	58.45	73.42	72.70	63.69	37.55	25.14	0.6381
Transformer	33.5M	68.87	52.51	68.36	67.51	58.14	34.66	23.31	0.5650
G-S2A	25.8M	**75.18**	**60.21**	**74.73**	**73.98**	**65.32**	**38.67**	**25.94**	**0.6426**

We use the Noam Optimizer [44] with a warm-up steps of 2000 and a label-smoothing constant of 0.1 for training. Our model converges in 100 epochs, where each epoch takes approximately 10 minutes with a mini-batch size of 128 on a GTX1070 GPU. Finally, We compare our model with the following baselines.

TextRank-kw is a simple keyword extractor based on the TextRank [29] method. We include three keyword & key-phrase extraction baselines to prove that search query generation is a more challenging task. We set the number of target keywords to 8.

TextRank-kw-cheat is also a TextRank-based keyword extractor. However, for this baseline we provide the number of words in the truth queries as additional clues.

TextRank-kp is a TextRank-based key-phrase extractor. We extract the most salient key-phrase from the document and use it as the generated query. TextRank baselines do not require training.

Seq2Seq-Attn is the classic encoder-decoder model [40] with the *general* attention mechanism [26]. We feed the entire document as one sequence into the model. We implement this baseline with a Bi-GRU encoder and a Uni-GRU attentional decoder.

Seq2Seq-Attn-Copy further incorporates the copy mechanism [10], which allows the model to directly copy important words from the input. Consider the fact that many query words also appears in the document, as indicated in Table 1. We believe this baseline is a strong competitor.

HRED is a RNN-based hierarchical encoder-decoder model proposed by [38]. Our implementation uses a hierarchical Bi-GRU as the encoder, which is essentially, the same hierarchical sequential encoder in our model. The decoder is a Uni-GRU.

Transformer is the new generative model proposed by [44]. We fix the number of encoder and decoder layers to 6 and the number of attention heads to 8. We tune the attention dimension and dropout probability. The position-wise feed-forward dimension is set to be 4 times of the attention dimension.

5.3 Results & Discussion

In addition to the metrics mentioned in Sec. 5.2, we report the number of trainable weights (#Weights) in every supervised model. We believe the number of trainable weights is critical factor to consider in real-world applications, because a large model often implies a higher cost of training.

The best hidden size for G-S2A is 128. As we can observe in Table 2, G-S2A outperforms all baselines on all metrics, which proves the effectiveness of the proposed additional components. We also note that the performance for keyword extractors are far worse than other models, which indicates that they are not suitable for our problem. Therefore, search query generation and keyword & key-phrase extraction are different problems.

Another interesting conclusion we could draw from Table 2 is that hierarchical models like HRED and G-S2A achieve better performance, but have less trainable weights. We believe this is because the hierarchical encoders break down long documents at an intuitive sentence-level, which effectively exposes its structural information. Other models without the hierarchical design would need to learn such information with more trainable weights.

Table 3: Ablation study results

Model variants	ROUGE-1	ROUGE-L
S2A	73.98	73.55
S2A+sGCN	74.15	73.72
S2A+sGCN+kwGCN	74.30	73.85
G-S2A	75.18	74.73

5.4 Ablation Study

We further evaluate G-S2A through an ablation study and report the changes in ROUGE-1 and ROUGE-L scores.

S2A only contains the hierarchical Bi-GRU encoder and the attentional Transformer decoder. We remove the two GCNs and the Max-Pooling layer from G-S2A to create this model.

S2A+sGCN incorporates the sentence-level GCN.

S2A+sGCN+kwGCN further includes the keyword-level GCN, yet without the Max-Pooling layer.

From Table 3, we observe that S2A initially outperforms the HRED model. We believe this is attributed to the Multi-Head Attention mechanism, which the HRED lacks. With the addition of more components, the performance of the model consistently improves. This proves our initial claim that exploring sentence-level and keyword-level graphical representations for long documents is extremely helpful in the search query generation task.

6 CONCLUSION AND FUTURE WORK

In this paper, we investigate the new problem of search query generation from long documents. We propose that graphical document representations help build a deeper semantic understanding of context and improve query term selection & inference. We introduce the G-S2A model, which incorporates a sentence-level GCN and keyword-level GCN to learn from the corresponding document

graphs. G-S2A outperforms a number of generative baselines, at the same time it is more light-weight. We believe the effective learning of graph features, combined with a modularized decoder, makes G-S2A a superior model for the problem of search query generation. For future work, we plan to extend our model toward multi-query generation and explore other forms of document representations.

REFERENCES

[1] Sanjeev Arora, Rong Ge, and Ankur Moitra. 2012. Learning topic models–going beyond SVD. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*. IEEE, 1–10.

[2] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[3] Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A Full-Text Learning to Rank Dataset for Medical Information Retrieval. In *European Conference on Information Retrieval*. Springer, 716–722.

[4] Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252* (2016).

[5] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[6] Adrien Bougouin Florian Boudin Béatrice Daille. 2013. Graph-Based Topic Ranking for Keyphrase Extraction. (2013).

[7] Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research* 22 (2004), 457–479.

[8] Eibe Frank, Gordon W Paynter, Ian H Witten, Carl Gutwin, and Craig G Nevill-Manning. 1999. Domain-specific keyphrase extraction. In *16th International joint conference on artificial intelligence (IJCAI 99)*, Vol. 2. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 668–673.

[9] Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018. Bottom-up abstractive summarization. *arXiv preprint arXiv:1808.10792* (2018).

[10] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393* (2016).

[11] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. 1693–1701.

[12] William Hersh, Chris Buckley, TJ Leone, and David Hickam. 1994. OHSUMED: an interactive retrieval evaluation and new large test collection for research. In *SIGIR-94*. Springer, 192–201.

[13] Thomas Hofmann. 2000. Learning the similarity of documents: An information-geometric approach to document retrieval and categorization. In *Advances in neural information processing systems*. 914–920.

[14] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*. 2042–2050.

[15] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2333–2338.

[16] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[17] Stanislas Lauly, Yin Zheng, Alexandre Allauzen, and Hugo Larochelle. 2017. Document neural autoregressive distribution estimation. *The Journal of Machine Learning Research* 18, 1 (2017), 4046–4069.

[18] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*. http://research.microsoft.com/~cyl/download/papers/WAS2004.pdf

[19] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out* (2004).

[20] Marina Litvak, Mark Last, Hen Aizenman, Inbal Gobits, and Abraham Kandel. 2011. DegExt-A language-independent graph-based keyphrase extractor. In *Advances in Intelligent Web Mastering–3*. Springer, 121–130.

[21] Bang Liu, Ting Zhang, Di Niu, Jinghong Lin, Kunfeng Lai, and Yu Xu. 2018. Matching Long Text Documents via Graph Convolutional Networks. *arXiv preprint arXiv:1802.07459* (2018).

[22] Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. 2018. Toward abstractive summarization using semantic representations. *arXiv preprint arXiv:1805.10399* (2018).

[23] Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198* (2018).

[24] Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*. Association for Computational Linguistics, 257–266.

[25] Zhengdong Lu and Hang Li. 2013. A deep architecture for matching short texts. In *Advances in Neural Information Processing Systems*. 1367–1375.

[26] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[27] Lars Maaloe, Morten Arngren, and Ole Winther. 2015. Deep belief nets for topic modeling. *arXiv preprint arXiv:1501.04325* (2015).

[28] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*. 55–60.

[29] Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

[30] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents.. In *AAAI*. 3075–3081.

[31] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023* (2016).

[32] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition.. In *AAAI*. 2793–2799.

[33] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 311–318.

[34] Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304* (2017).

[35] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[36] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory* (2010), 1–20.

[37] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368* (2017).

[38] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models.. In *AAAI*, Vol. 16. 3776–3784.

[39] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 373–374.

[40] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[41] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1171–1181.

[42] Peter D Turney. 2000. Learning algorithms for keyphrase extraction. *Information retrieval* 2, 4 (2000), 303–336.

[43] Yasin Uzun. 2005. Keyword extraction using naive bayes. In *Bilkent University, Department of Computer Science, Turkey www. cs. bilkent. edu. tr/~guvenir/courses/CS550/Workshop/Yasin_Uzun. pdf*.

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[45] Jingang Wang, Junfeng Tian, Long Qiu, Sheng Li, Jun Lang, Luo Si, and Man Lan. 2018. A Multi-task Learning Approach for Improving Product Title Compression with User Search Log Data. *arXiv preprint arXiv:1801.01725* (2018).

[46] Chengzhi Zhang. 2008. Automatic keyword extraction from documents using conditional random fields. *Journal of Computational Information Systems* 4, 3 (2008), 1169–1180.

[47] Kuo Zhang, Hui Xu, Jie Tang, and Juanzi Li. 2006. Keyword extraction using support vector machine. In *International Conference on Web-Age Information Management*. Springer, 85–96.

[48] Ting Zhang, Bang Liu, Di Niu, Kunfeng Lai, and Yu Xu. 2018. Multiresolution Graph Attention Networks for Relevance Matching. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 933–942.

[49] Yin Zheng, Yu-Jin Zhang, and Hugo Larochelle. 2016. A deep and autoregressive approach for topic modeling of multimodal data. *IEEE transactions on pattern analysis and machine intelligence* 38, 6 (2016), 1056–1069.

Learning Novelty-Aware Ranking of Answers to Complex Questions

Shahar Harel
Technion - Israel Institute of Technology
sshahar@cs.technion.ac.il

Sefi Albo
Technion - Israel Institute of Technology
sefialbo@campus.technion.ac.il

Eugene Agichtein
Emory University and Amazon Research
eugene.agichtein@emory.edu

Kira Radinsky
Technion - Israel Institute of Technology
kirar@cs.technion.ac.il

ABSTRACT

Result ranking diversification has become an important issue for web search, summarization, and question answering. For more complex questions with multiple aspects, such as those in community-based question answering (CQA) sites, a retrieval system should provide a *diversified* set of *relevant* results, addressing the different aspects of the query, while minimizing redundancy or repetition. We present a new method, *DRN* , which learns novelty-related features from unlabeled data with minimal social signals, to emphasize diversity in ranking. Specifically, *DRN* parameterizes question-answer interactions via an LSTM representation, coupled with an extension of neural tensor network, which in turn is combined with a novelty-driven sampling approach to automatically generate training data. *DRN* provides a novel and general approach to complex question answering diversification and suggests promising directions for search improvements.

ACM Reference Format:
Shahar Harel, Sefi Albo, Eugene Agichtein, and Kira Radinsky. 2019. Learning Novelty-Aware Ranking of Answers to Complex Questions. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313457

1 INTRODUCTION

Search engines and conversational assistants are expected to answer increasingly complex questions, such as requests for advise or opinions. Perhaps the most important source are Community Question Answering (CQA) sites, which offer an opportunity for millions of online users to contribute, share, and discover knowledge. The number of answers for community questions can range from a few, to dozens, but it is not realistic to expect a user to read them all. At the same time, a user might be interested in several aspects of the answers, and would benefit from being exposed to several relevant answer aspects. For example, when looking for recommendations of a movie or a restaurant, one could be interested in several viewpoints, and not just a single answer.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313457

An ideal ranking of answers to such a question should combine relevance to the asked question, as well as novelty – by addressing different aspects of the question instead of repeating already seen information in previous answers.

So far, the best performing state-of-the-art systems addressing the CQA novelty problem have been supervised, requiring extensive training datasets [20]. For these methods, two types of training labels are required – relevance labels, and explicit aspects diversity judgments. Relevance judgments datasets are more common, and can be inferred based on implied ranking of users on social sites (e.g., through clicks or user ratings of answers) and indeed have been used in a form of a distant supervision for training learning-to-rank systems. In contrast, diversity-labeled datasets in large amounts need to be obtained for each new domain to enable supervised learning. Additionally, supervised learning-to-rank systems typically apply statistical models to features, engineered for a specific domain. The required feature engineering for each problem leads to complex and brittle systems with many steps and dependencies. In contrast, we present an approach that avoids feature engineering, yet reaches high ranking performance for multiple types of questions, by introducing a novel deep learning algorithm leveraging large unlabeled data sets with minimal distant supervision based on social signals, and does not require direct diversity judgments.

In this work, we introduce *DRN* (Diversity Ranking Network), which, to the best of our knowledge, is the first neural network approach for community-question-answering ranking that addresses both relevance of the answers, and their marginal novelty. Specifically, our model captures relationships between questions and their corresponding answers with a non-linear tensor layer, which promotes answers that are both relevant to the question, and diversified with respect to other answers. Our method combines the trained neural network scores with an iterative algorithm for efficiently ranking the community answers with respect to previously chosen answers. Our system reaches state-of-the-art results, compared to previous novelty-ranking unsupervised approaches, and shows comparable results to the supervised methods that use extensive feature engineering and labeled datasets. Specifically, our contributions are:

- A novelty-driven sampling approach (Section 3.3), for leveraging unlabeled data and minimal social signals as distant supervision to optimize a novelty-driven loss function.
- A novel end-to-end deep-learning approach, modeling interactions of question and answers for identifying novelty and relevance of answers without manual feature engineering.

Question	How can I lose weight I'm 73 lbs I need to exercise but I'm fat and lazy so can you please help me does someone have tips to lose weight?
Answer 1	Exercise, Nutrient dense foods. If you lose weight very quickly you will be a lot less likely to keep it off. The first thing I would suggest is to believe in you.
Answer 2	The more you feel good about yourself the easier and faster it is to lose weight. Exercise, exercise, exercise! Please consult your doctor before beginning an exercise or weight loss program. Safest way is to lose 2-3 lbs a week.
Answer 3	diet always trumps exercise. Add or lose muscle. Too much body fat is not good for your health. Your lifestyle habits can influence . You have to diet and exercise.
Answer 4	I've gone down from 103 kg to 89 kg in three months by reducing my caloric intake and walking at least 10 kms a day and sometimes I even up to 20 km
Answer 5	trust me it wont be healthy why are people seriously giving you tips on losing weight? even if it is through healthy means, you don't need tips on losing weight.
Answer 6	If indeed some infections contribute to obesity in people , we could have a potentially very simple and effective prevention strategy vaccination...

Figure 1: An example question with candidate answers from the TREC LiveQA dataset, with relevant answers partitioned into propositions corresponding to aspects. Each aspect is color-coded according to human annotation.

- A new method for novelty-aware scoring of answers for complex questions over social media (Section 3) using a neural architecture, and an iterative algorithm for ranking answers, reaching state-of-the-art results.

We report results of an extensive experimental evaluation of different variations of our *DRN* method, showing significant improvements over state-of-the-art approaches (Section 5) on two benchmark datasets, one of which we created and will share with the research community.

Our *DRN* model lays the foundation for deep learning for answer diversification, that is aware of both relevance and novelty. Making it a promising approach for many other ranking tasks.

2 RELATED WORK

Ranking social media content, and specifically CQA content, has been an active area of research [18, 19]. CQA content in particular has a number of distinct characteristics, such as the duality of questions and answers, and social signals, that have been explored for improving retrieval performance. For example, Xue et al. [35] experimented with variations of language models for retrieving question-answer pairs from CQA archives, while Wang et al. [31] demonstrated improvements by modeling question-answer relationships. It has also been shown that additional features for CQA ranking can be more naturally incorporated using a supervised Learning-to-Rank (LTR) approach (e.g., see Bian et al. [3], Cao et al. [5], Surdeanu et al. [27]), with the added benefit that LTR systems can be optimized for the measure of interest, such as result diversity, in addition to relevance. However, supervised LTR approaches require extensive training data, a shortcoming that we address in the next section by introducing an deep learning method trained in a distant supervised manner. While numerous studies on document novelty and diversification for Web Search and IR have been done in the past [17, 23, 33] the CQA setting exhibits significant differences. First, the answers provided to a specific question are mostly relevant, as opposed to document ranking where many documents are irrelevant to the query. Therefore, common methods in IR, such as MMR perform significantly worse on this task (see Section 4.1). Additionally, many of those methods [33] are supervised and require significant labeling per domain, which is impractical for CQA. Second, CQA answers are shorter than most documents and therefore pose additional challenges to diversification. Finally, the length of the queries in Web search is significantly shorter compared to the length of the CQA questions. Specifically in the CQA domain, Szpektor et al. [28] presented a recommendation algorithm for suggesting questions to potential answerers in CQA sites. They showed that ignoring diversity degrades recommendation performance. Recently, deep learning techniques have been applied to many variants of LTR and Question Answering [13, 30, 32, 34] and specifically for Learning To Rank in CQA, e.g., [6, 21, 33, 36, 37]. While building on these previous works, our effort is distinct in that we develop an end-to-end method for automatically learning to diversify CQA ranking with only minimal distant supervision based on social signals.

3 *DRN*: DEEP NETWORK ANSWER DIVERSIFICATION MODEL

Our goal is to train a ranking function for CQA that emphasizes both relevance and marginal novelty. Consider an example CQA question in Figure 1, which shows the first six out of eleven answers to the question. The first three answers contain a wide range of aspects, with some shared across answers, and others specific to a certain answer. Answer 4 is relevant to the question, but does not add novelty with respect to former answers. Answer 5 is not highly relevant to the question, and does not address many aspects. Finally, Answer 6 is irrelevant and should be ranked last.

We model the problem as a supervised-learning problem, optimizing to rank a triplet consisting a question and two possible answers, where a positive example is one consisting of both relevant and diverse answers. We learn a fixed-size vector representation of question and candidate answers using a LSTM layer. Then, triplets of question and two answers are combined with a neural tensor layer, and combined score for each triplet is computed. Figure 2 presents *DRN* architecture at high level. The key innovation of our

work is a novel, novelty-driven sampling process, which emphasizes relevance and novelty simultaneously for training the model. A greedy algorithm selects at each round an answer that is both relevant to the question and introduces most novelty with respect to former ranked answers.

3.1 Low-Dimensional Embedding of Question and Answer Text

To be able to perform the neural network matrix operations over question and answer, we need to represent them as a fixed-length vector. In this work, we use a deep recurrent neural network architecture (RNN) to process sequentially each word in the question and the answers, and to produce a low dimensional vector at each step. Intuitively, the output vector at each step represents the information given in the sentence, until the current word. We use an LSTM model [11, 12] and obtain the last word in the sentence embedding as the semantic vector representation for the sentence.

3.2 Modeling Question and Answer Context

An answer to a question can be a good candidate or a bad candidate depending on the context where it is given, in our case, within the context of the question asked. Modeling the interactions between the answer and the question is a key aspect in being able to judge its relevance and usefulness. One could try to model these interactions by designing hand-crafted features, which is not an easy task. Our approach is to discover such features automatically. In the past, the use of tensors to model interactions was shown to be successful [21, 26]. While those models were required to model a single interaction, to capture *novelty*, it must capture the interaction of the question with several answers. To address *novelty*, we argue that one should consider interactions between *all entries* in a triplet ($question, answerA, answerB$). With these interactions, we can model both relevance of an answer to a question, and the novelty that the answer introduces with respect to previous chosen answer. Intuitively, we want this score to be high if the first answer is relevant to the question, and diverse with respect to the second answer. Modeling such a score function can be done by linearly combining two score functions, each trained for different desired property, e.g., relevance and novelty. Using a more general non-linear function, allows the algorithm to learn to optimize both relevance and diversity simultaneously. To accomplish this, we present an extension to [26] for modeling triplets, denoted $TNTN$, the parameters of the network will be shared across the triplet, thus able to encode features representing semantic interactions between *all entries* in the triplet. This is done by firstly introducing the question into the parametrization, $W = M^{[1:n\cdot r]}v_q$, and then multiplying the answers with a bilinear tensor product. $TNTN$ can be described as follows:

$$s(q, a, a') = u^T f\left(v_a^T W^{[1:r]} v_{a'} + V\begin{bmatrix} v_a \\ v_q \\ v_{a'} \end{bmatrix} + b\right) \quad (1)$$

where $M^{[1:n\cdot r]} \in R^{n\times n\times n\times r}$, $V \in R^{r\times 3n}$,$b \in R^r$ and $u \in R^r$ $v_q, v_a \in R^n$.

v_q, v_a and v'_a are the question and two answer vector representations from LSTM layer, f is a nonlinearity applied element-wise,

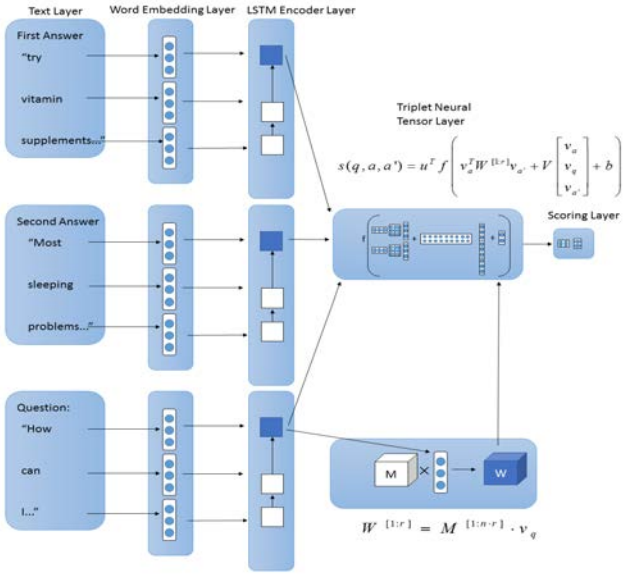


Figure 2: Visualization of the *DRN* network

The tensor product $M^{[1:n\cdot r]}v_q$ is done by multiplying $n \cdot r$ matrices of $n \times n$ by v_q, thus results at r matrices of size $n \times n$, or in other words a tensor $W^{[1:r]} \in R^{n\times n\times r}$. The bilinear tensor product $v_a^T W^{[1:r]} v'_a$ results in a vector $h \in R^r$, where each entry is computed by one slice i = $1, \ldots, r$ of the tensor. The other parameters to be learned are $V \in R^{r\times 3n}$, $b \in R^r$ and $u \in R^r$.

3.3 Novelty Driven Sampling

Intuitively, our scoring function should learn to address both relevance and novelty. To tackle this, we introduce a novel sampling approach, which presents three types of examples to our network, one positive and two types of negative examples. To guide the sampling, we introduce a $Novelty$ function that is able to measure the amount of diversity between two given answers.

To construct the positive example for a given question – $(q, Ba, BaDiv)$ for both high relevance and novelty, we select the best answer Ba in a distant supervised manner to be the most voted answer by Yahoo! users, which is commonly assumed to be highly relevant to the question. To address novelty with respect to Ba, we construct $BaDiv = argmax_{a\in A(q)} Novelty(Ba, a)$ via the $Novelty$ function.

We construct two types of negative examples – one expressing low novelty of a relevant answer, and the second representing low relevance. The first negative example is constructed by selecting an answer Ba as before, and $BaSim = argmin_{a\in A(q)} Novelty(Ba, a)$ is an answer the $Novelty$ function suggests to be similar to Ba and therefore not Novel.

The second type of a negative example, $(q, Ra, RaDiv)$ is constructed by selecting a random answer Ra to a random question, and therefore assumed to have poor relevance to the question q of the triplet. Such a triplet is used as a negative example, even though the answer $RaDiv = argmax_{a\in A(Random(q))} Novelty(Ra, a)$ is *novel* with respect to Ra by the $Novelty$ function.

Several *Novelty* function families can be used for the sampling process without any significant alternations to *DRN* . We experimented with cosine distance over topic representation of the text, e.g. LDA [4], and over embedding space, e.g. doc2vec [16]. On validation dataset, we found that the LDA function performed better for the diversification task. Thus, we represent each answer with its corresponding LDA topic probabilities vectors and their level of diversity is measured with the cosine distance.

Loss function for training *DRN* : Our loss function is inspired by the MMR [7] framework. Our approach generalizes the idea of combining relevance and diversity functions by encoding them both into a loss function. We train the network with max-margin criterion, which provides an alternative to probabilistic, likelihood-based estimation methods by concentrating directly on the robustness of the decision boundary of a model [29]. Intuitively, a good example will be separated in space by some margin from the sampled corrupted examples. Our loss function for diversification is defined as:

$$\begin{aligned} L = \sum_{C,C'} & - [s(q, Ba, Ba_{Div}) - s(q, Ba, Ba_{Sim})] \\ & - [s(q, Ba, Ba_{Div}) - s(q, Ra, Ra_{Div})] \\ & + \gamma + \lambda \left\|\Theta\right\|_2^2 \end{aligned} \tag{2}$$

where $\gamma > 0$ is the margin hyper-parameter, C is the training collection of good examples $(q, Ba, BaDiv)$ triplets from the data and C' denotes the collection of all corrupted triplet examples $(q, Ba, BaSim)$ and $(q, Ra, RaDiv)$. To minimize our objective function we used the Adam Optimizer [14].

Algorithm 1 Ranking for Diversification Algorithm

1: NN Inference: Compute s(q,a,a') - $n \times n$ scores matrix for question q
2: R $\leftarrow$ *SelectFirstAnswer*
3: for $j = 1$ to $|A(q)| - 1$ do
4: R $\leftarrow argmax_a \Sigma_{a' \in R} s(q, a, a')$
5: end

3.4 Ranking Algorithm

In the previous sections we described the construction of an embedding of questions and answers (section 3.1), which is then used in a neural tensor network to assign relevance and novelty scores. In this Section we leverage these scores into a novelty-driven ranking Algorithm (1). The algorithm greedily selects answers to maximize scores learned by the neural network. For each question and the set of its n corresponding answers, we perform $n \times n$ forward inference steps, thus producing a score for every combination of the question and two of its answer. This step results in an $n \times n$ scores matrix for each question and its answers. We then select the first answer and add it to the ranked answers list. We note that in this work, we focus on the diversification aspect, i.e., given a selected first answer, we wish to select relevant answers that introduce new aspects. To initialize our ranking algorithm, we must choose the first answer. Selecting the best answer to a question is a well researched task in the literature [2, 25]. Inspired by prior work, we choose the first answer using a classifier optimized to select answer with high number of aspects.

Going back to our ranking algorithm, in line 4 we iteratively choose the next answer as the one with the highest sum of scores with respect to the question and the answers chosen so far. Intuitively, the answer that maximize this step, is the one with high relevance to the question, and simultaneously a high level of novelty with respect to the answers chosen in earlier steps, where all these answers are taken into account.

4 EXPERIMENTAL SETTING

In this section we describe the state of the art baselines that we compare to our method, *DRN* , as well as the evaluation metric and the CQA datasets used in the experiments. All the methods, including baselines, first used basic pre-processing, including: tokenization, stop-words removal, and spell checking, which is an important step in CQA domain due to its noisy nature.

Tuning and Final Configuration Based on a validation set, we tuned our network hyper-parameters, with final values of r = 5, learning rate = 0.001, batch size = 100, negative examples = one of each type, regularization λ = 0.0001, LSTM state size = 50.

4.1 Evaluation Metrics

One of the most prominent metrics for diversification measurement in QA and IR systems is $\alpha NDCG$ [9]. To compute $\alpha NDCG$ we first define $r_k^i = 1$ as an indicator function for the event where the i-th answer contain proposition related to the k-th aspect. The complete $\alpha NDCG$ formulation is:

$$\alpha NDCG = \sum_k \frac{1}{\log_2(k+1)} \sum_i r_k^i (1-\alpha)^{s_{i,k-1}} \tag{3}$$

where $s_{i,k-1} = \sum_{j=1}^{k-1} r_j^i$.

Intuitively, $\alpha NDCG$ rewards answers with the highest discounted cumulative gain, where gain is measured by the novelty the answer aspects introduce with respect to previously ranked answers.

Additionally, we present results with an additional common IR metric for diversification – ERR-IA [8]. ERR-IA is defined as the expectation of ERR over the different aspects, where R_i is the probability that the user will be satisfied with the i-th document (or in our case, the i-th answer) and n is the number of answers retrieved. Specifically, *ERR* is defined as:

$$ERR = \sum_{r=1}^{n} \frac{1}{r} \prod_{i=1}^{r-1} (1 - R_i) R_r \tag{4}$$

Together, $\alpha NDCG$ and ERR-IA comprise the two main metrics recommended to use for evaluating search result diversity, as surveyed in reference [24].

4.2 Baseline Methods

BM25 [22]: Originally proposed for ad-hoc information retrieval, BM25 ranks documents according to term frequencies.

MMR [7]: The maximal marginal relevance is a traditional method promoting diversity by combining two standalone similarity and diversity functions.

LDARanker [15]: This is an unsupervised algorithm that aims to diversify ranking by selecting answers with good coverage over the set of all answers to a question.

SimRanker [20]: Serves as our state-of-the-art *supervised* baseline method. This algorithm use the notion of propositions and leverage on data with diversity labels for training parameters to

predict whether two propositions are diverse, then a greedy procedure selects iteratively answer that best combines a set of relevant propositions that are both diverse, and important.

SimRanker-ESA [20]: Serves as our state-of-the-art *unsupervised* baseline method. *SimRanker − ESA* only differs from the supervised version in the similarity function between propositions, as it uses a cosine similarity of the "Explicit Semantic Similarity", or CQA-ESA[10] representations of each proposition.

Due to lack of access to a proprietary method described in [20], our method *DRN* is compared to *SimRanker* and *SimRanker − ESA* only on the first benchmark with results given by the authors, for calibration of our method to the state of the art. On the second benchmark, we compare *DRN* with *BM*25, *MMR* and *LDARanker*.

4.3 Datasets

In 2009, Yahoo published a dataset with questions and their corresponding answers from Yahoo Answers web site. The corpus contains 4,483,032 questions and their answers. In addition to question and answer text, the corpus contains a small amount of metadata, (e.g. most voted answer). We use a subset of this dataset with randomly chosen 250k questions and their corresponding answers for generating training data with our novel sampling approach.

Yahoo-Novelty Dataset: Recently, [20] developed a dataset specifically for diversification, the dataset consist a random sample from Health category in Yahoo answers. The data consist of questions and answers, with their relevant propositions clustered into aspects.

LiveQA-Novelty Dataset: To be able to test our system on larger scale, and on questions from a variety of categories, we have developed an additional larger dataset designed specifically for diversification, following the methodology presented by Omari et al. [20]. The questions were sampled from three categories of the LiveQA 2015 dataset [1]: Health, Pets, and Arts & Humanities. We then selected the subset of the questions with at least 7 answers. First, each answer is partitioned into propositions, and each proposition is then manually judged for its relevance to the question. Propositions found to be relevant to a question from all of its answers are then grouped into aspects.

Specifically, annotation of the LiveQA-Novelty data was done in 3 phases: (1) Identifying relevant propositions in answers[1] - We have used Amazon Mechanical Turk for the annotations. Each HIT* (A Human Intelligence Task) contained a question and an answer partitioned automatically into propositions, and the MTurk workers were given the instruction to mark each proposition as relevant or not to the question. (2) Filtering questions [2] - Second step of relevant propositions filtering was done manually by the authors. During this phase we removed all the questions with fewer than 5 total relevant propositions in their answers. (3) Aspects clustering - the authors manually grouped relevant propositions for each of the test question into clusters representing aspects. We computed the answer/aspect-cluster agreement of two annotators using Cohen's kappa statistic. The score $kappa = 0.72$ suggests moderate to high agreement indicating the labeling is reliable. As a result of this labeling, our LiveQA-Novelty dataset contains 207 questions, with 2,488 answers in total, with at least 7 answers for each question. We make our labeled LiveQA-Novelty Dataset available for the research community [1].

[1] https://github.com/shaharharel/CQA_Diversification

Model	α NDCG (Yahoo)	α NDCG (LiveQA)	ERR-IA (Yahoo)	ERR-IA (LiveQA)
LDARanker	0.62	0.58	0.6	0.51
BM25	0.67	0.52	0.53	0.43
MMR	0.66	0.56	0.57	0.46
SimRanker-ESA	0.75	-	-	-
SimRanker	**0.80***	-	-	-
DRN	**0.81***	**0.68***	**0.65***	**0.56***

Table 1: $\alpha NDCG$ for all methods over Yahoo-Novelty dataset (Left) and LiveQA-Novelty dataset (Right); significant improvements are marked with *.

5 EXPERIMENTAL RESULTS

In this section we present our experimental results, and analyze the performance of *DRN* . Then, we take deeper look at *DRN* internal latent space behavior.

5.1 Main Results

Table 1 report the performance of *DRN* and the baseline algorithms on Yahoo-Novelty and LiveQA-Novelty datasets respectively. Statistically significant ($p < 0.05$) improvements are marked with "*"s. The results show that on both datasets, *DRN* outperforms the baselines *BM*25, *MMR*, *LDARanker* and *SimRanker − ESA*. We hypothesize that our algorithm is able to automatically learn novelty-driven features defined over interactions of questions and answers, while *BM*25 and *SimRanker − ESA* are mostly dependent on semantic similarity between questions and answers. One might consider *LDARanker* comparable to our method, as both are based on a topic models (i.e., *DRN* utlizes it during the sampling phase). However, *LDARanker* does not consider the interactions with the question when considering the novelty-driven answers ranking, thereby ignoring essential information. The only method directly comparable to ours is the *SimRanker*-Supervised. However, this is a supervised approach, which requires data with diversity labels for training. In contrast, our method *DRN* is trained in a distant supervised manner, which requires only minimal social signals and does not require any human-annotated diversity labels.

We can observe a significant difference between the performances measured by αNDCG in the Tables 1. The reason for that, is the rate of relevant answers in Yahoo Answers Novelty Based Answer Ranking dataset compared to liveQA Answers Novelty Based Answer Ranking dataset. The former holds a relevant answer rate of 0.8 which means, 80% of the answers contain 1 relevant aspect or more, while the latter holds a rate of 0.45, which makes it a more realistic and difficult environment for all discussed algorithms. Also liveQA Answers contains questions from various categories and therefore complicates the ranking.

5.2 Performance Analysis

In the example in Figure 3, eight relevant aspects appear in ten answers that were provided for the question. *DRN* is able to cover three of them in the first answer, adds another four in the following one where two of them are novel aspects. The third answer the algorithm choose has no relevant aspects – presenting an example where our approach might fail. This failure can be explained by the topic-driven nature of our *Novelty* function (Section 3). The third answer contains a lot of seemingly new topics (e.g., "meditation" and "house plant"). Since our algorithm rewards diverse topics, it

Question	How can I sleep longer? Ok, so it may seem like a silly question, but how do I sleep longer? No matter how tired I am at night, it's impossible for me to sleep more than 5 or 6 hours...
Answer 1; Aspects: 2,4,7	try vitamin supplements. . At night be sure to turn off all electronics that receive radio waves! . Make sure your diet and exercise are at a standard level.
Answer 2; Aspects: 0, 1, 2, 4	Most sleeping problems are probably caused by stress, depression, anxiety and worry. To sleep better just relax and switch off. some light exercise. Common OTC sleep aids include Chamomile tea, 5-HTP, Melatonin.
Answer 3; Aspects: ∅	When it comes to sleep, if I don't get enough, my day gets all out of wack.got a house plant for my bedroom, tried meditation before bed.started drinking chamomile tea.In aggregate, all those things really just moved the needle...
Answer 4; Aspects: 3, 4, 5, 6, 7	Nice shower before bed helps. make sure no disturbance from any noise. Good nutrition and vitamins everyday helps too who knows maybe the cellular radiation has effect on your sleep rhythm. Dont watch tv too late. Make sure room is dark enough so no disruption in the morning.

Figure 3: Example question with its top four answers ranked by *DRN*. The aspects are numbered in the left column and colored in different colors across answers according to human gold-standard annotation.

BM25 Ranking
Answer 1 - Aspects: 2, 4, 7
Answer 2 - Aspects: 2
Answer 3 - Aspects: ∅
Answer 4 - Aspects: 0, 1, 2, 4

Table 2: BM25 ranking by aspects over the same example as in figure 3.

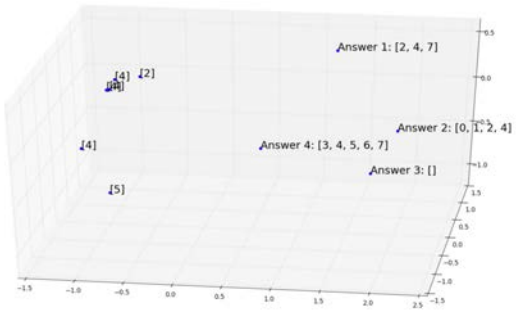


Figure 4: *DRN* LSTM answers vector space for the same question as in figure 3.

might fail if the answer topics are diverse and have a disambiguation in relevance. This issue can be addressed by introducing the network with other types of negative examples which are not based on topic modeling. Another possible explanation is that this answer is controversially annotated as having no relevant aspects to the question. Going back to the ranking, we observe that the 4th ranked answer contains five relevant aspects, of which four are novel aspects. For comparison, table 2 reports the BM25 ranking over the same example from figure 3. This example illustrates the capability of *DRN* to emphasize relevance and novelty simultaneously, by selecting answers that are both relevant to the question and novel with respect to the previously chosen answers.

As discussed earlier, our sampling process and objective function are aimed at learning novelty-driven features automatically, thus we expect the LSTM representation for the answers to express these novel features. Figure 4 gives some intuition of *DRN* behavior and latent features for the example answers from 3. Using the PCA method, we project the answer LSTM vector representations into three most informative dimensions. Each dot in space is an answer, annotated with the list of the aspects it contains. Answers presented in Figure 3 are also labeled with the answer number. We can see a large gap between answers contain low number of aspects (left side), and the answers containing many of the aspects (right side). Moreover, observing the aspects' behavior across answers, we can see that answers with similar aspects tend to be nearby in space, for example, all four answers which contain only the aspect 4 are close, while answers with only the aspects 2 or 5 are further. On the right side of the figure, the answers are far from each other because of the large number of novel aspects they introduce with respect to each other. One exception is Answer 3, analyzed previously.

6 CONCLUSIONS

Modeling both relevance and novelty for ranking answers is a key requirement for effective and user-friendly complex question answering and result presentation, especially with increasing adaption of conversational interfaces and small-screen mobile devices. Previous state of the art approaches have been supervised, requiring extensive manual annotations to obtain both relevance and diversity labels. To address this problem, we presented *DRN*, a novel method which reaches state-of-the-art results using distant supervision for relevance, and no labeled data for diversity. Our method automatically learns a novelty-aware answer scoring function from CQA archives, based on the Neural Tensor Network representations of the interactions between questions and answers, and a novel sampling approach for training, which emphasizes relevance and novelty. We present a new answer ranking algorithm utilizing our model, which iteratively ranks answers by their diversity and relevance with respect to the formerly chosen answers. Experimental results based on two data sets demonstrated that our algorithm outperformed the state of the art baselines that required significant feature engineering and labeled datasets. In future work, we plan to address multiple forms of diversity, by extending our sampling phase to emphasize multiple types of *Novelty* functions and novelty-driven negative examples. Our method could be extended to other settings of complex question answering over social media, and more broadly to complex question answering and web search tasks that require retrieving relevant, useful, and novel answers.

REFERENCES

[1] Eugene Agichtein, David Carmel, Dan Pelleg, Yuval Pinter, and Donna Harman. 2015. Overview of the TREC 2015 LiveQA Track.. In *TREC*.

[2] Eugene Agichtein, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. 2008. Finding high-quality content in social media. In *Proceedings of the 2008 international conference on web search and data mining*. ACM, 183–194.

[3] Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. 2008. Finding the right facts in the crowd: factoid question answering over social media. In *Proceedings of the 17th international conference on World Wide Web*. ACM.

[4] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[5] Xin Cao, Gao Cong, Bin Cui, and Christian S Jensen. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *Proceedings of the 19th international conference on World wide web*. ACM, 201–210.

[6] Ziqiang Cao, Furu Wei, Li Dong, Sujian Li, and Ming Zhou. 2015. Ranking with Recursive Neural Networks and Its Application to Multi-Document Summarization.. In *AAAI*. 2153–2159.

[7] Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 335–336.

[8] Olivier Chapelle, Donald Metlzer, Ya Zhang, and Pierre Grinspan. 2009. Expected reciprocal rank for graded relevance. In *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 621–630.

[9] Charles LA Clarke, Maheedhar Kolla, Gordon V Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM.

[10] Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis.. In *IJcAI*, Vol. 7. 1606–1611.

[11] Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850* (2013).

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[13] Mohit Iyyer, Jordan L Boyd-Graber, Leonardo Max Batista Claudino, Richard Socher, and Hal Daumé III. 2014. A Neural Network for Factoid Question Answering over Paragraphs.. In *EMNLP*. 633–644.

[14] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Ralf Krestel and Nima Dokoohaki. 2011. Diversifying product review rankings: Getting the full picture. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01*. IEEE Computer Society, 138–145.

[16] Quoc V Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents.. In *ICML*, Vol. 14. 1188–1196.

[17] Shangsong Liang, Zhaochun Ren, and Maarten De Rijke. 2014. Fusion helps diversification. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval (SIGIR)*. 303–312.

[18] Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 27–48.

[19] Preslav Nakov, Lluís Màrquez, Walid Magdy, Alessandro Moschitti, Jim Glass, and Bilal Randeree. 2015. Semeval-2015 task 3: Answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. 269–281.

[20] Adi Omari, David Carmel, Oleg Rokhlenko, and Idan Szpektor. 2016. Novelty based ranking of human answers for community questions. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 215–224.

[21] Xipeng Qiu and Xuanjing Huang. 2015. Convolutional Neural Tensor Network Architecture for Community-Based Question Answering.. In *IJCAI*. 1305–1311.

[22] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[23] Rodrygo LT Santos, Craig Macdonald, and Iadh Ounis. 2010. Exploiting query reformulations for web search result diversification. In *Proceedings of the 19th international conference on World wide web*. ACM, 881–890.

[24] Rodrygo LT Santos, Craig Macdonald, Iadh Ounis, et al. 2015. Search result diversification. *Foundations and Trends® in Information Retrieval* 9, 1 (2015), 1–90.

[25] Chirag Shah and Jefferey Pomerantz. 2010. Evaluating and predicting answer quality in community QA. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 411–418.

[26] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[27] Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2011. Learning to rank answers to non-factoid questions from web collections. *Computational linguistics* 37, 2 (2011), 351–383.

[28] Idan Szpektor, Yoelle Maarek, and Dan Pelleg. 2013. When relevance is not enough: promoting diversity and freshness in personalized question recommendation. In *Proceedings of the 22nd international conference on World Wide Web (WWW)*. 1249–1260.

[29] Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. 2005. Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd international conference on Machine learning*. ACM, 896–903.

[30] Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 707–712.

[31] Xin-Jing Wang, Xudong Tu, Dan Feng, and Lei Zhang. 2009. Ranking community answers by modeling question-answer relationships via analogical reasoning. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 179–186.

[32] Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural qa as simple as possible but not simpler. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. 271–280.

[33] Long Xia, Jun Xu, Yanyan Lan, Jiafeng Guo, and Xueqi Cheng. 2016. Modeling Document Novelty with Neural Tensor Network for Search Result Diversification. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 395–404.

[34] Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *International Conference on Machine Learning*. 2397–2406.

[35] Xiaobing Xue, Jiwoon Jeon, and W Bruce Croft. 2008. Retrieval models for question and answer archives. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 475–482.

[36] Xiaodong Zhang, Sujian Li, Lei Sha, and Houfeng Wang. 2017. Attentive Interactive Neural Networks for Answer Selection in Community Question Answering.. In *AAAI*. 3525–3531.

[37] Zhou Zhao, Hanqing Lu, Vincent W Zheng, Deng Cai, Xiaofei He, and Yueting Zhuang. 2017. Community-Based Question Answering via Asymmetric Multi-Faceted Ranking Network Learning.. In *AAAI*. 3532–3539.

Spatio-Temporal Capsule-based Reinforcement Learning for Mobility-on-Demand Network Coordination

Suining He
The University of Michigan–Ann Arbor
suiningh@umich.edu

Kang G. Shin
The University of Michigan–Ann Arbor
kgshin@umich.edu

ABSTRACT

As an alternative means of convenient and smart transportation, mobility-on-demand (MOD), typified by online ride-sharing and connected taxicabs, has been rapidly growing and spreading worldwide. The large volume of complex traffic and the uncertainty of market supplies/demands have made it essential for many MOD service providers to *proactively* dispatch vehicles towards ride-seekers.

To meet this need effectively, we propose STRide, an MOD coordination-learning mechanism reinforced spatio-temporally with capsules. We formalize the adaptive coordination of vehicles into a reinforcement learning framework. STRide incorporates spatial and temporal distributions of supplies (vehicles) and demands (ride requests), customers' preferences and other external factors. A novel spatio-temporal capsule neural network is designed to predict the provider's rewards based on MOD network states, vehicles and their dispatch actions. This way, the MOD platform adapts itself to the supply-demand dynamics with the best potential rewards. We have conducted extensive data analytics and experimental evaluation with three large-scale datasets (~21 million rides from Uber, Yellow Taxis and Didi). STRide is shown to outperform state-of-the-arts, substantially reducing request-rejection rate and passenger waiting time, and also increasing the service provider's profits, often making 30% improvement over state-of-the-arts.

CCS CONCEPTS

• **Information systems** → **Spatial-temporal systems**; **Data mining**.

KEYWORDS

Mobility-on-demand, ride-sharing platform, smart transportation coordination, reinforcement learning, capsule network, smart city.

ACM Reference Format:
Suining He and Kang G. Shin. 2019. Spatio-Temporal Capsule-based Reinforcement Learning for Mobility-on-Demand Network Coordination . In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313401

1 INTRODUCTION

By integrating online information of rides demands and supplies, transit network operation and communication, cooperative mobility-on-demand (MOD) systems, such as Uber, Lyft, Didi and connected taxicabs, have provided unprecedented transportation alternatives.

Given its significant economic and social values, we should coordinate the MOD operations, *i.e.,* dispatching (matching) supplies (i.e., available vehicles/drivers) towards demands (*i.e.,* passengers/requesters/riders). A coordination policy/strategy usually uses current observations to determine where and when to relocate vehicles for maximization of MOD service providers' profit and satisfaction of riders' desire/requirement. During each phase of coordination, the idle/vacant MOD vehicles are dispatched towards different service zones (a discretized city map), and matched with their nearest requesters. The resultant rides also "connect" the zones, forming an MOD network.

However, increasingly complex urban traffic networks and miscoordination of demand-supply dynamics often under/over-serve many service zones, thus degrading the providers' profitability, service quality, passenger satisfaction and drivers' enthusiasm. The problem is particularly severe during rush hours when people travel in similar directions between home and work. Despite numerous efforts and enormous historical ride data available, designing an MOD coordination mechanism remains to be difficult for the following reasons.

First, due to urbanization and MOD market expansion, the *static* coordination based upon old data in some zones cannot be applied and scaled throughout the dynamically-changing MOD network, thus calling for an adaptive mechanism. Second, coordination of complex MOD supplies and demands makes sequential and long-term effects. A single vehicle dispatching action may introduce profound consequences to the environment and other vehicles, that cannot be easily foreseen by a heuristic coordination mechanism. Third, there usually exists *multi-level periodicity* within traffic routines, commute patterns, annual festival events, and ride preferences. Weather can also affect the ride requests, which will, for example, surge during rainy hours. Without comprehensive modeling of these factors, the conventional coordination methods cannot easily and accurately capture the repeating patterns.

We meet these challenges by proposing STRide, deep reinforcement learning (RL) based on spatio-temporal capsules for coordinating the MOD network. Specifically, we design an RL framework with a data-driven emulator, adjusting the coordination policy with an online *self-adapting* mechanism. A state in this RL represents the spatial MOD demand, supply and external temporal factors, and each action there represents the zones that vehicle can be relocated to. The multi-objective reward function characterizes the platform profitability, cost and service coverage, whose long-term values are maximized by STRide. The framework also takes into account the

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313401

contextual scopes of vehicles and travel time estimation to emulate a fine-grained learning environment.

Since it is difficult to specify the long-term coordination effects on future demands and supplies, we design a novel capsule-based neural network to comprehensively learn the relationship between observed states, coordination actions and potential rewards. This way, STRide foresees an upcoming demand–supply imbalance and proactively provides optimal decisions, achieving fine-grained coordination. STRide incorporates the spatial demand-supply dynamics, temporal external influence factors and ride preferences to capture the complex periodicity in MOD ride demands. The optimized coordination policy is stored in the capsule network for efficient online use by service providers.

This paper makes the following three major contributions:

- *Comprehensive Learning Framework for Proactive & Efficient MOD Coordination*: We have formalized the dynamic vehicle dispatching and rides matching into a spatio-temporal RL framework. STRide accounts for spatial and temporal distributions of supplies and demands, ride preferences and other external factors. Using double deep Q-network (DQN) learning and contextual scope processing, STRide learns how to dispatch each participating vehicle effectively and efficiently.
- *Spatio-Temporal Capsule-based Policy Learning*: Within the RL framework, we integrate a novel spatio-temporal capsule neural network, accurately and efficiently mapping the observed MOD network states, vehicles and dispatch actions to the provider's rewards. This way, STRide finds the spatio-temporally adaptive coordination policy with the best platform profitability, service quality, passenger satisfaction and driver incentivization.
- *Extensive Data-driven Analytics & Evaluation*: Based on above, we have conducted large-scale (a total of 21,150,163 rides) data analytics and comprehensive experimental evaluation of STRide. Our results based on data of Uber/Yellow Taxi, New York City and Didi, Chengdu, China show STRide to outperform other state-of-the-arts, lowering request reject rate, and passenger waiting time, and also enhancing the platform's profits (often making more than 30% improvement over the state-of-the-arts).

2 PROBLEM FORMULATION & SYSTEM OVERVIEW

2.1 Preliminary

Zones, Rides & MOD Network: A large-area city map is discretized into R zones, $\mathbf{Z} = \{z_1, \ldots, z_R\}$, while balancing between coordination granularity and computation efficiency. The shape and size of a zone can be subject to the performance goal and platform customization. In our prototype, the entire map is discretized into $L_{\text{lon}} \times L_{\text{lat}}$ rectangular "zones", the shape of each of which may be altered to reflect the existence of buildings, rivers, roads, etc.

Let $\mathbf{T}(i, j)$ be the set of directed MOD rides from z_i to z_j, and $\mathbf{T} = \{\mathbf{T}(i, j);\ i, j = 1, \ldots, z_R\}$. The MOD network is then a directed graph $\mathbb{G}(\mathbf{Z}, \mathbf{T})$ and formed by the end-to-end (e2e) MOD rides $\mathbf{T}$ across R different zones. The thus-formed $\mathbb{G}(\mathbf{Z}, \mathbf{T})$ serves as the *environment* of our coordination learning.

Discretization of Time Domain: The data structure for STRide's training is prepared by following the practice in RL framework [29] and slicing ride records $\mathbf{T}$ (sorted by their pick-up timestamps) into N_{epi} identical non-overlapping chunks. Each chunk, or *episode*, is a time period within a day, during which the MOD platform maximizes the financial returns. Similarly to the map discretization, the time domain of each trip chunk is discretized into N_{step} equal intervals (30 min each in our prototype). Each interval k ($k \in \{1, \ldots, K\}$) corresponds to a *learning step*. Then, for each z_i in a step k, we let $D_i^{(k)}$ be the number of aggregated pick-ups (requests).

Vehicles: Considering the connectivity of the MOD network, the service provider monitors the status of $M^{(k)}$ participating vehicles in step k. Each of these vehicles, denoted as v_m ($m \in \left\{1, \ldots, M^{(k)}\right\}$), contains its unique identifier, the current location (longitude and latitude *loc*; zone z_i), availability (vacant or not), and destination of ride/dispatch, if any. Each vehicle is in one of the following 4 states: *dispatching* (relocating to another zone for potential pick-ups), *matching* (heading to the pick-up location after accept), *occupied* (between pick-up and drop-off), and *vacant* (staying in the same zone after "dispatching" or "occupied" is over). Besides, each v_m is associated with estimated time of arrival, ETA_m.

2.2 Problem Formulation

Coordination Problem: Given the spatio-temporal distributions of MOD ride pick-ups and drop-offs, we would like to proactively decide on *where and when* each available vehicle should be coordinated to serve ride requests so as to maximize the service provider's profitability and passenger-perceived service quality.

This problem is characterized with the following five major components: $\{\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{U}}, \tau, \pi, Q\}$.

a) **State $\boldsymbol{\mathcal{S}}$**: We consider that an MOD network or environment is coordinated by a web-based/online coordination center, or an *agent*, connecting a large group of spatially-distributed vehicles and passengers with mobile apps. The state space $\boldsymbol{\mathcal{S}}^{(k)}$ that the agent observed at the learning step k consists of:

- *MOD vehicles* $\mathbf{V}$: the 2-D (an $L_{\text{lon}} \times L_{\text{lat}}$ matrix) distribution of all vehicles v_m's. From $\mathbf{V}$, we derive the 2-D distribution of vacant/available vehicles, denoted as $\mathbf{A}$. Both $\mathbf{V}$ and $\mathbf{A}$ capture the participating vehicles.
- *Departures/demands/pick-ups* $\mathbf{D}$: the 2-D distribution ($L_{\text{lon}} \times L_{\text{lat}}$) of requests or departures of passengers.
- *External factors* $\mathbf{E}$: Since the different events (e.g., holidays or not), meteorological metrics (e.g., wind speed) and weather conditions (e.g., rainy or snowy) affect the demand/supply [38] as well as the resultant coordination performance, we form them into an L_{ext}-D vector as the additional hints for model training.

We model distributions of $\mathbf{D}$, $\mathbf{A}$ and $\mathbf{V}$ into frames of *heatmaps* (each is an $L_{\text{lon}} \times L_{\text{lat}}$ matrix; warmer colors indicate more passenger requests or vehicles) such that they can be processed by our spatio-temporal learning algorithm as *input features*. At each step k, we find the comprehensive state or observation $\boldsymbol{\mathcal{S}}^{(k)}$ as the important spatial features characterizing the MOD environment, *i.e.,*

$$\boldsymbol{\mathcal{S}}^{(k)} = \left\{\mathbf{D}^{(k)}, \mathbf{A}^{(k)}, \mathbf{V}^{(k)}, \mathbf{E}^{(k)}\right\}. \tag{1}$$

b) **Action $\boldsymbol{\mathcal{U}}$**: An action is a coordination solution. The action space $\boldsymbol{\mathcal{U}}_m$ for each vehicle v_m is defined as a set of discrete transits to any of its neighboring rectangle zones, plus staying where it is. Let L ($\leq R$) be the number of neighboring zones that a vehicle can relocate to. For each v_m, we consider $\boldsymbol{\mathcal{U}}_m$ of size $(L+1)$ is centered

at her/his current zone, and d_{ml} represents a destination zone l relative to the current center. Then, the dispatch *action space* for v_m is $\mathcal{U}_m = \{d_{m0}, d_{m1}, \ldots, d_{ml}, \ldots, d_{mL}\}$, where d_{m0} represents the action of staying where it resides.

c) **Reward** τ: Given the settings of states $\mathcal{S}$ and actions $\mathcal{U}$, each of $M^{(k)}$ available vehicles dispatched by the agent, arrives at the next state, and is returned with an immediate *reward, i.e.,* $\mathcal{S} \times \mathcal{U} \times \mathcal{S} \rightarrow \tau$. Specifically, each vehicle (driver) m at learning step k is associated with an instant reward function $\tau_m^{(k)}$. $\tau_m^{(k)}$ takes into account the platform revenues and coordination cost (often subsidized by MOD service providers [11]), such that maximizing individual rewards can also maximize the platform profitability (Sec. 3.2). Driver incentivization and passenger satisfaction are also figured in the fine-grained multi-objective formulation.

d) **Policy** π & **Long-term Value** Q: Intuitively, the MOD coordination actions have long-term effects upon the vehicle distributions. STRide aims at maximizing the expected reward in each episode, and mitigating the demand-supply imbalance. From the platform's perspectives, this coordination policy, as a joint mapping function, not only predicts gaps between future demands and supplies based on current market status, but also yields the highest reward by relocating idle vehicles. Specifically, at each step k, we find the subsequent cumulative returns based on a certain policy of coordination $\mathcal{S} \times \mathcal{U} \rightarrow \pi$, with a weight parameter $\gamma \in (0, 1)$ that differentiates rewards in terms of temporal proximity. For each v_m, at step k STRide expects the long-term values in the remaining steps of the episode as

$$Q_m^{(k)} = \tau_m^{(k)} + \gamma \tau_m^{(k+1)} + \cdots + \gamma^{K-k} \tau_m^{(K)}. \tag{2}$$

Then, the optimal value function $Q_m^*(\cdot)$ is the maximum expected long-term reward of all candidate dispatch decisions, *i.e.,*

$$Q_m^*\left(\mathcal{S}^{(k)}, \mathcal{U}^{(k)}\right) \triangleq \max_{\pi} \mathbb{E}\left[Q_m^{(k)} \middle| \mathcal{S}^{(k)}, \mathcal{U}^{(k)}, \pi\right], \tag{3}$$

which fulfills the Bellman equation [29] for iterative learning as

$$Q_m^*\left(\mathcal{S}^{(k)}, \mathcal{U}^{(k)}\right) = \mathbb{E}_{\mathcal{S}'}\left[\tau_m^{(k)} + \gamma \max_{\mathcal{U}'} Q_m\left(\mathcal{S}', \mathcal{U}'\right) \middle| \mathcal{S}^{(k)}, \mathcal{U}^{(k)}\right], \tag{4}$$

where $\mathcal{S}'$ and $\mathcal{U}'$ represent the given state and action of subsequent step $(k+1)$. Note that vehicles with the same spatio-temporal states are considered homogeneous. In other words, vehicles in the same zone and step (time interval) share the same coordination policy and value function. Due to the difficulty of specifying the sophisticated long-term value Q, we design a spatio-temporal deep capsule network as the Q-*network* (Sec. 4) to store policy π.

2.3 System Framework & Flow

Fig. 1 overviews the STRide's system design with two phases, *offline learning* and *online coordination*. The entire system consists of following 3 major components:

1) **Feature Extraction & Processing**: Given the MOD data of rides from mobile apps and other external factors from the MOD (online) environment (Sec. 3.1), STRide first extracts features, structuring the batched data into states $\mathcal{S}$ as Eq. (1) for ease of the following offline emulation and model learning. The historical (offline learning) and real-time (online coordination) external knowledge can be obtained via weather station records or Internet [38].

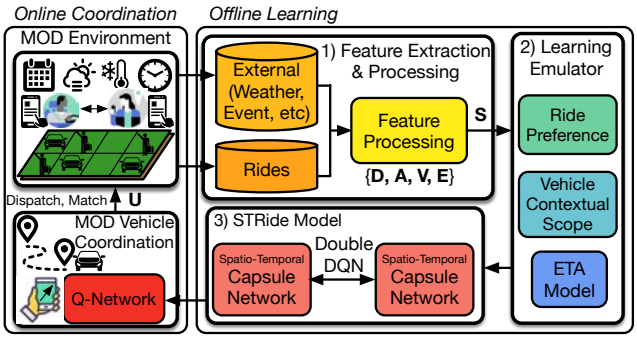


Fig. 1: The system framework of STRide.

2) **Learning Emulator**: In our RL design, the emulator [29, 31] provides the offline and emulated environment derived from real-world ride data and the city map for model training. Its design (Sec. 3.2) accounts for the ride preferences, the contextual scope for each individual vehicle. Meanwhile, STRide finds the estimated time of arrival (ETA) for each matching or dispatching transit. Note that the emulator can be cold-started by several episodes of rides without STRide's coordination. Historical ride data can also be used for offline learning to train an initial model of STRide, .

3) **STRide Model**: The environment states, the agent's coordination actions and resultant rewards at each step are used to train our deep capsule network model for policy learning, minimizing the Q estimation loss. During model training, the emulator is reset given each episode of ride data, while the model is updated *w.r.t.* each learning step k. After taking multiple steps, the double DQN mechanism learns the coordination policy, and the STRide model is returned for next episode. The offline learning ends when all episodes of data are examined. The learned capsule network returns the optimal policy (subject to ϵ-greedy mechanism) and actions for online coordination, e.g., routing vacant vehicles to destinations via mobile apps [19].

3 DATA-DRIVEN LEARNING EMULATOR

3.1 Data Sets for Analytics & Evaluation

Our emulator is built for data analytics and performance evaluation based on the following three large-scale MOD datasets:

- *Uber, NYC* [5]: The ride sharing data of Uber in New York City (NYC), June 2015, contains a total of 2,816,895 rides.
- *Yellow Taxis, NYC* [4]: The ride data of Yellow Taxis in May 2016 contains 11,588,760 rides, their pick-up/drop-off locations and timestamps. Taxis share many similar characteristics with ride-sharing/hailing vehicles (e.g., driving distance and time), making this experimental study feasible [27].
- *Didi, Chengdu* [6]: The ride-sharing data provided by Didi Chuxing [6], contains a total of 6,744,508 rides (with pick-up/drop-off locations and timestamps) from the city of Chengdu, Sichuan Province, China in November 2016.

For each of these datasets, we also include local weather [2], weekday/weekend and festivals/events as the external factors (**E**) in the model, as summarized in Fig. 3. We also obtain the road network from OpenStreetMap (OSM) [3].

3.2 Design of Comprehensive Learning Emulator

We design and implement a *data-driven* learning platform that emulates a real-world MOD platform with the following considerations.

a) **Contextual setting**: A driver usually focuses on her/his neighborhood observations or *contexts* for (re)location. The relocation policy for each vehicle is made fine-grained by cropping and padding (zeros) [13] in the heatmap frames in $\boldsymbol{\mathcal{S}}$ such that the *local* scope $\boldsymbol{\mathcal{S}}_m$ of each v_m (*i.e.,* the observed contexts in its neighborhood) is centered around its current zone. This way, STRide can learn the coordination policy based on each vehicle's neighborhood observations. This also reflects the drivers' tendency in (re)locating to nearby zones for less time/fuel consumption. STRide will also pay less computation overhead. At step k, we find the contextual scopes of $M^{(k)}$ available vehicles from the global state $\boldsymbol{\mathcal{S}}^{(k)}$. The state of each vehicle m to be dispatched is $\boldsymbol{\mathcal{S}}_m^{(k)} = \left\{\mathbf{D}_m^{(k)}, \mathbf{A}_m^{(k)}, \mathbf{V}_m^{(k)}, \mathbf{E}_m^{(k)}\right\}$. Each of $\mathbf{D}_m^{(k)}$, $\mathbf{A}_m^{(k)}$ and $\mathbf{V}_m^{(k)}$ is cropped from $\boldsymbol{\mathcal{S}}^{(k)}$ into a sub-frame of smaller size $L'_{\text{lon}} \times L'_{\text{lat}}$ ($L'_{\text{lon}} < L_{\text{lon}}$ and $L'_{\text{lat}} < L_{\text{lat}}$), and centered at m's current zone like $\boldsymbol{\mathcal{U}}_m$. $\mathbf{E}_m^{(k)}$ is an $(L_{\text{ext}} + 2)$-D vector consisting of L_{ext} external factors and v_m's current 2-D location.

In a sequential-learning setting [29], we consider all idle vehicles sequentially determine where to relocate to. Each vehicle considers all other peers' present status, while its dispatch decision is independent of the peers' next moves/actions [37]. In state $\boldsymbol{\mathcal{S}}_m^{(k)}$, v_m's relocation to a neighboring zone $d_{ml}^{(k)}$ at step k leads to a subsequent $\boldsymbol{\mathcal{S}}_m^{(k+1)}$ observation and an instant reward $\tau_m^{(k)}$. Then, a *transition sample* $\mathbf{H}^{(k)}$ of consecutive states at k and $(k+1)$ is given by

$$\mathbf{H}^{(k)} = \left\{\boldsymbol{\mathcal{S}}_m^{(k)}, d_{ml}^{(k)}, \boldsymbol{\mathcal{S}}_m^{(k+1)}, \tau_m^{(k)}\right\}, \tag{5}$$

which is stored in the *experience replay* $\mathbf{O}$ and resampled for further training of STRide model in Sec. 4.

b) **Multi-objective ride reward function**: For each vehicle, STRide accounts for the platform profitability (in proportion to the vehicle's earning), service coverage and configuration cost in characterizing the agent's reward function. Intuitively, more rewards are expected if more ride fares are earned and less time of dispatching/idle driving is consumed. To accommodate this, we consider for each v_m in the k-th step two critical perspectives: *earning score* $P_m^{(k)}$ in terms of fulfilled rides and revenues, and the *relocation cost* $F_m^{(k)}$ related to dispatching and idle driving time (and fuel consumption). Then, v_m's reward at step k is defined as the sum of earning scores minus the relocation costs in a window of w steps, *i.e.,*

$$\tau_m^{(k)} \triangleq \sum\nolimits_{\widetilde{k}=k-w}^{k} \left(\alpha P_m^{(\widetilde{k})} - F_m^{(\widetilde{k})}\right), \tag{6}$$

where $\alpha > 0$ is an adjustable parameter. In our prototype, we empirically set $w = 15$. In other words, the more pick-up revenues and the less relocation time and fuel consumption, the higher reward a vehicle could achieve. The earning score $P_m^{(k)}$ is defined as the weighted sum of serviced ride fares among the zones based on the historical ride preferences $\omega(i,j)$'s, *i.e.,*

$$P_m^{(k)} \triangleq \sum\nolimits_{i=1}^{R} \sum\nolimits_{j=1}^{R} \left|\mathbf{T}_m^{(k)}(i,j)\right| \cdot e_{ij}, \tag{7}$$

where $\left|\mathbf{T}_m^{(k)}(i,j)\right|$ represents the number of actual rides from zones i to j provided by v_m, and e_{ij} is the resultant earning. In our emulator prototype, the driver revenue or ride fare is set as

$$e_{ij} = a \cdot \delta_{ij} + b, \quad a > 0, \quad b > 0, \tag{8}$$

where a is the unit price *w.r.t.* distance δ_{ij} and b is the base price (our prototype uses the local ride rates [5, 6]).

c) **Ride preference & zone-to-zone connectivity**: The drivers and passengers usually have frequent travel patterns among zones due to their commute routes and ride preferences. Considering the MOD network $\mathbb{G}$ connecting the zones $\mathbf{Z}$ with rides $\mathbf{T}$, inspired by the *first-order proximity* in network embedding [30], we design a ride preference metric $\omega(i,j)$ for rides $\mathbf{T}(i,j)$ between z_i and z_j as

$$\omega(i,j) \triangleq \left(1 + \exp\left(-\vec{c}_{ij} \cdot \vec{c}_{ji}\right)\right)^{-1} \tag{9}$$

where the relative proportion of rides $\vec{c}_{ij}$ is defined as a vector of

$$\vec{c}_{ij} \triangleq \left[\frac{|\mathbf{T}(i,j)|}{\sum_{k=1,k\neq i}^{R} |\mathbf{T}(i,k)|}, \quad 1 - \frac{|\mathbf{T}(i,j)|}{\sum_{k=1,k\neq i}^{R} |\mathbf{T}(i,k)|}\right]. \tag{10}$$

That is, the more rides recorded between z_i and z_j, the higher $\omega(i,j)$, indicating a stronger connectivity between the two zones.

We incorporate the above ride tendency of zones into STRide's formulation, not relying only upon individually-aggregated demand values. Our prototype discretizes each day into four 6-hour periods, and aggregates rides of the same period for different sets of $\omega(i,j)$'s. Then, to calculate $F_m^{(k)}$, we consider in Eq. (6) the relocation travel time $\Gamma(i,j)$, and the recent weight $\omega(i,j)$ between z_i and z_j belonging to the same historical periods (say, 06:00 – 12:00 of the same weekday in its preceding week), and find the weighted sum of

$$F_m^{(k)} \triangleq \sum\nolimits_{i=1}^{R} \sum\nolimits_{j=1}^{R} \frac{\beta \Gamma(i,j)}{\omega(i,j)}, \tag{11}$$

where $\beta > 0$ is the unit cost related to the relocation time or petrol prices [1] (say, a subsidy rate by the MOD platform for dispatching [10]). Rides starting and ending within the same zone, *i.e.,* when $i = j$, are also considered within Eq. (9).

d) **Estimated time of arrivals**: For fine-grained evaluation, the learning emulator also calculates the estimated travel time between two locations for characterizing: (1) the time of travel from current location to destination when passengers are served; (2) the time of passenger wait if vehicles and certain passengers are matched; and (3) the time of idle driving if vehicles are dispatched to another location for potential requests. Specifically, we implement a random forest regression [14] (other more advanced models [18, 33, 34] may apply) to estimate the travel time ETA_m between one location to another given the input vector of start/end coordinates, length of the shortest path between them, day of a week and hour/minute of a day belonging to the start time. The length of interim road segments during the vehicle's travel are derived from OSM [3].

4 CAPSULE-BASED COORDINATION LEARNING

Characterizing Q, the relationship between states, actions and long-term values, is essential for STRide to decide on the best coordination policy π while adapting to the environment. We propose the use of capsule-based Q-network for more comprehensive MOD coordination learning, dynamically capturing and learning $Q_m(\mathcal{S}_m, \mathcal{U}_m)$. A *capsule* is a structured group of neurons [28], and many capsules can be grouped together in one layer. A link between two capsules in consecutive layers becomes a vector. Such vectorized (instead of scalar) representation of data is propagated between layers. Hence, more comprehensive ride patterns, including spatial co-existence

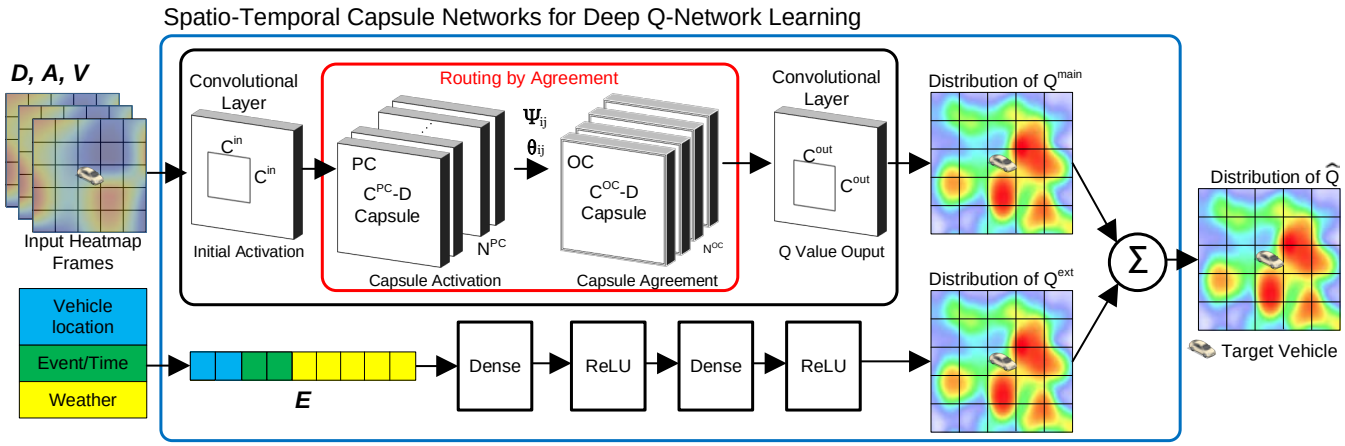


Fig. 2: Core module of spatio-temporal capsule-based Q-network.

of multiple demand surges during rush hours or rainy days, are captured with the vector representation as *instantiated entities* [16].

Fig. 2 illustrates the processing structures of the main and external state features, as detailed below.

a) **Main spatial features** $\{\mathbf{D}, \mathbf{A}, \mathbf{V}\}$: Given the input heatmap frames, $\left\{\mathbf{D}_m^{(k)}, \mathbf{A}_m^{(k)}, \mathbf{V}_m^{(k)}\right\}$, the core deep neural network in STRide captures the *spatial* relationship between states, actions and rewards. In particular, a fine-grained capsule network takes in the vehicle's 2-D state representation, and returns the estimated Q-values, denoted as $\boldsymbol{Q}^{\text{main}}$, *w.r.t.* the 2-D action space $\mathcal{U}_m$.

We further illustrate the basic structures and learning process of capsule network in STRide. Specifically, given the input heatmap $\mathbf{X}^{\text{input}} = \left\{\mathbf{D}_m^{(k)}, \mathbf{A}_m^{(k)}, \mathbf{V}_m^{(k)}\right\}$ processed from state $\boldsymbol{\mathcal{S}}_m^{(k)}$, the capsule network consists of four sequential major components, *i.e.,*

$$\begin{aligned} \mathbf{X}_1 &= \mathsf{Conv}\left(\mathbf{X}^{\text{input}}\right), \quad \mathbf{X}_2 = \mathsf{PC}\left(\mathbf{X}_1\right), \\ \mathbf{X}_3 &= \mathsf{OC}\left(\mathbf{X}_2\right), \quad \boldsymbol{Q}^{\text{main}} = \mathsf{Conv}\left(\mathbf{X}_3\right), \end{aligned} \tag{12}$$

where the first and fourth two-dimensional convolutional layers (Conv) are used for transformation between the physical heatmap frames (scalar-based) and hidden capsule layers (vector-based), *Primary Capsule* (PC) and *Output Capsule* (OC).

In our implementation, each capsule contains a structured group of neurons reshaped and regrouped from convolutional layers [28]. Each cuboid in PC/OC of Fig. 2, as a capsule, corresponds to a group of convolutional units or neurons (each neuron as a dimension is with a 9×9 kernel and a stride of 2 in our prototype). Specifically, PC is comprised of N^{PC} C^{PC}-D capsules, while OC is made of N^{OC} C^{OC}-D capsules. The input Conv layer has N^{in} $C^{\mathsf{in}} \times C^{\mathsf{in}}$ convolutional kernel filters, while the output one has N^{out} $C^{\mathsf{out}} \times C^{\mathsf{out}}$ filters.

The learning process is presented as follows. In a nutshell, the parameters of STRide's capsules, consisting of traditional *neuron weights* and additional *capsule probabilities*, are propagated and refined between the layers of PC and OC. Specifically, let θ_{ij} be the logarithm prior probability captured by a preceding capsule i in PC and its succeeding peer j in OC. A softmax function [13] is then applied upon the θ_{ij}'s, returning the coupling coefficient f_{ij} as

$$f_{ij} = \exp(\theta_{ij}) \cdot \left(\sum\nolimits_l \exp(\theta_{il})\right)^{-1}. \tag{13}$$

θ_{ij}'s capture the strengths of vehicle distributions in the map.

Similarly to the traditional neuron structure, the capsule network also has the weight coefficient across capsules i and j, denoted as Ψ_{ij}, learned through the conventional back-propagation algorithm [13]. All coefficients thus form an $N^{\mathsf{PC}} \times N^{\mathsf{OC}}$ weight matrix $\boldsymbol{\Psi}$. For each succeeding capsule j, let $\mathbf{q}_i$ be the ride prediction vector returned from a preceding peer i in PC. The propagated vector from capsules i to j, denoted as $\mathbf{q}_{j|i}$, is given by the product of neural network weights Ψ_{ij} and prediction vectors $\mathbf{q}_i$, *i.e.,* $\mathbf{q}_{j|i} = \Psi_{ij}\mathbf{q}_i$, is then fed to capsule j as a weighted average by f_{ij}'s, *i.e.,*

$$\mathbf{e}_j = \sum\nolimits_i f_{ij}\mathbf{q}_{j|i}. \tag{14}$$

A *routing-by-agreement* between capsules [28] is used to differentiate the vectors by their strengths of mutual agreement. Capsule training can then be regarded as extracting and refining the active *routes* from a preceding capsule layer (PC) to a succeeding one (OC). An active route across layers means a specific coordination strategy with certain zones, as an entity, is "memorized", while a deactivated one represents that the unimportant connections can be "forgotten". Specifically, a *squash function* $\mathbf{o}_j(\cdot)$ is applied first upon $\mathbf{e}_j$ to characterize its length [28], which is given by

$$\mathbf{o}_j(\mathbf{e}_j) \triangleq \frac{\|\mathbf{e}_j\|^2}{1+\|\mathbf{e}_j\|^2} \cdot \frac{\mathbf{e}_j}{\|\mathbf{e}_j\|}. \tag{15}$$

In other words, an increase in the vector length of the spatial ride distribution saturates the output towards one, which is identified and captured by the capsule network. The logarithm prior probabilities θ_{ij} are updated with the product of prediction $\mathbf{q}_{j|i}$ and adjustment $\mathbf{o}_j(\mathbf{e}_j)$, *i.e.,*

$$\theta_{ij} \leftarrow \theta_{ij} + \mathbf{q}_{j|i} \cdot \mathbf{o}_j(\mathbf{e}_j). \tag{16}$$

The resultant θ_{ij} is returned to Eq. (13) for another routing iteration. Via routing-by-agreement, STRide finds the vectors with higher agreement, preserving the ride correlations across zones.

b) **External temporal features** $\mathbf{E}$: $\mathbf{E}$ is processed as in Fig. 3. Due to $\mathbf{E}$'s lower dimension than vehicle and passenger distributions, we form it into a vector of the external features, and design a fully-connected neural network (Dense) to learn the coordination. Specifically, the sequential model with two layers of dense/fully-connected networks (with respective output dimensions C_1^{Den} and

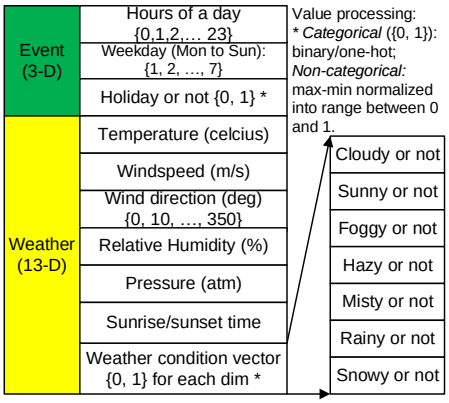


Fig. 3: External factors E.

C_2^{Den}) and subsequent ReLU activation functions [13] is given by

$$\mathbf{E}_1 = \texttt{Dense}\left(\mathbf{E}^{\text{input}}\right), \quad \mathbf{E}_2 = \texttt{ReLU}\left(\mathbf{E}_1\right), \\ \mathbf{E}_3 = \texttt{Dense}\left(\mathbf{E}_2\right), \quad \boldsymbol{Q}^{\text{ext}} = \texttt{Reshape}\left(\texttt{ReLU}(\mathbf{E}_3)\right). \tag{17}$$

where the final output $\boldsymbol{Q}^{\text{ext}}$ is reshaped from a vector back to 2-D matrix *w.r.t.* the vehicle's action space $\mathcal{U}_m$.

Finally, the estimated Q-value function *w.r.t.* each state and action is given by merging $\boldsymbol{Q}^{\text{main}}$ and $\boldsymbol{Q}^{\text{ext}}$, *i.e.,*

$$\widehat{\boldsymbol{Q}} = \boldsymbol{Q}^{\text{main}} + \boldsymbol{Q}^{\text{ext}}. \tag{18}$$

Then, in coordination STRide finds the zone with the maximum Q-value in $\widehat{\boldsymbol{Q}}$ for dispatching (subject to ϵ-greedy). The set of parameters to be trained, denoted as $\boldsymbol{\Omega}$, is hence made of those contributing to $\boldsymbol{Q}^{\text{main}}$ (including Ψ_{ij}'s and θ_{ij}'s) and those neuron weights for $\boldsymbol{Q}^{\text{ext}}$. We implement a double deep Q-network (DQN) learning mechanism [31] within STRide to train the core deep Q-network.

5 EXPERIMENTAL EVALUATION

Experimental settings: We compare STRide with the following typical algorithms: (1) *CRL*: an online learning framework [12] leveraging the conventional convolutional neural network (CNN) for coordination policy learning [25, 36]. (2) *CONT*: a conventional model-based approach, formulating a traffic control problem [7, 25] to find the dispatching and matching strategy based on the demand-supply balancing model. (3) *GD*: a heuristic vehicle dispatching without optimizing or learning the ride dynamics. MOD vehicles are greedily dispatched towards zones with the highest demand-supply imbalances [7, 27]. All evaluated schemes (the detailed parameter settings can be found in the corresponding references) use the same ride data, ETA module and learning emulator settings.

We comparatively evaluate the performance based on the following metrics: profits (platform; rides revenues minus relocation subsidies on fuel costs), idle driving time (the time when a vehicle is not occupied but still incurs the driving cost), reject rate (due to unavailability of vehicles nearby), (passenger's) waiting time, and coordination time (overhead of each online decision for MOD request matching and vehicle dispatching).

Unless stated otherwise, we use the following default parameters. In the learning emulator, we empirically set the total number of vehicles $M = 8,000$, $\alpha = 1.0$, $\beta = 0.3$, and $\epsilon = 0.1$. The total number of episodes is set to 12, while each of them is 30 hours long. Each step lasts for 30 min. A rejection happens if the distance between a request and the nearest vacant vehicle is greater than 5.0 km. For

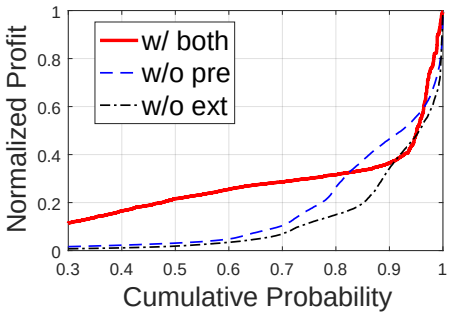


Fig. 4: Components (Uber).

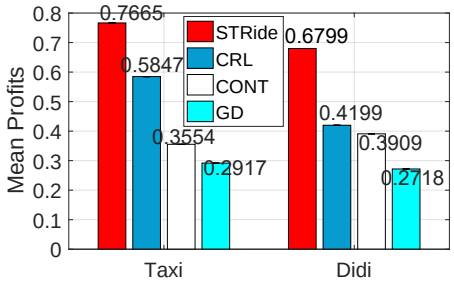


Fig. 5: Mean profit (Taxi&Didi).

Uber/Taxi, $a = 5$ and $b = 3$; for Didi, $a = 5$ and $b = 1$. The platform revenues are 20% of the ride fares, and the subsidies are 20% of the relocation costs based on studies in [10, 17, 27]. The city map is partitioned into $L_{\text{lon}} \times L_{\text{lat}} = 219 \times 219$ zones, while each vehicle is considered to move in its neighborhood of 15×15 zones ($\mathcal{U}_m$). Each vehicle's scope is set $L'_{\text{lon}} \times L'_{\text{lat}} = 23 \times 23$.

In the core Q-network, we set each layer of the spatio-temporal capsule network as follows: $\left(N^{\text{PC}}, C^{\text{PC}}\right) = (8, 8)$, $\left(N^{\text{OC}}, C^{\text{OC}}\right) = (8, 50)$, $\left(N^{\text{in}}, C^{\text{in}}\right) = (64, 9)$, $\left(N^{\text{out}}, C^{\text{out}}\right) = (64, 6)$ (each Conv has a stride of 1 and ReLU activation) for main feature components in Fig. 2; $\left(C_1^{\text{Den}}, C_2^{\text{Den}}\right) = (10, 15^2)$ for the rest components handling the 18 external features ($L_{\text{ext}} = 16$ and 2-D location in $\mathbf{E}$ as in Fig. 3). With the above settings, we obtained the following experimental results.

Experimental Results: *Profits & Efficiency*: Fig. 4 shows STRide's normalized profits without either ride preference (pre) or external factors (ext), and the complete model with both components. Introducing ride preferences helps STRide relocate vehicles across popular zones with strong connectivities, leading to more pick-ups and hence more profits. Inclusion of auxiliary factors related to MOD rides help STRide capture more intrinsic routines in the spatio-temporal ride distribution, hence achieving more profits.

Fig. 5 shows the mean profit (normalized *w.r.t.* each dataset) of all schemes. STRide is shown to achieve much higher profits than other schemes. Taking the large-scale NYC taxi dataset as an example, we show the CDFs of computation time in Fig. 6. Due to fast weight propagation across the layers, STRide achieves much better computational efficiency, which is essential for real-time coordination.

Idle driving time: Fig. 7 shows that shorter idle driving time for NYC Taxis. It is due mainly to the highly accurate Q-value approximation within STRide, capturing the spatio-temporal relations

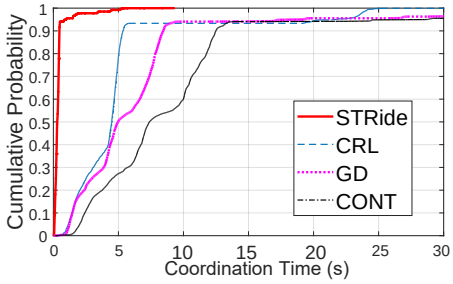


Fig. 6: Online overhead (Taxi).

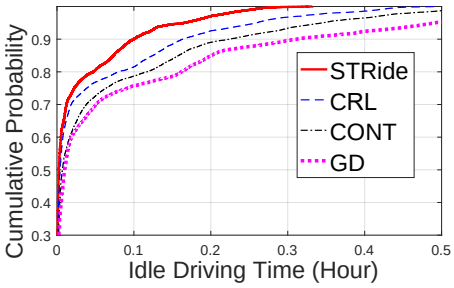


Fig. 7: Idle time (Taxi).

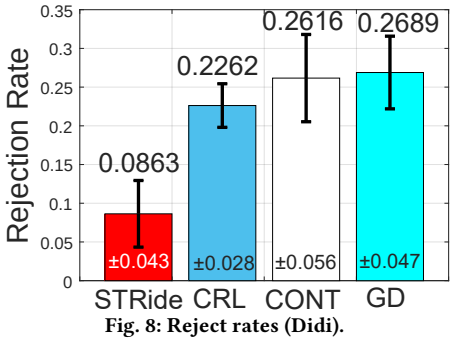


Fig. 8: Reject rates (Didi).

among states, actions and consequent rewards. Less idle driving may also lead to lower subsidies and costs from the MOD platform, thus enhancing its overall profitability.

Rejection rate: Fig. 8 shows the lower mean rejection rates of the schemes (with standard deviation) in the Didi dataset. Thanks to STRide's proactive relocation of MOD vehicles, ride requests are rejected less. On the other hand, STRide accurately learns the locations of potentially high demands, and determines proactive dispatch regions within the rejection distance threshold.

Waiting time: Fig. 9 shows the shorter passenger waiting time of STRide than other related algorithms. With more proactive dispatching, the supplies match demands in time and hence overall shorter waits are needed. This way, the MOD platforms can enhance service quality and passengers' satisfaction.

6 RELATED WORK

Numerous *optimization*-based schemes have been proposed for transportation management [7, 15, 32, 40], including control-based methodology [24], combinatorial optimization [37], and queueing theory [8]. Powered by exploding big data [23] and facilitating parallelism [9, 39], we have witnessed unprecedented advances in *learning*-based transportation management [20, 22, 26, 35]. Wen *et al.* [36] conducted preliminary studies upon learning-based MOD rebalancing. Lin *et al.* [21] studied an RL-based mechanism managing the fleet with a scalar-based neural network. Similarly, Xu *et al.* [37] explored the order dispatching, focusing on the sequential dispatch optimization problem. Oda *et al.* [25] proposed a fleet management system based on convolutional neural networks (CNNs), finding the optimal policy for relocating connected taxicabs.

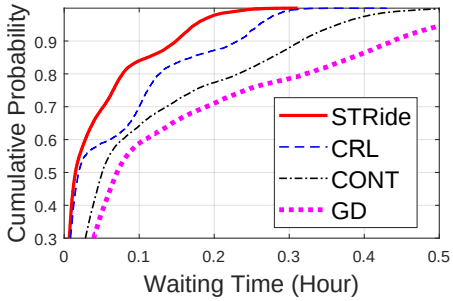


Fig. 9: Waiting time (Didi).

7 CONCLUSION & ACKNOWLEDGMENT

We have proposed STRide, a spatio-temporal reinforcement learning framework for MOD coordination. Given MOD ride data, STRide builds a learning emulator for coordination policy training. We have designed a spatio-temporal capsule network in STRide to map the states and dispatch actions towards the expected future rewards. Spatial distributions of demands and supplies, and temporal external factors like events and weather conditions, are considered together to learn the coordination policy. We have conducted extensive data analytics and experimental evaluation on three large-scale datasets. STRide is shown to outperform state-of-the-arts, with lower request rejection rates, shorter waiting times, and higher platform profitability, often by more than 30% improvement.

We would like to thank DiDi Chuxing GAIA Open Dataset Initiative for the shared ride data.

REFERENCES

[1] 2018. Global Petrol Prices. https://www.globalpetrolprices.com/gasoline_prices/.
[2] 2018. National Centers for Environmental Information, National Oceanic and Atmospheric Association (NOAA) – Data Tools: Local Climatological Data (LCD). https://www.ncdc.noaa.gov/cdo-web/datatools/lcd.
[3] 2018. Open Street Map. https://www.openstreetmap.org/.
[4] 2018. TLC Trip Record Data. http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml.
[5] 2018. Uber pickups in New York City. https://www.kaggle.com/fivethirtyeight/uber-pickups-in-new-york-city/data.
[6] 2019. Didi Chuxing Technology Co. www.didiglobal.com.
[7] Niels Agatz, Alan Erera, Martin Savelsbergh, and Xing Wang. 2012. Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research* 223, 2 (2012), 295 – 303.
[8] Siddhartha Banerjee, Ramesh Johari, and Carlos Riquelme. 2015. Pricing in Ride-Sharing Platforms: A Queueing-Theoretic Approach. In *Proc. ACM EC*. 639–639.
[9] Qingpeng Cai, Aris Filos-Ratsikas, Pingzhong Tang, and Yiwei Zhang. 2018. Reinforcement Mechanism Design for e-Commerce. In *Proc. WWW*. 1339–1348.
[10] Rachel Dovey. 2017. 5 Florida Cities Team Up to Subsidize Uber Rides. https://nextcity.org/daily/entry/five-florida-cities-subsidize-uber-rides.
[11] Zhixuan Fang, Longbo Huang, and Adam Wierman. 2017. Prices and Subsidies in the Sharing Economy. In *Proc. WWW*. 53–62.
[12] Yong Gao, Dan Jiang, and Yan Xu. 2018. Optimize taxi driving strategies based on reinforcement learning. *IJGIS* 32, 8 (2018), 1677–1696.
[13] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep Learning*. Vol. 1. MIT Press Cambridge.

[14] Jiawei Han, Jian Pei, and Micheline Kamber. 2011. *Data mining: Concepts and techniques*. Elsevier.

[15] Suining He and Kang G. Shin. 2018. (Re)Configuring Bike Station Network via Crowdsourced Information Fusion and Joint Optimization. In *Proc. ACM MobiHoc*. 1–10.

[16] Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. 2018. Matrix capsules with EM routing. In *Proc. ICLR*.

[17] Josh Horwitz. 2017. One year after the Uber-Didi merger, it's only getting harder to hail a ride in China. Respondent: https://qz.com/1045268/one-year-after-the-uber-didi-merger-its-only-getting-harder-to-hail-a-ride-in-china/ and waiting time: http://news.sina.com.cn/c/2017-07-26/doc-ifyinryq6222913.shtml.

[18] Yaguang Li, Kun Fu, Zheng Wang, Cyrus Shahabi, Jieping Ye, and Yan Liu. 2018. Multi-task Representation Learning for Travel Time Estimation. In *Proc. ACM KDD*. 1695–1704.

[19] Yaguang Li, Han Su, Ugur Demiryurek, Bolong Zheng, Tieke He, and Cyrus Shahabi. 2017. PaRE: A System for Personalized Route Guidance. In *Proc. WWW*. 637–646.

[20] Yexin Li, Yu Zheng, and Qiang Yang. 2018. Dynamic Bike Reposition: A Spatio-Temporal Reinforcement Learning Approach. In *Proc. ACM KDD*. 1724–1733.

[21] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. 2018. Efficient Large-Scale Fleet Management via Multi-Agent Deep Reinforcement Learning. In *Proc. ACM KDD*. 1774–1783.

[22] Zhidan Liu, Zhenjiang Li, Kaishun Wu, and Mo Li. 2018. Urban Traffic Prediction from Mobility Data Using Deep Learning. *IEEE Network* 32, 4 (July 2018), 40–46.

[23] Chenglin Miao, Qi Li, Lu Su, Mengdi Huai, Wenjun Jiang, and Jing Gao. 2018. Attack Under Disguise: An Intelligent Data Poisoning Attack Mechanism in Crowdsourcing. In *Proc. WWW*. 13–22.

[24] Fei Miao, Shuo Han, Shan Lin, John A Stankovic, Desheng Zhang, Sirajum Munir, Hua Huang, Tian He, and George J Pappas. 2016. Taxi Dispatch With Real-Time Sensing Data in Metropolitan Areas: A Receding Horizon Control Approach. *IEEE Trans. Automation Science and Engineering* 13, 2 (April 2016), 463–478.

[25] Takuma Oda and Carlee Joe-Wong. 2018. MOVI: A Model-Free Approach to Dynamic Fleet Management. In *Proc. IEEE INFOCOM*. 2708–2716.

[26] Ling Pan, Qingpeng Cai, Zhixuan Fang, Pingzhong Tang, and Longbo Huang. 2018. A Deep Reinforcement Learning Framework for Rebalancing Dockless Bike Sharing Systems. In *Proc. AAAI*.

[27] Lisa Rayle, Danielle Dai, Nelson Chan, Robert Cervero, and Susan Shaheen. 2016. Just a better Taxi? A survey-based comparison of taxis, transit, and ridesourcing services in San Francisco. *Transport Policy* 45 (2016), 168–178.

[28] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In *Proc. NIPS*. 3856–3866.

[29] Richard S Sutton and Andrew G Barto. 1998. *Introduction to Reinforcement Learning*. Vol. 135. MIT Press Cambridge.

[30] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proc. WWW*. 1067–1077.

[31] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep Reinforcement Learning with Double Q-Learning.. In *Proc. AAAI*, Vol. 2. 5.

[32] Erwin Walraven, Matthijs TJ Spaan, and Bram Bakker. 2016. Traffic flow optimization: A reinforcement learning approach. *Engineering Applications of Artificial Intelligence* 52 (2016), 203–212.

[33] Dong Wang, Wei Cao, Jian Li, and Jieping Ye. 2017. DeepSD: Supply-demand prediction for online car-hailing services using deep neural networks. In *Proc. IEEE ICDE*. 243–254.

[34] Zheng Wang, Kun Fu, and Jieping Ye. 2018. Learning to Estimate the Travel Time. In *Proc. ACM KDD*. 858–866.

[35] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. 2018. Intellilight: A reinforcement learning approach for intelligent traffic light control. In *Proc. ACM SIGKDD*. 2496–2505.

[36] Jian Wen, Jinhua Zhao, and Patrick Jaillet. 2017. Rebalancing shared mobility-on-demand systems: A reinforcement learning approach. In *Proc. IEEE ITSC*. 220–225.

[37] Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping Ye. 2018. Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms: A Learning and Planning Approach. In *Proc. ACM KDD*. 905–913.

[38] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction.. In *Proc. AAAI*. 1655–1661.

[39] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, Xiuwen Yi, and Tianrui Li. 2018. Predicting citywide crowd flows using deep spatio-temporal residual networks. *Artificial Intelligence* 259 (2018), 147 – 166.

[40] Huanyang Zheng and Jie Wu. 2017. Online to Offline Business: Urban Taxi Dispatching with Passenger-Driver Matching Stability. In *Proc. IEEE ICDCS*. 816–825.

An Investigation of Cyber Autonomy on Government Websites

Hsu-Chun Hsiao
National Taiwan University

Tiffany Hyun-Jin Kim
HRL Laboratories

Yu-Ming Ku
National Taiwan University

Chun-Ming Chang
National Taiwan University

Hung-Fang Chen
National Taiwan University

Yu-Jen Chen
National Taiwan University

Chun-Wen Wang
Cornell University

Wei Jeng
National Taiwan University

ABSTRACT

From a national security viewpoint, a higher degree of *cyber autonomy* is crucial to reduce the reliance on external, oftentimes untrustworthy entities, in order to achieve better resilience against adversaries. To probe into the concept of government cyber autonomy, this study examines the *external dependency of public-facing government websites* across the world's major industrialized, Group of Seven (G7) countries. Over a two-year period, we measured HTTPS adoption rates, the autonomy status of CAs, and the autonomy status of CPs on G7 government websites. We find that approximately 85% of web resources loaded by G7 government sites originate from the United States. By reviewing policy documents and surveying technicians who maintain government websites, we identify four significant forces that can influence the degree of a government's autonomy, including government mandates on HTTPS adoption, website development outsourcing, the citizens' fear of large-scale surveillance, and user confusion. Because a government websites are considered critical information infrastructures, we expect this study to raise awareness of their complex dependency, thereby reducing the risk of blindly trusting external entities when using critical government services.

CCS CONCEPTS

• **Security and privacy → Web application security**.

KEYWORDS

Cyber autonomy; government website security; HTTPS adoption

ACM Reference Format:
Hsu-Chun Hsiao, Tiffany Hyun-Jin Kim, Yu-Ming Ku, Chun-Ming Chang, Hung-Fang Chen, Yu-Jen Chen, Chun-Wen Wang, and Wei Jeng. 2019. An Investigation of Cyber Autonomy on Government Websites. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313645

1 INTRODUCTION

A central guideline to cybersecurity is to reduce reliance on *external entities*, especially for those systems under different jurisdictions or governance. From a national security standpoint, it is critical to reduce the reliance on external entities in order to achieve *cyber autonomy*[1]. Recently, cyber autonomy has gained increased attention by worldwide governments due to the revelation of state-sponsored hacktivism [8, 13], including hardware/software shipped with spyware [23, 29], compelled certificates issued for Internet interception [53], and zero-day exploits developed by rival government agencies [26].

To understand how cyber autonomy is practiced by governments worldwide, we investigated a critical facet of a government's cyber autonomy: *external dependency of public-facing government websites*. Since public-facing government websites are the front line for delivering official information and accessing government services, attacks targeting these websites can not only cause widespread panic and social instabilities, e.g., spreading inaccurate information or falsifying alerts [11, 16, 25], but also steal citizens' credentials and sensitive information [17, 19, 22, 24, 30].

In our investigation, we examined the top 100[2] government websites for each of the world's major industrialized countries, known as Group of Seven (G7), in 2017 and 2018. We considered three crucial elements for the cyber autonomy of governments: HTTPS adoption status and dependency on two external entities, namely Certificate Authorities (CAs) and Content Providers (CPs).

HTTPS adoption on government websites can protect sensitive information of their citizens. Unfortunately, our findings reveal the uneven HTTPS adoption rates across G7 countries. In 2017, the US (97%) and the UK (81%) had the highest HTTPS adoption rates, followed by Germany (57%), Canada (53%), France (46%), Japan (40%), and Italy (25%). There has been a steady increase in the adoption rate (13.7% on average) in 2018; a notable increase is by Germany, whose adoption rate has jumped to 90% in 2018. However, among these G7 government websites supporting HTTPS, 91% (in 2017) and 77% (in 2018) are incorrectly configured and thus still vulnerable to attacks. The adoption rates of HSTS, a countermeasure to HTTPS downgrade attacks, remain low (below 50%) in most G7 counties (except the US, reaching 83% in 2018).

CAs are the roots of trust on the Web. With hundreds of eligible CAs to issue certificates, an identified challenge is preventing rogue CAs from spoofing certificates or issuing certificates for phishing domains. For example, CAs can be forced by local governments to issue compelled certificates for HTTPS interception [41]. Among the G7 government websites with valid certificates, approximately

[1] We use cyber autonomy to refer to "self-controlling and free from external influence" [1] in achieving security protection.
[2] Websites are ranked based on Alexa's statistics. See §2.1.1 for details.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313645

10% are using the Extended Validation (EV) certificates, which are designed to mitigate phishing. France, Japan, and the US operate their own government root CAs (i.e., root CAs run by a government), but their root certificates are untrusted by Mozilla and Apple, causing multiple certificate errors.

CPs are servers that host web resources (e.g., images, JavaScript) for websites. When a website loads resources from untrusted servers, a variety of undesirable consequences can occur, such as including malicious content, executing malicious scripts that secretly steal credentials, mining cryptocurrencies, or sending denial-of-service traffic [44, 47]. Among the 660 websites that we crawled in 2018, 287 accessed resources from oversea countries, and each website made an average of 1.89 requests to load resources hosted on foreign servers. Among 2,160 external resources loaded by the G7 government websites, 912 (42%) were from 126 location-dependent URLs, which were served by different destination servers based on the source locations. Of the 1,248 location-independent resources, 1,063 (85%) of them originated from the US.

To understand the forces that influence the dependency level of government websites, we also reviewed policy documents and conducted a survey on technicians who maintain government websites. We highlight our findings below:

1. Government mandates help increase autonomy. The HTTPS adoption rates in the countries that have announced mandates over a year are significantly higher than the rest of the G7 countries.

2. Outsourcing undermines cyber autonomy. In our survey, 53 of 63 government websites are created and maintained by third-party contractors. While contractors are legally bound to provide services, they may be unable to respond immediately to attacks.

3. Fear of large-scale government surveillance hinders autonomy. Pinning government websites to government root certificates could eliminate the impact of rogue CAs spoofing government website certificates. However, because some countries have exploited government-issued certificates for Internet surveillance, the trustworthiness of government root CAs has been questioned, and few government root certificates are currently included in all major browsers' trust stores [14].

4. User confusion. Partial inclusion of government root CAs in the browsers' trust stores results in certificate warnings, confusing users whether the government websites in question are trustworthy. The lack of distinct government domain indication may result in users falling for attacks that could have been easily identifiable with government-specific domain names.

Contributions. This paper attempts to measure the current practice of building cyber autonomy on government websites. Over a two-year period, we measured HTTPS adoption rates and the autonomy status of CAs and CPs on G7 government websites. We also conducted a survey on technicians who maintain government websites. Our findings indicate the importance of having government mandates to enhance the overall security of government websites, as well as synchronizing autonomy-oriented government policies with corresponding entities (CAs and CPs) such that they can provide consistent, trustworthy services without errors.

Table 1: The weakness of eleven TLS attacks.

Type	Attack	Weakness
Weakness in cryptography	FREAK Attack [34]	Support RSA_EXPORT
	Logjam Attack [31]	Support DHE_EXPORT
	RC4 Attack [32, 56]	Support RC4 stream cipher
	Sweet32 Attack [35]	Support block ciphers with 64-bit block size and CBC mode
Oracle attacks	BEAST Attack [38]	Support TLS versions earlier than TLS 1.0 and CBC mode of block cipher
	CRIME Attack [39]	Support SSL/TLS compression
	DROWN Attack [33]	Use the same RSA keys in both SSL2.0 and newer versions of SSL/TLS
	POODLE Attack [46]	Support SSL3.0 without using TLS_FALLBACK_SCSV
Weakness in implementation	Heartbleed Attack [6, 40]	Support OpenSSL versions 1.0.1-1.0.1f
Weakness in protocol	ECDH-DH Cross-Protocol Attack [45]	Support ECDH and secp384r1 (ECC parameter) in server side and DH in client side
	Renegotiation Attack [51]	Support insecure renegotiation mechanism

2 HTTPS ADOPTION

HTTPS [3] runs HTTP over Transport Layer Security (TLS), which is a fundamental security protocol that enables end-to-end encryption and authentication for HTTP connections. Without HTTPS, traffic can be intercepted by any en-route adversary, as well as off-path adversaries who are capable of hijacking routes (e.g., malicious ISPs). TLS attacks can be categorized into four types based on the following weaknesses [37].

Weaknesses in cryptography: Websites supporting outdated cryptographic algorithms (e.g., RC4 and MD5) or weak key lengths (e.g., RSA_EXPORT and DHE_EXPORT) enables attackers to crack TLS.

Weaknesses in protocol design: attackers can replay, interleave, or manipulate protocol messages to deceive the client or server into an unintended state, such as deliberately downgrading the protocol version or a cipher suite.

Weaknesses in implementation: attackers can exploit vulnerabilities in the TLS software libraries (e.g., OpenSSL).

Oracle attacks: attackers can adaptively interact with a victim running TLS and derive secret information based on responses.

Table 1 summarizes 11 critical TLS attacks examined in our study. Most of the attacks can be prevented by changing TLS configuration or disabling outdated protocol versions. The last two attacks would require updating to the latest protocol version, TLS 1.3.

2.1 Methodology

In this section, we describe our approach of collecting the G7 governments' domain datasets and measuring their security.

2.1.1 URL Collection. Because there is no single source maintaining a list of up-to-date government websites, we combined multiple publicly-available datasets and systematically curated them. For each of the G7 countries, we first crawled the list of domains under the Regional/Continent Name/Country Name/Government category on Alexa [7].

Because the scope and number of websites vary by country,[3] we improved the data quality of the combination of Alexa and public datasets by heuristically restricting domain names to the corresponding country code's top-level domains (cc-TLD) or second-level domain (e.g., gov, go), assuming that the majority of the collected government domains in that country adhere to a naming convention. As we noticed that the Alexa datasets are often incomplete and

[3]The US dataset contains 1,078 federal domains, while the dataset for Japan has only 77 for "the Cabinet, Ministries and Agencies, the Diet, the Supreme Court, and other government agencies". The UK dataset contains 3,631 central and local domains; the Germany data contains 620 federal and 8,468 non-federal domains.

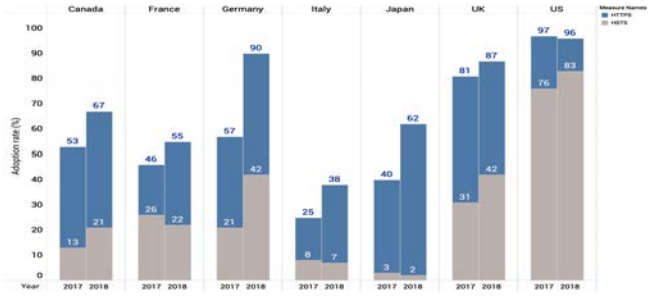


Figure 1: HTTPS and HSTS adoption rates of G7 top 100 government websites in 2017 and 2018

mixed with non-government sites (such as politicians' websites), we also searched for publicly-available datasets online. Another rationale to do so is to avoid the risk of using single evidence to generate our result [48]. We thus found official lists of government websites for the US, the UK, Canada, and Japan, but not the rest of G7. The list and final dataset we collect are available and can be accessed at the Open Science Framework project page [2].

2.1.2 Website Scanning. For each of the G7's government domains, we formed the request URL by adding four common prefixes: `http://`, `http://www`, `https://`, and `https://www`. Measurements were performed in September 2017 and August 2018. We further examined the resilience to the 11 TLS attacks (Table 1) if the domain supports HTTPS (i.e., at least one of the four requests gets redirected to an HTTPS page). Websites with certificate errors are not classified as HTTPS-supporting.

2.2 Findings

HTTPS adoption rate. Figure 1 shows the HTTPS and HSTS adoption rates of G7 government domains in 2017 and 2018. The colored bars stand for the HTTPS adoption rates, where the gray shade within each bar represents its HSTS adoption rates. The countries from the highest to the lowest HTTPS adoption rate in 2017 are as follows: the US (97%), the UK (81%), Germany (57%), Canada (53%), France (46%), Japan (40%) and Italy (25%). Both the US and UK remain equivalently high in 2018, while Germany increases from 53% to 90% within a year. The other countries show an average increase of 14.5% from 2017 to 2018, conforming to the global trend of HTTPS adoption.

HSTS adoption. HTTP Strict Transport Security (HSTS) [4] is a web security policy mechanism that allows a server to declare that it accepts only HTTPS (not HTTP) connections from browsers. HSTS is designed to prevent a man-in-the-middle (MitM) attack, which downgrades HTTPS to plain HTTP, or a *SSL stripping attack*. An HTTP request to an HSTS-enabled site will be automatically upgraded to HTTPS by supported browsers. As shown in Figure 1, the HSTS adoption rates of government websites in Japan (3%) and Italy (8%) are extremely low in 2017, and even decrease by 1% in 2018. France also has a 4% decrease in 2018. While HSTS adoption rates for the US, the UK, Germany and Canada have increased from 2017 to 2018, all except the US remain below 50%.

Resilience to critical TLS attacks. Among these G7 government websites supporting HTTPS, 91% (in 2017) and 77% (in 2018) are incorrectly configured and thus still vulnerable to attacks. Given the

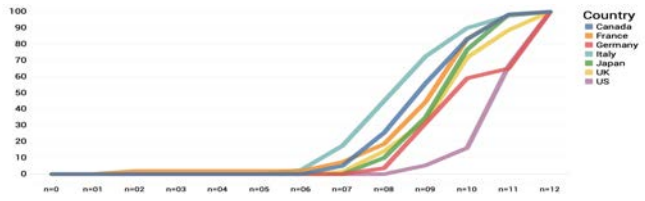


Figure 2: Resilience to 11 TLS attacks and SSL stripping in G7 in 2018

11 TLS attacks (Table 1) and the SSL stripping attack, we quantified a website's resilience level by the number of attacks against which the website can defend. Figure 2 shows the cumulative distribution of government websites resilience levels in each G7 country. More than 80% of US government websites can defend against at least 11 types of attacks. Germany makes significant progress in 2018: more than two-thirds of government websites can defend against more than 10 types, whereas it had 47% in 2017.

Autonomy implications. Increase in HTTPS and HSTS adoption rates on G7 government websites are promising, and as we discuss in §5.1, one catalyst to increase adoption rates is a government mandate. On the other hand, incorrect configurations may end up providing a false sense of security to users while leaving the door open for attackers. Hence, cross-validating implementations and keeping them up to date are crucial for secure cyber autonomy.

3 DEPENDENCY ON CA

Because HTTPS relies on digital certificates for entity authentication, trust in HTTPS is bootstrapped from certificate authorities, which issue certificates that bind a site's public key to the domain name after validating its identity. CAs are structured hierarchically for scalability: *intermediate* CAs can certify other CAs' identities, and users can determine which *root* CAs are trusted. Most users trust the root certificates in the default trust store distributed with the underlying operating systems (OS) or browsers.

Unfortunately, with hundreds of CAs eligible to issue certificates, rogue CAs can easily spoof certificates for phishing or HTTPS interception. In addition, issues in certificates and certificate chains undermine the trustworthiness of a website.

3.1 Methodology

Using the G7 governments' top 100 website lists as described in §2.1.1, we analyzed their certificates. For each website, we examined the *country* field of its root certificate to assess its dependency on foreign CAs , whether an Extended Validation (EV) certificate is used, and whether the certificate and its chain are valid.

We consider a certificate to be invalid if any of the following errors is detected: (1) a *name mismatch* error, which occurs when the hostname is inconsistent with the leaf certificate's common name; (2) an *untrusted root CA* error, which occurs when the root certificate's issuer is not in the trust store, potentially resulting in untrustworthy self-signed certificates or certificates signed by untrusted entities; (3) an *invalid time* error, which occurs when the current time is before or after the valid certificate lifetime; or (4) a *certificate issuer error*, which occurs when a leaf certificate's issuer is inconsistent with its parent's subject field.

3.2 Findings

Government root CAs. We analyzed the root certificates included in the trust stores of the major browsers and OS. 168, 359, and 149 trusted root certificates are pre-installed in the latest trust store of Apple macOS 10.13 (High Sierra) [9], Microsoft Windows [20], and Mozilla [21], respectively. Microsoft Windows's trust store contains 53 government root certificates[4] from 26 countries, including France, Japan, and the US. The Mozilla's trust store contains eight government root certificates from Hong Kong, the Netherlands, Spain, Taiwan, and Turkey. Apple macOS's trust store contains only two government root certificates, from Taiwan and Finland. As some government root certificates are not included in all trust stores, using these certificates will trigger warnings on some browsers. In the 2017 dataset, 11 Japanese government websites relied on ApplicationCA2 Root, a Japanese government root CA that is untrusted by Mozilla and Apple macOS.

Certificate type. Table 2 summarizes the number of websites with valid certificates, domestic root CAs, and EV certificates in 2018. The percentages in the last two columns are computed over websites with valid certificates. Among G7 government websites with valid certificates, about 10% are using Extended Validation (EV) certificates, which are designed to mitigate phishing. Interestingly, despite its great share of the CA market, Let's Encrypt (LE)[5] certificates are rarely used by government websites. As of November 2018, LE issues more than 60% of certificates on the Internet [10]. In our 2018 dataset, there are 8 from Germany and 2 from Italy, and each of the remaining G7 countries has only one government website using LE certificates.

Root CA dependency. Governments can reduce the root of trust and increase legal accountability by restricting CAs to those registered in the same country (e.g., via DNS-based Authentication of Named Entities). Hence, we examined the reliance of the government websites on domestic CAs. A root certificate's country is determined by the country field in its certificate information.

Figure 3 uses a circular layout for visualizing the relationship between the G7 countries' government websites and their root CAs' countries of origin as of August 2018. For instance, the clockwise curve from Country A to Country B represents higher dependency of A's government websites on B's root CAs. A larger circle area of a country indicates that more websites rely on the CAs in the country. Among all G7 government sites we analyzed, the most popular root CAs and their registered countries are GlobalSign Root CA in Belgium (accounting for 16% of 496 G7 sites with valid certificates), AddTrust External CA Root in Sweden (6%), DigiCert

[4]We consider that a certificate is issued by a government if it contains "government", "gov", or "federal" in the description.
[5]Let's Encrypt is a non-profit certificate authority that issues free certificates.

Table 2: Certificate analysis and Domestic CA of top 100 government sites in G7, 2018

Country	w/ HTTPS protocol	Valid Cert	Domestic CA	EV Cert
Canada	70	67	0 (0%)	7 (10.4%)
France	65	55	21 (38.1%)	6 (10.9%)
Germany	92	90	44 (48.8%)	6 (6.6%)
Italy	44	38	9 (23.7%)	4 (10.5%)
Japan	65	62	27 (43.5%)	7 (11.3%)
UK	91	87	9 (10.3%)	10 (11.5%)
US	96	96	89 (92.7%)	10 (10.4%)
Total/Average	523/74.7	495/70.7	200/28.6	51/7.3

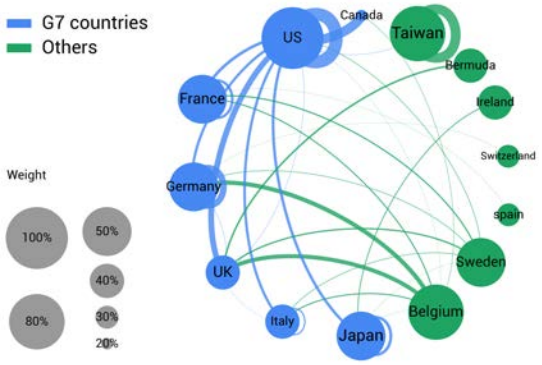


Figure 3: Dependency on root CAs by countries.

Global Root CA in the US (34%), and Entrust Root Certification Authority in the US (11%). Among G7, all but the US have more than 50% of government websites using root CAs registered in foreign countries in 2018. Ninety out of 96 US government sites adopt US-based root CAs, including one using Let's Encrypt and three using Entrust.net CA, the third most popular CA in G7.

The validity of certificates and certificate chains. Table 3 summarizes common misconfigurations. In 2017, 45% of France's Top 100 government sites suffered from name mismatch errors. Several certificates used `www.snakeoil.dom` or `ssl3.ovh.net` as their common name, which are used as examples in online tutorials. Japan has the highest percentage of untrusted root certificates: 11 (out of 12) use the same root certificate, ApplicationCA2 Root, a Japanese government root CA not trusted by all major browsers.

Autonomy implications. While governments are moving towards CA autonomy by restricting the usage of external CAs and promoting government root CAs, we observed inconsistencies across the trust stores of the browsers and OS. Although different browser and OS vendors have different policies, such inconsistencies imply some concerns about including the government root CAs to the trust store. Hence, further analysis must be conducted to resolve the conflicts, which may require cooperation among the browser and OS vendors. As we discuss in §5.3, skepticism about government surveillance activities on its citizens can be addressed by restricting the government root CAs' usage to government websites only, or by delegating the management of government root CAs to non-profit organizations within the same jurisdiction.

4 DEPENDENCY ON CP

Content Providers (CP) provide web resources (e.g., images and JavaScript) that can be included on websites. However, when a site loads resources from untrusted servers, it may include malicious content or even execute malicious scripts [44, 47].

Table 3: G7 certificate scanning result for the indicated issues. A lower percentage is better in terms of security.

Country	Domains w/ cert.		Name mismatch		Issuer error		Untrusted root		Invalid time	
	2017	2018	2017	2018	2017	2018	2017	2018	2017	2018
Canada	53	70	2%	4%	0%	0%	4%	0%	0%	0%
France	46	65	45%	15%	9%	0%	13%	6%	6%	3%
Germany	57	92	10%	2%	0%	0%	0%	0%	0%	0%
Italy	25	44	25%	13%	0%	0%	15%	0%	12%	4%
Japan	40	65	0%	4%	0%	0%	28%	28%	1%	1%
UK	81	91	2%	3%	0%	0%	1%	0%	0%	0%
US	97	96	0%	0%	0%	0%	0%	0%	0%	0%

Table 4: # websites loading resources from oversea servers.

country	ca	fr	de	it	jp	UK	US	total/Average
CP_ASIA	48	51	26	46	30	55	31	287/41
CP_NA	52	49	28	55	29	61	12	287/40.9

4.1 Methodology

We loaded each website from the top 100 government website dataset for each country using Google Chromium Version 68.0.3440.106 for 20 seconds, and extracted the request URLs logged by the browser's network panel. As some content may be served by nearby servers, we loaded these websites from servers located in Asia and North America in October 2018 to capture such location-dependent behaviors. We fetched the IP from the remote address field in each request header, and used pygeoip GeoIP API along with the Maxmind geolocation database[6] to identify its country of residence.

Since we are interested in external CPs that may provide untrusted resources, we removed URLs with the same domain name (and thus are considered local), and focused on URLs that retrieve the most common external resources such as image, JavaScript, and JSON files [43] from foreign servers, as these resources can be used to exploit the website [44, 47]. Our CP_ASIA and CP_NA datasets contain 2,160 and 2,185 URLs for crawling from Asia and North America, respectively. Among these total 4,345 URLs, 1,960 (45.2%), 2,151 (49.5%), and 231 (5.3%) are for JavaScript, image, and JSON files, respectively.

We say that a request URL is *location-aware* if it is routed to different destination servers when it is loaded from different source locations. A location-aware URL could have a negative impact on cyber autonomy because the user cannot be assured that the resource is always loaded from domestic servers. We identified 126 location-aware URLs that appear in both the CP_ASIA and CP_NA datasets, and in total 912 (of out 2,160; 42%) external resources were loaded from these location-aware URLs. Excluding the location-aware ones, the number of external JavaScript, image, and JSON files are 1,292 (51%), 1182 (46%), and 74 (3%), respectively.

4.2 Findings

Websites loading resources from foreign servers. Among the 660 websites that were successfully crawled, each website made an average of 1.89 requests to load resources hosted on foreign servers, with a median of 1. Table 4 summarizes the percentage of websites, among the top 100 government websites in each G7 country, that load resources from foreign servers. All G7 countries have a non-negligible degree of dependency, from 8% (US) to 23% (Japan, Germany) and more than 40% (the rest of G7), when users access government websites from Asia. The percentage reduces drastically for US government websites when accessed from North America. However, most G7 countries tend to have a high degree of dependency regardless of access location.

We found several government websites hosted in foreign countries. For example, the National Film Board of Canada (www.nfb.ca) is hosted in the US and loads resources from US-based servers.

CP dependency by country. The three most common destinations of external resources are the US (1,173), the Philippines (269), and Taiwan (225) in the CP_ASIA dataset, with a total of 2,160 external resources, and the US (1,915), Ireland (120), and the Netherlands (72) in the CP_NA dataset. These results align with common beliefs that the US hosts a large portion of web resources and that Content Distribution Network (CDN) servers are prevalent. Figure 4 shows the CP dependency by country, utilizing the similar circular layout as Figure 3. The orange lines represent the location-aware resources, whose IP addresses were different in the two datasets collected from different locations. Of the 1,248 location-independent resources, 1,063 (85%) of them originate from the United States.

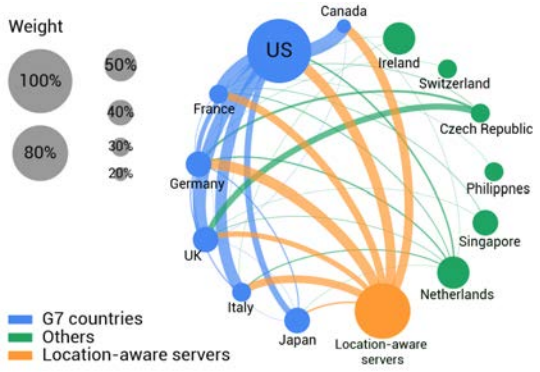


Figure 4: Dependency on external resources by countries.

Autonomy implications. In general, we observed greater reliance on US content providers for all G7 governments. Regarding location-aware URLs, a majority of the government websites may be loading content from foreign CPs, possibly to tolerate the latency. However, proving better usability with short delays have negative effects towards having cyber autonomy. Further investigation on the trade-offs between latency and autonomy may be needed to analyze the benefits of risking security for enhanced time delays.

5 DISCUSSION

5.1 Government Policies

Government policies play an important role in cyber autonomy. We found that at least four of the G7 countries have mandates that promote HTTPS adoption on government websites, and some have policies limiting the use of weak cryptographic algorithms and parameters for HTTPS and certificates. However, we did not find policies that restrict access to external resources. Considering the issuance or enforcement date of their security policies, we found that countries with established HTTPS adoption policies indeed exhibit a significantly higher level of security than others in terms of HTTPS deployment. The adoption rates of the US and UK were the highest in 2017, and Germany has improved in 2018. We briefly describe the findings on G7's HTTPS policies below.

The US. On June 8, 2015, the White House Office of Management and Budget memorandum M-15-13 was released to "Require Secure Connections across Federal Websites and Web Services" [5]. The memorandum provides a timeframe for compliance, requesting that all federal websites and services upgrade to HTTPS-only (with HSTS) by the end of 2016. Note that this memorandum applies to federal sites only, not to the state or local sites. On October 16, 2017, the Department of Homeland Security issued Binding Operational Directive 18-01, requiring federal agencies to remove

[6]The Maxmind database has 96-98% accuracy in the country level [49] and has been used to validate the IP geolocation information released by Regional Internet Registries(RIR) [57].

support for outdated cryptographic algorithms and protocols. There is no restriction on acceptable CAs for federal domains.

The UK. The Digital Service Standard and Service Manual in the UK requires all Government Digital Service (GDS) websites to be accessible only through HTTPS starting October 1, 2016 and provides guidelines to configure HTTPS and HSTS [18].

Germany. In January 2018, the Federal Office for Information Security (BSI) issued BSI-TR-02102-2 (TLS), a technical guideline providing recommendations for TLS usage, and stating that the "Federal Administration is obliged to encrypt all data traffic over unprotected networks using SSL/TLS" [27].

Japan. The Japanese government announced a standard security guideline [15] in July 2018, which explicitly states that servers should install legitimate certificates for supporting TLS protocol.

EU. Enforced on May 25 2018, the General Data Protection Regulation (GDPR) is a new regulation that aims to enhance personal privacy on the Internet. Adopting HTTPS is considered a baseline security measure for protecting the PII during transmission.

5.2 Outsourcing

To understand the factors that influence the dependency level of government websites, we conducted two online surveys aimed at web technicians who maintain government websites in Taiwan[7] in September 2017 and September 2018. To recruit more survey participants, we contacted 161 government agencies (not limited to the top 100). Among 73 websites with valid responses, only three (4%) are self-developed by the institutions themselves; the rest have been outsourced to third-party companies. Outsourcing the website to third-party contractors is reasonable for small government agencies, which often lack sufficient resources and budgets to hire dedicated staff specialized in network security and web development. Unfortunately, outsourcing development to third-party contractors may introduce an additional barrier in communicating required security features, thereby further complicating the security configuration and maintenance of government websites. When the majority of the websites are developed by third-party contractors, technical specification plays an important role in defining the requirements in website development. However, 8 of the 70 websites did not put HTTPS in their specification.

Our survey results also indicate that government website administrators consciously consider whether a relevant mandate exists when prioritizing their tasks. Six out of 23 participants (26%) reported that the main reason for not fully upgrading to HTTPS is beause there is "no mandate from the supervising agencies", possibly reducing the priority of HTTPS adoption given other tasks. This echoes our observation that government policies can expedite the deployment of security measures.

5.3 Fear of large-scale Surveillance

Although pinning government websites to government root CAs could eliminate reliance on hundreds of CAs in the browser's default trust store, governments root CAs may be abused for large-scale surveillance [12, 14, 28]. Due to this privacy concern, several government root certificates are excluded from trust stores. For example, the government root certificates of France, Japan, and the US are included in Microsoft Windows' trust store, but not in Apple macOS or Mozilla. However, such partial inclusion may confuse users who see inconsistent certificate warnings. Another complication is that certificate pinning mechanisms are not widely supported on the web. HTTP Public Key Pinning (HPKP) is rarely used by websites and has been deprecated by the latest version of Chrome. DNS-based Authentication of Named Entities (DANE) has not been fully supported by major browsers either.

[7]The HTTPS adoption rate among the top 100 gov websites in Taiwan is 87% in 2018, and among the 68 websites with valid certificates, 63 (92.6%) are using the domestic CAs, and 60 (88%) using the government root CAs. As for foreign resources, there were only one URL to Germany and four to US loaded by the top 100 government websites.

5.4 Usability Issues

For cyber autonomy, it is crucial that the citizens can distinguish government websites from potentially spoofed websites. Some countries use government-specific (top level) domains, such as .gov for the US and UK, and .go.jp for Japan; hence, users can easily recognize if the website in question belongs to the government of the representing country. Unfortunately, without such a domain requirement, users may fall for attacks that might have been easily identifiable with government-specific domain policies.

Another usability issue arises when governments support their own root CAs for cyber autonomy, but the CAs are not recognized as trustworthy by the browsers or OS. As reported in §3.2, Japan operates its government root CAs, GPKI ApplicationCA2 Root and Japanese Government ApplicationCA, but they are only included in Microsoft Windows' trust store, not Apple macOS or Mozilla. Consequently, users experience frequent certificate errors when they use Safari or Mozilla to access a Japanese government website whose certificate is endorsed by either of these two CAs. Such frequent errors may result in a user (1) ignoring certificate warnings [54], or (2) losing trust in the government websites.

6 RELATED WORK

To the best of our knowledge, this is the first study examining the external dependency of public-facing government websites; prior work focuses on one aspect of dependency of popular websites.

A large body of research investigates the current practices of HTTPS. These studies usually scan the IPv4 address space or popular domains, or analyze collected datasets to uncover statistical facts and correlations [36, 50]. Felt et al. [50] observed an increase of HTTPS support from 2016 to 2017. In addition to HTTPS adoption rates, we also investigated websites vulnerable to known TLS attacks and discuss the effectiveness of government policies on prompting HTTPS adoption. To identify HTTPS deployment challenges encountered by security experts, Krombholz et al. [42] performed a lab experiment with 28 technically-competent students and interviewed seven security auditors. Our survey targeted government website operators instead of developers, and the results suggest infrastructural and administrative challenges that government websites may encounter when using HTTPS.

Fadai et al. [41] analyzed the trust stores of multiple browsers and operating systems and identified CAs owned by companies and governmental institutions that might not be trustworthy. Vallina-Rodriguez et al. [55] compared the trust stores on Android devices

with different OS versions and manufacturers. We focused on analyzing government root certificates and the certificates used by government websites.

Simeonovski et al. [52] proposed to model web dependencies using a property graph, and applied the model to analyze Alexa's Top 100k domains. Kumar et al. [43] also investigated the dependencies on external services of Alexa's Top Million domains. They found that the median of external resources loaded by websites is 23, which is much higher than for government websites. Such differences suggest the importance of performing in-depth studies based on website types, because each type of website may have unique challenges and incentives for security deployment.

7 CONCLUSIONS AND FUTURE WORK

This paper attempts to measure the current practice of building cyber autonomy on government websites. Our findings indicate the importance of having government mandates, as well as synchronizing autonomy-oriented government policies with corresponding entities. An interesting future direction is to investigate dependency on software (e.g., web frameworks and libraries), hardware (e.g., servers and routers), and network (e.g., routing paths), as well as to assess a government's cyber autonomy from other complementary perspectives. We hope that our findings serve as a stepping stone towards building cyber autonomy.

ACKNOWLEDGMENT

This work was financially supported by Ministry of Science and Technology in Taiwan, under Grant MOST108-2636-E-002-005, MOST108-2636-H-002-002, and MOST107-3017-F-002-004, as well as the Grant (#107L900204) by the Ministry of Education (MOE) in Taiwan.

REFERENCES

[1] [n. d.]. autonomy. In *The Oxford Dictionary*. https://en.oxforddictionaries.com/definition/autonomy
[2] [n. d.]. Data collected in this study. https://osf.io/r3qke/?view_only=36da7880978646739489469e2f4593e8.
[3] Accessed: 2017-10-30. HTTP Over TLS. https://tools.ietf.org/html/rfc2818.
[4] Accessed: 2017-10-30. HTTP Strict Transport Security (HSTS). https://tools.ietf.org/html/rfc6797.
[5] Accessed: 2017-10-30. The HTTPS-Only Standard.
[6] Accessed: 2017-10-30. The Heartbleed Bug. http://heartbleed.com/.
[7] Accessed: 2018-11-05. Alexa. https://www.alexa.com.
[8] Accessed: 2018-11-05. An update on state-sponsored activity. https://www.blog.google/technology/safety-security/update-state-sponsored-activity/.
[9] Accessed: 2018-11-05. ApplemacOS 10.13 (High Sierra). https://support.apple.com/en-us/HT208127.
[10] Accessed: 2018-11-05. As of November 2018, it issues more than 60% ofcertificates on the Internet. https://censys.io/certificates/report?q=tags%3Atrusted&field=parsed.issuer.organization.raw&max_buckets=10.
[11] Accessed: 2018-11-05. County Election Websites Can Be Easily Spoofed to Spread Misinformation. https://www.darkreading.com/vulnerabilities-threats/county-election-websites-can-be-easily-spoofed-to-spread-misinformation/d/d-id/1333132.
[12] Accessed: 2018-11-05. French gov used fake Google certificate to read its workers' traffic. https://www.theregister.co.uk/2013/12/10/french_gov_dodgy_slcert_reprimand/.
[13] Accessed: 2018-11-05. Germany, seeking independence from U.S., pushes cyber security research. https://www.reuters.com/article/us-germany-cyber/germany-seeking-independence-from-u-s-pushes-cyber-security-research-idUSKCN1LE1FX.
[14] Accessed: 2018-11-05. Government Certification Authorities. https://wiki.mozilla.org/CA:GovernmentCAs#Concerns_about_Government_CAs.
[15] Accessed: 2018-11-05. Guidelines for formulating measures standards by government agencies (Heisei 30th edition) . https://www.nisc.go.jp/active/general/pdf/guide30.pdf.
[16] Accessed: 2018-11-05. Hawaii missile alert: False alarm sparks panic in US state. https://www.bbc.com/news/world-us-canada-42677604.
[17] Accessed: 2018-11-05. How to spot a fake government website. https://www.gov.sg/news/content/how-to-spot-a-fake-government-website.
[18] Accessed: 2018-11-05. Managing service domains. https://www.gov.uk/service-manual/technology/managing-domain-names.
[19] Accessed: 2018-11-05. Microsoft says it has found a Russian operation targeting U.S. political institutions. https://www.washingtonpost.com/business/economy/microsoft-says-it-has-found-a-russian-operation-targeting-us-political-institutions/2018/08/20/52273e14-a4d2-11e8-97ce-cc9042272f07_story.html?noredirect=on&utm_term=.db741731087b.
[20] Accessed: 2018-11-05. Microsoft Windows (as of January 30, 2018). https://gallery.technet.microsoft.com/Trusted-Root-Certificate-70150b50.
[21] Accessed: 2018-11-05. Mozilla (as of October 19, 2017). https://wiki.mozilla.org/CA/Included_Certificates.
[22] Accessed: 2018-11-05. myGov scam tricking victims into handing over bank details through cloned website. https://www.abc.net.au/news/2018-07-05/mygov-scam-clones-government-website-medicare-phishing-email/9942908.
[23] Accessed: 2018-11-05. NSA reportedly intercepting laptops purchased online to install spy malware. https://www.theverge.com/2013/12/29/5253226/nsa-cia-fbi-laptop-usb-plant-spy.
[24] Accessed: 2018-11-05. People in France warned over scam versions of government websites. https://www.thelocal.fr/20180917/people-in-france-warned-over-scam-versions-of-government-websites.
[25] Accessed: 2018-11-05. Syrian hackers claim AP hack that tipped stock market by $136 billion. Is it terrorism? https://www.washingtonpost.com/news/worldviews/wp/2013/04/23/syrian-hackers-claim-ap-hack-that-tipped-stock-market-by-136-billion-is-it-terrorism/?noredirect=on&utm_term=.266ad973fec7.
[26] Accessed: 2018-11-05. THE LEAKED NSA SPY TOOL THAT HACKED THE WORLD. https://www.wired.com/story/eternalblue-leaked-nsa-spy-tool-hacked-world/.
[27] Accessed: 2018-11-05. The State of IT Security in Germany 2017 . https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Securitysituation/IT-Security-Situation-in-Germany-2017.pdf.
[28] Accessed: 2018-11-05. Turkish government agency spoofed Google certificate "accidentally". https://arstechnica.com/information-technology/2013/01/turkish-government-agency-spoofed-google-certificate-accidentally/.
[29] Accessed: 2018-11-05. U.S. ban on sales to China's ZTE opens fresh front as tensions escalate. https://www.reuters.com/article/us-china-zte/u-s-ban-on-sales-to-chinas-zte-opens-fresh-front-as-tensions-escalate-idUSKBN1HN1P1.
[30] Accessed: 2018-11-05. Washington State system hacked, data of thousands at risk. https://www.reuters.com/article/us-usa-hack-washingtonstate/washington-state-system-hacked-data-of-thousands-at-risk-idUSBRE9480YY20130509.
[31] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, et al. 2015. Imperfect forward secrecy: How Diffie-Hellman fails in practice. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*.
[32] Nadhem J AlFardan, Daniel J Bernstein, Kenneth G Paterson, Bertram Poettering, and Jacob CN Schuldt. 2013. On the Security of RC4 in TLS.. In *Proceedings of USENIX Security Symposium*.
[33] Nimrod Aviram, Sebastian Schinzel, Juraj Somorovsky, Nadia Heninger, Maik Dankel, Jens Steube, Luke Valenta, David Adrian, J Alex Halderman, Viktor Dukhovni, et al. 2016. DROWN: Breaking TLS Using SSLv2. In *Proceedings of USENIX Security Symposium*.
[34] Benjamin Beurdouche, Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cédric Fournet, Markulf Kohlweiss, Alfredo Pironti, Pierre-Yves Strub, and Jean Karim Zinzindohoue. 2015. A messy state of the union: Taming the composite state machines of TLS. In *Proceedings of IEEE Symposium on Security and Privacy (SP)*.
[35] Karthikeyan Bhargavan and Gaëtan Leurent. 2016. On the practical (in-) security of 64-bit block ciphers: Collision attacks on HTTP over TLS and OpenVPN. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*.
[36] Li Chang, Hsu-Chun Hsiao, Wei Jeng, Tiffany Hyun-Jin Kim, and Wei-Hsi Lin. 2017. Security Implications of Redirection Trail in Popular Websites Worldwide. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*.
[37] Jeremy Clark and Paul C van Oorschot. 2013. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In *Proceedings of IEEE Symposium on Security and Privacy (SP)*.
[38] Thai Duong and Juliano Rizzo. 2011. Here come the ⊕ ninjas. *Unpublished manuscript* (2011).
[39] Thai Duong and Juliano Rizzo. 2012. The CRIME attack. In *Presentation at ekoparty Security Conference*.
[40] Zakir Durumeric, James Kasten, David Adrian, J Alex Halderman, Michael Bailey, Frank Li, Nicolas Weaver, Johanna Amann, Jethro Beekman, Mathias Payer, et al. 2014. The matter of heartbleed. In *Proceedings of ACM Internet Measurement Conference (IMC)*.

[41] Tariq Fadai, Sebastian Schrittwieser, Peter Kieseberg, and Martin Mulazzani. 2015. Trust me, I'm a Root CA! Analyzing SSL Root CAs in Modern Browsers and Operating Systems. In *International Conference on Availability, Reliability and Security (ARES)*.

[42] Katharina Krombholz, Wilfried Mayer, Martin Schmiedecker, Edgar Weippl, Wilfried Mayer, Martin Schmiedecker, and Edgar Weippl. 2017. "I Have No Idea What I'm Doing" - On the Usability of Deploying HTTPS. In *Proceedings of USENIX Security Symposium*.

[43] Deepak Kumar, Zane Ma, Ariana Mirian, Joshua Mason, J Alex Halderman, and Michael Bailey. 2017. Security Challenges in an Increasingly Tangled Web. In *In Proceedings of the World Wide Web Conference (WWW)*.

[44] Bill Marczak, Nicholas Weaver, Jakub Dalek, Roya Ensafi, David Fifield, Sarah McKune, Arn Rey, John Scott-Railton, Ron Deibert, and Vern Paxson. 2015. An analysis of china's "great cannon". *FOCI. USENIX* (2015), 37.

[45] Nikos Mavrogiannopoulos, Frederik Vercauteren, Vesselin Velichkov, and Bart Preneel. 2012. A cross-protocol attack on the TLS protocol. In *Proceedings of ACM SIGSAC conference on Computer and communications security*.

[46] Bodo Möller, Thai Duong, and Krzysztof Kotowicz. 2014. *This POODLE Bites: Exploiting The SSL 3.0 Fallback*. Technical Report. Google.

[47] Nick Nikiforakis, Luca Invernizzi, Alexandros Kapravelos, Steven Van Acker, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2012. You are what you include: large-scale evaluation of remote javascript inclusions. In *Proceedings of the 2012 ACM conference on Computer and communications security*.

[48] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhooby, Maciej Korczynskiz, and Wouter Joosen. 2019. TRANCO: A Research-Oriented Top Sites Ranking Hardened Against Manipulation.

[49] Ingmar Poese, Steve Uhlig, Mohamed Ali Kaafar, Benoit Donnet, and Bamba Gueye. 2011. IP Geolocation Databases: Unreliable?. In *Proceedings of ACM SIGCOMM*.

[50] Adrienne Porter Felt, Richard Barnes, April King, Chris Palmer, Chris Bentzel, and Parisa Tabriz. 2017. Measuring HTTPS Adoption on the Web. In *Proceedings of USENIX Security Symposium*.

[51] Marsh Ray and Steve Dispensa. 2009. Renegotiating TLS. IETF-76.

[52] Milivoj Simeonovski, Giancarlo Pellegrino, Christian Rossow, and Michael Backes. 2017. Who controls the internet?: Analyzing global threats using property graph traversals. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee.

[53] Christopher Soghoian and Sid Stamm. 2011. Certified lies: Detecting and defeating government interception attacks against SSL (short paper). In *International Conference on Financial Cryptography and Data Security*.

[54] Joshua Sunshine, Serge Egelman, Hazim Almuhimedi, Neha Atri, and Lorrie Faith Cranor. 2009. Crying Wolf: An Empirical Study of SSL Warning Effectiveness. In *Proceedings of the 18th USENIX Security Symposium*.

[55] Narseo Vallina-Rodriguez, Johanna Amann, Christian Kreibich, Nicholas Weaver, and Vern Paxson. 2014. A Tangled Mass: The Android Root Certificate Stores. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*.

[56] Mathy Vanhoef and Frank Piessens. 2015. All Your Biases Belong to Us: Breaking RC4 in WPA-TKIP and TLS.. In *Proceedings of USENIX Security Symposium*.

[57] Sebastian Zander. 2012. *How Accurate is IP Geolocation Based on IP Allocation Data?* Technical Report. Swinburne University of Technology.

Transfer Meets Hybrid: A Synthetic Approach for Cross-Domain Collaborative Filtering with Text

Guangneng Hu
Department of Computer Science and Engineering, Hong Kong University of Science and Technology
njuhgn@gmail.com

Yu Zhang
Department of Computer Science and Engineering, Hong Kong University of Science and Technology
yu.zhang.ust@gmail.com

Qiang Yang
Department of Computer Science and Engineering, Hong Kong University of Science and Technology
qyang@cse.ust.hk

ABSTRACT

Collaborative Filtering (CF) is the key technique for recommender systems. CF exploits user-item behavior interactions (e.g., clicks) only and hence suffers from the data sparsity issue. One research thread is to integrate auxiliary information such as product reviews and news titles, leading to hybrid filtering methods. Another thread is to transfer knowledge from source domains such as improving the movie recommendation with the knowledge from the book domain, leading to transfer learning methods. In real-world applications, a user registers for multiple services across websites. Thus it motivates us to exploit both auxiliary and source information for recommendation in this paper. To achieve this, we propose a Transfer Meeting Hybrid (TMH) model for cross-domain recommendation with unstructured text. The proposed TMH model attentively extracts useful content from unstructured text via a memory network and selectively transfers knowledge from a source domain via a transfer network. On two real-world datasets, TMH shows better performance in terms of three ranking metrics by comparing with various baselines. We conduct thorough analyses to understand how the text content and transferred knowledge help the proposed model.

CCS CONCEPTS

• **Information systems** **Personalization**.

KEYWORDS

Recommender Systems; Collaborative Filtering; Deep Learning

ACM Reference Format:
Guangneng Hu, Yu Zhang, and Qiang Yang. 2019. Transfer Meets Hybrid: A Synthetic Approach for Cross-Domain Collaborative Filtering with Text. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313543

1 INTRODUCTION

Recommender systems are widely used in various domains and e-commerce platforms, such as recommending products to buy at Amazon and videos to watch on Youtube. Collaborative Filtering (CF) is an effective approach based on an intuition that if users rated items similarly in the past then they are likely to rate items similarly in the future. Matrix Factorization (MF) techniques are its main cornerstone [26, 35] since they can learn latent factors for users and items. Recently, neural networks like multilayer perceptrons (MLP) are used to learn non-linear interaction functions from data [10, 17]. Both MF and neural CF suffer from the data sparsity and cold-start issues.

One solution is to integrate CF with the content information, leading to hybrid methods. Items are usually associated with unstructured text like the news articles and product reviews. Additional information alleviates the data sparsity issue and is essential for recommendation beyond user-item interactions. For application domains like recommending research papers and news articles, the unstructured text associated with an item is its text content [1, 47]. Other domains like recommending products, the unstructured text associated with the item is its user reviews which justify the rating behavior of consumers [21, 33, 56]. Recently, neural networks have been proposed to exploit the item content. For example, memory networks [43] are used to model item reviews [19], or to model a user's neighbors who rated the same items with this user [11].

Another solution is to transfer the knowledge from relevant domains and the cross-domain recommendation techniques address such problems [3, 27, 37]. In real-world applications, a user typically registers multiple service systems to acquire different information need. For example, a user installs applications in an app store and reads news from another website. It brings us an opportunity to improve the recommendation performance in the target service (or all services) by learning from across domains. In the above example, we can represent the app installation feedback using a binary matrix whose entries indicate whether a user has installed an app. Similarly, we use another binary matrix to indicate whether a user has read a news article. Typically these two matrices are highly sparse, and it is beneficial to learn them simultaneously. This idea is sharpened into the Collective Matrix Factorization (CMF) approach [42] which jointly factorizes these two matrices by sharing the user latent factors. It combines CF on a target domain and another CF on an auxiliary domain, enabling knowledge transfer [36, 54]. In terms of neural networks, given two activation maps from two tasks, cross-stitch network [34] and its sparse variant [22] learn linear combinations of both the input activations and feed these combinations as input to the successive layers, and hence enabling the knowledge transfer between two domains.

These two threads motivate us to exploit information from both the content and cross-domain information for recommendation in this paper. To capture text content and to transfer cross-domain

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313543

knowledge, we propose a novel neural model, TMH, for cross-domain recommendation with unstructured text. TMH can not only attentively extract useful content via a memory network but also selectively transfer knowledge across domains by a novel transfer network. A shared layer of feature interactions is stacked on the top to couple high-level representations learned from both networks. On real-world datasets, TMH shows the better performance in various settings. We conduct thorough analyses to understand how the content and transferred knowledge help the proposed model.

Our contributions are summarized as follows:

- The proposed model is a novel deep model that transfers cross-domain knowledge for recommendation with unstructured text by using an attention based neural network. (Sec. 4)
- We interpret the memory networks to attentively exploit the text content to match word semantics with user preferences.
- The transfer network can selectively transfer source items with the guidance of target user-item interactions by the attentive weights.
- Our model alleviates cold-user and cold-item start issues, and outperforms various baselines on real-world datasets. (Sec. 5)

2 RELATED WORKS

We review related works on three topics: collaborative filtering, hybrid methods, and cross-domain recommendation.

Collaborative Filtering Recommender systems aim at learning user preferences on unknown items from their past history. Content-based recommendations are based on the matching between user profiles and item descriptions. It is difficult to build the profile for each user when there is no/few content. CF alleviates this issue by predicting user preferences based on the user-item interaction behavior, agnostic to the content [9]. Latent factor models learn feature vectors for users and items mainly based on MF [26] which has probabilistic interpretations [35]. Factorization machines (FM) can mimic MF [40]. To address the data sparsity, an item-item matrix called Shifted Positive Pointiest Mutual Information (SPPMI) is constructed from the user-item interaction matrix in the CoFactor model [28]. It then simultaneously factorizes the interaction matrix and the SPPMI matrix in a shared item latent space, enabling the usage of co-click information to regularize the learning of the user-item matrix. In contrast, we use independent unstructured text and source domain information to alleviate the data sparsity issue in the user-item matrix. Neural networks are proposed to push the learning of feature vectors towards non-linear representations, including the Neural Network Matrix Factorization (NNMF) and MultiLayer Perceptron (MLP) [10, 17]. The basic MLP architecture is extended to regularize the factors of users and items via social and geographical information [51]. Other neural approaches learn from the explicit feedback for the rating prediction task [5, 56]. We focus on learning from the implicit feedback for top-N recommendation [50].

Hybrid Filtering Items are usually associated with the content information such as unstructured text (e.g., abstracts of articles and reviews of products). CF approaches can be extended to exploit the content information [1, 47, 48] and user reviews [16, 20, 33]. Combining matrix factorization and topic modelling technique (e.g., Topic MF) is an effective way to integrate ratings with item contents [2, 29, 33]. Item reviews justify the rating behavior of a user, and item ratings are associated with their attributes hidden in reviews [14]. Topic MF methods combine latent item factors in ratings with latent topics in reviews [2, 33]. The behavior factors and topic factors are aligned with a link function such as softmax transformation in the Hidden Factors and hidden Topics (HFT) model [33] or an offset deviation in the Collaborative Topic Regression (CTR) model [47]. The CTR model assumes the latent item vector learnt from the interaction data is close to the corresponding topic proportions learnt from the text content, but allows them to be divergent from each other if necessary. Additional sources of information are integrated into CF to alleviate the data sparsity issues including knowledge graph [49, 55]. Convolutional Networks (CNNs) have been used to extract features from audio signals for music recommendation [45] and from image for product and multimedia recommendation [6, 16]. Autoencoders are used to learn an intermediate representations from text [48, 53]. Recurrent networks [1] and convolutional networks [5, 24, 56] can exploit the word order when learning the text representations. Memory networks can reason with an external memory [43]. Due to the capability of neurally learnt word embeddings to address the problems of word sparseness and semantic gap, a memory module can be used to model item content [19] or the neighborhood of users [11]. Memory networks can learn to match word semantics with the specific user. We follow this thread by using neural networks to attentively extract important information from text content.

Cross-domain Recommendation Cross-domain recommendation [3] is an effective technique to alleviate the data sparsity issue. A class of MF-based methods has been applied to cross-domain recommendation. Typical methods include the CMF approach which jointly factorizes two rating matrices by sharing the latent user factors and hence it enables knowledge transfer. CMF has its heterogeneous variants [37], and codebook transfer [27]. The coordinate system transfer can exploit heterogeneous feedbacks [38, 52]. Multiple source domains [32] and multi-view learning [13] are also proposed for integrating information from several domains. Transfer Learning (TL) aims at improving the performance of the target domain by exploiting knowledge from source domains [36]. Similar to TL, Multitask Learning (MTL) is to leverage useful knowledge in multiple related tasks to help each other [4, 54]. The cross-stitch network [34] and its sparse variant [22] enable information sharing between two base networks for each domain in a deep way. Robust learning is also considered during knowledge transfer [15]. These methods treat knowledge transfer as a global process with shared global parameters and do not match source items with the specific target item given a user. We follow this research thread by using neural networks to selectively transfer knowledge from the source items. We introduce a transfer network to exploit the source domain knowledge.

3 A BASIC NEURAL CF NETWORK

We adopt a Feedforward Neural Network (FFNN) as the base neural CF model to parameterize the interaction function [7, 8, 17]:

$$f(\boldsymbol{x}_{ui}|\boldsymbol{P}, \boldsymbol{Q}, \theta_f) = \phi_o(...(\phi_1(\boldsymbol{x}_{ui}))...), \tag{1}$$

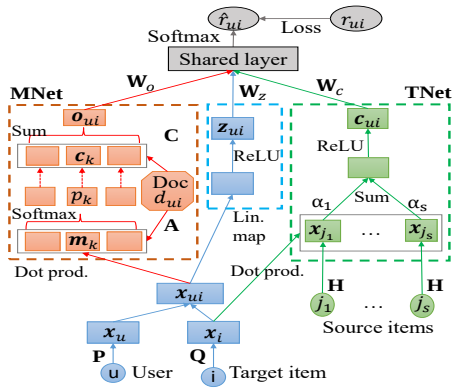


Figure 1: The architecture of TMH.

where input $\boldsymbol{x}_{ui} = [\boldsymbol{P}^T\boldsymbol{x}_u, \boldsymbol{Q}^T\boldsymbol{x}_i] \in \mathbb{R}^{2d}$ is concatenated from embeddings of user u and item i and is a projection of their one-hot encodings $\boldsymbol{x}_u \in \{0,1\}^m$ and $\boldsymbol{x}_i \in \{0,1\}^n$ via embedding matrices $\boldsymbol{P} \in \mathbb{R}^{m\times d}$ and $\boldsymbol{Q} \in \mathbb{R}^{n\times d}$, respectively. The output and hidden layers are computed by ϕ_o and $\{\phi_l\}$ in FFNN. The sizes of users and items are m and n, respectively. The rating r_{ui} is 1 if user u has an interaction with item i and 0 otherwise.

The base network consists of four modules with the information flow from the input (u, i) to the output $\hat{r}_{ui}$ as follows.

Input: $(u, i) \rightarrow \mathbb{1}_u, \mathbb{1}_i$ This module encodes user-item interaction indices. We adopt the one-hot encoding. It takes user u and item i, and maps them into one-hot encodings $\mathbb{1}_u \in \{0,1\}^m$ and $\mathbb{1}_i \in \{0,1\}^n$ where only the element corresponding to that index is 1 and all others are 0.

Embedding: $\mathbb{1}_u, \mathbb{1}_i \rightarrow \boldsymbol{x}_{ui}$ This module firstly embeds one-hot encodings into continuous representations $\boldsymbol{x}_u = \boldsymbol{P}^T\mathbb{1}_u$ and $\boldsymbol{x}_i = \boldsymbol{Q}^T\mathbb{1}_i$ by embedding matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ respectively, and then concatenates them as $\boldsymbol{x}_{ui} = [\boldsymbol{x}_u, \boldsymbol{x}_i]$, to be the input of following building blocks.

Hidden layers: $\boldsymbol{x}_{ui} \rightsquigarrow \boldsymbol{z}_{ui}$. This module takes the continuous representations from the embedding module and then transforms through several layers to a final latent representation $\boldsymbol{z}_{ui} = (\ldots(\phi_1(\boldsymbol{x}_{ui})\ldots)$. This module consists of hidden layers to learn nonlinear interaction between users and items.

Output : $\boldsymbol{z}_{ui} \rightarrow \hat{r}_{ui}$. This module predicts the score $\hat{r}_{ui}$ for the given user-item pair based on the representation $\boldsymbol{z}_{ui}$ from the last layer of multi-hop module. Since we focus on one-class collaborative filtering, the output is the probability that the input pair is a positive interaction. This can be achieved by a softmax layer: $\hat{r}_{ui} = \phi_o(z_{ui}) = \frac{1}{1+\exp(-\boldsymbol{h}^T\boldsymbol{z}_{ui})}$, where $\boldsymbol{h}$ is the parameter.

4 THE PROPOSED TMH MODEL

The architecture of the TMH model is illustrated in Fig. 1.

Matching Word Semantics with User Preferences We adapt a memory network (MNet) to integrate unstructured text since it can learn to match word semantics with user preferences [12, 19, 23, 44]. The MNet consists of one internal memory matrix $\mathbf{A} \in \mathbb{R}^{L\times 2d}$, where L is the vocabulary size (typically $L = 8,000$ after processing [47]) and $2d$ is the dimension of each memory slot, and one external memory matrix $\mathbf{C}$ with the same dimensions as $\mathbf{A}$. The function of the two memory matrices works as follows.

Given a document $d_{ui} = (w_1, w_2, \ldots, w_l)$ corresponding to the (u, i) interaction, we form the memory slots $\boldsymbol{m}_k \in \mathbb{R}^{2d}$ by mapping each word w_k into an embedding vector with matrix $\mathbf{A}$, where $k = 1, \ldots, l$ and the length of the longest document is equal to the memory size. We form a preference vector $\boldsymbol{q}^{(ui)}$ corresponding to the given document d_{ui} and the user-item interaction (u, i) where each element encodes the relevance of user u to these words given item i as: $q_k^{(ui)} = \boldsymbol{x}_u^T\boldsymbol{m}_k^{(u)} + \boldsymbol{x}_i^T\boldsymbol{m}_k^{(i)},\ k = 1, \ldots, l$, where we split the $\boldsymbol{m}_k = [\boldsymbol{m}_k^{(u)}, \boldsymbol{m}_k^{(i)}]$ into the user part $\boldsymbol{m}_k^{(u)}$ and the item part $\boldsymbol{m}_k^{(i)}$.

Then, we compute the attentive weights over words for a given user-item interaction to infer the importance of each word's unique contribution: $p_k^{(ui)} = \text{Softmax}(q_k^{(ui)}) = \frac{\exp(\beta q_k^{(ui)})}{\sum_{k'}\exp(\beta q_{k'}^{(ui)})}$, where parameter β is introduced to stabilize the numerical computation and can amplify or attenuate the precision of the attention like a temperature [18]. We set $\beta = d^{-\frac{1}{2}}$ by scaling along with the dimensionality [46].

We construct the high-level representations by interpolating the external memories with the attentive weights as the output:

$$\boldsymbol{o}_{ui} = \sum\nolimits_k p_k^{(ui)}\boldsymbol{c}_k, \tag{2}$$

where the external memory slot $\boldsymbol{c}_k \in \mathbb{R}^d$ is another embedding vector for word w_k by mapping it with matrix $\mathbf{C}$.

Selecting Source Items to Transfer We propose a novel transfer network (TNet) which can selectively transfer source knowledge for specific target item in a coarse-to-fine way. Given the source items $[j]^u = (j_1, j_2, \ldots, j_s)$ with which the user u has interacted in the source domain, TNet learns a transfer vector $\boldsymbol{c}_{ui} \in \mathbb{R}^d$ to capture the relations between the target item i and source items given the user u. The similarities between target item i and source items can be computed by their dot products: $a_j^{(i)} = \boldsymbol{x}_i^T\boldsymbol{x}_j,\ j = 1, \ldots, s$, where $\boldsymbol{x}_j \in \mathbb{R}^d$ is the embedding for the source item j by an embedding matrix $\boldsymbol{H} \in \mathbb{R}^{n_S\times d}$. This score computes the compatibility between the target item and the source items consumed by the user.

Then, we normalize similarity scores to be a probability distribution over source items: $\alpha_j^{(i)} = \text{Softmax}(a_j^{(i)})$. Finally the transfer vector is a weighted sum of the corresponding source item embeddings:

$$\boldsymbol{c}_{ui} = \text{ReLU}(\sum\nolimits_j \alpha_j^{(i)}\boldsymbol{x}_j), \tag{3}$$

where we introduce non-linearity on the transfer vector by the rectified linear unit $\text{ReLU}(x) = max(0, x)$.

Putting It All Together We firstly use a simple neural CF model (CFNet) which has one hidden layer to learn a nonlinear representation for the user-item interaction:

$$\boldsymbol{z}_{ui} = \text{ReLU}(\boldsymbol{W}\boldsymbol{x}_{ui} + \boldsymbol{b}), \tag{4}$$

where $\boldsymbol{W}$ and $\boldsymbol{b}$ are the weight and bias parameters in the hidden layer. Usually the dimension of $\boldsymbol{z}_{ui}$ is half of that $\boldsymbol{x}_{ui}$ in a typical tower-pattern architecture.

The outputs from the three individual networks can be viewed high-level features of the content text, source domain knowledge, and the user-item interaction. They come from different feature space learned by different networks. Thus, we use a shared layer on

the top of the all features: $\hat{r}_{ui} = \frac{1}{1+\exp(-\boldsymbol{h}^T \boldsymbol{y}_{ui})}$, where $\boldsymbol{h}$ is the parameter. The joint representation, $\boldsymbol{y}_{ui} = [\boldsymbol{W}_o \boldsymbol{o}_{ui}, \boldsymbol{W}_z \boldsymbol{z}_{ui}, \boldsymbol{W}_c \boldsymbol{c}_{ui}]$, is concatenated from the linear mapped outputs of individual networks where matrices $\boldsymbol{W}_o, \boldsymbol{W}_z, \boldsymbol{W}_c$ are the corresponding linear mapping transformations.

Learning Due to the nature of the implicit feedback and the task of item recommendation, the squared loss $(\hat{r}_{ui} - r_{ui})^2$ may be not suitable since it is usually for rating prediction. Instead, we adopt the binary cross-entropy loss: $\mathcal{L} = -\sum_{(u,i)\in\mathcal{S}} r_{ui} \log \hat{r}_{ui} + (1 - r_{ui}) \log(1 - \hat{r}_{ui})$, where the training samples $\mathcal{S} = \boldsymbol{R}_T^+ \cup \boldsymbol{R}_T^-$ are the union of observed target interaction matrix and randomly sampled negative pairs. Usually, $|\boldsymbol{R}_T^+| = |\boldsymbol{R}_T^-|$ and we do not perform a predefined negative sampling in advance since this can only generate a fixed training set of negative samples. Instead, we generate negative samples during each epoch, enabling diverse and augmented training sets of negative examples to be used. The objective function can be optimized by stochastic gradient descent (SGD) and its variants like adaptive moment(Adam) method [25].

Complexity In the model parameters Θ, the embedding matrices $\boldsymbol{P}$, $\boldsymbol{Q}$ and $\boldsymbol{H}$ contain a large number of parameters since they depend on the input size of users and (target and source) items, and their scale is hundreds of thousands. Typically, the number of words, i.e., the vocabulary size is $L = 8,000$ [47]. The dimension of embeddings is typically $d = 100$. Since the architecture follows a tower pattern, the dimension of the outputs of the three individual networks is also limited within hundreds. In total, the size of model parameters is linear with the input size and is close to the size of typical latent factors models [42] and neural CF approaches [17] with a hidden layer. During training, we compute the outputs of the three individual networks in parallel using mini-batch stochastic optimization which can be trained efficiently by back-propagation. TMH is scalable to the number of the training data. It can easily update when new data examples come by just feeding them into the training mini-batch. Thus, TMH can handle the scalability and dynamics of items and users similar to an online fashion. In contrast, topic modeling techniques have difficulty in benefitting from these advantages to some extent.

5 EXPERIMENTS

In this section, we conduct empirical study to answer the following questions: 1) how the proposed TMH model performs compared with state-of-the-art recommender systems; and 2) how the text content and the source domain information contributes to the proposed framework. We firstly introduce the evaluation protocols and experimental settings, and then compare the performance of different recommender systems. We further analyze the TMH model to understand the impact of the memory and transfer components. We also investigate that the improved performance comes from the cold-users and cold-items to some extent.

5.1 Experimental Settings

Dataset We evaluate on two real-world cross-domain datasets. The first dataset, **Mobile**, is provided by a large internet company, i.e., Cheetah Mobile (http://www.cmcm.com/en-us/) [30]. The information contains logs of user reading news, the history of app installation, and some metadata such as news publisher and user gender collected in one month in the US. We removed users with fewer than 10 feedbacks. For each item, we use the news title as its text content. Following [47], we filter stop words and use tf-idf to choose the top 8,000 distinct words as the vocabulary. This yields a corpus of 612K words. The average number of words per news is less than 10. The dataset we used contains 477K user-news reading records and 817K user-app installations. There are 15.8K shared users which enable the knowledge transfer between the two domains. We aim to improve the news recommendation by transferring knowledge from app domain. The data sparsity is over 99.6%.

Table 1: Datasets and statistics.

Dataset	Domain	Statistics	Amount
Mobile News	Shared	#Users	15,890
	Target	#News	84,802
		#Reads	477,685
		Density	0.035%
		#Words	612,839
		Avg. Words Per News	7.2
	Source	#Apps	14,340
		#Installations	817,120
		Density	0.359%
Amazon Product	Shared	#Users	8,514
	Target	#Clothes (Men)	28,262
		#Ratings/#Reviews	56,050
		Density	0.023%
		#Words	1,845,387
		Avg. Words Per Review	32.9
	Source	#Products (Sports)	41,317
		#Ratings/#Reviews	81,924
		Density	0.023%

The second dataset is a public **Amazon** dataset (http://snap.stanford.edu/data/web-Amazon.html), which has been widely used to evaluate the performance of collaborative filtering approaches [16]. We use two categories including Amazon Men and Amazon Sports as two domains [16, 20]. The original ratings are from 1 to 5 where five stars indicate that a user shows a positive preference on the item while the one star is not. We convert the ratings of 4-5 as positive samples. The dataset we used contains 56K positive ratings on Amazon Men and 81K positive ratings on Amazon Sports. There are 8.5K shared users, 28K Men products, and 41K Sports goods. We aim to improve the recommendation on the Men domain by transferring knowledge from relevant Sports domain. The data sparsity is over 99.7%. We filter stop words and use tf-idf to choose the top 8,000 distinct words as the vocabulary [47]. The average number of words per review is 32.9.

The statistics of the two datasets are summarized in Table 1. As we can see, both datasets are very sparse and hence we hope to improve performance by transferring knowledge from the auxiliary domain and exploiting the text content as well. Note that Amazon dataset are long text of product reviews (the number of average words per item is 32), while Cheetah Mobile contains short text of news titles (the number of average words per item is 7).

Table 2: Results on Amazon data.

Method	$topK = 5$			$topK = 10$			$topK = 20$		
	HR	NDCG	MRR	HR	NDCG	MRR	HR	NDCG	MRR
BPRMF	.0810	.0583	.0509	.1204	.0710	.0561	.1821	.0864	.0602
CDCF	.1295	.0920	.0797	.2070	.1167	.0897	.3841	.1609	.1015
CMF	.1498	.0950	.0771	.2224	.1182	.0863	.3573	.1521	.0957
HFT	.1077	.0815	.0729	.1360	.0907	.0767	.2782	.1252	.0854
TextBPR	.1517	.1208	.1104	.1777	.1291	.1138	.2268	.1414	.1171
CDCF++	.1314	.0926	.0800	.2102	.1177	.0901	.3822	.1605	.1016
MLP	.2100	.1486	.1283	.2836	.1697	.1371	.3820	.1899	.1426
MLP++	.2263	.1626	.1417	.2992	.1862	.1514	.3810	.2069	.1570
CSN	.2340*	.1680*	.1462*	.3018*	.1898*	.1552*	.3944*	.2091*	.1605*
LCMR	.2024	.1451	.1263	.2836	.1678	.1356	.3951	.1918	.1420
TMH	**.2575**	**.1796**	**.1550**	**.3490**	**.2077**	**.1666**	**.4443**	**.2311**	**.1727**
Our improve	10.04%	6.90%	6.01%	15.63%	9.43%	7.34%	12.65%	10.52%	7.60%

Evaluation Protocol For item recommendation task, the Leave-One-Out (LOO) evaluation is widely used and we follow the protocol in [17]. That is, we reserve one interaction as the test item for each user. We determine hyper-parameters by randomly sampling another interaction per user as the validation/development set. We follow the common strategy which randomly samples 99 (negative) items that are not interacted by the user and then evaluate how well the recommender system can rank the test item against these negative ones. Since we aim at top-K item recommendation, the typical evaluation metrics are hit ratio (HR), Normalized Discounted Cumulative Gain (NDCG), and Mean Reciprocal Rank (MRR), where the ranked list is cut off at $topK = \{5, 10, 20\}$. HR intuitively measures whether the reserved test item is present on the top-K list and is defined as: $HR = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\delta(p_u \leq topK)$, where p_u is the hit position for the test item of user u, and $\delta(\cdot)$ is the indicator function. NDCG and MRR also account for the rank of the hit position, respectively, and they are defined as: $NDCG = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{\log 2}{\log(p_u+1)}$, and $MRR = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{1}{p_u}$. A higher value with lower cutoff indicates better performance.

Baselines We compare with various baselines, categorized as shallow/deep, single/cross-domain, and hybrid methods. MLP++: We combine two MLPs by sharing user embeddings. CDCF++: We extend CDCF by augmenting the feature vector with words.

Baselines	Shallow method	Deep method
Single-domain	BPRMF [41]	MLP [17]
Cross-domain	CDCF [31], CMF [42]	MLP++, CSN [34]
Hybrid	HFT [33], TextBPR [16, 20]	LCMR [19]
Cross + Hybrid	CDCF++	TMH (ours)

Implementation For BPRMF, we use LightFM's implementation which is a popular CF library. For CDCF and CDCF++, we adapt the official libFM implementation. For CMF, we use a Python version reference to the original Matlab code. For HFT and TextBPR, we use the code released by their authors. The word embeddings used in the TextBPR are pre-trained by GloVe [39]. For latent factor models, we vary the number of factors from 10 to 100 with step size 10. For MLP, we use the code released by its authors. The MLP++ and CSN are implemented based on MLP. The LCMR model is similar to our MNet model. Our methods are implemented using TensorFlow. Parameters are randomly initialized from Gaussian $\mathcal{N}(0, 0.01^2)$. The optimizer is Adam with initial learning rate 0.001. The size of mini batch is 128. The ratio of negative sampling is 1. The MLP and MLP++ follows a tower pattern, halving the layer size for each successive higher layer. Specifically, the configuration of hidden layers in the base MLP network is $[64 \rightarrow 32 \rightarrow 16 \rightarrow 8]$ as in the original paper [17]. For CSN, it requires that the number of neurons in each hidden layer is the same and the configuration is $[64]*4$ (i.e., $[64 \rightarrow 64 \rightarrow 64 \rightarrow 64]$). We study several typical configurations $\{16, 32, 64, 80\} * 4$. The embedding dimension is $d = 75$.

5.2 Comparison Results

In this section, we report the recommendation performance of different methods. The comparison results are shown in Table 3 and Table 2 respectively on the Mobile and Amazon datasets where the last row is the relative improvement of TMH vs the best baseline. We have the following observations. Firstly, we can see that the proposed TMH model performs better than all baselines on the two datasets at each setting, including the MLP network, shallow cross-domain models (CMF and CDCF), deep cross-domain models (MLP++ and CSN), and hybrid methods (HFT and TextBPR, LCMR). These results demonstrate the effectiveness of the proposed neural model.

On the Mobile dataset, the differences between TMH and other methods are more pronounced for small numbers of recommended items including top-5 or top-10 where we achieve average 2.25% relative improvements over the best baseline. This is a desirable feature since we often recommend only a small number of top ranked items to consumers to alleviate the information overload issue.

Note that the relative improvement of the proposed model vs. the best baseline is more significant on the Amazon dataset than on the Mobile dataset, obtaining average 9.56% relative improvements over the best CSN baseline, though the Amazon data is sparser than the Mobile data (see Table 1). We show the benefit of combining text content by comparing with CSN. One explanation is that the relatedness of the Men and Sports domains is larger than that

Table 3: Results on Mobile data.

Method	$topK = 5$			$topK = 10$			$topK = 20$		
	HR	NDCG	MRR	HR	NDCG	MRR	HR	NDCG	MRR
BPRMF	.4380	.3971	.3606	.4941	.4182	.3694	.5398	.4316	.3730
CDCF	.5066	.3734	.3293	.5325	.4089	.3441	.5452	.4374	.3519
CMF	.4789	.3535	.3119	.5846	.3879	.3263	.6662	.4086	.3320
HFT	.4966	.3617	.3175	.5580	.4093	.3365	.6547	.4379	.3445
TextBPR	.4948	.4298	.3826	.5466	.4499	.3913	.6123	.4682	.3958
CDCF++	.4981	.3693	.3267	.6055	.4041	.3411	.6244	.4335	.3491
MLP	.5380	.4121	.3702	.6176	.4381	.3810	.6793	.4529	.3851
MLP++	.5524	.4284	.3871	.6319	.4535	.3976	.6910	.4691	.4019
CSN	.5551*	.4323*	.3920*	.6327*	.4574*	.4025*	.6908	.4732*	.4068*
LCMR	.5476	.4189	.3762	.6311	.4460	.3874	.6927*	.4619	.3918
TMH	**.5664**	**.4427**	**.4018**	**.6438**	**.4680**	**.4124**	**.6983**	**.4820**	**.4163**
Our improve	2.04%	2.42%	2.51%	1.75%	2.32%	2.47%	0.81%	1.86%	2.34%

between the news and app domains. This will benefit all cross-domain methods including CMF, CDCF, MLP++, and CSN, since they exploit information from both two domains. Another reason is that the text content contains richer information on the Amazon dataset. As it is shown in Table 1, the average number of words in the product reviews is more than that in the news titles. This will benefit all hybrid methods including HFT, TextBPR, and LCMR. We show the benefit of transferring source items by comparing with LCMR.

The hybrid TextBPR model composes a document representation by averaging the words's embeddings. This can not distinguish the important words to match user preferences. This may explain that it has some difficulty in improving the recommendation performance when integrating text content. For example, it cannot consistently outperform the pure CF method, MLP. The CSN model transfers every representations from the source network with the same coefficient. This may have a risk in transferring the noise and harm the performance, as pointed out in its sparse variant [22]. On the Amazon dataset, it is inferior to the TMH model by a large margin (though TMH leverages content information). In contrast, the memory and transfer components are both selective to extract useful information based on the attention mechanism. This may explain that our model is consistently the best at all settings.

There is a possibility that the noise from auxiliary domain and some irrelevance information contained in the unstructured text pose a challenge for exploiting them. This shows that the proposed model is more effective since it can select useful representations from the source network and attentively focus on important words to match preferences of users. In summary, the empirical comparison results demonstrate the superiority of the proposed TMH model to exploit the text content and source domain knowledge for recommendation.

5.3 Impact of Text and Source Domain

We have shown the effectiveness of the two memory and transfer components together in the proposed framework. We now investigate the contribution of each network to the TMH by eliminating the impact of text content and source domain from it in turn. 1) **TMH\M\T**: Eliminating the impact of both content and source information from TMH. This is a collaborative filtering recommender. Actually, it is equivalent to a single hidden layer MLP model. 2) **TMH\M**: Eliminating the impact of content information (MNet) from TMH. This is a novel cross-domain recommender which can adaptively select source items to transfer via the attentive weights. 3) **TMH\T**: Eliminating the impact of source information (TNet) from TMH. This is a novel hybrid filtering recommender which can attentively match word semantics with user preferences.

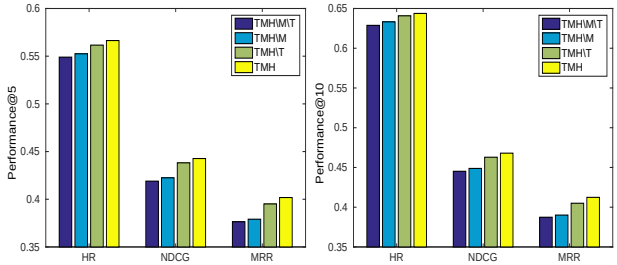


Figure 2: Contributions from unstructured text and cross-domain knowledge on Mobile Data.

The ablation analyses of TMH are shown in Figure 2 (Results on Amazon data are not shown due to space limit). The performance degrades when either memory or transfer modules is eliminated. This is understandable since we lose some information. In other words, the two components can extract useful knowledge to improve the recommendation performance. For example, TMH\T and TMH\M reduce 1.1% and 4.3% relative NDCG@10 performance, respectively, by comparing with TMH on the Mobile dataset (they are 8.5% and 16.1% on Amazon), suggesting that both memory and transfer networks learn essential knowledge for recommendation. On the two datasets, removing the memory component degrades the performance worse than removing the transfer component. This may be due to that the text content contains richer information or the source domain contains much more noise or both.

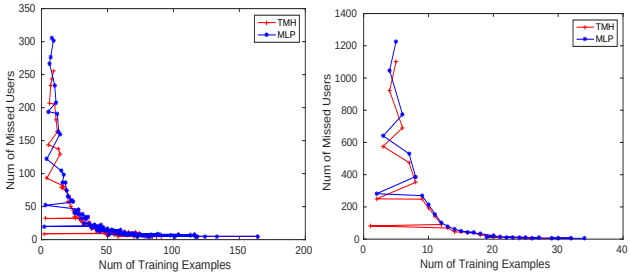


Figure 3: The Missed Hit Users (MHU) distribution (not normalized) over the number of training examples on the Mobile (left) and Amazon (right) datasets.

5.4 Improvement on Cold Users and Items

The cold-user and cold-item problems are common issues in recommender systems. When new users enter into a system, they have no history that can be exploited by the recommender system to learn their preferences, leading to the cold-user start problem. Similarly, when latest news are released on the Google News, there are no reading records that can be exploited by the recommender system to learn users' preferences on them, leading to the cold-item start problem. In general, it is very hard to train a reliable recommender system and make predictions for users and items that have few interactions. Intuitively, the proposed model can alleviate both the cold-user and cold-item start issues. TMH alleviates the cold-user start issue in the target domain by transferring his/her history from the related source domain. TMH alleviates the cold-item start issue by exploiting the associated text content to reveal its properties, semantics, and topics. We now investigate whether TMH indeed improves the performance over the cold users and items by comparing with the pure neural collaborative filtering method, MLP.

We analyse the distribution of Missed Hit Users (MHUs) of TMH and MLP (at cutoff 10). We expect that the cold users in MHUs can be reduced by using the TMH model. The more amount we can reduce, the more effective that TMH can alleviate the cold-user start issues. The results are shown in Figure 3 where the number of training examples can measure the "coldness" of a user. Naturally, the MHUs are most of the cold users who have few training examples. As we can see, the number of cold users in MHUs of MLP is higher than that of TMH. If the cold users are defined as those with less than seven training examples, then TMH reduces the number of cold users from 4,218 to 3,746 on the Amazon dataset, achieving relative 12.1% reduction. On the Mobile dataset, if the cold users are those with less than ten training examples (Mobile is denser than Amazon), then TMH reduces the number of cold users from 1,385 to 1,145 on the Mobile dataset, achieving relative 20.9% reduction. These results show that the proposed model is effective in alleviating the cold-user start issue. The results on cold items are similar and we omit them due to the page limit.

6 CONCLUSION

It is shown that the text content and the source domain knowledge can help improve the recommendation performance and they can be effectively integrated under a neural architecture. The sparse target user-item interaction matrix can be reconstructed with the knowledge from both kinds of information. The results demonstrate that our model outperforms the baseline that relies on the memory network only or relies on the transfer network only. In real-world services, data sources may belong to different providers. The data privacy is a big issue when we combine the multiple data sources. In our future work, it is worth developing new learning techniques to learn a combined model while protecting user privacy.

Acknowledgment The research has been supported by Hong Kong CERG projects 16211214/16209715/16244616, NSFC 61673202, and HKPFS PF15-16701.

REFERENCES

[1] T. Bansal, D. Belanger, and A. McCallum. Ask the gru: Multi-task learning for deep text recommendations. In *ACM RecSys*, 2016.
[2] Yang Bao, Hui Fang, and Jie Zhang. Topicmf: Simultaneously exploiting ratings and reviews for recommendation. In *AAAI*, 2014.
[3] I. Cantador, I. Fernández-Tobías, S. Berkovsky, and P. Cremonesi. Cross-domain recommender systems. In *Recommender Systems Handbook*. 2015.
[4] R. Caruana. Multitask learning. *Machine Learning*, 1997.
[5] R. Catherine and W. Cohen. Transnets: Learning to transform for recommendation. In *ACM RecSys*, 2017.
[6] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *ACM SIGIR*, 2017.
[7] H.-T. Cheng, L. Koc, J. Harmsen, et al. Wide & deep learning for recommender systems. In *Workshop on Deep Learning for Recommender Systems*, 2016.
[8] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *ACM RecSys*, 2016.
[9] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems*, 2004.
[10] G. Dziugaite and D. Roy. Neural network matrix factorization. *arXiv:1511.06443*, 2015.
[11] T. Ebesu, B. Shen, and Y. Fang. Collaborative memory network for recommendation systems. In *ACM SIGIR*, 2018.
[12] Travis Ebesu, Bin Shen, and Yi Fang. Collaborative memory network for recommendation systems. In *ACM SIGIR*, 2018.
[13] A. Elkahky, Y. Song, and X. He. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *WWW*, 2015.
[14] Gayatree Ganu, Noemie Elhadad, and Amélie Marian. Beyond the stars: improving rating predictions using review text content. In *WebDB*, 2009.
[15] Ming He, Jiuling Zhang, Peng Yang, and Kaisheng Yao. Robust transfer learning for cross-domain collaborative filtering using multiple rating patterns approximation. In *ACM WSDM*, 2018.
[16] R. He and J. McAuley. Vbpr: visual bayesian personalized ranking from implicit feedback. In *AAAI*, 2016.
[17] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *WWW*, 2017.
[18] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
[19] G. Hu, Y. Zhang, and Q. Yang. Lcmr: Local and centralized memories for collaborative filtering with unstructured text. *arXiv preprint arXiv:1804.06201*, 2018.
[20] Guang-Neng Hu and Xin-Yu. Dai. Integrating reviews into personalized ranking for cold start recommendation. In *Pacific-Asia Knowledge Discovery and Data Mining*, 2017.
[21] Guang-Neng Hu, Xin-Yu Dai, Feng-Yu Qiu, Rui Xia, Tao Li, Shu-Jian Huang, and Jia-Jun Chen. Collaborative filtering with topic and social latent factors incorporating implicit feedback. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(2):23, 2018.
[22] Guangneng Hu, Yu Zhang, and Qiang Yang. Conet: Collaborative cross networks for cross-domain recommendation. *CIKM*, 2018.
[23] Haoran Huang, Qi Zhang, Xuanjing Huang, et al. Mention recommendation for twitter with end-to-end memory network. In *IJCAI*, 2017.
[24] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. Convolutional matrix factorization for document context-aware recommendation. In *ACM RecSys*, 2016.
[25] D. Kingma and J. Ba. Adam: A method for stochastic optimization. 2015.
[26] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.
[27] B. Li, Q. Yang, and X. Xue. Can movies and books collaborate?: cross-domain collaborative filtering for sparsity reduction. In *IJCAI*, 2009.
[28] Dawen Liang, Jaan Altosaar, Laurent Charlin, and David M Blei. Factorization meets the item embedding: Regularizing matrix factorization with item

co-occurrence. In *ACM RecSys*, 2016.
[29] Guang Ling, Michael R Lyu, and Irwin King. Ratings meet reviews, a combined approach to recommend. In *ACM RecSys*, 2014.
[30] Bo Liu, Ying Wei, Yu Zhang, Zhixian Yan, and Qiang Yang. Transferable contextual bandit for cross-domain recommendation. In *AAAI*, 2018.
[31] B. Loni, Y. Shi, M. Larson, and A. Hanjalic. Cross-domain collaborative filtering with factorization machines. In *European conference on information retrieval*, 2014.
[32] Z. Lu, E. Zhong, L. Zhao, E. Xiang, W. Pan, and Q. Yang. Selective transfer learning for cross domain recommendation. In *SIAM International Conference on Data Mining*, 2013.
[33] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *ACM RecSys*, 2013.
[34] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert. Cross-stitch networks for multi-task learning. In *IEEE CVPR*, 2016.
[35] A. Mnih and R. Salakhutdinov. Probabilistic matrix factorization. In *NIPS*, 2008.
[36] S. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 2010.
[37] W. Pan, N. Liu, E. Xiang, and Q. Yang. Transfer learning to predict missing ratings via heterogeneous user feedbacks. In *IJCAI*, 2011.
[38] Weike Pan, Evan W Xiang, Nathan N Liu, and Qiang Yang. Transfer learning in collaborative filtering for sparsity reduction. In *AAAI*, 2010.
[39] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.
[40] S. Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology*, 2012.
[41] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, 2009.
[42] A. Singh and G. Gordon. Relational learning via collective matrix factorization. In *ACM SIGKDD*, 2008.
[43] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *NIPS*, 2015.
[44] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. Latent relational metric learning via memory-based attention for collaborative ranking. In *WWW*, 2018.
[45] Aäron van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *NIPS*, 2013.
[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, 2017.
[47] C. Wang and D. Blei. Collaborative topic modeling for recommending scientific articles. In *ACM SIGKDD*, 2011.
[48] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *ACM SIGKDD*, 2015.
[49] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. Dkn: Deep knowledge-aware network for news recommendation. In *WWW*, 2018.
[50] Y. Wu, C. DuBois, A. Zheng, and M. Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *ACM WSDM*, 2016.
[51] C. Yang, L. Bai, C. Zhang, Q. Yuan, and J. Han. Bridging collaborative filtering and semi-supervised learning: A neural approach for poi recommendation. In *ACM SIGKDD*, 2017.
[52] D. Yang, J. He, H. Qin, Y. Xiao, and W. Wang. A graph-based recommendation across heterogeneous domains. In *ACM CIKM*, 2015.
[53] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. In *ACM SIGKDD*, 2016.
[54] Y. Zhang and Q. Yang. A survey on multi-task learning. *arXiv:1707.08114*, 2017.
[55] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. Meta-graph based recommendation fusion over heterogeneous information networks. In *ACM SIGKDD*, 2017.
[56] L. Zheng, V. Noroozi, and P. Yu. Joint deep modeling of users and items using reviews for recommendation. In *ACM WSDM*, 2017.

Unbiased LambdaMART: An Unbiased Pairwise Learning-to-Rank Algorithm

Ziniu Hu*
University of California, Los Angeles, USA
bull@cs.ucla.edu

Yang Wang
ByteDance AI Lab, Beijing, China
wangyang.127@bytedance.com

Qu Peng
ByteDance AI Lab, Beijing, China
pengqu@bytedance.com

Hang Li
ByteDance AI Lab, Beijing, China
lihang.lh@bytedance.com

ABSTRACT

Recently a number of algorithms under the theme of 'unbiased learning-to-rank' have been proposed, which can reduce position bias, the major type of bias in click data, and train a high-performance ranker with click data. Most of the existing algorithms, based on the inverse propensity weighting (IPW) principle, first estimate the click bias at each position, and then train an unbiased ranker with the estimated biases using a learning-to-rank algorithm. However, there has not been a method for unbiased pairwise learning-to-rank that can simultaneously conduct debiasing of click data and training of a ranker using a pairwise loss function. In this paper, we propose a novel framework to accomplish the goal and apply this framework to the state-of-the-art pairwise learning-to-rank algorithm, LambdaMART. Our algorithm named Unbiased LambdaMART can jointly estimate the biases at click positions and the biases at unclick positions, and learn an unbiased ranker. Experiments on benchmark data show that Unbiased LambdaMART can significantly outperform existing algorithms by large margins. In addition, an online A/B Testing at a commercial search engine shows that Unbiased LambdaMART can effectively conduct debiasing of click data and enhance relevance ranking.

CCS CONCEPTS

• **Information systems → Learning to rank**.

KEYWORDS

Learning-to-Rank, Unbiased Learning-to-Rank, LambdaMART

ACM Reference Format:
Ziniu Hu, Yang Wang, Qu Peng, and Hang Li. 2019. Unbiased LambdaMART: An Unbiased Pairwise Learning-to-Rank Algorithm. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313447

*The full version of this paper is avaiable at arxiv. This work was done when the first author was an intern at ByteDance AI Lab.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313447

1 INTRODUCTION

Learning-to-rank, which refers to machine learning techniques on automatically constructing a model (ranker) from data for ranking in search, has been widely used in current search systems. Existing algorithms can be categorized into pointwise, pairwise, and listwise approaches according to the loss functions they utilize [11–13]. Among the proposed algorithms, LambdaMART is a state-of-the-art algorithm [3, 18]. The data for training in learning-to-rank is usually labeled by human assessors so far, and the labelling process is often strenuous and costly. This raises the question of whether it is possible to train a ranker by using click data collected from the same search system. Click data is indicative of individual users' relevance judgments and is relatively easy to collect with low cost. On the other hand, it is also noisy and biased [8, 19]. Notably, the orders of documents in the search results affect users' judgments. Users tend to more frequently click documents presented at higher positions, which is called position bias. This has been preventing practical use of click data in learning-to-rank.

Recently a new research direction, referred to as unbiased learning-to-rank, is arising and making progress. Unbiased learning-to-rank aims at eliminating bias in click data, particularly position bias, and making use of the debiased data to train a ranker. Wang et al. [16] and Joachims et al. [9] respectively propose employing the inverse propensity weighting (IPW) principle [15] to learn an 'unbiased ranker' from click data. It is proved that the objective function in learning using IPW is an unbiased estimate of the risk function defined on a relevance measure (a pointwise loss). The authors also develop methods for estimating position bias by randomization of search results online. Wang et al. [17] further develop a method for estimating position bias from click data offline. More recently Ai et al. [1] propose a method that can jointly estimate position bias and train a ranker from click data, again on the basis of IPW. In the previous work, the IPW principle is limited to the pointwise setting in the sense that position biases are only defined on click positions.

In this paper, we address the problem of jointly estimating position biases and training a ranker from click data for pairwise learning-to-rank, particularly using a pairwise algorithm, LambdaMART. To do so, we extend the inverse propensity weighting principle to the pairwise setting, and develop a new method for jointly conducting position bias estimation and ranker training.

Specifically, we give a formulation of unbiased learning-to-rank for the pairwise setting and extend the IPW principle. We define position biases as the ratio of the click probability to the relevance

probability at each position, as well as the ratio of the unclick probability to the irrelevance probability. This definition takes both the position biases at click positions and those at unclick positions into consideration. We prove that under the extended IPW principle, the objective function becomes an unbiased estimate of risk function defined on pairwise loss functions. In this way, we can learn an unbiased ranker using a pairwise ranking algorithm.

We then develop a method for jointly estimating position biases for both click and unclick positions and training a ranker for pairwise learning-to-rank, called Pairwise Debiasing. The position bias and the ranker can be iteratively learned through minimization of the same objective function. As an instance, we further develop Unbiased LambdaMART[1], an algorithm of learning an unbiased ranker using LambdaMART.

Experiments on the Yahoo learning-to-rank challenge benchmark dataset demonstrate that Unbiased LambdaMART can effectively conduct debiasing of click data and significantly outperform the baseline algorithms in terms of all measures, for example, 3-4% improvements in terms of NDCG@1. An online A/B Testing at a commercial news search engine, Jinri Toutiao, also demonstrates that Unbiased LambdaMART can enhance the performance of relevance ranking at the search engine.

The contribution of this paper includes the following proposals.

- A general framework on unbiased learning-to-rank in the pairwise setting, particularly, an extended IPW.
- Pairwise Debiasing, a method for jointly estimating position bias and training a pairwise ranker.
- Unbiased LambdaMART, an algorithm of unbiased pairwise learning-to-rank using LambdaMART.

2 RELATED WORK

Learning-to-rank is to automatically construct a ranking model from data, referred to as a ranker, for ranking in search. The algorithms of learning-to-rank can be categorized as pointwise approach, pairwise approach, and listwise approach, based on the loss functions in learning [11–13]. The pairwise and listwise algorithms usually work better than the pointwise algorithms [12], because the key issue of ranking in search is to determine the orders of documents but not to judge the relevance of documents, which is exactly the goal of the pairwise and listwise algorithms.

Recently, a new direction in learning-to-rank, referred to as unbiased learning-to-rank, is arising and making progress. The goal of unbiased learning-to-rank is to develop new techniques to conduct debiasing of click data and leverage the debiased click data in training of a ranker[2].

Wang et al. [16] apply unbiased learning-to-rank to personal search. They conduct randomization of search results to estimate query-level position bias and adjust click data for training of a ranker in personal search on the basis of inverse propensity weighting (IPW) [15]. Joachims et al. [9] theoretically prove that with the inverse propensity weighting (IPW) principle, one can obtain an unbiased estimate of a risk function on relevance in learning-to-rank. They also utilize online randomization to estimate position bias and perform training of a RankSVM model. Wang et al. [17] employ a regression-based EM algorithm to infer position bias by maximizing the likelihood of click data. The estimated position bias is then utilized in learning of LambdaMART. Recently, Ai et al. [1] design a dual learning algorithm which can jointly learn an unbiased propensity model for representing position bias and an unbiased ranker for relevance ranking, by optimizing two objective functions. Both models are implemented as neural networks. Their method is also based on IPW, while the loss function is a pointwise loss function.

Our work mainly differs from the previous work in that:

- In previous work, position bias is defined as the observation probability, and thus IPW is limited to the pointwise setting. In this work, we give a more general definition of position bias, and extend IPW to the pairwise setting, in which the loss function is pairwise and debiasing is carried out at both click positions and unclick positions.
- In previous work, estimation of position bias either relies on randomization of search results online, which can hurt user experiences [9, 16], or resorts to separate learning of a propensity model from click data offline, which can be suboptimal to relevance ranking [1, 17]. In this paper, we propose to jointly conduct estimation of position bias and learning of a ranker through minimizing one objective function.

3 FRAMEWORK

In this section, we give a general formulation of unbiased learning-to-rank, for both the pointwise and pairwise settings, and extend the inverse propensity weighting principle to the pairwise setting.

3.1 Pointwise Unbiased Learning-to-Rank

In learning-to-rank, given a query document pair denoted as x, the ranker f assigns a score to the document. The documents with respect to the query are then ranked in descending order of their scores. Traditionally, the ranker is learned with labeled data. In the pointwise setting, the loss function in learning is defined on a single data point x.

Let q denote the query and D_q the set of documents associated with q. Let d_i denote the i-th document in D_q and x_i the feature vector of q and d_i. Let r_i^+ and r_i^- represent that d_i is relevant and irrelevant respectively. The risk function in learning is defined as

$$R_{rel}(f) = \int L(f(x_i), r_i^+)\, dP(x_i, r_i^+) \tag{1}$$

where f denotes a ranker, $L(f(x_i), r_i^+)$ denotes a pointwise loss function based on an IR measure [9] and $P(x_i, r_i^+)$ denotes the probability distribution on x_i and r_i^+.

Suppose that there is a labeled dataset in which the relevance of documents with respect to queries is given. One can learn a ranker $\hat{f}_{rel}$ through the minimization of the empirical risk function (objective function) as follows.

$$\hat{f}_{rel} = \arg\min_f \sum_q \sum_{d_i \in D_q} L(f(x_i), r_i^+) \tag{2}$$

One can also consider using click data as relevance feedbacks from users, more specifically, viewing clicked documents as relevant documents and unclicked documents as irrelevant documents, and training a ranker with a click dataset. This is what we call 'biased learning-to-rank', because click data has position bias, presentation

[1]Code is available at https://github.com/acbull/Unbiased_LambdaMart

bias, etc. For convenience, let us assume that document d_i in D_q is exactly the document ranked at position i by the original ranker. Let c_i^+ and c_i^- represent that document d_i is clicked and unclicked in the click dataset respectively. The risk function and minimization of empirical risk function can be defined as follows.

$$R_{click}(f) = \int L(f(x_i), c_i^+)\, dP(x_i, c_i^+) \tag{3}$$

$$\hat{f}_{click} = \arg\min_f \sum_q \sum_{d_i \in D_q} L(f(x_i), c_i^+) \tag{4}$$

The loss function is defined on clicked documents with label c_i^+. The ranker $\hat{f}_{click}$ learned in this way is biased, however.

Unbiased learning-to-rank aims to eliminate the biases, for example position bias, in the click data and train a ranker with the debiased data. The training of ranker and debiasing of click data can be performed simultaneously or separately. The key question is how to fill the gap between click and relevance, that is, $P(c_i^+|x_i)$ and $P(r_i^+|x_i)$. Here we assume that the click probability is proportional to the relevance probability at each position, where the ratio $t_i^+ > 0$ is referred to as bias at a click position i.

$$P(c_i^+|x_i) = t_i^+ P(r_i^+|x_i) \tag{5}$$

There are k ratios corresponding to k positions. The ratios can be affected by different types of bias, but in this paper, we only consider position bias.

We can conduct learning of an unbiased ranker $\hat{f}_{unbiased}$, through minimization of the empirical risk function as follows.

$$R_{unbiased}(f) = \int \frac{L(f(x_i), c_i^+)}{t_i^+}\, dP(x_i, c_i^+) \tag{6}$$

$$= \int \frac{L(f(x_i), c_i^+)}{\frac{P(c_i^+|x_i)}{P(r_i^+|x_i)}}\, dP(x_i, c_i^+) \tag{7}$$

$$= \int L(f(x_i), c_i^+)\, dP(x_i, r_i^+) \tag{8}$$

$$= \int L(f(x_i), r_i^+)\, dP(x_i, r_i^+) = R_{rel}(f) \tag{9}$$

$$\hat{f}_{unbiased} = \arg\min_f \sum_q \sum_{d_i \in D_q} \frac{L(f(x_i), c_i^+)}{t_i^+} \tag{10}$$

In (9) click label c_i^+ in the loss function is replaced with relevance label r_i^+, because after debiasing click implies relevance.

One justification of this method is that $R_{unbiased}$ is in fact an unbiased estimate of R_{rel}. This is the so-called inverse propensity weighting (IPW) principle proposed in previous work. That is to say, if we can properly estimate position bias (ratio) t_i^+, then we can reliably train an unbiased ranker $\hat{f}_{unbiased}$.

3.2 Pairwise Unbiased Learning-to-Rank

In the pairwise setting, the ranker f is still defined on a query document pair x, and the loss function is defined on two data points x_i and x_j. Traditionally, the ranker is learned with labeled data.

Let q denote a query. Let d_i and d_j denote the i-th and j-th documents with respect to query q. Let x_i and x_j denote the feature vectors from d_i and d_j as well as q. Let r_i^+ and r_j^- represent that document d_i and document d_j are relevant and irrelevant respectively. Let I_q denote the set of document pairs (d_i, d_j) where d_i is relevant and d_j is irrelevant. For simplicity we only consider binary relevance here and one can easily extend it to the multi-level relevance case. The risk function and the minimization of empirical risk function are defined as

$$R_{rel}(f) = \int L(f(x_i), r_i^+, f(x_j), r_j^-)\, dP(x_i, r_i^+, x_j, r_j^-) \tag{11}$$

$$\hat{f}_{rel} = \arg\min_f \sum_q \sum_{(d_i, d_j) \in I_q} L(f(x_i), r_i^+, f(x_j), r_j^-) \tag{12}$$

where $L(f(x_i), r_i^+, f(x_j), r_j^-)$ denotes a pairwise loss function.

One can consider using click data to directly train a ranker, that is, to conduct 'biased learning-to-rank'. Let c_i^+ and c_j^- represent that document d_i and document d_j are clicked and unclicked respectively. Let I_q denote the set of document pairs (d_i, d_j) where d_i is clicked and d_j is unclicked. The risk function and minimization of empirical risk function can be defined as follows.

$$R_{click}(f) = \int L(f(x_i), c_i^+, f(x_j), c_j^-)\, dP(x_i, c_i^+, x_j, c_j^-) \tag{13}$$

$$\hat{f}_{click} = \arg\min_f \sum_q \sum_{(d_i, d_j) \in I_q} L(f(x_i), c_i^+, f(x_j), c_j^-) \tag{14}$$

The ranker $\hat{f}_{click}$ is however biased.

Similar to the pointwise setting, we consider dealing with position bias in the pairwise setting and assume that the click probability is proportional to the relevance probability at each position and the unclick probability is proportional to the irrelevance probability at each position. The ratios $t_i^+ > 0$ and $t_j^- > 0$ are referred to as position biases at a click position i and an unclick position j.

$$P(c_i^+|x_i) = t_i^+ P(r_i^+|x_i) \tag{15}$$

$$P(c_j^-|x_j) = t_j^- P(r_i^-|x_j) \tag{16}$$

There are $2k$ position biases (ratios) corresponding to k positions.

We can conduct learning of an unbiased ranker $\hat{f}_{unbiased}$, through minimization of the empirical risk function as follows.

$$R_{unbiased}(f) = \int \frac{L(f(x_i), c_i^+, f(x_j), c_j^-)}{t_i^+ \cdot t_j^-}\, dP(x_i, c_i^+, x_j, c_j^-) \tag{17}$$

$$= \int\int \frac{L(f(x_i), c_i^+, f(x_j), c_j^-) dP(c_i^+, x_i) dP(c_j^-, x_j)}{\frac{P(c_i^+|x_i)P(c_j^-|x_j)}{P(r_i^+|x_i)P(r_i^-|x_j)}} \tag{18}$$

$$= \int\int L(f(x_i), c_i^+, f(x_j), c_j^-) dP(r_i^+, x_i) dP(r_i^-, x_j) \tag{19}$$

$$= \int L(f(x_i), r_i^+, f(x_j), r_j^-) dP(x_i, r_i^+, x_j, r_j^-) \tag{20}$$

$$= R_{rel}(f) \tag{21}$$

$$\hat{f}_{unbiased} = \arg\min_f \sum_q \sum_{(d_i, d_j) \in I_q} \frac{L(f(x_i), c_i^+, f(x_j), c_j^-)}{t_i^+ \cdot t_j^-} \tag{22}$$

In (17) it is assumed that relevance and click at position i are independent from those at position j. In (20), click labels c_i^+ and c_j^-

are replaced with relevance labels r_i^+ and r_j^- because after debiasing click implies relevance and unclick implies irrelevance.

Similar to the pointwise setting, $R_{unbiased}$ is an unbiased estimate of R_{rel}. Therefore, if we can accurately estimate the position bias (ratios) t_i^+ and t_j^-, then we can reliably train an unbiased ranker $\hat{f}_{unbiased}$. This is an extension of the inverse propensity weighting (IPW) principle to the pairwise setting.

4 APPROACH

In this section, we present Pairwise Debiasing as a method of jointly estimating position bias and training a ranker for unbiased pairwise learning-to-rank. Furthermore, we apply Pairwise Debiasing on LambdaMart and describe the learning algorithm of Unbiased LambdaMART.

4.1 Learning Strategy

We first give a general strategy for pairwise unbiased learning-to-rank, named Pairwise Debiasing.

A key issue of unbiased learning-to-rank is to accurately estimate position bias. Previous work either relies on randomization of search results online, or resorts to a separate learning of position bias from click data offline. In this paper, we propose to simultaneously conduct estimation of position bias and learning of a ranker offline through minimizing the following regularized loss function.

$$\min_{f,t^+,t^-} \mathcal{L}(f, t^+, t^-) \tag{23}$$

$$= \min_{f,t^+,t^-} \sum_q \sum_{(d_i,d_j)\in I_q} \frac{L(f(x_i), c_i^+, f(x_j), c_j^-)}{t_i^+ \cdot t_j^-} + ||t^+||_p^p + ||t^-||_p^p \tag{24}$$

$$s.t.\ t_1^+ = 1,\ t_1^- = 1 \tag{25}$$

where f denotes a ranker, t^+ and t^- denote position biases (ratios) at all positions, L denotes a pairwise loss function, $||\cdot||_p^p$ denotes L_p regularization. Because the position biases are relative values with respect to positions, to simplify the optimization process we fix the position biases of the first position to 1 and only learn the (relative) position biases of the rest of the positions. Here $p \in [0, +\infty)$ is a hyper-parameter. The higher the value of p is, the more regularization we impose on the position biases.

In the objective function, the position biases t^+ and t^- are inversely proportional to the pairwise loss $L(f(x_i), c_i^+, f(x_j), c_j^-)$, and thus the estimated position biases will be high if the losses on those pairs of positions are high in the minimization. The position biases are regularized and constrained to avoid a trivial solution of infinity.

It would be difficult to directly optimize the objective function in (24). We adopt a greedy approach to perform the task. Specifically, for the three optimization variables f, t^+, t^-, we iteratively optimize the objective function $\mathcal{L}$ with respect to one of them with the others fixed; we repeat the process until convergence.

4.2 Estimation of position bias ratios

Given a fixed ranker, we can estimate the position biases at all positions. There are in fact closed form solutions for the estimation.

The partial derivative of objective function $\mathcal{L}$ with respect to position bias t^+ is

$$\frac{\partial \mathcal{L}(f^*, t^+, (t^-)^*)}{\partial t_i^+} = \sum_q \sum_{j:(d_i,d_j)\in I_q} \frac{L(f^*(x_i), c_i^+, f^*(x_j), c_j^-)}{-(t_i^+)^2 \cdot (t_j^-)^*} + p \cdot (t_i^+)^{p-1} \tag{26}$$

Thus, we have

$$\arg\min_{t_i^+} \mathcal{L}(f^*, t^+, (t^-)^*) = \left[\sum_q \sum_{j:(d_i,d_j)\in I_q} \frac{L(f^*(x_i), c_i^+, f^*(x_j), c_j^-)}{p \cdot (t_j^-)^*} \right]^{\frac{1}{p+1}} \tag{27}$$

$$t_i^+ = \left[\frac{\sum_q \sum_{j:(d_i,d_j)\in I_q} (L(f^*(x_i), c_i^+, f^*(x_j), c_j^-) \ / \ (t_j^-)^*)}{\sum_q \sum_{k:(d_1,d_k)\in I_q} (L(f^*(x_1), c_1^+, f^*(x_k), c_k^-) \ / \ (t_k^-)^*)} \right]^{\frac{1}{p+1}} \tag{28}$$

In (28) the result is normalized to make the position bias at the first position to be 1.

Similarly, we have

$$t_j^- = \left[\frac{\sum_q \sum_{i:(d_i,d_j)\in I_q} (L(f^*(x_i), c_i^+, f^*(x_j), c_j^-) \ / \ (t_i^+)^*)}{\sum_q \sum_{k:(d_k,d_1)\in I_q} (L(f^*(x_k), c_k^+, f^*(x_1), c_1^-) \ / \ (t_k^+)^*)} \right]^{\frac{1}{p+1}} \tag{29}$$

In this way, we can estimate the position biases (ratios) t^+ and t^- in one step given a fixed ranker f^*. Note that the method here, referred to as Pairwise Debiasing, can be applied to any pairwise loss function. In this paper, we choose to apply the pairwise learning-to-rank algorithm LambdaMART.

4.3 Learning of Ranker

Given fixed position biases, we can learn an unbiased ranker. The partial derivative of $\mathcal{L}$ with respect to f can be written in the following general form.

$$\frac{\partial \mathcal{L}(f, (t^+)^*, (t^-)^*)}{\partial f} = \sum_q \sum_{(d_i,d_j)\in I_q} \frac{1}{(t_i^+)^* \cdot (t_j^-)^*} \frac{\partial L(f(x_i), c_i^+, f(x_j), c_j^-)}{\partial f} \tag{30}$$

We employ LambdaMART to train a ranker. LambdaMART [4, 18] employs gradient boosting or MART [6] and the gradient function of the loss function called lambda function. Given training data, it performs minimization of the objective function using the lambda function.

In LambdaMART, the lambda gradient λ_i of document d_i is calculated using all pairs of the other documents with respect to the query.

$$\lambda_i = \sum_{j:(d_i,d_j)\in I_q} \lambda_{ij} - \sum_{j:(d_j,d_i)\in I_q} \lambda_{ji} \tag{31}$$

$$\lambda_{ij} = \frac{-\sigma}{1 + e^{\sigma(f(x_i) - f(x_j))}} \left| \Delta Z_{ij} \right| \tag{32}$$

where λ_{ij} is the lambda gradient defined on a pair of documents d_i and d_j, σ is a constant with a default value of 2, $f(x_i)$ and $f(x_j)$ are the scores of the two documents given by LambdaMART, ΔZ_{ij}

denotes the difference between NDCG[7] scores if documents d_i and d_j are swapped in the ranking list.

Following the discussion above, we can make an adjustment on the lambda gradient $\tilde{\lambda}_i$ with the estimated position biases:

$$\tilde{\lambda}_i = \sum_{j:(d_i,d_j)\in I_q} \frac{\lambda_{ij}}{(t_i^+)^* \cdot (t_j^-)^*} - \sum_{k:(d_k,d_i)\in I_q} \frac{\lambda_{ki}}{(t_k^+)^* \cdot (t_i^-)^*} \quad (33)$$

Thus, by simply replacing the lambda gradient λ_i in LambdaMART with the adjusted lambda gradient $\tilde{\lambda}_i$, we can reliably learn an unbiased ranker with the LambdaMART algorithm. We call the algorithm Unbiased LambdaMART.

4.4 Learning Algorithm

The learning algorithm of Unbiased LambdaMART is given in Algorithm 1. The input is a click dataset $\mathcal{D}$. The hyper-parameters are regularization parameter p and total number of boosting iterations M. The output is an unbiased ranker f and estimated position biases at all positions t^+, t^-. As is outlined in Algorithm 1, Unbiased LambdaMART iteratively calculates adjusted lambda gradient in line 4, re-trains a ranker with the gradients in line 6, and re-estimates position biases in line 7.

Algorithm 1 Unbiased LambdaMART

Require: click dataset $\mathcal{D} = \{(q, D_q, C_q)\}$; hyper-parameters p, M;
Ensure: unbiased ranker f; position biases (ratios) t^+ and t^-;
1: Initialize all position biases (ratios) as 1;
2: **for** $m = 1$ to M **do**
3: **for** each query q and each document d_i in D_q **do**
4: Calculate $\tilde{\lambda}_i$ with $(t^+)^*$ and $(t^-)^*$ using (33);
5: **end for**
6: Re-train ranker f with $\tilde{\lambda}$ using LambdaMART algorithm
7: Re-estimate position biases (ratios) t^+ and t^- using (28) and (29)
8: **end for**
9: **return** f, t^+, and t^-;

5 EXPERIMENTS

In this section, we present the results of two experiments on our proposed algorithm Unbiased LambdaMART. One is an offline experiment on a benchmark dataset. The other experiment is an online A/B testing at a commercial search engine.

5.1 Experiment on Benchmark Data

We made use of the Yahoo! learning-to-rank challenge dataset[1] to conduct an experiment. The Yahoo dataset is one of the largest benckmark dataset for learning-to-rank. It consists of 29921 queries and 710k documents. Each query document pair is represented by a 700-dimensional feature vector manually assigned with a label denoting relevance at 5 levels [5].

There is no click data associated with the Yahoo dataset. We followed the procedure in [1] to generate synthetically click data from the Yahoo dataset for offline evaluation.

[1] http://webscope.sandbox.yahoo.com

5.1.1 Click Data Generation. The click data generation process in [1] is as follows. First, one trains a Ranking SVM model using 1% of the training data with relevance labels, and uses the model to create an initial ranking list for each query. Next, one samples clicks from the ranking lists by simulating the browsing process of users. The position-based click model (PBM) [14] is utilized. It assumes that a user decides to click a document according to probability $P(c_i^+) = P(o_i^+)P(r_i^+)$. Here $P(o_i^+)$ and $P(r_i^+)$ are the observation probability and relevance probability respectively.

The probability of observation $P(o_i^+)$ is calculated by

$$P(o_i^+|x_i) = \rho_i^\theta$$

where ρ_i represents position bias at position i and $\theta \in [0, +\infty]$ is a parameter controlling the degree of position bias. The position bias ρ_i is obtained from an eye-tracking experiment in [8] and the parameter θ is set as 1 by default.

The probability of relevance $P(r_i^+)$ is calculated by

$$P(r_i^+) = \epsilon + (1-\epsilon)\frac{2^y - 1}{2^{y_{max}} - 1}$$

where $y \in [0, 4]$ represents a relevance level. The parameter ϵ represents click noise, which is set as 0.1 by default.

5.1.2 Baseline Methods. We made comprehensive comparisons between our method and the baselines. The baselines were created by combining the state-of-the-art debiasing methods and learning-to-rank algorithms. There were six debiasing methods. To make fair comparison, we used click model to generate 165660 query sessions as training dataset, and utilized the same dataset for all debiasing methods. All the hyper-parameters of the baseline models were the same as those in the original papers.

Randomization: The method, proposed by Joachims et al. [9], uses randomization to infer the observation probabilities as position biases. We randomly shuffled the rank lists and then estimated the position biases as in [1].

Regression-EM: The method, proposed by Wang et al. [17], directly estimates the position biases using a regression-EM model implemented by GBDT.

Dual Learning Algorithm: The method, proposed by Ai et al. [1], jointly learns a ranker and conducts debiasing of click data. The algorithm implements both the ranking model and the debiasing model as deep neural networks.

Pairwise Debiasing: Our proposed debiasing method, which is combined with LambdaMART. In this experiment, we set the hyper-parameter p as 0 by default. As explained below, a further experiment was conducted with different values of p.

Click Data: In this method, the raw click data without debiasing is used to train a ranker, whose performance is considered as a lower bound.

Labeled Data: In this method, human annotated relevance labels without any bias are used as data for training of ranker, whose performance is considered as an upper bound.

There were three learning-to-rank algorithms.

DNN: A deep neural network was implemented as a ranker, as in [1]. We directly used the code provided by Ai et al.[2].

[2] https://github.com/QingyaoAi/Dual-Learning-Algorithm-for-Unbiased-Learning-to-Rank

Table 1: Comparison of different unbiased learning-to-rank methods.

Ranker	Debiasing Method	MAP	NDCG@1	NDCG@3	NDCG@5	NDCG@10
LambdaMART	Labeled Data (Upper Bound)	0.854	0.745	0.745	0.757	0.790
	Pairwise Debiasing	**0.836**	**0.717**	**0.716**	**0.728**	**0.764**
	Regression-EM [17]	0.830	0.685	0.684	0.700	0.743
	Randomization	0.827	0.669	0.678	0.690	0.728
	Click Data (Lower Bound)	0.820	0.658	0.669	0.672	0.716
DNN	Labeled Data (Upper Bound)	0.831	0.677	0.685	0.705	0.737
	Dual Learning Algorithm [1]	0.828	0.674	0.683	0.697	0.734
	Regression-EM	0.829	0.676	0.684	0.699	0.736
	Randomization	0.825	0.673	0.679	0.693	0.732
	Click Data (Lower Bound)	0.819	0.637	0.651	0.667	0.711
RankSVM	Labeled Data (Upper Bound)	0.815	0.631	0.649	0.675	0.707
	Regression-EM	0.815	0.629	0.648	0.674	0.705
	Randomization [9]	0.814	0.628	0.644	0.672	0.707
	Click Data (Lower Bound)	0.811	0.614	0.629	0.658	0.697

RankSVM: We directly used the Unbiased RankSVM Software provided by Joachims et al.[3], with hyper-parameter C being 200.

LambdaMART: We implemented Unbiased LambdaMART by modifying the LambdaMART tool in LightGBM [10]. We utilized the default hyper-parameters of the tool. The total number of trees was 300, learning rate was 0.05, number of leaves for one tree was 31, feature fraction was 0.9, and bagging fraction was 0.9.

In summary, there were 13 baselines to compare with our proposed Unbiased LambdaMART algorithm. Note that Dual Learning Algorithm and DNN are tightly coupled. We did not combine Pairwise Debiasing with RankSVM and DNN, as it is beyond the scope of this paper.

5.1.3 Experimental Results. Table 1 summarizes the results. We can see that our method of Unbiased LambdaMART (LambdaMART + Pairwise Debiasing) significantly outperforms all the other baseline methods. The results of Regression-EM and Dual Learning Algorithm are compariable with those reported in the original papers. In particular, we have the following findings.

- Our method of LambdaMART+Pairwise Debiasing (Unbiased LambdaMART) achieves better performances than all the state-of-the-art methods in terms of all measures. For example, in terms of NDCG@1, our method outperforms LambdaMART+Regression-EM by 3.2%, outperforms DNN+Dual Learning by 4.3%, and outperforms RankSVM+Randomization by 8.9%.
- Pairwise Debiasing works better than the other debiasing methods. When combined with LambdaMART, Pairwise Debiasing outperforms Regression-EM by 3.2%, outperforms Randomization by 4.8% in terms of NDCG@1.
- LambdaMART trained with human labeled data achieves the best performance (upper bound). An unbiased learning-to-rank algorithm can still not beat it. This indicates that there is still room for improvement in unbiased learning-to-rank.

[3] https://www.cs.cornell.edu/people/tj/svm_light/svm_proprank.html

Table 2: Relative increases of first click ratios by Unbiased LambdaMART in online A/B testing.

Measure	Click@1	Click@3	Click@5
Increase	2.64%	1.21%	0.80%
P-value	0.001	0.004	0.023

5.2 A/B Testing at Commercial Search Engine

We further evaluated the performance of Unbiased LambdaMART by deploying it at the search engine of Jinri Toutiao, a commercial news recommendation app in China with over 100 million daily active users. We trained two rankers with Unbiased LambdaMART and LambdaMART + Click Data using click data of approximately 19.6 million query sessions collected over two days at the search engine. Then we deployed the two rankers at the search system to conduct A/B testing. The A/B testing was carried out for 16 days. In each experiment group, the ranker was randomly assigned approximately 1.5 million queries per day.

In the online environment, we observed that different users have quite different click behaviors. It appeared to be necessary to have a tighter control on debiasing. We therefore set the hyper-parameter p as 1, i.e., we conducted L_1 regularization to impose a stronger regularization on the position biases. We validated the correctness of this hyper-parameter selection on a small set of relevance dataset.

We compared the results of the two rankers in terms of first click ratios, which are the percentages of sessions having first clicks at top 1,3,5 positions among all sessions. A ranker with higher first click ratios should have better performance.

As shown in Table 2, Unbiased LambdaMART can significantly outperform LambdaMART + Click Data in terms of first click ratios at the A/B Testing. It increases the first click ratios at positions 1,3,5 by 2.64%, 1.21% and 0.80%, respectively, which are all statistically significant (p-values < 0.05). It indicates that Unbiased LambdaMART can make significantly better relevance ranking.

6 CONCLUSION

In this paper, we have proposed a general framework for pairwise unbiased learning-to-rank, including the extended inverse propensity weighting (IPW) principle. We have also proposed a method called Pairwise Debiasing to jointly estimate position biases and train a ranker by directly optimizing a same objective function within the framework. We develop a new algorithm called Unbiased LambdaMART as application of the method. Experimental results show that Unbiased LambdaMART achieves significantly better results than the existing methods on a benchmark dataset, and is effective in relevance ranking at a real-world search system.

REFERENCES

[1] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W. Bruce Croft. 2018. Unbiased Learning to Rank with Unbiased Propensity Estimation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*. 385–394.

[2] Qingyao Ai, Jiaxin Mao, Yiqun Liu, and W. Bruce Croft. 2018. Unbiased Learning to Rank: Theory and Practice. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018*. ACM, 2305–2306.

[3] Chris J.C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report.

[4] Christopher J. C. Burges, Robert Ragno, and Quoc Viet Le. 2006. Learning to Rank with Nonsmooth Cost Functions. In *Proceedings of the 20th Annual Conference on Neural Information Processing Systems, NIPS 2006, Vancouver, British Columbia, Canada, December 4-7, 2006*. 193–200.

[5] Olivier Chapelle and Yi Chang. 2011. Yahoo! Learning to Rank Challenge Overview. In *Proceedings of the Yahoo! Learning to Rank Challenge, held at ICML 2010, Haifa, Israel, June 25, 2010*. 1–24.

[6] Jerome H. Friedman. 2000. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics* 29 (2000), 1189–1232.

[7] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* 20, 4 (2002), 422–446.

[8] Thorsten Joachims, Laura A. Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, August 15-19, 2005*. 154–161.

[9] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*. 781–789.

[10] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, NIPS 2017, 4-9 December 2017, Long Beach, CA, USA*. 3149–3157.

[11] Hang Li. 2011. A Short Introduction to Learning to Rank. *IEICE Transactions* 94-D, 10 (2011), 1854–1862.

[12] Hang Li. 2014. *Learning to Rank for Information Retrieval and Natural Language Processing, Second Edition*. Morgan & Claypool Publishers.

[13] Tie-Yan Liu. 2009. Learning to Rank for Information Retrieval. *Foundations and Trends in Information Retrieval* 3, 3 (2009), 225–331.

[14] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. 521–530.

[15] Paul R. Rosenbaum and Donald B. Rubin. 1983. The Central Role of the Propensity Score in Observational Studies for Causal Effects. *Biometrika* 70 (1983), 41–55.

[16] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*. 115–124.

[17] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position Bias Estimation for Unbiased Learning to Rank in Personal Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*. 610–618.

[18] Qiang Wu, Christopher J. C. Burges, Krysta Marie Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Inf. Retr.* 13, 3 (2010), 254–270.

[19] Yisong Yue, Rajan Patel, and Hein Roehrig. 2010. Beyond position bias: examining result attractiveness as a source of presentation bias in clickthrough data. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*. 1011–1018.

Domain-aware Neural Model for Sequence Labeling using Joint Learning

Heng Huang
Alibaba Group
Hangzhou, Zhejiang, China
streakly@gmail.com

Yuliang Yan
Alibaba Group
Hangzhou, Zhejiang, China
yyl8781697@live.com

Xiaozhong Liu*
Indiana University Bloomington
Bloomington, Indiana, USA
liu237@indiana.edu

ABSTRACT

Recently, scholars have demonstrated empirical successes of deep learning in sequence labeling, and most of the prior works focused on the word representation inside the target sentence. Unfortunately, the global information, e.g., domain information of the target document, were ignored in the previous studies. In this paper, we propose an innovative joint learning neural network which can encapsulate the global domain knowledge and the local sentence/token information to enhance the sequence labeling model. Unlike existing studies, the proposed method employs domain labeling output as a latent evidence to facilitate tagging model and such joint embedding information is generated by an enhanced highway network. Meanwhile, a redesigned CRF layer is deployed to bridge the 'local output labels' and 'global domain information'. Various kinds of information can iteratively contribute to each other, and moreover, domain knowledge can be learnt in either supervised or unsupervised environment via the new model. Experiment with multiple data sets shows that the proposed algorithm outperforms classical and most recent state-of-the-art labeling methods.

ACM Reference Format:
Heng Huang, Yuliang Yan, and Xiaozhong Liu. 2019. Domain-aware Neural Model for Sequence Labeling using Joint Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313566

1 INTRODUCTION

Sequence labeling, including tasks like segmentation, named entity recognition, chunking, etc., is an essential stage for a number of downstream nature language processing applications, e.g., natural language understanding and semantic role labeling. Recently, there have been huge improvements in sequence labeling tasks and these breakthroughs were largely fueled by recent advances in deep learning models. For NLP tasks, most of the newly explored methods focused on characterizing (local) word-level or character-level embedding representation, which uncover the important latent semantic information inside the document for sequence labeling. However, for all of those works [2, 15, 17, 29], the global information, e.g., domain/category information, was ignored for model generation. Meanwhile, experience from other NLP tasks, e.g., question answering and semantic search, tells that domain information, when combining with local features, can be potentially important to further enhance the algorithm performance.

Take the following sentences as an example [Domain: Travel] *Big Apple could be a good choice for all-inclusive vacations*, [Domain: Technology] *In the past black friday, a big Apple sale event was first accounted by...*, the word pair **big apple** plays different roles in these sentences, and domain information could be useful to discover such variation. From sequence labeling viewpoint, "*big apple*" could be labeled differently (e.g., named entity or not) in the different contexts, and the global domain knowledge may contribute to the target named entity recognition model.

In this paper, in order to address this problem, we propose a novel domain-aware sequence labeling neural model, which encapsulates both global and local information by learning an independent joint embedding in each possible domain. The domain features can be characterized in either supervised or unsupervised environment. Moreover, a redesigned CRF layer is utilized to estimate output labels probability based on the statistic distribution which is generated by the joint embeddings learned from the last iteration.

The contribution of this paper contains three aspects. Firstly, to the best of our knowledge, this study is the first attempt to explore the domain knowledge in the neural model for sequence labeling tasks, and unlike prior methods, the latent global information is used as a distributional component in a joint learning framework. Secondly, a two level Bi-LSTM with an innovative highway layer is proposed to enable communications between layers and to distinguish the joint embedding from the direct concatenation of contextual and domain representations. Last but not least, we have verified the proposed method on three open corpora with domain labels for the tasks named entity recognition (NER) and segmentation, and another standard dataset without domain labels associated with the chunking task. The experiments show the superiority of the new model compared with a number of classical and most recent state-of-the-art baselines. Meanwhile, experiment results support our initial hypothesis that domain information can be important for sequence labeling tasks.

2 RELATED WORK

For sequence labeling tasks, classic linear statistical methods aim to maximize the joint probability over the linguistic observation data for label generation by leveraging the Markov approach, such like hidden Markov models [22], maximum entropy Markov models [19] and conditional random fields [14]. These methods have been successfully applied in various kinds of NLP tasks. While these models rely heavily on the hand-crafted features, and the long-range

*Corresponding Author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313566

dependency restricts the algorithm performance. In order to break this boundary, in the past few years, neural network approach, e.g., convolutional neural networks (CNN) and recurrent neural network (RNN) approaches, along with the word embedding features were employed as an important alternative for sequence labeling. Collobert et al. [6], for instance, utilized feed-forward neural network to learn the word representations from unlabeled data instead of man-made features and used CNN to extract high-level features. In addition, several models based on RNN [7] and its variation long short-term memory (LSTM) [10] or gated recurrent unit (GRU) [3], were proposed to capture the long-range dependency of the sequential data. More recently, several RNN-like models were proposed for sequence tagging, e.g., deep RNN for speech recognition [8], encoder-decoder via LSTM cell for sequence-to-sequence learning [26], etc. Recent studies also showed the benefits of merging the neural networks and CRF because the representative features obtained by the RNNs exactly fit the need of CRF. For example, Huang et al. [11] combined bidirectional LSTM (Bi-LSTM) with CRF layer and Ma [17] added CNN on Bi-LSTM-CRF for tagging sentences. Furthermore, language models were introduced for calculating the probability of context [16, 21, 23], which produced promising results in the tasks like chunking, named entity recognition, part-of-speech tagging and so on. However, to the best of our knowledge, all the existing approaches only focused on the local information and the global domain potentials were ignored.

Experience from prior studies shows that global information, e.g., domain knowledge, can be potentially important for a number of text mining tasks, e.g., information extraction [1], information retrieval [27, 28], and information summarization [4]. However, most of the previous works employed global information as a kind of simple feature or side-information for model generation. More sophisticated investigations are quite sparse in the deep learning community.

To take advantage of the domain knowledge, in this paper, we propose an innovative domain-aware sequence labeling model. Unlike prior methods, the new model encapsulates both local contextual information and global domain knowledge into a joint learning framework. For each possible domain, the proposed algorithm extracts independent joint representations of context and domain information via a multi-channel highway layer which integrates the outputs of two-level Bi-LSTM, and then estimates output labels probability using a redesigned CRF layer based on the domain distribution.

3 JOINT LEARNING FOR SEQUENCE LABELING WITH DOMAIN KNOWLEDGE

3.1 Problem Definition

To introduce global domain features, we tailor the original linear-CRF by adding a new vertex d that represents the domain knowledge of input sequence X, which is illustrated as the new probabilistic graphical model as Fig.1. Based on the Bayesian chain-rule in DAG [18, 24] the expected output labels Y^* should has a maximum marginal probability as follow:

$$Y^* = \underset{Y}{\operatorname{argmax}} \sum_{i=1}^{k} P(Y|X, d_i)P(d_i|X) \tag{1}$$

$$Y = (y_1, y_2, \cdots y_m)$$

where d_i is the ith domain and $i \in \{1, 2, \cdots k\}$. Following this model and the latent variable d_i, the label sequence depends on both input data X (local features) and domain knowledge d (global features) as an integration of all the distributional components for k domains. Consequently, we build up a joint learning framework to estimate the first factor $P(Y|X, d_i)$ (the output label sequence probability given X in the target domain d_i) and the second $P(d_i|X)$ (the probability that input sequence X falls into the domain d_i) iteratively via extracting joint embedding information for each (X, d_i).

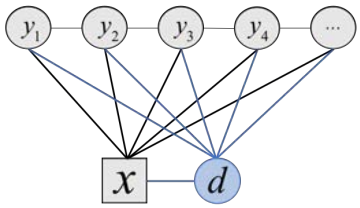


Figure 1: The new DAG for modeling sequence labeling problem based on CRF, in which the vertex d denotes the variant of domain label.

3.2 Proposed model

Based on the proposed model framework, the output label sequence is determined by both joint embedding information and domain distribution. The former factor heavily can be depend on the latter. Hence we propose a joint learning framework illustrated as Fig.2 that contains two parts. The sequence labeling part applies an innovative domain-aware CRF layer to estimate the output label probability via joint representations of domain and context through a joint representation extractor. The other, domain knowledge extractor, is a multi-class classifier for calculating domain distribution of the input sequence. Each token, in the input sequence, is represented as a concatenation of word embedding and character-level embedding, and then passed through two extractors respectively to obtain the joint representation and domain distribution. Finally, the domain-aware CRF layer calculates the probability of the label sequence by using both joint features and the domain knowledge. The two parts in the framework can be trained iteratively by forward pass and back-propagation through time (BPTT) while contributing to each other.

3.3 Joint representation extractor

In the proposed model, in order to learn the joint representation based on context and domain embedding information, a two-level depth bidirectional LSTM and an enhanced highway layer are implemented as shown in Fig.3, where x is the inputs and o denotes the outputs of LSTM cell with $\leftarrow$ or $\rightarrow$ as the forward or backward directions. v_i is the ith domain vector that is initialized randomly, and $s_!^i$ describes the joint representation for the token with the ith domain knowledge.

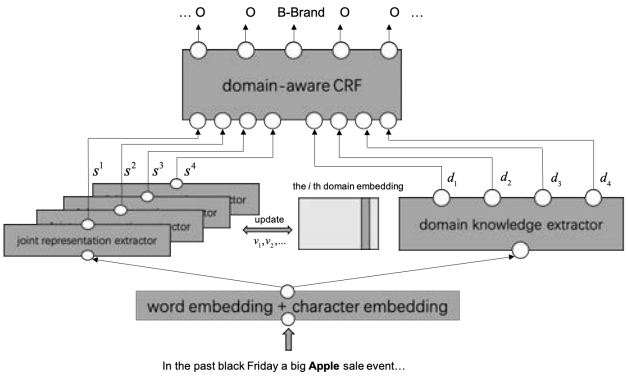


Figure 2: The overview of the proposed model. Domain knowledge extractor generates the global domain distribution and the joint representation extractor learns the joint embedding information. Domain-aware CRF layer calculates the output label probability based on the two modules.

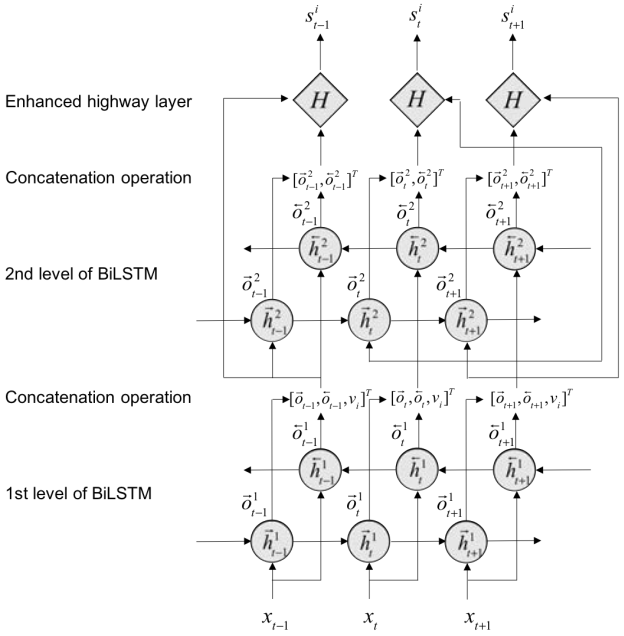


Figure 3: For the ith domain, we employ two-level Bi-LSTM to embed both context and domain features. The novel multi-channel highway layer balances the contextual and domain representations as the joint representation.

The deep Bi-LSTMs extract the contextual representation and domain embedding information respectively utilizing each level of Bi-LSTMs. Next, for adaptively learning a joint representation from context and domain embeddings, an enhanced highway layer is deployed whose cell has multiple gates to balance the effects of multi-channel inputs. Note that this highway design is quite different from the original one [25]. The multi-channel highway layer allows the networks rationally copy and transform multiple inputs and then output a joint embedding representation. The function of multi-channel highway cell can be formalized as follows:

$$
\begin{aligned}
s_t^i = &\sum_c \lambda_c (F_c(\alpha_{c,t}^i, W_c^F) \cdot T(\alpha_{c,t}^i, W_c^T) \\
&+ \alpha_{c,t}^i \cdot (1 - T(\alpha_{c,t}^i, W_c^T))) \\
\alpha_{c,t}^i &\in \{[\overrightarrow{o}_t, \overleftarrow{o}_t, v_i]^T, [\overrightarrow{o}_t^2, \overleftarrow{o}_t^2]^T\} \\
s.t. &\quad \sum_c \lambda_c = 1
\end{aligned}
\tag{2}
$$

where $\alpha_{c,t}^i$ is the output vector (hidden states) of cth layer of deep Bi-LSTMs at t-step in ith domain and $\cdot$ is element-wise product. $F_c(\cdot,\cdot)$ is the activation function of cth gate and $T(\cdot,\cdot)$ denotes the transition function that determines the proportion of transformed and original input vector α. $W_\cdot^\cdot$ is the weights for each function that can be learned automatically with the subscript for channel and superscript for function. Note that, for scale invariance, the transformation of the multi-channel highway cell subjects to the criterion that the summation of each weight λ_c equals to 1.

3.4 Domain knowledge extractor

Domain knowledge extractor is essentially a supervised multi-class classifier in this proposed framework. We build up bidirectional LSTM networks and concatenate the final states of the forward and backward networks to optimize and obtain the global features. A softmax layer is deployed to estimate the likelihood that the input sentence should be classified into the target domain. The detailed domain knowledge extractor is depicted in Fig.(4), where x is the input sequence with length n, and d denotes the domain distribution conditioned on input sequence x. For generality, this model could be estimated by MLE and the loss based on softmax function is formularized as follows:

$$
J(\theta) = -\frac{1}{m} \sum_{i=1,j=1}^{m,k} 1\{y_i = d_j\} \log \frac{e^{\theta_j^T f_i}}{\sum_{l=1}^{k} e^{\theta_l^T f_i}} \tag{3}
$$

where $1\{\cdot\}$ is indicator function equals to 1 when the ith output y_i matches the ground-truth label d_j. Note that f_i is the concatenation of final states $[\overrightarrow{f}_i, \overleftarrow{f}_i]$ for the ith sentence and θ is the parameters.

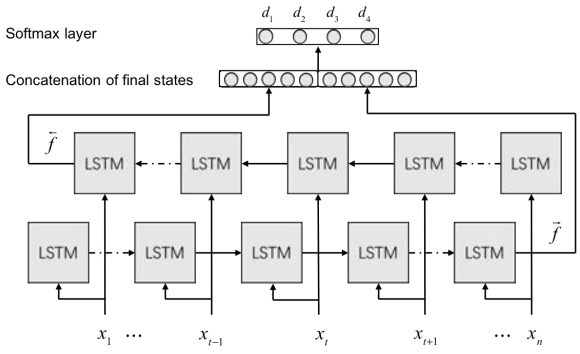


Figure 4: We use the final states of Bi-LSTM for generating the domain distribution.

To generalize the model, the proposed algorithm is able to work in two different scenarios: training samples with or without domain labels. For the latter one, unsupervised clustering algorithms, like K-means, can be cast as a pre-processing and then each training sample can be labeled with a cluster index (pseudo domain label). Finally, we train the model as a supervised learning process.

3.5 Domain-aware CRF

Based on equation (1) and fundamental theorem of random fields [9], the conditional probability can be rewritten as the joint distribution of the minimal cliques:

$$
\begin{aligned}
P(Y|X) &= \sum_{d} P(d|X) \frac{\exp(E(y,x,d))}{Z(X,d)} \\
E(y,x,d) &= \sum_{k} W_k \sum_{t} f_k(y_{t-1}, y_t, x_t, t, d) \\
Z(X,d) &= \sum_{y} \exp(\sum_{k} W_k F_k(y,x,d))
\end{aligned}
$$

where $f_k(\cdot)$ is the kth feature function within W_k as the parameter vector. x_t and y_t denote respectively the tth input token and output label subscripted with t as the position. d denotes domain distribution and $Z(\cdot)$ is the normalization which accumulates over all possible label sequences in the finite label set $\{y_l\}$. Note that the feature function $f_k(\cdot)$ consists of two parts: one is to calculate the transition probability from last state to the current and the other represents the correlation between labels and input tokens. Hence, the energy function $E_k(\cdot)$ has the form as the following:

$$
\begin{aligned}
E(y,x,d) = &\sum_{k} U_k \sum_{t} \lambda_k(y_{t-1}, y_t, x_t, t, d) \\
&+ \sum_{k} V_k \sum_{t} \mu_k(y_t, x_t, t, d)
\end{aligned}
$$

whose form is similar to recurrent neural networks. From this perspective, the redesigned domain-aware CRF block is depicted as Fig.5, in which the hidden state h_t is an integration of the probabilistic components in the different domains and d_i denotes the probability corresponding to the input sentence dropping in the ith domain. s_t^i is the jointly embedding vector in the ith domain produced by joint representation extractor. Note that each block C in domain-aware CRF layer is implemented based on forward-backward algorithm using RNN cell [5].

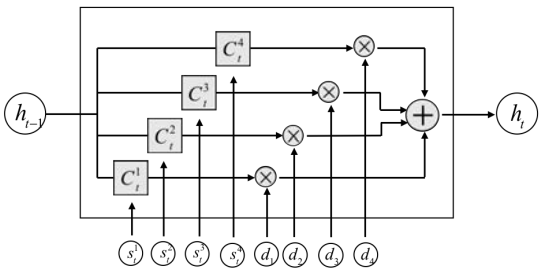


Figure 5: The domain-aware CRF block calculates the marginal probability based on equation (1).

As a discriminative model, the domain-aware CRF can be trained by maximum likelihood estimation (MLE), whose objective function $L(W)$ is a collection from all the domain channels as follows:

$$
\begin{aligned}
L(W) &= \sum_{d} L_d(W_d) \\
&= \sum_{d} \log(P(d|x) \prod_{x,y} P(y|x,d)^{\overline{P}(x,y)}) \\
&= \sum_{d} \overline{P}(x,y) \log P(d|x) \\
&+ \sum_{d} \sum_{x,y} \overline{P}(x,y) \sum_{k} W_K F_k(x,y,d) \\
&- \sum_{d} \sum_{x} \overline{P}(x) \log \sum_{y} e^{\sum_k W_k F_k(x,y,d)}
\end{aligned}
\tag{4}
$$

where $\overline{P}(x,y)$ is the empirical distribution that can be calculated from training samples. Hence, the gradient can be derived with feature vector $F(\cdot)$ as follows:

$$
\begin{aligned}
\frac{\partial L}{\partial W} &= \sum_{d} \sum_{x,y} \overline{P}(x,y) F(x,y,d) \\
&- \sum_{d} \sum_{x,y} \overline{P}(x) P(y|x,d) F(x,y,d)
\end{aligned}
\tag{5}
$$

3.6 Model training

Since the joint representations learning in the proposed model is determined by both global domain distribution of input data X and the local context features, multi-task learning is applied for training with two different but related tasks: domain label classifier (a sub-task), and the sequence labeling model generation (main-task). For each batch in an epoch, the two tasks are trained alternatively. We firstly train the multi-class classifier using forward pass to calculate the loss $J(\theta)$ as eq.(3) and apply backward propagation to update the parameters θ. After the domain distribution is obtained, the joint representation s_i can be computed, and then the objective $L(W)$ of labeling model can be computed via domain-aware CRF as eq.(4). Finally the gradient can be computed as eq.(5) and back-propagation is deployed to update the parameters W and domain embedding v_i of main-task. Note that training will be stopped if the performance on the development set is no longer improving for 15 epochs or the maximum number of iterations has been achieved. The iterative training process is clarified in Algo.1.

4 EXPERIMENTS

4.1 Dataset

In order to validate the proposed method for sequence labeling, we use two types of datasets: one is blended data crossing multiple domains, and the other is a single data set without domain labels. The former ones are generated by combining several independent data sets following the works of Kurata et al. [13] and Zhai et al. [30]. Three considerably large data sets, LARGE3, LARGE4, and LARGE5 are obtained for the tasks of *named entity recognition (NER)* and *segmentation*. LARGE*k* represents how many sub sets it contains. For the latter one, we use CoNLL 2000 shared task for *text chunking* to test the effectiveness of our model on the data without specific domain information. For comparison, several baselines are selected

Table 1: Details of each sub set in LARGEs.

	training set	development set	test set	label scale	domain
ATIS	3,983	995	893	64	airline travel
MIT Restaurant	3,270	775	1521	9	restaurant
MIT Movie	7,726	2,049	2,443	13	movie
SemEval.2017.t10	2,418	415	847	4	scientific publications
SemEval.2018.t8	9,424	1,213	618	4	cyber-security reports

Algorithm 1 Training algorithm

randomize v_i,θ,W
for each epoch **do**
  if sub-task **then**
    forward and backward pass based on the loss $J(\theta)$ as eq.(3)
  end if
  if main-task **then**
    for each domain i **do**
      calculate$P(d_i|x) = softmax_i(f)$
      calculate s^i as eq.(2)
    end for
    forward pass $L(W)$ as eq.(4)
    backward pass $\nabla_W L(W)$ as eq.(5)
  end if
  update all parameters
  stop if early stopping condition satisfied
end for

including classic linear probabilistic models and neural networks. The details of datasets and baselines are depicted as follows.

LARGEs. The construction of this kind of data set is aiming for containing a large amount of data sequences from several domains, which is motivated by the fact that many practical applications of sequence tagging involve a large scale of documents crossing multiple domains. Specifically, five open data sets for NER and segmentation are included: ATIS, MIT Restaurant Corpus, MIT Movie Corpus, SemEval 2017 task 10 and SemEval 2018 task 8. LARGEk denotes the combination of k-headed sub sets, and we collect the training set, development set and test set from each subset respectively for constructing the corresponding samples of LARGEs. The detail of each involved dataset are shown as Tab.1 and all the subsets can be found online.

CoNLL 2000. To verify our model can be applied to the data without domain labels, we use CoNLL 2000 to test the effectiveness. This data set is extracted from Wall Street Journal corpus, containing 8,068 training, 868 development and 893 test sentences. 12 syntactic labels are defined for text chunking, e.g. NP, VP, ADJP, other, etc.

4.2 Baselines

For comparison, several baselines are selected including classic linear probabilistic models and neural networks. We first choose CRF [14] as the classical linear probabilistic model baseline for its broad applications. Then, three recent state-of-art neural models, BiLSTM-CRF [11], LMcost [23], and LM-LSTM-CRF [21], are selected as the deep learning baselines. All of those models are implemented using open-source code offered by authors and we test them on the LARGEs and CoNLL 2000 data sets to evaluate the algorithms' performance.

4.3 Preparation

The proposed model applies same settings of pre-trained word embeddings and optimization strategy as baseline models referred before. To initialize the word lookup table, we deploy GloVe [20], 300-dimension pre-trained word embeddings, and the character/domain embeddings are both 100-dimension. To avoid bias, each test is repeated 10 times for averaging the final score. In LARGEs, each sample has a specific field label because its original data source covers a certain field as shown in Tab.1. However, unlike LARGEs, samples in CoNLL 2000 do not associate with explicit domain labels, hence, we apply K-means clustering to explore the distributional distinctiveness among the data sequences and labeled them with the cluster index. After the label system has been defined, the domain knowledge extractor is trained as a multi-class classifier. Then the labeling model can work based on the domain distribution.

The novel approach in this paper is noted as DA-BiLSTM-CRF++ in the following sections. In order to validate the usefulness of the domain knowledge in the sequence labeling tasks, we launch another two experiment models, DA-BiLSTM-CRF and DA-BiLSTM-CRF+, which simplify the joint representation of the proposed model. The key difference between these models and the DA-BiLSTM-CRF++ is whether the second layer of Bi-LSTM and the enhanced highway layer participate the joint representation learning. For details, joint representation extractor in DA-BiLSTM-CRF directly connects the contextual representation with domain embedding, and feeds the domain-aware CRF with the vector $[\overrightarrow{o}_t^1, \overleftarrow{o}_t^1, v_i]^T$. While the joint representation extractor in DA-BiLSTM-CRF+ applies the second layer of Bi-LSTM to learn a joint representation $[\overrightarrow{o}_t^2, \overleftarrow{o}_t^2]^T$ from the direct concatenation $[\overrightarrow{o}_t^1, \overleftarrow{o}_t^1, v_i]^T$ to feed the top domain-aware CRF layer. Moreover, DA-BiLSTM-CRF++ uses a multi-channel highway layer to balance the embeddings of the first and the second Bi-LSTM layer.

4.4 Experimental results

To evaluate the performance of the novel approach on the datasets with domain discrimination, we perform named entity recognition (NER) and segmentation on the test sets of LARGE3, LARGE4 and LARGE5. The performance is measured in terms of F1-score, which is calculated by the public script *conlleval.pl*. After the named entity

Table 2: The results of NER and segmentation on LARGEs. Note that * denotes p-value < 0.05 and ** denotes p-value < 0.005.

	NER			segmentation			chunking
	LARGE3	LARGE4	LARGE5	LARGE3	LARGE4	LARGE5	CoNLL 2000
CRF	80.42	76.54	70.27	85.19	76.54	73.89	86.18
BiLSTM-CRF	85.79	77.81	75.04	88.85	81.32	78.65	94.25
LMcost	86.08	77.65	75.39	89.47	82.67	79.87	94.03
LM-LSTM-CRF	86.2	77.98	75.48	89.67	83.22	79.98	94.41
DA-BiLSTM-CRF	86.71	78.12	75.47	89.68	82.54	79.9	94.7
DA-BiLSTM-CRF+	86.67	**78.6****	75.68	**89.91***	83.38	80.18	94.89
DA-BiLSTM-CRF++	**86.89****	78.53	**75.72***	89.83	**83.43***	**80.35****	**94.95***

labels have been obtained, we reassign each word with IOB labels to assess the segmentation performance. Also, we report the F1-score of segmentation which is computed by *conlleval.pl* in the Tab.2. The results indicate the novel model produces significant improvements (p-value < 0.05) due to the effects of the domain knowledge. DA-BiLSTM-CRF++ and DA-BiLSTM-CRF+ perform better than the basic version DA-BiLSTM-CRF in most of cases, because the former two models learn the joint embedding information via contextual and domain representation rather than a direct concatenation of them. DA-BiLSTM-CRF+ is functionally equivalent to DA-BiLSTM-CRF++ and performs roughly the same.

The robustness of the proposed model has also been tested. Several experiments on CoNLL 2000 shared dataset are deployed for testing the chunking performance, which is a typical sequence labeling task as well as NER or segmentation. The best F1-score is achieved with the parameter $k = 2$ when K-means clustering has been applied as a pre-processing to obtain the scattering distinctiveness of unlabeled data. In the same way, we report the F1-score via *conlleval.pl*, and the chunking results are displayed in Tab.2. The evaluation results show the effectiveness of proposed model (p-value < 0.05). When the explicit training labels are not available, the proposed approach of obtaining (pseudo) domain knowledge is promising. However, the performance is vulnerable to the incorrect clustering labels.

4.5 Analysis.

The model is validated for three main sequence labeling tasks by the experiments aforementioned, and the essential reasons of the superiority against prior works should be further investigated. As Fig.1 illustrated, the fundamental distinction of the proposed approach is the contribution of both the sequence features and the domain knowledge, which contains two important factors. First, we employ the joint representation of the input sequence X and domain knowledge d; second, the marginal probability calculated by domain-aware CRF based on eq.(1) is utilized to enhance the labeling performance. The improvements produced by the first point can be observed through the comparisons between the basic model DA-BiLSTM-CRF and DA-BiLSTM-CRF+/DA-BiLSTM-CRF++ that deploy some extra neural networks to learn the joint representation from the direct connection of contextual and domain embeddings. The second factor can be proved via the advantages of the basic model DA-BiLSTM-CRF in the experiments using the redesigned CRF layer to integrate each domain component. Furthermore, we investigate the correlation between performance and the class divergence in each dataset using Silhouette Coefficient [12], which is widely deployed as a relocating measurement. Tab.3 reveals the positive relationship between Silhouette Coefficient and the model performance, which proves that our method maximizes its potential if the experiment corpus can be clearly divided into a number of independent partitions (as domain knowledge). Note that S.C. represents the Silhouette Coefficient and F1-gap is the gap between the best-performed and the second-performed F1-scores.

Table 3: The relationship between Silhouette Coefficient and F1-gap on each dataset.

	S.C.	F1-gap
LARGE3	0.2128	0.69
LARGE4	0.2042	0.62
LARGE5	0.1246	0.24
CoNLL 2000@$k = 2$	0.2671	0.54
CoNLL 2000@$k = 3$	0.1249	0.49
CoNLL 2000@$k = 4$	0.0807	0.31
CoNLL 2000@$k = 5$	0.0610	0.17

5 CONCLUSION AND FUTURE WORK

In this study, we propose a novel neural sequence labeling model in a joint learning framework. While the model takes the domain probability distribution into account, the global and local features are integrated via a novel multi-channel highway learning for efficient joint embedding optimization. Moreover, an innovative CRF layer integrates the joint representations that range over all the possible domains and calculates the probability of the output labels. The proposed method is trained in a multi-task learning framework. Three datasets (with or without explicit domain labels) associated with different sequence labeling tasks were employed for algorithm evaluation. Experiment result indicated that the proposed model significantly outperforms the previous classical and recent baselines in a number of different sequence labeling tasks, like NER, segmentation and chunking. In this work, clustering based approach was implemented to address the problem of domain label missing, and in the future, we will investigate more sophisticated end-to-end method to consider domain knowledge into sequence labeling task when explicit domain labels are not available.

REFERENCE

[1] Kai-Wei Chang, Scott Wen-tau Yih, Bishan Yang, and Chris Meek. 2014. Typed Tensor Decomposition of Knowledge Bases for Relation Extraction. (2014).

[2] Jason PC Chiu and Eric Nichols. 2015. Named Entity Recognition with Bidirectional LSTM-CNNs. *arXiv preprint arXiv:1511.08308* (2015).

[3] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *arXiv preprint arXiv:1409.1259* (2014).

[4] Arman Cohan and Nazli Goharian. 2017. Contextualizing Citations for Scientific Summarization Using Word Embeddings and Domain knowledge. *arXiv preprint arXiv:1705.08063* (2017).

[5] Michael Collins. 2013. The Forward-Backward Algorithm. (2013).

[6] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (almost) from Scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[7] Christoph Goller and Andreas Kuchler. 1996. Learning Task-Dependent Distributed Representations by Backpropagation Through Structure. In *Neural Networks, 1996., IEEE International Conference on*, Vol. 1. IEEE, 347–352.

[8] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech Recognition with Deep Recurrent Neural Networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 6645–6649.

[9] John M Hammersley and Peter Clifford. 1971. Markov fields on Finite Graphs and Lattices. (1971).

[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[11] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *arXiv preprint arXiv:1508.01991* (2015).

[12] L. Kaufman and P.J. Rousseeuw. 1990. Finding Groups in Data: an Introduction to Cluster Analysis.

[13] Gakuto Kurata, Bing Xiang, Bowen Zhou, and Mo Yu. 2016. Leveraging Sentence-Level Information with encoder LSTM for Semantic Slot Filling. *arXiv preprint arXiv:1601.01530* (2016).

[14] John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*. 282–289.

[15] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. *arXiv preprint arXiv:1603.01360* (2016).

[16] Liyuan Liu, Jingbo Shang, Frank Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. 2017. Empower Sequence Labeling with Task-Aware Neural Language Model. *arXiv preprint arXiv:1709.04109* (2017).

[17] Xuezhe Ma and Eduard Hovy. 2016. End-to-End Sequence Labeling via Bi-Directional LSTM-CNNs-CRF. *arXiv preprint arXiv:1603.01354* (2016).

[18] David JC MacKay. 1996. Equivalence of Linear Boltzmann Chains and Hidden Markov Models. *Neural Computation* 8, 1 (1996), 178–181.

[19] Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. 2000. Maximum Entropy Markov Models for Information Extraction and Segmentation. In *ICML*, Vol. 17. 591–598.

[20] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[21] Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-Supervised Sequence Tagging with Bidirectional Language Models. *arXiv preprint arXiv:1705.00108* (2017).

[22] Lawrence R Rabiner. 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proc. IEEE* 77, 2 (1989), 257–286.

[23] Marek Rei. 2017. Semi-Supervised Multitask Learning for Sequence Labeling. *arXiv preprint arXiv:1704.07156* (2017).

[24] Lawrence K Saul and Michael I Jordan. 1995. Boltzmann Chains and Hidden Markov Models. In *Advances in Neural Information Processing Systems*. 435–442.

[25] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway Networks. *arXiv preprint arXiv:1505.00387* (2015).

[26] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Advances in neural information processing systems*. 3104–3112.

[27] Howard Turtle and W Bruce Croft. 2017. Inference Networks for Document Retrieval. In *ACM SIGIR Forum*, Vol. 51. ACM, 124–147.

[28] Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit Semantic Ranking for Academic Search via Knowledge Graph Embedding. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1271–1279.

[29] Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. 2016. Multi-Task Cross-Lingual Sequence Tagging from Scratch. *arXiv preprint arXiv:1603.06270* (2016).

[30] Feifei Zhai, Saloni Potdar, Bing Xiang, and Bowen Zhou. 2017. Neural Models for Sequence Chunking.. In *AAAI*. 3365–3371.

A Multimodal Text Matching Model for Obfuscated Language Identification in Adversarial Communication*

Longtao Huang
Alibaba Group
Beijing, China
hlightening@163.com

Ting Ma
School of Cyber Security, University of Chinese Academy of Sciences
Beijing, China
mating@iie.ac.cn

Junyu Lin†
Institute of Information Engineering, Chinese Academy of Sciences
Beijing, China
linjunyu@iie.ac.cn

Jizhong Han
Institute of Information Engineering, Chinese Academy of Sciences
Beijing, China
hanjizhong@iie.ac.cn

Songlin Hu
School of Cyber Security, University of Chinese Academy of Sciences
Beijing, China
husonglin@iie.ac.cn

ABSTRACT

Obfuscated language is created to avoid censorship in adversarial communication such as sensitive information conveying, strong sentiment expression, secret actions plan, and illegal trading. The obfuscated sentences are usually generated by replacing one word with another to conceal the textual content. Intelligence and security agencies identify such adversarial messages by scanning with a watch-list of red-flagged terms. Though semantic expansion techniques are adopted, the precision and recall of the identification is limited due to the ambiguity and the unbounded creation way. To this end, this paper frames the obfuscated language identification problem as a text matching task, where each message is checked whether matches a red-flagged term. We propose a multimodal text matching model which combining textual and visual features. The proposed model extends a Bi-directional Long Short Term Memory network with a visual-level representation component to achieve the given task. Comparative experiments on real-world dataset demonstrate that the proposed method could achieve a better performance than the previous methods.

CCS CONCEPTS

• **Information systems** → *Web searching and information discovery*; • **Computing methodologies** → *Phonology / morphology*.

KEYWORDS

Obfuscated Language Identification, Text Matching, Natural Language Understanding

*This research is supported in part by the National Key Research and Development Program of China (No. 2017YFB1010000) and the National Natural Science Foundation of China (No. 61702500).
†Junyu Lin is the corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313410

ACM Reference Format:
Longtao Huang, Ting Ma, Junyu Lin, Jizhong Han, and Songlin Hu. 2019. A Multimodal Text Matching Model for Obfuscated Language Identification in Adversarial Communication. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313410

1 INTRODUCTION

Motivation: Nowadays, online social networks have become more and more important in the diffusion of information by increasing the spread of live content and diverse opinions [2]. The development of online social networks provide convenience for people to get a very vast source of information on an unprecedented scale[9]. But at the same time, the security of cyberspace also faces enormous challenges. Some netizens have utilized online social networks to discuss sensitive information, plan for secret actions, trade illegal products, spread political rumours, or express strong sentiment. In order to detect such harmful communications, intelligence and security agencies would maintain a watch-list of red-flagged terms such as attack, bomb and heroin. The watch-list is used for keyword-spotting in online spread messages which are filtered as suspicious information [8].

However, netizens usually create and use obfuscated language to evade the censorship of intelligence and security agencies[13]. The original information is obfuscated by replacing the red-flagged term with an ordinary word or a brand new word (not a word before). Such substitution words are called *morphs* of the original red-flagged terms[11]. For example, "金太阳(Kim Sun)" is a morph generated from the original term "金日成(Kim Il-sung)" because the character "日" in the original term means "太阳(Sun)". The morph "火乍弓单" is generated from the original term "炸弹(Bomb)" by decomposing the Chinese characters. The morph "P0RN" of "PORN" is generated by substituting the letter "O" with the number "0" because they are visually similar. Such kind of obfuscated language can help to avoid the automatic detection since the watch-list of red-flagged terms is always manually maintained and impossible to cover all morphs.

Problem Statement: This paper addresses the problem of identifying obfuscated language that is created to evade censorship. The

raw input of this task is a watch-list of red-flagged terms and a set of documents to be scanned. The target is to find all documents that mention any of the red-flagged terms even the original terms are obfuscated. An intuitive solution is to expand the original terms by enumerating all possible morphs. However, morphs are created in unbounded way and evolve rapidly over time. Furthermore, some morphs are hardly identified even by human. So enumerating all morphs corresponding to the original terms is impossible. This opens an unexplored area of obfuscated language processing [13]. In detail, this task entails the following challenges:

- *Synonymy and Polysemy.* Due to the synonymy of natural language, many words or phrases can express similar meaning with the original terms. Some researchers have tried to apply an external knowledge base[1, 18], while existing knowledge bases can hardly cover a wide range of original terms and similar words referred by morphs. Meanwhile, considering the polysemy of natural language, a document might be innocuous even if the red-flagged terms or corresponding morphs are used. For example, *ice* and *skiing* refer to *Cocaine* in illegal drug dealing. But they refer to a kind of winter sports in more common situations.
- *Domain Knowledge.* Some morphs are generated based on domain knowledge. It is difficult to identify such obfuscations even for a man who is not familiar with this domain. For example, "薄熙来(Xilai Bo)" is a former Chinese politician. In Chinese social networks, Bo is morphed as "平西王(Conquer West King)" who was a historical figure four hundreds years ago. They have been linked together because they have similar experiences, both governed the same region and experienced a downfall and an arrest at the end of their career.
- *Unbounded candidates.* Obfuscated language effectively hides in plain sight because anything can be a candidate referred to the original term[25]. Netizens create morphs with a motley variety of functions. Some researchers have summarized the categories of obfuscated morphs [12, 24], such as phonetic substitution, spelling decomposition, translation and transliteration, semantic interpretation,etc. However, people's imagination is infinite and any connections can be utilized to create obfuscated language. So it is impractical to enumerate all obfuscation functions.

State of the Arts: With our observation, no matter how complicated rules are applied to create obfuscated language, the core features used can fall into two types, textual features and visual features. Previous researchers mainly focus on textual features to identify obfuscated language. Some methods [3, 4, 10] utilize the known morphs as seeds and find the patterns of the co-occurrence of the known morphs and the original terms, then find new morphs based on the acquired patterns. However, morphs are not used together with the original terms in most obfuscated language. Some methods pre-define rules to generate morphs [12, 24, 25]. The rules are all based on lexical relations, sematic relations or context relations between the original terms and morphs. Our method goes beyond the previous ones by targeting at identifying the adversarial passages obfuscated with various morphs rather than only finding morphs corresponding to the original terms.

Furthermore, we observe that some morphs are generated without any textual relations. But from the visual aspects, they are similar with the original terms. We name such relations as visual features. For example, the above mentioned morphs "火乍弓单" and "P0RN" belong to such type. The morphs based on visual features are created more casually than textual features based obfuscation. And it is impossible to summarize some generation rules since people can use any visually similar characters or symbols to replace the original terms. Thus makes previous methods fail to identify most content with visual features based obfuscation.

Our Solutions: To this end, we propose a multimodal text matching model to identify obfuscated language.

Firstly, we transfer this problem to a text matching task, where each document is checked whether matches a red-flagged term based on the matching degree of their relevance. Thus can help to overcome the first challenge since we mainly care about the relevance between documents and red-flagged terms no matter the original terms appears in the documents or not.

Besides, we adopt the Bi-directional Long Short Term Memory network(Bi-LSTM) to model textual features, which can help to extract the context information. This is useful to overcome the second challenge. When the original terms or morphs are used in a specific domain, their surrounding information (people, organizations, topics, events, etc) is always similar.

In order to overcome the third challenge, we propose to involve visual features to cover as many as possible candidates. We involve visual features by regarding each document as a source image and each red-flagged term as a template image in a template matching problem from computer vision area [6].

At last, we integrate both textual features and visual features in the neural network to achieve the final matching results. To evaluate the proposed method, we construct a large-scale dataset from Sina Weibo[1], which contains manually annotated labels. Experimental results show that our method outperforms previous methods.

Contributions: This paper presents a multimodal text matching model for identifying obfuscated language given a watch-list of red-flagged terms. To sum up, the contributions are as follows:

- the task of obfuscated language identification is novel and has not been carefully studied in previous methods. In this paper, we frame the problem as a text matching task and evaluate several methods suitable for this task.
- we introduce a multimodal text matching model that combines textual and visual features to identify obfuscated language.
- we conduct experiments on real-world dataset and validate the effectiveness of obfuscated language identification with multimodal features.

2 PROBLEM DEFINITION

The target of this paper is to identify obfuscated language in adversarial communication. The core problem of obfuscated language identification is to check whether a document is relevant to a red-flagged term in the given watch-list. To this end, we frame the given problem as a text matching task formalized as follows.

[1] http://weibo.com

Definition 2.1. Given a set of documents $D = \{d_1, d_2, ..., d_m\}$ and a watch-list of red-flagged term $Q = \{q_1, q_2, ..., q_n\}$. The matching task is to measure each pair of $< d_i, q_j >$ through a scoring function based on the representation:

$$match(d_i, q_j) = F_S(\Phi(d_i), \Phi(q_j))$$

where Φ is a function to map d_i and q_j to a representation vector, and F_S is the scoring function based on the relevance between them.

In this paper, we add a virtual term q_0 to the set Q which means none red-flagged. For each $d_i \in D$, if d_i have the highest matching score with $q_j (j \neq 0)$, then d_i is identified as an obfuscated document relevant to q_j. Note that the original terms or the substitution might appear, or both or neither appears in the obfuscated document.

3 THE PROPOSED MODEL

In this section, we describe our multimodal model for matching each document d_i to a given term q_j. The overall architecture of our proposed model is shown in Figure 1. For clarity, we describe our model in three parts: Textual-level Representation, Visual-level Representation, and Multimodal Fusion. The input of our model is a document d_i and a set of given terms Q. The output of our model is the selected term which gets the maximum probability that d_i belongs to. We will illustrate the details of the proposed framework in the following section.

3.1 Textual-level Representation

The textual-level representation tries to capture textual features of each document. The proposed method is based on the following hypothesis:

Hypothesis 1: When people talk about a subject, the textual context (entities, attributes or events) will be similar no matter whether the corresponding morphs are used or not.

For example, "Big Yao" is a morph of the NBA star "Ming Yao". When people use "Big Yao" to describe "Ming Yao", "Houston Rockets", "NBA", "Basketball" are mostly mentioned in the context, which is almost the same when talking about "Ming Yao".

According to hypothesis 1, the context around the terms and the corresponding morphs would not change much. Thus, we adopt word2vec[15] to generate the feature vector $V = \{v(x_1), ..., v(x_n)\}$ for each document, where $\{x_1, ..., x_n\}$ symbolizes the segments of the document.

To capture the context from both the past and the future, we adopt the Bidirectional LSTM(Bi-LSTM) network, which is a bidirectional formal model of LSTM [16]. Bi-LSTM can use the historical and future states data of input series simultaneously. It connects two recurrent neural networks with positive(forward state) and negative(backward state) time direction to a single output which has proved to be a good solution for expressing context information. With this structure, the output layer can obtain more tremendous information. Meanwhile, Bi-LSTM has the ability to solve the issue of learning long-term dependencies.

The structure of Bi-LSTM model is shown in Figure 2. Two directional neurons of Bi-LSTM do not interact. The general training process is with similar algorithms as LSTM. Forward and backward states are processed first in the forward pass, then do forward pass for output neurons. For the backward pass, output neurons are processed before forward and backward states are passed. Backward pass is similar to standard Back Propagation which involves reusing chain rules. Weights are updated only after the forward and backward passes are complete.

In a single LSTM unit, at the time slice t, the updating equations are as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{1}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{3}$$

$$C_t = f_t \times C_{t-1} + i_t \times \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{4}$$

$$h_t = o_t \times \tanh(C_t) \tag{5}$$

where f_t, i_t, o_t denote the forget gate's activation vector, the input gate's activation vector, and the output gate's activation vector respectively. σ and $tanh$ denote the sigmoid function and the $tanh$ function. C_t denotes the cell state vector and h_t denotes the output vector of the LSTM unit. $W_f, W_i, W_o, W_C, b_f, b_i, b_o, b_C$ denote the weight matrices and the bias vector parameters which need to be learned during training.

Bi-LSTM contains two separate LSTMs to extract both past and future information, where one encodes the sentence from the start to the end, and the other encodes the sentence with the reverse direction. Thus, we can obtain two representations $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$, then the two representations are concatenated to acquire the final output:

$$h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}] \tag{6}$$

After achieving the final hidden states $h_t (t = 1, 2, ..., n)$, the matching probability vector P_{text} corresponding to terms Q can be obtained using $Softmax$, which is the extracted feature of the textual-level representation.

$$P_{text} = Softmax(h_1, h_2, ..., h_n) \tag{7}$$

3.2 Visual-level Representation

The visual-level representation module captures the visual features based on template matching technique in computer vision. Template matching has proven to be a useful solution in digital image processing for finding small parts of a source image which matches a template image. This is similar to match documents with red-flagged terms. The red-flagged terms can be regarded as template images, and the documents to be checked can be regarded as source images. Then the matching process is like the template matching process to find the source images that match the template images. An example of a source image, a template image and the search area is shown in Figure 3. The arrow indicates the direction of the movement of the template.

Firstly, the red-flagged terms Q and documents D are all transferred to corresponding text images, where the font, size and color are unified. Given a document d_i and a red-flagged term q_j, the corresponding images are denoted as I and T. Then the template matching process is as follows.

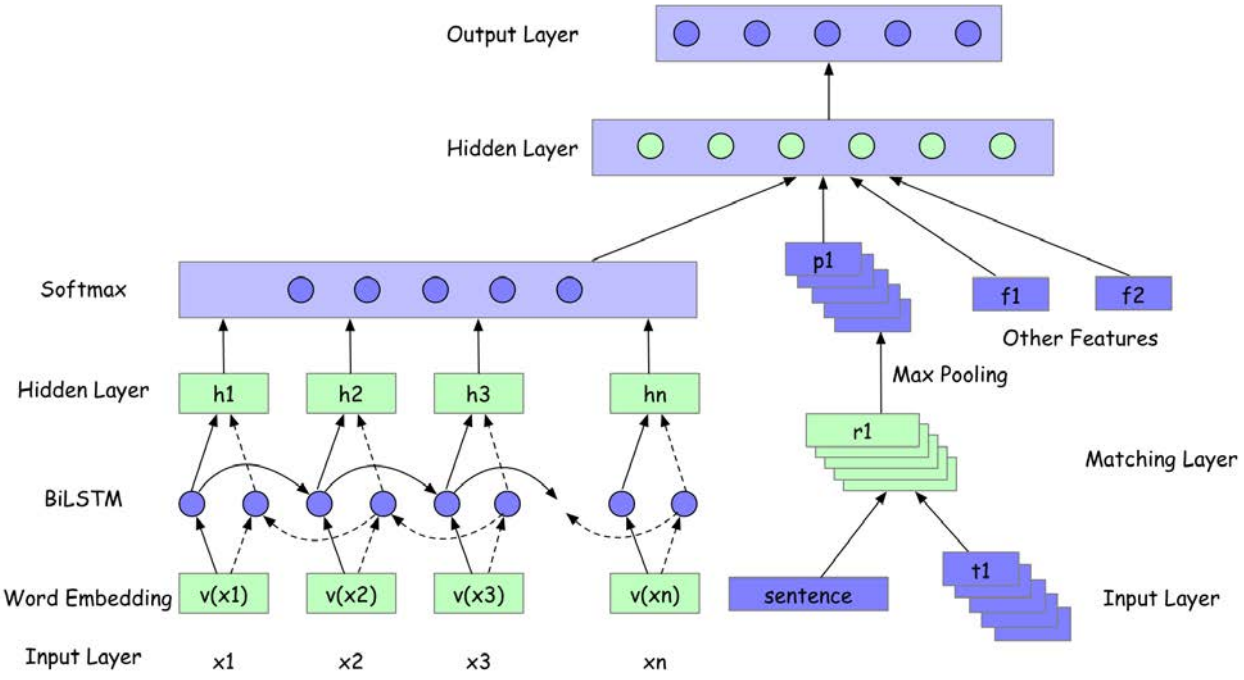


Figure 1: Architecture of the proposed multimodal model.

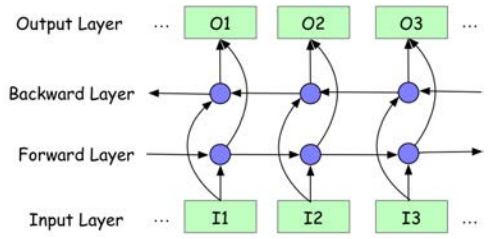


Figure 2: Structure of the Bi-LSTM model.

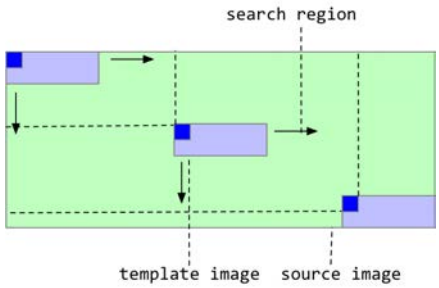


Figure 3: Source image, template image and search area.

(1) Position the template T over the image I at every possible location. In order to avoid phrases spanning multiple lines, I is generated with all content in only one line. That is, the height of I is identical to T. Then T is moved horizontally over I with one pixel each time.

(2) For each time T is positioned, we will compute the numeric metric of similarity between the template T and the image segment it currently overlaps with. The metric represents how similar the template T is to that segment of the source image I.

(3) For each location of T over I, the metric is stored in the result matrix R. Since the height of I is identical to T, R is an array in our experiment. Each location (x, y) in R contains the matching metric (normalized correlation matching):

$$R(x,y) = \frac{\sum_{x',y'} T(x',y') \cdot I(x+x', y+y')}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot I(x+x', y+y')^2}} \tag{8}$$

where (x, y) is the coordinate of the location T over I. (x', y') is the coordinate of the pixels in T. $T(x, y)$ and $I(x, y)$ is the RGB value of the pixel at the location.

(4) After T is slidden over the whole area of I and each $R(x, y)$ is calculated, the global max pooling is applied on the $R(x, y)$, which is done by applying a max filter to non-overlapping subregions of the initial representation. Then a vector P_{visual} is obtained:

$$P_{visual} = MaxPool(R) \tag{9}$$

Thus, P_{visual} is the output of the visual-level representation module. The highest value $Smax$ in P_{visual} corresponds to the matching position in R.

3.3 Multimodal Fusion

After achieving the outputs of the textual-level and the visual-level representations with the equations (7) and (9), we come to the multimodal fusion process. An existing work demonstrates that some non-linear interactions between the sentence vectors can help to learn good representations[17]. To achieve this, we include two common pair-wise vector operations: subtraction and multiplication. Then the integration process is as follows.

$$f_1 = P_{text} - P_{visual} \tag{10}$$

$$f_2 = P_{text} \times P_{visual} \tag{11}$$

$$F = P_{text} \oplus P_{visual} \oplus f_1 \oplus f_2 \tag{12}$$

where f_1 and f_2 denote two combining operations of P_{text} and P_{visual}. F denotes the final fusion features.

Finally, we utilize a Multi-Layer Perceptron (MLP) to serve as a matching model in which F is a input vector. Then F is considered to compute the probability distribution and the output layer presents the matching results.

4 EXPERIMENTS

4.1 Datasets

We crawl 1,245,812 tweets from Sina Weibo from October 1 to 31, 2017. We select 55 keywords as red-flagged terms and 186 morphs corresponding to the terms are found from the tweets. Some examples are shown in Table 1. Then we sample 43,206 non-redundant tweets as the experimental dataset and label them by three independent annotators. The annotated dataset is then randomly split into training set(27,642 tweets), development set(6,910 tweets) and test set(8,654 tweets). We tune parameters on the development set and apply them to get final results on test set.

All the 43,206 tweets are annotated by three Chinese native speakers. Each tweet is annotated with a red-flagged term or not. A tweet would be labeled with a given term as long as the tweet mentions the meaning of the term no matter the original term is used or not. Due to the polysemy of natural language, a tweet including a red-flagged term is possible to be annotated irrelevant with the given term.

4.2 Baselines

In this part, we will describe the methods in the comparative experiments. We mainly concern three groups of methods suitable for the given problem: classification methods, morph recognition methods and text matching methods. The methods are listed as follows:

- **Classification methods**: Such methods treat the problem in this paper as a classification problem. Each document will be classified into two categories (Relevant with the red-flagged terms or not). We select four popular models: **Naive Bayes (NB)**[20], **Logistic Regression (LR)**[21], **Decision Tree (DT)**[14] and **Support Vector Machine (SVM)**[22].
- **Morph Recognition methods**: Such methods are used to find all possible morphs to expand red-flagged terms for scanning. We select two state-of-art work on morph recognition **CEMD**[25] and **KIEM**[12]
- **Text Matching methods**: Such methods are used to match each document to the given terms, which is the same as our methods. We select one state-of-art work **MatchPyramid**[19], which takes text matching as image recognition.

4.3 Evaluation Metrics

To evaluate different methods mentioned above, we compare the outputs of different methods against the ground truth. The metrics for comparison are precision, recall and F1.

$$precision = \frac{|M_i \cap GT|}{|M_i|} \tag{13}$$

$$recall = \frac{|M_i \cap GT|}{|GT|} \tag{14}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \tag{15}$$

where M_i is the set of output tweets by the i-th method, GT is the set of tweets with red-flagged labels.

4.4 Parameter Setting

To initialize the word embeddings used in our method, we pre-train the word embeddings on 2 million documents from Sina Weibo and news websites. The sentence length is set to 20. The dimensions of the word embeddings and Bi-LSTM units are set 200. The model is trained using RMSprop optimizer with a learning rate 0.001.

4.5 Results Analysis

Overall Evaluation. Table 2 shows the comparison results on precision, recall and F-measure. From the results, we can observe that our approach performs better than the other methods. The morph recognition methods CEMD and KIEM perform better than the classification methods because they extend the original terms by finding more morphs. But they fail to generate all correct morphs so that the precision and recall are not as good as our proposed methods. Though MatchPyramid also solve this problem in a text matching way, the F1 of our approach is 12% better than it. This validates the benefit of involving visual features.

Zero-shot Evaluation. We now turn to the evaluation of generalization behavior with respect to some obfuscated morphs that have been never seen by the systems. We consider the zero-shot

Table 1: Some Examples in Datasets

Red-flagged Terms	Morphs	Tweets
炸弹(Bomb)	火(Fire)乍(Sudden)弓(Arrow)单(Single)	如何组装N2火乍弓单 How to assemble the bomb N2
赌博(Gamble)	堵(block)博(doctor)	我的梦想就是靠堵博发家致富 My dream is to become rich through gambling
裸体(Nude)	果(Fruit)体(Body)	电影中十大令人难忘的果体场面... Top-10 nude scenes in movies...

Table 2: Overall Evaluation Result

Methods	Precision	Recall	F1
NB	0.82	0.64	0.72
LR	0.83	0.66	0.74
DT	0.84	0.63	0.72
SVM	0.85	0.66	0.74
CEMD	0.86	0.81	0.83
KIEM	0.88	0.85	0.86
MatchPyramid	0.87	0.77	0.82
Our Approach	**0.93**	**0.91**	**0.92**

setting, i.e. some morphs for the original terms only appear in the test set and are not used during training. This is more similar with practical cases because morphs are not static and evolves rapidly over time. The evaluation results are summarized in Table 3. Our approach also performs best in comparison with the other methods. The drop ratio means the decrease compared with the overall evaluation. We can observe that our approach also achieves the lowest drop ratio. Previous methods fail to keep high precision and recall if they never see the morphs in training set since the context might also change when a new morph is created. The result also validates that our approach can cover more kinds of obfuscation by combining textual and visual features.

Table 3: Zero-Shot Evaluation Result

Methods	Precision	Recall	F1	Drop Ratio
NB	0.49	0.34	0.40	44.16%
LR	0.67	0.41	0.51	30.82%
DT	0.67	0.43	0.52	27.25%
SVM	0.74	0.45	0.56	24.68%
CEMD	0.76	0.67	0.71	14.63%
KIEM	0.78	0.71	0.74	14.04%
MatchPyramid	0.75	0.64	0.69	15.46%
Our Approach	**0.84**	**0.82**	**0.83**	**9.79%**

5 RELATED WORK

Obfuscated Language Identification has been recently proposed [13]. Several previous work on name alias detection, word obfuscation, and morph resolution can contribute to this task. The basic idea of detecting name alias is automatically finding the patterns between the real names and aliases, and then these patterns are utilized to search for candidates. In the end, various ranking scores are calculated to measure the associations between a name and its candidates [3, 4, 10]. Yin et al. introduce a system to extract and rank name aliases in emails [23]. However, the co-occurrence of the real names and aliases is not common in most adversarial communications. Brennan et al. summarize three types of approaches to create adversarial messages: obfuscation, imitation, and translation [5]. They find manual circumvention approaches work best, while automated translation approaches are not so effective. Other recent work detects word obfuscation in adversarial communication using existing commonsense Knowledge Bases such as ConceptNet to expand similar words for substitution [1, 7]. Morph resolution is another closely related work to this task which aims to resolve the morphs to the target entities. Encoding and decoding entity morph is a special case of coded name alias to hide the original entities for evading censorship[11, 12, 24, 25]. Recent works propose a number of novel methods to automatically encode implicit and relevant morphs[24], including Phonetic Substitution, Spelling Decomposition, Nickname Generation, Translation and Transliteration and Historical Figure Mapping, which can effectively avoid decoding detection. They also study on effective morph decoder methods, which can automatically recognize and resolve entity morphs to their real targets [11, 12, 25]. Such methods need to take the encoding rules into consideration.

Our model differs from these previous methods, and improves the model in two critical ways: (1) propose a multimodal text matching model which can overcome the challenges of Synonymy and Polysemy and Domain Knowledge. (2) incorporate visual features to cover as many as possible morphs in obfuscated language

6 CONCLUSIONS

In this work, we propose a multimodal text matching model for obfuscated language identification. We introduce a multimodal model by integrating a text-level module and a visual-level module. In textual level, we employ Bi-LSTM model to extract the matching probability feature. In visual level, we utilize template matching to get the visual matching feature. Finally, we evaluate our approach with a manually annotated dataset collected from Sina Weibo. Compared to other state-of-the-art models that use only textual features, the performance of the proposed method is much better. We will try to extend to other languages in the future.

REFERENCES

[1] S. Agarwal and A. Sureka. 2015. Using common-sense knowledge-base for detecting word obfuscation in adversarial communication. In *2015 7th International Conference on Communication Systems and Networks (COMSNETS)*. 1–6. https://doi.org/10.1109/COMSNETS.2015.7098738

[2] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. 2012. The Role of Social Networks in Information Diffusion. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*. ACM, New York, NY, USA, 519–528. https://doi.org/10.1145/2187836.2187907

[3] Danushka Bollegala, Taiki Honma, Yutaka Matsuo, and Mitsuru Ishizuka. 2008. Mining for personal name aliases on the web. In *International Conference on World Wide Web*. 1107–1108.

[4] Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2011. Automatic Discovery of Personal Name Aliases from the Web. *IEEE Transactions on Knowledge & Data Engineering* 23, 6 (2011), 831–844.

[5] Michael Brennan, Sadia Afroz, and Rachel Greenstadt. 2012. Adversarial stylometry:Circumventing authorship recognition to preserve privacy and anonymity. *Acm Transactions on Information & System Security* 15, 3 (2012), 1–22.

[6] Roberto Brunelli. 2009. *Template Matching Techniques in Computer Vision: Theory and Practice*. WILEY. 261 pages.

[7] Sonal N. Deshmukh, Ratnadeep R. Deshmukh, and Sachin N. Deshmukh. 2014. Performance Analysis of Different Sentence Oddity Measures Applied on Google and Google News Repository for Detection of Substitution. *IRJES Com* (2014).

[8] Sw. Fong, D. Roussinov, and D. B. Skillicorn. 2008. Detecting Word Substitutions in Text. *IEEE Transactions on Knowledge & Data Engineering* 20, 8 (2008), 1067–1076.

[9] Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A. Zighed. 2013. Information Diffusion in Online Social Networks: A Survey. *SIGMOD Rec.* 42, 2 (July 2013), 17–28. https://doi.org/10.1145/2503792.2503797

[10] Johannes Hoffart, Yasemin Altun, and Gerhard Weikum. 2014. Discovering emerging entities with ambiguous names. In *International Conference on World Wide Web*. 385–396.

[11] Hongzhao Huang, Zhen Wen, Dian Yu, Heng Ji, Yizhou Sun, Jiawei Han, and He Li. 2013. Resolving Entity Morphs in Censored Data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. 1083–1093.

[12] Longtao Huang, Lin Zhao, Shangwen Lv, Fangzhou Lu, Yue Zhai, and Songlin Hu. 2017. KIEM: A Knowledge Graph Based Method to Identify Entity Morphs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. ACM, New York, NY, USA, 2111–2114. https://doi.org/10.1145/3132847.3133123

[13] Heng Ji and Kevin Knight. 2018. Creative Language Encoding under Censorship. In *Proceedings of COLING2018 Workshop on NLP for Internet Freedom*. 23–33.

[14] D Landgrebe. 2002. A survey of decision tree classifier methodology. *IEEE Transactions on Systems Man & Cybernetics* 21, 3 (2002), 660–674.

[15] Changzhou Li, Yao Lu, Junfeng Wu, Yongrui Zhang, Zhongzhou Xia, Tianchen Wang, Dantian Yu, Xurui Chen, Peidong Liu, and Junyu Guo. 2018. LDA Meets Word2Vec: A Novel Model for Academic Abstract Clustering. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018*. 1699–1706.

[16] Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016*.

[17] Allen Nie, Erin D. Bennett, and Noah D. Goodman. 2017. DisSent: Sentence Representation Learning from Explicit Discourse Relations. *CoRR* abs/1710.04334 (2017). http://arxiv.org/abs/1710.04334

[18] Xiaoman Pan, Taylor Cassidy, Ulf Hermjakob, Heng Ji, and Kevin Knight. 2015. Unsupervised Entity Linking with Abstract Meaning Representation. In *The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*. 1130–1139.

[19] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016*. 2793–2799.

[20] Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, and David R. Karger. 2003. Tackling the Poor Assumptions of Naive Bayes Text Classifiers. In *Machine Learning, Proceedings of the Twentieth International Conference ICML 2003,*. 616–623.

[21] Walter A. Shewhart and Samuel S. Wilks. 2005. *Applied Logistic Regression, Second Edition*.

[22] Abhisek Ukil. 2002. Support Vector Machine. *Computer Science* 1, 4 (2002), 1–28.

[23] Meijuan Yin, Xiaonan Liu, Junyong Luo, and Xiangyang Luo. 2013. A System for Extracting and Ranking Name Aliases in Emails. *Journal of software* 8, 3 (2013), 737–745.

[24] Boliang Zhang, Hongzhao Huang, Xiaoman Pan, Heng Ji, Kevin Knight, Zhen Wen, Yizhou Sun, Jiawei Han, and Bulent Yener. 2014. Be Appropriate and Funny: Automatic Entity Morph Encoding. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014*. 706–711.

[25] Boliang Zhang, Hongzhao Huang, Xiaoman Pan, Sujian Li, Chin-Yew Lin, Heng Ji, Kevin Knight, Zhen Wen, Yizhou Sun, Jiawei Han, and Bulent Yener. 2015. Context-aware Entity Morph Decoding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015*. 586–595.

The Chain of Implicit Trust: An Analysis of the Web Third-party Resources Loading

Muhammad Ikram
muhammad.ikram@mq.edu.au
Macquarie University
University of Michigan

Rahat Masood
rahat.masood@data61.csiro.au
UNSW and Data61, CSIRO

Gareth Tyson
g.tyson@qmul.ac.uk
Queen Mary University of London

Mohamed Ali Kaafar
dali.kaafar@mq.edu.au
Macquarie University and Data61, CSIRO

Noha Loizon
noha.loizon@data61.csiro.au
Data61, CSIRO

Roya Ensafi
ensafi@umich.edu
University of Michigan

ABSTRACT

The Web is a tangled mass of interconnected services, where websites import a range of external resources from various third-party domains. The latter can also load resources hosted on other domains. For each website, this creates a dependency chain underpinned by a form of implicit trust between the first-party and transitively connected third-parties. The chain can only be loosely controlled as first-party websites often have little, if any, visibility on where these resources are loaded from. This paper performs a large-scale study of dependency chains in the Web, to find that around 50% of first-party websites render content that they did not directly load. Although the majority (84.91%) of websites have short dependency chains (below 3 levels), we find websites with dependency chains exceeding 30. Using VirusTotal, we show that 1.2% of these third-parties are classified as suspicious — although seemingly small, this limited set of suspicious third-parties have remarkable reach into the wider ecosystem.

ACM Reference Format:
Muhammad Ikram, Rahat Masood, Gareth Tyson, Mohamed Ali Kaafar, Noha Loizon, and Roya Ensafi. 2019. The Chain of Implicit Trust: An Analysis of the Web Third-party Resources Loading. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313521

1 INTRODUCTION

In the modern web ecosystem, websites often load resources from a range of third-party domains such as ad providers, tracking services, content distribution networks (CDNs) and analytics services. This is a well known design decision that establishes an *explicit trust* between websites and the domains providing such services. However, often overlooked is the fact that these third-parties can further load resources from other domains, creating a *dependency chain*. This results in a form of *implicit trust* between first-party websites and any domains loaded further down the chain.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313521

Consider the `bbc.com` webpage, which loads JavaScript from `widgets.com`, which, upon execution loads additional content from another third-party, say `ads.com`. Here, `bbc.com` as the first-party website, *explicitly* trusts `widgets.com`, but *implicitly* trusts `ads.com`. This can be represented as a simple dependency chain in which `widgets.com` is at level 1 and `ads.com` is at level 2. Past work tends to ignore this, instead, collapsing these levels into a single set of third-parties [4, 22]. Here, we argue that this overlooks a vital aspect of website design. For example, it raises a notable security challenge, as first-party websites lack visibility on the resources loaded further down their domain's dependency chain. This potential threat should not be underestimated as errant active content (*e.g.,* JavaScript) opens the way to a range of further exploits, *e.g.,* Layer-7 DDoS attacks [23] or massive ransomware campaigns [15].

This paper studies dependency chains in the web ecosystem. Although there has been extensive work looking at the presence of third-parties in general [4, 20, 22], little work has focused on how content is indirectly loaded. We start by inspecting how extensive dependency chains are across the Alexa's top-200K (Section 2). We confirm their prominence, finding that around 50% of websites *do* allow third-parties to form dependency chains (*i.e.,* they implicitly trust third-parties they do not directly load). The most commonly *implicitly* trusted third-parties are well known operators, *e.g.,* `google-analytics.com` and `doubleclick.net`: these are implicitly imported by 68.3% (134,510) and 46.4% (91,380) websites respectively. However, we also observe a wide range of more obtuse third-parties such as `pippio.com` and `51.la` imported by 0.52% (1,146) and 0.51% (1,009) of websites. Although the majority (84.91%) of websites have short chains (with levels of dependencies below 3), we find first-party websites with dependency chains exceeding 30 in length. This not only complicates page rendering, but also creates notable attack surface.

We then proceed to inspect if *suspicious* or even potentially *malicious* third-parties are loaded via these long dependency chains (Section 4). We do not limit this to just traditional malware, but also include third-parties that are known to mishandle user data and risk privacy leaks. Using the VirusTotal API [14], we classify third-party domains into innocuous *vs.* suspicious. When using a reasonable classification threshold, we find that 1.2% of third-parties are classified as suspicious. Although seemingly small, we find that this limited set of suspicious third-parties have remarkable reach. 73% of websites under-study load resources from suspicious third-parties,

and 24.8% of first-party webpages contain at least 3 third-parties classified as suspicious in their dependency chain. This, of course, is impacted by many considerations which we explore — most notably, the power-law distribution of third-party popularity, which sees a few major players on a large fraction of websites. We share our datasets, experimental testbed code and scripts used in this paper with the wider research community for further analysis of the consequences of implicit trust `https://wot19submission.github.io`.

2 DATASET AND DATA ENRICHMENT

2.1 Data Collection

We obtain the resource dependencies of the Alexa top-200K websites' main pages[1] using the method described in [18]. This Chromium-based Headless [6] crawler renders a given website and tracks resource dependencies by recording network requests sent to third-party domains. The requests are then used to reconstruct the dependency chains between each first-party website and its third-party URLs. Note that each first-party can trigger the creation of multiple dependency chains (to form a tree structure). To construct the dependency tree, we identify third-party requests by comparing the second level domain of the page (*e.g.,* `bbc.com`) to the domains of the requests (*e.g.,* `cdn.com` and `ads.com` via `widgets.com`). Those with different second level domains are considered third-party. We ignore the sub-domains so that a request to a domain such as `player.bbc.com` is not considered as third-party. Due to the lack of purely automated mechanism to disambiguate between site-specific sub-domains (*e.g.,* `player.bbc.com`) or country-specific sub-domains (*e.g.,* `bbc.co.uk`), we leverage Mozilla Public Suffix list [27] and `tldextract` [19] for this task. From the Alexa Top-200k websites, we collect 11,287,230 URLs which consist of 6,806,494 unique external resources that correspond to 68,828 and 196,940, respectively, unique second level domains of third- and first-parties.

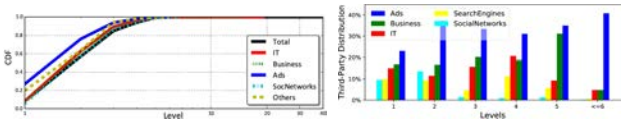
Constructing the dependencies between objects in a webpage is a non-trivial task. In cases where third-party JavaScript gets loaded into a first-party context, and then makes an AJAX request, the HTTP(S) request appears to be from the first-party (i.e. the *referrer* will be the first-party). To overcome such cases and to preserve the information on relations between the nested resource dependencies, we allow the crawler to include the URL of the third-party from which the JavaScript was loaded by first-party.

To augment the dependency data, we use VirusTotal which is an online solution which aggregates the scanning capabilities provided by more than 68 AV tools, scanning engines and datasets. It has been commonly used in the academic literature to detect malicious apps, executables, software and domains [7, 11, 13, 16, 17]. We use the VirusTotal report APIs to obtain the VTscore for each third-party URL. This score represents the number of AV tools that flagged the website as malicious (max. 68). The reports also contain meta-information such as the first scan date, scan history, domain name resolution (DNS) history, website or domain category, reverse DNS, and whois information. We further supplement each domain with their WebSense [31] category provided by the VirusTotal's record API. During the augmentation, we eliminate repeating, unresponsive or invalid URLs in each dependency chain. Thus, we collect the above metadata for each second level domain in our dataset. This results in a final sample of 196,940 first-party websites, and 68,828 third-party domains.

[1]We select the top 200K as this gives us broad coverage of globally popular websites, whilst also remaining tractable for our subsequent data enrichment activities.

Table 1: Overview of the Dataset for different ranges of Alexa's ranking. The rows indicate the proportion of Alexa's Top-X websites that explicitly and implicitly trust at least one third-party (*i*) resource (of any type); and (*ii*) JavaScript.

	Alexa Rank					
	1-200K	1-10K	190-200K	10-50K	50-100K	100-200K
F.-Parties that trust:						
All Resources:						
Explicit (Lvl. 1)	95%	95%	95%	94%	95%	95%
Implicit (Lvl. ≥ 2)	49.7%	55.1%	47.9%	51.8%	50.23%	48%
JavaScript:						
Explicit	91%	92%	91%	91%	91%	90%
Implicit	49.5%	55%	47.8%	51.69%	50%	47.8%



(a) (b)

Figure 1: (a) CDF of dependency chain lengths (broken down into categories of first-party websites); and (b) distribution of third-party websites across various categories and levels.

3 EXPLORING THE CHAINS

We begin by exploring the presence and usage of implicit trust chains. We first confirm if websites do, indeed, rely on implicit trust and then explore how these chains are used.

3.1 Do websites rely on implicit trust?

Overall, the Top-200k dataset makes 11,287,230 calls to 6,806,494 unique external resources, with a median of 27 external resources per first-party website. To dissect this, Table 1 presents the percentage of webpages in each Alexa range that load explicitly and implicitly trusted third-parties. Confirming prior studies [4, 20], it shows that 95% of websites import external resources, with 91% importing externally hosted JavaScript. More important is that observation that around 50% of the websites *do* rely on implicit trust chains, *i.e.,* they allow third-parties to load further third-parties on their behalf. The propensity to form dependency chains is marginally higher in more popular websites; for example, 55% in the Alexa top 10K have dependency chains compared to 48% in the bottom 10K (*i.e.,* rank 190-200K). In other words, more popular websites tend to rely more on implicitly trusted third-parties.

These implicitly trusted third-parties appear at various positions in the dependency chain. Intuitively, long chains are undesirable as they typically have a deleterious impact on page load times [30] and increase attacks surface. Figure 1a presents the CDF of chain length for all first-party websites. For context, websites are separated into their sub-categories.[2] It shows that 80% of the first-party websites create chains of trust of length 3 or below. However, there is also a small minority that dramatically exceed this chain length:

[2]We only include the most popular categories.

Table 2: Breakdown of resource types requested by the Top-200K websites across each level in the dependency chain.

Lev.	Total	Image	JS	Data	Font/CSS	Uncat.
1	9,212,245	34.4%	30.6%	16.0%	7.8%	11.3%
2	1,566,841	48.8%	16.7%	11.7%	3.3%	19.4%
3	405,390	45.0%	12.3%	11.1%	1.3%	30.2%
4	78,107	41.8%	18.4%	8.0%	8.1%	23.6%
5	14,413	40.6%	18.0%	12.8%	2.0%	26.4%
≥6	10,208	36.6%	12.3%	13.0%	1.2%	36.8%

we find that all website categories import ≈2% of their external resources from level 3 and above. In the most extreme case, we see `rg.ru` (news) with a chain containing 38 levels, consisting of mutual calls between `adriver.ru` (ad provider) and `admelon.ru` (IT website). Other notable examples include `thecrimson.com.bg` (Harvard's student newspaper), `argumenti.ru` (news), `mundomax.com` (IT news), `lifestyle.bg` (entertainment), which have a maximum dependency level of 15. We argue that these complex configurations make it extremely difficult to reliably audit such websites, as a first-party cannot be assured of which objects are later loaded.

3.2 What objects exist in the chain?

The previous section has confirmed that a notable fraction of websites create dependency chains with (up to) tens of levels. We next inspect the types of resources imported within these dependency chains. We classify resources into four main types: Image, JavaScript, Data (consisting of HTML, JSON, XML, plain text files), and CSS/Fonts. Table 2 presents the volume of each resource type imported at each level in the trust chain. We observe that the make-up of resources varies dramatically based on the level in the dependency chain. For example, the fraction of images imported tend to increase — this is largely because third-parties are in-turn loading images (*e.g.,* for adverts). In contrast, the fraction of JavaScript decreases as the level in the dependency chain increases: 30.6% of resources at level 1 are JavaScript compared to just 12.3% at level 3. This trend is caused by the fact that new levels are typically created by JavaScript execution (thus, by definition, the fraction of JavaScript must deplete along the chain). However, it remains at a level that should be of concern to web engineers as this confirms a significant fraction of JavaScript code is loaded from potentially unknown implicitly trusted domains.

To build on this, we also inspect the *categories* of third-party domains hosting these resources. Figure 1b presents the make-up of third-party categories at each level in the chain. It is clear that, across all levels, advertisement domains make up the bulk of third-parties. We also notice other highly demanded third-party categories such as search engines, Business and IT. These are led by well known providers, *e.g.,* `google-analytics.com` (web-analytics[3]) is on 68.3% of pages. The figure also reveals that the distributions of categories vary across each dependency level. For example, 23.1% of all loaded resources at level 1 come from advertisement domains, 37.3% at level 2, 46.2% at level 3, *i.e.,* the proportion increases across dependency levels. In contrast, social network third-parties (*e.g.,* Facebook) are mostly presented at level 1 (9.58%) and 2 (13.57%) with a significant drop at level 3. The dominance of advertisements is not, however, caused by a plethora of ad domains: there are far fewer ad domains than business or IT (see Table 3). Instead, it is driven by the large number of requests to advertisements: Even though ad domains only make-up 1.5% of third-parties, they generate 25% of resource requests. Importantly, these popular providers can trigger further dependencies; for example, `doubleclick.com` imports 16% of its resources from further implicitly trusted third-party websites. This makes such domains an ideal propagator of malicious resources for any other domains having implicit trust in it.

4 FINDING SUSPICIOUS CHAINS

We next study the existence of *suspicious* third-parties, which could lead to abuse of the implicit trust. Within this section we use the term *suspicious* (to be more generic than malicious) because VirusTotal covers activities ranging from low-risk (*e.g.,* sharing private data over unencrypted channels) to high-risk (malware).

4.1 Do chains contain suspicious parties?

First, we inspect the fraction of third-party domains that trigger a warning by VirusTotal. From our third-party domains, 2.5% have a VTscore of 1 or above, *i.e.,* at least one virus checker classifies the domain as suspicious. If one treats the VTscore as a ground truth, this confirms that popular websites *do* load content from suspicious third-parties via their chains of trust. However, we are reticent to rely on VTscore ≥ 1, as this indicates the remaining 67 virus checkers did not flag the domain.[4] Thus, we start by inspecting the presence of suspicious third-parties using a range of thresholds.

Table 3 shows the fraction of third-parties that are classified as suspicious using several VTscore thresholds. For context, we separate third-parties into their respective categories (using WebSense). The table confirms that a noticeable subset of suspicious third-party domains exist; for example, if we classify any resource with a VTscore ≥ 10 as suspicious, we find that 1.2% of third-party domains are classified as suspicious with 6.2% of all resource calls in our dataset going to these third-parties. Notably this only drops marginally (to 5.7%) with a *very* conservative VTscore of ≥ 40. We observe similar results when considering thresholds in the [3..50] range. This confirms, with a high certainty, that approximately 6% of resource calls in the dependency chains are towards domains that engage in suspicious activity (further detailed in [12]). We will conservatively refer to domains with a VTscore ≥ 10 as suspicious in the rest of this analysis.

We also check if dependency chains contain suspicious JavaScript resources. We focus on JavaScript as active content poses greater threats, *e.g.,* cross-site scripting (XSS) and advanced phishing [20]. Table 4 shows the top first-party domains, ranked according to the number of unique suspicious third-parties in their chain of dependency. We note that the top ranked (most vulnerable) first-party domains belong to various categories such as Content Sharing, News, or IT. This indicates that there is no one category of domains that inherits suspicious JavaScript. However, we note that

[3]Grouped as in business category as per VirusTotal reports.

[4]Diversity is likely caused by the virus databases used by the different virus checkers [2]

Table 3: Overview of suspicious third-parties in each category. Col.2-4: number of third-party websites in different categories, the number of resource calls to resources, and the proportion of calls to suspicious JavaScript. Col.5-9: Fraction of third-party domains classified as suspicious (*Num.*), and fraction of resource calls classified as suspicious (*Vol.*), across various VTscores (i.e., ≥ 3 and ≥ 55).

				VTScore ≥ 3		VTScore ≥ 10		VTScore ≥ 20		VTScore ≥ 40		VTScore ≥ 55	
Category	Third-Parties	Total Calls	Suspicious JS	Num.	Vol.	Num.	Vol.	Num.	Vol.	Num.	Vol.	Num.	Vol.
All	68,828	11,287,204	270,758 (2.4%)	1.6%	6.4%	1.2%	6.2%	1.0%	6.1%	0.6%	5.7%	≤ 0.1%	≤ 0.1%
Business	6,786	1,924,591	184,360 (9.6%)	1.5%	21.5%	1.1%	21.5%	1.0%	21.4%	0.5%	20.6%	0%	0%
Ads	1,017	2,870,482	7,924 (0.3%)	3.5%	0.1%	3.3%	0.1%	2.9%	0.1%	1.6%	≤ 0.1%	0%	0%
IT	8,619	1,646,287	10,547 (0.6%)	2.2%	3.8%	1.5%	3.6%	1.2%	3.5%	0.6%	3.0%	≤ 0.1%	≤ 0.1%
Other	52,406	4,845,844	67,927 (1.4%)	1.4%	4.6%	1.1	4.3%	0.9%	4.2%	0.6%	3.8%	≤ 0.1%	≤ 0.1%

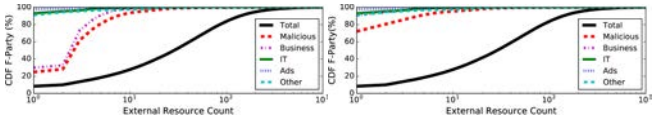
first party websites categorized as "Business" represent the majority of most exposed domains at Level ≥2, with 16% of the total number of first-party domains implicitly trusting suspicious JavaScript belonging to the Business Category, with distant second being the "News & Media" Category and third the "Adult" category. The number of suspicious JavaScript codes loaded by these first-party domains ranges from 4 to 31. We note the extreme case of `amateur-fc2.com` website *implicitly* importing 31 unique suspicious JavaScript programs from 4 unique suspicious domains. Moreover, we observe at most 7 unique third-parties (combining both explicit and implicit level) that is a cause of suspicious JavaScripts in first-parties. This happens for `privet-rostov.ru` domain, having third-party domains such as `charter.com, vk.com. rambler.ru, doubleclick.net, dx.com, cdn.adlegend.com, syncsw.pool.datamind.ru`.

Table 4: Top 5 most exposed first-party domains (with VTscore ≥ 10) ranked by the number of unique suspicious domains.

	Unique Suspicious Domains at Level = 1					
#	First-party Domain	Alexa Rank	# Mal. JSes	Unique Susp. Doms.	Category	Chain Len.
1	theinscribermag.com	46,242	6	5	Blogs	5
2	skynet-system.com.ua	192,549	6	5	Busin.	4
3	nodwick.com	194,823	13	4	Enter.	4
4	iphones.ru	12,045	4	4	IT	4
5	privet-rostov.ru	193,024	6	4	LifeStyle	4
	Unique Suspicious Domains at Level ≥ 2					
1	traffic2bitcoin.com	33,513	6	5	Games	7
2	radionetplus.ru	166,003	8	4	SW Download	6
3	studiofow.tumblr.com	85,483	11	4	Adult	4
4	amateur-fc2.com	52,556	31	4	Adult	5
5	fasttorrent.ru	24,250	9	4	File Sharing	7

4.2 How widespread are suspicious parties?

We next inspect how widespread these suspicious third-parties are at each position in the dependency chain, by inspecting how many websites utilize them. Figure 2a displays the cumulative distribution (CDF) of resource calls to third-parties made by each first-party webpage in our dataset. Within the figure, we decompose the third-party resources into various groups (including total *vs.* suspicious). The figure reveals that suspicious parties within the dependency chains are commonplace: 24.8% of all first-party webpages contain at least 3 third-parties classified as suspicious in their dependency chain. Remarkably, 73% of first-party websites load resources from



(a) All websites (b) Without google-analytics

Figure 2: CDF of resources loaded per-website from various categories of third-parties.

third-parties at least once. Hence, even though only 1.6% of third-party domains are classified as suspicious, their reach covers nearly three quarters of websites (indirectly via implicit trust).

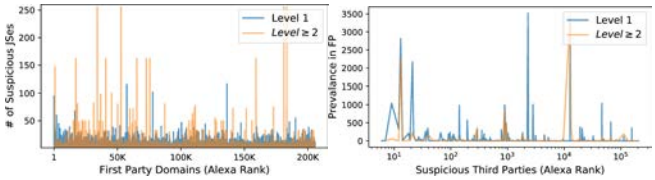
The above is demonstrated in Table 5, which presents the top 10 most frequently encountered suspicious third-party domains that are providing suspicious JavaScript. It can be seen that popular third-party domains exist across *many* first-party sites. The top 20% of third-party domains cover 86% (9,650,582) of all resource calls. Closer inspection shows that it is driven by one prominent third-party: `google-analytics.com`. At first, we thought that this was an error, however, during the measurement period `google-analytics.com` obtained a VTscore of 51, suggesting a high degree of certainty. This was actually caused by `google-analytics.com` loading another third-party, `sf-helper.net`, which is known to distribute adwares and spywares. It is unclear why Google was performing this. We therefore repeated these checks in October 2018, to confirm that this activity has ceased, and `sf-helper.net` is no longer loaded. To understand the impact its new de-classification has, Figure 2b shows the distribution of resource calls to third-party categories when `google-analytics.com` is benign. This reduces the number of first-party websites exposed to suspicious resources by 63%. This highlights effectively the impact of high centrality third-parties being permitted to load further resources: the infection of just one can immediately effect a significant fraction of websites.

4.3 How popular are suspicious third-parties?

We next test if widespread suspicious third-parties are also highly ranked within Alexa. We treat this as a proxy for global popularity. Beyond `google-analytics.com` we find several other suspicious third-party domains from the Top 100 Alexa ranking. For-instance, `vk.com`, a social network website mostly geared toward East European countries has been used by 3,094 first-parties and is ranked 13 by Alexa. This website is found to be one of the most prevalent suspicious third-party domains at both level 1 and levels ≥ 2. An obvious reason for this domain's presence is because of other infected (malware-based) apps that try to authenticate users from

Table 5: Top 5 most prevalent suspicious third-party domains (with VTscore ≥ 10) on level 1 (explicit trust) and beyond (implicit trust) providing resources to first-parties. #FP refers to the number of First-party domains having the corresponding suspicious third-party domain in their chain of dependency.

#	Third-party Domain	Alexa Rank	# FP	Category
Prevalence of Third-parties at Level = 1				
1	google-analytics.com	13,200	43,156	Business (Web Analytics)
2	gravater.com	2,292	3,520	IT
3	charter.com	12,714	3,425	Business
4	vk.com	13	2,815	Social Network
5	statcounter.com	2,265	2,327	Business (Web Analytics)
Prevalence of Third-parties at Level ≥ 2				
1	charter.com	12,714	3,452	Business
2	vk.com	13	2,290	Social Network
3	livechatinc.com	888	851	Web Chat
4	onesignal.com	950	467	Business
5	rambler.ru	291	370	SearchEngine

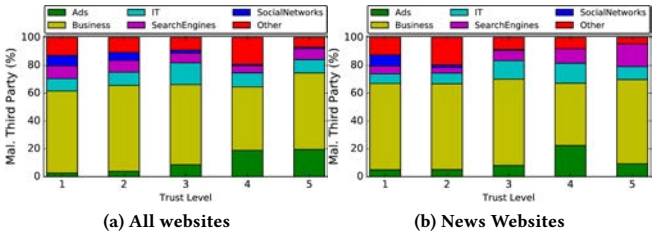


(a) (b)

Figure 3: Figure (a) depicts the number of suspicious JavaScript content imported (explicitly and implicitly) by first-party domains shown according to their Alexa ranking; and (b) shows the number of impacted first-party domains as function of the ranking of domains of Suspicious JavaScript.

such domains [24]. Other websites such as `statcounter.com` or `gravater.com` are also among the most prevalent third party domains in level 1. These websites were reported to contain malware in their Javascript codes [3]. For instance, users in statcounter forums reported it as malicious because a Javascript running its website redirects users to a malware website `gocloudly.com`, and forces users to click the button [5].

More generally, we observe the presence of a wide range of Alexa ranks in the list of most prevalent domains at levels ≥ 2. In Figure 3a, we show the number of suspicious JavaScripts imported by the first-party domains (Y-axis) according to their Alexa rank (X-axis). Overall, first-party domains import a larger number of suspicious third-party JavaScript codes at levels ≥ 2, however, the first-party domains seem to be equally vulnerable to the implicit import of suspicious content regardless of their rank. There are exceptions though, signified by the peaks in the number of suspicious JavaScripts — these are near exclusively driven by a large number of ≥level-2 scripts (implicit trust). We also encounter an interesting case, which we exclude from the graphs for readability purposes: The first-party domain `kikar.co.il` imports 2,592 JavaScript codes originating from the third-party `hwcdn.net`, a well-known browser hijacker that has been reported to force users to visit spam pages [29]. The VirusTotal API indicates a VTscore of 22 for this suspicious domain. We also note that 35 other first-party



(a) All websites (b) News Websites

Figure 4: Distribution of calls to suspicious third-party websites per category at each level, for all top-200K websites (Figure 4a) and most vulnerable first-party category (Figures 4b).

domains have this domain in their chain of dependency. Again, this highlights the risk of implicit trust.

In Figure 3b we show the number of impacted first-party domains as a function of the Alexa Rank of suspicious third-party domains (limited to a maximum Alexa Rank of 1 million) — note the log scale of X-axis. We observe that some very prevalent third-parties have a high Alexa ranking (even excluding `google-analytics.com`). For instance, note a spike around the 2000 rank, which reaches a prevalence of 3500 first-party domains at level 1. This spike is caused by `gravatar.com`, propagating suspicious Javascripts. This supports our statements earlier (from Table 5) where `gravatar.com` is ranked second top most suspicious domain. Similarly, a spike around 10K rank indicates the presence of `charter.com` both at level 1 and 2 respectively. These findings demonstrate the wide variety of third-party suspicious JavaScript content loaded from various, not necessarily "obscure", third-party domains.

4.4 At which level do suspicious third-parties occur?

Next, we inspect the location(s) in the dependency chain where these suspicious third-parties are situated. This is vital, as implicitly trusted (≥level 2) resources are far more difficult for a first-party administrator to remove. Table 6 presents the proportion of websites that import at least one resource with a VTscore ≥ 10. We separate resources into their level in the dependency chain. The majority of resources classified as suspicious are located at level 1 in the dependency chain (*i.e.,* they are explicitly trusted by the first-party). 73% of websites containing suspicious third-parties are "infected" via level 1. This suggests that these website operators are not entirely diligent in monitoring their third-party resources. This might include websites that purposefully utilize such third-parties [8]. Perhaps more important, the above leaves a significant minority of suspicious resources imported via *implicit* trust (*i.e.,* level ≥ 2). In these cases, the first-party is potentially unaware of their presence. The most vulnerable category is news: over 15% of news sites import *implicitly* trusted resources from level 2 with a VTscore ≥ 10. Notably, among the 56 news websites importing suspicious JavaScript resources from trust level 3 and deeper, we find 52 loading advertisements from `adadvisor.net`. One possible reason is that ad-networks could be infected or victimized with malware to perform malvertising [21, 28].

Table 6: Proportion of top-200K websites importing resources classified as suspicious (with VTscore ≥ 10) at each level.

Lv.	All		News		Sports		Entertainment		Forums	
	All	JS	All	JS	All	JS	All	JS	All	JS
1	61.30%	57.70%	75.40%	73.50%	75.70%	73.20%	69.30%	65.60%	67.40%	65.50%
2	5.20%	2.20%	13.40%	5.60%	11.10%	3.70%	8.60%	4.10%	9.10%	4.05%
3	1.30%	0.18%	2.90%	0.45%	3.60%	0.28%	2.70%	0.30%	3.20%	0.15%
4	0.22%	≤ 0.1%	0.64%	0.08%	0.80%	≤ 0.1%	0.70%	0.08%	0.60%	0.00%
≥ 5	≤ 0.1%	0	0.002	≤ 0.1%	0.001%	≤ 0.1%	0.002%	≤ 0.1%	≤0.001%	0.00%

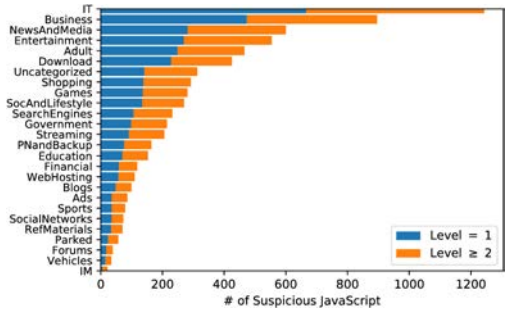


Figure 5: Breakdown of JavaScript resources based on category of domain. Uncategorized category includes domain such as `newmyvideolink.xyz` **and** `cooster.ru`

Similar, albeit less extreme, observations can be made across Sports, Entertainment, and Forum websites. Briefly, Figure 4 displays the categories of (suspicious) third-parties loaded at each level in the dependency chain — it can be seen that the majority are classified as business. This is, again, because of several major providers classified as suspicious such as `convexity.net` and `charter.com`. Furthermore, it can be seen that the fraction of advertisement resources also increases with the number of levels due to the loading of further resources (*e.g.,* images).

Figure 5 also presents the breakdown of the domain categories specifically for suspicious JavaScript resources. Clear trends can be seen, with IT (e.g., `dynaquestpc.com`), Business (*e.g.,*`vindale.com`), News and Media (e.g., `therealnews.com`), and Entertainment (*e.g.,*`youwatchfilm.net`) dominating. Clearly, suspicious JavaScript resources cover a broad spectrum of activities. Interestingly, we observed that 70% and 67%, respectively, of Business (Web analytics) and Ads JavaScripts are loaded from level ≥ 2 in contrast to 17% and 31% of JavaScripts of Government and Shopping loaded at level 1. We next strive to quantify the level of suspicion raised by each of these JavaScripts. Intuitively, those with higher VTscores represent a higher threat as defined by the 68 AV tools used by VirusTotal. Hence, Figure 6 presents the cumulative distribution of the VTscores for all JavaScript resources loaded with VTscore > 0. We separate the JavaScripts into their location in the dependency chain. Clear difference can be observed, with level 2 obtaining the highest VTscore (median 28). In fact, 78% of the suspicious JavaScript resources loaded on trust level 2 have a VTscore > 52 (indicating *very* high confidence). It is also worth noting that the VTscore for resources loaded further down the dependency chain is lower (*e.g.,* level 4). For example, 80% of level 4 resources receive a VTscore below 5. This suggests that the activity of these resources is more contentious, with a smaller number of AV tools reaching consensus. It is impossible to state the reason for this, hence in our extended work [12] we analyze the dynamic activities of these JavaScript content.

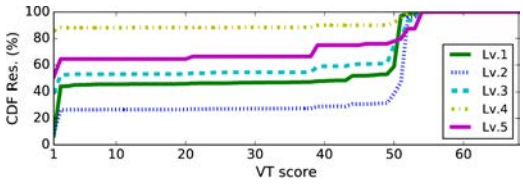


Figure 6: CDF of suspicious JavaScripts (VTscores ≥ 10) at different levels in the chain.

5 RELATED WORK

There has been a wealth of research into the utilisation and exploitation of third-parties and JavaScript libraries [9, 10, 20, 22, 25, 26]. Our work differs quite substantially from these in that we are not interested in the JavaScript code itself, nor the simple presence of third-party domains in a webpage. Instead, we are interested in *how* third-parties are loaded, and their use of "implicit" trust (*i.e.,* dependency chains). In contrast to our work, these prior studies ignore the presence of dependency chains and treat all third-parties as "equal", regardless of where they are loaded in the dependency chain. Closer to our own work is Bashir *et al.* [1], who studied websites' resource inclusion trees and analyzed retargeted ads using crowdsourcing. This allowed them to identify and classify ad domains, as well as predominant cookie matching partners in the ad exchange environment. Our study is far broader, and sheds light on dependency chains across many different types of websites rather than simply inspecting advertisements. More related is Kumar *et al.* [18], who recently characterized websites' resource dependencies on third-party services. In-line with our work, they found that dependency chains are widespread. This means, for example, that 55% of websites are prevented from fully migrating to HTTPS by their dependencies. Their focus was not, however, on identifying suspicious or malicious activities. To the best of our knowledge, this paper is the first study to analyze the role of implicit trust from a security perspective to better understand the role of dependency chains in loading suspicious third-party content.

6 CONCLUDING REMARKS

This paper has explored dependency chains in the web ecosystem. Inspired by the lack of prior work focusing on how resources are loaded, we found that over 40% of websites *do* rely on implicit trust. Although the majority (84.91%) of websites have short chains, we found first-party websites with chains exceeding 30 levels. Of course, the most commonly *implicitly* trusted third-parties are well known operators (*e.g.,* `doubleclick.net`), but we also observed various less known implicit third-parties. We hypothesized that this might create notable attack surfaces. To confirm this, we classified the third-parties using VirusTotal to find that 1.2% of third-parties are classified as potentially malicious. These resources have remarkable reach — largely driven by the presence of highly central third-parties, *e.g.,* `google-analytics.com`. With this in mind, in our extended work [12], we perform sandbox experiments on the

suspicious JavaScript to understand their actions. We witness extensive download activities, much of which consisted of *dropper* files and malware, which is installed on the machine. It was particularly worrying to see that JavaScript resources loaded at level ≥ 2 in the dependency chain tend to have more aggressive properties, particularly as exhibited by their higher VTscore. This is only the first step in our research agenda. We wish to perform longitudinal measurements to understand how these metrics of maliciousness evolve over time. We are particularly interested in understanding the (potentially) ephemeral nature of threats.

REFERENCES

[1] M. A. Bashir, S. Arshad, W. Roebertson, and C. Wilson. Tracing information flows between ad exchanges using retargeted ads. In *USENIX Security Symposium*, 2016.

[2] J. Canto, M. Dacier, E. Kirda, and C. Leita. Large scale malware collection: lessons learned. In *IEEE SRDS Workshop on Sharing Field Data and Experiment Measurements on Resilience of Distributed Computing Systems*. Citeseer, 2008.

[3] I. X. Exchange. Statcounter session hijack. https://exchange.xforce.ibmcloud.com/vulnerabilities/20506, 2005.

[4] M. Falahrastegar, H. Haddadi, S. Uhlig, and R. Mortier. Anatomy of the third-party web tracking ecosystem. *Traffic Measurements Analysis Workshop (TMA)*, 2014.

[5] S. C. Forum. http://www.statcounter.com/ counter/counter.js has malware inside it ! https://forum.statcounter.com/threads/http-www-statcounter-com-counter-counter-js-has-malware-inside-it.43792/, 2016.

[6] Google. Headless chromium. https://chromium.googlesource.com/chromium/src/+/lkgr/headless/README.md, 2018.

[7] D. Ibosiola, I. Castro, G. Stringhini, S. Uhlig, and G. Tyson. Who watches the watchmen: Exploring complaints on the web. In *Web Conference*, 2019.

[8] D. Ibosiola, B. Steer, A. Garcia-Recuero, G. Stringhini, S. Uhlig, and G. Tyson. Movie pirates of the caribbean: Exploring illegal streaming cyberlockers. *International AAAI Conference on Web and Social Media (ICWSM)*, 2018.

[9] M. Ikram, H. Asghar, M. A. Kaafar, and A. Mahanti. On the intrusiveness of javascript on the web. In *CoNEXT Student Workshop*, 2014.

[10] M. Ikram, H. J. Asghar, M. A. Kaafar, A. Mahanti, and B. Krishnamurthy. Towards seamless tracking-free web: Improved detection of trackers via one-class learning. *Proceedings on Privacy Enhancing Technologies*, 2017(1):79–99, 2017.

[11] M. Ikram and M. A. Kaafar. A first look at mobile ad-blocking apps. In *Network Computing and Applications (NCA), 2017 IEEE 16th International Symposium on*, pages 1–8. IEEE, 2017.

[12] M. Ikram, R. Masood, G. Tyson, M. A. Kaafar, N. Loizon, and R. Ensafi. The chain of implicit trust: An analysis of the web third-party resources loading. *arXiv preprint arXiv:1901.07699*, 2019.

[13] M. Ikram, N. Vallina-Rodriguez, S. Seneviratne, M. A. Kaafar, and V. Paxson. An analysis of the privacy and security risks of android vpn permission-enabled apps. In *IMC*, 2016.

[14] V. Inc. Virustotal public api. https://www.virustotal.com/en/documentation/public-api/, 2017.

[15] S. Jerome. Large angler malvertising campaign hits top publishers. https://blog.malwarebytes.com/threat-analysis/2016/03/large-angler-malvertising-campaign-hits-top-publishers/. Accessed: 2018-09-18.

[16] A. Kantchelian, M. C. Tschantz, S. Afroz, B. Miller, V. Shankar, R. Bachwani, A. D. Joseph, and J. D. Tygar. Better malware ground truth: Techniques for weighting anti-virus vendor labels. In *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, pages 45–56. ACM, 2015.

[17] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge, and E. Kirda. Cutting the gordian knot: A look under the hood of ransomware attacks. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 3–24. Springer, 2015.

[18] D. Kumar, Z. Ma, A. Mirian, J. Mason, J. A. Halderman, and M. Bailey. Security Challenges in an Increasingly Tangled Web. In *Proceedings of the 2017 World Wide Web Conference on World Wide Web*, 2017.

[19] J. Kurkowski. Accurately separate the TLD from the registered domain and subdomains of a url, using the public suffix list. https://github.com/john-kurkowski/tldextract, 2018.

[20] T. Lauinger, A. Chaabane, S. Arshad, W. Robertson, C. Wilson, and E. Kirda. Thou shalt not depend on me: Analysing the use of outdated javascript libraries on the web. In *NDSS*, 2017.

[21] Z. Li, K. Zhang, Y. Xie, F. Yu, and X. Wang. Knowing your enemy: understanding and detecting malicious web advertising. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 674–686. ACM, 2012.

[22] N. Nikiforakis, L. Invernizzi, A. Kapravelos, S. Van Acker, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna. You are what you include: Large-scale evaluation of remote javascript inclusions. In *CCS*, 2012.

[23] G. Pellegrino, C. Rossow, F. J. Ryba, T. C. Schmidt, and M. WÃđhlisch. Cashing out the great cannon? on browser-based ddos attacks and economics. In *USENIX*, 2015.

[24] B. Popa. 85 infected android apps stealing social network passwords found on play store. https://news.softpedia.com/news/85-infected-android-apps-stealing-social-network-passwords-found-on-play- store-518984.shtml, 2017.

[25] J. Su, Z. Li, S. Grumbach, M. Ikram, K. Salamatian, and G. Xie. Web tracking cartography with dns records. In *IEEE 37th International Performance Computing and Communications Conference (IPCC)*, 2018.

[26] J. Su, Z. Li, S. Grumbach, M. Ikram, K. Salamatian, and G. Xie. A cartography of web tracking using dns records. *Computer Communications*, 134:83 – 95, 2019.

[27] M. P. Suffix. View the public suffix list. https://publicsuffix.org/list/, 2018.

[28] A. VANCE. Times web ads show security breach. https://www.nytimes.com/2009/09/15/technology/internet/15adco.html, 2009.

[29] Q. R. Virus. How do i remove hwcdn.net from my pc. https://quickremovevirus.com/how-do-i-remove-hwcdn-net-from-my-pc/, 2017.

[30] X. S. Wang, A. Balasubramanian, A. Krishnamurthy, and D. Wetherall. Demystify page load performance with wprof. In *Proc. of the USENIX conference on Networked Systems Design and Implementation (NSDI)*, 2013.

[31] Websense. Real-time threat analysis with csi: Ace insight. https://csi.websense.com/, 2018.

Efficient Interaction-based Neural Ranking with Locality Sensitive Hashing

Shiyu Ji
Department of Computer Science,
University of California
Santa Barbara, California

Jinjin Shao
Department of Computer Science,
University of California
Santa Barbara, California

Tao Yang
Department of Computer Science,
University of California
Santa Barbara, California

ABSTRACT

Interaction-based neural ranking has been shown to be effective for document search using distributed word representations. However the time or space required is very expensive for online query processing with neural ranking. This paper investigates fast approximation of three interaction-based neural ranking algorithms using Locality Sensitive Hashing (LSH). It accelerates query-document interaction computation by using a runtime cache with precomputed term vectors, and speeds up kernel calculation by taking advantages of limited integer similarity values. This paper presents the design choices with cost analysis, and an evaluation that assesses efficiency benefits and relevance tradeoffs for the tested datasets.

ACM Reference Format:
Shiyu Ji, Jinjin Shao, and Tao Yang. 2019. Efficient Interaction-based Neural Ranking with Locality Sensitive Hashing. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.4090049

1 INTRODUCTION AND RELATED WORK

Neural networks have been applied in information retrieval and text mining tasks (e.g. [5, 9, 20, 23, 24, 33–36]). In general, exploiting neural representation of words adds significant rank scoring computation cost during query processing. While parallel computing with a large number of CPUs or GPUs can accelerate, this paper studies how to speed up neural rank scoring with an algorithmic approximation to reach a reasonable query response time without resorting to expensive hardware. This is important for a large online query processing website which needs to dedicate more hardware resource for handling high query traffic. Fast scoring also opens opportunities to integrate a more complex neural scheme.

The previous research on neural ranking falls into the following two categories: interaction-based or representation-based models [23]. The earlier work has focused on the representation based models [11, 27] where each document and a query are separately represented as vectors through a sequence of neural computations and the final ranking is based on the similarity of the two representative vectors. The recent studies have focused on interaction-based neural ranking models [5, 9, 34] where the word or term-level similarity of a query and a document is explored first based on their embedding vectors before applying an additional sequence of neural computation. These studies have shown their interaction based models outperform the earlier representation-based models and thus our paper is focused on the acceleration of the three interaction-based models, more specifically DRMM [9], KNRM [34], and CONV-KNRM [5]. We find that the query processing time cost in the above interaction-based models for computing query-document interaction and deriving kernel values is dominating. For CONV-KNRM, the time to prepare term vectors is very expensive.

The contribution of this paper is an LSH approximation of these three neural methods with fast histogram-based kernel calculation and term vector precomputing for a runtime cache. Another contribution is an analysis and experimental evaluation of design choices and a demonstration using more complex dual embeddings leveraging fast neural scoring. While our study is focused on the above three models, LSH approximation may be used for other interaction-based methods that employ cosine similarity (e.g. graph embedding [35, 36]) and this is a future work.

Our work is motivated by the rich history of research on approximation with hashing such as LSH [1–4, 7, 8, 13, 15, 37] and in various data-intensive applications [6, 10, 17, 18, 30]. The reason to choose the Random Hyperplane LSH [4] is that the computation with hashed embeddings can not only preserve the semantic similarity within a small error bound, but also allow us to take advantage of the binary nature of sketch vectors for designing a fast kernel calculation.

The acceleration of neural network computation has been studied for image classification with complex neural network structure (e.g. [12, 26, 28] for binarization). Their work is less applicable to our context where neural network involved is relatively simple and cost of forward neural computation in query processing is one or two orders of magnitudes smaller than interaction and kernel computation. To our knowledge, there is no research work on accelerating neural ranking models for fast online query processing.

2 INTERACTION-BASED NEURAL RANKING

Interaction-based neural ranking can be formalized as performing computation: RankingScore = $f(Ker(\vec{q} \otimes \vec{d}))$, where $\vec{q}$ and $\vec{d}$ are two sequences of embedding vectors for the query and the document respectively, $\otimes$ is the interaction operation, Ker is the operation for kernel computation, and f is the operation involving neural network computation. Embedding vectors in $\vec{q}$ and $\vec{d}$ can be term vectors for unigram/n-gram in query/document [5, 9, 34], or can be entity vectors for existing entities in query/document [35]. Those embedding vectors can be learned from one of many existing neural embedding models [19, 25] or generated by additional neural operations [5].

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.4090049

Table 1: Frequently used notations.

Notation	Definition
$\vec{q}$	Embedding vectors for query terms.
$\vec{d}$	Embedding vectors for document terms.
m	Number of words in the query.
n	Number of words in the document.
l	Dimension of pre-trained unigram word embedding vectors.
h	The largest span of n-gram considered in CONV-KNRM [5].
F	The number of filters in CONV-KNRM [5].
M	The interaction matrix between query and document.
V	Vocabulary size of unigrams in the corpus.
H	Vocabulary size of n-grams in the corpus where $n \leq h$.
N	Number of documents in the corpus.
R	Number of kernels in the neural ranking model.
μ_k	Mean of the k-th kernel in KNRM and CONV-KNRM.
σ_k	Standard deviation of the k-th kernel.
$K_k(M_i)$	The k-th kernel value for the i-th query term.
b	Number of bits in hashing footprint.

Interaction operator $\otimes$. This step outputs an interaction matrix M containing vector similarities for all pairs of one query term vector and one document term vector.

- **DRMM [9] and KNRM [34]** consider only the interaction between unigrams in the query and unigrams in the document. Each matrix element M_{ij} is the cosine similarity between the term vectors of the i-th query term and the j-th document term.
- **CONV-KNRM [5]** extends KNRM to incorporate n-grams in building a set of interaction matrices. Let h be the maximum n-gram length considered and there are h^2 interaction matrices. Each interaction matrix $M^{r,t}$ where $1 \leq r, t \leq h$ denotes a cosine similarity score matrix between query term vector modeled with r-gram embeddings and document term vectors modeled with t-gram embeddings. These n-gram embeddings are computed from convolution filters and pre-trained unigram embedding vectors. The dimension of n-gram embeddings depends on the number of filters involved (defined as F).

Since computing each interaction matrix requires a dot product computation for each matrix element, given a document with n terms, the time complexity to compute the similarities of this document for m-word query is $\Theta(nml)$ for DRMM and KNRM, and $\Theta(nmF)$ for CONV-KNRM.

Kernel computation operator Ker. Ker is used for extracting multi-level soft matching features and generating inputs for neural network computation.

- In DRMM, each interaction matrix element represents the similarity between a query term and a document term, varying from -1 to 1. The interval $[-1, 1]$ is divided into a set of non-overlapping ranges. Thus a histogram can be generated where each bucket represents one range, and the value of a bucket represents the number of similarities falling into that bucket. Different buckets contain soft matching signals of different levels, just like kernels in the Kernel Pooling [34]. Thus in this paper, for simplicity, we denote each bucket as a kernel. Calculating all kernel/bucket values requires $\Theta(nm)$ time complexity.
- In KNRM and CONV-KNRM, each kernel is defined using a distribution and computation involved is called kernel pooling. There are R kernels defined in advance and symbols μ_k and σ_k denote the mean and standard deviation of the k-th kernel respectively. Each element value of an interaction matrix contributes kernel computation. For KNRM with one interaction matrix M, the kernel value of the k-th kernel for the i-th query term is:

$$K_k(M_i) = \sum_{j=1}^{n} \exp(-\frac{(M_{ij} - \mu_k)^2}{2\sigma_k^2})$$

  where n is the number of terms in the document. The output of kernel computation is a vector of R size: $(\sum_{i=1}^{m} \log K_1(M_i), \cdots, \sum_{i=1}^{m} \log K_R(M_i))^T$. Building this kernel for each query term requires $\Theta(nmR)$ time given there are R kernels.
  In CONV-KNRM, there are h^2 interaction matrices and the total cost of this kernel computation is $\Theta(nmRh^2)$. This expression does not include the cost of preparing term vectors and we discuss this issue further in Section 3.3.

Forward neural network computation f. This step uses a forward neural network to compute the final ranking score based on a vector of kernel values computed in the previous step.

- In DRMM, a multilayer perception network with one hidden layer of a constant size is used and the logarithms of the kernel/bucket values of the similarity ranges for the i-th query terms form an input vector. The final ranking score is a weighted linear combination of the neural network outputs of all query terms. The time cost of computing is in $\Theta(mR)$.
- In KNRM, a simple single-layer network is used and the input vector is a vector of size R computed in the previous step. The time cost for neural network computation is in $\Theta(R)$.
- In CONV-KNRM that extends KNRM, the formula of forward neural computation in this step is about the same as KNRM except that the output of the previous step produces a vector of dimension Rh^2 given h^2 interaction matrices and this vector is the input of this step. Thus the time cost for forward neural network computation is in $\Theta(Rh^2)$.

3 NEURAL RANKING WITH LSH

We summarize the time cost of above interaction-based neural ranking steps for query processing in Table 2. This section proposes three techniques to bring down time cost of query processing: approximation of term vectors with LSH in Section 3.1, fast kernel computing in Section 3.2, and precomputing of term vectors for CONV-KNRM in Section 3.3. The query-processing time cost after using our three techniques is listed in Table 2 also.

3.1 LSH for fast interaction computation

We adopt a hyperplane LSH [4], motivated by the previous work in clustering and nearest neighbor search [17, 30] that adopts such a scheme also. In [17, 30], multiple LSH signatures are produced for each document and the probability of similar documents falls into the one of LSH buckets is computable. In our context, we only use one LSH signature to approximate one term vector.

Table 2: Time complexity in Θ notion for ranking one single document with and without LSH.

Models	DRMM		KNRM		CONV-KNRM without precomputing		CONV-KNRM with precomputing	
With LSH?	No	Yes	No	Yes	No	Yes	No	Yes
Vectors	-	-	-	-	$(n+m)h^2lF$	$(n+m)(hbF+h^2lF)$	h^2mlF	$(mhbF+h^2mlF)$
Interaction	nml	nm	nml	nm	h^2nmF	h^2nm	h^2nmF	h^2nm
Kernel	nm	nm	nmR	$nm+bmR$	h^2nmR	$h^2(nm+bmR)$	h^2nmR	$h^2(nm+bmR)$
Neural	mR	mR	R	R	Rh^2	Rh^2	Rh^2	Rh^2

Table 3: Dominating factor of storage space cost in bytes with and without LSH using a reference or value based method.

Models	DRMM		KNRM		CONV-KNRM without precomputing		CONV-KNRM with precomputing	
With LSH?	No	Yes	No	Yes	No	Yes	No	Yes
Reference	$4nN+4Vl$	$4nN+Vb/8$	$4nN+4Vl$	$4nN+Vb/8$	$4nN+4Vl$	$4nN+4Vl$	$4hnN+4HF$	$4hnN+Hb/8+4Vl$
Value	$4nNl$	$nNb/8$	$4nNl$	$nNb/8$	$4nNl+4Vl$	$4nNl+4Vl$	$4hnFN+4HF$	$hnbN/8+4Vl$

Formally, for each embedding vector, we generate b-bit footprint

$$\text{LSH}(v)[i] = \begin{cases} 1 & r_i \cdot v > 0, \\ 0 & r_i \cdot v \le 0. \end{cases}$$

Here $\text{LSH}(v)[i]$ denotes the i-th bit of the LSH footprint of the vector v, and each vector r_i is independently sampled by a multivariate normal distribution from the vector space of the embeddings. The time complexity to compute the above LSH for an embedding is $\Theta(bl)$, where l is the embedding length. In particular, for CONV-KNRM we need to compute LSH for the n-gram vectors in the query, which needs $\Theta(mhbF)$, since query of m words has mh n-grams (s.t. $n \le h$), and the length of each n-gram vector is F.

Then we estimate the angle between two embeddings by the Hamming distance between their LSH footprints. Formally, given any two embeddings x and y with angle θ between them, estimator $\hat{\theta}$ is defined as $\hat{\theta} = \frac{\pi}{b} D(\text{LSH}(x), \text{LSH}(y))$, where $D(\cdot,\cdot)$ denotes Hamming distance (Eq. (2) in [30]). Modern hardware is fast at finding Hamming distance between two words, or equivalently, counting the set bits of the XORed words. The time complexity of deriving the Hamming distance that approximates the cosine similarity is $\Theta(nm)$ for one n-word document and m-word query.

The above approximation error is estimated as follows. Given any two embedding vectors u, v with angle θ between them, the probability that both hyperplane LSH footprints have the identical i-th bit is $\Pr[\text{LSH}(u)[i] = \text{LSH}(v)[i]] = 1 - \theta/\pi$ (also in [4, 17, 30]). Since each bit in the footprint is sampled independently, the Hamming distance between the LSH footprints is a random variable of Binomial distribution with $p = \theta/\pi$. Let b be the number of bits in the LSH footprint. Then the mean and variance of the Hamming distance are $\mathbb{E}[D(\text{LSH}(u), \text{LSH}(v))] = b\frac{\theta}{\pi}$, $\text{Var}[D(\text{LSH}(u), \text{LSH}(v))] = b\frac{\theta}{\pi}\left(1 - \frac{\theta}{\pi}\right)$. For the above angle estimator between u and v, $\mathbb{E}[\hat{\theta}] = \frac{\pi}{b}\mathbb{E}[D(\text{LSH}(u), \text{LSH}(v))] = \theta$. Note that for an unbiased estimator, mean squared error (MSE) equals to its variance [31]. Then:

$$\begin{aligned} MSE = \text{Var}[\hat{\theta}] &= \frac{\pi^2}{b^2}\text{Var}[D(\text{LSH}(u), \text{LSH}(v))] \\ &= \pi\frac{\theta}{b}\left(1 - \frac{\theta}{\pi}\right) = \frac{\pi^2}{b}\frac{\theta}{\pi}\left(1 - \frac{\theta}{\pi}\right) \le \frac{\pi^2}{4b}. \end{aligned}$$

For a relatively large b value, the upper bound of MSE is fairly small. For example, when $b = 256$, $MSE \le 0.00963$.

There is a line of work to derive feature hashing [32] that can preserve cosine similarity with the same error variance. But each hashed value in [32] is a sequence of float point numbers and thus using such a method only accomplishes the dimensionality reduction, does not significantly decrease the time complexity.

3.2 Fast kernel computation

Once interaction matrices are derived, kernel computation still takes significant time compared to other steps in neural ranking. Our key idea for the second optimization technique is to take advantages that the distinct number of similarity values is limited after the use of Hamming distances as an approximation. With this in mind, we transform the kernel computation for a histogram-based formula with a significantly lower computational cost. With b-bit hyperplane LSH-based footprints, since there are at most $b+1$ values of Hamming distance, we first build a histogram on Hamming distances, and then compute kernels based on the histogram values.

For interaction matrix M in KNRM with R kernels, each kernel can be computed as

$$\begin{aligned} K_k(M_i) &= \sum_{j=1}^{n} \exp\left(-\frac{(\cos\hat{\theta}_{ij} - \mu_k)^2}{2\sigma_k^2}\right) \\ &= \sum_{j=1}^{n} \exp\left(-\frac{(\cos(\frac{\pi}{b}D_{ij}) - \mu_k)^2}{2\sigma_k^2}\right) \\ &= \sum_{d=0}^{b} \underbrace{|\{j : D_{ij} = d\}|}_{\textbf{online computed}} \cdot \underbrace{\exp\left(-\frac{(\cos(\frac{\pi}{b}d) - \mu_k)^2}{2\sigma_k^2}\right)}_{\textbf{offline computed}}, \end{aligned}$$

where D_{ij} denotes the Hamming distance between the i-th query term footprint $\text{LSH}(\mathbf{q}[i])$ and the j-th document term footprint $\text{LSH}(\mathbf{d}[j])$. Set $\{j : D_{ij} = d\}$ is the histogram on Hamming distances between query and document terms. The above formula marks the part of computation conducted in the offline time.

The above histogram-based method reduces time complexity of kernel computing from $\Theta(nmR)$ for the original algorithm to

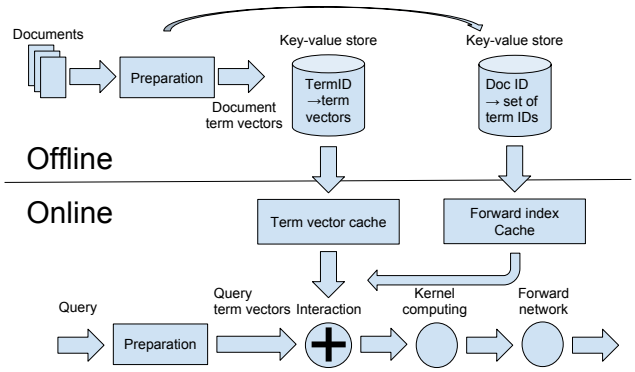


Figure 1: Architecture of interaction-based neural ranking.

$\Theta(nm+bmR)$. For web page ranking, a web page has 500 to 1K words on average, and neural ranking models usually take 20 to 50 kernels, there is a big cost difference. With $b = 256$, $n = 807$, and $R = 30$, the reduction ratio from above cost order expression $\Theta(nmR)$ to $\Theta(nm + bmR)$ is about 2.85. In our evaluation, we have observed around 417.38x speed improvement with the above histogram-based method. The reason for this large extra acceleration is partially because the original computation uses floating point arithmetic and our histogram-based method uses integer arithmetic. Another reason is that it allows us to re-arrange computation for better cache performance using a matrix notation and we will present more details on computation re-arrangement in an extended paper.

For CONV-KNRM with h^2 interaction matrices $M^{r,t}$ where $1 \le r, t \le h$, the above transformation is applied to each of R kernels for each of such matrices. Thus

$$K_k(M_i^{r,t}) = \sum_{d=0}^{b} \underbrace{|\{j : D_{ij}^{r,t} = d\}|}_{\textbf{online computed}} \cdot \underbrace{\exp(-\frac{(\cos(\frac{\pi}{b}d) - \mu_k)^2}{2\sigma_k^2})}_{\textbf{offline computed}}$$

where $D_{i,j}^{r,t}$ is the Hamming distance between the i-th LSH footprint of r-gram query term and the j-th LSH footprint of t-gram document term. The cost of kernel computation drops from the original complexity of $\Theta(h^2nmR)$ to $\Theta(h^2(nm + bmR))$.

3.3 Preparation of Term Vectors and Space Cost

For DRMM and KNRM, term vectors used for neural ranking directly adopt pretrained word embeddings (e.g. [19]), and thus there is no extra computation needed. In CONV-KNRM, a set of F convolution filters is applied to n-gram vectors based on word embeddings. The time complexity to prepare these document term vectors is $\Theta(nh^2lF)$ while deriving query term vectors costs $\Theta(mh^2lF)$.

Since $n >> m$, the time of this convolution computation for deriving document term vectors is dominating if it is conducted at the runtime when a query is handled. Our evaluation shows that without precomputing, CONV-KNRM spends too much time in preparing document term vectors. To reduce this cost, precomputation of these vectors can be employed to conduct such computation at the offline time. On the other hand, the disadvantage of precomputing is that it requires a substantial amount of space to store precomputed vectors, and LSH mapping can reduce this need significantly. The impact of incorporating LSH is described as follows.

- Without precomputing, following the original step of CONV-KNRM, we need to perform the convolution computation to derive all term vectors. Then we need extra $\Theta(mhbF)$ time to apply LSH to all unigram and n-gram term vectors for a query and documents to be ranked.
- With precomputing, the convolution computation and LSH mapping of all document term vectors are conducted in the offline time. The online query processing only needs to spend time to prepare query term vectors, which costs $\Theta(mhbF + h^2mlF)$.

Caching and space cost. Table 3 lists the dominating factors of disk space cost in bytes for the three algorithms with and without LSH or precomputing, assuming each term ID takes 32 bits and each floating number takes 32 bits. We describe two ways of storing document term representations as follows.

1) **Reference based method**.

Each term including a unigram or n-gram in the vocabulary is assigned an ID. Each term ID gives a reference to where the word embedding or term vector is stored. For each document, we need to store all the IDs of its necessary words or terms based on vector precomputing. Figure 1 illustrates the flow of neural ranking with two caches for runtime query processing. One cache is called the forward index cache which gives a mapping from a document ID to a set of term IDs this document owns. The second cache is called the term vector cache which is a mapping from a term ID to a term vector or its hash footprint. There is much higher access traffic to the second cache in ranking a document and thus the high hit ratio of the second cache is extremely important otherwise the cache miss would lead to a large number of random I/O operations to the disk.

2) **Value based method**. We store the values of embedding vectors or LSH footprints together with each document ID and they are directly accessible by a document ID. This method is useful when the memory cache demand is too high in the above reference method while it demands a larger amount of disk space.

For DRMM and KNRM with LSH, the main storage space cost for the reference method is about $4hnN$ to store the term IDs of documents and $Vb/8 + 4V$ bytes to store the map from term IDs to hash footprints where cost factor $4V$ is less significant. The second portion is the memory cache demand for fast access of hash footprints. The overall storage with LSH has a decent decrease compared to that without LSH. The size of the term-vector cache drops from about $4Vl$ without LSH to about $Vb/8$ with LSH. When $b = 256, l = 300$, the reduction ratio is about 32.8x.

For CONV-KNRM with LSH, if the reference based method is used with precomputing, the disk space needs $4hnN$ bytes for storing term IDs of N documents, and $Hb/8 + 4H$ bytes for hash footprints and index where $4H$ is less significant. Under precomputing, the size of the term-vector cache demand drops from about $4HF$ without LSH to about $Hb/8$ with LSH. When $F = 128, b = 256$, the cache space demand reduction ratio is about 6x.

To compare the storage cost of reference-based and value-based methods, using ClueWeb Category B dataset parameters with about 50 million documents, the dominating cost factor of storage space in CONV-KNRM with precomputing is $4hnN$ using the reference-based method and $hnbN/8$ using the value based method. The ratio of the value-based storage space cost over reference-based cost is approximately $\frac{b}{32}$, which is 8 when $b = 256$. With this size of the

database, the ratio of the value-based space cost over reference-based cost is also approximately $\frac{nNb/8}{4nN} = 8$ for DRMM with LSH and KNRM with LSH, respectively. With $b = 256$, the term vector disk space of the value-based method for CONV-KNRM is about 2.6TB for the ClueWeb dataset while the reference-based method costs about 351GB. Our evaluation uses the reference-based method.

4 EVALUATION

The objectives of our evaluation are to compare ranking relevance after applying LSH approximation with our design choices, demonstrate its time and space benefits, and assess the impact of using more complex embeddings on relevance and time cost.

Datasets and settings. We mainly report the performance using ClueWeb09 Category B with 50M webpages and 200 queries from the TREC Web Tracks 2009-2012 with 5-fold cross validation. We have also used two other smaller TREC datasets: TREC45 based on TREC Disks 4 and 5 with 0.5M news articles and AQUAINT with 1M news articles. Following the settings in [5, 9, 34], we first exclude spam webpages and rerank for top 1,000 documents of each query given by Galago initial ranking in the Lemur system [16]. For DRMM, the neural network has $R = 30$ kernels as input, and has a hidden layer of 5 neurons. For KNRM and CONV-KNRM, we do not update the embeddings during the training for simplicity, hence the accuracy scores may have a degradation compared with the results from the original papers. Other parameters used are: $l = 300$, $F = 128$, $h = 2$. We do not compare with the representation-based methods since the previous work shows that the interaction-based models perform better [23].

Word representation with dual embeddings. The previous work in DRMM, KNRM, CONV-KNRM typically used pretrained embeddings (e.g. word2vec) as a starting point. Nalisnick et al. firstly introduced Dual Embedding [21] for document ranking, given the fact that word2vec [19] with Negative Sampling actually trains two sets of word embedding vectors, defined as "input" (IN) and "output" (OUT) vector representations. Vector interactions between IN and IN are typically used in DRMM, KNRM, CONV-KNRM, and others for measuring term similarity. To leverage dual embedding interaction-based neural ranking models, we add a "dual" interaction matrix using IN vector representations and OUT vector representations and applying the same kernel computing techniques with the doubled time cost.

Ranking relevance and query processing time with hyperplane LSH. Table 4 shows the NDCG@20 scores [14] and average query time for different neural ranking models when using regular word embedding vectors or dual embeddings. The query processing C++ code for DRMM, KNRM, and CONV-KNRM is compiled with an optimization flag in a Linux machine which has Intel Xeon E5-2680v3 2.5 GHz dual socket and 128 GB DDR4 DRAM, and a local SSD with about 0.1 ms latency [29]. All the results are within confidence interval ±0.01 with p-value < 0.05. When increasing the number of bits for LSH, the NDCG score is getting closer to the original ones starting 64 bits, and the scores of 256 bits are fairly close to the original ones and we select this number as our choice in conducting other comparison. The competitiveness using LSH for NDCG@5 and @10 is similar. From Row 2, even time cost is roughly doubled, dual embeddings does yield a good improvement of relevance score compared to regular embeddings and LSH approximation can make it one or two orders of magnitude faster while retaining most of the relevance benefits from dual embeddings.

For average query time in Table 4, we use precomputing to derive the bigram and unigram embeddings for CONV-KNRM with and without LSH. The query response time in this table does not include the I/O cost to access document term embedding vectors. With 256 bits, DRMM with LSH is 4.12x faster than DRMM; KNRM with LSH 80.54x faster than KNRM; CONV-KNRM with LSH is 106.52x faster than CONV-KNRM. A similar speedup is accomplished when dual embeddings are used. Here we assume that term IDs of each document in processing a query are cached in memory. This forward index cache shown in Figure 1 does not need to be large and in the worst case with a cache miss, the I/O cost of accessing 1,000 such vectors is about 100ms from a local SSD. Thus we do not discuss the size demand of such a cache in this paper.

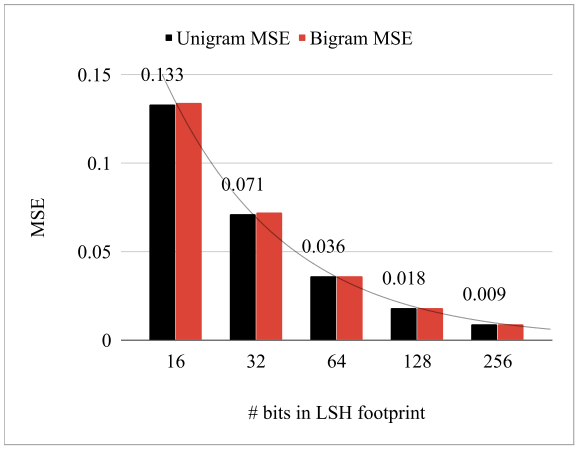


Figure 2: Errors of cosine similarity estimated with LSH.

To explain why the above three algorithms with LSH approximation can still be competitive in relevance when using a reasonable number of hash bits, Figure 2 shows that the mean squared errors of LSH-based cosine similarity approximation is relatively small for word2vec embeddings of unigrams and bigrams when increasing the number of hyperplane LSH bits. For example, when the number of LSH bits becomes 256, the actual approximation error is within 0.009, which corroborates that the relevance loss in Table 4 is reasonably small with such approximation.

Query time breakdown and impact of design optimization. With $b = 256$, Table 5 shows query time breakdown for different models in four steps discussed in Section 2 and Section 3 to re-rank 1,000 documents for a query for ClueWeb. Again document I/O cost is not included, which can be as high as 100ms per query. The underlined numbers are estimated since they require either too much time or too much space in experiments. For all methods, the time for forward neural computation step is relatively less significant compared with other steps. For DRMM, the interaction matrix computing time dominates and LSH produces a great reduction. For CONV-KNRM, the time cost for preparing term vectors corresponding word and n-gram embeddings is very expensive, and precomputing yields a significant advantage. For both KNRM and

Table 4: NDCG@20 scores and online query processing time in milliseconds.

Model	DRMM		KNRM		CONV-KNRM		Dual DRMM		Dual KNRM		Dual CONV-KNRM	
	NDCG@20	Time	NDCG@20	Time	NDCG@20	Time	NDCG@20	Time	NDCG@20	Time	NDCG@20	Time
Original	0.2580	140	0.2581	1,047	0.2688	4,900	0.2600	287	0.2665	2,073	0.2786	11,813
16-bit LSH	0.2449	7	0.2437	6	0.2563	13	0.2372	20	0.2371	7	0.2364	40
32-bit LSH	0.2464	13	0.2418	7	0.2521	18	0.2469	20	0.2557	13	0.2569	43
64-bit LSH	0.2489	20	0.2464	8	0.2571	20	0.2526	27	0.2563	26	0.2585	46
128-bit LSH	0.2528	27	0.2491	11	0.2619	27	0.2542	33	0.2633	27	0.2629	68
256-bit LSH	0.2518	34	0.2522	13	0.2640	46	0.2586	54	0.2627	33	0.2724	127
512-bit LSH	0.2551	48	0.2576	27	0.2680	86	0.2582	73	0.2624	80	0.2709	260
1024-bit LSH	0.2522	60	0.2561	53	0.2662	160	0.2573	140	0.2594	147	0.2753	548

Table 5: Query time breakdown in milliseconds for ranking 1000 documents.

Models	DRMM	KNRM	CONV-KNRM	CONV-KNRM Precomputing	DRMM LSH	KNRM LSH	CONV-KNRM LSH	CONV-KNRM Precomp. LSH
Vect. preparation	0	0	76,800.5	0.5	0	0	109,568.7	0.7
Interaction $\otimes$	121.0	50.6	1842.2	1842.2	13.7	9.7	32.0	32.0
Kernel Comp. Ker	10.1	993.2	3046.9	3046.9	11.8	1.8	7.3	7.3
Forward NN f	9.0	3.3	9.9	9.9	8.8	1.7	6.7	6.7
Total (ms)	140.1	1047.1	81,700.0	4899.5	34.3	13.2	109,614.7	46.7

Table 6: The size of disk storage and the memory cache for term vectors in GB.

Models	DRMM		KNRM		CONV-KNRM w/o precomputing		CONV-KNRM w/ precomputing	
With LSH?	No	Yes	No	Yes	No	Yes	No	Yes
TREC45	1.8 (0.8)	1.0 (0.02)	1.8 (0.8)	1.0 (0.02)	1.8 (0.8)	1.8 (0.8)	10.3 (8.3)	3.4 (1.3)
AQUAINT	2.7 (0.9)	1.8 (0.02)	2.7 (0.9)	1.8 (0.02)	2.7 (0.9)	2.7 (0.9)	12.5 (8.9)	5.1 (1.4)
ClueWeb09	178.4 (16.4)	162.5 (0.5)	178.4 (16.4)	162.5 (0.5)	178.4 (16.4)	178.4 (16.4)	492.4 (169.5)	351.0 (28.2)

CONV-KNRM, LSH reduces interaction matrix and kernel value computation significantly. The time for kernel computation is no longer dominating and our histogram-based kernel algorithm yields 417.38x reduction.

Table 6 compares the space requirements for three datasets with $b = 256$. The parameters of the datasets are listed blow. For the TREC45 data set, $N = 0.528$ million, average $n = 484$, $V = 0.67$ million, $H = 16.2$ million. For the AQUAINT data set, $N = 1.03$ million, $n = 440$, $V = 0.72$ million, $H = 17.3$ million. For the ClueWeb09 dataset, $N = 50.22$ million, $n = 807$, $V = 13.6$ million, $H = 328.43$ million [22]. The table lists two numbers in each entry. One is the total amount of disk storage space needed and another number inside a parenthesis is the size of the term-vector cache.

Our method using LSH reduces the size of term-vector cache significantly for DRMM/KNRM and CONV-KNRM with precomputing. For ClueWeb, the reference based method for CONV-KNRM without LSH needs about 492.4 GB in disk while memory size for term-vector cache is 169.5 GB. If such a memory cache is not available, the foot prints of each document need to be fetched with a random disk I/O. For ranking top 1000 documents matched for a query, the total I/O access cost can take about 100 seconds in our experiments. By integrating LSH, 351 GB disk space is still needed and the size of the term-vector cache is reduced by 6.01x to only 28.2 GB, which is fairly affordable.

5 CONCLUSION

This paper presents the design choices and algorithm optimization to improve the efficiency of three interaction-based neural ranking algorithms with LSH and a histogram-based kernel computing method. The evaluation shows that the proposed design choice yields 4.12x, 80.54x, and 106.52x time speedups for DRMM, KNRM, and CONV-KNRM respectively in the tested ClueWeb dataset with a reasonable response time and competitive relevance. Precomputing is used in our CONV-KNRM implementation for avoiding expensive document term vector preparation while it does carry extra space cost. Our LSH integration moderately decreases the overall storage space while making a major reduction for the term vector cache space demand, and the reduction ratio is 32.8x for DRMM and KNRM, and 6.01x for CONV-KNRM. This is critical to enable fast in-memory access of (hashed) embedding contents for a large dataset. More complex dual embeddings can improve relevance while its fast LSH approximation still preserves the most of its benefits.

ACKNOWLEDGMENTS

This work is supported in part by NSF IIS-1528041. It has used the NSF-supported resource in the Extreme Science and Engineering Discovery Environment (XSEDE) under allocation IRI190005 and

in the Center for Scientific Computing at UCSB, and we thank Michael Agun, Sharon Solis, and Mahidhar Tatineni for their help. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

REFERENCES

[1] Alexandr Andoni and Piotr Indyk. 2006. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*. IEEE, 459–468.

[2] Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya Razenshteyn, and Ludwig Schmidt. 2015. Practical and optimal LSH for angular distance. In *Advances in Neural Information Processing Systems*. 1225–1233.

[3] Alexandr Andoni and Ilya Razenshteyn. 2015. Optimal data-dependent hashing for approximate near neighbors. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. ACM, 793–801.

[4] Moses S Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. ACM, 380–388.

[5] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 126–134.

[6] Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. 2007. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 271–280.

[7] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*. ACM, 253–262.

[8] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. 1999. Similarity Search in High Dimensions via Hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases (VLDB '99)*. 518–529.

[9] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of CIKM'16*. ACM, 55–64.

[10] Monika Henzinger. 2006. Finding near-duplicate web pages: a large-scale evaluation of algorithms. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 284–291.

[11] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2333–2338.

[12] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2016. Binarized neural networks. In *Advances in neural information processing systems*. 4107–4115.

[13] Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*. ACM, 604–613.

[14] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* 20, 4 (Oct. 2002), 422–446.

[15] Brian Kulis and Trevor Darrell. 2009. Learning to hash with binary reconstructive embeddings. In *Advances in neural information processing systems*. 1042–1050.

[16] Lemur. [n. d.]. http://www.lemurproject.org/.

[17] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. 2014. *Mining of massive datasets*. Cambridge university press.

[18] Gurmeet Singh Manku, Arvind Jain, and Anish Das Sarma. 2007. Detecting near-duplicates for web crawling. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 141–150.

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[20] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to match using local and distributed representations of text for web search. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1291–1299.

[21] Bhaskar Mitra, Eric Nalisnick, Nick Craswell, and Rich Caruana. 2016. A dual embedding space model for document ranking. *arXiv preprint arXiv:1602.01137* (2016).

[22] Gregory B Newby, Chris Fallen, and Kylie McCormick. 2009. *Lucene for n-grams using the ClueWeb Collection*. Technical Report. ALASKA UNIV ANCHORAGE ARTIC REGION SUPERCOMPUTING CENTER.

[23] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. 2017. A deep investigation of deep IR models. In *SIGIR 2017 Workshop on Neural Information Retrieval (Neu-IR'17)*.

[24] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Jingfang Xu, and Xueqi Cheng. 2017. Deeprank: A new deep architecture for relevance ranking in information retrieval. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 257–266.

[25] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[26] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*. Springer, 525–542.

[27] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 373–374.

[28] Wei Tang, Gang Hua, and Liang Wang. 2017. How to train a compact binary neural network with high accuracy?. In *AAAI*. 2625–2631.

[29] J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, R. Roskies, J. R. Scott, and N. Wilkins-Diehr. 2014. XSEDE: Accelerating Scientific Discovery. *Computing in Science & Engineering* 16, 5 (Sept.-Oct. 2014), 62–74. https://doi.org/10.1109/MCSE.2014.80

[30] Ferhan Ture, Tamer Elsayed, and Jimmy Lin. 2011. No free lunch: brute force vs. locality-sensitive hashing for cross-lingual pairwise similarity. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 943–952.

[31] Dennis Wackerly, William Mendenhall, and Richard L Scheaffer. 2014. *Mathematical statistics with applications*. Cengage Learning.

[32] Kilian Weinberger, Anirban Dasgupta, Josh Attenberg, John Langford, and Alex Smola. 2009. Feature hashing for large scale multitask learning. *arXiv preprint arXiv:0902.2206* (2009).

[33] Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. 2017. Word-entity duet representations for document ranking. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 763–772.

[34] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 55–64.

[35] Chenyan Xiong, Zhengzhong Liu, Jamie Callan, and Tie-Yan Liu. 2018. Towards Better Text Understanding and Retrieval Through Kernel Entity Salience Modeling. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. ACM, 575–584.

[36] Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 1271–1279.

[37] Han Zhu, Mingsheng Long, Jianmin Wang, and Yue Cao. 2016. Deep Hashing Network for Efficient Similarity Retrieval.. In *AAAI*. 2415–2421.

Triple Trustworthiness Measurement for Knowledge Graph

Shengbin Jia*
Tongji University
Shanghai, China
shengbinjia@tongji.edu.cn

Yang Xiang
Tongji University
Shanghai, China
shxiangyang@tongji.edu.cn

Xiaojun Chen
Tongji University
Shanghai, China
xiaojunchen@tongji.edu.cn

ABSTRACT

The Knowledge graph (KG) uses the triples to describe the facts in the real world. It has been widely used in intelligent analysis and applications. However, possible noises and conflicts are inevitably introduced in the process of constructing. And the KG based tasks or applications assume that the knowledge in the KG is completely correct and inevitably bring about potential deviations. In this paper, we establish a knowledge graph triple trustworthiness measurement model that quantify their semantic correctness and the true degree of the facts expressed. The model is a crisscrossing neural network structure. It synthesizes the internal semantic information in the triples and the global inference information of the KG to achieve the trustworthiness measurement and fusion in the three levels of entity level, relationship level, and KG global level. We analyzed the validity of the model output confidence values, and conducted experiments in the real-world dataset FB15K (from Freebase) for the knowledge graph error detection task. The experimental results showed that compared with other models, our model achieved significant and consistent improvements.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; *Natural language processing*; *Knowledge representation and reasoning*; Reasoning about belief and knowledge; Probabilistic reasoning.

KEYWORDS

Knowledge graph; trustworthiness; neural network; error detection

ACM Reference Format:
Shengbin Jia, Yang Xiang, and Xiaojun Chen. 2019. Triple Trustworthiness Measurement for Knowledge Graph. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313586

1 INTRODUCTION

A knowledge graph (KG), aims to describe the various entities or concepts and their relationships existing in the objective world [43]. It usually stores knowledge in the form of triples (head entity, relationship, tail entity), which can be simplified to (h, r, t). It lays the foundation for the knowledge-based organization and intelligent application in the Internet age with its powerful semantic processing capabilities and open organization capabilities.

In recent years, an increasing number of researchers are committed to productively extracting information directly from unstructured web text, such as ORE [2, 9, 19], NELL [7], and to automatically constructing large-scale knowledge graphs, such as Freebase [3], DBpedia [1], and Wikidata[1]. However, some noises and errors are inevitably introduced in the process of automation. References [23] and [14] verify the existence and problems of errors in the KG. Existing knowledge-driven learning tasks or applications [11, 13, 22, 27], assume knowledge in the existing KG is completely correct and therefore bring about potential errors [28, 42].

For a piece of knowledge in a KG, especially from a professional field, it is difficult to clearly determine whether it is true when it is not tested in practice or is not strictly and mathematically proven. Therefore, we introduce the concept of KG triple trustworthiness, which indicates the degree of certainty that the knowledge expressed by the triple is true. It's value is set to be within the interval [0, 1]. The closer the value is to 0, the greater the probability that the triple is in error. Based on this, we can find possible errors in the existing KG and improve knowledge quality in the KG.

There are intricate relationships among the entities in the KG, the same relationship can occur between different entities, and multiple relationships can associate with the same entity at the same time. It is a challenge to study how to use appropriate methods to evaluate the trustworthiness for a knowledge triple. We propose a knowledge graph triple trustworthiness measurement model (KGTtm), which is a crisscrossed neural network-based structure. We measure the trustworthy probability from multiple levels, including the entity level (correlation strength between an entity pair), the relationship level (translation invariance of relation vectors), and the KG global level (inference proof of triple related reachable paths). Corresponding to different levels, we generate three essential questions and focus on solving them by designing three kinds of Estimators. Next, a comprehensive triple confidence value is output through a Fusioner.

The main contributions of this work include: (1) We propose a knowledge graph triple trustworthiness measurement method that makes comprehensive use of the triple semantic information and globally inferring information. We can achieve three levels of measurement and an integration of confidence value at the entity level, relationship level, and the knowledge graph global level. (2) We empirically verify the effectiveness of the triple trustworthiness on benchmark data created from a real-world, large-scale KG Freebase. Experimental results show that the error or noise instances are assigned low confidence values, meanwhile, high trustworthiness

*This author is the one who did all the really hard work.

[1]https://www.wikidata.org/wiki/Wikidata

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313586

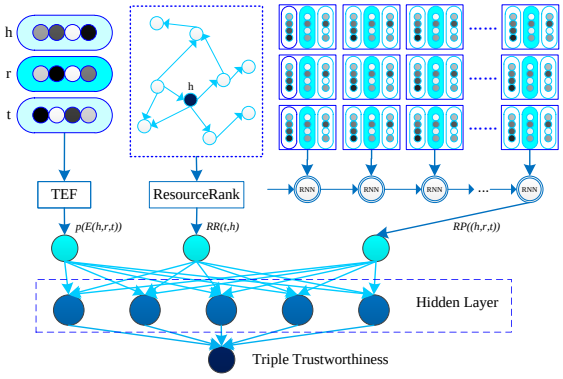


Figure 1: The triple trustworthiness measurement model for KG.

for true triples. (3) The trustworthiness calculated by the KGTtm could be utilized in knowledge graph construction or improvement. We evaluate the model on the knowledge graph error detection task. Experimental results show that the KGTtm can effectively detect the error triples and achieve promising performances.

2 RELATED WORK

The concept of Trustworthiness has been applied to knowledge graph related tasks to some extent. Reference [42] proposed a triple confidence awareness knowledge representation learning framework, which improved the knowledge representation effect. There were three kinds of triple credibility calculation methods using the internal structure information of the KG. This method used only the information provided by the relationship, ignoring the related entities. The NELL [7] constantly iterated the extracting template and kept learning new knowledge. It used heuristics to assign confidence values to candidate relations and continuously updated the values through the process of learning. This method was relatively simple but lacked semantic considerations. Dong et al. [8] constructed a probabilistic knowledge base (Knowledge Vault), where the reliable probability of a triple was a fusion of some estimators. This method was tailored for their knowledge base construction and did not have good generalization capabilities. Li et al. [22] used the neural network method to embed the words in ConceptNet and provide confidence scores to unseen tuples to complete the knowledge base. This method considered only the triples themselves, ignoring the global information provided by the knowledge base. The above models used the trustworthiness to solve various specific tasks. It shows that the triple trustworthiness is important for applications and research. However, there is a lack of systematic research on the knowledge triple trustworthiness calculation method at present. Our work is devoted to this basic research and proposes a unified measurement model that could facilitate a variety of tasks.

In this work, we verify the effect of the triple trustworthiness on the knowledge graph error detection task. The Knowledge graph error detection task is dedicated to identifying whether a triple in the KG is in error. The error detection can actually be regarded as a special case of the trustworthiness measurement, which is divided into two kinds of Boolean value types: “true (trusted)" and “error (untrusted)". Traditional methods [7, 14, 16, 21] were still based on manual detection, and the cost was considerable. Recently, some people have begun to study automatic KG error detection methods [8, 23, 31, 35]. In particular, embedding-based methods [5, 22, 38] have gained a significant amount of attention. we can efficiently measure the semantic correlations of entities and relations in the vector space. Whether two entities have a potential relationship could be predicted by matrix-vector operations of their corresponding embeddings. They have good efficiency and prospect. Furthermore, the Knowledge representation learning (KRL) technology is used to project the entities and relations in the KG into a dense, real-valued and low-dimensional semantic embeddings. The main methods include TransE [5], TransH [41], TransR [25], TransD [18], PTransE [24], ComplEx [39] and others.

3 THE TRIPLE TRUSTWORTHINESS MEASUREMENT MODEL

The triple trustworthiness measurement (KGTtm) Model for knowledge graph is presented based on a crisscrossing neural network structure, as shown in figure 1. Longitudinally, it can be divided into two levels. The upper is a pool of multiple trustworthiness estimate cells (Estimator). The output of these Estimators forms the input of the lower-level fusion device (Fusioner). The Fusioner is a Multi-layer perceptron to generate the final trustworthiness value for each triple. Viewed laterally, for a given triplet (h, r, t), we consider the triple trustworthiness from three progressive levels and correspondingly answer three hierarchical questions. 1) Is there a possible relationship between entity pairs (h, t)? 2) Can a certain relationship r occur between entity pairs (h, t)? 3) From a global perspective, can other relevant triples in the KG infer that the triple is trustworthy? For answering these questions we designed three kinds of Estimators, as described below.

3.1 Is there a possible relationship between the entity pairs?

We use the association strength between a given entity pair (h, t) to measure the likelihood of an undetermined relationship occurring between the pair. If a pair of entities has heavily weak relevance, it seems to be hopeless that there is a relationship between the entity pairs. The trustworthiness of the triples formed by the entity pair will be greatly compromised. As shown in figure 2, there are dense edges (relationships) from node (entity) A to node E, that is, there is a high association strength about (A, E). We can easily guess that there is a relationship between entity pair (A, E). However, it is impossible to reach F from G following the directed edges. We can also guess that there is no a relationship between (G, F).

We propose an algorithm named **ResourceRank**, to characterize the association strength between an entity pair according to the idea of Resource allocation [24, 26, 42, 45]. The algorithm assumes that the association between entity pairs (h, t) will be stronger, and more resource is passed from the head h through all associated paths to the tail t in a graph. The amount of resource aggregated into t ingeniously indicates the association strength from h to t.

The ResourceRank algorithm mainly includes three steps: 1) Constructing a directed graph centered on the head entity h. 2) Iterating

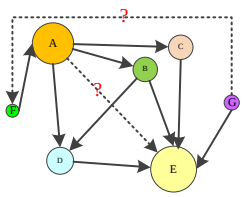


Figure 2: The graph of resource allocation in the ResourceRank algorithm.

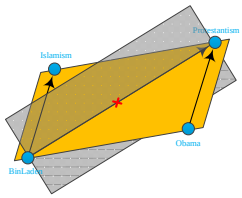


Figure 3: Effects display of the Translation based energy function.

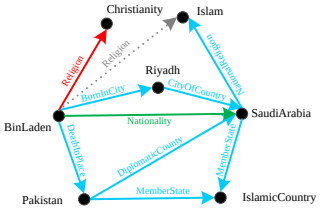


Figure 4: The inference instances for triple trustworthiness.

the flow of resources in the graph until it converges and calculates the resource retention value of the tail entity t. 3) Synthesizing other features and output the likelihood of $(h, ?, t)$.

Specific details are described below: Each entity is abstracted into a node. If there is a relationship from the entities e_1 to e_2, a directed edge will exist from node e_1 to e_2. Therefore, the KG can be mapped as a directed graph. This graph is weakly connected, but starting from h each node in the graph can be reached. In the initial state, the resource amount of h is 1, others is 0, and the sum of all nodes is always 1. If a node does not exist in the graph, then its resource is always 0. Moreover, there may be multiple relations between an entity pair (e_1, e_2) but only one directed edge from e_1 to e_2 in the graph. Depending on the number of these relations, each edge will have a different bandwidth. The larger the bandwidth is, the more resource flows through the edge.

The resource owned by node h will flow through all associated paths to each other nodes in the entire graph. We simulate the flow of resource flowing until distribution steady based on the PageRank [6, 33] algorithm. The value of the resource on the tail entity is $R(t \mid h)$, it is calculated as follows:

$$R(t \mid h) = (1-\theta) \sum_{e_i \in M_t} \frac{R(e_i \mid h) \cdot BW_{e_i t}}{OD(e_i)} + \frac{\theta}{N}. \tag{1}$$

Where, M_t is the set of all nodes that have outgoing links to the node t, $OD(e_i)$ is the out-degree of the node e_i and the $BW_{e_i t}$ is the bandwidth from the e_i to t. Thus, for each node e_i in M_t, the resource transferred from e_i to t should be $\frac{R(e_i|h) \cdot BW_{e_i t}}{OD(e_i)}$. Because the KG noises, the graph is imperfect. And there may be closed loops affecting the resources flowing. In order to improve the model fault-tolerance, we assume that the resource flow from each node may directly jump to a random node with the same probability θ. This part of resource that flows to t randomly is $\frac{1}{N}$, and N is the total number of nodes.

Different states of nodes in the graph reflect information of entity. Considering the following six characteristics: 1) $R(t \mid h)$; 2) In-degree of head node $ID(h)$; 3) Out-degree of head node $OD(h)$; 4) In-degree of tail node $ID(t)$; 5) Out-degree of tail node $OD(t)$; 6)The depth from head node to tail node Dep, we can construct a feature vector V. After being activating, the vector is transformed into a probability value as $RR(h, t)$, indicating the likelihood that there may be one or more relationships between the head entity h and the tail entity t. This transformation is:

$$\begin{cases} u = \alpha(W_1 V + b_1) \\ RR(h,t) = W_2 u + b_2 \end{cases} \tag{2}$$

Here, α is a nonlinear activation function, W_i and b_i are parameter matrices that can be trained during model training. The $RR(h, t)$ is within the range [0, 1]. The closer it is to 1, the more likely it is that there is a relationship between h and t, which answer the questions shown in the title in the entity layer.

3.2 Can the determined relationship r occur between the entity pair (h, t) ?

The above Estimator can only measure the likelihood that one undetermined relationship occurring between the entity pair, but not what kind of relationship. We next calculate the possibility of such a relation r occurring between the entity pair (h, t) by the **Translation-based energy function (TEF)** algorithm.

Inspired by the translation invariance phenomenon in the word embedding space [29, 30], the relationship in the KG is regarded as a certain translation between entities; that is, the relational vector $\mathbf{r}$ is as the translating operations between the head entity embedding $\mathbf{h}$ and the tail entity embedding $\mathbf{t}$ [5]. As illustrated in figure 3, in the vector space, the same relational vector can be mapped to the same plane and freely translated in the plane to remain unchanged. The triples (BinLaden, Religion, Islam) and (Obama, Religion, Protestantism) should be all correct. However, according to translational invariance of relation vectors, (BinLaden, Religion, Protestantism) must be wrong. Therefore, An trustworthy triple (h, r, t), should satisfy $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. The energy function is defined as $E(h, r, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|$. The higher the degree of fit between h, r, and t, the smaller the value of $E(h, r, t)$ will be. We believe that the smaller the $E(h, r, t)$ value is, the probability that the relationship r is established between the entity pair (h, t) will be greater, and the trustworthiness of (h, r, t) will be better, and vice versa.

The TEF algorithm operates as follows: Firstly, knowledge representation learning technology is used to implement a low-dimensional distributed representation for entities or relations, and we compute $E(h, r, t)$ for each triple. Then, a modified sigmoid function is used to convert $E(h, r, t)$ into the probability that the entity pair (h, t) constitutes the relationship r. The conversion formula is as follows:

$$P(E(h,r,t)) = \frac{1}{1 + e^{-\lambda(\delta_r - E(h,r,t))}} \tag{3}$$

Here, δ_r is a threshold related to the relationship r. When $E(h, r, t) = \delta_r$, the probability value P is 0.5. If $E(h, r, t) < \delta_r$, then $P > 0.5$. The λ is a hyperparameter used for smoothing and can be adjusted dynamically along with the model training. The $P(E(h, r, t))$ answers the second question in the relation layer.

3.3 Can the relevant triples in the KG infer that the triple is trustworthy?

Inspired by "social identity" theory [17, 40], we make an metaphor: regarding the KG as a social group, where each triple is an individual. The degree of acknowledgements from other individuals to the targeted individual (target triple) reflects whether the targeted individual can properly integrate into the society (i.e., the KG). We believe that only a true triple can achieve popular recognition. Vice versa, if a triple is well accepted, we tend to believe that it is trustworthiness. Therefore, the answer is yes to the question in the title. How to infer the credibility of the target triple by evaluating the acknowledgements of the relevant triples in the KG?

We design a **Reachable paths inference (RPI)** algorithm to meet it. There are many substantial multi-step paths from head entities to tail entities, which indicate the semantic relevance and the complex inference patterns among triples [34]. These reachable paths will be important evidences for judging the triple trustworthiness. For example, as shown in figure 4, there are multiple reachable paths between entity pairs "Bin Laden" and "Saudi Arabia". According to the path "Bin Laden $\overset{BornInCity}{\rightarrow}$ Riyadh $\overset{CityOfCountry}{\rightarrow}$ Saudi Arabia", we can firmly infer the fact triple (Bin Laden, Nationality, Saudi Arabia). In addition, we suppose there is a pseudo-triple (Bin Laden, Religion, Christianity) in the KG. The related paths will be very few and illogical, and we should doubt the credibility of this tuple. In contrast, we can find the correct triple (Bin Laden, Religion, Islam) because it gets good acknowledgements depending on multiple reachable paths. To exploit the reachable paths for inferring triple trustworthiness, we need to address two key challenges:

3.3.1 Reachable Paths Selection. In a large-scale KG, the number of reachable paths associated with a triple may be enormous. It is consuming to weigh all the paths. meanwhile, not all paths are meaningful and reliable. For example, the path "Bin Laden $\overset{DeathInPlace}{\rightarrow}$ Pakistan $\overset{DiplomaticCountry}{\rightarrow}$ Saudi Arabia" provided only scarce evidence to reason about the credibility of the triple (Bin Laden, Nationality, Saudi Arabia). Therefore, it is necessary to choose the most efficient reachable paths to use. Previous works believed that the paths that led to lots of possible tail entities were mostly unreliable for the entity pair. They proposed a path-constraint resource allocation algorithm to select relation paths [24, 42]. Such a method ignored the semantic information of the paths. However, we find that the reliability of the reachable path is actually a consideration of the semantic relevance of the path with the target triple. Therefore, we propose a **Semantic distance-based path selection** algorithm, which is described as Algorithm 1.

3.3.2 Reachable Paths Representation. After the paths are selected, it is necessary to map each path to a low-dimensional vector for subsequent calculations. The previous methods [24, 42] merely considered the relations in the paths. Here, we consider whole triple in the paths, including not only relations but also the head, tail entities, since the entities can also provide significant semantic information. The embeddings of the three elements of each triple are concatenated as a unit s. Therefore, a path is transformed into an ordered sequence $S = \{s_1, s_2, ..., s_n\}$. We use the recurrent neural networks (RNNs) [15], which are good at capturing temporal semantics of a sequence, to learn the semantic information contained in the path. The RNN layer encodes s_t by considering forward information from s_1 to s_t. We use the output vector $\mathbf{h_t}$ of the last time to represent the semantic information of each path. We stitch the output $\mathbf{h_t}$ of the $TopK$ paths together to form a vector. The vector is nonlinear transformed (using the method in eq(2).) into a recognition value as $RP((h, r, t))$, indicating the recognition degree that the relevant triples in the KG infer the target triple being credible.

Algorithm 1 Reachable Paths Selecting Algorithm

Require:
The knowledge graph (KG); A given target triple (h, r, t).
Ensure:
Multiple reachable paths most relevant to target triple.

1: Search the reachable paths from h to t in the KG and store in $P_{(h,r,t)} = \{p_1, ..., p_n\}$;
2: For each $p_i = \{(h, l_1, e_1), (e_1, l_2, e_2), ..., (e_{n-1}, l_n, t)\}$, calculate
1) the semantic distance between the r and all relations in p_i,
as, $SD(p_i(l), r) = \frac{1}{n} \sum_{l_j \in p_i(l)} \frac{r \cdot l_j}{\|r\| \|l_j\|}$;
2) the semantic distance between the t and all head entities in p_i, as, $SD(p_i(e), t) = \frac{1}{n} \sum_{e_j \in p_i(e)} \frac{t \cdot e_j}{\|t\| \|e_j\|}$;
3) the semantic distance between the h and all tail entities in p_i, as, $SD(p_i(e), h) = \frac{1}{n} \sum_{e_j \in p_i(e)} \frac{h \cdot e_j}{\|h\| \|e_j\|}$;
3: Calculate the average distance
$\bar{SD}(p_i) = \frac{1}{3}(SD(p_i(e), t) + SD(p_i(l), r) + SD(p_i(e), h))$;
4: Select first $TopK$ paths with the highest $\bar{SD}(p_i)$ scores.
5: Return $\{p_i \mid 1 \leqslant i \leqslant TopK, Sort(\bar{SD}(p_i), descend)\}$

3.4 Fusing the Estimators

We designed a Fusioner based on a multi-layer perceptron [12] to output the final triples trustworthiness values. A simple way to combine the above Estimators is to splice their outputs into a feature vector $f(s)$ for each triple $s = (h, r, t)$ and,

$$f(s) = \left[RR(h, t), p(E(s)), RP(s)\right] \tag{4}$$

The vector $f(s)$ will be inputted into the Fusioner and transformed passing multiple hidden layers. The output layer is a binary classifier by assigning a label of $y = 1$ to true tuples and a label of $y = 0$ to fake ones. A nonlinear activation function (logistic sigmoid) is used to calculate $p(y = 1|f(s))$ as,

$$\begin{cases} h_i = \sigma(W_{h_i} f(s) + b_{h_i}) \\ p(y = 1 \mid f(s)) = \varphi(W_o h + b_o) \end{cases} \tag{5}$$

Where h_i is the i_{th} hidden layer, W_{h_i} and b_{h_i} are the parameter matrices to be learned in the i_{th} hidden layer, and W_o and b_o are the parameter matrices of the output layer.

4 EXPERIMENTS

4.1 Experimental Settings

We focus on Freebase [4], which is one of the most popular real-world large-scale knowledge graphs, and we perform our experiments on the FB15K [5], which is a typical benchmark knowledge

graph extracted from Freebase. As for the FB15K, there are 1,345 relations and 14,951 entities and the corresponding 592,213 triples. We use all of the 592,213 triples to construct graphs which are described in Section 3.1. Each head entity is the core of a graph, so we can construct 14,951 graphs. These graphs are used for ResourceRank algorithm and Reachable paths inference algorithm.

There are no explicit labelled errors in the FB15K. Considering the experience that most errors in real-world KGs derive from the misunderstanding between similar entities, we use the methods described in [42] to generate fake triples as negative examples automatically where the picked entity should once appear at the same position. For example, (Newton, Nationality, American) is a potential negative example rather than the obvious irrational (Newton, Nationality, Google), given a true triple (Newton, Nationality, England), as England and American are more common as the tails of Nationality. We assure that the number of negative examples is equal to that of positive examples. In a random but quantitatively balanced manner, one of the three kinds of fake triples may be constructed for each true triple: one by replacing head entity, one by replacing relationship, and one by replacing tail entity. We assign a label of 1 to positive examples and 0 to negative examples. Therefore, we build a corpus for our experiments which contains double 592,213 triples. It is separated for training (double 483,142 triples), valid (double 50,000 triples), and testing (double 59,071 triples)

We implement the neural network using the Keras library[2][3]. The dimension of the entity and relation embeddings is 100. The batch size is fixed to 50. We use early stopping [10] based on the performance on the validation set. The number of RNN units is 100. Parameter optimization is performed with the Adam optimizer [20], and the initial learning rate is 0.001. In addition, to mitigate overfitting, we apply the dropout method [37] to regularize our model.

In addition, there are some adjustable parameters during the model training. θ is the probability that the resource flow from a node directly jumps to a random node. According to the value in the PageRank algorithm [6, 33], we set $\theta = 0.15$. We set $K = 4$ and $TopK = 3$. If the two parameters are set too large, the consumption of model training will be greatly increased. If they are set too small, the related algorithm will be affected. Thus this is the trade-off after repeated attempts. The relation-specific threshold δ_r can be searched via maximizing the classification accuracy on the validation triples, which belong to the relation r.

4.2 Interpreting the Validity of the Trustworthiness

To verify whether the triple trustworthiness outputs from KGTtm is valid, we do the following analysis on the test set.

We display the triple confidence values in a centralized coordinate system, as shown in figure 5(a). The left area shows the distribution of the values of the negative examples, while the right area shows that of the positive examples. It can be seen that the confidence values of the positive examples are mainly concentrated in the upper region (> 0.5)[4]. In contrast, the values of the negative examples are mainly concentrated in the lower region (< 0.5). It is consistent with the natural law of judging triple trustworthiness, proving that the triple confidence values output from our model are meaningful.

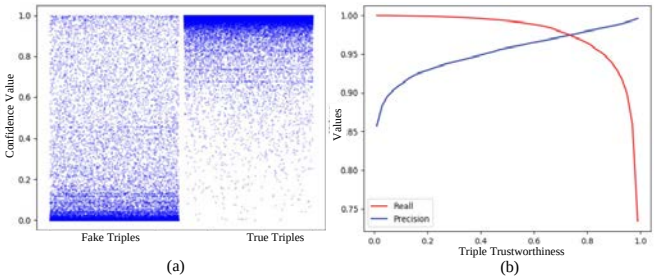


Figure 5: (a) The scatter plot of the triple confidence values distribution. (b) The various value cures of precision and recall with the triple confidence values.

In addition, by dynamically setting the threshold for the triple confidence values (only if the value of a triple is higher than the threshold can it be considered trustworthy.), we can measure the curves of the precision and recall of the output, as shown in figure 5(b). As the threshold increases, the precision continues to increase, and the recall continues to decrease. When the threshold is adjusted within the interval [0, 0.5], there is no obvious change in the recall, and it remains at a high level. However, if the threshold is adjusted within the interval [0.5, 1] , the recall tends to decline. In particular, the closer the threshold is to 1, the greater the decline rate will be. These show that the positive examples universally have higher confidence values (> 0.5). Moreover, the precision has remained at a relatively high level, even when the threshold is set to a small value, which indicates that our model can identify the negative instances well and assign them a small confidence value.

4.3 Comparing With Other Models on The Knowledge Graph Error Detection Task

The Knowledge graph error detection task is to detect possible errors in the KG according to their triple trustworthy scores. Exactly, it aims to predict whether a triple is correct or not, which could be viewed as a triple classification task [36].

We give several evaluation results. (1) The accuracy of classification. The decision strategy for classification is that if the confidence value of a testing triple (h, r, t) is below the threshold 0.5, it is predicted as negative, otherwise, it is positive. (2) The maximum F1-score when the given threshold is at [0, 1].

As shown in table 1, our model has better results in terms of accuracy and the F1-score than the other models. The Bilinear model [22] [32] [44] and Multi layer perceptron (MLP) model [8] [22] have been widely applied to the KG related tasks. They can calculate a score for the validity of triples through operations, such as tensor decomposition and nonlinear transformation. Here we convert the scores to the confidence values using the sigmoid function. Compared with the Bilinear and MLP models, our model shows improvements of more than 10% in the two evaluation indicators. We use the TEF algorithm (as illustrated in Section 3.2) to transform the output

[2]https://github.com/keras-team/keras
[3]The code can be obtained from https://github.com/TJUNLP/TTMF.
[4]The output-layer of our model is a binary classifier, and the confidence value of a triple is the probability that the triple is predicted as label=1, therefore we choose a threshold of 0.5.

Table 1: Evaluation results on the Knowledge graph error detection.

Models	MLP	Bilinear	TransE	TransH	TransD	TransR	PTransE	Ours_TransE	Ours_PTransE	Ours_TransH
Accuracy	0.833	0.861	0.868	0.912	0.913	0.902	0.941	0.977	0.978	**0.981**
F1-score	0.846	0.869	0.876	0.913	0.913	0.904	0.942	0.975	0.979	**0.982**

Table 2: Evaluation results of the three type noises.

Models	$(h, r, ?)$		$(h, ?, t)$		$(?, r, t)$	
	Recall	Quality	Recall	Quality	Recall	Quality
MLP	0.970	0.791	0.912	0.735	0.978	0.844
Bilinear	0.936	0.828	0.904	0.807	0.973	0.907
TransE	0.960	0.796	0.927	0.759	0.959	0.786
TransH	0.935	0.826	0.927	0.811	0.955	0.850
TransD	0.942	0.838	0.909	0.804	0.954	0.853
TransR	0.964	0.872	0.921	0.829	0.972	0.868
PTransE	0.944	0.841	0.973	0.888	0.957	0.863
Ours	**0.987**	**0.943**	**0.977**	**0.923**	**0.994**	**0.959**

Table 3: Evaluation results of each single estimator on the Knowledge graph error detection.

Models	TEF(TransE)	ResourceRank	RPI	KGTtm
Accuracy	0.868	0.811	0.881	0.977

of the embedding-based models of TransE, TransH, TransD, TransR, and PTransE into triple confidence values. These embedding-based models are better than the traditional method, but their results are affected by the quality of the embeddings. In comparison, our model does not rely on word embeddings. We introduce different embeddings into our model, as shown by Ours_TransE, Ours_TransH, and Ours_PTransE, which have very subtle effects. Since our model makes full use of the internal semantic information of the triple and the global inference information of the knowledge graph, it is more robust to achieve the three-level measure of trustworthiness.

4.4 Analyzing the ability of models to tackle the three type noises.

Three types of errors or noises are generated by replacing the head entity, tail entity, or relation in the triples. We measure the ability of the models to recall positive cases from candidate triples doped with a large number of noises.

We select only true triples (positive examples) in test set and divide them into three categories: all pairs of head and relation $(h, r, ?)$, all pairs of head and tail $(h, ?, t)$, and all pairs of tail and relation $(?, r, t)$. Then complement all empty positions with the objects in entity set or relationship set. In this way, for a certain pair of head and relation or a pair of tail and relation, 14,951 candidate triples can be constructed respectively. Similarly, for a pair of head and tail entities, 1,345 candidate triples can be generated. As for a complemented triple (h, r, t), we calculate its confidence value. When the value is higher than the threshold (> 0.5), we judge it to be correct. Two evaluation metrics are conducted as: (1) The recall of true triples in the test set (Recall). (2) The average trustworthiness values across each set of true triples (Quality) [22].

By analyzing the table 2, we find that our model achieves a higher recall on the three test sets compared to other models, it shows that our model can more accurately find the right from noisy triples. The average trustworthiness values of our model is higher than that of others, which show that our model can better identify the correct instances and with high confidence values. In addition, it can be found that the output of almost all models satisfies this phenomenon: The $(?, r, t)$ set achieves the best results, but the worst among the $(h, ?, t)$ set. It is difficult to judge the relation types of an entity pair, because there may be various relations between a certain entity pair, which increases the difficulty of model judgment.

4.5 Analyzing the Effects of Single Estimators

To measure the effect of single Estimators, we separate each Estimator as an independent model to calculate the confidence values for triples. The results in the knowledge graph error detection are shown in table 3. It can be found that the accuracy obtained by each model is above 0.8, which proves the effectiveness of each Estimator. Among them, the Reachable paths inference (RPI) based method achieves better results than the other two Estimators. After combining all the Estimators, the accuracy obtained by the global model (KGTtm) has been greatly improved, which shows that our model has good flexibility and scalability. It can well integrate multiple aspects of information to obtain a more reasonable trustworthiness.

It is worth emphasizing that our model is flexible and easy to extend. The newly added estimators can train their parameters together with the model frame. In addition, the confidence value generated by a single estimator can be extended to the feature vector $f(s)$ straightly.

5 CONCLUSION

In this paper, we establish a knowledge graph triple trustworthiness measurement model (KGTtm) to detect and eliminate errors in the KG. The KGTtm evaluates the trustworthiness of the triples from three perspectives and synthetically uses the triple semantic information and the global inference information of the knowledge graph. Experiments were conducted on the popular Freebase. The results confirmed the capabilities of our model. In the future, we will explore adding more estimators to the model to further improve the effectiveness of the trustworthiness. We will also try to apply the trustworthiness to more knowledge-based applications.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant 71571136, and in part by the Project of Science and Technology Commission of Shanghai Municipality under Grant 16JC1403000 and Grant 14511108002.

REFERENCES

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: A nucleus for a Web of open data. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 4825 LNCS. 722–735. https://doi.org/10.1007/978-3-540-76298-0_52

[2] Michele Banko, Mj Cafarella, and Stephen Soderland. 2007. Open information extraction for the web. In *the 16th International Joint Conference on Artificial Intelligence (IJCAI 2007)*. 2670–2676. https://doi.org/10.1145/1409360.1409378

[3] Kurt Bollacker, Robert Cook, and Patrick Tufts. 2007. Freebase: A shared database of structured general human knowledge. *Proceedings of the national conference on Artificial Intelligence* (2007).

[4] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data (SIGMOD 2008)*. 1247–1250. https://doi.org/10.1145/1376616.1376746

[5] Antoine Bordes, Nicolas Usunier, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-Relational Data. *Advances in NIPS* (2013). https://doi.org/10.1007/s13398-014-0173-7.2 arXiv:arXiv:1011.1669v3

[6] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. 2000. Graph structure in the Web. *Computer Networks* 33, 1 (2000), 309–320.

[7] Andrew Carlson, Justin Betteridge, and Bryan Kisiel. 2010. Toward an Architecture for Never-Ending Language Learning. *In Proceedings of the Conference on Artificial Intelligence (AAAI) (2010)* (2010). https://doi.org/10.1002/ajp.20927

[8] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '14* (2014). https://doi.org/10.1145/2623330.2623623 arXiv:arXiv:1301.3781v3

[9] Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*. 1535–1545. https://doi.org/10.1234/12345678

[10] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech Recognition with Deep Recurrent Neural Networks. In *arXiv preprint arXiv:1303.5778*. IEEE, 6645–6649. arXiv:1303.5778 http://arxiv.org/abs/1303.5778

[11] Sanping Guan, Xioalong Jin, Yantao Jia, Yuanzhuo Wang, and Cheng Xueqi. 2018. Knowledge graph oriented knowledge inference methods: A survey. *Ruan Jian Xue Bao/Journal of Software* 29(10) (2018), 1–29.

[12] John B Hampshire and Barak Pearlmutter. 1991. Equivalence proofs for multilayer perceptron classifiers and the Bayesian discriminant function. In *Connectionist Models*. Elsevier, 159–172.

[13] Baoli Han, Ling Chen, and Xiaoxue Tian. 2018. Knowledge based collection selection for distributed information retrieval. *Information Processing and Management* 54, 1 (2018), 116–128.

[14] Stefan Heindorf, Martin Potthast, Benno Stein, and Gregor Engels. 2016. Vandalism Detection in Wikidata. In *ACM International on Conference on Information and Knowledge Management*. 327–336.

[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[16] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence* 194 (2013), 28–61.

[17] Paul James. 2015. Despite the terrors of typologies: the importance of understanding categories of difference and identity. *Interventions* 17, 2 (2015), 174–195.

[18] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding via Dynamic Mapping Matrix. In *Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*. 687–696.

[19] Shengbin Jia, Maozhen Li, Yang Xiang, and Others. 2018. Chinese Open Relation Extraction and Knowledge Base Establishment. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* 17, 3 (2018), 15.

[20] Diederik Kingma and Jimmy Ba. 2014. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014), 1–13. https://doi.org/10.1109/ICCE.2017.7889386 arXiv:1412.6980

[21] J. Lehmann. 2015. DBpedia: A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[22] Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. 2016. Commonsense Knowledge Base Completion. In *Meeting of the Association for Computational Linguistics*. 1445–1455.

[23] Jiaqing Liang, Yanghua Xiao, Yi Zhang, Seung-won Hwang, and Haixun Wang. 2017. Graph-Based Wrong IsA Relation Detection in a Large-Scale Lexical Taxonomy.. In *AAAI*. 1178–1184.

[24] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling Relation Paths for Representation Learning of Knowledge Bases. (2015). https://doi.org/10.18653/v1/D15-1082 arXiv:1506.00379

[25] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence Learning* (2015).

[26] Linyuan Lü and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications* 390, 6 (2011), 1150–1170.

[27] Denis Lukovnikov, Asja Fischer, and Jens Lehmann. 2017. Neural Network-based Question Answering over Knowledge Graphs on Word and Character Level. In *International Conference on World Wide Web*. 1211–1220.

[28] Michel V Manago and Yves Kodratoff. 1987. Noise and knowledge acquisition. In *International Joint Conference on Artificial Intelligence*. 348–354.

[29] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *IJCAI International Joint Conference on Artificial Intelligence* (2013). https://doi.org/10.1016/j.physletb.2012.02.070 arXiv:1301.3781

[30] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in neural information processing systems*. 3111–3119. https://doi.org/10.1162/jmlr.2003.3.4-5.951 arXiv:1310.4546

[31] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2015. A Review of Relational Machine Learning for Knowledge Graph. *Proc. IEEE* (2015). https://doi.org/10.1109/JPROC.2015.2483592 arXiv:1503.00759

[32] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In *ICML*.

[33] L Page, S Brin, R Motwani, and T Winograd. 1998. The PageRank citation ranking:bringing order to the web. *Technical report, Stanford Digital Library Technologies Project* (1998). https://doi.org/SIDL-WP-1999-0120

[34] Giuseppe PirrÃš. 2015. Explaining and Suggesting Relatedness in Knowledge Graphs. In *International Semantic Web Conference*.

[35] Baoxu Shi and Tim Weninger. 2016. Discriminative predicate path mining for fact checking in knowledge graphs. *Knowledge-Based Systems* 104, C (2016), 123–133.

[36] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[37] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15, 1 (2014), 1929–1958. https://doi.org/10.1214/12-AOS1000 arXiv:1102.4807

[38] Kristina Toutanova, Xi Victoria Lin, Wen-Tau Yih, Hoifung Poon, and Chris Quirk. 2016. Compositional Learning of Embeddings for Relation Paths in Knowledge Bases and Text. In *Proceedings of the 54nd Annual Meeting on Association for Computational Linguistics (ACL 2016)*. 1434–1444. arXiv:1611.07232

[39] Théo Trouillon, Christopher R Dance, Éric Gaussier, Johannes Welbl, Sebastian Riedel, and Guillaume Bouchard. 2017. Knowledge graph completion via complex tensor factorization. *The Journal of Machine Learning Research* 18, 1 (2017), 4735–4772.

[40] John C Turner and Penelope J Oakes. 1986. The significance of the social identity concept for social psychology with reference to individualism, interactionism and social influence. *British Journal of Social Psychology* 25, 3 (1986), 237–252.

[41] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. *AAAI Conference on Artificial Intelligence* (2014). arXiv:1506.00379

[42] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Does William Shakespeare REALLY Write Hamlet? Knowledge Representation Learning with Confidence. *arXiv preprint arXiv:1705.03202* (2017).

[43] Z L Xu, Y P Sheng, L R He, and Y F Wang. 2016. Review on knowledge graph techniques. *Journal of University of Electronic Science and Technology of China* (2016).

[44] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. (2014). arXiv:1412.6575

[45] Tao Zhou, Jie Ren, Matúš Medo, and Yi Cheng Zhang. 2007. Bipartite network projection and personal recommendation. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* (2007). https://doi.org/10.1103/PhysRevE.76.046115 arXiv:0707.0540

A Tree-Structured Neural Network Model for Household Energy Breakdown

Yiling Jia
University of Virginia
Charlottesville, VA
yj9xs@virginia.edu

Nipun Batra
IIIT Gandhinagar
Gandhinagar, India
nipun.batra@iitgn.ac.in

Hongning Wang
University of Virginia
Charlottesville, VA
hw5x@virginia.edu

Kamin Whitehouse
University of Virginia
Charlottesville, VA
whitehouse@virginia.edu

ABSTRACT

Residential buildings constitute roughly one-fourth of the total energy use across the globe. Numerous studies have shown that providing an *energy breakdown* increases residents' awareness of energy use and can help save up to 15% energy. A significant amount of prior work has looked into source-separation techniques collectively called non-intrusive load monitoring (NILM), and most prior NILM research has leveraged high-frequency household aggregate data for energy breakdown. However, in practice most smart meters only sample hourly or once every 15 minutes, and existing NILM techniques show poor performance at such a low sampling rate.

In this paper, we propose a TreeCNN model for energy breakdown on low frequency data. There are three key insights behind the design of our model: i) households consume energy with regular temporal patterns, which can be well captured by filters learned in CNNs; ii) tree structure isolates the pattern learning of each appliance that helps avoid magnitude variance problem, while preserves relationship among appliances; iii) tree structure enables the separation of known appliance from unknown ones, which de-noises the input time series for better appliance-level reconstruction. Our TreeCNN model outperformed seven existing baselines on a public benchmark dataset with lower estimation error and higher accuracy on detecting the active states of appliances.

CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous computing**; *Empirical studies in ubiquitous and mobile computing*.

KEYWORDS

energy breakdown; convolutional neural networks

ACM Reference Format:
Yiling Jia, Nipun Batra, Hongning Wang, and Kamin Whitehouse. 2019. A Tree-Structured Neural Network Model for Household Energy Breakdown. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313405

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW'19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313405

1 INTRODUCTION

Residential buildings constitute roughly one-fourth of the total energy usage across the global [20]. Studies have shown that providing an energy breakdown can motivate behavioral changes, potentially reducing energy consumption by 15% [2].

Various energy breakdown approaches have been proposed since the pioneering work on non-intrusive load monitoring (NILM) [9]. NILM algorithms are designed for high-frequency data (sampling frequencies > 1/60 Hz), and do not apply when dealing with low sampling rates. However, high-frequency sensors are expensive; and smart meter specifications [21] across the world suggest that the largest proportion of smart meters sample at an hourly rate. This urges the need to develop algorithms suited for time series data with lower sampling rates. On the other end of the spectrum, there are approaches providing energy breakdown at a monthly level, e.g., using monthly bills as aggregate energy consumption [4, 6, 7]. The key idea is that common design patterns create a shared structure in residential buildings and give rise to a sparse set of features contributing to energy variations across homes. Matrix factorization [7] and kernel density estimation [6] techniques are introduced to exploit the sparsity structure. Nevertheless, such techniques cannot be directly applied to higher sampling rates, which rapidly increase the dimension of observations and model complexity.

Our extensive data analysis on a large public U.S. residential energy dataset suggests that sparsity and temporal regularity also exist in hourly appliance energy usage, such as the time of a day, day of a week. This motivates us to view such time series data as a high dimensional compound, rather than just a one-dimension sequence. For example, time of a day might differentiate the use pattern of microwaves from other appliances, while the day of a week might indicate usage pattern of dryers. Each of such temporal patterns creates a unique dimension to recognize a particular type of appliance's energy usage in the aggregate energy readings. But it is clearly impossible to manually exhaust such temporal patterns for each appliance beforehand. We appeal to a learning-based solution to automatically extract such patterns from data. We view each temporal pattern as a latent basis of the high dimensional compound, and assume each appliance can be uniquely characterized by a subset of them. The energy use of each appliance can be isolated

from the aggregate readings by applying its corresponding set of bases. For example, at mealtime, the observed energy consumption should more likely come from a microwave than a dryer.

In this paper, we perform household energy breakdown at an hourly rate. We extract the temporal bases and predict the appliance energy consumption from aggregate energy readings via a set of convolutional neural networks (CNN) [15], which are organized in a tree structure. Thus, we name the solution TreeCNN. At each node, a CNN model is placed to reconstruct appliance energy. The root node of the tree takes aggregate energy reading as input and reconstructs its designated appliance's reading as output. The residual, i.e., the difference between its input and output, is passed to the child node as its input. The reconstruction is thus performed by recursively traversing the tree. Such an iterative procedure isolates the appliance model learning in each step while preserving all appliances as a whole. Thus, each appliance's usage pattern is modeled with "refined" aggregate energy consumption to avoid the overshadow magnitude problem. Further, with such a tree structure, the unknown consumption can be modeled as a special appliance to further de-noise the aggregate readings. It is known that finding the optimal tree structure is NP-complete, and thus we introduce a greedy approach to find the tree structure.

We used the public Dataport [17] dataset for evaluation. We compared TreeCNN against nine state-of-the-art baselines and found TreeCNN provides the most promising performance. Our evaluation shows that the tree structure suggested by our greedy approach performs only 4% worse compared to the optimal order found via an exhaustive search.

2 RELATED WORK

The related work in energy breakdown can be broadly classified as: event-based and total-load based learning approaches.

Event-based methods [8] find step changes in the power signal and assign them to different appliances. Such methods are generally used when high sampling frequency is available, as the events cannot be recognized at low frequencies. Besides, they do not work well when appliances change states simultaneously, nor for appliances that have a highly variable power draw like electronics.

Total-load based methods model the aggregate consumption as a sum of constituent loads, while estimating these constituent loads at all sample points. Factorial Hidden Markov Model (FHMM) has been successfully applied to this problem [14], where each appliance is modelled as a Gaussian HMM. However, it only incorporates Markovian-type relationships in power draw and is not suited for capturing repeated patterns. There is a line of work for energy breakdown at a monthly level. The key insight of such approaches is that common design for buildings creates a sparse set of features contributing to energy variation across homes. Matrix factorization [7] and kernel density estimation [6] have been used to exploit such sparsity. But such solutions cannot be directly applied to higher sampling rate, as their model complexity increases exponentially with the sampling frequency. Sparse coding based approaches [13] have been proposed to address these techniques' limitations on hourly data. But all such solutions assume the aggregate equals to the sum of the appliances and thus suffer under practical settings.

Table 1: Energy statistics from Dataport dataset.

	HVAC	Fridge	Dryer	Dishwasher	Microwave
α(min)	5	5	5	5	2
δ_a	230	20	250	55	10
Active	73.9%	97.8%	4.9%	4.1%	11.3%
Max	5099.7	428.6	4364.1	1021.7	980.6
Mean	1162.7	88.6	1303.6	369.5	59.5
Std	800.2	40.2	756.2	206.5	53.1

More recently, neural network based approaches for energy breakdown have been proposed: [11, 12] applied recurrent neural networks (RNN) to capture the time-series dependency of the energy signals sampled at a high frequency. However, a RNN model captures the one-dimension relationships in power draw, but is incompetent to capture other types of temporal dependencies. For example, in the hourly sampled data, appliances like microwave can be well recognized by the time-of-day pattern, while others like dryer is easier to be modelled by day-of-week pattern. Our solution considers time-series energy data as a high dimension compound of various temporal bases, and learns the bases from data to recognize different types of appliances from the aggregate readings.

3 DATA ANALYSIS OF APPLIANCE USAGE PATTERNS

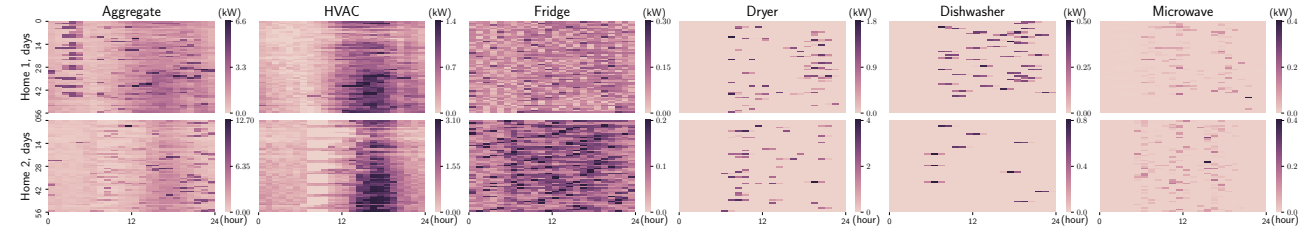
The goal of this section is to explore the temporal patterns of energy consumption in residential buildings towards the development of our proposed energy breakdown method.

In this work, we use the public Dataport [17] dataset, which is the largest public residential home energy dataset. It contains power readings logged at minute intervals from hundreds of homes in the U.S.. We used 112 days worth data from 68 homes from mid-June on-wards for the year 2015, as this period has the least amount of data issues (missing or incorrectly collected data). We use the data of household total consumption and five major appliances: i) air conditioning system (HVAC); ii) fridge; iii) dryer; iv) dishwasher; v) microwave. These appliances contribute significantly to the total consumption. Besides, they also represent a diverse class of appliances: background (fridge) v.s. interactive (microwave), weather dependent (HVAC) v.s. time dependent (dryer, dishwasher).

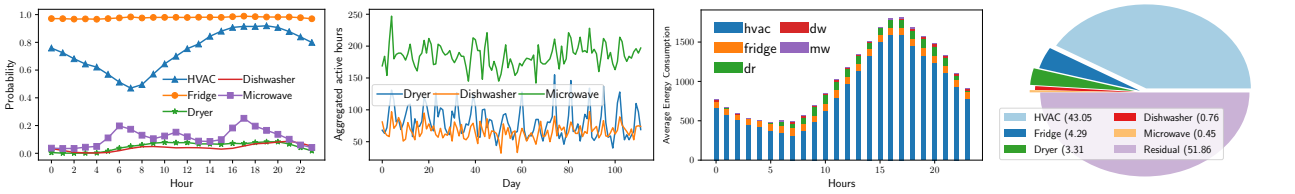
Our focused appliances can generally be classified into two categories [3]: i) appliances that are constantly ON, such as fridge; and ii) ON/OFF appliances, such as washing machine. When dealing with low sampling rates, ON/OFF appliances introduce additional challenges - many of these appliances would only be used partially within an hour, which is the main reason that existing NILM algorithms fail at a low sampling rate. To understand the significance of this phenomenon in our dataset, we studied the shortest active time interval (α) of the 5 appliances (detailed results are in Table 1).

For hourly energy breakdown, the existence of short active intervals begs the question - how much energy should an appliance consume within an hour to be considered as "actively used". On consultation with domain experts, we set the active threshold δ_a for each appliance a as:

$$\delta_a = \frac{\alpha_a}{60} \times \frac{1}{H} \sum_{h}^{H} \max_{d,t} E_{h,a,d,t} \tag{1}$$



(a) Heatmap of aggregate and appliance-level energy consumptions from two randomly selected homes.



(b) Active probability. (c) Aggregated active hours. (d) Average consumption. (e) Consumption proportion.

Figure 1: Data analysis for residential home temporal energy consumption patterns.

where H represents the number of homes, $E_{h,d,a,t}$ is the energy consumed by appliance a on day d at hour t for home h, and α_a is the minimum active time for appliance a. As we know, for appliance occasionally used, it is easy to get good overall performance by giving all zero-predictions (e.g., microwave is OFF over 88% of time). However, such false negative prediction violates the original intention of energy breakdown, i.e., provide the opportunity of energy saving by informing users of how much energy each appliance consumes. With such an active threshold, we can recognize different states of appliances and evaluate a model's performance in two classes, i.e., error in ON/OFF states.

The basic statistics about this dataset with the active threshold are reported in Table 1. The constantly ON appliances, i.e., HVAC and fridge, are almost always on (active percentage: 73.9% and 97.8%); but their energy consumption patterns are different: fridge consumes roughly constant energy over time, while HVAC's consumption varies significantly (std = 800.2). For the ON/OFF appliances, such as dryer, it is seldom used, but once used, it consumes almost the highest energy. This macro-level analysis suggests the need of different temporal bases across appliances.

Previous works [4, 7] show that the energy consumption pattern is sparse owing to the common design of residential buildings. Our data analysis suggests that due to the temporal human behavior patterns, such sparsity also exists at an hourly and daily level. Figure 1a shows the aggregate and five appliances' energy consumption from two randomly sampled homes over 24 hours across 56 days. We can recognize strong patterns within a day across these 56 days: i) both homes tend to consume more energy by HVAC in the afternoon and less in the morning; ii) fridge constantly runs with regular working peaks; and iii) dishwasher and microwave are more likely to be used at the mealtime. Figure 1b, which presents the probability of appliances being in active state during 24 hours, further indicates the hourly patterns. Besides, Home 2 consumes less HVAC energy in the morning and this pattern only appears on the weekdays. Further, people tend to use dryer periodically across days. Figure 1c shows the aggregated active hours across homes in each day for the ON/OFF appliances. It shows that, for dryer, the total number of active hours has a peak every week while dishwasher and microwave are used on an everyday basis.

Besides, energy consumption is highly imbalanced among appliances. "Minor" appliances are often a problem for many existing NILM algorithms owing to their small magnitude of consumption. A detailed comparison is shown in Figure 1d. We can notice that throughout a day, most energy is consumed by HVAC. For those ON/OFF appliances, such as microwave and dryer, when both are on, the one with smaller energy consumption is overshadowed by the larger one, which makes it even harder to differentiate their uses, as "minor" appliances can be confused with "noise". It should be noted that despite low contribution, simply predicting zero use is misleading, and detecting the energy use of these appliances has been shown useful for applications such as elderly monitoring [1].

In addition to the known appliances, it is important to note that the aggregate consumption is often not equal to the sum of the considered individual appliances' consumption. For example, the aggregate consumption of Home 1 shows regular high consumption in the early morning, which is not observed in the known appliances. Such unknown consumption comes from various sources, such as the living room usage, or electric cars. Figure 1e shows the energy consumption proportion of different energy sources, where the unknown consumption can take up 51.86% total energy in a home, and the failure to model them leads an algorithm to classify them to known appliances. To improve the accuracy of energy breakdown, such unknown consumption has to be carefully handled.

4 METHODOLOGY

We study the problem of disaggregating the aggregate energy in a single home to its constituent appliances at hourly intervals. Based on previous discussions, the hourly sampled energy data has several important properties, i) the existence of sparsity and regularity in multiple temporal dimensions, ii) energy consumption magnitude varies significantly across appliances, and iii) the existence of unknown consumption sources. We will discuss our solutions to handle each of them in the following sections.

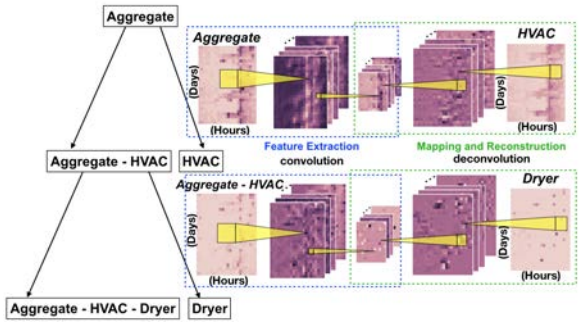


Figure 2: Our tree-structured iterative energy breakdown approach shown for two appliances (HVAC and Dryer).

4.1 TreeCNN Model

The key intuition of TreeCNN lies in two aspects: i) the distinct and multi-dimensional temporal patterns of appliance energy constitute the sparsity and regularity in appliance energy use; ii) the aggregate energy is a composition of various and complicated appliance energy, such that the decomposition should be performed in a joint and recursive manner to avoid the errors introduced by the magnitude problem and potential unknown consumption. We now discuss each component of our proposed TreeCNN model.

• **Convolutional Neural Network (CNN).** Our analysis results show that different appliances have distinct temporal patterns. But, simply modeling the hourly time-series data as a one-dimension sequence cannot fully describe appliances. For example, microwave is more frequently used during the meal time (a hourly pattern), while the dryer is easier to model across days for its periodical usage (a daily pattern). Thus, this time-series data should be viewed as a high dimensional compound of various temporal patterns.

Besides the patterns observed in the appliance usage data, there might also exist other higher order temporal patterns that cannot be simply exhausted manually. Thus, we turn to learning-based solutions to automatically extract the latent bases from data. Inspired by the successful applications of convolutional neural networks (CNN) in image analysis [15], we appeal to CNN models to extract energy usage basis. The key component of CNN model is the filters that capture the spatial features of an image. In an analogy, the hourly energy readings can also be viewed as a 2-D matrix (Figure 1a) and thus can be well described by the spatial filters learned from CNNs.

With CNN model, distinguishable filters can be learned for appliances with distinct temporal patterns. For example, filters learned on microwave may emphasize more on the hour dimension, filters for dryer will emphasize on the day dimension, and the filters for HVAC may be a compound of patterns on both dimensions. With such filters, the aggregate readings can be projected into its corresponding appliance usage. Figure 2 shows an example of a CNN model which learns the mapping from aggregate readings to HVAC consumption. In the convolution phase, CNN model takes the aggregate as input and tries to reduce it to a much denser representation. Due to the sparsity and granularity, the temporal patterns can be extracted and the input will be represented as a denser matrix with a lower dimensionality. In the deconvolution phase, the decoder performs the opposite operations that reverse the action of encoders.

• **Tree Structure.** The model complexity in energy breakdown increases exponentially with the number of sources constituting the aggregate. Further, the usage of some appliances can get overshadowed by others (Figure 1e) creating a "magnitude" problem.

Different from conventional techniques, which either estimate appliance usage independently, or disaggregate the energy altogether at once, we propose a tree-structured model to extract appliance patterns in a "stage-wise manner". With the tree structure, our approach performs an iterative energy breakdown: at each iteration, we subtract out a source from the aggregate and use it as input to recognize the designated appliance. Figure 2 depicts an example of our tree-structured model with HVAC and dryer. The root node takes the aggregate readings as input and reconstructs the HVAC consumption as its output. The difference between them will be passed to the child node as refined input for the next appliance, e.g., dryer in the figure. The magnitude problem is thus eased for the minor appliances, such as microwave, if we place them at the lower end of the tree. In contrast, if we jointly decompose the aggregate readings into appliances' readings, the minor appliances will be overshadowed by major appliances, and mostly given zero predictions, which defeats the original intention of energy breakdown.

In our TreeCNN model, we effectively simplify the energy breakdown iteratively. In each node, the CNN model performs an end-to-end learning for the target appliance, which isolates pattern learning across appliances to avoid the overshadow problem while preserving all appliances as a whole.

• **Modeling Unknown Consumption.** In addition to the magnitude problem caused by the various constitutes of aggregate, the unknown consumption also introduces errors in energy breakdown. From the previous analysis, the unknown energy consumption comes from various sources and therefore is hard to specify beforehand. To the best of our knowledge, no existing work models the unknown consumption. In our tree-structured model, the unknown consumption can be viewed as a special appliance which consists of multi-dimension temporal patterns. Modeling the unknown consumption makes it possible to remove such energy from the true aggregate consumption, which leads to a more accurate estimation of the observed appliances.

4.2 Tree Order

Given N appliances, we would have $N!$ possible tree structures. For a residential home at the U.S., one can usually expect 7-10 major appliances in monitoring. For any larger values of N, exhaustively finding the "optimal" tree order can be computationally expensive, where "optimality" is defined as per given energy breakdown metric M. Since the error of one decomposition will be propagated through the tree structure, the tree order is essential to our model.

We propose a greedy algorithm to find a suitable tree order to mitigate error and reduce the search space. The key operation is to estimate the overall performance with partial information via the inverse propensity weighting scheme [16, 22]. Assume we have N appliances and metric $M(E(a_i), \hat{E}(a_i))$ is used to compute the error for appliance a_i where $\hat{E}(a_i)$ and $E(a_i)$ denote the estimation and the ground-truth of appliance a_i. In the first iteration, we will create N candidate splits. Among these N models, we select k models with the smallest *estimated energy breakdown error* ($EEBE^{GR}$) on the validation set. Next, $(N-1)$ sub-trees are created for each selected

k parent tree. The selection repeats until we have constructed the whole tree. In particular, local metric $EEBE^{GR}$ is used to estimate the overall energy breakdown error, before the whole tree has been constructed. This metric calculates the ratio between a chosen metric for an appliance and the proportion of energy consumed by this appliance:

$$EEBE^{GR}(a_i) = \frac{M(E(a_i), \hat{E}(a_i))}{\frac{\sum_h \sum_d \sum_t E(h, a_i, d, t)}{\sum_h \sum_a \sum_d \sum_t E(h, a, d, t)}} \quad (2)$$

The rationale behind $EEBE^{GR}$ is that it assumes if an appliance has an error e and contributes x proportion to aggregate, then the aggregate would have an expected error of $\frac{e}{x}$ from this appliance. Thus, we can estimate the final error by estimating the prediction error of each appliance during the tree construction. In such way, we get the error of the entire tree before it is constructed, which makes the sequential decisions of tree order possible.

5 EMPIRICAL EVALUATION

In this section, we evaluate the our model on the hourly data collected from 68 homes over 112 days in the Dataport dataset.

5.1 Experimental settings

5.1.1 Baselines. We first describe the baselines

• **Mean Energy:** This baseline computes the predicted energy of an appliance as its mean energy in the training set.

• **Factorial Hidden Markov Model (FHMM):** FHMMs [14] model each appliance as a Gaussian hidden Markov model and couple the individual appliance HMM in a factorial structure.

• **Tensor Factorization:** Canonical polyadic decomposition [4] (CP) is used to factorize the energy tensor into latent matrices. They proposed a modified CP (MCP) to mitigate the scaling problem.

• **Sparse Coding:** Sparse coding [13] model approximates the bases and activations for each appliance with sparsity constraints. The authors also proposed a structured prediction based method called discriminative sparse coding (DSC).

• **Recurrent Neural Networks (RNN):** We performed the decomposition with individual RNN model and TreeRNN model, which captures the time-series dependency of the energy signals.

• **Convolutional Neural Networks (CNN):** We use individual CNNs and JointCNN. Individual CNNs estimates appliances' energy separately, and JointCNN estimates them all together at once.

5.1.2 Approach settings. Among all methods, we used 5-fold cross-validation in the experiments. The final 20% of the train set is set for validation purpose. For each algorithm, the optimal parameters are learned via grid search. The optimal parameters that give the best performance on the validation set are used for testing. For FHMM model, we vary the number of states per appliance from 2 to 5 [5, 23]. CP and MCP are optimized with Adagrad [4], and we vary the rank of latent factors from 1 to 12. For sparse coding models, we vary the rank of latent factors from 1 to 50.

We implemented all neural network models with PyTorch [19]. For RNN models, we have the following parameters: cell type: {GRU, LSTM, RNN}; number of hidden units: {20, 50}; number of layers: {1, 2, 3}; number of iterations: {1000, 2000, 3000}. For CNN models, complex network will easily cause overfitting due to the limited training data. Thus, we have the encoders consist of two convolutional layers and two deconvolutional layers with normalization [10] to accelerate the training process, and ReLU activation function to introduce non-linearity. We choose the learning rate from {0.01, 0.1, 1} and the number of iterations from {1000, 2000, 3000}. For tree-structured models, we perform both exhaustive and greedy search. We use top-k = 3 results at each stage of greedy search. We use the L1-loss as the objective function. For neural network based methods, we clamp the estimated consumption to a maximum of the observed aggregate energy. Our entire codebase, baselines, analysis and experiments can be found on Github [1].

5.1.3 Metric. Based on prior literature [4, 7], we evaluate the performance with mean absolute error (MAE). Denote the ground-truth and estimation for home h, appliance a, day d and hour t as $E(h, a, d, t)$ and $\hat{E}(h, a, d, t)$, for appliance a, MAE is computed as:

$$MAE(a) = \frac{\sum_h \sum_d \sum_t |E(h, a, d, t) - \hat{E}(h, a, d, t)|}{H \times D \times T} \quad (3)$$

where H, D, T indicate the number of homes and days, and hours in a day. We use the average MAE across appliances to measure the model accuracy. Lower mean MAE indicates better performance.

As shown before, in some ON/OFF appliances, the active time is generally low. The MAE alone cannot fully reflect the performance, as zero predictions can also give a good MAE. Thus, we separate MAE into two parts, corresponding to the active and inactive states based on the ground-truth (threshold is reported in Table 1).

5.2 Experiment Results

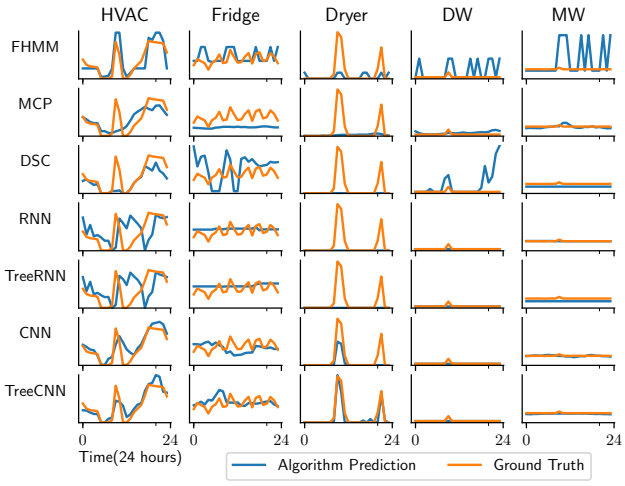


Figure 3: Comparison of baseline algorithms for a sample day. (DW: Dishwahser, MW: Microwave)

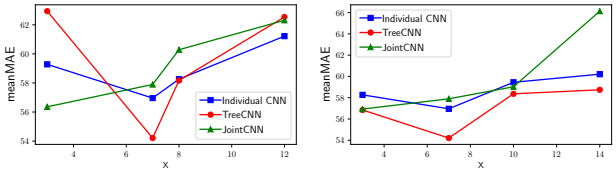
In the following sections, we first test the energy breakdown capabilities in an ideal case, where the aggregate energy equals the sum of selected appliances, and then perform the same experiments on the true aggregate dataset where the unknown consumption is included. Further, we compare the baselines and our TreeCNN model with and without modeling the unknown consumption to

[1] https://github.com/yilingjia/TreeCNN-for-Energy-Breakdown.git

Table 2: MAE on artificial aggregate and true aggregate dataset.

		FHMM	CP	MCP	SC	DSC	Mean	RNN	TreeRNN	CNN	JointCNN	TreeCNN
Artificial Aggregate	MAE	114.99	106.19	103.46	92.20	172.95	100.34	67.34	64.26	56.96	57.90	**51.64***
	MAE on Active	360.76	390.39	390.90	411.93	388.29	404.03	388.90	383.84	310.51	332.45	**261.30***
	MAE on Inactive	80.34	96.11	96.93	28.61	**15.03**	49.13	37.87	36.79	41.53	42.07	40.05
True Aggregate	MAE	134.99	123.90	125.30	245.70	218.43	126.86	97.8	94.52	89.15	91.35	**86.94***
	MAE on Active	400.76	414.01	412.33	515.54	521.35	421.50	43538	433.01	417.85	429.87	**391.50***
	MAE on Inactive	82.94	103.10	107.81	215.57	190.84	102.22	60.32	**54.85**	68.28	69.26	61.69

study the effectiveness of unknown consumption modeling. Last, we report the results of greedy algorithm on tree order estimation.



(a) Filter size: 7 × X **(b) Filter size: X × 7**

Figure 4: Effect of filter size tuning on CNN models.

5.2.1 Filter size in CNN models. In CNN-based models, filter size plays an important role in capturing the temporal patterns. We explored the effect of different filter sizes in the first layer of CNN models. From Figure 4, we can observe that the performance of CNN models is quite sensitive with the filter sizes. When the size equals to 7×7, most models achieve the best performance, as such filters can well capture the patterns across hours and days. With small sized filters, the model might miss some periodical patterns across multiple dimensions; and with large sized filters, it will fail to capture the local features and generate redundant information. In the following experiments, the filters of each layer are set to 7×7 and 2×2. And the decoder is a mirrored version of the encoders with two deconvolutional layers.

5.2.2 Ideal case. Like previous studies [13, 23], we simulate the ideal case by manually setting the artificial aggregate. Shown in Table 2, TreeCNN algorithm outperforms all the baselines (p-value is calculated between the predictions of TreeCNN and the second best model). Figure 3 shows the energy estimation from a set of baselines of a randomly chosen day of one randomly selected home.

FHMM assumes the appliances can be modeled with a Gaussian HMM using discrete states, which considers the transition and emission probabilities between states. But it is not well suited to sparsely used appliances. MCP and DSC algorithms are both of a similar vein focusing on learning various "basis" of energy consumption. Both algorithms performed reasonably well in general. However, they are not well-tuned for the instances when energy consumption patterns differ from the average patterns. And they are poor at capturing the active states of the ON/OFF appliances. From Table 2, we can observe that for MAE on Inactive, SC and DSC have better performance. This is because the design of their two-stage decomposition adds the equality constraint between the sum of appliances and aggregate readings. The mean baseline, though simple, performs reasonably well. However, it does so as it models the appliances to be mostly off and thus a low MAE for the inactive cases, but high MAE for the active cases.

Table 3: Effect of UC modelling. (UC: Unknown Consumption, DW: Dishwasher, MW: Microwave)

	HVAC	Fridge	Dryer	DW	MW	**Average**
TreeRNN w.o. UC	351.83	29.88	66.93	15.45	8.53	94.52
TreeRNN w. UC	337.69	30.02	67.16	15.43	8.43	91.75
TreeCNN w.o. UC	306.38	33.74	70.09	15.49	9.00	86.94
TreeCNN w. UC	296.11	33.34	69.01	15.45	8.83	**84.55**

Table 4: TreeCNN performance under different tree orders. (UC: Unknown Consumption, agg: aggregate)

	Worst	**Average**	**Greedy**	**Best**
Artificial agg	84.42	68.72	54.21	51.64
True agg w.o. UC	105.60	96.54	88.38	86.94
True agg w. UC	110.75	98.64	87.21	84.55

Compared with the non-neural network baselines, the MAE is largely reduced by RNN and CNN based models. Between the individual neural network models, CNN outperforms RNN. The main reason is with the learned filters, CNN can capture the multi-dimensional usage patterns, e.g., hourly pattern and daily pattern in our dataset, while RNN treats the energy readings as a one-dimension time-series ignoring the periodical patterns. For example, in Figure 3 with CNN models, the active states of the ON/OFF appliances, such as dryer, is well detected, while RNN gives zero predictions. From Table 2 and Figure 3, the performance of neural network models are both improved with the tree structure. Though JointCNN also encodes the relationship among appliances, direct decomposition could not overcome the magnitude problem, which generates significant larger error in MAE on Active as the "minor" appliances are easily overshadowed.

5.2.3 Real-world case. Now, we evaluate the models on the true aggregate dataset. From Table 2, we can see that with true aggregate, TreeCNN model still outperforms the other baselines. Comparing the results from these two settings, we can notice that all algorithms show poorer performance with true aggregate. As discussed before, this can be explained by the high amount and variety of unknown consumption. In our TreeCNN, the complex latent bases of unknown consumption can be captured via filters. Table 3 shows the improvement when we consider the unknown consumption in the model. Paired t-test shows almost all the estimations significantly improve except for the dryer. In this work, we only set one model for the unknown consumption, while it might come from a combination of various sources, which we defer to future work.

5.2.4 TreeCNN with greedy v/s exhaustive tree orders. Table 4 compares the mean MAE performance of TreeCNN model with the best

and worst tree orders found by exhaustive search and the order found with greedy search. We also report average MAE over all tree orders explored in the exhaustive search. It shows that our greedy algorithm performs substantially better than the average and is only about 4% worse than the best order found via exhaustive enumeration. The learned tree order also discloses interesting property of this problem: placing HVAC in the first few levels in the tree leads to poorer performance, as the HVAC energy is easily to be over-estimated to "eat" up the energy of other appliances.

6 CONCLUSIONS

In this paper, we presented a new approach for hourly energy breakdown. Our data analysis revealed that hourly energy data has notable high-dimensional sparsity and temporal regularity, which can be exploited for energy breakdown by learning their temporal bases. We introduced a tree-structured CNN model to estimate such temporal patterns and handle some of the shortcomings of existing methods. Empirical evaluation on a real-world household energy data set confirmed the effectiveness of our solution. With the vast amount of hourly smart meter data, we believe our approach has the scope to be scaled to millions of homes.

We would like to explore a few future extensions. First, TreeCNN currently treats the residual as one dummy appliance. However, residual could be a compound of various sources of energy consumption. We can introduce several residual models, or using prior models [18] to first extract latent appliances and generate a combined residual estimation. Second, our current approach does not fully incorporate the dependencies that might exist between different appliances (e.g., correlation between dryer and washing machine). We can incorporate such dependencies by creating additional links between different appliances, giving us a more general graph.

7 ACKNOWLEDGEMENTS

This work is based upon work supported by the National Science Foundation under grant CNS-1646501, IIS-1553568, IIS-1718216, and NVIDIA GPU grant.

REFERENCES

[1] José Alcalá, Oliver Parson, and Alex Rogers. 2015. Detecting anomalies in activities of daily living of elderly residents via energy disaggregation and cox processes. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*.

[2] K Carrie Armel, Abhay Gupta, Gireesh Shrimali, and Adrian Albert. 2013. Is disaggregation the holy grail of energy efficiency? The case of electricity. *Energy Policy* 52 (2013), 213–234. https://doi.org/10.1016/j.enpol.2012.08.062

[3] Sean Barker, Sandeep Kalra, David Irwin, and Prashant Shenoy. 2013. Empirical characterization and modeling of electrical loads in smart homes. In *IEEE IGCC*. Arlington, VA, USA. https://doi.org/10.1109/IGCC.2013.6604512

[4] Nipun Batra, Yiling Jia, Hongning Wang, Kamin Whitehouse, et al. 2018. Transferring Decomposed Tensors for Scalable Energy Breakdown across Regions. *AAAI* (2018).

[5] Nipun Batra, Jack Kelly, Oliver Parson, Haimonti Dutta, William Knottenbelt, Alex Rogers, Amarjeet Singh, and Mani Srivastava. 2014. NILMTK: An Open Source Toolkit for Non-intrusive Load Monitoring. In *Fifth International Conference on Future Energy Systems*. Cambridge, UK. https://doi.org/10.1145/2602044.2602051 arXiv:1404.3878

[6] Nipun Batra, Amarjeet Singh, and Kamin Whitehouse. 2016. Gemello: Creating a Detailed Energy Breakdown from Just the Monthly Electricity Bill. In *SIGKDD 2016*. https://doi.org/10.1145/2939672.2939735

[7] Nipun Batra, Hongning Wang, Amarjeet Singh, and Kamin Whitehouse. 2017. Matrix Factorisation for Scalable Energy Breakdown.. In *AAAI*. 4467–4473.

[8] Mario E Berges, Ethan Goldman, H Scott Matthews, and Lucio Soibelman. 2010. Enhancing electricity audits in residential buildings with nonintrusive load monitoring. *Journal of industrial ecology* 14, 5 (2010), 844–858.

[9] George William Hart. 1992. Nonintrusive appliance load monitoring. *Proc. IEEE* 80, 12 (1992), 1870–1891. https://doi.org/10.1109/5.192069

[10] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*.

[11] Jack Kelly and William Knottenbelt. 2015. Neural NILM: Deep neural networks applied to energy disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. ACM, 55–64.

[12] Jihyun Kim, Thi-Thu-Huong Le, and Howon Kim. 2017. Nonintrusive load monitoring based on advanced deep learning and novel signature. *Computational intelligence and neuroscience* 2017 (2017).

[13] J. Z. Kolter, S. Batra, and A. Y. Ng. 2010. Energy Disaggregation via Discriminative Sparse Coding. In *NIPS 2010*. Vancouver, BC, Canada.

[14] J. Z. Kolter and T. Jaakkola. 2012. Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*. La Palma, Canary Islands.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[16] Roderick JA Little. 1988. Missing-data adjustments in large surveys. *Journal of Business & Economic Statistics* 6, 3 (1988), 287–296.

[17] Oliver Parson, Grant Fisher, April Hersey, Nipun Batra, Jack Kelly, Amarjeet Singh, William Knottenbelt, and Alex Rogers. 2015. Dataport and NILMTK: A building data set designed for non-intrusive load monitoring. In *GlobalSIP 2015*. IEEE.

[18] Oliver Parson, Siddhartha Ghosh, Mark J Weal, and Alex Rogers. 2012. Non-Intrusive Load Monitoring Using Prior Models of General Appliance Types.. In *AAAi*.

[19] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. 2017. Pytorch.

[20] Luis Pérez-Lombard, José Ortiz, and Christine Pout. 2008. A review on buildings energy consumption information. *Energy and buildings* 40, 3 (2008), 394–398.

[21] Staff Report. 2017. 2017 Assessment of Demand Response and Advanced Metering.

[22] Jeffrey M Wooldridge. 2007. Inverse probability weighted estimation for general missing data problems. *Journal of Econometrics* 141, 2 (2007), 1281–1301.

[23] Mingjun Zhong, Nigel Goddard, and Charles Sutton. 2015. Latent Bayesian melding for integrating individual and population models. In *NIPS 2015*.

Improving Neural Response Diversity with Frequency-Aware Cross-Entropy Loss

Shaojie Jiang
University of Amsterdam
Amsterdam, The Netherlands
s.jiang@uva.nl

Pengjie Ren
University of Amsterdam
Amsterdam, The Netherlands
p.ren@uva.nl

Christof Monz
University of Amsterdam
Amsterdam, The Netherlands
c.monz@uva.nl

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

Sequence-to-Sequence (Seq2Seq) models have achieved encouraging performance on the dialogue response generation task. However, existing Seq2Seq-based response generation methods suffer from a *low-diversity problem*: they frequently generate generic responses, which make the conversation less interesting. In this paper, we address the low-diversity problem by investigating its connection with *model over-confidence* reflected in predicted distributions. Specifically, we first analyze the influence of the commonly used Cross-Entropy (CE) loss function, and find that the CE loss function prefers high-frequency tokens, which results in low-diversity responses. We then propose a Frequency-Aware Cross-Entropy (FACE) loss function that improves over the CE loss function by incorporating a weighting mechanism conditioned on token frequency. Extensive experiments on benchmark datasets show that the FACE loss function is able to substantially improve the diversity of existing state-of-the-art Seq2Seq response generation methods, in terms of both automatic and human evaluations.

CCS CONCEPTS

• **Information systems → Users and interactive retrieval**.

KEYWORDS

Chatbot; Dialogue system; Sequence-to-sequence model

ACM Reference Format:
Shaojie Jiang, Pengjie Ren, Christof Monz, and Maarten de Rijke. 2019. Improving Neural Response Diversity with Frequency-Aware Cross-Entropy Loss. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313415

1 INTRODUCTION

Recently, dialogue response generation has attracted a lot of attention due to its potential for applications, e.g., within intelligent customer service agents and personal assistants. Most state-of-the-art approaches to this task are based on *Sequence-to-Sequence* (Seq2Seq) frameworks [3, 11–13, 17, 19]. Current Seq2Seq frameworks suffer from a *low-diversity problem*. As a result, these approaches frequently generate generic responses such as "*I don't know*" or "*I'm sorry*" [3, 11, 12, 19].

To address the issue of low response diversity, previous studies have produced several hypotheses about the cause of the low-diversity problem with corresponding solutions. For example, Li et al. [3] argue that the usual max a posteriori (MAP) objective function might favor frequent responses. Instead, they propose to use a Maximum Mutual Information (MMI) objective function to encourage responses with higher mutual information regarding the user's utterance. The downside of this method is that it relies heavily on beam search and/or an inverse model that is trained by swapping inputs and outputs. Tao et al. [15] assume that the single attention layer commonly used in Seq2Seq models is only able to focus on a single semantic aspect of the input sequence, so they propose a Multi-Head Attention Mechanism (MHAM) to allow the decoder to generate more diverse responses. There are also studies that try to improve response diversity by introducing randomness [12, 19].

Recently, Jiang and de Rijke [1] have shown that there is a strong connection between the low-diversity problem and what they call *model over-confidence* following Pereyra et al. [10], the phenomenon that a model incorrectly assigns most probability to only a few tokens. However, the cause of this phenomenon remains unknown. In §2, we investigate and conclude that the model over-confidence problem is caused by an imbalanced training of tokens with variable frequencies, which favors frequent tokens and results in low-diversity. To correct for this, we propose a *Frequency-Aware Cross-Entropy* (FACE) loss function that improves the traditional *Cross-Entropy* (CE) loss function by taking token frequency into consideration. More specifically, we first analyze the influence of the commonly used CE loss function, and find that it prefers high-frequency tokens, which results in model over-confidence and low-diversity responses. Then we propose a FACE loss function that improves over the CE loss function by incorporating a weighting mechanism conditioned on token frequency.

The primary differences between FACE and previous studies in addressing the low-diversity problem are two-fold: (1) The diversity improvements brought by FACE do not rely on beam search or

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313415

Table 1: Frequency ranks of leading tokens and their percentage (%). Validation rank and Test rank are for the validation and test model responses, Training rank is for the ground-truth responses in the training data.

	i	the	you	we	he	it
Validation rank	1 (74)	2 (5)	3 (5)	4 (3)	5 (3)	6 (3)
Test rank	1 (72)	2 (7)	3 (5)	5 (3)	6 (3)	4 (2)
Training rank	1 (14)	7 (12)	3 (6)	6 (3)	8 (3)	4 (3)

randomness; and (2) FACE does not introduce new layers or hyper-parameters to Seq2Seq models.

We perform extensive experiments on two benchmark datasets, namely OpenSubtitles database (OSDb) and Twitter. Compared with deterministic Seq2Seq-based methods, like MMI [3] and MHAM [15], FACE achieves the highest diversity performance and it does so with minimum modifications to the original Seq2Seq model structure and existing hyper-parameters.[1]

The main contributions we make in this paper are:

- We examine the influence of token frequency on model over-confidence and response diversity.
- We propose a *Frequency-Aware Cross-Entropy* (FACE) loss function to balance the per-token loss, which alleviates model over-confidence and, hence, improves response diversity.
- We investigate two token frequency calculation methods and corresponding frequency-based weighting mechanisms for FACE.

2 LOW DIVERSITY, MODEL OVER-CONFIDENCE AND LOSS IMBALANCE

As illustrated by Jiang and de Rijke [1], model over-confidence [10] can cause Seq2Seq-based conversation models to have low response diversity. In this section, we show that model over-confidence and low-diversity are statistical and empirical symptoms of the same problem. Imbalanced training is the actual underlying reason.

Existing Seq2Seq-based conversation models tend to be over-confident during prediction and place most probability on only a few tokens. We define a *leading* token in a response to be a token that appears first. From Table 1, we can see that all of the most frequent leading tokens in model responses are actually very likely to appear first in training ground-truth responses, but the percentages of the top-ranked token "i" are much higher in the validation and test sets. This suggests that the model is over-fitting for frequent leading tokens. We also find that the frequency of some subsequent tokens (e.g., *'m* and *not*) is much higher in model responses than in the training ground truth, as illustrated in Table 2. This is also due to model over-fitting, since these tokens are *relatively* more frequent compared to other tokens. For example, given that the previous token is "i", the frequency of the following token "'m" is much higher than others in the training data.

To understand this phenomenon and introduce our solution, we first look into the commonly used *max a posteriori* (MAP) objective function of Seq2Seq models. Given a dataset of message-response pairs (X, Y), where $X = (x_1, x_2, \ldots, x_{|X|})$ and $Y = (y_1, y_2, \ldots, y_{|Y|})$ are the input and output sequences, respectively, the goal of Seq2Seq training is to maximize the conditional probability $P(Y|X)$. Since the decoder Recurrent Neural Network (RNN) can only give one output at each time step t, and to generate grammatical responses, we are actually maximizing token-wise probability at each time step during training:

$$\max P(Y|X) = \max \prod_{t=1}^{|Y|} P(y_t|y_{<t}, X), \tag{1}$$

where $y_{<t} = (y_1, y_2, \ldots, y_{t-1})$ are tokens generated in previous time steps. At test time, the response is generated with respect to:

$$\hat{Y} = \arg\max_Y P(Y|X). \tag{2}$$

In practice, we usually maximize the aforementioned conditional probability by minimizing the prediction loss at each step t, which is actually the Cross-Entropy (CE) loss:

$$\mathrm{CE}(y_t) = -\sum_{i=1}^{N} \delta_i(y_t) \log(P(c_i|y_{<t}, X)), \tag{3}$$

where $(c_1, c_2, \ldots, c_N)$ is the search space of y_t, $\delta_i(y_t) = 1$ if $y_t = c_i$ and 0 otherwise; $P(c_i|\cdot)$ is the predicted probability of candidate token c_i and is calculated using the softmax function:

$$P(c_i|y_{<t}, X) = \frac{\exp(f_\theta(h_{t-1}^{dec}, y_{t-1}, c_i, X))}{\sum_{j=1}^{N} \exp(f_\theta(h_{t-1}^{dec}, y_{t-1}, c_j, X))}, \tag{4}$$

where $f_\theta(\cdot)$ is a non-linear scoring function with parameters θ; $f_\theta(\cdot)$ takes the hidden state h_{t-1}^{dec}, last generation y_{t-1} and X as inputs, and calculates a score for each possible candidate c_i.

During training, the loss calculated using Eq. (3) is back-propagated through the whole network. The effect is that f_θ will assign higher scores for $c_i = y_t$ in the future, so that the loss in Eq. (3) will decrease, and meanwhile the predicted probability in Eq. (4) will increase. The ultimate effect is to maximize the probabilities of ground-truth outputs given input sequences, as illustrated in Eq. (1). The model is frequently penalized by a small number of frequent tokens, as a result of which the total loss (TL) for these tokens is higher than for less-frequent ones:

$$\mathrm{TL}(c_i) = \sum_{t=1}^{N_t} \mathrm{CE}(y_t = c_i), \tag{5}$$

where t denotes the time step, with maximum training steps N_t:

$$N_t = \sum_{i=1}^{N} \mathrm{freq}(c_i). \tag{6}$$

Here, $\mathrm{freq}(c_i)$ represents the frequency that c_i appears in the training ground truth. Assume that the expected value $\mathbb{E}[\mathrm{CE}(c_i)]$ is roughly the same for $c_i, \forall i \in \{1, 2, \ldots, N\}$, then tokens with a higher frequency will have a larger total loss during training. We refer to this phenomenon as *loss imbalance*.

Due to loss imbalance, a Seq2Seq model favors frequent tokens and thus is over-confident about them. This is especially true for the leading token as observed in Table 1, since the decoder language

Table 2: Frequency ranks of example tokens and their percentage (%) in validation and test model outputs, and ground truth outputs of the training data.

	.	i	'm	not	n't
Validation rank	1 (11)	2 (10)	3 (5)	4 (5)	8 (3)
Test rank	1 (10)	2 (9)	3 (5)	4 (5)	8 (3)
Training rank	1 (7)	4 (3)	35 (1)	32 (1)	13 (1)

[1]Source code for both FACE and the baselines, together with the validation and test sets used in our experiments can be found at https://github.com/ShaojieJiang/FACE.

model does not have a strong effect in the beginning of prediction. When a frequent token is selected as leading token, the search space for subsequent tokens is hugely restricted by the language model, and this will likely result in a frequent *generic* response.

3 FREQUENCY-AWARE CROSS-ENTROPY LOSS

Now that we have identified *loss imbalance* as a cause of low-diversity problems, we propose to balance the total loss for each token by applying a weight factor to TL:

$$\text{WTL}(c_i) = w_i \sum_{t=1}^{N_t} CE(y_t = c_i), \tag{7}$$

where w_i is the weight corresponding to c_i. By absorbing w_i into the CE loss function, we obtain the FACE loss function:

$$\text{FACE}(y_t) = -\sum_{i=1}^{N} w_i \delta_i(y_t) \log(P(c_i|y_{<t}, X)). \tag{8}$$

Our key solution to the low-diversity problem lies in the weight factors w_i. Based on the analysis in §2, one straightforward way to learning w_i is to take advantage of the token frequency $\text{freq}(c_i)$, so that frequent tokens will have lower weights. Below we propose two methods to estimate $\text{freq}(c_i)$: *ground-truth or GT frequency* and *output frequency*.

GT frequency: Token frequency in the ground-truth responses. As illustrated in Eq. (6), the number of training steps N_t equals the sum of frequencies of tokens in the training ground-truth responses, so it is intuitive to use these token frequencies to adjust the weight in Eq. (8). However, during training, the model is given data sequentially in random order. As a result, the real-time token frequency seen by the model is likely to differ from that of the entire training data. Our solution is to calculate the *batch* token frequency instead:

$$\text{freq}_b(c_i) = \text{freq}_{b-1}(c_i) + \text{freq}_o(c_i), \tag{9}$$

where b is the number of training batches seen so far and o represents the newly *observed* batch.

Output frequency: Token frequency in model responses. Alternatively, we can employ a train-and-refine strategy: the responses of a pre-trained Seq2Seq model can reflect which tokens the model is already overfitted for; by directly penalizing those tokens with a fine-tuning procedure, we can improve the response diversity without retraining it from scratch. The output frequency may have a more obvious effect on improving response diversity than the GT frequency, because diversity is directly exhibited by model outputs.

3.1 Weight calculation

Given the frequency of each token, we introduce the following two methods to calculate the weight factor.

3.1.1 Pre-weight. This method derives the weight factor w_i prior to seeing new training examples, i.e., *pre-weight function*:

$$w_i = a \times RF_i + 1. \tag{10}$$

Here we formulate it as a linear function of *relative frequency*: $RF_i = \text{freq}(c_i)/\sum_j \text{freq}(c_j)$, and $a = -1/\max RF_j, \forall j \in \{1, \ldots, N\}$, is the slope, and the bias is 1 so that w_i falls in $[0, 1]$. We then normalize $\{w_1, w_2, \ldots, w_N\}$ to have a mean of 1. The pre-weight function can make sure that tokens with a higher RF value will get lower weights. In other words, the influence of high frequency tokens will be penalized by using the pre-weight function.

3.1.2 Post-weight. This method tries to penalize the model's *conservativeness*: if the output token y_t has a higher frequency than the ground truth c_i, which indicates the model conservatively picked a "safe" token, then its loss will be scaled up by $w_i > 1$, otherwise $w_i = 1$ due to ReLU activations:

$$w_i = 1 + \frac{ReLU(\text{freq}(y_t) - \text{freq}(c_i))}{\sum_j^N \text{freq}(c_j)}. \tag{11}$$

Since we can only apply this weighting function after obtaining the model outputs, we refer to it as the *post-weight function*.

3.2 Empowered by confidence penalty

As illustrated in [1], *Confidence Penalty* (CP) methods can alleviate the low-diversity problem. In the experiments of this paper, we also test the performance of the CP function [1]:

$$\text{CP}(y_t) = \text{CE}(y_t) - \beta H(p(y_t|y_{<t}, X)), \tag{12}$$

where $p(y_t|y_{<t}, X)$ is the predicted distribution at t, and $H(\cdot)$ is its entropy. However, during experiments we found that the parameter β needs to be carefully chosen, otherwise the loss $\text{CP}(y_t)$ will be negative, which is counter-intuitive since losses should be greater than 0. Instead, we propose a *parameter-free* CP function:

$$\text{CP}_{free}(y_t) = \text{CE}(y_t) + \frac{1}{H(p(y_t|y_{<t}, X))}. \tag{13}$$

FACE and CP can be easily combined to further improve response diversity. A trivial combination is to replace the CE loss function with the FACE loss function in Eq. (12) and Eq. (13). We also propose a CP weighting function using the entropy in Eq. (13):

$$w = 1 + \frac{1}{H(p(y_t|y_{<t}, X))}, \tag{14}$$

where w (without subscript) is assumed to be independent of c_i, which is different from that in Eq. (10) and Eq. (11). By using w as the weight of FACE in Eq. (8), we can penalize the model confidence by adjusting the weight of FACE.

4 EXPERIMENTAL SETUP

To prove the effectiveness of our proposed methods, we design experiments to answer the following questions: (Q1) Which combination of our proposed frequency methods (GT frequency and output frequency) and weighting functions (pre- and post-weight) performs best? (Q2) Does FACE improve the diversity of Seq2Seq conversation models? (Q3) Does CP improve the diversity of Seq2Seq conversation models? (Q4) Does the combination of FACE and CP further improve performance? (Q5) Does FACE improve the response quality besides diversity?

4.1 Baselines and datasets

The first baseline is a vanilla Seq2Seq model with *general* attention [6] as implemented within the ParlAI platform [7]. We also choose the MMI models proposed in [3] and the MHAM models proposed in [15]. These models are all deterministic methods. In Table 3, we

[2]The MMI-bidi method needs a reverse model which is pre-trained using inverse training examples: (response, message).

Table 3: Components of different models.

Model	Greedy	BS	#Attn	Extras
Seq2Seq	Yes	No	1	No
MMI	No	Yes	1	Reverse[2]
MHAM	Yes	No	5	No
FACE	Yes	No	1	No
CP	Yes	No	1	No

list the main differences between our methods and the baselines in terms of greedy decoder, beam search (BS), number of attention layers (#Attn), and other mandatory components (Extras). To allow for a fair comparison, we implement all our methods and baselines using ParlAI. We choose two publicly available benchmark datasets: OSDb and Twitter, for evaluating the baselines and our proposed methods. We follow Li et al. [3], Tao et al. [15] to use short-history conversations; see below for details.

4.1.1 OSDb. The OSDb dataset [16] is an online-available corpus of movie subtitles.[3] Here, we use the 2011 version and set aside ~60M lines, which constitutes ~30M message-response pairs for training. Following Li et al. [3], we randomly select 2K pairs from the IMSDB dataset [18] for validation and test sets, respectively, and we filter out pairs whose responses or messages are shorter than 6 tokens.

4.1.2 Twitter. Twitter is a commonly used source of data in dialogue generation research. For ease of reproducibility of the results reported in this paper, we did not follow Li et al. [3], Sordoni et al. [13] who used ~130M context-message-response triples before preprocessing. Instead, we use the version released by [12] for training, which contains ~4M tweet IDs in total and can be scraped in 3 days.[4] After formatting the tweets as *context-message-response* triples, we have 904K training examples and we concatenate context-message as input. For the training, validation and test sets, we exclude all overlapping IDs. Then we restrict the sequence length in both validation and test sets to the range [6, 18], resulting in 18,162 validation and 1,897 test triples. Furthermore, since tuning the MMI models is quite time-consuming, we randomly select 2K triples from the validation set for hyper-parameter selection.

4.2 Evaluation

4.2.1 Automatic evaluation. Although reported not to correlate well with human judgments [5], we still report BLEU scores [8] for fair comparisons with baselines. To evaluate response diversity, we use the *d-1* and *d-2* metrics proposed in [3] that are calculated as the number of distinct uni- and bigrams, divided by the total number of tokens generated.

4.2.2 Human evaluation. Following Sordoni et al. [13], we recruit crowd-source workers to perform pairwise qualitative comparisons. Since the conversation history of the OSDb dataset is only a single turn, which is hard for human annotators to judge, we choose the Twitter dataset for human evaluation, which has two-turn histories. We randomly select 1K test examples from the Twitter test set, and paired model responses (*FACE* vs. *baseline*) are shown to 3 evaluators in random order. Evaluators are told to choose a better response in terms of *relevance*, *interestingness* and *grammar*. Ties are allowed. We then carry out Welch's t-test on the human preferences obtained in this manner.

[3] http://www.opensubtitles.org/

[4] On July 11 2018, when we finished scraping, only 2.6M IDs in total are still valid.

Table 4: Performance (%) on the OSDb dataset of different variants of FACE. Highest scores in bold face.

Model	d-1	d-2	BLEU
FACE-OPR	4.32	**20.47**	**8.03**
FACE-OPO	4.56	14.96	6.76
FACE-GPR	2.87	10.51	7.56
FACE-GPO	**5.03**	19.66	6.92

Table 5: Performance (%) on the Twitter dataset of different variants of FACE. Highest scores are in bold face.

Model	d-1	d-2	BLEU
FACE-OPR	**6.23**	**24.18**	8.33
FACE-OPO	5.73	17.95	**8.80**
FACE-GPR	4.13	15.03	7.99
FACE-GPO	5.69	17.78	8.72

4.3 Implementation details

For the OSDb and Twitter datasets, we keep the most frequent 25,000 tokens and replace other tokens with *_UNK_*. For the OSDb corpus, we use a 4-layer Long Short-Term Memory (LSTM) network for both encoder and decoder. The hidden size (HS) and word embedding size are set to 1,000. On the smaller Twitter corpus, we use a smaller network with a 2-layer LSTMs for encoder and decoder. The HS and word embedding sizes are set to 512 and 200, respectively. For both networks, we randomly initialize the model parameters from a uniform distribution $\mathcal{U}(-\sqrt{1/\text{HS}}, \sqrt{1/\text{HS}})$ and the word embeddings from a normal distribution $\mathcal{N}(0, 1)$. We also employ dropout [14] with drop ratio $p = 0.1$. We use Adam [2] as our optimization method. For the hyper-parameters of the Adam optimizer, we set the learning rate $\alpha = 0.001$, two momentum parameters $\beta 1 = 0.9$ and $\beta 2 = 0.999$, respectively, and $\epsilon = 10^{-8}$. We also clip gradients [9] to 5 during training to avoid gradient explosion. To speed up training and convergence, we use mini-batches of size 256. Since the model chosen with minimum training loss usually has very low diversity, we choose *d-1* as the early stopping criterion. A scheduler is used to reduce the learning rate by a factor of 0.5 when a *d-1* plateau is detected with $patience = 3$. β in Eq. (12) is set to 0.01.

5 RESULTS AND ANALYSES

5.1 FACE variants

To answer Q1, we first identify the best performing variant of FACE to be used in later experiments. In Tables 4 and 5, we list the scores for four variants of FACE, on the OSDb and Twitter datasets, respectively: Output token frequency & PRe-weight (*FACE-OPR*), Output token frequency & POst-weight (*FACE-OPO*), GT frequency & PRe-weight (*FACE-GPR*) and GT frequency & POst-weight (*FACE-GPO*).

Table 6: Results (%) on the OSDb dataset. Highest scores in each column are highlighted using bold face.

Model	d-1	d-2	BLEU
Seq2Seq	2.70	8.63	7.34
Seq2Seq-refine	3.61	14.16	7.61
MMI-antiLM	2.73	10.68	7.83
MMI-bidi	3.06	12.19	7.01
MHAM	3.03	9.47	7.13
CMHAM	4.10	12.92	6.88
FACE	4.32	**20.47**	**8.03**
CP	4.18	15.59	7.27
CP_{free}	**5.48**	18.59	7.08
FACE-CP	4.63	18.32	7.89
FACE-CP$_{free}$	4.69	18.67	6.98

Table 7: Results (%) on the Twitter dataset. Highest scores in each column are highlighted using bold face.

Model	d-1	d-2	BLEU
Seq2Seq	5.87	17.73	8.77
Seq2Seq-refine	5.69	17.54	8.82
MMI-antiLM	**6.23**	18.21	6.51
MMI-bidi	5.42	15.16	8.20
MHAM	5.52	17.04	**8.96**
CMHAM	4.99	14.91	8.71
FACE	**6.23**	**24.18**	8.33
CP	5.97	18.15	8.84
CP_{free}	6.00	18.67	8.82
FACE-CP	6.07	23.50	8.25
FACE-CP$_{free}$	5.89	17.81	8.85

On the OSDb dataset, we can see from Table 4 that *FACE-OPR* and *FACE-GPO* perform better than the other two in terms of diversity scores. Similar results are shown on the Twitter dataset in Table 5. By comparing the results in Table 4 and 5, we find that the performance of *FACE-OPR* is very stable. We therefore choose *FACE-OPR* as our primary model and refer to it as *FACE* in the following sections. In contrast, *FACE-GPR* performs worst on both datasets, which indicates that GT frequency does not work well with pre-weight. The BLEU scores in Table 4 and 5 show no obvious patterns.

It is worth noting that all variants reported in Table 4 and 5 are using the train-and-refine strategy: first training the Seq2Seq model with CE, then fine-tuning it with FACE. We also tried training from scratch using FACE. However, the performance did not show consistent improvements, which is probably because our *equal expected loss* assumption (under Eq. (6)) is violated in the early training stages.



Figure 1: Word cloud showing top-200 frequent tokens of model responses on the OSDb dataset. Left: the larger the font, the higher the frequency. Right: the larger the font, the higher the weight (pre-weight used). (Best viewed in color.)

5.2 Automatic evaluation

We turn to Q2, Q3, and Q4. The automatic evaluation results are shown in Tables 6 and 7. *Seq2Seq* is vanilla-Seq2Seq with *general* attention mechanism [6]. *Seq2Seq-refine* is a fine-tuned version of *Seq2Seq* with a smaller batch size of 30. *MMI-antiLM* adjusts the *Seq2Seq* prediction probability using the decoder language model, while *MMI-bidi* utilizes a reverse model. Both methods use beam size 200. *MHAM* projects encoder hidden-states to 5 different semantic spaces, and *CMHAM* forces those 5 spaces to be perpendicular to each other. All of our models are fine-tuned *Seq2Seq* models with a smaller batch size of 30. *FACE* is fine-tuned using the FACE function and output token frequency. *CP* and CP_{free} are fine-tuned using Eq. (12) and (13), respectively. *FACE-CP* is a linear combination of FACE and CP. *FACE-CP*$_{free}$ is the multiplicative combination of FACE and the CP weighting function in Eq. (14).

As shown in Table 6, all the methods proposed in this paper outperform the baselines in terms of diversity metrics on the OSDb dataset. *FACE* achieves the highest d-2 score (increase of 6.3%). *FACE-CP* and *FACE-CP*$_{free}$ achieve slightly higher d-1 and lower d-2 scores than *FACE*, which can be viewed as the trade-off between *FACE* and *CP*'s. Although the penalty strength β of *CP* is carefully selected, it is interesting to see that our hyper-parameter free method CP_{free} outperforms *CP* by a large margin and achieves the highest d-1 score (1.9% improvement), which demonstrates the effectiveness of Eq. (13). *FACE* performs best in terms of BLEU scores, which suggests that *FACE* can generate higher-quality responses.

Although the MMI-based and MHAM-based methods all exhibit various improvements over *Seq2Seq* in terms of d-1 and d-2 scores, they mostly perform worse than *Seq2Seq-refine*. This suggests that on the OSDb dataset, carefully designed MMI-based and MHAM-based methods are not able to outperform a fine-tuned *Seq2Seq* baseline.

Similar to Table 6, Table 7 shows that all our methods can increase the diversity of *Seq2Seq* on the Twitter dataset. *FACE* achieves the highest d-1 score (0.4% increase) together with *MMI-antiLM*, but the highest d-2 score of *FACE* (6.5% improvement) demonstrates that our method fares better.

In Table 7 however, *Seq2Seq-refine* decreases the diversity of *Seq2Seq*, indicating that fine-tuning does not help to address the low-diversity problem on the Twitter dataset. This is probably because

Table 8: Results of the pairwise human evaluation (%) on the Twitter dataset. "Win", "Lose" and "Gain" correspond to "FACE wins", "Baseline wins" and their difference (*Win* − *Lose*), respectively. Highest scores are highlighted in bold face, and *,**,*** symbols indicate significant improvements with *p-value* $< 0.05, < 0.01, < 0.005$, respectively.

Comparison	Win	Lose	Gain
FACE vs Seq2Seq	**38.61*****	21.54	17.07
FACE vs MMI-antiLM	**51.30*****	19.35	31.95
FACE vs MMI-bidi	**61.91*****	20.92	40.99
FACE vs MHAM	**50.93****	42.56	8.37
FACE vs CMHAM	**43.75***	38.85	4.90

the diversity of *Seq2Seq* is already relatively high on this dataset, and there is limited space for further improvements.

While *MMI-bidi* performs better than *MMI-antiLM* on the OSDb dataset, on the Twitter dataset, however, *MMI-bidi* performs much worse and degrades the diversity of the *Seq2Seq* model. The reason for this phenomenon is probably because of the reverse model of *MMI-bidi*: the input consisting of Twitter triples contains two turns from different speakers (*context-message*), thus given the *response* only, the prediction probability of *context-message* is very unreliable.

Although it achieves the highest BLEU score, *MHAM* slightly hurts diversity. Similarly, *CMHAM* degrades the diversity even more. Closer inspection reveals that most of the attention weights of both methods are on the first several tokens of the input sequence (i.e., *context*). It is unlikely to be able to properly learn the relation between *context-message* and *response* by only paying attention to the *context*.

To illustrate the effectiveness of the weighting function, we display the frequencies and the corresponding weights of some tokens in Figure 1. From this figure, we can see that the most frequent tokens have very small weights, and less frequent tokens receive larger weights. Please note that the tokens with the largest weights are in the scope of top-200 frequent tokens.

5.3 Human evaluation

We now turn to Q5. Human evaluation results are reported in Table 8. We can see that *FACE* is significantly better than all baselines. This means that by increasing response diversity, FACE can improve relevance and interestingness of responses, without sacrificing grammatical accuracy. Specifically, by penalizing frequent tokens, *FACE* gives more opportunities to less frequent tokens, so the interestingness is higher than *Seq2Seq*. While high-frequency tokens may be appropriate responses in more situations than low-frequency ones, it is for the same reason that higher frequency tokens convey less information, thus making the responses seem generic. Therefore, by encouraging low-frequency tokens, *FACE* has the potential to increase relevance of responses. By looking at the human evaluation scores in Table 8 together with the BLEU scores in Table 7, we can re-confirm that BLEU scores do not correlate well with human qualitative evaluation [5].

Since *FACE* learns a language model relying entirely on training data, the grammatical accuracy of *FACE* is as least as good as that of *Seq2Seq*. The examples in Table 9 show that *FACE* can actually

Table 9: Example responses from different models.

History	Responses
Context: i need 4 mo followers ..	***FACE***: no i do n't think so .. i 'm watching it now ..
Message: u ever watched criminal minds suspect behavior??	***Seq2Seq***: i do n't know what to say ..
Context: dont you dare steal my idea. i call shenanigans	***FACE***: well then you should have said it like that .
Message: Totally not stealing. Everyone collaborated on that idea.	***MMI-antiLM***: come on . come on go back to you*
Context: I never watched Spongebob's movie tho, was it good? You know Hey Arnold?	***FACE***: haha , i 'm watching it now , it 's so good : d
Message: yes it is good u should watch :D haha hey arnould with footballegg face :P	***MMI-bidi***: oh , that
Context: perhaps a recipe is in order? I have room in my fridge... ;)	***FACE***: sounds good ! i 'll be sure to check it out ! thanks for sharing the mocha cakes with the butter cakes ;)
Message: trying this one for 1st time LOL http://bit.ly/nrFMIU start from bottom recipe & work up	***MHAM***: i 'm sure you 'll find a recipe for me !
Context: Don't laugh at my picture! That's not nice.	***FACE***: i 'm not a fan of the picture but it 's not that bad , it 's just a pic of me saying " i 'm a girl " haha : -)
Message: i'm laughing because you actually posted the pic. I believed you:-) lol	***CMHAM***: i 'm not laughing at you .

generate more relevant and interesting responses. In contrast, although *MMI-antiLM* improves response diversity, it increases the risk of grammatical errors since it penalizes the language model. For example, the response of *MMI-antiLM* in the second example of Table 9 is ungrammatical and irrelevant to the conversation history. Similarly, *MMI-bidi* uses a reverse model to adjust the model outputs, in the hope that the responses can be more diverse and relevant. However, as explained earlier, the reverse model is highly unreliable, making many of the responses look weird, see, e.g., the third row of Table 9.

Although *MHAM* and *CMHAM* can improve response diversity to some extent, it is not guaranteed that the multiple attention heads are well distributed throughout the whole history. In fact, there are many cases where the attention weights are on the **Context** part of the history, making the responses irrelevant to the **Message**, such as the fourth and fifth examples in Table 9.

6 CONCLUSION

In this work, we have proposed a Frequency-Aware Cross-Entropy (FACE) function for tackling the low-diversity problem of Seq2Seq-based conversation models. Experiments on short-history conversation datasets demonstrate that the FACE loss function can effectively improve the diversity and quality of responses. FACE achieves the improvements with minimum modifications to the original Seq2Seq model, which makes it flexible to extend. We also propose a hyper-parameter free CP_{free}, which exhibits better performance than the original parameter-dependent CP.

A limitation of FACE is that the learning procedure is not as stable as CE, which increases the difficulty of training. In future work, we would like to investigate this phenomenon in depth. We also hope to test FACE in a long-history conversation setting by applying FACE to hierarchical Seq2Seq [11]. Besides, we also plan

to apply FACE to stochastic models [12, 19] and examine how FACE can be used in an adversarial dialogue generation setup [4].

ACKNOWLEDGMENTS

This research was supported by the China Scholarship Council, Ahold Delhaize, the Association of Universities in the Netherlands (VSNU), and the Innovation Center for Artificial Intelligence (ICAI). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Shaojie Jiang and Maarten de Rijke. 2018. Why are Sequence-to-Sequence Models So Dull? Understanding the Low-Diversity Problem of Chatbots. In *Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd International Workshop on Search-Oriented Conversational AI (SCAI '18)*.

[2] Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR '15)*.

[3] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT '16)*. 110–119.

[4] Ziming Li, Julia Kiseleva, and Maarten de Rijke. 2019. Dialogue generation: From imitation learning to inverse reinforcement learning. In *AAAI 2019: 33rd AAAI Conference on Artificial Intelligence*. AAAI.

[5] Chia-Wei Liu, Ryan Lowe, Iulian Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP '16)*. 2122–2132.

[6] Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP '15)*. 1412–1421.

[7] Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A Dialog Research Software Platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP '17)*. 79–84.

[8] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL '02)*. 311–318.

[9] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the Difficulty of Training Recurrent Neural Networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML '13)*. 1310–1318.

[10] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. 2017. Regularizing Neural Networks by Penalizing Confident Output Distributions. In *International Conference on Learning Representations (ICLR '17)*.

[11] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI '16)*. 3776–3784.

[12] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI '17)*. 3295–3301.

[13] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT '15)*. 196–205.

[14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[15] Chongyang Tao, Shen Gao, Mingyue Shang, Wei Wu, Dongyan Zhao, and Rui Yan. 2018. Get The Point of My Utterance! Learning Towards Effective Responses with Multi-Head Attention Mechanism. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI '18)*. 4418–4424.

[16] Jörg Tiedemann. 2009. News from OPUS-A Collection of Multilingual Parallel Corpora with Tools and Interfaces. In *Recent Advances in Natural Language Processing (RANLP '09)*, Vol. 5. 237–248.

[17] Oriol Vinyals and Quoc V. Le. 2015. A Neural Conversational Model. In *ICML Deep Learning Workshop*.

[18] Marilyn A Walker, Ricky Grant, Jennifer Sawyer, Grace I Lin, Noah Wardrip-Fruin, and Michael Buell. 2011. Perceived or Not Perceived: Film Character Models for Expressive NLG. In *International Conference on Interactive Digital Storytelling (ICIDS '09)*. 109–121.

[19] Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL '17)*. 654–664.

A Novel Generative Topic Embedding Model by Introducing Network Communities

Di Jin
College of Intelligence and Computing, Tianjin University, China
jindi@tju.edu.cn

Jiantao Huang
College of Intelligence and Computing, Tianjin University, China
huangjt@tju.edu.cn

Pengfei Jiao
Center for Biosafety Research and Strategy, Tianjin University, China
College of Intelligence and Computing, Tianjin University, China
pjiao@tju.edu.cn

Liang Yang*
School of Artificial Intelligence, Hebei University of Technology, China
yangliang@vip.qq.com

Dongxiao He
College of Intelligence and Computing, Tianjin University, China
hedongxiao@tju.edu.cn

Françoise Fogelman-Souliè
College of Intelligence and Computing, Tianjin University, China
francoise.soulie@outlook.com

Yuxiao Huang
Data Science, Columbian College of Arts & Sciences, George Washington University, USA
yuxiaohuang@gwu.edu

ABSTRACT

Topic models have many important applications in fields such as Natural Language Processing. Topic embedding modelling aims at introducing word and topic embeddings into topic models to describe correlations between topics. Existing topic embedding methods use documents alone, which suffer from the topical fuzziness problem brought by the introduction of embeddings of semantic fuzzy words, e.g. polysemous words or some misleading academic terms. Links often exist between documents which form document networks. The use of links may alleviate this semantic fuzziness, but they are sparse and noisy which may meanwhile mislead topics. In this paper, we utilize community structure to solve these problems. It can not only alleviate the topical fuzziness of topic embeddings since communities are often believed to be topic related, but also can overcome the drawbacks brought by the sparsity and noise of networks (because community is a high-order network information). We give a new generative topic embedding model which incorporates documents (with topics) and network (with communities) together, and uses probability transition to describe the relationship between topics and communities to make it robust when topics and communities do not match. An efficient variational inference algorithm is then proposed to learn the model. We validate the superiority of our new approach on two tasks, document classifications and visualization of topic embeddings, respectively.

*corresponding author. Email: yangliang@vip.qq.com

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313623

CCS CONCEPTS

• **Information systems → Document topic models**.

KEYWORDS

Document networks; Topic embedding; Community structure

ACM Reference Format:
Di Jin, Jiantao Huang, Pengfei Jiao, Liang Yang, Dongxiao He, Françoise Fogelman-Souliè, and Yuxiao Huang. 2019. A Novel Generative Topic Embedding Model by Introducing Network Communities. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313623

1 INTRODUCTION

Topic models, such as Latent Dirichlet Allocation (LDA) [3] and Probability Latent Semantic Analysis (PLSA) [11], have a variety of applications in areas such as Natural Language Processing (NLP). A limitation of traditional topic models is that, they have not considered the correlation between topics. Some studies have made effort to characterize topic correlations. For example, correlated topic models [1, 2, 5] replace Dirichlet distribution with logistic normal distribution, and topic embedding models [19] represent topics in a low-dimension embedding space.

Compared to correlated topic models, topic embedding modelling is a straightforward and efficient alternative with lower computational cost [9, 14]. In most topic embedding models, word embedding (instead of the traditional one-hot encoding for words) is applied, to describe the semantic correlations between words. Topic embedding is used by replacing the original word distribution of topics, to represent topics in a continuous low-dimensional space. A link function is then defined to connect word embeddings to topic embeddings to describe the relationship between words and

topics. The main merit of topic embedding models is their capacity for capturing words co-occurrence at different semantic levels to derive correlations between topics while they still have problems. On the one hand, there are often semantic fuzzy words in documents, which have ambiguous semantic meanings. E.g., the polysemous word 'apple' can mean a kind of fruit or a mobile phone brand, and the academic terms 'parallel' and 'efficient' belonging to high-performance computing area may be also included in a paper of artificial intelligence. The existence of these words affects the accurate semantic representation of word embeddings. On the other hand, word embeddings also influence topic representations since they are connected by link functions. Thus, the semantic fuzziness of word embeddings (brought by the existence of semantic fuzzy words) may also lead to the semantic fuzziness of topics in topic embedding models.

At the same time, links are ubiquitous among documents [27, 28]. E.g., papers are linked via citations in DBLP and webpages are connected via hyperlinks in Wikipedia. The linked documents can be represented as a document network. Different from documents themselves that capture objective semantics, network structure denoted by connections between documents often provides subjective semantics which may complement document contents. Then, a pair of linked documents should have similar topic distributions [13, 26]. By doing so, networks may be able to help mitigate the semantic fuzziness of topics. That is to improve topic embeddings and topic distributions of documents, which often suffer from the semantic fuzziness due to word embeddings of semantic fuzzy words. However, we also find that using links directly may not perform well on improving topics since this low-order network information is often sparse and noisy [16, 22, 24]. That is, some nearby documents may not be connected due to the sparsity of networks, and meanwhile, several uncorrelated documents may be connected due to noisy links. In special cases, this may even destroy the quality of topics.

Fortunately, community structure, as a type of high-order network information, may be suitable to solve this problem. A community in a document network consists of documents which are connected more tightly than those in different communities. Also, it is often believed that documents in the same community are often topic-related [10, 12, 30]. So, rather than using links directly, one may have a good reason to use communities, to make documents in a same community own similar topic distributions. By doing so, the problem brought by sparsity and noise of networks may be nicely alleviated.

To be specific, we present a generative topic embedding model, namely Community-Enhanced Topic Embedding (CeTe), which incorporates documents and network structure together. CeTe consists of two main components connected by a probability transition mechanism. The first is the document component which incorporates word embeddings into topic modelling to describe correlations between topics and capture the local word co-occurrence. The second is the network component in which communities are described based on stochastic blockmodels [17]. It plays the role of alleviating the problem of semantic fuzziness of topics with the help of topic-related network communities. We further use probability transition to describe the intrinsic relationship between topics and communities. In this way, the model will work robustly even when the topics (from documents) and communities (from networks) are not well matched. We finally give an efficient variational inference algorithm to learn the model.

The contributions of this paper are as follows:

- Topic embedding models suffer from the semantic fuzziness of topics brought by the semantic fuzziness of word embeddings caused by the existence of semantic fuzzy words. The low-order link information is not suitable to be used directly for this problem due to the sparsity and noise of networks. To the best of our knowledge, this is the first time high-order network information (i.e., communities that are topic related) is used to solve this problem.
- We give a novel generative topic embedding model by using documents with topics and network structure with communities together. Their relationship is further described by using probability transition to make this new model robust even when topics and communities do not match well. We derive an efficient variational inference algorithm to learn this model.
- The superior performance of CeTe is tested on two tasks, i.e. document classification by comparing with seven state-of-the-art methods and a topic visualization application.

2 THE MODEL

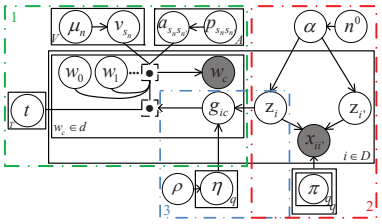


Figure 1: Graphical representation of CeTe. Part 1 in the green box (top left) denotes a document component describing topics. Part 2 in the red box (on the right) denotes a topological component describing network communities. Part 3 in the blue box (on the bottom) denotes the probabilistic transition mechanism connecting these two parts.

In this section, we give a formal description of the proposed model, i.e., Community Enhanced Topic Embedding (CeTe), with the purpose of improving document representation (topic distributions of documents) and topic embedding via introducing communities in networks. The graphical representation of this CeTe model is shown in Figure 1.

As shown in Figure 1, CeTe contains three components:

Document component with topics. This component depicts the generative process of each word given its topic and context words. In a set of documents $D = \{d_1, ..., d_N\}$ with K topics, each document d_i has L_i words, and the words in all documents form the vocabulary set $S = \{s_1, ..., s_W\}$. Each word w_{ij} in document d_i is assigned to a topic indexed by g_{ij}, where $\mathbf{G} = (g_{ij})_{N \times L_i}$ denotes the matrix of topics. Besides topics, the observed word is also affected by its context words. The context of a focus word w_c includes the c words before w_c in the document, denoted by $w_0 : w_{c-1}$. Both the topic and context words are represented in the

embedding vector form. That is, the embedding of word $w_{c'}$ and topic $g_{c'}$ are represented as $v_{c'}$ and $t_{g_{c'}}$, and we assume that their dimensions are the same. Then, matrices of all word embeddings and topic embeddings are $\mathbf{V} = (\mathbf{v}_{s_1},...,\mathbf{v}_{s_W})$ and $\mathbf{T} = (\mathbf{t}_1,...,\mathbf{t}_K)$. Thus the conditional distribution of a word given its context and topic can be factorized into two parts. The first is the distribution of this word and its context $P(w_c|w_0 : w_{c-1})$. It corresponds to the link function of a word embedding method such as PSDVec [20]. The distribution is:

$$P(w_c|w_0 : w_{c-1}) \approx P(w_c) \cdot exp\{\mathbf{v}_{w_c}^T \cdot \sum_{c'=0}^{c-1} \mathbf{v}_{w_{c'}} + \sum_{c'=0}^{c-1} a_{w_{c'}w_c}\}, \tag{1}$$

where $a_{w_{c'}w_c}$ is the bigram residual, which is non-linear and cannot be captured by $\mathbf{v}_{w_c}^T \mathbf{v}_{w_{c'}}$, and $P(w_c)$ is the probability of w_c appearing in the word corpus.

The second part is the distribution of the word and its topic. The topic can be also taken as a latent word, so that we can get the following distribution:

$$P(w_c|g_c) \approx P(w_c) \cdot exp\{\mathbf{v}_{w_c}^T \mathbf{t}_{g_c} + r_{g_c}\}, \tag{2}$$

where r_{g_c} is the logarithm of the normalized constant, which we call the topic residual. All topic residuals form a matrix $\mathbf{r}$=(r_1,...,r_K).

Since $\sum_{w_c \in S} P(w_c|k) = 1$ we can substitute it into Eq. (2) to approximate the r_k as:

$$r_k = -log\Big(\sum\nolimits_{s_j \in S} P(s_j) exp\{\mathbf{v}_{s_j}^T \mathbf{t}_k\}\Big). \tag{3}$$

Eq. (3) can be also represented in matrix form:

$$\mathbf{r} = -log\Big(\mathbf{u} exp\{\mathbf{V}^T \mathbf{T}\}\Big), \tag{4}$$

where $\mathbf{u}$ is the row vector of unigram probabilities.

Finally, according to Eqs. (1) and (2), we give the following link distribution for a document d_i:

$$\begin{aligned} &P(w_c|w_0 : w_{c-1}, g_c, d_i) \\ &\approx P(w_c) \cdot exp\{\mathbf{v}_{w_c}^T \cdot (\sum_{c'=0}^{c-1} \mathbf{v}_{w_{c'}} + \mathbf{t}_{g_c}) + \sum_{c'=0}^{c-1} a_{w_{c'}w_c} + r_{g_c}\}, \end{aligned} \tag{5}$$

where we simplify w_{ij} as w_c and $\mathbf{t}_{g_{ij}}$ as $\mathbf{t}_{g_c}$. Note that in order to avoid overfitting and alleviate the negative effect on topics, we constrain the magnitudes of all topic embeddings in a hyperball of radius λ.

Topological component with communities. This component describes the community structure that is often topic related to help model topics. Let a network have N documents which consist of Q communities. The adjacency matrix $\mathbf{X} = (x_{ii'})_{N \times N}$ is used to represent connections between documents. First, we generate $\boldsymbol{\alpha}$=(α_1,...,α_Q) by Dirichlet distribution, where α_q denotes the proportion of documents belonging to community q, and then get the following distribution:

$$P(\boldsymbol{\alpha}|\mathbf{n}^0) = [\Gamma(\sum\nolimits_{q=1}^{Q} n_q^0) / \prod\nolimits_{q=1}^{Q} \Gamma(n_q^0)] \prod\nolimits_{q=1}^{Q} \alpha^{n_q^0 - 1},$$

where $\mathbf{n}^0$=(n_1^0,...,n_Q^0) is the hyper-parameter and $\Gamma(\cdot)$ denotes the Gamma function.

Next, we use the Multinomial distribution to sample z_i in $\mathbf{Z} = (z_1, ..., z_N)$, where z_i is the community label of document d_i. Then, the distribution is defined as:

$$P(\mathbf{Z}|\boldsymbol{\alpha}) = \prod\nolimits_{i=1}^{N} P(z_i|\boldsymbol{\alpha}) = \prod\nolimits_{i=1}^{N} \alpha_{z_i}.$$

Finally, given the sampled community labels of documents d_i and d_j, i.e., z_i and $z_{i'}$, we sample $x_{ii'}$ from a Bernoulli distribution, defined as:

$$P(\mathbf{X}|\boldsymbol{\pi}, \mathbf{Z}) = \prod\nolimits_{i<i'} (deg_i deg_{i'} \pi_{z_i z_{i'}})^{x_{ii'}} (1 - deg_i deg_{i'} \pi_{z_i z_{i'}})^{1 - x_{ii'}}.$$

The distribution above is based on the degree-corrected stochastic block model [17]. Here, $\boldsymbol{\pi} = (\pi_{ql})_{Q \times Q}$ is the block matrix where π_{ql} denotes the connection probability between nodes from communities q and l, and deg_i is the degree of document d_i.

Probability transition connecting topics and communities. This component describes the transition process from communities to topics to make the model work even when communities and topics do not match well. First, the probability transition matrix $\mathbf{H} = (\eta_{qk})_{Q \times K}$ is generated by a Dirichlet distribution, where η_{qk} represents the probability that document d_i is in the k-th semantic topic given it belongs to the q-th community. This distribution is defined as:

$$P(\mathbf{H}|\boldsymbol{\rho}) = \prod\nolimits_{q=1}^{Q} [\Gamma(\sum\nolimits_{k=1}^{K} \rho_k) / \prod\nolimits_{k=1}^{K} \Gamma(\rho_k)] \prod\nolimits_{k=1}^{K} \eta_{ik}^{\rho_k - 1},$$

where $\boldsymbol{\rho}$ is the hyper-parameter.

Next, given the community label z_i of document d_i, we use a Multinomial distribution to sample g_{ij}, defined as:

$$P(\mathbf{G}|\mathbf{H}, \mathbf{Z}) = \prod\nolimits_{i=1}^{N} \prod\nolimits_{j=1}^{L_i} \eta_{z_i g_{ij}}.$$

After finishing the definition of these three components, a complete generative process of this model is shown in the following. Note that, word embedding $\mathbf{v}_{s_n}$ is drawn from a Gaussian distribution as well as bigram residuals $a_{s_n s_{n'}}$. But for clarity, here we only focus on modelling topic embeddings while ignoring the generative process of word embeddings.

1. Choose $\boldsymbol{\alpha} \sim Dir(\boldsymbol{n^0})$
2. For each community q: Choose $\boldsymbol{\eta_q} \sim Dir(\boldsymbol{\rho})$
3. For the k-th topic: Choose $\mathbf{t}_k \sim Unif(B_\lambda)$
4. For each document d_i:
   (a) Draw community assignment $z_i \sim Mult(\boldsymbol{\alpha})$
   (b) For each document $d_{i'}$ with $i' > i$:
   Draw link $x_{ii'} \sim Bernoulli(deg_i deg_{i'} \pi_{z_i z_{i'}})$
   (c) For the j-th word:
   (*i*) Draw topic assignment $g_{ij} \sim Mult(\boldsymbol{\eta}_{z_i})$
   (*ii*) Draw word w_{ij} from S according to $P(w_{ij}|w_{i,j-c}, w_{i,j-1}, g_{ij}, d_i)$

where Dir, $Unif$ and $Mult$ are the Dirichlet, Uniform and Multinomial distributions respectively.

Then, the complete data likelihood is:

$$\begin{aligned} &P(D, \mathbf{X}, \mathbf{A}, \mathbf{V}, \mathbf{T}, \mathbf{G}, \mathbf{Z}, \mathbf{H}, \boldsymbol{\alpha}|\boldsymbol{\pi}, \mathbf{n}^0, \boldsymbol{\rho}, \lambda, \boldsymbol{\mu}) \\ &= \prod_{n=1}^{W} P(\mathbf{v}_{s_n}; \mu_n) \prod_{n,n'=1}^{W,W} P\Big(a_{s_n s_{n'}}; f(p_{s_n s_{n'}})\Big) \prod_{k}^{K} Unif(B_\lambda) \\ &\cdot \prod_{i=1}^{N} \prod_{j=1}^{L_i} \Big(Mult(\boldsymbol{\eta}_{z_i}) \cdot P(d_i|\mathbf{V}, \mathbf{A}, \mathbf{T}, \mathbf{G})\Big) \prod_{i=1}^{N} Mult(z_i|\boldsymbol{\alpha}) \\ &\cdot \prod_{q=1}^{Q} Dir(\boldsymbol{\eta}_q|\boldsymbol{\rho}) \cdot Dir(\boldsymbol{\alpha}|\mathbf{n}^0) \prod_{i<i'} P(x_{ii'}|z_i, z_{i'}, deg_i deg_{i'} \pi_{z_i z_{i'}}). \end{aligned}$$

In the steps above, $P(\mathbf{v}_{s_n}; \mu_n)$ and $P(a_{s_n s_{n'}}; f(p_{s_n s_{n'}}))$ are both defined as Gaussian priors, and $P(d_i|\mathbf{V}, \mathbf{A}, \mathbf{T}, \mathbf{G})$ is a simple expression of $P(w_{ij}|w_{i,j-c}, w_{i,j-1}, g_{ij}, d_i)$.

3 MODEL OPTIMIZATION

In this section, we aim at learning the embeddings of words and topics $\{\mathbf{V}, \mathbf{T}\}$, model parameters $\{\mathbf{H}, \boldsymbol{\alpha}\}$, and latent variables $\{\mathbf{G}, \mathbf{Z}\}$. Due to the coupling between word-related variables $\{\mathbf{V}, \mathbf{A}\}$ and topic-related variables $\{\mathbf{T}, \mathbf{G}, \mathbf{Z}, \mathbf{H}\}$, their simultaneous optimization is too time consuming in general. So as was done by other topic embedding methods [19], we also use a word embedding method, i.e. PSDVec [20], to learn $\{\mathbf{V}, \mathbf{A}\}$ first, and then optimize $\{\mathbf{T}, \mathbf{G}, \mathbf{Z}, \mathbf{H}, \boldsymbol{\alpha}\}$ with the fixed $\{\mathbf{V}^*, \mathbf{A}^*\}$. But this is also a non-trivial task. For this optimization problem, we propose a variational expectation-maximization (EM) algorithm that includes variational inference and parameter maximization, respectively.

3.1 Variational Inference

In this step, we first fix $\mathbf{T}$ as the constant $\mathbf{T}^*$ and then maximize the posterior $P(\mathbf{G}, \mathbf{H}, \mathbf{Z}, \boldsymbol{\alpha}|D, \mathbf{X}, \boldsymbol{\pi}, \mathbf{T}^*, \mathbf{A}^*, \mathbf{V}^*)$ via variational inference. First, we define the following variational distributions:

$$q(\boldsymbol{\alpha}, \mathbf{H}, \mathbf{Z}, \mathbf{G}) = q(\boldsymbol{\alpha})q(\mathbf{H})q(\mathbf{Z})q(\mathbf{G}),$$

where the variables are mutually independent and their distributions are specified as follows:

$$q(\boldsymbol{\alpha}) = \prod_{q=1}^{Q} Dir(\alpha_q|n_q), \quad q(\mathbf{H}) = \prod_{q=1}^{Q}\prod_{k=1}^{K} Dir(\eta_{qk}|\gamma_{qk}),$$
$$q(\mathbf{Z}) = \prod_{i=1}^{N}\prod_{q=1}^{Q} Mult(z_{iq}|\tau_{iq}), \quad q(\mathbf{G}) = \prod_{i=1}^{N}\prod_{j=1}^{L_i} Mult(g_{ij}|\boldsymbol{\beta}_{ij}).$$

Variational inference here is to minimize the KL divergence between the true posterior distribution $P(\mathbf{G}, \mathbf{H}, \mathbf{Z}, \boldsymbol{\alpha}|D, \mathbf{X}, \boldsymbol{\pi}, \mathbf{T}^*, \mathbf{A}^*, \mathbf{V}^*)$ and $q(\boldsymbol{\alpha}, \mathbf{H}, \mathbf{Z}, \mathbf{G})$, so that the posterior of $\{\boldsymbol{\alpha}, \mathbf{G}, \mathbf{Z}, \mathbf{H}\}$ can be nicely approximated. This is equivalent to maximizing the evidence lower bound (ELBO) $L(q, \mathbf{T})$:

$$L(q, \mathbf{T}) \quad = E_q log\left(\frac{P(D, \mathbf{X}, \mathbf{A}, \mathbf{V}, \mathbf{G}, \mathbf{Z}, \mathbf{H}, \boldsymbol{\alpha}|\mathbf{T}, \boldsymbol{\pi}, \mathbf{n^0}, \boldsymbol{\rho}, \lambda, \boldsymbol{\mu})}{\mathrm{q}(\boldsymbol{\alpha}, \mathbf{H}, \mathbf{Z}, \mathbf{G})}\right).$$

Next, in order to maximize the ELBO $L(q, \mathbf{T})$, we calculate its partial derivative with respect to $n_q, \tau_{iq}, \gamma_{qk}$ and $\beta_{ij,k}$, respectively, and set these derivatives to 0. That is,

$$\nabla L(q, \mathbf{T}) = (\frac{\partial L}{\partial n}, \frac{\partial L}{\partial \tau}, \frac{\partial L}{\partial \gamma}, \frac{\partial L}{\partial \beta}) = 0.$$

After this inference, we can collect the optimal terms:

$$\mathrm{n}_q = n_q^0 + \sum_{i=1}^{N} \tau_{iq}, \tag{6}$$

$$\tau_{iq} \propto exp\Big\{\psi(n_q) - \psi(\sum_{q=1}^{Q} n_q) + \sum_{j=1}^{L_i}\sum_{k=1}^{K} \beta_{ij,k}\Big(\psi(\gamma_{qk}) - \psi(\sum_{k=1}^{K}\gamma_{qk})\Big) + \sum_{i'\neq i}^{N}\sum_{l=1}^{Q}\tau_{i'l}\Big(x_{ii'}log(deg_i deg_{i'}\pi_{ql}) + (1-x_{ii'})log(1-deg_i deg_{i'}\pi_{ql})\Big)\Big\}, \tag{7}$$

$$\beta_{ij,k} \propto exp\Big\{\sum_{q=1}^{Q}\tau_{iq}\Big(\psi(\gamma_{qk}) - \psi(\sum_{k=1}^{K}\gamma_{qk})\Big) + \mathbf{v}_{w_{ij}}^T\mathbf{t}_k + r_{ik}\Big\}, \tag{8}$$

$$\gamma_{qk} = \rho_k + \sum_{i=1}^{N}\sum_{j=1}^{L_i}\tau_{iq}\beta_{ij,k}, \tag{9}$$

where $\psi(\cdot)$ denotes the Digamma function.

3.2 Parameter Maximization

In this step, $\mathbf{n}, \boldsymbol{\tau}, \boldsymbol{\gamma}, \boldsymbol{\beta}$ are fixed, and our purpose is to optimize $\mathbf{T}$. First, we take the derivatives of $L(q, \mathbf{T})$ with respect to $\mathbf{T}$ as:

$$\frac{\partial L}{\partial \mathbf{T}} = \sum_{i=1}^{N}\sum_{j=1}^{L_i}(\mathbf{v}_{w_{ij}}\boldsymbol{\beta}_{ij}^T + \sum_{k=1}^{K}\beta_{ij,k}\frac{\partial r_k}{\partial \mathbf{T}}). \tag{10}$$

In the following, we set $\sum_{j=1}^{L_i}\beta_{ij,k} = \overline{m}_{ik}$, which is the sum of variational probabilities of every word in the document d_i being assigned into the k-th topic. Then we use the Gradient Descent method [19]. The k'-th column of the gradient matrix $\partial r_k/\partial \mathbf{T}$ is $\partial r_k/\partial \mathbf{t}_{k'}$. According to Eq. (3), when $k' = k$, we can derive the following equation:

$$\frac{\partial r_k}{\partial \mathbf{t}_k} = -exp\{r_k\}\cdot exp\{\mathbf{t}_k^T\mathbf{V}\}(\mathbf{u}\circ\mathbf{V}),$$

where $\mathbf{u}\circ\mathbf{V}$ denotes the element of each column of $\mathbf{V}$ multiplied by the corresponding element of $\mathbf{u}$. When $k' \neq k$, $\partial r_k/\partial \mathbf{t}_{k'} = \mathbf{0}$ can also be easily derived.

Based on the description above we have $\partial r_k/\partial \mathbf{T} = (\mathbf{0}, \ldots \partial r_k/\partial \mathbf{t}_k, \ldots, \mathbf{0})$. By substituting it into Eq. (10), we can obtain:

$$\frac{\partial L}{\partial \mathbf{T}} = \sum_{i=1}^{N}\Big(\sum_{j=1}^{L_i}\mathbf{v}_{w_{ij}}\boldsymbol{\beta}_{ij}^T + (\overline{m}_{i1}\frac{\partial r_1}{\partial \mathbf{t}_1}, \ldots, \overline{m}_{iK}\frac{\partial r_K}{\partial \mathbf{t}_K})\Big).$$

At this time, $\mathbf{T}$ can be optimized via a gradient descent process:

$$\mathbf{T}^m = \mathbf{T}^{m-1} + \Big(\sum_{i=1}^{N}\lambda(m, L_i)\Big)(\frac{\partial L}{\partial \mathbf{T}}), \tag{11}$$

where m is the ongoing iteration step during the update process, $\lambda(m, L_i) = \frac{L_0\lambda_0}{m\cdot\max\{L_i, L_0\}}$ is a function of the learning rate, λ_0 is the initial learning rate, L_i is the length of the i-th document, and L_0 is the threshold of document length. Besides, since the constraint $||\mathbf{t}_{ik}^{(m)}|| \leq \lambda$, we normalize it by $\lambda/||\mathbf{t}_k^{(m)}||$ when $||\mathbf{t}_k^{(m)}|| > \lambda$.

Also of note, after updating $\mathbf{T}_i$, we can further update the topic residuals $\mathbf{r}$ according to Eq. (4) to refine this process.

3.3 Algorithm and Complexity Analysis

The process of CeTe is shown in Algorithm 1, where we omit the word embedding process for clarity. In the step of variational inference, the time complexity for updating n_q, τ_{iq}, γ_{qk} and $\beta_{ij,k}$ is $O(N)$, $O(K + NQ)$, $O(N\bar{L})$ and $O(QK)$, respectively, where N,

K and Q are the number of documents, topics and communities respectively in the document network and $\bar{L}$ is the average number of words in all documents. Taking advantage of the sparsity of document networks, the above time complexities can be further reduced to $O(NQ)$, $O(K+NQ)$, $O(N\bar{L})$, $O(NQK+MQ^2)$, $O(QKN\bar{L})$ and $O(N\bar{L}QK^2)$, respectively, where M is the number of links in the network. In addition, the time complexity for the step of parameter maximization is $O(N(K+\bar{L}))$. Thus, the overall time complexity of this algorithm is $O(N\bar{L}QK^2+MQ^2)$, which is near linear on the scale of networks (N or M).

Algorithm 1 The Process of CeTe

Input: $\mathbf{X}$, D, Q, K, a threshold ε, $count_{max}$
Output: $\mathbf{T}, \mathbf{H}, \boldsymbol{\alpha}, \mathbf{G}, \mathbf{Z}$
1. Initialize $\boldsymbol{\pi}$, $\mathbf{T}$, $\mathbf{r}$ and variational parameters randomly
2. count=1
3. **repeat:**
   (a) Update $n_q, \tau_{iq}, \beta_{ij,k}, \gamma_{qk}$ via (6)-(9)
   (b) Update $\mathbf{T}$ via (11) and $\mathbf{r}$ via (4)
   (c) Calculate ELBO L^{count} and count=count+1
   Until $L^{count} - L^{count-1} < \varepsilon$ or $count > count_{max}$

4 EXPERIMENTS

We first introduce the experiment setup. We then use document classification, the gold method to validate topic embedding models, to evaluate the quality of document representations (the topic distribution of documents) derived by our new approach CeTe. Next, we demonstrate the visualization of topic embeddings to further validate its superiority. We finally use a case study to show why CeTe works.

4.1 Experiment Setup

4.1.1 Datasets. We conducted the experiments on three public datasets, including one DBLP dataset [25] and two hep-th datasets[1]. DBLP includes a collection of papers in computer field. In DBLP, the title and abstract are extracted as texts for each paper and the citation relationships are used to form links between papers. We extract its largest connected component as done by other related works [6, 17, 29]. The ground truth of the papers is fixed according to the CCF (China Computer Federation) classification. The hep-th includes a large corpus of physics-related papers. Its data processing method is similar to that used by DBLP, except that here we use Journals to form the ground truth. In addition, due to the high imbalance of document quantities in different categories for both DBLP and hep-th, we choose the five largest categories to form the DBLP dataset; and for hep-th, we use the four largest categories to form the first subdataset called large-hep, and the three smaller categories to form the other subdataset called small-hep. The datasets details are shown in Table 1.

4.1.2 Baselines. We test our approach against seven state-of-the-art methods, which can be divided into five categories. The first includes three topic models, i.e. Latent Dirichlet Allocation (LDA) [3], Laten Feature Topic Modelling (LFTM) [23] and GibbsLDA [7].

[1]https://www.cs.cornell.edu/projects/kddcup/datasets.html

Table 1: Datasets statistics.

Datasets	Documents	Edges	Words	Categories
DBLP	6,936	12,353	506,269	5
small-hep	397	812	18,718	3
large-hep	11,752	134,956	622,642	4

The second is a topic model that also incorporates network links, named Relational Topic Model (RTM) [4]. The third is a topic embedding method called TopicVec [19]. The fourth is a community detection method named Degree-Corrected Stochastic Block Model (DCSBM) [17], in which we use the community memberships of documents as their representations. The fifth is a document representation method called Doc2Vec [18].

4.1.3 Parameter Setting. As done by TopicVec [19], word embeddings are pre-trained by PSDVec [20] on the latest Wikipedia snapshot[2]. The dimensions of topic embeddings and word embeddings are both set to 500. For datasets DBLP, large-hep and small-hep, we set the number of topics to 500, 200 and 30, respectively. We set the hyper-parameters $n^0 = \rho = 1$ and $\lambda = 7$, which are similar with those used in baselines.

4.2 Test on Document Classification

On this test, according to [19], we first randomly select 80% of documents as the training set and 20% as the test set for each dataset. We then use ℓ-1 regularized SVM as classifier. We use the scikit-learn library[3] to train an ℓ-1 regularized linear SVM one-vs-all classifier for fairness. We adopt macro-averaged Precision, Recall and F1-Score as the evaluation indices. The document representations derived from each method compared are used as input instances to the classifier.

Based on the results shown in Table 2, our new approach CeTe performs the best on all datasets. Neither the topic models (LDA, GibbsLDA and LFTM) using documents alone, nor the community detection method (DCSBM) only using network topology performs well. While RTM (a topic model incorporating links) performs better in most cases, it is still not competitive with our approach CeTe. The reason should be that we introduce the words co-occurrence information at different levels and correlations between topics by using topic embedding to better model topic information. In addition, compared to the state-of-the-art topic embedding method TopicVec, our CeTe still has an obvious improvement. This should be because, by introducing the link information (especially the high-order community structure), the problem of semantic fuzziness of topics suffered by topic embedding methods has been alleviated by our approach. These all validate the effectiveness of this new approach.

4.3 Test by Visualization of Topic Embeddings

We illustrate the visualization of topic embeddings derived by our CeTe method to further validate its performance. The results on DBLP with 500 topics are shown in Figure 2. We find that some closely connected articles correspond to each topic. For example, in the upper-left set, the words 'wireless', 'communication', 'network' and 'sensor' are all related to a field of computer network.

[2]https://dumps.wikimedia.org/enwiki/
[3]http://scikit-learn.org/stable/modules/svm.html

Table 2: Performance of different methods on document classification. Bold indicates the best score.

	DBLP			small-hep			large-hep		
	Precison	Recall	F1-Score	Precison	Recall	F1-Score	Precison	Recall	F1-Score
LDA	0.700	0.457	0.404	0.287	0.287	0.281	0.414	0.411	0.401
GibbsLDA	0.786	0.722	0.715	0.399	0.400	0.396	0.346	0.405	0.346
Doc2Vec	0.523	0.496	0.448	0.454	0.438	0.431	0.385	0.394	0.350
LFTM	0.739	0.653	0.634	0.619	0.575	0.569	0.419	0.418	0.405
RTM	0.560	0.563	0.549	0.565	0.550	0.553	0.407	0.411	0.393
DCSBM	0.461	0.430	0.393	0.451	0.487	0.439	0.359	0.358	0.325
TopicVec	0.729	0.689	0.681	0.624	0.600	0.599	0.405	0.417	0.393
CeTe	**0.790**	**0.736**	**0.731**	**0.671**	**0.650**	**0.654**	**0.437**	**0.433**	**0.417**

It partly shows that the topic embeddings derived by our method have a high quality. We also noticed that, there is a word 'software' in the lower-middle set, which should originally belong to the 'software engineering' field, but actually belongs to the area of 'high-performance computing' here. That is, the keywords such as 'distributed', 'power' and 'energy' in this topic are all from 'high-performance computing', while there is another keyword 'software' which seems to be a mistake. But after a deep investigation we find that, the 'high-performance computing' topic also contains some papers using the idea of software engineering. At the same time, these papers are linked with other papers in 'high-performance computing' and are classified in the same community and thus corrected by our method. This just corresponds to the real situation where these papers (with semantic fuzzy words) belong to the field of 'high-performance computing'. This further validates that the problem of semantic fuzziness of topic embeddings brought by the existence of semantic fuzzy words has been addressed by using links with communities. This visualization can further help discover the meaningful semantic structures to understand topics in real applications.

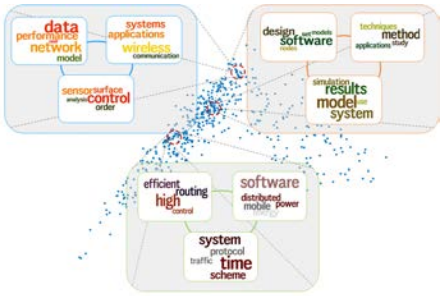


Figure 2: An example of visualizing the 500 topic embeddings derived by CeTe on the DBLP dataset. Each point in the cloud represents a latent topic. The distance between points indicates the correlation between topics, and smaller distance means stronger correlation. We mainly show three sets of points (topics) with small distance in the embedding space. Each topic is characterized by keywords according to the word distribution of this topic (by calculating the cosine similarity between topic embedding and word embeddings). The word sizes are proportional to the probability that this word belongs to the topic. Edges represent the strength of the correlations between topics.

4.4 A case study

Here is to illustrate how our new model CeTe alleviates the topical fuzziness problem of topic embedding methods by introducing the high-order community structure. We also test it on the DBLP dataset with three categories (i.e. theoretical computer science, high-performance computing and computer network), and compare it with TopicVec which is one of the state-of-the-art topic embedding models [19]. We randomly select 10% of the documents as test set, and use the topic distributions of documents derived from CeTe (and TopicVec) as document representations. There are 78 of 375 documents that are classified into wrong categories by TopicVec, while our method corrects about half of them. The results are shown in Figure 3. Due to space limit, we only discuss three of the papers in details since the rests are similar.

The first sample paper is 'Microcode compaction with timing constraints' which is clearly in the field of 'high-performance computing'. However, some words such as 'timing constraints', 'optimization' and 'heuristic' often appear in its maintext. The embedding of these fuzzy topical words is the reason why this paper is assigned to the 'theoretical computer science' category by TopicVec. However, this paper and other seven papers in 'high-performance computing' are divided into the same community by our CeTe, and thus have similar topic distributions. In other words, the topical fuzziness of this paper has been corrected.

The second is the paper 'MILP approach to pattern generation in logical analysis of data' which belongs to the 'theoretical computer science' category. But there are also some high-frequency fuzzy topical words such as 'linear', 'efficient', 'optimal' and 'hundreds of' in this paper which lead to that it is misplaced by TopicVec. However, through links, this paper and three other papers in 'theoretical computer science' are classified into the same community, and hence its topic is corrected again.

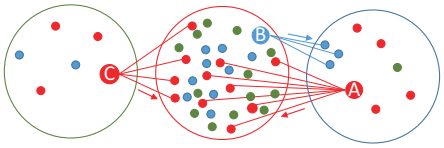


Figure 3: An example that uses network communities to correct inaccurate document representations suffered from the semantic fuzziness of topics. The blue, red and green circles are used to represent the three categories, i.e. theoretical computer science, high-performance computing and computer network, respectively. The blue (red or green) nodes denotes that it belongs to the blue (red or green) category. The nodes with connections in this figure denotes that they are divided into the same community by the topological component of our model. Here only some typical examples are shown in each category. Most of them are papers indeed influenced by semantic fuzziness of word embeddings, which are wrongly classified by TopicVec while corrected by our CeTe. Especially, the titles of papers A, B and C are 'Microcode compaction with timing constraints', 'MILP approach to pattern generation in logical analysis of data' and 'Fast Path-Based Neural Branch Prediction', respectively.

The third paper is 'Fast Path-Based Neural Branch Prediction' which belongs to 'high-performance computing'. But as shown in Figure 3, it is incorrectly assigned to 'computer network' by TopicVec. After a deeper investigation of its maintext, we found that

there are often some keywords such as 'latency', 'path', 'branch' and 'instructions' frequently appear in this paper which may mislead its topic distribution. But by utilizing links between papers, this paper and other three papers in 'high-performance computing' are assigned to the same community, which helps to correct its topic distributions by our CeTe.

To sum up, this case study further validates that, with the help of network community, the inaccurate topic distributions of documents brought by embedding semantic fuzzy words can be well alleviated by our CeTe approach.

5 CONCLUSION AND DISCUSSIONS

We proposed a novel generative topic embedding model CeTe using documents with topics and network structure with communities together, and developed an efficient variational expectation-maximization algorithm to learn the model. The new approach utilizes the high-order network information, i.e., community structure, to solve the problem of topical fuzziness suffered by topic embedding models which is from introducting the embeddings of semantic fuzzy words. The experiment results on two tasks (i.e. document classification and topic visualization) validate its superiority compared with the state-of-the-art methods.

There have been some topic embedding methods presented recently [14, 19, 21]. However, it is the first time, to the best of our knowledge, to use network topology, specifically the community structure, to improve their performance. In addition, there are also some topic models [4, 8, 15, 31] incorporating network information. However, they have essential differences. First, the introduction of links in topic models is to capture the additional correlations between documents, while our purpose is to use links to alleviate the topical fuzziness problem suffered by topic embedding methods (due to the introduction of word embeddings of the semantic fuzzy words). Second, rather than using the links directly, we use the high-order network information, i.e., community structure, to overcome the shortcomings brought by the noise and sparseness of networks. Third, we further use probability transition to describe the relationship between topics and communities, in order to make the method robust even when the topics and communities do not match very well.

ACKNOWLEDGEMENTS

This work was partly supported by the National Key R&D Program of China (No.2017YFC0820106) and the National Natural Science Foundation of China (No.61772361, 61876128, 61503281, 61502334).

REFERENCES

[1] David Blei and John Lafferty. 2006. Correlated topic models. *Advances in neural information processing systems* 18 (2006), 147.
[2] David M Blei and John D Lafferty. 2007. A correlated topic model of science. *The Annals of Applied Statistics* 1, 1 (2007), 17–35.
[3] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.
[4] Jonathan Chang and David Blei. 2009. Relational topic models for document networks. In *Artificial Intelligence and Statistics*. 81–88.
[5] Jianfei Chen, Jun Zhu, Zi Wang, Xun Zheng, and Bo Zhang. 2013. Scalable inference for logistic-normal topic models. In *Advances in Neural Information Processing Systems*. 2445–2453.
[6] Pin-Yu Chen and Alfred O Hero. 2015. Deep community detection. *IEEE Transactions on Signal Processing* 63, 21 (2015), 5706–5719.
[7] Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences* 101, suppl 1 (2004), 5228–5235.
[8] Dongxiao He, Zhiyong Feng, Di Jin, Xiaobao Wang, and Weixiong Zhang. 2017. Joint identification of network communities and semantics via integrative modeling of network topologies and node contents. In *Thirty-First AAAI Conference on Artificial Intelligence*. AAAI Press, 116–124.
[9] Junxian He, Zhiting Hu, Taylor Berg-Kirkpatrick, Ying Huang, and Eric P Xing. 2017. Efficient correlated topic modeling with topic embedding. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 225–233.
[10] Yuan He, Cheng Wang, and Changjun Jiang. 2018. Discovering canonical correlations between topical and topological information in document networks. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 30, 3 (2018), 460–473.
[11] Thomas Hofmann. 1999. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 289–296.
[12] Xin Huang and Laks VS Lakshmanan. 2017. Attribute-driven community search. *Proceedings of the VLDB Endowment* 10, 9 (2017), 949–960.
[13] Seungil Huh and Stephen E Fienberg. 2012. Discriminative topic modeling based on manifold learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 5, 4 (2012), 20.
[14] Di Jiang, Lei Shi, Rongzhong Lian, and Hua Wu. 2016. Latent topic embedding. In *Proceedings of the 26th International Conference on Computational Linguistics*. 2689–2698.
[15] Di Jin, Xiaobao Wang, Ruifang He, Dongxiao He, Jianwu Dang, and Weixiong Zhang. 2018. Robust detection of link communities in large social networks by exploiting link semantics. In *Thirty-Second AAAI Conference on Artificial Intelligence*. AAAI Press, 314–321.
[16] Indika Kahanda and Jennifer Neville. 2009. Using Transactional Information to Predict Link Strength in Online Social Networks. *The International AAAI Conference on Web and Social Media* 9 (2009), 74–81.
[17] Brian Karrer and Mark EJ Newman. 2011. Stochastic blockmodels and community structure in networks. *Physical Review E* 83, 1 (2011), 016107.
[18] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*. 1188–1196.
[19] Shaohua Li, Tat-Seng Chua, Jun Zhu, and Chunyan Miao. 2016. Generative topic embedding: a continuous representation of documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 666–675.
[20] Shaohua Li, Jun Zhu, and Chunyan Miao. 2015. A Generative Word Embedding Model and its Low Rank Positive Semidefinite Solution. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1599–1609.
[21] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical Word Embeddings.. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2418–2424.
[22] Sharad Nandanwar and M Narasimha Murty. 2016. Structural neighborhood based classification of nodes in a network. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1085–1094.
[23] Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. 2015. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics* 3 (2015), 299–313.
[24] Guo-Jun Qi, Charu Aggarwal, Qi Tian, Heng Ji, and Thomas Huang. 2012. Exploring context and content links in social media: A latent space method. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 5 (2012), 850–862.
[25] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. ArnetMiner: Extraction and Mining of Academic Social Networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 990–998.
[26] Mirwaes Wahabzada, Zhao Xu, and Kristian Kersting. 2010. Topic models conditioned on relations. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 402–417.
[27] Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. 2007. Manifold-Ranking Based Topic-Focused Multi-Document Summarization.. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, Vol. 7. 2903–2908.
[28] Suhang Wang, Jiliang Tang, Charu Aggarwal, and Huan Liu. 2016. Linked document embedding for classification. In *Proceedings of the 25th ACM international on conference on information and knowledge management*. ACM, 115–124.
[29] Joyce Jiyoung Whang, David F Gleich, and Inderjit S Dhillon. 2016. Overlapping community detection using neighborhood-inflated seed expansion. *IEEE Transactions on Knowledge and Data Engineering* 28, 5 (2016), 1272–1284.
[30] Wenhui Wu, Sam Kwong, Yu Zhou, Yuheng Jia, and Wei Gao. 2018. Nonnegative matrix factorization with mixed hypergraph regularization for community detection. *Information Sciences* 435 (2018), 263–281.
[31] Ge Zhang, Di Jin, Jian Gao, Pengfei Jiao, Francoise Fogelman-Soulié, and Xin Huang. 2018. Finding communities with hierarchical semantics by distinguishing general and specialized topics. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. AAAI Press, 3648–3654.

A Scalable Hybrid Research Paper Recommender System for Microsoft Academic

Anshul Kanakia
Microsoft Research
Redmond, Washington
ankana@microsoft.com

Zhihong Shen
Microsoft Research
Redmond, Washington
zhihosh@microsoft.com

Darrin Eide
Microsoft Research
Redmond, Washington
darrine@microsoft.com

Kuansan Wang
Microsoft Research
Redmond, Washington
kuansanw@microsoft.com

ABSTRACT

We present the design and methodology for the large scale hybrid paper recommender system used by Microsoft Academic. The system provides recommendations for approximately 160 million English research papers and patents. Our approach handles incomplete citation information while also alleviating the cold-start problem that often affects other recommender systems. We use the Microsoft Academic Graph (MAG), titles, and available abstracts of research papers to build a recommendation list for all documents, thereby combining co-citation and content based approaches. Tuning system parameters also allows for blending and prioritization of each approach which, in turn, allows us to balance paper novelty versus authority in recommendation results. We evaluate the generated recommendations via a user study of 40 participants, with over 2400 recommendation pairs graded and discuss the quality of the results using P@10 and nDCG scores. We see that there is a strong correlation between participant scores and the similarity rankings produced by our system but that additional focus needs to be put towards improving recommender precision, particularly for content based recommendations. The results of the user survey and associated analysis scripts are made available via GitHub and the recommendations produced by our system are available as part of the MAG on Azure to facilitate further research and light up novel research paper recommendation applications.

KEYWORDS

recommender system, word embedding, big data, k-means, clustering, document collection

ACM Reference Format:
Anshul Kanakia, Zhihong Shen, Darrin Eide, and Kuansan Wang. 2019. A Scalable Hybrid Research Paper Recommender System for Microsoft Academic. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, Article 4, 7 pages. https://doi.org/10.1145/3308558.3313700

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313700

1 INTRODUCTION AND RELATED WORK

Microsoft Academic[1] (MA) is a semantic search engine for academic entities [26]. The top level entities of the Microsoft Academic Graph (MAG) include papers and patents, fields of study, authors, affiliations (institutions and organizations), and venues (conferences and journals), as seen in Fig. 1. As of October, 2018, there are over 200 million total documents in the MA corpus, of which approximately 160 million are English papers or patents. These figures are growing rapidly [11].

The focus of this article is to present the recommender system and paper similarity computation platform for English research papers developed for the MAG. We henceforth define the term 'paper' to mean English papers and patents in the MAG, unless otherwise noted. Research paper recommendation is a relatively nascent field [3] in the broader recommender system domain but its value to the research community cannot be overstated.

The current approach to knowledge discovery is largely manual — either following the citation graph of known papers or through human curated approaches. Additionally, live feeds such as publisher RSS feeds or manually defined news triggers are used to gain exposure to new research content. This often leads to incomplete literature exploration, particularly by novice researchers since they are almost completely reliant on what they see online or their advisors and peer networks for finding new relevant papers. Following the citation network of known papers to discover content often leads to information overload, resulting in dead ends and hours of wasted effort. The current manual approaches are not scalable, with tens of thousands of papers being published everyday and the number of papers published globally increasing exponentially [11, 12]. To help alleviate this problem, a number of academic search engines have started adding recommender systems in recent years [1, 7, 18, 19, 28]. Still other research groups and independent companies are actively producing tools to assist with research paper recommendations for both knowledge discovery and citation assistance [2, 5, 13, 17]; both these use cases being more-or-less analogous.

Existing paper recommender systems suffer from a number of limitations. Besides Google Scholar, Microsoft Academic, Semantic Scholar, Web of Science, and a handful of other players, the vast majority of paper search engines are restricted to particular research domains such as the PubMed database for Medicine &

[1] https://preview.academic.microsoft.com



Figure 1: The Microsoft Academic Graph (MAG) website preview and statistics as of October, 2018.

Biology, and IEEE Xplore for Engineering disciplines. As such, it is impossible for recommender systems on these field-specific search sites to suggest cross-domain recommendations. Also, a number of proposed user recommender systems employ collaborative filtering for generating user recommendations [9]. These systems suffer from the well known cold-start problem[2]. There is little to no readily available data on metrics such as user attachment rates for research paper search and recommendation sites and so it becomes difficult to evaluate the efficacy of collaborative filtering techniques without a solid active user base. Moreover, with the introduction of privacy legislation such as the General Data Protection Regulation act (GDPR) in Europe, it is becoming increasing difficult and costly to rely on user data — which is why the Microsoft Academic website does not store personal browsing information — making collaborative filtering all the more difficult. Finally, besides Google Scholar, the self attested paper counts of other research paper databases are in the tens of millions while the estimated number of papers from our system (as well as Google Scholar estimates) put the total number of published papers easily in the hundreds of millions. Having a tenth of the available research corpus can heavily dilute recommendations, providing incomplete and unsatisfactory results. The MA paper recommender platform aims to alleviate some of the aforementioned shortcomings by,

(1) employing the entire MAG citation network and interdisciplinary corpus of over 200 million papers,
(2) using a combination of co-citation based and content embedding based approaches to maximize recommendation coverage over the entire corpus while circumventing the cold-start problem,
(3) and providing the computed paper recommendations to the broader research and engineering community so they can be analyzed and improved by other research groups.

We present two possible interaction modes with the MA paper recommender platform. The first mode is via the "Related Papers" section on the MA search engine paper details page. Users can browse the MA search site using the novel semantic interpretation engine powering the search experience to view their desired paper [26]. The paper details page, as seen in Fig.2, contains a tab for browsing related papers (with dynamic filters) that is populated using the techniques mentioned here. The second mode of interaction is via Microsoft Azure Data Lake (ADL) services. The entire MAG is published under the ODC-By open data license[3] and available for use via Microsoft Azure. Users can use scripting languages such as U-SQL and python not just to access the pre-computed paper recommendations available as part of the MAG but also to generate on-the-fly paper recommendations for arbitrary text input in a performant manner. The means by which this functionality is achieved is described in the following sections.

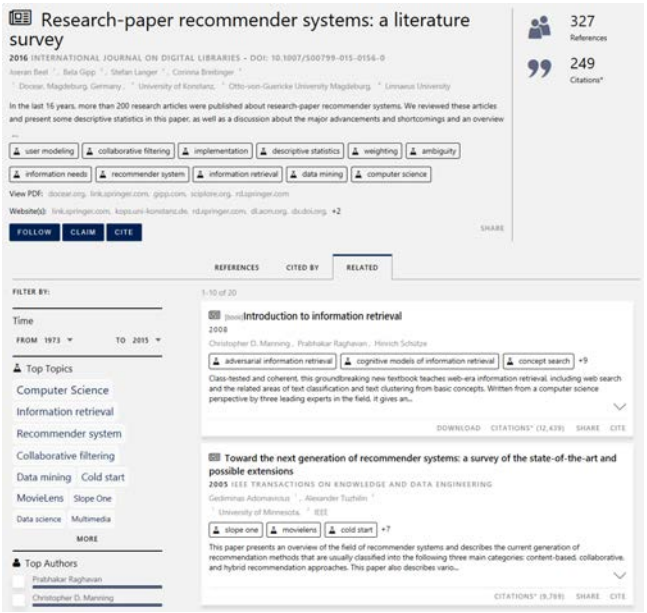


Figure 2: The "Related Papers" tab on Microsoft Academic website paper details page with additional filters visible on the left pane.

[2]Some collaborative filtering approaches assume paper authors will be system users and use citation information to indicate user intent. Even so, new authors/users still suffer from cold-start problem

[3]https://opendatacommons.org/licenses/by/1.0/

The complexity of developing a recommendation system for MA stems from the following feature requirements:

- **Coverage**: Maximize recommendation coverage over the MAG corpus.
- **Scalability**: Recommendation generation needs to be done with computation time and storage requirements in mind as the MAG ingests tens of thousands of new papers each week.
- **Freshness**: All new papers regularly ingested by the MA data pipeline must be assigned related papers and may themselves be presented as 'related' to papers already in the corpus.
- **User Satisfaction**: There needs to be a balance between authoritative recommendations versus novel recommendations so that newer papers are discoverable without compromising the quality of recommendations.

To tackle these requirements we have developed a hybrid recommender system that uses a tunable mapping function to combine both content-based (CB) and co-citation based (CcB) recommendations to produce a final static list of recommendations for every paper in the MAG. As described in [8], our approach employs a *weighted mixed* hybridization approach. Our content-based approach is similar to recent work done on content embedding based citation recommendation [5] but differs mainly in the fact that we employ clustering techniques for additional speedup. Using purely pairwise content embedding similarity for nearest neighbor search is not viable as this is an $O(n^2)$ problem over the entire paper corpus, which in our case would be over 2.56×10^{16} similarity computations.

2 RECOMMENDATION GENERATION

The process outlined in this section describes how the paper recommendations seen on the MA website, as well as those published in the MAG Azure database, are generated. More information on the MAG Azure database is available online[4] but the important thing to note is that our entire paper recommendation dataset is openly available as part of this database, if desired.

The recommender system uses a hybrid approach of CcB and CB recommendations. The CcB recommendations show a high positive correlation with user generated scores, as discussed in section 3, and so we consider them to be of high quality. But the CcB approach suffers from low coverage since paper citation information is often difficult to acquire. To combat this issue we also use content embedding similarity based recommendations. While CB recommendations can be computationally expensive, and of lower quality than CcB recommendations they have the major advantages of freshness and coverage. Only paper metadata including titles, keywords and available abstracts are needed to generate these recommendations. Since all papers in the MAG are required to have a title, we can generate content embeddings for all English documents in the corpus, just relying on the title if need be.

The resulting recommendation lists of both approaches are finally combined using a parameterized weighting function (see section 2.3) which allows us to directly compare each recommendation pair from either list and join both lists to get a final paper recommendation list for nearly every paper in the graph. These lists are dynamically updated as new information comes into the MAG, such as new papers, paper references or new paper metadata such as abstracts. The recommendation lists are thus kept fresh from week to week.

[4] https://docs.microsoft.com/en-us/academic-services/graph/

2.1 Co-citation Based Recommendations

Consider a corpus of papers $P = \{p_1, p_2, p_3 \dots p_n\}$. We use $c_{i,j} = 1$ to denote p_i is citing p_j, 0 otherwise. The co-citation count between p_i and p_j is defined as:

$$cc_{i,j} = \sum_{k=1}^{n} c_{i,k} c_{j,k} \tag{1}$$

When $c_{i,j} \geq 1$, we call that p_i is a co-citation of p_j and vice-versa. Notice, $cc_{i,j} = cc_{j,i}$. This method presumes that papers with higher co-citation counts are more related to each other. Alternatively if two papers are often cited together, they are more likely to be related. This approach of recommendation generation is not new, it was originally proposed in 1973 by Small et al. [27]. Since then others such as [14] have built upon the original approach by incorporating novel similarity measures. While we stay true to the original approach in this paper, we are investigating other co-citation and network based similarity measures as future work.

CcB similarity empirically resembles human behavior when it comes to searching for similar or relevant material. Having access to paper reference information is a requirement for generating CcB recommendations. This presents a challenge since reference lists are often incomplete or just unavailable for many papers, particularly for older papers that were published in print but not digitally. Moreover, CcB recommendations are biased towards older papers with more citations by their very nature. CcB recommendations prove ineffective for new papers that do not have any citations yet and therefore cannot have any co-citations. The MAG contains complete or partial reference information for 31.19% of papers, with each paper averaging approx. 20 references. As a result, only about 32.5% of papers have at least one co-citation. Nevertheless, CcB recommendations tend to be of high quality and therefore cannot be overlooked.

2.2 Content Based Recommendations

2.2.1 Generating paper embeddings. CB recommendations provide a few crucial benefits to overcome the information limitations from CcB recommendations as well as the privacy concerns, system complexity, and cold-start problem inherent in other user based recommender systems approaches such as collaborative filtering. CB recommendations only require metadata about a paper that is easy to find such as its title, keywords, and abstract. We use this textual data from the MAG to generate word embeddings using the well known *word2vec* library [22, 23]. These word embeddings are then combined to form paper embeddings which can then be directly compared using established similarity metrics like cosine similarity [21].

Each paper is vectorized into a paper embedding by first training word embeddings, $\boldsymbol{w}$, using the word2vec library. The parameters used in word2vec training are provided in Table. 1 The training data for word2vec are the titles, and abstracts of all English papers in the MA corpus. At the same time, we compute the term frequency

vectors for each paper (TF) as well as the inverse document frequency (IDF) for each term in the training set. A normalized linear combination of word vectors weighted by their respective TF-IDF values is used to generate a paper embedding, $\boldsymbol{D}$. Terms in titles and keywords are weighed twice as much terms in abstracts, as seen in Eqn. 2. This approach has been applied before for CB document embedding generation, as seen in [24], where the authors assigned a 3× weight to words in paper titles compared to words in the abstract. Finally, the document embedding is normalized, $\hat{\boldsymbol{D}} = \boldsymbol{D}/|\boldsymbol{D}|$. Since we use cosine similarity as a measure of document relevance [21] embedding normalization makes the similarity computation step more efficient since the norm of each paper embedding does not need to computed every time it is compared to other papers, just the value of the dot product between embeddings is sufficient.

$$\boldsymbol{D} = 2. \sum_{\substack{w \in title \\ w \in keywords}} TFIDF(w).\hat{\boldsymbol{w}} + \sum_{w \in abstract} TFIDF(w).\hat{\boldsymbol{w}} \tag{2}$$

2.2.2 Clustering paper embeddings. Our major contribution to the approach of CB recommendation is improving scalability using clustering for very large datasets. The idea to use spherical k-means clustering for clustering large text datasets is presented in [10]. The authors of [10] do a fantastic job of explaining the inherent properties of document clusters formed and make theoretical claims about concept decompositions generated using this approach. Besides the aforementioned benefits, our desire to use k-means clustering stems from the speedup it provides over traditional nearest neighbor search in the continuous paper embedding space. We utilize spherical k-means clustering [15] to drastically reduce the number of pairwise similarity computations between papers when generating recommendation lists. Using trained clusters drops the cost of paper recommendation generation from a $O(n^2)$ operation in the total number of papers to a $O(n * (|c| + \lambda_c))$ operation, where $|c|$ is the trained cluster count and λ_c is the average size of a single paper cluster. Other possible techniques commonly used in CB paper recommender system optimization, such as model classifiers [4], and trained neural networks [5] were investigated but ultimately rejected due to the their considerable memory and computation footprint compared to the simpler k-means clustering approach, particularly for the very large corpus size we are dealing with.

Cluster centroids are initialized with the help of the MAG topic stamping algorithm[25]. Papers in the MAG are stamped according to a learned topic hierarchy resulting in a directed acyclic graph of topics, such that every topic has at least one parent, with the root being the parent or ancestor of every single field of study. As of October, 2018 there are 229,370 topics in the MAG but when the centroids for clustering were originally generated — almost a year ago — there were about 80,000 topics. Of these, 23,533 were topics with no children in the hierarchy, making them the most focused or narrow topics with minimal overlap to other fields. The topic stamping algorithm also assigns a confidence value to the topics stamped for each paper. By using papers stamped with a high confidence leaf node topic we can guess an initial cluster count as well as generate initial centroids for these clusters. Therefore, our initial centroid count $k = 23,533$. Cluster centroids are initialized by taking at most 1000 random papers for each leaf node topic and averaging their respective paper embeddings together. Another initialization mechanism would be to take descriptive text (such as Wikipedia entries) for each leaf node topic and generate an embedding - in much the same way paper embeddings are generated using trained word embeddings - to act as an initial cluster centroid. The centroid initialization method choice is ultimately dependent on the perceived quality of the final clusters and we found that averaging paper embeddings for leaf node topics provided sufficient centroids for initializing clusters. The rest of the hyper parameters used for k-means clustering are provided in Table. 1.

k-means	init. clusters (k) 23,533	max iter. (n) 10	min error (δ) 10^{-3}
word2vec	method *skipgram*	emb. size 256	loss fn. *ns*
	window size 10	max iter. 10	min-count cutoff 10
	sample 10^{-5}	negative 10	

Table 1: Parameters used for spherical k-means clustering and word2vec training.

K-means clustering then progresses as usual until the clusters converge or we complete a certain number of training epochs. Cluster sizes range anywhere from 51 papers to just over 300,000 with 93% of all papers in the MAG belonging to clusters of size 35,000 or less. The distribution of cluster sizes is important since we do not want very large clusters to dominate computation time when generating CB recommendations. Remember that clustering reduces computation complexity of CB recommendation generation from $O(n^2)$ to $O(n * (|c| + \lambda_c))$ because each paper embedding now need only be compared to the embeddings of other papers in the same cluster. Here $|c| = k$ and λ_c is the average cluster size. For a single paper the complexity of recommendation generation is just the second term, i.e. $|c| + \lambda_{c_i}$ where c_i is the size of the cluster-i that the paper belongs to so if λ_{c_i} gets too large then it dominates the computation time. In our pipeline we found that the largest 100 clusters ranged in size from about 40,000 to 300,000 and took up more than 40% of the total computation time of the recommendation process. We therefore limit the cluster sizes that we generate recommendation for to 35,000. For now, papers belonging to clusters larger than this threshold (about 7% of all papers) only have CcB recommendation lists generated. In the future, we plan on investigating sub-clustering or hierarchical clustering techniques to break up the very large clusters so as to be able to generate CB recommendations for them as well.

2.3 Combined Recommendations

Finally, both CcB and CB candidate sets for a paper are combined to create a unified final set of recommendations for papers in the MAG. CcB candidate sets have co-citation counts associated with each paper-recommendation pair. These co-citation counts are mapped to a score between $(0, 1)$ to make it possible to directly compare them with the CB similarity metric. The mapping function used is

a modified logistic function as seen in Eq. 3.

$$\sigma\left(cc_{i,j}\right) = \frac{1}{1 + e^{\theta(\tau - cc_{i,j})}} \tag{3}$$

θ and τ are tunable parameters for controlling the slope and offset of the logistic sigmoid, respectively. Typically, these values can be estimated using the mean and variance of the domain distribution or input distribution to this function, under the assumption that the input distribution is Gaussian-like. While the distribution of co-occurrence counts tended to be more of a Poisson-like distribution with a long tail, the majority of co-occurrence counts (the mass of the distribution) was sufficiently Gaussian-like. We settled on values $\tau = 5$ and $\theta = 0.4$ based on the mean co-occurrence count of all papers and a factor of standard deviation, respectively. In general, changing the tunable parameters of Eq. 3 allows one to weigh CcB versus CB recommendations resulting in different final sets of recommendation lists that balance authoritative recommendations from CcB method with novel recommendations from the CB method. Once CcB and CB recommendation similarities are mapped to the same range $[0, 1]$ comparing them becomes trivial and generating a unified recommendation list involves ordering relevant papers from both lists just based on their similarity to the target paper.

3 USER STUDY

We evaluated the results of the recommender system via an online survey. The survey was set up as follows. On each page of the study, participants were presented with a pair of papers. The top paper was one that had been identified in the MAG as being authored (or co-authored) by the survey participant while the bottom paper was a recommendation generated using the hybrid recommender platform described in the previous section. Metadata for both papers as well as hyperlinks to the paper details page on the MA website were also presented to the participant on this page. The participant was then asked to judge — on a scale of 1 to 5, with 1 being *not relevant* to 5 being *very relevant* — whether they thought the bottom paper was relevant to the top paper (See Fig. 3). Participants could decide to skip a survey page if they were not comfortable providing a score and carry on with the remainder of the survey.

The dataset of paper/recommended-paper pairs to show for a particular participant were generated randomly selecting at most 5 of that participant's authored papers. This was done to ensure familiarity with the papers the participants were asked to grade, which we thought would make the survey less time consuming thereby resulting in a higher response rate. Note that while participants were guaranteed to be authors of the papers, they may not have been authors of the recommended papers. For each of a participant's 5 papers, we then generated at most 10 recommendations using the CcB approach, and 10 recommendations using the CB approach. Some newer papers may have had fewer than 10 co-citation recommendations. This resulted in each participant having to rate at most 100 recommendations. All participants were active computer science researchers and so the survey, as a whole, was heavily biased towards rating computer science papers. We wish to extend this survey to other domains as future work since the MAG contains papers and recommendations from tens of thousands of different research domains. For now, we limited our scope to computer science due to familiarity, the ease of participant access and confidence in participant expertise in this domain.

4 RESULTS AND DISCUSSION

The user survey was sent to all full-time researchers at Microsoft Research and a total of 40 users responded to the survey, resulting in 2409 scored recommendation pairs collected. Of these, 984 were CcB recommendations, 15 recommendation pairs that were both content and co-citation, and 1410 CB recommendation pairs. The raw result dataset is available on GitHub[5]. Since at most 10 recommendations were presented to a user using each of the two methods, we computed P@10 for CcB and CB recommendations as well as P@10 for combined recommendations.

Since we did not include any type of explicit score normalization for participants during the survey, Table 2 shows precision computed assuming, both a user score of at least 3 as a true positive result and another row assuming a score of at least 4 as a true positive result. Recall that users were asked to score recommended paper pairs on a scale of 1 being not relevant to 5 being most relevant. We also compute the normalized discounted cumulative gain (nDCG) for each of the three methods. Note that we use exponential gain, $(2^{score} - 1)/log_2(rank + 1)$ instead of linear gain when computing DCG.

	CcB	CB	Combined
P@10-3	0.315	0.226	0.533
P@10-4	0.271	0.145	0.41
nDCG	0.851	0.789	0.891

Table 2: Evaluation metrics for CcB, CB and combined recommender methods. P@10-N indicates that a user score at least N between $[1, 5]$ is considered a true positive.

We generated Fig. 4 to see how the similarities computed using the MA recommender system lined up with user scores from the study. Each bar in the figure is generated by first aggregating the similarity scores for all paper recommendation pairs that were given a particular score by users. The first column is all paper recommendation pairs given a score of 1 and so on. Each column is then divided into sections by binning the paper recommendation pairs according to the similarity computed using the combined recommender method, e.g. The orange section of the leftmost bar denotes all paper recommendation pairs with a similarity between $[0.6, 0.7)$ that were given a score of 1 by the user study participants. While the absolute values of the similarity is not very important, what is important to gather from this figure is that it shows a clear positive correlation between the similarity values computed by the hybrid recommender platform and user scores. This would seem to indicate that recommendation pairs with higher computed similarity are more likely to be relevant for users, which is the desired outcome for any recommender system. This fact is reinforced by the nDCG values of each of the recommender methods. A combined nDCG of 0.891 indicates a strong correlation between the system's computed rankings and those observed from the survey participants.

[5]https://github.com/akanakia/microsoft-academic-paper-recommender-user-study

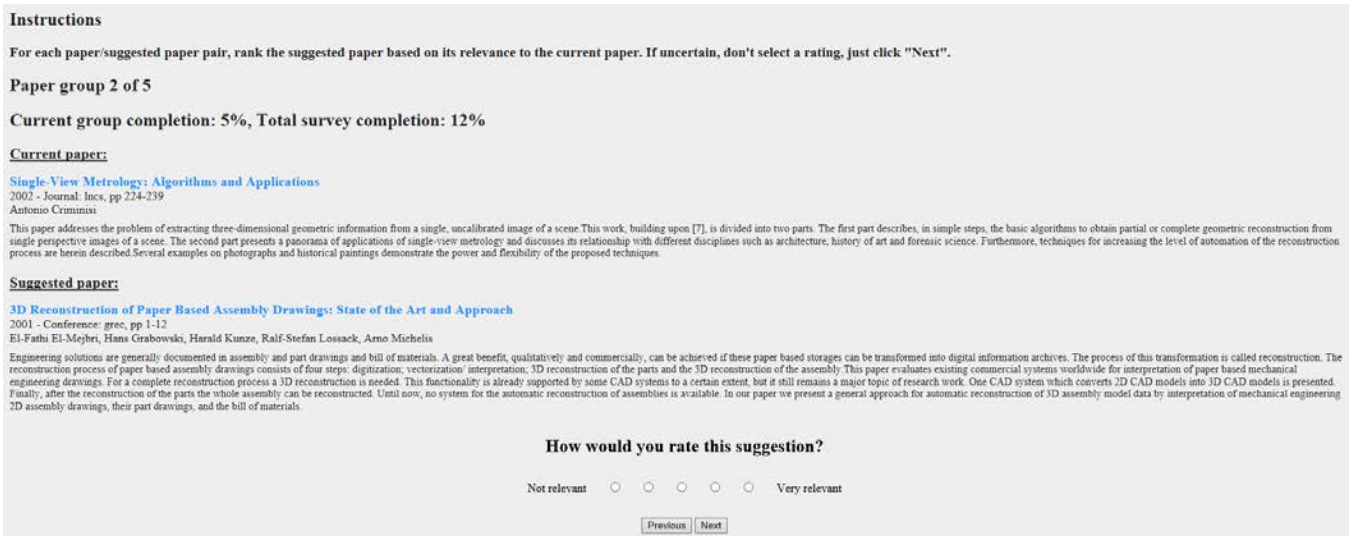


Figure 3: A screenshot of the online recommender system survey.

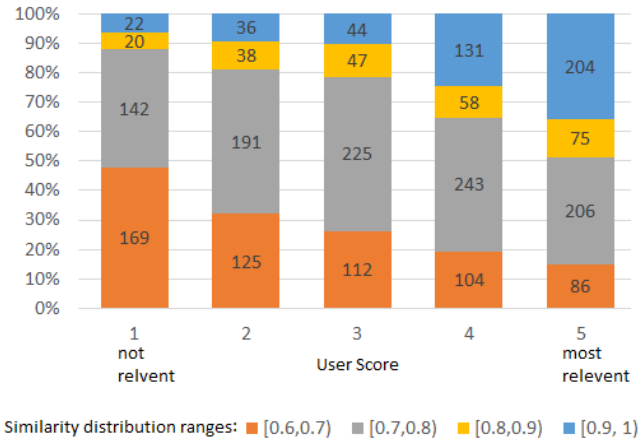


Figure 4: The distribution of similarity ranges over user responses ranging from 1 being *not relevant* to 5 being *most relevant.*

On the other hand, the observed precision values indicate that there is room for improving user satisfaction of the presented recommendations, particularly for the CB method. While we expected CcB recommendations to outperform CB results, having CB recommendations be half as precise as CcB recommendations would seem to indicate that additional effort needs to be spent in improving CB recommender quality.

5 CONCLUSION

A natural question to ask about this approach is why not use other vectorization techniques such as doc2vec[20], fasttext[6] or even incorporate deep learning language models such as ULMFit[16]? While we settled on word2vec for the current production system, we are constantly evaluating other techniques and have this task set aside as future work. A good experiment would be to generate a family of embeddings and analyze recommendation results, perhaps with the aid of a follow-up user study to understand the impact of different document vectorization techniques on the result recommendation set. Another avenue for further research lies in tuning the weights and hyper-parameters of the recommender, such as the θ and τ parameters in the co-citation mapping function (Eq. 3). We hypothesize that a reinforcement learning approach could be used to learn these parameters, given the user study as labeled ground-truth data for the training model.

In the broader scope of evaluating research paper recommender systems, there is a notable lack of literature that compares existing deployed technologies. Furthermore, there is a general lack of data on metrics such as user adoption and satisfaction, and no consensus on which approaches — like the content and co-citation hybrid presented in this paper, or collaborative filtering, and graph analysis to name a few others — prove the most promising in helping to tackle this problem. Part of the reason for this is that large knowledge graphs and associated recommender systems are often restricted behind paywalls, not open access or open source and hence difficult to analyze and compare. We hope to at least partly alleviate this problem by providing the entire MAG and precomputed paper recommendations under the open data license, ODC-By, so that other researchers may easily use our data and reproduce the results presented here as well as conduct their own research and analysis on our knowledge graph.

In conclusion, we presented a scalable hybrid paper recommender platform used by Microsoft Academic that used co-citation and content based recommendations in maximize coverage, scalability, freshness, and user satisfaction. We examined the quality of results produced by the system via a user study and showed a strong correlation between our system's computed similarities and user scores for pairs of paper recommendations. Finally, we made the results of our user study as well the actual recommendation lists used by MA available to researchers to analyze and help further research in research paper recommender systems.

REFERENCES

[1] Luiz Barroso. 2006. Exploring the scholarly neighborhood [Blog Post]. https://googleblog.blogspot.com/2006/08/exploring-scholarly-neighborhood.html.

[2] Joeran Beel, Akiko Aizawa, Corinna Breitinger, and Bela Gipp. 2017. Mr. DLib: recommendations-as-a-service (RaaS) for academia. In *Digital Libraries (JCDL), 2017 ACM/IEEE Joint Conference on*. IEEE, 1–2.

[3] Joeran Beel, Bela Gipp, Stefan Langer, and Corinna Breitinger. 2016. Research-paper recommender systems: a literature survey. *International Journal on Digital Libraries* 17, 4 (Nov. 2016), 305–338.

[4] Steven Bethard and Dan Jurafsky. 2010. Who should I cite: learning literature search models from citation behavior. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. 609–618.

[5] Chandra Bhagavatula, Sergey Feldman, Russell Power, and Waleed Ammar. 2018. Content-Based Citation Recommendation. In *NAACL HLT 2018: 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 238–251.

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606* (2016).

[7] Kurt D Bollacker, Steve Lawrence, and C Lee Giles. 1999. A system for automatic personalized tracking of scientific literature on the web. In *Proceedings of the fourth ACM conference on Digital libraries*. ACM, 105–113.

[8] Robin D. Burke. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-adapted Interaction* 12, 4 (2002), 331–370.

[9] Tsung Teng Chen and Maria R. Lee. 2018. Research Paper Recommender Systems on Big Scholarly Data.. In *PKAW*. 251–260.

[10] Inderjit S. Dhillon and Dharmendra S. Modha. 2001. Concept Decompositions for Large Sparse Text Data Using Clustering. *Machine Learning* 42 (2001), 143–175.

[11] Yuxiao Dong, Hao Ma, Zhihong Shen, and Kuansan Wang. 2017. A Century of Science: Globalization of Scientific Collaborations, Citations, and Innovations. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1437–1446.

[12] Santo Fortunato, Carl T. Bergstrom, Katy Borner, James A. Evans, Dirk Helbing, Stasa Milojevic, Alexander M. Petersen, Filippo Radicchi, Roberta Sinatra, Brian Uzzi, Alessandro Vespignani, Ludo Waltman, Dashun Wang, and Albert Laszlo Barabasi. 2018. Science of science. *Science* 359, 6379 (2018).

[13] Bela Gipp, JÃűran Beel, and Christian Hentschel. 2009. Scienstein : A Research Paper Recommender System. *ICETiCâĂŹ09 : International Conference on Emerging Trends in Computing* (2009), 309–315.

[14] Khalid Haruna, Maizatul Akmar Ismail, Abdullahi Baffa Bichi, Victor Chang, Sutrisna Wibawa, and Tutut Herawan. 2018. A Citation-Based Recommender System for Scholarly Paper Recommendation.. In *International Conference on Computational Science and Its Applications*. 514–525.

[15] Kurt Hornik, Ingo Feinerer, and Martin Kober. 2012. Spherical k-Means Clustering. *Journal of Statistical Software* 50, 1 (2012), 1–22.

[16] Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 328–339.

[17] Wenyi Huang, Zhaohui Wu, Prasenjit Mitra, and C Lee Giles. 2014. Refseer: A citation recommendation system. In *Digital Libraries (JCDL), 2014 IEEE/ACM Joint Conference on*. IEEE, 371–374.

[18] Kris Jack. 2012. Mahout Becomes a Researcher: Large Scale Recommendations at Mendeley. https://www.slideshare.net/KrisJack/mahout-becomes-a-researcher-large-scale-recommendations-at-mendeley.

[19] Ajith Kodakateri Pudhiyaveetil, Susan Gauch, Hiep Luong, and Josh Eno. 2009. Conceptual recommender system for CiteSeerX. In *Proceedings of the third ACM conference on Recommender systems*. ACM, 241–244.

[20] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. *international conference on machine learning* (2014), 1188–1196.

[21] Maake Benard Magara, Sunday O. Ojo, and Tranos Zuva. 2018. A comparative analysis of text similarity measures and algorithms in research paper recommender systems. In *2018 Conference on Information Communications Technology and Society (ICTAS)*.

[22] Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781* (2013).

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[24] Cristiano Nascimento, Alberto H.F. Laender, Altigran S. da Silva, and Marcos AndrÃĽ GonÃğalves. 2011. A source independent framework for research paper recommendation. ACM Press, 297.

[25] Zhihong Shen, Hao Ma, and Kuansan Wang. 2018. A web-scale system for scientific knowledge exploration. *meeting of the association for computational linguistics* (2018).

[26] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June Paul Hsu, and Kuansan Wang. 2015. An Overview of Microsoft Academic Service (MAS) and Applications. In *Proceedings of the 24th International Conference on World Wide Web*. 243–246.

[27] Henry Small. 1973. CoâĂŘcitation in the scientific literature: A new measure of the relationship between two documents. *Journal of The American Society for Information Science* 24, 4 (July 1973), 265–269.

[28] Roberto Torres, Sean M. McNee, Mara Abel, Joseph A. Konstan, and John Riedl. 2004. Enhancing digital libraries with TechLens+. In *Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*. ACM, 228–236.

Topic Structure-Aware Neural Language Model: Unified language model that maintains word and topic ordering by their embedded representations

Noriaki Kawamae
NTT COMWARE
Tokyo
kawamae@gmail.com

ABSTRACT

Our goal is to exploit a unified language model so as to explain the generative process of documents precisely in view of their semantic and topic structures. Because various methods model documents in disparate ways, we are motivated by the expectation that coordinating these methods will allow us to achieve this goal more efficiently than using them in isolation; we combine topic models, embedding models, and neural language models. As we focus on the fact that topic models can be shared among, and indeed complement embedding models and neural language models, we propose Word and topic 2 vec (Wat2vec), and Topic Structure-Aware Neural Language Model (TSANL). Wat2vec uses topics as global semantic information and local semantic information as embedding representations of topics and words, and embeds both words and topics in the same space. TSANL uses recurrent neural networks to capture long-range dependencies over topics and words. Since existing topic models demand time consuming learning and have poor scalability, both due to breaking the document's structure such as order of words and topics, TSANL maintains the orders of words and topics as phrases and segments, respectively. TSANL reduces the calculation cost and required memory by feeding topic recurrent neural networks, and topic specific word networks with these embedding representations. Experiments show that TSANL maintains both segments and topical phrases, and so enhances previous models.

KEYWORDS

Word embeddings, Topic Models, Recurrent Neural Network, Neural Language Model

ACM Reference Format:
Noriaki Kawamae. 2019. Topic Structure-Aware Neural Language Model: Unified language model that maintains word and topic ordering by their embedded representations. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.XXXXXX

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.XXXXXX

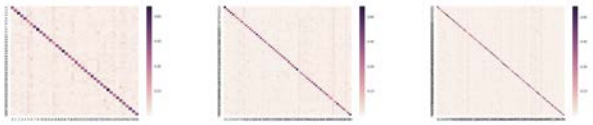


Figure 1: Heat map of topic transition in LDA with (left) #topics=40, (center) #topics=60, (right) #topics=80. Both x and y denote topic id. The higher the probability is, the darker its color is.

1 INTRODUCTION

As many people are expressing their opinions on-line and their volume exceeds the analysis capabilities of any individual or group of individuals, Latent Dirichlet Allocation (LDA) [2] and related topic models have become popular tools for modeling documents. These models uncover the latent structure of a given document collection as topics, and so are enjoying commercial success. Each topic has own multinomial distribution over terms, and has statistical strength and structure that is shared among documents. While their topic representations offer some degree of human interpretability, they are strictly non-sequential models.

As phrases carry more meaning than the sum of the individual words and word meaning depends on context, phrases simplify the human interpretation of topics compared to unigrams. Although some models focus on the order of words [16], they assume that the assignment of a topic is independent from that of other topics. We ran LDA over 2022 ACM papers and show the transition probability between adjacent tokens in Figure 1. This figure shows that 1) these transitions probabilities are asymmetric, and 2) the transition probability to the same topic, p_{ii}, is higher than to other probabilities, $p_{ij}(j \neq i)$, and increases as the #topics increases (e.g. the average of p_{ii} with #topics=40, #topics=60, and #topics=80 is 0.51, 0.57, and 0.61, respectively). Here, we can say that topic appearance and order is affected by their continuity in each document and forms the syntactic structure seen.

Motivated by the above background, we propose both Word and Topic 2 Vec (Wat2Vec) and Topic Structure-Aware Neural Language Model (TSANL) to explain the generative process of documents as based both semantic information and syntactic structures. As we focus on the fact that topic models can be shared among, and indeed complement embedding models [1] and neural language models, we extend embedding models to Wat2vec, and propose TSANL as neural

Table 1: TSANL(+Wat2vec) inherits the interpretability of topic models, the semantic relationships of embedding models, and the prediction power of neural language models: The learning approaches u, and t, denote unified and two-step, respectively.

		Topic model			Embedding models		Neural Language models			
		NTSeg	PDLDA	NOC	CLM	TWE	Topic RNN	TDLM	LLA	TSANL
Reference		[10]	[16]	[12]	[28]	[17]	[4]	[13]	[30]	[this paper]
Semantic	word embeddings	-	-	-	√	√	-	√	√	√
	topic embeddings	-	-	-	√	-	-	√	√	√
Structure	Word	√	√	√	-	-	√	√	-	√
	Topic	-	-	√	-	-	-	-	√	√
embedding learning		-	-	-	u	t	t	u	t	t

language model. Wat2vec uses topic models as global statistical information and local context information as embedding representations of topics and words. Unlike existing topic models, which break the document's structure such as the order of words and topics, TSANL uses these representations with LSTMs [8] to capture their long-range dependencies.

Through experiments, we confirm the key advantages of Wat2vec and TSANL as follows.

Theoretical contribution: Wat2vec embeds both words and topics in the same semantic space, and computes both the contextual word embeddings and topic embeddings for a word and topic given their context. As TSANL marries topic models with neural language models, it maintains the order of words and topics as phrases and segments, respectively.

Practical contribution: As learning phrases and segments is very time-consuming, we develop an efficient sampling technique. TSANL detects candidate phrases and assigns them to the same topic as a constraint in the inference stage.

2 PREVIOUS WORK

2.1 Topic models

To explore word co-occurrence patterns and their order, several topic models form N-grams via a predictive probability distribution for the next word conditioned on the previous words [11, 26, 31]. For example, Phrase Discovering LDA [16] segments a corpus into phrases of varying lengths and assigns topics. NTSeg [10] maintains the document's structure, such as paragraphs and sentences, by employing the notion of word-topics and segment-topics. NOC [12] reveals a tree of topics that captures the semantic relationship between topics from a given corpus as context, and enables each document to maintain its thematic coherence and form N-grams over context. As we are motivated by the observations that a sequence of topics exhibits their dependencies in the form of phrases, we extend these models to preserve the topic order in order to extract the syntactic structure.

2.2 Embedding models

Distributed representations with neural probabilistic language [1] have been proposed to represent words and documents as low-dimensional and dense real value vectors in single semantic spaces, and significant results in many tasks have been demonstrated [23]. Herbelot and Vecchi [7] explored word embeddings and their utility for modeling language semantics. Word2vec [21] is designed to capture a large number of syntactic and semantic word relationships by introducing the continuous Skip-gram model [19]; it achieves better NLP task performance by grouping similar words and gaining distributed representations of words.

Although the probability distributions gained from topic models describe the statistical relationships of word occurrences in the corpus rather than the best choice for feature representation [23], the combination of embedding models and topic models was proposed to represent words in one semantic space. Topical Word Embedding (TWE) [17] forms the topical word embedding by using pre-trained topic structures and concatenating the topical embedding with the word embedding. By integrating this local semantic information with neural language models, our approach captures the long-range dependencies at the embedding level and reduces the computation cost by decreasing the number of parameters.

2.3 Neural language models

While topic models reflect the co-occurrence relationships of words, they fail to sufficiently model the sequential dependency across sentences. Another approach to modeling the sequences of words and topics is employing RNN [20], such as Long Short Term Memory (LSTM) [8]. LSTMs require a large number of parameters, notwithstanding the simplicity of the underlying dynamics, rending the results uninterpretable [30], and models characterize the long range dependencies of a sequence. Topically Driven Neural Language Model (TDLM) [13] has two components, a language model and a topic model, which are jointly trained using neural networks. While Topic RNN [4] can be learned in an end-to-end fashion to directly capture long-range dependencies between words via latent topics, Latent LSTM Allocation (LLA) [30] models sequence of topics using LSTM and generate word similar to LDA. bridges the gap between RNN's and topic models, and model the dynamics of topic evolution. capture semantic meaning in an end-to-end fashion. Our approach uses topic models to learn topics, explore the embeddings of words and topics via learnt topics, and their embedding representation neural language models to maintain phrases, and segments.

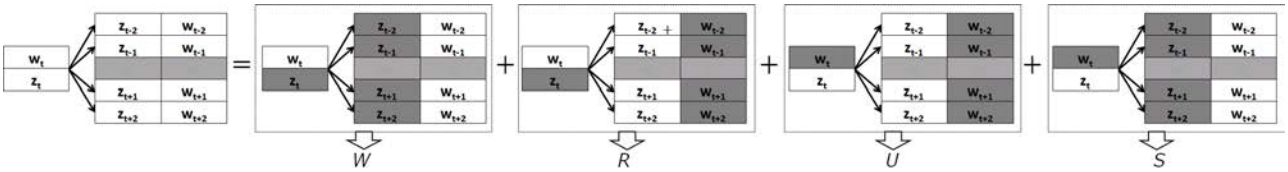


Figure 2: Learning architectures of Skip-gram of Wat2vec: In this figure, w_t is the t-th word token and z_t is the corresponding topic. Skip-gram predicts the surrounding words $(w_{t-2}, w_{t-1}, w_{t+1}w_{t+2})$ and topics $(z_{t-2}, z_{t-1}, z_{t+1}z_{t+2})$ from current word w_t and current topic z_t. We decompose this task into four tasks, where each word and topic predicts the surrounding words and topics individually. Then, Wat2vec learns embeddings using this framework individually with word-word co-occurrence matrix, W, word-topic co-occurrence matrix, R, topic-topic co-occurrence matrix, U, and topic-word co-occurrence matrix, S.

3 WORD AND TOPIC EMBEDDINGS

Inspired by TWE [17], our approach learns both topic and word embeddings to enhance the neural probabilistic language model. Our model receives embedding representations instead of words as input, as these representations learn the latent semantic for each word and so capture useful syntactic and semantic properties. Our proposal Wat2vec (embedding task, Word and Topic 2 vec) learns distributed word representations along with distributed topic representations. Like previous embedding models, Wat2vec requires Skip-gram [19], the state-of-the-art word embedding model, to gain continuous vector representations of words and topics from a given corpus. While Skip-Gram aims to predict context words given a target word in a sliding window, our model extends it to predict surrounding words $(w_{t-2}, w_{t-1}, w_{t+1}w_{t+2})$ and surrounding topics $(z_{t-2}, z_{t-1}, z_{t+1}z_{t+2})$ given current w_t and z_t. As TSNAL aims to predict context topics and words for a target topic and word in a sliding window, Wat2vec learns their embeddings symmetrically to exploit, in equal degree, their semantic and contextual information in the given corpus. That is, Wat2vec allows each word-topic pair w_t, and z_t to have own embedding parameters $\mathbf{a_t}$, and $\mathbf{c_t}$, and to share these parameters between the same words, and topics, respectively. As shown in Figure 2, the target word and its corresponding topic predict the identical surrounding words, and identical surrounding topics, individually.

As mentioned in Levy and Goldberg [14], the Skip-Gram framework with negative sampling (SGNS) is equivalent to implicitly factorizing a word-context matrix, the Pointwise Mutual Information (PMI) of the respective word and context pairs. After making the connection between Skip-Gram and PMI, Levy and Goldberg [14] show how to perform word embedding by spectral dimensionality reduction on the shifted positive PMI (SPPMI) matrix:

$$SPPMI(i,j) = max\{PMI(i,j) - logk, 0\}, \quad (1)$$

where k is a hyperparameter that controls the density of the SPPMI matrix. This matrix can be factorized by either using an eigenvalue method or solving the matrix factorization problem iteratively [29]. As SGNS does not require optimization procedure tuning and can capture the relationships between a word and its context within small sliding windows [19, 21], Wat2vec follows this approach, as do other models [15, 25]. Defining the word co-occurrence matrix, $\mathbf{W} \in \mathbb{V}^V (v = 1, ..., V)$ as the co-occurrence SPPMI matrix instead of the raw frequency matrix, yields word embedding matrix $\mathbf{A}$ and context word embedding matrix $\mathbf{B}$ by factorizing $\mathbf{W}$:

$$L_W = ||W - A^T B||_2^2 + \lambda_w(||A||_2^2 + ||B||_2^2), \quad A \ \ and \ \ B \geq 0, \quad (2)$$

where λ_w is the parameter that prevents overfitting.

Likewise, we construct the topic co-occurrence matrix, $\mathbf{U} \in \mathbb{O}^O (o = 1, ..., O)$ as the co-occurrence SPPMI matrix, and learn topic embedding matrix $\mathbf{C}$ and context topic embedding matrix $\mathbf{D}$ by factorizing matrix $\mathbf{U}$.

$$L_U = ||U - C^T D||_2^2 + \lambda_u(||C||_2^2 + ||D||_2^2), \quad C \ \ and \ \ D \geq 0, \quad (3)$$

where λ_u is the parameter that prevents overfitting.

Following the idea of SPPMI, Wat2vec exploits word and topic embeddings and the corresponding context topic and word, respectively, by factorizing matrix word-topic co-occurrence matrix, $\mathbf{R} \in \mathbb{V}^O$, and topic-word co-occurrence matrix, $\mathbf{S} \in \mathbb{O}^V$:

$$\begin{aligned} L_R &= ||R - A^T D||_2^2 + \lambda_r(||A||_2^2 + ||D||_2^2), \quad A \ \ and \ \ B \geq 0, \\ L_S &= ||S - C^T B||_2^2 + \lambda_s(||C||_2^2 + ||B||_2^2), \quad C \ \ and \ \ B \geq 0, \end{aligned} \quad (4)$$

where λ_r and λ_s are parameters that prevent overfitting.

Because word embedding matrix $\mathbf{A}$ appears in the factorization of both word-word co-occurrence matrix, $\mathbf{W}$, and word-topic co-occurrence matrix ,$\mathbf{R}$, and topic embedding matrix $\mathbf{C}$ appears in the factorization of both topic-topic co-occurrence matrix, $\mathbf{U}$, and topic-word co-occurrence matrix, $\mathbf{S}$, our approach exploits word and topic embeddings by factorizing matrix $\mathbf{W}$, $\mathbf{U}$, $\mathbf{R}$, and $\mathbf{S}$. The objective functions that combine these matrixes are written as::

$$\begin{aligned} L_{W,U,R,S} &= \omega_w(||W - A^T B||_2^2 + \lambda_w(||A||_2^2 + ||B||_2^2)) \\ &+ \omega_u(||U - C^T D||_2^2 + \lambda_u(||C||_2^2 + ||D||_2^2)) \\ &+ \omega_r(||R - A^T D||_2^2 + \lambda_r(||A||_2^2 + ||D||_2^2)) \\ &+ \omega_s(||S - C^T B||_2^2 + \lambda_s(||C||_2^2 + ||B||_2^2)) \end{aligned} \quad (5)$$

where ω_w, ω_u, ω_r, and ω_s are the parameters that control the weights of matrixes $\mathbf{W}$, $\mathbf{U}$, $\mathbf{R}$, and $\mathbf{S}$. This combination enables us to learn, in a balanced manner, the embeddings of words and topics that allow measurement of their similarities.

Following the Alternative Least Squares (ALS) matrix factorization method [9], we obtain the following closed form

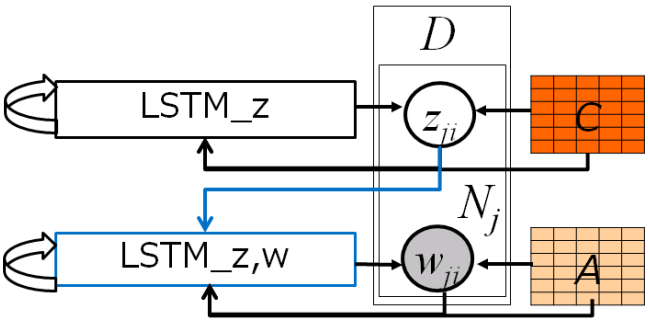


Figure 3: Architecture of TSANL: In this figure, TSANL is a neural language model. The i-th topic, z_{ji}, and word w_{ji} in the j-th document are fed to LSTMs together with their embeddings c_{ji} and a_{ji}, respectively

coordinate updates by iteratively setting the gradient to zero:

$$
\begin{aligned}
a_v &= (\omega_w \sum_{\dot{v}=1}^{V} b_{\dot{v}} b_{\dot{v}}^T + \omega_r \sum_{\dot{o}=1}^{O} d_{\dot{o}} d_{\dot{o}}^T + (\omega_w \lambda_w + \omega_r \lambda_r) I)^{-1} \\
&\times (\omega_w \sum_{\dot{v}=1}^{V} w_{v,\dot{v}} b_{\dot{v}} + \omega_r \sum_{\dot{o}=1}^{O} r_{v,\dot{o}} d_{\dot{o}}) \\
b_{\dot{v}} &= (\omega_w \sum_{v=1}^{V} a_v a_v^T + \omega_s \sum_{o=1}^{O} c_o c_o^T + (\omega_w \lambda_w + \omega_s \lambda_s) I)^{-1} \\
&\times (\omega_w \sum_{v=1}^{V} w_{v,\dot{v}} a_v + \omega_s \sum_{o=1}^{O} s_{o,\dot{v}} c_o) \\
c_o &= (\omega_u \sum_{\dot{o}=1}^{O} d_{\dot{o}} d_{\dot{o}}^T + \omega_s \sum_{\dot{v}=1}^{V} b_{\dot{v}} b_{\dot{v}}^T + (\omega_u \lambda_u + \omega_s \lambda_s) I)^{-1} \\
&\times (\omega_u \sum_{\dot{o}=1}^{O} u_{o,\dot{o}} d_{\dot{o}} + \omega_s \sum_{\dot{v}=1}^{V} s_{o,\dot{v}} b_{\dot{v}}) \\
d_{\dot{o}} &= (\omega_r \sum_{v=1}^{V} a_v a_v^T + \omega_u \sum_{o=1}^{O} c_o c_o^T + (\omega_u \lambda_u + \omega_r \lambda_r) I)^{-1} \\
&\times (\omega_u \sum_{o=1}^{O} u_{o,\dot{o}} c_o + \omega_r \sum_{v=1}^{V} r_{v,\dot{o}} a_v)
\end{aligned}
\tag{6}
$$

where $\mathbf{a_v}$, $\mathbf{b_{\dot{v}}}$, $\mathbf{c_o}$, and $\mathbf{d_{\dot{o}}}$ represent word embedding of the v-th word, context word embedding of the $\dot{v}$-th word, topic embedding of the o-th topic, and context topic embedding of the $\dot{o}$-th topic, respectively.

4 TOPIC STRUCTURE-AWARE NEURAL LANGUAGE MODEL

4.1 Motivation and Methodology

Here, we explain the motivation underlying Topic Structure-Aware Neural Language Model (TSANL); its concept is described using Figure 2. Because TSANL aims to explain the generative process of documents precisely in view of their global semantic and local semantic information and structures such as segments and phrases, it trains LSTMs by using embedding representations of topics and words. Like other topic models, TSANL represents each document as a mixture of underlying topics, $d_j = \{z_{j,1}, ..., z_{j,i}, ..., z_{j,N_j}\}$, and generates each word $w_{j,i}$ from one topic.

Unlike other topic models, TSANL preserves the sequence of ordered topics, instead of topic-collocations. As finding better ways to model long-range dependencies in language modeling is an open research issue [4], our model uses both global semantic and local semantic information to predict succeeding topics and words. That is, TSANL preserves the unit of topic as the document's structure, and segments a document into coherent topic sequences instead of topic-collocations. This allows TSANL, unlike other N-gram topic models [10, 11, 16, 27], to maintain correctly both an observed sequence of words and an observed sequence of topics. Since we assume that the sequence of topics conveys more meaning than the sum of the individual topics, TSANL forms these sequences, segments, by training LSTM, to directly predict the next topic conditioned on the sequence of topics, as does LLA [30]. LSTM keeps the internal state, and at each time step, takes an input and updates its state by applying the transition function consisting of the neural network of the previous time step's state and input. In order to incorporate the local context information, TSANL adds to LSTM the embedding of previous topic, $c_{j,i-1}$, instead of z_{ji}.

In forming a sequence of words over the same topic (phrases), TSANL trains LSTM to predict the embedding representation of the next word conditioned on the sequence of word embeddings on the same topic, unlike other neural language models, topicRNN [4], TDLM [13] and LLA, and to predict the next word that has the embedding representation vector closest to the embedding vector identified by LSTM.

4.2 Generative process

TSANL requires pre-trained topic models to gain embedding representations. With reference in the graphical model shown in Figure 2, we overview the procedure of TSANL as follows:
1. Initialize

- Initialize $LSTM_z$, and $LSTM_{z,w}$
- Initialize topic embedding representations, $\mathbf{C}$
- Initialize word embedding representations, $\mathbf{A}$

2. For each document d in Corpus D:

- For i-th token in j from 1 to N_j:
  - Get topic embedding representation, $a_{j,i}$ from the state of $LSTM_z$
  - Choose topic $z_{j,i}$ that has the topic embedding representation vector closest to the embedding vector, $c_{j,i}$, as found in the topic embedding look-up table, $\mathbf{C}$.
  - if $i \leq 1$ and $z_{j,i}! = z_{j,i-1}$, Chose $LSTM_{z,w}(c_{j,i})$
  - Get word embedding representation $a_{j,i}$ from the state of $LSTM_{z,w}(\mathbf{w})$, where $\mathbf{w}$ represents the sequence of preceding words over the same topic

Table 2: Basic statistics of the datasets in this paper

category		#documents (D)	#vocabulary (V)
20Newsgropus		11,296	10,785
Amazon	Movies and TV	20,197	45,155
	CDs and Vinyl	17,766	37,170

– Choose word $w_{j,i}$ that has the word embedding representation vector closest to embedding vector $a_{j,i}$, as found in the word embedding look-up table, $\mathbf{A}$.

In this process, topic and word embedding vectors are held in the lookup tables, $\mathbf{A} \in \mathbb{K}^{\mathbf{M}\times\mathbf{V}}$(word embedding vector), $\mathbf{C} \in \mathbb{K}^{\mathbf{M}\times\mathbf{K}}$(topic embedding vector), where M is the dimension of the vector, V is the size of vocabulary, and K is the number of topics.

4.3 Phrase detection

This task is optionally performed only once before the training step. Since conventional N-gram topic models obtain N-gram after inference terminates, they require both large memory and much time for their convergence, and yield low quality high-order $N(\geq 3)$-grams [5]. To conquer these problems, our approach mines candidates for these N-grams before the inference arep. By detecting candidates, Wat2vec restricts all constituent terms within each N-gram to those that share the same latent topic, and constructs a trie by inserting them. As this phase inherently makes N-grams of unlimited length, and to avoid phrase detection error, we use a trie [3] to highlight candidate phrases. A trie is an ordered tree structure that is used to store a dynamic set or associative array where the keys are usually strings, and all the descendants of a node have the common prefix of the string associated with that node. As this structure is composed of words instead of characters, we call this tree structure the candidate phrase trie.

4.4 Training

We outline here the execution of the whole learning process and inference step. As with language model learning, TSANL trains LSTMs to learn sequences of topics and words from a given corpus with the embeddings of observed words and topics learnt from this corpus. Each training pattern of these networks is comprised of N time steps in the embeddings of words, $(w_{t-N}...w_{t-1})$, and those of topics, $(z_{t-N}...z_{t-1})$, followed by one word, w_t, and topic, z_t, respectively. To create these sequences, we slide the window along the word and its topic assignment in the given corpus so that each word, and topic is learned from the preceding N words, and topics, respectively. That is, we train $LSTM_z$ to predict the embedding of topic, z_t, from the preceding N words' embedding representations, and $LSTM_{z,s}$ to predict the embedding of word, w_t, associated with topic z from the preceding N words' embedding representations over the same topic.

5 EXPERIMENTS

5.1 Datasets and Experiment design

We conducted evaluations using 20 Newsgroups data sets[1] and the Amazon review data sets[2], as they are well-known and publicly available [6, 17, 18, 28]. First, we selected two subsets from the Amazon review data sets, counted the number of reviews per product, and used the reviews of the top 137/115 products (object ID) in each subset, where each product had at least 100 reviews. Then, we removed the stop words and rare words (frequency lower than 5 in each collection) using NLTK[3], and converted all words to lowercase. Statistics about the data set are shown in Table 2.

To illustrate the utility of semantic relationships between words, we visualize the two-dimensional PCA projection of word embeddings, where we set the number of iterations to 5000. We calculated this projection from all words' embeddings, but show only the top 500 frequent words in Fig 4.

5.2 Document Classification

We evaluate embedding quality by using the quality of document classification results, a popular way of evaluating the effectiveness of a learned document representation yielded by a topic model [24, 25]. Generally speaking, more accurate topic modeling enables better document representation, resulting in improved document classification accuracy. As this experiment aims to evaluate the document-level representation, we use Doc2vec (PD-DBOW) as the document level representation benchmark, TWE(TWE-1) [4] with, GibbsLDA++[5] and CLM as the topic awareness embedding representation benchmark. Unless otherwise stated, documents are classified by Support Vector Machine (SVM) over each representation vector. As in [17], we represent the semantics of a document by aggregating over all topical word embeddings of each word in the document, i.e., $d = \sum_{w\in d} Pr(w|d)a_w$, where $Pr(w|d)$ can be weighted with TFIDF scores of the words in d. Since topic embeddings can be used as the semantic representation of a document by aggregating over all topic embeddings like word embeddings, i.e., $d = \sum_{w\in d} Pr(t|d)c_t$, where $Pr(t|d)$ can be weighted with TFIDF scores of the topics in d, we call these variations of TWE and Wat2vec as TWE(z) and Wat2vec(z), respectively.

Using their document-topic distribution or document embedding, each model was tasked with classifying 20Newsgroup documents, and Amazon reviews into 20 classes, where each corpus was randomly split into training and testing data sets at the ratio of 4:1. To compare the overall performance over all document classes or labels, we compared them using the macro-averaged precision, recall, and F1 coherence measure using gensim[6] over all models. The classifiers of TWE, CLM,

[1] http://qwone.com/ jason/20Newsgroups/
[2] http://jmcauley.ucsd.edu/data/amazon/qa/
[3] NLTK 3.2.4: http://www.nltk.org
[4] https://github.com/largelymfs/topical_word_embeddings/tree/master/TWE-1
[5] http://gibbslda.sourceforge.net
[6] https://radimrehurek.com/gensim/

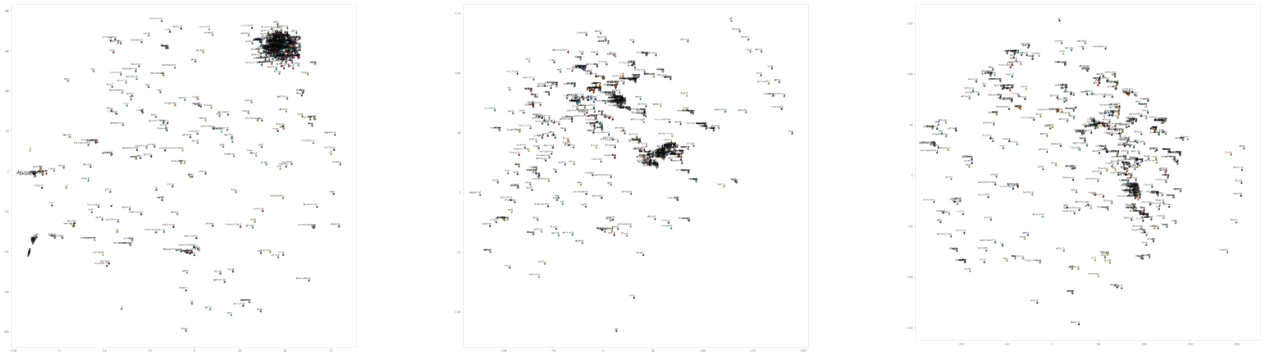


Figure 4: Visualization of the word embeddings from (left) 20News, (center) Movie, (right) CDs, using Wat2vec

Table 3: Performance of various models in document classification task: The skip length (context window size) of all embedding models is set to 5, and the negative sampling number is set to 5.

			20News			Movie			CD		
	K	D	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Doc2Vec	-	100	0.67	0.67	0.67	0.52	0.48	0.49	0.48	0.48	0.47
	-	200	0.67	0.67	0.67	0.56	0.50	0.52	0.51	0.50	0.50
	-	300	0.67	0.67	0.67	0.53	0.50	0.50	0.48	0.47	0.46
TWE	100	100	0.60	0.58	0.56	0.31	0.30	0.29	0.42	0.40	0.40
	100	300	0.63	0.62	0.60	0.32	0.31	0.31	0.49	0.48	0.47
	150	100	0.56	0.55	0.53	0.35	0.32	0.32	0.43	0.41	0.41
	150	300	0.61	0.61	0.59	0.37	0.37	0.36	0.48	0.47	0.46
CLM	100	100	0.72	0.73	0.71	0.43	0.43	0.42	0.49	0.47	0.46
	100	300	0.79	0.78	0.77	0.55	0.56	0.55	0.64	0.63	0.62
	150	100	0.74	0.74	0.72	0.43	0.42	0.41	0.47	0.47	0.46
	150	300	0.77	0.78	0.76	0.58	0.57	0.56	0.62	0.62	0.61
Wat2vec2	100	100	0.74	0.69	0.67	0.46	0.44	0.43	0.50	0.48	0.47
	100	300	0.77	0.74	0.73	0.58	0.59	0.58	0.64	0.64	0.63
	150	100	0.73	0.70	0.68	0.47	0.45	0.45	0.48	0.48	0.46
	150	300	0.77	0.76	0.74	0.60	0.60	0.59	0.63	0.63	0.62
CLM(z)	100	100	0.67	0.39	0.41	0.39	0.38	0.38	0.38	0.39	0.38
	100	300	0.72	0.43	0.45	0.40	0.39	0.39	0.42	0.44	0.42
	150	100	0.61	0.39	0.37	0.36	0.35	0.35	0.41	0.43	0.41
	150	300	0.65	0.42	0.42	0.42	0.39	0.40	0.42	0.45	0.43
Wat2vec(z)	100	100	0.68	0.70	0.69	0.44	0.46	0.44	0.43	0.41	0.39
	100	300	0.64	0.66	0.64	0.36	0.37	0.36	0.44	0.45	0.44
	150	100	0.58	0.59	0.57	0.43	0.44	0.43	0.42	0.44	0.42
	150	300	0.59	0.62	0.60	0.45	0.45	0.44	0.44	0.46	0.44

and Wat2vec were trained for each by applying the topic distribution of the training data set; the SVM classifier received the document embedding vectors as input.

Table 3 shows that our model yielded word representations effective for document classification; document-topic distributions yield better performance in this task than embedding models. While CLM and Wat2vec offer similar performance for word embeddings, Wat2vec(z) outperforms CLM for topic embeddings. This comparison demonstrates that the word embedding obtained from skip-grams with topics play an important role in allowing TSANL to learn LSTMs over topics.

6 DISCUSSION

The problem of employing deep learning for natural language processing is that it is difficult to attach a reasonable interpretation to each dimension of the generated distributions [1, 17, 22, 23]. Topic models and word embedding represent the global and local semantics, as topics encode word co-occurrence patterns in each document and the embedding representations elucidate context information. TSANL incorporates LSTM [8] as it supports both the sequential dependency of word and topic coherence.

7 CONCLUSION

This paper proposes Wat2vec as embedding models and TSANL as neural language models. As TSANL aims to capture long-range dependencies of topics and words, as segments and phrases, respectively, Wat2vec uses topics as global semantic information and local semantic information as embedding representations of topics and words, and embeds both words and topics in the same space. Experiments showed that Wat2vec learns these embedding representations for TSANL to maintain the ordering of both segments and topical phrases, and enhances previous models. Future work is to extend TSANL to deal with streaming data by developing an online algorithm.

REFERENCES

[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* 3 (Mar 2003), 1137–1155.

[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *JMLR* 3 (2003), 993–1022.

[3] Rene De La Briandais. 1959. File Searching Using Variable Length Keys. In *Papers Presented at the the March 3-5, 1959, Western Joint Computer Conference*. 295–298.

[4] Adji B. Dieng, Chong Wang, Jianfeng Gao, and John William Paisley. 2016. TopicRNN: A Recurrent Neural Network with Long-Range Semantic Dependency. *CoRR* abs/1611.01702 (2016).

[5] Ahmed El-Kishky, Yanglei Song, Chi Wang, Clare R. Voss, and Jiawei Han. 2014. Scalable Topical Phrase Mining from Text Corpora. *Proc. VLDB Endow.* 8, 3 (Nov 2014), 305–316.

[6] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW*. 507–517.

[7] Aurélie Herbelot and Eva Maria Vecchi. 2015. Building a shared world: mapping distributional to model-theoretic semantic spaces. In *EMNLP*. 22–32.

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[9] Y. Hu, Y. Koren, and C. Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM*. 263–272.

[10] Shoaib Jameel and Wai Lam. 2013. An Unsupervised Topic Segmentation Model Incorporating Word Order. In *SIGIR*. 203–312.

[11] Noriaki Kawamae. 2014. Supervised *N*-gram topic model. In *WSDM*. 473–482.

[12] Noriaki Kawamae. 2016. *N*-gram over Context. In *WWW*. 1045–1055.

[13] Jey Han Lau, Timothy Baldwin, and Trevor Cohn. 2017. Topically Driven Neural Language Model. In *ACL*. 355–365.

[14] Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding As Implicit Matrix Factorization. In *NIPS*. 2177–2185.

[15] Dawen Liang, Jaan Altosaar, Laurent Charlin, and David M Blei. 2016. Factorization Meets the Item Embedding: Regularizing Matrix Factorization with Item Co-occurrence. In *RecSys*. 59–66.

[16] Robert V. Lindsey, William P. Headden, III, and Michael J Stipicevic. 2012. A phrase-discovering topic model using hierarchical Pitman-Yor processes. In *EMNLP-CoNLL*. 214–222.

[17] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical Word Embeddings. In *AAAI*. 2418–2424.

[18] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *SIGIR*. 43–52.

[19] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* (2013).

[20] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*. 1045–1048.

[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*. 3111–3119.

[22] Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. 2015. Improving Topic Models with Latent Feature Word Representations. *TACL* 3, 4 (2015), 299–313.

[23] Liqiang Niu, Xinyu Dai, Jianbing Zhang, and Jiajun Chen. 2015. Topic2Vec: Learning distributed representations of topics. In *IALP*. 193–196.

[24] Bei Shi, Wai Lam, Shoaib Jameel, Steven Schockaert, and Kwun Ping Lai. 2017. Jointly Learning Word Embeddings and Latent Topics. In *SIGIR*. 375–384.

[25] Tian Shi, Kyeongpil Kang, Jaegul Choo, and Chandan K. Reddy. 2018. Short-Text Topic Modeling via Non-negative Matrix Factorization Enriched with Local Word-Context Correlations. In *WWW*. 1105–1114.

[26] H. Wallach. 2006. Topic Modeling: Beyond Bag-of-Words. In *ICML*. 977–984.

[27] X. Wang, A. McCallum, and X. Wei. 2007. Topical N-Grams: Phrase and Topic Discovery, with an Application to Information Retrieval.. In *ICDM*. 697–702.

[28] Guangxu Xun, Yaliang Li, Jing Gao, and Aidong Zhang. 2017. Collaboratively Improving Topic Discovery and Word Embeddings by Coordinating Global and Local Contexts. In *KDD*. 535–543.

[29] Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic Word Embeddings for Evolving Semantic Discovery. In *WSDM*. 673–681.

[30] Manzil Zaheer, Amr Ahmed, and Alexander J. Smola. 2017. Latent LSTM Allocation: Joint Clustering and Non-Linear Dynamic Modeling of Sequence Datay. In *ICML*. 3967–3976.

[31] Qingyuan Zhao, Murat A. Erdogdu, Hera Y. He, Anand Rajaraman, and Jure Leskovec. 2015. SEISMIC: A Self-Exciting Point Process Model for Predicting Tweet Popularity. In *KDD*. 1513–1522.

Fairness in Algorithmic Decision Making: An Excursion Through the Lens of Causality

Aria Khademi
Pennsylvania State University
khademi@psu.edu

Sanghack Lee
Purdue University
lee2995@purdue.edu

David Foley
Pennsylvania State University
djf47@psu.edu

Vasant Honavar
Pennsylvania State University
vhonavar@psu.edu

ABSTRACT

As virtually all aspects of our lives are increasingly impacted by algorithmic decision making systems, it is incumbent upon us as a society to ensure such systems do not become instruments of unfair discrimination on the basis of gender, race, ethnicity, religion, etc. We consider the problem of determining whether the decisions made by such systems are discriminatory, through the lens of causal models. We introduce two definitions of group fairness grounded in causality: *fair on average causal effect* (FACE), and *fair on average causal effect on the treated* (FACT). We use the Rubin-Neyman *potential outcomes* framework for the analysis of cause-effect relationships to robustly estimate FACE and FACT. We demonstrate the effectiveness of our proposed approach on synthetic data. Our analyses of two real-world data sets, the Adult income data set from the UCI repository (with gender as the protected attribute), and the NYC Stop and Frisk data set (with race as the protected attribute), show that the evidence of discrimination obtained by FACE and FACT, or lack thereof, is often in agreement with the findings from other studies. We further show that FACT, being somewhat more nuanced compared to FACE, can yield findings of discrimination that differ from those obtained using FACE.

ACM Reference Format:
Aria Khademi, Sanghack Lee, David Foley, and Vasant Honavar. 2019. Fairness in Algorithmic Decision Making: An Excursion Through the Lens of Causality. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313559

1 INTRODUCTION

With the growing adoption of algorithmic decision making systems, e.g., AI and machine learning systems, across many real-world decision making scenarios on the Web and elsewhere, there is a pressing need to make sure that such systems do not become vehicles of unfair discrimination, inequality, and social injustice [2, 3]. Of particular interest in this context is the task of detecting and preventing discrimination or unfair treatment of individuals or groups on the basis of gender, race, religion, etc. Such discrimination is traditionally addressed using one of two legal frameworks: *disparate treatment* (which aims to enforce procedural fairness, namely, the equality of treatment that prohibits the use of the protected attribute in the decision process); and *disparate impact* [3] (which aims to guarantee outcome fairness, namely, the equality of outcomes between protected groups relative to other groups). It is clear that enforcing procedural fairness within the disparate treatment framework does not guarantee non-discrimination within the disparate impact framework.

There is growing interest in algorithmic decision making systems that are demonstrably *fair* (see [4] for a review). Much of this literature relies on precise definitions that quantify fairness to avoid discrimination with respect to *protected* attributes, e.g., race, gender, on the basis of the legal notions of disparate treatment or disparate impact [3] (see [3, 4, 32, 40, 60] for reviews). Some examples include: fairness through unawareness [14], individual fairness [11], equalized odds [15, 54], calibration [8], demographic (or statistical) parity [6, 21–23], the 80% rule (disparate impact) [13, 55], representational fairness [33, 56], and fairness under composition [12].

Unfortunately, choosing the appropriate definition of fairness in a given context is extremely challenging due to a number of reasons. First, depending on the relationship between a protected attribute and data, enforcing certain definitions of fairness can actually increase discrimination [28]. Second, different definitions of fairness can be impossible to satisfy simultaneously [4, 8, 26]. Many of these difficulties can be attributed to the fact that fairness criteria are based *solely* on the joint probability distribution of the random variables of interest, namely, $\hat{Y}$ (predicted outcome), Y (actual outcome), $\tilde{X}$ (features), and A (sensitive attributes). [15] recently showed any such definition for fairness of a predictor that depends merely on the joint probability distribution is not necessarily capable of detecting discrimination. Hence, it is tempting to approach the problem of fairness through the lens of causality [1].

Answering questions of fairness through the lens of causality entails replacing the question "Is the decision discriminatory with respect to a protected attribute?" by: "Does the protected attribute have a causal effect on the decision?" A practical difficulty in using this approach is that, in general, establishing a causal relationship between a protected attribute and a decision requires the results of experimental manipulation of the protected attribute. Fortunately, however, existing frameworks for determining causal effects from observational data [20, 36] provide a rich set of theoretical results as well as practical tools for elucidating causal effects, and

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313559

specifically, answering questions about *counterfactuals* or *potential outcomes*, i.e., results of *hypothetical* experimental interventions from observational data, whenever it is possible to do so. Hence, there is a growing body of work (see [32] for a recent review) focused on explicitly causal (as opposed to purely joint distribution based or *observational*) definitions for fairness (e.g., [5, 7, 25, 28–30, 35, 50, 53, 57–59]). While some, e.g., [59], have focused on testing fairness (or conversely, determining whether there is discrimination), others, e.g., [25, 28] have sought to design machine learning algorithms that yield predictive models that are demonstrably fair. However, most of the existing work on defining fairness in causal terms has focused on variants of individual fairness. Against this background, we focus on robust methods for detecting and quantifying discrimination against protected groups, which is a necessary prerequisite for developing predictive models that are provably non-discriminatory.

Contributions. We reduce the problem of quantifying discrimination against protected groups to the well-studied problem of estimating the causal effect of some variable(s) on a target (outcome) variable. We introduce two explicitly causal definitions of fairness in a population, *fair on average causal effect* (FACE), and in a protected group, *fair on average causal effect on the treated* (FACT), both with respect to a protected attribute (e.g., gender, race). We use the Rubin-Neyman *potential outcomes* framework [20, 45, 49] for robust estimation of FACE and FACT. We demonstrate the effectiveness of the proposed approach in detecting and quantifying group fairness using synthetic data, as well as two real-world data sets: the Adult income data from the UCI repository [10] (with gender being the protected attribute), and the NYC Stop and Frisk data (with race being the protected attribute). We show that the evidence of discrimination, or lack thereof, obtained by FACE and FACT is often in agreement with other studies. We further show that FACT, being somewhat more nuanced compared to FACE, can yield findings of discrimination that differ from those obtained using FACE.

2 FAIRNESS: A CAUSAL PERSPECTIVE

Assume we have observational data on a population of individuals. Let $\tilde{X} \in \mathcal{X}$ be the vector of non-protected attributes, $A \in \mathcal{A} = \{a, a'\}$ be a binary protected attribute, and $Y \in \mathcal{Y}$ an outcome of interest. The question we want to answer is: Are individuals being discriminated against, on average, with respect to outcomes or decisions Y on the basis of a protected attribute A? From a causal perspective, such a question is *equivalent* to the following question: Does A have a causal effect on Y? In other words, how much would Y change, on average, were the value of A to change? Both Structural Causal Models [36] and the Rubin-Neyman Causal Model (RCM) [20] (also called the *potential outcomes* model) offer methods for estimating such causal effects from observational data.

We introduce two explicitly causal definitions for fairness "on average" in a population or a protected group (as opposed to causal definitions of individual fairness, e.g., counterfactual fairness [28]) with respect to a protected attribute (e.g., gender, race). Let $Y_i^{(a)}$ and $Y_i^{(a')}$ be the potential outcomes of a data point i had their value of A been a and a', respectively. Let $h : \mathcal{X} \times \mathcal{A} \to \mathcal{Y}$ be a decision function (or a predictive model trained using machine learning) that is used to support decision making. $\mathbb{E}[\cdot]$ is the expectation of a random variable. We define the following.

Definition 2.1. (FACE: Fair on Average Causal Effect). A decision function h is said to be fair, on average over all individuals in the population, with respect to A, if $\mathbb{E}[Y_i^{(a)} - Y_i^{(a')}] = 0$.

Definition 2.2. (FACT: Fair on Average Causal Effect on the Treated). A decision function h is said to be fair with respect to A, on average over individuals with the same value of A, if $\mathbb{E}[Y_i^{(a)} - Y_i^{(a')} \mid A_i = a] = 0$.

Example. Imagine we are given the hiring data of a company containing demographic information about applicants, as well as A = {male, female} as their gender, and Y = {hired, rejected} as whether they were hired by the company. Our task is to determine whether the company's hiring decisions are fair on average with respect to gender. FACE contrasts the expected outcomes (i.e., hiring) between men vs. women with the expectation taken over the entire population. FACT contrasts the expected outcomes observed for a specific protected group (e.g., women) and the hypothetical (counterfactually inferred) outcomes for the group had they not been members of the protected group (with the expectation taken only over the members of the protected group), e.g., hiring outcomes for women contrasted with outcomes for the same individuals had their gender been different with all other attributes remaining unchanged. Obviously, such counterfactual outcomes cannot be obtained from observational data[1] and ought to be estimated.

3 ESTIMATING FACE AND FACT

We use tools offered by the potential outcomes framework [20] to estimate FACE and FACT. These tools rely on the following key assumptions: i) *Consistency* which requires that for a data point i, the potential outcome of i under any level of treatment a, i.e., Y_i^a, equals the *actual* outcome observed for that data point, Y_i^{obs}, had they been exposed to treatment a. Formally, under consistency, $Y_i^{obs} = a\, Y_i^{(a)} + a'\, Y_i^{(a')}$ would hold for all i. This assumption, used in existing literature [7, 34, 35, 38], is a rather natural one to make in our setting. ii) *Positivity* which asserts that the probability $Pr(A = a \mid X = x) > 0$ for all values of A. In our setting, this means each value of the protected attribute has a non-zero probability. iii) *Stable Unit Treatment Value Assumption (SUTVA)* [47] which consists of two sub-assumptions: 1) Absence of *interference* between individuals [9], which means that an individual's potential outcome is unaffected by the treatment assigned to any other individual. While this assumption is plausible in our setting, it may be violated in some settings, in which case, such violations should be accounted for [16]. 2) Presence of only one form of treatment (and control). For example, if a treatment involves administering a drug, then all individuals who take the drug, take it in the same form (e.g., injection). This assumption is trivially satisfied in our setting because treatment is simulated by the protected attribute. iv) *Unconfoundedness of the treatment mechanism* which implies that given a set of observables, the potential outcomes of each individual are jointly independent of the corresponding treatment [46]. Unconfoundedness cannot be verified or contradicted entirely on the

[1] This is called the Fundamental Problem of Causal Inference (FPCI) from observational data [18].

basis of observational data. However, sensitivity analysis [31, 42] can be a useful tool for analyzing the estimated causal effects under violations of the unconfoundedness assumption. Strong ignorability refers to the combination of unconfoundedness and positivity [43]. Strong ignorability is a sufficient condition for the causal effect to be identifiable [16] and is equivalent to the *back-door criterion* [37], which is required for identifiability of the causal effects in Pearl's model of causality [37]. In our work, as in the case of existing work on causal definitions of fairness [35], we assume strong ignorability.

3.1 Estimating and Interpreting FACE

We use Inverse Probability Weighting (IPW), also known as Inverse Probability of Treatment Weighting (IPTW) in Marginal Structural Models (MSM) [39] to estimate FACE. Specifically, for each individual i, we calculate a *stabilized weight*: $sw_i = \frac{Pr(A_i=a)}{Pr(A_i=a \mid \tilde{X}_i=\tilde{x}_i)}$ (call it the weight model). We obtained stabilized weights using the R package *ipw* (version 1.0-11) [52]. Assigning such a weight to every data point, we generate a "pseudo-population" in which there are sw_i copies of each data point i. Subsequently, the associative parameter β in the weighted regression (call it the outcome model) of the (continuous) outcome Y on the protected attribute A: $\mathbb{E}[Y^{(A)}] = \delta + \beta A + \tilde{\theta}^\top \tilde{X}$, would be the causal effect of A on Y. For a binary output Y, we use the weighted logistic regression model: $\text{logit}\,(\mathbb{E}[Y^{(A)}]) = \delta + \beta A + \tilde{\theta}^\top \tilde{X}$. In the absence of unmeasured confounders, if *either* the weight model *or* the outcome model are correctly specified, then $\hat{\beta}$ is an unbiased estimator of the average causal effect [39]. For example, suppose Y is salary and A is gender. At the chosen level of statistical significance α, $\hat{\beta} = 0$ implies that salary is fair with respect to gender on average over the entire population of individuals; $\hat{\beta} \neq 0$ implies that, on average, women's salary differs from that of men by a factor of $\hat{\beta}$ (across the entire population). For a continuous outcome Y, $\hat{\beta}$ is simply the average causal effect of A on Y. For a binary outcome Y, $\hat{\beta}$ corresponds to the causal odds ratio of salary for women versus men.

3.2 Estimating and Interpreting FACT

We use *matching* to estimate FACT. Consider the example of salary discrimination based on gender. For a woman, we can never observe what the salary would have been, *had she been a man* (i.e., her counterfactual salary). Hence, we estimate the counterfactual salary as follows [20]: 1) Using a suitable matching technique (see Section Matching Methods), we match the woman i, to a man j who is closest to i with respect to a distance measure $d(i, j)$. 2) The matching process is repeated as needed until matches are of acceptable quality (see Section Quality of Matches). 3) After matching, we use the salary of the matched man j (i.e., Y_j), as the counterfactual salary of the woman i.

Matching Methods. The results of matching depend on the choice of distance measure $d(\cdot, \cdot)$ as well as the matching process. Several matching methods exist (see [51] for a survey). In what follows, for simplicity and brevity, we refer to individuals with protected attribute set to $A = a$ as the *treated* individuals and those with the protected attribute set to $A = a'$ as the *controlled* individuals. We used the matching methods implemented within the R package *MatchIt* (version 3.0.2) [17] with all parameters set to their default values unless otherwise noted: (i) Exact Matching (EM); (ii) Nearest Neighbor Matching (NNM) with propensity score [43]. Following [48], we estimated the propensity scores using the logit link and transformed them to the linear scale. Then, we ran NNM with replacement, based on the linear propensity scores, and discarded the data points (both from treated and controlled) that fall outside the support of the distance measure; (iii) Nearest Neighbor Matching with a Propensity Caliper (NNMPC). NNMPC includes only matches within a certain number of standard deviations of the distance measure and discards the rest. In NNMPC, we use the same procedure as in NNM, augmented with a caliper = 0.25 [44], resulting in the matches outside 0.25 times the standard deviation of the (transformed) linear propensity score, being discarded; (iv) Mahalanobis Metric Matching within the Propensity Caliper [48] (MMMPC). MMMPC determines for each data point, a "donor pool" of available matches within the propensity caliper. Mahalanobis metric matching is then performed among the data points chosen in the previous step mimicking blocking in randomized experiments [48]. We ran MMMPC with caliper, replacement, and discarding strategy as described above in NNM; and (v) Full Matching (FM) [41]. We used the same distance measure and discarding strategy as described above in NNM.

Quality of Matches. To ensure accurate estimation of FACT, it is crucial to measure the "goodness-of-match." If the data points are well matched, then one can proceed to estimate FACT. Common diagnostics for examining the quality of match include both numerical and graphical criteria. Among the numerical criteria, following [48], we compare the standardized difference in the means of the treated and the controlled data points in terms of the distance measure. We denote the absolute value of this difference in means on the original, and matched data, by $\overline{D}_{a,a'}$, and $\overline{D}^m_{a,a'}$, respectively. For the match to be of good quality, $\overline{D}^m_{a,a'}$ has to be close to 0. Among the graphical criteria, we use quantile-quantile (QQ), and jitter plots recommended by [17, 51].[2]

Outcome Analysis After Matching. With good quality matched pairs identified, we can proceed to conduct outcome analysis for FACT estimation. Matching methods often assign appropriate weights to the matched data points to balance the treated and controlled data distributions. After obtaining the weights via matching, we run the following weighted regression models: $\mathbb{E}[Y^{(A)}] = \delta + \gamma A + \tilde{\theta}^\top \tilde{X}$, for continuous, and $\text{logit}\,(\mathbb{E}[Y^{(A)}]) = \delta + \gamma A + \tilde{\theta}^\top \tilde{X}$, for binary outcomes, both *on the matched data set*, to estimate FACT. The estimated coefficient for A in the equations above, i.e., $\hat{\gamma}$, estimates FACT. The resulting estimate is "doubly robust" in that if *either* the matching model, *or* the outcome model, are correctly specified, $\hat{\gamma}$ would be statistically consistent [17].

Interpreting $\hat{\gamma}$ as a Measure of FACT. Suppose Y is salary and A is gender. At the chosen level of statistical significance α, $\hat{\gamma} = 0$ implies that there is no significant difference in expected salary for women compared to what their salary would have been had

[2] We avoid the commonly used hypothesis tests for *assessing feature balance* in diagnosing the quality of matches because such tests have been shown to be misleading in general [19].

they been men (with all non-protected attributes $\tilde{X}$ remaining unchanged, a condition that is approximated by counterfactual inference using matching), thus implying no gender-based discrimination in salary for women; $\hat{\gamma} \neq 0$ implies that, on average, women's salary is statistically significantly different from what it would have been, had they been men, thus implying gender-based discrimination in salary. For a continuous outcome Y, e.g., the salary in US dollars, if statistically significant, $\hat{\gamma} \neq 0$ means that on average, *considering men and women that are matched based on their feature vector* $\tilde{X}$, the difference between women's salary and that of men is $\hat{\gamma}$. For a binary outcome, e.g., salaries binarized with an arbitrary threshold τ, $\hat{\gamma}$ is the causal odds ratio of women's salary compared to that of men, for those women and men who are similar.

Impact of Unmeasured Confounders on $\hat{\gamma}$. What if the strong ignorability assumption (i.e., no hidden confounders) is violated? In the absence of unmeasured confounding, matching estimators are unbiased if the matching model is specified correctly, i.e., if balance is achieved over the *observed attributes*. However, it is conceivable that the results of matching could change in the presence of *unobserved confounders* (i.e., hidden bias). We perform sensitivity analysis [31, 42] to investigate the degree to which the unmeasured confounders impact $\hat{\gamma}$. Let Γ be the odds ratio of matched (using any matching method) data points i and j receiving a treatment. Sensitivity analysis proceeds by first assuming $\Gamma = 1$ (i.e., no hidden bias). Then, it increases Γ (e.g., $1, \ldots, 5$), thus mimicking the presence of hidden bias, and examines the resulting changes to statistical significance of $\hat{\gamma}$. The Γ at which the significance of the upper bound for the p-value would change (e.g., from < 0.05 to > 0.05) is the point at which $\hat{\gamma}$ is no longer robust to hidden bias. We ran sensitivity analysis using the R package *rbounds* (version 2.1) [24].

4 EXPERIMENTS AND RESULTS

We tested our approach on a synthetic data set (where the discrimination based on a protected attribute can be varied in a controlled fashion), and two real-world data sets that have been previously used in studies of fairness. In each case, we designated a protected attribute and estimated FACE and FACT as measures of discrimination based on that attribute. We run all of our statistical significance tests with $\alpha = 0.05$. We proceed to describe the data sets, experiments, as well as our FACE and FACT analyses in detail.

4.1 Data sets

Synthetic data set. We generated 1000 data points, each with a feature vector $\tilde{X} = (X_1, \ldots, X_5)$, a protected attribute $A \in \{0, 1\}$, and an outcome variable Y according to the following: $X_1, \ldots, X_5 \overset{iid}{\sim} \mathcal{N}(0, 1)$; $A \mid X \sim \text{Bernoulli}\left(\text{logit}^{-1}(\sum_{i=1}^{5} X_i)\right)$; $Y \mid X, A = \sum_{i=1}^{5} X_i W_i$, where $\tilde{W} = (W_1, \ldots, W_5)$ is a weight vector (fixed for all data points) with each element drawn randomly in [0,1]. The resulting generative model ensures there are no hidden confounders and there is no discrimination, as measured by FACE and FACT, with respect to the outcome variable Y on the basis of the protected attribute A.

The Adult data set. The Adult income data set [27][3], contains information about individuals as well as their salaries. The data set includes 48842 individuals each with 14 attributes, 6 continuous and 8 categorical, including demographic and work-related information such as age, gender, hours of work per week, etc. We examined whether there is gender-based discrimination in salaries by designating *gender* as the sensitive attribute. We encoded categorical variables using one-hot-encoding and removed data records with missing values, yielding a data set with 46033 individuals and 45 features (excluding gender, the protected attribute). We designated the outcome Y to be a binary variable denoting whether the person's annual salary is $> \$50K$ (Y=1), or $\leq \$50K$ (Y= 0).

The NYC Stop and Frisk (NYCSF) data set. We retrieved the publicly available stop, search, and frisk data from The New York Police Department (NYPD)[4] website which serves demographic and other information about drivers stopped by the NYC police force. Our question is whether the arrests made after stops have been discriminatory with respect to race. Following [28], we restricted our experiment to the year 2014 yielding a total of 45787 records. We selected the subset of records corresponding to only Black-Hispanic and White men. We designated *race* as the protected attribute with $A = 1$ denoting Black-Hispanic and $A = 0$ denoting White. We dropped the data records with missing values and encoded categorical variables with one-hot-encoding. The resulting data consist of 7593 records each with 73 features (excluding race, the sensitive attribute). The outcome Y denotes whether an arrest was made ($Y = 1$), or not ($Y = 0$).

4.2 FACE Check: Fairness Analysis Using FACE

We report our analysis of fairness using FACE, for the synthetic, Adult, and NYCSF data sets. The estimated FACE ($\hat{\beta}$) are shown in Table 1. In all cases, the null hypothesis is $H_0 : \beta = 0$. In the case of synthetic data, we find insufficient evidence to reject H_0, suggesting the outcome is fair with respect to the protected attribute (an expected conclusion given the design of the generative model in Section 4.1, which ensures that the outcome is fair with respect to the protected attribute). In the case of Adult data, we reject H_0 and find that $\hat{\beta}$, the average causal effect of gender on salaries, is -1.069. This means that on average, over the entire population, the odds of women having a salary $> \$50K$ a year is $\exp(-1.069) \approx 0.34$ times that of men, suggesting gender-based discrimination against women as measured by FACE. This finding is in agreement with the conclusions reported in [30, 35]. In the case of NYCSF data, we reject H_0 and find that $\hat{\beta}$ is 0.273 which means that on average, the odds of Black-Hispanics being arrested after a stop by the police, is $\exp(0.273) \approx 1.31$ times that of Whites, suggesting possible racial bias against non-Whites.

4.3 FACT Check: Fairness Analysis Using FACT

We report our analysis of fairness using FACT, for the synthetic, Adult, and NYCSF data sets.

Matching Quality Analyses. Because the quality of matched pairs used to estimate FACT impacts the conclusions that can be

[3] https://archive.ics.uci.edu/ml/datasets/adult
[4] https://www1.nyc.gov/site/nypd/stats/reports-analysis/stopfrisk.page

Table 1: Estimates of FACE ($\hat{\beta}$) obtained on the synthetic, Adult, and NYCSF data sets.

Data set	$\hat{\beta}$	Standard Error	P-value
Synthetic	-1.130×10^{-16}	6.991×10^{-17}	**0.106**
Adult	**−1.069**	3.614×10^{-1}	0.003
NYCSF	**0.273**	1.259×10^{-1}	0.030

drawn using it, we compare the FACT estimates obtained using several widely-used matching methods described in Section 3.2. We present some analyses to verify that the generated matches are of sufficiently high quality for estimating FACT.

We observe that before matching, $\overline{D}_{a,a'}$ is 1.6400, 3.3508, and 1.1616, on the synthetic, Adult, and NYCSF data sets, respectively. The matching methods dramatically reduced $\overline{D}_{a,a'}$ on all of the data sets (see Table 2). Overall, NNM and FM achieved the lowest $\overline{D}^m_{a,a'}$ as compared to other matching methods on all data sets. The greater the number of pairs that are matched, the harder it is to achieve balance, and the trade-off between the two can be application dependent. We observed that considering the trade-off between the number of matches and $\overline{D}^m_{a,a'}$, FM yields higher quality matches on all data sets as compared to other methods.

The QQ plots are generated for each feature in each data set. In Figure 1 we show the QQ plots before and after FM for the first three features of the synthetic data set. The features lie far away from the 45 degree line before FM. After FM, the features are much better aligned to the diagonal line showing a more desirable feature balance. We also show the jitter plots of FM on all data sets in Figure 2. It is clear that the distribution of propensity scores of the treated and controlled data points are very similar to each other after matching. Having verified that the results of matching are of adequate quality, we proceed to use them for estimating FACT.

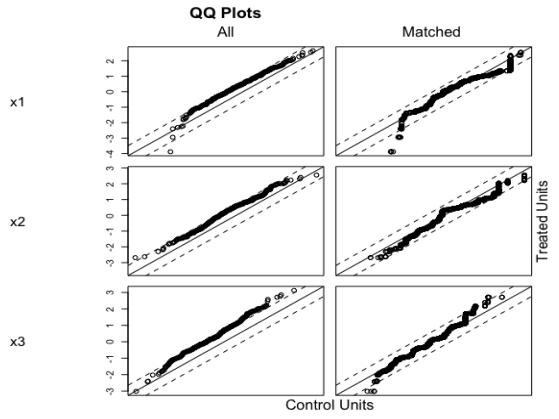


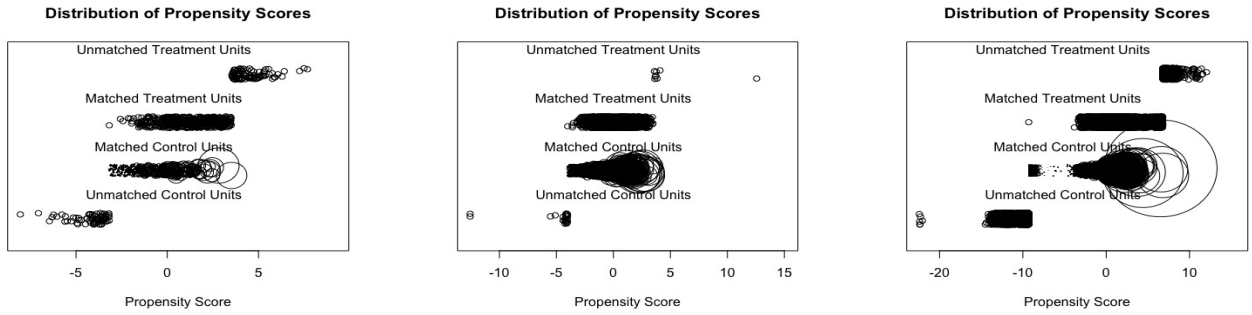
Figure 1: QQ plots of the first three features from the synthetic data set before (left) and after (right) FM.

FACT Estimates. The results of FACT analyses on the synthetic, Adult, and NYCSF data sets are summarized in Table 2 (Note that EM did not yield any matches and hence is omitted from Table 2). In all cases, the null hypothesis is $H_0 : \gamma = 0$. In the case of synthetic data, FACT analyses show that for NNM and MMMPC, there is not enough evidence to reject H_0. The p-values in the case of NNMPC and FM are < 0.05, but the magnitude of the estimated $\hat{\gamma}$ is close to zero. We conclude that the synthetic data set is fair on average with respect to FACT. On the Adult data, we can reject H_0, suggesting that salaries of women are significantly lower than those of men who match them on the non-protected attributes. For example, using FM, we find that $\hat{\gamma} = -0.573$, thus the odds of women earning $> \$50K$ a year, is $\exp(-0.573) \approx 0.56$ times that of men. We conclude that in the Adult data, there is evidence of gender-based discrimination in salary, on average, against women. On the NYCSF data, interestingly, FACT analyses show that H_0 cannot be rejected, suggesting a lack of evidence for racial bias, on average, in arrests after stops (when Black-Hispanics are compared with Whites who match them on non-protected attributes). This conclusion contradicts the finding of racial bias based on counterfactual fairness analysis (Supplementary Material S6 in [28]) which suggests discrimination against *individuals*, as well as FACE analysis (see Section 4.2). We conjecture that the apparent discrepancy can be explained by noting that (i) fairness (or discrimination) on average does not necessarily imply individual-level fairness (or individual-level discrimination), and (ii) FACT compares the observed outcomes of members of a protected group with the hypothetical (counterfactual) outcomes they would have experienced had they not been members of the protected group (with all non-protected attributes remaining unchanged), whereas FACE compares such counterfactual outcomes on the entire population.

Impact of Unmeasured Confounders. We ran sensitivity analysis of our estimates of FACT for $\Gamma = 1, \ldots, 10$ (where larger values of Γ correspond to greater bias introduced by hidden confounders) on the Adult and NYCSF data sets. We find that all of our estimates obtained with various matching methods are quite robust to hidden confounder bias. Specifically, on the Adult data set, for all matching methods except FM, the estimates are robust to such bias, and for FM, they are robust up to $\Gamma = 4.5$, which corresponds to a fairly large amount of bias. On the NYCSF data set, estimates obtained via NNM and MMMPC are robust to hidden confounder bias, and NNMPC and FM are robust up to Γ equals 8.5, and 3, respectively. These results mean that our FACT estimates (and hence our findings of discrimination on the basis of protected attributes, or lack thereof) are fairly robust to hidden confounder bias.

5 SUMMARY AND DISCUSSION

We have approached the problem of detecting whether a group of individuals that share a sensitive attribute, e.g., race, gender, have been subjected to discrimination in an algorithmic decision-making system, through the lens of causality. We have introduced two explicitly causal definitions of group fairness: *fair on average causal effect* (FACE), and *fair on average causal effect on the treated* (FACT). We have shown how to robustly estimate FACE and FACT, and use the resulting estimates to detect and quantify discrimination based on specific attributes (e.g., gender, race). The results of our experiments on synthetic data show that our proposed methods are effective at detecting and quantifying group fairness. Our analyses of the Adult data set for evidence of gender-based discrimination



(a) Synthetic data set. **(b) NYCSF data set.** **(c) Adult data set.**

Figure 2: Jitter plots of distribution of the propensity scores on the linear logit scale after FM on the synthetic (left), NYCSF (middle), and Adult (right) data sets. Each circle represents a data point. Area of the circle is proportional to the weight given to the data point. Female, Black-Hispanic = treated, and male, White = controlled.

Table 2: Estimates of FACT ($\hat{\gamma}$) obtained via various matching methods on the synthetic, NYCSF, and Adult data sets.

Synthetic data set						
Matching Method	# of Treated Matches	# of Control Matches	$\overline{D}^m_{a,a'}$	$\hat{\gamma}$	Standard Error	P-value
NNM	454	155	0.0032	-6.972×10^{-17}	4.947×10^{-17}	0.159
NNMPC	454	168	0.0234	-9.196×10^{-17}	3.020×10^{-17}	0.002
MMMPC	454	174	0.0308	7.424×10^{-18}	3.192×10^{-17}	0.816
FM	454	386	0.0031	-8.263×10^{-17}	3.702×10^{-17}	0.026
Adult data set						
Matching Method	# of Treated Matches	# of Control Matches	$\overline{D}^m_{a,a'}$	$\hat{\gamma}$	Standard Error	P-value
NNM	13330	4922	0.0009	−0.637	0.128	6.560×10^{-7}
NNMPC	13301	5258	0.0714	−0.650	0.113	1.050×10^{-8}
MMMPC	13301	5838	0.0584	−0.586	0.131	7.650×10^{-6}
FM	13330	15320	0.0009	−0.573	0.115	5.700×10^{-7}
NYCSF data set						
Matching Method	# of Treated Matches	# of Control Matches	$\overline{D}^m_{a,a'}$	$\hat{\gamma}$	Standard Error	P-value
NNM	2605	1305	0.0001	0.049	0.186	0.788
NNMPC	2605	1414	0.0266	0.246	0.160	0.124
MMMPC	2605	1264	0.0231	0.324	0.183	0.078
FM	2605	4958	0.0000	0.155	0.171	0.364

in salary, and of the NYCSF data set for evidence of racial bias in arrests after traffic stops, yield evidence of discrimination, or lack thereof, that is often in agreement with other studies.[5] We show on the real-world data that our estimates of FACE and FACT are robust to unmeasured confounding. Our results further show on the real-world data that FACE and FACT based findings do not always agree. Our FACT analyses also demonstrate that group-fairness (or discrimination) does not necessarily imply individual-level fairness (or individual-level discrimination).

Some directions for further research include: relaxing the assumption that the data are independent and identically distributed (i.i.d.) in settings where individuals are related to each other through family ties or other relationships; examining the relationships between different causal notions of fairness; and designing automated decision support systems that are demonstrably non-discriminatory with respect to given outcome(s) and protected attribute(s).

Acknowledgements. This work was funded in part by grants from the NIH NCATS through the grant UL1 TR000127 and TR002014 and by the NSF through the grants 1518732, 1640834, and 1636795,

[5] The regression and matching-based methods we employed to estimate FACE and FACT adjust for covariates that might be potential confounders of the protected attribute, which although necessary in general, may be unnecessary in the case of gender and race, because they are unlikely to be caused by any other covariate. Consequently, the reported estimates of FACE and FACT are likely to represent *direct* causal effects as opposed to *total* causal effects.

the Edward Frymoyer Endowed Professorship in Information Sciences and Technology at Pennsylvania State University and the Sudha Murty Distinguished Visiting Chair in Neurocomputing and Data Science funded by the Pratiksha Trust at the Indian Institute of Science (both held by Vasant Honavar). The content is solely the responsibility of the authors and does not necessarily represent the official views of the sponsors.

REFERENCES

[1] C. Barabas, M. Virza, K. Dinakar, J. Ito, and J. Zittrain. 2018. Interventions over Predictions: Reframing the Ethical Debate for Actuarial Risk Assessment. In *Conference on Fairness, Accountability and Transparency*. 62–76.

[2] S. Barocas, E. Bradley, V. Honavar, and F. Provost. 2017. Big Data, Data Science, and Civil Rights. *arXiv preprint arXiv:1706.03102* (2017).

[3] S. Barocas and A. D. Selbst. 2016. Big data's disparate impact. *Cal. L. Rev.* 104 (2016), 671.

[4] R. Berk, H. Heidari, S. Jabbari, M. Kearns, and A. Roth. 2017. Fairness in criminal justice risk assessments: the state of the art. *arXiv preprint arXiv:1703.09207* (2017).

[5] F. Bonchi, S. Hajian, B. Mishra, and D. Ramazzotti. 2017. Exposing the probabilistic causal structure of discrimination. *International Journal of Data Science and Analytics* 3, 1 (2017), 1–21.

[6] T. Calders, F. Kamiran, and M. Pechenizkiy. 2009. Building classifiers with independency constraints. In *Data mining workshops, 2009. ICDMW'09. IEEE international conference on*. IEEE, 13–18.

[7] S. Chiappa and T. PS. Gillam. 2018. Path-specific counterfactual fairness. *arXiv preprint arXiv:1802.08139* (2018).

[8] A. Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data* 5, 2 (2017), 153–163.

[9] D. R. Cox. 1958. Planning of experiments. (1958).

[10] D. Dheeru and E. Karra Taniskidou. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[11] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. 2012. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. ACM, 214–226.

[12] C. Dwork and C. Ilvento. 2018. Fairness Under Composition. *arXiv preprint arXiv:1806.06122* (2018).

[13] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. 2015. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 259–268.

[14] N. Grgic-Hlaca, M. B. Zafar, K. P. Gummadi, and A. Weller. 2016. The case for process fairness in learning: Feature selection for fair decision making. In *NIPS Symposium on Machine Learning and the Law*, Vol. 1. 2.

[15] M. Hardt, E. Price, and N. Srebro. 2016. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*. 3315–3323.

[16] M. A. Hernan and J. M. Robins. 2018. *Causal Inference*. Boca Raton: Chapman & Hall/CRC, forthcoming.

[17] D. E. Ho, K. Imai, G. King, and E. A. Stuart. 2011. MatchIt: nonparametric preprocessing for parametric causal inference. *Journal of Statistical Software* 42, 8 (2011), 1–28.

[18] P. W. Holland. 1986. Statistics and Causal Inference. *J. Amer. Statist. Assoc.* 81, 396 (1986), 945–960.

[19] K. Imai, G. King, and E. Stuart. 2008. Misunderstandings Among Experimentalists and Observationalists about Causal Inference. *Journal of the Royal Statistical Society, Series A* 171, part 2 (2008), 481–502.

[20] G. W. Imbens and D. B. Rubin. 2015. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press.

[21] J. E. Johndrow and K. Lum. 2017. An algorithm for removing sensitive information: application to race-independent recidivism prediction. *arXiv preprint arXiv:1703.04957* (2017).

[22] F. Kamiran and T. Calders. 2009. Classifying without discriminating. In *Computer, Control and Communication. IC4 2009. 2nd International Conference on*. IEEE, 1–6.

[23] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma. 2012. Fairness-aware classifier with prejudice remover regularizer. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 35–50.

[24] L. Keele. 2010. An overview of rbounds: An R package for Rosenbaum bounds sensitivity analysis with matched data. *White Paper. Columbus, OH* (2010), 1–15.

[25] N. Kilbertus, M. R. Carulla, G. Parascandolo, M. Hardt, D. Janzing, and B. Schölkopf. 2017. Avoiding discrimination through causal reasoning. In *Advances in Neural Information Processing Systems*. 656–666.

[26] J. Kleinberg, S. Mullainathan, and M. Raghavan. 2016. Inherent trade-offs in the fair determination of risk scores. *arXiv preprint arXiv:1609.05807* (2016).

[27] R. Kohavi. 1996. Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Vol. 96. 202–207.

[28] M. J. Kusner, J. Loftus, C. Russell, and R. Silva. 2017. Counterfactual fairness. In *Advances in Neural Information Processing Systems*. 4069–4079.

[29] M. J. Kusner, C. Russell, J. R. Loftus, and R. Silva. 2018. Causal Interventions for Fairness. *arXiv preprint arXiv:1806.02380* (2018).

[30] J. Li, J. Liu, L. Liu, T. D. Le, S. Ma, and Y. Han. 2017. Discrimination detection by causal effect estimation. In *Big Data (Big Data), 2017 IEEE International Conference on*. IEEE, 1087–1094.

[31] W. Liu, S. J. Kuramoto, and E. A. Stuart. 2013. An introduction to sensitivity analysis for unobserved confounding in nonexperimental prevention research.

Prevention Science 14, 6 (2013), 570–580.
[32] J. R. Loftus, C. Russell, M. J. Kusner, and R. Silva. 2018. Causal Reasoning for Algorithmic Fairness. *arXiv preprint arXiv:1805.05859* (2018).
[33] C. Louizos, K. Swersky, Y. Li, M. Welling, and R. Zemel. 2015. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830* (2015).
[34] D. Madras, E. Creager, T. Pitassi, and R. Zemel. 2019. Fairness through Causal Awareness: Learning Causal Latent-Variable Models for Biased Data. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. ACM, 349–358.
[35] R. Nabi and I. Shpitser. 2018. Fair inference on outcomes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 2018. NIH Public Access, 1931.
[36] J. Pearl. 2009. *Causality*. Cambridge university press.
[37] J. Pearl. 2010. The foundations of causal inference. *Sociological Methodology* 40, 1 (2010), 75–149.
[38] J. Pearl. 2019. On the Interpretation of do(x). *Journal of Causal Inference, forthcoming* (2019).
[39] J. M. Robins, M. A. Hernán, and B. Brumback. 2000. Marginal Structural Models and Causal Inference in Epidemiology. *Epidemiology* 11, 5 (2000), 550–560.
[40] A. Romei and S. Ruggieri. 2014. A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review* 29, 5 (2014), 582–638.
[41] P. R. Rosenbaum. 1991. A characterization of optimal designs for observational studies. *Journal of the Royal Statistical Society. Series B (Methodological)* (1991), 597–610.
[42] P. R. Rosenbaum. 2005. Sensitivity analysis in observational studies. *Encyclopedia of Statistics in Behavioral Science* 4 (2005), 1809–1814.
[43] P. R. Rosenbaum and D. B. Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70, 1 (1983), 41–55.
[44] P. R. Rosenbaum and D. B. Rubin. 1985. Constructing a control group using multivariate matched sampling methods that incorporate the propensity score. *The American Statistician* 39, 1 (1985), 33–38.
[45] D. B. Rubin. 1974. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology* 66, 5 (1974), 688.
[46] D. B. Rubin. 1978. Bayesian inference for causal effects: The role of randomization. *The Annals of statistics* (1978), 34–58.
[47] D. B. Rubin. 1980. Randomization analysis of experimental data: The Fisher randomization test comment. *J. Amer. Statist. Assoc.* 75, 371 (1980), 591–593.
[48] D. B. Rubin. 2001. Using propensity scores to help design observational studies: application to the tobacco litigation. *Health Services and Outcomes Research Methodology* 2, 3-4 (2001), 169–188.
[49] D. B. Rubin. 2005. Causal inference using potential outcomes: Design, modeling, decisions. *J. Amer. Statist. Assoc.* 100, 469 (2005), 322–331.
[50] C. Russell, M. J. Kusner, J. Loftus, and R. Silva. 2017. When worlds collide: integrating different counterfactual assumptions in fairness. In *Advances in Neural Information Processing Systems*. 6417–6426.
[51] E. A. Stuart. 2010. Matching methods for causal inference: A review and a look forward. *Statistical Science: a review journal of the Institute of Mathematical Statistics* 25, 1 (2010), 1.
[52] W. M. van der Wal, R. B. Geskus, et al. 2011. Ipw: an R package for inverse probability weighting. *J Stat Softw* 43, 13 (2011), 1–23.
[53] T. J. VanderWeele and W. R. Robinson. 2014. On causal interpretation of race in regressions adjusting for confounding and mediating variables. *Epidemiology (Cambridge, Mass.)* 25, 4 (2014), 473.
[54] M. B. Zafar, I. Valera, M. G. Rodriguez, and K. P. Gummadi. 2017. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1171–1180.
[55] M. B. Zafar, I. Valera, M. G. Rogriguez, and K. P. Gummadi. 2017. Fairness Constraints: Mechanisms for Fair Classification. In *Artificial Intelligence and Statistics*. 962–970.
[56] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. 2013. Learning fair representations. In *International Conference on Machine Learning*. 325–333.
[57] J. Zhang and E. Bareinboim. 2018. Fairness in Decision-Making–The Causal Explanation Formula. In *32nd AAAI Conference on Artificial Intelligence*.
[58] L. Zhang, Y. Wu, and X. Wu. 2016. Situation testing-based discrimination discovery: a causal inference approach. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. 2718–2724.
[59] L. Zhang, Y. Wu, and X. Wu. 2017. A Causal Framework for Discovering and Removing Direct and Indirect Discrimination. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*.
[60] I. Zliobaite. 2015. A survey on measuring indirect discrimination in machine learning. *arXiv preprint arXiv:1511.00148* (2015).

MVAE: Multimodal Variational Autoencoder for Fake News Detection

Dhruv Khattar
International Institute of Information Technology
Hyderabad, India
dhruv.khattar@research.iiit.ac.in

Jaipal Singh Goud
International Institute of Information Technology
Hyderabad, India
jaipal.singh@research.iiit.ac.in

Manish Gupta*
International Institute of Information Technology
Hyderabad, India
manish.gupta@iiit.ac.in

Vasudeva Varma
International Institute of Information Technology
Hyderabad, India
vv@iiit.ac.in

ABSTRACT

In recent times, fake news and misinformation have had a disruptive and adverse impact on our lives. Given the prominence of microblogging networks as a source of news for most individuals, fake news now spreads at a faster pace and has a more profound impact than ever before. This makes detection of fake news an extremely important challenge. Fake news articles, just like genuine news articles, leverage multimedia content to manipulate user opinions but spread misinformation. A shortcoming of the current approaches for the detection of fake news is their inability to learn a shared representation of multimodal (textual + visual) information. We propose an end-to-end network, Multimodal Variational Autoencoder (MVAE), which uses a bimodal variational autoencoder coupled with a binary classifier for the task of fake news detection. The model consists of three main components, an encoder, a decoder and a fake news detector module. The variational autoencoder is capable of learning probabilistic latent variable models by optimizing a bound on the marginal likelihood of the observed data. The fake news detector then utilizes the multimodal representations obtained from the bimodal variational autoencoder to classify posts as fake or not. We conduct extensive experiments on two standard fake news datasets collected from popular microblogging websites: Weibo and Twitter. The experimental results show that across the two datasets, on average our model outperforms state-of-the-art methods by margins as large as ~6% in accuracy and ~5% in F_1 scores.

CCS CONCEPTS

• **Information systems → Social networks**; **Multimedia and multimodal retrieval**; • **Computing methodologies → Neural networks**.

*Author is also a Principal Applied Researcher at Microsoft. (gmanish@microsoft.com)

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313552

KEYWORDS

Fake news detection, multimodal fusion, variational autoencoders, microblogs

ACM Reference Format:
Dhruv Khattar, Jaipal Singh Goud, Manish Gupta, and Vasudeva Varma. 2019. MVAE: Multimodal Variational Autoencoder for Fake News Detection. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313552

1 INTRODUCTION

The change to neoteric methods of news consumption in recent times has brought the issue of fake news and misinformation to the forefront of discussion. As thousands of new news articles proliferate social media networks every day, with each having neither a credibility nor a validation check, an ecosystem fuelled by misinformation (the inadvertent sharing of false information) and disinformation (the deliberate creation and sharing of information known to be false) has been established. Social media networks have enabled news articles to evolve from traditional text-only news to news with images and videos which can provide a better storytelling experience and has the ability to engage more readers. Recent fake news articles leverage this very change to visual context aided news. Fake news articles can now contain misrepresented, irrelevant and forged images to mislead readers. Peer-to-peer networks (primarily social media networks) allow fake news (propaganda) to be targeted at users who are more likely to accept and share a particular message. Fake news has been in the limelight in recent years for having an extensive negative effect on public events. A major turning point of realization was the 2016 U.S. presidential elections. It was believed that within the final three months leading up to the election, fake news favoring either of the two nominees was accepted and shared by more than 37 million times on Facebook [1]. This makes the task of detecting fake news a crucial one.

Figure 1 shows three examples of fake news from the Twitter dataset. Each tweet has certain textual content and an image associated with it. For the tweet on the left, both the image and text indicate that it is probably a fake news. In the tweet on the right, the image does not add substantial information but the text indicates that it may be a fake news. In the tweet in the middle, it is difficult to reach a conclusion from the text, but the morphed image suggests that it is possibly a fake news. This example reflects the hypothesis



Text: **Elephant carved out of solid rock. Magnificent!**



Text: **New species of fish found in Arkansas.**



Text: **Woman, 36, gives birth to 14 children from 14 different fathers**

Figure 1: Fake News examples from the Twitter dataset

that pairs of visual and textual information can give better insights into fake news detection.

On a conceptual level, the task of detecting fake news has undergone a variety of labels from misinformation to rumor. The target of our study is detection of news content that is fabricated and can be verified to be false. Rumor and fake news detection techniques range from traditional learning methods to deep learning models. Initial approaches [4] tried to detect fake news using only linguistic features extracted from the text content of news stories. Ma et al. [14] explored the possibility of representing tweets with deep neural networks by capturing temporal-linguistic features followed by Chen et al. [5] who built upon this by introducing attention mechanism into the RNNs.

Recent works [26] in the field of deep learning to detect fake news have shown performance improvements over traditional methods due to their enhanced ability to extract relevant features. Jin et al. [9] combine visual, textual and social context features, using an attention mechanism to make predictions about fake news. Wang et al. [26] use an additional event discriminator to learn common features shared among all the events with the goal of doing away with non-transferable event-specific features, and claim that they can handle novel and newly emerged events better. A shortcoming of the existing models is that they do not have any explicit objective function to discover *correlations across the modalities*.

Our work, inspired by the idea of autoencoders tries to learn shared representations in a multimodal setting. Kingma et al. [12] proposed that latent variable models act as a powerful approach to generatively model complicated distributions and brought forth the ideas of Variational Autoencoder (VAE). VAEs can learn probabilistic latent variable models by optimizing a bound on the marginal likelihood of the observed data. Overcoming the limitations of the current models, we propose a multimodal variational autoencoder capable of learning shared (visual + textual) representations, trained to discover correlations across modalities in tweets. The VAE is then coupled with a classifier to detect fake news.

To summarize, the contributions of our work are as follows:

- We propose a novel approach for classifying social media posts using only the content of the post, i.e., the text and the attached image.
- The proposed MVAE model uses a Multimodal Variational Autoencoder trained jointly with a Fake News Detector to detect if a post is fake or not.
- We extensively evaluate the performance of our model on two real-world datasets. The results reveal that our proposed model learns better multimodal features and outperforms the state-of-the-art multimodal fake news detection models.
- We show that our proposed model is able to discover correlations across the modalities and thus come up with better multimodal shared representations.

The rest of this paper is organized as follows: In Section 2, we discuss previous work on fake news detection and basics of autoencoders. In Section 3, we present our proposed model and its different components. In Section 4, we describe the datasets, baselines and provide the implementation details of our proposed model. Results and analysis are shown in Section 5, and we conclude with a brief summary in Section 6.

2 RELATED WORK

The task of fake news detection is similar to various other interesting challenges ranging from spam detection [29] to rumor detection [24] to satire detection [20]. As every individual may have their very own intuitive definition of such related ideas, each paper embraces its own definition. Following the previous work [21, 22], we specify that the target of our study is detecting news content that is fabricated and can be verified as false.

A few early studies tried to detect fake news based on linguistic features extracted from the text of news stories. Castillo et al. [4] employ a set of linguistic features such as special characters, sentiment positive/negative words, emojis, etc. to detect fake news. Popat et al. [19] use stance and language stylistic features like assertive verbs, discourse markers, etc. to assess the credibility of a claim. Context-free grammar rules were used to identify deception in Feng et al. [6]. Ma et al. [14] were the first to explore the possibility of representing tweets with deep neural networks by capturing temporal-linguistic features. Chen et al. [5] incorporated attention mechanism into recurrent neural networks (RNNs) to pool out distinct temporal-linguistic features with a particular focus.

The social connection established as a result of the interaction between users and tweets give birth to rich social context for posts. Wu et al. [28] infer embeddings of social media user profiles and leverage an LSTM network over propagation pathways to classify fake news. Liu et al. [13] model the propagation path of a news story as a multivariate time series and detect fake news through propagation path classification with a combination of RNNs and CNNs. Ma et al. [15] learn representations of a tweet using recursive neural models based on tree-structured neural networks. Jin et al. [8] used hand-crafted social context features such as the number of followers, retweets, etc. Most social context features are unstructured and usually involve an intensive amount of manual labour to collect. Also, they cannot provide sufficient information for newly emerged events.

Recent studies have shown that visual features (images) play a very important role in detecting fake news [27]. However, verifying the credibility of multimedia content on social media has received less amount of scrutiny. Extraction of basic features of attached images has been explored in [18]. However, these features are still hand-crafted and can hardly represent complex distributions of image content.

Deep neural networks have yielded immense success in learning image and textual representations. They have been successfully applied to various tasks including image captioning [10, 25], visual

question answering [2], and fake news detection [8, 26]. Jin et al. [8] proposed a model which extracts the visual, textual and social context features, and fuses them by attention mechanism. Wang et al. [26] learn event-invariant features using an adversarial network along with a multimodal feature extractor. However, both these models do not have any explicit objective to discover *correlations across the modalities*.

To overcome the limitations of the existing multimodal fake news detectors, we propose a multimodal variational autoencoder (MVAE) that learns a shared representation of both the modalities, text as well as image. The multimodal variational autoencoder is trained to reconstruct both the modalities from the learned shared representation and thus discovers *correlations across modalities*. We jointly train the multimodal variational autoencoder along with a classifier to detect fake news. Also, we use less information than our baselines to detect fake news i.e. we don't use any social or event related information.

3 THE PROPOSED MVAE MODEL

3.1 MVAE Overview

We propose a novel deep multimodal variational autoencoder (MVAE) to address the problem of fake news detection. The basic idea behind MVAE is to learn a unified representation of both the modalities of a tweet's content. The overall architecture of the proposed MVAE is illustrated in Figure 2. It has three main components:

- Encoder: It encodes the information from text and image into a latent vector.
- Decoder: It reconstructs back the original image and text from the latent vector.
- Fake News Detector: It uses the learned shared representation (latent vector) to predict if a news is fake or not.

3.2 Encoder

The inputs to the encoder are the text of the post as well as the image attached to it, and it outputs a shared representation of the features learnt from both the modalities. The encoder can be broken down into two sub-components: textual encoder and visual encoder.

3.2.1 Textual Encoder. The input to the textual encoder is the sequential list of words in the posts, $T = [T_1\ T_2\ \dots\ T_n]$, where n is the number of words in the text. Each word $T_i \in T$ in the text is represented as a word embedding vector. The embedding vector for each word is obtained with a deep network which is pre-trained on the given dataset in an unsupervised way.

To extract features from textual content, we use recurrent neural networks (RNNs) with Long-Short Term Memory (LSTM) cells. RNNs are a class of artificial neural networks which utilize sequential information and maintain history through their intermediate layers. A vanilla RNN has an internal state whose output at every time-step can be expressed in terms of the previous time step. However, it has been seen that vanilla RNNs suffer from a problem of vanishing and exploding gradients [7, 17]. This leads to the model learning inefficient dependencies between words that are a few steps apart. To overcome this problem, LSTM extends the basic RNN by storing information over long time periods by their use of memory units and efficient gating mechanisms.

Let $[h_1\ h_2\ \dots\ h_n]$ represent states of the LSTM and its state updates satisfy the following equations:

$$\left[f_t,\ i_t,\ o_t\right] = \sigma\left(Wh_{t-1} + Ux_t + b\right) \tag{1}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tanh\left(Wh_{t-1} + Ux_t + b\right) \tag{2}$$

$$h_t = o_t \circ \tanh\left(c_t\right) \tag{3}$$

where, σ is the logistic sigmoid function, f_t, i_t, o_t represent the forget, input and output gates respectively , x_t denotes the input, h_t represents the hidden state at time t. The forget, input and output gates control the flow of information throughout the sequence. W, U are matrices which represent the weights and b are the biases associated with the connections.

We employ stacked bi-directional LSTM units to extract textual features. The final hidden state of LSTM is obtained by concatenating the forward and backward states. Finally, we pass the LSTM output through a fully connected layer (enc text fc) to get the textual features.

$$R_T = \phi\left(W_{tf}R_{lstm}\right) \tag{4}$$

where, R_{lstm} is the output of the LSTM, W_{tf} is the weight matrix of the fully connected layer and ϕ is the activation function used.

3.2.2 Visual Encoder. The input to the visual encoder is the image attached with the post, V. As seen in various visual understanding problems, image descriptors trained using convolutional neural networks (CNNs) over large amounts of data have proven to be very effective. The implicit learning of spatial layout and object semantics in the later layers of the network has contributed to the success of these features.

We employ the pre-trained network of VGG-19 architecture [23] trained over the ImageNet database and use the output of the fully-connected layer (FC7). During the joint training process, we freeze the parameters of the VGG network to avoid parameter explosion. Finally, we pass the VGG output through multiple fully connected layers (enc vis fc*) to get the same sized representation of the image as that of text.

$$R_V = \phi\left(W_{vf}R_{vgg}\right) \tag{5}$$

where, R_{vgg} is the feature representation obtained from the VGG-19, W_{vf} is the weight matrix of the fully connected layer and ϕ is the activation function used.

The textual feature representation R_T and the visual feature representation R_V are then concatenated and passed through a fully connected layer to form the shared representation. We then obtain two vectors μ and σ from the shared representation. They can be treated as the mean and variance respectively of the distribution of the shared representation. Furthermore, a random variable ϵ is sampled from a previous distribution (e.g., Gaussian distribution). The final reparameterized multimodal representation denoted by R_m can be calculated as follows:

$$R_m = \mu + \sigma \circ \epsilon \tag{6}$$

We denote the encoder as $G_{enc}(M,\ \theta_{enc})$, where θ_{enc} denotes all the parameters to be learned in encoder, and M represents the set of multimedia posts. Hence, the output of the encoder for a multimedia post, m, is the multimodal representation:

$$R_m = G_{enc}\left(m,\ \theta_{enc}\right) \tag{7}$$

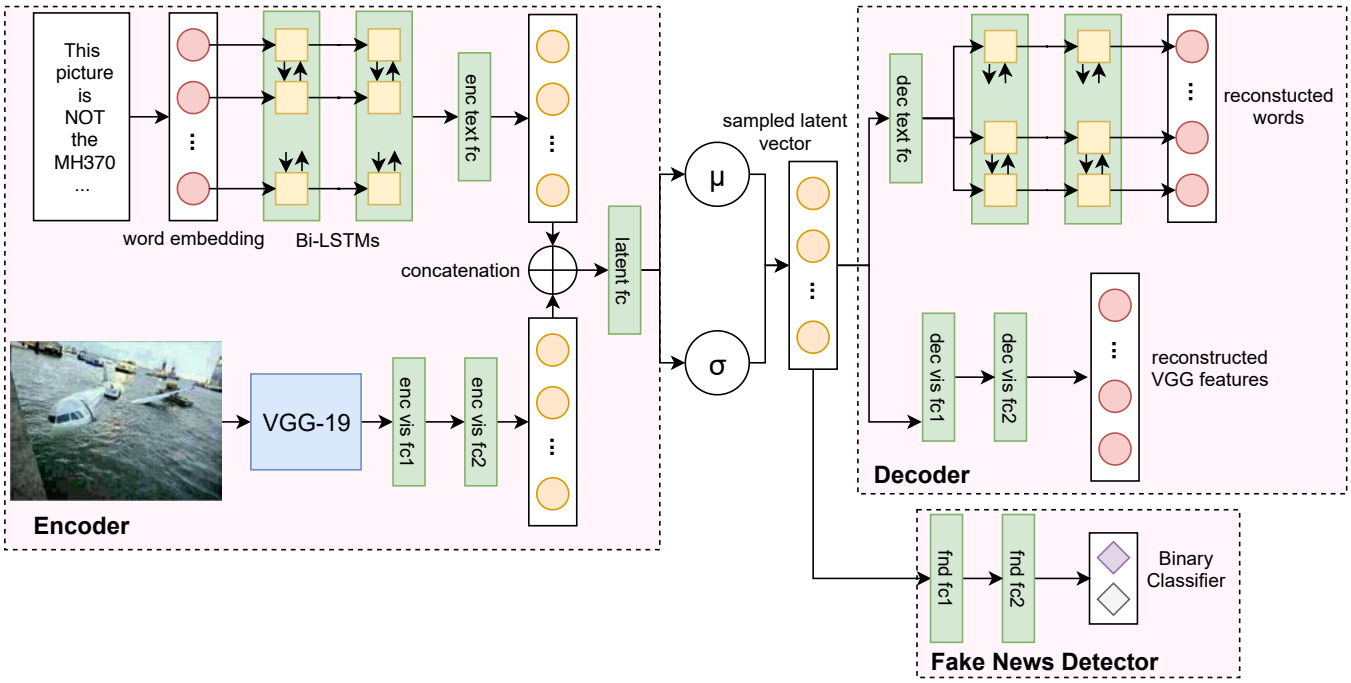


Figure 2: Network architecture of the proposed Multimodal Variational Autoencoder (MVAE). It has three components: Encoder, Decoder and Fake News Detector.

3.3 Decoder

The architecture of the decoder is similar to that of the encoder but inverted. The goal of the decoder is to reconstruct data from the sampled multimodal representation. Just like the encoder, it can be broken down into two sub-components: textual decoder and visual decoder.

3.3.1 Textual Decoder. The textual decoder takes as input the multimodal representation and reconstructs the words in the text. The multimodal representation is passed through a fully connected layer (dec text fc) to create inputs for the bi-directional LSTM. Then, we have similar stacked bi-directional LSTM units as used in the textual encoder sub-network. Finally, we pass the LSTM outputs through a time distributed fully connected layer with softmax activation to get the probability of each word in that time step.

3.3.2 Visual Decoder. The goal of the visual decoder is to reconstruct the VGG-19 features from the multimodal representation. The visual decoder sub-network is just the reverse of the visual encoder's corresponding sub-network. The multimodal representation is passed through multiple fully connected layers (dec vis fc*) to reconstruct the VGG-19 features.

We denote the decoder as $G_{dec}(R_m,\ \theta_{dec})$, where θ_{dec} denotes all the parameters in the decoder. Hence, the output of the decoder for a multimedia post, m, is a matrix of probability of every word for each position in the text, $\hat{t}_m$, and the reconstructed VGG-19 features of the image, $\hat{r}_{vgg_m}$.

$$\left(\hat{t}_m, \hat{r}_{vgg_m}\right) = G_{dec}\left(R_m,\ \theta_{dec}\right) \tag{8}$$

VAE models are trained by optimizing the sum of the reconstruction loss and the KL divergence loss. Therefore, we employ categorical cross-entropy loss for reconstruction of text and mean squared error for reconstruction of image features. The KL divergence between two probability distributions simply measures how much they diverge from each other. Minimizing the KL divergence here means optimizing the probability distribution parameters (μ and σ) to closely resemble that of the target distribution (Normal distribution). These can be calculated as follows:

$$\mathcal{L}_{rec_{vgg}} = \mathbb{E}_{m\sim M}\Big[\frac{1}{n_v}\sum_{i=1}^{n_v}\left(\hat{r}_{vgg_m}^{(i)} - r_{vgg_m}^{(i)}\right)^2\Big] \tag{9}$$

$$\mathcal{L}_{rec_t} = -\mathbb{E}_{m\sim M}\Big[\sum_{i=1}^{n_t}\sum_{c=1}^{C}\mathbf{1}_{c=t_m^{(i)}}\log\hat{t}_m^{(i)}\Big] \tag{10}$$

$$\mathcal{L}_{kl} = \frac{1}{2}\sum_{i=1}^{n_m}\left(\mu_i^2 + \sigma_i^2 - \log(\sigma_i) - 1\right) \tag{11}$$

where, M is the set of multimedia posts, n_v is the dimensionality of VGG-19 features, n_t is the number of words in text, n_m is the dimensionality of multimodal features, and C is the vocabulary size. We minimize the VAE loss by seeking optimal parameters $\hat{\theta}_{enc}$ and $\hat{\theta}_{dec}$ and this can be represented as follows.

$$\left(\theta_{enc}^*, \theta_{dec}^*\right) = \underset{\theta_{enc},\theta_{dec}}{\operatorname{argmin}}\left(\mathcal{L}_{rec_{vgg}} + \mathcal{L}_{rec_t} + \mathcal{L}_{kl}\right) \tag{12}$$

3.4 Fake News Detector

Fake News Detector takes multimodal representation as input and aims to classify the post as fake or not. It consists of multiple fully connected layers with corresponding activation functions. We

denote the fake news detector as $G_{fnd}(R_m,\ \theta_{fnd})$, where θ_{fnd} denotes all the parameters in the fake news detector. The output of the fake news detector for a multimedia post, m, is the probability of this post being a fake news.

$$\hat{y}_m = G_{fnd}\left(R_m,\ \theta_{fnd}\right) \tag{13}$$

We can view the value of $\hat{y}_m$ as a label 1 meaning the multimedia post m is fake, and 0 otherwise. In order to constrain the values between 0 and 1, we use the sigmoid logistic function. Hence, to calculate the classification loss, we employ cross-entropy as follows.

$$\mathcal{L}_{fnd}\left(\theta_{enc}, \theta_{fnd}\right) = -\mathbb{E}_{(m,y)\sim(M,Y)}\Big[y \log\left(\hat{y}_m\right) + \left(1-y\right)\log(1-\hat{y}_m)\Big] \tag{14}$$

where, M represents the set of multimedia posts and Y represents the set of ground truth labels. We minimize the classification loss by seeking the optimal parameters $\hat{\theta}_{fnd}$ and $\hat{\theta}_{enc}$, and this can be represented as follows.

$$\left(\theta^*_{enc}, \theta^*_{fnd}\right) = \underset{\theta_{enc}, \theta_{fnd}}{\operatorname{argmin}}\ \mathcal{L}_{fnd} \tag{15}$$

3.5 Putting it all together

The complete architecture of the proposed MVAE model is shown in Figure 2. The output of the encoder is fed to the decoder as well as the fake news detector. The decoder aims to reconstruct the data, while the fake news detector aims to classify the post as fake news or not. We jointly train the VAE and the fake news detector. Hence, the final loss can be written as follows.

$$\mathcal{L}_{final}\left(\theta_{enc}, \theta_{dec}, \theta_{fnd}\right) = \lambda_v \mathcal{L}_{rec_{vgg}} + \lambda_t \mathcal{L}_{rec_t} + \lambda_k \mathcal{L}_{kl} + \lambda_f \mathcal{L}_{fnd} \tag{16}$$

where, λs can be used to balance the individual terms of the loss function. In this paper, we simply set all the λs as 1. The optimal parameters can then be calculated by minimizing the final loss as follows.

$$\left(\theta^*_{enc}, \theta^*_{dec}, \theta^*_{fnd}\right) = \underset{\theta_{enc}, \theta_{dec}, \theta_{fnd}}{\operatorname{argmin}}\ \mathcal{L}_{final}\left(\theta_{enc}, \theta_{dec}, \theta_{fnd}\right) \tag{17}$$

4 EXPERIMENTS

In this section, we first describe the two social media datasets used in the experiments. We then discuss in brief the state-of-the-art fake news detection approaches along with some state-of-the-art language-vision models.

4.1 Datasets

Given the sparse availability of structured multimedia data, we make use of two standard datasets to evaluate our architecture for fake news detection. The two datasets consist of real social media information collected from Twitter and Weibo. To the best of our knowledge, these are the only available datasets that have paired image and textual information.

4.1.1 Twitter Dataset. As part of MediaEval [3], the Twitter dataset was released for Verifying Multimedia Use task. The task was aimed at detecting fake multimedia content on social media. The dataset consists of tweets (short messages posted on Twitter) and each tweet has textual content, image/video and social context information associated with it. The dataset has around 17000 unique tweets spanning over different events. The dataset is split into two parts: the development set (9000 fake news tweets, 6000 real news tweets) and the test set (2000 tweets). They are split in such a manner that the tweets have no overlapping events. Given our focus on the image and textual information, we filter out all tweets which have videos attached to them in the set. We use the development set for training and the test set for testing to keep the same data split scheme as the benchmark.

Table 1: Performance of MVAE vs other methods on two different datasets

Dataset	Method	Accuracy	Fake News			Real News		
			Precision	Recall	F_1	Precision	Recall	F_1
Twitter	Textual	0.526	0.586	0.553	0.569	0.469	0.526	0.496
	Visual	0.596	0.695	0.518	0.593	0.524	0.7	0.599
	VQA	0.631	0.765	0.509	0.611	0.55	0.794	0.65
	Neural Talk	0.610	0.728	0.504	0.595	0.534	0.752	0.625
	att-RNN	0.664	0.749	0.615	0.676	0.589	0.728	0.651
	EANN	0.648	**0.810**	0.498	0.617	0.584	0.759	0.660
	MVAE	**0.745**	0.801	**0.719**	**0.758**	**0.689**	**0.777**	**0.730**
Weibo	Textual	0.643	0.662	0.578	0.617	0.609	0.685	0.647
	Visual	0.608	0.610	0.605	0.607	0.607	0.611	0.609
	VQA	0.736	0.797	0.634	0.706	0.695	0.838	0.760
	Neural Talk	0.726	0.794	0.713	0.692	0.684	0.840	0.754
	att-RNN	0.772	0.854	0.656	0.742	0.72	**0.889**	0.795
	EANN	0.782	0.827	0.697	0.756	0.752	0.863	0.804
	MVAE	**0.824**	**0.854**	**0.769**	**0.809**	**0.802**	0.875	**0.837**

4.1.2 Weibo Dataset. The Weibo dataset, used in [8] for the task of fake news detection consists of data collected from Xinhua News Agency, an authoritative news source of China and Weibo, a Chinese microblogging website. The fake news collected from Weibo is obtained over a span of time ranging from May 2012 to January 2016 and is verified by Weibo's official rumour debunking system. The system encourages common users to report suspicious tweets on Weibo which are then examined by a committee of reputable users that classify the suspicious tweet as false or real. In accordance with previous work [14, 27], this system also acts as the authoritative source for collecting rumour news. The non-rumour tweets are tweets verified by the Xinhua News Agency. Preprocessing of the dataset is done using a method similar to that of [8]. Preliminary steps involve removal of duplicate images (using locality sensitive hashing) and low-quality images to ensure homogeneity across the dataset. The dataset is then split into training and testing sets with an approximate tweet ratio of 4:1 as in Jin et al. [8].

4.2 Experimental Settings

For word embeddings, we employ the distributed Word2Vec representation for words [16]. For the Twitter dataset, we have posts which are not in English, so we translated them into English to keep the data coherent. We follow standard text pre-processing for the Twitter dataset. For the Weibo dataset, text is in Chinese. Chinese text is written without spaces between words. We use the Stanford Word Segmenter for Chinese text tokenization into words.

We pre-train the Word2Vec model on the training dataset in an unsupervised fashion, with dimension size of 32.

For visual features, we use the output of the second to the last layer of a 19-layer VGGNet pre-trained on ImageNet set [23]. The dimensions of features obtained from VGG-19 is 4096. We do not fine tune the weights of the VGG, i.e., the weights of the VGG network are frozen.

The textual encoder consists of LSTMs with dimension size of hidden layers as 32. The fully connected layer used in the textual encoder is of size 32. The visual encoder consists of two fully connected layers of size 1024 and 32. The decoder consists of layers with the same dimensions as that of the encoder. The fake news detector has two fully connected layers of size 64 and 32.

We use a batch size of 128 instances in the training of the whole network. The model is trained for 300 epochs with a learning rate of 10^{-5} with an early stopping to report the results. We use the hyperbolic tangent function as the non-linear activation function. To prevent overfitting, we use L2-regularizer on the weights of our model. We experimented with a weight penalty of [0, 0.05, 0.1, 0.3, 0.5] and set it to 0.05 for the encoder and the decoder, and 0.3 for the fake news detector. To seek optimal parameters for our model, we use Adam [11] as the optimizer. We make the code publicly available[1].

4.3 Baselines

To validate the performance of the proposed multimodal variational autoencoder, we compare it with two categories of baseline models: single modality models and multimodal models.

4.3.1 Single Modality Models. MVAE leverages information from both visual and textual data to identify potentially fake news. As against such multimodal approaches, we also experimented with two unimodal models as described below.

- **Textual**: This model uses only textual information present in posts to classify them as fake or not. Each word is represented as a 32-dimensional vector. The word-embeddings are trained on the text content of the posts. Individual posts are then fed into a Bi-LSTM to extract textual features F_T. F_T is then fed into a 32-dimensional fully connected layer with a softmax function coupled with the Bi-LSTM that is responsible for making final predictions.
- **Visual**: The visual model uses only images from posts to classify them as fake or not. Images are fed into a pre-trained VGG-19 network with a fully-connected layer to extract visual features F_V. Similar to the textual model, the visual features F_V are then fed into a 32-dimensional fully connected layer for making the prediction.

4.3.2 Multimodal Models. Multimodal approaches utilize information from multiple modalities for the task of fake news classification.

- **VQA** [2]: Visual Question Answering aims to answer questions about given images. We adapted the Visual QA model which was originally designed for a multi-class classification task to our binary classification task. This is done by replacing the final multi-class layer with a binary-class layer. We use a one-layer LSTM with number of hidden units set to 32.
- **Neural Talk** [25]: The work of Vinyals et al. [25] in the domain of image captioning, proposes generation of natural language sentences describing an image using a deep recurrent framework. Following a structure similar to theirs, we obtain latent representations by averaging the output of the RNN at each time step as the joint representation of the image and text in tweets. These representations are then fed into a fully connected layer followed by an entropy loss layer to make predictions.
- **att-RNN** [8]: att-RNN uses attention mechanisms to combine textual, visual and social context features. In this end-to-end network, image features are incorporated into the joint representation of text and social context, obtained using an LSTM network. The neural attention from the outputs of the LSTM is an integral part for fusing the visual features. For a fair comparison, in our experiments, we remove the part dealing with social context information.
- **EANN** [26]: The Event Adversarial Neural Network (EANN) consists of three main components: the multimodal feature extractor, the fake news detector and the event discriminator. The multimodal feature extractor extracts textual and visual features from posts. It works with the fake news detector to learn the discriminative representation for detection of fake news. The event discriminator is responsible for removing any event-specific features. It is also possible to detect fake news using only two components, the multimodal feature extractor and the fake news detector. Hence, for a fair comparison, in our experiments, we work with a variant of EANN which does not include the event discriminator.

Note that fair comparisons with methods proposed in [13, 15] are not possible since they use addition information like tweet propagation data. Also, these methods are not developed for multimodal datasets. We report the accuracy, recall, precision, and F_1 scores of all the baselines along with our proposed model in Table 1.

5 RESULTS AND ANALYSIS

Table 1 shows the results of the baselines as well as our proposed method on both the datasets. We report the accuracy of our fake news detector as well as the precision, recall and F_1 score of our method for both fake news and real news. We can clearly see that our proposed method performs much better than the baselines.

On the Twitter dataset, among the single modality models, the visual model performs better than the textual model. This can be attributed to the fact that image features learnt with the help of VGG-19 have more shareable patterns to classify news as compared to textual features. Although visual features perform better than textual features, single modality models perform worse than the multimodal models.

Among multimodal models, att-RNN beats EANN which tells us that the attention mechanism can help in improving the performance of the model by considering the parts of the image which are related to the text. Our proposed model MVAE outperforms the baseline models by a huge margin and increases the accuracy from 66.4% to 74.5% and increases the F_1 scores from 66% to 73%.

On the Weibo dataset, we see similar trends in the results. The textual model beats the visual model in single modality models.

[1] https://github.com/dhruvkhattar/MVAE

Among multimodal methods, EANN and att-RNN which were proposed for this task perform better than Neural Talk and VQA. MVAE outperforms all the baselines and boosts the performance from 78.2% to 82.4% in terms of accuracy and shows an increase of ∼5% in F_1 scores, compared to the previous best baseline. This validates the effectiveness of our proposed method MVAE in detecting fake news on social media.

6 CONCLUSIONS

In this work, we explored the task of multimodal fake news detection. Overcoming the limitations of the current models, we tackle the challenge of learning correlations between modalities in tweets and to do so, propose a multimodal variational autoencoder that learns shared (visual + textual) representations to aid fake news detection. Our model consists of three main components, an encoder, a decoder and a fake news detector. Our proposed model, MVAE is trained by jointly learning the encoder, decoder and the fake news detector. The performance of our proposed architecture is evaluated on two real-world datasets. The MVAE model outperforms the current state-of-the-art architectures. In the future, we plan to extend MVAE using tweet propagation data and user characteristics.

REFERENCES

[1] Hunt Allcott and Matthew Gentzkow. 2017. Social Media and Fake News in the 2016 Election. 31 (05 2017), 211–236.

[2] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*. 2425–2433.

[3] Christina Boididou, Symeon Papadopoulos, Duc-Tien Dang-Nguyen, Giulia Boato, Michael Riegler, Stuart E Middleton, Andreas Petlund, and Yiannis Kompatsiaris. [n. d.]. Verifying Multimedia Use at MediaEval 2016. ([n. d.]).

[4] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*. ACM, 675–684.

[5] Tong Chen, Lin Wu, Xue Li, Jun Zhang, Hongzhi Yin, and Yang Wang. 2017. Call Attention to Rumors: Deep Attention Based Recurrent Neural Networks for Early Rumor Detection. *arXiv preprint arXiv:1704.05973* (2017).

[6] Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. Association for Computational Linguistics, 171–175.

[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[8] Zhiwei Jin, Juan Cao, Han Guo, Yongdong Zhang, and Jiebo Luo. 2017. Multimodal fusion with recurrent neural networks for rumor detection on microblogs. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 795–816.

[9] Zhiwei Jin, Juan Cao, Yongdong Zhang, Jianshe Zhou, and Qi Tian. 2017. Novel visual and statistical image features for microblogs news verification. *IEEE transactions on multimedia* 19, 3 (2017), 598–608.

[10] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3128–3137.

[11] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[12] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[13] Yang Liu and Yi fang Brook Wu. 2018. Early Detection of Fake News on Social Media Through Propagation Path Classification with Recurrent and Convolutional Networks. In *AAAI*.

[14] Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting Rumors from Microblogs with Recurrent Neural Networks.. In *IJCAI*. 3818–3824.

[15] Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor detection on twitter with tree-structured recursive neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1980–1989.

[16] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[17] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*. 1310–1318.

[18] Dong ping Tian et al. 2013. A review on image feature extraction and representation techniques. *International Journal of Multimedia and Ubiquitous Engineering* 8, 4 (2013), 385–396.

[19] Kashyap Popat, Subhabrata Mukherjee, Jannik Strötgen, and Gerhard Weikum. 2016. Credibility assessment of textual claims on the web. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 2173–2178.

[20] Victoria Rubin, Niall Conroy, Yimin Chen, and Sarah Cornwell. 2016. Fake news or truth? using satirical cues to detect potentially misleading news. In *Proceedings of the Second Workshop on Computational Approaches to Deception Detection*. 7–17.

[21] Natali Ruchansky, Sungyong Seo, and Yan Liu. 2017. Csi: A hybrid deep model for fake news detection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 797–806.

[22] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter* 19, 1 (2017), 22–36.

[23] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[24] Tetsuro Takahashi and Nobuyuki Igata. 2012. Rumor detection on twitter. In *Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS), 2012 Joint 6th International Conference on*. IEEE, 452–457.

[25] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3156–3164.

[26] Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. EANN: Event Adversarial Neural Networks for Multi-Modal Fake News Detection. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 849–857.

[27] Ke Wu, Song Yang, and Kenny Q Zhu. 2015. False rumors detection on sina weibo by propagation structures. In *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*. IEEE, 651–662.

[28] Liang Wu and Huan Liu. 2018. Tracing fake-news footprints: Characterizing social media messages by how they propagate. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 637–645.

[29] Yin Zhu, Xiao Wang, Erheng Zhong, Nathan Nan Liu, He Li, and Qiang Yang. 2012. Discovering Spammers in Social Networks.. In *AAAI*.

Mobile App Risk Ranking via Exclusive Sparse Coding*

Deguang Kong
Yahoo Research
701 First Avenue, Sunnyvale, CA
doogkong@gmail.com

Lei Cen
Twitter Inc
1355 Market St #900, San Francisco, CA
lcen@twitter.com

ABSTRACT

To improve mobile application (App for short) user experience, it is very important to inform the users about the apps' privacy risk levels. To address the challenge of incorporating the heterogeneous feature indicators (such as app permissions, user review, developers' description and ads library) into the risk ranking model, we formalize the app risk ranking problem as an exclusive sparse coding optimization problem by taking advantage of features from different modalities via the maximization of the feature consistency and enhancement of feature diversity. We propose an efficient iterative re-weighted method to solve the resultant optimization problem, the convergence of which can be rigorously proved. The extensive experiments demonstrate the consistent performance improvement using the real-world mobile application datasets (totally 13786 apps, 37966 descriptions, 10557681 user reviews and 200 ad libraries).

CCS CONCEPTS

• **Networks** → **Mobile and wireless security**; • **Computing methodologies** → **Supervised learning by classification**.

KEYWORDS

App; Mobile; Security; LASSO

ACM Reference Format:
Deguang Kong and Lei Cen. 2019. Mobile App Risk Ranking via Exclusive Sparse Coding. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313589

1 INTRODUCTION

Improving user experience is an important task to increase app user engagement and monetize mobile apps. Risk management and assessment has been proposed to improving the user experiences from analysis and diagnosis of potential hazards and risk factors that may cause harm to the mobile app users, such as release of location, user preference, contact and even other personal bio-metric information. A privacy risk score quantifies the risk levels of mobile apps, which allows the users to better perceive the risk levels, which has been shown to have a "significant positive effects" [11] to improving user experience.

*Thanks for Purdue University and Samsung for the support and resources provided while the authors were doing this work. All authors contribute significantly to the project according to ACM policy.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313589



Figure 1: Mobile app risk ranking using heterogeneous privacy risk indicators, including user review, description, permission and ads. The demonstrated four apps, i.e., `Yahoo finance`, `Tango message`, `Angry bird`, and `Fruit Ninja` are categorized as `none`, `low`, `medium` and `high`, respectively.

To efficiently and effectively perceive the risk of apps is a challenging task, which generally requires the deep understanding of mobile app framework, and expert knowledge of app permission and advertisement libraries. Therefore, machine learning technique is naturally applied to learn the risk scores of apps. Given the heterogeneous risk features indicating different attributions of risks, how to combine them to give a comprehensive assessment by leveraging feature correlations, consistency and diversity, is under-explored but highly desirable. Unfortunately, recent works, including permission usage pattern mining [9] [7], app permission prediction from meta-data [28] [19], malicious app detection [15] [39], API link analysis [24], mobile app recommendation [44] [21], deep model compression on mobile device [20] [18], however, do not essentially solve this problem. To bridge this gap, in this paper, we propose a novel approach to rank app risks using heterogeneous features from multiple sources in a unified optimization framework, which considers both intra-view and inter-view feature information while encouraging their competitions. With our method, one can easily understand which feature play more important role for risk ranking and therefore the result is *interpretable* for illustrating the underlying reasons. Further, once the model is well-trained using the labeled app corpus, the model can be easily deployed and served to automatically label the new apps in the market with high accuracy. To summarize, the key contributions are listed as follows.

• To the best of our knowledge, this paper is the first work that explores heterogeneous privacy indicators for app risk ranking, which, we believe, is very important for improving user experience on mobile platforms.

• We formalize the risk ranking problem as a multi-view feature learning problem via exclusive sparse coding strategy, which can

Table 1: Mobile app risk category

Category	Examples
High	Drag Racing, Fruit Ninja, $\cdots$
Medium	Angry bird, $\cdots$
Low	Temple Run 2, Tango message, $\cdots$
Few/None	Wechat, Yahoo Finance, Instagram, $\cdots$

Table 2: Notations used in the paper

notation	description
$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$	$\mathbf{x}_i^v \in \Re^{p_v}$, v-th view of feature
$\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n]$	$y_i \in \Re^K$, risk category for app i
w_k^v	the weight for category k with the v-th view indicator

automatically select the most discriminant features by enforcing both inter-view and intra-view feature competitions.

• We derive a *novel* iteratively re-weighted algorithm to tackle the resultant optimization problem, which can actually handle *any* feature group structures, regardless of coherent or exclusive group feature patterns.

• We crawled real-world datasets (totally 13, 786 apps, 37, 966 descriptions, 10, 557, 681 user reviews and 200 ads libraries), and used them to comprehensively evaluate the effectiveness of our approach. Our method not only consistently outperforms the baseline methods, but also offers valuable insights to understand the risk features attributed to different sources.

Notations Let n be the number of data points, p_v be the dimension for data from view v, k be the number of class in dataset, V be the number of total views. For any vector $\mathbf{x} \in \Re^p$, ℓ_q norm of $\mathbf{w}$ is $\|\mathbf{w}\|_q = (\sum_{i=1}^p \|w_i^q\|)^{\frac{1}{q}}$ for $\forall q \in (0, 2)$.

2 PROBLEM STATEMENT

More formally, assume we have n mobile apps, each mobile app is abstracted as a data point $\mathbf{x}_i$ denoting the privacy indicators extracted from the app. In the paper next, we denote the features extracted from v–th privacy indicator ($1 \le v \le V$) as the v-th view feature. In mobile app context, let $\mathbf{x}_i^v \in \Re^{p_v}$ be v-th view feature (*i.e.*, features extracted from permission, user review, description, ads library, *etc*) of a mobile app i. Consider all the mobile apps, $\mathbf{X}^v = [\mathbf{x}_1^v, \mathbf{x}_2^v, \cdots, \mathbf{x}_n^v]$, where each data column vector is $\mathbf{x}_i^v \in \Re^{p_v \times 1}$. The simple concatenation of all views of apps gives:

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}^1 \\ \mathbf{X}^2 \\ \vdots \\ \mathbf{X}^V \end{bmatrix} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n], \quad \mathbf{x}_i = \begin{pmatrix} \mathbf{x}_i^1 \\ \mathbf{x}_i^2 \\ \vdots \\ \mathbf{x}_i^V \end{pmatrix}.$$

with dimension $p = \sum_v p_v$.

Given the task of risk ranking for mobile apps, the risk of mobile app is categorized into four classes[1]:

`High > Medium > Low > None`,

where the risk levels rank from high to none.

Then each mobile is assigned to a single category. For example, *Yahoo Finance* belongs to *none* risk category. Table 1 gives examples of risk levels for different mobile apps. More formally, let $\mathbf{y}_i$ be risk class label for each $\mathbf{x}_i$ then $\mathbf{y}_i \in \Re^K$ where K is number of categories (K=4 in our case). Written in a matrix format, $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n]$, where $\mathbf{Y} \in \Re^{K \times n}$. The goal of multi-view feature learning task is to learn the projection matrix $\mathbf{W} \in \Re^{p \times K}$ such that the risk levels can be projected into features from different sources, i.e.,

$$\mathbf{W} = [\mathbf{w}_1 \cdots \mathbf{w}_K], \quad \mathbf{w}_k = [\mathbf{w}_k^1, \mathbf{w}_k^2, \cdots, \mathbf{w}_k^V]^T, \tag{1}$$

where $\mathbf{w}_k^v$ has the same size as data $\mathbf{x}_i^v$ from view v. Table 2 summarizes the notations used throughout the paper.

Table 3: Privacy risk issues

Privacy risk issue	Behavior Descriptions
Spamming	Advertisement in notification bar, ads via email, ads via SMS, Pop-up ads, Fishing, *etc*
Financial	Paid for In-App-Purchase (IAP), but do not get the item, free to premium, *etc*
Over-claimed Permission	Request too much permissions than users' expectations
Data leakage	Access privacy data without user acknowledgement, *e.g.*, user account, contact, location, *etc*

An illustration In the proposed risk ranking framework, after learning the projection $\mathbf{W}$, we can automatically label the risk category for the new apps following the same feature extraction strategy. For each app, the extracted features $\mathbf{x}_i^v$ are from four perspectives including: (i) permission features with one-hot coding denoting the critical resources (e.g., location, contact) that an app allows to access, (ii) developer description features via tf-idf representation reflecting the developers' awareness [30], (iii) user review features using annotated semantics that provide valuable feedback of users by considering users' expectations (shown in Table 3), (iv) advertisement library features via one-hot coding relating to monetizing purpose [8] with users' personal data.

3 HETEROGENEOUS RISK LEARNING

In this section, we propose a new method for risk ranking using the heterogeneous features. The key idea of our approach to leverage the rich information from different views of features that indicate different aspects of risk of apps while considering both inter-view feature competitions and intra-view feature correlations, which is guided by the following key observations.

(i) For each app risk level, the discriminant features for decision could be different. For example, the dominant features that are used to determine *high* and *none* categories might be "location" and "game", respectively.

(ii) Given the app risk level, some views of features are more important than other views. The competition among the features from different view exists, *i.e.*, *not* all features from each view are needed for risk ranking. For example, in terms of category *low* and *medium*, *advertisement* features are generally more important than other features because mobile apps usually access or even send out users' data stealthily whereas "*permission*" features may be more important for differentiating *high* from *medium* category.

(iii) For the features from the same view, there exist feature correlations and redundancies. From information gain perspective, *less is more*. Thus Feature selection is necessary to select the discriminant features in each view, so as to achieve the minimum

[1]This category is based on expert suggestions from Android developers. `privacygrade: https://www.hcii.cmu.edu/research/privacygrade` from CMU adopts the same category.

redundancy while maintaining maximum relevance at intra-view level.

Based on the above observations, in order to find the projection matrix $\mathbf{W} \in \Re^{p\times K}$ that can map the input features into the risk category, we propose to solve the following objective function:

$$\min_{\mathbf{W}} f(\mathbf{W};\mathbf{X},\mathbf{Y}) + \alpha\|\mathbf{W}\|_{\mathrm{G}} + \beta\|\mathbf{W}\|_{EG} \tag{2}$$

where $f(\mathbf{W};\mathbf{X},\mathbf{Y})$ is a loss function given multi-view data matrix $\mathbf{X} \in \Re^{p\times n}$ and class labels $\mathbf{Y} \in \Re^{K\times n}$, which is used to minimize the distance between the observations and prediction. Any loss function can be applied here, e.g., cross-entropy loss, logistic loss, hinge loss, etc. For example, with cross-entropy loss, the objective function is:

$$f(\mathbf{W};\mathbf{X},\mathbf{Y}) = \sum_{ik} -Y_{ik} log\widehat{Y_{ik}}, \tag{3}$$

$$\widehat{Y_{ik}} = \frac{1}{1+\exp^{-\mathbf{w}_k^T\phi(\mathbf{x}_i,\theta)}}, \tag{4}$$

where $\phi(\mathbf{x}_i,\theta)$ represents features after projection, either can be in the original feature space, i.e.,

$$\phi(\mathbf{x}_i,\theta) = \mathbf{x}_i, \tag{5}$$

or in projected space after multiple layer non-linear projection:

$$\phi(\mathbf{x}_i,[\theta,\varphi]) = \varphi_L\Big(\theta_L^t[...[\varphi_1(\theta_1^t\mathbf{x}_i)]]\Big), \tag{6}$$

where L is the number of layers, θ_ℓ, φ_ℓ is the feature projection and activation function (such as ReLu, tanh function) for layer ℓ.

The regularization terms are used to incorporate both *local intra-view* via $\|\mathbf{W}\|_{EG}$ and *global inter-view* information via $\|\mathbf{W}\|_{\mathrm{G}}$ given the projection $\mathbf{W} = [\mathbf{w}_1\mathbf{w}_2\cdots\mathbf{w}_K]$, i.e.,

$$\|\mathbf{W}\|_{\mathrm{G}} = \sum_{k=1}^{K}\sum_{v=1}^{V}\|\mathbf{w}_k^v\|_2, \quad \|\mathbf{W}\|_{EG} = \sum_{k=1}^{K}\sum_{v=1}^{V}\|\mathbf{w}_k^v\|_1^2. \tag{7}$$

Essentially, we explore both *local intra-view* and *global inter-view* features for multi-view feature selection purpose.

Mathematical Explanation In optimization of $\mathbf{W}$, we optimize each $\mathbf{w}_k(1 \le k \le K)$ separately, because it is corresponding to the feature coefficient computed from class k, independent of $\mathbf{w}_\ell$ from class ℓ, $\forall k \ne \ell$. This indicates that the selected features are unique to each category (as evident in observation i).

(i) $\|\mathbf{W}\|_{\mathrm{G}}$ is used to enforce feature competitions at inter-view level. As a result, features from several views are selected. This term has the same structure as that of group lasso [41] or $\ell_{2,1}$ norm [22] but is applied for multi-view setting. The "group" concept is naturally extended for each view, instead of traditional feature selection ([1, 26]) (as evident in observation ii).

(ii) $\|\mathbf{W}\|_{EG}$ is used to enforce feature competitions at *intra-view* level, because generally features from the same views, could be highly correlated. Thus, as evident in observation (iii), we let features in the same view compete for survival, *i.e.*, only a few number of features in each view could survive. This is achieved by by applying LASSO exclusively on features in each view.

4 OPTIMIZATION ALGORITHM

Fortunately, all terms of Eq.(2) are convex functions, and thus we can obtain a global optimal solution. It is generally felt that joint $\ell_{2,1}$ and $\ell_{1,2}$ terms are much more difficult to solve than the lasso term via shrinkage thresholding. Existing algorithm can formulate them as a quadratic programming problem [27], which can be solved by interior point method or active set method. However, the computational cost is expensive, which limits its use in practice. Recently, a primal-dual algorithm [38] is proposed to solve the similar problem, which casts the non-smooth problem into a min-max problem. However, the algorithm is a gradient descent type method and converges slowly. Moreover, the algorithm is designed for multi-task learning problem, and cannot be applied directly for both group and exclusive lasso problem.

4.1 Proximal Gradient Optimization

Proximal gradient method (a.k.a FISTA method) [25] is widely used to solve $\min_{\mathbf{W}} f(\mathbf{W}) + \Omega(\mathbf{W})$ problem. Here we adopt this method to solve Eq.(2). The key idea of proximal gradient method is to optimize $\mathbf{W}^t$ at each iteration t, *i.e.,*

$$\begin{aligned}\mathbf{W}^{t+1} &= \arg\min_{\mathbf{U}} \quad f(\mathbf{W}^t) + \nabla f(\mathbf{W}^t)^T(\mathbf{U}-\mathbf{W}^t) \\ &\qquad +\frac{L}{2}\|\mathbf{U}-\mathbf{W}^t\|_F^2 + \Omega(\mathbf{U}) \\ &= \arg\min_{\mathbf{U}} \quad \frac{1}{2}\|\mathbf{U}-\mathbf{A}\|_F^2 + \frac{1}{L}\Omega(\mathbf{U}),\end{aligned} \tag{8}$$

where $\mathbf{A} = \mathbf{W}^t - \frac{1}{L}\nabla f(\mathbf{W}^t)$, and L is a parameter chosen at each iteration using certain searching strategy, $\Omega(\mathbf{U})$ is the regularization term and $\Omega(\mathbf{U}) = \alpha\|\mathbf{U}\|_{\mathrm{G}} + \beta\|\mathbf{U}\|_{EG}$. Thus the problem is transformed to the minimization problem of Eq.(8).

Their major computation efforts in each iteration is to compute the gradient of $\nabla f(\mathbf{W}^t)$ depending on $f(\mathbf{W}^t)$. For example, for least square loss, it can be computed within $O(pnK)$. For logistic type loss $g(\mathbf{W})$,

$$g(\mathbf{W})' = g(\mathbf{W})(1-g(\mathbf{W})). \tag{9}$$

With appropriate choose of step size L, the proximal gradient method will achieve ϵ-optimal solution in $O(\frac{1}{\sqrt{\epsilon}})$ iterations. Then the key step is to solve the associated proximal operator:

$$\min_{\mathbf{U}} \quad \frac{1}{2}\|\mathbf{U}-\mathbf{A}\|_F^2 + \frac{1}{L}\Omega(\mathbf{U}). \tag{10}$$

4.2 Iteratively re-weighted method

We present an iteratively re-weighted algorithm to solve Eq.(10). Let $\mathbf{u}_k$ be k-th column of $\mathbf{U}$, the same for $\mathbf{a}_k$. Notice that

$$\begin{aligned}&\frac{1}{2}\|\mathbf{U}-\mathbf{A}\|_F^2 + \frac{1}{L}\Omega(\mathbf{U}) \\ &= \sum_k\Big[\frac{1}{2}\|\mathbf{u}_k-\mathbf{a}_k\|_2^2 + \sum_{v=1}^{V}(\alpha'\|\mathbf{u}_k^v\|_2 + \beta'\|\mathbf{u}_k^v\|_1^2)\Big],\end{aligned} \tag{11}$$

where $\alpha' = \frac{\alpha}{L}, \beta' = \frac{\beta}{L}$. Then we only need to minimize $J_1(\mathbf{u})$ w.r.t to $\mathbf{u} \in \Re^{p\times 1}$, *i.e.,*

$$J_1(\mathbf{u}) = \frac{1}{2}\|\mathbf{u}-\mathbf{a}\|^2 + \sum_{v=1}^{V}(\alpha'\|\mathbf{u}^v\|_2 + \beta'\|\mathbf{u}^v\|_1^2). \tag{12}$$

To the best of our knowledge, this is the *first* work which can tackle both $\ell_{2,1}$ and $\ell_{1,2}$ terms simultaneously. The idea is to find an auxiliary function for Eq.(12) which can be easily solved. Then the updating rules for $\mathbf{u}$ is derived. Finally, we prove the solution is

exactly the optimal solution we are seeking for the original problem. Since Eq.(12) is convex, the optimal solution is exact the global optimal solution. The algorithm is named as *iteratively reweighted method*, because it updates $\mathbf{u}$ iteratively, until the algorithm converges. Moreover, our algorithm can be easily extended to handle arbitrary feature structures, such as linear structure [40], tree structure [23], graph structure [16], forest structure [6], etc

4.2.1 Algorithm. Instead of directly optimizing Eq.(12), we propose to optimize the following objective (the reasons will be seen immediately below), i.e.,

$$J_2(\mathbf{u}) = \frac{1}{2}||\mathbf{u}-\mathbf{a}||^2 + \frac{1}{2}\alpha'\mathbf{u}^T\mathbf{E}\mathbf{u} + \frac{1}{2}\beta'\mathbf{u}^T\mathbf{F}\mathbf{u}, \tag{13}$$

where $\mathbf{E}, \mathbf{F} \in \Re^{p\times p}$ are both diagonal matrices, and given by

$$\mathbf{E}_{ii} = \frac{1}{\sqrt{\sum_{j\in v(i)}\mathbf{u}_j^2}}, \quad \mathbf{F}_{ii} = \frac{2\sum_{j\in v(i)}|\mathbf{u}_j|}{|\mathbf{u}_i|}, \tag{14}$$

where $v(i)$ denotes the view where feature i belongs to. Note the computation of $\mathbf{E}, \mathbf{F}$ depends on $\mathbf{u}$, thus minimization of $\mathbf{u}$ depends on both $\mathbf{E}, \mathbf{F}$ respectively. In the following, we propose an iteratively re-weighted algorithm to find out the optimal global solution for $\mathbf{u}$, where in each iteration, $\mathbf{u}$ is updated using current $\mathbf{E}, \mathbf{F}$, and $\mathbf{E}, \mathbf{F}$ are updated using current $\mathbf{u}$. This process is iterated until the algorithm converges. Taking the derivative of Eq.(13) w.r.t $\mathbf{u}$ and set it to zero. We have

$$\frac{\partial J_2}{\partial \mathbf{u}} = \mathbf{u} - \mathbf{a} + \alpha'\mathbf{E}\mathbf{u} + \beta'\mathbf{F}\mathbf{u} = 0. \tag{15}$$

Thus we obtain the optimal solution for Eq.(13) is given by,

$$\mathbf{u} = (\mathbf{I} + \alpha'\mathbf{E} + \beta'\mathbf{F})^{-1}\mathbf{a}. \tag{16}$$

Note $\mathbf{E}, \mathbf{F}$ are all diagonal matrix, and thus the $(\mathbf{I} + \alpha'\mathbf{E} + \beta'\mathbf{F})^{-1}$ can be efficiently computed, where $\mathbf{I} \in \Re^{p\times p}$ is identity matrix. Let $\mathbf{e} \in \Re^p, \mathbf{f} \in \Re^p, e_i = E_{ii}, f_i = F_{ii}, \mathbf{Q} = (\mathbf{I} + \beta_1\mathbf{E} + \beta_2\mathbf{F})^{-1}$, then $\mathbf{Q}$ is a diagonal matrix and given by,

$$\mathbf{u} = \mathbf{Q}\mathbf{a}, \quad \mathbf{Q}_{ii} = \frac{1}{1 + \alpha' e_i + \beta' f_i}. \tag{17}$$

Using the updating rule of Eq.(17), we can obtain the global optimal solution for Eq.(12). To summarize, Algorithm 1 gives the whole updating process of each individual variable $\mathbf{u}$, $\mathbf{E}$, $\mathbf{F}$.

Algorithm 1 Iterative reweighted algorithm to solve Eq.(12)

Input: $\mathbf{a}$, α', β'
Output: $\mathbf{u}$
Procedure:
1: $t = 0$.
2: **while** Not converge **do**
3: Update $\mathbf{u}^t$ using Eq.(17).
4: Update $e_i^t = \frac{1}{\sqrt{\sum_{j\in v(i)}\mathbf{u}_j^2}}, f_i^t = \frac{2\sum_{j\in v(i)}|\mathbf{u}_j|}{|\mathbf{u}_i|}$ of Eq.(14)
5: $t = t + 1$.
6: **end while**

Time complexity analysis In practice, we find the above algorithm converges very fast. Typically, it converges to global optimal solution within 10-20 iterations (to precision 1e-6). Moreover, the running time of our algorithm is also very fast, because both $\mathbf{E}$ and $\mathbf{F}$ are diagonal matrices. In each iteration, we need to update $\mathbf{u}, \mathbf{e}, \mathbf{f}$ iteratively, which takes time cost $O(p), O(pV), O(pV)$, respectively, where V is the number of view. Thus the total time cost of each iteration is $O(pV)$, which is linear w.r.t the number of features and the number of groups. The convergence analysis is left in Appendix.

Table 4: Descriptions of several mobile datasets crawled in `Google play` at different periods, including user reviews and textual info. The ads and permission are encoded with one-hot coding.

Dataset	#app	#comment	# textual info
D1	6,234	5,028,359	18,157
D2	6,940	4,958,209	16,357
D3	612	571,113	3,452

5 RELATED WORKS

Structure sparsity (e.g.,LASSO [32], group LASSO [41], [37]) has been widely applied for feature learning, which can be extended for processing multi-view feature learning problem. Co-training [2] is widely applied to train two or more separate classifiers by exploring multi-view features. Multiple kernel learning [12] via a combination on the kernel-based method, multi-view deep ranking [4], joint subspace learning by learning a latent view shared by multi-view features [31], multi-Layer multi-View classification for Alzheimer's disease [42], deep heterogeneous feature learning for face recognition [3], multi-view correlated feature learning by uncovering shared component [35] [17], *etc* are also proposed for handling different multi-view learning problems. More detailed survey of multi-view learning can be found in [43], [34]. None of the above works, however, tackles the problem of heterogeneous data ranking problem in the context of app risk ranking. Moreover, most of the above works focus on image data and structured data. Due to the difference between image data and text data (and other unstructured data) investigated in this work, we offer a novel approach for capturing feature correlations and competitions across different views for both text and unstructured data.

6 EXPERIMENT RESULTS

Dataset Collection & Preparation We collected our dataset from *Google Play*. On *Google Play*, user reviews about Apps that he/she used are publicly available. Once we obtain the Google ID of a user, we can locate all Apps the user has reviewed. Therefore, we obtain a list of Google user IDs and write a crawler to retrieve all rated Apps of these users. We collect three datasets at different period in 2017. The first dataset (D1) is collected from Google Play at March which contains 6,234 apps. The second dataset (D2) contains 6,940 apps at July and the third dataset (D3) is collected from top 1000 most popularly apps (by removing the overlaps with D1 and D2). There are no overlaps for the above datasets, and the apps are from a wide range of functionalities, including `game, news, music, social, movies/TV, travel, life`, *etc.* The details are summarized in Table 4.

Multi-view feature pool F For each retrieved App, we crawled its reviews, developer descriptions and apk file as in [33]. Therefore, we have V=4 types of privacy indicators: including permission, user-review, descriptions and advertisement. For each mobile app,

Table 5: Comparisons of our method against other methods in terms of classification performance (mean ± standard deviation). Compared methods: features from each view: Ads library, permission, user comments, description, simple concatenation of all features, and other baselines on D1 dataset.

Method	dataset: D1
Ads library (only)	64.38 ± 1.63
Permission (only)	81.23 ± 2.12
User comments (only)	78.65 ± 2.01
Description (only)	49.23 ± 1.48
Feature concatenation	76.12 ± 2.31
Group [41]	85.23 ± 1.12
LASSO [32]	81.97 ± 2.12
$\ell_{2,1}$ [1]	84.23 ± 1.75
HMNB [29]	85.11 ± 0.34
GBDT [10]	82.47 ± 0.45
MVLR [36]	84.74 ± 0.17
MCCA [14]	83.19 ± 0.45
FCL [35]	85.87 ± 0.31
GElvL (logistic+our method)	88.35 ± 1.68
GElvNL (MLP+our method)	**88.92 ± 0.79**

Table 6: Comparisons of our method against other methods in terms of classification performance (mean ± standard deviation) on D2 dataset.

Method	dataset:D2
Ads library (only)	65.17 ± 1.75
Permission (only)	76.23 ± 2.31
User comments (only)	80.15 ± 1.54
Description (only)	64.56 ± 1.78
Feature concatenation	78.10 ± .93
Group [41]	86.55 ± 1.23
LASSO [32]	83.24 ± 0.78
$\ell_{2,1}$ [1]	85.87 ± 2.12
HMNB [29]	84.23 ± 1.12
GBDT [10]	86.34 ± 0.75
MVLR [36]	86.17 ± 0.29
MCCA [14]	85.23 ± 0.51
FCL [35]	84.29 ± 0.43
GElvL (logistic+our method)	90.47 ± 1.70
GElvNL (MLP+our method)	**91.25 ± 0.92**

the permission feature is encoded using one-hot encoding as 235-dimensional binary vector[2]. We adopt the annotation tool [5] developed to automatic label the user review at app-level with 11 privacy binary labels. The description texts are properly preprocessed and the *tf-idf* feature for each term w in App i's description is computed as $log(TF_w^i + 1)log(\frac{\#App}{DF_w+1})$, where TF_w^i is the term frequency of w for App i's description and DF_w is the number of Apps that have w in their descriptions. Regarding the advertisement feature, we use the method proposed in [13] as a guideline to implement our own analysis tool to extract the ads libraries and then generate the features using one-hot encoding. We carefully generate these features in experiments, and we believe feature construction and feature engineering are important parts in building the model. That is also the reason why we generate the heterogeneous features for app risk categorization.

For the ground-truth of risk category results, we use majority voting[3] based on three experts: (1) two android experts from the major Android phone company; (2) CMU `privacygrad`[4]. If all three reach an agreement, then the label is consistent. If two of them have the same decision, we use the agreement as the final label. If three of them are different, we use the *median* value of their votes as the labeling result.

6.1 Experimental setting

After generating the heterogeneous features, we learn the projection $\mathbf{W}$ in our model. Parameters of α, β are searched in the range of $(10^{-5}, 10^{-4}, \cdots, 10^4, 10^5)$. We feed the learned $\mathbf{W}$ for classification purpose. We compare our method against the following methods:

• Feature sets from each individual view: ads, user-review, description and permission. We view the result from each single-view as a baseline method.

• A simple concatenation of all features (shown as *CF*) and then feed all the features into logistic regression classifier. Notice that these heterogeneous features have different scales and dimensions, and therefore some feature may dominate the classification process while others are diluted.

• Features learned from only group lasso method (shown as *Group*) (i.e., $\beta = 0$ in Eq.(2) using standard logistic loss, where features in each view are put in the same group, without any distinction.

• Feature learning from standard structural sparsity with logistic loss functio using ℓ_{21}-based feature selection [1].

• Feature learning from standard LASSO (shown as *LASSO*) with logistic loss.

• Mixture of Hierarchical Naive Bayes model[5] [29] with all features feeding into the model, which automatically correlates the risk ranking with the features in a generative way.

• Gradient boost decision tree (GBDT) [10] with an ensemble of a series of decision trees using all the heterogeneous features for categorization purpose. The tree number is set 30–50, number of leaf node=30, learning rate=0.8 and sampling rate=0.8.

• Multi-view unified subspace learning (MVLR) [36] jointly conducts multi-view representation learning and subspace learning.

• Multiview Canonical Correlation Analysis (MCCA) [14] aims to find the common representation that maximizes class separation while minimizing the noises.

• Feature correlation learning (FCL) [35] learns the individual information in each view and captures feature correlations among multiple views after learning the shared component.

[2]Android system has 235 permissions

[3]We are not saying our decision is perfect, but at least is a good solution to eliminate the bias based on the expert voting.

[4]`https://www.hcii.cmu.edu/research/privacygrade`

[5]However, Peng et al [29] only applied this model to distinguish benign apps from malicious apps only using `permission` features.

Table 7: Comparisons of our method against other methods in terms of classification performance (mean ± standard deviation) on D3 dataset.

Method	dataset:D3
Ads library (only)	60.08 ± 2.31
Permission (only)	75.72 ± 1.75
User comments (only)	74.91 ± 0.81
Description (only)	51.13 ± 0.35
Feature concatenation	70.43 ± 1.50
Group [41]	79.23 ± 0.47
LASSO [32]	78.41 ± 1.56
$\ell_{2,1}$ [1]	79.98 ± 0.32
HMNB [29]	76.74 ± 0.83
GBDT [10]	78.08 ± 1.55
MVLR [36]	75.47 ± 0.32
MCCA [14]	77.98 ± 0.10
FCL [35]	79.05 ± 0.27
GElvL (logistic+our method)	82.32 ± 1.29
GElvNL (MLP+our method)	**82.96 ± 2.44**

• Our method with two versions: (i) GElvL:using logistic loss in Eq.(2); (ii) gElvNL: we use multi-layer perception (MLP) with 3-layers and ReLu activation function in Eq.(2).

6.2 Risk ranking result comparisons

On each dataset, we do five-fold cross validation, i.e., randomly select 80% apps as the labeled apps, and use the rest ones for testing the performance of different algorithms. The above process is repeated for 5 times before reporting the average performances. We use the measurement *accuracy* to evaluate the performance of risk ranking. The performance using 5-fold cross validation are shown in Tables. 5, 6 and 7.

We make several important observations from experiment results. **(i)** Proposed exclusive feature learning method consistently performs better than the other baselines by exploiting the feature correlations while promoting feature diversity. **(ii)** Compared to the performance from the best single-view features, exclusive feature fusion provides better results. **(iii)** For each single-view feature set, their performances vary differently. **(iv)** The performance uplift over group lasso is 4%− 5% on average. **(v)** Our method outperforms the HMNB because simple concatenation will treat features of different views equally, and therefore the learned parameters are biased due to the misalignment among different feature space. GBDT is popularly used for web ranking, however, its performance degrades when it is applied for the classification instead of a typical point-ranking task in the context of mobile apps. **(vi)** Subspace and feature correlation learning generally performs well on image datasets while our method excels on text and unstructured features for mobile apps. **(vii)** Our method gives slightly better performance when applying MLP to embed features in non-linear space, compared to logistic loss.

Discussions on feature types (i) On all datasets, permission indicator contributes the most for ranking the risks of mobile apps, which explains why researches [11] use the permission as the only risk ranking indicator. However, permission indicator is not the only contributor for risk ranking. If multi-modal features are used, the performance is improved. This suggests the necessity of using multi-modal features. **(ii)** Besides permission indicator, the user review contributes the second most for the risk ranking. From users' perspective, the user reviews reveal how the users feel about the app based on their personal experiences. Users' expectation plays a big role in on how much the users can tolerate the apps' behavior, and understand the apps' security risks. **(iii)** Ads-library is also an important indicator to infer the security risks of mobile apps, since the ads libraries communicate with the ads network to get and show ads, based on the app context and users' personal information such as location, context, etc. We believe the ads library will bring serious privacy risk for a mobile app. **(iv)** It is generally felt that developers' description may be more helpful to infer the risks of mobile apps. Our results, however, indicate that description is not as important as we think for evaluating the risks of different mobile apps. This is due to the fact that most of the descriptions are related to the functionality and quality of the app, while only few descriptions mention security/privacy concerns.

Case study. For a game app named `Cut the Rope FULL FREE` [6], it falls into `high` risk category because it loads many ads libraries such as `facebook`, `admob`, `flurry`, *etc*; it requires location, account and other sensitive resources; and users also complain about the ads and finance issues.

7 CONCLUSION

This paper presents a novel approach to learn app risk from heterogeneous features. We address the automatic feature learning problem from features generated from different views, where $\ell_{2,1}$-norm is enforced to select features at inter-view level, and exclusive $\ell_{1,2}$ is enforced to select features at intra-view level. We propose an effective iterative re-weighted algorithm to solve the problem, whose convergence is rigorously proved. We apply the method to solve the mobile app risk ranking problem, and extensive experiments on mobile app risk categorization tasks demonstrate the good performance of our method.

8 APPENDIX: ALGORITHM ANALYSIS

Theorem 8.1. *Under the updating rule of Eq.(16),* $J_1(\mathbf{u}^{t+1}) - J_1(\mathbf{u}^t) \leq 0$.

To prove Theorem 1, we need two lemmas.

Lemma 8.2. *Under the updating rule of Eq.(16),* $J_2(\mathbf{u}^{t+1}) < J_2(\mathbf{u}^t)$.

Lemma 8.3. *Under the updating rule of Eq.(16),*

$$\Big(J_1(\mathbf{u}^{t+1}) - J_1(\mathbf{u}^t)\Big) \leq \Big(J_2(\mathbf{u}^{t+1}) - J_2(\mathbf{u}^t)\Big). \tag{18}$$

REFERENCES

[1] A. Argyriou, T. Evgeniou, and M. Pontil. 2008. Convex multi-task feature learning. *Machine Learning* 73, 3 (2008), 243–272.

[2] A. Blum and T. Mitchell. 1998. Combining labeled and unlabeled data with co-training. *Proc. Comp. Learning Theo. (CLT1998) pp.92-100* (1998).

[3] Navaneeth Bodla, Jingxiao Zheng, and et al. 2017. Deep Heterogeneous Feature Fusion for Template-Based Face Recognition. *CoRR* abs/1702.04471 (2017). http://arxiv.org/abs/1702.04471

[4] Guanqun Cao, Alexandros Iosifidis, Moncef Gabbouj, Vijay Raghavan, and Raju Gottumukkala. 2018. Deep Multi-view Learning to Rank. *CoRR* abs/1801.10402 (2018).

[6] `https://play.google.com/store/apps/details?id=com.zeptolab.ctr.ads&hl=en`

[5] L Cen, L Si, N Li, and H Jin. 2014. User Comment Analysis for Android apps and CSPI Detection with Comment Expansion. In *SIGIR*.

[6] C. Chen, Y. Li, and J. Huang. 2012. Learning with Forest Sparsity. *CoRR* abs/1211.4657 (2012).

[7] Zhenpeng Chen, Xuan Lu, Wei Ai, Huoran Li, Qiaozhu Mei, and Xuanzhe Liu. 2018. Through a Gender Lens: Learning Usage Patterns of Emojis from Large-Scale Android Users. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. 763–772.

[8] W. Enck, D. Octeau, P. Mcdaniel, and S. Chaudhuri. 2011. A study of Android application security. In *In Proc. USENIX Security Symposium*.

[9] M. Frank, B. Dong, A. P. Felt, and D. Song. 2012. Mining Permission Request Patterns from Android and Facebook Applications. *CoRR* abs/1210.2429 (2012).

[10] Jerome H. Friedman. 2000. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics* 29 (2000), 1189–1232.

[11] C. S. Gates, J. Chen, N. Li, and R. W. Proctor. 2014. Effective Risk Communication for Android Apps. *IEEE Trans. Dependable Sec. Comput.* 11, 3 (2014), 252–265.

[12] Mehmet Gönen and Ethem Alpayd. 2011. Multiple Kernel Learning Algorithms. *Journal of Machine Learning Research* 12 (jul 2011), 2211–2268.

[13] M. C. Grace, W. Zhou, X. Jiang, and A.R. Sadeghi. 2012. Unsafe exposure analysis of mobile in-app advertisements. In *WISEC*. 101–112.

[14] David R. Hardoon and etc. 2004. Canonical Correlation Analysis: An Overview with Application to Learning Methods. *Neural Computation* 16 (2004), 2639–2664.

[15] Shifu Hou, Yanfang Ye, Yangqiu Song, and Melih Abdulhayoglu. 2017. HinDroid: An Intelligent Android Malware Detection System Based on Structured Heterogeneous Information Network. In *KDD*. ACM, 1507–1515.

[16] L. Jacob, G. Obozinski, and J.-P. Vert. 2009. Group lasso with overlap and graph lasso. In *ICML*. 55.

[17] Xiao-Yuan Jing, Rui-Min Hu, Yang-Ping Zhu, Shan-Shan Wu, Chao Liang, and Jing-Yu Yang. 2014. Intra-view and Inter-view Supervised Correlation Analysis for Multi-view Feature Learning. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI'14)*. 1882–1889.

[18] Deguang Kong. 2017. Science Driven Innovations Powering Mobile Product: Cloud AI vs. Device AI Solutions on Smart Device. *CoRR* abs/1711.07580 (2017). arXiv:1711.07580 http://arxiv.org/abs/1711.07580

[19] Deguang Kong, Lei Cen, and Hongxia Jin. 2015. AUTOREB: Automatically Understanding the Review-to-Behavior Fidelity in Android Applications. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*. ACM, New York, NY, USA, 530–541. https://doi.org/10.1145/2810103.2813689

[20] Dawei Li, Xiaolong Wang, and Deguang Kong. 2018. DeepRebirth: Accelerating Deep Neural Network Execution on Mobile Devices. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. 2322–2330.

[21] Bin Liu, Deguang Kong, Lei Cen, Neil Zhenqiang Gong, Hongxia Jin, and Hui Xiong. 2015. Personalized Mobile App Recommendation: Reconciling App Functionality and User Privacy Preference. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining (WSDM '15)*. ACM, New York, NY, USA, 315–324. https://doi.org/10.1145/2684822.2685322

[22] J. Liu, S. Ji, and J. Ye. 2012. Multi-Task Feature Learning Via Efficient l2,1-Norm Minimization. *CoRR* abs/1205.2631 (2012).

[23] J. Liu and J. Ye. 2010. Moreau-Yosida Regularization for Grouped Tree Structure Learning. In *NIPS*. 1459–1467.

[24] Yun Ma, Ziniu Hu, Yunxin Liu, Tao Xie, and Xuanzhe Liu. 2018. Aladdin: Automating Release of Deep-Link APIs on Android. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. 1469–1478.

[25] Y. Nesterov. 2007. Gradient methods for minimizing composite objective function. *ECORE Discussion Paper* (2007).

[26] F. Nie, H. Huang, X. Cai, and C.H.Q. Ding. 2010. Efficient and Robust Feature Selection via Joint $\ell_{2,1}$-Norms Minimization. In *NIPS*. 1813–1821.

[27] S.J Nocedal, J.; Wright. 2006. *Numerical Optimization*. Springer-Verlag, Berlin, New York.

[28] R. Pandita, X. Xiao, W. Yang, and et al. 2013. WHYPER: Towards Automating Risk Assessment of Mobile Applications. In *USENIX Security*. 527–542.

[29] Hao Peng, Chris Gates, Bhaskar Sarma, and et al. 2012. Using Probabilistic Generative Models for Ranking Risks of Android Apps. In *CCS*. 241–252.

[30] Z. Qu, V. Rastogi, X. Zhang, and et al. 2014. AutoCog: Measuring the Description-to-permission Fidelity in Android Applications. In *ACM CCS*. 1354–1365.

[31] Abhishek Sharma. 2012. Generalized Multiview Analysis: A Discriminative Latent Space. In *CVPR*. IEEE Computer Society, Washington, DC, USA, 2160–2167. http://dl.acm.org/citation.cfm?id=2354409.2355114

[32] R. Tibshirani. 1996. Regression shrinkage and selection via the LASSO. *J. Royal. Statist. Soc B.* 58 (1996), 267–288.

[33] Nicolas Viennot, Edward Garcia, and Jason Nieh. 2014. A Measurement Study of Google Play. In *ACM SIGMETRICS*. ACM, 221–233. https://doi.org/10.1145/2591971.2592003

[34] Chang Xu, Dacheng Tao, and Chao Xu. 2013. A Survey on Multi-view Learning. *CoRR* abs/1304.5634 (2013).

[35] Xiaowei Xue, Feiping Nie, Sen Wang, Xiaojun Chang, Bela Stantic, and Min Yao. 2017. Multi-View Correlated Feature Learning by Uncovering Shared Component. In *AAAI*. 2810–2816.

[36] Zhe Xue, Guorong Li, and Qingming Huang. 2016. Joint Multi-View Representation Learning and Image Tagging. In *AAAI*. 1366–1372.

[37] Haichuan Yang, Yijun Huang, Lam Tran, Ji Liu, and Shuai Huang. 2016. On Benefits of Selection Diversity via Bilevel Exclusive Sparsity. In *CVPR*. 5945–5954.

[38] T. Yang, R. Jin, M. Mahdavi, and S. Zhu. 2012. An Efficient Primal-Dual Prox Method for Non-Smooth Optimization. *CoRR* abs/1201.5283 (2012).

[39] Wei Yang, Deguang Kong, Tao Xie, and Carl A. Gunter. 2017. Malware Detection in Adversarial Settings: Exploiting Feature Evolutions and Confusions in Android Apps. In *Proceedings of the 33rd Annual Computer Security Applications Conference (ACSAC 2017)*. ACM, New York, NY, USA, 288–302. https://doi.org/10.1145/3134600.3134642

[40] L. Yuan, J. Liu, and J. Ye. 2011. Efficient Methods for Overlapping Group Lasso. In *NIPS*. 352–360.

[41] M. Yuan and M. Yuan. 2006. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B* 68 (2006), 49–67.

[42] Changqing Zhang, Ehsan Adeli, Tao Zhou, Xiaobo Chen, and Dinggang Shen. 2018. Multi-Layer Multi-View Classification for Alzheimer's Disease Diagnosis. In *AAAI*.

[43] Jing Zhao, Xijiong Xie, Xin Xu, and Shiliang Sun. 2017. Multi-view Learning Overview. *Inf. Fusion* 38 (2017), 43–54.

[44] H. Zhu, H. Xiong, Y. Ge, and E. Chen. 2014. Mobile App Recommendation with Security and Privacy Awareness. In *KDD*. ACM, 951–960.

A Neural Bag-of-Words Modelling Framework for Link Prediction in Knowledge Bases with Sparse Connectivity

Fanshuang Kong
BDBC and SKLSDE, School of Computer Science and Engineering, Beihang University
Beijing, China

Richong Zhang*
BDBC and SKLSDE, School of Computer Science and Engineering, Beihang University
Beijing, China

Hongyu Guo
Institute for Information Technology of the National Research Council Canada
Ottawa, Ontario, Canada

Samuel Mensah
BDBC and SKLSDE, School of Computer Science and Engineering, Beihang University
Beijing, China

Zhiyuan Hu
College of Information Science and Technology, Beijing University of Chemical Technology
Beijing, China

Yongyi Mao
School of Electrical Engineering and Computer Science, University of Ottawa
Ottawa, Ontario, Canada

ABSTRACT

Knowledge graphs such as DBPedia and Freebase contain sparse linkage connectivity, which poses severe challenge to link prediction between entities. In addressing this sparsity problem, our studies indicate that one needs to leverage model with low complexity to avoid overfitting the weak structural information in the graphs, requiring the simple models which can efficiently encode the entities and their description information and then effectively decode their relationships. In this paper, we present a simple and efficient model that can attain these two goals. Specifically, we use a bag-of-words model, where relevant words are aggregated using average pooling or a basic Graph Convolutional Network to encode entities into distributed embeddings. A factorization machine is then used to score the relationships between those embeddings to generate linkage predictions. Empirical studies on two real datasets confirms the efficiency of our proposed model and shows superior predictive performance over state-of-the-art approaches.

CCS CONCEPTS

• **Information systems** → **Data extraction and integration**; *Information extraction*; • **Computing methodologies** → Semantic networks.

KEYWORDS

Knowledge base; Link prediction

ACM Reference Format:
Fanshuang Kong, Richong Zhang, Hongyu Guo, Samuel Mensah, Zhiyuan Hu, and Yongyi Mao. 2019. A Neural Bag-of-Words Modelling Framework for Link Prediction in Knowledge Bases with Sparse Connectivity. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313550

*Corresponding author: zhangrc@act.buaa.edu.cn

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313550

1 INTRODUCTION

Knowledge Bases (KBs) such as DBPedia [8], Yago [18] and Freebase [1] contain precious large collaborative intelligence, on which many state-of-the-art NLP applications heavily rely, including question -answering [5], entity disambiguation [3], and information retrieval [4], amongst many others [7, 22, 23]. Knowledge within such graphs is typically represented as a triple (h, r, t), which indicates that there is a relation r from head entity h to tail entity t.

Unfortunately, the complexity of the real-world makes these knowledge graphs far from complete. In particular, the structural information provided in the existing knowledge triples is typically very sparse, which poses a severe challenge for link prediction in such graphs. For example, in the DBPedia knowledge graph, there are 99.41% entities (42966066 of entities in total) with at most 43 relations. Similarly, the Yago graph contains 95.17% of entities(38734252 of entities in total) but with at most 39 linkages [25]. To cope with such data scarcity, it is desirable to exploit additional information contained in the text descriptions of the entities and relations. To date, the distributed representations of the entities and relations are primarily constructed via encoding their corresponding text information. However, since an entity is only observed to participate in very few relations, and in more extreme cases, to participate in no relations at all (which gives rise to the OOKB, namely "out-of-knowledge-base" link prediction problem), each training example (an entity-relation-entity triple) only contains very weak supervisory signal to train the text encoder. Furthermore, this problem can be aggravated when the number of entities and the vocabulary size of the text are large, which is the case for current popular knowledge graphs. In such a scenario, one is naturally motivated to use a simple text encoder as possible so as to maintain a low model complexity and avoid over-fitting resulting from data sparsity.

In this paper, instead of promoting the use of sophisticated text encoders such as convolutional neural networks (CNNs), we advocate the philosophy of returning to the more basic bag-of-words models. Specifically, we propose a neural bag-of-word (NBOW) modelling framework, which contains three simple modules: selecting keywords to form bags of words, encoding entities from the bags of words, and computing the score of a triple using a factorization machine. To this end, we present two concrete examples in this framework, in which the entity encoding is carried out by simply averaging the word vectors in its bag of words or further modelled using a graph convolutional network (GCN) respectively. Our experimental results on two datasets, both old and new, demonstrate that a bag-of-words model is more effective than a CNN method in terms of entity encoding for link prediction in sparse knowledge graphs. In particular, we empirically show that, such simple models outperform significantly more sophisticated models, including the current art method ConMask in link prediction over sparse knowledge bases, for out-of-knowledge-base (or "open-world") link prediction. Our simple models have established new state-of-the-art results for the two benchmark datasets.

The rest of this paper is organized as follows: In Section 2, we present prior works on link prediction in knowledge base completion. In Section 3, we describe our model for entity encoding and relation scoring for knowledge base instances. In Section 4, we perform an experimental comparison of current methods against our model. We perform these experiments on the datasets DBPedia50K[17] and FB15K-237* which we generate from FB15K-237. We show that our method for link prediction is scalable and outperforms 'state-of-the-art' method, ConMask [17] in link prediction performance for the open-world assumption. We give concluding remarks in Section 5.

2 RELATED WORKS

Knowledge Base Completion (KBC) is one of the most important tasks in NLP. Many efforts have been invested in the link prediction task of KBC. For a typical link prediction model, we have an encoder and a decoder. An encoder learns feature representations of entities or relations and the decoder exploits these representations to predict the truth value of a given triple.

One of the first works of KBC falls under translation-based models [2, 9, 20]. These models generally adopt the same principle, $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, if the triple (h, r, t) is indeed a fact. More specifically, TransE [2] assumes a score funtion

$$f_r(h, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2 \tag{1}$$

which captures one-one relations, while other expressive translation-based models [9, 20] are capable of capturing other relations such as one-many, many-one and many-many by introducing a relationship-dependent parameter for the translation of head and tail entities.

Aside translational models, we have models such as ANALOGY [10] and its sub-models HolE [11], ComplexE [19], DistMult [24] addressing the problem of link prediction. ANALOGY addresses the problem by imposing analogical properties for the multi-relational encoding of entities and relations. ANALOGY decodes by using a relation-specific normal matrix W_r as a composition operator on the head and tail entity embeddings. The goal of ANALOGY is to receive a high score for a factual triple (h, r, t), where the score is formulated as

$$f_r(h, t) = \mathbf{h}^T W_r \mathbf{t} \tag{2}$$

ProjE[16] views the link prediction task as a ranking problem. ProjE is a neural model with a combination layer and a projection layer. Given a tail entity, the combination layer combines the tail entity embedding with a relation embedding. The resulting embedding vector is projected to all candidate head entities to compute a score for the head entity. The goal is to ensure that, top-ranked entities are the correct entities in the relation. The score for a candidate head h given a tail entity t and relation r is formulated as

$$f_h(t, r) = g(\mathbf{h} f(\mathbf{t} \oplus \mathbf{r}) + b_p) \tag{3}$$

where f and g are activation functions and $\oplus$ is a learnable combination operator.

Despite the promising successes in link prediction, the above model architectures restrict representation learning on the KB's structure. This constrains learning for only entities and relations in the KB. It also makes it non-trivial to leverage information captured in descriptions of entities which are usually accompanied with entities. Some recent works have addressed these issues and have achieved a boost in link prediction performance [17, 21].

DKRL [21] takes into account the textual information accompanied with entities. DKRL proposed a continuous bag-of-words model (CBOW) and a convolutional neural network (CNN) to build representations for word-based entity descriptions. Representations for RDF triples are learned from the structure of the KB. With the representations for both RDF triples and entity descriptions, they employ a translation-based model for decoding in the link prediction problem. ConMask [17] on the other hand uses a relationship-content mask, semantic averaging and a CNN to learn relationship-dependent embeddings for word-based entities. ConMask outperforms DKRL in link prediction at the expense of a high order of magnitude in parameters. In both models, they employ a CNN to assist in extracting representations for word-based entities. However, we show that a simpler framework, specifically, a neural bag-of-words model can be used to learn representations for word-based entities more efficiently.

In the present work, we develop a neural bag-of-words model, a simple model for link prediction in knowledge bases. We model the entity encoding for link prediction by constructing a bipartite graph of word-entity connectives, where entities are represented by their names or part of its descriptions (see Figure 1). Motivated by some recent works [15, 26] which employs a GCN to deal with highly multi-relational data such as knowledge graphs, we develop the encoder that operates directly on the bipartite graph and further employ a factorization machine for the link prediction task.

3 NEURAL BAG-OF-WORDS MODEL

3.1 Problem Definition

A Knowledge Base is a graph $\mathcal{G} = \{E, R, T\}$, where E is a set of entities-nodes, R is a set of relations-edges and T is a set of factual triples. $T = \{(h, r, t), h, t \in E, r \in R\}$. An entity of a triple may have a relevant textual description. We call this a snippet. We respectively denote h_s, t_s as the snippets of h, t. We denote (h', r, t') as a negative triple, i.e. a triple which does not exist in a knowledge base.

Similarly, snippets of entities of a negative triple is also denoted as h'_s and t'_s.

Our NBOW approach aims to find missing factual triples

$$T' = \{(h, r, t) | (h, r, t) \notin T, h, t \in E', r \in R\},$$

where E' is a set of entities such that $E \subset E'$. That is, OOKB entities belong to the set $E' \backslash E$.

For the link prediction task, the NBOW method consists of two steps: 1) encoding an entity's name, along with its snippet, and 2) predicting the possible linkage between encoded entities. We will discuss the two steps in the next two sections.

3.2 Entity Encoding

The entity encoding process aims to first extract keywords from an entity's name or description, resulting in a bag of words for each entity. The information in the bag of words is aggregated to form the distributed representation for the entity's name or its snippet. The aggregation can be efficiently achieved by either simply averaging the word vectors in its bag or leveraging a graph convolutional network to better capture the structural information of the words in the bag.

3.2.1 Content Extraction. We first extract relevant words from the entity's description. To this end, we use TF-IDF to highlight the important words (keywords) in the entity's description. Other works, such as ConMask [17] use a relation-dependent mask to extract relevant text given a relation. Unlike their model, the extracted content using TF-IDF can simultaneously involve several relations. We address this challenge by using a relation-attentive factorization model (we will discuss this in Section 3.3.1).

After having all distinct words from entity names and their snippets, we then construct a word embedding matrix $V \in \mathbb{R}^{l^w \times d^w}$ where l^w is the size of the vocabulary and d^w is the word embedding dimension. For a given triple (h, r, t), we build entity name representations $\mathbf{h}$ and $\mathbf{t}$ for both h and t. Each draws values from $\mathbb{R}^{d^w}$. We do this by aggregating the information in the bag for the entity, either by simply averaging the embeddings in the bag or leveraging a graph convolutional network to better infuse the knowledge in the bag. Similarly, we build entity snippet representations $\mathbf{h_s}$ and $\mathbf{t_s}$ for both h_s and t_s.

3.2.2 Aggregate Bag-of-Words with Average Pooling. The aggregation process (ψ) is defined as a function $\psi : S \to \mathbb{R}^{d^w}$, mapping a set of vectors drawn from $\mathbb{R}^{d^w}$ to the vector space $\mathbb{R}^{d^w}$. For n vectors, S can be represented as the set $S = \{v_1, v_2, \ldots, v_n\}$. The encoder $\psi(S)$ can take the following forms:

$$\psi(S) = \sum_{i=1} v_i \quad \text{(sum pool)} \tag{4}$$

$$\psi(S) = \frac{1}{n} \sum_{i=1} v_i \quad \text{(average pool)} \tag{5}$$

$$\psi(S) = \max(S) \quad \text{(max pool)} \tag{6}$$

where the function $\max(\cdot)$ is an elementwise max function [6]. In this paper, the average pool function is chosen due to its superior performance.

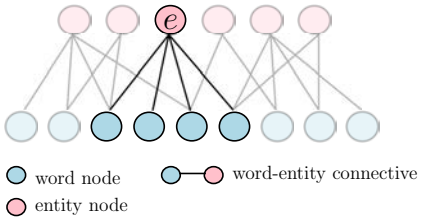


Figure 1: A bipartite graph induced from the KB. A word node is given by its word embeddings. An entity node is a word-based entity given by a feature vector resulting from the aggregation of word embeddings of its neighbors. An entity is connected to a word if that entity contains that word. As an example, the highlighted neighbors of entity e is aggregated using a continuous-bag-of-words encoder to form a feature representation for e.

3.2.3 Aggregating Bag-of-Words with GCN. Although one can leverage the simple average pool for aggregating information out of the bag-of-words model, other simple strategies can be used to further improve the information aggregation. To this end, we devise a basic graph convolutional network (GCN) model for this purpose.

We make use of word-based entities to induce two bipartite graphs $\mathcal{G}'_1$ and $\mathcal{G}'_2$ from $\mathcal{G}$ (see Figure 1). $\mathcal{G}'_1$ is a bipartite graph decomposed into word nodes and nodes for entity names, given by real-valued vectors. Similarly, $\mathcal{G}'_2$ is also decomposed into word nodes and nodes for relevant entity descriptions, given by real-valued vectors. Due to the similarity of the structure of $\mathcal{G}'_1$ and $\mathcal{G}'_2$, we denote either one of the bipartite graphs as $\mathcal{G}'_e$. Nodes are connected in $\mathcal{G}'_e$ if they share common words. Here and throughout the present work, we represent a word-based entity (entity name or entity snippet) as the italic word *entity* for simplicity and clarity.

The input to the GCN model is the bipartite graph $\mathcal{G}'_e$, where word nodes are initialized with pretrained word embeddings and *entity* nodes are initialized by the aggregation of the feature vectors of its neighbors. The GCN model proposed here makes efficient use of edges of subgraphs of $\mathcal{G}'_e$ to propagate and share information between nodes. More precisely, edges in $\mathcal{G}'_e$ serves as a transmission medium which has the sole purpose of propagating information to and from word nodes to *entity* nodes for the update of their embeddings. The GCN model is equipped with a CBOW encoder which extract *entity* embeddings for *entity* nodes by aggregating the word embeddings of its neighbors. The neighborhood of any node in $\mathcal{G}'_e$ is a bipartite subgraph which can be interpreted as an information propagation neural network. These subgraphs share parameters defining how information is transformed and propagated within the graph. The GCN encoder model learns convolution operators on these subgraphs for the transformation of information and the propagation across their edges to update embeddings of nodes.

When the GCN model operates on $\mathcal{G}'_e$, taking into account only the first-order neighborhood of nodes (word, *entity*), we learn parameters for a single update of the node embeddings. At each convolutional layer, convolution operations are imposed on the graph to update word nodes and *entity* nodes. To learn more informative representations, we successively apply convolutions on $\mathcal{G}'_e$

across the k-th order neighborhood. The GCN model architecture is illustrated in Fig. 2.

A single update for a word node takes the following form:

$$h_w^{k+1} = \phi\left(\sum_{j \in \mathcal{N}_w} \frac{1}{c_w} w_j^k h_j^k,\ w_w^k h_w^k\right) \tag{7}$$

Similarly, an update for an *entity* node takes the following form:

$$h_e^{k+1} = \phi\left(\sum_{j \in \mathcal{N}_e} \frac{1}{c_e} w_j^k h_j^k,\ w_e^k \psi(\mathcal{N}_e)\right) \tag{8}$$

where $(\cdot, \cdot)$ is an element-wise product, $h_e^k, h_w^k \in \mathbb{R}^{d^w}$ are the hidden state representation of *entity* e and word w respectively for the k-th convolutional layer of the neural network. $\mathcal{N}_w, \mathcal{N}_e$ correspond to the neighborhood of word w and *entity* e respectively. $\frac{1}{c_e}, \frac{1}{c_w}$ are normalization constants, where $c_e = |\mathcal{N}_e|$ and $c_w = |\mathcal{N}_w|$. $\psi(\cdot)$ is the bag-of-words model which we discussed in the previous section. $\phi(\cdot)$ is a non-linear activation function. For our model, we choose tanh function as the activation function $\phi(\cdot)$. w_j^k is a trainable parameter on layer k for the node j.

In short, for a given triple (h, r, t) with corresponding entity snippets h_s and t_s, the GCN model outputs real-valued representations $\mathbf{h}, \mathbf{t}, \mathbf{h_s}$ and $\mathbf{t_s}$.

3.3 Relation Scoring

To score the linkage predictions, we leverage a simple factorization machine which aggregates scores for different pairs of representations of a given triple. Traditional methods for interaction pair selection, specifically, [2, 20] measure scores of a candidate triple (h, r, t) by calculating the distance between the embedding $\mathbf{h}+\mathbf{r}$ and $\mathbf{t}$. Other expressive relation-dependent operators have successfully measured the score on candidate triples [11, 19].

The main assumption in these methods is that, there is a relation specific interaction which captures the relation r between h and t of any given triple. However, the presence of additional features h_s and t_s, the relevant descriptions of h and t offers new opportunities. ConMask incorporates some of these ideas by fusing different pair of variables in their model and achieve state-of-the-art results. In our approach, we postulate that, the pairs $h{-}r, t{-}r, h{-}t, h{-}t_s, h_s{-}t_s$ and $h_s - t$ captures informative interactions which can improve the link prediction task.

3.3.1 Relation-Attentive FM. We present a relation-attentive factorization model, a decoder which measure scores for different pair of representations. The decoder is likened to a factorization machine [13, 14] which highlights strong relationships between a pair of variables using a relation interaction parameter.

To capture interactions of pairs, we recall a training instance as a triple $(h, r, t) \in T$ with additional features h_s and t_s denoting snippets of the head entity h and tail entity t. Let $\mathcal{X}$ denote a training instance consisting of the set of variables $\{h, r, t, h_s, t_s\}$. We model the link prediction problem by measuring the overall scores of $h - r$ interactions, $t - r$ interactions, $h - t$ interactions, $h - t_s$ interactions, $h_s - t_s$ interactions and $h_s - t$ interactions for a training instance. Here, we extract *entity* embeddings from the encoder and initialize the relation r with a random vector. We define an interaction set I consisting of the 6 pair of interaction variables. For a given unordered pair $\{e_1, e_2\} \in I$, where $e_1, e_2 \in \mathcal{X}$, we formulate a relation interaction parameter γ(different from the embedding of relation r) which focuses on relation-specific parts of the interacting pair to compute a response. We define a score for the interaction of a pair as

$$g_\gamma(e_1, e_2) = \langle \mathbf{e_1} \odot \mathbf{e_2}, \gamma \rangle \tag{9}$$

The score assigns a scalar value to show the strength of interaction of a pair. This architecture enables the attention on latent information of a pair of variables which is closely related to the relation itself. This process is similar to that of ConMask, where a relationship-dependent content mask highlights relevant text at an early stage. We aggregate scores across all pairs in I. The total score for a training instance takes the form

$$f(\mathcal{X}) = \sum_{\{i,j\} \in I} g_\gamma(i, j) \tag{10}$$

The goal is to learn the relation interaction parameter γ such that, the model score high values for factual triples and low values for negative triples. See figure 3 for an illustration of this model.

3.3.2 Optimization Objective. For the model to distinguish factual triples from negative triples, we evaluate the model by generating a set of negative triples for each factual triple and add to the training set T. The optimization function we wish to minimize is

$$\mathcal{L} = -\sum_{\mathcal{X}, y} y \log \sigma(f(\mathcal{X})) - (1 - y) \log(1 - \sigma(f(\mathcal{X})) \tag{11}$$

where $\sigma(\cdot)$ is a softmax activation function, y is a binary value set to +1 if we have a factual training instance and 0 if we have a negative training instance.

4 EXPERIMENTS

In this section, we evaluate our NBOW model. The two versions of our model are denoted as AFM, where entity embeddings are extracted by aggregating relevant word embeddings using a CBOW encoder, and GCN+AFM, where the aggregated word embeddings are further enhanced using a GCN. In both models, we employ a factorization machine for the link prediction task in the open-world assumption. We propose both models to observe the boost in performance when we employ a GCN to model the noisy dependency of the entity embeddings. We compare our simple methods to the current state-of-the-art strategy ConMask and two more competitive baselines. In our evaluation, we report measures for Filter Mean Rank (FMR), Filtered Hit@10 (FHIT), Filtered Mean Reciprocal Rank (FMRR).

4.1 Datasets

We perform experiments on an extracted DBPedia dump; DBPedia50K introduced in [17]. DBPedia50K originally contains 32K entities. There are 28K redundant entities, thus, these entities have no factual triples in the dataset. Hence, they do not influence results for any model but just lead to a statistic for the number of entities as shown in Table 1. DBPedia50K is very different from some standard dataset like FB15K and FB15K-237. DBPedia50K has a sparse KB structure and a dense GCN graph structure. But FB15K and FB15K-237 have a dense KB structure and sparse GCN graph structure. To validate the efficiency of our model on the open-world

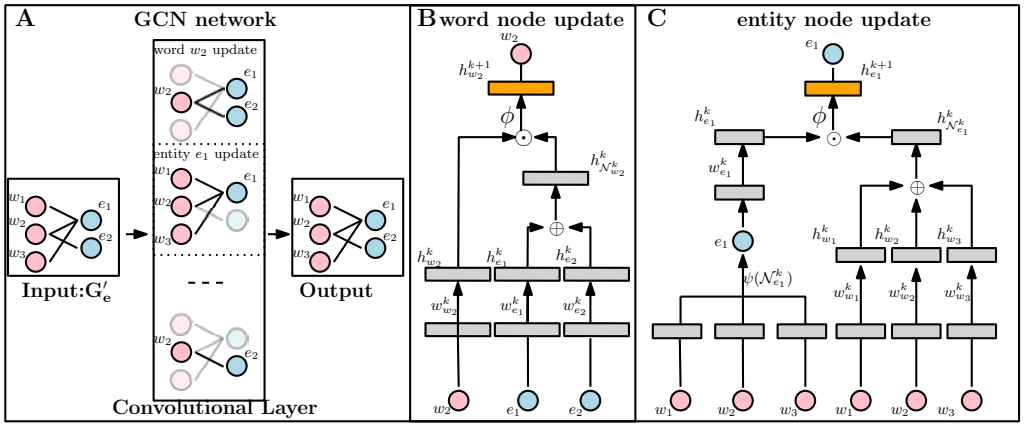


Figure 2: (A) Overview of a single layer GCN model architecture with first-order filters for word-based entity encoding. (B) Shown is a first-order filter imposed on the neighborhood of the word node (pink) to transform and propagate information from *entity* nodes (blue) to learn a representation for the word node. (C) Similar to that of B for information transformation, shown here is the propagation of information from the neighborhood of *entity* node to update its representation. Before update, a CBOW encoder ϕ aggregates the neighbors of the *entity* node resulting to an initial *entity* representation for the learning process. $\oplus$ is a vector summation operator and $\odot$ is an element-wise operator (B and C is based on the example given in A and follows Eq. 7 and 8 respectively)

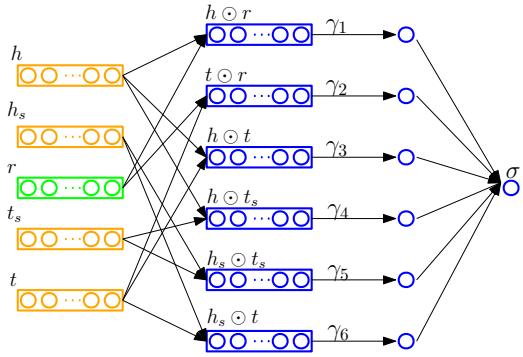


Figure 3: Relation-attention FM architecture

assumption for link prediction, we introduce a new dataset based on FB15K-237. We refer to this new dataset as FB15K-237*. FB15K however has many reciprocal triples so we don't generate an open world version of it.

FB15K-237* is generated by randomly selecting 10% of entities in the KB. These selected entities are classified as open-entities. Triples in the KB which contain these open-entities are held out for test and validation. All other triples are used for training. We define the validation set on 10% of the held out triples, the remaining 90% is used for the test set. The statistics of all datasets are shown in Table 1.

4.2 Parameter Settings

For a comparable study, we use similar parameters used in ConMask. We set the word embedding size $k = 200$ and draw embeddings from 200-dimensional trained GloVe vectors which have been made publicly available [12]. Maximum length for entity name $k_n = 32$, maximum length for entity snippet $k_c = 128$. Given a training instance $\mathcal{X} = \{h, t, h_s, t_s, r\}$, the encoder extracts *entity* embeddings for h, t, h_s, t_s and initializes a random vector for r. The relation-attentive model learns interactions from 6 unordered 2-tuples in the set $\{\{h, r\}, \{t, r\}, \{h, t\}, \{h, t_s\}, \{h_s, t_s\}, \{h_s, t\}\}$. During training, our neural network is composed of 1 convolutional layer as seen in (A) of Fig. 2. The batch size we use is $k_b = 200$ and we choose Adam as our optimizer with learning rate $lr = 0.01$. For a factual triple, we have 10 negative triples constructed by randomly corrupting either the head or tail entity. We train our model for 200 epochs.

4.3 Open-World Link Prediction

We follow evaluation protocols used in ConMask [17] for a comparable analysis. For the open-world link prediction task, we aim to predict missing entities h or t for a given factual triple (h, r, t). Given a partial triple $(?, r, t)$ or $(h, r, ?)$, this task requires that we rank a set of candidate entities for the position of missing entity. We refer to this missing entity as the target entity e. At the testing phase, for each partial test triple $(?, r, t)$ or $(h, r, ?)$, we apply a filtering method to rank specific entities. The filter skips all target entity candidates which have never been connected to the relation r in the training set. As noted by [17], it avoids the ranking of relation instances which might be nonsensical. For every target entity selected, we compute a similarity score according to our proposed score function f on the training instance $\mathcal{X}$ for the candidate triple (e, r, t) or (h, r, e). We use these scores to rank entities e in descending order and measure the performance of the model with the following evaluation metrics: (1) the average rank of correct entities (denoted as Filtered Mean Rank), (2) the proportion of correct entities in the top 10 of all entities (denoted as Filtered Hits@10), and (3) a filtered mean reciprocal rank. Note that, an effective model scores a *low* filtered mean rank (FMR), a *high* filtered mean reciprocal rank (FMRR) with a *high* Flitered Hits@10 (FHIT).

Table 1: Dataset Statistics: #Closed Ent. is the number of entities in the KB, #Open Ent. is the number of OOKB entities & #Words is the number of words we learn embeddings for the word-based entity embedding extraction task

Dataset	#Ent.	#Rel.	#Train	#Valid	#Test	#Closed Ent.	#Open Ent.	#Words.
FB15k-237*	14,438	237	241,962	6270	56,431	12,907	1,531	42,780
DBPedia50K	30,449	365	32,388	399	10,969	24,624	5,825	48,185

Table 2: Open-World Performance on DBPedia50K and FB15K-237*

Model	DBPedia50K						FB15K-237*					
	Head			Tail			Head			Tail		
	FMR	FHIT	FMRR	FMR	FHIT	FMRR	FMR	FHIT	FMRR	FMR	FHIT	FMRR
Target Filtering Baseline	573	0.08	0.04	104	0.23	0.11	474	0.1	0.05	292	0.22	0.12
DKRL	490	0.09	0.08	70	0.40	0.23	-	-	-	-	-	-
ConMask(Re-Run)	95	0.39	0.35	16	0.81	0.61	146	0.37	0.2	88	0.53	0.34
AFM	94	0.52	0.35	13	**0.83**	**0.64**	**122**	0.44	0.28	93	0.52	0.34
GCN+AFM	**89**	**0.56**	**0.39**	**12**	**0.83**	0.63	125	**0.47**	**0.31**	**85**	**0.54**	**0.35**

We evaluate AFM and GCN+AFM on the link prediction task in open-world KBC for the datasets summarized in table 1. We report results for models AFM, GCN+AFM, ConMask and other baseline methods implemented in the works of [17]. [17] re-implemented the method for DKRL [21] to handle the OOKB link prediction task. The overall performance of the models on DBPedia50K and FB15K-237* are shown in Table 2. The best performance are bold-faced.

As we can observe, AFM generally achieves better performance than the baseline methods on both datasets. More importantly, it shows evidence of superior performance to the state-of-the-art method ConMask in both DBPedia50K and FB15K-237*. To compare the detail performance metrics, we observe that AFM managed to outperform ConMask entirely on DBPedia50K, showing a significant boost for FMR, FHIT and FMRR in both head and tail target entity prediction. On the FB15K-237* dataset, the results are competitive for both ConMask and AFM. Here, we observe that AFM impressively outperforms ConMask when predicting head target entities. On the other hand, for the tail target entity prediction, AFM has a slight degradation in performance (lower by 0.01) in terms of FHIT, a higher FMR (higher by 5), and equal FMRR when comparing to ConMask. Promisingly, when we augment AFM with the GCN, namely the GCN+AFM model, it would be reasonable to expect a boost in predictive performance since we expect the GCN to enhance the entity embeddings. We clearly observe that this is the case, GCN+AFM outperforms all baselines including ConMask in every metric on the two datasets.

4.3.1 Scalability. In our efficiency analysis, we calculate the computational time for training an epoch (measured in seconds) on the DBPedia50K dataset. All training is done on a GeForce Titan Xp system with 2×12GB memory. We compare training times of our models and that of ConMask in particular. This result is significant because knowledge bases are known to be usually large in size and will therefore have a large number of training instances. For example, DBPedia500K [17] has 3, 102, 677 training instances. With this large number of instances, state-of-the-art models like ConMask with complex architectures requiring the tuning of hundreds of hyperparameters will take a longer time to train in 1 epoch. Models for KBC tasks must therefore be as simple as possible but efficient to learn large KBs. From the analysis, we observed that ConMask used 80*s* to complete an epoch training on DBPedia50K, while GCN+AFM required only 50*s* and AFM executed much faster as expected. Our proposed methods reduces about 40% of execution time when compared to that of ConMask. These results suggests that our simple methods have better scalability to large knowledge graphs. The results also informs us that our model makes better use of its parameters, and benefits from simplicity and efficiency compared to ConMask which has a significant model complexity.

5 CONCLUSION

In this study, we advocate the philosophy of exploiting simple models for link prediction in large knowledge graphs due to their sparse linkage connectivity between entities. We propose a simple neural bag-of-words model for link prediction in large knowledge graphs, named GCN+AFM, which accounts for both entity representation and link prediction. Specifically, our proposed model utilizes TF-IDF and average pooling and then enhances entity representations via a GCN. Experimental results show that the two versions of our model, namely AFM and GCN+AFM, achieve superior performance and reduce an order of magnitude in the number of parameters needed and about 40% execution time when compared to the current art model. We also demonstrate that the simple version of our proposed model, named AFM, achieves better performance to state-of-the-art methods. These facts make our proposed strategy suitable for learning from large real-world knowledge graphs with sparse entity connectivity.

ACKNOWLEDGMENTS

This work was supported in part by the China 973 Program under Grant 2015CB358700, in part by the National Natural Science Foundation of China under Grant 61772059 and Grant 61421003, in part by the Beijing Advanced Innovation Center for Big Data and Brain Computing, and in part by the State Key Laboratory of Software Development Environment.

REFERENCES

[1] Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. 1247–1250.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*. 2787–2795.

[3] Silviu Cucerzan. 2007. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. 708–716.

[4] Jeffrey Dalton, Laura Dietz, and James Allan. 2014. Entity query feature expansion using knowledge base links. In *Proceedings of the 37th international ACM SIGIR conference on Research and development in information retrieval*. 365–374.

[5] David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. 2010. Building Watson: An Overview of the DeepQA Project. *Ai Magazine* 31, 3 (2010), 59–79.

[6] Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. 2017. Knowledge Transfer for Out-of-Knowledge-Base Entities : A Graph Neural Network Approach. In *Twenty-Sixth International Joint Conference on Artificial Intelligence*. 1802–1808.

[7] Gjergji Kasneci, Fabian M. Suchanek, Georgiana Ifrim, Maya Ramanath, and Gerhard Weikum. 2008. NAGA: Searching and Ranking Knowledge. In *2008 IEEE 24th International Conference on Data Engineering*. 953–962.

[8] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, and others. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[9] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI'15 Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2181–2187.

[10] Hanxiao Liu, Yuexin Wu, and Yiming Yang. 2017. Analogical Inference for Multi-relational Embeddings. *international conference on machine learning* (2017), 2168–2178.

[11] Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. 2016. Holographic embeddings of knowledge graphs. *national conference on artificial intelligence* (2016), 1955–1961.

[12] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[13] Steffen Rendle. 2010. Factorization Machines. In *2010 IEEE International Conference on Data Mining*.

[14] Steffen Rendle. 2012. Factorization Machines with libFM. *ACM Transactions on Intelligent Systems and Technology* 3, 3 (2012), 57.

[15] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, Ivan Titov, Max Welling, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. *extended semantic web conference* (2018), 593–607.

[16] Baoxu Shi and Tim Weninger. 2016. ProjE: Embedding Projection for Knowledge Graph Completion. *arXiv preprint arXiv:1611.05425* 1 (2016).

[17] Baoxu Shi and Tim Weninger. 2018. Open-World Knowledge Graph Completion. *national conference on artificial intelligence* (2018).

[18] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*. 697–706.

[19] ThÃĺo Trouillon, Johannes Welbl, Sebastian Riedel, ÃĽric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. (2016).

[20] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI'14 Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. 1112–1119.

[21] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *AAAI'16 Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. 2659–2665.

[22] Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit Semantic Ranking for Academic Search via Knowledge Graph Embedding. In *WWW '17 Proceedings of the 26th International Conference on World Wide Web*. 1271–1279.

[23] Bishan Yang and Tom M. Mitchell. 2017. Leveraging Knowledge Bases in LSTMs for Improving Machine Reading.. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1436–1446.

[24] Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. *international conference on learning representations* (2015).

[25] Xiaowang Zhang, Mingyue Zhang, Peng Peng, Jiaming Song, Zhiyong Feng, and Lei Zou. 2018. gSMat: A Scalable Sparse Matrix-based Join for SPARQL Query Processing. *arXiv preprint arXiv:1807.07691* (2018).

[26] Marinka Zitnik, Monica Agrawal, and Jure Leskovec. 2018. Modeling Polypharmacy Side Effects with Graph Convolutional Networks. *intelligent systems in molecular biology* 34, 13 (2018), 258814.

FARE: Diagnostics for Fair Ranking using Pairwise Error Metrics

Caitlin Kuhlman
cakuhlman@wpi.edu
Worcester Polytechnic Institute

MaryAnn VanValkenburg
mevanvalkenburg@wpi.edu
Worcester Polytechnic Institute

Elke Rundensteiner
rundenst@wpi.edu
Worcester Polytechnic Institute

ABSTRACT

Ranking, used extensively online and as a critical tool for decision making across many domains, may embed unfair bias. Tools to measure and correct for discriminatory bias are required to ensure that ranking models do not perpetuate unfair practices. Recently, a number of error-based criteria have been proposed to assess fairness with regard to the treatment of protected groups (as determined by sensitive data attributes, e.g., race, gender, or age). However this has largely been limited to classification tasks, and error metrics used in these approaches are not applicable for ranking. Therefore, in this work we propose to broaden the scope of fairness assessment to include error-based fairness criteria for rankings. Our approach supports three criteria: *Rank Equality, Rank Calibration*, and *Rank Parity*, which cover a broad spectrum of fairness considerations from proportional group representation to error rate similarity. The underlying error metrics are formulated to be rank-appropriate, using pairwise discordance to measure prediction error in a model-agnostic fashion. Based on this foundation, we then design a fair auditing mechanism which captures group treatment throughout the entire ranking, generating in-depth yet nuanced diagnostics. We demonstrate the efficacy of our error metrics using real-world scenarios, exposing trade-offs among fairness criteria and providing guidance in the selection of fair-ranking algorithms.

CCS CONCEPTS

• **Information systems → Decision support systems**; **Data analytics**; **Content ranking**.

KEYWORDS

Fairness, Fair Ranking, Pairwise Fairness, Fairness Auditing

ACM Reference Format:
Caitlin Kuhlman, MaryAnn VanValkenburg, and Elke Rundensteiner. 2019. FARE: Diagnostics for Fair Ranking using Pairwise Error Metrics. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313443

1 INTRODUCTION

As sophisticated machine learning increasingly impacts our lives on and offline, there is growing concern that discriminatory practices will be baked into automated decision models [3]. The potential for harm is vast, highlighting the need for open and transparent procedures to audit and correct for unfair bias. To address this, recent work on algorithmic fairness assesses the treatment of protected groups by examining the errors made by automated decision making procedures. The main focus of this prior research has been on classification tasks, where predictive models determine a binary outcome [7, 8, 15, 17, 23, 27, 31]. In our work, we broaden the scope of error-based fairness assessment to include rankings.

Motivation. Evaluating ranked data is the de facto process used today for decision making, in particular to make sense of the vast amount of information available online. Rankings simplify the information, helping us to make sense of options and limiting the scope of choices to consider. We rely on ranking models for everyday tasks such as purchasing products to pivotal life decisions such as applying to colleges or jobs. Companies rely on rankings to organize information, whether evaluating candidates to hire [1], or evaluating potential customers using credit scoring [20].

However, rankings can have major pitfalls. Large socio-economic datasets cannot easily be distilled into simple rankings without exploiting patterns that exist in the data, including subtle encodings of historical inequalities. This may lead to unfair decisions, in particular for regulated domains such as employment, education, and lending [3]. For instance, Amazon recently revealed a failed attempt to design a hiring algorithm to screen candidates which they found inadvertently encoded a gender bias against women [10]. Use of such a tool risks automating illegal discrimination. Yet, the development of ranking algorithms for hiring is widespread [1], while no systematic approaches to audit these methods are available to date. Or, consider college rankings published online and used by students and faculty. Highly ranked colleges often have poor outcomes for low-income students, such as lower graduation rates or excessive debt after graduation [29]. The rankings could be considered unfair for only providing utility to high-income students. Left unchecked, such harm compounds through the creation of negative feedback loops. Colleges consistently given a low rank will attract less talent, decreasing their potential to improve [16, 26]. These institutions are entrenched in the position determined by the ranking model - often proprietary and not disclosed.

State-of-the-Art in Fair Ranking. To avoid these kinds of unfair and potentially illegal discriminatory practices, mechanisms to audit ranking models that drive our decision making are essential. Initial methods for measuring unfairness in rankings have recently been put forth [5, 28, 30, 32]. However, they target only a single type of fairness criterion, namely, *statistical parity*. This criterion requires that members of different groups have the same proportional representation among desirable outcomes, i.e., in a top position in the ranking. In our hiring example, such a criterion would dictate that the top candidates in the ranking have a similar proportion of men and women as in the entire applicant pool.

However, this particular criterion may not be appropriate for all applications. Alternatively, a rich variety of fairness criteria available in the literature for classification are *error-based*, meaning

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313443

they require that the predictive model make the same "mistakes" about each group. In our college example, such a criterion might dictate that colleges not be erroneously ranked lower than their counterparts solely due to supporting low income students. We postulate that there is value in considering error-based fairness criteria for rankings, which to date have not yet been applied.

Challenges. For classification, *error-based* definitions of unfairness are measured by within-class error rates, such as True Positive Rate, True Negative Rate or probability of assignment to positive class, for members of different groups [17, 23]. However these metrics require binary class labels, and cannot be applied to rankings learned from training data with non-binary labels. Measuring error made by a ranking model therefore requires metrics appropriate to the task. Ranking models optimize for different types of often application-specific error. For instance, in their fair ranking method, Singh and Joachims [28] define criteria specific to online search, balancing the item exposure in a ranked list of search results with considerations such as document relevance and click-through rate. This narrow definition cannot be directly applied for other ranking applications such as our college ranking example. Instead, a general approach to fair error-based ranking is preferable.

Further, notions of fairness for classification depend on a *preferred outcome* determined by membership in the positive class. Recent methods define an analogous notion based on the prefix of the ranking. While this approach appears to naturally capture the preferred rank outcome, focusing only on the top items in a ranking has severe shortcomings. For many applications, accuracy throughout the entire ranking matters. If the rank position determines funding, bottom-ranked institutions could lose resources and be forced to close. If rankings are used to determine peer groups for tournament-style competitions, inaccuracies low in the ranking would result in mismatched opponents. *In short, accuracy is important at all positions of the ranking.*

Proposed Approach. In this work, we design the first comprehensive auditing methodology for error-based fairness assessment of rankings. In particular, we design analogues for the three core types of fairness criteria in the literature on fair classification [7, 8, 13, 15, 17, 23, 27, 31]. Our proposed criteria, *Rank Equality*, *Rank Calibration*, and *Rank Parity*, together give a comprehensive nuanced assessment of group unfairness in rankings. Our analysis rests on the development of appropriate error metrics for fair ranking. For this, pair inversions, as in foundational rank evaluation techniques [22], are leveraged as basic building block for measuring rank errors. This proposed fairness assessment is model-agnostic and applicable for a wide range of ranking applications.

Further, we propose an auditing mechanism to apply our proposed criteria to measure fairness throughout an entire ranking instead of only in the top results. Our approach, called FARE (for Fair Auditing based on Rank Error), uses a sliding window technique to measure ranges of ranking error and generates error sequences which capture each group's treatment throughout the entire ranking. Custom FARE diagnostics provide a nuanced summary of results, while remaining interpretable to stakeholders who rely on rankings for decision making.

We evaluate our fairness metrics in a comprehensive experimental analysis. Using our FARE framework, we evaluate rankings created using state-of-the-art fair ranking methods [32] on real-world datasets [2, 19]. Our analysis identifies strengths and weaknesses of these existing techniques with regard to fairness. We demonstrate how our analysis guides the choice of appropriate fair ranking correction method to apply.

Key contributions of our work include:

(1) We define a new set of fairness criteria customized for rankings: *Rank Equality, Rank Calibration, and Rank Parity* based on novel rank-specific error metrics. These criteria are model-agnostic and applicable for many ranking applications, capturing key notions of fairness previously applied only for classification.
(2) We present FARE, the first comprehensive framework to audit ranking models using error-based fairness criteria. FARE offers a suite of fairness diagnostics for ease of interpretation.
(3) We demonstrate the application of our FARE framework to assess the capability of state-of-art rank correction methods to achieve fairness, and illustrate the power of FARE's diverse criteria to account for unfair bias in rankings across domains.

2 RELATED WORK

Measuring Fairness in Classification. No single rule has been shown to definitively determine the fairness of an algorithm. The majority of criteria for evaluating fairness focus on the treatment of *groups* determined by some protected data attribute. *Error-based group fairness criterion* [7, 17, 23, 27] are concerned with whether the predictions of a classifier are fair with respect to the group membership of items in the training dataset. To verify this, fairness assessment checks whether groups are treated 'similarly,' meaning error rates for each group are within some threshold. It has been observed that all criteria cannot necessarily be satisfied at the same time [7, 23]. Therefore the appropriate choice of fairness metric is context-dependent. Group fairness criteria can be categorized as:

- **Equalized Odds**, coined by Hardt et al. [17], seeks to ensure that the probability of an object being assigned a particular label by the classifier is independent of its group membership, conditional on the true class label [7, 17, 23]. To verify this, Equalized Odds stipulates that the *false positive and true positive error rates* must be similar across all groups.
- **Calibration**, a group-wise measure of fairness for probabilistic classifiers, requires that the calibration error for each group is similar [7, 23, 27]. Calibration error indicates the difference between the true likelihood of membership in the positive class and the probability given by the classifier. For example, if there are 100 objects assigned a 90% chance of being in the positive class by the classifier, approximately 90 of these objects should actually be positive.
- **Statistical Parity** requires proportional representation of each group [8, 15, 31]. This is similar in spirit to the 80% rule for evaluating disparate impact in judicial rulings [15] and affirmative action quotas employed by many institutions. Typically the desired outcome is that some minimal proportion of minority group members are predicted as positive.

Measuring Fairness in Rankings. Recent work has begun to address group fairness concerns in rankings [5, 28, 30, 32]. Most works [5, 30, 32] focus on enforcing only *Statistical Parity* criterion, and detect unfairness only in the top-k prefix of a ranking. Criteria

based on prediction error have not yet been applied for fair ranking to our best knowledge.

Yang and Stoyanovich [30] measure fairness according to a discounted cumulative scoring metric that evaluates the proportional representation of groups. In [5] fairness is defined according to a threshold on the maximum proportion of the majority group allowed in the prefix. Zehlike et al. [32] extend these statistical parity approaches, designing a greedy algorithm to ensure the ranking meets the criterion while optimizing for utility. Singh and Joachims target fair ranking in information retrieval specifically [28]. Metrics favoring accuracy at the top of the ranking are well-suited here since, out of possibly thousands of documents returned for a query, only a few top results are likely to be clicked. While they broaden their fairness definitions beyond statistical parity, metrics proposed in [28] are application-specific, defining fairness in terms of item exposure to users, clickthrough rates, and document relevance.

3 FAIR RANKING PROBLEM FORMULATION

Ranking can have different meanings in different contexts, and ranking models can be trained over various types of ground truth information. Rank predictions can be learned from training data with binary labels (e.g., in bipartite ranking [9]) or discrete labels with ordered classes (i.e., ordinal regression [18] with labels such as "best", "neutral", "worst"). Traditional regression ranks according to continuous scoring functions. Learning-to-rank approaches also include pairwise and listwise models [25].

Therefore, to be widely applicable, we target general rankings with a model-agnostic approach. We assume only that an ordering is imposed over a set of objects X according to some function which assigns each $x_i \in X$ a position relative to all others. Following previous work on error-based criteria for fair classification [7, 8, 13, 15, 17, 23, 27, 31], we formulate the task of auditing the fairness of a ranking as a supervised learning problem in that the true ranking over X is known. [1] Unique to the context of fairness analysis, objects have an associated *protected attribute*, such as gender, age, or race, which partitions the objects into a set of two or more disjoint *groups* $A = \{A_1, \ldots, A_m \mid \cup_{i=1}^{m} A_i = X, A_i \cap A_j = \emptyset, \forall i \neq j\}$. We consider only a single protected attribute, leaving intersectional fairness considerations with multiple protected attributes to future work.

Let $\rho = [x_1 \succ x_2 \succ \ldots \succ x_n]$ be the true ranking over all objects $x_i \in X$, where $\succ$ is an ordering relation on X such that $x_i \succ x_j$ implies that x_i appears at a more preferred position in the ranking than x_j. The number of items in the set is denoted by $|X| = n$. The position of a single object x_i in the true ranking is denoted $\rho(x_i)$. Our task is to evaluate a learned ranking $\hat{\rho}$ according to a given *Fairness Criterion* which relies on a group error function L.

Definition 1. *Given a group error metric $L_{A_i}(\rho, \hat{\rho})$, a* ***Fairness Criterion*** *(FC) is an evaluation rule which designates a ranking $\hat{\rho}$ as fair in relation to a true ranking ρ if:*

$$L_{A_i}(\rho, \hat{\rho}) \cong L_{A_j}(\rho, \hat{\rho}) \;,\; \forall A_i, A_j \in A,\; i \neq j$$

Fairness is evaluated by checking whether the error for each group is similar, or within some threshold, indicated by the symbol $\cong$. The larger the difference in the errors for each group, the more unfair the ranking is considered to be. Our assessment therefore hinges on the choice of a rank-appropriate group error function L.

[1] While a "truly fair" ground truth ranking may not be available, this assumption allows us to distinguish whether a proposed ranking is more or less fair than an alternative.

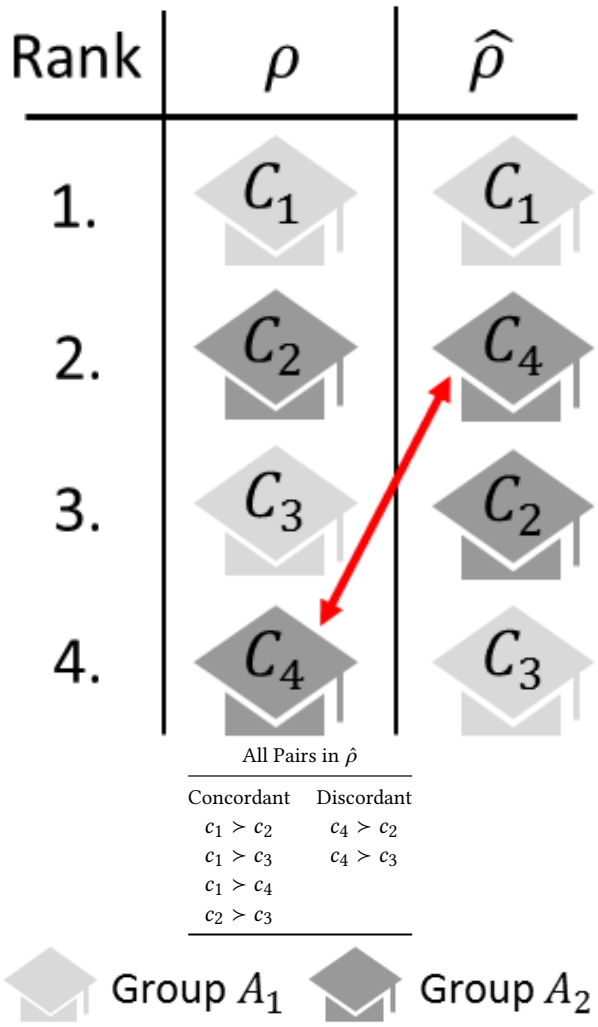


Figure 1: On the left is a true ranking of colleges ρ and predicted ranking $\hat{\rho}$ over two groups of colleges. The resulting discordant and concordant pairs are shown on the right.

4 PROPOSED ERROR METRICS FOR RANK FC

4.1 Pairwise Comparison of Group Errors

To design a general approach for evaluating group error in rankings, we consider foundational approaches [11]. One classic method is to sum the absolute difference in rank position between the true and predicted rankings for each object in the dataset (i.e., to use the Spearman footrule distance). Another popular methodology uses the pairwise error, or Kendall Tau distance [22], by counting the number of inverted pairs of objects in the predicted ranking compared to the true ranking. These two classic approaches to measuring rank similarity have been shown to be *equivalent*, meaning

the Kendall Tau is always within a constant factor of the Spearman footrule distance [12, 24]. Given this insight, the two metrics have been used interchangeably for tasks such as rank aggregation [14, 24]. For fairness assessment, the same reasoning applies in that either rank error metric could be applied. However, since we are concerned with the comparative ranking outcomes for different groups, the pairwise approach provides a natural formulation.

Given the position in the true ranking of a pair of objects where $\rho(x_i) \succ \rho(x_j)$, they are said to be discordant if $\hat{\rho}(x_i) \prec \hat{\rho}(x_j)$. Figure 1 shows the sets of concordant and discordant pairs resulting from an error between two rankings of colleges. We observe that a ranking containing objects from two different groups, A_i and A_j, can be divided into three subsets of pairs: those containing only objects from group A_i, those containing only objects from A_j, and the set of "mixed" pairs containing one object from each group. The total number of unordered pairs in a ranking over X is $\phi(X) = |X|(|X|-1)/2$. The cardinality of the set of mixed pairs is $\phi(X) - \phi(A_i) - \phi(A_j)$. We denote the number of discordant pairs in a set X as $\phi^D(X)$ and concordant pairs as $\phi^C(X)$. The number of mixed pairs favoring objects from one group A_i over another group A_j is denoted by $\phi_{i>j}(X)$. For instance, the cardinality of the set of discordant pairs favoring A_i over A_j is indicated as $\phi^D_{i>j}(X)$.

For simplicity, we henceforth consider two groups. However, rank error based on discordant pairs can easily be extended to multiple groups. For instance, to compute the number of discordant pairs favoring group A_1 given m groups, we compute $\sum_{i=2}^{m} \phi^D_{1>i}(X)$.

4.2 Proposed Rank Equality Criterion

The Equalized Odds criterion for classification measures fairness in terms of the rate at which groups are falsely assigned to the preferred or non-preferred classes. When evaluating a ranking, there are no binary assignments by which to gauge preference. However, *position* in a ranking does indicate a preferred or undesirable outcome - the top of the ranking being analogous to the positive class. When an object is overestimated by the model it is incorrectly assigned a more preferred position than in the true ranking. This is similar in effect to a false positive error made by a classifier. Accordingly, underestimating the position of an object in the ranking incorrectly penalizes it, as in a false negative. Following this principle, we compute the Rank Equality error for group A_i in terms of the number of discordant pairs which favor A_i over items from another group A_j. This proposed metric (Definition 2) captures the number of times that an object from group A_i is incorrectly *overestimated* compared to objects in A_j. The error is then normalized by the total number of mixed pairs ensuring that the error falls in a range of $[0, 1]$. Normalization creates an interpretable measure of preference and accounts for any imbalance in the size of the groups. To apply the Rank Equality FC, we simply compare the *Req* error for each group.

Definition 2. ***Rank Equality Error***

$$Req_{A_i}(\rho, \hat{\rho}) = \frac{\phi^D_{i>j}(X)}{\phi(X) - \phi(A_i) - \phi(A_j)}$$

where $\phi^D_{i>j}(X)$ denotes the number of discordant pairs which favor the target group A_i over A_j, $i \neq j$.

Rank Equality dictates that no group should be unfairly privileged or penalized compared to another group. As an example, consider the rankings shown in Figure 1. To compute the Rank Equality error for group A_2, we count the number of discordant pairs where an item from A_2 is favored over an item from A_1. Four pairs contain an object from each group: $(c_1, c_2), (c_1, c_4), (c_2, c_3), (c_3, c_4)$. One of these pairs (c_3, c_4) is discordant, since $\hat{\rho}(c_4) \succ \hat{\rho}(c_3)$ and $\rho(c_3) \succ \rho(c_4)$, and favors A_2. Thus $Req_{A_2}(\rho, \hat{\rho}) = \frac{1}{4}$.

4.3 Proposed Rank Calibration Criterion

Calibration is used to evaluate probabilistic classifiers in terms of the confidence of the model, using the mean squared error between predicted likelihood of assignment in the positive class and an estimated "true" probability [4]. Applied as an FC, this criterion checks how well the classifier predicts objects in each group. To evaluate the calibration of a ranking $\hat{\rho}$ for a group A_i, we propose to measure error in predicted rank position by counting the number of discordant pairs which contain at least one member of A_i, as given in Definition 3. This captures the overall error made for items in the group. The value is normalized by the total number of pairs containing objects from A_i. For example, the pairs containing objects from group A_2 in Figure 1 are $(c_1, c_2), (c_1, c_4), (c_2, c_3), (c_2, c_4), (c_3, c_4)$. Pairs (c_2, c_4) and (c_3, c_4) are both discordant, therefore following Definition 3, the rank calibration error is $Rcal_{A_2}(\rho, \hat{\rho}) = \frac{2}{5}$.

Definition 3. ***Rank Calibration Error***

$$Rcal\ _{A_i}(\rho, \hat{\rho}) = \frac{\phi^D_i(X)}{\phi(X) - \phi(A_j)}$$

Where $\phi^D_i(X)$ denotes the number of discordant pairs containing at least one object from the target group A_i.

4.4 Proposed Rank Parity Criterion

Finally, we also apply pair inversion to design a statistical parity metric like those explored in previous work on fair ranking [5, 30, 32]. Here, the goal is to ensure fair representation of members of each group among objects given a favorable rank position. We propose to capture this idea by counting the pairs in which one group is favored over the other in the learned ranking, regardless of their positions in the true ranking. We again normalize by the total number of mixed pairs in the learned ranking.

Definition 4. ***Rank Parity Error***

$$Rpar_{A_i}(\rho, \hat{\rho}) = \frac{\phi_{i>j}(X)}{\phi(X) - \phi(A_i) - \phi(A_j)}$$

Where $\phi_{i>j}(X)$ is the number of pairs of objects which favor a member of group A_i over a member of A_j, $i \neq j$.

In Figure 1, two pairs in $\hat{\rho}$ favor group A_2 over group A_1: (c_2, c_3) and (c_4, c_3). Therefore, $Rpar_{A_2}(\rho, \hat{\rho}) = \frac{2}{4}$. This matches our intuition of parity, since the groups are still somewhat evenly distributed through the ranking $\hat{\rho}$ in spite of the incorrect placement of c_4.

4.5 Discussion: Ranking Criteria and Their Interrelationships

We now analyze our metrics to understand their interrelationships and scope of applicability. Given a ranking $\hat{\rho}$ over g groups, there

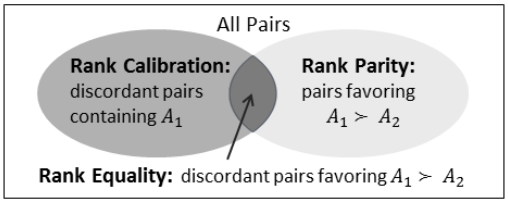


Figure 2: Relationship between the types of pairs used to compute the error for group A_1 (corresponding to Table 1).

are g^2 ways of choosing two objects from the ranking, allowing for group repetition. These pairs may be either *concordant* or *discordant*, resulting in $2g^2$ types of pairs. Table 1 shows the categories of pairs that can be formed for $g = 2$ groups. The colors in the table correspond to the colors in the Venn diagram in Figure 2, illustrating the relationship between the types of pairs used to compute errors for a single group, A_1.

Since our fairness analysis is concerned with the relative error made for each group, within-group concordant pairs are not considered when computing error metrics. All other types of pairs are included in the definition of at least one error metric. Discordant mixed pairs are used to compute all three error metrics. These pairs of objects intuitively capture the *major disparity between groups*: cases where one group is erroneously favored over the other. We define this as Rank Equality. Rank Calibration instead measures the *total error* for each group. This metric counts all pairs containing objects from a single group, capturing within-group as well as across-group errors. Finally, Rank Parity considers the *total advantage* of one group over the other.

FC are compatible in extreme cases. In a perfect prediction there is no error between $\hat{\rho}$ and ρ. In this case $Req = Rcal = 0$ for all groups, since no pairs are discordant. The corresponding FC deem $\hat{\rho}$ fair since the group errors are identical. The *Rpar* error in this case will simply measure the relative advantage of each group in the true ranking. It may be considered fair or unfair depending on the distribution of the objects in each group. Since this is independent of the other metrics, it is therefore possible for a perfect prediction to satisfy all three criteria. In the worst case, $\hat{\rho}$ ranks the objects in the reverse order from ρ. In this case $Rcal = 1$ since all pairs are discordant. Each group is predicted with the same amount of error, therefore by the Rank Calibration FC, $\hat{\rho}$ is considered fair. In this case no pairs are concordant, therefore $Req = Rpar$. Whether the ranking is considered fair according to the corresponding FC again depends on the distribution of groups throughout the ranking.

5 FAIR AUDITING BASED ON RANK ERROR

The FARE Methodology. We next design a non-parametric approach to assess rankings using our proposed error metrics. We want to understand the entire ranking of items from the preferred positions at top of the ranking to the lowest ranked objects. Aggregating the entire treatment of each group using a single fairness score provides only a coarse assessment. Therefore our FARE methodology (for Fair Auditing based on Rank Error) generates *sequences of within-range errors* for each group. The differences in these sequences tell a richer story than would a single value for each group, revealing disparity throughout the entire ranking.

Table 1: Categorization of pairs in $\hat{\rho}$ over two groups A_1, A_2.

Discordant	Concordant
$A_1 \succ {A_1}^D$	$A_1 \succ {A_1}^C$
$A_2 \succ {A_2}^D$	$A_2 \succ {A_2}^C$
$A_1 \succ {A_2}^D$	$A_1 \succ {A_2}^C$
$A_2 \succ {A_1}^D$	$A_2 \succ {A_1}^C$

To start, FARE sorts the data according to the predicted ordering $\hat{\rho}$ and bins it into k subsets $\langle B_1, B_2, \ldots B_k \rangle$. Error metrics are then applied to the objects in each bin. In the case of two groups A_1 and A_2 we evaluate errors $l_{1i} = L_{A_1}(\beta_i, \hat{\beta}_i)$ and $l_{2i} = L_{A_2}(\beta_i, \hat{\beta}_i)$ for the data in each bin B_i. This produces two error sequences: $S_1 = \langle l_{11}, l_{12}, \ldots l_{1k} \rangle$ and $S_2 = \langle l_{21}, l_{22}, \ldots l_{2k} \rangle$. An equi-width binning strategy compares the top-k ranked items across both groups in the first bin, the next k in the next bin, and so on. An equi-depth strategy is also possible, where each bin measures how well the ranking predicts $\frac{|A_i|}{k}\%$ of items from each group.

If the number of bins k is so large that there are only a few objects in each, then the sequence of error measurements may exhibit a high degree of variance. This could exaggerate differences between groups in the case where one is a minority. On the other hand, if bins contain many objects, the result is a rather coarse estimate. To capture the error at a sufficient number of positions throughout the ranking while achieving a reasonable bin size, we adopt a *sliding window approach*. This introduces a smoothing transformation over the data to account for high variance across bins. Each consecutive bin of size w overlaps the previous, offset by a fixed step size $s < w$. The first bin B_1 contains objects $\{x_1, x_2, .., x_w\}$, ordered according to their predicted positions $\hat{\rho}(x_i)$, the second bin $B_2 = \{x_{s+1}, x_{s+2}, \ldots x_{s+w}\}$, and so on. Each error sequence S_i contains $\lfloor \frac{|A_i|}{s} \rfloor$ bins.

Diagnostics for Analyzing Fairness. The next step in the auditing procedure is to compare the error sequences S_1 and S_2 produced by our FARE framework to see if they are similar, and therefore meet the fairness criterion, or if they differ in ways which indicate an unfair ranking. To facilitate this, FARE offers *audit plots*. Similar to reliability diagrams for assessing the calibration of classifiers [4], these visual depictions reveal differences in the shapes, patterns and values of the error sequences. Since our proposed error metrics are normalized, the y-axis on each plot has a fixed range of $[0, 1]$, providing an easily interpretable snapshot view of the error sequences generated during the audit process.

Audit plots are augmented by compact statistics, or *fairness scores*, indicating whether the ranking model satisfies the FC. Conceptually, any diagnostic metrics comparing the sequences can be plugged into the framework. We employ a distance diagnostic $dist(S_1, S_2) = \frac{1}{k} \sum_{i=1}^{k} |l_{1i} - l_{2i}|$ to summarize the similarity of the error sequences as a single value. These scores can be thresholded to flag unfair cases where the average magnitude of error for one group is much larger than the other or to apply strict FC cutoffs.

Complexity. A simple pair counting algorithm can be used to compute each of our proposed error metrics in $O(nlog(n))$ time

using an adaptation of the mergesort algorithm. Performing an audit using the FARE methodology can therefore also be done in logarithmic time, requiring $O(n/s(wlog(w))$ time for step size s and window size w to compute the error sequences for each group. In cases where performance is an issue, we can improve the pair counting procedure to run in $O(n\sqrt{log(n)})$ time [6].

6 EXPERIMENTAL EVALUATION: AUDITING RANK CORRECTION METHODS

We demonstrate the power of our proposed error metrics using FARE to audit post-processing techniques from the literature designed to correct existing rankings. Two state-of-the-art fair ranking methods which enforce statistical parity notions of fairness are compared. Using the FARE methodology, we reveal the resulting tradeoffs between this criterion and the prediction error introduced. We evaluate these fair ranking algorithms using FARE audit plots (Figure 3) and FARE distance diagnostics (Table 2).

State-of-Art Fair Correction Methods. We reproduce a subset of experiments presented by Zehlike et al. [32] using implementation and data made available by the authors."FA*IR" rankings are generated using a greedy algorithm proposed in [32] to create a fair top-k prefix ranking. The rankings target a user-specified minimum proportion of the minority group, subject to a statistical significance test. The proportion is indicated in the method name, e.g., FA*IR2 for 20%. Here we use the same proportions as the authors, chosen to be close to the actual group ratio over the entire dataset.

The "Feldman" method [15] is proposed as pre-processing step for fair classification in which data are ranked. In this method the utility scores for objects in the minority group are adjusted to match the distribution of the majority. We compare these rankings against a baseline of the true ranking with no correction.

Datasets. We evaluate the rank correction methods using two datasets. The Statlog German Credit Dataset [19] provides a "ground truth" ranking of people according to their credit-worthiness for our experiments. Three "fair" rankings are then created using *age* < 25, *age* < 35 and *gender* = female as protected group attributes. Prefix rankings with $k = 100$ are generated. Audit parameters for this dataset are $w = 30, s = 10$. The COMPAS recidivism dataset published by ProPublica in their investigation of racial bias in the criminal justice system is also utilized [2]. This dataset is ranked according to COMPAS scores indicating the likelihood of re-offending for the "true" ranking with $k = 1000$. "Fair" rankings are generated according to groups *race* = African American and *gender* = male. Audit parameters are $w = 100, s = 10$.

Metrics. We produce audit plots using our proposed metrics *Req*, *Rcal* and *Rpar*, and summarize the results using FARE distance diagnostics. In their experiments, Zehlike et al. [32] use a number of metrics to gauge the tradeoff between parity and prediction accuracy. We include two metrics for comparison: *NDCG*: normalized discounted cumulative gain [21] (commonly used in search), and *rank drop*: the maximum number of positions lost by an object.

Discussion. Table 2 summarizes the FARE diagnostics for our experiments. The rankings deemed most fair in this audit are highlighted in bold. Asterisks mark the conclusions which align with the analysis in [32]. For three out of five rankings FA*IR clearly outperforms Feldman, satisfying multiple fairness concerns.

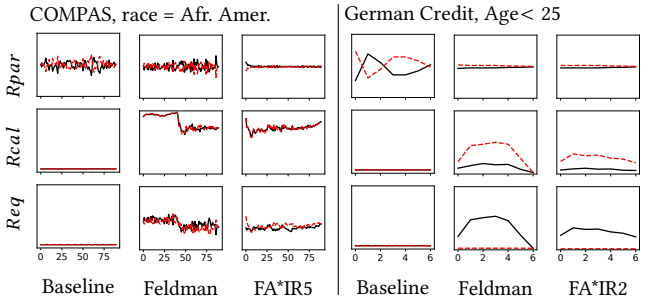


Figure 3: Audit plots for rank correction methods illustrate how errors (plotted on the y-axis and normalized between 0 and 1), manifest throughout the ranking. The x-axis represents the sliding window moving from highly ranked items on the left to the lowest on the right. Errors for group A_1 are shown as a solid black line, group A_2 as dashed red line.

The impact of both the FA*IR and Feldman rank correction techniques on statistical parity concerns is apparent, as measured by our Rank Parity FC. For instance, for the German Credit dataset using *age* < 25, *Rpar* distance is 0.25 in the baseline "true" ranking. Both methods are able to reduce this to 0.03. The degree to which error is introduced as a result of the correction algorithm is reflected in the *Req* and *Rpar* scores. By comparison, the NDCG metric is not sensitive to the rank correction methods, and therefore not expressive enough to capture unfairness. The rank drop values tend to align with the FARE diagnostics. However, this value is not very interpretable. We cannot observe which group had the farthest drop, or whether the position of many items dropped.

For such nuanced analysis we turn to our FARE audit plots, shown in Figure 3. For the German Credit dataset, we can visually discern that both FA*IR and Feldman systematically introduce *Req* and *Rcal* errors that indicate an unfair disparity in the treatment of the groups throughout the ranking. For the COMPAS dataset, we observe similar patterns and magnitude of error throughout the rankings for both groups. In this case, the correction methods are introducing error in a fair manner. Feldman shows a drop midway through the *Rcal* sequence while the FA*IR error appears more consistent. FARE compliments the use of these ranking methods by providing this in-depth view of the treatment of each group.

In the interest of reproducibility, all code, data, and a full analysis including additional diagnostics and datasets is available online:
Examples and analysis: https://github.com/caitlinkuhlman/fare
FARE python package: https://pypi.org/project/fare

7 CONCLUSION

In this work we present the first methodology for auditing rankings using group error metrics which capture popular notions of fairness. Our proposed fairness criteria together with our FARE auditing method comprise a powerful diagnostic tool for nuanced analysis of the treatment of groups being ranked. FARE can be applied in a general and model-agnostic fashion, for many applications where rankings are used to simplify complex socio-economic datasets, providing a crucial service of debunking unfair rankings.

Table 2: Fairness evaluation for rank correction methods. FARE distance diagnostics are shown in the center, and compared to standard error metrics.

			FARE				
Dataset	**Group**	**Method**	*Rpar*	*Rcal*	*Req*	**NDCG**	**Drop**
German Credit k=100	age< 25	Baseline	0.25	0.00	0.00	1.00	0
		Feldman	**0.03**	0.23	0.33	1.00	8
		FA*IR2	**0.03***	**0.18**	**0.27**	1.00	7
German Credit k=100	age< 35	Baseline	0.17	0.00	0.00	1.00	0
		Feldman	**0.04**	0.06	**0.26**	0.99	36
		FA*IR6	**0.04**	**0.04**	0.40	0.99	30
German Credit k=100	gen=f	Baseline	0.33	0.00	0.00	1.00	0
		Feldman	**0.05**	**0.10**	**0.27**	1.00	8
		FA*IR7	0.08	0.11	0.28	1.00	0
COMPAS k=1000	race	Baseline	0.09	0.00	0.00	1.00	0
		Feldman	0.08	**0.01**	0.08	0.98	393
		FA*IR5	**0.02***	**0.01**	**0.04**	0.99	319
COMPAS k=1000	gen=m	Baseline	0.13	0.00	0.00	1.00	0
		Feldman	0.09	0.03	0.09	1.00	294
		FA*IR8	**0.02***	**0.01**	**0.03**	1.00	161

ACKNOWLEDGMENTS

The authors thank the Computing Resources Association for Women for support through the CREU program. This work was also partially funded by NSF IIS-1815866, IIS-1560229, IIS-1815866 and US Department of Education GAANN Fellowship P200A150306.

REFERENCES

[1] Ifeoma Ajunwa, Sorelle Friedler, Carlos E Scheidegger, and Suresh Venkatasubramanian. 2016. Hiring by algorithm: predicting and preventing disparate impact. (2016).
[2] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. 2016. Machine bias. *Pro Publica* (2016).
[3] Solon Barocas and Andrew D Selbst. 2016. Big data's disparate impact. *Cal. L. Rev.* 104 (2016), 671.
[4] Rich Caruana and Alexandru Niculescu-Mizil. 2004. Data mining in metric space: an empirical analysis of supervised learning performance criteria. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 69–78.
[5] L Elisa Celis, Damian Straszak, and Nisheeth K Vishnoi. 2017. Ranking with fairness constraints. *arXiv preprint arXiv:1704.06840* (2017).
[6] Timothy M Chan and Mihai Pătraşcu. 2010. Counting inversions, offline orthogonal range counting, and related problems. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 161–173.
[7] Alexandra Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data* 5, 2 (2017), 153–163.
[8] Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. 2017. Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 797–806.
[9] Corinna Cortes and Mehryar Mohri. 2004. AUC optimization vs. error rate minimization. In *Advances in neural information processing systems*. 313–320.
[10] Jeffrey Dastin. 2018. Amazon scraps secret AI recruiting tool that showed bias against women. *Reuters* (2018). https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G
[11] Persi Diaconis. 1988. Group representations in probability and statistics. *Lecture Notes-Monograph Series* 11 (1988), i–192.
[12] Persi Diaconis and Ronald L Graham. 1977. Spearman's footrule as a measure of disarray. *Journal of the Royal Statistical Society. Series B (Methodological)* (1977), 262–268.
[13] Cynthia Dwork, Nicole Immorlica, Adam Tauman Kalai, and Mark DM Leiserson. 2018. Decoupled Classifiers for Group-Fair and Efficient Machine Learning. In *Conference on Fairness, Accountability and Transparency*. 119–133.
[14] Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. 2001. Rank aggregation methods for the web. In *Proceedings of the 10th International Conference on World Wide Web*. ACM, 613–622.
[15] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 259–268.
[16] Malcolm Gladwell. 2011. The order of things. *The New Yorker* 87, 1 (2011), 68–75.
[17] Moritz Hardt, Eric Price, Nati Srebro, et al. 2016. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*. 3315–3323.
[18] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. 1999. Support vector learning for ordinal regression. (1999).
[19] PDH Hofmann. 1994. Statlog (German Credit Data) Data Set. *UCI Repository of Machine Learning Databases* (1994).
[20] Mikella Hurley and Julius Adebayo. 2016. Credit scoring in the era of big data. *Yale JL & Tech.* 18 (2016), 148.
[21] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.
[22] Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika* 30, 1/2 (1938), 81–93.
[23] Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. 2017. Inherent Trade-Offs in the Fair Determination of Risk Scores. In *Proceedings of the 8th Conference on Innovation in Theoretical Computer Science*.
[24] Ravi Kumar and Sergei Vassilvitskii. 2010. Generalized distances between rankings. In *Proceedings of the 19th International Conference on World Wide Web*. ACM, 571–580.
[25] Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 3, 3 (2009), 225–331.
[26] Cathy O'Neil. 2017. *Weapons of math destruction: How big data increases inequality and threatens democracy*. Broadway Books.
[27] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. 2017. On fairness and calibration. In *Advances in Neural Information Processing Systems*. 5680–5689.
[28] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of Exposure in Rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2219–2228.
[29] Annie Waldman and SiSi Wei. 2015. Colleges Flush With Cash Saddle Poorest Students With Debt. *Pro Publica* (2015).
[30] Ke Yang and Julia Stoyanovich. 2017. Measuring fairness in ranked outputs. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*. ACM, 22.
[31] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. 2017. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th International Conference on World Wide Web*. ACM, 1171–1180.
[32] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. 2017. Fa* ir: A fair top-k ranking algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1569–1578.

Adversarial Adaptation of Scene Graph Models for Understanding Civic Issues

Shanu Kumar
Indian Institute of Technology Kanpur, India
sshanu@iitk.ac.in

Shubham Atreja
IBM Research AI, India
satreja1@in.ibm.com

Anjali Singh
IBM Research AI, India
ansingh8@in.ibm.com

Mohit Jain
IBM Research AI, India
mohitjain@in.ibm.com

ABSTRACT

Citizen engagement and technology usage are two emerging trends driven by smart city initiatives. Typically, citizens report issues, such as broken roads, garbage dumps, *etc.* through web portals and mobile apps, in order for the government authorities to take appropriate actions. Several mediums – text, image, audio, video – are used to report these issues. Through a user study with 13 citizens and 3 authorities, we found that image is the most preferred medium to report civic issues. However, analyzing civic issue related images is challenging for the authorities as it requires manual effort. In this work, given an image, we propose to generate a *Civic Issue Graph* consisting of a set of objects and the semantic relations between them, which are representative of the underlying civic issue. We also release two multi-modal (text and images) datasets, that can help in further analysis of civic issues from images. We present an approach for adversarial adaptation of existing scene graph models that enables the use of scene graphs for new applications in the absence of any labelled training data. We conduct several experiments to analyze the efficacy of our approach, and using human evaluation, we establish the appropriateness of our model at representing different civic issues.

KEYWORDS

Civic Engagement, Scene Graph Generation, Adversarial Adaptation, Smart Cities, Intelligent Systems on Web

ACM Reference Format:
Shanu Kumar, Shubham Atreja, Anjali Singh, and Mohit Jain. 2019. Adversarial Adaptation of Scene Graph Models for Understanding Civic Issues. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313681

1 INTRODUCTION

In recent years, there has been a significant increase in smart city initiatives [8, 26, 27]. As a result, government authorities are emphasizing the use of technology and increased citizen participation for better maintenance of urban areas. Various web platforms – SeeClickFix [23], FixMyStreet [1], ichangemycity [13] – have been introduced across the world, which enable the citizens to report civic issues such as poor road condition, garbage dumps, *etc.*, and track the status of their complaints. This has resulted in exponential increase in the number of civic issues being reported [2]. Even social media sites (Twitter, Facebook) have been increasingly utilized to report civic issues. These issues are reported online through various mediums – textual descriptions, images, videos, or a combination of them. Previous work [9] highlights the importance of different mediums in citizen participation. Yet, no prior work has tried to understand the role of these mediums in reporting of civic issues.

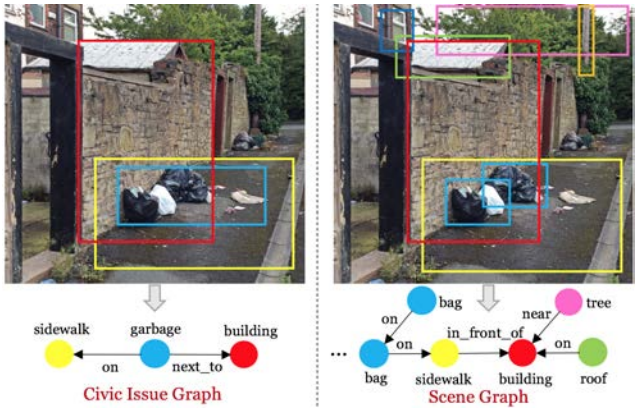


Figure 1: Comparison between Civic Issue Graph and Scene Graph for a given image. The scene graph provides a complete representation of all objects and relationships in the image, while the Civic Issue Graph only consists of those objects and relations which are representative of the civic issue.

In this work, we first identify the most preferred medium for reporting civic issues, by conducting a user study with 13 citizens and 3 government authorities. Using the 84 civic issues reported by the citizens using our mobile app, and follow-up semi-structured interviews, we found that images are the most usable medium for the citizens. In contrast, authorities found text as the most preferred medium, as images are hard to analyze at scale.

To fill this gap, several works have proposed methods to automatically identify a specific category of civic issues from images, such as garbage dumps [24] and road damage [20]. However, their methods are limited to the specific categories that they address.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313681

Furthermore, existing holistic approaches of analyzing civic issues are limited to text [4]. To this end, we propose an approach to understand various civic issues from input images, independent of the type of issue being reported.

In the field of image understanding, scene graphs [14], are used to get a complete representation of all objects in an image along with the relations between them. However, to understand a civic issue, only certain crucial objects and the relations between them need to be detected, which are representative of the civic issue in the image. Inspired from the task of scene graph generation, we propose to generate *Civic Issue Graphs* that provide complete representations of civic issues in images.

Figure 1 shows a comparison between the two representations. In contrast to the scene graph, the Civic Issue Graph only consists of objects conveying a civic issue, their bounding boxes, and the relation between these objects. We present a formal definition of this representation in Section 5.

Training a scene graph model requires a large amount of data consisting of images with grounded annotations (objects and relations in the images). Due to the lack of sufficient annotated images of civic issues, we use an existing scene graph model in a cross-domain setting, with partially annotated and unpaired data. We utilize a dataset extracted by collating and processing images from civic issue complaint forums, for training our model, and make this dataset publicly available. We present a novel adversarial approach that uses an existing scene graph model for a new task in the absence of any labelled training data. We conduct various experiments to establish the efficacy of our approach using metrics derived from standard scene graph evaluation metrics. Finally, through human evaluation, we demonstrate that civic issues from images can be appropriately represented using our Civic Issue Graph.

To summarize, the major contributions of this paper are: (i) understanding the usability of different mediums for reporting civic issues, (ii) introducing an unsupervised mechanism using adversarial adaptation of existing scene graph models to a new domain, (iii) experimental evaluation indicating significant performance gains for identification of civic issues from user uploaded images, and (iv) releasing two multi-modal (text and image) datasets with information on civic issues, to encourage future work in this domain.

2 RELATED WORK

Civic Issue Detection and Analysis. Social media provides a convenient interface that allows citizens to report civic issues [3, 15]. Several works try to analyze online platforms to automatically mine issues related to civic amenities [22, 25], but the analysis is limited to textual descriptions. Specific to images, [20] and [24] use object detection and image segmentation techniques to identify road damage and garbage dumps respectively, from input images. However, their methods are also limited to the specific category of civic issues that they address. One of the more recent works, 'Citicafe' [4] employs machine learning techniques to understand and analyze different types of civic issue from the user input. However, they do not provide a method for understanding images reporting civic issues, as we do in this paper.

Scene-Graph Generation. Several works [16, 33] propose methods for generating scene graphs from images to represent all objects in an image and the relationships between them. Zellers et al. [33] present the state-of-the-art for generating scene graphs by establishing that several structural patterns exist in scene-graphs (which they call motifs) and showing how object labels are highly predictive of relation labels by analyzing the Visual Genome dataset [17]. However, their approach requires a large set of images for training with grounded annotations for objects and relations. Some works [19, 34] utilize zero shot learning for generating a scene-graph. However, their results show that the learning is restricted to the task of detecting new predicates which were not seen during the training phase. Our approach can be used to generalize existing scene graph models to predict new relations belonging to a different domain, which are absent from the training data.

Domain Adaptation. Domain adaptation is a long studied problem, where approaches range from fine-tuning networks with target data [28] to adversarial domain adaptation methods [31]. Adversarial methods [10, 29, 31] employ a domain classifier to learn mappings from the source domain to target domain, and these mappings are used to generalize the model to the target domain. These methods have shown promising results for image understanding tasks such as captioning [5] and object detection [7]. In this work, we propose to use Adversarial Discriminative Domain Adaptation (ADDA) [31] for adapting scene graph models to our new task.

3 USER STUDY

We conducted a user study to understand the preference of different mediums – text, audio, image and video – to report civic issues, both from citizens and authorities perspective. For this, we developed a custom Android app, with the landing page having four buttons, each corresponding to the four mediums. To report an issue, any of the medium(s) could be used any number of times, *e.g.,* a report can comprise of 1 video, few lines of text, and 2 images.

13 participants (9 male, 4 female, age=28.5±6.1 years) reported civic issues over a period of 7-10 days. All the participants were recruited using word-of-mouth and snowball sampling. All of them were experienced smartphone users, using it for the past 6.2±2.2 years, and well educated (highest education: 1 high school, 3 Bachelors, 6 Masters, 4 PhDs). However, only two of them have previously reported civic issues on online web portals. At the end of the study, a 30-mins semi-structured interview was conducted, to delve deeper into the reasons for (not) using specific medium(s).

Furthermore, we interacted with 3 government authorities (3 male, age = 35-45 years) for 30-mins each, to understand their perspective on the medium of the received complaints. All interviews were audio-recorded, and later transcribed for analysis.

Results: Overall, 84 (6±3.7) civic issues were reported by the 13 participants, mainly in the category of garbage (11/13 participants), potholes (9), blocked sidewalk (6), traffic (5), illegal car parking (3), and stray dogs (3). 81 of these issues consisted of image, text, or their combination, while only 2 had audio and 1 had video. Hence, we only focus on image and text as preferred mediums.

A majority of the participants (10/13) found images the best for reporting civic issues, followed by text (2/13) and video (1/13). Images were preferred primarily as it is quick and easy to click an image, and they convey a lot of information: "*An image is worth 1000 words.*"-P_4, "*its super quick to take pics... even when I pause at a traffic signal, I can take a pic*"-P_{10}. Participants also felt that images are best for conveying the severity of an issue. Interestingly, participants thought that images can "*act as a proof of the problem... images don't lie*"-P_6. Participants felt that people might '*bluff*'/'*exaggerate*' when reporting issues using text. However, participants complained that images can not be used to capture the temporal variations of civic problem, *e.g.*, "*images can't say tell you for how long the garbage has been lying there*"-P_2. For reporting the temporal variations, participants favored text. But participants also found that texting requires more time and effort, compared to clicking images.

When participants were asked to choose the best combination of mediums for reporting civic issues, majority of them (9/14) chose image with text. The combination allows them the freedom to show severity and truthfulness of the issue using image(s), along with adding other details in text.

Following this, we interviewed 3 government authorities, and found about the process of human annotators analyzing the received images to generate captions describing the issue. These captions are then passed on to the relevant authority in writing or via phone calls to take appropriate actions. Also the authorities confirm that a majority of the received complaints comprise of images. However, these images never reach them due to lack adequate technological infrastructure. This confirms that image is the most preferred medium for users, but authorities rely only on textual complaints. To bridge this gap, in this work, we generate text-based descriptions of images that are used for reporting civic issues.

4 DATASET

An extensive dataset of images with annotations for a wide variety of civic issues is currently unavailable. To this end, we mined 485,927 complaints (with 131,020 images) from two civic issue reporting forums – FixMyStreet [1] and ichangemycity [13]. We use them to generate two datasets.

Dataset-1 consists of human-annotated images with the bounding boxes and object labels for 4 object categories – *garbage, manhole, pothole, water logging*. Some of these object categories are not present in any publicly available image datasets. We utilize the annotations from two existing datasets for garbage [24] and potholes [20], and add new images representative of the new object categories along with their annotations, to build Dataset-1, with a total of 1505 images and 2302 bounding boxes.

Dataset-2 consists of examples of Civic Issue Graphs, represented through triples of the form $[object1, predicate, object2]$, specifying the relationship ($predicate$) between a pair of objects ($object1$ and $object2$). We use natural language processing techniques [21, 30] to extract these triples from complaint descriptions. We manually define a set of 19 target object categories which are relevant to the civic domain and map the objects from these triples to our set of target objects based on semantic similarity [12]. We retain only those triples where the predicate defines positional relations (manually determined) and for which both objects are matched with a similarity value greater than 0.4. This dataset consists of 44,353 Civic Issue Graphs, where 8204 are paired with images. There are total 5799 unique relations with 19 object classes and 183 predicate classes. These datasets can be found here[1].

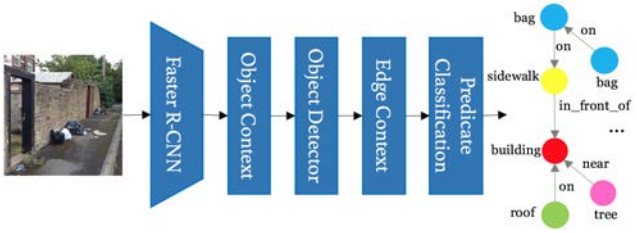


Figure 2: An overview of the MotifNet model

5 CIVIC ISSUE GRAPH GENERATION

We now present our approach for understanding civic issues from input images. We first present the formal definition of Civic Issue Graphs, followed by our detailed approach, consisting of scene graph generation and adversarial domain adaptation.

Formal Definition: A scene graph is a structured representation of objects and the relationships present between them in an image. It consists of triples or relations (used inter-changeably) of the form $[object1, predicate, object2]$ where $predicate$ defines the relationship between the two objects and both $object1$ and $object2$ are grounded to their respective bounding box representations in the image. While a scene graph provides a complete representation of the contents of the scene in an image, our proposed Civic Issue Graph (CG) only consists of objects conveying a civic issue, their bounding boxes, and the predicate between these objects. We use the following notations to define a CG:

- $B = \{b_1, \ldots, b_n\}$: Set of bounding boxes $b_i \in \mathbb{R}^4$;
- $O^1 = \{o_1^1, \ldots, o_n^1\}$: Set of objects essential for defining a civic issue, *e.g.*, '*pothole*', '*garbage*', *etc.*
- $O^2 = \{o_1^2, \ldots, o_n^2\}$: Set of objects that define the context of objects in O^1, *e.g.*, '*street*', '*building*', *etc.*
- $O = O^1 \cup O^2$: Set of all objects that assign a class label $o_i \in O$ to each b_i
- $P = \{p_1, \ldots, p_n\}$: Set of predicates defining geometric or position-based relationships between $o_i^1 \in O^1$ and $o_i^2 \in O^2$, *e.g.*, '*above*', '*next_to*', '*in*', *etc.*
- $R_{CG} = \{r_1, \ldots, r_n\}$: Set of CG relations with nodes $(b_i, o_i^1) \in B \times O^1$, $(b_j, o_j^2) \in B \times O^2$, and predicate label $p_{i \to j} \in P$, *e.g.*, $[garbage, on, street]$, where *'garbage'* $\in O^1$, *'street'* $\in O^2$, and *'on'* $\in P$

5.1 Scene Graph Generation

The MotifNet model, proposed by Zellers et al. [33], is the current state-of-the-art for generating scene graphs and we utilize this model for demonstrating our approach. However, the approach is generic and can be applied to other models with similar architecture.

[1] https://github.com/Sshanu/civic_issue_dataset

MotifNet Model: Fig 2 presents a high-level overview of the MotifNet model. The model consists of an *Object Detector* that provides object labels and their bounding regions in the input image. Following this, the *Object Context* module uses bidirectional LSTMs to generate contextualized representation for each object, and this representation is used by the *Object Decoder* to predict the final object labels. The *Edge Context* module then generates the contextualized representation for each object pair using additional bidirectional LSTMs. Finally, *Predicate Classification* contains a softmax layer to identify the predicate label, using the object-pair representation as input. The reader may refer to the extended version of this work [18] or the original MotifNet paper [33] for a detailed explanation about the individual modules and their mathematical formulations.

5.2 Adversarial Domain Adaptation

Domain adaptation involves using an existing model trained on 'source' domain where labelled data is available, and generalizing it to a 'target' domain, where labelled data is not available. Domain adaptation has been helpful for tasks such as image captioning [6] that require a large corpora of images and their labels, as getting this data for each and every domain is unfeasible. More recently, adversarial methods for domain adaptation [31] have also been proposed, where the training procedure is similar to the training of Generative Adversarial Networks (GANs) [11]. We present an adversarial training approach for a scene graph model, which, to the best of our knowledge, has not been explored before. Domain adaptation for scene graphs is challenging due to the large domain shift in the images as well as the feature space of relations. For instance, the Visual Genome dataset (VG) [17] used for training scene graph models, consists of a mix of indoor and outdoor scenes with more object instances, whereas our dataset of civic issues consists of specific outdoor scenes depicting a civic issue. Moreover, some of the relations observed in the civic issue domain are not even present in the visual genome dataset (e.g., $[garbage, on, street]$). In the following subsections, we provide more details about our cross-domain setting followed by our approach for adversarial domain adaptation.

5.2.1 Cross-Domain Setting. Scene graph models trained on a particular dataset can detect only those relations that are already *seen* by the model, or in other words, present in the training dataset. For our task of generating *CG*, the model needs to detect R_{CG}, *i.e.*, the set of relations contained in *CG*. Note that the set of relations in R_{CG} can be further divided into R_s and R_n, where R_s is the set of relations previously *seen* by the model, *e.g.*: $[tree, over, fence]$ and R_n is the set of relations previously *unseen* by the model *e.g.*: $[garbage, on, street]$. In the absence of any labelled data for R_n, we want to generalize the model trained on R_s, to adapt to R_n as well.

5.2.2 Adversarial Approach. Adversarial approach for domain adaptation consists of two models – a pre-trained generator model and a discriminator model. In our setting, we use the MotifNet model pre-trained on VG dataset as the generator and propose a discriminator model that can distinguish between R_s and R_n. During pre-training, the MotifNet model learns a representation for the object pairs which is used to predict the final set of relations (R_s). Without adversarial training, the model has not learned the representation for any *unseen* pair of objects from the civic domain and will not be able to predict such relations (R_n). Therefore, during adversarial training, the objective of the MotifNet model is to learn a mapping of target object pairs (*unseen*) to the feature space of the source object pairs (*seen*). This objective is supported via the discriminator, which is a binary classifier between the source and target domains. The MotifNet model can be said to have learned a uniform representation of object pairs corresponding to R_s and R_n, if the classifier trained using this representation can no longer distinguish between R_s and R_n. Therefore, we introduce two constrained objectives which seek to – i) find the best discriminator model that can accurately classify R_s and R_n, and ii) "maximally confuse" the discriminator model by learning new mapping for R_n. Once the source and target feature spaces are regularized, the predicate classifier trained on the *seen* object pairs can be directly applied to *unseen* object pairs, thereby eliminating the need for labelled training data.

Fig 3 summarizes our adversarial training procedure. We first pre-train the MotifNet model on the VG dataset using cross-entropy loss and then update it using adversarial training. During adversarial training, the parameters for the MotifNet model and the discriminator are optimised according to a constrained adversarial objective. To optimize the discriminator model, we use the standard classification loss (L_d). In order to optimize the MotifNet model, we use the standard loss function (L_a) with inverted labels (seen $\rightarrow$ unseen, unseen $\rightarrow$ seen) thereby satisfying the adversarial objective. This entire training process is similar to the setting of GANs. We iteratively update the MotifNet model and the Discriminator with a ratio of N_m:N_d with $N_m < N_d$, *i.e.*, the Discriminator is updated more often than the MotifNet model. We now provide a mathematical formulation of our training approach.

Discriminator We define the Discriminator as a binary classifier with *seen* and *unseen* as the two set of classes. For each object pair (o_i, o_j), the Discriminator is provided with two inputs: 1) $\mathbf{g}_{i,j}$: final representation of the object pair generated by the model and 2) $(\mathbf{W}_h \mathbf{d}_i) \circ (\mathbf{W}_t \mathbf{d}_j)$: contextualized representation of the object pair without the visual features. We further experimented with different inputs to the discriminator (details provided in the extended version of this work [18]). The Discriminator consists of 2 fully connected layers, followed by a softmax layer to generate probability $C_d(l|o_i, o_j)$, where $l \in \{seen, unseen\}$. The mathematical formulation of the discriminator for a given object pair (o_i, o_j) is:

$$\mathbf{F}_{i,j} = \mathbf{Dis}([\mathbf{g_{i,j}}; (\mathbf{W_h d_i}) \circ (\mathbf{W_t d_j})]) \tag{1}$$

$$\mathbf{C}_d = softmax(\mathbf{W}_d \mathbf{F}_{i,j} + \mathbf{b}_d) \tag{2}$$

Training Discriminator Let OP_{cv} be the set of all object pairs identified by the model for an image, I_{CV} belonging to the civic domain.

$$OP_{cv} = MotifNet(\mathbf{I_{cv}})$$

The goal of the Discriminator is formulated as a supervised classification training objective:

$$\mathcal{L}_d(\theta_d) = -\sum_{OP_{cv},\ n=1}^{N} \log \mathbf{C}_d(l^n|y^n) \tag{3}$$

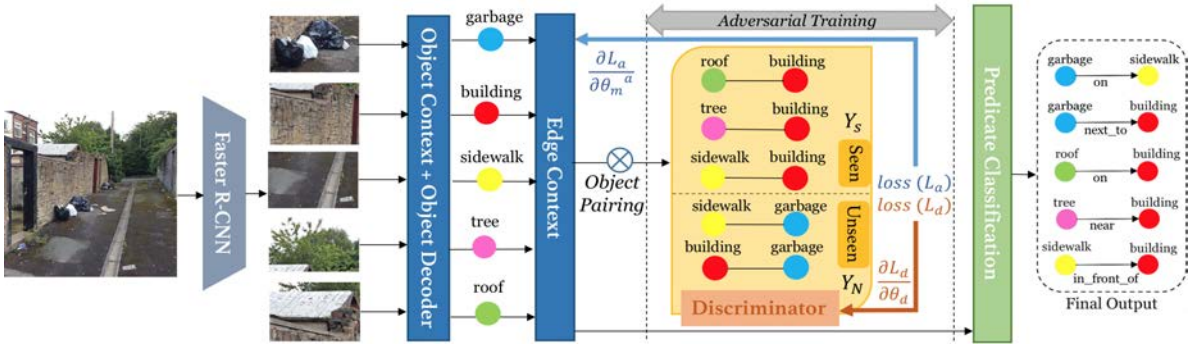


Figure 3: An illustration of our model: Faster R-CNN provides the object labels and their bounding regions. Object context generates a contextualized representation for each object. Edge context generates a contextualized representation for each edge using the representation of the object pairs. During *adversarial training*, information regarding the edge context is passed on to the Discriminator, which learns to distinguish between the *seen* and *unseen* object pairs. The training objective of the Discriminator results in gradients flowing into the Discriminator as well as the edge context layer. The loss for the model decreases as the model learns to fool the Discriminator by adapting a uniform representation for *seen* and *unseen* classes.

$$l^n = \begin{cases} 1(seen) & \text{if } y^n \in Y_s \\ 0(unseen) & \text{if } y^n \in Y_n, \end{cases}$$

where $y^n = (o_i, o_j)^n$, and Y_n and Y_s are the set of object pairs corresponding to R_n and R_s, respectively. θ_d denotes the parameters of the Discriminator to be learned. We minimize $\mathcal{L}_d$ while training the discriminator.

Training Model In accordance with the inverted label loss described above, the training objective of the model is as follows:

$$\mathcal{L}_a(\theta_m^a) = -\sum_{OP_{cv},\, n=1}^{N} \log \mathrm{C}_d(l^n|y^n) \quad \forall y^n \in Y_n$$

Here $\theta_m^a : \{\mathbf{W}_h, \mathbf{W}_t\}$ denotes the parameters of the model that are updated during adversarial training. We minimize $\mathcal{L}_a$ while updating the model.

6 EXPERIMENTS AND EVALUATION

The simplest approach to identify the civic issue from an image is to classify the image into predefined categories. To this end, we trained a classifier using a set of 10 most frequent categories as defined on FixMyStreet forum. On the test dataset, while the accuracies for the three most accurate classes were 86.5%, 83.6% and 75.2%, 4 out of 10 classes had their accuracy less than 17%. Such large variation in accuracies for different classes indicates that classifying images into pre-defined categories is not sufficient. For more details of the classifier, please refer to the extended version of this work [18]. We now provide the implementation details of our model.

6.1 Implementation Details

Discriminator used in adversarial training consists of 3 fully connected layers: two layers with 4096 hidden units followed by the final softmax output. Each hidden layer is followed by a batch normalization, leakyReLU activation function with negative slope of 0.2 and apply a dropout in the training phase with keeping probability of 0.5. Both discriminator and model are trained using ADAM optimizer with a learning rate of 1.2 x 10^{-2} and 1.2 x 10^{-3}, respectively. The value of N_d is set to 150 steps, while N_m is set to 50 steps, with the model and the discriminator being trained iteratively for 12 epochs. For complete implementation details regarding the pre-training of the scene graph model and the Faster R-CNN training, please refer to the extended version of this work [18].

6.2 Evaluation Metrics

Previous work [32] defines three modes for analyzing a scene graph model: 1) Predicate Classification (PredCls) which measures the model's performance for detecting the predicate, given a set of object pairs, in isolation from other factors, 2) Scene Graph Classification (SGCls), which measures the model's performance for predicting the right object labels and predicates, given a set of localized objects, and 3) Scene Graph Generation (SGGen), which requires the model to simultaneously detect the set of objects and predict the right predicate for each object pair. For our approach of generating *CG* using existing scene graph models, it is appropriate to report: (i) the performance of the existing model when generalized to this new domain, and (ii) the accuracy of the output *CG* for representing the civic issue in the image. Deriving from the existing set of tasks, we define a new set of tasks which can help in evaluating our model along these dimensions:

- **OPCls**: the task is to predict the set of object pairs which are indicative of the civic issue present in the image.
- **CGCls**: the task is to predict the set of relations which can represent the civic issue present in the image.
- **CGGen**: the task is to simultaneously detect the region in the image and predict the relations indicative of the civic issue.

For task **OPCls**, we report the experimental results, and use human evaluation for the task **CGCls** and **CGGen**. In accordance with previous work, for **OPCls**, we report results for the image-wise recall metrics (R@k). Since our task is to predict object pairs which are found in civic domains, we report results for *R*@1, *R*@5, *R*@10 & *R*@20 metrics. For **CGCls** and **CGGen**, we report the results using both Precision and Recall metrics ($k : \{1, 3, 5\}$)



Figure 4: Qualitative examples presenting the Civic Issue Graphs generated by our model. We show the top 3 relations and highlight the ones that are representative of the civic issue along with their bounding regions

6.3 Experimental Results

6.3.1 Removing Object Decoder. The MotifNet model after adversarial training performed poorly when tested on the images from civic domain ($R@10 = 10.5$). We found that the object decoder is not able to predict the correct object labels when the input image contains new objects from the civic domain, as the model has not been trained on these labels. On removing the decoder during test time (denoted as $MotifNet_{Adv}$ in the table), the performance improves significantly ($R@10 = 76.0$, Table 1). Adapting the decoder to a new domain requires ground-truth data in terms of the sequence of objects and the labels, which may not be possible for the civic domain. Therefore, we decided to pre-train the MotifNet model without the decoder (denoted by $MotifNet^{wd}$) and directly use the object labels predicted by the Faster R-CNN. On updating the new model using adversarial training (denoted by $MotifNet^{wd}_{Adv}$), the performance improved significantly, particularly for $R@1$ and $R@5$. Table 1 shows the comparison between the different settings with $MotifNet^{wd}_{Adv}$ performing significantly better than all other models, for all the metrics.

6.3.2 Adversarial Training vs Fine-tuning. The results so far show that using adversarial training can significantly improve the performance of the model. As an alternative approach, we also try to adapt the pre-trained model to our new domain by fine-tuning the predicate classification in the model. Mathematically, we aim to increase the value of $\mathbf{P}(p_{i\to j}|B, O))$, where $(o_i, p_{i\to j}, o_j)$ correspond to R_{CG}. The training objective for this phase is defined as:

$$\mathcal{L}_f(\theta^f_m) = -\sum_{OP_{cv},\ n=1}^{N} l \log \mathbf{P}(p^n_{i\to j}|B^n, O^n) \tag{4}$$

$$l = \begin{cases} 1 & \text{if } p^n_{i\to j} \in R_{CG} \\ 0 & \text{otherwise,} \end{cases}$$

where $\theta^f_m : \{\mathbf{W}_r, \mathbf{w}_{o_i,o_j}\}$, *i.e.*, the weights and bias of the predicate classifier of the model. We minimize L_f while fine-tuning the model which is trained for 6 epochs. Table 1 shows that fine-tuning a pre-trained MotifNet model ($MotifNet^{wd}_{fine}$) brings slight improvement in the performance when compared to the original model ($MotifNet$). However, the model with adversarial training ($MotifNet^{wd}_{Adv}$) performs significantly better than the fine-tuned model ($MotifNet^{wd}_{fine}$). Fine-tuning the model will only improve the detection of relations which are already *seen* by the model, while adversarial training will generalize the performance across both *seen* and *unseen* classes. Further fine-tuning the adversarially updated MotifNet ($MotifNet^{wd}_{Adv+fine}$) model brings no improvement in the performance.

Model Settings	R@1	R@5	R@10	R@20
MotifNet	35.6	64.9	75.4	79.7
$\text{MotifNet}_{\text{Adv}}$	37.7	65.7	76.0	79.8
$\text{MotifNet}^{\text{WD}}$	37.7	63.0	73.3	78.9
$\text{MotifNet}^{\text{WD}}_{\text{Adv}}$	**43.3**	**67.7**	**76.3**	**80.2**
$\text{MotifNet}^{\text{WD}}_{\text{FINE}}$	38.9	63.6	73.8	79.2
$\text{MotifNet}^{\text{WD}}_{\text{Adv+FINE}}$	43.1	67.7	76.3	80.2

Table 1: Recall for different settings; *Adv*: Adversarial Training; *fine*: fine-tuning; *wd*: *without decoder* setting

6.4 Human Evaluation

To establish the efficacy of our model at appropriately representing civic issues from images, we recruited Amazon Mechanical Turk workers. We randomly sampled 300 images from the test set; each image was evaluated by 3 workers. In accordance with our definition of *CG*, we retained only the top 5 relations from the output relations for which $o_i \in O_1$ and $o_j \in O_2$, where (o_i, o_j) denotes the unordered pair of objects in a relation.

The evaluation was carried out for the tasks **CGCLS** and **CGGEN** in two phases. For **CGCLS**, workers were shown an image along with 5 relations and were asked to select 0 or more relations that appropriately represented the civic issue(s) in that image. An option was also given to specify any additional relations separately. For **CGGEN**, we retrieved the relations which were marked as relevant for a given image. For each such relation, workers were shown the bounding regions for the objects present in the relation, and asked to evaluate the coverage of these bounding regions, on a scale of 0-10. We report two metrics – Precision and Recall, for both the tasks and consider only the majority voted relations with a minimum average rating of 5 for the bounding regions. Table 2 shows the performance of our model. The results show that 83.3% of the times, the relation representing a civic issue is present in the top 3 relations of our *CG*, and 53.0% of the times, the top relation itself represents a civic issue in the image. The accuracy on the **CGGEN** task further indicates that our model is capable of generating accurate groundings for the objects representing the civic issue.

7 CONCLUSION

We introduce a novel unsupervised mechanism of adapting existing scene graph models via adversarial training and present an application of our approach for generating Civic Issue Graph. Our approach can be utilized in existing platforms, which allow users to report civic issues using images. Once the user uploads an image,

	CGCLS			CGGEN		
	@1	@3	@5	@1	@3	@5
Precision	53.0	31.9	24.7	50.9	30.3	24.1
Recall	53.0	84.0	99.0	50.9	83.3	99.0

Table 2: Precision and Recall values for the tasks CGCLS and CGGEN based on human evaluation

our model can automatically generate text-based relations (*e.g.,* garbage-on-street, garbage-next to-building) depicting the civic issue in the input image. These text-based relational descriptions can be shared with the authorities, which can be utilized for large scale analysis, thereby automating the process and removing any dependency on the actual image uploaded by the user. Furthermore, if needed, natural language descriptions can be generated from these relations using a template-based approach. Our experimental analysis helps provide a framework for adapting scene graph models to other settings as well. We also release two multi-modal (text and images) datasets with information of civic issues, to encourage future work in this domain.

REFERENCES

[1] 2016. FixMyStreet. (2016). https://www.fixmystreet.com/

[2] 2017. Mayor's Management Report. https://www1.nyc.gov/assets/operations/downloads/pdf/mmr2017/2017_mmr.pdf. (2017).

[3] Deborah Agostino. 2013. Using social media to engage citizens: A study of Italian municipalities. *Public Relations Review* 39, 3 (2013), 232–234.

[4] Shubham Atreja, Pooja Aggarwal, Prateeti Mohapatra, Amol Dumrewal, Anwesh Basu, and Gargi B Dasgupta. 2018. Citicafe: An Interactive Interface for Citizen Engagement. In *23rd International Conference on Intelligent User Interfaces*. ACM, 617–628.

[5] Tseng-Hung Chen, Yuan-Hong Liao, Ching-Yao Chuang, Wan Ting Hsu, Jianlong Fu, and Min Sun. 2017. Show, Adapt and Tell: Adversarial Training of Cross-Domain Image Captioner.. In *ICCV*. 521–530.

[6] Tseng-Hung Chen, Yuan-Hong Liao, Ching-Yao Chuang, Wan Ting Hsu, Jianlong Fu, and Min Sun. 2017. Show, Adapt and Tell: Adversarial Training of Cross-Domain Image Captioner.. In *ICCV*. 521–530.

[7] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. 2018. Domain Adaptive Faster R-CNN for Object Detection in the Wild. In *Computer Vision and Pattern Recognition (CVPR)*.

[8] Annalisa Cocchia. 2014. Smart and digital city: A systematic literature review. In *Smart city*. Springer, 13–43.

[9] Peter Dahlgren. 2011. Parameters of online participation: Conceptualising civic contingencies. *Communication management quarterly* 21, 4 (2011), 87–110.

[10] Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised Domain Adaptation by Backpropagation. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15)*. JMLR.org, 1180–1189. http://dl.acm.org/citation.cfm?id=3045118.3045244

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2672–2680. http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[12] Lushan Han, Abhay L Kashyap, Tim Finin, James Mayfield, and Jonathan Weese. 2013. UMBC_EBIQUITY-CORE: Semantic textual similarity systems. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, Vol. 1. 44–52.

[13] IChangeMyCity. 2012. https://ichangemycity.com. (2012).

[14] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. 2015. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3668–3678.

[15] Maria Karakiza. 2015. The impact of social media in the public sector. *Procedia-Social and Behavioral Sciences* 175 (2015), 384–392.

[16] Matthew Klawonn and Eric Heim. 2018. Generating Triples with Adversarial Networks for Scene Graph Construction. *arXiv preprint arXiv:1802.02598* (2018).

[17] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, and others. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision* 123, 1 (2017), 32–73.

[18] Shanu Kumar, Shubham Atreja, Anjali Singh, and Mohit Jain. 2019. Adversarial Adaptation of Scene Graph Models for Understanding Civic Issues. *arXiv preprint arXiv:1901.10124* (2019).

[19] Kongming Liang, Yuhong Guo, Hong Chang, and Xilin Chen. 2018. Visual Relationship Detection with Deep Structural Ranking. (2018).

[20] Hiroya Maeda, Yoshihide Sekimoto, Toshikazu Seto, Takehiro Kashiyama, and Hiroshi Omata. 2018. Road Damage Detection Using Deep Neural Networks with Images Captured Through a Smartphone. *arXiv preprint arXiv:1801.09454* (2018).

[21] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*. 55–60.

[22] Graeme Mearns, Rebecca Simmonds, Ranald Richardson, Mark Turner, Paul Watson, and Paolo Missier. 2014. Tweet my street: a cross-disciplinary collaboration for the analysis of local twitter data. *Future Internet* 6, 2 (2014), 378–396.

[23] Ines Mergel. 2012. Distributed democracy: Seeclickfix. com for crowdsourced issue reporting. (2012).

[24] Gaurav Mittal, Kaushal B Yagnik, Mohit Garg, and Narayanan C Krishnan. 2016. Spotgarbage: smartphone app to detect garbage using deep learning. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 940–945.

[25] Nitish Mittal, Swati Agarwal, and Ashish Sureka. 2016. Got a Complaint?-Keep Calm and Tweet It!. In *International Conference on Advanced Data Mining and Applications*. Springer, 619–635.

[26] Taewoo Nam and Theresa A Pardo. 2011. Conceptualizing smart city with dimensions of technology, people, and institutions. In *Proceedings of the 12th annual international digital government research conference: digital government innovation in challenging times*. ACM, 282–291.

[27] Paolo Neirotti, Alberto De Marco, Anna Corinna Cagliano, Giulio Mangano, and Francesco Scorrano. 2014. Current trends in Smart City initiatives: Some stylised facts. *Cities* 38 (2014), 25–36.

[28] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. 2014. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1717–1724.

[29] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. 2018. Multi-Adversarial Domain Adaptation. In *AAAI*.

[30] Sebastian Schuster, Ranjay Krishna, Angel Chang, Li Fei-Fei, and Christopher D Manning. 2015. Generating semantically precise scene graphs from textual descriptions for improved image retrieval. In *Proceedings of the fourth workshop on vision and language*. 70–80.

[31] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, Vol. 1. 4.

[32] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. 2017. Scene graph generation by iterative message passing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2.

[33] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. 2018. Neural Motifs: Scene Graph Parsing with Global Context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5831–5840.

[34] Bohan Zhuang, Qi Wu, Chunhua Shen, Ian D Reid, and Anton van den Hengel. 2018. HCVRD: A Benchmark for Large-Scale Human-Centered Visual Relationship Detection.. In *AAAI*.

Redesigning Bitcoin's fee market

Ron Lavi
Technion
Israel
ronlavi@technion.ac.il

Or Sattath
Ben-Gurion University of the Negev
Israel
sattath@post.bgu.ac.il

Aviv Zohar
The Hebrew University of Jerusalem
Israel
avivz@cs.huji.ac.il

ABSTRACT

The Bitcoin payment system involves two agent types: Users that transact with the currency and pay fees and miners in charge of authorizing transactions and securing the system in return for these fees. Two of Bitcoin's challenges are (i) securing sufficient miner revenues as block rewards decrease, and (ii) alleviating the throughput limitation due to a small maximal block size cap. These issues are strongly related as increasing the maximal block size may decrease revenue due to Bitcoin's pay-your-bid approach. To decouple them, we analyze the "monopolistic auction" [8], showing: (i) its revenue does not decrease as the maximal block size increases, (ii) it is resilient to an untrusted auctioneer (the miner), and (iii) simplicity for transaction issuers (bidders), as the average gain from strategic bid shading (relative to bidding one's true maximal willingness to pay) diminishes as the number of bids increases.

CCS CONCEPTS

• **Theory of computation → Algorithmic mechanism design**; **Exact and approximate computation of equilibria**; **Computational pricing and auctions**; • **Computer systems organization → Peer-to-peer architectures**.

KEYWORDS

Bitcoin; Cryptocurrency; Blockchain; Auction-Theory; Fee-market

ACM Reference Format:
Ron Lavi, Or Sattath, and Aviv Zohar. 2019. Redesigning Bitcoin's fee market. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313454

1 INTRODUCTION

Bitcoin's security relies on its ability to attract honest miners and to incentivize them to invest large amounts of computational power [13]. This is done by rewarding miners for the creation of new blocks. The payment comprises a block reward that is a fixed amount per block, and a fee that each transaction pays for its inclusion in the block. Since the block reward is cut in half approximately every 4 years, transaction fees gradually become the miners' main incentive to participate. The decision on which transactions to include in the block is made using a "pay-your-bid" auction: each transaction submits its fee for its inclusion in the block. A miner, in turn, includes the highest bidding transactions that fit in the block. In this auction, larger block sizes imply smaller transaction fees, since all transactions will aim to pay the lowest possible fee for its inclusion in the block. The Bitcoin block size therefore significantly influences miners' revenue. Indeed, one of the main arguments against a block-size increase in Bitcoin has been that it may cause transaction fees to drop significantly. Thus, two of the Bitcoin system's main problems, i.e., obtaining sufficient revenue for the miners as the block reward gradually decreases and the throughput limitation as a result of the maximal block size, are in fact strongly related. This connection, however, is a major obstacle to their resolution. Our motivation in this paper is to decouple the issues of revenue and maximal block size (which should be determined according to other technical considerations, like block propagation times) and to provide scalability from the economic perspective of the fee market.

While most of auction theory assumes a trusted auctioneer that honestly follows the protocol, a miner (the auctioneer of the block) cannot be trusted. If adding fake bids can increase profit, a miner can add transactions by moving funds from one of its accounts to another. Similarly, a miner can ignore bids. An untrusted auctioneer can manipulate a 2^{nd} price auction, for example, by adding a fake bid that is very close to the highest bid. Bidders will realize that and shade down their bids. Though cryptography can, in general, help resolve some of these issues, as far as the authors are aware, cryptographic techniques are not applicable to Bitcoin: they assume a trusted setup, that the auctioneer is known in advance, or multiple communication rounds between all parties as in secure multi-party computation [1, 4, 12, 14].

We analyze here a potential solution to these issues, the *monopolistic auction* [8].[1] We observe that in the Bitcoin setting, the monopolistic auction is immune to untrusted auctioneers, on the one hand, and it decouples the revenue issue from the maximal block size issue, on the other. In this auction, given bids $b_1, ..., b_n$, the miner chooses which transactions to include in the block. All chosen transactions pay the same fee – the lowest bid in the block (as must be verified by the protocol). The maximal revenue resulting from this method cannot be increased by adding fake bids or by ignoring true bids. Other manipulations like side payments between miners and users are also not beneficial. In fact, we are not aware of any beneficial miner manipulation that can be exploited in the frame of this auction. Additionally and in contrast to a pay-your-bid auction, increasing the maximal block size does not decrease the revenue of the monopolistic auction. Choosing a threshold price is conceptually similar to dynamically choosing the block size – the miner chooses the size of the block it creates as a function of the

[1] Termed the "optimal single price omniscient auction" in [8], this auction was suggested as a benchmark, and as far as we know, its game-theoretic properties have not been analyzed in the Bitcoin context.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313454

bids received. In some bid instances, blocks will be small, while in others they will be large.

In a pay-your-bid auction, a user must strategically determine the lowest bid it can submit such that it will be included in the block ("bid shading"). Indeed, Bitcoin wallets use various fee estimation techniques. It is not clear how to do this optimally, and in any case, it requires some non-trivial computational effort (data gathering, statistical analysis, etc). Bid shading can also help in worst-case instances of the monopolistic auction, and as a result, [8] do not analyze its strategic properties but suggest it only as a benchmark for other truthful auctions they define. These other truthful auctions from [8] are less applicable vis-a-vis our needs (see Section 5). Although the approach is not truthful, in the monopolistic auction, a user, rather than paying her bid, pays some threshold value (a function of all bids) which w.h.p is only remotely related to one's bid. The main technical novelty of this paper is to show that the expected gain from optimal bid shading relative to truthfully bidding the maximal willingness to pay decreases to zero as the number of bids grows. On average, all users will benefit very little, if at all, from bid shading in the monopolistic auction. Since a near optimal strategy is to bid the maximal willingness to pay, bidding in the monopolistic auction is simpler. Section 2 defines the formal framework used and proves this statement, and Section 4 verifies it via simulations using synthetic and actual Bitcoin bid distributions. The greater simplicity of the monopolistic auction is another of its advantages over the pay-your-bid auction.

This analyzes a single block creation. "Patient users", who willing to wait for future blocks, might lower their bids if they anticipate less competition in the future. Two important issues that remain to be investigated in future research are (1) how patient users shade down their bids and (2) how non-myopic bidders who create persistent transactions affect the bid distribution. Note that these issues affect the revenue not only of the monopolistic auction, but also of the pay-your-bid auction. We do not know, however, which of the two auctions is more strongly affected. Our analytical results suggest that the monopolistic auction collects at least as much revenue from impatient users as Bitcoin's current mechanism.

Related Work. There are relatively few works devoted to analyses of the Bitcoin fee market and fewer still that suggest modifications to the basic mechanism. [10] provided an early analysis of the economics of Bitcoin mining, including a game theoretic analysis of the incentives to mine on the longest chain. [2] considered the incentives of miners to distribute transactions to each other and designed ways to share the fees in exchange for distribution. [6] explored the security of Bitcoin and the incentive to mine blocks properly when block rewards diminish and fees dominate the revenue of miners. They showed that variance in fees may undermine the security of the protocol and subject it to different forms of deviations and attacks. [5] explored related bribery attacks on the protocol that are paid for through promises of higher rewards for attackers that construct blocks off the longest chain. [15] considered the removal of the block limit altogether, arguing that delays in large block propagation times, which, in turn, imply a higher likelihood that the block is abandoned (often referred to as an orphaned block), will cause miners to restrict their own block sizes. The paper analyzed the resultant fee market that emerges. [9] modeled the transaction fee market and assumed that users benefit less if their transactions are delayed. They showed that such delays together with the congestion that may naturally occur in blocks due to queuing effects can lead to non-zero bids for transacting users even if blocks are not completely full. As far as we are aware, no previous work has explored a different mechanism for the fee market in crypto-currencies.

After the preprint version of this manuscript was published, Andrew Chi-Chih Yao [17] further analyzed the monopolistic price as a fee mechanism for Bitcoin, in the process resolving our main conjecture, Conjecture 2.6 – see the discussion there.

Basu, Easley, O'Hara and Sirer [3] used a modified monopolistic price auction to determine inclusion in the block. They penalize miners if the blocks are not full, and additionally, they average the reward from blocks between different miners. Their goal is to maximize social welfare rather than the miners' revenue. The main analytical difference between their work and ours is that theirs is based mostly on simulations while ours is proved for the most part formally. In terms of abstract conclusions, our results agree with theirs: they also establish that the incentive of miners and users to manipulate diminishes as n grows.

A full version with greater detail and the full proofs is in Ref. [11].

2 THE MONOPOLISTIC AUCTION

There are n transactions, each of which has a privately known maximal willingness to pay v_i for its inclusion in the block, e.g., an alternative transaction cost via a bank, credit card, etc. Transactions submit bids, and based on these, the miner decides which transactions to include in the block. The bid vector is sorted so that $\boldsymbol{b} = (b_1, \ldots, b_n)$ satisfies $b_1 \geq \ldots \geq b_n$. Let $k^*(\boldsymbol{b})$ be the index k that maximizes $k \cdot b_k$. In case of ties, k^* is chosen to be the maximal such index. Define:

$$R(\boldsymbol{b}) = k^*(\boldsymbol{b}) \cdot b_{k^*(\boldsymbol{b})} \ , \ p^M(\boldsymbol{b}) = b_{k^*(\boldsymbol{b})}, \tag{1}$$

where $R(\boldsymbol{b})$ is termed the "monopolistic revenue" and $p^M(\boldsymbol{b})$ is termed the monopolistic price. In the monopolistic auction, the miner includes in the block the $k^*(\boldsymbol{b})$ transactions who submitted the highest bids and charges a payment of $p^M(\boldsymbol{b}) = b_{k^*(\boldsymbol{b})}$ from each of these transactions. The miner's revenue from this block is therefore $R(\boldsymbol{b})$.

An untrusted auctioneer that can add fake bids does not have any incentive to do so in this auction. Specifically, the Bitcoin protocol can verify that all transactions in the same block pay the same price. However, it cannot prevent miners from adding fake bids by exploiting transactions that move funds between two accounts of the same miner. These fake bids can affect the payments of the "real" bids and increase the revenue of the auctioneer, as is the case in a second-price auction. In the monopolistic auction,in contrast, the addition of fake bids cannot increase the revenue: given any vector of real bids $\boldsymbol{b}$, since all transactions in the block pay the same price, the maximal revenue is $R(\boldsymbol{b})$ if the miner does not create another block in the near future. However, a bidder might benefit from submitting $b_i < v_i$, as demonstrated by the following examples:

Example 2.1. Suppose n bidders that have the same maximum fee $v_i = 1$. If all bids are $b_i = v_i$, then $R(\boldsymbol{b}) = n, k^*(\boldsymbol{b}) = n$, and

$p^M(\boldsymbol{b}) = 1$, i.e., all bids get accepted to the block and they all pay 1. A strategic bidder, say 1, can reduce her payment by reducing her bid to $b_1 = \frac{n-1}{n}$. With this bid, the monopolistic price decreases to $\frac{n-1}{n}$: if all bids of value 1 are accepted, the revenue is $n-1$, and if all bids of value at least $b_1 = \frac{n-1}{n}$ are accepted, the revenue is still $n-1$.

The gain from bid shading in this example vanishes as the number of users (and the block size) increases. In other cases, gains do not vanish even when the number of bidders grows:

Example 2.2. $v_1 = \ldots = v_{\frac{n}{2}+1} = 2, v_{\frac{n}{2}+2} = \ldots = v_n = 1$. Bidder 1 can reduce her price by bidding $\frac{n-1}{n}$ instead of 2.

However, if the values for example 2.2 are taken from realistic distributions, then the probability that it will be realized will usually be very small.

Therefore, rather than allege that there are no profitable deviations from truth-telling, we will only claim that the expected gain from any non-truthful bid diminishes as n grows. For example, if the distribution of values has a finite support, bidding the next lower value in the support *weakly dominates* truth-telling. Though more pronounced bid shading may be beneficial, it will not be a dominant strategy. We will show, however, that the expected gain from such a deviation or from any other deviation becomes extremely small as n grows. Therefore, we believe that it makes sense to conclude that only small deviations will likely happen.

Example 2.3. Consider the simplest equal-revenue distribution F with a finite support: $\Pr[v = 1] = 0.5, \Pr[v = 2] = 0.5$. For simplicity, assume that bidders can only submit bids of 1 or 2. Every bidder with $v = 2$ has a dominant strategy to bid 1, since she derives 0 utility when the monopoly price is set to be 2, and by bidding 1 she increases the chance that the monopoly price will be 1. The probability that a bidder with $v = 2$ can improve the outcome (i.e., change the empirical monopoly price from 2 to 1) by bidding b=1 in a 2n bidder auction is $\frac{\binom{2n}{n+1}}{2^{2n}} = \Theta\left(\frac{1}{\sqrt{n}}\right)$.

We validate this intuition more generally via both theoretical and empirical analyses. First we define some terms. Fix a user i and a vector of bids $\boldsymbol{b}_{-i} \equiv (b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_n)$ of the other users.

$$p^S(\boldsymbol{b}_{-i}) \equiv \min\{b_i \in \mathbb{R} \mid p^M(b_i, \boldsymbol{b}_{-i}) \leq b_i\}.$$

In words, $p^S(\boldsymbol{b}_{-i})$ is the lowest possible bid for i to be included in the block. Figure 1 gives an example. This definition does not capture a case where a bidder can provide false-name bids (i.e., can split her transaction into several transactions), which is discussed in Section 3.

Figure 2 presents the strategic prices in an example with $n = 256$ users. In this example, p^S is monotonically decreasing among winning bidders (higher winning bids imply lower strategic prices) but not among all bidders. This property is formally proved in the full version [11]. Additionally, here and in most other simulations, almost all winning bids have the same strategic price, the exception being the very few lowest winning bids (i.e., those with prices slightly higher than the monopolistic price). The strategic price of these bids is higher, i.e., they gain less from being strategic.

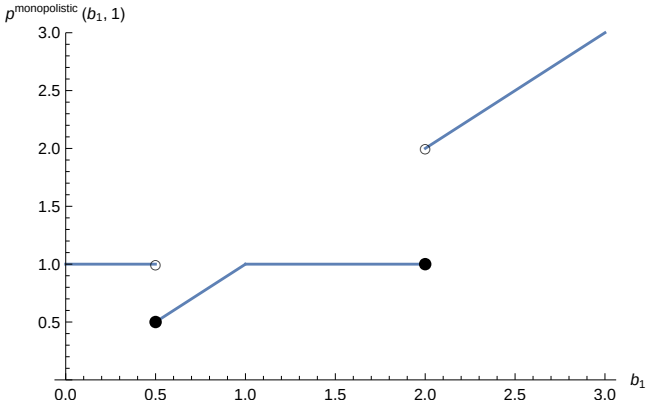


Figure 1: The function $p^M(b_1, 1)$ as a function of b_1, demonstrating that p^M is neither monotone nor continuous. As can be seen, the strategic price of the first bidder is 0.5.

While $p^S(\boldsymbol{b}_{-i})$ is the minimal price that bidder i must pay to be included in the block, she will pay $p^M(v_i, \boldsymbol{b}_{-i})$ if she bids her true maximal willingness to pay. This possible gain from bid shading is captured by the notion of the "discount ratio":

$$\delta_i(v_i, \boldsymbol{b}_{-i}) = \begin{cases} 1 - \frac{p^S(\boldsymbol{b}_{-i})}{p^M(v_i, \boldsymbol{b}_{-i})} & \text{if } v_i \geq p^S(\boldsymbol{b}_{-i}) \\ 0 & \text{otherwise.} \end{cases}$$

Clearly, $0 \leq \delta_i(v_i, \boldsymbol{b}_{-i}) \leq 1$. If $v_i < p^S(\boldsymbol{b}_{-i})$, every bid of at most v_i loses. Hence, the gain from bid shading is 0. If $v_i \geq p^S(\boldsymbol{b}_{-i})$, bidder i can win and pay $p^S(\boldsymbol{b}_{-i})$. With an honest bid, she will pay $p^M(v_i, \boldsymbol{b}_{-i})$. She can thus save a percentage of $1 - \frac{p^S(\boldsymbol{b}_{-i})}{p^M(v_i, \boldsymbol{b}_{-i})}$ from her price by strategic bid shading.

As we showed above that the discount factor can be large in the worst case, we therefore conduct an average-case analysis. Assume that all true values are drawn i.i.d. from some distribution F on $\mathbb{R}_{>0}$. The average discount ratio is then $\Delta_n^{average} = \mathbb{E}_{(v_1,\ldots,v_n)\sim F}[\delta_1(v_1, \boldsymbol{v}_{-1})]$ (the choice of bidder 1 is arbitrary since all are symmetric a-priori).

We also consider two stronger notions. In the first, for each realization $v_1, \ldots, v_n$, we consider the maximal discount ratio among all bidders:

$$\delta_{max}(\boldsymbol{v}) = \max_i \delta_i(v_i, \boldsymbol{v}_{-i}) \ , \ \Delta_n^{max} = \mathbb{E}_{(v_1,\ldots,v_n)\sim F}[\delta_{max}(\boldsymbol{v})]$$

Clearly, for all n, $\Delta_n^{max} \geq \Delta_n^{average}$ since for every $\boldsymbol{v}$ and every i, $\delta_{max}(\boldsymbol{v}) \geq \delta_i(v_i, \boldsymbol{v}_{-i})$.

Theorem 2.4. *For any distribution F with a finite support size,* $\lim_{n\to\infty} \Delta_n^{max} = 0$.

Section 2.1 proves this theorem.

In the second extension, defined only for distributions with a finite bounded support, we fix player 1 and assume that she deterministically has a value that maximizes her discount ratio (the worst possible value for her, from our perspective) while all other values are probabilistic. This depicts a situation commonly assumed in game theory wherein a user knows her own value and (only) the

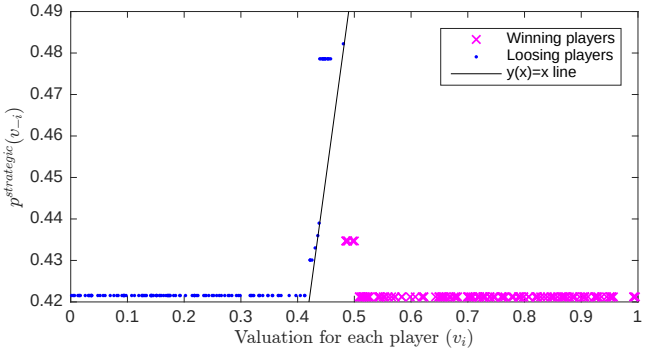


Figure 2: An example with $n = 256$ bids sampled i.i.d. from the uniform distribution on $[0, 1]$. For each user, the x-axis shows her values and the y-axis shows her strategic price. Winning bids, which are marked with 'x', must be at least as high as their respective strategic price, i.e., to the right of the black line.

distribution of the other values.

$$\delta_n^{GT}(\boldsymbol{b}_{-1}) = \max_{v_1 \in \text{Support}(F)} \delta_1(v_1, \boldsymbol{b}_{-1})$$

$$\Delta_n^{GT} = \mathbb{E}_{(v_2,\ldots,v_n)\sim F}[\delta_n^{GT}(\boldsymbol{v}_{-1})]$$

Theorem 2.5. *For any distribution F with a finite support size, $\lim_{n\to\infty} \Delta_n^{GT} = 0$.*

The proof of this theorem is very similar to that of Theorem 2.4.

Our empirical analysis focuses on Δ_n^{max}, since it is well-defined even for distributions with unbounded support.

While $\lim_{n\to\infty} |\Delta_n^{GT} - \Delta_n^{max}| = 0$, the relation between Δ_n^{GT} and Δ_n^{max} depends on the distribution, i.e., one term is not always larger than the other – see the full version for concrete examples. The e-print version contained the following conjectures:

Conjecture 2.6 (Nearly Bayesian-Nash Incentive-Compatibility).

1. *For any distribution F, $\lim_{n\to\infty} \Delta_n^{average} = 0$. In particular, $\Delta_n^{average} = O\left(\frac{1}{\sqrt{n}}\right)$, where the constant in the $O(\cdot)$ notation may depend on F.*
2. *If F has a bounded support (still possibly with an infinite cardinality) $\lim_{n\to\infty} \Delta_n^{\max} = 0$. In particular, $\Delta_n^{\max} = O\left(\frac{1}{\sqrt{n}}\right)$. The same holds for Δ_n^{GT}.*
3. *There exists a distribution F with unbounded support such that $\lim_{n\to\infty} \Delta_n^{\max} > 0$. The same holds for Δ_n^{GT}.*

The $O\left(\frac{1}{\sqrt{n}}\right)$ in the above conjecture can be achieved by Example 2.3.

Recently, Andrew Chi-Chih Yao resolved many issues that were left open in this work. Indeed, "The main purpose [of Yao's paper] is to settle the Nearly IC Conjecture for the monopolistic price in the positive" [17]. Specifically, the first sentences in items 1 and 2 were proved, and a slightly weaker claim was shown than the "in particular" claims in these items. This was proved also when accounting for multiple strategic bids – see Section 3. Item 3 was proved for the inverse distribution (see Eq. (3)), which was our primary candidate for this conjecture based on numerical data – see Section 4.

2.1 Proof of Theorem 2.4

Given $\boldsymbol{v}$, define: $i^* = \text{argmax}_{i=1,\ldots,n} v_i$, $k^* = k^*(\boldsymbol{v}_{-i^*})$, $p^H = p^M(\boldsymbol{v})$, and $p^S = p^S(\boldsymbol{v}_{-i^*})$.

Lemma 2.7. *For any F with finite support size, there exists a constant $c > 0$ (which may depend on F but not on n) s.t. $\lim_{n\to\infty} \Pr[k^* < c \cdot n] = 0$.*

Proof. Let $v_{max} = \max \text{Support}(F)$, k_{max} be the number of bidders in $\boldsymbol{v}$ who bid v_{max}, and $p_{max} = \Pr_{v_i\sim F}[v_i = v_{max}]$. By linearity of expectation, $\mathbb{E}[k_{max}] = p_{max} \cdot n$. A Chernoff bound implies that $\lim_{n\to\infty} \Pr\left(k_{max} < \frac{9np_{max}}{10}\right) = 0$. Bidders who bid v_{max} win; therefore, $k^* + 1 \geq k_{max}$. Thus, $\lim_{n\to\infty} \Pr\left(k^* + 1 < \frac{9np_{max}}{10}\right) = 0$. Choosing $c = \frac{8p_{max}}{10}$ completes the proof of the lemma. □

The following three claims are proved in the full version [11]:

Claim 1. $\forall \boldsymbol{v}, i,\ p^M(p^S(\boldsymbol{v}_{-i}), \boldsymbol{v}_{-i}) = p^S(\boldsymbol{v}_{-i})$.

Claim 2. $\forall \boldsymbol{v}, i, j$: *$v_i \geq v_j$ implies $\delta_i(v_i, \boldsymbol{v}_{-i}) \geq \delta_j(v_j, \boldsymbol{v}_{-j})$.*

Claim 3. *Let y be the smallest element in the support of F that is at least p^S. Then, $y = p^H$ implies $p^S \geq \frac{k^*}{k^*+1} \cdot p^H$.*

With these claims, we prove a second lemma:

Lemma 2.8. $\lim_{n\to\infty} \Pr(p^S < \frac{k^*}{k^*+1} p^H) = 0$.

Proof. Define A as the event where $p^S < \frac{k^*}{k^*+1} p^H$. The proof defines an event B s.t. (i) $\lim_{n\to\infty} \Pr(B) = 0$, and (ii) $A \subseteq B$. This implies $\lim_{n\to\infty} \Pr(A) = 0$ as claimed.

To define event B, fix any arbitrary two $x, y \in \text{Support}(F)$ such that $x > y$. Define $\text{num}(\boldsymbol{v}, z) \equiv |\{v_i | v_i \geq z\}|$. Define the random variables $n_z = \text{num}(\boldsymbol{v}, z)$ for any real number z, $h(x) = n_x \cdot x$, and $g(x) = (n_x - 1) \cdot x$. We first aim to bound the probability that $h(x) > h(y) \geq g(x)$. This is the same as the probability that $n_x > \frac{y}{x} n_y \geq n_x - 1$, which is true only if $n_x = \lfloor \frac{y}{x} n_y + 1 \rfloor$. The triple $(n_x, n_y - n_x, n - n_y)$ is a trinomial distribution with probabilities $p_1 = \Pr_{v_i\sim F}(v_i \geq x), p_2 = \Pr_{v_i\sim F}(x > v_i \geq y), p_3 = \Pr_{v_i\sim F}(v_i < y)$ (p_i depends on F but not on n). By simple analysis of the trinomial distribution, we show in the full version that

$$\lim_{n\to\infty} \Pr(h(x) > h(y) \geq g(x)) = 0. \tag{2}$$

Let B be the event in which $\exists x, y \in \text{Support}(F)$ s.t. $x > y$ and $h(x) > h(y) \geq g(x)$. Since the support is of finite size, there is a fixed number of such pairs $x > y \in \text{Support}(F)$. By Eq. (2) and the union bound, we conclude that $\lim_{n\to\infty} \Pr(B) = 0$. We show that $A \subseteq B$, i.e., $p^S < \frac{k^*}{k^*+1} p^H$ implies $\exists x > y \in \text{Support}(F)$ s.t. $h(x) > h(y) \geq g(x)$. Let $x = p^H$ and let y be the smallest element in the support of F, which is at least p^S. By Claim 3, the event $p^S < \frac{k^*}{k^*+1} p^H$ implies $x > y$. Since $p^M(v_{i^*}, \boldsymbol{v}_{-i^*}) = x$, it follows that $h(x) = n_x \cdot x > n_y \cdot y = h(y)$. Furthermore,

$$h(y) \geq h(p^S) = n_{p^S} \cdot p^S = \text{num}(\boldsymbol{v}, p^S) \cdot p^S = \text{num}((p^S, \boldsymbol{v}_{-i^*}), p^S) \cdot p^S$$
$$\geq \text{num}((p^S, \boldsymbol{v}_{-i^*}), x) \cdot x = (n_x - 1) \cdot x = g(x)$$

where the first step follows since $n_y = n_{p^S}$ and $p^S \le y$, the fourth step follows since $v_{i^*} \ge x > y \ge p^S$, the fifth step follows from Claim 1 that shows that $p^M(p^S, \boldsymbol{v}_{-i^*}) = p^S$, and the sixth step follows since $v_{i^*} \ge x > y \ge p^S$. □

Let the "bad" event E_1 be the case where $k^* < c \cdot n$ (c is taken from Lemma 2.7) and the "bad" event E_2 be the case where $p^S < \frac{k^*}{k^*+1} p^H$. By Claim 2, $\delta_{max}(\boldsymbol{v}) = \delta_{i^*}(\boldsymbol{v})$. Therefore, $\Delta_n^{max} = \mathbb{E}_{\boldsymbol{v}}[\delta_{i^*}(\boldsymbol{v})]$. If E_2 does not hold, then $p^S \ge \frac{k^*}{k^*+1} p^H$, and therefore, $\delta_{i^*}(\boldsymbol{v}) \le \frac{1}{k^*+1}$. To conclude:

$$\begin{aligned}
\lim_{n\to\infty} \Delta_n^{max} &= \lim_{n\to\infty} \Big[\Pr[E_1 \cup E_2]\, \mathbb{E}_v[\delta_{i^*}(v_{i^*}, v_{-i^*}) | E_1 \cup E_2] \\
&\quad + \Pr[E_1^c \cap E_2^c]\, \mathbb{E}_v \left[\delta_{i^*}(v_{i^*}, v_{-1}) | E_1^c \cap E_2^c \right] \Big] \\
&\le \lim_{n\to\infty} \left[\Pr[E_1] + \Pr[E_2] + \Pr[E_1^c \cap E_2^c] \cdot \mathbb{E}_v[\frac{1}{k^*+1} | E_1^c \cap E_2^c] \right] \\
&\le \lim_{n\to\infty} \left[\Pr[E_1] + \Pr[E_2] + \Pr[E_1^c \cap E_2^c] \cdot \frac{1}{c \cdot n} \right] \\
&\le \lim_{n\to\infty} \left[\Pr[E_1] + \Pr[E_2] + \frac{1}{c \cdot n} \right] = 0,
\end{aligned}$$

implying Theorem 2.4.

3 MULTIPLE STRATEGIC BIDS

In some cases, it is beneficial to split one's bid into separate transactions with several separate bids. In fact, such a strategy sometimes enables a losing transaction to be included in the block:

Example 3.1. Let $\boldsymbol{v} = (5, 2, 1, 1)$. With bids $\boldsymbol{b} = \boldsymbol{v}$, the monopolistic price is 5 and the second bidder loses. However, if she submits two separate bids of 1 each, the monopolistic price will be 1 and all bids will be included.

Our empirical evaluation below accounts for such situations, showing that the benefit from using multiple bids also goes to zero as n increases. In all the distributions we examined, except the discrete distribution, we *never* encountered a case in which the strategic player has an advantage placing multiple bids. Even in the discrete case, the effect of such multiple bids was negligible. See the full version for more detail.

4 EMPIRICAL EVALUATION

We provide empirical evidence to support the above conjectures and to supply greater detail on the rate at which the discount ratio converges to zero. We use both synthetic data as well as transaction data taken from the Bitcoin blockchain. The synthetic data are generated using four distributions:

(1) A uniform discrete distribution over the integers $1, \ldots, 100$. This distribution has a finite support size and therefore satisfies the requirements of Theorem 2.4.
(2) The uniform distribution over $[0, 1]$. Notice that here the support size is infinite.
(3) The half normal distribution. Here, the probability for maximum fee x decreases exponentially with x, hence, even though arbitrarily high transaction fees will be seen, they are highly unlikely.

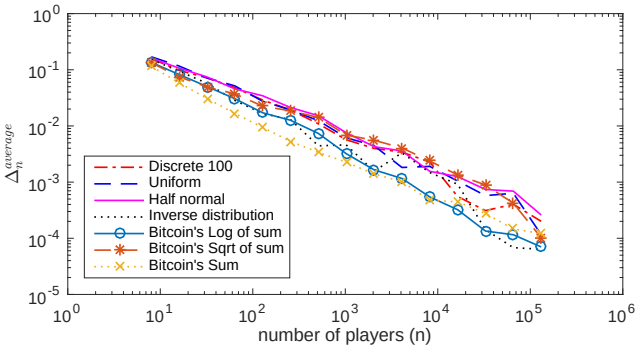


Figure 3: The average discount ratio of a player from selfish bidding as a function of the number of players that participate.

(4) The inverse distribution

$$F(x) = 1 - \frac{1}{x}, \quad x \in [1, \infty). \tag{3}$$

Here, the probability for a maximum fee x decreases polynomially with x.

The uniform distribution has no tail, the half normal distribution has a light tail (it decreases exponentially fast), and the inverse distribution has a heavy tail.

Bitcoin blockchain data was collected from 1000 consecutive blocks, which constitute roughly one week of activity ending on October 28th, 2016. The data obviously do not contain the maximum willingness to pay, since this is not how Bitcoin operates. The simulations estimate the maximum willingness to pay, v, as a function $v = v(x)$ of the transaction size x (x is the sum of all outputs of the Bitcoin transaction). We use three alternative functions: $v(x) = \log x, v(x) = \sqrt{x}, v(x) = x$. Note that multiplying all bids by a scalar does not change the discount ratio, which is the case when, e.g., the maximum willingness to pay is some percentage of $v(x)$.

We empirically evaluated $\Delta_n^{average}$ and Δ_n^{max} as a function of the number of bids, n. For each $n \in \{2^3, 2^4, \ldots, 2^{17}\}$, we conducted 100 simulation runs. Each run samples n bids and calculates the n empirical discount ratios. $\Delta_n^{average}$ is the average of the n individual discount ratios and Δ_n^{max} is the maximum over the n individual discount ratios. Each point in the graph is the average of 100 runs.

Figure 3 shows that $\Delta_n^{average}$ behaves almost identically for all tested distributions. In particular, as stated in the first part of Conjecture 2.6, $\Delta_n^{average}$ decreases linearly with the number of bids n and for all distributions, even those with an infinite and unbounded support size.

Figure 4 shows that Δ_n^{max} behaves differently for some of the tested distributions. For the uniform distribution, Δ_n^{max} decreases linearly with the number of bids, supporting the second part of Conjecture 2.6. The half normal distribution behaves similarly (even though it is not bounded). For the inverse distribution, Δ_n^{max} does not seem to decrease with the number of users, and we believe that this is in fact an example that supports the third part of Conjecture 2.6. The three Bitcoin distributions behave as follows: the log of the sum decreases the fastest, the square root of the sum decreases

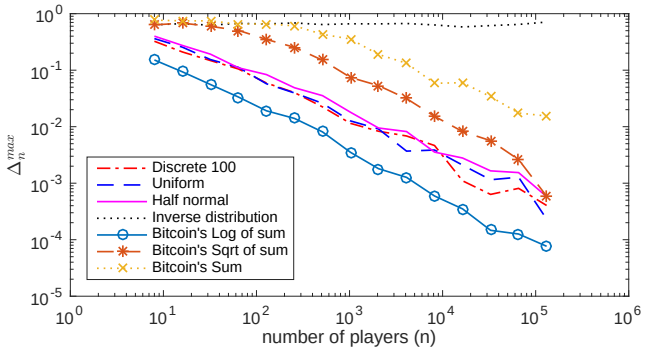


Figure 4: The maximal discount ratio of a player from selfish bidding as a function of the number of players.

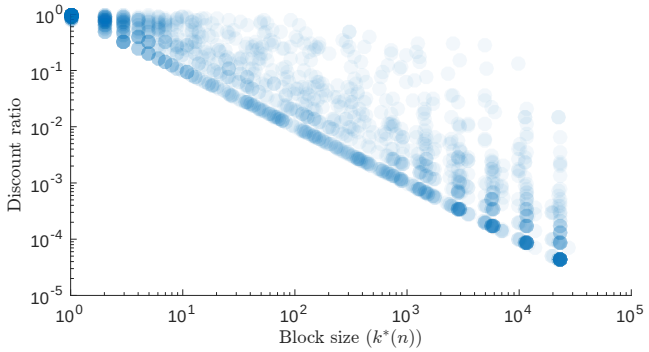


Figure 5: A scatter plot of all simulation points for the Bitcoin distribution with $v(x) = \log x$.

more slowly, and the sum decreases the slowest. For example, for $n = 2048$ (roughly the current Bitcoin block-size), Δ_n^{max} is about 0.2%, 5%, and 19% for the three distributions, respectively. When $n = 2^{17} \approx 130,000$, Δ_n^{max} is about 0.0007%, 0.05%, and 1%.

These findings show that there is a qualitative difference between $\Delta_n^{average}$ and Δ_n^{max}, mainly for the inverse distribution and for the Bitcoin distribution with $v(x) = x$. In these distributions, although the average transaction will benefit very little from bid shading, some transactions can nonetheless obtain significant benefit. This outcome can, in principle, result from two different reasons. The first is that $\Delta_n^{average}$ also accounts for the losing bidders, whose discount ratio is zero. The second is that Δ_n^{max} only accounts for the single winning bidder with the maximal discount ratio. However, typical bidder behavior (as demonstrated in Figure 2) shows that almost all winning bids have the lowest strategic price and, therefore, the highest discount ratio. Thus, the explanation for the difference between the two discount ratios seems to be the first.

Figure 5 shows all simulation points for the Bitcoin log of of sum distribubtion. The x-axis is the resulting *block size* of the simulation run (in the previous figures it was n) and the y-axis is the maximal discount ratio of the simulation run. Three main conclusions can be drawn from Figure 5. First, block sizes (number of winners) range from 1 to about 25,000 when $v(x) = \log x$ (recall that the total number of bids ranges from 2^3 to 2^{17}).[2] Second, this range in block sizes is the main reason why Δ_n^{max} is smaller in the Bitcoin log of the distribution of sums, as larger block sizes imply smaller discount factors. Third, the distribution of the simulation points, which include outliers, is not normal. We do not have an explanation for this finding. The last two remarks apply to all tested distributions.

5 SUMMARY AND CONCLUDING REMARKS

The simple pay-your-bid fee mechanism used in Bitcoin has several disadvantages: (i) A fixed (hard-coded) maximal block size implies non-optimal revenue extraction when the pre-fixed block-size is too small or too large for the current bid instance, (ii) Bitcoin wallets must invest computational effort in bid shading, and (iii) deciding on the maximal block size involves economic rather than purely technological considerations (e.g., block propagation time [7, 16]). For single block creation, the proposed monopolistic auction effectively solves all these issues while maintaining miner non-manipulability.

The monopolistic auction is ϵ-truthful (ϵ goes to zero as the number of bidders grows). Alternatively, we can take a truthful auction where a bidder cannot influence its winning price, e.g., the RSOP auction is truthful and its revenue $RSOP(\boldsymbol{v})$ satisfies $\lim_{n\to\infty} \max_{\boldsymbol{v}} \frac{R(\boldsymbol{v})}{RSOP(\boldsymbol{v})} = 1$ [8]. The full version describes an implementation in Bitcoin (e.g., the required randomness needs to be verified by other miners) and discusses major drawbacks of it. In light of these shortcomings, we believe that the monopolistic auction is better suited to implement improvements in Bitcoin's fee market.

Future Work. One important issue we did not cover is that of the temporal implications of patient users who are willing to wait for transactions to be included in future blocks, as in the current analysis we only address impatient users. The monopolistic auction can also be explored further via a careful equilibrium analysis. The astute reader may notice that the current model allows for an equilibrium in which all bidders bid 0, an unrealistic setting since in practice there exists a maximum limit on the block size, and not all transactions can be included in the block. We believe that in such a setting, no other meaningful equilibria exist other than truthfulness, which is an ϵ-BNE. To reach a definitive conclusion, however, a careful analysis is required.

6 ACKNOWLEDGMENTS

We thank Alaa Mozalbat, who designed and implemented the simulation code, and Reshef Meir for valuable discussions. R.L. was partly supported by a Marie-Curie fellowship "Advance-AGT", by an ARCHES award from the MINERVA foundation, and by the ISF-NSFC joint research program (grant No. 2560/17). O.S. was supported by ERC Grant 280157, by the Israel Science Foundation (grant 682/18), and by the Cyber Security Research Center at Ben-Gurion University. A.Z. was supported by the Israel Science Foundation (grant 616/13) and by a grant from the HUJI Cyber Security Research Center in conjunction with the Israel National Cyber Bureau (grant 039-9230).

[2]The throughput when $v(x) = \log x$ will therefore be about $0.1n$ (n is the number of all transactions). For $v(x) = x$, block sizes range from 1 to about 1,500, implying a throughput of about $0.01n$.

REFERENCES

[1] Abe, M., Suzuki, K.: M+1-st price auction using homomorphic encryption. In: International Workshop on Public Key Cryptography. pp. 115–124. Springer (2002)

[2] Babaioff, M., Dobzinski, S., Oren, S., Zohar, A.: On Bitcoin and red balloons. In: ACM Conference on Electronic Commerce, EC '12, Valencia, Spain, June 4-8, 2012. pp. 56–73 (2012). https://doi.org/10.1145/2229012.2229022, http://doi.acm.org/10.1145/2229012.2229022

[3] Basu, S., Easley, D., O'Hara, M., Sirer, E.G.: Towards a functional fee market for cryptocurrencies (2019), http://arxiv.org/abs/1901.06830

[4] Bogetoft, P., Damgård, I., Jakobsen, T., Nielsen, K., Pagter, J., Toft, T.: A practical implementation of secure auctions based on multiparty integer computation. In: International Conference on Financial Cryptography and Data Security. pp. 142–147. Springer (2006)

[5] Bonneau, J.: Why buy when you can rent? - bribery attacks on bitcoin-style consensus. In: Financial Cryptography and Data Security - FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers. pp. 19–26 (2016). https://doi.org/10.1007/978-3-662-53357-4_2, https://doi.org/10.1007/978-3-662-53357-4_2

[6] Carlsten, M., Kalodner, H.A., Weinberg, S.M., Narayanan, A.: On the instability of bitcoin without the block reward. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016. pp. 154–167 (2016). https://doi.org/10.1145/2976749.2978408, http://doi.acm.org/10.1145/2976749.2978408

[7] Decker, C., Wattenhofer, R.: Information propagation in the bitcoin network. In: 13th IEEE International Conference on Peer-to-Peer Computing, IEEE P2P 2013, Trento, Italy, September 9-11, 2013, Proceedings. pp. 1–10 (2013). https://doi.org/10.1109/P2P.2013.6688704, https://doi.org/10.1109/P2P.2013.6688704

[8] Goldberg, A.V., Hartline, J.D., Karlin, A.R., Saks, M.E., Wright, A.: Competitive auctions. Games and Economic Behavior **55**(2), 242–269 (2006). https://doi.org/10.1016/j.geb.2006.02.003, http://dx.doi.org/10.1016/j.geb.2006.02.003

[9] Huberman, G., Leshno, J.D., Moallemi, C.C.: Monopoly without a monopolist: An economic analysis of the bitcoin payment system. https://ssrn.com/abstract=3025604 (2017)

[10] Kroll, J.A., Davey, I.C., Felten, E.W.: The economics of bitcoin mining, or bitcoin in the presence of adversaries. In: Proceedings of WEIS. vol. 2013 (2013)

[11] Lavi, R., Sattath, O., Zohar, A.: Redesigning bitcoin's fee market (2017), http://arxiv.org/abs/1709.08881

[12] Lipmaa, H., Asokan, N., Niemi, V.: Secure vickrey auctions without threshold trust. In: International Conference on Financial Cryptography. pp. 87–101. Springer (2002)

[13] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008), http://www.bitcoin.org/bitcoin.pdf

[14] Naor, M., Pinkas, B., Sumner, R.: Privacy preserving auctions and mechanism design. In: Proceedings of the 1st ACM conference on Electronic commerce. pp. 129–139 (1999)

[15] Rizun, P.R.: A transaction fee market exists without a block size limit. Block Size Limit Debate Working Paper (2015), https://www.bitcoinunlimited.info/resources/feemarket.pdf

[16] Sompolinsky, Y., Zohar, A.: Secure high-rate transaction processing in bitcoin. In: Financial Cryptography and Data Security - 19th International Conference, FC 2015, San Juan, Puerto Rico, January 26-30, 2015, Revised Selected Papers. pp. 507–527 (2015). https://doi.org/10.1007/978-3-662-47854-7_32, https://doi.org/10.1007/978-3-662-47854-7_32

[17] Yao, A.C.: An incentive analysis of some bitcoin fee designs (2018), http://arxiv.org/abs/1811.02351

Measuring Political Personalization of Google News Search

Huyen Le
Department of Computer Science
University of Iowa

Raven Maragh
Department of Communication Studies
Gonzaga University

Brian Ekdale
School of Journalism and Mass Communication
University of Iowa

Andrew High
Department of Communication Arts and Sciences
Pennsylvania State University

Timothy Havens
Department of Computer Science
University of Iowa

Zubair Shafiq
Department of Computer Science
University of Iowa

ABSTRACT

There is a growing concern about the extent to which algorithmic personalization limits people's exposure to diverse viewpoints, thereby creating "filter bubbles" or "echo chambers." Prior research on web search personalization has mainly reported location-based personalization of search results. In this paper, we investigate whether web search results are personalized based on a user's browsing history, which can be inferred by search engines via third-party tracking. Specifically, we develop a "sock puppet" auditing system in which a pair of fresh browser profiles, first, visits web pages that reflect divergent political discourses and, second, executes identical politically oriented Google News searches. Comparing the search results returned by Google News for distinctly trained browser profiles, we observe statistically significant personalization that tends to reinforce the presumed partisanship.

CCS CONCEPTS

• **General and reference** → **Measurement**; • **Information systems** → **Personalization**.

KEYWORDS

Measurement; Personalization; Google; News; Search

ACM Reference Format:
Huyen Le, Raven Maragh, Brian Ekdale, Andrew High, Timothy Havens, and Zubair Shafiq. 2019. Measuring Political Personalization of Google News Search. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3312504

1 INTRODUCTION

When a person looks up information on a search engine, the search does not occur in a vacuum. Before any particular search is made, the person has already lived an active digital life—reading news stories, liking pages on online social media, and posting comments on blogs—activities that are tracked across the Web to train personalization algorithms. In the current divisive political climate [9], people might be more likely to access content online that conforms to their political ideologies, but little is known about whether accessing certain political content impacts personalization of other elements of a person's online life. In this paper, we ask: *If a person's digital life demonstrates a political bias, does that person receive personalized search results and do those search results reflect that person's bias?*

Researchers have long been concerned about the effect of selective exposure—seeking information that reinforces preexisting beliefs while avoiding other information—on democratic societies [38, 43]. Through selective exposure, people with biases seek out content that conforms with their preexisting beliefs. For example, selective exposure to conservative news is associated with support for strict immigration policies [44], and people who score high on a scale of modern racism are more likely to view non-traditional Internet sources as credible sources of news [26]. Researchers have recently started to question whether algorithms create distinct personalized experiences for users, leading to so-called "filter bubbles" or "echo chambers" [7, 19, 31]. Algorithms reflect the societies in which they are produced, so it is unsurprising that they are encoded with biases [13]. For example, personalization algorithms have been found to discriminate against women [11] and people of color [29].

While selective exposure requires deliberate acts of media choice, algorithmic personalization interprets past behavior as precedent for future preference. In other words, algorithmic personalization can intensify selective exposure beyond a person's choice, resulting in a vicious cycle that can contribute to an increasingly polarized society. According to Pew Research, Americans are more ideologically polarized now than at any point in the last two decades [9]. This increase in polarization coincides with the rise of search engines and social media sites as primary sources of information. Thus, it is important to study the role of personalization algorithms employed by search engines in reinforcing pre-existing biases.

Prior research has reported significant personalization in online advertising based on a user's browsing history. Personalization in online advertising [8, 11, 37, 41] is not surprising because behavioral targeting is its key design feature. To target ads, advertisers employ sophisticated online tracking techniques for profiling users' browsing habits across the Web [12]. Naturally, we would expect search engines to leverage browsing history information obtained by online tracking for search personalization as well. However, prior studies [16, 20, 42] have reported significant search personalization mainly based on location, not browsing history.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3312504

In this paper, we set out to study search personalization in Google News using a "sock puppet" auditing system [36] in which we use automated programs to impersonate users with different political ideologies. While controlling for other factors, we train a pair of fresh browser profiles by visiting websites that reflect pro-immigration and anti-immigration stances. We show that Google can infer browsing histories of our trained profiles through its pervasive online tracking network and can leverage it to personalize search results based on recent changes in its privacy policy [2, 6]. We then execute search queries on Google News related to a variety of political topics. We analyze the search results to quantify the magnitude and direction of personalization. While personalization varies depending on the search term, overall we are able to conclude that profiles trained by browsing websites reflecting distinct political positions indeed receive significant personalization that tends to reinforce the presumed partisanship.

We highlight the key contributions of our work as follows.

• **Methodology:** We expand on previous work (e.g., [15, 16]) by refining the sock puppet methodology for auditing algorithmic personalization. Our sock puppet system is designed to reflect the behavior of Twitter accounts with opposing political views. Specifically, we train browser profiles by visiting URLs extracted from the timelines of these Twitter accounts. We then search Google News simultaneously from the trained browser profiles using different search terms related to popular political issues such as immigration and foreign policy.

• **Findings:** Our comparative analysis of the search results returned by Google News provides evidence of personalization based on browsing history. Specifically, while personalization varies across different search terms, we note that the search terms receiving most personalization tend to get personalized results that reinforce the presumed partisanship. We believe that our findings are different from those reported in prior work on personalization in part because of our refined sock puppet system design.

Our work presents empirical evidence of political personalization on Google News based on browsing history. The results not only set the baseline for search personalization based on political biases in browsing history, but also contribute to the broader understanding of selective exposure and algorithmic personalization.

2 SOCK PUPPET AUDITING SYSTEM

The overarching goal of our study is to examine the extent to which a user's web browsing history affects algorithmic personalization in web search. We seek to understand both whether algorithmic personalization exists and whether it reinforces or dampens the preferences displayed in web browsing history. To this end, we design and implement a sock puppet auditing system to train fresh browser profiles by visiting hyperlinks that embody distinct political discourses and then compare Google News search results for identical politically oriented search terms. Figure 1 depicts the four main components of our system that are discussed below.

2.1 Identifying Partisan Hyperlinks

We aim to evaluate Google News personalization based on browsing histories that reflect different stances on the topic of immigration. We focus on immigration because it was a key campaign issue in the recent 2016 U.S. presidential election [5, 23]. Our goal was to find a list of websites that reflect pro- and anti-immigration stances in the context of U.S. politics. To this end, we rely on two popular Twitter accounts that reflect both discourse communities. Discourse communities are groups of people that come together, whether physically or virtually, for the purpose of building community through shared goals and forms of speech [39]. For the anti-immigration stance, we use @wginfonetorg, which belongs to a popular anti-immigrant website whitegenocide.info that is dedicated to "fighting the crime of white genocide." For the pro-immigration stance, we use @DefineAmerican, which belongs to a popular pro-immigrant website defineamerican.com that is dedicated to "shift the conversation about immigrants, identity, and citizenship in a changing America." Both of these Twitter accounts are very active and fairly popular. As listed in Table 1, @wginfonetorg has posted more than 63K tweets and has 9.4K followers while @DefineAmerican has posted more than 17K tweets and has 30.3K followers. Because both accounts frequently share content from other websites that support their respective views on immigration, we collected the hyperlinks posted on the timelines of these Twitter accounts over the duration of two weeks in March 2017. We treat these two sets of hyperlinks as representations of pro- and anti-immigration discourses.

Screen Name	Number of Favorites	Number of Followers	Number of Followings	Number of Tweets	Account Creation
@DefineAmerican	10,043	30,253	1,776	17,350	May 2011
@wginfonetorg	1,253	9,424	223	63,916	April 2014

Table 1: Statistics of pro-immigration and anti-immigration Twitter accounts.

2.2 Training Browser Profiles

Using the two sets of hyperlinks collected from pro- and anti-immigration Twitter accounts, we now train fresh browser profiles. Specifically, to train the pro-immigration browser profile, we install a fresh copy of Firefox web browser and open hyperlinks from the timeline of the pro-immigration Twitter account. Similarly, to train the anti-immigration browser profile, we install a fresh copy of Firefox web browser and open hyperlinks from the timeline of the anti-immigration Twitter account. We take several steps to imitate a real user during browser profile training. First, we limit the number of hyperlinks that we use for training per day to 50. To this end, we randomly sample a subset of 50 hyperlinks from the hyperlinks posted on the timelines of both Twitter accounts. Second, we add a random delay averaged at five minutes between opening consecutive hyperlinks during training. Finally, we conduct the training between 9 am and 11 pm local time to reflect diurnal user activity.

It is noteworthy that we do not create or log-in to a Firefox account during training because it can save bookmarks, passwords, browsing history, and cookie information. We also do not create or log-in to a Google account that may be used by Google to link our account information with browsing history. Therefore, the only way for Google to know about our browsing history during training would be via third-party tracking using its own advertising/analytics network (e.g., DoubleClick and Google Analytics) via cookies or browser fingerprinting [12, 24]. As discussed in related work, prior research has reported on the extensive third-party tracking capabilities of Google.

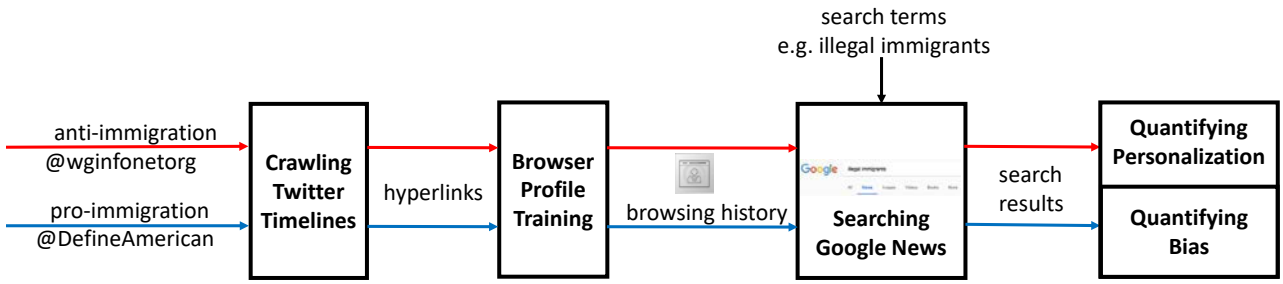


Figure 1: Our sock puppet auditing system to measure political personalization of Google News search.

Our analysis of the set of hyperlinks used for training browser profiles shows that pro-immigration training hyperlinks belong to more mainstream news outlets such as Washington Post while anti-immigration training hyperlinks belong to more alternative news sites such as Breitbart and user-generated content sites such as YouTube and WordPress. We find that several Google owned domains are top third-parties on these training hyperlinks. For example, doubleclick.net is the top third-party tracker on both anti- and pro-immigration training hyperlink sets. Thus, we conclude that Google is able to learn the browsing histories of our trained pro- and anti-immigration browser profiles and *can* later use it for personalization of search results.

2.3 Searching Google News

After training fresh browser profiles using hyperlinks crawled from pro- and anti-immigration Twitter accounts, we are set to conduct Google News searches. We are interested in testing if web browsing histories that reflect these divergent discourses would result in search personalization along politically partisan lines.[1] In other words, we want to test whether a user who consistently consumes anti-immigration content would receive more or less Republican-leaning news stories in the search results. We execute searches from three sets of browser profiles: (1) pro-immigration, (2) anti-immigration, and (3) control. The control browser profile, which is not trained (i.e., by visiting hyperlinks), provides us a "neutral" perspective to judge whether personalization occurs for pro- and/or anti-immigration profiles. We execute Google News searches using different search terms about a wide variety of political topics. We decide to search for news related to the top-10 most discussed policy issues on Twitter [23] during the 2016 U.S. presidential election campaign. We use five different search terms for each of these policy issues: Immigration, Foreign Policy, Healthcare, Economy, Abortion, Gay Rights, Gun Control, Climate Change, Education, and Veterans. For each search term, we repeat the search process every day over the duration of one week.

To ensure that the search results returned by Google are not impacted by anything other than the browsing histories, our search process is designed as follows. First, we use Selenium WebDriver [1] to automatically conduct searches using the Google News web interface instead of Google's search API. We configure Google News to return up to 100 search results. Second, we use a wait period of 11 minutes between consecutive searches to eliminate any carry-over effect [16]. Third, we conduct searches from pro-immigration, anti-immigration, and control browser profiles in a synchronized fashion to eliminate any temporal effects. Fourth, each browser profile conducts search queries on a separate Amazon EC2 cloud instance to avoid any interdependencies between different profiles.[2] Fifth, we use a static DNS entry for Google News to ensure that our search queries are routed to the same datacenter. Finally, to eliminate potential noise in search results due to A/B testing [30], we train four separate profiles for pro-immigration, anti-immigration, and control browser profiles. In the absence of A/B testing, we would expect the same search results for these four profiles. When there is A/B testing, we can mitigate its effect by eliminating search result differences. To this end, we compute the pairwise intersection of search results among the four profiles. We then use search results from a randomly selected profile from the pair with the maximum intersection.

2.4 Quantifying Search Personalization

We use trained browser profiles for both pro- and anti-immigration stances as well as a control browser profile to execute Google News searches. We can quantify personalization by comparing the presence and ranking of search results across different browser profiles. First, we quantify personalization simply in terms of the differences in search results. Let P and A respectively denote the lists of search results for the pro- and anti-immigration browser profiles. Let C denote the list of search results for the control browser profile. We can measure the differences in search results by comparing P, A, and C in a pairwise manner. To this end, let $P - A$ represent the set of unique search results for the pro-immigration profile that do not appear for the anti-immigration profile. Let $P - C$ and $A - C$ respectively represent the set of unique search results for the pro-immigration and anti-immigration profiles with respect to the control. We use $|P - A|$, $|P - C|$, and $|A - C|$ to quantify the differences in search results in the range of [0%,100%], where 0% indicates no personalized search results and 100% indicates all 100 search results are personalized. Note that since we get 100 search results for each search term, we have $|P| = |A| = 100$. In this case,

[1]Note that the extreme anti-immigration position of those concerned about white genocide are not wholly aligned with the Republican Party's nor are the political views of our pro-immigration profile identical to the Democratic Party platform.

[2]Note that while each Amazon cloud instance has a different IP address, they all belong to the same /24 IP prefix range geolocated in Oregon. Therefore, we do not expect any IP geolocation based search personalization as has been reported in prior work [16, 20].

the difference in search results is symmetric in terms of quantities (i.e. $|P - A| = |A - P|$) although it may be not be symmetric as the set (i.e. $P - A \neq A - P$). Second, we quantify personalization while taking into account the ranking of search results across different browser profiles. To this end, we measure edit distance among P, A, and C in a pairwise manner. Specifically, we compute the Damerau-Levenshtein distance [10] which represents the number of deletions, insertions, or substitutions needed to make a pair of lists identical. If two lists are identical, the edit distance is 0. The greater the edit distance, the more different the lists are in terms of their membership and ordering. Note that both the difference and edit distance are calculated based on search results as URLs and we do not consider title or text of these URLs.

In addition to quantifying personalization, we also analyze whether personalized search results reflect political bias. To assess political bias, we prefer automatic methods over manual coding because the latter are expensive and time consuming. Thus, we first try to use methods in prior literature [21, 22] that can automatically measure political bias of URLs by analyzing their sharing patterns on Twitter. Since these methods require a large number of tweets containing the news URLs, we cannot use them because very few of the personalized search results (URLs) are frequently tweeted. Therefore, we use mediabiasfactcheck.com which provides the political bias of 1,540 media sources (identified as domains) on a 5-point scale: left, left-center, center, right-center, and right [4]. For further quantification, we convert this 5-point scale to specific scores as: left = -100, left-center = -50, center = 0, right-center = 50, and right = 100. Note that mediabiasfactcheck.com provides the political bias for domains (e.g., nytimes.com) not URLs. Since we cannot automatically measure the political bias of personalized news URLs, we estimate their political bias using their domain's political bias from mediabiasfactcheck.com.

3 EVALUATION

Table 2 reports the personalization results quantified as pairwise difference and edit distance across pro-immigration (P), anti-immigration (A), and control (C) profiles.

In terms of difference, although it varies across different search terms, the average difference between pro-immigration and anti-immigration ($P - A$) is 3.2%, between pro-immigration and control ($P - C$) is 3.9%, and between anti-immigration and control ($A - C$) is 3.8%. Using the standard t-test, we are able to conclude that these average differences are significantly different from zero at 0.0001 significance level. The average edit distance between pro-immigration and anti-immigration is 8.4, between pro-immigration and control is 10.3, and between anti-immigration and control is 10.3. Using the standard t-test, we are again able to conclude that these edit distances are significantly different from zero at 0.0001 significance level. The edit distance values are higher than the difference because edit distance not only considers the difference in two lists of search results but also the changes in relative rankings of search results. Since difference and edit distance show the same trend in personalization across different search terms, we focus on the difference metric for discussion in the rest of this section.

To analyze personalization across different search terms, we sort search terms in Table 2 in descending order of $P - A$ values for each policy category. Note that the noise in search results due to A/B testing among four identical browser profiles for both pro- and anti-immigration is 0.5% on average. Thus, we focus our attention on search terms for which $P - A$ average difference exceeds 5% because it is at least an order of magnitude more than the average noise. A total of nine search terms meet this criterion: (1) comprehensive immigration reform, (2) uninsured Americans, (3) medicare for all, (4) national debt, (5) flat tax, (6) pro-life, (7) Paris climate agreement, (8) carbon footprint, and (9) support our veterans.

The comparison of pro-immigration versus anti-immigration and control profiles for these search terms reveals interesting insights.

Policies	Search Terms	Difference (%)			Edit Distance		
		$P-A$	$P-C$	$A-C$	$E(P,A)$	$E(P,C)$	$E(A,C)$
Immigra-tion	Comprehensive immigration reform	**6.3**	18.2	13.5	14.7	38.5	26.5
	White nationalism	4.3	4.2	2.7	8.5	8.7	7.2
	Dream Act	3.0	5.8	4.0	6.7	15.8	11.5
	Anchor babies	2.5	4.3	6.8	5.8	14.2	20.0
	Illegal immigrants	1.2	2.3	1.2	5.7	14.2	9.2
Foreign Policy	ISIS	1.7	0.8	1.8	4.3	2.3	5.3
	Benghazi	1.3	5.7	6.0	4.8	13	15.8
	Syria war	1.0	1.2	1.0	3.0	2.7	3.3
	Iran deal	0.5	0.5	0.3	0.8	1.0	0.5
	Aleppo	0.0	0.2	0.2	0.0	0.2	0.2
Health-care	Uninsured Americans	**7.2**	11.3	10.5	15.7	24.2	22.5
	Medicare for all	**6.2**	10.0	4.8	16.2	26.0	11.5
	Health insurance	4.3	2.0	3.0	15.3	7.2	9.5
	Affordable Care Act	2.7	5.0	6.8	8.2	17.8	24.5
	Obamacare	0.3	4.3	4.3	0.7	14.5	14.7
Economy	National debt	**12.0**	16.5	12.2	27.3	39.5	30.7
	Flat tax	**6.7**	6.2	1.0	13.0	12.0	2.0
	NAFTA	2.8	3.3	6.0	7.0	10.8	17.3
	Wall street	2.8	0.3	2.5	6.8	0.7	6.2
	Federal budget	1.5	1.3	0.7	10.2	8.2	5.0
Abortion	Pro-life	**5.7**	2.3	4.0	19.3	7.3	13.3
	Planned parenthood	3.5	7.0	5.3	9.0	14.3	13.0
	Roe v. Wade	2.8	2.2	4.3	7.0	5.8	11.8
	Pro-choice	1.7	3.0	4.0	4.8	6.7	10.7
	Women's rights	0.5	2.8	2.8	2.0	16.0	16.2
Gay Rights	LGBT	4.3	3.2	7.0	14.5	7.7	21.2
	Traditional marriage	4.2	2.8	2.8	9.7	6.2	6.0
	Gay couple	3.8	3.3	4.3	10.2	8.0	11.7
	Marriage equality	3.0	2.8	4.3	8.2	7.2	11.8
	Same-sex marriage	2.0	1.8	1.7	5.0	4.7	3.5
Gun Control	Gun license	4.3	2.2	5.7	10.8	5.0	14.3
	Background checks	4.3	2.2	2.8	13.8	6.7	8.2
	NRA	3.0	0.3	3.0	8.8	0.7	8.7
	Gun control	2.2	3.8	2.3	5.2	9.8	6.8
	Gun accessibility	0.7	0.5	0.2	5.8	3.2	2.7
Climate Change	Paris climate agreement	**7.2**	7.2	8.8	14.2	14.2	15.7
	Carbon footprint	**5.2**	5.3	3.0	11.8	12.3	5.8
	Climate debate	4.5	3.0	2.3	13.8	8.7	7.0
	Greenhouse gases	2.2	2.2	8.8	5.0	4.7	16.0
	Global warming	0.2	2.8	2.7	0.3	11.5	11.2
Educa-tion	No Child Left Behind	5.0	4.7	0.8	9.3	8.5	1.5
	Department of Education	3.3	2.7	1.5	12.2	9.7	6.8
	College affordability	2.8	3.0	0.2	10.3	10.2	0.7
	Race to the Top	1.0	1.8	2.3	3.8	6.5	9.2
	Free community college	0.3	2.7	2.8	0.7	6.0	6.3
Veterans	Support our veterans	**7.2**	5.8	5.5	15.5	13.2	12.7
	Veterans affairs	3.5	3.7	1.3	9.7	11	3.7
	Veterans	1.2	1.3	1.3	3.7	3.7	4.5
	Veteran benefits	0.8	1.2	1.2	1.7	3.0	3.0
	PTSD	0.5	3.3	3.2	0.8	8.7	8.2
Average		**3.2**	**3.9**	**3.8**	**8.4**	**10.3**	**10.3**

Table 2: Personalization (quantified using difference and edit distance) for 50 search terms. Note that $E(P,A)$, $E(P,C)$, and $E(A,C)$ are pairwise edit distance among pro-immigration, anti-immigration, and control profiles.

Top-k Search Results	Difference (%)			k-Edit Distance		
	$P-A$	$P-C$	$A-C$	$E(P,A)$	$E(P,C)$	$E(A,C)$
k = 10	2.3	2.4	2.7	5.7	5.7	6.7
k = 20	2.2	2.4	2.5	6.4	6.3	7.4
k = 30	2.4	2.5	2.7	6.6	7.3	8.0
k = 40	2.6	2.9	3.0	7.1	8.3	8.7
k = 50	2.7	3.2	3.1	7.5	8.9	9.1
k = 60	3.0	3.4	3.3	7.9	9.4	9.5
k = 70	3.2	3.7	3.6	8.3	9.9	9.9
k = 80	3.2	3.8	3.7	8.5	10.1	10.2
k = 90	3.3	3.8	3.7	8.6	10.2	10.2
k = 100	3.2	3.9	3.8	8.4	10.3	10.3

Table 3: Average personalization for all 50 search terms according to top-k (k∈{10,20,...,100}) ranked search results. We normalize edit distance (as k-edit distance) to make a fair comparison across different k values.

First, we observe that five of them (comprehensive immigration reform, uninsured Americans, national debt, Paris climate agreement, and support our veterans) also have high $P-C$ and $A-C$ differences. For example, national debt search term has 12.0% $P-A$ difference, 16.5% $P-C$ difference, and 12.2% $A-C$ difference. This shows that both pro- and anti-immigration profiles receive personalized search results that are different from each other as well as different from the control. Second, we observe that three of them (flat tax, medicare for all, and carbon footprint) have high $P-C$ difference but low $A-C$ difference. For example, flat tax search term has 6.7% $P-A$ difference, 6.2% $P-C$ difference, and 1.0% $A-C$ difference. This shows that the pro-immigration profile receives personalized search results that are different from both the control and anti-immigration profile. Third, we observe that only one of them (pro-life) has a high $A-C$ difference but low $P-C$ difference. Specifically, pro-life search term has 5.7% $P-A$ difference, 2.3% $P-C$ difference, and 4.0% $A-C$ difference. This shows that the anti-immigration profile receives personalized search results that are different from both the control and pro-immigration profile.

Note that there are search terms for which $P-A$ difference is not high (e.g., < 1.5%), but it does not necessarily mean that there is no personalization because their differences to control (both $P-C$ and $A-C$) may be high. This is because both pro- and anti-immigration profiles receive similar personalized search results with respect to the control so they are not much different from each other. For example, Benghazi search term has only 1.3% $P-A$ difference, but 5.7% $P-C$ difference and 6.0% $A-C$ difference.

Next, we study the rank of personalized search results to gauge whether they disproportionately appear at the top or bottom of the list of results. Table 3 reports the average personalization for all search terms based on the top-k (k∈{10,20,...,100}) search results. We note that average $P-A$ difference started at 2.3% for k=10, slightly increases for increasingly k values, and reaches 3.2% for k=100. We observe a similar trend for $P-C$ and $A-C$ differences as well as in terms of edit distance. Note that we normalize edit distance as k-edit distance (edit distance/k) for a fair comparison across different k values. Overall, while personalization slightly increases at bottom ranks, we conclude that personalization remains substantial for top ranked search results. Since personalization (in terms of both difference and edit distance) for several search terms exceeds the average by multiple factors, we gather that there exists significant personalization even in top-10 search results.

Search Terms	Political Bias		
	$B(P)$	$B(A)$	$B(P)-B(A)$
Carbon footprint	-30.0	10.0	-40.0
Comprehensive immigration reform	-35.3	-3.1	-32.2
Paris climate agreement	-23.3	-5.9	-17.4
Pro-life	36.1	52.8	-16.7
Support our veterans	14.3	25.0	-10.7
Uninsured Americans	-18.8	-10.9	-7.9
Flat tax	-4.5	-13.3	8.8
Medicare for all	-27.8	-37.0	9.2
National debt	11.5	-6.5	18.0
Average	-8.6	1.3	-9.9

Table 4: Political bias for nine top-personalized search terms. Note that $B(P)$, $B(A)$, and $B(P)-B(A)$ respectively are political bias of personalized search results for the pro-immigration and anti-immigration profiles, and the difference between their political bias. Table is sorted in ascending order of $B(P)-B(A)$ values. A negative $B(P)-B(A)$ value indicates the pro-immigration profile received more Democratic-leaning personalized search results, while a positive $B(P)-B(A)$ value indicates the pro-immigration profile received more Republican-leaning personalized search results.

We further analyze political bias of the personalized search results for the nine most personalized search terms in Table 2. Using mediabiasfactcheck.com, we are able to estimate the political bias of 373 news stories out of 792 (47%) personalized results for these nine most personalized search terms. Table 4 reports the average political bias of personalized search results for the pro- and anti-immigration profiles, and the difference between their political bias averages. The average political bias of pro-immigration profile is -8.6 which is Democratic-leaning and that of anti-immigration profile is 1.3 which is Republican-leaning (negative values represent Democratic-leaning and positive values represent Republican-leaning). In other words, the pro-immigration profile receives more Democratic-leaning personalized search results than that of the anti-immigration profile. We use the Kolmogorov-Smirnov test [25] to compare the political bias distributions of personalized search results for pro- and anti-immigration profiles. We are able to reject the null hypothesis that both distributions are the same at the 0.05 significance level. Thus, we conclude that the search terms receiving most personalization tend to get personalized results that reinforce the presumed partisanship. Note that political bias of personalized search results varies across different search terms. Personalization reinforces the presumed partisanship for six out of nine (carbon footprint, comprehensive immigration reform, Paris climate agreement, pro-life, support our veterans, and uninsured Americans) search terms that received most personalization. For the remaining search terms, it counters the presumed partisanship.

4 RELATED WORK

The most ubiquitous application of personalization is in online behavioral advertising, which targets users with personalized ads based on their browsing history. Several studies have reported personalization in online advertising [8, 11, 14, 41]. Researchers have also investigated personalization in terms of price steering and price discrimination on ecommerce sites [17, 27, 28, 40].

A long line of research has looked at web search personalization. Hannak et al. [16] examined several features such as browser user-agent, user's ethnicity, browsing history, and search query that Google can use to personalize search results. However, they only found significant personalization based on Google account login status and IP geolocation. In a follow-up [18], the author extended this study to Bing and DuckDuckGo. They found that, on average, 15.8% search results on Bing are different due to personalization based on the same features as on Google. They found no significant personalization on DuckDuckGo.

To further examine the role of location in triggering personalization, Xing et al. [42] used a browser extension called Bobble and reconfirmed that a user's location (inferred by geolocating IP addresses) is the most dominant trigger factor for personalization. Furthermore, Kliman-Silver et al. [20] examined the relationship between distance and location-based personalization, and the impact of location-based personalization on different types of queries. They found that the differences between search results grow as the physical distance between locations of the users increases.

Puschmann [32, 33] analyzed crowdsourced Google search results from approximately 4,000 users for 16 terms related to German political parties and candidates [3]. He found that search results are personalized primarily based on location, language, and time. He concluded that a small fraction of personalized search results, which could not be explained based on the aforementioned factors, may be trigged by latent (or non-observable) factors such as browsing history and Google account information that they did not collect from users.

Robertson et al. [35] audited personalization on Google Search by asking crowd workers to install a browser extension which collected Google search results for a variety of political search queries in standard and incognito modes. They found significant personalization based on Google account login status, self-reported usage of Google/Alphabet services such as YouTube, and self-reported rating of Trump but not self-reported political party affiliation. In a follow-up [34], the authors reported that personalized search results have minimal differences in terms of political bias.

While there is evidence of personalization in ads and ecommerce sites based on browsing history [8, 41], prior research has found Google search personalization primarily due to geolocation and account login status. In contrast to prior work, we believe that our sock puppet auditing system can help reveal the impact of browsing histories, which are explicitly designed to reflect different political ideologies, on search personalization.

5 DISCUSSION AND CONCLUSION

Overall, we observe significant personalization based solely on browsing history. The personalized results tend to reinforce the presumed partisanship of browser profiles that are trained by visiting web pages reflecting divergent political discourses. Our findings provide further empirical evidence for the underlying causes of filter bubbles or echo chambers. The explanation of why we found evidence of significant web search personalization while past work did not could be due to the following.

First, our research is different from prior work in terms of the methodology in training browser profiles. While previous work [15, 16] trained browser profiles to reflect different demographics, we trained browser profiles to reflect different political stances. Specifically, both Hannak et al. [16] and Haim et al. [15] trained browser profiles to represent different demographic groups such as gender, age, income, lifestyle, and ethnicity. In contrast, we trained browser profiles to explicitly reflect opposing views on the topic of immigration rather than different demographics. Specifically, we trained browser profiles using news stories posted by Twitter accounts who clearly demonstrated distinct political stances on the topic of immigration.

The second reason could be the difference in search terms that were used to test personalization. Prior literature reported that different search terms can trigger different magnitudes of personalization [16, 21]. Unlike previous work [15, 16, 35] which used search terms covering a variety of topics popular at the time, we used search terms related to the training topic of immigration. Specifically, Haim et al. [15] used general search terms such as "Germany". Hannak et. al. [16] used general search terms about topics such as news sources and literature. Robertson et al. [35] used general political search terms such as "US President." In contrast, after training browser profiles reflecting opposing views on the topic of immigration, we used immigration-related search terms such as "comprehensive immigration reform" and "illegal immigrants". We also used search terms about other relevant political topics [23] such as foreign policy.

Last but not least, the changing nature of Google's personalization algorithm could be another reason. Google is known to continuously tinker personalization algorithms as well as update their data sources over time. For example, Google changed its privacy policy in 2012 [2] and more recently in 2016 [6] allowing them to combine user data collected across all of its services (e.g., Search, Gmail, Google Analytics, DoubleClick) for targeted advertising and content personalization. Thus, Google can now more effectively personalize search results based on a user's browsing history inferred from its third-party analytics and tracking network. Since personalization algorithms are continuously being tweaked, we plan to longitudinally study personalization for different search terms over an extended period of time as part of our future work.

Prior work using controlled experiments as well as surveys of real users [15, 16, 32–35] has not found evidence of significant web search personalization based on browsing history. While simulating behaviors of real users completely will always be impossible, controlled experiments such as our sock puppet auditing system enable us to isolate the effect of different types of browsing history on search personalization. Thus, in this paper, we studied search personalization on Google News in a controlled setting by using browser profiles that were specifically trained to reflect strongly divergent opinions on the topic of immigration. Using this strategy, we find evidence of significant search personalization on Google News. This finding creates an opportunity to conduct further research on other factors that can trigger search personalization.

ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation (grant number 1715152) and by the Obermann Center for Advanced Studies.

REFERENCES

[1] Selenium webdriver. https://www.seleniumhq.org/projects/webdriver/.

[2] Google user data to be merged across all sites under contentious plan. The Guardian. http://bit.ly/2DluqbV, 2012.

[3] Data donation - German Parliamentary Election 2017. http://bit.ly/2t9QcLf, 2017.

[4] Media bias/fact check - the most comprehensive media bias resource. https://mediabiasfactcheck.com/, 2018.

[5] W. Andrews and T. Kaplan. Where the candidates stand on 2016's biggest issues. the new york times. http://nyti.ms/2ERzPEn, 2016.

[6] J. Angwin. Google Has Quietly Dropped Ban on Personally Identifiable Web Tracking. ProPublica. http://bit.ly/2eAjC9w, 2016.

[7] E. Bakshy, S. Messing, and L. A. Adamic. Exposure to ideologically diverse news and opinion on Facebook. *Science*, 2015.

[8] R. Balebako, P. Leon, R. Shay, B. Ur, Y. Wang, and L. Cranor. Measuring the effectiveness of privacy tools for limiting behavioral advertising. In *Web 2.0 Workshop on Security and Privacy (W2SP)*, 2012.

[9] P. R. Center. Political Polarization in the American Public: How Increasing Ideological Uniformity and Partisan Antipathy Affect Politics, Compromise and Everyday Life. http://pewrsr.ch/2n8oI2I, 2014.

[10] F. J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM (CACM)*, 7:171–176, 1964.

[11] A. Datta, M. C. Tschantz, and A. Datta. Automated experiments on ad privacy settings: A tale of opacity, choice, and discrimination. In *Privacy Enhancing Technologies Symposium (PETS)*, 2015.

[12] S. Englehardt and A. Narayanan. Online tracking: A 1-million-site measurement and analysis. In *ACM Conference on Computer and Communications Security (CCS)*, 2016.

[13] M. Garcia. Racist in the Machine: The Disturbing Implications of Algorithmic Bias. *World Policy Journal*, 33(4):111–117, 2016.

[14] S. Guha, B. Cheng, and P. Francis. Challenges in measuring online advertising systems. In *ACM Internet Measurement Conference (IMC)*, 2010.

[15] M. Haim, A. Graefe, and H.-B. Brosius. Burst of the filter bubble? effects of personalization on the diversity of google news. *Digital Journalism*, 2017.

[16] A. Hannak, P. Sapiezynski, A. M. Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove, and C. Wilson. Measuring personalization of web search. In *ACM International Conference on World Wide Web (WWW)*, 2013.

[17] A. Hannak, G. Soeller, D. Lazer, A. Mislove, and C. Wilson. Measuring price discrimination and steering on e-commerce web sites. In *ACM Internet Measurement Conference (IMC)*, 2014.

[18] A. Hannï£¡k, P. Sapie?y?ski, A. M. Khaki, D. Lazer, A. Mislove, and C. Wilson. Measuring personalization of web search. https://arxiv.org/abs/1706.05011, 2017.

[19] K. H. Jamieson and J. N. Cappella. *Echo chamber: Rush Limbaugh and the conservative media establishment*. Oxford University Press, 2008.

[20] C. Kliman-Silver, A. Hannak, D. Lazer, C. Wilson, and A. Mislove. Location, location, location: The impact of geolocation on web search personalization. In *ACM Internet Measurement Conference (IMC)*, 2015.

[21] J. Kulshrestha, M. Eslami, J. Messias, M. B. Zafar, S. Ghosh, K. P. Gummadi, and K. Karahalios. Quantifying search bias: Investigating sources of bias for political searches in social media. In *The 20th ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW)*, 2017.

[22] H. Le, Z. Shafiq, and P. Srinivasan. Scalable News Slant Measurement Using Twitter. In *AAAI International Conference on Weblogs and Social Media (ICWSM)*, 2017.

[23] H. T. Le, G. Boynton, Y. Mejova, Z. Shafiq, and P. Srinivasan. Revisiting The American Voter on Twitter. In *ACM Conference on Human Factors in Computing Systems (CHI)*, 2017.

[24] T. Libert. Exposing the Invisible Web: An Analysis of Third-Party HTTP Requests on 1 Million Websites. *International Journal of Communication*, 2015.

[25] F. J. Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46:68–78, 1951.

[26] D. B. Melican and T. L. Dixon. News on the net: Credibility, selective exposure, and racial prejudice. *Journal of Communication Research*, 35:151–168, 2008.

[27] J. Mikians, L. Gyarmati, V. Erramilli, and N. Laoutaris. Detecting price and search discrimination on the internet. In *ACM Workshop on Hot Topics in Networks (HotNets)*, 2012.

[28] J. Mikians, L. Gyarmati, V. Erramilli, and N. Laoutaris. Crowd-assisted search for price discrimination in e-commerce: first results. In *ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2013.

[29] S. U. Noble. *Algorithms of Oppression: How Search Engines Reinforce Racism*. 2018.

[30] A. Pansari and M. Mayer. This is a test. this is only a test. google official blog. http://bit.ly/2DngZbg, 2006.

[31] E. Pariser. *The Filter Bubble: How the New Personalized Web Is Changing What We Read and How We Think*. The Penguin Group, 2012.

[32] C. Puschmann. How significant is algorithmic personalization in searches for political parties and candidates? http://bit.ly/2fEhDqK, 2017.

[33] C. Puschmann. How significant is algorithmic personalization in searches for political parties and candidates? (Part II). http://bit.ly/2E7n5KL, 2017.

[34] R. E. Robertson, S. Jiang, K. Joseph, L. Friedland, D. Lazer, and C. Wilson. Auditing partisan audience bias within google search. In *The 21st ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW'18)*, 2018.

[35] R. E. Robertson, D. Lazer, and C. Wilson. Auditing the personalization and composition of politically-related search engine results pages. In *The 29th International World Wide Web Conference Committee (IW3C2)*, 2018.

[36] C. Sandvig, K. Hamilton, K. Karahalios, and C. Langbort. Auditing Algorithms: Research Methods for Detecting Discrimination on Internet Platforms. In *Data and Discrimination*, 2014.

[37] T. Speicher, M. Ali, G. Venkatadri, F. N. Ribeiro, G. Arvanitakis, F. Benevenuto, K. P. Gummadi, P. Loiseau, and A. Mislove. On the Potential for Discrimination in Online Targeted Advertising. In *Conference on Fairness, Accountability, and Transparency (FAT*)*, 2018.

[38] N. Stroud. Polarization and partisan selective exposure. *Journal of Communication*, (60):556–576, 2010.

[39] J. Swales. *Genre analysis: English in academic and research settings*. 1990.

[40] T. Vissers, N. Nikiforakis, N. Bielova, and W. Joosen. Crying wolf? on the price discrimination of online airline tickets. In *Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs)*, 2014.

[41] C. E. Wills and C. Tatar. Understanding what they do with what they know. In *ACM Workshop on Privacy in the Electronic Society (WPES)*, 2012.

[42] X. Xing, W. Meng, D. Doozan, N. Feamster, W. Lee, and A. C. Snoeren. Exposing inconsistent web search results with bobble. In *International Conference on Passive and Active Network Measurement (PAM)*, 2013.

[43] D. Zillmann and J. Bryant. *Selective Exposure to Communication*. Routledge, 2013.

[44] H. G. d. Zuniga, T. Correa, and S. Valenzuela. Selective exposure to cable news and immigration in the u.s.: The relationship between fox news, cnn, and attitudes toward mexican immigrants. *Journal of Broadcasting and Electronic Media*, 2012.

TiSSA: A Time Slice Self-Attention Approach for Modeling Sequential User Behaviors

Chenyi Lei
Alibaba Group
Hangzhou, China
chenyi.lcy@alibaba-inc.com

Shouling Ji
Zhejiang University
Hangzhou, China
sji@zju.edu.cn

Zhao Li
Alibaba Group
Hangzhou, China
lizhao.lz@alibaba-inc.com

ABSTRACT

Modeling user behaviors as sequences provides critical advantages in predicting future user actions, such as predicting the next product to purchase or the next song to listen to, for personalized search and recommendation. Recently, recurrent neural networks (RNNs) have been adopted to leverage their power in modeling sequences. However, most of the previous RNN-based work suffers from the complex dependency problem, which may lose the integrity of highly correlated behaviors and may introduce noises derived from unrelated behaviors. In this paper, we propose to integrate a novel **Ti**me **S**lice **S**elf-**A**ttention (TiSSA) mechanism into RNNs for better modeling sequential user behaviors, which utilizes the time-interval-based gated recurrent units to exploit the temporal dimension when encoding user actions, and has a specially designed time slice hierarchical self-attention function to characterize both local and global dependency of user actions, while the final context-aware user representations can be used for downstream applications. We have performed experiments on a huge dataset collected from one of the largest e-commerce platforms in the world. Experimental results show that the proposed TiSSA achieves significant improvement over the state-of-the-art. TiSSA is also adopted in this large e-commerce platform, and the results of online A/B test further indicate its practical value.

CCS CONCEPTS

• **Information systems → Recommender systems**; • **Computing methodologies → Neural networks**; *Ranking*.

KEYWORDS

Attention Mechanism; Recurrent Neural Networks; User Modeling

ACM Reference Format:
Chenyi Lei, Shouling Ji, and Zhao Li. 2019. TiSSA: A Time Slice Self-Attention Approach for Modeling Sequential User Behaviors. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313495

1 INTRODUCTION

Due to the increasing abundance of information on the Web, helping users filter information according to their current intents/preferences is more and more required, which emphasizes the importance of personalized search and recommendation. Analogous to the assumption in natural language processing (NLP) that the topic of an article can be represented by the sequential words or sentences of the article [22], user's current intents/preferences can be captured by his/her sequential behaviors over the timeline. To model sequential user behaviors for downstream applications, recurrent neural network (RNN) has been considered [14, 26, 29], due to its remarkable performance on many sequential learning problems. For example, [26] endowed both users and movies with a long- and short-term memory (LSTM) to predict user's future behavioral trajectories. Despite a certain success of above RNN-based methods, modeling sequential user behaviors remains a challenging problem.

One issue is that long-term dependencies are still very hard to preserve while maintaining middle- and short-term user behaviors. Recent work has indicated that attention mechanism can help to achieve state-of-the-art performance on a large number of NLP and computer vision (CV) tasks, such as neural machine translation [1] and image captioning [13]. Attention mechanism introduced into deep networks provides the functionality to focus on portions of input data or features to fulfill the given task, which might bring benefits for the problem of complex dependencies within user historical behaviors.

However, previous work along this line for modeling sequential user behaviors has some limitations [16, 19, 28, 31]. Both of [16, 19] only considered the primary attention mechanisms, which need to be better designed. Furthermore, they employ the bi-directional RNN structure, which is quite complicated and makes it challenging to be parallelized in the real world. [28, 31] abandoned RNN structures and proposed the self-attention architecture instead for different type sequential user behaviors. Nevertheless, the sequence structure information is incomplete or even lost in the RNN-free framework, whereas RNN's autoregressive modeling can, in theory, capture potentially infinitely long-term sequence with a small number of parameters [3].

Another critical issue is that most of the previous work projects each user action into an individual embedding vector. However, user sequential behaviors are usually not consistent with action time and intents, which is hard to be distinguished and encoded [32]. Consequently, the single user action embedding is not always robust. In recent work, [20] demonstrated that taking session-level representations could have very different expressive power. Nevertheless, it is not still reasonable to divide sequential user behaviors according to fix-length time window or online session of users. Furthermore, the relationships among sessions are not considered in previous session-based related work [8, 9].

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313495

These two issues need to be optimized in advanced approaches and systems. In this paper, we propose a time slice self-attention approach and corresponding ranking system, called TiSSA, for modeling sequential user behaviors, which we currently use for the personalized recommendation. We first project user sequential behaviors into session-level representations by our proposed time-interval-based gated recurrent unit (GRU). Then we split these session-level representations into slices according to the multi-scale time window. The intra-slice self-attention is performed on each slice independently, to capture the local dependency within each slice. We further perform inter-slice self-attention for outputs of each slice to exploit the global dependency. Finally, a feature fusion gate combines the outputs of intra- and inter-slice self-attention to generate context-aware representations for the user. We perform vanilla attention between these context-aware representations and the ranking item vectors, whose outputs are fed into a ranking neural network for making recommendations. The effectiveness of our adapted approach and ranking system is evaluated in our offline and online experiments on Tmall[1].

2 BACKGROUND AND RELATED WORK

2.1 Gated Recurrent Units

A gated recurrent unit (GRU) was proposed by [5] to overcome drawbacks of normal RNNs, i.e., vanishing and exploding gradients. Specifically, the activation h_t of the GRU at time t is a linear interpolation between the previous activation h_{t-1} and the candidate activation $\tilde{h}_t$:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{1}$$

where $\odot$ is an element-wise multiplication, and an update gate z_t decides how much the unit updates its activation. The update gate is computed by

$$z_t = \sigma(W_z^{(1)} x_t + W_z^{(2)} h_{t-1}) \tag{2}$$

where x_t is input at time t, and σ is sigmoid function. The candidate activation $\tilde{h}_t$ is computed by

$$\tilde{h}_t = tanh(W_h^{(1)} x_t + W_h^{(2)} (r_t \odot h_{t-1})) \tag{3}$$

where r_t is a reset gate and computed by

$$r_t = \sigma(W_r^{(1)} x_t + W_r^{(2)} h_{t-1}) \tag{4}$$

In Eq. 2 - Eq. 4, weight parameters W_z^*, W_h^* and W_r^* connect different inputs and gates.

2.2 Attention Mechanism

Vanilla attention is introduced by [1] firstly in the encoder-decoder framework, to provide more accurate alignment for each position in the machine translation task. In particular, given token embeddings of a source sequence $\mathbf{x} = \{x_1, x_2, ..., x_n\} \in \mathcal{R}^{d \times n}$ and the vector of a query $q \in \mathcal{R}^d$, where d is the feature size of x_i and q, attention computes the alignment score between x_i and q by a function $f(x_i, q) \in \mathcal{R}$. A softmax function then transforms the scores $\{f(x_i, q)\}$ to a probability distribution $p(a|\mathbf{x}, q)$ by normalizing over all the n tokens of $\mathbf{x}$. Here a is an indicator of which token in $\mathbf{x}$ is important to q on specific task. Then the attention-based pooling can be formularized as $C = \sum_{i=1}^{n} p(a = i|\mathbf{x}, q) x_i \in \mathcal{R}^d$. In reality, additive attention is usually used, whose $f(x_i, q)$ is formulated as

$$f(x_i, q) = w^T \sigma(W^{(1)} x_i + W^{(2)} q) \tag{5}$$

where w is a weight vector to project additive function $\sigma(W^{(1)} x_i + W^{(2)} q)$ to a scalar.

Self-attentions are also studied in different mechanisms [25], which are very expressive and flexible for capturing inner-relations of the data at the encoder side. "Token2entirety" self-attention [12, 15] is one of widely used self-attention mechanism, which explores the importance between token x_i and the entire sequence $\mathbf{x}$, and compresses the sequence $\mathbf{x}$ into a vector. It removes q from Eq. 5 and adds bias term, i.e.,

$$f(x_i) = w^T \sigma(W^{(1)} x_i + b^{(1)}) + b^{(2)} \tag{6}$$

Recently, [23] further demonstrated the advantage of mask-style "token2token" self-attention mechanism, which explores the dependency between token x_i and token x_j from the same sequence $\mathbf{x}$, and generates context-aware coding for each element. At the same time, lots of work have shown that linear projections over multi-subspaces could improve the performance in various tasks [25, 31].

Another work related to our paper is hierarchical attention mechanism, which is also demonstrated as a practical design in CV/NLP tasks [17, 24, 27]. It is also a valid design for our problem, as it addresses the local and global dependency issue, which profoundly influences the accuracy of user intention summarization.

3 TWO PROPOSED IMPROVEMENTS

We propose two significant improvements to GRU structure and basic attention mechanism in this paper.

3.1 Time-GRU

Since GRU is originally designed for NLP tasks, there is no consideration of time intervals within inputs, which are very important for modeling sequential user behaviors. Furthermore, as discussed in Section 1, we argue that taking session-level inputs could have very different expressive power comparing to inputs of individual user behaviors (c.f. Section 5.4 for an in-depth analysis). Because that user behavior sessions are changing along time with the property of overlapping [7], the natural idea is to divide and embed user behaviors into sessions over behavior time smoothly. Inspired by some previous work with multi-timescale designs [18], we propose a time-interval-based GRU, called Time-GRU, to model user session-level representations. Specifically, we rewrite reset gate as a time-dependent format:

$$r_t = \sigma(W_r^{(1)} x_t + W_r^{(2)} h_{t-1} + W_r^{(3)} \triangle t_t + b_r) \tag{7}$$

Similarly, we have time-dependent update gate:

$$z_t = \sigma(W_z^{(1)} x_t + W_z^{(2)} h_{t-1} + W_z^{(3)} \triangle t_t + b_z) \tag{8}$$

In Eq. 7 and Eq. 8, $\triangle t_t \in \mathcal{R}$ is the time interval between adjacent actions, and b_* are bias terms. Our motivation is to utilize time interval information as a filter. In this way, the final activation h_t

[1] www.tmall.com, one of the largest e-commerce platforms in the world.

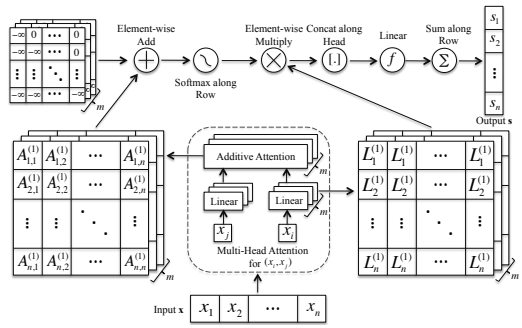


Figure 1: The proposed directional multi-head attention mechanism. Here, we use $A_{i,j}^{(k)}$ and $L_i^{(k)}$ to denote $head_{i,j}^{(k)} \in \mathcal{R}$ and $(x_i W_{ck}) \in \mathcal{R}^{d_c}$ in Eq. (9), respectively, where $k \in m$.

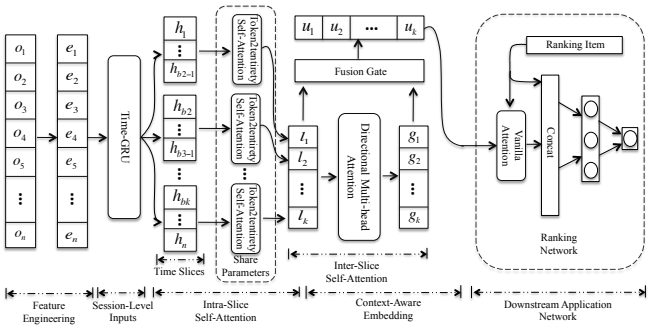


Figure 2: The proposed Time Slice Self-Attention approach and ranking system.

captures the session-level information by time t over changing time smoothly.

3.2 Directional Multi-Head Attention

Inspired by previous work [23, 31], we propose a new token2token self-attention mechanism to introduce the advantage of linear projections over multi-subspaces, which relates elements at directional positions from a single sequence by computing the multi-head attention [25] between each pair of tokens and generates context-aware coding for each $x_i \in \mathbf{x}$. In particular, as illustrated in Fig. 1, multi-head attention between each $x_i \in \mathbf{x}$ and $x_j \in \mathbf{x}$ is firstly computed. We linearly project x_i and x_j, which are both d-dimensional vectors, m times with different, learned linear projections to d_c and d_q dimensions, respectively. Then attention heads are computed. Specifically, for the k-th head of x_i and x_j, we have

$$head_{i,j}^{k} = w^T \sigma(W^{(1)}(x_i W_{ck}) + W^{(2)}(x_j W_{qk})) \tag{9}$$

where the projections are parameter matrices $W_{ck} \in \mathcal{R}^{d \times d_c}$, $W_{qk} \in \mathcal{R}^{d \times d_q}$, and $head_{i,j}^{k} \in \mathcal{R}, k \in m$. In practice, we set $d_c = d_q = d/m$. All attention heads are organized as a matrix $Rel \in \mathcal{R}^{n \times n \times m}$, whose each element is an attention scalar. Linear projections of x_i are organized as another matrix $Pro \in \mathcal{R}^{n \times n \times m}$, whose each row has the same element $(x_i W_{ck}) \in \mathcal{R}^{d_c}$.

In order to preserve temporality information in sequence, a directional mask matrix M with the same dimension with Rel is proposed,

$$M_{ij}^{k} = \begin{cases} 0, & i < j \\ -\infty, & otherwise \end{cases} \tag{10}$$

Given M and Rel, we perform an element-wise add operation, and follow a softmax normalization, each element of whose results is the attention weight for corresponding element in Pro. After weighting Pro with element-wise multiplication, matrix Pro is concatenated along attention heads as a new matrix with $n \times n$ dimensions, whose each element is a $(m \times d_c)$-dimentional vector and then projected to d-dimentional vector with a sharing learnable linear function. Finally, the projected matrix is summed along matrix row to generate token2token self-attention for all elements from $\mathbf{x}$, which is $\mathbf{s} = \{s_1, s_2, ...s_n\} \in \mathcal{R}^{d \times n}$ with the same dimensions with $\mathbf{x}$.

4 TIME SLICE SELF-ATTENTION APPROACH AND RANKING SYSTEM

A user can be represented as all his/her sequential behaviors $u = \{(o_i, t_i) | i = 1, 2, 3, ..., n\}$, where (o_i, t_i) indicates the interaction between user u and object o_i at timestamp t_i. For recommendation task, the objective of modeling sequential user behaviors is to predict the conditional probability of user's next action $p(o|u)$ for a certain given user u.

In the following of this section, we introduce our proposed time slice self-attention approach (TiSSA) and ranking system in detail, and the overall system is illustrated in Fig. 2. We divide the framework into several parts for a better description.

Feature Engineering. In our work, we take the user behavior object with deep and wide features as inputs, which is illustrated in Fig. 3. Each object o is represented by a multi-hot vector $o = \{v_1, v_2, ..., v_F\}$, where v_i indicates i-th one-hot feature such as ID-features and discretized statistical features. Following multi-hot vector inputs, behavior embedding layer transforms the multi-hot vector of o_i into a low-dimensional dense vector $e_i \in \mathcal{R}^{d_e}$ by linear mapping each feature of o_i, e.g., item id, to a dense vector and concatenating them. Note that, we enable embedding sharing when objects have features in common, e.g, item id and shop id, referring to the same entities. Then we have the behavior embedding vector for acted object o_i,

$$e_i = [W_{emb}^{(1)} v_1, W_{emb}^{(2)} v_2, ..., W_{emb}^{(F)} v_F] \tag{11}$$

where $[.]$ denotes concatenating operation and $W_{emb}^{(i)}$ is learnable embedding function parameters for i-th feature.

Session-Level Inputs. As discussed in Section 1 and 2, we prefer to employ session-level representations as inputs for better capacity. In particular, we put behavior embedding vectors $e \in \mathcal{R}^{d_e \times n}$ into our proposed Time-GRU sequentially, whose hidden states $\mathbf{h} \in \mathcal{R}^{d_h \times n}$ represent the session-level inputs. In Time-GRU, sessions of user behaviors are distinguished by time intervals smoothly.

Hierarchical Self-Attention. Since much previous work indicates that time information has substantial power in modeling user intents, we further divide session-level inputs $\mathbf{h}$ into k multi-scale time slices, which is proved as a simple yet efficient method (c.f. Section 5.4). Different user behavior platforms have different granularities of time slicing, which are hyper-parameters. After that,

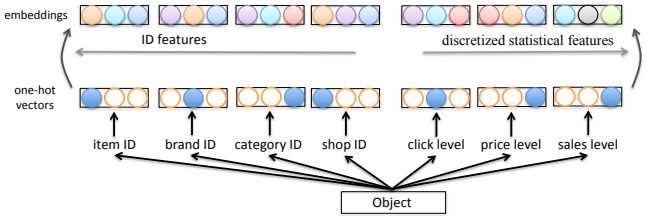


Figure 3: Feature engineering, which integrates both ID-features and statistical features.

we perform intra-slice and inter-slice self-attention hierarchically for modeling sequence dependencies at different levels.

Intra-Slice Self-Attention. We have two considerations for this level of self-attention. One is to project variable-length behaviors in different time slices into a fix-length encoding vector (zero-vector for empty time slice). The other is to capture the local dependency within a time slice. In particular, we employ token2entirety self-attention (c.f. Section 2.2) for this purpose, where shares all parameters in Eq. 6. Finally, local-context representations $\mathbf{l} = \{l_1, l_2, ..., l_k\} \in \mathcal{R}^{d_l \times k}$ at intra-slice level are generated by intra-slice self-attention, where k is the amount of time slices.

Inter-Slice Self-Attention. Another important challenging for modeling sequential user behaviors is to exploit the long-range global dependency among time slices. We perform our proposed directional multi-head self-attention (c.f. Section 3.2) after the intra-slice self-attention mechanism, which can not only capture dependencies but also preserve temporal orders. The outputs of inter-slice self-attention operation have the same dimension with inputs $\mathbf{l}$, which are denoted as $\mathbf{g} = \{g_1, g_2, ..., g_k\} \in \mathcal{R}^{d_g \times k}$.

Feature Fusion. The final context-aware representations $\mathbf{u} \in \mathcal{R}^{d_u \times k}$, where $d_u = d_g = d_l$, for each time slice is obtained by combining local and global context representations with a fusion gate. Motivated by the update processing of GRU, we design feature fusion gate as follows,

$$F = \sigma(W_F^{(1)}\mathbf{g} + W_F^{(2)}\mathbf{l} + b_F) \tag{12}$$

$$\mathbf{u} = F \odot \mathbf{l} + (1 - F) \odot \mathbf{g} \tag{13}$$

where W_F^* and b_F are learnable parameters.

Downstream Application Network. With generated context-aware representations, we can ensemble various kinds of neural networks according to downstream task requirement. In this paper, we focus on evaluating predictions of interactions within the user and items, and we set the downstream application network to be a point-wise ranking network. For each candidate item, its embedding vector q is calculated by the same embedding layer in our framework, and vanilla attention between $\mathbf{u}$ and q is performed to produce the attention pooling vector for the given user. After a concatenating operation and two fully connected layers, we have final sigmoid cross entropy loss function:

$$-\sum_{u,q} y_q \ log\, \sigma(f(u,q)) + (1 - y_q) \ log\, (1 - \sigma(f(u,q))) \tag{14}$$

where $y_q \in \{0, 1\}$ is the ground truth that indicates the user interacts the candidate item or not, and f is the downstream application network.

It is notable that feature engineering strategy can apply to many machine learning models where the designed features and ID-features exist. Time slice self-attention approach can be generalized to handle tasks involving sequential data with time intervals. The final context-aware representations can be further utilized by various downstream applications such as the clustering task.

5 EXPERIMENT

5.1 Offline Dataset

This huge dataset is real data and collected from Tmall, which describe the entire page viewed items and user feedbacks. Each instance contains user historical sequential clicked items with timestamps and the observed items currently. All items contain item id, category id, shop id, brand id and pre-designed discretized statistical features, including price level, sales level and click level. In our task, the target is to predict whether the user would have click/purchase behavior for each item in currently viewed items. 15 days user click-through behaviors are sampled for training, and we evaluate the data of the next day. Related statistics are shown in Table. 1.

5.2 Compared Approaches

We compared our proposed TiSSA approach and ranking system against several baselines from different aspects.

libFM. The task in this paper can also employ traditional recommendation methods that capture the content and collaborative information. In our work, we use the popular libFM model [21] as a representative of these methods.

Wide and Deep network (WD) [4]. This method only employs the wide and deep features of current items to construct deep neural network without consideration of sequential information.

Time-GRU. The method in [30] is employed, whose LSTM structure are replaced as our proposed Time-GRU in this paper because of the adoption of Time-GRU in our framework.

Time-GRU + Attention. A vanilla attention is added on top of Time-GRU method.

EDRec [16]. EDRec employs a bi-directional RNNs as an encoder for modeling user sequential behaviors and a traditional vanilla-attention-based RNN as a decoder to predict the user's next action.

ATRank[2] [31] and **SHAN** [28]. Two very recent state-of-the-art work on modeling user behaviors, which only utilize fully connected layer with specially designed attention mechanism. Both of them employ the same downstream application network and loss function with TiSSA.

Variant TiSSA. To justify each component in our proposed TiSSA approach, we separate out each component in TiSSA systematically, including TiSSA without RNNs (TiSSA w/o RNNs), TiSSA without Time Slices (TiSSA w/o TiS), TiSSA without Inter-Slice Self-Attention (TiSSA w/o Inter-SSA), TiSSA without multi-head attention (TiSSA w/o Multi-head) and TiSSA with Fix-Scale Time Slices (TiSSA w. Fix-TiS).

5.3 Implementations and Metrics

Our method and all compared approaches use common raw features as inputs, whose hyper-parameters are tuned on the validation

[2]https://github.com/jinze1994/ATRank

Table 1: Statistics of offline dataset. PV denotes page view. Avg. Len. and Med. Len. indicate average and median length of user historical sequential behaviors, respectively.

Data	Users	Items	Categories	Shops	Brand	PV	Avg. Len.	Med. Len.
Amount	62.6 M	27.5 M	10.6 K	0.18 M	0.16 M	1.2 B	85.8	46

set using random search [2]. All approaches are implemented on the TensorFlow machine learning system, utilizing 100 parameter servers and 2000 workers, each of which runs with 15 CPU cores.

For libFM, we extract tuples $\langle user_id, item_id, label \rangle$, where the label is 1 if the user has an action on the item and 0 otherwise. Then the multi-hot vectors of ID-features and discretized statistical features are concatenated to be $\mathbf{x}$ and the label is $\mathbf{y}$ in libFM. For other methods based on neural network, we have following settings.

Feature Engineering. All methods based on neural network employ the same feature embedding network (c.f. Section 4). All embeddings are randomly initialized with mean value 0.0 and standard deviation 1.0. We set the embedded dimension size of item IDs and category IDs, shop IDs, brand IDs as 64, 32, 32, 32, respectively. All discretized statistical features are embedded into 16 dimensions, which include click level, price level and sales level. All of these embedding dimensions result from tuning on the validation set.

Network Shape. All hidden states sizes in the RNNs structure are set to be 64. For variant TiSSA except for TiSSA w. Fix-TiS, we set time slices as $[0, 1)$, $[1, 2)$, $[2, 4)$, ..., $[2^7, +\infty)$ (#hour) based on the prior knowledge of the property of time decay, and set the amount of attention heads of inter-slice self-attention as 8 according to [25]. Time slices of TiSSA w. Fix-TiS are $[0, 2)$, $[2, 4)$, $[4, 6)$, ..., $[14, +\infty)$ (#day) for comparison. Sizes of remaining fully connected layers of all approaches are set to be 128, which also results from tuning on the validation set.

Batch Size. The batch size of all methods is set to be 512. All the batches in the data set are trained for two epochs.

Optimizer. We use AdaGrad [6] as the optimizer for all approaches. The initial learning rate is 0.01 for variant TiSSA, and 0.1 for rest approaches. The l_2-loss weight of regularization is set to 10^{-6} for RNN-based methods, and 5×10^{-5} for rest methods.

Note that, items embedding can be pre-computed and stored, and each hidden state of GRU only depends on the last one. In the inference procedure of real-world applications, TiSSA can save historical hidden states and employ parallel computing processes similar to RNN-free models, e.g., ATRank, after computing one GRU cell. Specifically, in our practical online large e-commerce platform, for one user and a batch of 3K items, TiSSA runs less than 50ms to give ranking results.

To measure the performance of each model, we follow the practice in [31]. Specifically, Area Under ROC Curve (AUC) is employed. The larger the value of AUC, the better the performance is.

5.4 Experimental Results on Offline Dataset

Firstly, we report the overall performance. Table. 2 shows the AUC results for measuring the performance of different methods. According to overall AUC results, we can observe that TiSSA achieves significant improvement than all other methods, which demonstrates the rationality of our motivations and the effectiveness of TiSSA approach on modeling sequential user behaviors, on the macro level. Next, we make some comparisons and summarize our findings as follows.

Table 2: Comparison results on offline dataset. Overall AUC denotes AUC for the entire dataset, while Long-Term AUC indicates AUC for users with more than 150 historical behaviors and 3 active days. Bold typeset indicates the best performance. The improvements from baselines to TiSSA is statistically significant according to two-tailed t-test ($p < 0.05$).

Method	Overall AUC	Long-Term AUC
libFM	0.696	0.695
WD	0.697	0.697
Time-GRU	0.702	0.698
Time-GRU + Attention	0.709	0.714
EDRec	0.709	0.711
ATRank	0.724	0.721
SHAN	0.714	0.711
TiSSA w/o RNNs	0.732	0.726
TiSSA w/o TiS	0.729	0.733
TiSSA w/o Inter-SSA	0.735	0.732
TiSSA w/o Multi-head	0.740	0.739
TiSSA w. Fix-TiS	0.737	0.738
TiSSA	**0.742**	**0.744**

Baselines libFM and WD captures content and collaborative filtering features at the same time but fails to capture the sequential information of user behaviors, while other methods capture all these information. This fact explains why libFM and WD perform worse than other models.

Due to the limitation of Time-GRU in modeling complex dependency, Time-GRU performs worse than Time-GRU + Attention and TiSSA, which reveals the benefits of attention mechanism. TiSSA w/o RNNs exhibits the improvement of overall AUC by a remarkable margin comparing to ATRank and SHAN. This result owes to the advantage that our proposed time slices hierarchical self-attention mechanism cannot only capture local dependency but the long-range global dependency of user historical behaviors efficiently. TiSSA, ATRank and SHAN outperform EDRec by a significant improvement indicate that special designed attention-mechanism brings much more benefits.

Comparing TiSSA to TiSSA w/o RNNs, TiSSA achieves improvements as much as 1.4% for AUC. As the only difference between

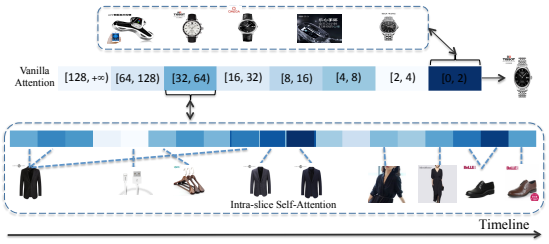


Figure 4: (Best view in color). Case study of making prediction on whether the user interacts with next item according to historical sequential behaviors. Deeper color in the heat map indicates higher attention weight.

them is whether taking session-level representations as inputs, this result validates the motivation of adoption of Time-GRU for transforming individual behaviors into session-level representations. In the comparison results of TiSSA w/o TiS, TiSSA w/o Inter-SSA and TiSSA, we demonstrate the effectiveness of the hierarchical self-attention designs. AUC declines when we remove multi-head attention in inter-slice attention mechanism. One explanation is that multi-head attention captures the relationship of inter-slice more detailed by projecting source token into multiple subspaces. Moreover, the result of TiSSA w. Fix-TiS is worse than TiSSA, which is consistent with previous studies on user behaviors on e-commerce websites [10], e.g., the property of time decay.

For the in-depth analysis of the problem of long- and short-term dependencies, we also report long-term AUC results in Table. 2, which relies heavily on the modeling of long- and short-term dependencies. The results show that TiSSA achieves the best performance against all other methods. We also observe that performances of methods without RNNs or attention mechanism drop a lot compared with overall AUC, which is reasonable for the advantage of RNNs + Attention framework at modeling long-term sequential data. Long-term AUC of TiSSA w/o Inter-SSA also drops due to the capacity of Inter-SSA for capturing long-range global dependency.

Table. 3 shows average attention scores for different time slices over all offline Dataset, which decrease with time generally except middle slices, e.g., [8, 16). It is because of the properties of user behaviors on e-commerce websites, such as time decay and cross-time dependency. Then we present a real case in Fig. 4. The middle heat map is attention scores for different time slices with target item (a mechanical watch), which is easy to understand the reason that [0, 2) slice has the highest attention score (the top line of Fig. 4). It is interesting to find that [32, 64) slice has the second high attention score. With in-depth analysis, we can find that the user has lots of interactions with formal wear at [32, 64) slice (the bottom line of Fig. 4), which is highly related to the target item and captured by our context-aware representations produced by TiSSA. Intra-slice self-attentions of [32, 64) slice are also shown in Fig. 4.

5.5 Online A/B Test on Tmall

Online test with real-world e-commerce users is carried out to study the effectiveness of our proposed method. In particular, we integrate TiSSA into the search engine of Tmall. A standard A/B test is conducted online [11]. Users of Tmall are randomly divided into multiple buckets, and we randomly select two buckets for experiments. For users in A bucket, we use the existing highly optimized ranking solution of Tmall search engine, which ensembles results from many powerful models. For users in B bucket, we further integrate the results produced by TiSSA. Specifically, for a given user, his/her context-aware representations are produced by TiSSA in real time according to collected sequential behaviors. All recall items of the search engine are preformed downstream application network in this paper and produced an interacting probability to attend existing online item ranking strategy, i.e., LTR. The test was performed within 12 days, and comparative results are given in Table. 4. It can be observed that our proposed TiSSA achieves 1.56% and 2.09% relative improvement for uCTR and uCVR than the baseline, respectively. We perform the statistical test and the resulting p-values for uCTR and uCVR are 9.1×10^{-12} and 0.0414, respectively, both less than 0.05. Thus, we think the improvement is significant. Such improvements are significant for Tmall search engine systems and have significant business value. Furthermore, the key metric to optimize in Tmall, i.e. GMV (Gross Merchandise Volume), has an improvement of 3.66%. Considering the traffic of Tmall, it would result in a significant boost in revenue. TiSSA has been adopted into the search engine of Tmall.

6 CONCLUSION

In this paper, we propose a time slice hierarchical self-attention approach to generate user context-aware representations for modeling sequential user behaviors called TiSSA, which captures both local and long-range global dependency of user actions. We also propose a Time-GRU structure to produce session-level inputs for better robustness and a directional multi-head attention. The empirical evaluations on a huge dataset collected from the real world for prediction of user interaction task show that our proposed approach can be the new state-of-the-art solution for related tasks. The ablation analyses on the separate parts of our TiSSA demonstrate the importance of each proposed component in our model. Results of online A/B test on Tmall search engine further demonstrate its practical value.

7 ACKNOWLEDGEMENT

This work was partly supported by NSFC under No. 61772466 and U1836202, the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under No. LR19F020003, the Provincial Key Research and Development Program of Zhejiang, China under No. 2017C01055, and the Alibaba-ZJU Joint Research Institute of Frontier Technologies.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.
[2] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research* (2012), 281–305.
[3] Shiyu Chang, Yang Zhang, Wei Han, Mo Yu, Xiaoxiao Guo, Wei Tan, Xiaodong Cui, Michael Witbrock, Mark Hasegawa-Johnson, and Thomas S. Huang. 2017. Dilated recurrent neural networks. In *NIPS*.
[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide and deep Learning for recommender systems. *arXiv:1606.07792* (2016).

Table 3: Average vanilla attention scores for different time slices over the entire offline dataset.

Time Slices (#Hour)	[0, 2)	[2, 4)	[4, 8)	[8, 16)	[16, 32)	[32, 64)	[64, 128)	[128, $+\infty$)
Avg. Attention Score	0.2067	0.1242	0.1139	0.1446	0.1329	0.1182	0.0916	0.0678

Table 4: Results of online A/B test. The user Click Through Rate (uCTR), user Click Conversion Rate (uCVR) and Gross Merchandise Volume (GMV) are adopted for measuring. Improvement indicates a relative growth of Group B compared to Group A, e.g. $1.56\% \approx (44.8567 - 44.1667)/44.1667$**. The absolute values of GMV are omitted for business confidential.**

	uCTR	uCVR	GMV
Group A (baseline)	44.1667 ± 0.261%	8.8767 ± 0.22%	–
Group B (TiSSA)	44.8567 ± 0.234%	9.0622 ± 0.227%	–
Improvement	1.56%	2.09%	3.66%

[5] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *arXiv:1406.1078*.

[6] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12 (July 2011), 2121–2159.

[7] Carlos A. Gomez-Uribe and Neil Hunt. 2016. The netflix recommender system: algorithms, business value, and innovation. *ACM Transactions on Management Information Systems* 6, 13 (January 2016).

[8] BalÃązs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *arXiv:1706.03847*.

[9] BalÃązs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *RecSys*. ACM, Boston, Massachusetts, 241–248.

[10] Peng Jiang, Yadong Zhu, , Yi Zhang, and Quan Yuan. 2015. Life-stage Prediction for Product Recommendation in E-commerce. In *KDD*. ACM, Sydney, NSW, Australia, 1879–1888.

[11] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M. Henne. 2009. Controlled experiments on the web: survey and practical guide. *Data Mining and Knowledge Discovery* 18, 1 (February 2009), 140–181.

[12] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *ICLR*.

[13] Chenxi Liu, Junhua Mao, Fei Sha, and Alan Yuille. 2017. Attention correctness in neural image captioning. In *AAAI Conference on Artificial Intelligence*. AAAI, 4176–4182.

[14] Qiang Liu, Shu Wu, and Liang Wang. 2017. Multi-behavioral sequential prediction with recurrent log-bilinear model. *IEEE Transactions on Knowledge and Data Engineering* 29, 6 (June 2017), 1254–1267.

[15] Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning natural language inference using bidirectional lstm model and inner-attention. In *arXiv:1605.09090*.

[16] Pablo Loyola, Chen Liu, and Yu Hirate. 2017. Modeling user session and intent with an attention-based encoder-Decoder architecture. *RecSys* (2017), 147–151.

[17] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2017. Hierarchical question-image co-attention for visual question answering. In *NIPS*.

[18] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. 2016. Phased LSTM: accelerating recurrent network training for long or event-based sequences. In *NIPS*.

[19] Wenjie Pei, Jie Yang, Zhu Sun, Jie Zhang, Alessandro Bozzon, and David M. J. Tax. 2017. Interacting attention-gated recurrent networks for recommendation. In *CIKM*. ACM, Singapore, Singapore, 1459–1468.

[20] Massimo Quadrana, Alexandros Karatzoglou, BalÃązs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *RecSys*. ACM, Como, Italy.

[21] Steffen Rendle. 2012. Factorization machines with libfm. *TIST* 3 (2012).

[22] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hananneh Hajishirzi. 2017. Bi-directional attention flow for machine comprehension. In *ICLR*. Toulon, France.

[23] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. DiSAN: directional self-Attention network for RNN/CNN-free language understanding. In *AAAI Conference on Artificial Intelligence*. AAAI.

[24] Tao shen, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. 2018. Bi-directional block self-attention for fast and memory-efficient sequence modeling. *arXiv:1804.00857* (2018).

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

[26] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent recommender networks. In *WSDM*. ACM, Cambridge, United Kingdom, 495–503.

[27] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL*.

[28] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential recommender system based on hierarchical attention network. In *IJCAI*.

[29] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model fornext basket recommendation. In *SIGIR*. ACM, Pisa, Italy, 729–732.

[30] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014. Sequential click prediction for sponsored search with recurrent neural networks. In *AAAI Conference on Artificial Intelligence*. AAAI.

[31] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. ATRank: an attention-based user behavior modeling framework for recommendation. In *AAAI Conference on Artificial Intelligence*. AAAI.

[32] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to do next: modeling user behaviors by time-lSTM. In *IJCAI*. 3602–3608.

Search Mindsets

Understanding Focused and Non-Focused Information Seeking in Music Search

Ang Li*
University of Pittsburgh
Pittsburgh, PA
ANL125@pitt.edu

Jennifer Thom
Spotify
Boston, MA
jennthom@spotify.com

Praveen Chandar
Spotify
New York, NY
praveenr@spotify.com

Christine Hosey
Spotify
Boston, MA
chosey@spotify.com

Brian St. Thomas
Spotify
Boston, MA
brianstt@spotify.com

Jean Garcia-Gathright
Spotify
Boston, MA
jean@spotify.com

ABSTRACT

Music listening is a commonplace activity that has transformed as users engage with online streaming platforms. When presented with anytime, anywhere access to a vast catalog of music, users face challenges in searching for what they want to hear. We propose that users who engage in domain-specific search (e.g., music search) have different information-seeking needs than in general search. Using a mixed-method approach that combines a large-scale user survey with behavior data analyses, we describe the construct of search mindset on a leading online streaming music platform and then investigate two types of search mindsets: focused, where a user is looking for one thing in particular, and non-focused, where a user is open to different results. Our results reveal that searches in the music domain are more likely to be focused than non-focused. In addition, users' behavior (e.g., clicks, streams, querying, etc.) on a music search system is influenced by their search mindset. Finally, we propose design implications for music search systems to best support their users.

CCS CONCEPTS

• **Information systems → Users and interactive retrieval**.

KEYWORDS

music search, information need, mixed methods

ACM Reference Format:
Ang Li, Jennifer Thom, Praveen Chandar, Christine Hosey, Brian St. Thomas, and Jean Garcia-Gathright. 2019. Search Mindsets: Understanding Focused and Non-Focused Information Seeking in Music Search. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313627

*The author completed this work as part of an internship at Spotify.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313627

1 INTRODUCTION

Music listening has long been recognized as an important part of everyday life. As proposed by Merriam in 1964, *"... There is probably no other human cultural activity which is so all-pervasive and which reaches into, shapes and often controls so much of human behavior"*[32]. Over the past few decades, there has been a shift in how people access and consume music. Advances in technologies have enabled listeners to move from physical storage devices (e.g., cassettes or CDs) to digital storage through computers or mobile devices. Currently, supported by online music streaming services and platforms (e.g., Apple, Spotify), music consumption is even more pervasive than ever before – people can access millions of songs almost anywhere at anytime through a variety of modalities[39].

However, the ability to access vast amounts of music can contribute to information overload for users. While an abundance of content is available for consumption, only a small amount is actually relevant to users and their desired listening goals. Although music streaming platforms provide strategies to guide and recommend users to discover relevant music through personalization[12, 31], search remains an important way for users to find music.

Prior research in general web search categorized search activities into two broad categories: lookup and exploratory[30]. Lookup searches have well-defined information needs in the searcher's mind, whereas exploratory searches usually involve an open-ended, persistent and multi-faceted information seeking problem[48]. In addition, researchers have observed that lookup searches can be different than exploratory searches in terms of users' search activities, behaviors and strategies[1]. Because the user's information needs and goals can differ on whether or not that search is exploratory, search systems that fulfill lookup searches well may not perform as well when users want to engage in exploratory search[48].

However, we have little knowledge on whether this holds true for searches within the music domain since information needs for music listeners are specific and can be different than general web searchers. The most common reason of seeking music was to "listen for entertainment[25]". In addition, people who consume music are driven by their emotional, social, and cognitive needs[16], which suggests that user goals in music search may be driven by broader needs than those described by [6] and [40]. In a recent qualitative study of music search on an online music streaming platform, researchers identified the existence of distinct music search **mindsets** – focused, open and exploratory[17]. Results suggested that

users searching with a focused mindset had a particular item in mind. In an open mindset, users had a seed of an idea in mind while searching. The exploratory mindset described users who employed search to learn more deeply about music.

In this mixed-method study, our goal is to characterize and distinguish between focused mindsets and non-focused mindsets to validate the findings from prior qualitative research[11] and identify associated behavior patterns at a large scale. We choose to collapse mindsets into two categories as users could distinguish actions and behaviors for focused mindset searches clearly [11]. First, we design a pop-up survey to collect users' mindset in real-time when they approach a music search. To further observe how mindset influences search behavior, we analyzed the surveyed users' behavior as they continued to use the platform. Specifically, our study answers the following three research questions:

RQ1: How does music search mindset vary among users?
RQ2: How does users' search behavior differ between mindsets?
RQ3: Can we infer users' mindset based on their behavioral signals when they interact with the music streaming platform?

Our research extends the previous literature in the general search domain and provides a novel contribution by studying the music search mindset. We found that music search mindset can depend on users' music preference and influence user behavior and activities in variety of ways. These behavioral signals can then be used to infer the search mindset. Altogether, our study provides insights for music as well as other content discovery and consumption services to best support their users when they approach a search.

2 RELATED WORK

Categorization of general search activities. Research in general web search have put considerable effort into understanding and characterizing the needs and goals of users. Search activity has been categorized as navigational (e.g., finding the URL of a desired web page) , informational (e.g., learning something from a web search) and transactional (e.g., performing a desired activity) [6, 40]. Beyond user goals, search behavior is influenced by factors such as the perceived task complexity and knowledge of the user[2, 27–29]. Liu et al.[29] found that web search behaviors, such as task completion time, number of different search engines used, and queries issued are all affected by the complexity and the preciseness of the search goal. Aula et al.[2] found that when tasks become more difficult, users issued numerous search queries, viewed many results, and spent more time on search result pages. Because these goals are meant to characterize general web search, they are broader in nature and describe user motivations that may not be consistent with how music listeners approach search.

Examples of domain-specific search. Music search is a type of a domain-specific search where user goals and needs may differ from general web search. Bainbridge et al. [3] found that users express their music needs in variety ways, ranging from bibliographic queries, genre and lyrics. Laplante and Downie [24] distinguish between hedonic and utilitarian outcomes in music information seeking. For utilitarian outcomes, users are looking for something to listen to or hoping to gain information about music.

Another example of domain-specific search is product search. Although product search has similar search categories as web search (browsing vs. directed search)[41], the shoppers' decision-making process is different[26]. Recent studies focus on using behavioral data to improve the search experience to better align with shoppers' needs. Wu et al.[49] used implicit feedback to identify two stages of searching: comparing search result pages and deciding to purchase on description pages. Researchers from Pinterest have revealed that goal specificity (e.g., visiting a web site for a specific goal) and temporal range (e.g., when the user envisions the goal will be achieved) influence user intent[10]. Su et al. [47] determined three types of product search intents in a commercial product search engine that also range in specificity and focus.

Prior work in describing information needs in product search suggests that there is a value in differentiating between general web search and more targeted domain-specific search, such as music. In this paper, we describe the particular information seeking behavior of music listeners when they either have something specific they want to consume, or they are more open to suggestion.

The role of music in society. Scholars from various disciplines have long studied the origin and purposes of music and why people listen to music. In general, scholars agree on the fundamental function that music provides for humans – to produce pleasure[46] and pass the time[18]. One line of research focuses on the broad impact of music on culture and society, referring music as a medium with multiple social and cultural benefits[4, 18, 32, 33, 36]. Other studies focus on the ways in which people use music in their everyday lives [43–45].

User characteristics influencing music consumption. Research has suggested that the behavior of people interacting with music can be shaped by listener's individual differences such as age[9, 34], gender[5, 9, 23, 34], culture [37], personality traits[8, 9, 13], and musical preferences [15, 42–44]. Studies also demonstrate that music listening is context-dependent and influenced by the situations (e.g., where and when). Researchers have observed that music listening predominantly occurs at home, while driving, or while using public transport [14, 22, 35]. Krause et al. [22] found that the intensity of the consequences of music listening varies across listening locations. The time of day or the day of the week when music listening occurs is also important – music is more likely to help people concentrate during the workday (8:00 a.m. – 4:59 p.m.) than during the evening (5:00 – 11:00 p.m.)[35, 38].

In sum, as music serves different functions for society and individuals, our study intends to extend prior work on general web search to examine search in the music domain.

3 RESEARCH PLATFORM

We conduct our study on Spotify, a widely used music streaming platform available on mobile devices. Search is a prominent feature on this platform and it requires users to type into a search bar located at the top of the screen. Currently, the feature relies on an instant search system, which updates the search engine results page (SERP) with each keystroke. The SERP itself features a single top result displayed prominently at the top. For some artist or genre searches, the SERP features a carousel of relevant playlists that users can scroll through horizontally underneath the top result. Other relevant results follow, clustered according to entity type in the following order: songs, artists, podcasts, albums, playlists, podcast

Table 1: Feature definitions

User features		
Demographics	Gender/Age	User's self report gender (female or male) and age
Engagement	Account age	The total days as a user of the platform
	Days active	The total days active
Music Preference	Music discovery rate	The percentage of novel tracks listened to by the user in the past month.
Session features		
Session context	Time of the day	Day parts include: early morning, morning, noon, afternoon, evening, night
	Weekend	Weekdays or weekends (Saturday and Sunday)
Session length	Session duration	Time duration from start until the end of the focal session (in seconds)
Session switching activities	Number of switching activities	Number of times the user switches between behavior types (e.g., keystrokes, clicks, streams, etc.)
Activity features		
Query	Query edit distance	Max edit distance [20] of the queries issued in the focal session.
	Final query length	Number of characters of the final query issued in the focal user session.
	Query keystroke shape	Query shapes (D, B, L, and Γ patterns) [7] for all search queries issued in the focal user session.
Searching	Search duration	Time spent engaging in search activity (in seconds)
	Number of searches	Number of searches conducted within the focal session
	Number of addition/deletion	Number of characters typed/deleted
Clicking and Streaming	Time to first click	Time (in seconds) spent from the start of the session to the first click
	Rank of the first/last click	Rank position of the first/last click
	Entity type of the first/last click	Entity type: playlist, track, album, artist, and others
	Music stream	Whether the current user session includes a music stream longer than 30 seconds

episodes, and user profiles. In addition, users are able to click on a context menu for each song result, where they can take actions such as saving the song, adding to playlist, sharing, or viewing the album/artist.

4 SURVEY DESIGN AND DATA COLLECTION

4.1 Survey Design and Usability Test

Our survey design is guided by usability tests. We ran the tests with 10 users (4 internal employees and 6 external participants) to ensure that we understood a) whether our survey was able to capture respondents' mindset with fidelity; b) how respondents would interpret and answer the questions; c) how the survey triggering process could possibly impact users' subsequent experience. To qualify for the usability study, participants needed to be Android mobile phone users and had to have searched on Spotify at least 3 times in the week before our study. Participants ranged in age from 21 to 39, and included six females and four males. External participants in the usability study received a $50 gift card.

We conducted face-to-face, 30 minute usability tests combined with semi-structured interviews with each of the participants. The entire study included three phases: warm-up, search tasks, and post-interview. We began the study with a warm-up phase by discussing participants' music taste and habits in general and then narrowed the conversation to focus on search in the platform. In the search tasks phase, participants were asked to recall their most recent searches under three different scenarios: a general search (any recent search), a non-focused mindset search (open-ended music needs), and a focused mindset search (specific music needs). For each scenario, we asked participants to provide details and walk us through their search process using an Android mobile phone that we provided. The survey was triggered when participants navigated to the search page. During each task, we observed participants' reaction and recorded their responses. We followed up by asking the questions "What do the survey question/response options mean to you?" and "How did you decide your answer?' After completing the tests, we concluded the study with a brief interview to understand the participants' overall experience with the survey and the triggering process. We wanted to ensure that surveying the user's mindset was reasonable, and the process was not overly disruptive nor changed user's subsequent experience.

We designed the survey to have only one multiple-choice question because we wanted to minimize the interruption of the pop-up survey and allow users to continue their desired behavior as quickly as possible. The survey question and the response options intended to capture the current mindset of users while searching. We tested different versions of the survey question and response options, and we updated the design based on the results of the usability test.

After several rounds of feedback, the final survey design included the question: "Do you know what exactly you want to find?" Response options included *"Yes. I am looking for one thing"* to capture the users' **focused** mindset, and *"No. I am open to suggestions"* to capture the users' **non-focused** mindset. Users could also opt-out from the survey at anytime by clicking the "DISMISS" button. Based on our usability tests, this version best captured the users' search mindset. For this final version, study participants reported that the survey question and selections are "clear and answerable." In addition, they reported that the final version would not change their subsequent experiences in using the platform. In terms of the triggering process, users reported that the "it would be annoying if this happens on my phone every time...one time would be OK though." We therefore decided that the survey would only be triggered once in a user's lifetime on the platform.

We sent out the surveys in three waves to 50,000 Android users in the United States during the month of July 2018. To qualify for the survey, users had to have searched at least 3 times on Spotify in the week before we sent out the survey. We chose this cutoff to make sure that users had encountered the search feature prior to receiving the survey. To minimize the coverage bias, we used random sampling and controlled for the product type – free users and subscription users were equally likely to receive the survey.

4.2 Survey Responses and Observational Data

User Session. The second component of our methodology captured users' behavioral data when they interact with the search platform. As users interact with the search platform, the system records a constant stream of search event logs. User sessions are then segmented: each user session begins with a keystroke where the user types something in the search bar, and it ends if the user is idle for more than 10 minutes[19]. We summarize each user session using three general sets of features to describe: the users who conduct a search, the search session and its context, and the user activities in the session. Informed by the prior research in Section 2, we construct the three categories of features for each user session: user features, session features and activity features (see Table 1).

Survey Responses and User Sessions. The survey remained live for two weeks for data collection during the month of July 2018. In the end, 16% of the surveyed users triggered the survey, and 27% of them responded the survey. In total, 2,234 users responded to the survey and generated 27,504 user sessions within the two weeks.

To understand how the users' behavior is associated with different search mindsets, we focus on the search activities in the *first user session* immediately following the survey response time. As some users responded to the survey without searching for anything within 10 minutes, we excluded those user responses. In total, we had 1,779 survey responses that we were able to associate with a user session. We refer this set of session data for 1,779 unique users as the *labeled dataset*. The median time interval from the survey response time to the start of the first search activity is 3.6 seconds. We refer to the rest of 25,725 user sessions without a mindset label as the *unlabeled dataset*. User search sessions have a median time duration of 23.7 seconds and the median number of searches is two.

5 ANALYSES AND RESULTS

In this section, we address our three research questions. We used the *labeled dataset* to discover the patterns associated with the different music search mindsets and then utilized the *unlabeled dataset* to validate those patterns at scale. To answer RQ3, we built a supervised machine learning model based on the features in Section 4.2.

RQ1: Music searches are more likely to be focused, and the search mindsets depend on user's personal music preference.

From the survey responses, we observed that music search mindsets vary across users. Our data indicated that users are more likely to have a focused mindset when they approach a search: 65% of the survey responses indicated focused searches. In addition, we performed a regression analysis to assess the main effects of the user features described in Table 1 to their search mindsets. The regression analysis was conducted with each user session as the unit of analysis with the search mindset as a binary dependent variable and user features as independent variables. Overall, there is a significant effect of user's *Music Discovery Rate* (Wald χ^2=4.039, df=1. Sig.<.05). Users who discovered more novel music tracks within the past month are associated with an increased likelihood (β = 0.635) to search with a non-focused mindset. We found no significant effects of user gender, account age, and days active on the search mindset.

RQ2: Users' searching, clicking and streaming behavior differ between focused vs. non-focused mindsets.

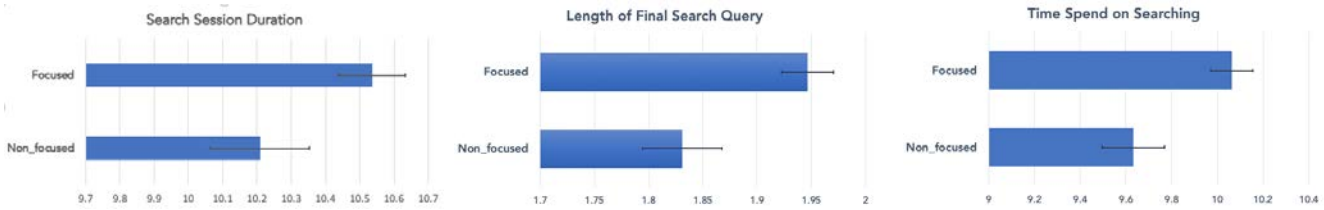
Sessions. Our second analysis focused on assessing each mindset's relationship to the search session features in terms of the session context, the duration and the number of switching activities. We conducted a regression analysis with the search mindset as a binary dependent variable and session features as independent variables. The analysis was conducted with each user session as the unit of analysis nested within the user gender, to account for the non-independence of sessions among users with the same gender. We include other user features (i.e. user age, account age, active, discovery rate) in the model as control variables. The results demonstrate a significant association between the session duration and mindset (Wald χ^2=4.049, df=1. Sig.<.05): search sessions with the focused mindset tend to last longer (β = 0.037) than the search sessions with the non-focused mindset (Figure 1a). There exists no significant effects of the search context on the mindsets: user search mindset does not vary by time of day or day of week.

Activities. To evaluate if user activities differ between search mindsets, we conducted three regression analyses to assess the relationships between search mindset and queries, search behavior, and clicking and streaming activities. These analyses were conducted separately to avoid the effect of multicollinearity. All the regression analyses modeled search mindset as a binary dependent variable, and each analysis included the query features, searching features, and clicking/streaming features separately as independent variables. The analyses have each user session as the unit of analysis nested within the user gender and session context, to account for the non-independence of sessions among users with the same gender and searching at the same time. We included other user and session features as control variables.

Query: We found that there are significant associations between search mindset and querying behavior. Features including query length (Wald χ^2=3.885, df=1. Sig.<.05) and keystroke patterns (Wald χ^2= 4.850, df=3. Sig.<.05) have significant effects. Specifically, users search with focused mindset tend to issue longer queries using more characters (β = 0.151) than the queries issued by non-focused mindset search (Figure 1b). In addition, users searching with focused mindset are less likely to issue D-shape[7] queries (β = -0.132) compared to non-focused mindset.

Searching: Users' search activities in terms of the search duration (Wald χ^2= 9.507, df=1. Sig.<.01) and the total number of characters the user typed (Wald χ^2= 6.888, df=1. Sig.<.01) significantly differ between search mindsets. Specifically, users who search with a focused mindset tend to spend a longer time while searching (β = 0.171) and have more total number of characters added (β = 0.148) than those users with a non-focused mindset. Figure 1c illustrates the estimated mean of time spent for search under each mindset.

Clicking: Users' clicking behaviors, such as time to the first click (Wald χ^2= 4.677, df=1. Sig.<.05), rank of the first click (Wald χ^2= 11.842, df=1. Sig.<.001), entity type of the first click (Wald χ^2= 15.211, df=4. Sig.<.01) and last click (Wald χ^2= 16.649, df=4. Sig.<.001), differ significantly between search mindsets. Users searching with a focused mindset tend to spend more time before their first click (β = 0.023). The first clicked entity by users with a focused mindset tend to have larger rank positions (β = .141), which suggest that they tend to scroll down and click on a lower ranked entity. In



(a) Mean of session duration (in seconds) for focused searches vs. non-focused searches

(b) Mean of final search query length issued by focused vs. non-focused searches

(c) Mean of time (in seconds) spent on search for focused vs. non-focused searches

Figure 1: RQ2: Users' behavior differs between mindsets

terms of their clicked entity type, we observed that the non-focused mindset searchers are more likely to click on a playlist compared with other types of entities such as tracks, albums, or artists. A playlist can be generated by editors based on certain topics or popularity, by algorithms based on the personalized recommendation, or even by other users. Therefore, it may contain a more diverse set of musics favored by the users who search with a non-focused mindset.

Streaming: We found that focused mindset search sessions are less likely to involve a music streaming (Wald χ^2= 3.165, df=1. Sig.=.075, β = -0.045) directly after a search, but they are more likely to have clicks on the Context Menu button (Wald χ^2= 3.223, df=1. Sig.=.073, β = 0.093) – this button is one step before users saving search results to their personal music repository. However, the two observations are marginally significant with small effect sizes.

Unsupervised clustering validates the patterns at scale. So far, we have identified a set of behavior patterns using the labeled dataset. Next, we validate the patterns at scale based on the clustering analysis of the larger unlabeled dataset and interpreting each cluster. We ran K-means clustering, and used the "elbow test" to set the optimized number of clusters (k)[21].

We report the results of our clustering analysis in Figure 2. Figure 2 is a column-standardized heatmap of the cluster centers for six groups with respect to the features. We found that the clusters reflect the behavior patterns described in Section 5, which correspond to focused and non-focused search sessions. Specifically, the user sessions from cluster 6 have a longer search duration and time to first click. Users from this cluster had more keystrokes, issued long queries, scrolled down and clicked on the lower-ranked entities. They used the context-menu button frequently, but engaged less with music streaming and playlist clicks. We interpret this cluster as the "**focused mindset**." In contrast, the user sessions from cluster 5 had shorter search duration and time to first click. Users from these search sessions used fewer keystrokes and issued short queries, but they clicked on playlists more frequently. This cluster includes user sessions whose behavior patterns correspond closely to the patterns identified for the **non-focused mindset**.

Interestingly, the clustering results suggest that the search mindset may be related to user goals, which is consistent with the findings from the prior work about user goals[17]. User sessions from clusters 2 and 4 spent more time searching, querying and music

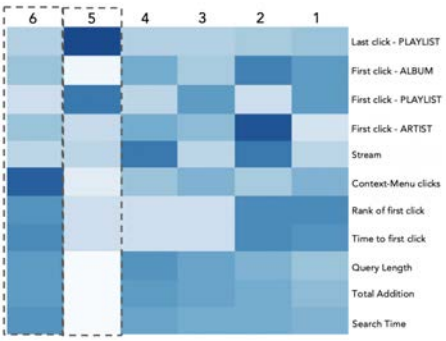


Figure 2: Clustering heatmap: 5 & 6 represent the non-focused and focused mindset session cluster respectively

streaming, suggesting that user goals in these clusters may be listening. However, these two clusters presented behaviors (e.g. time to first click, rank of clicked entities, clicked entity type) associated with different mindsets. User sessions from cluster 2 demonstrate the behavior patterns corresponding to a focused mindset – users engaged a longer time before the first click, clicked on lower ranked entities, and clicked on artists and albums frequently, whereas user sessions from cluster 4 follow the behavior patterns that we observed for the non-focused mindset.

RQ3: Search mindset can be inferred from lower-level user behavioral signals.

So far, we have described how music search mindset is associated with different behavior patterns. However, is it possible to use these behavioral signals to infer users' search mindset? Such information would be useful in adjusting the user interface and search results to better serve users based on their needs. In this section, we answer our third research question by building predictive models.

We utilized the features proposed in Section 4.2 and consider the following two prediction tasks: 1) how is the prediction power improved by adding behavior features chronologically throughout the session, and 2) after we have observed all the features within a session, can we predict whether the search mindset was focused or non-focused? As different features can be observed in chronological order during the user session, these two tasks can not only indicate how well we can infer the search mindset, but can also suggest how quickly we can infer search mindset. For example, user features and session context features are available before the start of a user

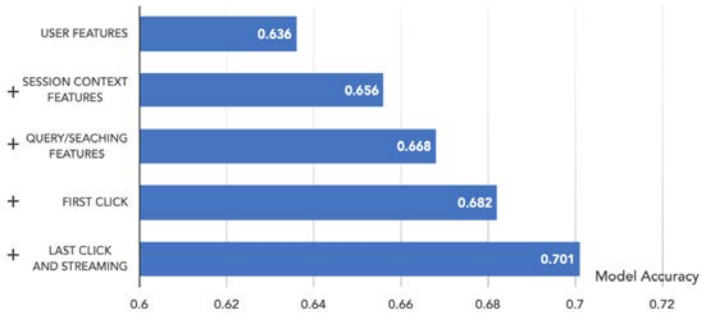


Figure 3: Model performance improved by adding features

session, but we can only obtain the clicking and streaming features at the end of the user session.

We implemented several supervised machine learning models to predict the search mindset for a user session using the labeled dataset as ground truth. As the dataset is unbalanced with majority user sessions having a focused mindset, a random guess would achieve classification accuracy of a 65% in predicting focused mindset. We trained our model by adding class weights to consider for the unbalanced data. We used logistic regression as a baseline model, and three separate models using bagging, boosting and random forest ensemble methods on a decision tree classifier to improve the model performance. We split the dataset to have 70% training data and 30% test data. We conducted a three-fold cross validation on the training data to find the optimized parameters for the model and applied the trained model into the test dataset. We report our model performance using accuracy, precision, recall, and F1 score.

Overall, our results demonstrate the potential to infer music search mindset based on our proposed user session features. Figure 3 demonstrates the improvement of the model predictive power by adding features based on the approximate chronological order that they can be observed throughout the user session. By including all the features, the model is able to predict the focused mindset with an accuracy rate of 70.1% (F1 = 0.806, Recall = 0.917, Precision = 0.718), a significant improvement of accuracy rate from both the random condition and the baseline model.But, when we only include the user and session context features in the model, the predictive power is quite low (65.6%) and the user's music discovery rate is still the most individually predictive feature. This result also indicates that it can be very challenging to predict search mindset at the session start. But the model's predictive power improves by including more user activities. Specifically, by adding features from search and query activities that occur in the early stage of a user session, the model performance improves to 66.8% accuracy. Eventually, by adding features extracted from users' clicking and streaming activities that occur in the late stage of a user session, the model performance achieves its best performance.

6 DISCUSSION AND CONCLUSION

Our results demonstrate that users' behaviors differ between search mindsets during search sessions. Specifically, users expend more effort while conducting **focused mindset** searches: they issue longer search queries, spend a longer time searching and a longer time to first click. In addition, they scroll further down the search result list and click on lower-ranked entities. They tend not to listen and stream these search results directly after the search, but are more willing to add the search results into their saved music library immediately after the search. Users searching with a **non-focused mindset** tend to put in less effort but rely more on the system for good search results and suggestions. For these users, their queries are shorter; they spend less time on searching and before their first click; but they are more willing to click on the very first entity the system returned, and those clicked entities are more likely to be a playlist. Finally, based on supervised machine learning techniques, we used these discovered behavioral patterns to recover a user's music search mindset in a user session.

These insights provide opportunities for content discovery and consumption services broadly (e.g., YouTube or Netflix) to develop strategies to better serve their users when they approach a search. Although it may be very challenging to infer search mindset at the very beginning of the search session, users' past music or content discovery preferences can provide a baseline estimation of what the search mindset for the current session is likely to be. As users' activities and behavior signals increase chronologically, the search system and interface can be adjusted at real-time to accommodate for the specific needs of distinct user mindsets. For focused mindset searches, the system should provide support that reduces users' searching and querying efforts, such as query auto-completion tools. To address non-focused mindset searches, the system can offer additional content recommendations, as these users rely more on the system for search results and music suggestions. For example, music streaming services can prioritize showing the playlists for non-focused mindset searches while showing the specific and targeted content (e.g., music track or album) for focused mindset searches.

Several limitations exist in the current study. First, the survey triggered only once per user's lifetime, so our labeled dataset includes only one session per user. As users' previous search sessions can also impact their current search session, we can potentially include that information when inferring searching mindset. Second, mindset may change over time. We partially mitigate this issue: we survey user mindset immediately before the beginning of the session and consider only the first user session, which tends to be relatively short with a median time duration of 23.7 seconds. Future studies can investigate whether and how the user's search mindset would change over time. Third, while we sought to present a broad overview of search behaviors and activities associated with two types of mindsets, our results from the clustering analysis suggest that user goals in searching music can also play a role. Future work can more deeply investigate the role of user goals as they differ across types of mindsets. Finally, as our data consists of Android users on one specific music streaming platform within one country over two weeks, future work can reproduce the current study by collecting more data with from a broader set of users across different content streaming systems for a longer time period to validate generalizability across platforms.

REFERENCES

[1] Kumaripaba Athukorala, Dorota Głowacka, Giulio Jacucci, Antti Oulasvirta, and Jilles Vreeken. 2016. Is exploratory search different? A comparison of information

search behavior for exploratory and lookup tasks. *Journal of the Association for Information Science and Technology* 67, 11 (2016), 2635–2651.
[2] Anne Aula, Rehan M Khan, and Zhiwei Guan. 2010. How does search behavior change as search becomes more difficult?. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 35–44.
[3] David Bainbridge, Sally Jo Cunningham, and J Stephen Downie. [n. d.]. How People Describe Their Music Information Needs: A Grounded Theory Analysis Of Music Queries. ([n. d.]), 2.
[4] Jeanette Bicknell. 2007. Explaining strong emotional responses to music. *Journal of Consciousness Studies* 14, 12 (2007), 5–23.
[5] Diana Boer, Ronald Fischer, Hasan Gürkan Tekman, Amina Abubakar, Jane Njenga, and Markus Zenger. 2012. Young people's topography of musical functions: Personal, social and cultural experiences with music across genders and six societies. *International Journal of Psychology* 47, 5 (2012), 355–369.
[6] Andrei Broder. 2002. A taxonomy of web search. 36, 2 (2002), 3. https://doi.org/10.1145/792550.792552
[7] Inci Cetindil, Jamshid Esmaelnezhad, Chen Li, and David Newman. 2012. Analysis of Instant Search Query Logs.. In *WebDB*. 7–12.
[8] Tomas Chamorro-Premuzic, Montserrat Gomà-i Freixanet, Adrian Furnham, and Anna Muro. 2009. Personality, self-estimated intelligence, and uses of music: A Spanish replication and extension using structural equation modeling. *Psychology of Aesthetics, Creativity, and the Arts* 3, 3 (2009), 149.
[9] Tomas Chamorro-Premuzic, Viren Swami, and Blanka Cermakova. 2012. Individual differences in music consumption are predicted by uses of music and age rather than emotional intelligence, neuroticism, extraversion or openness. *Psychology of Music* 40, 3 (2012), 285–300.
[10] Justin Cheng, Caroline Lo, and Jure Leskovec. 2017. Predicting Intent Using Activity Logs: How Goal Specificity and Temporal Range Affect User Behavior. In *Proceedings of the 26th International Conference on World Wide Web Companion (WWW '17 Companion)*. International World Wide Web Conferences Steering Committee, 593–601. https://doi.org/10.1145/3041021.3054198
[11] Blinded for review. 2019. Submission under review. In *Under review*. ACM, Currently Under Review.
[12] Jean Garcia-Gathright, Brian St. Thomas, Christine Hosey, Zahra Nazari, and Fernando Diaz. 2018. Understanding and Evaluating User Satisfaction with Music Discovery. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. ACM, 55–64. https://doi.org/10.1145/3209978.3210049
[13] J Ginsborg, A Lamont, M Phillips, and S Bramley. 2015. The use of music in everyday life and personality: A cross sectional study of a whole sample of school children in Germany. (2015).
[14] Alinka E Greasley and Alexandra Lamont. 2011. Exploring engagement with music in everyday life using experience sampling methodology. *Musicae Scientiae* 15, 1 (2011), 45–71.
[15] Fabian Greb, Wolff Schlotz, and Jochen Steffens. 2017. Personal and situational influences on the functions of music listening. *Psychology of Music* (2017), 0305735617724883.
[16] David J Hargreaves and Adrian C North. 1999. The functions of music in everyday life: Redefining the social in music psychology. *Psychology of music* 27, 1 (1999), 71–83.
[17] Christine Hosey, Lara Vujovic, Brian St. Thomas, Jean Garcia-Gathright, and Jennifer Thom. 2019. Just Give Me What I Want: How People Use and Evaluate Music Search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2019). ACM.
[18] David Huron. 2001. Is music an evolutionary adaptation? *Annals of the New York Academy of sciences* 930, 1 (2001), 43–61.
[19] Rosie Jones and Kristina Lisa Klinkner. 2008. Beyond the Session Timeout: Automatic Hierarchical Segmentation of Search Topics in Query Logs. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM '08)*. ACM, 699–708. https://doi.org/10.1145/1458082.1458176
[20] Dan Jurafsky and James H Martin. 2014. *Speech and language processing*. Vol. 3. Pearson London.
[21] Trupti M Kodinariya and Prashant R Makwana. 2013. Review on determining number of Cluster in K-Means Clustering. *International Journal* 1, 6 (2013), 90–95.
[22] Amanda E Krause, Adrian C North, and Lauren Y Hewitt. 2016. The role of location in everyday experiences of music. *Psychology of Popular Media Culture* 5, 3 (2016), 232.
[23] Emmanuel Kuntsche, Lydie Le Mével, and Irina Berson. 2016. Development of the four-dimensional Motives for Listening to Music Questionnaire (MLMQ) and associations with health and social issues among adolescents. *Psychology of Music* 44, 2 (2016), 219–233.
[24] Audrey Laplante and J. Stephen Downie. [n. d.]. The utilitarian and hedonic outcomes of music information-seeking in everyday life. 33, 3 ([n. d.]), 202–210. https://doi.org/10.1016/j.lisr.2010.11.002
[25] Jin Ha Lee, Hyerim Cho, and Yea-Seul Kim. 2016. Users' Music Information Needs and Behaviors: Design Implications for Music Information Retrieval Systems. 67, 6 (2016), 1301–1330. https://doi.org/10.1002/asi.23471
[26] Beibei Li, Anindya Ghose, and Panagiotis G. Ipeirotis. 2011. Towards a Theory Model for Product Search. In *Proceedings of the 20th International Conference on World Wide Web (WWW '11)*. ACM, 327–336. https://doi.org/10.1145/1963405.1963453
[27] Yuelin Li and Nicholas J Belkin. 2008. A faceted approach to conceptualizing tasks in information seeking. *Information Processing & Management* 44, 6 (2008), 1822–1837.
[28] Yuelin Li and Nicholas J Belkin. 2010. An exploration of the relationships between work task and interactive information search behavior. *Journal of the American Society for Information Science and Technology* 61, 9 (2010), 1771–1789.
[29] Jingjing Liu, Michael J Cole, Chang Liu, Ralf Bierig, Jacek Gwizdka, Nicholas J Belkin, Jun Zhang, and Xiangmin Zhang. 2010. Search behaviors in different task types. In *Proceedings of the 10th annual joint conference on Digital libraries*. ACM, 69–78.
[30] Gary Marchionini. 2006. Exploratory search: from finding to understanding. *Commun. ACM* 49, 4 (2006), 41–46.
[31] James McInerney, Benjamin Lacker, Samantha Hansen, Karl Higley, Hugues Bouchard, Alois Gruson, and Rishabh Mehrotra. 2018. Explore, Exploit, and Explain: Personalizing Explainable Recommendations with Bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys '18)*. ACM, 31–39. https://doi.org/10.1145/3240323.3240354
[32] A.P. Merriam. 1964. The Anthropology of Music. *Northwestern University Press, Chicago* (1964).
[33] Steven Mithen, Iain Morley, Alison Wray, Maggie Tallerman, and Clive Gamble. 2006. The Singing Neanderthals: the Origins of Music, Language, Mind and Body, by Steven Mithen. London: Weidenfeld & Nicholson, 2005. ISBN 0-297-64317-7 hardback£ 20 & US $25.2; ix+ 374 pp. *Cambridge Archaeological Journal* 16, 1 (2006), 97–112.
[34] Adrian C. North. 2010. Individual differences in musical taste. *American journal of psychology* 123, 2 (2010), 199–208.
[35] Adrian C North, David J Hargreaves, and Jon J Hargreaves. 2004. Uses of music in everyday life. *Music Perception: An Interdisciplinary Journal* 22, 1 (2004), 41–77.
[36] Jaak Panksepp and Günther Bernatzky. 2002. Emotional sounds and the brain: the neuro-affective foundations of musical appreciation. *Behavioural processes* 60, 2 (2002), 133–155.
[37] Minsu Park, Jennifer Thom, Sarah Mennicken, Henriette Cramer, and Michael Macy. [n. d.]. Global music streaming data reveal diurnal and seasonal patterns of affective preference. ([n. d.]), 1. https://doi.org/10.1038/s41562-018-0508-z
[38] Shabbir A Rana and Adrian C North. 2007. The role of music in everyday life among Pakistanis. *Music Perception: An Interdisciplinary Journal* 25, 1 (2007), 59–73.
[39] Edison Research. 2018. The Infinite Dial 2018. https://www.edisonresearch.com/infinite-dial-2018/
[40] Daniel E Rose and Danny Levinson. 2004. Understanding user goals in web search. In *Proceedings of the 13th international conference on World Wide Web*. ACM, 13–19.
[41] Jennifer Rowley. 2000. Product search in e-shopping: a review and research propositions. 17, 1 (2000), 20–35. https://doi.org/10.1108/07363760010309528
[42] Thomas Schäfer. 2016. The goals and effects of music listening and their relationship to the strength of music preference. *PloS one* 11, 3 (2016), e0151634.
[43] Thomas Schäfer and Peter Sedlmeier. 2009. From the functions of music to music preference. *Psychology of Music* 37, 3 (2009), 279–300.
[44] Thomas Schäfer and Peter Sedlmeier. 2010. What makes us like music? Determinants of music preference. *Psychology of Aesthetics, Creativity, and the Arts* 4, 4 (2010), 223.
[45] Thomas Schäfer, Peter Sedlmeier, Christine Städtler, and David Huron. 2013. The psychological functions of music listening. *Frontiers in psychology* 4 (2013), 511.
[46] Emery Schubert. 2009. The fundamental function of music. *Musicae Scientiae* 13, 2_suppl (2009), 63–81.
[47] Ning Su, Jiyin He, Yiqun Liu, Min Zhang, and Shaoping Ma. 2018. User Intent, Behaviour, and Perceived Satisfaction in Product Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, 547–555. https://doi.org/10.1145/3159652.3159714
[48] Ryen W White and Resa A Roth. 2009. Exploratory search: Beyond the query-response paradigm. *Synthesis lectures on information concepts, retrieval, and services* 1, 1 (2009), 1–98.
[49] Liang Wu, Diane Hu, Liangjie Hong, and Huan Liu. 2018. Turning Clicks into Purchases: Revenue Optimization for Product Search in E-Commerce. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. ACM, 365–374. https://doi.org/10.1145/3209978.3209993

Click Feedback-Aware Query Recommendation Using Adversarial Examples

Ruirui Li
UCLA
rrli@cs.ucla.edu

Liangda Li
Yahoo Research
liangda@yahoo-inc.com

Xian Wu
University of Notre Dame
xwu9@nd.edu

Yunhong Zhou
Yahoo Research
yunhongz@verizonmedia.com

Wei Wang
UCLA
weiwang@cs.ucla.edu

ABSTRACT

Search engine users always endeavor to formulate proper search queries during online search. To help users accurately express their information need during search, search engines are equipped with query suggestions to refine users' follow-up search queries. The success of a query suggestion system counts on whether we can understand and model user search intent accurately. In this work, we propose Click Feedback-Aware Network (*CFAN*) to provide feedback-aware query suggestions. In addition to modeling sequential search queries issued by a user, *CFAN* also considers user clicks on previous suggested queries as the user feedback. These clicked suggestions, together with the issued search query sequence, jointly capture the underlying search intent of users. In addition, we explicitly focus on improving the robustness of the query suggestion system through adversarial training. Adversarial examples are introduced into the training of the query suggestion system, which not only improves the robustness of system to nuisance perturbations, but also enhances the generalization performance for original training data. Extensive experiments are conducted on a recent real search engine log. The experimental results demonstrate that the proposed method, *CFAN*, outperforms competitive baseline methods across various situations on the task of query suggestion.

CCS CONCEPTS

• **Information systems** → **Query suggestion**; **Learning to rank**.

KEYWORDS

Query recommendation; context-aware; adversarial example

ACM Reference Format:
Ruirui Li, Liangda Li, Xian Wu, Yunhong Zhou, and Wei Wang. 2019. Click Feedback-Aware Query Recommendation Using Adversarial Examples. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313412

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313412

1 INTRODUCTION

The effectiveness of keyword-based search engines depends largely on the ability of a user to formulate expressive search queries. Expressive search queries clearly and unambiguously describe users' search intents and bring users directly to the desired information. In practice, translating human thoughts into concise sets of keywords to form queries is never straightforward [2]. This is particularly true for search engine users, who are mostly untrained casual users. In many cases, casual users have very limited background knowledge about the information they are searching for. To assist users in formulating queries, modern search engines are equipped with query suggestions [5, 16, 17, 20]. Given a user query, a query suggestion system deduces the search intent of the user by recommending a set of queries that are more expressive than the original user input. Our goal is to investigate the key concerns of existing query suggestion systems and to propose a novel approach to improve the suggestion performance.

Web search queries are usually very short, typically with only one or two keywords each [13, 24]. Short queries lead to the ambiguity issue. The ambiguity weakens the expressiveness of queries because the search engine may not understand the users' hidden search intents. This results in poor rankings of retrieval results and jeopardizes user experiences consequently.

A successful query suggestion system depends on understanding and modeling user search intents accurately. The user search intents lie in interactions, i.e., searches and clicks, between users and search engines. These interactions are generally partitioned into groups to form search sessions based on the time when they happened. Each search session is driven by consecutive related search queries and clicks of search engine users. In a search session, a user refines his/her original query and submits a sequence of follow-up queries to pinpoint his/her information need. Between two issued queries, the user may also click suggested queries to explore. The issued queries, together with the clicks, jointly allow us to accurately understand users' hidden search intents and come up with appropriate queries as suggestions. Unfortunately, most existing works merely consider clicked suggestions as feedback when modeling user search intents.

To conduct feedback-aware query suggestions, we mine search logs. A search log contains historical records, each of which registers the details of a web search conducted by a user. Table 1 shows some sample records extracted from the search log of a real search engine. Each record includes a query string, an anonymous user ID by whom the query was formulated, query submission time, the

Table 1: Sample search log records

UserID	Query String	Time	Clicked Suggestion	Suggested Queries
A	apple	2018-07-01 09:10:01	apple stock	apple store, **apple stock**, apple.com, apple iphone xs max
				apple id, itunes, icloud, google maps
A	apple price	2018-07-01 09:11:03	apple price per share	apple price targets, iphone apple price, **apple price per share**, apple
				apple fruit price, apple stock price, does apple price match, apple price today
B	disney	2018-07-03 12:12:02	disney cartoons	walt disney world, disney world tickets, disney world, disney experience
				disney orlando, **disney cartoons**, disney store, disney junior
B	disney shows	2018-07-03 12:22:20	disney channel cartoons	disney shows list, disney xd shows, disney tv shows, popular disney shows
				disney channel cartoons, list of disney tv series, 2018 disney channel shows

query suggestion subsequently clicked by the user (if done), and the suggested queries shown on the search page with the clicked ones highlighted in bold. The sample records show two search sessions. In the first session, user A first submitted query "apple" to the search engine, and then clicked the second query suggestion "apple stock" in the suggestion list. After that, the user issued the second query "apple price", and clicked the suggestion "apple price per share". The second search session involved a different user, B. S/he started searching with the query "disney", and then clicked "disney cartoons" in the suggestion list. He/she continued the search with the query "disney shows" and clicked "disney channel cartoons" subsequently. From the first search session, we can see that the clicked suggestion "apple stock", as feedback of the user, compensates for the ambiguity of the second issued query "apple price". It provides us with extra information to infer the underlying search intent of the user, which is to know the stock price of Apple instead of the price of Apple electronic products, such Apple iPhone. The clicked suggestion "disney cartoons" in the second search session also implicitly indicates that the user was interested in disney cartoons shows when the user search for "disney shows". These observations motivate us to include user feedback when modeling the search intents of search engine users.

Various research works have been carried out to tackle the task of query suggestion. Among them, neural network based models have made impressive progress over the past few years [7, 20, 25]. Deep neural network models mimic the learning process of human brains. Training these neural network models generally requires a large amount of training data to achieve excellent performance. However, for search related tasks, search queries involved are very sparse, and generally follow a long-tail distribution. Thus, those tail queries, with low probabilities in the data distribution, lack abundant training supports. In addition, the suggested queries are highly relevant to each other in most cases because they are all related to the search query in semantics. Distinguishing suggestions to be clicked from the ones that won't be clicked and ranking them take extreme efforts. Therefore, there tends to be a great performance gap between training and test. The trained model can be very sensitive to unseen perturbed queries. This is because queries are very short and a subtle perturbation on a query can lead to different or even the opposite search intent. In general, human perception and cognition are robust to a vast range of nuisance perturbations. However, neural networks are currently far from achieving the same level of tolerance of such perturbations [21]. This motivates us to investigate the robustness property of modern query suggestion systems and propose an appropriate method to achieve high tolerance.

Before we proceed to describe the details of the proposed approach, we enumerate the key properties that a good query suggestion system should observe:

- **[Context-Awareness]** The recommender should be aware of the sequential search queries issued by the user. These sequential queries implicitly inform the search intent of the user.
- **[Feedback-Awareness]** The recommender should also be aware of user clicks. User clicks, as an additional information source, can potentially reduce query ambiguities and allow us to understand user search intent comprehensively and more accurately.
- **[Robustness]** The recommender should increase the generalization of the original training data and improve the resistance to nuisance perturbations to the extent as extreme as possible. This reduces the performance gap between training and test, and provides favorable rankings of suggestions robustly.

2 RELATED WORK

Deep neural networks have been applied to many prediction and recommendation tasks successfully [11, 26, 27]. It also demonstrated excellent performances in query suggestion tasks [7, 15, 20, 25]. HRED [20], as the first study, employs a hierarchical encoder-decoder model to achieve context-aware query suggestion. [7] introduces the copy mechanism in addition to employing the encoder-decoder framework to generate suggested queries. [25] employs a feedback memory network to model titles of user clicked URLs. Reformulation inference network (RIN), proposed in [15], explicitly incorporates query reformulations during training.

Our work differs from the above neural network based works in that previously user clicked suggestions are explicitly incorporated as feedback. In addition, adversarial examples [6, 8] are dynamically constructed during the training of *CFAN*, which not only significantly improve the robustness of the model but also dramatically enhance the generic ranking performance.

3 PROBLEM STATEMENT

In this section, we formally define the objective of this paper. A search sequence S, represented as a sequence of search queries $<Q_1^S, Q_2^S, ..., Q_{M_1}^S>$, is submitted successively by a single user within a time interval. Each query in the search query is associated with corresponding click-through information (if happens), which is a set of clicked suggestions for that query. A clicked suggestion sequence C, $<Q_1^C, Q_2^C, ..., Q_{M_2}^C>$, is formed by ordering the clicked suggestions increasingly by the time the clicks happens. The goal of

this paper is: Given a set of candidate queries $\boldsymbol{Q}_{\text{can}}$, we would like to generate a ranking of candidates, with each candidate $Q_{\text{can}} \in \boldsymbol{Q}_{\text{can}}$, so that the ones to be clicked rank as high as possible.

4 METHODOLOGY

In this section, we discuss how to make query suggestions based on the search and click interactions between users and search engines. To achieve effective and efficient query suggestions, we decompose the task into two components. First, based on the search query, we generate a set of relevant queries as suggestion candidates. These suggestion candidates are informative to cover different interpretations and aspects of the search query. Second, we rank the suggestion candidates based user's entire search query sequence and clicked feedback. Only the top ones are selected and shown to the user as query suggestions.

4.1 Candidate Generation

Given a search query Q, the candidate generation component is responsible for constructing an informative set of expressive queries $\boldsymbol{Q}_{\text{can}}$ as suggestion candidates. An expressive suggestion candidate clearly and unambiguously describes the search intent of the user. An informative set of suggestion candidates are queries, that are diversified to cover different interpretations and aspects of the search query.

To generate such diversified informative and expressive suggestion candidates, multiple sources can be incorporated, such as search sessions and contextual information in search logs, related keywords from trending news, etc. In this work, we incorporate two sources based on the search log. The first source generates suggestion candidates based on query dependencies in search sessions. Given a search query, the follow-up search queries in the same search session tend to be more expressive and informative. Therefore they are adopted as suggestion candidates. Another source for candidate generation is based on prefix matching between the search query and historical search queries in the search log.

After generating suggestion candidates, the candidates are further ranked based on a combination of statistical features, i.e., the total number of submissions within last week, last month, and last year. Only top suggestion candidates are kept for further ranking in the candidate ranking component (see Section 4.2), while others are filtered out. The purpose of filtering out these less frequent suggestion candidates is to not only generate popular and fresh candidates, but also guarantee that recommendation services respond in real time. In a nutshell, given an input query Q, the candidate generation component returns a set of relevant queries as suggestion candidates.

4.2 Candidate Ranking

In this section, we discuss how we rank the suggestion candidates by mining the historical interactions between users and search engines. To better explain the ranking process, we first discuss the construction of training instances. Then, we go into details on how to utilize these instances to train *CFAN*.

Training instances are constructed from the historical search logs. Each instance contains a sequence of user issued queries S, a sequence of user clicked queries C, and a candidate suggested query Q_{can}. Two types of instances are constructed, i.e., positive instances and negative instances, denoted as Ins^+ and Ins^-, respectively. A positive instance is constructed if S and C jointly lead to the click of Q subsequently, while a negative instance is constructed if a query $\tilde{Q}$ is not clicked after S and C. In particular, to generate strong negative instances, only query $\tilde{Q}$, shown together with Q after searching the last query in S, is considered as a candidate query in the construction of negative instances. Positive instances Ins^+ and negative instances Ins^- jointly define the overall matching and ranking performance among a search query sequence S, a clicked suggestion sequence C, and a set of suggestion candidates $\boldsymbol{Q}_{\text{can}}$. We further define such a set of Ins^+ and Ins^- as a training view, denoted as V.

The clicks in the constructed training views allow us to tap into the wisdom of the crowds and provide a more favorable ranking of suggestions when running into the same or similar search and click scenarios. Given such scenarios, we expect the previously clicked suggestions rank ahead of those unclicked ones. To achieve this goal, Equation 1 shows the ranking loss function in *CFAN* over one training view V.

$$\text{Loss}_{\text{ranking}}(\Theta) = -\log\left(1 + \frac{\sum\limits_{\mathbf{ins}^+ \in V} g_\Theta(\mathbf{ins}^+)}{\sum\limits_{\mathbf{ins}^+ \in V} g_\Theta(\mathbf{ins}^+) + \sum\limits_{\mathbf{ins}^- \in V} g_\Theta(\mathbf{ins}^-)}\right), \tag{1}$$

where Θ is a set of parameters defining the ranking model, and $g_\Theta(\mathbf{ins})$ is a scoring function, which gives the ranking score of an instance. We use $\mathbf{ins}/\mathbf{ins}^+/\mathbf{ins}^-$ to represent the embedding of a general/positive/negative instance, respectively. By minimizing Equation 1 over all training views, $g_\Theta(\mathbf{ins}^+)$ is trained to have a high ranking score for clicked suggestions, while for unclicked suggestions, $g_\Theta(\mathbf{ins}^-)$ is trained to have a low score.

4.2.1 Search Intent Encoder. As we mentioned in the introduction, a user's search queries and clicked queries jointly help pinpoint his/her information need. A good search intent encoder is expected to incorporate both of them when modeling search intents. Note that both user search queries and clicked queries are inherently a sequence of queries; therefore, we first discuss how to model a single query through a query encoder. Then, we move on to the discussion of sequential query modeling through a query sequence encoder.

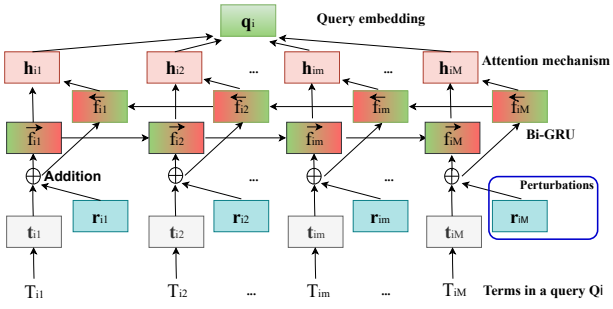


Figure 1: Query encoder with perturbed embeddings.

[Query Encoder.] Generally, queries contain different numbers of terms. In addition, different terms in a query contribute unequally

to the meaning of the query. For example, to search for the height of Barack Obama, users may formulate queries such as "how tall is Barack Obama", "height of Barack Obama", or even "Obama height". These three queries contain five, four, and two terms, respectively. Consider term importance in the first query, "tall", "Barack", and "Obama", these three terms, contribute more to understand the search intent of the query than the other terms, i.e., "how" and "is", which are generally less informative. In this work, we apply bidirectional GRU [18] and attention mechanism [22] to derive embeddings for a query Q.

Figure 1 shows the architecture of the query encoder. Given a query Q_i, composed of a sequence of terms $<T_{i1}, ..., T_{im}, ..., T_{iM}>$, where M is the number of terms in Q_i, we first embed each term T_{im} to an vector $\boldsymbol{t}_{im}$ through an embedding matrix as shown in Equation 2. To train robust models, adversarial examples are introduced by adding perturbed embeddings $\boldsymbol{r}$ into $\boldsymbol{t}$ (See more details in Section 4.2.3). We then use a bidirectional GRU to get the query embedding by summarizing information from both directions for terms in a query. The bidirectional GRU contains a forward GRU $\overrightarrow{f}$ which reads query Q_i from T_{i1} to T_{iM} and a backward GRU $\overleftarrow{f}$ which reads from T_{iM} to T_{i1}:

$$\boldsymbol{t}_{im} = Emb_t(T_{im}), m \in [1, M]. \tag{2}$$

$$\overrightarrow{\boldsymbol{f}}_{im} = \overrightarrow{GRU}(\boldsymbol{t}_{im}), m \in [1, M]. \tag{3}$$

$$\overleftarrow{\boldsymbol{f}}_{im} = \overleftarrow{GRU}(\boldsymbol{t}_{im}), m \in [M, 1]. \tag{4}$$

We obtain a representation $\boldsymbol{h}_{im}$ for each term T_{im} in query Q_i by concatenating the forward hidden state $\overrightarrow{\boldsymbol{h}}_{im}$ and backward hidden state $\overleftarrow{\boldsymbol{h}}_{im}$, i.e., $\boldsymbol{h}_{im} = [\overrightarrow{\boldsymbol{h}}_{im}; \overleftarrow{\boldsymbol{h}}_{im}]$, which summarizes the information of the whole query but centered around term T_{im}.

Since not all terms in a query contribute equally to the search intent embedded in the search query, attention mechanism is applied to extract such informative contributing terms and aggregate the embeddings of these terms to construct a query vector.

$$u_{im} = tanh(\boldsymbol{W}_w \boldsymbol{h}_{im} + b_w). \tag{5}$$

$$\alpha_{im} = \frac{\exp(u_{im}^T u_w)}{\sum_m \exp(u_{im}^T u_w)}. \tag{6}$$

$$\boldsymbol{q}_i = \sum_m \alpha_{im} \boldsymbol{h}_{im}. \tag{7}$$

Equations 5, 6, and 7 show the attention process on the query term level. The term embedding $\boldsymbol{h}_{im}$ is first fed into a one-layer MLP to get u_{im}. Then, u_{im} is further used to derive the normalized importance weight α_{im} through a softmax function. Finally, the query vector $\boldsymbol{q}_i$ is computed as a weighted sum of the term embeddings based on the weights.

[Query Sequence Encoder.] A query sequence is composed of a sequence of queries $<Q1, ...,Q_i, ..., Q_N>$, where N is the number of queries in the sequence. Both user search queries and clicked queries are essentially query sequences. Given a query sequence $<Q1, ...,Q_i, ..., Q_N>$, the query sequence encoder is expected to encode the hidden search intent in it. Similar to the structure of query encoder, the query sequence encoder contains a bidirectional GRU layer and an attention layer. The differences are two-fold. First, the query sequence encoder takes query embeddings as inputs. Second, there are no perturbations in query sequence encoder. Formally, a bidirectional GRU is first applied to encode the query sequence:

$$\overrightarrow{\boldsymbol{f}}_i = \overrightarrow{GRU}(\boldsymbol{q}_i), i \in [1, N], \tag{8}$$

$$\overrightarrow{\boldsymbol{f}}_i = \overleftarrow{GRU}(\boldsymbol{q}_i), i \in [N, 1], \tag{9}$$

The representation of a query Q_i in the sequence is constructed by concatenating $\overrightarrow{\boldsymbol{f}}_i$ and $\overrightarrow{\boldsymbol{f}}_i$, i.e., $\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i]$. $\boldsymbol{h}_i$ summarizes the neighbor queries around query Q_i but still focus on the meaning of query Q_i.

To reward queries that pinpoint the search intent of the user, query level attention mechanism is applied.

$$u_i = tanh(\boldsymbol{W}_s \boldsymbol{h}_i + b_s). \tag{10}$$

$$\alpha_i = \frac{\exp(u_i^T u_s)}{\sum_i \exp(u_i^T u_s)}. \tag{11}$$

$$\boldsymbol{s} = \sum_i \alpha_i \boldsymbol{h}_i. \tag{12}$$

Equations 10, 11, and 12 summarizes the attention process on query level. Finally, $\boldsymbol{s}$ gives the query sequence embedding that captures the search intent embedded in sequential queries $<Q1, ...,Q_i, ..., Q_N>$. In a nutshell, given a query sequence, we first use query encoder to derive the embedding for each query in the sequence. Then, we use query sequence encoder to further construct the embedding of the query sequence.

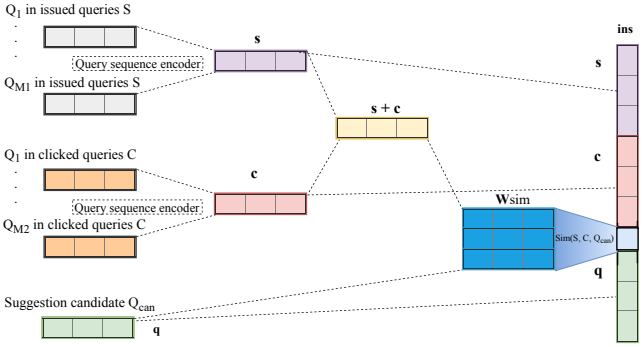


Figure 2: Training instance construction based on search sequence S, clicked sequence C, and suggestion candidate Q_{can}.

Given a training instance Ins, which is composed of a search query sequence S, a clicked query sequence C, and a suggestion candidate Q_{can}, we use the same query sequence encoder to derive the embeddings for S and C, denoted as $\boldsymbol{s}$ and $\boldsymbol{c}$, respectively, and we use a query encoder to encode the candidate Q_{can}, with its embedding denoted as $\boldsymbol{q}$. We use the concatenation of $\boldsymbol{s}$, $\boldsymbol{c}$, and $\boldsymbol{q}$, i.e., $[\boldsymbol{s}; \boldsymbol{c}; \boldsymbol{q}]$ to get the embedding **ins** to represent the instance Ins. Figure 2 summarizes the process.

4.2.2 Matching query sequences and suggestion query. Given a sequence of issued queries S, a sequence of clicked queries C, and a suggestion candidate Q_{can}, we aim to explicitly derive the matching among S, C, and Q_{can}. We follow approaches [3] and [19]. We define the matching score among $\boldsymbol{s}$, $\boldsymbol{c}$, and $\boldsymbol{q}$ as follows:

$$\text{Sim}(S, C, Q_{\text{can}}) = (\boldsymbol{s} + \boldsymbol{c})^T \boldsymbol{W}_{\text{sim}} \boldsymbol{q}, \tag{13}$$

where $\boldsymbol{W}_{\text{sim}} \in \mathbb{R}^{|\boldsymbol{s}|\times|\boldsymbol{q}|}$ is a similarity matrix. In this module, $\boldsymbol{s}+\boldsymbol{c}$ explicitly combines the search intent embedded in search queries S and clicked suggestions C. We seek a transformation of the candidate query Q_{can} that is the closest to $\boldsymbol{s}+\boldsymbol{c}$. The similarity matrix $\boldsymbol{W}_{\text{sim}}$ is a parameter of the network and is optimized during training. $\text{Sim}(S, C, Q_{\text{can}})$ is then explicitly appended to **ins** as a feature.

4.2.3 Learning and Optimization. When learning *CFAN* without adversarial examples, *CFAN* is optimized by learning only the ranker, formulated as finding a set of parameters Θ, that minimize an empirical ranking loss as shown in Equation 1 for a given set of training views. In adversarial training, we construct adversarial perturbations on training instances in order to cause a misbehavior of *CFAN* on a training view V. The misbehavior refers to the event that *CFAN* ranks unclicked suggestions ahead of clicked suggestions. To construct such adversarial perturbations, we introduce a binary classifier, which learns the labels of training instances in V. For each instance, the loss function $\text{Loss}_{\text{cls}}(\Theta)$ of the classifier focuses on reducing the binary cross-entropy [10] between the predicted probabilistic score $\hat{y}^{\Theta}_{\textbf{ins}}$ and the gold standard $y_{\textbf{ins}}$ as follows:

$$\text{Loss}_{\text{cls}}(\Theta) = -[y_{\textbf{ins}} \log(\hat{y}^{\Theta}_{\textbf{ins}}) + (1 - y_{\textbf{ins}}) \log(1 - \hat{y}^{\Theta}_{\textbf{ins}})], \quad (14)$$

where $\hat{y}^{\Theta}_{\textbf{ins}} = g_{\Theta}(\textbf{ins})$.

To increase the classification difficulty, we aim to identify a bounded perturbation $\boldsymbol{r}_{\text{adv}}$ for each **ins** such that the classification loss on $\textbf{ins} + \boldsymbol{r}_{\text{adv}}$ is as large as possible. The adversarial loss of the classifier becomes:

$$\text{Loss}_{\text{cls adv}}(\Theta) = -[y_{\textbf{ins}} \log(\hat{y}^{\Theta}_{\textbf{ins}+\boldsymbol{r}_{\text{adv}}}) + (1-y_{\textbf{ins}}) \log(1-\hat{y}^{\Theta}_{\textbf{ins}+\boldsymbol{r}_{\text{adv}}})], \quad (15)$$

where

$$\boldsymbol{r}_{\text{adv}} = \arg \min_{\boldsymbol{r}, ||\boldsymbol{r}|| \le \epsilon} [y_{\textbf{ins}} \log(\hat{y}^{\hat{\Theta}}_{\textbf{ins}+\boldsymbol{r}}) + (1 - y_{\textbf{ins}}) \log(1 - \hat{y}^{\hat{\Theta}}_{\textbf{ins}+\boldsymbol{r}})]. \quad (16)$$

$\hat{\Theta}$ is the set of current parameters defining the classifier, $\boldsymbol{r}$ is a perturbation on the input **ins**, $\hat{y}^{\hat{\Theta}}_{\textbf{ins}+\boldsymbol{r}}$ gives the predicted score of the perturbed instance based on the current set of parameters, and ϵ is a parameter to bound the perturbations.

Calculating the exact value of $\boldsymbol{r}_{\textbf{adv}}$ is not feasible, because exact minimization of Equation 16 with respect to $\boldsymbol{r}$ is intractable. To derive dynamic perturbations efficiently, we apply fast gradient sign method [8] to approximate the perturbation $\boldsymbol{r}_{\textbf{adv}}$. We linearize the cost function 16 around the current value of $\hat{\Theta}$, and obtain an optimal max-norm constrained perturbation of

$$\boldsymbol{r}_{\textbf{adv}} = -\epsilon \boldsymbol{l} / ||\boldsymbol{l}||_2, \quad (17)$$

where

$$\boldsymbol{l} = \nabla_{\textbf{ins}} \text{Loss}_{\text{cls}}(\hat{\Theta}). \quad (18)$$

Based on Equations 17 and 18, the perturbation for each instance can be calculated efficiently via back-propagation. Adversarial instances can be constructed by adding the derived perturbations $\boldsymbol{r}_{\text{adv}}$ to the corresponding instance embeddings **ins**.

To train a robust suggestion system with good ranking performance, the objective of *CFAN* combines the loss functions of both adversarial classification and ranking as follows:

$$\text{loss}(\Theta) = \sum_{\text{ins} \in V} \text{loss}_{\text{cls adv}}(\Theta) + \text{loss}_{\text{ranking}}(\Theta). \quad (19)$$

The classification component and the ranking component share the same set of parameters. The classification component dynamically generates adversarial instances, while the ranking component generates favorable rankings of query suggestions, pushing clicked suggestions to top positions and pulling unclicked suggestions to bottom positions in ranking lists.

5 EXPERIMENTS

In this section, we conduct extensive experiments and in-depth analysis to verify the performance of *CFAN* for query suggestions.

5.1 Datasets and Experimental Settings

Table 2: The statistics of queries with different context lengths in the training and test datasets.

Dataset	Context Length		
	Short (1 query)	Medium (2-3 queries)	Long (4+ queries)
Training	284,273	105,647	103,944
Test	31,577	11,998	11,566

Dataset. We use a search log collected from one of the largest search engines in the world. We pre-processed the data by eliminating non-alphanumeric characters, spelling error correction, and lower-casing. Then we segmented the search log into sessions, using a standard segmentation heuristic, i.e., intervals of at least 30 minutes idle time denotes a session boundary [12]. To partition search sessions into training and test sets, the first 90% data are utilized for training while the remaining sessions are the test data. Among the training data, 10% of sessions are randomly sampled as the validation set for parameter tuning. Finally, there are 3,684,008 training queries within 493,864 sessions and 406,063 test queries within 55,141 sessions. Furthermore, to evaluate the performance using different context lengths, the test set is partitioned into three subsets, including *Short Context* (1 query), *Medium Context* (2 to 3 queries), and *Long Context* (4 or more queries). Here the context length refers to the number of search queries in a search session. Table 2 shows the statistics of queries with different context lengths in the training and test datasets.

5.2 Evaluation Metrics

We employ Mean Reciprocal Rank (MRR) score [23] as the evaluation metric. Given a search query Q, let $Y(Q)$ be a ranked list of recommended queries determined by a suggestion method. We use $\text{rank}_1(Y(Q))$ to denote the rank of the first clicked query in $Y(Q)$. Formally,

$$\text{MRR} = \frac{1}{\text{rank}_1(Y(Q))}. \quad (20)$$

Essentially, the MRR score summarizes the ranks of the first clicked queries in the recommendation list $Y(Q)$ - A larger score indicates that the first clicked queries are ranked higher in the list.

5.3 Baselines Methods

In our evaluations, we evaluate the MRR performance of *CFAN* against the following state-of-the-art methods.

Table 3: The MRR performance of different methods in the test sets with different context lengths for the task of query suggestion.

Dataset	QVMM [9]	FBS [28]	RC [14]	MPS [7, 20]	Hybrid [1]	HRED [20]	RIN [15]	CFAN
Overall Context	0.441	0.458	0.466	0.478	0.473	0.556	0.573	**0.650**
Short Context (1 query)	0.446	0.473	0.460	0.476	0.473	0.548	0.563	**0.639**
Medium Context (2 to 3 queries)	0.430	0.438	0.477	0.483	0.473	0.564	0.591	**0.661**
Long Context (4 and more queries)	0.431	0.440	0.469	0.474	0.475	0.574	0.593	**0.682**

- **[Query-based Variable Markov Model (QVMM)]** [9] makes query suggestions by learning the probability of query transitions over search sessions with the variable memory Markov model.
- **[Feature Based Suggestion (FBS)]** [28] counts on statistical features to make suggestions. Suggestion candidates are generated based on 1) term matching between search queries and candidates and 2) query co-occurrences in search sessions. Query suggestions are made by ranking candidates based on a combination of statistical features.
- **[Reformulation-based Completion (RC)]** [14] is a non-deep learning based method exploiting query suggestions. 43 reformulation based features are proposed to capture user reformulation behaviors over search sessions with LambdaMART [4].
- **[Most Popular Suggestion (MPS)]** [7, 20] is a maximum likelihood method, which relies on "wisdom of the crowd". It ranks queries by the co-occurrence to the last query in the search sequence.
- **[Hybrid Suggestion (Hybrid)]** [1] considers both the context information and the popularity by ranking candidate queries based on a linear combination between the popularity and the similarity to recent search queries.
- **[Hierarchical Recurrent Encoder-Decoder (HRED)]** [20] is a deep learning based query suggestion method. HRED constructs a hierarchical encoder-decoder structure to model the sequential and hierarchical dependencies across terms and queries to make query suggestions.
- **[Reformulation inference network (RIN)]** [15] is the most state-of-the-art query suggestion method. Query reformulations between consecutive queries in search sessions are explicitly modeled in RIN to make suggestions.

5.4 Experimental Results

Table 3 shows the MRR performance of different methods over various context lengths. QVMM, a variable-memory Markov model, achieves the worst MRR performance among all the query suggestion methods. QVMM counts on query dependencies in search sessions to make query suggestions. The sparsity of search sessions leads to the poor ranking performance of QVMM. Feature based method (FBS) first selects candidate queries based on prefix matching and query dependencies. Then, candidate queries are ranked based on statistical popularity features. Compared with QVMM, FBS improves the overall MRR from 0.441 to 0.458. The reformulation-based completion method (RC) outperforms both QVMM and FBS. Its overall MRR score reaches 0.466. RC relies on generalized query dependency rules to make query suggestions. The improvements against QVMM and FBS demonstrate that generalized reformulation rules are more informative and powerful than simple query dependencies in search sessions. The popularity-based baseline method (MPS) slightly outperforms RC. The overall MRR score further increases from 0.466 to 0.478. It indicates that the query popularity plays an important role in driving users to click the suggestions. This is mainly because when query suggestions are highly related to the search query, users tend to click those popular and trending ones as follow-up search queries to explore. The hybrid suggestion method (Hybrid) performs slightly worse than MPS. The overall MRR score decreases from 0.478 to 0.473. This can be explained by the fact that the term-level similarities between suggestion candidates and search queries in Hybrid are not capable of capturing the their relationship in semantic, and thus fail to rank suggestion candidates properly when most suggestion candidates are similar to search queries in term levels.

Neural network based methods HRED, RIN, and *CFAN* outperform non-deep learning based methods by a large margin in general. HRED achieves MRR scores of 0.556, 0.548, 0.564, and 0.574 on overall, short, medium, and long contexts, respectively. RIN achieves MRR score of 0.573 on average. Among all the methods, *CFAN* achieves the best MRR performance. The MRR scores reach 0.650, 0.639, 0.661, and 0.682 on overall, short, medium, and long search contexts, respectively. In particular, *CFAN* achieves excellent ranking performance when the context information is rich. Rich context contains more user search queries and click feedback, which help deduce the query ambiguity and thus contribute to the ranking performance.

6 CONCLUSION AND FUTURE WORK

In this paper we investigated the problem of feedback-aware query suggestion. A novel neural network based method, *CFAN*, is proposed, which models both user search sequence and click sequence.In addition, to improve the robustness of the query suggestion system, adversarial examples are constructed via a classifier. *CFAN* is optimized by training the adversarial classifier and a ranker simultaneously. Extensive experiments demonstrate that our proposed model significantly outperforms the baseline methods for the task of query suggestion. The improvements are consistent across different datasets of various context lengths.

ACKNOWLEDGMENTS

This project was supported by NIH U01HG008488. We also appreciate the gift funds from NEC Lab and Visa.

REFERENCES

[1] Ziv Bar-Yossef and Naama Kraus. 2011. Context-sensitive query auto-completion. In *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*. 107–116.

[2] Ranieri Baraglia, Carlos Castillo, Debora Donato, Franco Maria Nardini, Raffaele Perego, and Fabrizio Silvestri. 2009. Aging effects on query flow graphs for query suggestion. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*. 1947–1950.

[3] Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014. Open Question Answering with Weakly Supervised Embedding Models. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part I*. 165–180.

[4] Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning* 11, 23-581 (2010), 81.

[5] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*. 875–883.

[6] Zhiqian Chen, Xuchao Zhang, Arnold P. Boedihardjo, Jing Dai, and Chang-Tien Lu. 2017. Multimodal Storytelling via Generative Adversarial Imitation Learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*. 3967–3973.

[7] Mostafa Dehghani, Sascha Rothe, Enrique Alfonseca, and Pascal Fleury. 2017. Learning to Attend, Copy, and Generate for Session-Based Query Suggestion. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*. 1747–1756.

[8] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and Harnessing Adversarial Examples. *CoRR* abs/1412.6572 (2014). arXiv:1412.6572

[9] Qi He, Daxin Jiang, Zhen Liao, Steven C. H. Hoi, Kuiyu Chang, Ee-Peng Lim, and Hang Li. 2009. Web Query Recommendation via Sequential Query Prediction. In *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29 2009 - April 2 2009, Shanghai, China*. 1443–1454.

[10] Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *science* 313, 5786 (2006), 504–507.

[11] Chao Huang, Junbo Zhang, Yu Zheng, and Nitesh V Chawla. 2018. DeepCrime: Attentive Hierarchical Recurrent Networks for Crime Prediction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1423–1432.

[12] Bernard J. Jansen, Amanda Spink, Chris Blakely, and Sherry Koshman. 2007. Defining a session on Web search engines. *JASIST* 58, 6 (2007), 862–871.

[13] Bernard J. Jansen, Amanda Spink, and Tefko Saracevic. 2000. Real life, real users, and real needs: a study and analysis of user queries on the web. *Inf. Process. Manage.* 36, 2 (2000), 207–227.

[14] Jyun-Yu Jiang, Yen-Yu Ke, Pao-Yu Chien, and Pu-Jen Cheng. 2014. Learning user reformulation behavior for query auto-completion. In *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast , QLD, Australia - July 06 - 11, 2014*. 445–454.

[15] Jyun-Yu Jiang and Wei Wang. 2018. RIN: Reformulation Inference Network for Context-Aware Query Suggestion. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*. 197–206.

[16] Ruirui Li, Ben Kao, Bin Bi, Reynold Cheng, and Eric Lo. 2012. DQR: a probabilistic approach to diversified query recommendation. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*. 16–25.

[17] Qiaozhu Mei, Dengyong Zhou, and Kenneth Ward Church. 2008. Query suggestion using hitting time. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008*. 469–478.

[18] Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing* 45, 11 (1997), 2673–2681.

[19] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015*. 373–382.

[20] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A Hierarchical Recurrent Encoder-Decoder for Generative Context-Aware Query Suggestion. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*. 553–562.

[21] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *CoRR* abs/1312.6199 (2013). arXiv:1312.6199

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. 6000–6010.

[23] Ellen M. Voorhees. 1999. The TREC-8 Question Answering Track Report. In *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999*.

[24] Ji-Rong Wen, Jian-Yun Nie, and HongJiang Zhang. 2001. Clustering user queries of a search engine. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*. 162–168.

[25] Bin Wu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2018. Query Suggestion with Feedback Memory Network. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. 1563–1571.

[26] Xian Wu, Yuxiao Dong, Chao Huang, Jian Xu, Dong Wang, and Nitesh V Chawla. 2017. Uapd: Predicting urban anomalies from spatial-temporal data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 622–638.

[27] Xian Wu, Baoxu Shi, Yuxiao Dong, Chao Huang, Louis Faust, and Nitesh V Chawla. 2018. Restful: Resolution-aware forecasting of behavioral time series data. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1073–1082.

[28] Dawei Yin, Yuening Hu, Jiliang Tang, Tim Daly Jr., Mianwei Zhou, Hua Ouyang, Jianhui Chen, Changsung Kang, Hongbo Deng, Chikashi Nobata, Jean-Marc Langlois, and Yi Chang. 2016. Ranking Relevance in Yahoo Search. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 323–332.

Learning Fast Matching Models from Weak Annotations

Xue Li*
Microsoft
xeli@microsoft.com

Zhipeng Luo*
Microsoft
Zhipeng.Luo@microsoft.com

Hao Sun
Microsoft
hasun@microsoft.com

Jianjin Zhang†
Microsoft
Tsinghua University
swjxzjj@gmail.com

Weihao Han
Microsoft
weihan@microsoft.com

Xianqi Chu
Microsoft
Xianqi.Chu@microsoft.com

Liangjie Zhang
Microsoft
liazha@microsoft.com

Qi Zhang
Microsoft
Zhang.Qi@microsoft.com

ABSTRACT

We propose a novel training scheme for fast matching models in Search Ads, motivated by practical challenges. The first challenge stems from the pursuit of high throughput, which prohibits the deployment of inseparable architectures, and hence greatly limits its model accuracy. The second problem arises from the heavy dependency on human provided labels, which are expensive and time-consuming to collect, yet how to leverage unlabeled search log data is rarely studied. The proposed training framework targets on mitigating both issues, by treating the stronger but undeployable models as annotators, and learning a deployable model from both human provided relevance labels and weakly annotated search log data. Specifically, we first construct multiple auxiliary tasks from the enumerated relevance labels, and train the annotators by jointly learning from those related tasks. The annotation models are then used to assign scores to both labeled and unlabeled training samples. The deployable model is firstly learnt on the scored unlabeled data, and then fine-tuned on scored labeled data, by leveraging both labels and scores via minimizing the proposed label-aware weighted loss. According to our experiments, compared with the baseline that directly learns from relevance labels, training by the proposed framework outperforms it by a large margin, and improves data efficiency substantially by dispensing with 80% labeled samples. The proposed framework allows us to improve the fast matching model by learning from stronger annotators while keeping its architecture unchanged. Meanwhile, it offers a principled manner to leverage search log data in the training phase, which could effectively alleviate our dependency on human provided labels.

CCS CONCEPTS

• **Information systems → Sponsored search advertising**.

*Both authors contributed equally to this work.
†This work is conducted during internship at Microsoft.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313466

KEYWORDS

Search Ads; relevance matching; teacher-student; weak annotations

ACM Reference Format:
Xue Li, Zhipeng Luo, Hao Sun, Jianjin Zhang, Weihao Han, Xianqi Chu, Liangjie Zhang, and Qi Zhang. 2019. Learning Fast Matching Models from Weak Annotations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313466

1 INTRODUCTION

The recent decade has witnessed a booming of sponsored search due to the outburst of search demands, which raises severe challenges to the design of practical search algorithms. As a starting step of typical sponsored search systems, the fast matching model discussed here is designed to recall from the entire ad corpus a list of candidates best matching a given query. All the subsequent processing steps will be applied on the selected list only. The extremely intensive user requests of Search Ads system poses stringent online latency requirement on the fast matching model. To some extent, its accuracy and latency forms the upper bound of the overall performance. To improve the fast matching model, we need either to leverage more powerful model architectures, or more effective training strategies. However, both directions turn out to be difficult, and will suffer from restrictions from both system and data.

The first restriction relates to model architecture. Commercial search engines need to handle extremely intensive user requests by searching from a daunting ad corpus, and their pursuit of high throughput prohibits the deployment of inseparable model architectures, especially those involving heavy online computations. For this reason, relevance model is usually implemented as Siamese networks such as CDSSM [25], which transforms the input query and ad individually to representing vectors residing in a common inner-product space, without knowledge of each other. Such an architecture allows us to reduce online computations following a space-for-time strategy, by hosting online a dictionary of precomputed ad vectors. When a query is issued, the model simply retrieves relevant ad vectors by matching it from the dictionary, and the matching could be effectively done by resorting to approximate nearest neighbor search algorithms such as NGS [28]. However, due to the deferment of interactions between queries and ads, Siamese

networks often suffer from losing details important to the matching task in the forward pass [11], which greatly limits matching performance. Note that better accuracy could also be achieved using more complex model architectures, especially those involving interactions between query and ad such as Deep Crossing, or model ensembles involving crossing features. However, such models are mostly inseparable, hence could not be deployed in the same way as Siamese networks.

The second restriction comes from the training data. Relevance model is typically trained on human provided labels, which is expensive and time-consuming to collect. In contrast to this, the search log contains tremendous amount of data, but relevance labels are unavailable. An alternative solution is to employ user clicks as a surrogate of relevance labels, by taking clicked <*query*, *ad*> pair as positive, and synthesizing negative pairs by fixing a query while randomly sampling ads from the same training batch. This strategy oversimplifies the learning task by transforming matching to a classification problem [29], and introduces ambiguities from three aspects. First of all, the arbitrariness and subjectivity of user behavior leads to a misalignment between user clicks and true relevance labels, polluting the training set with false positives. Secondly, data synthesizing may introduce false negatives, and most of the synthetic negative pairs share no common terms in queries and ads. Learning on such data may mislead the model to consider common terms as critical evidences of relevance, whereas lexically similar query and ad may have totally different intents, such as *iphone* and *iphone cover*. Last but not least, human labels typically have enumerated values to distinguish relevance with finer granularity, which are difficult to approximate by binary-valued clicks.

This paper targets on mitigating both the aforementioned issues. For this purpose, we employ powerful but undeployable models to assign weak annotations or scores to unlabeled data, and improve the deployable model by learning from both the weak annotations and true human labels. A schematic illustration of the proposed framework is plotted in Figure 1, which consists of four major steps. Specifically, we first construct auxiliary tasks using enumerated relevance labels, and train annotators by jointly learning multiple tasks, following the idea of Multi-task Learning (MTL). The learnt annotators are then used to assign scores to both unlabeled and labeled samples. Based on this, a deployable model is trained firstly on the scored unlabeled data, and then fine-tuned on scored labeled data. A label-aware weighted loss function is proposed in the last step to leverage both labels and scores.

The advantages of the proposed framework are twofold. At the foremost, it offers us great flexibility in the architecture design of fast matching model, since we can improve it by learning from stronger annotators, while keeping the model to be deployed as efficient as possible. Because the annotation model runs offline only, it is allowed to explore complicated model architectures involving heavy computations, and is able to leverage either policy-driven rules or features that might be too heavy to access for online processing. Secondly, this framework provides a principled way to leverage unlabeled data for learning fast matching models, which dispenses entirely with user clicks to avoid data pollution.

According to our experiments, compared with training directly on relevance labels, the proposed framework is able to improve *ROC AUC* by 4.5% and *PR AUC* by 3.2%, demonstrating the effectiveness of improving relevance model by leveraging unlabeled data and stronger annotators. In addition, with only 20% labeled samples, we are able to achieve performance comparable to the baseline trained on the entire labeled set, which effectively alleviates our dependency on human provided labels.

Our main contributions are summarized as follows.

- we propose a novel training framework for the learning of fast matching models in search ads, which is able to leverage more powerful but undeployable models as weak annotators, and learn a simple and deployable model from both the weak annotations and the true human labels. The weak annotators are trained following MTL scheme to fully leverage the fine-grained relevance labels, and a label-aware weighted loss is proposed to softly combine the weak annotations and human provided labels.
- we evaluate the proposed training framework on large datasets, where both *PR AUC* and *ROC AUC* are significantly improved, demonstrating its effectiveness. Our experiments also shows that training by the proposed framework could greatly reduce our dependency on human labels.

2 RELATED WORK

The proposed framework is mainly fueled by the recent progress in web search and model compression.

Existing methods on semantically matching queries and documents could be roughly grouped into two categories, namely traditional approaches and deep learning based models. Representative methods falling into the former category include LSA [5], pLSA [10], topic models such as LDA [3], Bi-Lingual Topic Models [7], etc. Methods belonging to the latter category are designed to extract semantic information via deep learning architectures, including auto-encoders [20], Siamese networks [6, 11, 12, 22, 24–26], interaction-based networks [11, 15, 27, 30, 31], and lexical and semantic matching networks [8, 17], etc. Readers may refer to [16] for a tutorial on this topic. The framework proposed in this paper adopts CDSSM [25] and Deep Crossing [23] as key building blocks. However, instead of improving its architecture, we alternatively attempt to improve its training scheme in this paper.

The proposed framework also draws inspirations from the teacher-student scheme in model compression. As an initial attempt, the authors of [4] train a smaller model using synthetic data labeled by a teacher model. In [1], the authors propose to train the student model by mimicking the logit values of the teacher model, instead of directly learning the softmax outputs. This is extended later in [19], by incorporating intermediate hidden layer outputs as target values as well. Hinton et al. [9] propose to adjust the temperature variable in the softmax function. Recently the authors of [21] introduce a noise-based regularizer while training student models. In this paper, the teacher-student framework is adapted in several aspects to improve relevance algorithms in sponsored search.

While writing this paper, we also notice a recent work in [29], which extends the teacher-student scheme to the learning of retrieval-based chatbots, where a sequence-to-sequence model is employed as a weak annotator, and the student model is trained by learning from both the weak signals and the unlabeled data. Both [29] and

our proposed framework are motivated by the success of teacher-student scheme, but we adapt it in different directions to align with our respective task.

3 PROPOSED FRAMEWORK

3.1 Overview

3.1.1 Sponsored Search. Below we list some useful terminologies, and we refer readers to [23] for more details.

Impression is an instance of an ad displayed to a user. *Click* indicates whether an impression is clicked by a user. *Click-Through Rate* or *CTR* is the ratio of the total number of clicks to total number of impressions. *Query* is a text string issued by a user in a search engine. *Keyword* is a text string concerning a certain product, specified by an advertiser. *Landing Page* or *LP* refers to the web page a user reaches after clicking the corresponding ad of a product. A brief description of the product is usually displayed as the title of the *LP*, called *LP* title. *Ad Copy* or *AC* refers to the content seen by a user before clicking, which typically includes ad title, description, displayed URL, landing page link, etc. *Listing* contains the bided keywords associated with Ad Copy, bid, etc. *Relevance labels* are a set of predefined judgements used to characterize the relevance of query and ad listing. We have two label sets, namely *AC* labels that measure the relevance between query and Ad Copy contents, and *LP* labels that measure relevance between query and landing page contents. Throughout this paper, our label set is {0, 1, 2, 3, 4} for *AC* labels, and {0, 1, 2, 3, 4, 5} for *LP* labels, with higher labels indicating better relevance.

Our system is akin to the one described in [14], which consists of several steps including *selection*, *relevance filtration*, *CTR prediction*, *ranking* and *allocation*.

3.1.2 Proposed Framework. Figure 1 sketches the main idea of the proposed framework, which contains four major steps. The first step trains annotation models on human labeled data, by jointly learning multiple related tasks. The second step scores both the labeled and unlabeled data using this annotation model. In the following step, a deployable model is trained on the scored unlabeled data. Finally, the deployable model is fine-tuned on both the labels and scores of the labeled data, by minimizing the label-aware weighted loss. All steps involve purely offline computations; only the final simple model will be deployed online. We instantiate our deployable model as CDSSM [24], and implement two annotation models, i.e. Deep Crossing (abbreviated as DC) in [23] and a decision tree ensemble model (abbreviated as DT) in [14].

3.2 Annotation Model Training via MTL

Two annotation models are considered in this paper, i.e. Deep Crossing and decision tree ensemble. Figure 2 shows the structure of Deep Crossing, which contains an input layer, an embedding layer, multiple residual layers, and a scoring layer. Deep Crossing introduces feature interactions immediately after embedding layers, hence is inseparable. Our decision tree ensemble model follows the one described in [14], which takes as input a set of features, some of which are difficult to access in the recall phase. Therefore, both annotation models could not be deployed as fast matching models directly. Annotation models are trained on labeled data, where each sample is a triple of query, ad listing and human judged relevance label chosen from a predefined label set, see Section 3.1.

We construct several auxiliary classification tasks by partitioning the label sets with different pivots, as shown in Table 1, where four auxiliary tasks are constructed for *LP* labels, and three for *AC* labels, along with the main task. The final loss is calculated as a weighted sum of the individual loss for each task, with the main task weighted as 0.5 while all the auxiliary tasks evenly weighted. The final score is computed analogously.

Table 1: Constructing auxiliary tasks.

Label Set	Task	Negative Labels
AC	main task	{0}
LP		{0}
AC	aux. task 1	{0, 1}
	aux. task 2	{0, 1, 2}
	aux. task 3	{0, 1, 2, 3}
LP	aux. task 1	{0, 1}
	aux. task 2	{0, 1, 2}
	aux. task 3	{0, 1, 2, 3}
	aux. task 4	{0, 1, 2, 3, 4}

3.3 CDSSM Training using Scores

The architecture of CDSSM is shown in Figure 3, which contains an input layer, a word hashing layer to transform each word into a count vector of letter-tri-grams, a convolutional layer to extract sliding window based contextual features around each word, a max-pooling layer to summarize most salient contextual features, followed by a semantic layer and the final cosine scoring layer.

The scores from the annotation models could be leveraged either in a hard manner by thresholding them to binary values, or in a soft way by maintaining their original values. In addition, the reliability of the scores also needs investigation. For this purpose, we define the targets and weights of the unlabeled samples as functions of annotation model scores:

$$y_i = f(s_i), \tag{1}$$

$$\omega_i = g(s_i), \tag{2}$$

where y_i, ω_i and s_i are the target, weight, and annotation model score for the ith sample in the unlabeled dataset, respectively, while $f(\cdot)$ and $g(\cdot)$ are mapping functions from scores to targets and weights. When multiple annotation models are available, s_i is calculated as their arithmetic mean. Following this definition, we define several mapping functions as below:

$$f_1(s_i) = \begin{cases} 1, & 0.5 \le s_i < 1; \\ 0, & 0 < s_i < 0.5; \end{cases} \tag{3}$$

$$f_2(s_i) = s_i, \tag{4}$$

$$g_1(s_i) = \begin{cases} 0, & t_1 < s_i < t_2; \\ 1, & \text{o.w.} \end{cases} \tag{5}$$

$$g_2(s_i) = \left|2s_i - 1\right|^p, \tag{6}$$

$$g_3(s_i) = 1. \tag{7}$$

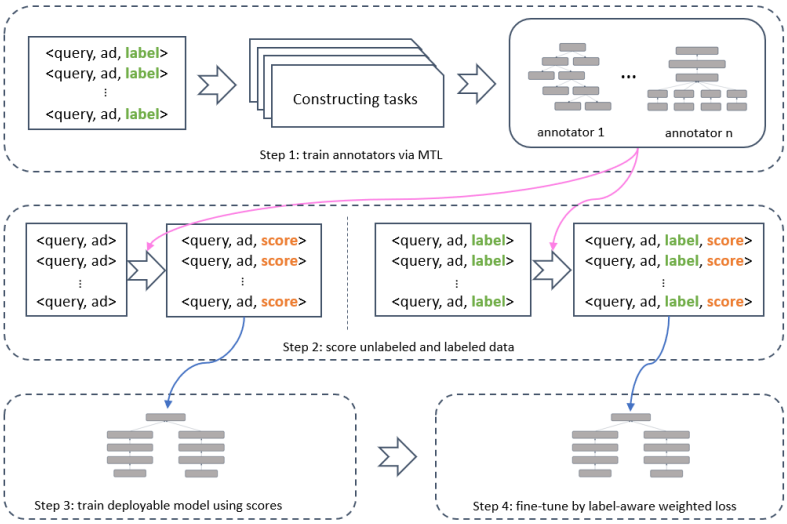


Figure 1: Illustration of the proposed framework, which includes four steps. The first step constructs multiple related tasks, and trains annotation models using labeled data, via MTL. The second step scores both the labeled and unlabeled data using the annotators. A deployable model is trained on the scored unlabeled data in the third step. In the last step, the deployable model is fine-tuned by minimizing the label-aware weighted loss. All steps involve offline computations only.

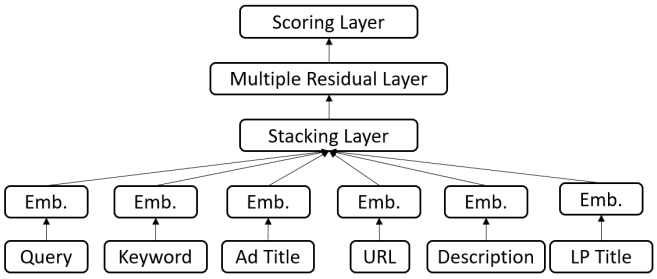


Figure 2: Architecture of Deep Crossing, which is inseparable and cannot be deployed as a fast matching model.

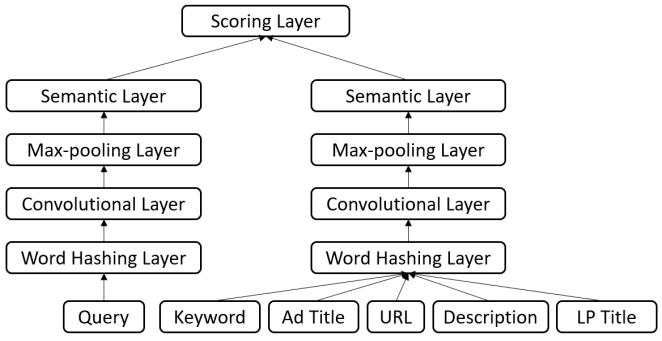


Figure 3: Architecture of CDSSM, which is separable and takes identical inputs with Deep Crossing.

Equation (3) defines a hard target mapping, and Equation (4) defines a soft one. Equation (5) to Equation (7) define three weight mappings. In implementation, Equation (3) is trained by minimizing cross-entropy loss, while Equation (4) by weighted mean square error (MSE) loss. Specifically, denoting the predicted score from CDSSM as $\hat{y}_i$ for the ith sample, we define its sample loss as below:

$$\ell_i = \omega_i \cdot (y_i - \hat{y}_i)^2. \tag{8}$$

3.4 Fine-tuning by Label-aware Weighted Loss

To balance human judged relevance labels and annotation model scores, we propose the label-aware weights, as illustrated in Figure 4. Denote the binary relevance labels as $\tilde{y}_i$ for sample i, where the binary labels are defined according to the main task in Table 1. As shown in the left of Figure 4, $\tilde{y}_i = 1$ indicates that sample i has positive relevance labels, making it more acceptable when the predicted score $\hat{y}_i$ is greater than the target value y_i obtained from annotation models, and less acceptable when $\hat{y}_i < y_i$. To reflect such differences, we define a weight $\tilde{\omega}_i$ for sample i, with $\tilde{\omega}_i < 1$ for acceptable loss and $\tilde{\omega}_i = 1$ otherwise. The right part of Figure 4 shows the opposite case, where $\tilde{y}_i = 0$ makes $\hat{y}_i > y_i$ less acceptable.

To facilitate the formulation of label-aware weights, we first define a sign function as shown below:

$$\delta_\theta(x) = \begin{cases} \theta, & \text{if } x \le 0; \\ 1, & \text{if } x > 0, \end{cases} \tag{9}$$

where $0 \le \theta \le 1$ is a hyper-parameter indicating the discount of punishment imposed on acceptable loss. Based on the sign function in Equation (9), our label-aware weighted loss could be defined as

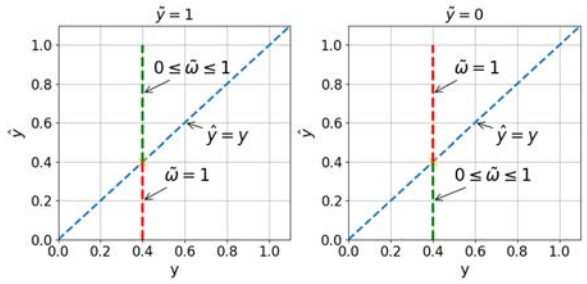


Figure 4: Illustration of the label-aware weights, where $\tilde{y}$ is the binary label, y is the target from annotation model scores, and $\hat{y}$ is the prediction of CDSSM. When $\tilde{y}_i = 1$ (left), $\hat{y}_i > y_i$ appears more acceptable than $\hat{y}_i < y_i$, and hence is assigned a smaller weight, as highlighted by green. The right part shows $\tilde{y}_i = 0$, where $\hat{y}_i < y_i$ is given a smaller weight.

Table 2: Statistics of datasets.

	Unlabeled	Clicked	Labeled	
			positive	negative
train	159,352 K	168,671 K	5,115 K	9,752 K
test	–	–	132 K	120 K

follows:

$$\begin{aligned}\tilde{\ell}_i &= \delta_\theta(y_i - \hat{y}_i)^{\tilde{y}_i} \cdot \left(\theta + 1 - \delta_\theta(y_i - \hat{y}_i)\right)^{(1-\tilde{y}_i)} \cdot \\ &\quad (y_i - \hat{y}_i)^2 \\ &\triangleq \tilde{\omega}_i \cdot (y_i - \hat{y}_i)^2, \end{aligned} \tag{10}$$

where label-aware weight $\tilde{\omega}_i$ is a function of y_i, $\tilde{y}_i$ and $\hat{y}_i$.

4 EXPERIMENTS

4.1 Experimental Setup

4.1.1 Datasets. Validating the proposed framework needs three datasets: a labeled one with human judged relevance labels (referred to as labeled), an unlabeled one with both clicked and unclicked samples (referred to as unlabeled), and an unlabeled one with only clicked samples (referred to as clicked). The unlabeled and clicked datasets are sampled from the search log of a commercial search engine. A subset with comparable size to the test set is randomly sampled from the training labeled set for validation. Statistics of the three datasets are shown in Table 2.

4.1.2 Implementation. We implement Deep Crossing following the description of [23], with several modifications including adding Batch Normalization [13] and soft-aligning query and ad using the neural attention similar to [2, 18], by concatenating ad inputs into a single string. We use identical CDSSM architecture in the last two steps in Figure 1. we conduct evaluations using *ROC AUC* and *PR AUC* as measurements, which represent the area under the Receiver Operating Characteristic curve and the Precision-Recall curve, respectively.

Table 3: Comparison of performance.

Annotator	CDSSM	*ROC AUC*	*PR AUC*
–	CDSSM-click	0.7185	0.7890
	CDSSM-labeled	0.8131	0.8711
DC	–	0.8425	0.8946
DT		0.8448	0.8940
DC + DT		0.8578	0.9034
DC	CDSSM	0.7937	0.8625
DT		0.7936	0.8604
DC + DT		0.8155	0.8753
DC	CDSSM-ft	0.8394	0.8939
DT		0.8442	0.8941
DC + DT		**0.8497**	**0.8990**

4.2 Performance

Table 3 summarizes our main results, where DC and DT represent Deep Crossing and decision tree ensemble, respectively, and DC + DT represents their combination. CDSSM-ft refers to fine-tuning using labeled data after training CDSSM on scored, unlabeled samples. As mentioned in Section 1, training by clicked data means taking clicked <*query*, *ad*> pair as positive, and randomly synthetic query and listing pairs as negative.

The first row in Table 3 presents the results of our click and labeled baseline, obtained by training CDSSM purely on user clicks and relevance labels. The next row gives results of our annotation models, where DC and DT achieves comparable performance. Their combination brings further improvements. Training by the proposed framework with DC as annotation model outperforms the click baseline by 9.8% for *ROC AUC*, and 9.32% by *PR AUC*. Very close result is observed for DT. Learning by the proposed framework achieves comparable performance to the labeled baseline when combining DT and DC. Finally, when further fine-tuned on labeled data, the *ROC AUC* using annotation model DC and DT outperforms the labeled baseline by 2.6% and 3.82% respectively. Combining DC and DT as annotation model outperforms the labeled baseline by a large margin of 4.5% for *ROC AUC*, and 3.2% for *PR AUC*. This demonstrates evidently the effectiveness of the proposed framework.

4.3 Annotation Models and Mapping Functions

Several configurations of annotation models and mapping functions are compared, with results summarized in Table 4. Specifically, the annotation model is implemented as several variants of Deep Crossing, including the original version, Deep Crossing with Batch Normalization (BN), Deep Crossing with BN and Attention (abbreviated as A), Deep Crossing with BN and MTL, and Deep Crossing with BN, attention and MTL. For each annotation model, three mapping functions are implemented, and *AUC* before and after fine-tuning are reported, with $\theta = 1$.

As shown in Table 4, the performance of CDSSM are highly correlated with that of annotation models. In addition, Table 4 also demonstrates that training annotation models by jointly learning multiple tasks could bring about considerable gains in both annotation models and CDSSM, despite its simplicity. Table 4 also reveals

Table 4: Influence of annotation models and mapping functions.

Annotator	*AUC*-annotator		*AUC*-CDSSM						
			before/after	f_1 g_1		f_1 g_2		f_2 g_3	
	ROC	*PR*	fine-tune	*ROC*	*PR*	*ROC*	*PR*	*ROC*	*PR*
DC	0.8046	0.8659	before	0.7164	0.8146	0.7273	0.8186	0.7143	0.8116
			after	0.7616	0.8341	0.7591	0.8328	0.7567	0.8304
DC-BN	0.8328	0.8877	before	0.7704	0.8464	0.7759	0.8496	0.7778	0.8517
			after	0.8219	0.8773	0.8262	0.8814	0.8301	0.8841
DC-BN-A	0.8351	0.8886	before	0.7744	0.8502	0.7797	0.8524	0.7862	0.8576
			after	0.8254	0.8838	0.8268	0.8848	0.8330	0.8887
DC-BN-MTL	0.8391	0.8932	before	0.7823	0.8565	0.7847	0.8578	0.7890	0.8605
			after	0.8277	0.8857	0.8296	0.8868	0.8343	0.8896
DC-BN-A-MTL	0.8425	0.8946	before	0.7761	0.8527	0.7851	0.8571	0.7937	0.8625
			after	0.8281	0.8863	0.8309	0.8887	0.8372	0.8922

Table 5: Comparison of different fine-tuning methods.

		ROC AUC	*PR AUC*
baseline	hard	0.8271	0.8833
	soft	0.8372	0.8922
θ	0.0	0.8390	0.8935
	0.2	0.8391	0.8935
	0.5	**0.8394**	0.8939
	0.8	0.8378	0.8926

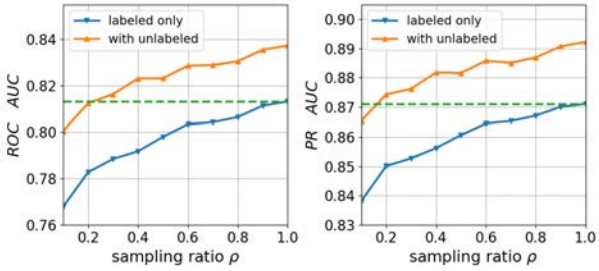


Figure 5: Performance of sampling labeled data by different ratios. Blue curves represent training on the sampled data, yellow curves represent training by the proposed method.

the importance of mapping functions, where the third configuration outperforms others in most cases, across different annotation models. This may due to the reason similar to prior analysis in [9], that the soft targets could offer more information than hard targets.

4.4 Label-aware Weights

To show the influence of label-aware weights, we conduct experiments on DC-BN-A-MTL by setting θ to be {0, 0.2, 0.5, 0.8, 1.0}, with results summarized in Table 5, where the hard baseline refers to directly learning binary label $\tilde{y}$, and the soft baseline refers to learning target y.

According to Table 5, fine-tuning by the hard baseline yields suboptimal performance, which coincides with prior analysis in [9]. In addition, the best *AUC* is achieved at $\theta = 0.5$, which equivalents to setting the weights for unacceptable loss twice as much as that of the acceptable ones. Compared with the case $\theta = 0.5$, both $\theta = 0$ and $\theta = 1$ give inferior performance. More specifically, $\theta = 1$ exhibits a larger gap with $\theta = 0.5$, due to its inability to leverage human provided labels. In contrast, $\theta = 0$ simply spares all acceptable loss, and hence has limited influence on the overall performance.

4.5 Improve Data Efficiency

This section investigates the potential of complementing labeled data using unlabeled samples. To figure out this, we randomly sample 9 subsets from the labeled data using a sampling ratio ρ ranging from 0.1 to 0.9, and repeat the steps in Figure 1 using the sampled labeled data only. The results are shown in Figure 5, where the blue curves show the *AUC* obtained by training CDSSM directly on the sampled labeled data, while the yellow curves denote performance achieved by following the proposed framework.

According to Figure 5, all *AUC* curves grow monotonically as ρ gets larger, but both *ROC AUC* and *PR AUC* grows slower and slower, indicating decreasing data efficiency. The green dashed lines in Figure 5 highlight the upper bound of the blue curves, which is easily exceeded when training using the proposed framework: the same *AUC* is achieved by the orange curve with only 20% labeled data. This validates that training by the proposed framework could greatly improve data efficiency. With 80% knowledge could be obtained from unlabeled data, our dependency on the expensive human labels could be greatly reduced.

5 CONCLUSIONS

This paper proposes a novel framework for learning the relevance model in sponsored search, allowing us to learn a simple model from stronger but undeployable annotators, and offers a principled way to leverage unlabeled data by completely dispenses clicks. Our main experimental results include 1) training by the proposed framework could substantially improve the performance of relevance models, and 2) the proposed framework is able to significantly increase data efficiency, so that training by 20% labeled data could achieve comparable results with the baseline trained by all labeled samples.

REFERENCES

[1] Jimmy Ba and Rich Caruana. 2014. Do Deep Nets Really Need to be Deep? In *Advances in Neural Information Processing Systems*. 2654–2662.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. (2014). arXiv:1409.0473 http://arxiv.org/abs/1409.0473

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3 (2003), 993–1022.

[4] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model Compression. In *International Conference on Knowledge Discovery and Data Mining*. ACM, 535–541.

[5] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science* 41 (1990), 391–407.

[6] Jianfeng Gao, Patrick Pantel, Michael Gamon, Xiaodong He, Li Deng, and Yelong Shen. 2014. Modeling Interestingness with Deep Neural Networks. In *Conference on Empirical Methods in Natural Language Processing*. 2–13.

[7] Jianfeng Gao, Kristina Toutanova, and Wen-tau Yih. 2011. Clickthrough-based Latent Semantic Models for Web Search. In *International SIGIR Conference on Research and Development in Information Retrieval (SIGIR '11)*. ACM, 675–684.

[8] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A Deep Relevance Matching Model for Ad-hoc Retrieval. In *International Conference on Information and Knowledge Management*. ACM, 55–64.

[9] G. Hinton, O. Vinyals, and J. Dean. 2015. Distilling the Knowledge in a Neural Network. (March 2015). arXiv:stat.ML/1503.02531

[10] Thomas Hofmann. 1999. Probabilistic Latent Semantic Analysis. In *Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 289–296.

[11] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional Neural Network Architectures for Matching Natural Language Sentences. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2042–2050.

[12] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning Deep Structured Semantic Models for Web Search Using Clickthrough Data. In *International Conference on Information and Knowledge Management*. ACM, 2333–2338.

[13] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. (2015). arXiv:1502.03167 http://arxiv.org/abs/1502.03167

[14] Xiaoliang Ling, Weiwei Deng, Chen Gu, Hucheng Zhou, Cui Li, and Feng Sun. 2017. Model Ensemble for Click Prediction in Bing Search Ads. In *International Conference on World Wide Web Companion*. 689–698.

[15] Zhengdong Lu and Hang Li. 2013. A Deep Architecture for Matching Short Texts. In *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 1367–1375. http://papers.nips.cc/paper/5019-a-deep-architecture-for-matching-short-texts.pdf

[16] Bhaskar Mitra and Nick Craswell. 2017. Neural Models for Information Retrieval. (2017). arXiv:1705.01509 http://arxiv.org/abs/1705.01509

[17] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to Match Using Local and Distributed Representations of Text for Web Search. In *International Conference on World Wide Web*. 1291–1299.

[18] Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A Decomposable Attention Model for Natural Language Inference. (2016). arXiv:1606.01933 http://arxiv.org/abs/1606.01933

[19] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. FitNets: Hints for Thin Deep Nets. (2014). arXiv:1412.6550 http://arxiv.org/abs/1412.6550

[20] Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Semantic Hashing. *International Journal of Approximate Reasoning* 50, 7 (2009), 969 – 978.

[21] Bharat Bhusan Sau and Vineeth N. Balasubramanian. 2016. Deep Model Compression: Distilling Knowledge from Noisy Teachers. (2016). arXiv:1610.09650 http://arxiv.org/abs/1610.09650

[22] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In *International Conference on Research and Development in Information Retrieval*. ACM, 373–382.

[23] Ying Shan, T. Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and JC Mao. 2016. Deep Crossing: Web-Scale Modeling Without Manually Crafted Combinatorial Features. In *International Conference on Knowledge Discovery and Data Mining*. ACM, 255–262.

[24] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A Latent Semantic Model with Convolutional-Pooling Structure for Information Retrieval. In *International Conference on Information and Knowledge Management*. ACM, 101–110.

[25] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning Semantic Representations using Convolutional Neural Networks for Web Search. In *International Conference on World Wide Web*. ACM, 373–374.

[26] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. (2015). arXiv:1503.00075 http://arxiv.org/abs/1503.00075

[27] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations. In *AAAI Conference on Artificial Intelligence*. 2835–2841.

[28] Jingdong Wang and Shipeng Li. 2012. Query-driven Iterated Neighborhood Graph Search for Large Scale Indexing. In *International Conference on Multimedia*. ACM, 179–188.

[29] Yu Wu, Wei Wu, Zhoujun Li, and Ming Zhou. 2018. Learning Matching Models with Weak Supervision for Response Selection in Retrieval-based Chatbots. (2018). arXiv:1805.02333 http://arxiv.org/abs/1805.02333

[30] Liu Yang, Qingyao Ai, Jiafeng Guo, and W. Bruce Croft. 2016. aNMM: Ranking Short Answer Texts with Attention-Based Neural Matching Model. In *International Conference on Information and Knowledge Management*. ACM, 287–296.

[31] Wenpeng Yin, Hinrich SchÃijtze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *Transactions of the Association for Computational Linguistics* 4 (2016), 259 – 272.

Multistream Classification for Cyber Threat Data with Heterogeneous Feature Space

Yi-Fan Li
The University of Texas at Dallas
Richardson, Texas
yli@utdallas.edu

Yang Gao
The University of Texas at Dallas
Richardson, Texas
yxg122530@utdallas.edu

Gbadebo Ayoade
The University of Texas at Dallas
Richardson, Texas
gga110020@utdallas.edu

Hemeng Tao
The University of Texas at Dallas
Richardson, Texas
htx160430@utdallas.edu

Latifur Khan
The University of Texas at Dallas
Richardson, Texas
lkhan@utdallas.edu

Bhavani Thuraisingham
The University of Texas at Dallas
Richardson, Texas
bhavani.thuraisingham@utdallas.edu

ABSTRACT

Under a newly introduced setting of multistream classification, two data streams are involved, which are referred to as source and target streams. The source stream continuously generates data instances from a certain domain with labels, while the target stream does the same task without labels from another domain. Existing approaches assume that domains for both data streams are identical, which is not quite true in real world scenario, since data streams from different sources may contain distinct features. Furthermore, obtaining labels for every instance in a data stream is often expensive and time-consuming. Therefore, it has become an important topic to explore whether labeled instances from other related streams can be helpful to predict those unlabeled instances in a given stream. Note that domains of source and target streams may have distinct features spaces and data distributions. Our objective is to predict class labels of data instances in the target stream by using the classifiers trained by the source stream.

We propose a framework of multistream classification by using projected data from a common latent feature space, which is embedded from both source and target domains. This framework is also crucial for enterprise system defenders to detect cross-platform attacks, such as Advanced Persistent Threats (APTs). Empirical evaluation and analysis on both real-world and synthetic datasets are performed to validate the effectiveness of our proposed algorithm, comparing to state-of-the-art techniques. Experimental results show that our approach significantly outperforms other existing approaches.

CCS CONCEPTS

• **Computing methodologies** → **Transfer learning**; • **Information systems** → *Data stream mining*; • **Security and privacy** → Intrusion detection systems.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313572

KEYWORDS

Attack detection; Multistream classification; Domain adaptation

ACM Reference Format:
Yi-Fan Li, Yang Gao, Gbadebo Ayoade, Hemeng Tao, Latifur Khan, and Bhavani Thuraisingham. 2019. Multistream Classification for Cyber Threat Data with Heterogeneous Feature Space. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313572

1 INTRODUCTION

Data streams are crucial in the modern connected Internet world, and they have attracted the attention of researchers worldwide. Given important applications of data streams–such as IoT, social networks, and surveillance–mining them properly is becoming a more and more important topic to explore. However, data stream mining is also a challenging task due to its distinctive nature, e.g., theoretically infinite in length, heterogeneous domains, and lack of label for emerging data [12, 14]. These unique properties are discussed seperately in the following context.

Regarding *data stream mining*, concept drift had been a challenge problem due to its theoretical infinite length. Detecting it in multivariate data streams concentrates on tracking any changes in the posterior class distribution, $P(y|x)$. Instead of tracking changes in $P(y|x)$ directly, the approaches proposed in [13] adopt the principle [11] to detect this change indirectly by tracking drift in the error rate of the underlying classifier. However, tracking drift in the error rate requires true labels of test data instances, which are scarce in practice. Recent studies focus on confidence [8] to detect changes in distribution between two different time windows or always buffering the most recent concept [19]. Apart from working on a single stream, these methods explicitly detect change points.

For *domain adaptaion*, the problem setting described in this paper is motivated by a Multistream Classification framework [8]. One of fundamental assumptions in data mining, known as the "stationary distribution assumption", is that both the training and test data are generated from the same domain and thus represent the same data distribution [25]. Therefore, common techniques normally cannot be directly deployed when training and test data are from different domains. The differences between domains can be normally considered as two aspects: 1. distinct number of features; 2. distinct feature distributions. Several approaches have been proposed to learn a

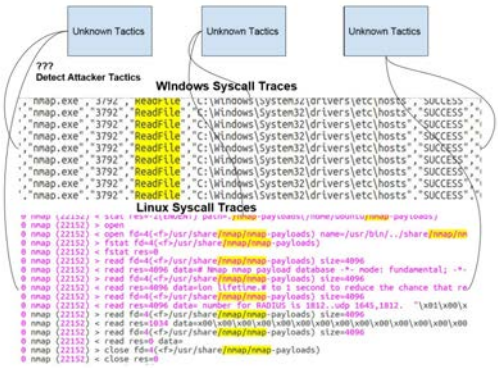


Figure 1: Sample Trace File showing syscall traces for an attacker scanning session with nmap network scanning tool

common feature representation for domain adaptation [2, 20, 23, 26]. The state-of-the-art methods, including deep learning, mostly address domain adaptation problems for stationary data except [26]. Thus, fitting domain adaptation frameworks to online data streams is yet to explore.

The multistream setting is very common in *cyber security area*. For example, organizations deploy threat monitoring tools that collect network system call events as data streams. Linking low-level traces to attacker tactics and intent is challenging. In Figure 1, a snapshot of a trace file is shown as scanning session of an attacker using the nmap tool to scan a victim's network. Also, data streams obtained from Linux machines cannot be used to predict attack events on a host with a Windows operating system. This is due to the difference in the way events are recorded on both systems (features may vary from Linux to Windows or vice versa). Machine learning algorithms have been deployed to learn attackers' behavioral patterns either using host-based [1, 6, 7, 24] or network-based [3, 10, 16, 17] techniques.

To sum up, the problem that we are trying to solve involves two data streams with heterogeneous feature spaces, and this is the aspect which no researchers have been working on in the past.

This paper proposes a framework, called MultiStream Domain Adaptation (MSDA), to handle the issues described above. The main idea is to find a common feature space for two distinct data streams. By applying the proposed projection algorithm, we try to find a latent feature space that the distributions in original and latent feature space are similar. Meanwhile, the structure of data should be preserved, which means distinct classes are still far apart, and the decision boundaries for different categories should still be preserved. Thus the core problem here is to find a feature space that maximize the similarity between original and projected latent feature space. Main contributions of this paper are as follows:

(1) An embedding-based domain adaptation method is introduced in multistream setting, to address the challenges of adapting source and target domains in a non-stationary environment.
(2) A concept drift detection method is deployed under the new multistream setting with domain adaptation. Under this setting, true labels in the target stream is not required, while only true labels in the source stream is used.

Table 1: Notations

Symbol	Meaning
D_s	Domain of source stream
D_t	Domain of target stream
S	Source stream data
T	Target stream data
X_s, X_t	Feature space of source stream with dimension m and that of target stream with dimension n
Y_s, Y_t	Label space of source and target stream
x_s, x_t	Data instance of source and target stream
y_s, y_t	True and predicted label of a data instance in source stream
W_s, W_t	Projection function to the source and target space
L_s, L_t	Projected data from source and target domain to latent space
P_s, P_t	Probability distribution function of source and target data
N_m	Size of sliding window

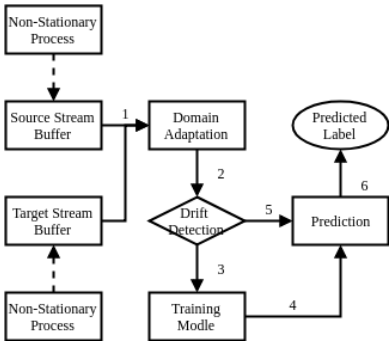


Figure 2: Multistream Classification with Domain Adaptation

(3) Our approach is empirically evaluated over several benchmark datasets, including security and non-security datasets. The results are compared with state-of-the-art methods. The performance of our proposed method is significantly better than the other three methods.

2 PROBLEM FORMULATION

2.1 Notations and Problem Statement

In Table 1, frequently used symbols in this paper are listed. There are two continuous stream of data instances generated from source domain D_s and target domain D_t. A data instance is denoted as (x, y), where x is a vector (m-dimensional in source stream and n-dimensional in target stream), and y is the corresponding class label. In the source stream, both x_s and y_s are available, while in the target domain only x_t is available. Therefore, a multistream classification with domain adaptation problem can be described as follows:

Suppose X_s is a set of m-dimensional vectors and Y_s is the corresponding labels in a source stream from a certain domain D_s, whereas X_t is a set of n-dimensional vectors in target stream from another domain D_t. In our problem setting, $m \neq n$. The objective is to construct a classifier M that predicts class labels of $x_t \in X_t$ using X_s, and Y_s.

Algorithm 1 MSDA Algorithm

Require: Labeled source stream S, Unlabeled target stream T, The size of sliding window N_m. Similarity parameter β.
Ensure: Labels predicted on T.

```
1: /* Initialization */
2: B_s, B_t ← readData(S, T)
3: /* DA for initial buffer */
4: W_s, W_t ← genProjectFunction(B_s, B_t, β)
5: L_s, L_t ← genProjectMatrix(B_s, B_t, W_s, W_t)
6: M ← buildModel(L_s, Y_s)
7: while S or T exists do
8:     B_s, B_t ← readData(S, T)
9:     /* Concept drift detection and correction */
10:    Call ChageDetection /* Algorithm 2 */
11:    if ChangeDetection = True then
12:        /* Update prediction model */
13:        /* DA for stream buffer */
14:        W_s, W_t ← genProjectFunction(B_s, B_t, β)
15:        L_s, L_t ← genProjectMatrix(B_s, B_t, W_s, W_t)
16:        M ← buildModel(L_s, Y_s)
17:    /* Generate predictions */
18:    ŷ_t ← getPrediction(M, L_t)
```

2.2 Challenges

In the problem setting of multistream classification with domain adaptation, there are two major challenges at the same time. The first challenge is the adaptation of both source and target domain, and it can be represented as $D_s \neq D_t$. Two streams may have different number of instances. However, without loss of generality, buffers (windows) from these streams representing contiguous data points will have the same number of instances N_m. Furthermore, each data point in source stream may have a different number of features compared to those in the target stream. Therefore, source data cannot be directly used as training data to learn the target task, and discovering a latent feature space with k dimensions is the key to handling the feature heterogeneity issue. The second challenge is the asynchronous concept drift in both source and target streams. This phenomenon means that the data pattern evolves, or more formally, the conditional probability distribution changes over time in both streams. Here, the problem can be described as $P_s^t(y \mid \mathbf{x}) \neq P_t^t(y \mid \mathbf{x})$ at time t. Furthermore, we assume that source and target data streams have asynchronous concept drifts, which means that the drifts in both streams occur independently.

3 PROPOSED APPROACH

In this paper, we propose a Multistream Domain Adaptation (MSDA) framework to address the major issues in our problem setting. To achieve this goal, we establish a framework with the following modules:

(1) A domain adaptation module that helps find an optimized latent subspace for both source and target streams.
(2) A concept drift detection module that detects concept drift in both source and target streams.

Applying these two modules together, once a concept drift is detected, we use data instances from both streams in the most recent window to update the feature mapping, so that the domain adaptation problem can be addressed. The diagram of this algorithm can be found in Figure 1.

First, a domain adaptation module is triggered to learn projection functions for both source and target data instances. Newly arriving instances are transformed to latent feature representation accordingly (line 8-11). Second, the change point detection module detects if there is a significant change in either source or target streams within the sliding windows Third, once a change point is detected, new classifiers are trained based on these two buffers from B_s and B_t (step 3 & line 12-16). Finally, the newly updated classifiers are used to predict the labels of adapted data instances from target stream T (step 4-6 & line 17-18). Details of our proposed method is as follows.

3.1 Initialization and Domain Adaptation

In our proposed framework, instances from source and target streams are stored in B_s and B_t respectively. Recalling our problem setting, data in B_s have true labels, while data in B_t come without true labels. The domain adaptation method is indeed a optimization problem solved by feature embedding [23]. Also, for computational purposes, we design our approach in a way that buffer size in both source and target streams are the same, which means the numbers of data instances in processing window for source and target streams are identical. Thus, given a certain time t, data that we have are: source stream window matrix $B_s \in \mathbb{R}^{N_m \times m}$; source stream window labels vector $Y_s \in \mathbb{R}^{N_m \times 1}$, and target stream window matrix $B_t \in \mathbb{R}^{N_m \times n}$. In this case, the best projection strategy of L_s and L_t in the latent feature space would be the minimization of following objective function:

$$O = \min_{L_s, L_t} \ell(B_s, L_s) + \ell(B_t, L_t) + \beta \mathbf{D}(L_s, L_t) \tag{1}$$

where $\ell(\cdot, \cdot)$ is a distance function that evaluates the differences between the original data and projected data. $\mathbf{D}(\cdot, \cdot)$ is the co-regularizer that promotes the similarities between the two projected domains (L_s and L_t). β is a parameter that determines how desirable the two projected data are similar. The first two terms are expected to preserve the structure of the original data as much as possible. Therefore, we define the loss function $\ell(\cdot, \cdot)$ as the Frobenius norm, which can also be expressed as a matrix trace norm $\ell(B_s, L_s)$ equals $\|B_s - L_s W_s\|_F^2$, and $\ell(B_t, L_t)$ equals $\|B_t - L_t W_t\|_F^2$.

Furthermore, we define $\mathbf{D}(Ls, L_t)$ as follows:

$$\mathbf{D}(L_s, L_t) = \frac{1}{2}\ell(B_s, L_t) + \frac{1}{2}\ell(B_t, L_s) \tag{2}$$

which is the mean value of cross-similarity between the original data and projected data. Finally, the parameter β controls the trade-off of importance of semantic similarity co-regularization by minimizing the differences between $\mathbf{D}(L_s, L_t)$.

Combining Equation 1 and 2 together, we obtain the overall optimization objective function as follows:

$$\begin{aligned} O = &\|B_s - L_s W_s\|_F^2 + \|B_t - L_t W_t\|_F^2 \\ &+ \frac{1}{2}\beta \|B_s - L_t W_s\|_F^2 + \frac{1}{2}\beta \|B_t - L_s W_t\|_F^2 \end{aligned} \tag{3}$$

Notice here the projection itself will perform rotation and scaling on the target matrix to minimize the difference. Since L_s and L_t are

Algorithm 2 ChangeDetection: Change Detection

Require: Source instances $B_s = \{B_s\}^{i=1}$, target instances $B_t = \{B_t\}^{j=1}$, new instance x, initial mean discrepancy $Disc$, the change parameter τ.

Ensure: $True$ if drift is detected, else $False$.

1: /* Instance is from source stream */
2: **if** $x \in S$ **then**
3: $B_s^{(i)}, L_s^{(i)} \leftarrow B_s^{(i+1)}, L_s^{(i+1)}, i = 1, ..., N_m$.
4: $B_s^{(N_m+1)}, L_s^{(N_m+1)} \leftarrow x, xW_s^+$.
5: $\mu_S = \frac{1}{N_m} \sum_{i=1}^{N_m} L_s^{(i)} W_s^+$.
6: Go to line 11.
7: /* Instance is from target stream */
8: $B_t^{(N_m+1)}, L_t^{(N_m+1)} \leftarrow x, xW_t^+$.
9: $B_t^{(j)}, L_t^{(j)} \leftarrow B_t^{(j+1)}, L_t^{(j+1)}, j = 1, ..., N_m$.
10: $\mu_T = \frac{1}{N_m} \sum_{j=1}^{N_m} L_t^{(j)} W_t^{-1}$.
11: /* Calculate the mean discrepancy $Disc_t$ at time t: */ $Disc_t = \|\mu_s^t - \mu_t^t\|^2$.
12: $s = ln \frac{Disc_t}{Disc_0}$.
13: **Return** $s > -ln(\tau)$.

orthogonal matrices, $B_s^\top B_s = I$. Also, we are applying the cyclic permutation property of trace. Thus, Equation 3 can be expanded by the representation of trace.

After the steps described above, the optimal projection for both source and target stream to the latent feature space would be formed accordingly.

3.2 Classification Module

We start to train a classifier with a small set of data instances from both S and T, which are referred to as the initialization data. The new data representation is obtained by using initialization data from B_s and B_t as follows:

$$\begin{cases} L_s^{(i)} = B_s^{(i)} W_s^+ & B_s^{(i)} \in B_s \\ L_t^{(j)} = B_t^{(j)} W_t^+ & B_t^{(j)} \in B_t \end{cases} \tag{4}$$

where W_s^+ and W_t^+ are the Moore-Penrose inverse of the matrix W_s and W_t respectively, which can be represented as $(W_s^\top W_s)^{-1} W_s^\top$ and $(W_t^\top W_t)^{-1} W_t^\top$.

As new instances arrive in S or T, the classifier is updated if there is a drift to ensure that it represents the current concepts. A new classification model is trained using data in B_s and B_t at that time. Concept drift detection and the updating method used by MSDA will be discussed in the next subsection. MSDA predicts the class label of an incoming target instance from the target stream after projecting the instance into the new data format.

3.3 Change Detection Module (CDM)

Previous work [8] of multistream classification uses prediction confidence to detect changes in distribution between two different time windows. Due to possible asynchronous concept drifts between source and target streams, an ensemble of classifiers is maintained and updated if a concept drift is detected in either of these streams of data. Therefore, complex ensemble algorithms may lead to very slow execution.

Table 2: Datasets

Data type	Datasets	# features	# instances
APT detection	Win	28	10587
	Linux	105	10801
Dark web	BlackHat	38	32414
	Nulled	38	9930
	Darkode	67	100,000
	Hack	67	100,000
Digits	USPS	256	10,000
	MNIST	780	60,000
Amazon review	Video	100	5,000
	DVD	200	30,000
	Music	300	30,000
Synthetic data	Syn01	100	10,000
	Syn02	200	20,000

We adopt the Maximum Mean Discrepancy (MMD) as the distance measure to compare different distributions. The distance between two distributions can be computed between the sample means of the two domains in the k-dimensional embeddings:

$$Disc(L_s, L_t) = \left\| \frac{1}{N_m} \sum_{i=1}^{N_m} L_s^{(i)} W_s^+ - \frac{1}{N_m} \sum_{j=1}^{N_m} L_t^{(j)} W_t^+ \right\|^2 \tag{5}$$

where L_s, L_t are the set of projected data points in the source and target windows.

The difference between the distributions is determined by the log ration between MMD. Therefore, a concept drift point is detected if it is more than a user-defined threshold τ, as follows. And details regarding online updating algorithm for change point detection can be found in Algorithm 2.

$$S = \log \frac{Disc_t(L_s^t, L_t^t)}{Disc_0(L_s^0, L_t^0)} > \tau \tag{6}$$

3.4 Complexity Analysis

In our proposed approach, an update model is trained from source to target stream anytime when a concept drift point is detected. Hence, the time complexity of classification module is significant in determining the overall model complexity. Within the classification model, the training process, which is the updating model determines the complexity of whole algorithm.

MSDA has three modules, Domain Adaptation (DA) module, Change Point Detection (CDM) module, and classification module. The DA module learns projection matrix W_s W_t respectively, and from the instances stored in the source and target sliding window. Since it is a sequence of matrix transformation, the processing time is $O(kN_m)$. The time complexity of CDM module is $O(N_m^2)$. The time complexity of classification depends on the learning algorithm used as the base model. Therefore, MSDA has total time complexity of $O(kN_m + N_m^2)$.

4 EXPERIMENT

4.1 Baseline Methods

To evaluate the effectiveness of our proposed our method MSDA, we compare it with several state-of-the-art domain adaptation or online learning frameworks. Also, we applied two different variations of our method, which are MSDA-SVM and MSDA-LR. These

Table 3: Comparison of performance (Error %)

Data type	Dataset	OTL	HeMap-S	HeMap-L	MSDA-S	MSDA-L
Cyber security	Win → Linux	28.53 ± 0.88	30.55 ± 0.86	29.08 ± 0.58	23.73 ± 0.62	**21.33 ± 1.62**
	BlackHat → Darkode	22.42 ± 0.85	24.70 ± 1.00	25.36 ± 1.00	22.12 ± 1.29	**20.82 ± 0.64**
	Nulled → Hack	28.86 ± 1.54	26.78 ± 0.95	29.26 ± 0.50	25.47 ± 1.06	**24.75 ± 1.36**
Non cyber security	USPS → MNIST	32.31 ± 0.94	38.19 ± 0.75	39.04 ± 0.68	**28.72 ± 0.56**	29.96 ± 0.64
	Music → DVD	33.54 ± 0.69	36.06 ± 0.51	35.85 ± 0.77	32.91 ± 0.49	**31.64 ± 0.57**
	Video → Music	38.01 ± 1.03	40.41 ± 0.65	39.72 ± 0.85	**31.76 ± 0.72**	32.02 ± 0.75
	Video → DVD	35.88 ± 0.88	40.28 ± 0.52	40.09 ± 0.61	**32.35 ± 0.49**	33.84 ± 0.65
Synthetic data	Syn01 → Syn02	37.53 ± 1.37	41.83 ± 0.83	42.12 ± 1.06	**33.50 ± 0.97**	35.47 ± 0.92

two methods are different in a way that classifiers applied are SVM and Logistic Regression. The following paragraphs describe details of baseline methods.

OTL [26]. Online Transfer Learning (OTL) uses a co-regularized method to project target domain data into the source domain. This method is applied in a supervised manner, which means that data in source domain are offline, while data in target domain are online. Furthermore, there is a strong assumption that features in source stream are a subset of those in target stream.

HeMap [23]. Heterogeneous Mapping (HeMap) projects data in two domains with correspondence onto a common latent space. This method is designed as a batch training, which means both source and target data are offline. Also, this method requires class labels in the target dataset. We adapt HeMap into our problem by applying sliding windows. Meanwhile, two classifiers, SVM and LR, are also applied in this method as variations, which are referred as HeMap-S and HeMap-L respectively.

4.2 Datasets

As shown in Table 2, we use both synthetic and real-world datasets to evaluate our methods [4]. Specifically, we incorperate security related datasets as well as non security related datasets here to validate our proposed framework.

Win and **Linux** This dataset consists of system-call events traces from windows host machine and linux host machines. We generated various attack instances which consist of attacks e.g. malicious exfiltration of data, password theft and scanning the system and services on a victim's network with nmap. For benign instances, we generated data which include web browsing activities and remote login using ssh. We collected the system call traces using sysdig trace tool for linux and procmon tool for windows. In total, we collected 10,587 and 10,801 trace instances. With Ngram, we extract features from contiguous sequences of system calls. Our goal is to classify whether a trace in the target stream is an attack.

BlackHat, Nulled, Darkode and **Hack** [21] are datasets extracted from dark web forums where illegal items are listed and sold . Note that all forums here are English forums. Our goal is to extract words that are products on the dark web forums. To satisfy the multistream setting with different domains, for BlackHat and Nulled, we extract the features using Penn Part of Speech Tags [22]. While, for Darkode and Hack datasets, features are extracted following the Stanford pos- tagger system [9]. Since forum posts are posted within a long time period, these streams satisfies the concept drift assumption.

USPS [15] and **MNIST** [18] contain gray-scale images of handwritten digits collected from different sources. In order to satisfy the concept drift assumption in this paper, we shift the concept of positive and negative classes in the middle of datasets. That is, in the first half of both USPS and MNIST, labels 0-4 represent "-" and labels 5-9 represent "+", while in the second half class labels 3-7 mean "+" and the rest class labels mean "-". Therefore, the concept drift assumption of data stream is satisfied in this setting.

Music, **DVD**, and **Video** [5] are text datasets generated by Amazon reviews based on product categories. We process these datasets in the same way as described in [5]. Since data in this stream covers more than one year, we assume that there are concept drift along the change of time by nature in these data streams.

Syn01, **Syn02** [4] are synthetic datasets that are generated by MOA framework. These datasets are generated in a way that the number of features in Syn01 is 100 while that in Syn02 is 200, so that our domain adaptation assumption can be satisfied. Meanwhile, we artificially add abrupt concept drift at the half position in these data streams.

4.3 Experiment Setup

Our MSDA approach involves multiple parameters. We use $N_m = 400$ as our default setting in the experiment. Meanwhile, $\beta = 1$, $k = 4$, and $\tau = 0.1$ are selected to conduct our initial experiment here. The sensitivity of parameters will be discussed later in this section.

4.4 Result Analysis

Figure 3 shows the distribution of features in the latent space. For a single window ($N_m = 400$), we plot the transformed source and target window together in one figure. Also, t-SNE is applied here to reduce the feature space to 2D. From this figure, we can see that binary classes from both source and target stream are well separated in two part of the latent space. This good separation makes it possible to find a hyper-plane in the space between two classes, which gives us confident and evidence to apply classifiers in this feature space.

Table 3 shows the average prediction error on the target stream T: $\frac{A_{wrong}}{m}$, where A_{wrong} is the total number of instances identified incorrectly, and m is the number of instances in the target stream. For security related datasets, for instance, the APT attack detection datasets, our method has 21.33% error rate, which outperforms all other benchmark algorithms. It has a better performance as the projected latent feature space. Also, our proposed framework has the best performance on identifying items on dark web forum datasets. Regarding non security datasets, we can tell that MSDA-S outperforms competing methods on almost every datasets. Also,

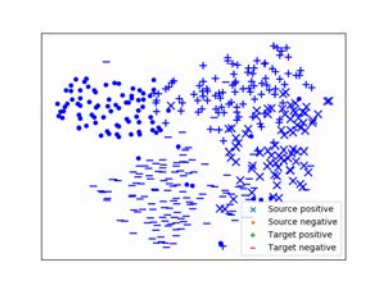


Figure 3: Latent Feature Embedding USPS→MNIST

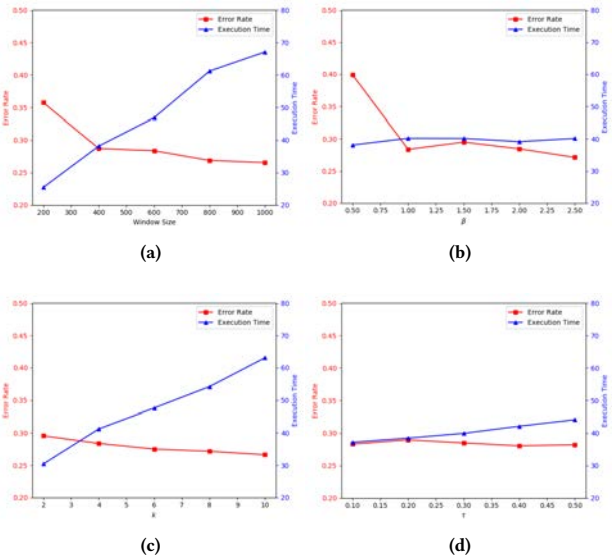


Figure 4: Sensitivity Analysis on USPS→MNIST

due to the fact that digits datasets are indeed very similar (all handwriting digits), which means that the source and target domain are pretty similar in this case, our algorithm shows best performance not only comparing to other benchmark methods, but also among all non security datastes. Furthermore, we can observe that our method works better when adapting from small feature space to larger feature space.

For example, the error rate of OTL, HeMap-S and MSDA-S on USPS → MNIST are 32.31%, 38.19% and 28.72% respectively. We can see that our proposed method has better performance by a significant margin compared to baseline methods. The reason is that for OTL, the algorithm has a strong assumption that features in the source data are a subset of those in the target stream, which is not quite the case here. Not all features in Video would exist in the DVD dataset in our settings. When it comes to HeMap, it also has its own shortcomings. HeMap is an offline method, which makes it not able to adapt the concept drift assumption in our problem. In this dataset specifically, the concept in the second half of data shifts from the first half of data (described in datasets section). Since we applied periodic updates every 4000 instances for the HeMap method, there is a delay on updating the model comparing our proposed MSDA-S approach.

In synthetic datasets, we observe similar phenomenon. Since the MOA framework can help generate concept drifts in the data stream, the two variations of HeMap don't perform well on this dataset. On the other hand, OTL seems to track the concept drift along the stream, however, since the dataset doesn't quite satisfy its assumption, the overall performance of MSDA-S beats other benchmarks by a significant margin.

4.5 Sensitivity Analysis

The results of our proposed method are further analyzed by tuning defined parameters N_m, β, k, and τ. All experiments are conducted on USPS → MNIST dataset. In this section, we vary N_m by setting it to {200, 400, 600, 800, 1000}, β to {0.5, 1, 1.5, 2, 2.5}, k to {2, 4, 6, 8, 10}, and τ to {0.1, 0.2, 0.3, 0.4, 0.5} respectively.

First, we can see in Figure 4 that the average error decreases along with window size increases, while the execution time increases with increasing window size. This is close to our expectation, as we state that the execution time of MSDA depends on N_m. Based on this observation, we should choose a medium value so that both error rate and execution time could be balanced.

Second, β determines the similarities between projected domains from both source and target stream. From Figure 4, we can tell that the error rate decreases significantly when β increases from 0.5 to 1. If $\beta > 1$, the decreasing trend of error rate is becoming not significant. Also, we can see from the figure that the execution time regarding different β remains similar in the experiment. Thus, we choose $\beta = 1$ as our recommended parameter here.

Third, parameter k determines how large the feature space is to apply our classifiers. From Figure 4, we can see that the performance of the model has improved marginally while the execution time increases dramatically as k increases. Thus, we need to choose as small k as possible, meanwhile we need to make sure that performance doesn't degrade. Thus, we choose $k = 4$ for our approach.

At last, we can see that for τ, which controls the threshold for concept drift detection. In this case, the performance of model doesn't quite change when τ varies. Therefore, based on performance of execution time, we need to choose a smaller τ. Hence, we select $\tau = 0.1$ in our model.

In all, sensitivity experiments indicate that MSDA is sensitive to N_m and k, while not quite sensitive to β and τ.

5 CONCLUSIONS

In this paper, a multistream classification framework that incorporates domain adaptation techniques is proposed. Two major challenges, namely heterogeneous domain and concept drift, are addressed simultaneously in two data streams. Our solution involves an embedding-based mapping approach to find a common latent space for both source and target streams, and an online update mechanism using MMD for concept drift correction. Additionally, the prediction model is updated if a concept drift is detected. Extensive experiments with both real-world and synthetic data show that our approach has significantly better performance in terms of error rate on various datasets, compared to existing state-of-the-art solutions.

ACKNOWLEDGMENTS

This material is based upon work supported by NSF award numbers: DMS-1737978, DGE 17236021. OAC-1828467; ARO award number: W911-NF-18-1-0249; IBM faculty award (Research); NSA awards; ONR award N00014-17-1-2995; and a gift from the Eugene McDermott family.

REFERENCES

[1] Khaled Al-Naami, Swarup Chandra, Ahmad Mustafa, Latifur Khan, Zhiqiang Lin, Kevin Hamlen, and Bhavani Thuraisingham. 2016. Adaptive encrypted traffic fingerprinting with bi-directional dependence. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*. ACM, 177–188.

[2] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine Learning* 79, 1-2 (2010), 151–175.

[3] Monowar H Bhuyan, Dhruba Kumar Bhattacharyya, and Jugal K Kalita. 2014. Network anomaly detection: methods, systems and tools. *Ieee communications surveys & tutorials* 16, 1 (2014), 303–336.

[4] Albert Bifet, Geoff Holmes, Richard Kirkby, and Bernhard Pfahringer. 2010. Moa: Massive online analysis. *Journal of Machine Learning Research* 11, May (2010), 1601–1604.

[5] John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*. 440–447.

[6] João BD Cabrera, Lundy Lewis, and Raman K Mehra. 2001. Detection and classification of intrusions and faults using sequences of system calls. *Acm sigmod record* 30, 4 (2001), 25–34.

[7] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 15.

[8] Swarup Chandra, Ahsanul Haque, Latifur Khan, and Charu Aggarwal. 2016. An adaptive framework for multistream classification. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1181–1190.

[9] Marie-Catherine De Marneffe and Christopher D Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: proceedings of the workshop on cross-framework and cross-domain parser evaluation*. Association for Computational Linguistics, 1–8.

[10] Eleazar Eskin, Andrew Arnold, Michael Prerau, Leonid Portnoy, and Sal Stolfo. 2002. A geometric framework for unsupervised anomaly detection. In *Applications of data mining in computer security*. Springer, 77–101.

[11] João Gama, Pedro Medas, Gladys Castillo, and Pedro Pereira Rodrigues. 2004. Learning with Drift Detection. In *Advances in Artificial Intelligence - SBIA 2004, 17th Brazilian Symposium on Artificial Intelligence, São Luis, Maranhão, Brazil, September 29 - October 1, 2004, Proceedings*. 286–295.

[12] Ahsanul Haque, Latifur Khan, and Michael Baron. 2016. SAND: Semi-Supervised Adaptive Novel Class Detection and Classification over Data Stream.. In *AAAI*. 1652–1658.

[13] Ahsanul Haque, Hemeng Tao, Swarup Chandra, Jie Liu, and Latifur Khan. 2018. A Framework for Multistream Regression With Direct Density Ratio Estimation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7*.

[14] Bo-Jian Hou, Lijun Zhang, and Zhi-Hua Zhou. 2017. Learning with Feature Evolvable Streams. In *Advances in Neural Information Processing Systems*. 1417–1427.

[15] Jonathan J. Hull. 1994. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence* 16, 5 (1994), 550–554.

[16] Jungwon Kim, Peter J Bentley, Uwe Aickelin, Julie Greensmith, Gianni Tedesco, and Jamie Twycross. 2007. Immune system approaches to intrusion detection–a review. *Natural computing* 6, 4 (2007), 413–466.

[17] Christopher Kruegel, Darren Mutz, William Robertson, and Fredrik Valeur. 2003. Bayesian event classification for intrusion detection. In *null*. IEEE, 14.

[18] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[19] Mohammad M Masud, Tahseen M Al-Khateeb, Latifur Khan, Charu Aggarwal, Jing Gao, Jiawei Han, and Bhavani Thuraisingham. 2011. Detecting recurring and novel classes in concept-drifting data streams. In *2011 IEEE 11th International Conference on Data Mining*. IEEE, 1176–1181.

[20] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2010), 1345–1359.

[21] Rebecca S Portnoff, Sadia Afroz, Greg Durrett, Jonathan K Kummerfeld, Taylor Berg-Kirkpatrick, Damon McCoy, Kirill Levchenko, and Vern Paxson. 2017. Tools for automated analysis of cybercriminal markets. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 657–666.

[22] Beatrice Santorini. 1990. Part-of-speech tagging guidelines for the Penn Treebank Project (3rd revision). *Technical Reports (CIS)* (1990), 570.

[23] Xiaoxiao Shi, Qi Liu, Wei Fan, S Yu Philip, and Ruixin Zhu. 2010. Transfer learning on heterogenous feature spaces via spectral transformation. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 1049–1054.

[24] Xiaokui Shu, Danfeng Yao, and Naren Ramakrishnan. 2015. Unearthing stealthy program attacks buried in extremely long execution paths. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 401–413.

[25] Bianca Zadrozny. 2004. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*. ACM, 114.

[26] Peilin Zhao and Steven C Hoi. 2010. OTL: A framework of online transfer learning. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 1231–1238.

Predicting pregnancy using large-scale data from a women's health tracking mobile application

Bo Liu[†]
Dept. of Computer Science, Stanford

Shuyang Shi[†]
Dept. of Computer Science, Stanford

Yongshang Wu[†]
Dept. of Computer Science, Stanford

Daniel Thomas
Clue by BioWink GmbH, Berlin

Laura Symul
Dept. of General Surgery and Dept. of Statistics, Stanford

Emma Pierson[*]
Dept. of Computer Science, Stanford

Jure Leskovec
Dept. of Computer Science, Stanford
Chan-Zuckerberg Biohub

ABSTRACT

Predicting pregnancy has been a fundamental problem in women's health for more than 50 years. Previous datasets have been collected via carefully curated medical studies, but the recent growth of women's health tracking mobile apps offers potential for reaching a much broader population. However, the feasibility of predicting pregnancy from mobile health tracking data is unclear. Here we develop four models – a logistic regression model, and 3 LSTM models – to predict a woman's probability of becoming pregnant using data from a women's health tracking app, Clue by BioWink GmbH. Evaluating our models on a dataset of 79 million logs from 65,276 women with ground truth pregnancy test data, we show that our predicted pregnancy probabilities meaningfully stratify women: women in the top 10% of predicted probabilities have a 89% chance of becoming pregnant over 6 menstrual cycles, as compared to a 27% chance for women in the bottom 10%. We develop a technique for extracting *interpretable* time trends from our deep learning models, and show these trends are consistent with previous fertility research. Our findings illustrate the potential that women's health tracking data offers for predicting pregnancy on a broader population; we conclude by discussing the steps needed to fulfill this potential.

KEYWORDS

Pregnancy prediction; mobile health tracking

ACM Reference Format:
Bo Liu, Shuyang Shi, Yongshang Wu, Daniel Thomas, Laura Symul, Emma Pierson, and Jure Leskovec. 2019. Predicting pregnancy using large-scale data from a women's health tracking mobile application. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313512

[*]To whom correspondence should be addressed: emmap1@cs.stanford.edu. [†] indicates co-first authorship.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313512

1 INTRODUCTION

Predicting pregnancy is a fundamental problem in women's health [11, 13, 27, 41]. Modeling the probability of pregnancy, and identifying behaviors which affect it, allows for more efficient family planning. Identifying people with a very low chance of becoming pregnant is important for targeting and treating infertility, which can be a devastating experience [18]. Because of the importance of predicting and facilitating pregnancy, the fertility services market is predicted to grow to $21 billion by 2020 [38].

Attempts to predict pregnancy go back decades, often using Bayesian methods [11, 13, 27, 41] to capture the complex dynamics which govern fertility. Many factors influence fertility, including age [12] and medical conditions [24, 30]. Fertility fluctuates over the course of the menstrual cycle (which averages 28-29 days in length, although this varies across women [8]), peaking during the "fertile window", around 14 days before the end of the cycle. For women with a regular cycle of 28-29 days, the fertile window occurs approximately midway through the cycle [3, 50]. Because the timing of the fertile window varies, and intercourse during the fertile window is much more likely to result in pregnancy, fertility prediction methods often focus on detecting and predicting the fertile window, using ovulation tests and measurements of cervical mucus and basal body temperature [5, 16].

Previous studies of fertility have relied on carefully curated data from medical studies [3, 11, 13, 27, 39–41, 43]: for example, participants are generally asked to regularly monitor critical features like basal body temperature and menstrual cycle starts and to take pregnancy tests after each menstrual cycle. Many people are unwilling or unable to engage in such careful monitoring, limiting the applicability of these methods to the general population.

In the last few years, the growing use of health tracking mobile apps offers a new potential data source for predicting pregnancy in broader populations. Health tracking apps have been recognized as delivering a "data bounty" [19] for healthcare [32, 49]. Within women's health specifically, health tracking apps are used by millions of women worldwide [7, 15, 37], and have already been used to study sexually transmitted infections [2], menstrual cycle fluctuations [35, 47], and menstrual cycle lengths [21]. These apps allow

users to create profiles which include age and birth control information and keep daily logs of symptoms relevant to pregnancy, including basal body temperature, period starts, and ovulation tests. Crucially, they also allow women to log positive and negative pregnancy tests, providing a ground truth source of pregnancy data. Pregnancy data collected by women's health mobile apps differs from that collected in medical studies in three ways:

(1) **Larger datasets.** The dataset we consider in this paper contains tens of millions of observations from tens of thousands of women, orders of magnitude more than previous datasets [14]. Complex models developed for these smaller datasets, which often rely on computationally intensive techniques like MCMC sampling, will not scale to datasets from mobile health tracking apps, necessitating new methods.
(2) **Broader populations.** While previous fertility studies have predominantly focused on filtered, mostly healthy or proven fertile populations that volunteer for research within a single country, health tracking apps can be used by anyone with a smartphone, and are used by women all over the world.
(3) **Missing data.** Critical features like basal body temperature (BBT) and pregnancy tests are less reliably recorded in mobile health tracking datasets than in medical studies, making missing data a pressing concern.

Mobile tracking datasets thus present new challenges but also new opportunities for predicting pregnancy in a much broader population. To this end, a number of companies have announced initiatives to predict pregnancy or fertility using mobile tracking data [1, 28, 33, 42]. To date, however, it is unclear whether their fertility prediction algorithms are reliable, because they are proprietary (so they cannot be compared to or independently assessed) and their efficacy has been disputed [26, 31, 46].

This work. Here we present what is to our knowledge the first study of pregnancy prediction using fully described methods applied to large-scale women's health tracking app data. Using a dataset from a women's health tracking app which includes ground truth data on pregnancy tests, we develop and assess four pregnancy prediction models — an interpretable logistic regression model, and three LSTM models — which integrate traditional fertility modeling methods and modern time series prediction methods. We first show that our models meaningfully stratify women by their probability of becoming pregnant: women in the top 10% of predicted probabilities have a 89% chance of becoming pregnant over 6 menstrual cycles, while women in the bottom 10% have only a 27% chance. We further provide an intuitive technique for extracting *interpretable* time trends from our LSTM models, overcoming a common failing of deep learning models, and show that these time trends are consistent with previous fertility research. Finally, we discuss the steps that should be taken to maximize the efficacy of pregnancy prediction from women's health tracking data.

2 PROBLEM SETUP

Our task is predicting, on the basis of the health tracking data a woman logs during a single menstrual cycle, whether she will log a positive pregnancy test at the end of that cycle. We first describe our dataset and then describe our task in more detail.

2.1 Dataset

We use data from the Clue women's health tracking app [17]. The app has been rated the most accurate menstrual cycle tracking app by gynecologists [29] and previously used in studies which show that it reliably replicates known women's health findings [2, 21, 35]. All data is de-identified and analysis was determined to be exempt from review by the Stanford Institutional Review Board.

Features. Our dataset consists of two types of features:

- **User features.** Users can record age and birth control method (e.g., "None", "Condoms", "IUD", or various types of birth control pill). We encode age using a two-element vector: a binary element which indicates whether age data is missing, and a continuous element which is the value of age if it is present, and 0 otherwise. We encode birth control using a vector with indicator variables for each type of birth control; the vector is all zeros if birth control data is missing.
- **Daily feature logs.** Each row in this dataset consists of one log of one feature for one user on one date. Users can log 110 *binary* features, including positive/negative pregnancy tests (which we use to define ground truth, and do not include in the inputs to our model), period bleeding, mood and behavior features (eg, happy mood or exercise:running), and taking daily birth control. Users can also log 3 *continuous* features: basal body temperature (BBT), resting heart rate, and body weight. We list all daily features in Table 1.

Data filtering. To ensure that users are regularly using the app, we filter for users who enter at least 300 daily feature logs. We apply basic quality control filters to the three continuous features to remove unreliable values (for example, BBT $> 110^{\circ}F$). To minimize the chance that users are already pregnant, we filter out all cycles after a user has logged a positive pregnancy test.

Defining menstrual cycle starts. Because fertility fluctuates over the course of the menstrual cycle, defining menstrual cycle starts – when the period begins – is necessary to predict pregnancy. We define a *cycle start* as a start of bleeding after the user has not recorded any bleeding for at least 7 days. So, for example, if the user records bleeding on May 1, 2, and 29, the cycle starts would be May 1 and May 29. A user's *cycle day* is the number of days since their most recent cycle start, with 0 denoting the day of cycle start.

Encoding continuous features. We encode continuous features (eg, weight) using a two-element vector: a binary element which indicates whether data is present or missing, and a continuous element which is the value for the feature if present, and 0 otherwise. For each cycle and each user, we subtract off the mean for each continuous feature (since the *change* in features like BBT is most indicative of cycle phase and therefore fertility [6]). We include each user's average value of the three continuous features as a user-specific feature.

2.2 Prediction task

Our task is predicting from the first 24 days of a user's cycle whether she will become pregnant in that cycle. We choose the 24-day interval because most women are very unlikely to become pregnant past this interval [50]. Our prediction task is binary, and each example

Table 1: All daily features which appear in our data. Each feature has both a category and a type. Features are binary except for those in the "continuous" category. While pregnancy tests are included in the binary features, we do not use them as predictive features. HBC=hormonal birth control; TNP=type not provided.

Category	Type
Ailment	Allergy, Cold/Flu Ailment, Fever, Injury
Appointment	Date, Doctor, Ob Gyn, Vacation
Collection Method	Menstrual Cup, Pad, Panty Liner, Tampon
Continuous	BBT, Resting Heart Rate, Weight
Craving	Carbs, Chocolate, Salty, Sweet
Digestion	Bloated, Gassy, Great Digestion, Nauseated
Emotion	Happy, PMS, Sad, Sensitive
Energy	Energized, Exhausted, High Energy, Low Energy
Exercise	Biking, Running, Swimming, Yoga
Fluid	Atypical, Creamy, Egg White, Sticky
Hair	Bad, Dry, Good, Oily
Injection HBC	Administered, Type Not Provided
IUD	Inserted, Removed, Thread Checked, TNP
Medication	Antibiotic, Antihistamine, Cold/Flu Medication, Pain
Mental	Calm, Distracted, Focused, Stressed
Motivation	Motivated, Productive, Unmotivated, Unproductive
Pain	Cramps, Headache, Ovulation Pain, Tender Breasts
Party	Big Night Party, Cigarettes, Drinks Party, Hangover
Patch HBC	Removed, Removed Late, Replaced, Replaced Late, TNP
Period	Heavy, Light, Medium, Spotting
Pill HBC	Double, Late, Missed, Taken, TNP
Poop	Constipated, Diarrhea, Great, Normal
Ring HBC	Removed, Removed Late, Replaced, Replaced Late, TNP
Sex	High Sex Drive, Protected, Unprotected, Withdrawal
Skin	Acne, Dry, Good, Oily
Sleep	0-3 Hrs, 3-6 Hrs, 6-9 Hrs, 9 Hrs, TNP
Social	Conflict, Sociable, Supportive, Withdrawn
Test	Ovulation Neg, Ovulation Pos, Pregnancy Neg, Pregnancy Pos

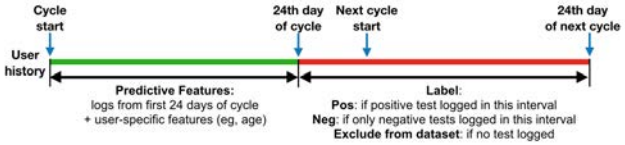


Figure 1: Prediction task. The model makes predictions using logs from the first 24 days of a cycle (green interval), and the cycle is labeled using pregnancy tests taken after day 24 of the cycle and before day 24 of the next cycle (red interval). The vast majority of pregnancy tests in our dataset are taken near when the user's cycle is supposed to start, consistent with proper use of pregnancy tests, so any positive pregnancy tests likely result from activity during the green interval, which will be included in the feature vector.

is one cycle for one woman. We define positive examples as cycles in which the woman logs a positive pregnancy test after day 24 of the cycle and before day 24 of the next cycle (Figure 1); we define negative examples as cycles with a negative pregnancy test and no positive pregnancy test. Thus, our dataset consists only of cycles followed by positive or negative pregnancy tests[1]. Importantly, predicting whether a test will be positive or negative is a difficult and useful prediction task, because it occurs in settings where the woman herself is sufficiently uncertain about whether she is pregnant that she believes taking a test is worthwhile. Our dataset consists of 16,580 positive cycles, 88,685 negative cycles, 65,276 women, and 79,423,281 symptom logs. The proportion of cycles which are positive in our dataset (16%) is consistent with previous work [10]. We use data from 90% of users for training/validation and report results on a test set of the remaining 10% of users. We train on a dataset balanced for positive and negative examples by downsampling each training batch (a standard technique for training on unbalanced data [23]). Following standard practice, we report all results on the unbalanced test dataset.

3 PREVIOUS WORK

Our predictive models rely on ideas both from the fertility modeling literature and from the deep learning time series prediction literature. We now summarize work in both areas.

[1]We define positive and negative examples in this way to mitigate missing data concerns. The alternative (i.e. assuming that any cycle without a positive pregnancy test is a negative example) is problematic for two reasons. First, some users may not bother to log positive tests, so the negative label is a false negative. Second, some users may not bother to log birth control, concealing the fact that they have very little chance of getting pregnant—consistent with this, we find that the fraction of positive examples using this method, even for users who log no birth control, is far lower than the previous literature would imply.

3.1 Fertility modeling

We briefly summarize three main approaches in the extensive fertility modeling literature; [14] provides a lengthier review.

(1) **Time-to-pregnancy (TTP) models** [20, 36, 44, 48] assume each individual menstrual cycle results in pregnancy with some probability μ which can vary across couples and over time. These models are useful for capturing the high level covariates which influence μ (eg, age) but they do not capture detailed daily dynamics within a single cycle (eg, sex during the fertile window is more likely to result in pregnancy). Therefore, they are less useful for modeling the health-tracking datasets we consider, whose strength is their detailed daily information.

(2) **Barrett-Marshall and Schwartz (BMS) models** [3, 11, 13, 39, 43, 45] allow for more detailed modeling of daily activity within each cycle. They assume each act of sex contributes independently to the probability a cycle results in pregnancy. The probability that someone becomes pregnant in a cycle is thus

$$1 - \prod_d (1 - f_d)^{S_d},$$

where d is the cycle day, f_d is the probability of getting pregnant by having sex only on day d, and S_d is a binary variable indicating whether sex occurred on day d. The term inside the product is the probability that day d does *not* result in pregnancy; it is 1 if S_d is 0, indicating that no sex occurred, and $1 - f_d$ if S_d is 1. Schwartz [43] extends this model by assuming that the overall probability of pregnancy also depends on a couple-specific parameter which captures, for example, the fact that some couples are infertile irrespective of sexual activity patterns. However, this additional parameter creates model identifiability problems.

(3) **Extended Time-to-Pregnancy (ETTP) models** [9, 40] provide an approximation to the BMS model by assuming that only sex on the most fertile day contributes to the probability of pregnancy, and sex on other days has no effect. As this model is only a pragmatic approximation to the true generative process, we instead develop a computationally efficient implementation of the BMS model.

All three of the above models were developed for datasets orders of magnitude smaller than the one we consider. Consequently, they perform parameter inference by estimating the posterior distribution, often using computationally intensive techniques like MCMC, and will not scale to our dataset. Therefore, in this work, we extend the BMS model using a scalable deep-learning-based model. Our extension scales to health-tracking datasets because it relies on more efficient backpropagation-based optimization.

3.2 Deep learning for time series prediction

Long Short Term Memory (LSTM) neural networks [22] have been successfully used to model medical time series data [4, 25, 34]. They maintain a hidden state at each timestep which can be used for prediction, and are a natural choice for prediction on time series with discrete timesteps (in our case, days of the menstrual cycle). Because they rely on more scalable backpropagation methods, they also scale to large health-tracking datasets.

4 MODELS

We develop four models. We select model hyperparameters – learning rate, hidden size, number of layers, batch size, dropout rate, and regularization strength – using grid search. Our models are publicly available at https://github.com/AndyYSWoo/pregnancy-prediction.

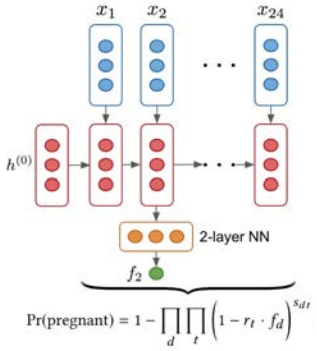


Figure 2: LSTM + BMS fertility model: we feed the daily features x (blue) into an LSTM model to obtain the hidden state h (red). f_d is then a function of the hidden state h parameterized by a neural network. The probability of becoming pregnant in a cycle is the function shown at bottom.

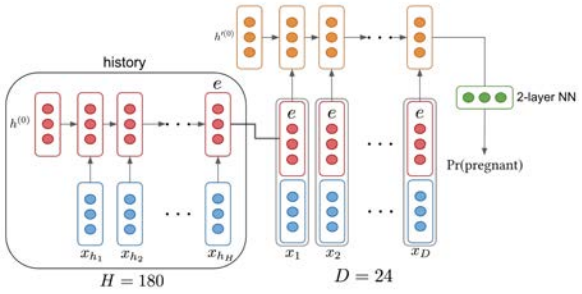


Figure 3: LSTM + user embedding: we use the history features, x_h, in the previous $H = 180$ days for the user and feed them into the "user history LSTM" (shown at left). The final hidden state becomes the user embedding vector e, which is then concatenated with the daily features from the current cycle and fed into the pregnancy prediction LSTM (right).

(1) **Logistic regression**. As an interpretable baseline, we use logistic regression with coefficients for each feature and each cycle day (for example, having unprotected sex on cycle day 14), and coefficients for each user-specific feature. Our feature vector for each example consists of 2,771 features.

(2) **LSTM**. We use an LSTM network with 24 timesteps (one for each cycle day). For each time-step, we feed as input the observed features for that cycle day (binary features plus continuous features, encoded as described in Section 2.1)

along with the cycle averages for continuous features. The final hidden state of the LSTM is concatenated with the user-specific features and fed into a fully connected layer to produce the prediction.

(3) **LSTM + BMS fertility model**. We develop an LSTM-based extension of the BMS model. As described in Section 3.1, the BMS model assumes each act of sex contributes independently to the probability a cycle results in pregnancy. Thus, we model a user's probability of becoming pregnant in a cycle as

$$\Pr(\text{pregnant}) = 1 - \prod_d \prod_t \left(1 - r_t \cdot f_d\right)^{s_{dt}}$$

where d is cycle day, t the type of sex (protected, unprotected, withdrawal, or none[2]), s_{dt} indicates if sex of type t was logged on day d, $f_d \in (0, 1)$ is the contribution of sex on cycle day d towards pregnancy, and $r_t \in (0, 1)$ is a learned parameter capturing the risk of a kind of sex (eg, $r_{\text{unprotected}} > r_{\text{protected}}$). The term inside the product is the probability that an act of sex of type t on cycle day d does *not* result in pregnancy. We model f_d as a function of the daily features x_d using an LSTM, as illustrated in Figure 2.

(4) **LSTM + user embeddings**. Using only the user's current cycle does not account for their full history: eg, a user who has frequently logged unprotected sex for a long time but has not yet become pregnant may be less fertile. To incorporate information prior to the current cycle, we use a "user history LSTM" to encode the 180 days of user history prior to the user's current cycle, then use this LSTM's final state as a user embedding vector which we concatenate onto the other features and feed into a second LSTM as before. We jointly train both LSTMs. Figure 3 illustrates the model architecture.

5 RESULTS

5.1 Predictive performance

Metrics. We evaluate model performance using AUC. Importantly, becoming pregnant is inherently a somewhat random process: it is impossible to *guarantee* a woman will become pregnant in a particular cycle. Consequently, achieving a very high AUC on our task is unlikely to be feasible, and we would not expect AUC to be very high. Nonetheless, we report AUC because it is a standard metric. As a second metric, we compare the probability p_{preg} that a user will log a positive pregnancy test after a cycle for the users in the top 10% of predicted pregnancy probabilities and users in the bottom 10% of predicted probabilities. (Note that p_{preg} is the true, not the predicted, probability of a positive pregnancy test.) However, the goal of fertility counseling is not to guarantee a woman will get pregnant in a particular cycle, but rather that over some reasonable time period (eg, 6 cycles) she has a good chance of becoming pregnant if she continues her current pattern of behavior. Thus, we also compute the 6-cycle probability of becoming pregnant, assuming that each cycle contributes independently to the probability of pregnancy and that woman continues her current pattern of behavior. For example, if a woman's single-cycle p_{preg} is 0.2, her six-cycle p_{preg} is $1 - (1 - 0.2)^6$.

Results. The LSTM with user embeddings has the highest AUC (0.67) (Table 2). This model is able to meaningfully stratify users by their probability of getting pregnant: for example, the top 10% of users have an 89% chance of pregnancy over 6 cycles, whereas the bottom 10% have only a 27% chance. While the other three models have somewhat lower AUCs, all four models usefully stratify users by pregnancy probability. The BMS model slightly worsens LSTM performance, possibly because of the more restrictive assumptions of its probability model – that is, that the overall probability of not getting pregnant in a cycle is the product of the probabilities of not getting pregnant on each cycle day. This demonstrates that traditional fertility models may not yield optimal predictive performance on mobile health datasets.

5.2 Interpretability

We next assess whether our models learn *interpretable* time trends which are consistent with prior fertility research. Because sexual activity is the feature most fundamentally associated with pregnancy, we examine time trends for the three types of sex logged in Clue data: unprotected, protected, and withdrawal sex.

Interpreting the logistic regression model. Extracting time trends from the logistic regression model is straightforward, because the model is specifically developed to be interpretable. To understand how a feature contributes to the probability of pregnancy, we simply plot the coefficients for logging the feature on each day (Figure 4, left). The daily feature which is most strongly associated with positive pregnancy tests (averaging weights across all cycle days) is unprotected sex; the feature which is most strongly associated with negative pregnancy tests is protected sex.

Interpreting the LSTM models. Interpreting deep learning models is much less straightforward, since they are high-dimensional and nonlinear and their coefficients do not have clear meanings. We use the following technique to quantify how logging binary feature b on day d influences the model's inferred probability of pregnancy:

(1) For each example in our test set, we set binary feature b to 1 on day d, and compute the modeled probability of pregnancy for the example, $\Pr(\text{preg}|x_{bd} = 1)$.
(2) For each example in our test set, we set binary feature b to 0 on day d, and compute the modeled probability of pregnancy for the example, $\Pr(\text{preg}|x_{bd} = 0)$.
(3) We compute $\Pr(\text{preg}|x_{bd} = 1) - \Pr(\text{preg}|x_{bd} = 0)$, averaging across examples. This corresponds to the average difference in modeled probabilities when the user does and does not log feature b on day d.

Figure 4 (right) shows results for the LSTM model with the best predictive performance (LSTM + user embeddings); results for the other two LSTM models are qualitatively similar. Like the logistic regression model, the LSTM model learns that unprotected sex (blue) is more positively associated with positive tests than protected sex (green) or withdrawal sex (orange). Unprotected sex during the middle of the cycle shows the strongest positive association.

[2]We allow the model to learn a non-zero probability of getting pregnant even if no sex is logged to account for missing data, where users neglect to log sex but still get pregnant; this significantly improves the BMS model's predictive accuracy.

Table 2: Predictive performance of all models. The third column provides the probability a pregnancy test is positive for users in the top 10% vs bottom 10% of pregnancy risk; the fourth column provides the probability over six cycles.

Model	AUC	Single cycle p_{preg}	Six cycle p_{preg}
Logistic regression	0.63	28% vs 5%	86% vs 26%
LSTM	0.65	30% vs 4%	88% vs 23%
LSTM + BMS	0.64	29% vs 4%	87% vs 22%
LSTM + user embeddings	0.67	30% vs 5%	89% vs 27%

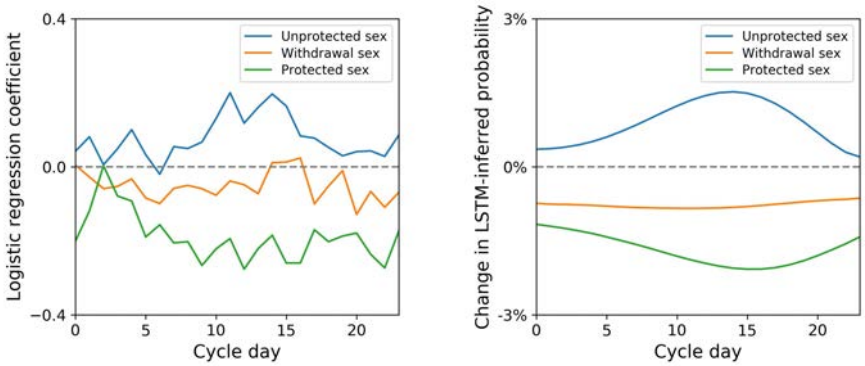


Figure 4: Model-learned time trends are interpretable for both the simple logistic regression model (left plot) and the best-performing LSTM + user embedding model (right plot). The horizontal axis is the cycle day. The vertical axis in the left plot is the logistic regression weight for logging a feature on that cycle day. The vertical axis in the right plot is how much logging a feature on a cycle day affects the LSTM-inferred probability of pregnancy. In both plots, positive y-values indicate associations with positive pregnancy tests, and negative y-values indicate associations with negative pregnancy tests. Both models learn that protected sex (green line) is negatively associated with pregnancy, while unprotected sex (blue line) is positively associated, particularly during the fertile window, and withdrawal sex (orange line) is intermediate.

Consistency with prior research. These modeled time trends are consistent with previous fertility research, which finds that unprotected sex during the "fertile window" [50] is most likely to result in pregnancy. The modeled increase in pregnancy probability due to unprotected sex on a single day is fairly small (<2% for the LSTM model); a plausible explanation for this is that many users are taking birth control. The negative weights for protected and withdrawal sex are of interest because they indicate that the associations learned by the model do not necessarily have causal interpretations. Obviously, having protected sex never reduces the chance of pregnancy in a *causal* sense; rather, it indicates that the woman is trying to avoid becoming pregnant. Overall, this analysis demonstrates that we can extract interpretable, biologically plausible time trends for sexual activity, the feature most fundamentally associated with pregnancy, for both the simple logistic regression model and the predictively superior LSTM models.

6 DISCUSSION

We develop four models for pregnancy prediction, combining ideas from both classical fertility modeling and modern deep learning techniques, and assess them using women's health tracking mobile data. We show we can stratify women by their probability of becoming pregnant and learn interpretable time trends consistent with prior fertility research. We develop a simple, intuitive technique to extract time trends from our LSTM models which is more broadly applicable to other time series datasets.

Based on our study, we recommend two steps to enable women's health tracking apps to reach their full potential for pregnancy prediction. First, companies should fully describe the algorithms used for pregnancy prediction; this will facilitate constructive criticism and increase confidence in the algorithms. Second, incentivize users to provide more complete data. In the Clue dataset, most users do not log pregnancy tests or predictively useful features like BBT; encouraging users to provide this data will improve predictions.

These two improvements can facilitate prediction tasks which build on the one described here. For example, early identification of people who are very unlikely to become pregnant can allow faster referrals to infertility treatment. Modeling the behaviors which *causally* contribute to pregnancy can be used to recommend behavior: for example, having sex during a person-specific fertile window. Progress on these prediction tasks will improve well-being for the millions of people pursuing pregnancy.

Acknowledgments. We thank Paula Hillard, Pang Wei Koh, Chris Olah, Nat Roth, and Camilo Ruiz for helpful comments; Amanda Shea at Clue for data assistance; all Clue users who volunteered their data for research; and the Hertz, NDSEG, and Swiss National Science Foundations for funding. This research was supported in part by the Stanford Data Science Initiative and the Chan Zuckerberg Biohub.

REFERENCES

[1] 2017. Glow and National Institutes of Health Collaborate to Advance Fertility Model. *PR Newswire* (2017).

[2] Alexandra Alvergne, Marija Vlajic Wheeler, and Vedrana Högqvist Tabor. 2018. Do sexually transmitted infections exacerbate negative premenstrual symptoms? Insights from digital health. *Evolution, Medicine, and Public Health* (2018).

[3] John C Barrett and John Marshall. 1969. The risk of conception on different days of the menstrual cycle. *Population Studies* 23, 3 (1969), 455–461.

[4] Inci M Baytas, Cao Xiao, Xi Zhang, Fei Wang, Anil K Jain, and Jiayu Zhou. 2017. Patient subtyping via time-aware LSTM networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 65–74.

[5] Jamie L Bigelow, David B Dunson, Joseph B Stanford, René Ecochard, Christian Gnoth, and Bernardo Colombo. 2004. Mucus observations in the fertile window: a better predictor of conception than timing of intercourse. *Human Reproduction* 19, 4 (2004), 889–892.

[6] Charles L Buxton and William B Atkinson. 1948. Hormonal factors involved in the regulation of basal body temperature during the menstrual cycle and pregnancy. *The Journal of Clinical Endocrinology* 8, 7 (1948), 544–549.

[7] Shraddha Chakradhar. 2018. Discovery cycle. *Nature Medicine* (2018).

[8] Leonard Chiazze, Franklin T Brayer, John J Macisco, Margaret P Parker, and Benedict J Duffy. 1968. The length and variability of the human menstrual cycle. *JAMA* 203, 6 (1968), 377–380.

[9] David Clayton and René Ecochard. 1997. Artificial insemination by donor: discrete time survival data with crossed and nested random effects. In *Proceedings of the First Seattle Symposium in Biostatistics*. Springer, 99–122.

[10] Bernardo Colombo, Guido Masarotto, and Menstrual Cycle Fecundability Study Group. 2000. Daily fecundability: first results from a new data base. *Demographic Research* 3 (2000).

[11] David B Dunson. 2001. Bayesian modeling of the level and duration of fertility in the menstrual cycle. *Biometrics* 57, 4 (2001), 1067–1073.

[12] David B. Dunson, Donna D. Baird, and Bernardo Colombo. 2004. Increased Infertility With Age in Men and Women. *Obstetrics & Gynecology* (2004).

[13] David B Dunson and Joseph B Stanford. 2005. Bayesian inferences on predictors of conception probabilities. *Biometrics* 61, 1 (2005), 126–133.

[14] René Ecochard. 2006. Heterogeneity in fecundability studies: issues and modelling. *Statistical Methods in Medical Research* 15, 2 (2006), 141–160.

[15] Daniel A Epstein, Nicole B Lee, Jennifer H Kang, Elena Agapie, Jessica Schroeder, Laura R Pina, James Fogarty, Julie A Kientz, and Sean Munson. 2017. Examining menstrual tracking to inform the design of personal informatics tools. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 6876–6888.

[16] Richard J Fehring. 2002. Accuracy of the peak day of cervical mucus as a biological marker of fertility. *Contraception* 66, 4 (2002), 231–235.

[17] Biowink GmbH. 2017. Clue by Biowink GmbH. helloclue.com

[18] Arthur L Greil. 1997. Infertility and psychological distress: a critical review of the literature. *Social Science & Medicine* 45, 11 (1997), 1679–1704.

[19] Erika Check Hayden. 2016. Mobile-phone health apps deliver data bounty. *Nature* 531, 7595 (2016), 422–423.

[20] James J Heckman, Richard Robb, and James R Walker. 1990. Testing the mixture of exponentials hypothesis and estimating the mixing distribution by the method of moments. *J. Amer. Statist. Assoc.* 85, 410 (1990), 582–589.

[21] Paula J Adams Hillard and Marija Vlajic Wheeler. 2017. Data from a Menstrual Cycle Tracking App Informs our Knowledge of the Menstrual Cycle in Adolescents and Young Adults. *Journal of Pediatric and Adolescent Gynecology* 30, 2 (2017), 269–270.

[22] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[23] Nathalie Japkowicz. 2000. The class imbalance problem: Significance and strategies. In *Proceedings of the International Conference on Artificial Intelligence*.

[24] Remah M Kamel. 2010. Management of the infertile couple: an evidence- based protocol. *Reproductive Biology and Endocrinology* (2010).

[25] Zachary C Lipton, David C Kale, Charles Elkan, and Randall Wetzel. 2015. Learning to diagnose with LSTM recurrent neural networks. *arXiv preprint arXiv:1511.03677* (2015).

[26] Natasha Lomas. 2018. Clinic reports Natural Cycles app for 37 unwanted pregnancies since September. *TechCrunch* (2018).

[27] Kirsten J Lum, Rajeshwari Sundaram, Germaine M Buck Louis, and Thomas A Louis. 2016. A Bayesian joint model of menstrual cycle length and fecundity. *Biometrics* 72, 1 (2016), 193–203.

[28] Courtney Miller. 2013. How the Period and Fertility Prediction Feature Works. *Kindara Blog* (2013).

[29] Michelle L Moglia, Henry V Nguyen, Kathy Chyjek, Katherine T Chen, and Paula M Castaño. 2016. Evaluation of smartphone menstrual cycle tracking applications using an adapted APPLICATIONS scoring system. *Obstetrics & Gynecology* 127, 6 (2016), 1153–1160.

[30] Merry-K. Moos, Anne L. Dunlop, Brian W. Jack, Lauren Nelson, Dean V. Coonrod, Richard Long, Kim Boggess, and Paula M. Gardiner. 2008. Healthier women, healthier reproductive outcomes: recommendations for the routine care of all women of reproductive age. *American Journal of Obstetrics & Gynecology* (2008).

[31] Katie Morley. 2018. Advert for "Natural Cycles" contraceptive app banned for exaggerating effectiveness. *The Telegraph* (2018).

[32] Catherine J O'shea, Andrew D McGavigan, Robyn A Clark, Derek PB Chew, and Anand Ganesan. 2017. Mobile health: an emerging technology with implications for global internal medicine. *Internal medicine journal* 47, 6 (2017), 616–619.

[33] Arielle Pardes. 2018. In Contraceptive Tech, the App's Guess Is as Good as Yours. *Wired* (2018).

[34] Trang Pham, Truyen Tran, Dinh Phung, and Svetha Venkatesh. 2016. Deepcare: A deep dynamic memory model for predictive medicine. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 30–41.

[35] Emma Pierson, Tim Althoff, and Jure Leskovec. 2018. Modeling Individual Cyclic Variation in Human Behavior. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 107–116.

[36] Robert G Potter. 1960. Length of the observation period as a factor affecting the contraceptive failure rate. *The Milbank Memorial Fund Quarterly* 38, 2 (1960), 140–152.

[37] Roni Caryn Rabin. 2015. How Period Trackers Have Changed Girl Culture. *New York Times* (2015).

[38] Rina Raphael. 2018. Can Silicon Valley Get You Pregnant? *Fast Company* (2018).

[39] J Patrick Royston. 1982. Basal body temperature, ovulation and the risk of conception, with special reference to the lifetimes of sperm and egg. *Biometrics* (1982).

[40] Patrick Royston and Alberto Ferreira. 1999. A new approach to modeling daily probabilities of conception. *Biometrics* 55, 4 (1999), 1005–1013.

[41] Bruno Scarpa and David B Dunson. 2007. Bayesian methods for searching for optimal rules for timing intercourse to achieve pregnancy. *Statistics in Medicine* 26, 9 (2007), 1920–1936.

[42] E Berglund Scherwitzl, O Lundberg, H Kopp Kallner, K Gemzell Danielsson, J Trussell, and R Scherwitzl. 2017. Perfect-use and typical-use Pearl Index of a contraceptive mobile app. *Contraception* 96, 6 (2017), 420–425.

[43] D Schwartz, PDM MacDonald, and V Heuchel. 1980. Fecundability, coital frequency and the viability of ova. *Population Studies* 34, 2 (1980), 397–400.

[44] Mindel C Sheps. 1964. On the time required for conception. *Population Studies* 18, 1 (1964), 85–97.

[45] Joseph B Stanford and David B Dunson. 2007. Effects of sexual intercourse patterns in time to pregnancy studies. *American Journal of Epidemiology* 165, 9 (2007), 1088–1095.

[46] Olivia Sudjic. 2018. "I felt colossally naive": the backlash against the birth control app. *The Guardian* (2018).

[47] Laura Symul, Katarzyna Wac, Paula Hillard, and Marcel Salathe. 2018. Assessment of Menstrual Health Status and Evolution through Mobile Apps for Fertility Awareness. *bioRxiv* (2018).

[48] Sara L Tietze and Richard Lincoln. 1987. Differential Fecundity and Effectiveness of Contraception. In *Fertility Regulation and the Public Health*. Springer, 129–134.

[49] John T Wilbanks and Eric J Topol. 2016. Stop the privatization of health data. *Nature News* 535, 7612 (2016), 345.

[50] Allen J Wilcox, David Dunson, and Donna Day Baird. 2000. The timing of the "fertile window" in the menstrual cycle: day specific estimates from a prospective study. *British Medical Journal* 321, 7271 (2000), 1259–1262.

Fuzzy Multi-task Learning for Hate Speech Type Identification

Han Liu*
School of Computer Science and Informatics, Cardiff University
Cardiff, United Kingdom
liuh48@cardiff.ac.uk

Pete Burnap
School of Computer Science and Informatics, Cardiff University
Cardiff, United Kingdom
burnapp@cardiff.ac.uk

Wafa Alorainy
School of Computer Science and Informatics, Cardiff University
Cardiff, United Kingdom
alorainyws@cardiff.ac.uk

Matthew L. Williams
School of Social Science, Cardiff University
Cardiff, United Kingdom
williamsm7@cardiff.ac.uk

ABSTRACT

In traditional machine learning, classifiers training is typically undertaken in the setting of single-task learning, so the trained classifier can discriminate between different classes. However, this must be based on the assumption that different classes are mutually exclusive. In real applications, the above assumption does not always hold. For example, the same book may belong to multiple subjects. From this point of view, researchers were motivated to formulate multi-label learning problems. In this context, each instance can be assigned multiple labels but the classifiers training is still typically undertaken in the setting of single-task learning. When probabilistic approaches are adopted for classifiers training, multi-task learning can be enabled through transformation of a multi-labelled data set into several binary data sets. The above data transformation could usually result in the class imbalance issue. Without the above data transformation, multi-labelling of data results in an exponential increase of the number of classes, leading to fewer instances for each class and a higher difficulty for identifying each class. In addition, multi-labelling of data is very time consuming and expensive in some application areas, such as hate speech detection. In this paper, we introduce a novel formulation of the hate speech type identification problem in the setting of multi-task learning through our proposed fuzzy ensemble approach. In this setting, single-labelled data can be used for semi-supervised multi-label learning and two new metrics (detection rate and irrelevance rate) are thus proposed to measure more effectively the performance for this kind of learning tasks. We report an experimental study on identification of four types of hate speech, namely: religion, race, disability and sexual orientation. The experimental results show that our proposed fuzzy ensemble approach outperforms other popular probabilistic approaches, with an overall detection rate of 0.93.

*The corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313546

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Applied computing** → **Sociology**.

KEYWORDS

Machine learning; Multi-task learning; Cyberhate detection; Text classification; Fuzzy classification

ACM Reference Format:
Han Liu, Pete Burnap, Wafa Alorainy, and Matthew L. Williams. 2019. Fuzzy Multi-task Learning for Hate Speech Type Identification. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313546

1 INTRODUCTION

Traditional supervised learning typically involves training of a classifier on a single-labelled data set, i.e. each instance is assigned one label only. In this context, each label is treated as a single class and different classes are assumed to be mutually exclusive. However, this assumption does not always hold in real world applications. For example, images can be labelled to show different concepts, different people and different objects. Also, a movie can be included in multiple categories. Therefore, researchers have been motivated to transform the problem to multi-label classification (learning).

In the context of multi-label learning, each instance can be assigned more than one label. Each distinct set of labels *LS* assigned to one or more instances can be generally treated as a single class C_k, such that the training of a classifier can still be undertaken through discriminating one class from the other classes. This strategy is referred to as 'Label Power-set' [1] in the setting of single-task learning, which aims to allow a traditional learning approach to be used for training a single classifier on multi-labelled data [22] as a single task. On the other hand, a multi-labelled data set D can also be transformed into n binary data sets $D_1, D_2, ..., D_n$ (each one D_k per label l_k), such that a traditional learning approach can be used to train a binary classifier h_k on each data set D_k for identifying the corresponding label l_k. This strategy is referred to as 'Binary Relevance' [39] in the setting of multi-task learning and the training of the binary classifiers is treated as multiple tasks [22]. A more detailed review of multi-label learning can be found in [32, 39].

In general, different labels can have specific relationships such as mutual independence and positive correlation (co-occurrence) , if the labels are not mutually exclusive. In this context, if the labels assigned to each instance are mostly independent of each other, the above Label Power-set strategy for handling multi-labelled data could result in an exponential increase of the number of classes and a reduced number of instances for each class [23, 32]. In this case, the complexity of the learning task is much increased leading to a higher risk of overfitting [30]. Also, if the labels assigned to each instance have some positive correlations, the above Binary Relevance strategy through data transformation would not only result in the class imbalance issue but also fail to identify potential correlations between labels [22, 30]. In addition, data labelling is very time consuming and expensive in some application areas such as hate speech detection [2], which indicates a higher degree of infeasibility of multi-labelling of data in these areas.

In order to overcome the above limitations, in this paper, we propose a three-level framework for hate speech detection, which involves identification of the presence or absence of hate speech in level 1, identification of the types of hate speech in level 2 and identification of topics and contexts of hate speech in level 3. However, we focus on level 2, since different types of hate speech could have some intersectionality [9] increasing the complexity of this problem and an appropriate formulation of this problem is needed to achieve effective identification. In particular, we propose a fuzzy ensemble approach for hate speech type identification in the setting of multi-task learning. In this setting, we show how single-labelled data can be used to enable semi-supervised multi-label learning through a new problem formulation and two new metrics are also defined for effective evaluation of the performance for this kind of learning tasks. An experimental study is reported to show that the proposed fuzzy approach is more suitable and effective for this kind of learning tasks, comparing with the popular approaches.

The rest of this paper is organized as follows: Section 2 provides an overview of cyberhate research. In Section 3, we introduce the proposed three-level framework in general and focus on illustration and justification of the proposed fuzzy approach for semi-supervised multi-label learning. In Section 4, we provide the general description of the data used for this experimental study and show the results for evaluation of the proposed approach in comparison with the state of the art ones. In Section 5, the contributions of this paper are summarized and further directions are suggested towards further advances.

2 RELATED WORK

Since the spread of online hate speech could lead to disruptive anti-social outcomes, cyberhate has thus been considered as a legal issue in many countries [5]. In particular, many European countries have already initiated legal actions to prevent online hate speech to be posted [6]. However, it has been complicated to take such actions since the World Wide Web is naturally borderless [25]. Also, due to different laws from different countries, it has become more difficult to prosecute the senders of online hate speech and this even results in the lack of power for removing any hateful contents posted from a location outside their territory [5]. Outside of legal procedures, social network providers, such as Facebook and Twitter, have also been responsible to take any necessary steps towards restricting online hate speech significantly [4]. This has motivated researchers to develop automatic tools for hate speech detection, such that the post of hateful contents can be effectively prevented [2]. In particular, machine learning has become a very popular tool for automatic detection of hate speech [8, 35, 40].

In the context of machine learning based cyberhate detection, some traditional learning algorithms, such as Support Vector Machines (SVM) [9], Naive Bayes (NB) [15, 29], Logistic Regression (LR) [33, 35] and Random Forests (RF) [8], have also been used in the previous studies. A pragmatic approach was proposed in [34] for detecting hateful and offensive expressions, based on unigrams and automatically collected patterns for training classifiers by using SVM, DT and RF. Also, some other state-of-the-art methods of feature extraction, such as N-grams (NG) [14] and Typed Dependencies (TD) [8, 9] and Text Embedding [26], have been used to capture hateful characteristics. In general, text embedding shows its advantages (in terms of reduction of the dimensionality and the sparsity) and better performance than the other methods, while the three learning algorithms SVM, NB and DT generally show their better suitability for training classifiers.

In recent years, deep learning methods have also been used for both feature extraction and training of classifiers. In particular, Gamback and Sikdar have recently used Convolutional Neural Networks (CNN) in [12] for classifying hate speech using different types of features and they showed that the use of embedding features extracted through word2vec led to the best performance on a data set that involves multiple classes [33]. In [3], multiple deep neural network (DNN) architectures, such as CNN and Long Short Term Memory (LSTM) Networks, were compared using the same data set. The DNN architectures were adopted to learn semantic word embeddings as features for training classifiers. The experimental results reported in [3] show that the use of embedding features led to better classification performance than the use of features extracted through BOW or NG. In terms of training classifiers on the above embedding features, the results show that GBT outperformed DNNs and traditional learning methods (SVM and LR).

A two-step CNN based hybrid approach was proposed in [27] for detecting racist and sexist speech using the same data set as [33]. In particular, the first step is aimed at detection of abusive language, whereas the second step is aimed at identification of the abusive language type (racist and sexist). In comparison of the two-step approach with the one step approach, the experimental results reported in [27] show that the use of hybrid CNN led to the best performance through the one-step approach and the use of LR led to the best performance through the two-step approach, where the one-step approach performed marginally better than the two-step approach. In [40], a gated recurrent unit layer was incorporated into CNN by locating this added layer after the pooling layer and a drop out layer was used for optimaizing the training of embeddings. The experimental results show that the proposed approach led to advances in the classification performance on 6 out of 7 data sets, in comparison with previous baseline results.

On the basis of the above review, we consider text embedding as the state of the art method of feature extraction. Moreover, probabilistic learning approaches including DT, NB, SVM, GBT and DNNs are considered as the state of the art ones for classifiers training.

3 PROPOSED FRAMEWORK OF CYBERHATE DETECTION

In this section, we present the proposed three-level framework for cyberhate detection, which involves polarity classification (hate or non-hate) in level 1, multi-task classification for identification of the hate speech types in level 2 and detection of topics and contexts of hate speech in level 3. Then we focus on level 2 for the problem formulation and the illustration of the procedure of the proposed fuzzy ensemble approach in the setting of multi-task learning.

The proposed framework essentially aims at different levels of abstraction for hate speech detection tasks and reduction of the complexity of a single task in a single level. There has been a plenty of works done on general detection of hate speech as a task involved in level 1, as reviewed in Section 2. However, to the best of our knowledge, very few works have been focused on identifying the presence or absence of multiple types of hate speech as a task involved in level 2. Different types of hate speech have potential intersectionality [9], which indicates that the hate speech type identification task can not be simply formulated as a traditional multi-class classification problem based on the assumption of mutual exclusion of different classes. In other words, multi-label classification would be a more appropriate formulation.

Multi-label classification is typically achieved through training classifiers in the setting of supervised learning. However, mutlilabelling of hate speech data is practically much less feasible, as stressed in Section 1. Therefore, we will introduce a new formulation of the problem in Section 3.1 to enable multi-label classification in the setting of semi-supervised learning and propose a fuzzy ensemble approach for hate speech type identification in the setting of mutli-task learning in Section 3.2.

3.1 Problem Formulation

In general, a single-labelled data set D involving n labels could be produced by taking the positive instances from n different binary data sets $D_1, D_2, ..., D_n$. In this way, each set of positive instances is associated with one of the n labels and the n sets of positive instances are merged into a new single-labelled data set D.

In the setting of semi-supervised multi-label learning, the problem is formulated as n Positive-Unlabelled (PU) learning sub-problems (Eqs. (1) and (2)), respectively, for identifying n labels (types of hate speech), similar to the Binary Relevance based problem transformation. In particular, for each single-labelled instance, it is considered as an unknown case that the instance is relevant or not to the other $n-1$ labels, i.e. it is treated as unlabelled regarding other labels.

$$h(\cdot) = \bigwedge_{j=1}^{n} \{h_j(\cdot)\} \tag{1}$$

$$h_j(\cdot) = \begin{cases} +1, & \text{if } h_j(\cdot) = l_j; \\ -1, & \text{otherwise.} \end{cases} \tag{2}$$

In the above context, for each predefined label $l_j \in LS$, instances that are assigned this label l_j are considered to be a set of positive ones, whereas the other instances are considered to be a set of unlabelled ones regarding the label l_j (i.e. a mixed set of potentially both positive and negative instances). The above two sets of instances are used for an independent binary classification task for each label l_j in the setting of PU learning [13, 16–18], which has been popularly adopted for semi-supervised single-label classification.

PU learning is a special type of semi-supervised learning, which does not involve labelled negative instances in the training set and can be achieved through three strategies as surveyed in [38]:

(1) The first strategy involves a two-step operation by identifying reliable negative instances from the unlabelled set and then adopting supervised learning on the positive instances and the reliable negative instances.
(2) The second strategy involves weighting the positive and unlabelled instances and then estimating the conditional probability of the positive class given the input vector of an instance.
(3) The third strategy involves simply treating the unlabelled training instances as highly noisy negative ones, assuming that the majority of the unlabelled instances belong to the negative class.

In the context of hate speech type identification, we take the third strategy since the positive (target) class is typically the minority one. For each binary classifier h_j trained in the setting of PU learning, the output +1 shown in Eq. (2) indicates the case that the target type l_j of hate speech is detected from an instance x_i. In contrast, the other output -1 indicates the case that the binary classifier h_j fails to detect the target type l_j of hate speech.

For adopting probabilistic approaches, it is not feasible to achieve the above setting of semi-supervised multi-label learning without transformation of the original single-labelled data. In particular, a single-labelled data set needs to be transformed into n binary data sets in the setting of Binary Relevance, i.e. from each binary data set D_j, the instances assigned label l_j are treated to be the positive ones and the other instances are treated as unlabelled ones, so a binary classifier h_j is trained to identify the label l_j.

In contrast, for adopting the proposed fuzzy approach, the above data transformation is not needed, since it is an instance based generative approach of fuzzy rule learning, which aims at considering iteratively whether each instance $x_i \in l_j$ is covered by an existing rule of the label l_j or another label $l_g \neq l_j$. More details on the fuzzy approach are shortly given in Section 3.2.

In the classification stage, it is possible that more than one binary classifier gives a positive output (i.e. a label $l_j \in LS$ rather than its negation $\neg l_j$) for the same instance x_i, so the instance is finally assigned multiple labels. If the target label l_t is originally assigned to the instance x_i as the ground truth and the binary classifier h_t successfully identifies that the instance x_i belongs to the label l_t, then it would be judged as successful detection of the target type l_t of hate speech from the instance x_i, regardless of the correctness of assigning other labels to the instance x_i. Also, it is possible that none of the binary classifiers provides a positive output, which would be simply identified as the case that the binary classifier h_t fails to detect the presence of the target type l_t of hate speech and none of the other types of hate speech is detected.

In the new problem formulation shown in Eqs. (1) and (2), two new metrics are proposed, which are referred to as 'detection rate' (Eq. (3)) and 'irrelevance rate' (Eq. (4)), respectively. The two terms given for the two metrics are inspired from two possible cases in information retrieval tasks, where the retrieved documents may be the target ones successfully detected or the irrelevant ones incorrectly identified. In Eq. (3), $|l_j|$ represents the number of instances

that are annotated the label l_j and $|l_j \cap \hat{l_j}|$ is the number of instances that are annotated the label l_j by people and are assigned either a single label l_j or multiple labels including l_j by classifiers. In Eq. (4), $|\hat{l_j}|$ is the number of instances that are assigned the label l_j by classifier h_j and $|\neg l_j|$ represents the number of instances that are annotated as $\neg l_j$ (i.e. the number of negative instances).

$$P = \frac{|l_j \cap \hat{l_j}|}{|l_j|} \tag{3}$$

$$R = \frac{|\hat{l_j}|}{|\neg l_j|} \tag{4}$$

In the context of hate speech types identification, non-hate speech instances should be used as negative ones (belonging to $\neg l_t$), since single-labelled hate speech instances (annotated label l_t) are not provided with ground truth on the absence of each other type $l_j \neq l_t$ of hate speech and thus cannot be used as negative instances for evaluating the overall confidence of relevantly assigning an instance $x_i \in l_t$ a label $l_j \neq l_t$ in addition to the label l_t.

3.2 Procedure of Multi-task Learning

The proposed multi-task learning approach is essentially based on the mixed fuzzy rule formation algorithm [7] with modifications, and the procedure of this algorithm is illustrated as below:

It involves a sequential and constructive generation of new rules and modification of existing rules in an instance-by-instance manner, i.e. each instance is checked, and a new rule is added into the rule set or some existing rules are modified. For each class label l_j, a subset of fuzzy rules is trained in a single task and the rule subset is treated as a binary classifier h_j trained for identifying the label l_j. Therefore, the training of the whole set of fuzzy rules for all class labels is undertaken in a multi-task learning manner.

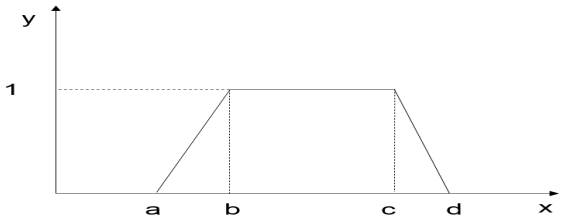


Figure 1: Trapezoidal Membership Function [20]

In the whole procedure, each rule r_t involves n membership functions (for n rule antecedents) and two additional parameters w and λ to be defined, where w represents the number of instances covered by rule r_t and λ is a so-called anchor that remembers the original instance triggering the generation of this rule r_t. A membership function generally involves four parameters a, b, c, d as shown in Fig. 1, where the interval $[b, c]$ represents a core region showing hard boundaries for an element to fully belong to a set and the rest represents a support region (a, b) or (c, d) showing soft boundaries for an element to partially belong to a set.

At each of a number of epochs, once each instance $x_i \in l_j$ is checked, one of the following cases needs to be identified:

- Covered: if an instance $x_i \in l_j$ is covered by a rule r_t, then the membership function defined for each rule antecedent needs to be adjusted to let the instance x_i obtain a membership degree of 1 to the rule r_t [7, 11].
- Commit: if the instance $x_i \in l_j$ is not covered by any rules, a new rule is generated and a membership function is initialized for each rule antecedent [7, 11].
- Shrink: if an instance $x_i \in l_j$ is covered by a rule r_t of label $l_g \neq l_j$, then conflict avoidance should be taken using some shrink heuristics [11].

Following the above procedure, a set of fuzzy rules is trained, which involves n subsets of rules for the n labels. The presence or absence of each target type of hate speech can be identified from each new instance through fuzzification, inference and defuzzification. The fuzzification operation is simply to map the value v_{ik} of each feature f_k of an instance x_i to a membership degree $\mu_{A_{tk}}(v_{ik})$ to a fuzzy set A_{tk} defined in a rule antecedent for f_k. The inference operation is adopted to compute the firing strength of each rule r_t by using a T-norm (e.g. the min function shown in Eq. (5)) and to derive the overall membership degree of the instance x_i for each label l_j (the presence degree of each type of hate speech) by using one of the T-conorms [11], e.g. the max function shown in Eq. (6).

$$T(\mu_{A_{t1}}(v_{i1}), \mu_{A_{t2}}(v_{i2}), ..., \mu_{A_{td}}(v_{id})) = \min_{k=1}^{d}\{\mu_{A_{tk}}(v_{ik})\} \tag{5}$$

$$S(\mu_{r_1}(x_i), \mu_{r_2}(x_i), ..., \mu_{r_q}(x_i)) = \max_{t=1}^{q}\{\mu_{r_t}(x_i)\} \tag{6}$$

It is proposed to use multiple fuzzy norms (dual pairs of T-norms and T-conorms) for training multiple classifiers. Fuzzification and inference are operated independently using each single classifier. However, the defuzzziciation operation is modified to suit identifying the presence or absence of each hate speech type from a new instance x_i according to a threshold (normally 0.5) of the overall membership degree for each label l_j, i.e. Eq. (7) shows that defuzzification is operated by fusing the m fuzzy classifiers through taking the maximum of the membership degrees $\{\mu_{l_j}^{h_1}(x_i), \mu_{l_j}^{h_2}(x_i), ..., \mu_{l_j}^{h_m}(x_i)\}$ computed through classifiers $\{h_1, h_2, ..., h_m\}$ for each label l_j.

$$\mu_{l_j}^{Ensemble}(x_i) = \max_{f=1}^{m}\{\mu_{l_j}^{h_f}(x_i)\} \tag{7}$$

$$\mu_{l_j}^{Ensemble}(x_i) = \frac{1}{m}\sum_{f=1}^{m}\{\mu_{l_j}^{h_f}(x_i)\} \tag{8}$$

The maximum fusion rule (Eq. (7)) is adopted instead of others such as the mean rule (Eq. (8)), in order to minimize the risk of missing a target type of hate speech to be detected from a tweet, while the type of hate speech is actually present. In other words, if one classifier in a fuzzy ensemble identifies that the membership degree for the target label (hate speech type) is high enough, the fusion of multiple classifiers in the ensemble would make sure the fused membership degree for the target label is high enough. In particular, if the fused membership degree $\mu_{l_j}^{Ensemble}(x_i)$ of instance x_i for a label l_j is greater than the predefined threshold, it would be judged as the case that the target type l_j of hate speech is detected successfully from the instance x_i.

4 EXPERIMENTAL STUDY

In this section, we report an experimental study on identification of hate speech types using four data sets collected for four target types of hate speech, namely: religion, race, disability and sexual orientation. The proposed fuzzy ensemble approach is evaluated by comparing with the state of the art probabilistic approaches such as SVM and DNNs, alongside the use of embedding features. The details on the data sets are described in Section 4.1 and the experimental setup and results are presented in Section 4.2.

4.1 Data

The data sets were collected from Twitter for a period immediately following selected 'trigger' events, which were: for religion, the attack on Lee Rigby in Woolwich, London on 22 May 2013 by Islamist Extremists; for race, the presidential re-election of Barack Obama starting November 6th 2012; for disability, the opening ceremony of the Paralympic games in London, UK on 29th August 2012; and for sexual orientation, the public announcement by Jason Collins on 30th April 2013 - the first active athlete in an American professional sports team to come out as gay. Each event produced datasets between 300,000 and 1.2 million, from which we randomly sampled 2,000 to be human coded. Coders were provided with each tweet and the question: 'is this text offensive or antagonistic in terms of religion/race/sexual orientation/disability?' They were presented with a ternary set of classes - yes, no, undecided.

The results of the annotation exercise produced four 'gold standard' data sets as follows: Religion - 1,901 tweets, with 222 instances of offensive or antagonistic content (11.68% of the annotated sample); Race - 1,876 tweets, with 70 instances of offensive or antagonistic content (3.73% of the annotated sample); Disability - 1,914 tweets, with 51 instances of offensive or antagonistic content (2.66% of the annotated sample); and Sexual Orientation - 1,803 tweets, with 183 instances of offensive or antagonistic content (10.15% of the annotated sample). More details on the data collection and annotation have been presented in [8, 9, 19].

4.2 Experimental Setup and Results

The experiments were conducted by taking the four types of hate speech instances, respectively, from the four single-labelled data sets, to form a training set for evaluating the detection rate of the presence of each hate speech type, through 10-fold cross validation. All the non-hate speech instances were taken from the four data sets to form a holdout test set for evaluating the irrelevance rate on each hate speech type, using the classifiers built on the training set.

For non-DNN based learning methods, all the tweets were pre-processed by converting the words to their lower case, and removing stop words, numbers, punctuation and words that contain less than 3 characters. The pre-processed tweets were then sent for embedding training. Furthermore, embedding features were prepared through adopting distributed bag of words (DBOW). In particular, the learning rate was set to 0.025 with the context window size of 2. Following the embedding learning with the batch size of 10000 for 45 epochs, all the words (appearing at least twice in the corpus) were used for embedding learning to transform each tweet into a document vector with 100 dimensions.

The embedding features were then sent to the learning algorithms for classifiers training. In particular, DT classifiers were trained by using the C4.5 algorithm [28] without pruning. SVM classifiers were trained with using the linear kernel [10]. For GBT training, all the attributes were used and attribute selection for each node of a tree was done by using the same set of attributes. In the setting of the proposed fuzzy ensemble approach, we selected Min/Max norm [37], Product norm [11], Lukasiewicz norm [24] and Yager norm [36], respectively, alongside the border based shrink heuristic as the parameters, for training fuzzy classifiers. The trained fuzzy classifiers were then fused through the maximum rule (Eq. (7)). For defuzzification, a membership degree $\mu_{l_j}(x_i) > 0.5$ is used for identifying the presence of a type of hate speech.

For DNN methods, embedding features were prepared according to the settings of CNN and LSTM reported in [12] and [3], respectively. Based on the prepared embedding features, the classifiers were trained through 2 fully connected layers with 100 units in each layer. The rectified linear unit (ReLU) activation function and the mean squared error (MSE) loss function were used with the learning rate of 0.01, and Stochastic Gradient Descent was used for optimizing the parameters over 20 epochs with the batch size of 10.

The results (rounded to 2 decimal places) on detection rate and irrelevance rate are shown in Tables 1 and 2, respectively. In particular, Table 1 shows that the proposed fuzzy ensemble approach performs the best in terms of the overall detection rate and the rate on the detection of the religion hate speech, whereas it performs the same as or marginally worse than SVM on the detection of the other types of hate speech.

Table 1: Detection rate on four types of hate speech

Learning Method	Overall	Religion	Race	Disability	Sexual Orientation
DT	0.84	0.86	0.70	0.75	0.90
NB	0.63	0.57	0.43	0.73	0.75
SVM	0.92	0.92	**0.84**	**0.86**	0.97
GBT	0.90	0.93	0.80	0.82	0.92
CNN	0.35	0.00	0.00	0.00	**1.00**
LSTM	0.33	0.00	0.00	0.00	0.95
Fuzzy	**0.93**	**0.96**	0.81	**0.86**	0.95

Table 2: Irrelevance rate on four types of hate speech

Learning Method	Overall	Religion	Race	Disability	Sexual Orientation
DT	0.57	0.76	0.58	0.19	0.21
NB	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
SVM	0.37	0.25	0.29	0.20	0.53
GBT	0.54	0.13	0.77	0.12	0.26
CNN	1.00	**0.00**	**0.00**	**0.00**	1.00
LSTM	1.00	**0.00**	**0.00**	**0.00**	1.00
Fuzzy	0.13	0.15	0.00	0.00	0.02

On the other hand, Table 2 shows that the proposed fuzzy ensemble approach outperforms all the other probabilistic approaches, except for NB, in terms of the overall irrelevance rate. In particular, the fuzzy approach performs with an extremely low irrelevance rate on identification of the presence of the race, disability and sexual hate speech (note: 2 non-hate speech instances identified irrelevantly as the race type and 1 identified irrelevantly as the

disability type leading to the rounded irrelevance rate of 0 for both hate speech types) and a fairly low irrelevance rate on identification of religion hate speech (although the rate is considerably higher than the ones on the other types of hate speech).

Although the fuzzy ensemble approach shows a considerably higher overall irrelevance rate than NB, the detection rate obtained using the fuzzy approach is much better than the one obtained using NB. In general, a low detection rate alongside a low irrelevance rate would indicate that the classifier has a strong tendency of not outputting the target class label. In this situation, the irrelevance rate can be extremely low (even equal to 0), since the classifier rarely or even never outputs the target class label.

Table 3: Correlation analysis between different types of hate speech

Label	Religion	Race	Disability	Sexual orientation
Religion	1	-0.221	-0.168	-0.524
Race	-0.221	1	-0.165	-0.221
Disability	-0.168	-0.165	1	-0.237
Sexual Orientation	-0.524	-0.221	-0.237	1

On the basis of the above discussions, the overall performance of the proposed fuzzy ensemble approach is better than the ones of the other probabilistic approaches. In addition, the fuzzy approach can also be used for effectively identifying the label correlations, where the probabilistic approaches fall short [31, 39]. The results on correlation analysis between different labels are shown in Table 3, which indicates the correlation coefficient for each pair of class labels in terms of the membership degrees.

In general, there are three extreme cases, namely, 'mutually exclusive' (the most strongly negative correlation), 'independent' and 'identical' (the most strongly positive correlation), when the correlation coefficients are -1, 0 and +1, respectively. The results shown in Table 3 indicate that the four types of hate speech generally involve very weakly negative correlations. The results on correlation analysis between different types of hate speech show supporting evidence that it is much more necessary to formulate the problem as multi-label classification instead of single-label one. In order to show the correlation analysis in more depth, we conduct an extended study on topic detection using the Latent Dirichlet Allocation (LDA) algorithm. The results are shown in Table 4.

In general, the results show high diversity of hate speech instances, i.e. diverse topics involved in hate speech instances. Some common topics can be found from different hate speech data sets. For example, both religion and race hate speech data involve the topics on black and white people. Also, the topics on christian can be found from both religion and sexual orientation hate speech data. In addition, sports related topics are involved in both disability and sexual orientation hate speech data.

The results on topic detection generally indicate that hate speech instances are highly diverse, leading to a high likelihood that a single tweet could be related to more than one type of hate speech, although the class labels may not have strongly positive correlations. In this case, the proposed fuzzy approach is even more suitable for dealing with the diversity, since it is essentially an instance based approach of fuzzy rule learning. In other words, the generation of each fuzzy rule is triggered by an instance without direct discrimination between different classes.

Table 4: Topics detected from hate speech instances in four data sets using LDA

Dataset	Topic ID	Word Cluster
Religion	0	nigger, paki, black, guy, incident
	1	black, people, shot, killed, white
	2	black, edl, nigga, home, send
	3	niggers, nigga, shame, kill, fuck
	4	black, niggas, paki, nigger, islam
Race	0	nigga, niggas, america, shit, won
	1	black, voted, people, tweets, cuz
	2	white, black, romney, won, gotta
	3	niggas, yall, president, hate, nigga
	4	nigga, real, mitt, house, white
Disability	0	irony, leg, called, lickmynippless, mybad
	1	dont, swimmer, ill, understand, drunk
	2	han, labradors, cancelled, highjumps, blind
	3	moving, arsed, wish, falling, women
	4	wheelchair, team, day, events, frankieboyle
Sexual Orientation	0	gay, chris, broussard, http, dick
	1	gay, hes, nba, call, ball
	2	gay, aint, niggas, pretty, espn
	3	nba, faggot, hes, gay, coming
	4	media, tebow, tim, christian, yourself

5 CONCLUSIONS

In this paper, we have proposed a theoretical framework that involves fuzzy mutli-task learning for hate speech types detection. In particular, a novel formulation of the problem has been introduced in the setting of semi-supervised multi-label learning from single-labelled data. An experimental study has been reported to evaluate the performance of the proposed fuzzy ensemble approach using the two new metrics referred to as 'detection rate' and 'irrelevance rate', respectively. The experimental results show that the proposed fuzzy approach outperforms the state of the art probabilistic approaches such as SVM and DNNs on embedding features. Also, the proposed fuzzy approach provides an intensity score for the presence of each type of hate speech from a tweet and enables the analysis of correlation between different labels, while the probabilistic approaches fall short in this aspect.

In the future, we will collect more diverse types of hate speech instances to increase the number of labels, and investigate the impacts of the increased number of labels on the detection performance of each type of hate speech as well as the intersectionality between different types. Also, an extension of the currently proposed fuzzy approach to involve multiple iterations of semi-supervised multi-labelling would be another interesting further direction in the setting of self-labelling based semi-supervised learning [21], i.e. the original single-labelled data becomes multi-labelled for further training after the first iteration of training.

ACKNOWLEDGMENTS

This paper is supported partially by Economic and Social Research Council (Grant Number: ES/P010695/1) and partially by the RAND Corporation (Award Number: 2016-MU-MU-0009).

REFERENCES

[1] Zahra Ahmadi and Stefan Kramer. 2018. A label compression method for online multi-label classification. *Pattern Recognition Letters* 111 (2018), 64–71.

[2] Wafa Alorainy, Pete Burnap, Han Liu, Amir Javed, and Matthew L. Williams. 2018. Suspended Accounts: A Source of Tweets with Disgust and Anger Emotions for Augmenting Hate Speech Data Sample. In *International Conference on Machine Learning and Cynernetics*. IEEE, Chengdu, China.

[3] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep Learning for Hate Speech Detection in Tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*. ACM, Perth, Australia, 759–760.

[4] Chara Bakalis. 2018. Rethinking cyberhate laws. *Information & Communications Technology Law*, 27, 1 (2018), 86–110.

[5] James Banks. 2010. Regulating hate speech online. *International Review of Law, Computers and Technology* 24, 3 (2010), 233–239.

[6] James Banks. 2011. European regulation of cross-border hate speech in cyberspace: The limits of legislation. *European Journal of Crime, Criminal Law and Criminal Justice* 19, 1 (2011), 1–13.

[7] Michael R. Berthold. 2003. Mixed Fuzzy Rule Formation. *International Journal of Approximate Reasoning* 32 (2003), 67–84.

[8] Pete Burnap and Matthew Williams. 2015. Cyber Hate Speech on Twitter: An Application of Machine Classification and Statistical Modeling for Policy and Decision Making. *Policy and Internet* 7, 2 (2015), 223–242.

[9] Pete Burnap and Matthew Williams. 2016. Us and them: identifying cyber hate on Twitter across multiple protected characteristics. *EPJ Data Science* 5, 11 (2016).

[10] Nello Cristianini. 2000. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, Cambridge.

[11] Thomas R. Gabriel and Michael R. Berthold. 2004. Influence of fuzzy norms and other heuristics on Mixed fuzzy rule formation. *International Journal of Approximate Reasoning* 35 (2004), 195–202.

[12] Bjorn Gamback and Utpal Kumar Sikdar. 2017. Using Convolutional Neural Networks to Classify Hate-Speech. In *1st Workshop on Abusive Language Online*. ACM, Vancouver, Canada.

[13] Tieliang Gong, Guangtao Wang, Jieping Ye, and Zongben Xu. 2018. Margin Based PU Learning. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. AAAI, New Orleans, Louisiana, USA, 3037–3044.

[14] Sebastian Koeffer, Dennis M Riehle, Steffen Hoehenberger, and Joerg Becker. 2018. Discussing the Value of Automatic Hate Speech Detection in Online Debates. In *Multikonferenz Wirtschaftsinformatik*. GITO-Verlag, Leuphana, Germany, 83–94.

[15] Irene Kwok and Yuzhou Wang. 2013. Locate the Hate: Detecting Tweets against Blacks. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*. AAAI Press, Bellevue, Washington, 1621–1622.

[16] Xiaoli Li and Bing Liu. 2005. Learning from Positive and Unlabeled Examples with Different Data Distributions. In *European Conference on Machine Learning*. Springer, Porto, Portugal, 218–229.

[17] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip Yu. 2003. Building Text Classifiers Using Positive and Unlabeled Examples. In *Proceedings of the Third IEEE International Conference on Data Mining*. IEEE, Melbourne, Florida, USA, 1–8.

[18] Bing Liu, Wee Sun Lee, Philip S Yu, and Xiaoli Li. 2002. Partially Supervised Classification of Text Documents. In *Proceedings of the Nineteenth International Conference on Machine Learning*. Morgan Kaufmann Publishers, Sydney, Australia, 387–394.

[19] Han Liu, Pete Burnap, Wafa Alorainy, and Matthew L. Williams. 2019. A Fuzzy Approach to Text Classification with Two Stage Training for Ambiguous Instances. *IEEE Transactions on Computational Social Systems* 6 (2019).

[20] Han Liu and Mihaela Cocea. 2017. Fuzzy Rule Based Systems for Interpretable Sentiment Analysis. In *International Conference on Advanced Computational Intelligence*. IEEE, Doha, Qatar, 129–136.

[21] Han Liu and Mihaela Cocea. 2018. *Granular Computing Based Machine Learning: A Big Data Processing Approach*. Springer, Berlin.

[22] Han Liu, Mihaela Cocea, and Weili Ding. 2018. Multi-Task Learning for Intelligent Data Processing in Granular Computing Context. *Granular Computing* 3, 3 (2018), 257–273.

[23] Han Liu, Mihaela Cocea, Alaa Mohasseb, and Mohamed Bader. 2017. Transformation of Discriminative Single-Task Classification into Generative Multi-Task Classification in Machine Learning Context. In *International Conference on Advanced Computational Intelligence*. IEEE, Doha, Qatar, 66–73.

[24] Jan Lukasiewicz. 1970. *Selected Works–Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing, Amsterdam.

[25] Irene Nemes. 2010. Regulating Hate Speech in Cyberspace: Issues of Desirability and Efficacy. *Journal of Information and Communications Technology Law* 11, 3 (2010), 193–220.

[26] Chikashi Nobata, Joel Tetreault, and Achint Thomas. 2016. Abusive Language Detection in Online User Content. In *Proceedings of the 25th International Conference on World Wide Web*. ACM, Montreal, Quebec, Canada, 145–153.

[27] Ji Ho Park and Pascale Fung. 2017. One-step and Two-step Classification for Abusive Language Detection on Twitter. In *1st Workshop on Abusive Language Online*. ACM, Vancouver, Canada.

[28] Ross J. Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Francisco.

[29] Amir H. Razavi, Diana Inkpen, Sasha Uritsky, and Stan Matwin. 2010. Offensive Language Detection Using Multi-level Classification. In *Proceedings of the 23rd Canadian conference on Advances in Artificial Intelligence*. Springer, Ottawa, Canada, 16–27.

[30] Jesse Read. 2013. Multi-label Classification. In *The Second School on Machine Learning and Knowledge Discovery in Databases*. Brazilian Computer Society, Sao Carlos, Brazil.

[31] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. 2011. Classifier chains for multi-label classification. *Machine Learning* 85 (2011), 333–359.

[32] Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining* 3, 3 (2007), 1–13.

[33] Zeerak Waseem and Dirk Hovy. 2016. Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In *Proceedings of NAACL-HLT 2016*. ACL, San Diego, California, USA, 88–93.

[34] Hajime Watanabe, Mondher Bouazizi, and Tomoaki Ohtsuki. 2018. Hate Speech on Twitter: A Pragmatic Approach to Collect Hateful and Offensive Expressions and Perform Hate Speech Detection. *IEEE Access* PP, 99 (2018), 1–11.

[35] Guang Xiang, Bin Fan, Ling Wang, Jason Hong, and Carolyn Rose. 2012. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. Springer, Maui, Hawaii, USA, 1980–1984.

[36] Ronald R. Yager, S. Ovchinnikov, R. M. Tong, and H. T. Ngugen. 1987. *Fuzzy Sets and Applications*. Wiley, New York.

[37] Lotfi Zadeh. 2015. Fuzzy Logic: A Personal Perspective. *Fuzzy Sets and Systems* 281 (December 2015), 4–20.

[38] Bangzuo Zhang and Wanli Zuo. 2008. Learning from Positive and Unlabeled Examples: A Survey. In *2008 International Symposiums on Information Processing*. IEEE, Moscow, Russia, 650–654.

[39] Min-Ling Zhang and Zhi-Hua Zhou. 2014. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering* 26, 8 (2014), 1819–1837.

[40] Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018. Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network. In *European Semantic Web Conference*. Springer, Heraklion, Crete, 745–760.

Neural Chinese Word Segmentation with Lexicon and Unlabeled Data via Posterior Regularization

Junxin Liu
Tsinghua University
Beijing, China
ljx16@mails.tsinghua.edu.cn

Fangzhao Wu
Microsoft Research Asia
Beijing, China
wufangzhao@gmail.com

Chuhan Wu
Tsinghua University
Beijing, China
wuch15@mails.tsinghua.edu.cn

Yongfeng Huang
Tsinghua University
Beijing, China
yfhuang@tsinghua.edu.cn

Xing Xie
Microsoft Research Asia
Beijing, China
Xing.Xie@microsoft.com

ABSTRACT

Chinese word segmentation (CWS) is very important for Chinese text processing. Existing methods for CWS usually rely on a large number of labeled sentences to train word segmentation models, which are expensive and time-consuming to annotate. Luckily, the unlabeled data is usually easy to collect and many high-quality Chinese lexicons are off-the-shelf, both of which can provide useful information for CWS. In this paper, we propose a neural approach for Chinese word segmentation which can exploit both lexicon and unlabeled data. Our approach is based on a variant of posterior regularization algorithm, and the unlabeled data and lexicon are incorporated into model training as indirect supervision by regularizing the prediction space of CWS models. Extensive experiments on multiple benchmark datasets in both in-domain and cross-domain scenarios validate the effectiveness of our approach.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; *Neural networks*; *Semi-supervised learning settings.*

KEYWORDS

Chinese word segmentation, Lexicon, Neural network

ACM Reference Format:
Junxin Liu, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2019. Neural Chinese Word Segmentation with Lexicon and Unlabeled Data via Posterior Regularization. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313437

1 INTRODUCTION

Chinese word segmentation (CWS) aims to segment Chinese sentence into words [7, 10, 22]. For example, "习近平常与特朗普通电话" is segmented into "习近平/常/与/特朗普/通/电话". Different from English texts where whitespace is used to separate words, there is no natural word delimiter in Chinese. Thus, CWS is very important for processing Chinese texts and is an essential step for many downstream tasks [3, 5, 15].

In recent years, neural network based methods have been widely used for CWS [1, 17, 23, 26]. Most of these methods model CWS as a sequence labeling problem [22, 30], and utilize neural networks to learn the hidden character features [2, 32]. For example, Chen et al. [2] used LSTM [8] to learn character features by capturing the global information of sentence. Peng and Dredze [18] proposed to use LSTM for character feature learning and CRF [9] for character label decoding. However, these methods usually rely on a large number of labeled sentences to train word segmentation models, which are expensive and time-consuming to annotate. Besides, these methods usually have difficulty in segmenting sentences with OOV (out of vocabulary) words or words that are rare in training data [27]. For example, if "习近平" and "特朗普" are OOV words in training data, then these methods will probably segment "习近平常与特朗普通电话" into "习近/平常/与/特朗/普通/电话".

Our work is motivated by following observations. First, the unlabeled Chinese sentences are usually easy to collect on a large scale and can provide useful information for Chinese word segmentation. For example, if the character sequences "习近平" and "特朗普" appear many times in unlabeled data with different contexts, then we can infer that they are probably Chinese words. Second, many high-quality Chinese lexicons have been built and can cover a large number of Chinese words. These lexicons can provide important information of whether a Chinese character sequence can be a valid Chinese word, which is useful for CWS. For example, if "习近平" and "特朗普" are included in a Chinese lexicon, then we can better segment aforementioned sentences. Thus, both unlabeled data and lexicons have the potential to improve the performance of CWS, especially on sentences with OOV and rare words.

In this paper, we propose a neural approach for Chinese word segmentation which can exploit the useful information in both Chinese lexicon and unlabeled data. More specifically, in our approach we propose a unified framework based on posterior regularization [6] to incorporate Chinese lexicon and unlabeled data as indirect supervision to regularize the prediction space of the neural CWS models. The neural CWS architecture used in our approach is CNN-CRF. The neural CWS model is trained based on both indirect supervision inferred from lexicon and unlabeled data and the direct supervision inferred from labeled sentences. Extensive experiments

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313437

are conducted on multiple benchmark datasets in both in-domain and cross-domain scenarios. The experimental results show that our approach can effectively improve the performance of Chinese word segmentation, especially when training data is insufficient.

2 RELATED WORK

In recent years, neural network based methods have been widely used for Chinese word segmentation. These methods usually regard CWS as a character-level sequence labeling task. For example, Chen et al. [2] proposed to apply LSTM to Chinese word segmentation. They used LSTM to learn hidden character features by capturing the global context information of sentences. They also used a character window to capture local contexts for building character features. Peng et al, [18] used LSTM to learn the contextual character features and used CRF to jointly decode the labels of characters. These neural methods for CWS usually rely on a large number of labeled sentences for model training. When the labeled data is insufficient, the performance usually declines heavily [21, 27, 28].

Incorporating the useful information in unlabeled sentences and lexicons into CWS has attracted increasing attentions [11, 20, 23]. For example, Li et al. [11] proposed to utilize unlabeled sentences for CWS by using punctuation marks as implicit annotations. However, punctuation marks are sparse in Chinese texts, and the annotations of most characters cannot be obtained in this way. Sun et al. [20] proposed to extract statistics-based character features such as mutual information and accessor variety from unlabeled data, and use these features to improve CWS. Designing these handcrafted features needs a large amount of domain knowledge. Zhang et al. [27] proposed to incorporate lexicon into a neural CWS method based on LSTM-CRF architecture. They designed several handcrafted feature templates to extract additional character features using lexicon, and used another LSTM to learn character representations from these lexicon based features. Liu et al. [12] proposed to utilize lexicon for neural CWS via multi-task learning. They designed an auxiliary task of word classification to exploit lexicon information. Then they jointly trained CWS and word classification models using a multi-task learning framework. However, these methods cannot exploit the useful information in unlabeled data.

There are a few methods which can incorporate both lexicon and unlabeled data into Chinese word segmentation. For example, Liu et al. [13] proposed to build partially annotated data using unlabeled data and lexicons via many handcrafted rules. Then they trained CWS models based on both labeled and partially annotated data using CRF. However, designing these rules requires a lot of domain knowledge. Zhao et al. [31] used a similar way to build partially labeled data for CWS by combining unlabeled data and Chinese lexicons. They trained neural CWS models on both labeled and partially labeled data using a variant of LSTM. However, in these methods the partially annotated data is built simply by word matching without considering the contexts of sentences. Thus, the partially annotated data may contain heavy noise and is not suitable to use directly as training data. Different from these methods, in our approach the lexicon and unlabeled data are incorporated to provide indirect supervision via posterior regularization.

Posterior regularization algorithm [6] can incorporate additional knowledge into model training as constraints over the posterior distributions of models. For instance, Zhang et al. [24] applied posterior regularization to integrate various kinds of knowledge such as bilingual dictionary, phrase table and coverage penalty into neural machine translation and gained huge performance improvements. Our approach is motivated by posterior regularization, and we apply it to incorporate the useful information in Chinese lexicon and unlabeled data into neural Chinese word segmentation.

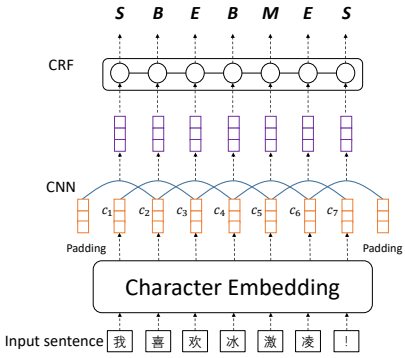


Figure 1: The CNN-CRF neural architecture for CWS.

3 OUR APPROACH

We first introduce the CNN-CRF neural architecture in our approach for Chinese word segmentation. Then we introduce our approach to incorporate lexicon and unlabeled data into neural CWS.

3.1 CNN-CRF Architecture

Following many previous works [14, 29], we model Chinese word segmentation as a character-level sequence labeling problem. For each character in a sentence, our model assigns a tag from a predefined tag set to it which indicates the position of this character in a word. The tag set used in our model is $\{B, M, E, S\}$, where B, M and E mean the beginning, middle and end position in a word respectively, and S means single character word.

We use CNN-CRF as the neural architecture for CWS which is illustrated in Fig. 1. The CNN-CRF architecture contains three layers. The first one is character embedding. Given a sentence $x = [c_1, c_2, ..., c_N]$, the character embedding layer will map each character to a low-dimensional dense vector. Here c_i is the i-th character in the sentence, and N is the sentence length. The output of this layer is $x = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_N]$, where $\mathbf{e}_i$ is the embedding of c_i.

The second layer is a CNN network. It is used to learn contextual representations of characters. The hidden feature of the i-th character learned by a convolutional kernel is:

$$h_i = f(\mathbf{w}^T \times \mathbf{e}_{i-\lceil \frac{k-1}{2} \rceil : i+\lfloor \frac{k-1}{2} \rfloor} + b), \quad (1)$$

where $\mathbf{w}$ and b are the parameters of the convolutional kernel, K is the kernel size, and $\mathbf{e}_{i-\lceil \frac{k-1}{2} \rceil : i+\lfloor \frac{k-1}{2} \rfloor}$ represents the concatenation of the embeddings from $i - \lceil \frac{k-1}{2} \rceil$-th character to $i + \lfloor \frac{k-1}{2} \rfloor$-th character. We use multiple kernels with different kernel sizes and concatenate the outputs of these convolutional kernels as the feature representation for each character. The output of CNN layer is

$[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_N]$, where $\mathbf{h}_i \in \mathcal{R}^F$, and F is the number of kernels in the CNN network.

The third layer is CRF [9]. Given a sentence $\mathbf{x} = [c_1, c_2, ..., c_N]$ and a tag sequence $\mathbf{y} = [y_1, y_2, ..., y_N]$, the score of sentence $\mathbf{x}$ having tag sequence $\mathbf{y}$ is formulated as follows:

$$s(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{N} U_{i,y_i} + \sum_{i=1}^{N-1} A_{y_i, y_{i+1}}, \tag{2}$$

where U_{i,y_i} represents the unary score of assigning tag y_i to the i-th character, and $A_{y_i,y_{i+1}}$ represents the score of jumping from tag y_i to tag y_{i+1}. The unary score $\mathbf{U}_i \in \mathcal{R}^T$ is formulated as:

$$\mathbf{U}_i = \mathbf{W}_c \mathbf{h}_i + \mathbf{b}_c, \tag{3}$$

where $\mathbf{W}_c \in \mathcal{R}^{T \times F}$ and $\mathbf{b}_c \in \mathcal{R}^T$ are trainable parameters, and T is the size of the tag set. Then the likelihood probability of sentence $\mathbf{x}$ having tag sequence $\mathbf{y}$ is defined as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{e^{\mathbf{s}(\mathbf{x},\mathbf{y})}}{\sum_{\mathbf{y}' \in \mathcal{Y}_\mathbf{x}} e^{\mathbf{s}(\mathbf{x},\mathbf{y}')}}, \tag{4}$$

where $\mathcal{Y}_\mathbf{x}$ represents the set of all possible tag sequences of sentence $\mathbf{x}$. And the loss function is formulated as follows:

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{N_l} \log(p(\mathbf{y}_i|\mathbf{x}_i;\boldsymbol{\theta})), \tag{5}$$

where $\mathbf{x}_i$ is the i-th training sentence, $\mathbf{y}_i$ is its ground truth tag sequence, N_l is the number of labeled sentences in training set, and $\boldsymbol{\theta}$ represents all parameters of the neural CWS model.

3.2 Neural CWS with Lexicon and Unlabeled Data via Posterior Regularization

In this section we introduce our approach to exploit lexicon and unlabeled data to train a neural CWS model. Our approach is based on posterior regularization [6], and we propose a unified framework to incorporate the useful information in lexicon and unlabeled data as indirect supervision into model training by regularizing the prediction space of neural CWS model. In our approach the neural CWS model is trained in an iterative manner. Following [24], in iteration t, the loss function of the indirect supervision is:

$$\mathcal{L}^{PR}(\boldsymbol{\theta}) = \sum_{i=1}^{N_u} \mathrm{KL}(Q(\mathbf{y}|\mathbf{x}_i;\boldsymbol{D},\boldsymbol{\theta}^t)||p(\mathbf{y}|\mathbf{x}_i;\boldsymbol{\theta})), \tag{6}$$

where $\boldsymbol{\theta}$ is the parameter set of CNN-CRF model, $\boldsymbol{\theta}^t$ is the model learned in iteration $t-1$, $\boldsymbol{D}$ represents Chinese lexicon, and N_u is the number of unlabeled sentences. KL is the KL divergence function. $Q(\mathbf{y}|\mathbf{x};\boldsymbol{D},\boldsymbol{\theta}^t)$ is the probability distribution of tag sequence $\mathbf{y}$ for unlabeled sentence $\mathbf{x}$ given lexicon $\boldsymbol{D}$ and previous model $\boldsymbol{\theta}^t$:

$$Q(\mathbf{y}|\mathbf{x};\boldsymbol{D},\boldsymbol{\theta}^t) = \frac{\exp(\phi(\mathbf{y},\mathbf{x};\boldsymbol{D},\boldsymbol{\theta}^t))}{\sum_{\hat{\mathbf{y}} \in \mathcal{Y}(\mathbf{x})} \exp(\phi(\hat{\mathbf{y}},\mathbf{x};\boldsymbol{D},\boldsymbol{\theta}^t))}, \tag{7}$$

where $\phi(\mathbf{y},\mathbf{x};\mathbf{D},\boldsymbol{\theta}^t)$ is a score function of tag sequence $\mathbf{y}$ for sentence $\mathbf{x}$, and $\mathcal{Y}(\mathbf{x})$ is the set of all possible tag sequences of $\mathbf{x}$. $\phi(\mathbf{y};\mathbf{x},\mathbf{D},\boldsymbol{\theta}^t)$ is designed to encode both lexicon information and the predictions of previous CWS model towards this sentence:

$$\phi(\mathbf{y};\mathbf{x},\boldsymbol{D},\boldsymbol{\theta}^t) = \frac{n(\mathbf{x},\mathbf{y};\boldsymbol{D})}{n(\mathbf{x},\mathbf{y})} + \alpha \cdot s(\mathbf{x},\mathbf{y};\boldsymbol{\theta}^t), \tag{8}$$

where $n(\mathbf{x},\mathbf{y})$ is the number of words in the segmentation result, $n(\mathbf{x},\mathbf{y};\boldsymbol{D})$ is the number of words in segmentation result which are included in lexicon $\boldsymbol{D}$, $s(\mathbf{x},\mathbf{y};\boldsymbol{\theta}^t)$ is the segmentation score predicted by the CWS model $\boldsymbol{\theta}^t$ trained in previous iteration according to Eq. (2), and α is a positive coefficient. According to Eq. (8), if a tag sequence $\mathbf{y}$ for an unlabeled sentence $\mathbf{x}$ can lead to more lexicon-included words and has higher segmentation score according to existing CWS model, then it will have a higher probability in Eq. (7), and we regularize our neural CWS model so that it tends to generate this tag sequence in Eq. (6). In this way, the useful information in lexicon and unlabeled sentences can be incorporated into the learning of neural CWS model as indirect supervision.

Since a sentence usually has many possible tag sequences, following [24], KL function in Eq. (6) is approximated as:

$$\sum_{\mathbf{y} \in \mathcal{S}(\mathbf{x}_i)} \tilde{Q}(\mathbf{y}|\mathbf{x}_i;\boldsymbol{D},\boldsymbol{\theta}^t) \log(\frac{\tilde{Q}(\mathbf{y}|\mathbf{x}_i;\boldsymbol{D},\boldsymbol{\theta}^t)}{p(\mathbf{y}|\mathbf{x}_i;\boldsymbol{\theta})}), \tag{9}$$

where $\mathcal{S}(\mathbf{x}_i)$ is a subset of $\mathcal{Y}(\mathbf{x}_i)$ whose elements have the highest prediction scores according to existing CWS model $\boldsymbol{\theta}^t$. $\tilde{Q}$ is an approximation of Q on the subset $\mathcal{S}(\mathbf{x}_i)$ as follows:

$$\tilde{Q}(\mathbf{y}|\mathbf{x}_i;\boldsymbol{D},\boldsymbol{\theta}^t) = \frac{\exp(\phi(\mathbf{y},\mathbf{x}_i;\boldsymbol{D},\boldsymbol{\theta}^t))}{\sum_{\hat{\mathbf{y}} \in \mathcal{S}(\mathbf{x}_i)} \exp(\phi(\hat{\mathbf{y}},\mathbf{x}_i;\boldsymbol{D},\boldsymbol{\theta}^t))}. \tag{10}$$

In our approach, in each iteration the neural CWS model $\boldsymbol{\theta}$ is updated based on both labeled sentences and the indirect supervision from lexicon and unlabeled data. The objective function for model update at iteration t is formulated as follows:

$$\mathcal{J}(\boldsymbol{\theta}) = -\sum_{i=1}^{N_l} \log(p(\mathbf{y}_i|\mathbf{x}_i;\boldsymbol{\theta})) - \lambda \sum_{i=1}^{N_u} \sum_{\hat{\mathbf{y}} \in \mathcal{S}(\mathbf{x}_i)} \tilde{Q}(\hat{\mathbf{y}}|\mathbf{x}_i;\boldsymbol{D},\boldsymbol{\theta}^t) \log(p(\hat{\mathbf{y}}|\mathbf{x}_i;\boldsymbol{\theta})), \tag{11}$$

where λ is a positive coefficient to control the relative importance of indirect supervision in model training. In the first iteration, the initial neural CWS model $\boldsymbol{\theta}^1$ is trained on labeled sentences.

4 EXPERIMENT

4.1 Datasets and Experimental Settings

Two benchmark datasets released by the third international Chinese language processing bakeoff[1] [10] are used in our experiments. The first one is the *MSRA* dataset, which contains 46,364 labeled sentences for training and 4,365 labeled sentences for test. The second one is the *UPUC* dataset, which contains 18,804 and 5,117 labeled sentences for training and test.

The lexicon used in our experiments is the Sogou Chinese lexicon[2]. The size of character embeddings is 200. These character embeddings are pretrained on the Sogou news corpus[3] using the word2vec [16] tool. We use 400 convolutional kernels in the CNN network, and the sizes of these kernels vary from 2 to 5. λ in Eq. (11) and α in Eq. (8) are set to 0.5 and 1 respectively. We apply dropout technique to the embedding layer and the CNN layer, and the dropout rate is 0.3. RMSProp [4] algorithm is used for model training. The learning rate is 0.001 and the batch size is 64. These hyper-parameters are selected using validation data. Following [2],

[1] http://sighan.cs.uchicago.edu/bakeoff2006/download.html
[2] http://www.sogou.com/labs/resource/list_lan.php
[3] http://www.sogou.com/labs/resource/ca.php

we use the last 10% sentences in the training set as validation data, and the remaining labeled sentences for model training. In addition, we randomly sample 50% training data as unlabeled data. Each experiment is repeated 5 times and the average results are reported.

4.2 Performance Evaluation

In this section we evaluate the performance of our approach by comparing it with many baseline methods for Chinese word segmentation. These methods include: (1) LSTM-CRF, the most popular neural method for CWS based on the LSTM-CRF architecture [18]; (2) CNN-CRF, the neural CWS method based on the CNN-CRF architecture, which is the basic model in our approach; (3) Chen [2], a neural CWS method using LSTM to learn character features and also considering local contexts; (4) Zhang [27], an LSTM-CRF based CWS method which integrates lexicon into model training via feature templates; (5) Liu [12], a neural CWS method which incorporates lexicon into model training via multi-task learning; (6) Liu [13], a CRF based CWS method which utilizes lexicon and unlabeled data to build partially labeled data for model training; (7) Zhao [31], an LSTM based CWS method which incorporates lexicon and unlabeled data via building partially labeled data; (8) LUPR, our proposed neural CWS approach with both lexicon and unlabeled data via posterior regularization. We conducted experiments on different ratios of training data, and the experimental results of different methods are summarized in Tables 1 and 2.

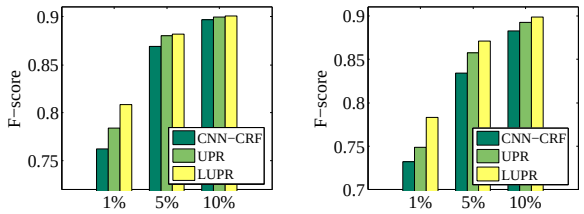
Table 1: The results on the *MSRA* dataset. P, R and F represent precision, recall and Fscore respectively.

	1%			5%			10%		
	P	R	F	P	R	F	P	R	F
LSTM-CRF	75.87	76.18	76.01	82.81	82.18	82.49	85.24	84.68	84.95
CNN-CRF	77.19	75.35	76.26	87.19	86.64	86.91	89.95	89.48	89.71
Chen [2]	77.20	74.32	75.73	84.19	83.44	83.80	87.50	86.05	86.76
Zhang [27]	76.64	76.55	76.60	87.15	86.73	86.94	89.49	89.10	89.29
Liu [12]	78.06	77.55	77.80	87.60	86.50	87.05	90.06	89.47	89.77
Liu [13]	81.48	78.92	80.18	83.76	81.58	82.66	85.20	83.09	84.13
Zhao [31]	80.68	**80.26**	80.47	86.94	85.67	86.30	88.69	87.21	87.94
LUPR	**81.98**	79.74	**80.84**	**88.42**	**87.92**	**88.17**	**90.35**	**89.83**	**90.09**

Table 2: The results on the *UPUC* dataset.

	1%			5%			10%		
	P	R	F	P	R	F	P	R	F
LSTM-CRF	70.49	73.44	71.92	79.66	80.95	80.29	82.97	84.73	83.84
CNN-CRF	72.03	74.50	73.22	82.60	84.23	83.40	87.75	88.79	88.27
Chen [2]	73.04	74.24	73.61	80.87	81.56	81.20	85.03	87.08	86.04
Zhang [27]	74.93	73.45	74.18	84.39	85.94	85.15	88.15	89.05	88.60
Liu [12]	73.01	76.33	74.63	83.82	85.70	84.75	87.71	89.27	88.48
Liu [13]	**79.50**	75.65	77.53	81.83	78.25	80.00	83.39	80.22	81.77
Zhao [31]	77.11	77.57	77.34	82.70	82.07	82.39	85.19	86.07	85.63
LUPR	78.18	**78.49**	**78.33**	**86.89**	**87.36**	**87.13**	**89.48**	**90.29**	**89.88**

According to Tables 1 and 2, our approach can outperform many neural Chinese word segmentation methods such as CNN-CRF, LSTM-CRF and Chen [2]. In addition, the advantage of our approach over these baseline methods becomes bigger when the number of training samples decreases. This is because these methods rely on a large number of labeled sentences to train neural CWS models, and cannot exploit the useful information in lexicon and unlabeled data.



(a) *MSRA* dataset. (b) *UPUC* dataset.

Figure 2: The performance of the basic CNN-CRF model and our approach with only unlabeled data (UPR) and with both lexicon and unlabeled data (LUPR).

When training data is insufficient, it is very difficult for these methods to train accurate and robust CWS models. Since our approach can exploit the useful information in both lexicon and unlabeled data, it can reduce the dependence on labeled sentences and achieve better performance than these baseline methods.

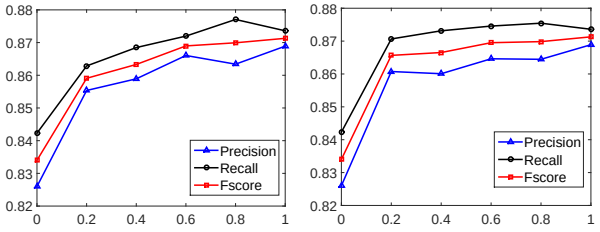
Although the methods proposed in [27] and [12] can also incorporate lexicon information into neural CWS model training, our approach can consistently outperform them. In [27] the lexicon is utilized via handcrafted feature templates, which need a lot of domain knowledge to design and may not be optimal. In addition, an extra LSTM network is incorporated to learn hidden character representations from these lexicon based features, making the neural model more difficult to train when labeled data is scarce. In [12] the lexicon is utilized via an auxiliary word classification task which is jointly trained with CWS model. Although these methods can utilize the lexicon for neural CWS, the useful information in massive unlabeled data is not considered. Our approach can exploit both lexicon and unlabeled data for neural Chinese word segmentation. Thus, our approach can consistently outperform them.

Although the methods in [13] and [31] can also exploit both lexicon and unlabeled data for CWS, our approach can still outperform them. In [13] and [31], the lexicon and unlabeled data are used to build partially annotated datasets. However, the partially annotated data constructed by word matching based on lexicons may contain heavy noise and the context information of sentences is not considered. In our approach, the lexicon and unlabeled data are used to provide indirect supervision for model training by regularizing the prediction space of the neural CWS model. The experimental results show that our approach is more effective in exploiting Chinese lexicon and unlabeled data for CWS than [13] and [31].

4.3 Effect of Lexicon and Unlabeled Data

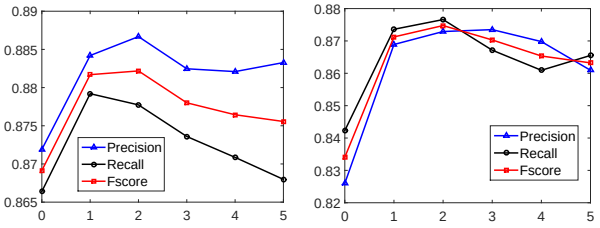
In this section, we conducted experiments to explore the effectiveness of lexicon and unlabeled data for neural Chinese word segmentation. The experimental results are summarized in Fig. 2.

According to Fig. 2, incorporating unlabeled data can effectively improve the performance of CWS in our approach. This is because the massive unlabeled data contains rich useful information for word segmentation. For example, if the character sequence "特朗普" frequently appear in unlabeled sentences with different contexts, then we can infer that it is probably a unique Chinese word. The result in Fig. 2 shows that our approach is effective in exploiting



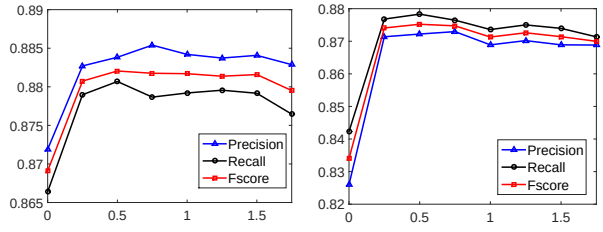
(a) Ratio of unlabeled data. (b) Ratio of words in lexicon.

Figure 3: Influence of unlabeled data and lexicon size.



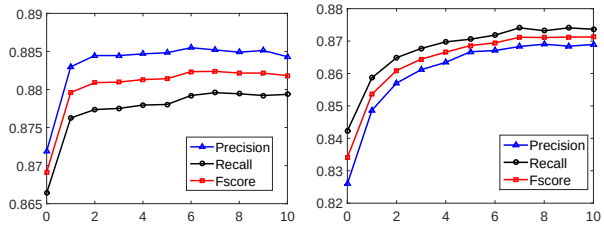
(a) *MSRA* dataset. (b) *UPUC* dataset.

Figure 4: Influence of S size in Eq. (9).



(a) *MSRA* dataset. (b) *UPUC* dataset.

Figure 5: Influence of λ in Eq. (11).



(a) *MSRA* dataset. (b) *UPUC* dataset.

Figure 6: Influence of iteration number.

the useful information in unlabeled data for CWS. In addition, according to Fig. 2 after incorporating lexicon the performance of our approach can be further improved. This is because the Chinese lexicon can provide important information of whether a character sequence can be a valid Chinese word, which is useful for the CWS task. The result in Fig. 2 validates that our approach can effectively exploit Chinese lexicon to improve the performance of neural CWS.

4.4 Size of Lexicon and Unlabeled Data

In this section, we conduct experiments to explore the influence of the sizes of unlabeled data and Chinese lexicon on the performance of our approach. The experiments were conducted on the *UPUC* dataset, and we randomly sampled 5% of the training data for model training. The experiments on the *MSRA* dataset show similar patterns. The experimental results of unlabeled data size are summarized in Fig. 3(a). We can see that as more unlabeled data is incorporated, the performance of our approach consistently improves. This result further validates that the unlabeled data contains a lot of useful information for Chinese word segmentation, and our approach is effective in exploiting unlabeled data for neural CWS methods. The experimental results of lexicon size are shown in Fig. 3(b). According to Fig. 3(b), as more Chinese words are included in the lexicon, the performance of our approach improves. This is because with more Chinese words in the lexicon, our approach can have better capacity in recognizing the boundaries of words which rarely or never appear in training data but are included in the Chinese lexicon.

4.5 Influence of Hyper-parameters

In this section, we conducted experiments to explore the influence of three important hyper-parameters, i.e., the size of set S in Eq. (9), λ in Eq. (11) and iteration number, on the performance of our approach. The experimental results of the size of set S in Eq. (9) are shown in Fig. 4. In these experiments, we randomly sampled 5% of the training data for model training. According to Fig. 4, when the size of S in Eq. (9) increases from 0 to 1, the performance of our approach significantly improves. This is because when the size of S is 0, the useful information in unlabeled data and lexicon is not incorporated. Thus, the performance is not optimal. When the size of S becomes too large, the performance of our approach slightly decreases. It indicates that the number of informative tag sequences of unlabeled sentences inferred from lexicon is usually limited, and incorporating too many of them may introduce some noisy information. A moderate size of S (e.g., 2) is most suitable for our approach.

The experimental results of λ are shown in Fig. 5. λ is used to control the relative importance of the indirect supervision from unlabeled data and lexicon in Eq. (11). According to Fig. 5, as λ increases, the performance of our approach first increases and then slightly decreases. This is because when λ is too small, the useful information in the lexicon and unlabeled data is not fully exploited. Thus, the performance is not optimal. However, when λ becomes too large, the indirect supervision inferred from lexicon and unlabeled data is over-emphasized and the labeled sentences are not fully respected. Thus, the performance starts to decline. A moderate value of λ is most appropriate for our approach.

The experimental results of iteration number are shown in Fig. 6. According to Fig. 6, as the iteration number grows, the performance of our approach first improves and then gradually becomes stable. This is because in each iteration the neural CWS model can be enhanced by incorporating the unlabeled data and lexicon, and the refined CWS model in turn can help improve the indirect supervision in the next iteration of our approach. This result validates the

effectiveness of our approach in iteratively training neural CWS model by exploiting both labeled sentences and the indirect supervision inferred from the unlabeled sentences and Chinese lexicons.

4.6 Domain Adaptation for CWS

In Chinese word segmentation field, several domains (e.g., news) have accumulated much labeled data, while in many other domains (e.g., medical records) labeled data for CWS is scarce and even nonexistent. Although annotating sufficient labeled data for these domains is time-consuming and expensive, the unlabeled sentences are usually easy to collect. In addition, in many target domains there are off-the-shelf lexicons or it is relatively easy to build one. Thus, an interesting application of our approach is domain adaptation for Chinese word segmentation, i.e., using labeled sentences in a source domain (e.g., news) and unlabeled sentences in a target domain (e.g., medical records) as well as the lexicon from the target domain to train a robust neural CWS model for target domain. In this section we conduct experiments to explore the performance of our approach in domain adaptation for word segmentation.

Two datasets are used in our experiments. The first one is the *Zhuxian* dataset[4] built by Zhang et al. [25] from a Chinese online novel. We used the same lexicon[5] as [31] for this domain. In addition, we used the same unlabeled data as [13] which contains about 16K sentences. The second dataset is the Weibo dataset[6] released by the Weibo word segmentation task of NLPCC2016 [19]. This dataset contains 20,135 sentences for training and 8,592 sentences for test. Since there is no off-the-shelf lexicon for Weibo word segmentation, we built one using the words extracted from the training data. In addition, we regarded the training sentences as unlabeled data. We used *Zhuxian* and *Weibo* datasets as two target domains, and used the *UPUC* dataset as source domain. The experimental results are summarized in Table 3. The settings of our approach and baseline methods are consistent with previous experiments.

Table 3: Experimental results of domain adaptation.

	Zhuxian			Weibo		
	P	*R*	*F*	*P*	*R*	*F*
CNN-CRF	87.95	87.68	87.81	86.11	88.23	87.16
LSTM-CRF	85.94	86.26	86.10	86.39	88.35	87.36
Chen [2]	85.24	87.57	86.39	86.87	89.28	88.06
Zhang [27]	88.70	87.89	88.29	88.94	**90.89**	89.90
Liu [12]	87.79	88.04	87.91	86.24	88.49	87.35
Liu [13]	89.86	89.04	89.44	**90.31**	89.89	90.10
Zhao [31]	88.50	**91.32**	89.90	89.83	85.07	87.39
LUPR	**90.53**	89.39	**89.95**	89.85	90.73	**90.29**

According to Table 3, the performance of neural models that are trained on labeled data of source domain such as CNN-CRF, LSTM-CRF and [2] is relatively low in target domains. This is because there is huge difference in word distribution between source and target domains. Many words in target domain may not or rarely appear in the labeled data of source domain, making it difficult for these CWS methods to segment the sentences in target domain. The methods proposed in [27] and [12] which incorporate lexicon can effectively improve the performance of CWS in target domains. Our approach can outperform these methods because beyond the lexicon information our approach can also incorporate massive unlabeled data into neural model training, which can provide useful information for CWS. Although [13] and [31] can incorporate both lexicon and unlabeled data for word segmentation, our approach can outperform them in the domain adaptation scenario. This result implies that incorporating the lexicon and unlabeled data as indirect supervision via posterior regularization is more suitable for domain adaptation of CWS than utilizing them to build partially labeled data through word matching, which may introduce heavy noise.

[4]http://zhangmeishan. github.io/eacl14mszhang.zip
[5]This lexicon is crawled from http://baike.baidu.com/view/18277.htm
[6]https://github.com/FudanNLP/NLPCC-WordSeg-Weibo

4.7 Case Study

In this section, we conducted several case studies to further explore why our approach can improve the performance of CWS. We studied several cases in the domain adaptation experiments, where the source domain is *UPUC* and the target domain is *Weibo*. Several examples are illustrated in Table 4.

Table 4: Several examples of Chinese word segmentation.

	Example 1	Example 2
Sentence	养老金调整三大焦点问题	穆勒操刀梅开二度
CNN-CRF	养/老金/调整/三/大/焦点/问题	穆勒/操刀/梅开/二/度
LUPR	养老金/调整/三/大/焦点/问题	穆勒/操刀/梅开二度

According to Table 4, our approach performs better than baseline methods in domain adaptation scenario, especially on the sentences with OOV and rare words of source domain. For example, in the first example “养老金" is a rare word in the training data of source domain, and in the second example “梅开二度" is an OOV word in source domain. The segmentation results of the basic CNN-CRF model on these words are not correct. Since “养老金" is a popular word in the unlabeled data of target domain and “梅开二度" is included in the target domain lexicon, our approach can correctly segment these sentences. Thus, our approach can improve the performance of Chinese word segmentation by exploiting the useful information in both lexicon and unlabeled data.

5 CONCLUSION

In this paper, we propose a neural approach for Chinese word segmentation which can exploit the useful information in both Chinese lexicon and unlabeled sentences for model training. Our approach is based on the posterior regularization algorithm, and we propose a unified framework to incorporate both unlabeled data and lexicon to provide indirect supervision for model training by regularizing the prediction space of the neural CWS models. Extensive experiments are conducted on multiple benchmark datasets in both in-domain and cross-domain scenarios. The experimental results validate that our approach can effectively improve the performance of neural Chinese word segmentation.

ACKNOWLEDGMENTS

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFC1604002 and in part by the National Natural Science Foundation of China under Grant U1705261, Grant U1536201, Grant U1536207.

REFERENCES

[1] Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and Accurate Neural Word Segmentation for Chinese. In *ACL*. 608–615.

[2] Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015. Long short-term memory neural networks for chinese word segmentation. In *EMNLP*. 1197–1206.

[3] Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial Multi-Criteria Learning for Chinese Word Segmentation. In *ACL*. 1193–1203.

[4] Yann Dauphin, Harm de Vries, and Yoshua Bengio. 2015. Equilibrated adaptive learning rates for non-convex optimization. In *NIPS*. 1504–1512.

[5] Schubert Foo and Hui Li. 2004. Chinese word segmentation and its effect on information retrieval. *Information Processing & Management* 40, 1 (2004), 161–190.

[6] Kuzman Ganchev, Jennifer Gillenwater, Ben Taskar, et al. 2010. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research* 11, Jul (2010), 2001–2049.

[7] Jianfeng Gao, Mu Li, Andi Wu, and Chang-Ning Huang. 2005. Chinese word segmentation and named entity recognition: A pragmatic approach. *Computational Linguistics* 31, 4 (2005), 531–574.

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[9] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*. Morgan Kaufmann, 282–289.

[10] Gina-Anne Levow. 2006. The Third International Chinese Language Processing Bakeoff: Word Segmentation and Named Entity Recognition. In *SIGHAN@COLING/ACL*. Association for Computational Linguistics, 108–117.

[11] Zhongguo Li and Maosong Sun. 2009. Punctuation as implicit annotations for chinese word segmentation. *Computational Linguistics* 35, 4 (2009), 505–512.

[12] Junxin Liu, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2018. Neural Chinese Word Segmentation with Dictionary Knowledge. In *NLPCC*. 80–91.

[13] Yijia Liu, Yue Zhang, Wanxiang Che, Ting Liu, and Fan Wu. 2014. Domain adaptation for CRF-based Chinese word segmentation using free annotations. In *EMNLP*. 864–874.

[14] Jin Kiat Low, Hwee Tou Ng, and Wenyuan Guo. 2005. A Maximum Entropy Approach to Chinese Word Segmentation. In *SIGHAN@IJCNLP 2005*. ACL, 161–164.

[15] Wencan Luo and Fan Yang. 2016. An Empirical Study of Automatic Chinese Word Segmentation for Spoken Language Understanding and Named Entity Recognition. In *NAACL*. 238–248.

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013).

[17] Wenzhe Pei, Tao Ge, and Baobao Chang. 2014. Max-margin tensor neural network for Chinese word segmentation. In *ACL*. 293–303.

[18] Nanyun Peng and Mark Dredze. 2017. Multi-task Domain Adaptation for Sequence Tagging. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*. 91–100.

[19] Xipeng Qiu, Peng Qian, and Zhan Shi. 2016. Overview of the NLPCC-ICCPOL 2016 Shared Task: Chinese Word Segmentation for Micro-Blog Texts. In *NLPCC/ICCPOL (Lecture Notes in Computer Science)*, Vol. 10102. Springer, 901–906.

[20] Weiwei Sun and Jia Xu. 2011. Enhancing Chinese word segmentation using unlabeled data. In *EMNLP*. 970–979.

[21] Jingjing Xu, Shuming Ma, Yi Zhang, Bingzhen Wei, Xiaoyan Cai, and Xu Sun. 2017. Transfer Deep Learning for Low-Resource Chinese Word Segmentation with a Novel Neural Network. In *NLPCC (Lecture Notes in Computer Science)*, Vol. 10619. Springer, 721–730.

[22] Nianwen Xue. 2003. Chinese word segmentation as character tagging. *IJCLCLP* 8, 1 (2003), 29–48.

[23] Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural Word Segmentation with Rich Pretraining. In *ACL*. 839–849.

[24] Jiacheng Zhang, Yang Liu, Huanbo Luan, Jingfang Xu, and Maosong Sun. 2017. Prior knowledge integration for neural machine translation using posterior regularization. In *ACL*. 1514–1523.

[25] Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. 2014. Type-supervised domain adaptation for joint segmentation and pos-tagging. In *EACL*. 588–597.

[26] Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Transition-based neural word segmentation. In *ACL*, Vol. 1. 421–431.

[27] Qi Zhang, Xiaoyu Liu, and Jinlan Fu. 2018. Neural Networks Incorporating Dictionaries for Chinese Word Segmentation. In *AAAI*. 5682–5689.

[28] Yanna Zhang, Jinan Xu, Guoyi Miao, Yufeng Chen, and Yujie Zhang. 2018. Addressing Domain Adaptation for Chinese Word Segmentation with Instances-Based Transfer Learning. In *CCL (Lecture Notes in Computer Science)*, Vol. 11221. Springer, 24–36.

[29] Hai Zhao, Changning Huang, and Mu Li. 2006. An Improved Chinese Word Segmentation System with Conditional Random Field. In *SIGHAN@COLING/ACL*. Association for Computational Linguistics, 162–165.

[30] Hai Zhao, Changning Huang, Mu Li, and Bao-Liang Lu. 2006. Effective Tag Set Selection in Chinese Word Segmentation via Conditional Random Field Modeling. In *PACLIC*. ACL, 87–94.

[31] Lujun Zhao, Qi Zhang, Peng Wang, and Xiaoyu Liu. 2018. Neural Networks Incorporating Unlabeled and Partially-labeled Data for Cross-domain Chinese Word Segmentation. In *IJCAI*. 4602–4608.

[32] Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for Chinese word segmentation and POS tagging. In *EMNLP*. 647–657.

User-Video Co-Attention Network for Personalized Micro-video Recommendation

Shang Liu
School of Remote Sensing and Information Engineering
Wuhan University, China
shangliu@whu.edu.cn

Zhenzhong Chen*
School of Remote Sensing and Information Engineering
Wuhan University, China
zzchen@whu.edu.cn

Hongyi Liu
Amazon Alexa, USA
liuhn@amazon.com

Xinghai Hu
Facebook Inc, USA
xinhai.me@gmail.com

ABSTRACT

With the increasing popularity of micro-video sharing where people shoot short-videos effortlessly and share their daily stories on social media platforms, the micro-video recommendation has attracted extensive research efforts to provide users with micro-videos that interest them. In this paper, a hypothesis we explore is that, not only do users have multi-modal interest, but micro-videos have multi-modal targeted audience segments. As a result, we propose a novel framework User-Video Co-Attention Network (UVCAN), which can learn multi-modal information from both user and microvideo side using attention mechanism. In addition, UVCAN reasons about the attention in a stacked attention network fashion for both user and micro-video. Extensive experiments on two datasets collected from Toffee present superior results of our proposed UVCAN over the state-of-the-art recommendation methods, which demonstrate the effectiveness of the proposed framework.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Recommendation, micro-video, personalization, deep learning, attention mechanism

ACM Reference Format:
Shang Liu, Zhenzhong Chen, Hongyi Liu, and Xinghai Hu. 2019. User-Video Co-Attention Network for Personalized Micro-video Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313513

1 INTRODUCTION

In the era of mobile Internet, people are now able to shoot micro-videos (or short-videos) effortlessly and share their daily stories on social media platforms, such as Musical.ly and TikTok. With the usage of social media platforms rapidly increasing, millions of micro-videos are being generated from users of different kinds. For example, Figure 1 gives an example of micro-video on Musical.ly. The ever-increasing volume of micro-videos makes it harder for users to find the micro-video that best interests them. This has increased the demand for recommender system more than ever before.

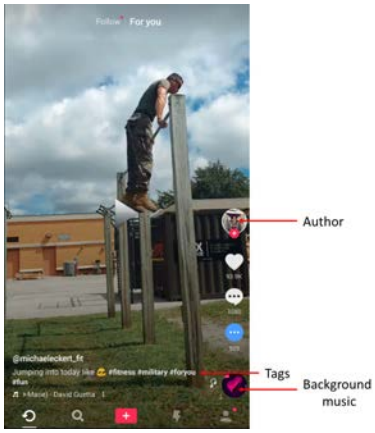


Figure 1: An example of micro-video on Musical.ly.

An offline recommender system predicts user preference and retrieves micro-video list which will be recommended to them by using users' features such as user ID or cookies, and personalizes the micro-videos for individual users [9]. However, traditional ID-based recommendation methods, such as collaborative filtering (CF) [27] and low-rank factorization [15], only consider the linear users' interaction with micro-videos without exploring non-linear and more complicated user-video interaction representation based on their features, which is important for users' hidden preferences learning. To solve this problem, recently Ma et al. [24] have proposed a genre aware neural network micro-video recommendation model on micro-video recommendation. Micro-video attention based model [5] has been explored for recommendation, where the attention mechanism learns a video representation highlighting micro-video frames relevant to user's preference. In this paper, we argue that, not only do users have multi-modal interest, but micro-videos have multi-modal targeted audience segments. In addition to visual feature, we also consider more micro-video side information that has effect on user's preference.

*The corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313513

In this paper, we present a novel User-Video Co-Attention Network (UVCAN) framework for personalized micro-video recommendation which can learn multi-modal information from both user and micro-video side using attention mechanism. Unlike previous work, which connects these side information features as the user or micro-video embedding representation and only focuses on video attention, UVCAN proposes alternating co-attention to attend both user and video modalities. Through the user-video co-attention network, micro-video's and user's representations focus more on key features that demonstrate user's hidden preference. In addition, for each attention processing, we design stacked attention network to learn attention representation since only one layer of reasoning may be not enough to learn complex multi-modal features representation. Besides, we adopt a connection to associate the learned user representation, micro-video multi-modal representation, user ID embedding and micro-video ID embedding, hoping to combine collaborative filtering and attention mechanism learning for personalized micro-video recommendation.

The main contributions of this work are summarized as follows:

- A novel co-attention mechanism for personalized micro-video recommendation is proposed to jointly perform user-guided micro-video attention and video-guided user attention. Moreover, we explore this mechanism with multi-modal micro-video features and user profile.
- A stacked attention network is proposed for each attention processing to deal with complex multi-modal information, and attention learning on user attributes or micro-video multi-modal features is refined with the network step by step.
- Real-world micro-video datasets are constructed, which consist of user's attributes (user gender and location), micro-video's content information (micro-video author, background music and tags) and user-video interaction records. Extensive experiments on our collected micro-video datasets with about 336K interactions demonstrate that our proposed UVCAN outperforms current state-of-the-art methods that use hybrid approaches.

2 RELATED WORK

2.1 Video Recommendation

The existing approaches to deal with video recommendation fall into three categories: collaborative filtering [1, 11, 18], content-based filtering [9, 10] and hybrid approaches [3, 5]. Majority of the existing approaches deal with traditional videos (e.g., YouTube) while little research has been conducted for micro-videos. Because of the increasing popularity of micro-video sharing, researchers have paid more attention to micro-video data analytics. For example, Ma et al. [24] proposed a latent genre aware neural network micro-video recommendation model on micro-video recommendation. More recently, Chen et al. [7] proposed temporal attention network for user behavior modeling for micro-video recommendation. In this paper, we build new micro-video datasets involving more side information, such as user's attribute information as well as micro-video's background music and author, for studying personalized micro-video recommendation. Our proposed model using stacked co-attention deep network to sequentially model user and micro-video representations is also distinctive from the aforementioned work.

2.2 Recommendation with Neural Networks

Recently, deep neural networks are being used to extract refined latent features [14] from the user-item interaction data. The early work [26] proposes Restricted Boltzmann Machines (RBM) with an input layer and a hidden layer for collaborative filtering. Auto-Encoder has been applied to recommendation tasks, e.g., AutoRec [28], DAE [22] and CDAE [33]. Cheng et al. [8] propose Wide& Deep approach for app recommendation, where the deep component is a MLP on concatenated feature embedding vectors.

Attention mechanism has been applied to deal with a variety of problem including computer vision [34], neural language processing [23] and recommender systems [2, 4, 30, 31]. For example, Chen et al. [4] introduce a attention mechanism to explore the usefulness of reviews, and propose a neural attentional regression model with review-level explanations for recommendation. Wei et al. [36] propose a user-guided attention network to attend image's multi-modal information for image representation learning. However, Chen's and Wei's work [4, 36] only utilize user-guided attention on item to generate item representation, and do not attend user side attention. These methods are fundamentally different from our proposed approach that employs user-video co-attention network to jointly describe user and micro-video attention mechanism.

3 PRELIMINARIES

In this section, we first introduce the notation and define the problem. We then introduce the embedding of user's and micro-video's information to construct the input representation.

3.1 Problem Formulation and Notations

Table 1: Definitions of notations.

Notation	description
$\mathcal{U}, \mathcal{V}$	user set, micro-video set
S^u, S^v	multi-modal information embedding of u, v
u, v	a specific user, item
$\boldsymbol{h}^u, \boldsymbol{h}^v$	identity embedding vector of u, v. $h^u, h^v \in \mathbb{R}^K$
$\boldsymbol{s}^u, \boldsymbol{s}^v$	feature embedding vector of u, v. $s^u, s^v \in \mathbb{R}^K$
K	dimensionality of the embedding vector
$\boldsymbol{Y}$	user-video interaction matrix $\boldsymbol{Y} \in \mathbb{R}^{U \times I}$
$\delta(\cdot)$	the logistic sigmoid function

We are tackling the micro-video recommendation problem by jointly considering user and micro-video attention. We consider the task of recommending micro-videos denoted as $\mathcal{V} = \{v_1, v_2, ..., v_{|\mathcal{V}|}\}$ to users denoted as $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$. Each user u is associated with some attribute features denoted as S^u (e.g., user gender, location): $S^u = (S^u_1, S^u_2, ..., S^u_{|S^u|})$. The micro-video content information is denoted as S^v (e.g., micro-video visual content, author, background music): $S^v = (S^v_1, S^v_2, ..., S^v_{|S^v|})$. We define a user-video interaction matrix as $\boldsymbol{Y} \in \mathbb{R}^{U \times I}$, where the entry y_{uv} is defined

from user's implicit feedback, $y_{uv} = 1$ indicates that the user u has given the micro-video v a thumbs-up and $y_{uv} = 0$ indicates that there is no observed data about the interaction (user u, micro-video v). Our goal is to predict the user's preference for micro-videos not exposed to them before, and recommend micro-videos accordingly. Some notation frequently used throughout this paper is introduced in Table 1. Matrices are denoted with bold symbols.

3.2 Construction of Input Representation

For side information of users and micro-videos, the original representation are strings (e.g., 'gender=F' for a user, 'tag=fun' for a micro-video). We convert these sparse categorical features to low-dimensional vectors, which are referred as embedding vectors and initialized randomly. Incorporating user's identity vector $\boldsymbol{h}^u$ and side information, the embedding vectors for user u is extended as $\{\boldsymbol{h}^u, \boldsymbol{s}^u_1, \boldsymbol{s}^u_2, ..., \boldsymbol{s}^u_{|s^u|}\}$. The side information for the user is user gender and location. The side information for micro-video is tags, author id and background music id.

For micro-video visual feature, we follow the work [21] by extracting visual feature from micro-videos using pre-trained model. In particular, we extract image features at 1-frame-per-second using an Inception-v3 network [29]. We fetch the Relu activation of the last hidden layer and apply PCA (and whitening) to reduce feature dimensions to 1024 for storage and computational reasons. These frame-level features are aggregated into video-level by average pooling. For convenience of calculation, we use a single layer perceptron to convert each visual vector into a new vector that has the same dimension K as other side information feature vector. Incorporating micro-video identity vector $\boldsymbol{h}^v$ and visual feature, the embedding vectors for micro-video v is extended as $\{\boldsymbol{h}^v, \boldsymbol{s}^v_1, \boldsymbol{s}^v_2, ..., \boldsymbol{s}^v_{|s^v|}\}$, where $\boldsymbol{s}^v$ is the multi-modal information embedding for micro-video.

4 USER-VIDEO CO-ATTENTION NETWORK (UVCAN)

In this section, we first present our User-Video Co-Attention Network (UVCAN) model in detail. We then go through the learning details of UVCAN. The overall architecture of the model is shown in Figure 2.

For the user's embedding column vectors, we concatenate them to form the whole user feature representation matrix: $\boldsymbol{q}^u = concat^{(1)}(\boldsymbol{h}^u, \boldsymbol{s}^u_1, \boldsymbol{s}^u_2, ..., \boldsymbol{s}^u_{|s^u|})$ where $\boldsymbol{q}^u \in \mathbb{R}^{K\times(|s^u|+1)}$. For the micro-video's embedding column vectors, we concatenate them to form the whole micro-video feature representation matrix: $\boldsymbol{q}^v = concat^{(1)}(\boldsymbol{h}^v, \boldsymbol{s}^v_1, \boldsymbol{s}^v_2, ..., \boldsymbol{s}^v_{|s^v|})$ where $\boldsymbol{q}^v \in \mathbb{R}^{K\times(|s^v|+1)}$. Because not only do users have multi-modal interest, but micro-video have multi-modal targeted audience segments, we propose a user-video co-attention network that learn multi-modal information from both user and micro-video side using attention mechanism.

We first define the attention operation $\hat{x} = \mathcal{A}(\boldsymbol{X}, \boldsymbol{g})$, which takes the micro-video (or user) features $\boldsymbol{X}$ and attention guidance $\boldsymbol{g}$ derived from user (or micro-video) as inputs, and outputs the attended micro-video (or user) vector x. The operation can be denoted in the following.

$$\begin{aligned} \boldsymbol{H} &= tanh(\boldsymbol{W}_x\boldsymbol{X} + (\boldsymbol{W}_g\boldsymbol{g})\boldsymbol{I}^T), \\ \boldsymbol{a}^x &= softmax(\boldsymbol{w}^T_{hx}\boldsymbol{H} + \boldsymbol{b}), \\ \hat{\boldsymbol{x}} &= \sum a^x_i \boldsymbol{x}_i, \end{aligned} \tag{1}$$

where $\boldsymbol{I}$ is a vector with all elements to be 1. $\boldsymbol{W}_x, \boldsymbol{W}_g \in \mathbb{R}^k$ are parameters. $\boldsymbol{a}^x$ is the attention weight of feature $\boldsymbol{X}$.

4.1 User-guided Micro-video Attention

In most cases, a user is attracted by some specific micro-video features. For example, the user may pay more attention to the micro-video's visual content and author, which will play vital importance role for micro-video representation that caters user's personal preference. Hence, instead of using a global vector connecting micro-video features, we use a user-guided attention layer to filter and find features that are relevant user's personalized preference.

Before the micro-video attention, we first take user's features $\boldsymbol{q}^u$ and attention guidance $\boldsymbol{g} = \boldsymbol{0}$ to summarize the user features into a single vector $\hat{\boldsymbol{q}}_0$. Next, we take micro-video features $\boldsymbol{q}^v$ as $\boldsymbol{X}$ and the guidance $\boldsymbol{g}$ is the intermediate attended user feature $\hat{\boldsymbol{q}}_0$. Based on the attention probability of each micro-video feature, the new representation of the micro-video is constructed as weighted sum of the micro-video vector. we then use the new micro-video representation $\hat{\boldsymbol{v}}_v$ to guide the user attention.

4.2 Video-guided User Attention

Compared to models that only use user identity to guide item attention, the co-attention network model develops a more informative representation by mutually attending micro-video and user. After the phrase of user-guided micro-video attention (see Figure 2), UVCAN further incorporates user attention guided by micro-video representation since user's preference on micro-video varies with different user attributes such as gender and location. In order to obtain the user attention distribution, we take the new representation $\hat{\boldsymbol{v}}^v$ of micro-video to query the original user feature matrix $\boldsymbol{q}^u$ according to Equation 1. We then generate a new representation $\hat{\boldsymbol{v}}^u$ for user.

4.3 Stacked Attention Networks

For each attention processing, we design stacked attention network to learn attention representation which can explore some subtle relationship among query and value by iteratively querying the original feature matrix in each $\mathcal{A}$. The formula can be summarized as follows: for the m-th attention layer (where m is greater than or equal to 2), we compute the distribution of video-guided (or user-guided) by user (or micro-video) query and generate a new representation for the micro-video (or user) based on the attention probability. The new query vector is formed by adding the new feature vector to previous query vector, the equation for micro-video attention guided by user vector is as follows:

$$q^u_m = \hat{v}^v_m + q^u_{m-1}, \tag{2}$$

where q^u_1 is initialized to be $\hat{q}_0$. The user attention is as same.

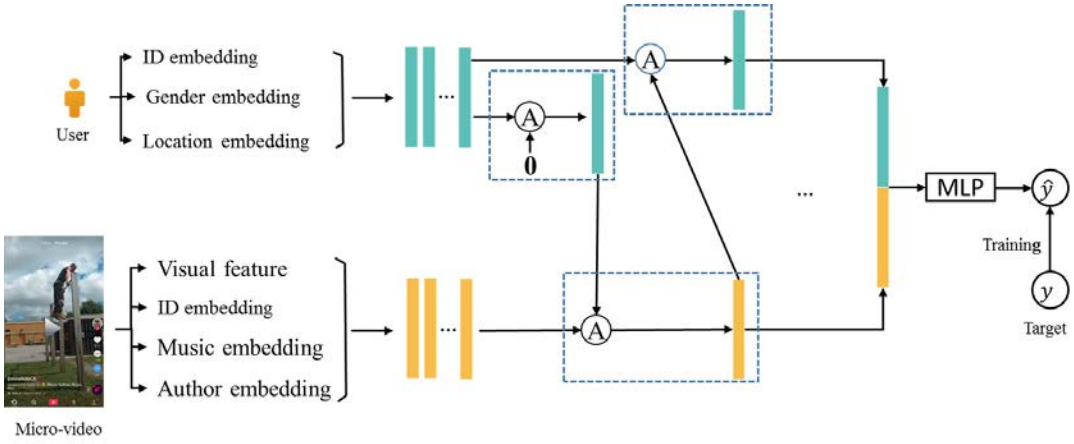


Figure 2: The illustration of Collaborative filtering stacked attentive network framework with three-step reasoning to explore user's attention to micro-video's features.

4.4 Learning for Recommendation

To test the effectiveness of learned user and micro-video representations via co-attention network on personalized micro-video recommendation, we connect these two learned multi-modal information representation vectors, user ID embedding vector and micro-video ID embedding vector to get the combination of collaborative filtering and attention mechanism. And we utilize a simple 3-layer feed-forward neural network to generate final preference rating, which dose not incur much modal complexity and ensures the capacity of nonlinear modeling for recommendation. Specifically, we define the computational formula as follows:

$$z = \begin{bmatrix} \hat{\boldsymbol{v}}^u \\ \hat{\boldsymbol{v}}^v \\ \boldsymbol{h}^u \\ \boldsymbol{h}^v \end{bmatrix}, \tag{3}$$

$$\hat{y}_{uv} = \text{MLP}(z), \tag{4}$$

where the final output layer is the estimated scores $\hat{y}_{uv}$ and training is carried out by minimizing the point wise loss function between estimated score $\hat{y}_{uv}$ and real value y_{uv}.

To learn the model parameters, UVCAN is trained as a binary classification problem. Considering the one-class nature of implicit feedback, we can view $y_{ui} = 1$ as a positive label that means user u likes the micro-video v otherwise it is a negative one. The UVCAN predicts the likelihood of user u favors micro-video v. With sigmoid $\delta(\cdot)$ as the activation function for the output layer, we can constrain the output $\hat{y}_{uv}$ in $[0, 1]$. Based on the above setting, we minimized the cross-entropy loss for true labels and sampled negative instances:

$$L(y_{uv}, \hat{y}_{uv}) = \sum_{(u,v)\in Y^+ \cup Y^-} -y_{uv}\log(p) - (1-y_{uv})\log(1-p), \tag{5}$$

where $p = \delta(\hat{y}_{uv}) = 1/(1 + exp(-\hat{y}_{uv}))$ represents the sigmoid function on the output $\hat{y}_{uv}$. Y^+ and Y^- denote the observed user-video pairs and unobserved user-video pairs, respectively. We uniformly sample negative instances Y^- from unobserved interaction according to the fixed sampling ratio 4:1 to the number of observed interactions in each iteration. An Adaptive Moment Estimation (Adam) [19] algorithm is adopted to optimize the loss function.

5 EXPERIMENTS

In this section, we conduct extensive experiments on the collected micro-video datasets from Toffee to compare UVCAN against other state-of-the-art recommendation methods. We first briefly depict the collected two real-world micro-video datasets, followed by experimental setting and evaluation scheme. Then we introduce the baseline algorithms and parameter settings. Finally, we present our experimental results and discussions, with comparison to different kinds of baseline methods.

5.1 Dataset

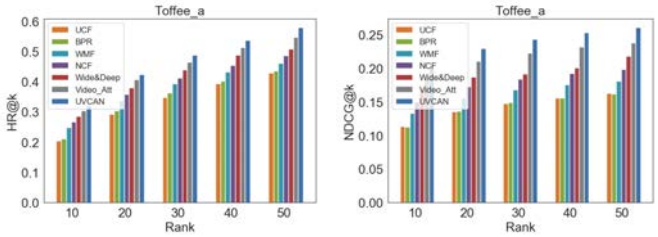
Table 2: Basic statistics of datasets.

Dataset	#User	#Micro-video	#Interaction	Span
Toffee_a	780	73,015	233,144	2017.01-2017.06
Toffee_b	3,231	50,574	336,460	2017.07-2018.06

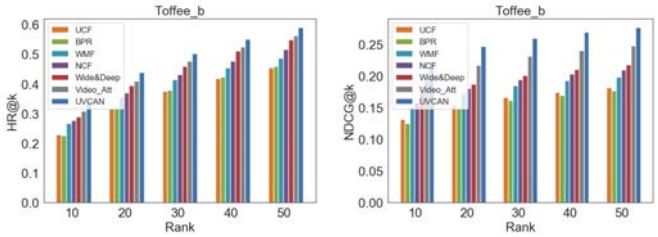
We construct our datasets using micro-videos collected from Toffee, one of China's popular micro-video social media platform. As far as we know, there is no publicly available micro-video social media dataset that contains users' metadata, content information of micro-videos, and the user-to-video connections simultaneously. In addition to establish connections (make new friends) between users, Toffee allows users to publish and share videos to others, and send thumbs-up to different videos. Toffee is a micro-video social media app which starts run from Jan 2017. To study the effect of different phrase of development on recommendation, we crawled micro-videos of Jan 2017 to Jun 2017 as the developing phrase and Jul 2017 to Jun 2018 as the plateau.

We constructed the two datasets from the Toffee in a user-centric method, which is also adopted by [6]. We first crawled one week of public timeline micro-videos and random sampled active users.

We defined active users who have at least interacted (send a thumb-up to the video to express liking) with 5 micro-videos. Then we expanded the seed users by crawling their followees, as the followees of active users were more likely to be active users relative to their followers. We performed three layers of crawling and further crawled users' ID and registering information, the ID list of their preferred micro-videos and the content information of these micro-video such as video file, tag, author ID and music ID. To test a scenario where user profile is known, we filtered out users without location or gender information. With this crawling method, we got the two datasets for different developing phrase: Toffee_a and Toffee_b. The detailed statistics of these datasets after preprocessing are shown in Table 2. The sparsity ($\#interaction/(\#user \times \#video)$) of the two datasets is 0.4094% for Toffee_a and 0.2059% for Toffee_b.



(a) Toffee_a — HR@k (b) Toffee_a — NDCG@k

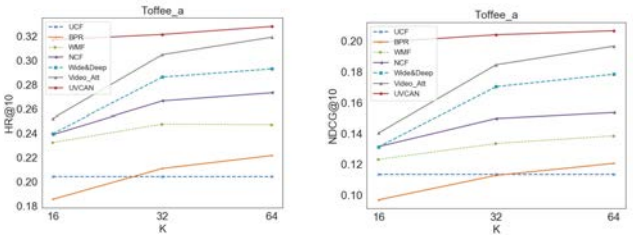


(c) Toffee_b — HR@k (d) Toffee_b — NDCG@k

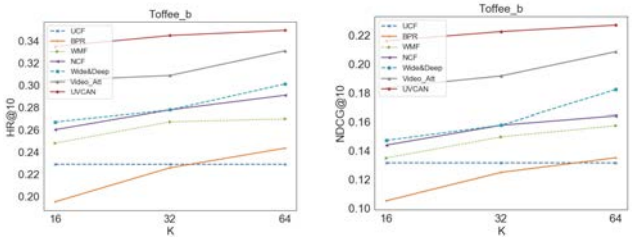
Figure 3: Performance of top-k recommendation where k range from 10 to 50 on Toffee_a.

5.2 Evaluation Scheme

For each dataset, we randomly hold 20% of micro-videos associated with each user to form the test set, and put the other micro-videos in the training set. The splitting methods is widely used in previous work on recommendation system [16, 32, 35]. In term of performance evaluation, for each (u, v) pair in the test set, we random select 1000 additional micro-video and predict scores by u for v and the other 1000 micro-videos. The evaluation criteria is to measure how well the method ranks the correct pair (u, v) against the other random micro-videos. This evaluation strategy is commonly used by [12, 13, 20]. We evaluate our method and other compared ones on micro-video recommendation by using two popular metrics, i.e., Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG), which regard liking as positive labels.



(a) Toffee_a — HR@10 (b) Toffee_a — NDCG@10



(c) Toffee_b — HR@10 (d) Toffee_b — NDCG@10

Figure 4: Performance of HR@10 and NDCG@10 w.r.t dimensionality of predictive factors on two datasets.

5.3 Baselines

We compare our proposed UVCAN with a series of state-of-the-art baseline algorithms as follows: **UCF** [37], **BPR** [25], **WMF** [17], **NCF** [14], **Wide&Deep** [8] and **Video-Att**. **Video-Att** is a variant of our proposed model, which only use user-guided micro-video attention. Similar work is done by [5], which proposes a linear model considering user-guided attention to item while **Video-Att** extends the implement to non-linear model.

5.4 Parameter Settings

We use a Gaussian distribution to randomly initialized the model parameters with a mean of 0 and standard deviation of 0.1 for our UVCAN model, and optimize the model through mini-batch Adaptive Moment Estimation (Adam). We tested the learning rate of [0.0001, 0.001, 0.01, 0.1], latent feature dimension of [16, 32, 64] and the regularizer of [0, 0.001, 0.01, 0.1]. The grid search method is used to determine hyper-parameters. If not specified, we employed three hidden layers for multi-Layer perceptron and show the results of $K = 32$ for all methods because of the consistent across the dimension of latent vector. Without special mention, we employ two attention layers for UVCAN.

5.5 Results and Analysis

5.5.1 Model Comparison. The performance of all competitors with HR@10 and NDCG@10 on micro-video recommendation is shown in Table 3. Figure 3 demonstrates the performance of Top-k recommendation list with ranking position starting from 10 to 50 on Toffee_a and Toffee_b dataset. From the table and figure, we have the following observations:

Table 3: Recommendation performance in terms of the HR@10 and NDCG@10.

Dataset		(a) UCF	(b) BPR	(c) WMF	(d) NCF	(e) Wide&Deep	(f) Video-Att	(g) UVCAN	Improvement (g) vs. (c)	(g) vs. (e)	(g) vs. Best
Toffee_a	HR10	0.2043	0.2110	0.2478	0.2666	0.2863	0.3049	**0.3217**	29.81%	12.36%	5.50%
	NDCG10	0.1135	0.1128	0.1335	0.1497	0.1705	0.1847	**0.2042**	52.92%	19.74%	10.53%
Toffee_b	HR10	0.2290	0.2259	0.2672	0.2780	0.2891	0.3089	**0.3448**	29.06%	19.28%	11.64%
	NDCG10	0.1317	0.1250	0.1497	0.1578	0.1765	0.1917	**0.2226**	48.72%	26.13%	16.13%

(a) On average, our proposed method outperforms the second-best method by 8.57% in terms of HR@10 and 13.33% in terms of NDCG@10 across the two datasets. We speculate that learning users' hidden preferences through combining micro-video and user mutual attention mechanism improves the performance of recommendation, which can be also seen from the superior performance of UVCAN than that of Video_Att.

(b) Using attention networks to explore a more powerful latent representation for user and micro-video improves the performance of recommendation. UVCAN and Video_Att outperform the model that concatenates identity embedding and side feature embedding of user and micro-video and feeds them directly in deep neural networks (namely, NCF and Wide&Deep).

(c) As can be seen from Figure 3, UVCAN shows consistent improvement compared with other methods in different ranking positions. While the Toffee_b dataset is more sparse than Toffee_a, the performance of all competitor on Toffee_b is much better. The possible reason may be that in Toffee_b dataset, which is from the developing plateau of the social media platform, the connection between users is constructed more densely, which also explain why the performance of UCF on Toffee_b is better than that of BPR.

5.5.2 Effects of the number of predictive factors. Figure 4 shows the performance of HR@10 and NDCG@10 with respect to different size of predictive factors on two datasets. When other factors remain the same, increasing the number of predictive factors introduces more information. As we can observe from Figure 4, compared to other competitor methods, the performance improvement of UVCAN also increases substantially. The possible reason is that the combination of collaborative filtering and attention mechanism requires larger embedding vector to incorporate the abundant information. For Toffee_b, even with a small predictive factor of 16, UVCAN performs substantially perform better than WMF and BPR with a large factor of 64. This shows the high effectiveness of UVCAN by learning users' hidden preferences.

5.5.3 Effects of the number of attention layers. We experiment with different number of attention layers to investigate the effect on the recommendation performance of UVCAN on Toffee_a and Toffee_b datasets, which is shown in Tables 4 and 5. We only show the results of NDCG, and the results of HR admit the same trend thus they are omitted. AL-2 indicates the micro-video attention and user attention both have two attention layers with other factors remaining the same. As can be seen, the performance improves at the beginning which also indicates the effectiveness of stacked attention layers. We owe this improvement to the multi-step reasoning of user-guided attention on micro-video's side information and video-guided attention on user's attribute information. When using more attention layers than two or three, the performance does not further improve significantly.

Table 4: NDCG@10 of UVCAN with different attention layers on Toffee_a.

factors	AL-1	AL-2	AL-3	AL-4
16	0.1947	**0.1993**	0.1948	0.1920
32	0.2022	**0.2042**	0.1927	0.1982
64	0.1947	0.1968	**0.2009**	0.2008

Table 5: NDCG@10 of UVCAN with different attention layers on Toffee_b.

factors	AL-1	AL-2	AL-3	AL-4
16	0.2045	**0.2161**	0.2123	0.2124
32	0.2161	**0.2226**	0.2116	0.2094
64	0.2105	0.2171	**0.2219**	0.2159

6 CONCLUSION

In this paper, we propose a User-Video Co-Attention Network (UVCAN) framework for personalized micro-video recommendation, based on the hypothesis that not only do users have multi-modal interest but micro-videos have multi-modal targeted audience segments. UVCAN achieves superior performance over state-of-the-art methosd by following advantages: 1) It uses embedding-based method to represent side information features and users' history interaction behavior with micro-videos; 2) It explores users' hidden preferences on side information of micro-videos, which views user's features as an input query and learn micro-video attention through multi-step reasoning with stacked attention network. 3) In addition, it explores user's diversity preference from different attribute, which views the learned micro-video representation as query and learn user attention through multi-step reasoning with stacked attention network. We conduct extensive experiments on two collected real-world micro-video datasets to compare UVCAN with state-of-the-art methods and demonstrate its effectiveness.

ACKNOWLEDGEMENTS

This work was supported in part by National Key R&D Program of China under contract No. 2017YFB1002202, and National Natural Science Foundation of China under contract No. 61771348.

REFERENCES

[1] Shumeet Baluja, Rohan Seth, D Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. 2008. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proceedings of the 17th International Conference on World Wide Web(WWW'08)*. ACM, 895–904.

[2] Da Cao, Xiangnan He, Lianhai Miao, Yahui An, Chao Yang, and Richang Hong. 2018. Attentive Group Recommendation. In *Proceedings of the 41th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'18)*. ACM.

[3] Bisheng Chen, Jingdong Wang, Qinghua Huang, and Tao Mei. 2012. Personalized video recommendation through tripartite graph propagation. In *Proceedings of the 20th ACM International Conference on Multimedia (MM'12)*. ACM, 1133–1136.

[4] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural A entional Rating Regression with Review-level Explanations. In *Proceedings of the 27th International Conference on World Wide Web (WWW'18)*. International World Wide Web Conferences Steering Committee.

[5] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and TatSeng Chua. 2017. Attentive collaborative filtering: multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'17)*. ACM, 335–344.

[6] Tao Chen, Xiangnan He, and Min-Yen Kan. 2016. Context-aware image tweet modelling and recommendation. In *Proceedings of the 24nd ACM International Conference on Multimedia (MM'16)*. ACM, 1018–1027.

[7] Xusong Chen, Dong Liu, Zheng-Jun Zha, Wengang Zhou, Zhiwei Xiong, and Yan Li. 2018. Temporal Hierarchical Attention at Category-and Item-Level for Micro-Video Click-Through Prediction. In *Proceedings of the 26nd ACM International Conference on Multimedia (MM'18)*. ACM, 1146–1153.

[8] HengTze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys'16)*. ACM, 7–10.

[9] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys'16)*. ACM, 191–198.

[10] Peng Cui, Zhiyu Wang, and Zhou Su. 2014. What videos are similar with you?: Learning a common attributed representation for video recommendation. In *Proceedings of the 22nd ACM International Conference on Multimedia (MM'14)*. ACM, 597–606.

[11] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. 2010. The YouTube video recommendation system. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys'10)*. ACM, 293–296.

[12] Chao Du, Chongxuan Li, Yin Zheng, Jun Zhu, and Bo Zhang. 2018. Collaborative Filtering with User-Item Co-Autoregressive Models. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence (AAAI'18)*. AAAI Press.

[13] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *Proceedings of the 24th International Conference on World Wide Web (WWW'13)*. International World Wide Web Conferences Steering Committee, 278–288.

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and TatSeng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web(WWW'17)*. International World Wide Web Conferences Steering Committee, 173–182.

[15] Xiangnan He, Hanwang Zhang, MinYen Kan, and TatSeng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'16)*. ACM, 549–558.

[16] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative metric learning. In *Proceedings of the 26th International Conference on World Wide Web (WWW'17)*. International World Wide Web Conferences Steering Committee, 193–201.

[17] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 15th IEEE International Conference on Data Mining (ICDM'08)*. IEEE, 263–272.

[18] Yanxiang Huang, Bin Cui, Jie Jiang, Kunqian Hong, Wenyu Zhang, and Yiran Xie. 2016. Real-time video recommendation exploration. In *Proceedings of the 2016 International ACM SIGMOD Conference on Management of Data (SIGMOD'16)*. ACM, 35–46.

[19] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 2015 International Conference on Learning Representations (ICLR'2015)*.

[20] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'08)*. ACM, 426–434.

[21] Joonseok Lee, Sami AbuElHaija, Balakrishnan Varadarajan, and Apostol Paul Natsev. 2018. Collaborative Deep Metric Learning for Video Understanding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'18)*. ACM, 481–490.

[22] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep collaborative filtering via marginalized denoising auto-encoder. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM'15)*. ACM, 811–820.

[23] Qiao Liu, Haibin Zhang, Yifu Zeng, Ziqi Huang, and Zufeng Wu. 2018. Content Attention Model for Aspect Based Sentiment Analysis. In *Proceedings of the 27th World Wide Web Conference on World Wide Web (WWW'18)*. International World Wide Web Conferences Steering Committee, 1023–1032.

[24] Jingwei Ma, Guang Li, Mingyang Zhong, Xin Zhao, Lei Zhu, and Xue Li. 2018. LGA: latent genre aware micro-video recommendation on social media. *Multimedia Tools and Applications* (2018), 2991–3008.

[25] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI'09)*. AUAI Press, 452–461.

[26] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*. 791–798.

[27] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Darius Braziunas. 2016. On the Effectiveness of Linear Models for One-Class Collaborative Filtering. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, 229–235.

[28] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web(WWW'15)*. International World Wide Web Conferences Steering Committee, 111–112.

[29] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15)*. 1–9.

[30] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent relational metric learning via memory-based attention for collaborative ranking. In *Proceedings of the 27th World Wide Web Conference on World Wide Web (WWW'18)*. International World Wide Web Conferences Steering Committee, 729–739.

[31] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Multi-Pointer Co-Attention Networks for Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'18)*. ACM.

[32] Keqiang Wang, Yuanyuan Jin, Haofen Wang, Hongwei Peng, and Xiaoling Wang. 2018. Personalized Time-Aware Tag Recommendation.. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence (AAAI'18)*. AAAI Press.

[33] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM'16)*. ACM, 153–162.

[34] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the 32nd International Conference on Machine Learning (ICML'2015)*. 2048–2057.

[35] Wenhui Yu, Huidi Zhang, Xiangnan He, Xu Chen, Li Xiong, and Zheng Qin. 2018. Aesthetic-based clothing recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web (WWW'18)*. International World Wide Web Conferences Steering Committee, 649–658.

[36] Wei Zhang, Wen Wang, Jun Wang, and Hongyuan Zha. 2018. User-guided hierarchical attention network for multi-modal social image popularity prediction. In *Proceedings of the 27th World Wide Web Conference on World Wide Web (WWW'18)*. International World Wide Web Conferences Steering Committee, 1277–1286.

[37] ZhiDan Zhao and MingSheng Shang. 2010. User-based collaborative-filtering recommendation algorithms on hadoop. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'10)*. ACM, 478–481.

Recommender Systems with Heterogeneous Side Information

Tianqiao Liu
TAL AI Lab
Beijing, China
liutianqiao@100tal.com

Zhiwei Wang*
Data Science and Engineering Lab
Michigan State University
wangzh65@msu.edu

Jiliang Tang
Data Science and Engineering Lab
Michigan State University
tangjili@msu.edu

Songfan Yang
TAL AI Lab
Beijing, China
yangsongfan@100tal.com

Gale Yan Huang
TAL AI Lab
Beijing, China
galehuang@100tal.com

Zitao Liu†
TAL AI Lab
Beijing, China
liuzitao@100tal.com

ABSTRACT

In modern recommender systems, both users and items are associated with rich side information, which can help understand users and items. Such information is typically heterogeneous and can be roughly categorized into flat and hierarchical side information. While side information has been proved to be valuable, the majority of existing systems have exploited either only flat side information or only hierarchical side information due to the challenges brought by the heterogeneity. In this paper, we investigate the problem of exploiting heterogeneous side information for recommendations. Specifically, we propose a novel framework jointly captures flat and hierarchical side information with mathematical coherence. We demonstrate the effectiveness of the proposed framework via extensive experiments on various real-world datasets. Empirical results show that our approach is able to lead a significant performance gain over the state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Recommender systems; Side information; Hierarchical structure

ACM Reference Format:
Tianqiao Liu, Zhiwei Wang, Jiliang Tang, Songfan Yang, Gale Yan Huang, and Zitao Liu. 2019. Recommender Systems with Heterogeneous Side Information. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, Article 4, 7 pages. https://doi.org/10.1145/3308558.3313580

1 INTRODUCTION

Recommender systems can mitigate the information overload problem by providing online users with the most relevant information [23, 25, 27]. A successful recommender system often requires accurate understanding of user preferences. Collaborative filtering, which models the interactions between users and items, is one of the most popular techniques to achieve this goal [9, 11, 25]. Traditional collaborative filtering based recommender systems have been proven to be suffered from the data sparsity and cold-start problems [1, 27]. On the other hand, in addition to interactions, users and items are often associated with side information, which has become increasingly available in real-world recommender systems [5, 21]. Such side information provides independent sources for recommendations, which can mitigate the data sparsity and cold start problems and have great potentials to boost the performance. As a consequence, a large body of research has been developed to exploit side information for recommendations [15, 23, 31, 32]. Side information is typically heterogeneous, which can be roughly categorized into flat and hierarchical side information [31]. Flat and hierarchical side information are referred to attributes associated with users and items presenting no hierarchical and hierarchical structures, respectively [31]. Take books for example, side information of one book can include the publish year, the book authors, the written language, and the genres it belongs to. Attributes such as year, author and language, presenting no hierarchical structures, are flat. The genres, however, contain *subtypeOf* relationship and can be organized in a hierarchical structure. Figure 1 gives a concrete example, which shows six attributes of the book *Animal Farm* with colored background. The flat information includes *George Orwell*, *1945* and *English* and are listed in the left part of figure. *Literature&Fiction*, *Historical Fiction*, and *Political* are the genres

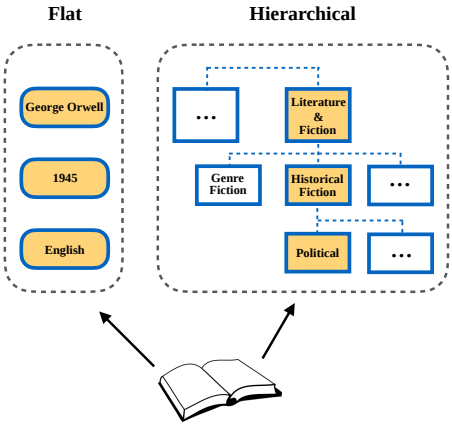


Figure 1: An illustration of flat and hierarchical side information.

*Work was done when the author did internship in TAL AI Lab.
†Corresponding author: Zitao Liu.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313580

Animal Farm belongs to according to Amazon Web Store. As shown in the figure, these genres are organized into a hierarchical structure as genre→subgenre→detailed-genre such that *Animal Farm* firstly belongs to the genre *Literature&Fiction*, under which there are sub-genres such as *Historical Fiction* and *Genre Fiction*. In this sub-genre level, *Animal Farm* belongs to *Historical Fiction*. It further falls into a more detailed-category *Political*. Likewise, users are also associated with both flat information such as age, gender, education level and hierarchical information such as communities they belong to and the places of their birth.

There are numerous works incorporating flat or hierarchical side information for recommendations [5, 15, 21, 23, 31, 32]. Most of these systems have been designed to exploit either only flat side information or only hierarchical side information. The major reason is that flat and structured side information are intrinsically different and it is challenging to jointly exploit them. One trivial solution is to flatten the hierarchy and treat it as flat information. However, such solution ignores the unique properties of hierarchical information, which have been proven beneficial by previous works [15, 19, 31]. In fact, both flat and hierarchical side information can provide valuable information for understanding user preferences and item characteristics. For instance, female are generally more interested in high heel shoes than male and items belonging to the same detailed-genre are likely to be more similar than those in the same sub-genre. Thus, it is desired to design frameworks incorporating the two types of side information simultaneously.

In this paper, we investigate the problem of exploiting both flat and hierarchical side information for recommendations. We propose a novel framework, which aims to address two challenges: (1) how to jointly capture heterogeneous side information and (2) how to mathematically use them for recommendations. The main contributions of our work can be summarized as follows:

- We provide a principled approach to simultaneously capture both flat and hierarchical information mathematically.
- We introduce a unified recommendation framework HIRE, which can model **H**eterogeneous side **I**nformation for **RE**commendation coherently.
- We conduct extensive experiments with various real-world datasets to validate the effectiveness of the proposed framework and understand the importance of flat and hierarchical side information for recommendations.

2 METHODOLOGY

In this section, we present the proposed recommendation framework that coherently captures flat and hierarchical information of both users and items. Specifically, we firstly introduce the notations that will be used in the rest of the paper and then describe a basic model which forms the basis of the framework. After that, we go into details of the framework components that model the flat and heterogeneous information, respectively, combining of which leads to an optimization problem. Finally, we propose an efficient algorithm to solve it.

Throughout this paper, regular letters are used to denote scalers. The vectors and matrices are represented by bold lower-case letters such as $\mathbf{h}$ and bold upper-case letters such as $\mathbf{W}$, respectively. In addition, for an arbitrary matrix $\mathbf{W}$, we use $\mathbf{W}(i,:)$ and $\mathbf{W}(:,j)$ to denote the i^{th} row and j^{th} column of it, respectively; and the $(i,j)^{th}$ entry of $\mathbf{W}$ is represented as $\mathbf{W}(i,j)$. The transpose and Frobenius norm of $\mathbf{W}$ is denoted as $\mathbf{W}^T$ and $\|\mathbf{W}\|_F^2$, respectively. Moreover, let $\mathcal{U} = \{u_1, u_2, \cdots, u_n\}$ to be the set of n users and $\mathcal{V} = \{v_1, v_2, \cdots, v_m\}$ be the set of m items. We assume there exists a user-item rating matrix $\mathbf{R} \in \mathbb{R}^{n\times m}$ and if a user u_i has rated item v_j, $\mathbf{R}(i,j) > 0$ denotes the rating score, otherwise, $\mathbf{R}(i,j) = 0$. In addition, let $\mathbf{X} \in \mathbb{R}^{d_x\times n}$ and $\mathbf{Y} \in \mathbb{R}^{d_y\times m}$ be the matrices that contain flat side information of users and items with d_x and d_y associated attributes, respectively. Next, we will describe a basic recommendation model, based on which we will build the whole framework.

2.1 The Basic Model

Weighted matrix factorization is an effective approach used in collaborative filtering based recommender systems to obtain users' and items' representations that contain information regarding users' preference and items' characteristics. Specifically, it decomposes the rating matrix and models the users and items in the same low-dimensional latent space. Mathematically, weighted matrix factorization solves the following optimization problem:

$$\min_{\mathbf{U},\mathbf{V}} \underbrace{\|\mathbf{M}\odot(\mathbf{R}-\mathbf{U}\mathbf{V})\|_F^2 + \lambda(\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2)}_{f(\mathbf{U},\mathbf{V})}$$

where $\odot$ is the Hadamard operation denoting element-wise multiplication. $\mathbf{M} \in \mathbb{R}^{n\times m}$ is the indication matrix such that $\mathbf{M}(i,j) = 1$ if $\mathbf{R}(i,j) > 0$, otherwise, $\mathbf{M}(i,j) = 0$. The obtained matrices $\mathbf{U} \in \mathbb{R}^{n\times d}$ and $\mathbf{V} \in \mathbb{R}^{d\times m}$ are the corresponding representations of users and items in the latent space. Thus, the rating score given by u_i to v_j is approximated by the dot item of their latent representations, that are $\mathbf{U}(i,:)$ and $\mathbf{V}(:,j)$, respectively. λ is used to control the weight of regularization terms that are adopted to avoid overfitting. One of the most important strength of matrix factorization based model is that it allows to incorporate other information in addition to the ratings [11]. Next, we will base on the basic weighted matrix factorization model to build our framework.

2.2 Capturing The Flat Information

The side information of items or users are intrinsically heterogeneous, which can be flat and heterogeneous. For example, in Figure 1, a book can have both flat attributes such as author, year, and hierarchical attributes such as genres it belongs to. The difference between different types of side information requires special treatment for each individual. In this subsection, we describe the model component that aims to capture the flat side information. To simplify, we first focus on capturing side information of users and then generalize it to that of items.

In weighted matrix factorization, users' latent representation matrix $\mathbf{U}$ contains their preference indicated by the rating scores they give to items. The side information, however, provides another independent source from which users' preference could be inferred. For example, it is very likely that a programmer is more interested in a mechanical keyboard than a dancer, which suggests that users' occupation could provide an important indicator whether an item should be recommended or not. In addition, both hidden representation $\mathbf{U}(i,:)$ and side information $\mathbf{X}(:,i)$ describe the same user u_i.

Hence, in the same latent space, $\mathbf{U}(i,:)$ and $\mathbf{X}(:,i)$ should be similar. With this intuition, we extend the basic weight matrix factorization model to capture the flat side information contained in $\mathbf{X}$ as follows:

$$\min_{\mathbf{U},\mathbf{V}} f(\mathbf{U},\mathbf{V}) + \gamma \|\mathbf{S}^u\mathbf{U}^T - \mathbf{X}\|_F^2$$

where $\mathbf{S}^u \in \mathbb{R}^{d_x \times d}$ is the projection matrix that projects users' hidden representations $\mathbf{U}$ into the same latent space as $\mathbf{X}$. The Frobenius norm indicates the distance between the representations of users from two perspectives, which are forced to be close. In this way, the learned users representations $\mathbf{U}$ also capture flat side information contained in $\mathbf{X}$.

However, in practice, $\mathbf{X}$ is usually very sparse. For example, while a user profile could include various types of information, making $\mathbf{X}$ high-dimensional, many users may only provide a part of the profile, which renders $\mathbf{X}$ very sparse. To address this issue, we adopt autoencoders, which provide a way to obtain robust feature representations in an unsupervised manner [3] and have been successfully applied in various tasks such as speech enhancement [16], natural language generation [13] and face alignment [33]. In this work, we choose to incorporate marginalized denoising autoencoders (MDA) into the proposed model, as it is much more computationally efficient than others [4] and we leave incorporating other autoencoders as one future direction.

MDA firstly takes the side information $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$ as input and corrupts the features to obtain noising version $\tilde{\mathbf{x}}_i$ for each user u_i. The corruption process can be done in different ways. In this paper, we follow the practice in [4] and corrupt features by randomly setting each feature to be 0 with the probability $p \geq 0$. In contrast to traditional stacked denoising autoencoders that have the two-level encoder and decoder structures, in MDA, only one single mapping $\mathbf{W}^u : \mathbb{R}^{d_x} \rightarrow \mathbb{R}^{d_x}$ is used to reconstruct the original features and the reconstruction loss is defined as follows:

$$\mathcal{L}(\mathbf{W}^u) = \frac{1}{2n}\sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{W}^u\tilde{\mathbf{x}}_i\|^2 \tag{1}$$

The random corruption of the features may lead to the solution $\mathbf{W}^u$ of high variance. In order to avoid this, k-times corruption is performed and the overall reconstruction loss becomes:

$$\mathcal{L}(\mathbf{W}^u) = \frac{1}{2nk}\sum_{j=1}^{k}\sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{W}^u\tilde{\mathbf{x}}_{i,j}\|^2 \tag{2}$$

where $\tilde{\mathbf{x}}_{i,j}$ denotes the corrupted version of feature $\mathbf{x}_i$ at the j^{th}-time. Written in matrix form, the above loss function can be expressed as:

$$\mathcal{L}(\mathbf{W}^u) = \frac{1}{2nk}\|\bar{\mathbf{X}} - \mathbf{W}^u\tilde{\mathbf{X}}\|_F^2 \tag{3}$$

where $\bar{\mathbf{X}} = [\mathbf{X}, \mathbf{X}, \cdots, \mathbf{X}] \in \mathbb{R}^{d_x \times nk}$ and $\tilde{\mathbf{X}} = [\tilde{\mathbf{X}}_1, \tilde{\mathbf{X}}_2, \cdots, \tilde{\mathbf{X}}_k] \in \mathbb{R}^{d_x \times nk}$ with $\tilde{\mathbf{X}}_j$ denoting the j^{th} corrupted version of the original features $\mathbf{X}$. The solution of the minimization of the loss function defined in Eq (3) can be written in a closed form: $\mathbf{W}^u = \mathbf{K}\mathbf{J}^{-1}$, where $\mathbf{K} = \bar{\mathbf{X}}\tilde{\mathbf{X}}^T$ and $\mathbf{J} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^T$. Ideally, we would like to make infinitely corrupted versions of $\mathbf{X}$ to obtain the most stable mapping $\mathbf{W}^u$. This can be achieved by letting $k \rightarrow \infty$ and computing the expectations of $\mathbf{K}$ and $\mathbf{J}$ [4].

With the obtained mapping layer $\mathbf{W}^u$, robust features of users can be easily constructed from original feature matrix $\mathbf{X}$ by $\mathbf{W}^u\mathbf{X}$.

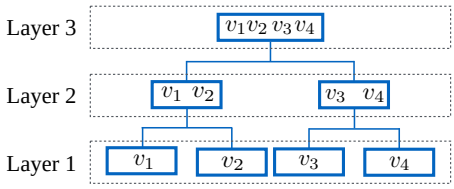


Figure 2: An illustrative example of hierarchical structure.

Hence, the framework that captures the flat side information of users with robust feature mapping becomes:

$$\min_{\mathbf{U},\mathbf{V},\mathbf{S}^u,\mathbf{W}^u} f(\mathbf{U},\mathbf{V}) + \gamma(\|\mathbf{S}^u\mathbf{U}^T - \mathbf{W}^u\mathbf{X}\|_F^2 + \|\bar{\mathbf{X}} - \mathbf{W}^u\tilde{\mathbf{X}}\|_F^2) \tag{4}$$

Similarly, the flat information of items can also be captured as follows:

$$\min_{\mathbf{U},\mathbf{V},\mathbf{S}^v,\mathbf{W}^v} f(\mathbf{U},\mathbf{V}) + \theta(\|\mathbf{S}^v\mathbf{V} - \mathbf{W}^v\mathbf{Y}\|_F^2 + \|\bar{\mathbf{Y}} - \mathbf{W}^v\tilde{\mathbf{Y}}\|_F^2) \tag{5}$$

where $\mathbf{S}^v \in \mathbb{R}^{d_y \times d}$ is the project matrix that projects $\mathbf{V}$ into the same space as $\mathbf{Y}$, $\mathbf{W}^v \in \mathbb{R}^{d_y \times d_y}$ is the mapping layer that obtains robust features from $\mathbf{Y}$, $\bar{\mathbf{Y}} = [\mathbf{Y}, \mathbf{Y}, \cdots, \mathbf{Y}] \in \mathbb{R}^{d_y \times mk}$ and $\tilde{\mathbf{Y}} = [\tilde{\mathbf{Y}}_1, \tilde{\mathbf{Y}}_2, \cdots, \tilde{\mathbf{Y}}_k] \in \mathbb{R}^{d_y \times mk}$ with $\tilde{\mathbf{Y}}_j$ representing the j^{th} corrupted version of $\mathbf{Y}$.

2.3 Incorporating The Hierarchical Information

Unlike flat information, features in hierarchical information are structured. For example, as shown in Figure 1, the genres of books can be organized into a hierarchical structure. It is very likely that books belong to the detailed genres are more similar than those in sub-genres. Thus, it should be desirable to recommend the book that is in the same detailed-genre with the one that has received high rating score from the user. As hierarchical information is intrinsically different with flat information, the approach introduced in the previous subsection is not suitable. Hence, in this subsection, we introduce how to incorporate hierarchical information by extending the basic matrix factorization model. Without the loss of generality, we firstly introduce the approach to incorporate hierarchical side information of items, which can be naturally applied to that of users. Typically, we can use a tree to represent a hierarchical structure. In a tree, each parent node can have different numbers of child nodes, which can also be the parents of the nodes in the next layer. Recall the example given by Figure 1, which shows the hierarchical structure of genres of a book. The genre *Literature & Fiction* is the parent node of several child nodes, such as *GenreFiction*, *Historical Fiction*, etc. The child node *HistoricalFiction* is also the parent of other nodes such as *Political*. The leaf nodes are those who have no child such as *Political*. From this example, it is easily observed that the hierarchical structure can naturally be characterized by the *parent-child* relation. With this intuition, next we introduce how to incorporate the structure information from *parent-child* perspective.

The item characteristic matrix $\mathbf{V} \in \mathbb{R}^{d \times m}$ shows the latent representations of items in a d-dimensional latent space. To incorporate the hierarchical structure, we can further decompose $\mathbf{V}$ into two

matrices $\mathbf{V}^1 \in \mathbb{R}^{m_1 \times m}$ and $\mathbf{V}^{2\prime} \in \mathbb{R}^{d \times m_1}$ such that $\mathbf{V} \approx \mathbf{V}^{2\prime}\mathbf{V}^1$, assuming there are m_1 nodes (or sub-categories) in the second layer. Hence, $\mathbf{V}^1$ indicates the *parent-child* relation between the m_1 categories in the second layer and m items in the first layer. Moreover, $\mathbf{V}^{2\prime}$ gives the latent representations of the m_1 categories. In this way, the latent representation of v_j can be expressed by the j^{th} item's parent categories and their latent representations as $\mathbf{V}^{2\prime}\mathbf{V}^1(:, j)$. If the structured information has more than two layers, we can further decompose $\mathbf{V}^{2\prime}$ such that $\mathbf{V}^{2\prime} \approx \mathbf{V}^{3\prime}\mathbf{V}^2$, where $\mathbf{V}^2 \in \mathbb{R}^{m_2 \times m_1}$ denotes the *parent-child* relation between categories in the third and second layers, respectively. Similarly, $\mathbf{V}^{3\prime} \in \mathbb{R}^{d \times m_3}$ denotes the representations of m_2 categories in the third layer. With this, j^{th} item's representation can be expressed by $\mathbf{V}^{3\prime}\mathbf{V}^2\mathbf{V}^1(:, j)$.

The above process can be repeated q-1 times to capture hierarchical structure of q layers as follows:

$$\mathbf{V} \approx \mathbf{V}^q \cdots \mathbf{V}^3\mathbf{V}^2\mathbf{V}^1 \tag{6}$$

where $\mathbf{V}^i \in \mathbb{R}^{m_i \times m_{i-1}} (1 \le i < q)$, $\mathbf{V}^q \in \mathbb{R}^{d \times m_{q-1}}$ and $m_0 = m$.

The *parent-child* relations indicated by Eq. (6) are implicit and they should be in conformity with explicit ones suggested by hierarchical side information. To achieve this, next we extend Eq. (6) to incorporate the structures in the side information. Let $\mathbf{T}^k \in \mathbb{R}^{m_{k-1} \times m_k}$ indicate the *parent-child* relation between categories in layer k and layer $k+1$ of the hierarchical structure of side information. Specifically, $\mathbf{T}^k(i, j) = 1$ denotes that the i^{th} category in layer k is the child of j^{th} category in layer $k+1$. Figure 2 gives an example, where $\mathbf{T}^2 \in \mathbb{R}^{2\times 1}$ and $\mathbf{T}^2(1, 1) = 1, \mathbf{T}^2(2, 1) = 1$. Thus, the hierarchical structure can be defined by the set $\mathcal{T} = \{\mathbf{T}^1, \mathbf{T}^2, \cdots, \mathbf{T}^q\}$, assuming there are q layers. Intuitively, the presentation of a parent category should be aggregated from those of all the child categories it contains. Thus, a natural way to capture the *parent-child* relations in $\mathcal{T}$ is to make parent representations denoted as $\mathbf{V}^q \cdots \mathbf{V}^i$ to be close to the aggregation of their children's representations denoted as $\mathbf{V}^q \cdots \mathbf{V}^{i-1}$.

In this work, we choose the mean function to be the aggregation function due to its computational efficiency and leave exploring other choices as one future work. Thus, the structure indicated by the item side information can be captured as follows:

$$\min \underbrace{\sum_{i=2}^{q} \|\mathbf{V}^q \cdots \mathbf{V}^i - \mathbf{V}^q \cdots \mathbf{V}^{i-1}\mathbf{Q}^{i-1}\|_F^2}_{f^v(\mathbf{V}^i, \mathbf{Q}^{i-1})}$$

where $\mathbf{Q}^k$ is the normalized version of $\mathbf{T}^k$ and $\mathbf{Q}^k(i, j) = \frac{\mathbf{T}^k(i,j)}{\sum_{i=1}^{m_{k-1}} \mathbf{T}^k(i,j)}$.

In a similar way, we can also incorporate hierarchical information of users as:

$$\min \underbrace{\sum_{i=2}^{p} \|\mathbf{U}^i \cdots \mathbf{U}^p - \mathbf{P}^{i-1}\mathbf{U}^{i-1} \cdots \mathbf{U}^p\|_F^2}_{f^u(\mathbf{U}^i, \mathbf{P}^{i-1})}$$

where $\mathbf{U}^i \in \mathbb{R}^{n_{i-1} \times n_i}$ and $\mathbf{P}^k(i, j) = \frac{\mathbf{C}^k(i,j)}{\sum_{i=1}^{n_{k-1}} \mathbf{C}^k(i,j)}$ is the normalized version of $\mathbf{C}^k \in \mathbb{R}^{n_{k-1} \times n_k}$, which indicates the *parent-child* relationship in hierarchical side information of users. p is number of layers of the hierarchy and $1 \le i < p$.

2.4 The Proposed Framework HIRE

Previous subsections introduce the model components that aim to capture both flat and hierarchical side information. Combining them, the framework HIRE is to solve the following optimization problem:

$$\begin{aligned} \min_{\substack{\mathbf{U}^1, \cdots, \mathbf{U}^p, \mathbf{S}^u, \mathbf{W}^u \\ \mathbf{V}^1, \cdots, \mathbf{V}^q, \mathbf{S}^v, \mathbf{W}^v}} & f(\mathbf{U}, \mathbf{V}) + \alpha f^u(\mathbf{U}^i, \mathbf{P}^{i-1}) + \beta f^v(\mathbf{V}^i, \mathbf{Q}^{i-1}) \\ & + \gamma(\|\mathbf{S}^u\mathbf{U}^T - \mathbf{W}^u\mathbf{X}\|_F^2 + \|\bar{\mathbf{X}} - \mathbf{W}^u\tilde{\mathbf{X}}\|_F^2) \\ & + \theta(\|\mathbf{S}^v\mathbf{V} - \mathbf{W}^v\mathbf{Y}\|_F^2 + \|\bar{\mathbf{Y}} - \mathbf{W}^v\tilde{\mathbf{Y}}\|_F^2) \end{aligned} \tag{7}$$

where $\mathbf{U} = \mathbf{U}^1\mathbf{U}^2 \cdots \mathbf{U}^p$ and $\mathbf{V} = \mathbf{V}^q \cdots \mathbf{V}^2\mathbf{V}^1$. γ and θ control the contribution of flat information, α and β decides the contribution of hierarchical information. Hence, the proposed framework simultaneously models both flat and hierarchical side information with mathematical coherence. Following the tradition [11], we will use the gradient descent method to optimize the formulation of the proposed framework. Next we will use $\mathbf{U}^i$ as one example to illustrate how to get the gradient of parameters due to the page limitation. Before calculating the gradient, we define $\mathbf{A}^i$, $\mathbf{H}^i$, $\mathbf{D}^i$, $\mathbf{G}_i^k$ and $\mathbf{B}_i^k$ that can be used to simplify the expressions:

$$\mathbf{A}^i = \begin{cases} \mathbf{U}^1\mathbf{U}^2 \ldots \mathbf{U}^{i-1}, & \text{if } i \neq 1 \\ \mathbf{I}, & \text{if } i = 1 \end{cases}$$

$$\mathbf{H}^i = \begin{cases} \mathbf{U}^{i+1} \ldots \mathbf{U}^p\mathbf{V}^q \ldots \mathbf{V}^1, & \text{if } i \neq p \\ \mathbf{V}^q \ldots \mathbf{V}^1, & \text{if } i = p \end{cases}$$

$$\mathbf{D}^i = \begin{cases} \mathbf{U}^{i+1}\mathbf{U}^{i+2} \ldots \mathbf{U}^p, & \text{if } i \neq p \\ \mathbf{I}, & \text{if } i = p \end{cases}$$

$$\mathbf{G}_i^k = \begin{cases} \mathbf{U}^{k+1} \cdots \mathbf{U}^{i-1}, & \text{if } k \neq i-1 \\ \mathbf{I}, & \text{if } k = i-1 \end{cases}$$

$$\mathbf{B}_i^k = \mathbf{P}^k\mathbf{U}^k\mathbf{G}_i^k$$

where $1 \le i \le p$ and $1 \le k < i-1$. By dropping irrelevant terms in Eq. (7), remaining terms related to $\mathbf{U}^i$ are as follows:

$$\begin{aligned} \mathcal{L}(\mathbf{U}^i) &= \|\mathbf{M} \odot (\mathbf{R} - \mathbf{A}^i\mathbf{U}^i\mathbf{H}^i)\|_F^2 + \lambda\|\mathbf{U}^i\|_F^2 + \alpha\|(\mathbf{I} - \mathbf{P}^i\mathbf{U}^i)\mathbf{D}^i\|_F^2 \\ &+ \alpha \sum_{k=1}^{i-1} \|(\mathbf{G}_i^k - \mathbf{B}_i^k)\mathbf{U}^i\mathbf{D}^i\|_F^2 + \gamma\|\mathbf{S}^u(\mathbf{A}^i\mathbf{U}^i\mathbf{D}^i)^T - \mathbf{W}^u\mathbf{X}\|_F^2 \end{aligned}$$

Now, we can obtain the gradient of $\mathbf{U}^i$ as:

$$\begin{aligned} \frac{\partial \mathcal{L}(\mathbf{U}^i)}{\partial \mathbf{U}^i} &= \mathbf{A}^{iT}[\mathbf{M} \odot (\mathbf{A}^i\mathbf{U}^i\mathbf{H}^i - \mathbf{R})]\mathbf{H}^{iT} + \lambda\mathbf{U}^i \\ &+ \alpha \sum_{k=1}^{i-1} (\mathbf{G}_i^k - \mathbf{B}_i^k)^T(\mathbf{G}_i^k - \mathbf{B}_i^k)\mathbf{U}^i\mathbf{D}^i\mathbf{D}^{iT} \\ &+ \alpha\mathbf{P}^{iT}(\mathbf{P}^i\mathbf{U}^i\mathbf{D}^i - \mathbf{D}^i)\mathbf{D}^{iT} + \gamma\mathbf{A}^{iT}(\mathbf{A}^i\mathbf{U}^i\mathbf{D}^i\mathbf{S}^{uT} - \mathbf{X}^T\mathbf{W}^u)\mathbf{S}^u\mathbf{D}^{iT} \end{aligned} \tag{8}$$

2.4.1 Time Complexity Analysis. The most expensive operations in the optimization process are updating $\mathbf{U}^i$ and $\mathbf{V}^i$, which in each iteration will cost $O\Big(nn_1n_{i-1} + (2n_in_{i+1} + mm_1)d + n_{i-1}^2(n_i + \sum_{k=1}^{i-1} n_k)\Big)$ and $O\Big(mm_1m_{i-1} + (2m_im_{i+1} + nn_1)d + m_{i-1}^2(m_i + \sum_{k=1}^{i-1} m_k)\Big)$, respectively. Thus, assume N iterations are needed in total, the overall time complexity of the optimization process is $O\Big(\sum_i^p \big(nn_1n_{i-1} +$

$(2n_i n_{i+1}+mm_1)d+n_{i-1}^2(n_i+\sum_{k=1}^{i-1} n_k)\big)+\sum_i^q \Big(mm_1 m_{i-1}+(2m_i m_{i+1}+nn_1)d + m_{i-1}^2(m_i + \sum_{k=1}^{i-1} m_k)\big)\Big)$.

3 EXPERIMENTS

In this section, we firstly introduce the experimental settings. Then, we compare the proposed framework HIRE with representative baselines to answer the first question. Finally, we analyze each model component, which gives answer to the section question. To encourage the reproducible results, we make our code publicly available at: https://github.com/tal-ai/Recommender-Systems-with-Heterogeneous-Side-Information.

3.1 Experimental Settings

We evaluate the proposed framework on three benchmark datasets MovieLens (100K), MovieLens (1M), and BookCrossing and all of them are publicly available [8, 34].

- MovieLens (100K) and MovieLens (1M) are collected from a movie review website[1] where users can give movie rating scores on a scale from 1-5. MovieLens (100k) contains 100,000 ratings from 1000 users on 1700 movies and MovieLens (1M) contains 1 million ratings from 6000 users on 4000 movies. For movies, we use genres as hierarchical information; for users, we use age and gender as flat information and occupation as the hierarchical information.
- BookCrossing is a book rating dataset collected from BookCrossing[2] community and the rating score is from 1 to 10. After basic data cleaning, we get 17028 ratings from 1009 users on 1816 books. For books, we use publish year and author as flat information and publisher as hierarchical information; for users, we use age and location as the flat and hierarchical information, respectively.

For each dataset, we split it into training and test sets such that training set contains $x\%$ of the data and test contains $1 - x\%$. We vary x as $\{40, 60, 80\}$. We choose the commonly used Root Mean Square Error (RMSE) as the measurement metrics of the recommendation performance and lower value of RMSE indicates better performance. In fact, a small improvement in RMSE means a significant improvement of recommender systems [10].

3.2 Recommendation Performance Comparison

In this subsection, we evaluate the recommendation performance of proposed framework by comparing it with following representative baselines:

- **SVD [6]:** It is a matrix factorization technique that factorizes a user-item rating matrix to obtain latent representations of customers and products via singular-value decomposition (SVD). In this method, only rating information is used;
- **NMF [7]:** Non-negative matrix factorization (NMF) is one of the most popular algorithms used in recommender systems. Unlike SVD, it adds non-negative constraints to the latent representations. This method also only uses rating information;
- **I-CF [25]:** This is a item-based collaborative filtering approach that recommends items to users based on the similarity computed from the rating matrix;
- **NeuMF [9]:** NeuMF replaces inner product by combining GMF and MLP neural architectures with sharing embedding layer and is able to significantly improves recommendation performance. In this method, only rating information is used.
- **mSDA-CF [14]:** This method integrates matrix factorization and deep feature learning and achieves state-of-the-art performance. It uses both rating and flat side information and ignores hierarchical one.
- **HSR [31]:** HSR is a state-of-the-art algorithm which is able to capture both rating and structured side information. However, flat information is ignored in this method.

Note that the parameters of all methods are selected through five-fold cross validation and the details of parameter selection of the proposed framework are discussed in the later subsections. we repeat each experiment five times and report the average performance in Table 1. The following observations can be made from the table:

- NeuMF is likely to outperform other traditional CF methods, which suggests the power of deep learning in recommendations. Currently our basic model is based on matrix factorization and it has great potential to choose NeuMF as the basic model to further improve the performance.
- Systems incorporating side information tend to obtain better performance compared to their corresponding systems without side information. This observation supports the importance of side information.
- The proposed framework HIRE achieves the best performance in most of the cases. We contribute the superior performance to its ability to capture both flat and structured side information. More details regarding the contribution of each component will be discussed in following subsection.

With the above observation, we are able to draw a conclusion to answer the first question: the proposed framework that incorporates heterogeneous side information significantly improves the recommendation performance over the state-of-the-art methods. In the next subsection, we will give a detailed analysis of the contribution from flat and hierarchical side information, respectively.

3.3 Component Analysis

In this subsection, we systematically examine the effect of key components by constructing following model variants:

- HIRE-FU: it eliminates the contribution of flat side information of users by setting $\gamma = 0$ in Eq. (7).
- HIRE-FV: it eliminates the contribution of flat side information of items by setting $\theta = 0$ in Eq. (7).
- HIRE-SU: it eliminates the contribution of hierarchical side information of users by setting $\alpha = 0$ in Eq. (7).
- HIRE-SV: it eliminates the contribution of hierarchical side information of items by setting $\beta = 0$ in Eq. (7).

The recommendation performance on MovieLens (100K) is shown in Figure 3. Since we observe similar results on other datasets, we only show that on MovieLens (100K) dataset to save space. From the

[1] https://movielens.org/
[2] http://www.bookcrossing.com/

Table 1: Recommendation performance comparison. All prediction differences between HIRE and other methods are statistically significant.

Methods	MovieLens (100K)			MovieLens (1M)			BookCrossing		
	40%	60%	80%	40%	60%	80%	40%	60%	80%
SVD	1.0152	0.9704	0.9491	0.9161	0.9087	0.8947	4.7746	2.8866	2.0899
NMF	1.0352	0.9955	0.9715	0.9446	0.9293	0.9227	2.9381	2.7832	2.6055
I-CF	1.0601	1.0485	1.0343	1.0229	1.0065	0.9975	2.0216	1.9989	2.2250
NeuMF	1.0928	1.0877	1.0849	0.9872	0.9834	0.9825	2.0191	1.8708	1.8586
mSDA-CF	1.0968	1.0891	1.0792	1.0498	1.0482	1.0466	3.0015	2.1992	1.8692
HSR	0.9879	0.9647	0.9376	0.9074	0.8906	0.8742	4.8821	4.1072	3.6137
HIRE	0.9703	0.9398	0.9243	0.8957	0.8778	0.8607	2.3364	1.9193	1.8432

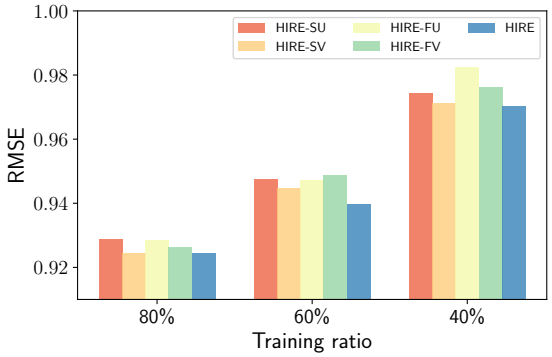
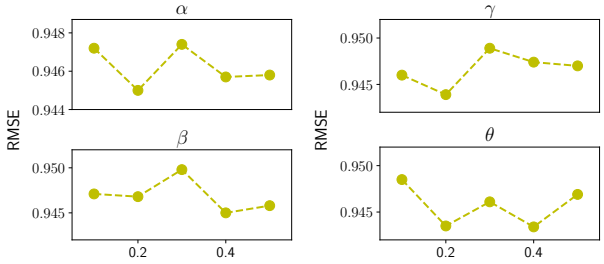


Figure 3: Performance analysis of HIRE with different components.



(a) Performance with α abd β (b) Performance with γ and θ

Figure 4: Parameter analysis with α, β, γ and θ

Figure 3, we can easily observe that HIRE obtains the least RMSE error among all its variants in all cases. This suggests that recommendation performance degrades when ignoring any type of side information. Thus, it is important to incorporate heterogeneous side information in recommender systems.

3.4 Parameter Analysis

In this subsection, we further analyze the sensitivity of the four key parameters γ, θ, α and β that control the contributions of flat side information of users, flat side information of items, hierarchical side information of users, and hierarchical information of items, respectively. In detail, for each of the four parameters, we conduct experiment with the proposed framework by varying the value of it while fixing the others. The performance is shown in Figure 4. Similarly, only performance on MovieLens (100K) is reported due to the space limitation. From both figures, we clearly see that the performance tends to first increase and then decrease, which further supports the importance of side information in recommendations.

4 RELATED WORK

In this section, we give a brief overview of the related recommender systems. A large body of research has been devoted to developing algorithms to improve the performance of recommender systems, which play a crucial role in the increasingly digitalized society. Among them, collaborative filtering based approaches have achieved great success. Roughly, collaborative filtering can be categorized into two type: (1) memory-based approaches [18, 22, 25, 30], which aim at exploring neighborhood information of users or items for recommendation; and (2) model-based methods [7, 11, 17], which try to model the underlying mechanism that governs user rating process. Generally, model-based methods show superior performance than the content-based ones. In particular, Matrix Factorization (MF) based collaborative filtering have gained great popularity due to their high performance and efficiency [11, 12, 20, 24, 26]. Despite of its success, collaborative filtering approaches are known to suffer from data sparsity issues, as the number of items or users is typically very large but the number of ratings is relatively small. One popular way to address this issue is to incorporate the increasingly available side information in the model [2, 5, 15, 28, 29, 31]. The majority of studies exploit either only flat side information [2, 5], or only hierarchical side information [15, 31] due to the challenges brought by the inherent difference between these two types of information. However, our work addresses these challenges and is able to incorporate the two types of information simultaneously.

5 CONCLUSION

In this paper, we investigate the problem of exploiting heterogeneous side information for recommendations. Specifically, we propose a novel recommendation framework HIRE that is able to capture both flat and hierarchical side information with mathematical coherence. Extensive experiments on three benchmark datasets verify the effectiveness of the framework and demonstrate the impact of both flat and hierarchical side information on recommendation performance.

ACKNOWLEDGMENTS

Jiliang Tang is supported by the National Science Foundation (NSF) under grant numbers IIS-1714741, IIS-1715940 and CNS-1815636, and a grant from Criteo Faculty Research Award.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *TKDE* 6 (2005), 734–749.

[2] Gediminas Adomavicius and Alexander Tuzhilin. 2011. Context-aware recommender systems. In *Recommender systems handbook*. Springer, 217–253.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.

[4] Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. *arXiv preprint arXiv:1206.4683* (2012).

[5] Yi Fang and Luo Si. 2011. Matrix co-factorization for recommendation with rich side information and implicit feedback. In *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*. ACM, 65–69.

[6] Gene H Golub and Christian Reinsch. 1970. Singular value decomposition and least squares solutions. *Numerische mathematik* 14, 5 (1970), 403–420.

[7] Quanquan Gu, Jie Zhou, and Chris Ding. 2010. Collaborative filtering: Weighted nonnegative matrix factorization incorporating user and item graphs. In *SDM*. SIAM, 199–210.

[8] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2016), 19.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.

[10] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*. ACM, 426–434.

[11] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[12] Daniel D Lee and H Sebastian Seung. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature* 401, 6755 (1999), 788.

[13] Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057* (2015).

[14] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep collaborative filtering via marginalized denoising auto-encoder. In *CIKM*. ACM, 811–820.

[15] Kai Lu, Guanyuan Zhang, Rui Li, Shuai Zhang, and Bin Wang. 2012. Exploiting and exploring hierarchical structure in music recommendation. In *Asia Information Retrieval Symposium*. Springer, 211–225.

[16] Xugang Lu, Yu Tsao, Shigeki Matsuda, and Chiori Hori. 2013. Speech enhancement based on deep denoising autoencoder.. In *Interspeech*. 436–440.

[17] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. 2008. Sorec: social recommendation using probabilistic matrix factorization. In *CIKM*. ACM, 931–940.

[18] Prem Melville, Raymond J Mooney, and Ramadass Nagarajan. 2002. Content-boosted collaborative filtering for improved recommendations. *Aaai/iaai* 23 (2002), 187–192.

[19] Aditya Krishna Menon, Krishna-Prasad Chitrapura, Sachin Garg, Deepak Agarwal, and Nagaraj Kota. 2011. Response prediction using collaborative filtering with hierarchies and side-information. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 141–149.

[20] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *NIPS*. 1257–1264.

[21] Xia Ning and George Karypis. 2012. Sparse linear methods with side information for top-n recommendations. In *RecSys*. ACM, 155–162.

[22] Alexandrin Popescul, David M Pennock, and Steve Lawrence. 2001. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *UAI*. Morgan Kaufmann Publishers Inc., 437–444.

[23] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2015. Recommender systems: introduction and challenges. In *Recommender systems handbook*. Springer, 1–34.

[24] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML*. ACM, 880–887.

[25] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*. ACM, 285–295.

[26] Nathan Srebro, Jason Rennie, and Tommi S Jaakkola. 2005. Maximum-margin matrix factorization. In *NIPS*. 1329–1336.

[27] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009).

[28] Jiliang Tang, Suhang Wang, Xia Hu, Dawei Yin, Yingzhou Bi, Yi Chang, and Huan Liu. 2016. Recommendation with Social Dimensions.. In *AAAI*. 251–257.

[29] Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-prod2vec: Product embeddings using side-information for recommendation. In *RecSys*. ACM, 225–232.

[30] Jun Wang, Arjen P De Vries, and Marcel JT Reinders. 2006. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR*. ACM, 501–508.

[31] Suhang Wang, Jiliang Tang, Yilin Wang, and Huan Liu. 2018. Exploring Hierarchical Structures for Recommender Systems. *TKDE* 30, 6 (2018), 1022–1035.

[32] Jie Yang, Zhu Sun, Alessandro Bozzon, and Jie Zhang. 2016. Learning hierarchical feature influence for recommendation by recursive regularization. In *RecSys*. ACM, 51–58.

[33] Jie Zhang, Shiguang Shan, Meina Kan, and Xilin Chen. 2014. Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment. In *ECCV*. Springer, 1–16.

[34] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *WWW*. ACM, 22–32.

Globally-Optimized Realtime Supply-Demand Matching in On-Demand Ridesharing

Yifang Liu
Smule
ifnliu@gmail.com

Will Skinner
Engine ML
will@engineml.com

Chongyuan Xiang
Opendoor Labs
xiangcy93@gmail.com

ABSTRACT

In on-demand ridesharing, optimizing the quality of supply-demand matches and minimizing the number of unfulfilled trip requests is an important business problem. One approach is to match supply with demand based on greedy local heuristics, which is sub-optimal at the market level. Another approach is to find the globally optimal matching over the whole marketplace. However, because the computation has high latency, and usually requires aggregate supply and demand data over time, instead of only the supply and demand at the time of individual trip requests, it is unfeasible to perform the global optimization for every trip request in realtime. This paper proposes a solution that performs a global optimization offline periodically based on forecasted supply and demand data, and uses the offline results to guide realtime supply and demand matching. The solution is a novel two-stage robust optimization scheme for supply-demand matching in on-demand ridesharing. We implemented the proposed solution and evaluated it using simulated trips based on public industry data. It reduces the number of unfulfilled trips significantly compared to a state-of-the-art approach.

KEYWORDS

Ridesharing, Supply-Demand Matching, Robust Optimization

ACM Reference Format:
Yifang Liu, Will Skinner, and Chongyuan Xiang. 2019. Globally-Optimized Realtime Supply-Demand Matching in On-Demand Ridesharing. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313579

1 INTRODUCTION

A core value in the on-demand sharing economy is to effectively match supply and demand in realtime. Improving supply-demand matching can increase the utilization of suppliers and therefore reduce the service price. Through network effects, lower prices and higher utilization can help expand both supply and demand further in the market.

In this paper, we will focus on ridesharing, which has been used by hundreds of millions of people across the world. In this context, a supply unit stands for a driver, and a demand unit stands for a rider. How to match riders and drivers is at the core of a successful ridesharing platform's marketplace efficiency optimization.

The primary goal of supply-demand matching is to minimize the total unfulfilled demand. This calls for a non-trivial global optimization for trip-level supply-demand matching over all individual units of demand and supply in the whole geospace of a market (e.g. San Francisco Bay Area or Greater Los Angeles). However, in practice, a realtime trip request usually requires a matching decision to be made within only a few seconds. The global supply-demand matching optimization is often too complicated to finish in such a small amount of time.

As a result, one of the main limitations existing matching systems often have is the sacrifice of global optimality for quick suboptimal matching. They usually use local, greedy heuristics to perform supply-demand matching at individual trip request level, which often yield sub-optimal matching solutions, and cause high volumes of unfilled trip requests.

Another usual limitation of existing matching systems is the handling of future supply and demand. Existing matching methods often rely on point estimation, i.e., without considering the probability distribution of the predicted values. This is a main source of sub-optimal matching solutions.

To tackle the two challenges above, we propose a new approach, Globally-Optimized Realtime Matching (GORM). The algorithm has three main steps. First, we use historical time series data to predict the probabilistic distribution of supply and demand over the whole marketplace in the next 5 minutes. Second, we perform global optimization of supply-demand matching at an aggregated level over the whole market for the next 5 minutes using the forecasted data. To cope with the forecasting error, our global optimization will also find supplies to match the unpredicted extra true demands. Third, we use the pre-computed global optimization result to guide a local heuristic for realtime per-trip supply-demand matching.

In order to keep computational complexity low and maintain reasonable signal-to-noise ratios, we use hexagons (about 600 meters in diameter), as the smallest unit of geo area. For hexagonal geofences, we use Uber's open source library, H3 [3, 11].

The main contributions of this paper include:

(1) We present a new framework (GORM) for realtime efficient supply-demand matching for on-demand ridesharing. GORM combines periodical (offline) global optimization for market-wide aggregate supply-demand matching and realtime (online) local heuristics for per-trip supply-demand matching.
(2) We present a novel linear programming formulation for robust optimization that is resilient to the uncertainty in the underlying market data.
(3) We tested the proposed solution with simulated market dynamics based on real-world ridesharing data, and showed over 7-22% reduction on unfulfilled demand compared with a state-of-the-art supply-demand matching algorithm.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313579

2 RELATED WORK

Because supply-demand matching is very important to the ridesharing business, there has been much related work on this topic.

One type of previous work focused on the realtime factor of the problem, and proposed to greedily find the nearest drivers for every incoming trip request [1, 4, 5, 8, 9]. Those approaches were likely to find suboptimal solution in terms of overall market efficiency.

Another type of previous work used more complex algorithms to find a globally more optimized matching of supplies and demands. The authors of the Didi Chuxing's Taxi order dispatch system [12] proposed a combinatorial optimization approach. They first predicted riders' destinations based on historical data, then estimated drivers' order acceptance rate based on the predictions, and finally searched for a globally optimized matching based on the above estimation. The main drawback of this approach was that it was hard to do such a large scale computation in realtime. Also, this approach did not put dispatch radius as a constraint, which could lead to long dispatch time and unsatisfactory rider experience. Another work [10] proposed an agent based model called NTuCab, where each agent processed N rider/driver pairs, and jointly minimized the rider waiting time. However, this approach did not try to minimize unfulfilled demands and could lead to suboptimmal market efficiency.

3 THE GORM FRAMEWORK

This section explains the globally-optimized realtime matching (GORM) framework for the global matching problem. The effectiveness of the GORM framework is related to the following aspects:

1) Forecasting the spatial and temporal distribution of demand and supply across the whole market's geo space over a short time period, e.g., 5 minutes in our case. The aggregate demand/supply forecast is always a probability distribution, not a point estimate. The GORM framework relies on the probabilistic estimation as the input to the global optimization problem.

2) Computing the globally-optimized matching solution into the future, based on the forecasts of demand and supply - their probability distribution. The global optimization needs to be resilient to the prediction error in the forecasts. GORM achieves this by optimizing the expected value of the cost function over the probabilistic demand/supply forecasts.

3) Performing the local trip-level matching, which is a heuristic guided by the forward-looking global-optimization result. There is a time constraint on the local matching decision. This way, each trip request can be matched to a supply in seconds, while the decision is aligned with the globally-optimized supply-demand matching at aggregate market level.

The GORM framework combines the forward-looking global matching optimization and the realtime local matching heuristic, by performing the 3 steps in Algo 1 for all the supply-demand matching decisions across the whole market over every 5-minute time window.

In the GORM framework, Step 1 can be performed with an existing time series forecasting algorithm, e.g., multivariate ARIMA. Since it is a relatively mature practice with off-the-shelf statistical tools, we will skip the explanation on it, and spend time on Steps 2 and 3 in the main body of the GORM framework.

ALGORITHM 1: Globally-Optimized Realtime Matching

input : the 5-minute time window to perform supply-demand matching over; all the hexagons in the whole marketplace

output : at the hexagon aggregate level,globally-optimized inter-hexagon supply relocating flow $x_{i,j}, \forall i, j \in \{hexagons\}$; at individual request level, a driver (if feasible) being matching to a trip request.

1 Step 1) Perform time series forecasting (with confidence intervals) for demand and supply in the next 5 minutes in the whole market, i.e., obtaining their confidence intervals ($\alpha = 0.05$): $\forall i \in \{hexagons\}, [T_i^l, T_i^u], [S_i^l, S_i^u]$ for the next 5 minutes;

2 Step 2) Perform optimization on the global matching problem, taking as the input the demand/supply forecasting across the whole market over the next 5-minute window, i.e., $\forall i \in \{hexagons\}, [T_i^l, T_i^u], [S_i^l, S_i^u]$. As a result, we solve the problem for $\forall i, j \in \{hexagons\}, x_{i,j}$ optimizing the whole market's efficiency over the next 5 minutes;

3 Step 3) Perform realtime supply-demand matching for each individual trip request, based on the globally optimized inter-hexagon supply flows from Step 2. Basically a demand in hexagon j is preferred to be matched to a supply from hexagon I with the highest supply-to-demand ratio among all corresponding candidate hexagons $\forall i$ where $x_{i,j} > 0$.;

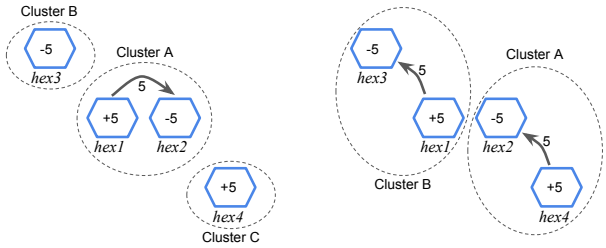


Figure 1: Supply-Demand Matching with Different Hexagon Clusters.

4 GLOBAL MATCHING

This section explains the global supply-demand matching, including its optimization goal, geospatial constraints, its implications for the ridesharing business, and our novel technique for handling the uncertainty in market forecasts, which are used as input to the global optimization problem.

An accurate yet compact representation of the spatial distribution of supply and demand requires dividing the whole market geospace into small basic geo units. Among all commonly-used shapes of the basic geo units, hexagons are our choice in solving this problem, because of their geometrical consistency, good coverage over the whole surface of the earth, and its roundness. Next we illustrate how global optimization can make a big difference in trying to achieve high market efficiency. In the following example, we consider two quite extreme ways of supply-demand matching:

1) matching individual demand to currently nearby supply by local greedy heuristics, without considering their interaction with other demand and supply further in the whole market; 2) matching all demand to all supply in the whole geo space of the marketplace across a certain time period, while minimizing the total unfulfilled demand globally.

In Fig. 1, we use a spatial cluster (i.e., Cluster A, B, or C in the example) to depict a group of one or more hexagons, between which supplies are matched to demands. There are a total of 4 hexagons in the whole marketplace, each of which is associated with the amount of supply units subtracting the amount of demand units. $hex2$ and $hex3$ both have 5 units of demand and no supply. $hex1$ and $hex4$ both have no demand and 5 units of supply. The spatial clusters on the left give an example of local greedy supply-demand matching. When demands in $hex2$ occur, we will match them with supplies from $hex1$ because the clusters are closest to each other. After that, when demand starts to surface in $hex3$, there are no supplies in Cluster B to match. Moreover, $hex1$ already run out of available supplies, and $hex4$ is too far away from $hex3$. Therefore, the system can only fulfill 5 out of 10 demand units. In contrast, the spatial clusters on the right side of the figure give an example of better globally optimized supply-demand matching in the same market scenario. In this case, we use $hex1$ to supply $hex3$, and $hex4$ to supply $hex2$. Therefore, all 10 demand units are fulfilled.

Now it is clear that the supply-demand matching problem needs a global optimization, we start to explain the problem and considerations around it. The input to our global matching optimization problem is a forecast of the demand and supply volumes across all individual hexagons in the whole market over a period of time (5 minutes in our case).

The primary goal of our matching problem is to minimize the total unfulfilled demand over all hexagons, i.e., minimize: $\sum_i \max(T_i - S_i, 0)$, where T_i denotes the volume of demand in hexagon i; and S_i denotes the total volume of supply in hexagon i, which is the summation of the supply originating from hexagon i and the supply relocated from other hexagons to hexagon i.

This goal is achieved by optimizing the supply movement flow between different pairs of hexagons. That is, the decision variable $x_{i,j}$ indicates the volume of supply that is relocated from hexagon i to hexagon j, in order to fulfill the demand in hexagon j. The supply movement from hexagon i to hexagon j effectively means dispatching the corresponding amount of supply in hexagon i to demand in hexagon j.

Because the forecast is subject to prediction error, our optimization problem formulation needs to take into account the uncertainty in the forecast. For the clarity of explanation, the rest of this section will describe our deterministic optimization formulation (without considering forecasting error) first, and then present the robust optimization formulation for coping with forecasting uncertainty.

4.1 The deterministic optimization

Before diving into the global optimization problem formulation, we explain a geospatial constraint on the supply movement. Supply can only move between two hexagons if they are within a dispatch radius to each other. This constraint is very important to a real-world on-demand ridesharing service: a unit of supply should be matched to a nearby unit of demand, to keep the estimated time to pickup (ETP) reasonably short for realtime trip requests.

The global matching optimization can be formulated into the following linear programming problem.

$$
\begin{aligned}
\text{Min: } & f(x) = \alpha \sum_i h_i \\
\text{S.t.: } & \forall i \in \mathbb{A}, h_i \geq T_i - s_i, h_i \geq 0 \\
& \forall i \in \mathbb{A}, s_i = S_i - \sum_j x_{i,j} + \sum_k x_{k,i} \\
& \forall i \in \mathbb{A}, \sum_j x_{i,j} \leq S_i \\
& \forall i,j \in \mathbb{A}, Dist_{i,j} \leq O, x_{i,j} \geq 0, x_{j,i} \geq 0
\end{aligned} \tag{1}
$$

where $\mathbb{A}$ denotes the set of all hexagons in the whole market; $x_{i,j}$ is a decision variable representing the supply volume flowing from hexagon i to hexagon j; h_i is an auxiliary variable representing the unfulfilled demand units in hexagon i; s_i is an auxiliary variable representing the total volume of supply in hexagon i (including those originated in hexagon i and those moved in-and-out from/to other hexagons); O denotes the maximum distance between two hexagons $Dist_{i,j}$ that allows the inter-hexagon supply flows.

4.2 The robust optimization

Because forecasts are inherently uncertain, the optimization process must be robust to actual supply and demand being different from what was forecasted. Furthermore, we must deal with prediction errors in a manner that is consistent with the overall optimization. This is similar to a 2-stage stochastic programming problem in general [2], which can be formulated as follows in our case.

$$
\begin{aligned}
\text{Min}_x\text{: } & g(x) = f(x) + E_\zeta[Q(x,\zeta)] \\
\text{S.t.: } & Q(x,\zeta) = \min_y \left(q(y,\zeta) = T(\zeta)x + W(\zeta)y \right)
\end{aligned} \tag{2}
$$

where ζ represents the vector of one possible combination of true supply and demand volumes over all hexagons in the marketplace; $Q(x,\zeta)$ denotes the optimal value of the second stage problem on a specific value of ζ; y indicates the recourse decision (matching on individual trip request level in our case) in the face of ζ; $E_\zeta[Q(x,\zeta)]$ is meant to capture the expected value of cost by all possible true volume values of demand and supply over all hexagons (there is a probability distribution over these possible values).

In our case, we create an uncertainty-resilient linear programming formulation for our matching problem. Basically, the robust optimization formulation wants to minimize the expected unfulfilled demand over the probability distribution of possible volume values given by the confidence interval of the market forecasting. We choose to use 95% confidence interval, corresponding to $\alpha = 0.05$, in the forecast of the market in the next 5 minutes.

In its essence, the robust optimization part of the formulation tries to approximate different probabilities of an amount of supply occurring in a hexagon in reality with different unit costs (they are set as problem parameters) of using the corresponding supply, i.e., matching the supply with some demand. Similarly, the unfulfilled demand's probabilistic unit costs are formed this way.

This is done by setting the unit cost of supply proportional to $1 - CDF$ (the reverse cumulative distribution function) at the volume range corresponding to the supply being charged. Let's use a simplified made-up example to see how this works. Assume our forecast with confidence interval says that S_i, the value of the supply volume originates in hexagon i, has 5% chance to randomly fall at somewhere between 1 and 10, while the chance for the value to take a random place between 11 and 15 is 95%. Now if we use supply within the [1,10] range, on average 97.5% of the time, we will be able to find the supply in the true market reality, because when S_i takes a value in [11,15] by 95% chance, we are warranted with obtaining supply from the first 10 of them. Therefore, we assign the unit cost of supply in [1,10] with $\beta(1-97.5\%)$. By similar reasoning, we can assign the unit cost of supply in [11,15] with $\beta(1-47.5\%)$.

This way, the linear relation between the cost and the probability of forecasted demand/supply approximates the expected value of the uncertain cost term $E_\zeta[Q(x,\zeta)]$, and our robust optimization formulation approximates the stochastic programming formulation given in Equ. 2.

In practice, without loss of generality, we simplify the forecast probability distribution to being piece-wise uniform, just like in the simplified example above. The approximation of more sophisticated forecast probability distribution can be done in similar manner.

$$
\begin{aligned}
\text{Min: } & g(x) = \alpha \sum_i h_i + \sum_i \beta_i \tilde{s_i} + \sum_i \gamma_i \tilde{h_i} \\
\text{S.t.: } & \forall i \in \mathbb{A}, h_i \geq T_i - s_i, h_i \geq 0 \\
& \forall i \in \mathbb{A}, s_i = S_i - \sum_j x_{i,j} + \sum_k x_{k,i} \\
& \forall i \in \mathbb{A}, \sum_j x_{i,j} \leq S_i \\
& \forall i,j \in \mathbb{A}, Dist_{i,j} \leq O, x_{i,j} \geq 0, x_{j,i} \geq 0 \\
& \forall i \in \mathbb{A}, h'_i \geq (T_i + T'_i) - (s_i + s'_i), h'_i \geq 0 \\
& \forall i \in \mathbb{A}, \tilde{h_i} \geq h'_i - h_i, \tilde{h_i} \geq 0 \\
& \forall i \in \mathbb{A}, s'_i = \tilde{s_i} + s_i \\
& \forall i \in \mathbb{A}, 0 \leq \tilde{s_i} \leq S'_i
\end{aligned} \tag{3}
$$

where $x_{i,j}$ and h_i remains their representation as in the deterministic optimization formulation Equ. 1; T_i and S_i are the lower bounds T_i^l, S_i^l of the confidence intervals of the predicted demand and supply, $[T_i^l, T_i^u]$, $[S_i^l, S_i^u]$, respectively. T'_i and S'_i are the spans of the confidence intervals of the predicted demand and supply, $T_i^u - T_i^l$ and $S_i^u - S_i^l$, respectively. As a result, $\tilde{t_i}$ and $\tilde{s_i}$ represent the usage of demand and supply in their corresponding confidence intervals, respectively.

5 REALTIME MATCHING

Realtime matching at the individual request level corresponds to the second stage decisions in a typical 2-stage stochastic programming. It is a simple, fast heuristic that uses the globally-optimized inter-hexagon supply flows (which are obtained from the global matching) to guide the realtime dispatch for individual demand. This way, our final matching on individual trip requests can leverage the globally-optimized uncertainty-resilient aggregate matches, while being very runtime efficient so as to comply to the realtime matching time limit.

More specifically, for each trip request, the heuristic consists of two main operations.

(1) *Request matching guided by global optimization result*: for each trip request in hexagon j, collect the candidate hexagon set that contains all hexagon i that either is hexagon j itself or having a positive supply flow to j, i.e., $x_{i,j} > 0$, according to the current value of the global optimization result $x_{i,j}$. Rank the candidate hexagons by their supply-to-demand ratio in the order from high to low, and go through the candidate list from top to bottom while trying to find the hexagon to dispatch a supply in it to the trip request.
(2) *Request matching updating the global optimization result*: with the selected hexagon i for the trip request, update the corresponding supply flow value by: $x_{i,j} = x_{i,j} - 1$.

6 EXPERIMENT

This section evaluates the effectiveness and efficiency of GORM's application on the globally-optimized realtime matching problem. We compare GORM against several existing matching heuristics and state-of-the-art matching optimization algorithms. Two publicly-available raw input data sources are used in generating the simulated demand/supply distribution data for the experiments: 1) Uber passenger pickup data in New York City [6], short named pickup data; 2) Google Maps street speed data in New York City [7], short named speed data.

The snapshots of the demand and supply distribution over the marketplace of NYC are generated in 4 parts based on the raw input data: 1) the matched demand in reality, 2) the matched supply in reality, 3) the unmatched demand estimated by simulation, and 4) the unmatched supply estimated by simulation. Parts 1 and 2 are obtained directly from the pickup data, except that the dispatch-time location of a matched supply is randomly assigned within the dispatch radius from the pickup location. Part 3 is estimated by simulation based on the street speed at the corresponding location. Basically, the slower the driving speed around the pickup location is, the heavier the traffic around the location is, and likely the more demand is around the location. Part 4 is estimated based on the volume of the unmatched demand. If there is unmatched demand, the surrounding area should have no unmatched supply; otherwise, the unmatched supply is randomly assigned to locations in proportion to the nearby pickups.

The experiments are performed over the simulated snapshots of demand and supply distribution in 5-minute windows over the marketplace of different sizes. Three sub-regions of NYC in different sizes - mimicking marketplace of different geo sizes - were used in the experiments: city-big with $80k$ hexagons, city-medium with $40k$ hexagons, and city-small with $20k$ hexagons, respectively.

The experiments are set up as follows: For each sub-region of the city, snapshots of 5-minute market dynamics are used as the test time windows. Inside each 5-minute window, individual units of demand and supply surface and expire in their respective time limits. For example, individual demand's expiration time, the realtime supply-demand matching time limit, is set to 10, 15, 20, 25 and 30 seconds in the experiments.

The simulated supply and demand data of each of the 5-minute windows are available to serve as the "historical" data for the market forecasting in the corresponding 5-minute window, should it be needed by an algorithm in the experiment. For simplicity, we assume the chosen matching algorithm will not significantly change supply and demand at the hexagon level over 5-minute windows. This allows us to use the simulated data without running a true experiment with interventions.

We experiment and observe how the matching solution quality responds to changes on the problem parameters, including the error in demand/supply forecasts, realtime matching time limit. On the other hand, some problem parameters are fixed for all experiments. For example, the maximum-allowed geospatial distance in dispatching a supply in a hexagon to another hexagon is 6 hexagons away.

6.1 The algorithms in the experiments

We test and compare our uncertainty-resilient globally-optimized realtime supply-demand match method, GORM, with three baseline algorithms. The three baseline algorithms represent greedy local matching heuristics, and global matching that needs more than a few seconds' runtime to come up with the solution, global matching without treatment to deal with the uncertainty in the market forecasts, respectively. They are briefly described as follows.

(1) Baseline algorithm 1 (LOC-GRD): Nearest neighbor dispatch with a preference of matching supply in hexagons having higher supply-to-demand ratio. This represents greedy, local matching heuristics. While missing global optimization, this method satisfies the realtime matching requirement with ease. It does not need market forecast, and thus it does not face the challenge of prediction uncertainty.
(2) Baseline algorithm 2 (GLO-HIL): A modified implementation of Didi Chuxing's hill-climbing optimization method, which is held against the realtime matching time constraints. Because this method does not depend on a forecast of the market, all the demand/supply considered and optimized by this method is limited to which have occurred in the corresponding matching time window. Thus, the maximum runtime of the optimization algorithm is the matching time limit, and the best solution obtained by the stopping point is used as the matching solution.
For example, if the realtime matching time limit is set to be 15 seconds, then the 5-minute experiment window is divided into 20 15-second sub-windows, during each of which the matching algorithm is performed among all demand and supply occurring in that sub-window, and the optimization has at most 15 seconds to run. This represents globally-optimized matching that usually makes pre-mature sub-optimal matching decision, due to its extensive runtime for matching every trip request with an open supply in a limited market scope. Note that because our supply-demand matching problem ensures a close-to-100% order acceptance rate by the supply, a prediction of driver acceptance rate is unnecessary. For the simplicity of demonstration, we enforce the supply acceptance in all of our supply-demand matching decisions.
(3) Baseline algorithm 3 (GLO-DET): Globally-optimized realtime matching without robust optimization, i.e., global deterministic optimization. This method represents global optimization driven realtime matching, which depends on a market forecast in point estimation style, but does not consider or optimize over prediction error.
(4) Our algorithm (GLO-OUR): Globally-optimized uncertainty-resilient realtime matching. This represents our algorithm that depends on a market forecast with a 95% confidence interval.

6.2 Solution quality evaluation

The solution quality is primarily measured by the total volume of unfulfilled demand (**UFD**), defined as $UFD = \sum_i \max(T_i - S_i, 0)$.

To calculate the theoretical minimum total unfulfilled demand (UFD) for the matching solution, we run the global optimization formulated as Equ. 1 over the true market demand and supply dynamics that is given by the simulation in the corresponding 5-minute window in the market. Summing up the unfulfilled demand over all hexagons in the optimization result gives us the theoretical minimum value of the total unfulfilled demand (UFD), which is the lowest possible UFD for the matching problem.

The effectiveness of a matching algorithm can be better measured in UFD relative to the theoretical minimum UFD. The relative UFD (rUFD) is simply the difference between the UFD yielded by an algorithm and the theoretical minimum UFD. In other words, rUFD is the UFD gap between an algorithm's solution and the ideal best solution.

Next, we review how the 4 algorithms in our experiments perform with their matching solutions. Change problem parameters: demand/supply perturbation, different city sub-regions, and realtime decision time constraint - algorithm runtime limit.

In the following result review, we will use the short names for the 4 algorithms in our experiments. All metric values in the following tables is the percentage of the measurement over the total volume of demand in the whole market. The UFD values from all 4 algorithms are measured as the average over the same 5 snapshots of 5-minute market. Note that the lower the UFD and rUFD values are, the better; the higher the Fulfillment values are, the better.

The columns labeled with "abs" and "relative" indicate the measurement itself and the percentage of its delta relative to the same measurement of the first baseline algorithm (LOC-GRD), respectively. The column named as "Min UFD" indicates the theoretically lowest UFD possible.

Overall, the improvements on rUFD achieved by GORM over the base algorithms range from 24% to 60%. Given the usual unfulfilled demand percentage shown in the experiments, this rUFD improvement by our method translates into roughly $3-7\%$ increase on the total fulfilled trips. More detailed solution-quality experiment result on the 3 sub-regions can be found in Tables 1, 2, and 3.

Now we observe how different algorithms respond to the change of problem parameters, namely the uncertainty in demand/supply prediction (perturbed by $0.5\sigma, 1.0\sigma, 3.0\sigma$) and the realtime matching time limit (10, 15, 20, 25, or 30 seconds).

Figure 2 shows our algorithm GLO-OUR and GLO-DET's reaction to forecast uncertainty change in city-medium. GORM's

Table 1: UFD: 5-minute matching in city-small

	Min UFD	UFD		rUFD		Fulfillment	
		abs	relative	abs	relative	abs	relative
LOC-GRD	3.51%	15.07%	↓ 0%	11.56 %	↓ 0%	84.93%	↑ 0%
GLO-HIL	3.51%	13.66%	↓ 9.36%	10.15%	↓ 12.19%	86.34%	↑ 1.7%
GLO-DET	3.51%	11.09%	↓ 26.41%	7.58%	↓ 34.43%	88.91%	↑ 4.7%
GLO-OUR	3.51%	8.64%	↓ 42.67%	5.13%	↓ 55.62%	91.36%	↑ 7.6%

Table 2: UFD: 5-minute matching in city-medium

	Min UFD	UFD		rUFD		Fulfillment	
		abs	relative	abs	relative	abs	relative
LOC-GRD	5.83%	22.52%	↓ 0%	16.69%	↓ 0%	77.48%	↑ 0%
GLO-HIL	5.83%	21.69%	↓ 3.69%	15.86%	↓ 4.97%	78.31%	↑ 1.07%
GLO-DET	5.83%	19.89%	↓ 11.68%	14.06%	↓ 15.76%	80.11%	↑ 3.39%
GLO-OUR	5.83%	18.51%	↓ 17.81%	12.68%	↓ 24.03%	81.49%	↑ 5.18%

Table 3: UFD: 5-minute matching in city-big

	Min UFD	UFD		rUFD		Fulfillment	
		abs	relative	abs	relative	abs	relative
LOC-GRD	6.88%	11.93%	↓ 0%	5.05%	↓ 0%	88.07%	↑ 0%
GLO-HIL	6.88%	11.63%	↓ 22.10%	4.75%	↓ 5.94%	88.37%	↑ 0.34%
GLO-DET	6.88%	10.34%	↓ 30.74%	3.46%	↓ 31.49%	89.66%	↑ 1.81%
GLO-OUR	6.88%	8.91%	↓ 49.03%	2.03%	↓ 59.80%	91.09%	↑ 3.43%

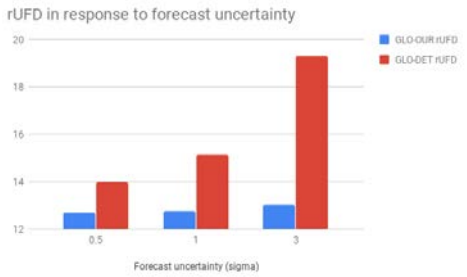


Figure 2: rUFD in response to forecast uncertainty

uncertainty resiliency keeps its solution from significant degradation, like what GLO-DET faces. GORM's fundamental advantage of being resilient to uncertainty usually expands as the uncertainty in the demand/supply forecasting increases.

Figure 3 shows our algorithm GLO-OUR and GLO-HIL's reaction to matching time limit change in city-medium. GLO-HIL's temporally local view of the market cannot improve the solution all the way to GLO-OUR's level, despite the extended optimization runtime. GORM's advantage over baseline algorithms usually expands when the time limit on realtime decision is lower (or equivalently when the trip expiration time is lower).

6.3 Algorithm Runtime

Algorithms in the experiment ran on a single 16-core Xeon machine. The runtime evaluation of all the algorithms is done in two runtime components: 1) offline market forecasting and optimization happening once every 5 minutes, if applicable to a specific algorithm. 2) online supply-demand matching at trip request level happening at realtime upon individual trip requests.

In all experiments, our algorithm - GLO-OUR - always finishes the offline forecasting and optimization under 2 minutes, while

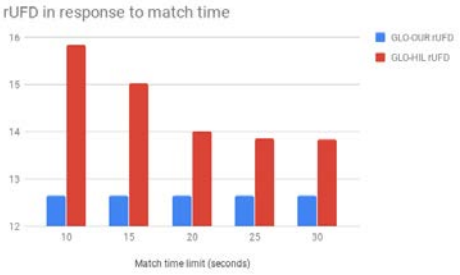


Figure 3: rUFD in response to matching time limit

takes less than a second to perform the realtime matching. Our algorithm performs both parts with less than 2 GB of memory.

More details of the runtime of all 4 algorithms can be found at Table 4. All the runtime measurements are in seconds.

Note that the runtime of the offline component cannot exceed 5 minutes; the runtime of the online component cannot exceed the matching time limit, which is set to 10 seconds in the solution quality experiments.

Notice that GLO-HIL's online component runtime is always 10 seconds, because it needs more than 10 seconds to complete a fully-optimized solution, but has to be cut off at the realtime matching time limit - 10 seconds.

Table 4: Algorithm runtime in seconds

	small city		medium city		big city	
	offline	online	offline	online	offline	online
LOC-GRD	0	<1	0	<1	0	<1
GLO-HIL	0	10	0	10	0	10
GLO-DET	52	<1	63	<1	79	<1
GLO-OUR	61	<1	76	<1	86	<1

7 FUTURE WORK

There are a range of algorithm/model improvements that can be considered in the GORM framework. Some examples include:

(1) Time-series optimization with minute-by-minute forecast.
(2) Global optimization on sliding time windows.

8 CONCLUSION

This paper presents the Globally-Optimized Realtime Matching framework (GORM) for supply-demand matching in ridesharing market. By applying GORM to a real world ridesharing matching problem, we showed significant improvements on market efficiencies - minimizing unfulfilled trip requests. Compared to several representative existing methods in the field, our method has the advantages of realtime matching runtime performance, globally-optimized supply-demand matching, and uncertainty-resilient robust optimization for forward-looking market optimization.

REFERENCES

[1] G. R. Andrey Glaschenko, Anton Ivaschenko and P. Skobelev. Multi-agent real time scheduling system for taxi companies. In *8th International Conference on Autonomous Agents and Multiagent Systems*, pages 29–36, 2009.

[2] J. Birge. State-of-the-art-survey'stochastic programming: Computation and applications. In *Informs Journal on Computing*, pages 111–133, 1997.
[3] I. Brodsky. H3: Uber hexagonal hierarchical spatial index. In *https://eng.uber.com/h3/*, 2018.
[4] L. C. Chung. Gps taxi dispatch system based on a* shortest path algorithm. In *Ph.D. Dissertation Submitted to the Department of Transportation and Logistics at Malausia University of Science and Technology (MUST)*, 2005.
[5] R. C. Der-Horng Lee, Hao Wang and S. Teo. Taxi dispatch system based on current demands and real-time traffic conditions. In *Transportation Research Record: Journal of the Transportation Research Board 1882*, pages 193–200, 2004.
[6] FiveThirtyEight. Uber trip data from a freedom of information request to nyc's taxi & limousine commission. In *https://github.com/fivethirtyeight/uber-tlc-foil-response*, 2015.
[7] Google. Google maps. In *https://www.google.com/maps*, 2019.
[8] H. K. Y. Y. P. K. J. Lee, G.L. Park and S. Kim. A telematics service system based on the linux cluster. In *International Conference on Computational Science - Springer*, pages 660–667, 2007.
[9] N. H. D. Kiam Tian Seow and D.-H. Lee. A collaborative multiagent taxi-dispatch system. In *IEEE Transactions on Automation Science and Engineering*, pages 607–616, 2010.
[10] K. T. Seow, N. H. Dang, and D. Lee. A collaborative multiagent taxi-dispatch system. *IEEE Transactions on Automation Science and Engineering*, 7(3):607–616, July 2010.
[11] Uber. Hexagonal hierarchical geospatial indexing system. In *https://uber.github.io/h3/*, 2018.
[12] L. Zhang and et al. A taxi order dispatch model based on combinatorial optimization. In *International Conference on Knowledge Discovery and Data Mining*, pages 2151–2159, 2017.

MCVAE: Margin-based Conditional Variational Autoencoder for Relation Classification and Pattern Generation*

Fenglong Ma
University at Buffalo, Buffalo, NY
fenglong@buffalo.edu

Yaliang Li
Alibaba Group, Bellevue, WA
yaliang.li@alibaba-inc.com

Chenwei Zhang
University of Illinois at Chicago, Chicago, IL
czhang99@uic.edu

Jing Gao
University at Buffalo, Buffalo, NY
jing@buffalo.edu

Nan Du
Tencent Medical AI Lab, Palo Alto, CA
ndu@tencent.com

Wei Fan
Tencent Medical AI Lab, Palo Alto, CA
davidwfan@tencent.com

ABSTRACT

Relation classification is a basic yet important task in natural language processing. Existing relation classification approaches mainly rely on distant supervision, which assumes that a bag of sentences mentioning a pair of entities and extracted from a given corpus should express the same relation type of this entity pair. The training of these models needs a lot of high-quality bag-level data. However, in some specific domains, such as medical domain, it is difficult to obtain sufficient and high-quality sentences in a text corpus that mention two entities with a certain medical relation between them. In such a case, it is hard for existing discriminative models to capture the representative features (i.e., common patterns) from diversely expressed entity pairs with a given relation. Thus, the classification performance cannot be guaranteed when limited features are obtained from the corpus. To address this challenge, in this paper, we propose to employ a generative model, called conditional variational autoencoder (CVAE), to handle the pattern sparsity. We define that each relation has an individually learned latent distribution from all possible sentences expressing this relation. As these distributions are learned based on the purpose of input reconstruction, the model's classification ability may not be strong enough and should be improved. By distinguishing the differences among different relation distributions, a margin-based regularizer is designed, which leads to a margin-based CVAE (MCVAE) that can significantly enhance the classification ability. Besides, MCVAE can automatically generate semantically meaningful patterns that describe the given relations. Experiments on two real-world datasets validate the effectiveness of the proposed MCVAE on the tasks of relation classification and relation-specific pattern generation.

ACM Reference Format:
Fenglong Ma, Yaliang Li, Chenwei Zhang, Jing Gao, Nan Du, and Wei Fan. 2019. MCVAE: Margin-based Conditional Variational Autoencoder for Relation Classification and Pattern Generation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313436

*This work was done when the first and third authors were interns and the second author worked at Tencent Medical AI Lab.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313436

1 INTRODUCTION

Relation classification is a core task in natural language processing (NLP), which aims to identify semantic relations between a pair of entities in the given sentence. Relation classification can be widely used for multiple NLP tasks, such as knowledge graph completion and question answering. Traditional relation classification models [13, 27, 39] usually require high-quality annotated data, which makes them impossible to be applied to large-scale datasets.

Recently, distant supervision approach [26] has been proposed, which assumes that if a pair of entities has a relation in a knowledge base, then all the sentences that contain the two entities express this relation. This approach makes it easy to collect multiple sentences for different entity pairs and their relations, but it is hard to control the quality of the collected data. To remove the effect of "noisy" sentences, many approaches are proposed [8, 14, 15, 23, 31, 32, 37, 40]. All these studies are discriminative models, which aim to extract the accurate feature representations of relations to improve the prediction performance. However, if the quality of the input data is not very high, then they may not correctly capture the characteristics of the target, which will lead to unsatisfactory performance.

The quality of the collect data mainly depends on both the diversity of relation expressions and the quantity of sentences mentioning each entity pair. For a relation, it may be related to multiple entity pairs in the knowledge base. The extracted sentences mentioning those pairs may use different expressions to describe the semantic meanings of the relation. *If only a small part of commonly used expressions appear in the training data, it is hard for existing discriminative models to recognize the other possible expressions in the testing procedure.* This issue stems from the drawback of discriminative models, i.e., they are not designed to facilitate the **generalization ability**.

On the other hand, discriminative models focus on capturing those expressions with high occurrences among different sentences, i.e., recognizing relation patterns. In common domains, there may exist enough sentences with mentioned entity pairs on the same relation that can be used for the discovery of clear patterns. However, in specific domains, such as medical domain, given a pair of medical entities extracted from a medical knowledge base, to recognize the relation between the two entities, we need to extract sentences

from a given corpus. Unfortunately, the text corpus used for medical sentence extraction is usually not big. Though we can extract sentences for multiple entity pairs, the number of clean patterns among these sentences is limited, which leads to the *sparsity* of relation patterns and further decreases the quality of the dataset. Therefore, how to design a model that can handle the **diversity of relation expressions and sparsity of relation patterns** in the training dataset is a challenge for relation classification.

To tackle this challenge, instead of using discriminative models to capture the representative features of relations, we design a *generative* model, named MCVAE, which employs a Margin-based Conditional Variational AutoEncoder. The benefit of applying a generative model to solve a classification problem is that the generalization ability of generative models is significantly better than that of discriminative ones. Furthermore, such an ability helps generative models to handle the issue of pattern sparsity. Particularly, the proposed MCVAE consists of four modules: An **encoder** is used to learn the embeddings of input sentences. By concatenating the embeddings with the given relation, the encoder outputs the mean vector and the diagonal vector of covariance matrix for constructing the latent space of the given relation. A **decoder** aims to reconstruct the input sentence using a sample drawn from the constructed latent space. Motived by [22], a margin-based **regularizer** is designed to model the differences among different constructed latent spaces of relations and further increase the ability of classification. Finally, a **generator** is adopted to generate sentences and patterns for the given relation type based on the trained model. Experimental results on two datasets show that the proposed MCVAE outperforms existing relation classification models and produces meaningful patterns which describe the given relations. Here, we highlight our contributions as follows:

- To the best of our knowledge, this is the first work to design a generative model for solving the problems of expression diversity and pattern sparsity in the relation classification task.
- The proposed model not only accurately identifies the relation types of the input sentences with the designed margin-based regularizer, but also generates meaningful sentences and patterns for the given relations using the conditional variational autoencoder.
- Experiments on two real world datasets demonstrate the effectiveness of the proposed MCVAE for both relation classification and relation-specific sentence generation.

2 RELATED WORK

In this section, we review existing work from two aspects: relation classification and deep generative models.

Relation Extraction or Classification [1, 3, 10, 20, 21, 28, 29, 33, 34, 42] is an important sub-task of Information Extraction (IE). IE can be done in unsupervised [2, 7, 9] or semi-supervised domain [4, 36], and even in the form of OpenIE [16], where facts are extracted from the data along with the relation phrases, i.e., without predefined ontology or relation classes. Supervised relation classification aims to identify the relation between two entities with the given text. Many models are developed to solve this problem, such as deep learning based models with distant supervision technique [8, 14, 15, 23, 31, 32, 37, 40]. Distant supervision is proposed in [26] to collect large-scale datasets for training, but inevitably there is noise in the collected data. To reduce the effect of the noise in the data, many studies cast the problem of relation classification as a multiple instance learning problem [14, 32, 40]. Recently, sentence-level attention mechanism over multiple instances are proposed to assign lower weights to those noisy sentences [15, 23, 24]. Moreover, reinforcement learning-based sentence selection approaches [8, 31] are introduced to further improve the performance.

Deep generative models, such as generative adversarial networks (GAN) [11], have attracted much attention recently. Different from GAN, which generates data based on the arbitrary noise, variational autoencoder (VAE) tries to model the underlying probability distribution of data by constructing the latent variables, which makes it possible to generate new samples from the learned distributions. Actually, there are many VAE-based models [18, 38] that can be used to generate different kinds of data, such as images [12, 30], natural languages [5, 25] and structured medical entity pairs [41].

The aforementioned models either only focus on classification or generation. Different from them, the proposed model can handle both classification and generation simultaneously. The generative ability is obtained by the designed conditional variational autoencoder, while the ability of classification is achieved by the designed margin-based regularizer.

3 THE PROPOSED MODEL: MCVAE

In this paper, we introduce a novel margin-based conditional variational autoencoder (MCVAE) for distant supervised relation classification task. It is a fact that the relation between a pair of entities can be expressed by many ways (i.e., expression diversity). Thus, using a continuous vector to model a relation with multiple expressions is not enough. Moreover, the number of sentences used for relation classification may be small, which results in the problem of pattern sparsity. To solve these challenges, we propose to use a *generative model* for modeling the diverse expressions of relations. The benefits of the proposed MCVAE model are three-fold: (1) The relations are embedded into distributions, instead of vectors, which directly increases the ability of generalization; (2) given a relation, the proposed model is able to generate sentences on this relation, which may contain the frequently used patterns for this relation; and (3) the model can guarantee the prediction performance by modeling the differences among relation distributions. Next, we will introduce the details of the proposed MCVAE architecture.

3.1 Model Overview

Figure 1 shows the architecture of the proposed MCVAE. It consists of four modules: *encoder*, *decoder*, *regularizer*, and *generator*. The **encoder** module takes a relation r and a sentence $\mathbf{x} = \{w_i\}$ as input data, where $\{w_i\}$ represents a set of ordered words. For each word w_i, an embedding layer is employed to map the discrete word to a continuous vector representation. An RNN is then used to encode the whole sentence (i.e., a set of word embeddings) to a hidden vector. Combining the hidden vector with the input relation r, we can construct a latent space represented by the means and the diagonal vectors of covariances. The latent variable $\mathbf{z}$ can be

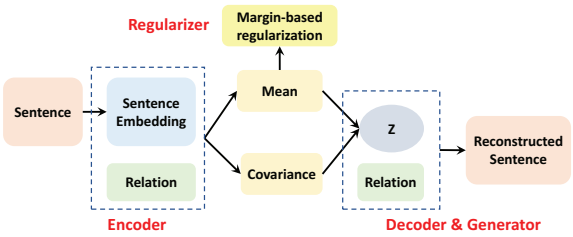


Figure 1: The Proposed MCVAE Model.

drawn from the learned latent space, which is one of the inputs of the decoder, and the other input is the label of the relation r.

The **decoder** tries its best to reconstruct the original input $\mathbf{x}$ based on another RNN. Through training the reconstruction loss and the KL-divergence as described in CVAE, we can learn all the parameters. However, a drawback of the naive model is that the constructed latent spaces for different relations are indistinguishable. In other words, the model does not have the ability for classifying the input data. To gain such an ability, we design a novel margin-based **regularizer**. The intuition behind the regularizer is that we want to increase the margins among latent spaces of different relations.

By training the proposed model MCVAE, we can generate sentences based on the given the relation types with the generator. The **generator** has the same structure with the decoder. However, instead of reconstructing the original inputs, it directly generates samples from the learned latent spaces of relations. The details of different modules are introduced in the following sections.

3.2 Encoder

Traditional relation classification models embeds a "bag" of sentences, which contains the same entity pair, into a common vector representation for the given entity pair. Different from existing models, we consider the sentence-level embeddings. We believe that each extracted sentence with distant supervision more or less expresses the relation between two entities.

For each word $w_i \in \mathbf{x}$, a word embedding layer is used to map it to a continuous vector $\mathbf{v}_i = f_{\text{EMB}}(w_i) \in \mathbb{R}^d$, where d is the size of the word embeddings, and $\mathbf{v}_i$ is the input of the encoder RNN. In particular, we employ the Gated Recurrent Unit (GRU) [6] as the encoder network in the proposed model. The output of the encoder is a hidden vector $\mathbf{h} \in \mathbb{R}^g$ produced by the GRU, which is defined as follows:

$$\mathbf{h} = f_{\text{ENC}}(\mathbf{x}) = \text{GRU}_{\text{ENC}}(\mathbf{V}; \varphi), \tag{1}$$

where g is the dimensionality, $\mathbf{V} \in \mathbb{R}^{s \times d}$ is the matrix of all the word embeddings, s is the length of the input $\mathbf{x}$, and φ is the parameter set of the GRU.

Based on the latent representation of the input $\mathbf{x}$ and its relation r, we can construct the latent space $Q_\phi(\mathbf{z}|\mathbf{x}, r)$ of the given relation r, which is represented by the mean μ and σ, and $\mathbf{z}$ is a sample drawn from the latent space. Towards this goal, we first concatenate the hidden vector $\mathbf{h}$ and the one-hot vector $\mathbf{r} \in \mathbb{R}^n$, and then use two separate linear functions to learn μ and σ as follows:

$$\mu = [\mathbf{h}; \mathbf{r}] \cdot \mathbf{W}_\mu + \mathbf{b}_\mu, \tag{2}$$

$$\sigma = [\mathbf{h}; \mathbf{r}] \cdot \mathbf{W}_\sigma + \mathbf{b}_\sigma, \tag{3}$$

where n denotes the number of relations, $\mathbf{W}_\mu$, $\mathbf{W}_\sigma \in \mathbb{R}^{(g+n)\times m}$ and $\mathbf{b}_\mu$, $\mathbf{b}_\sigma \in \mathbb{R}^m$ are the parameters to be learned, and m is the number of dimensions. With the learned μ and σ, we can obtained the latent space of the given relation r. Next, we will introduce how to reconstruct the input sentence x with the learned μ and σ.

3.3 Decoder

Given μ and σ, we can directly sample a latent variable $\mathbf{z} \in \mathbb{R}^m$ from the constructed relation space $\mathcal{N}(\mu, \sigma)$. However, the direct sampling dose not make the whole model differentiable, which leads to the failure of existing optimization approaches to compute the gradients. To address this problem, we follow the method in [19] and apply the reparameterization trick. It works as follows: Instead of directly sampling from $\mathcal{N}(\mu, \sigma)$, we first sample from a standard normal distribution $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and then obtain a reparameteried $\mathbf{z} = \mu + \sigma \odot \epsilon$, where $\odot$ represents the element-wise multiplication. Since sampling from ϵ dose not depend on the network, it makes the proposed model differentiable again.

The decoder aims to reconstruct the input x with a recurrent neural network, which means that the input and output must be sequences. To decode the input sentence x, we need to add two special symbols: $\langle sos \rangle$ (the start of the sequence) and $\langle eos \rangle$ (the end of the sequence). The input of the decoder is $\langle sos \rangle + \mathbf{x}$, and the target is $\mathbf{x} + \langle eos \rangle$.

Since reconstructing relation-specific sentences is the goal of the designed decoder, the relation information should be considered in the decoding procedure, and the decoder can be denoted as $P_\theta(\mathbf{x}+\langle eos \rangle|\mathbf{z}, r)$. To this end, we first concatenate the latent variable $\mathbf{z}$ and the one-hot relation vector $\mathbf{r}$, and then convert it to a new vector $\mathbf{h}' \in \mathbb{R}^g$ with a linear function. The mapped vector $\mathbf{h}'$ is considered as the initial hidden state of the decoder RNN. Similar to the encoder, we need to embed the input words into continuous vector representations. Here, we use dropout technique [35] to remove the irrelevant information on the word embeddings of the decoder input, and then these embeddings $\mathbf{V}' \in \mathbb{R}^{(s+1)\times d}$ are treated as the inputs of the decoder RNN. Mathematically, we use the following formulation to represent the decoder RNN:

$$\mathbf{C} = f_{\text{DEC}}(\langle sos \rangle + \mathbf{x}) = \text{GRU}_{\text{DEC}}(\mathbf{V}', \mathbf{h}'; \varphi'), \tag{4}$$

where $\mathbf{C} \in \mathbb{R}^{(s+1)\times g}$, and φ' is the parameter set of the decoder RNN. To generate discrete words, we need to obtain the probability of each word using a softmax layer after a linear mapping as follows:

$$\mathbf{O} = \text{softmax}(\mathbf{C}\mathbf{W}_o + \mathbf{b}_o), \tag{5}$$

where $\mathbf{O} \in \mathbb{R}^{(s+1)\times|V|}$ is the probability matrix, $\mathbf{W}_o \in \mathbb{R}^{g\times|V|}$ and $\mathbf{b}_o \in \mathbb{R}^{|V|}$ are parameters to be learned, and $|V|$ is the vocabulary size. Cross entropy between the probability matrix $\mathbf{O}$ and the target $\mathbf{x} + \langle eos \rangle$ can be used to optimize the objective. To simplify the notations, we remove the special symbols (i.e., $\langle sos \rangle$ and $\langle eos \rangle$) in the following sections. We use $\mathbf{x}$ to represent the input and output of the decoder.

3.4 Training with Large-Margin Regularizer

Given the encoder and decoder, and following the loss of CVAE, we can define the reconstruction loss to minimize the variational

lower bound:

$$\mathcal{L}_{\text{ENC+DEC}}(\mathbf{x}, r; \theta, \phi) = \\ -\text{KL}(Q_\phi(\mathbf{z}|\mathbf{x}, r)\|P_\theta(\mathbf{z}|r)) + \mathbb{E}[\log P_\theta(\mathbf{x}|\mathbf{z}, r)]. \tag{6}$$

The first term is in charge of measuring the difference between the prior distribution $P_\theta(\mathbf{z}|r)$ and a sample distribution $Q_\phi(\mathbf{z}|\mathbf{x}, r)$ with KL-divergence loss. Actually, the unknown true distribution can be simplified by a prior distribution $P_\theta(\mathbf{z}|r)$. The second term is the reconstruction loss from the decoder. Let $P_\theta(\mathbf{z}|r) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ be the prior distribution, and then the KL-divergence can be rewritten as follows:

$$\begin{aligned} &-\text{KL}(Q_\phi(\mathbf{z}|r)\|P_\theta(\mathbf{z}|r)) = -\text{KL}(\mathcal{N}(\mu, \sigma)\|\mathcal{N}(\mathbf{0}, \mathbf{I})) \\ &= -\frac{1}{2}\sum_{i=1}^{m}(-\log(\sigma_i) + \sigma_i + \mu_i^2 - 1). \end{aligned} \tag{7}$$

Since the exponential term is more stable than the log term, in the experiments, we use $\log(\sigma)$ to model σ, i.e,

$$-\text{KL}(Q_\phi(\mathbf{z}|r)\|P_\theta(\mathbf{z}|r)) = -\frac{1}{2}\sum_{i=1}^{m}(-\sigma_i + \exp(\sigma_i) + \mu_i^2 - 1). \tag{8}$$

As discussed in the decoder, the cross entropy loss can be used, i.e.,

$$\mathbb{E}[\log P_\theta(\mathbf{x}|\mathbf{z}, r)] = -\sum_{j=1}^{s+1}\left(\mathbf{y}_j^\top \log(\mathbf{O}_j) + (\mathbf{1} - \mathbf{y}_j)^\top \log(\mathbf{1} - \mathbf{O}_j)\right), \tag{9}$$

where $\mathbf{y}_j$ is the one-hot representation of the j-th word in $\mathbf{x} + \langle eos \rangle$, and $\mathbf{O}_j$ is the estimated probability of the j-th word. Using both Eq. (8) and Eq. (9), we can optimize the model with the gradient-based optimizer, such as Adam [17]. However, this basic model does not have the ability of classification. The reason is that all the relation distributions are close to the prior distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$, which means that different relation spaces are mixed together. Thus, for a given sentence, it is hard for the basic model to assign the correct category label.

To gain the ability of distinguishing different relation types, the proposed model requires to learn the differences among relation spaces. In the proposed MCVAE, we add a large-margin regularizer into the objective of $\mathcal{L}_{\text{ENC+DEC}}$. There are n relation types, and for each sentence, it has a relation type r. In the encoder, we can obtain the mean vector $\mu_r \in \mathbb{R}^m$ of the relation distribution r. To ensure the classification ability of the proposed model, μ_r should be far away from the mean vectors of other relation distributions. Meanwhile, $\mathcal{N}(\mu_r, \sigma_r)$ should be closer to the prior distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$. Following these motivations, we design a margin-based regularizer, which is defined as follows:

$$\mathcal{L}_{\text{REG}} = \max(\alpha + f_{\text{D}}(\mu_r) - \beta, 0), \tag{10}$$

where $\alpha > 0$ is the predefined margin, $f_{\text{D}}()$ is the distance function (i.e., $f_{\text{D}}(\mu_r) = \sum_{i=1}^{m}\mu_{ri}^2$) to calculate the distance between the mean vector μ_r and the mean of the prior vector $\mathbf{0} \in \mathbb{R}^m$. β is defined as follows:

$$\beta = \min(f_{\text{D}}(\mu_1), \cdots, f_{\text{D}}(\mu_{r-1}), f_{\text{D}}(\mu_{r+1}), \cdots, f_{\text{D}}(\mu_n)). \tag{11}$$

From the designed regularizer (Eq. (10)), we can observe that if $f_{\text{D}}(\mu_r) > \beta - \alpha$, then $\mathcal{L}_{\text{REG}} > 0$. In this case, the proposed model should penalize the loss of $\mathcal{L}_{\text{REG}}$, which indicates that the difference value between $f_{\text{D}}(\mu_r)$ and β is at least α. The final loss function of the proposed MCVAE used in the training procedure is formulated as follows:

$$\mathcal{L}_{\text{MCVAE}} = \mathcal{L}_{\text{ENC+DEC}} + \mathcal{L}_{\text{REG}}. \tag{12}$$

In the testing procedure, assume that there is a bag of sentences $\{\mathbf{x}_1, \cdots, \mathbf{x}_t\}$ which mentions a pair of entities. For each sentence x_i, we first use the encoder to obtain the mean vector according to Eq. (2) for each relation r, and then calculate the distance using $f_{\text{D}}(\mu_r)$. Since we have n relations, for each sentence, n distances can be obtained and are denoted as $\mathbf{d}_i = [f_{\text{D}}(\mu_1), \cdots, f_{\text{D}}(\mu_n)]$. As there are t sentences in the bag, we aggregate all the distance vectors $\mathbf{d}_1, \cdots, \mathbf{d}_t$ to obtain an average vector $\mathbf{d}_{avg} \in \mathbb{R}^n$. Finally, we use the index corresponding to the minimum value in $\mathbf{d}_{avg}$ as the predicted label $\hat{r}$.

3.5 Generator

Based on the trained model, we can generate sentences which are related to a specific relation. Here, we use a density-based sampling approach for the generator to sample $\hat{\mathbf{z}}$ directly from the distribution of relations' latent spaces as [41]. The intuition of using density-based sampling approach is that the dense area (i.e., the area around the mean vector) contains "patterns" to express the relation r. A pattern means that a phrase or a set of words frequently appears in the sentences that express the relation r. Therefore, using the patterns in the dense area of the latent space, the generator can generate multiple meaningful sentences. The benefit of applying the density-based sampling method is that we do not need to cast a sentence with its relation into the encoder for learning μ and σ for the generator.

Particularly, given a relation type r, its one-hot vector representation can be obtained, which is denoted as $\mathbf{r} \in \mathbb{R}^n$. The following step is to sample a latent variable $\hat{\mathbf{z}}$ from the latent space $P_\theta(\hat{\mathbf{z}}|r)$. If the model is well-trained, then the latent variable $\hat{\mathbf{z}}$ will be drawn from $P_\theta(\hat{\mathbf{z}}|r) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The concatenation of $\hat{\mathbf{z}}$ and $\mathbf{r}$ and the embedding of $\langle sos \rangle$ are the inputs of the generator RNN, which has the same architecture with the decoder RNN, to generate the first word. The output of the generator RNN and the embedding of the first generated word are used to produce the second word. By repeating this procedure, the generator will generate the whole sentence.

4 EXPERIMENTS

In this section, we conduct experiments to show the effectiveness of the proposed model MCVAE.

4.1 Experimental Setting

In this subsection, we introduce the datasets, baselines, parameter settings and evaluation measures.

• **Datasets**. In our experiments, we adopt two datasets. The first one is a widely used public available dataset, called NYT[1], generated by [32]. For this dataset, we remove the sentences in the NA relation. For each entity pair, at most 500 sentences are kept. Finally, there are 136,844 sentences and 19,539 entity pairs (bags) in the training dataset. In the testing dataset, we have 6,437 sentences and 1,755 entity pairs. The total number of relation types is 54, which are extracted from Freebase.

[1] http://iesl.cs.umass.edu/riedel/ecml/

The other dataset, named MedBook, is generated from medical books with Tencent medical knowledge graph. In the training dataset, there are 8,509 sentences and 2,213 entity pairs. The testing dataset contains 592 entity pairs with 740 sentences, and there are 14 relations in the MedBook dataset.

• **Baselines**. Since the proposed MCVAE is a generative model, and there is no existing work for relation classification task using generative models. Thus, we use the variant of the proposed model as a baseline method, which is CVAE. CVAE removes the proposed large-margin regularizer from the proposed MCVAE. The proposed model can be used for not only sentence generation but also relation classification. Therefore, we compare MCVAE with discriminative models for relation classification: PRNN+AVG and PRNN+ATT. PRNN+ATT uses attention mechanism over sentences in a bag as the way in [23]. The only difference is that we use RNN instead of CNN. The reason is that the proposed model is based on RNN, and designing a CNN-based generative model is our future work. PRNN+AVG learns the embeddings of entity pairs by averaging representations of all the sentences in the same bag.

• **Parameter Settings**. For the NYT dataset, we use a pretrained word embeddings[2]. The parameters are set as follows: For each sentence, the max length s is 70; the size of word embeddings d is 50; the hidden size of RNN g is 256; the size of the latent spaces of relations m is 16; and the margin $\alpha = 5$. The learning rate for Adam is 0.001, and the dropout rate for the decoder inputs is 0.5. The batch size is 256 on the NYT dataset and 32 on the MedBook dataset. The number of epochs is set as 50 for both datasets.

• **Evaluation Measures**. To fairly evaluate the proposed MCVAE, we utilize two measures: accuracy and weighted F1 score. The number of entity pairs for different relations in the testing dataset is significantly different, so the weighted F1 score is introduced, which considers the number of instances when calculating the score. The higher the measures, the better the performance.

4.2 Performance of Relation Classification

Table 1 shows the results for relation classification task on the two datasets. We can observe that the overall performance of the proposed MCVAE is better than that of the baselines.

Table 1: Performance of Different Methods.

Method	NYT		MedBook	
	Accuracy	Weighted F1	Accuracy	Weighted F1
PRNN+AVG	0.4650	0.2951	0.2010	0.0775
PRNN+ATT	0.5031	**0.5019**	0.1537	0.0975
CVAE	0.0501	0.0537	0.2010	0.2180
MCVAE	**0.5288**	0.4875	**0.3108**	**0.2821**

On the NYT dataset, the variant of the proposed MCVAE, i.e., CVAE, has the worst performance. Both the accuracy and weighted F1 value are largely lower than those of other approaches. The reasons are two-fold: (1) When CVAE is well trained, all the latent spaces of different relations are close to the prior distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$. It means that CVAE only focuses on how to generate new sentences based on the given relations, instead of paying attention to classification. Thus, the latent spaces of different relations may be mixed together. In such a case, using the learned μ's for classifying different relations is impossible. (2) There are a lot of shared common words to describe multiple relations on this dataset. For example, the relations, */people/person/nationality*, */people/person/place_of_birth* and */people/person/place_lived*, may share many context words, which leads to the failure of CVAE on the relation classification task. However, by adding the proposed large-margin regularizer with CVAE, the proposed MCVAE can significantly improve the performance. This demonstrates that when using generative models to do the classification tasks, it is essential for them to consider the differences among different classes.

Compared with the discriminative models, the performance of the proposed MCVAE is better than that of PRNN+AVG, and is comparable with PRNN+ATT. Since the number of sentences for each entity pair is large on the NYT dataset, the attention mechanism can effectively assign greater weights to those sentences semantically expressing the given relations. Moreover, the classifier can correctly capture the characteristics of different relations. Thus, PRNN+ATT can achieve better performance compared with PRNN+AVG. Different from the learning procedure of discriminative models, the proposed MCVAE tries to enhance the ability of generalization. It means that MCVAE aims to use latent distributions to model different expressions of relations. MCVAE intends to put all the words or patterns describing the same relation together, but designs a margin-based regularizer to distinguish the differences among different distributions.

On the MedBook dataset, since the number of sentences in each bag (i.e., entity pair) is small, which leads to the bad performance of using attention mechanism. Thus, PRNN+AVG performs better than PRNN+ATT. However, both the proposed MCVAE and its variant CVAE achieve better performance than PRNN+AVG and PRNN+ATT. It is because on this dataset, for the same relation, there are many different expressions, and the occurrence of those expressions is much lower than that on the NYT dataset. Without the generalization ability being facilitated, discriminative models are less likely to capture features that are generalizable for classification during the test phase. However, for the generative models, they can collect all the related features to construct the relation distributions. Since the dataset is not big, even for CVAE, the learned relation distributions are still distinguishable. Therefore, the generative models outperform existing discriminative models. It shows the advantage of employing the designed margin-based regularizer for relation classification task.

4.3 Case Study

The benefit of the proposed MCVAE is that it not only guarantees the performance for relation classification, but also generates sentences that express the semantics of the given relations. To validate this claim, we conduct a case study on the NYT dataset. Given a relation type, the proposed MCVAE can produce sentences with the designed generator. In this case study, we show the sentences generated from the following four relations: */people/person/nationality*, */people/person/place_lived*, */business/company/founders*, and */location/us_state/capital*. The generated sentences are shown in Table 2.

[2]https://github.com/thunlp/NRE

Table 2: Case Study on the NYT Dataset.

Relation	Generated Sentences
/people/person/nationality	(1) he was born in <unk> , germany , and raised in 1958 , england , and received a bachelor (2) president hamid_karzai of afghanistan said he believed that he had been <unk> by the taliban and passport (3) president bush <unk> prime minister , ariel_sharon , said that he had spoken that the united states would not be
/people/person/place_lived	(1) <unk> , the former senator from north_carolina , said he was <unk> by the (2) a former senator from nebraska , said he was <unk> by the senate speaker (3) the former governor of massachusetts , who was elected to represent the senate seat <unk>
/business/company/founders	(1) but <unk> , the founder of microsoft , has been (2) the news_corporation <unk> chairman , rupert_murdoch , objected to the news_corporation (3) but larry_page , google <unk> chief executive , said that
/location/us_state/capital	(1) showdown in ethiopia <unk> capital of addis_ababa , ethiopia (2) new_york wants to endorse the state of albany , the state of new_york , which has been broadcast in albany (3) <unk> <unk> , born in columbus , ohio , on august 10 , 2005 .

From Table 2, we can observe that the proposed MCVAE indeed generates some meaningful patterns on the given relations. Although there are some noisy cases, such as the third sentence in relation */people/person/nationality* and the first sentence in relation */location/us_state/capital*, most generated sentences are reasonable. For the relation */people/person/place_lived*, there is a clear pattern extracted by the proposed MCVAE, i.e., "the former [title] of [location name]". This observation can also be found in the relation */business/company/founders*. When expressing the relation between company and founders, the proposed MCVAE can recognize the key phases: "the founder of [company name]", "chairman", "[company name] chief executive", and so on. Even for the noisy cases, we can observe the ability of generalization of the proposed MCVAE. For example, the first sentence in relation */location/us_state/capital* is obviously wrong. Since the target relation is about the states in USA and their capitals, the country is not "ethiopia". Though the relation of the generated sentence is different from the target one, the relation between "addis_ababa" and "ethiopia" is indeed related to capitals.

4.4 Relation Pattern Generation

It is obvious that the generated sentences shown in Table 2 contain noisy information, and it may be hard for us to discover clear patterns that describe the given relations from the noisy generated sentences. To further enhance the understandability and readability of the discovered patterns, we conduct another experiment on the NYT dataset. In this experiment, for each entity, we first extract its types from Freebase and manually assign a type to it according to the semantic meaning of the relation. Next, we replace the entities in the sentences with their types. Finally, we run the proposed MCVAE on the new typed NYT dataset with the same parameter settings.

The accuracy on the typed NYT dataset is **0.7761**, and the weighted F1 value is **0.7358**, which are much greater than those on the original NYT dataset. The reason is that after mapping the entities into their types, the proposed MCVAE can effectively recognize the patterns of different relations, and the margin-based regularizer can successfully distinguish these latent distributions of relations. Similar to the case study conducted on the original NYT dataset, we list some patterns extracted by the proposed MCVAE on the new typed NYT dataset shown in Table 3. From Table 3, we can observe that the generated patterns are highly related to the semantic expressions of all the four listed relations. Based on the results in this experiment, we can safely conclude that the proposed MCVAE is able to generate meaningful and distinguishable patterns for different relation categories.

Table 3: Generated Patterns on the Typed NYT Dataset.

Relation	Generated Patterns
/people/person/nationality	(1) the deputy president of $LOCATION , $PEOPLE (2) president $PEOPLE of $LOCATION (3) the former prime minister of $LOCATION , $PEOPLE
/people/person/place_lived	(1) senator $PEOPLE of $LOCATION (2) representative $PEOPLE of $LOCATION (3) $PEOPLE , the $LOCATION borough president
/business/company/founders	(1) $PEOPLE , the founder of $BUSINESS (2) $PEOPLE , the chief executive of $BUSINESS (3) the company owned by $PEOPLE , $BUSINESS
/location/us_state/capital	(1) located in $LOCATION , $LOCATION (2) who was born in $LOCATION , $LOCATION (3) who lives in $LOCATION , $LOCATION

5 CONCLUSIONS

In this paper, we propose a novel generative model for relation classification task, which can handle the problems of both expression diversity and pattern sparsity. The proposed MCVAE not only guarantees the classification performance, but also generates useful patterns which describe the semantic meanings of relations. MCVAE consists of four modules: encoder, decoder, regularizer and generator. The encoder embeds each input sentence into a vector representation, which is further adopted for constructing the latent space of the given relation. The decoder is in charge of reconstructing the input sentences with a designed decoder RNN. The designed regularizer is the core module of the proposed MCVAE, which equips MCVAE with the ability of distinguishing

different relation distributions. The generator is employed to generate sentences based on the given relations. Experiments on two real datasets demonstrate the effectiveness of the proposed MCVAE and the strong ability of generation.

ACKNOWLEDGMENTS

This work is supported in part by the US National Science Foundation under grants IIS-1553411 and IIS-1747614. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Azad Abad, Moin Nabi, and Alessandro Moschitti. 2017. Self-crowdsourcing training for relation extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vol. 2. 518–523.

[2] Alan Akbik, Larysa Visengeriyeva, Johannes Kirschnick, and Alexander Löser. 2013. Effective selectional restrictions for unsupervised relation extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. 1312–1320.

[3] Roberto Basili, Cristina Giannone, Chiara Del Vescovo, Alessandro Moschitti, and Paolo Naggar. 2009. Kernel-Based Learning for Domain-Specific Relation Extraction. In *Proceedings of the Congress of the Italian Association for Artificial Intelligence*. 161–171.

[4] Lidong Bing, William Cohen, Bhuwan Dhingra, and Richard Wang. 2016. Using Graphs of Classifiers to Impose Constraints on Semi-supervised Relation Extraction. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*. 1–6.

[5] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 10–21.

[6] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *arXiv preprint arXiv:1409.1259* (2014).

[7] Hady Elsahar, Elena Demidova, Simon Gottschalk, Christophe Gravier, and Frederique Laforest. 2017. Unsupervised Open Relation Extraction. In *Proceedings of the European Semantic Web Conference*. 12–16.

[8] Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement Learning for Relation Classification from Noisy Data. In *AAAI*.

[9] Kata Gábor, Haïfa Zargayouna, Isabelle Tellier, Davide Buscaldi, and Thierry Charnois. 2016. Unsupervised Relation Extraction in Specialized Corpora Using Sequence Mining. In *Proceedings of the International Symposium on Intelligent Data Analysis*. 237–248.

[10] Cristina Giannone, Roberto Basili, Chiara Del Vescovo, Paolo Naggar, and Alessandro Moschitti. 2009. Kernel-based relation extraction from investigative data. In *Proceedings of The Third Workshop on Analytics for Noisy Unstructured Text Data*. 93–100.

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of the Advances in neural information processing systems*. 2672–2680.

[12] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. 2015. DRAW: A Recurrent Neural Network For Image Generation. In *Proceedings of the International Conference on Machine Learning*. 1462–1471.

[13] Zhou GuoDong, Su Jian, Zhang Jie, and Zhang Min. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics, 427–434.

[14] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 541–550.

[15] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2017. Distant Supervision for Relation Extraction with Sentence-Level Attention and Entity Descriptions. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. 3060–3066.

[16] Shengbin Jia, Yang Xiang, and Xiaojun Chen. 2018. Supervised Neural Models Revitalize the Open Relation Extraction. *arXiv preprint arXiv:1809.09408* (2018).

[17] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Proceedings of the Advances in Neural Information Processing Systems*. 3581–3589.

[19] Diederik P Kingma and Max Welling. 2014. Stochastic gradient VB and the variational auto-encoder. In *Proceedings of the Second International Conference on Learning Representations (ICLR)*.

[20] Shantanu Kumar. 2017. A Survey of Deep Learning Methods for Relation Extraction. *arXiv preprint arXiv:1705.03645* (2017).

[21] Tuur Leeuwenberg and Marie-Francine Moens. 2017. Structured learning for temporal relation extraction from clinical records. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. 1150–1158.

[22] Chongxuan Li, Jun Zhu, and Bo Zhang. 2018. Max-margin deep generative models for (semi-) supervised learning. *IEEE transactions on pattern analysis and machine intelligence* 40, 11 (2018), 2762–2775.

[23] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 2124–2133.

[24] Tianyu Liu, Kexiang Wang, Baobao Chang, and Zhifang Sui. 2017. A Soft-label Method for Noise-tolerant Distantly Supervised Relation Extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1790–1795.

[25] Diego Marcheggiani and Ivan Titov. 2016. Discrete-state variational autoencoders for joint discovery and factorization of relations. *Transactions of the Association for Computational Linguistics* 4 (2016), 231–244.

[26] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*. Association for Computational Linguistics, 1003–1011.

[27] Raymond J Mooney and Razvan C Bunescu. 2006. Subsequence kernels for relation extraction. In *Proceedings of the Advances in neural information processing systems*. 171–178.

[28] Truc-Vien T Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Convolution kernels on constituent, dependency and sequential structures for relation extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. 1378–1387.

[29] Barbara Plank and Alessandro Moschitti. 2013. Embedding semantic similarity in tree kernels for domain adaptation of relation extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 1498–1507.

[30] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. 2016. Variational autoencoder for deep learning of images, labels and captions. In *Proceedings of the Advances in neural information processing systems*. 2352–2360.

[31] Pengda Qin, Weiran Xu, and William Yang Wang. 2018. Robust Distant Supervision Relation Extraction via Deep Reinforcement Learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. ACL, Melbourne, Victoria, Australia.

[32] Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 148–163.

[33] Rudolf Schneider, Cordula Guder, Torsten Kilias, Alexander Löser, Jens Graupmann, and Oleksandr Kozachuk. 2016. Interactive Relation Extraction in Main Memory Database Systems. In *Proceedings of the 26th International Conference on Computational Linguistics*. 103–106.

[34] Meilun Sheng, Lin Qiu, Chenyang Wu, Haofen Wang, and Yong Yu. 2013. Effective Chinese Relation Extraction by Sentence Rolling and Candidate Ranking. In *Proceedings of the China Semantic Web Symposium and Web Science Conference*. 147–160.

[35] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[36] Ang Sun, Ralph Grishman, and Satoshi Sekine. 2011. Semi-supervised relation extraction with large-scale word clustering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. 521–529.

[37] Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*. Association for Computational Linguistics, 455–465.

[38] Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. 2017. Variational Autoencoder for Semi-Supervised Text Classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. 3358–3364.

[39] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of machine learning research* 3, Feb (2003), 1083–1106.

[40] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1753–1762.

[41] Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip S Yu. 2018. On the Generative Discovery of Structured Medical Knowledge. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2720–2728.

[42] Hong-Tao Zhang, Min-Lie Huang, and Xiao-Yan Zhu. 2012. A unified active learning framework for biomedical relation extraction. *Journal of Computer Science and Technology* 27, 6 (2012), 1302–1313.

Detect Rumors on Twitter by Promoting Information Campaigns with Generative Adversarial Learning

Jing Ma
The Chinese University of Hong Kong
Hong Kong SAR
majing@se.cuhk.edu.hk

Wei Gao
Victoria University of Wellington
New Zealand
wei.gao@vuw.ac.nz

Kam-Fai Wong
The Chinese University of Hong Kong
& MoE Key Lab of High Confidence
Software Technologies, CUHK

ABSTRACT

Rumors can cause devastating consequences to individual and/or society. Analysis shows that widespread of rumors typically results from deliberately promoted information campaigns which aim to shape collective opinions on the concerned news events. In this paper, we attempt to fight such chaos with itself to make automatic rumor detection more robust and effective. Our idea is inspired by adversarial learning method originated from Generative Adversarial Networks (GAN). We propose a GAN-style approach, where a generator is designed to produce uncertain or conflicting voices, complicating the original conversational threads in order to pressurize the discriminator to learn stronger rumor indicative representations from the augmented, more challenging examples. Different from traditional data-driven approach to rumor detection, our method can capture low-frequency but stronger non-trivial patterns via such adversarial training. Extensive experiments on two Twitter benchmark datasets demonstrate that our rumor detection method achieves much better results than state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Natural language processing**.

KEYWORDS

Information Campaigns; Rumor Detection; GAN

ACM Reference Format:
Jing Ma, Wei Gao, and Kam-Fai Wong. 2019. Detect Rumors on Twitter by Promoting Information Campaigns with Generative Adversarial Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA,* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313741

1 INTRODUCTION

The proliferation of rumors is a rampant phenomenon on social media. Rumor producers can cause devastating effects by manipulating public events. Information campaigns are frequently carried out by rumor makers via social networks to promote controversial memes, fake news, etc. with high volume of misinformation that competes with genuine ones for dragging people's attention to bogus claims. For example, during the US 2016 presidential election, Russia reportedly had coordinated thousands of "social bots" like covert human agents and automated programs to spread false information by corroborating each other[1]. The widespread of rumors has triggered huge debates and has already unprecedentedly influenced US politics.

Social psychology literature defines a rumor as a story or a statement whose truth value is unverified or deliberately false [1]. Fact-checking websites such as snopes.com and politifact.com rely on manual effort to track and debunk rumors, which have obvious limitation on efficiency and coverage. Existing automated methods typically resort to supervised classifiers trained on a wide range of hand-crafted features based on message contents, user profiles and diffusion patterns [3, 8–10, 13]. To avoid feature engineering effort, data-driven models were exploited more recently and demonstrated state-of-the-art detection performances. For example, Ma et al. [12] employed recurrent neural networks (RNNs), specifically LSTM and GRU, to learn hidden features from text content of relevant posts regarding given claims for detecting rumors. Yu et al. [20] used convolutional neural networks (CNNs) for obtaining local-global features from the relevant posts.

Nevertheless, existing data-driven approaches typically rely on finding indicative responses such as skeptical and disagreeing opinions for detection. Rumor producers can take advantage of promoted campaigns to entangle public opinions or influence collective stances to get it widely disseminated and amplified. This poses a major technical challenge to data-driven methods as text patterns and other explicit features become hardly discriminative. Various conflicting and uncertain voices that co-exist can seriously disturb the learning (or extraction) of useful features. Figure 1 illustrates a case of promoted campaign for the rumor about "Saudi Arabia beheads first female robot citizen", which shows how the popular indicative patterns expressing skepticism and disagreement such as "fake news", "not sure", "no truth" are inundated by the promoted posts. Therefore, developing a more robust feature learner for rumor detection is urgently desirable.

In this paper, we propose a radically new rumor detection method by leveraging the mechanism of information campaign and promoting it in a controlled manner in order to achieve more robust and effective detection. Our seemingly counter-intuitive idea is inspired by the Generative Adversarial Networks or dubbed as GAN [6, 7], where a discriminative classifier learns to distinguish whether an instance is from real world, and a generative model is trained to confuse the discriminator by generating proximately realistic examples. The harder are the generated examples to be distinguished from real-world ones, the stronger is the discriminator that can be

[1] http://time.com/4783932/inside-russia-social-media-war-america/

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313741

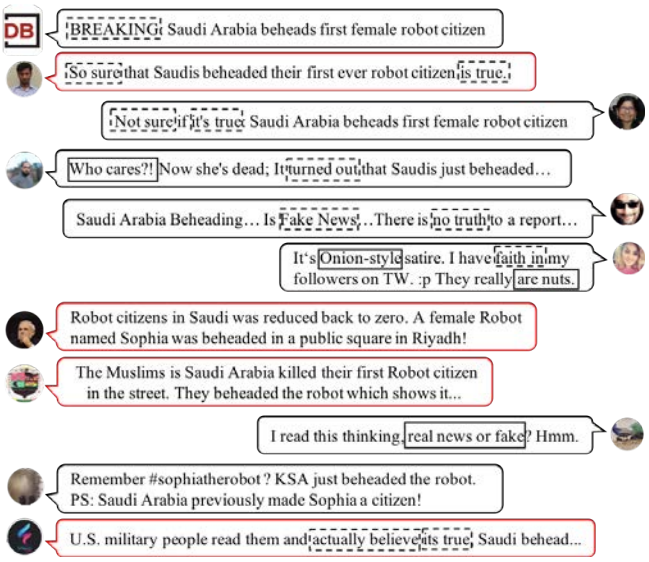


Figure 1: Sample responses to a rumor claim about "Saudi Arabia beheads its first female robot citizen" in a promoted campaign. Social bots activities are marked as red box. The supportive responses are listed at left side and denial (or uncertain) responses at right side. Patterns captured by existing methods are marked by dashed rectangles, and patterns that can be missing are marked by solid rectangles.

learned. We train our generator to output challenging examples by mimicking campaign promotions including misled grassroots conversations with uncertain and conflicting voices, so as to push our discriminator to strengthen feature learning from such difficult examples for capturing more discriminative patterns. Unlike typical GAN-style models such as in computer vision [6], which aim to learn a strong (image) generator, our goal however is to force our (rumor) *discriminator* to be more discriminative.

Intuitively, why can such a GAN-style method do better in feature learning? As shown in Figure 1, various users engagements can easily break the past data-driven methods that typically resort to repetitive patterns in responding posts. Due to the effect of generated campaign, high-frequency patterns such as "fake news" or "no truth" commonly occurring in the responses of rumors and "be true/sure" in those of non-rumors become less discriminative. As a result, the discriminator is adjusted adaptively to focus on capturing the relatively low-frequency patterns, such as "onion-style" and "be nuts", which are expected non-trivial as high-frequency ones while used to be ignored in existing feature learning methods. To retain the discriminative power of the original high-frequency patterns, we train the discriminator on an augmented training data with both generated campaign-like examples and the original examples.

The main contributions of our paper are four-fold:

- To the best of our knowledge, this is the first generative approach for rumor detection using a text-based GAN-style framework, where we make the text generator and discriminator adversarially strengthen each other for enhancing representation learning of rumor-indicative patterns.
- We model rumor dissemination as generative information campaigns for generating confusing training examples to challenge the discriminator of its detection capacity.
- Under the GAN-style framework, we reinforce our discriminator, which is trained on a set of more challenging examples replenished by the generator, to be focused on learning low-frequency yet discriminative patterns.
- We experimentally demonstrate that our model is more robust and effective than state-of-the-art baselines based on two public benchmark datasets for the tasks of rumor detection on Twitter.

2 PROBLEM STATEMENT

In general, rumor detection task can be defined as a binary classification problem, which aims to learn a classifier from training claims labeled as **rumor** or **non-rumor** for predicting the label of a test claim. A claim is a factual (rather than opinionated) assertion or statement that something is true, such as the example statement "Saudi Arabia beheads its first female robot citizen" in Figure 1.

Typically, a claim is short which contains very limited context. For reliable feature extraction, in Twitter rumor detection task, a claim is commonly represented by a set of posts (i.e., tweets) relevant to the claim which can be collected via Twitter's search function. Specifically, we represent a rumor dataset as $\{X\}$, where each $X = (y,\ x_1 x_2 \dots x_T)$ is a tuple representing a given claim: X consists of the ground-truth label $y \in \{N, R\}$ of the claim (i.e., Non-rumor or Rumor) and a sequence of relevant posts $x_1 x_2 \dots x_T$, where each x_t can represent a post or more generally a batch of posts in a time interval, and is indexed with a time step t. Thus, a claim can be considered as a time sequence of relevant posts. For clarity, we write an instance (claim) X as X_y, that is, X_R denotes a rumor and X_N a non-rumor.

3 GENERATIVE ADVERSARIAL LEARNING FOR RUMOR DETECTION

Information campaigns pose challenges to existing rumor detection models since frequent patterns indicative of veracity become distorted and misleading. Our basic idea is to strengthen representation learning of rumor indicative features, inspired by the mechanism of generative adversarial learning [7]. We propose a GAN-style model where a generator attempts to promote campaigns by generating hard examples and a discriminator aims to identify robust features to overcome the difficulty posed by the generator. Unlike the recent event adversarial model [18] for multi-modal fake news detection and the neural user response generator [16] for early detection, our idea and the adopted mechanisms are significantly different.

3.1 Controversial Example Generation

Our generative model aims to produce uncertain or conflicting voices regarding a given claim, making the differentiation of rumor and non-rumor harder which used to rely on repetitive patterns.

A straightforward way is to twist or complicate the opinions expressed in the original data examples via a handful of rule templates. For instance, we can 1) incorporate enquiry expressions such as "really?", "is it true?", "not sure", etc. into responding posts; 2) negate its stance of a post by adding a "not" after a "be" verb; and/or 3) apply antonym replacement at certain parts of the keywords, e.g.,

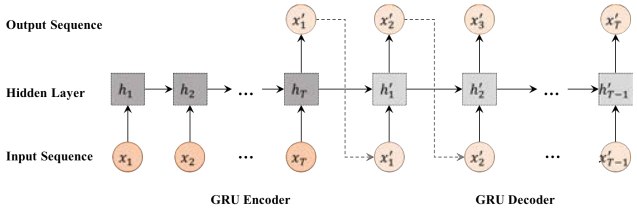


Figure 2: Framework of our generator with a neural sequence-to-sequence model. x'_t is the transformation of x_t, assuming that the length of output sequence equals to that of the input.

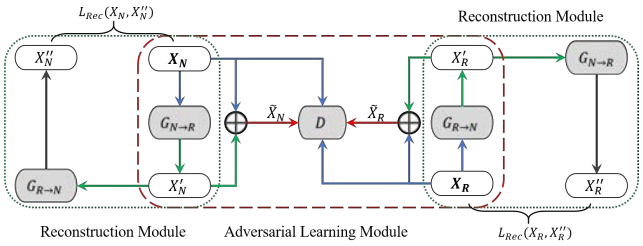


Figure 3: Overview of our GAN-style rumor detection model. $G_{N\to R}$ (or $G_{R\to N}$) is a generator. D denotes the discriminator.

by replacing "fake" with "true", "right" with "wrong", etc. However, it is difficult to generalize these rules for formally producing any kind of controversial voices. A general approach would be to cast our generator as trainable model that can cover a wide range of variations of expressions.

To this end, we design two generators, one for distorting non-rumor to make it look like a rumor, and the other for "whitewashing" rumor so that it looks like a non-rumor: 1) $G_{N\to R}$ generates skeptical or opposing voices against non-rumor claim; and 2) $G_{R\to N}$ generates supportive voices towards rumor claims. We define a function f_g to formulate our generative model:

$$X'_y = f_g(X_y) = \begin{cases} G_{N\to R}(X_y) & \text{if } y = N; \\ G_{R\to N}(X_y) & \text{if } y = R \end{cases} \tag{1}$$

where X_y is an original instance from training set which is either a rumor or non-rumor, and X'_y is the transformed instance with the generator while the label remains intact.

Considering the time sequence structure of posts in each instance, we use a sequence-to-sequence model [11, 17] for the generative transformation, which is illustrated in Figure 2. We encode an input sequence X_y into a hidden vector via an RNN encoder, and then generate the transformed sequence X'_y from it via an RNN decoder.

GRU-RNN Encoder: We batch relevant posts into time intervals and treat each batch as a single unit in the time sequence, following the similar time segmentation described in [12]. Using RNN, we map each input unit $x_t \in X_y$ into a hidden vector h_t, for which we use GRU [4] to store hidden representation:

$$h_t = \text{GRU}(x_t, h_{t-1}; \theta_g) \tag{2}$$

where GRU$(\cdot)$ denote standard GRU transition equations, x_t is the input unit represented as a vector of tf*idf values of vocabulary words computed from the posts falling into the t-th time step, h_{t-1} refers to the previous hidden state, and θ_g represent all the parameters of GRU.

The output of the last time step h_T from the GRU-RNN encoder will be the hidden representation of X_y. Note that the sequence length T is not fixed which can vary with different instances.

GRU-RNN Decoder: We now describe the GRU-RNN decoder that transforms h_T into the generated sequence $X'_y = x'_1 x'_2 \dots x'_T$. Specifically, each unit is sequentially generated using GRU followed by a softmax output function. In each step t, a softmax output layer maps the hidden states h'_t which is obtained via GRU into the target representation x'_{t+1} of a batch of posts by computing a distribution over vocabulary words:

$$\begin{aligned} h'_t &= \text{GRU}(x'_t, h'_{t-1}; \theta'_g) \\ x'_{t+1} &= \text{softmax}(V_g h'_t + b_g) \end{aligned} \tag{3}$$

where h'_{t-1} is the previous hidden state of GRU decoder and θ'_g represents all the parameters inside the GRU[2], V_g and b_g are trainable parameters of the output layer.

3.2 GAN-Style Adversarial Learning Model

In this encoder-decoder framework, $f_g(X_y)$ defines the policy that generates the representation of controversial posts which are used to confuse the discriminator based on the input X_y. A crucial issue is how to control the generator to generate the needed. To this end, we use the performance of discriminator as a reward to guide the generator. Before we present the discriminator, let us introduce the architecture and control mechanism of our GAN-style model as shown in Figure 3, which consists of an adversarial learning module and two reconstruction modules (one for rumor and the other for non-rumor).

Adversarial learning module: In our model, the generators are encouraged to produce campaign-like instances to fool the discriminator so that discriminator can be focused on learning more discriminative features. Such a goal suggests a training objective resembling adversarial learning [7]. We formulate adversarial loss as the *negative* of discriminator loss based on the generator-augmented training data:

$$L_{Adv} = -L_D(\bar{y}, \hat{y}) \tag{4}$$

where $L_D(\cdot)$ is the loss between the ground-truth class probability distribution $\bar{y}$ and the class distribution $\hat{y}$ predicted by discriminator given an input instance (see Eq. 11 for the specific form of L_D).

We combine the generated examples and original ones to augment the training set by taking the union of them, i.e., $\{\{X_y\} \cup \{\tilde{X}_y\}\}$, where $\tilde{X}_y = X_y \oplus X'_y$ is the elementwise addition of the original and generated examples. Note that the elementwise addition has the effect to cancel out influential high-frequency patterns and promote the chance of important less frequent patterns for being selected. Meanwhile, we do not want to seriously weaken those useful features in the original example. Thus, the original example X_y is combined with $\tilde{X}_y$ for training as shown in Figure 3.

[2]Note that each generator ($G_{R\to N}$ or $G_{N\to R}$) corresponds to a unique sequence-to-sequence model. The encoding and decoding GRUs have similar yet different set of parameters: θ_g and θ'_g.

Reconstruction module: It is likely that the generator could distort the original example towards unexpected direction by changing some essential aspects of a story. For example, the theme of "Saudi beheads robot citizen" might get distorted as "Saudi digs canal for Qatar" which is irrelevant and not helpful. To avoid that, we introduce a reconstruction mechanism to make the generative process reversible. The idea is that the opinionated voices will be reversible through two generators of opposite direction so as to minimize the loss of fidelity of information. We define the reconstruction function as follows:

$$X''_y = f_r(X_y) = \begin{cases} G_{R\to N}(G_{N\to R}(X_y)) & \text{if } y = N; \\ G_{N\to R}(G_{R\to N}(X_y)) & \text{if } y = R \end{cases} \tag{5}$$

where X''_y is the reconstructed instance from an original instance X_y via two opposite generators. We formulate the difference between X''_y and X_y as a reconstruction loss:

$$L_{Rec} = \frac{1}{T}\sum_{t=1}^{T} ||x_t - x''_t||_2 \tag{6}$$

where x_t and x''_t are the t-th unit in the original and reconstructed sequences, respectively, T is the sequence length, and $||\cdot||_2$ is the L2-norm of a vector.

Objective of optimization: The overall loss function of our GAN-style adversarial learning is defined as linear interpolation of L_{Adv} and L_{Rec}:

$$L = \alpha L_{Adv} + (1-\alpha)L_{Rec} \tag{7}$$

where α is the trade-off coefficient between adversarial and reconstruction losses. And the objective of adversarial learning takes a min-max form:

$$\max_{\Theta_D}\left(\min_{\Theta_G} L\right) \tag{8}$$

where $\Theta_G = \{\theta_g, \theta'_g, V_g, b_g\}$ are generators' parameters, and Θ_D are discriminator's parameters which will be detailed in the next section. In the min-max process, we first optimize Θ_G by minimizing adversarial loss L_{Adv} (i.e., maximizing discriminator loss L_D) and reconstruction loss L_{Rec} to generate confusing but reversible examples; we then optimize discriminator parameters Θ_D for classification by maximizing adversarial loss L_{Adv} (i.e., minimizing discriminator loss L_D), and note that L_{Rec} is independent of Θ_D.

3.3 Rumor Discriminator

With the training data augmented with the generative processing, the discriminator learns to capture more discriminative features, especially from low-frequency non-trivial patterns.

We build the discriminator based on a RNN rumor detection model [12]. Given an instance (either original or generated), the RNN model first maps relevant posts x_t at the t-th step into a hidden vector s_t using GRU:

$$s_t = \text{GRU}(x_t, s_{t-1}; \theta_d) \tag{9}$$

where s_{t-1} is the previous hidden vector and θ_d denotes all the GRU parameters in discriminator.

Following [12], we feed the hidden vector s_T at the last time step as the representation into a 2-class softmax function for classifying the instance:

$$\hat{y} = \text{softmax}(V_d s_T + b_d) \tag{10}$$

Algorithm 1: Generative adversarial training procedure.

Input : A set of training claims $\{X\}$, learning rate ϵ

1 Initialize Θ_G, and Θ_D with random weight values;
2 **for** *epoch from 1 to maxIter* **do**
3 **for** *each mini-batch* $\{\{X_N\},\{X_R\}\}$ **do**
4 Generate $\{\tilde{X}_N\}$: $\{X_N \oplus G_{N\to R}(X_N)\} \to \{\tilde{X}_N\}$;
5 Generate $\{\tilde{X}_R\}$: $\{X_R \oplus G_{R\to N}(X_R)\} \to \{\tilde{X}_R\}$;
6 Augment training set: $\big\{\{X_N\}\cup\{\tilde{X}_N\}, \{X_R\}\cup\{\tilde{X}_R\}\big\}$;
7 Compute loss L using Eq. 7;
  `/* Minimize L w.r.t. Θ_G */`
8 Compute gradient $\nabla(\Theta_G)$;
9 Update generators: $\Theta_G \leftarrow \Theta_G - \epsilon\nabla(\Theta_G)$;
  `/* Maximize L w.r.t. Θ_D */`
10 Compute gradient $\nabla'(\Theta_D)$;
11 Update discriminator: $\Theta_D \leftarrow \Theta_D - \epsilon\nabla'(\Theta_D)$;
12 **end for**
13 **end for**

where $\hat{y}$ is the vector of predicted probabilities over the two classes, V_d is the weight matrix of output layer and b_d is the trainable bias.

The loss of discriminator is defined as the square error between distributions of the predicted class and the ground-truth class:

$$L_D(\bar{y}, \hat{y}) = ||\bar{y} - \hat{y}||_2^2 + \lambda||\Theta_D||_2^2 \tag{11}$$

where $\bar{y}$ and $\hat{y}$ are respectively the ground-truth and predicted class probability distributions, $\Theta_D = \{\theta_d, V_d, b_d\}$ are discriminator parameters, and λ is the trade-off coefficient.

3.4 Generative Adversarial Training Algorithm

Algorithm 1 presents the iterative training process of the generators and discriminator in our GAN-style framework. Unlike original GAN [7] for obtaining better generators, our goal is to reinforce the discriminator to be more discriminative and generalizable.

The generators and discriminator are alternately trained using stochastic gradient decent with mini-batches [2]. In each epoch, controversial examples are generated and augmented into the original training data. We optimize the generator and discriminator against the augmented training examples with Eq. 8 that enforces a min-max game through steps 8-11 in Algorithm 1.

In training, we initialize model parameters with uniform distribution and update them by employing the derivative of the loss through back-propagation [5]; we iterate training until the maximum epoch number is met, which is set as 200; we fix the vocabulary size as 5,000, the size of hidden vector as 100, and tune the hyper parameters α, λ and ϵ using held-out dataset; post sequences take variable length dependent of specific instances by following [12].

4 EXPERIMENTS AND RESULTS

4.1 Datasets

We resort to two public datasets TWITTER [12] and PHEME [22] for experimental evaluation[3] The two datasets were used for binary classification of rumor and non-rumor with respect to a claim via

[3]https://figshare.com/articles/PHEME_dataset_of_rumours_and_non-rumours/4010619.

Table 1: Statistics of the datasets

Statistic	TWITTER	PHEME
Users #	491,229	29,387
claims #	992	2,246
Non-Rumors #	498	1,123
Rumors #	494	1,123
Avg. time length / claim	1,582.6 Hours	19.9 Hours
Avg. # of posts / claim	1,111	26
Max # of posts / claim	62,827	289
Min # of posts / claim	10	12

its relevant tweets. In particular, PHEME were collected based on 5 breaking news, thus its claims overlap more than TWITTER which was collected based on the claims reported on snopes.com. Moreover, we filtered out claims with less than 10 tweets and balanced the number of instances of the two classes. Statistics of the resulting datasets are given in Table 1.

4.2 Experimental Setup

We made comparisons among the following models:

DT-Rank: A Decision-Tree-based Ranking method that identifies trending rumors [21] through searching for disputed claims.

DTC: A Decision Tree Classifier for modeling Twitter information credibility [3] using various handcrafted features.

SVM-TS: A linear SVM classification model that uses time series to model the chronological variation of social context features [13].

BOW: A naive baseline representing the texts using bag-of-words and building the rumor classifier with linear SVM.

GRU: A RNN-based rumor detection model [12] with GRU for representation learning of relevant posts over time.

CNN: A CNN-based model for misinformation identification [20] for learning rumor representations by framing the relevant posts as fixed-length sequence.

GAN-GRU, **GAN-CNN** and **GAN-BOW**: Our GAN-style learning models where the discriminator adopts the data-driven models **GRU**, **CNN** and **BOW** above, respectively. Since replacing GRU with CNN or BOW as discriminator is straightforward, we omit describing the structures of **GAN-CNN** and **GAN-BOW**.

We hold out 10% of the claims in each dataset for tuning the hyper parameters, and for the rest of the claims, we conduct 5-fold cross-validation and use accuracy, precision, recall and F-measure as evaluation metrics. We implemented these models under comparison and will release our source codes publicly[4].

4.3 Result and Analysis

Table 2–3 demonstrate the performance of all the compared models based on the two datasets. The results indicate that our **GAN-GRU** model outperforms all the baselines, which confirms the advantage of generative adversarial learning for rumor detection task.

It is observed that the 3 baselines based on hand-crafted features perform clearly worse than all the 6 purely data-driven models. **SVM-TS** is relatively better because it incorporate additional temporal information into the conventional models. The results of

[4]https://github.com/majingCUHK/Rumor_GAN

Table 2: Results of comparison on TWITTER dataset

Method	Class	Accu.	Prec.	Rec.	F_1
DT-Rank	R	0.674	0.652	0.814	0.724
	N		0.716	0.519	0.602
DTC	R	0.732	0.694	0.794	0.741
	N		0.778	0.675	0.723
SVM-TS	R	0.798	0.778	0.837	0.807
	N		0.823	0.760	0.790
BOW	R	0.775	0.746	0.833	0.787
	N		0.811	0.716	0.761
CNN	R	0.821	0.815	0.831	0.823
	N		0.829	0.810	0.819
GRU	R	0.835	0.821	0.858	0.839
	N		0.852	0.812	0.832
GAN-BOW	R	0.792	0.761	0.850	0.803
	N		0.830	0.733	0.779
GAN-CNN	R	0.852	**0.853**	0.850	0.851
	N		0.853	**0.852**	0.852
GAN-GRU	R	**0.863**	0.843	**0.892**	**0.866**
	N		**0.885**	0.833	**0.858**

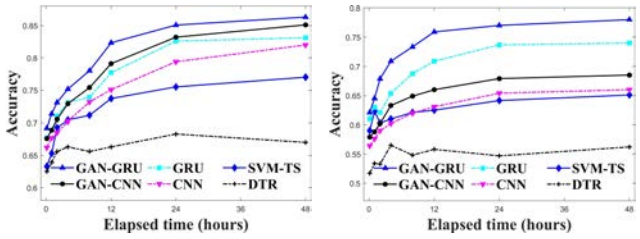
Table 3: Results of comparison on PHEME dataset

Method	Class	Accu.	Prec.	Rec.	F_1
DT-Rank	R	0.562	0.588	0.421	0.491
	N		0.549	0.704	0.617
DTC	R	0.581	0.582	0.573	0.578
	N		0.579	0.588	0.584
SVM-TS	R	0.651	0.663	0.617	0.639
	N		0.642	0.686	0.663
BOW	R	0.704	0.724	0.675	0.699
	N		0.687	0.734	0.710
CNN	R	0.665	0.671	0.652	0.661
	N		0.661	0.679	0.669
GRU	R	0.742	0.737	0.753	0.745
	N		0.754	0.730	0.739
GAN-BOW	R	0.736	0.755	0.701	0.727
	N		0.721	**0.772**	0.745
GAN-CNN	R	0.688	0.683	0.698	0.690
	N		0.695	0.678	0.685
GAN-GRU	R	**0.781**	**0.773**	**0.796**	**0.784**
	N		**0.791**	0.766	**0.778**

DT-Rank are poor due to the low coverage of the patterns using the regular expressions it defined.

BOW performs surprisingly well which is comparable to or even outperforms using hand-crafted features which confirms the advantage of using the simplest data-driven approach. **GRU** performs the best among all the baselines, which is not surprising since it takes advantage of deep neural nets to capture complex hidden features indicative of rumors beyond explicit and shallow patterns.

We conjectured that **CNN** should be comparable to **GRU** because both can learn deep latent features from data. This turned out to be incorrect on PHEME where CNN performs much worse. The reason is that RNN can inherently deal with variable-length sequence while CNN is essentially not a sequential model. The relevant posts per



(a) TWITTER dataset (b) PHEME dataset

Figure 4: Results of rumor early detection

claim in PHEME is significantly fewer than TWITTER, rendering lots of zero-valued input units to CNN that can worsen convolution operations, but RNN can easily get rid of zero input units by shortening sequence length. This also explains the overall better performance of all models on TWITTER.

Clearly, our three generative adversarial models outperform their counterparts in baselines that are not generative adversarial. The improvements of our models over these baselines range from 2.6% to 5.3% on the two datasets, indicating the adversarial learning with the generative-discriminative process is generally helpful and effective. However, two reasons may prevent further improvements on the two datasets: 1) The post content associated with PHEME claims overlap heavily since they come from only 5 breaking news, rendering high-frequency patterns more sensitive to topical categories than veracity categories. The claims in TWITTER dataset are however easier to classify as per veracity since each of them is an independent news topic, making high-frequency patterns well correlated to veracity rather than topic[5]. The potential of GAN-style learning by promoting the chance of low-frequency patterns is thus somewhat limited on TWITTER. This can be implied by the relatively lower improvement obtained on TWITTER than PHEME. 2) In addition to overlapping topics, PHEME is harder to classify also because there are only 26 posts per claim in average, thus useful information available is relatively limited.

Furthermore, rumor detection task emphasizes the identification performance on the rumor category. GAN-GRU achieves the highest recall on rumor category, indicating that more rumors can be found. From a balanced view of precision and recall, we also observe that the F1 scores of rumor category achieved by GAN-GRU are higher than non-rumor category.

4.4 Early Rumor Detection

Early alerts of rumors can prevent further spreading of rumorous content. By setting a detection delay time, only tweets posted no later than the delay can be used for evaluating early detection performance. We compare the accuracies of different baselines and our models as shown in Figure 4.

The accuracies of all methods increase with elapsed time, and our models demonstrate clear advantages at early stage. Particularly, **GAN-GRU** using less than 12-hour data, has already outperformed **GRU** (the best baseline) using all time data, indicating the superior early detection performance of our model. This is due to the generation component that can enrich training data at early stage when the volume of actual posts is generally low.

[5]This can be confirmed by the relatively higher performance of all models on TWITTER.

Table 4: Examples of original and generated posts

claim	**[Non-rumor:] Augusta County School Close over Arabic Assignment**
Real Posts	Believe that is scary ? Augusta County schools closed Friday ... Schools are shut down for a homework assignment on Arabic Augusta County schools closed Friday... Oh, the stupid -it burns. My god rt : Augusta County schools closed Friday ... Clear violation of church and state. The teacher was clearly wrong ...
$G_{N \to R}$	The assignment in school not well with great wrong , I disagree ... Why close school? seriously? I not believe . not sure but wrong :d I think the assignment will be a mess , do harm to... so disappointed . Those teacher have fault, that was crazy, not a truth , totally a rumor I am confused of not enough report about us. Will be a joke
claim	**[Rumor:] Dearborn Implement Sharia Law**
Real Posts	A satire website produces fake news for entertainment. An oninion-style ... No movement implement Sharia Law Dearborn just sustain Shawarma Law Link me your source, even better, public voting record doesn't exist I can assure you that no one is trying to implement Sharia Law. Is this true? the city council of Dearborn, Michigan became the first
$G_{R \to N}$	A Claim that Law probably offense and out of control :p #Wakeupamerica, A Claim expose that is well great via fully report Omg, a claim earn simultaneously smile and pleasant , Seriously? Report that breaking for nyc, I mean it Awesome! no rumor just fact I discover it a total fact , but claim is illegal, will appear to be wrong .

We also conduct experiments to explain why the generator can boost the discriminator in an adversarial manner. We sample a rumor and a non-rumor claim from TWITTER, and list some generated contents in Table 4. We observe that 1) the generations although seemed non-grammatical are relevant to the input claim; 2) the generations can distort the real-world posts by using conflicting or uncertain expressions; 3) incorporating the generations counterbalances the discriminative power of high-frequency patterns (in yellow), implying a higher chance of lower-frequency features (in blue) being captured by the discriminator.

Given the limited number of claims in the datasets, our models perform reasonably well. This is because there are a good number of relevant tweets per claim, where many indicative patterns exist, such as those highlighted keywords and phrases in Table 4. Therefore, the feature space is not sparse, which is generally advantageous for generators to generate diverse campaign-style texts and for the discriminator to capture discriminative features.

5 CONCLUSION AND FUTURE WORK

We propose a novel GAN-style model that can generate and exploit the effect of information campaigns for better rumor detection. Our neural-network-based generators create training examples to confuse rumor discriminator so that the discriminator are forced to learn more powerful features from the augmented training data. Experimental results confirm that our method is effective and robust based on two public benchmark datasets for rumor detection on Twitter. In our future work, we plan to use GAN to generate structured data such as rumor propagation trees to boost rumor detection performance and compare with structured models [14, 15, 19].

ACKNOWLEDGMENT

This work is partly supported by Innovation and Technology Fund (ITP/004/16LP), TBF (TBF18ENG002) and General Research Fund of Hong Kong (14232816, 14209416, 14204118).

REFERENCES

[1] G.W. Allport and L.J. Postman. 1965. *The psychology of rumor.* Russell & Russell.

[2] Antoine Bordes, Léon Bottou, and Patrick Gallinari. 2009. Sgd-qn: Careful quasi-newton stochastic gradient descent. *Journal of Machine Learning Research* 10, Jul (2009), 1737–1754.

[3] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*. 675–684.

[4] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).

[5] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[6] Emily L Denton, Soumith Chintala, Rob Fergus, et al. 2015. Deep generative image models using a laplacian pyramid of adversarial networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*. 1486–1494.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*. 2672–2680.

[8] Sejeong Kwon, Meeyoung Cha, and Kyomin Jung. 2017. Rumor Detection over Varying Time Windows. *PLOS ONE* 12, 1 (2017), e0168344.

[9] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th International Conference on Data Mining*. 1103–1108.

[10] Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. 2015. Real-time Rumor Debunking on Twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 1867–1870.

[11] Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1412–1421.

[12] Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. 3818–3824.

[13] Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 2015. Detect Rumors Using Time Series of Social Context Information on Microblogging Websites. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 1751–1754.

[14] Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect Rumors in Microblog Posts Using Propagation Structure via Kernel Learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 708–717.

[15] Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor Detection on Twitter with Tree-structured Recursive Neural Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1980–1989.

[16] Feng Qian, Chengyue Gong, Karishma Sharma, and Yan Liu. 2018. Neural user response generator: fake news detection with collective user intelligence. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. AAAI Press, 3834–3840.

[17] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. 3104–3112.

[18] Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, and Jing Gao. 2018. EANN: Event Adversarial Neural Networks for Multi-Modal Fake News Detection. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 849–857.

[19] Ke Wu, Song Yang, and Kenny Q Zhu. 2015. False rumors detection on sina weibo by propagation structures. In *Proceedings of 2015 IEEE 31st International Conference on Data Engineering*. 651–662.

[20] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2017. A convolutional approach for misinformation identification. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 3901–3907.

[21] Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 2015. Enquiring Minds: Early Detection of Rumors in Social Media from Enquiry Posts. In *Proceedings of the 24th International Conference on World Wide Web*. 1395–1405.

[22] Arkaitz Zubiaga, Maria Liakata, and Rob Procter. 2017. Exploiting context for rumour detection in social media. In *International Conference on Social Informatics*. Springer, 109–123.

Parametric Models for Intransitivity in Pairwise Rankings

Rahul Makhijani
Stanford University
rahulmj@stanford.edu

Johan Ugander
Stanford University
jugander@stanford.edu

ABSTRACT

There is a growing need for discrete choice models that account for the complex nature of human choices, escaping traditional behavioral assumptions such as the transitivity of pairwise preferences. Recently, several parametric models of intransitive comparisons have been proposed, but in all cases the model log-likelihood is non-concave, making inference difficult. In this work we generalize this trend, showing that there cannot exist an parametric model that both (i) has a log-likelihood function that is concave in item-level parameters and (ii) can exhibit intransitive preferences. Given this observation, we also contribute a new simple model for analyzing intransitivity in pairwise comparisons, taking inspiration from the Condorcet method (majority vote) in social choice theory. The *majority vote model* we analyze is defined as a voting process over independent Random Utility Models (RUMs). We infer a multidimensional embedding of each object or player, in contrast to the traditional one-dimensional embedding used by models such as the Thurstone or Bradley-Terry-Luce (BTL) models. We show that a three-dimensional majority vote model is capable of modeling arbitrarily strong and long intransitive cycles, and can also represent arbitrary pairwise comparison probabilities on any triplet. We provide experimental results that substantiate our claims regarding the effectiveness of our model in capturing intransitivity for various pairwise choice tasks such as predicting choices in recommendation systems, winners in online video games, and elections.

ACM Reference Format:
Rahul Makhijani and Johan Ugander. 2019. Parametric Models for Intransitivity in Pairwise Rankings. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313717

1 INTRODUCTION

Modeling discrete choice is a key challenge at the heart of many ranking problems, with applications to recommendation systems [8], information retrieval [12], and predicting the outcomes of diverse pairwise competitions [13]. The "choice" being modeled is sometimes a choice made by an individual over alternative options, or sometimes a matter of a competition or game "choosing" a player to win that competition. Prominent examples of ranking systems based on discrete choice models are the the Elo system for Chess ranking and the TrueSkill system [9] for matching players in online games. Meanwhile pairwise comparisons are used in recommendation systems to infer global quality measures from pairwise preferences of users [20], an idea that builds on Thurstone's original observation from the 1920's that comparative judgments are often easier than absolute judgments in many contexts [21].

Two of the most widely used models of ranking from pairwise comparisons are the Thurstone [21] and the Bradley-Terry-Luce (BTL) [3] models. These two models both belong to a broader class of Random Utility Models (RUMs) [1, 14], which assume that each alternative i is completely described by a random utility U_i with a innate quality λ_i, also known as the nominal utility, and a zero-mean noise term ϵ_i such that $U_i = \lambda_i + \epsilon_i$. For a random utility model with n items and utilities $U_1, \ldots, U_n$, it is assumed that the probability that an item is chosen can be written as:

$$P(a \text{ chosen from } S) = Pr(U_a \geq U_c, \forall c \in S),$$

for all possible alternatives $a \in S$ and choice sets S. For pairwise choices, this structure reduces to assuming that choices

$$P_{AB} := \Pr[A \text{ chosen from } \{A, B\}] = \Pr[U_A - U_B > 0].$$

The Thurstone model assumes that each ϵ_i is i.i.d. Normal while the BTL model assumes that each ϵ_i is i.i.d. Gumbel (sometimes also called double-exponential). A model is an *independent* RUM if the random variables ϵ_i are all independent. Thurstone's model and the BTL model are therefore both independent RUMs. For the Thurstone and BTL models, the difference of two utilities $U_A - U_B$ obeys a Normal distribution and Logistic distribution, respectively.

An important general feature of RUMs with i.i.d. noise is that they implicitly assume stochastic transitivity. Informally, if A beats B more often than not and B beats C more often than not, then any such model will assume that A beats C more often than not. The stochastic transitivity of these RUMs follows very specifically from the requirement that the noise be both independent *and* identically distributed. A famous examples of stochastic *intransitivity* is that of *non-transitive dice* [16], which can be formulated as a RUM with independent but non-identical noise distributions. Thus, independent RUMs need not exhibit stochastically transitive choices.

Non-transitive dice raise an intriguing question, namely whether it is possible to efficiently learn the parameters of an independent RUM that models intransitive pairwise comparisons using distributions defined by a small set of parameters (such the moments of the noise distributions). Several models of intransitive comparisons have recently been proposed, notably the 2-dimensional BTL model [4] and the Blade-Chest model [6]. But these and other models suffer from non-concave log-likelihood functions, meaning that maximum likelihood estimation is challenging. In this work we therefore formally ask: is there some overlooked model with a tractable concave log-likelihood that can support intransitive pairwise preferences? Our main result answers this question with a general "no."

Given this result, we investigate interpretable models of intransitivity, drawing inspiration from social choice theory. Under the

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313717

Condorcet method of voting, individual rankings may be transitive but overall the collective social decision procedures are not. While the log-likelihood of this model is still non-concave, we find that standard routines for non-convex optimization work well. Furthermore, the relatively well-understood structure of majority vote allows us to derive a collection of expressivity results regarding what sorts of intransitivity this model can and can't represent.

2 MODELLING INTRANSITIVITY

There are three basic mathematical definitions of stochastic transitivity - weak, strong, and antitransitivity. *Weak stochastic intransitivity* is defined as the existence of three alternatives A, B, C in a set of alternatives such that:

$$P_{AB} > 0.5, \;\; P_{BC} > 0.5, \;\; P_{CA} > 0.5,$$

and we are specifically focused on weak stochastic intransitivity in this work. As we will now show, any RUM with i.i.d. uncertainties, including the Thurstone and BTL models, exhibit weak stochastic transitivity.

Proposition 2.1. *For any Random Utility Model with random utilities $U_i = \lambda_i + \epsilon_i$ with i.i.d. ϵ_i and $\lambda_i \neq \lambda_j, \forall i, j$, weak transitivity holds.*

Proof. For any three alternatives A, B, and C, we have that ϵ_A, ϵ_B, and ϵ_C are i.i.d., implying that $\epsilon_A - \epsilon_B$, $\epsilon_B - \epsilon_C$, and $\epsilon_A - \epsilon_C$ are all identically distributed and also symmetric. We see then that $\Pr[U_i - U_j] > 0.5$ iff $\lambda_i > \lambda_j$, and thus the three alternatives are totally ordered by their means. □

There are three traditional approaches to introducing pairwise intransitivity into choice models: RUMs with possibly dependent distributions, RUMs with non-identical distributions, or departures from the RUM framework. Our work here falls in the last category. The study of dependent RUMs has led to an expansive modeling literature, but inference in these settings is generally quite difficult [23]. For intransitivity from independent but non-identical RUMs, we refer the reader to the literature on non-transitive dice [16], which is bounded in the intransitivity it can achieve, a matter that we elaborate on in Section 5.1. It is also possible to look at non-parametric models where each pairwise probability is its own "parameter" p_{ij}, which can be estimated using maximum likelihood [10]. Chatterjee [5] introduced the *non-parametric Bradley-Terry model* with a strong stochastic transitivity assumption that $p_{ik} \geq p_{jk}$ for $k \neq i, j$ whenever i beats j. This assumption facilitates an efficient consistent estimator. Without this transitivity assumption, a simple empirical estimator would be consistent but not very efficient. This motivates the need for parametric models where the p_{ij} can be expressed as a function of item-level parameters for i and j that may be efficiently estimated.

2.1 Prior work on multidimensional models

The use of higher dimensional scores to model pairwise intransitivity is an idea with a notable history [4, 15] and also recent advancements [6, 7, 17]. A two-dimensional extension of BTL is due to Causeur and Husson [4], parameterized by two-dimensional scores $\lambda_i \in \mathbb{R}^2$, $i = 1, \ldots, n$, as well as a binary matrix $\Sigma \in \{-1, +1\}^{n \times n}$ of pairwise dominances. During estimation, each $\sigma_{ij} \in \Sigma$ is determined from the data, taking the value $+1$ if player i wins the majority of the matches between player i and j, and -1 otherwise. From these parameters, $P_{ij} = S(\sigma_{ij}||\lambda_i - \lambda_j||_2)$, where $S(x) = 1/(1 + \exp(-x))$. Although the Casseur–Husson model can exhibit intransitivity, it is highly sensitive to changes in the empirical data since the dominance parameters Σ depend directly on the pairwise counts, and very little has been established regarding the transitive vs. intransitive properties of this model.

The Casseur–Husson model can be easily extended to obtain a d-dimensional BTL model where the score of each player $\lambda_i \in \mathbb{R}^d$. Note that such extensions require more constraints to impose uniqueness. As an example, for a 3-dimensional BTL of n players, the following constraints are sufficient for identifiability: $\sum_{i=1}^{3} \lambda_i = 0$ and $\lambda_i \cdot \lambda_j = 0 \; \forall i, j \in \{1, 2, 3\}, i \neq j$. The number of parameters for this *3D BTL model* are $3n - 6$ for λ (minus the six constraints) and $\binom{n}{2}$ for Σ. It is straight-forward to verify that the negative log-likelihood function of these extended BTL models are non-convex in general.

A recent model similarly based on a higher-dimensional embedding of alternatives is the blade-chest (BC) model [6, 7]. In this model, every player a is represented by two d-dimensional vectors, a "blade" vector $\mathbf{a}_{\text{blade}}$ and a "chest" vector $\mathbf{a}_{\text{chest}}$. There are two variations of the model, "BC distance" and "BC inner." In both models the probability of player a beating player b in the match is given by $S(Q(a, b))$ where $S(x) = 1/(1 + \exp(-x))$ is again the logistic function and $Q(a, b)$ is a choice of a so-called *match-up function* between players a and b. The match-up functions for the distance and inner variations of the Blade-Chest model are:

$$Q_{\text{dist}}(a, b) = |\mathbf{b}_{\text{blade}} - \mathbf{a}_{\text{chest}}|_2^2 - |\mathbf{b}_{\text{chest}} - \mathbf{a}_{\text{blade}}|_2^2 + \gamma_a - \gamma_b,$$
$$Q_{\text{inner}}(a, b) = \mathbf{b}_{\text{blade}} \cdot \mathbf{a}_{\text{chest}} - \mathbf{b}_{\text{chest}} \cdot \mathbf{a}_{\text{blade}} + \gamma_a - \gamma_b.$$

Here the γ parameters are additional parameters that assure that the BTL model is captured as a clean special case (when all blade and chest vectors are zero vectors). There are $2(d + 1)n$ parameters in such Blade-Chest models. BC models perform well when the dimension is large, specifically $O(n)$, leading to a total of $O(n^2)$ parameters. Performance improvements over BTL are modest when only a few dimensions are employed. High-dimensional BC models are computationally difficult to learn due to the necessary minimization of a non-convex non-separable negative log-likelihood function. These challenges of optimizing and interpreting high-dimensional models motivate the development of our low-dimensional model.

3 NON-CONCAVITY OF INTRANSITIVITY

We now give our main theoretical result, that intransitivity necessarily implies non-concavity for the log-likelihood of any item-parametric choice model. The extended BTL model and both Blade-Chest models are merely different choices of nonlinear pairwise probability functions that fall into this more general analysis. We begin with a useful definition.

Definition 3.1. A *pairwise probability function* $f : \mathbb{R}^d \times \mathbb{R}^d \to [0, 1]$ is a function mapping the d-dimensional parametric representations of two alternatives to a choice probability such that $f(i, j) + f(j, i) = 1, \forall i, j$.

For the BTL and Thurstone models, $f(a,b) = S(a-b)$ and $f(a,b) = \Phi(a-b)$, respectively, for scalars a and b and where S is the logit function and Φ is the cumulative distribution function of the standard Normal. For the extended BTL model we have that $f(i,j) = S(\sigma_{ij}||\lambda_i - \lambda_j||_2)$. For the Blade-Chest model we have e.g. $f_{\text{BC-dist}}(i,j) = S(Q_{\text{dist}}(i,j))$. For the Blade-Chest models we note that the dimension d of the score representation is different from the dimension D of the embedding: $d = 2(D+1)n$.

While high-dimensional BTL and Blade-Chest models are capable of exhibiting intransitivity, their negative log-likelihood functions are in general not convex in the parameters of the alternatives, which typically makes global minimization challenging. This difficulty raises a natural question: are there models of intransitive pairwise choice where the log-likelihood function is concave? Notice that for a generic pairwise probability function $f(i,j)$, the log-likelihood objective of the model given data becomes:

$$\ell(\lambda; \{n_{ij}\}) = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} n_{ij} \log(f(i,j)),$$

where n_{ij} denotes the number of times i is chosen over j.

In order to furnish a concave log-likehood function, we therefore seek pairwise probability functions $f(a,b)$ that are log-concave in the score parameters. Can any such f exhibit intransitivity? We obtain the following negative result.

Theorem 3.2. *For any pairwise probability function $f(\lambda_i, \lambda_j)$ with $\lambda_i, \lambda_j \in \mathbb{R}^d$, f can exhibit an intransitive cycle only if f is not log-concave.*

Proof. Let A, B and C be three objects with score parameters $\lambda_i \in \mathbb{R}^d$ and let them form an intransitive cycle, meaning that

$$\begin{aligned} P_{AB} &= f(\lambda_A, \lambda_B) > 0.5, \\ P_{BC} &= f(\lambda_B, \lambda_C) > 0.5, \\ P_{CA} &= f(\lambda_C, \lambda_A) > 0.5. \end{aligned} \tag{1}$$

Further assume that f is log-concave. We will show that this implies a contradiction.

Notice first that the function f is bounded, $0 \leq f(i,j) \leq 1$, $\forall i, j$. As f is log-concave we also have the following two log-concavity properties for pairs $f(p,q)$ and $f(x,y)$ and $0 \leq \alpha \leq 1$:

$$f(\alpha p + (1-\alpha)x, \alpha q + (1-\alpha)y) \geq \min\{f(p,q), f(x,y)\},$$

$$f(\alpha p + (1-\alpha)x, \alpha q + (1-\alpha)y) \geq f(p,q)^{\alpha} f(x,y)^{1-\alpha}.$$

Using this second log-concavity property for $\alpha = 1/2$ we have

$$f((\lambda_A + \lambda_B)/2, (\lambda_B + \lambda_C)/2) \geq \sqrt{f(\lambda_A, \lambda_B) f(\lambda_B, \lambda_C)}.$$

Further employing $f(\lambda_A, \lambda_B) > 0.5$ and $f(\lambda_B, \lambda_C) > 0.5$, from Equation (1) and the log-concavity, we have

$$f((\lambda_A + \lambda_B)/2, (\lambda_B + \lambda_C)/2) > 0.5. \tag{2}$$

Clearly $f(-\frac{\lambda_B}{2}, -\frac{\lambda_B}{2}) = 0.5$ because it is a pairwise probability function. By convexity between $f(-\frac{\lambda_B}{2}, -\frac{\lambda_B}{2})$ and Equation (2) with $\alpha = 1/2$, we obtain $f(\frac{\lambda_A}{4}, \frac{\lambda_C}{4}) > 0.5$ and hence $f(\frac{\lambda_C}{4}, \frac{\lambda_A}{4}) < 0.5$.

To obtain the contradiction we simply employ log-concavity for the pairs $\{\lambda_C, \lambda_A\}$ and $\{0,0\}$ to obtain $f(\frac{\lambda_C}{4}, \frac{\lambda_A}{4}) > 0.5$, contradicting the earlier conclusion. □

Among the many possible choices of intransitive pairwise probability functions f, the very general family of functions that define a model, we might have suspected that there were some function with favorable inferential properties, i.e. one that yielded an overall log-likelihood function that was concave. The above result tells us that no such function exists, at least for item-level parameterizations. We note that this result covers independent RUMs (non-transitive dice) where one might think it possible to achieve intransitivity using multiple "score parameters" to separately represent the location, variance, and/or skewness of the random utilities. Such a model does have the capacity to model intransitive choice, but the above result applies even to such models, meaning they can not have a concave log-likelihood function.

In the absence of a convexity-based optimization argument for one pairwise probability function over another—motivating one choice of model over another—we now turn to a particularly intuitive and low-dimensional pairwise probability function based on the majority vote model from social choice theory.

4 MAJORITY VOTE MODEL

In this section we present and investigate a model of pairwise comparisons based on the basic concept of majority vote. In the majority vote model, each object (player) has an attribute representation $\mu \in \mathbb{R}^k$, $k \geq 1$ odd, and the pairwise probabilities of choosing an object (player) is modeled as a simple dimension-wise majority vote function over the different attribute dimensions.

Consider a contest where there are n players competing and each player i has a vector of attributes μ_i. A game between players i and j is won by player i if $\mu_i - \mu_j + \xi$ is positive in $(k+1)/2$ or more coordinates, and player j otherwise. Here ξ is a noise vector defined in terms of a difference distribution, a definition due to Yellott [25].

Definition 4.1. If f and F are the probability density function (p.d.f.) and cumulative distribution function (c.d.f.) of two i.i.d. random variables X_1 and X_2, then the *difference density function* and *difference distribution function* d_f and D_f are the p.d.f. and c.d.f. of $X_1 - X_2$ respectively.

The noise vector ξ in the majority vote model is drawn independently for each match-up from a multivariate distribution G with independence between the dimensions such that $G = \prod_{i=1}^{k} g_i$, where g_i is a distribution function such that the difference distribution function of two g_i's is continuous and strictly increasing over $\mathbb{R}$. Two useful examples of g_i are the Normal distribution and the Gumbel distribution, leading to d_{g_i} being Normal and Logistic, respectively. We refer to the model with Gaussian noise as the Gaussian majority vote model. The Gaussian majority vote model reduces to the Thurstone model if the "voting" is conducting using only one dimension.

Although this majority vote model can be extended to arbitrarily large dimensions, we focus our analysis on the simple case of $k = 3$ dimensions and furnish a complete derivation of the maximum likelihood estimate of the parameter representations in this case. This choice of k limits the number of parameters to $3n$, where n is

the number of alternatives (players). We show in Section 5 of this work that three dimensions is sufficient to (1) model strong and long intransitive cycles and (2) completely express the pairwise choice probabilities of any triplet. Based on these arguments we argue that three dimensions are sufficient for many applications. That said, it is straight-forward to generalize the MLE derivation below to arbitrarily large dimensions.

4.1 MLE derivation for three dimensions

For the three-dimensional model we let $M \in \mathbb{R}^{3n}$ denote the matrix of attributes μ of all the n players. The i^{th} row of matrix M is then $\mu_i = (\mu_i^1, \mu_i^2, \mu_i^3)$, the attributes of the i^{th} player. Let $p_1(i,j), p_2(i,j)$, and $p_3(i,j)$ be the probabilities that $\mu_i - \mu_j + \xi$ are positive in the first, second, and third dimensions, respectively. We write $p_k(i,j)$ as p_k for notational simplicity. Hence

$$p_k = D_g(\mu_i^k - \mu_j^k), \quad \forall k = 1,2,3.$$

For additional notational simplicity we define the win probability w_{ij} to be the pairwise probability function f evaluated for two players i and j, were $x \geq y$ indicates that x is greater than y in at least two coordinates:

$$\begin{aligned} w_{ij} &= \Pr[\mu_i - \mu_j + \xi \geq 0] \\ &= p_1p_2p_3 + (1-p_1)p_2p_3 + p_1(1-p_2)p_3 + p_1p_2(1-p_3). \end{aligned}$$

The log-likelihood $\ell(M;N)$ is, dropping additive constants and focusing on only pairs $i < j$:

$$\ell(M;N) = \sum_{i,j:i<j} n_{ij}\log(w_{ij}) + n_{ji}\log(w_{ji}).$$

Here N denotes the pairwise comparison matrix and n_{ij} be the number of times i has been chosen over j. The gradients for the majority vote model can be calculated easily for gradient-based optimization methods [2]. As an example, for the Gaussian majority vote model we obtain the following gradient for parameter μ_{i1}:

$$\frac{\partial \ell}{\partial \mu_{i1}} = \sum_{i<j}\left[\left(\frac{n_{ij}}{w_{ij}} - \frac{n_{ji}}{1-w_{ij}}\right)\phi(\mu_{i1}-\mu_{j1})\Big(p_2 + p_3 - 2p_2p_3\Big)\right].$$

5 CHARACTERISTICS OF THE MAJORITY VOTE MODEL

A possibly desirable property of a model capturing intransitivity is the ability to represent all pairwise win probabilities between players, but the ability to represent any $n \times n$ matrix of pairwise probabilities would clearly require $\binom{n}{2}$ parameters. In this work we are interested in models with few parameters, at most $O(n)$ and specifically $3n$ for our 3D majority vote model, and seek focused results that establish the expressiveness of the majority vote model despite only having $3n$ parameters. Specifically, we show that it is capable of representing arbitrarily long intransitive cycles and that it is capable of representing any submatrix of probabilities between a triplet of players (though this is not the same as simultaneously being able to represent any submatrix on all triplets).

5.1 Cycle expressivity

The random utility model (RUM) framework is an important framework for the characterization of various ranking models [1, 14]. Random Utility Models in general do not necessarily assume stochastic transitivity, not even for independent RUMs: consider the example of non-transitive Efron dice [16] discussed in the introduction. There are, however, limits on how "strong" the intransitivity for independent RUMs can be: Trybula provides conditions for existence of independent random variables $X_1, X_2, ..., X_n$ such that $\min\{P(X_1 > X_2), P(X_2 > X_3), ..., P(X_n > X_1)\}$ is bounded by a constant $c_n < 1$ for all n [24]. The sequence c_n is increasing and converges to 3/4. It is easy to determine that $c_3 = (\sqrt{5}-1)/2$ and $c_4 = 2/3$. In this subsection we show that majority vote models exhibit no such bound on the strength of intransitivity.

Our majority vote model can be understood as an aggregation of independent RUM models. There is no statistical dependence between the random variables used to represent players, yet we will show that the 3D majority vote model exceeds Trybula's bounds. Specifically, we will prove that the intransitivity of the majority vote model is not bounded away from 1 by any constant. As a result we are able to conclude that our model, despite not relying on any dependence between individuals or heterogeneous noise distributions, is qualitatively different from any univariate independent RUM.

Theorem 5.1. *Given $\epsilon > 0$ and n i.i.d. random variables with distribution g whose pairwise difference follow the difference distribution function D_g which is continuous, strictly increasing on $\mathbb{R}$ with finite variance c, there exists an attribute matrix M for n players such that*

$$\min\{P(X_1 > X_2), P(X_2 > X_3), ..., P(X_n > X_1)\} \geq 1 - \epsilon.$$

Proof. The proof is by construction. Let X_i and X_{i+1} denote cyclically consecutive players, where X_n is cyclically adjacent to X_1. For all i we have that:

$$\Pr(X_i > X_{i+1}) = p_2p_3 + p_1p_3 + p_1p_2 - 2p_1p_2p_3.$$

As before, we write $p_k(i,j)$ as p_k when the intended (i,j) pair is unambiguous. As defined before $p_k = D_g(\mu_i^k - \mu_{i+1}^k)$ where $k \in \{1,2,3\}$.

For our construction an appropriate μ is selected for the first three players, where we choose μ_1, μ_2, and μ_3 to be

$$\mu_1 = (a, 3a, -a), \quad \mu_2 = (-a, a, 3a), \quad \mu_3 = (3a, -a, a),$$

where $a \in \mathbb{R}^+$ is a constant to be determined. As D_g is a continuous cumulative distribution function there exists a constant $a_g \in \mathbb{R}^+$ s.t. $\forall a \geq a_g$ for the random variables X_1 and X_2 , $p_1 \geq 1 - \frac{\epsilon}{2}, p_2 \geq 1 - \frac{\epsilon}{2}$. As $p_3 \geq 0$ and $p_1 + p_2 - 2p_1p_2 \geq 0$, we observe that

$$\Pr(X_1 > X_2) \geq p_1p_2,$$

and hence $\Pr(X_1 > X_2) \geq 1 - \epsilon$. A similar argument holds for the pair $\{X_2, X_3\}$.

For players $i \geq 4$ the μ values are defined recursively:

$$\mu_i = (\mu_{i-1} + \mu_1)/2, \text{ for } i = 4, \ldots, n.$$

We aim to show the inequality $\Pr(X_i > X_{i+1}) \geq 1 - \epsilon$ for $i = 3$ and onward. As $p_2 \geq 0$ and $p_1 + p_3 - 2p_1p_3 \geq 0$, we observe that

$$\Pr(X_i > X_{i+1}) \geq p_1p_3.$$

By the given construction of the μ parameters, $\mu_i - \mu_{i+1}$ is decreasing in dimensions one and three for $i = 3, ..., n$. Hence the product p_1p_3 is decreasing for players X_i and X_{i+1} as i is increasing

since g is a strictly increasing function.

Define $h(i,j) = p_1(i,j)p_3(i,j)$ for players i and j. By construction we automatically have $h(3,4) > h(4,5) > \ldots > h(n-1,n) = h(n,1)$. The last equality follows from the fact $\mu_{n-1} - \mu_n = \mu_n - \mu_1$. Hence we only need to prove the inequality $p_1(i,j)p_3(i,j) \geq 1-\epsilon$ for players $i = n, j = 1$. For players X_n and X_1 we then have:

$$\Pr(X_n > X_1) \geq p_1 p_3, \text{ where } p_1 = p_3 = D_g\left(\frac{2a}{2^{n-3}}\right).$$

Now recall Chebyshev's inequality,

$$P(Z \geq z) \leq \frac{Var(Z)}{z^2}.$$

We apply the inequality with Z set to the first and the third dimensions of $X_n - X_1$. As the variance is bounded, $\mathrm{Var}(Z) \leq c$ where c is a positive constant and $\Pr(Z \geq z) = 1 - \Phi(z)$, we have that $D_g(z) \geq 1 - \frac{c}{z^2}$. Using this property with $z = \frac{2a}{2^{n-3}}$, we have

$$\begin{aligned}\Pr(X_n > X_1) &\geq p_1 p_3 \geq \left(1 - \frac{c}{z^2}\right)^2 \geq \left(1 - \frac{2^{2n-8}c}{a^2}\right)^2 \\ &\geq 1 - \frac{2^{2n-7}c}{a^2},\end{aligned}$$

where for the last part we employ the fact that $(1-x)^2 \geq 1-2x$ when $x > 0$. We conclude that $1 - \frac{2^{2n-7}c}{a^2} \geq 1-\epsilon$ when $\epsilon \geq \frac{2^{2n-7}c}{a^2}$. Combining this analysis with the earlier basic analysis for the initial $i = 1, 2, 3$, we can choose a such that $a \geq \max\{a_g, \frac{2^{n-3.5}\sqrt{c}}{\sqrt{\epsilon}}\}$ to complete the proof. □

The above theorem has two important implications. First, large intransitive cycles can be created using the majority vote model in merely three dimensions. Second, we can conclude that the majority vote model for a broad class of difference densities d_g is not equivalent to any independent RUM model.

5.2 Triplet expressivity

The lemma below shows how the three-dimensional majority vote model can represent all the pairwise win probability relations for a set of any three players.

Lemma 5.2. *Given $\epsilon > 0$ and a matrix W of pairwise win probabilities for any three players, there exists an attribute matrix M which can be used to construct a matrix $\tilde{W}$ of pairwise win probabilities from a Majority Vote model such that $\max_{i,j} |W_{ij} - \tilde{W}_{ij}| \leq \epsilon$.*

Proof. The proof is by construction. We are given a pairwise probability matrix W with elements w_{ij}. Let $c_1 = D_g^{-1}(w_{23})$. Define c_2 and c_3 similarly. Let $\alpha_1 = \frac{1-w_{12}}{w_{12}}$. Define α_2 and α_3 similarly in terms of w_{23} and w_{31} respectively. For the three players we can construct M as defined above to be

$$\begin{bmatrix} c_3 & 0 & 0 \\ 0 & -C + c_1 & C \\ C & -C & c_2 \end{bmatrix},$$

where D_g is continuous and strictly increasing and C is a positive constant such that $D_g(-C) \leq \frac{\epsilon}{2}$ and $D_g(C - c_i) \geq 1 - \alpha_i \epsilon$ for $i = 1, 2, 3$. The existence of C is guaranteed by the continuity of D_g.

Table 1: Pairwise comparison datasets.

Dataset	Players	Match-ups	Intransitive triplets
Election A5	16	44298	5
Election A9	12	95888	3
Election A17	13	21037	18
Election A48	10	25848	0
Election A81	11	20803	3
Jester	100	333956	90
Street Fighter (SF)	35	25000	476

We need to prove $|w_{i,j} - \tilde{w}_{i,j}| \leq \epsilon$ for $1 \leq i \neq j \leq 3$. Consider $i = 1$ and $j = 2$. We have $\tilde{w}_{12} = p_1 p_2 + p_2 p_3 + p_3 p_1 - 2p_1 p_2 p_3$. With $p_1 = w_{12}$, $p_3 \leq \frac{\epsilon}{2}$, and $1 - \frac{\alpha_1 \epsilon}{2} \leq p_2 \leq 1$ we have

$$\tilde{w}_{12} \leq w_{12} + (\epsilon/2) + (\epsilon/2)w_{12} \leq w_{12} + \epsilon.$$

Next we need to prove the bound in the other direction. Since

$$\tilde{w}_{12} \geq p_1 p_2 - 2p_1 p_2 p_3,$$

we have $\tilde{w}_{12} \geq w_{12} - \epsilon$. An identical argument can be applied to the other two pairs within the triplet, yielding the requisite bound on the maximum over pairs (i,j), as desired. □

It is not hard to verify that a three-dimensional Majority Vote model is not fully expressive for larger subsets of players (five players or more), but we consider the two expressivity results, for cycles and triplets, to be sufficiently flexible to model realistic competition data. Most significantly, we note that neither of these results can be achieved by any independent RUM.

6 EXPERIMENTAL RESULTS

We now study the predictive performance of the different parametric models of intransitive pairwise choice we have discussed: higher dimensional BTL models, the Blade Chest model, and our Gaussian Majority Vote model where each model is furnished with roughly the same number of free parameters ($\approx 3n$) for comparison. For the 3D BTL model, we do not count the elements of Σ as parameters for this comparison. We test the models on three datasets: a commonly studied set of election datasets (A5, A9, A17, A48) [22], the Jester joke rating dataset (underlying a recommendation system) [11], and the *Super Street Fighter IV* dataset (SF) (pairwise match-ups from competitive play between different video game characters) [6]. The election and Jester datasets are ranking datasets which were converted to pairwise comparisons datasets. Dataset details are given in Table 1, where the intransitive triplets count when the empirical pairwise probabilities violate weak stochastic transitivity.

Each pairwise comparison dataset was randomly split into training and test sets using 5-fold cross-validation, splitting at the matchup level[1]. The main evaluation metric used are root mean square error (RMSE) between the predicted probabilities $\tilde{p}$ from the model and the empirical probabilities $\hat{p}$ from the test data scaled by the number

[1]For datasets derived from rankings, ideally the splitting should be done based on rankings, but original rankings were not always available. We compared the splitting based on rankings vs matches for the Jester dataset, where rankings were available, and did not observe any significant difference in the results or the inference

of players:

$$\text{RMSE} = \frac{1}{n}\sqrt{\sum_{i,j\in n, j\neq i}(\hat{p}(i,j) - \tilde{p}(i,j))^2}.$$

We also consider the log-likelihood of each dataset under the models. The Thurstone and 3D Majority Vote models are nested, allowing us to apply a likelihood ratio test between the model estimates.

The non-concave likelihoods of all models were maximized using the Nelder–Mead simplex algorithm. The optimization was repeated multiple times to help escape possible poor local optima.

The RMSE results are provided in Table 2. We see that across the election datasets there is relatively little difference between the intransitive models. For the Jester dataset our model reports an RMSE that is 32% lower than the best Blade-Chest model. For the Blade-Chest models we again use the best results between "inner" and "dist" versions. That said, we still see significant improvements over a one-dimensional Thurstone model with all our models of intransitivity. And of all the models, we see that our 3D Majority Vote model generally has the lowest RMSE and highest likelihood. This empirical performance is particularly impressive given the simplicity of the model. Meanwhile Table 3 provides a comparison for the average cross-validated log-likelihood for the same datasets. We also perform a chi-square test for the nested models, 3D Gaussian MV and 1D Thurstone [18]. The p-values suggest that the unrestricted model (3D Gaussian MV) better fit the datasets specifically when large number of intransitive triplets are present.

6.1 Spectral analysis of dimension

In order to understand the role of higher dimensions in modeling skill, we perform an analysis of the singular values of the attribute matrix $\hat{M}$ learned from data for our 3D Gaussian Majority Vote model. In general if the dimensions of $\hat{M}$ are uncorrelated then this indicates the presence of multiple uncorrelated skills. If the variance of $\hat{M}$ is concentrated on just one dimension, the competition data can be viewed as mostly transitive. We use the explained variance of the vectors of the singular value decomposition of $\hat{M}$ as measures of how correlated the skill dimensions are in the learned representation. In Table 4 we observe that most, but not all, of the variance is explained by the first singular vector of $\hat{M}$ in all the data sets we examine. The model that is "the least one-dimensional" is the model of the Street Fighter data, in the sense that the leading singular vector explains the last percentage of the variance. If of interest, the leading dimension across which the maximum variance is spread could be used as a method for producing ranked orderings of all the alternatives/players.

Table 2: RMSE results, lowest numbers in bold.

Dataset	Thurstone	3D MV	3D BTL	Blade-Chest
A5	0.193 ±0.014	**0.026 ±0.001**	0.033 ±0.009	0.029 ±0.004
A9	0.228 ±0.006	0.018 ±0.002	0.018 ±0.002	**0.017 ±0.002**
A17	0.305 ±0.013	0.032 ±0.003	0.046 ±0.022	**0.031 ±0.002**
A48	0.258 ±0.010	**0.025 ±0.002**	0.027 ±0.002	0.027 ±0.003
A81	0.203 ±0.007	**0.031 ±0.004**	0.035 ±0.003	0.032 ±0.004
Jester	0.187 ±0.017	**0.021 ±0.001**	0.031 ±0.005	0.031 ±0.002
SF	0.280 ±0.018	**0.06 ±0.003**	0.064 ±0.004	**0.06 ±0.002**

Table 3: Negative Log-likelihood values, highest in bold, and likelihood ratio test p-values between Thurstone and 3D Majority Vote models.

Dataset	Thurstone	3D MV (p-value)	3D BTL	Blade-Chest
A5	22136	**22368** (4.2e-5)	21786	22217
A9	49742	**49909** (<1e-8)	49877	49897
A17	11501	**11523** (8.2e-3)	11468	11508
A48	13692	13703 (0.23)	**13721**	13717
A81	10879	**10890** (0.34)	10852	10876
Jester	170021	**178180** (<1e-8)	171830	171510
SF	13256	**13753** (<1e-8)	13199	13680

Table 4: Fractional variance explained by the singular vectors of the Gaussian Majority Vote model matrix.

Dataset	σ_1	σ_2	σ_3
A5	0.62 ±0.08	0.29 ±0.04	0.09 ±0.04
A9	0.61 ±0.05	0.30 ±0.03	0.08 ±0.05
A17	0.65 ±0.10	0.31 ±0.10	0.03 ±0.01
A48	0.66 ±0.05	0.24 ±0.03	0.09 ±0.04
A81	0.68 ±0.05	0.25 ±0.03	0.07 ±0.02
Jester	0.72 ±0.01	0.18 ±0.01	0.08 ±0.01
SF	0.53 ±0.02	0.28 ±0.02	0.18 ±0.03

Table 5: Variance fraction explained by the singular vectors the Gaussian MV model attribute matrix for synthetic data.

Dataset	σ_1	σ_2	σ_3
Synthetic 1D	0.81 ±0.08	0.13 ±0.05	0.06 ±0.05
Synthetic 2D	0.58 ±0.09	0.26 ±0.05	0.16 ±0.05
Synthetic 3D	0.51 ±0.04	0.29 ±0.05	0.20 ±0.03

To complement the empirical analysis, we also simulate data which is inherently one, two, and three dimensional, respectively. This is done by simulating a 3D Majority Vote model with a μ vector that is drawn from a uniform distribution for each player which is 1, 2, or 3-dimensional. For the 3-dimensional μ, the Gaussian Majority Vote model is then simulated. For the 1-dimensional and 2-dimensional μ, a version of the Gaussian Majority Vote model is then simulated where the excess dimensions are unbiased coin-flips. We use this process to produce a pairwise comparison matrices for 50 players with 10,000 games for each pair. We train the majority vote model based on the the simulated data, repeating the training five times, and calculate the fraction of variance explained by each singular vector of the learned $\hat{\mu}$. We see that even when the data is truly three-dimensional, there is still a "dominant" vector that explains most of the variation. When the data is two or one dimensional, the majority of the variation can be explained by fewer singular vectors, as expected.

7 CONCLUSIONS

We establish that intransitivity implies non-convexity in the log-likelihood objective of item-parametric choice models. This is a broad statement about many models of choice, though we note that convexity is possible for parameterizations that are not item-level [19]. We then introduce the Majority Vote model that can

represent intransitive relations richer than what are possible with independent RUM models. The model exhibits empirical performance that is comparable to earlier multidimensional models, with the added benefit of simplicity and interpretability. Another significant benefit is the ability to interpret the set of three- (or more) dimensional representations as a point cloud in a latent space, and we give a straight-forward spectral approach to measuring the "dimensionality" of intransitive data using singular values.

Acknowledgements. This work was supported in part by an ARO Young Investigator Award.

REFERENCES

[1] Marschak Jacob Block, Henry David. 1960. Random orderings and stochastic theories of responses. *Contributions to probability and statistics* (1960), 97 –132.

[2] Olivier Bottou, Léon; Bousquet. 2008. The Tradeoffs of Large Scale Learning. *Advances in Neural Information Processing Systems* (2008), 161–168.

[3] Ralph Allan Bradley and Milton E. Terry. 1952. Rank analysis of incomplete block designs the method of paired comparisons. *Biometrika, 39(3-4):324–345,* (1952).

[4] D. Causeur and F. Husson. 1 December 2005. A 2-dimensional extension of the Bradley–Terry model for paired comparisons. *Journal of Statistical Planning and Inference, Volume 135, Issue 2* (1 December 2005), 245–259.

[5] Sourav Chatterjee. 2015. Matrix estimation by Universal Singular Value Thresholding. *The Annals of Statistics, 43(1)* (2015), 177 – 214.

[6] Shuo Chen and Thorsten Joachims. 2016. Modeling Intransitivity in Matchup and Comparison Data. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM '16)*. ACM, New York, NY, USA, 227–236.

[7] Shuo Chen and Thorsten Joachims. 2016. Predicting Matchups and Preferences in Context. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 775–784.

[8] David F. Gleich and Lek-heng Lim. 2011. Rank Aggregation via Nuclear Norm Minimization. (2011), 60–68.

[9] Ralf Herbrich, Tom Minka, and Thore Graepel. 2006. TrueSkill™: A Bayesian Skill Rating System. In *Proceedings of the 19th International Conference on Neural Information Processing Systems (NIPS'06)*. MIT Press, Cambridge, MA, USA, 569–576.

[10] R. Hunter. 2004. MM algorithms for generalized Bradley-Terry models. *The Annals of Statistics* 32 (2004), 2004.

[11] D. Gupta K. Goldberg, T. Roeder and C. Perkins. 2001. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval, 4(2):* (2001), 133–151.

[12] Tie-Yan Liu. 2009. Learning to Rank for Information Retrieval. *Found. Trends Inf. Retr.* 3, 3 (March 2009), 225–331.

[13] C. Varin M. Cattelan and D. Firth. 2013. Dynamic Bradley-Terry modelling of sports tournaments. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, (2013).

[14] Charles F. Manski. 1977. The structure of random utility models. *Theory and decision* (1977), 8(3):229–254.

[15] K. O. May. 1954. Intransitivity, utility, and the aggregation of preference patterns. *Econometrica: Journal of the Econometric Society* (1954), 1–13.

[16] Richard P.Savage. 1960. The Paradox of Nontransitive Dice. *The American Mathematical Monthly Vol. 101, No. 5* (1960), 97 –132.

[17] Stephen Ragain and Johan Ugander. 2016. Pairwise Choice Markov Chains. *Advances in Neural Information Processing Systems* (2016), 3198–3206.

[18] Karin Schermelleh-Engel, Helfried Moosbrugger, and Hans Müller. 2003. Evaluating the fit of structural equation models: Tests of significance and descriptive goodness-of-fit measures. *Methods of psychological research online* 8, 2 (2003), 23–74.

[19] Arjun Seshadri, Alexander Peysakhovich, and Johan Ugander. 2019. Discovering Context Effects from Raw Choice Data. *arXiv preprint arXiv:1902.03266* (2019).

[20] Christoph Freudenthaler Steffen Rendle. 2014. Improving pairwise learning for item recommendation from implicit feedback. *Proceedings of the 7th ACM international conference on Web search and data mining* (2014).

[21] Louis L. Thurstone. 1927. A law of comparative judgment. *Psychological review, 34(4):273* (1927).

[22] N. Tideman. 2006. Collective decisions and voting. *Ashgate Burlington* (2006).

[23] Kenneth E Train. 2009. *Discrete Choice Methods with Simulation.* Cambridge University Press.

[24] S Trybula. 1965. On the paradox of n random variables. *Zastos Math 8.* (1965), 143– 154.

[25] J Yellott. 1977. Relationship between Luces choice axiom, thurstones theory of Comparative judgement, and double exponential distribution. *Journal of Mathematical Psychology Vol. 15* (1977), 109–144.

A Large-scale Study on the Risks of the HTML5 WebAPI for Mobile Sensor-based Attacks

Francesco Marcantoni
Univ. of Illinois at Chicago, USA
fmarca2@uic.edu

Michalis Diamantaris
FORTH, Greece
diamant@ics.forth.gr

Sotiris Ioannidis
FORTH, Greece
sotiris@ics.forth.gr

Jason Polakis
Univ. of Illinois at Chicago, USA
polakis@uic.edu

ABSTRACT

Smartphone sensors can be leveraged by malicious apps for a plethora of different attacks, which can also be deployed by malicious websites through the HTML5 WebAPI. In this paper we provide a comprehensive evaluation of the multifaceted threat that mobile web browsing poses to users, by conducting a large-scale study of mobile-specific HTML5 WebAPI calls used in the wild. We build a novel testing infrastructure consisting of actual smartphones on top of a dynamic Android app analysis framework, allowing us to conduct an end-to-end exploration. Our study reveals the extent to which websites are actively leveraging the WebAPI for collecting sensor data, with 2.89% of websites accessing at least one mobile sensor. To provide a comprehensive assessment of the potential risks of this emerging practice, we create a taxonomy of sensor-based attacks from prior studies, and present an in-depth analysis by framing our collected data within that taxonomy. We find that 1.63% of websites could carry out at least one of those attacks. Our findings emphasize the need for a standardized policy across browsers and the ability for users to control what sensor data each website can access.

CCS CONCEPTS

• **Security and privacy → Mobile platform security**.

KEYWORDS

Android; mobile HTML5; sensor attack taxonomy; WebAPI

ACM Reference Format:
Francesco Marcantoni, Michalis Diamantaris, Sotiris Ioannidis, and Jason Polakis. 2019. A Large-scale Study on the Risks of the HTML5 WebAPI for Mobile Sensor-based Attacks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3308558.3313539

1 INTRODUCTION

Smartphones have become almost ubiquitous, with the volume of Internet traffic from mobile devices surpassing that of desktop computers worldwide [35], while 56% of the traffic to top-sites in the US was from mobile devices [65]. Apart from the obvious usability benefits smartphones offer, they have also introduced a plethora of risks. Users are becoming increasingly aware of privacy issues including online tracking and internet surveillance, and employ private browsing among other techniques to remain anonymous online [70, 71]. However, adversaries can still track users through fingerprints [53], making it possible to identify which device is navigating a given webpage [42]. Prior work has also shown that imperfections in sensors' hardware render them fingerprintable [25]. Websites can access such mobile sensor data through the widely supported HTML5 WebAPI. Thus, a plethora of attacks that were previously limited to mobile apps can "migrate" to the mobile web, as modern browsers provide access to a device's sensors.

In this paper we present a *quantitative* and *qualitative* large-scale study of mobile-specific WebAPI calls made by websites in the wild. We build a unique crawling infrastructure that uses Android devices and perform an end-to-end analysis of WebAPI requests. Using our crawling infrastructure, we measure the prevalence of mobile-specific WebAPI calls across 183,571 of the most popular websites during March-November 2018. Our experiments capture the true scale of this phenomenon, as we detect 5,313 unique domains accessing at least one mobile WebAPI call; 35.89% of those also result in sensors being accessed by third-party scripts. To better understand the implications we survey prior literature on attacks from malicious apps that leverage data from mobile sensors. Based on this diverse yet representative selection of papers we create a taxonomy of attacks that could be potentially carried out by modern websites and conduct an in-depth analysis of our dataset. We argue that with browsers enforcing different access policies, as do the plethora of apps that support WebView, there is dire need for a standardized, fine-grained universal mechanism that allows users to control access to *all* types of mobile sensor data.

Overall, this paper makes the following contributions:

- We build a novel crawling infrastructure and conduct a high-fidelity, large-scale end-to-end study of websites targeting mobile-specific sensors.
- We provide a taxonomy of previously reported sensor-based attacks and reframe them within the modern mobile ecosystem. Guided by our taxonomy we conduct a qualitative analysis of our collected data and a provide a comprehensive assessment of the threat posed by the mobile WebAPI.
- We publicly release our data: https://www.cs.uic.edu/~webapi.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313539

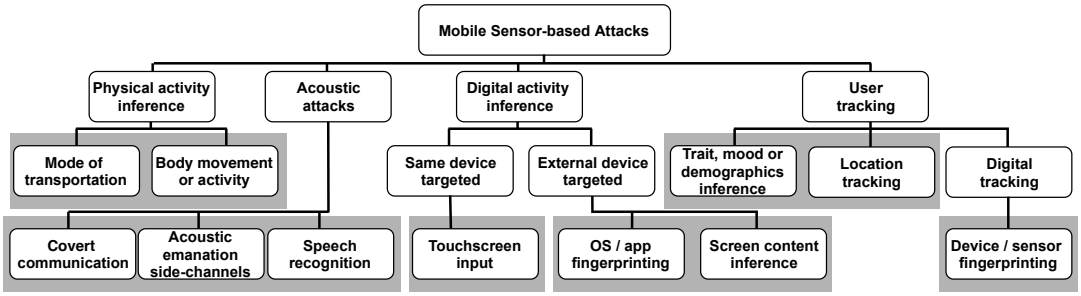


Figure 1: Taxonomy of attacks demonstrated in prior studies that leverage data from mobile sensors.

2 BACKGROUND

Attack Taxonomy. A plethora of research papers have demonstrated mobile-based attacks that employ sensor data. While a considerable number of attacks present similar characteristics, e.g., demonstrating different techniques for inferring a user's touchscreen input or fingerprinting the user's device, a wide range of different attacks have been proposed. Here we introduce a taxonomy of attacks compiled from the literature that captures the vast potential of how mobile sensor data can be misused by adversaries. Typically these attacks assume that attackers are able to obtain sensor data through a malicious app installed on the device. However, in practice, modern browsers can mediate data exchange between websites and sensor data through the HTML5 WebAPI. This leads to a different threat model and an increased attack surface, as it removes the constraint of users having to install a malicious app.

In Figure 1 we present our taxonomy which aims to highlight the variety of attacks enabled by sensor-data, while simultaneously obscuring the type of sensor used for each attack. We do not include explicit sensor information in our taxonomy, as prior attacks often obtain the same objective while using different combinations of sensors (as can be seen in Table 1). At the same time we opt for a relatively fine-grained first level, and specifically consider acoustic attacks as a separate class due to their unique and diverse nature, instead of including them as sub-classes of physical and digital activity inference attacks. Next we describe the 4 main classes from our taxonomy's first level and refer to some of the presented attacks.

Physical activity inference. Numerous studies [27, 36, 38, 46, 52, 58] have demonstrated that mobile sensors can be used to infer information about personal everyday activities. For example it is possible to infer whether the user is walking, running or their mode of transportation, by leveraging the Motion and GPS sensors [58].

Acoustic attacks. [11, 24, 32, 33, 44, 45, 47, 62] showed that access to Motion, Orientation or the Vibration API can be used to infer users' credit card numbers by listening for specific frequencies [62] or what a user is typing on the keyboard [45], and bypassing dynamic analysis and antivirus through covert channel attacks [44].

Digital activity inference. This class includes a wide range of attacks, with prior work [14, 16, 17, 30, 37, 46, 56, 64, 72] showing that sensor information (including the Accelerometer and Gyroscope) can be used to predict what the user is typing on the smartphone's touchscreen(e.g., [46, 56]). This is possible because typing leads to changes in the position of the screen, its orientation and the device's motion. In a different study, the Light sensor was used to identify the content of an external display and even classify users' digital activities into different categories with an 85% accuracy [17].

User tracking. User tracking has garnered much attention [9, 10, 15, 20–23, 25, 30, 36, 39, 40, 47, 52, 58, 59, 73, 74], and can be conducted in different ways – from coarse-grained location tracking that does not require any user-permission (using the Accelerometer or Gyroscope) [36, 52], to fine-grained device fingerprinting using rich and high-resolution data from smartphone sensors(e.g., [10, 25]). In this category we also include alternative attacks that could track users by inferring demographic information (e.g., age [23], gender [47] and fingerprints [30]), physical traits such as their gait [40, 59], or information about their mental state or mood [74].

Deconstructing sensor attacks. Table 1 lists the attacks described in the studies that guided our taxonomy. We classify previous attack papers based on the taxonomy introduced in Figure 1. Subsequently, we break down all the attacks presented in those papers based on the type of sensor data needed to carry out the attacks. For example [58] infers the body movement or activity of a user by accessing the motion and GPS sensors. On the other hand, [52] only requires the orientation sensor, but using the Motion or Magnetometer sensor can further improve the attack.

3 METHODOLOGY AND SYSTEM DESIGN

In this section we present our system design and experimental methodology. We give an overview of our system's architecture, and provide implementation details about the in-line hooking methods.

System architecture. Our system employs a proxy server that intercepts network traffic by using `mitmproxy` [18]. We configured all the Android devices used in our experiments with `mitm`'s certificate in order to intercept both HTTP and HTTPS traffic. The proxy server injects a JavaScript component that hooks and monitors JavaScript calls to mobile-specific WebAPIs. However, our aim is to obtain an in-depth view of sensor data access. In general, browsers are responsible for mediating access between high-level JavaScript function calls and low-level Android API calls. Understanding how this mechanism works for every browser would be time consuming and in many cases infeasible due to proprietary code. As such, we have opted for a generic and browser-agnostic approach, where we intercept Android system calls using a custom Xposed [60] module that (i) detects and hooks requests to sensor-specific Android API calls and (ii) identifies which of these API calls are permission-protected. By intercepting these low-level function calls we are able

Table 1: Sensor based side-channel attacks.

Ref.#	Attack	Motion	Orien/tion	Light	Vibr/tion	Magn/ter	Media (Cam,Mic)		GPS
[58]	Mode of transportation	◖	-	-	-	-	-	-	◖
[36, 38]		●	-	-	-	-	-	-	-
[52]		○	●	-	-	○	-	-	-
[58]	Movement/activity	◖	-	-	-	-	-	-	◖
[46]		◖	◖	-	-	-	-	-	-
[58]	Location Tracking	◖	-	-	-	-	-	-	◖
[36]		●	-	-	-	-	-	-	-
[52]		○	●	-	-	○	-	-	-
[45]	Acoustic emanation side-channels	●	-	-	-	-	-	-	-
[32, 33]		-	-	-	-	-	-	●	-
[47]	Speech recognition	-	●	-	-	-	-	-	-
[11]		◖	◖	-	-	-	-	-	-
[16, 37, 46, 72]	Touchscreen input	◖	◖	-	-	-	-	-	-
[56]		●	-	-	-	-	-	-	-
[64]		-	-	●	-	-	-	-	-
[30]		-	-	-	-	-	●	-	-
[14]	OS/app fingerprinting	-	-	-	-	●	-	-	-
[17]	Screen inference	-	-	●	-	-	-	-	-
[23, 59, 74]	Physical or demographics	●	-	-	-	-	-	-	-
[47]		-	●	-	-	-	-	-	-
[30]		-	-	-	-	-	●	-	-
[40]		◖	◖	-	-	-	-	-	-
[21, 22, 39]	Device/sensor fingerprinting	◖	◖	-	-	-	-	-	-
[15]		●	-	-	-	-	-	●	-
[10]		●,◖	●	-	-	-	◖	-	-
[25]		●	-	-	-	-	-	-	-
[20, 73]		-	-	-	-	-	-	●	-
[9]		◖	◖	-	-	◖	-	◖	-
[44]	Covert communication side-channels	-	-	-	●	-	-	-	-
[62]		-	-	-	-	-	-	●	-
[24]		◖	-	-	◖	-	-	-	-

Sensors marked with (●) are sufficient for performing the specific attack. When a combination of multiple sensors is required to perform the attack, they are marked with (◖). We denote optional sensors that enhances the accuracy of the attack with (○). Grey columns denote sensor data that should require explicit user permission according to the W3C.

to validate that the JavaScript interception was successful and calls requesting sensor data were correctly logged.

Mobile HTML5 Functions. We identified the functions that retrieve mobile-specific data through the official mobile HTML5 WebAPI [31].We consider as *mobile-specific* any calls that obtain information originating from an integrated sensor of a mobile device. The HTML5 WebAPI calls interact with the webpage using either a direct one-time communication (i.e., Vibration and Media capture)

or through an event listener, since some sensors (i.e., Motion, Orientation, Proximity, and Ambient Light) continuously fire events in order to provide up-to-date readings in real time. For Geolocation one call exists for each category.

JavaScript Calls Interception. We build our component for hooking JavaScript methods upon the `javascript-hooker` Node.js module [13]. In our experiments we do not overwrite the original function but only need to identify whether a function is called. Thus, whenever a WebAPI is called the JavaScript modules creates a log entry for further analysis and executes the original function. Since `javascript-hooker` also takes the arguments of the original function, we can also intercept the arguments of the `addEventListener` and check for events of interest. Our code is injected in the `head` of the document (if there is one) or the page `body` otherwise.

In order to listen to events and associate a function to a specific target we need to intercept the `setter` property. Even though this is possible using `Object.defineProperty()` the original value will be lost and the webpage may not function as expected. Therefore, we follow the approach employed by Chameleon [34] and overwrite the `getter` property of each event prototype. As such, every time the property is read, our custom function is called. While certain sensor data may normally remain the same during navigation (e.g., properties related to display characteristics), remaining constant might be considered "suspicious" for other sensors. For instance, when an actual human uses the device, small changes in the gyroscope readings would be expected. As such, to make our crawling more realistic, our system intercepts the values returned by certain sensors and slightly modifies their value. In general, data retrieved through events, is handled in two different ways: listening to `addEventListener` on the target object while checking if the argument matches the desired event and defining new `getters` for the properties of the event's prototype.

Identifying the JavaScript source. Apart from logging WebAPI calls we also want to identify the origin of the JavaScript files being executed. This information is important in order to identify if the script belongs to a first-party domain or a third-party domain. We register the source of the URL by utilizing the `stack` property of the `Error` object. Our hooking script implements a mechanism that creates an `Error` object and reads its `stack` property.

Android API call interception. Each mobile HTML5 WebAPI is associated with a low-level Android API call. In order to validate the results of the JavaScript interception and to identify which ones require a permission, we use the PermissionHarvester [26] module that hooks every Android permission protected API call. Since access to some of the sensors does not require an Android permission, we also manually identified and hooked the functions that give access to non-permission-protected sensor data. Android apps (including the browser) cannot directly read the current value of a sensor and are required to register a listener in order to consequently read the captured events. Each sensor can be obtained by declaring a listener and specifying its name with the `getDefaultSensor()` function. Then, the listener is registered using the `registerListener()` method. Our module intercepts both of these function calls.

Experimental setup. In our experiments we use Mozilla Firefox (v.59.0.1) as our browser on three Android Google Nexus 5X and a OnePlus One device, all running AOSP 7.1.2. We controlled the devices using the Android Debug Bridge. We first evaluated the effectiveness of our methodology by creating a dummy website that executes all possible mobile HTML5 WebAPIs. Since browsers require a valid certificate in order to call certain APIs we used a self-signed certificate and confirmed that our approach can successfully intercept and monitor access to the devices' sensors. Our system also simulates brief user interaction through random gestures which are issued for approximately 30 seconds on average for each website, while an extra module monitors for potential redirections to different domains which are rolled back so the original website can continue to be processed.

Table 2: Number of domains using mobile WebAPIs calls.

WebAPI	#Domains	WebAPI	#Domains
Device orientation	2,199	Ambient light sensor	152
Geolocation	1,688	Proximity sensor	142
Device motion	1,360	Vibration	84
Screen orientation change	645	Media capture	12
Total 6,282			

4 DATA COLLECTION AND ANALYSIS

Our crawling list included the 200K most popular websites according to Alexa, as returned on 03/24/2018. Our system was unable to access or complete the crawling process for 16,199 (8%) of the domains in our list (e.g., 503-timeout, 502-Bad Gateway, or DNS errors), and omitted 230 domains flagged as malicious by the Google SafeBrowsing API. Our crawling experiments took place between 03/24/2018-09/03/2018 and 11/11/2018-11/22/2018, from US-based IP addresses.

In Table 2 we can see the prevalence of the mobile-specific WebAPI calls logged by our system among the 183,571 domains processed by our crawling infrastructure. We logged 5,313 (2.89%) websites using at least one of the targeted APIs, while 807 request access to sensor data using more than one of the API calls. The most prevalently accessed data is from the acceleration and orientation sensors which do not require the user's permission, as well as geolocation data which requires permission in major browsers. While the Geolocation API can also return information for desktop computers (using "information about nearby wireless access points and the IP address" [4]), we consider it mobile-specific due to smartphones' integrated GPS receivers which provide real-time location information. While geolocating users based on landline IP addresses is considerably accurate [68], that is not the case for mobile IP addresses [12, 66]. It is important to note that the Media capture and Geolocation APIs should explicitly request permissions from the user; while this is enforced in major browsers, it is not always the case with other browsers (e.g., for Geolocation [41]). For the remaining WebAPI calls, users will be unaware that such information is being retrieved by the website even for major browsers.

As Figure 2 shows the use of mobile-specific WebAPI calls is not uniform across our dataset. The highest concentration is found in the top 5K websites with 250 domains. We observe that domains also access more *permission-free* WebAPIs (gray bars) independently of their rank, indicating the importance of this sensor data. This type of information can be used for a plethora of attacks, and users should have the ability to explicitly grant permission for them.

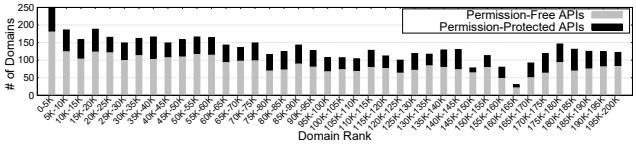


Figure 2: Number of domains using mobile WebAPI calls.

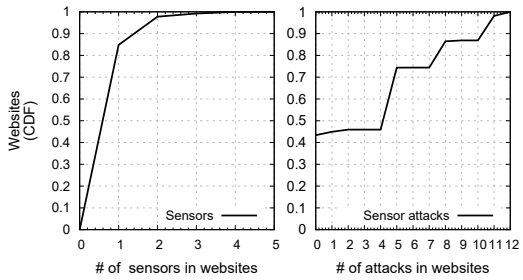


Figure 3: CDF of websites requesting access to sensors (left) and the number of feasible attacks (right).

Table 3: Breakdown of sensor based attacks and the number of domains capable of deploying them.

ID	Mobile Sensor-based Attack	#Domains
1	Mode of transportation	2,861
2	Body movement or activity	720
3	Location tracking	2,861
4	Acoustic emanation side-channels	1,372
5	Speech recognition	2,199
6	Touchscreen input	1,425
7	Screen content inference	152
8	Inferring user's age	1,360
9	Inferring user's sex	2,199
10	Inferring user's fingerprints	12
11	Inferring user's gait	1,360
12	Inferring user's mood	1,360
13	Device/sensor fingerprinting	2,873
14	Covert channels	96

As Figure 3 (left) shows the majority of websites issue request access to a single sensor through the WebAPI, while 15.1% of the domains we processed target at least two different types of sensor data. As shown in Table 1, only accessing the Motion sensor can lead to six different attacks, while a combination of two sensors (Motion and Orientation) leads to eight attacks. Furthermore, as can be seen in Figure 3 (right) 56.6% of the domains that issue mobile-specific WebAPI calls are able to perform at least one attack.

Sensor-based attacks. We continue our analysis by framing our dataset within our taxonomy based on representative prior work. It is important to note that in our analysis we do not take into account or argue for (or against) the *plausibility* of the attacks presented in previous studies. Instead, our goal is to measure the potential risk that mobile users face due to web browsing by identifying websites that request access to specific sensor data and could *potentially* misuse them in an invasive or malicious manner.

Table 3 breaks down the number of domains for each attack. We observe that the most common attacks across websites that access WebAPIs are device/sensor fingerprinting and trait, mood or demographic inference (54.07%), location tracking and mode of transportation (53.84%), speech recognition (41.38%) and touchscreen input (26.82%). We argue that any information gained from sensors poses a risk for users and an access control policy should be enforced, either through some form of run-time permissions [5] or using a mechanism similar to GDPR [3].

Banking sites. While fingerprinting allows third parties to track users across the Web [53], fingerprints can be used as an additional factor for authentication [8]. As such, banking websites are well-suited for deploying such a security mechanism [54] due to the significant implications of compromised accounts. As details of such practices are not typically disclosed, we further explore the prevalence of sensor-based information access across e-banking domains. We compiled a list of bank domains using [1, 6] and cross-referenced it with our dataset. We identified 65 banking domains that request access to at least one mobile sensor. Banking domains request access for 1.38 sensors on average, which is higher than the average of 1.17 in other domains, indicating that they are more likely to leverage the HTML5 WebAPI for accessing sensor data. We find that 24 of the domains obtain access to the sensor data necessary to conduct at least one of the attacks included in our taxonomy. Interestingly, all of those banks collect the sensor data leveraged in prior work for device fingerprinting, while 40 banks request access to the user's geolocation which can also be used for enhancing the authentication process [8]. We find that efirstbank.com actually requests access to more sensors than any other domain in our entire dataset. Overall, while accessing sensor data could be motivated by enhancing the authentication process, this practice raises privacy concerns as argued by privacy advocates [48].

Request origin. Next we explore the origin of WebAPI requests (first or third party) and whether it was included in an iframe. Different browsers implement different policies regarding which sensors can be accessed for these three different types of origin. We present statistics for all the websites that requested access, even if those requests were blocked by Firefox during our experiments.

Iframes. Our system collects all the calls executed by every element of a website, including iframes. In every log we record the source domain name of the element that is accessing sensor information. By comparing the URL of the address bar and the URL in the logfiles, we can identify whether WebAPIs are accessed by the DOM or by an iframe. Our analysis shows that 991 websites out of 5,313 contain iframes that use WebAPIs to access mobile specific information. We analyzed all iframes from our experiments and found that specific iframes are found in different websites. The two most frequent domains injected inside iframes exist in 389 webpages (or 39.3% of pages with iframes collecting data) and are related to online media players.

External sources. Among the websites that issue API calls for mobile specific information we found 40 scripts from external domains (either as a third-party scripts or inside an iframe) that collect data from 2461 websites 46.3%. We manually analyzed these scripts and found that they offer services for media-players and advertisements and they collect information about the orientation and motion of the device. In Table 4 we list the domains that appear in more than 50 websites and collect data from sensors. The first column is the origin of the script being executed. The second and third column show how many websites and iframes host this script. Given that

these third-party domains are used in 35.89% of websites that access sensor data, we classified them based on the type of service they provide using *Cyren*. The last two columns show which sensors the script accessed and their corresponding attacks. We observe that most of these domains call the motion and orientation WebAPIs which enable a plethora of attacks. Moreover, domains classified as search engines and ad-networks gain access to characteristics that can track users across the web.

From Table 4 we can see that the domain *api.b2c.com* enables 12 different attacks. After investigating this domain through VirusTotal [7] we found that scripts served from this domain and Android apps that communicate with it are classified as intrusive adware and even malware by some antivirus vendors. Another domain, *c.adsco.re*, is flagged as malware by Cyren, even though it is not considered malicious by the Google SafeBrowsing API. We manually analyzed the content of the script that retrieves the data and found that apart from retrieving information about the Motion and the Orientation sensors it also exhibits behavior which is a strong indicator of device fingerprinting, such as creating and manipulating canvas elements [50] and reading different Navigator, Screen, Storage and Window properties. Interestingly the *adsco.re* domain states that it is used for traffic validation by Adscore, a bot detection service. In total, these two domains which are considered malicious by certain security lists, were found on 5.4% of all the sensor-accessing domains logged by our system, which again raises concerns regarding browser policies that allow third party domains to access sensor data without explicit user permission.

Android internals. Our crawling system allows an end-to-end analysis of sensor data access. Apart from providing high call-detection fidelity, since we can match requests logged by our injected JavaScript to actions at the operating system level, it also revealed sub-optimal browser behavior. We found that while Firefox prevents iframes from accessing sensor data, in practice Firefox simply "omits" returning the sensor data instead of blocking (i.e., ignoring) the actual request. Specifically, Firefox allows iframes to create event listeners, which then trigger the necessary WebAPI calls which then trigger the corresponding Android-level processing and permission checks for obtaining the sensor data; the data is then returned to the browser but not provided to the iframe.

Malicious domains. Even though our system checked Google's SafeBrowsing API before visiting a domain, it is possible that visited domains could be flagged as malicious later on, or by different blacklists. As such, we submitted all the domains that issued WebAPI requests to VirusTotal. Figure 4 presents the websites flagged as malicious (sorted by their rank), the number of accessed sensors per website and feasible attacks. Out of those, 149 domains were flagged by one AV engine and 17 domains were flagged by two. We can see that higher ranking malicious domains are more likely to access more sensors which results in a higher number of feasible attacks. We found 11 websites being flagged by at least 3 AV engines. Finally, we found two websites, namely *goggle.com* and *yotube.com*, that are flagged by eight AV engines as malicious. Apart from likely examples of typosquatting [49, 69], these websites requested access to sensor data that could be used to perform one and eleven different attacks respectively.

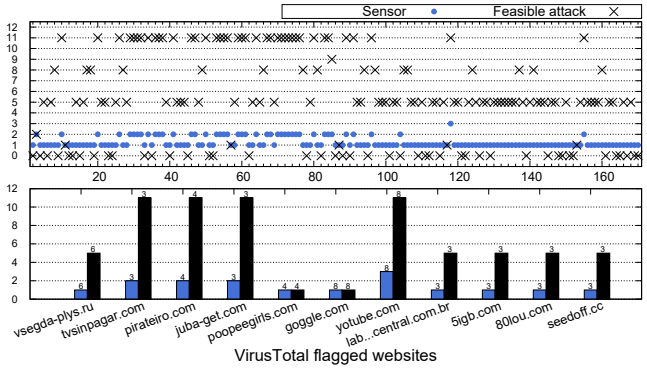


Figure 4: Number of accessed sensors and feasible attacks for websites flagged as malicious (top). Websites that were flagged by at least three AV engines (bottom) – the number on the bars shows how many AV engines flagged the domain.

Table 4: Third-party scripts accessing WebAPI calls.

Script origin	#Sites	iframes	Sensors	AttackID
f.vimeocdn.com	275	275	O	1, 3, 5, 9
fast.wistia.com	467	3	OC	-
fast.wistia.net	125	115	OC	-
c.adsco.re	211	-	M,O	1-6, 8, 9, 11-13
g.alicdn.com	170	65	O,G	1, 3, 5, 9
aeu.alicdn.com	127	83	O	1, 3, 5, 9
api.b2c.com	76	-	M,O,P,L	1-9, 11-13
cdn.admixer.net	169	-	M	1, 3, 4, 6, 8, 11-13
static.yieldmo.com	107	-	O	1, 3, 5, 9
secure-ds.serving-sys.com	51	35	M	1, 3, 4, 6, 8, 11-13
dlswbr.baidu.com	77	69	M	1, 3, 4, 6, 8, 11-13
client.perimeterx.net	73	-	M	1, 3, 4, 6, 8, 11-13

M: motion, G: geolocation, P: proximity, O: orientation, L: light, OC: orientation_change

WebView usage is extremely widespread [51], so we tested three popular WebView-based browsers, namely Dolphin, UC, and WebView (info.android1.webview), along with Facebook and Messenger, and found that they all allow iframes to obtain motion and orientation data. As such, even if users use Firefox or Chrome for browsing, which currently block iframes from accessing sensor data, clicking a link within such popular apps can expose them to attacks.

Transience of web measurements. Scheitle et al. [61] found significant fluctuation in the websites contained in ranking lists used by academic studies, with Alexa being the most volatile list. As a result, similar measurement experiments that use an Alexa list from a different date could result in a significantly different view of the web ecosystem. To quantify and frame this effect within the dataset we have collected, we compare to the recently released dataset by Das et al. [19] which was part of their concurrent study on mobile sensor fingerprinting. While their collection set up was different (they used a modified version of OpenWPM as opposed to actual mobile devices) they also logged mobile sensor APIs used by popular websites. When comparing the domains that accessed mobile-specific WebAPI calls during our experiments to those in their dataset, we find only 403 overlapping domains – 7.9% of our detected websites. However, our system detected WebAPI calls in 2,252 domains that are in their two US-based datasets but with no calls logged during their experiments. Given that both of our experiments were conducted at similar times, including some overlap in May 2018, and used Alexa's list (our version is from 03/24/2018 while their version is from 05/12/2018), this is a surprising result.

Another important dimension that needs to be considered is that the modern web is highly dynamic and websites often introduce new functionality or may even remove existing functionality. To further explore how a view of the web can change through time, we compare the actual WebAPI calls reported for those 403 overlapping domains. While we find that for the vast majority (91.8%) of domains both datasets report the same calls across the two datasets, there are differences for 33 websites. In more detail, for those domains our system logged a total of 74 WebAPI calls, while the datasets from [19] contain 62 calls. This difference is partially due to that study targeting a *subset* of the calls that our study explores. However, there are other domains [2] where the two datasets report different sensor data being requested, which correspond to ~ 3.47% of the domains detected by both systems. While that number is not very large, it is non-negligible and highlights the dynamic and ever-evolving nature of the web.

5 RELATED WORK

The WebAPI has standardized many features providing greater support for developers and improving the user experience [57]. Snyder et al. [63] presented a cost-benefit analysis of the WebAPI using a small set of websites (10K) and focusing on desktop browsing.

In an independent and concurrent recent study Das et al. [19] presented a study on web scripts accessing mobile sensors. While their study also targets WebAPI calls for mobile sensors, our work presents significant differences. In regards to the actual datasets, our study is on a considerably larger set of domains while also having little overlap due to the fluctuation of the Alexa list [61]. Moreover, their system detects a subset of the mobile-specific WebAPI calls handled by our system, and their study focuses on sensor-based fingerprinting thus offering a limited examination of the risks that users face; we frame our findings within our attack taxonomy and provide a more comprehensive evaluation of the feasibility of a wide range of sensor-based attacks. Furthermore, our crawling infrastructure uses actual mobile devices and provides a unique end-to-end view of data requests and access, while their crawlers rely on a modified version of OpenWPM running on desktop machines which could be detected by evasive websites [43].

Browser fingerprinting has gathered a lot of attention and the research community has extensively studied the techniques that make it possible [28, 29]. With the growing usage of smartphones, traditional desktop fingerprinting techniques [67] are becoming less effective as some information is being standardized in many mobile browsers [39]. On the other hand, the development of new mobile-specific HTML5 WebAPIs offered new avenues for trackers to exploit other types of data that were not present in desktops. As previous work [9, 10, 15, 20–23, 25, 30, 36, 39, 40, 47, 52, 55, 58, 59, 73, 74] has shown, the huge amount of input collected by smartphones sensors resulted in new opportunities for device fingerprinting.

6 CONCLUSION

We presented a comprehensive evaluation of the threats that mobile users face when browsing the Web, due to capabilities offered by modern browsers, by conducting the largest and most extensive study to date on the use of mobile-specific WebAPI calls in the wild. Our study was conducted using a novel crawling infrastructure built on top of actual smartphones. Our findings demonstrate that WebAPI capabilities are actively being used by websites for accessing mobile sensors. To provide the appropriate context that highlights the true threat posed by this practice, we created a taxonomy of sensor-based attacks compiled from a wide range of attacks demonstrated in prior work. Our subsequent in-depth analysis correlated the sensor data currently being accessed by websites and the data-requirements of prior attacks, leading to several alarming findings. We argue that our findings support the need for more stringent policies for websites attempting to access sensor data.

ACKNOWLEDGMENTS

This project has received funding from Horizon 2020 under grant agreements No 786890 and No 740787, the DARPA ASED Program and AFRL under contract FA8650-18-C-7880. This paper reflects only the view of the authors and the funding bodies are not responsible for any use that may be made of the information it contains.

REFERENCES

[1] [n. d.]. Alexa - Top 50 Banks and Institutions. https://www.alexa.com/topsites/category/Business/Financial_Services/Banking_Services/Banks_and_Institutions.
[2] [n. d.]. Domains exhibiting differences in the WebAPI calls reported by our system and Das et al. https://www.cs.uic.edu/~webapi/.
[3] [n. d.]. The EU General Data Protection Regulation. https://eugdpr.org.
[4] [n. d.]. Mozilla Support - Does Firefox share my location with websites? https://support.mozilla.org/en-US/kb/does-firefox-share-my-location-websites.
[5] [n. d.]. Request prompts for dangerous permissions. https://developer.android.com/guide/topics/permissions/overview#dangerous-permission-prompt.
[6] [n. d.]. US Banks on the Internet. http://www.thecommunitybanker.com/bank_links/.
[7] [n. d.]. VirusTotal: Analyze suspicious files and URLs to detect types of malware. https://www.virustotal.com.
[8] Furkan Alaca and Paul C van Oorschot. 2016. Device fingerprinting for augmenting web authentication: classification and analysis of methods. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*. ACM, 289–301.
[9] Irene Amerini, Rudy Becarelli, Roberto Caldelli, Alessio Melani, and Moreno Niccolai. 2017. Smartphone fingerprinting combining features of on-board sensors. *IEEE Transactions on Information Forensics and Security* 12, 10 (2017), 2457–2466.
[10] Irene Amerini, Paolo Bestagini, Luca Bondi, Roberto Caldelli, Matteo Casini, and Stefano Tubaro. 2016. Robust smartphone fingerprint by mixing device sensors features for mobile strong authentication. *Electronic Imaging* 2016, 8 (2016), 1–8.
[11] S Abhishek Anand and Nitesh Saxena. 2018. Speechless: Analyzing the Threat to Speech Privacy from Smartphone Motion Sensors. In *2018 IEEE Symposium on Security and Privacy (SP). Vol. 00*. 116–133.
[12] Mahesh Balakrishnan, Iqbal Mohomed, and Venugopalan Ramasubramanian. [n. d.]. Where's That Phone?: Geolocating IP Addresses on 3G Networks. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement (IMC '09)*.
[13] Ben Alman. [n. d.]. Monkey-patch (hook) functions for debugging and stuff. https://github.com/cowboy/javascript-hooker. Accessed: 2018-04-23.
[14] Sebastian Biedermann, Stefan Katzenbeisser, and Jakub Szefer. 2015. Hard drive side-channel attacks using smartphone magnetic field sensors. In *International Conference on Financial Cryptography and Data Security*. Springer, 489–496.
[15] Hristo Bojinov, Yan Michalevsky, Gabi Nakibly, and Dan Boneh. 2014. Mobile device identification via sensor fingerprinting. *arXiv preprint arXiv:1408.1416* (2014).
[16] Liang Cai and Hao Chen. 2011. TouchLogger: Inferring Keystrokes on Touch Screen from Smartphone Motion. *HotSec* 11 (2011), 9–9.
[17] Supriyo Chakraborty, Wentao Ouyang, and Mani Srivastava. 2017. LightSpy: Optical eavesdropping on displays using light sensors on mobile devices. In *Big Data (Big Data), 2017 IEEE International Conference on*. IEEE, 2980–2989.
[18] Cortesi, Aldo and Hils, Mayimilian and Kriechbaumer, Thomas. [n. d.]. mitmproxy. https://mitmproxy.org. v. 3.0.3.
[19] Anupam Das, Gunes Acar, Nikita Borisov, and Amogh Pradeep. 2018. The Web's Sixth Sense: A Study of Scripts Accessing Smartphone Sensors. In *Proceedings of ACM CCS, October 2018*.
[20] Anupam Das, Nikita Borisov, and Matthew Caesar. 2014. Do you hear what i hear?: Fingerprinting smart devices through embedded acoustic components. In

Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM, 441–452.

[21] Anupam Das, Nikita Borisov, and Matthew Caesar. 2016. Tracking Mobile Web Users Through Motion Sensors: Attacks and Defenses.. In *NDSS*.

[22] Anupam Das, Nikita Borisov, and Edward Chou. 2018. Every Move You Make: Exploring Practical Issues in Smartphone Motion Sensor Fingerprinting and Countermeasures. *Proceedings on Privacy Enhancing Technologies* 2018, 1 (2018), 88–108.

[23] Erhan Davarci, Betul Soysal, Imran Erguler, Sabri Orhun Aydin, Onur Dincer, and Emin Anarim. 2017. Age group detection using smartphone motion sensors. In *Signal Processing Conference (EUSIPCO), 2017 25th European*. IEEE, 2201–2205.

[24] Luke Deshotels. 2014. Inaudible Sound as a Covert Channel in Mobile Devices.. In *WOOT*.

[25] Sanorita Dey, Nirupam Roy, Wenyuan Xu, Romit Roy Choudhury, and Srihari Nelakuditi. 2014. AccelPrint: Imperfections of Accelerometers Make Smartphones Trackable.. In *NDSS*.

[26] Michalis Diamantaris, Elias P. Papadopoulos, Evangelos P. Markatos, Sotiris Ioannidis, and Jason Polakis. 2019. REAPER: Real-time App Analysis for Augmenting the Android Permission System. In *9th ACM Conference on Data and Application Security and Privacy, CODASPY '19*. ACM.

[27] Kostas Drakonakis, Panagiotis Ilia, Sotiris Ioannidis, and Jason Polakis. 2019. Please Forget Where I Was Last Summer: The Privacy Risks of Public Location (Meta)Data. In *26th Annual Network and Distributed System Security Symposium (NDSS '19)*.

[28] Peter Eckersley. 2010. How unique is your web browser?. In *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 1–18.

[29] Steven Englehardt and Arvind Narayanan. 2016. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1388–1401.

[30] Tobias Fiebig, Jan Krissler, and Ronny Hänsch. 2014. Security Impact of High Resolution Smartphone Cameras.. In *WOOT*.

[31] Maximiliano Firtman. [n. d.]. Mobile HTML5 Compatibility on Mobile Devices. http://mobilehtml5.org/. Accessed: 2018-04-22.

[32] Daniel Genkin, Mihir Pattani, Roei Schuster, and Eran Tromer. 2018. Synesthesia: Detecting Screen Content via Remote Acoustic Side Channels. In *International cryptology conference*.

[33] Daniel Genkin, Adi Shamir, and Eran Tromer. 2014. RSA key extraction via low-bandwidth acoustic cryptanalysis. In *International cryptology conference*. Springer, 444–461.

[34] ghostwords. [n. d.]. Browser fingerprinting protection for everybody. https://github.com/ghostwords/chameleon. Accessed: 2018-06-09.

[35] Global Stats. [n. d.]. Mobile and tablet internet usage exceeds desktop for first time worldwide. http://gs.statcounter.com/press/mobile-and-tablet-internet-usage-exceeds-desktop-for-first-time-worldwide. Accessed: 2018-04-19.

[36] Jun Han, Emmanuel Owusu, Le T Nguyen, Adrian Perrig, and Joy Zhang. 2012. Accomplice: Location inference using accelerometers on smartphones. In *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*. IEEE, 1–9.

[37] Duncan Hodges and Oliver Buckley. 2018. Reconstructing what you said: Text Inference using Smartphone Motion. *IEEE Transactions on Mobile Computing* (2018).

[38] Jingyu Hua, Zhenyu Shen, and Sheng Zhong. 2017. We can track you if you take the metro: Tracking metro riders using accelerometers on smartphones. *IEEE Transactions on Information Forensics and Security* 12, 2 (2017), 286–297.

[39] Thomas Hupperich, Davide Maiorca, Marc Kührer, Thorsten Holz, and Giorgio Giacinto. 2015. On the robustness of mobile device fingerprinting: Can mobile users escape modern web-tracking mechanisms?. In *Proceedings of the 31st Annual Computer Security Applications Conference*. ACM, 191–200.

[40] Felix Juefei-Xu, Chandrasekhar Bhagavatula, Aaron Jaech, Unni Prasad, and Marios Savvides. 2012. Gait-id on the move: Pace independent human identification using cell phone accelerometer dynamics. In *Biometrics: Theory, Applications and Systems (BTAS), 2012 IEEE Fifth International Conference on*. IEEE, 8–15.

[41] Hyungsub Kim, Sangho Lee, and Jong Kim. 2014. Exploring and mitigating privacy threats of HTML5 geolocation API. In *Proceedings of the 30th Annual Computer Security Applications Conference*. ACM, 306–315.

[42] Pierre Laperdrix. 2017. *Browser Fingerprinting: Exploring Device Diversity to Augment Authentification and Build Client-Side Countermeasures*. Ph.D. Dissertation. Rennes, INSA.

[43] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. 2016. Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints. In *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 878–894.

[44] Claudio Marforio, Hubert Ritzdorf, Aurélien Francillon, and Srdjan Capkun. 2012. Analysis of the communication between colluding applications on modern smartphones. In *Proceedings of the 28th Annual Computer Security Applications Conference*. ACM, 51–60.

[45] Philip Marquardt, Arunabh Verma, Henry Carter, and Patrick Traynor. 2011. (sp) iPhone: decoding vibrations from nearby keyboards using mobile phone accelerometers. In *Proceedings of the 18th ACM conference on Computer and communications security*. ACM, 551–562.

[46] Maryam Mehrnezhad, Ehsan Toreini, Siamak F Shahandashti, and Feng Hao. 2018. Stealing PINs via mobile sensors: actual risk versus user perception. *International Journal of Information Security* 17, 3 (2018), 291–313.

[47] Yan Michalevsky, Dan Boneh, and Gabi Nakibly. 2014. Gyrophone: Recognizing Speech from Gyroscope Signals.. In *USENIX Security Symposium*. 1053–1067.

[48] Elinor Mills. [n. d.]. Device identification in online banking is privacy threat, expert says. https://www.cnet.com/news/device-identification-in-online-banking-is-privacy-threat-expert-says/.

[49] Tyler Moore and Benjamin Edelman. 2010. Measuring the perpetrators and funders of typosquatting. In *International Conference on Financial Cryptography and Data Security*. Springer, 175–191.

[50] Keaton Mowery and Hovav Shacham. 2012. Pixel Perfect: Fingerprinting Canvas in HTML5. In *Proceedings of W2SP 2012*.

[51] Patrick Mutchler, Adam Doupé, John Mitchell, Chris Kruegel, and Giovanni Vigna. 2015. A large-scale study of mobile web app security. In *Proceedings of the Mobile Security Technologies Workshop (MoST)*.

[52] Sashank Narain, Triet D Vo-Huu, Kenneth Block, and Guevara Noubir. 2016. Inferring user routes and locations using zero-permission mobile sensors. In *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 397–413.

[53] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2013. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *Security and privacy (SP), 2013 IEEE symposium on*. IEEE, 541–555.

[54] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2014. On the workings and current practices of web-based device fingerprinting. *IEEE Security & Privacy* 12, 3 (2014), 28–36.

[55] Łukasz Olejnik, Gunes Acar, Claude Castelluccia, and Claudia Diaz. 2015. The leaking battery. In *Data Privacy Management, and Security Assurance*. Springer, 254–263.

[56] Emmanuel Owusu, Jun Han, Sauvik Das, Adrian Perrig, and Joy Zhang. 2012. ACCessory: password inference using accelerometers on smartphones. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*. ACM, 9.

[57] Ashis Kumar Ratha, Shibani Sahu, and Priya Meher. 2018. HTML5 in Web Development: A New Approach. (2018).

[58] Sasank Reddy, Min Mun, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani Srivastava. 2010. Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks (TOSN)* 6, 2 (2010), 13.

[59] Yanzhi Ren, Yingying Chen, Mooi Choo Chuah, and Jie Yang. 2013. Smartphone based user verification leveraging gait recognition for mobile healthcare systems. In *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2013 10th Annual IEEE Communications Society Conference on*. IEEE, 149–157.

[60] rovo89. [n. d.]. Xposed framework. https://repo.xposed.info. Accessed: 2018-06-14.

[61] Quirin Scheitle, Oliver Hohlfeld, Julien Gamba, Jonas Jelten, Torsten Zimmermann, Stephen D. Strowes, and Narseo Vallina-Rodriguez. 2018. A Long Way to the Top: Significance, Structure, and Stability of Internet Top Lists. In *IMC*.

[62] Roman Schlegel, Kehuan Zhang, Xiao-yong Zhou, Mehool Intwala, Apu Kapadia, and XiaoFeng Wang. 2011. Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones.. In *NDSS*, Vol. 11. 17–33.

[63] Peter Snyder, Lara Ansari, Cynthia Taylor, and Chris Kanich. 2016. Browser feature usage on the modern web. In *Proceedings of the 2016 Internet Measurement Conference*. ACM, 97–110.

[64] Raphael Spreitzer. 2014. Pin skimming: Exploiting the ambient-light sensor in mobile devices. In *Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices*. ACM, 51–62.

[65] Greg Sterling. [n. d.]. Mobile Devices Now Driving 56 Percent Of Traffic To Top Sites.

[66] Sipat Triukose, Sebastien Ardon, Anirban Mahanti, and Aaditeshwar Seth. [n. d.]. Geolocating IP Addresses in Cellular Data Networks. In *Proceedings of the 13th International Conference on Passive and Active Measurement (PAM'12)*.

[67] Randika Upathilake, Yingkun Li, and Ashraf Matrawy. 2015. A classification of web browser fingerprinting techniques. In *New Technologies, Mobility and Security (NTMS), 2015 7th International Conference on*. IEEE, 1–5.

[68] Yong Wang, Daniel Burgener, Marcel Flores, Aleksandar Kuzmanovic, and Cheng Huang. 2011. Towards Street-Level Client-Independent IP Geolocation.. In *NSDI*, Vol. 11. 27–27.

[69] Yi-Min Wang, Doug Beck, Jeffrey Wang, Chad Verbowski, and Brad Daniels. 2006. Strider Typo-Patrol: Discovery and Analysis of Systematic Typo-Squatting. *SRUTI* 6 (2006), 31–36.

[70] Yuxi Wu, Panya Gupta, Miranda Wei, Yasemin Acar, Sascha Fahl, and Blase Ur. 2018. Your Secrets Are Safe: How Browsers' Explanations Impact Misconceptions About Private Browsing Mode. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 217–226.

[71] Yuanyi Wu, Dongyu Meng, and Hao Chen. 2017. Evaluating private modes in desktop and mobile browsers and their resistance to fingerprinting. In *Communications and Network Security (CNS), 2017 IEEE Conference on*. IEEE, 1–9.

[72] Zhi Xu, Kun Bai, and Sencun Zhu. 2012. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 113–124.

[73] Zhe Zhou, Wenrui Diao, Xiangyu Liu, and Kehuan Zhang. 2014. Acoustic fingerprinting revisited: Generate stable device id stealthily with inaudible sound. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 429–440.

[74] John Zulueta, Andrea Piscitello, Mladen Rasic, Rebecca Easter, Pallavi Babu, Scott A Langenecker, Melvin McInnis, Olusola Ajilore, Peter C Nelson, Kelly Ryan, et al. 2018. Predicting Mood Disturbance Severity with Mobile Phone Keystroke Metadata: A BiAffect Digital Phenotyping Study. *Journal of medical Internet research* 20, 7 (2018).

PYTHIA: a Framework for the Automated Analysis of Web Hosting Environments

Srdjan Matic
University College London
s.matic@cs.ucl.ac.uk

Gareth Tyson
Queen Mary University of London
gareth.tyson@qmul.ac.uk

Gianluca Stringhini
Boston University
gian@bu.edu

ABSTRACT

A common approach when setting up a website is to utilize third party Web hosting and content delivery networks. Without taking this trend into account, any measurement study inspecting the deployment and operation of websites can be heavily skewed. Unfortunately, the research community lacks generalizable tools that can be used to identify *how* and *where* a given website is hosted. Instead, a number of ad hoc techniques have emerged, e.g., using Autonomous System databases, domain prefixes for CNAME records. In this work we propose PYTHIA, a novel lightweight approach for identifying Web content hosted on third-party infrastructures, including both traditional Web hosts and content delivery networks. Our framework identifies the organization to which a given Web page belongs, and it detects which Web servers are self-hosted and which ones leverage third-party services to provide contents. To test our framework we run it on 40,000 URLs and evaluate its accuracy, both by comparing the results with similar services and with a manually validated groundtruth. Our tool achieves an accuracy of 90% and detects that under 11% of popular domains are self-hosted. We publicly release our tool to allow other researchers to reproduce our findings, and to apply it to their own studies.

CCS CONCEPTS

• **Networks** → *Public Internet*.

KEYWORDS

CDNs, infrastructure, topology, web

ACM Reference Format:
Srdjan Matic, Gareth Tyson, and Gianluca Stringhini. 2019. PYTHIA: a Framework for the Automated Analysis of Web Hosting Environments. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313664

1 INTRODUCTION

When deploying a website, companies have the choice to either host it on their own servers, or to offload the content to third-parties, e.g., Web hosts or Content Delivery Networks (CDNs). Choosing a third party can have multiple advantages, including cost savings, reliability, and the ability to sustain larger amounts of traffic (even during distributed denial of service attacks). We argue that understanding the Web hosting landscape is important for a number of reasons. These range from allowing us to assess how critical certain hosting infrastructures are and to estimate the impact that a network attack could have on the Web [8, 17, 24], to being able to determine who is responsible when incidents (e.g., malware hosting) occur [26]. Despite the importance of the problem, the research community lacks scalable methods to map the hosting landscape. Instead, multiple studies tend to take an ad hoc approach, relying on various assumptions, which differ between papers. Although there are third party services that offer this functionality [18, 30], they do not disclose their methodology, creating concerns for both reproducibility, as well as accuracy of their results.

To fill this gap, this paper *presents an open source tool to the community,*[1] *which can determine whether the webpage of an organization is self-hosted or it uses a third-party hosting provider.* Our tool, PYTHIA, leverages the HTML code of the webpage, domain information, and the network ownership obtained from RDAP records, to determine if the content is hosted on a third-party infrastructure. This is done by computing the ownership of both the webpage *and* the hosting provider, such that the two can be compared. PYTHIA is built with a modular design and it is capable of obtaining information of the landing webpage even in the presence of complex HTML structures which use redirects.

To evaluate the efficacy of PYTHIA, we run it on 40,000 URLs generated from the Alexa top-10k domains [2]. Our validation process shows that our framework outperforms similar applications available on the Web, and it achieves an accuracy of 90% in detecting when a webpage is hosted by a third party. Furthermore, our measurement reports that over 89% of the popular domains that we inspected, take advantage of third parties for their hosting needs.

PYTHIA is open source and allows the research community to reproduce our findings. We intend this to become a shared community effort, allowing third party researchers to avoid the complexity involved in devising and building their own independent methodologies for this commonly encountered task.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313664

[1] The source code of PYTHIA is available at https://bitbucket.org/srdjanmatic/pythia.git

2 BACKGROUND

Understanding and measuring the Web hosting ecosystem is a complex endeavor. To complete this task, we need both information about the ownership of domains and the ownership of the IP addresses where webpages are hosted. In this section we introduce the concepts on which our approach is based, and the type of data that we retrieve to determine whether webpages are self-hosted or not.

2.1 Third-Party Hosting

Content Delivery Networks (CDNs) and Hosting Services are two popular mechanisms for delivering content to end users on behalf of other organizations. By offloading the task of serving the content of a website to third parties, these solutions are designed to provide better availability, scalability, faster content loads, redundancy, and enhanced security. These technologies have become so widespread that according to recent statistics, more than 60% of the most visited websites use CDNs to serve content to their users [4]. In this work we study the deployment of any kind of solution that delivers Web content for third parties, and for this reason we use the term *hosting* to refer to the *network where Web servers offering a service are based.*

There are countless papers that have explored the hosting patterns of websites, each taking a slightly different approach. A common approach is to launch large-scale distributed measurements [1, 9, 25], which perform DNS queries around the world to retrieve and classify DNS responses. This, unfortunately, is extremely complex and costly; furthermore, it cannot alone confirm if the infrastructure is third-party operated without further inspection. Calder et al. [6] utilized the EDNS-0 Client Subnet extension to simulate distributed queries towards Google's CDN. Although it revealed a large number of servers, all were operated by Google rather than third-parties. These techniques also do not work well for Anycast CDNs [7], which do not necessarily return DNS responses containing redirects. Another strategy employed is to utilize domain prefix lists, which map CNAME responses to their respective CDNs [21]. These, however, are limited to CDNs that exclusively rely on CNAME redirects (e.g., this excludes Bing). Furthermore, the list requires constant maintenance to remain up-to-date. Lastly, some studies utilize IP address to Autonomous System (AS) mappings [14] or metadata encoded into DNS records [3]; these, however are vulnerable to misattributing ownership, e.g, when a CDN places a cache in a third-party network. Such techniques have also been complimented with manually curated AS annotations, which stipulate the type of AS [19]. Again, these suffer from both manual annotation errors and require substantial upkeep. We argue that these diverse ad hoc techniques are driven by the lack of a standardized tool within the community, which can provide metadata on website hosting patterns.

2.2 The RDAP Protocol

To acquire information about a domain's ownership we use the Registration Data Access Protocol (RDAP) [22]. This protocol was designed to replace the WHOIS [16] protocol as the authoritative source for registering information about IP addresses, ASes and domain names. While its predecessor retrieved free text content, RDAP leverages a RESTful interface to deliver the data in a machine-readable JSON format. This simplifies the parsing process and allows us to easily extract information (e.g., the type of entity to which a range of IP addresses has been assigned, or the description of an AS). In this paper, we use the RDAP protocol to retrieve information about the ownership of an IP address to which a domain resolved.

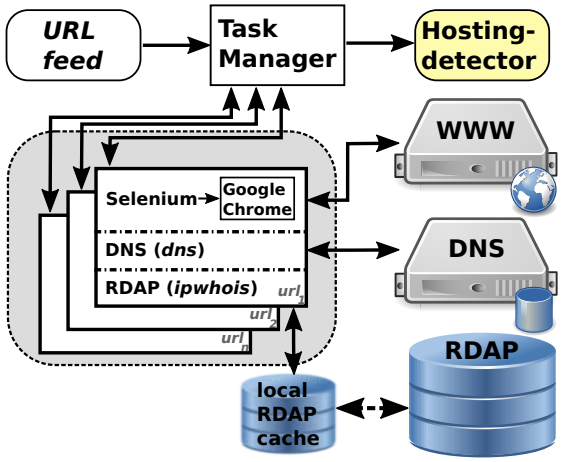


Figure 1: An overview of PYTHIA. The task manager processes URLs from a queue and instantiates a *crawler* task for each URL. The crawler collects information about a webpage and the environment from which the page was retrieved. The output of the crawler is stored in a dedicated data structure, which is later analyzed with the module that checks if content was served from a self-hosted network environment.

3 OVERVIEW OF PYTHIA

PYTHIA is entirely written in Python, and Figure 1 provides an overview of its system components. In a pipeline, PYTHIA performs the following two steps, (*i*) *Data Collection:* it takes a URL feed, and renders the list of websites, recording detailed information on all resources loaded, before complementing the data with RDAP records for the IP address hosting the website; and (*ii*) *Hosting Detector:* it then passes this information through a hosting detector, which decides if the owner of the webpage is the same as the owner of the hosting infrastructure where the server is located. The outcome is a structured JSON file that details if the website is self-hosted or operated by a third-party. In addition to this, PYTHIA also provides information on the ownership of both the webpage and the hosting service.

3.1 Data Collection

We first present the methods we use to collect the necessary data to infer ownership. This includes Web data, DNS information and, finally, RDAP records for all domains loaded.

Web Data Collection. Upon receiving a list of URLs, the crawler obtains Web content through Selenium,[2] a popular framework used for testing Web applications. We instrument Selenium to take a URL and to render it within a fully fledged instance of the Google Chrome browser. After the page has been loaded and rendered, our module outputs the retrieved HTML, in addition to a list of all the HTTP requests/responses (URLs) that were generated during the process. Each request/response is accompanied by metadata including the HTTP status code, and HTTP headers (e.g., server, content-type).

An important part of the crawling process is to determine when a webpage has finished loading. Since we do not know how long this process will take, we use an adaptive mechanism that leverages the information logged by the Web browser. We continuously monitor the browser logs and we consider the page loaded once all the requests to external resources have received their corresponding response. At the same time, we also set a hard timeout, after which we will close the browser session independently if the loading succeeded or not.

As part of the process, we also follow all the redirects that occur while loading an URL. This includes not just CNAME or HTTP redirects, but also those triggered by the refresh meta tag or a script. However, being primarily a tool for developers who need to check their own webpages, Selenium does not provide access to the browser internals. Hence, PYTHIA extracts information about the redirection chain from the browser log. After filtering out all the request to external resources, we identify the landing page and URLs on which the browser terminated the navigation.

Domain Resolution. Since Web browser logs do not contain information about domain resolution, PYTHIA launches DNS queries for all domains encountered (at a modest cost of only around 3.2% additional overhead per webpage). An advantage of this systems is that we can to use our own DNS server and we do not need to rely on the built-in mechanisms of proprietary resolvers, which might be hardcoded in the browser.

IP Ownership. Using the DNS results, we then determine the "owner" of the IP space where the content is hosted. PYTHIA uses the RDAP protocol to find the network prefix to which an IP address belongs to, and to identify the owner of that range. Our framework uses a local RDAP cache to overcome rate-limiting issues of RDAP servers and to avoid querying ranges for which we already have fresh information. After successfully completing the RDAP resolutions, all the data generated by the above steps is stored in a JSON data structure.

[2] https://www.seleniumhq.org/

3.2 Detecting Hosting Infrastructures

The next step is to use the above information to detect if a website is self-hosted, or whether it uses on a third-party infrastructure. Our tool identifies the organization/company behind a domain name and a webpage, and searches for evidence that the page owner is the same as the owner of the network prefix or the AS hosting the Web server. We do not differentiate among various types of hosting services (i.e., VPS, CDN, or generic web hosting) and we do not use any precompiled list of popular or known hosting services. Instead we extract our information from the URL and the HTML retrieved from the landing page, and we match this data with the RDAP response of the IP address that is hosting the web server. This process allows us to detect the third-party network infrastructures even in the presence of CDN caches located at ISPs: even if we did not identify correctly the provider, our algorithm will detect a mismatch in the ownership of the webpage and the IP range.

To identify the organization that owns a webpage we use both the information from the URL and the HTML code. In particular, from the URL component we extract the Effective Second Level Domain (ESLD), and from the HTML we use the content of the <title> tag. Before retrieving any ownership information for a RDAP response, we first filter unnecessary details from the data such as "comments" or the "symbolic name of the network", which can contain references to the owner of the webpage even when the IP range is assigned to a completely different organization. After this step, each string contained in the *HTML title* or the *RDAP fields* has its leading space delimiters removed, is cleaned from punctuation characters and stop words, converted into lower case, and finally is split into tokens on space delimiters. The DNS system does not allow domain names to contain space delimiters and it is common have domains, such as "bankofamerica.com", where the ESLD is a combination of multiple words. To overcome this issue, the ESLD string follows the same cleaning process of the title and the RDAP, with the only difference in the tokenization, which is performed following the technique described in [23].

This process results in a series of string tokens that represent ownership features of both the webpage and the domain/IP address hosting it. The next step is to compare these tokens to see if they correspond. Our algorithm does six checks: four with the strings contained in the title/ESLD/RDAP, and an additional two with the tokenized versions of those strings. First, the algorithm verifies if the HTML title or the ESLD appears as a sub-string in any of the RDAP fields. Subsequently it repeats the same procedure with each string in the RDAP fields by comparing it both with the HTML title and the ESLD. The output of this process is binary and if the algorithm finds a match, it concludes that the owner of the webpage is also the owner of the network. As a final step, the algorithm checks for the presence of common tokens among the lists of tokens obtained from the HTML title/ESLD and the RDAP information. In this case a single match is not enough to conclude that the same

organization owns both the webpage and the network, and we require that the common tokens represent at least 50% of the overall number of tokens in the shortest list.

4 VALIDATION AND EVALUATION

Pythia is intended to be both accurate and straightforward to use for the community. To validate its capabilities we next run it over multiple datasets.

4.1 Validating the Data Collection

Our first goal is to test the efficacy of our crawler in collecting the necessary data to perform the hosting classification. Hence, we run Pythia over a series of URL lists. A first dataset includes 10,000 unique domain names obtained from a snapshot of the "Alexa top 10,000 websites" (top-10k) on 1st of May 2018. A second dataset (top-20k-www) is an extended version of the previous one, where domains are extended with the "www." prefix. Finally, a third dataset (top-40k-URLs) includes all the entries from top-20k-www expanded with "http://" and "https://" prefixes.

Data Collection. We run Pythia over the top-40k-URLs to collect information about their home pages. We split our dataset into chunks of 350 elements and we process each chunk separately. Each element of the chunk is a unique domain name, which is crawled both with and without the "www." prefix and with the two protocols that Pythia supports (HTTP and HTTPS). This means that when we successfully crawl an entire chunk, we obtain information for 1400 unique URLs. We refer to the initial URL from which we begin our crawling, and that we load in the browser, as "starting URL"; similarly, the URL on which the crawl terminates is called "landing URL". Once a chunk is processed, Pythia waits for 120 seconds before switching to the next one. Each chunk is analyzed using 20 parallel instances of our Crawler module, which uses a maximum timeout of 60 seconds while waiting for a webpage to finish loading. Note that when collecting IP Ownership information from RDAP, we randomize waiting timeouts, with of a maximum of 90 seconds, before retrying a query that triggered an exception; after 3 consecutive exceptions, the module marks an IP as "no info available" before switching to the next one. The entire data collection process took place from a single machine, although we note it is possible to split the dataset and run parallel instances of Pythia on different machines.

Data Collection Performance. The overall process of downloading the HTML, resolving DNS names and collecting RDAP data, took 35 hours to complete for the top-40k-URLs. Figure 2 presents the Cumulative Distribution Function (CDF) of the increase of number successful crawls across time. To be able to use fresh entries from our local RDAP cache, we crawled the 4 starting URLs linked to each domain at the same time. Due to this choice, the CDFs of each "URL format" have very similar shapes. Hence *four* distributions in Figure 2 are almost stacked on the top of each other, and the red line connecting the values is the average across

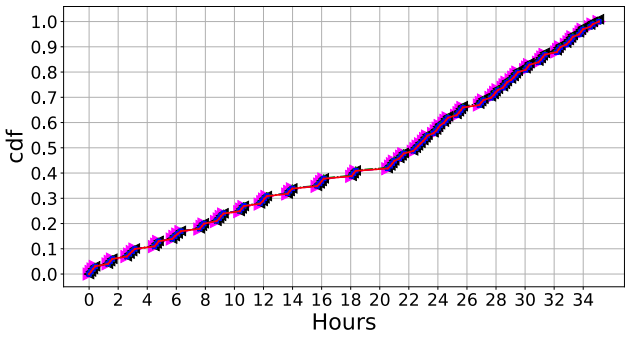


Figure 2: Cumulative distribution functions of the successful crawls with different URL formats.

those distributions. After 24 hours, we had crawled only 50% of the URLs, and in the last 1/3 of the time, we obtained the information for the remaining half of the dataset. The dataset contained shuffled entries, which did not follow any ranking depending on the "popularity of a domain" and the likelihood to have an "unreachable URL" at the beginning or at the end the crawl is the same. We explain the spike in the increase of the number of downloads after the 24th hour with the presence of our RDAP cache. As we will later show, the majority of URLs/domains use a third-party hosting provider and as times passes we observe an increase of the number of RDAP queries which can be resolved with our local cache. Those local resolutions increase our crawling speed allowing us to allow us to gather information for the same amount of elements, in half of the time.

In Table 1 we summarize the results of the data collection process using Pythia. The first thing to notice is that 85% of the starting URLs are successfully reached. Overall, our crawler visited 1,736,929 external URLs, which were retrieved from 54,410 different domains. This suggests an average factor of 42 URLs per "starting URL". It is therefore clear that each HTML page contains a considerable number of external resources, although it should be noted that this not only includes links to script and images, but also *redirects* from a starting to a landing webpage. Independently of the protocol and the presence of the "www." prefix, the 6th column in the Table shows that redirects are extremely popular. On average we pass via 2.4 intermediate URLs before reaching a landing page. For this estimate we *only* use redirects that happen when loading a "starting URL" in our browser, and we excluded any redirects triggered by the external resources embedded in the HTML of the landing webpage. In general, redirects seem to be more popular for the HTTP protocol, but the average difference with HTTPS is minimal.

Related to the redirect phenomenon, we also notice that more than 60% of the crawls observed a "change of the domain" name among the "starting URL" and the "landing URL". This happens with only 1/3 of that frequency value if we crawl domains without the "www." prefix. The reason is that often the redirection happens from one domain to the same domain expanded with the "www." prefix. A similar

	URLs (Domains)									IPs		
	All	Starting	Crawls				Landing					
Format	Tot.	Tot.	Completed	Domain Change	Protocol Change	AVG. Redirects	NON-triggering exceptions	self-host	3rd-host	Tot.	NON-triggering exceptions	Landing Domains
http	874,574 (46,679)	10,000 (10,000)	9,204	6,114	6,701	2.6	8,968 (8,897)	941 (935)	8,027 (7,962)	34,332	33,728	8,178
http + www	885,035 (43,359)	10,000 (10,000)	9,280	2,606	6,627	2.4	9,033 (8,962)	977 (971)	8,056 (7,991)	32,496	31,973	8,214
https	745,939 (40,696)	10,000 (10,000)	8,215	5,147	580	2.3	8,037 (7,989)	877 (873)	7,160 (7,116)	30,640	30,217	7,353
https + www	794,449 (39,787)	10,000 (10,000)	8,673	2,229	491	2.2	8,462 (8,404)	918 (913)	7,544 (7,491)	30,177	29,781	7,683
All	1,736,929 (54,410)	40,000 (20,000)	35,372	16,096	14,399	2.4	13,940 (11,253)	1,559 (1,220)	12,381 (10,033)	38,092	37,492	9,188

Table 1: Results of running PYTHIA on the top-40k-URLs dataset. The values in the brackets indicates the unique number of elements for each entry.

trend is observable for the "change of the protocol", when crawling URLs with HTTP (which get upgraded to HTTPS).

The overall number of unique IPs of the landing pages is slightly less than 10,000, and it reflects the fact that crawling the same domain with the four different formats, most of the time will lead to the same landing URL/domain. On average there are 1.22 domain per each landing IP (comparison of columns 7 and 12 in Table 1). This is explained by the presence of large hosting providers with many different customers. The same argument explains why we observe a similar relationship of 1.43 domains per each IP, when considering the dataset of all URLs. Finally, the results of PYTHIA indicate that 89% of the landing URLs are served from a third-party hosting infrastructure which does not belong to the owner of the webpage. In the following section we illustrate how we tested the accuracy of our classification, by using a manually validated groundtruth and by comparing with similar applications.

4.2 Classification Validation

We next validate the efficacy of our tool by compiling a groundtruth classification, and comparing it against PYTHIA.

Compiling a Comparative Dataset. To the best of our knowledge, no groundtruth dataset exists regarding Web hosting. To build this, we randomly select 324 domain names to manually annotate. These are taken from the TOP-10K dataset, crawled with the HTTP protocol. For each of these domains, we load the landing webpage in a browser and use search engines to check if the owner of the IP prefix is an organization offering Web hosting or CDN services to its customers.

We note that 324 domains are not enough to evaluate our tool. Thus, we also collect equivalent data from a variety of public tools that allow users to "*discover who is hosting a website*". This allows us to compare our results against their outputs. Example of those services include HOSTINGCOMPASS.COM which can detect who is hosting an ESLD, or HOSTINGDETECTOR.COM and WHAT'S MY CDN? which allow more fine grained queries including "www." as prefix to the ESLD [11, 12, 31]. We choose to use those three application because they are free Web-based services that do not require any registration. We query those services with the URLs from our TOP-10K and TOP-20K-WWW datasets,

	Self-hosting		3rd-party hosting		
Tool/Service	TP	FN	TP	FN	F1-score
PYTHIA	26	3	279	16	0.73
hostingcompass	27	2	102	193	0.21
hostingdetector	12	17	239	56	0.25
whatsmycdn	27	2	127	168	0.24
cdnfinder	28	1	65	230	0.2

Table 2: Performance comparison of PYTHIA and other applications on our manually validated groundtruth.

depending on the service. As the services mentioned above do not provide any detail about the methodology they use to detect hosting providers, in addition to those Web applications, we also use CDNFINDER, an open-source project which aims to detect the usage of CDNs within websites. The tool uses PHANTOMJS and a hard-coded list of hostnames to load a webpage and detect the presence of external resources which are hosted on a CDN [20, 28]. Analogously to the Web services, we downloaded the tool and run it on our TOP-40K-URLs dataset. In total, this results in 5 datasets to compare PYTHIA against.

Comparison with Manual Annotations. Table 2 contains the results of comparing PYTHIA against the above online services, using the manual annotations as the groundtruth. Our algorithm was specifically designed for detecting the presence of "self-hosting" environments. Consequently, a domain will be flagged as "hosted on third-parties" in any situation where the webpage owner differs from the owner of the network prefix (e.g., a private Web server run at at home, where the broadband ISP is the owner of the network prefix). Despite this limitation, in the binary classification problem where a website is either self-hosted or hosted on a third-party service, PYTHIA still outperforms all the other services that we tested, achieving an F1-score which is almost three times larger than the average for the other services. Indeed, on our manual groundtruth we observe an accuracy of over 95%, even when, instead of verifying self-hosting, we focus on the complementary problem of detecting the presence of third-party hosting providers. CDNFINDER performs well in detecting self hosting, but has a very high false positive rate when classifying domains as third-party hosting. Similarly, HOSTINGDETECTOR achieves the highest accuracy in detecting external hosting

SERVICE	DOMAIN	WWW. + DOMAIN	HTTP + DOMAIN	HTTPS + DOMAIN	HTTP + WWW. + DOMAIN	HTTPS + WWW. + DOMAIN
HOSTINGCOMPASS	3,481 (3,283)	-	-	-	-	-
HOSTINGDETECTOR	3,229 (2,879)	5,607 (4,973)	-	-	-	-
WHATSMYCDN	978 (963)	3,268 (3,202)	-	-	-	-
CDNFINDER	-	-	59 (55)	403 (395)	149 (144)	1,849 (1,700)

Table 3: Comparison of PYTHIA with similar services/applications when evaluated on all of our datasets. The values in the brackets indicates results obtained by PYTHIA.

services, at a cost of an extremely high false negative rate (59%), when a single organization is in control of both the webpage and the network prefix.

Comparison with Similar Services. To further test the accuracy of our framework, we compare our results with the four tools/services mentioned earlier. The results of this comparison are shown in Table 3. The goal of this is to show that PYTHIA achieves similar results to other applications. To this end, we narrow our goal to identify all domains which are hosted on third-party network infrastructures. As mentioned in the previous sections, PYTHIA follows any kind of redirect. Hence, Table 1 only presents the classification using the "landing URLs/domains". Since we do not know what are the exact capabilities of the four services that we tested, and how they handle redirects, we decided to back-propagate the results of our classification from the "landing URL" to the corresponding "starting URL", and "starting domain", from where the navigation started. In this way we are able to compare our results with each one of those services and verify that our framework has a detection rate close to those of the other services.

For almost all of the domains inspected, PYTHIA achieves an accuracy of around 90%, and it identifies a third-party-hoster every time one of the other four services detects its presence. Since the highest number of misclassified services originate from the sets of domains analyzed with HOSTINGDETECTOR, we sampled 20 domains without the "www." prefix and another 20 with the "www." prefix. We then manually verify if they are actually hosted on a third-party infrastructure. For 27 out of 40 cases, HOSTINGDETECTOR failed to identify self-hosted domains and PYTHIA correctly labeled those as "self-hosting". Ten of those cases were domains of large universities with their own network prefixes. For another 7 cases, the landing page is the home page of large hosting services such as "Google", "Salesforce" or "1and1". PYTHIA correctly labeled these as self-hosted. For four domains our PYTHIA did not succeed in downloading the RDAP information, and PYTHIA could not classify those domains. The remaining 9 domains were hosted on a third-party infrastructure, but we did not detect them. According to those results, we conclude that the accuracy of PYTHIA is inline with similar services we compared to.

5 RELATED WORK

A significant amount of research has been done in the field of Content Delivery Networks and cloud computing. Krishnamurthy et al. [15] were the first to analyze the rise of CDNs and the benefits that they provide to end-users. After them several studies investigated this trend [1, 7, 13, 25]. Similar work has tried to uncover cloud usage patterns and which Web services are running on a cloud-associated IP address [10, 29].

Our techniques relies on a mix of methodologies, particularly exploiting RDAP data. There have been a small set of past papers that rely on similar data. For example, Cai et al. proposed to combine WHOIS information with the ASN in order to generate a comprehensive AS-to-organization mapping [5]. Tajalizadehkhoob et al. were the first ones to explore the identification of hosting provides by combining passive DNS with WHOIS information [27]. Unfortunately their approach leverages a classification of 2,000 ASes to filter out organization such as ISPs, education and government. This list has limited size and is manually generated, and this raises concerns about its reliability across time. Contrary to previous studies, our work does not use any precomplied list of organization names and it focuses on identifying self-hosting environments. PYTHIA allows other researchers to reproduce our results and it does not require any manual analysis or a priori knowledge of network prefixes or the ASes in charge for routing the network traffic.

6 CONCLUSION

In this work we presented PYTHIA, a tool for collecting information about a webpage and the environment where the page is hosted. Our framework extracts information from the retrieved HTML, the DNS and the ownership information associated to a network prefix. PYTHIA then exploits this data to infer if the website is self-hosted, or is reliant on a third party operator, e.g., a Content Delivery Network. We tested PYTHIA on 40,000 URLs and compared the results with similar applications that detect the presence of known hosting providers. Our framework is accurate and outperforms all other applications, when tested on a manually validated groundtruth. PYTHIA is released as open source and is built in a modular way, which gives the possibility to integrate it with new capabilities and extensions.

7 ACKNOWLEDGEMENTS

This research was supported by the PETRAS IoT hub (EPSRC grant EP/N023242/1) All opinions, findings and conclusions, or recommendations expressed in this material are those of the and do not necessarily reflect the views of the sponsors.

REFERENCES

[1] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Web content cartography. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 585–600. ACM, 2011.

[2] Alexa. Top 1M Sites, 2018. https://www.alexa.com/topsites.

[3] T. Böttger, F. Cuadrado, G. Tyson, I. Castro, and S. Uhlig. Open connect everywhere: A glimpse at the internet ecosystem through the lens of the netflix cdn. *SIGCOMM Comput. Commun. Rev.*, 48(1), 2018.

[4] BuiltWith. DN technologies Web Usage Statistics. https://trends.builtwith.com/cdn, 2018. [Online; accessed 14-May-2018].

[5] X. Cai, J. Heidemann, B. Krishnamurthy, and W. Willinger. Towards an AS-to-Organization Map. In *ACM Internet Measurement Conference*, 2010.

[6] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. Heidemann, and R. Govindan. Mapping the expansion of google's serving infrastructure. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 313–326. ACM, 2013.

[7] M. Calder, A. Flavel, E. Katz-Bassett, R. Mahajan, and J. Padhye. Analyzing the performance of an anycast cdn. In *Proceedings of the 2015 Internet Measurement Conference*, pages 531–537. ACM, 2015.

[8] A. Delignat-Lavaud and K. Bhargavan. Network-based origin confusion attacks against https virtual hosting. In *Proceedings of the 24th International Conference on World Wide Web*, pages 227–237, 2015.

[9] R. Fanou, G. Tyson, P. Francois, and A. Sathiaseelan. Pushing the frontier: Exploring the african web ecosystem. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016.

[10] K. He, A. Fisher, L. Wang, A. Gember, A. Akella, and T. Ristenpart. Next stop, the cloud: Understanding modern web service deployment in ec2 and azure. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 177–190. ACM, 2013.

[11] HostingCompass. We analyze ip address and web hosting! https://www.hostingcompass.com.

[12] HostingDetector.com. Web Hosting Lookup - Find out who is hosting any website. https://hostingdetector.com/.

[13] C. Huang, A. Wang, J. Li, and K. W. Ross. Measuring and evaluating large-scale cdns. In *ACM IMC*, volume 8, pages 15–29, 2008.

[14] D. Ibosiola, B. Steer, A. Garcia-Recuero, G. Stringhini, S. Uhlig, and G. Tyson. Movie pirates of the caribbean: Exploring illegal streaming cyberlockers. *AAAI International Conference on Web Blogs and Social Media (ICWSM)*, 2018.

[15] B. Krishnamurthy, C. Wills, and Y. Zhang. On the use and performance of content distribution networks. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 169–182. ACM, 2001.

[16] D. L. RFC 3912, whois protocol specification. Technical report, IETF — Internet Engineering Task Force, 2004.

[17] J. Liang, J. Jiang, H. Duan, K. Li, T. Wan, and J. Wu. When HTTPS Meets CDN: A Case of Authentication in Delegated Service. In *IEEE Symposium on Security and Privacy*, 2014.

[18] NetCraft. Hosting Provider Server Count. https://www.netcraft.com/internet-data-mining/hosting-provider-server-count/, 2018. [Online; accessed 24-May-2018].

[19] A. Noroozian, M. Korczyński, C. H. Gañan, D. Makita, K. Yoshioka, and M. van Eeten. Who gets the boot? analyzing victimization by ddos-as-a-service. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 368–389. Springer, 2016.

[20] phantomjs.org. PhantomJS - Scriptable Headless Browser. http://phantomjs.org/.

[21] S. Quirin, J. Jonas, H. Oliver, C. Luca, and C. Gorg. Structure and Stability of Internet Top Lists. In *Proceedings of the 19th International Conference on Passive and Active Measurement*. Springer, 2018.

[22] H. Scott and K. N. RFC 7482,registration data access protocol (rdap) query format. Technical report, IETF — Internet Engineering Task Force, 2015.

[23] T. Segaran and J. Hammerbacher. *Beautiful data: the stories behind elegant data solutions*. "O'Reilly Media, Inc.", 2009.

[24] M. Simenovski, G. Pellegrino, C. Rossow, and M. Backes. Who controls the internet? analyzing global threats using property graph traversals. In *Proceedings of the 2017 Web Conference*, 2017.

[25] A.-J. Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante. Drafting behind akamai: Inferring network conditions based on cdn redirections. *IEEE/ACM Transactions on Networking (TON)*, 17(6):1752–1765, 2009.

[26] S. Tajalizadehkhoob, C. Gañán, A. Noroozian, and M. v. Eeten. The role of hosting providers in fighting command and control infrastructure of financial malware. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 575–586. ACM, 2017.

[27] S. Tajalizadehkhoob, M. Korczyński, A. Noroozian, C. Ganán, and M. van Eeten. Apples, oranges and hosting providers: Heterogeneity and security in the hosting market. In *Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP*, pages 289–297. IEEE, 2016.

[28] TurboBytes. GitHub - turbobytes/cdnfinder: Webapp and cli-tool to detect CDN usage of websites. https://github.com/turbobytes/cdnfinder.

[29] L. Wang, A. Nappa, J. Caballero, T. Ristenpart, and A. Akella. WhoWas: A Platform for Measuring Web Deployments on IaaS Clouds. In *ACM Internet Measurement Conference*, 2014.

[30] WebHosting. https://webhosting.info/. https://webhosting.info, 2018. [Online; accessed 24-May-2018].

[31] whatsmycdn.com. What's My CDN? http://www.whatsmycdn.com/.

Event Detection using Hierarchical Multi-Aspect Attention

Sneha Mehta
Virginia Tech, USA
snehamehta@cs.vt.edu

Mohammad Raihanul Islam
Virginia Tech, USA
raihan8@cs.vt.edu

Huzefa Rangwala
George Mason University, USA
rangwala@cs.gmu.edu

Naren Ramakrishnan
Virginia Tech, USA
naren@cs.vt.edu

ABSTRACT

Classical event encoding and extraction methods rely on fixed dictionaries of keywords and templates or require ground truth labels for phrase/sentences. This hinders widespread application of information encoding approaches to large-scale free form (unstructured) text available on the web. Event encoding can be viewed as a hierarchical task where the coarser level task is event detection, i.e., identification of documents containing a specific event, and where the fine-grained task is one of event encoding, i.e., identifying key phrases, key sentences. Hierarchical models with attention seem like a natural choice for this problem, given their ability to differentially attend to more or less important features when constructing document representations. In this work we present a novel factorized bilinear multi-aspect attention mechanism (FBMA) that attends to different aspects of text while constructing its representation. We find that our approach outperforms state-of-the-art baselines for detecting civil unrest, military action, and non-state actor events from corpora in two different languages.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Natural language processing**; **Supervised learning**; **Neural networks**.

KEYWORDS

Neural Networks, Hierarchical Attention, Multi-Aspect Attention, Event Encoding

ACM Reference Format:
Sneha Mehta, Mohammad Raihanul Islam, Huzefa Rangwala, and Naren Ramakrishnan. 2019. Event Detection using Hierarchical Multi-Aspect Attention. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313659

1 INTRODUCTION

Given the large volumes of text available on the web in the form of news, social media, blogs and discussion forums, it is crucial to identify and extract meaningful nuggets of information. A wide range of applications from question answering [24], knowledge base construction [21] and named entity recognition [19] to informing critical decisions in domains ranging from national security to cyber security [22] rely on the event extraction and encoding process. Event analysis refers to the extraction of specific information about certain events from text. It can be categorized as a hierarchical task where the coarser level task is event detection, i.e., identification of documents containing a specific event and the fine-grained task is one of event encoding, i.e., identifying key phrases and sentences describing event related information such as the type of the event, type of people involved in the event, and relationships between these aspects. The sheer diversity of applicable event domains and types combined with the multitude of data sources, and scarcity of fine-level labels make both these tasks challenging.

Prior work in event encoding has focused on extracting entities, detecting trigger terms, and matching slots on predefined templates [3, 16, 26]. However there are a few shortcomings of these approaches. The first drawback is that they rely on fine-grained labeled training data which is hard to obtain for a variety of domains and different types of events. On the other hand, labels at the document level are easier to obtain. Second, they use of sentence-level embedding [16] removes contextual information resulting in false negatives because event occurrences do not neatly partition into unique sentences. Multiple instance learning (MIL) approaches have been proposed [9, 23] as a solution to partly alleviate these problems. MIL approaches view documents as bags of sentences, make predictions at the sentence level, aggregate sentence level probabilities for a select few sentences to obtain document-level predictions. In this work we model the tasks of event detection and key sentence identification from a news article in a unified framework without explicit labels at the sentence level by leveraging the implicit hierarchy in textual corpora. Hierarchical attention based networks seem like a natural choice for this problem, given their ability to differentially attend to more or less important words and sentences when constructing document representations [25].

We use a recurrent neural network (RNN) based hierarchical model (with attention mechanisms at both levels in the hierarchy) that constructs sentence and document representations. The sentence representation is constructed by attending to all words in the sentence, whereas the document representation is constructed by attending to all sentences. To capture the fact that a sentence might capture multiple aspects related to an event (e.g., cause, location, population) we adapt hierarchical attention network models [25] by using a novel multi-aspect attention mechanism that allows for multiple attention distributions over words for a single sentence where each distribution can be thought of as attending to a different aspect of a sentence.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313659

We evaluate the proposed hierarchical attention models for the task of event detection on civil unrest (CU) and military action/non-state actor (MANSA) datasets.
Our contributions in this work can be succinctly summarized as follows:

- We present a novel factorized bilinear multi-aspect attention mechanism (FBMA) that constructs a sentence representation using multiple attention distributions and that when used with hierarchical models improves performance for event detection.
- Our FBMA approach achieves state-of-the-art results for event detection on three event datasets from two different domains in two different languages.

2 RELATED WORK

2.1 Event Extraction

In event extraction supervised approaches usually rely on manually labeled training datasets and handcrafted ontologies. Li et al. [10] utilize annotated arguments and specific keyword triggers in text to develop an extractor. Supervised approaches have also been studied using dependency parsing by analyzing the event-argument relations and discourse of event interactions [14]. These approaches are usually limited by the availability of fine-grained labeled data and require elaborately designed features. In contrast to these approaches our method uses attention mechanisms to implicitly weigh words and sentences and is able to extract event extents and trigger words with labels provided only at the document level. This formulation is suitable because labels at document level are easier to obtain than at the per-sentence level or at the word level. This makes the task of event extraction also amenable to multiple instance learning (MIL) [5] solutions. In MIL 'bags' are groups of 'instances' which are to be classified. In a standard MIL formulation individual instance level labels are not available and labels are provided only at the group/bag level. Each bag is labeled positive if it contained at least one positive instance and negative otherwise. Kotzias et al. [9] focus on instance-level predictions from group level labels and allow for the application of general aggregation functions for sentiment classification. Wang et al. [23] use a similar idea and hold the previous state-of-the-art results on one of the datasets we evaluate on. Contrary to these approaches our method is hierarchical and computes the feature representation for the next level in the hierarchy using a weighted average of feature representations in the current layer.

2.2 Attention for Structured Representations

Models have been proposed that compute multiple attention distributions over a single sequence of words. The multi-view networks proposed by Guo et al. [6] use a different set of parameters for each view which leads to a large increase in the number of parameters with increasing number of views. Lin et al. [12] alleviate this problem by producing a matrix embedding from a single set of parameters. Both these methods use a special case of additive attention proposed by Bahdanau et al. [1] in the context of neural machine translation. Luong et al. [13] simplify additive attention operations by introducing the notion of multiplicative attention which is faster to compute. In multiplicative attention, the score between two feature vectors is learned using a bilinear projection matrix. Dot product attention [13] is a special case of multiplicative attention where the score between two features vectors is computed by a simple dot product between them. Yang et al. [25] use dot product attention to compute the similarity of word hidden representation to a word-level context vector which is learned with the rest of the model. In our work we compute the score between the context vector and the word hidden representation using a bilinear projection matrix and then we use an approach inspired by multi-modal low rank bilinear pooling proposed by Kim et al. [8] to factorize the matrix into two low rank matrices to compute multiple attention distributions over words. Contrary to Guo et al. [6] we use matrix factorization to alleviate the problem of increasing parameters with increasing views and our approach uses fewer parameters than Lin et al. to compute multiple attention calculations and performs superior to their approach. We refer to this as multi-aspect attention as it attends to different aspects or parts of a sentence for constructing a sentence embedding.

3 PROPOSED MODEL

The event detection problem can be defined as follows: Given a corpus containing N news articles $\{x_1, x_2, ..., x_N\}$, each article is associated with an event label $y \in \{0, 1\}$, with 1 corresponding to articles containing an event. For each news article we aim to predict its label, indicating if it contains an event or not.

3.1 Sequence Encoder

Consider a news article containing n sentences with each sentence containing T words. A sentence consists of word tokens w_{it}, $t \in [0, T]$ where every word is converted to a real valued word vector $\mathbf{x}_{it}$ using the pre-trained embedding matrix $\mathbf{W}_e = R^{dx|V|}$, $\mathbf{x}_{it} = \mathbf{W}_e\mathbf{w}_{it}$, $t \in [1, T]$ where d is the embedding dimension and V is the vocabulary. We encode the sentence using a bi-directional Gated Recurrent Unit [4](bi-GRU) RNN that summarizes information in both directions along the sentence to get a contextual annotation of a word. In a bi-GRU the hidden state at time step t is represented as a concatenation of hidden states in the forward and backward direction. The forward GRU denoted by $\overrightarrow{GRU}$ processes the sentence from w_{i1} to w_{iT} whereas the backward GRU denoted by $\overleftarrow{GRU}$ processes it from w_{iT} to w_{i1}.

$$\mathbf{x}_{it} = \mathbf{W}_e\mathbf{w}_{it} \tag{1}$$

$$\overrightarrow{\mathbf{h}_{it}} = \overrightarrow{GRU}(\mathbf{x}_{it}, \mathbf{h}_{i(t-1)}, \boldsymbol{\theta}) \tag{2a}$$

$$\overleftarrow{\mathbf{h}_{it}} = \overleftarrow{GRU}(\mathbf{x}_{it}, \mathbf{h}_{i(t+1)}, \boldsymbol{\theta}) \tag{2b}$$

Here the word annotation $\mathbf{h}_{it}$ is obtained by concatenating the forward hidden state $\overrightarrow{\mathbf{h}_{it}}$ and the backward hidden state $\overleftarrow{\mathbf{h}_{it}}$.

3.2 Word-Level Attention

For event extraction the presence of certain words increases the probability of a sentence containing the event. Such words should be given higher weight than other words while computing a sentence representation. Since an event related trigger word can occur

anywhere in a sentence we choose the global attention mechanism [13] in which the sentence representation is computed by attending to all words in the sentence.

3.2.1 Bilinear Attention. Let $\mathbf{h}_{it}$ be the annotation corresponding to the word $\mathbf{x}_{it}$. First we transform $\mathbf{h}_{it}$ using a one layer Multi-Layer Perceptron (MLP) to obtain its hidden representation $\mathbf{u}_{it}$.

$$\mathbf{u}_{it} = tanh(\mathbf{W}_w \mathbf{h}_{it} + \mathbf{b}_w) \tag{3}$$

We measure the importance of word by computing an alignment score of $\mathbf{u}_{it}$ to a word level context vector $\mathbf{u}_w$ using a bilinear model:

$$f_{it} = \mathbf{u}_w^T \mathbf{W}_i \mathbf{u}_{it} \tag{4}$$

Here, $\mathbf{W}_i$ is a bilinear projection matrix, $\mathbf{u}_w$ is randomly initialized and jointly learned with other parameters during training. Similar to [25], $\mathbf{u}_w$ can be seen as a high dimensional representation of the fixed query 'What is the informative word'. $\mathbf{u}_w \in \mathbb{R}^l$, $\mathbf{u}_{it} \in \mathbb{R}^{2h}$ and $\mathbf{W}_i \in \mathbb{R}^{2hx2h}$. The attention weight for the word $\mathbf{x}_{it}$ can be computed through a softmax function.

$$\alpha_{it} = \frac{exp(f_{it})}{\sum_{t'} exp(f_{it'})} \tag{5}$$

3.2.2 Factorized Bilinear Multi-Aspect Attention. The attention distribution above usually focuses on a specific component of the sentence, like a special set of trigger words or phrases. So it is expected to reflect an aspect, or component of the semantics in a sentence. However there can be multiple aspects that describe an event like who were involved in the event, what were the causes of the event or where did the event occur. For this we introduce the novel factorized bilinear multi-aspect attention (FBMA) mechanism. Suppose m aspects are to be extracted from a sentence, we need m alignment scores between each word hidden representation $\mathbf{u}_{it}$ and the context vector $\mathbf{u}_w$. To obtain an m dimensional output $\mathbf{f}_{it}$, we need to learn $\mathbf{W} = [\mathbf{W}_1, ..., \mathbf{W}_m] \in \mathbf{R}^{lx2hxm}$ as demonstrated in previous works. Although bilinear model might be effective in capturing pairwise interaction it introduces a huge number of parameters that may lead to a high computational cost. Inspired by multi-modal low rank bilinear pooling approach proposed by Kim et. al [8] and the matrix factorization approaches proposed in [11, 18] the bilinear projection matrix $\mathbf{W}_i$ can be factorized into two rank 1 matrices $\mathbf{P}$ & $\mathbf{Q}$. Eq.4 can be written as:

$$\mathbf{f}_{it} = \mathbf{u}_w^T \mathbf{W}_i \mathbf{u}_{it} = \mathbf{P}^T \mathbf{u}_w \circ \mathbf{Q}^T \mathbf{u}_{it} = \tilde{\mathbf{u}}_w \circ \tilde{\mathbf{u}}_{it} \tag{6}$$

Here $\mathbf{P} \in \mathbb{R}^{lxm}$ and $\mathbf{Q} \in \mathbb{R}^{2hxm}$ are two rank 1 matrices, m is the number of aspects to extract and $\circ$ is the Hadamard product or elementwise multiplication. This brings the two feature vectors $\mathbf{u}_{it} \in \mathbf{R}^{2h}$, the word hidden representation and $\mathbf{u}_w \in \mathbf{R}^l$, the word level context vector in the common space and are given by $\tilde{\mathbf{u}}_{it}$ and $\tilde{\mathbf{u}}_w$ respectively. $\mathbf{f}_{it} \in \mathbb{R}^m$ now is a multi-aspect alignment vector for the word $\mathbf{x}_{it}$. The multi-aspect attention vector $\boldsymbol{\alpha}_{it} \in \mathbb{R}^m$ is obtained by computing a softmax function along the sentence length:

$$\boldsymbol{\alpha}_{it} = \frac{exp(\mathbf{f}_{it})}{\sum_{t'} exp(\mathbf{f}_{it'})} \tag{7}$$

Before computing softmax, similar to [8] we apply an additional tanh nonlinearlity to $\mathbf{f}_{it}$. Since elementwise multiplication is introduced the values of neurons may vary a lot so we apply an l_2 normalization layer across the m dimension, ($\mathbf{f}_{it} \leftarrow \frac{\mathbf{f}_{it}}{||\mathbf{f}_{it}||}$) after the Hadamard product.

3.2.3 Sentence Representation. Let $\mathbf{H}_i = (\mathbf{h}_{i1}, \mathbf{h}_{i2}, ...\mathbf{h}_{iT})$ be a matrix of all word annotations in a sentence; $\mathbf{H}_i \in \mathbb{R}^{Tx2h}$. Let $\mathbf{A}_i = (\boldsymbol{\alpha}_{i1}, \boldsymbol{\alpha}_{i2}, ...\boldsymbol{\alpha}_{iT})$ be the multi-aspect attention matrix for the sentence; $\mathbf{A}_i \in \mathbb{R}^{mxT}$. The sentence representation for an aspect j given by $\boldsymbol{\alpha}_{ij} = \{\alpha_{j1}, \alpha_{j2}, ..\alpha_{jT}\}$ can be computed by taking a weighted sum of all word annotations.

$$\mathbf{s}_{ij} = \sum_{k=1}^{T} \mathbf{h}_{ik} * \alpha_{jk} \tag{8}$$

Similarly, sentence representation can be computed for all aspects and is given in a compact form by:

$$\mathbf{S}_i = \mathbf{A}_i \mathbf{H}_i \tag{9}$$

Here $\mathbf{S}_i \in \mathbb{R}^{mx2h}$ is a matrix sentence embedding and contains as many rows as the number of aspects. Each row contains an attention distribution for a new aspect. It can be flattened by concatenating all rows for further processing.

3.3 Sentence-Level Attention

News articles consist of many sentences with a few of them describing the event and the rest describing the supporting facts. The ones containing event related information should be assigned higher weights. Instead of hard selecting top K sentences and aggregating their probabilities as in prior MIL approaches such as [23] we use the global attention mechanism over the sentence annotations to get the document representation. Specifically, given a document containing sentence embeddings $\{\mathbf{s}_1, ..., \mathbf{s}_i, ..., \mathbf{s}_n\}$ where each $\mathbf{s}_i$ is a flattened representation of the matrix sentence representation $\mathbf{S}_i$ as given by eq. 9 we get the document vector as follows.

$$\overrightarrow{\mathbf{h}_i} = \overrightarrow{GRU}(\mathbf{s}_i, \mathbf{h}_{i-1}, \boldsymbol{\theta}) \tag{10a}$$

$$\overleftarrow{\mathbf{h}_i} = \overleftarrow{GRU}(\mathbf{s}_i, \mathbf{h}_{i+1}, \boldsymbol{\theta}) \tag{10b}$$

$$\mathbf{h}_i = \{\overrightarrow{\mathbf{h}_i}, \overleftarrow{\mathbf{h}_i}\} \tag{10c}$$

The sentence annotation $\mathbf{h}_i$ is obtained by concatenating the forward and backward hidden representations of the bi-GRU. Document representation is obtained by attention over sentences.

$$\mathbf{u}_i = tanh(\mathbf{W}_s \mathbf{h}_i + \mathbf{b}_s), \tag{11a}$$

$$\alpha_i = \frac{exp(\mathbf{u}_i^T \mathbf{u}_s)}{\sum_{t'}' exp(\mathbf{u}_{t'}^T \mathbf{u}_s)} \tag{11b}$$

$$\mathbf{d} = \sum_i \alpha_i \mathbf{h}_i \tag{11c}$$

Here $\mathbf{u}_s$, the sentence level context vector, is randomly initialized and learned along with other model parameters while training and $\mathbf{d}$ is the document representation that summarizes all the information

Table 1: First row indicates population classes participating in a protest in the CU dataset. Second row indicates the causes of protest.

Event Population	General Population, Business,Legal, Labor, Agricultural, Education, Medical, Media
Event Type	Government Policies, Employment and Wages, Energy and Resources, Economic Policies, Housing

in the article. Given the document representation, we use a two hidden layer MLP with dropout to get the class scores.

$$\hat{y} = \mathbf{W}_c\mathbf{d} + \mathbf{b}_c. \tag{12}$$

Loss for the document is computed using the standard cross entropy.

$$l = -(y \log(\hat{y}) + (1-y)\log(1-\hat{y})) \tag{13}$$

The sentence embedding given by eq. 9 can suffer from redundancy issues if the attention mechanism always provides similar weights for all the m aspects. To attend to a small set of trigger words in each aspect and to encourage diversity in different aspects we use penalization as described by Lin et al. [12] that is added to the loss:

$$P = \frac{\sum_i \left\|\mathbf{A}_i\mathbf{A}_i^T - \mathbf{I}\right\|_F^2}{n} \tag{14}$$

Where $\|\bullet\|_F$ stands for the Frobenius norm of a matrix and the summation is taken over all sentences in the document. The final training loss is given by:

$$L = \sum_d l + \lambda P \tag{15}$$

The summation is taken over all the documents in the batch and λ is a hyperparameter. We use the mini-batch stochastic gradient descent algorithm [7] with momentum and weight decay for optimizing the loss function and the backpropogation algorithm is used to compute the gradients.

3.3.1 Hyperparameters. We use a word embedding size of 100. The embedding matrix $\mathbf{W}_e$ is pretrained on the corpus using the gensim [1] implementation of the widely used distributed representations model word2vec [15]. All words appearing less than 5 times are discarded. The GRU hidden state is set to $h = 50$. In FBMA the dimension of $\mathbf{u}_{it}$ is given by the dimension of the GRU hidden state, but the dimension of $\mathbf{u}_w$ can be tuned. Empirically we find that setting the dimension of $\mathbf{u}_w$ to 32 gives us the best results. We set the classifier MLP hidden state to 512 and apply a 0.4 dropout to the hidden layer. We use a batch size of 64 for training and an initial learning rate of 0.05. For early stopping we use $patience = 5$.

4 EXPERIMENTS

4.1 Datasets

To evaluate our approach we use three event datasets - (i) the Civil Unrest Gold Standard Report labeled manually by analysts from MITRE corporation [17]. It contains encodings of civil unrest events from 10 Latin American countries in Argentina, Brazil, Chile, Colombia, Ecuador, El Salvador, Mexico, Paraguay, Uruguay, and Venezuela. The encodings are obtained from major national newspapers as identified by 4imn.com. It contains a total of 24,110 news articles out of which 18% mention a protest event while the rest are non-protest articles. All the articles are in Spanish and we refer to this dataset as CU (Spanish) in this paper. (ii) The AutoGSR dataset – This dataset comes from the EMBERS AutoGSR system [20] which is a web based system that generates validated civil unrest events extracted from news articles. It contains a total of 32,019 news articles out of which 18% describe protest events (protest articles) and the rest do not describe any protest events (non-protest articles). We refer to this dataset as CU (English) in the paper. For each protest article, the CU English & Spanish datasets contain the population type and protest event type. The population type indicates which class of population are involved in the protest and the event type indicates the main reason behind the protest. These are listed in table 1.

Finally, we evaluate on iii) the Military Action and Non-State Actor (MANSA) GSR dataset which is in English and Arabic. This contains event encodings from gulf countries namely, Bahrain, Egpyt, Iraq, Jordan, Lebanon, Qatar, Saudi Arabia and Syria. The event types include 'Military Actions' (MA) which are actions by military, police, or security organization and 'Non-State Actor'(NSA) which are actions initiated by non-governmental groups or individuals to further political, social, religious or ideological objectives. These events are encoded from news articles collected from the web and print media. Event collection techniques include Google Advanced Search (limited to the newspaper website), Nexis queries, and IHS Janes. Google Advanced Search is used to collect events in online media. Nexis and IHS Janes are used to collect events in print media. About 34% articles describe an event and rest are non-event articles. We refer to this dataset as MANSA dataset in the paper. MA & NSA events are further divided into subtypes. In this work we combine NSA & MA events together for detection. Please refer to table 2 for the overview of our datasets.

Table 2: Dataset statistics. Total number of news articles, average number of sentences per article and average number of words per article in the datasets.

Datset	# articles	# sents	# words
MANSA	105858	6.3	250.3
CU Spanish	24110	11.4	337.1
CU English	32019	10.1	358.2

Table 3: Various baselines and their key characteristics. The proposed FBMA model is hierarchical, with bilinear multi-aspect attention.

Feature / Method	hierarchical	key words	key sents	bilinear attention	multi-aspect attention
MICNN	✓	×	✓	×	×
MIGCNN	✓	✓	✓	×	×
HAN	✓	✓	✓	×	×
BSA	✓	✓	✓	✓	×
HSA	✓	✓	✓	×	✓
FBMA	✓	✓	✓	✓	✓

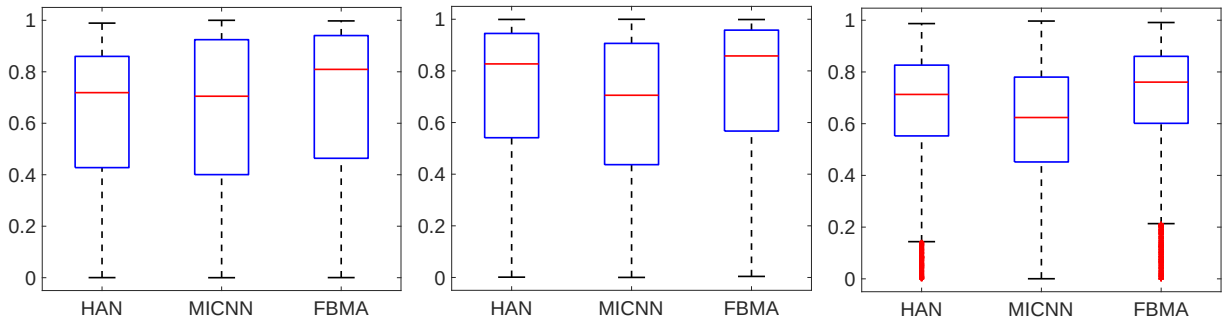
4.2 Comparative Methods

Table 3 shows the different approaches that were evaluated in this study along with their key characteristics. CNNs within a multiple instance learning (MIL) framework have been used by Wang et al. [23]. We consider the MI-CNN model proposed by them and its

[1] https://radimrehurek.com/gensim/

Table 4: Results of Event Detection. FBMA refers to the proposed Factorized Bilinear Multi-aspect Attention mechanism. BSA refers to the Bilinear Single-aspect Attention mechanism presented in eq. 4, MI-GCNN refers to our MIL model, where the CNN encoder is replaced by the RNN encoder followed by simple dot product attention to extract key words. HAN, HSA & MI-CNN are other baselines.

Dataset	Method	**Precision** (std.)	**Recall** (std.)	**F1** (std.)
MANSA	MI-CNN **(Wang et al.)**	0.731 (0.012)	0.785 (0.004)	0.686 (0.021)
	MI-GCNN (This paper)	**0.793 (0.003)**	0.622 (0.013)	0.697 (0.007)
	HSA **(Lin et al.)**	0.733 (0.001)	0.823 (0.009)	0.775 (0.003)
	HAN **(Yang et al.)**	0.740 (0.007)	0.831(0.007)	0.783 (0.004)
	BSA (This paper)	0.737 (0.007)	**0.834 (0.003)**	0.782 (0.003)
	FBMA (This paper)	0.747(0.003)	0.831 (0.003)	**0.787(0.002)**
CU (Spanish)	MI-CNN **(Wang et al.)**	0.742 (0.036)	**0.813 (0.041)**	0.775 (0.006)
	MI-GCNN (This paper)	**0.834 (0.011)**	0.721 (0.009)	0.773 (0.005)
	HSA **(Lin et al.)**	0.763 (0.009)	0.745 (0.016)	0.754 (0.011)
	HAN **(Yang et al.)**	0.811 (0.011)	0.775 (0.011)	0.793 (0.005)
	BSA (This paper)	0.812 (0.015)	0.779 (0.011)	0.795 (0.007)
	FBMA (This paper)	0.816 (0.017)	0.784 (0.010)	**0.800 (0.008)**
CU (English)	MI-CNN **(Wang et al.)**	**0.824 (0.01)**	0.644(0.009)	0.723 (0.009)
	MI-GCNN (This paper)	0.815 (0.006)	0.68 (0.017)	0.742 (0.011)
	HSA **(Lin et al.)**	0.746 (0.008)	0.710(0.017)	0.727 (0.010)
	HAN **(Yang et al.)**	0.779 (0.012)	0.746 (0.024)	0.762 (0.015)
	BSA (This paper)	0.786 (0.008)	**0.757 (0.007)**	**0.771 (0.006)**
	FBMA (This paper)	0.785 (0.006)	0.745 (0.007)	0.764 (0.005)
NB	HAN **(Yang et al.)**	0.779 (0.012)	0.746 (0.024)	0.762 (0.015)
	BSA (This paper)	0.786 (0.008)	**0.757 (0.007)**	**0.771 (0.006)**
	FBMA (This paper)	0.785 (0.006)	0.745 (0.007)	0.764 (0.005)



(a) English (b) Spanish (c) MANSA

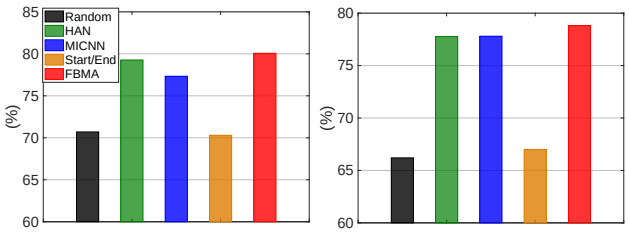
Figure 1: Comparison of event probabilities assigned by FBMA, MICNN and HAN on the CU English, Spanish and MANSA datasets. We can clearly see that mean probability is greater for FBMA in all the datasets for the event class. This depicts that FBMA is usually more confident than other methods in classifying the event articles. We also observe MANSA dataset contains some outliers (shown in red dots at the bottom).

variants as our baselines. The MI-CNN approach first constructs a sentence vector by applying convolution in the temporal dimension followed by $k-$ *maxpooling*. A document vector is formed from sentence vectors in a similar way. Instance representation for each sentence is constructed by concatenating sentence and document representations. Finally probabilities at instance level are aggregated to compute the document level probability. In our first baseline we replace the convolutional sentence encoder by a GRU sentence encoder followed by a simple attention mechanism that attends to words while constructing a sentence representation. We refer to this model as MI-GCNN and it extracts both trigger words and key sentences.

Since, the proposed model is a multi-aspect attention mechanism, we compare it with another multi-aspect attention mechanism proposed by Lin et al. [12]. We refer to as Hierarchical Self Attention (HSA) where we replace the word-level attention mechanism by the Self-Attentive mechanism proposed by them.

We also evaluated the Hierarchical Attention Network (HAN) model proposed by Yang et. al. [25], which attends to both trigger words and keys sentences while constructing sentence and document representations respectively but it only consists of a single aspect attention mechanism at both levels.

Finally, we replace the word level attention in the HAN model with the bilinear attention mechanism given by eq. 4 which has shown promising results in question answering tasks [2, 8].We refer to



(a) Population (b) Event

Figure 2: Event Extraction accuracy for event type detection and population class detection from key sentences extracted by different models. FBMA outperforms all the methods.

this model as Bilinear Single Aspect Attention model (BSA). We compare the performance of these models with the proposed FBMA model. In all the datasets, the event class is a rare class and hence we report the precision, recall and F1 score of that class for the test set. For each dataset we trained our model using 5-fold cross-validation with an 80/20 train/test split and employed early stopping. Models took less than an hour on a single Tesla P100 GPU to train.

5 RESULTS

5.1 Event Detection

Table 4 reports the performance of our models compared to the baseline approaches. On average, the FBMA model outperforms the MIL-based MICNN and MIGCNN models by 14.2 % & 12.9% on MANSA, 32.2 % & 3.5 % on CU Spanish and 6.6% & 3.9% on CU English datasets respectively. The FBMA model also outperforms HAN across the three datasets. Moreover, unlike HAN our model attends to different aspects while constructing a sequence representation, which results in an increased model size due to added parameters. One key aspect of our model is the ability to tune the dimensionality of the word level context vector $\mathbf{u}_w$. We find that models with $l \leq 2h$ where l is the dimension of $\mathbf{u}_w$ and $2h$ is the dimension of the word hidden representation tend to outperform models with $l > 2h$. The FBMA model is forced to learn a more compact representation of the word-level context vector and thus, retains only the most relevant information.

FBMA also beats HSA on CU English, CU Spanish & MANSA datasets by 5.1 %, 6.1 % and 1.5% respectively. Moreover our attention mechanism uses fewer parameters than the Self-Attentive model proposed by Lin et al. [12].

We also observe that the simple bilinear attention models outperform the dot product based attention models in HAN. For the CU English dataset, the BSA approach outperforms HAN & MIL-based baselines by 1.2% and 6.4% & 3.9% respectively. For the CU English and CU Spanish datasets we set $l = 2h$, whereas for MANSA dataset we set $l = 64$. These values were empirically found to perform best on the corresponding datasets.

5.2 Event Probabilities

For event encoding it is important that models have a high precision otherwise it may lead to false-positives and incorrect representations. This is also an issue with traditional detection approaches that detect events based on occurrence of certain set of trigger words from a pre-curated list, without taking into account the sentence context or relationship between different entities. Hence, it is important for the models to assign a high confidence to positive articles and a low confidence to negative articles. We present a distribution of event probabilities assigned to the positive articles by the FBMA, HAN and MICNN models for the test sets for all the three event datasets in Figure 1. We observe that generally the average probability assigned by FBMA is higher than MICNN and HAN confirming our hypothesis that having multi-aspect attention for each sentence increases the model confidence for a positive article.

6 CONCLUSION

In this paper we presented a framework for event detection. We used hierarchical attention based models for the task because of their ability to attend to words and sentences while constructing sentence and document representations respectively. With the hierarchical models we used our proposed FBMA mechanism which computes multiple attention distributions over words which leads to contextual sentence and document representations. Our results showed that this mechanism performed better than several other approaches and especially single-aspect mechanisms that miss out on the context because there is only one attention distribution. The proposed attention mechanism could easily be used for other tasks where document context is important. Moreover, it uses less number of parameters than other similar approaches, and hence can be scaled for larger datasets. An important open question to investigate in the future is that if the attention weights in each aspect can be constrained using certain rules to capture co-occurrence patterns of event arguments. This will lead to a wider distinction between different aspects captured and more precise event extraction.

7 ACKNOWLEDGEMENTS

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via contract 2018-18030100003 and by the U.S. Department of Homeland Security under grant award number 2017-ST-061-CINA01. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, US Department of Homeland Security, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014).

[2] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051* (2017).

[3] Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 167–176.

[4] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[5] Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence* 89, 1-2 (1997), 31–71.

[6] Hongyu Guo, Colin Cherry, and Jiang Su. 2017. End-to-end multi-view networks for text classification. *arXiv preprint arXiv:1704.05907* (2017).

[7] Jack Kiefer, Jacob Wolfowitz, et al. 1952. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics* 23, 3 (1952), 462–466.

[8] Jin-Hwa Kim, Kyoung-Woon On, Woosang Lim, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. 2016. Hadamard product for low-rank bilinear pooling. *arXiv preprint arXiv:1610.04325* (2016).

[9] Dimitrios Kotzias, Misha Denil, Nando De Freitas, and Padhraic Smyth. 2015. From group to individual labels using deep features. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 597–606.

[10] Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 73–82.

[11] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. [n. d.]. Factorized bilinear models for image recognition. ([n. d.]).

[12] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130* (2017).

[13] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[14] David McClosky, Mihai Surdeanu, and Christopher D Manning. 2011. Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*. 1626–1635.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[16] Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, Vol. 2. 365–371.

[17] Naren Ramakrishnan, Patrick Butler, Sathappan Muthiah, Nathan Self, Rupinder Khandpur, Parang Saraf, Wei Wang, Jose Cadena, Anil Vullikanti, Gizem Korkmaz, et al. 2014. 'Beating the news' with EMBERS: forecasting civil unrest using open source indicators. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1799–1808.

[18] Steffen Rendle. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 995–1000.

[19] Giuseppe Rizzo and Raphaël Troncy. 2012. NERD: A Framework for Unifying Named Entity Recognition and Disambiguation Extraction Tools. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*. 73–76.

[20] Parang Saraf and Naren Ramakrishnan. 2016. EMBERS AutoGSR: Automated Coding of Civil Unrest Events. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 599–608.

[21] Daisy Zhe Wang, Yang Chen, Sean Goldberg, Christan Grant, and Kun Li. 2012. Automatic Knowledge Base Construction Using Probabilistic Extraction, Deductive Reasoning, and Human Feedback. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*. 106–110.

[22] Wei Wang. 2018. *Event Detection and Extraction from News Articles*. Ph.D. Dissertation. Virginia Tech.

[23] Wei Wang, Yue Ning, Huzefa Rangwala, and Naren Ramakrishnan. 2016. A Multiple Instance Learning Framework for Identifying Key Sentences and Detecting Events. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. 509–518.

[24] Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. *CoRR* abs/1502.05698 (2015).

[25] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[26] Mo Yu, Matthew R Gormley, and Mark Dredze. 2015. Combining word embeddings and feature embeddings for fine-grained relation extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1374–1379.

Signals Matter: Understanding Popularity and Impact of Users on Stack Overflow

Arpit Merchant
IIIT-Delhi
arpitdm@gmail.com

Daksh Shah
IIIT-Delhi
daksh17336@iiitd.ac.in

Gurpreet Singh Bhatia
IIIT-Delhi
gurpreetb@iiitd.ac.in

Anurag Ghosh
IIIT-Hyderabad
anurag.ghosh@research.iiit.ac.in

Ponnurangam Kumaraguru*
IIIT-Delhi
pk@iiitd.ac.in

ABSTRACT

Stack Overflow, a Q&A site on programming, awards reputation points and badges (game elements) to users on performing various actions. Situating our work in Digital Signaling Theory, we investigate the role of these game elements in characterizing social qualities (specifically, popularity and impact) of its users. We operationalize these attributes using common metrics and apply statistical modeling to empirically quantify and validate the strength of these signals. Our results are based on a rich dataset of 3,831,147 users and their activities spanning nearly a decade since the site's inception in 2008. We present evidence that certain non-trivial badges, reputation scores and age of the user on the site positively correlate with popularity and impact. Further, we find that the presence of costly to earn and hard to observe signals qualitatively differentiates highly impactful users from highly popular users.

CCS CONCEPTS

• **Human-centered computing → Empirical studies in collaborative and social computing**; *Reputation systems.*

KEYWORDS

Crowdsourced Knowledge; Digital Signaling

ACM Reference Format:
Arpit Merchant, Daksh Shah, Gurpreet Singh Bhatia, Anurag Ghosh, and Ponnurangam Kumaraguru. 2019. Signals Matter: Understanding Popularity and Impact of Users on Stack Overflow. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313583

1 INTRODUCTION

Stack Overflow has evolved from a simple Q&A site to a massive social community where knowledge seekers and knowledge providers of all levels of expertise interact with each other to solve programming difficulties [2]. It has significantly impacted the way programmers learn, communicate and collaboratively build content repositories for future reference [6, 31, 34]. Due to its widespread use, it has become an integral part of the software development ecosystem and developers increasingly rely on it for their daily programming needs. Moreover, users on other platforms such as mailing lists, Github, etc. actively encourage their participants to refer back to posts on Stack Overflow for solutions [38].

This rise in the site's importance stems from four factors: (i) users can find multiple high quality answers for questions on nearly every programming language, tool, framework and software [28], (ii) if what is needed is not available, they can create a post themselves and receive answers extremely quickly [24], (iii) virtual rewards (reputation points and badges) incentivize users to contribute [14], and (iv) the rich interface enables them to display their expertise to potential recruiters [9]. These factors facilitate the transparent nature of the site. Each user has their own dedicated profile page that aggregates their contributions and achievements on the site. As a result, other users and recruiters can form impressions about their expertise of topics, their programming abilities, skills and experience [13]. In such a highly competitive environment, users that stand out are those that successfully acquire visible traces to attract attention [12]. One such significant way for users to stand out is by acquiring a large number of reputation points and badges.

Social status and reward system design. Virtual rewards act as symbols of social status, despite having no explicit value of their own. Some badges require users to expend costly effort and are therefore earned by few. These confer a higher status value since they distinguish members within the community. Others are easier to earn and act as motivations, and sources of learning. Badges serve various socio-psychological functions on crowdsourced platforms [4]. A prominent theme in literature has focused on the roles of badges in incentive structures [10, 19, 22, 27, 29]. Immorlica, et al. [21] show that the optimal design employs threshold badges where only users above a pre-defined number of contributions receive badges. Easley, et al. [16] take a game-theoretic approach to analyze the effectiveness of systems of such threshold badges.

Effects of virtual rewards. A separate line of research has analyzed the qualitative and quantitative effects of virtual rewards in diverse settings such as open-source software [29, 37] and knowledge repositories [38, 39]. Anderson, et al. [3] define a formal model that predicts how badges steer user behaviour. Mutter, et al. [26] provide empirical evidence that as users' proximity to goals defined by badges increases, so does the level of the users' contributions (goal-gradient hypothesis). First-time badges, awarded after a user takes a specific action for the first time, causally affect user behaviour and

*This work was done in part while Ponnurangam Kumaraguru was on sabbatical at IIIT Hyderabad.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313583

Table 1: General statistics of users' activities in our dataset.

Total number of active users	3,831,147
Total number of questions	15,711,957
Total number of answers	24,492,236
Mean reputation of active users	111
Mean number of badges earned by active users	22
Mean year when users joined the site	2015

also improve the functioning of the site itself [23]. The reputation points of users on Stack Overflow along with community activity dynamics are good predictors of the long-term value of questions and answers [2]. But, what attributes these virtual rewards can signal about users themselves is not yet well understood.

Research Questions. In this paper, we focus on finding important markers of user attributes since they are known to relate to dynamics of identity, crowdlearning, social benefits and societal acceptance [5, 36]. Specifically, we ask:

RQ1: According to Stack Overflow users, what social qualities (if any) do reputation scores and badges intend to signal?

RQ2: To what extent do these game elements actually signal or indicate the qualities that users expect them to?

The paper most closely related to ours is that of Trockman, et al [35]. They analyze various categories of badges such as Quality Assurance, Dependency Management, etc., in the `npm` ecosystem on Github as signals of *repository properties* such as dependency freshness, test suite quality and popularity. Some of these signals are subjective. Also, maintainers of the repositories can choose which badges they wish to display and which they do not. We consider Stack Overflow with a completely different and more complex system of reputation points and badges that it awards *to users* and is based on objective, pre-defined metrics [3].

We summarize our contributions below.

- We conduct a survey of Stack Overflow users and draw preliminary insights about how they view reputation points and badges as indicators of various social qualities.
- We perform empirical investigations on a large dataset of 3,831,147 users and the complete time-stamped history of their actions on Stack Overflow spanning a decade.
- Employing nonlinear regression models, we find that the presence of certain non-trivial badges correlates with higher popularity and impact. We also provide evidence that badges add more explanatory power compared to reputation scores.
- Statistical analyses of user activity show distinct differences in patterns of engagement between popular and impactful users.

Through these findings, we shed new light onto the role of virtual rewards in studying user qualities on crowdlearning platforms.

2 THEORETICAL FRAMEWORK

The widespread adoption of game elements on Stack Overflow invites a deeper examination of their effects on its users. Reputation scores are received for taking various positive actions whereas badges are awarded for "being especially helpful". We argue that given the variety of actions rewarded through reputation scores and badges, they are important signals of underlying qualities of users. We thus investigate their value from a signaling perspective.

Adverse Selection. Users on Stack Overflow possess different levels of information about various topics as well as other users on the platform. Users have a better understanding of their own expertise and limitations. They thus choose to participate selectively in order to maximize their benefits. Users however, tend to be uncertain about the preferences of heterogeneous audiences in terms of how they will respond to their actions. At the same time, the audience's qualitative assessment of users' abilities is based on limited information. Such a state where neither party has complete knowledge about the other is called information asymmetry [32]. This causes adverse selection, i.e. bias towards only particular kinds of actions [30]. For instance, most individuals prefer high returns and so they differentially choose low-hanging fruits, and broadly useful actions, while a few others may prefer more niche and challenging questions. As a result, participation is severely affected.

Digital Signaling. Signaling is a well-studied and popular solution to the problem of adverse selection [20, 32]. Signals are images, symbols and signs that allow users to communicate information and meaning with appropriate context. Signals that are costly to generate for the signaler and cognitively easy to process for the observer tend to be very reliable [11]. The design of sets of such assessment signals can specifically combat the inefficiencies arising due to information asymmetry [15]. The audience on Stack Overflow upvotes or downvotes posts to indicate that they approve or disapprove of them. This is a basic signal that is cheap to produce. Conversely, reputation scores and badges can help highlight deep technical qualities of a user since they require significant effort to achieve. This allows the user to potentially make better decisions in the future and the audience to gain more knowledge about him/her.

Gamification. Gamification is the use of game design elements in non-game contexts [14]. Badges on Stack Overflow are automatically earned by users based on their performance, unlike Github, where they are voluntarily displayed [7]. A single badge can holistically combine multiple qualitative actions whereas reputation points can be earned for every positive unit of action. This involves users in a social environment thereby motivating increased participation. In this paper, we consider game elements such as reputation scores and badges to be *digital signals* and investigate whether they are indicative of the performance and qualities of users.

3 DATA DESCRIPTION

Our experiments are conducted on a publicly available dataset containing all individual time-stamped actions of Stack Overflow users from the site's inception on July 31, 2008 to June 5, 2018 [33]. Table 1 describes a summary of the general statistics of our data.

Reputation. Reputation[1] scores are officially considered a "rough measurement of how much the community trusts you". Reputation is earned (or lost) when a user's question or answer is upvoted (or downvoted), when an answer is marked accepted by the user who originally asked the question, when bounties are received (or spent), or when suggested edits are accepted.

Badges. Badges are awarded in addition to reputation scores when the corresponding pre-defined set of actions and/or reactions are performed. They can be classified in two primary ways.

[1] https://stackoverflow.com/help/whats-reputation

– *Class-wise:* Bronze class badges are the easiest to obtain, Silver class badges require additional effort and Gold class badges are the hardest to earn. For instance, *Popular Question* is a bronze badge awarded to users when they ask a question that receives at least 1,000 views. *Notable Question* (silver) and *Famous Question* (gold) badges are awarded when the question receives at least 2,500 and 10,000 views respectively.
– *Category-wise:* Categories include Question, Answer, Participation, Moderation, Documentation, Tag and Other. Each category incentivizes users to conduct different kinds of actions. A user can earn multiple category badges as long as the requirements are fulfilled each time. For instance, a user obtains a new Nice Question badge for every question with a score of 10 or more.

Currently, there are 91 different badges[2] available on the site.

Active Users. We define active users as those who have asked at least one question, or have written at least one answer. We only consider these participants for our experiments to reduce the noise introduced by non-active users. Some users create throwaway accounts to ask a question. Our results hold when considering users who created up to 10 posts (questions and answers combined).
We also use other information available such as the time-stamps when users joined the site, number of questions, answers and comments they made, etc. The complete list of fields in the dataset along with detailed descriptions is available here.

4 USER SURVEY

We conducted an online survey of Stack Overflow users to gauge their views about the game elements and the platform in general.

Survey Design. We extracted email addresses of 2,740 users who had voluntarily shared this information in the About Me section of their profile pages on Stack Overflow. We divided these users into three groups based on their reputation scores (low, medium and high). Then, we randomly selected 500 users from each group and sent personalized invitations to participate in the survey. We received a total of 56 responses. Our respondents have a mean of 10 years of experience with coding/programming.
The survey[3] focused on two themes namely, (a) inferences regarding what reputation scores and badges can say about users, and (b) perceived effects of these game elements on the community. We also requested participants to indicate names of specific badges they considered important along with free-text boxes for longer comments, if any. The survey was piloted first.

Survey Results. The general consensus is that Stack Overflow is a good site to get multiple high-quality answers to programming questions. Across the three groups combined, 87% of the respondents either Strongly Agree or Agree with the statement *"Stack Overflow, in general, is more trusted than other communities for programmers."* Respondents felt that reputation scores tend to convey engagement, experience, contributions, helpfulness and knowledge. They consider badges like Good Answer, Popular Question, Pundit, Necromancer and Populist to be important. However, some answers mentioned that they did not consider reputation scores and badges to be important at all. Sixty five percent of respondents agreed with the statement that reputation scores are indicative of helpfulness but only 51% somewhat agreed that they indicate knowledge.

Survey Insights. Users interpret the importance of badges differently, yet badges and reputation scores have a subconscious impact on their future actions [23]. Our survey responses support these results. But they also point towards the question of what user attributes they can signal and to what extent. We focus on the following two attributes:

– *Popularity:* Respondents with low reputation consider reputation scores to be better indicators of user popularity than badges; yet majority of respondents with high reputation only somewhat agree with this statement.
– *Impact:* Fifty one percent of respondents feel that badges are a good measurement of how helpful and knowledgeable a user is (while another 30% somewhat feel the same). A larger majority, 67%, find reputation scores to be good indicators of expertise.

Note, we do not conclude our survey respondents' views to be representative of the entire community. Rather, we use these insights to design hypotheses which we then test empirically on the massive dataset described in Section 3.

5 CHARACTERIZING EFFECTS OF SIGNALS

Our goal is to identify important signals of popularity and impact of users based on their behaviour and actions on Stack Overflow.

Hypotheses. Based on survey insights, we test the following:

H1: Reputation scores and Badges are positively correlated with popularity as well as impact of users.
H2: Reputation scores are better indicators of popularity as well as impact compared to Badges.

Operationalization. To operationalize the two attributes in question, we adopt measures proposed by users, moderators and administrators on Meta Stack Exchange (a sister site for discussions on the workings and policies of Stack Overflow).

– *Popularity Score:* A user's place in the social landscape of Stack Overflow is the result of how they are perceived and how well they are known. We define the perceived popularity of a user to be the total number of distinct views on their profile page.
– *Impact Score:* A user's reach on the site is the number of people who have benefited from the user's actions. We consider the impact score of a user, as defined on Meta Stack Exchange[4], as the sum total views on questions, and answers with non-zero scores that have either been accepted, or are in the top 3 answers, or have a score of at least 5, or have at least 20% of the vote count.

Data Preparation. We preprocess our data as follows; (1) Since the distributions of both popularity and impact scores are heavy-tailed, we z-score transform them to capture the relative variation across users. (2) We create three sets of features namely, control, reputation, and number of badges. The Control Model (CM) consists of features such as number of days since the user joined the site, number of questions asked, number of answers given, etc. The Reputation Model (RM) consists of all the control features and one additional feature, i.e. the reputation score of the user. And the Badges Model (BM) similarly consists of all the control features plus

[2] https://stackoverflow.com/help/badges
[3] Complete survey questionnaire as administered to participants available here.
[4] Definition of impact score of users - A discussion on Meta Stack Exchange.

Table 2: Summary of importances of the exogenous variables for the Control, Reputation and Badges models. Table 2a and Table 2b show the results for predicting the popularity and impact of users, respectively. In each case, we present scores of the five most important badges. The importance values are relative to other exogenous variables within the specific model only.

Features	Control Model	Reputation Model	Badges Model
Age on the site	**0.319**	0.225	**0.191**
Number of questions	0.055	0.074	0.008
Number of answers	0.250	0.047	0.075
Number of upvotes	0.122	0.123	0.021
Number of downvotes	0.115	0.092	0.048
Reputation score		**0.313**	
Nice Answer Badges			0.062
Populist Badges			0.052
Enlightened Badges			0.029
Necromancer Badges			0.039
Good Answer Badges			0.031
	R^2 = 0.911	R^2 = 0.939	R^2 = 0.957

(a) Regression models for predicting Popularity of users.

Features	Control Model	Reputation Model	Badges Model
Age on the site	**0.321**	0.225	0.065
Number of questions	0.129	0.129	0.015
Number of answers	0.250	0.094	0.119
Number of upvotes	0.085	0.067	0.013
Number of downvotes	0.033	0.049	0.006
Reputation score		**0.394**	
Great Answer Badges			0.069
Revival Badges			0.055
Enlightened Badges			0.071
Necromancer Badges			**0.188**
Good Answer Badges			0.043
	R^2 = 0.685	R^2 = 0.767	R^2 = 0.858

(b) Regression models for predicting Impact of users.

one additional feature per badge, i.e. number of each such badges earned by the user (for all 91 badges on the site). We conduct an ablation study to compare their performances.

Model Fitting. We propose a Gradient Tree Boosting Regression model[5] to analyze the fit of the endogenous variable (popularity or impact score) from the exogenous variables (feature sets). We set the maximum tree depth as 3, learning rate as 0.1 and the number of boosted trees to fit as 100. We divide the data into training and testing sets and average the results over 50 runs of the experiment. We validate the model using the R^2 metric. Since it denotes how well the model fits the data points, higher values are better.
We also compute the relative importance scores of the features in each model. This score estimates the improvement in the squared error risk due to each feature compared to that for a constant fit [18]. Specifically, it is the average total decrease in impurity of a node across all trees in the ensemble. Decrease in impurity is the number of times a feature is used to split a node divided by the number of samples that it splits. It thus indicates how useful the feature was in the construction of the boosted decision tree model.

5.1 Signals of Popularity

Results. Table 2a presents the relative feature importances for fitting popularity scores within CM, RM and BM. For brevity, we report only the top five badges ordered according to their importance scores. BM explains 95.7% of the variance, while RM and CM explain 93.9% and 91.1% of the variance respectively.

Analysis. We observe RM provides more explanatory power compared to CM with a small, but significant increase in R^2 scores. This improvement in the model fit is due to the reputation points feature which also has the highest importance score. This indicates that it is a good predictor of popularity. That is, users with high reputation points tend to attract other users to their profile pages.

Notably, BM outperforms with RM and CM in terms of goodness-of-fit. This happens because badges aggregate various sets of actions thereby providing more information than just the reputation score. For instance, the most important badge feature is the number of Nice Answer badges. This badge is earned every time a user provides an answer that receives a score of 10 or more. More generally, we find that the five most important badges are all Answer Badges.

5.2 Signals of Impact

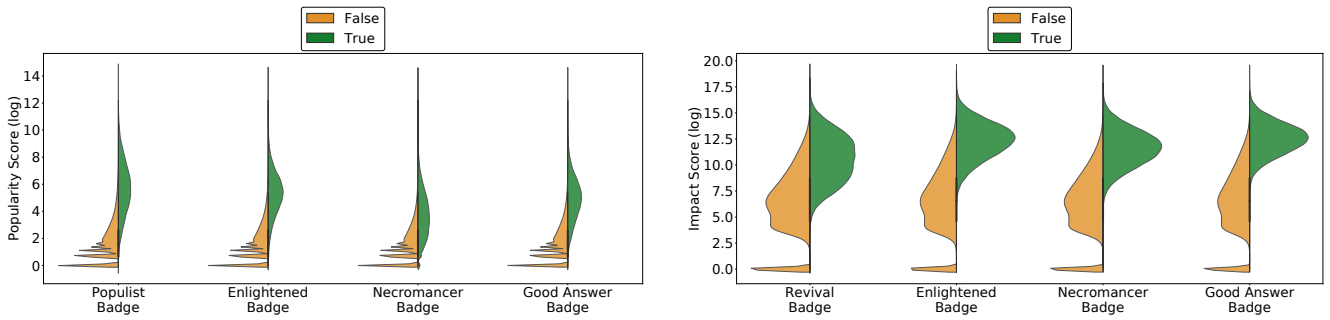
Results. Table 2b similarly presents the performances of the three models in fitting impact scores. CM, RM and BM models achieve R^2 scores of 68.5%, 76.7% and 85.8% respectively. Here too, we find that BM significantly outperforms the other two models.

Analysis. Once again, we find that reputation points are good predictors of impact. Yet, BM improves upon RM and CM because badges capture a more nuanced summary of the user's contribution. Reputation points increase not only due to upvotes on posts, but also on performing other actions such as useful edits, winning bounties, etc. This combines all positive actions into a single score thereby diluting it's effect. Contrasted with BM, consider the number of Necromancer badges earned by the user. A Necromancer[6] badge is awarded on posting an answer to a question at least 60 days after it has been asked and which receives a score of 5 or more. Two answers with the same score, but one written on the same day (say) and another written 60 days after the question was posted represent different value to the community. This is captured by the badge and not the reputation score. It is interesting to note, that once again each of the 5 most important badges are Answer badges.

As we can see in Table 2b, the importance score of the best feature is almost twice that of the next best feature in RM and BM both. But these two models exhibit largely different performance characteristics. Due to the smaller number of features, RM is extremely

[5]The code is available on Github at this url.

[6]https://stackoverflow.com/help/badges/17/necromancer



(a) **Popularity** (b) **Impact**

Figure 1: Distribution of popularity and impact scores of users (in log-scale) in the presence/absence of badges.

fast. BM is slower because it has more features and, in the context of Stack Overflow, they suffer from multi-collinearity. However, BM is more accurate because the ensemble of decision trees is able to separate users that have made helpful contributions on a variety of different metrics of helpfulness.

5.3 Discussion

Hypotheses. Our empirical results on the active Stack Overflow community mostly support our hypotheses and reveal interesting observations. High reputation corresponds with higher popularity and impact. Not all badges are good predictors; Documentation and Other badges show no correlation at all. However, certain Answer badges (such as, Necromancer, Enlightened and Good Answer) tend to be reliable signals. Figure 1 compares the distribution of users along their popularity and impact scores in the presence and absence of badges. This shows that users having these specific answer badges tend to be more popular and impactful than users that do not. The time since the user joined the site is an important signal of impact. And lastly, contrary to the impressions of users in our survey and the subsequent hypothesis (**H2**), badges are better predictors of popularity and impact compared to reputation.

Implications. Broadly useful and important questions can be asked by experienced and novice users alike. But writing impactful answers sometimes requires domain expertise. Different answers to the same question can be helpful to different users or to the same user at different times. Our results indicate that reputation scores seemingly fail to capture such nuances, whereas Answer badges appear adept at doing so. One potential explanation is that answers to some relatively easier questions with long-term value may have been posted during the initial years of the site [2]. Such answers yield a substantially high return on reputation to the original poster. Moreover, there exists evidence to suggest that reputation scores are easier to "farm" than badges through strategies such as writing answers in niche communities, or during off-peak hours [8].

Threats to Validity. We identify three primary threats to the validity of our approach. First, our metrics for computing popularity and impact scores are reductive. They are biased towards estimates of the number of views on profile pages and user posts obtained via internal site analytics. Second, we focus specifically on reward-based features and do not incorporate content-based features. Future work could examine linguistic attributes of posts that affect performance. Third, past evidence [17, 25] has shown that women have faced significant barriers to participating on the site. This suggests that game elements may be biased against some users. Thus, we advise caution in inferring broader interpretations of our results since we do not guarantee whether the positive links between badges and user attributes are causal or not.

6 DIFFERENTIATING POPULAR AND IMPACTFUL USERS

We now ask whether these game elements act as *differentiating signals* between popular and impactful users, and if so, why.

Thematic Representation of Users. Figure 2 depicts user distribution along two axes, popularity and impact. Time since the user joined the site is strongly associated with their standing in the community. As expected, a large concentration of users have low popularity and impact. Most new users along with a large fraction of the older users belong to this category. On the other hand, most highly popular and impactful users joined during the early years.

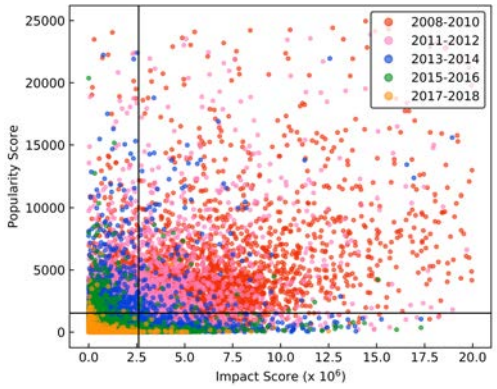
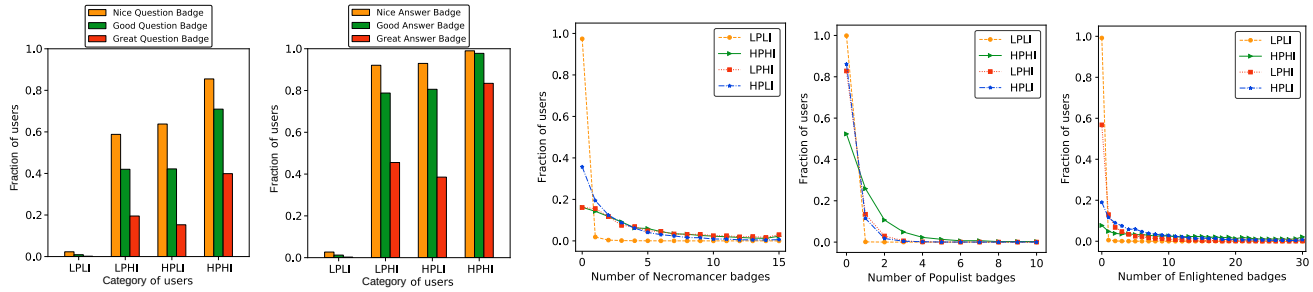


Figure 2: Distribution of users across popularity and impact scores based on the year in which they joined Stack Overflow. The horizontal and vertical black lines segment the population into the top 0.1%.

We segment users into four groups based on whether or not they belong to the top 0.1% of the community along the two social attributes: (a) high popularity, high impact (HPHI), (b) high popularity,



(a) **Fraction of users with badges for high scoring questions** (b) **Fraction of users with badges for high scoring answers** (c) **Distribution of number of Necromancer badges earned** (d) **Distribution of number of Populist badges earned** (e) **Distribution of number of Enlightened badges earned**

Figure 3: Relationship of LPLI, HPLI, LPHI and HPHI users to badges earned. Figures 3a and 3b study the presence of good quality question and answer badges among the different groups. Figures 3c, 3d and 3e depict the fraction of users in different communities that have earned multiple Necromancer, Populist and Enlightened badges.

low impact (HPLI), (c) low popularity, high impact (LPHI), (d) low popularity, low impact (LPLI). This segmentation is represented by the horizontal and vertical black lines in Figure 2. HPHI have mean popularity and impacts scores of 13,134 and 11,972,950 respectively, whereas LPLI have mean scores of 23 and 35,081 respectively. This shows the vast gulf between the two groups.

Figures 3a and 3b show the fraction of users belonging to each of the four categories HPHI, HPLI, LPHI and LPLI that have the particular badge. Interestingly, more LPHI, HPLI and HPHI users have badges for well-received answers (such as Nice Answer Badge) than they do for well-received questions (such as Nice Question Badge). Figures 3c, 3d and 3e display the distribution of the number of Necromancer, Populist and Enlightened badges earned by the four groups of users respectively. Consider the case of the Enlightened badge. We see that nearly 60% of LPHI users have zero Enlightened badges whereas only about 20% of HPLI users do not have that badge. We argue that there must be meaningful explanations that can be learned by comparing between these two groups.

Table 3: Differentiating between HPLI and LPHI users. ** = p < 0.01, *** = p < 0.001 represents statistical significance of Welch's t-statistic after Bonferroni correction ($p/14$).

Feature	HPLI	LPHI	t-statistic	Sig
Questions	54.65	42.02	-6.23	***
Answers	452.63	137.89	-39.97	***
Question Scores	233.15	286.61	4.38	**
Answer Scores	1190.07	679.83	-24.09	***
Reputation	16304.64	8672.31	-30.20	***
Necromancer Badges	2.47	6.32	29.6	***
Populist Badges	0.174	0.218	4.041	**
Great Answer Badges	0.682	0.887	7.84	***

We therefore examine HPLI and LPHI users and expect there to be differences in the way they contribute as well as reception to their contributions. Using Welch's t-test, we study the differences present between these two groups and present the features with the most significant differences between them in Table 3. We find that the number of questions and answers posted are significantly higher among HPLI, reflecting that they are more active. Conversely, the number of Necromancer, Populist and Great Answer Badges are higher for LPHI users. These badges appear to be signals that are costly to earn but not easily observable. Site design dictates that upvotes on answers return double the reputation points compared to upvotes on questions. Our findings show that LPHI users have a proportionally higher number of question and answer posts/scores. This implies that answers drive popularity, but it is questions that offer more influence. Further, some users link their SO accounts with other platforms such as LinkedIn, Github, etc. that may explain why they may be better known [1]. This is another potential source of divergence between high popularity and high impact.

7 CONCLUSION

The diverse range of actions and users, and massive quantity of content on Stack Overflow obfuscates the quality of information and efficiency of deliverables. It increases the transaction costs of participation. Game elements such as badges and reputation scores aim to provide incentives to balance these costs. But the design of these incentive structures has led to problems of adverse selection. In this paper, we present evidence that some of these game elements also act as reliable digital signals of social qualities such as popularity and impact. Our experiments reveal that certain non-trivial answer badges, high reputation scores and age of the user on the site indicate significant correlations. We also find differentiating characteristics that distinguish communities of popular and impactful users. We believe these insights offer guidance on combating inefficiencies arising out of bias towards specific actions. Our results encourage further exploration of the role of game elements as symbols of social status in socio-technical systems.

8 ACKNOWLEDGEMENTS

We thank Indira Sen, Shubham Singh, Shwetanshu Singh, and Srishti Aggarwal for useful discussions and the anonymous reviewers for detailed, helpful comments. We thank Language Technologies Research Centre (LTRC, IIIT-H) and Precog (IIIT-D) for their support. We are grateful to the respondents of our survey and to Stack Overflow for releasing their data publicly.

REFERENCES

[1] Kristen M Altenburger, Rajlakshmi De, Kaylyn Frazier, Nikolai Avteniev, and Jim Hamilton. 2017. Are there gender differences in professional self-promotion? an empirical case study of linkedin profiles among recent mba graduates. In *Eleventh International AAAI Conference on Web and Social Media*.

[2] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2012. Discovering value from community activity on focused question answering sites: a case study of stack overflow. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 850–858.

[3] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2013. Steering user behavior with badges. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 95–106.

[4] Judd Antin and Elizabeth F Churchill. 2011. Badges in social media: A social psychological perspective. In *CHI 2011 Gamification Workshop Proceedings*. ACM New York, NY, 1–4.

[5] Srinivasan Balakrishnan and Mitchell P Koza. 1993. Information asymmetry, adverse selection and joint-ventures: Theory and evidence. *Journal of economic behavior & organization* 20, 1 (1993), 99–117.

[6] Andrew Begel, Jan Bosch, and Margaret-Anne Storey. 2013. Social networking meets software development: Perspectives from github, msdn, stack exchange, and topcoder. *IEEE Software* 1 (2013), 52–66.

[7] Kelly Blincoe, Jyoti Sheoran, Sean Goggins, Eva Petakovic, and Daniela Damian. 2016. Understanding the popular users: Following, affiliation influence and leadership on GitHub. *Information and Software Technology* 70 (2016), 30–39.

[8] Amiangshu Bosu, Christopher S Corley, Dustin Heaton, Debarshi Chatterji, Jeffrey C Carver, and Nicholas A Kraft. 2013. Building reputation in stackoverflow: an empirical investigation. In *2013 10th Working Conference on Mining Software Repositories (MSR)*. IEEE, 89–92.

[9] Andrea Capiluppi, Alexander Serebrenik, and Leif Singer. 2013. Assessing technical candidates on the social web. *IEEE software* 30, 1 (2013), 45–51.

[10] Yan Chen, F Maxwell Harper, Joseph Konstan, and Sherry Xin Li. 2010. Social comparisons and contributions to online communities: A field experiment on movielens. *American Economic Review* 100, 4 (2010), 1358–98.

[11] Brian L Connelly, S Trevis Certo, R Duane Ireland, and Christopher R Reutzel. 2011. Signaling theory: A review and assessment. *Journal of management* 37, 1 (2011), 39–67.

[12] Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. 2012. Social coding in GitHub: transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 conference on computer supported cooperative work*. ACM, 1277–1286.

[13] Laura Dabbish, Colleen Stuart, Jason Tsay, and James Herbsleb. 2013. Leveraging transparency. *IEEE software* 30, 1 (2013), 37–43.

[14] Sebastian Deterding, Miguel Sicart, Lennart Nacke, Kenton O'Hara, and Dan Dixon. 2011. Gamification. using game-design elements in non-gaming contexts. In *CHI'11 extended abstracts on human factors in computing systems*. ACM, 2425–2428.

[15] Judith Donath. 2007. Signals in social supernets. *Journal of Computer-Mediated Communication* 13, 1 (2007), 231–251.

[16] David Easley and Arpita Ghosh. 2016. Incentives, gamification, and game theory: an economic approach to badge design. *ACM Transactions on Economics and Computation (TEAC)* 4, 3 (2016), 16.

[17] Denae Ford, Justin Smith, Philip J Guo, and Chris Parnin. 2016. Paradise unplugged: Identifying barriers for female participation on stack overflow. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. ACM, 846–857.

[18] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2001. *The elements of statistical learning*. Vol. 1. Springer series in statistics New York, NY, USA:.

[19] Jordan Frith. 2013. Turning life into a game: Foursquare, gamification, and personal mobility. *Mobile Media & Communication* 1, 2 (2013), 248–262.

[20] Tim Guilford and Marian Stamp Dawkins. 1991. Receiver psychology and the evolution of animal signals. *Animal behaviour* 42, 1 (1991), 1–14.

[21] Nicole Immorlica, Greg Stoddard, and Vasilis Syrgkanis. 2015. Social status and badge design. In *Proceedings of the 24th international conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 473–483.

[22] Shaili Jain, Yiling Chen, and David C Parkes. 2009. Designing incentives for online question and answer forums. In *Proceedings of the 10th ACM conference on Electronic commerce*. ACM, 129–138.

[23] Tomasz Kuśmierczyk and Manuel Gomez-Rodriguez. 2018. On the Causal Effect of Badges. (2018).

[24] Lena Mamykina, Bella Manoim, Manas Mittal, George Hripcsak, and Björn Hartmann. 2011. Design lessons from the fastest q&a site in the west. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2857–2866.

[25] Anna May, Johannes Wachs, and Anikó Hannák. 2019. Gender differences in participation and reward on Stack Overflow. *Empirical Software Engineering* (2019), 1–23.

[26] Tobias Mutter and Dennis Kundisch. 2014. Behavioral mechanisms prompted by badges: The goal-gradient hypothesis. (2014).

[27] Oded Nov, Ofer Arazy, and David Anderson. 2014. Scientists@ Home: what drives the quantity and quality of online citizen science participation? *PloS one* 9, 4 (2014), e90375.

[28] Chris Parnin, Christoph Treude, Lars Grammel, and Margaret-Anne Storey. 2012. Crowd documentation: Exploring the coverage and the dynamics of API discussions on Stack Overflow. *Georgia Institute of Technology, Tech. Rep* (2012).

[29] Jeffrey A Roberts, Il-Horn Hann, and Sandra A Slaughter. 2006. Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects. *Management science* 52, 7 (2006), 984–999.

[30] Michael Rothschild and Joseph Stiglitz. 1978. Equilibrium in competitive insurance markets: An essay on the economics of imperfect information. In *Uncertainty in economics*. Elsevier, 257–280.

[31] Vandana Singh, Michael B Twidale, and David M Nichols. 2009. Users of open source software-how do they get help?. In *System Sciences, 2009. HICSS'09. 42nd Hawaii International Conference on*. IEEE, 1–10.

[32] Michael Spence. 2002. Signaling in retrospect and the informational structure of markets. *American Economic Review* 92, 3 (2002), 434–459.

[33] Stack Exchange Inc. 2018. 2018. Stack Exchange Data Dump 2018-05-01. https://archive.org/details/stackexchange/.

[34] Margaret-Anne Storey, Christoph Treude, Arie van Deursen, and Li-Te Cheng. 2010. The impact of social media on software engineering practices and tools. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*. ACM, 359–364.

[35] Asher Trockman, Shurui Zhou, Christian Kästner, and Bogdan Vasilescu. 2018. Adding Sparkle to Social Coding: An Empirical Study of Repository Badges in the npm Ecosystem. In *Proc. International Conference on Software Engineering (ICSE)(ICSE). ACM.*

[36] Utkarsh Upadhyay, Isabel Valera, and Manuel Gomez-Rodriguez. 2017. Uncovering the Dynamics of Crowdlearning and the Value of Knowledge. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 61–70.

[37] Bogdan Vasilescu. 2014. Human aspects, gamification, and social media in collaborative software engineering. In *International Conference on Software Engineering, Doctoral Symposium (ICSE)*. ACM, New York, NY, USA, 646–649. https://doi.org/10.1145/2591062.2591091

[38] Bogdan Vasilescu, Alexander Serebrenik, Prem Devanbu, and Vladimir Filkov. 2014. How social Q&A sites are changing knowledge sharing in open source software communities. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. ACM, 342–354.

[39] Gang Wang, Konark Gill, Manish Mohanlal, Haitao Zheng, and Ben Y Zhao. 2013. Wisdom in the social crowd: an analysis of quora. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1341–1352.

Fine-grained Type Inference in Knowledge Graphs via Probabilistic and Tensor Factorization Methods

A B M Moniruzzaman
Queensland University of Technology
Brisbane, Australia
a.moniruzzaman@qut.edu.au

Richi Nayak
Queensland University of Technology
Brisbane, Australia
r.nayak@qut.edu.au

Maolin Tang
Queensland University of Technology
Brisbane, Australia
m.tang@qut.edu.au

Thirunavukarasu Balasubramaniam
Queensland University of Technology
Brisbane, Australia
thirunavukarasu.balas@qut.edu.au

ABSTRACT

Knowledge Graphs (KGs) have been proven to be incredibly useful for enriching semantic Web search results and allowing queries with a well-defined result set. In recent years much attention has been given to the task of inferring missing facts based on existing facts in a KG. Approaches have also been proposed for inferring types of entities, however these are successful in common types such as 'Person', 'Movie', or 'Actor'. There is still a large gap, however, in the inference of fine-grained types which are highly important for exploring specific lists and collections within web search. Generally there are also relatively fewer observed instances of fine-grained types present to train in KGs, and this poses challenges for the development of effective approaches. In order to address the issue, this paper proposes a new approach to the fine-grained type inference problem. This new approach is explicitly modeled for leveraging domain knowledge and utilizing additional data outside KG, that improves performance in fine-grained type inference. Further improvements in efficiency are achieved by extending the model to probabilistic inference based on entity similarity and typed class classification. We conduct extensive experiments on type triple classification and entity prediction tasks on Freebase FB15K benchmark dataset. The experiment results show that the proposed model outperforms the state-of-the-art approaches for type inference in KG, and achieves high performance results in many-to-one relation in predicting tail for KG completion task.

KEYWORDS

Knowledge Graph, Type Inference, Fine-grained Type, Semantic Web Search, Tensor Factorization

ACM Reference Format:
A B M Moniruzzaman, Richi Nayak, Maolin Tang, and Thirunavukarasu Balasubramaniam. 2019. Fine-grained Type Inference in Knowledge Graphs via Probabilistic and Tensor Factorization Methods. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313597

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313597

1 INTRODUCTION

Knowledge Graphs (KGs) mainly describe real-world objects and their interrelations in various topical domains by organized in graphs. KGs are web-scale probabilistic knowledge bases that store billions of facts about people, places, and things that are often represented in the form of a triplet (head entity, relation, tail entity). The way KGs are enriching information about the real worlds objects and connections by combine extractions from web contents (obtained via analysis of text, tabular data, page structure, and human annotations) with prior knowledge derived from existing knowledge repositories [42] [47]. Examples of prominent large-scale KGs are Yago [54] [28] [11], Freebase [16] [3], DBpedia [14] [37] [1], NELL [20] [5], Google's Knowledge Graph, Microsoft's Satori, Probase [7] , and Google Knowledge Vault [27]. These KGs have become powerful assets for improving Web search by providing richer data for Entity Pane, Carousel, and Facts Across Segments in the search panel. Additionally, KGs are becoming important resources for different Artificial Intelligence (AI) and Natural Language Processing (NLP) applications. Despite their seemingly large size, these knowledge graphs are still incomplete with missing information, such as type, properties, and relations [42] [47] [23] [57] [63] [50].

Types in KG are used to express the concept of classes. According to KG idiomatic usage, a KG object "has X, Y, Z types" is equivalent to an object "is a member of the X, Y, Z classes". In the case of U.S. President Barack Obama, the KG object for Obama would have the types *person* and *US president* to indicate that the object is a member of the Persons and U.S. Presidents. Fine-grained types are sub-classes of type classes in KG. For example, a type class *'Actor'* may has *American Actor, Film Actor, TV Actor, Hollywood Actor, Female Actor, Voice Actor* or *Comedy Actor* sub classes. The Actor typed entities in KG are therefore can be further defined with these fine-grained types.

Defining fine-grained types of entities in KG allows Web search queries with a well-defined result set. When someone doing a search on the web, the search engines need to understand the words what they refer to real-world things, and by this, search engines can do a better search with just the contents when someone looks up on the web. The types of entities in KG provide the meaning of

the words, such as building is a building object and animal are animal objects; they are not just a random string of characters. Fine-grained typed entities and entities with complete set of types in KG, make possible to explore lists and collections with web searching by 'Knowledge Graph Carousel' in popular search engines. For example, you can explore, how many woman won the Noble prize? or list of Renaissance painters.

Type inference is concerned with inferring the existence (or probability of existence) of missing types for instances (objects or entities) in KGs. In a KG, not every entity may come with proper type information. Untyped entities and entities with incomplete set of types are a common problem in Semantic Web KGs [42] [47]. In recent years much attention has been observed in KG completion task that automatically infer missing facts based on existing facts in a KG [23] [57]. Approaches have also been proposed for inferring types of entities, however these are successful in common types such as 'Person', 'Movie', or 'Actor'. There is still a large gap, however, in the inference of fine-grained types which are highly important for exploring specific lists and collections within web search. Generally, very limited numbers of observed fine-grained types data (RDF-triples with fine-grained type information) present in KGs, and this poses a high challenge in developing approaches for inferring entities of the fine-grained types.

This paper proposes a new approach based on a tensor model for fine-grained type inference. This new approach is explicitly modeled for leveraging domain knowledge and utilizing additional data outside KG, that allows a ranked representation for type inference and result in richer data for tensor factorization. Further improvements in efficiency are achieved by extending the model to *probabilistic inference* based on entity similarity and typed class classification. We evaluate our approach with two standard evaluation tasks *type triple classification* and *entity prediction* on benchmark dataset Freebase FB15K dataset [4] and FB15K-237 Knowledge Base Completion Dataset [2]. The experiment results show the proposed model outperforms the state-of-the-art approaches for type inference in KG (10% to 19% on different fine-grained types compare to Rescal [43]), and achieve high results in many-to-one relation in predicting tail for KG completion task.

2 RELATED WORK

Given the importance of missing type assertions task, there are some approaches based on *Reasoning*, *Probabilistic Method* [49], *Hierarchical Classification Approach* [40] [41] [60], *Association Rule Mining* [32] [46], *Topic Modelling* [52], *Support Vector Machine (SVM)* [51], *k-nearest neighbors classifier (KNN)* [45], and *Tensor Factorization* [44] have been proposed and applied on different in KGs are summarized in Table 1. All approaches have merits and demerits in performance for missing type assertions task in KG.

In [52],the use of *Topic Modeling* for type prediction is proposed. Entities in a KG are represented as documents, on which *Latent Dirichlet Allocation (LDA)* [15] is applied for finding topics. In [45], [51] authors exploit interlinks between the KGs, and Wikipedia predict types in a KG. In [46], exploit such association rules to predict missing types in DBpedia based on redundancies different type systems of KGs. As topics, the interlinks and the type redundancies between KGs can only be covered limited associated top level types, these approaches are not capable for fine-grained type inference.

Some approaches have been proposed for type inference in KG such as SDType [49] a heuristic approach, SNCN [41], a hierarchical classification approach. Since SDType is based on statistical method and there are not enough facts exits in KG for fine-grained types [48], [49], this approach is not capable to produce meaningful results in predicting fine-grained types. Fine-grained types basically are not systematically structured with a type hierarchy in KGs and SNCN relay on the hierarchical type structure in KG, this approach is therefore not efficient in fine-grained type inference. Besides, some KGs (such as Freebase) are not systematically structured with a type hierarchy, in this case, this approach is not useful for this task. Though, SDType, and SNCN algorithm is successful in predicting higher-level classes (Actor), while predicting fine-grained classes (such as VoiceActor or AmericanActor) is much more difficult. None of these methods are able to capture the interactions of latent features that obtain probabilistic likelihoods of type-relations existing between entities (objects) in KG. Tensor modeling, a well-known approach to represent and analyze latent relationships inherent in multi-dimensional data [35].

RESCAL [43] [44] is the state-of-the-art method for link prediction and type inference in KGs that has been used for type inference on YAGO [11] entire KG [44]. Though, RESCAL has been achieved good performance among the state-of-the-art approaches for type inference in KG, the performance of type inference (fine-grained types) are significantly lower than the performance of top level (common) types in KG. Furthermore, this method and other existing methods for type inference in KG are not able to utilize the domain knowledge in KG and the additional sources of information outside of KG. However, for type inference in KG, it has not been explored that the utilization of additional sources of information by combining knowledge base and text. Our approach is based on tensor factorization method which is specially designed for fine-grained type inference in KG. This approach is able to capture interactions of entities outside of KG and to learn type-domain knowledge in KG. Furthermore, this model is designed to learn type hierarchy structure and similarities of entities for type; and utilize these information to improve fine-grained type inference results in KG.

3 UTILIZING ADDITIONAL DATA

Semantic Web KGs are represented in the form of the Resource Description Framework (RDF) [8] triples <*subject, predicate, object*>, where the subject and object are entities and the predicate is the relation type, (e.g. <*Steve Jobs, founderOf, Apple*>). Uniform Resource Identifier (URI) uniquely identifying each entity in Semantic Web KGs. For instance, entity *Donald Trump* can be found as *http://dbpedia.org/page/Donald_Trump* in DBpedia, and as *https://en.wikipedia.org/wiki/Donald_Trump* in Wikipedia. Every entity in Semantic Web KGs (e.g. DBpedia or Freebase) has anchor link to Wikipedia page, where a document of information is available for an associated entity.

Each entity in Freebase is uniquely identified by a MID (Machine ID). For instance, */m/0cqt90* as MID is used for *Donald Trump* entity. Using MID and the predicate *http://www.w3.org/2002/07/owl#sameAs*

for each entity in Freebase, we can issue a SPARQL query to retrieve the corresponding DBpedia, or Wikipedia URIs. For example, Donald Trump, the current President of USA, is mapped via the triple by SPARQL query. The texts (information) available for an associated entity can be represented as full lexicalized dependency paths. For instance, the entity in KG such as *Donald Trump*, or *United States* has textual information in Wikipedia page, or other online news repositories (e.g. New York Times or ClueWeb), as illustrated in Fig 1.

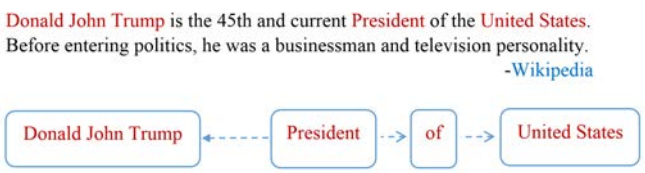


Figure 1: A full lexicalized dependency path from textual data.

Figure 1 shows the lexicalized dependency path between two entities that occur together in a sentence. An instance of textual relation of type "<nsubj> president <prep> of <obj> United States", corresponding to the sentence is added as additional data based on this occurrence. These full lexicalized dependency path from textual data are used as relations between Freebase entities by mapping with MID. For instance, the entity */m/0cqt90* (Donald Trump) in RDF triple relationship in Freebase as: (*/m/0cqt90, <President_of>, /m/01kw0b*). In addition to KG some other relations can be obtained from other sources by mapping MID. Relations connected to entity */m/0cqt90* (MID) from additional data sources (see in Fig 2). Relations come from additional data sources (outside from KG) may be useful information about entities in KG. From modelling perspective, tensor representations are appealing to RDF KGs because they provide an elegant way to represent multiple RDF triples.

We propose the *weighted-tensor* interpretation scheme to effectively model KG and text-based data for constructing the tensor. Without applying weight in tensor construction, the objective function would place equal importance on fitting both observed and unobserved values. As KGs are typically very large scale, and constructed tensors are therefore often very sparse, this will result in fitting a large number of unobserved data and uncertainty in the observations. The weighting tensor prevents this from happening by emphasizing only the observed (or certain) entries. Each RDF triples in KG expresses facts (complete meaningful information) about real world object, and whereas each text based triples provides less important facts (auxiliary information) about the real world object. The observed data for factorization is therefore obtained from combination of KG and other dataset which contains textual relations. The interpretation of RDF KG can be interpreted as a tensor, where first mode of a tensor therefore models the occurrences of all entities as a subject, the second mode models the occurrences of all entities as an object, and the third mode models the different relations. The incorporation of additional data for type inference in KG becomes possible via a tensor factorization model where KG and additional data consisting of e entities and r different relations can then be represented in form of a tensor $X \in R^{e \times e \times r}$ with entities

$$X_{i,j,k} = \begin{cases} 2, & \text{if the relationship } r_k(e_i; e_j) \text{ exists in KG.} \\ 1, & \text{if the relationship } r_k(e_i; e_j) \text{ exists in additional data.} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The values ('2' and '1') and ('0') of X_{ijk} come from tensor model are regarded as observed and un-observed data respectively for tensor factorization. Weighting the tensor by this scheme allows a ranked representation for type inference and result in richer data for tensor factorization, that leads to an improvement in efficiency for fine-grained type inference.

4 LEVERAGING DOMAIN KNOWLEDGE

Most KGs store facts about real-world objects covering only numbers of specific domains (e.g. *"Movie"*, *"Book"*, or *"Place"*). For instance, types in KG such as *actor, film, director,* or *producer* and fine-grained types such as *filmActor, TVActor, regularActor, guestActor, executiveDirector, AssistantProducer* are in *"film"* domain. Given the importance the fine-grained inference task in KG, typed entities (objects) for given fine-grained types in one domain (such as *"film"* domain) are less likely to be entities in other domains (such as *"book"*, or *"place"*). For instance, inferring entities for fine-grained types (such as regularActor, guestActor) would be a typed in Actor, those entities generally are in same domain in KG.

The collaborative activities between the entities in *"film"* domain are therefore higher importance for fine-grained type inference in this domain. However, interactions among entities with different relation types can be among different domains as well, as types of entities in KG may be come from different domains. Domain information can be obtained from KG, such as in Freebase each relation start with domain e.g. */film/film/starring/film/performance/actor*, and */tv/tv_program/regular_cast/tv/regular_tv_appearance/actor*; in DBPedia domain information can be found in the property of a relation by SPARQL [9] query with *http://www.w3.org/1999/02/22-rdf-syntax-ns#Property*. For example, a set of actor typed entities can be a set of author typed where these entities are involved with different domains. Ignoring these information (inter-domains collaboration activities of entities) may effect on latent features learning by factorization. Modelling domain knowledge (the interactions of entities in the domain, and inter-domins) is therefore prevents this happening and leverages the domain information in factorization for an improvement in fine-grained type inference.

We propose the *domain-relevance weighted tensor (DrWT)* method for fine-grained type inference, that explicitly model a mapping from the domain-based representation to the observed tensor in factorization. Given the KG and additional data along with list of labelled domain, *DrTW* classifies the entries correspondence to the domains in KG and additional dataset.

DrWT Algorithm:

Input: *Knowledge Graph* $G = (D, E, R)$*; where set* D *is a set of domains in KG,* $D = \{d_1, ..., d_{|d|}\}$*; set* E *is a set of entities (objects),* $E = \{e_1, ..., e_{|e|}\}$*, and* E_s *as subject set of entities which occur as subject in relation links, where* $E_s \in E$*; and and* E_o *as object set of entities which occur as object in relation links where* $\{E_o \in E\}$ *and, the set* R *is a set of relations (Predicates) between entities,* $R = \{r_1, ..., r_{|r|}\}$.

Output: set ∇T_d is a list of triples $<< (E_s)_i, (R)_i, (E_o)_i >>$ for each $d_i \in D$ (Domains)
Start
01: **function** DomainRelevance (D, E, R)
02: $d \leftarrow domain$
03: **for each** $r_i \in R$ **do**
04: **for each** $r_i \in d$ **do**
05: $source\ (e_s) : R \rightarrow E_s$ ▷ return source(e)
06: $target\ (e_o) : R \rightarrow E_o$ ▷ return target(e)
07: $e_d \leftarrow (e_s \cup e_o)$
08: **while** $e_i \in e_d$ OR $r_i \in d$ **do**
09: $T_d \leftarrow$ SELECT-TRIPLES $\{(e_s)_i, (r)_i, (e_o)_i\}$
10: **end while**
11: $\nabla T_d \leftarrow T_d$
12: **end for**
13: **end for**
14: **end function**

Applying *DrWT* the populated tensor is then represented the collaborative activities of the type domain can be modelled subject, object, predicate and domain in 4th-order tensor presentation. Once the 4th-order tensor representations are constructed applying DrWTF method, the next steps is latent factors generation via tensor factorization, where tensor factorization technique is applied to learn effectively the latent factors for fine-grained type inference.

5 FINE-GRAINED TYPE INFERENCE WITH TENSOR MODEL

We use CP [21] [30] based *4th-order Tensor Factorization* for deriving the latent relationships between dimensions of the tensor model. After latent factors generation via tensor factorization, we therefore follow *tensor reconstruction* process to reveal new entries that are inferred from the latent factors. the *4th-order Tensor Factorization* model, CP factorizes each forth-order domain-relevance weighted tensor $\mathcal{X} \in \mathbb{R}^{S \times O \times P \times D}$, can be formulated as

$$f(\mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{H}) = \|\mathcal{X} - \mathbf{E}, \mathbf{F}, \mathbf{G}, \mathbf{H}\|^2 \tag{2}$$

where $\mathcal{X}$ is factorized using a CP model on each mode-n matricization and results in four latent factors matrices, $\mathbf{E} \in \mathbb{R}^{S \times R}$, $\mathbf{F} \in \mathbb{R}^{O \times R}$, $\mathbf{G} \in \mathbb{R}^{P \times R}$ and $\mathbf{H} \in \mathbb{R}^{D \times R}$ corresponding to the each dimensions of tensor $\mathcal{X}$. R is the column size of the corresponding latent factor matrices; S, O, P and D are denoted as *Subject, Object, Predicate* and *Domain* of entries to tensor. The diagonal core tensor $\hat{\mathcal{C}} \in \mathbb{R}^{R \times R \times R \times R}$, that defines the interaction between the *Subject, Object, Predicate* and *Domain* can be calculated by multiplying all the latent factors. Afterwords, the reconstructed tensor $\hat{\mathcal{X}}$ is derived multiplying the core tensor by all latent factor matrices:

$$\hat{\mathcal{X}} := \hat{\mathcal{C}} \times_s \hat{\mathbf{E}} \times_p \hat{\mathbf{F}} \times_o \hat{\mathbf{G}} \times_d \hat{\mathbf{H}} \tag{3}$$

Where the core tensor $\hat{\mathcal{C}}$ and the feature matrices $\hat{\mathbf{E}}$, $\hat{\mathbf{F}}$, $\hat{\mathbf{G}}$ and $\hat{\mathbf{H}}$ are the model parameters that have to be learned and $\mathcal{X}$ is the tensor product to multiply a matrix on dimension x with a tensor.

The task of fine-grained type inference in KG is to infer which subject entities (s) of a type relation (p) is missing to object entity (o). That means that the model has to infer from the values of $\hat{x}_{s,p,o,d}$ of the domain tensor $\hat{\mathcal{X}}$ (from equation no. 5). Instead of inferring single element (subject entities) from $\hat{\mathcal{X}}$, this model provides a list of best possible (ranked) elements (subject entities) of a fine-grained type relation. Given a reconstructed tensor $\hat{\mathcal{X}}$ the list of top of the N highest scoring entities for a given fine-grained type relation (p) and an object entity (o)

$$Top(N, p, o, d) := argmax^N_{t \in T}\{\hat{x}_{s,p,o,d}\} \tag{4}$$

To infer entities for fine-grained type inference, we define appropriate thresholds for each of fine-grained type relations. These thresholds are therefore used as metric in the inference model so that the missing entries can be regarded as either "relevant" (score higher than the threshold) or "irrelevant" (score lower than the threshold) entries from the revealed new entries that are inferred from the latent factors. In order to further improve the performance of inference to the missing entities for fine-grained types, the inference model is extended to *probabilistic inference* approach with multilabel classification of type classes.

5.1 Probabilistic Inference with Collective Multilevel Type Classification and Typed Entity Similarity

The probabilistic inference in an alternative approach to define the probability of one set typed entities to be other fine-grained typed by observing data from classification of type classes, and the latent similarity of entities in KG. In this case, the list of fine-grained types is ranked and generated based on the probability score of an entity to select candidate entities, in which the entity typed classes, and entity similarity to the fine-grained type classes are taken into account. The underlying idea of probabilistic ranking is that collective multilabel classification of entities where entities are allowed to have more than one class types and related entities often share identical classes in RDF KG. For instance, en entity of *Actor* typed is also typed of *Artist, Person,* and *Thing* classes (see in Fig. 2). Fine-grained type inference further can be cast as a multilabel

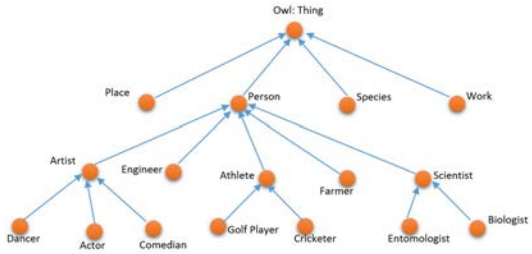


Figure 2: A part of DBPedia type hierarchy.

classification problem, by introducing type classes as entities to the data, and inferring the probability of relationships *rdf-type*.

As RDF KG is multi-relational data, the similarity of entities is therefore determined by the similarity of their relationships, following the intuition that "if two objects are in the same relation to the same object, this is evidence that they may be the same object" The collaborative activities of entities as subjects $\mathbf{A_s} \in \mathbb{R}^{S \times P}$ and objects $\mathbf{A_o} \in \mathbb{R}^{O \times P}$ in relations in a domain can be modelled by the entity matrix $\hat{\mathbf{A}}$, where $\hat{\mathbf{A}}$, is QR matrix factorization [22] [29] of $\sum(A_s + A_o)$. For each domain the the latent space $\hat{\mathbf{A}}$ therefore

reflects the similarity of entities in the relational domain. The type or fine-grained type classes set $C_e = \{t_1, t_2, t_3, ...t_n\}$ where C_e is a set of Types in one KG. A list of type or fine-grained type classes that are considered for given fine-grained type. For each fine-grained type in C_e the candidate entities set, $\hat{E}_t = \{\hat{e}_1, \hat{e}_2, \hat{e}_3, ...\hat{e}_n\}$ where E_t is a set of typed entities in one KG.

We use the Bayes' theorem [26] [61] for predicting the class candidate entity E_t that have the highest posterior probability given C_e, $p(C_e|E_t)$. The posterior probability is utilized to calculate the preference probability of an entity e to be fine-grained typed t in C_e type classes by observing current type classes of entity e, and latent similarity of entity e to fine-grained typed entity. The conditional probability can be formulated as:

$$p(C_e|E_t) = \frac{p(E_t|C_e)p(C_e)}{p(E_t)} \tag{5}$$

where *prior probability* $p(C_e)$ is the prior distributions of parameter set C_e in a single domain before E_t is observed, that is relative frequency with which observations from that class occur in a population. Generally, prior probability for fine-grained type classes are lower compared to top level type classes in KG. $p(C_e|E_t)$ is the joint probability of observing type class preference set C_e given E_t, and entity similarity preference given fine-grained type t. Using the assumption of multinomial event model distribution for the Naive Bayes classifier, the posterior probability p_{e_n,t_r} for fine-grained type t_e with fine-grained type class C_e for candidate entity e_n, an instance of E_t, is obtained by multiplying the *prior probability* of t_e, $P(C_e = t_e)$, with the probability of preference candidate entity e_n, an instance of E_t, given t_e, $P(e_n|C_e = t_e)$:

$$p_{e_n,t_e} = p(t_e|C_e) = \sum_{t=1}^{|C_e|} P(e_n|C_e = t_e) \prod_{\hat{e}=1}^{|\hat{E}_e|} P(\hat{E}_e|C_e) \tag{6}$$

where, $P(\hat{E}_e|C_e)$ is probability of likelihood for t_e in C_e, is derived from the entities set, $\hat{E}_t = \{\hat{e}_1, \hat{e}_2, \hat{e}_3, ...\hat{e}_n\}$ where values from reconstructed tensor $\hat{X}$, and entity similarity values from $\hat{A}$ are used.

Algorithm: Probabilistic Inference

Input: *Initial tensor* $\mathbf{X} \in \mathbb{R}^{S \times O \times P \times D}$, *reconstructed tensor* $\hat{X}$, *the type or fine-grained type classes set* $C_e = \{t_1, t_2, t_3, ...t_n\}$, *and number N rank list for top candidate entities.*

Output: *A list of top entities set* $\nabla\hat{E}_t$ *for each fine-grained type* t_i.

Start:

01: Get the candidate entities set $\hat{E}_t = \{\hat{e}_1, \hat{e}_2, \hat{e}_3, ...\hat{e}_n\}$:

$\hat{E}_t \leftarrow \{\hat{X} \,|(t_i \in \hat{X}$ AND $t_i \in C_e)$ AND $(\hat{X} \leftarrow \hat{x}_{s,p,o,d})\}$

02: Get the similarity of entities set $\hat{A}_{e(t)} = \{\hat{a}_1, \hat{a}_2, \hat{a}_3, ...\hat{a}_n\}$:

$\hat{A}_{e(t)} \leftarrow \{\hat{A} \,|(e(t)_i \in \hat{A}$ AND $e(t)_i \in \hat{X})\}$

03: ***function*** CalculatePosteriorProbability $(\hat{X}, \hat{E}_t, \hat{A}_{e(t)}, N)$

04: ***while*** $e_i \in \hat{E}_t$ ***do***

05: $p_{e_n,t_e} \leftarrow p(t_e|C_e)$ ▷ *from Equation no.* (9) *where,* $P(\hat{E}_e|C_e)$ *is probability of likelihood for* t_e *in* C_e

06: ***end while***

07: ***for*** $n \leftarrow 1toN$ ***do***

08: $\hat{E}_t \leftarrow argmax^N_{e_i \in \hat{E}_t} \{p_{e_n,t_e}\}$

09: ***if*** $n \leftarrow 1toN$ ***do***

10: ***end for***

11: $\nabla\hat{E}_t \leftarrow \hat{E}_t$

14: ***end function***

Further accuracy improvements are therefore achieved by extending the inference model by probabilistic inference with collective multilevel type classification and typed entity similarity.

6 EMPIRICAL EVALUATION

We evaluate our approach on Freebase FB15K dataset [4] and FB15K-237 Knowledge Base Completion Dataset [2] (see Table 1). Freebase [16] [3] is a large collaborative knowledge base consists of a large number of the world facts, such as triplets (*Donald Trump, PresidentOf, United States*) and (*Johnny Depp, profession, actor*). The FB15K (Bordes et al. 2013), is a subset of Freebase which has been commonly used to evaluate various KG completion models [18] [58] [39] [39] [33] [38] [31] [59]. In the FB15K-237 Knowledge Base Completion Dataset, the triples (entity- textual-entity) are derived from 200 million sentences from the ClueWeb12 corpus coupled with Freebase entity. There are around 3.9 million text descriptions corresponding to the relation types in Freebase. The FB15K-237 dataset has been used in [56] [55] [62] for embedding representations for textual relations with Freebase entity mention annotations.

Table 1: Datesets used in the experiments.

Datasets	# Entities	# Relations	# Triples
FB15K	14,951	1,345	486,641
FB15K-327	14,951	2,766,477	3,977,677

For implementation, we use *tensor-toolbox* [10] and *poblano-toolbox* [6] in Matlab. We apply *weighted tensor scheme* in constructing tensor where the tensor size (14951 x 14951 x 2,767,822) with 486541 entries from KG (FB15K); and 4460819 entries from textual dataset (FB15K-237). We then apply *domain-relevance weighted tensor (DrWT)* to construct 4th order tensor with domain entries, where the tensor size (14951 x 14951 x 2,767,822 x 52) . We use CP [21] [30] based *4th-order Tensor Factorization* for the latent factor generation, and use CP-ALS algorithm [24] [34] [25] for computing tensor factorization. Since domain information is not depended in one other dimension of the tensor, we use 4th order tensor factorization instead of using *Coupled Matrix Tensor Factorization (CMTF)* [12] [13]. We also apply *non-negativity constrain* [36] for effectively interpreting factor components from tensor factorization.

Two standard tasks, *Entity Prediction* and *Type Triple Classification* as are proposed to evaluate our approach for fine-grained type inference. The triple classification task e.g. type triple classification for fine-grained type which was first introduced by by Socher et al in [53]. The link prediction task which is standard task for KG completion evaluation e.g entity inference for fine-grained type, was first introduced by Bordes et al. in [18]. Since then these standard tasks have been used to evaluate various KG completion models, such as Rescal [43], TransE [18], TransH [58], TransR [39], CTransR [39], TransD [33], PTransE [38], and KG2E [31].

6.1 Type Triple Classification Task

We evaluate our approach with *Type Triple Classification* aims to judge whether a given triplet (h, r, t) with fine-grained type relation

is correct or not, which is a binary classification task. In FB15K, types of a person entity can be obtained from a triples by both (/people/person/profession) relation and by (/people/profession/people_with_his_profession) relation. Fig. 10 shows some different type triplets distributions with both relations in *Film, Television,* and *Music* domains in FB15K.

We evaluate our approach by type triple classification task on FB15K with six general types and eight fine-grained types in three different domains: *'Film'*, *'TV'* and *'Music'* with the relevant state-of-the-art method RESCAL [43] [44]. As we have mentioned previous section, RESCAL has been achieved good performance among the state-of-the-art approaches for type inference in KG [44]. As SDType [49] and SNCN [41] are not latent factor model, we can not directly compare our model with them.

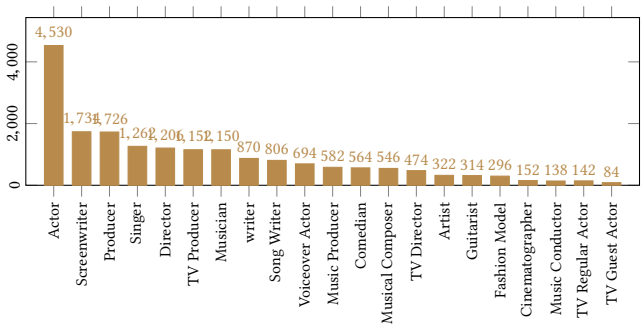


Figure 3: Different Type Triplets Distributions in FB15K.

Evaluation protocol. We follow the same protocol introduced by Socher et al in NTN [53]. Evaluation of classification needs negative labels. As FB15k has not released negative triples in previous works, we construct negative triples by corrupting the corresponding positive (observed) triples. This is followed with the same procedure used in [53]. Dissimilarity between triple and typed triple classifications is that, we only consider to corrupt observed triples that convey types or fine-grained types of entities as shown in Fig. 3. Corrupting observed triples for all relations is not required experiment for evaluation to our approach, as it is out of the scope of this work. The decision rule for classification is simple: for each type triple (h; r; t), if the similarity score (obtained by the models) is below a relation-specific threshold T, then the triple will be classified as positive. Otherwise, it will be classified as negative.

Discussion. From Table 2, we can see that our approach outperforms RESCAL significantly in all the reported top level types and fine-grained types. Some observations are as follows:

(a) From top class types, we can see that both RESCAL and our approach have achieved very high performance results for inferring *actor* type on FB15K. This is because high number of observed triples with actor type object (see Fig. 10), and therefore associated other relationships to actor type object present in FB15K, that positively impact both models for correctly scoring positive and negative triples.

(b) However, from fine-grained class types, we can see that our approach improves results for *Film Actor* and *TV Actor* significantly (11%) and (16%) respectively. Since *Film Actor* and *TV Actor* are in different domains (Film and TV), these domain entries for corresponding triples to the tensor model is defined by DrWT method. This prevents bias scoring positive and negative triples for *Film Actor* to *TV Actor* and vice-versa. Similar effect by DrWT method on results for *TV Director* and *TV Producer* fine-grained type inference.

(c) From fine-grained class types, we can see that our approach our approach significantly outperforms Rescal for *Regular TV Actor* and *Guest TV Actor* where very less number of observed triples with these fine-grained types object present in FB15K (see Fig. 10).

Table 2: Types Inference Comparison with RESCAL

Top Class Types		
	RESCAL	**Our Approach**
Actor	98.0	99.0
Director	90.0	96.0
Producer	89.0	93.0
Musician	78.0	89.0
Singer	74.0	81.0
Writer	70.0	76.0
Fine-grained Types		
	RESCAL	**Our Approach**
TV Director	82.0	91.0
Voiceover Actor	60.0	79.0
Song Writer	79.0	88.0
TV Producer	81.0	90.0
Film Actor	87.0	98.0
TV Actor	77.0	93.0
Regular TV Actor	58.0	78.0
Guest TV Actor	49.0	68.0

6.2 Entity Prediction Task

Entity prediction aims to predict the missing head (h) or tail (t) entity given relation type (r) and other entity, i.e. predicting h given (?; r; t) or predicting t given (h; r; ?) where ? denotes the missing element. In this task, for each position of missing entity, the system emphasizes more on ranking a set of candidate entities from the KG, instead of only giving one best result. In testing phase, for each test triple (h; r; t), we replace the head/tail entity by all entities in the KG, and rank these entities in descending order of similarity scores calculated by the models.

In FB15K, relations can be categorized according to the cardinalities of their head and tail arguments into four classes: one-to-one, one-to-many, many-to-one, and many-to-many. A given relationship is one-to-one if a head can appear with at most one tail, one-to-many if a head can appear with many tails, many-to-one if many heads can appear with the same tail, or many-to-many if multiple heads can appear with multiple tails (see in Table 3).
Evaluation protocol. We follow the same protocol in TransE [18]: For each testing triplet (h; r; t), we replace the tail t by every entity e (14951 in FB15K) in the KG and calculate a similarity score obtained by the model on the corrupted triple (h; r; e). Ranking the scores in ascending order, we then get the rank of the original correct triplet. Aggregated over all the testing triplets, the proportion of

Table 3: Examples of four types of relations in FB15K

Types of Relation	Relations in FB15K
one-to-one	/location/country/capital
one-to-many	/people/profession/people_with_his_profession /film/production_company/films /music/genre/artists /film/director/film
many-to-one	/people/person/place_of_birth /film/film/production_companies /people/person/profession /film/film/directed_by
many-to-many	/film/film/starring/film/performance/actor /film/actor/film/film/performance/film /film/actor/film

ranks not larger than 10 (denoted as Hits@10) metric is reported for evaluation. For this metric a higher Hits@10 is better.

Since the dataset is same, we directly report the experimental results of several baselines from the literature, (see Table no. 4). Instead of taking overall link prediction task results, we take link prediction reported results for predicting heads and prediction tails based on all types of relations: *one-to-one (1-1), one-to-many (1-N), many-to-one (N-1)* and *many-to-many (N-N)*) separately.

Table 4: Evaluation results of Predicting Head (HITS@10) by mapping properties of relations on FB15K

Dataset	**FB15K**			
Task	**Predicting Head**			
Metric	**(HITS@10)**			
Relation Category	**1-1**	**1-N**	**N-1**	**N-N**
Unstructured [17]	34.5	2.5	6.1	6.6
SE [19]	35.6	62.6	17.2	37.5
SME [17]	30.7	69.6	19.9	38.6
TransE [18]	43.7	65.7	18.2	47.2
TransH [58]	66.8	87.6	28.7	64.5
TransR [39]	78.8	89.2	34.1	69.2
CTransR [39]	81.5	89.0	34.7	71.2
PTransE [38]	90.1	92.0	58.7	**86.1**
TransD [33]	86.1	**95.5**	39.8	78.5
KG2E [31]	**92.3**	93.7	**66.0**	69.6
Our Approach	68.0	94.0	30.0	65.0
Task	**Predicting Tail**			
Metric	**(HITS@10)**			
Relation Category	**1-1**	**1-N**	**N-1**	**N-N**
Unstructured [17]	34.3	4.2	1.9	6.6
SE [19]	34.9	14.6	68.3	41.3
SME [17]	28.2	13.1	76.0	41.8
TransE [18]	43.7	19.7	66.7	50.0
TransH [58]	65.5	39.8	83.3	67.2
TransR [39]	79.2	37.4	90.4	72.1
CTransR [39]	80.8	38.6	90.1	73.8
PTransE [38]	90.1	**70.7**	87.5	**88.7**
TransD [33]	85.4	50.6	94.4	81.2
KG2E [31]	**92.6**	67.9	94.4	73.4
Our Approach	50.0	38.6	**96.0**	68.5

Discussion. For the comparison of Hits@10 of different kinds of relations, Table 4 shows the detailed results (entity prediction for heads and tails in triples) by mapping properties of relations on sparse dataset, FB15k. In predicting tail, we can see that our approach outperforms baseline approaches, e.g. TransE, TransH, TransR, CTransR significantly in *many-to-one relation*. In predicting head, we can see that our approach produces good results in *one-to-many relation*. KG2E outperforms other approaches in one-to-one relations, and PTransE has achieved highest results in one-to-many and many-to-many relations in predicting tail on FB15K. Compared with other types of relations, our approach does not outperform previous methods for one-to-one, one-to-many and many-to-many relations in predicting tail. This is because, our approach explicitly considers many-to-one relation for predicting tail, as this type of relationship between entities can only convey types or fine-grained types of entities in KG. For example, in the KGs, such as Freebase, DBPedia types of entities e.g. *Johnny Depp* can form in a KG triple as (Johnny Depp <subject entity> , type <relation> , actor <object entity>). In Freebase in DBPedia, type relation expresses as <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>. Fine-grained type inference is therefore ideologically comparable by many-to-one relation for predicting tail in KG.

7 CONCLUSION

This paper proposes a new approach based on a tensor model for fine-grained type inference in KG. This approach is explicitly modelled for leveraging domain knowledge in KG, and for utilizing the additional large amount of interactions among multiple entities and text descriptions. We further develop probabilistic inference based on collective multilevel type classification and latent similarity of typed entities. Experiment results show that utilizing additional data outside of KG, leveraging domain knowledge with tensor modelling; and developing 'probabilistic inference' by observing data from classification of type classes, and the latent similarity of entities significantly improves performance for inferring fine-grained types of entities in KG. Results in type triple classification and entity prediction tasks show the proposed model outperforms the state-of-the-art approaches for type inference in KG (10% to 19% on different fine-grained types compare to the state-of-the-art approach Rescal), and achieves high results in many-to-one relation in predicting tail for KG completion task.

REFERENCES

[1] "DBPedia" Public Semantic Knowledge Graph. http://wiki.dbpedia.org/. Accessed: August 08, 2017.
[2] "FB15K-237 Knowledge Base Completion Dataset. https://www.microsoft.com/en-us/download/details.aspx?id=52312. Accessed: September 29, 2017.
[3] "Freebase" downloadable version of the data in Freebase. https://developers.google.com/freebase/. Accessed: August 06, 2017.
[4] "Freebase" Knowledge Graph. https://developers.google.com/freebase/. Accessed: September 29, 2017.
[5] "NELL" Never-Ending Language Learning Project. http://rtw.ml.cmu.edu/rtw/index.php. Accessed: August 05, 2017.
[6] "Poblano Toolbox" Poblano - Sandia software - Sandia National Laboratories. https://software.sandia.gov/trac/poblano/. Accessed: August 29, 2018.
[7] "Probase" Knowledge Base. https://www.microsoft.com/en-us/research/project/probase/. Accessed: September 29, 2018.
[8] "RDF" Resource Description Framework. https://www.w3.org/RDF/. Accessed: September 29, 2018.
[9] "SPRQL" Query Language for RDF. https://www.w3.org/TR/rdf-sparql-query/. Accessed: September 29, 2018.
[10] "Tensor Toolbox" MATLAB Tensor Toolbox. https://www.sandia.gov/~tgkolda/TensorToolbox/index-2.6.html. Accessed: August 09, 2018.

[11] "YAGO" semantic knowledge base. http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/. Accessed: August 09, 2017.

[12] Evrim Acar, Tamara G Kolda, and Daniel M Dunlavy. All-at-once optimization for coupled matrix and tensor factorizations. *arXiv preprint arXiv:1105.3422*, 2011.

[13] Evrim Acar, Morten Arendt Rasmussen, Francesco Savorani, Tormod Næs, and Rasmus Bro. Understanding data fusion within the framework of coupled matrix and tensor factorizations. *Chemometrics and Intelligent Laboratory Systems*, 129:53–63, 2013.

[14] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. *The semantic web*, pages 722–735, 2007.

[15] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[16] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM, 2008.

[17] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, 2014.

[18] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

[19] Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. Learning structured embeddings of knowledge bases. In *AAAI*, volume 6, page 6, 2011.

[20] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, page 3, 2010.

[21] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of âĂIJeckart-youngâĂİ decomposition. *Psychometrika*, 35(3):283–319, 1970.

[22] Tony F Chan. Rank revealing qr factorizations. *Linear algebra and its applications*, 88:67–82, 1987.

[23] Liang Chang, Manli Zhu, Tianlong Gu, Chenzhong Bin, Junyan Qian, and Ji Zhang. Knowledge graph embedding by dynamic translation. *IEEE Access*, 5:20898–20907, 2017.

[24] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

[25] Lieven De Lathauwer and Dimitri Nion. Decompositions of a higher-order tensor in block termsâĂŤpart iii: Alternating least squares algorithms. *SIAM journal on Matrix Analysis and Applications*, 30(3):1067–1083, 2008.

[26] Pedro Domingos and Michael Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine learning*, 29(2-3):103–130, 1997.

[27] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610. ACM, 2014.

[28] MS Fabian, K Gjergji, WEIKUM Gerhard, et al. Yago: A core of semantic knowledge unifying wordnet and wikipedia. In *16th International World Wide Web Conference, WWW*, pages 697–706, 2007.

[29] Ming Gu and Stanley C Eisenstat. Efficient algorithms for computing a strong rank-revealing qr factorization. *SIAM Journal on Scientific Computing*, 17(4):848–869, 1996.

[30] Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

[31] Shizhu He, Kang Liu, Guoliang Ji, and Jun Zhao. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 623–632. ACM, 2015.

[32] Jochen Hipp, Ulrich Güntzer, and Gholamreza Nakhaeizadeh. Algorithms for association rule miningâĂŤa general survey and comparison. *ACM sigkdd explorations newsletter*, 2(1):58–64, 2000.

[33] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 687–696, 2015.

[34] Hyunsoo Kim, Haesun Park, and Lars Eldén. Non-negative tensor factorization based on alternating large-scale non-negativity-constrained least squares. In *Bioinformatics and Bioengineering, 2007. BIBE 2007. Proceedings of the 7th IEEE International Conference on*, pages 1147–1151. IEEE, 2007.

[35] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[36] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.

[37] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.

[38] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. *arXiv preprint arXiv:1506.00379*, 2015.

[39] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187, 2015.

[40] André Melo, Heiko Paulheim, and Johanna Völker. Type prediction in rdf knowledge bases using hierarchical multilabel classification. In *Proceedings of the 6th International Conference on Web Intelligence, Mining and Semantics*, page 14. ACM, 2016.

[41] Andre Melo, Johanna Völker, and Heiko Paulheim. Type prediction in noisy rdf knowledge bases using hierarchical multilabel classification with graph and latent features. *International Journal on Artificial Intelligence Tools*, 26(02):1760011, 2017.

[42] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.

[43] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 809–816, 2011.

[44] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st international conference on World Wide Web*, pages 271–280. ACM, 2012.

[45] Andrea Giovanni Nuzzolese, Aldo Gangemi, Valentina Presutti, and Paolo Ciancarini. Type inference through the analysis of wikipedia links. In *LDOW*, 2012.

[46] Heiko Paulheim. Browsing linked open data with auto complete. 2012.

[47] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508, 2017.

[48] Heiko Paulheim and Christian Bizer. Type inference on noisy rdf data. In *International Semantic Web Conference (1)*, volume 8218, pages 510–525, 2013.

[49] Heiko Paulheim and Christian Bizer. Improving the quality of linked data using statistical distributions. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 10(2):63–86, 2014.

[50] Riccardo Porrini, Matteo Palmonari, and Isabel F Cruz. Facet annotation using reference knowledge bases. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 1215–1224. International World Wide Web Conferences Steering Committee, 2018.

[51] Jennifer Sleeman and Tim Finin. Type prediction for efficient coreference resolution in heterogeneous semantic graphs. In *Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on*, pages 78–85. IEEE, 2013.

[52] Jennifer Sleeman, Tim Finin, and Anupam Joshi. Topic modeling for rdf graphs. In *LD4IE@ ISWC*, pages 48–62, 2015.

[53] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934, 2013.

[54] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007.

[55] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, 2015.

[56] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, 2015.

[57] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

[58] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119, 2014.

[59] Han Xiao, Minlie Huang, and Xiaoyan Zhu. Transg: A generative model for knowledge graph embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2316–2325, 2016.

[60] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. Representation learning of knowledge graphs with hierarchical types. In *IJCAI*, pages 2965–2971, 2016.

[61] Harry Zhang. The optimality of naive bayes. *AA*, 1(2):3, 2004.

[62] Jingyuan Zhang, Chun-Ta Lu, Bokai Cao, Yi Chang, and S Yu Philip. Connecting emerging relationships from news via tensor factorization. In *Big Data (Big Data), 2017 IEEE International Conference on*, pages 1223–1232. IEEE, 2017.

[63] Kaja Zupanc and Jesse Davis. Estimating rule quality for knowledge base completion with the relationship between coverage assumption. In *Proceedings of the Web Conference 2018*, pages 1–9, 2018.

Augmenting Knowledge Tracing by Considering Forgetting Behavior

Koki Nagatani
Fuji Xerox Co., Ltd.
Yokohama, Japan
nagatani.koki@fujixerox.co.jp

Qian Zhang
Fuji Xerox Co., Ltd.
Yokohama, Japan
qian.zhang@fujixerox.co.jp

Masahiro Sato
Fuji Xerox Co., Ltd.
Yokohama, Japan
sato.masahiro@fujixerox.co.jp

Yan-Ying Chen
FX Palo Alto Laboratory
Palo Alto, California
yanying@fxpal.com

Francine Chen
FX Palo Alto Laboratory
Palo Alto, California
chen@fxpal.com

Tomoko Ohkuma
Fuji Xerox Co., Ltd.
Yokohama, Japan
ohkuma.tomoko@fujixerox.co.jp

ABSTRACT

Computer-aided education systems are now seeking to provide each student with personalized materials based on a student's individual knowledge. To provide suitable learning materials, tracing each student's knowledge over a period of time is important. However, predicting each student's knowledge is difficult because students tend to forget. The forgetting behavior is mainly because of two reasons: the lag time from the previous interaction, and the number of past trials on a question. Although there are a few studies that consider forgetting while modeling a student's knowledge, some models consider only partial information about forgetting, whereas others consider multiple features about forgetting, ignoring a student's learning sequence. In this paper, we focus on modeling and predicting a student's knowledge by considering their forgetting behavior. We extend the deep knowledge tracing model [17], which is a state-of-the-art sequential model for knowledge tracing, to consider forgetting by incorporating multiple types of information related to forgetting. Experiments on knowledge tracing datasets show that our proposed model improves the predictive performance as compared to baselines. Moreover, we also examine that the combination of multiple types of information that affect the behavior of forgetting results in performance improvement.

CCS CONCEPTS

• **Applied computing** → **Learning management systems**; • **Social and professional topics** → *Student assessment*.

KEYWORDS

knowledge tracing, deep neural network, forgetting behavior

ACM Reference Format:
Koki Nagatani, Qian Zhang, Masahiro Sato, Yan-Ying Chen, Francine Chen, and Tomoko Ohkuma. 2019. Augmenting Knowledge Tracing by Considering Forgetting Behavior. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313565

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313565

1 INTRODUCTION

Computer-aided education (CAE) systems are widespread. Because a huge amount of learning logs can be collected from CAE systems, they can provide personalized materials to students based on their individual knowledge. This helps to reinforce knowledge by repeating materials that are difficult for a student and reducing learning costs by skipping materials that a student already knows.

To recommend suitable learning materials to each student, we should understand each student's knowledge accurately. The knowledge tracing task, which consists of: (1) modeling a student's knowledge through their interactions with the learning system contents and (2) predicting how a student will perform on future interactions, is a well-established and challenging problem in the field of CAE. The more precise the knowledge tracing is, the more satisfactory and suitable the contents provided by the system are [4]. On the contrary, a user study has shown that providing contents and questions to students at an inappropriate level of difficulty causes a decrease in their engagement [1]. Therefore, it is important to model and predict a student's knowledge precisely. This, in turn, is directly related to how efficiently a student can learn and how satisfied a student is with the CAE system.

For knowledge tracing, Bayesian knowledge tracing (BKT) [3] and performance factors analysis (PFA) [14] have been explored widely and applied to intelligent tutoring systems [10, 19, 20]. Recently, as deep learning models are outperforming the conventional models in a range of domains such as pattern recognition and natural language processing, the deep knowledge tracing (DKT) model [17], based on deep learning, can model a student's knowledge more precisely as compared to the conventional models mentioned above. The DKT model represents a student's knowledge state using the hidden variable of a recurrent neural network (RNN), which is a low-dimensional and dense vector. The BKT model represents a student's knowledge state as a binary variable for each skill. A skill is a class of related material, such as by grammar, and is identified with a question tag or ID. Hence, the DKT model can represent a student's knowledge more richly than the BKT model, which leads to improvements in knowledge tracing performance.

Predicting a student's knowledge precisely is a difficult task because students do forget. The recall rate five minutes after the previous learning may be different as compared to that after five days. Besides, the number of times students learn a target skill changes the behavior of forgetting: the more number of times they

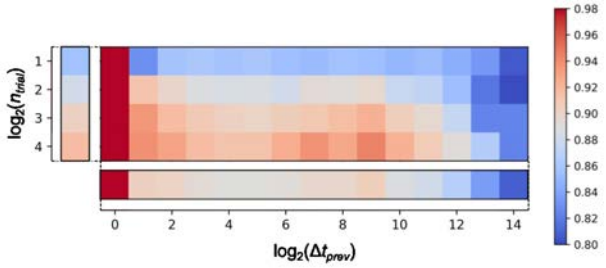


Figure 1: The quantized distribution map of the probability of a student correctly answering a question against the lag time from the previous interaction with the same skill id, Δt_{prev} and the number of past trials, n_{trial}. The distribution averaged over either the lag time or the number of trials are on the left and bottom of the distribution map, respectively. Note that single feature consideration does not capture the complex behavior of forgetting.

learn, the less they are likely to forget. To confirm the complexity of human forgetting, we analyzed the slepemapy.cz dataset, which is real-world learning logs acquired from an online learning platform. Figure 1 shows how the probability of answering correctly depends on both the lag time from the previous interaction with the same skill id and the number of past trials of that skill id. We aggregated interactions that students answered the previous question correctly. The marginal (averaged) distribution over the number of trials and the lag time are shown at the bottom and to the left of the distribution map, respectively. The forgetting behavior for a given lag time differs among the number of trials: as the number of trials increases, the student is less likely to forget. Such complex forgetting behavior cannot be captured if only lag time information is considered. Hence, this comprehensive context related to forgetting should be incorporated to achieve an accurate knowledge modeling.

There are only a few studies that have addressed forgetting in the field of knowledge tracing [11, 15, 18, 21], which either simplify the forgetting behavior or are independent of past sequences. Extensions of BKT only consider a fragment of the information related to forgetting: either the number of past trials [11], or the lag time from the previous interaction with the same skill. [18]. Regression models for knowledge tracing incorporate both the number of trials and the lag time from the previous interaction with the same skill. However, these regression models use features obtained from the interactions with the same skill [15, 21]. Considering a student's whole sequence makes it possible to modify the student's knowledge state on a skill with the student's interactions on other related skills. Otherwise, a model does not acquire any benefits from interactions on other skills.

In this study, we focus on modeling and predicting a student's knowledge considering the forgetting behavior from large amounts of learning logs to support providing adaptive learning materials in CAE systems. We propose an extension of the DKT model which can incorporate the information related to forgetting in the model. We have conducted experiments to examine the predictive performance of our model using two knowledge tracing datasets: Assistments 2012 dataset [7] and slepemapy.cz dataset [13]. In addition, we have investigated the properties of our model by analyzing the output of the model.

The contributions of the paper are summarized as follows.

- We propose a knowledge tracing model that extends the DKT model to consider both a learning sequence and the forgetting behavior by explicitly modeling the forgetting behavior using multiple features.
- We have conducted experiments showing that our proposed model outperforms conventional methods in terms of the predictive performance on the knowledge tracing datasets.
- We have also examined how the combination of multiple types of forgetting information influences the performance.

2 RELATED WORK

2.1 Knowledge tracing

Conventional knowledge tracing models such as BKT [3] and PFA [14] have been explored widely and applied to actual intelligent tutoring systems [10, 19, 20]. The BKT models a student's knowledge using a hidden Markov model (HMM). Each student's knowledge is represented as a set of binary latent variables, and the value of each variable corresponds to whether the student has mastered a skill or not. The PFA is a logistic regression model that predicts the probability of answering correctly based on the number of previous successes and failures. Although the PFA can easily incorporate several features, it basically cannot consider the whole sequence of interactions; instead, it considers the interactions only from the same skill.

As deep learning models are outperforming the conventional models in a range of domains such as pattern recognition and natural language processing, the DKT model [17] has shown that deep learning can model a student's knowledge more precisely as compared to the conventional models mentioned above. The DKT models a student's knowledge by an RNN which is often used for sequential processing. Although the DKT model is a sequence model as well as the BKT model, the DKT model represents a student's knowledge state using the hidden variable of a RNN, which is a low-dimensional and dense vector; the BKT model represents a student's knowledge state as a binary variable for each skill. In doing so, the DKT model can represent a student's knowledge more richly than the BKT model, which leads to an improvement of the predictive accuracy for the student's future performance. Following the DKT model, there are increasing amounts of research on deep learning-based knowledge tracing models [2, 12, 23, 24]. Zhang et al. improved DKT by incorporating additional features into it [24]. Although their approach has similarity to ours in terms of incorporating additional features to the DKT model, they neither focused on modeling forgetting and modeling across time, nor used time-based features such as lag time from the previous interaction.

2.2 Considering forgetting for student's knowledge tracing

There are several research branches that focus on improving knowledge tracing tasks [16]. Considering the forgetting behavior is one of the directions to better understand a student's knowledge. One of the early studies about forgetting [5] revealed that the retention

Table 1: Comparison between our proposed approach and previous work. Sequence indicates the method used to model the sequence of interactions. Time-based forgetting information is the lag time between interactions, and the count-based forgetting information is the number of trials in the past.

Method	Sequence	Forgetting info.	
		Time-based	Count-based
BKT [3]	HMM	-	-
PFA [14]	-	-	-
DKT [17]	RNN	-	-
Qiu et al. [18]	HMM	✓	-
Khajah et al. [11]	HMM	-	✓
Pelánek [15]	-	✓	✓
Settles and Meeder [21]	-	✓	✓
Ours	RNN	✓	✓

rate decreases exponentially as time passes by, and also, increasing the number of repetitions helps a user avoid forgetting. Without considering forgetting behavior in a knowledge tracing model, a student's knowledge state cannot be changed as time passes by. This assumption is not reasonable in reality.

The original BKT model does not consider forgetting behavior. Qiu et al. [18] extended the BKT model to consider forgetting behavior for the first time. A new day flag was added to BKT to model forgetting one day after the previous interaction. However, this system could not model forgetting that occurred on a much shorter time scale. Khajah et al. then extended the BKT model to estimate the probability of forgetting by applying the number of trials in the past [11]. However, the lag time between the current and previous interactions was not considered.

For the PFA and the related regression models, there are two studies that focused on incorporating forgetting behavior into these models. Pelánek incorporated a time factor into the memory activation, which determines the probability of answering correctly [15]. In this research, applying a staircase function, which boosts the memory activation when the lag time is in a certain bound, resulted in the best performance. On the other hand, Settles and Meeder proposed a half-life regression model which extends the PFA model with the essence of Ebbinghaus's forgetting curve [21]. Although these models incorporated information related to forgetting, the problem still remains. These regression models consider only the information about the skill and neglect the interaction sequence of other skills.

The difference between our proposed approach and the previous works is that we extended DKT, which is a sequential knowledge model, to consider a student's whole sequence of interactions, and incorporate multiple types of information to represent the complex forgetting behavior. Table 1 shows the comparison between our proposed approach and previous works.

3 PRELIMINARIES

In this section, we describe the knowledge tracing task and briefly explain the DKT model, which is the base component of our model.

3.1 Knowledge Tracing Task

The knowledge tracing task aims to model a student's knowledge through interactions with questions and to predict how a student will perform on the future interactions. The task can be formulated as a supervised learning problem: given a student's past interactions $\mathbf{x}_0, \cdots, \mathbf{x}_t$, the student's performance is predicted in the next interaction. An interaction $\mathbf{x}_t = (q_t, a_t)$ is defined as a tuple containing the skill id q_t of a question that a student attempts at time step t, and the label a_t which represents whether the student answers correctly or not. We define the number of skills as Q. In this study, the skill id q_t is identified by either a question tag or an ID, and a_t is a binary variable (one represents correct and zero represents incorrect answers). The values of a_{t+1} are predicted for the given values of q_{t+1}.

3.2 Deep Knowledge Tracing

The deep knowledge tracing model is the first deep neural network-based knowledge tracing model [17]. The structure of the DKT model is shown in Fig. 2 (a). DKT models a student's knowledge transition by using various RNNs such as Elman RNN [6] and long short term memory (LSTM) [9]. The student knowledge state is represented by a hidden state vector of RNN $\mathbf{h}_t \in \mathbb{R}^k$, where k is the number of hidden state dimensions. The DKT model can be separated into two processes: modeling a student's knowledge and predicting a student's performance.

In the process of modeling a student's knowledge, for a given input $\mathbf{x}_t = (q_t, a_t)$, the model updates a student's knowledge state $\mathbf{h}_t$ at each time step t. The input $\mathbf{x}_t$ is a one-hot vector, which is a Cartesian product of q_t and a_t. The input vector $\mathbf{x}_t$ is then transformed into a low-dimensional and dense real-valued vector $\mathbf{v}_t$. In [17], a random vector $\mathbf{n}_{q,a} \sim N(0, \mathbf{I})$ is assigned to each input tuple as v_t. With the embedding vector $\mathbf{v}_t$ and the previous student knowledge state vector $\mathbf{h}_{t-1}$, the student knowledge vector $\mathbf{h}_t$ is updated as:

$$\mathbf{h}_t = \phi(\mathbf{v}_t, \mathbf{h}_{t-1}),$$

where ϕ is the cell module of RNN.

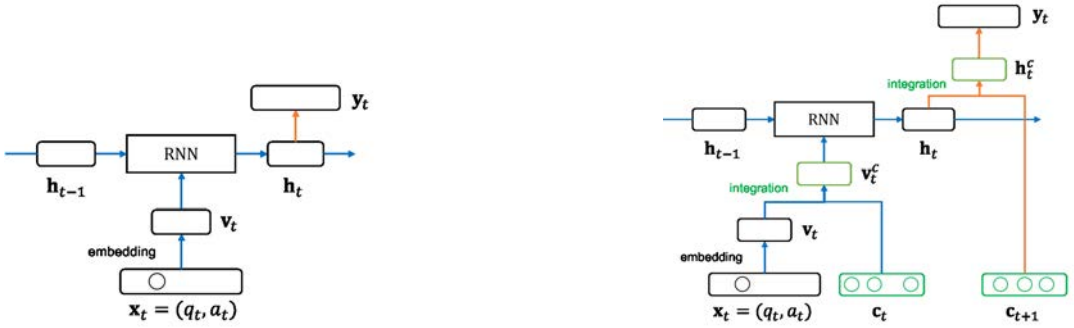
In the process of predicting a student's performance, the probabilities of answering correctly for all skills $\mathbf{y}_t \in \mathbb{R}^Q$ based on the updated student's knowledge state $\mathbf{h}_t$ is calculated as:

$$\mathbf{y}_t = \sigma(\mathbf{b}^{\text{out}} + \mathbf{W}^{\text{out}}\mathbf{h}_t),$$

where $\sigma(\cdot)$ is the sigmoid function, $\mathbf{W}^{\text{out}} \in \mathbb{R}^{Q \times k}$ is the weight matrix and $\mathbf{b}^{\text{out}} \in \mathbb{R}^Q$ is the bias vector of output.

4 PROPOSED APPROACH

We have utilized the DKT model as our base model because it is a state-of-the-art model in the knowledge tracing task, and a deep neural network that can easily incorporate multiple input sources and capture the nonlinear dynamics among them. By extending DKT so that it can consider forgetting, the model can adapt the student performance to the student's forgetting.



(a) Architecture for deep knowledge tracing.

(b) Architecture for proposed model.

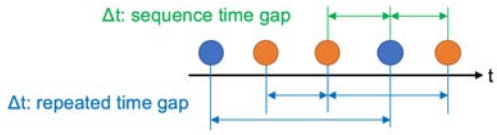
Figure 2: Architectures for (a) deep knowledge tracing and (b) proposed model. The blue arrows describe a process of modeling a student's knowledge. The orange arrows describe a process of predicting a student's performance. Our proposed model incorporates the additional information related to forgetting $\mathbf{c}_t$, $\mathbf{c}_{t+1}$ and an integration component in both processes, which are colored by green. $\mathbf{c}_t$, $\mathbf{c}_{t+1}$ are multi-hot vectors which quantize the information related to forgetting: repeated time gap, sequence time gap, and past trial counts.

4.1 Information related to forgetting

Earlier research has revealed that the retention rate differs depending on the combination of two aspects: the lag time from the previous interaction and the number of times a student learns the learning material. The lag time can be calculated for the repetition of same skill ids or for sequence of any skill ids. Hence, in this study, we have considered the following three features:

- repeated time gap: the lag time between an interaction and the previous interaction with the same skill id,
- sequence time gap: the lag time between an interaction and the previous interaction in the sequence; the skill id of an interaction does not matter, and
- past trial counts: the number of times a student answers questions with the same skill id.

Figure 3 illustrates these features. As for time gap, the repeated time gap has been commonly used in the previous researches [15, 21]. The reason we have used the sequence time gap is if both the skill of the question that student attempts and the skill of the previous question are related to each other, the lag time between these interactions may affect the performance of the question. Incorporating the sequence time gap into the model may capture this effect. The two time gap features are used by the minute and all three features are discretized at $\log_2$ scale.



Time step	t_1	t_2	t_3	t_4	t_5
Repeated time gap	-	-	Δt_{32}	Δt_{41}	Δt_{53}
Sequence time gap	-	Δt_{21}	Δt_{32}	Δt_{43}	Δt_{54}
Past exercise counts	0	0	1	1	2

Figure 3: The additional information related to forgetting from a student's sequence of interactions. Each point corresponds to an interaction and the same color represents the same skill. In the table, the time gap $\Delta_{ij} = t_i - t_j$.

4.2 Model

We extended DKT by incorporating information related to forgetting into two spaces of the model: the input and the output space of the RNN. The overall model architecture is shown in Fig. 2 (b).

To model a student's knowledge process, we have used a trainable embedding matrix $\mathbf{A}$ to calculate the embedded vector for $\mathbf{v}_t$ for the interaction vector $\mathbf{x}_t$ instead of assigning the random values. In addition, we have added the information related to forgetting, $\mathbf{c}_t$, discussed in Sect. 4.1, as the input as well as the interaction at time step t. $\mathbf{c}_t$ is a multi-hot vector because the three features in the information are represented as one-hot vectors and then concatenated. Before passing into the RNN module, the embedding vector $\mathbf{v}_t$ and the additional information vector $\mathbf{c}_t$ are integrated with each other:

$$\mathbf{v}_t^c = \theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t),$$

where θ^{in} is the input integration function that incorporates additional information into the student's knowledge state. We describe this integration function in Sect. 4.3. Next, the student's knowledge state vector $\mathbf{h}_t$ is updated with the integrated input $\mathbf{v}_t^c$ and the previous student's knowledge state $\mathbf{h}_{t-1}$:

$$\mathbf{h}_t = \phi(\mathbf{v}_t^c, \mathbf{h}_{t-1}).$$

Thereby, a student's knowledge state is updated considering both the response of a student for the skill and the information related to forgetting. If a student answers correctly with a long time gap, the model updates the skill in the direction of acquiring more mastery.

Similarly in the process of predicting a student's performance, we have integrated the additional information at the next time step $\mathbf{c}_{t+1}$ with the updated student's knowledge state vector $\mathbf{h}_t$:

$$\mathbf{h}_t^c = \theta^{\text{out}}(\mathbf{h}_t, \mathbf{c}_{t+1}),$$

where θ^{out} is the output integration function that brings additional information into the prediction. Thereby, the model can predict

Table 2: Statistics of the data.

Dataset	#interactions	#users	#items
ASSISTments 2012	5,818,868	45,675	266
slepemapy.cz	10,087,305	87,952	1,458

a student's performance considering forgetting: if the student attempts a question with a short time lag, the model revised the probability of answering correctly upward. This integration function is described in detail in Sect. 4.3. Subsequently, the model predicts the probability of answering correctly for all skills $\mathbf{y}_t \in \mathbb{R}^Q$ based on the updated student's knowledge state $\mathbf{h}_t^c$:

$$\mathbf{y}_t = \sigma(\mathbf{b}^{\text{out}} + \mathbf{W}^{\text{out}}\mathbf{h}_t^c).$$

4.3 Integration Methods

In both processes of modeling a student's knowledge and predicting a student's performance, we have incorporated the integration functions, θ^{in} and θ^{out}, into our model to consider the forgetting behavior. We have explored the following four integration methods:

- concatenation: $\theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t) = [\mathbf{v}_t; \mathbf{c}_t]$,
- multiplication: $\theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t) = \mathbf{v}_t \odot \mathbf{C}\mathbf{c}_t$,
- concatenation and multiplication: $\theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t) = [\mathbf{v}_t \odot \mathbf{C}\mathbf{c}_t; \mathbf{c}_t]$, and
- bi-interaction: $\theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t) = \sum_i \sum_{j \neq i} \mathbf{z}_i \odot \mathbf{z}_j, \mathbf{z}_i \in \{\mathbf{v}_t, \mathbf{C}_i c_t^i \mid c_t^i \neq 0\}$,

where $\mathbf{C}$ is the trainable transformation matrix, and $\odot$ denotes the element-wise multiplication. For the output integration function θ^{out}, we have used the same integration methods but applied to the student's knowledge vector $\mathbf{h}_t$ instead of $\mathbf{v}_t$. We have shared the transformation matrix C for both the input integration function θ^{in} and the output integration function θ^{out}. Concatenation stacks the interaction vector with the feature vector. Hence, this integration does not change the interaction vector itself. On the other hand, multiplication modifies an interaction vector by the contextual information. Concatenation and multiplication are common ways to integrate vectors. The first three integration methods have been used in [22]. Bi-interaction, which has been proposed in [8], encodes the second-order interactions between the interaction vector and context information vector, and between context information vectors.

4.4 Training

The learning parameters of our proposed model are the embedding matrix $\mathbf{A}$ for the interaction vector $\mathbf{x}_t$, the weights in the RNN module, the weight matrix $\mathbf{W}^{\text{out}}$ and the bias $\mathbf{b}^{\text{out}}$ for the prediction, and the transformation matrix $\mathbf{C}$ for additional information $\mathbf{c}_t$ if we use multiplication or bi-interaction integration. These parameters are jointly learned by minimizing a standard cross entropy loss between the predicted probability of correctly answering the next question for the skill id q_{t+1} and the true label a_{t+1}:

$$\mathcal{L} = -\sum_t (a_{t+1}\log(\mathbf{y}_t^{\mathrm{T}}\delta(\mathrm{q}_{t+1})) + (1 - a_{t+1})\log(1 - \mathbf{y}_t^{\mathrm{T}}\delta(\mathrm{q}_{t+1}))),$$

where $\delta(q_{t+1})$ is the one-hot encoding for which skill id is answered in the next time step $t + 1$.

5 EXPERIMENTS

In this section, we present evaluation results for our model. Firstly, we evaluated the predictive performance by comparing our proposed model with other baselines on two real-world public datasets collected from online learning platforms. Next, we conducted comparative experiments to examine the contribution of features related to forgetting in our proposed model for further model exploration.

5.1 Datasets

We used the two datasets: ASSISTments 2012[1] [7] and slepemapy.cz[2] [13]. The statistics of the two datasets are shown in Table 2.

ASSISTments 2012: This dataset is gathered from the ASSISTments, which is an online tutor system that simultaneously teaches and assesses students in mathematics. For the dataset, we defined *skill_id* as the identifier of a skill id. We also removed users with only one interaction. After preprocessing, the dataset includes 5,818,868 interactions of 45,675 users and 266 skills.

slepemapy.cz: This dataset is from an online system used for practicing geography. We defined *place_asked* as the identifier of a skill id. After removing the users with only one interaction, the dataset includes 10,087,305 interactions of 87,952 users and 1,458 skills.

5.2 Experimental setup

We evaluated the knowledge tracing models using five-fold cross validation in which each dataset was split based on students. We extracted 10% of the students from the training set for a validation set that was used to tune hyperparameters and perform early stopping. On each dataset we used area under the curve (AUC) as an evaluation metric, which ranges from 0 (worst) to 1 (best). We reported the average test AUC.

We used an LSTM for both DKT and our model. To examine the effectiveness of incorporating the forgetting features, we used the same embedding technique for the DKT model, which is described in Section 4.2. The performance of DKT with trainable embedding was almost the same as the original DKT model using random embedding (an AUC of 0.7235 for the DKT with trainable embedding was comparable with an AUC of 0.7233 for the original DKT on ASSISTments dataset). We used the DKT with trainable embedding as our baseline. Both the size of the embedding vector and the dimension of the LSTM hidden state were set to 100. We used the Adam algorithm to optimize the models. The mini batch size was set to 100 for ASSISTments 2012 dataset and 30 for slepemapy.cz dataset. We explored the optimal hyperparameters in terms of weight decay in a validation set.

5.3 Performance Prediction

We compared our proposed models with different integration methods against the DKT and the HLR model, which is the regression model considering forgetting. Table 3 shows the results on the ASSISTments and slepemapy.cz datasets. Because the HLR model can predict only the interactions of questions which a student has attempted in the past, we also evaluated on the subset. In our experiments, the regression-based HLR model performed worse than

[1] https://sites.google.com/site/assistmentsdata/home/2012-13-school-data-with-affect

[2] https://www.fi.muni.cz/adaptivelearning/?a=data

Table 3: Test AUC results for two datasets. We conducted five-fold cross-validation and calculated the average test AUC. *all* is all interactions except the first interaction of each student in a test set. *sub* is the subset of the interactions with questions which a student has attempted even once in the past. The standard deviation values are 0.0014 and 0.0034 on ASSISTments and slepemapy.cz datasets, respectively.

Model	ASSISTments		slepemapy.cz	
	all	sub	all	sub
HLR	-	0.5846	-	0.6317
DKT	0.7235	0.7213	0.7871	0.7929
DKT + forgetting (Ours)				
concat	0.7283	0.7257	0.7999	0.8061
multi	0.7301	0.7280	0.8028	0.8093
concat + multi	**0.7309**	**0.7287**	**0.8046**	**0.8115**
bi-interaction	0.7289	0.7267	0.8037	0.8103

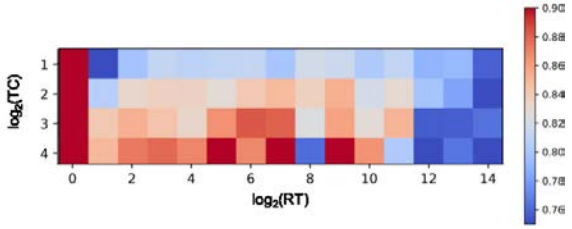


Figure 4: The quantized distribution map of the prediction of our model against the repeated time gap (RT) and the past trial counts (TC) on slepemapy.cz dataset. This figure is similar to Fig. 1, which indicates that our model can reproduce the complex forgetting behaviors.

any of the deep-learning based models. DKT achieved an average test AUC of 0.7235 on the ASSISTments dataset, and an AUC of 0.7871 on the slepemapy.cz dataset. Compared with DKT, our proposed model, which considers the information related to forgetting, performed better ($p < 0.01$ for both datasets). For reference, the Pelánik model [15], which incorporated a time factor into the PFA model, performed with an AUC of 0.7614 on the slepemapy.cz dataset. Although their experimental condition differs from ours, since they filtered the data to remove items with low counts, our model outperforms their model. We also investigated how the prediction of our model differs depending on both the repeated time gap and the number of past trials of the skill id, which is shown in Fig. 4. As in Fig. 1, we aggregated the interactions that students answered the previous question correctly. Our model reproduced the complex forgetting behavior shown in Fig. 1: as the number of trials increases, the student is less likely to forget.

Among the different integration methods, the combination of concatenation and multiplication (hybrid) achieved the best AUC on both the ASSISTments and slepemapy.cz datasets, with values of of 0.7309 and 0.8046, respectively. The multiplication integration achieved the second-best AUC score on ASSISTments dataset while the bi-interaction integration achieved the second-best AUC score on slepemapy.cz. The better result of the multiplication integration method than the concatenate integration method indicates that modifying an interaction vector using the information related to forgetting is better.

Table 4: Comparative analysis of the features related to forgetting. We used one or more features and compared the AUC performance. We used the hybrid integration of concatenation and multiplication. In the table, RT is repeated time gap, ST is sequence time gap, and TC is past trial counts.

Model	RT	ST	TC	ASSISTments	slepemapy.cz
DKT				0.7235	0.7871
DKT + forgetting	✓			0.7288	0.7972
+ one feature		✓		0.7284	0.7940
			✓	0.7258	0.7976
+ two features	✓	✓		0.7292	0.7992
	✓		✓	0.7304	0.7989
		✓	✓	0.7301	0.7993
+ three features (Ours)	✓	✓	✓	0.7309	0.8046

5.4 Comparative Study: Forgetting features

We investigated the effectiveness of incorporating the combination of features related to forgetting in the DKT model. Specifically, we used one or multiple forgetting features for modeling and observed how that affects performance.

Table 4 shows the results of the experiments. During performance comparison, while changing the number of features, we obtained the same trend on both ASSISTments and slepemapy.cz datasets: the more features used for modeling, the better the predictive performance. Adding two features resulted in a better performance than adding one feature with statistical significance ($p = 0.012$). Adding three features had a better performance than adding two features, though not statistically significant. These results support that considering multiple aspects about forgetting is important for modeling a student's knowledge.

The effectiveness of incorporating the combination of the features was different in two datasets. For the ASSISTments dataset, the combination of repeated time gap and trial count (RT+TC) was superior to other combinations. For the slepemapy.cz dataset, the combination of sequence time gap and trial count (ST+TC) seemed slightly more effective. As seen from the result of one feature in Table 4, these combinations do not comprise the top two features that had a good performance. This indicates that these features correlate with each other.

6 CONCLUSION

In this paper, we focused on considering forgetting behavior in knowledge tracing. We extended the DKT model so that the information related to forgetting can be incorporated into the model. The proposed solution achieved better predictive performance compared with the DKT model on two real-world datasets for knowledge tracing. We also revealed that the combination of several types of information which affect the behavior of forgetting resulted in improved performance.

In future work, we intend to extend this study to model forgetting behavior for each student since the tendency of forgetting differs among students. We also consider to use this model for the knowledge tracing with side information.

REFERENCES

[1] Panagiotis Adamopoulos. 2013. What Makes a Great MOOC? An Interdisciplinary Analysis of Student Retention in Online Courses. In *34th International Conference on Information Systems*. Association for Information Systems, United States. The most heavily-cited paper from the ICIS 2013 proceedings (as of August 15th, 2016).

[2] Haiqin Cheung, Lap Pongand Yang. 2017. Heterogeneous Features Integration in Deep Knowledge Tracing. In *Neural Information Processing*. Springer International Publishing, 653–662.

[3] Albert T. Corbett and John R. Anderson. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 4 (01 Dec 1994), 253–278. https://doi.org/10.1007/BF01099821

[4] Yossi Ben David, Avi Segal, and Ya'akov (Kobi) Gal. 2016. Sequencing Educational Content in Classrooms Using Bayesian Knowledge Tracing. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge (LAK '16)*. ACM, New York, NY, USA, 354–363. https://doi.org/10.1145/2883851.2883885

[5] Hermann Ebbinghaus. 1885. *Memory: A Contribution to Experimental Psychology*. Teachers Collage, Columbia University, New York.

[6] Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science* 14, 2 (1990), 179 – 211. https://doi.org/10.1016/0364-0213(90)90002-E

[7] Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction* 19, 3 (01 Aug 2009), 243–266. https://doi.org/10.1007/s11257-009-9063-7

[8] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. ACM, New York, NY, USA, 355–364. https://doi.org/10.1145/3077136.3080777

[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[10] Jussi Kasurinen and Uolevi Nikula. 2009. Estimating Programming Knowledge with Bayesian Knowledge Tracing. In *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE '09)*. ACM, New York, NY, USA, 313–317. https://doi.org/10.1145/1562877.1562972

[11] Mohammad Khajah, Robert Lindsey, and Michael Mozer. 2016. How Deep is Knowledge Tracing?. In *Proceedings of the ninth International Conference on Educational Data Mining (EDM'16)*. 94–101.

[12] Byung-Hak Kim, Ethan Vizitei, and Varun Ganapathi. 2018. GritNet: Student Performance Prediction with Deep Learning. *CoRR* abs/1804.07405 (2018).

[13] Jan Papoušek, Radek Pelánek, and Vit Stanislav. 2016. Adaptive Geography Practice Data Set. *Journal of Learning Analytics* 3, 2 (2016), 317–321. https://doi.org/10.18608/jla.2016.32.17

[14] Philip I. Pavlik, Hao Cen, and Kenneth R. Koedinger. 2009. Performance Factors Analysis –A New Alternative to Knowledge Tracing. In *Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 531–538. http://dl.acm.org/citation.cfm?id=1659450.1659529

[15] Radek Pelánek. 2015. Modeling Students' Memory for Application in Adaptive Educational Systems. In *Proceedings of eighth International Conference on Educational Data Mining*. 480–483. https://files.eric.ed.gov/fulltext/ED560907.pdf

[16] Radek Pelánek. 2017. Bayesian Knowledge Tracing, Logistic Models, and Beyond: An Overview of Learner Modeling Techniques. *User Modeling and User-Adapted Interaction* 27, 3-5 (Dec. 2017), 313–350. https://doi.org/10.1007/s11257-017-9193-2

[17] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. 2015. Deep Knowledge Tracing. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 505–513. http://papers.nips.cc/paper/5654-deep-knowledge-tracing.pdf

[18] Yumeng Qiu, Yingmei Qi, Hanyuan Lu, Zachary A. Pardos, and Neil T. Heffernan. 2011. Does Time Matter? Modeling the Effect of Time with Bayesian Knowledge Tracing. In *Proceedings of the eighth International Conference on Educational Data Mining (EDM '11)*. 139–148.

[19] Jonathan P. Rowe and James C. Lester. 2010. Modeling User Knowledge with Dynamic Bayesian Networks in Interactive Narrative Environments. In *Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE'10)*. AAAI Press, 57–62. http://dl.acm.org/citation.cfm?id=3014666.3014678

[20] Thorsten Schodde, Kirsten Bergmann, and Stefan Kopp. 2017. Adaptive Robot Language Tutoring Based on Bayesian Knowledge Tracing and Predictive Decision-Making. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction (HRI '17)*. ACM, New York, NY, USA, 128–136. https://doi.org/10.1145/2909824.3020222

[21] Burr Settles and Brendan Meeder. 2016. A Trainable Spaced Repetition Model for Language Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1848–1858. https://doi.org/10.18653/v1/P16-1174

[22] Elena Smirnova and Flavian Vasile. 2017. Contextual Sequence Modeling for Recommendation with Recurrent Neural Networks. In *Proceedings of the 2Nd Workshop on Deep Learning for Recommender Systems (DLRS 2017)*. ACM, New York, NY, USA, 2–9. https://doi.org/10.1145/3125486.3125488

[23] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris H. Q. Ding, Si Wei, and Guoping Hu. 2018. Exercise-Enhanced Sequential Modeling for Student Performance Prediction. In *AAAI*.

[24] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. Dynamic Key-Value Memory Networks for Knowledge Tracing. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 765–774. https://doi.org/10.1145/3038912.3052580

Think Outside the Dataset: Finding Fraudulent Reviews using Cross-Dataset Analysis

Shirin Nilizadeh
University of Texas, at Arlington
shirin.nilizadeh@uta.edu

Hojjat Aghakhani
University of California Santa Barbara
hojjat@ucsb.edu

Eric Gustafson
University of California Santa Barbara
edg@ucsb.edu

Christopher Kruegel
University of California Santa Barbara
chris@ucsb.edu

Giovanni Vigna
University of California Santa Barbara
vigna@ucsb.edu

ABSTRACT

While online review services provide a two-way conversation between brands and consumers, malicious actors, including misbehaving businesses, have an equal opportunity to distort the reviews for their own gains. We propose OneReview, a method for locating fraudulent reviews, correlating data from multiple crowd-sourced review sites. Our approach utilizes *Change Point Analysis* to locate points at which a business' reputation shifts. Inconsistent trends in reviews of the same businesses across multiple websites are used to identify *suspicious* reviews. We then extract an extensive set of textual and contextual features from these suspicious reviews and employ supervised machine learning to detect *fraudulent* reviews.

We evaluated OneReview on about 805K and 462K reviews from Yelp and TripAdvisor, respectively to identify fraud on Yelp. Supervised machine learning yields excellent results, with 97% accuracy. We applied the created model on suspicious reviews and detected about 62K fraudulent reviews (about 8% of all the Yelp reviews). We further analyzed the detected fraudulent reviews and their authors, and located several spam campaigns in the wild, including campaigns against specific businesses, as well as campaigns consisting of several hundreds of socially-networked untrustworthy accounts.

CCS CONCEPTS

• **Information systems → Trust**; **Reputation systems**.

KEYWORDS

Review Websites; Fraudulent Reviews and Campaigns; Cross-Dataset Analysis; Change-Point Analysis;

ACM Reference Format:
Shirin Nilizadeh, Hojjat Aghakhani, Eric Gustafson, Christopher Kruegel, and Giovanni Vigna. 2019. Think Outside the Dataset: Finding Fraudulent Reviews using Cross-Dataset Analysis. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313647

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313647

1 INTRODUCTION

Many crowd-sourced review platforms, such as Yelp, TripAdvisor, Google and Foursquare, provide a shared space for people to write reviews and rate local businesses. With the substantial impact of businesses' online ratings on their selling [4], many businesses add themselves to multiple websites to more easily be discovered. Some might also engage in reputation management, which could range from rewarding their customers for a favorable review, or a complex review campaign, where armies of accounts post reviews to influence a business' average review score.

Several previous works have attempted to address this problem. Most of them use supervised machine learning, and only focus on textual and stylometry features [1, 16, 19, 37]. Their obtained ground truth data is also not large and comprehensive [19, 25, 27, 37, 39, 51]. Finally, these works assume a limited attacker model, e.g., an attacker's activity is assumed to be found near sudden shifts in the data [39], or they only try to detect positive campaigns.

In this paper, we propose OneReview, a system for finding fraudulent content on a crowd-sourced review site, leveraging correlations with other independent review sites, and the use of textual and contextual features. Our system leverages the intuition that an attacker may not be able to exert the same influence over a business' reputation on several websites, due to increased cost, with reviews costing between $10 and $25 each [5, 6, 45], and must be customized for each site. Even in the case of machine-generated reviews explored in [47, 49], the text generated by these systems may be customized, but the supporting metadata is not. Even when reviews can be purchased as a service [41, 44], these services charge more to target additional review sites.

OneReview focuses on isolating anomalous changes in a business' reputation across multiple review sites, to locate malicious activity without relying on specific patterns. Our intuition is that a business's reputation should not be very different in multiple review sites; *e.g.,* if a restaurant changes its chef or manager, then the impact of these changes should appear on reviews across all the websites. OneReview utilizes *Change Point Analysis* method on the reviews of every business independently on every website, and then uses our proposed *Change Point Analyzer* to evaluate change-points, detect those that do not match across the websites, and identify them as suspicious. Then, it uses supervised machine learning, utilizing a combination of textual and metadata features to locate fraudulent reviews within the set of suspicious reviews.

We evaluated our approach, using data from two review websites, namely Yelp and TripAdvisor, to find fraudulent activity on Yelp. We

obtained Yelp reviews through the Yelp Data Challenge [50], and used our Change Point Analyzer to correlate this with data crawled from TripAdvisor. Since realistic, varied ground truth data is not currently available, we used a combination of our change point analysis and crowd-labeling to create a set of 5,655 labeled reviews. We used k-cross validation (k=5) on our ground truth and obtained 97% (+/- 0.01) accuracy, 91% (+/- 0.03) precision and 90% (+/- 0.06) recall. Then the model was used on the suspicious reviews, which classified 61,983 reviews, about 8% of all reviews, as fraudulent.

We further detected fraudulent campaigns that are actively initiated by, or targeted toward specific businesses. We identified 3,980 businesses with fraudulent reviews, as well as, 14,910 suspected spam accounts, where at least 40% of their reviews are classified as fraudulent. We also used community detection algorithms to locate several large astroturfing campaigns. These results show the effectiveness of OneReview in detecting fraudulent campaigns.

2 RELATED WORK

Text and Metadata Features. Previous works have explored a multitude of combinations of machine learning techniques and features to help locating fraudulent reviews. Most works rely only on textual [16, 25, 37, 39, 51], and stylometry [16] features. Jindal and Liu [19] first used meta-data features to detect fraudulent reviews on Amazon. OneReview uses some of the suggested features if applicable. Considerable work tried to identify the spam accounts, using regression models [26], heterogeneous graphs [48], unsupervised anomaly detection [22, 31, 32, 46], and behavioral models [32, 33]. OneReview uses user-related metadata features, but does not make any such assumptions about the users in question.

Using Temporal Data for Fraud Detection. Some works [15, 24, 27, 39] take advantage of temporal and spatial information in their detection algorithms. Li et al. [24] observed some correlations between the time of writing reviews on Dianping, and the location of their authors, with the spam reviews. Others use bursts in the number of reviews to locate suspicious patterns [15, 39, 39]. This approach has two limitations compared to OneReview. First, OneReview can distinguish if reviews that have significant impact on the overall score of the business are actually suspicious, by comparing the pattern on the other website, eliminating potential false positives. Second, our threat model considers both the overall review score, as well as the more recent effects of reviews.

Ground Truth Creation. One main obstacle in detecting fake reviews is the absence of ground truth. Some early work leverages duplicate reviews as the source of fraudulent labeled data [19]. Others explore asking humans to write deceptive reviews [25, 37, 51]. However, the ground truth dataset that has been used in these methods might not reflect the dynamics of fraudulent reviews in a commercial website [34]. Rahman et. al [39] leveraged some forums where Elite Yelp users reveal and initiate the discussion on fraudulent Yelp reviews. Unfortunately, these forums are no longer available. In contrast, our dataset (see Section 6) contains data that is user-labeled, not user-generated, and additionally contains reviews chosen based on social graph information, and duplicate reviews.

3 THREAT MODEL

We assume that the adversary wishes to modify the reputation of a business, either positively or negatively, through any means, including both the overall reputation score and its recent reviews. We assume the adversary can utilize any functionality or behavior available to a normal, registered user of the service, including creating accounts, posting reviews, adding social connections, and so on. We also assume the adversary may compromise the accounts of legitimate users, or hire some users to post fraudulent reviews [6, 30, 41, 44].

An active adversary may know of the deployment of a system such as OneReview, and actively attempt to avoid detection by cross-posting the same ratings on all the crowd-sourced review websites over similar time spans. However, there is a trade-off between costs and benefits. Since the adversary needs to avoid detection by each of these review websites, as well as detection by OneReview itself, then it needs to employ complex and expensive techniques equally on all the websites which makes it unprofitable.

4 DATA

In this paper we prototype OneReview with data from two widely popular crowd-review sites, Yelp and TripAdvisor. Both sites allow users to search for businesses in their area; the page for each business offers users an overview of its basic information, reputation, and a list of reviews. Both allow users to submit reviews with "star" ratings. Users can provide feedback about the reviews themselves, such as Yelp's "funny" or "useful" ratings. Both also offer a ranking system for users (Yelp's "Elite", or TripAdvisor's "TripCollective").

Yelp Dataset. We obtained Yelp reviews from the dataset released for the 9th Yelp Data Challenge [50]. For more easily demonstrating our system, we only examine the reviews on restaurants that are located in Pittsburgh, Charlotte, Urbana-Champaign, Phoenix, Las Vegas, Madison, and Cleveland. Our Yelp dataset includes data for 1.4M reviews for about 16K restaurants and 469K users. It also includes metadata about businesses, reviews, and users.

TripAdvisor Dataset. In March 2017, we crawled TripAdvisor, and extracted public reviews and reviewer information about restaurants in the same seven US cities we selected in the Yelp dataset, resulting in about 656K reviews, submitted by 305K users, for 10K businesses. As with Yelp, data from TripAdvisor includes various metadata about each review, reviewer, and business.

Comparing the overall star ratings, we found that businesses on TripAdvisor obtain higher average star ratings (4.11) than those on Yelp (3.76). *The difference between the overall star ratings suggests that when using two sources of data, it might not be helpful (and it can be even misleading) to directly compare the ratings. However, it can be helpful to compare the trends in this data.*

5 ONEREVIEW SYSTEM DESIGN

OneReview is a system to detect *fraudulent reviews* and campaigns in crowd-sourced review websites through comparing trends with other sites, and machine learning. Figure 1 illustrates an overview of OneReview, including: (1) obtaining data, (2) matching businesses, (3) identifying inconsistent and suspicious change points in *star* trends, (4) extracting textual and contextual features, (5) employing

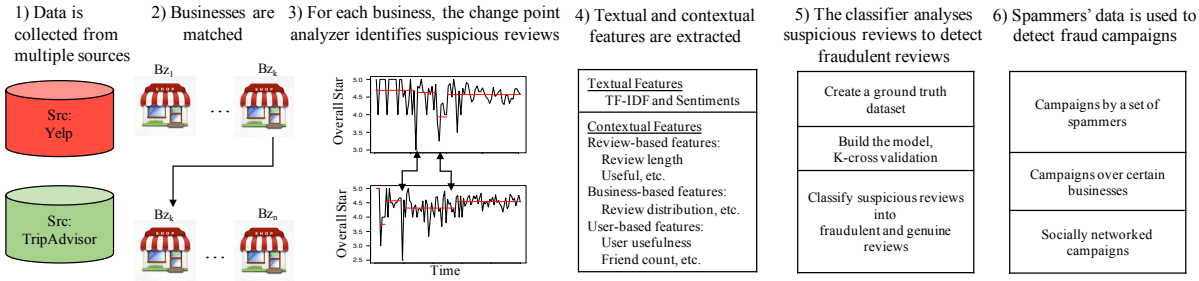


Figure 1: OneReview System Overview.

supervised machine learning to detect fraudulent reviews, and (6) detecting fraud campaigns. We explain each step in more details.

5.1 Linking Multiple Datasets

In crowd-sourced review websites, usually the name and address of businesses are not complete, which makes it not trivial matching a business between multiple websites. Our matching algorithm has two steps. Consider a restaurant in site A, called b_A, located in city c. First, the address attributes of the restaurant are compared with those of all restaurants in a site B that are also located in the same city c. Any restaurant in site B with an address similar to b_A is added to the set of possible matches M. The addresses of two restaurants located in the same city are considered similar if: 1) their ZIP codes are equal or missing, 2) their house numbers are equal or missing, and, 3) the Jaccard similarity index between their street name strings is above a threshold τ_s, or any of their street names is missing. Second, the restaurant name for b_A is compared with the names of all the restaurants in M by computing the Jaccard similarity index between their restaurant name strings. The restaurant b_A is matched to a restaurant b_B in M with the maximum Jaccard similarity index, provided that it is above a threshold τ_n.

For our purpose, the mapping algorithm should have a *perfect precision* score to avoid wrong comparison of businesses. We sampled 700 restaurants (100 restaurants per city) from the two datasets described in Section 4, and for several values of τs manually checked the precision of the mappings. We found that providing $\tau_s = 0.3$ and $\tau_n = 0.3$, the precision of matched restaurants is 1.0. The smaller dataset, Tripadvisor, includes about 10K businesses, and we could successfully match about 60% of them to the Yelp businesses. We believe that over time, eventually all businesses would have precise profiles on all websites, which results in more coverage.

Our final Yelp dataset after matching 6,068 businesses includes 805,608 reviews, and 341,399 reviewers, while our final TripAdvisor dataset includes 462,820 reviews, and 234,577 reviewers.

5.2 Comparing Trends

Intuitively, one would expect that the reputation of a business should be reasonably similar in several crowd-sourced websites. Therefore, we assume that at least the *trends* in terms of the overall ratings on multiple sites match over time, *i.e.,* within a time window, their ratings are either increasing or decreasing. OneReview analyzes the difference of trends by detecting change points in the time series that are created on reviews' overall scores. Change points are those data points at which the statistical properties such as mean, variance, correlation, or spectral density of a series of ratings change. For example, if the star ratings for a business in the months of January and February are $\{5, 5, 4, 5, 5\}$ and $\{2, 1, 5, 4\}$, respectively, then at least one change point is detected at end of month of January, because the ratings are suddenly dropped from 5s at the end of January to 2 and 1.[1]

Not all businesses receive reviews at the same rate, or at a constant rate over time, which can have an impact on the performance of Change Point Analysis. For example, based on our dataset, on Yelp less than 8% of restaurants receive daily reviews, while more than 80% obtain reviews monthly. To overcome this limitation, we compute the mean of overall ratings for each business per month, and use the sequence of these monthly mean values as the data points in the time series. While a period of one month might cause our system to miss short-lived campaigns, it is a good trade-off between detecting suspicious reviews that actually impact a business and not having effective change point analysis.

Change Point Detection. For each set of matched businesses, OneReview generates time-series from the original crawled data, one for a business' star ratings on Yelp and one for its ratings on TripAdvisor. These time-series are the inputs to the change point analysis. More formally, for each business, we have an ordered sequence of data points, $s_{1:n}^{src_1} = (s_1, \ldots, s_n)$, where s_i is the mean of stars in the i^{th} month in a data source, src_1. A change point occur at time $\tau \in \{1, \ldots, n-1\}$, such that the statistical properties of $\{s_1, \ldots, s_\tau\}$ and $\{s_{\tau+1}, \ldots, s_n\}$ are different in some way.

There are a variety of Change Point Analysis methods, differing in terms of which statistics they monitor changes in [7, 14, 21, 42, 43]. The most common method is MeanVar PELT [21], a multi-change point method, which leverages both mean and variance.

Change point analysis is a statistical analysis method. Shifts are detected as change points, if they are above certain thresholds and statistically significant. The number of detected change points then depends on the *penalty* parameter. A lower penalty value results in more change points being identified, *i.e.,* it is more sensitive. In our case study of Yelp and TripAdvisor, we tested several values including $\{p, log(n), 0.5 * log(n)\}$, where n is the number of data points in the time series and p is derived from an "elbow" plot [18].

Using MeanVar PELT method and penalty= p, $log(n)$ and $0.5 * log(n)$, we identified 26,835, 37,339 and 45,086 change points on the

[1]Note that depending on sensitivity threshold of the change point analysis method, more than one change point can be identified in these two months.

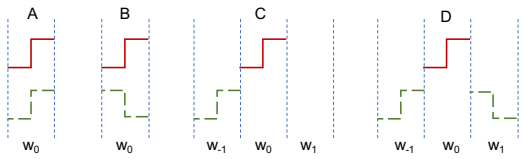


Figure 2: CPA identifies scenarios B and D as suspicious.

Yelp time-series, respectively. The set of change points identified with penalty=p is a subset of those identified with penalty $log(n)$ and $0.5 * log(n)$. The penalty can be used for tuning the sensitivity of OneReview. A more conservative system would choose the larger penalty value, which results in lower number of suspicious reviews.

In addition to identifying change points, OneReview also stores their directions, *i.e.,* positive or negative. We define a positive direction $d_{\uparrow}$ for a change point at time τ when the statistical value of $\{s_1, \ldots, s_\tau\}$ is smaller than that for $\{s_{\tau+1}, \ldots, s_n\}$. For example, a trend is considered positive, if the mean value of a business' star rating in a month is higher than its previous month. Similarly, a negative direction $d_{\downarrow}$ is defined for a change point at time τ when the statistical value of $\{s_1, \ldots, s_\tau\}$ is larger than that for $\{s_{\tau+1}, \ldots, s_n\}$.

Detecting Suspicious Change Points. We propose a Change Point Analyzer (CPA) that evaluates the change points detected in multiple websites with the goal of locating suspicious change points and reviews associated with them. The evaluation is based on several scenarios, from which four are shown in Figure 2.

First, consider at time window w_0, a change point is detected in both sources. If the direction of both change points are the same, both positive (scenario A in Figure 2) or negative, $d_{\uparrow}^{src_1}$ and $d_{\uparrow}^{src_2}$, or $d_{\downarrow}^{src_1}$ and $d_{\downarrow}^{src_2}$, then CPA labels them as *benign*. However, if their directions are not aligned, *i.e.,* one is positive and the other one is negative (scenario B in Figure 2) , $d_{\uparrow}^{src_1}$ and $d_{\downarrow}^{src_2}$, or $d_{\downarrow}^{src_1}$ and $d_{\uparrow}^{src_2}$, then they are labeled as *suspicious*.

Second, at time window w_0, a change point may be detected in src_1 but not in src_2 (scenarios C and D in Figure 2). This can indicate a suspicious behavior, however it also can be due to the sparsity of data in src_2. In other words, sometimes there is a delay in genuine change points simply because no review is provided for the business at some specific date. To minimize the false positives resulting from this phenomenon, when at time w_0 only a change point is detected in src_1 and not in src_2 , CPA extends the comparison to change points in neighbor time windows, w_{-1} and w_1. In particular, if in src_2, in time w_{-1}, any change point exists, then if the direction of any of the detected change points matches with that of the change point in src_1, then that change point in src_1 is labeled as *benign* (scenario C in Figure 2), otherwise as *suspicious* scenario D in Figure 2). This algorithm can be generalized for considering more than two review sources so that change points are labeled as *suspicious* if the majority of the change points differ in direction or do not exist. With penalty= p and $0.5 * log(n)$, CPA identified 26,835 and 61,817 change points as *suspicious*.

Finally, for every suspicious change point, OneReview retrieves the reviews contributed to that change point and label them as *suspicious*. Not all of the suspicious reviews are *fraudulent*; OneReview next employs machine learning using the textual and contextual features to identify fraudulent reviews within this set.

5.3 Feature Selection

For each review, a set of textual and contextual features from the review, its author, and the business are extracted. While inspired by previous work, we proposed some new features: "Author Overall Review Distribution," "Author Fresh Review Distribution," "Author Star Similarity," "Overall Author Usefulness," "Fresh Author Usefulness," and "Business star similarity". In this section, we use the following notations. For a review r, r_{text} and r_{date} are the text of r and the date it is posted. $A(r)$ and $B(r)$ indicate the author and business of r. $R(A)$ and $R(B)$ are reviews posted by A or written for B. A_f indicates "friends" of the author.

Review-based Features

n-grams. We extract bi-grams and uni-grams from all reviews and considered those with TF-IDF values higher than 0.5, which results in 168 uni-grams and about 6K bi-grams [40, 53].

Similarity Score. We use MinHash algorithm [23] to identify pairs of reviews that are similar to each other.

Sentiment. Li et al. [25] showed that over-emphasized sentiment offers robust cues to deceptive reviews. Using *NLTK* [8], each review is assigned a positivity, a negativity, and neutrality score.

Text Length. Previous works find that short reviews are more likely to be fraudulent [28]. Also, in our Yelp dataset, longer reviews are more likely to get *useful* votes. Text Length is defined as a logarithmic function of text length: $log(len(r_t) + 1)$.

Review Usefulness. It is defined as $\sqrt{r_{\text{useful}}}$, where r_{useful} is the number of useful votes received by others. Intuitively, *useful* votes are a gauge of the perceived review quality and trustworthiness.

Author-based Features

Author Overall Review Stars Distribution. We use Maximum Log Likelihood Estimation [2] to measure the similarity between distribution of ratings by an author, and all ratings.

Author "Fresh" Review Stars Distribution. Similar to the previous feature, it measures the similarity between distribution of all reviews stars and that of *fresh* reviews written by the author in the same month that the review has been posted.

Friend Count. This feature is defined as $log(|A(r)_f| + 1)$. Spam reviews are usually posted by users with fewer friends [28].

Elite Score. Yelp users can be recognized as "Elite" for each year they qualify. Elite status is intended to indicate that reviews are more helpful. We define this feature by $\sum_{e \in elites} \frac{1}{(|r_{\text{date}}.\text{year} - e| + 1)^2}$, where e is the year the user was awarded an Elite badge, $elites$ are all the Elite badges received by the author, and r_{date} is the review date. It measures the distance between the time of the review's publication and the user's Elite status being awarded.

Author Review Count. It is a logarithmic function of number of reviews by the author: $\log(|R(A)| + 1)$.

Author Star Similarity. It measures the similarity of a review's rating to the average ratings of all reviews by A: $|r_{stars} - \text{average}(R(A))|$.

Overall Author Usefulness. It measures the number of "useful" votes that an author has received: $\log(|\text{usefulness}(A)| + 1)$.

Fresh Author Usefulness. This feature measures the number of "useful" votes that an author has received for their *fresh* reviews.

Business-based Features

Business Overall Review Stars Distribution. This feature measures the similarity between distribution of all reviews stars and the distribution of reviews stars that have been written for B.

Business Fresh Review Stars Distribution. It compares distribution of all reviews with that of B's *fresh* reviews.

Business Star Similarity. The similarity of a review's rating to the average rating of all reviews for business B: $|r_{\text{stars}} - \text{average}(R(B))|$.

5.4 Classification

OneReview employs supervised machine learning using the features to classify a review as *fraudulent* or *benign*. Obtaining a ground truth of fraud on content containing people's opinions is a hard problem. We discuss our approach in Section 6. For classification, we chose Random Forests [11, 20]. This is due to its usefulness in a wide variety of applications [12], its resistance to over-fitting, and its utility in understanding feature importance [17]. For validating our classification, we apply k-fold cross validation, and use the resulting model to classify *fraudulent reviews*.

6 GROUND TRUTH DATASETS

There is no pre-existing, and reliable ground truth corpus of fraudulent reviews (discussed in Section 2). We employed a combination of human workers and algorithms to create a ground truth dataset of real-world fraudulent reviews. One major advantage of this dataset is that it includes both the *text* of the reviews and their *metadata*.

Obtaining Fraudulent Labeled Data

Our fraudulent ground truth dataset has 841 reviews, including:

379 Yelp "Not Recommended" reviews. Yelp employs a fraudulent detection mechanism to detect *fraudulent reviews* [35], and separate them as 'Not Recommended reviews.' We crawled Yelp on March 2017, about 2 months after Yelp dataset was published, and identified 1,341 reviews that since then Yelp had identified as Not Recommended. Yelp filters reviews for reasons beyond fraud, such as harassment, discrimination, or other offensive language. Therefore, we contribute 379 Not Recommended reviews that are also among suspicious reviews identified during change-point analysis.

370 duplicate reviews. We found 370 duplicate reviews that are posted for the same businesses, and are Yelp-to-Yelp matches. Interestingly, none were written by the same user, therefore, considered as fraudulent reviews. It is not trivial to accurately match users in Yelp to the users in TripAdvisor, and we could not validate if a duplicate review is written by the same user. As a result, we discarded all the 2,761 duplicate Yelp-to-TripAdvisor and TripAdvisor-to-Yelp duplicate reviews that are written for the same businesses. We do not consider identical reviews that have a length shorter than 100 characters, because they can be genuine identical short reviews.

92 crowd-labeled fraudulent reviews. To find reviews above and beyond those that are merely duplicates, or already labeled by Yelp's algorithm, we conduct a study using Amazon Mechanical Turk (AMT) to identify fraudulent reviews. Our research study was approved by the IRB of University of California, Santa Barbara.

Study Design. We created a pool of 1,700 *Human Intelligence Tasks* (HITs) [3]. Each HIT shows five different reviews to the worker, where they decide whether a review is *Strongly Fraudulent*, *Fraudulent*, *Cannot Tell*, *Benign*, and *Strongly Benign*. We only allow a single submission per Turker. Turkers are located in United States, and maintain an approval rating of at least 98% and have more than 10,000 approved HITs. To reduce the risk of dishonest AMT workers [38], we inserted verification questions into each HIT, asking the Turker to pick a specific option. Turkers are expected to spend 15-20 minutes on a HIT, and are paid 2.25 US dollar for an accepted submission. Reviews shown to the workers are sampled randomly. To give the workers some context, we provide them with additional supporting data, including review, author, and business information. We also provide a sample of four reviews posted for this business around the date of the review in question.

Study Results. We employed 1,837 Turkers, and had to discard 137 responses. Previous research [10, 36] has shown that manual labeling of opinions is not easy and human judgment is not particularly accurate at determining a fraudulent review. Therefore, we computed the majority vote by count of four, *i.e.,* if four Turkers identify a review *Strongly Fraudulent (Strongly Benign)* or *Fraudulent (Benign)*, then that review is labeled as *Fraudulent (Benign)*. Ultimately, from 1,700 reviews, we obtained 92 fraudulent reviews.

Obtaining Benign Labeled Data

Useful reviews by Elites. We use two of Yelp's meta-datas, including Elite and Usefulness. To be more cautious, we obtained reviews that are tagged as *useful* at least 3 times. We trust the reviews in the intersection of these two sets to be benign, $\text{Benign}_{\{\text{elite, useful}\}} = R_{\text{elite}} \cap R_{\text{useful}^{\geq 3}}$, which includes more than 46K reviews.

Reviews by trusted real-life acquaintances. We noticed that reviews that users find "useful," and users that Yelp marks as "Elite" share a bias towards longer, more detailed reviews, from very active accounts. Thus, we supplemented the dataset with reviews obtained using the authors' Yelp social connections, yielding 614 reviews.

Unbalanced Dataset

Prior work has shown that constructing labelled datasets, where the ratio of benign to malicious samples does not match that in practice can result in differences of more that an order of magnitude in the classification errors [29, 52]. Thus, we create a more realistic ground truth dataset with a genuine-to-fraudulent ratio of 4:1. We chose this ratio due to an estimation that the ratio of fake reviews on Yelp is 20% [28]. Therefore, we sampled 4,200 useful review by Elites. *Our final benign ground truth dataset is created by union of these samples, Benign* $= \text{Benign}_{\{\text{elite, useful}\}} \cup \text{Benign}_{\{\text{connections}\}}$, *including 4,814 reviews.*

7 EVALUATION

We show the effectiveness of our approach by statistical analysis. We first examined the classifier and the affect of Change Point Analysis on the results. We then apply OneReview on our dataset to detect and characterize real-world fraud campaigns.

Classification Performance. Table 1 shows the performance of the random forest classifier on our ground truth dataset. The classifier successfully detects fraudulent reviews with high accuracy (97%), precision (91%) and recall (90%). The standard error values (+/-) indicate that the performance is not drastically different for different number of trees or cross-folds. The reported results are with $k = 5$ folds, and 1000 trees. We picked Gini impurity to measure the quality of a split, and used bootstrapping when building trees. Table 1 also shows the performance of classifier with only textual features (TF-IDFs and sentiment features). The performance

Table 1: The performance of OneReview.

Features	Accuracy	F1-score	Precision	Recall
All	0.97 (+/- 0.01)	0.90 (+/- 0.04)	0.91 (+/- 0.03)	0.90 (+/- 0.06)
Textual	0.86 (+/- 0.01)	0.59 (+/- 0.05)	0.55 (+/- 0.05)	0.64(+/- 0.11)

is poor with 55% precision and 64% recall. We also tested various subsets of features however, none reached the same level of performance. The results illustrate that using all the contextual features substantially increases the performance. Since *similarity*, *review usefulness*, and *Elite* scores are used in the creation of the ground truth dataset, we did not use these features for classification. We overcome the limitation of unbalanced ground truth dataset, by using a well-known over-sampling technique called SMOTE [13].

Impact of Change Point Analysis. We tested our classifier on all the reviews in our Yelp dataset. It classified 375,359 reviews as fraudulent, which is equivalent to about 47% of all the 805K Yelp reviews. It is not reasonable to consider 47% of the reviews as fraudulent. However, from our MTurk study, we found that fraudulent reviews appear in suspicious reviews with higher probability. In particular, from 92 fraudulent reviews identified by Turkers, about 32% also appear in suspicious dataset while this dataset consists of only 24% of all the reviews. OneReview only performs the classification on the suspicious reviews to predict *fraudulent reviews*. As explained in Section 5.2, we obtained two sets of suspicious reviews with (1) $penalty = p$, and (2) $penalty = 0.5 * log(n)$, where the first set is a subset of the second. From 73K suspicious reviews in the first set, OneReview identified 61,983 fraudulent reviews, about 85% of the suspicious and 8% of all the 805K Yelp reviews. From 165K suspicious reviews in the second set, it identified 96,445 fraudulent reviews, about 58% of the suspicious and 12% of all the reviews. These results provide a lower and upper bound for the number of *fraudulent reviews* in Yelp. In the reminder of the paper, we focus on the *fraudulent reviews* from the smaller set of suspicious reviews.

Comparison with Yelp Reviews. We compared characteristics of detected *fraudulent reviews* and their authors with those of Yelp reviews and Not Recommended reviews. Due to space brevity, we do not provide the details of this comparison. In summary, this analysis showed that OneReview does not detect only one specific type of fraudulent review, it detects both positive and negative reviews, short and long reviews, those posted for both new and popular restaurants, and even those posted by Elite users.

Examining Fraud Campaigns. We refer to a *campaign* as one or more human actors controlling more than one account, with the aim of posting multiple reviews to influence the reputation of a business. These untrustworthy accounts can be bots, sybil accounts or individual spammers. We found that some untrustworthy accounts are created and used only once for a specific campaign while some are used several times for multiple campaigns.

Untrustworthy Accounts. We analyzed untrustworthy accounts, *i.e.,* Yelp users who have posted *fraudulent reviews*. In total, 58,157 untrustworthy accounts are identified (17% of total Yelp users). We found that about 8% of them (i.e., 7,064) have only posted one fake review. In the Yelp dataset, 27,347 of users only posted one review. Thus, OneReview does not simply identify every reviewer with one review as a untrustworthy account. Moreover, 14,910 of them are mainly posting fraudulent reviews with more than 40% of their reviews being fraudulent. This can indicate untrustworthy accounts who post legitimate reviews in an attempt to avoid the detection.

Campaigns Targeting Specific Businesses. Out of 6,068 businesses in our dataset, fraudulent reviews are posted for 3,980 businesses (about 66%). Some businesses have received many fraudulent reviews, *e.g,* 1,332 and 56 of businesses have received more than or equal to 10 and 100 fraudulent reviews, respectively. We further examined if these fraud campaigns are positive or negative. A fraudulent review is considered positive or negative if its rating is bigger or smaller than the overall rating of the business. Overall, the set of *fraudulent reviews* consists of 40,069 (65%) positive and 21,914 (35%) negative reviews. While it is expected that most campaigns are positive, it is interesting that still a good amount of *fraudulent reviews* are negative. Among 3,980 businesses with *fraudulent reviews*, 501 and 344 only have received positive and negative fraudulent reviews, respectively.

Socially-Networked Fraud Campaigns. A set of untrustworthy accounts can simply add each other as friends. OneReview does not consider the structure of the social network as a feature, still it can detect *fraudulent reviews* posted by fraud campaigns. We constructed the social network of untrustworthy accounts, which includes 16,738 nodes and 24,909 edges. We used the Louvain community detection algorithm [9], and found several communities. Some of these communities include several hundreds of nodes, *e.g.,* the top 5 largest communities include 1671, 1428, 1385, 565, and 539 of them. We found some patterns that can indeed distinguish them as untrustworthy accounts. For example, we found 40 untrustworthy accounts from a campaign which posted 40 *fraudulent reviews* for a single restaurant. These 40 users, between them have ever posted only 90 reviews, all within a month, spread across 47 distinct venues, while all sharing one venue in common. We further studied the content of these reviews and compared with the *fraudulent reviews*. We found that the authors of these reviews tended to use more superlative words, such as "definitely", "great" or "really." For example, the word occurrence ratio of the word "love" is almost two times bigger than the corresponding ratio in *fraudulent reviews*, while for the word "like" the opposite trend is observed.

8 LIMITATIONS

We included data from a mix of sources, to help reduce any bias during creation of the ground truth dataset, although any bias still present in the data is inherently difficult to measure. Change Point Analyzer evaluates time-series on a one-month interval. While this may produce some latency between a review being posted and OneReview's classification in a real deployment, this is an artifact of the sparsity of our data sources. This parameter can be adjusted, even at the granularity of a single business, when higher-frequency data is available.

9 CONCLUSION

We presented OneReview, an approach for finding fraudulent activity in crowd-sourced review services, using correlated analysis across multiple independent services. Our method leveraged change point analysis, in tandem with a cross-dataset matching scheme, to find points in the stream of reviews where a significant change in

a business' reputation did not occur evenly across the review websites. We believe our approach of integrating external data sources and features, or "thinking outside the dataset," is an important part of our system's efficacy, and allows our system to perform in a way that no machine or human could before. For future work, we aim to evaluate OneReview when having access to data from a third review website such as Google.

REFERENCES

[1] Hojjat Aghakhani, Aravind Machiry, Shirin Nilizadeh, Christopher Kruegel, and Giovanni Vigna. 2018. Detecting Deceptive Reviews using Generative Adversarial Networks. *arXiv preprint arXiv:1805.10364* (2018).

[2] Hirotogu Akaike, BN Petrov, and F Csaki. 1973. Information theory and an extension of the maximum likelihood principle. (1973).

[3] Amazon. 2018. *About Amazon Mechanical Turk.* https://www.mturk.com/worker/help

[4] Michael Anderson and Jeremy Magruder. 2012. Learning from the crowd: Regression discontinuity estimates of the effects of an online review database. *The Economic Journal* 122, 563 (2012), 957–989.

[5] Anonymous. 2018. Get Paid to Write Reviews: 27 Sites That Pay You (with Cash & Free Stuff!). http://moneypantry.com/get-paid-to-write-reviews/. (2018).

[6] Anonymous and Symon, Evan V. . 2016. I Get Paid To Write Fake Reviews For Amazon. http://www.cracked.com/personal-experiences-2376-i-get-paid-to-write-fake-reviews-amazon.html. (2016).

[7] Ivan E Auger and Charles E Lawrence. 1989. Algorithms for the optimal identification of segment neighborhoods. *Bulletin of mathematical biology* 51, 1 (1989), 39–54.

[8] Steven Bird. 2006. NLTK: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions.* Association for Computational Linguistics, 69–72.

[9] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[10] Charles F Bond Jr and Bella M DePaulo. 2006. Accuracy of deception judgments. *Personality and social psychology Review* 10, 3 (2006).

[11] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[12] Rich Caruana and Alexandru Niculescu-Mizil. 2006. An Empirical Comparison of Supervised Learning Algorithms. In *Proceedings of the 23rd International Conference on Machine Learning (ICML '06).* ACM, New York, NY, USA, 161–168. https://doi.org/10.1145/1143844.1143865

[13] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.

[14] Jie Chen and Arjun K Gupta. 2011. *Parametric statistical change point analysis: with applications to genetics, medicine, and finance.* Springer Science & Business Media.

[15] Geli Fei, Arjun Mukherjee, Bing Liu, Meichun Hsu, MalÃž Castellanos, and Riddhiman Ghosh. 2013. Exploiting Burstiness in Reviews for Review Spammer Detection.. In *ICWSM.* The AAAI Press.

[16] Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2.* Association for Computational Linguistics, 171–175.

[17] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2001. *The elements of statistical learning.* Vol. 1. Springer series in statistics Springer, Berlin. 587–588 pages.

[18] Douglas M Hawkins. 2001. Fitting multiple change-point models to data. *Computational Statistics & Data Analysis* 37, 3 (2001), 323–341.

[19] Nitin Jindal and Bing Liu. 2008. Opinion Spam and Analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining (WSDM '08).* ACM, New York, NY, USA, 219–230. https://doi.org/10.1145/1341531.1341560

[20] H Tim Kam. 1995. Random decision forest. In *Proc. of the 3rd Int'l Conf. on Document Analysis and Recognition, Montreal, Canada, August.* 14–18.

[21] Rebecca Killick, Paul Fearnhead, and Idris A Eckley. 2012. Optimal detection of changepoints with a linear computational cost. *J. Amer. Statist. Assoc.* 107, 500 (2012), 1590–1598.

[22] Raymond Y. K. Lau, S. Y. Liao, Ron Chi-Wai Kwok, Kaiquan Xu, Yunqing Xia, and Yuefeng Li. 2012. Text Mining and Probabilistic Language Modeling for Online Review Spam Detection. *ACM Trans. Manage. Inf. Syst.* 2, 4, Article 25 (Jan. 2012), 30 pages. https://doi.org/10.1145/2070710.2070716

[23] Hee Andy Lee, Rob Law, and Jamie Murphy. 2011. Helpful reviewers in TripAdvisor, an online travel community. *Journal of Travel & Tourism Marketing* 28, 7 (2011), 675–688.

[24] Huayi Li, Zhiyuan Chen, Arjun Mukherjee, Bing Liu, and Jidong Shao. 2015. Analyzing and Detecting Opinion Spam on a Large-scale Dataset via Temporal and Spatial Patterns.. In *ICWSM.* 634–637.

[25] Jiwei Li, Myle Ott, Claire Cardie, and Eduard H Hovy. 2014. Towards a General Rule for Identifying Deceptive Opinion Spam.. In *ACL (1).* Citeseer, 1566–1576.

[26] Ee-Peng Lim, Viet-An Nguyen, Nitin Jindal, Bing Liu, and Hady Wirawan Lauw. 2010. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM international conference on Information and knowledge management.* ACM, 939–948.

[27] Yuming Lin, Tao Zhu, Hao Wu, Jingwei Zhang, Xiaoling Wang, and Aoying Zhou. 2014. Towards online anti-opinion spam: Spotting fake reviews from the review sequence. In *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on.* IEEE, 261–264.

[28] Michael Luca and Georgios Zervas. 2016. Fake it till you make it: Reputation, competition, and Yelp review fraud. *Management Science* (2016).

[29] Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. 2011. Learning to detect malicious urls. *ACM Transactions on Intelligent Systems and Technology (TIST)* 2, 3 (2011), 30.

[30] Microsoft. 2017. Febipos.A Malware. http://www.microsoft.com/security/portal/threat/encyclopedia/Entry.aspx?Name=Trojan:JS/Febipos.A. (September 2017).

[31] Arjun Mukherjee, Abhinav Kumar, Bing Liu, Junhui Wang, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Spotting opinion spammers using behavioral footprints. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 632–640.

[32] Arjun Mukherjee, Bing Liu, and Natalie Glance. 2012. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st international conference on World Wide Web.* ACM, 191–200.

[33] Arjun Mukherjee, Bing Liu, Junhui Wang, Natalie Glance, and Nitin Jindal. 2011. Detecting Group Review Spam. In *Proceedings of the 20th International Conference Companion on World Wide Web (WWW '11).* ACM, New York, NY, USA, 93–94. https://doi.org/10.1145/1963192.1963240

[34] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie Glance. 2013. What yelp fake review filter might be doing?. In *Seventh international AAAI conference on weblogs and social media.*

[35] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie Glance. 2013. What Yelp Fake Review Filter Might Be Doing? http://www.aaai.org/ocs/index.php/ICWSM/ICWSM13/paper/view/6006

[36] Myle Ott, Claire Cardie, and Jeffrey T Hancock. 2013. Negative Deceptive Opinion Spam.. In *HLT-NAACL.* 497–501.

[37] Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1.* Association for Computational Linguistics, 309–319.

[38] Gabriele Paolacci, Jesse Chandler, and Panagiotis G Ipeirotis. 2010. Running experiments on amazon mechanical turk. (2010).

[39] Mahmudur Rahman, Bogdan Carbunar, Jaime Ballesteros, George Burri, Duen Horng, et al. 2014. Turning the Tide: Curbing Deceptive Yelp Behaviors.. In *SDM.* SIAM, SIAM, 244–252.

[40] Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning.*

[41] Reviews That Stick. 2017. Buy Positive Yelp Reviews. http://reviewsthatstick.com/yelp/. (2017).

[42] Gordon J Ross et al. 2013. Parametric and nonparametric sequential change detection in R: The cpm package. *Journal of Statistical Software* 78 (2013).

[43] ANDREW JHON Scott and M Knott. 1974. A cluster analysis method for grouping means in the analysis of variance. *Biometrics* (1974), 507–512.

[44] Review Shepherd. 2017. How To Get Yelp Reviews. https://reviewshepherd.com/articles/get-yelp-reviews/. (2017).

[45] Tim Parker. 2017. Posting Fake Reviews For Your Business May Cost You. https://quickbooks.intuit.com/r/marketing/posting-fake-reviews-for-your-business-may-cost-you/. (2017).

[46] Bimal Viswanath, M. Ahmad Bashir, Mark Crovella, Saikat Guha, Krishna P. Gummadi, Balachander Krishnamurthy, and Alan Mislove. 2014. Towards Detecting Anomalous User Behavior in Online Social Networks. In *23rd USENIX Security Symposium (USENIX Security 14).* San Diego, CA, 223–238.

[47] Gang Wang, Tianyi Wang, Haitao Zheng, and Ben Y Zhao. 2014. Man vs. Machine: Practical Adversarial Detection of Malicious Crowdsourcing Workers.. In *USENIX Security Symposium.* 239–254.

[48] Guan Wang, Sihong Xie, Bing Liu, and Philip S Yu. 2012. Identify online store review spammers via social review graph. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3, 4 (2012), 61.

[49] Yuanshun Yao, Bimal Viswanath, Jenna Cryan, Haitao Zheng, and Ben Y. Zhao. 2017. Automated Crowdturfing Attacks and Defenses in Online Review Systems. In *ACM Conference on Computer and Communications Security (CCS ?17).* Dallas, Texas.

[50] Yelp. 2016. Yelp Dataset Challenge. (September 2016). https://www.yelp.com/dataset_challenge.

[51] Kyung-Hyan Yoo and Ulrike Gretzel. 2009. Comparison of deceptive and truthful travel reviews. *Information and communication technologies in tourism 2009* (2009), 37–47.

[52] Bianca Zadrozny, John Langford, and Naoki Abe. 2003. Cost-sensitive learning by cost-proportionate example weighting. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE, 435–442.

[53] Wen Zhang, Taketoshi Yoshida, and Xijin Tang. 2011. A comparative study of TF* IDF, LSI and multi-words for text classification. *Expert Systems with Applications* 38, 3 (2011), 2758–2765.

Entity Personalized Talent Search Models with Tree Interaction Features

Cagri Ozcaglar, Sahin Geyik, Brian Schmitz, Prakhar Sharma, Alex Shelkovnykov, Yiming Ma, Erik Buchanan
LinkedIn
{caozcagl,sgeyik,bschmitz,prsharma,ashelkovnykov,yma,ebuchanan}@linkedin.com

ABSTRACT

Talent Search systems aim to recommend potential candidates who are a good match to the hiring needs of a recruiter expressed in terms of the recruiter's search query or job posting. Past work in this domain has focused on linear and nonlinear models which lack preference personalization in the user-level due to being trained only with globally collected recruiter activity data. In this paper, we propose an entity-personalized Talent Search model which utilizes a combination of generalized linear mixed (GLMix) models and gradient boosted decision tree (GBDT) models, and provides personalized talent recommendations using nonlinear tree interaction features generated by the GBDT. We also present the offline and online system architecture for the productionization of this hybrid model approach in our Talent Search systems. Finally, we provide offline and online experiment results benchmarking our entity-personalized model with tree interaction features, which demonstrate significant improvements in our precision metrics compared to globally trained non-personalized models.

CCS CONCEPTS

• **Information systems → Web searching and information discovery**; **Content ranking**; **Personalization**.

KEYWORDS

Search ranking, Personalization, Nonlinear personalized models, GLMix models, XGBoost

ACM Reference Format:
Cagri Ozcaglar, Sahin Geyik, Brian Schmitz, Prakhar Sharma, Alex Shelkovnykov, Yiming Ma, Erik Buchanan. 2019. Entity Personalized Talent Search Models with Tree Interaction Features. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313672

1 INTRODUCTION

The task of *Talent Search* is to connect job seekers with hiring managers or recruiters who are looking for candidates to fill available positions in their companies. The main way we approach this task is from the perspective of the recruiter; that is, we would like to recommend the best candidates to a recruiter who either composes an explicit search query or posts a job description. Previous work in the Talent Search domain aimed to learn a global model for the whole candidate space, by looking at the two-way interactions between recruiters and candidates, such as sending a message for an opportunity (i.e. interest from the recruiter towards the candidate, in the context of the current query or job) and receiving a positive or negative response to such messages (i.e. interest from the candidate for the opportunity) [5], [23]. While there has been considerable success with such methods, they still suffer because there is no personalization of preference inference for recruiters. While context of the job or query aims to contain some degree of personalization, there is often the need to learn different recommendation models at the recruiter level; so that there is a better match with the recruiter and the type of candidates that are recommended to them.

In order to personalize recommendations for different users of the same system, generalized linear mixed (GLMix) Models [27], [17] are often utilized, which generate per-entity linear models. In practice, GLMix models work similar to a decision tree in the way that they retrieve a different set of linear coefficients based on the id of an entity (e.g. they could employ a recruiter id → model coefficients mapping for our application case). While this introduces nonlinearity to the learned models, the features are still combined in a linear manner, and hence does not exploit the full interaction information. On the other hand, nonlinear models such as gradient boosted decision trees (GBDT) are costly to generate per entity, due to the required processing power, the size of each of the learned models, and because they are susceptible to overfitting in the case of limited data available for personalization. In this work, we propose to have the best of both worlds, by having entity-level personalization through GLMix models, while using tree interaction features [10] from a globally learned GBDT model (via XGBoost [2]), utilizing the usage data of our talent search system.

The contributions of the current work are:

- Utilization of entity-level (i.e. recruiters and contracts) personalized models for talent search domain,
- Achieving non-linear feature transformations within GLMix models via the tree interaction features computed by globally trained GBDT models,
- Extensive offline experimentation of such models on recruiter recommendation data, exploring different levels of personalization (i.e. different entity-types), and,
- Real-world online A/B test results demonstrating significant improvements in the successful search sessions, where we improve the two-way interest (both recruiter and candidates are matched within the context of an opportunity) success rate.

The organization for the rest of the paper is as follows. First, we present a brief background of the talent search domain, as well

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313672

as previous personalization efforts in Section 2, followed by the details of our methodology for personalized models in Section 3. We present the offline and online architecture for the usage of personalized models in real-world talent search applications in Section 4; and our offline and online experiment results in Section 5. Finally, we conclude the paper with a summary and discussion of future directions in Section 6.

2 BACKGROUND

2.1 Talent Search Models

There have been several previous work that aim to solve the talent search task in different perspectives and under different constraints. One such system which generates candidates' inferred skill expertise scores using collaborative filtering based on matrix factorization, then utilizes them as features in a supervised learning-to-rank model optimizing for normalized discounted cumulative gain (NDCG, [11]) is presented in [6]. An approach which aims to provide in-session personalization due to immediate user feedback is presented in [5]. We distinguish our work from this effort mainly due to the fact that we have access to the globally collected data offline to train warm-start personalized models.

An exploratory work on utilizing representation and deep learning models for talent search is presented in [23]. There has also been a shift from querying by keyword to querying by example in talent search models [8]. Finally, *search by ideal candidates* approach prompts recruiters to enter ideal candidates for their open position's requirements, and a query is generated based on these ideal candidates [7]. Following query generation, the talent search system retrieves and ranks results based on the match to the query.

2.2 Personalized models

Personalized models can be unsupervised learning models used in recommender systems, or supervised learning models, such as extensions of generalized linear models (GLMs) with user-specific model components. Recommender systems can be grouped into three categories: content-based systems, collaborative filtering systems, and latent factor models [22]. Content-based systems use the attributes of items highly rated by users, and recommend similar items to the user [14]. Collaborative filtering based systems, on the other hand, utilize user-to-user similarity based on their item preferences, and recommend the items liked by similar users [16] [3]. Hybrid of content-based filtering and collaborative filtering methods have also been used to improve recommendations [1] [18] [15]. A third approach is latent factor models, using methods such as principal component analysis (PCA), singular value decomposition (SVD), or more recent methods such as the BellKor recommender system, which won Netflix Challenge Prize in 2009 [13].

Supervised personalized models originate from standard machine learning models, with extensions in the form of additional per-entity model components. Among various methods used for prediction of users' responses to items, a common method is to use GLMs. Examples of such models are logistic regression for the case of binary response prediction, and linear regression in the case of real-valued response prediction. In use cases where the data is large and each entity has a sufficiently large dataset to make a generalization, it is possible to build a personalized model for each entity. GLMix models are an improvement over GLMs, where in addition to a global model, entity-level models are added to introduce personalization [17], [27].

2.3 Tree Ensemble Models

Tree-based methods partition the feature space into a set of rectangular subspaces, and then fit a model to each region [9]. Decision trees are widely used for nonlinear prediction tasks, allowing feature interactions. However, decision trees can produce unstable models which change significantly with the training data, resulting in high variance and susceptibility to overfitting. For these reasons, ensemble averaging methods such as bagging and boosting are used to reduce the variance of tree models [4]. Gradient Boosted Trees are tree ensemble models, where the ensemble is generated in a step-wise manner by sequentially fitting a function to residuals at each step. Each individual decision tree model is a weak learner, built by constraining the depth of the tree. Various implementations of Gradient boosted tree models have been proposed, including PLANET [19], XGBoost [2], parallel GBRT for web search ranking [25], stochastic gradient boosted distributed decision trees [26], and PaloBoost [20].

3 METHODOLOGY

3.1 Generalized Linear Mixed models (GLMix)

$$g(\underbrace{P(r,c,re,ca,co)}_{\text{Positive Response Prob.}}) = \underbrace{\beta_{global} \cdot f_{ltr}^{r,c,re,ca,co}}_{\text{Global model}} + \underbrace{\beta_{re} \cdot f_{mem}^{ca}}_{\text{Per-recruiter model}} + \underbrace{\beta_{co} \cdot f_{mem}^{ca}}_{\text{Per-contract model}} \quad (1)$$

where $P(r, c, re, ca, co)$ is the joint probability of recruiter re sending a communication request on the position s/he would like to fill to candidate ca, and the candidate responding positively to it. Search query is represented by request id r, context c (content of the query, properties of the recruiter, its company, the suggested position, etc.), and contract of the recruiter co (an agreement between the recruiter and the company about the recruiting task). $g(P(r, c, re, ca, co)) = log\frac{P(r,c,re,ca,co)}{1-P(r,c,re,ca,co)}$ is the logit function. The first term on the right-hand side is the global model score with β_{global} as the global model (i.e. fixed effect model) coefficients, the second term is the per-recruiter model score with β_{re} as the per-recruiter model coefficients for recruiter re which capture recruiter re's preferences, and the third term is the per-contract model score with β_{co} as the per-contract model coefficients for contract co, which capture contract co's preferences. Here, feature set $f_{ltr}^{r,c,re,ca,co}$ corresponds to Learning-To-Rank (LTR) features for the search impression, uniquely identified by (r, c, re, ca, co), and f_{mem}^{ca} corresponds to standardized member feature vector for candidate ca.

The degree of entity-level personalization in GLMix models depends on the amount of data available per entity. In the Talent Search domain, for the GLMix global model + per-recruiter model + per-contract model represented in Equation 1, there are new recruiters and contracts arriving to the system with almost no historical data, as well as recruiters and contracts who have large enough historical data for entity-level personalization. For recruiters and

candidates with almost no data, the posterior means of corresponding entity-level coefficient vectors β_{re} and β_{co} will be close to zero, and the model for the corresponding impression uniquely identified by (r, c, re, ca, co) tuple will fall back to the global model. On the other hand, for an impression with recruiter re and contract co with sufficient historical data, per-recruiter model with model coefficient vector β_{re} and per-contract model with model coefficient vector β_{co} will have impact on the final GLMix model score, which will be personalized to recruiter re and contract co.

3.2 Gradient Boosted Decision Trees

Gradient Boosted Decision Tree (GBDT) models are tree ensemble models which overcome decision trees' susceptibility to overfitting and high model variance problems via model averaging. GBDT models are computationally intensive to train. Multiple scalable implementations of GBDT model training algorithms exist in literature: [19] [25] [26]. A commonly used GBDT model training algorithm implementation is XGBoost [2]. XGBoost improves upon existing implementations with the following methods: (*i*) an approximate algorithm for enumerating all possible splits on all features using weighted quantile sketching, (*ii*) a sparsity-aware algorithm for parallel tree split finding and learning, (*iii*) a cache-aware block structure for out-of-core tree learning. With these improvements, XGBoost provides a highly scalable end-to-end tree boosting system.

3.3 Best of Both Worlds: GLMix Models with Tree Interaction Features

In the Talent Search domain, where data is abundant, entity-level personalization via GLMix models has a high potential of improving response prediction accuracy when compared to GLM counterparts. However, GLMix models do not allow nonlinear feature interactions. GBDT models enable nonlinear feature interactions through individual decision trees and tree ensembles. On the other hand, it is infeasible to add entity-level personalization to GBDT models, due to the required processing power, size of resulting personalized models, and susceptibility to overfitting in cases of limited data availability. Therefore in this work, we use a hybrid approach: first, we employ a pre-trained GBDT model via XGBoost to generate tree interaction features and scores[1] for each recruiter search impression; next, we utilize raw features, tree interaction features, and GBDT model scores as features to build a personalized GLMix model.

The GLMix global + per-recruiter + per-contract model with tree interaction features for a given (requestId, contextId, recruiterId, candidateId, contractId), represented by (r, c, re, ca, co), can be formulated as follows:

$$g(\underbrace{P(r,c,re,ca,co)}_{\text{Positive Response Prob.}}) = \underbrace{\beta_{global} \cdot f_{all}}_{\text{Global model}} + \underbrace{\beta_{re} \cdot f_{all}}_{\text{Per-recruiter model}} + \underbrace{\beta_{co} \cdot f_{all}}_{\text{Per-contract model}} \quad (2)$$

where,

- $g()$ is the logit function,
- $P(r, c, re, ca, co)$ is the probability of recruiter re communicating with candidate ca for the open position and receiving a positive response,
- β_{global} is the global model coefficient vector,
- β_{re} is the per-recruiter model coefficient vector for recruiter re,
- β_{co} is the per-contract model coefficient vector for contract co,
- $f_{all} = f_{ltr} \bigcup f_{xgb} \bigcup f_{int}$ is the feature vector, where f_{ltr} is the set of Learning-To-Rank (LTR) features; f_{xgb} represents the score from a pre-trained global GBDT model; and f_{int} represents the tree interaction features from the same pre-trained global GBDT model. As an example, if $feature_1$ is one of the LTR features, a GBDT model may generate a rule such as $feature_1 > 0.7$ within one of the internal nodes of one of its decision trees. A tree interaction score represented by one of the leaf nodes in a decision tree of the GBDT model then is a combination of multiple rules encoded into a single rule set.

The two feature sets f_{xgb} and f_{int} are generated by a pre-trained GBDT model used to score the same training set. Let the training set be $D = \{x_i, y_i\}$, where $x_i \in \mathbb{R}^m$, $y_i \in \mathbb{R}$ with $|D| = n$ examples represented using m features each. A GBDT model generated by XGBoost is an ensemble of tree models, which uses K regression trees represented as K additive functions to predict the final score of data point x_i:

$$g_{xgb}(x_i) = \sum_{k=1}^{K} t_k(f_{ltr}(x_i)),\ t_k \in T \quad (3)$$

where $T = \{t(x) = w_{(q(x))}\}$ is the set of regression trees, $q : \mathbb{R}^m \rightarrow L$ represents the structure of tree q which maps a data point $x_i \in \mathbb{R}^m$ to the corresponding leaf index in the tree, w_q represents the leaf weight in the independent tree structure q, L represents the number of leaves in the tree, and $f_{ltr}(x_i)$ represents LTR features of data point x_i. Given this representation, for each data point x_i, we score it using each tree t_k, where $k \in \{1, .., K\}$, and x_i lands on leaf node l_i^k in tree t_k. Each data point ends up in one leaf node of each tree in the ensemble, which encodes a set of rules chosen from the root to the leaf node of the tree. We represent these leaf nodes or tree interaction features for data point x_i as a (name, term, value) triple as follows:

$$f_{int}(x_i) = \bigcup_{k=1}^{K} (k, l_i^k, 1) \quad (4)$$

where k is the tree index, l_i^k is the index of the leaf node for the k-th tree which data point x_i landed on, and binary value 1 represents that data point x_i landed on this leaf node. We also use a third set

[1] Hyper-parameter optimization used for selecting regularization parameters for GLMix utilizes a grid search over a continuous large space for each entity-personalization level. In this work, we have preferred to use the GBDT score, instead of doing a more exhaustive hyper-parameter search with a model using tree interaction features + LTR features only, to regenerate GBDT score itself, especially helpful for global model. This reduces the computational complexity of training the GLMix model, particularly hyper-parameter optimization step.

of feature, $f_{xgb}(x_i)$, which is the GBDT model score:

$$f_{xgb}(x_i) = g_{xgb}(x_i) \tag{5}$$

For the pre-trained XGBoost model, we ran grid search on (training set, validation set) pair for (number of trees, maximum depth) hyperparameter pairs, and 100 trees with maximum depth of 2 was selected as the final hyperparameter. For the GLMix model, our hyperparemeters are the regularization weights for the global model, per-recruiter model, and per-contract model. We also ran grid search on (training set, validation set) pair for this regularization weight triple, and (100, 100, 100) was selected as the regularization weights for the global model, per-recruiter model, and per-contract model respectively. Figure 1 shows the pipeline for building GLMix models using learning-to-rank features, tree interaction features, and GBDT model scores. Learning-to-rank features are used as input to pre-trained GBDT model, which generates tree ensembles that are encoded into tree interaction features and GBDT model score for each data point. Then, using the original learning-to-rank features and their nonlinear transformations in the form of tree interaction features and GBDT model scores, we build a GLMix model with recruiter-level and contract-level personalization.

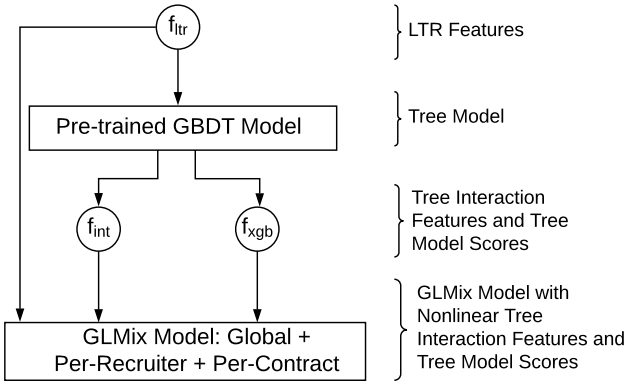


Figure 1: Pipeline for GLMix models with nonlinear tree interaction features.

3.4 Implementation Details

In this section, we summarize the implementation details of the specific GLMix model and GBDT model training algorithms used, as well as the utilized system to combine those into a model training workflow. For GLMix, the open-sourced Photon-ML library [21] was used, and for GBDTs, the open-sourced XGBoost [2] library was used. Both are built on top of the Apache Spark distributed training architecture[2]. Our training implementation used HDFS as the data storage platform, and YARN as the resource manager for Apache Spark.

The Photon-ML algorithm for GLMix model training works as follows. The input data is cloned to create a unique dataset for each fixed and random effect component of the GLMix model. The data in each set is identical, except for the feature vectors, which contain only the features used by that particular fixed/random effect. The data in each dataset is distributed amongst the Spark nodes used for training. The component models are trained one-by-one, starting with the fixed effect model, followed by random-effect models. After each component model is updated, residual scores are computed for all data samples using the updated model. These scores are used to offset the raw scores of successive random effect models during training. Note that the score is the value produced by a GLM prior to the application of any link function [27].

The XGBoost distributed algorithm works as follows. The input data is distributed amongst the Spark nodes in groupings known as blocks. XGBoost then runs a standard gradient tree boosting algorithm, with some modifications. Since the data is distributed, the exact greedy splitting algorithm cannot be used. Instead, XGBoost uses an approximate algorithm which proposes candidate split points according to the percentiles of feature distribution, computed using a weighted quantile sketch algorithm. The features are mapped into buckets along the proposed split points, and XGBoost chooses the best solution from amongst the proposals. XGBoost will also determine a default direction during splitting, used when feature data is missing [21].

In our Talent Search production system, we generate this hybrid modeling approach of entity-level personalization with tree interaction features as follows. In an offline pipeline, we generate a single day worth of training dataset using recruiter search impressions every day, hence we have a day-by-day separated dataset that accumulates over time. Then, we use 45-days of data for training, and one-day of data, immediately following the training set time period, for testing. During training / test data generation step, we also utilize a pre-trained GBDT model trained using XGBoost, in order to generate tree model score and tree interaction features for each search impression. After training and test data are generated, we build the personalized GLMix model on the 45-day training set, and test it on the one-day test set to check for and prevent extreme model performance drifts. We update the GLMix model every day, in order to personalize the model for new entities, recruiters, and contracts. Then, we upload the GLMix global model, per-recruiter model, and per-contract model coefficients into a key-value store for online utilization. More details about the online architecture of this production system are included in *System Architecture* section.

4 SYSTEM ARCHITECTURE

In this section, we first present the offline pipeline we utilize to generate and store GLMix models. Then, we present the online candidate recommendation architecture, which retrieves candidates matching a user query and applies personalized ranking via a GLMix model.

4.1 Offline Architecture

In Figure 2, we present the offline computation details of our layered GLMix model, previously introduced in Figure 1. We first collect the impression data, as well as the action data from the usage of the Recruiter product. This data mainly contains the following information:

[2] https://spark.apache.org

- Raw features collected, at the time of impression, for each candidate recommended to the users of the product[3].
- Recruiter's feedback to the recommended candidates, and the candidate responses to them, as labels. This label is positive/1 if a message was sent to the candidate, and the candidate responded to the message positively. This way, we are able to model two-way interest between recruiters and candidates.

To deal with presentation bias [12], impression and action data is collected from a subset of search query instances where we randomize the order of candidates that are recommended. Later, we feed our dataset into our pre-trained GBDT models[4] [5] to generate the tree interaction features and add them to the feature vector of each training record.

We use a two-level ranking system in online architecture, which we match in the offline architecture. In the first level (L1), top K=1000 candidates are selected by model, and in the second level, a subset of 125 candidates from this L1 ranking result is selected by another model. For our hybrid GLMix model with tree interaction features, we use XGBoost model scores generated in L1, and tree interaction features generated in L2. Note that at each ranking level, available raw LTR feature set is different, that is, L1 XGBoost model generating XGBoost model score and L2 XGBoost model generating tree interaction features are different models.

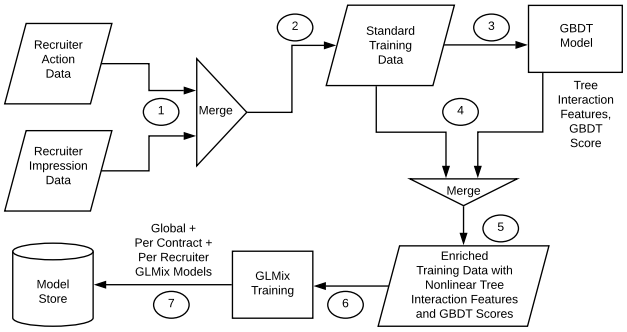


Figure 2: Offline Architecture for generating GLMix models. The numbers denote the order in the workflow.

Once we have our enriched dataset with raw features, tree interaction features, and XGBoost model scores, we train a GLMix model [27] with global, per-recruiter, and per-contract model components (the number of personalized models is on the order of tens of thousands). The overall training process takes between 3 and 5 hours. We currently run this pipeline daily to capture the most recent recruiter and contract data, and to incorporate the latest preferences into the personalized models. Once the GLMix model is learned, the coefficients are uploaded to an online key-value store, to be used during the online serving of candidates.

4.2 Online Architecture

Figure 3 presents the online ranking architecture for candidates which is a two-level system. The first level retrieves and returns

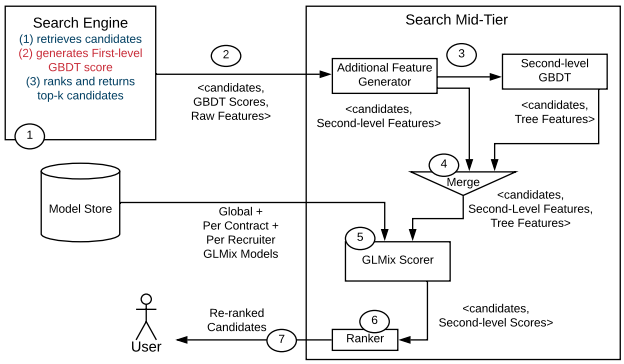


Figure 3: Online Architecture for Ranking with GLMix models.

top-k candidates using a GBDT model. The second level utilizes our proposed GLMix model picked up from the model storage (a key-value store) to re-rank the top candidates with recruiter and contract-level personalization.

At query time, the search mid-tier fetches the personalized models from the key-value store and performs the following steps:

(1) Gets the search request from the front-end (user),
(2) Sends the query belonging to the search request into the search engine. The search engine performs the retrieval and ranking using a GBDT model (trained via XGBoost) and returns top-k candidates along with their scores and computed features. These are the first-level scores and features,
(3) Generates second-level features: There are certain features which are available only in second-level ranking due to performance constraints. Search mid-tier generates these features for the candidates returned in the first-level response,
(4) Generates tree interaction features: Tree interaction features are generated using the GBDT model in the mid-tier, and added to the second-level feature set. The model has to be the same as the one utilized in the offline pipeline to train GLMix models,
(5) Generates the second-level score using the GLMix model,
(6) Re-ranks candidates based on the second-level score, and,
(7) Sends the ranked list of candidates to the front-end (user).

5 EXPERIMENT RESULTS

In order to validate the performance of GLMix models with nonlinear tree interaction features, we ran experiments in Recruiter Search domain. The key evaluation metric is precision@k, more specifically, *Positive Responses*@k[5], for values $k \in \{1, 5, 25\}$, which is averaged over all search requests. In offline experiments, we benchmarked the GLMix model variants against a pointwise GBDT / XGBoost model since that was the online baseline at the time of the offline experiments. In online experiments, we benchmarked GLMix model variants against a pairwise GBDT / XGBoost model,

[3] We are unable to reveal our raw feature list due to company policy.
[4] While we utilize XGBoost [2] to train the utilized GBDT models, the details of how they are trained, features as well as hyper-parameters such as depth and number of trees, are out of the scope of this paper.

[5] Average number of impressions that ended in the recruiter communicating with the candidate for the opportunity, and the candidate responding positively, for the first k recommended candidates.

Model (Type, features)	Precision@1	Precision@5	Precision@25
Pointwise GBDT (Baseline)	-	-	-
GLMix Global, LTR + GBDT Score	+7.352%	+3.637%	+0.766%
GLMix Global + Contract, LTR + GBDT Score	+7.688%	+2.609%	+0.683%
GLMix Global + Contract + Recruiter, LTR + GBDT Score	+5.429%	+2.267%	+0.683%
GLMix Global, LTR + GBDT Score & Interaction Features	+4.753%	+3.185%	+1.144%
GLMix Global + Contract, LTR + GBDT Score & Interaction Features	+5.909%	+3.941%	+1.594%
GLMix Global + Contract + Recruiter, LTR + GBDT Score & Interaction Features	+8.506%	+4.742%	+2.010%

Table 1: Offline experiment results benchmarking GLMix model variants against GBDT/XGBoost model, based on *Precision*@k (*Positive Responses*@k) values. The values are the lifts in Precision@k values, raw Precision@k values not shown due to company policy.

since the baseline had moved between the original offline experimentation and online productization. Note that, other methods have been benchmarked in Talent Search domain in the past, and we use the best performing one, pointwise / pairwise XGBoost models, as baseline models in our experiments [5] [23].

5.1 Offline Experiments

In offline experiments, we benchmarked GLMix model variants against the production model at the time, which is a pointwise GBDT/XGBoost model using learning-to-rank features. We tested GLMix model variants including:

(1) GLMix global model,
(2) GLMix global + per-contract model,
(3) GLMix global + per-contract + per-recruiter model.

For GLMix model variants, we used first-level GBDT model scores and second-level GBDT model tree interaction features, optimizing for Precision@25 (the default number of results on the first page is 25 for recruiters). We employed the grid search approach for hyper-parameter selection (i.e. selection of regularization weights for global and per-entity models). Table 1 shows the comparison of Precision@k values of pointwise GBDT model and GLMix model variants. Based on these offline experiments, we observed 8.5% / 4.7% / 2% lift in Precision @1 / @5 / @25 with the best-performing GLMix model variant, which is GLMix global + per-contract + per-recruiter model using LTR features, GBDT model score, and tree interaction features. Given that we optimize for Precision@25 (first page) for GLMix model when running hyper-parameter optimization, the performance of "GLMix Global + Contract + Recruiter, LTR + GBDT Score & Interaction Features" model is better compared to all other GLMix variants, which is also the case for Precision@1 and Precision@5. This is also the GLMix model variant which we ran our online experiments with, and our A/B test results are presented in the next section.

5.2 Online Experiments

To better understand the effect of personalization on the quality of candidates recommended to the recruiters, we have also performed real-world online tests on our LinkedIn Recruiter users, which are in the order of hundreds of thousands (and we recommend them candidates in the order of tens of millions). We are reporting the results of our A/B test which was run for around three weeks within the months of May and June 2018 in Table 2. In

Metric	Improvement over Baseline	p-value
1-day Positive Response Rate	+2.3%	0.03
3-day Positive Response Rate	+3%	0.01
7-day Positive Response Rate	+2.7%	0.01

Table 2: Online Results of our Three Week A/B Test

the presented results, 1-day positive response rate is the improvement on the percentage of communication requests sent by the recruiters that were positively responded to by the candidates (i.e. candidates were interested in the opportunity), within the first day (first three days for 3-day metric, and first seven days for 7-day metric) of sending the request. Online experiment results in this table show that GLMix model with tree interaction features results in statistically significant improvements of 2.3%, 3%, and 2.7% on the 1-day, 3-day, and 7-day positive response rates, respectively, compared to the baseline pairwise GBDT/XGBoost model (as we have previously mentioned, the baseline GBDT model shifted from pointwise to pairwise during the implementation efforts for GLMix in production). Per our experience, these metrics are quite hard to move, and we can clearly see the advantages of personalization by matching the most suitable candidates based on the preferences of the recruiter and contract. The difference between the online and offline results in terms of the lifts, as presented in Tables 1 and 2, is due to the change of baseline model.

6 CONCLUSION AND FUTURE WORK

In this article, we proposed the utilization of GLMix models with tree interaction features from gradient boosted decision trees (GBDT) for Talent Search, combining the benefits of entity-level personalization and nonlinear feature interactions. We have designed an offline model training and experimentation framework, as well as an online system to employ these GLMix models in production. Our offline experiments show that the best GLMix model variant, a global + per-contract + per-recruiter model using learning-to-rank (LTR) features, GBDT model score, and GBDT model tree interaction features, outperforms the baseline pointwise GBDT model, as well as all other GLMix model variants. Our online experiments also show that the best GLMix model variant improves positive

response rate with statistically significant difference over the baseline pairwise GBDT model. This method can be extended to any application in any domain where the goal is to provide personalized recommendations with nonlinear feature interactions.

For future work, one potential venue we would like to explore is to improve the hyper-parameter selection for GLMix training using Bayesian hyper-parameter optimization [24] in the daily training pipeline, compared to the currently employed grid-search approach. We also are looking into the possibility of training pairwise GLMix models instead of pointwise approach, based on our observation of high performance of pairwise GBDT models, and also due to the need to match the loss function type between our current baseline pairwise GBDT model and the GLMix model in our proposed hybrid-model solution. We also plan to replace pre-trained GBDT model with deep neural models as the nonlinear feature generator that we can feed into personalized GLMix models, which will generate wide-and-deep personalized recommendations for Talent Search.

REFERENCES

[1] Justin Basilico and Thomas Hofmann. 2004. Unifying Collaborative and Content-based Filtering. In *Proceedings of the Twenty-first International Conference on Machine Learning (ICML '04)*. ACM, New York, NY, USA, 9–. https://doi.org/10.1145/1015330.1015394

[2] T. Chen and C. Guestrin. 2016. XGBoost: A scalable tree boosting system. In *KDD*.

[3] Evangelia Christakopoulou and George Karypis. 2016. Local Item-Item Models For Top-N Recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. ACM, New York, NY, USA, 67–74.

[4] Thomas G. Dietterich. 2000. Ensemble Methods in Machine Learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems (MCS '00)*. Springer-Verlag, London, UK, UK, 1–15. http://dl.acm.org/citation.cfm?id=648054.743935

[5] S. C. Geyik, Q. Guo, B. Hu, C. Ozcaglar, K. Thakkar, X. Wu, and K. Kenthapadi. 2018. Talent Search and Recommendation Systems at LinkedIn: Practical Challenges and Lessons Learned. In *SIGIR*.

[6] Viet Ha-Thuc, Ganesh Venkataraman, Mario Rodriguez, Shakti Sinha, Senthil Sundaram, and Lin Guo. 2016. Personalized Expertise Search at LinkedIn. *CoRR* abs/1602.04572 (2016). arXiv:1602.04572 http://arxiv.org/abs/1602.04572

[7] Viet Ha-Thuc, Ye Xu, Satya Pradeep Kanduri, Xianren Wu, Vijay Dialani, Yan Yan, Abhishek Gupta, and Shakti Sinha. 2016. Search by Ideal Candidates: Next Generation of Talent Search at LinkedIn. In *Proceedings of the 25th International Conference Companion on World Wide Web (WWW '16 Companion)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 195–198. https://doi.org/10.1145/2872518.2890549

[8] Viet Ha-Thuc, Yan Yan, Xianren Wu, Vijay Dialani, Abhishek Gupta, and Shakti Sinha. 2017. From Query-By-Keyword to Query-By-Example: LinkedIn Talent Search Approach. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. ACM, New York, NY, USA, 1737–1745. https://doi.org/10.1145/3132847.3132869

[9] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer New York Inc., New York, NY, USA.

[10] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quiñonero Candela. 2014. Practical Lessons from Predicting Clicks on Ads at Facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising (ADKDD'14)*. ACM, New York, NY, USA, Article 5, 9 pages. https://doi.org/10.1145/2648584.2648589

[11] K. Jarvelin and J. Kekalainen. 2002. Cumulated Gain-based Evaluation of IR Techniques. *ACM Trans. on Information Systems (TOIS)* (2002).

[12] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. 2005. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR*.

[13] Yehuda Koren. 2009. The BellKor Solution to the Netflix Grand Prize.

[14] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. 2011. Content-based Recommender Systems: State of the Art and Trends. In *Recommender Systems Handbook*, Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor (Eds.). Springer, 73–105. http://dblp.uni-trier.de/db/reference/rsh/rsh2011.html#LopsGS11

[15] Zhongqi Lu, Zhicheng Dou, Jianxun Lian, Xing Xie, and Qiang Yang. 2015. Content-based Collaborative Filtering for News Topic Recommendation. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, 217–223. http://dl.acm.org/citation.cfm?id=2887007.2887038

[16] Marlin, Benjamin M. 2004. Modeling User Rating Profiles For Collaborative Filtering. In *Advances in Neural Information Processing Systems 16*, S. Thrun, L. K. Saul, and B. Schölkopf (Eds.). MIT Press, 627–634. http://papers.nips.cc/paper/2377-modeling-user-rating-profiles-for-collaborative-filtering.pdf

[17] Charles E. McCulloch. 2003. *Chapter 4: Generalized linear mixed models (GLMMs)*. Regional Conference Series in Probability and Statistics, Vol. Volume 7. Institute of Mathematical Statistics and American Statistical Association, Beechwood OH and Alexandria VA, 28–33. https://projecteuclid.org/euclid.cbms/1462106064

[18] Prem Melville, Raymod J. Mooney, and Ramadass Nagarajan. 2002. Content-boosted Collaborative Filtering for Improved Recommendations. In *Eighteenth National Conference on Artificial Intelligence*. American Association for Artificial Intelligence, Menlo Park, CA, USA, 187–192. http://dl.acm.org/citation.cfm?id=777092.777124

[19] Biswanath Panda, Joshua S. Herbach, Sugato Basu, and Roberto J. Bayardo. 2009. PLANET: Massively Parallel Learning of Tree Ensembles with MapReduce. In *Proceedings of the 35th International Conference on Very Large Data Bases (VLDB-2009)*. http://www.bayardo.org/ps/vldb2009.pdf

[20] Yubin Park and Joyce C. Ho. 2018. PaloBoost: An Overfitting-robust TreeBoost with Out-of-Bag Sample Regularization Techniques. *CoRR* abs/1807.08383 (2018). https://arxiv.org/abs/1807.08383

[21] Photon-ML. 2016. *Photon-ML GitHub page*. https://github.com/linkedin/photon-ml.

[22] Anand Rajaraman and Jeffrey David Ullman. 2011. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA.

[23] Rohan Ramanath, Hakan Inan, Gungor Polatkan, Bo Hu, Qi Guo, Cagri Ozcaglar, Xianren Wu, Krishnaram Kenthapadi, and Sahin Cem Geyik. 2018. Towards Deep and Representation Learning for Talent Search at LinkedIn. In *CIKM*.

[24] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems (NIPS)*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2951–2959. http://papers.nips.cc/paper/4522-practical-bayesian-optimization-of-machine-learning-algorithms.pdf

[25] Stephen Tyree, Kilian Q. Weinberger, Kunal Agrawal, and Jennifer Paykin. 2011. Parallel Boosted Regression Trees for Web Search Ranking. In *Proceedings of the 20th International Conference on World Wide Web (WWW '11)*. ACM, New York, NY, USA, 387–396. https://doi.org/10.1145/1963405.1963461

[26] Jerry Ye, Jyh-Herng Chow, Jiang Chen, and Zhaohui Zheng. 2009. Stochastic Gradient Boosted Distributed Decision Trees. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM '09)*. ACM, New York, NY, USA, 2061–2064. https://doi.org/10.1145/1645953.1646301

[27] XianXing Zhang, Yitong Zhou, Yiming Ma, Bee-Chung Chen, Liang Zhang, and Deepak Agarwal. 2016. GLMix: Generalized Linear Mixed Models For Large-Scale Response Prediction.. In *KDD*. ACM, 363–372.

Value-aware Recommendation based on Reinforcement Profit Maximization

Changhua Pei*
Alibaba Group
changhua.pch@alibaba-inc.com

Xinru Yang*◇
Carnegie Mellon University
xyang@andrew.cmu.edu

Qing Cui
Alibaba Group
cuiqing.cq@alibaba-inc.com

Xiao Lin
Alibaba Group
hc.lx@alibaba-inc.com

Fei Sun
Alibaba Group
ofey.sunfei@gmail.com

Peng Jiang
Alibaba Group
jiangpeng.jp@alibaba-inc.com

Wenwu Ou
Alibaba Group
santong.oww@taobao.com

Yongfeng Zhang§
Rutgers University
yongfeng.zhang@rutgers.edu

ABSTRACT

Existing recommendation algorithms mostly focus on optimizing traditional recommendation measures, such as the accuracy of rating prediction in terms of RMSE or the quality of top-k recommendation lists in terms of precision, recall, MAP, etc. However, an important expectation for commercial recommendation systems is to improve the final revenue/profit of the system. Traditional recommendation targets such as rating prediction and top-k recommendation are not directly related to this goal.

In this work, we blend the fundamental concepts in online advertising and micro-economics into personalized recommendation for profit maximization. Specifically, we propose value-aware recommendation based on reinforcement learning, which directly optimizes the economic value of candidate items to generate the recommendation list. In particular, we generalize the basic concept of click conversion rate (CVR) in computational advertising into the conversation rate of an arbitrary user action (XVR) in E-commerce, where the user actions can be clicking, adding to cart, adding to wishlist, etc. In this way, each type of user action is mapped to its monetized economic value. Economic values of different user actions are further integrated as the reward of a ranking list, and reinforcement learning is used to optimize the recommendation list for the maximum total value. Experimental results in both offline benchmarks and online commercial systems verified the improved performance of our framework, in terms of both traditional top-k ranking tasks and the economic profits of the system.

CCS CONCEPTS

• **Information systems → Recommender systems**.

*Equal contribution. Listing order is random.
◇This work was done when Xinru Yang was an intern at Alibaba.
§Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313404

KEYWORDS

Recommender Systems; Reinforcement Learning; Economics of Data Science

ACM Reference Format:
Changhua Pei, Xinru Yang*, Qing Cui, Xiao Lin, Fei Sun, Peng Jiang, Wenwu Ou, and Yongfeng Zhang. 2019. Value-aware Recommendation based on Reinforcement Profit Maximization. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313404

1 INTRODUCTION

Recommender system has become a fundamental service in many online applications. Years of research have witnessed the great advancement on the development of recommendation systems, and various different techniques have been proposed to support better performance in practice, including but not limited to content-based methods [23], user/item-based collaborative filtering methods [8], matrix factorization techniques [17], and the more recent deep learning-based approaches [33]. Most of the existing methods focus on two of the most fundamental tasks, namely, rating prediction and top-k recommendation. They usually aim at the optimization of rating or ranking-related measures, such as root mean square error (RMSE), mean average precision (MAP), and many others.

However, an important and sometimes the ultimate goal of commercial recommendation system is to gain revenue and profits for the platform, but traditional recommendation research are not directly related to this goal. They mostly focus on whether or not the algorithm can predict accurate ratings (rating prediction), or if the system can rank the clicked items correctly (top-k recommendation). Nevertheless, researchers have shown that improved performance on rating prediction does not necessarily improve the top-k recommendation performance in terms of user purchase [5]. And even better performance on top-k recommendation does not necessarily improve the profit of the system [34], because the purchased recommendations may not convert into the best profit due to their difference in price. For example, by recommending daily necessities, the system has a better chance of getting the user to purchase the recommendation, but the expected profit achieved from the purchase may be smaller than if a luxury good was purchased, though the latter has a smaller chance of being purchased.

In this work, we propose value-aware recommendation for profit maximization, which directly optimizes the expected profit of a recommendation list. The basic idea is to balance the unit profit of a recommendation and the chance that the recommendation is purchased, so that we can find the recommendation list of the maximum expected profit. To achieve this goal, we generalize the key concept of click conversion rates (CVR) [1, 19] in online advertising to the conversion rate of arbitrary user actions (XVR), which scales each user action into the monetized profit of the system based on large-scale user behavior logs.

Once the user actions are converted into expected profits, we further design an aggregated reward function based on the monetized user actions, and then adopt reinforcement learning to optimize the recommendation list towards the maximized expected profit for the system. For training, we developed a benchmark dataset by collecting the user behaviors in a real-world E-commerce system. Furthermore, we conducted online experiments on this E-commerce system to validate the online performance of value-aware recommendation in terms of both ranking performance and system profits.

The key contributions of the paper can be summarized as follows:

- We propose *value-aware recommendation* to maximize the profit of a recommendation system directly, which to the best of our knowledge, is the first time to explicitly optimize the profit of the personalized recommendation list in large-scale online system.
- We propose to generalize the basic concept of click conversation rate into the conversion rate of arbitrary user actions, so that each user action can be monetized into the profit of the system.
- By aggregating the monetized user actions into a unified profit reward, we develop a reinforcement learning-based algorithm for value-aware recommendation.
- We conduct both offline experiments with benchmark datasets and online experiments with real-world commercial users to validate the performance of value-aware recommendation on ranking and profit maximization.

2 RELATED WORK

In this section, we introduce the key related work in terms of three perspectives: economic recommendation, computational advertising, and reinforcement learning.

2.1 Economic Recommendation

For a long time, recommendation system research has been working on rating- or ranking-related tasks such as rating prediction, top-k recommendation, sequential recommendation, etc., and a lot of successful models have been proposed, which greatly advanced the performance of recommendation systems. To name a few, this includes content-based methods [23], user/item-based collaborative filtering [8], shallow latent factor models such as matrix factorization [17, 18, 22, 24], and more recent deep learning based methods [14, 31, 33]. However, the related methods seldom consider the economic value/profit that a recommendation list brings about to the system, although this is one of the most important goals for real-world commercial recommender systems. Some recent research on economic recommendation has begun to take care of the economic value of recommendation[3, 7, 39]. For example, Zhang et al. [34] proposed to maximize social surplus for recommendation, Zhao et al. [35] proposed to learn the substitutional and complementary relations between items for utility maximization, and Ge et al. [10] further proposed to maximize the marginal utility per dollar for recommendation to benefit user preference. However, none of the above methods explicitly maximizes the expected profits of a recommendation list to generate recommendations.

2.2 Computational Advertising

Currently, computational advertising [6] has been playing a crucial role in online shopping platforms, rendering it essential to apply an intelligent advertising strategy that can maximize the profits by ranking numerous advertisements in the best order [2, 11, 12]. In this scenario, the goal of advertisers is to maximize clicks and conversions such as item purchase. Therefore, cost-per-click (CPC) and cost-per-action (CPA) models are often adopted to optimize the click-through-rate (CTR) and the conversion-rate (CVR), respectively. Although technically there are a lot of analogy between advertising and recommendation, there is not much work trying to bridge these two principles. This work is one of our first attempts to integrate the economic consideration in adversting into recommender systems for value-aware recommendation.

2.3 Reinforcement Learning

Reinforcement Learning (RL) aims at automatically learning an optimal strategy from the sequential interactions between the agent and the environment by trial-and-error [29]. Some existed works have made efforts on exploiting reinforcement learning for recommendation systems. For instance, conventional RL methods such as POMDP (Partially Observable Markov Decision Processes) [27] and Q-learning [30] estimate the transition probability and store the Q-value table in modeling. However, they soon become inefficient with the sharp increase in the number of items for recommendation. Meanwhile, other approaches have been proven useful with an effective utilization of deep reinforcement learning. For instance, [38] employs Actor-Critic framework to learn the optimal strategy by a online simulator. Furthermore, [37] considers both positive and negative feedback from users recent behaviors to help find optimal strategy. [9] uses the multi-agent reinforcement learning to optimize the multi-scenario ranking. Meanwhile, [36] adopts RL to recommend items on a 2-D page instead of showing one single item each time.

To mitigate the performance degradation due to high-variance and biased estimation of the reward, [4] provides a stratified random sampling and an approximate regretted reward to enhance the robustness of the model. Similarly, [16] introduces DPG-FBE algorithm to maintain an approximate model of the environment to perform reliable updates of value functions.

3 GENERALIZED CONVERSION RATE

In order to maximize the profit of recommender system, we need to measure the the value of different user actions on the platform. In this section, we introduce how to generalize the basic concept of click conversion rate to rescale each user action into monetized profit. Here we use Gross Merchandise Volume (GMV) to measure the profit of the system. GMV is a measure used in online retailing, which indicates the total sales dollar value of the merchandise sold

through a particular marketplace over a certain time frame[1]. We use GMV and profit interchangeably aftermentioned.

3.1 Conversion Rate of Clicks

Click conversion rate (CVR) is widely used in online advertising to estimate the potential profit of an advertising item to the advertising platform [21, 40]. It is usually defined as the result of dividing the number of conversions by the number of total clicks that can be tracked to a conversion during the same period of time. In E-commerce system, it is difficult to learn an optimal recommendation strategy by simply calculating the profits of transactions because purchasing an item is a relatively sparse action of users. Instead, users tend to click an item much more frequently. CVR provides a way to map the dense action of clicking into a scaled range of profits. For E-commerce system, CVR is the first step towards value-aware recommendation[28, 32]. Eq, 1 shows the total estimated GMV brought by click, which shows that CVR plays a crucial role in profit maximization,

$$E[GMV] = \sum_i I(click, i) \cdot CVR(i) \cdot price(i) \tag{1}$$

where price(i) is the unit price of item i, $I(click, i)$ is an indicator of the occurrence of click on item i. If item i is clicked, $I(click, i) = 1$, otherwise, 0.

3.2 Generalization to other Actions

In E-commerce platform, a series of behaviors of a user is often formed step by step with transforming probabilities. In Figure **??**, the general shopping process is simplified into four steps. First, the user sees something of interests and gets an impression on the item, then the user will click to see more details and to eventually add it to the cart. The user can also add it to the wishlist so he can add it to the cart later. Eventually, some items in the cart will be purchased together and contribute to final profits. Adding to wishlist and adding to cart reveal a strong desire for the item even if no transaction is made eventually. Therefore, these actions are also considered to contribute to the value.

Here we generalize the concept of CVR to XVR to map each type of user action x to its monetized economic value. XVR is defined as the possibility of transition from certain action x to purchase. Using XVR, we can estimate the total profits of different actions on platform. By generalizing Eq. 1, we get Eq. 2,

$$E[GMV] = \sum_{x,i} V(x, i) = \sum_{x,i} I(x, i) \cdot XVR(i) \cdot price(i) \tag{2}$$

where E[GMV] denotes the expected GMV of the platform, x represents certain action including *click*, *add to cart*, *add to wishlist*, and *purchase*. Here $V(x, i)$=$I(x, i) \cdot XVR(i) \cdot price(i)$ is the expected profit given user has an action x.

4 REINFORCED PROFIT OPTIMIZATION

The main goal of this work is to maximize the total profit of recommender system in an E-commerce scenario. Here the profit indicate a total sales dollar value for merchandise sold online in a certain time frame. Using the generalized conversation rate, we can transfer all user actions to monetized profit and maximize it. Reinforcement learning (RL) aims to learn a policy that maximizes the accumulated reward. The profit can naturally be used as the reward with which we can find better actions (ranking policy). Besides, the view of interact between the environment (users) and RL agent fits our recommendation scenario well. Therefore, in this paper, we choose reinforcement learning, evolution strategy more specifically[25], as our algorithm for its simplicity and effectiveness.

[1]https://en.wikipedia.org/wiki/Gross_merchandise_volume

4.1 Reinforcement Learning Modeling

Figure 2 illustrates how our value-based RL agent interacts with the user to maximize the profit. First, the user comes to the system at some time and makes an initial request, the system receives some information about the user as well as the candidate items as a state and responds to the request, which virtually means taking an action by generating an order to rank items. Next, the user shows a certain behaviors regarding the items such as click, add to wishlist, add to cart or purchase, which is used to calculate the reward to the system for what kind of action it takes.

In our recommendation platform, items are shown in cascade on a mobile App one by one. Each time the user initiates a request, 50 items are recommended to him/her. As user scrolls down the list and have seen all 50 items, a new request is triggered. This process is repeated until the user leaves the App or return the top of the cascade, labelled as "exit" in Figure 2. We use a metric called "pageid" to distinguish different requests in this interaction, similar to the concept of "page" to a search engine. As the user and the system interact with each other, the system learns how to respond to the state to obtain an optimized accumulative reward.

States: States are used to describe an user's request, including user-side, item-side, and contextual features. The user-side features include *age*, *gender*, *purchase power*. The item-side features are *ctr*, *cvr*, *price*. The contextual features include *pageid*, *request time*. In total, the states in our model can be formulated as <*age*, *gender*, *purchase power*, ctr_0, ..., ctr_{49}, cvr_0, ..., cvr_{49}, $price_0$, ..., $price_{49}$, *pageid*, *request time*>, the subscript i denotes the corresponding features of i^{th} item in the corresponding page.

All these features work together to influence the final GMV. For example, purchase power is calculated by how many and how expensive the bought items are. The higher the level of purchase power is, the more money is spent on the platform. Fig. 3a, 3b, and 3c show average GMV per user, number of users and summed GMV under different purchasing power and degree of activity. Degree of activity indicates the time a user spends on the platform. Fig. 3a shows users with higher purchase power have higher average GMV. Thus the model tends to recommend items with higher price to user who has high purchase power. However, for summed GMV, users with middle level of purchase power contribute most because the large number of users, as shown in Fig. 3b.

Request time (labelled as hour in Fig. 4) and pageid are also features used in our model. Fig. 4 shows that the the first page whose pageid equal to 0 contributes most to the summed GMV, as user scrolls downwards, pages with larger pageid have smaller possibilities to have impression on users. Thus the model should recommend items with higher conversion rate in the first page. Besides, model will learn different weights at different hour as Fig. 4 show the summed GMV varies with request time.

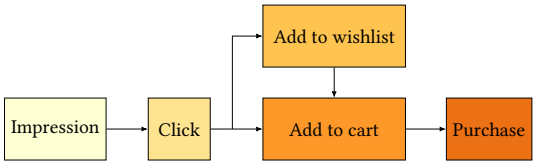


Figure 1: A flow of the user behavior series.

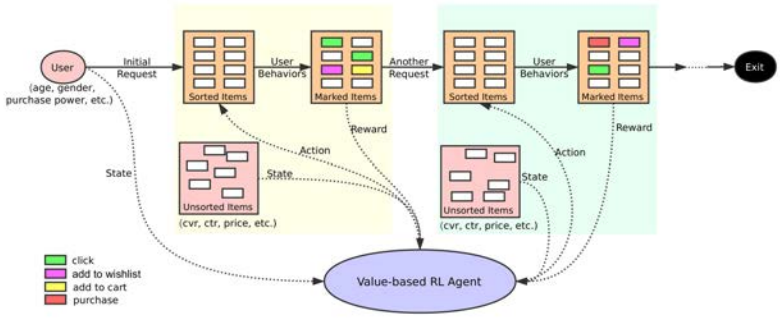
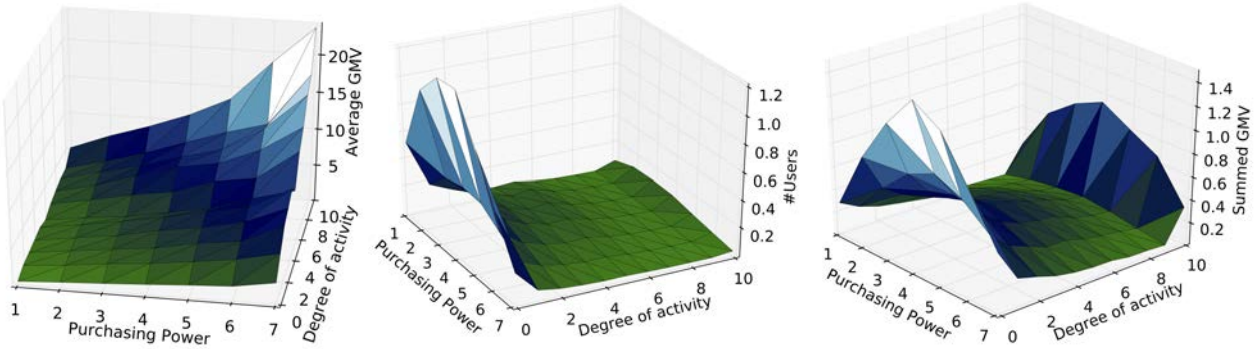


Figure 2: Structure of the model. When user initiates a request, the system will generate a recommendation list, and the user will interact with the list with certain actions. These Actions will be used to calculate the reward to update the RL Agent.



(a) Average GMV. **(b) Number of Users.** **(c) Summed GMV.**

Figure 3: Average/Summed GMV and the number of users under different level of purchasing power and degree of activity.

Actions: Action is the coefficient vector in the ranking formular which decides the order of candidate items. We design a ranking formula shown in Eq. 3 to order the items,

$$rankscore(i) = \sum_{x \in \mathcal{A}} P(x,i)^{\alpha_x} \cdot XVR(i)^{\beta_x} \cdot price(i)^{\gamma} \quad (3)$$

where $\mathcal{A}$ is the user action set which contains click, add to cart, add to wishlist. $P(x,i)$ is the possibility for a given user has action x on item i. $XVR(i)$ is the generalized conversion rate of item i. In total, the action can be formulated as $<\alpha_{click}, \alpha_{cart}, \alpha_{wishlist}, \beta_{click}, \beta_{cart}, \beta_{wishlist}, \gamma>$. If we only consider click in the action set $\mathcal{A}$, rankscore can be simplified to Eq. 4.

$$\begin{aligned} rankscore(i) &= P(click,i)^{\alpha_{click}} \cdot XVR^{\beta_{click}} \cdot price(i)^{\gamma} \\ &= ctr(i)^{\alpha_{click}} \cdot cvr(i)^{\beta_{click}} \cdot price(i)^{\gamma} \end{aligned} \quad (4)$$

Changing an action means to change the coefficient vector for the ranking formula.

Reward: In our value-based method, we use the expected profit as reward, which contains the monetized profit converted from all kinds of user actions. Based the definition of the expected GMV in Eq. 2, for a given item i, the reward R_i can be defined as:

$$R_i = V(click,i) + V(wishlist,i) + V(cart,i) + V(pay,i) \quad (5)$$

In this paper, unless specifically mentioned, we mainly use click and pay in reward, *i.e.*, $R_i = V(click,i) + V(pay,i)$. In the last section, we evaluate the performance when taking adding to cart and adding to wishlist into reward. The total reward of a recommendation list in a page that contains T items can be defined as $R_{page} = \sum_{i=0}^{T} R_i$.

4.2 Model Training

4.2.1 Offline Reward. In the above section, the reward of the model R_{page} is directly calculated with user's feedback online. This requires that the model is deployed online directly and learns the policy from the real-time data streaming. However, the online traffic is expensive. It is risky to let RL model learns directly online before we validate the effectiveness of the value-based method. To overcome this problem, we propose a simulated environment which leverage the historical data to approximate the feedback of the users. The simulated environment adopts a simple idea of NDCG (normalized discounted cumulative gain) that good policy

Table 1: Overview of the benchmark Dataset

clause	size(x10^6)
#distinct users	49
#distinct items	200
#requests	500
#clicks	670
#adding to carts	60
#adding to wishlists	30
#purchases	3

should rank the item user actually clicked and paid in the front. In this way, an offline reward R'_{page} is used to evaluate actions as following:

$$R'_{page} = \sum_{i=0}^{T} R_i * W_{\pi(i)} \tag{6}$$

For a given page with T items $1, \cdots, T$, the ranking policy can generate an order list, where $\pi(i) = k$ means item i is ranked at position k. We assign a discounted weight for each position k and higher position has greater weight, i.e. $W_{\pi(i)}$. Then we represent the reward as the weighted sum of item reward. In this paper, $W_{\pi(i)}$ is represented by exponential function which is $W_{\pi(i)} = \exp(-\pi(i))$.

4.2.2 Learning Algorithm. As the main goal of this paper is to evaluate the effectiveness of the value-aware recommendation, we choose the simplest model in reinforcement learning model family to exclude the influence of complex models. Finally we adopt evolution strategy algorithm (abbreviated ES algorithm). We train the model offline and deploy it online to perform A/B tests. The following introduces how the ES algorithm learns in the training process.

Our task is to learn a policy which maps state to action that maximizes the expected reward. To keep it simple, we use a linear model as the policy and θ is the parameters to be optimized. The ES algorithm starts from initial parameter θ_0 and then updates θ_t in the loop where t=1, 2, ... At each iteration loop t, the user's request for a new page and pass the state vector s_t to the model. The algorithm generates n baby-agent and each baby-agent j samples a set of increments ϵ_t^j from a normal distribution. Each model variant with parameter $\theta_t + \epsilon_t^j$ maps the state to an action. Each action orders the list in a different way and the user reacts to different ranking polices by clicking or purchasing certain items in the list. We calculate the reward of each policy j using Eq. 6 and labeled as $R'_{page}(j) = F(\theta_t + \epsilon_t^j)$. We collect all the rewards together and update θ_t eventually:

$$\theta_{t+1} \leftarrow \theta_t + \alpha_{lr} \frac{1}{n\sigma} \sum_{j=1}^{n} R'_{page}(j) \tag{7}$$

where σ and α_{lr} are hyper parameters, σ is the noise standard deviation and α_{lr} is the learning rate.

5 EXPERIMENTS

In this section, we first introduce the settings of our benchmark dataset used for training or offline evaluation. Then we evaluate our method both online and offline together with baselines. [2]

[2]The code and dataset of the paper are released at https://github.com/rec-agent/rec-rl.

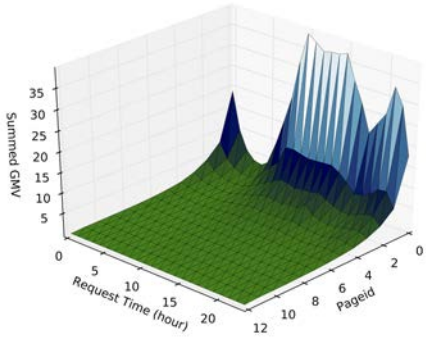


Figure 4: GMV across different pages and time.

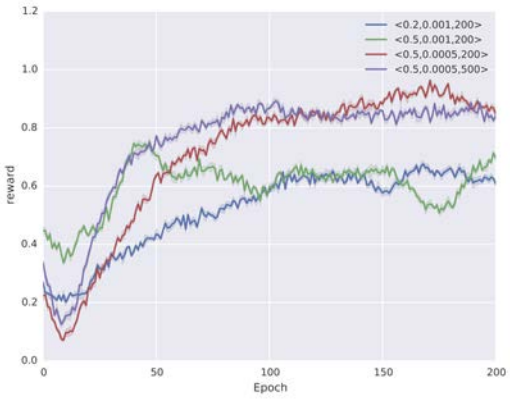


Figure 5: The training curve under different parameters: <noise standard deviation, learning rate, batch size>

5.1 Benchmark Dataset

Our benchmark dataset is collected from real-world E-commerce platform and can simulate the interactive environment and users' behaviors. Each piece of data is comprised of two parts: the *states* and users' feedback: click, add to cart, add to wishlist, purchase. Table 1 lists the details of the dataset.

5.2 Experimental Setup

5.2.1 ***Baselines****:* We use the following three methods as baselines to compare with our value-based RL method: Item-based Collaborative Filtering, LR-based learning to rank(LTR), DNN-based learning to rank.

- **Item-based CF**: The classic Item-based CF [26] is widely used for recommendation. compared to user-based CF [15], Item-based CF is more scalable and accurate in E-commerce recommendation because the relationships between items are more stable. In this paper, we use a fine-tuned item-based collaborative filtering method as one of our baselines. Other CF methods such as matrix factorization can also be used as baselines. However, as they are all non value-aware methods, we omit them in this paper for simplicity.
- **LR-based LTR**: The Point-wise Learning-To-Rank paradigm [20] is adopted to learn the rankings of recommended items with Logistic Regression(LR) as the ranking model. We use clicked and purchased items as positive samples to learn a binary classification model, and the price is used as the weight of loss. To reduce the variance of price and make the optimization stable, we actually use $\log(price)$ as the weight. For fair comparison, here

Table 2: Offline Evaluation Results (p-value < 0.005). Numbers in the brackets are improvements compared to Item-based CF and LR-based LTR.

	E[GMV]	Average R'_{page}	Precision@20(%)	Recall@20(%)	MAP@20(%)
Item-based CF	0.40	19.73	2.96	27.93	8.69
LR-based LTR	0.49 (22.5%/-)	25.87 (31.1%/-)	3.45 (16.6%/-)	32.42 (16.1%/-)	11.04 (27.0%/-)
DNN-based LTR	0.50 (25.0%/2.0%)	25.88 (31.2%/0.04%)	3.65 (23.3%/5.8%)	34.01 (21.8%/4.9%)	12.27 (41.2%/11.1%)
Value-based RL	0.53 (32.5%/8.2%)	27.78 (40.8%/7.4%)	3.74 (26.4%/8.4%)	34.82 (24.7%/7.4%)	12.36 (42.2%/12.0%)

Table 3: Online Evaluation Results (p-value < 0.005).

	GMV	CTR(%)	IPV
Item-based CF	7.57	3.04	2.48
LR-based LTR	8.95 (18.3%/-)	3.26 (7.4%/-)	2.67 (7.5%/-)
DNN-based LTR	9.06 (19.7%/1.2%)	3.28 (8.1%/0.6%)	2.69 (8.3%/0.7%)
Value-based RL	9.68 (27.9%/8.2%)	3.29 (8.2%/0.9%)	2.70 (8.8%/1.1%)

we employ the same feature set as our proposed value-based RL. There are three hyper-parameters involved in the training of LR-based LTR, *i.e.*, the ℓ_2 regularization weight, the learning rate and the batch size. These parameters are set to 0, 0.5 and 128 after tuning. We also test a repression model for predicting the profit. However, it cannot beat classification models like LR. This is consistent with previous work [13], regression model may not be suitable for ranking problems in recommender system.

- **DNN-based LTR**: This method is the same as LR-based LTR, except that the ranking model is a Deep Neural Network (DNN) instead. The neural network involves two hidden layers with 32 and 16 neurons respectively. The ℓ_2 regularization weight, the learning rate and the batch size are set to 0, 0.05 and 1024. We also tested network structures including RNN, attention networks, and Wide&Deep network. Optimizer, activation functions, normalization, and other hyper parameters are tuned by greedy search. However, the improvement of these networks is not significant on very large dataset.

5.2.2 ***Value-based RL****:.* Different hyper parameters are tried by a greedy search to find the better training performance. In ES algorithm, under the same training set, the larger reward a model can achieve, the better the model is. We name some important parameters in Fig. 5 to illustrate this. Having a larger deviation (from 0.2 to 0.5) can greatly improve the maximum reward the model can achieve. However, the model is not stable so that the reward will drop after some iterations. We adjust the learning rate in Adam optimizer from 0.001 to 0.0005 and the training curve become stable. We also try larger batch(from 200 to 500), the improvement is not that much. Finally, <0.5,0.0005,200> are used as noise standard deviation, learning rate and batch size in our RL model respectively.

5.3 Key Results

Offline Evaluation: In offline evaluation, we train the model on the data of one week and evaluate it on the data of the following week. Precision, recall and MAP within the top 20 items is used to measure the performance. Besides the traditional metrics, average R'_{page} (Eq. 6) and E[GMV] (Eq. 2) are used as value-based metrics.

Table 2 shows that the Learning-To-Rank (LTR) baseline outperforms item-based CF in all measures which indicates the LTR methods can predict the profit more effectively. We leverage price in two ways. On one hand, LTR methods use price as one of the features. On the other hand, LTR methods use price as the weight for loss in the learning process. Our value-based approach has 6.0% and 7.3% improvements in E[GMV] and R'_{page} respectively when compared to DNN-based LTR. Besides, Value-based RL achieves better performance in precision(2.5%), recall(2.4%) and MAP(0.7%) than DNN-based LTR with same features. Our approach can precisely monetize an arbitrary user action into the profit by generalizing the basic concept of click conversion rate (CVR) to CXR.

Table 4: Offline performance when considering adding to cart and adding to wishlist (p-value < 0.005).

	E[GMV]	Precision @20(%)	Recall @20(%)	MAP (%)
click	42.2	3.67	34.5	12.1
click,cart,wishlist	43.5	3.88	35.8	12.9
	(3.1%)	(5.7%)	(3.8%)	(6.6%)

Online Evaluation: An online A/B test is used to evaluate different methods. Metrics we use in A/B test are ctr, IPV and GMV. IPV is the absolute number of clicked items on the platform, which acts a supplement metric to ctr. To make the online evaluation comparable, each bucket in A/B test has the same number of users and contains over millions of users.

Table 3[3] shows the average results for one week. Online evaluation shows results are consistent with the offline evaluation. The DNN-based LTR improves both GMV, ctr and IPV by 19.7%, 8.1% and 8.3% respectively compared to Item-based CF. Even though the DNN-based LTR does not explicitly optimizing the GMV, it has 19.7% improvements in GMV compared to item-based CF. By explicitly mapping the value of different actions into the GMV, our value-based RL method can bring another 6.8% improvement on GMV compared to DNN-based LTR. The improvements on ctr and IPV is 0.3% and 0.4% respectively. Online test is done in real-world system (involving 2 hundred million clicks per day, 7 days). Based on significance test (p<0.005), the improvement are all significant and have business values for E-commercial platform, because every thousandth of improvement on ctr means hundreds of millions of extra clicks in real-world systems.

5.4 Click vs. Add to Cart and Add to Wishlist

In this part, we take adding to cart and adding to wishlist into consideration to evaluate the performance of our value-based model. Results in Table 4 show offline performance when considering the value of adding to cart and adding to wishlist actions into the model. The improvement on expected GMV is 3.1%. The improvements on precision, recall and MAP are 5.7%, 3.8% and 6.6% respectively. It

[3]Normalized related value instead of the original absolute value of the online measures are shown for business reason. The relative improvement is consistent with real experimental results.

means adding to cart and adding to wishlist actions are useful for the value-based profit maximization.

6 CONCLUSIONS AND FUTURE WORK

Great advances have been achieved by existing research to improve the accuracy of rating prediction and the quality of top-k recommendation lists. However, the economic value of recommendation is rarely studied. For large-scale commercial recommendation system, state-of-the-art algorithms seldom maximize the final revenue/profit of the system. To eliminate this gap, we propose value-aware recommendation to maximize the profit of commercial recommendation system directly. Specifically, we generalize the basic concept of click conversion rate (CVR) to the conversation rate of an arbitrary user action on the platform (XVR), where different actions of users(*click, add to cart, add to wishlist, purchase*)can be monetized into the profit of the system. Then we use reinforcement learning (RL) to maximize the profit whose reward is the aggregated monetized user actions. Both offline and online experiments show that our value-based RL model not only performs better on traditional metrics such as precision, recall and MAP, but also greatly improves the final profit than existing methods. This paper acts as the first step towards value-aware recommendation and further improvement can be achieved by designing more powerful features and RL models in the future.

REFERENCES

[1] Ashish Agarwal, Kartik Hosanagar, and Michael D Smith. 2011. Location, location, location: An analysis of profitability of position in online advertising markets. *Journal of marketing research* 48, 6 (2011), 1057–1073.

[2] Andrei Broder, Marcus Fontoura, Vanja Josifovski, and Lance Riedel. 2007. A semantic approach to contextual advertising. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 559–566.

[3] Robin Burke, Gediminas Adomavicius, Ido Guy, Jan Krasnodebski, Luiz Pizzato, Yi Zhang, and Himan Abdollahpouri. 2017. VAMS 2017: Workshop on Value-Aware and Multistakeholder Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 378–379.

[4] Shi-Yong Chen, Yang Yu, Qing Da, Jun Tan, Hai-Kuan Huang, and Hai-Hong Tang. 2018. Stabilizing reinforcement learning in dynamic environment with application to online recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

[5] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 39–46.

[6] Kushal Dave, Vasudeva Varma, et al. 2014. Computational advertising: Techniques for targeting relevant ads. *Foundations and Trends® in Information Retrieval* 8, 4–5 (2014), 263–418.

[7] Gediminas Adomavicius Dietmar Jannach. 2017. Price and Profit Awareness in Recommender Systems. *arXiv preprint arXiv:1801.00209* (2017).

[8] Michael D Ekstrand, John T Riedl, Joseph A Konstan, et al. 2011. Collaborative filtering recommender systems. *Foundations and Trends® in Human–Computer Interaction* 4, 2 (2011), 81–173.

[9] Jun Feng, Heng Li, Minlie Huang, Shichen Liu, Wenwu Ou, Zhirong Wang, and Xiaoyan Zhu. 2018. Learning to Collaborate: Multi-Scenario Ranking via Multi-Agent Reinforcement Learning. In *WWW*. 1939–1948.

[10] Yingqiang Ge, Shuyuan Xu, Shuchang Liu, Shijie Geng, Zuohui Fu, and Yongfeng Zhang. 2019. Maximizing Marginal Utility per Dollar for Economic Recommendation. In *WWW*.

[11] Anindya Ghose and Sha Yang. 2009. An empirical analysis of search engine advertising: Sponsored search in electronic markets. *Management science* 55, 10 (2009), 1605–1622.

[12] Avi Goldfarb and Catherine Tucker. 2011. Online display advertising: Targeting and obtrusiveness. *Marketing Science* 30, 3 (2011), 389–404.

[13] Asela Gunawardana and Guy Shani. 2009. A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research* 10, Dec (2009), 2935–2962.

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[15] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. 2017. An algorithmic framework for performing collaborative filtering. In *ACM SIGIR Forum*, Vol. 51. ACM, 227–234.

[16] Yujing Hu, Qing Da, Anxiang Zeng, Yang Yu, and Yinghui Xu. 2018. Reinforcement Learning to Rank in E-Commerce Search Engine: Formalization, Analysis, and Application. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[18] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*. 556–562.

[19] Kuang-chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. 2012. Estimating conversion rate in display advertising from past erformance data. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 768–776.

[20] Tie-Yan Liu. 2009. Learning to Rank for Information Retrieval. *Found. Trends Inf. Retr.* 3, 3 (March 2009), 225–331.

[21] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire Space Multi-Task Model: An Effective Approach for Estimating Post-Click Conversion Rate. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 1137–1140.

[22] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*.

[23] Michael J Pazzani and Daniel Billsus. 2007. Content-based recommendation systems. In *The adaptive web*. Springer, 325–341.

[24] Steffen Rendle and Christoph Freudenthaler. 2014. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 273–282.

[25] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. 2017. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864* (2017).

[26] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[27] Guy Shani, David Heckerman, and Ronen I Brafman. 2005. An MDP-based recommender system. *Journal of Machine Learning Research* 6, Sep (2005), 1265–1295.

[28] SIGKDD. 2015. Life-stage Prediction for Product Recommendation in E-commerce. ACM, 1879–1888.

[29] Richard S Sutton, Andrew G Barto, et al. 1998. *Reinforcement learning: An introduction*. MIT press.

[30] Nima Taghipour and Ahmad Kardan. 2008. A hybrid web recommender system based on q-learning. In *Proceedings of the 2008 ACM symposium on Applied computing*. ACM, 1164–1168.

[31] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[32] Qiaolin Xia, Peng Jiang, Fei Sun, Yi Zhang, Xiaobo Wang, and Zhifang Sui. 2018. Modeling Consumer Buying Decision for Recommendation Based on Multi-Task Deep Learning. In *CIKM*. ACM, 1703–1706.

[33] Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep learning based recommender system: A survey and new perspectives. *arXiv preprint arXiv:1707.07435* (2017).

[34] Yongfeng Zhang, Qi Zhao, Yi Zhang, Daniel Friedman, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Economic recommendation with surplus maximization. In *WWW*.

[35] Qi Zhao, Yongfeng Zhang, Yi Zhang, and Daniel Friedman. 2017. Multi-product utility maximization for economic recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 435–443.

[36] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018. Deep Reinforcement Learning for Page-wise Recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM.

[37] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendation with Negative Feedback via Pairwise Deep Reinforcement Learning. *Proceedings of the 24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (2018).

[38] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Dawei Yin, Yihong Zhao, and Jiliang Tang. 2017. Deep Reinforcement Learning for List-wise Recommendations. *arXiv preprint arXiv:1801.00209* (2017).

[39] Yong Zheng. 2017. Multi-Stakeholder Recommendation: Applications and Challenges. *arXiv preprint arXiv:1707.08913* (2017).

[40] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. [n. d.]. Optimized cost per click in taobao display advertising.

Semi-Supervised Entity Alignment via Knowledge Graph Embedding with Awareness of Degree Difference

Shichao Pei, Lu Yu, Robert Hoehndorf, Xiangliang Zhang
King Abdullah University of Science and Technology
Thuwal, 23955, SA
{shichao.pei,lu.yu,robert.hoehndorf,xiangliang.zhang}@kaust.edu.sa

ABSTRACT

Entity alignment associates entities in different knowledge graphs if they are semantically same, and has been successfully used in the knowledge graph construction and connection. Most of the recent solutions for entity alignment are based on knowledge graph embedding, which maps knowledge entities in a low-dimension space where entities are connected with the guidance of prior aligned entity pairs. The study in this paper focuses on two important issues that limit the accuracy of current entity alignment solutions: 1) labeled data of priorly aligned entity pairs are difficult and expensive to acquire, whereas abundant of unlabeled data are not used; and 2) knowledge graph embedding is affected by entity's degree difference, which brings challenges to align high frequent and low frequent entities. We propose a **semi-supervised entity alignment** method (SEA) to leverage both labeled entities and the abundant unlabeled entity information for the alignment. Furthermore, we improve the **knowledge graph embedding with awareness of the degree difference** by performing the adversarial training. To evaluate our proposed model, we conduct extensive experiments on real-world datasets. The experimental results show that our model consistently outperforms the state-of-the-art methods with significant improvement on alignment accuracy.

KEYWORDS

Knowledge Graph; Entity Alignment; Semi-supervised Learning

ACM Reference Format:
Shichao Pei, Lu Yu, Robert Hoehndorf, Xiangliang Zhang. 2019. Semi-Supervised Entity Alignment via Knowledge Graph Embedding with Awareness of Degree Difference. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313646

1 INTRODUCTION

Knowledge graphs have been constructed and widely applied to organize and represent the knowledge of different domains, including the most popular ones such as Freebase [1], YAGO [27] and DBpedia [17]. Even in the same domain, knowledge graphs (KG) are generated by different methods in different languages. To comprehensively represent the knowledge in one domain, it is thus essential to connect multiple knowledge graphs in same domain. More specifically, entities in different knowledge graphs are aligned if they are semantically same (as known as entity alignment problem).

Given two knowledge graphs with labeled entities that are known to be same in semantics, existing entity alignment approaches need human-involved feature design [20] or extra resources [16, 22, 27], such as information of the entity and the relation, as the training set, to supervise the learning process of entity alignment. Most of these methods require a sufficient number of labeled entities to generalize well in downstream applications. However, getting labeled entities is difficult and expensive. The number of accessible prior entity is usually a small proportion of a whole knowledge graph [5, 29]. ITransE [41], IPTransE [41], BootEA [29] and KDCoE [4] tried to propose new aligned pairs of entities as training data iteratively during training process. However, the focus is on how to acquire more aligned entities, leaving the abundant unaligned entities without consideration. Therefore, this paper targets on designing **semi-supervised entity alignment** model, which learns from both labeled and unlabeled entities. The underlying distribution of abundant unaligned entities will help on mitigating the risk of generalization errors caused by limited amount of aligned entities [42].

Recently, knowledge graph embedding methods, e.g., TransE [3] and PTransE [18] show significant improvement on entity alignment. For example, MtransE [5] encodes entities and relations of each language in a separated embedding space, and learns transitions to map each embedding vector to its cross-lingual counterparts in other spaces. ITransE [41] and IPTransE [41] were proposed to encode both entities and relations of different KGs into a unified low-dimensional space jointly and iteratively. JAPE [28], KDCoE [4], and Graph Convolutional Network-based approach [34] all jointly model structure and attribute information of knowledge graphs. Our work also takes advantage of embedding methods for building a semi-supervised entity alignment model. However, rather than directly using existing embedding methods, we address an important issue in the embedding process, which is caused by the **degree difference of entities** in different knowledge graphs. This phenomenon is demonstrated in Figure 1.

Entities in KGs have different degrees, i.e., popular entities are more connected with other concepts than rare entities. As pointed in recent study of natural language processing [7, 21] and machine translation [24, 25], word embedding methods encode more frequency information than semantic information of words in the resulted low-dimension space. We also investigate and find the same problem in knowledge graph embedding. Our findings verify

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313646

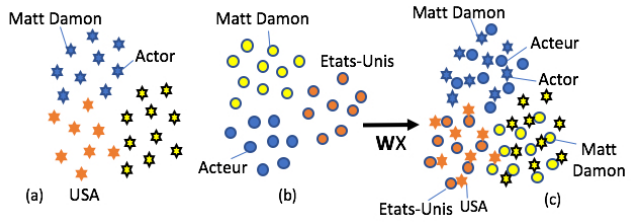


Figure 1: Illustration of the impact of entity's degree difference on embedding and alignment results. (a): embedding of (only entities in) a KG in English, where "Matt Damon" and "Actor" are both popular (with high degree) and thus mapped close in blue. (b): embedding of a KG in French, where "Acteur" is more popular (in blue) than "Matt Damon" (in yellow). (c): alignment of English KG and French KG ("Actor" with "Acteur", and "USA" with "Etats-Unis"), which shows that "Matt Damon" in French is far from the "Matt Damon" in English, even they are actually the same entity.

that entities with similar degree tend to be aggregated into a same region in the embedding space.

This deficiency of embedding methods brings challenges to entity alignment problem. Figure 1 shows a toy example of the problem, when aligning "Matt Damon" in English and French KGs. "Matt Damon" is a very popular name in English KG, while "Matt Damon" in French has fewer records. Embedding results of English KG show "Matt Damon" is close to other popular entities like "Actor" (in color blue in Figure 1 (a)), while the results of French KG (in Figure 1 (b)) have "Matt Damon" in yellow region that is distant to popular entity "Acteur" in blue region. By aligning "Actor" in English and "Acteur" in French, "USA" in English and " Etats-Unis" in French, the embedding space of French KG can be transformed to the embedding space of English KG (e.g., with linear transformation $\mathbf{W}$). The alignment is illustrated in Figure 1 (c)), where we can see that "Matt Damon" in French is far from the "Matt Damon" in English, even they are actually the same entity.

To tackle the above challenges, we propose a **semi-supervised entity alignment** framework called SEA, which takes advantages of both aligned with unaligned entities, which are represented in low-dimension space by knowledge graph embedding with **awareness of the degree difference**. In particular, the impact of degree difference is mitigated by an adversarial training, which prevents entities with similar degree from being aggregated into the same region in the embedding space during training. We also design a cycle-consistency based translation loss using the unaligned entities to reduce the search space when learning the mappings. Thus, the learned mappings can better align the entity in one knowledge graph to the corresponding entity in another knowledge graph.

Our main contributions in this work are summarized as follows:

- We propose to solve entity alignment in a semi-supervised way, not only using the given aligned entity, but also incorporating the unaligned entity to enhance the performance.
- We indicate the impact of entity's degree difference on embedding of knowledge graph, and address the problem under the adversarial training framework.
- We conduct extensive experiments on four real-world datasets to evaluate the proposed SEA model for the task of entity alignment. The results demonstrate its advantages over the state-of-the-art methods, with significant improvement on several datasets.

The rest of the paper is organized as follows. We discuss the related works in Section 2. The proposed method is described in Section 3 and followed by the experimental results in Section 4. Finally Section 5 concludes the whole paper.

2 RELATED WORK

2.1 KG Embedding

Knowledge graph (KG) embedding has became an important tool for knowledge graph analysis and semantic information modeling tasks with the fast growth of large-scale knowledge graphs. The KG embedding approaches can be roughly categorized into two groups: translational distance models and semantic matching models [33].

TransE [3] is the most representative translational distance model. It considers a relation as the translation from its head entity to its tail entity and represents both entities and relations as vectors into a same low dimensional vector space. TransE [3] characterizes a triple (h, r, t) following a common assumption $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, where $\mathbf{h}$ and $\mathbf{t}$ are the representations of h and t, $\mathbf{r}$ is the representation of r. TransE has shown its feasibility for KG modeling, and it has been improved by many following studies, such as TransH [35], TransR [19], TransD [12], TranSparse [13], and PTransE [18].

Semantic matching models use the similarity-based scoring functions instead of the distance-based scoring functions. RESCAL [23] and Bilinear [11] model each relation as a matrix and associate each entity with a vector. DistMult [36] restricts the relation matrices to diagonal matrices. And HolE combines RESCAL [23] and DistMult [36] to get expressive and efficiency model. ComplEx [31] introduces complex-valued embeddings for better asymmetric relations modeling. Several works in [2, 26] conduct semantic matching using neural network.

The effect of word's frequency difference on embedding results has been recognized in [7, 21] and [24, 25], but the effect of entity's degree difference on embedding results has not been explored. In this paper, we address the effect of entity's degree on the embedding and propose to improve the most widely-used TransE method by mitigating the effect of entity's degree difference. The obtained KG embedding are used for improving entity alignment, especially for the rare and popular entities.

2.2 Entity Alignment

Some forerunners proposed to address the entity alignment problem using crowdsourcing [17, 32] and well-designed hand-crafted features [20]. However, the approaches suffer from the requirement of heavy human efforts, and is thus costly and labor-expensive. Many works leverage the extra resources, such as OWL properties [9], entity descriptions [37], information of entities and relations [16, 22, 27]. Such methods are complex and usually limited by the availability of the extra information about a knowledge graph.

In recent study, KG embedding-based approaches become the most popular solution for entity alignment. MTransE [5] is the representative work which encodes entities and relations of each

language in a separated embedding space, and learns transitions to map each embedding vector to its cross-lingual counterparts in the other space. Then ITransE [41] and IPTransE [41] are proposed to encode both entities and relations of different KGs into a unified low-dimensional space jointly by an iterative and parameter sharing method. BootEA [29] addresses the lack of labeled data by bootstrapping strategy and tries to reduce error accumulation during iterations by employing an alignment editing method. It also proposes an improved KG embedding approach with the limit-based loss and truncated uniform negative sampling. BootEA achieves a significant performance improvement. Further, several works, such as JAPE [28] and KDCoE [4], consider to jointly model the structure and attributes information of KGs. Recently, a Graph Convolutional Network-based entity alignment approach [34] uses GCNs to embed entities of each language into a unified vector space combining structural and attributes information.

All the methods above focus on the utilization of aligned entities, including BootEA [29], which tried iteratively enlarge the labeled entity pairs based on the bootstrapping strategy. The abundant unaligned entities that have rich content information are not used in the alignment process. Therefore, we propose a semi-supervised framework to align entities in different KGs based on both labeled and unlabeled data.

3 METHODOLOGY

In this section, we firstly introduce the notation and problem definition, then describe the proposed method in detail.

3.1 Notation and Problem Definition

A knowledge graph can be noted as $G = (E, R, T)$, where E is the set of entities, R is the set of relations, and T is the set of triples, each of which is a triple (h, r, t), including the head entity h, the relation r and the tail entity t. By using KG embedding, each triple can be presented as $(\mathbf{h}, \mathbf{r}, \mathbf{t})$, in which boldfaced $\mathbf{h}$, $\mathbf{r}$, and $\mathbf{t}$ represent the embedding vectors of head h, relation r, and tail t, respectively.

Let $G_1 = (E_1, R_1, T_1)$ and $G_2 = (E_2, R_2, T_2)$ be two KGs in different languages. $AS^L = \left\{(e_{i_1}, e_{i_2}) | e_{i_1} \in E_1^L, e_{i_2} \in E_2^L\right\}$ is a set of labeled entity pairs that are same in semantics, e.g., e_{i_1} in G_1 shares same meaning with its counterpart e_{i_2} in G_2. Entity alignment is a task to find and align the remaining semantically same entities $\{e_{i_1} \in E_1^U\}$ and $\{e_{i_2} \in E_2^U\}$ where $E_1^U = E_1 \setminus E_1^L$ and $E_2^U = E_2 \setminus E_2^L$. Unlike the previous study which builds alignment model based on AS^L only, our approach SEA builds the semi-supervised alignment model based on both AS^L and E_1^U and E_2^U.

The framework of our proposed method SEA is shown in Figure 2. SEA has two modules, knowledge graph embedding with awareness of degree difference of entities (called degree-aware KGE for short), and semi-supervised entity alignment. We introduce them in details next.

3.2 Degree-Aware Knowledge Graph Embedding

Like the previous works in [4, 5, 41], we build our degree-aware KG embedding model by following TransE [3], which is the most representative translational distance model. It is worth mentioning that our ideas of mitigating the effect of entity's degree difference on embedding can also be applied to other KG embedding methods, which is not the focus of this work.

When applying TransE on both knowledge graphs G_1 and G_2, entities and relations are projected into the same low-dimensional vector space by encoding the triples (h, r, t), and making $\mathbf{h}+\mathbf{r} \approx \mathbf{t}$ when (h, r, t) holds. Specifically, the embeddings of relations can translate the embeddings of head entities to tail entities. The margin-based ranking object function minimized by TransE over a knowledge graph G_i is defined as:

$$L_{G_i}(G_i; \varphi^i, \theta_e^i) = \sum_{(h,r,t) \in T_i} L_t(h, r, t) \tag{1}$$

where φ^i refers to the model parameters for G_i, θ_e^i presents the learned embedding from G_i, and $L_t(h, r, t)$ is the object function for a triple (h, r, t):

$$L_t(h, r, t) = \sum_{(h', r, t') \in T'_{(h,r,t)}} \left[\gamma + E(h, r, t) - E(h', r, t')\right]_+ \tag{2}$$

where $[x]_+ = \max\{0, x\}$ denotes the positive part of x, γ is a margin hyper-parameter which is greater than 0, and $E(h, r, t)$ indicates the energy function:

$$E(h, r, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2 \tag{3}$$

and T' denotes the negative sample set for the triple (h, r, t):

$$T'_{(h,r,t)} = \{(h', r, t) | h' \in E\} \cup \{(h, r, t') | t' \in E\} \tag{4}$$

where (h', r, t) and (h, r, t') are the Bernoulli negative-sampled triples by replacing h or t in (h, r, t). Finally, we can get the object function for both knowledge graphs G_1 and G_2:

$$L_G = L_{G_1} + L_{G_2} \tag{5}$$

Next, we present the degree-aware KGE method to address the impact of entity's degree difference on the knowledge graph embedding. As discussed before, the entities with similar degree values tend to be aggregated into the same regions in the embedding space. However, for entity alignment, entities with similar semantic information are expected to be closer without the impact of entities' degree. Thus, we design the degree-aware KGE model by training the knowledge graph embeddings in an adversarial framework, inspired by Generative Adversarial Network (GAN) [8]. Given a graph G_i, we design two discriminators to classify the entities with different degrees in G_i, and the degree-aware KGE model can be regarded as a generator which produces "Good" embeddings to fool the discriminators. We catalog the degree of entities into three levels, *high* degree, *normal* degree, and *low* degree. One discriminator D_1 categorizes the entities with *high* degree and *normal* degree, while the other discriminator D_2 is in charge of classifying the entities with *low* degree and *normal* degree. The expectation is: the learned knowledge graph embeddings not only minimize the margin-based ranking loss function defined above, but also can fool the two discriminators. The impact of degree is thus removed from the learned embeddings when the two discriminators cannot distinguish entities according to the information of degree.

Let D_1 be the first discriminator with parameters ϕ_1^i, and D_2 be the other discriminator with parameter ϕ_2^i. The inputs of D_1 are the entities from E_{hd}^i and E_{nd}^i, which are entities with *high* degree and *normal* degree in graph G_i. The inputs of D_2 are the entities

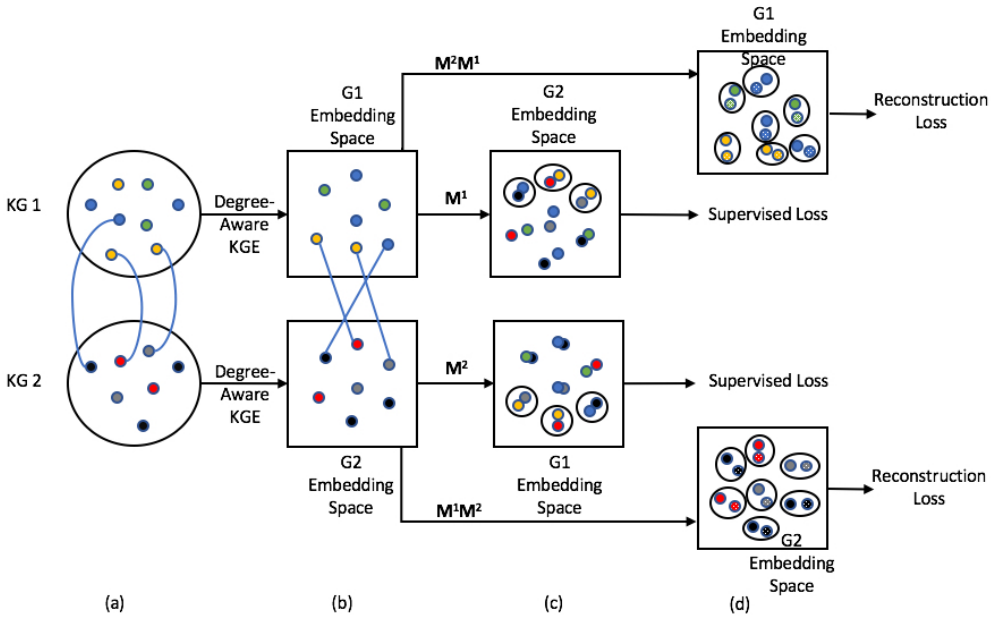


Figure 2: Framework of SEA for semi-supervised entity alignment. (a) shows two knowledge graphs G_1 and G_2 with three pairs of aligned entities. (b) presents the learned embeddings via the degree-aware KG embedding approach. (c) demonstrates the embedding spaces after transferring G_1 to G_2 and G_2 to G_1 using $\mathbf{M}^1$ and $\mathbf{M}^2$ learned by the pairs of aligned entities. The entities within small circles are close due to the supervised alignment guidance. (d) shows the cycle consistency of entities after being transferred back the original embedding spaces. The transferred embedding of each entity should be close to the original embedding of this entity in the original embeddings spaces (as indicated in the small circles).

from E^i_{ld} and E^i_{nd}, which are entities with *low* degree and *normal* degree in graph G_i. Both discriminators are learned to minimize loss functions:

$$L_{D_1} = \frac{1}{|E^i_{hd}|}\sum_{e\in E^i_{hd}} \log D_1(\theta^i_e, \phi^i_1) + \frac{1}{|E^i_{nd}|}\sum_{e\in E^i_{nd}} \log(1 - D_1(\theta^i_e, \phi^i_1)) \tag{6}$$

$$L_{D_2} = \frac{1}{|E^i_{ld}|}\sum_{e\in E^i_{ld}} \log D_2(\theta^i_e, \phi^i_2) + \frac{1}{|E^i_{nd}|}\sum_{e\in E^i_{nd}} \log(1 - D_2(\theta^i_e, \phi^i_2)) \tag{7}$$

Taking also the objective function Eq. (1), the over all minimax object function for learning embeddings of G_i will be:

$$\min_{\varphi^i,\theta^i_e}\max_{\phi^i_1,\phi^i_2} L_{G_i}(G_i;\varphi^i,\theta^i_e) - \alpha L_{D_1}(E^i_{hd}, E^i_{nd};\phi^i_1,\theta^i_e) - \alpha L_{D_2}(E^i_{ld}, E^i_{nd};\phi^i_2,\theta^i_e) \tag{8}$$

where α is a tradeoff parameter. In the adversarial training, L_{G_i} is minimized w.r.t. φ^i and θ^i_e to produce "Good" embeddings to fool D_1 and D_2. Meanwhile, D_1 and D_2 are trained w.r.t. ϕ^i_1 and ϕ^i_2 to distinguish entities with different degree levels. Following the iterative training process of GAN [8], we first train the KG embedding model with D_1 and D_2 fixed, and then train the discriminators with KG embedding model fixed. The same process will be applied to G_1 and G_2.

3.3 Semi-Supervised Entity Alignment

After obtaining entity embeddings of graph G_1 and G_2, labeled entities are aligned to minimize

$$\sum_{(e_i,e_j)\in AS^L} \left\|\mathbf{M}^1\theta^1_{e_i} - \theta^2_{e_j}\right\|_2 + \left\|\mathbf{M}^2\theta^2_{e_j} - \theta^1_{e_i}\right\|_2 \tag{9}$$

where $\mathbf{M}^1$ and $\mathbf{M}^2$ are the $d \times d$ translation matrices, d is the dimension of embeddings of entities. Note that our model jointly learns the translation of embeddings of entities in both directions for each knowledge graph. That is to say, $\mathbf{M}^1$ is learned to transfer the embeddings of G_1 into the embeddings space of G_2, and $\mathbf{M}^2$ is to transfer the embeddings of G_2 into the embeddings space of G_1. Due to the limited labeled set, the learned $\mathbf{M}^1$ and $\mathbf{M}^2$ cannot be generalized well to all entities in the two knowledge graphs.

To improve generalizability and incorporate unlabeled entities of two knowledge graphs in the alignment process, we define a cycled consistent loss, inspired by the work of CycleGAN [38] in computer vision field, where transitivity is used to regularize structured data [38] to do visual tracking [14, 30], 3D shape matching [10], co-segmentation [39, 40] and depth estimation [6]. Translation $\mathbf{M}^1$ and $\mathbf{M}^2$ should be able to

1) bring e_i in G_1 back after mapping cycle $e_i \to e_j \to \hat{e}_i$, i.e., the distance d_{g_1} between e_i and $\hat{e}_i$ should be small;

2) bring bring e_j back after mapping cycle $e_j \to e_i \to \hat{e}_j$, i.e., the distance d_{g_2} between e_j and $\hat{e}_j$ should be small.

Formally, the cycle process is:

$$\theta^1_{e_i} \rightarrow \mathbf{M^1}\theta^1_{e_i} \rightarrow \mathbf{M^2}\mathbf{M^1}\theta^1_{e_i} \tag{10}$$

$$\theta^2_{e_j} \rightarrow \mathbf{M^2}\theta^2_{e_j} \rightarrow \mathbf{M^1}\mathbf{M^2}\theta^2_{e_j} \tag{11}$$

Combing the alignment loss function in Eq. (9) and the cycle consistency restriction, we define the loss function of our semi-supervised entity alignment as:

$$\begin{aligned} L_{SEA}(\mathbf{M^1}, \mathbf{M^2}) = \alpha_1 \sum_{(e_i,e_j)\in AS^L} \left\|\mathbf{M^1}\theta^1_{e_i} - \theta^2_{e_j}\right\|_2 + \left\|\mathbf{M^2}\theta^2_{e_j} - \theta^1_{e_i}\right\|_2 \\ + \left\|\mathbf{M^2}\mathbf{M^1}\theta^1_{e_i} - \theta^1_{e_i}\right\|_1 + \left\|\mathbf{M^1}\mathbf{M^2}\theta^2_{e_j} - \theta^2_{e_j}\right\|_1 \\ + \alpha_2 \sum_{e_i\in E^U_1} \left\|\mathbf{M^2}\mathbf{M^1}\theta^1_{e_i} - \theta^1_{e_i}\right\|_1 + \alpha_2 \sum_{e_j\in E^U_2} \left\|\mathbf{M^1}\mathbf{M^2}\theta^2_{e_j} - \theta^2_{e_j}\right\|_1 \end{aligned} \tag{12}$$

where α_1 and α_2 are the tradeoff parameters for balancing the loss between labeled and unlabeled data.

We initialize the embeddings of KGs by drawing from a Gaussian initialization, and initialize the matrices by using orthogonal initialization. We use SGD as our optimizer, and normalize all embeddings by $L2$ norm. The trade-off parameter α, α_1 and α_2 are set by grid search. Once $\mathbf{M}^1$, $\mathbf{M}^2$ are learned, an entity e in G_1 can be aligned by first transferring to G_2 as $\mathbf{M}^1\theta_e$ and then selecting the most similar entity in G_2. Similarly, an entity e in G_2 can be aligned by first transferring to G_1 as $\mathbf{M}^2\theta_e$ and then selecting the most similar entity in G_1.

4 EXPERIMENTS

In this section, we conduct experiments on several real-world datasets with different sizes, and evaluate our proposed method for entity alignment.

4.1 Datasets and Baselines

To comprehensively evaluate the effectiveness of our SEA method, we use two trilingual knowledge graph datasets from WK31 provided in [5]. WK31 datasets consist of English(En), French(Fr), and German(De) knowledge graphs which are extracted from Person domain of DBpedia's with known aligned entities as ground truth. WK31 includes two datasets with different sizes, which are WK31-15K and WK31-120K. The statistics of the datasets are given in Table 2 and Table 3. For datasets WK31-15K and WK31-120K, we extract the aligned entities from aligned triples.

To verify the effectiveness of our proposed method, we include the following methods for performance comparison, including: **MTransE** [5], **ITransE** [41], **JAPE** [28], **GCN-based method**[34], **BootEA**[29], **SEA w/o DA** (a variant of the proposed SEA, by removing the degree-aware KGE part), and **SEA**.

4.2 Evaluation Metrics and Parameter Settings

We adopt two popular metrics, *Hits@k* and *MRR* for evaluating entity alignment results. *Hits@k* measures the proportion of correctly aligned entities ranked in the top k proposed candidates. In our work, we report *Hits*@1, *Hits*@5 and *Hits*@10. Both metrics are preferred to be higher to present better performance.

Table 1: Statistics of the WK31 dataset

dataset	#Triple	#Entity	#Relation
WK31-15K En-Fr	En: 203,502 Fr: 170,605	En: 15,170 Fr: 15,393	En: 2,228 Fr: 2,422
WK31-15K En-De	En: 203,502 De: 145,616	En: 15,127 De: 14,603	En: 1,841 De: 596
WK31-120K En-Fr	En: 1,376,011 Fr: 767,750	En: 119,749 Fr: 118,591	En:3,109 Fr:2,336
WK31-120K En-De	En: 1,376,011 De: 391,108	En: 67,650 De: 61,942	En: 2,393 De: 861

Table 2: Number of aligned entity in different datasets.

Dataset	En-Fr	Fr-En	En-De	De-En
WK31-15K	10,108	10,164	11,594	11,445
WK31-120K	117,947	117,212	55,640	54,287

For all methods compared in 4.3, we set the dimension of knowledge graph embeddings $d = 100$ on all datasets. We find the optimal parameter settings for all baseline methods.

For our SEA method, we set that the *high* degree entities are those with top 20% degree values, while the *low* degree entities are those with bottom 20% degree values, and the rest are the *normal* degree entities. we search the margin γ among $\{0.5, 1, 1.5, 2\}$, and the tradeoff parameter α among $\{0.1, 0.3, 0.5, 0.7, 1.0\}$, α_1 and α_2 among $\{1, 2.5, 5, 7.5\}$ and $\{0.05, 0.15, 0.25, 0.35, 0.45\}$, respectively. The best configuration is $\gamma = 0.5$, $\alpha = 0.5$, $\alpha_1 = 2.5$ and $\alpha_2 = 0.25$. Discriminators are set as two-layers MLPs with 500 hidden units. We use Adam [15] to optimize the object function. Meanwhile, we use $L2$ norm to avoid potential over-fitting.

We randomly sample 30% of the aligned entities as the training set, and the rest aligned entities for testing. Each evaluation is repeated 5 times and we report the averaged *Hits@k* and *MRR*.

4.3 Performance Evaluation Results

The evaluation results are presented in Table 3-4. The best results are shown in bold among the group of methods, along with the percentage of improvement when comparing SEA without degree-aware (SEA w/o DA) and SEA with the best baseline methods. From these evaluation results, we have the following findings:

(1) **Our proposed SEA consistently outperforms the baseline methods on all datasets under different evaluation metrics**. This observation verifies that our proposed model effectively unifies the labeled and unlabeled data for improving entity alignment accuracy. In particular, our method achieves significant improvement when matching the top-1 ranked entity (*Hits*@1 is improved by more than 10%, or even 30-56%). Especially on the largest dataset WK31-120k, our SEA method has improvements on all metrics from 18% to 56%. BootEA is often the second best baseline due to the effective bootstrapping strategy for selecting labeled data. However, on the largest dataset WK31-120k, it sometimes performs worse than MTransE, mainly because it is difficult for bootstrapping to propose effective entities to label in a large dataset. In addition, bootEA need more time to calculate the similarity between each

Table 3: Entity alignment results of different methods on WK31-15K dataset. The best results are in bold, along with the percentage of improvement when comparing SEA w/o DA and SEA with the best baseline methods.

Language	En-Fr				Fr-En			
Metric	Hits@1	Hits@5	Hits@10	MRR	Hits@1	Hits@5	Hits@10	MRR
MtransE	16.77	21.64	25.35	0.198	19.85	31.27	38.21	0.261
ITransE	18.21	24.34	27.41	0.214	18.61	33.64	36.28	0.248
JAPE	15.68	23.45	28.69	0.208	16.22	28.93	34.71	0.219
GCN	17.24	27.29	31.16	0.220	17.58	30.82	36.21	0.237
BootEA	29.72	52.92	61.19	0.395	30.77	55.44	63.67	0.428
SEA w/o DA	36.78	54.89	62.37	0.454	38.61	58.69	62.51	0.481
SEA	**37.28**	**55.91**	**63.56**	**0.468**	**39.76**	**59.32**	**66.31**	**0.489**
Improvement %	**23.75 / 25.47**	**3.72 / 5.65**	**1.93 / 3.87**	**14.93 / 18.48**	**25.48 / 29.22**	**5.86/ 7.00**	**-1.82 / 4.14**	**12.38 / 14.25**
Language	En-De				De-En			
Metric	Hits@1	Hits@5	Hits@10	MRR	Hits@1	Hits@5	Hits@10	MRR
MtransE	6.170	8.48	10.39	0.078	4.69	6.61	7.74	0.059
ITransE	15.98	28.63	32.71	0.218	13.42	25.63	31.17	0.205
JAPE	16.85	27.32	34.74	0.226	13.92	22.15	29.68	0.189
GCN	18.25	31.30	37.26	0.248	15.70	27.53	33.31	0.217
BootEA	33.13	54.13	61.70	0.435	30.47	45.33	53.52	0.381
SEA w/o DA	37.74	54.81	**65.74**	0.462	**32.86**	46.21	53.53	0.393
SEA	**38.59**	**55.21**	64.06	**0.473**	32.11	**47.53**	**55.82**	**0.402**
Improvement %	**13.91 / 16.48**	**1.26 / 2.01**	**6.55 / 3.82**	**6.21 / 8.74**	**7.84 / 5.38**	**1.94/ 4.85**	**0.19 / 4.31**	**3.15 / 5.51**

Table 4: Entity alignment results of different methods on WK31-120K.

Language	En-Fr				Fr-En			
Metric	Hits@1	Hits@5	Hits@10	MRR	Hits@1	Hits@5	Hits@10	MRR
MtransE	21.01	22.24	22.82	0.217	21.11	23.63	25.24	0.227
ITransE	11.54	20.41	23.92	0.176	13.35	21.20	24.18	0.197
JAPE	6.98	16.10	22.74	0.127	8.64	17.85	23.38	0.134
GCN	9.32	18.62	25.48	0.146	10.81	18.22	26.39	0.153
BootEA	17.56	27.41	31.85	0.235	18.46	28.65	31.85	0.241
SEA w/o DA	26.48	34.38	38.57	0.315	27.82	35.99	39.77	0.320
SEA	**28.02**	**35.82**	**39.70**	**0.321**	**28.72**	**36.74**	**41.37**	**0.331**
Improvement %	**26.04 / 33.36**	**25.42 / 30.68**	**21.10 / 24.65**	**34.04 / 36.59**	**31.78 / 36.05**	**25.62/ 28.23**	**24.87 / 29.89**	**32.78 / 37.34**
Language	En-De				De-En			
Metric	Hits@1	Hits@5	Hits@10	MRR	Hits@1	Hits@5	Hits@10	MRR
MtransE	5.38	6.53	7.33	0.062	4.97	7.39	9.12	0.066
ITranE	7.62	15.54	19.41	0.112	6.41	12.82	15.27	0.085
JAPE	4.37	12.91	14.49	0.076	5.23	10.46	14.10	0.071
GCN	6.32	15.14	20.77	0.109	5.91	13.85	17.68	0.092
BootEA	11.57	22.08	27.75	0.179	10.32	22.11	26.36	0.169
SEA w/o DA	16.21	25.49	30.73	0.213	14.59	25.66	29.76	0.202
SEA	**17.23**	**27.48**	**32.83**	**0.227**	**16.11**	**27.06**	**32.48**	**0.218**
Improvement %	**40.10 / 48.91**	**15.44 / 24.45**	**10.74 / 18.31**	**19.01 / 26.82**	**41.38 / 56.10**	**16.06/ 22.38**	**12.89 / 23.21**	**19.53 / 28.99**

entity pair in the dataset except the training data. All the results show that the advantage of our semi-supervised method.

(2) **Our proposed degree-aware knowledge graph embedding approach produces improved entity representations, and thus improves entity alignment results**. This conclusion is drawn from the comparison of improvement made by SEA w/o DA and SEA. After mitigating the effect of entity's degree difference on embedding results, SEA can further improve the semi-supervised results obtained by SEA w/o DA, with only few exceptions. It is worth noting here that the aligned entities with *high* and *low* degree is a small proportion of the whole aligned entity set. How much degree-aware KE embedding can help is limited by this small proportion of entities with *high* and *low* degree levels. Therefore, the improvement made from SEA w/o DA to SEA is justifiable.

5 CONCLUSION

Entity alignment is an important research problem in knowledge graph analysis and management. In this work, we propose a semi-supervised method with degree-aware KG embedding to do the entity alignment. We design a cycle-consistency based translation loss to leverage the unaligned entity to enhance the ability of alignment instead of only using labeled aligned entities. In addition, we observe that the entity degree can influence the learned embeddings and degrade the performance of downstream application. We thus adopt adversarial training to alleviate the problem and improve the embedding results. We conduct experiments on four real-world datasets. The experimental results show that our model consistently outperforms the state-of-art methods on the entity alignment task. Based on the success of this first attempt of semi-supervised entity alignment, in future, we will consider the relation information in the graph to enhance the model.

ACKNOWLEDGMENTS

The research reported in this publication was supported by funding from King Abdullah University of Science and Technology (KAUST), under award number FCC/1/1976-19-01.

REFERENCES

[1] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM, 1247–1250.

[2] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning* 94, 2 (2014), 233–259.

[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*. 2787–2795.

[4] Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. 2018. Co-training Embeddings of Knowledge Graphs and Entity Descriptions for Cross-lingual Entity Alignment. *IJCAI* (2018).

[5] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2016. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. *arXiv preprint arXiv:1611.03954* (2016).

[6] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. 2017. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, Vol. 2. 7.

[7] Chengyue Gong, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2018. FRAGE: Frequency-Agnostic Word Representation. *Advances in neural information processing systems* (2018).

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[9] Wei Hu, Jianfeng Chen, and Yuzhong Qu. 2011. A self-training approach for resolving object coreference on the semantic web. In *Proceedings of the 20th international conference on World wide web*. ACM, 87–96.

[10] Qi-Xing Huang and Leonidas Guibas. 2013. Consistent shape maps via semidefinite programming. In *Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing*. Eurographics Association, 177–186.

[11] Rodolphe Jenatton, Nicolas L Roux, Antoine Bordes, and Guillaume R Obozinski. 2012. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems*. 3167–3175.

[12] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 687–696.

[13] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge Graph Completion with Adaptive Sparse Transfer Matrix.. In *AAAI*. 985–991.

[14] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. 2010. Forward-backward error: Automatic detection of tracking failures. In *Pattern recognition (ICPR), 2010 20th international conference on*. IEEE, 2756–2759.

[15] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[16] Simon Lacoste-Julien, Konstantina Palla, Alex Davies, Gjergji Kasneci, Thore Graepel, and Zoubin Ghahramani. 2013. Sigma: Simple greedy matching for aligning large knowledge bases. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 572–580.

[17] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, and others. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[18] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling relation paths for representation learning of knowledge bases. *arXiv preprint arXiv:1506.00379* (2015).

[19] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion.. In *AAAI*, Vol. 15. 2181–2187.

[20] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M Suchanek. 2013. Yago3: A knowledge base from multilingual wikipedias. In *CIDR*.

[21] Jiaqi Mu, Suma Bhat, and Pramod Viswanath. 2017. All-but-the-top: Simple and effective postprocessing for word representations. *arXiv preprint arXiv:1702.01417* (2017).

[22] Thanh Nguyen, Viviane Moreira, Huong Nguyen, Hoa Nguyen, and Juliana Freire. 2011. Multilingual schema matching for Wikipedia infoboxes. *Proceedings of the VLDB Endowment* 5, 2 (2011), 133–144.

[23] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data.. In *ICML*, Vol. 11. 809–816.

[24] Myle Ott, Michael Auli, David Granger, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. *arXiv preprint arXiv:1803.00047* (2018).

[25] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1715–1725.

[26] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[27] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 697–706.

[28] Zequn Sun, Wei Hu, and Chengkai Li. 2017. Cross-lingual entity alignment via joint attribute-preserving embedding. In *International Semantic Web Conference*. Springer, 628–644.

[29] Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping Entity Alignment with Knowledge Graph Embedding.. In *IJCAI*. 4396–4402.

[30] Narayanan Sundaram, Thomas Brox, and Kurt Keutzer. 2010. Dense point trajectories by GPU-accelerated large displacement optical flow. In *European conference on computer vision*. Springer, 438–451.

[31] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*. 2071–2080.

[32] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.

[33] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.

[34] Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. 2018. Cross-lingual Knowledge Graph Alignment via Graph Convolutional Networks.. In *EMNLP*.

[35] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes.. In *AAAI*, Vol. 14. 1112–1119.

[36] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575* (2014).

[37] Yang Yang, Yizhou Sun, Jie Tang, Bo Ma, and Juanzi Li. 2015. Entity matching across heterogeneous sources. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1395–1404.

[38] Zili Yi, Hao (Richard) Zhang, Ping Tan, and Minglun Gong. 2017. DualGAN: Unsupervised Dual Learning for Image-to-Image Translation.. In *ICCV*. 2868–2876.

[39] Tinghui Zhou, Yong Jae Lee, Stella X Yu, and Alyosha A Efros. 2015. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1191–1200.

[40] Tinghui Zhou, Philipp Krahenbuhl, Mathieu Aubry, Qixing Huang, and Alexei A Efros. 2016. Learning dense correspondence via 3d-guided cycle consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 117–126.

[41] Hao Zhu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Iterative entity alignment via joint knowledge embeddings. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 4258–4264.

[42] Xiaojin Zhu and Andrew B Goldberg. 2009. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning* 3, 1 (2009), 1–130.

Event-Driven Analysis of Crowd Dynamics in the *Black Lives Matter* Online Social Movement

Hao Peng
University of Michigan
haopeng@umich.edu

Ceren Budak
University of Michigan
cbudak@umich.edu

Daniel M. Romero
University of Michigan
drom@umich.edu

ABSTRACT

Online social movements (OSMs) play a key role in promoting democracy in modern society. Most online activism is largely driven by critical offline events. Among many studies investigating collective behavior in OSMs, few has explored the interaction between crowd dynamics and their offline context. Here, focusing on the *Black Lives Matter* OSM and utilizing an event-driven approach on a dataset of 36 million tweets and thousands of offline events, we study how different types of offline events—police violence and heightened protests—influence crowd behavior over time. We find that police violence events and protests play important roles in the recruitment process. Moreover, by analyzing the re-participation dynamics and patterns of social interactions, we find that, in the long term, users who joined the movement during police violence events and protests show significantly more commitment than those who joined during other times. However, users recruited during other times are more committed to the movement than the other two groups in the short term. Furthermore, we observe that social ties formed during police violence events are more likely to be sustained over time than those formed during other times. Contrarily, ties formed during protests are the least likely to be maintained. Altogether, our results shed light on the impact of bursting events on the recruitment, retention, and communication patterns of collective behavior in the *Black Lives Matter* OSM.

KEYWORDS

online social movement, social media, black lives matter

ACM Reference Format:
Hao Peng, Ceren Budak, and Daniel M. Romero. 2019. Event-Driven Analysis of Crowd Dynamics in the *Black Lives Matter* Online Social Movement. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313673

1 INTRODUCTION

Social movements often signify social change [5, 30, 35]. They help transform what have been considered radical ideas of one generation to the common sense of the next [14]. Social media has become instrumental in driving and facilitating politically contentious online social movements (OSMs) throughout the world [12, 20, 36, 44].

Previous research studied various OSMs such as the 2010 Arab Spring [28], the 2011 Egyptian Revolution [41, 50], the 2011 Occupy Wall Street [9, 20], through the lens of social media, mainly Twitter [2, 3, 22, 32, 40]. Topics they investigated range from participant motivation [44], movement framing [24, 43], informal learning [20], to information diffusion [22, 28, 41]. There are also studies focusing on crowd behavior in OSMs. Subjects they examined include recruitment process [23], user engagement [6, 9, 40], and online interactions [34], as well as the temporal evolution of crowd behavior [2, 4, 9, 11, 48]. Although these studies focus on different OSMs that were driven by very different political agendas, most of them convey the same message: *the unique characteristics of crowd behavior are often associated with critical offline events*. Yet, there is limited research on investigating how different types of offline events influence the behavior of online crowds and affect their interactions with the movement over time.

An event helps a person think about how the world diverges from his basic values [25]. On one hand, exogenous shocking events such as police shootings can contribute to the recruitment into the *Black Lives Matter* movement by creating "moral shocks" that can transform the public emotion from dread to outrage [17, 18, 25]. These complex emotional processes usually generate broader impact on the public sphere and can suddenly cause wide-scale unplanned movement participation of peripheral users in online social networks [8, 27, 31]. On the other hand, endogenous events such as self-organized protests can have an effect on crowd behavior with the help of movement networks in two stages. First, planned protests in physical locations are likely to be preceded by considerable communication and coordination among highly involved activists. Second, participants in offline protests often interact with the movement on social media to gain awareness, articulate their core beliefs, and offer a frame to the public [24]. As a result, offline events can generate massive participation in OSMs and are thus crucial to the emergence, growth, and decline of online activism [13].

In this paper, we analyze crowd dynamics in the *Black Lives Matter (BLM)* online movement using a combination of 36 million tweets and thousands of compiled offline events. We focus our analysis on crowd dynamics around three types of time periods: cases of police violence, large-scale offline protests, and periods of neither police violence or protests referred to as "other times". Our event-driven investigation provides two major contributions. First, we find that police violence events and protests not only act as magnets to mobilize people around the movement, but they also greatly encourage user commitment in the long term. Second, we reveal that police violence events and protests have significantly different impact on the maintenance of social interactions established during these events. Altogether, our results shed light on how online crowds behave during bursting events.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313673

2 RELATED WORK

Online Social Movements, Crowds, and Events. Social media is now used as an organizing tool for collective action and has enabled the proliferation of online low-risk activism [31], which may indeed have been a productive component of recent political uprisings [41]. In addition, studies have revealed that social media largely promotes social movements by means of influencing individual decisions about participation [20], providing new sources of information beyond the control of authorities, and shaping the discourse, logistics, and outcomes of protests [44]. This suggests that social media facilitates the emergence of massive offline mobilization and can serve as a good proxy of studying OSMs.

Several studies in OSMs have been conducted through the lens of Twitter [2, 6, 22, 28, 32, 41, 48]. Existing work tends to focus on the characteristics of information generated during the course of various online movements. They are interested in analyzing movements along dimensions including the variation of the amount of information produced, the way how information spreads [22, 28, 41], and the contribution from different social actors involved in the movement [2, 6, 32, 48].

Other research examines crowd behavior and its dynamics in OSMs. Topics covered range from online social interactions [34], the recruitment dynamics [23], and community polarization [50], to opinion evolution [4]. There are also studies exploring the connections between the dynamics of OSMs and relevant offline events [7, 52]. In [42], Steinert et al. studied the relationship between online activities and the volume of protests on the ground. The work in [3] explored the possibility of forecasting onsite protests using social media stream data. Spiro et al. found that key time points with unique online social structure often map to exogenous events in physical locations [40]. These studies inform us about the relationship between the movement dynamics and relevant protests unfolded on the ground. However, to our knowledge, there is little work investigating how different types of offline events affect crowd dynamics in OSMs.

Outside of movement research, the interplay between crowd behavior and shocking events has been a rising theme of considerable interest in setting such as organizations [37], crowdsourcing platforms [53, 54], and global scale mobile communications [1]. Bagrow et al. analyzed the behavioral change and societal response of crowds to external perturbation during emergencies [1]. Romero et al. investigated the link between stock market shocks and the structure of the communication network in a large hedge fund [37]. A recent work [53] studied the collective response of editors of Chinese Wikipedia to the government censorship in mainland China. These studies reveal that online crowds are especially sensitive during external perturbations [19, 21, 46]. The above literature thus forms a significant part of the motivation for our effort on analyzing crowd dynamics in OSMs using an event-driven approach.

Research in the *BLM* Movement. *BLM* is an international activist movement initiated by the African American community in the U.S. after the killing of Trayvon Martin in 2012. The movement is centered around issues related to racial inequality and police brutality. Since the extensive adoption of the hashtag *#BlackLivesMatter* on social media after the death of Eric Garner and Michael Brown in 2014, the movement has generated thousands of on-site protests and demonstrations globally. Over time, the *BLM* movement gained worldwide recognition and the general public began to participate in the movement using a variety of hashtags on Twitter [24].

Several studies have analyzed the *BLM* online movement. In [33], Olteanu et al. provided a demographic characterization of users involved in this movement. Stewart et al. examined the frame process between competing online communities in the *BLM* movement through the use of related hashtags [43]. Ince et al. investigated how Twitter users interact with the *BLM* movement through hashtags and thus modify its framing [24]. A recent work [16] studied the divergent discourse between the *#BlackLivesMatter* protest and the *#AllLivesMatter* counter-protest on Twitter. Twyman et al. studied the dynamics of attention and knowledge production related to the *BLM* movement using Wikipedia [45]. However, as far as we know, there is limited work focusing on crowd dynamics in the *BLM* online movement and its relationship to offline events.

Most related to our work, De Choudhury et al. investigated the participation in the *BLM* online movement over time in different geographical locations and explored its relationship with protests unfolded on the ground [11]. Our work differs in that we systematically investigate the dynamics of user participation and patterns of social interactions, and explore how different types of offline events affect these dynamics in the *BLM* online movement.

3 DATA AND BACKGROUND

Twitter Dataset. Research indicates that the public use hashtags to interact with *BLM* to promote their messages, articulate core beliefs, and offer political solutions [11, 24, 33]. Here, we use data collected by [6] which includes tweets that are tagged with 53 *BLM* related hashtags. These race-related hashtags cover the stream of demonstrations and riots against the systemic racism and police violence toward African Americans [6].

The dataset spans from Jan. 2014 to May 2015, which covers the initial periods of the *BLM* movement on Twitter since it did not gain enough momentum until 2014 [11, 33]. It includes *every tweet* posted during this time period that contained one of the 53 *BLM* related hashtags mentioned above. There are 36 million tweets posted by 4 million users, among which we obtain 27 million retweets.

Offline Events. We compiled a list of *BLM* related events including police violence against African Americans and numerous on-site protests. The police violence events are based on news media reports and here we focus on the most prominent ones, which are shown in Table 1. The protest events are collected from *Elephrame*[1] [11]. This website keeps track of the *BLM* related demonstrations using information from multiple online sources. It estimates the number of demonstrators based on photos, videos, and media reports about each demonstration. Here we focus on protests with significant public awareness and potential impact on online crowds by defining a threshold $\mathcal{T}_{\text{protest}}$ of the number of participants and filtering those below it. We are able to identify 37 heightened protests with $\mathcal{T}_{\text{protest}} = 500$ in this study (The qualitative result does not change for $\mathcal{T}_{\text{protest}} = 100$).

[1] https://elephrame.com/textbook/BLM/chart

Table 1: Five police violence events used in our analysis and their number of NYTimes articles.

Victim	Date	Location	# articles
Eric Garner	2014-07-17	New York City	518
Michael Brown	2014-08-09	Ferguson, Missouri	820
Tamir Rice	2014-11-23	Cleveland, Ohio	67
Walter Scott	2015-04-04	Charleston, South Carolina	47
Freddie Gray	2015-04-12	Baltimore, Maryland	169

Event Categorization. To explore the interplay between crowd behavior on Twitter and identified offline events in the *BLM* movement, we create an event-type categorization of the Twitter dataset.

For each instance of police violence, we label the week after the time when the event occurs as a *police violence* period. For each protest, we label the week before and the week after the event as a *protest* period, reflecting the possibility that participants use social media to organize and recruit for the upcoming protest [16]. We combine adjacent time periods of the same event type into a single period. When a protest period overlaps with a police violence period, it is likely that the former is immediately driven by the latter, thus the overlapping period is labeled as *police violence*. Time periods between different events are labeled as *other times*, which can be considered as normal times when public interactions with the movement are not induced by external factors. Throughout this study, we exclude tweets posted before July 17, 2014 since there is no protest data in *Elephrame* before this date. The event categorization is shown as the *background* in Figure 1.

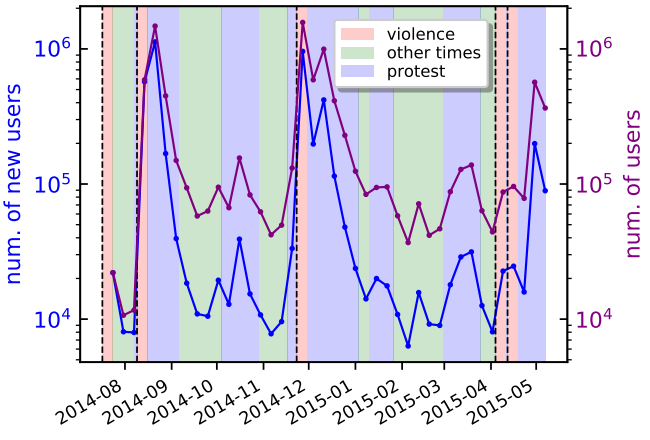


Figure 1: Number of new/all participants in each of the non-overlapping, one-week time windows starting from the first police violence event in the offline dataset. (*Background*): Event categorization of the twitter dataset used in this analysis. Five dashed vertical lines represent five police violence events in Table 1. A rectangular area denotes a time period, and its color represents its event type.

We first analyze the recruitment process in the *BLM* online movement using this event categorization. We split the whole periods into non-overlapping bins of one week starting from the time of the police violence against Eric Garner. We define the arrival time D_u of user u, as the time when u tweeted for the first time. For a given time window w_i, let n_i denote the number of new adopters whose arrival time D_u is during week w_i. Let p_i denote the number of participants who posted at least one tweet in week w_i. Figure 1 illustrates the time series of n_i and p_i respectively. It shows that there are repeating temporal variations in terms of the participation volume in the movement, with police violence events generating initial spikes followed by slight declines during protests and then fluctuating in other times. We also see that police violence events and protests function as magnets to attract new participants and mobilize people around the movement over time, which highlights the importance of examining mobilization patterns from the perspective of such events. The increase in activity during these events naturally leads to the question of whether users who became activated during different events behave and interact differently with the movement in the future.

4 DYNAMICS OF USER PARTICIPATION

Identifying characteristics of continued participation has been the central theme in recent OSM research [2, 6, 32, 48]. Yet, little is known about the evolution of user commitment as a function of offline events. In this section, we explore how arriving at the movement during a type of event is related to continued user participation in the *BLM* online movement.

We define a tracking window of t weeks to measure, for each user u, their commitment using their re-participation intensity within t weeks starting from D_u (their join time) along three dimensions. The first dimension is a binary variable y_u indicating if u re-participated or not within t weeks. The second one is the number of times f_u that u re-participated within t weeks (tweets in the movement). The third one is the number of days p_u that u re-participated within t weeks (active days in the movement). These three dimensions are meant to construct a thorough characterization of user commitment.

Modeling re-participation dynamics is complex. In addition to the events happening when they join, many other factors can directly influence users' re-participation behavior. To test whether the event during which users join the movement is related to their future participation, we utilize a prediction framework to control for potential confounds. For the first dimension, we adopt a logistic regression model where y_u is used as the dependent variable (Eq. 1). We employ a linear regression model for the second and the third dimensions separately, where $\log(f_u + 1)$ and $\log(p_u + 1)$ are used as the dependent variable, respectively (Eq. 2).

$$logit(p(y = 1)) = \beta_1 I + \sum_2^k \beta_k x_k + \epsilon \quad (1)$$

$$\log(y + 1) = \beta_1 I + \sum_2^k \beta_k x_k + \epsilon \quad (2)$$

The 8 independent variables are the same in all regressions, which are listed in Table 2. Variable 1 is categorical for which we use *other times* as the baseline, with two dummy variables indicating *police violence* and *protest*, respectively (I in Eq. 1 and Eq. 2). Variables $2-8$ are numerical and are used as controls in the regressions. Variables $2-4$ take into account the fact that users arrived at different time and the events following a user's start date can impact their re-participation dynamics irrespective of the events that occurred when they arrived. For instance, users may be more likely

Table 2: Independent variables for predicting user re-participation.

1	the event type of D_u (categorical)
2	the week number of D_u
3	the fraction of days labeled as *violence* in t weeks starting from D_u
4	the fraction of days labeled as *protest* in t weeks starting from D_u
5	the tweet count of u at D_u
6	the friend count of u at D_u
7	the follower count of u at D_u
8	the account age (in days) of u at D_u
Note: Variables $5-8$ are user meta data.	

to return to the movement if there is an upcoming police violence event since these events usually encourage participation. Variables $5-8$ control for the effect that can be explained by the difference between various user attributes. The Variance Inflation Factor (VIF) of all controls are below 5 in a multicollinearity test, which suggests that all of them should be included in the model. We standardize all numerical variables before performing the regression.

We train 10 models for each dimension by varying the tracking window from $t = 1$ to $t = 10$. To avoid right censoring bias, we focus on the same set of users whose join time D_u allow us to track their future participation for at least 10 weeks in our dataset. In total, we have 3.8 million users, among which 54% joined the *BLM* online movement during police violence periods, 42% joined during protest periods, and 4% joined during other times.

Our main objective is to estimate the impact of offline events on user commitment, therefore we focus on the first independent variable—the event type of users' starting time, and measure the relative change for *police violence* and *protest* over *other times*. Namely, for each dummy variable, we calculate the percentage increase in each dimension over the baseline. In figure 2, we show the effects of events on user commitment in the *BLM* online movement.

How likely are users to re-participate? The results indicate that both police violence events and protests have a time-dependent effect on users' likelihood of re-participation. We find that, in the long term, users who joined the movement during police violence and protest periods are significantly more likely to come back than those who joined during other times. Figure 2A shows that, compared to users who came during other times, the odds of re-participating in the movement within $8-10$ weeks are nearly doubled for users who started in violence periods ($p < 10^{-10}$), while the odds of re-participation for users who joined during protests is 50% higher than the baseline group (other times) ($p < 10^{-10}$).

However, in the short term, the odds of re-participating in the movement decrease by about 40% for users in the protest group over the baseline ($p < 10^{-10}$), and users in the other two groups are comparable overall with no clear dominance of one group over the other in terms of the likelihood of re-participation.

How frequently and how persistently do they re-participate? Merely assessing the presence or absence of continued participation in an activist movement is not enough to measure user dedication. We thus further examine users' commitment by investigating their participation frequency and persistence, namely the number of tweets a user has posted and the number of days that he has participated in a given time window. The latter complements the former by capturing more nuanced participation dynamics—some users can post up to hundreds of tweets a day but stay silent thereafter.

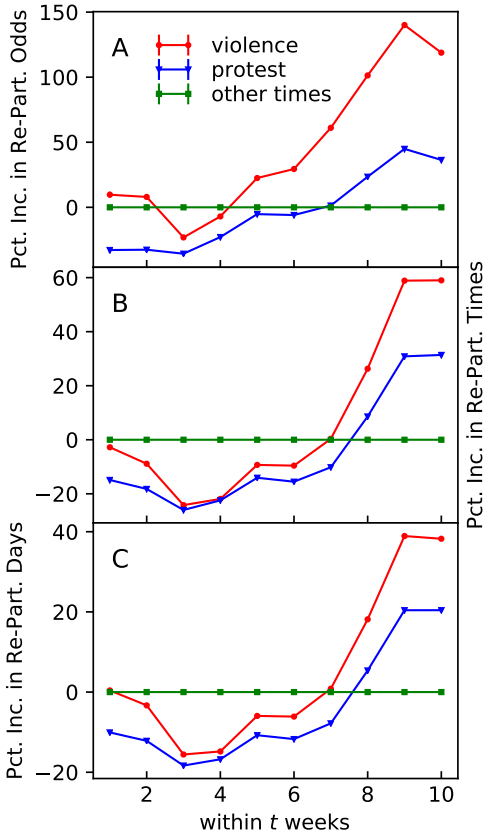


Figure 2: (A) The percentage increase in odds of re-participation within t weeks for users who joined during police violence and protest periods over those who joined during other times. The error bars indicate a 95% confidence interval. (B) The same as (A), but for the percentage increase in users' re-participation times (Frequency dimension). (C) The same as (A), but for the percentage increase in users' re-participation days (Persistence dimension).

We find that police violence events and protests have similar two-fold effects on users' future commitment. According to Figure 2 B and Figure 2C, in the long term, joining the movement during violence periods contributes to an approximate 60% increase in re-participation frequency and a 40% increase in persistence over those recruited during other times, and protest group experiences about a 25% and a 20% increase, respectively. In the short term, however, the two groups have a $10\% - 20\%$ decrease in the frequency and the persistence dimensions compared to other times group.

Discussion of findings. Overall, we find that police violence events and heightened protests significantly affect individual participation in the *BLM* online movement. These events have both short and long term effects on the dynamics of user commitment. One possible explanation is that users who joined during other times are

self-motivated and thus are more active in the initial periods, but their energy decreases over time. On the other hand, arriving to the movement during high activity events, when the entire movement is more energized, may influence users to be less active in the short term as they process the large volume of information they are being exposed to, but more active in the long term. Indeed, studies have shown that attending protests leads to higher long term political and protest participation [29, 39]. Another plausible reason is that people who joined during different periods may respond differently to changes in the movement composition over time [26, 51]. Alternatively, people joining the movement during different types of events can have different motivations and thus different responses to insufficient gratification that affect long term movement commitment [47]. We acknowledge that future work is needed to determine which, if any, of these factors drive the time-dependent participation response.

5 PATTERNS OF SOCIAL INTERACTIONS

Individual participation is only one component of OSMs. The interactions among movement participants make the community strong and robust [19]. It has been suggested that crowds display higher strong tie interactions and interpersonal exchanges during shocks and societal upheavals [19, 37]. But to what extend do external events affect tie strength over time? In this section, we characterize the formation and maintenance of social ties formed during different types of offline events in the *BLM* online movement.

Specifically, we consider retweets as social interactions between participants involved in the movement. For a social tie $e_{i,j} = (u_i, u_j)$, we define $D_{e_{i,j}}$ as the first time u_i retweeted a tweet by u_j that contains one of the hashtags. We measure tie strength using its interaction intensity within t weeks (the tracking window) starting from $D_{e_{i,j}}$ along three dimensions similar to the ones we applied to user re-participation. The first dimension is a binary variable $y_{e_{i,j}}$ indicating if u_i retweeted u_j again within t weeks ($e_{i,j}$ recurred or not). The second one is the number of times $f_{e_{i,j}}$ that u_i retweeted u_j again within t weeks (the recur frequency of $e_{i,j}$). The third one is the number of days $p_{e_{i,j}}$ that u_i retweeted u_j again within t weeks (the recur persistence of $e_{i,j}$). These three dimensions can provide a rich description of the strength of social interactions.

Similar to user participation, many factors can influence the patterns of social interactions. To test whether the creation time (event type) of a tie is related to its recur patterns, we leverage a prediction framework to control for possible confounding factors. We follow the same pipeline for each dimension as in Section 4 (Eq. 1 for the first dimension and Eq. 2 for the other two).

The 20 independent variables are the same in all regressions, which are listed in Table 3. We define D_0 as the time of the first tweet in our dataset. Variables 2−20 are used as controls in the regressions, which capture the variations between ties' creation time in the movement (4, 19, 20), the tendency for users to retweet (11, 13), the tendency for users to be retweeted (12, 14), users' activity level prior to and after tie creation ($5-10, 17, 18$), number of users they retweeted in common (16), and users previous interactions (15), etc.

The Variance Inflation Factor (VIF) of all numerical variables ($4-20$) are below 5 in a multicollinearity test. We standardize all numerical variables before performing the regression. Again, we train 10 models for each dimension by varying the tracking window from $t = 1$ to $t = 10$. We focus on the same set of social ties whose creation time $D_{e_{i,j}}$ allow us to track their future interactions for at least 10 weeks in the dataset. In total, we obtain 17 million social ties, among which 41% were created during police violence events, 53% during protest periods, and 6% in other times.

Table 3: Independent variables for predicting tie strength.

1	the event type of $D_{e_{i,j}}$
2	the event type of D_{u_i}
3	the event type of D_{u_j}
4	the week number of $D_{e_{i,j}}$
5	the # protest tweets of u_i in $[D_0, D_{e_{i,j}}]$
6	the fraction of (5) in *violence*
7	the fraction of (5) in *protest*
8	the # protest tweets of u_j in $[D_0, D_{e_{i,j}}]$
9	the fraction of (8) in *violence*
10	the fraction of (8) in *protest*
11	the times u_i has retweeted others in $[D_0, D_{e_{i,j}}]$
12	the times u_i has been retweeted by others in $[D_0, D_{e_{i,j}}]$
13	the times u_j has retweeted others in $[D_0, D_{e_{i,j}}]$
14	the times u_j has been retweeted by others in $[D_0, D_{e_{i,j}}]$
15	the times u_j has retweeted u_i in $[D_0, D_{e_{i,j}}]$
16	the # users u_i and u_j have retweeted in common in $[D_0, D_{e_{i,j}}]$
17	the # protest tweets of u_i in t weeks from $D_{e_{i,j}}$
18	the # protest tweets of u_j in t weeks from $D_{e_{i,j}}$
19	the fraction of days labeled as *violence* in t weeks from $D_{e_{i,j}}$
20	the fraction of days labeled as *protest* in t weeks from $D_{e_{i,j}}$
Note: Variables $5-20$ are calculated based on our dataset.	

We apply the same procedure to the first independent variable as we did in Section 4. We measure the relative change in the strength for social ties formed during police violence events and protests over those in the baseline group. The results are shown in Figure 3.

How likely are interactions to recur? We find that social interactions started in police violence periods are significantly more likely to be sustained in the long term than those created during other times. Figure 3A shows that, within $8-10$ weeks, the odds of recur for social ties created in violence periods are $10\%-20\%$ higher than that of other times group ($p < 10^{-15}$). However, the short term differences between two groups are less pronounced as there is oscillation between them. On the other hand, protests have a more unified influence on the maintenance of social interactions, i.e., irrespective of the time effect, communication bonds established in protests are less likely to recur than do those started in other times (about 10% decrease overall, $p < 10^{-9}$).

How frequently and how persistently do they recur? We further examine tie strength along the frequency and persistence dimensions. Figure 3B and 3C show that the overall influence of police violence events and protests on the recur frequency and persistence of social ties are similar to what we observe in the first dimension, but the result also indicates that violence events and protests have marginal influence on the sustainability of social ties compared to the baseline group (only 1% increase or decrease, $p < 10^{-10}$). This may be explained by the fact that, in general, people communicate

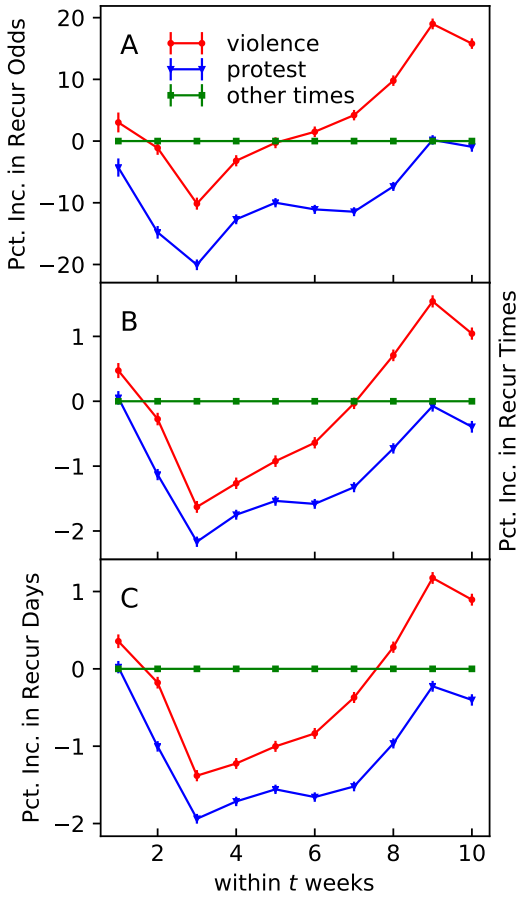


Figure 3: (A) The percentage increase in odds of recur for social interactions started during police violence and protest events over those created during other times. The error bars indicate a 95% confidence interval. (B) The same as (A), but for the percentage increase in recur frequency. (C) The same as (A), but for the percentage increase in recur persistence.

less selectively and form short-lived connections during political upheavals [38], thus reducing the influence of external events on tie strength and its evolution.

Discussion of findings. In summary, we find that communications started in police violence events are more likely to be maintained over time than interactions formed in normal times, which supports the idea that communities bond and build strong ties during shocks [15, 19, 37]. However, social ties formed during protests recur less likely, less frequently and less persistently than do those established in other times. We conjecture that social connections forged during protests are more about informational support which may be less likely to persist compared to emotional support [10, 49].

6 CONCLUSION

We conduct a large-scale empirical investigation of crowd dynamics in the *BLM* online movement using a combination of social media data and a comprehensive list of offline events. By analyzing user participation and social interactions through the lens of different types of events, we are able to reveal the dynamics of online crowd behavior and its relationship to offline events in a politically contentious movement around race.

First, we find that both police violence events and intensified protests have time-dependent effects on users' future commitment. In the long term, users who joined the movement during two type of events are significantly more likely to re-participate than those who joined during other times. However, in the short term, joining the movement in these events are associated with decreased future commitment. **Second**, we find that police violence events and protests have different effects on the continuation of social interactions formed during these periods, with the former showing both long-term and short-term effects while the latter displaying a more unified pattern. Specifically, we find that communications started in police violence events are significantly more likely to be sustained in the long run than those started in other times. But in the short term, they are less likely to be maintained. However, regardless of the time, social ties established during protests are less likely to recur compared to other times group.

Implications. Our study has implications for policymakers, movement organizers, and online social movement observers. These escalated events, unexpected or expected, function as driving forces that organize people to participate in the movement over time. They also play a great role in user retention and motivating social interactions. Knowing about these patterns, movement organizers and organizations may allocate more resources on the recruitment process during normal times. A social movement similar to the *BLM* would also try to encourage newcomers who arrived when no special event was happening to sustain participation in the long term, since they are known to drop out over time. Similarly, cultivating stronger social relationships formed during protests may be an effective way to stimulate sustained communications and facilitate information sharing about the movement in the future. Finally, this study provides insights to research on collective behavior and crowd response in the face of bursting events.

Limitations and future work. This work has limitations. First, we measure user commitment solely based on twitter dataset. However, it's unclear whether individuals who participated in the movement at its initial stage did not divert their participation to other places that are not captured in our data. Second, users may participate on Twitter in ways other than using hashtags. Third, we only study one online social movement, which limits the generalizability of our findings to other type of movements. Furthermore, although we find strong correlation between online crowd dynamics and different types of offline events, we cannot make causal claims based on these findings. Investigating whether users were intrinsically different or they were just influenced by the events when they joined the movement is an open direction for future research.

ACKNOWLEDGMENTS

This research was partly supported by the National Science Foundation Grants No. IIS-1617820 and IIS-1815875. We would like to thank David Jurgens for helpful discussion, and Lia Bozarth for data preparation.

REFERENCES

[1] James P Bagrow, Dashun Wang, and Albert-Laszlo Barabasi. 2011. Collective response of human populations to large-scale emergencies. *PLOS ONE* 6, 3 (2011).

[2] Pablo Barberá, Ning Wang, Richard Bonneau, John T Jost, Jonathan Nagler, Joshua Tucker, and Sandra González-Bailón. 2015. The critical periphery in the growth of social protests. *PLOS ONE* 10, 11 (2015).

[3] Marco T Bastos, Dan Mercea, and Arthur Charpentier. 2015. Tents, tweets, and events: The interplay between ongoing protests and social media. *Journal of Communication* 65, 2 (2015), 320–350.

[4] Javier Borge-Holthoefer, Walid Magdy, Kareem Darwish, and Ingmar Weber. 2015. Content and network dynamics behind Egyptian political polarization on Twitter. In *CSCW*. ACM.

[5] Philip J Bostic. 2013. Social Movement. (2013). https://www.learningtogive.org/resources/social-movement

[6] Lia Bozarth and Ceren Budak. 2017. Is Slacktivism Underrated? Measuring the Value of Slacktivists for Online Social Movements. In *ICWSM*.

[7] Ceren Budak and Duncan J Watts. 2015. Dissecting the spirit of Gezi: Influence vs. selection in the Occupy Gezi movement. *Sociological Science* 2 (2015), 370–397.

[8] Michael Conover, Jacob Ratkiewicz, Matthew R Francisco, Bruno Gonçalves, Filippo Menczer, and Alessandro Flammini. 2011. Political polarization on twitter. *ICWSM* (2011).

[9] Michael D Conover, Emilio Ferrara, Filippo Menczer, and Alessandro Flammini. 2013. The digital evolution of occupy wall street. *PLOS ONE* 8, 5 (2013).

[10] Carolyn E Cutrona and Julie A Suhr. 1992. Controllability of stressful events and satisfaction with spouse support behaviors. *Communication Research* 19, 2 (1992), 154–174.

[11] Munmun De Choudhury, Shagun Jhaver, Benjamin Sugar, and Ingmar Weber. 2016. Social Media Participation in an Activist Movement for Racial Equality. In *ICWSM*.

[12] Donatella Della Porta and Alice Mattoni. 1999. *Social movements*. Wiley Online Library.

[13] Mario Diani. 2003. *Social movements and networks*. Wiley Online Library.

[14] Peter Dreier. 2012. Social Movements: How People Make History. *Mobilizing Ideas* (2012). https://mobilizingideas.wordpress.com/2012/08/01/social-movements-how-people-make-history

[15] Kimberly Falconer, Mieke Sachsenweger, Kerry Gibson, and Helen Norman. 2011. Grieving in the internet age. *New Zealand Journal of Psychology* 40, 3 (2011).

[16] Ryan J Gallagher, Andrew J Reagan, Christopher M Danforth, and Peter Sheridan Dodds. 2018. Divergent discourse between protests and counter-protests:# BlackLivesMatter and# AllLivesMatter. *PLOS ONE* 13, 4 (2018).

[17] William A Gamson. 1992. *Talking politics*. Cambridge university press.

[18] William A Gamson, Bruce Fireman, and Steven Rytina. 1982. *Encounters with unjust authority*. Dorsey Press.

[19] Kimberly Glasgow, Clayton Fink, and Jordan L Boyd-Graber. 2014. "Our Grief is Unspeakable": Automatically Measuring the Community Impact of a Tragedy. In *ICWSM*.

[20] Benjamin Gleason. 2013. # Occupy Wall Street: Exploring informal learning about a social movement on Twitter. *American Behavioral Scientist* 57, 7 (2013), 966–982.

[21] Lewis R Goldfrank, Allison M Panzer, Adrienne Stith Butler, et al. 2003. *Preparing for the psychological consequences of terrorism: A public health strategy*. National Academies Press.

[22] Sandra González-Bailón, Javier Borge-Holthoefer, and Yamir Moreno. 2013. Broadcasters and hidden influentials in online protest diffusion. *American Behavioral Scientist* 57, 7 (2013), 943–965.

[23] Sandra González-Bailón, Javier Borge-Holthoefer, Alejandro Rivero, and Yamir Moreno. 2011. The dynamics of protest recruitment through an online network. *Scientific Reports* 1 (2011).

[24] Jelani Ince, Fabio Rojas, and Clayton A Davis. 2017. The social media response to Black Lives Matter: how Twitter users interact with Black Lives Matter through hashtag use. *Ethnic and Racial Studies* 40, 11 (2017), 1814–1830.

[25] James M Jasper. 1998. The emotions of protest: Affective and reactive emotions in and around social movements. *Sociological Forum* 13, 3 (1998), 397–424.

[26] Bert Klandermans. 1994. Transient identities? Membership patterns in the Dutch peace movement. *New social movements: From ideology to identity* (1994), 168–184.

[27] Fannie Liu, Denae Ford, Chris Parnin, and Laura A. Dabbish. 2017. Selfies as Social Movements: Influences on Participation and Perceived Impact on Stereotypes. *PACMHCI* 1 (2017).

[28] Gilad Lotan, Erhardt Graeff, Mike Ananny, Devin Gaffney, Ian Pearce, et al. 2011. The Arab Spring| the revolutions were tweeted: Information flows during the 2011 Tunisian and Egyptian revolutions. *International Journal of Communication* 5 (2011), 31.

[29] Doug McAdam. 1989. The biographical consequences of activism. *American sociological review* (1989), 744–760.

[30] Doug McAdam, John D McCarthy, and Mayer N Zald. 1996. *Comparative perspectives on social movements: Political opportunities, mobilizing structures, and cultural framings*. Cambridge University Press.

[31] Evgeny Morozov. 2009. The brave new world of slacktivism. *Foreign Policy* 19, 05 (2009).

[32] Christine Ogan and Onur Varol. 2017. What is gained and what is left to be done when content analysis is added to network analysis in the study of a social movement: Twitter use during Gezi Park. *Information, Communication & Society* 20, 8 (2017), 1220–1238.

[33] Alexandra Olteanu, Ingmar Weber, and Daniel Gatica-Perez. 2016. Characterizing the demographics behind the# blacklivesmatter movement. In *2016 AAAI Spring Symposium Series*.

[34] Leysia Palen and Sarah Vieweg. 2008. The emergence of online widescale interaction in unexpected events: assistance, alliance & retreat. In *CSCW*. ACM.

[35] Pamela Paxton. 2002. Social capital and democracy: An interdependent relationship. *American Sociological Review* (2002), 254–277.

[36] Tom Postmes and Suzanne Brunsting. 2002. Collective action in the age of the Internet: Mass communication and online mobilization. *Social Science Computer Review* 20, 3 (2002), 290–301.

[37] Daniel M Romero, Brian Uzzi, and Jon Kleinberg. 2016. Social networks under stress. In *WWW*.

[38] Cuihua Shen, Peter Monge, and Dmitri Williams. 2014. The evolution of social ties online: A longitudinal study in a massively multiplayer online game. *Journal of the Association for Information Science and Technology* 65, 10 (2014), 2127–2137.

[39] Darren E Sherkat and T Jean Blocker. 1997. Explaining the political and personal consequences of protest. *Social Forces* 75, 3 (1997), 1049–1070.

[40] Emma S Spiro and Andrés Monroy-Hernández. 2016. Shifting Stakes: Understanding the Dynamic Roles of Individuals and Organizations in Social Media Protests. *PLOS ONE* 11, 10 (2016).

[41] Kate Starbird and Leysia Palen. 2012. (How) will the revolution be retweeted?: information diffusion and the 2011 Egyptian uprising. In *CSCW*. ACM.

[42] Zachary C Steinert-Threlkeld, Delia Mocanu, Alessandro Vespignani, and James Fowler. 2015. Online social networks and offline protest. *EPJ Data Science* 4, 1 (2015).

[43] Leog Stewart, Ahmer Arif, A Conrad Nied, Emma S Spiro, and Kate Starbird. 2017. Drawing the Lines of Contention: Networked Frame Contests Within# BlackLivesMatter Discourse. *Proc. ACM Human Computer Interaction* 1 (2017).

[44] Zeynep Tufekci and Christopher Wilson. 2012. Social media and the decision to participate in political protest: Observations from Tahrir Square. *Journal of Communication* 62, 2 (2012), 363–379.

[45] Marlon Twyman, Brian C Keegan, and Aaron Shaw. 2017. Black Lives Matter in Wikipedia: Collective memory and collaboration around online social movements. In *CSCW*.

[46] Robert J Ursano, Carol S Fullerton, Tzu-Cheg Kao, and Vivek Bhartiya. 1995. Longitudinal assessment of posttraumatic stress disorder and depression after exposure to traumatic death. *Journal of Nervous and Mental Disease* (1995).

[47] Jacquelien Van Stekelenburg and Bert Klandermans. 2017. Individuals in movements: A social psychology of contention. In *Handbook of social movements across disciplines*. Springer, 103–139.

[48] Onur Varol, Emilio Ferrara, Christine L Ogan, Filippo Menczer, and Alessandro Flammini. 2014. Evolution of online user behavior during a social upheaval. In *WebSci*. ACM.

[49] Zijian Wang and David Jurgens. 2018. It's going to be okay: Measuring Access to Support in Online Communities. In *EMNLP*.

[50] Ingmar Weber, Venkata R Kiran Garimella, and Alaa Batayneh. 2013. Secular vs. islamist polarization in egypt on twitter. In *Proceedings of the ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 290–297.

[51] Nancy Whittier. 1997. Political generations, micro-cohorts, and the transformation of social movements. *American Sociological Review* (1997), 760–778.

[52] Donghyeon Won, Zachary C Steinert-Threlkeld, and Jungseock Joo. 2017. Protest Activity Detection and Perceived Violence Estimation from Social Media Images. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM.

[53] Ark Fangzhou Zhang, Danielle Livneh, Ceren Budak, Lionel P. Robert, and Daniel M. Romero. 2017. Shocking the Crowd: The Effect of Censorship Shocks on Chinese Wikipedia. In *ICWSM*.

[54] Ark Fangzhou Zhang, Danielle Livneh, Ceren Budak, Lionel P. Robert, Jr., and Daniel M. Romero. 2017. Crowd Development: The Interplay Between Crowd Evaluation and Collaborative Dynamics in Wikipedia. *PACMHCI* 1 (2017).

CnGAN: Generative Adversarial Networks for Cross-network user preference generation for non-overlapped users

Dilruk Perera
School of Computing
National University of Singapore
dilruk@comp.nus.edu.sg

Roger Zimmermann
School of Computing
National University of Singapore
rogerz@comp.nus.edu.sg

ABSTRACT

A major drawback of cross-network recommender solutions is that they can only be applied to users that are overlapped across networks. Thus, the non-overlapped users, which form the majority of users are ignored. As a solution, we propose CnGAN, a novel multi-task learning based, encoder-GAN-recommender architecture. The proposed model synthetically generates source network user preferences for non-overlapped users by learning the mapping from target to source network preference manifolds. The resultant user preferences are used in a Siamese network based neural recommender architecture. Furthermore, we propose a novel user-based pairwise loss function for recommendations using implicit interactions to better guide the generation process in the multi-task learning environment. We illustrate our solution by generating user preferences on the Twitter source network for recommendations on the YouTube target network. Extensive experiments show that the generated preferences can be used to improve recommendations for non-overlapped users. The resultant recommendations achieve superior performance compared to the state-of-the-art cross-network recommender solutions in terms of accuracy, novelty and diversity.

CCS CONCEPTS

• **Information systems → Recommender systems**; **Personalization**; **Multimedia and multimodal retrieval**.

KEYWORDS

Cross-network Recommendations; Generative Adversarial Networks; Collaborative Filtering; Deep learning; Implicit Feedback

ACM Reference Format:
Dilruk Perera and Roger Zimmermann. 2019. CnGAN: Generative Adversarial Networks for Cross-network user preference generation for non-overlapped users. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313733

1 INTRODUCTION

Cross-network recommender systems utilize auxiliary information from multiple source networks to create comprehensive user profiles for recommendations on a target network. Therefore, unlike traditional recommender solutions which are limited to information within a single network, cross-network solutions are more robust against cold-start and data sparsity issues [17]. For example, the additional information from Twitter and Flicker, increased recommender precision on Delicious by 10% [1]. Similarly, the transfer of information from various source networks to target networks such as Google+ to YouTube [10], Twitter to YouTube [20, 24] and Wikipedia to Twitter [18] have consistently improved recommender accuracy. Furthermore, cross-network solutions allow user preferences to be captured from diverse perspectives, which increases the overall recommender quality in terms of diversity and novelty [20, 21]. However, despite the growing success of cross-network recommender solutions, the majority of existing solutions can only be applied to users that exist in multiple networks (overlapped users). The remaining non-overlapped users, which form the majority are unable to enjoy the benefits of cross-network solutions.

Deep learning and generative modeling techniques have been successfully used in the recommender systems domain in the past few years. For example, restricted Boltzmann machines [26], Autoencoders [19], Hidden Markov Models [25] and Recurrent Neural Networks (RNN) [13] are some of the models used for rating prediction tasks. Recent advancements in Generative Adversarial Networks (GANs) have also drawn recommender systems researchers to apply the new minimax game framework to information retrieval. For example, IRGAN [28] and RecGAN [5] were designed to predict relevant documents to users. Existing solutions use GAN to generate simple rating values [5, 28, 30] or synthetic items (e.g., images [14]). In contrast, we use GAN to generate complex user data (i.e., user preferences on the source network).

We propose a novel GAN model (CnGAN) to generate cross-network user preferences for non-overlapped users. Unlike the standard GAN model, we view the learning process as a mapping from target to source network preference manifolds. The proposed model solves two main tasks. First, a generator task learns the corresponding mapping from target to source network user preferences and generates auxiliary preferences on the source network. Second, a recommender task uses the synthetically generated preferences to provide recommendations for users who only have interactions on the target network. Furthermore, we propose two novel loss functions to better guide the generator and recommender tasks. The proposed solution is used to provide recommendations for YouTube users by incorporating synthesized preferences on Twitter. However, the solution is generic and can be extended to incorporate multiple source networks.

In this paper, we first briefly provide preliminaries to the proposed model and present the proposed CnGAN solution detailing its two main components, the generator and recommender tasks. Then, we compare the proposed model against multiple baselines

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313733

to demonstrate its effectiveness in terms of accuracy, novelty and diversity. We summarize our main contributions as follows:

- To the best of our knowledge, this is the first attempt to apply a GAN based model to generate missing source network preferences for non-overlapped users.
- We propose CnGAN, a novel GAN based model which includes a novel content loss function and user-based pairwise loss function for the generator and recommender tasks.
- We carry out extensive experiments to demonstrate the effectiveness of CnGAN to conduct recommendations for non-overlapped users and improve the overall quality of recommendations compared to state-of-the-art methods.

2 MODEL

2.1 Model Preliminaries

2.1.1 Bayesian Personalized Ranking (BPR):. Learning user preferences from implicit feedback is challenging since implicit feedback does not indicate preference levels toward items, it only indicates the presence or absence of interactions with items. BPR [23] was proposed to rank items based on implicit feedback using a generic optimization criterion. BPR is trained using pairwise training instances $S = \{(u, i, j) | u \in U \wedge i \in I_{u+} \wedge j \in I \setminus I_{u+}\}$, where I is the set of all items and I_{u+} is the set of items user u has interacted with. BPR uses a pairwise loss function to rank interacted items higher than non-interacted items, and maximizes their margin using a pairwise loss as follows:

$$L_{BPR}(S) = \sum_{(u,i,j)\in S} -\ln \sigma(\hat{r}_{uij}) + \lambda_\Theta \|\Theta\|^2 \quad (1)$$

where $\hat{r}_{uij} = \hat{r}_{ui} - \hat{r}_{uj}$, Θ are model parameters, $\sigma(x) = e^x / e^{x+1}$ is a sigmoid function and λ_Θ are regularization parameters.

2.1.2 Feature extraction on a continuous space: CnGAN is trained using target network interactions of each user at each time interval and their corresponding source network interactions. We used topic modeling to capture user interactions on a continuous topical space since CnGAN is a GAN based model and requires inputs in a continuous space. Let $u_o \in U = [Tn_u^t, Sn_u^t]$ denote an overlapped user at time t, where Tn_u^t and Sn_u^t are target and source network interactions spanned over $T = [1, \ldots, t]$ time intervals. A non-overlapped user $u_{no} \in U = [Tn_u^t]$ is denoted only using target network interactions. We used YouTube as the target and Twitter as the source network to conduct video recommendations on YouTube. Therefore, Tn_u^t is the set of interacted videos (i.e., liked or added to playlists) and Sn_u^t is the set of tweets. We assumed that each interaction (video or tweet) is associated with multiple topics, and extracted the topics from the textual data associated with each interaction (video titles, descriptions and tweet contents). We used the Twitter-Latent Dirichlet Allocation (Twitter-LDA) [32] for topic modeling since it is most effective against short and noisy contents. Based on topic modeling, each user u on the target network is represented as a collection of topical distributions $tn_u = \{\boldsymbol{tn_u^1}; \ldots; \boldsymbol{tn_u^t}\} \in \mathbb{R}^{T \times K^t}$ over T time intervals, where K^t is the number of topics. Each vector $\boldsymbol{tn_u^t} = \{tn_u^{t,1}; \ldots; tn_u^{t,K^t}\} \in \mathbb{R}^{K^t}$ is the topical distribution at time t, where $tn_u^{t,k} = \hat{tn}_u^{t,k} / \sum_{c=1}^{K^t} \hat{tn}_u^{t,c}$ is the relative frequency of topic k, and $\hat{tn}_u^{t,k}$ is the absolute frequency of the corresponding topic. Similarly, on the source network, interactions are represented as $sn_u = \{\boldsymbol{sn_u^1}; \ldots; \boldsymbol{sn_u^t}\} \in \mathbb{R}^{T \times K^t}$. The resultant topical frequencies indicate user preference levels towards corresponding K^t topics. Therefore, $u = \{tn_u, sn_u\} \in \mathbb{R}^{T \times 2K^t}$ represents user preferences over T time intervals on a continuous topical space, and forms the input to the proposed model.

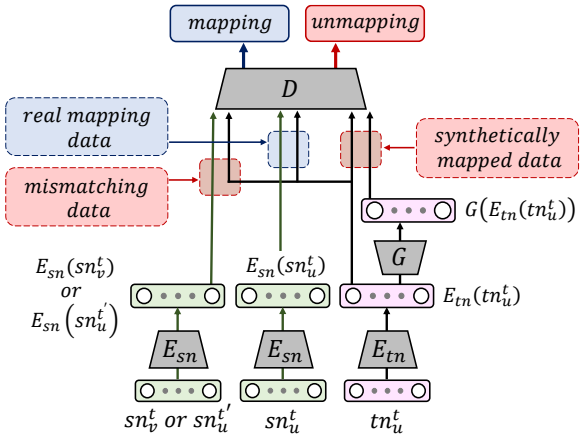


Figure 1: Adversarial learning process between *E*, *D* and *G*.

2.2 Generator Task

The generator task learns the function that maps the target network preferences manifold to the source network preferences manifold using Encoders (E), Discriminator (D) and Generator (G).

2.2.1 Encoders: User preferences captured as topical distribution vectors ($\boldsymbol{tn_u^t}$ and $\boldsymbol{sn_u^t}$) become sparse when $\boldsymbol{K^t}$ is set to a sufficiently large value to capture finer level user preferences, and/or the number of interactions at a time interval is low. One of the most effective methods to train machine learning models with highly sparse data is to model interactions between input features [7] (e.g., using neural networks [12]). Therefore, we used two neural encoders E_{tn} and E_{sn} for target and source networks to transform the input topical distributions to dense latent encodings. The resultant encodings are represented as $E_{tn}(tn_u) \in \mathbb{R}^{T \times En}$ and $E_{sn}(sn_u) \in \mathbb{R}^{T \times En}$, where En is the dimensionality of the latent space. These encodings form the input to the generator task.

2.2.2 Generator task formulation: For a non-overlapped user u, let $E_{tn}(\boldsymbol{tn_u^t})$ denote the target network encoding at time interval t. The generator task aims to synthetically generate the mapping source network encoding that closely reflects the missing source network encoding $E_{sn}(\boldsymbol{sn_u^t})$. Hence, G uses target encoding $E_{tn}(\boldsymbol{tn_u^t})$ and attempts to generate the mapping source encoding to fool D. Similarly, D tries to differentiate between real source encoding $E_{sn}(\boldsymbol{sn_u^t})$ and generated encoding $G(E_{tn}(\boldsymbol{tn_u^t}))$. Note that, we refer to the actual and generated target and source network encodings of non-overlapped users $\left(E_{tn}(\boldsymbol{tn_u^t}), G(E_{tn}(\boldsymbol{tn_u^t}))\right)$ as *synthetically mapped data* and the actual target and source network encodings of overlapped users $\left(E_{tn}(\boldsymbol{tn_u^t}), E_{sn}(\boldsymbol{sn_u^t})\right)$ as *real mapping data.*

2.2.3 Discriminator: Analogous to D in standard GANs, the real and synthetically mapped pairs could be used to learn D. However, D may only learn to differentiate between actual and generated source network encodings in a given pair of inputs, without learning to check if the given pair is a mapping pair. Therefore, during training, we input mismatching source and target network encodings to ensure that an effective mapping is learned. We modified the loss function for D to minimize output scores for mismatching pairs and maximize output scores for matching pairs. For a given target network encoding, we drew mismatching source encodings only from real data to avoid potential biases. Mismatching pairs were created by pairing real encodings from different users at the same time interval or by pairing encodings from the same user at different time intervals (see Figure 1). Accordingly, the loss function for D is formed as follows:

$$\begin{aligned}
&\max_{E_{tn},E_{sn},D} V(E_{tn},E_{sn},D) \\
&= \mathbb{E}_{\boldsymbol{tn_u^t},\boldsymbol{sn_u^t}\sim p_{data}(tn,sn)} L_{real}\Big(E_{tn}(\boldsymbol{tn_u^t}), E_{sn}(\boldsymbol{sn_u^t})\Big) \\
&+\mathbb{E}_{\boldsymbol{tn_u^t}\sim p_{data}(tn)} L_{fake}\Big(E_{tn}(\boldsymbol{tn_u^t}), G\Big(E_{tn}(\boldsymbol{tn_u^t})\Big)\Big) \\
&+\mathbb{E}_{\boldsymbol{tn_u^t},\overline{\boldsymbol{sn}}_{\boldsymbol{u}}^{\boldsymbol{t}}\sim \overline{p}_{data}(tn,\overline{sn})} L_{mismatch}\Big(E_{tn}(\boldsymbol{tn_u^t}), E_{sn}(\overline{\boldsymbol{sn}}_{\boldsymbol{u}}^{\boldsymbol{t}})\Big)
\end{aligned} \tag{2}$$

where $p_{data}(tn,sn)$ and $\overline{p}_{data}(tn,\overline{sn})$ are matching and mismatching target and source network topical distributions, $p_{data}(tn)$ are the target network topical distributions and $G(x)$ is the generated matching source network encoding for the given target network encoding x. Furthermore, $L_{real}(x,y) = [\log(D(x,y))]$, $L_{fake}(x,y) = [1-\log(D(x,y))]$ and $L_{mismatch}(x,y) = [1-\log(D(x,y))]$, where $D(x,y)$ is the probability that x and y are matching target and source network encodings. Therefore, D and E work together to maximize the discriminator value function $V(E_{tn},E_{sn},D)$. The encoders (E_{tn} and E_{sn}) assist the process by encoding the topical distributions that are optimized for learning D (e.g., extract encodings that express the latent mapping between target and source networks).

2.2.4 Generator: Once D and E are trained, they adversarially guide G by effectively identifying real and synthetically generated data. Specifically, G takes in a target network encoding $E_{tn}(\boldsymbol{tn_u^t})$ and synthetically generates the mapping source network encoding that resemble $E_{sn}(\boldsymbol{sn_u^t})$ drawn from real mapping data.

GAN models are typically used to generate real-like images from random noise vectors. For a given noise vector, different GANs can learn multiple successful mappings from the input space to the output space. In contrast, CnGAN aims to learn a specific mapping from target to source network encodings. Hence, for given pairs of real mapping data $(\boldsymbol{tn_u^t},\boldsymbol{sn_u^t})$, G minimizes an additional content loss between the generated $G(E_{tn}(\boldsymbol{tn_u^t}))$ and real $E_{sn}(\boldsymbol{sn_u^t})$ source encodings to provide additional guidance for G. Therefore, we formulated the loss function for G as follows:

$$\begin{aligned}
&\min_G V(G) = \mathbb{E}_{\boldsymbol{tn_u^t}\sim p_{data}(tn)} L_{fake}\Big(E_{tn}(\boldsymbol{tn_u^t}), G\Big(E_{tn}(\boldsymbol{tn_u^t})\Big)\Big) \\
&+\mathbb{E}_{\boldsymbol{sn_u^t},\boldsymbol{tn_u^t}\sim p_{data}(tn,sn)} L_{content}\Big(E_{sn}(\boldsymbol{sn_u^t}), G\Big(E_{tn}(\boldsymbol{tn_u^t})\Big)\Big)
\end{aligned} \tag{3}$$

where $L_{content}(x,y) = \|x,y\|_1$ is the content loss computed as the $\ell 1$-norm loss between real and generated encodings.

2.3 Recommender Task

The recommender task uses the generated preferences to conduct recommendations for non-overlapped users.

2.3.1 Recommender task formulation: Let $E_{tn}(\boldsymbol{tn_u^t}), G(E_{tn}(\boldsymbol{tn_u^t}))$ denote a non-overlapped user u at time t based on synthetically mapped data, which reflects current preferences. To capture previous preferences, we used a previous interaction vector $R_u^{t-} \in \mathbb{R}^M$, where M is the number of items. Each element $r_{u,i}^{t-} \in \boldsymbol{R_u^{t-}}$ is set to 1 if the user had an interaction with item i before time t, and 0 if otherwise. Thus, we formulated the recommender task as a time aware Top-N ranking task where, current and previous user preferences are used to predict a set of N items that the user is most likely interact with, in the next time interval $t+1$.

2.3.2 Recommender loss: Typically, BPR optimization is performed using Stochastic Gradient Descent (SGD) where, for each training instance (u,i,j), parameter (Θ) updates are performed as follows:

$$\begin{aligned}
\Theta &\leftarrow \Theta - \alpha \frac{\partial}{\partial \Theta} L_{BPR}(S) \\
\Theta &\leftarrow \Theta + \alpha\left(\frac{e^{-\hat{r}_{uij}}}{1+e^{-\hat{r}_{uij}}} \cdot \frac{\partial}{\partial \Theta}\hat{r}_{uij} + \lambda_\Theta \Theta\right)
\end{aligned} \tag{4}$$

where α is the learning rate, $\hat{r}_{uij} = \hat{r}_{ui} - \hat{r}_{uj}$ is the pairwise loss for a given (u,i,j) triplet and $\hat{r}_{ui}$ is the predicted rating for user u and item i. Since BPR is a generic optimization criterion, $\hat{r}_{ui}$ can be obtained using any standard collaborative filtering approach (e.g., MF). Considering the rating prediction function in MF ($\hat{r}_{ui} = \sum_{f=1}^{K} \boldsymbol{w_{uf}} \cdot \boldsymbol{h_{if}}$, where $\boldsymbol{w_{uf}} \in \mathbb{R}^K$ and $\boldsymbol{h_{if}} \in \mathbb{R}^K$ are latent representations of user u and item i), $\hat{r}_{uij}$ for MF based BPR optimization can be defined as $\hat{r}_{uij} = \sum_{f=1}^{K} w_{uf} \cdot (h_{if} - h_{jf})$.

For each (u,i,j) instance, three updates are performed when $\Theta = w_{uf}$, $\Theta = h_{if}$ and $\Theta = h_{jf}$. For each update on the latent user representation (w_{uf}), two updates are performed on the latent item representations (h_{if} and h_{jf}). Hence, the training process is biased towards item representation learning, and we term the standard BPR optimization function as an *item-based* pairwise loss function. Since our goal is to learn effective user representations to conduct quality recommendations, we introduce a *user-based* pairwise loss function (R loss), which is biased towards user representation learning.

We used a fixed time-length sliding window approach in the training process where, at each time interval, the model is trained using the interactions within the time interval (see Section 3.2.1). Thus, we denote the training instances for the new user-based BPR at each time interval t as $S^t = \{(u,v,i) | u \in U_{i+}^t \wedge v \in U \setminus U_{i+}^t \wedge i \in I\}$ where U_{i+}^t is the set of users who have interactions with item i, at time t. Compared to (u,i,j) in the standard item-based BPR, user-based BPR contains (u,v,i) where, for item i at time t, user u has an interaction and user v does not. Thus, intuitively, the predictor should assign a higher score for (u,i) compared to (v,i). Accordingly, we defined the user-based loss function to be minimized as follows:

$$L_{UBPR}(S^t) = \sum_{(u,v,i)\in S^t} -\ln \sigma(\hat{r}_{uvi}^t) + \lambda_\Theta \|\Theta\|^2 \tag{5}$$

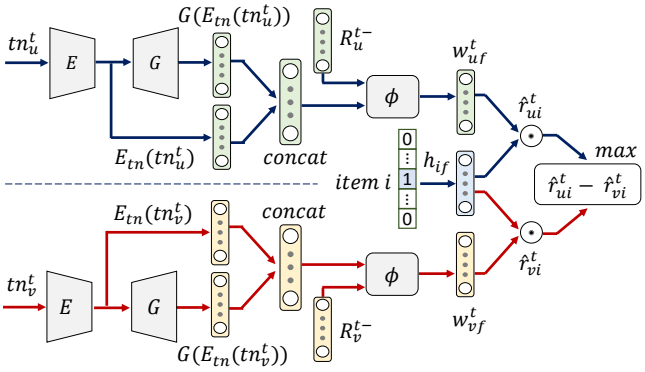


Figure 2: Recommender task architecture as a Siamese network.

where $\hat{r}_{uvi}^t = \hat{r}_{ui}^t - \hat{r}_{vi}^t = \sum_{f=1}^{K}(w_{uf}^t - w_{vf}^t)h_{if}$. During optimization, Θ updates are performed as follows:

$$\Theta \leftarrow \Theta + \alpha\left(\frac{e^{-\hat{r}_{uvj}^t}}{1+e^{-\hat{r}_{uvj}^t}} \cdot \frac{\partial}{\partial\Theta}\hat{r}_{uvj}^t + \lambda_\Theta\Theta\right) \tag{6}$$

For different Θ values, the following partial derivations are obtained:

$$\frac{\partial}{\partial\Theta}\hat{r}_{uvj}^t = \begin{cases} h_{if}, \text{if } \Theta = w_{uf}^t \\ -h_{if}, \text{if } \Theta = w_{vf}^t \\ (w_{uf}^t - w_{vf}^t), \text{if } \Theta = h_{if} \\ 0, \text{otherwise} \end{cases} \tag{7}$$

Since the user-based BPR performs two user representation updates for each training instance, it is able to efficiently guide G to learn user representations during training.

2.3.3 Recommender architecture: We used a Siamese network for the recommender architecture since it naturally supports pairwise learning (see Figure 2). Given a non-overlapped user u with current preferences $E_{tn}(\boldsymbol{tn_u^t})$, $G(E_{tn}(\boldsymbol{tn_u^t}))$ and previous preferences $\boldsymbol{R_u^{t-}}$ at time t, we learned a transfer function Φ, which maps the user to the latent user space $w_{u,f}^t$ for recommendations as follows:

$$w_{u,f}^t = \Phi(E_{tn}(\boldsymbol{tn_u^t}), G(E_{tn}(\boldsymbol{tn_i^t})), \boldsymbol{R_u^{t-}}) \tag{8}$$

The transfer function Φ is learned using a neural network since neural networks are more expressive and effective than simple linear models. Accordingly, for a non-overlapped user u at time t, the predicted rating $\hat{r}_{ui}$ for any given item i is obtained using the inner-product between the latent user and item representations as follows:

$$\hat{r}_{ui}^t = \sum_{f=1}^{K} \Phi(E_{tn}(tn_u^t), G(E_{tn}(tn_u^t)), \boldsymbol{R_u^{t-}}) \cdot \boldsymbol{h_{if}} \tag{9}$$

2.4 Multi-Task Learning (MTL)

Although the generator and recommender tasks are separate, they depend on each other to achieve a higher overall performance. The generator task increases the accuracy of the recommender by synthesizing source network encodings that effectively represent user preferences. The recommender task guides the generator to efficiently learn the target to source mapping by reducing the search space. Therefore, we trained both interrelated tasks in a MTL environment to benefit from the training signals of each other. Hence, we formulated the multi-task training objectives as follows:

$$\min_{G} \max_{E_{tn}, E_{sn}, D} V(E_{tn}, E_{sn}, G, D) \tag{10}$$

$$\min_{w_{uf}^t, w_{vf}^t, h_{if}} L_{UBPR}(S^t) \tag{11}$$

where $V(E_{tn}, E_{sn}, G, D)$ is the value function for E, G and D, $L_{UBPR}(S^t)$ is the R loss function, and w_{uf}^t, w_{vf}^t and h_{if} are model parameters. The R loss is back-propagated all the way to G since w_{uf}^t and w_{vf}^t are compositions of the functions G and Φ (see Equation 8) and the generator and recommender tasks are trained as an end-to-end process. Hence, equation 11 can be restated as follows:

$$\min_{\Phi, G, h_{if}} L_{UBPR}(S^t) \tag{12}$$

3 EXPERIMENTS

3.1 Dataset

Due to the lack of publicly available timestamped cross-network datasets, we extracted overlapped users on YouTube (target) and Twitter (source) networks from two publicly available datasets [16, 29]. We scraped timestamps of interactions and associated textual contents (video titles, descriptions and tweet contents) over a 2-year period from 1st March 2015 to 29th February 2017. In line with common practices, we filtered out users with less than 10 interactions on both networks for effective evaluations. The final dataset contained 2372 users and 12,782 YouTube videos.

3.2 Experimental Setup

The recommender systems literature often uses training and testing datasets with temporal overlaps [9], which provides undue advantages as it does not reflect a realistic recommender environment. To avoid such biases, at each time interval, the proposed model is trained using only the interactions during the current and previous time intervals. We randomly selected 50% of users as non-overlapped users and removed their source network interactions. The model was first trained offline using overlapped users (see Section 3.2.1). Then, testing and online training were conducted using both overlapped and non-overlapped users (see Section 3.2.2).

3.2.1 Offline Training: We used the sliding window approach and data from the first 16 months ($2T/3$) of overlapped users to learn the mapping from target to source network preferences for the generator task, the neural transfer function (Φ), and item latent representations (h_{if}) for the recommender task. At each training epoch, the generator task is learned first. Hence, E, D and G are trained using real mapping data and the trained G is used to generate synthetically mapped data. The generated preferences are used as inputs to the recommender task to predict interactions at the next time interval. The ground truth interactions at the same time interval are used to propagate the recommender error from the recommender to the generator task and learn parameters of both processes.

3.2.2 Testing and Online Training: We used data from the last 8 months ($2T/3$ onward) to test and retrain the model online in a simulated real-world environment. At each testing time interval t, first, mapping source network preferences are generated for non-overlapped users based on their target network preferences. Second, the synthetically mapped data for non-overlapped users are used as inputs to conduct Top-N recommendations for $t+1$ (see Equation 9). The entire model is then retrained online before moving to the next time interval, as follows: First, the recommender parameters (Φ and $\boldsymbol{h}^t_{if}$) are updated based on the recommendations for both overlapped and non-overlapped users. Second, the components of the generator task (E, D and G) are updated based on the real mapping data from overlapped users. Finally, to guide G from the training signals of the recommender, the model synthesizes mapping data for overlapped users, which are used as inputs for recommendations. The error is back-propagated from the recommender task to G, and consequently, the parameters G, Φ and $\boldsymbol{h}^t_{if}$ are updated. This process is repeated at subsequent time intervals as the model continues the online retraining process. In line with common practices in sliding window approaches, the model is retrained multiple times before moving to the next time interval.

3.3 Evaluation

We formulated video recommendation as a Top-N recommender task and predicted a ranked set of N videos that the user is most likely to interact with at each time interval. To evaluate model accuracy, we calculated the Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) [11]. Both metrics were calculated for each participating user at each time interval and the results were averaged across all users and testing time intervals.

3.3.1 Baselines. We evaluated CnGAN against single network, cross-network, linear factorization and GAN based baselines.

- **TimePop**: Recommends the most popular N items in the previous time interval to all users in the current time interval.
- **TBKNN** [8]: Time-Biased KNN computes a set of K neighbors for each user at each time interval based on complete user interaction histories, and recommends their latest interactions to the target user at the next time interval. Similar to the original work, we used several K values from 4 to 50, and the results were averaged.
- **TDCN** [20]: Time Dependent Cross-Network is an offline, MF based cross-network recommender solution, which provides recommendations only for overlapped users. TDCN learns network level transfer matrices to transfer and integrate user preferences across networks and conduct MF based recommendations.
- **CRGAN**: Due to the absence of GAN based cross-network user preference synthesizing solutions, we created Cross-network Recommender GAN, as a variation of our solution. Essentially, CRGAN uses the standard D and G loss functions and does not contain the $L_{mismatch}$ and $L_{content}$ loss components.
- **NUBPR-O** and **NUBPR-NO**: Due to the absence of neural network based cross-network recommender solutions, we created two Neural User-based BPR solutions, as variations of our solution. Essentially, both models do not contain the generator task. NUBPR-O considers both source and target network interactions and NUBPR-NO only considers target network interactions to conduct recommendations.

3.4 Model Parameters

We used Adaptive Moment Estimation (Adam) [15] for optimizations since Adam adaptively updates the learning rate during training. We set the initial learning rate to 0.1, a fairly large value for faster initial learning before the rate is updated by Adam. We used only one hidden layer for all neural architectures, and given the size of the output layer H_L, the size of the hidden layer was set to $H_L \times 2$ to reduce the number of hyper-parameters. We used the dropout regularization technique and the dropout was set to 0.4 during training to prevent neural networks from overfitting.

4 DISCUSSION

4.0.1 Offline training loss: We compared the offline training loss of D, G and R (see Figure 3). All three processes converge around 50 epochs, and compared to D and R, the drop in G is low. Therefore, we further examined the proposed G loss against the vanilla G loss for the same training process (see Figure 4). Inline with standard GANs, the vanilla G loss component in the proposed G loss increases while the proposed G loss slowly decreases. Hence, despite the low decrease in the overall G loss, $L_{content}$ loss has notably decreased.

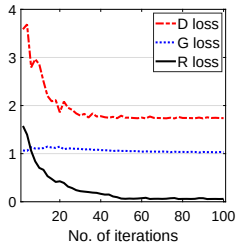


Figure 3: Offline training loss of D, G and R.

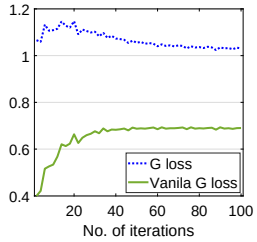


Figure 4: Offline training loss of G and vanilla G.

4.0.2 Online training loss: We observed R loss during the online training process (see Figure 5). Updates are most effective within the first 10-15 iterations and after around 20 iterations, the recommender accuracy reaches its peak. Additional iterations tend to overfit the model and degrade performance. Therefore, the proposed solution is feasible in real-world applications since online training requires only a few iterations and retraining is costly.

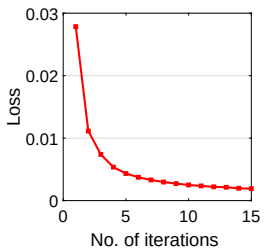


Figure 5: Online training loss of the recommender task.

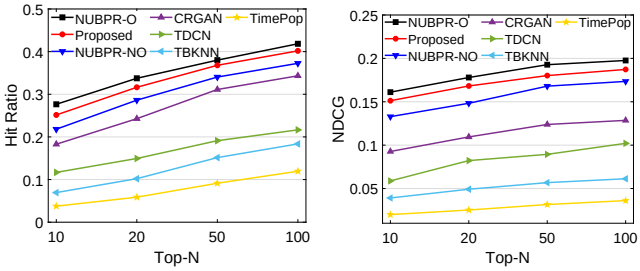


Figure 6: Hit Ratio and NDCG for different Top-N values.

4.0.3 Prediction accuracy: We compared recommender accuracy (HR and NDCG) against multiple Top-N values (see Figure 6). The neural network based solutions (Proposed, NUBPR-O, NUBPR-NO and CRGAN) show a higher accuracy since they capture complex relationships in user preferences. Among the non-neural network based solutions, TimePop has the lowest accuracy since it is based on a simple statistic - the popularity of videos. TDCN outperforms TBKNN and TimePop since it effectively utilizes both source and target network interactions to model user preferences.

As expected, NUBPR-O which does not have a data generation process and is only trained and tested with real mapping data from overlapped users has the best accuracy. We used NUBPR-O as the benchmark, since the goal is to synthesize data similar to the data used to train NUBPR-O. The proposed model shows the closest accuracy to NUBPR-O and consistently outperforms NUBPR-NO, which only used target network interactions. Therefore, the proposed model is able to generate user preferences that increases the recommender accuracy, even when the user overlaps are unknown. Furthermore, the improvements gained over CRGAN, which uses vanilla GAN loss functions, show the effectiveness of the proposed D and G loss functions.

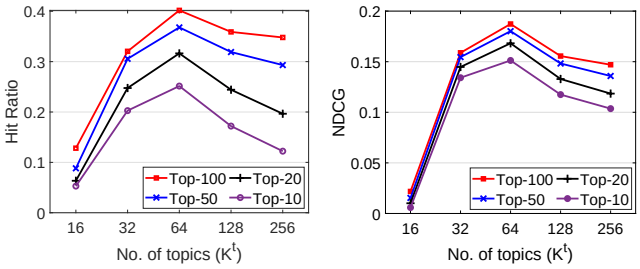


Figure 7: Hit Ratio and NDCG for different number of topics.

4.0.4 Prediction accuracy for different number of topics (K^t): We compared recomender accuracy against the dimensionality of the encoded topical space (K^t) (see Section 2.1.2 and Figure 7). A grid search algorithm was used to select an effective K^t value, and the highest accuracy is achieved when K^t is around 64 topics for all Top-N values. Smaller number of topics fail to effectively encode diverse user preferences, while a higher number of topics create highly sparse inputs and reduce recommender performance. Further experiments using the standard perplexity measure [6] showed that 64 topics lead to a smaller perplexity with faster convergence.

4.0.5 Diversity and novelty: A higher recommendation accuracy alone is insufficient to capture overall user satisfaction. For example, continuously recommending similar items (in terms of topics, genre, etc.) could lead to a decline in recommendation accuracy as users lose interest over time. Therefore, we compared the diversity [4] and novelty [31] of recommended items. The proposed model was able to recommend videos with better novelty and diversity, and was the closest to NUBPR-O model (only a 2.4% and 3.8% novelty and diversity drop). Compared to the single-network based solutions, cross-network solutions showed better results as they utilize rich user profiles with diverse user preferences. Against the closest CRGAN approach, the proposed model showed considerable improvements in novelty (by 8.9%) and diversity (by 12.3%).

4.0.6 Alternative GAN architectures: Designing and training GAN models can be challenging due to the unbalance between D and G, diminishing gradients and non-convergence [2, 27]. Various GAN architectures and training techniques were recently introduced to handle such issues. Therefore, despite the effectiveness of CnGAN, we designed two alternative solutions based on two widely popular GAN architectures, Deep convolutional GAN (DCGAN) [22] and Wasserstein GAN (WGAN) [3] to replace the G and D learning processes. We found that DCGAN becomes highly unstable in this environment, where D error is constantly increased. This can be because, DCGAN is based on Convolution Neural Networks, known to capture the local features within the input data. However, unlike typical image data, user preferences in the vectors may not provide any interesting local features. Hence, DCGAN was less effective. Further experiments using WGAN also did not improve the performances.

5 CONCLUSION AND FURTHER WORK

Typical cross-network recommender solutions are applied to users that are fully overlapped across multiple networks. Thus to the best of our knowledge, we propose the first Cross-network Generative Adversarial Network based model (CnGAN), which generates user preferences for non-overlapped users. The proposed model first uses a generator task to learn the mapping from target to source network user preferences and synthesize source network preferences for non-overlapped users. Second, the model uses a recommender task based on a Siamese network to incorporate synthesized source network preferences and conduct recommendations. The proposed model consistently outperformed multiple baselines in terms of accuracy, diversity and novelty of recommendations. As future work, we plan to investigate the recommender quality improvements when using both generated and real data for overlapped users. Furthermore, the model can be extended to use social information of the users. Overall, CnGAN alleviates a significant limitation in cross-network recommender solutions, and provides a foundation to make quality cross-network recommendations for all users.

Acknowledgments

This research has been supported by Singapore Ministry of Education Academic Research Fund Tier 2 under MOE's official grant number MOE2018-T2-1-103. We gratefully acknowledge the support of NVIDIA Corporation with the donation of a Titan Xp GPU used for this research.

REFERENCES

[1] Fabian Abel, Samur Araújo, Qi Gao, and Geert-Jan Houben. 2011. Analyzing cross-system user modeling on the social web. In *International Conference on Web Engineering*. Springer, 28–43.

[2] Martin Arjovsky and Léon Bottou. 2017. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862* (2017).

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875* (2017).

[4] Iman Avazpour, Teerat Pitakrat, Lars Grunske, and John Grundy. 2014. Dimensions and metrics for evaluating recommendation systems. In *Recommendation systems in software engineering*. Springer, 245–273.

[5] Homanga Bharadhwaj, Homin Park, and Brian Y Lim. 2018. RecGAN: Recurrent generative adversarial networks for recommendation systems. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 372–376.

[6] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[7] Mathieu Blondel, Masakazu Ishihata, Akinori Fujino, and Naonori Ueda. 2016. Polynomial networks and factorization machines: New insights and efficient training algorithms. In *Proceedings of International Conference on Machine Learning*.

[8] Pedro G Campos, Alejandro Bellogín, Fernando Díez, and J Enrique Chavarriaga. 2010. Simple time-biased KNN-based recommendations. In *Proceedings of the Workshop on Context-Aware Movie Recommendation*. ACM, 20–23.

[9] Pedro G Campos, Fernando Dıez, and Iván Cantador. 2012. A performance comparison of time-aware recommendation models.

[10] Zhengyu Deng, Jitao Sang, Changsheng Xu, et al. 2013. Personalized video recommendation based on cross-platform user modeling. In *2013 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 1–6.

[11] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1661–1670.

[12] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[13] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *International Conference on Learning Representations* (2015).

[14] Wang-Cheng Kang, Chen Fang, Zhaowen Wang, and Julian McAuley. 2017. Visually-aware fashion recommendation and design with generative image models. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 207–216.

[15] Diederik P Kingma and Jimmy Lei Ba. 2015. Adam: Amethod for stochastic optimization. In *International Conference on Learning Representation*.

[16] Bang Hui Lim, Dongyuan Lu, Tao Chen, and Min-Yen Kan. 2015. # mytweet via instagram: Exploring user behaviour across multiple social networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on*. IEEE, 113–120.

[17] Bhaskar Mehta, Claudia Niederee, Avare Stewart, Marco Degemmis, Pasquale Lops, and Giovanni Semeraro. 2005. Ontologically-enriched unified user modeling for cross-system personalization. In *International Conference on User Modeling*. Springer, 119–123.

[18] Miles Osborne, Saša Petrovic, Richard McCreadie, Craig Macdonald, and Iadh Ounis. 2012. Bieber no more: First story detection using Twitter and Wikipedia. In *SIGIR 2012 Workshop on Time-aware Information Access*.

[19] Yuanxin Ouyang, Wenqi Liu, Wenge Rong, and Zhang Xiong. 2014. Autoencoder-based collaborative filtering. In *International Conference on Neural Information Processing*. Springer, 284–291.

[20] Dilruk Perera and Roger Zimmermann. 2017. Exploring the use of time-dependent cross-network information for personalized recommendations. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 1780–1788.

[21] Dilruk Perera and Roger Zimmermann. 2018. LSTM Networks for Online Cross-Network Recommendations.. In *IJCAI*. 3825–3833.

[22] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations (ICLR)*.

[23] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[24] Suman Deb Roy, Tao Mei, Wenjun Zeng, and Shipeng Li. 2012. Socialtransfer: Cross-domain transfer learning from social streams for media applications. In *Proceedings of the 20th ACM International Conference on Multimedia*. ACM, 649–658.

[25] Nachiketa Sahoo, Param Vir Singh, and Tridas Mukhopadhyay. 2012. A hidden Markov model for collaborative filtering. *MIS Quarterly* (2012), 1329–1356.

[26] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine learning*. ACM, 791–798.

[27] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*. 2234–2242.

[28] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 515–524.

[29] Ming Yan, Jitao Sang, and Changsheng Xu. 2014. Mining cross-network association for youtube video promotion. In *Proceedings of the 22nd ACM International conference on Multimedia*. ACM, 557–566.

[30] Jaeyoon Yoo, Heonseok Ha, Jihun Yi, Jongha Ryu, Chanju Kim, Jung-Woo Ha, Young-Han Kim, and Sungroh Yoon. 2017. Energy-based sequence gans for recommendation and their connection to imitation learning. *Proceedings of ACM Conference*.

[31] Liang Zhang. 2013. The definition of novelty in recommendation system. *Journal of Engineering Science & Technology Review* 6, 3.

[32] Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. 2011. Comparing twitter and traditional media using topic models. In *European Conference on Information Retrieval*. Springer, 338–349.

Learning Clusters through Information Diffusion

Liudmila Prokhorenkova
Moscow Institute of Physics and Technology
Yandex
Moscow, Russia
ostroumova-la@yandex.ru

Alexey Tikhonov
Yandex
Berlin, Germany
altsoph@gmail.com

Nelly Litvak
University of Twente
Eindhoven University of Technology
Enschede, The Netherlands
n.litvak@utwente.nl

ABSTRACT

When information or infectious diseases spread over a network, in many practical cases, one can observe when nodes adopt information or become infected, but the underlying network is hidden. In this paper, we analyze the problem of finding communities of highly interconnected nodes, given only the infection times of nodes. We propose, analyze, and empirically compare several algorithms for this task. The most stable performance, that improves the current state-of-the-art, is obtained by our proposed heuristic approaches, that are agnostic to a particular graph structure and epidemic model.

KEYWORDS

Community detection; information propagation; information cascades; network inference; likelihood optimization

ACM Reference Format:
Liudmila Prokhorenkova, Alexey Tikhonov, and Nelly Litvak. 2019. Learning Clusters through Information Diffusion. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313560

1 INTRODUCTION

Diffusion processes in networks include spreading of infectious diseases [14], spread of computer viruses [36], promotion of products via viral marketing [16], propagation of information [31], etc. In many practical scenarios one can observe when nodes become infected but the underlying network is hidden. For instance, during an epidemic, a person becomes ill but cannot tell who infected her; in viral marketing, we observe when customers buy products but not who influenced their decisions; in Twitter, for each retweet only information about the root node of the cascade is available. The problem of hidden network inference received a lot of attention recently [7, 27, 29, 30]. While these papers aim to recover the actual network connections, in many applications only some global properties of the underlying network are important. For example, in viral marketing, one may wish to find the most influential users, while for recommendation systems one may look for groups of users with similar preferences.

In this paper, we analyze the problem, which recently attracted interest in the literature [5, 26], of inferring *community structure* of a given network based solely on *cascades* (e.g., information or epidemic) propagating through this network. Communities are groups of highly interconnected nodes with relatively few edges joining nodes of different groups [9], such as groups by interests in social networks or pages on related topics on the Web. Discovering communities is one of most prominent tasks in network analysis. Compared to the traditional community detection, our work is quite different, because we do not have the network available to us, we have only cascade data observed on this network. For each cascade, we observe only infected nodes and their infection timestamps.

We propose and analyze several algorithms to solve this problem and compare them to the state-of-the-art. The contributions of this paper are the following. First, we give a systematic treatment to the new problem of inferring community structure based on cascade data, as stated formally in Section 3. Second, we propose two types of approaches for this task: based on likelihood maximization under specific model assumptions, and based on clustering of a surrogate network (Section 4). Third, we conduct extensive experiments on a variety of real-world networks (see Section 5). We conclude that the most stable performance is obtained by our proposed heuristics that are agnostic to a particular graph structure and epidemic model. These heuristics work equally well on different networks, for epidemics of different types.

2 RELATED WORK

2.1 Network Inference from Cascades

A series of recent papers addressed the following task: by observing infection (activation) times of nodes in several cascades, infer the edges of the underlying network. NetInf algorithm developed in [11] is based on the maximization of the likelihood of the observed cascades, based on a specific epidemic model. To make optimization feasible, for each cascade NetInf considers only the most likely propagation tree. This was later improved by MultiTree [30] that includes all directed trees in the optimization and has better performance if the number of cascades is small. NetRate algorithm [28] infers not only edges, but also infection rates. NetRate builds on an epidemic model that is more tractable for theoretical analysis (we describe this model in Section 3.2). For this model the likelihood optimization problem turns out to be convex. ConNIe algorithm [21] also uses convex optimization, with the goal to infer transmission probabilities for all edges. In [13, 29], it is additionally assumed that the underlying network is not static and the proposed algorithm InfoPath provides on-line estimates of the structure and temporal dynamics of the hidden network. KernelCascade [8] also extends NetRate, here the authors avoid assumptions about a particular parametric form of the influence function. The DANI algorithm [27] is interesting because it explicitly accounts for the

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313560

community structure to enhance the inference of networks' edges. There are some other network inference algorithms not covered here, see, e.g., [7, 12, 22, 32, 34, 35, 37, 38, 40].

2.2 Community Inference from Cascades

To the best of our knowledge, the paper [4] extended in [5] for the first time addressed the problem of community detection given cascades. In [4, 5], the cascade model includes influence of individual nodes, and a membership level of a node in each community is inferred using the maximum likelihood approach. The authors propose two algorithms: C-IC takes into account only participation of a node in a cascade; C-Rate includes the time stamps, but limits the node's influence by its own community. Recently, [26] proposed an alternative maximum likelihood approach, which exploits the Markov property of the cascades. As an input, similarity scores of node pairs are computed, based on their joint participation in cascades. The R-CoDi algorithm in [26] starts with a random partition, while D-CoDi starts with a partition obtained by DANI [27]. We use all four mentioned algorithms as our baselines.

3 PROBLEM SETUP

3.1 General Setup

Cascades. We observe a set of cascades $\mathcal{C} = \{C_1, \dots, C_r\}$ that propagate on a latent undirected network $G = (V, E)$ with $|V| = n$ nodes and $|E| = m$ edges. Each cascade $C \in \mathcal{C}$ is a record of observed node activation times, i.e., $C = \{(v_i, t^C_{v_i})\}_{i=1}^{n_C}$, where v_i is a node, $t^C_{v_i}$ is its activation time in C, $|C| = n_C$ is a size of a cascade. Note that we do not observe who infected whom.

Communities. We assume that G is partitioned into communities: $\mathcal{A} = \{A_1, \dots, A_k\}$, $\cup_{i=1}^k A_i = V$, $A_i \cap A_j = \emptyset$ for $i \neq j$. We expect to observe a high intra-community density of edges compared to inter-community density. In our experiments, the ground truth partitions $\mathcal{A}$ are available for all datasets (see Section 5.1.1). By observing only a set of cascades $\mathcal{C}$ we want to find a partition $\mathcal{A}'$ similar to $\mathcal{A}$.

3.2 Cascade Models

3.2.1 SIR model. The main model for our experiments is a well-known SIR (Susceptible-Infected-Recovered) model [15]. Each node in the network can be in one of the three states: susceptible, infected, or recovered. An infected node infects its susceptible neighbors with rate α, the infection is spread simultaneously and independently along all edges. An infected node recovers with rate β and then stops spreading infection in the network. For each cascade C we sample its own infection rate α_C from the Lomax (shifted Pareto) distribution in order to model a variety of cascades: there can be minor or widely circulated news, small scale epidemics or pandemics, etc. The source node of a cascade is chosen uniformly at random.

3.2.2 SI model with bounded duration (SI-BD). In some cases, SIR model might not be tractable for theoretical analysis, so we assume a simpler diffusion model introduced in [28]. In this model, again, an activated node infects its neighbors after an exponentially distributed time with intensity α, but there is no recovery rate. Instead, all nodes recover simultaneously at some threshold time T_{max} and the epidemic stops. For simplicity we further assume T_{max} to be fixed, but our methods allow varying T_{max} for different epidemics.

3.2.3 Community-based SI-BD model (C-SI-BD). Another model which allows for a simpler theoretical analysis is based on the setting from [33]. It is assumed that the spreading does not occur over the edges of a graph G but depends solely on the community structure $\mathcal{A}$. As before, the first node of a cascade is chosen uniformly at random. Each activated node can infect all other susceptible nodes independently after an exponentially distributed time. If a susceptible node belongs to the same community, then the infection rate is α_{in}, otherwise it is α_{out}, $\alpha_{out} < \alpha_{in}$. Epidemic stops at time T_{max}.

4 ALGORITHMS

4.1 Background on Likelihood Maximization

Recall that t_i^C denotes the activation time for node i in cascade C; we often omit the index C when the context is clear. Without loss of generality, for the first node of a cascade, we set $t_i = 0$. Finally, if a node i is not infected during an epidemic, then we set $t_i = T_{max}$.

Denote $\Delta^C_{i,j} = \Delta_{i,j} = |t_i - t_j|$. The log-likelihood $\log L(C)$ of the cascade C for SI-BD with varying infection rates (i infects a susceptible neighbor j after an exponentially distributed time with rate $\alpha_{i,j}$) is given in [28]. For our purposes, it is convenient to write this expression as:

$$\log L(C) = -\sum_{i,j:i<j} \alpha_{i,j}\Delta_{i,j} + \sum_{i:t_i\in(0,T_{max})} \log \sum_{j:t_j<t_i} \alpha_{j,i}\,. \quad (1)$$

The log-likelihood for all cascades is $\log L(\mathcal{C}) = \sum_{C\in\mathcal{C}} \log L(C)$. We will next introduce two methods based on maximizing $\log L(C)$.

4.2 ClustOpt

Consider the C-SI-BD model described in Section 3.2.3. Denote by $a(i)$ the community assignment for a node i. Then C-SI-BD is equivalent to the model used in [28] with $\alpha_{i,j} = \alpha_{in}$ if $a(i) = a(j)$ and $\alpha_{i,j} = \alpha_{out}$ otherwise. The log-likelihood in (1) becomes:

$$\log L(C, \mathcal{A}) = -(\alpha_{in} - \alpha_{out}) \sum_{\substack{i,j:i<j,\\ a(i)=a(j)}} \Delta_{i,j} - \alpha_{out} \sum_{i,j:i<j} \Delta_{i,j}$$
$$+ \sum_{i:t_i\in(0,T_{max})} \log\Big((\alpha_{in} - \alpha_{out})|\{j : t_j < t_i, a(j) = a(i)\}|$$
$$+\alpha_{out}|\{j : t_j < t_i\}|\Big)\,. \quad (2)$$

.

We propose the following algorithm to maximize (2).

ALGORITHM 1: ClustOpt

(1) Find initial partition $\mathcal{A}_{init}$;
(2) Find $\hat{\alpha}_{in}, \hat{\alpha}_{out} = \arg\max_{\alpha_{in},\alpha_{out}} \log L(C, \mathcal{A}_{init})$;
(3) For fixed $\hat{\alpha}_{in}, \hat{\alpha}_{out}$ find $\hat{\mathcal{A}} = \arg\max_{\mathcal{A}} \log L(C, \mathcal{A})$.

Let us now discuss how the steps (1)–(3) are implemented.

Step (1): We have noticed that a stable and fast approach is to start from some initial reasonable partition and optimize over α_{in}, α_{out} and $\mathcal{A}$ only once, thus avoiding iterations of the costly optimization

steps (2) and (3). In the current paper we propose to start from Clique(0) explained in Section 4.4.

Step (2): Without loss of generality, we set $\alpha_{in} = (\delta + 1)\alpha_{out}$. Substituting this into (2), we find optimal α_{out} in terms of δ and $\mathcal{A}$:

$$\hat{\alpha}_{out} = \frac{\sum_{C \in \mathcal{C}} (|C| - 1)}{\sum_{C \in \mathcal{C}} \left(\delta \sum_{i,j:a(i)=a(j)} \Delta^C_{i,j} + \sum_{i,j} \Delta^C_{i,j} \right)}. \quad (3)$$

Unfortunately, due to the summation of logarithms in (2), we cannot find another simple analytical relation between the optimal values of δ and α_{out}. Hence, we resort to a numerical solution. Due to (3), it is sufficient to numerically find only the optimal δ, this will give us the optimal values for both α_{in} and α_{out}.

Step (3): We follow [25], where the Louvain algorithm [6] is modified for optimizing a wide range of functions that depend on community structure, besides the original modularity measure. We adapt this algorithm for the likelihood given in (2) by computing the gain in $\log L(\mathcal{C}, \mathcal{A})$ obtained by moving a node v from one community to another. In view of computational complexity, we consider only moving single nodes from one community to another and do not attempt to move groups of nodes or merge communities.

4.3 GraphOpt

In this section, we drop the assumption of fixed infection rates within and between communities. Instead, we assume the SI-BD cascade model defined in Section 3.2.2. Our task is much more complex in this setting because now we have a hidden graph G.

We propose an expectation-maximization-based method, where G is a latent variable. We assume the following generative probabilistic process. For a given partition $\mathcal{A}$ of n nodes we construct a graph G according to some model (distribution) $\mathrm{P}(G|\mathcal{A})$. Then, based on G, we generate a set of cascades $\mathcal{C}$ according to another model $\mathrm{P}(\mathcal{C}|G)$. We observe $\mathcal{C}$ and our aim is to recover the partition $\bar{\mathcal{A}}$ which maximizes the likelihood, i.e., $\bar{\mathcal{A}} = \arg\max_{\mathcal{A}} \mathrm{P}(\mathcal{C}|\mathcal{A})$. Let us first explain how the standard expectation-maximization (EM) approach applies to this problem:

(1) Choose some initial partition $\hat{\mathcal{A}}$;
(2) Find the distribution $\mathrm{P}(G|\mathcal{C}, \hat{\mathcal{A}})$;
(3) Update $\hat{\mathcal{A}}$: $\hat{\mathcal{A}} = \arg\max_{\mathcal{A}} \mathbb{E}_G \log \mathrm{P}(\mathcal{A}\, G|\mathcal{C})$;
(4) Iterate (2)-(3) until convergence.

This algorithm cannot be applied directly because, as we explain below, steps (2) and (3) are computationally intractable. Our algorithm GraphOpt is obtained by simplifying these two steps.

First, let us rewrite the distribution in (2):

$$\mathrm{P}(G|\mathcal{C}, \hat{\mathcal{A}}) = \frac{\mathrm{P}(\mathcal{C}\, G\, \mathcal{A})}{\mathrm{P}(\mathcal{C}\, \hat{\mathcal{A}})} = \frac{\mathrm{P}(\mathcal{C}|G, \hat{\mathcal{A}})\mathrm{P}(G\, \hat{\mathcal{A}})}{\mathrm{P}(\mathcal{C}|\hat{\mathcal{A}})\mathrm{P}(\hat{\mathcal{A}})}$$
$$= \frac{\mathrm{P}(\mathcal{C}|G)\mathrm{P}(G|\hat{\mathcal{A}})}{\mathrm{P}(\mathcal{C}|\hat{\mathcal{A}})} \propto \mathrm{P}(\mathcal{C}|G)\mathrm{P}(G|\hat{\mathcal{A}}), \quad (4)$$

where we used that $\mathrm{P}(\mathcal{C}|\hat{\mathcal{A}})$ does not depend on G and $\mathrm{P}(\mathcal{C}|G, \hat{\mathcal{A}}) = \mathrm{P}(\mathcal{C}|G)$ since, given the graph, the division into communities does not affect the cascades.

Now, we rewrite the probability in (3): $\mathrm{P}(\mathcal{A}\, G|\mathcal{C}) = \frac{\mathrm{P}(\mathcal{C}\, G\, \mathcal{A})}{\mathrm{P}(\mathcal{C})} = \frac{\mathrm{P}(\mathcal{C}|G)\mathrm{P}(G|\mathcal{A})\mathrm{P}(\mathcal{A})}{\mathrm{P}(\mathcal{C})}$. Since we do not have any prior assumptions for $\mathcal{A}$, we assume that $\mathrm{P}(\mathcal{A})$ is constant. Also, $\mathrm{P}(\mathcal{C})$ and $\mathrm{P}(\mathcal{C}|G)$ do not depend on $\mathcal{A}$, so

$$\arg\max_{\mathcal{A}} \mathbb{E}_G \log \mathrm{P}(\mathcal{A}\, G|\mathcal{C}) = \arg\max_{\mathcal{A}} \mathbb{E}_G \log \mathrm{P}(G|\mathcal{A}). \quad (5)$$

From (4) and (5) we see the computational bottlenecks of the EM algorithm. Indeed, even if for each G we can estimate the probability in (4), it is still impossible to even compute $\mathbb{E}_G \log \mathrm{P}(G|\mathcal{A})$, because the expectation is over all possible realizations of G. Clearly, we cannot hope to maximize this expression over $\mathcal{A}$. Therefore, we simplify the procedure: instead of computing the entire probability distribution over all graphs G in (4), we find only the most likely graph $\hat{G}$, with the idea that $\hat{G}$ gives the largest contribution to $\mathbb{E}_G \log \mathrm{P}(G|\mathcal{A})$. Then the optimization problem in (5) is solved only for $\hat{G}$. As a result, we obtain the following algorithm.

ALGORITHM 2: GraphOpt

(1) Choose some initial partition $\hat{\mathcal{A}}$ and graph $\hat{G}$;
(2) Update $\hat{G}$: $\hat{G} = \arg\max_G \left(\log \mathrm{P}(\mathcal{C}|G) + \log \mathrm{P}(G|\hat{\mathcal{A}}) \right)$;
(3) Update $\hat{\mathcal{A}}$: $\hat{\mathcal{A}} = \arg\max_{\mathcal{A}} \log \mathrm{P}(\hat{G}|\mathcal{A})$;
(4) Iterate (2)-(3) until convergence.

Let us describe how each step of Algorithm 2 is implemented.

Step (1): We start from a trivial partition $\hat{\mathcal{A}}$ in which each node belongs to its own cluster. We also create an initial graph[1] $\hat{G} = (V, E_0)$ as follows: for each cascade C consider the path of $|C| - 1$ edges, where each edge connects two nodes that are subsequently activated in this cascade; E_0 is a union of such paths over all $C \in \mathcal{C}$.

Step (2): $\mathrm{P}(G|\hat{\mathcal{A}})$ is defined by the ILFR model proposed in [25], where the formula for $\log \mathrm{P}(G|\hat{\mathcal{A}})$ is also provided. This model was shown to give the best fit to a variety of real-world networks in terms of likelihood. For $\mathrm{P}(\mathcal{C}|G)$, we assume the SI-BD cascade model, so we use (1) with $\alpha_{i,j} = \alpha$ if i and j are connected in G and $\alpha_{i,j} = 0$ otherwise. Then $\log \mathrm{P}(\mathcal{C}|G) = \sum_{C \in \mathcal{C}} \log \mathrm{P}(C|G)$ with

$$\log \mathrm{P}(C|G) = -\alpha \sum_{(i,j) \in E(G)} \Delta_{i,j} + (|C| - 1) \log \alpha$$
$$+ \sum_{i:t_i \in (0, T_{max})} \log |\{j : t_j < t_i, (i,j) \in E(G)\}|. \quad (6)$$

A nice feature of (6) is that for given $\mathcal{C}$ and G it is easy to compute α that maximizes the overall likelihood:

$$\alpha_{opt} = \frac{\sum_{C \in \mathcal{C}} (|C| - 1)}{\sum_{C \in \mathcal{C}} \sum_{(i,j) \in E(G)} \Delta^C_{i,j}}. \quad (7)$$

Finally, we need to find $\hat{G} = \arg\max_G \left(\log \mathrm{P}(G|\hat{\mathcal{A}}) + \log \mathrm{P}(\mathcal{C}|G) \right)$. We use a greedy approach to approximately find $\hat{G}$: we iteratively add and remove edges to increase $\log \mathrm{P}(G|\hat{\mathcal{A}}) + \log \mathrm{P}(\mathcal{C}|G)$ (the detailed description is omitted due to space constrains).

Step (3): This is a standard likelihood optimization task used in community detection. We use the Louvain-based algorithm proposed in [25].

[1] This initial graph is needed for step (2), since our update algorithm requires initialization with non-zero likelihood.

4.4 Clustering of Surrogate Graphs

In this section, we present simple yet effective methods that construct a surrogate graph $\hat{G}$ and then cluster this graph (using the Louvain algorithm [6]). It is crucial that $\hat{G}$ does *not* need to be similar to G, it just needs to capture the community structure on an aggregated level. Our experiments show that clustering of $\hat{G}$ often performs better than first inferring G and then clustering it.

4.4.1 Path algorithm. Assume again that cascades $\mathcal{C}$ are generated by the SI-BD model. Now, for $C \in \mathcal{C}$ let $G' = G'(C)$ be a graph with exactly $|C| - 1$ edges that maximizes the probability $\mathrm{P}(C|G')$ in (6). It is easy to check that G' is merely a path connecting subsequently activated nodes. Note that we have already used such paths to initialize Algorithm 2. This leads to the following algorithm.

ALGORITHM 3: Path

(1) Construct weighted graph $\hat{G}$, where the weight of each edge e in $\hat{G}$ is the number of $G'(C)$ including e;[2]
(2) Find clusters in $\hat{G}$ using the Louvain algorithm.

4.4.2 Clique. Another approach is to include *all* possible edges that could participate in the cascade weighing them by the proxy of their likelihood. Then each cascade C results in a weighted clique of size $|C|$. The weights can be chosen, for example, as follows. For a cascade C and two nodes i, j let us consider the probability $P^C(i,j) = \mathrm{P}(j \text{ was infected from } i|C)$. If $t_i > t_j$, then, obviously, $P^C(i,j) = 0$. If $t_i < t_j$, then, as in [11], we assume that $P^C(i,j)$ decreases exponentially with $\Delta_{i,j}$; namely, $P^C(i,j) = c_j\, e^{-a\Delta_{i,j}}$, where c_j is a constant depending on j. Since j was infected from exactly one previous node, we must have $\sum_{i:t_i<t_j} P^C(i,j) = 1$. Therefore, $P^C(i,j) = \frac{e^{-a\Delta_{i,j}}}{\sum_{l:t_l<t_j} e^{-a\Delta_{l,j}}}$.

Clique constructs $\hat{G}$ as the weighted graph with weight of $\{i,j\}$ given by $\sum_{C\in\mathcal{C}}(P^C(i,j) + P^C(j,i))$, which, under our assumptions, is the expected number of times infections passed between i and j. Note that we directly model $P^C(i,j)$ rather than making assumptions about infection times. Parameter a essentially balances between paths and cliques: if a is large, then we mostly take into account subsequent nodes with small $\Delta_{i,j}$; for small a all pairs of nodes participated in C are important. To make Clique insensitive to the speed of epidemics, we take $a = \frac{1}{\Delta}$, where Δ is the average time between infected times (we average over all pairs of infected nodes belonging to the same cascade). We also consider Clique(0) with $a = 0$, which is a natural choice for the SI-BD model because it mimics the memory-less property of exponential infection times.[3]

ALGORITHM 4: Clique

(1) Construct $\hat{G}$ as the weighted graph with weight of $\{i,j\}$ given by $\sum_{C\in\mathcal{C}}(P^C(i,j) + P^C(j,i))$.
(2) Find clusters in $\hat{G}$ using the Louvain algorithm.

[2]Weighting edges is important, since for large $|C|$ unweighted union of edges can be a complete graph, with no meaningful partition.
[3]We noticed that Clique is not too sensitive to varying a in a reasonable interval, while for too large a Clique becomes very similar to Path.

4.5 Baselines

MultiTree uses the algorithm from [30] to find an inferred graph $\hat{G}$, which is then clustered by the Louvain algorithm [6].

Oracle algorithm can be considered as superior for all possible network inference algorithms: we construct a graph $\hat{G}$ consisting of all edges participated in cascades (assuming these edges are known, hence, the name Oracle). The obtained graph $\hat{G}$ is clustered by the Louvain algorithm.

Finally, we used algorithms proposed for solving the same problem as in our work: C-IC and C-Rate [5], as well as R-CoDi and D-CoDi [26]. We use the publicly available implementations provided by the authors. For C-IC and C-Rate we use the real number of communities as an input parameter.[4]

5 EXPERIMENTS

5.1 Experimental Setup

5.1.1 Networks. We used real-world datasets of different sizes and nature, for which the ground truth partition of nodes into k non-overlapping communities is available: Zachary's karate club [39] ($n = 34$, $m = 78$, $k = 2$), dolphin social network [20] ($n = 62$, $m = 159$, $k = 2$), American college football [24] ($n = 115$, $m = 613$, $k = 11$), books about politics [23] ($n = 105$, $m = 441$, $k = 3$), political blogs [1] ($n = 1224$, $m = 16715$, $k = 2$), email-Eu-core dataset [19] ($n = 986$, $m = 16064$, $k = 42$).

5.1.2 Metric. We evaluate the quality of the obtained communities compared to the ground truth using the Normalized Mutual Information (NMI) similarity measure [2, 9]. Let $V_0 \subset V$ denote the set of all nodes that participated in cascades. To compute NMI, we assign all nodes in $V \backslash V_0$ to one cluster labeled "unknown".

5.1.3 Cascades. We took the datasets with ground truth communities described in Section 5.1.1 and, in order to cover various possible diffusion processes, we generated the cascades according to all models discussed in Section 3.2. Importantly, there are both edge-based and community-based models.

W.l.o.g., we set $\beta = 1$ for the SIR model and $T_{max} = 1$ for SI-BD and C-SI-BD. We noticed that to get an informative set of cascades, the parameters α, α_{in}, α_{out}, and the parameter of the Lomax distribution used by SIR have to be different for different datasets. For SIR and SI-BD we choose parameters so that the average size of a cascade is 2; for C-SI-BD we take $\alpha_{in} = 10\alpha_{out}$ and choose α_{out} such that the number of cascades consisting of one node is about 20%. We checked that the obtained distribution of cascade sizes is similar to observed for real data [3, 10, 12, 17, 18]. Single-node cascades were removed.

5.2 Results

We have run the algorithms on all datasets and varied the number of cascades $|C|$. Figures 1 and 2 show the results obtained for SIR and C-SI-BD models, respectively.[5] All results are averaged over 5 samples of generated cascades $\mathcal{C}$.

[4]Although there are modifications of these algorithms automatically choosing the number of communities, these modifications often failed on our data.
[5]Due to space constraints, we omit plots for the SI-BD model since they lead to similar conclusions.

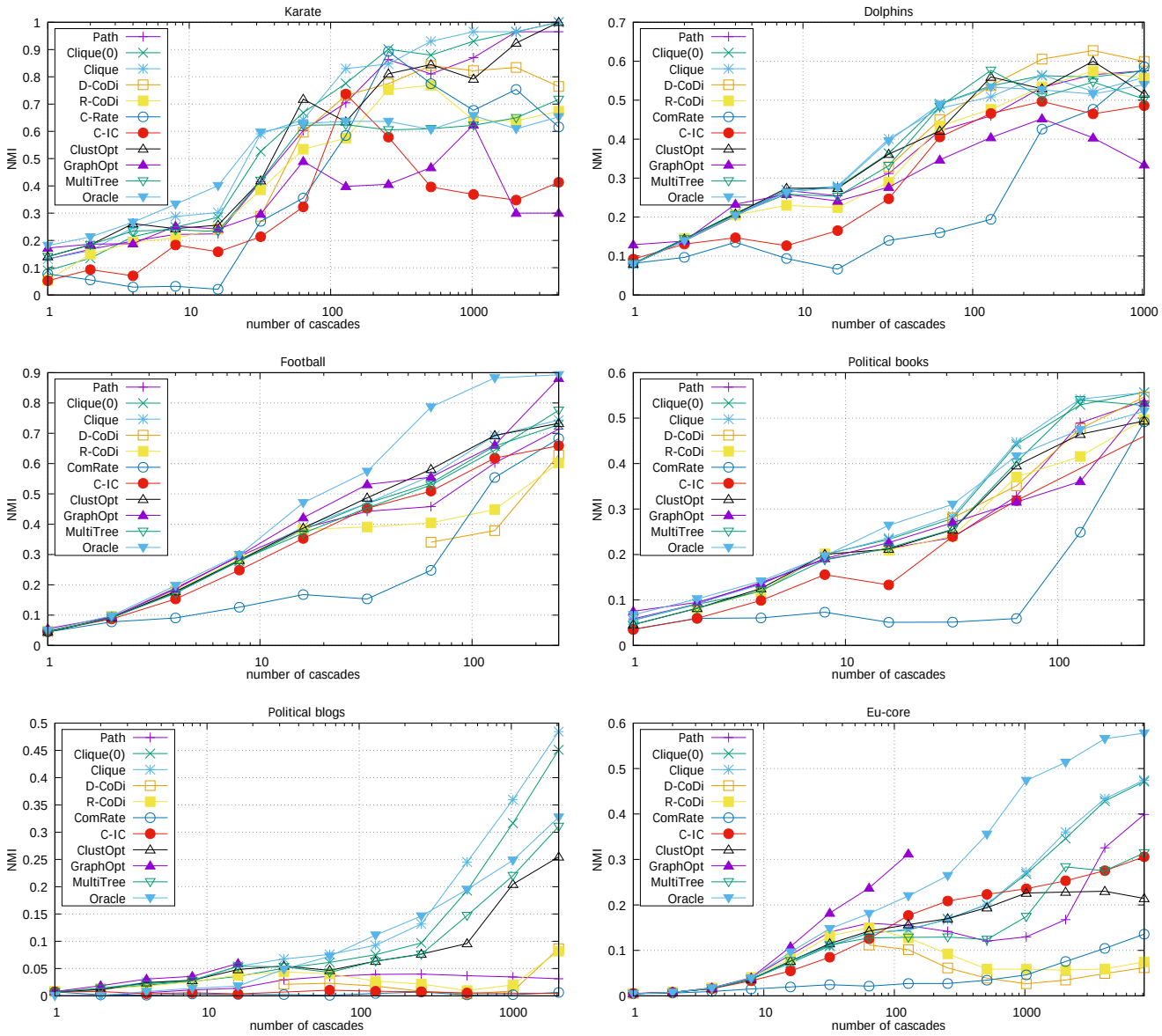


Figure 1: Comparison of algorithms on real-world datasets, SIR cascades

We noticed that the results are quite different for different datasets and cascades models. Nevertheless, the proposed Clique algorithm has the most stable performance, implying that the corresponding surrogate graph is able to capture the information about the community structure. Importantly, both Clique(0) and Clique work equally well for all cascade models since they are not based on a particular one, as discussed in Section 4.4.

Note that Path is usually worse than Clique as it uses less information.

The baselines C-IC and C-Rate are very unstable and in many cases they show the worst results. This is expected, since C-IC and C-Rate are based on a specific community-based model. Indeed, on C-SI-BD C-IC shows a reasonable performance in some cases, especially on Political blogs. Note that C-IC is almost always better than C-Rate (except for some cascade sizes on Karate with SIR model). D-CoDi and R-CoDi are essentially based on clustering of a surrogate graph, so their performance is stable for all epidemic models. However, in all cases D-CoDi failed on cascades of small sizes (we included only successful points). D-CoDi can be both better and worse than R-CoDi. Having reasonable performance in many cases, these two baselines may have unexpectedly bad points: see, e.g., the large number of cascades on Political blogs and Eu-Core with all epidemic models. MultiTree turns out to have

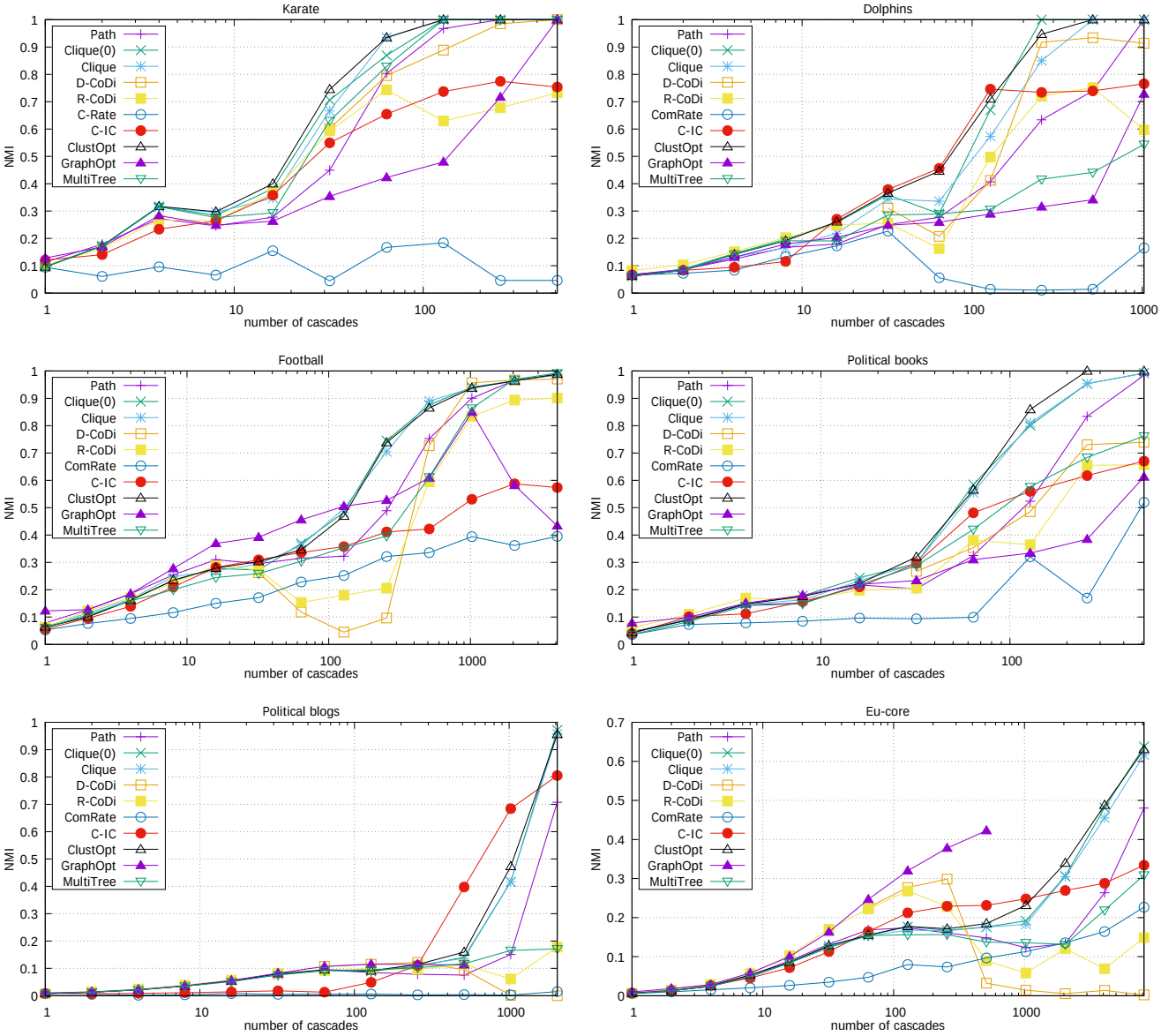


Figure 2: Comparison of algorithms on real-world datasets, C-SI-BD cascades

the most stable performance among the baselines but it is usually beaten by Clique(0) and Clique.

GraphOpt shows an excellent performance, e.g., on Eu-core for all types of epidemics, on Football for SIR cascades and for small number of C-SI-BD cascades. However, in general its quality is unstable, and it is also much slower than surrogate-based methods.

Oracle is considered only as an upper bound for all network inference algorithms.[6] If the number of cascades is large enough, then Oracle essentially clusters the original graph. Interestingly, in many cases Oracle is beaten by the surrogate-graph-based algorithms. This basically means that errors made by Louvain on original graph can be reduced by clever weighting of edges provided by the surrogate-graph-based algorithms.

To conclude, we propose using the universal Clique method for the problem of community inference based on cascade data. This algorithm is simple, efficient, and works well for all considered cascade models.

[6]We do not plot Oracle for C-SI-BD, since in this model graph is not used by the propagation processes and. In particular, for large $|C|$ we may get a complete graph.

ACKNOWLEDGMENTS

This study was funded by RFBR according to the research project 18-31-00207.

REFERENCES

[1] Lada A Adamic and Natalie Glance. 2005. The political blogosphere and the 2004 US election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*. ACM, 36–43.

[2] James P Bagrow. 2008. Evaluating local community methods in networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008, 05 (2008), P05001.

[3] Eytan Bakshy, Jake M Hofman, Winter A Mason, and Duncan J Watts. 2011. Identifying influencers on twitter. In *Fourth ACM International Conference on Web Seach and Data Mining (WSDM)*.

[4] Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. 2013. Influence-based network-oblivious community detection. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE, 955–960.

[5] Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. 2017. Efficient methods for influence-based network-oblivious community detection. *ACM Transactions on Intelligent Systems and Technology (TIST)* 8, 2 (2017), 32.

[6] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[7] Hadi Daneshmand, Manuel Gomez-Rodriguez, Le Song, and Bernhard Schoelkopf. 2014. Estimating Diffusion Network Structures: Recovery Conditions, Sample Complexity & Soft-thresholding Algorithm.. In *ICML*. 793–801.

[8] Nan Du, Le Song, Ming Yuan, and Alex J Smola. 2012. Learning networks of heterogeneous influence. In *Advances in Neural Information Processing Systems*. 2780–2788.

[9] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* 486, 3 (2010), 75–174.

[10] Wojciech Galuba, Karl Aberer, Dipanjan Chakraborty, Zoran Despotovic, and Wolfgang Kellerer. 2010. Outtweeting the twitterers-predicting information cascades in microblogs. *WOSN* 10 (2010), 3–11.

[11] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. 2010. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1019–1028.

[12] Manuel Gomez-Rodriguez, Jure Leskovec, and Bernhard Schölkopf. 2013. Modeling Information Propagation with Survival Theory.. In *ICML*. 666–674.

[13] Manuel Gomez Rodriguez, Jure Leskovec, and Bernhard Schölkopf. 2013. Structure and dynamics of information pathways in online media. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 23–32.

[14] Lars Hufnagel, Dirk Brockmann, and Theo Geisel. 2004. Forecast and control of epidemics in a globalized world. *Proceedings of the National Academy of Sciences of the United States of America* 101, 42 (2004), 15124–15129.

[15] Matt J Keeling and Ken TD Eames. 2005. Networks and epidemic models. *Journal of the Royal Society Interface* 2, 4 (2005), 295–307.

[16] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 137–146.

[17] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. 2010. What is Twitter, a social network or a news media?. In *Proceedings of the 19th international conference on World wide web*. AcM, 591–600.

[18] Kristina Lerman, Rumi Ghosh, and Tawan Surachawala. 2012. Social contagion: An empirical study of information spread on Digg and Twitter follower graphs. *arXiv preprint arXiv:1202.3162* (2012).

[19] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2007. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1, 1 (2007), 2.

[20] David Lusseau, Karsten Schneider, Oliver J Boisseau, Patti Haase, Elisabeth Slooten, and Steve M Dawson. 2003. The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology* 54, 4 (2003), 396–405.

[21] Seth Myers and Jure Leskovec. 2010. On the convexity of latent social network inference. In *Advances in Neural Information Processing Systems*. 1741–1749.

[22] Praneeth Netrapalli and Sujay Sanghavi. 2012. Learning the graph of epidemic cascades. *ACM SIGMETRICS Performance Evaluation Review* 40, 1 (2012), 211–222.

[23] Mark EJ Newman. 2006. Modularity and community structure in networks. *Proceedings of the national academy of sciences* 103, 23 (2006), 8577–8582.

[24] Mark EJ Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical review E* 69, 2 (2004), 026113.

[25] Liudmila Prokhorenkova and Alexey Tikhonov. 2019. Community detection through likelihood optimization: in search of a sound model. *Proceedings of the 2019 World Wide Web Conference (WWW '19)* (2019).

[26] Maryam Ramezani, Ali Khodadadi, and Hamid R Rabiee. 2018. Community Detection Using Diffusion Information. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 12, 2 (2018), 20.

[27] Maryam Ramezani, Hamid R Rabiee, Maryam Tahani, and Arezoo Rajabi. 2017. DANI: A Fast Diffusion Aware Network Inference Algorithm. *arXiv preprint arXiv:1706.00941* (2017).

[28] Manuel Gomez Rodriguez, David Balduzzi, and Bernhard Schölkopf. 2011. Uncovering the temporal dynamics of diffusion networks. *arXiv preprint arXiv:1105.0697* (2011).

[29] Manuel Gomez Rodriguez, Jure Leskovec, David Balduzzi, and Bernhard Schölkopf. 2014. Uncovering the structure and temporal dynamics of information propagation. *Network Science* 2, 01 (2014), 26–65.

[30] Manuel Gomez Rodriguez and Bernhard Schölkopf. 2012. Submodular inference of diffusion networks from multiple trees. *arXiv preprint arXiv:1205.1671* (2012).

[31] Daniel M Romero, Brendan Meeder, and Jon Kleinberg. 2011. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th international conference on World wide web*. ACM, 695–704.

[32] Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda. 2009. Learning continuous-time information diffusion model for social behavioral data analysis. In *Asian Conference on Machine Learning*. Springer, 322–337.

[33] Jaron Sanders and Alexandre Proutière. 2017. Optimal Clustering Algorithms in Block Markov Chains. *arXiv preprint arXiv:1712.09232* (2017).

[34] Arlei Silva, Hérico Valiati, Sara Guimarães, and Wagner Meira Jr. 2011. From individual behavior to influence networks: A case study on twitter. In *Proc. of the 17th Brazilian Symposium on Multimedia, Hypermedia and Web*.

[35] Tristan Mark Snowsill, Nick Fyson, Tijl De Bie, and Nello Cristianini. 2011. Refining causality: who copied from whom?. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 466–474.

[36] Chenxi Wang, John C Knight, and Matthew C Elder. 2000. On computer viral infection and the effect of immunization. In *Computer Security Applications, 2000. ACSAC'00. 16th Annual Conference*. IEEE, 246–256.

[37] Liaoruo Wang, Stefano Ermon, and John E Hopcroft. 2012. Feature-enhanced probabilistic models for diffusion network inference. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 499–514.

[38] Shuang-Hong Yang and Hongyuan Zha. 2013. Mixture of mutually exciting processes for viral diffusion. In *International Conference on Machine Learning*. 1–9.

[39] Wayne W Zachary. 1977. An information flow model for conflict and fission in small groups. *Journal of anthropological research* 33, 4 (1977), 452–473.

[40] Ke Zhou, Hongyuan Zha, and Le Song. 2013. Learning Social Infectivity in Sparse Low-rank Networks Using Multi-dimensional Hawkes Processes.. In *AISTATS*, Vol. 13. 641–649.

Constructing Test Collections using Multi-armed Bandits and Active Learning

Md Mustafizur Rahman
The University of Texas at Austin
nahid@utexas.edu

Mucahid Kutlu
TOBB Economy and Tech. University
m.kutlu@etu.edu.tr

Matthew Lease
The University of Texas at Austin
ml@utexas.edu

ABSTRACT

While test collections provide the cornerstone of system-based evaluation in information retrieval, human relevance judging has become prohibitively expensive as collections have grown ever larger. Consequently, intelligently deciding which documents to judge has become increasingly important. We propose a two-phase approach to intelligent judging across topics which does not require document rankings from a shared task. In the first phase, we dynamically select the next topic to judge via a multi-armed bandit method. In the second phase, we employ active learning to select which document to judge next for that topic. Experiments on three TREC collections (varying scarcity of relevant documents) achieve $\tau \approx 0.90$ correlation for P@10 ranking and find 90% of the relevant documents at 48% of the original budget. To support reproducibility and follow-on work, we have shared our code online[1].

KEYWORDS

Information Retrieval; Evaluation; Active Learning; Multi-Armed Bandits

ACM Reference Format:
Md Mustafizur Rahman, Mucahid Kutlu, and Matthew Lease. 2019. Constructing Test Collections using Multi-armed Bandits and Active Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313675

1 INTRODUCTION

Cranfield-based evaluation of information retrieval (IR) systems [4, 23] relies on the construction of a *test collection*: a document collection, a set of search topics, and human judgments of document relevance for those search topics. However, large-scale IR evaluation is becoming increasingly challenging and economically infeasible because there are too many documents in the collection to be judgedConsequently, there is a growing need for improved methods to create test collections at minimal cost.

Developing a reliable, low-cost IR test collection requires intelligent budget allocation across search topics and careful selection of documents for human annotation. However, existing approaches [28, 32] either allocate the same budget across all topics or estimate a budget per topic rather than allocating the total budget dynamically across topics based on their judgment needs. For example, in TREC, each topic is traditionally allotted a budget that equals the size of its *top-k* pool, constructed by pooling the highest ranked k documents from the submitted runs for that topic.

However, developing a test collection by running a shared task has several drawbacks. First of all, running a shared task is difficult, slow, and expensive. In some cases, it may be nearly impossible to garner enough participants, such as for less studied languages (e.g., Turkish) or search tasks (e.g., historical search). In addition, Li and Kanoulas [17] recently noted that some document selection approaches based on participant rankings [9, 18] may overly bias selection toward the best-performing runs. Furthermore, Voorhees [28] reports empirical evidence that some dynamic document selection methods [18] can also produce less reusable test collections.

Consequently, if our goal is simply to build a new test collection at minimal cost, it would be preferable to be able to do this without having to run a shared task [26]. However, this poses its own set of challenges. In the absence of document rankings from shared task participants, we must find another means to select which documents should be judged for relevance to search topics. In addition, while the size of the *top-k* pools for different topics are known to vary widely, since we have no document rankings with which to apply pooling, we must find another means of allocating the judging budget across search topics.

We propose a two-phase approach to constructing test collections without needing to organize a shared task. Because it is known that evaluation becomes more reliable as more relevant documents are found [24], our main goal is to find as many relevant documents as possible for a given budget. In the first phase (topic selection), we want to select whichever topic is most likely to supply relevant documents. In the second phase (document selection), we want to select one or more documents for the given topic that are the most likely to be relevant. We implement the first phase via a multi-armed bandit (MAB) [21] method. We are not familiar with any prior work exploring MAB for intelligent topic selection. During the second phase, we investigate several alternative active learning [25] (AL) strategies to select which documents to judge for each search topic. Finally, we use the collected relevance judgments to update both the active learning classifier for document selection and the bandit statistics for topic selection, thereby iteratively improving topic and document selection in successive rounds.

Across three TREC collections with varying scarcity of relevant documents, our best method achieves $\tau \approx 0.90$ Kendall correlation on average with the original system ranking for P@10, using less than 50% of the budget allotted in the original pool. Results for MAP-based evaluation are weaker but still encouraging, suggesting the approach is promising but future work is still needed.

[1] https://github.com/mdmustafizurrahman/MAB_AL_TestCollection/

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313675

2 RELATED WORK

A considerable amount of research [1, 2, 8, 9, 17–20, 30, 31] has been conducted to select documents for human annotation. These approaches can be mainly categorized into two groups: i) static [1, 2, 19, 30, 31] and ii) dynamic [8, 9, 17, 18, 20] selection. In a static selection process, the relevance judgment process is initiated only after the document selection process is completed. In contrast, in a dynamic selection process, document selection and relevance judgment process occur in a feedback loop, where the relevance judgments that are collected so far affect the selection process of the subsequent documents. Our proposed approach is closely related to the dynamic selection approaches. However, our approach does not require organizing a shared-task while prior work relies on the runs submitted by participants of a shared-task.

2.1 Document Selection Strategies

Various dynamic document selection methods have been proposed to intelligently select which of the documents retrieved in a shared task should be judged. The *Move-to-front* (MTF) pooling [9] is one of the pioneer examples of the dynamic document selection process. For a given topic, MTF maintains a priority list of submitted runs. At any round, MTF selects the top-ranked unjudged document from the current highest priority run for annotation. Then based on the relevance judgment of the selected document, MTF updates the priority of the selected run and moves to the next round.

Carterette et al. [3] propose to construct minimal test collections to judge ranking systems with confidence. They select documents to be judged iteratively such that the document that might differentiate systems' performances the most is selected at each iteration. However, their method completely depends on submissions from multiple systems, while the main goal of our work is eliminating the massive cost of running shared-tasks.

Rajput et al. [20] develop a framework to construct test collections using an iterative reinforcement method between nuggets and documents. Their iterative process starts with a few manually created nuggets which are utilized to select documents for the relevance judgment. The relevant documents found in the relevance judgment are used to update the weights of the existing nuggets and extract new nuggets automatically which are then used in the subsequent iteration of document selection. However, should automatic nugget extraction fails (e.g. Web Track 2009), their document selection process solely depends on the submitted runs of the shared task, whereas we consider a no shared task context.

Losada et al. [18] develop a document selection process by utilizing several multi-armed bandits [21] approaches. Their best performing approach, the *MaxMean* (MM) approach, assigns a weight to a run proportional to the ratio of the number of relevant documents found, and the total number of documents judged so far for that run; then it selects the run with the highest weight. Losada et al. empirically prove their superiority over MTF in terms of the number of relevant documents found.

However, Li and Kanoulas [17] point out that dynamic document selection based on a shared task context creates a bias towards the good runs (e.g., runs with more relevant documents). Their proposed approach also utilizes a shared task by inducing a probability distribution from the participating systems, and a probability distribution over the ranks of the documents; then it actively samples documents from the joint distribution to construct a test collection.

Recently, Voorhees [28] further supports Li and Kanoulas's statement by developing the TREC 2017 Common Core track collection via utilizing *MaxMean*(MM) proposed by Losada et al. [18]. The author finds out that only a single run contributes the large percentage of relevant documents in the collection [28]. In contrast, our test collection construction approach in the absence of a shared task is free from this bias towards good runs.

The use of AL in our document selection phase closely follows work by Cormack and Grossman [7] (and we adopt their nomenclature). However, their study is situated in a rather different domain, "e-discovery", focusing on set-based retrieval rather than ranked retrieval. Moreover, the judging cost is measured differently in this domain: no document can be selected automatically since all of the documents must be reviewed for privilege following discovery.

2.2 Topic Selection Strategies

Previous studies [11, 12, 14, 24, 28, 32] treat topic selection as a budget allocation problem. These studies either allocate an immutable budget for a topic at the outset of the relevance judgment using various information e.g., the number of relevant documents at the *top-k* pool [28, 32], or recommend to utilize many topics with a fewer number of relevance judgments [24].

Zobel [32] proposes to collect more judgments for topics with a higher estimated number of relevant documents. The estimation is computed using the relevance judgment collected for each topic in its *top-k* pool, and according to Zobel, it follows the power law distribution with the rank of the document. Voorhees follows the same *top-k* pool with a different formulation ([28], Page 411) to estimate the budget for a topic. However, none of the approaches treat the budget allocation as a topic selection problem as we have done here, where the selection of a subsequent topic for relevance judgment depends on the relevance judgments collected so far. Besides, both of the approaches [28, 32] utilize a shared task which is absent in our task setting.

Prior work [11, 12, 14] also investigates the possibility of selecting a subset of topics. Guiver et al. [11] experimentally show that if we can find a "right" subset of topics, we can achieve a ranking of systems that is very similar to the ranking when all of the topics are used for evaluation. Even though they do not propose a method for finding the "right" subset of topics, their work motivated many researchers to work on the topic selection problem [12, 14]. Studies on topic selection treat each selected topic equally (i.e. collect the same amount of judgments per each), while no judgments are selected for the topics that are not selected. On the contrary, we collect a varying number of judgments per topic.

3 PROPOSED APPROACH

Our approach consists of two phases which alternate between topic selection and document selection. In topic selection phase, we estimate the prevalence of relevant documents for each topic, and select the topic with the highest estimated prevalence. In document selection phase, we use a classifier to predict the relevance of all unjudged documents for the selected topic, and use those predictions

to select which documents to judge next for that topic. We then use the collected relevance judgments to update both the given classifier and the topic statistics. This iterative process of topic and document selection continues until the budget is exhausted.

Algorithm 1 describes our two-phase approach in detail.

Algorithm 1: Two-phase Topic and Document Selection

Input : Unjudged documents U • batch size N • Budget b • Document selection policy p
Output: Relevance judgments $R^{1:J}$ for topics $1:J$

$R^{1:J} \leftarrow \emptyset$
for *topic* $j \leftarrow 1$ **to** J **do**
  Select seed document set $S^j \in U$ for topic j
  $R^j \leftarrow$ judge_relevance (j, S^j) ▷ Collect judgments
  $U \leftarrow U - S^j$ ▷ Update set of unjudged documents
  $b \leftarrow b - |S^j|$ ▷ Update remaining budget
  $C^j \leftarrow$ train_classifier (S^j, R^j)
  $\alpha_j \leftarrow 1, \beta_j \leftarrow 1$ ▷ Initialize Beta distribution for bandit

while remaining budget b >= batch size N **do**
  for *topic* $j \leftarrow 1$ **to** J **do**
    $P_j \leftarrow$ sample probability from $Beta\,(\alpha_j, \beta_j)$
  Selected topic $j \leftarrow \underset{j}{\text{argmax}}\, P_j$
  $\widehat{R^j} \leftarrow$ predict_relevance (C^j, U) ▷ for all unjudged
  S^j = select_documents (U, $\widehat{R^j}$, batch size N, policy p)
  $R^j \leftarrow$ judge_relevance (j, S^j) ▷ Collect judgments
  $U \leftarrow U - S^j$ ▷ Update set of unjudged documents
  $b \leftarrow b - N$ ▷ Update remaining budget
  $C^j \leftarrow$ update_classifier (C^j, S^j, R^j)

  for *document* $i \leftarrow$ **to** N **do** ▷ Update bandits
    if $R^j(i)$ *is relevant* **then**
      $\alpha_j \leftarrow \alpha_j + 1$
    else
      $\beta_j \leftarrow \beta_j + 1$

3.1 Phase 1: Topic Selection

Each topic may need different number of judgments based on the number of relevant documents found in the collection. Thus, allocating a static pre-defined budget for all topics may incur more/less cost than required for topics. Therefore, we prefer to allocate the budget across topics dynamically based on the needs of each topic. Given the fact that we do not have any prior knowledge about the number of judgments needed for a topic, an exploration-exploitation situation arises naturally, which can be approached using the multi-armed bandits (MAB) techniques [21].

In the standard MAB problem, we have to repeatedly make a choice among J bandits (or slot machines), each of which has a hidden probability of winning and losing. Depending on the slot machine j we select at round t, we receive a reward (e.g. 1 for winning, or 0 for losing). In the long run, we want to maximize our final reward. This multi-armed bandits problem is a natural fit for our topic selection problem. Each topic j can be referred to as a slot machine and each of these topics has a hidden probability of supplying a relevant document. At each round t, based on the selected topic j, we either receive a relevant or non-relevant document. At any round t, we can estimate which topic is more likely to provide more relevant documents. The exploration-exploitation dilemma arises when deciding between whether we should keep selecting that topic or we should explore other topics; we do not know which decision will help us maximize the number of relevant documents.

To allocate the budget across topics dynamically, we apply a Bayesian approach [10] where each topic's hidden probability distribution over relevant and non-relevant documents is endowed with a prior distribution. With no prior knowledge about the topic's prevalence, we start with a uniform prior for each topic. Note that a uniform distribution is a special case of a $Beta(\alpha, \beta)$ distribution, when $\alpha = 1$ and $\beta = 1$. In our Algorithm 1, α and β denote the number of relevant and non-relevant documents found so far for a topic j, respectively. At any round t, we start by selecting a topic j from the $Beta(\alpha, \beta)$ distribution. Next we select N documents to judge using a document selection process (Section 3.2) for topic j. Finally, the binary outcome, O, (either relevant or non-relevant) for each selected document is used to update the hidden probability distribution of topic j. With a $Beta(\alpha, \beta)$ prior and a binary outcome, the posterior O is also a $Beta(\alpha + O, \beta + 1 - O)$ distribution.

3.2 Phase 2: Document Selection

In this phase, we select N documents to judge for a given topic j. However, in the absence of a shared-task, we do not have any ranking information of documents. Therefore, if we select documents randomly, it is very unlikely to find any relevant document. Thus, we employ active learning [25], a learning paradigm where a classifier decides which document should be annotated next in order to maximize the learning curve of the classifier.

3.2.1 Task Definition and Learning Model. To train a topic-specific active classifier, we must collect the topic-specific training data. Let us assume that a training pair for a topic j is denoted by $\langle x^i, y^i_j \rangle$, where x^i denotes the feature representation of document, i and y^i_j denote the binary relevance judgment for <document i, topic j>. We adopt logistic regression as a classifier to infer the probability of relevance $P(y^i_j|x^i)$ of each document x^i for topic j:

$$p(y^i_j|x^i) = h_\theta(x_i) = \frac{1}{1 + \exp(-\vec{\theta}^T x_i)} \tag{1}$$

with $\vec{\theta} \in \mathbb{R}^D$ denotes model parameters.

3.2.2 Document Selection Criteria. Utilizing the posteriori probability of documents for a given topic j, we can select documents to be judged in two different ways: *Simple Active Learning* (SAL) and *Continuous Active Learning* (CAL).

SAL [15] selects a document for relevance judgment when the classifier is most uncertain about the label of that document. We employ an entropy-based uncertainty function [25] for this:

$$Uncertainty(x) = -\sum_{y \in Y} P(y|x) \log P(y|x) \tag{2}$$

Table 1: Test collection statistics. As collections have grown larger, judging budgets have also shrunk, leading to increased prevalence of relevant documents in later tracks.

Track	Collection	Topics	#Docs	#Judged	%Rel
WT'14[6]	ClueWeb12[4]	251-300	52M	14,432	39.2%
WT'13[5]	ClueWeb12	201-250	52M	14,474	28.7%
TREC-8[29]	Disks45-CR[5]	401-450	528K	86,830	5.4%

where y is either relevant or non-relevant. With binary relevance, SAL selects:

$$x^{\star} = \underset{i}{\operatorname{argmin}} \ |p(relevant|x^i) - 0.5| \tag{3}$$

In contrast, CAL selects a document based on how likely the document is to be relevant according to the prediction of the classifier.

$$x^{\star} = \underset{i}{\operatorname{argmax}} \ p(relevant|x^i) \tag{4}$$

3.2.3 Seed Document Selection. In order to learn an initial classifier for each topic, a minimal *seed set* of relevance judgments for each topic is required. We assume a *single* off-the-shelf or in-house IR system (e.g., Apache Lucene[2] or Indri[3]) is used to produce a document ranking for each topic. The assessor is then asked to proceed down the document ranking for each topic until at least m number of relevant and m number of non-relevant documents have been found, or some maximum effort is reached without success, in which case, the topic is discarded. In this paper, we assume $m = 5$. Note that this single IR system is used only once at the outset to guide seed set annotation, and one might even do without this if one is willing to resort to boolean search or random selection to identify the seed documents [7].

4 EXPERIMENTS

4.1 Experimental Setup

Datasets. We conduct our experimental evaluation on three TREC test collections (See **Table 1**). We assume binary relevance and collapse NIST graded relevance judgments to binary.

We use only pooled documents of each collection because unpooled documents are not judged and typically assumed to be not-relevant. However, relevant documents may exist outside the pool, and assuming them to be non-relevant could hurt classifier training and prediction. One could judge the unpooled documents selected by our method, though this could yield inconsistency between the original judgments and new judgments. This issue could be usefully revisited in future work. In the remainder of this paper, we refer to the set of pooled judged documents as the "qrels."

Document Selection. In the document selection phase, we have to represent the documents in a feature vector for the classifier of the active learning approach. We first pre-process the documents using `IndriBuildIndex`[6]. The pre-processing consists of text normalization, stopword [16] removal, and Krovetz stemming [13]. Finally, each pre-processed document is represented using a 15K dimensional TF-IDF [22] vector. Each iteration of AL selects one document to be judged next. In order to select the seed documents of AL, we randomly select one of the runs for each test collection and assume it as our off-the-shelf IR system.

Baselines. We have one baseline method for document selection and another baseline method for topic selection.

- **Simple Passive Learning (SPL):** This baseline method for document selection uses uniform random selection.
- **Round-Robin (RR):** RR simply cycles through topics and then repeats, thus allocating the same budget for each topic. No information is used regarding topic prevalence ratio or collected relevance judgments.

Upperbound. We develop **Oracle** as the upperbound for our topic selection method. Whereas MAB must learn the prevalence ratio of topics during dynamic topic selection, the Oracle knows the exact prevalence ratio of each topic at each round t. To do this, we initialize the α and β parameters of each topic with the topic's total number of relevant and non-relevant documents in the original qrels. At each round t, we update the parameters, decreasing α and β based on the relevance judgments collected during that round.

Table 2: Avg. number of relevant documents found under varying budget per topic on TREC-8. For MoveToFront (MTF), MaxMean Non-Stationary (MM-NS), and MAB+CAL topic selection, MAB+CAL consistently performs best.

	Average number of judgments per topic						
Method	**100**	**300**	**500**	**700**	**900**	**1100**	**all**
MTF	34.06	58.48	71.78	79.22	84.5	87.58	94.04
MM-NS	36.96	64.62	77.3	82.5	86.34	89.2	94.04
MAB+CAL	**46.3**	**78.4**	**86.5**	**90.3**	**91.3**	**93.5**	94.04

4.2 Results and Discussion

Effectiveness of topic selection. How well can MAB identify relevant documents? **Figure 1** compares the recall of MAB with Oracle and RR for WT2014 and WT2013 collections. The overall Area Under Curve (AUC) effectiveness across all budget points is reported in Figure 1. The x-axis of each plot represents the total allotted budget. Ultimately, each strategy achieves complete recall because all pooled documents are judged. As expected, RR is the weakest performing topic selection method across all six plots of Figure 1, suggesting that we should select topics intelligently to maximize recall of relevant documents. Also as expected, Oracle performs best in every case, and the recall curve of MAB consistently lies between Oracle and RR.

Dynamic vs. random document selection. Figure 1 also reports the performance of the three document selection strategies. We observe that CAL outperforms the SPL baseline in both test collections and in all topic selection strategies. However, we cannot infer the same conclusion about SAL because SAL outperforms SPL only in 3/6 of cases based on AUC scores. CAL selects the documents that are most likely to be relevant, yielding a higher recall than others. On the other hand, SAL selects documents whose relevance is uncertain, and thus tends to select more non-relevant documents, ultimately decreasing its recall.

The allotted budget for each topic. In **Figure 2**, we report the budget allotted to each topic using CAL for document selection, and MAB and RR for topic selection on TREC8, WT2013, and WT2014

[2] https://lucene.apache.org/
[3] https://www.lemurproject.org/indri.php
[6] www.lemurproject.org/indri.php

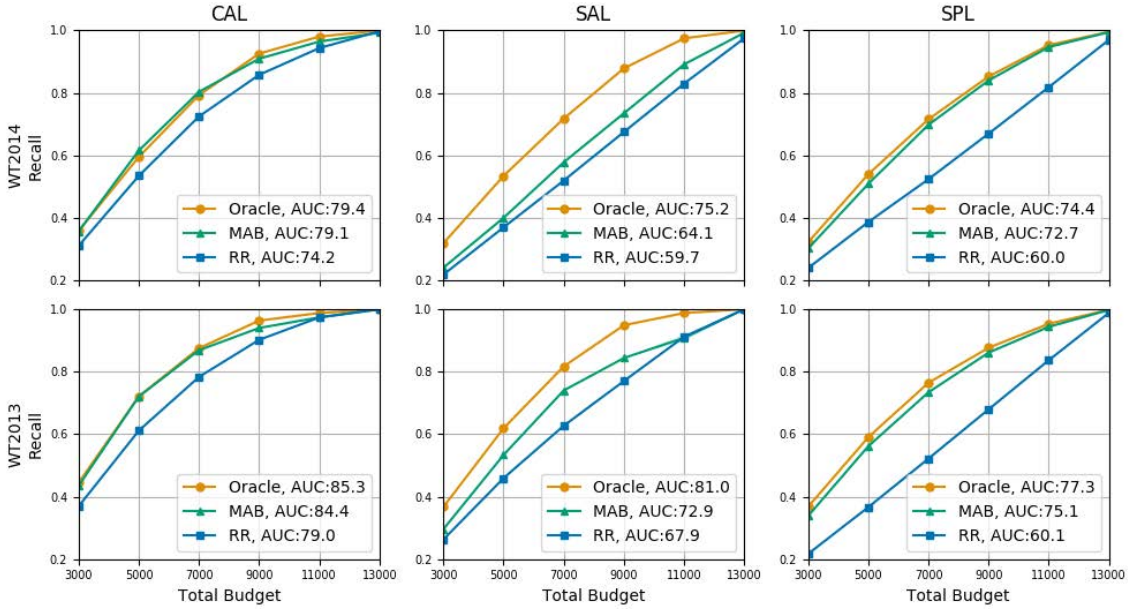


Figure 1: Recall of relevant documents achieved by Oracle, Multi-armed bandits (MAB), and Round-robin (RR) topic selection methods as a function of varying evaluation budget on WT2014 and WT2013 collections. The x-axis shows total cost over all topics. Plots are grouped vertically for CAL, SAL, and SPL document selection methods.

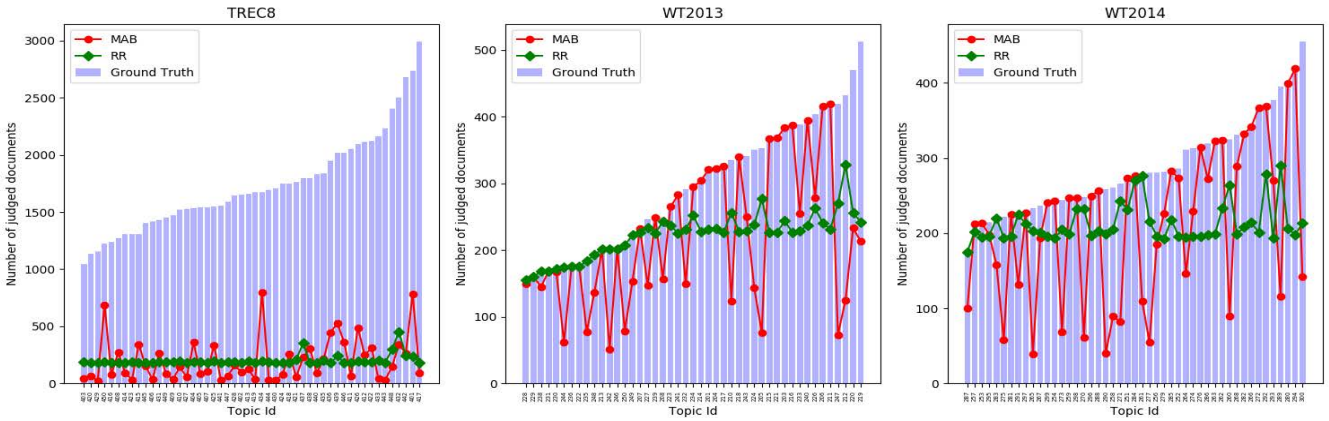


Figure 2: The budget (# of judged documents) across topics using MAB and RR with CAL document selection on TREC8, WT2013 and WT2014 test collections with total budget of 10,000. The bar plots show per-topic budget in the original qrels.

when the overall budget is 10,000. We can see that MAB-based topic selection varies the judging budget used across topics. For RR, the number of judgments per topic varies only due to the varying number of seed documents used to start AL. Ignoring the cost of seed documents, all topics would have the same number of judgments per topic in RR. Thus as expected, RR has a lower variance in budget allocation per topic than MAB.

Comparison against pool-based methods. We compare our approach, MAB with CAL (MAB+CAL) against two state-of-the-art pool-based dynamic document selection methods: MoveToFront (MTF) [9] and the MaxMean Non-Stationary (MM-NS) [18]. **Table 2** presents the average number of relevant documents found per topic of TREC-8 collection. We copy the MM and MTF results reported in [18]. Note that for MTF and MM-NS, we have to manually specify a fixed budget per topic. In contrast, MAB+CAL intelligently learns the budget allocation per topic dynamically.

Table 2 shows that MAB+CAL finds a higher average number of relevant documents per topic than MTF and MM-NS in all cases, *despite* the fact that both MTF and MM-NS utilize the document ranking information from the submitted runs in a shared task.

Rank Correlation. One of the ways to evaluate the reliability of a low-cost evaluation method is to compare the resultant ranking of IR systems with the ranking produced by all qrels. Thus, we build qrel sets using CAL for document selection, and RR, MAB, and

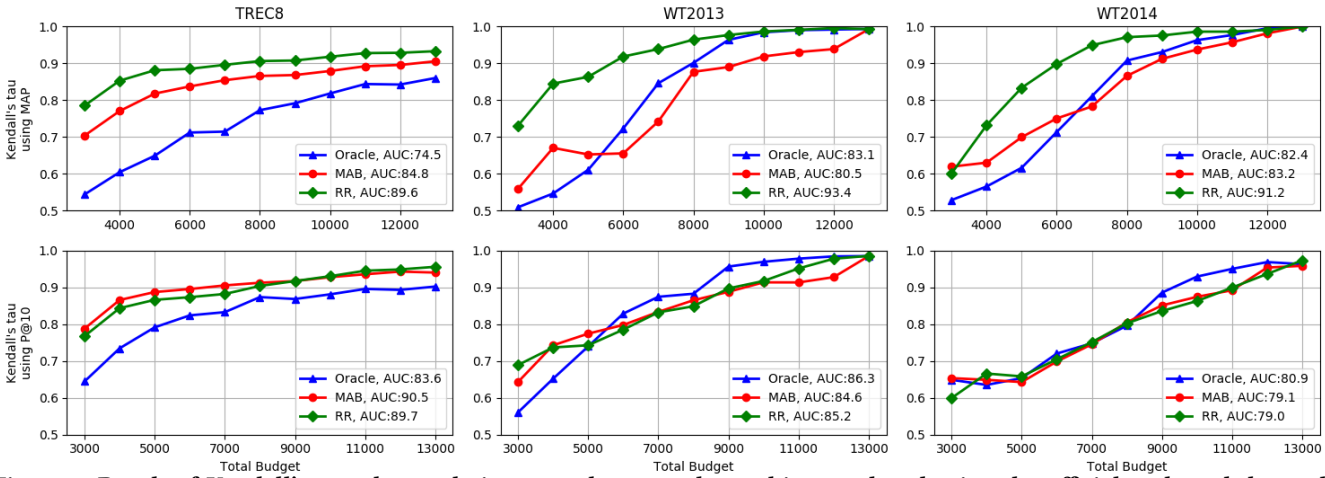


Figure 3: Result of Kendall's τ rank correlation score between the ranking produced using the official qrels and the ranking produced using qrels created by Oracle, MAB and RR with CAL over varying budgets on TREC8, WT2014 and WT2013 collections. Rankings are produced using MAP (top row) and P@10 (bottom row).

Oracle for topic selection varying the total budget from 3K to 13K, and calculate Kendall's τ rank correlation between ground truth ranking and the resultant ranking for each case. We use MAP and P@10 as the evaluation metric. The results are shown in **Figure 3** for TREC8, WT2013 and WT2014 test collections.

For the bottom row of Figure 3 (Kendall's τ rank correlation for P@10 evaluation), we see that with low judging budget, MAB provides better τ correlation than RR for test collections with lower prevalence ratio per topic (i.e., TREC8). Specifically, MAB achieves $\tau = 0.9$ (a traditionally-accepted threshold for acceptable correlation [27]) when the total budget is 6500, which is only 7.4% of the original allotted budget for TREC8. However, results are less clear for test collections with a higher prevalence ratio. For example, MAB outperforms RR for WT2014 but RR outperforms MAB for WT2013 in terms of AUC.

When we compute Kendall's τ for MAP evaluation (Figure 3, Top Row), both Oracle and MAB actually perform worse than RR across test collections. Since MAP is a recall-based metric, it considers the full judgment pool for each topic, whereas P@10 takes account only the first 10 documents. This suggests the current MAB approach may work well for shallow metrics but not for deep ones.

Recall that RR allocates the same budget across all topics, whereas MAB-based topic selection yields a varying number of documents across topics (Figure 2). This is because MAB seeks to optimize the number of relevant documents [9, 18]. Oracle optimizes this same objective function as MAB but even better, knowing the exact prevalence ratio of each topic, which MAB must learn. Results thus suggest that our MAB-based approaches spend the budget excessively on topics with high prevalence ratio, causing judging few documents for the topics with low prevalence ratio. Sometimes, MAB outperforms Oracle seemingly because its imperfect estimate of prevalence leads it to explore more and exploit less. Thus, while MAB approaches succeed in finding more relevant documents, we observe a metric divergence between what is being optimized (recall) and what we actually care about: reliable evaluation, as measured by rank correlation. Moreover, results underscore the importance of evaluating test collection creation methods via rank correlation, and not only by recall of relevant documents [18].

Furthermore, when we compare the results across collections for MAP score in Figure 3, we find that the performance difference between RR and MAB-based approaches are much higher in collections with a high prevalence ratio (i.e., WT2013 and WT2014) than a low prevalence ratio (i.e., TREC8). This signifies the fact that in terms of MAP, MAB-based approach produces a more reliable evaluation for test collections with a low prevalence ratio than test collections with a high prevalence ratio. This is another promising result of our MAB-based approach which encourages further empirical evaluation.

5 CONCLUSION AND FUTURE WORK

This work investigates the feasibility of developing a minimal cost test collection without organizing a shared task. In the absence of shared task document rankings to prioritize relevance judging, we utilize active learning [25] instead. Furthermore, we intelligently and dynamically allocate judging effort across topics via multi-armed bandits [21]. Results on the three different TREC test collections indicate that by utilizing an average of only 48% of the total budget (Table 1) allocated in these test collections, our best approach, MAB+CAL, not only finds 90% of the relevant documents (Figure 1) but also achieves a Kendall's τ rank correlation value of 0.90 with the original system ranking when the correlation is computed using P@10 (Figure 3, Bottom Row). However, further work is needed to effectively support more recall-oriented IR evaluation metrics such as MAP. Current results suggest that our MAB-based approaches spend the budget excessively on topics with high prevalence ratio, judging too few documents for topics with low prevalence ratio. Future work might usefully explore more sophisticated bandit approaches (e.g. contextual bandits).

ACKNOWLEDGMENTS

This work was supported by NPRP grant # NPRP 7-1313-1-245 from the Qatar National Research Fund. The statements made herein are solely the responsibility of the authors.

REFERENCES

[1] Javed A Aslam, Virgil Pavlu, and Emine Yilmaz. 2006. A statistical method for system evaluation using incomplete judgments. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 541–548.

[2] Javed A Aslam and Emine Yilmaz. 2007. Inferring document relevance from incomplete information. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM, 633–642.

[3] Ben Carterette, James Allan, and Ramesh Sitaraman. 2006. Minimal test collections for retrieval evaluation. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 268–275.

[4] Cyril Cleverdon. 1967. The Cranfield tests on index language devices. In *Aslib proceedings*, Vol. 19. MCB UP Ltd, 173–194.

[5] Kevyn Collins-Thompson, Paul N. Bennett, Fernando Diaz, Charlie Clarke, and Ellen M. Voorhees. 2013. TREC 2013 Web Track Overview. In *Proceedings of The Twenty-Second Text REtrieval Conference, TREC 2013, Gaithersburg, Maryland, USA, November 19-22, 2013*.

[6] Kevyn Collins-Thompson, Craig Macdonald, Paul Bennett, Fernando Diaz, and Ellen M Voorhees. 2015. *TREC 2014 web track overview*. Technical Report. DTIC Document.

[7] Gordon V Cormack and Maura R Grossman. 2014. Evaluation of machine-learning protocols for technology-assisted review in electronic discovery. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 153–162.

[8] Gordon V Cormack and Maura R Grossman. 2018. Beyond pooling. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 1169–1172.

[9] Gordon V Cormack, Christopher R Palmer, and Charles LA Clarke. 1998. Efficient construction of large test collections. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 282–289.

[10] Ole-Christoffer Granmo. 2008. A Bayesian Learning Automaton for Solving Two-Armed Bernoulli Bandit Problems. In *Proceedings of the 2008 Seventh International Conference on Machine Learning and Applications (ICMLA '08)*. IEEE Computer Society, Washington, DC, USA, 23–30. https://doi.org/10.1109/ICMLA.2008.67

[11] John Guiver, Stefano Mizzaro, and Stephen Robertson. 2009. A few good topics: Experiments in topic set reduction for retrieval evaluation. *ACM Transactions on Information Systems (TOIS)* 27, 4 (2009), 21.

[12] Mehdi Hosseini, Ingemar J Cox, Natasa Milic-Frayling, Milad Shokouhi, and Emine Yilmaz. 2012. An uncertainty-aware query selection model for evaluation of IR systems. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 901–910.

[13] Robert Krovetz. 1993. Viewing morphology as an inference process. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 191–202.

[14] Mucahid Kutlu, Tamer Elsayed, and Matthew Lease. 2018. Intelligent topic selection for low-cost information retrieval evaluation: A New perspective on deep vs. shallow judging. *Information Processing & Management* 54, 1 (2018), 37–59.

[15] David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*. Springer-Verlag New York, Inc., 3–12.

[16] David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. 2004. SMART stopword list. *Journal of Machine Learning Research* (2004).

[17] Dan Li and Evangelos Kanoulas. 2017. Active Sampling for Large-scale Information Retrieval Evaluation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 49–58.

[18] David E Losada, Javier Parapar, and Álvaro Barreiro. 2016. Feeling lucky?: multi-armed bandits for ordering judgements in pooling-based evaluation. In *proceedings of the 31st annual ACM symposium on applied computing*. ACM, 1027–1034.

[19] V Pavlu and J Aslam. 2007. *A practical sampling strategy for efficient retrieval evaluation*. Technical Report. College of Computer and Information Science, Northeastern University.

[20] Shahzad Rajput, Matthew Ekstrand-Abueg, Virgil Pavlu, and Javed A Aslam. 2012. Constructing test collections by inferring document relevance via extracted relevant information. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 145–154.

[21] Herbert Robbins. 1985. Some aspects of the sequential design of experiments. In *Herbert Robbins Selected Papers*. Springer, 169–177.

[22] Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management* 24, 5 (1988), 513–523.

[23] Mark Sanderson et al. 2010. Test collection based evaluation of information retrieval systems. *Foundations and Trends® in Information Retrieval* 4, 4 (2010), 247–375.

[24] Mark Sanderson and Justin Zobel. 2005. Information retrieval system evaluation: effort, sensitivity, and reliability. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 162–169.

[25] Burr Settles. 2012. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6, 1 (2012), 1–114.

[26] Ian M Soboroff. 2013. Building Test Sollections (without running a community evaluation). In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. 1132–1132. https://isoboroff.github.io/Test-Colls-Tutorial/Tutorial-slides/.

[27] Ellen M Voorhees. 2000. Variations in relevance judgments and the measurement of retrieval effectiveness. *Information processing & management* 36, 5 (2000), 697–716.

[28] Ellen M. Voorhees. 2018. On Building Fair and Reusable Test Collections Using Bandit Techniques. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. ACM, New York, NY, USA, 407–416. https://doi.org/10.1145/3269206.3271766

[29] Ellen M. Voorhees and Donna Harman. 2000. Overview of the Eighth Text REtrieval Conference (TREC-8). 1–24.

[30] Emine Yilmaz and Javed A Aslam. 2006. Estimating average precision with incomplete and imperfect judgments. In *Proceedings of the 15th ACM international conference on Information and knowledge management*. ACM, 102–111.

[31] Emine Yilmaz and Javed A Aslam. 2008. Estimating average precision when judgments are incomplete. *Knowledge and Information Systems* 16, 2 (2008), 173–211.

[32] Justin Zobel. 1998. How reliable are the results of large-scale information retrieval experiments?. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 307–314.

A Multi-modal Neural Embeddings Approach for Detecting Mobile Counterfeit Apps

Jathushan Rajasegaran
Data61, CSIRO
Sydney, Australia
brjathu@gmail.com

Naveen Karunanayake
Data61, CSIRO
Sydney, Australia
naveenharshitha95@gmail.com

Ashanie Gunathillake
The University of Sydney
Sydney, Australia
ashanie87@gmail.com

Suranga Seneviratne
The University of Sydney
Sydney, Australia
suranga.seneviratne@sydney.edu.au

Guillaume Jourjon
Data61, CSIRO
Sydney, Australia
guillaume.jourjon@data61.csiro.au

ABSTRACT

Counterfeit apps impersonate existing popular apps in attempts to misguide users. Many counterfeits can be identified once installed, however even a tech-savvy user may struggle to detect them before installation. In this paper, we propose a novel approach of combining *content embeddings* and *style embeddings* generated from pre-trained convolutional neural networks to detect counterfeit apps. We present an analysis of approximately 1.2 million apps from Google Play Store and identify a set of potential counterfeits for top-10,000 apps. Under conservative assumptions, we were able to find 2,040 potential counterfeits that contain malware in a set of 49,608 apps that showed high similarity to one of the top-10,000 popular apps in Google Play Store. We also find 1,565 potential counterfeits asking for at least five additional dangerous permissions than the original app and 1,407 potential counterfeits having at least five extra third party advertisement libraries.

CCS CONCEPTS

• **Networks → Mobile and wireless security**.

KEYWORDS

Mobile Apps; App Security; Security; Fraud Detection

ACM Reference Format:
Jathushan Rajasegaran, Naveen Karunanayake, Ashanie Gunathillake, Suranga Seneviratne, and Guillaume Jourjon. 2019. A Multi-modal Neural Embeddings Approach for Detecting Mobile Counterfeit Apps. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313427

1 INTRODUCTION

Availability of third party apps is one of the major reasons behind the wide adoption of smartphones. The two most popular app markets, Google Play Store and Apple App Store, hosted approximately 3.5 million and 2.1 million apps by early 2018 [16, 24]. Handling such large numbers of apps is challenging for app market operators since there is always a trade-off between how much scrutiny is put into checking apps and encouraging developers by providing fast time-to-market. As a result, problematic apps of various kinds including malware have made it into the app markets [42, 56].

One category of such problematic apps is *counterfeits* (i.e. apps that attempt to impersonate popular apps). Reasons behind app impersonations include harvesting user credentials, increasing advertising revenue, and spreading malware. Many popular apps such as *Netflix*, *IFTTT*, *Angry Birds*, and *banking apps* have been reported to be affected by counterfeits [8, 25, 39, 50]. In Figure 1, we show an example counterfeit named *Temple Piggy*[1] which shows a high visual similarity to the popular arcade game *Temple Run*.[2]



a) Original (Temple Run) b) Counterfeit (Temple Piggy)

Figure 1: An example counterfeit for the game Temple Run

In this paper, we propose a neural embedding-based approach to identify counterfeit apps from a large corpus of apps. We leverage the recent advances in Convolutional Neural Networks (CNNs) to generate feature embeddings from given images using pre-trained models such as VGGNet [45]. In contrast to commonly used *content embeddings* generated from fully connected layers before the last soft-max layer, we show that combining *content embeddings* with *style embeddings* generated from the Gram matrix of convolutional layer feature maps achieve better results in detecting visually similar app icons. Following are the main contributions of this paper.

- We show that the novel method of using combined *style* and *content* embeddings generated from pre-trained CNNs outperforms many baseline image retrieval methods for the task of detecting visually similar app icons. We also validate this method using two standard image retrieval datasets.
- Using a large dataset of over 1.2 million app icons, we show that *combined content and style embeddings* achieve 8%–12% higher *precision@k* and 14%–18% higher *recall@k*.

[1] *Temple Piggy* is currently not available in Google Play Store.
[2] *Temple Run* - https://play.google.com/store/apps/details?id=com.imangi.templerun.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313427

- We show that adding *text embeddings* [31] as an additional modality, further increases the performance by 3%–5% and 6%–7% in terms of *precision@k* and *recall@k*.
- We identify a set of 7,246 potential counterfeits to the top-10,000 apps in Google Play and show that 2,040 of them may contain malware. We further show that out of those; 1,565 ask for at least five extra dangerous permissions and 1,407 have at least five extra third party ad libraries.

2 RELATED WORK

2.1 Mobile Malware & Greyware

While there is a plethora of work on detecting mobile malware [14, 22, 44, 52, 54] and various fraudulent activities in app markets [15, 21, 43, 47, 53], only a limited amount of work focused on the *similarity of mobile apps*. Viennot et al. [49] used the Jaccard similarity of app resources in the likes of images and layout XMLs to identify clusters of similar apps and then used the developer name and certificate to differentiate clones from rebranding. Crussell et al. [17] proposed to use features generated from the source codes to identify app clones. In contrast to above work, our work focuses on identifying visually similar apps (i.e. counterfeits) rather than the exact similarity (i.e. clones), which is a more challenging problem.

Limited amount of work focused on identifying visually similar mobile apps [7, 35, 36, 46]. For example, Sun et al. [46] proposed DroidEagle that identifies the visually similar apps based on XML layouts. While the results are interesting this method has several limitations. First, all visually similar apps may not be necessarily similar in XML layouts and it is necessary to consider the similarities in images. Second, app developers are starting to use code encryption methods, thus accessing codes and layout files may not always possible. Third, dependency of specific aspects related to one operating system will not allow to make comparisons between heterogeneous app markets. Recently, Malisa et al. [35] studied how likely would users detect a spoofing application using a complete rendering of the application itself. In contrast to above work, the proposed work intends to use neural embeddings derived from app icons and text descriptions that will better capture visual and functional similarities.

2.2 Visual Similarity & Style Search

Several work focused on transferring style of an image to another. For example, Gatys et al. [18, 20] proposed a *neural style transfer* that is able to transfer the stylistic features of well-known artworks to target images. Other methods proposed to achieve the same objective either by updating pixels in the image iteratively or by optimising a generative model iteratively and producing the styled image through a single forward pass. A summary of style transfer algorithms can be found in the survey by Jing et al. [27].

Johnson et al. [28] proposed a feed-forward network architecture capable of real-time style transfer by solving the optimisation problem formulated by Gatys et al. [20]. Similarly, to style transfer, CNNs have been successfully used for image searching. In particular, Bell & Bala [12] proposed a Siamese CNN to learn a high-quality embedding that represent visual similarity and demonstrated the utility of these embeddings on several visual search tasks such as searching similar products. Tan et al. [48] and Matsuo & Yanai [37] used embeddings created from CNNs to classify artistic styles. In contrast to these works, our work focuses on retrieving visually similar Android apps and we highlight the importance of style and combined (multi-modal) embeddings in this particular problem.

3 DATASET

We collected our dataset by crawling Google Play Store using a Python crawler between January and March, 2018. The crawler was seeded with the web pages of the top apps as of January, 2018 and it recursively discovered apps by following the links in the seeded pages and the subsequently discovered pages. For each app, we downloaded the metadata such as app name, app description, and number of downloads as well as the app icon in *.jpg* or *.png* format (of size 300 x 300 x 3 - height, width, and three RGB channels).

We collected information of ***1,278,297 apps*** during this process. For each app, we also downloaded the app executable using *Google Play Downloader* [1] by simulating a Google Pixel. We were able to download APKs for ***1,023,521 apps*** out of the total ***1,278,297 apps*** we discovered. The reasons behind this difference are the *paid apps* and the apps that did not support the used virtual device.

Labelled set: To evaluate the performance of various image similarity metrics we require a ground truth dataset that contains similar images to a given image. We used a heuristic approach to shortlist a possible set of visually similar apps and refined it by manual checking. Our heuristic is based on the fact that there are apps having multiple legitimate versions. For example, popular game app *Angry Birds* has multiple versions such as *Angry Birds Rio*, *Angry Birds Seasons*, and *Angry Birds Go* with similar app icons and descriptions.

Thus, we first identified the set of developers who has published more than two apps and one app having at least 500,000 downloads. In the set of apps from the same developer, the app with the highest number of downloads was selected as the *base app*. For each other app in the set, we calculated the *character level cosine similarity* of its *app name* to the base app name and selected the apps having over 0.8 similarity and in the same *app category* as the base app. We identified 2,689 such groups. Finally, we manually inspected each group and discarded the apps that were not visually similar.

During our later evaluations, the highest number of neighbours we retrieve is 20. Therefore, we ensured that the maximum number of apps in a group was 20 by randomly removing apps from the groups having more than 20 apps. At the end of this process we had 806 app groups having a total of 3,539 apps as our ***labelled set***.

Top-10,000 popular apps: To establish a set of potential counterfeits, we used top-10,000 apps since counterfeits majorly target popular apps. We selected top-10,000 popular apps by sorting the apps by the number of downloads, number of reviews, and average rating similar to what was proposed in [42].

Other image retrieval datasets: To benchmark the performance of our proposed combined embeddings, we use two existing ground-truth datasets; UKBench [38] and Holidays [26]. Both datasets contain groups of similar images, containing images of same scenes or objects from different angles and illumination levels.

4 METHODOLOGY

The main problem we are trying to address is that *"given an app can we find potential counterfeits from a large corpus?"*. Since counterfeit

apps focus more on being visually similar to original apps, we mainly focus on finding similar app icons to a given app icon. We also focus on the similarity between text as an additional modality.

4.1 App Icon Encoding and Embeddings

We encode the original app icon image of size 300×300×3 to a lower dimension for efficient search and to avoid false positives happening due to L_2 distance at large dimensions [4]. We create several low dimensional representations of the images. As baseline methods, we use state-of-the-art image *hashing methods*, *feature-based image retrieval methods*, and SSIM (Structural Similarity). From a pre-trained VGGNet, we derive *content* and *style embeddings*.

i) Hashing methods: Hashing methods we evaluate include *average* [29], *difference* [30], *perceptual* [55], and *wavelet* [5] hashing. All four methods first scale the app icon to a 32×32 grayscale image and represent it as a $1x1024$ binary feature vector.

ii) Feature-based image retrieval methods: Feature-based methods extract features from an image and describe them using neighbouring pixels. Thus, such methods have two steps; *feature detection* and *feature description*. Some algorithms perform both tasks together while others perform them separately. In this paper, we use four feature matching methods; *Scale-Invariant Feature Transform (SIFT)* [34], *Speeded-Up Robust Features (SURF)* [11], *Accelerated KAZE (AKAZE)* [6], and *Learned Arrangements of Three Patch Codes (LATCH)* [32]. SIFT and SURF describe an app icon by a $f_i \times 128$ integer descriptor matrix, where f_i is the number of features detected for app icon i. AKAZE and LATCH describe the app icon by a $f_i \times 64$ binary descriptor matrix.

iii) Structural Similarity Index Matrix (SSIM): SSIM [51] compares the local pattern of pixel intensities in two images and calculate a similarity score. This method gives a high similarity score even for images with significant pixel-wise difference as it does not compare images point-by-point basis. SSIM does not represent an image by a vector/matrix. Therefore, we scale the input app icons into a 32×32 grayscale images and calculate the similarity score.

iv) Content embeddings: To extract the content representation of an icon, we used a pre-trained *VGGNet* [45]. We fed all 1.2M app icons to the VGGNet, and used the *content embeddings*, $C \in \mathbb{R}^{4096}$, generated at the last fully connected layer of VGGNet (fc_7 layer) that have shown good results in the past [10, 13].

v) Style embeddings: Content similarity alone is not sufficient for counterfeit detection as sometimes developers keep the visual similarity and change the content. Thus, we require an embedding that represents the style of an image.

Several work demonstrated that the Gram matrix of filter responses of CNNs can be used to represent the style of an image [19, 20]. We followed a similar approach and used the fifth convolution layer ($conv5_1$) of the VGGNet to obtain the style representation of the image, as previous work indicated that $conv5_1$ provides better performance in style similairty [37]. We passed each icon through the VGGNet, and at $conv5_1$ the icon was convolved with pre-trained filters and activated through *ReLU* function. The $conv5_1 - layer$ of the VGGNet we used had 512 filters, resulting a Gram matrix of size $G^5 \in \mathbb{R}^{512\times512}$. We only considered the upper half of the Gram matrix as our style representation vector, $S \in \mathbb{R}^{131,328}$ as the Gram matrix is symmetric. To further reduce the dimension of style embeddings we used the *very sparse random projection* [33] and ensured the size of style embeddings is $1x4096$. In Figure 2, we show a summary of our icon encoding process.

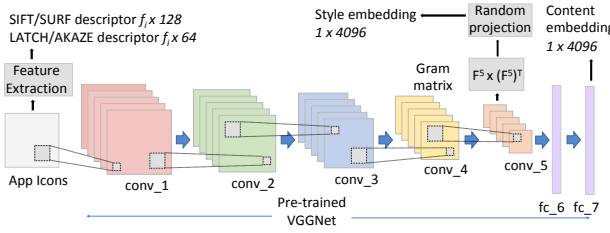


Figure 2: Image embeddings generation process

4.2 Text Embeddings

The app descriptions from Google Play Store can contain a maximum of 4000 characters, describing the apps' functionalities and features. As such, counterfeits are likely to show some similarities to original apps' description. To capture this similarity, we first used standard text pre-processing methods on app descriptions and trained a Paragraph Vectors Model [31] to create vectors of size 100.

4.3 Retrieving Similar Apps

During the similar app retrieval process we take an app and calculated the required embeddings and search in the encoded space for *k nearest neighbours* using *cosine distance*, L_2 *distance*, or *hamming distance* based on the applicability to the embedding under consideration. Let X_i^y be a vectored representation of an app i using the encoding scheme y and X_t^y be the corresponding representation of the target app we are comparing, we calculate the various distance metrics for different representations as summarised in Table 1.

5 RESULTS

5.1 Evaluation of Embeddings

To quantify the performance of the different embeddings, we evaluate them in four different test scenarios using multiple datasets. In each scenario, for a given query embedding, we retrieved *k-nearest neighbours* (k-NN) based on the distances considered in Table 1. We tested four values of k; 5, 10, 15, and 20. The four scenarios are:

i) Holidays dataset: Holidays dataset contains 1,491 images from 500 groups. We took the encoded representation of the first image to search the entire corpus and retrieved the *k-nearest neighbours*.

ii) UKBench dataset: UKBench dataset contains 10,200 images from 2,550 groups. Same method as the Holidays dataset was used.

iii) Apps - Labelled set only: Labelled set contains 3,539 images from 806 groups. From each group, the *base app icon embedding* was taken as the query to retrieve *k-NNs* among remaining icons.

iv) Apps - Labelled set and all remaining icons and text: This dataset contains 1.2M images. The embedding of the base app icon of each group in the labelled set was taken as the query to retrieve the *k-NNs* from the entire dataset.

Table 1: Summary of encoding methods and distance metrics

Encoding Method	Size (n)	Distance function
Hashing methods (Hamming distance)		
Average	1,024	$\|\|X_i^{avg} \oplus X_t^{avg}\|\|_1$
Difference	1,024	$\|\|X_i^{diff} \oplus X_t^{diff}\|\|_1$
Perceptual	1,024	$\|\|X_i^{perc} \oplus X_t^{perc}\|\|_1$
Wavelet	1,024	$\|\|X_i^{wave} \oplus X_t^{wave}\|\|_1$
Feature based methods (L_2 distance)		
SIFT	$f_i^{sift} \times 128$	$\sum_{x_i \in X_i} \min_{x_t \in X_t} [\|\|x_i^{sift} - x_t^{sift}\|\|_2]$
SURF	$f_i^{surf} \times 128$	$\sum_{x_i \in X_i} \min_{x_t \in X_t} [\|\|x_i^{surf} - x_t^{surf}\|\|_2]$
Feature based methods (Hamming distance)		
AKAZE	$f_i^{akaze} \times 64$	$\sum_{x_i \in X_i} \min_{x_t \in X_t} [\|\|x_i^{akaze} \oplus x_t^{akaze}\|\|_2]$
LATCH	$f_i^{latch} \times 64$	$\sum_{x_i \in X_i} \min_{x_t \in X_t} [\|\|x_i^{latch} \oplus x_t^{latch}\|\|_2]$
Structural similarity		
SSIM	Directly returns a dissimilarity	
Neural embeddings (Cosine distance)		
Content (C_{cos})	4,096	$1 - \frac{X_i^{cont}.X_t^{cont}}{\|\|X_i^{cont}\|\|_2 \|\|X_t^{cont}\|\|_2}$
Style (S_{cos})	4,096	$1 - \frac{X_i^{style}.X_t^{style}}{\|\|X_i^{style}\|\|_2 \|\|X_t^{style}\|\|_2}$
Text (T_{cos})	100	$1 - \frac{X_i^{text}.X_t^{text}}{\|\|X_i^{text}\|\|_2 \|\|X_t^{text}\|\|_2}$
Content+Style	8,192	$\alpha\ C_{cos} + \beta\ S_{cos}$
Content+Style+Text	8,292	$\alpha\ C_{cos} + \beta\ S_{cos} + \gamma\ T_{cos}$

The intuition behind above scenarios is that if a given embedding is a good representation, the *k-NN*s we retrieve must be from the same group as the query. Thus, for each scenario, we present ***precision@k*** and ***recall@k***, where $k \in \{5, 10, 15, 20\}$, as the performance metrics. ***Precision@k*** gives the percentage of relevant images among the retrieved images. ***Recall@k*** is the percentage of relevant retrieved images out of the all relevant images.

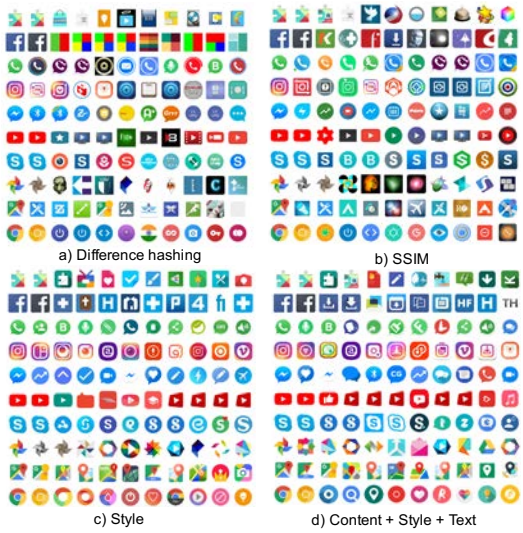


Figure 3: 10-Nearest neighbours of the top-10 popular apps

We present precision@k and recall@k values for all four test scenarios in Table 2 and Table 3. To choose the best β and γ values in multi-modals neural embeddings, we varied β and γ from 1 to 10 with an interval of one. We achieved the best results when $\beta = 5$ and $\gamma = 4$ and we report those results in Tables 2 and 3. The main takeaway messages from results in these two tables are:

- In all scenarios, neural embeddings outperform other methods. For example, for all four k-NN scenarios, the style embeddings have approximately 4%–14% and 11%–26% higher performance in precision@k and recall@k in all apps dataset.
- In UKBench and Holidays datasets, content, style, and combined embeddings increase *precision@k* and *recall@k* by 10%–15% and 12%–25%, respectively when retrieving five nearest neighbours. Combining style embeddings with content embeddings achieves 12% higher precision@k and 14% higher recall@k in all apps dataset compared to hashing and feature-based baselines when $k = 5$. Only scenario where combined content and style embeddings did not outperform all other methods is the UKBench dataset.
- It is also noticeable that adding text embedding further increases the performance by 3%-5% and 6%-7% in terms of precision@k and recall@k, respectively, compared to the best neural embedding method when $k \in \{5, 10\}$.
- Results also show that increasing the k value increases the recall@k, however, significantly decreases precision@k. The main reason is that average number of images per groups in all four datasets is less than 5 and thus the number of false positive images in the retrieved image set increases with k.

To elaborate further on the performance of the embeddings qualitatively, in Figure 3, we present the 10-nearest neighbours we retrieved using difference hashing, SSIM, style embeddings, and content+style+text embeddings for the top-10 most popular apps in Google Play Store. Figure 3-(a) shows that hashing methods do not identify visually similar apps apart from the first 1-2 similar apps (E.g. row 9 - Google Maps). Neural embeddings based methods in Figure 3-(c) and Figure 3-(d) have identified better fits in several cases (E.g. row 1 - Google Play Services and row 9 - Google Maps). Also, Figure 3-(c) shows that style embeddings have retrieved app icons that have the same *"look and feel"* in terms of colour.

5.2 Retrieving Potential Counterfeits

We next use the embeddings that performed best ($\text{Content}_{cos} + \beta\text{Style}_{cos} + \gamma\text{Text}_{cos}$ where $\beta = 5$ and $\gamma = 4$) to retrieve 10-nearest neighbours for ***top-10,000 apps*** from the corpus of 1.2 million apps that are not from the same developer. The 10-nearest neighbour search is forced to return 10 nearest apps, irrespective of the distance. As such, there can be cases where the nearest neighbour search returns apps that are very far from the query app. Thus, we applied a distance threshold to further narrow down the results. From the retrieved 10 results for each query app, we discarded the results that are having distances greater than an empirically decided threshold. The threshold was chosen as the knee-point [40] of the cumulative distribution of all the distances with the original apps. This process returned 60,638 unique apps that are potentially counterfeits of one or more apps with in top-10,000 popular apps. Out of this 60,638 we had APK files for 49,608 apps.

		Average	Difference	Perceptual	Wavelet	SIFT	SURF	LATCH	AKAZE	SSIM	C_{cos}	S_{cos}	$C_{cos} + \beta S_{cos}$	$C_{cos} + \beta S_{cos} + \gamma T_{cos}$
Holidays	**5-NN**	24.56	22.68	21.60	24.48	33.00	31.12	29.16	31.00	21.88	46.36	46.72	**47.92**	N/A
	10-NN	13.08	11.74	10.96	12.98	17.58	16.66	15.18	15.90	11.26	25.28	25.24	**25.92**	N/A
	15-NN	9.00	8.08	7.36	8.92	12.15	11.55	10.43	10.91	7.76	17.47	17.25	**17.89**	N/A
	20-NN	6.95	6.19	5.58	6.83	9.34	8.91	7.99	8.31	5.98	13.31	13.13	**13.57**	N/A
UKBench	**5-NN**	27.29	22.44	21.63	26.37	55.27	52.97	44.82	41.77	28.46	**70.22**	65.01	70.06	N/A
	10-NN	15.01	11.70	10.97	14.26	28.99	27.93	23.72	21.98	15.22	**36.90**	33.86	36.62	N/A
	15-NN	10.51	7.95	7.38	9.99	19.82	19.12	16.22	15.04	10.55	**25.03**	22.95	24.79	N/A
	20-NN	8.18	6.08	5.59	7.75	15.11	14.60	12.37	11.47	8.17	**18.97**	17.40	18.75	N/A
Labelled	**5-NN**	45.14	48.41	47.62	44.44	48.92	47.67	46.63	44.22	45.34	56.43	60.42	62.23	**64.76**
	10-NN	23.98	28.10	27.42	25.50	26.79	27.05	26.34	25.07	25.59	33.69	35.39	36.04	**38.47**
	15-NN	18.59	19.92	19.45	18.08	18.86	19.00	18.45	17.54	18.06	24.05	25.25	25.57	**27.19**
	20-NN	14.52	15.56	15.24	14.16	14.57	14.69	14.24	13.5	14.09	18.69	19.66	19.86	**21.09**
All	**5-NN**	34.89	38.01	37.07	34.17	38.23	39.13	37.32	36.87	37.39	45.51	50.72	50.91	**55.96**
	10-NN	19.43	21.53	20.79	19.08	21.82	22.10	21.09	20.81	20.73	26.08	29.57	29.81	**32.99**
	15-NN	13.69	15.30	14.74	13.32	15.31	15.52	14.82	14.63	14.47	18.30	20.90	21.12	**23.46**
	20-NN	10.63	11.89	11.40	10.36	11.87	11.97	11.46	11.33	11.15	14.07	16.14	16.31	**18.23**

Table 2: precision@k for all test scenarios (NN* - Nearest Neighbours)

		Average	Difference	Perceptual	Wavelet	SIFT	SURF	LATCH	AKAZE	SSIM	C_{cos}	S_{cos}	$C_{cos} + \beta S_{cos}$	$C_{cos} + \beta S_{cos} + \gamma T_{cos}$
Holidays	**5-NN**	41.18	38.03	36.22	41.05	55.33	52.18	48.89	51.98	36.69	77.73	78.34	**80.35**	N/A
	10-NN	43.86	39.37	36.75	43.53	58.95	55.87	50.91	53.32	37.76	84.78	84.64	**86.92**	N/A
	15-NN	45.27	40.64	37.02	44.87	61.10	58.08	52.45	54.86	39.03	87.86	86.79	**90.01**	N/A
	20-NN	46.61	41.52	37.42	45.81	62.64	59.76	53.59	55.73	40.11	89.27	88.06	**91.01**	N/A
UKBench	**5-NN**	34.11	28.05	27.04	32.96	69.09	66.22	56.03	52.23	35.58	**87.78**	81.27	87.58	N/A
	10-NN	37.51	29.25	27.42	35.66	72.47	69.84	59.3	54.96	38.03	**92.25**	84.65	91.54	N/A
	15-NN	39.41	29.82	27.69	37.46	74.34	71.69	60.83	56.4	39.56	**93.85**	86.08	92.96	N/A
	20-NN	40.92	30.82	27.93	38.73	75.57	72.99	61.83	57.34	40.86	**94.83**	86.98	93.75	N/A
Labelled	**5-NN**	51.40	55.35	54.22	50.61	55.43	54.28	53.09	50.35	51.60	64.26	68.80	69.82	**73.75**
	10-NN	59.17	64.08	62.44	58.07	61.00	61.60	59.99	57.11	58.24	76.72	80.59	82.09	**87.62**
	15-NN	63.49	68.04	66.46	61.77	64.42	64.91	63.04	59.93	61.66	82.17	86.27	87.34	**92.88**
	20-NN	66.12	70.90	69.40	64.48	66.37	66.91	64.88	61.51	64.12	85.14	89.55	90.45	**96.07**
All	**5-NN**	39.73	42.29	42.22	38.91	43.29	44.30	42.47	41.96	42.55	51.82	57.76	57.98	**63.72**
	10-NN	44.25	49.03	47.36	43.46	49.42	50.04	48.01	47.36	47.19	59.40	67.34	67.90	**75.13**
	15-NN	46.76	52.27	50.35	45.49	52.02	52.73	50.61	49.93	49.39	62.53	71.40	72.14	**80.16**
	20-NN	48.43	54.14	51.94	47.19	53.75	54.23	52.16	51.57	50.75	64.09	73.52	74.29	**83.05**

Table 3: recall@k for all test scenarios (NN* - Nearest Neighbours)

5.3 Malware Analysis

We then checked each of the 49,608 potential counterfeits using the private API of the online malware analysis tool *VirusTotal* [2]. VirusTotal scans the APKs with over 60 commercial anti-virus tools (AV-tools) and provides a report on how many of those tools identified whether the submitted APKs contain malware. In Figure 4, we show a summary of the number of apps that were tagged as possible malware by one or more AV-tools in VirusTotal and their availability in Google Play Store as of *24-10-2018*. There were 7,246 APKs that are tagged by at least one of the AV-tool.

However, a single AV-tool tagging an APK as malware may not mean that the APK contains malware. Thus, previous work used different thresholds for the number of AV-tools that must report to consider an APK as malware. Ikram et al. [23] used a conservative threshold of 5 and Arp et al. [9] used a relaxed threshold of 2. Figure 4 shows that we have 3,907 apps if the AV-tool threshold is 2 and 2,040 apps if the threshold is 5, out of which 2,067 and 1,080 apps respectively, are still there in Google Play Store. ~46% of the apps (3,358) that were tagged by at least one AV-tool are currently not available in Google Play Store. One possible reason is that these apps were removed by Google after binary analysis. In Table 4, we show some example apps that were tagged as containing malware.

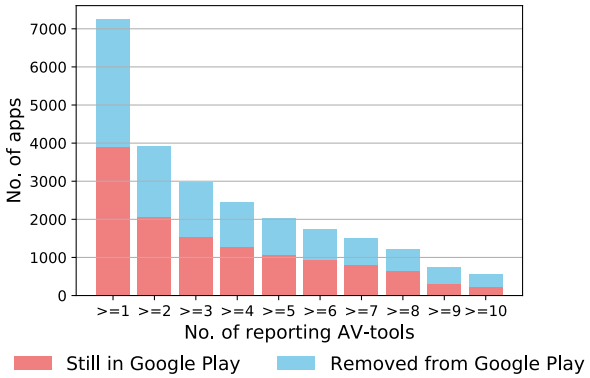
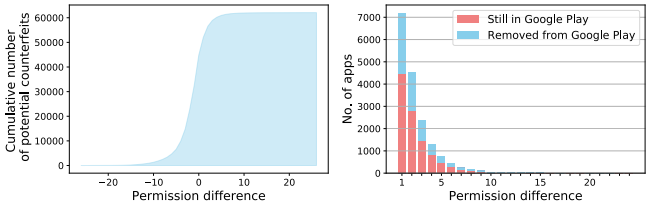


Figure 4: No. of apps against the no. of reporting AV-tools

Table 4: Example similar apps that contain malware

Original app	Similar app	AV-tools	Downloads (Original)	Downloads (Similar)
Clean Master	Ram Booster*	12	500 million - 1 billion	500 - 1,000
Temple Run	Endless Run*	12	100 million - 500 million	5,000 - 10,000
Temple Run 2	Temple Theft Run*	12	500 million - 1 billion	500, 000 - 1 million
Hill Climb Racing	Offroad Racing: Mountain Climb	9	100 million - 500 million	1 million - 5 million
Parallel Space	Double Account*	17	50 million - 100 million	100, 000 - 500, 000

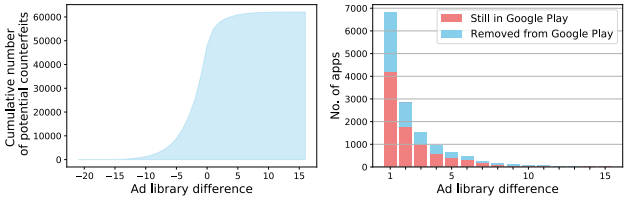
*The app is currently not available in Google Play Store



(a) Cumulative number of apps against *permission difference*

(b) Play Store availability of apps with positive *permission difference*

Figure 5: Counterfeits requesting extra permissions



(a) Cumulative number of apps against *ad library difference*

(b) Play Store availability of apps with positive *ad library difference*

Figure 6: Counterfeits with additional ad libraries

5.4 Permission Requests

Another motivation behind counterfeiting can be collecting personal data. We considered the 26 dangerous Android permissions [3] and to identify the potential counterfeits that ask for more permissions than the original app, we define a metric, ***permissions difference***, which is the difference between the number of dangerous permissions requested by the potential counterfeit but not the original app and number of dangerous permissions requested by the original app but not by the potential counterfeit app. If the ***permissions difference*** is a positive value that means the counterfeit asks for more dangerous permissions than the original app and vice versa if it is negative. For the 49,608 potential counterfeits we had the APK files, we calculated the *permission difference*.

The cumulative sum of number of apps against the *permission difference* is shown in Figure 5a. The majority of the potential counterfeits did not ask for more dangerous permissions than the original app. However, there are 17,230 potential counterfeits that are asking at least one dangerous permission than the original app (13,857 unique apps), and 1,866 potential counterfeits (1,565 unique apps) asking at least five additional dangerous permissions. In Figure 5b we show Google Play Store availability of the 17,230 apps with positive ***permissions difference*** as of *24-10-2018*. Figure shows approximately 37% of the potential counterfeits with a permission difference of five is currently not available in the Google Play Store.

5.5 Advertisement Libraries

Another motivation behind developing counterfeits can be monetisation using advertisements. To quantify this, we defined; ***ad library difference*** using the list of 124 mobile advertising and analytics libraries provided in [41]. ***Ad library difference*** is the difference between the number of advertisement libraries embedded in the potential counterfeit but not in the original app and number of advertisement libraries embedded in the original app but not in the potential counterfeit app. We show the cumulative number of counterfeits over the range of ***ad library difference*** in Figure 6a. According to the figure, 13,997 apps (11,281 unique apps) have a positive ad library difference and out of that 1,841 (1,407 unique apps) have an ad library difference greater than or equal to five. Figure 6b shows the Google Play store availability of apps with a positive ad library difference. Approximately 33% of the apps we identified are currently not available in the Google Play Store.

6 CONCLUSION

We proposed an icon encoding method that allows to efficiently search potential counterfeits to a given app, using neural embeddings generated by CNNs. Specifically, for app counterfeit detection problem, we showed that content and style neural embeddings generated from a pre-trained VGGNet significantly outperforms hashing and feature-based image retrieval methods.

We used our multi-modal embedding method to retrieve potential counterfeits for the top-10,000 apps in Google Play and investigated the possible inclusion of malware, permission usage, and embedded third party ad libraries. We found that 2,040 potential counterfeits we retrieved were marked by at least five commercial antivirus tools as malware, 1,565 asked for at least five additional dangerous permissions, and 1,407 had at least five additional embedded third party ad libraries. Finally, we showed that as of now (6-10 months since we discovered the apps), 27%–46% of the potential counterfeits we identified are not available in Google Play Store, potentially removed due to customer complaints.

ACKNOWLEDGEMENT

This project is partially funded by the Google Faculty Rewards 2017, NSW Cyber Security Network's Pilot Grant Program 2018, and the Next Generation Technologies Program. Authors would

like to thank VirusTotal for kindly providing access to the private API, which was used for the malware analysis in this paper.

REFERENCES

[1] 2018. https://github.com/matlink/gplaycl.
[2] 2018. https://www.virustotal.com.
[3] 2018. https://developer.android.com/guide/topics/permissions.
[4] Charu C Aggarwal, Alexander Hinneburg, and Daniel A Keim. 2001. On the surprising behavior of distance metrics in high dimensional spaces. In *ICDT*. Springer.
[5] Fawad Ahmed and M. Y. Siyal. 2006. A Secure and Robust Wavelet-Based Hashing Scheme for Image Authentication. In *Advances in Multimedia Modeling*. 51–62.
[6] Pablo Fernández Alcantarilla, Jesús Nuevo, and Adrien Bartoli. 2013. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. In *BMVC*. 1–9.
[7] Benjamin Andow, Adwait Nadkarni, Blake Bassett, William Enck, and Tao Xie. 2016. A study of grayware on Google Play. In *Security and Privacy Workshops (SPW), 2016 IEEE*. IEEE.
[8] Ionut Arghire. 2017. Fake Netflix App Takes Control of Android Devices. http://www.securityweek.com/fake-netflix-app-takes-control-android-devices.
[9] Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, Konrad Rieck, and CERT Siemens. 2014. DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket.. In *NDSS*.
[10] Artem Babenko, Anton Slesarev, Alexander Chigorin, and Victor S. Lempitsky. 2014. Neural Codes for Image Retrieval. *CoRR* abs/1404.1777 (2014). arXiv:1404.1777
[11] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. 2006. SURF: Speeded Up Robust Features. In *Computer Vision-ECCV*. Springer Berlin Heidelberg, 404–417.
[12] Sean Bell and Kavita Bala. 2015. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (TOG)* (2015).
[13] Sean Bell and Kavita Bala. 2015. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (TOG)* (2015).
[14] Iker Burguera, Urko Zurutuza, and Simin Nadjm-Tehrani. 2011. Crowdroid: Behavior-based malware detection system for android. In *Proc. of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*. ACM, 15–26.
[15] Rishi Chandy and Haijie Gu. 2012. Identifying spam in the iOS app store. In *Proc. of the 2nd Joint WICOW/AIRWeb Workshop on Web Quality*. ACM, 56–59.
[16] Sam Costello. 2018. How Many Apps Are in the App Store? https://www.lifewire.com/how-many-apps-in-app-store-2000252. Accessed: 2018-04-12.
[17] Jonathan Crussell, Clint Gibler, and Hao Chen. 2013. Andarwin: Scalable detection of semantically similar Android applications. In *European Symposium on Research in Computer Security*. Springer, 182–199.
[18] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2015. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576* (2015).
[19] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2015. Texture synthesis and the controlled generation of natural stimuli using convolutional neural networks. *CoRR* abs/1505.07376 (2015). arXiv:1505.07376
[20] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using Convolutional Neural Networks. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*.
[21] Clint Gibler, Ryan Stevens, Jonathan Crussell, Hao Chen, Hui Zang, and Heesook Choi. 2013. Adrob: Examining the landscape and impact of Android application plagiarism. In *Proc. of the 11th MobiSys*. ACM.
[22] Michael Grace, Yajin Zhou, Qiang Zhang, Shihong Zou, and Xuxian Jiang. 2012. Riskranker: Scalable and accurate zero-day Android malware detection. In *Proc. of the 10th international conference on Mobile systems, applications, and services*. ACM, 281–294.
[23] Muhammad Ikram, Narseo Vallina-Rodriguez, Suranga Seneviratne, Mohamed Ali Kaafar, and Vern Paxson. 2016. An Analysis of the Privacy and Security Risks of Android VPN Permission-enabled Apps. In *Proc. of the 2016 ACM on Internet Measurement Conference*.
[24] Statista Inc. 2018. Number of available applications in the Google Play Store from December 2009 to December 2017. https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/.
[25] Chris Jager. 2018. Scam Alert: Fake CBA And ANZ Bank Apps Discovered On Google Play Store. https://www.lifehacker.com.au/2018/09/scam-alert-fake-cba-and-anz-banking-apps-found-on-google-play-store/. Accessed: 2018-10-15.
[26] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2008. Hamming embedding and weak geometric consistency for large scale image search. In *European conference on computer vision*. Springer, 304–317.
[27] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, and Mingli Song. 2017. Neural Style Transfer: A Review. *arXiv preprint arXiv:1705.04058* (2017).
[28] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*. Springer, 694–711.
[29] Neal Krawetz. 2013. http://www.hackerfactor.com/blog/?/archives/432-Looks-Like-It.html.
[30] Neal Krawetz. 2013. http://www.hackerfactor.com/blog/?/archives/529-Kind-of-Like-That.html.
[31] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proc. of the 31st ICML*. 1188–1196.
[32] G. Levi and T. Hassner. 2016. LATCH: Learned arrangements of three patch codes. In *IEEE Winter Conference on Applications of Computer Vision*. 1–9.
[33] Ping Li, Trevor J Hastie, and Kenneth W Church. 2006. Very sparse random projections. In *Proc. of the 12th ACM SIGKDD*. ACM, 287–296.
[34] David G Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International journal of computer vision* 60, 2 (2004).
[35] Luka Malisa, Kari Kostiainen, and Srdjan Capkun. 2017. Detecting Mobile Application Spoofing Attacks by Leveraging User Visual Similarity Perception. In *Proc. of the Seventh ACM on Conference on Data and Application Security and Privacy (CODASPY '17)*. ACM, New York, NY, USA, 289–300. https://doi.org/10.1145/3029806.3029819
[36] Luka Malisa, Kari Kostiainen, Michael Och, and Srdjan Capkun. 2016. Mobile application impersonation detection using dynamic user interface extraction. In *European Symposium on Research in Computer Security*. Springer, 217–237.
[37] Shin Matsuo and Keiji Yanai. 2016. CNN-based style vector for style image retrieval. In *Proc. of the 2016 ACM ICMR*. ACM, 309–312.
[38] David Nister and Henrik Stewenius. 2006. Scalable Recognition with a Vocabulary Tree. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.
[39] Sarah Perez. 2013. Developer Spams Google Play With Ripoffs Of Well-Known Apps Again. http://techcrunch.com.
[40] Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. 2011. Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In *Distributed Computing Systems Workshops (ICDCSW), 2011 31st International Conference on*. IEEE.
[41] Suranga Seneviratne, Harini Kolamunna, and Aruna Seneviratne. 2015. A measurement study of tracking in paid mobile applications. In *Proc. of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. ACM, 7.
[42] Suranga Seneviratne, Aruna Seneviratne, Mohamed Ali Kaafar, Anirban Mahanti, and Prasant Mohapatra. 2015. Early detection of spam mobile apps. In *Proc. of the 24th International Conference on World Wide Web*.
[43] Suranga Seneviratne, Aruna Seneviratne, Mohamed Ali Kaafar, Anirban Mahanti, and Prasant Mohapatra. 2017. Spam Mobile Apps: Characteristics, Detection, and in the Wild Analysis. In *To Appear in Proc. of Transactions on the Web (TWEB)*. ACM.
[44] Asaf Shabtai, Uri Kanonov, Yuval Elovici, Chanan Glezer, and Yael Weiss. 2012. Andromaly: A behavioral malware detection framework for android devices. *Journal of Intelligent Information Systems* 38, 1 (2012), 161–190.
[45] Karen Simonyan and Andrew Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* abs/1409.1556 (2014). arXiv:1409.1556
[46] Mingshen Sun, Mengmeng Li, and John Lui. 2015. DroidEagle: Seamless detection of visually similar Android apps. In *Proc. of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. ACM.
[47] Didi Surian, Suranga Seneviratne, Aruna Seneviratne, and Sanjay Chawla. 2017. App Miscategorization Detection: A Case Study on Google Play. *IEEE TKDE* 29, 8 (2017).
[48] Wei Ren Tan, Chee Seng Chan, Hernán E Aguirre, and Kiyoshi Tanaka. 2016. Ceci n'est pas une pipe: A deep convolutional network for fine-art paintings classification. In *Image Processing (ICIP), 2016 IEEE International Conference on*. IEEE.
[49] Nicolas Viennot, Edward Garcia, and Jason Nieh. 2014. A measurement study of Google Play. In *ACM SIGMETRICS Performance Evaluation Review*. ACM.
[50] Kyle Wagner. 2012. Fake Angry Birds Space Android App Is Full Of Malware. https://www.gizmodo.com.au/2012/04/psa-fake-angry-birds-space-android-app-is-full-of-malware/.
[51] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* 13, 4 (2004), 600–612.
[52] Dong-Jie Wu, Ching-Hao Mao, Te-En Wei, Hahn-Ming Lee, and Kuo-Ping Wu. 2012. Droidmat: Android malware detection through manifest and api calls tracing. In *Information Security (Asia JCIS), 2012 Seventh Asia Joint Conference on*. IEEE, 62–69.
[53] Zhen Xie and Sencun Zhu. 2015. AppWatcher: Unveiling the underground market of trading mobile app reviews. In *Proc. of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. ACM.
[54] Zhenlong Yuan, Yongqiang Lu, Zhaoguo Wang, and Yibo Xue. 2014. Droid-Sec: Deep learning in Android malware detection. In *ACM SIGCOMM Computer Communication Review*.
[55] H. Zhang, M. Schmucker, and X. Niu. 2007. The Design and Application of PHABS: A Novel Benchmark Platform for Perceptual Hashing Algorithms. In *IEEE International Conference on Multimedia and Expo*. 887–890.
[56] Yajin Zhou and Xuxian Jiang. 2012. Dissecting Android malware: Characterization and evolution. In *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE.

Context-Aware Sequential Recommendations with Stacked Recurrent Neural Networks

Lakshmanan Rakkappan
Department of Computer Science
School of Computing, National University of Singapore
e0223207@u.nus.edu

Vaibhav Rajan
Department of Information Systems and Analytics
School of Computing, National University of Singapore
vaibhav.rajan@nus.edu.sg

ABSTRACT

Sequential history of user interactions as well as the context of interactions provide valuable information to recommender systems, for modeling user behavior. Modeling both contexts and sequential information simultaneously, in context-aware sequential recommenders, has been shown to outperform methods that model either one of the two aspects. In long sequential histories, temporal trends are also found within sequences of contexts and temporal gaps that are not modeled by previous methods. In this paper we design new context-aware sequential recommendation methods, based on Stacked Recurrent Neural Networks, that model the dynamics of contexts and temporal gaps. Experiments on two large benchmark datasets demonstrate the advantages of modeling the evolution of contexts and temporal gaps – our models significantly outperform state-of-the-art context-aware sequential recommender systems.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; • **Information systems** → *Content ranking*; Collaborative filtering.

KEYWORDS

Sequential Recommendation; Context-Aware Recommendation; Stacked Recurrent Neural Network

ACM Reference Format:
Lakshmanan Rakkappan and Vaibhav Rajan. 2019. Context-Aware Sequential Recommendations with Stacked Recurrent Neural Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313567

1 INTRODUCTION

In a world of overwhelming number of choices, recommender systems have become indispensable as personalized information filters, especially for e-commerce. Developing accurate recommendation algorithms poses several challenges [1] and continues to be of great interest to both industry and academia. Among the large number of algorithms developed, arguably, the most well known are Collaborative Filtering methods that factorize the historical user-item preference matrix to obtain latent user representations, that, in turn, are used to predict future behaviour.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313567

Sequential history of users provide valuable clues to predict future behaviour. Recommender systems based on Markov chains [2] and combinations of matrix factorization and Markov chains [15] leverage such historical sequential behavior. But, methods based on Markov chains only capture local sequential patterns, and Recurrent Neural Networks (RNN) are found to be more effective in capturing global sequential history [24] thereby yielding better recommendations. However, these methods do not explicitly model the dependencies on the temporal gaps (also called *temporal context* in previous literature) between events.

Additional *input context* information, such as location, time of day, and local holiday information, include another dimension of information, that is often available (e.g., through smartphone apps). Context-aware recommender systems, effectively use such information and have been extensively studied [16, 18]. Although most of the previous literature has studied them separately, contexts and sequential information are not independent. For example, gap statistics between purchases can differ with location and other contextual variables. Liu *et al.* demonstrate the advantages of simultaneous modeling of context and sequential history over modeling either of them separately through their method Context-Aware RNN [10], that also explicitly models the temporal context. However, their model requires the number of possible input and temporal contexts (through discretizing time intervals into specific bins) to be pre-specified. A separate parameter matrix is learnt for each user-specified context, during training. Thus, the number of model parameters grow with the number of input and temporal contexts, making the task of training increasingly more difficult.

With long sequential histories of users, we observe that temporal trends are found even within changing input and temporal contexts, that, to our knowledge, has not been modeled by previous recommendation methods. We design new context-aware sequential recommender methods using stacked RNNs that model the dynamics of input and temporal contexts. Thus, instead of modeling separate contexts with different parameter matrices, we model how the context evolves with a single recurrent parameter matrix. This also effectively addresses the limitation of previous models: (1) different possible temporal contexts need not be specified in advance during training and (2) number of temporal context parameters do not increase with increasing number of contexts. To summarize, our main contributions in this paper are:

- We design new context-aware sequential recommendation methods:
  - that model the dynamics of input and temporal contexts, that has not been modeled in previous methods.
  - where model parameters do not depend on the number of temporal contexts; in the best previous models, the

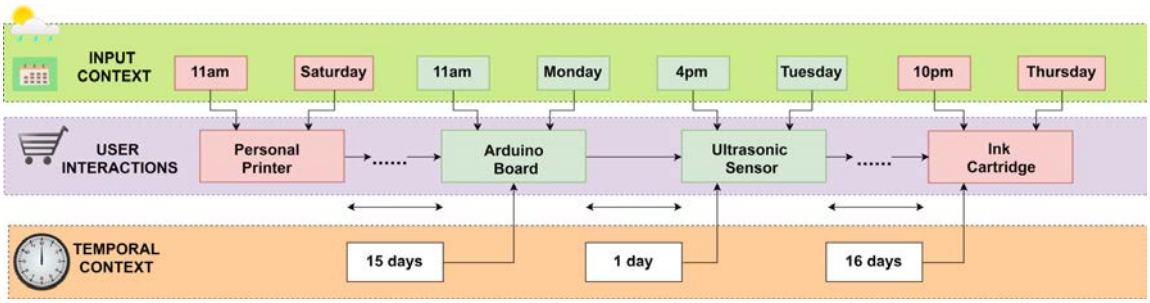


Figure 1: Schematic of user interactions with input and temporal contexts

number of parameters increase with increasing number of temporal and input contexts, thereby making training increasingly harder.

- Our experiments on two large benchmark datasets show that our models significantly outperform state-of-the-art context-aware sequential recommendation methods.

2 PROBLEM DEFINITION

Let u denote a user. Let $X = \{x_1^u, x_2^u, ...\}$ denote the sequence of items the user u has interacted with: x_t^u denotes the item that the user u has interacted with at timestep t. Interaction could be purchase, click, view or any other activity relevant to the application. With each interaction x_t^u of a user, we associate an **input context**, c_t^u, and a **temporal context**, z_t^u. Input context can represent physical or social context. For s types of input contexts, the input context c_t^u is represented by an se-dimensional embedding. We have Σ such embeddings for each possible value of an input context. Temporal context represents the interval between consecutive interactions, i.e., z_t^u refers to the time interval between interactions x_t^u and $x_{(t-1)}^u$. It is a scalar, in a suitable unit (like days) that can be application dependent. Figure 1 shows a schematic of user interactions and both contexts. The sequential recommendation task is to predict $x_{(t+1)}^u$ for a user u, given historical data consisting of past interactions $\{x_1^u, \ldots, x_t^u\}$ together with their input contexts $\{c_1^u, \ldots, c_t^u\}$ and temporal contexts $\{z_1^u, \ldots, z_t^u\}$.

3 RELATED WORK

Sequential Recommendation. Sequential (or next basket) recommendation algorithms based on Markov chains (e.g., [2]) use features derived from past transactions to predict a user's next interaction. These methods typically estimate a transition matrix from past data of all users, which in turn is used to predict the probability of the next interaction. Factorizing Personalized Markov Chains (FPMC) [15] combine the strengths of Markov chains and matrix factorization. In FPMC, a transition cube is estimated, the added third dimension for personalizing user-specific transitions. This transition tensor is factorized to learn latent factors and enables information propagation among similar users, items and transitions. Since the operations used are linear, FPMC cannot capture interactions among multiple factors because each component influences a user's next interaction independently [22]. The Hierarchical Representation Model (HRM) addresses this problem by summarizing multiple interacting factors through a nonlinear max pooling operation [22]. HRM uses continuous-valued representations of users and items and builds a hybrid representation over users and items from the previous interaction. Both FPMC and HRM model only local interactions between consecutive transactions.

There may be correlations between items that are not adjacent in a user's interaction history, e.g., many users may purchase an ink cartridge after buying a printer, but the time interval between the purchases may differ across users, depending on their usage differences. Further, the purchases may not be in consecutive baskets. The Dynamic REcurrent bAsket Model (DREAM) uses a pooling operation, similar to that of HRM, but uses Recurrent neural networks (RNN) to model global sequential interactions[24]. The hidden layer within the RNN is used as the dynamic user representation which is used to compute a per-item score at a time-step, with a higher score corresponding to higher likelihood of interaction. The Bayesian Pairwise Ranking (BPR) objective function [14] is optimized using Back Propagation Through Time (BPTT) [17]. DREAM significantly outperforms HRM and FPMC demonstrating the effectiveness of RNNs in mining global sequential interactions.

Context-Aware Recommendation. Contextual information such as weather, location, holidays etc. can provide significant performance improvement to personalized recommendations [12]. For instance, users may buy some items before a specific social occasion that they may not purchase often. Context-aware recommendations, broadly, use three types of strategies. In pre-filtering strategies, data may be filtered before applying collaborative filtering. In post-filtering strategies, collaborative filtering is applied first, ignoring contextual information, and then the recommendations are adjusted taking the context into account [1]. Thirdly, the context can directly be taken into account in the recommendation model using generalizations of matrix factorization, such as collective matrix factorization [19] and tensor factorizations [8], or through the use of factorization machines FM [16]. Treating the context in a way similar to users or items, cannot model the semantic operations of contexts. Both CARS2 [18] and COT [11] address this problem through the use of context-specific latent representations. Recent related work include the use of attention-based neural models in predicting next item within a transactional context [23]. However, these methods do not model sequential dependencies in the user history.

Sequential Context-Aware Recommendation. To model both contextual information and sequential dependencies, the Context-Aware

Recurrent Neural Network (CARNN) model was proposed [10]. They use an RNN based dynamic representation of users (similar to that of DREAM). Modeling both contextual and sequential information yields significant performance improvements over both sequential recommendation methods (like DREAM, FPMC and HRM) and context-aware recommendation methods (like CARS2 and FM).

Other Related Work. Collaborative nowcasting for 'intent monitoring' also models contexts and sequential evolution of user representations [20, 21]. However they do not model sequential evolution of contexts. Moreover, their models, based on Kalman filters, can only capture linear dynamics of user representations that are not as flexible as RNNs that can capture non-linear dynamics. A Bayesian Hidden Markov Model for modeling user contexts was proposed in [7] where sequential dependencies in contexts were modeled and effectively utilized for personalized context recognition. They do not evaluate their model for sequential recommendations. Further, their model is semi-supervised and models only linear dynamics. The method CDUE integrates vector autoregressive and dynamic topic modeling of contents information and matrix factorization [4]. They learn users' interests and items' topics and their evolutions collaboratively and simultaneously. But their problem setting is different: they assume access to the user's browsing history (that can be considered a specific type of contextual information), through which topic evolution is learnt, to predict a future item.

4 CONTEXT-AWARE SEQUENTIAL RECOMMENDATION

4.1 Background

Recurrent Neural Networks (RNN) effectively model sequential user interactions by learning a recurrent hidden user representation h_t, from the user's current interaction x_t^u, at each time step t:

$$h_t = g(x_t^u M + h_{t-1} W) \tag{1}$$

where g is a non-linear activation function (e.g. sigmoid). Transition matrix M models user behavior with respect to the current interaction and weight matrix W models the recurrent connection that propagates signal between consecutive hidden states. Both M, W are learnt from historical data. In the following we call this RNN, the user RNN or *URNN*.

CARNN [10] models contextual effects through input and temporal context matrices within URNN. Specifically, the hidden representation of a user, h_t, at timestep t is given by:

$$h_t = g(x_t^u M_{c_I} + h_{t-1} W_{z_T})$$

where M_{c_I} and W_{z_T} are context-specific weight matrices for input contexts (c_I) and temporal contexts (z_T) respectively. In CARNN, the number of input and temporal context matrices (M_{c_I} and W_{z_T}) have to be pre-specified during training. For instance, the day of the week and hour of the day can be used as contexts yielding a total of $7 \times 24 = 168$ different contexts (requiring up to 168 parameter matrices). Also, since it is impossible to learn transition context weight matrix (W_{z_T}) for every possible time interval, the intervals are discretized into chosen bins, and a matrix is learnt for each such discretized interval. For long sequences, the number of temporal contexts can be very large. They could be reduced by choosing larger bins but that may not be able to capture the effects of different temporal contexts. Liu, Wu and Wang [11] also recognize the problem of growing number of context-specific parameters with increase in number of contexts and address the problem in their context-aware (but not sequential) recommendation model.

4.2 Our Approach

We model the dynamics of temporal and input contexts and their effects on hidden representations of users. The evolution of temporal contexts (z_t^u) is modeled through an RNN (that we call *TCRNN*):

$$p_t = g(z_t^u E + p_{t-1} Q),$$

and the evolution of input contexts (c_t^u) is modeled through another RNN (that we call *ICRNN*):

$$v_t = g(c_t^u A + v_{t-1} K),$$

in addition to URNN (equation 1) that models the evolution of user representation. In URNN, since matrix W propagates information between consecutive time steps, we replace W with a signal from TCRNN to incorporate the effects of evolving temporal contexts. Similarly we replace M with a signal from ICRNN to model the effects of evolving input contexts. The details of how these signals are effected are given below.

Thus we have two (or more) interacting RNNs where the evolving hidden representation of the user h_t is also affected by the recurrent hidden context representations. Our models can be seen as variants of stacked RNNs [3, 13]. Stacked RNNs are constructed by stacking multiple recurrent hidden layers, with the goal of capturing representations at different timescales in each recurrent level. However, in our models, each RNN models a distinct temporal process with different inputs at each time step. Thus, they can also be viewed as multiple interacting RNNs.

We describe three models – STAR, SIAR and SITAR. STAR and SIAR model evolving temporal and input contexts respectively, and SITAR models the evolution of both temporal and input contexts.

STAR: Evolving Temporal Contexts

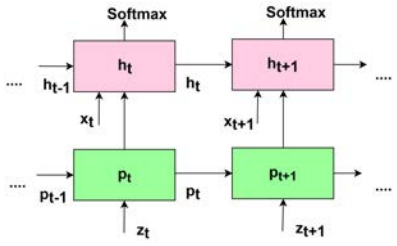


Figure 2: Stacked Temporal-context Aware RNN (STAR)

A schematic of our model Stacked Temporal-context Aware RNN (STAR) is shown in figure 2. Each block is an RNN cell at a single timestep (t). The RNN shown above in figure 2 models the user's evolving representation (h_t) and is dependent on not just the current interaction (x_t^u) and previous user representation (h_{t-1}) but also the hidden representation (p_t) of the evolving temporal context (z_t^u) modeled by the lower TCRNN.

In TCRNN, the recurrent connection weight matrix Q propagates sequential signals between consecutive hidden representations p_t and p_{t-1}, and E is a transition matrix between the temporal context and its hidden representation. We use an intermediate feedforward layer to transform p_t to J_t – to maintain a lower dimension of d in the TCRNN's hidden representation p_t and transform to a higher dimension of $d \times d$ in J_t that is used, instead of W, in URNN. The dynamic d-dimensional hidden user representation h_t is given by:

$$\begin{aligned} p_t &= g(z_t^u E + p_{t-1} Q) \\ J_t &= g(p_t L) \\ h_t &= g(x_t^u M + h_{t-1} J_t) \end{aligned}$$

where g denotes the activation function. Matrix M plays the role of a transition matrix between the user interaction (x_t^u) and user representation (h_t). The transformed temporal context representation J_t (reshaped from $1 \times d^2$ to $d \times d$) is used to propagate the signal between h_{t-1} and h_t. Weight matrices E, Q, L and M are of dimensions $1 \times d$, $d \times d$, $d \times d^2$ and $d \times d$ respectively, and are learnt during training. The key difference between URNN (equation 1) and STAR is that the weight matrix W that propagates information between consecutive time steps in URNN, is replaced by J_t from TCRNN to incorporate the effects of evolving temporal contexts.

SIAR: Evolving Input Contexts

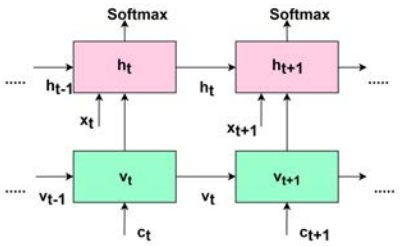


Figure 3: **Stacked Input-context Aware RNN (SIAR)**

A schematic of our model Stacked Input-context Aware RNN (SIAR) is shown in figure 3. Each block is an RNN cell at a single timestep (t). The RNN shown above in figure 3 models the user's evolving representation (h_t) and is dependent on not just the current interaction (x_t^u) and previous user representation (h_{t-1}) but also the hidden representation (v_t) of the evolving input context (c_t^u) modeled by the lower ICRNN. In ICRNN, the recurrent connection weight matrix K propagates sequential signals between consecutive hidden representations v_t and v_{t-1}, and A is a transition matrix between the input context and its hidden representation. Just like in the case of STAR, we use an intermediate feedforward layer to transform v_t to I_t. This is done to maintain a lower dimension of d in the ICRNN's hidden representation v_t and transform to a higher dimension of $d \times d$ in I_t that is used, instead of M, in URNN. The dynamic d-dimensional hidden user representation h_t is given by:

$$\begin{aligned} v_t &= g(c_t^u A + v_{t-1} K) \\ I_t &= g(v_t F) \\ h_t &= g(x_t^u I_t + h_{t-1} W) \end{aligned}$$

where g denotes the activation function. The matrix W is learnt to propagate the signal between h_{t-1} and h_t. The transformed input context representation I_t (reshaped from $1 \times d^2$ to $d \times d$) is used to model the transition matrix between the user interaction (x_t^u) and user representation (h_t). A, K, F and W are weight matrices of dimensions $se \times d$, $d \times d$, $d \times d^2$ and $d \times d$ respectively that are learnt during training. For s types of input contexts, the input context c_t^u is represented by an se-dimensional embedding. The main difference between URNN (equation 1) and SIAR is that the weight matrix M that models the effect of current interaction in URNN, is replaced by I_t from ICRNN to incorporate the effects of evolving input contexts.

SITAR: Evolving Temporal and Input Contexts

Our model, Stacked Input-context and Temporal-context Aware RNN (SITAR), models both evolving temporal and input contexts through TCRNN and ICRNN respectively and signals from both the RNNs are used to model the recurrent user representation as shown in figure 4.

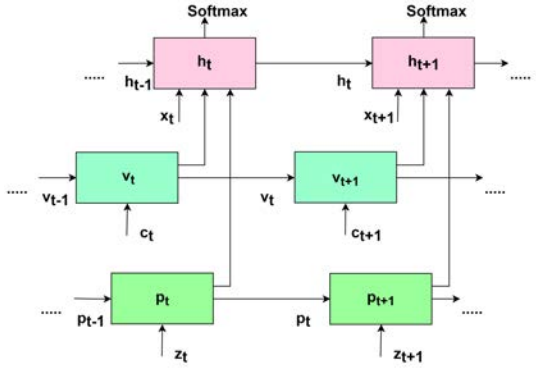


Figure 4: **Stacked Input-context and Temporal-context Aware RNN (SITAR)**

The user's evolving representation (h_t) is thus dependent on the current interaction (x_t^u), the previous user representation (h_{t-1}) as well as the hidden representations (p_t, v_t) of the evolving input context (c_t^u) and temporal context (z_t^u). The dynamic d-dimensional hidden user representation h_t is given by:

$$\begin{aligned} v_t &= g(c_t^u A + v_{t-1} K) \\ I_t &= g(v_t F) \\ p_t &= g(z_t^u E + p_{t-1} Q) \\ J_t &= g(p_t L) \\ h_t &= g(x_t^u I_t + h_{t-1} J_t) \end{aligned}$$

where weight matrices A, K, F, E, Q, L (as defined in models STAR and SIAR earlier) are learnt during training. Comparing this model with URNN (equation 1) and CARNN, we can view this model as using time-dependent parameters for both input and temporal contexts (through I_t and J_t), that implicitly model different possible input and temporal contexts through their temporal evolution, instead of explicitly modeling different possible contexts.

4.3 Training and Hyperparameter Settings

Backpropagation Through Time (BPTT) was used to learn the model parameters in all our models. We used the Adam optimizer [9] for training our models with a learning rate of 0.001, $\beta_1, \beta_2, \epsilon$ values set to 0.9, 0.999 and 10^{-8} respectively. We used cross-entropy loss with L2 regularization, sigmoid activation function (g), and random initialization. The source code for our models and all experimental results are available on our public repository.[1]

Number of Context-dependent Parameters. For s input context types, let Σ be the sum of possible values and Π be the number of combinations of values of the input contexts. E.g., with weekday and hour as contexts, $s = 2, \Sigma = 7 + 24 = 31, \Pi = 7 \times 24 = 168$. With p temporal contexts, d-dimensional hidden representation, and e-dimensional input embedding, the number of context-dependent parameters to be learnt in our models and CARNN are listed in table 1.

Table 1: Number of context-dependent parameters

Model:	STAR	SIAR	SITAR	CARNN
# Parameters:	0	$sed + e\Sigma$	$sed + e\Sigma$	$(\Pi + p)d^2$

Both d, e are fixed numbers that do not grow with increasing number of contexts. In our experiments, the models perform well at $e = d = 20$ and the performance does not significantly change at higher values of d (figures 5 and 6). However, the number of temporal contexts (p) can potentially be very large in long sequential user histories. The number of parameters in SIAR and SITAR are independent of the number of temporal contexts while in STAR, they are independent of both input and temporal contexts. In contrast, number of parameters in CARNN increase with both contexts.

Time Complexity. The per-timestep training time complexity is $O(d^3)$ for STAR and $O(d^3 + sed)$ for SIAR and SITAR.

4.4 Prediction

In our models, we use a softmax layer that takes the hidden representation of the user h_t at each timestep to generate a probability vector for all the items. The user representation is transformed to a vector m of size N through a linear transformation, $m = h_t B$, where N is the total number of items, to which probabilities are assigned at each time step, and B is learnt during training. The probability of the next item j is obtained through softmax: $\frac{e^{m_j}}{\sum_k e^{m_k}}$. The most probable item is used as the prediction. In our experiments, BPR [14] (instead of softmax) did not improve results.

5 EXPERIMENTS

5.1 Datasets

We used two large benchmark datasets:
1. **Movielens-1M** data[2]: that contains 1,000,209 movie ratings of 3,900 movies that has been collected from 6,040 users [5].
2. **Amazon Books** data[3]: that has 22,507,155 user ratings of 2,330,066 books from 8,026,324 users that was collected between May 1996 to July 2014 [6].

[1] https://bitbucket.org/cdal/stackedcontextawarernn
[2] http://grouplens.org/datasets/movielens/1m/
[3] http://jmcauley.ucsd.edu/data/amazon/

In each dataset, we utilised the first 80% of each user's history for training and the remaining 20% as the test set. We did not use data of users and items with very few interactions to avoid data sparsity: in the Movielens dataset, we selected users with at least 10 records and items with at least 3 records. In the Amazon dataset, we chose users and items with at least 25 records each. We extracted both input and temporal contextual information from the timestamps of ratings in both datasets. For temporal contexts, we measured time intervals in days using upward rounding. All intervals greater than 30 days were mapped to a single value. For input contexts, we only used the day of the week and the hour of the day to have a fair comparison with our main baseline CARNN (that used the same contexts), to have 7 day-based contexts and 24 hour-based contexts.

5.2 Baselines

CARNN [10] is our main baseline since it is a state-of-the-art method that models both contextual information and sequential dependencies. It was shown to outperform both context-aware recommenders (CARS2 and FM) and sequential recommenders (DREAM, FPMC and HRM) in their experiments. In addition, we use the RNN-based model **DREAM** [24] and Markov chain based model **FPMC** [15] as baselines, to compare with competitive sequential recommender models. The parameter transaction length is set to 1 week in both these methods. Methods that take neither context nor sequential information into account, **BPR** [14] (using matrix factorization) and **POP**, are used as additional baselines. Publicly available code provided by the authors were used for these baselines, except for POP, where items are ranked by frequency and the most frequent item is recommended.

5.3 Evaluation Metrics

For each user we predict the final 20% of the interactions. We denote this list of items, concatenated over all users, by Y and Y_q denotes the item in the q^{th} position in this list. At each step, our models output a list of probabilities (through softmax). Let $\hat{Y}_q$ denote this list, $\{\hat{Y}_q^1, \hat{Y}_q^2, \ldots, \hat{Y}_q^N\}$ at the q^{th} step with top-N predictions. The j^{th} element of this list, at the q^{th} step, is denoted by $\hat{Y}_q^j$. Following Liu et al. [10], we use metrics Recall@N (R@N), Precision@N (P@N), F1@N and normalized discounted cumulative gain (nDCG) averaged over the total number of items (where I denotes the indicator function) as follows:

$$P@N = \frac{1}{|Y|} P_q@N = \frac{1}{|Y|} \sum_{p=1}^{|Y|} \frac{|Y_q \cap \hat{Y}_q|}{|\hat{Y}_q|}$$

$$R@N = \frac{1}{|Y|} R_q@N = \frac{1}{|Y|} \sum_{p=1}^{|Y|} \frac{|Y_q \cap \hat{Y}_q|}{|Y_q|}$$

$$F1@N = 2 \times \frac{P@N \times R@N}{P@N + R@N}$$

$$nDCG = \frac{1}{|Y|} \sum_{q=1}^{|Y|} \sum_{j=1}^{|\hat{Y}|} \frac{2^{I(|\hat{Y}_q^j \cap Y_q|)} - 1}{log_2(j+1)}$$

5.4 Results

Table 2 show the results of all our models and the baseline methods on the Movielens-1M dataset. Among the baselines, CARNN has the best performance over all the metrics. Note that *all* the three

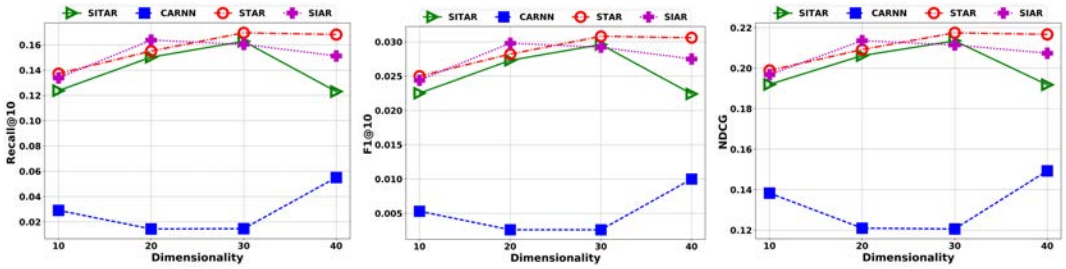


Figure 5: Performance of STAR, SIAR, SITAR and CARNN with varying dimensionality on Movielens dataset

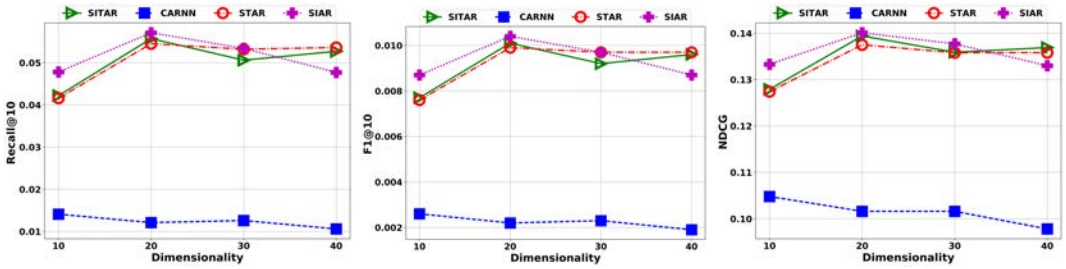


Figure 6: Performance of STAR, SIAR, SITAR and CARNN with varying dimensionality on Amazon Books dataset

models, STAR, SIAR, and SITAR, outperform CARNN in all the metrics. STAR is overall the best performing model.

Table 2: Performance of our models and baselines on the Movielens-1M dataset. Column-wise best results in bold, best baseline results in italics.

Model	R@1	R@5	R@10	F1@1	F1@5	F1@10	nDCG
POP	0.0016	0.0052	0.0118	0.0016	0.0017	0.0021	0.0434
BPR	0.0043	0.0237	0.0436	0.0043	0.0079	0.0033	0.1188
FPMC	0.0054	0.0255	0.0440	0.0054	0.0085	0.0080	0.1291
DREAM	0.0063	0.0318	0.0484	0.0063	0.0106	0.0088	0.1364
CARNN	*0.0075*	*0.0394*	*0.0551*	*0.0075*	*0.0131*	*0.0100*	*0.1493*
STAR	**0.0334**	**0.1070**	**0.1682**	**0.0334**	**0.0357**	**0.0306**	**0.2167**
SIAR	0.0303	0.1003	0.1604	0.0303	0.0334	0.0292	0.2115
SITAR	0.0300	0.1011	0.1628	0.0300	0.0337	0.0296	0.2135

Table 3 shows the results of all our models and the baseline methods on the Amazon Books dataset. Among the baselines, FPMC has the best performance over all the metrics. Note that all the three models, STAR, SIAR, and SITAR, outperform FPMC in all the metrics. Model SIAR has the best performance for metrics R@10, F1@10 and nDCG, whereas model STAR has the best performance for metrics R@1, R@5, F1@1 and R@5.

These results indicate that modeling the temporal evolution of input and temporal contexts yields significant improvement in recommendation performance. In the Movielens-1M dataset temporal context dynamics appears to have a more prominent role, whereas in the Amazon Books dataset, dynamics of both the contexts seem to be important. Although SITAR has significantly better performance than CARNN, it does not outperform STAR or SIAR. This could be due to the larger number of parameters that has to be learnt in SITAR.

5.4.1 Impact of Dimensionality. Figures 5 and 6 compare the performance of our models – STAR, SIAR and SITAR with CARNN, using evaluation metrics R@10, F1@10 and NDCG, at dimensions

Table 3: Performance of our models and baselines on the Amazon Books dataset. Column-wise best results in bold, best baseline results in italics.

Model	R@1	R@5	R@10	F1@1	F1@5	F1@10	nDCG
POP	0.0005	0.0016	0.0036	0.0005	0.0005	0.0007	0.0843
BPR	0.0008	0.0041	0.0078	0.0008	0.0014	0.0014	0.1016
FPMC	*0.0039*	*0.0163*	*0.0285*	*0.0039*	*0.0054*	*0.0052*	*0.1231*
DREAM	0.0008	0.0034	0.0063	0.0008	0.0011	0.0011	0.0915
CARNN	0.0015	0.0060	0.0106	0.0015	0.0020	0.0019	0.0978
STAR	**0.0247**	**0.0419**	0.0536	**0.0247**	**0.0140**	0.0097	0.1358
SIAR	0.0209	0.0413	**0.0571**	0.0209	0.0138	**0.0104**	**0.1401**
SITAR	0.0180	0.0397	0.0557	0.0180	0.0132	0.0101	0.1394

10, 20, 30 and 40. In all cases, our models outperform CARNN. In the Movielens-1M dataset, figure 5, performance of STAR, SIAR and SITAR are comparable at $d = 30$ whereas STAR outperforms the others at $d = 40$. In the Amazon Books dataset, figure 6, the best results of STAR, SIAR and SITAR are comparable, at $d = 20$, with SIAR marginally better than the others. In both the datasets, increasing the dimension beyond 20 does not significantly change the performance of STAR, SIAR and SITAR.

6 CONCLUSION

We design and evaluate context-aware sequential recommendation methods that, for the first time, model the dynamics of input and temporal contexts. Further, the number of parameters in STAR, one of our best performing models does not increase with the number of contexts, which was a limitation of the best previous model. Our experiments on large benchmark datasets show that modeling the evolution of contexts has considerable impact on recommendation performance – all our models significantly outperform state-of-the-art context-aware sequential recommendation models.

ACKNOWLEDGEMENTS

This work is supported by Singapore Ministry of Education Academic Research Fund Tier 1 [R-253-000-138-133].

REFERENCES

[1] Charu C Aggarwal. 2016. *Recommender Systems.* Springer.

[2] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. 2012. Playlist prediction via metric embedding. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 714–722.

[3] Salah El Hihi and Yoshua Bengio. 1996. Hierarchical recurrent neural networks for long-term dependencies. In *Proceedings of the 8th International Conference on Neural Information Processing Systems (NIPS)*. 493–499.

[4] Li Gao, Jia Wu, Chuan Zhou, and Yue Hu. 2017. Collaborative Dynamic Sparse Topic Regression with User Profile Evolution for Item Recommendation.. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*. 1316–1322.

[5] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TIIS)* 5, 4 (2016), 19.

[6] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*. 507–517.

[7] Baoxing Huai, Enhong Chen, Hengshu Zhu, Hui Xiong, Tengfei Bao, Qi Liu, and Jilei Tian. 2014. Toward personalized context recognition for mobile users: A semisupervised Bayesian HMM approach. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 9, 2 (2014), 10.

[8] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. 2010. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM Conference on Recommender Systems*. ACM, 79–86.

[9] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference for Learning Representations (ICLR)*.

[10] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. 2016. Context-aware sequential recommendation. In *Proceedings of the 16th International Conference on Data Mining (ICDM), 2016 IEEE*. IEEE, 1053–1058.

[11] Qiang Liu, Shu Wu, and Liang Wang. 2015. COT: Contextual Operating Tensor for Context-Aware Recommender Systems.. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. 203–209.

[12] Cosimo Palmisano, Alexander Tuzhilin, and Michele Gorgoglione. 2008. Using context to improve predictive modeling of customers in personalization applications. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 20, 11 (2008), 1535–1549.

[13] Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2013. How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026* (2013).

[14] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence (UAI)*. 452–461.

[15] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*. ACM, 811–820.

[16] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 635–644.

[17] David E Rumelhart and James L McClelland. 1986. *Parallel distributed processing: explorations in the microstructure of cognition. volume 1. foundations.* MIT Press, Cambridge, MA.

[18] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, and Alan Hanjalic. 2014. Cars2: Learning context-aware representations for context-aware recommendations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM)*. ACM, 291–300.

[19] Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 650–658.

[20] Yu Sun, Nicholas Jing Yuan, Yingzi Wang, Xing Xie, Kieran McDonald, and Rui Zhang. 2016. Contextual intent tracking for personal assistants. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 273–282.

[21] Yu Sun, Nicholas Jing Yuan, Xing Xie, Kieran McDonald, and Rui Zhang. 2016. Collaborative nowcasting for contextual recommendation. In *Proceedings of the 25th International Conference on World Wide Web*. 1407–1418.

[22] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 403–412.

[23] Shoujin Wang, Liang Hu, Longbing Cao, Xiaoshui Huang, Defu Lian, and Wei Liu. 2018. Attention-based transactional context embedding for next-item recommendation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

[24] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 729–732.

Improved Cross-Lingual Question Retrieval for Community Question Answering

Andreas Rücklé
Ubiquitous Knowledge Processing Lab, Department of Computer Science
Technische Universität Darmstadt
www.ukp.tu-darmstadt.de

Krishnkant Swarnkar*
Ubiquitous Knowledge Processing Lab, Department of Computer Science
Technische Universität Darmstadt
www.ukp.tu-darmstadt.de

Iryna Gurevych
Ubiquitous Knowledge Processing Lab, Department of Computer Science
Technische Universität Darmstadt
www.ukp.tu-darmstadt.de

ABSTRACT

We perform cross-lingual question retrieval in community question answering (cQA), i.e., we retrieve similar questions for queries that are given in another language. The standard approach to cross-lingual information retrieval, which is to automatically translate the query to the target language and continue with a monolingual retrieval model, typically falls short in cQA due to translation errors. This is even more the case for specialized domains such as in technical cQA, which we explore in this work. To remedy, we propose two extensions to this approach that improve cross-lingual question retrieval: (1) we enhance an NMT model with monolingual cQA data to improve the translation quality, and (2) we improve the robustness of a state-of-the-art neural question retrieval model to common translation errors by adding back-translations during training. Our results show that we achieve substantial improvements over the baseline approach and considerably close the gap to a setup where we have access to an external commercial machine translation service (i.e., Google Translate), which is often not the case in many practical scenarios. Our source code and data is publicly available.[1]

CCS CONCEPTS

• **Information systems → Multilingual and cross-lingual retrieval**; *Learning to rank*; *Question answering*.

KEYWORDS

Community Question Answering, Question Retrieval, Cross-lingual Retrieval, Representation Learning, Neural Machine Translation

ACM Reference Format:
Andreas Rücklé, Krishnkant Swarnkar, and Iryna Gurevych. 2019. Improved Cross-Lingual Question Retrieval for Community Question Answering. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313502

*This author is currently at the Indian Institute of Technology (BHU) Varanasi. The present work was performed as part of an internship at UKP Lab.
[1]https://github.com/UKPLab/www19-xling-question-retrieval

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313502

1 INTRODUCTION

Question retrieval is an important task in community question answering (cQA) that allows us to use the existing knowledge which is present in cQA platforms to automatically answer non-factoid questions. We do so by retrieving previously answered questions that are similar to the input question (or query) with the goal of selecting one of their answers as the output [6, 10, 24, 30, 38].

Even though this is a research field with a long-standing history, approaches to question retrieval were traditionally proposed for monolingual scenarios only. However, *cross-lingual* question retrieval is crucial when considering languages other than English because there exists substantially less cQA data in these cases. This is particularly apparent within technical domains, on which we focus in this work, where English platforms such as StackOverflow[2] and AskUbuntu[3] are predominant.

To date, cross-lingual question retrieval has only been explored by two previous works for the language pair English/Arabic and only for general domains, i.e., questions from the Qatar Living forum[4]. However, cross-lingual question retrieval for specialized domains, such as technical cQA, presents us with different challenges because there exists less non-English in-domain data.

First, Joty et al. [22] use adversarial training with feed-forward networks to learn language-invariant feature representations for question pairs. Adversarial training requires in-domain questions from both languages, which are typically not available for specialized domains (e.g., there is no Arabic StackOverflow). And second, Martino et al. [27] apply a mixture of a cross-lingual tree kernel with a dictionary and machine translation with monolingual features. Importantly, the effects from using machine translation were relatively small in their case. However, applying machine translation (MT) to specialized domains is typically more error-prone because large parallel corpora which are used to train MT models mostly contain sentences from general domains [12, 26].

In this work we explore cross-lingual question retrieval for technical domains, i.e., for cQA platforms that cover programming and operating systems topics. Similar to Martino et al. [27] we first use a standard approach to cross-lingual information retrieval, which is to machine translate the query to our target language and then continue with a monolingual question retrieval model [17, 25]. Different to previous work, machine translations would lead to a higher performance decrease for cross-lingual question retrieval in to our specialzed domain. To remedy, we extend this approach with two cross-lingual adaptations that (1) improve the in-domain

[2]www.stackoverflow.com
[3]www.askubuntu.com
[4]www.qatarliving.com/forum

translation quality and (2) increase the robustness of the question retrieval model to common translation errors.

We achieve this by (a) performing domain-adaptation for the Transformer [42], a state-of-the-art neural machine translation (NMT) approach, by extending the training corpus with synthetic parallel in-domain sentences from cQA platforms. The method we use is known as back-translation in MT literature and has proven to be effective before [3, 11, 23, 31, 37]. In contrast, we test this method extrinsically within question retrieval.

And (b) we increase the robustness of RCNN [24], a state-of-the-art neural question retrieval model, to common translation errors by adding back-translated texts during training. Back-translating is sometimes used to automatically generate paraphrases, e.g., to train paraphrastic sentence embeddings [43] or to obtain more diverse sentences for other (monolingual) tasks [9, 19, 45]. Different to them, we adapt this method to our cross-lingual retrieval scenario.

We perform experiments for the language pair English/German and evaluate our approaches on two datasets from different technical cQA domains—the Askubuntu benchmark [10, 24] and a dataset that we create based on questions from StackOverflow.

Our results show that there exists a substantial cross-lingual performance decrease compared to the monolingual setup, which is in contrast to previous work that performed cross-lingual question retrieval in general domains [27]. In our analysis we observe that this is mostly due to translation errors in our specialized domains. Finally, our proposed adaptations substantially improve the cross-lingual question retrieval performance and close the gap by up to 85% to a scenario where we use a state-of-the-art external commercial MT service (Google Translate) with no adaptations.

Our contributions:

(1) We are the first to explore cross-lingual question retrieval in *technical cQA* and we observe that the retrieval performance is strongly influenced by the translation quality.
(2) We evaluate the impact of NMT domain adaptation extrinsically in cQA question retrieval.
(3) We reduce the cross-lingual performance gap to a setup that has access to an external state-of-the-art commercial MT service with no adaptations by up to 85%. This demonstrates that we can use freely available NMT approaches for cross-lingual question retrieval within technical cQA.

2 RELATED WORK

Early approaches to **monolingual question retrieval** use translation models [2, 20, 44, 49], which can further be enhanced with question category information [6], and topic models [21, 46].

More recent work, especially in the context of the SemEval cQA challenges [30], improve upon this and use tree kernels [8, 34], text alignment features [13], multi-task learning [4], and unsupervised methods with shallow lexical matching and mismatching [47].

In technical cQA domains, neural representation learning methods have proved to be most effective. Dos Santos et al. [10], for example, learn representations of questions with CNNs and compare them with cosine similarity for scoring. Lei et al. [24] propose RCNN, which extends the CNN with a recurrent mechanism (adaptive gated decay). This approach was further extended with question-type information [16], which relies on a question taxonomy for general cQA domains that does not cover the different types of technical questions (e.g., for programming topics).

Other approaches that were applied to technical cQA focus on adversarial domain transfer [38] and on programming-specific association features [48].

Cross-lingual question retrieval has received considerably less attention. Martino et al. [27] explore the language pair English/Arabic within general domains and used a combination of a cross-lingual tree-kernel and machine translation for retrieval. Similarly, Joty et al. [22] explore the language pair English/Arabic within general domains and instead use adversarial training in order to learn language invariant feature representations of question pairs.

Highly related fields are cross-lingual information retrieval and cross-lingual question answering, which was also part of challenges in CLEF 2008 [14], NTCIR-8 [29], and BOLT [39]. Most approaches in this context rely on standard machine translation systems or dictionaries to map questions to their target language [5, 17, 25, 32] and some combine results from multiple MT systems [41], multiple translation outputs from the same MT system [36], or jointly train SMT and IR components [18] to achieve better results.

To the best of our knowledge, we are the first to explore cross-lingual question retrieval in *technical* cQA domains. In contrast to previous work in cross-lingual question retrieval, because specialized domains are more affected by translation errors, we explore the standard approach to cross-lingual information retrieval with two adaptations to (1) improve the translation quality and (2) increase the robustness of a question retrieval model to translation errors.

3 TASK AND METHOD

Given an input question q_0 (the query) and a list of N (potentially) related questions $Q = q_1, q_2, \ldots, q_N$ from a cQA platform that are retrieved with a search engine, the goal is to re-rank this list according to each related question's relevance in regard to q_0.

In our cross-lingual setup, the query is given in a source language L1 (German) and the related questions are in a target language L2 (English). We indicate languages with a superscript, e.g., q_0^{L1} stands for the query in the source language and q_0^{L2} stands for the query in the target language. To perform question retrieval given q_0^{L1} and Q^{L2} we first translate q_0^{L1} to L2 using neural machine translation (NMT). Then we continue with a monolingual L2 question retrieval model to re-rank Q^{L2} according to the translated q_0^{L2}.

This is a standard technique in cross-lingual information retrieval [17, 25] and it has been applied to cross-lingual question retrieval in general domains [27]. We go beyond this by exploring the effects of NMT to the performance of neural question retrieval in specialized domains, i.e., programming and operating systems cQA.

Most importantly, our domains contain a specialized vocabulary which makes NMT—and thus also this standard retrieval approach—more error-prone. This could be even more the case for statistical machine translation [7], which has been used in previous question retrieval work [27]. Thus, in §4 we present two adaptations to improve the translation and ranking performances in our domains.

In the following, we present the NMT model and the neural re-ranking approach. We visualize the full pipeline in Figure 1.

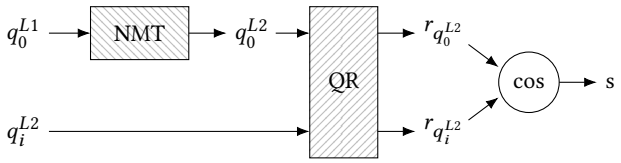


Figure 1: The cross-lingual pipeline. The question retrieval (QR) model learns (monolingual) question representations, which are then compared with cosine similarity to determine a ranking score s. L2 is always English in our setup.

NMT: Synthetic Parallel Data (§4.1)		
↷	orig	Unable to change the launcher icon size
↳	en→de	Nicht in der Lage, die Starter-Symbolgröße zu ändern
QR: Data Augmentation (§4.2)		
↷	orig	How to convert a map to list in java?
↷	en→de	Wie konvertiert man eine Map in Java?
↳	de→en	How to convert a map in Java?

Table 1: Examples for our two data-driven adaptations.

3.1 Neural Machine Translation (NMT)

NMT is a popular research field that has seen rapid progress in recent years. The most effective approaches use the encode-decoder architecture [40] and integrate various forms of attention [1, 15, 42].

In this work, we use Transformer [42], which is a state-of-the-art NMT approach and is based on multi-head self-attention layers. Because it does not make use of recurrent units, Transformer can be efficiently trained on modern hardware while achieving better translation results compared to computationally more expensive models. Edunov et al. [11], for example, use this approach and achieve state-of-the-art results on the WMT'14 en-de benchmark.

For our cross-lingual question retrieval approach we train two models, one for each direction en→de and de→en using the public WMT13 and WMT18 training corpora with more than 4.5M parallel sentences. Even though a small number of sentences in the training data are on technical topics, there is no specific data that is related to technical cQA. To the best of our knowledge there are no publicly available parallel corpora that cover technical questions from cQA.

In addition, for a more thorough comparison, we use an external commercial translation service, namely Google Translate. Even though we expect Google Translate to produce higher quality translations because their model was trained on large amounts of proprietary parallel data, it is not suitable for many practical cQA scenarios due to (1) potential technical restrictions (requiring internet access), (2) legal reasons (confidential information might be transmitted), or (3) associated costs (commercial translation services are not free). The same restrictions do not apply to Transformer models because they are available on-premise.

In the rest of this work we indicate the use of Transformer with TR and the use of Google Translate with GT.

3.2 Question Retrieval (QR)

We use RCNN [24] as a basis for our experiments. This approach combines recurrent units and convolutional neural networks, and it achieves state-of-the-art results on the AskUbuntu question retrieval benchmark [10, 24]. Even though RCNN was recently extended with a separate question type classifier and information from a question taxonomy to improve performances within general domains [16], we use standard RCNN to avoid introducing an additional in-domain question taxonomy (question-type classification would be another source of error in our cross-lingual setup).

For an input question q RCNN learns a fixed-size dense vector representation r_q. This representation is obtained by applying the recurrent CNN sequentially on each token in q where the last output state is considered as the question representation. Model details are given in [24]. This is a strictly monolingual model, which means that we can only learn representations for texts that are in L2.

To rank a related question q_i^{L2} in regard to the query q_0^{L2}, we compare the learned representations with cosine similarity to determine a ranking score s:

$$s = \cos\left(r_{q_i^{L2}},\ r_{q_0^{L2}}\right)$$

To train RCNN we obtain labeled question paraphrases (also called duplicates) from the cQA platform, i.e., q_0^{L2} and q_+^{L2}. We also obtain one negative sample q_-^{L2} by randomly selecting N questions from the whole corpus and choosing the one with the highest ranking score according to the currently trained model. We then minimize the max-margin hinge loss:

$$\mathcal{L} = \max\left[0,\ m - \cos\left(r_{q_0^{L2}}, r_{q_+^{L2}}\right) + \cos\left(r_{q_0^{L2}}, r_{q_-^{L2}}\right)\right]$$

where m is a non-negative margin.

We additionally perform unsupervised pre-training as proposed in [24], where RCNN is pre-trained in an encoder/decoder setup. The encoder RCNN learns a suitable representation of the question title and/or body and the decoder tries to reconstruct the title form the learned representation.

In general, RCNN can use both the question title and the question body during scoring by applying the element-wise average over the learned representations for both texts. In this work, we use only question titles because it allows us to compare individual sentences for error analysis in our cross-lingual setup. In addition, the performances of models that utilize body information was only marginally better compared to the title-only variant (with pre-training).

4 CROSS-LINGUAL ADAPTATIONS

4.1 In-Domain NMT with Synthetic Parallel Sentences

One disadvantage of training Transformer models on public WMT corpora is that they only contain texts from common domains—they do not include parallel sentences from technical cQA.

Because applying NMT models in out-of-domain scenarios often leads to sub-par translation performances [12, 26], we adapt the Transformer to our specialized domains. There exist several options for domain adaptation, which range from data-centric methods that extend the training corpus to model-centric methods that change the neural network architecture (see [7] for a survey).

In this work we choose a data-centric approach based on back-translation, which has been proven to be effective in both statistical MT [3, 23] and NMT [11, 31, 37] before. Here we train a Transformer model on synthetic parallel sentences which we generate from monolingual in-domain texts. To generate these parallel sentences we apply another Transformer model, which we originally trained on the general WMT corpora, on sentences from technical cQA.

Because we translate queries from L1 to L2 in our cross-lingual question retrieval approach (see Figure 1), we use monolingual L2 data to generate L1 sentences, i.e., the synthetic data is generated by translating in-domain sentences from the target to the source language. This is suitable in our setup because L2 is always English, which means that large-scale in-domain data exists for L2. Also, Lambert et al. [23] show that using translations that were obtained from target to source side results in better domain adaptation compared to using translations from source to target side.

An example of our synthetic data is shown in Table 1. We introduce the source dataset later in §5.1. Based on the synthetic data we then train the in-domain Transformer model, which we denote `TR-cQA`, by back-translating the generated sentences from L1 to L2. During training we also include the WMT corpora.

4.2 QR Data Augmentation

In addition to adapting the NMT model to our specialized domain, we also enhance the monolingual question retrieval model with the goal of increasing its robustness to common translation errors.

Similar to our NMT adaptation we choose a data-centric approach because it allows us to use the same `RCNN` network architecture with pre-training as before. In contrast, however, we now augment the training data with back-translated texts, i.e., we translate the titles in the training data from L2 to L1 and back to L2 and use these generated texts during training.

Back-translations are often used as an automatic method to obtain paraphrases, e.g., they can be used to train sentence embeddings [43], to increase the model robustness against adversarial attacks [19], or to obtain more syntactically diverse sentences for training [9, 45]. Here we use them to introduce the QR model to common translation errors. An example is given in Table 1 (bottom), where the aspect of converting sth. *to a list* is lost during translation.

We use `GT` to obtain back-translations because in initial experiments we found that `TR` back-translations were too noisy and in many cases they completely altered the meaning of a sentence. The reason is that in the direction en→de we cannot obtain a good `TR-cQA` model because there exists no large German/L1 technical cQA platform from which we could generate synthetic parallel sentences.

We use back-translated texts in the pre-training and in the regular model training of `RCNN` (see §3.2). In both phases we add a new training example for every data point by replacing the original question title with its back-translated text.

In the rest of this work we denote the augmented model `RCNN-A`.

5 EXPERIMENTS

5.1 Experimental Setup

Data. We test our approaches on two datasets from different technical domains. The dataset statistics are shown in Table 2.

Dataset	Number of Examples			N	Paraphrases
	Train	Valid	Test		Valid+Test
AskUbuntu	12,724	189	186	20	6.0
StackOverflow	23,558	1,770	2,779	20	1.3

Table 2: Dataset statistics. N is the number of (potentially) related questions for re-ranking.

First, we use the publicly available *AskUbuntu* benchmark [10] which has been extended with additional manual relevance annotations in [24]. For our cross-lingual setup we got the queries in the validation and test splits translated to German by a German native speaker with proficient English skills and in-domain knowledge.

Second, we obtained questions with labeled duplicates from StackOverflow to create a similar dataset for a different technical domain. We removed all questions that were not tagged with Java or Python so that our dataset has a similar size compared to AskUbuntu. We used BM25 [33] to obtain 20 potentially related questions for every query in our validation and test splits. Questions that were labeled as duplicates by StackOverflow users are considered as relevant (i.e., should be retrieved). In contrast to AskUbuntu we chose much larger validation and test splits to better reduce the effects from random neural network initialization. Due to this larger size, we used Google Translate to obtain the German query questions. Using Google Translate to obtain cross-lingual data has been used before and it was shown that this only minorly impacts the evaluation outcome in other NLP tasks [35].

In the rest of this work we refer to this dataset as *StackOverflow*.

Models and baselines. We evaluate our cross-lingual approach as described in §3 (`RCNN TR`) with all combinations of our extensions, i.e., `RCNN` with in-domain NMT (`RCNN TR-cQA`), with data augmentation (`RCNN-A TR`) and with both extensions (`RCNN-A TR-cQA`). We also evaluate the same approach with Google Translate instead of Transformer to provide a comparison to a commercial state-of-the-art MT system (`RCNN GT` and `RCNN-A GT`).

Additionally, we evaluate `TF*IDF`, `RCNN`, and `RCNN-A` in the monolingual setup to provide the monolingual baseline and upper-bound.

Training procedure. We train the `TR` models with the official Transformer implementation[5]. To obtain synthetic in-domain training data for `TR-cQA` we use `TR` to translate the titles from the training split of both AskUbuntu and StackOverflow datasets to German (no data from the evaluation splits was used in `TR-cQA`).

To train `RCNN` models we use the code provided by Lei et al. [24].

Neural network setup. We use the standard hyperparameters of `RCNN` except for the batch size and margin parameter m where we perform grid search for each model and dataset combination. This is necessary due to the different sizes of the datasets and our data augmentation. The values for the batch size and margin are in {128, 256} and {0.1, 0.5, 1.0} respectively.

For our AskUbuntu experiments we use the 200D word embeddings of Lei et al. [24], which were trained on in-domain data. For StackOverflow we train new 200D word embeddings with Word2Vec [28] on text from stackoverflow.com.

[5] https://github.com/tensorflow/tensor2tensor

Evaluation. For all models we report mean average precision (MAP), mean reciprocal rank (MRR), and precision@1 (P@1; accuracy). To mitigate the effects from random initialization of neural network weights, we report averaged scores over five runs.

5.2 Experimental Results

Table 3 compares the monolingual and cross-lingual performances of the different models for both tasks.

The results show that there exists a considerable performance gap between both setups, e.g., compared to the monolingual RCNN, RCNN TR decreases by 6.6 MAP on StackOverflow and by 3.1 MAP on AskUbuntu.[6] This is in contrast to previous work, where much smaller effects were observed when applying phrase-based MT to cross-lingual question retrieval for general domains [27].

Furthermore, we observe that the translation quality greatly varies in technical cQA and has strong effects on the cross-lingual question retrieval performance—with GT we can achieve substantially better results compared to TR (3.1 MAP on average).

The results also show that our proposed adaptations are effective. They consistently outperform RCNN TR and they substantially close the performance gap to RCNN GT. This is visualized in Figure 2 where we measure the performance improvements in terms of how much they close this performance gap. It shows that adapting TR to technical cQA domains by using synthetic training data has the strongest impact, i.e., it closes the performance gap by over 50% (question retrieval performance increases by 1.8 MAP, 1.9 MRR, and 2.1 P@1, on average). Augmenting the training data of RCNN with back-translated questions further increases the performance and closes the gap by another 19pp to 34pp (+0.6 MAP, +0.9 MRR, and +1.4 P@1). At the same time we observe that data augmentation is also effective for RCNN GT and within the monolingual scenario. The reason for this is that during training the model receives a form of regularization (through noisy samples) and processes additional variations of the monolingual training questions.

The combined model RCNN-A TR-cQA finally substantially reduces the gap to RCNN GT by 75–85%.

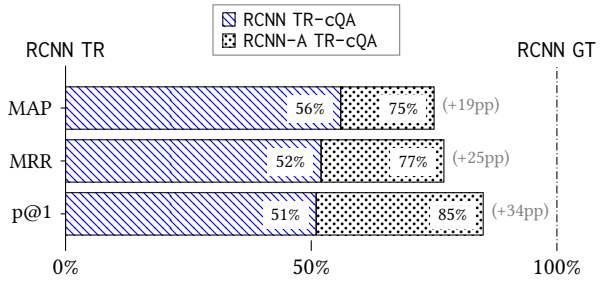


Figure 2: A visualization that shows how much our adaptations close the performance gap between RCNN TR and RCNN GT. Results are averaged over both datasets.

6 ANALYSIS

In-domain NMT performance. To test if the improvements with TR-cQA over TR are due to improved translations, we evaluate both models with our human translations from the AskUbuntu evaluation splits and measure their translation performance for de→en. TR-cQA achieves 54.96 BLEU whereas TR has a substantially lower score of 41.70 BLEU. Even though the number of parallel sentences is small (375), the large difference suggests that the improvements of RCNN TR-cQA are indeed due to more accurate translations.

At the same time TR-cQA only decreases marginally on the WMT'14 en/de benchmark (28.04 BLEU vs. 28.30 BLEU).[7]

Error analysis. We compared all cases in which RCNN-A TR-cQA ranks a relevant question first whereas RCNN TR ranks an unrelated question first. Generally, RCNN-A TR-cQA is more robust if the translations slightly differ from the English original query, e.g., when translating "throwing an exception" to "triggering an exception" (which is not semantically equal). In most cases, however, the effects from the improved translations are more apparent.

Figure 4 shows a case where TR discards information (i.e., "discrete graphics"). In contrast, TR-cQA generates a translation that is mostly equal to the English source text. During analysis we found that this is a common error of TR, which is greatly improved by TR-cQA due to its better in-domain translation quality. In a small number of cases, however, TR-cQA still discards important information, e.g., when translating long compounds. Another common issue of TR is that it translates words without considering their in-domain meaning, e.g., Figure 5 shows an example where TR translates "null" to "zero".[8] Such errors are often avoided by TR-cQA.

We also compared RCNN GT, RCNN-A TR-cQA, and the monolingual RCNN to understand how we can further improve the cross-lingual question retrieval performances. Most importantly, both RCNN GT and RCNN-A TR-cQA suffer from the same type of problems as described before, but on a smaller scale. This suggests that further improvements are likely with better MT domain adaptation.

7 DISCUSSION

Commercial MT services often achieve better results compared to publicly available models[9] because they have access to large proprietary parallel corpora. In §5.2 we observed a similar trend where RCNN GT still performed better than TR with cross-lingual adaptations. Further, data augmentation in RCNN showed improvements in all cases, including GT. *Why can't we just use Google Translate then?*

There exist several disadvantages when using external MT services in practical scenarios. For example, they can be costly and the data is transferred to third parties. Consider, for example, a non-English company that wants to provide an automatic tool to their software developers that helps them to find related English questions within the StackOverflow data dump (if nothing is found the input question will be sent to colleagues via an internal cQA platform). Because questions of software developers can contain critical and confidential information—e.g., information about the

[6] We also evaluated cross-lingual variants of TF*IDF (with the same NMT approaches), and we tested Spanish RCNN GT with a Spanish version of our StackOverflow dataset. They showed the same trends as the reported RCNN models. We omit the results here for clarity and due to space reasons.

[7] BLEU scores in AskUbuntu are higher because the sentences (i.e., question titles) are considerably shorter compared to WMT'14 sentences.

[8] Null in German typically refers to the number zero. There exist many similar cases, e.g., TR translates (http-) post to "mail".

[9] See, e.g., https://www.deepl.com/quality.html

Setup	StackOverflow Dev			/ Test			AskUbuntu Dev			/ Test		
Model	MAP	MRR	P@1	MAP	MRR	P@1	MAP	MRR	P@1	MAP	MRR	P@1
Monolingual												
TF*IDF	58.9	61.0	47.7	54.6	56.7	43.0	53.0	67.5	54.5	52.0	64.5	50.5
RCNN	65.3	67.3	54.5	60.2	62.3	47.9	58.3	72.1	60.1	60.3	73.9	60.6
RCNN-A	65.9	67.9	54.9	60.8	62.9	48.7	58.4	72.8	60.5	60.6	75.7	64.2
Cross-lingual												
RCNN GT	62.3	64.2	50.8	57.7	59.7	45.3	57.3	70.1	56.1	59.0	73.7	60.3
RCNN-A GT	62.9	64.8	51.2	58.5	60.5	46.3	57.8	71.1	57.8	59.2	74.9	62.7
RCNN TR	57.2	59.0	44.5	53.6	55.4	40.4	55.9	68.3	54.3	57.2	70.8	56.9
RCNN TR-cQA	60.2	62.0	48.1	55.4	57.4	42.5	57.0	69.8	55.9	58.3	71.7	58.0
RCNN-A TR	58.1	59.9	45.3	54.4	56.2	41.4	56.1	68.4	54.4	58.3	**73.2**	**60.4**
RCNN-A TR-cQA	**61.2**	**63.1**	**49.3**	**56.0**	**58.0**	**43.1**	**57.6**	**70.5**	**57.4**	**58.4**	72.8	60.3

Table 3: The monolingual and cross-lingual question retrieval performances of the different models.

AskUbuntu

Query: wie kann man diskrete grafik in ubuntu 14.04 deaktivieren
Original text (en): how to disable discrete graphic in ubuntu 14.04

RCNN TR-cQA	RCNN TR
Translation: how to disable discrete graphics in ubuntu 14.04	**Translation:** how to disable grafik in ubuntu 14.04
Top-3 retrieved questions	
✓ 14.04 how to disable discrete graphics card	✗ overheating laptop dual ati gpu and discrete
✓ how can i disable ati discrete graphic gpu at startup in ubuntu 14.04 without bios	✓ disabling discrete gpu at startup without system crash
✗ disabling discrete gpu at startup without system crash	✓ how can i disable ati discrete graphic gpu at startup in ubuntu 14.04 without bios

Table 4: A comparison of RCNN TR-cQA and RCNN TR on AskUbuntu, which gives an example of a case where TR discards information during translation.

StackOverflow

Query: try catch exception gibt immer null zurück
Original text (en): try catch exception always returns null

RCNN TR-cQA	RCNN TR
Translation: try catch exception always returning null	**Translation:** try catch exception always returns zero
Top-3 retrieved questions	
✓ catching null exception	✗ catch same exception multiple times
✗ exception handling with multiple catch block	✗ exception handling with multiple catch block
✗ catch same exception multiple times	✗ is it expensive to use try - catch blocks even if an exception is never thrown ?

Table 5: A comparison of RCNN TR-cQA and RCNN TR on StackOverflow, which gives an example in-domain vs. out-of-domain translation (see null vs. zero in the translations).

software architecture, stack traces, etc.—it is not suitable to send this data to an external service for translation. In such cases MT needs to be performed with a model that is available on-premise to avoid exposing internal information to third parties.

8 CONCLUSION

In this work we performed cross-lingual question retrieval for *technical cQA* by machine translating the queries to our target language and continuing with a monolingual question retrieval model.

We observed a considerable performance decrease in the cross-lingual scenario compared to the monolingual setup, which is due to a high number of MT errors within our specialized domains.

To remedy this, we first performed domain adaptation for a state-of-the-art NMT model by adding synthetic in-domain parallel sentences to the training corpus. This greatly improved the cross-lingual question retrieval performance due to better in-domain translations. Second, we added back-translated texts during the training of the question retrieval model to increase its robustness, which further improved cross-lingual question retrieval. Finally, the combination of both adaptations substantially closed the performance gap by up to 85% to a model that has access to an external state-of-the-art commercial MT system (with no adaptations), which is often not the case in many practical scenarios.

ACKNOWLEDGEMENTS

This work has been supported by the German Research Foundation (DFG) as part of the QA-EduInf project (grant GU 798/18-1 and grant RI 803/12-1) and by the German Federal Ministry of Education and Research (BMBF) as part of the Software Campus program under the promotional reference 01IS17050.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations (ICLR 2015)*.

[2] Delphine Bernhard and Iryna Gurevych. 2009. Combining lexical semantic resources with question & answer archives for translation-based answer finding. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Association for Computational Linguistics, 728–736.

[3] Nicola Bertoldi and Marcello Federico. 2009. Domain Adaptation for Statistical Machine Translation with Monolingual Resources. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, 182–189. http://aclweb.org/anthology/W09-0432

[4] Daniele Bonadiman, Antonio Uva, and Alessandro Moschitti. 2017. Effective shared representations with Multitask Learning for Community Question Answering. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*. Association for Computational Linguistics, 726–732. http://aclweb.org/anthology/E17-2115

[5] Gosse Bouma, Jori Mur, Gertjan van Noord, Lonneke van der Plas, and Jörg Tiedemann. 2008. Question Answering with Joost at CLEF 2008. In *9th Workshop of the Cross-Language Evaluation Forum (CLEF 2008)*. 257–260.

[6] Xin Cao, Gao Cong, Bin Cui, Christian S Jensen, and Quan Yuan. 2012. Approaches to exploring category information for question retrieval in community question-answer archives. *ACM Transactions on Information Systems (TOIS)* 30, 2 (2012), 7.

[7] Chenhui Chu and Rui Wang. 2018. A Survey of Domain Adaptation for Neural Machine Translation. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*. Association for Computational Linguistics, 1304–1319. http://aclweb.org/anthology/C18-1111

[8] Giovanni Da San Martino, Alberto Barrón Cedeño, Salvatore Romeo, Antonio Uva, and Alessandro Moschitti. 2016. Learning to re-rank questions in community question answering using advanced features. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM 2016)*. ACM, 1997–2000.

[9] Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to Paraphrase for Question Answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*. Association for Computational Linguistics, 875–886. https://doi.org/10.18653/v1/D17-1091

[10] Cicero Dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. 2015. Learning Hybrid Representations to Retrieve Semantically Equivalent Questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Association for Computational Linguistics, 694–699. https://doi.org/10.3115/v1/P15-2114

[11] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381* (2018).

[12] M. Amin Farajian, Marco Turchi, Matteo Negri, Nicola Bertoldi, and Marcello Federico. 2017. Neural vs. Phrase-Based Machine Translation in a Multi-Domain Scenario. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*. Association for Computational Linguistics, 280–284. http://aclweb.org/anthology/E17-2045

[13] Simone Filice, Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2016. KeLP at SemEval-2016 Task 3: Learning Semantic Relations between Questions and Answers. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. Association for Computational Linguistics, 1116–1123. https://doi.org/10.18653/v1/S16-1172

[14] Pamela Forner, Anselmo Peñas, Eneko Agirre, Iñaki Alegria, Corina Forăscu, Nicolas Moreau, Petya Osenova, Prokopis Prokopidis, Paulo Rocha, Bogdan Sacaleanu, et al. 2008. Overview of the clef 2008 multilingual question answering track. In *9th Workshop of the Cross-Language Evaluation Forum (CLEF 2008)*. 262–295.

[15] Jonas Gehring, Michael Auli, David Grangier, and Yann N Dauphin. 2016. A convolutional encoder model for neural machine translation. *arXiv preprint arXiv:1611.02344* (2016).

[16] Deepak Gupta, Rajkumar Pujari, Asif Ekbal, Pushpak Bhattacharyya, Anutosh Maitra, Tom Jain, and Shubhashis Sengupta. 2018. Can Taxonomy Help? Improving Semantic Question Matching using Question Taxonomy. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*. Association for Computational Linguistics, 499–513. http://aclweb.org/anthology/C18-1042

[17] Sven Hartrumpf, Ingo Glöckner, and Johannes Leveling. 2008. Efficient question answering with question decomposition and multiple answer streams. In *Workshop of the Cross-Language Evaluation Forum for European Languages*. Springer, 421–428.

[18] Felix Hieber and Stefan Riezler. 2015. Bag-of-Words Forced Decoding for Cross-Lingual Information Retrieval. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2015)*. Association for Computational Linguistics, 1172–1182. https://doi.org/10.3115/v1/N15-1123

[19] Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2018)*. Association for Computational Linguistics, 1875–1885. https://doi.org/10.18653/v1/N18-1170

[20] Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management (CIKM 2005)*. 84–90.

[21] Zongcheng Ji, Fei Xu, Bin Wang, and Ben He. 2012. Question-answer Topic Model for Question Retrieval in Community Question Answering. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM '12)*. ACM, New York, NY, USA, 2471–2474. https://doi.org/10.1145/2396761.2398669

[22] Shafiq Joty, Preslav Nakov, Lluís Màrquez, and Israa Jaradat. 2017. Cross-language Learning with Adversarial Neural Networks. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Association for Computational Linguistics, 226–237. https://doi.org/10.18653/v1/K17-1024

[23] Patrik Lambert, Holger Schwenk, Christophe Servan, and Sadaf Abdul-Rauf. 2011. Investigations on Translation Model Adaptation Using Monolingual Data. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, 284–293. http://aclweb.org/anthology/W11-2132

[24] Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi Jaakkola, Katerina Tymoshenko, Alessandro Moschitti, and Lluis Marquez. 2016. Semi-supervised Question Retrieval with Gated Convolutions. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2016)*, 1279–1289. https://doi.org/10.18653/v1/N16-1153

[25] Chuan-Jie Lin and Yu-Min Kuo. 2010. Description of the NTOU Complex QA System.. In *NTCIR-8 Workshop*. 47–54.

[26] Minh-Thang Luong and Christopher D Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proceedings of IWSLT*. 76–79.

[27] Giovanni Da San Martino, Salvatore Romeo, Alberto Barrón-Cedeño, Shafiq R. Joty, Lluís Màrquez i Villodre, Alessandro Moschitti, and Preslav Nakov. 2017. Cross-Language Question Re-Ranking. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017)*.

[28] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems (NIPS 2013)*. 3111–3119.

[29] Teruko Mitamura, Eric Nyberg, Hideki Shima, Tsuneaki Kato, Tatsunori Mori, Chin-Yew Lin, Ruihua Song, Chuan-Jie Lin, Tetsuya Sakai, Donghong Ji, et al. 2008. Overview of the NTCIR-7 ACLIA Tasks: Advanced Cross-Lingual Information Access. In *Proceedings of the 8th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering, and Cross-Lingual Information Access (NTCIR-8)*.

[30] Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 Task 3: Community Question Answering.

[31] Alberto Poncelas, Dimitar Shterionov, Andy Way, Gideon Maillette de Buy Wenniger, and Peyman Passban. 2018. Investigating Backtranslation in Neural Machine Translation. *arXiv preprint* (2018). http://arxiv.org/abs/1804.06189

[32] Amir Pouran Ben Veyseh. 2016. Cross-Lingual Question Answering Using Common Semantic Space. In *Proceedings of TextGraphs-10: the Workshop on Graph-based Methods for Natural Language Processing*. Association for Computational Linguistics, 15–19. https://doi.org/10.18653/v1/W16-1403

[33] Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 232–241.

[34] Salvatore Romeo, Giovanni Da San Martino, Alberto Barrón-Cedeno, Alessandro Moschitti, Yonatan Belinkov, Wei-Ning Hsu, Yu Zhang, Mitra Mohtarami, and James Glass. 2016. Neural attention for learning to rank questions in community question answering. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016)*. 1734–1745.

[35] Andreas Rücklé, Steffen Eger, Maxime Peyrard, and Iryna Gurevych. 2018. Concatenated Power Mean Embeddings as Universal Cross-Lingual Sentence Representations. *arXiv* (2018). https://arxiv.org/abs/1803.01400

[36] Shadi Saleh and Pavel Pecina. 2016. Reranking Hypotheses of Machine-Translated Queries for Cross-Lingual Information Retrieval. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*. Springer International Publishing, 54–66.

[37] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*. Association for Computational Linguistics, 86–96. https://doi.org/10.18653/v1/P16-1009

[38] Darsh Shah, Tao Lei, Alessandro Moschitti, Salvatore Romeo, and Preslav Nakov. 2018. Adversarial Domain Adaptation for Duplicate Question Detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*. Association for Computational Linguistics, 1056–1063. http://aclweb.org/anthology/D18-1131

[39] Ian Soboroff, Kira Griffitt, and Stephanie Strassel. 2016. The BOLT IR Test Collections of Multilingual Passage Retrieval from Discussion Forums. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR 2016)*. ACM, 713–716.

[40] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems (NIPS 2014)*. 3104–3112.

[41] Ferhan Ture and Elizabeth Boschee. 2016. Learning to Translate for Multilingual Question Answering. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*. Association for Computational Linguistics, 573–584. https://doi.org/10.18653/v1/D16-1055

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS 2017)*. 5998–6008.

[43] John Wieting, Jonathan Mallinson, and Kevin Gimpel. 2017. Learning Paraphrastic Sentence Embeddings from Back-Translated Bitext. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*. Association for Computational Linguistics, 274–285. http://aclweb.org/anthology/D17-1026

[44] Xiaobing Xue, Jiwoon Jeon, and W Bruce Croft. 2008. Retrieval models for question and answer archives. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2008)*. ACM, 475–482.

[45] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. 2018. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. *International Conference on Learning Representations (ICLR 2018)* (2018).

[46] Kai Zhang, Wei Wu, Haocheng Wu, Zhoujun Li, and Ming Zhou. 2014. Question Retrieval with High Quality Answers in Community Question Answering. *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM 2014)* (2014), 371–380.

[47] Minghua Zhang and Yunfang Wu. 2018. An Unsupervised Model with Attention Autoencoders for Question Retrieval. In *Proceedings of the ThirtySecond AAAI Conference on Artificial Intelligence (AAAI 2018)*. 4978–4986.

[48] Wei Emma Zhang, Quan Z Sheng, Jey Han Lau, Ermyas Abebe, and Wenjie Ruan. 2018. Duplicate Detection in Programming Question Answering Communities. *ACM Transactions on Internet Technology (TOIT)* 18, 3 (2018), 37.

[49] Guangyou Zhou, Li Cai, Jun Zhao, and Kang Liu. 2011. Phrase-Based Translation Model for Question Retrieval in Community Question Answer Archives. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2011)*. Association for Computational Linguistics, 653–662. http://aclweb.org/anthology/P11-1066

TurkScanner: Predicting the Hourly Wage of Microtasks

Susumu Saito
Waseda University
Tokyo, Japan
susumu@pcl.cs.waseda.ac.jp

Chun-Wei Chiang
West Virginia University
Morgantown, WV, USA
cc0051@mix.wvu.edu

Saiph Savage
Universidad Nacional Autonoma de Mexico (UNAM)
Mexico City, Mexico
saiph.savage@mail.wvu.edu

Teppei Nakano
Waseda University
Tokyo, Japan
teppei@pcl.cs.waseda.ac.jp

Tetsunori Kobayashi
Waseda University
Tokyo, Japan
koba@pcl.cs.waseda.ac.jp

Jeffrey P. Bigham
Carnegie Mellon University
Pittsburgh, PA, USA
jbigham@cs.cmu.edu

ABSTRACT

Workers in crowd markets struggle to earn a living. One reason for this is that it is difficult for workers to accurately gauge the hourly wages of microtasks, and they consequently end up performing labor with little pay. In general, workers are provided with little information about tasks, and are left to rely on noisy signals, such as textual description of the task or rating of the requester. This study explores various computational methods for predicting the working times (and thus hourly wages) required for tasks based on data collected from other workers completing crowd work. We provide the following contributions. *(i)* A data collection method for gathering real-world training data on crowd-work tasks and the times required for workers to complete them; *(ii)* TurkScanner: a machine learning approach that predicts the necessary working time to complete a task (and can thus implicitly provide the expected hourly wage). We collected 9,155 data records using a web browser extension installed by 84 Amazon Mechanical Turk workers, and explored the challenge of accurately recording working times both automatically and by asking workers. TurkScanner was created using ~150 derived features, and was able to predict the hourly wages of 69.6% of all the tested microtasks within a 75% error. Directions for future research include observing the effects of tools on people's working practices, adapting this approach to a requester tool for better price setting, and predicting other elements of work (*e.g.*, the acceptance likelihood and worker task preferences.)

CCS CONCEPTS

• **Human-centered computing → Systems and tools for interaction design**.

KEYWORDS

crowdsourcing; Amazon Mechanical Turk; hourly wage

ACM Reference Format:
Susumu Saito, Chun-Wei Chiang, Saiph Savage, Teppei Nakano, Tetsunori Kobayashi, and Jeffrey P. Bigham. 2019. TurkScanner: Predicting the Hourly Wage of Microtasks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313716

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313716

1 INTRODUCTION

Workers on crowd platforms struggle to earn adequate wages [18, 20, 27]. This is problematic, given that one of the main motivators for crowd workers is to earn sufficient money to make a living [4, 5, 25, 26]. It is usually difficult for workers to earn higher than minimum wage, because crowd markets provide limited information on the offered microtasks. Workers struggle to gauge how much time a task will require, and to make informed judgements on whether tasks are worth their time. In this study, we introduce a machine learning approach to estimate the time required to complete a task and the approximate hourly wage of the task.

Our work builds on recent work investigating crowd work wages. Callison-Burch et al. developed the CrowdWorkers browser extension [6], which records working times on Amazon Mechanical Turk (AMT). An analysis of this data revealed that the median hourly worker wage was only 2 USD/h [12]. This is significantly lower than the U.S. minimum wage (7.25 USD/h). TurkBench leveraged historical records of completions of specific microtasks (called HITs on AMT) to suggest lucrative work to workers [11], but this was limited by which tasks had previously be seen. TurkBench could not estimate the hourly wage the first time that a particular type of task was performed. New tasks are frequently posted in crowd marketplaces, and so it is important to be able to predict hourly wages for previously unseen microtasks.

In this paper, we present TurkScanner, a machine learning approach for predicting the working times of microtasks to calculate their hourly wages based on previous logs of other workers for other tasks. This allows workers to judge whether microtasks are worth doing, even for new tasks that no other workers have completed.

The first challenge we addressed in building TurkScanner was to collect reliable ground truth data on the working times of real tasks. Estimating working times automatically is difficult, because we know that worker behavior patterns during microtasks and the motivations behind them are diverse [23], such as visiting external websites to complete the tasks, taking breaks, or accepting a number of tasks and completing them in a row [12]. Rather than attempting to calculate a single working time, we collected three different times (two types of automatic recording and manual recording by

the worker), along with the workers' a posteriori judgments on which were likely to be most accurate. Our extension collected 9,155 data records of microtask submissions from 84 unique workers. For each worker, we collected information on each of the tasks they completed, including the task (HIT) metadata and HTML content, the reputation of the requester, and the worker profile. We aimed to collect all of the data that workers would view before actually completing a task. Intuitively, we expected results such as "HITs with longer times provided in their metadata may take longer," "HITs posted by requesters with better ratings pay better," and "HITs that included many input elements take longer to complete."

Our second challenge was to predict working times (and thus hourly wages) for microtasks using a machine learning-based approach. To the best of our knowledge, TurkScanner represents the first work to utilize machine learning to estimate the working times of microtasks. We extracted ~150 features from our data collected from AMT. Our cross-validated results showed that TurkScanner achieved hourly wage predictions within a 75% working-time error for 69.6% of all the microtasks (and within a 100% error for 84.3%).

We conclude this paper by suggesting future research directions, to apply TurkScanner to support tools that empower both workers and requesters. We believe these types of machine learning mechanisms will enable a future in which crowd work becomes a more transparent and beneficial activity for all stakeholders.

2 RELATED WORK

A wealth of previous literature has shown that workers are underpaid and unfairly treated in crowd markets [15, 17, 24]. Most crowdsourcing platforms allow requesters to freely create their microtasks and set their prices. Requesters can also assess worker performances to control the quality of their answers, such as by screening workers' eligibility to work with qualifications [12], or by rejecting submitted tasks that do not meet their criteria [3]. On the other hand, workers are usually provided with very limited information to select better microtasks [20]. For instance, most crowd markets only provide the task price, the name of the requester, a title, and a simple description. Such a lack of information makes it very challenging for workers to find good microtasks and requesters. This power imbalance between requesters and workers limits workers' ability to ensure that they receive fair wages, and this results in many workers being paid below minimum wage [14, 16, 18, 20, 21, 26, 27]. This problem was formalized after Hara et al. recently revealed that AMT workers earn a median hourly wage of only 2 USD/h, based on their data-driven analysis [12]. This surprising fact clearly emphasizes the necessity of tools to help crowd workers earn better wages.

Several researchers and practitioners have explored approaches to provide workers with better opportunities to obtain information on microtasks when selecting tasks [27]. There are several web-based platforms on which workers communicate with each other about microtasks and requesters. For instance, Turkopticon [20] is an online forum and a tool where workers post reputation scores of requesters concerning several criteria, as well as free comments. Among other support tools, Turkopticon is highly appreciated and actively utilized by workers [23]. Many workers also join online social communities, such as Turker Nation [1] and MTurk Crowd [2]. On these community websites, workers not only share reputations, but also recommend specific tasks and introduce tools and skills utilized by expert workers. Web tools are also available to workers to directly facilitate crowd work within the crowd market [8]. CrowdWorkers [6] is a web browser extension for AMT workers, which collects microtask submission logs and visualizes the average hourly wage for each posted task. TurkBench [11] is another web platform for AMT workers, which renders a personalized, adaptive working schedule based on collected task logs, to suggest the most lucrative microtasks.

Like other tools, we aim to help workers select better microtasks. More specifically, the goal of TurkScanner is to "predict" the hourly wages of new unseen microtasks, based on other tasks completed by other workers, rather than by calculations based on worker records for the same task. We not only contribute the prediction algorithm, but also address the challenge of defining the "working time" of a microtask, so that we can calculate hourly wage based on this, which has not been sufficiently discussed in prior work.

3 METHODS

In this section, we first describe our approach to defining the "working time" of a microtask. Next, we explain our browser extension for data collection, designed based on our working time definitions. We then explain the features of the dataset and its statistical analysis results. Finally, we present the details of TurkScanner, which predicts the working times of microtasks and their hourly wages.

3.1 Defining Working Time

The "working time" refers to the amount of time required for workers to complete particular tasks. To predict the working time using machine learning, it is essential to collect real-world data on actual working times. However, measuring the working times is highly challenging. We know that various worker behavior patterns exist, which prevents us from stably obtaining the correct working time using simple methods. During microtasks, workers often browse external websites in other tabs (either related or unrelated to the task), take breaks, work on multiple tasks in parallel, and so on.

In our approach, we recorded three different types of working time for every microtask. Each of these has unique advantages and disadvantages. When workers submitted microtasks, they were subsequently asked to choose one of the working times to finally label the collected data with. The options were as follows:

- **TIME_ALL**. Automatic recording method, consisting of the time from when workers accept the task until they finish it. *Pros*: This is the most reliable working time when a worker immediately starts and completes a task without taking a break. *Cons*: All of the time during which the worker is distracted (by checking emails, getting coffee, etc.) is counted.
- **TIME_FOCUS**. An automatic recording method similar to TIME_ALL, but only recording the time during which the microtask page tab is in focus. *Pros*: This is a reliable working time even if a worker is distracted by task-irrelevant content in other tabs. *Cons*: This approach cannot count the working time for microtasks that guide workers to other tabs at which the actual task resides (*e.g.,* surveys or Google searches).

- **TIME_BTN**. A manual recording method carried out by workers themselves, by toggling buttons to indicate when they are in the working status. *Pros*: This is the most accurate working time when a worker utilizes the button correctly, covering all possible worker behavior patterns. *Cons*: This approach is vulnerable to human error (*e.g.,* spams).

For workers' final decisions, we also introduced the following working-time type as a fourth option:

- **TIME_CUSTOM**. Manual input for the working time by workers in the case that none of the above three options seem to be correct. *Pros*: This can provide workers with a last-resort option, to label the correct working time when all recording methods failed. *Cons*: Errors may be present in workers' answers.

We proposed a few hypotheses concerning workers' judgments of the working time. We expected the most dominant choice would be TIME_BTN, because we anticipated that most workers would browse external websites or temporarily leave their computers for diverse reasons, and that the working time manually recorded by themselves would look more reliable in most cases. Next, we supposed that TIME_ALL would be the second most dominant choice. However, we expected that this would still be chosen less than TIME_BTN, because there are a certain number of workers who interleave tasks on crowdsourcing platforms [23], such that the working time cannot be correctly measured by this approach. Finally, we expected that TIME_FOCUS and TIME_CUSTOM would be chosen considerably less, but would still be useful when workers accidentally ignored the recording button.

3.2 Data Collection With Web Browser Script

We developed a Google Chrome extension for data collection, which crawls data from every completed microtask together with the actual working time. The extension records the working time automatically, provides AMT workers with a button for manually recording the working time, and asks workers to select the best choice for the working time with which to finally label the data.

3.2.1 Worker Recruitment with Pre-Survey. To recruit AMT workers, we prepared a pre-survey that asks for worker profile information (*e.g.*, gender, age, country, household income, worker experience, and worker hours per week). After the survey, we asked workers to install our extension according to instructions and a URL for the installation page. After installation, each participant was given a unique "installation code," to be copy-and-pasted back into the HIT to verify that they both installed the extension and completed the task. The survey took about 4 min to complete, including the extension installation. We paid workers 0.60 USD (*i.e.,* an expected 9 USD/h) to complete the pre-survey and correctly paste the installation code.

3.2.2 HIT Crawling using Chrome Extension. When workers finished installing the extension, they were ready to begin data collection. Workers were asked to work on microtasks as usual. The extensions collected data for up to 10 days after being installed, yet allowing the workers to uninstall it at any time (the workers were paid in proportion to their contribution until uninstallation). The data collection proceeded according to the three following steps (a), (b), and (c) (with workers paid a 5-cent bonus for completing each of (b) and (c) for each a HIT):

Table 1: List of input features parsed from the collected data. The features consist of three categories and 12 subcategories. The parenthesized numbers in bold text represent the feature dimension sizes.

HIT (71) – HIT-relevant information		
META	**(3)**	HIT metadata set by requesters (*e.g.,* reward, # of available HITs)
TMPL	**(11)**	HIT templates natively provided by AMT as of October 2018 (one-hot vector)
URL	**(6)**	URL counts per content type included in a HIT page (*e.g.,* anchor links, images, audios)
INP	**(18)**	Input tag counts per type included in a HIT page ("type" attributes such as *radio* and *text*)
TXT	**(1)**	Visible word counts in a HIT page
KW	**(32)**	Task-relevant keyword occurrence in either HIT title, HIT description, or a HIT page (keywords such as "survey" and "summarize" arbitrary extracted by the authors)
WKR (28) – Worker-relevant information		
PRFL	**(16)**	Worker profile information collected in pre-surveys (*e.g.,* age, worker experience in years, est. hourly wage)
EXT	**(8)**	AMT worker helper extension tools installed in web browser (*e.g.,* CrowdWorkers [6], MTurk Suite [13])
HIST	**(4)**	Working history information as workers (*e.g.,* approval rate, total earnings, # of HIT submission in a HIT group)
REQ (49) – Requester reputation information		
TO	**(7)**	Turkopticon [20] (*e.g.,* average of 5-point scale requester evaluation of generosity, promptness, etc.)
TO2	**(34)**	Turkopticon 2 [21] (*e.g.,* average of 5-point scale HIT evaluation of recommendability, communicativeness, etc.)
TV	**(8)**	TurkerView [9] (*e.g.,* requester reputation for hourly wage settings, # of reviews, etc.)

(a) Background data scraping. Once workers installed the extension, they were asked to select and work on any HIT as they would usually do on the platform. The extension recorded the following data features, as shown in Table 1. When a worker visited a HIT page, the extension scraped HIT-relevant information (`HIT`), such as the HIT metadata and HTML contents of the HIT page (*i.e.,* the text/media content information) that would possibly define task goals and required interactions in the task. In addition, the extension crawled worker-related information (`WKR`) from the worker dashboard and the list of installed AMT-relevant extensions once per day. `WKR` would also be important, because its features would represent worker capabilities for microtasks [28], such as their skill levels and the learning-curve effects [29]. Our extension also obtained requester-relevant information (`REQ`) concerning the reputations of requesters, provided via RESTful APIs in some third-party platforms (*i.e.*, Turkopticon [19], Turkopticon 2 [22], and TurkerView [9]). This would help to understand how reasonably the microtask would be paid based on the requester ratings.

(b) Manual working time recording. While working on HITs, workers were asked to manually record the working time they spent on each HIT. When workers accepted HITs, a red button was shown on the project details bar located at the top of a HIT page (Figure 1a). Workers recorded times at which they paused and resumed working on microtasks by toggling the button. The two following additional

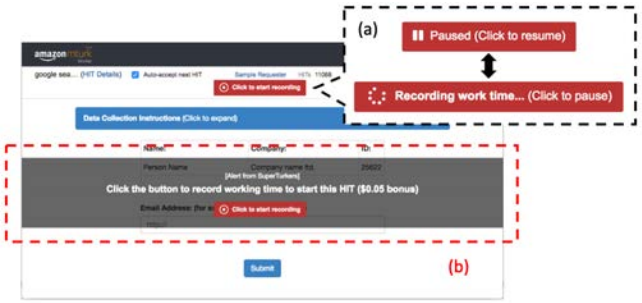


Figure 1: Interface to record TIME_BTN. (a) The button at the top of the HIT page can be toggled to pause/resume recording working time. (b) A black screen is rendered over the HIT at the beginning as a reminder workers to start the timer.

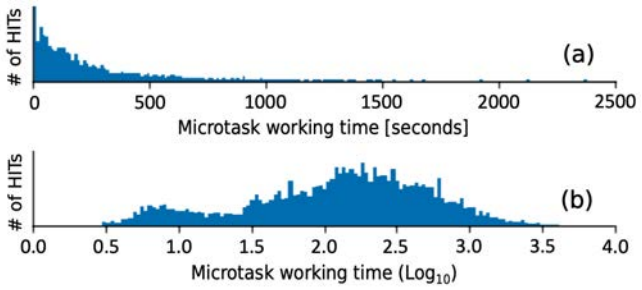


Figure 2: Working time distribution of microtasks in the dataset. (a) Working time in seconds, which admits a long-tail distribution. (b) log_{10} working time, which can suppress the prediction error due to outliers.

features were implemented to remind workers to click the button when they start working on the HIT. First, workers were prompted with a black screen rendered over the page (Figure 1b). This screen disappeared immediately once they clicked the button. This made it less likely for workers to complete a HIT without dismissing the alert. Second, a red frame was displayed around the window while the button was in the recording state, to assist workers in recognizing the recording state. When workers interleaved HITs, (*i.e.*, worked on multiple HIT pages in multiple tabs simultaneously, by moving back and forth), none of the HITs would record their working times simultaneously, because we assumed that perfect multi-tasking of HITs is not possible.

(c) Post-HIT survey. As soon as workers submitted a HIT, they were prompted with a "Post-HIT survey" in a popup window and they were asked to select the most accurate working time from the various methods with which to label HIT records. The survey suggested four options for the HIT working time: TIME_ALL, TIME_FOCUS, TIME_BTN, and TIME_CUSTOM (see Section 3.1 for details). Workers were required to input working time, in the form of X minutes and X seconds, only when they chose TIME_CUSTOM. The pop up window disappeared immediately after workers submitted their choices by clicking a “Submit” button.

3.3 Data Description

Data collection was conducted for 10 days during late October 2018. Our task dataset consisted of 9,155 HIT submission records collected by 84 unique workers. The recorded HITs belonged to 1,641 unique HIT groups, posted by 998 unique requesters. On average, workers contributed for 6.5 days (SD = 3.5; Median = 8.1) and worked on 109 HITs (Min = 1; Max = 1,958; SD = 238.1; Median = 34).

Data Cleaning. To guarantee the quality of the dataset, we filtered out the following HIT submission records: *(i)* All HIT records of spam workers who used automated scripts (N=230); *(ii)* HIT records for which the reported working time was abnormally short or long (N=213); *(iii)* A large part of all the HIT records in the top three most-submitted HIT groups, keeping only as many HIT records as in the fourth largest HIT group, in order to reduce bias in the dataset (N=1,104). After the data cleaning, 7,608 (83.1%) HITs remained from 83 unique workers, for 1,587 HIT groups posted by 977 unique requesters. This dataset was utilized for the analysis in the remainder of the paper.

Working Time Labels. On average, the reported working time was 277.9 s (SD = 380.2; Median = 148.3). As in Figure 2(a), the distribution of the working time had a long-tail; 69.0% of all the working times were below the average. The hourly wage averaged $9.15 (SD = 29.11; Median = 4.23). Among all HIT records, 2,270 (29.8%, in 674 HIT groups (42.4%)) were above the U.S. minimum wage (7.25 USD).

For the final labels for the working time, workers chose TIME_ALL for 3,802 (50.0%) HIT records, TIME_BTN for 2,525 (33.2%) records, TIME_FOCUS for 748 (9.8%) records, and TIME_CUSTOM for the rest 533 (7.0%) records. This partly supported our hypotheses, in that TIME_FOCUS and TIME_CUSTOM occurred less than the others, but went against our expectation that TIME_ALL would occur more than TIME_CUSTOM by 16.8 points.

We then conducted a follow-up analysis on TIME_BTN. Our investigation into the workers’ button usage showed that the button was clicked at least once in 7,037 (92.5%) HITs, that the usage per worker averaged 95.8%, and that merely 16 workers (19.3%) were below the average. The button appeared to be correctly utilized in most cases. However, the analysis also revealed that participants seldom stopped the timer. The timer was stopped once in 73 HITs, twice in 8 HITs, and three times or more in 8 HIT,s while the timer was never stopped in the remaining 6,948 HITs. We then analyzed the differences between TIME_BTN and TIME_ALL for microtasks which TIME_BTN was chosen for the final choices. The results showed that the differences in 72.5% of the records were less than 5 s, and those in 85.1% were less than 10 s. The results indicate that most of the TIME_BTN records that were chosen for the final labels were nearly equal to TIME_ALL, (*i.e.,* the button was immediately clicked when tasks started and finished without taking a break). On the other hand, it was not possible for us to rigidly evaluate whether the button was correctly used, because our collected dataset did not contain tracking information of workers during HITs. We leave this as a topic for future work.

3.4 TurkScanner: Hourly Wage Prediction

TurkScanner predicts the hourly wages of microtasks in two steps: 1) estimate the working times of microtasks based on `HIT`, `WKR`,

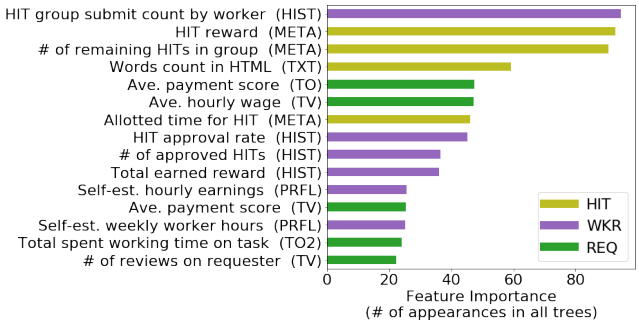


Figure 3: Top-15 feature importance rankings. Features for batch size, submission counts, and some explicit features (*i.e.,* price, time, microtask contents) seem to implicate working time length. Worker profiles are possibly effective, representing workers' proficiency on tasks.

and REQ, in a machine learning-based approach; 2) calculate the hourly wage from the rewards and the estimated working times.

We predicted the working times of microtasks through regression using gradient boosted decision tree (GBDT) [10]. The model was trained with 148-dimensional feature vectors of task-relevant information (see Section 3.2), by minimizing the mean absolute error between the predicted and actual working times. Upon training and testing, our GBDT model outputs the working times of microtasks on a *log scale (base=10)*. As shown in Figure 2, the working times of microtasks in our dataset admit a long-tail distribution. Taking the logarithm prevents the model from being excessively optimized for short-length microtasks and being affected by outliers.

The model evaluation was conducted through four-fold cross validation. When partitioning the dataset, we picked 25% of the 83 workers, and used all their HIT submission records for the test set. This means that the same worker never belonged to both the training and test sets. Therefore, the validation results indicate the extent to which the model is capable of predicting the working time without being trained using HIT submission records of the same worker. To obtain the predicted working times for all the microtasks in the dataset, we tested the model for all the validation pairs, and analyzed them all together. To prevent the results from being too dependent in each trial, we iterated training and testing 50 times, and then calculated the average working time for each HIT record to obtain the results for subsequent analysis.

4 RESULTS AND DISCUSSION

In this section, we describe the evaluation results of TurkScanner. We first analyze feature importance, followed by analyses of the prediction accuracy for the working time and hourly wage.

4.1 Feature Importance

We first measured the feature importance, to better understand how each feature dimension contributes to predicting the working time. Among the diverse methods available to measure the feature importance, we selected "weight" provided by XGBoost [7], which counts how many times a feature is utilized for splitting across all generated trees. We iterated the training of the initial model 50 times, and then took the averages of the feature importance values for the following analysis.

Figure 3 visualizes the importance ranking of the features for the working time prediction. Features that could provide indications to estimate the microtask size appeared to be especially important. For instance, larger HIT group submission count numbers (first place in the ranking), the number of remaining HITs in the group (third), and the word count in the HTML would imply how quickly HITs can be completed, because shorter HITs are more likely to be performed repeatedly. More intuitively, features concerning the price (HIT reward: second), time (time allotted for a HIT: seventh), and task contents (word count in HTML: fourth) could also help to represent the working time. These features also appeared to be effective as reputations by workers (REQ features: fifth, sixth, 12th, 14th, and 15th). Worker profiles and task submission histories were also thought to be effective (WKR features: 8th–11th and 13th). All these features represent workers' proficiencies with their working hours, accepted HITs, and earned rewards. This could possibly help to ensure the reliability of the working time prediction.

4.2 Working Time Prediction

Figure 4 presents a heat map of the confusion matrix representing the distribution of the working time prediction results per working time bin. The size of each bin increases gradually towards the right, where the bin of the leftmost columns gathers all the HIT records whose actual working times are between 3 and 8 s, whereas that of the second right-most column includes those between 600 and 1,200 s. Note that we only utilized this binning rule for the analysis: TurkScanner outputs consist of float values for the working time.

The heat map indicates that a large proportion of the predicted working times hit the bin or a nearby bin. Seventeen percent of all the HIT submission records are in the diagonal cells in the heat map, surrounded by bold grid lines, (*i.e.*, they are categorized in the correct bins). Allowing the prediction results to be categorized into neighboring bins, the proportion of correct predictions increases to 47.4% with a one-cell difference (indicated by thin grid lines), and 70.8% with a two-cell difference (indicated by dotted lines). We also note that the predicted working times of HIT records with shorter working time labels (less than 60 s) are likely to be longer. Likewise, the predicted working times of HITs with longer working time labels (more than 600 seconds) tend to be shorter. This may be because the distribution of the logarithmic working time labels is closer to the normal distribution (see Figure 2b). As mentioned in Section 3.4, the objective function of the GBDT algorithm trains the model such that the overall error across all the data is minimized. Therefore, the model may have been trained to reduce the prediction error for HIT records with medium-length working times, where the largest amount of data samples are.

4.3 Hourly Wage Calculation

We calculated the hourly wage using the rewards and predicted working times of microtasks. Over all the tested HIT records, the predicted hourly wage averaged 5.21 USD (SD = 4.53; Median = 4.20). For N = 5,297 (69.6% of all the collected HIT records) the hourly wage was predicted within a 75% error, and for N = 6,412 (84.3% of all the records) it was predicted within a 100% error.

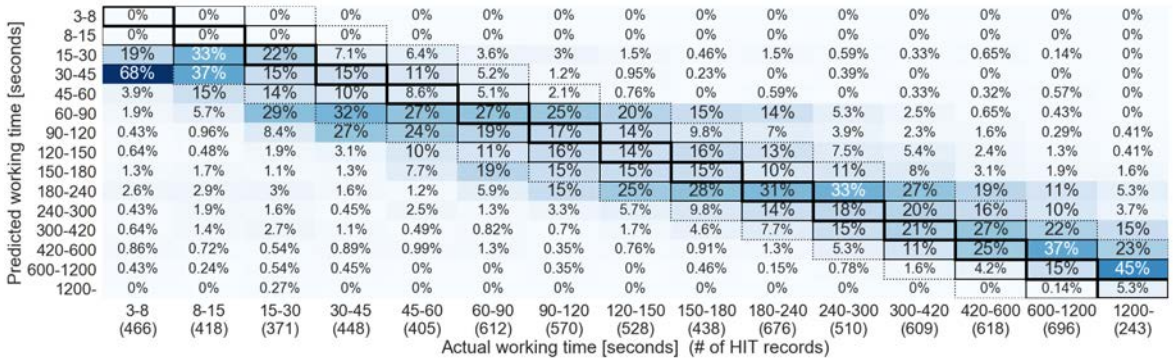


Figure 4: Working time prediction results in a confusion matrix, illustrated by a heat map. A large portion of the prediction results are distributed diagonally, which implies that the model successfully captured the trend in the working time prediction.

The prediction performed reasonably well for HIT records with actual hourly wages lower than around 15 USD. On the other hand, many of the HIT records with higher hourly wages were not predicted to be as high as they actually were. To further analyze the incorrect results, we directly inspected the corresponding HIT records. As a result, we determined that most of these were *(i)* survey HITs with external URL(s) and *(ii)* microtasks for which contents were dynamically rendered with JavaScript. These two types of HITs are very similar in that they do not have much static HTML content by themselves. Because we revealed in our feature analysis that some types of HTML content (*e.g.*, text counts in the HIT page, URL counts, and input tags) affected the prediction results, the model might have not been able to predict these HITs accurately.

5 LIMITATIONS AND FUTURE WORK

5.1 Limitations

We first describe limitations concerning the data cleaning described in Section 3.3. Our data cleaning method could be improved by scraping additional information, so that more noisy data are removed from the dataset. First, tracking the statuses of completed microtasks (*i.e.,* accepted or rejected) could filter out unreliable data. If microtasks are rejected, then some of the scraped data are likely to be inconsistent. For instance, the working time may be too short compared to the task. Second, we could track worker behavior such as cursor movements, scrolling, and the keyboard input for a certain time window. Using such information, we can verify whether the history of recorded working times corresponds to the user behavior, and remove records if not.

We could also collect a more refined dataset by tracking microtask bonuses. TurkScanner only considered fixed rewards for microtasks, and bonuses were not taken into account for the hourly wage prediction. This would underestimate the value of microtasks whose reward schemes largely depend on bonuses for the microtask content. Tracking the sizes of bonuses paid for each microtask would help TurkScanner to build a more accurate prediction model.

5.2 Future Work

Our next step will be to develop TurkScanner as a support tool for workers and requesters to solve real-world problems. We will observe how TurkScanner could help workers reach better decisions on which microtasks to start. We will also incorporate aspects that can be found in other similar recommender systems, such as predicting HIT acceptance rates, or recommending HITs based on workers' task preferences. TurkScanner could also assist requesters with performing better price setting. It could predict the hourly wage for microtasks prior to posting on the platform, and suggest that requesters increase their rewards for better wages. This would greatly contribute to improving the average microtask price in the market, and accelerate future crowdsourcing research.

More generally, TurkScanner represents a new approach to understanding and predicting the times required for arbitrary user interface tasks. This could be utilized in a variety of settings, such as setting wages in other domains, helping people to better organize or schedule their time, or in automated usability testing. One can imagine a scenario in which every new task a government or company posts comes accompanied with an estimate of the cost of introducing the new task. It may still be required to fill out an extra web form to justify conference travel, but at least it would be explicitly known how much time (and thus money) would be required to do so.

6 CONCLUSION

In this study, we tackled the challenge of predicting the hourly wages of microtasks based on data collected from previous workers. We first presented a data collection method with our web browser extension for gathering data about crowd work and labeling the data with accurate working times. We asked workers to select their answers from choices of working times recorded either automatically and manually by the workers themselves. We then proposed TurkScanner, a system based on the GBDT regression model, to predict working times, and thus calculate hourly wages as its final output. Our evaluation results indicated TurkScanner would need further improvement on its prediction performance. Nonetheless, we clearly showed the possibility that workers can know whether their crowd work will be worth the pay before actually embarking on it, which would make crowd work more transparent and beneficial for workers and requesters.

REFERENCES

[1] [n. d.]. Turker Nation. Retrieved November 3, 2018 from https://www.reddit.com/r/turkernation/

[2] 2016. MTurk Crowd. Retrieved November 3, 2018 from https://www.mturkcrowd.com/

[3] Benjamin B Bederson and Alexander J Quinn. 2011. Web workers unite! addressing challenges of online laborers. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*. ACM, 97–106.

[4] Janine Berg. 2015. Income security in the on-demand economy: Findings and policy lessons from a survey of crowdworkers. *Comp. Lab. L. & Pol'y J.* 37 (2015), 543.

[5] Robin Brewer, Meredith Ringel Morris, and Anne Marie Piper. 2016. Why would anybody do this?: Understanding older adults' motivations and challenges in crowd work. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2246–2257.

[6] Chris Callison-Burch. 2014. Crowd-workers: Aggregating information across turkers to help them find higher paying work. In *Second AAAI Conference on Human Computation and Crowdsourcing*.

[7] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 785–794.

[8] Chun-Wei Chiang, Anna Kasunic, and Saiph Savage. 2018. Crowd Coach: Peer Coaching for Crowd Workers' Skill Growth. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 37.

[9] ChrisTurk. 2018. TurkerView. Retrieved November 4, 2018 from https://turkerview.com/

[10] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[11] Benjamin V Hanrahan, Jutta K Willamowski, Saiganesh Swaminathan, and David B Martin. 2015. TurkBench: Rendering the market for Turkers. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 1613–1616.

[12] Kotaro Hara, Abigail Adams, Kristy Milland, Saiph Savage, Chris Callison-Burch, and Jeffrey P Bigham. 2018. A Data-Driven Analysis of Workers' Earnings on Amazon Mechanical Turk. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 449.

[13] Brandon Hellman. 2017. MTurk Suite. Retrieved November 3, 2018 from http://mturksuite.com/

[14] Paul Hitlin. 2016. Research in the crowdsourcing age, a case study. *Pew Research Center* 11 (2016).

[15] John J Horton. 2011. The condition of the Turking class: Are online employers fair and honest? *Economics Letters* 111, 1 (2011), 10–12.

[16] John Joseph Horton and Lydia B Chilton. 2010. The labor economics of paid crowdsourcing. In *Proceedings of the 11th ACM conference on Electronic commerce*. ACM, 209–218.

[17] International Labour Office (ILO). 2016. Non-standard employment around the world: Understanding challenges, shaping prospects.

[18] Panagiotis G Ipeirotis. 2010. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students* 17, 2 (2010), 16–21.

[19] Lilly C Irani and M Silberman. 2013. Turkopticon. https://turkopticon.ucsd.edu/

[20] Lilly C Irani and M Silberman. 2013. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 611–620.

[21] Lilly C Irani and M Silberman. 2016. Stories we tell about labor: Turkopticon and the trouble with design. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. ACM, 4573–4586.

[22] Lilly C Irani and M Silberman. 2017. Turkopticon 2. https://turkopticon.info/

[23] Toni Kaplan, Susumu Saito, Kotaro Hara, and Jeffrey P Bigham. 2018. Striving to Earn More: A Survey of Work Strategies and Tool Use Among Crowd Workers.. In *HCOMP*. 70–78.

[24] Miranda Katz. 2017. Amazon Mechanical Turk Workers Have Had Enough. https://www.wired.com/story/amazons-turker-crowd-has-had-enough/

[25] Siou Chew Kuek, Cecilia Paradi-Guilford, Toks Fayomi, Saori Imaizumi, Panos Ipeirotis, Patricia Pina, and Manpreet Singh. 2015. The global opportunity in online outsourcing. (2015).

[26] David Martin, Benjamin V Hanrahan, Jacki O'Neill, and Neha Gupta. 2014. Being a turker. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. ACM, 224–235.

[27] Brian McInnis, Dan Cosley, Chaebong Nam, and Gilly Leshed. 2016. Taking a HIT: Designing around rejection, mistrust, risk, and workers' experiences in Amazon Mechanical Turk. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. ACM, 2271–2282.

[28] Jeffrey M Rzeszotarski and Aniket Kittur. 2011. Instrumenting the crowd: using implicit behavioral measures to predict task performance. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 13–22.

[29] Louis E Yelle. 1979. The learning curve: Historical review and comprehensive survey. *Decision sciences* 10, 2 (1979), 302–328.

A Large-scale Study of Wikipedia Users' Quality of Experience

Flavia Salutari
Telecom ParisTech
flavia.salutari@telecom-paristech.com

Diego Da Hora
Telecom ParisTech
diego.hora@gmail.com

Gilles Dubuc
Wikimedia Foundation
gilles@wikimedia.org

Dario Rossi
Huawei Technologies, Co. Ltd
dario.rossi@huawei.com

ABSTRACT

The Web is one of the most successful Internet application. Yet, the quality of Web users' experience is still largely impenetrable. Whereas Web performances are typically gathered with controlled experiments, in this work we perform a large-scale study of one of the most popular websites,namely Wikipedia, explicitly asking (a small fraction of its) users for feedback on the browsing experience. We leverage user survey responses to build a data-driven model of user satisfaction which, despite including state-of-the art quality of experience metrics, is still far from achieving accurate results, and discuss directions to move forward. Finally, we aim at making our dataset publicly available, which hopefully contributes in enriching and refining the scientific community knowledge on Web users' quality of experience (QoE).

CCS CONCEPTS

• **Information systems → World Wide Web**; • **Networks → Network measurement**; *Network performance analysis.*

KEYWORDS

Web Performance; Quality Of Experience; Web Browsing QoE

ACM Reference Format:
Flavia Salutari, Diego Da Hora, Gilles Dubuc, and Dario Rossi. 2019. A Large-scale Study of Wikipedia Users' Quality of Experience. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313467

1 INTRODUCTION

Since its inception, the World Wide Web has sometimes been dubbed as World Wide "Wait" [9]. Slow rendering of webpages happened due to dial-up connections in the 80s, slow 2G connections in the 90s and so on, but it also persists nowadays for several reasons including unexpected sources of latencies [16], interactions between network protocols [22], the growingly more complex structure of websites [45], an increased usage of mobile devices [19, 34] and the emergence of new protocols [40]. Yet, whereas the study of Web performance is commonly [19, 22, 29, 34, 40, 45–47] tackled via simple objective metrics [20], and rather typically the Page Load Time (PLT), the quality of Web users' experience is still largely impenetrable [15, 28]. As such, a number of alternative metrics that attempt at better fitting the human cognitive process (such as SpeedIndex, user-PLT etc.) have been proposed as a proxy of QoE.

At the same time, studies involving more advanced metrics are typically validated with rather small-scale experiments, either with a small number of volunteers, or by using crowdsourcing platforms to recruit (cheap) labor and produce a dataset labeled with user opinion. Often, *videos* of Webpages rendering process is used (as opposite to actual browsing), with possibly very specific instruction (e.g., such as in A/B testing, by clicking on the fastest of two rendering processes) that are however rather different from the cognitive process in action during the typical user browsing activities. Additionally, such tests are carried on a limited number of fixed conditions, with a small heterogeneity of devices, OSs and browsers, and are not exempt from cheating so that ingenuity is needed to filter out invalid answers from the labeled dataset [24, 44]. Finally, because these tests are carried on a limited number of pages, it is possible to evaluate computationally costly metrics, such as those that require processing the visual rendering of the webpage, which would hardly be doable in the World "Wild" Web.

Our aim is instead to take a completely different approach and perform a large-scale study of a popular website in operation, by explicitly asking a fraction of users for feedback on their browsing experience. Clearly, the approach is challenging but it opens the possibility to gather more relevant user-labels, as they are issued from *real users of a real service*, as opposite to crowdworkers payed to play a game (e.g., find which video completes first as in A/B testing). We do so by launching a measurement campaign over Wikipedia, that at time of writing has gathered over 62k survey responses in nearly 5 months. We complement the collection of user labels with objective metrics concerning the user browsing experience (ranging from simple PLT [20] to sophisticated SpeedIndex [1]), and harvest several data sources to further enrich the dataset so that each user survey answer is associated with over 100 features. Summarizing our main contributions:

- first, we use survey data to characterize user satisfaction, finding that on average 85% of users are satisfied;
- second, we build a supervised data-driven model of user experience: despite our features include state-of-the art quality of experience metrics, we find that they fall short to model user QoE in operational settings;
- third, in spirit with research reproducibility, we plan to release both the collected *dataset* and our developed *code* as

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313467

open-source, as we hope this can help the scientific community in refining its understanding of Web users' experience.

After overviewing the related work (§2), this paper describes our feedback collection process and dataset (§3), that we leverage to build a data-driven model of Wikipedia user experience (§4), finally discussing its current limits and directions to circumvent them (§5).

2 BACKGROUND

Assessment of Web users' quality of experience can be traced back to [36], that was among the first to adapt classic results of psycho-behavioral studies gathered in the *computer* domain [30] (in turn inspired by work by Weber and Fechner in the late 1800s), to the *computer-network* domain. This knowledge was later embedded into standards ITU-T G1030 [26, 39] (and models [23]) that encode the Weber-Fechner logarithmic [26, 39] (or exponential [23]) relationship between a stimulus (e.g., a delay) and its perceived impact (e.g., nuisance for Web users). However, while logarithmic models are valid for simple waiting tasks (e.g., file downloads), the case of interactive Web browsing is knowingly much more complex, as ITU-T G1031 [27] and [21] first pointed out.

Still, with few exceptions [15, 18, 44, 48] most studies still rely on simple metrics such as the Page Load Time (PLT) to assess the expected impact of new Web protocols [22, 40, 45, 47], Web accelerators [29, 46] and devices [34, 37]. While reducing delay is clearly a desirable objective, it is however unclear if (and by how much) a latency reduction translate into a better perceived experience, which is the ultimate goal of the above studies. In other words, while the importance of *delay* in human perception is agreed upon, the exact relationship between the Web response time and user satisfaction appear much less clear than it appeared to be [33], and motivated a proliferation of new metrics proposals and validation studies attempting at going beyond PLT.

Web QoE metrics: As we are interested in measuring browsing experience on individual pages, *engagement* metrics such as those used in [11, 31] are clearly out of scope. As such, objective metrics of interest for Web user QoE can be divided in two classes. On the one hand, there are metrics that either *pinpoint precise time instants*: notable examples include the time at which the DOM is loaded or becomes interactive (TTI), the time at which the first element is painted (TTFP) or the time when the Above The Fold (ATF) portion of the page is rendered [14] etc. Most of these metrics are available from the browser navigation timing [20], and are easy to include (though not necessarily relevant) as proxy of user experience.

On the other hand, there are metrics that *integrate all events of the waterfall* representing the visual progress of the page, such as SpeedIndex [1] and variants [2, 12, 24], that have received significant attention lately. Denoting with $x(t) \in [0, 1]$ the visual completeness ratio of a page, metrics in the SpeedIndex family are defined as the integral of the residual completion $\int (1 - x(t))dt$ and differ in the way they express $x(t)$. Initial definitions in this family required capturing movies of the rendering process [1], or to further use similarity metrics SSim [24], making them difficult to use outside a lab environment. To counter this issue, simple approximations such as the ObjectIndex/ByteIndex [12] that merely count the fraction of objects/bytes received (over the total amount), or as the RUM SpeedIndex (RSI) [2] that use areas of rectangles for objects as they are painted on screen (over the total screen size) have been proposed. In this paper, we use RSI, which is among the most advanced Web QoE metrics considered to be the current industry standard. Finally, while we are aware that more complex approaches involving the spatial dimension (i.e., eye gaze) also exist [15, 28], we prefer to leave them for future work (cfr §5).

Metrics Validation: At the same time, the above metrics suffer from a limited validation with user feedback. Typical approaches are to crowdsource the validation with A/B testing [24, 44], or by performing experiments on real pages in controlled conditions [18, 33, 35]. Both approaches have their downsides. Controlled experiments with real HTTP server/clients and emulated network conditions for a more faithful and interactive browsing experience, but are harder to scale, topping to few hundreds users and few thousands data points [18]. A/B tests try to circumvent this limit, but introduce other limitations. First and foremost, A/B testing is hardly representative of Web browsing activity, since crowdworkers are instructed to select which among two videos, that they are passively screening side-by-side and that correspond to two different Web rendering processes, appears to finish first – whereas it is known that even for a simple Web browsing task such as information seeking, already different types of searches are rather different from the user standpoint in terms of cognition, emotion and interaction [32]. In other words, these experiments inform us that humans can perceive differences in these rendering processes, however they fail to signify if these perceptible rendering changes would impact the user satisfaction through the course of a normal browsing session.

The time at which users consider the process finished is denoted as user-PLT (uPLT)[44] or Time To Click (TTC) [24] and is often used as a ground truth of user perception. Yet, when users select a uPLT in [44], they are proposed with similar frames at earlier times, which has the beneficial effect of clustering answers and make uPLT more consistent at the price of possibly inducing a bias. Similarly, [24] employs SpeedIndex and TTC to forecast which among the left or right video was selected by the user at time TTC: the classifier in [24] is accurate in predicting which of the two videos is perceived as fastest by users. Yet, findings in [24] are not informative about whether the user would have been dissatisfied from the slower rendering had s/he been truly browsing.

Our contribution: To get beyond these limitations, in this work we are the first to query, at scale, Web users for their feedback on the quality of their browsing experience. which, to the best of our knowledge, has not been attempted before on the wide and wild Web. Instead of collecting user feedback on a 5-grade ACR scale, we ask for a (slightly more than) binary feedback (see §3), which let us formulate a simple (yet hard, see §4) binary classification problem.

Compared to recent literature , we are the first to involve a large number of real users ($62k$ from $59k$ distinct IP addresses) accessing a diverse set of pages ($46k$ Wikipedia pages, which are more likely similar among them than the set of different websites used in other studies), gathering over $62k$ user responses overall (more than twice the survey responses collected in similar large-scale Wikipedia studies [41]). Particularly, whereas most of the studies involving lab volunteers & crowdworkers employ a single browser and hardware (since crowdworkers are shown videos rendered with a single

browser and hardware combination) on a relatively small set of synthetic controlled network conditions, in our dataset we observe 45 distinct browsers software used on over 2,716 hardware devices on 3,827 ISPs – a significant change with respect to controlled lab conditions, which make the dataset of particular interest.

3 USER FEEDBACK COLLECTION

Wikipedia is, according to Alexa [3], the 5th most popular website, with over 1 billion monthly visitors, that spend over 4 minutes over 3 pages on average per day on the site. We engineer a survey that is triggered after the page ends loading and collects user feedback (§3.1), that we augment with additional information (§3.2).

We note that, while this paper is not the first in leveraging Wikipedia surveys in general (see e.g., [41]) this is the first to gather user feedback on quality of Web browsing experience from operational websites, for which we believe releasing the dataset can be valuable for the community. To make sharing of the dataset possible[1], we take special care into making user and content deanonymization as hard as possible, without hurting the dataset informative value as much as possible (§3.3). In this section, we also perform a preliminary assessment of the collection methodology, to confirm the absence of bias in the response process (§3.4).

3.1 Technical aspects of the survey collection

Due to limitations in Wikimedia's caching infrastructure, the survey is injected into the page via client-side code. Wikimedia continuously collects navigation timing performance of a randomly selected sample $\mathcal{T}$ of page views (less than 1 every 1,000 pageviews); the survey is displayed to a randomly selected sub-sample $\mathcal{S}$ of this population (less than 1 every 1,000 of the pageviews with navigation timing information) and only part of the surveys do receive an answer $\mathcal{A}$. Since $\mathcal{A} \subset \mathcal{T}$, several features (that we detail in §3.3) related to page loading performances are also available for pages sampled in the survey responses.

The survey appears on Russian, French and Catalan Wikipedias, as well as English Wikivoyage, and it is displayed in the appropriate language to the viewer. We collect the survey on mobile & desktop version of the site (but not on the mobile app). Instead of asking users a 5-grade Absolute Category Ranking (ACR) score, we opt for a simpler yet still very relevant feedback [25], as users can respond with a *positive*, *neutral* or *negative* experience. Neutral feedback is meant for, e.g, users that have no honest opinion, as well as users who were not paying attention during the rendering, or users that do not understand the question, etc. to avoid biasing the results (§3.4). For the sake of completeness, a snapshot of the survey as it is rendered for English readers is reported in Fig. 1.

The survey is injected in the DOM after the page finished loading (i.e., when `loadEventEnd`[20] fires). In order to give the survey visibility, it is consistently inserted in the top-right area of the wiki article, ensuring that it typically appears above the fold, with randomization of answer order. However, as the users can freely browse the page before the survey appears, it might be out of sight when it's injected in the DOM, which is why we also record the time elapsed between the `loadEventEnd` and the moment the user sees the survey. Also users that are shown the survey are free *not* to respond to the survey, or might as well respond very late (e.g., possibly browsing to other tabs in the meanwhile).

[1]The Wikimedia feature vetting process is still ongoing at time of writing.



Figure 1: Appearance of the Survey in the English Wikipedia (answer order is randomized).

Table 1: Collected corpus of Wikipedia users' QoE feedback.

Period	May 24th – Oct 15th		
No. of survey requests	$\lvert\mathcal{S}\rvert$ =	1746799	
No. of survey answers	$\lvert\mathcal{A}\rvert$ =	62740	$\lvert\mathcal{S}\rvert/\lvert\mathcal{A}\rvert$=3.6%
No. of positive answers	$\lvert\mathcal{A}^+\rvert$ =	53208	$\lvert\mathcal{A}^+\rvert/\lvert\mathcal{A}\rvert$ = 84.8%
No. of neutral answers	$\lvert\mathcal{A}^0\rvert$ =	4838	$\lvert\mathcal{A}^0\rvert/\lvert\mathcal{A}\rvert$ = 7.7%
No. of negative answers	$\lvert\mathcal{A}^-\rvert$ =	4694	$\lvert\mathcal{A}^-\rvert/\lvert\mathcal{A}\rvert$ = 7.5%

Overall, users responded as reported in Tab.1 to about 3.6% of the over 1.7M surveys that have been displayed in the period, for a total of over 62k anwers: 84.8% of the users respond positively to the survey with an almost equal split of the remaining answers to a neutral (7.7%) or negative (7.5%) grades.

3.2 Collected features

We enrich the collected corpus with external sources that are instrumental to the purpose of feedback prediction (§4). A terse summary of the metrics collected is reported in Tab.2, while rationales of the selection for those publicly available is given in §3.3.

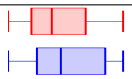
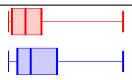
Page: For each page, we record 15 features that concerns it (e.g., its URL, revision ID, size, etc.) and that thus are critical from a privacy point of view. We additionally record the time lapse at which the survey is shown to users, which is instead innocuous.

Performance: Since $\mathcal{S} \subset \mathcal{T}$, then all the 32 navigation-timing performance-related metrics (such as DOM, PLT, TTI, TTFP, connection duration, number of HTTP redirect, DNS wait time, SSL handshake time, etc.) are also collected. Finally, we compute the page download speed which is a simple, yet non linear, transformation of page size and connection duration. These informations are specific to page views, and are less critical to be shared.

User: The 32 collected user-related metrics include the browser, device and OS families. Additionally, we know whether users are logged in Wikipedia, if they are accessing Wikipedia through a tablet device and the number of edits that users have made (coarse bins). These informations are of course highly critical.

Environment: The 36 environmental collected features pertain time, network, geolocation and techno-economic aspects. Except for the time information, directly available from the survey query, we extensively use external data to extract environmental features.

Table 2: Summary of the available/collected (A/C) features that are associated to each users' survey response. The mutual information between the survey answer and A/C features in the class is reported as a boxplot.

Class	A/C	Sample features	MI(x,y)
Page	2/15	Page ID, Page size, Survey viewtime, etc.	
Performance	26/32	DOM, PLT, TTI TTFP, RSI, etc.	
User	21/32	Device, Browser, OS, editCountBucket, etc.	
Environment	12/36	Connection Type, Time, Geolocation, etc.	
Overall	61/115		

As for the network, we leverage MaxMind [4] for IP to ASN and ISP mappings and for geolocation at country (and city) granularity. ISP and ASN mappings are potentially interesting as it can be expected that performances (for the same access technology) vary across ISPs (access technology is also available for about 2/3 of the samples). Concerning geolocation, whereas databases are known not to be reliable for city-level geolocation of server addresses [38], they are generally sufficiently accurate for resolving customer IP addresses, and especially when only ISO-3166-2 country-level precision is required. Country-level precision also allows us to relatively compare performances across users in the same environment, i.e., we normalize the page download speed with respect to the median per-country speed observed in our dataset (in terms of ratio, absolute and relative error).

Additionally, ties between country wealth and network traffic volumes have been established in the literature (particularly, deviation from expected volume [42]): it is thus worth investigating whether there also exist ties between wealth and users' impatience. We use the Gross Domestic Product (GDP) information made available by the World Bank Open Data project [5]. The per-country economic features we consider (namely, per-country GDP, per-country per-capita GDP, etc.) are expressed in terms of Geary-Khamis dollars, which relate to the purchasing power parity, i.e, how much money would be needed to purchase the same goods and services in two countries. The rationale in so doing is that, albeit Web users perception is tied to psychophysics laws [39], there may be environmental conditions tuning this law differently in each country. For instance, a fixed amount of delay may have a smaller perceptual value to users of countries with poor Internet access which GDP-related features might capture: e.g., in other words, one can expect users in a high-GDP country to have better average performance and thus be more impatient than users from a low-GDP one.

Finally, we expect user-home gateways [43] and particularly end-user devices [19, 34] to have a direct impact on the overall performance. As such, we complement the ISP-level view with a device-level information. Particularly, we harvest the Web [6] to find techno-economic information about user devices and in particular, collect device CPU, memory and pricing[2] information. Intuitively, this information complements the per-country GDP information as, e.g., there may be further perceptual differences between users with a costly smartphone in low-GDP vs high-GDP countries. We recognize that device CPU and memory specs are only an *upper-bound* of the achievable performance, as it is the mixture of applications installed and running on a device that determine the amount of *available* CPU and RAM resources, from which user perception will be ultimately affected [19, 34]. Missing this information on a per-sample basis, we attempt to at least construct the per-device statistics, by considering navigation timing information of a large representative sample of Wikipedia users. Particularly, we consider the month of August 2018 during which we observe over 30 million navigation time samples from 29,336 different devices, including all 2,716 devices in our survey. We then construct *deciles* of per-device performance (e.g., of page load time and similar timing information): indeed, it can be expected that users of knowingly slow devices be less impatient, which this additional data source could provide.

3.3 Ethics

The dataset we collect contains obviously sensitive information allowing to deanonymize Wikipedia visitors (such as IP addresses, version of their browser and handsets), as well as linking them to the content they visited (e.g., page, revision ID, time of their visit, etc.). Despite the dataset release policy explicitly forbids user deanonymization, in the interest of respecting personal privacy we have to obscure information so to render user deanonymization as hard as possible, while still allowing meaningful information to be extracted from the data.

Method: Specifically, we opt for an approach where we transform data in a non bijective way (e.g., IP to ASN and ISP mappings that provide network-related properties, while preventing user deanonymization at the same time), or aggregate at a sufficiently coarse grain (e.g., country-level geolocation; obfuscation of browser major/minor version; aggregation of unpopular devices, etc.). For the same reason, we decide to aggregate time-related information at a coarse-grain (hour-level) and drop most content-related features (e.g., page ID). We quantize the page size with a resolution of 10KB, to also make it hard to reverse-engineer which page was visited. We maintain most of the navigation timing related performance, that have the highest mutual information, which we obfuscate wherever necessary (e.g., given that with precise PLT and download speed one could easily reverse engineer the page size, and thus the content, we quantize the download speed in steps of 100Kbps). We point out that, since the Wikimedia feature vetting process is still ongoing at time of writing, shall the bag of available features ultimately differ from the one described here, it will be properly documented.

Results: As a consequence, this loss of information potentially has

[2]Note that we collect pricing information at the time of our query, and not at the time when the device was actually bought; we also ignore price differences among countries, and per-ISP offer bundles.

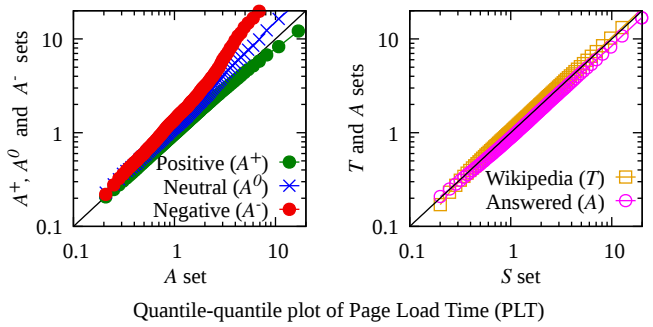


Figure 2: Quantile-quantile plot of PLT statistics for different sets ($\mathcal{T} \supset \mathcal{S} \supset \mathcal{A} = \mathcal{A}^+ \cup \mathcal{A}' \cup \mathcal{A}^-$).

an impact on the global prediction accuracy, which we assess in §4: at the same time, from results presented in Tab. 2, we can expect this effect to be rather limited. Indeed, Tab. 2 reports the number of the publicly available out of the collected (A/C) features in each class (second column), and additionally reports boxplots of the mutual information $MI(x, y)$ between features in the class and the survey answer (last column). *MI* expresses the amount of information (in bits) that can be obtained about the survey answers through the observed variable. Tab. 2 shows that, while we only report about half of the collected features, the available ones overall have a *higher MI* (particularly, note that the 25th percentile, median and 75th percentile are higher in the available feature set, and the feature having the maximum MI is also exported). Particularly, under this angle it is fortunate that features belonging to the performance class, which are those exhibiting the highest mutual information with the user grade, are also the least critical to share.

3.4 Validity of the collection methodology

Despite our care in engineering the survey questioning process, we cannot exclude a-priori the existence of bias in the user survey answer process. For instance, users might refrain to answer when the page loading experience was positive, and be more willing to express their opinion in case of bad experience, which would lead to under-estimate the user satisfaction.

To assess whether our survey collection methodology yields to such (or other) biases, we compare three sets of page view experiences, namely (i) the set $\mathcal{T}$ where we record navigation timing information from the browser (ii) the set $\mathcal{S}$ where users have been *shown* the survey (iii) the set $\mathcal{A}$ where users have actually *answered* to the survey. Finally, we further slice the set of answered surveys $\mathcal{A}$ according to the answer in three additional datasets with (iv) positive $\mathcal{A}^+$, (v) neutral $\mathcal{A}^0$ and (vi) negative $\mathcal{A}^-$ grades.

Among the numerous features we collect, without loss of generality we now limitedly consider the Page Load Time (PLT) distribution. Since $\mathcal{S} \subset \mathcal{T}$ is selected with uniform random sampling, by construction we have that $\mathcal{S}$ and $\mathcal{T}$ are statistically equivalent as far as individual features, such as PLT, are concerned. However, in case where users *decision* to answer to the survey would be biased by the performance of the page, then the PLT statistics should differ among the set of displayed $\mathcal{S}$ vs answered $\mathcal{A}$ surveys. The right-side of Fig. 2 reports a quantile-quantile (QQ)-plot of the empirical PLT distribution, using quantiles of $\mathcal{S}$ on the x-axis and $\mathcal{T}$, $\mathcal{A}$ on the y-axis, from which one can clearly remark the absence of such bias.

Conversely, one would expect that, shall the PLT affect the actual grading of the browsing experience, then PLT statistics should differ among the $\mathcal{A}^+ \cup \mathcal{A}^0 \cup \mathcal{A}^- = \mathcal{A}$ sets. This is shown in the left-side of Fig. 2, comparing the quantiles of the answer set $\mathcal{A}$ on the x-axis to its per-grade slices on the y-axis. Several remarks are in order. First, it can clearly be seen that browsing experience with negative scores fall above the equality line, confirming as expected that pages with longer download time yield to poor experience. Second, similar considerations hold for neutral (slightly above) and positive (slightly below) answers, although they are less visible – in part, this is due since positive grades represent the bulk of the answers $|\mathcal{A}^+|/|\mathcal{A}| = 84.8\%$, for which the PLT statistics of $\mathcal{A}^+$ and $\mathcal{A}$ are mechanically more similar (we will take care of class imbalance when appropriate later on in §4). Third, notice that a range of PLT values are present in the set of positive, neutral and negative answers, indicating as expected that the PLT alone cannot fully capture user perception.

4 USER FEEDBACK PREDICTION

Problem formulation: Disregarding the neutral scores, we now build data-driven models to forecast user answers. This allows to turn the problem into a binary classification one: this simple formulation enables immediate statements of performance objective, that we express in terms of the classic information retrieval metrics.

Clearly, from an operational standpoint a *conservative* estimation of user satisfaction is preferable. Indeed, the service operator wants to avoid that a malfunctioning service that is truly affecting user experience goes undetected, as when the ratio of dissatisfied users increases above a given level this can prompt alert to repair or ameliorate the service. In our settings, conservative prediction results translate into *maximizing the recall of negative scores*.

Reference classification results: Given the class imbalance, we have to preliminarily downsample the dataset[3]: indeed, given that after discarding neutral scores 92% of the users are satisfied, a naïve 0-R classifier that just learns the relative frequency of the scores and systematically answers with the majority class, would achieve 0.92 accuracy – but would entirely miss negative scores, having thus a null $\mathcal{A}^-$ recall. Hence, a more appropriate baseline for recall of unsatisfied users is that of a uniform random selection.

Fig. 3 reports a confusion matrix, additionally highlighting the average accuracy, precision and recall of the unsatisfied users. Results are gathered on all the 115 collected features with a 20-trees random forest [13] on a 10-fold validation using 80% of the samples for training. We obtain very similar results with XGBoost [17], with a slightly higher accuracy but lower $\mathcal{A}^-$ recall, which we do not report to avoid cluttering the pictures. Prediction outcome is clearly deceiving and only slightly better than the naïve baseline, despite the relatively large number of features collected: only 62%

[3]We prefer to avoid the diametrically opposite approach of synthetically generating users score, which is in stark contrast with the very same nature of our work.

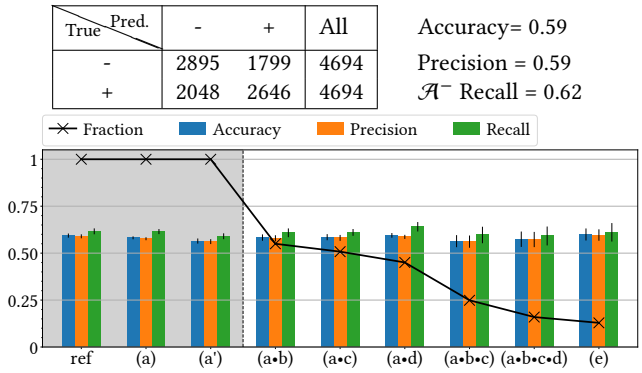


Figure 3: Classification results: Confusion matrix for all collected features as a reference (top) and performance obtained by limiting the (a) 61 available features, (a') 26 features of the performance class, and restricting the attention to (b) Chrome-only browser, (c) Russian population (d) Android OS and (e) top-1000 pages (and combinations thereof).

of the unsatisfied users are correctly captured, with a precision of 0.59. Interestingly, not shown in the picture, performance on the *remaining dataset*, i.e., the set of positive scores filtered out due to class imbalance, remains consistent with an accuracy of 0.65.

Feature subsampling: We next consider how the classification results change with respect to the above reference when considering only the 61/115 publicly available features, which is shown in Fig. 3-(a): as expected, since features in the available set are fewer but with better mutual information, classification results are practically unaffected. We reduce this set even further by only considering the 26 features of the performance class in (a'), which shows a slightly higher, but still very limited, reduction of classification performance: on the one hand, performance-related features consistently rank high in terms of Gini importance, though on the other hand they lack discriminative power for telling user answers apart.

Dataset subsampling: We finally spatially condition the dataset, investigating whether classification performance mechanically improves by reducing the heterogeneity in the dataset, in an attempt to recreate more homogeneous conditions as usually done in the lab studies. Particularly, (a·b) uses publicly available features and restrict the attention to the most popular browser, namely Chrome, considering both mobile and desktop flavors. In (a·c) we instead restrict to users of the prevalent country, and in (a·d) to Android users. We also combine these filters altogether (a·b·c) and (a·b·c·d), and finally consider (e) the top-1000 pages in our dataset. Clearly, conditioning the dataset implies that a smaller fraction of original dataset is available, which we also have to re-balance (solid black line in Fig. 3): in turn, confidence intervals for the metrics of interest increase for decreasing dataset fractions, which is expected. Yet, it is easy to gather that classification performances are only minimally affected in all the above cases, so that the state-of-the performance metrics we collect are apparently not enough to discriminate satisfied vs unsatisfied users.

5 DISCUSSION

In this paper we engineer, collect and (plan to) share at [7, 8] user survey scores pertaining to the quality of their Web browsing experience. Out of over 1.7 million queries, we gather over 62k answers corresponding to either positive (84.8%), neutral (7.7%) or negative (7.5%) experiences. We then develop data driven models to predict user scores: under this angle, the most important (and equally disturbing) takeaway is that it is surprisingly hard to predict even a very coarse-grained indication of user satisfaction. This can be tied in part to the lack of more informative indicators in our dataset, and also raises a number of interesting community-wide challenges, which we discuss next.

Collection and validation methodologies: We remark that this work is the first to collect user feedback from real users in real browsing activity, from an operational deployment. This is in stark contrast with most lab research, where volunteers or crowdworkers are exposed to a very limited heterogeneity (e.g., single device/browser), are not carrying on a browsing activity (e.g., A/B testing uses videos) and are not asked about their satisfaction but about other metrics as a proxy (e.g., which video finished first?). We argue that lab/crowdsourcing experiments and collection in the wild should *coexist*. Particularly, we argue that surveys such as those we are carrying on should be kept *running continuously*, as it is commonplace for VoIP applications that regularly poll their users for a QoE opinion. Operating continuously would lower barriers for further experiments [10], empower website operators with a very relevant performance indicator for their service, informing them in near-real time about impact of new features deployment, and ultimately helping to ameliorate data-driven models.

RSI: not needed, or not enough ?: Concerning Web user QoE metrics, this study seems to suggest a poor discriminative power of the RumSpeedIndex (RSI) so as to predict users scores, at least for Wikipedia users. In part, this may be due to the structure of Wikipedia pages (where, e.g., text may be more prevalent that in other pages in the Alexa top 100 typically considered in similar studies, see §2), which nevertheless raises the question so as to whether it is possible to design more specific metrics that are better fit to the spatial structure of any given page.

Per-device statistics: Given that "computation activities are the main bottleneck when loading a page on mobile browsers" [34], collecting per-device statistics [19] seems a mandatory step. Unfortunately, average per-device performance we considered in this work are not telling enough, as they merely report the resource *upper-bound* (i.e., CPU and RAM capacity) as opposite to the *actual state* of the device (i.e., free RAM and available CPU cycles) corresponding to the page view that the user answered about – which could hopefully ameliorate prediction performance.

ACKNOWLEDGEMENTS

We thank Tilman Bayer, Leila Zia and Nuria Ruiz from the Wikimedia Foundation and Saint Johann from the Russian Wikipedia community, whose input and help shaped this work.

REFERENCES

[1] [n. d.]. https://sites.google.com/a/webpagetest.org/docs/using-webpagetest/metrics/speed-index.
[2] [n. d.]. https://github.com/WPO-Foundation/RUM-SpeedIndex.
[3] [n. d.]. https://www.alexa.com/topsites.
[4] [n. d.]. https://www.maxmind.com/.
[5] [n. d.]. https://data.worldbank.org/.
[6] [n. d.]. https://www.gsmarena.com.
[7] [n. d.]. https://webqoe.telecom-paristech.fr.
[8] [n. d.]. https://meta.wikimedia.org/wiki/Research:Study_of_performance_perception_on_Wikimedia_projects.
[9] 2010. World Wide Wait. https://www.economist.com/science-and-technology/2010/02/12/world-wide-wait.
[10] Eytan Bakshy, Dean Eckles, and Michael S. Bernstein. 2014. Designing and Deploying Online Field Experiments. In *Proc. of the 23rd International Conference on the World Wide Web (WWW)*. https://doi.org/10.1145/2566486.2567967
[11] Athula Balachandran, Vaneet Aggarwal, Shobha, He Yan, et al. 2014. Modeling Web Quality-of-experience on Cellular Networks. In *Proc. ACM MOBICOM*. ACM. https://doi.org/10.1145/2639108.2639137
[12] Enrico Bocchi, Luca De Cicco, and Dario Rossi. 2016. Measuring the Quality of Experience of Web Users. *Proc. ACM SIGCOMM, Internet-QoE Workshop* (2016). https://doi.org/10.1145/3027947.3027949
[13] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (oct 2001), 5–32. https://doi.org/10.1023/A:1010933404324
[14] Jake Brutlag, Zoe Abrams, and Pat Meenan. [n. d.]. Above the fold time: Measuring Web page performance visually. http://conferences.oreilly.com/velocity/velocity-mar2011/public/schedule/detail/18692.
[15] Michael Butkiewicz, Daimeng Wang, Zhe Wu, Harsha V. Madhyastha, and Vyas Sekar. 2015. Klotski: Reprioritizing Web Content to Improve User Experience on Mobile Devices. In *Proc. of the USENIX Conference on Networked Systems Design and Implementation (NSDI)*.
[16] Vint Cerf, Van Jacobson, Nick Weaver, and Jim Gettys. 2012. BufferBloat: what's wrong with the internet? *Commun. ACM* 55, 2 (2012), 40–47. https://doi.org/10.1145/2076450.2076464
[17] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proc. of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 785–794. https://doi.org/10.1145/2939672.2939785
[18] Diego Da Hora, Alemnew Asrese, Vassilis Christophides, Renata Teixeira, and Dario Rossi. 2018. Narrowing the gap between QoS metrics and Web QoE using Above-the-fold metrics. In *Proc. Passive and Active Measurement Conference (PAM)*. https://doi.org/10.1007/978-3-319-76481-8
[19] Mallesham Dasari, Santiago Vargas, Arani Bhattacharya, Aruna Balasubramanian, Samir R. Das, and Michael Ferdman. 2018. Impact of Device Performance on Mobile Internet QoE. In *Proc. ACM Internet Measurement Conference*. https://doi.org/10.1145/3278532.3278533
[20] Zhiheng Wang (Ed.). 2012. Navigation Timing. http://www.w3.org/TR/2012/REC-navigation-timing-20121217/. In *"W3C Recommendation"*.
[21] Sebastian Egger, Peter Reichl, Tobias Hoßfeld, and Raimund Schatz. 2012. "Time is bandwidth"? Narrowing the gap between subjective time perception and Quality of Experience. In *Proc. IEEE International Conference on Communications (ICC)*. https://doi.org/10.1109/ICC.2012.6363769
[22] Jeffrey Erman, Vijay Gopalakrishnan, Rittwik Jana, and K. K. Ramakrishnan. 2013. Towards a SPDY'Ier Mobile Web?. In *ACM CoNEXT*. 303–314. https://doi.org/10.1145/2535372.2535399
[23] Markus Fiedler, Tobias Hoßfeld, and Phuoc Tran-Gia. 2010. A generic quantitative relationship between quality of experience and quality of service. *IEEE Network* 24, 2 (2010), 36–41. https://doi.org/10.1109/MNET.2010.5430142
[24] Qingzhu Gao, Prasenjit Dey, and Parvez Ahammad. 2017. Perceived Performance of Top Retail Webpages In the Wild: Insights from Large-scale Crowdsourcing of Above-the-Fold QoE. In *Proc. ACM SIGCOMM, Internet-QoE Workshop*. https://doi.org/10.1145/3098603.3098606
[25] Tobias Hoßfeld, Poul E Heegaard, Martín Varela, and Sebastian Möller. 2016. QoE beyond the MOS: an in-depth look at QoE via better metrics and their relation to MOS. *Quality and User Experience* 1, 1 (2016), 2.
[26] ITU-T. 2014. Estimating end-to-end performance in IP networks for data application.
[27] ITU-T. 2014. QoE factors in web-browsing.
[28] Conor Kelton, Jihoon Ryoo, Aruna Balasubramanian, and Samir R Das. 2017. Improving User Perceived Page Load Time Using Gaze. In *Proc. of the USENIX Conference on Networked Systems Design and Implementation (NSDI)*.
[29] Yun Ma, Xuanzhe Liu, Shuhui Zhang, Ruirui Xiang, Yunxin Liu, and Tao Xie. 2015. Measurement and Analysis of Mobile Web Cache Performance. In *Proc. of the 24th International Conference on the World Wide Web (WWW)*. https://doi.org/10.1145/2736277.2741114
[30] Robert B Miller. 1968. Response time in man-computer conversational transactions. In *Proc. AFIPS Fall Joint Computer Conference*. ACM.
[31] Ben Miroglio, David Zeber, Jofish Kaye, and Rebecca Weiss. 2018. The Effect of Ad Blocking on User Engagement with the Web. In *Proc. of the 27th International Conference on the World Wide Web (WWW)*. https://doi.org/10.1145/3178876.3186162
[32] Yashar Moshfeghi and Joemon M. Jose. 2013. On Cognition, Emotion, and Interaction Aspects of Search Tasks with Different Search Intentions. In *Proc. of the 22nd International Conference on the World Wide Web (WWW)*. https://doi.org/10.1145/2488388.2488469
[33] Fiona Fui-Hoon Nah. 2004. A study on tolerable waiting time: how long are web users willing to wait? *Behaviour & Information Technology* 23, 3 (2004), 153–163.
[34] Javad Nejati and Aruna Balasubramanian. 2016. An In-depth Study of Mobile Browser Performance. In *Proc. of the 25th International Conference on the World Wide Web (WWW)*. https://doi.org/10.1145/2872427.2883014
[35] Ravi Netravali, Anirudh Sivaraman, Keith Winstein, Somak Das, Ameesh Goyal, and Hari Balakrishnan. 2014. Mahimahi: a lightweight toolkit for reproducible web measurement. In *ACM SIGCOMM Computer Communication Review*, Vol. 44. ACM, 129–130. https://doi.org/10.1145/2740070.2631455
[36] Jakob Nielsen. [n. d.]. Response Times: The 3 Important Limits. https://www.nngroup.com/articles/response-times-3-important-limits/.
[37] Ashkan Nikravesh, Hongyi Yao, Shichang Xu, David Choffnes, and Z. Morley Mao. 2015. Mobilyzer: An Open Platform for Controllable Mobile Network Measurements. In *Proc. ACM MobiSys*. https://doi.org/10.1145/2742647.2742670
[38] Ingmar Poese, Steve Uhlig, Mohamed Ali Kaafar, Benoit Donnet, and Bamba Gueye. 2011. IP Geolocation Databases: Unreliable? *ACM SIGCOMM CCR* 41, 2 (2011). https://doi.org/10.1145/1971162.1971171
[39] Peter Reichl, Sebastian Egger, Raimund Schatz, and Alessandro D'Alconzo. 2010. The logarithmic nature of QoE and the role of the Weber-Fechner law in QoE assessment. In *Proc. IEEE International Conference on Communications (ICC)*. https://doi.org/10.1109/ICC.2010.5501894
[40] Sanae Rosen, Bo Han, Shuai Hao, Z. Morley Mao, and Feng Qian. 2017. Push or Request: An Investigation of HTTP/2 Server Push for Improving Mobile Performance. In *Proc. of the 26th International Conference on the World Wide Web (WWW)*. https://doi.org/10.1145/3038912.3052574
[41] Philipp Singer, Florian Lemmerich, Robert West, Leila Zia, Ellery Wulczyn, Markus Strohmaier, and Jure Leskovec. 2017. Why We Read Wikipedia. In *Proc. of the 26th International Conference on the World Wide Web (WWW)*. https://doi.org/10.1145/3038912.3052716
[42] Chris Smith-Clarke and Licia Capra. 2016. Beyond the Baseline: Establishing the Value in Mobile Phone Based Poverty Estimates. In *Proc. of the 25th International Conference on the World Wide Web (WWW)*. https://doi.org/10.1145/2872427.2883076
[43] Srikanth Sundaresan, Walter de Donato, Nick Feamster, Renata Teixeira, Sam Crawford, and Antonio Pescapè. 2011. Broadband Internet Performance: A View from the Gateway. In *ACM SIGCOMM*. 134–145. https://doi.org/10.1145/2018436.2018452
[44] Matteo Varvello, Jeremy Blackburn, David Naylor, and Konstantina Papagiannaki. 2016. EYEORG: A Platform For Crowdsourcing Web Quality Of Experience Measurements. In *Proc. ACM CoNEXT*. https://doi.org/10.1145/2999572.2999590
[45] Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall. 2014. How Speedy is SPDY?. In *Proc. of the USENIX Conference on Networked Systems Design and Implementation (NSDI)*.
[46] Xiao Sophia Wang, Arvind Krishnamurthy, and David Wetherall. 2016. Speeding up Web Page Loads with Shandian. In *Proc. of the USENIX Conference on Networked Systems Design and Implementation (NSDI)*.
[47] Torsten Zimmermann, Jan Ruth, Benedikt Wolters, and Oliver Hohlfeld. 2017. How HTTP/2 Pushes the Web: An Empirical Study of HTTP/2 Server Push. In *Proc. IFIP Networking*. https://doi.org/10.23919/IFIPNetworking.2017.8264830
[48] Torsten Zimmermann, Benedikt Wolters, and Oliver Hohlfeld. 2017. A QoE Perspective on HTTP/2 Server Push. In *Proc. ACM SIGCOMM, Internet-QoE Workshop*. https://doi.org/10.1145/3098603.3098604

Genre Differences of Song Lyrics and Artist Wikis: An Analysis of Popularity, Length, Repetitiveness, and Readability

Markus Schedl
Johannes Kepler University Linz
Institute of Computational Perception
Linz, Austria
markus.schedl@jku.at

ABSTRACT

Music is known to exhibit different characteristics, depending on genre and style. While most research that studies such differences takes a musicological perspective and analyzes acoustic properties of individual pieces or artists, we conduct a large-scale analysis using various web resources. Exploiting content information from *song lyrics*, contextual information reflected in *music artists' Wikipedia articles*, and listening information, we particularly study the aspects of *popularity*, *length*, *repetitiveness*, and *readability* of lyrics and Wikipedia articles. We measure popularity in terms of song play count (PC) and listener count (LC), length in terms of character and word count, repetitiveness in terms of text compression ratio, and readability in terms of the Simple Measure of Gobbledygook (SMOG). Extending datasets of music listening histories and genre annotations from Last.fm, we extract and analyze 424,476 song lyrics by 18,724 artists from LyricWiki.

We set out to answer whether there exist significant genre differences in song lyrics (RQ1) and artist Wikipedia articles (RQ2) in terms of repetitiveness and readability. We also assess whether we can find evidence to support the cliché that lyrics of very popular artists are particularly simple and repetitive (RQ3). We further investigate whether the characteristics of popularity, length, repetitiveness, and readability correlate within and between lyrics and Wikipedia articles (RQ4).

We identify substantial differences in repetitiveness and readability of lyrics between music genres. In contrast, no significant differences between genres are found for artists' Wikipedia pages. Also, we find that lyrics of highly popular artists are repetitive but not necessarily simple in terms of readability. Furthermore, we uncover weak correlations between length of lyrics and of Wikipedia pages of the same artist, weak correlations between lyrics' reading difficulty and their length, and moderate correlations between artists' popularity and length of their lyrics.

KEYWORDS

music listening behavior; song lyrics; repetitiveness; readability; Last.fm; Wikipedia; analysis

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313604

ACM Reference Format:
Markus Schedl. 2019. Genre Differences of Song Lyrics and Artist Wikis: An Analysis of Popularity, Length, Repetitiveness, and Readability. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313604

1 INTRODUCTION

The way music is created and consumed has substantially changed over time. Various music genres and styles have evolved, prospered, and disappeared again from public knowledge and appreciation. Existing research studying differences in genre and style commonly takes a musicological perspective and analyzes acoustic properties of individual pieces, such as rhythm, dynamics, melody, harmony, and timbre. While being a valid approach, it foregoes exploiting the abundance of web-based data sources that provide information on music pieces and artists.

In contrast, we conduct a large-scale analysis of almost half a million songs by artists of a variety of genres (given by a commercial genre taxonomy and a user-generated genre folksonomy). Extracting content information from lyrics, contextual information provided in artists' Wikipedia articles, and combining these with listening information, we set out to study the characteristics and interplay of *popularity*, *length*, *repetitiveness*, and *readability* of lyrics and Wikipedia articles.

More precisely, we formulate and investigate the following research questions:
RQ1: Can we observe (and if so describe) significant genre differences in song lyrics with regard to repetitiveness and readability?
RQ2: Can we observe (and if so describe) significant genre differences in artist Wikipedia pages with regard to repetitiveness and readability?
RQ3: Is there evidence to support the cliché that lyrics of very popular artists are particularly simple and repetitive?
RQ4: Can we observe (and if so describe) significant correlations between characteristics of lyrics and Wikipedia articles, notably popularity, length, repetitiveness, and readability?

The answers to these RQs will help gaining a better understanding of the role of genre and popularity when characterizing music by content (lyrics) and context (Wikis) information. The results will further allow to ameliorate user preference models by incorporating respective characteristics, and in turn improve personalized music access systems such as information retrieval or recommendation systems. For instance, together with sociodemographic information of music listeners, e.g., provided in the LFM-1b datasets [9, 10, 12], the findings of the study at hand could

Table 1: Statistical summaries of properties of song lyrics (L) and Wikipedia pages of music artist (W).

	Mean	Std.	Min.	Median	Max.
LC	1,990	4,021	5	594	48,640
PC	44,449	149,531	6	6,840	3,838,604
L_characters	1,151	535	108	1,035	4,339
L_words	229	106	23	206	869
L_CR	58.44	7.24	15.04	58.88	90.27
L_SMOG	12.59	3.04	3.00	12.47	31.93
W_characters	9,945	11,621	161	5993	111,835
W_words	1,650	1,931	27	994	18,347
W_CR	50.32	4.25	16.58	51.03	66.89
W_SMOG	11.35	1.31	5.24	11.37	27.19

be used to create country-specific user models of lyrics preferences. Similar to previous work that showed superior performance of culture-aware music recommender systems that integrate descriptors of country-specific music mainstream [11], integrating user models of lyrics preferences could outperform methods that are unaware of this kind of information.

In the remainder of this article, we discuss related literature (Section 2), detail the data sources and data acquisition procedure we base our study upon (Section 3), describe the methodology we use to characterize lyrics and Wikipedia articles (Section 4), elaborate on our study to approach the RQs and discuss its results (Section 5), and conclude including further perspectives (Section 6).

2 RELATED WORK

Previous work on dynamics of music characteristics on a scale larger than individual pieces or artists has predominantly focused on the temporal evolution of music properties, investigating acoustic content features (e.g., rhythm, melody, harmony, and timbre). The analysis is commonly performed on datasets of several hundreds to thousands of music pieces only, and is often biased towards highly popular songs that appeared in the charts.

For instance, Schellenberg and von Scheve [13] investigate the evolution of tempo and key (major or minor) for about 1,000 top-40 songs in the charts, covering 25 years. They find a trend of increasing use of minor key (commonly perceived sadder than major key) and decrease in tempo. They also discover an increase in song duration and proportion of female artists over the years.

Mauch et al. [6] investigate 17,094 songs that appeared in the Billboard Hot 100 charts, covering the years 1960 to 2010. They compute harmonic and timbre features from the audio and apply a topic modeling approach (latent Dirichlet allocation), as a result of which each song is described as a mixture over eight chord progression clusters (topics) and eight timbral clusters. The authors report a strong and continuous decline over time of the topic reflecting dominant 7^{th} chords (often found in Jazz and Blues) and a tremendous increase of music with no clear chord structure in the early 1990s (corresponding to the rise of Hip-Hop and Rap music). This rise is also reflected in a timbral topic that characterizes music as "energetic", "speech", and "bright". Another timbral topic described as "guitar", "loud", and "energetic" reflects the predominance of Arena Rock bands in the 1980s, for instance, Kiss, Queen, and Bon Jovi.

Conducting a large-scale analysis, Serrà et al. [14] examine pitch, timbre, and loudness of 464,411 music pieces recorded between 1955 and 2010. The authors uncover a tendency of simpler pitch sequences (less complex melodies), increasing loudness levels during mastering (lower volume dynamics), and homogenization of timbre (fewer variety in sound color or texture) over the course of time.

Also song lyrics have been an object of study. They are investigated, for instance, by DeWall et al. [1]. The authors consider lyrics of the 10 most popular songs on the Billboard Hot 100 year-end charts, released between 1980 and 2007. They perform several linguistic analyses and find that the use of first person singular pronouns ("I", "me", etc.) increased over time, whereas the use of first person plural pronouns ("we", "our", etc.) decreased; concluding an increase in narcissism. Furthermore, the authors identify a decrease of words related to social interactions ("sharing", "talking", etc.), an increase of words expressing anger and antisocial behavior ("hate", "kill", etc.), and a decrease of words expressing positive emotions ("love", "happy", etc.) over time.

Morris [7] investigates lyrics of 15,000 songs from the Billboard Hot 100 between 1958 and 2017. Similar to our approach, the author measures repetitiveness in terms of how well lyrics can be compressed. He identifies Rihanna, Britney Spears, and Beyoncé as the artists with the most repetitive lyrics (they can be compressed by $\geq 60\%$), whereas lyrics by Frank Sinatra, Elvis Presley, and Brad Paisley can be compressed by the least amount ($\leq 38\%$).

A study of album reviews by Amazon customers on the large scale is conducted by Oramas et al. [8], who analyze 263,525 user reviews of 65,566 albums. They investigate the evolution of sentiments expressed in reviews between 2000 and 2014, using as features degree of affectivity, fraction of positive words among all words, and emotion strength. The authors do not find a clear temporal trend or pattern in reviews with regard to sentiment, except for a peak in 2008 which they try to explain with the election of Barack Obama as US president. Categorizing the reviews according to album release year (1960 to 2014) instead of review publication year, they further discuss the evolution of the two genres Pop and Reggae. While for the former review sentiments have been very stable over the years, positive sentiments in Reggae album reviews peaked between the late 1970s and the mid 1980s.

In contrast to existing work, we take a larger variety and amount of music into account by not only focusing on popular music or chart songs, but considering songs listened to by a large web community of music aficionados, i.e., users of Last.fm. Furthermore, we jointly investigate characteristics of musical content (lyrics) and contextual information (Wikipedia articles). We also refrain from investigating the temporal evolution of these characteristics, instead we provide an analysis of their differences between music genres according to genre definitions of coarse and of fine granularity.

3 DATA ACQUISITION

We base our study on the LFM-1b dataset [9, 10], which contains roughly 1.09 billion listening events scrobbled[1] between 2005 and

[1]Among Last.fm users it is common to use the term "scrobble" to indicate that a music piece has been listened to and to share this information with other users.

2014 by more than 120,000 Last.fm users. Last.fm is one of the largest online music streaming platforms and provides various connectors to other services, including Spotify, Pandora, iTunes, and Youtube, through which users can share what they are listening to.[2]

To analyze genre differences, we obtain **genre** information about artists using the *LFM-1b User Genre Profile* (LFM-1b UGP) dataset [12], which describes each artist by one or several genre and style annotations. In this dataset, two different dictionaries of genres/styles are considered: 20 broad (expert-defined) genres from the commercial music guide Allmusic[3] and 1,998 fine-grained (user-defined) musical style descriptors from Freebase.[4]

3.1 Acquiring Song Lyrics

For the 585,095 artists in the LFM-1b dataset, we gather the top tracks (most frequently listened to by Last.fm users as of January 2017), using the Last.fm API.[5] We then fetch the corresponding song lyrics from LyricWiki.[6] To obtain reliable results for the subsequent readability scoring, which is tailored to the English language, we next perform automatic language detection of the lyrics, using Google's language detection library,[7] which is reported to reach almost 100% precision for English using a naïve Bayes classifier trained on Wikipedia abstracts. Furthermore, to obtain significant results on the artist level, we only consider artists for which we could acquire at least 10 different song lyrics. This filtering eventually yields a dataset of 424,476 song lyrics by 18,724 artists.

3.2 Acquiring Artist Wikipedia Articles

For the 18,724 artists for which we could acquire (English) lyrics of at least 10 songs, we fetch the textual content of English Wikipedia articles (as of October 2018), using the Python wrapper Wikipedia-API.[8] To alleviate disambiguation issues and increase precision when retrieving Wikipedia articles, we use several keyword-based heuristics, e.g., identifying disambiguation pages or requiring that the first sentence contains at least one music-related term.[9] Following this approach, we eventually obtain 11,363 Wikipedia articles.

To ensure reproducibility of our study, the used datasets can be shared upon request. Please contact the author of the paper on this matter.

4 CHARACTERIZING LYRICS AND WIKIPEDIA ARTICLES

We adopt the following scoring approaches to quantify popularity, length, repetitiveness, and readability of lyrics and Wikipedia articles.

[2]https://www.last.fm/about/trackmymusic
[3]https://www.allmusic.com
[4]https://developers.google.com/freebase
[5]https://www.last.fm/api/show/artist.gettoptracks
[6]http://lyrics.wikia.com/Lyrics_Wiki
[7]https://code.google.com/archive/p/language-detection
[8]https://pypi.org/project/Wikipedia-API
[9]After manual inspection of hundreds of Wikipedia pages about music artists, we identified a set of terms at least one of which is used in (almost) every music artist-related page. These words include *music*, *band*, *singer*, *songwriter*, *musician*, *entertainer*, *rapper*, and *DJ*.

4.1 Measuring Popularity

To quantify artist popularity, we aggregate all listening events in the LFM-1b dataset for each artist and compute **play count (PC)** and **listener count (LC)** values, the former referring to the total number of listening events the respective artist attracted, the latter to the number of unique users who listened to the respective artist (at least once).

4.2 Measuring Length

To characterize an artist a's lyrics in terms of length, we compute the arithmetic mean of the number of **characters** and **words** in all song lyrics gathered for a. To describe a's Wikipedia article, we use the same two measures.

4.3 Measuring Repetitiveness

In order to approximate how repetitive a song's lyrics are, we adopt the idea that repetitiveness can be approximated by how much one can compress a text, applied to song lyrics by Morris [7]. More precisely, we use the deflation variant of the LZ77 algorithm [16] provided by the zlib library.[10] For each song, we compute the compressed length in characters of its lyrics and relate it to the uncompressed length, i.e., we compute the lyrics' **compression ratio (CR)**. We then define the repetitiveness score of an artist as the arithmetic mean of the CR of the artist's songs' lyrics in the dataset.

4.4 Measuring Readability

To assess the readability of lyrics, we use the **Simple Measure of Gobbledygook (SMOG)** [5]. It estimates the years of education required to comprehend the text under consideration and has frequently been validated empirically for English documents [2, 15]. Similar to the CR score, we define readability on the artist level as arithmetic mean of the SMOG scores of the artist's lyrics.

5 ANALYSIS AND RESULTS

A general statistical summary of popularity, length, repetitiveness, and readability scores, for both lyrics and Wikipedia articles, computed over all artists, is provided in Table 1. We observe, not surprisingly, that repetitiveness of lyrics (L_CR) is considerably higher on average than that of Wikipedia articles (W_CR), respectively, 58% vs. 50%. However, lyrics are less stable in this regard (standard deviation of 7% vs. 4%). In terms of readability, lyrics (L_SMOG) are slightly more difficult to understand than Wikipedia articles (W_SMOG), cf. SMOG score of 13 vs. 11, observing again less stability for lyrics (standard deviation of 3 vs. 1).

5.1 Genre Differences

Detailing the approach and results of our investigations to answer the RQs, we first focus on the question of genre differences in lyrics (**RQ1**) and Wikipedia articles (**RQ2**), with respect to the repetitiveness and readability scores. To this end, we illustrate in Tables 2 and 3 the means and standard deviations of CR and SMOG scores, for lyrics and Wikipedia articles, respectively, using the coarse genre taxonomy of Allmusic and the fine-grained Freebase folksonomy.

[10]http://www.zlib.net

Table 2: Compression ratio (CR) and readability score (SMOG) of lyrics and Wikipedia pages per genre, using the Allmusic genre taxonomy. Higher values of means are depicted in darker shades of green; higher values of standard deviations in darker shades of blue.

		Lyrics				Wiki			
		CR		SMOG		CR		SMOG	
Genre	**Artists**	**Mean**	**Std.**	**Mean**	**Std.**	**Mean**	**Std.**	**Mean**	**Std.**
Rock	7160	58.92	6.75	12.37	2.80	50.78	3.92	11.41	1.27
Alternative	5880	59.06	6.84	12.43	2.78	50.57	3.90	11.43	1.26
Pop	5322	60.40	6.18	12.48	2.71	50.77	3.90	11.36	1.20
Electronic	2623	59.98	7.81	12.47	3.04	50.44	4.27	11.51	1.32
Folk	2372	57.89	5.94	12.33	2.56	50.56	3.58	11.57	1.20
Punk	2321	58.22	7.48	12.21	2.85	50.91	3.72	11.30	1.28
Jazz	1336	59.09	6.48	12.50	2.92	50.90	3.42	11.59	1.17
Blues	1318	59.16	5.24	12.11	2.56	51.33	3.26	11.60	1.11
RnB	1169	62.83	5.57	13.01	2.61	51.42	3.78	11.31	1.04
Rap	1057	58.36	6.76	13.65	3.20	50.61	4.36	11.27	1.23
Heavy Metal	968	57.13	7.37	12.86	3.31	51.02	4.17	11.59	1.26
Country	866	58.72	5.01	12.44	2.45	50.80	3.52	11.53	1.10
Easy Listening	769	60.59	5.01	12.53	2.50	51.21	3.42	11.44	1.04
Vocal	637	59.85	5.78	12.38	2.69	50.64	3.81	11.35	1.19
Mean		**59.30**		**12.56**		**50.85**		**11.45**	

Considering the *Allmusic* genres, Table 2 reveals that RnB lyrics show the highest repetitiveness (CR of 62.83%), and they do so consistently (one of the lowest standard deviations of 5.57%). On the other hand, Heavy Metal lyrics are found to be least repetitive (CR of 57.13%), however, not consistently between this genre's artists (high standard deviation of 7.37%). As for readability, understanding Rap lyrics by far requires the highest reading capabilities (SMOG of 13.65), whereas Punk and Blues lyrics are the easiest to comprehend (SMOG of ≈12). For Wikipedia articles, no substantial differences between music genres in terms of CR and SMOG are observable when using the Allmusic taxonomy.

Investigating the results obtained with the fine-grained *Freebase* genres (Table 3), we observe highest repetitiveness for lyrics in the genres Dance and Electropop (both CR of ≈63%). In contrast, lowest CR is observed for Death metal and Hardcore punk (both CR of ≈53%). The most consistent genres in terms of CR are Soft rock, Americana, and Rock and Roll (all standard deviation of CR <5%). The genres whose lyrics require the highest reading grades are Hip-Hop, Death metal, and Progressive metal (all SMOG >13). Easiest to understand are lyrics of the genres Post-rock, Psychedelic rock, and Dream pop (all SMOG <11.5). As for artist Wikipedia pages, analogous to the Allmusic taxonomy, both repetitiveness and readability scores occupy a quite narrow range: CR between 49.45 for Indie folk and 52.02 for Classic rock; SMOG between 10.99 for Pop punk and 12.25 for Avant-garde.

5.2 Differences for Popular Artists

To investigate the cliché that song lyrics of very popular artists are particularly simple and repetitive (**RQ3**), we compare the top 20 artists in the LFM-1b dataset (popularity measured in terms of LC) with all other artists in the set. Table 4 depicts the corresponding

Table 3: CR and SMOG using the Freebase genre folksonomy.

		Lyrics				Wiki			
		CR		SMOG		CR		SMOG	
Genre	**Artists**	**Mean**	**Std.**	**Mean**	**Std.**	**Mean**	**Std.**	**Mean**	**Std.**
Indie	4474	58.76	6.83	12.35	2.82	50.41	3.92	11.46	1.24
Alternative rock	3730	59.25	6.45	12.38	2.71	50.84	3.77	11.40	1.24
Singer-songwriter	3544	59.56	5.83	12.61	2.62	50.78	3.66	11.47	1.16
Indie rock	3184	58.30	6.65	12.18	2.73	50.28	3.88	11.43	1.25
Metal	2634	55.95	8.08	12.86	3.51	50.66	4.30	11.42	1.32
Pop rock	2263	61.83	5.55	12.84	2.57	51.20	3.84	11.30	1.20
Indie pop	2246	58.97	6.66	12.15	2.71	50.02	3.95	11.42	1.18
Experimental	1997	56.08	7.69	12.12	3.11	50.47	3.76	11.78	1.30
Classic rock	1968	60.17	5.45	12.24	2.58	52.02	2.94	11.55	1.11
Dance	1804	63.00	7.02	12.72	2.96	50.89	4.27	11.30	1.21
Hard rock	1689	59.73	6.36	12.37	2.72	51.52	3.66	11.43	1.22
Punk rock	1680	58.24	7.27	12.18	2.85	50.98	3.70	11.19	1.29
Electronica	1635	60.04	8.12	12.36	3.20	50.12	4.41	11.49	1.32
Soul	1623	61.37	5.98	12.65	2.73	50.91	3.78	11.36	1.11
Hip-Hop	1585	59.19	6.87	13.33	3.10	50.66	4.16	11.28	1.19
Emo	1546	58.93	7.33	12.28	2.84	51.04	3.92	11.23	1.26
Chill out	1504	59.13	6.91	12.51	2.87	50.39	3.93	11.53	1.23
UK 82	1440	60.31	6.97	12.31	2.85	51.38	3.47	11.63	1.18
Hardcore	1364	56.06	8.29	12.23	3.26	50.88	4.03	11.21	1.30
Folk rock	1278	57.67	5.21	12.23	2.57	50.74	3.43	11.72	1.13
Post-punk	1151	57.72	6.67	12.12	2.81	50.77	3.31	11.57	1.21
New Wave	1127	59.95	6.57	12.36	2.77	50.96	3.49	11.55	1.25
Funk	1123	60.73	6.43	12.71	2.93	51.13	3.52	11.40	1.15
Psychedelic	1108	56.74	6.87	11.56	2.69	51.16	3.40	11.96	1.24
Progressive rock	1051	57.34	6.63	12.19	2.94	51.31	3.58	11.78	1.22
Power pop	955	60.54	6.03	12.56	2.71	50.88	3.66	11.21	1.25
Synthpop	912	60.32	7.49	12.56	2.90	50.26	4.20	11.55	1.30
Electro	892	60.96	8.37	12.42	3.17	50.11	4.28	11.50	1.29
Soft rock	881	61.41	4.76	12.75	2.42	51.80	3.44	11.40	1.04
Pop punk	878	59.41	7.12	12.34	2.78	50.75	4.05	10.99	1.30
Britpop	872	60.92	6.09	12.48	2.56	51.35	3.55	11.42	1.13
Americana	853	57.53	4.75	12.36	2.37	50.69	3.44	11.58	1.10
Soundtrack	845	60.57	6.02	12.36	2.68	51.13	3.43	11.39	1.13
Progressive	815	54.76	7.60	12.15	3.35	50.84	4.01	11.86	1.33
Indie folk	750	56.70	5.70	11.93	2.51	49.45	3.75	11.58	1.16
Post-hardcore	734	55.63	8.35	11.52	3.02	50.63	4.08	11.17	1.35
Lo-fi	712	55.58	6.57	11.56	2.72	49.98	3.60	11.68	1.34
Death metal	705	53.01	8.27	13.59	4.03	50.41	4.29	11.51	1.49
Blues-rock	674	59.27	5.00	11.72	2.56	51.75	2.96	11.58	1.04
Electropop	661	62.96	7.60	12.70	2.87	50.47	4.29	11.44	1.26
Screamo	625	57.49	8.00	11.84	3.02	51.23	3.90	11.16	1.27
Christian	616	56.97	7.09	12.46	2.93	50.37	4.21	11.06	1.29
Post-rock	612	54.30	7.59	11.45	2.75	49.77	4.28	11.73	1.43
Alternative metal	607	59.38	7.34	12.67	2.96	50.93	4.48	11.37	1.36
Metalcore	604	56.10	8.30	12.24	3.54	50.75	4.27	11.13	1.33
Psychedelic rock	595	57.56	6.60	11.46	2.62	51.59	3.13	11.96	1.18
Trip hop	594	58.12	7.20	12.55	3.14	49.82	4.10	11.52	1.31
Rock and Roll	573	60.86	4.96	11.79	2.47	51.89	2.88	11.42	1.08
Dream pop	570	56.61	6.83	11.48	2.74	49.78	3.96	11.72	1.26
Grunge	565	59.42	6.24	12.28	2.74	51.35	3.22	11.37	1.15
Gothic	562	55.76	7.11	12.61	2.81	49.95	4.37	11.74	1.38
Garage rock	558	59.24	5.93	11.52	2.63	51.35	3.14	11.48	1.16
Industrial	540	57.13	7.74	12.61	3.11	50.11	4.18	11.68	1.39
Hardcore punk	521	52.81	7.95	11.66	3.24	50.44	3.92	11.08	1.39
Progressive metal	519	53.76	7.93	13.36	3.86	50.67	4.58	11.82	1.42
Avant-garde	514	53.78	7.64	12.19	3.32	50.42	3.87	12.25	1.35
Mean		**58.32**		**12.30**		**50.77**		**11.48**	

popularity, length, repetitiveness, and readability scores. It further shows the means of these scores computed over all artists in the dataset (in row "Overall mean").

We observe several pronounced differences between the top artists as well as between the top artists and all artists (overall means). To assess statistical significance of these differences, we conduct a t-test for equivalence of sample means, comparing the top artists with the other artists in terms of each property. The respective p-values are shown in the last row of Table 4, results significant ($p < 0.001$) highlighted. With respect to RQ3, indeed, the top artists' lyrics are significantly more repetitive (higher CR values) than others' lyrics. No significant difference can be observed, however, for readability nor length of lyrics. Differences for Wikipedia articles are highly significant for length, CR, and SMOG scores.

5.3 Correlations Between Characteristics

Concerning correlations between popularity, length, repetitiveness, and readability within and between lyrics and Wikipedia pages (**RQ4**), we compute over all artists in the dataset Spearman's rank-order correlation coefficient ρ to cope with the different value ranges of the respective measures. The correlation figures are reported in Table 5; statistically significant values at $p < 0.001$ are highlighted. We observe that, while significant, most non-obvious[11] correlations are only weak. In particular, we see that artist popularity does barely correlate with lyrics length ($0.02 < \rho < 0.09$), but does weakly to moderately correlate with Wikipedia article length ($0.33 < \rho < 0.37$). Supporting the findings from our analysis of top artists (RQ3; cf. Section 5.2 and Table 4), also over all artists in the dataset, popularity is (weakly, but significantly) positively correlated with repetitiveness; however, only when measuring popularity in terms of listener count ($\rho \approx 0.15$). Almost no correlation can be found for play count ($\rho \approx 0.05$). This might be because artists with more repetitive lyrics may appeal to a larger variety of listeners (reflected in the LC value) than those with less repetitive lyrics, whereas the latter are listened to more frequently by a smaller number of listeners (reflected in the PC value).

Readability difficulty of lyrics shows a tendency to slightly increase with their length ($0.29 < \rho < 0.32$); the same holds for Wikipedia readability and length ($0.34 < \rho < 0.36$). We exemplify this observation by considering the genres Rap and Hip-Hop, which are among those with highest SMOG scores (cf. Tables 2 and 3) and at the same time show highest lyrics length: the average length of a Rap song's lyrics is 2,099 characters or 414 words, that of a Hip-Hop song equals 1,921 characters or 381 words. In contrast, over all genres, the average lyrics length is 1,151 characters or 229 words.

Furthermore, weak cross-category correlations (between lyrics and Wikipedia pages) exist between length of lyrics and length of Wikipedia pages ($0.21 < \rho < 0.22$). It seems that artists whose songs have longer lyrics stimulate a slightly higher level of activity or dedication of Wikipedia authors than those with shorter lyrics.

[11] Not surprisingly, popularity measures (PC and LC) as well as length measures (characters and words) strongly correlate. Medium correlations exist between compression ratio and length of lyrics; strong to very strong between CR and length of Wikipedia pages.

6 CONCLUSIONS AND FUTURE WORK

We performed a large-scale analysis of more than 420,000 song lyrics and more than 10,000 Wikipedia pages about music artists, statistically investigating popularity, length, repetitiveness, readability, as well as their correlations. We identified substantial differences in repetitiveness and readability of lyrics between genres, most pronounced when using a fine-grained genre folksonomy. For artist Wikipedia pages, no substantial differences between genres were observable, regardless of the used genre taxonomy or folksonomy.

Evidence supporting the cliché that lyrics of very popular artists are particularly simple and repetitive could be found partially. On the one hand, compression rates of top artists' lyrics are indeed significantly higher than those of others, indicating a higher repetitiveness of those lyrics. On the other hand, this does not hold for readability. Therefore, our analysis showed that lyrics of top artists are repetitive but not necessarily simple in terms of readability.

As a matter of fact, there exist limitations of our study. In particular, community biases are likely to affect the distribution of data items (e.g., artist listening events on Last.fm or songs for which lyrics are available on LyricWiki). Last.fm users' artist and genre preferences which barely generalize to the population at large represent another form of community bias [4] our approach is prone to. Also the distribution of demographics of the involved web services' users is unlikely to correspond to that of the general population [3]. Being aware of these limitations (which we plan to reduce in future work), we nevertheless believe that our findings will enable a better understanding of the role of genre and popularity when characterizing music by content (lyrics) and context (Wikis) information. Integrated into user models, the investigated lyrics and Wiki characteristics may also improve personalized music retrieval and recommendation systems.

Future work will include investigating additional data sources such as artist fan pages or album reviews. Also, while we currently only use the (text) content of Wikipedia's artist pages, metadata about the pages could be included in the investigation, e.g., the number of revisions or contributors to the Wikipedia page. In addition, further features could be defined and analyzed, for instance related to emotion or sentiment expressed in lyrics, Wikis, reviews, etc. Another avenue for further studies is to consider user-specific characteristics, including age, gender, culture, education, knowledge, or personality, and to investigate to which extent the preference for certain categories of songs or artists (described by content and context features) is reflected in such user characteristics.

REFERENCES

[1] C. Nathan DeWall, Richard S. Pond Jr., W. Keith Campbell, and Jean M. Twenge. 2011. Tuning in to psychological change: Linguistic markers of psychological traits and emotions over time in popular U.S. song lyrics. *Psychology of Aesthetics, Creativity, and the Arts* 5, 3 (March 2011), 200–207. http://dx.doi.org/10.1037/a0023195

[2] Cecilia Conrath Doak, Leonard G. Doak, and Jane H. Root. 1996. *Teaching Patients with Low Literacy Skills.* Lippincott Philadelphia, USA. 36–59 pages.

[3] Thomas Krismayer, Markus Schedl, Peter Knees, and Rick Rabiser. 2018. Predicting User Demographics from Music Listening Information. *Multimedia Tools and Applications* (May 2018). https://doi.org/10.1007/s11042-018-5980-y

[4] Paul Lamere. 2008. Social Tagging and Music Information Retrieval. *Journal of New Music Research: Special Issue: From Genres to Tags – Music Information Retrieval in the Age of Social Tagging* 37, 2 (2008), 101–114.

[5] G. Harry Mc Laughlin. 1969. SMOG Grading—A New Readability Formula. *Journal of Reading* 12, 8 (1969), 639–646. http://www.jstor.org/stable/40011226

Table 4: Properties of lyrics (L) and Wikipedia articles (W) for top artists in terms of listener count (LC). Numbers printed in bold are the maxima in each column; those in gray are the minima. The last but one row contains the overall means for *all* analyzed artists. The last row reports the (rounded) p-values of a t-test for equivalence of sample means between the top artists and others; * denotes significance at $p < 0.001$.

Artist	LC	PC	L_chars	L_words	L_CR	L_SMOG	W_chars	W_words	W_CR	W_SMOG
Coldplay	48,640	2,576,390	922	187	62	11	41,942	7,006	56	12
Radiohead	44,707	3,437,326	786	154	59	13	44,541	7,146	56	14
Daft Punk	44,356	2,523,537	2,249	444	**81**	14	38,466	6,346	55	12
Nirvana	41,488	1,878,647	994	206	71	11	34,288	5,625	56	13
Red Hot Chili Peppers	40,421	2,221,660	1,686	336	67	14	79,842	13,460	56	12
Queen	39,874	1,614,548	1,298	265	62	11	68,061	11,463	56	13
Muse	39,726	2,460,597	841	163	65	16	27,692	4,493	55	12
The Rolling Stones	39,033	1,483,385	1,262	255	65	11	78,327	13,272	55	11
Rihanna	38,692	1,313,643	2,000	414	73	15	63,397	10,391	56	13
Adele	38,345	1,386,940	1,463	302	66	13	43,257	7,514	56	11
David Bowie	37,340	1,685,010	1,351	263	63	12	84,252	13,874	53	13
Foo Fighters	35,259	1,444,212	1,289	259	70	13	35,963	6,087	56	12
The Killers	35,199	1,367,039	1,236	252	63	13	30,002	5,044	56	12
Michael Jackson	34,817	958,130	2,390	486	72	14	**95,760**	**15,585**	55	13
The Beatles	34,778	**3,838,604**	1,104	223	67	13	92,065	14,747	54	**15**
Eminem	34,427	1,445,767	**4,339**	**869**	60	**17**	62,174	10,404	54	13
Florence + the Machine	34,208	1,729,489	1,449	299	69	12	23,333	3,950	56	12
Gorillaz	34,157	1,354,351	1,250	246	64	11	24,947	4,211	55	13
Pink Floyd	33,834	2,990,318	878	170	49	9	70,794	11,633	55	13
Linkin Park	33,672	2,296,327	1,568	317	71	**17**	37,141	6,160	**57**	12
Overall mean	2,000	44,961	1,151	229	58	13	9,954	1,652	50	11
***p*-value of t-test**	0*	0*	10^{-3}	10^{-3}	10^{-6*}	10^{-1}	10^{-62*}	10^{-62*}	10^{-7*}	10^{-4*}

	LC	PC	L_chars	L_words	L_CR	L_SMOG	W_chars	W_words	W_CR	W_SMOG
LC	1.000*	0.955*	0.091*	0.093*	0.145*	-0.033*	0.369*	0.369*	0.302*	0.137*
PC		1.000*	0.030	0.023	0.054*	-0.026	0.334*	0.331*	0.300*	0.164*
L_chars			1.000*	0.993*	0.506*	0.315*	0.214*	0.220*	0.188*	-0.032*
L_words				1.000*	0.535*	0.293*	0.217*	0.223*	0.186*	-0.044*
L_CR					1.000*	0.102*	0.230*	0.238*	0.199*	-0.058*
L_SMOG						1.000*	0.000	-0.000	0.016	-0.001
W_chars							1.000*	0.999*	0.834*	0.357*
W_words								1.000*	0.833*	0.344*
W_CR									1.000*	0.266*
W_SMOG										1.000*

Table 5: Spearman's rank-order correlations between lyrics (L) and Wikipedia pages (W) characteristics; * denotes significance at $p < 0.001$.

[6] Matthias Mauch, Robert M. MacCallum, Mark Levy, and Armand M. Leroi. 2015. The Evolution of Popular Music: USA 1960–2010. *Royal Society Open Science* 2, 5 (2015). https://doi.org/10.1098/rsos.150081

[7] Colin Morris. 2017. Are Pop Lyrics Getting More Repetitive? (2017). https://pudding.cool/2017/05/song-repetition (accessed: October 2018).

[8] Sergio Oramas, L. Espinosa-Anke, A. Lawlor, Xavier Serra, and H. Saggion. 2016. Exploring Customer Reviews for Music Genre Classification and Evolutionary Studies. In *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR 2016)*. New York, NY, USA.

[9] Markus Schedl. 2016. The LFM-1b Dataset for Music Retrieval and Recommendation. In *Proceedings of the 6th ACM International Conference on Multimedia Retrieval (ICMR 2016)*. ACM, New York, NY, USA, 103–110. https://doi.org/10.1145/2911996.2912004

[10] Markus Schedl. 2017. Investigating country-specific music preferences and music recommendation algorithms with the LFM-1b dataset. *International Journal of Multimedia Information Retrieval* 6, 1 (2017), 71–84. https://doi.org/10.1007/s13735-017-0118-y

[11] Markus Schedl and Christine Bauer. 2017. Introducing Global and Regional Mainstreaminess for Improving Personalized Music Recommendation. In *Proceedings of the 15th International Conference on Advances in Mobile Computing & Multimedia (MoMM 2017)*. ACM, Salzburg, Austria, 74–81. https://doi.org/10.1145/3151848.3151849

[12] Markus Schedl and Bruce Ferwerda. 2017. Large-Scale Analysis of Group-Specific Music Genre Taste from Collaborative Tags. In *Proceedings of the 2017 IEEE International Symposium on Multimedia (ISM)*. IEEE, Taichung, Taiwan, 479–482. https://doi.org/10.1109/ISM.2017.95

[13] E. Glenn Schellenberg and Christian von Scheve. 2012. Emotional cues in American popular music: Five decades of the Top 40. *Psychology of Aesthetics, Creativity, and the Arts* 6, 3 (August 2012), 196–203. http://dx.doi.org/10.1037/a0028024

[14] Joan Serrà, A. Corral, M. Boguñá, M. Haro, and Josep Lluis Arcos. 2012. Measuring the Evolution of Contemporary Western Popular Music. *Scientific Reports* 2 (July 2012). https://doi.org/10.1038/srep00521

[15] Kevin Wong and Jessica R. Levi. 2017. Readability Trends of Online Information by the American Academy of Otolaryngology – Head and Neck Surgery Foundation.

Journal of Otolaryngology – Head & Neck Surgery 156, 1 (2017), 96–102. https://doi.org/10.1177/0194599816674711

[16] Jacob Ziv and Abraham Lempel. 1977. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory* 23, 3 (May 1977), 337–343. https://doi.org/10.1109/TIT.1977.1055714

Growing Attributed Networks through Local Processes

Harshay Shah
University of Illinois at Urbana-Champaign, IL, USA
hrshah4@illinois.edu

Suhansanu Kumar
University of Illinois at Urbana-Champaign, IL, USA
skumar56@illinois.edu

Hari Sundaram
University of Illinois at Urbana-Champaign, IL, USA
hs1@illinois.edu

ABSTRACT

This paper proposes an attributed network growth model. Despite the knowledge that individuals use limited resources to form connections to similar others, we lack an understanding of how local and resource-constrained mechanisms explain the emergence of structural properties found in real-world networks. We make three contributions. First, we propose a simple and accurate model of attributed network growth that jointly explains the emergence of in-degree, local clustering, clustering-degree relationship and attribute mixing patterns. Second, we make use of biased random walks to develop a model that forms edges locally, without recourse to global information. Third, we account for multiple sociological phenomena: bounded rationality; structural constraints; triadic closure; attribute homophily; preferential attachment. Our experiments show that the proposed Attributed Random Walk (ARW) model accurately preserves network structure and attribute mixing patterns of real-world networks; it improves upon the performance of eight well-known models by a significant margin of 2.5–10×.

CCS CONCEPTS

• **Information systems** → **Web applications**; *Data mining*; *Web mining*; • **Applied computing** → *Sociology*.

KEYWORDS

Network growth; Network Structure; Attributed networks

ACM Reference Format:
Harshay Shah, Suhansanu Kumar, and Hari Sundaram. 2019. Growing Attributed Networks through Local Processes. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313640

1 INTRODUCTION

We present a network growth model that explains how distinct structural properties of attributed networks can emerge from a local edge formation process. In real-world networks, individuals form edges with limited information and partial network access. Moreover, phenomena such as triadic closure and homophily simultaneously influence individuals' decisions to form connections. Over time, these decisions cumulatively shape real-world networks to exhibit rich structural properties: heavy-tailed in-degree distribution, skewed local clustering and diverse attribute mixing patterns. However, we lack an understanding of local, resource-constrained mechanisms that incorporate sociological factors to jointly explain the emergence of multiple structural properties. Additionally, accurate network growth models are useful for synthesizing networks and extrapolating existing real-world networks.

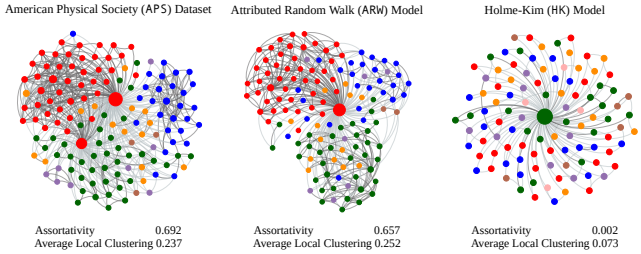


Figure 1: We contrast our proposed model, Attributed Random Walk (ARW), with a non-attributed growth model [20] to underscore the importance of using attributes for network growth.

Well-known models of network growth tend to make unrealistic assumptions about how individuals form edges. Consider a simple stylized example: the process of finding a set of papers to cite when writing an article. In preferential attachment [3] or fitness [5, 10, 43] based models, a node making m citations would pick papers from the *entire* network in proportion to their in-degree or fitness respectively. This process assumes that individuals possess complete knowledge of in-degree or fitness of every node in the network. An equivalent formulation—vertex copying [25]—induces preferential attachment: for every citation, a node would pick a paper uniformly at random from *all* papers, and either cite it or copy its citations. Notice that vertex copying assumes individuals have complete access to the network and form each edge independently. Although these models explain the emergence of power law degree distributions, they are unrealistic: preferential attachment and vertex copying require global node-level knowledge and complete network access respectively. Additionally, they do not account for the role of assortative mixing [35] via nodal attributes (e.g., venue of paper, political interests of Facebook users) in network formation.

Recent papers account for resource constraints [32, 44, 45] and nodal attributes [12, 17]. However, the former disregard attributes and the latter do not provide a realistic representation of edge formation under constraints. Furthermore, both sets of models do not jointly preserve multiple structural properties.

We aim to develop a growth model that accounts for resource constraints and sociological phenomena influencing edge formation in addition to preserving global network structure. We make three key contributions. First, we propose a simple and accurate model of attributed network growth. Second, our model is based on local processes to form edges, without recourse to global network information. Third, our model unifies multiple sociological phenomena—bounded rationality; structural constraints; triadic

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313640

closure; attribute homophily; preferential attachment—to jointly model global network structure and attribute mixing patterns.

The proposed model—Attributed Random Walk (ARW)—jointly explains the emergence of in-degree distribution, local clustering, clustering-degree relationship and attribute mixing patterns through a resource constrained mechanism based on random walks (see Figure 1). In particular, the model relies entirely on local information to grow the network, without access to information of all nodes. In ARW, incoming nodes select a seed node based on attribute similarity and initiate a biased random walk: at each step of the walk, the incoming node either jumps back to its seed or chooses an outgoing link or incoming link to visit another node; it links to each visited node with some probability and halts after it has exhausted its budget to form connections. Our experiments on six large-scale network datasets indicate that the proposed growth model outperforms eight state-of-the-art network growth models by a statistically significant margin of 2.5–10×.

The rest of the paper is organized as follows. We begin by defining the problem statement in Section 2. In Section 3, we outline six network datasets, describe key structural properties of real-world networks and discuss insights from sociological studies. Then, in Section 4, we describe the network growth model. Then, We present experiments in Section 5, discuss related work in Section 6 and conclude in Section 7.

2 PROBLEM STATEMENT

Consider an attributed directed network $G = (V, E, B)$, where V & E are sets of nodes & edges and each node has an attribute value $b \in B$. The goal is to develop a directed network growth model that preserves structural and attribute based properties observed in G. The growth model should be normative, accurate and parsimonious:

(1) **Normative**: The model should account for multiple sociological phenomena that influence how individuals form edges under constraints of limited global information and partial network access.
(2) **Accurate**: The model should preserve key structural and attribute based properties: degree distribution, local clustering, degree-clustering relationship and attribute mixing patterns.
(3) **Parsimonious**: The model should be able to generate networks with tunable structural properties, while having few parameters.

Next, we present empirical analysis on real-world datasets to motivate our attributed random walk model.

3 EMPIRICAL ANALYSIS

We first describe six large-scale network datasets. Then, we describe global network properties, insights from empirical studies and common assumptions in network modeling.

3.1 Datasets

We consider six citation networks of different scales (size, time) from diverse sources: research articles, utility patents and judicial cases. Table 1 lists their summary statistics and global network properties. Three of the six datasets are attributed networks; that is, each node has a categorical attribute value.

We focus on citation networks for two reasons. First, since nodes in citation networks form all outgoing edges to existing nodes at the time of joining the network, these datasets provide a clean basis to study edge formation in attributed networks. Second, the node-level, temporal information in datasets that span long time periods (e.g. USSC) enables us to study structural properties at different time stages via network snapshots. Next, we study the structural and attribute properties of these networks.

3.2 Global Network Properties

Statistical descriptors of network properties [33] such as degree distribution, local clustering, and attribute assortativity quantify the extent to which edge formation shapes global network structure.

Degree distribution: Real-world networks tend to exhibit heavy tailed degree distributions in which a small but significant fraction of the nodes turn into high-degree hubs. We observe that Log-normal fits, with parameters listed in Table 1, well describe the in-degree distributions, consistent with Broido and Clauset's [9] observation that scale-free, real-world networks are rare.

Local Clustering: Real-world networks exhibit high local clustering (LCC), as shown in Table 1. Local clustering can arise from triadic closure [34, 39], where nodes with common neighbor(s) have an increased likelihood of forming a connection. The coefficient of node i equals the probability with which two randomly chosen neighbors of the node i are connected. In directed networks, the neighborhood of a node i can refer to the nodes that link to i, nodes that i links to or both. We define the neighborhood to be the set of all nodes that link to node i. In Figure 2, we show that (a) average

Table 1: Summary statistics & global properties of six network datasets: $|V|$ nodes join the networks and form edges $|E|$ over time period T. In attributed networks, each node has a categorical attribute value that belongs to set A of size $|A|$. The networks exhibit lognormal (LN) in-degree distribution with mean μ and standard deviation σ, high average local clustering (LCC) & attribute assortativity (AA) coefficient and densify over time with densification power law (DPL) exponent α.

Network	Description	$\|V\|$	$\|E\|$	T	A, $\|A\|$	LN (μ, σ)	DPL α	Avg. LCC	AA r
USSC [14]	U.S. Supreme Court cases	30,288	216,738	1754-2002	-	(1.19, 1.18)	2.32	0.12	-
HEP-PH [15]	ArXiv Physics manuscripts	34,546	421,533	1992-2002	-	(1.32, 1.41)	1.67	0.12	-
Semantic [2]	Academic Search Engine	7,706,506	59,079,055	1991-2016	-	(1.78, 0.96)	1.58	0.06	-
ACL [36]	NLP papers	18,665	115,311	1965-2016	VENUE, 50	(1.93, 1.38)	1.43	0.07	0.07
APS [1]	Physics journals	577,046	6,967,873	1893-2015	JOURNAL, 13	(1.62, 1.20)	1.26	0.11	0.44
Patents [27]	U.S. NBER patents	3,923,922	16,522,438	1975-1999	CATEGORY, 6	(1.10, 1.01)	1.94	0.04	0.72

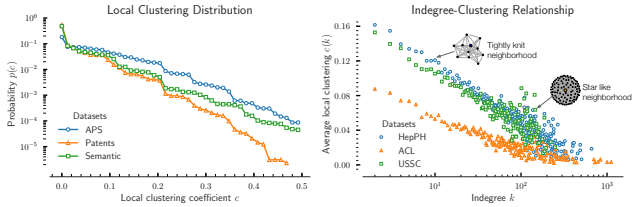


Figure 2: Real-world networks exhibit skewed local clustering distribution (left subplot) and a negatively correlated relationship between in-degree and average local clustering (right subplot).

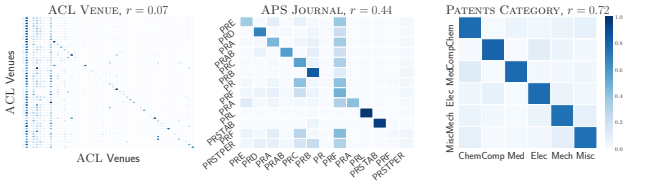


Figure 3: Attributed networks ACL, APS and Patents exhibit homophily w.r.t attributes Venue ($r = 0.07$), Journal ($r = 0.44$) and Category ($r = 0.72$) respectively.

local clustering is not a representative statistic of the skewed local clustering distributions and (b) real-world networks exhibit a negative correlation between in-degree and clustering. That is, low in-degree nodes have small, tightly knit neighborhoods and high in-degree nodes tend have large, star-shaped neighborhoods.

Homophily: Attributed networks tend to exhibit homophily [29], the phenomenon where similar nodes are more likely to be connected than dissimilar nodes. The assortativity coefficient [35] $r \in [-1, 1]$, quantifies the level of homophily in an attributed network. Intuitively, assortativity compares the observed fraction of edges between nodes with the same attribute value to the expected fraction of edges between nodes with same attribute value if the edges were rewired randomly. In Fgiure 3, we show that attributed networks ACL, APS and Patents exhibit varying level of homophily with assortativity coefficient ranging from 0.07 to 0.72.

Increasing Out-degree over Time: The out-degree of nodes that join real-world networks tends to increase as functions of network size and time. This phenomenon densifies networks and can shrink effective diameter over time. Densification tends to exhibit a power law relationship [27] between the number of edges $e(t)$ and nodes $n(t)$ at time t: $e(t) \propto n(t)^{\alpha}$. Table 1 lists the densification power law (DPL) exponent α of the network datasets.

To summarize, citation networks tend to be homophilic networks that undergo accelerated network growth and exhibit regularities in structural properties: heavy tailed in-degree distribution, skewed local clustering distribution, negatively correlated degree-clustering relationship, and varying attribute mixing patterns.

3.3 Insights from Sociological Studies

Sociological studies on network formation seek to explain how individuals form edges in real-world networks.

Interplay of Triadic Closure and Homophily: Empirical studies [6, 24] that analyze the interplay between triadic closure and homophily indicate that *both* structural proximity and homophily are statistically significant factors that simultaneously influence edge formation. Homophilic preferences [29] induce edges between similar nodes, whereas structural factors such as network distance limit edge formation to proximate nodes (e.g. friend of a friend).

Bounded Rationality: Extensive work [16, 28, 40] on decision making shows that individuals are boundedly rational actors; constraints such as limited information, cognitive capacity and time impact decision making. This suggests that resource-constrained individuals that join networks are likely to employ simple rules to form edges using limited information and partial network access.

Current preferential attachment and fitness-based models [3, 13, 41] make two assumptions that are at variance with these findings. First, by assuming that successive edge formations are independent, these models disregard the effect of triadic closure and structural proximity. Second, as discussed in section 1, these models require complete network access or knowledge of node-level properties.

Insights from sociological studies indicate that edge formation in real-world networks comprises biases towards nodes that are similar, well-connected or structurally proximate. Coupled with empirical analyses, it also motivates the need to model how resource-constrained edge formation processes collectively shape global network properties of large-scale networks over time.

4 ATTRIBUTED RANDOM WALK MODEL

We propose an Attributed Random Walk (ARW) model to explain the emergence of key structural properties of real-world networks through entirely local edge formation mechanisms.

Consider a stylized example of how a researcher might go about finding relevant papers to cite. First, the researcher broadly identifies one or more relevant papers, possibly with the help of external information (e.g. Google Scholar). These initial set of papers act as seed nodes. Then, acting under time and information constraints, she will examine papers cited by the seed and papers that cite the seed. Thus, she navigates a chain of backward and forward references to identify similar, relevant papers. Next, through careful analysis, she will cite a subset of these papers. Similarly, users in online social networks might form new friendships by navigating their social circle (e.g., friends of friends) to find similar others.

ARW grows a directed network as new nodes join the network. The mechanism is motivated by the stylized example: an incoming node selects a seed node and initiates a random walk to explore the network by navigating through neighborhoods of existing nodes. It halts the random walk after connecting to a few visited nodes.

In this section, we describe the edge formation mechanisms underlying ARW, explain how ARW unifies multiple sociological phenomena, and discuss methods required to fit ARW to network data.

4.1 Model Description

The Attributed Random Walk (ARW) model grows a directed network $\{\hat{G}_t\}_{t=1}^{T}$ in T time steps. At every discrete time step t, a new node u, with attribute value $B(u)$, joins the network $\hat{G}_t$. After joining the network, node u forms $m(t)$ edges to existing nodes.

The edge formation mechanism consists of two components: Select-Seed and Random-Walk. As shown in Figure 4, an incoming node u with attribute value $B(u)$ that joins the network at time t first selects a seed node using Select-Seed.

Select-Seed accounts for homophilic preferences of incoming nodes using parameters p_{same} and p_{diff} to tune attribute preferences.

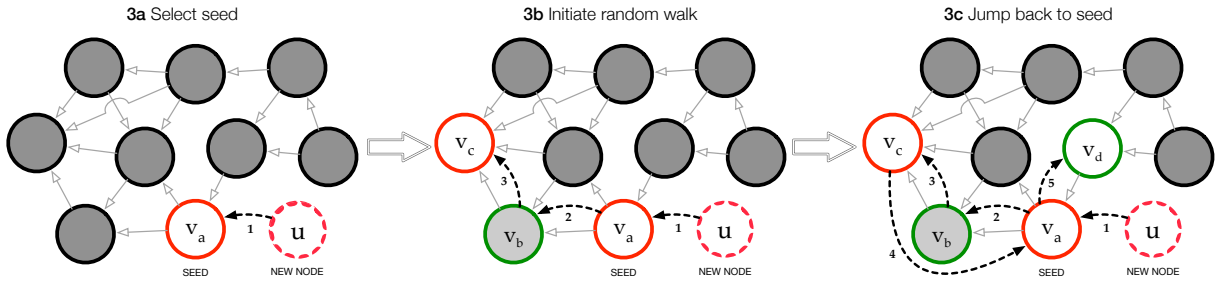


Figure 4: Edge formation in ARW: consider an incoming node u with outdegree $m = 3$ and attribute value $B(u) =$ RED $\in$ {RED, GREEN}. In fig. 3a, u joins the network and selects seed v_a via SELECT-SEED. Then, in fig. 3b, u initiates a RANDOM-WALK and traverses from v_a to v_b to v_c. Finally, u jumps back to its seed v_a and restarts the walk, as shown in fig. 3c. Node u halts the random walk after linking to v_a, v_c & v_d.

In Figure 4, node u selects a seed node and initiates a random walk using RANDOM-WALK to form $m(t)$ links.

> SELECT-SEED
>
> (1) With probability $p_{\text{same}}/p_{\text{same}}+p_{\text{diff}}$, randomly select a seed node from existing nodes that have the same attribute value, $B(u)$.
>
> (2) Otherwise, with probability $p_{\text{diff}}/p_{\text{same}}+p_{\text{diff}}$, randomly select a seed node from existing nodes that do *not* have the same attribute value, $B(u)$.

The RANDOM-WALK mechanism consists of four parameters: attribute-based parameters p_{same} & p_{diff} model edge formation decisions and the jump parameter p_{jump} & out-link parameter p_{out} characterize random walk traversals:

> RANDOM-WALK
>
> (1) At each step of the walk, new node u visits node v_i.
> - If $B(u) = B(v_i)$, u links to v_i with probability p_{same}
> - Otherwise, u links to v_i with probability p_{diff}
>
> (2) Then, with probability p_{jump}, u jumps back to seed s_u.
>
> (3) Otherwise, with probability $1 - p_{\text{jump}}$, u continues to walk. It picks an outgoing edge with prob. p_{out} *or* an incoming edge with prob. $1 - p_{\text{out}}$ to visit a neighbor of v_i.
>
> (4) Steps 1-3 are repeated until u links to $m(t)$ nodes.

When attribute data is absent, ARW simplifies further. A single link parameter p_{link} replaces both attribute parameters p_{same} & p_{diff}. SELECT-SEED reduces to uniform seed selection and in RANDOM-WALK, the probability of linking to visited nodes equals p_{link}.

Note that ARW has two exogenous parameters: the out-degree $m(t)$ and attribute $B(u)$ of incoming nodes. The attribute distribution varies with time as new attribute values (e.g., journals) crop up, necessitating an exogenous parameter. The parameter $m(t)$ is the mean-field value of out-degree m at time t in the observed network. While it is straightforward to model $m(t)$ endogenously by incorporating a densification power-law DPL exponent, exogenous factors (e.g., venue, topic) may influence node out-degree.

Next, we explain how each parameter is necessary to conform to normative behavior of individuals in evolving networks.

4.2 ARW and Normative Behavior

The Attributed Random Walk model unifies multiple sociological phenomena into its edge formation mechanisms.

Phenomenon 1. *(Limited Resources) Individuals are boundedly rational [16, 28, 40] actors that form edges under constraints of limited information and partial network access.*

As shown in Figure 4, RANDOM-WALK only requires information only about the 1-hop neighborhood of a few visited nodes, thereby accounting for the constraints of limited information and partial network access.

Phenomenon 2. *(Structural Constraints) Network distance act as a constraint that limits long-range connections. [24]*

We incorporate structural constraints using p_{jump}, the probability with which a new node jumps back to its seed node after every step of the random walk. This implies that the probability with which the new node is at most k steps from its seed node is $(1 - p_{\text{jump}})^k$; as a result, p_{jump} controls the extent to which nodes' random walks explore the network to form edges.

Phenomenon 3. *(Triadic Closure) Nodes with common neighbors have an increased likelihood of forming a connection. [39]*

When attribute data is absent, ARW controls the effect of triadic closure on link formation using p_{link} because with probability proportional to p_{link}^2, a new node u closes a triad through its random traversal by linking to both, a visited node and its neighbor,

Phenomenon 4. *(Attribute Homophily) Nodes that have similar attributes are more likely to form a connection. [29]*

The attribute parameters p_{same} and p_{diff} modulate attribute assortativity. When $p_{\text{same}} > p_{\text{diff}}$, nodes are more likely to connect if they share the same attribute value, thereby resulting in a homophilic network over time. Similarly, $p_{\text{same}} < p_{\text{diff}}$ and $p_{\text{same}} = p_{\text{diff}}$ make edge formation heterophilic and attribute agnostic respectively.

Phenomenon 5. *(Preferential Attachment) Nodes tend to link to high degree nodes that have more visibility. [3]*

ARW controls preferential attachment by adding structural bias to the random walk traversal using outlink parameter p_{out}, instead of relying on the global degree distribution. Random walks that traverse outgoing edges only (i.e., $p_{\text{out}} = 1$) eventually visit old nodes that tend to have high in-degree. Similarly, random walks that traverse incoming edges only (i.e., $p_{\text{out}} = 0$) visit recently joined nodes that tend to have low indegree. As a result, we use p_{out} to adjust bias towards node degree.

To summarize: ARW incorporates five well-known sociological phenomena—limited resources; structural constraints; triadic closure; attribute homophily; preferential attachment—into a single edge formation mechanism based on random walks.

4.3 Model Fitting

We now briefly describe methods to estimate model parameters, initialize $\hat{G}$, densify $\hat{G}$ over time and sample nodes' attribute values.

Parameter Estimation. We use grid search to estimate the four parameters using evaluation metrics and selection criterion described in Subsection 5.1.

Initialization. ARW is sensitive to a large number of weakly connected components (WCCs) in initial network $\hat{G}_0$ because incoming nodes only form edges to nodes in the same WCC. To ensure that $\hat{G}_0$ is weakly connected, we perform an undirected breadth-first search on the observed, to-be-fitted network G that starts from the oldest node and halts after visiting 0.1% of the nodes. The initial network $\hat{G}_0$ is the small WCC induced from the set of visited nodes.

Node Out-degree. Node out-degree increases non-linearly over time in real-world networks. We coarsely mirror the growth rate of observed network G as follows. Each incoming node u that joins $\hat{G}$ at time t corresponds to some node that joins the observed network G in year $y(t)$; the number of edges $m(t)$ that u forms is equal to the average out-degree of nodes that join G in year $y(t)$.

Sampling Attribute Values. The distribution over nodal attribute values $P_G(B)$ tends to change over time. The change in the attribute distribution over time is an exogenous factor and varies for every network. Therefore, we sample the attribute value $B(u)$ of node u, that joins $\hat{G}$ at time t, from $P_G(B \mid \text{year} = y(t))$, the observed attribute distribution conditioned on the year of arrival of node u.

To summarize, ARW uses four parameters —$p_{\text{same}}, p_{\text{diff}}, p_{\text{jump}}, p_{\text{out}}$— to incorporate resource-constrained individuals' biases towards similar, proximate and high degree nodes. Next, we discuss our experiments on the performance of ARW in preserving global properties of real networks.

5 MODELING NETWORK STRUCTURE

In this section, we evaluate ARW's efficacy in preserving observed network structure relative to well-known growth models.

5.1 Setup

In this subsection, we introduce eight representative growth models and describe evaluation metrics used to fit models to the datasets.

State-of-the-art Growth Models. We compare ARW to eight models representative of the key edge formation mechanisms: preferential attachment, fitness, triangle closing and random walks. Two of the eight models account for attribute homophily and preserve attribute mixing patterns, as listed in Table 2.

Ensuring Fair Comparison. To ensure fair comparison, we modify existing models in three ways. First, for DMS, SAN, KA do not have an explicitly defined initial graph, so we use initialization method used for ARW, described in subsection 4.3. Second, we extend models that use constant node outdegree m by increasing outdegree over time $m(t)$ using the method described in subsection 4.3. In the absence of model-specific parameter estimation methods, we use grid search to estimate the parameters of every network model, including ARW, using evaluation metrics and selection criterion described below.

Evaluation Metrics. We evaluate the model fit by comparing four properties of G & $\hat{G}$: degree distribution, local clustering distribution, degree-clustering relationship and attribute assortativity. We use Kolmogorov-Smirnov (KS) statistic to compare univariate distributions and Weighted Relative Error (WRE) for the degree-clustering relationship. WRE aggregates the relative error between the average local clustering $c(k)$ and $\hat{c}(k)$ of nodes with in-degree k in G and $\hat{G}$ weighted by the fraction of nodes with indegree k in G.

Jointly preserving multiple structural properties is a multi-objective optimization problem. Therefore, for each model, the selection criterion for the grid search parameter estimation method chooses the model parameters that minimizes the ℓ^2-norm of the aforementioned evaluation metrics. We normalize the metrics before computing the ℓ^2-norm to prevent unwanted bias towards any particular metric. We note that the parameter sensitivity of the Forest Fire (FF) model necessitates a manually guided grid search method.

Table 2: We evaluate the performance of our model ARW relative to three preferential attachment (PA) models, two pref. attachment & triangle closing (PA+TC) models and 3 random walk (RW) models.

Model	Abbreviation	Type	Attributed?
Dorogovtsev et al. [13]	DMS	PA	✗
Relay Linking [41]	RL	PA	✗
Kim-Altmann [22]	KA	PA	✓
Social Attribute Network [17]	SAN	PA+TC	✓
Holme-Kim [20]	HK	PA+TC	✗
Herera-Zufiria [19]	HZ	RW	✗
Saramaki-Kaski [37]	SK	RW	✗
Forest Fire [27]	FF	RW	✗

5.2 Results

Now, we evaluate the performance of ARW relative to eight models on the datasets introduced in Subsection 3.1. For every pair of model and dataset, Figure 5 tabulates the evaluation metrics described in Subsection 5.1. These metrics are averaged over 100 runs and measure the accuracy with which the fitted models preserve key global network properties: degree distribution, local clustering distribution, degree-clustering relationship and attribute assortativity.

We use one-sided permutation tests [18] to evaluate the relative performance of ARW. If ARW performs better than a model on a dataset with significance level $\alpha = 0.01$ or $\alpha = 0.001$, the corresponding cells in Figure 5 are shaded gray ( ) or dark gray ( ) respectively. We also group models that have similar edge formation mechanisms by color-coding the corresponding rows in Figure 5. We use green ticks in Figure 5 to annotate models that preserve attribute assortativity up to two decimal places.

Existing models fail to *jointly* preserve multiple properties because they either do not account for triadic closure and homophily or are not flexible enough to generate networks with tunable structural properties. Preferential attachment models—DMS, RL, KA—preserve in-degree distributions (Figure 5A) but not clustering because they do not account for triadic closure (Figure 5B & Figure 5C). Models that use triangle closing mechanisms—HK, SAN—lead to considerable improvement over DMS and KA in modeling local clustering, but perform poorly w.r.t. degree-clustering relationship.

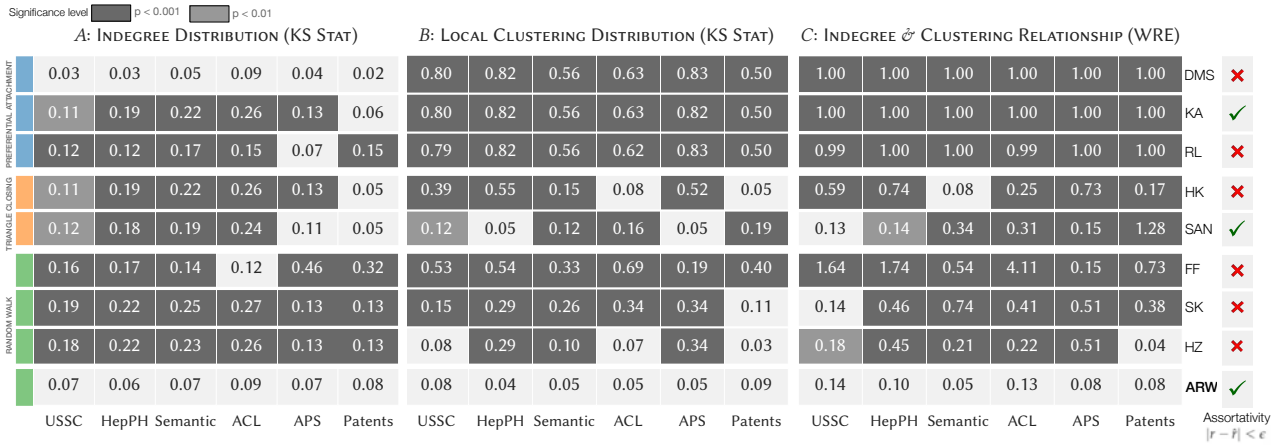


Figure 5: Modeling network structure. Tables 5A, 5B and 5C measure the accuracy of eight models in fitting key properties of real-world networks. Our model, ARW, jointly preserves all four properties accurately and often performs considerably better than existing models: the cells are shaded gray or dark gray if the proposed model ARW performs better at significance level $\alpha = 0.01$ ( ) or $\alpha = 0.001$ ( ) respectively.

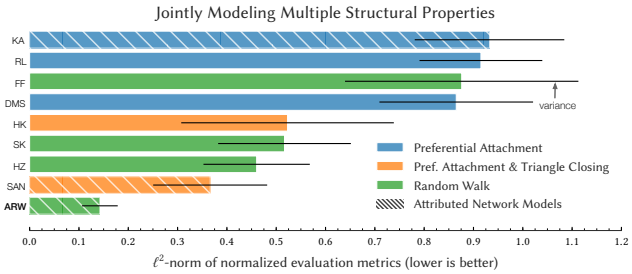


Figure 6: ARW outperforms existing models in jointly preserving key structural properties of node in-degree and local clustering by a significant margin of 2.5x-10x.

Existing random walk models—FF, SK, HZ— do not account for homophily and attribute mixing patterns. FF, in particular, considerably overestimates local clustering because of its recursive edge formation process.

Figure 5 clearly indicates the effectiveness of ARW in jointly preserving multiple global network properties. ARW preserves observed in-degree distributions by adjusting nodes' bias towards high degree nodes using p_{out}. ARW matches the local clustering distribution (Figure 5B) and in-degree & clustering relationship (Figure 5C) with high accuracy using p_{jump} and p_{link}. ARW also preserves attribute assortativity using the attribute parameters p_{same} and p_{diff}.

To summarize, ARW unifies sociological phenomena into a single mechanism to jointly preserve key network properties significantly better than existing models, as shown in Figure 6. Code pertinent to ARW is available online [1]. Please refer to the extended version of our paper [38] for more information.

6 RELATED WORK

Preferential attachment and fitness-based models [4, 5, 10, 30] can preserve heavy-tailed degree distribution, small diameter [8] and temporal dynamics [43] of real-world networks. Extensions of preferential attachment [32, 44, 45] that account for partial network access disregard properties such as clustering and mixing patterns. Models [20, 23, 26] couple preferential attachment with triangle closing to incorporate triadic closure. This increases average local clustering by forming edges between nodes with one or more common neighbors, as shown in Section 5.

Models [12, 17, 21, 46] that account for attribute mixing can be largely categorized as (a) fitness-based model that define fitness as a function of attribute similarity and (b) "microscopic" growth models that require complete temporal information about edge insertions & deletion. In Subsection 5.2, we show that attributed network models SAN and KA preserve mixing patterns but do not account for other structural properties of real-world networks.

First introduced by Vazquez [42], random walk models are inherently local. Models [7] in which new nodes only link to terminal nodes of short random walks generate networks with power law degree distributions [11] and small diameter [31] but do not preserve clustering. Models such as SK [37] and HZ [19], in which new nodes probabilistically link to each visited nodes incorporate triadic closure but are not flexible enough to preserve skewed local clustering, as shown in Subsection 5.2. Recursive random walk models such as FF [27] preserve temporal properties such as shrinking diameter but considerably overestimate local clustering. Furthermore, existing random walk models do not account for nodal attributes.

To summarize, existing models do not accurately explain how resource constrained processes shape well-defined global properties of attributed networks over time. Please refer to the extended version of the paper [38] for a detailed review of existing work.

7 CONCLUSION

In this paper, we proposed a simple, interpretable model of attributed network growth. ARW grows a directed network in the following manner: an incoming node selects a seed node based on attribute similarity, initiates a biased random walk to explore the network by navigating through neighborhoods of existing nodes, and halts the random walk after connecting to a few visited nodes. To the best of our knowledge, ARW is the first model that unifies multiple sociological phenomena—bounded rationality; structural constraints; triadic closure; attribute homophily; preferential attachment—into a single local process to model global network structure *and* attribute mixing patterns. We explored the parameter space of the model

[1] Code: https://github.com/CrowdDynamicsLab/ARW

to show how each parameter intuitively controls one or more key structural properties. Our experiments on six large-scale citation networks showed that ARW outperforms relevant and recent existing models by a statistically significant factor of 2.5–10×.

REFERENCES

[1] [n. d.]. APS Datasets for Research. ([n. d.]). https://journals.aps.org/datasets

[2] Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, Rodney Kinney, Sebastian Kohlmeier, Kyle Lo, Tyler Murray, Hsu-Han Ooi, Matthew Peters, Joanna Power, Sam Skjonsberg, Lucy Lu Wang, Chris Wilhelm, Zheng Yuan, Madeleine van Zuylen, and Oren Etzioni. 2018. Construction of the Literature Graph in Semantic Scholar. In *NAACL*.

[3] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science* 286, 5439 (1999), 509–512.

[4] Michael Bell, Supun Perera, Mahendrarajah Piraveenan, Michiel Bliemer, Tanya Latty, and Chris Reid. 2017. Network growth models: A behavioural basis for attachment proportional to fitness. *Scientific Reports* 7 (2017), 42431.

[5] Ginestra Bianconi and Albert-László Barabási. 2001. Bose-Einstein condensation in complex networks. *Physical review letters* 86, 24 (2001), 5632.

[6] Per Block and Thomas Grund. 2014. Multidimensional homophily in friendship networks. *Network Science* 2, 2 (2014), 189–212.

[7] Avrim Blum, TH Hubert Chan, and Mugizi Robert Rwebangira. 2006. A random-surfer web-graph model. In *2006 Proceedings of the Third Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*. SIAM, 238–246.

[8] Béla Bollobás and Oliver Riordan. 2004. The diameter of a scale-free random graph. *Combinatorica* 24, 1 (2004), 5–34.

[9] Anna D Broido and Aaron Clauset. 2018. Scale-free networks are rare. *arXiv preprint arXiv:1801.03400* (2018).

[10] Guido Caldarelli, Andrea Capocci, Paolo De Los Rios, and Miguel A Munoz. 2002. Scale-free networks from varying vertex intrinsic fitness. *Physical review letters* 89, 25 (2002), 258702.

[11] Prasad Chebolu and Páll Melsted. 2008. PageRank and the random surfer model. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1010–1018.

[12] Maurício L de Almeida, Gabriel A Mendes, G Madras Viswanathan, and Luciano R da Silva. 2013. Scale-free homophilic network. *The European Physical Journal B* 86, 2 (2013), 38.

[13] Sergey N Dorogovtsev, Jose Ferreira F Mendes, and Alexander N Samukhin. 2000. Structure of Growing Networks: Exact Solution of the Barabási–Albert's Model. *arXiv preprint cond-mat/0004434* (2000).

[14] James H Fowler and Sangick Jeon. 2008. The authority of Supreme Court precedent. *Social networks* 30, 1 (2008), 16–30.

[15] Johannes Gehrke, Paul Ginsparg, and Jon Kleinberg. 2003. Overview of the 2003 KDD Cup. *ACM SIGKDD Explorations Newsletter* 5, 2 (2003), 149–151.

[16] Gerd Gigerenzer and Daniel G Goldstein. 1996. Reasoning the fast and frugal way: models of bounded rationality. *Psychological review* 103, 4 (1996), 650.

[17] Neil Zhenqiang Gong, Wenchang Xu, Ling Huang, Prateek Mittal, Emil Stefanov, Vyas Sekar, and Dawn Song. 2012. Evolution of social-attribute networks: measurements, modeling, and implications using google+. In *Proceedings of the 2012 Internet Measurement Conference*. ACM, 131–144.

[18] Phillip Good. 2013. *Permutation tests: a practical guide to resampling methods for testing hypotheses*. Springer Science & Business Media.

[19] Carlos Herrera and Pedro J Zufiria. 2011. Generating scale-free networks with adjustable clustering coefficient via random walks. In *Network Science Workshop (NSW), 2011 IEEE*. IEEE, 167–172.

[20] Petter Holme and Beom Jun Kim. 2002. Growing scale-free networks with tunable clustering. *Physical review E* 65, 2 (2002), 026107.

[21] Fariba Karimi, Mathieu Génois, Claudia Wagner, Philipp Singer, and Markus Strohmaier. 2017. Visibility of minorities in social networks. *arXiv preprint arXiv:1702.00150* (2017).

[22] Kibae Kim and Jörn Altmann. 2017. Effect of homophily on network formation. *Communications in Nonlinear Science and Numerical Simulation* 44 (2017), 482–494.

[23] Konstantin Klemm and Victor M Eguiluz. 2002. Highly clustered scale-free networks. *Physical Review E* 65, 3 (2002), 036123.

[24] Gueorgi Kossinets and Duncan J. Watts. 2009. Origins of Homophily in an Evolving Social Network. *Amer. J. Sociology* 115 (2009), 405–450. http://www.journals.uchicago.edu/doi/abs/10.1086/599247

[25] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D Sivakumar, Andrew Tomkins, and Eli Upfal. 2000. Stochastic models for the web graph. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*. IEEE, 57–65.

[26] Jure Leskovec, Lars Backstrom, Ravi Kumar, and Andrew Tomkins. 2008. Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 462–470.

[27] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2005. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 177–187.

[28] Barton L Lipman. 1995. Information processing and bounded rationality: a survey. *Canadian Journal of Economics* (1995), 42–67.

[29] Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology* 27, 1 (2001), 415–444.

[30] Matúš Medo, Giulio Cimini, and Stanislao Gualdi. 2011. Temporal effects in the growth of networks. *Physical review letters* 107, 23 (2011), 238701.

[31] Abbas Mehrabian and Nick Wormald. 2016. Itś a small world for random surfers. *Algorithmica* 76, 2 (2016), 344–380.

[32] Stefano Mossa, Marc Barthelemy, H Eugene Stanley, and Luis A Nunes Amaral. 2002. Truncation of power law behavior in scale-free network models due to information filtering. *Physical Review Letters* 88, 13 (2002), 138701.

[33] Mark Newman. 2010. *Networks: an introduction*. Oxford university press.

[34] Mark EJ Newman. 2001. Clustering and preferential attachment in growing networks. *Physical review E* 64, 2 (2001), 025102.

[35] Mark EJ Newman. 2002. Assortative mixing in networks. *Physical review letters* 89, 20 (2002), 208701.

[36] Dragomir R. Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. 2013. The ACL anthology network corpus. *Language Resources and Evaluation* (2013), 1–26. https://doi.org/10.1007/s10579-012-9211-2

[37] Jari Saramäki and Kimmo Kaski. 2004. Scale-free networks generated by random walkers. *Physica A: Statistical Mechanics and its Applications* 341 (2004), 80–86.

[38] Harshay Shah, Suhansanu Kumar, and Hari Sundaram. 2017. Growing Attributed Networks through Local Processes. *arXiv preprint arXiv:1712.10195* (2017).

[39] Georg Simmel. 1950. *The sociology of georg simmel*. Vol. 92892. Simon and Schuster.

[40] Herbert A Simon. 1972. Theories of bounded rationality. *Decision and organization* 1, 1 (1972), 161–176.

[41] Mayank Singh, Rajdeep Sarkar, Pawan Goyal, Animesh Mukherjee, and Soumen Chakrabarti. 2017. Relay-linking models for prominence and obsolescence in evolving networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1077–1086.

[42] Alexei Vazquez. 2000. Knowing a network by walking on it: emergence of scaling. *arXiv preprint cond-mat/0006132* (2000).

[43] Dashun Wang, Chaoming Song, and Albert-László Barabási. 2013. Quantifying long-term scientific impact. *Science* 342, 6154 (2013), 127–132.

[44] Li-Na Wang, Jin-Li Guo, Han-Xin Yang, and Tao Zhou. 2009. Local preferential attachment model for hierarchical networks. *Physica A: Statistical Mechanics and its Applications* 388, 8 (2009), 1713–1720.

[45] Jianyang Zeng, Wen-Jing Hsu, and Suiping Zhou. 2005. Construction of scale-free networks with partial information. *Lecture notes in computer science* 3595 (2005), 146.

[46] Elena Zheleva, Hossam Sharara, and Lise Getoor. 2009. Co-evolution of social and affiliation networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1007–1016.

ϵ-Diagnosis: Unsupervised and Real-time Diagnosis of Small-window Long-tail Latency in Large-scale Microservice Platforms

Huasong Shan
JD.com

Yuan Chen
JD.com

Haifeng Liu
JD.com
University of Science and Technology of China

Yunpeng Zhang
JD.com

Xiao Xiao
JD.com

Xiaofeng He
JD.com

Min Li
JD.com

Wei Ding
JD.com

ABSTRACT

Microservice architectures and container technologies are broadly adopted by giant internet companies to support their web services, which typically have a strict service-level objective (SLO), tail latency, rather than average latency. However, diagnosing SLO violations, e.g., long tail latency problem, is non-trivial for large-scale web applications in shared microservice platforms due to million-level operational data and complex operational environments.

We identify a new type of tail latency problem for web services, *small-window long-tail latency* (SWLT), which is typically aggregated during a small statistical window (e.g., 1-minute or 1-second). We observe SWLT usually occurs in a small number of containers in microservice clusters and sharply shifts among different containers at different time points. To diagnose root-causes of SWLT, we propose an unsupervised and low-cost diagnosis algorithm– ϵ-Diagnosis, using two-sample test algorithm and ϵ-statistics for measuring similarity of time series to identify root-cause metrics from millions of metrics. We implement and deploy a real-time diagnosis system in our real-production microservice platforms. The evaluation using real web application datasets demonstrates that ϵ-Diagnosis can identify all the actual root-causes at runtime and significantly reduce the candidate problem space, outperforming other time-series distance based root-cause analysis algorithms.

CCS CONCEPTS

• **General and reference** → *Performance*; *Measurement*; • **Information systems** → *Online analytical processing*.

KEYWORDS

Root-cause analysis; tail latency; time series similarity

ACM Reference Format:
Huasong Shan, Yuan Chen, Haifeng Liu, Yunpeng Zhang, Xiao Xiao, Xiaofeng He, Min Li, and Wei Ding. 2019. ϵ-Diagnosis: Unsupervised and Real-time Diagnosis of Small- window Long-tail Latency in Large-scale Microservice Platforms . In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313653

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313653

1 INTRODUCTION

Tail latency [3, 7–9, 18, 19, 22, 25, 29, 29, 38, 44, 48, 49] becomes the main consideration as Service Level Objective (SLO) on performance [1, 30] for today's most of systems and applications, such as search engine [7, 25], key-value store [9, 21], datacenter network system [22, 48], shared networked storage [49], online data-intensive applications [44], big data system [8] etc. Strict SLO usually requires tail latency, rather than average latency. For example, Google requires the 99th percentile service time (written *TP*99) for its webSearch cluster within tens of milliseconds [25].

Some e-commerce companies try their best to provide the fastest user experience, they expect their web services can serve the requests within several milliseconds. Using a large statistical window to calculate the tail latency can not accurately identify the long-tail latency problem, thus they calculate the tail latency for their online shopping web services during 1-minute period, even during 1-second period. We call this type of tail latency as *small-window tail latency*. We study the small-window tail latency from the real-production web services in container-based microservice platforms (Section 2), finding that a new type of long-tail latency problem– *small-window long-tail latency* (SWLT), which usually occurs in a small number of containers in the microservice clusters, sharply shifts among different containers at different time points, and the tail latency of each container varies significantly.

Diagnosing root-causes of SWLT quickly is a challenging task. Firstly, as the business requirements increase, the number of web services dramatically increases. Figure 1 records the weekly number of web services from an internet company since July 2018, consequently the monitored performance metrics increase in million level [43]. Once SWLT occurs, identifying root-causes of SWLT seems like finding a needle from haystack. Secondly, current machine learn and deep learning based root-cause analysis approaches, heavily depend on training of massive data with highly computing cost, which is suitable for diagnosing daily [23] or seasonal anomaly [46], not fit for diagnosing the long-tail latency at extremely small timescales which have a heavy-tail, frequent-change

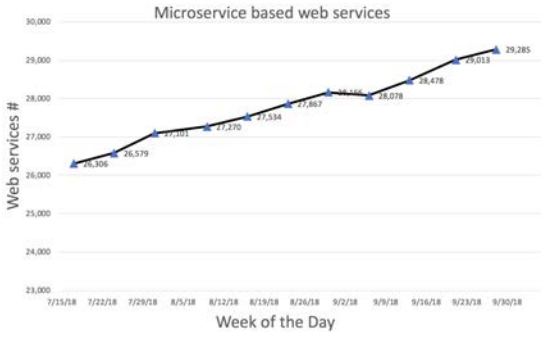


Figure 1: Number of web services in an internet company.

and high-variance feature. Diagnosing root-causes of SWLT requires the algorithm with low computation cost and high recall, and real-time delivery of analytical results.

In this paper, we present an unsupervised and low-cost root-cause analysis algorithm to diagnose root-causes of SWLT at runtime for web services in large-scale microservice platforms. In particular, we make the following contributions.

- We identify a new type of tail latency problem, *small-window long-tail latency* (e.g., in an 1-minute or 1-second period), which has a heavy-tail and high-variance characterization (Section 2). To the best of our knowledge, no current root-cause analysis algorithm can scale to such granularity.
- We propose an unsupervised and low-cost root-cause analysis algorithm–ϵ-Diagnosis (Section 3), using two-sample test algorithm and ϵ-statistics for measuring the similarity of time series, to identify root-causes of SWLT from millions of metrics for on-line web services at runtime. The ϵ-statistics is specially well-suited for the root-cause analysis for the heavy-tailed and highly-variant web applications.
- We implement a real-time diagnosis system in our real-production microservice platforms (Section 3). The evaluation using real-production datasets demonstrates the effectiveness and efficiency of the proposed root-cause analysis algorithm (Section 4). Our results show that ϵ-Diagnosis can identify all the root-cause metrics of SWLT with highest confidence level (lowest confidence threshold) compared to other time series distance based analysis algorithms, and reduce the candidate problem metrics to about 10%.

2 SMALL-WINDOW TAIL LATENCY

Long-tail phenomenons have been broadly observed in data-center [1, 7–10, 20, 22, 25, 37, 38, 44, 47–49], we focus on studying the tail latency at extremely small timescales (e.g., 1 minute, even 1 second) for web services deployed in container based microservice platforms.

Datasets. We have running tens of thousands of web services in our microservice platforms as shown in Figure 1, 90% applications are deployed in a cluster of less than 100 containers. So we select 4 types of representative applications with different cluster sizes, from small, medium, big, to super as shown in Table 1. We identify SWLT, and manually verify the problematic containers and root-cause metrics. We record 13 types of metrics for each container,

Table 1: Datasets from real-production web services.

DataSet	Small	Medium	Big	Super
Problematic Containers (#)	2	2	7	13
Total Containers (#)	15	55	99	260
Root-cause Metrics (#)	2	3	12	17
Total Metrics (#)	13*15	13*55	13*99	13*260
Alarm Windows (minutes)	15	30	10	5

such as CPU utilization, memory usage, Disk I/O utilization etc., all the metrics are observed with 1-minute resolution.

Observations. Figure 2 shows the 99th percentile response time (called *TP*99, which are calculated during 1 minute. Here, we use *TP*99 as an example of tail latency) and its variability of each container for the four application datasets.

From Figure 2a, we can see *only a very small number of containers are problematic when SWLT occurs*. For application Small, Medium, Big, there is only 1 container with problems most of the time. Application Super can have up to 5 problematic containers, which can have the problem at the same time. For all cases, the number of problematic containers is very small compared to the total number of containers. Figure 2a also shows that *the small-window tail latency of each container changes very sharply*. For example, the tail latency increases from several milliseconds to approximately 2.5 seconds during the 25-minute monitoring period at the time of 18 for application Small.

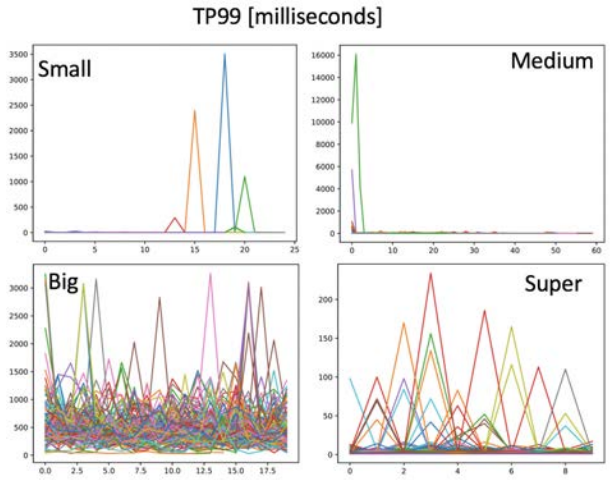
To analyze the variability of TP99 for each container, We calculate the coefficient of variation (COV) [45] for each container in the container clusters for the four applications. *COV* is the ratio of the standard deviation and the mean of the dataset, if $COV > 1$, it means high-variance distribution, otherwise, it's low-variance. In Figure 2b, y-axis is *COV* for each container. We can see *the small-window tail latency for all the applications is highly variant*. For application Super, there are 24 high-variance containers, Big is 12, Medium is 18, Small is 7.

Due to heavy-tail and high-variance characterization, identifying root-causes of SWLT seems like finding a needle from haystack. Developing an intelligent and real-time analysis system to automatically diagnose root-cause metrics of SWLT from the application cluster for large-scale web services has an important implication for application administrators to identify SLO violations.

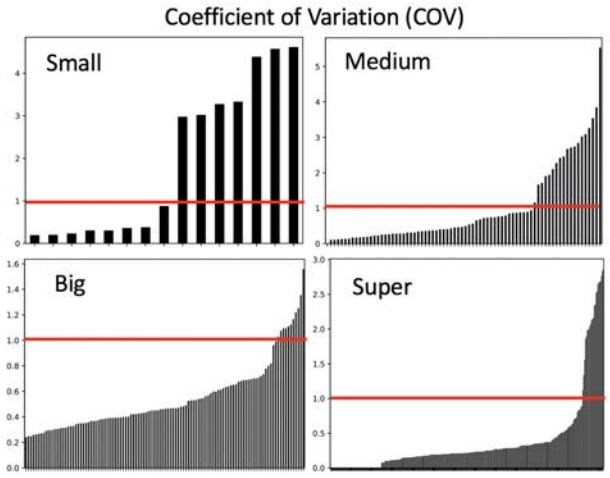
Goals. To diagnose root-causes of SWLT with high-variance and frequent-shift in large-scale microservice platforms, the objectives of designing the diagnosis algorithm and system are two-fold: (1) the algorithm and the system can quickly diagnose root-causes at runtime with low computation cost, (2) the algorithm can significantly reduce the problem space (metrics) while guaranteeing not to miss any actual root-cause.

3 ϵ-DIAGNOSIS SWLT

We propose an unsupervised and low-cost root-cause analysis algorithm, ϵ-Diagnosis, which can diagnose root-causes of SWLT for large-scale web services at runtime.



(a) TP99 of 4 APPs. Each line represents the $TP99$ for a container.



(b) COV for 4 APPs. Each bar represents the COV for a container.

Figure 2: $TP99$ and COV for 4 application clusters.

We assume that once the long response time occurs, root-cause metrics in the problematic container might significantly change between the abnormal and normal periods. Such that, root-cause analysis is to identify the significantly-changed metrics. Therefore, in order to identify the significantly-changed metrics as the candidate root-cause metrics, we can use two-sample null hypothesis test (abnormal and normal samples) [26], which can use various time series similarity measurement algorithms [17]. Here, we adopt ϵ-statistics test (energy distance correlation) algorithms.

Detecting SWLT. To diagnose the root-cause of long-tail latency, the first task is to detect the long-tail latency. Threshold-based detection [12] is the simplest and widely-used anomaly detection approach. Our diagnosis system provides the alarm threshold interface (e.g., *TP99 threshold*) for the application administrators to detect SWLT for their web services. For example, the administrators can define a rule: if the 99th percentile response time during 1 minute for one service is bigger than 2000ms, it will trigger an alarm. Furthermore, the system provides *alarm window* interface to aggregate the alarm number for the same type of alarms during the alarm window. For example, if the alarm window is 15 minutes, the alarm system only reports the first alarm during the 15-minute time window. Once we detect a long-tail latency, it triggers ϵ-Diagnosis algorithm to analyze root-causes.

Selecting two samples from the snapshot. To identify the significantly-changed metrics as the root-cause, we store the context of web applications in the snapshot for comparison analysis when long-tail latency occurs. The snapshot includes various time-series metrics data, extracted from both the application layer and infrastructure layer cross millions of containers.

For application layer, we aggregate various performance metrics from the log files of various servers (e.g., Apache, Tomcat, MySQL), the metrics include throughput, QPS, concurrent loads, response time, number of error log, number of log, number of database connections etc. For infrastructure layer, we record all the metrics about CPU, memory, disk, network of each container. We formulate these time-series metrics as a time-series vector by

$$S_{(t)} = [x_1, x_2, \ldots, x_n]$$

where x_i is the aggregation value during the statistical/sampling period for each metric.

The system provides the alarm window interface to aggregate the alarm number for the same type of alarms during the alarm window. Thus we can guarantee there must exist some anomaly metrics during the alarm window leading to long tail latency. We use time-series metrics data during the alarm window as the *abnormal sample* (S_A). We choose time-series metrics data during the normal period from the snapshot as the *normal sample* (S_N).

Two-sample null hypothesis test. We would like to find the significantly-changed time series metrics as root-cause metrics. The two-sample test [36] is one of the most commonly used hypothesis tests when you want to compare two independent datasets to see if they are statistically similar or not. So we use two-sample test as the algorithm flow of ϵ-Diagnosis. The hypothesis of the two-sample test can be expressed by:

$$\begin{cases} H_0: & S_A = S_N. \\ H_a: & S_A \neq S_N. \end{cases} \tag{1}$$

if H_0 is true, it means that abnormal samples (S_A) and normal samples (S_N) are statistically equal. Otherwise, they are statistically different.

Further, we can use permutation test [31] or bootstrapping [40] to do hypothesis test, and calculate the p-value (P) using the sampling distribution of the test statistic under the null hypothesis. For different confidence level, we can get a confidence threshold to accept or reject the hypothesis. If the distribution of the test statistic of Hypothesis(1) is symmetric about 0, the test statistic is a two-sided test; otherwise it is a one-side test. For example, if the confidence level is 99% and it is a two-sided test, the confidence threshold (α) is 0.05,

$$\begin{cases} P < \alpha, S_A \neq S_N. \\ P \geqslant \alpha, S_A = S_N. \end{cases} \tag{2}$$

Here, if $P < \alpha$, we reject the hypothesis, which means that the anomaly sample and the normal sample are significantly different, so the corresponding metrics of the samples are potential root-causes. The overall ϵ-Diagnosis algorithm can be described in Algorithm 1.

Algorithm 1 Pseudo-code for the ϵ-Diagnosis algorithm

Input: small-window long-tail latency, M time-series metrics of N containers, confidence threshold α, alarm window
Output: candidate root-cause metrics, problematic containers

```
procedure ε-DIAGNOSIS
    for ContainerN ← 1 to N do
        S_A = getAnomalySample
        S_N = getNormalSample
        for MetricM ← 1 to M do
            ( ρ (S_A, S_N), P ) = Calculate Energy distance correlation coefficient of S_A and S_N using Equation (3) with P-value
            if P < α then
                /* Reject Hypothesis S_A != S_N */
                Add MetricM as a candidate root-cause metric
                Add ContainerN as a candidate problematic container
            else
                /* Accept Hypothesis S_A = S_N */
            end if
        end for
    end for
end procedure
```

The confidence threshold (α) plays a critical role in root-cause accuracy for our root-cause service. If α is too low, we might miss some true root-cause metrics, leading to higher false negative. If α is too high, we might consider more metrics as the potential root-cause, leading to higher false positive. So we have to make a trade-off between recall and precision. In the evaluation section, we empirically get the optimal confidence threshold α. We leave the auto-tuning of α as our future work.

ϵ-Statistics (Energy distance correlation). There are a lot of literature work on time series similarity measurement [17]. From observations in Figure 2) in Section 2, we note that the tail latency at extremely small timescale for web services is heavy-tailed and highly-variant. Energy distance based ϵ-statistics is specially well-suited for heavy-tailed and highly-variant datasets with a low computation cost [42]. Thus, we adopt ϵ-statistics test (energy distance correlation) to measure the similarity of two samples.

Energy distance is a variation of squared pairwise distance. The Energy correlation coefficient ($\rho(S_A, S_N)$) between anomaly samples (S_A) and normal samples (S_N) is defined as the square root of,

$$\rho^2(S_A, S_N) = \begin{cases} \frac{cov^2(S_A,S_N)}{\sqrt{\sigma^2(S_A)\sigma^2(S_N)}}, & \sigma^2(S_A)\sigma^2(S_N) > 0. \\ 0, & \sigma^2(S_A)\sigma^2(S_N) = 0. \end{cases} \quad (3)$$

where cov is the covariance of the two samples, σ is the standard deviation of each sample. The benefit and speciality of ϵ-statistics test is distribution-free, scale-equivariant and rotation-invariant. Therefore, it is suitable for diagnosing root-causes of long-tail latency for response time sensitive user-facing web services, in which case the tail latency is aggregated at small timescales, e.g., during 1 minute or 1 second.

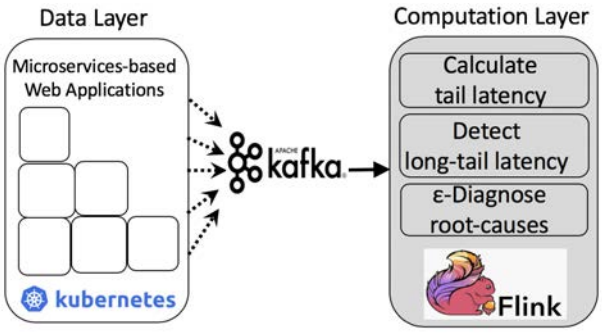


Figure 3: System architecture.

Real-time diagnosis system. We implement an automatic and intelligent diagnosis system at runtime using ϵ-Diagnosis algorithm to analyze root-causes of the long-tail latency at extremely small timescales (e.g., 1 minute or 1 second) for tens of thousands web applications deployed in our microservice platforms managed by Kubernetes [1] as shown in Figure 3. It consists of two main components: data layer and computing layer.

To support population-scale applications, enabling their SLO violation monitoring and delivering results as fast as possible (seconds) are non-trivial. One challenge is to transfer the large volume operation data from the distributed containers. To reduce the amount of data to transfer, we adopt Apache thrift [2] in the data collection agents. Thrift can work with plug in serialization protocols and data compression (e.g. gzip). We observe the compression ratio is around 1/23. The data is pipelined by kafka [3]. Kafka is able to be scaled quickly and easily without incurring any downtime, and handle many terabytes of data with consistent performance.

In computation layer, we calculate the small-window tail latency, detect SWLT by comparing the tail latency with the predefined alarm thresholds, and use ϵ-Diagnosis algorithm to identify the significantly-changed metrics as root-cause metrics. We adopt Apche flink [4] to implement these functions in computation layer, which can process the stream data with high performance and low latency. All the long-tail alarms are stored in event database (e.g., MySQL), all the time-series metrics are stored in time-series database (e.g., ClickHouse [5]), time-series database [33] can provide scalable performance for analytics and aggregation of time-series.

4 EVALUATIONS

4.1 Operational Data in Real-production

Our monitoring cluster consists of approximately 300 containers, which can monitor the small-window tail latency of approximately 30000 web services in our microservice platforms as shown in Figure 1. We deployed the root-cause analysis system in the monitoring cluster. Figure 4 shows the hourly amount of long-tail latency

[1] https://kubernetes.io/
[2] https://thrift.apache.org/
[3] https://kafka.apache.org/
[4] https://flink.apache.org/
[5] https://clickhouse.yandex/

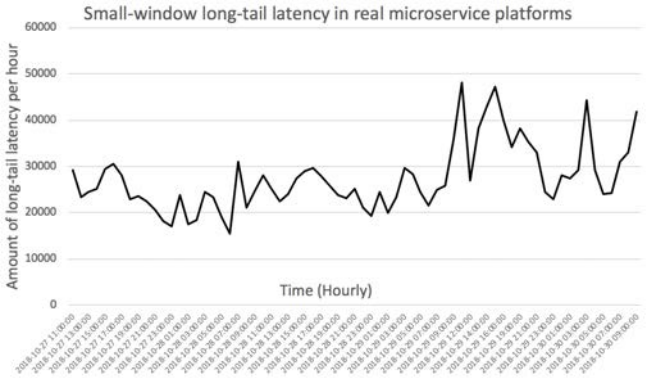


Figure 4: Service rate of ϵ-Diagnosis in a real-life production.

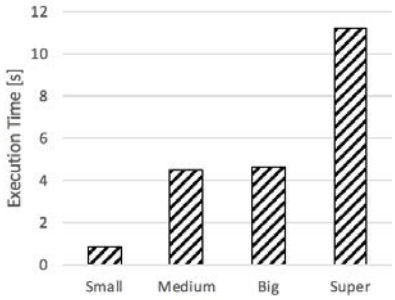


Figure 5: Execution time of ϵ-Diagnosis, in seconds.

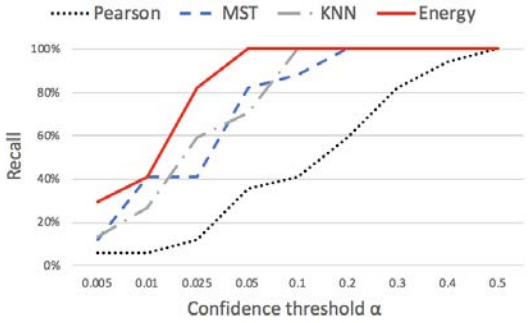


Figure 6: Energy can reach 100% recall quickly as α increases.

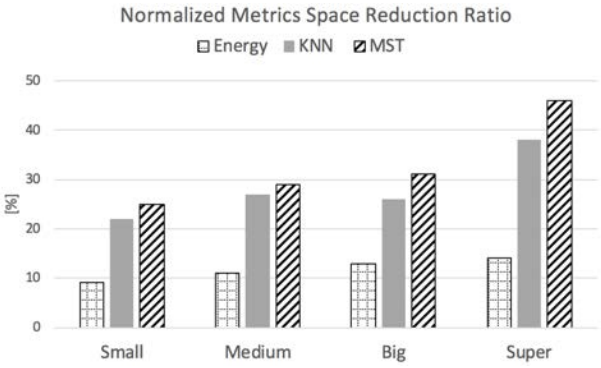


Figure 7: Energy can reduce metrics to approximately 10%.

alarms in three days for our web services. We detected approximately 30,000 long-tail latency with a peak of 50,000 in every hour. The request rate of root-cause diagnosis in total is 8.3 per second with a peak of 13.9 per second.

Execution time. One goal of designing the algorithm and the system is to provide an diagnosis service to identify root-causes of SWLT in mocroservice platforms at runtime. So we evaluate the computation cost for ϵ-Diagnosis.

We run the ϵ-Diagnosis system in a container equipped with a quad-core Intel Core i7 2.8 GHz CPU and 16 GB 2133 MHz LPDDR3 memory. Figure 5 shows the execution time for ϵ-Diagnosis algorithm for the datasets in Table 1. For Small application, the execution time of ϵ-Diagnosis is less than 1 second. As the number of the containers increases, the running time increases. Medium and Big are in the same level, since the sample size (alarm window) reduces from 30 to 10 as shown in Table 1, although the container amount increases from Medium to Big. In our microservice platforms, 80% applications are small web applications. ϵ-Diagnosis can finish the analysis within 1 second, so we use 10 containers to serve the requests of root-cause analysis as shown in Figure 4.

4.2 Performance of Algorithms

Next, we compare the performance of ϵ-Diagnosis with other time-series distance based root-cause analysis algorithms.

Experimental Methodology. We use an empirical approach in our experiments to define α. For example, we define α as a set of values [0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5], which are corresponding to the confidence level of two-sample test [99%, 98%, 96%, 90%, 80%, 60%, 40%, 20%, 0%] (two-sides test). In our experiments, we gradually increase α and evaluate the performance of the algorithms.

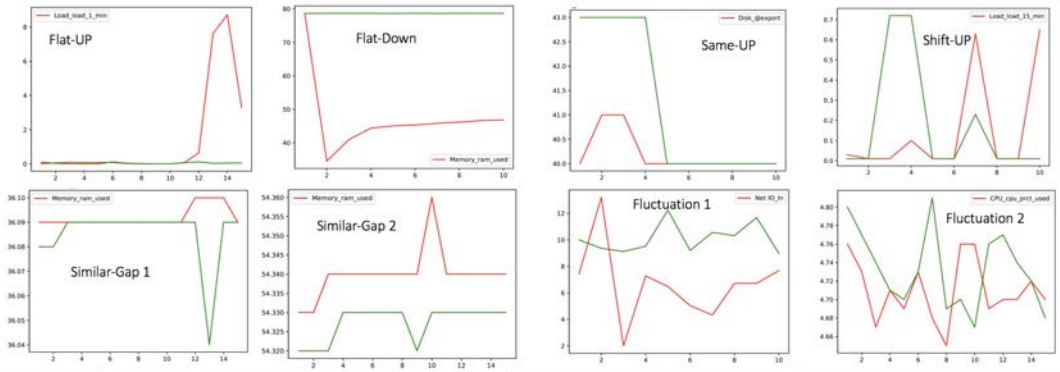
We compare the performance of the ϵ-Diagnosis algorithm (based on energy distance) with other three time-series similarity measurements. Pearson's distance is the simplest one, defined by,

$$\rho(S_A, S_N) = \frac{cov(S_A, S_N)}{\sigma(S_A)\sigma(S_N)} \tag{4}$$

where cov is the covariance of the two samples, σ is the standard deviation of each sample. K-NN [16, 26] and MST [15] algorithms both discretize the time-series samples, transform the distance of two time-series samples into inter-point distance, and use graph-theoretic algorithm to measure their similarity. KNN algorithm uses nearest neighbours to model the inter-point distance of two time series, meanwhile MST algorithm uses minimum spanning tree to represent their inter-point distance.

Results. We require that the algorithm can significantly reduce the problem space from million-level metrics while guaranteeing not missing any actual root-cause, namely guaranteeing 100% recall.

Figure 6 shows the recall of the four time-series measurement algorithms on all four datasets. As the confidence threshold α increases, the recall increases. We note that Energy can achieves 100% recall when α equals 0.05. The two graphic-theory based algorithm (MST and KNN) are 0.2 and 0.1, respectively. As α increases, Energy can identify all the root-causes at the fastest speed. Pearson can not find all the actual root-causes until α equals 0.5. So Pearson can



Algorithm	Flat-UP	Flat-Down	Same-UP	Shift-UP	Similar-Gap 1	Similar-Gap 2	Fluctuation 1	Fluctuation 2
Pearson	0.903	0.041	0.060	0.924	0.072	0.073	0.867	0.901
MST	0.568	0	0.774	0.582	0.430	0	0	0.737
KNN	0.063	0.011	0.033	0.296	0.103	0.106	0	0.712
Energy	0.043	0	0.061	0.047	0.031	0.001	0	0.211

Figure 8: Various two-sample patterns. Each line represents a sample. Smaller P, more significantly different time series.

Table 2: Precision of the algorithms for the four datasets.

DataSet	Small	Medium	Big	Super
MST (%)	62	59	67	66
KNN (%)	76	67	65	62
Energy (%)	79	72	76	72

not fit our case and operational environments, and we will omit the evaluation of Pearson.

Figure 7 shows that how much the time-series similarity measurement algorithms can narrow down the candidate problematic metrics from the total metrics under the condition of 100% recall (For Energy, KNN, MST, α is 0.05, 0.1, and 0.2). Energy can reduce metrics to approximately 10%, outperforming the other two, since it uses the smallest α, counting smaller metrics as the potential metrics. For the same reason, Energy can also get the best precision (e.g., 79% for the small application) as shown in Table 2.

To explain why Energy can outperform the other three algorithms, Figure 8 lists several representative time-series patterns from the infrastructure layer metrics and their corresponding p-value (P). Smaller P, more significantly different the two samples. We can note Energy is more sensitive than the other algorithms for all of the cases, specifically, it can identify more patterns as different samples, so it can achieve 100% recall with the smallest α. Meanwhile, there also exists false positive for Energy, since it might falsely identify some more metrics patterns as root-causes (e.g., "Fluctuation 2"). Pearson can not identify the the difference of "Flat-UP", which is disappointing, because several root-cause metrics have this type of peak pattern from our observations in our microservice platforms, it will miss this type of root-cause metrics.

5 RELATED WORK

Traditional rule (or signature) based performance diagnosis approaches for distributed systems, heavily depend on the operational fault knowledge base (such as Fingerprint [2, 6, 11]), and detect the anomaly and performance bottlenecks through tracing the request process flowing in large-scale distributed system (such as GoogleÃćâĈňâĎćs Dapper [39], Mace [24], Pinpoint [5], Pip [34], Stardust [35], X-Trace [14], Whodunit [4], Retro [27], Pivot Tracing [28], Quanto [13]). However, it is hard to build both operational fault knowledge base and call relationship in complex micro-service based applications with tens of thousands of invocation graphs [32]. Compared to rule based root-cause analysis, we do not need any operational knowledges and the training process.

Models and techniques for root-cause analysis are extensively summarized in the survey paper [41]. Luo et al. [26] model incident diagnosis as a two-sample problem and apply k-NN statistic based method to correlate event and time-series. Daily chronic problems [23] are statistically diagnosed, and seasonal KPIs alarms [46] are detected by an unsupervised anomaly detection algorithm based on VAE (Variational Auto-Encoder). Compared to statistical learning based root-cause analysis, our approach is an unsupervised approach with low computation cost. More importantly, our diagnosis target is more fine-grained long-tail latency problem at extremely small timescale (e.g., 1-minute or 1-second) for the large-scale on-line web services. To the best of our knowledge, no current root-cause analysis algorithm can scale to such granularity.

6 CONCLUSIONS

We identify a new type of SLO violation for web services, SWLT in large-scale microservice platforms, which has a heavy-tail and high-variance characterization. To diagnose root-causes of SWLT, we propose an unsupervised and low-cost algorithm–ϵ-Diagnosis,

using two samples test algorithm based on ϵ-statistics. We develop and deploy the root-cause diagnosis system in the microservice platforms to help identify the potential root-causes of SWLT from millions of metrics for tens of thousands of web services. Through evaluating with real-production datasets, we show that ϵ-Diagnosis can outperform other time-series distance based root-cause analysis algorithms in finding all the actual root-cause metrics of SWLT with highest confidence level.

REFERENCES

[1] Salman A Baset. 2012. Cloud SLAs: present and future. *ACM Operating Systems Review* (2012).

[2] Peter Bodik, Moises Goldszmidt, Armando Fox, Dawn B Woodard, and Hans Andersen. 2010. Fingerprinting the datacenter: automated classification of performance crises. In *Proceedings of the 5th European conference on Computer systems*. ACM, 111–124.

[3] Valeria Cardellini, Emiliano Casalicchio, Francesco Lo Presti, and Luca Silvestri. 2011. Sla-aware resource management for application service providers in the cloud. In *Network Cloud Computing and Applications (NCCA), 2011 First International Symposium on*. IEEE, 20–27.

[4] Anupam Chanda, Alan L Cox, and Willy Zwaenepoel. 2007. Whodunit: Transactional profiling for multi-tier applications. In *ACM SIGOPS Operating Systems Review*, Vol. 41. ACM, 17–30.

[5] Yen-Yang Michael Chen, Anthony J Accardi, Emre Kiciman, David A Patterson, Armando Fox, and Eric A Brewer. 2004. Path-based failure and evolution management. (2004).

[6] Ira Cohen, Steve Zhang, Moises Goldszmidt, Julie Symons, Terence Kelly, and Armando Fox. 2005. Capturing, indexing, clustering, and retrieving system history. In *ACM SIGOPS Operating Systems Review*, Vol. 39. ACM, 105–118.

[7] Kristal Curtis, Peter Bodík, Michael Armbrust, Armando Fox, Mike Franklin, Michael Jordan, and David Patterson. 2010. *Determining SLO Violations at Compile Time.*

[8] Jeffrey Dean and Luiz André Barroso. 2013. The tail at scale. *Commun. ACM* (2013).

[9] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. 2007. Dynamo: amazon's highly available key-value store. In *ACM SIGOPS operating systems review*, Vol. 41. ACM, 205–220.

[10] Pamela Delgado, Diego Didona, Florin Dinu, and Willy Zwaenepoel. 2018. Kairos: Preemptive Data Center Scheduling Without Runtime Estimates. In *ACM SoCC*.

[11] Songyun Duan and Shivnath Babu. 2008. Guided problem diagnosis through active learning. In *Autonomic Computing, 2008. ICAC'08. International Conference on*. IEEE, 45–54.

[12] Hector Fernandez, Corina Stratan, and Guillaume Pierre. 2014. Robust performance control for web applications in the cloud. In *4th International Conference on Cloud Computing and Services Science*.

[13] Rodrigo Fonseca, Prabal Dutta, Philip Levis, and Ion Stoica. 2008. Quanto: Tracking Energy in Networked Embedded Systems.. In *OSDI*, Vol. 8. 323–338.

[14] Rodrigo Fonseca, Michael J Freedman, and George Porter. 2010. Experiences with Tracing Causality in Networked Services. *INM/WREN* 10 (2010), 10–10.

[15] Jerome H Friedman and Lawrence C Rafsky. 1979. Multivariate generalizations of the Wald-Wolfowitz and Smirnov two-sample tests. *The Annals of Statistics* (1979), 697–717.

[16] Jerome H Friedman, Lawrence C Rafsky, et al. 1983. Graph-theoretic measures of multivariate association and prediction. *The Annals of Statistics* 11, 2 (1983), 377–391.

[17] Tak-chung Fu. 2011. A review on time series data mining. *Engineering Applications of Artificial Intelligence* 24, 1 (2011), 164–181.

[18] Anshul Gandhi, Yuan Chen, Daniel Gmach, Martin Arlitt, and Manish Marwah. 2011. Minimizing data center SLA violations and power consumption via hybrid resource provisioning. In *Green Computing Conference and Workshops (IGCC), 2011 International*. IEEE, 1–8.

[19] Anshul Gandhi, Yuan Chen, Daniel Gmach, Martin Arlitt, and Manish Marwah. 2012. Hybrid resource provisioning for minimizing data center SLA violations and power consumption. *Sustainable Computing: Informatics and Systems* 2, 2 (2012), 91–104.

[20] Liu Haifeng, Jinjun Zhang, Huasong Shan, Min Li, Yuan Chen, Xiaofeng He, and Xiaowei Li. 2019. JCallGraph: Tracing Microservices at Massive Scale at JD.com. In *Cloud Computing (CLOUD), the 2019 International Conference on*.

[21] Vikas Jaiman, Sonia Ben Mokhtar, Vivien Quéma, Lydia Chen, and Etienne Rivière. 2018. Héron: Taming Tail Latencies in Key-Value Stores under Heterogeneous Workloads. In *International Symposium on Reliable Distributed Systems (SRDS) 2018*.

[22] Keon Jang, Justine Sherry, Hitesh Ballani, and Toby Moncaster. 2015. Silo: Predictable message latency in the cloud. In *ACM SIGCOMM Computer Communication Review*, Vol. 45. ACM, 435–448.

[23] Soila P Kavulya, Scott Daniels, Kaustubh Joshi, Matti Hiltunen, Rajeev Gandhi, and Priya Narasimhan. 2012. Draco: Statistical diagnosis of chronic problems in large distributed systems. In *Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on*. IEEE, 1–12.

[24] Charles Edwin Killian, James W Anderson, Ryan Braud, Ranjit Jhala, and Amin M Vahdat. 2007. Mace: language support for building distributed systems. In *ACM SIGPLAN Notices*, Vol. 42. ACM, 179–188.

[25] David Lo, Liqun Cheng, Rama Govindaraju, Parthasarathy Ranganathan, and Christos Kozyrakis. 2015. Heracles: Improving resource efficiency at scale. In *ACM SIGARCH Computer Architecture News*, Vol. 43. ACM, 450–462.

[26] Chen Luo, Jian-Guang Lou, Qingwei Lin, Qiang Fu, Rui Ding, Dongmei Zhang, and Zhe Wang. 2014. Correlating events with time series for incident diagnosis. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1583–1592.

[27] Jonathan Mace, Peter Bodik, Rodrigo Fonseca, and Madanlal Musuvathi. 2015. Retro: Targeted Resource Management in Multi-tenant Distributed Systems.. In *NSDI*. 589–603.

[28] Jonathan Mace, Ryan Roelke, and Rodrigo Fonseca. 2015. Pivot tracing: Dynamic causal monitoring for distributed systems. In *Proceedings of the 25th Symposium on Operating Systems Principles*. ACM, 378–393.

[29] Aniket Mahanti, Niklas Carlsson, Anirban Mahanti, Martin Arlitt, and Carey Williamson. 2013. A tale of the tails: Power-laws in internet measurements. *IEEE Network* 27, 1 (2013), 59–64.

[30] Anton Michlmayr, Florian Rosenberg, Philipp Leitner, and Schahram Dustdar. 2009. Comprehensive qos monitoring of web services and event-based sla violation detection. In *Proceedings of the 4th international workshop on middleware for service oriented computing*. ACM, 1–6.

[31] Anders Odén, Hans Wedel, et al. 1975. Arguments for Fisher's permutation test. *The Annals of Statistics* 3, 2 (1975), 518–520.

[32] Fábio Oliveira, Sahil Suneja, Shripad Nadgowda, Priya Nagpurkar, and Canturk Isci. 2017. Opvis: extensible, cross-platform operational visibility and analytics for cloud. In *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference: Industrial Track*. ACM, 43–49.

[33] Tuomas Pelkonen, Scott Franklin, Justin Teller, Paul Cavallaro, Qi Huang, Justin Meza, and Kaushik Veeraraghavan. 2015. Gorilla: A fast, scalable, in-memory time series database. *Proceedings of the VLDB Endowment* 8, 12 (2015), 1816–1827.

[34] Patrick Reynolds, Charles Edwin Killian, Janet L Wiener, Jeffrey C Mogul, Mehul A Shah, and Amin Vahdat. 2006. Pip: Detecting the Unexpected in Distributed Systems.. In *NSDI*, Vol. 6. 9–9.

[35] Raja R Sambasivan, Alice X Zheng, Michael De Rosa, Elie Krevat, Spencer Whitman, Michael Stroucken, William Wang, Lianghong Xu, and Gregory R Ganger. 2011. Diagnosing Performance Changes by Comparing Request Flows.. In *NSDI*, Vol. 5. 1–1.

[36] Howard J Seltman. 2012. Experimental design and analysis. *Online at: http://www.stat. cmu. edu/, hseltman/309/Book/Book. pdf* (2012).

[37] Huasong Shan, Qingyang Wang, and Calton Pu. 2017. Tail attacks on web applications. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1725–1739.

[38] Huasong Shan, Qingyang Wang, and Qiben Yan. 2017. Very Short Intermittent DDoS Attacks in an Unsaturated System. In *International Conference on Security and Privacy in Communication Systems*. Springer, 45–66.

[39] Benjamin H Sigelman, Luiz Andre Barroso, Mike Burrows, Pat Stephenson, Manoj Plakal, Donald Beaver, Saul Jaspan, and Chandan Shanbhag. 2010. *Dapper, a large-scale distributed systems tracing infrastructure.* Technical Report. Technical report, Google, Inc.

[40] Kesar Singh and Minge Xie. 2008. Bootstrap: a statistical method. *Unpublished manuscript, Rutgers University, USA. Retrieved from http://www. stat. rutgers. edu/home/mxie/RCPapers/bootstrap. pdf* (2008).

[41] Marc Solé, Victor Muntés-Mulero, Annie Ibrahim Rana, and Giovani Estrada. 2017. Survey on models and techniques for root-cause analysis. *arXiv preprint arXiv:1701.08546* (2017).

[42] GJ Székely. 2003. E-Statistics: The energy of statistical samples. *Bowling Green State University, Department of Mathematics and Statistics Technical Report* 3, 05 (2003), 1–18.

[43] Jörg Thalheim, Antonio Rodrigues, Istemi Ekin Akkus, Pramod Bhatotia, Ruichuan Chen, Bimal Viswanath, Lei Jiao, and Christof Fetzer. 2017. Sieve: actionable insights from monitored metrics in distributed systems. In *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference*. ACM, 14–27.

[44] Balajee Vamanan, Jahangir Hasan, and TN Vijaykumar. 2012. Deadline-aware datacenter tcp (d2tcp). *ACM SIGCOMM Computer Communication Review* 42, 4 (2012), 115–126.

[45] Akshat Verma, Gargi Dasgupta, Tapan Kumar Nayak, Pradipta De, and Ravi Kothari. 2009. Server workload analysis for power minimization using consolidation. In *Proceedings of the 2009 conference on USENIX Annual technical conference*. USENIX Association, 28–28.

[46] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. 2018. Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 187–196.

[47] Shungeng Zhang, Huasong Shan, Qingyang Wang, Jianshu Liu, Qiben Yan, and Jinpeng Wei. 2019. Tail Amplification in n-Tier Systems: A Study of Transient Cross-Resource Contention Attacks. In *ICDCS*.

[48] Timothy Zhu, Daniel S Berger, and Mor Harchol-Balter. 2016. SNC-Meister: Admitting more tenants with tail latency SLOs. In *Proceedings of the Seventh ACM Symposium on Cloud Computing*. ACM, 374–387.

[49] Timothy Zhu, Alexey Tumanov, Michael A Kozuch, Mor Harchol-Balter, and Gregory R Ganger. 2014. Prioritymeister: Tail latency qos for shared networked storage. In *Proceedings of the ACM Symposium on Cloud Computing*. ACM, 1–14.

Adaptive matrix completion for the users and the items in tail

Mohit Sharma
University of Minnesota
Twin Cities, MN, USA
sharm163@umn.edu

George Karypis
University of Minnesota
Twin Cities, MN, USA
karypis@umn.edu

ABSTRACT

Recommender systems are widely used to recommend the most appealing items to users. These recommendations can be generated by applying collaborative filtering methods. The low-rank matrix completion method is the state-of-the-art collaborative filtering method. In this work, we show that the skewed distribution of ratings in the user-item rating matrix of real-world datasets affects the accuracy of matrix-completion-based approaches. Also, we show that the number of ratings that an item or a user has positively correlates with the ability of low-rank matrix-completion-based approaches to predict the ratings for the item or the user accurately. Furthermore, we use these insights to develop four matrix completion-based approaches, i.e., Frequency Adaptive Rating Prediction (FARP), Truncated Matrix Factorization (TMF), Truncated Matrix Factorization with Dropout (TMF + Dropout) and Inverse Frequency Weighted Matrix Factorization (IFWMF), that outperforms traditional matrix-completion-based approaches for the users and the items with few ratings in the user-item rating matrix.

CCS CONCEPTS

• **Information systems → Collaborative filtering**; **Personalization**; **Recommender systems**.

KEYWORDS

Recommender systems; Collaborative filtering; Matrix completion; Matrix factorization

ACM Reference Format:
Mohit Sharma and George Karypis. 2019. Adaptive matrix completion for the users and the items in tail. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313736

1 INTRODUCTION

Recommender systems are used in e-commerce, social networks, and web search to suggest the most relevant items to each user. Recommender systems commonly use methods based on Collaborative Filtering [15], which rely on historical preferences of users over items in order to generate recommendations. These methods predict the ratings for the items not rated by the user and then select the unrated items with the highest predicted ratings as recommendations to the user.

In practice, a user may not rate all the available items, and hence we observe only a subset of the user-item rating matrix. For the task of recommendations, we need to complete the matrix by predicting the missing ratings and select the unrated items with high predicted ratings as recommendations for a user. The matrix completion approach [3] assumes that the user-item rating matrix is low rank and estimates the missing ratings based on the observed ratings in the matrix. The state-of-the-art collaborative filtering methods, e.g., Matrix Factorization (MF) [10] are based on the matrix completion approach.

Assuming that the user-item rating matrix is low-rank, it was shown that in order to accurately recover the underlying low-rank model of a $n \times n$ matrix of rank r, at least $O(nr \log(n))$ entries in the matrix should be sampled uniformly at random [4]. However, most real-world rating matrices exhibit a skewed distribution of ratings as some users have provided ratings to few items and certain items have received few ratings from the users. This skewed distribution may result in insufficient ratings for certain users and items, and can negatively affect the accuracy of the matrix completion-based methods.

This paper investigates how the skewed distribution of ratings in the user-item rating matrix affects the accuracy of the matrix completion-based methods and shows by extensive experiments on different low-rank synthetic datasets and as well as on real datasets that the matrix completion-based methods tend to have poor accuracy for the items and the users with few ratings. Moreover, this work illustrates that as we increase the number of latent dimensions, the prediction performance for the items and the users with sufficiently many ratings continues to improve, whereas the accuracy of the items and the users with few ratings degrades. This suggests that because of over-fitting, the matrix completion-based methods for large number of latent dimensions do not generalize well for the items and the users with few ratings.

Building on this finding, we develop four matrix completion-based approaches that explicitly consider the number of ratings received by an item or provided by a user to estimate the rating of the user on the item. Specifically, we introduce (i) Frequency Adaptive Rating Prediction (FARP) method, which uses multiple low-rank models for different frequency of the users and the items; (ii) Truncated Matrix Factorization (TMF) method, which estimates a single low-rank model that adapts with the number of ratings a user and an item has; (iii) Truncated Matrix Factorization with Dropout (TMF + Dropout) method, which is similar to TMF but probabilistically select the ranks for the users and the items; and (iv) Inverse Frequency Weighted Matrix Factorization (IFWMF) method,

This work was supported in part by NSF (1447788, 1704074, 1757916, 1834251), Army Research Office (W911NF1810344), Intel Corp, and the Digital Technology Center at the University of Minnesota. Access to research and computing facilities was provided by the Digital Technology Center and the Minnesota Supercomputing Institute.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313736

which weighs the infrequent users and items higher during low-rank model estimation. Extensive experiments on various datasets demonstrate the effectiveness of the proposed approaches over traditional MF-based methods by improving the accuracy for the items (up to 53% improvement in RMSE) and the users (up to 8% improvement in RMSE) with few ratings.

2 RELATED WORK

The current state-of-the-art methods for rating prediction are based on matrix completion, and most of them involve factorizing the user-item rating matrix [7, 9, 10]. In this work, our focus is on analyzing the performance of the matrix completion-based MF approach and use the derived insights to develop an approach that performs better for the users and the items with few ratings in the user-item rating matrix. These approaches estimate user-item rating matrix as a product of two low-rank matrices known as the user and the item latent factors. If for a user u, the vector $\boldsymbol{p}_u \in \mathbb{R}^r$ denotes the r dimensional user's latent factor and similarly for the item i, the vector $\boldsymbol{q}_i \in \mathbb{R}^r$ represents the r dimensional item's latent factor, then the predicted rating ($\hat{r}_{u,i}$) for user u on item i is given by

$$\hat{r}_{u,i} = \boldsymbol{p}_u \boldsymbol{q}_i^T. \tag{1}$$

The user and the item latent factors are estimated by minimizing a regularized square loss between the actual and predicted ratings

$$\underset{\boldsymbol{p}_u, \boldsymbol{q}_i}{\text{minimize}} \quad \frac{1}{2} \sum_{r_{ui} \in R} \left(r_{ui} - \boldsymbol{p}_u \boldsymbol{q}_i^T\right)^2 + \frac{\beta}{2} \left(||\boldsymbol{p}_u||_2^2 + ||\boldsymbol{q}_i||_2^2\right), \tag{2}$$

where R is the user-item rating matrix, r_{ui} is the observed rating of user u on item i, and parameter β controls the Frobenius norm regularization of the latent factors to prevent overfitting.

In another related work [21], it was shown that the lack of uniform distribution of ratings in the observed data could lead to folding, i.e., the unintentional affinity of dissimilar users and items in the low-rank space estimated by matrix completion-based methods. For example, the absence of an explicit rating between a child viewer and a horror movie could lead to an estimation of the corresponding latent factors such that both the child viewer and the horror movie are close in the low-rank space leading to the erratic recommendation of horror movies to the child viewer. We believe that the non-uniform distribution of the ratings is the reason behind this phenomenon.

The non-uniform distribution of ratings can also be viewed as an instance of rating data *missing not at random* (MNAR) [12]. The proposed solutions to MNAR model the missing data to improve the generated recommendations [13, 17, 19, 20]. However, in our work, we focus on the skewed distribution of ratings which often comes as a result of either new items or new users that are added to the system, or the items that are not popular to get many ratings. We use our analysis to develop a matrix completion-based approach to improve rating prediction for the users who have provided few ratings on items or for the items that have received few ratings from the users.

Table 1: Datasets used in experiments

Dataset	users	items	ratings	μ_u[a]	σ_u[b]	μ_i[a]	σ_i[b]	%[†]
Flixster (FX)	147K	48K	8.1M	55	226	168	934	1e-1
Movielens (ML)	229K	26K	21M	92	190	786	3269	3e-1
Yahoo Music (YM)	143K	136K	9.9M	69	199	73	141	5e-4
Netflix (NF)	354K	17K	9.5M	27	59	535	1693	1e-3

[a] Average ratings per user (μ_u) or per item (μ_i).

[b] Standard deviation of ratings per user (σ_u) or per item (σ_i).

[†] The percentage of observed ratings in the dataset.

3 IMPACT OF SKEWED DISTRIBUTION

As described in Section 1, the matrix completion-based methods can accurately recover the underlying low-rank model of a given low-rank matrix provided entries are observed uniformly at random from the matrix. However, the ratings in the user-item rating matrix in real-world datasets represent a skewed distribution of entries because some users have provided ratings to few items and certain items have received few ratings from the users.

In order to study how the skewed distribution of ratings in real datasets affects the ability of matrix completion to accurately complete the matrix (i.e., predict the missing entries) we performed a series of experiments using synthetically generated low-rank rating matrices. In order to generate a rating matrix $R \in \mathbb{R}^{n \times m}$ of rank r we followed the following protocol. We started by generating two matrices $A \in \mathbb{R}^{n \times r}$ and $B \in \mathbb{R}^{m \times r}$ whose values are uniformly distributed at random in $[0, 1]$. We then computed the singular value decomposition of these matrices to obtain $A = U_A \Sigma_A V_A^T$ and $B = U_B \Sigma_B V_B^T$. We then let $P = \alpha U_A$ and $Q = \alpha U_B$ and $R = PQ^T$. Thus, the final matrix R of rank r is obtained as the product of two randomly generated rank r matrices whose columns are orthogonal. The parameter α was determined empirically in order to produce ratings in the range of $[-10, 10]$.

We used the above approach to generate full rating matrices whose dimensions are those of the two real-world datasets, i.e., Flixster (FX) and Movielens (ML), shown in Table 1. For each of these matrices we select the entries that correspond to the actual user-item pairs that are present in the corresponding dataset and give it as input to the matrix completion algorithm. For each dataset we generated five different sets of matrices using different random seeds and we performed a series of experiments using synthetically generated low-rank matrices of rank 5 and 20. For each rank, we report the average of performance metrics in each set from the estimated low-rank models over all the synthetic matrices.

3.1 Results

3.1.1 Effect of item frequency in synthetic datasets. In order to investigate if the number of ratings an item has, i.e., item frequency, has any influence on the accuracy of the matrix completion-based methods for the item, we ordered all the items in decreasing order by their frequency in the rating matrix. Furthermore, we divided these ordered items into ten buckets and for a user computed the RMSE for items in each bucket based on the error between the predicted rating by the estimated low-rank model and the ground-truth rating. We repeated this for all the users and computed the average of the RMSE of the items in each bucket over all the users.

Figure 1 shows the RMSEs across the buckets along with the average frequency of the items in the buckets. As can be seen in

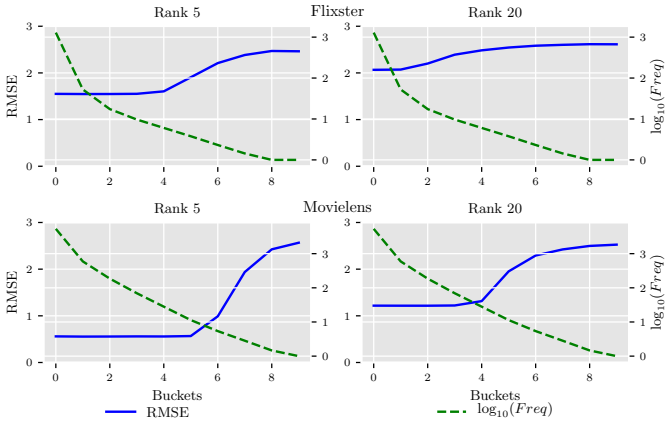


Figure 1: RMSE of the predicted ratings as the frequency of the items decreases.

the figure, the predicted ratings for the frequent items tend to have lower RMSE in contrast to infrequent items for all the datasets.

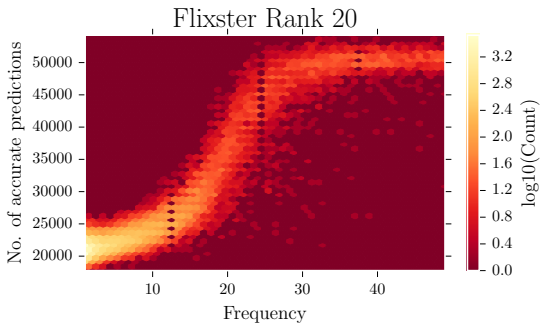


Figure 2: Scatter map of items having different frequency against their number of accurate predictions (Mean absolute error (MAE) $\leq$ 0.5) for low-rank model with rank 20.

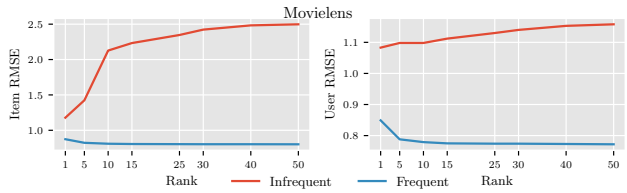


Figure 3: Variation in Test RMSE with increase in rank for the items and the users with different frequency.

Figure 2 shows the scatter map of items in FX having different frequency against the number of instances where the absolute difference between the original and the predicted rating, i.e., *Mean Absolute Error (MAE)*, is $\leq$ 0.5. As can be seen in the figure, the number of accurate predictions is significantly lower for items having fewer ratings ($\leq$ 20) compared to that of the items having a large number of ratings ($\geq$ 30). The lower error of the frequent items is because they have sufficient ratings to estimate their latent factors accurately. Hence for the real datasets, items appearing at the top in ordering by frequency and having high predicted scores will form a reliable set of recommendations to a user.

3.1.2 Effect of frequency on accuracy in real datasets. In order to assess the finding that the infrequent items are not estimated accurately by the matrix completion method, we evaluated matrix completion on a random held-out subset of the real datasets. We followed the standard procedure of cross-validation and exhaustive grid search for hyperparameters for model selection. We computed RMSE over the infrequent items in the test split, i.e., the items that have few ratings in the training split. For the analysis, we ordered the items in increasing order by the number of ratings in training splits. Next, we divided these ordered items into quartiles and identified the items in the first and the last quartile as the infrequent and the frequent items, respectively.

Figures 3 show the RMSE for the items and the users in the test for the Movielens (ML) dataset. We can see that the RMSE of the frequent items (or users) is lower than that of the infrequent items (or users). Furthermore, we observed similar trends in the remaining datasets (results not shown here due to space constraints). These results suggest that the matrix completion method fails to estimate the preferences for the infrequent items (or users) accurately in the real datasets. Also, the RMSE of the infrequent items increases with the increase in the rank while that of frequent items decreases with the increase in the rank. Similarly, the RMSE of the infrequent users increases with the increase in the rank. The increase in RMSE with the increase in ranks suggests that infrequent items or infrequent users may not have sufficient ratings to estimate all the ranks accurately thereby leading to the error in predictions for such users or items. The finding that infrequent items or infrequent users have better accuracy for fewer ranks follows from the result that $O(nr \log(n))$ entries are required to recover the underlying low-rank model of a $n \times n$ matrix of rank r [4], and therefore for fewer entries (e.g., infrequent users or infrequent items) we may recover only fewer ranks of the underlying low-rank model accurately.

4 METHODS

The analysis presented in the previous section showed that as the underlying rank of the low-rank model that describes the data increases, the error associated with estimating such a low-rank model from the skewed data increases for the infrequent users and the infrequent items. We use these observations to devise multiple approaches to improve the accuracy of the low-rank models for such users and items.

4.1 Frequency Adaptive Rating Prediction (FARP)

Since the error of the predictions from the estimated low-rank models increases for the infrequent users or items in skewed data, we propose to estimate lower dimensional latent factors for the infrequent users or items, and estimate higher dimensional latent factors for the frequent users or items. In this approach, we propose to learn multiple low-rank models with different ranks from all the available data and while predicting the rating of a user on an item we select the model that performed the best for the infrequent user or the item associated with the rating. Hence, the predicted rating

of user u on item i is given by

$$\hat{r}_{ui} = \boldsymbol{p}_{uk}\boldsymbol{q}_{ik}^T, \tag{3}$$

where $\boldsymbol{p}_{uk}$ and $\boldsymbol{q}_{ik}$ are the user and the item latent factors from the kth low-rank model. For example, if $f_u < f_i$ then we select the kth low-rank model for prediction such that it has the best performance for users having frequency f_u, and similarly if $f_i < f_u$ then we select the model with the best performance for items with frequency f_i.

The user and items can be assigned to different low rank models based on the number of ratings that exists for them in the dataset. One approach that we investigated is to order the users and the items by the number of ratings and divide them into equal quartiles and save the best performing model for each quartile.

4.2 Truncated Matrix Factorization (TMF)

An alternate approach we develop is to estimate only a subset of the ranks for these users or items. In this approach, the estimated rating for user u on item i is given by

$$\hat{r}_{u,i} = \boldsymbol{p_u}(\boldsymbol{q_i} \odot \boldsymbol{h_{u,i}})^T, \tag{4}$$

where $\boldsymbol{p_u}$ denotes the latent factor of user u, $\boldsymbol{q_i}$ represents the latent factor of item i, $\boldsymbol{h_{u,i}}$ is a vector containing 1s in the beginning followed by 0s, and $\odot$ represents the elementwise Hadamard product between the vectors. The vector $\boldsymbol{h_{u,i}}$ is used to select the ranks that are *active* for the (u, i) tuple. The 1s in $\boldsymbol{h_{u,i}}$ denote the active ranks for the (u, i) tuple.

4.2.1 Frequency adaptive truncation. One approach that we investigated for selecting the active ranks, i.e., $\boldsymbol{h_{u,i}}$, for a user-item rating is based on the frequency of the user and the item in the rating matrix. In this approach, for a given rating by a user on an item, first, we determine the number of ranks to be updated based on either the user or the item depending on the one having a lower number of ratings. In order to select the ranks, we normalize the frequency of the user and the item, and use a non-linear activation function, e.g., sigmoid function, to map this frequency of the user or the item in $[0, 1]$. Finally, we used this mapped value as the number of active ranks selected for the update of the user and the item latent factors. The number of active ranks to be selected is given by

$$k_{u,i} = \frac{r}{1 + e^{-k(f_{min} - z)}}, \tag{5}$$

where r is the dimension of the user and the item latent factors, $f_{min} = \min(f_u, f_i)$, f_u is the frequency of user u, f_i is the frequency of item i, k controls the steepness of the sigmoid function and z is the value of the sigmoid's midpoint. The use of such a function assists in identifying the users or the items that can not be estimated accurately using all the ranks and we can only estimate few ranks more accurately for such users or items. The active ranks for a user or an item can be chosen either from the beginning of all ranks or end of all ranks or can be chosen arbitrarily among all ranks until the same active ranks are used consistently for the user and the item. For ease of discussion and simplicity, we will assume that active ranks are chosen from the beginning of all ranks. Hence, the active ranks to be selected are given by

$$\boldsymbol{h_{u,i}}[j] = \begin{cases} 1, & \text{if } j \le k_{u,i} \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

We will refer to this method as Truncated Matrix Factorization (TMF).

4.2.2 Frequency adaptive probabilistic truncation. An alternative way to select the active ranks is to assume that the number of active ranks follows a Poisson distribution with parameter $k_{u,i}$. This method is similar to Dropout [18] technique in neural networks, where parameters are selected probabilistically for updates during learning of the model. Similar to regularization it provides a way of preventing overfitting in learning of the model. The active ranks to be selected are given by

$$\boldsymbol{h_{u,i}}[j] = \begin{cases} 1, & \text{if } j \le \theta_{u,i} \\ 0, & \text{otherwise,} \end{cases}$$

where $\theta_{u,i} \sim Poisson(k_{u,i})$. We will call this method as Truncated Matrix Factorization with Dropout (TMF + Dropout). Similar to Equation 2, the parameters of the model, i.e., the user and the item latent factors can be estimated by minimizing a regularized square loss between the actual and the predicted ratings.

4.2.3 Rating prediction. After learning the model the predicted rating for user u on item i for TMF model is given by

$$\hat{r}_{u,i} = \boldsymbol{p_u}(\boldsymbol{q_i} \odot \boldsymbol{h}_{u,i})^T, \tag{7}$$

where the active ranks, i.e., $\boldsymbol{h_{u,i}}$, is given by Equation 6. The predicted rating for the user and the item under TMF + Dropout model is given by the least number of ranks for whom the cumulative distribution function (CDF) for Poisson distribution with parameter k_{ui} obtains approximately the value of 1. The active ranks, i.e., $\boldsymbol{h_{u,i}}$, for prediction under TMF + Dropout are given by

$$\boldsymbol{h_{u,i}}[j] = \begin{cases} 1, & \text{if } j \le s \\ 0, & \text{otherwise,} \end{cases}$$

where s is the least number of ranks for whom the CDF, i.e, $P(x <= s) \approx 1$ and $x \sim Poisson(k_{u,i})$.

Unlike FARP, which requires us to estimate multiple models, TMF estimates a single model however it involves tuning of more hyperparameters in comparison to FARP.

4.3 Inverse Frequency Weighted Matrix Factorization (IFWMF)

In addition to the above approaches, we explored a weighted matrix factorization-based approach where we weigh the reconstruction error higher for the infrequent users and items. We propose to estimate the user and the item latent factors by minimizing a regularized weighted square error loss between the actual and the predicted ratings

$$\underset{\boldsymbol{p_u}, \boldsymbol{q_i}}{\text{minimize}} \quad \frac{1}{2}\sum_{r_{ui} \in R} w_{ui}\left(r_{ui} - \boldsymbol{p}_u\boldsymbol{q}_i^T\right)^2 + \frac{\beta}{2}\left(||\boldsymbol{p}_u||_2^2 + ||\boldsymbol{q}_i||_2^2\right), \tag{8}$$

where the the weight w_{ui} is given by

$$w_{ui} = \frac{1}{1.0 + \rho f_{\text{min}}}, \tag{9}$$

where ρ is a constant, $f_{\min} = \min(f_u, f_i)$, f_u and f_i are the normalized frequency of user u and item i, respectively. Essentially, we weigh the error in predictions more for the infrequent users and the infrequent items. This resembles the weighted matrix factorization [7, 8] where the weight of the reconstruction error is proportional to the confidence on the observed rating of user u on item i however in our method we weigh the error inversely proportional to the frequency of ratings observed for user u or item i. This is similar to the inverse propensity model-based approach [17], where the propensity is proportional to the frequency of the user or the item. The up-weighting of the reconstruction error associated with the infrequent users or the infrequent items may lead to over-fitting as we only have few ratings for these users and items.

5 EXPERIMENTAL EVALUATION

5.1 Comparison algorithms

We compared our proposed approaches against the the state-of-the-art Matrix Factorization [9] and LLORMA [11] method. LLORMA assumes that the different parts of the user-item rating matrix can be approximated by different low-rank models and the complete user-item rating matrix is approximated as a weighted sum of these individual low-rank models. We have used the LibRec [5] software package to compare the proposed methods against the LLORMA approach.

5.2 Model selection

We performed grid search to tune the dimensions of the latent factors, regularization hyper-parameters, constant (ρ), and sigmoid function's parameters, i.e., k and z. We searched for regularization weights (λ) in the range [0.001, 0.01, 0.1, 1, 10], dimension of latent factors (r) in the range [1, 5, 10, 15, 25, 50, 75, 100], constant (ρ) in the range [1, 10, 50], steepness constant (k) in the range [1 5, 10, 20, 40], and mid-point (z) in the range [-0.75, -0.50, -0.25, 0, 0.25, 0.50, 0.75]. The final parameters were selected based on the performance on the validation split. For LLORMA, we varied the number of local models (l_m) in the range [1, 5, 8, 15, 25, 50].

For FARP, we ordered the users in ascending order by frequency and divided them into equal quartiles. For each quartile, we saved the best performing model, i.e., the model having the lowest RMSE for all the users in that quartile in the validation split. Similarly, we ordered the items in ascending order by frequency and divided them into equal quartiles. Similar to users, we saved the best performing model for each quartile of items. At the time of prediction of rating for a user on an item, we choose the the model associated with the quartile of the user if the user is having lower number of ratings than the item, and if the item is having lower number of ratings than the user than we choose the model associated with the quartile of the item .

5.3 Datasets

In addition to Flixster (FX) [22] and Movielens 20M (ML) [6] datasets, we evaluated our proposed methods on subsets of the Yahoo Music (YM) [1, 16] and Netflix (NF) [2] datasets that we created in order to have a skewed distribution. These datasets were generated as follow. First, for each user we randomly selected the number of ratings that we want to sample from the user's ratings and randomly sampled these ratings for all users. Next, from the sampled ratings in previous step, for each item we randomly selected the number of ratings that an item has received from users in sampled ratings and randomly sampled these ratings for all the items. After following the above two steps, the sampled ratings from these datasets follows a skewed distribution and characteristics of all the datasets used in experiments are presented in Table 1.

5.4 Evaluation methodology

To evaluate the performance of the proposed methods we divided the available ratings in different datasets into training, validation and test splits by randomly selecting 20% of the ratings for each of the validation and the test splits. The validation split was used for model selection, and the model that was selected was used to predict ratings on the test split. We repeated this process three times and report the average RMSE across the runs.

In addition to computing RMSE obtained by different methods for the ratings in the test split, we also investigated the performance of the proposed approaches for the items and the users with a different number of ratings in the training split. To this end, we ordered the items and the users in increasing order by their number of ratings in training split and divided them equally into quartiles. We will report the RMSE achieved by different methods for ratings in the test split for the users and the items in these quartiles.

6 RESULTS AND DISCUSSION

6.1 Performance for rating prediction on entire dataset

Table 2 shows the results achieved by the proposed methods on the various datasets. As can be seen in the table for the task of rating predictions for all the ratings in the test splits the proposed approaches perform better than the MF method for FX, ML, YM and NF datasets. Interestingly, the proposed approaches have performed even better than the state-of-the-art LLORMA method and this suggests that LLORMA can be further improved by estimating local low-rank models that considers the skewed distribution of ratings in datasets. We found the difference between the predictions of different methods to be statistically significant (p-value ≤ 0.01 using two sample t-test). The performance is significantly better for FX in comparison to that of other datasets. Additionally, on FX dataset, the MF method outperforms the LLORMA method and a possible reason for this is that because of the skewed distribution LLORMA is not able to estimate a model that is as accurate as a global MF model. Moreover, LLORMA needs a significantly large number of local low-rank models in comparison to the proposed approaches. Also, by comparing the number of latent dimensions used by the models shown in Table 2 we can see that, for ML, TMF + Dropout and LLORMA needs significantly fewer ranks, i.e., 25, in comparison to that of MF, i.e., 100, to achieve the same performance and we believe that this could be because of MF overfitting the ML dataset for higher dimension of latent factors.

Table 2: Test RMSE of the proposed approaches for different datasets. The RMSE for the users and the items in different quartiles order by their frequency. Q1 refers to the quartile containing the least frequent users or items followed by remaining in Q2, Q3, and Q4. Table 4 shows the average number of test ratings in different quartiles for different datasets.

	Flixster (FX)						Movielens (ML)					
	MF	LLORMA	IFWMF	FARP	TMF	TMF + Dropout	MF	LLORMA	IFWMF	FARP	TMF	TMF + Dropout
No. of low-rank models	1	20	1	8	1	1	1	8	1	8	1	1
Rank	15	50	15	NA†	10	15	100	25	100	NA†	100	25
All test ratings	0.864	0.871	0.867	0.863	0.851	0.847	0.806	0.806	0.804	0.797	0.804	0.804
Item Q1	1.302	1.705	1.289	1.258	1.252	1.256	2.527	1.501	2.382	1.178	2.377	2.115
Item Q2	0.961	1.099	0.961	0.962	0.944	0.944	1.619	0.974	1.449	0.937	1.499	1.123
Item Q3	0.800	0.863	0.801	0.798	0.785	0.780	0.891	0.841	0.869	0.855	0.866	0.851
Item Q4	0.864	0.865	0.867	0.863	0.851	0.847	0.799	0.805	0.799	0.795	0.798	0.801
User Q1	1.292	1.388	1.261	1.246	1.247	1.260	1.174	1.092	1.120	1.083	1.115	1.078
User Q2	1.177	1.255	1.156	1.143	1.144	1.151	0.975	0.964	0.965	0.965	0.964	0.968
User Q3	0.974	1.002	0.969	0.967	0.964	0.964	0.853	0.856	0.852	0.853	0.852	0.863
User Q4	0.853	0.852	0.857	0.853	0.841	0.836	0.767	0.774	0.769	0.761	0.769	0.769
	Yahoo Music (YM)*						Netflix (NF)*					
	MF	LLORMA	IFWMF	FARP	TMF	TMF + Dropout	MF	LLORMA	IFWMF	FARP	TMF	TMF + Dropout
No. of low-rank models	1	25	1	8	1	1	1	25	1	8	1	1
Rank	100	5	75	NA†	75	75	75	25	75	NA†	75	75
All test ratings	1.170	1.177	1.162	1.152	1.163	1.164	0.906	0.903	0.901	0.901	0.903	0.904
Item Q1	1.245	1.250	1.238	1.235	1.239	1.234	1.067	1.059	1.054	1.047	1.051	1.060
Item Q2	1.165	1.171	1.159	1.152	1.161	1.162	1.018	1.006	1.007	1.005	1.008	1.014
Item Q3	1.154	1.164	1.149	1.139	1.151	1.152	0.967	0.964	0.956	0.955	0.959	0.962
Item Q4	1.170	1.172	1.161	1.150	1.162	1.163	0.896	0.893	0.891	0.892	0.893	0.894
User Q2	1.429	1.452	1.412	1.408	1.407	1.395	1.281	1.378	1.258	1.274	1.259	1.256
User Q3	1.236	1.231	1.220	1.210	1.224	1.225	1.018	1.011	1.008	1.006	1.010	1.013
User Q4	1.155	1.160	1.148	1.138	1.150	1.15	0.879	0.874	0.875	0.875	0.877	0.877

†The ranks used for FARP are in Table 3.

*Due to sampling, the test splits are not having any ratings for users in Q1 for these datasets.

Table 3: Rank used by FARP for different datasets.

	Item				User			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
FX	1	1	15	15	1	1	1	15
ML	1	5	10	100	1	15	30	100
YM	75	75	100	100	5	5	100	100
NF	50	50	50	75	50	50	50	75

Table 4: Average number of ratings in quartiles of test splits.

	Item				User			
	Q1 (10^3)	Q2 (10^3)	Q3 (10^3)	Q4 (10^6)	Q1 (10^3)	Q2 (10^3)	Q3 (10^3)	Q4 (10^6)
FX	4.8	6.3	22.0	1.4	16.5	15.9	54.7	1.4
ML	2.7	9.5	92.5	3.7	111.2	194.8	533.5	2.9
YM	83.4	175.4	303.2	1.2	NA†	38.9	181.4	1.6
NF	13.6	40.1	136.0	1.5	NA†	32.0	209.8	1.4

†Due to sampling, the test splits do not have ratings for users in Q1.

6.2 Performance for the users and the items with different number of ratings

Table 2 also shows the performance achieved by the different methods across the different quartiles of users and items. By comparing the performance of the different schemes we can see that the proposed methods significantly outperform the MF method and state-of-the-art LLORMA method for lower quartiles for majority of the datasets. This illustrates the effectiveness of the developed methods for the users and the items with few ratings. The better performance of the proposed methods for the users and the items with few ratings is because we can estimate accurately only a few ranks for them, and unlike MF and LLORMA the proposed approaches are effective in model estimation or predicting the ratings for these users and items.

Among the proposed approaches, the TMF-based approaches (TMF and TMF + Dropout), perform better in most of the quartiles in FX dataset. Surprisingly, FARP consistently performs better than the MF and LLORMA across most of the datasets, and this is promising as compared to TMF-based approaches FARP has fewer hyperparameters to tune. Specifically, in TMF-based approaches we have to tune regularization weights (λ), dimension of latent factors (r), steepness constant (k) and mid-point (z), while in FARP we have to tune only three parameters, i.e., number of low-rank models, regularization weights (λ) and dimension of latent factors (r). This might be of interest to practitioners because multiple low-rank models under FARP can be estimated in parallel using vanilla MF widely available in off-the-shelf packages, e.g., SPARK [23] and scikit-learn [14].

While the IFWMF method performs better than MF in lower quartiles, the other proposed approaches, i.e., FARP and TMF-based methods, outperform IFWMF for most of the quartiles in all the dataset thereby illustrating the effectiveness of FARP and TMF-based methods in preventing over-fitting and generating better predictions.

7 CONCLUSION

In this work, we have investigated the performance of the matrix completion-based low-rank models for estimating the missing ratings in real datasets and its impact on the item recommendations. We showed in Section 3 that the matrix completion-based methods because of skewed distribution of ratings fail to predict the missing entries accurately in the matrices thereby leading to an error in predictions and thus affecting item recommendations. Based on these insights we presented different methods in Section 4, which considers the frequency of both the user and the item to estimate the low-rank model or for predicting the ratings. The experiments on real datasets show that the proposed approaches significantly outperforms the state-of-the-art Matrix Factorization method for rating predictions for the users and the items having few ratings in the user-item rating matrix.

REFERENCES

[1] 2008. Yahoo! Music User Ratings of Songs with Artist, Album, and Genre Meta Information, v. 1.0. https://webscope.sandbox.yahoo.com/

[2] James Bennett, Stan Lanning, et al. 2007. The netflix prize. In *Proceedings of KDD cup and workshop*, Vol. 2007. New York, NY, USA, 35.

[3] Emmanuel J. Candès and Benjamin Recht. 2009. Exact Matrix Completion via Convex Optimization. *Foundations of Computational Mathematics* 9, 6 (2009), 717–772. https://doi.org/10.1007/s10208-009-9045-5

[4] Emmanuel J. Candès and Terence Tao. 2010. The Power of Convex Relaxation: Near-optimal Matrix Completion. *IEEE Trans. Inf. Theor.* 56, 5 (May 2010), 2053–2080. https://doi.org/10.1109/TIT.2010.2044061

[5] Guibing Guo, Jie Zhang, Zhu Sun, and Neil Yorke-Smith. 2015. LibRec: A Java Library for Recommender Systems.. In *UMAP Workshops*, Vol. 4.

[6] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2016), 19.

[7] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*. Ieee, 263–272.

[8] Noam Koenigstein and Yehuda Koren. 2013. Towards scalable and accurate item-oriented recommendations. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 419–422.

[9] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.

[10] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37. https://doi.org/10.1109/MC.2009.263

[11] Joonseok Lee, Seungyeon Kim, Guy Lebanon, and Yoram Singer. 2013. Local low-rank matrix approximation. In *International Conference on Machine Learning*. 82–90.

[12] Roderick JA Little and Donald B Rubin. 2014. *Statistical analysis with missing data*. Vol. 333. John Wiley & Sons.

[13] Benjamin M Marlin and Richard S Zemel. 2009. Collaborative prediction and ranking with non-random missing data. In *Proceedings of the third ACM conference on Recommender systems*. ACM, 5–12.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[15] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW '01)*. ACM, New York, NY, USA, 11.

[16] Sebastian Schelter, Stephan Ewen, Kostas Tzoumas, and Volker Markl. 2013. All roads lead to rome: optimistic recovery for distributed iterative data processing. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM, 1919–1928.

[17] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. *arXiv preprint arXiv:1602.05352* (2016).

[18] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[19] Harald Steck. 2010. Training and Testing of Recommender Systems on Data Missing Not at Random. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '10)*. ACM, New York, NY, USA, 713–722. https://doi.org/10.1145/1835804.1835895

[20] Harald Steck. 2013. Evaluation of Recommendations: Rating-prediction and Ranking. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys '13)*. ACM, New York, NY, USA, 213–220. https://doi.org/10.1145/2507157.2507160

[21] Doris Xin, Nicolas Mayoraz, Hubert Pham, Karthik Lakshmanan, and John R Anderson. 2017. Folding: Why Good Models Sometimes Make Spurious Recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 201–209.

[22] R. Zafarani and H. Liu. 2009. Flixster dataset, Social Computing Data Repository at ASU. http://socialcomputing.asu.edu/datasets/Flixster

[23] Matei Zaharia, Reynold S. Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J. Franklin, Ali Ghodsi, Joseph Gonzalez, Scott Shenker, and Ion Stoica. 2016. Apache Spark: A Unified Engine for Big Data Processing. *Commun. ACM* 59, 11 (Oct. 2016), 56–65. https://doi.org/10.1145/2934664

What is in Your Password? Analyzing Memorable and Secure Passwords using a Tensor Decomposition

Youjin Shin
The State University of New York, Korea
Incheon, South Korea
Stony Brook University, NY, USA
youjin.shin.1@stonybrook.edu

Simon S. Woo*
Department of Applied Data Science
SKKU Institute for Convergence
Sungkyunkwan University, Suwon, South Korea
swoo@g.skku.edu

ABSTRACT

In the past, there have been several studies in analyzing password strength and structures. However, there are still many unknown questions to understand what really makes passwords both memorable and strong. In this work, we aim to answer some of these questions by analyzing password dataset through the lenses of data science and machine learning perspectives. We use memorable 3,260 password dataset collected from prior IRB-approved user studies over 3 years and classify passwords into three strength groups using online and offline attack limits. Then, we apply a *tensor decomposition* to analyze password dataset by constructing a 3rd-order tensor with passwords' syntactic and semantic features. In particular, we used PARAFAC2 tensor decomposition to uncover the main characteristics and features that affect password strength. We quantitatively identified the underlying factors that are more frequently observed in strong and memorable passwords. We hope that our finding can validate widely accepted advice for creating strong passwords and provide useful insights to design a better password suggestion system.

CCS CONCEPTS

• **Security and privacy → Authentication**; • **Computing methodologies → Factorization methods**.

KEYWORDS

Password, Authentication, Tensor Decomposition, PARAFA2, Classification

ACM Reference Format:
Youjin Shin and Simon S. Woo. 2019. What is in Your Password? Analyzing Memorable and Secure Passwords using a Tensor Decomposition. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313690

1 INTRODUCTION

Although there have been several attempts to find strong alternatives to textual passwords, passwords are still widely used for many applications to authenticate users due to simplicity and convenience. However, textual passwords require two seemingly conflicting security and memorability requirements in order to serve today's user authentication needs. That is, users need to create both memorable and strong passwords. However, there is a clear tension between memorability and security. Strong passwords such as random strings are difficult to remember, while memorable passwords are easy to be guessed. Therefore, creating both strong and memorable passwords is a challenging task for a user. We, as security researchers, have to help users create both memorable and strong passwords, based on qualitative evidence and quantitative measurements.

To address these issues, several research [7, 9, 10, 19, 23, 25, 33, 34] have tackled the problems of analyzing and understanding the detailed structures of passwords and have shed lights on how people create memorable passwords. Also, Ur et al. [30] have uncovered some of users' misconception about strong passwords and users' detailed strategy to create strong passwords through small-scale lab study. However, prior research has heavily relied on leaked passwords dataset, which does not provide information on what passwords are memorable. Also, a small-scale qualitative study lacks on generalizing results based on quantitative measures. In addition, there are limited research in understanding the structures of passwords using the latest machine learning and data analysis techniques to identify latent factors that influence the creation of strong and memorable passwords.

In this research, we aim to understand syntactic and semantic features that strongly influence the creation of both strong and memorable passwords, using a *tensor decomposition*. A tensor decomposition is widely exploited and adopted in data mining, statistics, bioinformatics, psychology, physics, computer vision, and signal processing areas as a powerful tool to model a wide variety of heterogeneous, multiaspect, and higher-order data [24]. Especially, a tensor decomposition can effectively extract useful latent information out of multiaspect data tensors. In this work, in order to complement prior research, we consider passwords as multi-dimensional tensors and perform a tensor decomposition to uncover and validate the characteristics of both memorable and strong passwords. We leverage the memorable password dataset collected from prior IRB-approved user study by Woo et al. [36] as a baseline password dataset. Furthermore, we measure password strength in guess numbers using Monte-Carlo sampling method [13] and classify them to weak, medium, and strong strength group based on online (10^6 guesses) and offline guessing limit (10^{14} guesses) [14]. This allows us to analyze the characteristics of both memorable and

*Corresponding Author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313690

strong password structures using a tensor. Our key contributions are summarized as follows:

- We are the first to model passwords as tensors and apply a tensor decomposition to extract useful underlying features from passwords.
- We further provide the explainability on the syntactic and semantic features that predominately occur in strong and memorable passwords.
- Through distance analysis, we show and validate our results with findings from prior research.

Our quantitative finding supports that password length is the most dominating feature in explaining strong passwords. On the other hand, the predominant use of dictionary words are shown in weak memorable passwords. Our work provides the confirmation and validation of common advice in creating strong passwords, and can help guide users create both strong and memorable passwords.

2 RELATED WORK

Jakobsson and Dhiman [19] attempted to better understand the password structures. However, their research is limited in capturing diverse password patterns. Also, much research has investigated distribution of characters and types of mangling patterns [9, 10, 23, 34]. On the other hand, Ur et al. [30] conducted the small scale qualitative lab studies to uncover the rationale behind average users' misconception about weak and strong passwords. Although they provided very useful insights, their main focus is on the qualitative study. Rao et al. [25] found that people tend to use grammatical structures for long passwords and passphrases. They found that some Part-of-speech (POS) tags are more frequently used than others. In addition, Vera et al. [33] presented the first framework for automatically segmenting, classifying, and capturing the semantic essence of passwords. They employed the natural language processing techniques to effectively segment passwords into more meaningful semantic segments. However, their analysis is based on the leaked password datasets and cannot explain the specific factors influence the strength and memorability of passwords. From our understanding, no research efforts have made to systematically understand both syntactic and semantic latent factors that influence the construction of both strong and memorable passwords from a data science perspective. In this work, we employ a *tensor decomposition* to achieve these goals.

A tensor is a multi-dimensional array [24]. Due to its ability to express multimodal or multiaspect data, it is a very powerful tool in applications that inherently create such data. In mathematics, geometric vectors and scalars can be considered as the simplest tensors. A 1st-order tensor is a vector, a 2nd-order tensor is a matrix, and a 3rd-order tensor is a cube. In general, Nth-order tensor $\boldsymbol{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is represented by the outer product $\circ$ of N vector spaces as follows:

$$\boldsymbol{X} = a^1 \circ a^2 \circ \cdots \circ a^N, \tag{1}$$

where a^N is the vector in Nth dim. A rank (order) in a tensor indicates the number of components in the decomposed matrices and every tensor can be expressed as a sum of a rank-1 tensor [1, 22].

Expressing a tensor as a sum of a rank-1 tensor was first proposed by Hitchcock [17, 18]. Practical algorithms have begun to appear in the late 20th century. In 1970, Carroll and Chang [8] proposed a canonical decomposition (CANDECOMP) and Harshman [16] suggested parallel factor decomposition (PARAFAC), which is an extended version of 2-way factorization for higher-order data. Since CANDECOMP and PARAFAC have same concept, the CANDECOMP/PARAFAC decomposition formulated by Kiers [20] has been widely used. It decomposes Nth-order data into a linear sum of rank-1 tensor as described in Fig. 1.(a) and a 3rd-order tensor can be decomposed into three component matrices A, B, and C.

Harshman [16] suggested PARAFAC2 model to relax the limitation of CANDECOMP/PARAFAC model. Whereas CANDECOMP /PARAFAC decomposition requires the identical factors across a parallel set of matrices, PARAFAC2 requires only one identical factor mode (dimension) and relaxes the other factor matrices to vary. We use PARAFC2 for our password dataset analysis.

The Tucker decomposition is another commonly-used tensor decomposition, which was first introduced by Tucker [27] and refined later [28, 29]. It has a core tensor, which can be viewed as compression of original tenser $\boldsymbol{X}$. For a 3rd-order tensor, it decomposes a tensor into a core tensor and three matrices which have different ranks as shown in Fig. 1.(b).

Applellof and Davidson [6] pioneered the use of CANDECOMP/ PARAFAC model to extract information from a chemical system. Andersson and Bro [5] dedicated developing practical description and application of tensors. And Acar *et al.* [2, 3] was the first case to apply a tensor decomposition to data mining. They analyzed online chatroom data to understand how social groups evolved in cyberspace. They constructed a 3rd-order tensor with $user \times keyword \times time$ spaces. De Lathauwer and Vandewalle [12] applied the Tucker decomposition for dimensionality reduction in higher-order signal processing. Pioneers of the use of the Tucker decomposition in computer vision are Vasilescu and Terzopoulos [31]. They extended conventional Singular Values Decomposition (SVD) to N-mode SVD for a tensor. They also proved that it has significantly better performance than standard Principal Component Analysis (PCA) in image recognition [32].

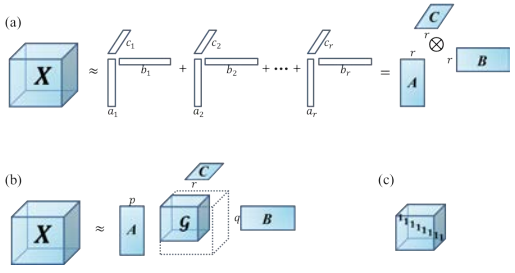


Figure 1: Examples of different tensor decomposition methods from a given 3rd-order tensor: (a) CANDECOMP/PARAFAC (CP) decomposition, where $\boldsymbol{X}$ is an approximated sum of r rank-1 tensors. (b) Tucker decomposition, where $\boldsymbol{X}$ is a rank-(p, q, r) tensor and g is a core tensor. (c) A super-diagonal core tensor with ones in all diagonal values. If the core tensor of Tucker decomposition is a super-diagonal and all of the ranks are same, it can be thought as same as CANDECOMP/PARAFAC decomposition.

3 TENSOR DECOMPOSITION OVERVIEW

Tensor decomposition is one of the most effective unsupervised methods to extract features and characteristics of Nth-dimensional

data and is capable of classifying data in the absence of training labels. Also, tensor decomposition can offer more accurate results by keeping the Nth-order structure of data without any feature mapping such as kernel trick as shown by other research [4, 11, 21].

Tucker decomposition: A given 3rd-order tensor $\boldsymbol{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, the Tucker decomposition is described in Fig. 1.(b). and defined as follows:

$$\boldsymbol{X} \approx \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} \mathbf{g}_{pqr} a_p \circ b_q \circ c_r = [\![\mathbf{g}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!], \tag{2}$$

where $a_p \in \mathbb{R}^{I_1}$ for p=1,. . ., P, $b_q \in \mathbb{R}^{I_2}$ for q=1,. . ., Q, and $c_r \in \mathbb{R}^{I_3}$ for r=1,. . ., R. g is a core tensor.

CANDECOMP / PARAFAC decomposition: As briefly discussed in Section 2, CANDECOMP/PARAFAC decomposition is a special case of the Tucker decomposition and can be constructed as follows:

$$\boldsymbol{X} \approx \sum_{r=1}^{R} \lambda_r a_r \circ b_r \circ c_r = [\![\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!], \tag{3}$$

where R is rank, λ is weights $a_r \in \mathbb{R}^{I_1}$, $b_r \in \mathbb{R}^{I_2}$, and $c_r \in \mathbb{R}^{I_3}$, for r=1,. . .,R. For details of numerical tensor calculus, see Hackbusch's book [15]. CANDECOMP/PARAFAC decomposition can be considered as a special case of the Tucker decomposition where a core is super-diagonal and P = Q = R in Eq. 2. A super-diagonal is illustrated in Fig. 1.(c) [21].

PARAFAC2 decomposition: On the other hand, PARAFAC2 is a weak constraint version of PARAFAC. A given 3rd-order tensor, it is expressed by Acar and Yener [4] as below:

$$\boldsymbol{X}_k = \mathbf{A}\mathbf{D}_k\mathbf{B}^\top + \mathbf{E}_k, \tag{4}$$

where $\boldsymbol{X}_k$ represents a 3rd-order tensor, and $\mathbf{A}$ and $\mathbf{B}$ are the component matrices in the first and second modes, respectively. $\mathbf{D}_k$ is a diagonal matrix, whose diagonal elements are the kth row of the 3rd-component matrix $\mathbf{C}$. $\mathbf{E}_k$ contains residuals. Whereas PARAFAC must apply identical factors across the matrices, PARAFAC2 allows factors of matrices to be different but keeps only one factor invariance. Mathematically, it can be expressed by Acar and Yener [4] as following:

$$\begin{aligned} \boldsymbol{X}_k &= \mathbf{A}_k\mathbf{D}_k\mathbf{B}^\top + \mathbf{E}_k \\ \textit{subject to } &\mathbf{A}_k{}^\top\mathbf{A}_k = \Phi, k = 1, 2, \ldots, K, \end{aligned} \tag{5}$$

where $\mathbf{A}_k{}^\top\mathbf{A}_k = \Phi$ is orthonormal constraint and imposed to improve the uniqueness properties by Harshman [16] and all components are same as above.

4 DATASET

We leveraged the password data collected over three years from various authentication research by Woo et al. [35]. In the course of the study, these passwords were created and were successfully recalled after two days. The dataset includes more than 3,200 passwords. We used these successfully recalled passwords to understand features, which make passwords strong and memorable. First, we measured the strength of memorable passwords, using the Monte Carlo method by Dell'Amico, and Filippone [13]. We trained the guessing algorithm with a total of 21 million leaked passwords. Based on the estimated passwords' strength, we classified each password into the weak, medium or strong category, using the estimated number of guesses for online (10^6 guesses) and offline (10^{14} guesses) attack limits as boundaries between weak, medium, and strong password strength categories [14]. Number of passwords (%) in each strength group are summarized in Table 1. We found that majority of these collected passwords can withstand the online attacks, indicating these are fairly realistic passwords than now easily crackable leaked passwords from RockYou. After classifying memorable passwords into three different strength groups, we generated both *syntactic* and *semantic* features. Then, we applied the PARAFAC2 tensor decomposition to understand the latent syntactic and semantic features among different password strength groups.

Table 1: Memorable password dataset, categorized into three different strength groups

Strength Category	No. of Passwords (%)
Weak (guesses $< 10^6$)	109 (3.34%)
Medium ($10^6 \leq$ guesses $< 10^{14}$)	2,276 (69.82%)
Strong (guesses $\geq 10^{14}$)	875 (26.84%)
Total	3,260 (100%)

Syntactic Feature Generation: We generated 12 syntactic features for each password. First, we calculate the length of password (label: length) and the number of digits (label:D), special chars (label: S), and uppercase letter (label: UL). If there are more than three character classes among digits, special chars, uppercase, and lowercase letters, we define it as a 3class8, which is one of the widely used password composition policy. The number of class flips (change) is counted, when a character changes its class from one to another. Also, we counted the total number of flips (label: tFlips) as well as the following six sub flip cases: lowercase letter to special char (label: flipLS), lowercase letter to digit (label: flipLD), digit to special char (label: flipDS), digit to lowercase letter (label: flipDL), special char to lowercase letter (label: flipSL), and special char to digit (label: flipSD). We extract these features from each password.

Semantic Feature Generation: We used Vera et al.'s semantic segmentation parser [33] to segment the password and label each segment, which uses CLAWS7 tag set using Part-of-speech (POS) tagger by University Centre for Computer Corpus Research on Language (UCREL) [26]. In addition to CLAWS7 tag set, we have the following tags: "special$n$" and "char$m$", which represent the n consecutive special symbols and the m consequence characters. We extracted a total number of 114 different semantic features (tags). Through these initial tag generation process, we obtain a total of 67 features from 12 syntactic and 55 semantic features.

4.1 Pre-processing and Feature Refinement

Before actually constructing a tensor using the dataset with 67 features, we performed the following two pre-processing methods to make analysis much more reliable: 1) near zero variance removal and 2) normalization, which are popular pre-processing techniques.

1) Removing Near Zero Variance features: In one feature, if the majority of values do not vary much, then this feature cannot offer much useful information. And this would have an adverse effect on statistical analysis, since the variance is too small. These features with near zero variance values need to be eliminated prior to actual data analysis. In particular, we remove the near zero variance features for the following cases: the percentage of unique

values is less than 10% and the frequency ratio is greater than 19 (95/5). After removing near zero variance features using these two methods, we obtained a total of 31 features (12 syntactic and 19 semantic features) out of initial 67 features, where 12 syntactic tags are *length, D, S, UL, 3class8, filpLS, flipLD, flipDS, flipDL, flipSL, flipSD*, and *tFlip*. And 19 semantic tags include *article, verb, adjective, pronoun, nn, nn1, nn2, np1, npm, num, char1, char2, char3, special1, special2, number1, number2, number3*, and *number4*.

2) Feature Normalization: Since features are distributed in different ranges at different scale (i.e.: length, number of symbols), we should normalize each feature individually. We applied min-max normalization, which is one of the well-known feature scaling techniques as shown in Eq. 6. Min-max normalization changes all values into the range between 0 and 1, where the minimum value is mapped to 0 and the maximum value is mapped to 1.

$$Z = \frac{x - min(x)}{max(x) - min(x)}, \tag{6}$$

where *min* and *max* are the minimum and maximum value from the given x.

5 EXPERIMENTS

We have three pre-processed datasets for each weak, medium, and strong password group with its 31 features. Since we have 109, 2276, and 875 passwords in *weak, medium*, and *strong* strength group respectively, the size of the *weak* dataset is 109×31, the *medium* dataset is 2276×31, and the *strong* dataset is 875×31, where 31 is the number of features in each password group. Since our dataset has different size of rows across the strength and only one equivalent feature dimension, it cannot be a cube and we need to build up a 3rd-order tensor in different row size as shown in the left side of Fig. 2. The final dimension of our tensor becomes $I[3] \times J \times K = (109, 2276, 875) \times 31 \times 3$ and each dimension represents passwords (I[3]), syntactic and semantic features (J=31), and levels of password strength (K=3) from *weak, medium*, and *strong* as shown in Fig. 2.

Next, we decompose this 3rd-order password tensor into component matrices, **A**, **B**, and **C**, using PARAFAC2 algorithm including orthogonal constraint. After decomposition, we obtain component matrices **A**, **B** and **C** containing, where **A** consists of three password group-to-factor sub matrices ($(109,2276,875) \times r$) showing the factor dependency on each password across the strength matrices. And **B** represents the feature-to-factor matrix ($31 \times r$) indicating how much each feature (syntactic and semantic features) is affected by each factor. Final **C** captures strength-to-factor matrix ($3 \times r$) accounting for how much each factor affect passwords in each strength group.

Selecting an Optimal Rank *r*: When decomposing a tensor, it is critical to choose the optimal factor (rank) r, which is the number of factors to decompose a tensor. However, unlike 2-way factorization, there is no general straightforward algorithm to select the optimal r. Instead, a reconstruction error R_{error} is often used to find r, which is the difference between original tensor $\boldsymbol{X}$ and approximate tensor $\hat{\boldsymbol{X}}$ and it is calculated using the outer product of all matrices. It is defined as the Frobenius norm (Euclidean distance) $\|\boldsymbol{X} - \hat{\boldsymbol{X}}\|_{\mathbf{F}}$. In our case, from the 3rd-order tensor$\boldsymbol{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, our goal is to find the following: $R_{error} = \min_{\hat{\mathbf{X}}} \|\boldsymbol{X} - \hat{\mathbf{X}}\|_{\mathbf{F}}$, where $\hat{\mathbf{X}}$ is generated in the same way in Eq. 5 and $\|\boldsymbol{X}\|$ is calculated as root-mean-square as follow:

$$\|\boldsymbol{X}\| = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \sum_{i_3=1}^{I_3} X_{i_1 i_2 i_3}^2}. \tag{7}$$

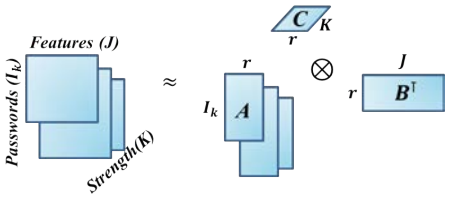


Figure 2: PARAFAC2 decomposition with password dataset, where the dim$(I[K] \times J \times K) = (I[3] \times 31 \times 3)$. $I[1] \times J$, $I[2] \times J$, and $I[3] \times J$ are the password-to-feature matrix in weak, medium, and strong password group.

In addition, we use the alternating least squares (ALS) algorithm to find an optimal value, and we must set the condition to stop the iteration for finding optimal solution. Otherwise, there is a possibility of falling into an infinite loop. We set 500 as the maximum iteration and 10^{-4} as the error tolerance, which is the gap of residuals between the present and previous iteration. Since the optimal r makes the reconstruction error the smallest, we trace Frobenius norm residuals and observe the percentage of variance to find an optimum.

A factor r and the residuals are mostly in inversely proportional to each other. It means that we can have a smaller residual as r increases. However, unfortunately, the ALS algorithm can perform poorly, if the number of factors r is too large. Therefore, we should choose a small r. And the Elbow method is one of the popular methods to achieve this goal, where the value at the sharp bending part of norm residual graph can be the optimal r. We present our optimum r in the following section.

6 RESULTS

Optimal Rank (r): We analyzed at the variance of the reconstruction ratio (R^2_{ratio}=$||\hat{\boldsymbol{X}}||^2/||\boldsymbol{X}||^2$) of the estimated tensor $||\hat{\boldsymbol{X}}||^2$ against the original $||\boldsymbol{X}||^2$ by changing the number of r from 2 to 10, as shown in Fig. 3. The Y-axis in Fig. 3 shows the percentage of variance between R_{error} as a red line and R^2_{ratio} values as a blue line, where R^2_{ratio} has an inverse relation from R_{error}. The sharp bending point shown in a dotted red circle is an elbow, which is the optimal r = 2. When r = 2, we further analyzed the reconstruction error R_{error} and found that R_{error} is less than 1% and the dropping rate is 73.80% showing drastic decrease compared to other neighboring points. The average dropping rate of other points between 3 and 10 were 18.48%. However, PARAFAC2 does not guarantee the uniqueness of decomposition solution. Hence, instead of relaxing the factor invariance constraint across the parallel set of matrices, we calculated the average R_{error} and R^2_{ratio} of each r by repeating 10 times.

With the factor value r = 2, our goal is to discover specifically which features have an dominating effect on the password memorability and strength from component matrices **B** and **C**. When we decomposed matrices, we applied the non-negativity constraints

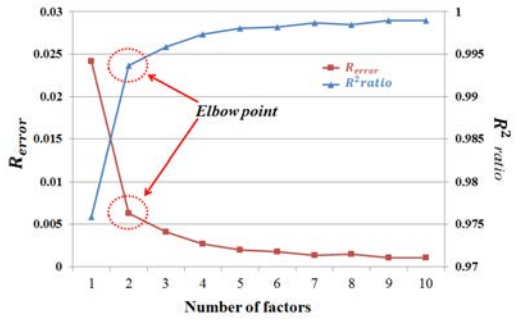


Figure 3: R_{error} (red line with the left Y-axis) and R^2_{ratio} by the number of factor (blue line with the right Y-axis). Each point is calculated as the average of 10 experiments. Red dotted circle indicates the elbow point.

to matrices **B** and **C**. Since non-negativity constraints not only mitigate the uniqueness problem but also provide positive values to component matrix after decomposition, it is more interpretable and practically used. Even though we mitigate the uniqueness issue by imposing non-negativity constraints, it does not eliminate the issue completely. Hence, we repeated the experiments 10 times and verified that all of them showed consistent patterns. Therefore, all results in this section are obtained from the one out of 10 representative experimental results.

Result 1. Syntactic + Semantic Features: We obtained the component matrices **B** and **C** from PARAFAC2 decomposition. To compute the decomposition, we use the alternating least square (ALS) method to select an optimal rank *r*. Specifically, a component matrix **C** captures the factors that directly influence password strength, as shown in Fig. 2. The detailed results of **C** are presented in Table 2, where passwords in *weak* strength group is affected by *factor1* with 0.7427 and *factor2* with 1.7622. On the other hand, passwords in *strong* strength group is impacted by *factor1* with 1.1229 and *factor2* with 2.8128. In addition, a component matrix **B** captures the relationship between two factors and each password feature. The detailed results of matrix **B** are presented in Table. 3. Note that, in tensor decomposition, it is not important to understand and interpret the actual value of factors. However, the relative effect and relationships among features are more important as shown in other research [2].

Table 2: A component matrix C, which reflects the relationship between two factors and strength group.

	Weak	Medium	Strong
factor 1	0.7427	10.9210	1.1229
factor 2	1.7622	3.5415	2.8128

Table 3: A component matrix B, which reflects the relationship between two factors and each feature.

	length	D	S	UL	3class8
factor 1	1.1825	1.2220	1.7247	1.0810	1.0816
factor 2	4.5474	1.3056	0.8986	0.5292	0.5715
	flipLS	**flipLD**	**flipDS**	**flipDL**	**flipSL**
factor 1	0.8386	0.7471	0.8063	0.6525	1.8137
factor 2	0.4487	0.5724	0.3768	0.3674	0.7957
	flipSD	**tFlip**	**article**	**number4**	**np1**
factor 1	0.5066	0.9256	1.0885	1.4400	0.2527
factor 2	0.2423	1.0480	0.7758	0.6400	0.2567
	char1	**special1**	**nn1**	**number3**	**number1**
factor 1	0.8178	1.2084	1.5271	0.7259	1.0695
factor 2	0.4551	0.6389	0.7424	0.3235	0.5625
	number2	**special2**	**verb**	**adjective**	**nn2**
factor 1	0.8536	0.3212	0.4912	0.1142	0.6428
factor 2	0.4225	0.1440	0.2336	0.0784	0.2797
	pronoun	**nn**	**char2**	**char3**	**num**
factor 1	0.0774	1.5661	0.6900	0.9926	1.0850
factor 2	0.0638	0.8208	0.3338	0.4134	0.7775
	np				
factor 1	0.0678				
factor 2	0.1956				

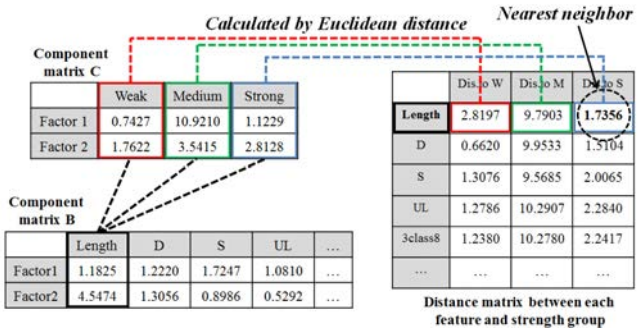


Figure 4: Illustration of finding the nearest neighbor of each feature from the component matrices B and C

Next, we need to identify features that have strong influence to each password strength group. We used the Euclidean distance (ED) algorithm to find the distance from each feature (i.e. length, D) to three password strength groups, where ED is computed as follows:

$$ED_{str} = \sqrt{(B^{str}_{f1} - C^{ft}_{f1})^2 + (B^{str}_{fr2} - C^{ft}_{f2})^2}, \quad (8)$$

where $str\ is\ strength \in \{weak, medium, strong\}$, $f\ is\ factor$, and $ft\ is\ feature$. B^{str}_{f1} is the $factor1$ value in Table 3 and C^{ft}_{f1} is the $factor1$ value from each feature in Table 2. The actual example of computing *ED* for length is illustrated in Fig. 4.

Each row in Table 4 is the Euclidean distance calculated using Eq. 8 and gray colored value in each row is the nearest neighbor (min.) distance. The smaller distance means the stronger impact to each password. As shown in Table. 4, only *length* is consistently computed as *strong*. This means that password length is the consistent and dominating feature that is related to strong passwords. In other words, a longer password triumphs over a more complex but shorter password, where similar results are found in other research [36].

Table 4: Distance matrix between 31 features and password strength. Length is the dominant contributor in making stronger passwords.

	Dist. to Weak	Dist. to Medium	Dist. to Strong
length	2.8197	9.7903	1.7356
D	0.6620	9.9533	1.5104
S	1.3076	9.5685	2.0065
UL	1.2786	10.2907	2.2840
3class8	1.2380	10.2780	2.2417
filpLS	1.3170	10.5461	2.3811
filpLD	1.1898	10.5982	2.2717
filpDS	1.3868	10.5982	2.4564
filpDL	1.3977	10.7479	2.4903
flipSL	1.4426	9.5121	2.1321
flipSD	1.5381	10.9244	2.6434
tFlip	0.7373	10.3017	1.7758
article	1.0453	10.2141	2.0373
number4	1.3212	9.9150	2.1958
np1	1.5832	11.1626	2.7002
char1	1.3092	10.5641	2.3774
special1	1.2160	10.1370	2.1756
nn1	1.2866	9.8021	2.1095
number3	1.4388	10.6909	2.5208
number1	1.2434	10.2921	2.2509
number2	1.3442	10.5394	2.4054
special2	1.6721	11.1310	2.7866
verb	1.5492	10.9418	2.6555
adjective	1.7972	11.3481	2.9145
nn2	1.4859	10.7834	2.5782
pronoun	1.8241	11.3876	2.9411
nn	1.2507	9.7425	2.0407
char2	1.4294	10.7220	2.5165
char3	1.3717	10.4094	2.4029
num	1.0425	10.2169	2.0356
np	1.7057	11.3572	2.8218

Result 2. Semantic features only: We found that the dominant contributor to strength is *length*. However, we still need to uncover what other semantic features can influence on both memorable and strong password creations. As we strongly hypothesize that semantic aspects would contribute in memorability, we conducted an additional experiment to examine semantic features other than length and syntactic features. In this experiment, we excluded syntactic features and only used 19 semantic features for a tensor decomposition.

Among 19 semantic features, *char1* was classified as *strong* most of the times. And *number4*, *nn1* (singular common noun, e.g. book), and *number2* (two consecutive digits: e.g. 37) was classified as *strong* 8 times. These four features can be considered as dominant contributors to generating strong passwords. Whereas *verb*, *adjective*, *nn2* (plural common noun, e.g. girls), *np* (proper noun and neutral for number, e.g. IBM), and *num* (every number, not digit, e.g. three) never appeared for *strong* strength group. Also, *np1* (singular proper noun, e.g. London) and *pronoun* were classified as *strong* only once.

From 19 semantic features, 10 features were based on dictionary word and 9 features non-dictionary word such as *char1*. Only one dictionary word feature (*nn1*) was shown to influence to strong password, as shown in Table. 5. It indicates dictionary words are rarely observed in strong passwords. It indirectly captures that users are making an effort to use non dictionary words to create strong passwords. If dictionary based words occur, they tended to be mostly *nn1* in strong passwords, which can be a crucial factor to improve the memorability of their passwords.

Table 5: Distance matrix between semantic features and password strength: dominant contributors are *number3, char1, nn1,* and *number2*. Most dictionary words features are not close to *strong* strength group. Only *nn1* is classified as *strong* among all dictionary word features.

	Dist. to Weak	Dist. to Medium	Dist. to Strong
article	2.6494	1.5463	2.4830
number4	0.2515	0.8782	0.1755
np1	1.4964	0.9686	1.4219
char1	0.4515	0.6565	0.2878
special1	1.3039	0.2056	1.1374
nn1	0.8761	1.0084	0.7335
number3	0.0634	1.1711	0.2501
number1	0.7042	0.4169	0.5257
number2	0.3244	0.7847	0.1781
special2	0.3199	1.4258	0. 4919
verb	0.2562	1.3524	0.4155
adjective	0.2632	1.3565	0.4193
nn2	0.3561	1.4520	0.5147
pronoun	0.2326	1.3387	0.4060
nn	1.1882	0.9352	1.0200
char2	0.1801	1.2826	0.3484
char3	0.3326	1.4390	0.5054
num	2.7266	1.6237	2.5604
np	1.6317	1.0366	1.5519

7 DISCUSSION AND LIMITATION

In summary, the purpose of our research is to identify the features to make a password strong and memorable. Using syntactic and semantic features, we analyze password dataset and provide the insight to creating memorable and strong password. With 2-dim. password dataset, we constructed a 3rd-order tensor consisting of $(passwords \times features \times strength) = ((109, 2276, 875) \times 31 \times 3)$ and decomposed them using PARAFAC2. We discovered that password length is the dominant contributor in making a password stronger, compared to all other features. Among semantic features, most of them did not influence a strong password, except common nouns. Hence, we can conclude that creating a long password is the most powerful and practical way to strengthen memorable passwords. Although PARAFAC2 tensor decomposition provides a high degree of flexibility, it suffers from the uniqueness issue. Future research is needed in this area to further improve the performance of PARAFAC2. Also, we used more than 3,200 memorable passwords but more dataset would be helpful in more accurately identifying features. We also plan to perform our analysis with additional password datasets to verify our findings.

8 CONCLUSION

A tensor decomposition is the powerful method to quantitatively capture the importance and relationships among of underlying features in a dataset, providing the higher level of explainability. In this work, we applied the tensor decomposition for password data analysis and validate the features that strongly influence the creation of memorable and strong passwords. Our work complements the existing qualitative research and confirm the importance of creating a longer password using non dictionary words. Our work can be applied for better understanding the password structures for an unlabelled password dataset. Further, it can be integrated in generating password suggestions and advice for users to create memorable and strong passwords.

9 ACKNOWLEDGMENTS

We thank anonymous reviewers for their helpful feedback on drafts of this paper. This research was supported by the KEIT (Korea Institute for Industrial Economics & Trade), Korea, under the "Global Advanced Technology Center" (10053204), and by the MSIT (Ministry of Science and ICT), Korea, under the ICT Consilience Creative Program (IITP-2017-R0346-16-1007) supervised by the IITP (Institute for Information & Communications Technology Promotion), and by NRF of Korea by the MSIT (NRF-2017R1C1B5076474).

REFERENCES

[1] J. B. Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and Its Applications* 18, 2 (J. B. Kruskal), 95–138. https://doi.org/10.1016/0024-3795(77)90069-6

[2] Evrim Acar, Seyit A Çamtepe, Mukkai S Krishnamoorthy, and Bülent Yener. 2005. Modeling and multiway analysis of chatroom tensors. In *International Conference on Intelligence and Security Informatics*. Springer, 256–268.

[3] Evrim Acar, Seyit A Camtepe, and Bülent Yener. 2006. Collective sampling and analysis of high order tensors for chatroom communications. In *International Conference on Intelligence and Security Informatics*. Springer, 213–224.

[4] Evrim Acar and Bülent Yener. 2009. Unsupervised multiway data analysis: A literature survey. *IEEE transactions on knowledge and data engineering* 21, 1 (2009), 6–20.

[5] Charlotte Møller Andersen and R Bro. 2003. Practical aspects of PARAFAC modeling of fluorescence excitation-emission data. *Journal of Chemometrics: A Journal of the Chemometrics Society* 17, 4 (2003), 200–215.

[6] Carl J Appellof and Ernest R Davidson. 1981. Strategies for analyzing data from video fluorometric monitoring of liquid chromatographic effluents. *Analytical Chemistry* 53, 13 (1981), 2053–2056.

[7] Joseph Bonneau. 2012. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *Security and Privacy (SP), 2012 IEEE Symposium on*. IEEE, 538–552.

[8] J Douglas Carroll and Jih-Jie Chang. 1970. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika* 35, 3 (1970), 283–319.

[9] Claude Castelluccia, Markus Dürmuth, and Daniele Perito. 2012. Adaptive Password-Strength Meters from Markov Models.. In *NDSS*.

[10] Hsien-Cheng Chou, Hung-Chang Lee, Hwan-Jeu Yu, Fei-Pei Lai, Kuo-Hsuan Huang, Chih-Wen Hsueh, et al. 2013. Password cracking based on learned patterns from disclosed passwords. *IJICIC* 9, 2 (2013), 821–839.

[11] Andrzej Cichocki, Danilo Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar Caiafa, and Huy Anh Phan. 2015. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine* 32, 2 (2015), 145–163.

[12] Lieven De Lathauwer and Joos Vandewalle. 2004. Dimensionality reduction in higher-order signal processing and rank-(R1, R2,..., RN) reduction in multilinear algebra. *Linear Algebra Appl.* 391 (2004), 31–55.

[13] Matteo Dell'Amico and Maurizio Filippone. 2015. Monte Carlo Strength Evaluation: Fast and Reliable Password Checking. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 158–169.

[14] Dinei Florêncio, Cormac Herley, and Paul C Van Oorschot. 2016. Pushing on string: The'don't care'region of password strength. *Commun. ACM* 59, 11 (2016), 66–74.

[15] Wolfgang Hackbusch. 2012. *Tensor spaces and numerical tensor calculus*. Vol. 42. Springer Science & Business Media.

[16] Richard A Harshman. 1970. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis. (1970).

[17] Frank L Hitchcock. 1927. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics* 6, 1-4 (1927), 164–189.

[18] Frank L Hitchcock. 1928. Multiple invariants and generalized rank of a p-way matrix or tensor. *Journal of Mathematics and Physics* 7, 1-4 (1928), 39–79.

[19] Markus Jakobsson and Mayank Dhiman. 2013. The benefits of understanding passwords. In *Mobile Authentication*. Springer, 5–24.

[20] Henk AL Kiers. 2000. Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics: A Journal of the Chemometrics Society* 14, 3 (2000), 105–122.

[21] Tamara G Kolda and Brett W Bader. 2009. Tensor decompositions and applications. *SIAM review* 51, 3 (2009), 455–500.

[22] J. B. Kruskal. 1989. Multiway Data Analysis. North-Holland Publishing Co., Amsterdam, The Netherlands, The Netherlands, Chapter Rank, Decomposition, and Uniqueness for 3-way and N-way Arrays, 7–18. http://dl.acm.org/citation.cfm?id=120565.120567

[23] Arvind Narayanan and Vitaly Shmatikov. 2005. Fast dictionary attacks on passwords using time-space tradeoff. In *Proceedings of the 12th ACM conference on Computer and communications security*. ACM, 364–372.

[24] Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. 2017. Tensors for data mining and data fusion: Models, applications, and scalable algorithms. *ACM Transactions on Intelligent Systems and Technology (TIST)* 8, 2 (2017), 16.

[25] Ashwini Rao, Birendra Jha, and Gananand Kini. 2013. Effect of grammar on security of long passwords. In *Proceedings of the third ACM conference on Data and application security and privacy*. ACM, 317–324.

[26] Paul Rayson, Dawn Archer, Scott Piao, and Anthony M McEnery. 2004. The UCREL semantic analysis system. (2004).

[27] Ledyard R Tucker. 1963. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change* 15 (1963), 122–137.

[28] Ledyard R Tucker. 1964. The extension of factor analysis to three-dimensional matrices. *Contributions to mathematical psychology* 110119 (1964).

[29] Ledyard R Tucker. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika* 31, 3 (1966), 279–311.

[30] Blase Ur, Fumiko Noma, Jonathan Bees, Sean M Segreti, Richard Shay, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2015. "I added '!' at the end to make it secure": Observing password creation in the lab. In *Proc. SOUPS*.

[31] M Alex O Vasilescu and Demetri Terzopoulos. 2002. Multilinear analysis of image ensembles: Tensorfaces. In *European Conference on Computer Vision*. Springer, 447–460.

[32] M Alex O Vasilescu and Demetri Terzopoulos. 2002. Multilinear image analysis for facial recognition. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, Vol. 2. IEEE, 511–514.

[33] Rafael Veras, Christopher Collins, and Julie Thorpe. 2014. On the semantic patterns of passwords and their security impact. In *Network and Distributed System Security Symposium (NDSS'14)*.

[34] Matt Weir, Sudhir Aggarwal, Breno De Medeiros, and Bill Glodek. 2009. Password cracking using probabilistic context-free grammars. In *Security and Privacy, 2009 30th IEEE Symposium on*. IEEE, 391–405.

[35] Simon Woo, Elsi Kaiser, Ron Artstein, and Jelena Mirkovic. 2016. Life-experience passwords (leps). In *Proceedings of the 32nd Annual Conference on Computer Security Applications*. ACM, 113–126.

[36] Simon S Woo and Jelena Mirkovic. 2018. GuidedPass: Helping Users to Create Strong and Memorable Passwords. In *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 250–270.

Understanding Reader Backtracking Behavior in Online News Articles

Uzi Smadja
Technion - Israel Institute of Technology
Haifa, Israel
uzi.smadja@gmail.com

Max Grusky
Cornell Tech, Cornell University
New York, NY, USA
grusky@cs.cornell.edu

Yoav Artzi
Cornell Tech, Cornell University
New York, NY, USA
yoav@cs.cornell.edu

Mor Naaman
Cornell Tech, Cornell University
New York, NY, USA
mor@jacobs.cornell.edu

ABSTRACT

Rich engagement data can shed light on how people interact with online content and how such interactions may be determined by the content of the page. In this work, we investigate a specific type of interaction, *backtracking*, which refers to the action of scrolling back in a browser while reading an online news article. We leverage a dataset of close to 700K instances of more than 15K readers interacting with online news articles, in order to characterize and predict backtracking behavior. We first define different types of backtracking actions. We then show that "full" backtracks, where the readers eventually return to the spot at which they left the text, can be predicted by using features that were previously shown to relate to text readability. This finding highlights the relationship between backtracking and readability and suggests that backtracking could help assess readability of content at scale.

KEYWORDS

Backtracking, User Engagement, Readability, Online News

ACM Reference Format:
Uzi Smadja, Max Grusky, Yoav Artzi, and Mor Naaman. 2019. Understanding Reader Backtracking Behavior in Online News Articles. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313571

1 INTRODUCTION

Monitoring the interaction of users with Web pages at a large scale has contributed to a better understanding of reading behaviors [19] and to connecting patterns of readers' engagement to the textual content of articles [7]. At the same time, textual features have been used to model important aspects of writing, such as how readable [5] or engaging [21] a specific document may be. The latter, however, does not take into account the empirical behavior of readers, nor does it take into account specific attributes of the publication audience.

In this work, we use page instrumentation data[1] collected during reading sessions of online news articles in order to measure and study a reading behavior we refer to as *backtracking*. This under-explored signal of engagement occurs when readers scroll back or upward (see Figure 1) within an online article. A reader may backtrack, for instance, to re-read a previous section of the text. This definition of backtracking diverges from how backtracking was defined in lab studies of readability and text comprehension. For example, studies of *saccades* [31] — short, rapid eye movements between two close fixation points — refer to backtracking as regressive saccades, when the reader goes back a few words in the text. In contrast, our notion of online backtracking, while not as fine grained, presents multiple advantages. Such signals can be collected at scale by simple instrumentation of a Web page, allowing for regular observation of reading behavior in naturalistic settings over long periods of time.

Studying fine-grained signals of engagement such as backtracking at a large scale has the potential to greatly inform our understanding of reading and text. Recent work has examined similar signals to understand reading behavior in relation to textual content. In particular, Lagun and Lalmas [19] use within-page measurements of engagement, such as dwell time, to categorize reading sessions depending on the time spent in different areas of the page. Grusky et al. [8] use similar data to show that reading speeds (i.e., words-per-minute reading rates) are consistent with the ones observed in lab studies. Finally, recent work has begun to explore the modeling of user behavior using content-based features. Grinberg [7] has used accumulated information gain in the article to predict the farthest point read by users on the page.

In this work, we focus on backtracking behavior and its potential to reflect the qualities of the text. To this effect, we examine 1.4M sessions in which about 25K users read and engage with 8,000 online news articles across two major online publications. Because readers can backtrack for a variety of reasons, we define, explore, and measure the prevalence of three different forms of backtracking within articles. To better understand the connection between backtracking and article text, we develop text-based models that predict which articles experience high and low frequency of backtracking actions. We find that the signal we characterize as *full backtracking*– where a reader, after backtracking, eventually returns to the spot

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313571

[1]The anonymized instrumentation data was graciously provided by Chartbeat for research purposes as detailed in Section 3.

at which they stopped reading–is related to properties of the text such as coreference and lexical features. Our results point towards the potential of backtracking as a new audience-driven approach to predict text comprehension at scale.

2 RELATED WORK

User engagement has been studied in depth since the early days of the Web. Signals have been gathered at page level (e.g. click-through rate [16]) or within pages, and in different settings, either transparently via instrumented pages or in controlled user studies. Such signals have then been leveraged for multiple tasks, such as understanding interaction with news articles [19], search satisfaction in browsers via mouse movements [14], or in mobile devices via touch actions [10]. Within the large body of work on this topic, we focus on work pertaining to reading behavior observed via eye-tracking devices, to readability, and to measuring the relationship between engagement and content.

Eye-tracking devices have been used in lab studies in order to examine readers' behavior. Most related to our work is the study of *saccades*, which are defined as a rapid movement of the eye between fixation points [31]. Saccades occur most often when scanning a text or trying to spot a specific piece of information while reading. Regressive saccades consist of going back by a few words, or a line at most, and represent about 10-15% of saccade events. They have also been referred to as backtracking, but differ from our definition of online backtracking as they operate at a much smaller scale. Eye tracking has also been used to explore the processing complexity of text when reading [17]. Other studies [22–24, 32] have incorporated eye-tracking and psychology in qualitative experiments and showed that good readers backtrack efficiently, while poor readers sometimes struggle to find the location that caused the difficulty, under the conjecture that comprehension failures are the cause of the regressive fixation. In contrast, our backtracking signal uses scrolling as a proxy for eye movement on the news article and is less granular, but requires no equipment and can be observed at scale.

Reading behavior has been studied extensively through the analysis and prediction of text readability. Traditional readability measures are typically derived from simple ratios of text properties, such as the average number of words per sentence, or the frequency of difficult words [3]. We take advantage of commonly used reading metrics, such as the Automated Readability Index (ARI) [34], which is defined as a function of the average character-per-word and word-per-sentence ratios. Other readability measures that we explore count the number of syllables per word [6, 26] or compute a ratio of multi-syllable words and the number of words and sentences [9, 18] to assess reading ease or grade level difficulty. Other reading indices include the Dale-Chall formula [4] that leverages a predefined list of "easy" words, with every other word being considered "difficult," or the nonlinear SMOG index [25], which uses syllable counting in order to identify challenging polysyllable words. Readability studies have also explored more complex language models [1, 29] considering grammatical structures [12, 13] and methods such as entity-grid representation [30, 33]. Feng et al. [5] have compared between different features for readability assessment, including coreferences, lexical and discourse features. These readability studies focus on assessing readability by grade level, using for instance manually-tagged datasets for grades 2-5 [28], or are targeted to non-native speakers, who are trying to learn a new language [29]. This makes these approaches not applicable to our domain of online news articles, for which most readers are past grade-level reading and are to the most part native speakers. For such an analysis, we lack large-scale manually tagged corpora, thus adopting the backtracking signal instead.

Similar to our work, which does not rely on manually-tagged corpora, other work has taken advantage of the textual content of Web pages to understand user behavior. In particular, Grusky et al. [8] and Lagun and Lalmas [19] have examined ways of measuring, validating, and modeling how users engage with online news articles. Similarly, Grinberg [7] has looked at semantic information gain and the development of ideas within the body of articles along with their topics in order to predict different types of engagement on news articles. We extend this existing line of work by focusing on a new signal of engagement, backtracking, which captures different aspects of reading behavior.

3 UNDERSTANDING BACKTRACKING

We first define backtracking events and identify the different types of backtracking seen in the data. We also describe backtracking patterns at page level, user level, and session level.

Backtracking takes place when readers break their sequential reading of an online document and scroll up, returning to a previous point in the text. One possible interpretation of such an event in the context of online news is that the user interrupted their sequential reading in order to re-read portions of the document, for example to find earlier references, to names, or to get additional clarity on content that may impact their understanding of the rest of the article [22–24]. Backtracking events may also involve other tasks the reader may perform on the page, such as, navigation or response to ad placement [36]. We explore several definitions that capture different aspects of backtracking.

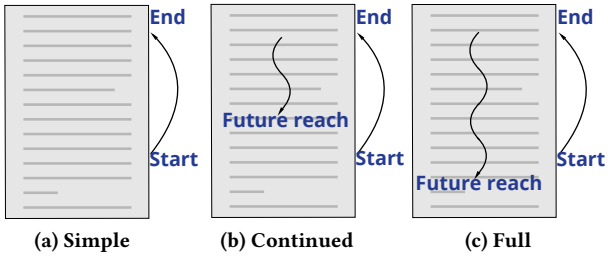


Figure 1: Different types of backtracking on Web pages

Defining backtracking (BT) events. A backtracking event occurs during a page-read event and is represented by an origin and target point defined as the tuple:

$$b = (start, end, reach)$$

Figure 1 presents the reading pattern for three different types of backtracking events. The first two values, *start* and *end*, represent the actual backtracking action and indicate a location in the article (hence, $start > end$). We also define *reach* as the furthest location

Table 1: P_{20} dataset size and number of backtracking events

	Users	Pages	Sessions	Simple BT	Continued BT	Full BT
LONG	15,949	3,968	694,728	57,897	28,840	246,787
SHRT	11,117	3,957	763,335	136,153	49,211	127,421

reached by the user following the current backtracking event. Capturing the *reach* value for each backtracking action helps interpret the context of the action by noting whether the user leaves the page, keeps scrolling or eventually returns to the *start* point. We define three types of backtracking events:

- Simple backtrack: Reader r scrolls back on page p but does not keep scrolling afterwards. Based on our previous definition, for the tuple $(start, end, reach)$, $reach \leq end$. This is the most basic backtracking signal.
- Continued backtrack: Reader r scrolls back, and proceeds to scroll forward afterwards but does not return to the *start* location ($start > reach > end$).
- Full backtrack: Reader r scrolls back, and eventually returns all the way to the backtrack starting point ($reach \geq start$).

In order to better capture intentional backtracks and meaningful movements of readers, we only consider backtracking of at least 100 pixels in range (consisting of about a dozen lines of text), where the user spends at least 15 seconds on the page after completing the backtrack.

Dataset description. Our dataset was provided by *Chartbeat* and includes data from two popular news websites from 2015. The first website is a popular news and entertainment magazine with long-form content, which we will refer to as LONG. The second, a popular short-format news website, will be referred to as SHRT. The raw data consists of "pings," which are defined as a record of viewport information [8] associated with a single user reading session on a single page. Pings are collected every 15 seconds, or at longer time intervals if a user is inactive. Every ping contains information about the visited page in the browser by a specific user, including height, width, length of the news article and the current position of the user on the page, all measured in pixels. We keep only data pertaining to English articles and originating from desktop (non-mobile) sessions. We consider only sessions that are longer than 30 seconds, as they are more likely to be indicative of actual reading behavior.

We refer to the session where a single reader r is reading a single page p as a *read event*. For most of the analysis below, we are using a subset of the dataset with pages p that have at least 20 reading sessions which we refer to as P_{20}. We chose to use P_{20} in order to ensure we have sufficient signal of behavior for each page, and reduce the effect of outliers (We perform sensitivity analysis in Section 4 to show how this choice effects performance in a prediction task.) The number of users, pages sessions and backtracking events performed in P_{20} are shown in Table 1.

Table 1 presents statistics on the different types of backtracking for pages that have passed the initial filtering P_{20}. We see that for roughly the same number of pages and reading sessions, LONG has almost twice as many full backtracking events, but less than half the number of simple backtracking events. There are two key factors that may explain this divergence. First, on the LONG site, pages are typically longer (5,000 pixels on average) than on SHRT (3,000 pixels on average). The longer the page, the more chances it will experience a backtracking action. In addition, we observed that on SHRT, a large proportion of simple backtracks end at the top of the page. This is likely a result of a static the navigation bar at the top of the page on SHRT website, causing the user to return to the top of the page in order to navigate to another area in the site.

As shown in Table 1, the P_{20} the dataset contains 1.4M reading sessions, but far fewer backtracking events. The majority of reading sessions have no backtracking event: 78% of the sessions in the LONG dataset, and 88% of the sessions on SHRT do not include any full backtracking event.

Backtracking patterns. Figure 2 shows the typical sizes of backtracking actions using a cumulative distribution function (CDF) of sizes for the three types of backtracking, measured in pixels. Given backtracking event $b \in B$, where B is the set of all backtracking events, let $\text{size}(b) = \text{start}(b) - \text{end}(b)$. Given any size x (x-axis), the CDF is $F(x) = P(\text{size}(b) \leq x)$ for all $b \in B$ (y-axis). As the figure shows, even though the pages on LONG are significantly longer than on SHRT, the typical size of full backtracking is smaller on LONG. For example, about 80% of the full backtracks (top curve in each figure) are under 550 pixels in size in LONG, while the same is true for only 75% of the backtracks on SHRT (top curve on both figures).

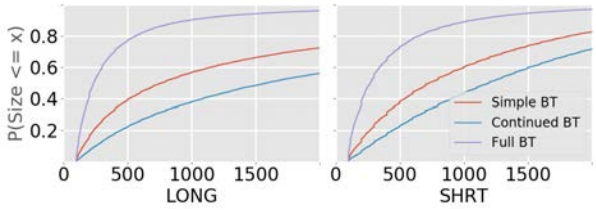


Figure 2: Backtracking sizes (in pixels) CDF

When readers backtrack, do they backtrack more than once? Figure 3 shows the number of backtrack events for sessions where there was at least one backtrack, for each type of backtracking event. For example, the right-most pane shows that 15% of LONG (red) sessions have exactly one full backtracking event. The rest of the sessions, that account for about 7% of the total number of sessions, contain two or more full backtracking events.

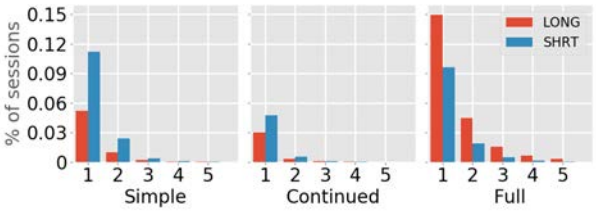


Figure 3: Number of backtracking events occurring across all sessions

We expect some pages to have more backtracking events than others, and we indeed verified that the variance of pages is much

greater than would have been be expected if the backtracking was a random event, e.g., modeled as a Bernoulli variable. We consider backtracking behavior to be close in nature to a Bernoulli, as we can see that backtracking more than once is rare and as such it is better to ask whether or not any backtracking event occurred. Figure 4 shows the page-level backtracking distribution for articles in P_{20}. The x-axis shows the backtracking-per-session average for each page, which is the number of backtracking events divided by number of reading sessions per page. The y-axis shows the number of pages with this average in each bin. For example, the right-most pane shows that, on LONG, very few pages have an average of more than 0.6 backtracking events per session. We see that for full backtracking, there is a significant difference in both variance and mean of the backtracking event distributions of the two domains.

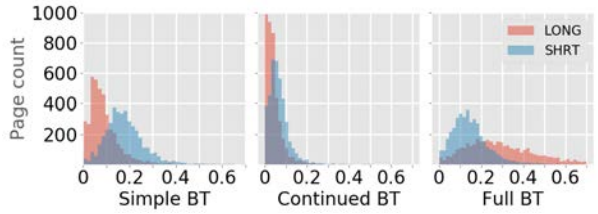


Figure 4: Are some pages more prone to backtracking? Pages' mean number of backtracking events per reading session

Are readers different in their tendency to perform backtracking? To answer this question we examine a slightly different subset of the data, U_{20}, comprised of reading sessions by users that have at least 20 reading sessions. Figure 5 shows the histogram of users' mean number of backtracks per session. Again, we see that the observed variance is much greater than if users backtracked at random, as a Bernoulli variable. The x-axis represents the average number of backtracking events per session, and the y-axis is the number of users within that bin. For example, half of LONG's readers (red) perform a full backtrack (right pane) once every three reading session on average. Compared to the page data (Figure 4), both the simple and full backtracking figures are more right-skewed, for both domains. For example, for the SHRT users (blue), there are many users that rarely perform any full backtracking.

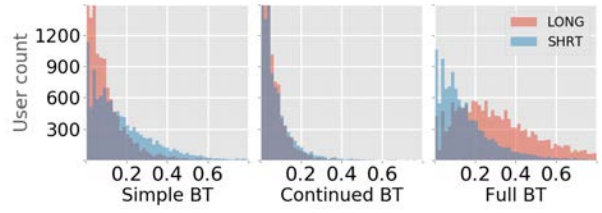


Figure 5: Do some users backtrack more frequently? Users' mean number of backtracking events per reading session

4 PREDICTING BACKTRACKING

How well can backtracking events be predicted, and what is the contributions of different features in predicting backtracking events? In this section we explore this question, also touching on the difference between types of backtracking events. The descriptive analysis suggests that LONG pages generate higher numbers of full backtracking events, which are more likely to relate to readability (as we also show below). Considering this finding, the analysis in this section is using only the LONG articles (line 1 in Table 1).

We define a binary classification task where we predict whether a page is likely to have a high or low number of backtracks per reading session. Based on the hypothesis that backtracking occurs when readers re-read a piece of text that they did not fully understand, we suggest that backtracking is an indicator of readability issues. We therefore examine several readability-related measures for texts as part of our prediction task, using a set of features inspired by previous work on readability by Feng et al. [5]. We consider four families of features at the article level:

Lexical features. These features have been shown to have strong predictive power in readability tasks, and can capture many different aspects of the text. The features include several established readability measures mentioned earlier, like the Dale-Chall score, and some simple text properties, such as average sentence length and the number of syllables per word. These features are computationally cheap to generate and do not require deep computational analysis.

Entity-density features. Entity-density features have been shown to be useful in readability prediction [5] mostly due to the fact that new concepts are often introduced in a narrative by entities (e.g. names, locations) and might impose additional burden on readers' working memory. In order to collect entities, we used spaCy[2] [15].

Part-of-speech (POS) density features. POS-based grammatical features are widely used for various tasks, including readability. In particular, [5] showed that noun-based POS features generate the highest classification accuracy in predicting correct readability grade level. In order to generate POS features, we use the spaCy POS tagger [15].

Coreference chain features. Coreference features were first studied in [5] for readability modeling, though they did not perform as well as two previous families of features in the earlier work. Our conjecture is that these features may help in representing the cognitive load on the user and should directly affect backtracking. We extract these features using the *End-to-end Neural Coreference Resolution* library[3] based on work by Lee et al. [20].

Table 2 provides more details about the features used for each family.

4.1 Defining the backtracking prediction task

Since our goal is to predict whether or not an article will have a high or low number of backtracking per reading session, we divide the training data into three equally sized sections, representing *high backtracking*, *medium backtracking*, and *low backtracking* based on the distribution shown in Figure 4. By excluding the third of the pages with *medium backtracking* and focusing on the equally sized *high backtracking* and *low backtracking* pages, we end up with

[2]https://spacy.io/
[3]https://github.com/kentonl/e2e-coref

Table 2: List of features computed for every article.

	Lexical Features
Dale-Chall score	Readability formula based on frequency of difficult words
ARI	Readability formula based on the relative length of words and sentences
Avg. syllables	Average syllables per word
SMOG grade	Readability formula based on the number of polysyllables (3 or more syllables)
Flesch-Kincaid score	Readability formula based on the ratio between sentences, words and syllables
Images	Number of images in article
Words	Number of words in article
Sentences	Number of sentences in article
	Entity-density Features
Entities	Number of named entities in article
Unique entities	Number of unique entities in article
	Coreference Features
Coreference chains	Number of coreference chains in article
Avg. coreference chain span	Average distance (in words) between words in the same chain
Avg. coreference chain size	Average number of words per chain
	POS Density Features (Nouns)
Percentage of nouns	Percentage of nouns in article (including entities) out of all tokens
Percentage of unique nouns	Percentage of unique nouns in article (including entities) out of all tokens.

balanced classes and a binary class label $y \in \{0, 1\}$, for two thirds of our training examples. We perform the same procedure for the development and test sets. We tested in preliminary experiments a set of classifiers such as SVM, logistic regression, and decision trees, we found that random forests performed best across all sets of features. Therefore, we use random forests for our analysis.

4.2 Prediction based on textual features

We examine how well the prediction performs with the full set of features, and assess the importance of the different types of features. From the P_{20} dataset, we keep only articles from LONG and split the samples into training, development and held out test sets with $75\% - 10\% - 15\%$ proportions. After filtering the *medium backtracking* pages, this dataset consists of 1,991-256-404 pages and 324,708-38,927-71,364 user sessions for training, development, and test sets. The allocation of pages to training, development and test set was determined by a random hash of the page URL. With a similar setup, but on a different subject, we perform an analysis akin to [2], including feature ablation. As mentioned before, the data for the prediction task is balanced and thus the expectation for a random prediction model is 50%. However, if we consider using the length (in pixels) of the page as the only prediction feature, the baseline to surpass is roughly 75%, in terms of accuracy.

The results, achieved on the held-out test set (and thus only computed after the work was complete) are shown in Table 3. The best performance was achieved using a random forest model[4] for full backtracking events, with an accuracy of 83.8%. We performed feature ablation by choosing a subset of features for the prediction task based on Table 2. For the feature ablation procedure, we associate each feature type with the length of the page, our control variable. We use length because, as mentioned above, longer pages provide more opportunity for backtracking, and length alone been shown to have significant predictive power for backtracking.

[4]hyper-parameters: maximum depth:10, minimum sample split:4

Table 3: Feature ablation for full backtracking prediction

Features	Accuracy	F1	AUC
All features	0.838	0.839	0.901
Lexical + length	0.824	0.826	0.889
Coreferences + length	0.802	0.806	0.861
Nouns + length	0.786	0.786	0.856
Entities + length	0.785	0.788	0.841
Length only	0.747	0.739	0.809

We notice that the lexical features hold the most predictive power, followed by coreference features. The strong performance of the coreference features, which characterize the average length and span of coreference chains in the text, is especially illuminating. These features represent a single facet of the textual context, compared to the set of different features included in the "lexical" set

When applied to the other two forms of backtracking, the complete model with all features can predict simple and continued backtracking with accuracy of 55% and 60% respectively, compared to the 85% accuracy of full backtracking prediction, the reduced performance of these models suggests that textual features are less useful in predicting simple and continued backtracking events. This follows our hypothesis that these two forms of backtracking are driven to other aspects of user actions, such as scrolling for inter-page navigation purposes.

Another aspect of the prediction task is the amount of information we have on every page. As mentioned earlier, pages with fewer than 20 views were discarded from our prediction task. We wish to know if requiring more information (a larger number of sessions) per page could improve the prediction task. We performed the training, test and prediction task on different subsets P_i of the full dataset, using a different minimum session count i for pages ($i = 1, 5, 10, 15, \ldots, 50$). The results are shown in Figure 6. The x-axis represents the minimum number of sessions per page (i), and the y-axis represents the average prediction accuracy, calculated in the same manner as defined earlier. For example, the prediction accuracy in P_{10} with the full model (red line, full circle markers) is over 80% (close to P_{20} performance), while length-only prediction on the same dataset (green line, empty circles) is 73.7%. We see that there is an increase in performance with a higher threshold for the minimum number of sessions per page, despite using fewer data points as i grows. Looking at the different sets of features, we see that all sets of features increase with a higher threshold of minimum views, but the improvement curve flattens out after $i = 20$. While we expect larger thresholds to yield better performance when more data is available, Figure 6 shows that it is sufficient to use 20 sessions per page to get enough signal for a prediction task.

4.3 Controlling for Users and Topics

We show above that the textual features of an article could help predict pages with high versus low levels of full backtracking. However,

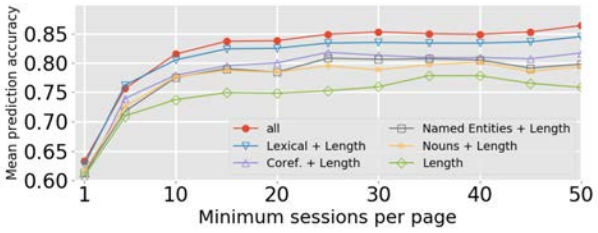


Figure 6: Prediction accuracy of different minimum sessions

we do not rule out the possibility that the text-based features are a function of the topic of the article (e.g., "health" or "sports"), or that the reader's choice of topics and articles to read is the one causing the observed differences in backtracking. For example, it could be that sports articles are easier to read or that users who choose to read sports articles tend to backtrack more often. In this section, we extend the analysis above to control for topics and users.

Topic vector. Article topics have been shown to have an effect on a user's reading behavior [7, 19]. Do topics have an effect on backtracking as well, and do they explain away the impact of any of the textual features? To examine this question, we used the set topics associated with every article as provided by LONG. We use the 35 most representative topics by removing from the dataset those that are either too rare or too frequent. From the selected set of topics, we produce a binary topic vector $T_i \in \{0,1\}^{35}$ for every page. We then add this vector as a new feature to our previously described backtracking prediction model. We do not observe any significant changes in the results, reaching +-1% in terms of accuracy, F1, and AUC compared to the results of the original model, as shown in Table 3. We also verified that the topic vector on its own is not a strong predictor of backtracking. Using the length of a page as control, a length + topic vector prediction task reaches 0.755, 0.747, 0.815 in terms accuracy, F1, and AUC — i.e. does not performs any better than a prediction based on length alone (as shown in Table 3).

Types of users. To effectively account for the effect of users choice of articles on backtracking we perform a similar prediction model to the one above, this time using the reduced-dimensionality user-page matrix as a feature in the prediction. Kallus et al [27] have shown that the use of matrix factorization to infer confounders can reduce the bias caused by measurement noise. Assuming specific users visit specific pages, we wish to learn the latent factors capturing different types of users, and further explore their significance on backtracking behavior. To this effect, we have built a binary page-user matrix $X \in \mathbb{R}^{mxn}$, where m, n represent the number of pages and users respectively from Table 1 and where X_{ij} is assigned the value 1 if user j visited page i and the value 0 otherwise.

In order to discover latent factors or low-dimensional embedding, we performed the matrix factorization $X \approx \hat{X} = UV$ or $\hat{X}_{ij} = U_i^T V_i$. Matrix $U \in \mathbb{R}^{mxk}$ holds the pages latent factors, while $V \in \mathbb{R}^{kxn}$ holds the users' latent factors. We take as a simplifying assumption that the difference between X and $\hat{X}$ is normally distributed, which allows us to use the mean-square error as our loss function.

We note as Ω the set of all observed (i, j) sessions. In the current settings, all sessions are observed, and we know for a fact whether or not a user has visited a given page. However, in order to better generalize our latent factors and account for the noise in our observed matrix, we hold out part of the of indices and perform cross-validation. This method is commonly used in collaborative filtering methods [35] in order to avoid overfitting. We add normally distributed regularization parameters to our model. This results into a new loss function, as stated below:

$$J = \sum_{(i,j)\in\Omega} (X_{ij} - U_i^T V_j)^2 + \lambda(\|U\|_F + \|V\|_F) \tag{1}$$

We use the alternative least squares [11] algorithm to minimize our loss function in an iterative way. This algorithm alternatively sets one matrix (U or V), and estimates the other by minimizing the loss function via derivation. This step is followed by adding the factor below to the latent factors' U, V matrices, iteratively until convergence:

$$\begin{cases} U_i = \left(\sum\limits_{j\in\Omega_i} v_j v_j^T + \lambda I \right)^{-1} \sum\limits_{j\in\Omega_i} (X_{ij} - \mu)\, v_j \\ V_j = \left(\sum\limits_{i\in\Omega_j} u_i u_i^T + \lambda I \right)^{-1} \sum\limits_{i\in\Omega_j} (X_{ij} - \mu)\, u_i \end{cases} \tag{2}$$

where Ω_i is the set of all users that visited page i, and Ω_j is the set of all pages that user j visited.

Similarly to the topics vector, adding the latent factors of pages to the prediction model showed a slight degradation in the prediction result on the held-out set, reaching 81.2%, 81.3%, 86.7% for accuracy, F1 score and AUC. In addition, the low-dimensional factors (with length of page as control) perform similar to our length-only baseline and reach 75.3%, 75.2%, 79.2% for accuracy, F1 score and AUC respectively.

5 CONCLUSION AND FUTURE WORK

This paper examines an under-explored signal, backtracking, and what it can teach us about the content of online news articles. We define and explore different types of backtracking actions on Web-based articles, showing that different news sites exhibit different backtracking patterns. We also demonstrate that full backtracking events can be accurately predicted with textual features related to readability assessment. By controlling for readers and topics, we discover that backtracking behavior is driven primarily by the textual content of an article. Our findings open a promising opportunity towards revisiting readability using large-scale web signals.

Our work has several limitations, related to data size and scope. Future work can extend the current work by analyzing a larger news corpora, or different content types, such as e-books. While our work only examined content from desktop interactions, understanding backtracking behavior across both desktop and mobile devices could expose different types of behaviors and perhaps more detailed understanding of backtracking. Finally, future work can leverage the insights about language's impact on readability and backtracking behavior. Such lines of work include supporting authors and editors in crafting content and helping struggling readers navigate text with language features known to hinder reading, such as coreferences. In both cases, the impact of these interventions could be measured at scale using the backtracking signal.

6 ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1840751, and by Yahoo! Research through the Connected Experiences Lab at Cornell Tech. We thank the Chartbeat data science team for generosity with their time and their data.

REFERENCES

[1] Sandra Aluisio, Lucia Specia, Caroline Gasperin, and Carolina Scarton. 2010. Readability Assessment for Text Simplification. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications (IUNLPBEA '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 1–9.

[2] Justin Cheng, Lada Adamic, Alex Dow, Jon Kleinberg, and Jure Lescovec. 2014. Can Cascades be Predicted?. In *Proceedings of the 23rd international conference on World wide web (WWW '14)*. ACM, New York, NY, USA, 925–936.

[3] William H, DuBay. 2007. The Classic Readability Studies. *Online Submission* (2007).

[4] Jeanne S Chall, Edgar Dale. 1948. A formula for predicting readability: Instructions. *Educational research bulletin* (1948), 37–54.

[5] Lijun Feng, Martin Jansche, Matt Huenerfauth, and Noémie Elhadad. 2010. A Comparison of Features for Automatic Readability Assessment. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters (COLING '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 276–284.

[6] Rudolph Flesch. 1948. A new readability yardstick. *Journal of applied psychology* 32, 3 (1948), 221.

[7] Nir Grinberg. 2018. Identifying Modes of User Engagement with Online News and Their Relationship to Information Gain in Text. In *Proceedings of the 2018 Web Conference (WWW '18)*. ACM, New York, NY, USA.

[8] Max Grusky, Jeiran Jahani, Josh Schwartz, Dan Valente, Yoav Artzi, and Mor Naaman. 2017. Modeling Sub-Document Attention Using Viewport Time. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 6475–6480.

[9] Robert Gunning. 1969. The fog index after twenty years. *Journal of Business Communication* 6, 2 (1969), 3–13.

[10] Qi Guo, Haojian Jin, Dmitry Lagun, Shuai Yuan, and Eugene Agichtein. 2013. Mining Touch Interaction Data on Mobile Devices to Predict Web Search Result Relevance. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval (SIGIR '13)*. ACM, New York, NY, USA, 153–162.

[11] Trevor Hastie, Rahul Mazumder, Jason D Lee, and Reza Zadeh. 2015. Matrix completion and low-rank svd via fast alternating least squares. *Journal of Machine Learning Research* (2015), 3367–3402.

[12] Michael Heilman, Kevyn Collins-Thompson, James P. Callan, and Maxine Eskénazi. 2007. Combining Lexical and Grammatical Features to Improve Readability Measures for First and Second Language Texts. In *HLT-NAACL*.

[13] Michael Heilman, Kevyn Collins-Thompson, and Maxine Eskenazi. 2008. An Analysis of Statistical Models and Features for Reading Difficulty Prediction. In *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications (EANL '08)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 71–79.

[14] Rosie Jones Henry A. Feild, James Allan. 2010. Predicting Searcher Frustration. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '10)*. ACM, New York, NY, USA, 34–41.

[15] Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

[16] Thorsten Joachims and Laura Granka. 2005. Accurately Interpreting Clickthrough Data as Implicit Feedback. In *Proceedings of the 28 International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '05)*. ACM, New York, NY, USA.

[17] Alan Kennedy, Robin Hill, and Joël Pynte. 2003. The dundee corpus. In *Proceedings of the 12th European conference on eye movement*.

[18] J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. *Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel*. Technical Report. Naval Technical Training Command Millington TN Research Branch.

[19] Dmitry Lagun and Mounia Lalmas. 2016. Understanding User Attention and Engagement in Online News Reading. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM '16)*. ACM, New York, NY, USA, 113–122.

[20] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

[21] Annie Louis and Ani Nenkova. 2013. What Makes Writing Great? First Experiments on Article Quality Prediction in the Science Journalism Domain. *TACL* 1 (2013), 341–352.

[22] Keith Rayner, Lyn Frazier. 1982. Making and Correcting Errors during Sentence Comprehension: Eye Movements in the Analysis of Structurally Ambiguous Sentences. *Cognitive Psychology* (1982).

[23] Patricia A. Carpenter, Marcel Adam Just. 1980. A Theory of Reading: From Eye Fixations to Comprehension. *Psychological review* (1980).

[24] Pekka Niemi, Marja Vauras, Jukka Hyona. 1992. Comprehending coherent and incoherent texts: evidence from eye movement patterns and recall performance. *Journal of Research in reading* (1992).

[25] G. Harry, Mc Laughlin. 1969. SMOG grading-a new readability formula. *Journal of reading* 12, 8 (1969), 639–646.

[26] Ta Lin Liau, Meri Colema. 1975. A computer readability formula designed for machine scoring. *Journal of Applied Psychology* 60, 2 (1975), 283.

[27] Madeleine Udell, Nathan Kallus, Xiaojie Mao. [n. d.]. Causal Inference with Noisy and Missing Covariates via Matrix Factorization. In *arXiv:1806.00811*.

[28] Sarah E. Petersen and Mari Ostendorf. 2009. A Machine Learning Approach to Reading Level Assessment. *Comput. Speech Lang.* 23, 1 (Jan. 2009), 89–106.

[29] Ildikó Pilán, Sowmya Vajjala, and Elena Volodina. 2016. A Readable Read: Automatic Assessment of Language Learning Materials based on Linguistic Complexity. *Int. J. Comput. Linguistics Appl.* 7 (2016), 143–159.

[30] Emily Pitler and Ani Nenkova. 2008. Revisiting Readability: A Unified Framework for Predicting Text Quality. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 186–195.

[31] Keith Rayner. 1998. Eye Movements in Reading and Information Processing: 20 Years of Research. *Psychological Bulletin* (1998).

[32] Keith Rayner, Kathryn H. Chace, Timothy J. Slattery, and Jane Ashby. 2006. Eye Movements as Reflections of Comprehension Processes in Reading. *Scientific Studies of Reading* (2006).

[33] Mirella Lapata, Regina Barzilay. 2008. Modeling Local Coherence: An Entity-based Approach. *Comput. Linguist.* 34, 1 (March 2008), 1–34.

[34] Edgar A Smith and RJ Senter. 1967. Automated readability index. *AMRL-TR. Aerospace Medical Research Laboratories (US)* (1967), 1–14.

[35] Chris Volinsky, Yehuda Koren, Robert Bell. 1982. Matrix Factorization Techniques for Recommender Systems. *Computer* (1982), 30–37.

[36] Chris Volinsky, Yehuda Koren, Robert Bell. 2009. Eye movements when viewing advertisements. *Journal of frontiers in psychology* 43 (2009). Issue 8.

Unnecessarily Identifiable: Quantifying the fingerprintability of browser extensions due to bloat

Oleksii Starov†, Pierre Laperdrix†, Alexandros Kapravelos‡, Nick Nikiforakis†
†Stony Brook University
‡North Carolina State University

ABSTRACT

In this paper, we investigate to what extent the page modifications that make browser extensions fingerprintable are necessary for their operation. We characterize page modifications that are completely unnecessary for the extension's functionality as *extension bloat*. By analyzing 58,034 extensions from the Google Chrome store, we discovered that 5.7% of them were unnecessarily identifiable because of extension bloat. To protect users against unnecessary extension fingerprinting due to bloat, we describe the design and implementation of an in-browser mechanism that provides coarse-grained access control for extensions on all websites. The proposed mechanism and its built-in policies, does not only protect users from fingerprinting, but also offers additional protection against malicious extensions exfiltrating user data from sensitive websites.

ACM Reference Format:
Oleksii Starov†, Pierre Laperdrix†, Alexandros Kapravelos‡, Nick Nikiforakis†. 2019. Unnecessarily Identifiable: Quantifying the fingerprintability of browser extensions due to bloat. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313458

1 INTRODUCTION

The past two decades have witnessed the web expanding and evolving at a tremendous rate with a similar expansion of the devices that users utilize to browse the web. However, this booming diversity of devices brought with it a new tracking technique called browser fingerprinting [8, 10–13, 16, 17]. Browser extensions are a recent addition to the fingerprinting domain [14, 18, 20, 23]. In order to provide added functionality, many extensions require full access to the pages that users visit. Some extensions modify existing elements from the Document Object Model (DOM) of webpages while others create new ones. Interacting with the DOM has the unfortunate consequence of causing side-effects which can later be detected and attributed back to installed extensions. In previous work, we showed that the presence of many browser extensions can be inferred just by looking at the modifications that they perform to the DOM [23]. Any malicious script running on the same webpage can then try to fingerprint the list of extensions installed in the browser. This poses a serious privacy risk as extension fingerprintability can lead to the identification of a user.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313458

Fixing the problem of fingerprintable browser extensions is not straightforward. Browser extensions are intended to access and modify webpages that users are visiting and, in many cases, this is their main and desired functionality (e.g., hiding ads, highlighting search results, or enlarging photo thumbnails). At the same time, the question remains whether such functional side effects need to manifest on all pages unconditionally. For example, if a webpage does not contain ads, search results, photos, or other triggering content, it is completely unnecessary for an extension to disclose its presence since it will be hurting the privacy of the user while delivering no useful functionality.

In this paper, we investigate whether the page modifications that make extensions fingerprintable are necessary for their operation. We define the notion of *bloat* in the implementation logic of browser extensions as the unnecessary side effects that offer no functionality to users and can be abused for extension identification. By analyzing 58,034 extensions from the Google Chrome store, we identify the behaviors characteristic of bloat and quantify its prevalence in the wild. We study the origin of fingerprintable bloat and discover cases of shared libraries and common coding practices, which contribute to it. Moreover, we show that bloat is responsible for a large fraction of all fingerprintable on-page changes from extensions.

Orthogonally to our measurement and characterization of extension bloat, we argue that it is important for users to have client-side control over the reach of browser extensions and their interactions with the DOM of webpages. We provide the design and implementation details of a client-side access control mechanism that mitigates the aforementioned privacy risks that derive from bloat by empowering users to selectively enable/disable extensions on arbitrary websites. Moreover, we discuss how the same client-side access-control mechanism protects against general privacy risks of untrustworthy browser extensions, i.e., limiting or altogether stopping the leakage of browsing history and other sensitive on-page information by buggy and malicious extensions.

2 BLOAT-RELATED FINGERPRINTABILITY OF BROWSER EXTENSIONS

Software bloat refers to software including unnecessary code and libraries while the process of *debloating* denotes the removal of the excess components. For this paper, we focus on software bloat in terms of unnecessary implementation logic inside browser extensions, which results in them being unnecessarily fingerprintable. We use the following definition:

Definition. *Bloat-related side effects of browser extensions are artifacts on a web page which do not deliver functionality that is desired by users, yet reveal the extension's presence to trackers and fingerprinting scripts.*

2.1 Types of bloat

When the user visits a page with an installed extension, both the extension and the page have access to the same Document Object Model (DOM), which provides a programmatic interface of the page to JavaScript code. The extension can add or remove elements from the DOM of the visited page by injecting JavaScript. The core observation of our definition with regard to extension bloat, is that an extension should not modify the DOM of the visited page, unless it is absolutely necessary for its functionality. Unnecessary DOM modifications by the extension disclose its presence to visited pages and therefore to trackers situated on those pages.

Some extensions will perform changes on the visited page that are part of their intended functionality. For example, static panels filled with information (e.g., showing website statistics), toolbars with URL-independent tools (e.g., CSS inspectors), and modified text (e.g., extensions to correct the spelling of words) are all intentional. One may notice that the common feature of all of these examples is the presence of useful functionality, even if that functionality manifests on all pages.

To identify cases of true extension bloat, i.e., modifications that are unnecessary since they do not provide any useful function, we manually analyzed the types of changes that extensions perform on an empty page and their purpose. Our intuition was that an empty webpage has no existing content to trigger the logic of browser extensions. As such, any DOM modifications are potentially the result of bloat and should be investigated. Through this manual experiment we identify the following extension-independent categories of unnecessary DOM modifications:

•**Injected empty placeholders**. We discovered that it is a common practice among many extensions to include empty or invisible DOM elements first, which will be filled with data later based on a trigger (e.g., when specific content appears on a page, or when a user clicks the extension's icon) (Figure 1). The direct consequence of this preemptive DOM modification is that these placeholders unnecessarily increase the fingerprintability of extensions especially in the cases where the extension remains otherwise inactive.

•**Injected script tags**. Some extensions add script tags to the page's DOM, which are visible to any other script executing on the page. However, extensions have the ability to add and hide script tags, before any original code is executed on the page, by properly registering content scripts (i.e. scripts running together with webpages with access to a shared DOM).

•**Injected style tags**. Many extensions inject CSS tags into the page's DOM, which are visible to any script on the page. This behavior is unnecessary because there exist special APIs for extensions to inject CSS stylesheets in a page (e.g. `chrome.tabs.insertCSS`). When an extension uses this API, the added styles are not visible via standard CSS-querying APIs, such as, `document.styleSheets`. Using these methods, the injection does not modify the DOM tree.

•**Attributes for body/head/html tags**. Bloat also occurs when an extension sets attributes to parent DOM nodes, such as the `body`, `head` or `html` tag. We discovered many cases of custom attributes with words "installed" or "injected", which do nothing except disclose the extension's presence. Even if these attributes are used in CSS selectors, given that they target well-defined tags, they are unnecessary and can be removed from the corresponding CSS rules.

```
// Unnecessary and invisible placeholders
<div class="plugin-body" style="display:none;">
</div>
<div id="addManualImageDiv" style="display: none;">
</div>
<div id="mystickies"></div>
// Unnecessary top-level attributes
<html style="visibility: visible;">
<body screen_capture_injected ="true">
<body style="">
```

Figure 1: Examples of bloat-related DOM modifications of browser extensions.

Similarly to empty placeholders, attributes with empty values are signs of bloat in the extension's logic. The bottom half of Figure 1 illustrates several real examples of such behavior.

•**Window messages**. The browser provides a programmatic way for different JavaScript execution contexts to communicate via messages. Extensions can send a message to the visited page using `window.postMessage`, but that message can be received by any script running on the page that has registered the appropriate callback function. Since these messages reveal the presence of the extension to the visited page, we consider the presence of such messages on an empty webpage as extension bloat.

2.2 Quantification of extension bloat

To measure the fingerprintability due to browser-extension bloat, we collected 58,034 extensions from the Chrome Store in October 2017. These include 25,779 extensions with permissions to modify web pages on any URL. We evaluated each extension according to the necessary condition of our bloat definition. Namely, we retrieved all the fingerprintable DOM modifications that an extension introduces on any domain and filtered out ubiquitous side effects which were part of an extension's desirable functionality.

In order to automate the testing of extensions, we modified and used the source code of XHound [23]. Given the more narrow definition of the bloat-related DOM modifications (compared to *all* modifications that XHound can uncover), we simplified XHound by taking each extension to a custom domain only once, simulating the visiting of an arbitrary website. Moreover, we served an empty HTML page to extensions in order to eliminate the possibility of any specific content triggers, and we disabled the usage of the XHound's OnTheFlyDOM library as we are not interested in DOM modifications in response to specific content or user actions. As such, we collected only those fingerprintable side-effects that occur on an arbitrary webpage without any triggering content and without user actions. To further minimize the probability of false positives, we filtered out any cases of discovered "bloat" which do not conform to our definition, therefore calculating a lower bound of bloat-related fingerprintability.

Overall, we discovered 3,320 extensions, or 5.7% of all the extensions in Chrome's webstore, which have at least one fingerprintable side effect because of bloat. Out of those, 2,189 inject unnecessary nodes into the DOM (e.g., div, span, script, style), and 1,526 set unnecessary attributes to the body, head or HTML nodes. Moreover, 65% out of 3,320 extensions had *all* their changes categorized as bloat. In other words, these 2,145 extensions should have been completely invisible on our test pages. Figure 2 shows that both popular and less popular extensions are affected by this type of

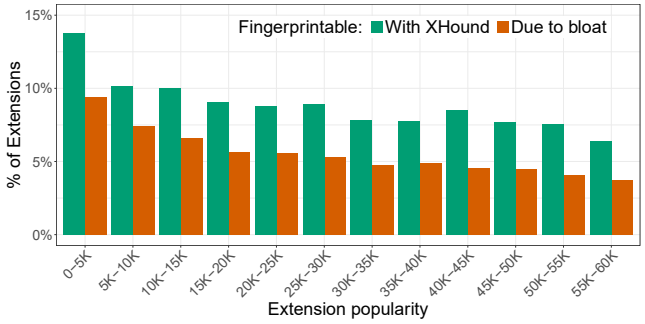


Figure 2: Percentage of fingerprintable extensions, and extensions with fingerprintable bloat, shown by extension popularity. High-ranked extensions tend to be more fingerprintable, and bloat-related side effects contribute to a stable large fraction of fingerprints.

bloat with more than 9% of the top 5K extensions containing bloat, and approximately 4% of the less popular extensions.

Apart from bloat-related DOM modifications, we discovered that 213 extensions reveal their presence by posting disclosing messages to web pages upon installation. We were surprised to find such a large number of extensions since Chrome APIs provide the ability to set up communication channels with only selected web pages with the `externally_connectable` permission. Moreover, we discovered 637 extensions listening to messages from any web page. This means that a tracker can potentially craft and send expected messages and wait for visible side effects from extensions in response. For example, as shown on Figure 3, Crypto-Plugin, a banking extension with 304,730 active users, responds to a specially-crafted message if there is an input element with id `CryptoPluginInstance` and a value containing a JSON-formatted command. Note that this behavior of taking arbitrary content and treating it as command can, in addition to make extensions unnecessarily identifiable, lead to severe security issues.

2.3 Impact on extension fingerprintability

In this section, we compare our results to the unmodified XHound prototype, which deploys additional techniques for triggering the functionality of browser extensions and can thus discover more fingerprintable side-effects. We find that a large fraction of fingerprintable extensions are actually fingerprintable due to bloat and could thus become invisible to trackers, if their bloat-related DOM modifications are removed.

Overall, using XHound, we could identify 5,323 fingerprintable extensions. Given our earlier finding of 3,320 extensions introducing bloat-related fingerprintable side effects, approximately 62% of all the fingerprintable extensions have at least one bloat-related side effect, which adds to their fingerprint. Moreover, even when using XHound's on-the-fly triggering content, 1,073 extensions remain with bloat only, suggesting that this bloat can be safely removed to make extensions invisible from prying webpages.

The bloat-related side effects of 3,320 Chrome extensions resulted in 2,032 extensions sharing their bloat-related fingerprint with no other extension (i.e. their bloat-related DOM modifications are unique to them and could be used to uniquely identify them). In terms of anonymity set sizes: 61.2% had totally distinct fingerprints, 8.8% shared their fingerprint with up to 10 other extensions, 7.2%

```
window.addEventListener("message", function(event) {
  if (event.data.sender == "crypto.plugin.native") {
    console.log(JSON.stringify(event.data));
  }
}, false);

var dummy_input = document.createElement("input");
dummy_input.setAttribute("value", "{}");
dummy_input.id = "CryptoPluginInstance";
document.body.append(dummy_input);

window.postMessage({sender: "crypto.plugin.js"}, "*")

// CONSOLE OUTPUT:
{"answer":{"errorCode":1,"errorText":"Wrong function"},
"type":"error", "sender":"crypto.plugin.native"}
```

Figure 3: Detecting the Crypto-Plugin extension by sending an appropriately-crafted message, simulating a triggering DOM content, and listening to the response.

shared with up to 100 extensions, while 22.8% shared their fingerprint with more than 100 extensions because of similar bloating changes related to jQuery (we discuss this in the next section). As a result, unnecessary bloat-related fingerprintability is, in and of itself, a clearly dangerous extension-fingerprinting vector.

2.4 Origin of bloat in browser extensions

According to our manual analysis of extension bloat (Section 2.1), poorly-designed internal logic is the most common reason for extension bloat. First, we discovered 1,070 extensions with the design flaw of adding placeholders before filling them with useful content. Second, we witnessed several cases of implementation bugs when a DOM modification appears because of poor coding practices. For example, the code in Figure 4 adds an empty class attribute to the body node even if `eod_disabled` was not set initially.

Next to extensions having their custom logic leading to unique cases of bloat, we discovered several libraries that perform unnecessary modifications in the DOM. For example, the jQuery library, when included as a content script, injects an empty style to the body of web pages (discovered on more than 980 extensions). Moreover, particular versions of jQuery have other hardcoded side effects like an injected div tag with id `pg_hgfkj4kj32mda` (version 1.7.2), or the added style `zoom: 1;` to the body of web pages (version 1.9.0). We also discovered the FancyBox JavaScript library on 65 extensions, which adds custom styles to web pages. Additionally, the CrossriderAPI, a cloud-based extension development tool, adds its own attribute to a page's body, namely `crossrider_data_store_temp`, on at least 45 extensions. In addition to the aforementioned cases of bloat, we discovered other cases of the same bloat modifications manifesting across tens of extensions but we could not attribute them to a particular library. These types of modifications could be due to reused code across related extensions or extensions developed by the same developers.

In summary, we discovered that a large number of browser extensions introduce fingerprintable modifications that could have been avoided and which are not necessary for the proper operation of the extension. On the one hand, this is a positive development since this finding suggests that the more careful design and implementation of browser extensions would allow users to be less fingerprintable, without the need to avoid using these extensions. On the other hand, solely relying on extension developers to voluntarily make their

```
document.documentElement.className =
  document.documentElement.className.replace(
    /eod_disabled/g, "")
```

Figure 4: An example of implementation bug in browser extension, which results in unnecessarily fingerprintable DOM modification

extensions more robust against browser fingerprinting, is likely not going to lead to drastic improvements with regard to user privacy. As such, we argue that there is a need to protect users, even when extension authors are not committed to change the logic of their extensions. We discuss such a solution in Section 3.

By comparing the set of fingerprintable/undetectable extensions throughout the years, we discovered that most extensions retain their original status. That is, if the first version of an analyzed extension was fingerprintable, the extension remains fingerprintable through subsequent updates. Overall, only 258 extensions from the initial 1,000 fingerprintable set were not fingerprintable in their first version. Similarly, most extensions that were originally undetectable, remain undetectable. This finding suggests that the fingerprintable bloat of extensions remains stable over time and thus requires an explicit attempt to remove it (i.e. will not disappear in future versions of extensions).

In the following, we present a few case studies of extensions which migrated from undetectable to fingerprintable and vice-versa. One example of an extension which became fingerprintable is Extended JS Console. It was undetectable for two years until the developer used the Crossrider framework to make development across different browsers easier. This framework injects a `crossrider_data_store_temp` attribute to the list of body tags that makes the framework easily detectable. Another example is the Dayboard extension which, after 18 undetectable versions, started to inject a `db-visibility-check` empty div on all visited page. Fortunately, there are also 72 extensions which were once fingerprintable but later became undetectable, e.g., the Web PKI, which, since February 2018, has stopped injecting empty div placeholders.

3 COUNTERMEASURES

Since extension authors cannot be trusted to appropriately limit the access of their own extensions (either due to the inability to specify a finite whitelist that works for all users, a lack of motive, or outright maliciousness), the limiting of those extensions must happen at the client-side.

This means that, in addition to extension authors specifying on which websites a given extension should run, there should be a second layer of access-control which empowers users to limit the access of extensions according to their needs. For example, users may want to disable ad-blockers on specific websites which they want to support with ad-revenue, even if the ad-blockers do not support per-website whitelisting. Similarly, users may want to disable all extensions on their banking website, to limit their exposure, in case of accidental information leakage [22] or malicious data exfiltration [7, 9, 15].

Instead of relying on users to whitelist/blacklist each of their installed extensions on every domain that they visit, we argue that we can build higher-level access-control primitives which can be intuitively utilized by users with limited technical expertise. Specifically, we propose primitives that take the form of the following access-control modes:

```
{   "ExtensionSettings":{
      #UBlock Origin
      "cjpalhdlnbpafiamejdnhcphjbkeiagm":{
         "runtime_blocked_hosts":[
            "*://*.bankofamerica.com"
         ]
      },
      #Dark Mode
      "dmghijelimhndkbmpgbldicpogfkceaj":{
         "runtime_blocked_hosts":[
            "*://*.google.com",
            "*://*.bankofamerica.com"
         ]
}}}
```

Figure 5: Example of an ExtensionSettings policy for two separate extensions.

•**Blocking extensions by default**. The goal of this mode is to stop "drive-by fingerprinting", i.e., the complete fingerprinting of a user's extensions just because a user visited an untrusted website.

•**Automatic disabling of extensions on sensitive domains**. We argue that there is often no need for extensions to be able to run in any and all websites that the users visit. As such, given a list of sensitive URLs (such as banking websites, social networks, dating websites, and healthcare-related websites), the browser can automatically disable all extensions on these domains and allow users to re-enable only the ones that they absolutely need.

•**Content-based automated blocking**. Next to maintaining lists of sensitive websites, we argue that a user's exposure can be further reduced if a browser disables extensions when a webpage allows users to input content. For example, disabling extensions on login pages could help reduce the likelihood of an extension exfiltrating user credentials.

Given the HTTPS issues of client-side proxies and the lack of APIs that a browser extension can use to block other browser extensions, a client-side access control mechanism for browser extensions *has* to include some level of browser modifications. By inspecting the source code of Chromium (our browser of choice given the availability of its source code and its market share) and discussing with developers of Chromium, we discovered the presence of an *Enterprise* mechanism built in the browser. Enterprise mechanisms are typically utilized in corporate environments where system administrators wish to limit the features that are available to individual deployments of powerful software, such as, a web browser. Even though these features are completely invisible to regular users, we discovered that the Enterprise features of Chromium were introduced in 2012 and are still part of the code base of the regular browser that users download.

Chromium's enterprise mechanisms are driven by an enterprise policy written in a platform-dependent format (e.g. JSON on Linux) which allows administrators to manage the browser's configuration and behavior. These policies cover a wide range of browser settings, from power management and proxy configurations, to printers, startup pages, and the default search engine. For our purposes, the `ExtensionSetting` policy allows administrators to disable specific extensions, and stop users from installing any extension that requests specific permissions [2]. The following two directives that control which websites can be accessed by extensions, are of particular interest for our client-side access control mechanism:

•`runtime_blocked_hosts`: List of strings representing hosts whose webpages the extension will be blocked from modifying.

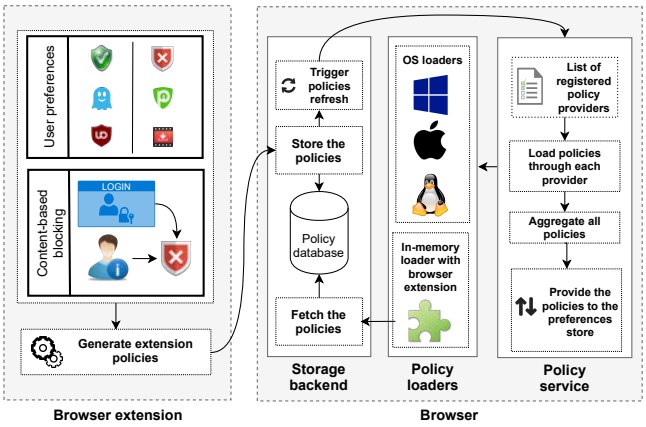


Figure 6: Implementation overview of the client-side access control mechanism for browser extensions.

•`runtime_allowed_hosts`: List of strings representing hosts that an extension can interact with, regardless of whether they are listed in `runtime_blocked_hosts`.

Figure 5 shows an example of an *ExtensionSettings* policy. The first ID refers to the ad blocker "uBlock Origin" and the second to the "Dark Mode" extension (an extension that changes the background color of a website and which we will use in a later demonstration). This policy instructs the browser to block both extensions from accessing any webpages from Bank of America but only the ad blocker is allowed to operate on `google.com`.

The files containing the Enterprise policies, are expected to be present in predefined filesystem paths, according to the operating system of the user. If such a file is discovered during the startup of the browser, it is loaded and enforced by Chromium.

By retrofitting the already available enterprise mechanism and exposing it to regular users through a dedicated UI that converts user choices to JSON policies, we can enable client-side access control of extensions with minimal changes to the browser's code. In addition to modifying as little as possible of the browser's existing code (therefore increasing our chances of adoption), we also rely on a proven mechanism that provides comprehensive extension blocking in a way that is not possible by combining blocking directives available to extension authors (such as the `exclude_matches` directive which allows extension authors to opt-out of running on specific domains and URLs).

The source code of our modified Chromium browser along with a fully compiled version can be found on GitHub: `https://github.com/plaperdr/extension-access-control`.

3.1 Interfacing with the user

Since Enterprise policies are fully controlled by JSON-formatted files available in obscure file paths, we cannot reasonably expect users to directly interact with them. To bridge the Enterprise policy mechanism with the interfaces that users are already familiar with, we developed a browser extension that creates policies based on user choices. Next to allowing users to individually disable extensions on websites through our extension's UI, we provide support for the following three blocking strategies, which map to the access-control primitives described in Section 3:

•**Flexible mode (default mode)**: All extensions are allowed on all webpages and the user can disable (or re-enable) at any time an extension on a specific website. Note that this mode does not interfere with the selective exclusion that extension authors may request using the `exclude_matches` directive in their manifest files. As such, extensions that were previously active only on selected URLs (e.g. a video-downloading extension being active only on `youtube.com`) remain active only on the same URLs.

•**Sensitive mode**: This mode is similar to the *Flexible mode* but comes preloaded with a list of sensitive websites where all extensions are automatically blocked. For our proof-of-concept implementation, this list consists of 50 popular banking websites. This mode also supports content-based blocking where extensions can be disabled if our mechanism detects the presence of sensitive content on a website that is not part of our predefined lists. Specifically, our current implementation blocks extensions whenever a password input field is detected in a webpage's HTML code.

•**Strict mode**: In this mode, all extensions are by default blocked on all websites. Users can selectively enable extensions wherever they require them.

Given a desired policy of blocking and enabling browser extensions on specific websites, this policy needs to be communicated to the enterprise policy-parsing mechanisms of the browser. Chromium has a relatively complex policy-loading system that spans dozens of files and thousands of lines of code. Figure 6 provides an overview of our modifications. First, our user-facing extension transforms all user choices and all rules derived from content-based blocking into an *ExtensionSettings* policy readable by the browser, indicating for each installed extension which hosts are blocked and which are allowed. This policy is then stored in a database that is available to our extension.

As mentioned earlier, the Enterprise mechanism expects policies to be available on different filesystem paths depending on the user's operating system. Specifically, Chrome's policy loader uses GPO (Group Policy Object) on Windows, XML files on MacOS, and JSON files on Linux. All of these locations are encoded as different "providers" in one combined mechanism called a *Policy Service*. For our implementation, we modified the Policy Service and added a new policy provider which pulls new policies directly from the database of our user-facing extension. Whenever a policy changes (e.g. because the user whitelisted/blacklisted an extension, or switched modes), our service triggers a "refresh" which causes Chromium's Policy Service to pull a new copy of the policy. The policies from all policy providers are then combined into a unique policy and transmitted to Chromium's *Preferences Store* that will parse it and apply it.

3.2 Evaluation

Performance Overhead. To test the overhead of our client-side access control mechanism, we visited the Alexa Top 50 websites with the original (i.e. unmodified) version of the Google Chromium browser (v.71) and our modified copy of the same version of the same browser. For each visited page, we recorded the time it takes for the browser to completely render every element. Specifically, we collected the timestamps corresponding to the following three events from the *window.performance.timing* object:

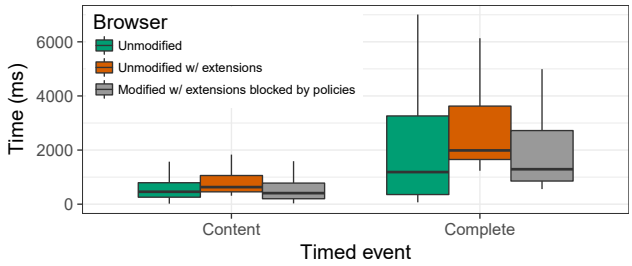


Figure 7: Performance measurements of unmodified Chromium version 71 with and without installed extensions, against our modified Chromium browser.

• the *DOMLoading* event is fired when the parser starts parsing the received webpage.

• the *DOMContentLoaded* event marks the point when both the DOM is ready and there are no stylesheets that are blocking JavaScript execution.This means that the browser can start combining the DOM and CSSOM into a render tree.

• the *DOMComplete* event is fired when the page and all of its subresources are ready.

We tested three different configurations: unmodified Chromium with no running extensions, unmodified Chromium with extensions, and our custom Chromium in "Strict" mode with extensions (i.e. all extensions are present but blocked by our custom policy). For our tests, we utilized five popular extensions: AdBlock [1], Google Keep [3], Grammarly [4], Honey [5], and LastPass [6]. The number of extensions was chosen to match the average number of extensions that regular users install, according to Starov et al. [23].

We repeated the measurements ten times and averaged the results, which are shown in Figure 7. The "Content" category corresponds to the time between the *DOMLoading* event and the *DOMContentLoaded* one. The "Complete" one is the time between *DOMLoading* and *DOMComplete*.

For both categories, we observe that the timings of our modified browser with all extensions disabled by the applied "Strict" policy closely match the timings of the unmodified browser without any extensions installed. Given that our mechanism makes use of existing Chromium code which we retrofit to give users the ability to make client-side access-control decisions, the obtained performance results make intuitive sense. Overall, our findings demonstrate that client-side access control of extensions not only will not slow down a user's browser but will, in fact, increase its performance by disabling extensions when they are not necessary.

Blocking Coverage. To test whether the extensions disabled through our access-control mechanism are completely disabled, we performed the following experiment. We selected 1,000 extensions which were introducing bloat-related fingerprintable DOM side effects and installed them, one by one, on our modified Chromium browser. We then used that browser together with the XHound tool [23] and attempted to extract fingerprints from each extension.

For the 985 extensions that we could install in our modified browser (15 out of the 1K sample would not properly execute on Chromium version 71), XHound was unable to extract fingerprintable DOM modifications, when these extensions were blocked through the "Strict" policy of our client-side access control mechanism. Given that XHound searches for any and all DOM modifications that can be attributed to an extension, this gives us confidence that our modified browser successfully disabled all extensions.

4 RELATED WORK

In the last few years, the privacy implications of browsers extensions have received significant attention [9, 14, 18, 19, 21–23, 25]. To the best of our knowledge, this is the first paper that i) quantifies to what extent the fingerprintable DOM modifications performed by extensions are necessary, ii) points out the resulting issue of *bloat* in the context of browser extensions, and iii) proposes a client-side access control mechanism for reducing the footprint of extensions in a user's browser which can counter many of the attacks described in the aforementioned papers. In concurrent work, Sjösten et al. described the privacy issues of unique extension identifiers and also proposed a client-side access control system for extensions [19]. While the two systems are conceptually similar, our work utilizes existing code within the Chromium browser thereby avoiding the disabling of security checks of browser extensions and the need for extension rewriting.

On October 1, 2018 and in parallel with our work, Google announced that future versions of their Chrome browser will allow users to selectively enable and disable extensions on different websites [24]. We find this a welcoming development and a confirmation of the need of client-side access control for browser extensions. The current instantiation of that mechanism allows users to whitelist extensions on a site-by-site basis, or only enable them when the user interacts with them through the browser's UI. We argue that our proposed policies strike a better balance between security and usability (e.g. extensions enabled everywhere *except* on sensitive sites) and we therefore hope that Google, as well as other browser vendors, will follow our proposed design.

5 CONCLUSION

In this paper, we investigated the fingerprintability of browser extensions due to bloat, i.e., the unnecessary side-effects caused by faulty application logic that reveal an extension's presence without providing any useful functionality. Bloated extensions represent a risk to online privacy as they facilitate the fingerprinting of installed extensions. We analyzed 58,034 extensions from the Google Chrome store and found that 5.7% of them contained fingerprintable bloat. For 61% of these extensions, their bloat was unique which can be abused for their direct identification. Finally, we presented the design and implementation of a client-side access control mechanism for Google Chromium which enables users to control the reach of extensions, either on a one-by-one basis, or via access modes (e.g. automatically blocking all extensions on sensitive websites). Overall, our paper highlights the problems associated with bloat for both users and developers. We hope that this work will motivate the more careful implementation of extensions by their developers and the adoption of client-side access control by all modern browsers.

Acknowledgements: This work was supported by the Office of Naval Research (ONR) under grant N00014-17-1-2541 and by the National Science Foundation (NSF) under grants CNS-1813974, CMMI-1842020, CNS-1617593, CNS-1703375, and CNS-1527086.

REFERENCES

[1] 2018. AdBlock - Chrome Web Store. https://chrome.google.com/webstore/detail/adblock/gighmmpiobklfepjocnamgkkbiglidom.

[2] 2018. Extension Settings Full Description | The Chromium Projects. https://www.chromium.org/administrators/policy-list-3/extension-settings-full.

[3] 2018. Google Keep Chrome Extension - Chrome Web Store. https://chrome.google.com/webstore/detail/google-keep-chrome-extens/lpcaedmchfhocbbapmcbpinfpgnhiddi.

[4] 2018. Grammarly for Chrome - Chrome Web Store. https://chrome.google.com/webstore/detail/grammarly-for-chrome/kbfnbcaeplbcioakkpcpgfkobkghlhen.

[5] 2018. Honey - Chrome Web Store. https://chrome.google.com/webstore/detail/honey/bmnlcjabgnpnenekpadlanbbkooimhnj.

[6] 2018. LastPass: Free Password Manager - Chrome Web Store. https://chrome.google.com/webstore/detail/lastpass-free-password-ma/hdokiejnpimakedhajhdlcegeplioahd.

[7] Lawrence Abrams. 2018. MEGA Chrome Extension Hacked To Steal Login Credentials and CryptoCurrency. https://www.bleepingcomputer.com/news/security/mega-chrome-extension-hacked-to-steal-login-credentials-and-cryptocurrency/.

[8] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. 2014. The Web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the 21st ACM Conference on Computer and Communications Security (CCS)*.

[9] Quan Chen and Alexandros Kapravelos. 2018. Mystique: Uncovering Information Leakage from Browser Extensions. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*.

[10] Anupam Das, Gunes Acar, Nikita Borisov, and Amogh Pradeep. 2018. The Web's Sixth Sense: A Study of Scripts Accessing Smartphone Sensors. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*.

[11] Peter Eckersley. 2010. How Unique Is Your Browser?. In *Proceedings of the Privacy Enhancing Technologies Symposium (PETS)*.

[12] David Fifield and Serge Egelman. 2015. Fingerprinting web users through font metrics. In *Financial Cryptography and Data Security*. Springer, 107–124.

[13] Alejandro Gómez-Boix, Pierre Laperdrix, and Benoit Baudry. 2018. Hiding in the Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale. In *Proceedings of the World Wide Web Conference (WWW)*.

[14] Gabor Gyorgy Gulyas, Doliere Francis Some, Nataliia Bielova, and Claude Castelluccia. 2018. To Extend or Not to Extend: On the Uniqueness of Browser Extensions and Web Logins. In *Proceedings of the 2018 Workshop on Privacy in the Electronic Society (WPES'18)*.

[15] Alexandros Kapravelos, Chris Grier, Neha Chachra, Chris Kruegel, Giovanni Vigna, and Vern Paxson. 2014. Hulk: Eliciting Malicious Behavior in Browser Extensions. In *Proceedings of USENIX Security Symposium*.

[16] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. 2016. Beauty and the Beast: Diverting modern web browsers to build unique browser fingerprints. In *Proceedings of the IEEE Symposium on Security and Privacy*.

[17] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2013. Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting. In *Proceedings of the IEEE Symposium on Security and Privacy*.

[18] Iskander Sanchez-Rola, Igor Santos, and Davide Balzarotti. 2017. Extension Breakdown: Security Analysis of Browsers Extension Resources Control Policies. In *Proceedings of USENIX Security Symposium*.

[19] Alexander Sjösten, Steven Van Acker, Pablo Picazo-Sanchez, and Andrei Sabelfeld. 2019. LATEX GLOVES: Protecting Browser Extensions from Probing and Revelation Attacks. In *Network and Distributed System Security Symposium (NDSS)*.

[20] Alexander Sjösten, Steven Van Acker, and Andrei Sabelfeld. 2017. Discovering Browser Extensions via Web Accessible Resources. In *Proceedings of the ACM on Conference on Data and Application Security and Privacy (CODASPY)*.

[21] Alexander Sjösten, Steven Van Acker, and Andrei Sabelfeld. 2017. Discovering browser extensions via web accessible resources. In *Proceedings of the ACM on Conference on Data and Application Security and Privacy (CODASPY)*.

[22] Oleksii Starov and Nick Nikiforakis. 2017. Extended Tracking Powers: Measuring the Privacy Diffusion Enabled by Browser Extensions. In *Proceedings of the International Conference on World Wide Web (WWW)*.

[23] Oleksii Starov and Nick Nikiforakis. 2017. XHOUND: Quantifying the Fingerprintability of Browser Extensions. In *Proceedings of the IEEE Symposium on Security and Privacy*.

[24] James Wagner. 2018. Trustworthy Chrome Extensions, by Default. https://security.googleblog.com/2018/10/trustworthy-chrome-extensions-by-default.html.

[25] Michael Weissbacher, Enrico Mariconti, Guillermo Suarez-Tangil, Gianluca Stringhini, William Robertson, and Engin Kirda. 2017. Ex-Ray: Detection of History-Leaking Browser Extensions. In *Proceedings of the ACM Annual Computer Security Applications Conference (ACSAC)*.

Embarrassingly Shallow Autoencoders for Sparse Data

Harald Steck
Netflix
Los Gatos, California
hsteck@netflix.com

ABSTRACT

Combining simple elements from the literature, we define a linear model that is geared toward sparse data, in particular implicit feedback data for recommender systems. We show that its training objective has a closed-form solution, and discuss the resulting conceptual insights. Surprisingly, this simple model achieves better ranking accuracy than various state-of-the-art collaborative-filtering approaches, including deep non-linear models, on most of the publicly available data-sets used in our experiments.

KEYWORDS

Recommender System; Collaborative Filtering; Autoencoder; Neighborhood Approach; Linear Regression; Closed-Form Solution

ACM Reference Format:
Harald Steck. 2019. Embarrassingly Shallow Autoencoders for Sparse Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313710

1 INTRODUCTION

Many recent improvements in collaborative filtering can be attributed to deep learning approaches, e.g, [6, 10–12, 17, 25, 30, 31]. Unlike in areas like computer vision, however, it was found that a *small* number of hidden layers achieved the best recommendation accuracy. In this paper, we take this to the extreme, and define a linear model *without* a hidden layer (see Figure 1). The (binary) input vector indicates which items a user has interacted with, and the model's objective (in its output layer) is to predict the best items to recommend to the user. This is done by reproducing the input as its output, as is typical for *auto*encoders.[1] We hence named it Embarrassingly Shallow AutoEncoder (in Reverse order: EASE$^{\text{R}}$).

This paper is organized as follows: we define EASE$^{\text{R}}$ in the next section, using simple elements from the literature. In Section 3.1, we derive the closed-form solution of its convex training objective. This has several implications: (1) it reveals that the neighborhood-based approaches used in collaborative filtering are based on conceptually incorrect item-item similarity-matrices, while EASE$^{\text{R}}$ may be considered a principled neighborhood model, see Sections 3.2 and 4.3; (2) the code for training EASE$^{\text{R}}$ is comprised of only a few lines, see Section 3.3 and Algorithm 1; (3) if the model fits into memory, the wall-clock time for training EASE$^{\text{R}}$ can be several orders of magnitude less than for training a SLIM [20] model (see Section 3.4), which is the most similar model to EASE$^{\text{R}}$. Apart from that, we surprisingly found that EASE$^{\text{R}}$ achieved competitive ranking accuracy, and even outperformed various deep, non-linear, or probabilistic models as well as neighborhood-based approaches on most of the publicly available data-sets used in our experiments (see Section 5).

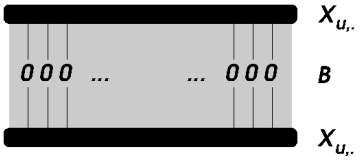


Figure 1: The self-similarity of each item is constrained to zero between the input and output layers.

2 MODEL DEFINITION

Like in many recommender papers that use implicit feedback data, we assume that the data are given in terms of a sparse (typically binary[2]) matrix $X \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$, regarding the sets of users $\mathcal{U}$ and items $\mathcal{I}$, where $|\cdot|$ denotes the size of a set. A positive value (typically 1) in X indicates that the user interacted with an item, while a value of 0 indicates that no interaction has been observed.

The parameters of the EASE$^{\text{R}}$ model are given by the item-item weight-matrix $B \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{I}|}$. Note that this is also similar to neighborhood-based approaches, see Section 4.3. In this weight matrix, self-similarity of an item in the input-layer with itself in the output layer is forbidden as to force the model to generalize when reproducing the input as its output (see Figure 1): hence the diagonal of the weight-matrix is constrained to zero, $\text{diag}(B) = 0$. This constraint is crucial, and is discussed in detail in the remainder of this paper. This constraint was first introduced in the SLIM model [20].

The predicted score $S_{u,j}$ for an item $j \in \mathcal{I}$ given a user $u \in \mathcal{U}$ is defined by the dot product

$$S_{uj} = X_{u,\cdot} \cdot B_{\cdot,j}, \tag{1}$$

where $X_{u,\cdot}$ refers to row u, and $B_{\cdot,j}$ to column j.

3 MODEL TRAINING

We use the following convex objective for learning the weights B:

$$\min_{B} \quad ||X - XB||_F^2 + \lambda \cdot ||B||_F^2 \tag{2}$$

$$\text{s.t.} \quad \text{diag}(B) = 0 \tag{3}$$

[1] Note, however, that the proposed model does not follow the typical architecture of autoencoders, being comprised of an encoder and a decoder: one may introduce an implicit hidden layer, however, by a (full-rank) decomposition of the learned weight-matrix of this model.

[2] Non-binary matrices may also be used.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313710

Several comments are in order:

- We choose the square loss ($||\cdot||_F$ denotes the Frobenius norm) between the data X and the predicted scores $S = XB$ over other loss functions because it allows for a closed-form solution (see next section). Training with other loss functions, however, might result in improved ranking-accuracy: in [17], it was observed that the multinomial likelihood resulted in better ranking-accuracy than training with the logistic likelihood (log loss) or the Gaussian likelihood (square loss). Directly optimizing a (surrogate) ranking loss might result in further accuracy gains–however, at possibly increased computational costs.
- We use L2-norm regularization of the weights B to be learned. The training objective hence has a single hyperparameter λ, to be optimized on a validation set.
- The constraint of a zero diagonal, $\text{diag}(B) = 0$, is crucial as to avoid the trivial solution $B = I$ (self-similarity of items), where I is the identity matrix. It was introduced in SLIM [20].

3.1 Closed-Form Solution

In this section, we show that the constrained convex optimization problem for learning the weight matrix B in Eqs. 2 and 3 can be solved in closed form.

We start by including the equality constraint in Eq. 3 into the objective function in Eq. 2 by forming the Lagrangian

$$L = ||X - XB||_F^2 + \lambda \cdot ||B||_F^2 + 2 \cdot \gamma^\top \cdot \text{diag}(B),$$

where $\gamma = (\gamma_1, ..., \gamma_{|\mathcal{I}|})^\top$ is the vector of Lagrangian multipliers. Its values will be chosen in Eq. 6 such that the constraint in Eq. 3 is fulfilled.

The constrained optimization problem in Eqs. 2 and 3 is solved by minimizing this Lagrangian. As a necessary condition, we hence set its derivative to zero, which yields the estimate $\hat{B}$ of the weight matrix after re-arranging terms:

$$\hat{B} = (X^\top X + \lambda I)^{-1} \cdot (X^\top X - \text{diagMat}(\gamma)),$$

where $\text{diagMat}(\cdot)$ denotes the diagonal matrix and I the identity matrix. Defining (for sufficiently large λ)

$$\hat{P} \triangleq (X^\top X + \lambda I)^{-1}, \tag{4}$$

this can be substituted into the previous equation:

$$\begin{aligned} \hat{B} &= (X^\top X + \lambda I)^{-1} \cdot (X^\top X - \text{diagMat}(\gamma)) \\ &= \hat{P} \cdot (\hat{P}^{-1} - \lambda I - \text{diagMat}(\gamma)) \\ &= I - \hat{P} \cdot (\lambda I + \text{diagMat}(\gamma)) \\ &= I - \hat{P} \cdot \text{diagMat}(\tilde{\gamma}) \end{aligned} \tag{5}$$

where we defined the vector $\tilde{\gamma} \triangleq \lambda \vec{1} + \gamma$ in the last line, with $\vec{1}$ denoting a vector of ones. The values of the Lagrangian multipliers γ, and hence $\tilde{\gamma}$, are determined by the constraint $\text{diag}(\hat{B}) = 0$. It follows from Eq. 5:

$$0 = \text{diag}(\hat{B}) = \vec{1} - \text{diag}(\hat{P}) \odot \tilde{\gamma} \tag{6}$$

where $\odot$ denotes the elementwise product, and hence:

$$\tilde{\gamma} = \vec{1} \oslash \text{diag}(\hat{P}),$$

where $\oslash$ denotes the elementwise division (which is well-defined given that $\hat{P}$ is invertible). Substituting this into Eq. 5 immediately results in the closed-form solution:

$$\hat{B} = I - \hat{P} \cdot \text{diagMat}\left(\vec{1} \oslash \text{diag}(\hat{P})\right). \tag{7}$$

In other words, the learned weights are given by:

$$\hat{B}_{i,j} = \begin{cases} 0 & \text{if } i = j \\ -\frac{\hat{P}_{ij}}{\hat{P}_{jj}} & \text{otherwise.} \end{cases} \tag{8}$$

This solution obviously obeys the constraint of a zero diagonal. The off-diagonal elements are determined by the matrix $\hat{P}$ (see Eq. 4), where the j^{th} column is divided by its diagonal element $\hat{P}_{jj}$. Note that $\hat{B}$ is an asymmetric matrix in general, while $\hat{P}$ is symmetric (see Eq. 4).

Eqs. 4 and 8 show that the sufficient statistics for estimating B is given by the data Gram-matrix $G \triangleq X^\top X$, which is an item-item matrix. This is a consequence of using the square loss in Eq. 2, and is helpful for estimating B from sparse data X: if X is a sparse binary matrix, then $G = X^\top X$ is a co-occurrence matrix. The uncertainty of a co-occurrence count G_{ij} is (approximately) determined by the standard deviation of the Poisson distribution, which is $\sqrt{G_{ij}}$. As long as the co-occurrence counts G_{ij} are 'sufficiently large', G and hence B can be estimated with small error. An interesting fact is that the entries of $G = X^\top X$ can be increased by two different mechanisms: (1) a denser X (due to users with increased activity), or (2) an increased number of users in X. The latter is particularly useful, as an increased sparsity of X can be compensated by an increased number of users. In other words, the problem that there possibly is only a small amount of data available for *each* user (i.e., data sparsity), does not affect the uncertainty in estimating B if the number of users in the data matrix X is sufficiently large.

3.2 Interpretation

In this section, we outline that the closed-form solution in Eq. 8 does not come as a complete surprise. To this end, let us consider the following special case throughout this section: let the training data X be an i.i.d. sample of $|\mathcal{U}|$ data points regarding a vector of $|\mathcal{I}|$ random variables $x \sim \mathcal{N}(0, \Sigma)$ that follows a Gaussian distribution with zero mean and covariance matrix $\Sigma \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$.

Then the estimate of the covariance matrix is $\hat{\Sigma} = X^\top X / |\mathcal{U}|$. If we further drop the L2-norm regularization in Eq. 4 and assume invertibility, then $\hat{P} = \hat{\Sigma}^{-1}$ is the estimate of the so-called precision (or concentration) matrix. Estimating the precision matrix from given data is a main objective in the area of graphical models (e.g., see [15]).

It is well known [5, 15] that the (univariate) *conditional distribution* of the random variable x_j given the vector of *all* the other variables, denoted by $x_{-j} \triangleq (x_k)_{k \in \mathcal{I} \setminus \{j\}}$, is a Gaussian distribution with variance $\text{var}(x_j | x_{-j}) = 1/P_{j,j}$ and mean

$$\begin{aligned} \mu_{j|-j} \triangleq \mathbb{E}[x_j | x_{-j}] &= -x_{-j} \cdot P_{-j,j} / P_{j,j} \\ &= x_{-j} \cdot B_{-j,j} \\ &= x \cdot B_{.,j} \end{aligned}$$

where the dot in the first line denotes the dot-product between the (row) vector x_{-j} and the j^{th} column of the precision matrix

$P = \Sigma^{-1}$, omitting the j^{th} element. The second line follows from Eq. 8, and the last line from $B_{jj} = 0$. Note that this is identical to the prediction rule of the EASE$^{\text{R}}$ model in Eq. 1. This shows that EASE$^{\text{R}}$ makes a principled point-prediction that user u will like item j *conditional* on the fact that the user's past interactions with *all* items are given by $X_{u,.} = x$.

A more well-known fact (e.g., see [19]) is that the absence of an edge between the random variables x_i and x_j in a Markov network corresponds to the *conditional independence* of the random variables x_i and x_j given the vector of *all* the other variables $(x_k)_{k \in \mathcal{I} \setminus \{i,j\}}$, which is also equivalent to a zero entry in the precision matrix. Note that this is different from a zero entry in the covariance matrix Σ, which signifies *marginal independence* of x_i and x_j. This shows that the precision matrix is the conceptually correct similarity-matrix to be used, rather than the covariance matrix, which (or some rescaled variant thereof) is typically used in state-of-the-art neighborhood-based approaches (see Section 4.3).

Learning the graph structure in Markov networks corresponds to learning a sparse precision matrix from data. Approaches developed in that field (e.g., see [2, 7, 8, 13, 19, 23, 26, 29]) might be useful for improved learning of a *sparse* matrix $\hat{B}$. This is beyond the scope of this short paper.

While the interpretation we outlined in this section is limited to the special case of normally distributed variables with *zero mean*, note that the derivation of Eq. 8 does not require that each column of the data matrix X has zero mean. In other words, in EASE$^{\text{R}}$ any definition of the Gram matrix $G \triangleq X^\top X$ may be used, e.g., G may be the co-occurrence matrix (if X is a binary matrix), proportional to the covariance matrix (if X is pre-processed to have zero mean in each column), or the correlation matrix (if each column of X is pre-processed to have zero mean and unit variance). After training EASE$^{\text{R}}$ on these transformed matrices X and then transforming the predicted scores back to the original space (as defined by the binary matrix X), we found in our experiments that the differences in the obtained ranking accuracies were of the order of the standard error, and we hence do not separately report these results in Section 5.

3.3 Algorithm

The Python code of the resulting learning algorithm is given in Algorithm 1. Note that the training of EASE$^{\text{R}}$ requires only the item-item matrix $G = X^\top X$ as input, instead of the user-item matrix X, and hence is particularly efficient if the size of G (i.e., $|\mathcal{I}| \times |\mathcal{I}|$) is smaller than the number of user-item-interactions in X. In this case, the expensive computation of $G = X^\top X$ can be done on a big-data pre-processing system, prior to the actual model training.

3.4 Computational Cost

The computational complexity of Algorithm 1 is determined by the matrix inversion of the data Gram-matrix $G \triangleq X^\top X \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$, which is $O(|\mathcal{I}|^3)$ when using a basic approach, and about $O(|\mathcal{I}|^{2.376})$ when using the Coppersmith-Winograd algorithm. Note that this is independent of the number of users as well as the number of user-item-interactions, as G can be computed in the pre-processing step.

This computational complexity is orders of magnitude lower than the cost of training a SLIM model and its variants [16, 20, 24]:

ALGORITHM 1: Training in Python 2 using numpy

Input: data Gram-matrix $G := X^\top X \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$,
L2-norm regularization-parameter $\lambda \in \mathbb{R}^+$.
Output: weight-matrix B with zero diagonal (see Eq. 8).

```
diagIndices = numpy.diag_indices(G.shape[0])
G[diagIndices] += λ
P = numpy.linalg.inv(G)
B = P / (-numpy.diag(P))
B[diagIndices] = 0
```

those approaches take advantage of the fact that the optimization problem regarding $|\mathcal{I}|$ items can be decomposed into $|\mathcal{I}|$ independent (and hence embarrassingly parallel) optimization problems involving $|\mathcal{I}| - 1$ items each, due to the identity $||X - XB||_F^2 = \sum_{j \in \mathcal{I}} |X_{.,j} - XB_{.,j}|_2^2$. If each of the $|\mathcal{I}|$ independent problems is solved based on an item-item matrix, the total computational cost is hence $O(|\mathcal{I}|(|\mathcal{I}| - 1)^{2.376})$. Note that the computational cost of solving those $|\mathcal{I}|$ problems is larger by almost a factor of $|\mathcal{I}|$ than training EASE$^{\text{R}}$, which requires only a *single* regression problem to be solved. In practice, however, SLIM and its variants are trained on the user-item-interactions in X, which may incur additional computational cost. This explains the vastly reduced training-times of EASE$^{\text{R}}$ observed in our experiments in Section 5.

In practice, the wall-clock time depends crucially on the fact if the number of items $|\mathcal{I}|$ is sufficiently small such that the weight matrix fits into memory, so that the matrix inversion can be computed in memory. This was the case in our experiments in Section 5.

4 RELATED WORK

EASE$^{\text{R}}$ can be viewed as an autoencoder, as a modified version of SLIM, and a neighborhood-based approach. We discuss each of the three related approaches in the following.

4.1 Deep Learning and Autoencoders

While the area of collaborative filtering has long been dominated by matrix factorization approaches, recent years have witnessed a surge in deep learning approaches [6, 10–12, 17, 25, 30, 31], spurred by their great successes in other fields. Autoencoders provide the model architecture that fits exactly the (plain-vanilla) collaborative filtering problem. While various network architectures have been explored, it was found that deep models with a large number of hidden layers typically do not obtain a notable improvement in ranking accuracy in collaborative filtering, compared to 'deep' models with only one, two or three hidden layers, e.g., [10, 17, 25, 31], which is in stark contrast to other areas, like computer vision. A combination of deep and shallow elements in a single model was proposed in [6].

In contrast, EASE$^{\text{R}}$ has no hidden layer. Instead, the self-similarity of each item in the input and output layer is constrained to zero, see also Figure 1. As a consequence, the model is forced to learn the similarity of an item in the output layer in terms of the *other* items in the input layer. The surprisingly good empirical results of EASE$^{\text{R}}$ in Section 5 suggest that this constraint might be more effective than using hidden layers with limited capacity as to force the model to generalize well to unseen data.

4.2 SLIM and Variants

While the SLIM model [20] has shown competitive empirical results in numerous papers, it is computationally expensive to train, e.g., see [17, 20] and Section 3.4. This has sparked follow-up work proposing various modifications. In [16], both constraints on the weight matrix (non-negativity and zero diagonal) were dropped, resulting in a regression problem with an elastic-net regularization. While competitive ranking results were obtained in [16], in the experiments in [17] it was found that its performance was considerably below par. The square loss in SLIM was replaced by the logistic loss in [24], which entailed that both constraints on the weight matrix could be dropped, as argued by the authors. Moreover, the L1-norm regularization was dropped, and a user-user weight-matrix was learned instead of an item-item matrix.

All these approaches take advantage of the fact that the optimization problem decomposes into independent and embarrassingly parallel problems. As discussed in the previous section, however, this is several orders of magnitudes more costly than training EASE^R if the weight matrix fits into memory.

Most importantly, while those modifications of SLIM dropped the constraint of a zero diagonal in the weight matrix, it is retained in EASE^R. In fact, we found it to be the most crucial property for achieving improved ranking accuracy (see Section 5). As we showed in Section 3.1, this constraint can be easily included into the training objective via the method of Lagrangian multipliers, allowing for a closed-form solution.

Compared to SLIM [20], we dropped the constraint of non-negative weights, which we found to greatly improve ranking accuracy in our experiments (see Table 1 and Figure 2). Moreover, we did not use L1-norm regularization for computational efficiency. We also did not find sparsity to noticeably improve ranking accuracy (see Section 5). The learned weight matrix $\hat{B}$ of EASE^R is dense.

Also note that extensions to SLIM like cSLIM [21], can be turned into an analogous extension of EASE^R.

4.3 Neighborhood-based Approaches

Numerous neighborhood-based approaches have been proposed in the literature (e.g., see [27, 28] and references therein). While model-based approaches were found to achieve better ranking accuracy on some data sets, neighborhood-based approaches dominated on others, e.g., the Million Song Data Competition on Kaggle [1, 18]. Essentially, the co-occurrence matrix (or some modified variant) is typically used as item-item or user-user similarity matrix in neighborhood-based methods. These approaches are usually heuristics, as the similarity matrix is not learned by optimizing an objective function (loss function or likelihood). More importantly, the closed-form solution derived in Eqs. 4 and 8 reveals that the *inverse* of the data Gram matrix is the conceptually correct similarity matrix,[3] see Section 3.2 for more details. This is in contrast to the typical neighborhood-based approaches, which use the data Gram-matrix without inversion. The use of the conceptually correct, inverse matrix in EASE^R may explain the improvement observed in Table 2 compared to the heuristics used by state-of-the-art neighborhood approaches.

[3]In fact, inverse matrices are used in many areas, for instance, the inverse covariance matrix in the Gaussian density function or in the Mahalanobis distance.

5 EXPERIMENTS

In this section, the proposed EASE^R model is empirically compared to several state-of-the-art approaches, based on two papers that provided publicly available code for reproducibility of results [17, 28]. Both papers together cover linear, non-linear, deep and probabilistic models, as well as neighborhood-based approaches.

5.1 Experimental Set-up

We will only summarize the experimental set-ups used in these papers, and refer the reader to these papers for details.

Summary of Set-up in [17]: This paper considers the following models:

- Sparse Linear Method (SLIM) [20]. Besides the original model, also a computationally faster approximation (which drops the constraints on the weights) [16] was considered, but its results were not found to be on par with the other models in the experiments in [17].
- Weighted Matrix Factorization (WMF) [14, 22], a linear model with a latent representation of users and items.
- Collaborative Denoising Autoencoder (CDAE) [30], a non-linear model with one hidden layer.
- denoising autoencoder (MULT-DAE) and variational autoencoder (MULT-VAE^{PR}) [17], both trained using the multinomial likelihood, which was found to outperform the Gaussian and logistic likelihoods. Best results were obtained in [17] for the MULT-VAE^{PR} and MULT-DAE models that were rather shallow 'deep models', namely with a 200-dimensional latent representation, as well as a 600-dimensional hidden layer in both the encoder and decoder. Both models are non-linear, and MULT-VAE^{PR} is also probabilistic.

Three data sets were used in the experiments in [17], and were preprocessed and filtered for items and users with a certain activity level, resulting in the following data-set sizes, see [17] for details:[4]

- MovieLens 20 Million (*ML-20M*) data [9]: 136,677 users and 20,108 movies with about 10 million interactions,
- Netflix Prize (*Netflix*) data [3]: 463,435 users and 17,769 movies with about 57 million interactions,
- Million Song Data (*MSD*) [4]: 571,355 users and 41,140 songs with about 34 million interactions.

The evaluation in [17] was conducted in terms of *strong generalization*, i.e., the training, validation and test sets are disjoint in terms of users. This is in contrast to *weak generalization*, where the training and test sets are disjoint in terms of the user-item interaction-pairs, but not in terms of users. Concerning evaluation in terms of ranking metrics, Recall@k for $k \in \{20, 50\}$ as well as Normalized Discounted Cumulative Gain, NDCG@100 were used in [17].

Summary of Set-up in [28]: Their paper focuses on neighborhood-based approaches, and the authors publicly shared code[5] regarding the experiments in their table 2 in [28], albeit only for the data-split where the training data was comprised of (at most) 30% of each user's interactions (and the remainder was assigned to the test data), which restricted our experimental comparison to this single

[4]The code regarding *ML-20M* in [17] is publicly available at `https://github.com/dawenl/vae_cf`. The authors kindly provided the code for the other two data sets upon request.

[5]`http://www.cs.toronto.edu/~mvolkovs/SVD_CF.zip`

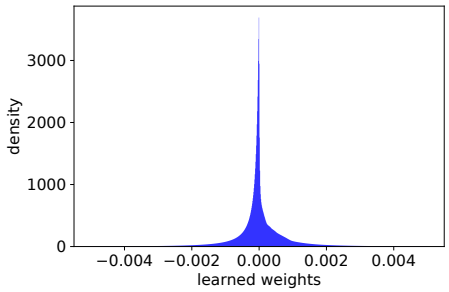


Figure 2: Histogram of the weights learned on *Netflix* data.

split in Table 2. They used the MovieLens 10 Million (*ML-10M*) data [9], which was binarized in [28] and is comprised of 69,878 users and 10,677 movies with 10 million interactions. Their evaluation was done in terms of *weak generalization*, and NDCG@10 was used as ranking metric for evaluation in [28].

5.2 Results

Despite the simplicity of EASE^R, we observed that EASE^R obtained considerably better ranking accuracy than any of the competing models on most of the data sets. This remarkable empirical result is discussed in detail in the following.

Comparison to SLIM: Table 1 shows that EASE^R achieved notably increased accuracy compared to SLIM on all the data sets. This suggests that dropping the L1-norm regularization as well as the non-negativity constraint on the learned weights is beneficial. Our analysis indicates that the latter is especially important: as illustrated in Figure 2 on the *Netflix* data (the histograms for *ML-20M* and *MSD* data look almost identical up to re-scaling, and are omitted), the learned weights in EASE^R are distributed around 0. Interestingly, it turns out that about 60% of the learned weights are negative on all the data sets in our experiments (regarding both papers [17, 28]). This indicates that it is crucial to learn also the dissimilarity (negative weights) between items besides their similarity (positive weights). Moreover, when we simply set the negative weights to zero (see EASE$^R\geq 0$ in Table 1), which obviously is not the optimal non-negative solution, the resulting accuracy drops and is very close to the one of SLIM. Apart from that, note that EASE$^R\geq 0$ is still quite dense (40% positive weights) compared to SLIM, which indirectly indicates that the sparsity of SLIM (due to L_1-norm regularization) did not noticeably improve the ranking accuracy of SLIM in our experiments.

Regarding regularization, the optimal L2-norm regularization parameter (λ) for EASE^R is about 500 on *ML-20M*, 1,000 on *Netflix*, and 200 on *MSD*. These values are much larger than the typical values used for SLIM, which often are of the order of 1, see [20]. Note that SLIM additionally uses L1-norm regularization, and hence has much fewer (non-zero) parameters than EASE^R.

As expected based on Section 3.4, we also found the (wall-clock) training-time of EASE^R to be smaller by several orders of magnitude compared to SLIM : [17] reports that parallelized grid search for SLIM took about two weeks on the *Netflix* data, and the *MSD* data was 'too large for it to finish in a reasonable amount of time' [17]. In contrast,

Table 1: Ranking accuracy (with standard errors of about 0.002, 0.001, and 0.001 on the *ML-20M*, *Netflix*, and *MSD* data, respectively), following the experimental set-up in [17].

(a) *ML-20M*	Recall@20	Recall@50	NDCG@100
popularity	0.162	0.235	0.191
EASE^R	0.391	0.521	0.420
EASE$^R\geq 0$	0.373	0.499	0.402
results reproduced from [17]:			
SLIM	0.370	0.495	0.401
WMF	0.360	0.498	0.386
CDAE	0.391	0.523	0.418
MULT-VAE^{PR}	0.395	0.537	0.426
MULT-DAE	0.387	0.524	0.419
(b) *Netflix*			
popularity	0.116	0.175	0.159
EASE^R	0.362	0.445	0.393
EASE$^R\geq 0$	0.345	0.424	0.373
results reproduced from [17]:			
SLIM	0.347	0.428	0.379
WMF	0.316	0.404	0.351
CDAE	0.343	0.428	0.376
MULT-VAE^{PR}	0.351	0.444	0.386
MULT-DAE	0.344	0.438	0.380
(c) *MSD*			
popularity	0.043	0.068	0.058
EASE^R	0.333	0.428	0.389
EASE$^R\geq 0$	0.324	0.418	0.379
results reproduced from [17]:			
SLIM	— did not finish in [17] —		
WMF	0.211	0.312	0.257
CDAE	0.188	0.283	0.237
MULT-VAE^{PR}	0.266	0.364	0.316
MULT-DAE	0.266	0.363	0.313

training EASE^R on the *Netflix* data took less than two minutes, and on the *MSD* data less than 20 minutes on an AWS instance with 64 GB RAM and 16 vCPUs in our experiments. These times have to be multiplied by the number of different hyperparameter-values to be grid-searched. Note, however, that EASE^R only has a single hyperparameter (regarding L2-norm regularization), while SLIM has two hyperparameters (concerning L1 and L2 norms) to be jointly optimized.

Comparison to linear and deep non-linear models in [17]: Table 1 shows that EASE^R was consistently outperformed on only the *ML-20M* data, and only by a small margin by the best competing model (MULT-VAE^{PR}). On the *Netflix* and *MSD* data, EASE^R obtained significantly better ranking results than any of the competing linear, non-linear, deep or probabilistic models evaluated in [17]. On the *MSD* data, EASE^R even improved over the best competing model by 25%, 17% and 23% regarding Recall@20, Recall@50, and NDCG@100, respectively. This is consistent with the results of the Million Song Data Challenge on Kaggle [18], where neighborhood-based approaches were found to vastly outperform model-based

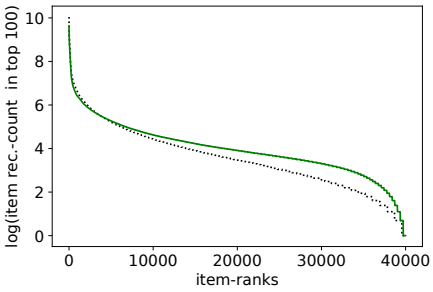


Figure 3: EASE^R (green) recommends long-tail items more often in the top-100, compared to MULT-VAE ^{PR} (dotted), on *MSD* data.

approaches [1]. As discussed in Section 4.3, EASE^R may also be viewed as a principled neighborhood approach.

As to explain EASE^R's relative improvements from *ML-20M* via *Netflix* to *MSD* data, various properties of the data sets may be considered. As shown by table 1 in [17], the number of user-item interactions, and the sparsity of the data sets do not appear well correlated with EASE^R's relative performance in Table 1. Only the number of users correlates well with the improvements of EASE^R over the competing models, which, however, appears to be spurious.

The explanation can be understood in terms of the tradeoff between recommending generally popular items vs. personally relevant items to each user, which is supported by two empirical findings: (1) we evaluated the popularity model in Table 1 as an additional baseline, where the items are ranked by their popularities (i.e., the number of users who interacted with an item). These unpersonalized recommendations obviously ignore the personalized relevance to a user. Table 1 shows that this popularity model obtains better accuracy on the *ML-20M* data than it does on the *Netflix* data, while its accuracy is considerably reduced on the *MSD* data. This suggests that good recommendations on the *MSD* data have to focus much more on personally relevant items rather than on generally popular items, compared to the *ML-20M* and *Netflix* data. (2) When counting how often an item was recommended in the top-100 across all test-users, and then ranking the items by their counts, we obtained Figure 3 for the *MSD* data: it shows that EASE^R recommended long-tail items more often than MULT-VAE ^{PR} did. In contrast, there was almost no difference between the two approaches on either of the data sets *ML-20M* and *Netflix* (figures omitted due to page limit).

The notable improvement of EASE^R over the other models on the *MSD* data suggests that it is able to better recommend personally relevant items on this data set. On the other hand, EASE^R's results on the *ML-20M* and *Netflix* data suggest that it is also able to make recommendations with an increased focus on popular items. We suspect that EASE^R's large number of parameters, combined with its constraint regarding self-similarity of items, provides it with sufficient flexibility to adapt to the various data sets. In contrast, the model architectures based on hidden layers with limited capacity seem to be unable to adapt well to the increased degree of personalized relevance in the *MSD* data.

Table 2: Comparison to the neighborhood-approaches in [28]: EASE^R considerably improves over 'ii-SVD-500' [28].

	EASE^R	EASE^R ≥ 0	reproduced from [28]: ii-SVD-500	item-item	WMF
NDCG@10	0.6258	0.6199	0.6113	0.5957	0.5969

Comparison to neighborhood-based approaches in [28]: Considerable improvements were obtained in [28] by first predicting the scores for all user-item interactions with a neighborhood-based approach ('item-item' in Table 2) that was followed by a low-rank singular value decomposition of the predicted user-item score-matrix ('ii-SVD-500' in Table 2): an increase in NDCG@10 by 0.0156 and 0.0144 compared to the baseline models 'item-item' and WMF, respectively, as reproduced in Table 2. In comparison, EASE^R obtained increases of 0.0301 and 0.0289 over the baseline models 'item-item' and WMF, respectively, see Table 2. This is about twice as large an improvement as was obtained by the approach 'ii-SVD-500' proposed in [28].

Given the small size of this training data-set, a large L2-norm regularization was required ($\lambda = 3000$) for EASE^R. Like in the previous experiments, also here about 60% of the learned weights were negative in EASE^R, and setting them to zero (EASE$^R \geq 0$ in Table 2) resulted in a small decrease in NDCG@10, as expected. In terms of wall-clock time, we found that training EASE^R was about three times faster than computing merely the factorization step in the ii-SVD-500 approach.

6 CONCLUSIONS

We presented a simple yet effective linear model for collaborative filtering, which combines the strengths of autoencoders and neighborhood-based approaches. Besides enabling efficient training (with savings of up to several orders of magnitude if the model fits into memory), the derived closed-form solution also shows that the conceptually correct similarity-matrix to be used in neighborhood approaches is based on the *inverse* of the given data Gram-matrix. In contrast, state-of-the-art neighborhood approaches typically use the data Gram-matrix directly. In our experiments, we found that allowing the learned weights to also take on negative values (and hence learn dissimilarities between items, besides similarities), was essential for the obtained ranking accuracy. Interestingly, the achieved ranking accuracy in our experiments was on par or even (notably) better than those of various state-of-the-art approaches, including deep non-linear models as well as neighborhood-based approaches. This suggests that models where the self-similarity of items is constrained to zero may be more effective on sparse data than model architectures based on hidden layers with limited capacity. We presented a basic version as to illustrate the essence of the idea, and leave various modifications and extensions for future work.

ACKNOWLEDGMENTS

I am very grateful to Tony Jebara and Nikos Vlassis for useful comments on an earlier draft, and especially to Dawen Liang for suggestions on related work with publicly available code.

REFERENCES

[1] F. Aiolli. 2013. Efficient top-N Recommendation for very large scale binary rated Datasets. In *ACM Conference on Recommender Systems (RecSys)*.

[2] O. Banerjee, L.E. Ghaoui, and A. d'Aspremont. 2008. Model Selection Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data. *Journal of Machine Learning Research* 9 (2008).

[3] J. Bennet and S. Lanning. 2007. The Netflix Prize. In *Workshop at SIGKDD-07, ACM Conference on Knowledge Discovery and Data Mining*.

[4] T. Bertin-Mahieux, D.P.W. Ellis, B. Whitman, and P. Lamere. 2011. The Million Song Dataset. In *International Society for Music Information Retrieval Conference (ISMIR)*.

[5] J. Besag. 1975. Statistical analysis of non-lattice data. *The Statistician* 24 (1975), 179–95. Issue 3.

[6] H.T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems (DLRS)*. 7–10.

[7] J. Friedman, T. Hastie, and R. Tibshirani. 2008. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* 9 (2008). Issue 3.

[8] D. Guillot, B. Rajaratnam, B. Rolfs, A. Maleki, and I. Wong. 2012. Iterative Thresholding Algorithm for Sparse Inverse Covariance Estimation. In *Advances in Neural Information Processing Systems (NIPS)*.

[9] F. M. Harper and J. A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 5 (2015). Issue 4.

[10] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. 2017. Neural Collaborative Filtering. In *International World Wide Web Conference (WWW)*.

[11] B. Hidasi and A. Karatzoglou. 2017. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In *International Conference on Information and Knowledge Management (CIKM)*. arXiv:1706.03847.

[12] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. 2015. Session-based Recommendations with Recurrent Neural Networks. arXiv:1511.06939.

[13] C.-J. Hsieh, M.A. Sustik, I.S. Dhillon, P.K. Ravikumar, and R. Poldrack. 2013. Sparse Inverse Covariance Estimation for a Million Variables. In *Advances in Neural Information Processing Systems (NIPS)*.

[14] Y. Hu, Y. Koren, and C. Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *IEEE International Conference on Data Mining (ICDM)*.

[15] S. L. Lauritzen. 1996. *Graphical Models*. Oxford University Press.

[16] M. Levy and K. Jack. 2013. Efficient Top-N Recommendation by Linear Regression. In *RecSys Large Scale Recommender Systems Workshop*.

[17] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *International World Wide Web Conference (WWW)*.

[18] B. McFee, T. Bertin-Mahieux, D. P. Ellis, and G. R. Lanckriet. 2012. The Million Song Dataset Challenge. In *International World Wide Web Conference (WWW)*. `http://www.kaggle.com/c/msdchallenge`.

[19] N. Meinshausen and P. Bühlmann. 2006. High-dimensional graphs and variable selection with the Lasso. *Annals of Statistics* 34 (2006). Issue 3.

[20] X. Ning and G. Karypis. 2011. SLIM: Sparse Linear Methods for Top-N Recommender Systems. In *IEEE International Conference on Data Mining (ICDM)*. 497–506.

[21] X. Ning and G. Karypis. 2012. Sparse Linear Methods with Side Information for Top-N Recommendations. In *ACM Conference on Recommender Systems (RecSys)*.

[22] R. Pan, Y. Zhou, B. Cao, N. Liu, R. Lukose, M. Scholz, and Q. Yang. 2008. One-Class Collaborative Filtering. In *IEEE International Conference on Data Mining (ICDM)*.

[23] M. Schmidt. 2011. *Graphical Model Structure Learning with L1-Regularization*. Ph.D. Dissertation. University of British Columbia, Vancouver, Canada.

[24] S. Sedhain, A. K. Menon, S. Sanner, and D. Braziunas. 2016. On the Effectiveness of Linear Models for One-Class Collaborative Filtering. *AAAI* (2016).

[25] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie. 2015. Autorec: Autoencoders meet Collaborative Filtering. In *International World Wide Web Conference (WWW)*.

[26] E. Treister and J.S. Turek. 2014. A Block-Coordinate Descent Approach for Large-scale Sparse Inverse Covariance Estimation. In *Advances in Neural Information Processing Systems (NIPS)*.

[27] K. Verstrepen and B. Goethals. 2014. Unifying Nearest Neighbors Collaborative Filtering. In *ACM Conference on Recommender Systems (RecSys)*.

[28] M. N. Volkovs and G. W. Yu. 2015. Effective Latent Models for Binary Feedback in Recommender Systems. In *ACM Conference on Research and Development in Information Retrieval (SIGIR)*.

[29] A. Wiesel and A.O. Hero. 2012. Distributed Covariance Estimation in Gaussian Graphical Models. *IEEE Transactions on Signal Processing* 60 (2012). Issue 1.

[30] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester. 2016. Collaborative Denoising Auto-Encoders for top-N Recommender Systems. In *ACM Conference on Web Search and Data Mining (WSDM)*.

[31] Y. Zheng, B. Tang, W. Ding, and H. Zhou. 2016. A Neural Autoregressive Approach to Collaborative Filtering. In *International Conference on Machine Learning (ICML)*.

Rock, Rap, or Reggaeton?: Assessing Mexican Immigrants' Cultural Assimilation Using Facebook Data

Ian Stewart
Georgia Institute of Technology
istewart6@gatech.edu

René D. Flores
University of Chicago
renedf@uchicago.edu

Timothy Riffe
Max Planck Institute for Demographic Research
riffe@demogr.mpg.de

Ingmar Weber
Qatar Computing Research Institute
iweber@hbku.edu.qa

Emilio Zagheni
Max Planck Institute for Demographic Research
zagheni@demogr.mpg.de

ABSTRACT

The degree to which Mexican immigrants in the U.S. are assimilating culturally has been widely debated. To examine this question, we focus on musical taste, a key symbolic resource that signals the social positions of individuals. We adapt an assimilation metric from earlier work to analyze self-reported musical interests among immigrants in Facebook. We use the relative levels of interest in musical genres, where a similarity to the host population in musical preferences is treated as evidence of cultural assimilation. Contrary to skeptics of Mexican assimilation, we find significant cultural convergence even among first-generation immigrants, which problematizes their use as assimilative "benchmarks" in the literature. Further, 2nd generation Mexican Americans show high cultural convergence vis-à-vis both Anglos and African-Americans, with the exception of those who speak Spanish. Rather than conforming to a single assimilation path, our findings reveal how Mexican immigrants defy simple unilinear theoretical expectations and illuminate their uniquely heterogeneous character.

CCS CONCEPTS

• **Human-centered computing** → *Empirical studies in collaborative and social computing*; • **Information systems** → *Online advertising*; • **Applied computing** → **Sociology**.

KEYWORDS

immigration, demography, social networks, online advertising

ACM Reference Format:
Ian Stewart, René D. Flores, Timothy Riffe, Ingmar Weber, and Emilio Zagheni. 2019. Rock, Rap, or Reggaeton?: Assessing Mexican Immigrants' Cultural Assimilation Using Facebook Data. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313409

*Study completed while first author was an intern at MPIDR..

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313409

1 INTRODUCTION

Are immigrants assimilating to U.S. culture? According to the "straight-line" assimilation hypothesis, developed by 20^{th}-century Chicago sociologists, the more time immigrants spend in the host society, the more culturally similar to natives they become [17, 25]. Early European immigrants seemingly conformed to this expectation. By the third generation, most European immigrant groups became largely culturally indistinguishable from Anglo-American natives [29]. Nevertheless, post-1965 immigrants from Latin America and Asia do not appear to be following a single, "linear" assimilative path. Instead, their assimilation appears "segmented" [30, 34]. While advantaged immigrants are still assimilating "linearly," disadvantaged groups are taking a less expected route. As they concentrate in poor neighborhoods, their children come in close contact with "downtrodden" native minorities like African-Americans and often adopt their cultural norms and styles [18].

What path are Mexican immigrants taking? Critics have questioned their willingness and/or capacity to assimilate to mainstream U.S. culture. Empirical evidence of their assimilation has been mixed [1, 23]. We focus on a key dimension of culture, musical taste, to assess the assimilation trajectories of Mexican Americans. Patterns of artistic consumption like musical taste can illuminate important aspects of the assimilation process [24]. Musical tastes both reflect and reproduce ethnic divides [27]. Hence, shrinking differences in musical tastes across generations would imply that cultural assimilation is taking place and that ethnic boundaries between groups are blurring.

We analyze the musical preferences of Mexican immigrants, Mexican Americans, non-Hispanic Whites (henceforth "Anglos"), and African-Americans on Facebook.[1] We find that 1st generation Mexican immigrants have relatively high levels of cultural assimilation. Further, rather than converging with the musical tastes of *either* Anglos or African-Americans, the 2nd generation Mexican-Americans have substantial cultural similarities with *both* groups, which highlights the manifold character of Mexican assimilation. Finally, assimilation varies significantly based on demographics including age, language preference, and education level.[2]

[1] All code for data collection and analysis available here: https://github.molgen.mpg.de/istewart/immigrant_assimilation.
[2] Supplementary material for data collection and analysis available here: https://github.molgen.mpg.de/istewart/immigrant_assimilation/blob/master/writings/immigrant_assimilation_supplement.pdf.

2 RELATED WORK

Sociologists of culture have identified musical tastes as key symbolic resources individuals draw on to negotiate and express their social positions [31]. According to Bourdieu [5], taste "classifies." It marks identity and social status. While taste for high-status musical genres like classical music or opera may signal a privileged class background, lower-status genres suggest a lower class position [8]. Cultural differences like musical taste may be the building blocks that reinforce and preserve ethnic boundaries [3, 32]. According to Telles and Ortiz, "the extent to which such cultural markers [of ethnicity] persist over generations may signify the extent to which ethnic groups remain viable."

However, few studies have examined the musical taste of Mexican immigrants. Using inter-generational survey data, Telles and Ortiz find that Mexican Americans express a taste for U.S. Latino music as well as for what they term "American, black" genres. Thomas [24] also uses survey data and finds that Mexican immigrants in the U.S. exhibit a wide array of differences in musical taste vis-à-vis Anglos, but these differences diminish among the U.S. born second-generation (particularly after adjusting for class). Though insightful, this nascent literature has limitations. Some of these studies only capture a few locales rather than national trends [23]. Further, first-generation immigrants are normally used as an "assimilative benchmark" to which later generation immigrants are compared [10], which ignores the fact that first-generation immigrants may themselves acculturate. Last, prior studies rely on immigrants' self reports of cultural taste on surveys, which may not capture their everyday preferences.

By relying on Facebook data, we are able to address these limitations. First, Facebook provides rich, nation-wide data on the expressed cultural preferences of its users, which allow researchers to obtain a broader sample than what is available through surveys. Second, Facebook's worldwide popularity allows us to compare Mexican immigrants' cultural preferences directly with those in the source-country: Mexicans living in Mexico. Third, on Facebook, immigrants spontaneously report the music styles that they prefer, which advertisers can then access to target specific subgroups. These data provide a more natural, bottom-up window on immigrant interests than a survey. While much of Facebook's data is private, researchers can use the advertising platform to obtain aggregate estimates for customized demographics, which can then be validated with ground-truth data [2, 19]. Using the advertising platform, Dubois et al. [9] proposed a method to measure immigrant assimilation via cultural interests on Facebook, which we apply to examine musical interests of Mexican immigrants.

3 DATA COLLECTION

3.1 Audience data

Facebook's advertising platform allows users to compute an estimated audience size for a proposed advertisement [14]. The audience can be defined by demographic attributes provided by Facebook, including gender, age, home location, ethnic affinity, and interests. Some attributes such as gender are self-reported, while others such as ethnic affinity (e.g. "Hispanic (US - All)") are inferred. Facebook users can report their preferences over multiple domains such as music, food, and sports [9]. In this paper we focus on musical genres as a marker of assimilation [24].[1]

3.2 Identification of second-generation Mexican immigrants

Sociologists typically assess immigrant assimilation by comparing the traits and life outcomes of immigrants across generations. The expectation is that the U.S. born children of first-generation immigrants will be significantly more acculturated [12] [24]. In this study, the key demographic dimension of interest is generational status. Unfortunately, Facebook's advertising platform does not permit this kind of targeting. We therefore use different Facebook user targets as proxies for the second or later Mexican-American generation:

(1) Hispanic American non-expats with interest in Mexico, i.e. "Mexican Americans (Mexico)";
(2) Hispanic Spanish-speaking American non-expats, i.e. "Mexican Americans (Spanish)"; and
(3) Hispanic Americans non-expat located in cities with a high concentration of Mexican-Americans, according to 2010 American Communities Survey, i.e. "Mexican Americans (communities)."

These methods are validated by comparing the estimated distribution of the population across different demographics (age, education, gender, marital status, and region) against known demographic distributions for second-generation Mexican Americans [11]. The sample mean and standard deviation for each demographic distribution are computed from the 2016 Public Use Microdata Sample (PUMS), following the sampling procedure outlined by Torrieri [26]. We compute the proportion of the population that falls into a given category $c \in C$ within each demographic group d as follows:

$$demo_{c,d} = \frac{count(c,d)}{\sum_{c' \in C} count(c',d)} \quad (1)$$

The demographic proportions are computed over the Mexican Americans in the PUMS data, and the three proxy populations described above (Mexico, Spanish, communities).

The demographic distributions in Figure 1 of the supplementary material show that most of the estimates are close to the survey estimates. The distributions of marital status, age, and education are slightly different from the ground-truth distributions, due to underlying differences in the Facebook user base as compared to the overall population [20]. This bias cannot be ignored, and we therefore caution readers to view our analysis as exploratory rather than definitive. Nonetheless, the overall distributions align well: the Kullback-Leibler divergence values [15] between the estimated and ground-truth populations are significantly lower than the values expected by chance ($KL = 0.240, 0.218, 0.236$ for Spanish-speaking, interest and communities as compared to $KL = 0.858, 0.936, 1.29$ by chance based on $N = 1000$ random samples from uniform distribution). We use all three audience estimates as proxies for 2+ generation Mexican Americans in the rest of the study.

3.2.1 Population statistics. We are interested in comparing the rate of assimilation among first and later generations of Mexican immigrants, with respect to the majority racial group (Anglos) and a minority racial group (African Americans). Our study focuses on

Population name	Facebook estimate	Population estimate
Anglos	112,000,000 (59%)	189,591,066
African Americans	29,000,000 (78%)	37,014,326
U.S. Hispanics	14,000,000 (57%)	24,695,129
Mexican-Americans (communities)	4,100,000 (18%)	23,407,709
Mexican-Americans (Spanish)	1,700,000 (9%)	19,380,816
Mexican-Americans (Mexico)	2,900,000	n/a
Mexican immigrants	9,400,000 (82%)	11,508,371
Mexicans in Mexico	62,000,000 (56%)	111,375,417

Table 1: Summary of all populations relevant to the study.

five main populations: (i) non-Hispanic white Americans ("Anglos"), (ii) African Americans, (iii) Mexican immigrants to the United States, (iv) Mexican-Americans, and (v) Mexican people living in Mexico. As mentioned above, we define the Mexican-American population in three different ways: interest in Mexico, speaking Spanish, and living in a predominantly Mexican American community. Note that the populations are not mutually exclusive: for instance, someone may have both an interest in Mexico and may live in a Mexican-American neighborhood. These ethnic groups are targeted with Facebook's "ethnic affinities" demographics, which has been shown to be reasonably accurate in prior work [21].

We provide a summary of all populations of interest in the study in Table 1. As expected, the Facebook estimates are generally smaller than the survey estimates in part because of incomplete penetration [20]. In addition to independent counts, it is worth noting that the Mexican American populations show relatively low overlap: 36.2% between what we define 'communities' and 'interest'; 27.2% between 'interest' and 'Spanish'; 32.2% between 'Spanish' and 'communities'; and 11.7% between all three. This provides further justification for treating the Mexican American populations separately throughout the rest of the study.

3.3 Collect musical interests

The study focuses on music as a marker of cultural identity, which requires identifying specific musical interests to use in data collection. An important advantage of using Facebook as data source is that the interests are defined by users, which provides a bottom-up perspective on cultural interests that is often missed by other data sources such as surveys. It is important to note that because the interests are user-selected, the cultural preferences expressed may be seen as more of an identity "performance" [6] rather than true preference.

We began with a initial set of Facebook musical genres collected in 2016 [4], expanded the list manually for Latin American-specific genres from Wikipedia, expanded the list with suggestions from Facebook's advertising platform (manually filtered for musical genre), then filtered to 741 genres with a worldwide audience size above 100,000 (smaller genres are unreliable for comparing sub-populations). We provide the complete set of genres $\mathbb{I}$ in the supplementary material.

These interests provide the basis for computing differences and similarities across sub-populations. Following the approach of Dubois et al. (2018) our analysis starts by computing *interest ratios*, which represent the proportion of a given population p that is interested in genre i:

$$I_{p,i} = \frac{count(p,i)}{\sum_{i' \in \mathbb{I}} count(p,i')} \tag{2}$$

For example, if 10 users from population p show interest in hip-hop music, 60 in rock music and 30 in rap music, then the interest ratio for hip-hop would be $I_{p,hip-hop} = \frac{10}{100} = 0.10$. This quantity measures the share of interest in a given genre. Note that there is potential for overlap between populations. The interest ratio does not account for that, as we have no way of knowing how many users like both hip-hop and rock music, so we treat them as separate populations.

3.4 Quantify assimilation

The intuition behind the measure of assimilation that we propose is that a population of migrants is considered more assimilated if their interests are similar to the interests of the population at the destination, and dissimilar from the interests of the population at the origin. If the destination population is highly interested in hip-hop music, the origin population shows little interest in hip-hop, but the expat population from origin to destination is highly interested in hip-hop, we would say that the expat population is highly assimilated to the destination population with respect to hip-hop, meaning that expats match the taste of natives for the specific genre.

To quantify the level of assimilation following Dubois et al. [9], we compute for each genre the assimilation ratio (*AR*) of ex-pat interest proportion and destination interest proportion:

$$AR_i = \frac{I_{expat,i}}{I_{dest,i}} \tag{3}$$

where $I_{expat,i}$ and $I_{dest,i}$ are the proportions of the ex-pat and destination populations that have indicated an interest in musical genre i (Equation 2). A higher *AR* score indicates that the ex-pat population is more assimilated to the source population's interest in genre i.

Before computing *AR*, we identify a subset of musical genres $\hat{\mathbb{I}}$ that are strongly identified with the destination population. The interests in genres are filtered as follows:

(1) Remove all interests $\mathbb{I}'$ that are more associated with the source population than with the destination population:
$\mathbb{I}' = \{i \in \mathbb{I}; I_{dest,i} < I_{source,i}\}; \hat{\mathbb{I}} = \mathbb{I} \setminus \mathbb{I}'$.
(2) Compute the difference Δ between destination and source interest proportions, and remove all interests $\mathbb{I}''$ for which the difference falls below the p^{th} percentile ($p = 50$ following [9]):
$\Delta = I_{dest} - I_{source}; \mathbb{I}'' = \{i \in \mathbb{I}; \Delta_i \leq P_p(\Delta)\}; \hat{\mathbb{I}} = \hat{\mathbb{I}} \setminus \mathbb{I}''$

This produces a set of interests $\hat{\mathbb{I}}$ that are distinctly more associated with the destination population than with the source population (i.e. more "American" than "Mexican"), which we use to evaluate assimilation. Note that this process is repeated for each separate destination population.

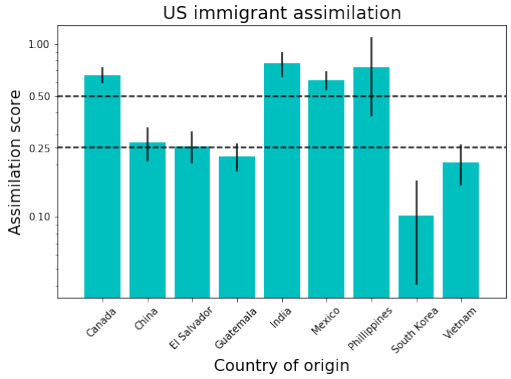


Figure 1: Median assimilation scores for selected immigrant groups. Error bars indicate 95% confidence intervals for the median.

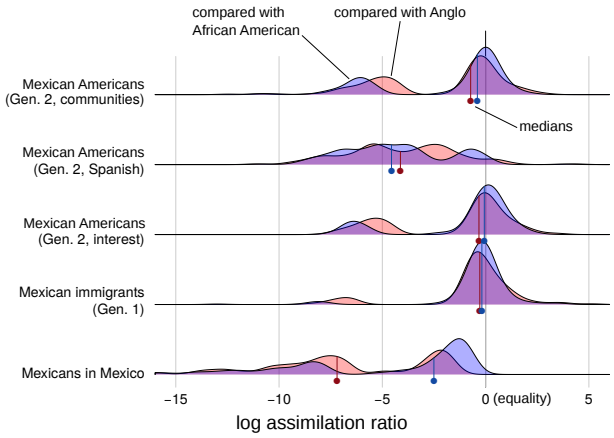


Figure 2: Assimilation scores for first- and second-generation Mexican immigrants, separated by target ethnic group (Anglo=red, African-American=blue) and log-transformed. Dotted line marks the expected assimilation value $AR = 0$, which implies an equal amount of interest for the expat and ethnic group.

4 RESULTS

4.1 Verifying cultural assimilation

In this study, we assume that the musical interests expressed on Facebook can measure cultural assimilation. To validate that assumption, we compute the assimilation ratio for several immigrant groups and validate against results from the literature [28]. We query Facebook for the audience counts for the top 100 U.S. music interests for all immigrants and natives from the following countries: Canada, China, El Salvador, Guatemala, India, Mexico, Philippines, South Korea, and Vietnam (following Vigdor [28]). We then compute all assimilation scores for each immigrant population using the top 50% of interests in the target country as compared with the source country.

The median assimilation scores and standard errors are shown in Figure 1. We see a clear divide between the high-assimilation immigrant groups (Canada, India, Mexico, and the Philippines) and the low-assimilation immigrant groups (China, El Salvador, Guatemala, South Korea, and Vietnam). This general trend mirrors the findings of Vigdor [28] (cf. Figure 6) who found that assimilation for Canadian and Filipino immigrants were higher than for immigrants from China, El Salvador, Guatemala, South Korea, and Vietnam.

While not exhaustive, this test provides some evidence that the behavior of the assimilation scores is consistent with prior work.

4.2 Comparing assimilation across ethnicity and generations

Are Mexican immigrants and their descendants becoming culturally assimilated in the U.S.? Prior work has typically measured assimilation with respect to the majority U.S. group ("Anglos"). However, some scholars have argued that Mexican Americans may be acquiring the tastes of marginalized native minorities, such as African Americans [7, 24]. Therefore, we compare the rate of assimilation across generations with respect to two reference populations: U.S.-born Anglos and African Americans.

Figure 2 shows the smoothed kernel estimate of assimilation scores between the different Mexican-origin groups and Anglos (in red) and between these groups and the musical tastes of African Americans (in blue). It shows that Mexicans in Mexico have the least similar musical tastes relative to Anglos. Also, their tastes are more similar to those of African Americans, which may be driven by the growing popularity of rap and hip-hop in Mexico. Nevertheless, such greater cultural proximity to African Americans' musical tastes cannot be found among 1st generation Mexican immigrants, who show great convergence with *both* Anglos and Blacks. This suggests that Mexican immigrants are a select group and/or that 1st generation immigrants have already acquired U.S. musical tastes during their time in the U.S.

The 2nd generation results are mixed. When we measure the Mexican-American 2nd generation by their interests ("interest") or by their residence ("communities"), we find a high degree of convergence with the musical preferences of *both* Anglos and Blacks, which calls into question unilinear expectations of cultural adaptation. In contrast, when we define the 2nd generation based on the ability to speak Spanish, we find greater cultural differences vis-à-vis both Anglos and Blacks perhaps because Spanish knowledge is a marker of "thick" ethnicity. Further, individuals in this group who are not of Mexican origin may be different from their Mexican counterparts.

These results call into question the use of 1st generation immigrants as an "assimilation benchmark" since these immigrants already show substantial cultural assimilation. They also show the sensitivity of assimilation measures to using different definitions of the 2nd generation in Facebook.

4.3 Comparing assimilation rates across demographics

Our next analysis investigates patterns of assimilation across different immigrant sub-populations. We compute the assimilation

ratios for the following demographic sub-populations of the Hispanic Mexican ex-pats living in the U.S.: age, education, gender[3], dominant language, and geographic region. These variables are known to modulate the distribution of musical taste among immigrants [24] and immigrant assimilation in general [22]. With respect to Equation 3, we set the destination population (for I_{dest}) to U.S.-born Anglo people and the source population (for I_{source}) to Mexico-born people.

The distributions in Figure 3 show substantial differentiation among subpopulations (top plot). There are significant differences for the ages and language demographics (according to Kruskal test, test statistic=21.6, ($p < 0.001$) for age, ($17.6, p < 0.001$) for language):

(1) Younger (13-18) immigrants and Mexican Americans have lower assimilation scores than the older age groups (all groups 29+).
(2) English-speaking immigrants and Mexican Americans have higher assimilation scores than Spanish-speaking and Bilingual groups.

The finding about language is more intuitive: English-speaking Facebook users are more likely to understand typical American music (mostly in English). The age finding is unusual, considering that younger immigrants are more likely to be willing to change their musical interests to align with their peers. However, analyzing the top-k interests for the youngest age group reveals why they show greater cultural dissimilarity: the top interests for 13-18 Mexican immigrants are highly concentrated under hip-hop and rap genres ("Southern hip hop", "Midwest hip hop", "Gangsta rap") and tend to miss the majority of typical Anglo interests ("Alternative rock", "Progressive rock", "Rock and roll").

In addition, subgroups of Mexican Americans, especially those defined by educational level, show notable differentiation: Mexican Americans with a Bachelor's degree or more tend to have higher assimilation scores than those with less than a high school degree. Musical taste is not necessarily inaccessible to those with a lower education, so this finding is unanticipated. In addition, female Mexican Americans have a slight tendency toward higher assimilation than male Mexican Americans, which may be related to different gender socialization among Mexican Americans versus Mexican immigrants.

4.3.1 Regression analysis. It is important to note that the prior demographic patterns may be complicated by cross-variable correlations. For instance, younger Hispanic Facebook users tend to speak more English than older Hispanic Facebook users. We therefore recompute the assimilation ratios across all cross-sections of the above demographics for all immigrant groups.[4] We run a linear regression to predict the level of assimilation from the demographic variables:

$$AR_i = \beta^T D_i + \epsilon \tag{4}$$

[3]Facebook only allows audience estimates for binary gender, so we restrict our analysis to male versus female while acknowledging that gender is a spectrum.
[4]The set of interests $\mathbb{I}$ is restricted to the top 20 music interests to reduce the number of queries.

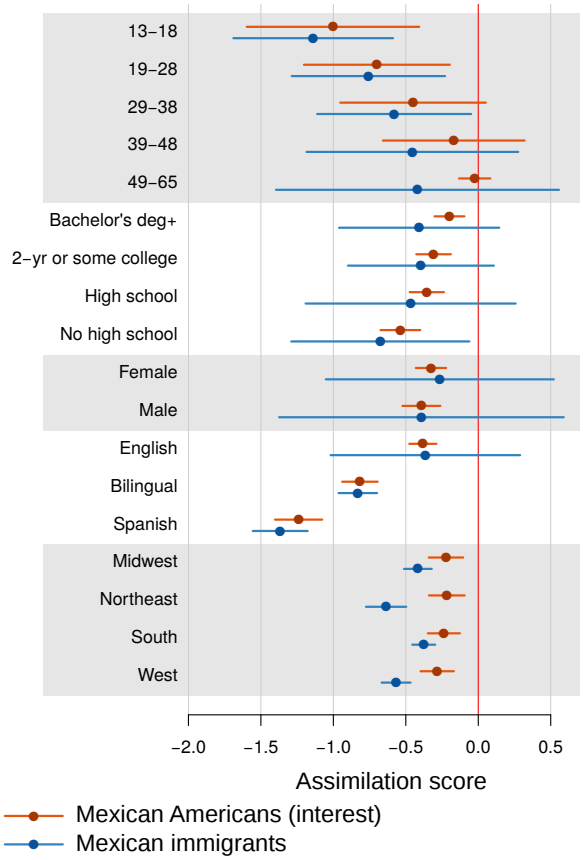


Figure 3: Median assimilation scores for Mexican immigrants and Mexican Americans (interest; i.e. all Hispanic Americans with a declared interest in Mexico), compared to Anglo interests and subdivided by demographic. The x axis shows log assimilation ratios.

where β is a vector of coefficients for all possible demographic values (male, female, etc.) and D_i is a binary vector to indicate the relevant demographic values for group i (male=1, female=0, etc.). We treat all demographics as categorical even though some may be ordinal (e.g. age, education) to account for possible nonlinear effects. We run this regression twice, using Anglo and African American Facebook users as the target populations.

The regression results for the Anglo assimilation scores are shown in Table 2. Some of the earlier results about demographic trends are corroborated here: younger, less educated and English-speaking populations have higher assimilation scores. We confirm the gender effect outlined earlier, which lends support to the notion of women as leaders of change rather than being culturally conservative. Lastly, another surprising finding shows a negative correlation with generation: later generations are expected to assimilate more, but after controlling for other variables, all definitions of the later generation have a negative correlation with assimilation. This may be due to different distributions of demographics across generations, such as the tendency of the later generation to be younger, which are controlled for in the following regression.

	$\beta(S.E.)$	p
Intercept	-0.791 (0.054)	***
Gender		
Male	-0.272 (0.027)	***
Education		
High school graduate	-0.001 (0.037)	
Less than high school graduate	-0.381 (0.038)	***
Two-year degree, Some college	-0.190 (0.038)	***
Age		
19-28	1.168 (0.042)	***
29-38	0.996 (0.043)	***
39-48	0.878 (0.043)	***
49-65	0.730 (0.043)	***
Language		
English	0.464 (0.033)	***
Spanish	-1.045 (0.035)	***
Region		
Northeast	-0.336 (0.042)	***
South	0.506 (0.036)	***
West	0.367 (0.037)	***
Gen. 2+ population		
Mexican American (Spanish)	-1.547 (0.052)	***
Mexican American (communities)	-1.203 (0.036)	***
Mexican American (interest)	-0.922 (0.033)	***

Table 2: Regression on log assimilation score with Anglo Americans as the destination population. $N = 20,145, R^2 = 0.217, F = 399.6 (p < 0.001)$**.** $^{***}\ p < 0.001$**, otherwise** $p > 0.05$**.**

The regression results for the African American assimilation scores are shown in the supplementary material. The general patterns remain the same as in the Anglo regression, albeit with some weakened effects ("West" and "female" have lower coefficients).

To test interactions among variables, we re-run the Anglo assimilation score regressions using three separate models:

(1) Model 1: interaction between age and education
(2) Model 2: interaction between age and language use
(3) Model 3: interaction between age and generation

We omit the table for space (see supplementary material) but summarize the main effects as follows. In Model 1, higher education correlates with lower assimilation but the effect is less strong for younger ("19-28") and somewhat older ("39-48") populations. In Model 2, older age has no effect on assimilation by itself but older populations that are English-speaking or Bilingual ("39-48 * English") tend to have higher assimilation scores. In Model 3, the later generation still tends to have lower assimilation, but younger members of the later generation ("19-28 * Gen. 2") have higher assimilation.

Across all analyses, we find that several high-level patterns are consistent for Mexican immigrants and Mexican Americans. We also find that there are important sources of heterogeneity such as different language effects and interactions between age and other demographics.

4.4 Summary of findings

(1) The musical tastes of Mexicans in Mexico are more similar to African Americans' tastes than to Anglos'.
(2) 1st generation Mexican immigrants exhibit high assimilation to Anglos and African Americans.
(3) 2nd generation Mexican Americans also exhibit significant assimilation to both Anglos and African Americans with the exception of those that predominantly use Spanish as their main language.
(4) In aggregate, higher-educated, older, and English-speaking populations show more assimilation to Anglo musical tastes. When controlling for cross-demographic effects, female, younger and 1st generation show more assimilation.

5 DISCUSSION

Critics have questioned the capacity and/or willingness of Mexican immigrants to assimilate culturally to the U.S. Nevertheless, we find evidence of substantial cultural assimilation even among 1st generation Mexican immigrants, which calls into question their use as a benchmark population. Further, we find that 2+ generation Mexican Americans also show high levels of cultural convergence with regards to Anglos and African Americans, with the exception of Spanish-speaking Mexican Americans. This could be evidence that Spanish maintenance is a marker of "thick" ethnicity [13].

Rather than conforming to a single assimilative path, our findings highlight how Mexicans immigrants defy simple unilinear theoretical expectations. Indeed, their remarkable convergence with both Anglo and Black musical tastes, along with the preservation of some ethnic musical genres, reflect Mexican-Americans' heterogeneous character [1]. Last, the greater similarity in musical preferences of Mexicans in Mexico vis-à-vis African Americans complicates interpreting a preference for "black" genres as a straightforward indicator of downward assimilation among immigrants.

The primary limitation of the study comes from Facebook's "black box" algorithms for inferring attributes such as ethnic affinity. Though we cannot validate the classifier outputs on a per-user level, previous work has used data on stocks of migrants [33] and voter registration data [21] to show that, by and large, the inference for country of origin and ethnic affinity are quite reliable. However, such errors are likely non-random, so we cannot rule out potential for bias. Replicating our findings with other music-related data sources, such as 'Last.fm' [16], could be worth pursuing.

Finally, it is important to consider privacy implications of our work. Whereas Cambridge Analytica and similar companies (1) use individual level data, and (2) actively try to change user behavior, we do neither in our work. We only work with anonymous, aggregate data that in terms of content is similar to census-level data.

6 ACKNOWLEDGMENTS

The authors would like to thank the researchers at the Max Planck Institute for Demographic Research for their feedback on an early version of this work in summer 2018; Matheus Araujo and Antoine Dubois for their help with accessing Facebook data; and the anonymous reviewers for their thoughtful feedback. This work received support from the guest program of the Max Planck Institute for Demographic Research.

REFERENCES

[1] Richard Alba, Tomás R Jiménez, and Helen B Marrow. 2014. Mexican Americans as a paradigm for contemporary intra-group heterogeneity. *Ethnic and Racial Studies* 37, 3 (2014), 446–466.

[2] Matheus Araujo, Yelena Mejova, Ingmar Weber, and Fabricio Benevenuto. 2017. Using Facebook Ads Audiences for Global Lifestyle Disease Surveillance: Promises and Limitations. In *Proceedings of WebSci*. 253–257.

[3] Fredrik Barth. 1969. *Ethnic Groups and Boundaries: The Social Organization of Culture Difference*. Waveland Press.

[4] Dieter Bohn and Brian Abelson. 2016. The top 2,001 things Facebook thinks we care about. https://www.theverge.com/2016/2/1/10884800/top-2001-things-interests-facebook

[5] Pierre Bourdieu. 1984. *Distinction: A Social Critique of the Judgment of Taste*.

[6] danah boyd. 2007. Why youth (heart) social network sites: The role of networked publics in teenage social life. *MacArthur foundation series on digital learning–Youth, identity, and digital media volume* 119 (2007), 142.

[7] Rogers Brubaker. 2001. The return of assimilation? Changing perspectives on immigration and its sequels in France, Germany, and the United States. *Ethnic and racial studies* 24, 4 (2001), 531–548.

[8] Bethany Bryson. 1996. "Anything but heavy metal": Symbolic exclusion and musical dislikes. *American sociological review* (1996), 884–899.

[9] Antoine Dubois, Emilio Zagheni, Kiran Garimella, and Ingmar Weber. 2018. Studying Migrant Assimilation Through Facebook Interests. In *SocInfo*.

[10] René D. Flores. 2015. The Resurgence of Race in Spain: Perceptions of Discrimination Among Immigrants. *Social Forces* 94, 1 (mar 2015), 237–269. https://doi.org/10.1093/sf/sov056

[11] Ana Gonzalez-Barrera and Mark Hugo Lopez. 2013. *Statistical Profile A Demographic Portrait of Mexican-Origin Hispanics in the United States*. Technical Report. Pew Research Center. http://www.pewhispanic.org/2013/05/01/a-demographic-portrait-of-mexican-origin-hispanics-in-the-united-states/

[12] Emily Greenman and Yu Xie. 2008. Is assimilation theory dead? The effect of assimilation on adolescent well-being. *Social Science Research* 37, 1 (2008), 109–137.

[13] Henry Hale. 2004. Explaining ethnicity. *Comparative political studies* 37, 4 (2004), 458–485.

[14] Michal Kosinski, Sandra C Matz, Samuel D Gosling, Vesselin Popov, and David Stillwell. 2015. Facebook as a research tool for the social sciences: Opportunities, challenges, ethical considerations, and practical guidelines. *American Psychologist* 70, 6 (2015), 543.

[15] Solomon Kullback and Richard Leibler. 1951. On information and sufficiency. *The Annals of Mathematical Statistics* 22, 1 (1951), 79–86.

[16] Minsu Park, Ingmar Weber, Mor Naaman, and Sarah Vieweg. 2015. Understanding Musical Diversity via Online Social Media. In *Proceedings of ICWSM*. 308–317.

[17] Robert E Park. 1928. Human migration and the marginal man. *American journal of sociology* 33, 6 (1928), 881–893.

[18] Alejandro Portes and Rubén G Rumbaut. 2001. *Legacies: The story of the immigrant second generation*. University of California Press.

[19] Francesco Rampazzo, Emilio Zagheni, Ingmar Weber, Maria Rita Testa, and Francesco Billari. 2018. Mater certa est, pater numquam: What can Facebook Advertising Data Tell Us about Male Fertility Rates?. In *Proceedings of the International Conference on Web and Social Media*. 672–675.

[20] Aaron Smith and Monica Anderson. 2018. *Social Media Use in 2018*. Technical Report March. Pew Research Center, Washington, D.C. 4 pages. www.pewresearch.org

[21] Till Speicher, Muhammad Ali, Giridhari Venkatadri, Filipe Nunes Ribeiro, George Arvanitakis, Fabrício Benevenuto, Krishna Gummadi, Patrick Loiseau, and Alan Mislove. 2018. Potential for Discrimination in Online Targeted Advertising. In *Conference on Fairness, Accountability and Transparency*, Vol. 81. 1–15.

[22] Edward Telles and Vilma Ortiz. 2016. Generations of exclusion. In *The New Latino Studies Reader: A Twenty-First Century Perspective*. University of California Press, 340—-366.

[23] Edward M Telles and Vilma Ortiz. 2008. *Generations of exclusion: Mexican-Americans, assimilation, and race*. Russell Sage Foundation.

[24] Kyla Thomas. 2017. Sounds of disadvantage: Musical taste and the origins of ethnic difference. *Poetics* 60 (2017), 29–47.

[25] William Isaac Thomas and Florian Znaniecki. 1918. *The Polish peasant in Europe and America: Monograph of an immigrant group*. Vol. 2. University of Chicago Press.

[26] Nancy Torrieri. 2014. *American Community Survey Design and Methodology*. Technical Report. United States Census Bureau, Washington, D.C. https://www2.census.gov/programs-surveys/acs/methodology/design_and_methodology/acs_design_methodology_report_2014.pdf

[27] Sandra Trienekens. 2002. 'Colourful' distinction: the role of ethnicity and ethnic orientation in cultural consumption. *Poetics* 30, 4 (aug 2002), 281–298. https://doi.org/10.1016/s0304-422x(02)00025-6

[28] Jacob Vigdor. 2011. Comparing Immigrant Assimilation in North America and Europe. *Manhattan Institute for Policy Research* (2011).

[29] William Lloyd Warner and Leo Srole. 1945. *The social systems of American ethnic groups*. Yale University Press.

[30] Mary C. Waters and Tomás R. Jiménez. 2005. Assessing Immigrant Assimilation: New Empirical and Theoretical Challenges. *Annual Review of Sociology* 31, 1 (2005), 105–125.

[31] Max Weber. 1958. *The Protestant ethic and the spirit of capitalism*.

[32] Andreas Wimmer. 2008. The making and unmaking of ethnic boundaries: A multilevel process theory. *American journal of sociology* 113, 4 (2008), 970–1022.

[33] Emilio Zagheni, Ingmar Weber, and Krishna Gummadi. 2017. Leveraging Facebook's Advertising Platform to Monitor Stocks of Migrants. *Population and Development Review* 43, 4 (2017), 721–734.

[34] Min Zhou and Carl Bankston. 1994. Social Capital and the adaptation of the second generation: the case of Vietnamese youth in New Orleans. *International Migration Review* 28, 4 (1994), 821–845.

Learning Intent to Book Metrics for Airbnb Search

Bradley C. Turnbull
Airbnb Inc.
San Francisco, California
bradley.turnbull@airbnb.com

ABSTRACT

Airbnb is a two-sided rental marketplace offering a variety of unique and more traditional accommodation options. Similar to other online marketplaces we invest in optimizing the content surfaced on the search UI and ranking relevance to improve the guest online search experience. The unique Airbnb inventory, however, surfaces some major data challenges. Given the high stakes of booking less traditional accommodations, users can spend many days to weeks searching and scanning the description page of many accommodation "listings" before making a decision to book. Moreover, much of the information about a listing is unstructured and can only be found by the user after they go through the details on the listing page. As a result, we have found traditional search metrics do not work well in the context of our platform. Basic metrics of single user actions, such as click-through-rates, number of listings viewed, or dwell time, are not consistently directionally correlated with our downstream business metrics. To address these issues we leverage machine learning to isolate signals of intent from rich behavioral data. These signals have key applications including analytical insights, ranking modeling inputs, and experimentation velocity. In this paper, we describe the development of a model-based user intent metric, "intentful listing view", which combines the signals of a variety of user micro-actions on the listing description page. We demonstrate this learned metric is directionally correlated with downstream conversion metrics and sensitive across a variety of historical search experiments.

CCS CONCEPTS

• **General and reference → Metrics**; **Experimentation**; • **Information systems → Electronic commerce**; • **Computing methodologies → Machine learning**.

KEYWORDS

User Intent Modeling, Experimentation, E-commerce Search

ACM Reference Format:
Bradley C. Turnbull. 2019. Learning Intent to Book Metrics for Airbnb Search. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313648

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313648

1 INTRODUCTION

Airbnb[1] is a two-sided rental marketplace where users can book unique as well as more traditional accommodations. The guest booking "funnel" is generally defined in three stages:

Search - A user executes searches, likely specifying a location, dates, and guest count. The user may also apply filters to their search to express specific preferences, such as price maximum and/or minimum. Available accommodation listings from our inventory are retrieved, ranked via our relevance models, and presented to the user. A user will explore the results by clicking listings to open the details page, with information such as checkin and checkout times, available amenities, sleeping arrangements, reviews, and more.

Contact - Once a user finds a listing they are interested in, they send a message to the host requesting to book their place. Note that guests can also send messages to hosts asking for clarification of listing details or special requests without sending a formal request to book.

Book - Once the host approves the request, the guest receives a notification and can then finalize the booking. Some hosts opt-in to a program, called "instant book", where booking requests are automatically accepted. It is also possible that the host will reject the booking request, at which point the guest will have to contact other listings to find accommodations.

The search team at Airbnb focuses on optimizing the guest experience from search to booking. We analyze the efficiency at which guests move through the different stages of the conversion funnel to inform product and model improvements, and measure their success in online experiments. We generally observe contacts and bookings almost simultaneously. Therefore, relying primarily on contacts and bookings signals poses two major challenges:

1. **Sparsity** - very few visitors and searchers end up contacting or booking,
2. **Signal Time Lag** - searchers can spend days to weeks searching before contacting or booking, so it can take a similarly long time for lifts or drops in booking signals to manifest within experiments.

Consequently, we are strongly motivated to develop upper-funnel metrics which are less sparse, meaning observed for non-bookers as well, and can be realized faster than downstream business metrics. This is a classical challenge in product optimization shared by many online platforms[2].

Our challenges and contributions are unique in that clicks, view duration, and traditional search metrics are not good leading indicators for bookings on our platform. Movements in click metrics are not consistently correlated with downstream business metrics,

[1] Airbnb.com
[2] https://www.slideshare.net/bonbonsuperbonbon/defining-true-north-metrics-to-quantify-engagement-at-linkedin

making it difficult to draw conclusions from their increases or decreases. We, therefore, develop a new metric based on a machine learning model, screening signals of friction from signals of intent. This metric captures a new stage in the booking funnel between Search and Contact, the searcher found a listing which is a good match for them. We can use this metric for a range of applications: drive more analytics insights of our booking funnel, increase experimentation velocity, and feed less sparse signals to our ranking models.

This paper outlines the challenges of using traditional search engagement metrics within the complex Airbnb booking funnel. However, by sacrificing some simplicity and turning towards a machine learning derived user intent metric, "intentful listing view", we were able to overcome these challenges. We demonstrate the machine learning based metric is directionally correlated with downstream business metrics and sensitive across a variety of historical search experiments. The contributions of our work are two-fold:

1. Developing a simple yet effective approach to capture intent from page engagement signals.
2. Creating a new conversion leading indicator for increased experimentation velocity and less sparse ranking optimization.

2 ENGAGEMENT SIGNALS IN AIRBNB SEARCH

The use of observable user behaviors, such as clicks, is a popular approach to evaluate the quality of online search retrieval and ranking systems [2]. These signals are inexpensive to obtain, especially compared to expert judgments, and can be gathered in large quantities. Given these attractive features, user behavior metrics are relatively well-adopted in the field of Web Search [4]. However, given the signal is naturally noisy and user behavior is multi-faceted, deriving metrics which truly reflect the relevance of results can be challenging [6].

The direct application of user behavior metrics to Airbnb search poses a few challenges. In the context of Web Search, the time it takes a user to complete a "search task" can be measured in minutes and at most a few query reformulations [8]. For Airbnb search we are not afforded such a luxury. Even the most eager to book user can spend 3 days searching, execute 12 unique searches, and view the description page of 11 listings before booking.

Motivated by click-through-rate we implemented a similar concept at Airbnb, number of listing views per user [5]. We evaluated the quality of this metric as a leading indicator of booking, via a meta-analysis of historical experiments. We ideally want to observe a strong correlation and directional agreement between movements in the leading indicator metric and booking conversion. Figure 1 shows there is very mild correlation between the listing view metric and booking conversion for 20 search experiments; confidence intervals are omitted as all results are statistically significant. In 40% of the experiments, represented by green dots, the number of listing views decreases yet booking conversion improves, or the inverse. There is also only a 0.06 Kendall's tau correlation between the metrics. The contradictions and lack of correlation make it difficult to utilize this click-based metric with confidence. At Airbnb clicks and views are not a strong signal of intent. Much of the information

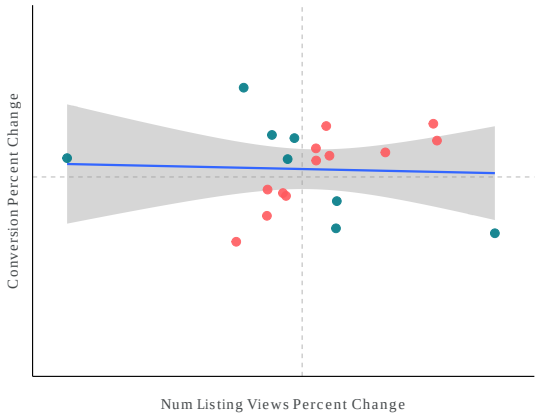


Figure 1: Correlation of Num Listing Views Metric with Booking Conversion Metric

about a listing is unstructured and can only be found after viewing the details on the listing page. Therefore, it is not clear if increases in number of listing views is a signal of consideration or increased friction.

In an effort to remove quickly abandoned listing views, we explored restricting to only long dwell-time views [9, 10]. Across a variety of duration thresholds and transformations as suggested by [11] and [12], we were unable to achieve reasonable correlation with business metrics. Listings have varying numbers of photos, description lengths, and number of reviews; this can cause unwanted bias and variation in comparing view duration across listings. We also explored Airbnb product specific engagement information such as clicking into the price details module on the listing page as well as searching for specific dates. Efforts produced metrics which either did not correlate with business metrics, or were too focussed on a specific path of engagement, and did not generalize well to all searchers. For example, some searchers without specific dates in mind will execute searches without entering dates, and determine their travel dates based on the availability of listings they like. Therefore, a metric related to searching for specific dates would not extend to that user segment.

Taking a step-back, we invested analytical resources in mapping out users' specific product surface touch-points on the path to booking. We identified that viewing the listing description page is a touch point which every guest passes through on their way to booking, and is a step which is relatively high-up in the funnel. We also identified that users perform a range of actions on the listing description page: clicking photos, paginating reviews, etc. The list of actions is extensive and performed in varying sequences, such that it is difficult to comprehend globally positive and negative actions from basic summary statistics. Note that users will also sometimes abandon a listing description page, only to return to it a day later.

Zhou et al. [13] successfully improved e-commerce recommendations using micro-actions between users and items, not solely relying on purchase signals. In addition, Kim et al. [7] trained a machine learning classification model using search query attributes, page attributes, and dwell-time, to predict if a web search result

click is associated with user satisfaction. Motivated by these successes, we attempt to build a machine learning model which can combine the rich collection of user micro-actions on the listing description page to produce a more reliable engagement signal.

3 INTENTFUL VIEW MODEL

Our objective is to build a model which can identify guest listing views which signal that a user is considering booking a listing, versus those which the user has dismissed from consideration. We formalize this objective as a classification task.

3.1 Problem Formulation

Given our focus is on the searching to contacting stage of the funnel, we utilize the action of contacting a listing as our target outcome variable. Therefore, for a given user, u_i, and listing, l_k, we model the probability a user's view of the listing, v_{u_i,l_k}, will result in a listing contact, i.e.

$$Pr\left(u_i \text{ contacts } l_k \mid v_{u_i,l_k}\right).$$

The above classification formulation is straightforward, but there is the added complexity that some users contact a listing and some do not. We include all users in our training data, contacters and non-contacters, so as to avoid the selection-bias of only including users who contact. We emphasize learning the delineation between a contacter and non-contacter while still focussing on user-listing level predictions. This is achieved by applying some filtering in terms of positive and negative examples in our training data:

> **Positives** - listing views of users which lead to contact of the viewed listing,
> **Negatives** - listing views which did not lead to a contact, and the searcher did not contact any other listings for the given trip.

Note that it is possible for a user to contact multiple listings for a trip, and all such examples are included in the data as positives.

It can be seen that a contacter's only contribution are the listings he or she contacted, and listings that were viewed but not contacted for the given trip are omitted. While this can be viewed as over simplifying the problem, it helps reinforce the objective of separating the actions of guests which have reached the stage to contact, versus those who will not contact. Previous analyses at Airbnb have shown that most users who contact a listing will view and strongly consider a set of listings. But since he or she can only stay in one place for a given location and dates, most users choose to contact only one of the listings they were considering. Therefore, including negatives from users who contact crosses into the territory of learning preferences, and less so a threshold of intent. Their omission improves our signal to noise ratio for modeling general intent to contact.

Figure 2 illustrates which user listing views are included in the training data (opaque), and their associated label, versus those which are excluded (transparent). The "thumbs up" denotes that a user was implicitly interested in a listing. Note that the first and second user "liked" multiple listings, but only the contacted listings' views are included in the training data, while all the listing views from the non-contacting user are included.

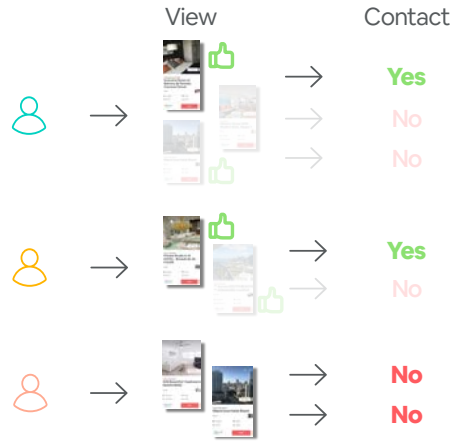


Figure 2: Illustration of Training Data Construction

Many guests view a listing multiple times before booking, this is a valuable signal of intent. We, therefore, update the probability of contact with each view, incorporating information about the current view as well as past views, up to a reasonable look-back time threshold. This updates our objective as follows:

$$Pr\left(u_i \text{ contacts } l_k \mid v_{u_i,l_k,t_1},\ldots,v_{u_i,l_k,t_m}\right),$$

where m denotes the rank number of the current view of listing l_k by user u_i at time t_j.

3.2 Feature Engineering

We implement granular logging of user micro-actions on the listing description page, which enables us to build a rich feature set. Motivated by Zhou et al. [13] we log dwell time on the entire listing page as well as page scroll depth. We also log number of clicks, dwell time, expand/collapse, and pagination for each individual section of the listing page, e.g. photo carousel, description text, reviews, etc. From this logging we generate four classes of features:

1. **Actions on Most Recent View** - This includes features of user actions for the most recent view of the listing.
2. **Actions from Previous Views** - We predict contact with respect to the current view as well as all previous views of the listing by the user. We therefore include a class of features which are aggregated summary statistics of the user actions from previous views of the listing. We use the same numeric features from the "Recent View" class, calculating the min, max, and mean value across all previous views. We also found it improved model performance to include as their own individual class the feature values for the very first view of the listing, in addition to the aggregates.
3. **Time Gap Info of Previous Views** - Information about the timing pattern of a user's multiple views of a listing also impacts model performance. We include features with basic time information: number of previous views of the listing, time since last view of the listing, and min, max, and average

time between views of the listing. For cases when a user has no previous views of the listing, these values are all set to null.

4. **Trip Information** - User behavior which signals intent can be different depending on how far out the trip is (lead time), the length of the trip, and/or how many guests are attending. We include features with this information, as well as how many unique checkin and checkout dates a user enters on the listing's availability module. Users with a strong affinity for a listing will sometimes modify their travel dates to fit the availability constraints of the listing.

3.3 Model Training

We train on user logs from a fixed time period and evaluate on an independent test set from a future time period; care is taken to ensure no user is represented in both sets. We perform some basic filtering by removing examples from users with a single view of a listing with duration less than 5 seconds; this threshold is chosen because it removes very low-intent negative examples without affecting the coverage of positive examples. In order to avoid overfitting to users with many views, we weight training examples by the reciprocal of the total number of examples per user. All continuous and count features are log-transformed. We train our model using the XGboost[3] library with a log-loss objective function. Tuning parameters are selected by evaluating performance on a separate validation set.

3.4 Evaluation of Model Performance

We evaluate the AUC performance of our XGboost classification model compared to a variety of alternative models:

(1) Total View Duration - We score the intent level of listing views by the sum total view duration of the current view and previous views of the listing; this is equivalent to the traditional dwell-time weighting of clicks.
(2) Number of Repeat Views - We score views by total number of previous views of the listing by the user
(3) Decision Tree Depth 2, 3, and 4 - Using the same features as in the full XGboost model, we train a few simple decision trees of varying depth with Gini impurity as the split criterion [1].

Figure 3 shows the ROC curve and AUC performance of each model on the test set. The XGboost model out performs all other models. This is not surprising given the increased complexity of the model, but the dramatic improvement in performance lends a strong argument to continue with the more complex model in lieu of something simpler.

3.5 Evaluation of Model Robustness

The listing description page is a product surface which is still under development. We test product changes attempting to improve booking conversion which can re-order sections, collapse/expand sections, alter the content within a section, or delete a section altogether. Our model includes features related to user actions on the listing description page UI, so we want to be sure the model is for the most part robust to UI changes.

[3]https://xgboost.readthedocs.io/en/latest/

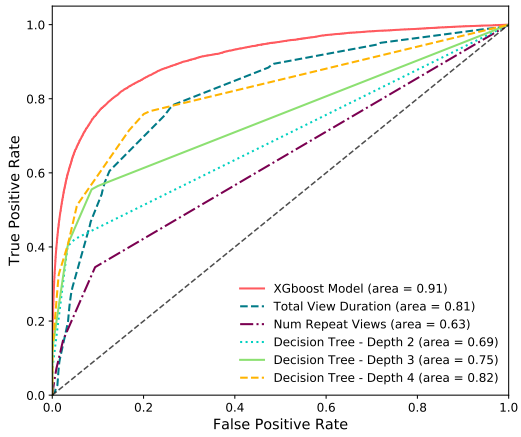


Figure 3: Model Performance Evaluation

Table 1: Model Evaluation on Different Listing Page Design

	Current Listing Page	Previous Listing Page
AUC	0.91	0.89

We compare the AUC performance of our model on the current listing page UI, on which it was trained, versus a previous version. The previous UI version differs fairly drastically from the current version in that there are additional sections and all sections are expanded by default, so the listing description page is longer. Table 1 shows that while there is an expected slight decline in model performance on the different design, the change is not too dramatic with an AUC drop of only 2%.

Since the model performance can be slightly influenced by UI changes, we invest in monitoring metrics related to model performance. Via Airflow[4], we calculate and report daily the model's predictive performance (AUC), as well as descriptive statistics and distributional characteristics of the prediction output of the model and feature inputs. In the future we intend to explore semi-regular automatic retraining of the model via a similar process.

4 APPLICATION TO EXPERIMENTATION

Similar to other online marketplaces, A/B experimentation is a vital component of optimizing the Airbnb search experience. As previously outlined, the long booking conversion funnel can pose challenges in terms of experimentation velocity. Leading indicator metrics help us more quickly determine which product changes are tracking towards improved booking conversion and those which are likely hurting conversion. It also enables us to understand how experiments are affecting different subsets of users, especially those with less traffic. In this section we demonstrate how we use the output from the intentful view model to build a strong leading indicator metric for search experiments, "intentful viewer".

An effective leading indicator metric is consistently correlated with downstream business metrics and ideally more sensitive, i.e.

[4]http://airbnb.io/projects/airflow

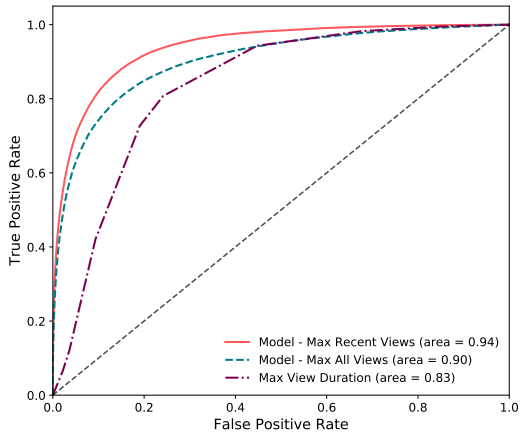


Figure 4: User-Level Prediction of Contacter Evaluation

reaches statistical significance before downstream metrics [3]. Depending on the product change there are cases in which the correlation will not hold, but the goal is for those cases to be rare and easily explainable. We therefore evaluate the quality of our proposed leading indicator metric based on those two factors.

4.1 User Level Prediction

Doing a view which signals intent to contact can be considered a milestone for a user in the search to book funnel. We, therefore, convert the output of our model from user-listing level to simply user level. We do not modify the model, but instead aggregate the model predictions across all listings viewed by the user. We make the reasonable assumption that a users overall intent to contact any listing is bounded from below by their maximum intent to contact an individual listing. More formally, for a user, u_i, with views across listings $\mathcal{L}_i = l_1, \ldots, l_r$, then

$$Pr\left(u_i \text{ contacts any listing}\right) >= \max_{l_k \in \mathcal{L}_i} Pr(u_i \text{ contacts } l_k)\,. \quad (1)$$

This formulation is likely a conservative probability of contacting, but enables us to generate a user level metric which is easier to interpret within the setting of an online experiment.

Figure 4 shows the ROC performance of the user-level prediction with target variable of contacter vs non-contacter. For users with multiple views of a listing, we find it improves predictions to only use the probability of contact from the most recent view of each listing in equation 1. We also include the comparison to an alternative model where user intent is ranked by maximum view duration on an individual listing. The intentful view model user-level prediction, based on most recent view, has a 13% better AUC than dwell time.

4.2 Metric Definition

We attempt to establish a probability threshold such that any user with a predicted probability above the threshold is declared "intentful". Therefore, within a given experiment we can compare the share of users in control and treatment who have reached this stage, with the conclusion that a product change which drives more users into an intentful state will lead to an increase in booking conversion.

Based on our previously outlined goals, we ideally want the selected threshold to result in a metric which is both:

1. Directionally correlated with our downstream booking conversion metrics, and
2. more sensitive than downstream metrics.

Quantifying if two metrics are directionally correlated is straightforward. The directional correlation of the relative lift of two metrics, m_1 and m_2, between branches A and B within an experiment, E_k, can be expressed as

$$D\left(m_1, m_2, E_k\right) = I\Big(\text{sgn}\Big(\,L(m_1, E_k)\,\Big) = \text{sgn}\Big(\,L(m_2, E_k)\,\Big)\Big), \quad (2)$$

where $L(m_j, E_k)$ denotes the lift of metric m_j, and $I(\cdot)$ and $\text{sgn}(\cdot)$ are the indicator function and sign operator respectively. We drop the references to branches A and B for simplicity.

Similar to Kharitonov et al. [6], we measure the sensitivity of a metric, m_j, on a given experiment using the z-score statistic, defined as

$$Z\left(m_j, E_k\right) = \frac{\bar{m}_{j_B} - \bar{m}_{j_A}}{\sqrt{\text{Var}\left(\bar{m}_{j_B} - \bar{m}_{j_A}\right)}},$$

where $\bar{m}_{j_A}$ and $\bar{m}_{j_B}$ are the samples means of metric m_j for experiment branches A and B respectively, in experiment E_k. A higher absolute value of the z-score implies higher confidence in the difference of the metric means between control A and treatment B.

When comparing sensitivity, we focus on the inequality comparison of sensitivity (e.g. greater-than) as opposed to the raw magnitude. For a given experiment, we therefore compare if the z-score of our derived metric is larger than that of our downstream metric. Based on the direction of the downstream metric z-score we transpose the scores such that an increase is always desired, i.e. branch B > A.

We present our downstream booking conversion metric as m_C, and derived intentful viewer metric as $m_I(\tau)$, which depends on a probability threshold τ. Using our previously defined expressions, for a set of experiments $\mathcal{E} = \{E_1, \ldots, E_N\}$, we wish to optimize two expressions:

$$\frac{1}{N}\sum_{E_k \in \mathcal{E}} D(\; m_I(\tau),\, m_C,\, E_k\;)\,, \quad (3)$$

$$\frac{1}{N}\sum_{E_k \in \mathcal{E}} I(\; Z\left(m_I(\tau), E_k\right) > Z\left(m_C, E_k\right)\;)\,. \quad (4)$$

We evaluate the above expressions over a range of τ values for a set of historical search experiments. Ideally, we would only use ground-truth experiments for which the booking conversion metrics are statistically significant. We, however, only have 20 such experiments with the required logging to score our intentful view model. In order to increase our sample size we include experiments which did not meet the traditional 0.05 significance level, expanding to experiments with a booking conversion metric p-value up to 0.3. This doubles our sample size of experiments, increasing it to 42.

Since we have stronger confidence in the results of some experiments versus others, we weight the observations by $\omega = (1 - \text{p-value})$, using the booking conversion metric p-value for the experiment. The intuition behind this weighting comes from the notion that

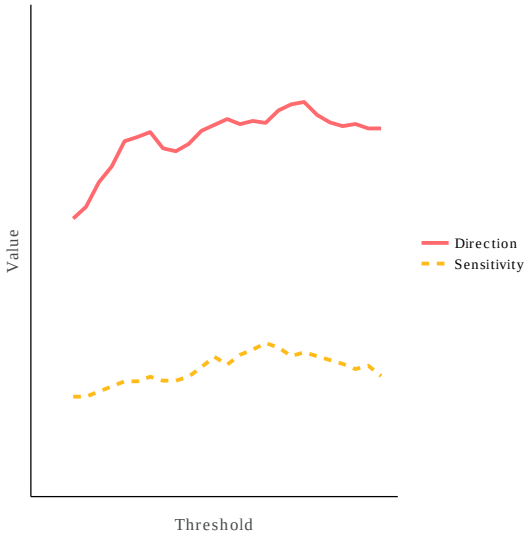


Figure 5: Intentful Viewer Directionality Agreement and Sensitivity for Range of τ Values

the p-value reflects the evidence against the null hypothesis. So the smaller the p-value, the more confidence we have in the experiment results, and therefore the experiment receives more weight. We modify expressions 3 and 4 as follows,

$$\frac{1}{\sum \omega_{m_C,E_k}} \sum_{E_k \in \mathcal{E}} \omega_{m_C,E_k} \, D(\, m_I(\tau),\, m_C,\, E_k \,), \tag{5}$$

$$\frac{1}{\sum \omega_{m_C,E_k}} \sum_{E_k \in \mathcal{E}} \omega_{m_C,E_k} \, I(\, Z\,(m_I(\tau), E_k) > Z\,(m_C, E_k) \,). \tag{6}$$

Figure 5 shows the evaluation of expressions 5 and 6 across the 42 experiments for a range of values of τ. There is a general concave shape for both directionality and sensitivity. Given the somewhat flat trend of sensitivity after a certain threshold, we select τ such that it maximizes directionality, we refer to this selected threshold as τ^*.

4.3 Results

For our selected τ^*, we plot in Figure 6 the correlation of the percent change in the intentful viewer metric versus the downstream booking metric across 20 statistically significant experiments. It can be seen our new metric achieves a strong correlation and directional agreement with the conversion metric, especially when compared to the number of listing views metric from Figure 1. In Table 2 we show a comparison of summary statistics of the two metrics. Our intentful viewer metric is able to achieve a directional agreement for 95% of the experiments, whereas the listing view count metric is only 60%. We also see that our new metric is more sensitive than the conversion metric 35% of the time. This shows there are still improvements to be made in terms of increasing the sensitivity, perhaps incorporating the magnitude of the difference of the z-score in the optimizing expression can help. We leave that for future work at this moment.

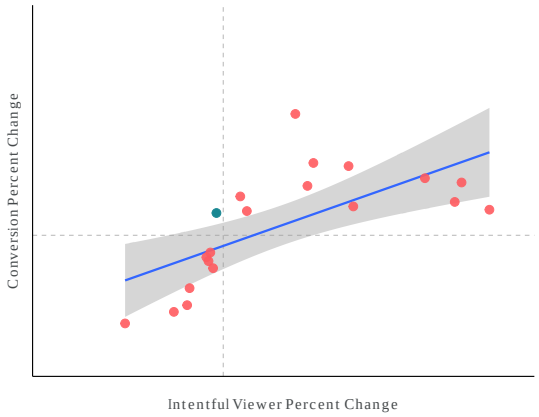


Figure 6: Correlation of Intentful Viewer Metric with Booking Conversion Metric

Table 2: Intentful Viewer vs Num Listing Views Metric

	Intentful Viewer	Num Listing Views
Direction Agreement	0.95	0.60
Kendall's Tau Cor	0.62	0.06
Greater Sensitivity	0.35	0.20

5 CONCLUSIONS AND FUTURE WORK

We present an effective approach, using a machine learning classification model, to capture user intent via user engagement signals. The predictive performance and robustness of the model are evaluated extensively. We use the output of the model to build a user intent metric, "intentful viewer". We demonstrate via a meta-analysis of historical experiments that this metric is strongly correlated with business output metrics.

In the future, as more historical experiments are available, we will explore alternative methods for determining the optimal probability threshold for our "intentful viewer" metric. We also currently predict at the listing-level and then aggregate model predictions up to the user-level. It is possible to jointly model, via a conditional probability representation, a user's overall intent and how that intent disperses to individual listings. We can also iterate on improving model predictive performance by transitioning to neural network sequence models. This would provide a more advanced method for incorporating a user's historical actions on a listing, as the user history can be fed as a raw sequence to the model as opposed to a set of summary statistics as we currently employ. Finally, we will explore the use of the intentful view predictions as a more reliable implicit feedback signal in our search ranking models.

ACKNOWLEDGMENTS

We would like to thank Claire Lebarz, Thomas Legrand, Navin Sivanandam, Richard Dear, and the entire Airbnb Search Team for their feedback and suggestions throughout the development of this work.

REFERENCES

[1] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. 1984. Classification and regression trees. (1984).

[2] Olivier Chapelle, Thorsten Joachims, Filip Radlinski, and Yisong Yue. 2012. Large-Scale Validation and Analysis of Interleaved Search Evaluation. *ACM Transactions on Information Systems* 30, 1 (Feb. 2012).

[3] Alex Deng and Xiaolin Shi. 2016. Data-Driven Metric Development for Online Controlled Experiments: Seven Lessons Learned. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 77–86. https://doi.org/10.1145/2939672.2939700

[4] Alexey Drutsa, Gleb Gusev, and Pavel Serdyukov. 2015. Future User Engagement Prediction and Its Application to Improve the Sensitivity of Online Experiments. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 256–266. https://doi.org/10.1145/2736277.2741116

[5] Diane Kelly and Jaime Teevan. 2003. Implicit Feedback for Inferring User Preference: A Bibliography. *SIGIR Forum* 37, 2 (Sept. 2003), 18–28. https://doi.org/10.1145/959258.959260

[6] Eugene Kharitonov, Alexey Drutsa, and Pavel Serdyukov. 2017. Learning Sensitive Combinations of A/B Test Metrics. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM '17)*. 651–659.

[7] Youngho Kim, Ahmed Hassan, Ryen W White, and Imed Zitouni. 2014. Modeling dwell time to predict click-level satisfaction. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 193–202.

[8] Ron Kohavi, Alex Deng, Brian Frasca, Roger Longbotham, Toby Walker, and Ya Xu. 2012. Trustworthy Online Controlled Experiments: Five Puzzling Outcomes Explained. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '12)*. ACM, New York, NY, USA, 786–794. https://doi.org/10.1145/2339530.2339653

[9] Chang Liu, Jingjing Liu, Nicholas Belkin, Michael Cole, and Jacek Gwizdka. 2011. Using dwell time as an implicit measure of usefulness in different task types. *Proceedings of the American Society for Information Science and Technology* 48, 1 (2011), 1–4.

[10] Ryen W White and Diane Kelly. 2006. A study on the effects of personalization and task information on implicit feedback performance. In *Proceedings of the 15th ACM international conference on Information and knowledge management*. ACM, 297–306.

[11] Songhua Xu, Hao Jiang, and Francis Chi-Moon Lau. 2011. Mining user dwell time for personalized web search re-ranking. In *International Joint Conference on Artificial Intelligence (IJCAI 2011)*. AAAI Press/International Joint Conferences on Artificial Intelligence.

[12] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond clicks: dwell time for personalization. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 113–120.

[13] Meizi Zhou, Zhuoye Ding, Jiliang Tang, and Dawei Yin. 2018. Micro Behaviors: A New Perspective in E-commerce Recommender Systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 727–735. https://doi.org/10.1145/3159652.3159671

Detection and Analysis of Self-Disclosure in Online News Commentaries

Prasanna Umar
Pennsylvania State University
University Park, Pennsylvania
pxu3@psu.edu

Anna Squicciarini
Pennsylvania State University
University Park, Pennsylvania
acs20@psu.edu

Sarah Rajtmajer
Pennsylvania State University
University Park, Pennsylvania
smr48@psu.edu

ABSTRACT

Online users engage in self-disclosure - revealing personal information to others - in pursuit of social rewards. However, there are associated costs of disclosure to users' privacy. User profiling techniques support the use of contributed content for a number of purposes, e.g., micro-targeting advertisements. In this paper, we study self-disclosure as it occurs in newspaper comment forums. We explore a longitudinal dataset of about 60,000 comments on 2202 news articles from four major English news websites. We start with detection of language indicative of various types of self-disclosure, leveraging both syntactic and semantic information present in texts. Specifically, we use dependency parsing for subject, verb, and object extraction from sentences, in conjunction with named entity recognition to extract linguistic indicators of self-disclosure. We then use these indicators to examine the effects of anonymity and topic of discussion on self-disclosure. We find that anonymous users are more likely to self-disclose than identifiable users, and that self-disclosure varies across topics of discussion. Finally, we discuss the implications of our findings for user privacy.

CCS CONCEPTS

• **Security and privacy** → Social aspects of security and privacy.

KEYWORDS

Online self-disclosure, public platforms, privacy, anonymity

ACM Reference Format:
Prasanna Umar, Anna Squicciarini, and Sarah Rajtmajer. 2019. Detection and Analysis of Self-Disclosure in Online News Commentaries. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313669

1 INTRODUCTION

Public discourse has been emboldened by the ubiquitous use of social media, including social networking websites, blogs, and online newspaper comment forums. Users can now share information, express opinions, and discuss various topics of interest with a wide audience through these platforms. This engagement often includes *self-disclosure*— the communication of personal information to others [10]. Online social networks and comment forums thrive on user-generated content and, therefore, encourage self-disclosure [33].

Self-disclosure is generally considered conscious behavior, in which users engage in pursuit of strategic goals including social connectedness, validation, self-expression, relational development, identity clarification, and social control [1, 6, 11]. However, there are associated costs of this behavior to users' privacy. Personal information posted online becomes shared knowledge and may be retained or shared downstream, in ways or by parties unintended by its original owner. Understanding self-disclosure is paramount to devising ways of mitigating these risks and achieving data parsimony [23]. We note that while work on self-disclosure in seemingly bounded environments (e.g. social networks [6, 23]) has shed light on users' motivations for and practices of self-disclosure, less is known about user disclosures in public commentaries. In this work, we address these gaps.

Following, we present in-depth analyses of self-disclosure in a longitudinal dataset of about 60,000 comments on 2202 news articles from four major English news websites [5]. Inspired by prior work on online disclosure and anonymity, we focus on two contextual features that we suspect are related to self-disclosure in public forums, namely anonymity and topic of a commentary.

Our contribution is two-fold. First, we provide an automated method for detecting language indicative of various types of self-disclosure, leveraging both syntactic and semantic information present in texts. We identified 9 categories of self-disclosure, ranging from users' personal attributes (e.g. location, sexual orientation) to subjective categories, such as opinions. Second, we use these results to examine the effects of anonymity and topic of discussion on self-disclosure. Our results indicate that users who have greater anonymity are more likely to self-disclose online, and that self-disclosure varies for different topics of discussion. Finally, we discuss the implications of our results on online self-disclosure for privacy. To the best of our knowledge, this is the first study which uses an automated detection method to analyze triggers of oversharing and personal information disclosure outside the classic boundaries of social network domains.

The remainder of the paper is organized as follows. In the next section, we present related work in this area. Next, we provide background and discuss our hypotheses. We then discuss a novel method that we used to detect self-disclosure in online public platforms, followed by validation of the method. The remainder of this paper is dedicated to hypotheses testing and discussion of findings.

2 RELATED WORKS

People engage in self-disclosing behavior consciously for various intrinsic and extrinsic benefits. As disclosing personal information

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313669

online has privacy risks, users try to maintain a balance between self-disclosure costs and benefits. They utilize different media affordances such as anonymity, audience representations, etc. in this pursuit. Users' self-disclosure decisions are therefore dependent on these media affordances. Previous studies have used survey-based functional models incorporating social media affordances to examine self-disclosure characteristics in social networking sites (SNS). A recent study [9] highlighted the differences in disclosures between private SNS and public SNS. The authors showed the roles of audience representations (bounded vs unbounded), and network characteristics (size and diversity) in describing self-disclosure. However, they noted the need for extending studies of self-disclosure to online platforms other than Facebook and Twitter.

Anonymity, as a social media affordance contributing to self-disclosure, has been extensively studied under the term online disinhibition [31]. Multiple studies have noted the positive relationships between anonymity and self-disclosure in computer mediated communication [16, 21]. However, the findings are not consistent for all types of online platforms. For instance, Qian and Scott [27] reported no association between increased visual anonymity with greater self-disclosure. Similarly, Chen et al. [8] showed negative effect of network anonymity on self-disclosure on Sina Weibo platform but reported positive effect of perceived anonymity. These mixed findings suggest that different online media have different affordances of anonymity and therefore, the relationship between anonymity and self-disclosure is dependent on media characteristics. In this work, we study the effect of anonymity on self-disclosure in online news commentaries which afford different levels of anonymity as described in in Section 3.

Studies on self-disclosure have mostly relied on manual coding and analysis of users' contents primarily into broad categories of personal information, personal thoughts and opinions, and personal feelings and emotions [4, 14]. Personal information is related to facts about a person while textual contents that convey thoughts, opinions, feelings and emotions are subjective in nature. Each of these broad categories may include several sub-categories such as biographic information, property, location, family details, etc [7, 14]. In line with existing literature, we adopt similar terminologies and consider two broader categories of self-disclosure namely: objective and subjective (See Table 1). Objective categorization include factual information about a person: birthday/age, race, sexual orientation, affiliation, money, relationships and experiences. Subjective categorization includes categories related to internal states of an individual: interests, opinions and feelings.

Automated detection of self-disclosure has been tackled by recent studies [7, 33]. Caliskan et al. [7] created a supervised machine learning method to detect private information in tweets through the use of privacy ontology, named entity recognition, topic modeling and sentiment analysis. They assigned privacy scores to each user based on the percentage of the userâĂŹs tweets annotated. We used a different approach as we rely on an unsupervised method to detect individual self-disclosure categories in texts. Bak et al. [3] created a supervised method with topic models and SVM to detect personally identifiable information (PII) and personally embarrassing information (PEI). Precision and recall for their supervised method of PII detection were 0.23 and 0.21 while for PEI precision and recall were 0.30 and 0.23 respectively. In comparison, our unsupervised method achieved greater precision and recall across most categories of self-disclosure. Further, Bak et al. [2] applied modified latent Dirichlet allocation (LDA) topic models for semi-supervised classification of Twitter conversations into three self-disclosure levels. They did not categorize texts into individual categories of self-disclosure as we do.

3 BACKGROUND AND HYPOTHESES

We study self-disclosure as it correlates to a few critical contextual dimensions. Here, we discuss our hypotheses, and ground them in relevant literature. The first dimension we consider is *anonymity*.

H1: Anonymous users are more likely to self-disclose than identifiable users in online public commentaries.

Other authors have found that anonymity contributes to increased self-disclosure both on- and offline [21, 25, 29]. In particular, previous studies have suggested that, emboldened by the cover of anonymity, people may feel less restrained and express fewer inhibitions [31]. Many online platforms offer some degree of anonymity to users in service to freedom of expression [30], where the nature of that anonymity varies across platforms. To comment within article commentaries, in most cases, users either need to sign in to a commenting platform (e.g., Disqus [12]), provide an email address (medium/high anonymity if a throw-away account is used [19]) or provide links to an existing social network account (high identifiability), e.g. Facebook.

Recent work indicates [32] there is a predictive relationship between anonymity state and perceived anonymity. Users intuit that they are more identifiable when linking social network accounts than they are when using pseudonyms. Our analyses consider two anonymity states among users, representing pseudonym vs. social-media-linked accounts. *Following, we refer to users logged into Disqus as "anonymous" and users logged in through existing social media accounts as "identifiable".* Our first hypothesis considers differences in self-disclosure between anonymous and identifiable users.

H2: Users' self-disclosure varies across topics.

A second contextual dimension that we suspect is relevant to self-disclosure is the *topic* of a discussion. Public discourse in news websites and opinion forums is topic-oriented. The news article or the opinion column usually determines the initial topic of discussion. Because individuals relate to content based on prior experiences and feelings, they respond differently to different topics [33] where self-disclosure is a subset of that response. Hence, we hypothesize that users self-disclose differently across topics.

4 DETECTION OF SELF-DISCLOSURE

We anchor our approach for detection of traces of self-disclosure in an opinion extraction technique [17], wherein opinion, opinion-holder and the opinion topic are extracted. However, our goal is not limited to detection of subjective language (opinion). We intend to detect objective language pertaining to self-disclosure as well. To do so, we leverage both the semantic and the syntactic resources in a sentence. The intuition is that a self-disclosing sentence has self-reference[1] as a subject or the object of reference is the self.

[1]We do not consider any indirect forms of self-disclosure with no explicit self-reference in the text and we do not detect the degree of self-disclosure

Table 1: Description of categories of self-disclosure used as instructions for labelling survey

Language	Categories	Description
Objective	Birthday/Age	Sharing one's own birthday information or references to own age.
	Race	Sharing one's own race such as being black, white, Hispanic, etc
	Sexual Orientation	Sharing one's sexual orientation and identity such as being straight or LGBT. Includes marital status
	Location	Sharing one's own location such as town, city, states, proximity to a landmark, etc
	Affiliation	Sharing one's nationality, religion, political affiliation, loyalties to groups and brands of a certain nationality, etc
	Money	Sharing one's own financial worth, monetary values of property, plans/goals related to money, etc
	Relationships	Sharing information about the family composition such as having children, brothers, sisters, etc
	Experience	Sharing past experiences of events, habits, work-life, etc. Includes positive or negative experiences and recollections of any past events or memories
Subjective	Interests	Sharing one's own hobbies and interests, including pastimes, favorites, tastes in music, movies, and books. Includes disclosures about pets, as well.
	Feelings	Expressions of deep personal feelings, including humiliation, desires, anxiety, depression, fears, pain, and beliefs which most people would likely disclose only to a friend or family
	Opinion	Discussing one's own opinions, attitudes, and beliefs about current and/or historical events that one is NOT relating to personal experience. Includes views on government, trends, specific events in entertainment/sports, religion, etc.
-	Other	Personal information about the author is revealed but can't be categorized in any categories
-	None	No information about the author revealed

Presence of first person pronouns is generally considered as linguistic markers for self-disclosing texts [15]. Moreover, several studies have used presence of first person pronouns as a prominent feature in determining self-disclosure in online contents [2, 11, 33].

The categories of self-disclosure as shown in Table 1 are distinguished based on related named entities present in the sentence in the vicinity of related verbs. For instance, anyone disclosing where he/she lives uses sentences with verbs related to location like live, stay, etc together with the place where they live. A typical example would be the sentence "I live in Pennsylvania". In this scenario, the author of this text includes the place of living (a location entity) and the verb "live" (related to location) with reference to the self (use of "I"). Hence, our method identifies self-disclosure categories in text through the knowledge of subject, verb, object and named entities. The overall method of detection and categorization of self-disclosure consists of four phases: (1) dictionary construction, (2) subject-verb-object triplet detection, (3) named entity recognition and (4) rule based matching.

4.1 Category Specific Dictionary Construction

We constructed a vocabulary of verbs often associated with the different categories of self-disclosure. The dictionary consists of verbs used in a labeled dataset from a study of self-disclosure in Airbnb profiles [22]. The dataset consisted of 5248 annotated sentences from 1234 user profiles. Each sentence was categorized into one or many of the eight categories. We extracted the frequent root verbs from sentences categorized into categories of work or education, origin or residence, travel, interests and tastes, and relationships to construct dictionaries of verbs associated with our categories. The dictionaries also consist of additional verbs manually added to aid in the categorization. We constructed multiple dictionaries of verbs: subjective (36), location (21), affiliation (20), , money (12), birthday/age (3), and race and sexual orientation (1). We also created additional dictionaries of attributes for categories of relationships, race, and sexual orientation. Specifically, names of different racial groups, types of sexual orientation and types of social and family relationships were compiled to create these individual dictionaries. We used these dictionaries as entities for the aforementioned three categories.

4.2 Subject, Verb and Object Extraction

We used the Python package Spacy based subject verb object extraction[2]. The implementation was adapted and modified to suit our goals. The subject, verb and object extractor takes text as input, pre-processes the text, splits it into individual sentences and further splits a sentence into individual clauses if present. Text pre-processing includes expanding the contractions such as "I'm" to "I am". Each clause in a sentence passes into the dependency parser and all the verbs are extracted. The dependencies or the syntactic relationships between other tokens in the clause with the verbs is used to determine the subject and objects in the text. Our method was aware of the passive voice and negation in sentences.

4.3 Named Entity Recognition

We attribute the presence of "real-world object" called a named entity like a location, nationality, etc in a sentence as a distinguishing feature among different categories of self-disclosure. We used a model[3] trained on OneNotes corpus, which can detect about 18 different types of entities[4]. Entities of DATE and CARDINAL (number) were used for birthday/age category. Location related entities such

[2]https://github.com/NSchrading/intro-spacy-nlp/blob/master/subject_object_extraction.py
[3]https://github.com/explosion/spacy-models/releases//tag/en_core_web_lg-2.0.0
[4]Details about entities: https://spacy.io/api/annotation

as GPE, FAC and LOC served as distinguishing entities for location category while NORP (Nationalities or religious or political groups) distinguished affiliation category in our categorization scheme.

4.4 Rule Based Matching

Overall, the self-disclosure categorization scheme remained similar for all the objective categories. Self-reference with presence of specific category related verb and appropriate named entities signaled self-disclosure in text. As the proximity of named entities in the sentences vary according to sentence construction, we added a proximity window of up to five words on either side of verb to effectively detect specific categories. It is useful in removing false positives as in case of sentence like *I have countless arguments with seemingly educated people in many countries on why Singapore works.* Here, all the attributes for location disclosure such as matched verb "have" with location entity "Singapore" and subject "I" are present but the proximity check prevents it's categorization into location category. Similarly, we added a proximity window of three words between subject and verb. It should also be noted that we do not include any named entities in the categorization scheme for subjective categories like personal interests, opinions and feelings. Such categories are solely based on presence of first person pronouns and verbs often used in expression of subjectivity.

5 VALIDATION OF METHOD

In this section, we describe process we used to validate our method of detecting and categorizing self-disclosure from user comments.

5.1 Dataset

Our work uses a dataset of user comments on news websites obtained from authors of the paper [5]. It consists of 309319 comments on 52260 news articles crawled from 10 selected news websites over a three month period. We constructed a smaller subset of the data by considering four news websites from majority native English speaking nations: ABC News, CNBC, The Huffington Post and Techcrunch. The time period of the contents was between March and August 2015. We cleaned the data by removing duplicates and comments with no readable text. In total, there are about 59249 comments from 22132 users (14219 identifiable and 7913 anonymous). As described in the earlier section, we differentiate anonymity states based on the account people use to comment on news websites. Commentors on The Huffington Post and Techcrunch used Facebook profile as their identity on these news websites. The users of CNBC and ABC News used Disqus accounts. Accordingly, we consider the users who commented on The Huffington Post and Techcrunch as identifiable and the rest as anonymous. The data consisted of 2202 news articles: The Huffington Post (1136), Techcrunch (119), CNBC (421) and ABC News (526). The anonymous users had higher average number of comments (4, SD = 10.85) than identifiable users (1.94, SD = 2.82).

In order to validate our method, we categorized all the comments using the method described earlier in the paper. Because the data consisted of numerous comments with incomplete sentences and imperative sentences (with no subject), we sampled the dataset after categorization. If we sampled before categorization, the sample size required to include samples of all the relevant categories of self-disclosure would be large. Hence, we performed stratified random sampling only after categorization.

The categorization scheme used here generated multi-label classification for each comment as any comment may contain different types of self-disclosure (including no self-disclosure category). We converted multi-label data into multi-class before sampling. We sampled 200 comments from each class and any class with number of samples less than 200 included all the examples of that class in the overall sample. In this way, we represented all types of examples in the final sample of 3516 comments.

5.2 Labeling

We used the crowd-sourced platform Amazon Mechanical Turk to obtain labels for our sample data. The labeling task was conducted under IRB protocol "STUDY00010405" approved by the Penssylvania State University's Institutional Review Board (IRB). Detailed instructions were given to the labelers about different categories of self-disclosure. Moreover, we included an example of categorization. Each comment was labeled by three different workers into atleast one of the categories in Table 1. To ensure the quality of labels and to ensure that the recruited workers read instructions carefully, the workers were required to answer an attention check question similar to the instructional manipulation check in the paper [26]. We rejected about 8.2% of the responses from the crowd-sourced workers failing the attention check.

We used Gwet's AC1 [13] as a reliability metric for the labeled data. Categorization scheme that classifies a comment into one or many categories of self-disclosure was treated as a group of binary classifiers: one binary classifier for presence/absence of a particular category of self-disclosure. We dropped the category of "organization" because of poor reliability. This category was related to the disclosures of one's workplace, school, membership in any organization, following of sports teams and ownership of properties of a certain brand. Overall, we observed good agreement among labelers: birthday/age (96.5%), race (97.7%), sexual orientation (97.7%), location (85.8%), affiliation (83.5%), money (89.3%), relationships (89.2%), experience (62.9%), interests (93.6%), feelings (81.4%), other (74.1%) and none (70.5%). Also, considering a single label for self-disclosure presence or absence in the text, the reliability score was 70.5%. For each comment, final labels (one or many) were obtained through majority voting. 9.76% of total number of comments did not have a consensus among the labelers.

5.3 Performance of Method

We evaluated the categorization scheme by comparing the crowd-sourced labels against the labels from the algorithm. Performance metrics were calculated only for 3174 comments which had final labels obtained through consensus among multiple labelers. We consider a comment to be self-disclosing in nature if it contains any one of the categories of self-disclosure (except Other) and if None, it is considered as not self-disclosing in nature. Overall, the precision, recall and F1 scores were 98, 89 and 93 respectively. Individual categories' classification performance metrics are summarized in Table 2 with a combined subjective category. We merged the categories of interests, opinions and feelings as a combined subjective category

because unlike objective facts about a person, the subjectivity in interests, opinion and feelings are difficult to distinguish and detect. As pointed out in [24, 34], "because subjectivity is a feature of a person's mind, it is not open to objective observation or verification". Hence, distinguishing between individual categories of subjective language is a task not in purview of our work.

Table 2: Performance metrics for categorization

Category (Support)	Precision	Recall	F1-Score
Birthday/Age (37)	19	43	26
Race (21)	46	62	53
Sexual Orientation (13)	50	62	55
Location (188)	26	70	38
Affiliation (106)	27	42	33
Money (150)	50	23	31
Relationships (206)	36	82	50
Experience (464)	29	50	37
None (97)	15	28	19
Subjective: interests, opinions and feelings (2383)	76	73	74
Overall self-disclosure	98	89	93

6 RESULTS ON HYPOTHESIS TESTING

We performed multiple tests to examine two specific hypotheses as described in detail in section 3. In this section, we present the statistical results of our hypotheses tests.

6.1 H1: Anonymity and Self-Disclosure

We tested our first hypothesis, namely, that anonymous users are more likely to self-disclose than identifiable users. We considered only those comments in a commentary which did not have any self-disclosing comments within five[5] preceding comments, in order to remove any peer effects (peer influence). We aggregated comments by user, obtaining histories of comments for 12,936 users—5218 anonymous and 7718 identifiable. We labeled users who self-disclosed in at least one comment *self-disclosing*. A Chi-squared test revealed that self-disclosure was significantly associated with anonymity [$\chi^2(1) = 59.538$, $p < 0.001$]. Binary logistic regression showed that anonymous users were more likely (1.366 times, $p < 0.001$) to self-disclose than identifiable users, adding support to H1.

Beyond absolute presence/absence of self-disclosure, we were also interested to see if there was a difference in frequency of self-disclosure between users of different anonymity states. Because users with higher number of comments have higher opportunities of self-disclosure, we suspected to see a difference in proportion of self-disclosure between two groups. Hence, we balanced the dataset for the average number of comments (2.91) for both user groups of different anonymity states. Specifically, we created a subset with total of 9468 users and comprising of 4734 identifiable and 4734 anonymous users. Stratified random sampling across strata of different number of comments was done to create this balanced data. We conducted a Mann-Whitney U test to test the differences between the two independent groups of anonymity states in terms of proportion of self-disclosure. The mean rank of proportion of self-disclosure was higher for anonymous users than identifiable users [U=11632245, $p < 0.001$], confirming our hypothesis.

In the next section, we further explore these findings on the effects of anonymity on self-disclosure by accounting for topic of conversation.

[5]We used five comments only because users are more likely to read only few preceding comments either the most popular ones or the most recent ones [20]

Table 3: Top five topics from topic model with frequent keywords

Topic	Keywords
Crime & Police Shootings	police, cop, people, kill, officer, shoot, death
Economy & Finance	money, tax, pay, people, government, wage, year
Security & Terrorism	people, war, kill, american, isis, fight, country
Technology	apple, watch, good, phone, time, car, people
Politics	people, time, hillary, country, vote, good, party

6.2 H2: Topics and Self-Disclosure

In order to validate our hypothesis about the effect of topics on self-disclosure, we first pre-processed our dataset for topical extraction. We performed topical modeling of users' comments using a Python wrapper[6] for LDA mallet. To get an optimal number of topics, multiple topic models were created with number of topics varying from 1 to 200. We used coherence score [28] as a measure of quality and interpretability of the topic models. Subsequent analysis revealed the best topic model with 20 topics and coherence score of 0.358. In this topic model, topics on related discussions were merged. For instance, two topics related to politics were merged into one. In this way, we merged similar topics to obtain 15 clear and distinct topics. Examples of some topics are shown in Table 3. Each news article with the associated comments were assigned the most dominant topic from topic model.

We used Kruskal-Wallis H test [18] to examine if users disclose about themselves differently for different topics. Kruskal Wallis H test is a non-parametric version of one way ANOVA and therefore, it is applicable to test our data (non-normal) for differences in proportion of self-disclosing users across 15 topics. Results showed that there was a statistically significant difference in proportion of self-disclosing users between the different topics of discussion [$\chi^2(16) = 29.928$, $p < 0.05$], confirming our hypothesis. Statistical significant differences were also observed for topics in proportion of users disclosing birthday/age, location, affiliation, relationships, money and experiences categories (each $p < 0.001$). We discuss the implications of these results in section 7. We refined the analysis to examine the effect of topic of discussion on the relationship between anonymity state and disclosure behavior. Multiple Mann-Whitney U tests were performed to test if the proportion of self-disclosure

[6]https://radimrehurek.com/gensim/models/wrappers/ldamallet.html

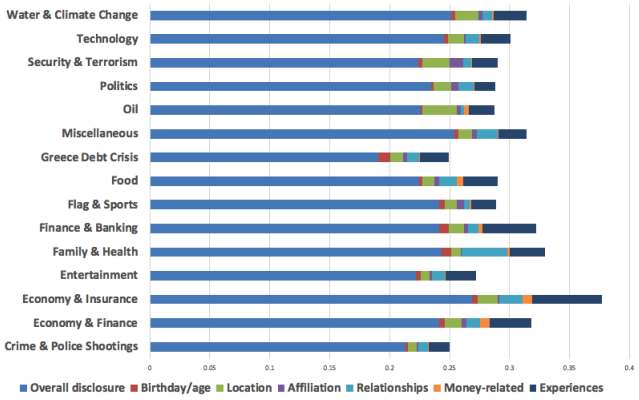


Figure 1: Average proportions of self-disclosing users per category across topics

for users differs within topics for different anonymity states. No significant differences were found after Bonferroni correction. Hence, the results show that the topic of discussion has no effect on the relationship between anonymity states and self-disclosing behavior.

7 DISCUSSION

In this study, we were able to detect language patterns of self-disclosure with differentiation into pertinent categories at reasonable accuracy levels. Because past studies on automatic detection of self-disclosure focused on supervised classification of levels of disclosure rather than unsupervised categorization of self-disclosure language [2, 7, 33], we cannot make like to like comparisons of our method's performance. Inherently, the task of detecting nuances of self-disclosure language is complex, even for humans as revealed by the agreement statistics between human annotators (See Section 5.2).

We used our categorization scheme to label users' comments in the experimental dataset. In general, *our findings confirmed our first hypothesis: users who have higher anonymity state are more likely to self-disclose than identifiable users.* The effect of anonymity as such to encourage or increase self-disclosure has been observed in social networking sites [8, 21] and our results show it is consistent even in online news commentaries.

Regarding the second contextual factor, topic of discussion, we found that proportion of self-disclosing users varied for different topics. Self-disclosing users' proportions across many individual categories also varied across topics (See Figure 1). Specifically, we observed discussions related to economy and insurance had the highest proportion of self-disclosing users. The two most relevant articles, as assigned by topic model to this topic, were centered around health-care policy. We observed many users shared their experiences and financial information, e.g. regarding health insurance, and note that this is reflected in the high proportion of users disclosing experiences and money-related information. Disclosures of specific categories in the context of certain topics are intuitive. For instance, highest proportion of users disclosed their relationships in the context of discussions about family and health. While we did not pursue specific hypotheses related to patterns of disclosure across topics, the findings here warrant further research in this regard. These results suggest the importance of context provided by the topic of discussion [33] of self-disclosure in SNS.

Our study on self-disclosure behavior has several implications. First, the results show that the online dis-inhibition actively drives responses from users on online news commentaries. Users utilize the affordances of anonymity provided by the online platform to exercise self-expression, especially subjective opinions and feelings. Hence, there are design implications for platforms to afford some levels of anonymity to maintain users' intimate participation in expression of opinions and feelings.

Second, the use of automated methods like ours to detect personal information in public comments highlights potentially hidden privacy threats. Users may be aware of the risks of disclosing too much in single comments (even with relative anonymity) and yet may be oblivious to the use of longitudinal aggregation of disclosed information. With close to 60,000 public comments over about 22k users, aggregated disclosures for each user revealed multiple categories of personal information. The ratio of number of anonymous users to number of identifiable users disclosing personal information increased with increase in number of categories disclosed. We found that higher number of users (48) disclosed four or more categories of personal information as compared to identifiable users (34). For example, multiple users disclose information about birthday/age and location. This can lead to unique identification, even if the users sign themselves with a truly random id. If such detection methods are carried out at larger scales in publicly available data (like public comments), we believe user profiles with sensitive information can be created and even anonymous users may be identified. Beyond loss of anonymity, aggregated personal information can be used by adversaries for malicious purposes.

8 CONCLUSIONS

In this study, we presented a novel method of detecting self-disclosure within commenting platforms online. We then examined two contextual factors contributing towards self-disclosure in online news commentaries: anonymity and topic of discussion. Using a dataset of user comments on online news commentaries, we tested and confirmed our hypotheses regarding these contextual factors. Specifically, our findings showed that anonymity elicits self-disclosure in online public comments. Additionally, context as specified by topic of discussion results in more likelihood of self-disclosure. Lastly, we provide implications of our results on online privacy. Self-disclosure behavior exhibited over time is perilous to user's privacy.

We see this work as a first step toward a more comprehensive study on online self-disclosure and the context within which it occurs. Our approach on automated self-disclosure detection is limited by the category-specific dictionaries. Also, albeit comprehensive, our taxonomy on self-disclosure categories is not exhaustive; there are several other categories of personal information revealed by online users. Further research is warranted in these areas.

ACKNOWLEDGEMENTS

We thank J. Barua, D. Patel and V. Goyal for the data used in this study. This work was supported by the National Science Foundation under grant 1453080.

REFERENCES

[1] Olga Abramova, Amina Wagner, Hanna Krasnova, and Peter Buxmann. 2017. *Understanding Self-Disclosure on Social Networking Sites-A Literature Review.* Technical Report. Darmstadt Technical University, Department of Business Administration, Economics and Law, Institute for Business Studies (BWL).

[2] JinYeong Bak, Chin-Yew Lin, and Alice Oh. 2014. Self-disclosure topic model for classifying and analyzing Twitter conversations. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* 1986–1996.

[3] Jin Yeong Bak, Suin Kim, and Alice Oh. 2012. Self-disclosure and relationship strength in Twitter conversations. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2.* Association for Computational Linguistics, 60–64.

[4] Azy Barak and Orit Gluck-Ofri. 2007. Degree and reciprocity of self-disclosure in online forums. *CyberPsychology & Behavior* 10, 3 (2007), 407–417.

[5] Jayendra Barua, Dhaval Patel, and Vikram Goyal. 2015. TiDE: Template-Independent Discourse Data Extraction. In *International Conference on Big Data Analytics and Knowledge Discovery.* Springer, 149–162.

[6] Natalya N Bazarova and Yoon Hyung Choi. 2014. Self-disclosure in social media: Extending the functional approach to disclosure motivations and characteristics on social network sites. *Journal of Communication* 64, 4 (2014), 635–657.

[7] Aylin Caliskan Islam, Jonathan Walsh, and Rachel Greenstadt. 2014. Privacy detective: Detecting private information and collective privacy behavior in a large social network. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society.* ACM, 35–46.

[8] Xi Chen, Gang Li, YunDi Hu, and Yujie Li. 2016. How anonymity influence self-disclosure tendency on sina weibo: An empirical study. *The Anthropologist* 26, 3 (2016), 217–226.

[9] Yoon Hyung Choi and Natalya N Bazarova. 2015. Self-disclosure characteristics and motivations in social media: Extending the functional model to multiple social network sites. *Human Communication Research* 41, 4 (2015), 480–500.

[10] Paul C Cozby. 1973. Self-disclosure: a literature review. *Psychological bulletin* 79, 2 (1973), 73.

[11] Munmun De Choudhury and Sushovan De. 2014. Mental Health Discourse on reddit: Self-Disclosure, Social Support, and Anonymity.. In *ICWSM.*

[12] Disqus. 2018. Disqus Login Page. https://disqus.com/

[13] Kilem Gwet. 2002. Kappa statistic is not satisfactory for assessing the extent of agreement between raters. *Statistical methods for inter-rater reliability assessment* 1, 6 (2002), 1–6.

[14] Erin E Hollenbaugh and Marcia K Everett. 2013. The effects of anonymity on self-disclosure in blogs: An application of the online disinhibition effect. *Journal of Computer-Mediated Communication* 18, 3 (2013), 283–302.

[15] David J Houghton and Adam N Joinson. 2012. Linguistic markers of secrets and sensitive self-disclosure in twitter. In *System Science (HICSS), 2012 45th Hawaii International Conference on.* IEEE, 3480–3489.

[16] Adam N Joinson. 2001. Self-disclosure in computer-mediated communication: The role of self-awareness and visual anonymity. *European journal of social psychology* 31, 2 (2001), 177–192.

[17] Soo-Min Kim and Eduard Hovy. 2006. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text.* Association for Computational Linguistics, 1–8.

[18] William H Kruskal and W Allen Wallis. 1952. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association* 47, 260 (1952), 583–621.

[19] Alex Leavitt. 2015. This is a throwaway account: Temporary technical identities and perceptions of anonymity in a massive online community. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing.* ACM, 317–327.

[20] Cong Liao, Anna Squicciarini, Christopher Griffin, and Sarah Rajtmajer. 2016. A hybrid epidemic model for deindividuation and antinormative behavior in online social networks. *Social Network Analysis and Mining* 6, 1 (2016), 13.

[21] Xiao Ma, Jeff Hancock, and Mor Naaman. 2016. Anonymity, intimacy and self-disclosure in social media. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems.* ACM, 3857–3869.

[22] Xiao Ma, Jeffrey T Hancock, Kenneth Lim Mingjie, and Mor Naaman. 2017. Self-Disclosure and Perceived Trustworthiness of Airbnb Host Profiles.. In *CSCW.* 2397–2409.

[23] Philipp K Masur and Michael Scharkow. 2016. Disclosure management on social network sites: Individual privacy perceptions and user-directed privacy strategies. *Social Media+ Society* 2, 1 (2016).

[24] Myriam D Munezero, Calkin Suero Montero, Erkki Sutinen, and John Pajunen. 2014. Are they different? Affect, feeling, emotion, sentiment, and opinion detection in text. *IEEE transactions on affective computing* 5, 2 (2014), 101–111.

[25] Melanie Nguyen, Yu Sun Bin, and Andrew Campbell. 2012. Comparing online and offline self-disclosure: A systematic review. *Cyberpsychology, Behavior, and Social Networking* 15, 2 (2012), 103–111.

[26] Daniel M Oppenheimer, Tom Meyvis, and Nicolas Davidenko. 2009. Instructional manipulation checks: Detecting satisficing to increase statistical power. *Journal of Experimental Social Psychology* 45, 4 (2009), 867–872.

[27] Hua Qian and Craig R Scott. 2007. Anonymity and self-disclosure on weblogs. *Journal of Computer-Mediated Communication* 12, 4 (2007), 1428–1451.

[28] Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining.* ACM, 399–408.

[29] Zick Rubin. 1975. Disclosing oneself to a stranger: Reciprocity and its limits. *Journal of Experimental Social Psychology* 11, 3 (1975), 233–260.

[30] Arthur D Santana. 2014. Virtuous or vitriolic: The effect of anonymity on civility in online newspaper reader comment boards. *Journalism Practice* 8, 1 (2014), 18–33.

[31] John Suler. 2004. The online disinhibition effect. *Cyberpsychology & behavior* 7, 3 (2004), 321–326.

[32] Michail Tsikerdekis. 2013. The effects of perceived anonymity and anonymity states on conformity and groupthink in online communities: AW ikipedia study. *Journal of the American Society for Information Science and Technology* 64, 5 (2013), 1001–1015.

[33] Yi-Chia Wang, Moira Burke, and Robert Kraut. 2016. Modeling self-disclosure in social networking sites. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing.* ACM, 74–85.

[34] Janyce M Wiebe. 1990. Identifying subjective characters in narrative. In *Proceedings of the 13th conference on Computational linguistics-Volume 2.* Association for Computational Linguistics, 401–406.

ViTOR: Learning to Rank Webpages Based on Visual Features

Bram van den Akker
University of Amsterdam
Amsterdam, The Netherlands
contact@bramvandenakker.nl

Ilya Markov
University of Amsterdam
Amsterdam, The Netherlands
i.markov@uva.nl

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

The visual appearance of a webpage carries valuable information about the page's quality and can be used to improve the performance of learning to rank (LTR). We introduce the *Visual learning TO Rank* (ViTOR) model that integrates state-of-the-art visual features extraction methods: (i) transfer learning from a pre-trained image classification model, and (ii) synthetic saliency heat maps generated from webpage snapshots. Since there is currently no public dataset for the task of LTR with visual features, we also introduce and release the ViTOR dataset, containing visually rich and diverse webpages. The ViTOR dataset consists of visual snapshots, non-visual features and relevance judgments for ClueWeb12 webpages and TREC Web Track queries. We experiment with the proposed ViTOR model on the ViTOR dataset and show that it significantly improves the performance of LTR with visual features.

KEYWORDS

Learning to rank; Visual features

ACM Reference Format:
Bram van den Akker, Ilya Markov, and Maarten de Rijke. 2019. ViTOR: Learning to Rank Webpages Based on Visual Features. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313419

1 INTRODUCTION

The design and appearance of a webpage are determining factors for a user to examine the page or to divert to another page [8, 9, 11, 17]. However, relatively little is known about the potential of visual appearance to help determine the perceived relevance of a webpage. Recently, visual features, extracted from snapshots of webpages and search engine results pages (SERPs), have been introduced into learning to rank (LTR) and have been shown to significantly improve the LTR performance [4, 18].

In this paper we continue studying LTR with visual features and propose the *Visual learning TO Rank* (ViTOR) model that integrates state-of-the-art visual features extraction methods. We present two implementations of the ViTOR model. First, we extract visual features from webpage snapshots using transfer learning and, in particular, by adopting the VGG-16 [16] and ResNet-152 [5] models pre-trained on ImageNet. Second, we introduce a novel set of visual features extracted from synthetic saliency heatmaps, which explicitly model how users view webpages [14].

Currently, there is no dataset available to support research on LTR with visual features. Fan et al. [4] experimented with webpages from the GOV2 collection.[1] However, GOV2 solely contains webpages within the .gov domain, i.e., pages with a relatively narrow scope, and, more importantly, these webpages do not contain their original images and styles. Hence, the GOV2 collection cannot fully support research on LTR with visual features, as it does not reflect visually rich and diverse webpages found on the internet today.

To overcome this issue, we release the ViTOR dataset that contains webpages from the ClueWeb12 collection[2] and queries from the TREC Web Tracks 2013 & 2014 [1, 2]. For each webpage of ClueWeb12 that also appears in the web tracks, the ViTOR dataset contains a snapshot and a set of content features, such as BM25 and PageRank. The introduced dataset is made publicly available.[3]

To assess the performance of the proposed ViTOR model, we run experiments on the introduced ViTOR dataset. Our experiments confirm that visual features significantly improve the LTR performance, which is inline with previous findings [4]. We also show that both implementations of the ViTOR model, namely transfer learning (with VGG-16 and ResNet-152) and synthetic saliency heatmaps, significantly outperform other LTR methods with visual features.

In summary, the main contributions of this work are: (i) We introduce transfer learning to extract visual features for LTR. (ii) We introduce synthetic saliency heatmaps as a novel set of LTR features. (iii) We introduce and publish ViTOR, an out-of-the-box dataset for LTR with visual features.

2 RELATED WORK

In this section, we discuss related research on the usage of visual information in LTR and the relation between visual appearance and user perception of webpages.

Using eye-tracking, Nielsen [9] and Pernice [11] demonstrate that webpage design and content placement influences the ability of users to find information they are looking for. Both studies show that by organizing the content in certain shapes (e.g., an F-shape), information can be navigated more efficiently. Wang et al. [17] show that the size of the fixation areas measured using eye-tracking is larger on webpages with more content, which increases the likelihood that the attention of a user is distracted. Such studies highlight the importance of the visual appearance of a webpage and its effect on how users perceive pages, demonstrating that visual information has to be taken into account when ranking webpages.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313419

[1] http://ir.dcs.gla.ac.uk/test_collections/gov2-summary.htm
[2] https://lemurproject.org/clueweb12/
[3] https://github.com/Braamling/learning-to-rank-webpages-based-on-visual-features/blob/master/dataset.md

Zhang et al. [18] consider using visual features for LTR and, specifically, for learning to re-rank. The authors propose a multi-modal architecture for re-ranking snippets on a SERP by learning their visual patterns. This work demonstrates that combining both visual and non-visual features can improve re-ranking performance.

The closest work to ours is the study by Fan et al. [4], which uses visual information to rank webpages instead of re-ranking snippets within an existing ranking. The authors use snapshots of webpages to extract visual features for LTR and show that such visual features significantly improve retrieval performance. They feed snapshots through a neural network that attempts to model the previously mentioned F-shape. The output of this neural network is then concatenated with more traditional ranking signals, such as BM25 and PageRank. Finally, the proposed model (called ViP) is trained end-to-end using a pairwise loss. In our paper, we continue this line of research and propose the ViTOR model for LTR with visual features that makes use of the state-of-the-art visual feature extraction techniques and shows superior performance compared to ViP. Also, we develop and publish the ViTOR dataset that contains visually diverse webpages compared to the GOV2 dataset used in [4], which lacks visual diversity and, importantly, does not contain images and style information together with webpages.

3 ViTOR MODEL

In this section, we introduce the ViTOR model for LTR with visual features. The proposed model consists of three parts. First, we introduce the model architecture in Section 3.1. Then, in Section 3.2 we describe two visual feature extractors used by ViTOR: VGG-16 [16] and ResNet-152 [5], both pre-trained on ImageNet. Finally, in Section 3.3 we propose to enhance ViTOR by generating synthetic saliency heatmaps for each of the input images. The implementation of the proposed ViTOR model is available online.[4]

3.1 Architecture

The ViTOR architecture is visualized in Figure 1. The process starts by taking an image x_i (1) as an input to the visual feature extraction layer (2) in order to create a generic visual feature vector x_{vf}. These features are considered generic because they can be extracted using convolutional filters trained on a different dataset and task. In order to transform generic visual features into LTR specific visual features, we use x_{vf} as an input to the visual feature transformation layer (3). This visual feature transformation layer outputs a visual feature vector x_{vl} that can be used in combination with other LTR features.

Separating the visual feature extraction and transformation layers allows us to significantly reduce the computational requirements when using a pre-trained model for visual feature extraction. In Section 7.1 we further elaborate on how the computational requirements can be reduced. In Section 3.2 below, we demonstrate how transfer learning can be applied to use pre-trained visual feature extraction methods in combination with the ViTOR architecture.

The ViTOR architecture also makes use of content features x_c (4), e.g., BM25, PageRank, etc. The final feature vector x_l is constructed by concatenating the visual features x_{vl} with the content features x_c. This final feature vector is then used as an input to the scoring component (5), which transforms the features into a single score x_s for each query-document pair. The resulting model is trained end-to-end using a pairwise hinge loss with L_2 regularization similarly to [4]. The scoring component uses a single fully connected layer with a hidden size of 10 and dropout set to 10%, which showed good performance in preliminary experiments.

[4] https://github.com/Braamling/learning-to-rank-webpages-based-on-visual-features

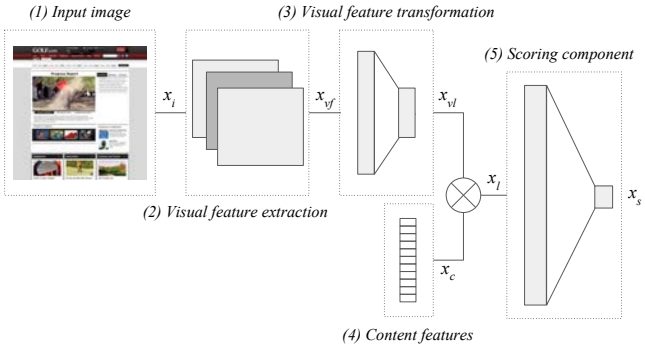


Figure 1: ViTOR architecture.

3.2 Visual feature extractors

In order to use webpage snapshots in LTR, the snapshots have to be converted to a vector representation that can then be used in combination with existing content features. Since preparing training data for LTR is costly and its amount is usually low, it is beneficial to use a visual feature extraction method that is already pre-trained. In this work we use the VGG-16 [16] and ResNet-152 [5] models, two well-explored image classification models which both have an implementation with ImageNet pre-trained parameters. Below, we describe how these two models are implemented within the ViTOR architecture.

VGG-16 [16] is commonly used for training transfer-learned models, because it provides a reasonable trade-off between effectiveness and simplicity [14]. Its architecture consists of a set of convolutional layers and fully connected layers. The convolutional layers extract features from an input image, which are then used by the fully connected layers to classify the image. The convolutional layers of VGG-16 are generic with respect to an input and task [3] and, thus, can be used as a visual feature extractor within the ViTOR architecture to create generic visual features x_{vf}. Hence, we use the convolutional layers as is, by freezing all the parameters during training. Because we do not alter any of the convolutional layers, the size of x_{vf} is determined by the output of the convolutional layers in the original VGG-16 model, being 1×25088.

The fully connected layers of VGG-16, instead, can be altered and retrained in order to be used with new inputs and tasks. Due to this, we utilize them as a visual feature transformation layer within the ViTOR architecture to produce LTR specific features x_{vl}. In particular, we replace the last fully connected layer of VGG-16 by a newly initialized fully connected layer. Then we optimize the parameters of all fully connected layers of VGG-16 during training. The size of x_{vl} is set to 30, as this size showed good performance in preliminary experiments.

The ResNet-152 [5] architecture was shown to outperform VGG-16 in ImageNet classification. The residual connections between convolutional layers of ResNet-152 allow for deeper networks to

be trained without suffering from vanishing gradients. Similarly to VGG-16, ResNet-152 has convolutional layers that extract features from an input image, which are in turn used by a fully connected layer to classify each image. We use these convolutional layers as the visual feature extraction layer, which transforms x_i to x_{vf}. All parameters of these convolutional layers are frozen during training. As with VGG-16, the size of x_{vf} is determined by the output size of the original convolutional layers in the ResNet-152 model, 1×2048.

Additionally, the original ResNet-152 architecture only has a single fully connected layer, which empirically showed to not be enough to successfully train the ViTOR model. Instead, we transform x_{vf} to x_{vl} by training a fully connected network from scratch. The transformation layer is constructed using three layers with each having 4096 hidden units and a final layer resulting in x_{vl} with a size of 30, which was empirically found to provide good performance in preliminary experiments.

3.3 Saliency heatmaps

In order to increase the ability to learn the visual quality of a webpage, we propose to explicitly model the user viewing pattern through synthetic saliency heatmaps. The use of saliency heatmaps could be advantageous compared to the use of raw snapshots for the following reasons. First, synthetic saliency heatmaps explicitly learn to predict how users perceive webpages by training an end-to-end model on actual eye-tracking data. We expect this information to better correlate with webpage relevance compared to raw snapshots. Second, saliency heatmaps reduce the average storage requirements by up to 90%, because they are gray-scale images and have large areas of the same color, which can be stored efficiently. This makes the use of saliency heatmaps attractive for practical applications. Figure 2 shows example snapshots with their corresponding heatmaps (first and third columns respectively).

Following [14], we use a two-stage transfer learning model that learns how to predict saliency heatmaps on webpages. Similarly to the visual feature extraction approaches above, [14] takes a pre-trained image recognition model and finetunes the output layers on the following two datasets in order respectively: (i) SALICON [6], a large dataset containing saliency heatmaps created with eye-tracking hardware on natural images, and (ii) the webpage saliency dataset from [15], a smaller dataset containing saliency heatmaps created with eye-tracking hardware on webpages.

The trained model is used to convert a raw snapshot into a synthetic saliency heatmap. This heatmap is then used as an input image x_i for the ViTOR model (see Figure 1).

4 ViTOR DATASET

In this section, we introduce the ViTOR dataset for LTR with visual features. Section 4.1 contains information about the underlying ClueWeb12 collection and TREC Web Track topics. Section 4.2 explains how the snapshots for ClueWeb12 are acquired. Section 4.3 discusses content features, such as BM25 and TF-IDF, included in the ViTOR dataset. Finally, Section 4.4 gives an overview of the structure in which the ViTOR dataset is presented and published.

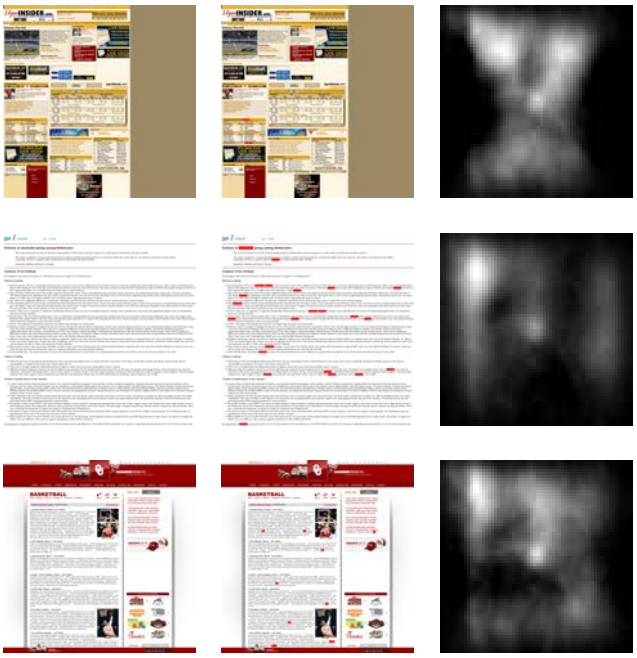


Figure 2: Examples of a vanilla snapshot, a red highlighted snapshot, and a saliency heatmap from left to right, respectively.

4.1 ClueWeb12 & TREC Web Track

For the ViTOR dataset we choose to use a combination of the ClueWeb12 document collection and the topics from the TREC Web Tracks 2013 & 2014 [1, 2], because this is currently the most recent combination of a large-scale webpage collection together with judged queries (with graded relevance).

ClueWeb12 is a highly diverse collection of webpages scraped in the first half of 2012. The total collection contains over 700 million webpages. We only use ClueWeb12 webpages that have judgements for any of the 100 queries in the TREC Web Tracks 2013 & 2014. In total, there are 28,906 judged webpages. Table 1 shows the breakdown of the total number of webpages and different relevance labels in the combined set of topics from 2013 and 2014.

Table 1: Number of webpages per source and the corresponding breakdown of TREC Web Track relevance grades.

Count/Label	TREC Web	Wayback	ClueWeb12	No image
Total	28,906	23,249	5,392	265
Nav grade (4)	40	36	4	0
Key grade (3)	409	347	62	0
Hrel grade (2)	2,534	2,222	295	17
Rel grade (1)	6,832	5,679	1,123	30
Non grade (0)	18,301	14,395	3,701	205
Junk grade (-2)	790	570	207	13

4.2 Snapshots

Although each entry in the ClueWeb12 collection contains the webpage's HTML source, many pages lack styling and images files

in order to render the full page. To overcome this issue, we use the Wayback Machine,[5] which offers various archived versions of webpages with styling and images since 2005. For each page in ClueWeb12, that is also judged in the TREC Web Tracks 2013 & 2014, we scrape an entry on the Wayback Machine that is closest to the original page scrape date as recorded in ClueWeb12. A snapshot is then taken using a headless instance of the Firefox browser. To reproduce [4], we also create a separate query-dependent dataset with the same snapshots where all query words are highlighted in red (HEX value: #ff0000). Examples of snapshots and snapshots with highlights are shown in Figure 2 (first two columns).

Since the Wayback Machine does not contain an archived or working version of each webpage in the ClueWeb12 collection, a filtering process is introduced to maximize the quality of each snapshot. Using the following criteria, a snapshot is selected for each available webpage: (1) Each webpage is requested from the Wayback Machine. (2) A webpage that is not on the Wayback Machine, times out, throws a JavaScript error, or results in a PNG snapshot smaller than 100KB is marked as broken. Such webpages are rendered again using the online rendering service provided by ClueWeb12.[6] (3) A manual selection is made between all webpages that are rendered from both sources. The Wayback version is used if it contains more styling elements and if the content is the same as in the rendering service. Otherwise, the rendering service version is used.

As a result, most of the 28,906 judged webpages have a corresponding snapshot from either the Wayback Machine or ClueWeb12 rendering service. Only 265 documents did not pass the filtering and were discarded. Table 1, row one, summarizes the results of the ViTOR dataset acquisition process. The table also shows the distribution of judgment scores among the snapshots that were taken from the Wayback Machine and ClueWeb12 rendering service.

4.3 Non-visual features

In LTR, documents are ranked based on various types of features, such as content features (e.g., BM25), quality indicators (e.g., PageRank) and behavioral features (e.g., CTR). In order to use the ViTOR dataset to measure the effect of visual features, we also add a set of content features and quality indicators, listed in Table 2. These 11 features are chosen so that they resemble most informative features of the most recent LETOR 4.0 dataset [12] and are easy to compute (see Section 7.2 for a detailed comparison of different feature sets). Also note that behavioral features, such as CTR, are not available for the TREC Web Tracks.

The content features, such as TF-IDF and BM25, are computed by doing a full pass over the complete ClueWeb12 collection. The PageRank scores are taken from the ClueWeb12 Related Data section.[7] The following modifications based on the feature transformations described in LETOR are made to stabilize training: (1) Free parameters k_1, k_3 and b for BM25 were set to 2.5, 0 and 0.8, respectively. (2) Since the PageRank scores are usually an order of magnitude smaller than all other scores, we multiply them by 10^5. (3) The log transformation is applied to each feature. (4) The log-transformed features are normalized per query.

[5] http://archive.org/web/
[6] http://boston.lti.cs.cmu.edu/Services/
[7] https://lemurproject.org/clueweb12/related-data.php

Table 2: Non-visual features provided with the ViTOR dataset.

Id	Description	Id	Description	Id	Description
1	Pagerank	5	Content TF-IDF	9	Title IDF
2	Content length	6	Content BM25	10	Title TF-IDF
3	Content TF	7	Title length	11	Title BM25
4	Content IDF	8	Title TF		

4.4 Final collection

In summary, the ViTOR dataset contains: (i) a directory with webpage snapshots (Section 4.2), and (ii) a set of files with content features divided into folds (Section 4.3). Each snapshot is stored as a PNG file that can be identified by its corresponding ClueWeb12 document id. The non-visual features are stored in LETOR formatted files containing the raw, logged and query normalized values. The query normalized values are randomly split per query into five equal partitions. These partitions are then used to create five folds, where each fold contains three partitions for training and the remaining two partitions for validation and testing.

5 EXPERIMENTAL SETUP

In this section, we discuss the configurations of the ViTOR architecture, baselines and metrics used during our experiments.

ViTOR configurations.. We experiment with four configurations of the ViTOR architecture. ViTOR baseline refers to the ViTOR model with only content features. This configuration is trained by feeding content features into the scoring component directly, without adding any visual features. ViTOR snapshots and ViTOR highlights use visual features extracted from vanilla snapshots of webpages and from snapshots of webpages with highlighted query terms, respectively. Finally, ViTOR saliency uses visual features extracted from synthetic saliency heatmaps. The snapshots, highlights and saliency heatmaps for each model respectively are used as the input image (component (1) of Figure 1. For the ViTOR configurations with visual features, we experiment with both VGG-16 and ResNet-152 visual feature extraction methods. The learning rates for VGG-16 and ResNet-152 are set to and 0.0001 and 0.00005, respectively. Each experimental run is generated using the Adam optimizer [7] with a batch size of 100.

Baselines. We compare our proposed ViTOR model to the ViP model by Fan et al. [4], the only existing LTR method that uses visual features. We train ViP on both vanilla and highlighted snapshots with the resulting configurations being ViP snapshots and ViP highlights, respectively. Following [4], we also compare the ViTOR model to a number of content-based ranking methods, namely BM25 and state-of-the-art LTR techniques, such as RankBoost, AdaRank, and LambdaMart.[8]

Metrics. To measure the retrieval performance, we use precision and ndcg at $\{1, 10\}$ and MAP. Statistical significance is determined using a two-tailed paired t-test (p-value ≤ 0.05).

6 RESULTS

In this section, we present experiments that are set out to test the following: (1) the ViTOR model improves the LTR performance

[8] The LTR methods are taken from https://sourceforge.net/p/lemur/wiki/RankLib.

Table 3: Results for the ViTOR model using only content features (baseline), vanilla snapshots, highlighted snapshots, and saliency heatmaps. All results significantly improve over the ViTOR baseline. Best results are shown in bold.

	p@1	p@10	ndcg@1	ndcg@10	MAP
ViTOR baseline	0.338	0.370	0.189	0.233	0.415
VGG snapshots	0.514	0.484	0.292	0.324	0.442
ResNet snapshots	0.550	0.452	0.310	0.301	0.437
VGG highlights	**0.560**	**0.520**	0.323	**0.346**	**0.456**
ResNet highlights	0.530	0.463	0.305	0.312	0.440
VGG saliency	0.554	0.453	0.310	0.302	0.422
ResNet saliency	**0.560**	0.476	**0.333**	0.321	0.442

when introducing visual features, (2) synthetic saliency heatmaps improve the LTR performance when used as an input the ViTOR model, and (3) the ViTOR model improves both visual and non-visual state-of-the-art ranking methods.

6.1 ViTOR model with VGG-16 and ResNet-152

In Table 3, we compare the performance of the ViTOR model when used with and without visual features. The first row shows the ViTOR baseline, when using only the content features as an input to the scoring component. The second to fifth rows show the performance of using VGG-16 and ResNet-152 with both vanilla and highlighted snapshots. These results clearly show that both VGG-16 and ResNet-152 visual feature extraction methods significantly improve the performance compared to the ViTOR baseline.

When comparing the results of the ViTOR model with visual features, we observe the following: (i) The highest ranking performance is achieved by using VGG-16 on the highlighted snapshots. (ii) For VGG-16, the values of all metrics are consistently better for highlighted snapshots compared to vanilla snapshots, which is in line with the findings of [4] and is to be expected: highlighted snapshots carry more information compared to vanilla snapshots. Based on these results, we conclude that the use of visual features in LTR significantly improves performance and that highlighted snapshots should on average be preferred over vanilla snapshots.

6.2 ViTOR model with saliency heat maps

The last two rows of Table 3 show the performance of the ViTOR model when using synthetic saliency heat maps as an input. The visual features are learned using both VGG-16 and ResNet-152. In this case, ResNet-152 consistently outperforms VGG-16. Although the highlighted snapshots with VGG-16 still outperform ResNet-152 with saliency heat maps on p@10, ndcg@10 and MAP, the saliency heat maps with ResNet-152 match and outperform VGG-16 with highlighted snapshots when looking at p@1 and ndcg@1. Hence, saliency heat maps should be preferred in applications where early precision is important, while highlighted snapshots should be used when a high overall performance is needed.

6.3 Baseline comparison

Table 4 compares the performance of the ViTOR model to BM25, non-visual LTR methods and the ViP model by Fan et al. [4]. Specifically, the table shows the performance of VGG-16 with highlighted snapshots and of ResNet-152 with synthetic saliency heatmaps, as these are the best-performing variants of the ViTOR model according to Table 3. Both methods have a significant performance increase compared to BM25, almost doubling the metrics values in many cases.

Table 4: Results for the VGG-16 with highlighted snapshots, ResNet-152 with saliency heatmaps, and baselines. † indicates a significant decrease in performance compared to VGG highlights and ‡ indicates a significant decrease in performance compared to both ViTOR implementations. Best results are shown in bold.

	p@1	p@10	ndcg@1	ndcg@10	MAP
BM25	0.300‡	0.316‡	0.153‡	0.188‡	0.350‡
RankBoost	0.450	0.444	0.258	0.288†	0.427
AdaRank	0.290‡	0.357‡	0.149‡	0.227‡	0.398
LambdaMart	0.470	0.420†	0.256	0.275†	0.418
ViP snapshots	0.392‡	0.398‡	0.217‡	0.254‡	0.421‡
ViP highlights	0.418‡	0.416‡	0.239‡	0.269‡	0.422‡
VGG highlights	**0.560**	**0.520**	0.323	**0.346**	**0.456**
ResNet saliency	**0.560**	0.476	**0.333**	0.321	0.442

When comparing to non-visual LTR methods, both ViTOR implementations show consistently better performance. However, not all metrics are improved significantly. We attribute this to the fact that, similarly to [4], the LTR component of the ViTOR model is based on pairwise hinge loss, which is a relatively simple loss function.

Finally, we compare the ViTOR implementations to ViP, the only existing LTR method with visual features. Here, we clearly see that both our implementations significantly outperform ViP on all metrics. Also note, that ViP loses to two out of three non-visual LTR baselines, namely RankBoost and LambdaMart. We believe this is due to the reason discussed above: ViP uses pairwise hinge loss as the LTR component [4], which may be suboptimal.

The above results show that the proposed ViTOR model outperforms baselines, whether they are supervised or unsupervised, use visual features or not. However, to achieve consistent significant improvements compared to the state-of-the-art LTR methods, different loss functions within the ViTOR model have to be investigated.

7 DISCUSSION

In this section, we address two practical aspects of the proposed ViTOR model and dataset: (i) the number of parameters of the ViTOR model and corresponding optimizations, and (ii) the performance of content features included in the ViTOR dataset.

7.1 Training and inference optimization

Both training and inference using deep convolutional networks are generally computationally expensive. By separating the feature extraction and transformation layers in the proposed ViTOR architecture (see Figure 1) we allow powerful computational optimizations for both training and inference.

Training optimization. Although we freeze the parameters in the visual feature extraction layer (component (2) of Figure 1) when using a pre-trained model, we still need to compute the output

from all the frozen parameters during the forward propagation. We can avoid the computational cost associated with the forward propagation on the frozen layers by storing the output of the visual feature extraction layer to disk prior to training. By storing these vectors to disk, we leave only the parameters of the fully connected layers (component (3) of Figure 1) to be calculated and stored in memory during the forward propagation.

By applying the above procedure, we can reduce the number of parameters of the ViTOR model. When using VGG-16, the visual feature extraction layer consists of 14, 714, 688 parameters, which is 12.3% of the parameters in the ViTOR model. When using ResNet-152, the visual feature extraction layer consists of 58, 144, 239 parameters, which is 49.5% of the parameters in the ViTOR model.

By using the stored output of the visual feature extraction layer x_{vf} instead of an actual image x_i as the input of the ViTOR model, we reduce the size of each input by 84.7% and 98.6% for VGG-16 and ResNet-152 respectively. This reduction in input size further reduces the memory required for training the model.

Real-time inference optimization. When using LTR in a large-scale production environment, the impact of newly introduced LTR features (visual features, in our case) on real-time computational requirements is of major concern.

Since both the vanilla snapshots and synthetic saliency heatmaps are query independent, the output of the visual feature transformation layer (component (3) of Figure 1) does not change for different document/query pairs. This enables offline inference, leaving only the scoring component to be inferred in real-time. Since a visual feature vector is 1×30, using the offline inferred feature vector would result in a LTR model with $30 + 11 = 41$ features. This increase in parameters is negligible in terms of computational costs.

7.2 Benchmarking content features

In LTR research, both the amount and type of considered features vary widely per dataset and study. The most recent LETOR 4.0 dataset [12] contains 46 features extracted for webpages from the GOV2 dataset and queries from the million query tracks 2007 and 2008 (MQ2007 and MQ2008). In the ViTOR dataset, we use 11 features that are a subset of the 46 features of LETOR (see Table 2). These 11 features are chosen to be both informative and easy to compute. Here, we compare our subset of 11 features to the full set of 46 features. The experiments are run on the GOV2 dataset and MQ2007 queries. The LTR methods considered are the same as in Section 6, namely RankBoost, AdaRank, and LambdaMart.

The results of running the considered LTR methods using both 46 and 11 features are shown in Table 5. From these results, we see that the number of features has a significant effect on the performance of AdaRank. However, for the best-performing RankBoost and LambdaRank methods the drop in performance is minor when using 11 features instead of 46 features. This indicates that the chosen 11 features included in the ViTOR dataset form a reasonable trade-off between effectiveness and computation cost.

Table 5: Comparison of 46 LETOR features and 11 LETOR features that are also used in ViTOR.

	p@1	p@10	ndcg@1	ndcg@10	MAP
RankBoost - 46	0.453	0.371	0.391	0.430	0.457
RankBoost - 11	0.448	0.372	0.381	0.431	0.453
AdaRank - 46	0.420	0.360	0.367	0.424	0.449
AdaRank - 11	0.385	0.287	0.364	0.394	0.386
LambdaMart - 46	0.452	0.384	0.405	0.444	0.463
LambdaMart - 11	0.448	0.380	0.397	0.443	0.455

8 CONCLUSION

In this paper, we considered the problem of LTR with visual features. We proposed the ViTOR model that extracts visual features from webpage snapshots using transfer learning (specifically, pre-trained VGG-16 and ResNet-152 models) and synthetic saliency heatmaps. In both cases the extracted visual features significantly improved the LTR performance. We also showed that the proposed ViTOR model significantly outperformed the visual LTR baselines.

In addition to the model, we released the ViTOR dataset, containing visually rich and diverse webpages with corresponding snapshots. With the ViTOR dataset it is now possible to comprehensively study LTR with visual features.

One direction for future work is the study of state-of-the-art LTR methods, such as RankBoost and LambdaMart, within the scoring components of the ViTOR model. This could improve performance without changing the visual feature extraction and transformation components. Another promising direction is to combine multiple visual features, i.e., visual features extracted from vanilla snapshots, snapshots with highlights and saliency heatmaps. Other visual feature extractors, such as the CapsuleNet [13] model, which is able to learn spatial relations in images, might also provide additional performance. Other methods of combining visual and textual features might also be worth exploring. Future work could also investigate the robustness of this method when using various rendering variations, e.g., different browser, resolutions. Finally, performing a more qualitative analysis on which elements cause improvements in LTR with visual features would be interesting, e.g., by interpreting the filters in the feature extractor [10].

Code and data

Both the ViTOR dataset[9] and the code used to run the experiments in this paper[10] are available online.

Acknowledgements

The Spark experiments in this work were carried out on the Dutch national e-infrastructure with the support of SURF Cooperative. Thanks to the Amsterdam Robotics Lab for using their computational resources. Thanks to Jamie Callan and his team for providing access to the online services for ClueWeb12.

This research was partially supported by Ahold Delhaize, the Association of Universities in the Netherlands (VSNU), and the Innovation Center for Artificial Intelligence (ICAI). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

[9] https://github.com/Braamling/learning-to-rank-webpages-based-on-visual-features/blob/master/dataset.md

[10] https://github.com/Braamling/learning-to-rank-webpages-based-on-visual-features

REFERENCES

[1] Kevyn Collins-Thompson, Paul Bennett, Fernando Diaz, Charlie Clarke, and Ellen M Voorhees. 2013. TREC 2013 Web Track overview. In *TREC*. NIST.

[2] Kevyn Collins-Thompson, Craig Macdonald, Paul Bennett, Fernando Diaz, and Ellen M Voorhees. 2015. TREC 2014 Web Track overview. In *TREC*. NIST.

[3] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*. 647–655.

[4] Yixing Fan, Jiafeng Guo, Yanyan Lan, Jun Xu, Liang Pang, and Xueqi Cheng. 2017. Learning visual features from snapshots for web search. In *CIKM*. ACM, 247–256.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

[6] Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. 2015. Salicon: Saliency in context. In *CVPR*. 1072–1080.

[7] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[8] Jakob Nielsen. 1999. *Designing web usability: The practice of simplicity*. New Riders Publishing.

[9] Jakob Nielsen. 2006. F-shaped pattern for reading web content. Jakob Nielsen's Alertbox. (2006).

[10] Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. 2018. The Building Blocks of Interpretability. *Distill* (2018).

[11] Kara Pernice. 2017. F-shaped pattern of reading on the web: Misunderstood, but still relevant (Even on mobile). nngroup.com. (2017).

[12] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 Datasets. *arXiv preprint arXiv:1306.2597* (2013).

[13] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In *NIPS*. 3859–3869.

[14] Wei Shan, Guangling Sun, Xiaofei Zhou, and Zhi Liu. 2017. Two-stage transfer learning of end-to-end convolutional neural networks for webpage saliency prediction. In *IScIDE*. Springer, 316–324.

[15] Chengyao Shen and Qi Zhao. 2014. Webpage saliency. In *ECCV*. Springer, 33–46.

[16] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[17] Qiuzhen Wang, Sa Yang, Manlu Liu, Zike Cao, and Qingguo Ma. 2014. An eye-tracking study of website complexity from cognitive load perspective. *Decision Support Systems* 62 (2014), 1–10.

[18] Junqi Zhang, Yiqun Liu, Shaoping Ma, and Qi Tian. 2018. Relevance Estimation with Multiple Information Sources on Search Engine Result Pages. In *CIKM*. ACM, 627–636.

Evaluating Neural Text Simplification in the Medical Domain

Laurens van den Bercken
Delft University of Technology
myTomorrows
laurens.vandenbercken@
mytomorrows.com

Robert-Jan Sips
myTomorrows
robert-jan.sips@mytomorrows.com

Christoph Lofi
Delft University of Technology
c.lofi@tudelft.nl

ABSTRACT

Health literacy, i.e. the ability to read and understand medical text, is a relevant component of public health. Unfortunately, many medical texts are hard to grasp by the general population as they are targeted at highly-skilled professionals and use complex language and domain-specific terms. Here, automatic text simplification making text commonly understandable would be very beneficial. However, research and development into medical text simplification is hindered by the lack of openly available training and test corpora which contain complex medical sentences and their aligned simplified versions. In this paper, we introduce such a dataset to aid medical text simplification research. The dataset is created by filtering aligned health sentences using expert knowledge from an existing aligned corpus and a novel simple, language independent monolingual text alignment method. Furthermore, we use the dataset to train a state-of-the-art neural machine translation model, and compare it to a model trained on a general simplification dataset using an automatic evaluation, and an extensive human-expert evaluation.

CCS CONCEPTS

• **Information systems** → **Data extraction and integration**; • **Applied computing** → **Consumer health**; *Health care information systems*; • **Computing methodologies** → Supervised learning.

KEYWORDS

Medical Text Simplification, Test and Training Data Generation, Monolingual Neural Machine Translation

ACM Reference Format:
Laurens van den Bercken, Robert-Jan Sips, and Christoph Lofi. 2019. Evaluating Neural Text Simplification in the Medical Domain. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313630

1 INTRODUCTION

The rapid increase of health information on the internet has resulted in more patients turning to the Internet as their first source of health information. In a recent structural review, Tan and Goonawardene found that patients consult the internet primarily to be actively involved in the decision making related to their health [29]. While finding more evidence of positive impact on decision making and the patient-physician relation from online health information searches, Tan and Goonawardene warn that information found on the internet "has the potential to misguide patients and make them excessively anxious". Moreover, they observe that discrepancies between a physician's advice and a patient's conclusions from online information can erode the patient-physician relationship, which could limit timely access to care.

In a 2018 report [31] by the World Health Organization, contextualize these warnings, by concluding that a majority of European citizens has insufficient health literacy (the ability to read and understand healthcare information, make appropriate health decisions and follow health instructions). The report stipulates that one of the major causes of this low health literacy is that health information is often inaccessible to the general public because the literacy demands of health information and the literacy skills of average adults are mismatched, the information is often poorly written, poorly designed and/or geared to an highly sophisticated audience. One way to improve health literacy is by simplifying (online) medical text, to match the literacy level and vocabulary of average adults.

To illustrate this, we provide an example from Wikipedia, and its simplified version from Simple Wikipedia below.

- Pituitary adenomas represent from 10% to 25% of all intracranial neoplasms and the estimated prevalence rate in the general population is approximately 17%.
- Pituitary adenomas represent from 10% to 25% of all brain tumors and is thought to happen in about 17% to 25% of most people.

Observe that the Wikipedia editors simplify text on two levels: (1) complicated medical terminology (*intracranial neoplasms*) is replaced by simpler terms and (2) complicated non-medical sentence structures (*estimated prevalence rate in the general population*) are simplified. There is an rapidly increasing body of online medical texts, such as electronic health records, clinical trials, medical research, drug labels and patient information leaflets. This rapid increase makes manual simplification unfeasible.

Therefore, in this paper, we focus on the following research question:

> **RQ**: To what extent can we use automated methods to simplify expert level health text to laymen level?

Current work in automated medical text simplification is mostly limited to simplifying medical terminology, either by the generation of explanations (explanation generation), or by replacing these terms with laymen terms or definitions (lexical simplification) [1, 5–8, 24, 25]. This ignores complex non-medical terms and

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313630

complicated sentence structures, which also hamper readability [33]. The state-of-the-art in automated text simplification, Neural Machine Translation [22, 28], shows promise to solve this second problem, but requires large parallel corpora for training, which are lacking in the medical domain. Recent work by Adduru et al. focused on the creation of such a medical text simplification corpus [2]. Unfortunately, the resulting set is not publicly available.

Original Contribution As there are no publicly available medical text simplification corpora, we create a new aligned corpus by semi-automatically filtering a set of aligned health-related sentences from an existing parallel text simplification corpus. In addition, we introduce a language independent monolingual text alignment method and use it for aligning additional health sentences from Wikipedia and Simple Wikipedia disease articles. The resulting data set is made publicly available for future research[1]. Furthermore, we propose a method using state-of-the-art Neural Machine Translation for medical text simplification, which can both simplify general text but also learns to translate medical concepts into simpler terms. We perform experiments with that method on the filtered health sentences. We report results of quantitative evaluations, and a qualitative one.

2 RELATED WORK

In this section we discuss relevant work on text simplification in the medical domain. We first discuss lexical simplification, text simplification which focuses on the replacement of complex terms. Secondly, we discuss syntactic simplification: simplification which focuses on the replacement of complicated sentence structures and conclude with combined approaches: text simplification which targets both the replacement of complicated terms and the replacement of complicated structures.

Lexical simplification Most work on medical lexical simplification is focused on the usage of large vocabularies, most prominently the Unified Medical Language System (UMLS) [4] to replace expert medical terms with consumer oriented synonyms. The UMLS is a meta-thesaurus, which contains unified entities from a large number of medical vocabularies (such as SnomedCT [9], MeSH [18] and CHV [37]). In summary, the state-of-the-art in lexical simplification is recognizing UMLS concepts from text and replacing them with a consumer-oriented synonym from the Consumer Health Vocabulary [37], an open access collection of consumer oriented synonyms for medical concepts. Despite significant efforts to automatically enrich and correct the Consumer Health Vocabulary from user-generated data [11, 13, 30, 36], evaluation of the effect of using these terms on the perceived simplicity of medical text by the lay population is lacking and recent work by Xie et al. articulates that medical concept replacement alone is not sufficient for medical text simplification [33].

Syntactic simplification There is little work investigating medical syntactic simplification in isolation. Leroy et al. investigate the (manual) splitting of complex noun phrases to improve readability of long sentences. However, they conclude that this approach does not necessarily improve readability and recommend that sentences should only be split when the split phrases "feel more natural" [17]. Furthermore, negations in medical texts were investigated and it was shown that easier text contains less morphological negations than difficult text [21]. An easy text contains for example "not clear" instead of "unclear", which could effectively be solved by a lexical simplification tool based on frequency analysis.

Combined approaches Monolingual machine translation, i.e. machine translation algorithms trained on a parallel corpus in the same language, have shown great promise in recent years. Such systems learn how to translate complex language into simple language, when trained of a parallel corpus of complex and simple sentences. In theory, such a translation combines lexical and syntactic simplification. Most prominent is the progress in Neural Machine Translation [22], which has demonstrated to achieve state-of-the-art performance on text simplification tasks for common language. Neural Machine Translation relies on the availability of a large parallel corpus for training and evaluation purposes. For common language, publicly available corpora are available from aligned Wikipedia and Simple Wikipedia sentences, the Parallel Wikipedia Simplification (PWKP) corpus [38] and a more recent corpus presented Hwang et al. [12] and from news articles, the Newsela corpus [34].

Algorithms trained on these datasets perform well on general language simplification, but have been shown to perform poorly on medical text simplification [2, 16]. For instance, an (statistical) algorithm trained on the PWKP dataset for simplifying cancer and other health text produced output that was "imperfect and required a second manual step to be consistent and accurate" [16].

To successfully employ Neural Machine Translation on health text, we would need a health specific parallel corpus. Unfortunately such a corpus is not available and, a first attempt by Adduru et al. to creating one [2] showed that this is not as easy as it seems. Adduru et al. used an array of methods to automatically align sentences from the medical subset of Wikipedia and Simple Wikipedia, as well as https://www.webmd.com and https://www.medicinenet.com. The result is a -proprietary- medical text simplification corpus of 2,493 sentence pairs. Adduru et al. present an automated evaluation of a Neural Machine Translation algorithm on these data, but do not include an human evaluation.

3 DATA

In this section we present two datasets we created for text simplification in the medical domain. The first dataset (EXPERT) is an expert-evaluated medical subset filtered from the aligned wikipedia corpus presented by Hwang et al. [12]. It is focusing on reliable high-quality sentence alignments such that it can be used as a test set for benchmarking. The second dataset (AUTOMATED) is a novel dataset created by automatic collection of aligned sentences from the medical subset of Wikipedia. Here, the focus lies on having a large dataset which can serve as training data, but we accept smaller losses in quality resulting from the automatic alignment. In table 1, we provide a summary of the presented datasets, compared to the medical corpus presented by Adduru et al. [2].

3.1 EXPERT dataset

Our EXPERT dataset is created using the aligned corpus presented in [12] as a baseline, which aligns sentences between Wikipedia and SimpleWikipedia. As the corpus does not focus on a particular

[1] available at https://research.mytomorrows.com/

	Fully aligned	Partially aligned
EXPERT	2,267	3,148
AUTOMATED	3,797	-
Adduru et al. [2]	2,493	-

Table 1: An overview of the datasets

domain, only few medical sentences are covered which motivates the creation of our EXPERT dataset. This initial corpus consists of manually and automatically generated *good* and *good partial* aligned sentence pairs, the former defined as "the semantics of the simple and standard sentence completely match, possibly with small omissions (e.g., pronouns, dates, or numbers)" and the latter as "a sentence completely covers the other sentence, but contains an additional clause or phrase that has information which is not contained within the other sentence". In the remainder of the paper we will refer to the *good* sentence pairs as *fully aligned* and to the *good partial* as *partially aligned* sentence pairs.

To generate the EXPERT dataset, we use a state-of-the-art medical named entity recognition tool, QuickUMLS [26] to sentences which may contain a medical topic from the fully aligned and partially aligned datasets. QuickUMLS is an approximate dictionary matching algorithm which matches terms from text with synonyms in the UMLS. We used QuickUMLS with the default setting for similarity threshold (0.7) and limited the semantic types to *Disease or Syndrome* and *Clinical Drug*). We consider a sentence pair a candidate medical sentence pair, when QuickUMLS recognizes at least one medical concept in either the complex or the simple medical sentence. After QuickUMLS processing, we provided the resulting candidate medical sentence pairs to a domain expert for additional validation, i.e. to confirm whether the sentence pair is indeed health-related. Using this approach, we created a filtered corpus of 2,267 *fully aligned* medical sentences and 3,148 *partially aligned* sentences.

3.2 AUTOMATED dataset

In addition to the labour-intensive manual filtering, we created a pipeline to automatically create an aligned dataset from Wikipedia and Simple Wikipedia, which allows efficient creation of a much larger dataset, at the cost of a slight loss in quality. In principle, such a pipeline has 3 distinct steps: (1) Collection of relevant articles and their related simplified version, (2) Splitting the articles in sentences and (3) Aligning the sentences into pairs.

3.2.1 Finding relevant articles. Recent work, by Kajiwara and Komachi [14] and Adduru et al. [2] focused on the creation of an aligned corpus from Wikipedia and Simple Wikipedia. The former presented a methodology to create a general corpus, the latter a medical corpus. Kajiwara and Komachi used a full dump of Wikipedia and Simple Wikipedia and aligned the articles with matching titles. Given the goal of creating a general-purpose corpus, they did not attempt to select articles based on topic. In their work, they identify a total of 126,725 Wikipedia articles with a matching Simple Wikipedia article in the English language. In contrast, Adduru et al. present an approach to collect a specific subset of medical Wikipedia articles. They manually selected a set of 164 articles, which they match to Simple Wikipedia articles with a matching title. Manual collection of such a dataset seems unnecessarily cumbersome. We propose an approach using DBPedia [3] and select all articles that fall in the *dbo: Disease* class. After title matching to Simple Wikipedia, this gives us a set of 1,098 aligned articles.

3.2.2 Splitting. Analogous to Kajiwara and Komachi, we extract the text from the Wikipedia and Simple Wikipedia articles, using the python Wikipedia API [2] and the NLTK 3.3 sentence tokenizer [3]. This gives an average number of words per sentence of 26.1 for the normal articles and 19.5 for the simple articles. The average numbers of sentences per article are 123.4 and 20.3 respectively. In comparison, Kajiwara and Komachi report an average number of words per sentence of 25.1 for the normal articles and 16.9 for the simple articles and an average number of sentences per article of 57.7 and 7.65, respectively. Medical articles (simple and normal), seems to be longer and more complex (they contain more and longer sentences).

3.2.3 Aligning. To align sentences from Wikipedia to Simple Wikipedia, we employ a two step approach: first we generate *candidate pairs*, by combining each sentence from the Wikipedia articles which each sentence of the related Simple Wikipedia article. This gives us a total of 3,660,064 candidate pairs from the 1,098 articles. Adduru et al. report 818,520 candidate pairs from 164 articles, demonstrating that their manually collected set contains longer articles than ours (3333.4 candidate pairs per article in our set versus 4991 candidate pairs per article in their set). Secondly, we select the most similar pairs from the candidate pairs. Various methods have been reported to perform this task, Kajiwara and Komachi employ pre-trained Word2Vec word embeddings to determine sentence similarity. Similarly, Hwang et al. present a method that relies on Wiktionary [12]. When aligning sentences where the distinctive (medical) terms are arguably very infrequent, such dependencies may not be wanted, as also noted by Adduru et al. who use a classifier to identify matching sentences.

BLEU alignment. We propose a simple metric, the BLEU score [23] to align the AUTOMATED dataset. The BLEU score is used commonly to evaluate Machine Translation algorithms, by comparing the similarity between the output of a translation algorithm to references sentence. In short, BLEU does this by counting overlapping word n-grams. For the sake of brevity, we refer to Papineni et al.[23] for details on the method. To the best of our knowledge, we're the first to employ BLEU for a sentence alignment task. To evaluate the quality of the BLEU alignment for the sentence alignment task, we compare BLEU alignment to the Maximum alignment reported by Kajiwara and Komachi[14], using the manual alignment set from Hwang et al. [12] for evaluation. This evaluation set contains 67,853 candidate sentence pairs, judged by human annotators. 277 were considered fully aligned, 281 partially aligned and 67,295 considered either not good enough partial alignments or bad alignments.

We test both methods in two sentence alignment scenarios: (1) Full alignment: Fully aligned sentences versus the rest and (2) Partial alignment: Fully and partially aligned sentences versus the rest.

Table 2 reports maximum F1-score and AUC for both methods in both scenarios. We observe that BLEU alignment performs on

[2] https://pypi.org/project/wikipedia/
[3] http://www.nltk.org

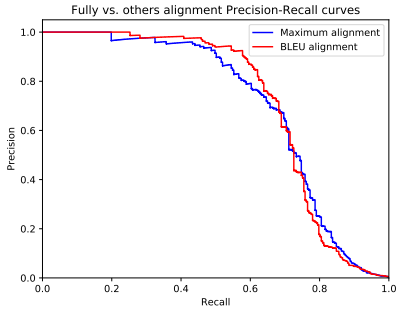


Figure 1: Precision-recall curves of Maximum alignment and BLEU alignment

Fully vs. rest	Max F1	AUC
BLEU alignment	0.717	0.714
Maximum alignment in [14]	0.717	0.730
Alignment used to align Wikipedia [12]	0.712	0.694
Fully and partially vs. rest		
BLEU alignment	0.534	0.484
Maximum alignment in [14]	0.638	0.618
Alignment used to align Wikipedia [12]	0.607	0.529

Table 2: Max F1 and AUC scores for identifying fully aligned and fully and partially aligned sentences

par with Maximum alignment for fully aligned sentences, but performance is worse on partially aligned sentences. In figure 1 we report the precision-recall curve for the fully aligned scenario.

We observe that BLUE alignment provides a useful method when performing sentence alignment on highly domain specific text. Despite the poorer performance on partial alignment, it does not depend on pretrained embeddings or external datasources to function. In addition, when aligning medical data, the vocabulary might contain a lot of words that are neither in the vocabulary of pretrained Word2Vec models nor in Wiktionary, which may deteriorate performance of approaches that use such sources. BLEU alignment only looks at overlapping n-grams, which makes it language independent.

In the AUTOMATED dataset, we only include fully aligned sentence pairs, given the poor performance BLEU alignment demonstrated on partial alignment. In presented dataset, we include sentence pairs with a BLEU score >0.29, which led to a maximum F1 score during evaluation. After filtering out sentences with MediaWiki mathematical formulas, the final set consists of 3,797 fully aligned medical sentences. In table 3 we present example aligned sentences from this set.

4 NEURAL TEXT SIMPLIFICATION

Most current research on text simplification in the medical domain focuses on simplifying medical concepts only. However, monolingual NMT has shown great potential in text simplification research but has not been applied to the medical domain yet. Therefore, we replicate the state-of-the-art NMT text simplification system of [22] as a baseline, and evaluate it on our expert-curated

Wikipedia	Simple Wikipedia	BLEU
Spinal tumors are neoplasms located in the spinal cord.	Spinal tumors is a form of tumor that grows in the spinal cord.	0.39
Aspirin is an appropriate immediate treatment for a suspected MI.	Aspirin is an early and important treatment for a heart attack.	0.33

Table 3: Example alignments using BLEU alignment

dataset. This system outperformed phrase-based [32] and syntax-based statistical machine translation [35] approaches, as well as an unsupervised lexical simplification approach [10]. Furthermore, we design a second NMT model which uses a combination of our AUTOMATED and other datasets, and replacing medical concepts with identifiers.

4.1 Training and Evaluation sets

For the setup of our experiments, we rely on the general dataset presented by Hwang et al.[12] and combine this with the EXPERT and AUTOMATED datasets described in the previous section. This gives us 4 datasets:

- **Fully aligned health sentences** f_{health} - Filtered and expert evaluated fully aligned health sentences from f_{wiki}: 2,267 sentences.
- **Partially aligned health sentences** p_{health} - Filtered and expert evaluated partially aligned health sentences from p_{wiki}: 3,148 sentences.
- **Fully aligned general domain sentences** $f_{\text{general}} = f_{\text{wiki}} - f_{\text{health}}$: 152.538 sentences.
- **Partially aligned general domain sentences** $p_{\text{general}} = p_{\text{wiki}} - p_{\text{health}}$: 126.785 sentences.

4.2 Baseline

We implemented the baseline system in OpenNMT[4], an open source framework for NMT. The architecture consists of two LSTM layers, states of size 500 and 500 hidden units and a 0.3 dropout probability. The vocabulary size is pruned to 50,000 in both the source and target language. Word embedding size is set to 500. We used pretrained Word2Vec embeddings from the Google News corpus [20] of size 300. The remaining part is learned during training of the NMT (while the pre-trained part remains fixed). Lastly, the decoder uses global attention with input feeding [19]. The system is trained for 15 epochs, using a SGD optimizer and an initial learning rate of 1.0. After epoch 9, the learning rate decay is 0.7, i.e. learning_rate = learning_rate * learning_rate_decay.

At translation time beam search is used to find the best prediction given the input. Beam search is an approximation of the best possible translation. At each step of the translation the k most likely words are generated given the input sentence. Here, k is called the beam size. Then, the most likely sequence (i.e. translation) is called hypothesis 1, the next hypothesis 2, etc. We will evaluate both hypothesis setting in the next section. The system that performed most changes and highest percentage of correct ones in [22] used a

[4]http://opennmt.net/

beam size of 12. This system is trained on general domain corpus, i.e. $f_{\text{general}} + p_{\text{general}}$.

4.3 Medical+CHV Replacement

Our second NMT system (MED-CHV) we evaluate in this paper follows a similar architecture as the baseline, but is trained on a combination of the general corpus and our health-related corpora (minus the corpus which is used as a test set): $f_{\text{general}} + p_{\text{general}} + f_{\text{health}} + p_{\text{health}} + f_{\text{BLEU-health}}$

In addition, we replace each medical concept encountered in the complex text with a Concept Unique Identifier (CUI) from UMLS. This approach reduces the (medical) vocabulary (and medical concept sparsity), since any textual variation of a concept is mapped (or normalized) to a single CUI, aggregating the references for each concept. For example, *atherosclerotic heart disease* and *coronary heart disease*, which are synonyms, are both replaced with C0010054. Furthermore, it enables us the replace CUIs with their CHV-preferred term if the CUI is not part of the source vocabulary of the NMT (i.e. rare medical concepts / CUIs). We used QuickUMLS [26] with a similarity threshold of 0.7, a value for which highest F1-scores were achieved in [26], to detect medical concepts and link them to a CUI. For the decoder we use pre-trained Word2Vec embeddings of size 200, trained on 10,876,004 English abstracts of biomedical articles from PubMed [15].

Note that we include the 50,000 most frequent words in the source and target vocabulary (so we have enough reference translations for each word in the vocabulary). This may cause that some CUIs are not in the source vocabulary and are therefore not translated. To overcome this, we replace CUIs that are out of vocabulary with their CHV-preferred term, if it exists, or copy the original source token. To do this, we make use of a phrase table, which can be pre-constructed before translation. Each entry in the phrase-table contains a CUI with its CHV-preferred term or its original source token. Instead of substituting out of vocabulary words with source words that have the highest attention weight, a possible translation in the phrase-table is looked up. This way the output does not contain any raw CUI.

5 EVALUATION

In this evaluation, we focus on the question of how well does NMT-based text simplification work in the medical domain. For our first experiment, we rely on an automated evaluation approach based on a reference test-set drawn from our EXPERT dataset. The second evaluation relies on human evaluators, and focuses on simplicity, understandability, and correctness of simplified sentences.

We randomly select 500 and 350 sentences as validation set and test set respectively from f_{health}. Automatic evaluation is done on the test set of size 350. Human evaluation is done in the first 70 sentences of the test set (since human evaluation is rather costly).

Automatic Evaluation: Text simplification is typically automatically evaluated using a traditional machine translation metric BLEU [23] and a text simplification specific metric SARI [35].

BLEU compares the output against references and produces a score between 0 and 1, with 1 representing a perfect translation (i.e. identical to one of the references). In our evaluation we use word n-grams up to 4. However, when used for simplification, it has to be handled with care as it is not uncommon that the source sentences (from Wikipedia) and the reference sentences (from Simple Wikipedia) are identical or very similar as Wikipedia editors just copied them over without or only with minor modifications. Therefore, a machine simplification which just keeps the source sentence as-is often has high BLEU scores, but is not simpler.

Hence, a specific text simplification metric was introduced in [35], called SARI, which compares **S**ystem output **A**gainst **R**eferences and against the **I**nput sentence. It focuses on lexical simplification, i.e. replacing complex words and phrases with simpler alternatives. "It explicitly measures the goodness of words that are *added*, *deleted* and *kept* by the systems" [35], by comparing the output with the source and the reference or multiple. SARI combines several aspects of adding and deleting words into a single numeric measure: the terms added by the simplification algorithm with respect to if they are also added in the reference simplification; and the terms removed by the simplification algorithm also with respect to if they are removed in the reference, and the terms which are kept stable between the reference and a simplification.

For this experiment, we evaluate the baseline system and the MED-CHV system, both with hypothesis 1 and 2 selection strategies (i.e., choose the most likely simplification and the second most likely one.) Furthermore, we consider an "Identity" simplification which just copies the source sentences without modifying.

Human Evaluation: As both metrics used in the automatic evaluation are insufficient to fully describe the capabilities of machine simplification, such evaluation need to be accompanied by a human evaluation. To this end, we obtain feedback on simplified sentences focusing on grammar, meaning preservation (both measured on a 1-5 Likert scale), and simplicity (on a scale of -2 to 2, with negative values representing that the text has become more complex). This follows the setup outlined in [22]. An evaluator is presented with a sentence pair (complex, simple) and asked to give the scores. We base our annotation guidelines on [27]. We slightly edited the guidelines, since their focus was on splitting (and deleting parts of) sentences, while our system mainly replaces words and deletes parts of sentences.

6 RESULTS AND DISCUSSION

In this section we report results of automatic and human evaluation.

Automatic Evaluation: In table 5, SARI, along with its three components, and BLEU scores are reported. The scores represent if the system is actually modifying the text, and how it relates to the test set reference sentences. "Identity" does not perform any text simplification, but simply uses the source sentence. This tells us how similar the source is to the reference. It serves as calibration scores for SARI and BLEU; e.g., not simplifying anything results in a BLEU score of 0.53 and a SARI score of 21.56. Both hypothesis 1 and 2 of the baseline (i.e. choosing the most likely or second likely simplification) are able to improve SARI scores. The main difference between them is that hypothesis 2 deletes with higher precision than hypothesis 1. Both hypotheses of the MED-CHV show comparable numbers for keeping and deleting terms, but a slightly higher number for adding terms. This may be because of the additional terms (medical concepts) the medical NMT is translating. BLEU scores of the identity and the baseline's

Source	Sentence
Wikipedia	Coronary artery disease (CAD) also known as atherosclerotic heart disease , coronary heart disease , or ischemic heart disease (IHD) , is the most common type of heart disease and cause of heart attacks .
Simple Wikipedia	Atherosclerosis is a form of heart disease .
Baseline, h-1	Coronary artery is the most common type of heart disease .
Baseline, h-2	Coronary artery is a type of disease .
Medical input	C1956346 (CAD) also known as C0010054 , C0010054 , or C0151744 (C0151744) , is the most common type of C0018799 and cause of C0027051 .
MED-CHV, h-1	$\{\text{coronary artery disease}\}_{\text{copied}}$ is the most common type of $\{\text{heart disease}\}_{\text{NMT}}$.
MED-CHV, h-2	$\{\text{coronary artery disease}\}_{\text{copied}}$ is the most common type of $\{\text{heart disease}\}_{\text{NMT}}$ and cause of $\{\text{heart attack}\}_{\text{NMT}}$.

Table 4: Example translations from different systems, medical concepts in MED-CHV are replaced with their CUI.

Approach	SARI	F_{add}	F_{keep}	P_{del}	BLEU
Identity	21.56	0.00	64.68	0.00	53.07
Baseline, h-1	28.14	1.91	60.37	22.15	54.78
Baseline, h-2	32.73	2.03	55.82	40.34	44.51
MED-CHV, h-1	32.27	2.24	57.10	37.47	47.48
MED-CHV, h-2	33.92	2.96	54.93	43.88	44.37

Table 5: Evaluations with automatic metrics

hypothesis 1 are highest. This may be due to that in hypothesis 1 the baseline is often producing the exact same sentence. The others are less conservative, i.e. perform more changes, which reduces BLEU. We showed that the NMT systems indeed improve SARI scores and therefore we expect that the output is simpler than the input. The medical NMT slightly increased SARI over the baseline (due to its F_{add} component). Therefore, we expect that simplicity scores will be at least similar to the baseline.

Manual Evaluation: Three laymen provided feedback on the first 70 sentences of the test set with respect to grammar, meaning preservation, and simplicity.

Table 6 shows that the baseline produces decent grammar and meaning preservation scores and indeed simplifies the text. However, MED-CHV scores show that grammar, meaning preservation and simplicity scores are all lower than the baseline. We assume that this is due to MED-CHV replacing out of vocabulary concepts with their CHV-preferred terms (which are expert curated simplified terms) instead of substituting them with source words that have the highest attention weight. While we assumed that using these expert term simplifications should perform well, also previous research concluded that "some CHV-preferred terms can be above the level of consumers' comprehension" [24].

Example translations are given in table 4. Note that the input of medical NMT is the Wikipedia sentence with medical concepts replaced with their CUI. Common medical concepts, such as heart disease (C0018799) and heart attack (C0027051), are part of the vocabulary and correctly translated by the NMT. Coronary artery disease (C1956346) is neither part of the vocabulary, nor a CHV-preferred term exists for it. Therefore, the source term is copied.

Approach	G	M	S
Simple Wikipedia	4.91	4.24	0.53
Baseline, h-1	4.85	4.30	0.22
Baseline, h-2	4.49	3.87	0.23
MED-CHV, h-1	4.23	3.82	-0.05
MED-CHV, h-2	4.19	3.76	-0.05

Table 6: Human evaluation scores. G:Grammar, M:Meaning preservation, S:Simplicity

7 CONCLUSION AND FUTURE WORK

Automated Medical Text Simplification can be a cornerstone technology to address insufficient health literacy. However, research into this domain is hampered by the lack of open training and test corpora. Therefore, in this paper we introduced such an open corpus which is based on the widely available Wikipedia-Simple Wikipedia text simplification corpus, and expanded with additional aligned sentences focusing on the medical domain. This corpus was created based on filtering with a medical expert from an existing aligned dataset, and by a novel simple, language independent monolingual text alignment method.

We used this corpus to evaluate two Neural Machine Translation models: one was trained on the aligned Wikipedia corpus (baseline), the other one was in addition trained on our corpus, but with medical terms replaced by their UMLS Concept Unique Identifiers. We assumed that the replacement would further boost performance. Both models were evaluated automatically and manually focusing only on the medical subset of the test data set we created. During automatic evaluation, it could be shown that the baseline performs fewer changes to sentences when simplifying. However, in the manual human-driven evaluation, it became clear that changing too many parts of the sentence can be detrimental, and that the baseline sentences were judged to be more understandable and simpler. We assume that this can be attributed to the act of replacing out of vocabulary medical concepts with their CHV-preferred terms. We therefore assume that training only with our extended dataset without additional replacements should yield superior performance. Due to the extreme costs of manually evaluating simplification results, this experiment will be covered in our future work. While this result was disappointing, it shows that automatic text simplification is a difficult task which demands future research.

In summary, we contributed a novel and open test and training dataset of aligned sentences focused on medical text simplifications, which easily allows such future research. Furthermore, we could show that even training a Neural Machine Translation model on a non-specialized corpus can still yield acceptable results in a complex domain like medical texts, clearly hinting at the potential of future endeavours.

REFERENCES

[1] Emil Abrahamsson, Timothy Forni, Maria Skeppstedt, and Maria Kvist. 2014. Medical text simplification using synonym replacement: Adapting assessment of word difficulty to a compounding language. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)*. 57–65.

[2] Viraj Adduru, Sadid Hasan, Joey Liu, Yuan Ling, Vivek Datla, and Kathy Lee. 2018. Towards dataset creation and establishing baselines for sentence-level neural clinical paraphrase generation and simplification. In *The 3rd International Workshop on Knowledge Discovery in Healthcare Data*.

[3] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.

[4] Olivier Bodenreider. 2004. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic acids research* 32, suppl_1 (2004), D267–D270.

[5] Jinying Chen, Emily Druhl, Balaji Polepalli Ramesh, Thomas K Houston, Cynthia A Brandt, Donna M Zulman, Varsha G Vimalananda, Samir Malkani, and Hong Yu. 2018. A Natural Language Processing System That Links Medical Terms in Electronic Health Record Notes to Lay Definitions: System Development Using Physician Reviews. *Journal of Medical Internet Research* 20, 1 (2018), e26.

[6] Jinying Chen, Abhyuday N Jagannatha, Samah J Fodeh, and Hong Yu. 2017. Ranking medical terms to support expansion of lay language resources for patient comprehension of electronic health record notes: adapted distant supervision approach. *JMIR medical informatics* 5, 4 (2017).

[7] Jinying Chen and Hong Yu. 2017. Unsupervised ensemble ranking of terms in electronic health record notes based on their importance to patients. *Journal of biomedical informatics* 68 (2017), 121–131.

[8] Jinying Chen, Jiaping Zheng, and Hong Yu. 2016. Finding important terms for patients in their electronic health records: a learning-to-rank approach using expert annotations. *JMIR medical informatics* 4, 4 (2016).

[9] Kevin Donnelly. 2006. SNOMED-CT: The advanced terminology and coding system for eHealth. *Studies in health technology and informatics* 121 (2006), 279.

[10] Goran Glavaš and Sanja Štajner. 2015. Simplifying lexical simplification: do we need simplified corpora?. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 63–68.

[11] Zhe He, Zhiwei Chen, Sanghee Oh, Jinghui Hou, and Jiang Bian. 2017. Enriching consumer health vocabulary through mining a social Q&A site: A similarity-based approach. *Journal of biomedical informatics* 69 (2017), 75–85.

[12] William Hwang, Hannaneh Hajishirzi, Mari Ostendorf, and Wei Wu. 2015. Aligning sentences from standard wikipedia to simple wikipedia. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 211–217.

[13] Ling Jiang and Christopher C Yang. 2015. Expanding consumer health vocabularies by learning consumer health expressions from online health social media. In *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*. Springer, 314–320.

[14] Tomoyuki Kajiwara and Mamoru Komachi. 2016. Building a monolingual parallel corpus for text simplification using sentence similarity based on alignment between word embeddings. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 1147–1158.

[15] Aris Kosmopoulos, Ion Androutsopoulos, and Georgios Paliouras. 2015. Biomedical semantic indexing using dense word vectors in bioasq. *J BioMed Semant Suppl BioMedl Inf Retr* 3410 (2015), 959136040–1510456246.

[16] Poorna Kushalnagar, Scott Smith, Melinda Hopper, Claire Ryan, Micah Rinkevich, and Raja Kushalnagar. 2018. Making cancer health text on the Internet easier to read for deaf people who use American Sign Language. *Journal of Cancer Education* 33, 1 (2018), 134–140.

[17] Gondy Leroy, David Kauchak, and Alan Hogue. 2016. Effects on text simplification: Evaluation of splitting up noun phrases. *Journal of health communication* 21, sup1 (2016), 18–26.

[18] Carolyn E Lipscomb. 2000. Medical subject headings (MeSH). *Bulletin of the Medical Library Association* 88, 3 (2000), 265.

[19] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[20] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[21] Partha Mukherjee, Gondy Leroy, David Kauchak, Srinidhi Rajanarayanan, Damian Y Romero Diaz, Nicole P Yuan, T Gail Pritchard, and Sonia Colina. 2017. NegAIT: A new parser for medical text simplification using morphological, sentential and double negation. *Journal of biomedical informatics* 69 (2017), 55–62.

[22] Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 85–91.

[23] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 311–318.

[24] Basel Qenam, Tae Youn Kim, Mark J Carroll, and Michael Hogarth. 2017. Text simplification using consumer health vocabulary to generate patient-centered radiology reporting: translation and evaluation. *Journal of medical Internet research* 19, 12 (2017).

[25] Isabel Segura-Bedmar and Paloma Martínez. 2017. Simplifying drug package leaflets written in Spanish by using word embedding. *Journal of biomedical semantics* 8, 1 (2017), 45.

[26] Luca Soldaini and Nazli Goharian. 2016. Quickumls: a fast, unsupervised approach for medical concept extraction. In *MedIR workshop, sigir*.

[27] Sanja Štajner and Goran Glavaš. 2017. Leveraging event-based semantics for automated text simplification. *Expert systems with applications* 82 (2017), 383–395.

[28] Elior Sulem, Omri Abend, and Ari Rappoport. 2018. Simple and Effective Text Simplification Using Semantic and Neural Methods. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 162–173.

[29] Sharon Swee-Lin Tan and Nadee Goonawardene. 2017. Internet health information seeking and the patient-physician relationship: a systematic review. *Journal of medical Internet research* 19, 1 (2017).

[30] VG Vinod Vydiswaran, Qiaozhu Mei, David A Hanauer, and Kai Zheng. 2014. Mining consumer health vocabulary from community-generated text. In *AMIA Annual Symposium Proceedings*, Vol. 2014. American Medical Informatics Association, 1150.

[31] World Health Organization (WHO and others. 2018. Health literacy. The solid facts. *Self* (2018).

[32] Sander Wubben, Antal Van Den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 1015–1024.

[33] Deborah X Xie, Ray Y Wang, and Sivakumar Chinnadurai. 2018. Readability of online patient education materials for velopharyngeal insufficiency. *International journal of pediatric otorhinolaryngology* 104 (2018), 113–119.

[34] Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association of Computational Linguistics* 3, 1 (2015), 283–297.

[35] Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics* 4 (2016), 401–415.

[36] Ming Yang and Melody Kiang. 2015. Extracting Consumer Health Expressions of Drug Safety from Web Forum. In *System Sciences (HICSS), 2015 48th Hawaii International Conference on*. IEEE, 2896–2905.

[37] Qing T Zeng and Tony Tse. 2006. Exploring and developing consumer health vocabularies. *Journal of the American Medical Informatics Association* 13, 1 (2006), 24–29.

[38] Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd international conference on computational linguistics*. Association for Computational Linguistics, 1353–1361.

Semantic Hilbert Space for Text Representation Learning

Benyou Wang, Qiuchi Li, Massimo Melucci
University of Padua
Padua, Italy
wang,qiuchili,melo@dei.unipd.it

Dawei Song
Beijing Institute of Technology
Beijing, China
dawei.song2010@gmail.com

ABSTRACT

Capturing the meaning of sentences has long been a challenging task. Current models tend to apply linear combinations of word features to conduct semantic composition for bigger-granularity units e.g. phrases, sentences, and documents. However, the semantic linearity does not always hold in human language. For instance, the meaning of the phrase "ivory tower" cannot be deduced by linearly combining the meanings of "ivory" and "tower". To address this issue, we propose a new framework that models different levels of semantic units (e.g. sememe, word, sentence, and semantic abstraction) on a single *Semantic Hilbert Space*, which naturally admits a non-linear semantic composition by means of a complex-valued vector word representation. An end-to-end neural network [1] is proposed to implement the framework in the text classification task, and evaluation results on six benchmarking text classification datasets demonstrate the effectiveness, robustness and self-explanation power of the proposed model. Furthermore, intuitive case studies are conducted to help end users to understand how the framework works.

CCS CONCEPTS

• **Information systems** → *Document structure*; *Content analysis and feature selection.*

KEYWORDS

text understanding, neural network, quantum theory

ACM Reference Format:
Benyou Wang, Qiuchi Li, Massimo Melucci and Dawei Song. 2019. Semantic Hilbert Space for Text Representation Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313516

[1] https://github.com/wabyking/qnn

† Benyou Wang and Qiuchi Li contribute equally and share the co-first authorship.
*Massimo Melucci (melo@dei.unipd.it) is the corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313516

1 INTRODUCTION

In natural language understanding, it is crucial, yet challenging, to model sentences and capture their meanings. Essentially, most statistical machine learning models [3, 9, 15, 20, 23] are built within a linear bottom-up framework, where words are the basic features adopting a low-dimensional vector representation, and a sentence is modeled as a linear combination of individual word vectors. Such linear semantic composition is efficient, but does not always hold in human language. For example, the phrase "ivory tower", which means "a state of privileged seclusion or separation from the facts and practicalities of the real world", is not a linear combination of the individual meanings of "ivory" and "tower". Instead, it carries a new meaning. We are therefore motivated to investigate a new language modeling paradigm to account for such intricate non-linear combination of word meanings.

Drawing inspiration from the recent findings in the emerging research area of quantum cognition, which suggest that human cognition [1, 2, 8] especially language understanding [6, 7, 29] exhibit certain non-classical phenomena (i.e. quantum-like phenomena), we propose a theoretical framework, named *Semantic Hilbert Space*, to formulate quantum-like phenomena in language understanding and to model different levels of semantic units in a unified space.

In Semantic Hilbert Space, we assume that words can be modeled as microscopic particles in superposition states, over the basic sememes (i.e. minimum semantic units in linguistics), while a combination of word meanings can be viewed as a mixed system of particles. The Semantic Hilbert Space represents different levels of semantic units, ranging from basic sememes, words and sentences, on a unified complex-valued vector space. This is fundamentally different from existing quantum-inspired neural networks for question answering [31, 32] which are based on a real vector space. In addition, we introduce a new semantic abstraction, named as *Semantic Measurements*, which are also embedded in the same vector space and trainable to extract high-level features from the mixed system.

As shown in Fig. 1, the Semantic Hilbert Space is built on the basis of quantum probability (QP), which is the probability theory for explaining the uncertainty of quantum superposition. As quantum superposition requires the use of the complex field, Semantic Hilbert Space has complex

values and operators. In particular, the probability function is implemented by a unique (complex) density operator.

Semantic Hilbert Space adopts a complex-valued vector representation of unit length, where each component adopts an amplitude-phase form $z = re^{i\phi}$. We hereby hypothesize that the amplitude r and complex phase ϕ can be used to encode different levels of semantics such as lexical-level co-occurrence, hidden sentiment polarity or topic-level semantics. When word vectors are combined, even in a simple complex-valued addition form, the resulting expression will entail a non-linear composition of amplitudes and phases, thus indicating a complicated fusion of different levels of semantics. A more detailed explanation is given in Sec. 3. In this way, the complex-valued word embedding is fundamentally different from existing real-valued word embedding. A series of ablation tests indicate that the complex-valued word embedding can increase performance.

The Semantic Hilbert Space is an abstract representation of our approach to modeling language through QP. At the level of implementation, an efficient and effective computational framework is needed to cope with large text collections. To do so, we propose an end-to-end neural network architecture, which provides means for training of the network components. Each component corresponds to a physical meaning of quantum probability with well-defined mathematical constraints. Moreover, each component is easier to understand than the kernels in convolutional neural network and cells in recurrent neural networks.

The network proposed in this paper is evaluated on six benchmarking datasets for text classification and achieves a steady increase over existing models. Moreover, it is shown that the proposed network is advantageous due to its high robustness and self-explanation capability.

2 SEMANTIC HILBERT SPACE

The mathematical foundation of Quantum Theory is established on a Hilbert Space over the complex field. In order to borrow the underlying mathematical formalism of quantum theory for language understanding, it is necessary to build such a Hilbert Space for language representation. In this study, we build a *Semantic Hilbert Space* $\mathcal{H}$ over the complex field. As is illustrated in Fig. 1, multiple levels of semantic units are modeled on this common Semantic Hilbert Space. In the rest of this section, the semantic units under modeling are introduced separately.

We follow the standard *Dirac Notation* for Quantum Theory. A unit vector and its transpose are denoted as a ket $|\mu\rangle$ and a bra $\langle\mu|$, respectively. The inner product and outer product of two unit vectors $\vec{u}$ and $\vec{v}$ are denoted as $\langle u|v\rangle$ and $|u\rangle\langle v|$ respectively.

2.1 Sememes

Sememes are the minimal non-separable semantic units of word meanings in language universals [13]. For example, the word "blacksmith" is composed of sememes "human", "occupation", "metal" and "industrial". We assume that the Semantic Hilbert Space $\mathcal{H}$ is spanned by a set of orthogonal basis $\{|e_j\rangle\}_{j=1}^n$ corresponding to a finite closed set of sememes $\{e_j\}_{j=1}^n$. In the quantum language, the set of sememes are modeled as *basis states*, which is the basis for representing any quantum state. In Fig. 1, the axes of the Semantic Hilbert Space correspond to the set of sememe states, and semantic units with larger granularity are represented based on quantum probability.

2.2 Words

The meaning of a word is a combination of sememes. We adopt the concept of *superposition* to formulate this combination. Essentially, a word w is modeled as a quantum particle in *superposition state*, represented by a unit-length vector in the Semantic Hilbert Space $\mathcal{H}$, as can be seen in Fig. 1. It can be written as a linear combination of the basis states for sememes:

$$|w\rangle = \sum_{j=1}^{n} r_j e^{i\phi_j} |e_j\rangle \tag{1}$$

where the complex-valued weight $r_j e^{i\phi_j}$ denotes how much the meaning of word w is associated with the sememe e_j. Here $\{r_j\}_{j=1}^n$ are non-negative real-valued amplitudes satisfying $\sum_{j=1}^n {r_j}^2 = 1$ and $\phi_j \in [-\pi, \pi]$ are the corresponding complex phases. We could also transfer the complex number in a complex plane as $re^{i\phi} = r\cos\phi + ir\sin\phi$.

It is worth noting that the complex phases $\{\phi_j\}$ are crucial as they implicitly entail the *quantum interference* between words. Suppose two words w_1 and w_2 are associated to weights $r_j^{(1)} e^{i\phi_j^{(1)}}$ and $r_j^{(2)} e^{i\phi_j^{(2)}}$ for the sememe e_j. The two words in combination are therefore at the state e_j with a probability of

$$\left| r_j^{(1)} e^{i\phi_j^{(1)}} + r_j^{(2)} e^{i\phi_j^{(2)}} \right|^2 = \left| r_j^{(1)} \right|^2 + \left| r_j^{(2)} \right|^2 + 2 r_j^{(1)} r_j^{(2)} \cos\left(\phi_j^{(1)} - \phi_j^{(2)}\right) \tag{2}$$

where the term $2r_j^{(1)} r_j^{(2)} \cos(\phi_j^{(1)} - \phi_j^{(2)})$ reflects the interference between the two words, where as the classical case corresponds to a particular case $\phi_j^{(1)} = \phi_j^{(2)} = 0$.

2.3 Semantic Compositions

As is illustrated in Fig. 1, we view a word composition (e.g. a sentence) as a bag of words [14], each of which is modeled as a particle in superposition state on the Semantic Hilbert Space $\mathcal{H}$. To obtain the semantic composition of words, we leverage the concept of *quantum mixture* and formulate the word composition as a mixed system composed of the word

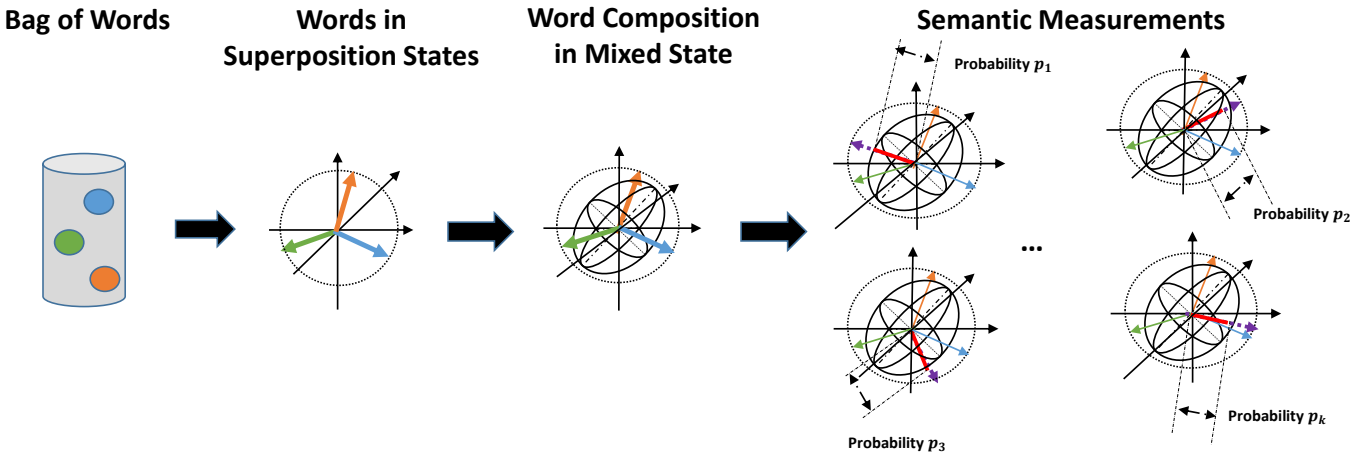


Figure 1: Illustration of Semantic Hilbert Space. The green, blue and orange colors correspond to three different words modeled as quantum particles. The black dotted circle represents the unit ball in the Semantic Hilbert Space. The ellipsoid in solid line refers to the quantum probability distribution defined by the density matrix of the word composition. The purple lines are semantic measurements. The intersections of the ellipsoids and semantic measurements are in thick red lines, the lengths of which correspond to measurement probabilities.

superposition states. The system is in a *mixed state* represented by a n-by-n density matrix ρ on $\mathcal{H}$, which is positive semi-definite with trace 1. It is computed as follows:

$$\rho = \sum_i p(i) \, |w_i\rangle \langle w_i|, \tag{3}$$

where $|w_i\rangle$ denotes the superposition state of the i-th word and $p(i)$ is the classical probability of the state $|w_i\rangle$ with $\sum_i p(i) = 1$. It determines the contribution of the word w_i to the overall semantics.

The complex-valued density matrix ρ can be seen non-classical distribution of sememes in $\mathcal{H}$. Its diagonal elements are real and form a classical distribution of sememes, while its complex-valued off–diagonal entries encode the interplay between sememes, which in turn gives rise to the interference between words. A density matrix assigns a probability value for any state on $\mathcal{H}$ such that the values for any set of orthogonal states sum up to 1 [12]. Hence it is visualized as an ellipsoid in Fig. 1, assigning a quantum probability to a unit vector with the intersection length.

2.4 Semantic Measurements

As a non-classical probability distribution, a sentence density matrix carries rich information and in particular it contains all the information about a quantum system. In order to extract the relevant information to a concrete task from the semantic composition, we build a set of measurements and compute the probability that the mixed system falls onto each of the measurements as a high-level abstraction of the semantic composition.

Suppose our proposed *semantic measurements* are associated with a set of measurement projectors $\{P_i\}_{i=1}^k$. According to the Born's rule [5], applying the measurement projector P_i onto the sentence density matrix ρ yields the following result:

$$p_i = tr(P_i \rho) \tag{4}$$

Here, we only consider pure states as measurement states, i.e. $P_i = |v_i\rangle \langle v_i|$. Moreover, we ignore the constraints of the measurements states $\{|v_i\rangle\}_{i=1}^k$ (i.e. orthogonality or completeness), but keep them trainable, so that the most suitable measurements can be determined automatically by the data in a concrete task, such as classification or regression. In this way, the trainable semantic measurements can be understood as a similar approach to supervised dimensionality reduction [11], but in a quantum probability framework with complex values.

3 QUANTUM PROBABILITY DRIVEN NETWORK

In order to implement the proposed framework, we further propose an end-to-end neural network based on quantum probability. Fig. 2 shows the architecture of the proposed Quantum Probability Driven Network (QPDN). The embedding layer, which is composed of a unit complex-valued embedding and a term-weight lookup table, captures the basic lexical features. The mixture layer is designed to combine the low-level bag-of-word features with an additive complex-valued outer product operation. The measurement layer adopts a set of trainable semantic measurements to extract the higher-level features for the final linear classifier.

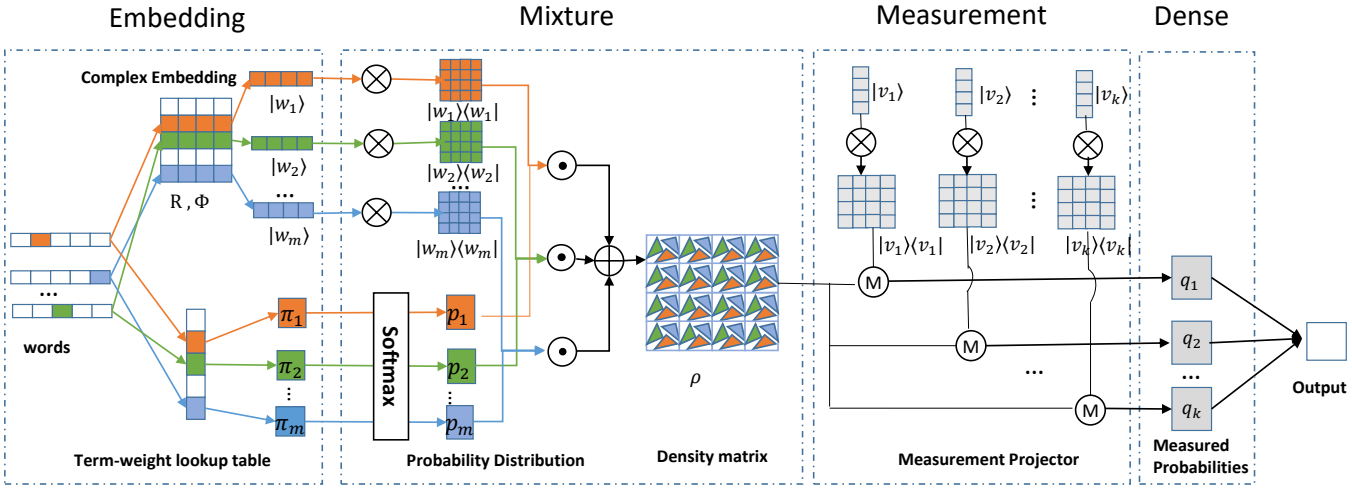


Figure 2: Architecture of Quantum probability-driven Neural Network. $\odot$ means that a matrix multiplies a number with each elements. $\oplus$ refers to a element-wise addition. $\otimes$ denotes a outer production to a vector, $\mathfrak{M}$ means a measurement operation according to Eq. 4.

In the following we will introduce the architecture layer by layer.

3.1 Embedding Layer

The parameters of the embedding layer are $\{R, \Phi, \Pi\}$, respectively, denoting the amplitude embedding, the phase embedding, and the term-weight lookup table. Eq. 1 expresses a quantum representation as a unit-length, complex-valued vector representation for a word w, i.e. $|w\rangle = [r_1 e^{i\phi_1}, r_2 e^{i\phi_2} ... r_n e^{i\phi_n}]^T$. The term-weight lookup table is used to weight words for semantic combinations, which will be described in the next subsection. During training, word embeddings need to be normalized to unit length after each batch.

This representation allows for a non-linear composition of amplitudes and phases in its mathematical form. Suppose two words w_1 and w_2 are of weights $r_j^{(1)} e^{i\phi_j^{(1)}}$ and $r_j^{(2)} e^{i\phi_j^{(2)}}$ for the j^{th} dimension (corresponding to the j^{th} sememe). The combination of w_1 and w_2 gives rise to a weight $r_j e^{i\phi_j}$ for the j^{th} dimension computed as

$$\begin{aligned} r_j e^{i\phi_j} &= r_j^{(1)} e^{i\phi_j^{(1)}} + r_j^{(2)} e^{i\phi_j^{(2)}} \\ &= \sqrt{|r_j^{(1)}|^2 + |r_j^{(2)}|^2 + 2 r_j^{(1)} r_j^{(2)} cos(\phi_j^{(1)} - \phi_j^{(2)})} \\ &\times e^{i \arctan\left(\frac{r_j^{(1)} \sin(\phi_j^{(1)}) + r_j^{(2)} \sin(\phi_j^{(2)})}{r_j^{(1)} \cos(\phi_j^{(1)}) + r_j^{(2)} \cos(\phi_j^{(2)})}\right)} \end{aligned} \quad (5)$$

Where both r_j and ϕ_j is a non-linear combination of $r_j^{(1)}$,$r_j^{(2)}$,$\phi_j^{(1)}$ and $\phi_j^{(2)}$. If the amplitudes and phases are associated to different levels of information, the amplitude-phase representation then naturally gives rise to a non-linear fusion of information.

3.2 Mixture Layer

A sentence is modeled as a density matrix, which is constructed in Sec. 2.3. Instead of using uniform weights in Eq. 3, word-sensitive weights are used for each word, which is commonly used in IR, e.g. inverse document frequency (IDF) as a word-dependent weight in TF-IDF scheme [28].

In order to guarantee the unit trace length for density matrix, the word weights which are from the lookup table in a sentence are normalized to a probability value through a softmax operation: $p(i) = e^{\pi(w_i)} / \sum_j^m e^{\pi(w_j)}$. Compared to the IDF weight, the normalized weight for a specific word in our approach is not static but updated adaptively in the training phase. Even in the inference/test phase, the real term weight i.e. $p(w_i)$ is also not static, but highly depends on the neighbor context words through nonlinear softmax function.

3.3 Measurement Layer

The measurement layer adopts a set of one-rank measurement projectors $\{|v_i\rangle\langle v_i|\}_{i=1}^k$ where $|v_i\rangle\langle v_i|$ is the outer product of its corresponding state in Semantic Hilbert Space $|v_i\rangle$. After each measurement, we can obtain one probability for each measurement state like $q_j = tr(\rho\, |v_j\rangle\langle v_j|)$. Finally, we can obtain a vector $\vec{q} = [q_1, q_2, ...q_k]$. Similarly to the word vectors, the states $|v_i\rangle$ are represented as unit states and normalized after several batches.

Dataset	train	test	vocab.	task	Classes
CR	4K	CV	6K	product reviews	2
MPQA	11k	CV	6K	opinion polarity	2
SUBJ	10k	CV	21k	subjectivity	2
MR	11.9k	CV	20k	movie reviews	2
SST	67k	2.2k	18k	movie reviews	2
TREC	5.4k	0.5k	10k	Question	6

Table 1: Dataset Statistics. (CV means 10-fold cross validation for testing performance.)

3.4 Dense Layer

The vector $\vec{q}$ in the measurement layer consists of k positive scalar numbers and it is used to infer the label for a given sentence. A dense layer with softmax activation is adopted after the measurement layer to get a classification probability distribution, i.e. $\widehat{\vec{y}} = \text{softmax}(\vec{q} \cdot W)$. The loss is designed as a cross-entropy loss between $\widehat{\vec{y}}$ and the one-hot label $\vec{y}$.

4 EXPERIMENTS

Our model was evaluated on six datasets for text classification: CR customer review [17], MPQA opinion polarity [30], SUBJ sentence subjectivity [25], MR movie review [25], SST binary sentiment classification [27], and TREC question classification [21]. The statistics of them are shown in Tab. 1.

We compared the proposed QPDN with various models, including Uni-TFIDF, Word2vec, FastText [18] and Sent2Vec [24] as unsupervised representation learning baselines, CaptionRep [15] and DictRep [16] as supervised representation learning baselines, as well as CNN [19] and BiLSTM [10] for advanced deep neural networks. In Tab. 2, we reported the classification accuracy values of these models from the original papers.

We used Glove word vectors [26] with 50,100,200 and 300 dimensions respectively. The amplitude embedding values are initialized by L2-norm, while the phases in complex-valued embedding are randomly initialized in $-\pi$ to π. We searched for the best performance in a parameter pool, which contains a learning rate in {1E-3, 1E-4, 1E-5, 1E-6}, an L2-regularization ratio in {1E-5, 1E-6, 1E-7, 1E-8}, a batch size in $\{8, 16, 32, 64, 128\}$, and the number of measurements in $\{5, 10, 20, 50, 100, 200\}$.

The main parameters in our model are R and Φ. Since both of them are $n \times |V|$ in shape, the number of parameters is roughly two times that of FastText [22]. For the other parameters, Π is $|V| \times 1$, $\{|v_i\rangle\}_{i=1}^{k}$ is $k \times 2n$, while W is $k \times |L|$ with L being the label set. Apart from word embeddings, the model is robust with limited scale at $k \times 2n + n \times |V| + k \times |L|$ for the number of parameters.

The results in Tab. 2 demonstrate the effectiveness of our model and an improvement of classification accuracies over some strong baseline supervised and unsupervised representation models on most of the datasets except MPQA. In comparison with more advanced models including BiLSTM and CNN, our model generally performed better than BiLSTM with increased accuracy values on the multi-class classification dataset (TREC) and three binary text classification datasets (MR, SST & SUBJ). However, it under-performed CNN on all 6 datasets with a difference of over 2% on 3 of them (MPQA, SST & TREC), probably because that it uses fewer parameters and simpler structures.

Table 2: Experimental Results in percentage (%). The best performed value (except for CNN/LSTM) for each dataset is in bold. where † means a significant improvement over FasText.

Model	CR	MPQA	MR	SST	SUBJ	TREC
Uni-TFIDF	79.2	82.4	73.7	-	90.3	85.0
Word2vec	79.8	**88.3**	77.7	79.7	90.9	83.6
FastText [18]	78.9	87.4	76.5	78.8	91.6	81.8
Sent2Vec [24]	79.1	87.2	76.3	80.2	91.2	85.8
CaptionRep [15]	69.3	70.8	61.9	-	77.4	72.2
DictRep [16]	78.7	87.2	76.7	-	90.7	81.0
Ours: QPDN	**81.0**†	87.0	**80.1**†	**83.9**†	**92.7**†	**88.2**†
CNN [19]	81.5	89.4	81.1	88.1	93.6	92.4
BiLSTM [10]	81.3	88.7	77.5	80.7	89.6	85.2

We argue that QPDN achieved a good balance between effectiveness and efficiency, due to the fact that it outperforms BiLSTM.

5 DISCUSSIONS

This section discusses the power of self-explanation and conducts an ablation test to examine the usefulness of important components of the network, especially the complex-valued word embedding.

Self-explanation Components. As is shown in Tab. 3, all components in our model have a clear physical meaning corresponding to quantum probability, where classical Deep Neural Network (DNN) can not well explain the role each component plays in the network. Essentially, we constructed a bottom-up framework to represent each level of semantic units on a uniform Semantic Hilbert Space, from the minimum semantic unit, i.e. sememe, to the sentence representation. The framework was operationalized through superposition, mixture and semantic measurements.

Ablation Test. An ablation test was conducted to examine how each component influences the final performance of QPDN. In particular, a double-length real word embedding network was implemented to examine the use of complex-valued word embedding, while mean weights and IDF weights were compared with our proposed trainable weights. A set of non-trainable orthogonal projectors and a dense layer on top of the sentence density matrix were

Table 3: Physical meanings and constraints

Components	DNN	QPDN
Sememe	-	basis vector / **basis state** $\{w \mid w \in C^n, \|\|w\|\|_2 = 1, \}$ complete &orthogonal
Word	real vector $(-\infty, \infty)$	unit complex vector / **superposition state** $\{w \mid w \in C^n, \|\|w\|\|_2 = 1\}$
Low-level representation	real vector $(-\infty, \infty)$	density matrix / **mixed system** $\{\rho \mid \rho = \rho^*, tr(\rho) = 1\}$
Abstraction	CNN/RNN $(-\infty, \infty)$	unit complex vector / **measurement** $\{w \mid w \in C^n, \|\|w\|\|_2 = 1\}$
High-level representation	real vector $(-\infty, \infty)$	probabilities/ **measured probability** $(0, 1)$

Table 4: Ablation Test

Setting	SST	Δ
FastText [18]	0.7880	-0.0511
FastText [18] with double-dimension real word vectors	0.7883	-0.0508
fixed amplitude part but trainable phase part	0.8199	-0.0192
replace trainable weights with fixed mean weights	0.8303	-0.0088
replace trainable weights with fixed IDF weights	0.8259	-0.0132
non-trainable projectors with fixed orthogonal ones	0.8171	-0.0220
replace projectors with dense layer	0.8221	-0.0170
QPDN	**0.8391**	-

implemented to analyze the effect of trainable semantic measurements.

Due to limited space, we only reported the ablation test result for SST, which is the largest and hence the most representative dataset. We used 100-dimensional real-valued word vectors and 50-dimensional complex-valued vectors for the models in the ablation test. All models under ablation were comparable in terms of time cost. Tab. 4 showed that each component plays an important role in the QPDN model. In particular, replacing complex embedding with double-dimension real word embedding led to a 5% drop in performance, which indicates that the complex-valued word embedding was not merely doubling the number of parameters.

The comparison with IDF and mean weights showed that the data-driven scheme gave rise to high-quality word weights. The comparison with non-trainable projectors and directly applying a dense layer on the density matrix showed that trainable measurements bring benefits to the network.

Discriminative Semantic Directions. In order to better understand the well-trained measurement projectors, we obtained the top 10 nearest words in the complex-valued vector space for each trained measurement state (like $|v_i\rangle$), using KD tree [4]. Due to limited space, we selected five measurements from the trained model for the MR dataset, and selected words from the top 10 nearest words to each measurement. As can be seen in Tab. 5, the first measurement was roughly about time, the second one was related to verb words which mainly mean 'motivating others'. The third measurement grouped uncommon non-English words together. The last two measurements also grouped words sharing similar meanings. It is therefore interesting to see that relevant words can somehow be grouped together into certain topics during the training process, which may be discriminative for the given task.

Table 5: The learned measurement for dataset MR. They are selected according to nearest words for a measurement vector in Semantic Hibert Space

Measurement	Selected neighborhood words
1	change, months, upscale, recently, aftermath
2	compelled, promised, conspire, convince, trusting
3	goo, vez, errol, esperanza, ana
4	ice, heal, blessedly, sustains, make
5	continue, warned, preposterousness, adding, falseness

6 CONCLUSIONS

In order to better model the non-linearity of word semantic composition, we have developed a quantum-inspired framework that models different granularities of semantic units on the same Semantic Hilbert Space, and implement this framework into an end-to-end text classification network. The network showed a promising performance on six benchmarking text datasets, in terms of effectiveness, robustness and self-explanation ability. Moreover, the complex-valued word embedding, which inherently achieved non-linear combination of word meanings, brought benefits to the classification accuracy in a comprehensive ablation study.

This work is among the first step to apply the quantum probabilistic framework to text modeling. We believe it is a promising direction. In the future, we would like to further extend this work by considering deeper and more complicated structures such as attention or memory mechanism in language, in order to investigate related quantum-like phenomena on textual data to provide more intuitive insights. Additionally, Semantic Hilbert Space in a tensor space is also worthy to be explored like [32], which may provide more interesting insights for current communities.

ACKNOWLEDGEMENT

This work is supported by the Quantum Access and Retrieval Theory (QUARTZ) project, which has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 721321, and by the National Key Research and Development Program of China (grant No. 2018YFC0831700), Natural Science Foundation of China (grant No. U1636203), and Major Project of Zhejiang Lab (grant No. 2019DH0ZX01).

REFERENCES

[1] Diederik Aerts, Liane Gabora, and Sandro Sozzo. 2013. Concepts and Their Dynamics: A Quantum-Theoretic Modeling of Human Thought. *Topics in Cognitive Science* (Sept. 2013). https://doi.org/10.1111/tops.12042 arXiv: 1206.1069.

[2] Diederik Aerts and Sandro Sozzo. 2014. Quantum Entanglement in Concept Combinations. *International Journal of Theoretical Physics* 53, 10 (Oct. 2014), 3587–3603. https://doi.org/10.1007/s10773-013-1946-z

[3] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2016. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. (Nov. 2016). https://openreview.net/forum?id=SyK00v5xx

[4] Jon Louis Bentley. 1975. Multidimensional binary search trees used for associative searching. *Commun. ACM* 18, 9 (1975), 509–517.

[5] Max Born. 1926. Zur Quantenmechanik der Sto\s svorgänge. *Zeitschrift für Physik* 37, 12 (Dec. 1926), 863–867. https://doi.org/10.1007/BF01397477

[6] Peter Bruza, Kirsty Kitto, Douglas Nelson, and Cathy McEvoy. 2009. Is there something quantum-like about the human mental lexicon? *Journal of Mathematical Psychology* 53, 5 (2009), 362–377.

[7] Peter D Bruza, Kirsty Kitto, Douglas McEvoy, and Cathy McEvoy. 2008. Entangling words and meaning. (2008).

[8] Jerome R Busemeyer and Peter D Bruza. 2012. *Quantum models of cognition and decision*. Cambridge University Press.

[9] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *EMNLP*. Association for Computational Linguistics, 670–680. http://aclweb.org/anthology/D17-1070

[10] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. *arXiv:1705.02364 [cs]* (May 2017). arXiv: 1705.02364.

[11] Ronald A Fisher. 1936. The use of multiple measurements in taxonomic problems. *Annals of eugenics* 7, 2 (1936), 179–188.

[12] Andrew M Gleason. 1957. Measures on the closed subspaces of a Hilbert space. *Journal of mathematics and mechanics* (1957), 885–893.

[13] Cliff Goddard and Anna Wierzbicka. 1994. *Semantic and lexical universals: Theory and empirical findings*. Vol. 25. John Benjamins Publishing.

[14] Zellig S Harris. 1954. Distributional structure. *Word* 10, 2-3 (1954), 146–162.

[15] Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning Distributed Representations of Sentences from Unlabelled Data. In *NAACL*. Association for Computational Linguistics, San Diego, California, 1367–1377. http://www.aclweb.org/anthology/N16-1162

[16] Felix Hill, KyungHyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to Understand Phrases by Embedding the Dictionary. *TACL* 4 (2016), 17–30. https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/\article/view/711

[17] Minqing Hu and Bing Liu. 2014. Mining and Summarizing Customer Reviews. (2014), 10.

[18] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* (2016).

[19] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *EMNLP* (2014), 1746–1751. https://doi.org/10.3115/v1/D14-1181

[20] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought Vectors. In *NIPS (NIPS'15)*. MIT Press, Cambridge, MA, USA, 3294–3302.

[21] Xin Li and Dan Roth. 2002. Learning Question Classifiers. In *COLING*. Association for Computational Linguistics.

[22] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[23] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2017. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. *arXiv:1703.02507 [cs]* (March 2017). arXiv: 1703.02507.

[24] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised Learning of Sentence Embeddings Using Compositional n-Gram Features. *NAACL* 1 (2018), 528–540. https://doi.org/10.18653/v1/N18-1049

[25] Bo Pang and Lillian Lee. 2005. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. Association for Computational Linguistics, 115–124. https://doi.org/10.3115/1219840.1219855

[26] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation.. In *EMNLP*, Vol. 14. 1532–1543.

[27] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. *EMNLP* (2013), 1631–1642.

[28] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* 28, 1 (1972), 11–21.

[29] Benyou Wang, Peng Zhang, Jingfei Li, Dawei Song, Yuexian Hou, and Zhenguo Shang. 2016. Exploration of quantum interference in document relevance judgement discrepancy. *Entropy* 18, 4 (2016), 144.

[30] Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating Expressions of Opinions and Emotions in Language. *Language Resources and Evaluation* 39, 2-3 (May 2005), 165–210. https://doi.org/10.1007/s10579-005-7880-9

[31] Peng Zhang, Jiabin Niu, Zhan Su, Benyou Wang, Liqun Ma, and Dawei Song. 2018. End-to-End Quantum-like Language Models with Application to Question Answering. (2018).

[32] Peng Zhang, Zhan Su, Lipeng Zhang, Benyou Wang, and Dawei Song. 2018. A Quantum Many-body Wave Function Inspired Language Modeling Approach. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1303–1312.

Learning Task-Specific City Region Partition

Hongjian Wang*
Twitter Inc.
hongjianw@twitter.com

Porter Jenkins
Penn State
prj3@ist.psu.edu

Hua Wei
Penn State
hzw77@ist.psu.edu

Fei Wu
Penn State
fxw133@psu.edu

Zhenhui Li
Penn State
jessieli@ist.psu.edu

ABSTRACT

The proliferation of publicly accessible urban data provide new insights on various urban tasks. A frequently used approach is to treat each region as a data sample and build a model over all the regions to observe the correlations between urban features (e.g., demographics) and the target variable (e.g., crime count). To define regions, most existing studies use fixed grids or pre-defined administrative boundaries (e.g., census tracts or community areas). In reality, however, definitions of regions should be different depending on tasks (e.g., regional crime count prediction vs. real estate prices estimation). In this paper, we propose a new problem of task-specific city region partitioning, aiming to find the best partition in a city w.r.t. a given task. We prove this is an NP-hard search problem with no trivial solution. To learn the partition, we first study two variants of Markov Chain Monte Carlo (MCMC). We further propose a reinforcement learning scheme for effective sampling the search space. We conduct experiments on two real datasets in Chicago (i.e., crime count and real estate price) to demonstrate the effectiveness of our proposed method.

CCS CONCEPTS

• **Information systems → Geographic information systems**; **Data mining**.

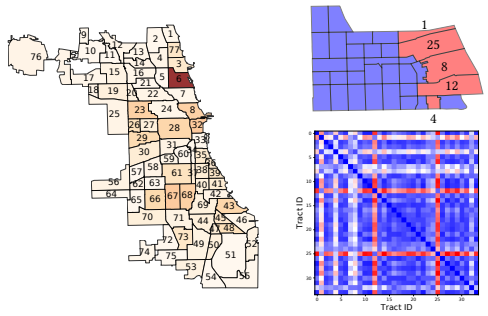
KEYWORDS

Region partition; reinforcement learning; crime prediction

ACM Reference Format:
Hongjian Wang, Porter Jenkins, Hua Wei, Fei Wu, and Zhenhui Li. 2019. Learning Task-Specific City Region Partition. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313704

1 INTRODUCTION

Given the growing amount of urban data, a number of data-driven models have been developed to provide insights for urban problems. A common approach is to treat each region as a data sample, take the region properties as features, and build a model to learn the correlation between region features and a target variable. Crime prediction is a good example. Criminologists are interested in knowing the correlation between demographics and crime [17, 21]. Each region i is taken as a data sample, with X_i as its demographic feature and Y_i as its crime count. A model (e.g., linear regression or negative binomial model) is built to estimate crime count vector Y using the feature matrix X. If some features (e.g., disadvantage index) show a significant correlation with crime count, researchers could relate this empirical result with criminology theory [8] and policy makers could further propose corresponding policies to address crime issues.



(a) Prediction errors (b) Community #6

Figure 1: (a) Heatmap of crime prediction error at community level in Chicago. Darker color means higher prediction error. The community area #6 is an outlier with the largest error. (b) Community #6 consists of 34 tracts (top) and their pair-wise similarity (bottom). It is clear that 5 tracts (red) on the east side are different from others.

Existing studies often use pre-defined administrative boundaries (e.g., street block, census tract, or community area) to define a region [17, 24]. A plot of 77 administrative community areas of Chicago is shown in Figure 1. However, such administrative boundaries might be too rigid and do not reflect the true spatial regions with regards to the targeted urban issues. One can alternatively propose to study the problem at the point level or small grid cell level (e.g., 10 meters by 10 meters grids). However, this could lead to data sparsity issues and jeopardize the integrity of the statistical model. Consequently, any inference made from the model would be at risk of bias. These concerns have both theoretical and policy implications. We use the following example to further illustrate the issue of using pre-defined community areas to study crime.

Example 1.1. Following previous work [17], we replicate a negative binomial model to predict crime count using demographic features by treating each community area as a data sample. Figure 1 plots the crime prediction error for each community area of Chicago. Community area #6 (i.e., Lake View area) shows an abnormally high error. In order to explain this outlier, we further investigate the internal structure of this area. Community #6 consists of 34 census tracts as shown in the top of Figure 1(b). In the bottom of Figure 1(b), we visualize pair-wise Euclidean distance between tracts based on demographic features. There are five tracts

*The work was done when the author was a student at Penn State University.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313704

that are different from others, and they are all located on the east side of community #6. These five tracts, when mixed with other tracts in this community, lead to inferior performance of crime count prediction in that area.

The observation above motivates us to learn a better region partition for crime study. In this paper, we propose a new problem of *task-specific region partitioning*. Given spatial variables and a selected model (e.g., linear regression), we aim to partition the city into regions such that the model trained by taking regions as data samples achieves optimal results.

Task-specific city region partitioning is a challenging problem. The key challenge lies in that the region properties (both features and target variable) and the model coefficients change simultaneously when we change the region partition. We prove that this is an NP-hard problem. We adopt a stochastic sampling approach to tackle our NP-hard problem. We start from a pre-defined region partition (e.g., community areas), and generate a new partition sample by flipping the membership of a smaller area (e.g., a tract). First, two variants of MCMC methods are proposed to generate new samples. Finally, we employ reinforcement learning to automatically learn how to generate the next sample that is more likely to improve the prediction performance.

We evaluate our method on two real datasets, i.e. crime counts and real estate prices. The learned region partitions are shown to consistently outperform the administrative boundaries and spatial clustering method. For example, our methods, on average, outperform the administrative boundary by 67% in MAE for a crime count prediction task.

To summarize, the contributions of this paper are three fold. First, we propose a novel problem on task-specific region partitioning, which is motivated by real-world urban studies [17]. Second, we prove the problem is NP-hard and use MCMC and reinforcement learning sampling strategies to solve the problem. Finally, we validate our method through extensive experiments on two real datasets.

2 REGION PARTITION PROBLEM

The input of our problem is a set of minimum spatial units. Without loss of generality, we use tract as the unit of study in this paper, and the set of tracts is denoted as $\mathcal{T} = \{t_1, t_2, \cdots, t_n\}$. Within each tract t_i, the following information are available $\langle p_i, y_i, x_i \rangle$, where p_i is a sequence of GPS coordinates representing tract exterior boundary, y_i is the target variable of interest, such as crime count and average house price, and $x_i \in \mathcal{R}^d$ is a d-dimension contextual feature vector. Note that each tract is a polygon with one connected component.

A city is partitioned into different community areas. We use community area as the proper unit to study the correlation between y and x. Each community area consists of several adjacent tracts, denoted as $Z_j = \{t_1^j, t_2^j, \cdots\}$. We derive $\langle P_j, Y_j, X_j \rangle$ for each community area Z_j by aggregating $\{\langle p_i, y_i, x_i \rangle | t_i \in Z_j\}$, where P_j is the exterior boundary, Y_j is the prediction target, and X_j is the contextual features of Z_j.

Definition 2.1 (Partition). A partition over $\mathcal{T}$ consists of a set of m non-overlapping community areas, denoted as $\mathcal{Z} = \{Z_1, Z_2, \cdots, Z_m\}$, satisfying the following four conditions

(1) (subset) $\forall j, Z_j \subset \mathcal{T}$;
(2) (non-overlapping) $\forall p, q, Z_p \cap Z_q = \emptyset$;
(3) (completeness) $\bigcup_{j=1}^{m} Z_j = \mathcal{T}$;
(4) (spatial-continuity) $\forall j$, P_j defines a polygon with exact one connected component.

A task is to learn a model f that takes input X and predicts Y on a given partition $\mathcal{Z}$. In order to learn a good model f, task-specific region partition problem is defined as follows.

Definition 2.2 (Task-specific region partition). Given a set of tracts $\mathcal{T}$ and a task f, find a partition $\mathcal{Z}$ with m components, such that the task error is minimized. Formally, we have

$$\arg\min_{\mathcal{Z}, f} \sum_{j=1}^{m} \Big(||Y_j - f(X_j)||_2 + G(Z_j) \Big), \tag{1}$$

where $G(\mathcal{Z}) = \sum_{j=1}^{m} G(Z_j)$ is a constraint on partition. For simplicity, we use $\mathcal{F}(\mathcal{Z}, f) = \sum_{j=1}^{m} \mathcal{F}(Z_j, f)$ as the quality measure of a partition $\mathcal{Z}$, where $\mathcal{F}(Z_j, f) = ||Y_j - f(X_j)||_2 + G(Z_j)$. Hence, the objective becomes

$$\arg\min_{\mathcal{Z}, f} \mathcal{F}(\mathcal{Z}, f). \tag{2}$$

The constraint function G ensures the partition $\mathcal{Z}$ has desirable property, such as small variance in community populations or balanced size in terms of community area.

NP-Hardness. The problem in Definition 2.2 is a combinatorial optimization problem. The decision version of the problem is to find a partition $\mathcal{Z}$, such that $\mathcal{F}(\mathcal{Z}, f) \leq \epsilon$, where ϵ is a constant. Here we prove such decision problem is NP-complete, and therefore the optimization version is NP-hard.

In the NP-completeness proof, we first approximate the decision problem above with an easier problem. The fact, that X_i, Y_i, and f are dynamically changing according to $\mathcal{Z}$, complicates the original problem. Therefore, we replace the jointly learned optimal f with a fixed f_0. Since f minimizes Equation 2 while f_0 does not, we have $\mathcal{F}(\mathcal{Z}, f) < \mathcal{F}(\mathcal{Z}, f_0)$. Next, we use the largest task error on one community area $\sup\{\mathcal{F}(Z_j, f_0)\}$ to approximate the overall task error $\mathcal{F}(\mathcal{Z}, f_0)$, where sup calculates supremum of a set. Namely, $\mathcal{F}(\mathcal{Z}, f) < \mathcal{F}(\mathcal{Z}, f_0) < m \cdot \sup\{\mathcal{F}(Z_j, f_0)\}$, because there are m community areas. Therefore, if we prove $\exists \mathcal{Z}$, such that $m \cdot \sup\{\mathcal{F}(Z_j, f_0)\} \leq \epsilon$, then we have $\mathcal{F}(\mathcal{Z}, f) \leq \epsilon$.

Now we prove the approximated decision problem is NP-complete. First, such approximated decision problem is NP, because given a partition $\mathcal{Z}_0$, we are able to validate $m \cdot \sup\{\mathcal{F}(Z_j, f_0)\} \leq \epsilon$ in polynomial time. Next, we prove the NP-hardness of this problem by reducing the (k, v)-balanced partitioning problem to the approximated decision problem. The (k, v)-balanced partition problem [2] is a proved NP-complete problem, which partition graph into k disjoint components of size at most $v\frac{n}{k}$, while the capacity of edge cut is less than ϵ. We construct the adjacency graph of all tracts $\mathcal{T}$, with weight $\sup\{\mathcal{F}(Z_j, f_0)\}$ on each edge. A solution to (m, v)-balanced partition problem on such adjacency graph is a solution to the approximated decision problem. The balanced partition problem achieve $k \cdot \sup\{\mathcal{F}(Z_j, f_0)\} \leq \epsilon$, where k is the number of edges to cut the graph. It is clear that $k > m$, and therefore we find a partition satisfying $m \cdot \sup\{\mathcal{F}(Z_j, f_0)\} \leq \epsilon$.

3 METHODS

We employ a stochastic search process to find approximate solutions due to the NP-hard nature of our problem. First, we propose two variants of Markov Chain Monte Carlo (MCMC) method with different sample proposal strategy. Then, we use reinforcement learning to automatically learn the sample strategy.

3.1 Markov Chain Monte Carlo

The optimal task function f in Equation (2) can be easily learned, when a partition $\mathcal{Z}$ is given. Therefore, the challenge lies in searching through the partition space. Toward this goal, we first adopt the MCMC method, or more specifically the Metropolis-Hastings algorithm to optimize $\mathcal{Z}$.

Markov Chain Monte Carlo [3] is a stochastic algorithm that constructs a Markov chain that will ultimately converge to p through stochastic sampling [11]. In our case, the state space is all possible partitions $\mathcal{Z}$, and the distribution $p(\mathcal{Z})$ defines how likely that $\mathcal{Z}$ is optimal. Clearly, it is difficult to calculate $p(\mathcal{Z})$, but a partition $\mathcal{Z}$ with lower $\mathcal{F}(\mathcal{Z})$ value is more likely to be optimal.

In addition to the quality function $\mathcal{F}$, MCMC employs a proposal function $q(\mathcal{Z}'|\mathcal{Z})$, which defines the transition probability from state $\mathcal{Z}$ to $\mathcal{Z}'$. The Markov chain moves toward $\mathcal{Z}'$ with acceptance probability $\gamma = \min\left[1, \frac{p(\mathcal{Z}')q(\mathcal{Z}|\mathcal{Z}')}{p(\mathcal{Z})q(\mathcal{Z}'|\mathcal{Z})}\right]$. And $p(\mathcal{Z}) = \frac{e^{-\mathcal{F}(\mathcal{Z})/T}}{P}$ is the Boltzmann distribution, where P is the normalization constant, and T is the temperature parameter. The proposal function q is very flexible, if not limitless. In practice, the choice of q generally affects how quickly the algorithm converges.

The MCMC algorithm is described below. First, we initialize the partition with the existing administrative boundary, denoted as $\mathcal{Z}_0$. Within each step, we draw $u \in [0, 1]$ from uniform distribution $\mathcal{U}_{[0,1]}$, draw the next partition $\mathcal{Z}'$, and calculate acceptance probability γ. Only if u is smaller than γ, we accept the new partition $\mathcal{Z}'$. We repeat the process above, until $\mathcal{F}(\mathcal{Z})$ converges.

MCMC with Naive Proposal. We begin by establishing a baseline MCMC with the simplest q we can devise. Namely, we generate a new partition by *uniformly randomly* selecting one tract $t_i \in \mathcal{T}$ that is on the boundary of some community area Z_j, and then flip t_i to the adjacent community area. With naive proposal strategy, q is symmetric, because $q(\mathcal{Z}'|\mathcal{Z})$. Therefore, the acceptance probability $\gamma = \min[1, \frac{p(\mathcal{Z}')}{p(\mathcal{Z})}]$.

Guided MCMC with Softmax Proposal. We propose another MCMC approach with a more intelligent proposal strategy than uniformly random selection. This approach is a greedy strategy that we prefer to adjust the community area with the highest prediction error to improve current partition.

Given this intuition, we heuristically design our guided MCMC method to sample a community area with large error first. To achieve this, we apply the softmax function over the prediction errors on different community areas to derive the sample probability of each community area, i.e. $p(Z_j) = \frac{\exp(||Y_j - f(X_j)||_2)}{\sum_{k=1}^{m} \exp(||Y_k - f(X_k)||_2)}$. Note that under the softmax proposal approach, the proposal function q is not symmetric. Therefore, we have to explicitly calculate the values for q functions.

Discussion: we acknowledge that there are plenty of other strategies to build random walks for MCMC. However, there is always a trade-off between efficiency and effectiveness. The naive MCMC is the simplest approach to conduct MCMC, while being slow to converge. The guided MCMC employs human heuristics to improve the efficiency at the expense of potentially converging to a worse local optima, because the greedy strategy may prune the search space too aggressively.

3.2 Reinforcement Learning

There are two main drawbacks of the MCMC method. First, when drawing a new sample, the MCMC methods do not account for any information from previous samples. Namely, the naive MCMC could repeatedly reject the same sample. Intuitively, all the samples we observed so far should carry certain information to assist sampling in the future. Second, the Markov chain-based stochastic search strategy is more likely to get stuck on a local optima, because the nature of MCMC sampling follows a depth first search. It is very likely that another chain leads to a better local optima, but the algorithm is not able to explore the alternative outcome.

To address these two issues of MCMC method, we propose a reinforcement learning (RL) [15] scheme for generating new samples. RL differs from standard supervised learning in that the correct input/output pairs are not presented at the same time. Instead, RL is concerned with how agents ought to take actions in an environment so as to maximize cumulative gain.

In what follows, we map the RL components to our problem. The set of tracts consists of the environment, and their community area assignment is the state. An action is to re-assign some tract t_i from Z_j to Z_p, denoted as tuple $\langle t_i, Z_p \rangle$. The immediate reward of such transition is defined by $\Delta\mathcal{F} = e^{-\mathcal{F}(\mathcal{Z}')} - e^{-\mathcal{F}(\mathcal{Z})}$, where the exponential function coverts the loss into gain. We define the cumulative gain Q as a function of the current state $\mathcal{Z}$ and action $\langle t^k, Z^k \rangle$ at step k, and Q satisfies the following condition

$$Q(\mathcal{Z}, \langle t^k, Z^k \rangle) = \Delta\mathcal{F} + \delta \cdot \sum_{a \in \{\langle t^{k+1}, Z^{k+1} \rangle\}} Q(\mathcal{Z}', a), \tag{3}$$

where δ is the discount factor on future reward and $\{\langle t^{k+1}, Z^{k+1} \rangle\}$ is the set of all possible actions given $\mathcal{Z}'$. Given Q function, at each state $\mathcal{Z}$, we are able to find the best action with the highest cumulative gain through

$$\arg\max_{\langle t, Z \rangle} Q(\mathcal{Z}, \langle t, Z \rangle). \tag{4}$$

In our problem, a reinforcement learning scheme faces the following three challenges.

Huge state space. The state space is exponential to the number of tracts. Consequently, we cannot track the exact reward for each state and actions. Instead, we rely on Deep Q-learning (DQN) [16], where a deep neural network approximates the Q function.

Our neural network architecture is shown in Figure 2. The input of our DQN model includes the partition $\mathcal{Z}$ and an action tuple $\langle t, Z \rangle$. Since the input consists of either indexes of community area or index of a tract, we employ an embedding layer to encode these positive integers into dense vectors. We concatenate $\mathcal{Z}$ and Z and apply the CA embedding layer $embed_{CA}$ to obtain d_{CA}-dimension

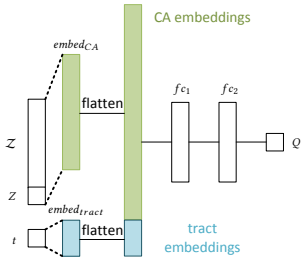


Figure 2: The Deep Q-learning neural network architecture. The input contains current partition $\mathcal{Z}$ and action $\langle t, Z\rangle$. We employ two embedding layers $embed_{CA}$ and $embed_{tract}$ for community area and tract respectively. The embeddings of all inputs are flattened and concatenated together as the intermediate representation of the partition and action. Multiple fully connected layers, $\{fc_1, fc_2, \cdots\}$, are used for predicting the final output Q.

vectors. The output from $embed_{CA}$ has a dimension of $(n+1)$-by-d_{CA}, where n is the number of tracts or the length of $\mathcal{Z}$. The index of target tract t is encoded into d_{tract}-dimension vector through tract embedding layer $embed_{tract}$. The output from $embed_{tract}$ has dimension of 1-by-d_{tract}. The embeddings of all inputs are flattened and concatenated together as the intermediate representation of the partition and action. Multiple fully connected layers, $\{fc_1, fc_2, \cdots\}$, are used for the prediction task. Finally, the output layer predicts the sigmoid of Q, which is the reward if the given action is taken on given partition. The detailed parameter setting of our DQN model is given later.

Large and dynamic action space. It is difficult to directly calculate the summation of future Q values in Equation (3), and to find the maximum over all actions in Equation (4). The reason is that for different partition states, the tracts on the boundaries are different, and thus the set of possible actions is also different. In this paper, we set the discount factor $\delta = 0$ to ease the calculation of Q. When searching for the best action with Equation (4), we sample a subset of $m = 32$ actions, and find the best action within such subset.

It is worthy to mention that when the discount factor is set to 0, the DQN only accounts for immediate reward instead of the total reward. Even so, the DQN method is still intuitively better than the MCMC method. The reason is that an MCMC approach cannot keep track of the information from rejected historical samples. The DQN method on the other hand can account for the similarity between partitions, and thus potentially avoid to repeatedly try the same action on similar partitions. The DQN is able to learn extra information from historical samples in the form of immediate reward, if not the total reward.

Training overhead is high. To train the Deep Q-learning model, we perform a sequence of random actions to generate a batch of training data. Such training overhead is significantly higher than the MCMC method. To improve the efficiency, we save our DQN model across different tasks and different rounds. The neural network is only updated when the found action cannot improve the cumulative reward.

4 EXPERIMENTS

4.1 Experiment Setting

4.1.1 Data description. The fundamental geographic unit of study in this paper is a tract, which is a small area established by the U.S. Census Bureau. Analysis at the tract level is attractive because it offers the finest level of granularity where demographic data are available.

We collect the *demographic data* of 801 tracts in Chicago from the 2010 US census survey [1]. The demographic data provide the raw count of households over 100 different categories, include ethnicity, income, education, etc. The *geographic boundary information* for 801 tracts are available through the U.S. census survey [1] as well. Each boundary is defined by a polygon with a sequence of GPS coordinates. The Chicago Data Portal [12] makes the Chicago *crime data* publicly available online. Our study focuses on data from 2010 and 2011 in which over 700,000 incidents of crime were recorded. For each crime incident, the date, location, and incident type are reported. *House Price Data* in the Chicago metropolitan area are obtained from Zillow, a popular real estate website [29]. We collect the last sale price, floor size, latitude, and longitude information for over 44, 000 properties that were sold between January 2015 and December 2017.

4.1.2 Prediction Tasks. We employ a negative binomial regression model [17] as our prediction task, namely

$$\mathbf{E}(Y) = \exp(\alpha X),$$

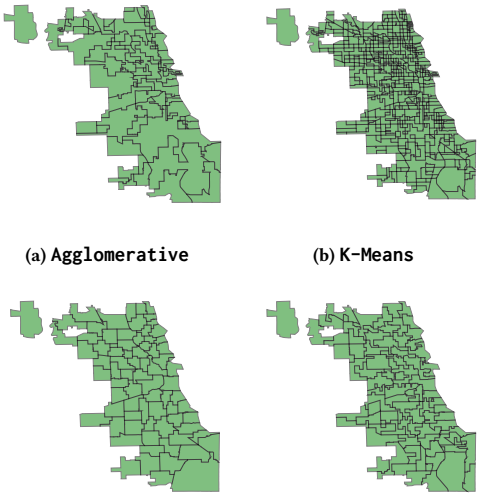
where $\mathbf{E}$ calculates the expectation. The link function used in the regression is a negative binomial distribution. The reason to use negative binomial regression is that 1) it is suitable for non-negative value prediction, and 2) it solves the over-dispersion problem by allowing the variance to be larger than the mean.

The demographics features at the community level are pre-processed as contextual features, X_i, including total population, percentage of household in each income category, diversity of ethnicity, etc.

Two prediction tasks are used to evaluate our region partition methods in the experiments. Both tasks use the same processed demographic features, X. The first task is crime count prediction, where we aggregate the total number of crime in a community as Y. The crime count in year 2010 is used as training, and 2011 data are used as testing. The second task is house price prediction, where we use the average price per square foot in a community as Y. The houses that are sold before August 1st, 2016 are used as training, while the rest are used as testing. The split point makes the training and testing data have roughly equal number of samples.

4.1.3 Compared Methods. The existing administrative boundary is a clear baseline partition. Since our region partition problem is similar to clustering problem, we employ various conventional clustering methods to derive different clustering partitions as alternative baselines. More specifically, we compare the following region partition methods.

- Administrative boundary **(`Admin`)** uses the existing administrative boundary defined by US Bureau of Census [1]. A visual depiction of this partition can be seen in Figure 1.
- Various clustering methods. **`K-Means`** clustering separates tracts into n groups of equal variance. **`Agglomerative`** clustering performs a hierarchical clustering using a bottom up approach. The ward linkage function, along with a tract adjacency graph, are used to guarantee spatial continuity. **`Spectral`** clustering does a low-dimensional embedding of the affinity matrix between



(a) **Agglomerative** (b) **K-Means**

(c) **Spectral** (d) **DQN**

Figure 3: (a - c) The clustering results for three clustering baselines. (d) The learned partition from DQN method for crime prediction task. Note that K-means generate non-connected communities, and thus smaller regions. In all cases $m = 77$.

samples first, and then applies the K-means method. Note that Spectral also takes the tract adjacency graph as the affinity graph.

- Two variants of MCMC method. **Naive** MCMC with a straightforward uniform proposal. **Softmax** MCMC with softmax proposal is another variant, which uses human heuristics.
- Q-learning is our proposed reinforcement learning method to search for an optimal partition. In our **DQN** model, the embedding dimension of community area d_{CA} is 2, while d_{tract} is 4. We employ two fully connected layers for the prediction task, where fc_1 has 200 neurons and fc_2 has 100 neurons.

Since Naive, Softmax, and DQN all learn a region partition based on a supervised task, these methods utilize information from both the feature matrix, X, and the target vector, Y. To ensure a fair comparison between these methods and the conventional clustering methods (Agglomerative, K-means, and Spectral), we run the latter under three different settings: X only, Y only, and both X and Y. In each instance, we set the number of clusters as $m = 77$, which equals the number of community areas in Admin. Note that Agglomerative is a stable clustering method and produces the exact same clustering result each time it is run. Meanwhile, the K-Means and Spectral methods do not always converge to the same partition, and therefore we run each 10 times. Similar to K-means and Spectral, our proposed Naive, Softmax, and DQN may converge to different local optimal partitions, and thus we run our methods 10 times as well. Finally, we compute the mean and standard deviations of their corresponding error measures.

4.1.4 Evaluation Metrics. Given a partition $\mathcal{Z}$, we evaluate the quality of this partition on the testing data set. Mean Absolute Error (MAE) and standard deviations are used to measure the performance of prediction tasks. We compute MAE metric using Leave-one-out Cross-validation.

Table 1: MAE and corresponding standard deviation of various partition methods, averaged over 10 runs. For all deterministic methods, we did not report the standard deviation.

Method	Crime	House price
Admin	1,826.84	31.29
Agglomerative X only	60,126.28	51.28
Agglomerative Y only	14,069.28	41.87
Agglomerative X and Y	147,465.30	51.32
K-means X only	2,513.35 (433.74)	31.74 (1.17)
K-means Y only	1,048.59 (99.47)	32.75 (1.03)
K-means X and Y	2,779.78 (307.68)	32.98 (1.76)
Spectral X only	1,373.07 (36.82)	30.85 (1.36)
Spectral Y only	1,356.17 (48.73)	31.35 (2.09)
Spectral X and Y	1,407.57 (60.21)	32.30 (1.12)
Naive	697.40 (60.21)	33.21 (13.53)
Softmax	671.90 (62.91)	32.99 (4.48)
DQN	**602.93** (91.65)	**22.64** (2.24)

4.2 Quantitative Evaluations

4.2.1 Effectiveness Study. In Table 1, we report the evaluation results of the various partition methods. The partitions from different baselines are visualized in Figure 3(a-c). We run each method 10 times, and report the MAE and corresponding standard deviations in the table. We hold constant the number of MCMC iterations at approximately 1,500. Early stopping is allowed if a convergence criterion (discussed below) is met. The final partition of DQN is visualized in Figure 3(d). Overall, we have the following three observations.

Clustering methods overall perform poorly. This is likely due to the fact that the clustering methods do not fully consider the task information in a supervised way. Even when we consider both $\mathbf{X}$ and $\mathbf{y}$, the clustering methods are outperformed by the proposed methods. In general, Agglomerative results in the highest prediction errors for both crime prediction and house price prediction task. This is likely due to the fact that Agglomerative utilizes tract connectivity as a hard constraint. As a result, the generated communities have a large variance in their sizes, as shown in Figure 3(a). K-Means gives worse result than that of Admin in almost all cases. While the number of communities, m is held constant in all cases, the generated partition of K-Means appears to be greater than $m = 77$ in Figure 3(b) because K-Means does not incorporate spatial continuity constraint. Consequently, one community can consist of several disconnected components. Spectral tends to yield the best results in both tasks among these baselines because it accounts for the affinity of tracts and generates communities with similar sizes. However, it is worth mentioning that Spectral method cannot guarantee the spatial continuity of generated communities. From Figure 3(c), we can also see that Spectral results in a similar partition as the original administrative boundary, shown in Figure 1(a). That is also why it achieves similar prediction accuracy as Admin.

The proposed MCMC method outperforms the baselines. Both variants of MCMC methods significantly outperform Admin and Spectral baselines. Such observations validate the effectiveness

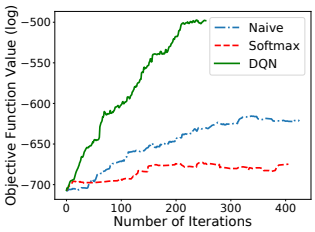


Figure 4: Convergence plots for proposed methods on house price prediction task.

of the MCMC strategy in searching for optimal solutions. On average, `Softmax` tends to outperform `Naive` on both tasks studied. However, it is inconclusive if this is a general result due to the incremental performance difference between the two in certain cases.

`DQN` *method performs the best among all.* It is clear that `DQN` finds a better local optimal solution than that of MCMC methods. On the crime prediction task, the average MAE is 602.93, which represents a 67% improvement over `Admin` baseline. On the house price prediction task, `DQN` consistently gives the best performance. The reason is that `DQN` explores over a subset of actions and picks the best one at each step, compared to the MCMC method, which searches the partition space in a depth first search fashion. Comparing the final partition of `DQN` and `Spectral`, we notice that `Spectral` partition is quite similar to the original administrative boundary in Figure 1. As a consequence, `Spectral` has similar MAE to `Admin`, but is much worse than `DQN`.

4.2.2 Convergence Study. In addition to discussing the proposed methods in the context of model accuracy, we also briefly investigate their performance in terms of convergence. In our method, we track the standard deviation of the $\mathcal{F}$ values from the last 50 iterations. When such standard deviation is less than a pre-defined threshold, we stop. In Figure 4 we visualize the log of quality measure, i.e. $-\mathcal{F}(\mathcal{Z})$, against the number of iterations for three proposed methods on the house price task.

We perform only one run of each method, which could result in slightly different results than what we observe on average. However, Figure 4 is common enough to understand typical convergence behavior. We observe that `DQN` finishes in a less than 300 iterations, while `Naive` and `Softmax` both take more than 400 iterations. Clearly, we observe that `DQN` converge to a better optimal solution with higher training gain. In this specific run, `Naive` is slower to converge than `Softmax`, but also finds a more optimal solution. There is more evidence that these methods may give very similar results, on average.

5 RELATED WORK

Urban Data Heterogeneity. As we collect more types of new urban data, we are able to solve a wide spectrum of urban problems. For example, recent studies use various urban data, such as POI and crowd mobility [19, 20], to estimate air quality [26], noise pollution [27], real estates values [7] and crime [17, 21].

These existing works focus on mining the subtle correlations across different domains of data. We generalize the urban problems above as a learning task, f, which maps some urban features to a target variable of interest. We study how to find a proper region partition as the domain of urban problem, f, so that the learned correlation is consistent and significant.

Traditional Region Partition Methods. There are mainly four types of region partition methods that are widely used in existing urban computing literature. 1) A fixed sized grid [18, 22, 26], 2) existing administrative boundaries [17], 3) clustering of point-wise urban data to get regions [10], and 4) other specifically designed partitions, such as road networks partition [24], cellular tower coverage partition [23], and fan-shaped partitions [28].

It is worthy mentioning that most existing partition methods are purely based on cartographic information, and do not make use of the urban data properties. In this paper, we try to partition the city with an explicit objective, so that the partition methods explicitly take the learning task f into consideration.

Partition Problem in Other Contexts. In the field of community detection problem [5, 6], the partition problem is well studied as well. While most of these works do not have an extra objective for partition, a few of them involves additional property as constraint. For example, the homophilic network decomposition [13] partitions the networks while characterize the degree of homophily of its nodes. The authors assign a dominant label within each group. Hence, the problem is essentially a classification problem, which is fundamentally different from ours.

Another general problem is to partition a set into k groups and fit a model within each group [25]. Such problem definition aims to show that the correlations between features and target vary over the space (e.g., in some area, disadvantage index correlates with crime; while in some other areas, it does not). A simplified version of such problem is segmented regression in 2D space, which could be solved efficiently with dynamic programming [4]. Compared to our problem, such problem formulation has different goal while being easier to solve.

Discrete Optimizations. Our problem is a discrete optimization problem. The objective is easy to derive but difficult to find an optimal solution for. MCMC sampling has been shown to be effective in optimizing discrete structures [14]. To improve the efficiency of MCMC, the learning to optimize technique [9] employs reinforcement learning framework to learn where to draw the next sample. Our solution is mostly inspired by this work.

6 CONCLUSION

In this paper, we proposed a new problem called task-specific region partition. The problem is motivated by the drawbacks of existing administrative boundaries. The task-specific region partition problem is NP-hard, and hence searching for a global optimum is difficult. Two variants of MCMC methods and a reinforcement learning method are proposed to efficiently search for an optimal solution. Our methods are evaluated on two prediction tasks using two evaluation metrics. The learned predictions consistently outperform the administrative boundaries in both tasks.

ACKNOWLEDGMENTS

The work was supported in part by NSF awards #1652525, #1618448, and #1639150. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] 2010. United States Census Bureau. Demographics Survey. http://www.census.gov

[2] Konstantin Andreev and Harald Racke. 2006. Balanced graph partitioning. *Theory of Computing Systems* 39, 6 (2006), 929–939.

[3] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. 2003. An introduction to MCMC for machine learning. *Machine learning* 50, 1-2 (2003), 5–43.

[4] Richard Bellman and Robert Roth. 1969. Curve fitting by segmented straight lines. *J. Amer. Statist. Assoc.* 64, 327 (1969), 1079–1084.

[5] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* 486, 3-5 (2010), 75–174.

[6] Santo Fortunato and Marc Barthelemy. 2007. Resolution limit in community detection. *Proceedings of the National Academy of Sciences* 104, 1 (2007), 36–41.

[7] Yanjie Fu, Yong Ge, Yu Zheng, Zijun Yao, Yanchi Liu, Hui Xiong, and Jing Yuan. 2014. Sparse real estate ranking with online user reviews and offline moving behaviors. In *Data Mining (ICDM), 2014 IEEE International Conference on*. IEEE, 120–129.

[8] Corina Graif, Andrew S Gladfelter, and Stephen A Matthews. 2014. Urban poverty and neighborhood effects on crime: Incorporating spatial and network perspectives. *Sociology compass* 8, 9 (2014), 1140–1155.

[9] Ke Li and Jitendra Malik. 2016. Learning to optimize. *arXiv preprint arXiv:1606.01885* (2016).

[10] Yexin Li, Yu Zheng, Huichu Zhang, and Lei Chen. 2015. Traffic prediction in a bike-sharing system. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 33.

[11] Kevin P. Murphy. 2012. *Machine Learning: A Probabilistic Perspective*. The MIT Press, Cambridge, Massachusetts.

[12] City of Chicago data portal. 2015. https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2.

[13] Giulio Rossetti, Luca Pappalardo, Riivo Kikas, Dino Pedreschi, Fosca Giannotti, and Marlon Dumas. 2016. Homophilic network decomposition: a community-centric analysis of online social services. *Social Network Analysis and Mining* 6, 1 (2016), 103.

[14] Malcolm Strens. 2003. Evolutionary MCMC sampling and optimization in discrete spaces. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 736–743.

[15] Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction*. Vol. 1. MIT press Cambridge.

[16] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep Reinforcement Learning with Double Q-Learning.. In *AAAI*, Vol. 16. 2094–2100.

[17] Hongjian Wang, Daniel Kifer, Corina Graif, and Zhenhui Li. 2016. Crime rate inference with big data. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 635–644.

[18] Hongjian Wang, Yu-Hsuan Kuo, Daniel Kifer, and Zhenhui Li. 2016. A simple baseline for travel time estimation using large-scale trip data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 61.

[19] Hongjian Wang and Zhenhui Li. 2017. Region Representation Learningvia Mobility Flow. In *In Proceedings of CIKM'17*. CIKM'17, 10 pages.

[20] Hongjian Wang, Xianfeng Tang, Yu-Hsuan Kuo, Daniel Kifer, and Zhenhui Li. 2019. A Simple Baseline for Travel Time Estimation Using Large-scale Trip Data. *ACM Trans. Intell. Syst. Technol.* 10, 2, Article 19 (Jan. 2019), 22 pages. https://doi.org/10.1145/3293317

[21] Hongjian Wang, Huaxiu Yao, Daniel Kifer, Corina Graif, and Zhenhui Li. 2017. Non-Stationary Model for Crime Rate Inference Using Modern Urban Data. *IEEE Transactions on Big Data* (2017).

[22] Fei Wu, Hongjian Wang, and Zhenhui Li. 2016. Interpreting traffic dynamics using ubiquitous urban data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 69.

[23] Fengli Xu, Yong Li, Huandong Wang, Pengyu Zhang, and Depeng Jin. 2017. Understanding mobile traffic patterns of large scale cellular towers in urban environment. *IEEE/ACM Transactions on Networking* 25, 2 (2017), 1147–1161.

[24] Jing Yuan, Yu Zheng, and Xing Xie. 2012. Discovering regions of different functions in a city using human mobility and POIs. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 186–194.

[25] Bin Zhang. 2003. Regression clustering. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE, 451–458.

[26] Yu Zheng, Furui Liu, and Hsun-Ping Hsieh. 2013. U-air: When urban air quality inference meets big data. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1436–1444.

[27] Yu Zheng, Tong Liu, Yilun Wang, Yanmin Zhu, Yanchi Liu, and Eric Chang. 2014. Diagnosing New York city's noises with ubiquitous data. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 715–725.

[28] Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. 2015. Forecasting fine-grained air quality based on big data. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2267–2276.

[29] Zillow.com. 2017. Real Estate Value in Chicago. https://www.zillow.com/.

Knowledge Graph Convolutional Networks for Recommender Systems

Hongwei Wang*
Shanghai Jiao Tong University
Shanghai, China
wanghongwei55@gmail.com

Miao Zhao
The Hong Kong Polytechnic University, Hong Kong, China
csmiaozhao@comp.polyu.edu.hk

Xing Xie
Microsoft Research Asia
Beijing, China
xingx@microsoft.com

Wenjie Li
The Hong Kong Polytechnic University, Hong Kong, China
cswjli@comp.polyu.edu.hk

Minyi Guo†
Shanghai Jiao Tong University
Shanghai, China
guo-my@cs.sjtu.edu.cn

ABSTRACT

To alleviate sparsity and cold start problem of collaborative filtering based recommender systems, researchers and engineers usually collect attributes of users and items, and design delicate algorithms to exploit these additional information. In general, the attributes are not isolated but connected with each other, which forms a knowledge graph (KG). In this paper, we propose **K**nowledge **G**raph **C**onvolutional **N**etworks (KGCN), an end-to-end framework that captures inter-item relatedness effectively by mining their associated attributes on the KG. To automatically discover both high-order structure information and semantic information of the KG, we sample from the neighbors for each entity in the KG as their receptive field, then combine neighborhood information with bias when calculating the representation of a given entity. The receptive field can be extended to multiple hops away to model high-order proximity information and capture users' potential long-distance interests. Moreover, we implement the proposed KGCN in a minibatch fashion, which enables our model to operate on large datasets and KGs. We apply the proposed model to three datasets about movie, book, and music recommendation, and experiment results demonstrate that our approach outperforms strong recommender baselines.

KEYWORDS

Recommender systems; Knowledge graph; Graph convolutional networks

ACM Reference Format:
Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge Graph Convolutional Networks for Recommender Systems. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313417

*This work is done during the author's internship at Microsoft Research Asia.
†Minyi Guo is the corresponding author. This work was partially sponsored by the National Basic Research 973 Program of China (2015CB352403) and National Natural Science Foundation of China (61272291).

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313417

1 INTRODUCTION

With the advance of Internet technology, people can access a vast amount of online content, such as news [25], movies [5], and commodities [26]. A notorious problem with online platforms is that the volume of items can be overwhelming to users. To alleviate the impact of information overloading, recommender systems (RS) is proposed to search for and recommend a small set of items to meet users' personalized interests.

A traditional recommendation technique is collaborative filtering (CF), which assigns users and items ID-based representation vectors, then models their interactions by specific operation such as inner product [16] or neural networks [8]. However, CF-based methods usually suffer from sparsity of user-item interactions and the cold start problem. To address these limitations, researchers usually turn to feature-rich scenarios, where attributes of users and items are used to compensate for the sparsity and improve the performance of recommendation [3, 17].

A few recent studies [9, 18, 19, 22–24] have gone a step further than simply using attributes: They point out that attributes are not isolated but linked up with each other, which forms a *knowledge graph* (KG). Typically, a KG is a directed heterogeneous graph in which nodes correspond to *entities* (items or item attributes) and edges correspond to *relations*. Compared with KG-free methods, incorporating KG into recommendation benefits the results in three ways [18]: (1) The rich semantic relatedness among items in a KG can help explore their latent connections and improve the *precision* of results; (2) The various types of relations in a KG are helpful for extending a user's interests reasonably and increasing the *diversity* of recommended items; (3) KG connects a user's historically-liked and recommended items, thereby bringing *explainability* to recommender systems.

Despite the above benefits, utilizing KG in RS is rather challenging due to its high dimensionality and heterogeneity. One feasible way is to preprocess the KG by *knowledge graph embedding* (KGE) methods [20], which map entities and relations to low-dimensional representation vectors [9, 19, 23]. However, commonly-used KGE methods focus on modeling rigorous semantic relatedness (e.g., TransE [1] and TransR [12] assume $head + relation = tail$), which are more suitable for in-graph applications such as KG completion and link prediction rather than recommendation. A more natural and intuitive way is to design a graph algorithm directly to

exploit the KG structure [18, 22, 24]. For example, PER [22] and FMG [24] treat KG as a heterogeneous information network, and extract meta-path/meta-graph based latent features to represent the connectivity between users and items along different types of relation paths/graphs. However, PER and FMG rely heavily on manually designed meta-paths or meta-graphs, which are hardly to be optimal in reality. RippleNet [18] is a memory-network-like model that propagates users' potential preferences in the KG and explores their hierarchical interests. But note that the importance of relations is weakly characterized in RippleNet, because the embedding matrix of a relation $\mathbf{R}$ can hardly be trained to capture the sense of importance in the quadratic form $\mathbf{v}^\top \mathbf{R}\mathbf{h}$ ($\mathbf{v}$ and $\mathbf{h}$ are embedding vectors of two entities). In addition, the size of ripple set may go unpredictably with the increase of the size of KG, which incurs heavy computation and storage overhead.

In this paper, we investigate the problem of KG-aware recommendation. Our design objective is to automatically capture both high-order structure and semantic information in the KG. Inspired by graph convolutional networks (GCN)[1] that try to generalize convolution to the graph domain, we propose **K**nowledge **G**raph **C**onvolutional **N**etworks (KGCN) for recommender systems. The key idea of KGCN is to aggregate and incorporate neighborhood information with bias when calculating the representation of a given entity in the KG. Such a design has two advantages: (1) Through the operation of neighborhood aggregation, the *local proximity structure* is successfully captured and stored in each entity. (2) Neighbors are weighted by scores dependent on the connecting relation and specific user, which characterizes both the *semantic information of KG* and *users' personalized interests in relations*. Note that the size of an entity's neighbors varies and may be prohibitively large in the worst case. Therefore, we sample a fixed-size neighborhood of each node as the receptive field, which makes the cost of KGCN predictable. The definition of neighborhood for a given entity can also be extended hierarchically to multiple hops away to model *high-order* entity dependencies and capture users' potential *long-distance* interests.

Empirically, we apply KGCN to three datasets: MovieLens-20M (movie), Book-Crossing (book), and Last.FM (music). The experiment results show that KGCN achieves average AUC gains of 4.4%, 8.1%, and 6.2% in movie, book, and music recommendations, respectively, compared with state-of-the-art baselines for recommendation.

Our contribution in this paper are summarized as follows:

- We propose knowledge graph convolutional networks, an end-to-end framework that explores users' preferences on the knowledge graph for recommender systems. By extending the receptive field of each entity in the KG, KGCN is able to capture users' high-order personalized interests.
- We conduct experiments on three real-world recommendation scenarios. The results demonstrate the efficacy of KGCN-LS over state-of-the-art baselines.
- We release the code of KGCN and datasets (knowledge graphs) to researchers for validating the reported results and conducting further research. The code and the data are available at https://github.com/hwwang55/KGCN.

[1]We will revisit GCN in related work.

2 RELATED WORK

Our method is conceptually inspired by GCN. In general, GCN can be categorized as spectral methods and non-spectral methods. Spectral methods represent graphs and perform convolution in the spectral space. For example, Bruna et al. [2] define the convolution in Fourier domain and calculates the eigendecomposition of the graph Laplacian, Defferrard et al. [4] approximate the convolutional filters by Chebyshev expansion of the graph Laplacian, and Kipf et al. [10] propose a convolutional architecture via a localized first-order approximation of spectral graph convolutions. In contrast, non-spectral methods operate on the original graph directly and define convolution for groups of nodes. To handle the neighborhoods with varying size and maintain the weight sharing property of CNN, researchers propose learning a weight matrix for each node degree [6], extracting locally connected regions from graphs [13], or sampling a fixed-size set of neighbors as the support size [7]. Our work can be seen as a non-spectral method for a special type of graphs (i.e., knowledge graph).

Our method also connects to PinSage [21] and GAT [15]. But note that both PinSage and GAT are designed for homogeneous graphs. The major difference between our work and the literature is that we offer a new perspective for recommender systems with the assistance of a heterogeneous knowledge graph.

3 KNOWLEDGE GRAPH CONVOLUTIONAL NETWORKS

In this section, we introduce the proposed KGCN model. We first formulate the knowledge-graph-aware recommendation problem. Then we present the design of a single layer of KGCN. At last, we introduce the complete learning algorithm for KGCN, as well as its minibatach implementation.

3.1 Problem Formulation

We formulate the knowledge-graph-aware recommendation problem as follows. In a typical recommendation scenario, we have a set of M users $\mathcal{U} = \{u_1, u_2, ..., u_M\}$ and a set of N items $\mathcal{V} = \{v_1, v_2, ..., v_N\}$. The user-item interaction matrix $\mathbf{Y} \in \mathbb{R}^{M \times N}$ is defined according to users' implicit feedback, where $y_{uv} = 1$ indicates that user u engages with item v, such as clicking, browsing, or purchasing; otherwise $y_{uv} = 0$. Additionally, we also have a knowledge graph $\mathcal{G}$, which is comprised of entity-relation-entity triples (h, r, t). Here $h \in \mathcal{E}$, $r \in \mathcal{R}$, and $t \in \mathcal{E}$ denote the head, relation, and tail of a knowledge triple, $\mathcal{E}$ and $\mathcal{R}$ are the set of entities and relations in the knowledge graph, respectively. For example, the triple (*A Song of Ice and Fire*, *book.book.author*, *George Martin*) states the fact that George Martin writes the book "A Song of Ice and Fire". In many recommendation scenarios, an item $v \in \mathcal{V}$ corresponds to one entity $e \in \mathcal{E}$. For example, in book recommendation, the item "A Song of Ice and Fire" also appears in the knowledge graph as an entity with the same name.

Given the user-item interaction matrix $\mathbf{Y}$ as well as the knowledge graph $\mathcal{G}$, we aim to predict whether user u has potential interest in item v with which he has had no interaction before. Our goal is to learn a prediction function $\hat{y}_{uv} = \mathcal{F}(u, v|\Theta, \mathbf{Y}, \mathcal{G})$, where $\hat{y}_{uv}$ denotes the probability that user u will engage with item v, and Θ denotes the model parameters of function $\mathcal{F}$.

3.2 KGCN Layer

KGCN is proposed to capture high-order structural proximity among entities in a knowledge graph. We start by describing a single KGCN layer in this subsection. Consider a candidate pair of user u and item (entity) v. We use $\mathcal{N}(v)$ to denote the set of entities directly connected to v,[2] and r_{e_i,e_j} to denote the relation between entity e_i and e_j. We also use a function $g : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ (e.g., inner product) to compute the score between a user and a relation:

$$\pi_r^u = g(\mathbf{u}, \mathbf{r}), \tag{1}$$

where $\mathbf{u} \in \mathbb{R}^d$ and $\mathbf{r} \in \mathbb{R}^d$ are the representations of user u and relation r, respectively, d is the dimension of representations. In general, π_r^u characterizes the importance of relation r to user u. For example, a user may have more potential interests in the movies that share the same "star" with his historically liked ones, while another user may be more concerned about the "genre" of movies.

To characterize the topological proximity structure of item v, we compute the linear combination of v's neighborhood:

$$\mathbf{v}_{\mathcal{N}(v)}^u = \sum_{e \in \mathcal{N}(v)} \tilde{\pi}_{r_{v,e}}^u \mathbf{e}, \tag{2}$$

where $\tilde{\pi}_{r_{v,e}}^u$ is the normalized user-relation score

$$\tilde{\pi}_{r_{v,e}}^u = \frac{\exp(\pi_{r_{v,e}}^u)}{\sum_{e \in \mathcal{N}(v)} \exp(\pi_{r_{v,e}}^u)}, \tag{3}$$

and $\mathbf{e}$ is the representation of entity e. User-relation scores act as *personalized filters* when computing an entity's neighborhood representation, since we aggregate the neighbors with bias with respect to these user-specific scores.

In a real-world knowledge graph, the size of $\mathcal{N}(e)$ may vary significantly over all entities. To keep the computational pattern of each batch fixed and more efficient, we uniformly sample a fixed-size set of neighbors for each entity instead of using its full neighbors. Specifically, we compute the neighborhood representation of entity v as $\mathbf{v}_{\mathcal{S}(v)}^u$, where $\mathcal{S}(v) \triangleq \{e \mid e \sim \mathcal{N}(v)\}$ and $|\mathcal{S}(v)| = K$ is a configurable constant.[3] In KGCN, $\mathcal{S}(v)$ is also called the (single-layer) *receptive field* of entity v, as the final representation of v is sensitive to these locations. Figure 1a gives an illustrative example of a two-layer receptive field for a given entity, where K is set as 2.

The final step in a KGCN layer is to aggregate the entity representation $\mathbf{v}$ and its neighborhood representation $\mathbf{v}_{\mathcal{S}(v)}^u$ into a single vector. We implement three types of aggregators $agg : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ in KGCN:

- *Sum aggregator* takes the summation of two representation vectors, followed by a nonlinear transformation:

$$agg_{sum} = \sigma\left(\mathbf{W} \cdot (\mathbf{v} + \mathbf{v}_{\mathcal{S}(v)}^u) + \mathbf{b}\right), \tag{4}$$

  where $\mathbf{W}$ and $\mathbf{b}$ are transformation weight and bias, respectively, and σ is the nonlinear function such *ReLU*.
- *Concat aggregator* [7] concatenates the two representation vectors first before applying nonlinear transformation:

$$agg_{concat} = \sigma\left(\mathbf{W} \cdot concat(\mathbf{v}, \mathbf{v}_{\mathcal{S}(v)}^u) + \mathbf{b}\right). \tag{5}$$

[2]The knowledge graph $\mathcal{G}$ is treated undirected.
[3]Technically, $\mathcal{S}(v)$ may contain duplicates if $\mathcal{N}(v) < K$.

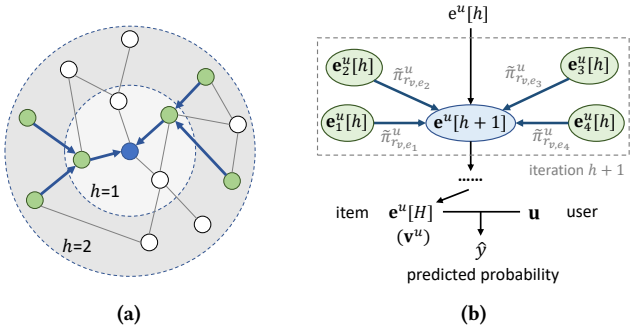


Figure 1: (a) A two-layer receptive field (green entities) of the blue entity in a KG. (b) The framework of KGCN.

- *Neighbor aggregator* [15] directly takes the neighborhood representation of entity v as the output representation:

$$agg_{neighbor} = \sigma\left(\mathbf{W} \cdot \mathbf{v}_{\mathcal{S}(v)}^u + \mathbf{b}\right). \tag{6}$$

Aggregation is a key step in KGCN, because the representation of an item is bound up with its neighbors by aggregation. We will evaluate the three aggregators in experiments.

3.3 Learning Algorithm

Through a single KGCN layer, the final representation of an entity is dependent on itself as well as its immediate neighbors, which we name 1-order entity representation. It is natural to extend KGCN from one layer to multiple layers to reasonably explore users' potential interests in a broader and deeper way. The technique is intuitive: Propagating the initial representation of each entity (0-order representation) to its neighbors leads to 1-order entity representation, then we can repeat this procedure, i.e., further propagating and aggregating 1-order representations to obtain 2-order ones. Generally speaking, the h-order representation of an entity is a mixture of initial representations of itself and its neighbors up to h hops away. This is an important property for KGCN, which we will discuss in the next subsection.

The formal description of the above steps is presented in Algorithm 1. H denotes the maximum depth of receptive field (or equivalently, the number of aggregation iterations), and a suffix $[h]$ attached by a representation vector denotes h-order. For a given user-item pair (u, v) (line 2), we first calculate the receptive field $\mathcal{M}$ of v in an iterative layer-by-layer manner (line 3, 13-19). Then the aggregation is repeated H times (line 5): In iteration h, we calculate the neighborhood representation of each entity $e \in \mathcal{M}[h]$ (line 7), then aggregate it with its own representation $\mathbf{e}^u[h-1]$ to obtain the one to be used at the next iteration (line 8). The final H-order entity representation is denoted as $\mathbf{v}^u$ (line 9), which is fed into a function $f : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ together with user representation $\mathbf{u}$ for predicting the probability:

$$\hat{y}_{uv} = f(\mathbf{u}, \mathbf{v}^u). \tag{7}$$

Figure 1b illustrates the KGCN algorithm in one iteration, in which the entity representation $\mathbf{v}^u[h]$ and neighborhood representations (green nodes) of a given node are mixed to form its representation for the next iteration (blue node).

Algorithm 1: KGCN algorithm

Input: Interaction matrix $\mathbf{Y}$; knowledge graph $\mathcal{G}(\mathcal{E}, \mathcal{R})$; neighborhood sampling mapping $\mathcal{S} : e \to 2^{\mathcal{E}}$; trainable parameters: $\{\mathbf{u}\}_{u\in\mathcal{U}}$, $\{\mathbf{e}\}_{e\in\mathcal{E}}$, $\{\mathbf{r}\}_{r\in\mathcal{R}}$, $\{\mathbf{W}_i, \mathbf{b}_i\}_{i=1}^{H}$; hyper-parameters: H, d, $g(\cdot)$, $f(\cdot)$, $\sigma(\cdot)$, $agg(\cdot)$

Output: Prediction function $\mathcal{F}(u, v|\Theta, \mathbf{Y}, \mathcal{G})$

1 **while** *KGCN not converge* **do**
2 **for** (u, v) *in* $\mathbf{Y}$ **do**
3 $\{\mathcal{M}[i]\}_{i=0}^{H} \leftarrow$ Get-Receptive-Field (v);
4 $\mathbf{e}^u[0] \leftarrow \mathbf{e}, \forall e \in \mathcal{M}[0]$;
5 **for** $h = 1, ..., H$ **do**
6 **for** $e \in \mathcal{M}[h]$ **do**
7 $\mathbf{e}^u_{\mathcal{S}(e)}[h-1] \leftarrow \sum_{e'\in\mathcal{S}(e)} \tilde{\pi}^u_{r_{e,e'}} \mathbf{e}'^u[h-1]$;
8 $\mathbf{e}^u[h] \leftarrow agg\left(\mathbf{e}^u_{\mathcal{S}(e)}[h-1], \mathbf{e}^u[h-1]\right)$;
9 $\mathbf{v}^u \leftarrow \mathbf{e}^u[H]$;
10 Calculate predicted probability $\hat{y}_{uv} = f(\mathbf{u}, \mathbf{v}^u)$;
11 Update parameters by gradient descent;
12 **return** $\mathcal{F}$;
13 **Function** Get-Receptive-Field (v)
14 $\mathcal{M}[H] \leftarrow v$;
15 **for** $h = H-1, ..., 0$ **do**
16 $\mathcal{M}[h] \leftarrow \mathcal{M}[h+1]$;
17 **for** $e \in \mathcal{M}[h+1]$ **do**
18 $\mathcal{M}[h] \leftarrow \mathcal{M}[h] \cup \mathcal{S}(e)$;
19 **return** $\{\mathcal{M}[i]\}_{i=0}^{H}$;

Note that Algorithm 1 traverses all possible user-item pairs (line 2). To make computation more efficient, we use a negative sampling strategy during training. The complete loss function is as follows:

$$\mathcal{L} = \sum_{u\in\mathcal{U}} \left(\sum_{v:y_{uv}=1} \mathcal{J}(y_{uv}, \hat{y}_{uv}) - \sum_{i=1}^{T^u} \mathbb{E}_{v_i\sim P(v_i)} \mathcal{J}(y_{uv_i}, \hat{y}_{uv_i}) \right) + \lambda \|\mathcal{F}\|_2^2, \tag{8}$$

where $\mathcal{J}$ is cross-entropy loss, P is a negative sampling distribution, and T^u is the number of negative samples for user u. In this paper, $T^u = |\{v : y_{uv} = 1\}|$ and P follows a uniform distribution. The last term is the L2-regularizer.

4 EXPERIMENTS

In this section, we evaluate KGCN on three real-world scenarios: movie, book, and music recommendations.

4.1 Datasets

We utilize the following three datasets in our experiments for movie, book, and music recommendation, respectively:

- **MovieLens-20M**[4] is a widely used benchmark dataset in movie recommendations, which consists of approximately 20 million explicit ratings (ranging from 1 to 5) on the MovieLens website.

[4]https://grouplens.org/datasets/movielens/

Table 1: Basic statistics and hyper-parameter settings for the three datasets (K: neighbor sampling size, d: dimension of embeddings, H: depth of receptive field, λ: L2 regularizer weight, η: learning rate).

	MovieLens-20M	Book-Crossing	Last.FM
# users	138,159	19,676	1,872
# items	16,954	20,003	3,846
# interactions	13,501,622	172,576	42,346
# entities	102,569	25,787	9,366
# relations	32	18	60
# KG triples	499,474	60,787	15,518
K	4	8	8
d	32	64	16
H	2	1	1
λ	10^{-7}	2×10^{-5}	10^{-4}
η	2×10^{-2}	2×10^{-4}	5×10^{-4}
batch size	65,536	256	128

- **Book-Crossing**[5] contains 1 million ratings (ranging from 0 to 10) of books in the Book-Crossing community.
- **Last.FM**[6] contains musician listening information from a set of 2 thousand users from Last.fm online music system.

Since the three datasets are explicit feedbacks, we transform them into implicit feedback where each entry is marked with 1 indicating that the user has rated the item positively, and sample an unwatched set marked as 0 for each user. The threshold of positive rating is 4 for MovieLens-20M, while no threshold is set for Book-Crossing and Last.FM due to their sparsity.

We use Microsoft Satori[7] to construct the knowledge graph for each dataset. We first select a subset of triples from the whole KG with a confidence level greater than 0.9. Given the sub-KG, we collect Satori IDs of all valid movies/books/musicians by matching their names with tail of triples *(head, film.film.name, tail)*, *(head, book.book.title, tail)*, or *(head, type.object.name, tail)*. Items with multiple matched or no matched entities are excluded for simplicity. We then match the item IDs with the head of all triples and select all well-matched triples from the sub-KG. The basic statistics of the three datasets are presented in Table 1.

4.2 Baselines

We compare the proposed KGCN with the following baselines, in which the first two baselines are KG-free while the rest are all KG-aware methods. Hyper-parameter settings for baselines are introduced in the next subsection.

- **SVD** [11] is a classic CF-based model using inner product to model user-item interactions.[8]
- **LibFM** [14] is a feature-based factorization model in CTR scenarios. We concatenate user ID and item ID as input for LibFM.

[5]http://www2.informatik.uni-freiburg.de/~cziegler/BX/
[6]https://grouplens.org/datasets/hetrec-2011/
[7]https://searchengineland.com/library/bing/bing-satori
[8]We have tried NCF [8], i.e., replacing inner product with neural networks, but the result is inferior to SVD. Since SVD and NCF are similar, we only present the better one here.

Table 2: The results of *AUC* and *F1* in CTR prediction.

Model	MovieLens-20M		Book-Crossing		Last.FM	
	AUC	*F1*	*AUC*	*F1*	*AUC*	*F1*
SVD	0.963 (-1.5%)	0.919 (-1.4%)	0.672 (-8.9%)	0.635 (-7.7%)	0.769 (-3.4%)	0.696 (-3.5%)
LibFM	0.959 (-1.9%)	0.906 (-2.8%)	0.691 (-6.4%)	0.618 (-10.2%)	0.778 (-2.3%)	0.710 (-1.5%)
LibFM + TransE	0.966 (-1.2%)	0.917 (-1.6%)	0.698 (-5.4%)	0.622 (-9.6%)	0.777 (-2.4%)	0.709 (-1.7%)
PER	0.832 (-14.9%)	0.788 (-15.5%)	0.617 (-16.4%)	0.562 (-18.3%)	0.633 (-20.5%)	0.596 (-17.3%)
CKE	0.924 (-5.5%)	0.871 (-6.5%)	0.677 (-8.3%)	0.611 (-11.2%)	0.744 (-6.5%)	0.673 (-6.7%)
RippleNet	0.968 (-1.0%)	0.912 (-2.1%)	0.715 (-3.1%)	0.650 (-5.5%)	0.780 (-2.0%)	0.702 (-2.6%)
KGCN-sum	**0.978**	**0.932***	**0.738**	**0.688***	0.794 (-0.3%)	0.719 (-0.3%)
KGCN-concat	0.977 (-0.1%)	0.931 (-0.1%)	0.734 (-0.5%)	0.681 (-1.0%)	**0.796***	**0.721***
KGCN-neighbor	0.977 (-0.1%)	**0.932***	0.728 (-1.4%)	0.679 (-1.3%)	0.781 (-1.9%)	0.699 (-3.1%)
KGCN-avg	0.975 (-0.3%)	0.929 (-0.3%)	0.722 (-2.2%)	0.682 (-0.9%)	0.774 (-2.8%)	0.692 (-4.0%)

* Statistically significant improvement by unpaired two-sample t-test with $p = 0.1$.

- **LibFM + TransE** extends LibFM by attaching an entity representation learned by TransE [1] to each user-item pair.
- **PER** [22] treats the KG as heterogeneous information networks and extracts meta-path based features to represent the connectivity between users and items.
- **CKE** [23] combines CF with structural, textual, and visual knowledge in a unified framework for recommendation. We implement CKE as CF plus a structural knowledge module in this paper.
- **RippleNet** [18] is a memory-network-like approach that propagates users' preferences on the KG for recommendation.

4.3 Experiments Setup

In KGCN, we set functions g and f as inner product, σ as *ReLU* for non-last-layer aggregator and *tanh* for last-layer aggregator. Other hyper-parameter settings are provided in Table 1. The hyper-parameters are determined by optimizing *AUC* on a validation set. For each dataset, the ratio of training, evaluation, and test set is $6:2:2$. Each experiment is repeated 3 times, and the average performance is reported. We evaluate our method in two experiment scenarios: (1) In click-through rate (CTR) prediction, we apply the trained model to predict each interaction in the test set. We use *AUC* and *F1* to evaluate CTR prediction. (2) In top-K recommendation, we use the trained model to select K items with highest predicted click probability for each user in the test set, and choose *Recall@K* to evaluate the recommended sets. All trainable parameters are optimized by Adam algorithm. The code of KGCN-LS is implemented under Python 3.6, TensorFlow 1.12.0, and NumPy 1.14.3.

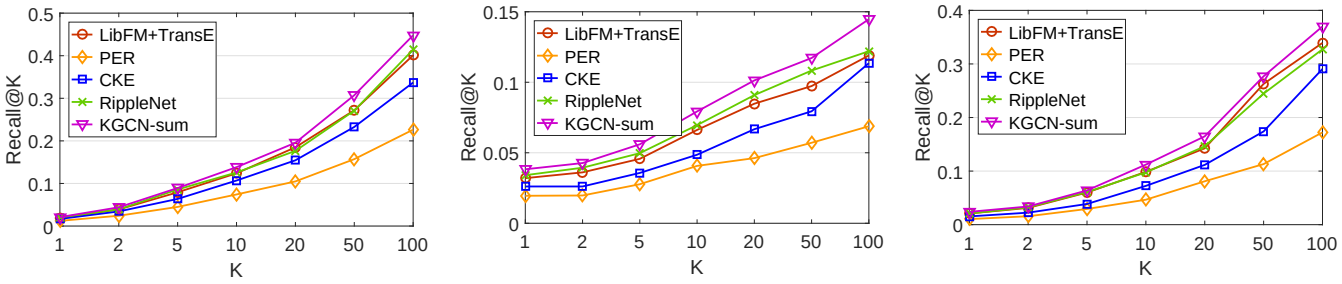
The hyper-parameter settings for baselines are as follows. For SVD, we use the unbiased version (i.e., the predicted rating is modeled as $r_{pq} = \mathbf{p}^\top \mathbf{q}$). The dimension and learning rate for the four datasets are set as: $d = 8$, $\eta = 0.5$ for MovieLens-20M, Book-Crossing; $d = 8$, $\eta = 0.1$ for Last.FM. For LibFM, the dimension is $\{1, 1, 8\}$ and the number of training epochs is 50. The dimension of TransE is 32. For PER, we use manually designed user-item-attribute-item paths as features (i.e., "user-movie-director-movie", "user-movie-genre-movie", and "user-movie-star-movie" for MovieLens-20M; "user-book-author-book" and "user-book-genre-book" for Book-Crossing, "user-musician-date_of_birth-musician" (date of birth is discretized), "user-musician-country-musician", and "user-musician-genre-musician" for Last.FM). For CKE, the dimension of the three datasets are 64, 128, 64. The training weight for KG part is 0.1 for all datasets. The learning rate are the same as in SVD. For RippleNet, $d = 8$, $H = 2$, $\lambda_1 = 10^{-6}$, $\lambda_2 = 0.01$, $\eta = 0.01$ for MovieLens-20M; $d = 16$, $H = 3$, $\lambda_1 = 10^{-5}$, $\lambda_2 = 0.02$, $\eta = 0.005$ for Last.FM. Other hyper-parameters are the same as reported in their original papers or as default in their codes.

4.4 Results

The results of CTR prediction and top-K recommendation are presented in Table 2 and Figure 2, respectively (SVD, LibFM and other variants of KGCN are not plotted in Figure 2 for clarity). We have the following observations:

- In general, we find that the improvements of KGCN on book and music are higher than movie. This demonstrates that KGCN can well address sparse scenarios, since Book-Crossing and Last.FM are much sparser than MovieLens-20M.
- The performance of KG-free baselines, SVD and LibFM, are actually better than the two KG-aware baselines PER and CKE, which indicates that PER and CKE cannot make full use of the KG with manually designed meta-paths and TransR-like regularization.
- LibFM + TransE is better than LibFM in most cases, which demonstrates that the introduction of KG is helpful for recommendation in general.
- PER performs worst among all baselines, since it is hard to define optimal meta-paths in reality.
- RippleNet shows strong performance compared with other baselines. Note that RippleNet also uses multi-hop neighborhood structure, which interestingly shows that capturing proximity information in the KG is essential for recommendation.

The last four rows in Table 2 summarize the performance of KGCN variants. The first three (sum, concat, neighbor) correspond to different aggregators introduced in the preceding section, while the last variant KGCN-avg is a reduced case of KGCN-sum where



(a) MovieLens-20M **(b) Book-Crossing** **(c) Last.FM**

Figure 2: The results of *Recall@K* in top-K recommendation.

Table 3: *AUC* result of KGCN with different neighbor sampling size K.

K	2	4	8	16	32	64
MovieLens-20M	0.978	**0.979**	0.978	0.978	0.977	0.978
Book-Crossing	0.680	0.727	**0.736**	0.725	0.711	0.723
Last.FM	0.791	0.794	**0.795**	0.793	0.794	0.792

Table 4: *AUC* result of KGCN with different depth of receptive field H.

H	1	2	3	4
MovieLens-20M	0.972	**0.976**	0.974	0.514
Book-Crossing	**0.738**	0.731	0.684	0.547
Last.FM	**0.794**	0.723	0.545	0.534

neighborhood representations are directly averaged without user-relation scores (i.e., $\mathbf{v}^u_{\mathcal{N}(v)} = \sum_{e \in \mathcal{N}(v)} \mathbf{e}$ instead of Eq. (2)). Therefore, KGCN-avg is used to examine the efficacy of the "attention mechanism". From the results we find that:

- KGCN outperforms all baselines by a significant margin, while their performances are slightly distinct: KGCN-sum performs best in general, while the performance of KGCN-neighbor shows a clear gap on Book-Crossing and Last.FM. This may be because the neighbor aggregator uses the neighborhood representation only, thus losing useful information from the entity itself.
- KGCN-avg performs worse than KGCN-sum, especially in Book-Crossing and Last.FM where interactions are sparse. This demonstrates that capturing users' personalized preferences and semantic information of the KG do benefit the recommendation.

4.4.1 Impact of neighbor sampling size. We vary the size of sampled neighbor K to investigate the efficacy of usage of the KG. From Table 3 we observe that KGCN achieves the best performance when $K = 4$ or 8. This is because a too small K does not have enough capacity to incorporate neighborhood information, while a too large K is prone to be misled by noises.

4.4.2 Impact of depth of receptive field. We investigate the influence of depth of receptive field in KGCN by varying H from 1 to 4. The results are shown in Table 4, which demonstrate that KGCN is more sensitive to H compared to K. We observe the occurrence of serious model collapse when $H = 3$ or 4, as a larger H brings massive noises to the model. This is also in accordance with our intuition, since a too long relation-chain makes little sense when inferring inter-item similarities. An H of 1 or 2 is enough for real cases according to the experiment results.

Table 5: *AUC* result of KGCN with different dimension of embedding.

d	4	8	16	32	64	128
MovieLens-20M	0.968	0.970	0.975	**0.977**	0.973	0.972
Book-Crossing	0.709	0.732	0.733	0.735	**0.739**	0.736
Last.FM	0.789	0.793	**0.797**	0.793	0.790	0.789

4.4.3 Impact of dimension of embedding. Lastly, we examine the influence of dimension of embedding d on performance of KGCN. The result in Table 5 is rather intuitive: Increasing d initially can boost the performance since a larger d can encode more information of users and entities, while a too large d adversely suffers from overfitting.

5 CONCLUSIONS AND FUTURE WORK

This paper proposes knowledge graph convolutional networks for recommender systems. KGCN extends non-spectral GCN approaches to the knowledge graph by aggregating neighborhood information selectively and biasedly, which is able to learn both structure information and semantic information of the KG as well as users' personalized and potential interests. We also implement the proposed method in a minibatch fashion, which is able to operate on large datasets and knowledge graphs. Through extensive experiments on real-world datasets, KGCN is shown to consistently outperform state-of-the-art baselines in movie, book, and music recommendation.

We point out three avenues for future work. (1) In this work we uniformly sample from the neighbors of an entity to construct its receptive field. Exploring a non-uniform sampler (e.g., importance sampling) is an important direction of future work. (2) This paper (and all literature) focuses on modeling item-end KGs. An interesting direction of future work is to investigate whether leveraging user-end KGs is useful in improving the performance of recommendation. (3) Designing an algorithm to well combine KGs at the two ends is also a promising direction.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*. 2787–2795.

[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. 2014. Spectral networks and locally connected networks on graphs. In *the 2nd International Conference on Learning Representations*.

[3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 7–10.

[4] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*. 3844–3852.

[5] Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J Smola, Jing Jiang, and Chong Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 193–202.

[6] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*. 2224–2232.

[7] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.

[9] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 505–514.

[10] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *the 5th International Conference on Learning Representations*.

[11] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 426–434.

[12] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion.. In *AAAI*, Vol. 15. 2181–2187.

[13] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. 2016. Learning convolutional neural networks for graphs. In *International Conference on Machine Learning*. 2014–2023.

[14] Steffen Rendle. 2012. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3, 3 (2012), 57.

[15] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of the 6th International Conferences on Learning Representations*.

[16] Hongwei Wang, Jia Wang, Miao Zhao, Jiannong Cao, and Minyi Guo. 2017. Joint Topic-Semantic-aware Social Recommendation for Online Voting. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 347–356.

[17] Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. 2018. SHINE: signed heterogeneous information network embedding for sentiment link prediction. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. ACM, 592–600.

[18] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM.

[19] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep Knowledge-Aware Network for News Recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. 1835–1844.

[20] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.

[21] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[22] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*. ACM, 283–292.

[23] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 353–362.

[24] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-graph based recommendation fusion over heterogeneous information networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 635–644.

[25] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A Deep Reinforcement Learning Framework for News Recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. 167–176.

[26] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1059–1068.

Tag2Vec: Learning Tag Representations in Tag Networks

Junshan Wang
Key Laboratory of Machine Perception, Ministry of Education, Peking University
Beijing, China
wangjunshan@pku.edu.cn

Zhicong Lu
Key Laboratory of Machine Perception, Ministry of Education, Peking University
Beijing, China
phyluzhicong@pku.edu.cn

Guojia Song*
Key Laboratory of Machine Perception, Ministry of Education, Peking University
Beijing, China
gjsong@pku.edu.cn

Yue Fan
Key Laboratory of Machine Perception, Ministry of Education, Peking University
Beijing, China
fanyue@pku.edu.cn

Lun Du
Key Laboratory of Machine Perception, Ministry of Education, Peking University
Beijing, China
dulun@pku.edu.cn

Wei Lin
Alibaba
Beijing, China
yangkun.lw@alibaba-inc.com

ABSTRACT

Network embedding is a method to learn low-dimensional representation vectors for nodes in complex networks. In real networks, nodes may have multiple tags but existing methods ignore the abundant semantic and hierarchical information of tags. This information is useful to many network applications and usually very stable. In this paper, we propose a tag representation learning model, Tag2Vec, which mixes nodes and tags into a hybrid network. Firstly, for tag networks, we define semantic distance as the proximity between tags and design a novel strategy, parameterized random walk, to generate context with semantic and hierarchical information of tags adaptively. Then, we propose hyperbolic Skip-gram model to express the complex hierarchical structure better with lower output dimensions. We evaluate our model on the NBER U.S. patent dataset and WordNet dataset. The results show that our model can learn tag representations with rich semantic information and it outperforms other baselines.

CCS CONCEPTS

• **Information systems** → **World Wide Web**; **Deep web**.

KEYWORDS

Network Embedding; Tag Networks; Hyperbolic Spaces

ACM Reference Format:
Junshan Wang, Zhicong Lu, Guojia Song, Yue Fan, Lun Du, and Wei Lin. 2019. Tag2Vec: Learning Tag Representations in Tag Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3308558.3313622

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3308558.3313622

1 INTRODUCTION

Network embedding is a method to learn low-dimensional representation vectors for nodes in complex network and has drawn much attention recently [16, 27, 30]. The representation vectors helps to complete many machine learning tasks, such as node classification [3], node clustering [23] and link prediction [21].

But in real networks, nodes are often attached with multiple tags, and each tag represents a characteristic of the node. For example, there are interested domain tags of researchers in academic networks and discussion topic tags of users in forum networks [20]. Tags have abundant semantic information, which can be used to measure the semantic distances between tags. For example, in academic networks, considering the semantic distance between two interested domain tags, Artificial Intelligence is closer to Machine Learning than Operating System. In addition, tags have abundant hierarchical information. For example, Optics and Transportation are the sub-categories of Mechanical, which should be considered when leaning their representations. But considering tags as discrete features with one-hot encoding in previous work is based on a simple assumption that different features are irrelevant. It will also cause the sparsity of vectors and inefficiency of algorithms.

Therefore, similar to learning the node representation vectors, we want to learn the tag representation vectors in networks, so that the semantic distances between tags can be reflected. On the one hand, because of the semantic and hierarchical information of tag representations, they can help to improve the performance of network applications and study the characteristics and behaviors of tags. On the other hand, as tags are usually stable, we can obtain good representations even though part of networks is lacking.

Existing network embedding methods, such as DeepWalk [27] and LINE [30], only learn the node representations. But tag networks are a special kind of heterogeneous networks which contains the interaction relationship between nodes as well as the affiliation relationship of tags and nodes. There are two main challenges of learning tag representations in the heterogeneous networks:

- *How to learn the semantic information of tags?* If two nodes have tags with similar semantic information, they will have small semantic distance. If two tags have many members with close relationship, they will be close as for semantic

distance. So in tag representation learning, we have to consider the semantic information of nodes and tags. Besides, we need to distinguish the interaction relationship between nodes and the affiliation relationship between tags.

- *How to represent the tags?* In tag representation, we describe the complex traverse and vertical structure of tags with the help of interaction relationships between nodes and affiliation relationships between nodes and tags. Traditional embedding methods learn interaction relationships in Euclidean spaces, but ignore affiliation relationships. If we use tree-like structure to describe the data in Euclidean spaces, it will causes the problem of high computational cost and data sparsity when the hierarchical structure becomes complex.

In this paper, we propose a tag representation learning model, Tag2Vec. We learn the representation vectors of tags to reflect their semantic information and hierarchical structure. Firstly, we mix tags and nodes into a hybrid network to consider the influence of semantic information between tags and nodes. We adopt the parameterized random walk strategy to generate the context that reflects the semantic distance between tags. Then, because of the disability of learning vertical affiliation relationship in Euclidean spaces, we propose the hyperbolic Skip-gram model to learn representation vectors in hyperbolic spaces. Hyperbolic spaces are a kind of metric spaces, and they expand exponentially with regard to the radius r. They confirm to the growth of hierarchical relationship of tags so that learning representations in hyperbolic spaces can better describe the affiliation relationship of tag networks.

The main contribution of this paper is:

- We propose a tag representation learning model Tag2Vec, which learns tag representation vectors in networks to reflect the semantic information of tags.
- We design a parameterized random walk strategy to generate the context with semantic information. It sets parameters to control the proportion of different semantic information.
- We propose hyperbolic Skip-gram model to learn representations in hyperbolic spaces. We improve the representative ability of tag vectors with the help of two advantages of hyperbolic spaces: capable to learn affiliation relationship and reduce the dimensions.

We conduct experiments on two datasets. The results demonstrate that our model can learn representations with rich semantic information of tags and is superior than baselines.

2 RELATED WORK

As tags are usually text or labels, they can be regarded as special nodes in heterogeneous networks or attributes of nodes. Meanwhile, the hierarchical structure of tags can be expressed in hyperbolic spaces. So we discuss related work of three kinds of network embedding methods [9] separately: structure-preserving, attribute-preserving and hyperbolic network embedding.

Many successful network embedding methods have been proposed recently for preserving topological structure [7, 12, 16, 27, 30, 31]. They are working on a homogeneous network but insufficient in networks with tags for the ignorance of tag information. A feasible method is to build heterogeneous networks, where tags are regarded as another type of nodes. Existing heterogeneous network embedding methods can be achieved by partitioning heterogeneous networks into a set of networks [29, 33], establishing a deep learning structure [8, 32], or using meta path based models [10, 13, 19]. However, these methods ignore the higher status of tags in hybrid networks and also leave the hierarchy between tags hidden. Therefore, Tag2Vec is proposed in order to represent tags with both semantic and hierarchical information clearly.

Attribute-preserving network embedding methods embed nodes according to both the graph topology and the node attributes. An attribute of a node can be a text [34], a label [18, 26], or the node content [26, 28]. If two nodes have similar attributes, they will have close embedding vectors. However, attribute-preserving network embedding does not give every attribute an embedding vector and does not learn semantic information between attributes explicitly. Tag2Vec learns the representations of tags and nodes simultaneously, modelling the relationship between nodes and between tags.

Many real world networks actually have hierarchical or tree-like structure [1]. Learning representations in Euclidean spaces requires large dimensions to hold a network with complex hierarchical relationships[6, 24]. Hyperbolic space [4, 15] can reflect the latent hierarchy of embedded nodes with low dimensions. Poincare network embedding [25] embeds words into hyperbolic space in a way such that their distance in the embedding space reflects their semantic similarity. [14] embeds acyclic graphs into hyperbolic space to learn their hierarchical features. But these methods only focus on learning representations in homogeneous networks without tags. In this work, we learn the representations of both nodes and tags under the hyperbolic space.

3 PROBLEM DEFINITION

In this section, we define some problems in tag embedding formally.

Definition 3.1. A **Tagged Network** is defined as $G =< V, E, T >$, where $V = \{v_i\}, i = 1, 2, ..., n$ represents the set of nodes, $E = \{e_{ij}\}, i, j = 1, 2, ..., n, i \neq j$ represents the set of edges, and $T = \{T_i\}, i = 1, 2, ..., n$ represents the set of tags that each node belongs to. Each tag corresponds to a literal symbol and labels a characteristic of a a node. Each node v_i have k tags $T_i = \{t_{i_1}, t_{i_2}, ..., t_{i_k}\}, t_{i_j} \in T$. If node v_i is labeled by tag t, node v_i has a tag t.

In a tagged Network, an edge between two nodes reflects their interaction relationship, while an edge between a node and a tag reflects their affiliation relationship. For example, in social networks, a node is a user, and two users have friend relationship if there exists an edge between them. A tag of the node may be a characteristic label, an interested topic or a community of the user. A node corresponds to multiple tags and a tag corresponds a set of nodes.

The text of a tag can only reflect part of the semantic information. Its semantic information is decided by the characteristic of its nodes in spirit. Therefore, we introduce ground-truth community to define the semantic distance between tags.

Definition 3.2. A **Ground-truth Community** C_t is the set of nodes labeled by tag t.

A tag corresponds to a ground-truth community. The distance between tags is defined by the relationship between their ground-truth communities. In the following, we introduce two types of relationships between Ground-truth Communities.

Definition 3.3. **Member Similarity** is the ratio of the common members to all members in two ground-truth communities, which is extended from the definition of common neighbor similarity [11]:

$$D_{MS}(C_{t_1}, C_{t_2}) = \frac{|C_{t_1} \cap C_{t_2}|^2}{|C_{t_1}| \cdot |C_{t_2}|}.$$

For two tags in a tagged network, they are closer semantically if their ground-truth communities have more common members.

Definition 3.4. **Social Closeness** is the ratio of the connected edges between members of two ground-truth communities to all their inter-connected edges:

$$D_{SC}(C_{t_1}, C_{t_2}) = \frac{|Edge(C_{t_1}, C_{t_2})|^2}{|Edge(C_{t_1}, V - C_{t_1})| \cdot |Edge(C_{t_2}, V - C_{t_2})|},$$

where $Edge(C_{t_1}, C_{t_2})$ is the set of the connected edges between members of C_{t_1} and C_{t_2}, and $Edge(C_{t_1}, V - C_{t_1})$ is the set of all inter-connected edges of C_{t_1}.

For two tags in a tagged network, they may be close semantically if the members of their ground-truth communities have close relationship.

Definition 3.5. Given a tagged network $G =< V, E, T >$, the problem of **Tag Embedding** aims to represent each tag $t \in T$ in a low-dimensional space $\mathbb{R}^d$. In other words, it aims to learn a mapping function $f : T \to \mathbb{R}^d$, where $d << |V|$, and the representation vectors in $\mathbb{R}^d$ preserve both member similarity and social closeness between ground-truth communities.

The computational complexity of member similarity and social closeness is $\Theta(|T|^2)$, where $|T|$ is the number of tags. As the complexity of social closeness is related to node numbers of communities, average degrees of communities, and neighborhood orders, it will be much higher in realit. Therefore, to avoid the expensive computation, we mix the nodes and tags into the same network to learn their semantic information, including the member similarity and social closeness between the tags. And we call the mixed network as node-tag hybrid network:

Definition 3.6. A **Node-Tag Hybrid Network** $H =< V', E' >$ is derived from a tagged network $G =< V, E, T >$, where $V' = V + T, E' = E + E_T$. For $v' \in V'$, v' is a plain node if $v' \in V$. Otherwise, it is a tag node if $v' \in T$. For $e' \in E'$, e' is an interaction edge if $e' =< v_1, v_2 > \in E$. Otherwise, it is an affiliation edge if $e' =< v, t >$ and node v is labeled by tag t.

4 TAG REPRESENTATION LEARNING MODEL

If we regard a node-tag hybrid network as a normal network and use traditional network embedding methods, we can get the representations for both plain nodes and tag node. However, the difference between plain nodes and tag nodes in the Node-Tag Hybrid Network are ignored. And we can't learn the semantic information of tags explicitly, neither.

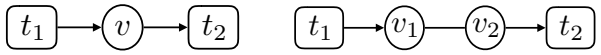
In this section, we propose Tag2Vec, a tag representation learning model. In a node-tag hybrid network, we design a new strategy, parameterized Random walk, to generate the context with both member similarity and social closeness information between tags. Meanwhile, We train a hyperbolic Skip-gram model with the generated context to learn the hyperbolic representation vectors of plain nodes and tag nodes. It can reduce the output dimensions and describe the transverse and vertical relationship between tags more accurately.

4.1 Parameterized Random Walk

We design a Tag2Vec parameterized random walk strategy in a node-tag hybrid network. Firstly, it can reflect both member similarity and social closeness between tags, and the ratio between their weights can be controlled by the parameter. Secondly, it should contain hierarchical information of both transverse and vertical relationship between tags for hyperbolic Skip-gram model, and the ratio between their weights can be controlled by the parameter, too. Specifically, we adopt two strategies to control parameters for different semantic information and different dimensional information.

4.1.1 Walk Patterns for Semantic Information. If two tags have similar semantic information, their representation vectors should be closer in the vector space. So, tag nodes with smaller semantic distance should appear more simultaneously in the context. As the semantic distance is decided by member similarity and social closeness, we consider two basic walk patterns for them respectively.

- **MS-driven walk.** As shown in Figure 1(a), in MS-driven walk, tag nodes with common members may appear simultaneously in the random walk sequences. There must be one and only one plain node between a pair of tag nodes, and the plain node is the common member of two tags.
- **SC-driven walk.** As shown in Figure 1(b), in SC-driven walk, tag nodes with connected members may appear simultaneously in the random walk sequences. There must be more than two plain nodes between a pair of tag nodes, and the adjacent plain nodes are connected.



(a) MS-driven walk. (b) SC-driven walk.

Figure 1: Walk Patterns.

We notice that in a node-tag hybrid network, there are no edges between tag nodes. Combining the two basic walk patterns, we find that when the current node is a tag node, the next node must be a plain node. When the current node is a plain node, the next node may be a tag node or a plain node.

When the current node is a tag node, we should select a member node from the ground-truth community of the tag. We consider that members with high social influence are more important to the community. So we calculate the weights of tag-to-plain edges according to the centrality, such as degree centrality, of plain nodes in the original network.

When the current node is a plain node, we should select the type of nodes as the next step, a tag node or a plain node. A tag node corresponds to MS-driven walk, and a tag node corresponds to SC-driven walk. So we set a parameter p as the probability to select a tag node and adopt the strategy of MS-driven walk.

4.1.2 Transverse and Vertical Relationship. As we discussed above, we introduce the hyperbolic representation vectors to distinguish the transverse interaction and vertical affiliation relationships, and discover the hierarchical semantic information of tags. This requires that the context generated by random walk must contain enough transverse and vertical relationships between tags. However, in a node-tag hybrid network, there are no edges between tags, and the correlation information between tags is obtained implicitly by the random walk sequence. We can know which type of relationship two tags have only when we finish the learning procedure. So we want to control the relationship between tags in the same sequence and the ratio between two types of relationships.

We find that the sizes of ground-truth communities are effective to infer the relationship between tags. If two tags have transverse interaction relationship, the difference between their community size may be small. While two tags with vertical affiliation relationship, the difference between their community size may be large. Thus, we can control the selection of the next tag in the random walk sequence according to their community sizes.

Specifically, in the random walk, if the current node is a plain node, we have to select a tag node from the tag neighbors as the next tag t. We consider the last tag t_{last} in the current random walk sequence and compare the community size of t_{last} with t. We consider two strategies.

- **Transverse Interaction Relationship.** If the next step is to sample for transverse social relationship, we tend to select the tags having similar community size with the last tag.

$$W_{simi}(v,t) = B \cdot \exp(-(|t| - |t_{last}|)^2)$$

- **Vertical Affiliation Relationship.** If the next step is to sample for vertical affiliation relationship, we tend to select the tags having different community size with the last tag.

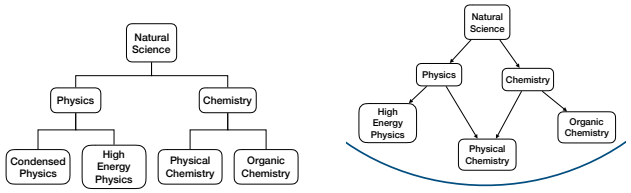
$$W_{diff}(v,t) = B \cdot \exp((|t| - |t_{last}|)^2)$$

Notice that B is the normalization constant. We set a parameter q as the probability to adopt the transverse interaction relationship.

4.2 Hyperbolic Skip-Gram

In tag representation learning, the relationships between tags contain both transverse interaction relationships and vertical affiliation relationships. But in fact, most tag relationship is between transverse and vertical. For example, in Figure 2(a), it is a tree describing the relationships between research fields. A node and one its children have affiliation relationship, while the children of a node may have interaction relationship. But as for two subjects, Physics and Physical Chemistry, we can't describe their relationship in this tree.

Hyperbolic geometry is a non-Euclidean geometry that studies the spaces of constant negative curvature. It is capable to model the hierarchical structure with low dimensions. Meanwhile, in continuous hyperbolic spaces, there are no distinct hierarchies, which means there are no distinct affiliation relationships. In general, nodes with higher levels are placed closer to the origin of the sphere and the children of a node are placed in the area with similar angle and longer distance to the origin. We show relationships discussed above in a two-dimensional hyperbolic space in Figure 2(b). It describes the relationships between Physical Chemistry, Physics and Chemistry suitably. Besides, Euclidean spaces expand polynomially



(a) Tree Structure (b) Hyperbolic Structure

Figure 2: The Structure of A Tag Network

while hyperbolic spaces expand exponentially with regard to the radius r. So we can learn the representations in hyperbolic spaces with less parameters and lower dimensions. Therefore, embedding in hyperbolic spaces can not only describe the complex relationship between tags, but also reduce the output dimensions.

Existing network embedding method[25] uses a pure strategy that models the edge probability and it is not a Skip-gram procedure. That is to say that we can't learn the hyperbolic representation vectors from the context generated by Tag2Vec paratemeterized random walk. Thus, we propose a method to learn the representations on the Poincare ball with Skip-gram, as Poincare disk is a common and simple way to model htperbolic geometry.

Euclidean and hyperbolic spaces are both Riemannian spaces. Their difference is that Euclidean spaces have isotropic Riemannian metric tensor $g^{(E)}$. While in hyperbolic spaces, the metric tensor is:

$$g_x^{(H)} = (\frac{2}{1 - \|\vec{x}\|^2})^2 g^{(E)}. \tag{1}$$

Therefore, in hyperbolic spaces, $g_x^{(H)}$ is related to the vector $\vec{x}$ and it is not shift-invariant. So there isn't a simple formation of global inner-product. But there is still the definition of the distance between two points in hyperbolic spaces:

$$d_H(\vec{u},\vec{v}) = \cosh^{-1}(1 + \frac{\|\vec{u} - \vec{v}\|^2}{(1 - \|\vec{u}\|^2)(1 - \|\vec{v}\|^2)}). \tag{2}$$

In traditional Skip-gram model, we use the scoring function $s(w,h)$ between the word w and the context h to evaluate their closeness. Then we have the probability distribution of the context h given the word w in the neural probabilistic language model [2] $P(h|w) = \frac{\exp(s(w,h))}{\sum_{h'} \exp(s(w,h'))}$.

Traditionally, $s(w,h)$ is a linear function, where we calculate the inner-product of $\vec{w}$ and $\vec{h}$. It is an operation in Euclidean spaces and we want to transfer it into hyperbolic spaces. But as mentioned above, there is no definition of inner-product in hyperbolic spaces. So we transform $s(w,h)$ into a function with regard to the distance:

$$s(w,h) = -A\|\vec{w} - \vec{h}\|^2 + B = -A(d^2(\vec{w},\vec{h})) + B, \tag{3}$$

where A and B are parameters. But as the probability is normalized by a constant $\sum_{h'} \exp(s(w,h'))$, they have no influence on the model. So the probability distribution $P(h|w)$ can be calculated according to the Euclidean distance:

$$P(h|w) = \frac{\exp(\Phi(d(w,h)))}{\sum_{h'} \exp(\Phi(d(w,h')))}, \tag{4}$$

where Φ can be a non-linear operator.

Therefore, we replace Euclidean distance d_E with hyperbolic distance d_H. For simplicity, we let Φ as an identify function and define the loss function using maximum likelihood estimation. We use negative sampling to accelerate and the final loss function is

$$L(\vec{w}) = -\sum_{(i,j)\in D} \log \sigma(-d_H(\vec{w}_i, \vec{h}_j)) + k \cdot \mathbb{E}_{h_n \sim P_n(h)} \left[\log \sigma(d_H(\vec{w}_i, \vec{h}_n))\right], \tag{5}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ and $P_n(h)$ is the noise distribution for negative sampling.

4.2.1 Optimization. As we use distance function $d_H(w, h)$ in hyperbolic spaces, the loss function can be optimized with Riemannian stochastic gradient descent (RSGD) optimization method [5].

5 EXPERIMENT

In order to demonstrate the superiority of our tag embeddings, we conduct the experiment on two datasets.

5.1 Dataset

We have to choose datasets with tags of nodes, such as communities, categories, and other discrete features. Meanwhile, these tags may have implicit hierarchical correlation.

- **NBER Patent Citation Data** [17] describes patents in U.S. over the last 30 years. Each patent is a node and each citation is an edge in our network. Each node has one class, one category and sub-category as its tags. There are 6 categories, 35 sub-categories, and 426 classes classified by USPTO. We extract a sub-network from the dataset with 401 nodes, 462 tags and more than 3000 edges. To be notified, the affiliation relationship between categories and sub-categories is not part of the input and we will learn it from our model.
- **WordNet Data** [22] is a lexical database. It groups English words into sets of synonyms called synsets and records relations among these synonym sets or their members. We extract a subset from the dataset and construct a network. Each node in the network is a leaf in the tree and its tags are k-order ancestors and each edge in the network represents the genetic relationship between two words.

5.2 Baselines

We use the following baselines comparing with our model. Except the first baseline, we learn the representation vectors of nodes and tags simultaneously in a node-tag hybrid network.

- Origin: We construct a plain network without tags and learn the representation vectors of nodes by DeepWalk. Then we encode the tags into one-hot vectors, and concatenate the node vectors and node vectors together.
- DeepWalk [27]: It generates context of nodes by random walks and adopts Skip-gram model.
- LINE [30]: It preserves both first-order and second-order proximities between nodes.
- Node2Vec [16]: It is an improvement of DeepWalk, and it adds more flexibility in exploring neighbors.

5.3 Tag Representation Visualization

Figure 3 shows the visualization of tags learned from our model on NBER Patent Citation dataset. It is obvious that the tag representations learned from our model have good hierarchical property. 6 categories are distant from each other and dispersed in the space. Sub-categories of the same category are close to each other and cluster around the category. For example, Optics and Transportation are sub-categories of Mechanical, and they are close around Mechanical.

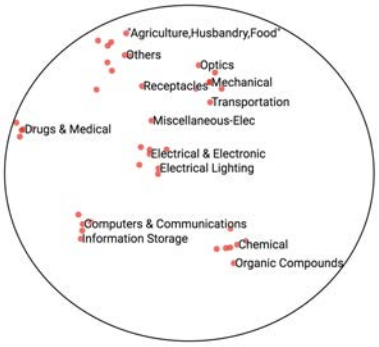


Figure 3: Tag Visualization

5.4 Results

5.4.1 Node Classification. On NBER patent citation dataset, we concatenate the node and tag vectors together as the expended representations to classify the nodes. The classes and sub-categories are tags of nodes. We use F1 score as the evaluation metric.

Figure 4 shows the results of node classification. It can be seen that origin has the worst result. This is because it encodes tags into one-hot vectors so that all tags have the same status and their distances are ignored. While the rest methods mix the nodes and tags into a hybrid network to learn the relationships between tags. An exceptional case is that LINE not performs well because it ignores the global information. Furthermore, our model outperforms DeepWalk and Node2Vec, as parametrized random walk can better learn the semantic information, and hyperbolic Skip-gram helps to learn the hierarchical information. It demonstrates that the representations learned from our model have stronger ability for classification.

5.4.2 Similar Community Detection. We detect the similar communities on the NBER patent citation dataset. A community corresponds to a tag, including classes, categories and sub-categories. We calculate distances between tag vectors. As for a community, the community with closest distance is its similar community. The similar community refers to the one with most common members or most connections. We use AUC as the evaluation metric.

Figure 5 shows the results of similar community detection. For all baselines, we mix the nodes and tags into a hybrid network. The representation vectors learned from our model can predict the similarity relationships between communities and analyze the community features of complex networks.

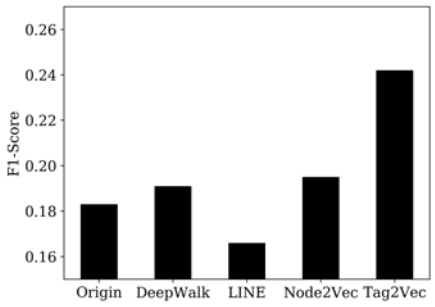


Figure 4: Node Classification

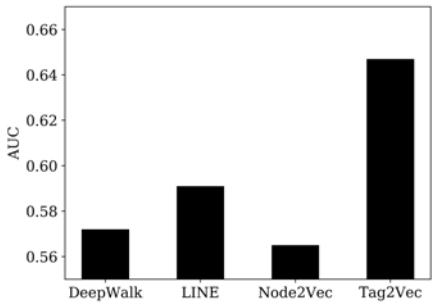


Figure 5: Similar Community Detection

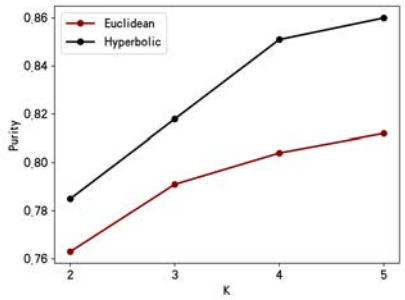


Figure 6: Reconstruction of K

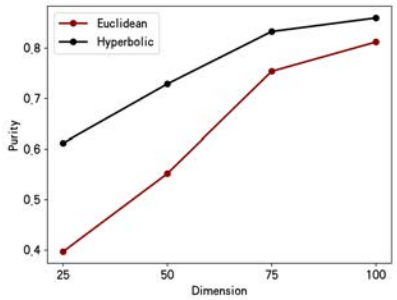


Figure 7: Reconstruction of Dimension

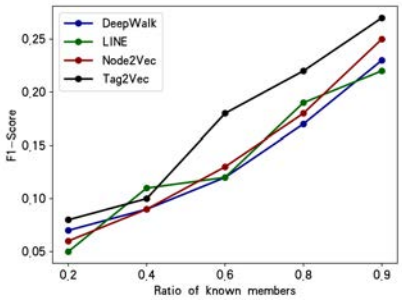


Figure 8: Stability

5.4.3 Hierarchical Structure Reconstruction. Comparing with Euclidean spaces, Skip-gram model in hyperbolic spaces has two advantages: reducing the dimensions of vectors and reflecting affiliation relationships better. To evaluate this, we conduct the experiment of hierarchical structure reconstruction on WordNet dataset to compare the performance of representation vectors from Euclidean and hyperbolic spaces. We reconstruct the word tree by clustering the tag representation vectors and evaluate the results with purity $\sum_i^K \max_j\{p_{ij}\}$, where K is cluster number, and p_{ij} is the probability that members in cluster i belong to class j. Purity measures the correctness of hierarchical structure reconstruction. If we can cluster correctly, it means that we can discover the complex hierarchical structure of ground-truth communities of tags with the help of the behaviors of their members in the network.

Figure 6 shows the purity of clustering with different order of ancestors k in two spaces. k measures the richness of tags and the complexity of hierarchical information. The larger the k is, the more tags a node has. The result shows that embedding in hyperbolic spaces is superior with the best reconstruction performance. Meanwhile, when there are more tags and more complex structure, the advantage of hyperbolic spaces is greater.

Figure 7 shows the purity of clustering with different dimensions. When the dimensions are lower, embedding in hyperbolic spaces performs significantly better than embedding in Euclidean spaces. It indicates that in hyperbolic spaces, we can learn good representations with low dimensions.

5.4.4 Tag Stability. In real networks, tags of nodes often make changes. Besides, part of interaction records may be missing. However, we consider that changes or lacking of few members may not influence the semantic information of tags greatly. To demonstrate the stability of tags learned from our model, we extract part of plain nodes as known members and all tag nodes to construct a network and learn the tag representations. Then we extract another part of plain nodes to construct another network without tags, learn the representation vectors of plain nodes and combine them with tag vectors learned before to predict the node labels.

The results are shown in Figure 8. When the known members reach a certain proportion(60%), F1-score of TagVec is higher than other baselines obviously. This is because in the parameterized random walk, we emphasize the importance of tags when generating context. When some members are missing, we can still learn good tag representation vectors thanks to their stronger stability.

6 CONCLUSION

In this paper, we analyze semantic and hierarchical information between tags and propose a novel tag representation learning model, Tag2Vec. The experimental results show that the tag representations learned from our model have strong representative ability.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grant No. 61876006 and No. 61572041).

REFERENCES

[1] Aaron B Adcock, Blair D Sullivan, and Michael W Mahoney. 2013. Tree-like structure in large social and information networks. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE, 1–10.

[2] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3, Feb (2003), 1137–1155.

[3] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. 2011. Node classification in social networks. In *Social network data analytics*. Springer, 115–148.

[4] Marián Boguná, Fragkiskos Papadopoulos, and Dmitri Krioukov. 2010. Sustaining the internet with hyperbolic mapping. *Nature communications* 1 (2010), 62.

[5] Silvere Bonnabel et al. 2013. Stochastic Gradient Descent on Riemannian Manifolds. *IEEE Trans. Automat. Contr.* 58, 9 (2013), 2217–2229.

[6] Guillaume Bouchard, Sameer Singh, and Theo Trouillon. 2015. On approximate reasoning capabilities of low-rank vector spaces. *AAAI Spring Syposium on Knowledge Representation and Reasoning (KRR): Integrating Symbolic and Neural Approaches* (2015).

[7] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 891–900.

[8] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. 2015. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 119–128.

[9] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2018. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering* (2018).

[10] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 135–144.

[11] Lun Du, Zhicong Lu, Yun Wang, Guojie Song, Yiming Wang, and Wei Chen. 2018. Galaxy Network Embedding: A Hierarchical Community Structure Preserving Approach.. In *IJCAI*. 2079–2085.

[12] Lun Du, Yun Wang, Guojie Song, Zhicong Lu, and Junshan Wang. 2018. Dynamic Network Embedding: An Extended Approach for Skip-gram based Network Embedding.. In *IJCAI*. 2086–2092.

[13] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1797–1806.

[14] Octavian Ganea, Gary Becigneul, and Thomas Hofmann. 2018. Hyperbolic Entailment Cones for Learning Hierarchical Embeddings. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Jennifer Dy and Andreas Krause (Eds.), Vol. 80. PMLR, StockholmsmÃďssan, Stockholm Sweden, 1646–1655. http://proceedings.mlr.press/v80/ganea18a.html

[15] Mikhael Gromov. 1987. Hyperbolic groups. In *Essays in group theory*. Springer, 75–263.

[16] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[17] Bronwyn H Hall, Adam B Jaffe, and Manuel Trajtenberg. 2001. *The NBER patent citation data file: Lessons, insights and methodological tools*. Technical Report. National Bureau of Economic Research.

[18] Xiao Huang, Jundong Li, and Xia Hu. 2017. Label informed attributed network embedding. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 731–739.

[19] Zhipeng Huang and Nikos Mamoulis. 2017. Heterogeneous information network embedding for meta path based proximity. *arXiv preprint arXiv:1701.05291* (2017).

[20] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2005. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 177–187.

[21] Linyuan Lü and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications* 390, 6 (2011), 1150–1170.

[22] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[23] Nina Mishra, Robert Schreiber, Isabelle Stanton, and Robert E Tarjan. 2007. Clustering social networks. In *International Workshop on Algorithms and Models for the Web-Graph*. Springer, 56–67.

[24] Maximilian Nickel, Xueyan Jiang, and Volker Tresp. 2014. Reducing the rank in relational factorization models by including observable patterns. In *Advances in Neural Information Processing Systems*. 1179–1187.

[25] Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*. 6338–6347.

[26] Shirui Pan, Jia Wu, Xingquan Zhu, Chengqi Zhang, and Yang Wang. 2016. Tri-party deep network representation. *Network* 11, 9 (2016), 12.

[27] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.

[28] Xiaofei Sun, Jiang Guo, Xiao Ding, and Ting Liu. 2016. A General Framework for Content-enhanced Network Representation Learning. *arXiv preprint arXiv:1610.02906* (2016).

[29] Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1165–1174.

[30] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1067–1077.

[31] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1225–1234.

[32] Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. 2018. SHINE: signed heterogeneous information network embedding for sentiment link prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 592–600.

[33] Linchuan Xu, Xiaokai Wei, Jiannong Cao, and Philip S Yu. 2017. Embedding of embedding (eoe): Joint embedding for coupled heterogeneous networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 741–749.

[34] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y Chang. 2015. Network representation learning with rich text information.. In *IJCAI*. 2111–2117.

The Silent Majority Speaks: Inferring Silent Users' Opinions in Online Social Networks

Lei Wang, Jianwei Niu, Xuefeng Liu, Kaili Mao*
State Key Laboratory of Virtual Reality Technology and System,
School of Computer Science and Engineering, Beihang University, Beijing 100191, China
{lei,niujianwei,liu_xuefeng,maokaili}@buaa.edu.cn

ABSTRACT

With the blossoming of social networking platforms like Twitter and Facebook, how to infer the opinions of online social network users on specific topics they had not directly given yet, has received much attention. Existing solutions mainly rely on one's previous posted messages. However, recent studies show that over 40% of users opt to be silent all or most of the time and post very few messages. Consequently, the performance of existing solutions will drop dramatically when they are applied to infer silent users' opinions, and how to infer the opinions of these silent users becomes a meaningful while challenging task. Inspired by the collaborative filtering techniques in cold-start recommendations, we infer the opinions of silent users by leveraging the text content posted by active users and their relationships between silent users. Specifically, we first consider both observed and pseudo relationships among users, and cluster users into communities in order to extract various kinds of features for opinion inference. We then design a coupled sparse matrix factorization (CSMF) model to capture the complex relations among these features. Extensive experiments on real-world data from Twitter show that our CSMF model achieves over 80% accuracy for the inference of silent users' opinions.

CCS CONCEPTS

• **Information systems** → **Social networks**; • **Computing methodologies** → *Factorization methods*.

KEYWORDS

Silent users; opinion inference; online social networks

ACM Reference Format:
Lei Wang, Jianwei Niu, Xuefeng Liu, Kaili Mao. 2019. The Silent Majority Speaks: Inferring Silent Users' Opinions in Online Social Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313423

1 INTRODUCTION

Recent years have witnessed a booming development of online social networking platforms like Twitter and Facebook. In online social networks, even active users will not publish messages to directly show their opinions towards all topics. As a result, inferring users' opinions on specific topics they had not directly given yet, has attracted increasing attention [11, 13, 26]. When inferring a user's opinion on a specific topic, existing solutions generally are based on his/her previous posted messages on other topics, in order to profile himself/herself [7] or find his/her neighbors with similar user personal interests [9, 17, 30]. For example, [7] extracted two types of features from a user's posted content about other topics to tell if the user has certain opinion bias on all topics (i.e., some users are more likely to be positive or negative). Then the user's opinion on a specific topic can be correlated with or similar as his/her opinions on some other topics.

However, recent work on social science [6, 10, 22] has shown that a significant proportion (over 40%) of users choose to be *silent* all or most of the time, and seldom publish messages to express their opinions towards various topics. Having a better understanding of these silent users' opinions is important, as it greatly helps us to avoid misjudgement of overall population-level opinions, and advances a variety of real-world applications, such as targeted advertising [5, 14] and political election predictions [8]. Inferring the opinions of silent users is a quite challenging task. As silent users do not generate sufficient content, existing solutions which rely on one's historical posted messages often only focus on users who have historical messages, but ignore these silent users who do not post anything in a long period time. The performance of existing solutions will drop dramatically when they are used to predict the opinions of silent users [7, 17]. Different from existing work on opinion inference, we take a step towards the inference of silent users' opinions in this paper.

We find that inferring silent users' opinions is intrinsically similar to cold-start recommendation problem, whose objective is to predict rating scores for specific items rated by new users without any historical preference information. In recommendation systems, users tend to have similar tastes with their friends. As a result, rating scores of new users can be predicted by considering the preference information of their friends in social networks [16, 19]. Similarly, existing studies on social science theories have verified that users that are connected (become friends) are prone to exhibit similar opinions on a certain topic [2, 3, 18]. Therefore, we argue that we can infer silent users' opinions by leveraging the text content generated by active users and their relationships between silent users.

However, applying the methods designed for cold-start recommendations to infer silent users' opinions still entails many challenges due to the following reasons. 1) *Diversity of user-user relations*: Existing studies [4, 24] on social science have found that

*J. Niu is also with the Beijing Advanced Innovation Center for Big Data and Brain Computing (BDBC) and Hangzhou Innovation Institute of Beihang University.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313423

observed relationships (i.e., the follower/followee network structure) are only one type of user-user relations, while other types of user-user relations are also very important for shared opinions among users. 2) *Heterogeneous fusion*: Many features need to be considered when characterizing silent users and inferring their opinions, including structural features of the user-relationship network, opinion information of text content, users' interests, etc. Simultaneously considering all these features extracted from heterogeneous sources is quite difficult.

To address the aforementioned issues, the inference of silent users' opinions is modeled as a matrix factorization problem in this work, and a coupled sparse matrix factorization (CSMF) model is proposed to fuse text content data and user-relationship data for accurate inference. The main contributions of this paper are summarized as follows:

- To extract features of silent users, we first infer user-user relations and cluster users into communities. Then three sets of features are extracted for each user.
- A coupled sparse matrix factorization (CSMF) model is proposed to capture the complex relations among various features and infer silent users' opinions.
- We validate the effectiveness of our CSMF model based on real-world dataset from Twitter. Experimental results demonstrate that our model can be effectively used to infer silent users' opinions.

2 DATA DESCRIPTION

In this paper, Twitter is employed as the basis for our opinion inference experiments. The Twitter data we used consists of over 70,000 messages on 20 hot political topics and the follower/followee network structure among over 6,000 users. For each Twitter message, it is assigned an opinion label: -1 (negative), 0 (neutral) or $+1$ (positive). We employ 20 people to manually assign an opinion label for each message. In this paper, we restrict our attention mainly to negative, neutral and positive user-topic opinions. Then the final opinion of each user on a specific topic is computed by averaging the opinion labels of all his/her posted messages on this topic.

One factor we need to consider is whether we should require both users in a potential pair to connect with each other. On Twitter, mutual connections presumably indicate closer relationships, while one-way connections may correspond to a desire to pay attention (e.g., to a famous singer), rather than necessarily personal relationships. Existing studies [23] have shown that attention effects may be more important than homophily effects with respect to shared opinions in social networks. Therefore, we consider two possibilities when we define that a connection (edge) between a pair of users exists.

- Directed follow network: user u_i follows user u_j (u_j may or may not follow u_i in return).
- Mutual follow network: user u_i follows user u_j, and u_j follows u_i at the same time.

3 FEATURE EXTRACTION FOR USERS

In this section, we extract features for both silent and active users based on text content data and follower/followee network structure.

3.1 Inferring User-User Relations

Existing work [4] has found that there are two types of user-user relations in online social networks: observed relationships (or explicit relationships) and pseudo-relationships. Observed relationships can be reflected by the follower/followee network structure, and provide information on personal relationships among users. Pseudo-relationships are extracted from sentimental text content, where a connection between a pair of users is created if they hold consistent opinions on different topics. Since the inference of silent users' opinions is largely dependent on user-user relations, it is not a trivial matter to infer pseudo-relationships among users and consider both observed and pseudo relationships when inferring silent users' opinions.

Intuitively, a pseudo-friend relationship between a pair of users is likely to exist when they share the same opinions on many topics. In contrast, a pseudo-foe relationship between two users may exist if they have conflicting opinions. To infer the pseudo-relationship between two users, we first introduce the *meta path*, which is used to capture a sequence of relations defined between users [21]. For example, two users u_i and u_j are connected via the path "user-message-topic-message-user" (i.e., U-M-T-M-U) if both u_i and u_j have posted at least one message about a same topic T. Given a meta path (e.g., $\mathcal{P}$=U-M-T-M-U), the similarity between u_i and u_j is computed as follows:

$$sim(u_i, u_j) = \frac{2 \sum_{m_k \in M_{u_i}, m_l \in M_{u_j}} (1_{(p_{u_i \rightsquigarrow u_j} \in \mathcal{P})} \cdot Sc(m_k, m_l))}{|\{p_{u_i \rightsquigarrow u_i} \in \mathcal{P}\}| + |\{p_{u_j \rightsquigarrow u_j} \in \mathcal{P}\}|} \tag{1}$$

where M_{u_i} denotes the set of messages published by user u_i, and $p_{u_i \rightsquigarrow u_j}$ represents a path instance between u_i and u_j that follows the defined meta path $\mathcal{P}$. $Sc(m_k, m_l)$ is calculated as:

$$Sc(m_k, m_l) = \begin{cases} +1 & \text{if } sl(m_k) = sl(m_l) \\ -1 & \text{otherwise} \end{cases} \tag{2}$$

where $sl(m_k)$ is the opinion label of message m_k.

From the definition of $sim(u_i, u_j)$, we can find that it is a real value ranging between -1 and 1. A score $sim(u_i, u_j)$ close to 1 indicates that users u_i and u_j hold similar opinions with respect to different topics, and we denote them as *pseudo-friends*. On the contrary, two users with a score $sim(u_i, u_j)$ close to -1 indicates that they are *pseudo-foes* who hold conflicting opinions on many topics. Moreover, a score close to 0 indicates that these two users share both consistent and conflicting opinions. In this paper, we define a pseudo-connection between users u_i and u_j exists if their $sim(u_i, u_j)$ is larger than a threshold γ.

3.2 Clustering Users into Communities

We need to cluster users into communities, and extract a user's features based on the text content posted by the users in the same community and community structure information. Therefore, in this subsection, we shift our focus to how to cluster users into communities.

A lot of community detection algorithms have been proposed in order to group the users of a network into groups of users with denser connections internally and sparser connections between groups [15, 28, 29]. With user-user relations (including observed

relationships and pseudo-relationships) in place, we employ the method proposed in [25], which can achieve the best performance of community detection on large social networks currently as far as we know, to cluster both silent and active users. This method unifies several commonly used clustering quality functions including modularity, cluster deletion, and sparsest cut, and places all these clustering quality functions within a unified optimization framework.

3.3 Features for Opinion Inference

In this subsection, we extract three sets of features for both silent and active users.

Content Features: Content features are used to describe the interested topics of users and the distribution of their opinions. As silent users publish very few messages, we cannot directly extract content features for them from their posted text content. To solve this problem, for each user, we consider the messages posted by both himself/herself and other users in the same community when extracting content features for him/her. For user u, his/her content feature vector is represented as:

$$f_u^C = (< tp_1, s_1 >, < tp_2, s_2 >, ..., < tp_N, s_N >) \tag{3}$$

where N is the total number of topics we consider, $tp_i (1 \leq i \leq N)$ equals to 1 if user u has posted messages about the i-th topic, and 0 otherwise. As mentioned before, we define the opinion of a user on a particular topic as the average opinion labels of all his/her published messages on this topic if he/she has posted messages on this topic. Here, we set s_i as user u's opinion on the i-th topic if u has posted messages on this topic, the average opinion of other users in the same community on the i-th topic if u has not but other users in the same community have posted messages on this topic, and the average opinion of all users in our dataset on this topic otherwise.

Intra-Community Structural Features: To extract intra-community structural features for each user, we consider the user-user relations among the whole users in the same community. For user u who resides in community C, we compute his/her degree, the average distance between u and all other users in C, the number of shortest paths between any two users in C that pass through u, and the Katz centrality [1], which is calculated as:

$$C_{Katz}(u) = \sum_{k=1}^{\infty} \sum_{v=1}^{|C|} \alpha^k (\mathbf{A}^k)_{u,v} \tag{4}$$

where α is an attenuation factor between 0 and 1, and matrix $\mathbf{A}$ is the adjacency matrix, i.e., $a_{u,v} = 1$ if user u is connected with user v, and $a_{u,v} = 0$ otherwise.

Inter-Community Structural Features: For each user, we employ inter-community structural features to indicate which community he/she is located in and the connections between the user and users in other communities. Given a set of users, if these users are clustered into K communities by the aforementioned community detection algorithm, the inter-community structural feature vector for user u can be represented as:

$$f_u^{Inter} = (s_1, s_2, ..., s_K) \tag{5}$$

where $s_i (1 \leq i \leq K) \in \{-1, 0, 1\}$. s_i equals to 0 if user u is located in the i-th community, 1 if u is connected with any user in the i-th community, and -1 otherwise.

4 INFERRING SILENT USERS' OPINIONS

In this section, we introduce how to utilize various kinds of features, and design a matrix factorization based framework to infer the opinions of silent users.

4.1 Data Modelling

We can extract various kinds of information from text content and user-user relation data. Here we build several matrices to capture features extracted from different sources and the relations among them.

Features of a user: For each user, we extract content features f^C, intra-community structural features f^{Intra} and inter-community structural features f^{Inter}. Then we place f^C, f^{Intra} and f^{Inter} of each user into a user feature matrix $\mathbf{Z}$, where each row denotes a user and each column stands for a kind of feature.

Opinion distribution patterns: To reflect opinion distribution patterns for individual users and communities, we extract user-topic opinion matrix $\mathbf{O}$ and community-topic opinion matrix $\mathbf{C}$. Specifically, each row of matrix $\mathbf{O}$ denotes a topic, and each column represents a user. Each element $o_{i,j}$ in $\mathbf{O}$ stands for the opinion of the j-th user on the i-th topic. $o_{i,j}$ equals to *null* if the j-th user had not posted messages on the i-th topic. Matrix $\mathbf{O}$ is very sparse because silent users seldom post messages and active users do not publish messages for all topics. To capture the community-topic opinion information, matrix $\mathbf{C}$ is built. Matrix $\mathbf{C}$ represents opinion distribution patterns for communities, where each row denotes a topic and each column stands for a community. Each community contains several users, and each element of $\mathbf{C}$ is the average opinion labels of users in a particular community on a particular topic. Compared with $\mathbf{O}$, $\mathbf{C}$ is a higher level but denser representation for opinion distribution patterns.

4.2 Coupled Sparse Matrix Factorization (CSMF) Model

In this subsection, we infer the opinions on different topics for each silent user, based on various matrices mentioned before. In the previous subsection, we build three matrices $\mathbf{O}$, $\mathbf{C}$ and $\mathbf{Z}$. Given the above settings, the goal of inferring the opinion of a specific user u_j on a particular topic t_i can be converted into filling the missing value in i-th row and j-th column of $\mathbf{O}$ with the help of $\mathbf{C}$ and $\mathbf{Z}$.

Basically, this goal can be achieved by solely factorizing matrix $\mathbf{O}$. Specifically, a more accurate but compact representation for users and topics in a low-rank space can be found through matrix factorization, and matrix $\mathbf{O}$ can be approximated by a multiplication of two low-rank matrices:

$$\mathbf{O} \approx \mathbf{T} \times \mathbf{U}^T \tag{6}$$

where $\mathbf{T} \in \mathbb{R}^{N \times D}$ and $\mathbf{U} \in \mathbb{R}^{M \times D}$ are low-rank matrices with $D \ll min(M, N)$, and M and N denote the total number of users and topics, respectively. The row vector $T(i,:)$ in $\mathbf{T}$ $(1 \leq i \leq N)$ and $U(j,:)$ in $\mathbf{U}$ $(1 \leq j \leq M)$ are the latent representations for the

i-th topic and j-th user, respectively. In traditional matrix factorization methods, matrix $\mathbf{O}$ can be approximated by minimizing the following objective function:

$$\min_{\mathbf{T},\mathbf{U}} \mathcal{L} = \frac{1}{2}||\mathbf{I} \circ (\mathbf{O} - \mathbf{T}\mathbf{U}^T)||_F^2 \tag{7}$$

where $||\cdot||_F$ represents the Frobenius norm of a matrix. $\mathbf{I}$ is an indicator matrix with its element $I_{i,j}$ equals to 0 if $o_{i,j}$ is *null*, 1 otherwise. The operator "$\circ$" denotes the entry-wise product (i.e., Hadamard product).

Note that in our case, the user-topic opinion matrix $\mathbf{O}$ is quite sparse since the observed user-topic opinions are only a small percent. As a result, the approximation of $\mathbf{O}$ will be not accurate enough if we solely factorize itself. To solve this problem, we can incorporate another two matrices $\mathbf{C}$ and $\mathbf{Z}$ to decompose these matrices collaboratively. To be more specific, we first factorize community-topic opinion matrix $\mathbf{C} \in \mathbb{R}^{N \times K}$ into two low-rank latent matrices $\mathbf{T} \in \mathbb{R}^{N \times D}$ and $\mathbf{G} \in \mathbb{R}^{K \times D}$, where K is the total number of communities. Since matrices $\mathbf{O}$ and $\mathbf{C}$ share the latent topic matrix $\mathbf{T}$, we can make $\mathbf{T}$ more accurate by jointly decomposing $\mathbf{O}$ and $\mathbf{C}$. Likewise, user feature matrix $\mathbf{Z}$ is also employed to help determine the latent user matrix $\mathbf{U}$ more accurately by factorizing $\mathbf{Z} \in \mathbb{R}^{M \times L}$ into $\mathbf{U} \in \mathbb{R}^{M \times D}$ and $\mathbf{F} \in \mathbb{R}^{L \times D}$, where L is the dimension of the user feature vector, and $\mathbf{O}$ and $\mathbf{Z}$ share the latent user matrix $\mathbf{U}$. Since matrices $\mathbf{C}$ and $\mathbf{Z}$ are much denser than $\mathbf{O}$, community-topic opinion matrix $\mathbf{C}$ and user feature matrix $\mathbf{Z}$ can make contributions to facilitating a more accurate factorization of $\mathbf{O}$ through shared matrices $\mathbf{T}$ and $\mathbf{U}$. That is to say, we can collaboratively decompose $\mathbf{O}$, $\mathbf{C}$ and $\mathbf{Z}$ as follows:

$$\mathbf{O} \approx \mathbf{T} \times \mathbf{U}^T; \mathbf{C} \approx \mathbf{T} \times \mathbf{G}^T; \mathbf{Z} \approx \mathbf{U} \times \mathbf{F}^T \tag{8}$$

where $\mathbf{O}$ and $\mathbf{C}$ share latent topic factor $\mathbf{T}$, and $\mathbf{O}$ and $\mathbf{Z}$ share latent user factor $\mathbf{U}$. Then we can achieve this goal by forming and minimizing the following objective function:

$$\begin{aligned}\min_{\mathbf{T},\mathbf{U},\mathbf{F},\mathbf{G}} \mathcal{L} = &\frac{1}{2}||\mathbf{I} \circ (\mathbf{O} - \mathbf{T}\mathbf{U}^T)||_F^2 + \frac{\lambda_1}{2}||\mathbf{C} - \mathbf{T}\mathbf{G}^T||_F^2 \\ &+ \frac{\lambda_2}{2}||\mathbf{Z} - \mathbf{U}\mathbf{F}^T||_F^2\end{aligned} \tag{9}$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are used to control the loss in the matrix factorization. After the factorization, we can recover matrix $\mathbf{O}$ through the production of $\mathbf{T}$ and $\mathbf{U}^T$.

To further alleviate the data sparsity problem, we also take the correlations between users into account. Let matrix $\mathbf{S}$ be the correlation matrix of users, with each element $s_{i,j}$ denoting the correlation (similarity) between users u_i and u_j. $\mathbf{D}$ is a diagonal matrix with the i-th diagonal element as $d_{i,i} = \sum_{j=1}^{M} s_{i,j}$, and $\mathbf{L_S} = (\mathbf{D} - \mathbf{S}) \in \mathbb{R}^{M \times M}$ is the Laplacian matrix of the user correlation graph. Here, we use Pearson correlation analysis method [20] to obtain user correlation matrix $\mathbf{S}$ and the corresponding Laplacian matrix $\mathbf{L_S}$. Then the correlation information between users is considered as the trace of matrix $\mathbf{U}^T\mathbf{L_S}\mathbf{U}$, and can be obtained through the following deduction, which can guarantee that for two users u_i and u_j with higher correlation (i.e., the value of $s_{i,j}$ is large), they will be also closer in distance between the row vectors $U(i,:)$ and $U(j,:)$ in the matrix $\mathbf{U}$.

$$\begin{aligned}&\frac{1}{2}\sum_{i,j} s_{i,j}||U(i,:) - U(j,:)||_F^2 = \frac{1}{2}\sum_{i,j}\sum_{k=1}^{D} s_{i,j}(U(i,k) - U(j,k))^2 \\ &= \sum_{i,j}\sum_{k=1}^{D} s_{i,j}U^2(i,k) - \sum_{i,j}\sum_{k=1}^{D} s_{i,j}U(i,k)U(j,k) \\ &= \sum_{k=1}^{D} U^T(:,k)(\mathbf{D} - \mathbf{S})U(:,k) = tr(\mathbf{U}^T(\mathbf{D} - \mathbf{S})\mathbf{U}) \\ &= tr(\mathbf{U}^T\mathbf{L_S}\mathbf{U})\end{aligned} \tag{10}$$

where $tr(\cdot)$ represents the matrix trace. In this way, user correlation matrix $\mathbf{S}$ contributes to a more accurate $\mathbf{U}$.

After all components mentioned are combined, the final objective function of the coupled sparse matrix factorization model is formally defined as:

$$\begin{aligned}\min_{\mathbf{T},\mathbf{U},\mathbf{F},\mathbf{G}} \mathcal{L} = &\frac{1}{2}\underbrace{||\mathbf{I} \circ (\mathbf{O} - \mathbf{T}\mathbf{U}^T)||_F^2}_{\text{user-topic factorization}} + \frac{\lambda_1}{2}\underbrace{||\mathbf{C} - \mathbf{T}\mathbf{G}^T||_F^2}_{\text{community-topic factorization}} \\ &+ \frac{\lambda_2}{2}\underbrace{||\mathbf{Z} - \mathbf{U}\mathbf{F}^T||_F^2}_{\text{user feature factorization}} + \frac{\lambda_3}{2}\underbrace{tr(\mathbf{U}^T\mathbf{L_S}\mathbf{U})}_{\text{user correlation}} \\ &+ \frac{\lambda_4}{2}\underbrace{(||\mathbf{T}||_{2,1} + ||\mathbf{U}||_{2,1} + ||\mathbf{G}||_{2,1} + ||\mathbf{F}||_{2,1})}_{\text{row-wise sparsity regularization}}\end{aligned} \tag{11}$$

Finally, we can utilize the factorized matrices $\mathbf{T}$ and $\mathbf{U}$ to reconstruct $\mathbf{O}$ to $\widetilde{\mathbf{O}}$, namely $\widetilde{\mathbf{O}} = \mathbf{T} \times \mathbf{U}^T$. For a silent user u_j, his/her opinion on topic t_i is the value in i-th row and j-th column of $\widetilde{\mathbf{O}}$. In the objective function, the last term is a regularization of penalty to prevent overfitting, where the $\ell_{2,1}$-norm regularization is added on each latent matrix. The features we extract contain different types of knowledge, and we assume that only parts of them are useful for opinion inference. The $\ell_{2,1}$-norm can promote row-wise sparsity of the target matrix, making it suitable for feature selection. Hence, the last term can also help control group feature selection.

5 OPTIMIZATION ALGORITHM

The objective function we defined in the previous section is not jointly convex to all the variables. As a result, it is not easy for us to obtain a closed-form solution for minimizing this objective function. To solve this problem, we utilize an iterative algorithm, where we alternatively update one variable while fix other variables until convergence. Note that $\ell_{2,1}$-norm is not continuous on the origin. Hence, we cannot minimize the compound $\ell_{2,1}$ objective function directly. Motivated by recent studies on half-quadratic minimization approaches [27], we can first introduce an auxiliary variable, and then apply this auxiliary variable to transform the $\ell_{2,1}$-norm term of the objective function into an approximate form. For instance, $\mathbf{G}$ can be transformed as:

$$||\mathbf{G}||_{2,1} \approx tr(\mathbf{G}^T\mathbf{Q_G}\mathbf{G}) \tag{12}$$

where $\mathbf{Q_G}$ is a diagonal matrix with the i-th diagonal element as $q^g_{i,i} = \frac{1}{2||\mathbf{g_i}||_2}$. $\mathbf{T}$, $\mathbf{U}$ and $\mathbf{F}$ can be dealt with in the same way.

After having the above transformations, we then compute the derivative of the objective function with respect to $\mathbf{T}$, $\mathbf{U}$, $\mathbf{G}$ and $\mathbf{F}$ respectively:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbf{T}} &= [\mathbf{I} \circ (\mathbf{T}\mathbf{U}^T - \mathbf{O})]\mathbf{U} + \lambda_1(\mathbf{T}\mathbf{G}^T - \mathbf{C})\mathbf{G} + \lambda_4 \mathbf{Q}_\mathbf{T}\mathbf{T} \\
\frac{\partial \mathcal{L}}{\partial \mathbf{U}} &= [\mathbf{I} \circ (\mathbf{T}\mathbf{U}^T - \mathbf{O})]^T\mathbf{T} + \lambda_2(\mathbf{U}\mathbf{F}^T - \mathbf{Z})\mathbf{F} \\
&\quad + \lambda_3 \mathbf{L}_\mathbf{S}\mathbf{U} + \lambda_4 \mathbf{Q}_\mathbf{U}\mathbf{U} \\
\frac{\partial \mathcal{L}}{\partial \mathbf{G}} &= \lambda_1[\mathbf{T}\mathbf{G}^T - \mathbf{C}]^T\mathbf{T} + \lambda_4 \mathbf{Q}_\mathbf{G}\mathbf{G} \\
\frac{\partial \mathcal{L}}{\partial \mathbf{F}} &= \lambda_2[\mathbf{U}\mathbf{F}^T - \mathbf{Z}]^T\mathbf{U} + \lambda_4 \mathbf{Q}_\mathbf{F}\mathbf{F}
\end{aligned} \tag{13}$$

With the derivatives in place, a conjugate gradient method can be employed to iteratively minimize the objective function. We show the detailed process of the optimization algorithm in Algorithm 1.

Algorithm 1 Algorithm to minimize the objective function

Input: User-topic opinion matrix $\mathbf{O}$, community-topic opinion matrix $\mathbf{C}$, user feature matrix $\mathbf{Z}$, Laplacian matrix $\mathbf{L}_\mathbf{S}$ and regularization parameters.

Output: $\mathbf{T}$ and $\mathbf{U}$.

1: Initialize $\mathbf{T}$, $\mathbf{U}$, $\mathbf{G}$ and $\mathbf{F}$ randomly.
2: Compute the diagonal matrices $\mathbf{Q}_\mathbf{T}$, $\mathbf{Q}_\mathbf{U}$, $\mathbf{Q}_\mathbf{G}$ and $\mathbf{Q}_\mathbf{F}$.
3: **repeat**
4: Update $\mathbf{T}$, $\mathbf{U}$, $\mathbf{G}$ and $\mathbf{F}$ according to Eq. (13) by using the conjugate gradient method.
5: Update $\mathbf{Q}_\mathbf{T}$, $\mathbf{Q}_\mathbf{U}$, $\mathbf{Q}_\mathbf{G}$ and $\mathbf{Q}_\mathbf{F}$ with $q^t_{i,i} = \frac{1}{2||\mathbf{t_i}||_2}$, $q^u_{i,i} = \frac{1}{2||\mathbf{u_i}||_2}$, $q^g_{i,i} = \frac{1}{2||\mathbf{g_i}||_2}$, $q^f_{i,i} = \frac{1}{2||\mathbf{f_i}||_2}$.
6: **until** convergence

6 EXPERIMENTS

6.1 Evaluation Metrics

We employ two popular metrics, i.e., root mean square error (RMSE) and Accuracy, to measure the performance of opinion inference task. The metric RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{N_T}\sum_{i,j}(o_{i,j} - \widetilde{o}_{i,j})^2} \tag{14}$$

where N_T stands for the number of opinions for testing. Since the value of $\widetilde{o}_{i,j}$ may be not an integer which exactly equals to -1, 0 or 1, we map the value of $\widetilde{o}_{i,j}$ with the *sign function* before computing the Accuracy. $\widetilde{o}_{i,j}$ is mapped to -1 if $\widetilde{o}_{i,j} \in (-\infty, -0.5]$, 0 if $\widetilde{o}_{i,j} \in (-0.5, 0.5)$, and 1 if $\widetilde{o}_{i,j} \in [0.5, +\infty)$. In experiments, a smaller RMSE value or a higher Accuracy value indicates a better inference performance.

6.2 Comparison with Baselines

In this subsection, we compare our CSMF model with several baseline methods. To analyze how the performance of these methods changes with the change of active user size, we select 30, 40, 60, 70 and 80 percent of the most active users as active users, and infer the opinions of the remaining portions of users (silent users). For users in the test set (silent users), all their posted messages are removed when extracting features for them, and their observed user-topic opinions are regarded as ground truth data. Then we predict their opinions using different methods. The baseline methods are listed as follows:

- RK: Regression-Kriging is a liner combination estimator that combines a regression of the dependent variable on auxiliary variables with the regression residuals.
- MF: This method is the basic low-rank matrix factorization model, as shown in Eq. (7).
- STMF: It is proposed in [17], where a matrix factorization framework is built to predict user-topic opinions by considering users' previous posted text content and the similarities between different users.
- DCMF: This method [12] first learns hash codes for users and topics from content information, and then introduces an interaction regularization to tackle the data sparsity problem.
- OP-SVM: It is an opinion prediction model based on two types of features (sentiment features and opinion features) extracted from one's historical posted messages on other topics [7].
- LoCo: It is a matrix factorization framework designed for cold-start recommendation systems [19]. The inference of silent users' opinions is intrinsically similar to cold-start recommendation problem. Therefore, we adopt this method as a baseline method.

Table 1: RMSE comparisons using different methods and sizes of active users.

Active users	*RK*	*MF*	*STMF*	*DCMF*	*OP-SVM*	*LoCo*	*CSMF*
30%	0.92	0.80	0.75	0.71	0.68	0.70	**0.59**
40%	0.88	0.76	0.71	0.67	0.66	0.65	**0.56**
60%	0.87	0.72	0.69	0.64	0.65	0.64	**0.52**
70%	0.83	0.71	0.68	0.63	0.62	0.60	**0.50**
80%	0.81	0.71	0.65	0.62	0.61	0.58	**0.49**

In this experiment, for the methods using regularization constraints, we adjust the regularization parameters of them in order to obtain their best performance. In our CSMF model, the best result is achieved when λ_1, λ_2, λ_3 and λ_4 are set as 1, 10^{-1}, 10^{-1} and 1 respectively. In addition, we only consider directed follow network when extracting features for users, and set γ, α and K as 0.7, 0.6 and 65 respectively based on the results of pre-performed testing for parameter tuning. We display the experimental results measured by RMSE and Accuracy in Table 1 and Figure 1.

From Table 1 and Figure 1, we can see that all these methods can achieve much better performance than randomly guessing (which is $\frac{1}{3}$ in our task). When we increase the size of active users, the performance of these methods gets improved gradually. RK method achieves the worst performance among all baseline methods, both by RMSE and Accuracy. The underlying reason may be that the relationships among different features are so complex that they cannot be captured by a liner model. Compared with basic MF model,

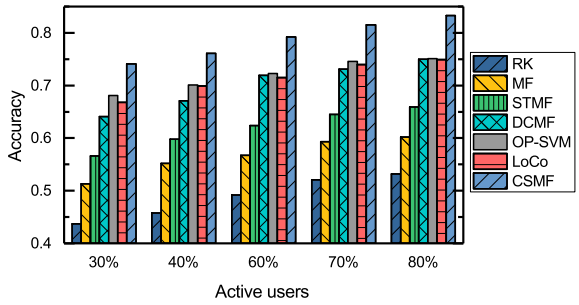


Figure 1: Accuracy comparisons using different methods and sizes of active users.

STMF, DCMF, LoCo and CSMF can obtain significantly improved results, indicating that the regularization constraints improve the inference of silent users' opinions. STMF performs much worse than DCMF, OP-SVM, LoCo and CSMF, which may be caused by that STMF relies on a large number of previous published messages per user to compute the similarities between users while few messages are available for each user in our dataset. For OP-SVM model, one type of features used in the model (i.e., opinion features) cannot be extracted when the user's all ground truth opinions on other topics are not available. As a result, OP-SVM model also obtains worse result than our CSMF model. Our proposed CSMF model always generates the best performance, and this observation demonstrates that incorporating text content and network structural features actually benefits the inference of silent users' opinions.

6.3 Effect of User-User Relations

When considering the explicit relationships (i.e., the follower/followee network structure) among users in the previous section, we define two types of connections between users, i.e., directed follow network and mutual follow network. In addition, we infer pseudo-relationships among users before extracting structural features for users. Here we analyze how the performance of inferring silent users' opinions is affected by the type of connection and pseudo-relationships. In this experiment, we select 60% of the whole users as active users. The experimental results are shown in Figure 2.

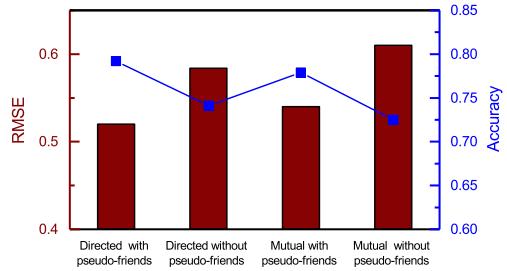


Figure 2: Effect of user-user relations on RMSE and Accuracy.

As we can see from Figure 2, the best performance is achieved when both directed follow network and pseudo-friend information are taken into consideration. For both two types of connections between users, much better results are generated after pseudo-relationships are combined with the explicit relationships. This result demonstrates that the pseudo-relationships are indeed highly correlated with shared opinions among users. Comparing the directed network with the mutual network, the directed network performs slightly better than the mutual network, indicating that attention effects are more important than homophily effects for shared opinions in social networks.

7 CONCLUSION

Existing studies on inferring users' opinions basically rely on one's previous posted messages. As silent users do not generate sufficient content, the performance of existing solutions will drop dramatically if they are applied to infer the opinions of silent users. To solve this problem, we model the opinion inference problem as a matrix factorization problem, and propose a coupled sparse matrix factorization (CSMF) model to infer the opinions of silent users. We propose several sets of features for opinion inference from different perspectives, and capture the complex relations among these features through the CSMF model. Experimental results based on real-world data from Twitter demonstrate that our model achieves much better result than the state-of-the-art methods.

ACKNOWLEDGMENTS

This work was supported by National Key R&D Program of China (2017YFB1301100), National Natural Science Foundation of China (61572060, 61772060, 61572218, 61728201) and CERNET Innovation Project (NGII20170315, NGII20160316).

REFERENCES

[1] Stephen P Borgatti. 2005. Centrality and network flow. *Social networks* 27, 1 (2005), 55–71.
[2] Hongxu Chen, Hongzhi Yin, Xue Li, Meng Wang, Weitong Chen, and Tong Chen. 2017. People opinion topic model: opinion based user clustering in social networks. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 1353–1359.
[3] Kewei Cheng, Jundong Li, Jiliang Tang, and Huan Liu. 2017. Unsupervised Sentiment Analysis with Signed Social Networks.. In *AAAI*. 3429–3435.
[4] Hongbo Deng, Jiawei Han, Hao Li, Heng Ji, Hongning Wang, and Yue Lu. 2013. Exploring and Inferring User-User Pseudo-Friendship for Sentiment Analysis with Heterogeneous Networks. In *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 378–386.
[5] Mauro Dragoni. 2017. A three-phase approach for exploiting opinion mining in computational advertising. *IEEE Intelligent Systems* 32, 3 (2017), 21–27.
[6] Wei Gong, Ee-Peng Lim, and Feida Zhu. 2015. Characterizing Silent Users in Social Media Communities.. In *ICWSM*. 140–149.
[7] Wei Gong, Ee-Peng Lim, Feida Zhu, and Pei Hua Cher. 2016. On Unravelling Opinions of Issue Specific-Silent Users in Social Media. ICWSM.
[8] Kristen Johnson, Di Jin, and Dan Goldwasser. 2017. Leveraging Behavioral and Social Information for Weakly Supervised Collective Classification of Political Discourse on Twitter. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 741–752.
[9] Jihie Kim, Jae-Bong Yoo, Ho Lim, Huida Qiu, Zornitsa Kozareva, and Aram Galstyan. 2013. Sentiment Prediction Using Collaborative Filtering.. In *ICWSM*.
[10] Yoree Koh. 2014. Report: 44% of Twitter Accounts Have Never Sent a Tweet. In *The Wall Street Journal*. News Corporation.
[11] Tao Li, Yi Zhang, and Vikas Sindhwani. 2009. A non-negative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*. Association for Computational Linguistics, 244–252.
[12] Defu Lian, Rui Liu, Yong Ge, Kai Zheng, Xing Xie, and Longbing Cao. 2017. Discrete content-aware matrix factorization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 325–334.
[13] Shangsong Liang, Xiangliang Zhang, Zhaochun Ren, and Evangelos Kanoulas. 2018. Dynamic Embeddings for User Profiling in Twitter. In *Proceedings of the 24th*

ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 1764–1773.

[14] Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies* 5, 1 (2012), 1–167.

[15] Zongqing Lu, Xiao Sun, Yonggang Wen, Guohong Cao, and Thomas La Porta. 2015. Algorithms and applications for community detection in weighted networks. *IEEE Transactions on Parallel and Distributed Systems* 26, 11 (2015), 2916–2926.

[16] Huifang Ma, Meihuizi Jia, Di Zhang, and Xianghong Lin. 2017. Combining tag correlation and user social relation for microblog recommendation. *Information Sciences* 385 (2017), 325–337.

[17] Fuji Ren and Ye Wu. 2013. Predicting user-topic opinions in twitter with social and topical context. *IEEE Transactions on Affective Computing* 4, 4 (2013), 412–424.

[18] Amin Salehi, Mert Ozer, and Hassan Davulcu. 2018. Sentiment-driven Community Profiling and Detection on Social Media. In *Proceedings of the 29th on Hypertext and Social Media (HT '18)*. ACM, New York, NY, USA, 229–237.

[19] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, Lexing Xie, and Darius Braziunas. 2017. Low-Rank Linear Cold-Start Recommendation from Social Data.. In *AAAI*. 1502–1508.

[20] Peter Sprent and Nigel C Smeeton. 2000. *Applied nonparametric statistical methods*. Chapman and Hall/CRC.

[21] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment* 4, 11 (2011), 992–1003.

[22] Andrea Tagarelli and Roberto Interdonato. 2013. Who's out there?: identifying and ranking lurkers in social networks. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 215–222.

[23] Chenhao Tan, Lillian Lee, Jie Tang, Long Jiang, Ming Zhou, and Ping Li. 2011. User-level sentiment analysis incorporating social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1397–1405.

[24] Rodoula H Tsiotsou. 2015. The role of social and parasocial relationships on social networking sites loyalty. *Computers in Human Behavior* 48 (2015), 401–414.

[25] Nate Veldt, David F Gleich, and Anthony Wirth. 2018. A Correlation Clustering Framework for Community Detection. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 439–448.

[26] Svitlana Volkova, Glen Coppersmith, and Benjamin Van. 2014. Inferring user political preferences from streaming communications. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 186–196.

[27] Yuqi Wang, Jiannong Cao, Lifang He, Wengen Li, Lichao Sun, and Philip S Yu. 2017. Coupled Sparse Matrix Factorization for Response Time Prediction in Logistics Services. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 939–947.

[28] Liang Yang, Xiaochun Cao, Di Jin, Xiao Wang, and Dan Meng. 2015. A unified semi-supervised community detection framework using latent space graph regularization. *IEEE transactions on cybernetics* 45, 11 (2015), 2585–2598.

[29] Lei Zhang, Hebin Pan, Yansen Su, Xingyi Zhang, and Yunyun Niu. 2017. A mixed representation-based multiobjective evolutionary algorithm for overlapping community detection. *IEEE Transactions on Cybernetics* 47, 9 (2017), 2703–2716.

[30] Guoshuai Zhao, Xueming Qian, and Xing Xie. 2017. User-service rating prediction by exploring social users' rating behaviors. *IEEE transactions on multimedia* 18, 3 (2017), 496–506.

A Novel Unsupervised Approach for Precise Temporal Slot Filling from Incomplete and Noisy Temporal Contexts

Xueying Wang
University of Notre Dame
Notre Dame, IN
xwang41@nd.edu

Haiqiao Zhang
University of Notre Dame
Notre Dame, IN
hzhang5@nd.edu

Qi Li
University of Notre Dame
Notre Dame, IN
qli8@nd.edu

Yiyu Shi
University of Notre Dame
Notre Dame, IN
yshi4@nd.edu

Meng Jiang
University of Notre Dame
Notre Dame, IN
mjiang2@nd.edu

ABSTRACT

The task of temporal slot filling (TSF) is to extract the values (or called facts) of specific attributes for a given entity from text data and find the time points when the values were valid. It is challenging to find precise time points with incomplete and noisy temporal contexts in the text. In this work, we propose an unsupervised approach of two modules that mutually enhance each other: one is a reliability estimator on fact extractors conditionally to the temporal contexts; the other is a fact trustworthiness estimator based on the extractor's reliability. The iterative learning process reduces the noise of the extractions. Experiments demonstrate that our approach, with the novel design, can accurately and efficiently extract precise temporal facts from newspaper corpora.

KEYWORDS

Temporal slot filling, Incompleteness, Information extraction

ACM Reference Format:
Xueying Wang, Haiqiao Zhang, Qi Li, Yiyu Shi, and Meng Jiang. 2019. A Novel Unsupervised Approach for Precise Temporal Slot Filling from Incomplete and Noisy Temporal Contexts. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313435

1 INTRODUCTION

Can AI read ten million news articles in two minutes and then fill in the three slots below:

("vicente_fox", per:is_president_of, "□", [□ , □]) ?
(entity, attribute, value, [beginTime, endTime])

The first slot □ is the value of a specific attribute (e.g., country's president) for an entity (e.g., the person "vicente_fox"). Here the value should be a country's name. The second and third slots are the *beginning* and *ending* time points of the attribute value being valid. We name this task "precise temporal slot filling" (PTSF).

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313435

PTSF techniques will facilitate the automation of knowledge base construction and question answering.

Existing TSF task presents the time as a single slot and expects to extract time expressions directly from sentences to fill the slot. For example, given a sentence in the newspaper:

"*...Vicente Fox served as the President of Mexico from 2000 to 2006,...*"

TSF looks for "mexico" to fill the value slot and the time expression "from 2000 to 2006" to fill the time slot. Decomposing this time expression into two time values is not hard in this case, in order to solve PTSF with the TSF result. However, we observe from a data set of ten million news articles that among all the sentences that contain time expressions, fewer than 0.1% contain at least two time points like "from 2000 to 2006:" most of the expressions are shorter like "in 2002," which indicates a time point of the fact but cannot precisely complete the *beginTime* and *endTime* slots.

We examined two information extraction methodologies: open-domain IE (OpenIE) and pattern-based IE (PatternIE). We find that the OpenIE approaches [1, 2, 7, 10, 25] generate *low precision on the value slots* and *lower-than-0.1 recall on the time slots*. First, OpenIE relies on predicates to extract relations between a subject and an object, however, the attribute may not be a predicate. Given the above sentence example, OpenIE generates ("vicente_fox", "serve_as", "president_of_mexico") instead of finding "mexico" as the value of attribute "is_president_of". Second, OpenIE could not find the precise time points if there was very few such long time expressions as "from 2000 to 2006" in the sentences that contain the fact. On the other hand, PatternIE methods [16, 17, 22] use textual patterns to find the attribute values, but very few methods associate the values with time information. A pattern-based temporal anchoring method [23] assumes that both types of temporal contexts, time expression and document post time, are accurate to be the time of the attribute value being valid. However, this assumption is too strong to find precise temporal slots from the noisy contexts. For example, given the sentences:

"*In 1979, the [former U.S. President Jimmy Carter] deregulated the American beer industry...*" (posted on August 5, 2010), "*[Donald Trump, now President of United States,] published his first book in 1987...*" (posted on June 3, 2017),

the post time in the first sentence "2010" is not in the presidential term of President Carter (1977–1981), and the time expression in

Table 1: Pattern's reliability scores for country's presidency.

Textual Pattern p	$r^{(post)}(p)$	$r^{(tag)}(p)$
$Country president $Person	0.91	0.53
$Country 's president $Person	0.86	0.84
president $Person of $Country	0.84	0.70
former $Country president $Person	-1	0.85
$Country 's former president $Person	-0.81	0.83
$Person , now president of $Country ,	0.95	-0.89
current $Country president $Person	0.93	-0.59
$Country 's current president , $Person ,	0.81	0
new $Country president , $Person ,	0.57	-0.25
$Country 's newly elected president , $Person ,	0.20	-0.64
current $Country prime minister $Person	-1	-1
$Country premier $Person	-0.82	-0.86
$Country foreign minister $Person	-1	-1
$Country golfer $Person	-1	-1

the second sentence "1987" is not in the term of President Trump, either. We observe that different textual patterns may have different *reliability* on extracting the time points, conditionally to the temporal contexts (including time expression and post time):

• Pattern [former $Country president $Person] is reliable for the time expression "1979", not the post time "2010."

• Pattern [$Person, now president of $Country,] is reliable for the post time "2018", not the time expression "1987."

In this work, we propose a novel unsupervised approach based on PatternIE to infer the precise temporal slots from *incomplete* and *noisy* temporal contexts. For an attribute, PatternIE discovers a large set of textual patterns and uses them to extract EVT-tuples from text data: $p \rightarrow \{(e, v, t)\}$, where p is a textual pattern, e denotes the entity, v denotes the attribute value, and t is a time point from either time expression or post time. The goal is to infer temporal fact tuples $\{(e, v, [t_b, t_e])\}$, where t_b/t_e is beginTime/endTime, from millions of EVT-tuples (e.g., 5.3M for country's president). Our idea is to jointly estimate the pattern's reliability and the tuple's trustworthiness: if a set of tuples (including the time point) are more trustworthy, the pattern that extracted these tuples is more likely to be reliable; and, if a pattern is more reliable, its extractions are more likely to be true. Effectiveness of this iterative truth-discovery methodology has been demonstrated for finding the book's true author names from information on book-selling websites [18, 34].

Unfortunately, the truth-discovery algorithms cannot be directly applied to the proposed problem because they often hold the single-truth assumption (e.g., a book has only one true author list) to find conflicting values and thus capture unreliable sources [9, 19, 31, 34]. However, in the task of fact extraction, an entity may have multiple values at multiple time points. Then how can we know that the tuple ("u.s.", "jimmy_carter", 2010) is wrong (w.r.t. country's president) when United States had multiple presidents? Our common sense can help: if we had a trustworthy tuple ("u.s.", "barack_obama", 2010) and we knew that one country is likely to have *only one* president in one year, we would have a chance to find the conflicts.

Based on the above observation, we propose to use *the World's invariants* (i.e., quantitative common senses) including time-irrelevant and time-relevant constraints to find the conflicts and the truth. In the following section, we show that with just one seed pattern, we can generate and validate (1) time-irrelevant hypotheses:

• H1: one country is likely to have *multiple* presidents in the history;
• H2: one president is likely to serve *only one* country;
and (2) time-relevant hypothesis:
• H3: in one year, one country is likely to have *only one* president.
H2 and H3 can be used as constraints to find conflicts between tuples while H1 cannot. For example, suppose the fact ("barack_obama", "u.s.", [2009, 2017]) is true. So ("barack_obama", "china", 2014), which was extracted from the following sentence:

"*[President Barack Obama visited China] and attended the APEC summit...*" (posted on November 11, 2014),

is false because of H2 and ("jimmy_carter", "u.s.", 2010) is false because of H3. Then we know that (1) the pattern [president $Person visited $Country] is unreliable to extract a fact of country's president and (2) [former $Country president $Person] is unreliable to claim the fact's time point as the post time.

Based on the above ideas, estimating pattern reliability and finding conflicts with the World's invariants, we propose a Truth Finding-driven framework using the World's INvariants, called TFWIN, to extract precise temporal facts from text corpus. First, it uses PatternIE to structure the corpus into textual patterns and (e, v, t)-tuples. Second, it uses hypothesis testing to derive time-irrelevant and time-relevant constraints. Third, it iteratively evaluates pattern's reliability (upon two different temporal context types), estimates time point's trustworthiness, updates beginTime/endTime slots, and detects false tuples using the constraints. Two important properties of this algorithms are: (1) the time complexity is *quasi-linear* to the corpus size; and (2) it requires *NO* expensive annotations or heavy parameter tuning.

Table 1 presents the reliability scores given to a few pattern examples by our algorithm, where $r^{(post)}(p)$ and $r^{(tag)}(p)$ denote the reliability of pattern p on claiming the post time and temporal tag (i.e., time expression) as a valid time point, respectively. The score ranges from -1 to 1: -1 means that the extractions by the pattern are very likely to be false w.r.t. the specific attribute; 1 means the extractions are likely to be true; 0 means the extractions have very little correlation with the attribute. We observed that a textual pattern could be positive or negative on both, or positive on one and negative on the other, or vice versa. The pattern reliability match our intuition and effectively estimates the truth of value, beginTime, and endTime slots.

We summarize our main contributions as follows.

- We study the problem of precise temporal slot filling and point out the limitations of existing OpenIE and PatternIE.
- We propose the ideas of estimating pattern reliability and detecting conflicts with the World's invariants to handle incompleteness and noise of temporal contexts in text data.
- We propose a novel unsupervised framework (TFWIN) to find precise temporal facts from massive general corpora with no requirement of human annotations.
- Experiments demonstrated the effectiveness. AUC and F1 were improved by 25+% over the state-of-the-art.

The rest of this paper is organized as follows. Section 2 provides data preprocessing and problem definition. Section 3 presents the overview and details of the proposed framework. Experimental

results can be found in Section 4. Section 5 surveys the literature of related works. Section 6 concludes the paper.

2 PRELIMINARIES

We first introduce the techniques to turn the text data into "pattern-to-(entity, value, time)-tuples." Then we define the problem.

2.1 Structuring Text into "Pattern-Tuple"

Pattern-based methods are the most popular for information extraction in an unsupervised way from massive text corpora. The idea is that the textual patterns become frequent when entity names in the patterns are replaced by symbols $E (entity) or $V (value) [12, 13, 33] or their types like $Person or $Country [16, 22]. The type-level textual patterns can generate a large set of concrete (entity, value)-tuples from sentences. Then we will introduce how to have the "pattern-(entity, value, *time*) tuple" structures in detail.

2.1.1 Entity recognition and typing. We use the entity recognition and typing techniques to jointly recognize entity names and their *fine-grained* types simultaneously. For example, country names such as "United States", "Mexico", and "Burkina Faso" are recognized and typed as "$Location.Country" (simplified as "$Country").

2.1.2 Textual pattern mining. We use the textual pattern mining method MetaPAD [16] to discover "meta patterns" as information extractors. The meta pattern is defined in [16] as below.

Definition 2.1 (Meta Pattern). A meta pattern refers to a frequent textual pattern of entity types (e.g., $Country, $Person), words, and possibly punctuation marks, which serves as an integral semantic unit in certain context.

2.1.3 EVT-tuples and precise temporal fact tuples. For a specific attribute (e.g., country's president), the meta patterns of the corresponding entity type (e.g., $Country) and value type (e.g., $Person) can generate a set of (entity, value)-pairs. To discover temporal facts, we attach two types of time signals to the tuples: One is the "post time" which is the time of the document being posted, and the other is "time expression" or called "temporal tag" which is the nearest temporal tags (within a 20-word window) to the entity mention. We use a popular tagging tool [29] to extract the temporal tags. Now we can define the EVT-tuples as below.

Definition 2.2 (EVT-tuple). For a specific attribute a that refines the entity type as $c_e(a)$ and the value type as $c_v(a)$, an EVT-tuple refers to an (e, v, t)-tuple, where the type of e is $c_e(a)$, the type of v is $c_v(a)$, (e, v)-pair is extracted by a pattern p, and t is the timestamp attached to the pair.

Given the text data, we use the above techniques to preprocess the data and find millions of textual patterns, EVT-tuples, and their associations. We look for precise temporal fact tuples:

Definition 2.3 (Precise temporal fact tuple). For a specific attribute a, a temporal fact tuple refers to an $(e, v, [t_b, t_e])$-tuple, where for any time $t \in [t_b, t_e]$, v is a valid attribute value of e's attribute a. The beginTime t_b and endTime t_e must be precisely specified as time values (e.g., a concrete year, month, or date) instead of text-based time expressions (e.g., "from ... to ...", "since ...").

Table 2: Symbols and their descriptions.

Symbol	Description
$\mathcal{D}^{(*)}$ $* \in \{\text{"}post\text{"}, \text{"}tag\text{"}\}$	"Pattern-tuple" extraction list in which time signal comes from $*$
$\mathcal{P}$	The set of textual patterns
(e, v, t)	EVT-tuple (entity, value, time)
$c^{(*)}(p, (e, v, t))$	The count of times p extracts (e, v, t)
$(e, v, [t_b, t_e])$	Precise temporal fact tuple
$\mathcal{F}$	The list of true temporal facts
$r^{(*)}(p)$	The reliability score of pattern p
$w((e, v, t))$	The trustworthiness score of EVT-tuple
$\mathcal{H}$	A set of hypotheses to define conflicts
$f((e, v, t), \mathcal{F}, \mathcal{H}) \in$ $\{\text{"}T\text{"}, \text{"}F\text{"}, \text{"}U\text{"}\}$	Flag of checking (e, v, t) with $\mathcal{F}$ on $\mathcal{H}$: True, False, Undetermined
$\mathcal{P}^{(*)}_{(e,v,t)}$	Pattern set that extracts (e, v, t) from $\mathcal{D}^{(*)}$
$\mathcal{D}^{(*)}_{p}$	EVT-tuple set extracted by p from $\mathcal{D}^{(*)}$

2.2 Problem Definition

Table 2 describes the symbols we use in this paper. With the above concepts defined in Section 2.1, we define the problem of precise temporal fact extraction on the "pattern-tuple" structured data.

Problem (Precise Temporal Fact Extraction)**. Given** two "pattern-tuple" structured extraction lists $\mathcal{D}^{(post)}$ and $\mathcal{D}^{(tag)}$, that can be represented as $\mathcal{D}^{(*)} = \left\{\left(p, (e, v, t), c\right)\right\}$ in which the time t comes from two kinds of signals, i.e., "post" for the time of document being posted and "tag" for the nearest temporal tag, and where c can be concretely written, for example, $c^{(tag)}(p, (e, v, t))$ is the count of times that textual pattern p extracts the (e, v, t)-tuple along the temporal tag t, **(1) estimate** the reliability of each textual pattern that is described as a function $r^{(*)}(p) : p \in \mathcal{P} \rightarrow [-1, 1]$, where $\mathcal{P}$ is the set of textual patterns, **(2) infer** the trustworthiness of each EVT-tuple that is described as a function: $w(e, v, t) : (e, v, t) \rightarrow [-1, 1]$, and **(3) find** the list of true temporal fact tuple $\mathcal{F} = \left\{\left(e, v, [t_b, t_e]\right)\right\}$.

3 THE PROPOSED APPROACH

3.1 Overview

Figure 1 presents the illustration of the proposed TFWIN framework using the attribute *country's president* as an example.

The unsupervised approach is initialized by one seed pattern (assuming that it has high reliability) and iteratively does two-step learning: step (a) is to estimate the tuple trustworthiness based on pattern reliability and to update the two precise time slots of temporal facts with trustworthy time points; step (b) is to find true EVT-tuples (if satisfies the precise temporal facts) and false EVT-tuples (if conflicts with the World's invariants), and then to estimate the pattern reliability based on tuple trustworthiness. It will converge when all the EVT-tuples were separated into two parts: one can be located into the precise temporal facts, and the other violate at least one precise temporal fact holding the constraints.

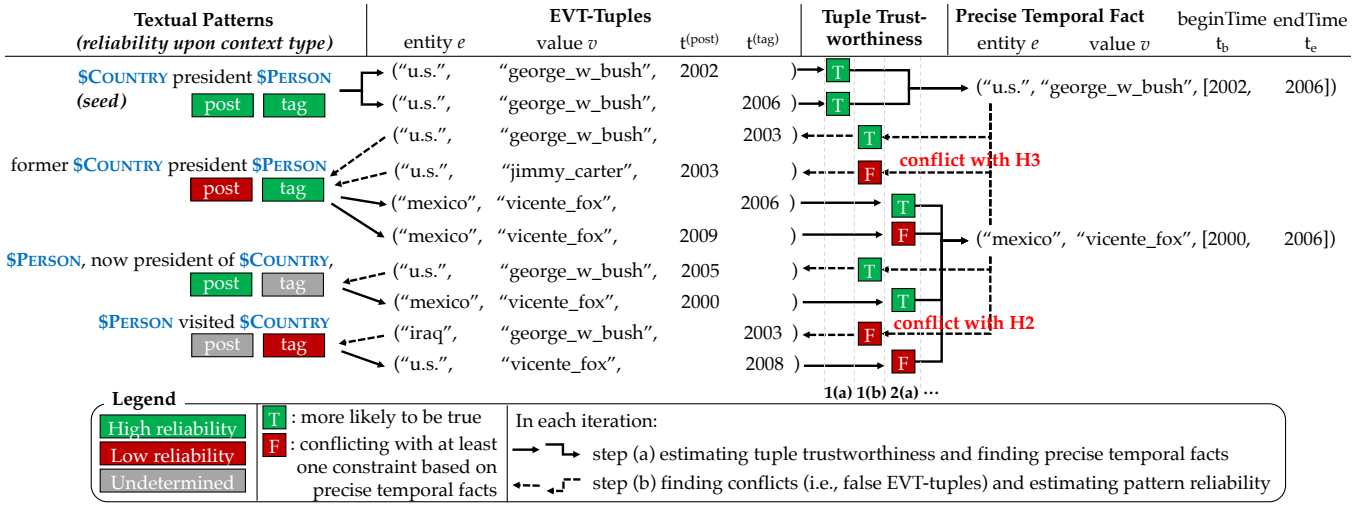


Figure 1: Our proposed TFWIN framework iteratively estimates reliability of textual patterns as information sources, infers trustworthiness of temporal facts, and resolves conflicts defined by the World's invariants (e.g., H2 and H3).

3.2 The Iterative Learning Algorithm

Generally, the algorithm is an iterative method. It starts with very light supervisory information, that is, *one* highly reliable pattern. Usually we use the pattern [$TypeOf*e* *a* $TypeOf*v*] as the seed pattern for attribute *a*. This pattern is not necessarily the most frequent one though most of the time it is. Since only one pattern is needed as the seed pattern, it will not take a lot of effort to find one. For example, [$Country president $Person] is a reliable seed pattern for the attribute *country's president*.

We use the frequent EVT-tuples extracted by the seed pattern to generate seed temporal fact tuples till *a conflict occurs*. Then we generate negative labels (i.e., false EVT-tuples) based on the constraints. With the positive and negative tuples, we iteratively estimate the reliability of textual patterns and infer the trustworthiness of undetermined tuples. We use tuples of good trustworthy scores, from the highest to low non-negatives, to update the positive labels (i.e., temporal fact tuples) till *a conflict occurs* or *the tuple's support is below a threshold* α and re-generate the negative labels. When it comes to convergence, the algorithm returns the final set of temporal facts.

3.2.1 Conflicts and negative label generation. Here we define a function of checking if an (e, v, t)-tuple is conflicted with existing true temporal facts $\mathcal{F}$ based on the set of hypotheses $\mathcal{H}$:

$$f((e,v,t),\mathcal{F},\mathcal{H}) = \begin{cases} \text{“}T\text{”}, & \text{if a fact tuple in } \mathcal{F} \text{ includes } (e,v,t); \\ \text{“}F\text{”}, & \text{if } (e,v,t) \text{ conflicts with } \mathcal{F} \text{ on any hypothesis } H \in \mathcal{H}; \\ \text{“}U\text{”}, & \text{else;} \end{cases}$$

where “T” denotes for True, “F” denotes for False, and “U” denotes for Undetermined. More formally, $f((e,v,t),\mathcal{F},\mathcal{H}) = \text{“}T\text{”}$ for $\forall\mathcal{H}$, if the statement

$$\exists(e,v,[t_b,t_e]) \in \mathcal{F}, t_b \le t \le t_e \tag{1}$$

is true. $f((e,v,t),\mathcal{F},\{H_{1e-to-1v}\}) = \text{“}F\text{”}$, if the statement

$$\exists(e,v',[t_b,t_e]) \in \mathcal{F}, v' \neq v \tag{2}$$

is true. $f((e,v,t),\mathcal{F},\{H_{1v-to-1e}\}) = \text{“}F\text{”}$, if the statement

$$\exists(e',v,[t_b,t_e]) \in \mathcal{F}, e' \neq e \tag{3}$$

is true. $f((e,v,t),\mathcal{F},\{H_{(1t)1e-to-1v}\}) = \text{“}F\text{”}$, if the statement

$$\exists(e,v',[t_b,t_e]) \in \mathcal{F}, t_b \le t \le t_e \text{ and } v' \neq v \tag{4}$$

is true. $f((e,v,t),\mathcal{F},\{H_{(1t)1v-to-1e}\}) = \text{“}F\text{”}$, if the statement

$$\exists(e',v,[t_b,t_e]) \in \mathcal{F}, t_b \le t \le t_e \text{ and } e' \neq e \tag{5}$$

is true. At the i-th iteration, given $\mathcal{F}^{(i)}$ and $\mathcal{H}$, we assign *polarized* trustworthiness score to tuples as below:

$$w((e,v,t)) = \begin{cases} 1, & \text{if } f((e,v,t),\mathcal{F},\mathcal{H}) = \text{“}T\text{”}; \\ 0, & \text{if } f((e,v,t),\mathcal{F},\mathcal{H}) = \text{“}U\text{”}; \\ -1, & \text{if } f((e,v,t),\mathcal{F},\mathcal{H}) = \text{“}F\text{”}. \end{cases} \tag{6}$$

Then we have positive/negative labels to estimate pattern reliability.

3.2.2 Pattern reliability estimation. The reliability score of pattern p can be estimated from the trustworthiness of its EVT-tuples:

$$r^{(*)}(p) = \frac{\sum_{(e,v,t)\in\mathcal{D}_p^{(*)}} c^{(*)}(p,(e,v,t))w((e,v,t))}{\sum_{(e,v,t)\in\mathcal{D}_p^{(*)}} c^{(*)}(p,(e,v,t))|w((e,v,t))|} \in [-1,1], \tag{7}$$

where $*$ is for temporal context type (or called time signal source, either “post” for post time or “tag” for temporal tag) and $c^{(*)}(p,(e,v,t))$ is the count of times that the (e,v,t)-tuples are extracted by p. The idea is that a pattern is more (un-)reliable, if its EVT-tuples are more (un-)trustworthy. Note that the pattern reliabilities are separately modeled based on different time signal sources. In the experiments, we will compare the performances of our algorithm's settings: (1) between source-aware and source-unaware modeling and (2) between considering and not considering counts.

3.2.3 Tuple trustworthiness inference. When the pattern reliabilities are estimated, we evaluate the trustworthiness of the *undetermined* tuples as below:

$$w((e,v,t)) = \frac{\sum_* \sum_{p \in \mathcal{P}^{(*)}_{(e,v,t)}} c^{(*)}(p,(e,v,t)) r^{(*)}(p)}{\sum_* \sum_{p \in \mathcal{P}^{(*)}_{(e,v,t)}} c^{(*)}(p,(e,v,t))} \in [-1,1], \quad (8)$$

where we integrate pattern reliability's contributions from both time signal sources. If an EVT-tuple is often extracted by (un-)reliable patterns, it is more (un-)trustworthy. In the experiments, we will investigate the effectiveness of considering counts c.

4 EXPERIMENTS

Here we conduct experiments to answer the following quesitons:

Q1. (Effectiveness) Is TFWIN effective in mining temporal facts from text data? Do both time sources (post time and temporal tag) help? Does the truth finding module improve the mining process? Does the World's Invariants derived from significance tests help?

Q2. (Interpretability) Do textual patterns have different reliabilities? For one pattern, are its reliabilities the same or different upon different time sources?

4.1 Experimental Settings

Text data description. The dataset has 9,876,086 news articles (4 billion words) published from 1994–2010.

Structured data size and ground truth. We focus on finding precise temporal facts on *country's president*. For *country's president*, we have 53,298 textual patterns of $Country and $Person, 116,631 unique EVT-tuples, and as many as **5,335,344** tuple extractions, where the timestamps are refined to the *year* level. We collected the ground truth from Google and Wikipedia, which contains 365 $(e,v,[t_s,t_e])$-tuples of 130 countries and can then be split into 3,175 (e,v,t)-tuples.

Evaluation methods. We evaluate the performance of our method and all baselines on mining the 3,175 true (e,v,t)-tuples using standard Information Retrieval metrics: *precision*, *recall*, *F1 score*, and *AUC* (Area Under the Curve). Precision is the the fraction of true (e,v,t)-tuples among the (e,v,t)-tuples split from $(e,v,[t_s,t_e])$-tuples. Recall is the fraction of true (e,v,t)-tuples among the (e,v,t)-tuples split from the ground truth. F1 score is the harmonic mean of precision and recall. For all of the metrics, the higher scores indicate that the method has better performance.

Baseline methods and parameter settings. There was no existing work that introduces the idea of truth discovery methodology to the problem of temporal fact extraction from unstructured data. Our work proposes to structure the data with textual patterns and use the World's invariants for truth finding. We compare our method with existing iterative-based (or called propagation-based) truth finding algorithms as well as its multiple variants when given the structures. As shown in Table 3, the methods are as follows.

(1) MajVote-t [11]: it uses the weighted majority voting strategy and returns the most frequent (e,v,t)-tuples;
(2) MajVote-$[t_s,t_e]$: it composites frequent (e,v,t)-tuples into $(e,v,[t_s,t_e])$-tuples, where $t_s = \min t$ and $t_e = \max t$;
(3) TruthFinder-$H_{1e-to-1v}$ [34]: we modify TruthFinder by taking its hypothesis as $H_{1e-to-1v}$, because it assumes "one book only has one author list". Then the patterns and facts of attribute are regarded as the *websites* and *books' author list*, respectively.
(4) TruePIE [17]: it assumes hypothesis $H_{1v-to-1e}$ without considering time-relevant invariants which are very important for temporal fact extraction;
(5) TFWIN and its variants: all TFWIN methods use the valid time-irrelevant hypothesis $H_{1v-to-1e}$ and the valid time-relevant hypothesis $H_{(1t)1e-to-1v}$ that were derived from hypothesis tests. The four variants discuss whether the count of pattern-tuple extraction $c(p,(e,v,t))$ matters in evaluating the pattern reliability $r(p)$ (Eq.(7)) and tuple trustworthiness $w((e,v,t))$ (Eq.(8)).

4.2 Experimental Results

In this section, we present results on both effectiveness and interpretability of the proposed approach.

4.2.1 Effectiveness. Table 3 presents the AUC and F1 of all the methods on finding *country's president* in the "pattern-tuple" structures from text data.

Overall performance. The best baseline method MajVote-$[t_s,t_e]$ gives an AUC of 0.4958 when integrating post time and temporal tag for tuple majority voting, and gives an F1 of 0.6049 on post-time-only tuples. Our TFWIN, which conducts truth finding with two valid hypotheses and source-aware pattern reliability modeling, generates an AUC of 0.6146 (**+24.0%** over the baseline) and an F1 of 0.7572 (**+25.2%**), when $\alpha = 10$. The best of TFWIN ($\alpha = 7$) shows an AUC of 0.6216 (**+25.4%**) and an F1 of 0.7654 (**+26.5%**). Details of different experimental settings will be discussed as follows:

TFWIN vs majority voting. Table 3 shows that MajVote-$[t_s,t_e]$ performs significantly better than MajVote-t (+48.6% AUC; +37.3% F1) because the attribute *president* has $[t_s,t_e]$-period property.

TFWIN vs TruthFinder and TFWIN: comparing with existing iterative-based truth finding methods. TruthFinder [34] assumes one object has only one true fact; however, it doesn't make sense to assume $H_{1e-to-1v}$ (one country has only one president). It is no longer the book's authorlist case. So the AUC and F1 are very poor (< 0.01). We derive two World's invariants, a time-irrelevant constraint $H_{1v-to-1e}$ and a time-relevant constraint $H_{(1t)1e-to-1v}$. TruePIE [17] can be applied to time-irrelevant cases that follow $H_{1v-to-1e}$. The AUC is 0.06 and the F1 is 0.14. The performance is better than TruthFinder, because TruthFinder does not use this correct hypothesis. Our TFWIN uses both valid invariants as well as the time-relevant invariants, and therefore, it outperforms TruthFinder and TruePIE: the AUC becomes 0.62 (10.6× higher) and the F1 is 0.76 (5.4× higher). These demonstrate the power of using the valid World's invariants in true fact discovery.

4.2.2 Interpretability. Table 1 presents the reliability scores of some textual patterns upon the *post time* and *temporal tag* sources. Table 4 shows a subset of temporal facts TFWIN generates.

Pattern reliability modeling. From Table 1, we observe that (1) the top three frequent patterns on *country's president* have good reliability scores upon both time sources; (2) if a pattern has the words indicating past tense like "former", it has good reliability

Table 3: TFWIN wins against truth-discovery baselines on finding true *country's president* from unstructured data.

	Weight in pattern reliability estimation in Eq.(7)	Weight in tuple trustworthiness inference in Eq.(8)	Post time only		Temporal tag only		Post time + Temporal tag			
							Source unaware		Source aware	
			AUC	F1	AUC	F1	AUC	F1	AUC	F1
MajVote-t [11]: return (e, v, t)-tuples			0.3022	0.4101	0.1356	0.2815	0.3336	0.4318	N/A	
MajVote-$[t_s, t_e]$: return $(e, v, [t_s, t_e])$-tuples			0.4202	**0.6049**	0.1670	0.3458	**0.4958**	0.5927		
TruthFinder[34]	$H_{1e-to-1v}$		AUC = 0.0006, F1 = 0.0012							
TruePIE[17]	$H_{1v-to-1e}$		AUC = 0.0587, F1 = 0.1430							
TFWIN	✗: 1	✗: 1	0.4411	0.6140	0.0818	0.1533	0.4403	0.6278	0.4614	0.6313
($H_{1v-to-1e}$,	✔: $c(p, (e, v, t))$	✗: 1	0.4440	0.6144	0.1094	0.1998	0.4209	0.6277	0.4680	0.6404
$H_{(1t)1e-to-1v}$,	✗: 1	✔: $c(p, (e, v, t))$	0.4713	0.6413	0.2335	0.3974	**0.5437**	**0.7242**	0.5822	0.7370
$\alpha = 10$)	✔: $c(p, (e, v, t))$	✔: $c(p, (e, v, t))$	**0.4764**	**0.6460**	0.2699	0.4340	0.4789	0.7065	**0.6146**	**0.7572**
TFWIN ($\alpha = 5$)	✔: $c(p, (e, v, t))$	✔: $c(p, (e, v, t))$	0.4737	0.6459	0.2979	0.4651	0.4405	0.6802	0.6101	0.7591
TFWIN ($\alpha = 7$)	✔: $c(p, (e, v, t))$	✔: $c(p, (e, v, t))$	0.4731	0.6448	0.2955	0.4670	0.4471	0.6829	**0.6216**	**0.7654**

Table 4: Temporal fact $(e, v, [t_s, t_e])$-tuples and the truth.

Attribute value v	Start year t_s (truth $\hat{t}_s$)	End year t_e (truth $\hat{t}_e$)
entity e = "United States", attribute a = "president"		
F. D. Roosevelt	1933	1944
H. S. Truman	1945	**1953** (1952)
D. D. Einsenhower	**1954** (1953)	1960
J. F. Kennedy	1961	**1963** (1962)
L. B. Johnson	**1964** (1963)	1968
R. M. Nixon	1969	**1974** (1973)
G. R. Ford	**1975** (1974)	1976
J. E. Carter	1977	1980
R. W. Reagon	1981	1988
G. H.W. Bush	1989	**1993** (1992)
W. J. Clinton	**1994** (1993)	2000
G. W. Bush	2001	2008
B. H. Obama	2009	2016
entity e = "Burundi", attribute a = "president"		
C. Ntaryamira	1994	1994
S. Ntibantunganya	**1995** (1994)	1996

upon temporal tag but poor reliability upon post time, because the person is absolutely no longer the president at the post time but the temporal tag around the pattern may present the time when the person was on the stage; (3) if a pattern has the words like "now", "newly" or "current", it has good reliability upon post time but poor reliability upon temporal tag; (4) if a pattern has the words of other occupations such as "premier" and "golfer", the reliabilities are negative upon both sources. For (3), note that if a person is newly elected as a president in a year, he/she may still not be in office, so the reliability upon post time is only 0.20.

Results on temporal fact mining. Table 4 shows the names and presidential terms of United States presidents since the year 1933. We observe that the president names v are all correct, and the t_s and t_e of TFWIN's prediction sometimes has only one year difference from the ground truth. The prediction well preserves the valid invariants $H_{1v-to-1e}$ and $H_{(1t)1e-to-1v}$.

5 RELATED WORK

In this section, we review two relevant fields to our work, temporal fact extraction and truth discovery.

5.1 Temporal Fact Extraction

The task is defined as extracting (entity, attribute name, attribute value)-tuples along with their time conditions from text corpora [4, 15, 26, 27, 35]. The concept of fact is broader than the relation between two entities. There are two series of existing natural language processing models: one is based on dependency parsing [5, 8, 21, 24], and the other is based on learning neural networks with human annotations [6, 20, 28]. These models usually work on individual sentences/paragraphs [10, 30], and suffer from high complexity and unavailability of training data [14]. It is important to leverage the data amount and evaluate the trustworthiness of extracted information using the truth finding technology. Fortunately, textual patterns, such as E-A patterns [12], parsing patterns (by PATTY [22]), and meta patterns (by MetaPAD [16]), have been proposed to turn text data into structures in an unsupervised way. However, it was not designed for the problem of temporal fact extraction: it did not consider the two types of temporal contexts. We infer precise temporal slots from post time and time expressions.

5.2 Truth Discovery

The era of big data draws the serious issue of "Veracity" on resolving conflicts among multi-source information [3]. Truth discovery, which integrates multi-source noisy information by estimating the reliability of each source, has emerged as a hot topic [19]. Several truth discovery methods have been proposed for various scenarios, and they have been successfully applied in diverse application domains. TruthFinder proposed the source consistency assumption, iteratively estimated source reliabilities and identified truths [34]. CRH estimated the source reliability on heterogeneous data [18] The evolution of source reliability has been explored in [32]. We propose to apply truth discovery of estimating information extractor reliability to temporal fact extraction.

6 CONCLUSIONS

In this paper, we studied a challenging problem of precise temporal slot filling and point out the limitations of existing OpenIE and PatternIE. We proposed a novel unsupervised, pattern-based, truth finding-driven framework to find precise temporal facts from text data without human annotations. Experiments demonstrated the effectiveness and efficiency.

ACKNOWLEDGEMENTS

This project is supported in part by the National Science Foundation Industry/University Cooperative Research Center on Alternative and Sustainable Intelligent Computing (ASIC) at University of Notre Dame, under grant CNS-1822099.

REFERENCES

[1] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 344–354.

[2] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web.. In *IJCAI*, Vol. 7. 2670–2676.

[3] Laure Berti-Equille. 2015. Data veracity estimation with ensembling truth discovery methods. In *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE, 2628–2636.

[4] Melisachew Wudage Chekol. 2017. Scaling probabilistic temporal query evaluation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 697–706.

[5] Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd annual meeting on association for computational linguistics*. Association for Computational Linguistics, 423.

[6] Dmitriy Dligach, Timothy Miller, Chen Lin, Steven Bethard, and Guergana Savova. 2017. Neural temporal relation extraction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, Vol. 2. 746–751.

[7] Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. 2011. Open information extraction: The second generation.. In *IJCAI*, Vol. 11. 3–10.

[8] Katrin Fundel, Robert Küffner, and Ralf Zimmer. 2006. RelExâĂŤRelation extraction using dependency parse trees. *Bioinformatics* 23, 3 (2006), 365–371.

[9] Alban Galland, Serge Abiteboul, Amélie Marian, and Pierre Senellart. 2010. Corroborating information from disagreeing views. In *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 131–140.

[10] Kiril Gashteovski, Rainer Gemulla, and Luciano Del Corro. 2017. Minie: minimizing facts in open information extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2630–2640.

[11] Sally A Goldman and Manfred K Warmuth. 1995. Learning binary relations using weighted majority voting. *Machine Learning* 20, 3 (1995), 245–271.

[12] Rahul Gupta, Alon Halevy, Xuezhi Wang, Steven Euijong Whang, and Fei Wu. 2014. Biperpedia: An ontology for search applications. *Proceedings of the VLDB Endowment* 7, 7 (2014), 505–516.

[13] Alon Halevy, Natalya Noy, Sunita Sarawagi, Steven Euijong Whang, and Xiao Yu. 2016. Discovering structure in the universe of attribute names. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 939–949.

[14] Julia Hirschberg and Christopher D Manning. 2015. Advances in natural language processing. *Science* 349, 6245 (2015), 261–266.

[15] Tuan-Anh Hoang-Vu, Huy T Vo, and Juliana Freire. 2016. A unified index for spatio-temporal keyword queries. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 135–144.

[16] Meng Jiang, Jingbo Shang, Taylor Cassidy, Xiang Ren, Lance M Kaplan, Timothy P Hanratty, and Jiawei Han. 2017. Metapad: Meta pattern discovery from massive text corpora. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 877–886.

[17] Qi Li, Meng Jiang, Xikun Zhang, Meng Qu, Timothy P Hanratty, Jing Gao, and Jiawei Han. 2018. Truepie: Discovering reliable patterns in pattern-based information extraction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1675–1684.

[18] Qi Li, Yaliang Li, Jing Gao, Bo Zhao, Wei Fan, and Jiawei Han. 2014. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 1187–1198.

[19] Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. 2016. A survey on truth discovery. *ACM Sigkdd Explorations Newsletter* 17, 2 (2016), 1–16.

[20] Chen Lin, Timothy Miller, Dmitriy Dligach, Steven Bethard, and Guergana Savova. 2017. Representations of time expressions for temporal relation extraction with convolutional neural networks. *BioNLP 2017* (2017), 322–327.

[21] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*. Association for Computational Linguistics, 1003–1011.

[22] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. PATTY: a taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 1135–1145.

[23] Nils Reimers, Nazanin Dehghani, and Iryna Gurevych. 2016. Temporal anchoring of events for the timebank corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 2195–2204.

[24] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 74–84.

[25] Michael Schmitz, Robert Bart, Stephen Soderland, Oren Etzioni, and others. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 523–534.

[26] Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *IEEE Transactions on Knowledge and Data Engineering* 30, 10 (2018), 1825–1837.

[27] Avirup Sil and Silviu-Petru Cucerzan. 2014. Towards Temporal Scoping of Relational Facts based on Wikipedia Data. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*. 109–118.

[28] Alejandro Sobrino, Cristina Puente, and José Ángel Olivas. 2017. Mining Temporal Causal Relations in Medical Texts. In *International Joint Conference SOCOâĂŹ17-CISISâĂŹ17-ICEUTEâĂŹ17 León, Spain, September 6–8, 2017, Proceeding*. Springer, 449–460.

[29] Jannik Strötgen and Michael Gertz. 2015. A baseline temporal tagger for all languages. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 541–547.

[30] David Tsurel, Dan Pelleg, Ido Guy, and Dafna Shahaf. 2017. Fun Facts: Automatic Trivia Fact Extraction from Wikipedia. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 345–354.

[31] Houping Xiao, Jing Gao, Qi Li, Fenglong Ma, Lu Su, Yunlong Feng, and Aidong Zhang. 2016. Towards confidence in the truth: A bootstrapping based truth discovery approach. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1935–1944.

[32] Houping Xiao, Yaliang Li, Jing Gao, Fei Wang, Liang Ge, Wei Fan, Long H Vu, and Deepak S Turaga. 2015. Believe it today or tomorrow? detecting untrustworthy information from dynamic multi-source data. In *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, 397–405.

[33] Mohamed Yahya, Steven Whang, Rahul Gupta, and Alon Halevy. 2014. Renoun: Fact extraction for nominal attributes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 325–335.

[34] Xiaoxin Yin, Jiawei Han, and S Yu Philip. 2008. Truth discovery with multiple conflicting information providers on the web. *IEEE Transactions on Knowledge and Data Engineering* 20, 6 (2008), 796–808.

[35] Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian Sadler, Michelle Vanni, and Jiawei Han. 2018. TaxoGen: Constructing Topical Concept Taxonomy by Adaptive Term Embedding and Clustering. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Learning Binary Hash Codes for Fast Anchor Link Retrieval across Networks

Yongqing Wang, Huawei Shen, Jinhua Gao and Xueqi Cheng
CAS Key Laboratory of Network Data Science and Technology,
Institute of Computing Technology, Chinese Academy of Sciences
Beijing 100190, China
{wangyongqing,shenhuawei,gaojinhua,cxq}@ict.ac.cn

ABSTRACT

Users are usually involved in multiple social networks, without explicit *anchor links* that reveal the correspondence among different accounts of the same user across networks. Anchor link prediction aims to identify the hidden anchor links, which is a fundamental problem for user profiling, information cascading, and cross-domain recommendation. Although existing methods perform well in the accuracy of anchor link prediction, the pairwise search manners on inferring anchor links suffer from big challenge when being deployed in practical systems. To combat the challenges, in this paper we propose a novel embedding and matching architecture to directly learn binary hash code for each node. Hash codes offer us an efficient index to filter out the candidate node pairs for anchor link prediction. Extensive experiments on synthetic and real world large-scale datasets demonstrate that our proposed method has high time efficiency without loss of competitive prediction accuracy in anchor link prediction.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; • **Computing methodologies** → **Machine learning**; • **Applied computing** → **Law, social and behavioral sciences**.

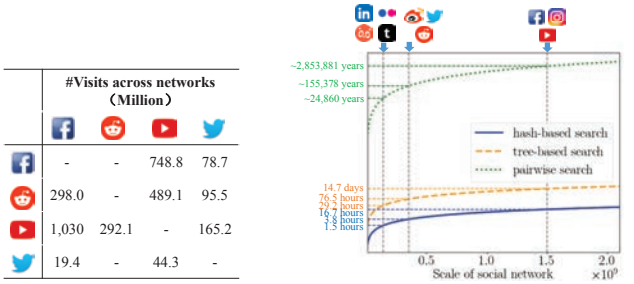
KEYWORDS

hash code, anchor link prediction, scalability

ACM Reference Format:
Yongqing Wang, Huawei Shen, Jinhua Gao and Xueqi Cheng. 2019. Learning Binary Hash Codes for Fast Anchor Link Retrieval across Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313430

1 INTRODUCTION

With the benefit of socialized online services, users are often active across multiple online social networks simultaneously. The integration of user's data from multiple social networks can help to describe comprehensive user profile and interests so as to provide various of applications, including precision marketing and cyber security [3, 25, 29]. Meanwhile, the transferred user data from well established social media sites can mitigate the data sparsity and solve cold start problem for new founded sites [11, 15]. The key of bridging user identities across multiple social networks can be formalized as *anchor link prediction* (ALP) [1, 21, 22, 36], i.e., identifying hidden inter-network links connecting the accounts from the same users across different networks.



(a) Traffics between the most popular social networks (from May 2018 to July 2018).

(b) Time consumption when executing 100 million ALP tasks on different scale of social networks by the different search strategies.

Figure 1: The necessity of anchor link prediction and high efficient solutions.

Most of existing studies in literatures attempt to improve inference accuracy leveraging types of information. The first category of methods relies on the profile information of users by pairwise similarity measures [12, 14, 33, 35]. However, the performance of these methods heavily depends on the availability and quality of user profile [4, 25, 30]. Consequently, these methods are difficult to be generalized to various scenarios [31, 32]. The second category of methods, in contrast, resorts to structural information, anticipating good generalization capability. Typical examples include exploiting local and global consistence pairwisely in topology and structures [7, 8, 19, 23, 24] across networks. Although existing methods perform rather well in the accuracy of anchor link prediction [9, 13, 16, 18, 20, 34], the pairwise search manners on inferring anchor links suffer from big challenge when being deployed in practical systems.

The most prominent challenge for anchor link prediction lies in *scalability*. According to the statistical report on the most popular social media[1], millions of co-visits exist between social media (see Figure 1(a)). However, the time consumption for searching anchor links would be extremely increased by the network size in traditional pairwise manners. As shown in the Figure 3(d), the

[1]https://www.statista.com/

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313430

time cost will be over 2 million years when executing 100 million ALP tasks on Facebook, Instagram or Youtube. Meanwhile, the time consumption is still up to 14.7 days even if applying tree-based search methods.

In this paper, we propose a novel architecture for learning binary hash codes across networks with characteristics in low storage cost and fast retrieval speed by the means of indexing candidate users in anchor link prediction. The goal of our proposed method is to map the structural features from different networks into a Hamming space of binary codes, preserving the similarity between user identities. The compact binary codes are capable of highly efficient retrieval on large-scale datasets [2, 5, 6, 27, 28]. Two anchor-link-aware stages, including embedding and matching, are introduced for efficient usage of labeled data in our proposed learning architecture. Firstly, we conduct network embedding on each network to capture structural features. Embedding method is compatible with learning effective representation of users in low-dimensional space. Besides, the learned embeddings are subject to the explicit constraints on labeled anchor links, eliminating the inference bias by inhomogeneous learning process across networks. In matching stage, the network embeddings from different networks are mapped into a common Hamming space which enables the retrieval process in an indexing way. More specifically, the main contributions of this work are three-folds:

- To solve the challenge in scalability, we propose a novel anchor link modeling method with learning binary hash codes according to network structures. The proposed modeling method is characteristic in low storage cost and fast retrieval speed.
- To solve the inhomogeneity in learned embeddings acr-oss networks, we introduce an anchor-link-aware constraint on network embedding, which can improve the prediction performance in the next matching stage.
- Experiments on time consumption show the potentials of our proposed method in real applications with large-scale networks. Meanwhile, the prediction performance resulted by our proposed method is also competitive to state-of-the-art methods.

2 PROBLEM DEFINITION

In this section, we introduce some basic notations and definitions in anchor link prediction task. A social network can be denoted as $G = \{\mathcal{U}, \mathcal{E}\}$, where $\mathcal{U}$ contains all user identities and $\mathcal{E} \subseteq \mathcal{U} \times \mathcal{U}$ is the set of social relationships in the network. Without loss of generality, we focus on the anchor link prediction on two social networks, denoted as network G^s and network G^t respectively. Note that the settings of anchor link prediction on two networks can be easily extended to multiple social networks. For each user identity in one network, the objective of anchor link prediction is to identify, if any, its counterpart in another network. This task can be formally formulated as follows.

Definition 2.1. **Anchor link prediction:** Given two social networks $G^s = \{\mathcal{U}^s, \mathcal{E}^s\}$ and $G^t = \{\mathcal{U}^t, \mathcal{E}^t\}$, the objective of anchor link prediction is to learn the discriminative function $\mathcal{F}$: $\mathcal{U}^s \times \mathcal{U}^t \to \{0, 1\}$, which estimates whether a pair of user identities from two networks belong to the same nature person, i.e.,

$$\mathcal{F}(u, v) = \begin{cases} 1, \text{if } u \in \mathcal{U}^s \text{ and } v \in \mathcal{U}^t \text{ belong to same person,} \\ 0, otherwise. \end{cases}$$

A labeled set of anchor links, denoted as $\mathcal{A} = \{(u, v) | u \in \mathcal{U}^s, v \in \mathcal{U}^t, \mathcal{F}(u, v) = 1\}$, is also provided as the supervised information for learning function $\mathcal{F}$.

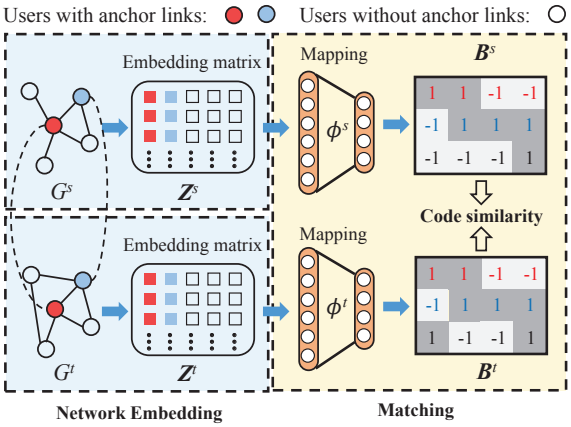


Figure 2: The framework of our proposed HALF model.

We propose a supervised **H**ash-b**A**sed **L**earning **F**ramework for ALP with two anchor-link-aware stages, called HALF. The architecture of proposed model is shown in Figure 2. In this model, the first stage conducts network embedding on each network respectively. The objective of network embedding is to compress the high dimensional sparse network structure (i.e., adjacent matrix) into a low dimensional dense representation, preserving social structures of users. The second stage maps network embedding from each network into a common Hamming space and identifies the anchor links in the mapping space. Formally, the objective function of our proposed model can be presented as

$$\min_{\mathbf{Z}^s, \mathbf{Z}^t, \mathbf{B}^s, \mathbf{B}^t} \mathcal{L} = O\left(G^s, G^t, \mathcal{A} | \mathbf{Z}^s, \mathbf{Z}^t\right) + \mathcal{F}\left(\mathcal{A} | \mathbf{B}^s, \mathbf{B}^t\right), \quad (1)$$

where $\mathbf{Z}^s$ and $\mathbf{Z}^t$ are the embedding matrices of network G^s and G^t respectively. The anchor-link-aware embedding loss $O\left(G^s, G^t, \mathcal{A} | \mathbf{Z}^s, \mathbf{Z}^t\right)$ reflects the capability of generating networks by $\mathbf{Z}^s$, $\mathbf{Z}^t$ under the constraints of $\mathcal{A}$. In the second part of the objective function, the symbol $\mathbf{B}$ refers to $\text{sign}(\Phi(\mathbf{Z}))$ with each element being defined as follows:

$$\mathbf{B} = \text{sign}(\Phi(\mathbf{Z})) = \begin{cases} 1, \Phi(\mathbf{Z}) \geq 0, \\ -1, \Phi(\mathbf{Z}) < 0, \end{cases}$$

where the mapping functions $\Phi : \mathbf{Z} \to \mathbf{R}^K$ try to map each user representation into a data point which can be easily transferred into a binary Hamming space. Note that the inputs and the parameters are independent in $\mathbf{B}^s$ and $\mathbf{B}^t$ respectively, we use the same notation for briefly illustrating the objective function. Therefore, the matching loss $\mathcal{F}\left(\mathcal{A} | \mathbf{B}^s, \mathbf{B}^t\right)$ depicts how well the learned binary hash codes capture the observed anchor links.

Direct optimization on Eq. (1) is quite difficult because of the interdependence between $\mathbf{Z}^s$, $\mathbf{Z}^t$, $\mathbf{B}^s$ and $\mathbf{B}^t$. Therefore, we turn to optimize the objective function by two stages separately, including network embedding and matching. Next we will introduce our proposed model in detail.

3 MODEL

In this section, we will introduce the detail of our proposed model by two anchor-link-aware stages: network embedding and matching.

3.1 Network embedding

Network embedding aims to embed a network into a low dimensional latent space, where each user is represented as a d-dimensional vector. The learned user representations are capable of depicting neighborhood similarity and community membership, which are important features for exploring anchor links. However, under the issues of anchor link prediction, independently learning embeddings for each network may lead to great matching errors when trying to map user representations from independent embedded spaces. Therefore, we introduce an anchor-link-aware network embedding method, exhibiting better performance in anchor link prediction.

Firstly, we introduce two vectors z_i and z'_i for user i, where z_i is user i's representation and z'_i is her *context* representation. For each directed edge $\langle u_i, u_j \rangle$, we define the probability of *context* u_j generated by user u_i as follows,

$$p\left(u_j|u_i\right) = \frac{\exp\left({z'_j}^T \cdot z_i\right)}{\sum_{k=1}^{|\mathcal{U}|} \exp\left({z'_k}^T \cdot z_i\right)}, \tag{2}$$

where $|\mathcal{U}|$ is the number of users in the network. The Eq. (2) captures the similarity between two users who share common neighborhoods. According to Eq. (2), the embedding loss of network G^s and G^t can be respectively formalized as

$$O\left(G^s\right) = -\sum_{\langle u_i^s, u_j^s \rangle \in \mathcal{E}^s} \log p\left(u_j^s|u_i^s\right), \tag{3}$$

and

$$O\left(G^t\right) = -\sum_{\langle u_i^t, u_j^t \rangle \in \mathcal{E}^t} \log p\left(u_j^t|u_i^t\right). \tag{4}$$

For circumventing the performance reduction introduced by independently embedding networks, we introduce constraints to all the nodes in the labeled anchor link set. For each anchor link $(u_i^t, u_j^s) \in \mathcal{A}$, we define the probability of its generation as

$$p\left(u_i^t, u_j^s\right) \sim \frac{z_i^t \cdot z_j^s}{\|z_i^t\| \|z_j^s\|}, \ \textit{iff}\ (u_i^t, u_j^s) \in \mathcal{A}, \tag{5}$$

and the negative logarithmic likelihood of anchor link set $\mathcal{A}$ can be formulated as

$$O\left(\mathcal{A}|G^s, G^t\right) = -\sum_{(u_i^t, u_k^s) \in \mathcal{A}} \log p\left(u_i^t|u_k^s\right) \tag{6}$$

According to Eq. (3), Eq. (4) and Eq. (6), we have the anchor-link-aware embedding loss in the embedding stage, described as

$$\min_{\mathbf{Z}^s, \mathbf{Z}^t} O\left(G^s\right) + O\left(G^t\right) + \gamma_e O\left(\mathcal{A}|G^s, G^t\right), \tag{7}$$

where γ_e is a hyper-parameter.

3.2 Matching

In previous works, finding one user's counterparts in other networks needs to loop over all possible candidates, resulting in huge, matching cost. In this paper, we adopt a learning method to map user representations from the original space into a Hamming space of binary codes. The binary codes can preserve the similarity in the original space and improve retrieval speed in anchor link prediction task. Thus, we introduce an end-to-end deep architecture for efficiently learning binary codes in the matching stage.

Firstly, we learn two mapping functions on each network, transferring the network embeddings into a continuous common space so as to determine whether the user identities are matched. We define $\phi_i^s = \Phi^s(z_i^s)$ and $\phi_j^t = \Phi^t(z_j^t)$, referring to the outputs of mapping function Φ^s and Φ^t with $z_i^s \in \mathbf{Z}^s$ and $z_j^t \in \mathbf{Z}^t$ serving as inputs respectively. The matching problem is formalized as a multi-class classification issue where the inputs are the outputs of mapping functions from two networks. We define the probability of the anchor link with u_i^s and its counterpart u_j^t in two networks as

$$p\left(u_j^t|u_i^s\right) = \frac{\exp\left({\phi_j^t}^T \cdot \phi_i^s\right)}{\sum_{u_k^t \in \mathcal{U}^t} \exp\left({\phi_k^t}^T \cdot \phi_i^s\right)}. \tag{8}$$

According to the probability defined in Eq. (8), the matching loss can be formulated as follows,

$$\mathcal{L}_{match} = -\sum_{(u_i^s, u_j^t) \in \mathcal{A}} \log p\left(u_j^t|u_i^s\right). \tag{9}$$

Then we learn the binary codes based on the continuous common space. We define the loss between binary codes and mapped user representations as

$$\mathcal{L}_{hash} = \|\mathbf{B}^s - \vec{\phi}^s\|_F^2 + \|\mathbf{B}^t - \vec{\phi}^t\|_F^2, \tag{10}$$

where $\vec{\phi}$ is the matrix of mapped user representations where the i-th column corresponds to user u_i. The loss limits the distance between the outputs of mapping functions and corresponding binary codes. The way of relaxation on binary codes avoids discrete learning problem, which may deteriorate the accuracy of the learned binary codes.

Overall, we define the loss of entire matching stage as follows,

$$\begin{aligned}\mathcal{F}\left(\mathcal{A}|\mathbf{B}^s, \mathbf{B}^t\right) =& \mathcal{L}_{match} + \gamma_m \mathcal{L}_{hash} \\ &+ \eta_m \left(\|\vec{\phi}^s\|_F^2 + \|\vec{\phi}^t\|_F^2\right),\end{aligned} \tag{11}$$

where γ_m and η_m are hyper-parameters. The third term $\left(\|\vec{\phi}^s\|_F^2 + \|\vec{\phi}^t\|_F^2\right)$ regularizes the learned mapping results.

After obtaining the binary code of each user in two networks, we can use hashing-based search algorithms, e.g., locality sensitivity hashing, to calculate the similarity between binary codes and choose the most similar ones as the inferring results.

4 OPTIMIZATION

In this section, we develop algorithm to estimate the parameters of the proposed model.

4.1 Optimization on embedding stage

The direct optimization on Eq. (2) is computational expensive, requiring the summation over the entire set of users. To address this problem, we adopt the approach of negative sampling proposed in [17]. We define the negative sampling by the formalization as follows

$$\log \sigma\left(z_j'^{T} \cdot z_i\right) + \sum_{k=1}^{K} E_{u_k \sim P_n(u)}\left[\log \sigma\left(-z_k'^{T} \cdot z_i\right)\right], \qquad (12)$$

which is used to replace every $\log p\left(u_j|u_i\right)$ term in the objective function of the embedding stage. The function $\sigma(x) = 1/\left(1+\exp(-x)\right)$ is the sigmoid function. The negative sampling distinguishs the edge $\langle u_i, u_j\rangle$ from K noise edges $\langle u_i, u_j\rangle$ sampled from noise distribution $P_n(u)$. We set the noise distribution $P_n(u) \propto {d_u}^{3/4}$, where d_u is the out-degree of user u. Besides, we introduce pre-trained process in order to stabilize the training process [10].

4.2 Optimization on matching stage

The optimization on logarithmic Eq. (8) also suffers high computational cost when summarizing compositions on all possible anchor links $\mathcal{A}$. To efficiently estimate the parameters on mapping, we maximize the $\log p\left(u_i^s, u_j^t\right)$ with negative sampling as

$$\log \sigma\left(\phi_j^{t\,T} \cdot \phi_i^s\right) + \sum_{k=1}^{K} E_{u_k \sim P_n'(u)}\left[\log \sigma\left(-\phi_k^{t\,T} \cdot \phi_i^s\right)\right], \qquad (13)$$

where the negative samples are drawn from $P_n'(u) = 1/|\mathcal{U}|$. Meanwhile, we construct a generalized linear function as the mapping functions with shared parameters across networks to handle the overfitting problem caused by scarce labeled anchor links in real applications, that is, $\phi_i = \tanh(w \cdot z_i + bias)$ where w is a parameter matrix.

According to the formalization on matching stage, the best matching performance is achieved when the binary codes from labeled anchor links are set to be the same. Hence we regularize $b_{(i,j)} = b_i^s = b_j^t$, *iff* $(u_i, u_j) \in \mathcal{A}$ in the training process. If we only consider the observed anchor links, the function $\|\mathbf{B}^s - \vec{\phi}^s\|_F^2 + \|\mathbf{B}^t - \vec{\phi}^t\|_F^2$ can be unfolded as

$$\sum_{(u_i,u_j)\in\mathcal{A}} \left(\|b_{(i,j)}\|^2 + \|\vec{\phi}_i^s\|^2 + \|\vec{\phi}_j^t\|^2 - 2b_{(i,j)}^T \cdot \left(\vec{\phi}_i^s + \vec{\phi}_j^t\right)\right).$$

It can be easily derived that the minimization of function $\|\mathbf{B}^s - \vec{\phi}^s\|_F^2 + \|\mathbf{B}^t - \vec{\phi}^t\|_F^2$ is equivalent to maximizing $b_{(i,j)}^T \cdot (\vec{\phi}_i^s + \vec{\phi}_j^t)$, when $\vec{\phi}^s$ and $\vec{\phi}^t$ are fixed. Therefore, we have $b_{(i,j)} = \text{sign}(\vec{\phi}_i^s + \vec{\phi}_j^t)$.

At the end, we can reformulate the matching loss in Eq. (11) as follows,

$$\begin{aligned}\mathcal{F}'\left(\mathcal{A}|\mathbf{B}^s, \mathbf{B}^t\right) = &-\log \sigma\left(\phi_j^{t\,T} \cdot \phi_i^s\right) - \sum_{k=1}^{K}\left[\log \sigma\left(-\phi_k^{t\,T} \cdot \phi_i^s\right)\right] \\ &+ \gamma_m \sum_{(u_i,u_j)\in\mathcal{A}} \left(\|b_{(i,j)} - \vec{\phi}_i^s\|^2 + \|b_{(i,j)} - \vec{\phi}_j^t\|^2\right) \\ &+ \eta_m \left(\|\vec{\phi}^s\|_F^2 + \|\vec{\phi}^t\|_F^2\right).\end{aligned}$$

The HALF algorithm for parameter estimation is described in Algorithm 1. Also we introduce optimization technics for achieving high computational efficiency and prediction accuracy, such as negative sampling and pre-training in embedding stage, parameter sharing in matching stage.

Algorithm 1 HALF

Input: Social network G^s and G^t, labeled anchor links $\mathcal{A}$.
Output: Network embedding $\mathbf{Z}^s$ and $\mathbf{Z}^t$, binary hash code matrices $\mathbf{B}^s$ and $\mathbf{B}^t$.

Initialization: Initialize parameters with random values, including $\mathbf{Z}^s$, $\mathbf{Z}^t$, ϕ^s and ϕ^t.
1. Embedding stage
while the value of objective function do not converge **do**
  Calculate $\partial O\left(G^s\right)/\partial \mathbf{Z}^s$ and $\partial O\left(G^s\right)/\partial \mathbf{Z}'^s$.
  Update $\mathbf{Z}^s$ and $\mathbf{Z}'^s$ with stochastic gradient descent.
end while
while the value of objective function do not converge **do**
  Calculate $\partial\left(O\left(G^t\right) + O\left(\mathcal{A}|G^s, G^t\right)\right)/\partial \mathbf{Z}^t$ and $\partial\left(O\left(G^t\right) + O\left(\mathcal{A}|G^s, G^t\right)\right)/\partial \mathbf{Z}'^t$ with fixed $\mathbf{Z}^s$.
  Update $\mathbf{Z}^t$ and $\mathbf{Z}'^t$ with stochastic gradient descent.
end while
2. Matching stage
while the value of objective function do not converge **do**
  Calculate $\partial\mathcal{F}\left(\mathcal{A}|\mathbf{B}^s, \mathbf{B}^t\right)/\partial\Phi^s$ and $\partial\mathcal{F}\left(\mathcal{A}|\mathbf{B}^s, \mathbf{B}^t\right)/\partial\Phi^t$.
  Update Φ^s and Φ^t with stochastic gradient descent.
end while

5 EXPERIMENT

In experiments, we compare our proposed method to the state-of-the-art methods for anchor link prediction on both synthetic data and real data. The results show that the proposed method achieves better performance in predicting anchor links with higher prediction accuracy and time efficiency.

5.1 Comparative methods

Previous anchor link prediction methods seldom can be well applied to large-scale data, suffering high computational cost in network embedding and matching. To better illustrate the performance of our proposed model, we choose scalable methods as baselines. The compared methods are summarized as follows:

- **FRUI-P** [37]: This work proposes an unsupervised method. We use the method to illustrate the performance when confronting with scarce label data.
- **MNA** [9]: This method models anchor link prediction as a classification problem based on human-craft features.
- **PALE-MLP** [16]: This method is a two-stage model, including separated network embedding and matching.
- **IONE** [13]: This method proposes a joint learning framework on embedding and matching.

As only orientation of vector is kept in Hamming space, hash codes would suffer from loss of information. For better illustrating the prediction accuracy, we introduce the codes from our proposed method without binarization to compare with the baselines.

5.2 Evaluation

We regard the prediction task of finding the matching node in another network as a ranking problem with the similarity score of two nodes' representations being served as ranking scores. Due to

the high matching cost in baselines, 10 random nodes are sampled for each node. These sampled nodes, together with the ground truth matching node, are provided as candidate set. Each model is expected to correctly recognize the ground truth matching node from the candidate set. The prediction performance is evaluate by *Mean Reciprocal Rank* (MRR). Larger MRR values indicate better performance.

5.3 Experiments on synthetic data

The goal of experiments on synthetic data is to validate the effectiveness of our proposed models in prediction accuracy and time complexity.

5.3.1 Dataset. The synthetic data is built from a dataset crawled from Blogcatalog[2], processed by Tang and Liu [26]. The dataset contains 10,312 bloggers and 333,983 social links of bloggers. We follow the sampling strategy mentioned in [16] where $\alpha_s = 0.5$ and $\alpha_c = 0.8$ to sample two sub-networks from Blogcatalog network to serve as source network G^s and target network G^t in anchor link prediction. Every node in one sub-network has a matched node in the other sub-network, marked as anchor link. All labeled anchor links are randomly separated into two datasets by a fixed proportion, denoted as training and test datasets. Next, we conduct experiments with different settings on training data size in order to validate the effectiveness of our proposed model on label sparsity.

5.3.2 The effects of label sparsity. We examine the effects of training data size by varying the size from 5% to 60% of the entire labeled data. The experimental results are shown in Figure 3(a). In the figure, we can see that our proposed method achieves the best performance on MRR than all baselines under different settings of training data size. It is interested that FRUI-P is the second-best solution when merely given 5% of labels for training. It indicates that supervised methods would be invalid under the problem of label sparsity. However, our proposed method is still competitive to unsupervised method with scarce labeled data, showing the better performance with 5% of labeled data for training.

5.4 Experiments on real data

The goal of the experiments on real data is to demonstrate the effectiveness of our proposed models in real applications. The performance is evaluated by prediction accuracy and efficiency in retrieval time.

5.4.1 Dataset. Networks from two popular social networking services in China are adopted as datasets:

- **Douban**[3]: Douban offers services for their users to freely share favorite content through social network, including movies, books, music and other off-line events.
- **Sina Weibo**[4]: Weibo is a very popular microblogging platform in China. The number of registered users has exceeded 300 millions by the end of 2017.

The anchor link set is crawled though Douban data service API. In Douban, users would prefer to post the links to their own Weibo accounts in descriptions. According to this reliable information, we link 15,009 pairs of accounts across the two social networks. Then we extract the social networks relative to the labeled users in each network. The basic statistics of extracted real data is illustrated in Table 1. As described in the table, we can see that only 1.9% of total Weibo users and 0.7% of total Douban users have their anchor links being labeled in our dataset. The scarcely labeled anchor links with huge network size pose a big challenge in both optimization and matching in anchor link prediction. In the following experiments, we conduct two series of experiments, denoting "D → W" where we try to find the connected Weibo account for a Douban account and vise versa for "W → D".

Table 1: Real data description

	#Users	#Links	#Anchor links
Douban	2,046,509	6,493,150	15,009
Weibo	788,524	4,413,187	

5.4.2 The length of binary codes. The length of binary hash codes has great influence on captured structural properties, storage cost and retrieval speed. However, long binary codes may suffer overfitting problem in modeling. Therefore, we conduct experiments with code length varying in 24, 28, 32 and 36 bits respectively, to validate its influence on prediction performance. The 70% of entire labeled anchor links serves as training data. The experimental results are recorded in Figure 3(b). As we can see, the best prediction performance is achieved when the length of binary codes is 32 bits. Short bits of binary codes may cause more conflicts in matching, resulting in the reduction of prediction performance. However, long bits of binary codes may suffer overfitting problem when the labeled anchor links is not enough to learn the matching model. Thus, we set the length of binary codes to be 32 bits in our following experiments.

5.4.3 Evaluation on prediction accuracy. We conduct experiments on 70% of labeled data and examine prediction accuracy of all compared methods on the rest of labeled anchor links. As shown in the Figure 3(c), our proposed method significantly outperform all baselines on both prediction tasks "D → W" and "W → D". The MRR values from our proposed method are 0.32(±0.003) and 0.32(±0.001) on "D → W" and "W → D" respectively, achieving 8% and 10% improvement than the second-best solution.

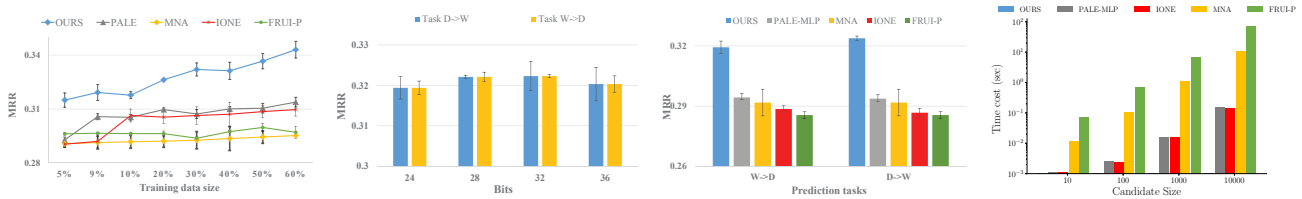
5.5 Time cost on inferring anchor links

With the binary hash codes, we can fast retrieve the potential anchor links by indexing way in our proposed method. The experiments are implemented with Python 2.7 on Intel(R) Xeon(R) CPU ES-2640 @ 2.40GHz with 10 cores and 128GB memory. We examine the time cost when inferring anchor links given 10, 100, 1,000 and 10,000 candidates. The experimental results are shown in Figure 3(d). We can see that our proposed method optimizes the inference time in order of magnitude comparing to other pair-wise solutions.

5.6 Other discussions

In this section, we discuss two remaining issues, i.e, effect of anchor-link-aware constraint formulated in Eq. (6) and the loss of binary codes with consideration of prediction accuracy. The experimental

[2]http://www.blogcatalog.com
[3]http://www.douban.com
[4]http://www.weibo.com



(a) (b) (c) (d)

Figure 3: (a) The experimental results on synthetic data with different training data size; (b) The experimental results on real data with different length of bits; (c) The prediction performance on real data; (d) Query time on 10, 100, 1,000 and 10,000 candidate size.

Table 2: The effect of anchor-link-aware constraint in embedding stage and the loss of binary codes in prediction accuracy.

	Synthetic data	Real data	
		W→D	D→W
Has Eq. (6)	**0.34(±0.005)**	**0.32(±0.003)**	**0.32(±0.001)**
No Eq. (6)	0.29(±0.003)	0.29(±0.005)	0.29(±0.004)
Cont. codes	**0.34(±0.005)**	**0.32(±0.003)**	**0.32(±0.001)**
Binary codes	0.29(±0.004)	0.29(±0.003)	0.29(±0.002)

p.s. Cont. codes refer to the codes without binarization.

results related to the two issues are show in Table 2. We can observe that the prediction performance is boosted with the supplementary of the anchor-link-awared constraint in embedding stage on both synthetic and real data. It means that the representation learning could be inductive to capture common structural features across networks, resulting high prediction accuracy. Besides, although the binary codes would cause loss of prediction accuracy in ALP, the results are still considerable when comparing to the MRR values resulted by PALE, MNA, IONE and FRUI-P methods under the same experimental settings which achieve to 0.31 ± 0.003, 0.29 ± 0.002, 0.31 ± 0.004, 0.30 ± 0.003 and 0.29 ± 0.002 on synthetic data, 0.29 ± 0.007, 0.28 ± 0.002 and 0.28 ± 0.002 on task "W→D", and 0.29 ± 0.007, 0.29 ± 0.002, 0.28 ± 0.002 and 0.28 ± 0.002 on task "D→W" respectively.

6 CONCLUSION

In this paper, we point out one of big challenges for ALP, i.e., scalability when deployed in practical systems. To combat this problem, we propose a novel learning architecture with two stages for learning binary codes across networks. In embedding stage, we embed each user from high-dimensional and sparse raw representaion into a low-dimensional and dense representaion in each network. In matching stage, we propose an efficient matching model for learning binary codes based on learned user representations. The experimental results show the great potential of our proposed model in real applications.

7 ACKNOWLEDGEMENT

This work was funded by the National Natural Science Foundation of China under Grant Numbers 61802371, 61433014, 61425016 and the National Key Research and Development Program of China under grant number 2016QY03D0504. Hua-wei Shen is also funded by K.C Wong Education Foundation and the Youth Innovation Promotion Association of the Chinese Academy of Sciences.

REFERENCES

[1] Yi Cui, Jian Pei, Guanting Tang, Wo Shun Luk, Daxin Jiang, and Ming Hua. 2013. In Finding email correspondents in online social networks. *World Wide Web* 16, 2, 195–218.
[2] Bo Dai, Ruiqi Guo, Sanjiv Kumar, Niao He, and Le Song. 2017. Stochastic Generative Hashing. In *Proceedings of the 34th International Conference on Machine Learning*, Vol. 70. 913–922.
[3] Ruben Enikolopov, Maria Petrova, and Konstantin Sonin. 2018. Social media and corruption. *American Economic Journal: Applied Economics* 10, 1 (2018), 150–74.
[4] Oana Goga, Patrick Loiseau, Robin Sommer, Renata Teixeira, and Krishna P Gummadi. 2015. On the reliability of profile matching across large online social networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1799–1808.
[5] Qing Yuan Jiang and Wu Jun Li. 2017. Deep cross-modal hashing. In *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition*. 3270–3278. arXiv:1602.02255
[6] Qing-Yuan Jiang and Wu-Jun Li. 2018. Asymmetric Deep Supervised Hashing. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*.
[7] Gunnar W. Klau. 2009. A new graph-based method for pairwise global network alignment. *BMC Bioinformatics* 10, SUPPL. 1 (2009).
[8] Giorgos Kollias, Shahin Mohammadi, and Ananth Grama. 2012. Network Similarity Decomposition (NSD): A Fast and Scalable Approach to Network Alignment. *IEEE Transactions on Knowledge and Data Engineering* 24, 12 (2012), 2232–2243.
[9] Xiangnan Kong, Jiawei Zhang, and Philip S. Yu. 2013. Inferring anchor links across multiple heterogeneous social networks. In *Proceedings of the 22nd ACM International Conference on Conference on Information and Knowledge Management*. 179–188.
[10] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521 (2015), 436–444.
[11] Chung-Yi Li and Shou-De Lin. 2014. Matching users and items across domains to improve the recommendation quality. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 801–810.
[12] Jing Liu, Fan Zhang, Xinying Song, Young-In Song, Chin-Yew Lin, and Hsiao-Wuen Hon. 2013. WhatâĂŹs in a Name? An Unsupervised Approach to Link Users across Communities. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining*. 495.
[13] Li Liu, William K Cheung, Xin Li, and Lejian Liao. 2016. Aligning Users Across Social Networks Using Network Embedding. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 1774–1780.
[14] Siyuan Liu, Shuhui Wang, Feida Zhu, Jinbo Zhang, and Ramayya Krishnan. 2014. HYDRA: Large-scale Social Identity Linkage via Heterogeneous Behavior Modeling. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA, 51–62.
[15] Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-domain recommendation: an embedding and mapping approach. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2464–2470.
[16] Tong Man, Huawei Shen, Shenghua Liu, Xiaolong Jin, and Xueqi Cheng. 2016. Predict anchor links across social networks via an embedding approach. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 1823–1829.

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*. 3111–3119.

[18] Xin Mu, Feida Zhu, Ee-Peng Lim, Jing Xiao, Jianzong Wang, and Zhi-Hua Zhou. 2016. User identity linkage by latent user space modelling. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1775–1784.

[19] Manikandan Narayanan and Richard M. Karp. 2007. Comparing Protein Interaction Networks via a Graph Match-and-Split Algorithm. *Journal of Computational Biology* 14, 7 (2007), 892–907.

[20] Yuanping Nie, Yan Jia, Shudong Li, Xiang Zhu, Aiping Li, and Bin Zhou. 2016. Identifying users across social networks based on dynamic core interests. *Neurocomputing* 210 (2016), 107–115.

[21] C Shi, Y Li, J Zhang, Y Sun, and P S Yu. 2017. A Survey of Heterogeneous Information Network Analysis. *IEEE Transactions on Knowledge and Data Engineering* 29, 1 (2017), 17–37.

[22] Kai Shu, Suhang Wang, Jiliang Tang, Reza Zafarani, and Huan Liu. 2017. User Identity Linkage Across Online Social Networks: A Review. *ACM SIGKDD Explorations Newsletter* 18, 2 (2017), 5–17.

[23] Rohit Singh, Jinbo Xu, and Bonnie Berger. 2007. Pairwise global alignment of protein interaction networks by matching neighborhood topology. In *Research in Computational Molecular Biology*. Springer, 16–31.

[24] Rohit Singh, Jinbo Xu, and Bonnie Berger. 2008. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences* 105, 35 (2008), 12763–12768.

[25] Jiliang Tang, Yi Chang, and Huan Liu. 2014. Mining social media with social theories: a survey. *ACM SIGKDD Explorations Newsletter* 15, 2 (2014), 20–29.

[26] Lei Tang and Huan Liu. 2009. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining*. 817–825.

[27] Daixin Wang, Peng Cui, Mingdong Ou, and Wenwu Zhu. 2015. Deep multimodal hashing with orthogonal regularization. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*. 2291–2297.

[28] Dan Wang, Heyan Huang, Chi Lu, Bo-Si Feng, Liqiang Nie, Guihua Wen, and Xian-Ling Mao. 2018. Supervised Deep Hashing for Hierarchical Labeled Data. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*.

[29] Yongqing Wang, Huawei Shen, Shenghua Liu, Jinhua Gao, and Xueqi Cheng. 2017. Cascade dynamics modeling with attention-based recurrent neural network. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2985–2991.

[30] Reza Zafarani and Huan Liu. 2009. Connecting Corresponding Identities across Communities.. In *Proceedings of the 3rd International Conference on Weblogs and Social Media*, Vol. 9. 354–357.

[31] Reza Zafarani and Huan Liu. 2014. Users Joining Multiple Sites : Distributions and Patterns User Membership Distribution across Sites. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*. 635–638.

[32] Reza Zafarani and Huan Liu. 2016. Users joining multiple sites: Friendship and popularity variations across sites. *Information Fusion* 28 (2016), 83–89.

[33] Reza Zafarani, Lei Tang, and Huan Liu. 2015. User Identification Across Social Media. *ACM Transactions on Knowledge Discovery from Data* 10, 2 (2015), 16.

[34] Jiawei Zhang, Jianhui Chen, Shi Zhi, Yi Chang, S Yu Philip, and Jiawei Han. 2017. Link prediction across aligned networks with sparse and low rank matrix estimation. In *Proceedings of the 33rd IEEE International Conference on Data Engineering*. IEEE, 971–982.

[35] Jiawei Zhang, Xiangnan Kong, and Philip S. Yu. 2014. Transferring heterogeneous links across location-based social networks. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*. 303–312.

[36] Jiawei Zhang and Philip S. Yu. 2015. Integrated anchor and social link predictions across social networks. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*. 2125–2132.

[37] Xiaoping Zhou, Xun Liang, Xiaoyong Du, and Jichao Zhao. 2017. Structure Based User Identification across Social Networks. *IEEE Transactions on Knowledge and Data Engineering* (2017).

Neural Chinese Named Entity Recognition via CNN-LSTM-CRF and Joint Training with Word Segmentation

Fangzhao Wu
Microsoft Research Asia
Beijing, China
wufangzhao@gmail.com

Junxin Liu
Tsinghua University
Beijing, China
ljx16@mails.tsinghua.edu.cn

Chuhan Wu
Tsinghua University
Beijing, China
wuch15@mails.tsinghua.edu.cn

Yongfeng Huang
Tsinghua University
Beijing, China
yfhuang@tsinghua.edu.cn

Xing Xie
Microsoft Research Asia
Beijing, China
xingx@microsoft.com

ABSTRACT

Chinese named entity recognition (CNER) is an important task in Chinese natural language processing field. However, CNER is very challenging since Chinese entity names are highly context-dependent. In addition, Chinese texts lack delimiters to separate words, making it difficult to identify the boundary of entities. Besides, the training data for CNER in many domains is usually insufficient, and annotating enough training data for CNER is very expensive and time-consuming. In this paper, we propose a neural approach for CNER. First, we introduce a CNN-LSTM-CRF neural architecture to capture both local and long-distance contexts for CNER. Second, we propose a unified framework to jointly train CNER and word segmentation models in order to enhance the ability of CNER model in identifying entity boundaries. Third, we introduce an automatic method to generate pseudo labeled samples from existing labeled data which can enrich the training data. Experiments on two benchmark datasets show that our approach can effectively improve the performance of Chinese named entity recognition, especially when training data is insufficient.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Information extraction**; *Neural networks.*

KEYWORDS

Named Entity Recognition; Word Segmentation; Neural Network

ACM Reference Format:
Fangzhao Wu, Junxin Liu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2019. Neural Chinese Named Entity Recognition via CNN-LSTM-CRF and Joint Training with Word Segmentation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313743

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313743

1 INTRODUCTION

The goal of named entity recognition (NER) is to identify entity names from texts and classify their types into different categories such as person, location and so on [9, 26]. For example, given a sentence "Bill Gates is the founder of Microsoft", a NER tool can recognize that "Bill Gates" is a named entity of person and "Microsoft" is a named entity of organization. NER is an important task in natural language processing field [25, 28], and an essential step for many downstream applications such as entity linking [11], relation extraction [20] and question answering [8].

NER has been studied for many years and various methods have been proposed [16, 25]. Most of these methods are designed for NER of English texts. For example, Settles [29] proposed to use conditional random fields (CRF) for English medical text NER and incorporated both orthographic features and semantic features. Huang et al. [14] proposed a LSTM-CRF neural model for English NER. They combined spelling features (e.g., whether a word starts with a capital letter), context features (unigrams and bigrams), and word embeddings to build features for words. Chiu and Nichols [3] proposed to learn word- and character-level features using LSTM and CNN networks for English NER. In their method, the word features include word embeddings, character embeddings learned from characters using CNN network, and capitalization features.

Compared with NER of English texts, Chinese NER is more difficult [6, 19]. First, Chinese texts lack strong indications of entity names existing in English texts such as capitalization [27]. Second, Chinese entity names are highly context-dependent. Almost every Chinese character and word can be used in Chinese entity names. The same word can be used as entity names or non-entity words in different contexts. For example, in the sentence "文献是朝阳区区长", "文献" is a named entity of person. However, in most cases the word "文献" is used as a non-entity word in Chinese texts with the meaning of "literature". In addition, the same word can be used as names of different kind entities. For instance, the word "阿里" can be the name of person entity (e.g., "拳王阿里是个传奇"), location entity ("西藏阿里美不胜收"), and organization entity ("杭州阿里吸引了很多人才"). Third, different from English texts, there is no delimiter such as whitespace to separate words in Chinese texts, making it more difficult to identify entity boundaries [24]. For example, a Chinese NLP tool may segment the sentence "习近平常与特朗普通电话" into "习近/平常/与/特朗/普通/电话" and

predicts that "特朗" is a person name, while in fact "习近平" and "特朗普" are correct named entities of person. Besides, annotating sufficient data for training CNER models is very expensive and time-consuming, and the training data in many domains is limited. Thus, CNER is quite challenging.

In order to handle these challenges, in this paper we propose a neural approach for Chinese named entity recognition. First, in our approach we propose a CNN-LSTM-CRF architecture to fully capture both local and long-distance contexts for CNER. Our model contains three layers, i.e., a convolutional neural network (CNN) [18] layer to learn contextual character representations from local contexts, a bidirectional long-short term memory (Bi-LSTM) [13] layer to learn contextual character representations from long-distance contexts, and a CRF layer to jointly decode character labels. In addition, we propose a unified framework to jointly train CNER and Chinese word segmentation (CWS) models in order to improve the ability of CNER model in predicting entity boundaries. In our framework, the CWS model shares the same character embeddings and CNN network with the CNER model, and has an independent CRF layer for label decoding. Besides, we propose an automatic method to generate pseudo labeled samples based on existing labeled data by randomly replacing entity names in labeled sentences with other entity names of the same concept. These pseudo labeled samples are usually correct in grammar and smooth in semantics, and can enrich the training data and improve the generalization ability of CNER model. Extensive experiments are conducted on two benchmark datasets. The experimental results show that our approach can significantly improve the performance of Chinese named entity recognition and outperform many baseline methods.

2 RELATED WORK

Chinese named entity recognition is usually modeled as a character-level sequence labeling problem [10, 22], since Chinese sentence is a string of characters and there is no explicit delimiter like whitespace to separate characters into words. Statistical sequence modeling methods such as CRF [15] are widely used in Chinese NER [24, 27]. A core step in these methods is building feature representation for each character in sentence. Traditionally, these character features are manually designed [1, 32]. For example, character unigrams, bigrams and clusters are used as character features in [1]. Zhang et al. [33] proposed to build character features using surrounding characters and whether target character is in external dictionaries. Yu et al. [32] proposed to combine character-level features, word-level features, POS tagging features and dictionary features for CNER. These handcrafted features require a large amount of domain knowledge to design and many of them rely on external resources such as gazetteers, which may not exist in many domains. Moreover, handcrafted features such as character n-grams usually cannot capture global context information. Different from above methods, our approach can learn character feature representations from data without feature engineering, and can incorporate both local and long-distance context information.

In recent years, neural network methods have been applied to English NER [14, 16]. Most of these methods are based on LSTM-CRF architecture. For example, Lample et al. [16] used Bi-LSTM to learn hidden representations of words and used CRF to do label decoding. These methods for English NER are usually based on word-level input, and cannot be directly applied to Chinese NER which is a character-level sequence labeling task. CNN is exploited in [3] and [25] to learn word representations from characters for English NER. These word representations are combined with word embeddings and/or other features to build word features. However, there is no explicit word boundaries in Chinese texts. Thus, these CNN incorporated methods for English NER are not suitable for Chinese NER. Different from these methods, in our approach CNN is used to learn representations of characters rather than words.

Recently, neural network methods are also applied to Chinese NER [6, 7, 12]. These methods are based on LSTM-CRF framework, where LSTM is used to learn hidden representations of characters and CRF is used for joint labeling decoding. Many of these methods also incorporate handcrafted features, such as radical features [7] and position features [12], to build character representations. Different from these methods, our approach can train CNER model in an end-to-end manner and does not need any manual feature engineering. In addition, our approach is based on the CNN-LSTM-CRF architecture to fully capture both local and long-distance contexts.

Jointly training NER model with related tasks has the potential to improve the performance of NER [23, 28]. For example, Luo et al. [23] proposed to jointly train NER and entity linking models, and achieved state-of-the-art performance on English NER. Since Chinese texts lack word delimiters and identifying entity boundaries is very challenging in CNER, we propose to jointly train CNER model with word segmentation to improve the ability of CNER model in identifying entity boundaries. Similar with our approach, Peng et al. [28] proposed to jointly train CNER and CWS models in domain adaptation scenario. In their method, CNER and CWS share the same character representation model and have different label decoding models. In our approach we regard word segmentation as an auxiliary task and use it to learn word boundary aware character representations for CNER. Experiments show that our approach is more effective. In addition, we propose to automatically generate pseudo labeled samples from labeled data, which to our best knowledge has not been explored in existing CNER methods.

3 OUR APPROACH

In this section we introduce our approach for CNER in detail. First, we present the CNN-LSTM-CRF neural architecture for CNER. Next, we introduce a unified framework to jointly train CNER and word segmentation models. Finally, we introduce an automatic method to generate pseudo labeled samples from existing labeled data.

3.1 CNN-LSTM-CRF Architecture for CNER

Our proposed CNN-LSTM-CRF neural architecture for Chinese named entity recognition is illustrated in Fig. 1. Next we introduce each layer from bottom to top in detail.

The first layer is character embedding, which aims to convert a sentence from a sequence of characters into a sequence of dense vectors. In this layer, an embedding matrix $\mathbf{E} \in \mathcal{R}^{D \times V}$ is used to map each character into a dense vector, where D is embedding dimension and V is vocabulary size. Denote an input sentence as $s = [w_1, w_2, ..., w_N]$, where N is sentence length and $w_i \in \mathcal{R}^V$ is the

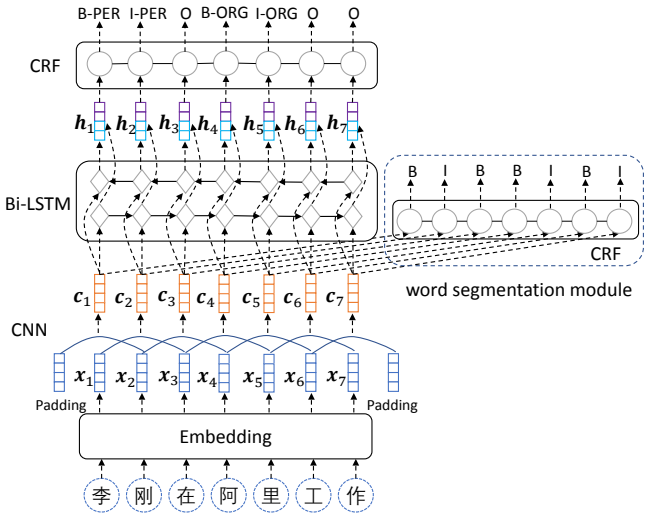


Figure 1: The framework of our approach.

one-hot representation of the i_{th} character. The output of this layer is a sequence of vectors $[\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N]$, where $\mathbf{x}_i = \mathbf{E}w_i \in \mathcal{R}^D$.

The second layer is a CNN network. CNN has been widely used in computer vision field to extract local information of images [17]. The local context of sentences is also very important for Chinese named entity recognition. For example, the word "阿里" is a location entity in "西藏阿里", while is a organization entity in "杭州阿里". Motivated by these observations, we propose to use CNN to capture the local context information for CNER. Denote $\mathbf{w} \in \mathcal{R}^{KD}$ as a filter in CNN where K is the window size, then the contextual representation of the i_{th} character learned by this filter is:

$$c_i = f(\mathbf{w}^T \times \mathbf{x}_{\lfloor i-\frac{K-1}{2} \rfloor : \lfloor i+\frac{K-1}{2} \rfloor}), \tag{1}$$

where $\mathbf{x}_{\lfloor i-\frac{K-1}{2} \rfloor : \lfloor i+\frac{K-1}{2} \rfloor}$ represents the concatenation of the embeddings of characters from $\lfloor i-\frac{K-1}{2} \rfloor$ to $\lfloor i+\frac{K-1}{2} \rfloor$, and f is the activation function. Here we select ReLU [17] as the activation function. In this layer, we use multiple filters with different window sizes (ranging from 2 to 5) to learn contextual character representations. Denote the filter number as M. The contextual representation of the i_{th} character (denoted as $\mathbf{c}_i$) is the concatenation of the outputs of all filters at this position. The output of the CNN layer is $\mathbf{c} = [\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N]$, where $\mathbf{c}_i \in \mathcal{R}^M$.

The third layer is a Bi-LSTM network. LSTM is a special type of recurrent neural networks (RNN) which can capture long-distance sequence information and is powerful in modeling sequential data. LSTM can generate hidden states for tokens in a sequence using all previous contexts, which is beneficial for Chinese named entity recognition. For example, in sentence "阿里是中国非常著名的IT企业" (Alibaba is a famous Chinese IT company), although "IT企业" has a long distance with "阿里", it is very important for inferring that "阿里" is an organization entity. In addition, both left and right contexts can be useful for recognizing named entities. Thus, we use bidirectional LSTM (Bi-LSTM) to learn hidden representations of characters from global contexts. Denote $\overrightarrow{lstm}$ as the LSTM scanning text from left to right, and $\overleftarrow{lstm}$ as the LSTM scanning text from right to left, then the hidden representations learned by these LSTMs can be represented as $[\overrightarrow{\mathbf{h}_1}, \overrightarrow{\mathbf{h}_2}, ..., \overrightarrow{\mathbf{h}_N}] = \overrightarrow{lstm}([\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N])$ and $[\overleftarrow{\mathbf{h}_1}, \overleftarrow{\mathbf{h}_2}, ..., \overleftarrow{\mathbf{h}_N}] = \overleftarrow{lstm}([\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N])$. The hidden representation of the i_{th} character is a concatenation of $\overrightarrow{\mathbf{h}_i}$ and $\overleftarrow{\mathbf{h}_i}$, i.e., $\mathbf{h}_i = [\overrightarrow{\mathbf{h}_i}, \overleftarrow{\mathbf{h}_i}]$. The output of the Bi-LSTM layer is $\mathbf{h} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_N]$, where $\mathbf{h}_i \in \mathcal{R}^{2S}$ and S is the dimension of hidden states in LSTM.

The fourth layer in our model is conditional random field (CRF). In NER, neighboring labels usually have strong dependencies [25]. For example, *I-PER* label usually follows *B-PER* and *I-PER*, but it cannot follow *B-ORG* or *I-ORG*. Thus, it is beneficial to jointly decode the labels of characters in a sequence rather than decode them independently [4]. In this paper, we use first-order linear chain CRF [15] to jointly decode the labels of characters.

Denote $\mathbf{y} = [y_1, y_2, ..., y_N]$ as the label sequence of sentence s, where $y_i \in \mathcal{R}^L$ is the one-hot representation of the i_{th} character's label, and L is the number of labels. The input of the CRF layer is the hidden representations of characters generated by the Bi-LSTM layer, i.e., $\mathbf{h} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_N]$, and the output of the CRF layer is the label sequence $\mathbf{y}$. CRF is a probabilistic model. The conditional probability of label sequence $\mathbf{y}$ given input $\mathbf{h}$ is computed as follows:

$$p(\mathbf{y}|\mathbf{h};\theta) = \frac{\prod_{i=1}^{N} \psi(\mathbf{h}_i, y_i, y_{i-1})}{\sum_{\mathbf{y}' \in \mathcal{Y}(s)} \prod_{i=1}^{N} \psi(\mathbf{h}_i, y'_i, y'_{i-1})}, \tag{2}$$

where $\mathcal{Y}(s)$ is the set of all possible label sequences of sentence s, θ is the parameter set, and $\psi(\mathbf{h}_i, y_i, y_{i-1})$ is the potential function. In our approach, the potential function is:

$$\psi(\mathbf{h}_i, y_i, y_{i-1}) = \exp(y_i^T \mathbf{W}^T \mathbf{h}_i + y_{i-1}^T \mathbf{T} y_i), \tag{3}$$

where $\mathbf{W} \in \mathcal{R}^{2S \times L}$ and $\mathbf{T} \in \mathcal{R}^{L \times L}$ are parameters of the CRF layer, and $\theta = \{\mathbf{W}, \mathbf{T}\}$ in Eq. (2).

The loss function of our approach is the negative log-likelihood over all training samples, which can be formulated as follows:

$$\mathcal{L}_{NER} = -\sum_{s \in \mathcal{S}} \log(p(\mathbf{y}_s|\mathbf{h}_s;\theta)), \tag{4}$$

where $\mathcal{S}$ is the set of sentences in training data, $\mathbf{h}_s$ and $\mathbf{y}_s$ are the hidden representations and label sequence of sentence s.

3.2 Joint Training with Word Segmentation

The NER task can be regarded as a combination of two sub-tasks: extracting entity names from texts (i.e., identifying the boundaries of entity names) and classifying their types. Identifying the boundaries of Chinese entity names is quite challenging, since there is no explicit word delimiters in Chinese texts. For example, many existing Chinese NLP tools such as LTP predict that "特朗" is an entity name in the sentence "习近平常与特朗普通电话", while the correct entity name is "特朗普".

In Chinese natural language processing field, the goal of Chinese word segmentation (CWS) is segmenting Chinese texts into words (in other words, predicting the boundaries of words in texts). Thus, CWS is highly related to CNER and has the potential to help CNER predict entity boundaries more accurately. Motivated

by this observation, we propose to enhance the ability of CNER model in identifying entity boundaries by jointly training CNER and CWS models. Similar with CNER, CWS is also usually modeled as a character-level sequence labeling problem [2, 10, 21], and CRF is widely used in CWS methods. Our unified framework for jointly training CNER with CWS is shown in Fig. 1. In our framework the CNER and CWS models share the same character embeddings and CNN network. In this way, the useful information in word segmentation can be encoded to learn word boundary aware contextual character representations, which is useful for predicting entity boundaries. The loss function of the CWS module is:

$$\mathcal{L}_{CWS} = -\sum_{s \in \mathcal{S}} \log(p(\mathbf{y}_s^{seg} | \mathbf{c}_s; \theta^{seg})), \tag{5}$$

where $\mathbf{y}_s^{seg}$ is the label sequence of sentence s for word segmentation, and θ^{seg} is the parameter set of CWS model. $\mathbf{c}_s$ is the hidden character representations of sentence s output by the CNN network.

The final objective function of our approach is the combination of the CNER loss and the CWS loss as follows:

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_{NER} + \lambda \mathcal{L}_{CWS}, \tag{6}$$

where $\lambda \in [0, 1)$ is a coefficient to control the relative importance of the CWS loss in the overall loss.

3.3 Pseudo Labeled Data Generation

The labeled data for Chinese named entity recognition is usually scarce in many domains. Manually annotating sufficient samples for CNER is time-consuming and expensive. Thus, an automatic method to generate labeled samples for CNER will be very useful. In this paper we propose to automatically generate pseudo labeled samples based on existing labeled data. Our method is motivated by the observation that if an entity name in a sentence is replaced by another entity name with the same concept, then the new sentence is usually correct in grammar and semantics. For example, a Chinese sentence may be "李刚在阿里工作", where "李刚" is a person entity and "阿里" is a company entity. If "王小超" is another person name and "谷歌" is another company name, then we can obtain a new sentence by replacing "李刚" with "王小超" and "阿里" with "谷歌", i.e., "王小超在谷歌工作". If we have the NER labels of the original sentence, then we can easily infer the NER labels of the pseudo sentence. For instance, if the NER labels of the original sentence in aforementioned example are "B-PER/I-PER/O/B-ORG/I-ORG/O/O", then the NER labels of the pseudo sentence are "B-PER/I-PER/I-PER/O/B-ORG/I-ORG/O/O". In addition, if we have CWS labels of the original sentence, then we can also automatically obtain CWS labels of the pseudo sentence, since entity names are usually regarded as independent words. For instance, in aforementioned example, if the CWS labels of the original sentence are "B/I/B/B/I/B/B/I", then the CWS labels of the pseudo sentence are "B/I/I/B/B/I/B/B/I".

In our method, given a set of labeled samples for CNER, we first extract all the entity names (denoted as *EN*) from them. Then we randomly select a labeled sentence, and replace each entity in this sentence with an entity randomly sampled from *EN* which has the same concept. In this way a pseudo sentence is generated and its NER labels (as well as CWS labels) can be automatically inferred. This step is repeated multiple times until a predefined number of pseudo labeled samples are generated.

4 EXPERIMENTS

4.1 Datasets and Experimental Settings

Two benchmark datasets for Chinese named entity recognition are used in our experiments. The first one is the MSRA corpus released by the third SIGHAN Chinese language processing bakeoff[1] (denoted as *Bakeoff-3*). It contains 46,364 sentences for training and 4,365 sentences for test. The second dataset is the MSRA corpus released by the fourth SIGHAN Chinese language processing bakeoff[2] (denoted as *Bakeoff-4*). This dataset contains 23,181 training sentences and 4,636 test sentences. Both datasets contain three types of named entities, i.e., *person* (denoted as *PER*), *location* (denoted as *LOC*) and *organization* (denoted as *ORG*). The statistics of these datasets are summarized in Table 1.

Table 1: The statistics of datasets. *#Sen* and *#Ent* represent the numbers of sentences and entities.

Dataset	*Training*		*Test*		
	#Sen	*#Ent*	*#Sen*	*#Ent*	*OOV Rate*
Bakeoff-3	46,364	74,528	4,365	6,190	28.16%
Bakeoff-4	23,181	37,811	4,636	7,707	21.42%

Following [31], we use *BIO* tagging scheme for CNER, where *B*, *I* and *O* respectively represent the beginning, inside and outside of entity name. The labels *B* and *I* are further combined with entity types, such as "B-PER" and "I-LOC". As for CWS, we use the *BI* tagging scheme. In the *Bakeoff-3* dataset the annotation of word segmentation in training set is provided, while in the *Bakeoff-4* dataset it is not provided. Thus, we used LTP[3] tool to segment the sentences in the training set of the *Bakeoff-4* dataset. Note that the word segmentation information is not involved in the test data.

In our experiments, the embeddings of characters are pretrained on the Sogou News Corpus[4] using the word2vec tool[5]. The dimension of character embedding is 200. These embeddings are tuned during model training. Since dropout method [30] is an effective way to mitigate overfitting, we apply it to the outputs of embedding, CNN and LSTM layers. Our approach is implemented using Tensorflow library, and RMSProp [5] is used as the optimization algorithm to learn model parameters. Hyperparameters are selected according to cross-validation on training data. In detail, the number of filters in CNN network is 400, the hidden state size of LSTM is 200, the dropout rate is 0.2, and the λ in Eq. (6) is 0.4. In each experiment, we randomly sample 10% of the training data for validation and the remaining for model training. Each experiment is repeated 10 times, and the average results are reported.

[1] http://sighan.cs.uchicago.edu/bakeoff2006/download.html
[2] https://www.aclweb.org/mirror/ijcnlp08/sighan6/chinesebakeoff.htm
[3] https://www.ltp-cloud.com/intro/en/
[4] http://www.sogou.com/labs/resource/ca.php
[5] https://code.google.com/archive/p/word2vec/

Table 2: The performance of different methods on the *Bakeoff-3* dataset. P, R and F represent precision, recall and Fscore respectively. R_{oov} stands for the recall of out-of-vocabulary entities.

	5%				25%				100%			
	P	R	F	R_{oov}	P	R	F	R_{oov}	P	R	F	R_{oov}
LSTM+Softmax	56.14	49.83	52.68	33.03	73.02	66.94	69.84	49.00	82.76	78.52	80.59	60.57
LSTM+CRF	61.00	54.32	57.37	37.09	77.75	70.42	73.90	53.05	87.30	82.93	85.06	68.06
MDA	65.12	57.90	61.27	42.74	79.71	73.16	76.29	58.34	87.78	82.43	85.02	67.15
LSTM+CRF+Pseudo	61.21	57.25	59.07	40.22	79.57	75.80	77.63	60.45	88.13	85.74	86.92	73.19
CNN+LSTM+Softmax	62.72	56.96	59.62	39.75	77.83	74.54	76.14	57.15	87.60	83.50	85.50	67.41
CLC	65.58	61.60	63.45	43.04	81.22	76.20	78.59	59.23	90.39	86.43	88.37	71.54
CLC+Joint	70.70	62.93	66.56	45.78	81.20	78.32	79.72	63.13	90.29	86.98	88.60	73.61
CLC+Pseudo	69.86	64.13	66.84	49.17	82.69	79.93	81.27	65.38	89.59	88.46	89.02	76.99
CLC+Joint+Pseudo	70.31	66.51	68.33	49.30	83.57	81.62	82.57	66.86	89.48	89.36	89.42	77.93

Table 3: The performance of different methods on the *Bakeoff-4* dataset.

	5%				25%				100%			
	P	R	F	R_{oov}	P	R	F	R_{oov}	P	R	F	R_{oov}
LSTM+Softmax	51.78	42.88	46.74	30.24	69.32	67.16	68.18	49.62	83.39	80.31	81.81	63.08
LSTM+CRF	54.18	47.53	50.48	30.64	74.90	71.64	73.21	56.07	87.10	85.07	86.07	69.96
MDA	56.66	49.75	52.90	34.93	76.29	73.91	75.05	59.09	87.56	85.01	86.26	70.77
LSTM+CRF+Pseudo	57.37	49.54	53.09	34.53	76.31	75.79	76.02	59.85	86.94	86.90	86.92	72.07
CNN+LSTM+Softmax	59.09	51.12	54.74	35.81	76.42	73.74	74.99	58.13	88.48	86.34	87.40	69.47
CLC	58.42	53.56	55.83	39.41	78.84	78.50	78.59	61.82	89.86	88.74	89.30	73.97
CLC+Joint	61.88	56.23	58.76	41.49	82.87	78.63	80.65	63.43	90.23	88.80	89.51	74.86
CLC+Pseudo	61.82	59.69	60.67	43.40	82.33	80.41	81.35	64.00	90.44	89.80	90.12	75.00
CLC+Joint+Pseudo	64.21	61.58	62.78	47.58	83.16	81.93	82.52	66.88	90.09	90.28	90.18	77.13

4.2 Performance Evaluation

First we compare the performance of our approach with different neural network based CNER methods, including: 1) *LSTM+CRF*, using Bi-LSTM to learn character representations and CRF to decode labels [28]; 2) *LSTM+Softmax*, using Bi-LSTM to learn character representations and softmax for label decoding; 3) *MDA*, multi-task domain adaptation method proposed by Peng et al. [28], which jointly trains CNER and CWS models via multi-task learning; 4) *LSTM+CRF+Pseudo*, *LSTM+CRF* trained on both original and pseudo labeled samples; 5) *CLC*, our CNN-LSTM-CRF architecture for CNER; 6) *CNN+LSTM+Softmax*, replacing the CRF layer in *CLC* with a softmax layer; 7) *CLC+Joint*, our unified framework for jointly training CNER and CWS models; 8) *CLC+Pseudo*, *CLC* trained on both original and pseudo labeled samples; 9) *CLC+Joint+Pseudo*, *CLC* with both joint training and pseudo labeled data.

We conducted experiments on different ratios (i.e., 5%, 25% and 100%) of training data to test the performance of different methods with insufficient and sufficient labeled samples. For those methods which involve pseudo labeled samples, the number of pseudo samples is the same with the real labeled samples. The experimental results on the *Bakeoff-3* and *Bakeoff-4* datasets are shown in Tables 2 and 3 respectively. We have several findings from the results.

First, our CNN-LSTM-CRF architecture (*CLC*) performs better than the popular *LSTM-CRF* architecture [28] on Chinese NER. For example, on the *Bakeoff-3* dataset, the improvement of *CLC* over *LSTM-CRF* in terms of Fscore is 6.08% on 5% training data and 3.31% on 100% training data. These results validate that our CNN-LSTM-CRF architecture is more suitable for Chinese NER than LSTM-CRF. This is because our model can capture both local and long-distance contexts of sentences to learn contextual character representations, which is useful for Chinese NER.

Second, joint label decoding via CRF is better than independent label decoding for Chinese NER . For instance, *CLC* performs much better than *CNN+LSTM+Softmax*, and *LSTM+CRF* can consistently outperform *LSTM+Softmax*. This is because there are strong dependencies between neighboring NER labels, and CRF can effectively capture these dependencies. In addition, *CNN+LSTM+Softmax* consistently outperforms *LSTM+Softmax*, although they both use independent label decoding. It further validates better character representations can be learned by combing CNN and LSTM to capture both local and long-distance contexts.

Third, by jointly training CNER and CWS models using our unified framework, our approach (i.e., *CLC+Joint*) can achieve better performance, especially when training data is insufficient. This is because there is inherent relatedness between CNER and CWS, since CNER model needs to identify entity boundaries and the goal of CWS is to identify word boundaries. Experimental results show that our unified framework is effective in exploiting the inherent

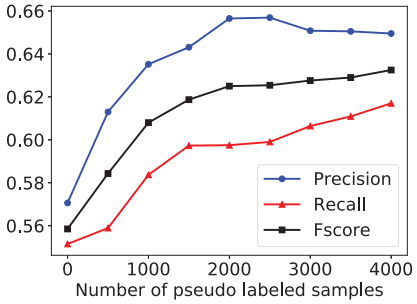


Figure 2: The influence of pseudo labeled data size.

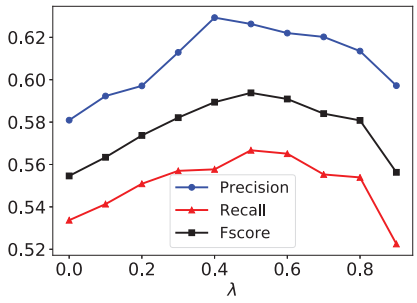


Figure 3: The influence of λ value.

relatedness between CNER and CWS, and can improve the ability of CNER model in identifying entity boundaries via jointly training CNER and CWS models. *MDA* method [28] also combines CNER and CWS. Our *CLC+Joint* approach can consistently outperform *MDA*. There are two major differences between our approach and *MDA*. First, our approach is based on the CNN-LSTM-CRF architecture while *MDA* is based on LSTM-CRF. Second, in *MDA* method CNER and CWS are regarded as equally important and share the same character representations, while in our approach CWS is regarded as an axillary task and used to learn word boundary aware character representations from local contexts. These results indicate our approach is more suitable to exploit the useful information in word segmentation for CNER.

Fourth, the automatically generated pseudo labeled samples can effectively improve the performance of our approach and baseline methods. For example, *CLC+Pseudo* significantly outperforms *CLC*, and *LSTM+CRF+Pseudo* consistently outperforms *LSTM+CRF*, especially in terms of OOV recall. Besides, the performance improvement becomes bigger when training data is insufficient. These results validate that the pseudo labeled samples automatically generated from labeled data can provide useful information for training CNER model. They can improve the generalization ability of CNER model by encouraging it to capture the contextual patterns of entity names, which is important for recognizing OOV entities. Moreover, accoding to Tables 2 and 3, after incorporating both pseudo labeled samples and joint training with CWS, the performance of our approach (*CLC+Joint+Pseudo*) can be further improved.

4.3 Influence of Pseudo Labeled Samples

Next we explore the influence of the number of automatically generated pseudo labeled samples on the performance of our approach. We conducted experiments on both datasets and the results on the *Bakeoff-4* dataset with 5% training data are shown in Figure 2. The results on the *Bakeoff-3* dataset show similar patterns.

According to Figure 2, as the number of pseudo labeled samples increases, the recall and Fscore of our approach consistently improve, while the precision first increases and slightly declines afterwards. These results show that the pseudo labeled samples automatically generated from labeled data can provide useful information for training CNER model and can improve its ability in capturing contextual patterns of entity names, which is beneficial for identifying OOV entities. Thus, the recall of our approach continuously improves when more pseudo labeled samples are incorporated. However, when too many pseudo labeled samples are used, the CNER model may overfit these patterns, resulting in the slight decline of precision.

4.4 Influence of Hyperparameters

In this section we conduct experiments to explore the influence of hyperparameters on the performance of our approach. Due to space limit, here we only show the influence of the most important hyperparameter on our approach, i.e., λ in Eq. (6). The results are summarized in Figure 3. This experiment was conducted on the *Bakeoff-4* dataset with 5% training data. λ is used to control the relative importance of CWS loss in the overall loss of our joint training framework (Eq. (6)). According to Figure 3, the performance of our approach improves when λ increases from 0, and declines when λ becomes too large. This is because when λ is too small, the useful information in word segmentation is not fully exploited. Thus, the performance is not optimal. When λ becomes too large, the auxiliary CWS task is over-emphasized and the loss of NER is not fully respected. Thus, the performance is also not optimal. A moderate value of λ is most suitable for our approach.

5 CONCLUSION

In this paper we propose a neural approach for Chinese named entity recognition. First, we propose a CNN-LSTM-CRF neural architecture for Chinese NER, where CNN and Bi-LSTM networks are used to learn character representations from both local and long-distance contexts, and CRF is used for joint label decoding. Second, we propose a unified framework to jointly train Chinese named entity recognition and word segmentation models to exploit the inherent relatedness between these two tasks and enhance the ability of Chinese NER model in identifying entity boundaries. In addition, we propose an automatic method to generate pseudo labeled samples from existing labeled data to enrich the training data. Extensive experiments on two benchmark datasets validate that our approach can effectively improve the performance of Chinese named entity recognition especially when training data is insufficient, and can consistently outperform many baseline methods.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Program of China under Grant number 2018YFC1604002, and the National Natural Science Foundation of China under Grant numbers U1705261, U1536201, and U1536207.

REFERENCES

[1] Aitao Chen, Fuchun Peng, Roy Shan, and Gordon Sun. 2006. Chinese named entity recognition with conditional probabilistic models. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*. 173–176.

[2] Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015. Long short-term memory neural networks for chinese word segmentation. In *EMNLP*. 1197–1206.

[3] Jason P. C. Chiu and Eric Nichols. 2016. Named Entity Recognition with Bidirectional LSTM-CNNs. *TACL* 4 (2016), 357–370.

[4] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[5] Yann Dauphin, Harm de Vries, and Yoshua Bengio. 2015. Equilibrated adaptive learning rates for non-convex optimization. In *NIPS*. 1504–1512.

[6] Chuanhai Dong, Huijia Wu, Jiajun Zhang, and Chengqing Zong. 2017. Multi-channel LSTM-CRF for Named Entity Recognition in Chinese Social Media. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*. 197–208.

[7] Chuanhai Dong, Jiajun Zhang, Chengqing Zong, Masanori Hattori, and Hui Di. 2016. Character-Based LSTM-CRF with Radical-Level Features for Chinese Named Entity Recognition. In *International Conference on Computer Processing of Oriental Languages*. 239–250.

[8] Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question answering over freebase with multi-column convolutional neural networks. In *ACL*. 260–269.

[9] Cicero dos Santos and Victor Guimarães. 2015. Boosting Named Entity Recognition with Neural Character Embeddings. In *Proceedings of the Fifth Named Entity Workshop*. 25–33.

[10] Jianfeng Gao, Mu Li, Andi Wu, and Chang-Ning Huang. 2005. Chinese word segmentation and named entity recognition: A pragmatic approach. *Computational Linguistics* 31, 4 (2005), 531–574.

[11] Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *SIGIR*. 765–774.

[12] Hangfeng He and Xu Sun. 2017. A Unified Model for Cross-Domain and Semi-Supervised Named Entity Recognition in Chinese Social Media.. In *AAAI*. 3216–3222.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[14] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).

[15] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*. 282–289.

[16] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *NAACL*. 260–270.

[17] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[18] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation* 1, 4 (1989), 541–551.

[19] Gina-Anne Levow. 2006. The third international Chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*. 108–117.

[20] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *ACL*. 2124–2133.

[21] Junxin Liu, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2018. Neural Chinese Word Segmentation with Dictionary Knowledge. In *NLPCC*. 80–91.

[22] Zhangxun Liu, Conghui Zhu, and Tiejun Zhao. 2010. Chinese named entity recognition with a sequence labeling approach: based on characters, or based on words? In *Advanced intelligent computing theories and applications with aspects of artificial intelligence*. Springer, 634–640.

[23] Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *EMNLP*. 879–888.

[24] Wencan Luo and Fan Yang. 2016. An Empirical Study of Automatic Chinese Word Segmentation for Spoken Language Understanding and Named Entity Recognition. In *NAACL*. 238–248.

[25] Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *ACL*. 1064–1074.

[26] Shotaro Misawa, Motoki Taniguchi, Yasuhide Miura, and Tomoko Ohkuma. 2017. Character-based Bidirectional LSTM-CRF with words and characters for Japanese Named Entity Recognition. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*. 97–102.

[27] Nanyun Peng and Mark Dredze. 2015. Named Entity Recognition for Chinese Social Media with Jointly Trained Embeddings.. In *EMNLP*. 548–554.

[28] Nanyun Peng and Mark Dredze. 2017. Multi-task Domain Adaptation for Sequence Tagging. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*. 91–100.

[29] Burr Settles. 2004. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications*. 104–107.

[30] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research* 15, 1 (2014), 1929–1958.

[31] Chunqi Wang, Wei Chen, and Bo Xu. 2017. Named Entity Recognition with Gated Convolutional Neural Networks. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*. 110–121.

[32] Xiaofeng Yu, Wai Lam, Shing-Kit Chan, Yiu Kei Wu, and Bo Chen. 2008. Chinese NER Using CRFs and Logic for the Fourth SIGHAN Bakeoff.. In *IJCNLP*. 102–105.

[33] Yuejie Zhang, Zhiting Xu, and Tao Zhang. 2008. Fusion of multiple features for chinese named entity recognition based on CRF model. In *Asia Information Retrieval Symposium*. 95–106.

Semi-supervised Multi-view Individual and Sharable Feature Learning for Webpage Classification

Fei Wu*
College of Automation, Nanjing University of Posts and Telecommunications
China
wufei_8888@126.com

Xiao-Yuan Jing
School of Computer, Wuhan University,
College of Automation, Nanjing University of Posts and Telecommunications
China
jingxy_2000@126.com

Jun Zhou
College of Automation, Nanjing University of Posts and Telecommunications
China
darkness127@163.com

Yimu Ji
College of Computer, Nanjing University of Posts and Telecommunications
China

Chao Lan
Department of Computer Science, University of Wyoming
United States

Qinghua Huang
School of Science, Nanjing University of Posts and Telecommunications
China

Ruchuan Wang
College of Computer, Nanjing University of Posts and Telecommunications
China

ABSTRACT

Semi-supervised multi-view feature learning (SMFL) is a feasible solution for webpage classification. However, how to fully extract the complementarity and correlation information effectively under semi-supervised setting has not been well studied. In this paper, we propose a semi-supervised multi-view individual and sharable feature learning (SMISFL) approach, which jointly learns multiple view-individual transformations and one sharable transformation to explore the view-specific property for each view and the common property across views. We design a semi-supervised multi-view similarity preserving term, which fully utilizes the label information of labeled samples and similarity information of unlabeled samples from both intra-view and inter-view aspects. To promote learning of diversity, we impose a constraint on view-individual transformation to make the learned view-specific features to be statistically uncorrelated. Furthermore, we train a linear classifier, such that view-specific and shared features can be effectively combined for classification. Experiments on widely used webpage datasets demonstrate that SMISFL can significantly outperform state-of-the-art SMFL and webpage classification methods.

CCS CONCEPTS

• **Information systems** → **World Wide Web**; *Data extraction and integration*; • **Computing methodologies** → machine learning.

KEYWORDS

Webpage classification, semi-supervised multi-view feature learning (SMFL), view-specific feature, view-shared feature

ACM Reference Format:
Fei Wu, Xiao-Yuan Jing, Jun Zhou, Yimu Ji, Chao Lan, Qinghua Huang, and Ruchuan Wang. 2019. Semi-supervised Multi-view Individual and Sharable Feature Learning for Webpage Classification. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313492

*corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313492

1 INTRODUCTION

In World Wide Web, there exists a huge repository of data in the form of webpages, and the size of the web is growing day-by-day. This makes it challenging for users to search relevant contents accurately. Classification of webpages, that is allocating a webpage to one category that can describe its contents [14], is regarded to be able to help improve the quality of search results [7]. Webpage usually contains multiple types of information, such as images, hyperlinks, URLs, HTML tags, etc. Different types of information is combined to describe the same webpage. Thus, webpage classification is a typical kind of multi-view data-based classification task [20, 25, 29, 37, 38]. In webpage classification task, we can only make use of a limited number of labeled webpages with a large number of unlabeled pages, since manually labeling a large repository of webpages is time and labor consuming and is infeasible. Hence, webpage classification is a semi-supervised learning problem [39, 40]. Besides, since a page usually contains rich information, the webpage data is high-dimensional [16]. Due to these characteristics, semi-supervised multi-view feature learning (SMFL) is a feasible solution for webpage classification.

In recent years, some SMFL methods have been presented. Chen et al. [1] developed a semi-paired and semi-supervised generalized correlation analysis (S^2GCA) algorithm. Volpi et al. [23] addressed manifold-regularized semi-supervised kernel canonical correlation analysis (MR-skCCA) for hyperspectral data classification. Jing

et al. presented the uncorrelated semi-supervised intra-view and inter-view manifold discriminant (USI^{2}MD) [5] method and further designed the semi-supervised multi-view correlation feature learning (SMCFL) [6] method for webpage classification. Wan et al. [24] addressed the cost sensitive semi-supervised canonical correlation analysis (CS^{3}CCA) algorithm. Zhang et al. [36] presented generalized semi-supervised structured subspace learning (GSS-SL) for cross-modal retrieval. To our knowledge, three SMFL methods, i.e., S^{2}GCA, USI^{2}MD and SMCFL, have been successfully used for webpage classification.

1.1 Motivation and Contribution

Most of existing SMFL methods generally map data of multiple views into a holistically latent space either through pursuing one projective transformation for each view, e.g., S^{2}GCA, USI^{2}MD, GSS-SL, etc., or by searching for a common transformation, e.g., SMCFL. As discussed in [6], correlation information exploration is important for webpage data. However, the first type of SMFL methods cannot make full use of the correlation information across multiple views. For the second type of SMFL methods, they cannot well preserve specific characteristic of each view. Therefore, how to effectively explore and utilize both the complementary information (particularity) and the correlation information (commonality) of different views is a significant research topic.

A few supervised/unsupervised multi-view learning methods have tried to preserve the specific characteristic of each view and uncover the shared component simultaneously [3, 12, 30]. Inspired by these works, we propose a novel SMFL approach for webpage classification. The contributions of our study can be summarized as following four points:

(1) We design a semi-supervised multi-view similarity preserving term. It jointly learns multiple view-individual transformations and one sharable transformation for exploring the particularity and the commonality of different views in the semi-supervised scenario. With this term, label information of labeled samples and the similarity information of unlabeled samples can be well utilized from both intra-view and inter-view aspects.

(2) To reduce the redundancy information across views, we design a multi-view statistical uncorrelation term, which makes the learned view-specific features to be statistically uncorrelated.

(3) We train a linear classifier based on view-specific and shared features, which further improves the discriminability of view-specific and shared features.

(4) By combining these designed terms, we propose a novel SMFL approach, named semi-supervised multi-view individual and sharable feature learning (SMISFL). Experiments on the widely used WebKB [1] and Internet Advertisements [8] datasets demonstrate the proposed approach can bring significant performance improvement for webpage classification.

2 RELATED WORK

2.1 Webpage Classification Methods

A number of webpage classification methods have been developed over the past decade, and most of these methods adopt machine learning-based techniques for performing the task [11, 28, 32]. Li et al. [10] presented the two-view transductive SVM (TTSVM) algorithm, which improves transductive SVM by employing a multi-view learning paradigm, and evaluated the effectiveness of the algorithm with webpage data. Wang et al. [26] studied large-scale cross-language webpage classification via dual knowledge transfer by using the nonnegative matrix trifactorization technique. Shivashankar et al. [19] built a positive and unlabeled learning model to determine whether two endpoint URLs refer to the same entity with the distant supervision. Saleh et al. [15] introduced a strategy called classification using multilayered domain ontology (CMDO) for vertical web page classification. These webpage classification works do not take all three characteristics of webpage classification into consideration.

In recent years, three SMFL methods, i.e., S^{2}GCA [1], USI^{2}MD [5], and SMCFL [6], have been applied to the webpage classification task. S^{2}GCA performs canonical correlation analysis on paired data, and learns features with the aim of preserving global structure of unlabeled data and achieving separability of different classes. USI^{2}MD learns a projective transformation for each view, and employs the semi-supervised intra-view and inter-view manifold discriminant learning schema and the semi-supervised uncorrelation constraint for discriminant features learning. SMCFL only learns a multi-view shared transformation that projects samples of multiple views to a common space, by considering the discriminant correlation of labeled and unlabeled samples. These three methods do not consider to explore both the particularity and the commonality across views for helping classification. In addition, they do not consider classifier design when training the classification models.

2.2 SMFL Methods

Multi-view learning in semi-supervised scenario is a hot research topic [2, 4, 13, 17, 31, 34, 35]. In recent years, some SMFL methods have been presented. MR-skCCA [23] performs kernel canonical correlation analysis between views on labeled and unlabeled data with the graph Laplacian-based manifold regularization. CS^{3}CCA [24] first uses the L2-norm manner to obtain the soft label for unlabeled data, and then embeds the misclassification cost into canonical correlation analysis. GSS-SL [36] designs the label graph constraint to help predict relevant class labels for unlabeled data, and takes the label space as a linkage to model the correlation among different modalities. Existing SMFL methods do not fully take exploration and utilization of both the view-specific and shared information into consideration.

Recently, a few supervised/unsupervised multi-view learning methods have tried to make use of the view-specific and shared information [3, 12, 30]. Xue et al. [30] presented a multi-view correlated feature learning with shared component algorithm, which learns individual information in each view and tries to capture feature correlation among multiple views by learning a shared component. Hu et al. [3] presented a sharable and individual multi-view deep metric learning (MvDML) algorithm, which learns an individual distance metric for each view to retain its specific property and learns a shared representation for different views to preserve their common properties based on deep neural network. These works are supervised/unsupervised methods without considering the joint use of labeled and unlabeled data. And these works do not well

study either the aspect of how to effectively extract discriminant features with low redundancy or the aspect of how to learn classifier with the aim of effectively using both the view-specific and shared features jointly.

3 PROPOSED APPROACH

3.1 Semi-supervised Multi-view Similarity Preserving

Let $X_m \in \mathbb{R}^{d\times N}$ be the training webpage sample set of the m^{th} $(m=1,\cdots,M)$ view, where M denotes the number of views and N is the number of samples in each view. Suppose that X_m consists of two parts of samples, i.e., $X_m = \{x_m^i\}_{i=1}^N = \left[\bar{X}_m, \widehat{X}_m\right]$, where $\bar{X}_m \in \mathbb{R}^{d\times N_l}$ denotes the labeled data set in the m^{th} view and $\widehat{X}_m \in \mathbb{R}^{d\times N_u}$ contains unlabeled data in the m^{th} view. Here, N_l and N_u separately denote the number of samples in $\bar{X}_m$ and $\widehat{X}_m$. We aim to learn the transformation $V_m = \begin{pmatrix} V_m^s \\ V^c \end{pmatrix} \in \mathbb{R}^{(d_v+d_c)\times d},\ m=1,\cdots,M$, such that sample x_m^i can be mapped into a discriminant subspace through $V_m x_m^i$. Here, $V_m^s \in \mathbb{R}^{d_v\times d}$ is the view-specific transformation, which embodies the intrinsic characteristic of the m^{th} view, and $V^c \in \mathbb{R}^{d_c\times d}$ is the view-sharable transformation, which reflects the correlation information of M views. For simplicity, d_c is considered to be equal to d_v in this paper.

For samples x_m^i and x_l^j, we define the following weight to characterize their similarity

$$w_{m,l}^{i,j} = \begin{cases} 1, \text{if } x_m^i \text{ and } x_l^j \text{ are from the same class} \\ -1, \text{if } x_m^i \text{ and } x_l^j \text{ are from different classes} \\ \exp(-\left\|x_m^i - x_l^j\right\|_2^2 / (2\sigma^2)), \text{if } x_m^i \text{ or } x_l^j \text{ is unlabeled} \end{cases}$$

where σ is calculated as the standard variation of samples in the m^{th} and the l^{th} views. $W_{m,l}\ (m=1,\cdots,M;\ l=1,\cdots,M)$ is the similarity matrix between the m^{th} and l^{th} views. And the overall similarity matrix can be defined as $W = \begin{bmatrix} W_{1,1} & W_{1,2} & \cdots & W_{1,M} \\ W_{2,1} & W_{2,2} & \cdots & W_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ W_{M,1} & W_{M,2} & \cdots & W_{M,M} \end{bmatrix}$.

In order to learn discriminant transformations, such that in the projected space, limited number of labeled samples from the same class cluster together, the labeled samples with different class labels can be well separable, and unlabeled samples can preserve the global structure, we define the following semi-supervised multi-view similarity preserving term

$$\begin{aligned} J_1(V_m|_{m=1}^M) &= \frac{1}{2M^2N^2}\sum_{m=1}^{M}\sum_{i=1}^{N}\sum_{l=1}^{M}\sum_{j=1}^{N} w_{m,l}^{i,j}\left\|\begin{pmatrix} V_m^s \\ V^c \end{pmatrix}x_m^i - \begin{pmatrix} V_l^s \\ V^c \end{pmatrix}x_l^j\right\|_F^2 \\ &= \frac{1}{M^2N^2}\sum_{m=1}^{M}\sum_{l=1}^{M} tr\left(V_m X_m L_{m,l} X_l^T V_l^T\right) \end{aligned} \tag{1}$$

where L is the Laplacian matrix containing the sub-matrices $L_{m,l}$ $(m=1,\cdots,M;\ l=1,\cdots,M)$, and L is defined as $L = D - W$. Here, D is a diagonal matrix with the i^{th} diagonal element being calculated as the sum of the i^{th} row of the overall similarity matrix W. We should minimize J_1.

This term can fully utilize the label information of labeled samples and similarity information of unlabeled samples from both intra-view and inter-view aspects.

3.2 Multi-view Statistical Uncorrelation

In order to reduce the redundancy information across views and learn pure view-specific features with view-individual transformations, we design the following multi-view statistical uncorrelation term

$$J_2(V_m^s|_{m=1}^M) = \frac{1}{M(M-1)}\sum_{m=1}^{M}\sum_{l=1,m\neq l}^{M}\left\|V_m^s Cov_{m,l} V_l^{sT}\right\|_F^2 \tag{2}$$

where $Cov_{m,l} = \frac{1}{N}\sum_{i=1}^{N}\left(x_m^i - \mu_m\right)\left(x_l^i - \mu_l\right)^T$ is the cross-covariance matrix between the m^{th} and the l^{th} views. Here, μ_m and μ_l are separately the mean samples of X_m and X_l. We should also minimize this term to enhance the complementarity of features extracted from different views.

3.3 View-specific and Shared Features-based Classification Loss

We also design a linear classifier, such that the learned view-specific and view-shared features can be effectively combined for classification. The designed view-specific and shared features-based classification loss term is given as

$$J_3(P,b) = \left\|P\left(\left(V_1\bar{X}_1\right)^T,\cdots,\left(V_M\bar{X}_M\right)^T\right)^T + b\mathrm{I}^T - Y\right\|_F^2 \tag{3}$$

where $Y \in \mathbb{R}^{C\times N_l}$ is a label matrix with each column being a one-hot label vector, $P = [P_1, P_2, \cdots, P_{2m-1}, P_{2m}, \cdots, P_{2M-1}, P_{2M}] \in \mathbb{R}^{C\times(2Md_v)}$ is the projective matrix, $b \in \mathbb{R}^C$ is a bias term, and $\mathrm{I} \in \mathbb{R}^{N_l}$ is a vector full of 1. Here, C denotes the number of classes, and each $P_{2m-1} \in \mathbb{R}^{C\times d_v}$ and $P_{2m} \in \mathbb{R}^{C\times d_v}$ are sub-matrices of P. With this term, the discriminant ability of learned features can be further improved.

3.4 Total Objective of SMISFL

By combining the terms of semi-supervised multi-view similarity preserving, multi-view statistical uncorrelation, and the classification loss, we formulate the overall objective as

$$\begin{aligned} \min_{V_1,\cdots,V_M,P,b} J = &\frac{1}{2M^2N^2}\sum_{m=1}^{M}\sum_{i=1}^{N}\sum_{l=1}^{M}\sum_{j=1}^{N} w_{m,l}^{i,j}\left\|\begin{pmatrix} V_m^s \\ V^c \end{pmatrix}x_m^i - \begin{pmatrix} V_l^s \\ V^c \end{pmatrix}x_l^j\right\|_F^2 \\ &+\alpha\frac{1}{M(M-1)}\sum_{m=1}^{M}\sum_{l=1,m\neq l}^{M}\left\|V_m^s Cov_{m,l} V_l^{sT}\right\|_F^2 \\ &+\beta\left\|P\left(\left(V_1\bar{X}_1\right)^T,\cdots,\left(V_M\bar{X}_M\right)^T\right)^T + b\mathrm{I}^T - Y\right\|_F^2 \\ &+\gamma\left(\frac{1}{M}\sum_{m=1}^{M}\left\|V_m^s\right\|_{2,1} + \|V^c\|_F^2\right) \end{aligned} \tag{4}$$

where α, β and γ are balance factors. $\frac{1}{M}\sum_{m=1}^{M}\left\|V_m^s\right\|_{2,1} + \|V^c\|_F^2$ is a regularization term to avoid the overfitting problem. We employ the $\ell_{2,1}$-norm based regularizer, which makes $V_m^s,\ m=1,\cdots,M$ to be sparse in rows, such that discriminant features can be selected for each view.

3.5 Optimization

The objective (4) is not jointly convex with regard to variables $\left\{V_m^s|_{m=1}^M, V^c, P, b\right\}$ simultaneously. We thus employ an alternating optimization strategy to optimize these variables.

(1) Fix $\{V_1^s, V_2^s, \cdots, V_M^s\}$ and V^c, and update P, b

The objective function (4) is reduced to

$$\min_{P,b} J_3 = \left\| P\left(\left(V_1\bar{X}_1\right)^T, \cdots, \left(V_M\bar{X}_M\right)^T\right)^T + b\mathrm{I}^T - Y\right\|_F^2 \tag{5}$$

When updating $P(b)$, $b(P)$ is fixed. By setting the derivative of J_3 w.r.t. P, b to zero, we have

$$P = -BA^T\left(AA^T\right)^{-1}, b = -\frac{1}{N_l}D\mathrm{I} \tag{6}$$

where $A = \left(\left(V_1\bar{X}_1\right)^T, \cdots, \left(V_M\bar{X}_M\right)^T\right)^T$, $B = b\mathrm{I}^T - Y$, and $D = P\left(\left(V_1\bar{X}_1\right)^T, \cdots, \left(V_M\bar{X}_M\right)^T\right)^T - Y$.

(2) Fix P and b, and update $\{V_1^s, V_2^s, \cdots, V_M^s\}$ and V^c

We update V_m^s view by view. When updating V_m^s, V^c and $\left\{V_l^s\right\}_{l\neq m}$ are fixed. The objective function (4) is reduced to

$$\begin{aligned}\min_{V_m^s} \tfrac{1}{M^2N^2}&\left(\begin{array}{l} tr\left(V_m^s X_m L_{m,m} X_m^T V_m^{sT}\right) \\ +2\sum_{l\neq m}^{M} tr\left(V_m^s X_m L_{m,l} X_l^T V_l^{sT}\right)\end{array}\right) \\ &+\alpha\tfrac{2}{M(M-1)}\sum_{l\neq m}^{M}\left\|V_m^s Cov_{m,l} V_l^{sT}\right\|_F^2 \\ &+\beta\left\|P_{2m-1}V_m^s\bar{X}_m + E_1\right\|_F^2 + \gamma\tfrac{1}{M}\left\|V_m^s\right\|_{2,1}\end{aligned} \tag{7}$$

where $E_1 = \sum_{l\neq m}^{M} P_{2l-1}V_l^s\bar{X}_l + \sum_{m=1}^{M} P_{2m}V^c\bar{X}_m + \left(b\mathrm{I}^T - Y\right)$.

Differentiating (7) with respect to V_m^s and setting it to zero, we have

$$\begin{aligned}&\tfrac{1}{M^2N^2}\left(V_m^s\left(X_m L_{m,m} X_m^T\right) + \sum_{l\neq m}^{M}\left(V_l^s X_l L_{m,l}^T X_m^T\right)\right) \\ &+\alpha\tfrac{2}{M(M-1)}\sum_{l\neq m}^{M}\left(V_m^s Cov_{m,l} V_l^{sT} V_l^s Cov_{m,l}^T\right) \\ &+\beta\left(P_{2m-1}^T P_{2m-1} V_m^s\bar{X}_m\bar{X}_m^T + P_{2m-1}^T E_1\bar{X}_m^T\right) + \gamma\tfrac{1}{M}G_m^s V_m^s = 0\end{aligned} \tag{8}$$

where G_m^s is a diagonal matrix with its i^{th} diagonal element as $G_m^s(i,i) = 1/\left(2\left\|V_m^s(i,:)\right\|\right)$. Here $V_m^s(i,:)$ represents the i^{th} row of V_m^s. According to [33], the element in G_m^s is usually calculated as $G_m^s(i,i) = \frac{1}{2\sqrt{\|V_m^s(i,:)\|^2+\varepsilon}}$, where ε is set as a small positive value.

Then, we can obtain V_m^s by solving the following problem

$$V_m^s H_m^s + Q_m^s V_m^s + K_m^s V_m^s O_m^s = R_m^s \tag{9}$$

where $H_m^s = \frac{1}{M^2N^2}X_m L_{m,m} X_m^T + \frac{2\alpha}{M(M-1)}\sum_{l\neq m}^{M} Cov_{m,l} V_l^{sT} V_l^s Cov_{m,l}^T Q_m^s = \gamma\frac{1}{M}G_m^s$, $K_m^s = \beta P_{2m-1}^T P_{2m-1}$, $O_m^s = \bar{X}_m\bar{X}_m^T$, and $R_m^s = -\left(\frac{1}{M^2N^2}\sum_{l\neq m}^{M}\left(V_l^s X_l L_{m,l}^T X_m^T\right) + \beta P_{2m-1}^T E_1\bar{X}_m^T\right)$. The optimization problem in (9) can be solved by using the algorithm in the literatures [9, 22].

Algorithm 1 SMISFL

(1) **Input**: Multi-view training samples X_m, $m = 1, \cdots, M$, test sample $y = \{y_m\}|_{m=1}^M$.
(2) **Output**: The class label of y.
(a) Randomly initialize $V_m^s|_{m=1}^M$, V^c, P and b.
(b) **Repeat**:
(c) Update P and b with Eq. (6).
(d) Update $V_m^s|_{m=1}^M$ and V^c by solving (9) and (11).
(e) **Until Convergence**
(f) Map $\{y_m\}|_{m=1}^M$ with the learned transformations $V_m^s|_{m=1}^M$ and V^c by $\begin{Bmatrix} V_m^s \\ V^c \end{Bmatrix} y_m$, and then utilize the classifier parameters P and b for classification.

When updating V^c, $\left\{V_1^s, V_2^s, \cdots, V_M^s\right\}$ is fixed. The objective function (4) can be reduced to

$$\begin{aligned}\min_{V^c} \tfrac{1}{M^2N^2}&\sum_{m=1}^{M}\sum_{l=1}^{M} tr\left(V^c X_m L_{m,l} X_l^T V^{cT}\right) \\ &+\beta\left\|\sum_{m=1}^{M} P_{2m}V^c\bar{X}_m + E_2\right\|_F^2 + \gamma\left\|V^c\right\|_F^2\end{aligned} \tag{10}$$

where $E_2 = \sum_{m=1}^{M} P_{2m-1}V_m^s\bar{X}_m + \left(b\mathrm{I}^T - Y\right)$.

By firstly differentiating (10) with respect to V^c and setting it to zero, we can obtain V^c by solving the following problem

$$V^c H^c + Q^c V^c + \sum_{m=1}^{M}\sum_{l=1}^{M} K_{ml}^c V^c O_{ml}^c = R^c \tag{11}$$

where $H^c = \frac{1}{M^2N^2}\sum_{m=1}^{M}\sum_{l=1}^{M} X_m L_{m,l} X_l^T$, $Q^c = \gamma\mathrm{I}$, $K_{ml}^c = \beta P_{2m}^T P_{2l}$, $O_{ml}^c = \bar{X}_l\bar{X}_m^T$, and $R^c = -\beta\sum_{m=1}^{M} P_{2m}^T E_2\bar{X}_m^T$. The problem (11) can also be solved by using the algorithm in the literatures [9, 22]. Algorithm 1 summarizes the proposed SMISFL approach.

4 EXPERIMENTS

4.1 Dataset

We validate the effectiveness of our approach on two widely used datasets, i.e., WebKB [1] and Internet Advertisements [8] (AD). WebKB contains 1051 webpages, described by two views, i.e., the page view that includes the textual content on the page and link view that includes the hyperlinks pointing to the page. 1051 pages can be categorized into two classes: the course class including 230 pages and the non-course class including 821 pages. We use the preprocessed version of this dataset provided by [18], where 3000-dimensional and 1840-dimensional feature vectors are separately extracted from the page and link documents. The AD dataset includes 3279 samples that can be categorized into two classes: 458 advertisement and 2821 non-advertisement samples. A preprocessed version of this dataset [8] is used, where each sample is regarded as a binary vector with quite large sparsity. As in [5, 6], three views, i.e., 495 base URL features, 472 destination URL features and 457 image URL features, are selected.

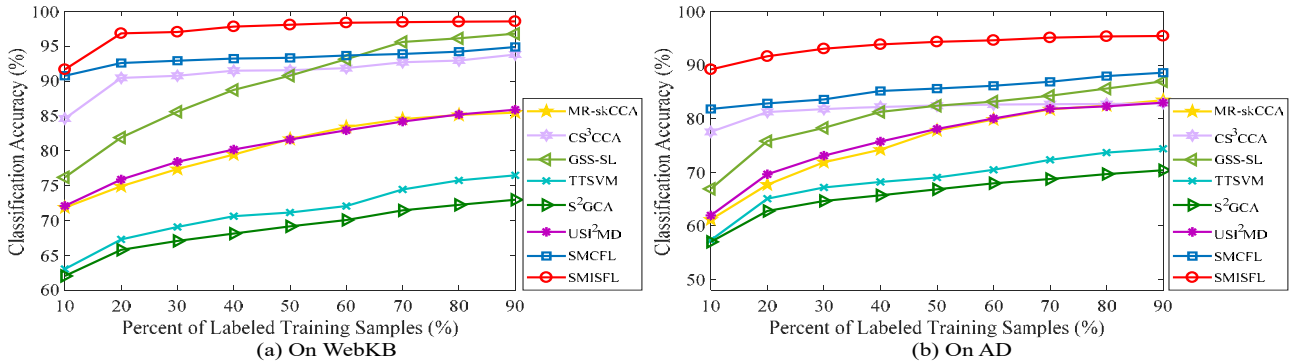


Figure 1: Average classification accuracies of compared methods on WebKB and AD datasets.

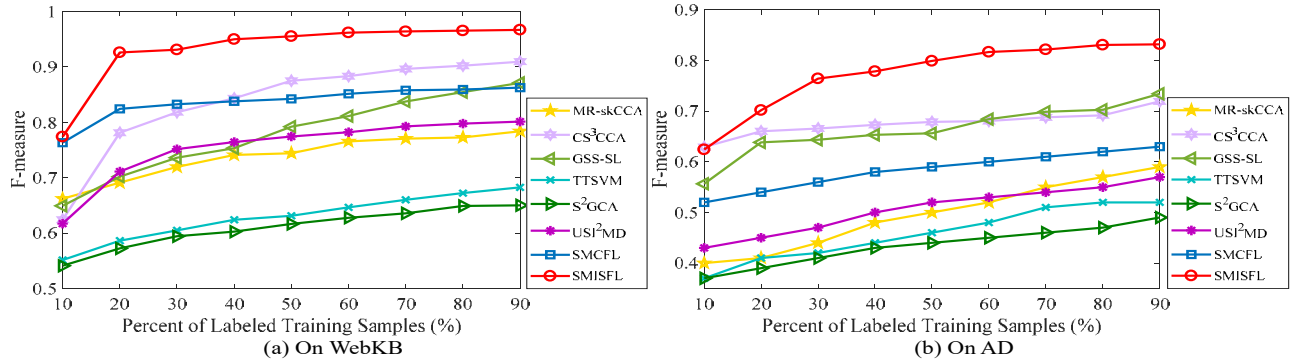


Figure 2: Average F-measure values of compared methods on WebKB and AD datasets.

Table 1: Average classification results (% for CA) and running time (seconds) of compared methods.

Method	WebKB				AD			
	CA	F1	TR	TE	CA	F1	TR	TE
CS^{3}CCA	91.13	0.84	0.56	0.13	81.80	0.68	5.80	1.37
GSS-SL	89.42	0.78	0.61	0.15	80.53	0.66	3.57	1.21
TTSVM	71.09	0.63	-	0.35	68.62	0.46	-	0.95
S^{2}GCA	68.75	0.61	2.84	0.34	65.96	0.43	7.78	1.12
USI^{2}MD	80.80	0.76	2.76	0.18	76.25	0.51	18.65	1.21
SMCFL	93.27	0.84	0.44	0.09	85.40	0.58	5.08	0.28
SMISFL	**97.25**	**0.93**	**0.21**	**0.05**	**93.63**	**0.77**	**0.45**	**0.08**

4.2 Experimental Settings

In experiments, we compare our SMISFL approach with two categories of state-of-the-art related methods including: (1) Webpage classification methods: **TTSVM** [10], **S^{2}GCA** [1], **USI^{2}MD** [5], and **SMCFL** [6]; (2) SMFL methods: **MR-skCCA** [23], **CS^{3}CCA** [24], and **GSS-SL** [36].

Like in [5, 6], PCA [21] is performed to reduce the dimensionalities of samples in different views to 1050 for WebKB and 456 for AD. On each dataset, we randomly select half of each class for training and use the remaining samples for testing. The training samples are further split into the labeled training sample set and the unlabeled training set according to a certain percentage (e.g., 20% of training samples used as labeled samples) for semi-supervised learning. TTSVM, S^{2}GCA and CS^{3}CCA were originally designed for two views-based problems. Following [5, 6], on AD dataset, two views that yield the best classification results are used for these three methods. For compared methods, we report their results with the listed results in [6] or with the codes provided by the authors. In experiments, the tuning parameters (α, β and γ in Eq. (4), and the dimensionality of V_m, i.e., d_v) of SMISFL are set as $\alpha = 1$, $\beta = 10^2$, $\gamma = 10^2$, and $d_v = 10$ on WebKB, and $\alpha = 1$, $\beta = 10^{-1}$, $\gamma = 10^2$, and $d_v = 10$ on AD.

4.3 Results and Analysis

In this paper, we adopt the measures of classification accuracy (CA) and F-measure (F1) [27] to evaluate the classification effects

of our approach and compared methods. We perform webpage classification experiments with the ratio of the number of labeled training samples to the total number of training samples increasing from 10% to 90%. Figs. 1 and 2 show the results measured by CA and F1 on two datasets, respectively. In these figures, each result is the average result across 20 random running. To statistically observe the results in the figures, we also calculate the mean result of each method across different percentages of labeled training samples, and we list these mean results in Table 1.

In Figs. 1 and 2, and Table 1, our SMISFL approach can always obtain better classification results with respect to the CA and F1 measures than compared webpage classification methods and state-of-the-art SMFL methods. From Table 1, SMISFL can improve the classification results by **3.98%**=(97.25%-93.27%) for CA and **0.09**=(0.93-0.84) for F1 on WebKB, and improve the classification results by **8.23%**=(93.63%-85.40%) for CA and **0.09**=(0.77-0.68) for F1 on AD. Better F1 results also mean that SMISFL can well classify the samples of the minority class, since samples used for training the model are class-imbalanced. The reason for the classification result improvement against all compared methods mainly lies in that the complementary information and the correlation information of different views of webpage data is effectively explored and leveraged under semi-supervised setting for classification.

Table 1 also reports the average training (TR) and testing (TE) time (for the total test set) of compared methods in the semi-supervised learning case that 20 percent of training samples are used as labeled samples. Our hardware configuration comprises i7-8700K CPU and 32 GB memory. From the table, SMISFL needs significantly less training and testing time than compared webpage classification and SMFL methods.

4.4 Discussion

In this subsection, we evaluate the important components of SMISFL. We separately call the version of SMISFL without the semi-supervised multi-view similarity preserving term as **SMISFL-s**, the version of SMISFL without the multi-view statistical uncorrelation term as **SMISFL-u**, the version of SMISFL without the view-specific and shared features-based classification loss term as **SMISFL-c**. For SMISFL-c, we concatenate feature representations of multiple views and employ the nearest neighbor classifier for classification. Furthermore, we conduct SMISFL without learning the view-individual transformation or the view-sharable transformation, and separately call these versions as **SMSFL** and **SMIFL**. Table 2 shows the result comparison.

From the table, the classification results of SMISFL-s, SMISFL-u and SMISFL-c are obviously inferior to that of SMISFL, and when only one of view-specific and shared transformations is used, SMISFL experiences an obvious performance degradation. This result means that the learned complementary and correlation features from multi-view data are useful for webpage classification, and the joint utilization of view-specific and shared features helps improve the performance. In addition, the designed three terms can help learn features with favorable discriminative capability.

Table 2: Classification results of variants of SMISFL.

Method	WebKB		AD	
	CA(%)	F1	CA(%)	F1
SMISFL-s	83.36	0.69	89.72	0.66
SMISFL-u	96.86	0.89	92.64	0.73
SMISFL-c	90.87	0.78	89.49	0.70
SMSFL	95.43	0.89	90.96	0.65
SMIFL	94.93	0.90	90.12	0.72
SMISFL	**97.25**	**0.93**	**93.63**	**0.77**

5 CONCLUSION

In this paper, we propose a novel SMFL approach SMISFL for webpage classification. It jointly learns the view-individual and sharable transformations by fully using of the labeled and unlabeled webpage data. Discriminant features with complementarity and correlation are well explored and effectively combined for classification. Experimental results on two widely used datasets demonstrate that SMISFL outperforms state-of-the-art SMFL and webpage classification methods. Results also validate the effectiveness of the semi-supervised view-individual and sharable feature learning scheme.

6 ACKNOWLEDGMENT

The work in this paper was supported by the National Natural Science Foundation of China (No. 61702280), National Postdoctoral Program for Innovative Talents (No. BX20180146), Natural Science Foundation of Jiangsu Province (No. BK20170900), and Scientific Research Staring Foundation for Introduced Talents in NJUPT (NUPTSF, No. NY217009).

REFERENCES

[1] Xiaohong Chen, Songcan Chen, Hui Xue, and Xudong Zhou. 2012. A unified dimensionality reduction framework for semi-paired and semi-supervised multi-view data. *Pattern Recognition* 45, 5 (2012), 2005–2018.

[2] Chen Gong, Dacheng Tao, Stephen J Maybank, Wei Liu, Guoliang Kang, and Jie Yang. 2016. Multi-modal curriculum learning for semi-supervised image classification. *IEEE Transactions on Image Processing* 25, 7 (2016), 3249–3260.

[3] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. 2018. Sharable and individual multi-view metric learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 9 (2018), 2281–2288.

[4] Yu Jiang, Jing Liu, Zechao Li, and Hanqing Lu. 2014. Semi-supervised unified latent factor learning with multi-view data. *Machine Vision and Applications* 25, 7 (2014), 1635–1645.

[5] Xiao-Yuan Jing, Qian Liu, Fei Wu, Baowen Xu, Yang-Ping Zhu, and Songcan Chen. 2015. Web page classification based on uncorrelated semi-supervised intra-view and inter-view manifold discriminant feature extraction. In *Proceedings of the 24th International Joint Conferences on Artificial Intelligence*. 2255–2261.

[6] Xiao-Yuan Jing, Fei Wu, Xiwei Dong, Shiguang Shan, and Songcan Chen. 2017. Semi-supervised multi-view correlation feature learning with application to webpage classification. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. 1374–1381.

[7] Vinod Kumar. 2016. *Design and implementation of an efficient framework for web page classification.* Ph.D. Dissertation. Thapar University.

[8] Nicholas Kushmerick. 1999. Learning to remove internet advertisements. In *Proceedings of the 3rd Annual Conference on Autonomous Agents*. 175–181.

[9] Peter Lancaster. 1970. Explicit solutions of linear matrix equations. *SIAM Rev.* 12, 4 (1970), 544–566.

[10] Guangxia Li, Kuiyu Chang, and Steven CH Hoi. 2012. Multiview semi-supervised learning with consensus. *IEEE Transactions on Knowledge and Data Engineering* 24, 11 (2012), 2040–2051.

[11] Daniel López-Sánchez, Juan M Corchado, and Angélica González Arrieta. 2017. A CBR system for image-based webpage classification: case representation with convolutional neural networks. In *Proceedings of the 30th International Florida Artificial Intelligence Research Society Conference*. 483–488.

[12] Shirui Luo, Changqing Zhang, Wei Zhang, and Xiaochun Cao. 2018. Consistent and specific multi-view subspace clustering. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. 3730–3737.

[13] Feiping Nie, Jing Li, and Xuelong Li. 2017. Convex multiview semi-supervised classification. *IEEE Transactions on Image Processing* 26, 12 (2017), 5718–5729.

[14] Xiaoguang Qi and Brian D Davison. 2009. Web page classification: Features and algorithms. *Comput. Surveys* 41, 2 (2009), Artical No. 12.

[15] Ahmed I Saleh, Mohammed F Al Rahmawy, and Arwa E Abulwafa. 2017. A semantic based web page classification strategy using multi-layered domain ontology. *World Wide Web* 20, 5 (2017), 939–993.

[16] Esra Saraç. 2010. *Web page classification using ant colony optimization*. Ph.D. Dissertation. MS Thesis, Çukurova University.

[17] XiaoBo Shen and QuanSen Sun. 2014. A novel semi-supervised canonical correlation analysis and extensions for multi-view dimensionality reduction. *Journal of Visual Communication and Image Representation* 25, 8 (2014), 1894–1904.

[18] Vikas Sindhwani, Partha Niyogi, and Mikhail Belkin. 2005. Beyond the point cloud: from transductive to semi-supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning*. ACM, 824–831.

[19] S Subramanian, Timothy Baldwin, Julian Brooke, and Trevor Cohn. 2017. Pairwise webpage coreference classification using distant supervision. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 841–842.

[20] Shiliang Sun. 2013. A survey of multi-view machine learning. *Neural Computing and Applications* 23, 7-8 (2013), 2031–2038.

[21] Matthew Turk and Alex Pentland. 1991. Eigenfaces for recognition. *Journal of Cognitive Neuroscience* 3, 1 (1991), 71–86.

[22] Charles F Van Loan. 2000. The ubiquitous Kronecker product. *J. Comput. Appl. Math.* 123, 1-2 (2000), 85–100.

[23] Michele Volpi, Giona Matasci, Mikhail Kanevski, and Devis Tuia. 2014. Semi-supervised multiview embedding for hyperspectral data classification. *Neurocomputing* 145 (2014), 427–437.

[24] Jianwu Wan, Hongyuan Wang, and Ming Yang. 2017. Cost sensitive semi-supervised canonical correlation analysis for multi-view dimensionality reduction. *Neural Processing Letters* 45, 2 (2017), 411–430.

[25] Huibing Wang, Lin Feng, Xiangzhu Meng, Zhaofeng Chen, Laihang Yu, and Hongwei Zhang. 2017. Multi-view metric learning based on KL-divergence for similarity measurement. *Neurocomputing* 238 (2017), 269–276.

[26] Hua Wang, Feiping Nie, and Heng Huang. 2015. Large-scale cross-language web page classification via dual knowledge transfer using fast nonnegative matrix trifactorization. *ACM Transactions on Knowledge Discovery from Data* 10, 1 (2015), Article No. 1.

[27] Fei Wu, Xiao-Yuan Jing, Ying Sun, Jing Sun, Lin Huang, Fangyi Cui, and Yanfei Sun. 2018. Cross-project and within-project semisupervised software defect prediction: a unified approach. *IEEE Transactions on Reliability* 67, 2 (2018), 581–597.

[28] Ou Wu, Ruiguang Hu, Xue Mao, and Weiming Hu. 2014. Quality-based learning for web data classification. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*. 194–200.

[29] Chang Xu, Dacheng Tao, and Chao Xu. 2013. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634* (2013).

[30] Xiaowei Xue, Feiping Nie, Sen Wang, Xiaojun Chang, Bela Stantic, and Min Yao. 2017. Multi-view correlated feature learning by uncovering shared component. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. 2810–2816.

[31] Yang Yang, De-Chuan Zhan, Xiang-Rong Sheng, and Yuan Jiang. 2018. Semi-supervised multi-modal learning with incomplete modalities. In *Proceedings of the 27th International Joint Conferences on Artificial Intelligence*. 2998–3004.

[32] Yiming Yang, Lei Zhang, Guiquan Liu, and Enhong Chen. 2015. UPCA: An efficient URL-Pattern based algorithm for accurate web page classification. In *Proceedings of the 12th International Conference on Fuzzy Systems and Knowledge Discovery*. 1475–1480.

[33] Qiyue Yin, Shu Wu, and Liang Wang. 2017. Unified subspace learning for incomplete and unlabeled multi-view data. *Pattern Recognition* 67 (2017), 313–327.

[34] Deming Zhai, Hong Chang, Shiguang Shan, Xilin Chen, and Wen Gao. 2012. Multiview metric learning with global consistency and local smoothness. *ACM Transactions on Intelligent Systems and Technology* 3, 3 (2012), Article No. 53.

[35] Chenghao Zhang and Wei-Shi Zheng. 2017. Semi-supervised multi-view discrete hashing for fast image search. *IEEE Transactions on Image Processing* 26, 6 (2017), 2604–2617.

[36] Liang Zhang, Bingpeng Ma, Guorong Li, Qingming Huang, and Qi Tian. 2018. Generalized semi-supervised and structured subspace learning for cross-modal retrieval. *IEEE Transactions on Multimedia* 20, 1 (2018), 128–141.

[37] Lei Zhang, Yao Zhao, Zhenfeng Zhu, Dinggang Shen, and Shuiwang Ji. 2018. Multi-view missing data completion. *IEEE Transactions on Knowledge and Data Engineering* 30, 7 (2018), 1296–1309.

[38] Jing Zhao, Xijiong Xie, Xin Xu, and Shiliang Sun. 2017. Multi-view learning overview: Recent progress and new challenges. *Information Fusion* 38 (2017), 43–54.

[39] Zhi-Hua Zhou, De-Chuan Zhan, and Qiang Yang. 2007. Semi-supervised learning with very few labeled training examples. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*. 675–680.

[40] Xiaojin Zhu and Andrew B Goldberg. 2009. Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 3, 1 (2009), 1–130.

On Convexity and Bounds of Fairness-aware Classification

Yongkai Wu
University of Arkansas
yw009@uark.edu

Lu Zhang
University of Arkansas
lz006@uark.edu

Xintao Wu
University of Arkansas
xintaowu@uark.edu

ABSTRACT

In this paper, we study the fairness-aware classification problem by formulating it as a constrained optimization problem. Several limitations exist in previous works due to the lack of a theoretical framework for guiding the formulation. We propose a general fairness-aware framework to address previous limitations. Our framework provides: (1) various fairness metrics that can be incorporated into classic classification models as constraints; (2) the convex constrained optimization problem that can be solved efficiently; and (3) the lower and upper bounds of real-world fairness measures that are established using surrogate functions, providing a fairness guarantee for constrained classifiers. Within the framework, we propose a constraint-free criterion under which any learned classifier is guaranteed to be fair in terms of the specified fairness metric. If the constraint-free criterion fails to satisfy, we further develop the method based on the bounds for constructing fair classifiers. The experiments using real-world datasets demonstrate our theoretical results and show the effectiveness of the proposed framework.

CCS CONCEPTS

• **Theory of computation → Convex optimization**; • **Computing methodologies → Supervised learning by classification**; • **Applied computing →** *Law, social and behavioral sciences.*

KEYWORDS

Fairness-aware machine learning; classification;constrained optimization; algorithmic bias

ACM Reference Format:
Yongkai Wu, Lu Zhang, and Xintao Wu. 2019. On Convexity and Bounds of Fairness-aware Classification. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313723

1 INTRODUCTION

Fairness-aware classification is receiving increasing attention in the machine learning fields. Since the classification models seek to maximize the predictive accuracy, individuals may get unwanted digital bias when the models are deployed for making predictions. As fairness becomes a more and more important requirement in machine learning, it is imperative to ensure that the learned classification models can strike a balance between accurate and fair predictions. Previous works on this topic can be mainly categorized into two groups: the in-processing methods which incorporate the fairness constraints into the classic classification models (e.g., [5, 7, 8, 17, 18]), and the pre/post-processing methods which modify the training data and/or derive fair predictions based on the potentially unfair predictions made by the classifier (e.g., [4, 6, 20–22]). In this work, we focus on the in-processing methods.

Very recently, several works have been proposed for formulating the fairness-aware classification as constrained optimization problems [5, 7, 8, 10, 16–18]. Generally, they aim to minimize a loss function subject to certain fairness constraints, e.g., demographic parity (i.e., the difference of the positive predictions between the sensitive group and non-sensitive group) is less than some threshold. However, most quantitative fairness metrics such as demographic parity [13], mistreatment parity [17], etc., are non-convex due to the use of the indicator function, thus making the optimization problem intractable. A widely-used strategy to achieve convexity in optimization is to adopt surrogate functions for both loss function and constraints. In [18], the authors applied the linear surrogate functions to non-convex risk difference as the decision boundary fairness for margin-based classifiers. Similarly in [5], a convex constraint is derived from the risk difference. One challenge is that, when surrogate functions are used to convert non-convex functions to convex functions, estimation errors must exist due to the difference between the surrogate function and the original non-convex function. Thus, achieving the fairness constraints represented by surrogate functions does not necessarily guarantee achieving the real fairness criterion. Hence, how to achieve fairness-aware classification via constrained optimization still remains an open problem.

In this paper, we propose a general framework for fairness-aware classification which addresses the gap incurred by the estimation errors due to the surrogate function. The framework can formulate various commonly-used fairness metrics (risk difference [12], risk ratio [12], equal odds [6], etc.) as convex constraints that are then directly incorporated into classic classification models. Within the framework, we first present a constraint-free criterion (derived from the training data) which ensures that any classifier learned from the data will guarantee to be fair in terms of the specified fairness metric. Thus, when the criterion is satisfied, there is no need to add any fairness constraint into optimization for learning fair classifiers. When the criterion is not satisfied, we need to learn fair classifiers by solving the constrained optimization problems. To connect the surrogated fairness constraints to the original non-convex fairness metric, we further derive the lower and upper bounds of the real fairness measure based on the surrogate function, and develop the refined fairness constraints. This means that, if the refined constraints are satisfied, then it is guaranteed that the real fairness measure is also bounded within the given interval. The bounds work for any surrogate function that is convex and differentiable at zero with the derivative larger than zero. In the experiments, we evaluate our method and compare with existing works using the real-world datasets. The results demonstrate the correctness of the

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313723

constraint-free criterion and the superiority of our method over existing ones in terms of achieving fairness and retaining prediction accuracy.

2 FAIRNESS-AWARE CLASSIFICATION

In this section we present our fairness-aware classification framework. We first introduce the unconstrained optimization formulation for the classic classification models as proposed in [2], and then present our constrained optimization formulation for fairness-aware classification. Throughout the paper, we use the vector $\mathbf{X} \in \mathcal{X}$ to denote the features used in classification, and $Y \in \mathcal{Y} = \{-1, 1\}$ to denote the binary label. We denote the sensitive attribute by S, assuming that it is associated with two values: sensitive group s^- and non-sensitive group s^+. The training data $\mathbb{D} = \{(\mathbf{x}_i, s_i, y_i)\}_{i=1}^N$ is a sample drawn from a unknown but fixed distribution.

2.1 Classification Problem

The learning goal of classification is to find a classifier: $f : \mathcal{X} \mapsto \mathcal{Y}$ that minimizes the average of the classification loss (a.k.a the empirical loss), given by

$$\mathbb{L}(f) = \mathbb{E}_{\mathbf{X},Y}[\mathbb{1}_{f(\mathbf{x})\neq y}], \tag{1}$$

where $\mathbb{1}_{[\cdot]}$ is an indicator function. The classification problem can then be formulated as an optimization problem:

$$\min_{f\in\mathcal{F}} \mathbb{L}(f) = \min_{f\in\mathcal{F}} \mathbb{E}_{\mathbf{X},Y}[\mathbb{1}_{f(\mathbf{x})\neq y}].$$

Directly solving this optimization problem is intractable since the objective function is non-convex [2]. For efficient computation, another predictive function h is adopted which is performed in real number domain $\mathcal{R}$, i.e., $h : \mathcal{X} \mapsto \mathcal{R}$. By letting $f = \text{sign}(h)$, the empirical loss can be reformulated as

$$\begin{aligned}\mathbb{L}(f) = \mathbb{L}(h) &= \mathbb{E}_{\mathbf{X},Y}\Big[\mathbb{1}_{\text{sign}\left(h(\mathbf{x})\right)\neq y}\Big] \\ &= \mathbb{E}_{\mathbf{X}}\Big[Pr(Y=1|\mathbf{x})\mathbb{1}_{h(\mathbf{x})<0} + Pr(Y=-1|\mathbf{x})\mathbb{1}_{h(\mathbf{x})>0}\Big].\end{aligned} \tag{2}$$

If we replace the indicator function (a.k.a 0-1 loss function) with a convex surrogate function ϕ, the empirical loss can be rewritten as

$$\mathbb{L}_\phi(h) = \mathbb{E}_{\mathbf{X}}\Big[Pr(Y=1|\mathbf{x})\phi\big(h(\mathbf{x})\big) + \big(1 - Pr(Y=1|\mathbf{x})\big)\phi\big(-h(\mathbf{x})\big)\Big],$$

which is known as the ϕ-loss, and the optimization problem is reformulated as $\min_{h\in\mathcal{H}} \mathbb{L}_\phi(h)$. In the past decades, a number of surrogate loss functions have been proposed and well studied, such as the hinges loss, the square loss, the logistic loss, the exponential loss, etc..

2.2 Fairness-aware Classification Problem

The fairness-aware classification aims to find a classifier that minimizes the empirical loss while satisfying certain fairness constraints. Several fairness notions or definitions are proposed in the literature, such as demographic parity [13], mistreatment parity [17], etc..

Demographic parity is the most widely-used fairness notion in the fairness-aware learning field. It requires the decision made by the classifier is independent to the sensitive attribute, such as sex or race. Usually, demographic parity is quantified with regard to risk difference [12], i.e., the difference of the positive predictions between the sensitive group and non-sensitive group. For example, in the context of hiring, risk difference can be given by the probability difference of being predicted to be hired between male applicants and female applicants. Using the same language as that in the previous subsection, the risk difference produced by a classifier f is expressed as

$$\mathbb{RD}(f) = \mathbb{E}_{\mathbf{X}|S=s^+}[\mathbb{1}_{f(\mathbf{x})=1}] - \mathbb{E}_{\mathbf{X}|S=s^-}[\mathbb{1}_{f(\mathbf{x})=1}]. \tag{3}$$

As a quantitative metric, we say that classifier f is considered as fair if $|\mathbb{RD}(f)| \leq \tau$, where τ is the user-defined threshold. For instance, the 1975 British legislation for sex discrimination sets $\tau = 0.05$. By directly incorporating the risk difference into the optimization problem, we formulate the fair classification problem as follows.

Problem Formulation 1. *The goal of the fairness-aware classification is to find a classifier f that minimizes the loss $\mathbb{L}(f)$ while satisfying fairness constraint $|\mathbb{RD}(f)| \leq \tau$. It can be approached by solving the following constrained optimization problem*

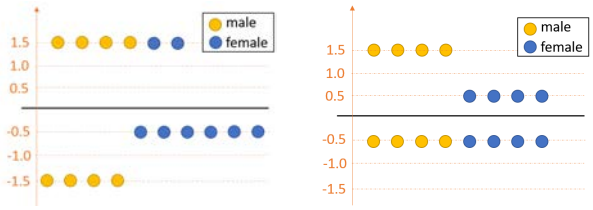
$$\begin{aligned}\min_{f\in\mathcal{F}} \quad & \mathbb{L}(f) \\ \textit{subject to} \quad & \mathbb{RD}(f) \leq \tau, \quad -\mathbb{RD}(f) \leq \tau,\end{aligned} \tag{4}$$

where $\mathbb{L}(f)$ and $\mathbb{RD}(f)$ are defined in Eq. (1) *and Eq.* (3).

Obviously, solving the above problem is computationally intractable, since both $\mathbb{L}(f)$ and $\mathbb{RD}(f)$ contain indicator functions.

The real-value function $h(\mathbf{x})$ and the surrogate functions have been proposed in the recent works [1, 8, 18, 19]. For example, Zafar et al. [19] have proposed the decision boundary covariance to quantify the fairness and serve as constraints, which is equivalent to applying the linear surrogate functions to Problem Formulation 1. They have set the constraint thresholds as c and $-c$, which specify the threshold for the covariance. However, solving the optimization problem with surrogated constraints does not necessarily result in a fair classifier in terms of the original non-convex fairness requirements, e.g., $-\tau \leq \mathbb{RD}(f) \leq \tau$. In fact, there is no any fairness guarantee on the produced classifier. We use an example to show this. Consider two margin-based classifiers where the surrogate functions are linear functions of the distance from the data point to the decision boundary. Therefore, the risk difference is computed by counting the number of data points above and below the decision boundary, and the surrogated risk difference (a.k.a the decision boundary covariance) is computed by measuring the average signed distance from the data points to the decision boundary. In the dataset shown in Figure 1a, we obtain that the surrogated risk difference is 0 but the real risk difference is 0.25. This means that a classier obtained by solving the constrained optimization problem actually can be very unfair. In the dataset shown in Figure 1b, the risk difference is 0 but the surrogated risk difference is 0.5, meaning that some fair classifiers cannot be obtained by solving the optimization problem with surrogated constraints.

The use of the surrogate function inevitably produces estimation errors and leads to the mismatch between the surrogated constraints and the original non-convex fairness constraints. Some intuitive techniques have been introduced to tune the threshold of the surrogated constraints for learning fair classifiers. For example, Zafar et al. [19] have proposed to build an unconstrained classifier and



(a) A classifier that meets the surrogated RD constraint makes unfair predictions.

(b) A classifier that doesn't meet the surrogated RD constraint makes fair predictions.

Figure 1: Two classifiers and their predictions.

consider its risk difference as the initial threshold, say c^*, then they heuristically select a factor $m \in [0, 1]$ and let the threshold $c = m \times c^*$. However, the relationship between the threshold c of the surrogated constraints and the hard threshold τ of the original metrics is unclear hence users have to repeatedly conduct experiments on the datasets.

3 CONVEX FAIRNESS CLASSIFICATION FRAMEWORK

In this section, we propose a general framework for fairness-aware classification which addresses the gap incurred by the estimation error due to the use of the surrogate function. Our framework can formulate various fairness metrics (e.g., risk difference, risk ration, equal odds, etc.) as convex constraints and incorporate them into classic classification model. In the following sections, we present our framework based on the risk difference. In the appendix, we show how our framework can be easily extended to other fairness metrics, e.g., risk ratio, equalized odds.

We first present a constraint-free criterion that is derived from the data. This criterion ensures that any classifier learned from the data are fair in terms of the specified fairness metric. Then when this criterion is satisfied, there is no need to incorporate any fairness constraints for learning fair classification. When this criterion is not met, we formulate the fairness-aware classification task as a convex optimization problem. To fill the gap between the surrogated constraints and the real fairness metrics, we derive the upper and lower bounds for the real fairness metrics and further develop refined convex constraints. If the refined constraints are satisfied, it is guaranteed that the original non-convex fairness requirements are satisfied, e.g., $-\tau \leq \mathbb{RD}(f) \leq \tau$.

3.1 Constraint-free Criterion

We propose a constraint-free criterion to determine whether the fairness constraints are necessary. As discussed in Section 2.1, the unconstrained classification problem is well studied and users can safely apply the classic methods for building a fair classifier.

We first define two special classifiers f_{max} and f_{min} which obtain the maximal and the minimal risk differences respectively.

Definition 1. *The maximal risk difference classifier $f_{\max}$ and the minimal risk difference classifier $f_{\min}$ are defined as:*

$$f_{\max}(\mathbf{x}) = \begin{cases} 1 & \text{if } \eta(\mathbf{x}) \geq p, \\ -1 & \text{otherwise,} \end{cases} \qquad f_{\min}(\mathbf{x}) = \begin{cases} -1 & \text{if } \eta(\mathbf{x}) \geq p, \\ 1 & \text{otherwise,} \end{cases}$$

where we denote $P(S = s^+|\mathbf{x})$ by $\eta(\mathbf{x})$ and $P(S = s^+)$ by p.

These two classifiers provide the maximum and minimum of risk difference among all classifiers f out of the model space $\mathcal{F}$:

Theorem 1. *For any classifier f, it always holds that $\mathbb{RD}^- \leq \mathbb{RD}(f) \leq \mathbb{RD}^+$, where $\mathbb{RD}^- = \mathbb{RD}(f_{\min})$ and $\mathbb{RD}^+ = \mathbb{RD}(f_{\max})$.*

The proof of Theorem 1 is included in the manuscript[1] on the arXiv. From Theorem 1, we directly obtain Corollary 2.

Corollary 2. *Given the threshold τ, for a training data if we have $\mathbb{RD}^+ \leq \tau$ and $\mathbb{RD}^- \geq -\tau$, then any classifier learned from this dataset is fair in terms of risk difference.*

Given a dataset, we can always build two classifiers f_{max} and f_{min}, then compute $\mathbb{RD}^+$ and $\mathbb{RD}^-$. If Corollary 2 is satisfied, users can safely apply any classification models to build classifiers without any fairness concern.

3.2 Convex Fairness-aware Classification

When the constraint-free criterion is not satisfied, it is required to incorporate fairness constraints when learning classifiers, e.g., solving Problem Formulation 1. To this end, we adopt two different surrogate functions for converting the original problem into a convex optimization. We firstly adopt a real-value predictive function h and let $f = \text{sign}(h)$, then rewrite the risk difference as

$$\begin{aligned} &\mathbb{RD}(f) = \mathbb{RD}(h) \\ &= \mathbb{E}_{\mathbf{X}|S=s^+}\left[\mathbb{1}\left[\text{sign}\left(h(\mathbf{x})\right) = 1\right]\right] - \mathbb{E}_{\mathbf{X}|S=s^-}\left[\mathbb{1}\left[\text{sign}\left(h(\mathbf{x})\right) = 1\right]\right] \\ &= \mathbb{E}_{\mathbf{X}|S=s^+}[\mathbb{1}_{h(\mathbf{x})>0}] + \mathbb{E}_{\mathbf{X}|S=s^-}[\mathbb{1}_{h(\mathbf{x})<0}] - 1. \end{aligned}$$

It follows that

$$\begin{aligned} \mathbb{RD}(f) &= \mathbb{E}_{\mathbf{X}}\left[\frac{P(S=s^+|\mathbf{x})}{P(S=s^+)}\mathbb{1}_{h(\mathbf{x})>0} + \frac{P(S=s^-|\mathbf{x})}{P(S=s^-)}\mathbb{1}_{h(\mathbf{x})<0} - 1\right] \\ &= \mathbb{E}_{\mathbf{X}}\left[\frac{\eta(\mathbf{x})}{p}\mathbb{1}_{h(\mathbf{x})>0} + \frac{1-\eta(\mathbf{x})}{1-p}\mathbb{1}_{h(\mathbf{x})<0} - 1\right], \end{aligned} \tag{5}$$

where we denote $P(S = s^+|\mathbf{x})$ by $\eta(\mathbf{x})$ and $P(S = s^+)$ by p for simplicity, thus $P(S = s^-|\mathbf{x}) = 1 - \eta(\mathbf{x})$ and $P(S = s^-) = 1 - p$.

It is intuitive that the indicator function in above formula can be replaced with the surrogate function. The challenge here is, two constraints $\mathbb{RD}(f) \leq \tau$ and $-\mathbb{RD}(f) \leq \tau$ are opposite to each other. Thus, replacing all indicator functions with a single surrogate function will result in a convex-concave problem, where only heuristic solutions for finding the local optima are known to exist. Therefore, we adopt two surrogate functions, a convex one $\kappa(\cdot)$ and a concave one $\delta(\cdot)$, each of which replaces the indicator function for one constraint. As a result, the formulated constrained optimization problem is convex and can be efficiently solved. We call the risk difference represented by $\kappa(\cdot)$ and $\delta(\cdot)$ as the κ, δ-risk difference, denoted by $\mathbb{RD}_\kappa(h)$ and $\mathbb{RD}_\delta(h)$. Almost all commonly-used surrogate functions can be adopted for $\kappa(\cdot)$ and $\delta(\cdot)$, by performing some shift or flip. Curves of some examples for $\kappa(\cdot)$ and $\delta(\cdot)$ are shown in Figure 2.

As a result, we obtain the following convex optimization formulation for learning fair classifiers.

[1] https://arxiv.org/abs/1809.04737

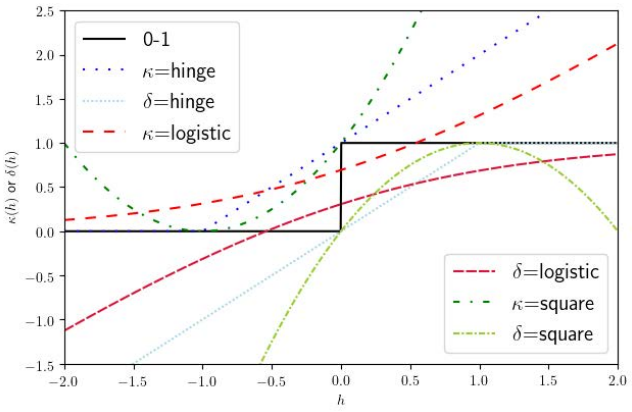


Figure 2: Curves of examples for $\kappa(\cdot)$ and $\delta(\cdot)$.

Problem Formulation 2. *The fairness-aware classification is converted into a convex optimization problem. The optimal solution h^* can be obtained by solving*

$$\begin{aligned} \min_{h\in\mathcal{H}} \quad & \mathbb{L}_\phi(h) \\ \textit{subject to} \quad & \mathbb{RD}_\kappa(h) \le c_1, \quad -\mathbb{RD}_\delta(h) \le c_2, \end{aligned}$$

where $\kappa(\cdot)$ is a convex surrogate function, $\delta(\cdot)$ is a concave surrogate function, c_1, c_2 are the thresholds of the κ, δ-risk difference, and

$$\mathbb{L}_\phi(h) = \mathbb{E}_{\mathbf{X}}\Big[Pr(Y=1|\mathbf{x})\phi\Big(h(\mathbf{x})\Big) + \Big(1-Pr(Y=1|\mathbf{x})\Big)\phi\Big(-h(\mathbf{x})\Big)\Big],$$

$$\mathbb{RD}_\kappa(h) = \mathbb{E}_{\mathbf{X}}\Big[\frac{\eta(\mathbf{x})}{p}\kappa\Big(h(\mathbf{x})\Big) + \frac{1-\eta(\mathbf{x})}{1-p}\kappa\Big(-h(\mathbf{x})\Big) - 1\Big],$$

$$\mathbb{RD}_\delta(h) = \mathbb{E}_{\mathbf{X}}\Big[\frac{\eta(\mathbf{x})}{p}\delta\Big(h(\mathbf{x})\Big) + \frac{1-\eta(\mathbf{x})}{1-p}\delta\Big(-h(\mathbf{x})\Big) - 1\Big].$$

After obtaining h^*, we build the fair classifier by letting $f^* = \text{sign}(h^*)$ and f^* is the final fair classifier. We emphasize that in Problem Formulation 2, the constraint thresholds are rewritten as c_1 and c_2 due to the difference between the surrogated constraints and the original non-convex constraints.

3.3 Refined Fairness-aware Classification

In this section, we develop the upper and lower bounds of the risk difference $\mathbb{RD}(h)$ with the κ, δ-risk difference $\mathbb{RD}_\kappa(h)$ and $\mathbb{RD}_\kappa(h)$. Based on the bounds, we present the method to derive c_1, c_2 for $\mathbb{RD}_\kappa(h), \mathbb{RD}_\delta(h)$, which provides a fairness guarantee that the solution $f^* = \text{sign}(h^*)$ to Problem Formulation 2 satisfies the fairness requirements, e.g., $-\tau \le \mathbb{RD}(f^*) \le \tau$. The method works for various types of surrogate functions (e.g., hinge, square, logistic, exponential, etc.).

We begin with defining the conditional risk difference $C^\eta\Big(h(\mathbf{x})\Big)$ for a specific subpopulation $\mathbf{x}$:

$$C^\eta\Big(h(\mathbf{x})\Big) = \frac{\eta(\mathbf{x})}{p}\mathbb{1}_{h(\mathbf{x})>0} + \frac{1-\eta(\mathbf{x})}{1-p}\mathbb{1}_{h(\mathbf{x})<0} - 1,$$

where η is the abbreviation of $\eta(\mathbf{x})$.

Then, according to Eq. (5), we have $\mathbb{RD}(f) = \mathbb{E}_{\mathbf{X}}[C^\eta\Big(h(\mathbf{x})\Big)]$. When surrogate function $\kappa(\cdot)$ (resp. $\delta(\cdot)$) is adopted, we similarly define the conditional κ-risk difference

$$C^\eta_\kappa\Big(h(\mathbf{x})\Big) = \frac{\eta(\mathbf{x})}{p}\kappa\Big(h(\mathbf{x})\Big) + \frac{1-\eta(\mathbf{x})}{1-p}\kappa\Big(-h(\mathbf{x})\Big) - 1,$$

and we have $\mathbb{RD}_\kappa(h) = \mathbb{E}_{\mathbf{X}}\Big[C^\eta_\kappa\Big(h(\mathbf{x})\Big)\Big]$.

Note that the values of $C^\eta\Big(h(\mathbf{x})\Big)$ and $C^\eta_\kappa\Big(h(\mathbf{x})\Big)$ depend on $\eta(\mathbf{x})$ and $h(\mathbf{x})$, which are determined by the subpopulation of the data specified by $\mathbf{x}$, as well as predictive function h. In order to study the general situations for any specific subpopulation and any possible predictive function, we denote $h(\mathbf{x})$ as α and define the generic conditional risk difference $C^\eta(\alpha)$ and the generic conditional κ-risk difference $C^\eta_\kappa(\alpha)$:

$$C^\eta(\alpha) = \frac{\eta}{p}\mathbb{1}_{\alpha>0} + \frac{1-\eta}{1-p}\mathbb{1}_{\alpha<0} - 1, \quad C^\eta_\kappa(\alpha) = \frac{\eta}{p}\kappa(\alpha) + \frac{1-\eta}{1-p}\kappa(-\alpha) - 1,$$

for any $\eta \in [0,1]$ and $\alpha \in \mathcal{R}$. Then, the minimal conditional risk difference $H^-(\eta)$ and the minimal conditional κ-risk difference $H^-_\kappa(\eta)$ for any arbitrary subpopulation and any possible predictive function are given by

$$H^-(\eta) = \min_{\alpha\in R} C^\eta(\alpha) = \min_{\alpha\in R}\Big[\frac{\eta}{p}\mathbb{1}_{\alpha>0} + \frac{1-\eta}{1-p}\mathbb{1}_{\alpha<0} - 1\Big],$$

$$H^-_\kappa(\eta) = \min_{\alpha\in R} C^\eta_\kappa(\alpha) = \min_{\alpha\in R}\Big[\frac{\eta}{p}\kappa(\alpha) + \frac{1-\eta}{1-p}\kappa(-\alpha) - 1\Big]. \quad (6)$$

It is straightforward that the minimal risk difference $\mathbb{RD}^-$ is equivalent to the expectation of $H^-(\eta(\mathbf{x}))$ since for any possible $\mathbf{x}$, $H^-(\eta(\mathbf{x}))$ provides the minimal conditional risk difference. Similarly, the minimal κ-risk difference achieved by any predictive function (denoted by $\mathbb{RD}^-_\kappa$) is the expectation of $H^-_\kappa(\eta(\mathbf{x}))$, as given by

$$\mathbb{RD}^-_\kappa = \mathbb{E}_{\mathbf{X}}\Big[H^-_\kappa(\eta(\mathbf{x}))\Big].$$

Finally, we define the minimal conditional κ-risk difference within interval α s.t. $\alpha(\eta - p) \ge 0$:

$$H^\circ_\kappa(\eta) = \min_{\alpha:\alpha(\eta-p)\ge 0} C^\eta_\kappa(\alpha). \quad (7)$$

We similarly define $H^+(\eta)$ the maximal conditional risk difference, $H^+_\delta(\eta)$ the maximal conditional δ-risk difference, $\mathbb{RD}^+_\delta$ the maximal δ-risk difference, as well as $H^\circ_\delta(\eta)$ the minimal conditional δ-risk difference within interval α s.t. $\alpha(\eta - p) \ge 0$.

Now, we are able to present our results, which are given in Theorem 3 and Corollary 4. The proof can be found in the manuscript[2] on the arXiv.

Theorem 3. *If $\kappa(\cdot)$ is convex and differentiable at zero with $\kappa'(0) > 0$, $\delta(\cdot)$ is concave and differentiable at zero with $\delta'(0) > 0$, then for any predictive function h, we have*

$$\begin{aligned} \psi_\kappa(\mathbb{RD}(h) - \mathbb{RD}^-) &\le \mathbb{RD}_\kappa(h) - \mathbb{RD}^-_\kappa, \\ \psi_\delta(\mathbb{RD}^+ - \mathbb{RD}(h)) &\le \mathbb{RD}^+_\delta - \mathbb{RD}_\delta(h), \end{aligned} \quad (8)$$

where

$$\psi_\kappa(\mu) = H^\circ_\kappa\Big(p(1-p)\mu + p\Big) - H^-_\kappa\Big(p(1-p)\mu + p\Big),$$

$$\psi_\delta(\mu) = H^+_\delta\Big(p(1-p)\mu + p\Big) - H^\circ_\delta\Big(p(1-p)\mu + p\Big).$$

[2] https://arxiv.org/abs/1809.04737

In Theorem 3, $\psi_\kappa(\mu)$ and $\psi_\delta(\mu)$ are directly derived from the surrogate function κ and δ. Some commonly-used surrogate functions κ, δ and their corresponding ψ_κ, ψ_δ functions are listed in Table 1. The inequalities in Theorem 3 bound the difference between $\mathbb{RD}(h)$ and $\mathbb{RD}^+, \mathbb{RD}^-$ by the differences $\mathbb{RD}_\kappa(h) - \mathbb{RD}_\kappa^-$ and $\mathbb{RD}_\delta^+ - \mathbb{RD}_\delta(h)$. Since $\mathbb{RD}^-, \mathbb{RD}^+, \mathbb{RD}_\kappa^-, \mathbb{RD}_\delta^+$ can be computed from the dataset, we connect the original non-convex constraints and the surrogated convex constraints.

We reformulate Theorem 3 and explicitly give the upper and lower bounds of $\mathbb{RD}(h)$ in Corollary 4.

Corollary 4. *For any predictive function h, let classifier $f = sign(h)$, if $\kappa(\cdot)$ is convex and differentiable at zero with $\kappa'(0) > 0$, $\delta(\cdot)$ is concave and differentiable at zero with $\delta'(0) > 0$, then risk difference $\mathbb{RD}(f)$ is bounded by following inequalities:*

$$\mathbb{RD}(f) \le \mathbb{RD}^- + {\psi_\kappa}^{-1}\Big(\mathbb{RD}_\kappa(h) - \mathbb{RD}_\kappa^-\Big),$$
$$\mathbb{RD}(f) \ge \mathbb{RD}^+ - {\psi_\delta}^{-1}\Big(\mathbb{RD}_\delta^+ - \mathbb{RD}_\delta(h)\Big).$$

Based on the upper and lower bounds of $\mathbb{RD}(f)$, we can derive the thresholds c_1, c_2 for the surrogated constraints in Problem Formulation 2. For example, if we aim to obtain a classifier f such that $-\tau \le \mathbb{RD}(f) \le \tau$, we only require the upper bound of $\mathbb{RD}(f)$ is smaller than τ and the lower bound is larger than $-\tau$. That is:

$$\mathbb{RD}^- + {\psi_\kappa}^{-1}\Big(\mathbb{RD}_\kappa(h) - \mathbb{RD}_\kappa^-\Big) \le \tau,$$
$$\mathbb{RD}^+ - {\psi_\delta}^{-1}\Big(\mathbb{RD}_\delta^+ - \mathbb{RD}_\delta(h) \ge -\tau.$$

Thus, we obtain the refined constraints and if the refined constraints are satisfied, the original risk difference requirements are guaranteed to be satisfied.

We modify Problem Formulation 2 to obtain Problem Formulation 3 with refined fairness constraints which guarantee the real non-convex fairness requirement.

Problem Formulation 3. *A classifier $f^* = sign(h^*)$ that achieves fairness guarantee $-\tau \le \mathbb{RD}(f) \le \tau$ can be obtained by solving the following constrained optimization*

$$\min_{h \in \mathcal{H}} \quad \mathbb{L}_\phi(h) \tag{9}$$
$$\textit{subject to} \quad \mathbb{RD}_\kappa(h) \le \psi_\kappa(\tau - \mathbb{RD}^-) + \mathbb{RD}_\kappa^-,$$
$$-\mathbb{RD}_\delta(h) \le \psi_\delta(-\tau + \mathbb{RD}^+) + \mathbb{RD}_\delta^+.$$

Note that the right-hand sides of above two inequalities are constants for a given dataset. Therefore, the constrained optimization problem is still convex. We can optimally solve this problem and the solution $f^* = sign(h^*)$ is guaranteed to satisfy $-\tau \le \mathbb{RD}(f^*) \le \tau$.

Table 1: Some common surrogate functions for κ-δ and the corresponding $\psi_\kappa(\mu)$ and $\psi_\delta(\mu)$.

Name of κ-δ	$\kappa(\alpha)$ for $\alpha \in \mathbf{R}$	$\delta(\alpha)$ for $\alpha \in \mathbf{R}$	$\psi_\kappa(\mu)$ or $\psi_\delta(\mu)$ for $\mu \in (0, 1/p]$
Hinge	$\max\{\alpha+1, 0\}$	$\min\{\alpha, 1\}$	μ
Square	$(\alpha+1)^2$	$1-(1-\alpha)^2$	μ^2
Exponential	$\exp(\alpha)$	$1-\exp(-\alpha)$	$(\sqrt{(1-p)\mu+1} - \sqrt{1-p\mu})^2$

4 EXPERIMENTS

4.1 Experimental Setup

Dataset. In the experiments we use two datasets: Adult and Dutch. The Adult dataset [9] contains a total of 48,842 instances, each of which is characterized by 14 attributes (e.g., `sex`, `age`, `work_class`, `education`, `income`, etc.). We consider `sex` as the sensitive attribute with two values, `male` and `female`. Then, we consider `income` as the class label. The Dutch dataset [23] contains a total of 60,420 instances, each of which is characterized by 12 attributes. Similarly, we use `sex` as the sensitive attribute, and `occupation` as the class label.

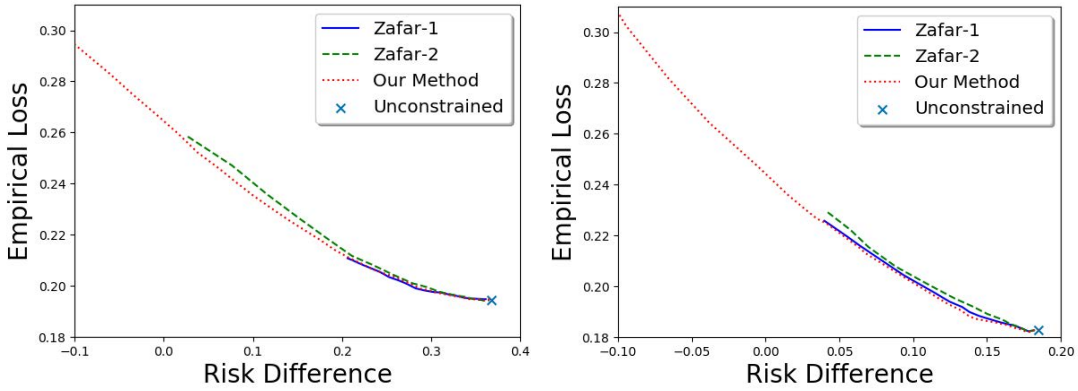
Baseline. We compare our method with two related works, referred to as Zafar-1 [18] and Zafar-2 [17], both of which formulate the fairness-aware classification problem as constrained optimization problems. In [18], the authors quantify fairness using the covariance between the users' sensitive attribute and the signed distance from the feature vectors to the decision boundary. The fairness constraint is formulated as covariance $\le m \times c^*$, where c^* is the measured fairness of the unconstrained optimal classifier and m is a multiplication factor $\in [0, 1]$. In [17], the fairness is quantified similarly with the distance function being replaced with a convex non-linear function. As a result, the obtained problem is a convex-concave optimization problem. In the experiments, we adopt the Disciplined Convex-Concave Programming (DCCP) [15] as proposed in [17] for solving the convex-concave optimization problem. For our method and Zafar-1, the convex optimization problem is solved using CVXPY [3]. The datasets and implementation are available at http://tiny.cc/fair-classification.

4.2 Constraint-free Criterion of Ensuring Fairness

To demonstrate the sufficiency criterion of learning fair classifiers, we build the maximal/minimal risk difference classifiers $f_{\min}, f_{\max}$ for both Adult and Dutch datasets, and measure the risk differences they produce, i.e., $\mathbb{RD}^-$ and $\mathbb{RD}^+$. The results are shown in the first two rows in Table 2. As can be seen, in both datasets we have large maximal and minimal risk differences. In order to evaluate a situation with small a risk difference, we also create a variant of Adult, referred to as Adult*, where all attributes are binarized and the sensitive attribute `sex` is shuffled to incur a small risk difference. Then, we build a number of classifiers including Linear Regression (LR), Support Vector Machine (SVM) with linear kernel, Decision Tree (DT), and Naive Bayes (NB), using the three datasets as the training data with with 5-fold cross-validation. After that, their risk differences are quantified on the testing data, as shown in the last four rows in Table 2. We can see that all values are within $\mathbb{RD}^-, \mathbb{RD}^+$ which are consistent with our constraint-free criterion.

4.3 Learning Fair Classifiers

We build our fair classifiers on both Adult and Dutch datasets by solving the optimization problem defined in Problem Formulation 2. For surrogate functions, we use the logistic function for $\phi(\cdot)$, and the hinge function for $\kappa(\cdot)$ and $\delta(\cdot)$. We also compare our methods with Zafar-1 and Zafar-2. The results are shown in Figure 3, which



(a) Adult **(b) Dutch**

Figure 3: Comparison of fair classifiers.

Table 2: $\mathbb{RD}^+$, $\mathbb{RD}^-$ and risk differences of Linear Regression (LR), Support Vector Machine (SVM), Decision Tree (DT), and Naive Bayes (NB).

$\mathbb{RD}(\cdot)$	Adult	Dutch	Adult*
$\mathbb{RD}^+$	0.967	0.516	0.046
$\mathbb{RD}^-$	-0.967	-0.516	-0.046
LR	0.371	0.185	0.000
SVM	0.434	0.156	0.001
DT	0.316	0.184	0.001
NB	0.447	0.144	0.001

depict the relationship between the obtained risk difference and empirical loss. For our method, different risk differences are obtained by adjusting relax terms c_1 and c_2, while for Zafar-1 and Zafar-2 different risk differences are obtained by adjusting the multiplication factor m. As can be seen, our method can achieve much smaller risk difference than Zafar-1 and Zafar-2. This may be because Zafar-1 linear functions to formulate the fairness constraints, which may incur large estimation errors; while Zafar-2 formulates a convex-concave optimization problem, where only the local optima can be reached. For the same reason, we can observe that our method produces better empirical loss than Zafar-2 given any same risk difference.

5 CONCLUSIONS

In this paper, we studied the fairness-aware classification problem and formulated it as the constrained optimization problem. We proposed a general framework which addresses all limitations of previous works in terms of: (1) various fairness metrics can be incorporated into classic classification models as constraints; (2) the formulated constrained optimization problem is convex and can be solved efficiently; and (3) the lower and upper bounds of real fairness measures are established using surrogate functions, which provide a fairness guarantee for our framework. Within the framework, we proposed a constraint-free criterion under which the learned classifier is guaranteed to be fair in terms of the specified fairness metric, as well as developed the method for learning fair classifiers if the constraint-free criterion fails to satisfy. The results demonstrate the correctness of the constraint-free criterion and the superiority of our method over existing ones in terms of achieving fairness and retaining prediction accuracy.

ACKNOWLEDGMENTS

This work was supported in part by NSF 1646654.

A OTHER FAIRNESS NOTIONS

Risk ratio is a common fairness notion [11, 14]. It also requires the decision is independent with the protected attribute. Different with the risk difference, the unfairness is quantified by the ratio of the positive decisions between the non-protected group and the protected group. Let's formalize the risk ratio $\mathbb{RR}(h)$ of classifier h:

$$\mathbb{RR}(h) = \frac{\mathbb{E}_{\mathbf{X}|S=s^+}\left[\mathbb{1}_{h(\mathbf{x})>0}\right]}{\mathbb{E}_{\mathbf{X}|S=s^-}\left[\mathbb{1}_{h(\mathbf{x})>0}\right]}.$$

The fairness constraints with regards to risk ratio could be expressed as

$$\mathbb{RR}(h) = \frac{\mathbb{E}_{\mathbf{X}|S=s^+}\left[\mathbb{1}_{h(\mathbf{x})>0}\right]}{\mathbb{E}_{\mathbf{X}|S=s^-}\left[\mathbb{1}_{h(\mathbf{x})>0}\right]} \leq \tau.$$

Similar to Eq. (5), we express the constraints as

$$\mathbb{E}_{\mathbf{X}}\left[\frac{\eta}{p}\mathbb{1}_{h(\mathbf{x})>0} + \tau\frac{1-\eta(\mathbf{x})}{1-p}\mathbb{1}_{h(\mathbf{x})>0}\right] - \tau \leq 0. \tag{10}$$

Equalized odds and equalized opportunity are proposed by Hardt et al. [6]. Equalized odds requires the protected attribute and the predicted label are independent conditional on the truth label. To quantify the strength of equalized odds, we simply propose the prediction difference between two groups conditional on the truth label. So, the equalized odds is

$$\mathbb{EO}(h) = \mathbb{E}_{\mathbf{X}|S=s^+,Y}[\mathbb{1}_{h(\mathbf{x})>0}] - \mathbb{E}_{\mathbf{X}|S=s^-,Y}[\mathbb{1}_{h(\mathbf{x})>0}].$$

Similarly, a classifier h is considered as fair with regard to equalized odds if $\mathbb{EO}(h) \leq \tau$.

Let's reformulate the equalized odds constraints:

$$\begin{aligned}\mathbb{EO}(h) &= \mathbb{E}_{\mathbf{X}|S=s^+,Y}[\mathbb{1}_{h(\mathbf{x})>0}] + \mathbb{E}_{\mathbf{X}|S=s^-,Y}[\mathbb{1}_{h(\mathbf{x})<0}] - 1 \\ &= \mathbb{E}_{\mathbf{X}|Y}\left[\frac{P(S=s^+|\mathbf{x},y)}{P(S=s^+|y)}\mathbb{1}_{h(\mathbf{x})>0} + \frac{1-P(S=s^+|\mathbf{x},y)}{1-P(S=s^+|y)}\mathbb{1}_{h(\mathbf{x})<0}\right] - 1 \leq \tau.\end{aligned} \tag{11}$$

Equalized opportunity is a relaxation of equalized odds where only the positive group ($Y = 1$) is taken into account:

$$\mathbb{EOP}(h) = \mathbb{E}_{\mathbf{X}|Y=1}\Big[\frac{P(S = s^+|\mathbf{x}, Y = 1)}{P(S = s^+|Y = 1)}\mathbb{1}_{h(\mathbf{x})>0} + \frac{1 - P(S = s^+|\mathbf{x}, Y = 1)}{1 - P(S = s^+|Y = 1)}\mathbb{1}_{h(\mathbf{x})<0}\Big] - 1 \leq \tau. \quad (12)$$

By simply replacing the indicator functions with surrogate functions, we can readily extend our framework to the constraints (10), (11), (12) with regard to the three notions. Our criterion and bounds are also extensible to the three notions.

REFERENCES

[1] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, and John Langford. 2017. A Reductions Approach to Fair Classification. In *Conference on Fairness, Accountability, and Transparency in Machine Learning*.

[2] Peter L Bartlett, Michael I Jordan, and Jon D. McAuliffe. 2006. Convexity, Classification, and Risk Bounds. *J. Amer. Statist. Assoc.* 101, 473 (mar 2006), 138–156. https://doi.org/10.1198/016214505000000907

[3] Steven Diamond and Stephen Boyd. 2016. CVXPY: A Python-Embedded Modeling Language for Convex Optimization. *Journal of Machine Learning Research* 17, 83 (2016), 1–5.

[4] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In *KDD '15*. ACM Press. https://doi.org/10.1145/2783258.2783311

[5] Gabriel Goh, Andrew Cotter, Maya Gupta, and Michael P Friedlander. 2016. Satisfying real-world goals with dataset constraints. In *NIPS'16*. 2415–2423.

[6] Moritz Hardt, Eric Price, Nati Srebro, et al. 2016. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*. 3315–3323.

[7] Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. 2011. Fairness-aware Learning through Regularization Approach. In *ICDMW'11*. IEEE, 643–650. https://doi.org/10.1109/ICDMW.2011.83

[8] Aditya Krishna and Robert C Williamson. 2018. The Cost of Fairness in Binary Classification. *Proceedings of Machine Learning Research* 81 (2018), 1–12. http://proceedings.mlr.press/v81/menon18a.html

[9] M Lichman. 2013. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[10] Mahbod Olfat and Anil Aswani. 2018. Spectral Algorithms for Computing Fair Support Vector Machines. In *International Conference on Artificial Intelligence and Statistics*. 1933–1942.

[11] Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. 2009. Measuring Discrimination in Socially-Sensitive Decision Records. In *Proceedings of the 2009 SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 581–592. https://doi.org/10.1137/1.9781611972795.50

[12] Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. 2012. A study of top-k measures for discrimination discovery. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing - SAC '12*. ACM Press, New York, New York, USA, 126. https://doi.org/10.1145/2245276.2245303

[13] Dino Pedreshi, Salvatore Ruggieri, and Franco Turini. 2008. Discrimination-aware data mining. In *KDD 08*. ACM Press, New York, New York, USA, 560. https://doi.org/10.1145/1401890.1401959

[14] Andrea Romei and Salvatore Ruggieri. 2014. A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review* 29, 05 (nov 2014), 582–638. https://doi.org/10.1017/S0269888913000039

[15] Xinyue Shen, Steven Diamond, Yuantao Gu, and Stephen Boyd. 2016. Disciplined Convex-Concave Programming. Cdc (2016), 1009–1014. https://doi.org/10.1109/CDC.2016.7798400

[16] Blake Woodworth, Suriya Gunasekar, Mesrob I Ohannessian, and Nathan Srebro. 2017. Learning Non-Discriminatory Predictors. In *Conference on Learning Theory*. 1920–1953.

[17] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. 2017. Fairness Beyond Disparate Treatment & Disparate Impact. In *WWW '17*. ACM Press, New York, NY, USA, 1171–1180. https://doi.org/10.1145/3038912.3052660

[18] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. 2017. Fairness Constraints: Mechanisms for Fair Classification. In *Artificial Intelligence and Statistics*. Fort Lauderdale, Florida, USA.

[19] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, Krishna P Gummadi, and Adrian Weller. 2017. From Parity to Preference-based Notions of Fairness in Classification. (jun 2017).

[20] Lu Zhang and Xintao Wu. 2017. Anti-discrimination learning: a causal modeling-based framework. *International Journal of Data Science and Analytics* 4, 1 (aug 2017), 1–16. https://doi.org/10.1007/s41060-017-0058-x

[21] Lu Zhang, Yongkai Wu, and Xintao Wu. 2017. A Causal Framework for Discovering and Removing Direct and Indirect Discrimination. In *IJCAI '17*. California, 3929–3935. https://doi.org/10.24963/ijcai.2017/549

[22] Lu Zhang, Yongkai Wu, and Xintao Wu. 2018. Achieving Non-Discrimination in Prediction. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. 3097–3103. https://doi.org/10.24963/ijcai.2018/430

[23] Indre Zliobaite, Faisal Kamiran, and Toon Calders. 2011. Handling Conditional Discrimination. In *2011 IEEE 11th International Conference on Data Mining*. IEEE, 992–1001. https://doi.org/10.1109/ICDM.2011.72

Understanding Urban Dynamics via State-sharing Hidden Markov Model

Tong Xia[1*], Yue Yu[1*], Fengli Xu[1], Funing Sun[2], Diansheng Guo[2], Depeng Jin[1], Yong Li[1]
[1]Beijing National Research Center for Information Science and Technology,
Department of Electronic Engineering, Tsinghua University, Beijing, China
[2]Tencent Inc., Beijing, China
liyong07@tsinghua.edu.cn

ABSTRACT

Modeling people's activities in the urban space is a crucial socio-economic task but extremely challenging due to the deficiency of suitable methods. To model the temporal dynamics of human activities concisely and specifically, we present State-sharing Hidden Markov Model (SSHMM). First, it extracts the urban states from the whole city, which captures the volume of population flows as well as the frequency of each type of Point of Interests (PoIs) visited. Second, it characterizes the urban dynamics of each urban region as the state transition on the shared-states, which reveals distinct daily rhythms of urban activities. We evaluate our method via a large-scale real-life mobility dataset and results demonstrate that SSHMM learns semantics-rich urban dynamics, which are highly correlated with the functions of the region. Besides, it recovers the urban dynamics in different time slots with an error of 0.0793, which outperforms the general HMM by 54.2%.

CCS CONCEPTS

• **Information systems** → **Spatial-temporal systems**; **Data mining**; • **Human-centered computing** → *Empirical studies in ubiquitous and mobile computing*.

KEYWORDS

Urban Dynamics; Mobility; Hidden Markov Model

ACM Reference Format:
Tong Xia[1*], Yue Yu[1*], Fengli Xu[1], Funing Sun[2], Diansheng Guo[2], Depeng Jin[1], Yong Li[1]. 2019. Understanding Urban Dynamics via State-sharing Hidden Markov Model. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313453

1 INTRODUCTION

The rapid urbanization process makes the topic of understanding urban dynamics, i.e. the temporal patterns of urban activities becomes increasingly important. However, traditional approaches rely on expensive manual surveys, yet the understanding is still coarse-grained and limited in geographical scope [1]. Fortunately, the advent of the mobile Internet and Location-Based Social Networks (LBSNs) makes it possible to collect population-scale urban mobility data, which sheds new light on this open problem. Previous works demonstrated that the movements of citizens can be utilized to infer the functions of urban regions [20, 22], and urban activities (e.g., working, resting, etc.) are closely correlated with urban mobility patterns [19, 23], which indicates the feasibility of leveraging urban mobility data to model urban dynamics.

In this paper, we aim to harness the power of massive urban mobility data to deepen the understanding of urban dynamics. The research problem is non-trivial mainly for three reasons: (1) Urban mobility behaviour is a *noisy* representation of urban activities. Therefore, it is difficult to robustly and accurately infer the underlying urban activities from the empirical observation of urban mobility behaviour. (2) The semantic-rich mobility data, i.e., check-in data on LBSNs, is sparse in urban space, especially in sparsely populated areas which poses challenges to extract reliable patterns of urban dynamics for different regions in urban systems. (3) Modern cities are complicated socio-economic systems, where each urban region possesses different urban dynamics due to its unique activities. Therefore, it is challenging to interpret the identified urban dynamics and reveal the underlying mechanisms.

Motivated by these challenges, we propose a novel State-sharing Hidden Markov Model (SSHMM), which considers the mobility behavior of an urban region to reveal the urban dynamics. Specifically, the model learns the urban dynamics of a region as transitions between hidden states, where each state maps to a certain urban activity. The corresponding mobility behavior is generated from the hidden state, which allows the same urban activities to be mapped to slightly different mobility behavior and effectively addresses the problem of data noisy. More importantly, SSHMM facilitates different urban regions to share the same set of hidden states. Therefore, it addresses the challenge of data sparsity , which fully exploits the correlation between different regions. Finally, as a generative model, SSHMM can also characterize the urban dynamics as hidden state sequences. Based on the identified state sequences, we further design an clustering analysis technique to reveal their correlation with urban functions (i.e., land use) and provides a meaningful interpretation of the urban dynamics.

The contributions of our research are three-fold:

(1) We propose a novel urban dynamics revealing model SSHMM. It can robustly and accurately infer the underlying urban activities from noisy and sparse urban mobility data by sharing model parameters across different regions. In addition, it achieves qualitative representations of urban dynamics as the transition patterns between urban activities.

* These two authors contributed equally to this research.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313453

(2) We propose an effective and efficient algorithm to infer the parameters of our model. By splitting the long observations into shorter ones and updating the parameters in parallel, we reduce the training time of learning R groups of parameters to that of only one group, where R is the number of regions.
(3) We evaluate our method using population-scale mobility dataset, which demonstrate that our SSHMM model learns meaningful and explainable urban dynamics.

2 RELATED WORK

Urban dynamics modeling. Urban dynamics, generally defined as how sociological indicators change over time [5] which focuses on characterizing human daily activity rhythms in the city [1, 14]. Time is an important dimension to understand the city. Abbar et al. [1] built activity time series in cities to identify different dynamic patterns via the geo-tagged data from Twitter. Zhang et al. [23] utilized the geo-tagged social media to model people's activities in the urban space via representative learning. Zong et al. [27] investigated to generate dynamic population distributions from static PoIs. Lin et al. [12] considered the impact of PoIs in different time of one day to achieve better crowd flow prediction. Besides, Yuan et al. [22] proposed an LDA model to detect the existence of different functional regions in a city through the GPS trajectory and PoI datasets, and Yao et al. [20] presented a city zone embedding framework using human mobility patterns to infer urban functions. Different from the existing works based on statistical analysis [1] and data visualization [14], we propose a specific model to learn hidden state from human mobility and check-ins to represent urban dynamics.

Hidden Markov Model and its application. Hidden Markov Model (HMM) is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (i.e., hidden) states [16]. In HMM, model mixture and parameter sharing are very helpful to deal with increasingly complex tasks [6]. One of the well-known mixture methods is the Gaussian mixture model based HMM (GMM-HMM). The second example is shared-distribution HMM, where clustering is carried out for parameters sharing and output distributions are shared with each other if they exhibit acoustic similarity [8]. Another model tied-mixture HMM uses both mixture and parameter sharing, which belongs to semi-continuous HMM [2, 7, 11].

Our proposed SSHMM is also a kind of parameter sharing HMM. While different from previous works, we design it to automatically learn a set of states for the continuous observations and no following clustering is conducted to force the parameters shared. Both based on the Gaussian emission function, we use multi-dimentional Gaussian instead of GMM. In addition, though HMM has been successfully applied into the topic of mobility modeling, most of the work concentrated on individual mobility prediction [13, 24, 26], and we are the first to apply it in urban dynamics modeling.

3 OVERVIEW

3.1 Problem Description

Before formally define our investigated problem, we give the definition of mobility behaviour observation as follows,

Definition 1 (Mobility Behaviour Observation). *The mobility behaviour sequences of region r is a time-ordered sequence $O_r = [O_{r,1}, O_{r,2}, ..., O_{r,N}]$, where $O_{r,n}$ is a tuple of length L, standing for the observation in n-th time slot. It contains two parts: (1) $\{o_{r,n,1}, o_{r,n,2}, o_{r,n,3}\}$ denotes the number of people who arrive at, leave from and stay in this region in this time slot. (2) $\{o_{r,n,4}, o_{r,n,5}, ..., o_{r,n,L}\}$ denotes the check-in frequency of different categories of PoIs.*

Now, we formulate the urban dynamics understanding problem. Given the mobility behaviour observations of a group of regions in the city, we aim to (1) discover hidden states represented by the volume of population and the visited frequency of different PoIs to understand the intrinsic life modes in the city; (2) reveal urban dynamics represented by hidden state sequences to understand the daily life rhythms and their correlation with urban functions; (3) characterize dynamic regularity by learning state transition probability to achieve dynamic prediction.

3.2 Model Description

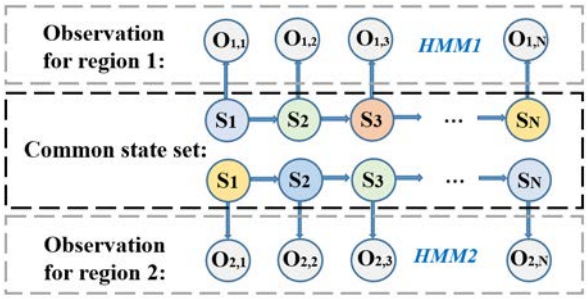


Figure 1: An illustration of the SSHMM, where two kinds of dynamics for region 1 and 2 are generated from the same set of states and each state is presented by a unique color.

Definition 2 (SSHMM). *HMM is parameterized by two parts, one is the state transition parameter characterizing how the states transit and the other is the state emission parameter characterizing how the observation generated by the state. SSHMM contains R groups of HMMs for R groups of observation sequences. However, these HMMs share the same parameters, which means all the observations are generated by the same set of hidden states.*

Our model is based on the general assumptions of HMM, where each observation $O_{r,n}$ is generated from one hidden state s_n, which has L dimensional features, and the n-th hidden state s_n merely depends on the previous state s_{n-1}. We use one HMM denoted by θ_r to model the dynamics of r-th region, therefore we need to learn R groups of transition parameters for R regions in the city. Figure 1 gives the illustration of our model. There are two observation sequences for region 1 and 2, so we learn two HMMs for them respectively. These observations are generated by the same set of states, since life modes (i.e., sleeping, working, etc.) in a city are limited and there exists some similarities between different regions with similar functions. Moreover, the massive mobility data is often sparse and mobility behaviour is a noisy representation of urban activities, which indicates that we can achieve more robust and accurate models by sharing states. These states appear in different time slots, which reveals different dynamics of these two regions.

4 METHODOLOGY

4.1 Data Pre-processing

We first adopt the map segment method [22] to obtain the geographical boundaries of each region formed by the road network. In view of the urban mobility patterns and life-styles, those regions are used as an unit to reveal the dynamics instead of simply dividing the city into grids. To share the states in the city, we normalize mobility behaviour observations to eliminate the problem regarding the difference of population between regions. For mobility, i.e., the number of arriving, leaving and staying, we directly conduct min-max normalization for each region over different time slots respectively. For check-ins, i.e., the visit frequency of PoIs, we first compute the TF-IDF [15] weights based on the region-PoI matrix in each time slot to indicate how popular a PoI category is in a certain time slot. Then, for each region we conduct the min-max normalization on the TF-IDF weights over different time slots to re-scale the mobility behaviour observations to $0-1$. After pre-processing, we regard the aggregated activities in the city as normalized time series, which is denoted by $O = \{O_1, O_2, ..., O_r, ..., O_R\}$ as **Definition 1**.

4.2 Model Definition

When generating $o_{r,n}$ from state s_n, we assume that the emission probability is Gaussian distribution, i.e., $p(o_{r,n,l}|s_n) = N(o_{r,n,l}|\mu_n, \sigma_n)$. The reason to chose the Gaussian is intuitive: μ_n describes the fundamental feature of s_n, while the corresponding observations generated by s_n is slightly different from the mean μ_n with the difference controlled by the variance σ_n. Therefore, we build the SSHMM parameterized by R groups of parameters $\theta = \{\theta_1, \theta_2, ..., \theta_r, ..., \theta_R\}$ with $\theta_r = \{\pi_r, A_r, \mu, \sigma\}$ denoted for r-th region, where
(1) $\pi_r \in \mathbb{R}^{K\times 1}$ denotes the initial distribution over K hidden states, i.e., $\pi_{r,k} = p(s_1 = k)(1 \le k \le K)$;
(2) $A_r \in \mathbb{R}^{K\times K}$ denotes the transition probabilities among the K hidden states. If $(n-1)$-th state is $s_{n-1} = j$, then the probability for n-th state s_n to be k is given by $A_{r,j,k}$, i.e.,$p(s_n = k|s_{n-1} = j) = A_{r,j,k}$;
(3) $\mu, \sigma \in \mathbb{R}^{K\times L}$ denotes the mean and variance of observation probability, i.e., $p(O_{r,n}|s_n = k) = \prod_{l=1}^{L} \frac{1}{\sqrt{2\pi\sigma_{k,l}}} \exp\left(-\frac{(o_{r,n,l}-\mu_{k,l})^2}{2\sigma_{k,l}}\right)$.

It is worth noting that we do not use the subscript r to distinguish μ, σ of different HMMs, because in SSHMM they are determined by the common state set.

4.3 Model learning

4.3.1 Parameter Inference. To infer the parameters, we use Expectation-Maximization method (EM) [4] as the solution. Since SSHMM shares the same set of hidden states, the existing famous Baum-Welch algorithm [18] cannot be applied directly. The insight of EM method is to maximize log likelihood, parameters can be updated by maximizing $Q(\theta_r, \theta_r^t)$ step by step. Therefore, we derive the Q-function for all regions as Eq.1, and we can maximize it to enable the log likelihood converge to its maximum.

$$\begin{aligned} Q(\theta, \theta^t) = &\sum_{r=1}^{R}\sum_{S} p(S|O_r, \theta_r^t) \ln \pi_{r,k} + \sum_{r=1}^{R}\sum_{S}\sum_{n=1}^{N-1} p(S|O_r, \theta_r^t) \ln p(s_{n+1}|s_n) \\ &+ \sum_{r=1}^{R}\sum_{S}\sum_{n=1}^{N} p(S|O_r, \theta_r^t) \ln p(O_{r,n}|s_{r,n}), \end{aligned} \tag{1}$$

where $p(O_r|S, \theta_r^t)$ can be calculated as follows,

$$\begin{aligned} p(O_r|S, \theta_r^t) = &\pi_{r,k} p(O_{r,1}|s_{r,1}) \cdot p(s_{r,2}|s_{r,1}) p(O_{r,2}|s_{r,2}) \\ &\cdots (s_{r,N}|s_{r,N-1}) p(O_{r,N}|s_{r,N}). \end{aligned} \tag{2}$$

From Eq. 1, we observe that optimization parameters π, A (i.e., $p(s_{n+1}|s_n)$) and $\{\mu, \sigma\}$ (i.e., $p(O_{r,n}|s_{r,n})$) appear independently, so we maximize each item separately. Consequently, the forward distribution $\alpha(s_{r,n})$ and backward distribution $\beta(s_{r,n})$ are defined as

$$\begin{aligned} \alpha(s_{r,n}) &= p(O_{r,n}|s_{r,n}) \sum_{s_{r,n-1}} \alpha(s_{r,n-1}) p(s_{r,n}|s_{r,n-1}), \\ \beta(s_{r,n}) &= \sum_{s_{r,n+1}} \beta(s_{r,n+1}) p(O_{r,n+1}|s_{r,n+1}) p(s_{r,n+1}|s_{r,n}), \end{aligned} \tag{3}$$

where $\alpha(s_{r,1}) = \pi_{r,k} p(O_{r,1}|s_{r,1} = k)$ and $\beta(s_{r,N} = k) = 1$. Then, two probability can be derived from α and β as follows,

$$\begin{aligned} \gamma(s_{r,n}) &= p(s_{r,n}|O_r) = \alpha(s_{r,n})\beta(s_{r,n})/p(O_r), \\ \xi(s_{r,n}, s_{r,n+1}) &= p(s_{r,n_1}, s_{r,n}|O_r) \\ &= \alpha(s_{r,n-1}) p(s_{r,n}|s_{r,n-1}) P(O_{r,n}|s_{r,n}) \beta(s_{r,n})/p(O_r), \end{aligned} \tag{4}$$

where $p(O_r) = \sum_{s_{r,N}} \alpha(s_{r,n})$. Based on Eq. 3 and 4, we can thus optimize the parameters π, A and $\{\mu, \sigma\}$ separately as follows.

$$\begin{aligned} \pi_{r,k}^{(t+1)} &= \gamma(s_{r,1}^k), \\ A_{r,j,k}^{(t+1)} &= \frac{1}{\Xi_j} \sum_{n=2}^{N} \xi(s_{r,n-1}^j, s_{r,n}^k), \\ \mu_{k,l}^{(t+1)} &= \frac{1}{\Gamma_K} \sum_{r=1}^{R}\sum_{n=1}^{N} \gamma(s_{r,n}^k) o_{r,n,l}, \\ \sigma_{k,l}^{(t+1)} &= \frac{1}{\Gamma_K} \sum_{r=1}^{R}\sum_{n=1}^{N} \gamma(s_{r,n}^k)(o_{r,n,l} - \mu_{k,l}^{(t+1)})^2, \end{aligned} \tag{5}$$

where $\Gamma_K = \sum_{r=1}^{R}\sum_{n=1}^{N} \gamma(s_{r,n}^k)$, $\Xi_j = \sum_{n=2}^{N}\sum_{i=1}^{K} \xi(s_{r,n-1}^j, s_{r,n}^i)$.

4.3.2 Learning algorithm. Another difficulty is that the process could be inefficient when applied into large-scale datasets due to two aspects. First, the large dataset with a long period N leads the computation of α and β complicated as Eq. 3, and the result may even cause the float number exceeding. Second, the time complexity of parameter inference is $O(RNLK^2)$. which is mainly costed by the calculation of ξ as Eq. 4. The computational complexity of which is quadratic in K, rendering it inefficient for large K. However, if the time period is short or K is not large enough, the model is unable to capture all the dynamic patterns in the city.

To overcome these problems, we first split the long observation sequences into several shorter sub-sequences for each round. By doing this, we not only utilize all the data for parameter learning but also avoid float-point number exceeding. Second, from Eq. 5, we can observe that the updating of $\pi_{r,k}$ and $A_{r,j,k}$ for region r is independent. Thus, we update them in parallel and cuts down the time complexity to $O(NLK^2)$ approximately. The detailed procedure is shown in Algorithm 1. In each iteration, we run EM-steps for W rounds, and in each round, we fed the subsequences of length N into the model. After calculating the $\alpha(s_{r,n})^{(t+1)}$, $\beta(s_{r,n})^{(t+1)}$, $\gamma(s_{r,n})^{(t+1)}$, $\xi(s_{r,n})^{(t+1)}$, $\pi_{r,k}^{(t+1)}$ and $A_{r,j,k}^{(t+1)}$ **in parallel** for all the regions in each round, we update the state parameters $\mu_{k,l}^{(t+1)}$, $\sigma_{k,l}^{(t+1)}$

until the convergence is realized. The code is made publicly available [1].

Algorithm 1: SSHMM parameter learning.

Input: Observations $O = \{O_1, O_2, \ldots, O_R\}$, Maximum Iterations $MaxIter$;
Output: $\theta_r = \{\pi_r, A_r, \mu, \sigma\}$ $\forall 1 \le r \le R$;
Procedure:
  Initialization: $t = 0$, initial $\pi^{(0)}_{r,k} = 1/K$, $A^{(0)}_{r,j,k} = 1/K$, $\mu^{(0)}_{k,l} = random(0, 1)$, $\sigma^{(0)}_{k,l} = random(0, 0.1)$, $\forall 1 \le j, k \le K, 1 \le l \le L$.
  while $t < MaxIter$ **do**
    for $w = 1, 2, \ldots, W$ **do**
      E-step: $\forall 1 \le r \le R$, calculate $\alpha(s_{r,n})^{(t+1)}$, $\beta(s_{r,n})^{(t+1)}$, $\gamma(s_{r,n})^{(t+1)}$, $\xi(s_{r,n})^{(t+1)}$ **in parallel** based on old parameters $\theta^{(t)}_r$ utilizing the $w\text{-}th$ subsequence of O_r.
      M-step: $\forall 1 \le r \le R$, update $\pi^{(t+1)}_{r,k}$, $A^{(t+1)}_{r,j,k}$ **in parallel**. Update $\mu^{(t+1)}_{k,l}$, $\sigma^{(t+1)}_{k,l}$ utilizing $\gamma(s_{r,n})^{(t+1)}$, $\xi(s_{r,n})^{(t+1)}$ $\forall 1 \le r \le R$.
    Update t: $t = t + 1$

4.4 Dynamics Revealing and Functions Inferring

With the model $\theta_r = \{\pi_r, A_r, \mu, \sigma\}$ obtained from Algorithm 1, the state sequences can be decoded by Viterbi algorithm [17], which can represent the dynamics of that state. Recall that regions with similar dynamic patterns could have similar functions, we match the dynamics with the functions by clustering the state sequence of different regions via k-medoids algorithm [9]. In k-medoids algorithm, we define the distance of two sequences as the sum of the Euclidean distance of the mean of the two corresponding states. Then, the distance between states is defined as the Euclidean distance between the mean value vector of Gaussian distribution of these two states. Finally, the distance between state sequences S_i, S_j is the average distance of all the corresponding state sequences.

We adopt *Davies-Bouldin index* (DBI) [3] to determine the number of clusters, which reflects the ratio between inter-cluster distance and inter-cluster distance. A smaller DBI usually indicates a more effective division. Finally, similar dynamics in each cluster would present the same kind of function in the city.

4.5 Dynamics Prediction

SSHMM also has the ability of prediction. Through Viterbi algorithm, we can identify the last state of the region based on the observation. With the advantage of a probabilistic model, we can firstly obtain the current state of the region according to the current observation, then predict the next state according to the latest state by maximizing the transition probability. Formally, given the current observation sequence of $r\text{-}th$ region denoted by $O_{r,1}O_{r,2}, ..., O_{r,n}$, we decode its corresponding hidden state sequence as $s_{r,1}s_{r,2}, ..., s_{r,n}$. If the latest state $s_{r,n}$ is $i\text{-}th$ state, then the state in the next time slot can be predicted as $s_{r,n+1} = \arg\max_{1 \le j \le K} A_{r,i,j}$. This is to say, we can predict the volume of population flows and the percentage of PoIs visited in the next time slot by utilizing the mean value μ, σ of the identified next state.

[1] https://github.com/XTxiatong/SSHMM

5 EXPERIMENTS

5.1 Data

The mobility dataset was collected by collaborating with *Tencent*[2], one of the largest Internet service providers in China. The collected dataset covers near 2 million users in Beijing, China from April 1st to 30th, 2018. Each trajectory record is characterized by an anonymized user ID, timestamp and the GPS location. Each check-in record consists of user ID, check-in time, check-in location and PoI, which is divided into nine categories: *Company, Agency, Shopping, Service, Entertainment, Attractions, Education* and *Residence*. We crawl the road network from Map service and divide the the downtown area in Beijing into 665 non-overlapping regions.

Ethics. To protect user privacy, all data is anonymous and stored in Tencent offline servers. We pre-process the data under their supervision and only take the aggregated results of different regions for further analysis. Our research has been reviewed and approved by both the company and Tsinghua University institutional board.

5.2 System Settings

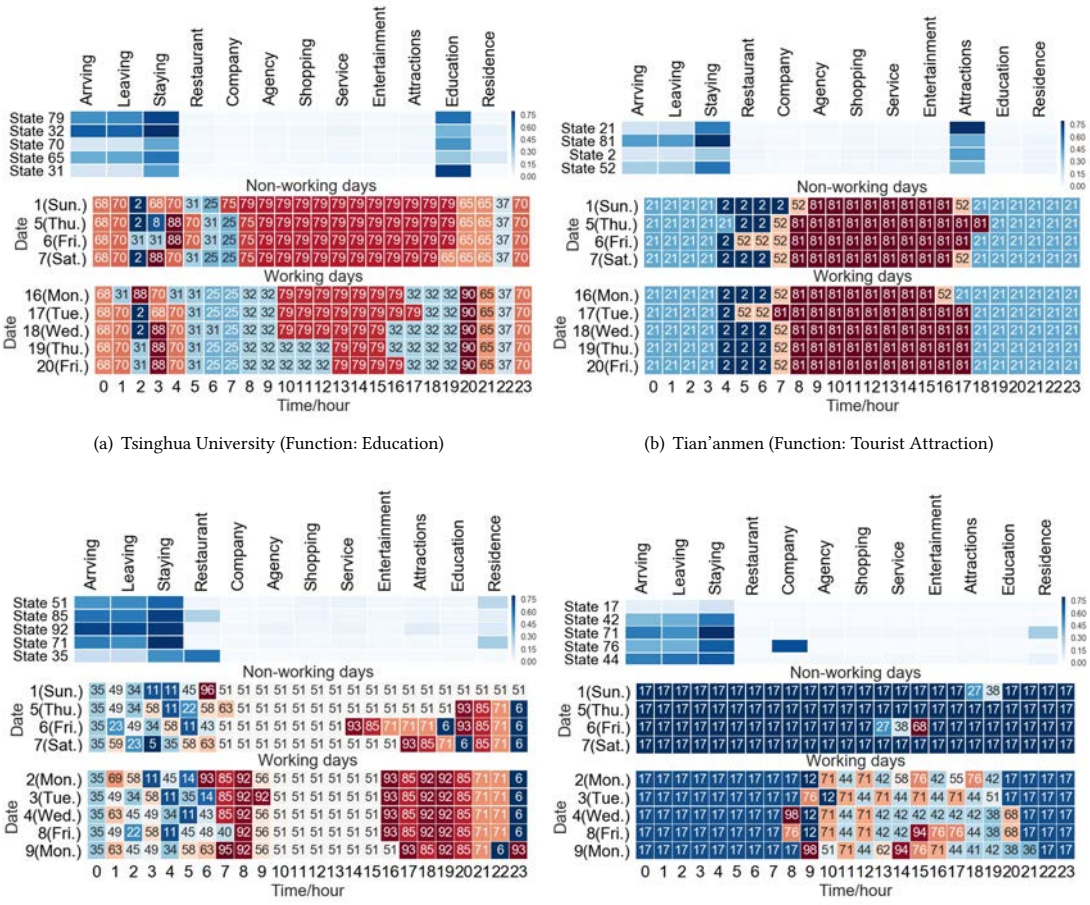
5.2.1 Data Usage. In the experiments, we divide the dataset into two parts. We utilize 21 days of data to generate the mobility behaviour observations to train our model and use the rest for prediction evaluation. We set the length of time slot to 1 hour. The observation in each time slot is 12-dimensional, including 3-dimensional population flow volume and 9-dimensional check-in frequency.

5.2.2 Model evaluation. To evaluate the effectiveness of our model and the inferred parameters, we utilize the obtained states to recover the observations by concatenating the mean value of corresponding state in the hidden state sequence in chronological order. We adopt $\epsilon = \sqrt{\frac{\sum_{r=1}^{R}\sum_{n=1}^{N}\sum_{l=1}^{L}(\mu_{n,l} - s_{r,n,l})^2}{R \times N \times L}}$, the error between all of the observations and the activities recovered by the mean value of the corresponding hidden state as an evaluation metric.

To determine the number of hidden states and the length of time slots for training, we searched different values. From fig. 3(a), we can observe that when K increases,ϵ decreases while the training time increases rapidly. Thus, we set $K = 100$ as a trade-off between model complexity and accuracy. As Figure 3(b) show, we find the length of time window has almost no impact on the training time and ϵ. We set is as 24, i.e., train the model day by day.

5.2.3 Prediction Performance. In terms of prediction, we use the rest 9 days' data for evaluation. We compare the mean value of the predicted next state with the ground-truth observations, and the baseline is general HMM. For mobility prediction, we adopt *RMSE*, the root-mean-square error between the normalized the population flow of predicted and the ground truth [25] for measurement. For check-ins, we adopt *TopM-accuracy* to reflect the accuracy on topN frequently-visited PoI prediction of all regions [21]. A lower *RMSE* or a higher *TopM-accuracy* indicates better prediction performance.

[2] Tencent Incorporation. https://lbs.qq.com/index.html



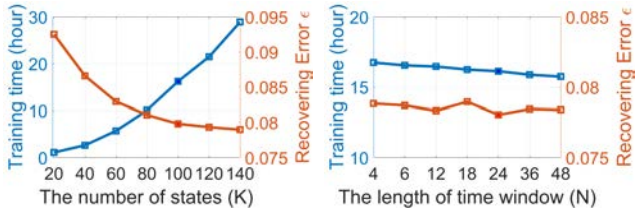
(a) Tsinghua University (Function: Education)

(b) Tian'anmen (Function: Tourist Attraction)

(c) Wangjing (Function: Residence)

(d) Zhongguancun Software Park (Function: Company)

Figure 2: Visualization for representative states and state sequences for different regions, where each row in the state sequences heatmap exhibits the state transition process of one day with the number indicating the corresponding state.



(a) Effects of K

(b) Effects of N

Figure 3: Effects of parameters.

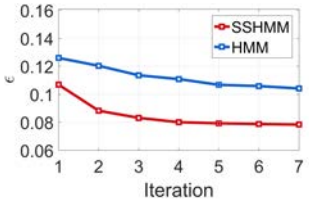


Figure 4: Model Effectiveness comparison.

5.3 Results

5.3.1 Model Effectiveness. Figure 4 shows the relationship between recovering error ϵ and the number of iterations. We compare it with the result of training independent HMM for each region with the same number of states. From Figure 4, we can observe that as the number of iterations increases, the recovering error ϵ decreases first, then tends to remain unchanged. Furthermore, our representative ability for human activities is better than HMM: When the model converges, our ϵ is 0.0793, which outperforms HMM by 54.2%.

5.3.2 Visualization of the Hidden States and Urban Dynamics. In order to demonstrate the ability of our model in discovering hidden states and revealing urban dynamics, we give a series of special examples and detailed explanations. Figure 2 visualizes the results for 6 regions. We first plot the mean value of the states with frequent occurrence, then show the state transition process in working day

and non-working day (Apr. 5th (Thur.), 6th (Fri.) and 7th (Sat.) are the Qingming Festival), respectively.

States: Each state represents two aspects: (1) The density of the population flow. For example, **state 32** presents a large volume of flows and high density of populations, **state 21** presents a small volume of flows while a high density of populations. (2) The visit frequency of different PoIs. For example, **state 31** indicates the most frequently visited PoI is *Education*, while **state 21** indicates the most frequently visited PoI is *Attractions*. As shown in Figure 2(a), *Tsinghua University* have the **state 79** during the day, and *Tian'anmen* has **state 81**. By combining these two semantics, we can infer the activity level and lifestyle of the region. For example, **state 79** shows active school status while **state 70** shows quiet school status.

Dynamics: Dynamics are represented by the state transition processes. Take the dynamics of *Tsinghua University* as an example. As shown in Figure 2(a), during the night, there are fewer people than the day as **state 70**. Besides, in working days, there is a sudden increase in crowd flow as **state 32** appear in 8:00-9:00 and 17:00-19:00. The transition from **state 70** to **32**, from **32** to **79** in working days reveals the dynamics that only students live in the region at night and many teachers go to school in the morning, causing the population denser than night. Compared with 2(a), 2(b) shows that both in working day and non-working day, the density of population is consistently high and the PoI visited most frequently is *Attractions*, as *Tian'anmen* is one of the most famous tourist spots in China. Another interesting finding is that for residential areas as Figure 2(c), non-working days are more prosperous and lively than working days, but working areas are quiet and peaceful as Figure 2(d) shown. To conclude, Figure 2 gives the insights as follows:
(1) From the state sequence heatmap, we notice that the state sequences are aligned by day, which indicates that the dynamics in the city have a period of one day, as the states in the same time slot but with different dates are usually the same.
(2) The dynamic patterns within working days or no-working days are very similar, while the patterns between working and non-working days are determined by the function of the region.

In summary, SSHMM not only detects the urban states with different activity patterns but also reveal the rhythm of daily life.

5.3.3 Visualization of Urban Functions. To further evaluate our model to infer the functions, we cluster the state sequences by the method introduced in Section 4.4 and found that 8 types is most suitable as the minimum DBI is achieved. We compare the result with the state-of-the-art functional zones discovering method, i.e., an LDA model by utilizing the static PoIs and mobility together [22]. The geographical distribution of the regions with their function types is shown in Figure 5, where different colors present different functions. After clustering, we label their functions by the revealed semantic dynamics and manually check some regions in Figure 2 to verify their functions, which shows most of the regions with the same functions are divided into one cluster. Besides, clusters obtained by our model and LDA model have with the Normalized Mutual Information (NMI) of 0.25, which measures the similarity of two divisions with the range from -0.5 to 1 [10]. Therefore, the functions we infer from dynamic patterns are validated by the labeled regions and the state-of-the-art method.

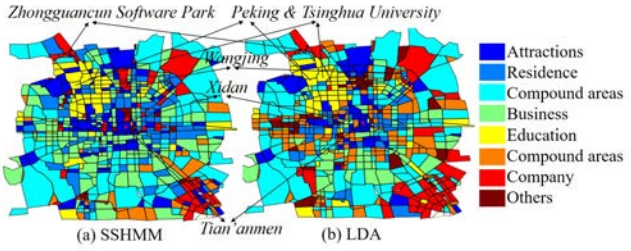
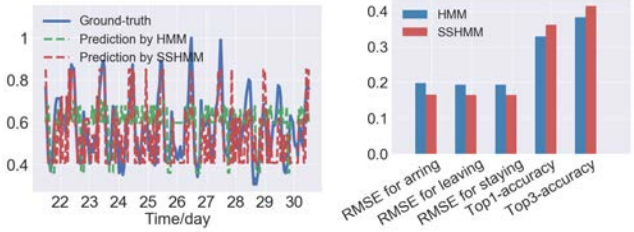


Figure 5: Visualization of the distribution of the regions with similar dynamics and functions.



(a) Prediction for the number of staying of *Tsinghua University*

(b) Evaluation metrics

Figure 6: Prediction performance compared with HMM.

5.3.4 Performance of Activity Prediction. In order to evaluate the prediction accuracy of our model, we first show the prediction of the number of staying for 9 days in *Tsinghua University* in Figure 6(a) as an example. Compared with the generate HMM, the recovered observations of our model are closer to the ground-truth in different time slots. The results for all metrics are shown in Figure 6(b), where the average *RMSE* of population flow prediction is 0.195 and the *Top3-accuracy* for PoI popularity prediction is 41.4%, outperforming the HMM by 16% and 8% respectively. These results demonstrate our SSHMM outperforms HMM in urban dynamics prediction problem.

6 CONCLUSION

In this paper, we study the problem of understanding urban dynamics. We propose a State-sharing Hidden Markov Model (SSHMM), where all the regions share a common state set, but each region has it own transition regularity. To make it practical, we not only derive the inference of the parameters but also give an efficient algorithm to update them. We evaluate our method via a real-life dataset in Beijing, which demonstrates that SSHMM learns semantics-rich urban dynamics model, recovers different activity regularities by a limited number of states and incurs low training cost. Our work also opens a new angle for dynamic urban region representation by learning the vector for both urban states (i.e, the mean of different feature dimension) and urban dynamics (i.e, the state sequence).

ACKNOWLEDGMENT

This work was supported in part by the National Key Research and Development Program of China under grant 2017YFE0112300, the National Nature Science Foundation of China under 61861136003,

61621091 and 61673237, Beijing National Research Center for Information Science and Technology under 20031887521.

REFERENCES

[1] Sofiane Abbar, Tahar Zanouda, Noora Al-Emadi, and Rachida Zegour. 2018. City of the People, for the People: Sensing Urban Dynamics via Social Media Interactions. In *Social Informatics*. 3–14.

[2] J. R. Bellegarda and D. Nahamoo. 1990. Tied mixture continuous parameter modeling for speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 38, 12 (Dec 1990), 2033–2045.

[3] D. L. Davies and D. W. Bouldin. 1979. A Cluster Separation Measure. *T-PAMI* (April 1979), 224–227.

[4] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)* (1977), 1–38.

[5] Manuel Garcia Docampo. 2014. Theories of Urban Dynamics. In *International Journal of Population Research*, Vol. 2014.

[6] Xuedong Huang, Fileno Alleva, Hsiao-Wuen Hon, Mei-Yuh Hwang, Kai-Fu Lee, and Ronald Rosenfeld. 1993. The SPHINX-II speech recognition system: an overview. *Computer Speech & Language* 7, 2 (1993), 137 – 148. http://www.sciencedirect.com/science/article/pii/S0885230883710077

[7] X. D. Huang. 1992. Phoneme classification using semicontinuous hidden Markov models. *IEEE Transactions on Signal Processing* 40, 5 (May 1992), 1062–1067.

[8] Mei-Yuh Hwang and Xuedong Huang. 1993. Shared-Distribution Hidden Markov Models for Speech Recognition. 1 (11 1993), 414 – 420.

[9] Xin Jin and Jiawei Han. 2011. K-medoids clustering. In *Encyclopedia of Machine Learning*. Springer, 564–565.

[10] Zeger F Knops, JB Antoine Maintz, Max A Viergever, and Josien PW Pluim. 2006. Normalized mutual information based registration using k-means clustering and shading correction. *Medical image analysis* 10, 3 (2006), 432–439.

[11] Kenneth Rose Liang Gu, Jayanth Nayak. 2000. Discriminative training of tied-mixture HMM by deterministic annealing. *ICSLP-2000* 4 (2000).

[12] Ziqian Lin, Jie Feng, Ziyang Lu, Yong Li, and Depeng Jin. 2019. DeepSTN+: Context-aware Spatial-Temporal Neural Network for Crowd Flow Prediction in Metropolis. In *AAAI*.

[13] Wesley Mathew, Ruben Raposo, and Bruno Martins. 2012. Predicting future locations with hidden Markov models. In *UbiComp*. ACM, 911–918.

[14] F. Miranda, H. Doraiswamy, M. Lage, K. Zhao, B. GonÃğalves, L. Wilson, M. Hsieh, and C. T. Silva. 2017. Urban Pulse: Capturing the Rhythm of Cities. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 791–800.

[15] Jiaul H. Paik. 2013. A Novel TF-IDF Weighting Scheme for Effective Ranking. In *SIGIR*. ACM, 343–352.

[16] L. R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77, 2 (Feb 1989), 257–286.

[17] A. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory* 13, 2 (April 1967), 260–269. https://doi.org/10.1109/TIT.1967.1054010

[18] Lloyd R Welch. 2003. Hidden Markov models and the Baum-Welch algorithm. *IEEE Information Theory Society Newsletter* 53, 4 (2003), 10–13.

[19] Fengli Xu, Tong Xia, Hancheng Cao, Yong Li, Funing Sun, and Fanchao Meng. 2018. Detecting Popular Temporal Modes in Population-scale Unlabelled Trajectory Data. *IMWUT* 2, 1 (March 2018), 46:1–46:25.

[20] Zijun Yao, Yanjie Fu, Bin Liu, Wangsu Hu, and Hui Xiong. 2018. Representing Urban Functions through Zone Embedding with Human Mobility Patterns.. In *IJCAI*. 3919–3925.

[21] Donghan Yu, Yong Li, Fengli Xu, Pengyu Zhang, and Vassilis Kostakos. 2018. Smartphone App Usage Prediction Using Points of Interest. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4, Article 174 (Jan. 2018), 21 pages.

[22] Nicholas Jing Yuan, Yu Zheng, Xing Xie, Yingzi Wang, Kai Zheng, and Hui Xiong. 2015. Discovering urban functional zones using latent activity trajectories. *IEEE Transactions on Knowledge and Data Engineering* 27, 3 (2015), 712–725.

[23] Chao Zhang, Keyang Zhang, Quan Yuan, Haoruo Peng, Yu Zheng, Tim Hanratty, Shaowen Wang, and Jiawei Han. 2017. Regions, Periods, Activities: Uncovering Urban Dynamics via Cross-Modal Representation Learning. In *WWW*. 361–370.

[24] Chao Zhang, Keyang Zhang, Quan Yuan, Luming Zhang, Tim Hanratty, and Jiawei Han. 2016. GMove: Group-Level Mobility Modeling Using Geo-Tagged Social Media. In *KDD*. ACM, 1305–1314.

[25] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction.. In *AAAI*. 1655–1661.

[26] Wanzheng Zhu, Chao Zhang, Shuochao Yao, Xiaobin Gao, and Jiawei Han. 2018. A Spherical Hidden Markov Model for Semantics-Rich Human Mobility Modeling.. In *AAAI*.

[27] Zefang Zong, Jie Feng, Kechun Liu, Hongzhi Shi, and Yong Li. 2019. DeepDPM: Dynamic Population Mapping via Deep Neural Network. In *AAAI*.

Efficient Path Prediction for Semi-Supervised and Weakly Supervised Hierarchical Text Classification

Huiru Xiao
Hong Kong University of Science and Technology
hxiaoaf@cse.ust.hk

Xin Liu
Hong Kong University of Science and Technology
xliucr@cse.ust.hk

Yangqiu Song
Hong Kong University of Science and Technology
yqsong@cse.ust.hk

ABSTRACT

Hierarchical text classification has many real-world applications. However, labeling a large number of documents is costly. In practice, we can use semi-supervised learning or weakly supervised learning (e.g., dataless classification) to reduce the labeling cost. In this paper, we propose a path cost-sensitive learning algorithm to utilize the structural information and further make use of unlabeled and weakly-labeled data. We use a generative model to leverage the large amount of unlabeled data and introduce path constraints into the learning algorithm to incorporate the structural information of the class hierarchy. The posterior probabilities of both unlabeled and weakly labeled data can be incorporated with path-dependent scores. Since we put a structure-sensitive cost to the learning algorithm to constrain the classification consistent with the class hierarchy and do not need to reconstruct the feature vectors for different structures, we can significantly reduce the computational cost compared to structural output learning. Experimental results on two hierarchical text classification benchmarks show that our approach is not only effective but also efficient to handle the semi-supervised and weakly supervised hierarchical text classification.

CCS CONCEPTS

• **Computing methodologies** → *Semi-supervised learning settings*; Classification and regression trees.

KEYWORDS

Hierarchical Text Classification, Semi-Supervised Learning, Weakly Supervised Learning, Structured Prediction

ACM Reference Format:
Huiru Xiao, Xin Liu, and Yangqiu Song. 2019. Efficient Path Prediction for Semi-Supervised and Weakly Supervised Hierarchical Text Classification. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313658

1 INTRODUCTION

Text classification has always been an important task, particularly with the vast growth of text data on the Web needed to be classified. The applications include news classification [8], product review classification [31], spam detection [25] and so on. Hierarchical classification (HC) and structured prediction are involved since the classes are usually organized as a hierarchy. In recent decades, many approaches have been proposed for HC. For example, top-down classification [34] classifies documents at the top layer and then propagates the results to next layer until the leaves. This greedy strategy propagates the classification error along the hierarchy. Contrarily, bottom-up classification [1] backpropagates the labels from the leaves to the top layer, making the leaves with less training data but sharing some similarities with their parents and siblings may not get well considered and trained. Moreover, structural output learning, such as structural perceptron [6] and structural SVM [37], can leverage the structural information in the class hierarchy well, but they need to do Kesler construction [11, 30] where for each sub-structure, the new features are constructed based on the existing features and the class dependencies. That is why structural output learning usually takes more time to train than top-down and bottom-up approaches. All the above approaches are supervised methods. When there are more unlabeled data, it is more challenging if we consider both class dependencies and efficiency of the practical use of hierarchical text classification.

There exist several ways to use the large amount of unlabeled data, among which semi-supervised learning (SSL) [4] and weakly supervised learning such as dataless classification [3, 33] are two representative ways. An example of SSL is [29]. It uses a mixture multinomial model to estimate the posterior probabilities of unlabeled data, which share the same parameters with the naive Bayes model for the labeled data. More parameters can be introduced to model the hierarchical structure, causing the model redundant and meanwhile not accurate enough. As for weakly supervised learning, dataless classification [33] uses the semantic similarities between label descriptions and document contents to provide weak labels for documents. When applying the weak labels, current approaches simply treat each label similarity independently and do not consider the path constraints in the label hierarchy.

To tackle the above problems for semi-supervised and weakly supervised hierarchical text classification, we propose a path cost-sensitive learning algorithm based on a generative model for text. When estimating the path posterior distribution, the path-dependent scores are incorporated to make the posteriori path-sensitive. The path-dependent score evaluates how accurate the current model is in terms of classifying a document among the paths in the class hierarchy. Then during inference, classification is constrained to keep the consistency of the hierarchy. By this mechanism, we develop a simple model with fewer parameters compared with existing approaches while maintaining the consistency property for the class dependencies in the hierarchy.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313658

The contributions of our paper are as follows:

(1) We propose **a new approach for hierarchical text classification** based on a probabilistic framework. We highlight its meanings in cost-sensitive learning and constraint learning.
(2) We show significant improvements on two widely used hierarchical text classification benchmarks and demonstrate our algorithm's **effectiveness** in **semi-supervised** and **weakly-supervised learning** settings.
(3) Our approach **reduces the complexity** of traditional methods. We achieve tens of times speedup while outperforming the state-of-the-art discriminative baseline.

The code and data used in the paper are available at https://github.com/HKUST-KnowComp/PathPredictionForTextClassification.

2 RELATED WORK

There are only a few studies on semi-supervised hierarchical (text) classification [9, 29], partially because of the difficulty to evaluate the class dependencies for unlabeled data and the time cost of using more complicated algorithms such as structural output learning [23]. Most semi-supervised hierarchical text classification works were based on EM algorithm introduced by [29]. Some are related with ours (see in Section 2.1), while others are not, e.g., [9] used EM algorithm to deal with incomplete hierarchy problem, which was not the same setting as ours. In this section, we simply start with the review of general hierarchical text classification and then explain the uniqueness and significance of our work.

Hierarchical text classification has been studied for several decades. Flat multi-label classification methods [36] ignore the hierarchy, thus poor for HC. Early works [12, 17, 22] often used "pachinko-machine models" which assigned a local classifier at each node and classified documents recursively. Top-down and bottom-up approaches utilize the local classifier ideas, but top-down is a greedy strategy so it may not find optimal solutions, while bottom-up approach does not well consider and train the classes with less training data.

To better exploit the class hierarchy, algorithms particularly designed for trees can assist. In practice, both generative and discriminative models are used. In the following, we will review the related work of these two categories.

2.1 Generative Models

[29] summarized the text generative model and provided the naive Bayes classifier and Expectation-Maximization (EM) algorithm for flat classification. As for HC, it introduced more parameters to account for the class dependencies. [24] remodeled the framework in another way. They applied shrinkage to smooth parameter estimates using the class hierarchy. [7] also used the same generative framework but proposed a clustering-based partitioning technique. These generative hierarchical methods can bring some structural information to the model, but they do not make full use of the hierarchy and have difficulties scaling to large hierarchies.

2.2 Discriminative Models

Discriminative methods are also popular for HC. Orthogonal Transfer [39] borrowed the idea of top-down classification where each node had a regularized classifier and each node's normal vector classifying hyperplane was encouraged to be orthogonal to its ancestors'. Hierarchical Bayesian Logistic Regression [16] leveraged the hierarchical dependencies by giving the children nodes a prior centered on the parameters of its parents. The idea was further developed in Hierarchically Regularized SVM and Logistic Regression [15], where the hierarchical dependencies were incorporated into the parameter regularization structure. More recently, the idea of hierarchical regularization has been applied to deep models and also showed some improvements [32]. [5] simplified the construction of classifier by building a binary classifier on each tree node and providing the cost-sensitive learning (HierCost). All the above approaches are still based on top-down or greedy classification which can result in non-optimal solutions. Another similar work with ours is [38]'s hierarchical loss for classification, which defined the hierarchical loss or win as the weighted sum of the probabilities of the nodes along the path. In contrast to their work, we use the sum of the (weakly) labeled instances along a path as score to perform path cost-sensitive learning.

To find more theoretically guaranteed solutions, some algorithms were developed based on structural output learning [6, 18, 35, 37], which can be proved to be global optimal for HC. Hierarchical SVM (HSVM) [2], one example of structural SVM, generalized SVM to structured data with a path-dependent discriminant function. In general, when performing structural output learning, Kesler construction is used to construct the feature vectors for comparing different structures [11, 30], which adds much more computation than top-down or bottom-up classification approaches.

In summary, generative and discriminative models can both be adapted to HC problems. Discriminative models achieve better performance with adequate labeled data [28], especially if a better representation for text can be found, e.g., using deep learning [32]. Whereas generative models have their advantage for handling more uncertainties [28] for limited labeled data and under noisy supervision. Our work is based on a generative model yet has the same parameter size as the flat classification. We find that it significantly boosts the performance of semi-supervised learning and weakly supervised learning as well as reduces the computational cost.

3 PATH PREDICTION FOR HIERARCHICAL CLASSIFICATION

In HC, the classes constitute a hierarchy, denoted as $\mathcal{T}$. $\mathcal{T}$ is a tree whose depth is d, with the root node in depth 0. Then the classes are distributed from depth 1 to d. We suppose that all leaf nodes are in depth d. This can always be satisfied by expanding the shallower leaf node (i.e. giving it a child) until it reaches depth d. When evaluating models, these dummy nodes from $\mathcal{T}$ can be easily removed to avoid affecting the performance measure.

Let $C_1, \ldots, C_d$ be the class sets in depth $1, \ldots,$ depth d accordingly, with sizes $M_1, \ldots, M_d$. To classify a document, we assign labels in each depth, i.e., the document gets d labels $\{c^k_{j_k} : k = 1, \ldots, d, j_k = 1, \ldots, M_k\}$. These $\{c^k_{j_k}\}$ form a path in $\mathcal{T}$ if the classification results in each depth are consistent with other depths. We want to maintain the consistency of the hierarchy, therefore we classify the documents by paths instead of by multi-label classes. After assigned a path, the document's classes are the nodes lying in the path. It is similar with structured prediction since a path can be

regarded as a structured object, which contains more information than a set of multi-label classes without path constraints.

To sum up, path prediction aims at making use of the structural information in the class hierarchy to train the classifier. Note that the classifier is for paths in the hierarchy instead of classes. The details of path prediction algorithm are given in the next section.

4 PATH COST-SENSITIVE LEARNING

In this section, we introduce our method which utilizes the structural information to learn the classifier, revealing its meanings in cost-sensitive learning and constraint learning.

4.1 Path-Generated Probabilistic Framework

We base our work on a widely-used probabilistic framework, which constructs a generative model for text. In the framework, text data are assumed to be generated from a mixture of multinomial distributions over words. Previous works [24, 29] assumed that the mixture components in the generative model have a one-to-one correspondence with the classes. However, in order to perform path prediction, we presume that the mixture components have a one-to-one correspondence with the paths.

Define $\mathcal{P}$ to be the set of all paths which start from the root node and end in some leaf node in the class hierarchy $\mathcal{T}$, so the size of $\mathcal{P}$ equals to that of the leaf nodes M_d. Let $\mathcal{V}$ be the vocabulary. Denote θ as the parameters for the mixture multinomial model. For a document x_i with length $|x_i|$, suppose x_{it} is the word frequency of word w_t in x_i, which is the document feature represented by vector space model [21]. Then the generative process runs as following.

First, select a mixture component, or equivalently a path $p_j \in \mathcal{P}$, from $P(p_j|\theta)$ (prior of p_j). Next, generate the document x_i by selecting the length $|x_i|$ and picking up $|x_i|$ words from $P(x_i|p_j;\theta)$. According to the law of total probability and the naive Bayes assumption that given the labels, the occurrence times of each word in a document are conditionally independent with its position as well as other words, the probability of generating x_i is

$$P(x_i|\theta) \propto P(|x_i|)\sum_{j=1}^{M_d} P(p_j|\theta)\prod_{w_t\in\mathcal{V}} P(w_t|p_j;\theta)^{x_{it}}. \quad (1)$$

In general, document lengths are assumed to be independent with classes, thus independent with paths. So model parameters θ include the path prior $\theta_{p_j} \equiv P(p_j|\theta)$ and the multinomial distribution over words for each path $\theta_{w_t|p_j} \equiv P(w_t|p_j;\theta)$.

4.2 Path-Dependent Scores

Given a data set $\mathcal{D} = \{(x_i, y_i) : x_i \in \mathcal{X}_l\} \cup \{x_i : x_i \in \mathcal{X}_u\}$, consisting of the labeled documents $\mathcal{X}_l$ and the unlabeled documents $\mathcal{X}_u$. We now derive the parameter estimation in a supervised manner. With only labeled data considered, we maximize $P(\theta|\mathcal{X}_l)$, which can be done by counting the corresponding occurrences of events. The event counts are usually the hard counts for flat classification. Here we use a path-dependent score to substitute it.

First we define the score of a node in $\mathcal{T}$ for a document. Suppose $x_i \in X_l$, $c^k_{j_k}$ is the k^{th} node in path p_j. The node score of $c^k_{j_k}$ for x_i, denoted as $S_i(c^k_{j_k})$, indicates the label of x_i. When x_i is labeled with the ground truth labels, $S_i(c^k_{j_k}) = 1$ if and only if $c^k_{j_k}$ is one of

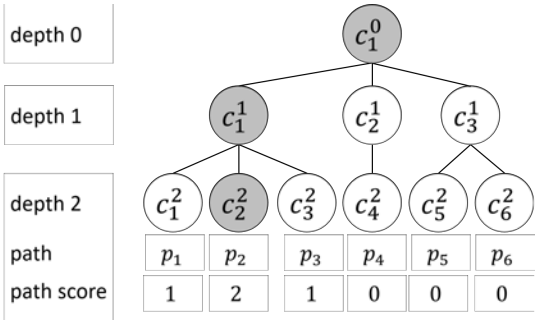


Figure 1: An example of path scores. Suppose x_i is labeled as $\{c^1_1, c^2_2\}$. S_{ij} for $j = 1, \ldots, 6$ is shown in the figure.

x_i's labels. We also consider the weakly supervised case. In [33]'s dataless text classification, for $x_i \in \mathcal{X}_l$, it is weakly labeled by the semantical similarities with classes. We assign value 1 to $S_i(c^k_{j_k})$ if x_i has the largest similarity with $c^k_{j_k}$ among all classes in depth k and 0 otherwise.

Next we introduce the path score. For $x_i \in X_l$, the score of path p_j, denoted as S_{ij}, is the sum of the nodes' scores $S_i(c^k_{j_k})$ lying in p_j except the root node since it makes no sense for classification.

$$S_{ij} \equiv S_i(p_j) = \sum_{k=1}^{d} S_i(c^k_{j_k}). \quad (2)$$

Take the hierarchy in Figure 1 as an example. x_i is labeled as $\{c^1_1, c^2_2\}$, then $S_{i2} = S_i(c^1_1) + S_i(c^2_2) = 2$, $S_{i1} = S_{i3} = 1$, while other paths score 0. If x_i is weakly labeled by the similarities, then we label it with the classes having the maximum similarity in each depth and obtain the path scores in the same way.

4.3 Path Cost-Sensitive Naive Bayes Classifier

While doing the empirical counts, the Laplace smoothing is often applied by adding one count to each event to avoid zero probability and shrink the estimator. Combining the event counts (i.e. the path scores) and the smoothing term, the parameter estimates are:

$$\hat{\theta}_{p_j} \equiv P(p_j|\hat{\theta}) = \frac{1 + \sum_{x_i\in\mathcal{X}_l} S_{ij}}{M_d + \sum_{k=1}^{M_d}\sum_{x_i\in\mathcal{X}_l} S_{ik}}, \quad (3)$$

$$\hat{\theta}_{w_t|p_j} \equiv P(w_t|p_j;\hat{\theta}) = \frac{1 + \sum_{x_i\in\mathcal{X}_l} S_{ij}x_{it}}{|\mathcal{V}| + \sum_{s=1}^{|\mathcal{V}|}\sum_{x_i\in\mathcal{X}_l} S_{ij}x_{is}}. \quad (4)$$

There are two aspects of using the path scores as event counts:

(1) Cost-sensitive performance measures are considered since different data samples are given different weights. In Figure 1, x_i is counted twice for p_2, once in c^1_1 and once in c^2_2, thus obtaining more weights. p_1 and p_3 are not right paths for x_i, but they still classify correctly in depth 1, thus get one count, less than p_2 but larger than other paths who have no correct labels at all. This path **cost-sensitive learning** behavior helps the model to maintain structural information.

(2) The path scores function as the measuring indicators of paths, capacitating the model to classify the documents by paths. The path prediction actually puts constraints on the classifier, where the prediction results must be consistent with the class

hierarchy. Furthermore, the **constraint learning** reduces the search space and improves efficiency.

After estimating θ from $\mathcal{X}_l$, for any test document x_i, the posterior probability distribution can be obtained by Bayes' rule:

$$P(y_i = p_j|x_i;\hat{\theta}) = \frac{P(p_j|\hat{\theta})P(x_i|p_j;\hat{\theta})}{P(x_i|\hat{\theta})} = \frac{\hat{\theta}_{p_j}\prod_{w_t\in\mathcal{V}}(\hat{\theta}_{w_t|p_j})^{x_{it}}}{\sum_{k=1}^{M_d}\hat{\theta}_{p_k}\prod_{w_t\in\mathcal{V}}(\hat{\theta}_{w_t|p_k})^{x_{it}}}. \quad (5)$$

Then x_i will be classified into $\arg\max_j P(y_i = p_j|x_i;\hat{\theta})$.

The path cost-sensitive naive Bayes classifier (PCNB) for the generative model are introduced above. Next we will present the semi-supervised path cost-sensitive learning algorithm.

4.4 Semi-Supervised Path Cost-Sensitive Learning

Until now, only the labeled data are used during training, but we want to make use of the unlabeled data to ameliorate the classifier. We follow [29] to apply EM technique for SSL.

When the initial parameters are given, the posterior probabilities of $\mathcal{X}_u$, computed through Eq. (5), can act as the path score S_{ij} for $x_i \in X_u$. Combining the labeled and unlabeled data together, the parameter estimates are changed into

$$\hat{\theta}_{p_j} = \frac{1+\sum_{x_i\in\mathcal{X}_l\cup\mathcal{X}_u} S_{ij}}{M_d + \sum_{k=1}^{M_d}\sum_{x_i\in\mathcal{X}_l\cup\mathcal{X}_u} S_{ik}}, \quad (6)$$

$$\hat{\theta}_{w_t|p_j} = \frac{1+\sum_{x_i\in\mathcal{X}_l\cup\mathcal{X}_u} S_{ij}x_{it}}{|\mathcal{V}| + \sum_{s=1}^{|\mathcal{V}|}\sum_{x_i\in\mathcal{X}_l\cup\mathcal{X}_u} S_{ij}x_{is}}. \quad (7)$$

Note that the numerical value of S_{ij} for $x_i \in X_u$ ranges in $[0, 1]$ since it is the posterior probability. Therefore, the unlabeled data weight less than the labeled data while estimating the parameters. It is reasonable because the labeled data are more authentic than the inference results of unlabeled data, especially in the early iterations where the model does not reach convergence.

The new θ obtained via Eqs. (6) and (7) are then used to compute the posterior probabilities of $\mathcal{X}_u$ again, which in turn update θ. The iterative process keeps maximizing the likelihood of the dataset $P(\theta|\mathcal{D})$, equivalent to maximizing the log likelihood:

$$l(\theta|X,Y) \propto log(P(\theta)P(\mathcal{D}|\theta)) = log(P(\theta)) + \sum_{x_i\in\mathcal{X}_l} log(P(y_i = p_j|\theta)P(x_i|y_i = p_j;\theta)) + \sum_{x_i\in\mathcal{X}_u} log\sum_{p_j\in\mathcal{P}} P(p_j|\theta)P(x_i|p_j;\theta). \quad (8)$$

Refer to [10], the convergence of EM can be guaranteed, but it reaches some local maxima. To enable the algorithm to find good local maxima, we initialize θ with those obtained through the naive Bayes classifier on $\mathcal{X}_l$. Algorithm 1 presents the EM algorithm for the path cost-sensitive classification (PCEM).

5 EXPERIMENTS

For empirical evaluation of effectiveness and efficiency of our approach, we design experiments on semi-supervised and weakly supervised hierarchical text classification tasks, compared to the representative and the state-of-the-art baselines.

Algorithm 1 Path Cost-Sensitive Algorithm with EM

Input: Training data $\mathcal{D} = \mathcal{X}_l \cup \mathcal{X}_u$, test data $\mathcal{D}_{test} = \mathcal{X}_{test}$, class hierarchy $\mathcal{T}$.

Training:

- Compute path scores for $\mathcal{X}_l$, using Eq. (2).
- Initialize the classifier $\hat{\theta}$ with the path-constrained naive Bayes classifier (Section 4.3).
- **while** $l(\hat{\theta})$ (Eq. (8)) not converged **do**
  E-step: use the current $\hat{\theta}$ to compute the path scores of $\mathcal{X}_u$ with Eq. (5).
  M-step: use the path scores of $\mathcal{X}_l$ and $\mathcal{X}_u$ to re-estimate $\hat{\theta}$, with Eqs. (6) and (7).
- **end while**

Testing:

- Compute the posterior probabilities for $\mathcal{X}_{test}$ with Eq. (5). Classify $\mathcal{X}_{test}$ by selecting $\hat{j} = \arg\max_j P(y_i = p_j|x_i;\hat{\theta})$ for $x_i \in \mathcal{X}_{test}, p_j \in \mathcal{P}$. The classes of x_i are nodes in $p_{\hat{j}}$.

Output: The classifier $\hat{\theta}$ and classification results for $\mathcal{D}_{test}$.

	#Training	#Test	#Features	#Leaves	#Nodes	Depth
20NG	15,077	3,769	103,363	20	27	2
RCV1	6,395	1,733	26,888	35	56	3

Table 1: Statistics of the datasets.

5.1 Experimental Design

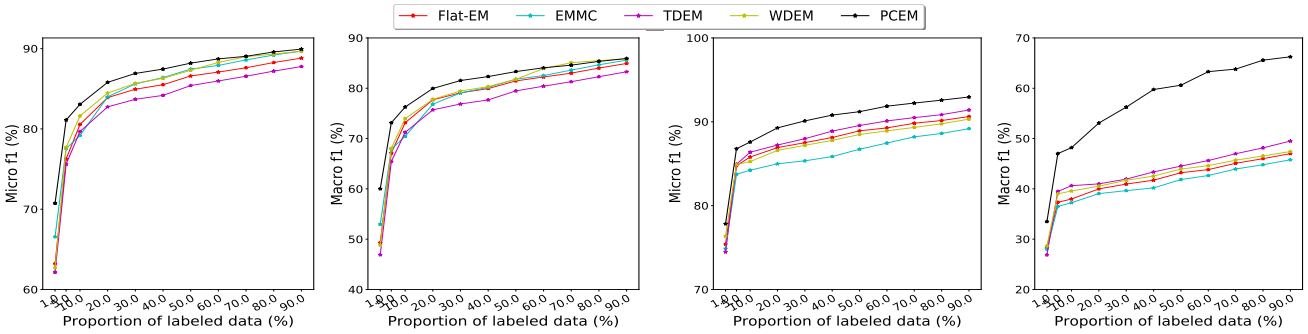
5.1.1 Datasets. We use two datasets, both of which have semi-supervised and weakly-supervised version. The statistics are listed in Table 1.

- **20NG**[1] [19] 20 Newsgroups is a widely-used text classification dataset. To experiment on weakly-supervised setting and compare with semi-supervised baselines, we use dataless 20NG provided in [33].
- **RCV1** [20] RCV1 is a collection of manually labeled Reuters News from 1996-1997. We also use the subset of data provided in [33], which consists of 8,668 single-labeled news documents on the dataless setting.

5.1.2 Baselines. We compare our path cost-sensitive algorithms (**PCNB** and **PCEM**) with the following baselines:

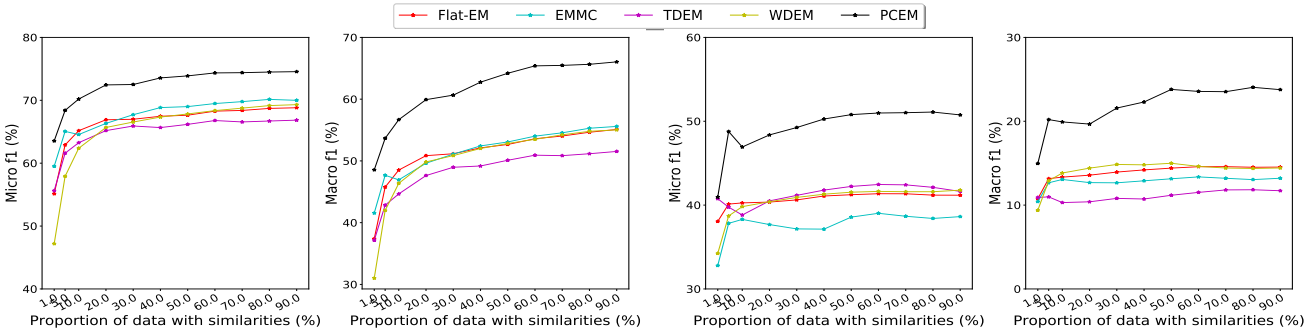
- **Generative baselines**
  (1) Flat naive Bayes classifier (**Flat-NB**) and **Flat-EM** algorithm: the flat classifiers introduced in [29].
  (2) Naive Bayes classifier with multiple components (**NBMC**) and **EMMC**: a more expressive model proposed by [29].
  (3) Top-down naive Bayes classifier (**TDNB**) and **TDEM**: the classifiers run in the top-down way.
  (4) Win-driven naive Bayes classifier (**WDNB**) and **WDEM**: the modified hierarchical loss for classification [38].
- **Discriminative baselines**

[1] http://qwone.com/~jason/20Newsgroups/



(a) Evaluation Results on 20NG (b) Evaluation Results on RCV1

Figure 2: Experimental results on semi-supervised classification. (a)-(d) shows the micro-F_1 score and macro-F_1 score on 20NG and RCV1. The horizontal axis is label rate (%). The vertical axis represents the F_1 scores (%).



(a) Evaluation Results on 20NG (b) Evaluation Results on RCV1

Figure 3: Experimental results on weakly-supervised classification. (a)-(d) shows the micro-F_1 scores and macro-F_1 scores on dataless 20NG and RCV1. The horizontal axis is the proportion of data with semantical similarities (%) in training set. The vertical axis represents the F_1 scores (%).

(1) Logistic regression (**LR**) and **SVM**: two classical discriminative methods. Our experiments use the LibLinear[2] [13] to train corresponding models and test. During the experiments, we found that dual solvers were much faster and even better in performance than primal solvers, so we chose dual solvers.
(2) **HierCost**[3] [5]: the state-of-the-art discriminative method for hierarchical text classification.

5.1.3 Evaluation Metrics. We use F_1 scores [40] to evaluate the performances of all methods. Denote TP_i, FP_i, FN_i as the instance numbers of true-positive, false-positive and false negative for class c_i. Let C be the set of all classes except the root node. Two conventional F_1 scores are defined as:

- **Micro**-$F_1 = \frac{2PR}{P+R}$, where $P = \frac{\sum_{c_i \in C} TP_i}{\sum_{c_i \in C}(TP_i+FP_i)}$ is the averaged precision and $R = \frac{\sum_{c_i \in C} TP_i}{\sum_{c_i \in C}(TP_i+FN_i)}$ is the averaged recall.
- **Macro**-$F_1 = \frac{1}{|C|}\sum_{c_i \in C} \frac{2P_iR_i}{P_i+R_i}$, where $P_i = \frac{TP_i}{TP_i+FP_i}$ and $R_i = \frac{TP_i}{TP_i+FN_i}$ are the precision and the recall for c_i.

For the two F_1 scores, we measure the overall performance of all classes in the hierarchy in our experiments.

5.2 Results

To evaluate our algorithms, we compare our algorithms with the baselines in semi-supervised and weakly supervised hierarchical text classification. Results on all datasets under 1% label rate are summarized in Table 2, where 1% label rate means there are 1% data in the training set are labeled or weakly-labeled, which is a common setting for semi-supervised text classification. To show that our approach (PCEM) indeed levareges unlabeled data and weakly labeled data well, we present the results under different label rates compared with other EM methods in Figure 2 and 3. For each experiment, we randomly split the training data into labeled and unlabeled according to the label rate, then run experiments using the splitted training data. The running is executed for 5 times and the mean F_1 scores are calculated. Next we will analyze the results. Time efficiency will also be discussed.

[2] https://www.csie.ntu.edu.tw/
[3] https://cs.gmu.edu/~mlbio/HierCost/

	20NG				RCV1			
	labeled		dataless		labeled		dataless	
LR	‡52.02	‡42.41	‡44.16	‡31.51	‡69.59	†24.43	†33.54	‡9.00
SVM	‡48.33	‡39.73	‡41.70	‡30.24	‡68.78	†23.97	†34.15	†9.72
HierCost	‡48.12	‡40.89	‡43.26	‡32.30	‡69.22	†24.98	†31.07	‡8.84
NB	‡53.39	‡39.94	‡47.29	‡30.67	‡70.68	†24.48	†33.29	‡8.39
NBMC	‡46.99	‡38.02	‡43.24	‡28.82	†69.84	†23.52	†28.91	‡6.91
TDNB	‡55.50	‡42.16	‡48.06	‡31.02	‡70.37	†24.65	33.67	†8.40
WDNB	‡53.66	‡41.53	‡47.19	‡31.02	‡70.89	†25.04	34.24	‡9.38
PCNB	‡58.33	‡48.04	‡52.14	†38.50	†73.63	29.95	37.06	12.47
EM	†63.21	†49.30	†55.13	†37.40	75.38	28.32	38.05	†10.76
EMMC	†66.56	†52.95	59.50	†41.56	74.86	28.04	32.79	†10.42
TDEM	†62.14	‡46.89	†55.62	†37.14	†74.48	†26.88	40.76	†10.91
WDEM	†62.71	‡48.85	‡47.19	‡31.02	76.35	28.66	34.24	‡9.38
PCEM	**70.73**	**60.02**	**63.54**	**48.56**	**77.83**	**33.49**	**40.96**	**14.96**

Table 2: F_1 scores (%) on 1% "labeled" data of each dataset. Under each dataset, the two columns are Micro-F_1 (left) and Macro-F_1 (right). The best results are shown in boldface. The statistical significance metrics are marked with either † if p-values < 0.05 or ‡ if p-values < 0.001.

5.2.1 Semi-Supervised Classification with True Labels. Table 2 shows that when the training data are partly labeled with the ground truth labels, PCEM has remarkable superiority over other methods all the time. The discriminative baselines do not have their advantages on the semi-supervised and weakly supervised settings. When compared with generative baselines, our approaches, either naive Bayes (PCNB) or semi-supervised (PCEM), are the best among the corresponding methods. It demonstrates that our algorithms makes good use of the structural information to improve the hierarchical classification.

As expected, EM approaches outperform the corresponding naive Bayes classifier under 1% label rate, which reveals the benefits from the unlabeled data. However, we also noticed that EM may be surpassed by NB when the label rate gets larger. That is related with whether the ratio between labeled and unlabeled data is suitable for SSL, as well as the bias of unlabeled data. This issue has been discussed in previous works [14].

To see the performance in SSL, we compare PCEM with other EM methods in Figure 2. The label rate ranges in $[1.0, 90.0](\%)$. We find that PCEM outperforms others steadily. Other hierarchical EM methods are close to Flat-EM, showing that they takes little advantage of the class hierarchy. The results reveal the effectiveness of PCEM under all label rates for semi-supervised classification.

5.2.2 Weakly-Supervised Classification on Dataless Setting. We also make a comparison with the baselines on dataless text classification. The experimental setting is the same as the semi-supervised classification, except that the training documents do not have labels. Instead, some of them are 'labeled' as classes with the maximal semantical similarities. We use the dataless 20NG and RCV1 datasets provided by [33]. Results are presented in Table 2 and Figure 3.

We find the consistent results with the semi-supervised setting. PCEM can always beat the baselines with significant improvements. PCNB is also better than other NB methods. It is worth noting that the gaps between our algorithms (PCNB and PCEM) and the baselines are bigger than those in the semi-supervised setting. We think the reason is that for this weakly-labeled dataset, the similarities can be seen as noisy labels for documents. In this noisy circumstance, our path cost-sensitive learning algorithm with the probabilistic framework is pretty good at making use of the structural information and features of unlabeled data to recover the true generative distribution.

5.2.3 Efficiency Comparison. Time complexity is also under consideration to evaluate our algorithms. PCNB is highly efficient, faster than all of the other methods except Flat-NB and even competitive with Flat-NB. PCEM is slightly slower than LR and SVM, but that is because EM methods leverage the unlabeled data, which cannot be used by discriminative methods. The trade-off is acceptable, especially considering the excellent performance of PCEM. Furthermore, PCEM also achieves tens of times speedup compared to HierCost.

6 CONCLUSIONS AND FUTURE WORK

We present an effective and efficient approach for hierarchical text classification. Our path cost-sensitive learning algorithm alters the traditional generative model of text with a path-generated model to constrain the classifier by the class hierarchy. We show that our algorithm outperforms other baselines on semi-supervised learning and weakly supervised learning. In addition, our model has the potential of extension to other models, not limited to the generative one, if the path-dependent scores are incorporated appropriately. For the possible future work, we will convert the current framework into a discriminative learning framework following [6] and apply deep neural models to learn a better representation for text [26, 27]. Discrimative framework will further improve the learning when there are more labeled data and deep neural models are more powerful to handle different kinds of weak supervision.

7 ACKNOWLEDGEMENT

This paper was supported by HKUST-WeChat WHAT Lab and the Early Career Scheme (ECS, No. 26206717) from Research Grants Council in Hong Kong.

REFERENCES

[1] Paul N. Bennett and Nam Nguyen. 2009. Refined experts: improving classification in large taxonomies. In *SIGIR*. ACM, 11–18.

[2] Lijuan Cai and Thomas Hofmann. 2004. Hierarchical document categorization with support vector machines. In *CIKM*. ACM, 78–87.

[3] Ming-Wei Chang, Lev-Arie Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of Semantic Representation: Dataless Classification. In *AAAI*. AAAI Press, 830–835.

[4] Olivier Chapelle, Bernhard Schlkopf, and Alexander Zien. 2010. *Semi-Supervised Learning* (1st ed.). The MIT Press.

[5] Anveshi Charuvaka and Huzefa Rangwala. 2015. HierCost: Improving Large Scale Hierarchical Classification with Cost Sensitive Learning. In *ECML/PKDD (1) (Lecture Notes in Computer Science)*, Vol. 9284. Springer, 675–690.

[6] Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *EMNLP*.

[7] Gao Cong, Wee Sun Lee, Haoran Wu, and Bing Liu. 2004. Semi-supervised Text Classification Using Partitioned EM. In *DASFAA (Lecture Notes in Computer Science)*, Vol. 2973. Springer, 482–493.

[8] Ido Dagan, Yael Karov, and Dan Roth. 1997. Mistake-Driven Learning in Text Categorization. In *EMNLP*. ACL.

[9] Bhavana Dalvi, Aditya Kumar Mishra, and William W. Cohen. 2016. Hierarchical Semi-supervised Classification with Incomplete Class Hierarchies. In *WSDM*. ACM, 193–202.

[10] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)* (1977), 1–38.

[11] Richard O Duda and Peter E Hart. 1973. Pattern classification and scene analysis. *A Wiley-Interscience Publication, New York: Wiley, 1973* (1973).

[12] Susan T. Dumais and Hao Chen. 2000. Hierarchical classification of Web content. In *SIGIR*. ACM, 256–263.

[13] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research* 9 (2008), 1871–1874.

[14] Patrick Fox-Roberts and Edward Rosten. 2014. Unbiased generative semi-supervised learning. *Journal of Machine Learning Research* 15, 1 (2014), 367–443.

[15] Siddharth Gopal and Yiming Yang. 2013. Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *KDD*. ACM, 257–265.

[16] Siddharth Gopal, Yiming Yang, Bing Bai, and Alexandru Niculescu-Mizil. 2012. Bayesian models for Large-scale Hierarchical Classification. In *NIPS*. 2420–2428.

[17] Daphne Koller and Mehran Sahami. 1997. Hierarchically Classifying Documents Using Very Few Words. In *ICML*. Morgan Kaufmann, 170–178.

[18] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*. 282–289.

[19] Ken Lang. 1995. NewsWeeder: Learning to Filter Netnews. In *ICML*. Morgan Kaufmann, 331–339.

[20] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research* 5 (2004), 361–397.

[21] Mingyoug Liu and Jiangang Yang. 2012. An improvement of TFIDF weighting in text categorization. *International proceedings of computer science and information technology* (2012), 44–47.

[22] Tie-Yan Liu, Yiming Yang, Hao Wan, Hua-Jun Zeng, Zheng Chen, and Wei-Ying Ma. 2005. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explorations* 7, 1 (2005), 36–43.

[23] Gideon S. Mann and Andrew McCallum. 2008. Generalized Expectation Criteria for Semi-Supervised Learning of Conditional Random Fields. In *ACL*. The Association for Computer Linguistics, 870–878.

[24] Andrew McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. 1998. Improving Text Classification by Shrinkage in a Hierarchy of Classes. In *ICML*. Morgan Kaufmann, 359–367.

[25] Michael Mccord and M Chuah. 2011. Spam detection on twitter using traditional classifiers. In *international conference on Autonomic and trusted computing*. Springer, 175–186.

[26] Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2018. Weakly-Supervised Neural Text Classification. In *CIKM*. ACM, 983–992.

[27] Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2019. Weakly-Supervised Hierarchical Text Classification. In *AAAI*. AAAI Press.

[28] Andrew Y. Ng and Michael I. Jordan. 2001. On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes. In *NIPS*. MIT Press, 841–848.

[29] Kamal Nigam, Andrew McCallum, and Tom Mitchell. 2006. Semi-supervised text classification using EM. *Semi-Supervised Learning* (2006), 33–56.

[30] N. J. Nilsson. 1965. *Learning machines: Foundations of Trainable Pattern-Classifying Systems* (1st ed.). McGraw-Hill.

[31] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. In *EMNLP*.

[32] Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-Scale Hierarchical Text Classification with Recursively Regularized Deep Graph-CNN. In *WWW*. ACM, 1063–1072.

[33] Yangqiu Song and Dan Roth. 2014. On Dataless Hierarchical Text Classification. In *AAAI*. AAAI Press, 1579–1585.

[34] Aixin Sun and Ee-Peng Lim. 2001. Hierarchical Text Classification and Evaluation. In *ICDM*. IEEE Computer Society, 521–528.

[35] Benjamin Taskar, Carlos Guestrin, and Daphne Koller. 2003. Max-Margin Markov Networks. In *NIPS*. 25–32.

[36] Domonkos Tikk and György Biró. 2004. A hierarchical test categorization approach and its application to FRT expansion. *Austr. J. Intelligent Information Processing Systems* 8, 3 (2004), 123–131.

[37] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large Margin Methods for Structured and Interdependent Output Variables. *Journal of Machine Learning Research* 6 (2005), 1453–1484.

[38] Cinna Wu, Mark Tygert, and Yann LeCun. 2017. Hierarchical loss for classification. *CoRR* abs/1709.01062 (2017).

[39] Lin Xiao, Dengyong Zhou, and Mingrui Wu. 2011. Hierarchical Classification via Orthogonal Transfer. In *ICML*. Omnipress, 801–808.

[40] Yiming Yang. 1999. An Evaluation of Statistical Approaches to Text Categorization. *Inf. Retr.* 1, 1-2 (1999), 69–90.

Hierarchical Neural Variational Model for Personalized Sequential Recommendation

Teng Xiao
School of Data and Computer Science, Sun Yat-sen University, China
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
cstengxiao@gmail.com

Shangsong Liang*
School of Data and Computer Science, Sun Yat-sen University, China
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
liangshangsong@gmail.com

Zaiqiao Meng
School of Data and Computer Science, Sun Yat-sen University, China
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China
zqmeng@aliyun.com

ABSTRACT

In this paper, we study the problem of recommending personalized items to users given their sequential behaviors. Most sequential recommendation models only capture a user's short-term preference in a short session, and neglect his general (unchanged over time) and long-term preferences. Besides, they are all based on deterministic neural networks, and consider users' latent preferences as point vectors in a low-dimensional continuous space. However, in real world, the evolutions of users' preferences are full of uncertainties. We address this problem by proposing a hierarchical neural variational model (HNVM). HNVM models users' three preferences: general, long-term and short-term preferences through an unified hierarchical deep generative process. HNVM is a hierarchical recurrent neural network that enables it to capture both user's long-term and short-term preferences. Experiments on two public datasets demonstrate that HNVM outperforms state-of-the-art sequential recommendation methods.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Sequential Recommendation; Recurrent Networks; Generative Model

ACM Reference Format:
Teng Xiao, Shangsong Liang, and Zaiqiao Meng. 2019. Hierarchical Neural Variational Model for Personalized Sequential Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313603

1 INTRODUCTION

Recommender Systems aim at helping users to find interesting items from rich candidates. Most traditional methods [7, 9, 18, 24, 37] are based on matrix completion where the goal is to predict the missing value. In most real scenarios, a user's actions are sequential, which can reflect the user's short-term preference and item's sequential information [25]. However, pervious methods [7, 9, 18, 24] based on matrix completion neglect sequential patterns for recommendation.

To capture users' sequential patterns, a number of methods called session-based recommendation methods [3, 10, 11, 16, 19, 34] built based on Recurrent Neural Networks (RNN) have been proposed. Session-based recommendation aims at predicting the next item that the user will interact with based on on pervious interactions with items inside the current session. Despite the appeal of these methods, they all assume that users are anonymous. However, users' profiles in many application platforms such as YouTube and Linkedin are available. In these platforms, a user's pervious session and profile can be utilized for capturing user's long-term and general preferences [15, 32] so as to personalize recommendation results. As unveiled by Jannach et al. [15, 22], both user's short-term and long-term preferences are crucial to improve recommendation performance. However, recent session-based methods [10, 11, 19] only consider user's most recent actions (short-term preference) in the current session, and neglect user's past session behavior (long-term preference) and users' profiles (general preference).

To consider both user's short-term and long-term preferences, some recent sequential models have been proposed [5, 13, 22, 26, 39]. These methods demonstrate a number of major drawbacks: (a) They do not model user's general (unchanged over time) preference corresponding to a user's profile, e.g., his id and demographics which can be leveraged to improve performance especially in job recommendation websites, e.g., Linkedin, (b) they don't model the relationship between user' past session and current session (with the exception of Quadrana et al [26]) , which is important for capturing users' long-term preferences, and (c) Recent work [26] is based on deterministic hierarchical recurrent neural networks which limit it to capture users' long-term preferences due to the restricted shallow generation process [30].

To address the aforementioned problems, we study the problem of sequential recommendation, which is to enhance personalization recommendation performances. We aim at jointly modeling users' general, long- and short-term preferences though a unified hierarchical deep generative model. Deep generative models [28, 40] have emerged as a powerful model to learn latent disentangled representations from data and improve the diversity of generative results [2, 14, 30, 41], while they have led to promising improvements in computer vision (CV) [14, 40] and neural language processing (NLP) tasks [4, 30, 41]. Specifically, we proposed a **H**ierarchical **N**eural **V**ariational **M**odel (HNVM), which models a hierarchical generative

*Shangsong Liang is the corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313603

process of three different preferences: general, long- and short-term preferences. Specifically, HNVM models user's general, long- and short-term preferences to be conditioned on user's profile, the past session and the current session, respectively. Our HNVM extends existing session-based recommendation in three directions: (a) The hierarchical generative process of HNVM makes itself to be able to effectively capture three kinds of preferences for a user, (b) It uses a two-level RNN network to model user's sequential pattern inside session and cross session, and (c) It is a hierarchical latent variables model, it is able to learn users' general and long-term preferences as latent Gaussian representations and can effectively handle the problem of restricted shallow generation process [26]. Our main contributions can be summarized as follows:

(1) We propose a novel hierarchical deep generative model which is the first model to jointly model different categories of preferences: general, long- and short- term preferences in the context of streams of sessions, such that user's profile and his past session information can be effectively captured for recommendation. (2) We propose a hierarchical stochastic gradient variational Bayes (SGVB) inference algorithm to jointly infer latent variables and estimate parameters. We derived a *conditional session − level variational lower bound* that leads to efficient Bayesian learning with back-propagation. (3) The experiment results demonstrate that our HNVM outperforms the state-of-the-art methods for sequential recommendation task.

2 PRELIMINARIES

We first introduce the task of personalized sequential recommendation, and then review the session-based recommendation with recurrent neural network (RNN) that related most to our HNVM.

2.1 Notations and Problem Formulation

Let $\mathcal{U} = \left\{u_1, u_2, ..., u_{|\mathrm{U}|}\right\}$ and $\mathcal{I} = \left\{i_1, i_2, ..., i_{|\mathrm{I}|}\right\}$ be sets of users and items, with $|\mathcal{U}|$ and $|\mathcal{I}|$ being the sizes, respectively. For each user u, let his sequential actions (sessions) sorted by time denote as a sequence of sessions $\mathbf{S}^u = \left\{\mathbf{s}_1^u, \mathbf{s}_2^u, ..., \mathbf{s}_{N_u}^u\right\}$, where N_u is the number of sessions for user u, $\mathbf{s}_n^u = \left\{i_{n,1}^u, i_{n,2}^u, ..., i_{n,|\mathbf{s}_n^u|}^u\right\}$ denotes the item sequence corresponding to the n-th session $\mathbf{s}_n^u$ of user u. Let $\mathbf{a}^u$ denote a user u's profile such as his attributes. The sequential recommendation can be formalized as a sequence prediction task: given a session sequence $\mathbf{S}^u = \left\{\mathbf{s}_1^u, \mathbf{s}_2^u, ..., \mathbf{s}_{N_u}^u\right\}$ and attribute $\mathbf{a}^u$ of user u, generate a sequence of items $\mathbf{s}_{N_u+1}^u$ that user u will interact with at session $N_u + 1$.

2.2 Session-based Recommendation with RNN

RNN has been widely used in session-based recommendation [10, 11, 19, 22, 23, 26, 32] due to it's power for modeling sequence data. However, traditional RNN suffers from vanishing gradient problem. To track this problem, two variant RNNs, namely Gated Recurrent Unit (GRU) [6] and Long Short Term Memory (LSTM) [12] networks have been proposed. Due to the simpler and less parameters, most session-based recommendation models [10, 11, 19, 22, 23, 26, 32] are built on GRU. The GRU takes as input the current item ID in the session and outputs a predicted score for next item in the session.

Given a sequence of actions (a session) $\mathbf{s}_n^u = \left\{i_{n,1}^u, i_{n,2}^u, \ldots, i_{n,|\mathbf{s}_n^u|}^u\right\}$ of user u. The session-based GRU network computes the following current hidden state vector $h_{n,t}^u$ conditional on the last hidden state vector and the current input item ID.

$$\mathbf{h}_{n,t}^u = GRU^{user}(\mathbf{h}_{n,t-1}^u, \mathbf{i}_{n,t}^u), \tag{1}$$

where the GRU^{user} represents the GRU unit [6] called user-level GRU. The $\mathbf{i}_{n,t}^u$ is a one-hot vector of current item ID $i_{n,t}^u$. The output of GRU at time t is the distributions of predicted scores for all items:

$$\hat{\mathbf{y}}_{n,t}^u = g(\mathbf{h}_{n,t}^u), \tag{2}$$

where $g(\cdot)$ is a ranking function such as softmax function. The network can be trained with several ranking loss function like cross-entropy or BPR [27] by stochastic gradient descent.

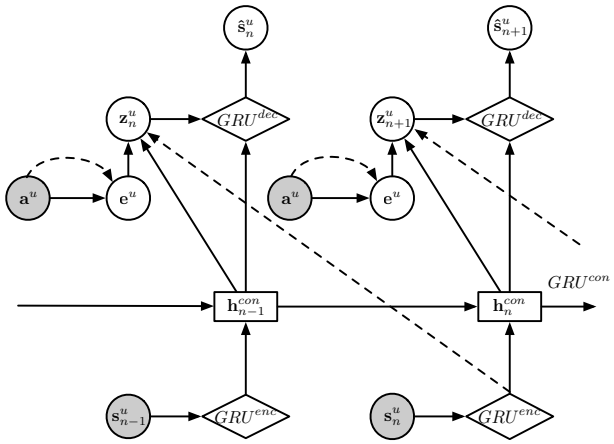


Figure 1: Graphical model of HNVM. Gray circles and white rounded boxes represent observed and latent variables, respectively. Bold solid and dashed lines represent generative and inference processes, respectively. For simplicity, we omit the latent states in encoder GRU and decoder GRU.

3 HIERARCHICAL NEURAL VARIATIONAL MODEL

We proposed a hierarchical neural variational model (HNVM) to model the session generative process for personalized recommendation. We will first introduce the hierarchical generative process, and then detail the hierarchical network in HNVM and the inference algorithm. The graphical model of HNVM is shown in Figure 1.

3.1 Generative Process

Our model HNVM can bee seen as a hierarchical generative model with a series stochastic latent variables. The goal of our model is to maximize the variational evidence lower bound [36] of the conditional log-likelihood of observed variables. For user u, the likelihood of his observed sessions is modeled as follows:

$$p(\mathbf{s}_1^u, \mathbf{s}_2^u, ..., \mathbf{s}_{N_u}^u) = \prod_{n=1}^{N_u} p(\mathbf{s}_n^u | \mathbf{s}_{<n}^u). \tag{3}$$

Due to each session $\mathbf{s}_n^u$ is a sequence of items $(i_{n,1}^u, i_{n,2}^u, ..., i_{n,3}^u)$, Eq. 3 can be further formulated as follows:

$$p(\mathbf{s}_1^u, \mathbf{s}_2^u, ..., \mathbf{s}_{N_u}^u) = \prod_{n=1}^{N_u} \prod_{t=1}^{|\mathbf{s}_n^u|} p(i_{n,t}^u | \mathbf{s}_{<n}^u, i_{n,<t}^u). \tag{4}$$

Our HNVM is a latent variable model which contains a series of latent variables $\mathbf{z}_n^u$ for each session $t = 1, ..., N_u$ in terms of user u. In order to incorporate the previous session information of user u into the next session, we first consider the prior of latent variable $\mathbf{z}_n^u$ (long-term preference) conditioned on all the past observed sessions (the session context) and the general preference $\mathbf{e}^u$:

$$\begin{aligned} p_\theta(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{e}^u) &= \mathcal{N}(\mathbf{z}_n^u | f_{\mu_z}(\mathbf{s}_{<n}^u, \mathbf{e}^u), f_{\sigma_z^2}(\mathbf{s}_{<n}^u, \mathbf{e}^u)) \\ &= \mathcal{N}(\mathbf{z}_n^u | \mu_{z_n^u}, diag(\sigma_{z_n^u}^2)), \end{aligned} \tag{5}$$

where $\mathcal{N}(\mu, \Sigma)$ is the multivariate normal distribution with mean $\mu \in \mathbb{R}^{D_z}$ and diagonal covariance matrix $\Sigma \in \mathbb{R}^{D_z \times D_z}$. The conditional prior of the long-term preference $\mathbf{z}_n^u$ is the multivariate Gaussian distribution whose mean and diagonal covariance matrix are parameterized by neural networks $f_{\mu_z}(\cdot)$ and $f_{\delta_z^2}(\cdot)$ with input $\mathbf{s}_{<n}^u$ and $\mathbf{e}^u$. Here, $\mathbf{e}^u$ denotes the latent embedding of user u which represents his general preference and the conditional prior of $\mathbf{e}^u$ is:

$$\begin{aligned} p_\theta(\mathbf{e}^u | \mathbf{a}^u) &= \mathcal{N}(\mathbf{e}^u | f_{\mu_e}(\mathbf{a}^u), diag(f_{\sigma_e^2}(\mathbf{a}^u))) \\ &= \mathcal{N}(\mathbf{e}^u | \mu_{\mathbf{e}^u}, diag(\sigma_{\mathbf{e}^u}^2)), \end{aligned} \tag{6}$$

where the conditional prior of $\mathbf{e}^u$ is also the multivariate Gaussian distribution whose mean and diagonal covariance matrix are parameterized by neural networks $f_{\mu_e}(\cdot)$ and $f_{\delta_e^2}(\cdot)$ with input $\mathbf{a}^u$. Note that we utilize two latent variables $\mathbf{e}^u$ and $\mathbf{z}_n^u$ to represents a user's general and long-term preferences, respectively.

We model the current session $\mathbf{s}_n^u$ to be generative from the long-term preference $\mathbf{z}_n^u$ and the user's context $\mathbf{s}_{<n}^u$:

$$p_\theta(\mathbf{s}_n^u | \mathbf{z}_n^u, \mathbf{s}_{<n}^u) = \prod_{t=1}^{|\mathbf{s}_n^u|} p_\theta(i_{n,t}^u | \mathbf{z}_n^u, \mathbf{s}_{<n}^u, i_{n,<t}^u). \tag{7}$$

According the above generative process of our model, the conditional distribution can be factorized as:

$$\begin{aligned} p_\theta(\mathbf{S}, \mathbf{E}, \mathbf{Z} | \mathbf{A}) = \prod_u \prod_{n=1}^{N_u} \prod_{t=1}^{|\mathbf{s}_n^u|} p_\theta(i_{n,t}^u | \mathbf{s}_{<n}^u, i_{n,<t}^u, \mathbf{z}_n^u) \cdot \\ p_\theta(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{e}^u) p_\theta(\mathbf{e}^u | \mathbf{a}^u), \end{aligned} \tag{8}$$

where $\mathbf{S}$ and $\mathbf{A}$ denote the observed sessions and the attributes of all users, respectively. $\mathbf{E}$ and $\mathbf{Z}$ denote the sets of latent variables of $\mathbf{e}^u$ and $\mathbf{z}_n^u$ of all users, respectively. We use θ to denote the all parameters in the generative model. In what follows, we will detail the hierarchical generative network architecture of HNVM.

3.2 The Hierarchical Network Architecture

As shown in Figure 1, our model is hierarchical, which contains three RNN networks: the *encoder* RNN, the *context* RNN [29] and the *decoder* RNN. The *encoder* RNN encodes session $\mathbf{s}_n^u$ into a fixed-size vector:

$$\mathbf{h}_{n,t}^u = GRU^{enc}(\mathbf{h}_{n,t-1}^u, \mathbf{i}_{n,t}^u), \ t = 1, 2, .., |\mathbf{s}_n^u|, \tag{9}$$

where $\mathbf{h}_{n,t}^u$ is the hidden state vector at time t, $\mathbf{i}_{n,t}^u$ is the one-hot vector of item $i_{n,t}^u$, and GRU^{enc} represents the non-linear activation GRU function [6]. The last hidden vector $\mathbf{h}_{n,|\mathbf{s}_n^u|}^u$ can be viewed as the summary of current session $\mathbf{s}_n^u$.

For the *context* RNN, it encodes the session sequence as:

$$\mathbf{h}_n^{con} = GRU^{con}(\mathbf{h}_{n-1}^{con}, \mathbf{h}_{n,|\mathbf{s}_n^u|}^u), \ n = 1, 2, .., |\mathbf{S}^u|, \tag{10}$$

where the GRU^{con} denotes the non-linear function parameterized by *context* RNN and $\mathbf{h}_n^{con}$ is the hidden state of *context* RNN. $\mathbf{h}_n^{con}$ can be viewed as the context-level representation which is the summary of all the observed previous sessions ($\mathbf{s}_{\leq n}^u$).

For the *decoder* RNN, it takes $\mathbf{h}_{n-1}^{con}$ and $\mathbf{z}_n^u$ as input to generate the next session $\mathbf{s}_n^u$:

$$\begin{aligned} \mathbf{h}_{n,0}^{dec} = \mathbf{0}, \ \mathbf{h}_{n,t}^{dec} = GRU^{dec}(\mathbf{h}_{n,t-1}^{dec}, \mathbf{i}_{n,t}^u, \mathbf{h}_{n-1}^{con}, \mathbf{z}_n^u), \\ t = 1, 2, .., |\mathbf{s}_n^u|, \end{aligned} \tag{11}$$

and the generative items are sampled according to Eq. 2. In our HNVM, the *decoder* RNN captures sequential pattern inside the current session (i.e., short-term preference).

As shown in Figure 1, the generative process (§ 3.1) of a session sequence for user u through the hierarchical network are:

(1) The *encoder* RNN encodes the $(n-1)$-th session ($\mathbf{s}_{n-1}^u$) of user u into a fix-size vector $\mathbf{h}_{n-1,|\mathbf{s}_n^u|}^{enc}$, which is taken by the *context* RNN as input to compute the context hidden state $\mathbf{h}_{n-1}^{con}$ (Eq. 10).

(2) Sample the user latent embedding $\mathbf{e}^u$ from conditional prior distribution (Eq. 6).

(3) A two-layer Multi Layer Perceptron (MLP) takes as input the concatenation of both $\mathbf{h}_{n-1}^{con}$ and the general preference $\mathbf{e}^u$. The network output is transformed to give the mean and diagonal covariance matrix of $\mathbf{z}_n^u$ in (Eq. 5) as:

$$\begin{bmatrix} \mu \\ \log(\sigma_{\mathbf{z}_n^u}) \end{bmatrix} = \text{MLP}(\mathbf{h}_{n-1}^{con}, \mathbf{e}^u). \tag{12}$$

(4) Draw latent variable $\mathbf{z}_n^u$ from the prior $p_\theta(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{e}^u)$ whose mean and covariance matrix provided by step (3).

(5) The *decoder* RNN takes context vector $\mathbf{h}_{n-1}^{con}$ and $\mathbf{z}_n^u$ as input to generative next session $\mathbf{s}_n^u$ of user u (Eq. 7).

3.3 Inference Process

As shown in graph model Figure 1, the goal of HNVM is to infer the latent variables $\mathbf{e}^u$ and $\mathbf{z}_n^u$ for every users and all their sessions, and estimate the network parameter θ. Due to the latent variables are hierarchical and the generative process is parameterized by the neural network, it is intractable to infer these latent variables and estimate network parameters by using traditional mean-filed approximation [36]. Inspired by Deep Latent Gaussian models (DLGm) [28, 31], we propose a hierarchical conditional stochastic gradient variational Bayes (SGVB) algorithm to infer our HNVM. For our model, we consider the following type of variational distribution factorization structure:

$$q_\phi(\mathbf{E}, \mathbf{Z} | \mathbf{S}, \mathbf{A}) = \prod_u q_\phi(\mathbf{e}^u | \mathbf{a}^u) \prod_{n=1}^{N_u} q_\phi(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{s}_n^u), \tag{13}$$

$$q_\phi(\mathbf{e}^u | \mathbf{a}^u) = \mathcal{N}(\mathbf{u} | g_{\mu_e}(\mathbf{a}^u), diag(g_{\sigma_e^2}(\mathbf{a}^u))) \tag{14}$$

$$= \mathcal{N}(\mathbf{e}^u | \tilde{\mu}_{\mathbf{e}^u}, diag(\tilde{\sigma}_{\mathbf{e}^u}^2)), \tag{15}$$

$$q_\phi(\mathbf{z}_n^u | \mathbf{s}_{<n}^u, \mathbf{s}_n^u) = \mathcal{N}(\mathbf{z}_n^u | g_{\mu_z}(\mathbf{s}_{<n}^u, \mathbf{s}_n^u), g_{\sigma_z^2}(\mathbf{s}_{<n}^u, \mathbf{s}_n^u)) \tag{16}$$

$$= \mathcal{N}(\mathbf{e}^u | \tilde{\mu}_{\mathbf{z}_n^u}, diag(\tilde{\sigma}_{\mathbf{z}_n^u}^2)). \tag{17}$$

The posteriors over $\mathbf{e}^u$ and $\mathbf{z}_n^u$ are all multivariate diagonal Gaussian distributions. Similar to the notations in generative model, we let $g_{\mu_e}(\cdot)$, $g_{\sigma_e^2}(\cdot)$, $g_{\mu_z}(\cdot)$ and $g_{\sigma_z^2}(\cdot)$ be neural networks whose parameters are denoted by ϕ. This factorization assumes the general preference $\mathbf{e}^u$ is only conditioned on u's attribute $\mathbf{a}^u$, whereas the long-term preference $\mathbf{z}_n^u$ is conditioned on $\mathbf{s}_{<n}^u$ which is the *context* RNN hidden state $\mathbf{h}_{n-1}^{con}$, and current session summary $\mathbf{h}_{n,|\mathbf{s}_n^u|}^u$. To infer latent variables and estimate network parameters, we maximize the hierarchical variational **E**vidence **L**ower **BO**und (ELBO) of the conditional log likelihood (i.e., Eq. 8):

$$\log p_\theta(\mathbf{S}|\mathbf{A}) \geq \sum_u \mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u) = \sum_u \sum_{n=1}^{N_u} \mathbb{E}_{q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)}\Big[\log p_\theta(\mathbf{s}_n^u|\mathbf{z}_n^u,\mathbf{s}_{<n}^u)\Big] - \mathbb{E}_{q_\phi(\mathbf{e}^u|\mathbf{a}^u)}\Big[\mathrm{KL}(q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)||p_\theta(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{e}^u)\Big] - \mathrm{KL}(q_\phi(\mathbf{e}^u|\mathbf{a}^u)||p_\theta(\mathbf{e}^u|\mathbf{a}^u)), \tag{18}$$

where $\mathrm{KL}(q||p)$ is the Kullback-Leibler (KL) divergence between distributions p and q. The derivation of the ELBO in Eq. 18 is included in Appendix A. Maximizing the ELBO (Eq. 18) is equivalent to maximize the log-likelihood of observed variables and to make the variational distributions $q_\phi(\mathbf{e}^u|\mathbf{a}^u)$ and $q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)$ of each user to be as close as possible to their intractable true posteriors $p_\theta(\mathbf{e}^u|\mathbf{S}^u,\mathbf{a}^u)$ and $p_\theta(\mathbf{z}_n^u|\mathbf{S}^u,\mathbf{a}^u)$, respectively. The last KL term in Eq. 18 has analytical form. However, for the two expectation terms, we can not compute them analytically. To solve this problem, we approximate the two terms using Monte Carlo sampling, and use suitable reparameterization trick [17, 28] with low variance.

Since the variational distribution of $\mathbf{e}^u$ does not depend on the entire session sequence $\mathbf{S}^u$, the *conditional sequence − level variational lower bound* $\mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u)$ can be decomposed into the sum of $\mathcal{L}(\theta,\phi;\mathbf{s}_n^u|\mathbf{a}^u)$, the *conditional session − level variational lower bound*, and the function $\mathcal{F}_{\theta,\phi}(\mathbf{a}^u)$ w.r.t θ and ϕ (see Eq. 22 in Appendix A):

$$\mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u) = \sum_{n=1}^{N_u} \mathcal{L}(\theta,\phi;\mathbf{s}_n^u|\mathbf{a}^u) + \mathcal{F}_{\theta,\phi}(\mathbf{a}^u). \tag{19}$$

Instead of sampling a batch at the sequence-level to maximize the ELBO, we can sample a batch at the session-level to maximize it. This method makes our model scalable when the session sequence $\mathbf{S}^u$ is extremely long, such that computing an entire sequence for a batched update is very computationally expensive.

4 EXPERIMENTAL SETUP

4.1 Research Questions

The main research questions guiding the paper are: (**RQ1**) Does HNVM outperform state-of-the-art sequential recommendation methods? (**RQ2**) How does the captured the long-term and general preferences impact recommendation accuracy? (**RQ3**) What is the impact of the number of users' past sessions for recommendation performance? (**RQ4**) Is HNVM effective on the other task like next-session recommendation?

4.2 Datasets and Data Preprocessing

In order to answer our research questions, we work with two publicly available datasets: RecSys Challenge 2015[1] (RSC) [10] and IJCAI-15 Competition datasets[2] (IJC) [26]. RSC dataset contains interactions on job posting for 770K users over a 80-days' period. IJC contains users' shopping log over six months' period in online shopping site Tmall. For RSC dataset, we first partitioned the interaction data into sessions by considering 30-minutes' idle threshold and removed interactions having type 'delete'. We hold out the last interacted item in the last session of each user as the test data, and the item before the last item as the validation set. The rest items are treated as training dataset. For the IJC dataset, we follow the the preprocessing procedure as same as that in [39]. For simplicity, we just use user's id number's one-hot vector as static profile representation $\mathbf{a}^u$ (Note that we don't use other attributes such age or gender, because there is no such information in IJC dataset). Table 1 shows the statistics of the two datasets after preprocessing.

Table 1: Statistics of datasets.

Datasets	RSC	IJC
Users	11,479	20,648
Items	59,297	25,129
Sessions	89,591	75,426
Training sessions	78,276	71,892
Test sessions	11,315	3,635
Interactions per session	6.1	2.73

4.3 Baselines and Settings

We make comparisons between the proposed HNVM model and the following state-of-the-art algorithms for sequential recommendation: (1) **POP**: This method always recommends the most-popular (the largest number of interactions in the training set) items. (2) **Item-KNN**: This method recommends items based on the cosine similarity between items. The similarity is defined by the co-occurrence number between items in sessions. (3) **BPR**: This method [27] optimizes a pair-wise ranking loss for general recommendation. (4) **NCF**: This method [9] is the state-of-the-art general recommendation method. (5) **GRU4Rec**: This model [10] utilizes the basic GRU [6] unit to the task of session-based recommendation. (6) **HRNN**: This model [26] utilizes past session information through a hierarchical RNN. We use it's HRNN-*All* version in our experiments. (7) **Caser**: This is the state-of-the-art sequential recommendation model [35] which uses convolutional filters to learn sequential patterns. It also captures user's general preference. (8) **STAMP**: This model [22] utilizes a short-term attention/memory mechanism to capture user's long- and short-term preferences.

For fair comparisons, all GRU based baselines (GRU4Rec, HRNN) including our HNVM use the BPR loss function. For all baselines, we use a grid search to find the optimal parameters using the validation set. For HNVM, we randomly initialize parameters using uniform distribution with variance 0.01 and bases equal 0. We use one-layer GRU for encoder, context and decoder RNN. The hyper-parameters are found via extensive grid search. We use AdaGrad [8] with momentum to optimize our model. The batch size is set to 256. The

[1] http://2016.recsyschallenge.com/

[2] https://ijcai-15.org/index.php/repeat-buyers-prediction-competition

hidden states of GRU are all set to 200. Since the high variance leads the gradients not stable, we multiple the diagonal covariance matrices of the prior and posterior distributions by 0.05. We use a dropout regularization [33] on the hidden states of GRU with a fixed dropout rate of 0.1. Similar to [1], we multiply the KL terms in Eq. 18 by scalar, which starts from 0 and linearly increases to 1.

4.4 Evaluation Metrics

Similar to that in [13, 26], we perform our model and the baselines on the next-item sequential recommendation task for comparisons. We use the widely used evaluation metrics: Recall@k (Recall at k), MRR@k (Mean Reciprocal Rank at k) [21] and Pre@k (Precision at k) [20, 38]. We compute the scores at depth 5, i.e., let k =5 for evaluation.

5 RESULTS AND ANALYSIS

5.1 Overall Performance (RQ1)

To start, for question **RQ1**, to evaluate the effectiveness of HNVM in personalized sequential recommendation, we examine the performance of HNVM on the next-item recommendation task. Table 2 shows the overall performance of all methods in terms of Recall@5, MRR@5 and Pre@5. From Table 2, we have the following conclusions: (1) Methods capturing users' short-term preferences like GRU4Rec, HRNN, Caser, STAMP and HNVM work better than general methods, i.e., BRP and NCF, which illustrates users' short-term preferences need to be captured for next-item recommendation task. (2) On the RSC dataset, our model HNVM and Caser outperform other sequential recommendation models, i.e., GRU4Rec and STAMP. The reason is that the user's static profile is more important in job search platform (RSC) than shopping or video platforms (IJC) and both our model HNVM and Caser can capture users' general preferences corresponding to users' profile. (3) Although our HNVM and Caser both can capture users' general and short-term preferences, our HNVM works better than Caser. The reason is that user has different purchase purpose in different sessions and Caser doesn't model this cross-session relationship (i.e., long-term preferences). (4) Our HNVM significantly outperforms HRNN, which demonstrates that the latent variable characteristic makes our model to be able to more effectively capture the users' long-term preferences. All of these findings demonstrate that HNVM can effectively capture users' different preferences and it is able to maintain significant improvements of sequential recommendation performance over the state-of-the-arts baseline methods.

5.2 Impact of the Long-term and General Preferences (RQ2)

Next, we turn to **RQ2**. For the purpose of analyzing the impact of user's long-term and general preferences on the next-item recommendation task, we spilt every sessions by the order of interaction within the sessions. We only consider sessions which have ≥ 5 interactions on RSC datasets (6736 sessions). We group interactions in these sessions as three groups: *Begin*, *Middle* and *End*. The *Begin* group contains the first 2 interactions in these sessions, the *Middle* contains 3rd -4th and the *End* contains the rest of interactions. We evaluate recommendation performance of our HNVM on these three groups, respectively. Since sessions' lengths in the IJC dataset are general very short, we only conduct this analysis on the RSC dataset. Figure 2 shows the Recall@5 and MRR@5 results with different positions of interactions on RSC datasets (the performance pattern in terms of Pre@5 is similar to that in terms of Recall@5, and thus we omit it here due to space limitations). From the results, some interesting findings can be observed: (1) For all RNN-based methods, i.e., HNVM, HRNN and GRU4Rec, the performance increases with the number of pervious items in the session, which demonstrates that these methods can both effectively capture users' short-term preference within the current session. (2) Personalized recommendation models, HNVM, HRNN and Caser significantly outperform vanilla RNN-based methods like GRU4Rec and STAMP by a large margin, and the margin is larger on the *Begin* group than that on the *Middle* group. The reason is that GRU4Rec and STAMP do not incorporate any user's past session information (i.e., long-term preferences) when user starts a new session and these models can hardly capture the user's short-term preference when she/he just clicks a few items. (3) Our HNVM significantly and consistently outperforms the best two baselines, HRNN and Caser, especially in the *Begin* and *End* groups, which illustrates that capturing both users' long-term (past session information) and general (profile information) preferences can significantly contribute to predict the first several actions and the last several actions in long sessions.

Table 2: Recommendation performance comparison on the RSC and IJC datasets in terms of Recall@5, MRR@5 and Pre@5. "Improv." denotes the improvement of HNVM relative to the best baseline.

	RSC			IJC		
	Recall	MRR	Pre	Recall	MRR	Pre
POP	.0253	.0174	.0084	.0049	.0025	.0009
Item-KNN	.0697	.0406	.0139	.0126	.0141	.0018
BPR	.0714	.0497	.0159	.0154	.0158	.0067
NCF	.0847	.0586	.0184	.0174	.0166	.0098
GRU4Rec	.1292	.0799	.0258	.0180	.0159	.0141
HRNN	.1356	.0832	.0284	.0325	.0265	.0157
Caser	.1422	.0921	.0301	.0344	.0254	.0182
STAMP	.1337	.0802	.0267	.0321	.0234	.0171
HNVM	**.1521**	**.0991**	**.0357**	**.0371**	**.0278**	**.0197**
Improv.	+6.9%	+7.6%	+18.6%	+7.8%	+4.9%	+8.2%

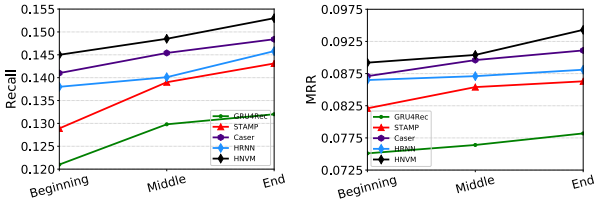


Figure 2: Performance comparison in terms of Recall@5 and MRR@5 on RSC dataset with different positions in sessions.

5.3 The Number of Past Sessions (RQ3)

We now examine the impact of the number of user's past sessions for performance. We partitioned users on RSC dataset into two groups based on the number of their past sessions: *Short* and *Long* groups. The *Short* group refers the number of user's sessions ≤ 6

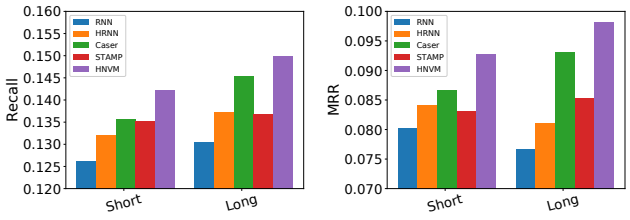


Figure 3: Performance comparison in terms of Recall@5 and MRR@5 on RSC datasets with different past session length.

Table 3: Next-session recommendation performance comparison in terms of Recall@5, MRR@5 and Pre@5.

	RSC			IJC		
	Recall	MRR	Pre	Recall	MRR	Pre
GRU4Rec	.0732	.0387	.0098	.0122	.0101	.0095
HRNN	.1054	.0607	.0149	.0213	.0198	.0157
Caser	.0987	.0531	.0128	.0187	.0177	.0139
STAMP	.0872	.0412	.0114	.0162	.0148	.0118
HNVM	**.1134**	**.0696**	**.0154**	**.0221**	**.0209**	**.0164**
Improv.	+7.6%	+14.7%	+3.4%	+3.8%	+5.6%	+4.5%

(67%), while the *Long* refers to > 6 (33%). We evaluate recommendation performance based on the two groups, respectively. As showed in Figure 3, we can observe that the performance of HRNN, Caser and HNVM on the *Long* group is better than that on the *Short* group. The reason is that HRNN, Caser and HNVM not only capture users' short-term preferences in the current session but also consider users' general (profile) or long-term (past session information) preferences , and the more the user past sessions are observed, the better the model learns the users' long-term preferences. Figure 3 shows that our model significantly outperforms the best baselines (i.e., HRNN and Caser) on next-item recommendation task.

5.4 Performance Comparison on the Next-session Recommendation Task (RQ4)

To further evaluate our model, similar to [13], we consider another sequential recommendation task: next-session recommendation. We use user's last session as test set, the session before the last as validation set and the rest of sessions as training set. Table 3 shows the next-session performance between HNVM with the best baselines. From Table 3, we can observe our HNVM and HRNN significantly outperform basic session-based methods, i.e., STAMP and GRU4Rec. The reason is that both our HNVM and HRNN effectively utilize users' past sessions and capture users' long-term preferences and these long-term preferences help models mining users' different purpose at different sessions. The finding that our HNVM outperforms HRNN indicts that the users' latent long-term preferences learnt by HNVM work better than those of HRNN.

6 CONCLUSION

In this paper, we have studied the problem of personalized sequential recommendation. To track this problem, we have proposed a hierarchical neural variational model, abbreviated as HNVM, that is the first attempt to capture users' different level preferences, i.e., general, long- and short-term preferences through a unified hierarchical generative process. HNVM can effectively model user's sequential patterns inside sessions and across sessions through it's hierarchical network structure. Our HNVM learns users' different preferences via a hierarchical Gaussian latent representations, which make it be able to capture better preferences than other deterministic model. To effectively infer our model, we have proposed hierarchical stochastic gradient variational Bayes inference algorithm. We also have derived a *conditional session − level variational lower bound*, which makes our HNVM be able to be effectively learned via sampling mini-batches at session-level. Experimental results on two publicly available datasets demonstrate the effectiveness of the proposed algorithms.

A DERIVATION OF CONDITIONAL SESSION-LEVEL VARIATIONAL LOWER BOUND

All notations below are defined in the body of the paper. The conditional variational ELBO for user u can be derived as follows:

$$\log p_\theta(\mathbf{S}^u|\mathbf{a}^u) \geq \mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u) =$$

$$\mathbb{E}_{q_\phi(\mathbf{e}^u,\mathbf{z}_n^u|\mathbf{S}^u)} \log \frac{p_\theta(\mathbf{e}^u|\mathbf{a}^u)\prod_{n=1}^{N_u} p_\theta(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{e}^u)p_\theta(\mathbf{s}_n^u|\mathbf{z}_n^u,\mathbf{s}_{<n}^u)}{q_\phi(\mathbf{e}^u|\mathbf{a}^u)\prod_{n=1}^{N_u} q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)}$$

$$= \sum_{n=1}^{N_u} \mathbb{E}_{q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)}\Big[\log p_\theta(\mathbf{s}_n^u|\mathbf{z}_n^u,\mathbf{s}_{<n}^u)\Big]$$

$$- \sum_{n=1}^{N_u} \mathbb{E}_{q_\phi(\mathbf{e}^u|\mathbf{a}^u)}\Big[\mathrm{KL}(q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)||p_\theta(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{e}^u)\Big] \quad (20)$$

$$- \mathrm{KL}(q_\phi(\mathbf{e}^u|\mathbf{a}^u)||p_\theta(\mathbf{e}^u|\mathbf{a}^u)), \quad (21)$$

where the expected KL terms in Eq. 20 are two Gaussian distributions, and the KL divergence $\mathrm{KL}(q_\phi(\mathbf{e}^u|\mathbf{a}^u)||p_\theta(\mathbf{e}^u|\mathbf{a}^u))$ in Eq. 21 can be computed analytically. Thus the hierarchical ELBO for a user u's sessions' sequence $\mathbf{S}^u$ as follows:

$$\mathcal{L}(\theta,\phi;\mathbf{S}^u,\mathbf{a}^u) = \sum_{n=1}^{N_u} \mathbb{E}_{q_\phi(\mathbf{z}_n^u|\mathbf{s}_{<n}^u,\mathbf{s}_n^u)}\Big[\log p_\theta(\mathbf{s}_n^u|\mathbf{z}_n^u,\mathbf{s}_{<n}^u)\Big]$$

$$+ \mathbb{E}_{q_\phi(\mathbf{e}^u|\mathbf{a}^u)}\Big[\frac{1}{2}\sum_{d=1}^{D_z} 1 + \log\frac{\tilde{\sigma}^2_{\mathbf{z}_n^u,d}}{\sigma^2_{\mathbf{z}_n^u,d}} - \frac{(\tilde{\mu}_{\mathbf{z}_n^u,d}-\mu_{\mathbf{z}_n^u,d})^2+\tilde{\sigma}^2_{\mathbf{z}_n^u,d}}{\sigma^2_{\mathbf{z}_n^u,d}}\Big]$$

$$+ \frac{1}{2}\sum_{d=1}^{D_e}(1 + \log\frac{\tilde{\sigma}^2_{\mathbf{e}^u,d}}{\sigma^2_{\mathbf{e}^u,d}} - \frac{(\tilde{\mu}_{\mathbf{e}^u,d}-\mu_{\mathbf{e}^u,d})^2+\tilde{\sigma}^2_{\mathbf{e}^u,d}}{\sigma^2_{\mathbf{e}^u,d}})$$

$$= \sum_{n=1}^{N_u} \mathcal{L}(\theta,\phi;\mathbf{s}_n^u|\mathbf{a}^u) + \mathcal{F}_{\theta,\phi}(\mathbf{a}^u), \quad (22)$$

where the two expectation terms can not be solved analytically. Therefore, we apply Monte Carlo sampling and the reparametrization trick [17] to approximate the two terms. The samples are:

$$\mathbf{e}^u = \tilde{\mu}_{\mathbf{e}^u} + \tilde{\sigma}_{\mathbf{e}^u} \odot \epsilon_{\mathbf{e}^u}, \quad \epsilon_{\mathbf{e}^u} \sim \mathcal{N}(\mathbf{0},\mathbf{I}_{D_e}), \quad (23)$$

$$\mathbf{z}_n^u = \tilde{\mu}_{\mathbf{z}_n^u} + \tilde{\sigma}_{\mathbf{z}_n^u} \odot \epsilon_{\mathbf{z}_n^u}, \quad \epsilon_{\mathbf{z}_n^u} \sim \mathcal{N}(\mathbf{0},\mathbf{I}_{D_z}), \quad (24)$$

where $\odot$ denotes an element-wise product and $\mathcal{N}(\mathbf{0},\mathbf{I})$ is the normal Gaussian distribution with mean and covariance being $\mathbf{0}$ and $\mathbf{I}$, respectively.

REFERENCES

[1] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 10–21.

[2] Kris Cao and Stephen Clark. 2017. Latent Variable Dialogue Models and their Diversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, Vol. 2. 182–187.

[3] Sotirios P Chatzis, Panayiotis Christodoulou, and Andreas S Andreou. 2017. Recurrent Latent Variable Networks for Session-Based Recommendation. In *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems*. ACM, 38–45.

[4] Hongshen Chen, Zhaochun Ren, Jiliang Tang, Yihong Eric Zhao, and Dawei Yin. 2018. Hierarchical Variational Memory Network for Dialogue Generation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1653–1662.

[5] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 108–116.

[6] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).

[7] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.

[8] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. *ICLR* (2016).

[11] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 241–248.

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[13] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 505–514.

[14] Unnat Jain, Ziyu Zhang, and Alexander G Schwing. 2017. Creativity: Generating Diverse Questions using Variational Autoencoders.. In *CVPR*. 5415–5424.

[15] Dietmar Jannach, Lukas Lerche, and Michael Jugovac. 2015. Adaptation and evaluation of recommendations for short-term shopping goals. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 211–218.

[16] Dietmar Jannach and Malte Ludewig. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 306–310.

[17] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[18] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

[19] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*. 1419–1428.

[20] Shangsong Liang, Ilya Markov, Zhaochun Ren, and Maarten de Rijke. 2018. Manifold learning for rank aggregation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. 1735–1744.

[21] Shangsong Liang, Xiangliang Zhang, Zhaochun Ren, and Evangelos Kanoulas. 2018. Dynamic embeddings for user profiling in twitter. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1764–1773.

[22] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1831–1839.

[23] Pablo Loyola, Chen Liu, and Yu Hirate. 2017. Modeling User Session and Intent with an Attention-based Encoder-Decoder Architecture. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 147–151.

[24] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.

[25] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-Aware Recommender Systems. *ACM Comput. Surv.* 51, 4 (July 2018).

[26] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 130–137.

[27] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[28] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *International Conference on Machine Learning*. 1278–1286.

[29] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models.. In *AAAI*, Vol. 16. 3776–3784.

[30] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[31] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*. 3483–3491.

[32] Gabriele Sottocornola, Panagiotis Symeonidis, and Markus Zanker. 2018. Session-based News Recommendations. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 1395–1399.

[33] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[34] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 17–22.

[35] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 565–573.

[36] Martin J Wainwright, Michael I Jordan, and others. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning* 1, 1–2 (2008), 1–305.

[37] Teng Xiao, Shangsong Liang, Weizhou Shen, and Zaiqiao Meng. 2019. Bayesian Deep Collaborative Matrix Factorization. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019)*. AAAI.

[38] Christopher C Yang. 2010. Search engines information retrieval in practice. *Journal of the American Society for Information Science and Technology* 61, 2 (2010), 430–430.

[39] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential Recommender System based on Hierarchical Attention Networks. In *the 27th International Joint Conference on Artificial Intelligence*.

[40] Li Yingzhen and Stephan Mandt. 2018. Disentangled sequential autoencoder. In *International Conference on Machine Learning*. 5656–5665.

[41] Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 654–664.

Focusing Attention Network for Answer Ranking

Yufei Xie
Qiniu AtLab
Shanghai, China
xieyufei@qiniu.com

Shuchun Liu
Qiniu AtLab
Shanghai, China
liushuchun@qiniu.com

Tangren Yao
Qiniu AtLab
Shanghai, China
yaotangren@qiniu.com

Yao Peng
Qiniu AtLab
Shanghai, China
pengyao@qiniu.com

Zhao Lu
East China Normal University
Shanghai, China
yufeixie@ica.stc.sh.cn

ABSTRACT

Answer ranking is an important task in Community Question Answering (CQA), by which "Good" answers should be ranked in the front of "Bad" or "Potentially Useful" answers. The state of the art is the attention-based classification framework that learns the mapping between the questions and the answers. However, we observe that existing attention-based methods perform poorly on complicated question-answer pairs. One major reason is that existing methods cannot get accurate alignments between questions and answers for such pairs. We call the phenomenon "attention divergence". In this paper, we propose a new attention mechanism, called **F**ocusing **A**ttention **N**etwork(FAN), which can automatically draw back the divergent attention by adding the semantic, and metadata features. Our Model can focus on the most important part of the sentence and therefore improve the answer ranking performance. Experimental results on the CQA dataset of SemEval-2016 and SemEval-2017 demonstrate that our method respectively attains 79.38 and 88.72 on MAP and outperforms the Top-1 system in the shared task by 0.19 and 0.29.

CCS CONCEPTS

• **Computing methodologies** → **Lexical semantics**; *Information extraction*; Discourse, dialogue and pragmatics.

KEYWORDS

Answer Ranking; Focusing Attention Network; Semantic Features; Metadata Features

ACM Reference Format:
Yufei Xie, Shuchun Liu, Tangren Yao, Yao Peng, and Zhao Lu. 2019. Focusing Attention Network for Answer Ranking. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313518

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313518

1 INTRODUCTION

Online community forums, such as Yahoo! Answers [1] and Stack Overflow [2], have attracted many users. These users can ask or answer questions, and these archived questions and answers are very valuable. However, some of these questions have many answers, and most of these answers are low quality. Thus, *answer selection* of community question answering, which could automatically find good ones from a large number of answers, has drawn much attention in the field of natural language processing [3, 28, 34]. There are related tasks at SemEval-2016 [1] and SemEval-2017 [18].

In this paper, we focus on the task of ranking good answers ahead of bad ones, which is another form of answer selection. We illustrate an example of Qatar-Living dataset [18] from the SemEval-2017 Task 3 in Figure 1. An original question Q is shown in Figure 1(a). For convenience, we only choose three of the ten answers of question Q and show in Figure 1(b). These answers are sorted by their submitted time. They have different human labels:"Good", "Potentially Useful" and "Bad". We need to train a model, which can give the score of each answer and the answers should be sorted by the score at last, as in Figure 1(c).

Previous work on answer ranking normally used feature engineering, linguistic tools, or external resources. For example, semantic features are based on WordNet in [34]. This model pairs semantically related words based on word semantic relations. In [28], the answer ranking problem is transformed to a syntactical matching between the question/answer parse trees.

In recent years, attention based models have shown advantages in representing natural language sentences [2]. It obtains great success in many NLP tasks such as machine translation, question answering and recognizing textual entailments. When building the representation of a sentence, some attention information is added to the hidden state. For example, in attention based recurrent neural networks models each time-step hidden representation is weighted by attention.

However, so far, these traditional attention based models were shown to be successful only in modeling relatively short query and answer pairs, roughly a sentence in length [22]. One of the reasons these models under-perform for complicated longer texts, is that longer answers often contain irrelevant information and the alignments estimated by the attention model are easily corrupted due to the complexity of sentences. In other words, the attention model

[1] https://answers.yahoo.com/
[2] http://stackoverflow.com/

Q:is there any place i can find scented massage oils in qatar?

(a) An example of question.

A1: Try Both ;) I'am just trying to be helpful. On a serious note - Please go there. you'll find what you are looking for.
Human: Potentially Useful
A2: What they offer?
Human: Bad
A3: Yes. It is right behind Kahrama in the National area.
Human: Good

(b) The original answers sort by their submitted time.

A3: Yes. It is right behind Kahrama in the National area.
Score: 5.0
A1: Try Both ;) I'am just trying to be helpful. On a serious note - Please go there. you'll find what you are looking for.
Score: 1.6
A2: What they offer?
Score: 0.5

(c) The resorted answers according to their scores.

Figure 1: An example question and its answers in CQA.

cannot accurately associate each question with the corresponding target region in the answers. We call this phenomenon *attention divergence*. That is the attention regions divergence in some degree from the proper regions of target terms in the answers. This motivates us to develop some mechanism to focus attention back on the right regions of the target terms in the answers. Therefore, we propose a Focusing Attention Network (**FAN**) to accurately identify the main question and answer terms to use for the joint representation by adding the semantic and metadata features. Semantic features, which can obtain the semantic similarities between a question and an answer; metadata features, which can capture characteristics that relevant answers tend to have. Both of them are very important features to represent the sentence and we call them focusing features.

We firstly obtain the question representation with bidirectional LSTMs, next we extract semantic features, metadata features from questions and their answers, then we obtain the attentive hidden states that are weighted by the question representation and above two kinds of features as answer representation. At last, we feed the question and answer representation to the softmax classifier for classification. The experimental results on SemEval-2016 Task3 and SemEval-2017 Task3 datasets show that our method outperforms the state-of-the-art work. The main contributions of our approach are three folds:

- We propose the concept of attention divergence, which explains the poor performance of existing attention based methods on complicated long question-answer pairs.
- We develop a novel method called **FAN** to solve the attention divergence problem, where in addition to the attention component existing in most existing methods, semantic features and metadata features is introduced, which can focus deviated attention back on the target areas.
- The experimental results have verified the effectiveness of our proposed model, and also provide a promising avenue to make full use of the **FAN** model for other tasks.

In the remainder of this paper, after discussing related work in Section 2, we introduce our model in Section 3. Then, we describe our experiments in Section 4. The conclusions and discussions are presented in Section 5.

2 RELATED WORK

Question-answer selection and ranking has been an active area of research for decades, presenting many solutions. For example, Filice [8] proposes a kernel-based method which uses the ranking features, heuristics features, thread-based features and stacking features. This method obtains the best result in SemEval-2017 Task 3 subtask A. The MAP value is 88.43. Feng [5] develops a ranking system based on traditional NLP features such as tf-idf cosine, longest common subsequence, word overlap and translation probability. This method also obtains a good result in SemEval-2017 Task 3 subtask A. The MAP value is 88.24. While these methods show effectiveness, they might ignore the sentence representation in the semantic space.

Most recently, various convolution neural networks (CNNs) and recurrent neural networks (RNNs) are employed in the task of answer ranking [20, 24, 26]. Neural networks based models are proposed to represent the sentence in a semantic space and then compare the question and answer candidates in this hidden space [26]. These models will not suffer from the availability of resources, the effort of feature engineering and the systematic complexity by introducing linguistic tools, such as parse trees and dependency trees [24]. Tan [24] utilizes a simple but efficient attention mechanism in order to generate the answer representation according to the question context. Wang [26] proposes an inner attention based recurrent neural network which adds attention information before RNN hidden representation and shows the advantage in representation. These methods have the characteristics of constructing relatively valuable vector representation and they improve the results in answer ranking task. However, these methods face the issue of information loss when obtaining the last output layer.

To tackle the issue of information loss, combining feature extraction and deep neural networks has attracted many researchers [12, 23, 35]. Wu [31] combines two different methods to represent question-comment pairs, i.e., supervised model using traditional features and convolutional neural network. Takahashi [23] proposes a deep neural network with acoustic features to solve the acoustic scene classification. Zhou [35] combines human engineered feature extraction as well as those learned by the deep neural network. Koreeda [12] proposes a method that combines neural network similarities and handcrafted comment plausibility features. Combining deep neural networks and traditional features have the ability of incorporating different kinds of information.

Different from the existing approaches, in this paper we introduce the semantic features and metadata features into the attention network, which can make the attention focus on the semantic and metadata of the sentences. Other than simply combines a traditional features and the output of the last layer of CNN [30].

Though attention divergence has been observed in attention training of speech recognition [11], where the authors proposed an MTL framework that combines CTC and attention network to handle this issue, our paper is the first work that formally puts forward the concept of attention divergence. Furthermore, we design a focus-mechanism to solve this problem in the answer ranking domain.

3 OUR MODEL

Figure 2 shows the architecture of our proposed FAN model. Specifically, we adopt the BiLSTM model to obtain the preliminary representations of the question and answer. To solve the problem of attention divergence, we extract semantic futures and metadata features to enrich the question representation and then obtain a focusing attention for answer representation. Finally, we employ the softmax classifier to make the classification. We describe the details of the model in this section.

3.1 Semantic Features

Semantic and metadata are the two important components of the sentence. In order to tackle the issue of attention divergences, semantic features are introduced into the task of answer ranking. We describe our approach of extracting semantic features from questions and their answers in details. In natural language processing (NLP), word embedding maps words or phrases from the vocabulary to vectors of real numbers. It involves a mathematical embedding from a space with one dimension per word to a continuous vector space with much lower dimension. Methods to generate this mapping include neural networks [17], dimensionality reduction on the word co-occurrence matrix [13–15], probabilistic models [9], and explicit representation in terms of the context in which words appear. For each piece of a text such as answer text and question body we constructed the centroid vector from the vectors of all words in that text.

$$centroid(w_1..w_n) = \frac{\sum_{i=1}^{n} w_i}{n} \tag{1}$$

We used various feature similarities calculated using the centroid word vectors on the question body and on the answer text, as well as on parts thereof:

Question to answer similarity. We assume that a good answer should have a centroid vector that is close to that for the question. We choose the question body to answer text vector similarities.

Dependency syntax tree based word vector similarity. We obtained the dependency syntax tree with the Stanford parser [4], and calculated similarities between centroid vectors of noun phrases from the answer text and the centroid vector of the noun phrases from the question body text. The assumption is that the same parts of dependency syntax tree between the question and the answer might be closer than other parts of dependency tree.

Aligned similarity. For each word in the question body, we chose the most similar word from the answer text and took the average of all the best word pair similarities as suggested in [25]

LDA topic similarity. We performed topic clustering using Latent Dirichlet Allocation (LDA) as implemented in the gensim toolkit [21]. We built topic models with 50 topics. The assumption here is that if the question and the answer share similar topics, they are more likely to be relevant to each other.

For the similarity measures mentioned above, we measure their cosine similarity as,

$$\frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|} \tag{2}$$

3.2 Metadata Features

Besides semantic features described above, we also used some metadata features:

Question length. If a question is longer, it may be more clear, which may help users to give a more relevant answer [6].

Answer length. The assumption here is that longer answers could bring more useful details [33].

Question to answer length. If a question is long and the answer is short, it may be less relevant.

3.3 Focusing Attention Network

In answer ranking, given a question $Q = \{q_1, q_2, q_3, ..., q_n\}$ where q_i is i-th word ,n is the question length, we can compute its representation in bidirectional LSTM architecture as follows:

$$X = D[q_1, q_2, ..., q_n] \tag{3}$$

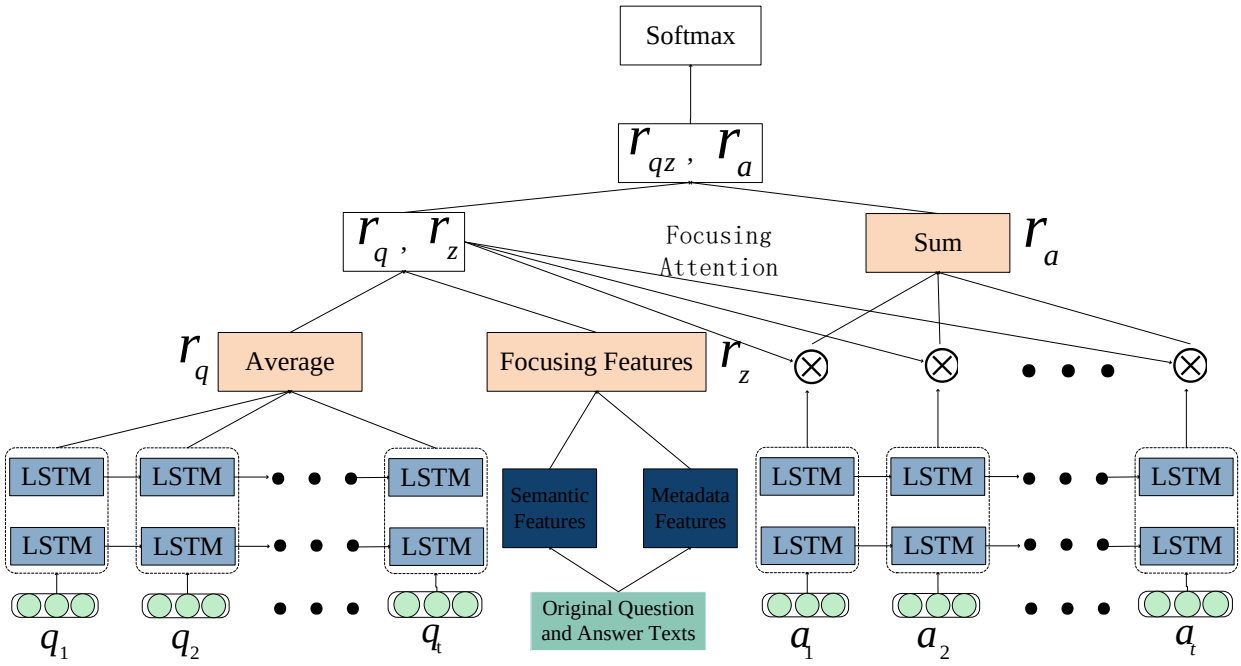


Figure 2: The architecture of our model.

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{4}$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{5}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{6}$$

$$\tilde{c_t} = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{7}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c_t} \tag{8}$$

$$h_t = o_t \odot \tanh(C_t) \tag{9}$$

where D is an embedding matrix that projects word to its embedding space in R^d; W, U are weight matrices and b are bias vectors; σ is the active function such as tanh.

In the LSTM architecture, there are three gates (input i, forget f and output o) and a cell memory vector c. The input gate can determine how incoming vectors x_t alter the state of the memory cell. The output gate can allow the memory cell to have an effort on the outputs. Finally, the forget gate allows the cell to remember or forget its previous state. Single direction LSTMs suffer a weakness of not utilizing the contextual information from the future tokens. Bidirectional LSTM utilizes both the previous and future context by processing the sequence on two directions, and generates two independent sequences of LSTM output vectors. One processes the input sequence in the forward direction, while the other process the input in the reverse direction. The output at each time step is the concatenation of the two output vectors from both directions,ie. $h_t = \overrightarrow{h_t} || \overleftarrow{h_t}$.

After the bidirectional LSTM, the last hidden variable h_n or all hidden states average $\frac{1}{n}\sum_{t=1}^{n} h_t$ is adopted as the preliminary question representation r_q. We can also obtain the last hidden variable H_a for the answer sentence, where $h_a(i)$ means the last hidden output of the i-th word in the answer.

$$H_a = [h_a(1), h_a(2), ..., h_a(m)] \tag{10}$$

When modeling the candidate answer sentence with length m: $S = s_1, s_2, s_3, ..., s_m$ in multiple features based attention model, instead of using the last hidden state or average hidden states, we use multiple features based attentive hidden states that are weighted by r_q and r_z, where r_z is the merge of semantic feature vector r_s and the metadata feature vector r_m:

$$s_t \propto f_{attention}(r_q, r_z, h_a(t)) \tag{11}$$

$$\tilde{h_a}(t) = h_a(t) s_t \tag{12}$$

$$r_a = \sum_{t=1}^{m} \tilde{h_a}(t) \tag{13}$$

where $h_a(t)$ is hidden state of the answer at time t. The attention function $f_{attention}$ is computed as:

$$m(t) = \tanh(W_{hm} h_a(t) + W_{qm} r_q + W_{sm} r_z) \tag{14}$$

$$f_{attention}(r_q, r_z, h_a(t)) = \exp(w_{ms}^T m(t)) \tag{15}$$

W_{hm}, W_{qm} and W_{sm} are attentive weight matrices and w_{ms} is attentive weight vector. So we can expect that the candidate answer sentence representation r_a may be represented in a question and different kinds of features guided way: when its hidden state $h_a(t)$ is irrelevant to the question and the different kinds of features(determined by attention weight s_t), it will take less part in the final representation; but when this hidden state is relavent to the question and different kinds of features, it will contribute more in representing r_a.

Table 1: Experimental dataset on the SemEval-2016 Task 3

	train	dev	test
number of questions	2,669	244	327
number of answers	17,900	2,440	3,270

Table 2: Experimental dataset on the SemEval-2017 Task 3

	train	dev	test
number of questions	4,879	244	293
number of answers	36,198	2,440	2,930

The final question vector r_{qz} and the answer vector r_a which we obtain using multiple features based attention are both useful in the ranking task. We splice them by the row and get the high level representation of the sentence $\bar{s}$. We use it to classify the answer of the question by a softmax function

$$p = softmax(W_c\bar{s} + b_c) \tag{16}$$

$$L = -\sum_{s} \log p_{sj} \tag{17}$$

where j is the label of answer s, W_c is the weight matrices and b_c is the bias vector.

4 EXPERIMENTS EVALUATION AND ANALYSIS

In this section, we describe our experimental setting. We first introduce the dataset and the competitive methods, then present the results and analysis.

4.1 Datasets Setup

We experiment with the dataset from SemEval-2016 Task3 and SemEval-2017 Task3 on answer ranking for Community Question Answering [18, 19]. The datasets contain question-answer threads from the Qatar Living forum. Each answer in the dataset is annotated with one of the following labels, reflecting how well it answers the question: Good, PotentiallyUseful, and Bad. At SemEval-2016 and SemEval-2017 Task3, the latter two classes were merged into Bad at testing time. The statistics of the datasets are summarized in Table 1 and Table 2.

4.2 Compared with Competitive Methods

We experimentally compare our model with the other models at SemEval-2016 and SemEval-2017 Task 3. They are Kelp, ConvKN, SemanticZ in SemEval-2016 and KeLP, Beihang-MSRA and ECNU in SemEval-2017.

- KeLP: Their system learned semantic relations between questions and answers using kernels and previously-proposed features from [3]. The System is based on the KeLP machine learning platform [7].
- ConvKN: They combined convolutional tree kernels and convolutional neural networks, together with text similarity and thread-specific features [10].

Table 3: Experimental Results on the SemEval-2016 Task 3

Model	MAP	AvgRec	MRR
Random baseline [1]	52.80	66.52	58.71
Search engine [1]	59.53	72.60	67.83
KeLP [1]	79.19	88.82	86.42
ConvKN [1]	77.66	88.05	84.93
SemanticZ [1]	77.58	88.14	85.21
Attention-BiLSTMs	77.60	88.27	85.05
FAN(only semantic features)	78.10	88.30	85.65
FAN(only metadata features)	78.20	88.38	85.73
FAN(both features)	**79.38**	**88.96**	86.18

- SemanticZ: They used the semantic similarity based on word embeddings and topics [16].
- KeLP: They used tree kernels with relational links in Support Vector Machine. The questions are aligned with the answers (or with the other questions) by means of a special REL tag, directly annotated in the parse trees. It used the homonymous tree kernel software toolkit, KeLP [8].
- Beihang-MSRA: They developed a ranking system for capturing semantic relations between text pairs with little word overlap. In addition to traditional NLP features, they introduced several neural network based matching features which enable their system to measure text similarity beyond lexicons [32].
- ECNU: They explored a traditional machine learning method which used multiple types of features, e.g., Word Match Features, Topic Model-based Features, and Lexical Semantic Similarity Features. Additionally, they also built a Convolutional Neural Network (CNN) model to learn joint representation for question-comment (Q-C) pair. They combined two different methods to represent question-comment pair [29].

4.3 Experimental Settings

The questions and answers are tokenized and lemmatized using NLTK [7]. We removed the stop words and changed all the letters to lowercase. We used semantic word embeddings obtained from Word2Vec [2] models trained on different unannotated data sources including the QatarLiving and DohaNews [16]. The dimension of word embedding was 200. Tokens that did not appear in the pre-trained word embeddings were replaced with a special token, of which the embedding was initialized randomly. The number of nodes in the hidden layer of the bidirectional LSTM is 128. Therefore, the question vector r_{qz} and the answer vector r_a are 256 dimensions. We merge the features by stitching them, so we obtain the 512 dimensions feature as input. The activation we used in the bidirectional LSTM is Relu and the dropout rate is set as 0.5. To evaluate the model performance, we adopt the mean average precision(MAP), AvgRec and MRR which are the primary metrics used in QA [26, 27].

4.4 Results and Analysis

The experimental results on the SemEval-2016 and SemEval-2017 Task 3 are shown in Table 3 and Table 4.

Table 4: Experimental Results on the SemEval-2017 Task 3

Model	MAP	AvgRec	MRR
Random baseline [18]	62.30	70.56	68.74
Search engine [18]	72.61	79.32	82.37
KeLP [18]	88.43	93.79	92.82
Beihang-MSRA [18]	88.24	93.87	92.34
ECNU [18]	86.72	92.62	91.45
Attention-BiLSTMs	83.48	89.34	88.25
FAN(only semantic features)	85.10	90.56	89.85
FAN(only metadata features)	86.23	91.28	90.47
FAN(both features)	**88.72**	**93.92**	92.63

Table 5: Effectiveness of FAN Model

Question	Traditional attentive word(top 5)	FAN word(top 5)	Answer
is there any place i can find scented massage oils in qatar?	massage, oils, qatar, affordable, prices	naseem, alnadir, smartlink, chinese, prices	It's called Naseem Alnadir. Right next to the Smartlink shop. You'll find the chinese salesgirls at affordable prices there.

We can see that our MAP(with both features) respectively improvement of 19.85 and 16.11 compared with the results obtained by the search engine in SemEval-2016 and SemEval-2017. We also compare the FPN(only semantic features), the FPN(only metadata features) and the FPN(both features) models and find that both of the features are effective to solve the attention divergence problem, especially the metadata features. The model with both features obtain the best result. KeLP used various kinds of features and got the best results in SemEval-2016 and SemEval-2017 which demonstrates that feature engineering is effective in answer ranking task. ConvKN ranked second in SemEval-2016. Beihang-MSRA obtained the best AvgRec and ECNU ranked fourth in SemEval-2017. Our MAP separately improves 0.19 and 0.29 compared with the results obtained by the KeLP in SemEval-2016 and SemEval-2017. Our model also gets the best AvgRec above the other methods in SemEval-2016 and SemEval-2017. The result proves that our model can outperform the state-of-the-art systems.

We also compare our model with the traditional attention-biLSTMs model. The result shows that our MAP respectively improves 1.78 and 5.24 in SemEval-2016 and SemEval-2017. Our **FAN** model can focus attention back on the right regions of the target terms and achieve significant improvements over the other models.

4.5 Case Study

In order to better understand our proposed model, we provided an example to show why our FAN model is more effective for the answer ranking task. As shown in Table 5, for the question "*is there any place i can find scented massage oils in qatar?*", it is required to find a place, and we observe that in the traditional attention model, the top 2 attentive words associate with the question are not location names, such as *massage* and *oils*. However, in the FAN model, we observe that top 2 attentive words associate with the question are location names, such as *naseem* and *alnadir*. In particular, these two words contain the information that associate with the key word "*Naseem Alnadir*" in the good answer, which solves the issues of attention divergence for answer ranking task. Therefore, our FAN model, which enhances the question representations to rank good answers ahead of bad ones.

5 CONCLUSION

In this work, we put forward the *attention divergence* to explain the poor performance of existing attention based methods of answer ranking on complicated long question-answer pairs, and propose a novel FAN to solve this problem. Different from the existing methods, firstly, we obtain the question representation with Bidirectional LSTMs, next we extract semantic features, metadata features from questions and their answers to form the focusing features, then we use the focusing attentive network to obtain answer representation. At last, we feed the question and answer representation to the softmax classifier for classification. Experiments on SemEval-2016 and SemEval-2017 have shown the effectiveness of our proposed model, especially in terms of the Mean Average Precision and precision.

Because semantic and metadata features are the important features in the sentence and the complicated long sentences can be seen in many tasks such as text classification and reading comprehension. In future work, we plan to extend the proposed idea to text classification and other related tasks.

REFERENCES

[1] PreslavNakov LluısMarquez AlessandroMoschitti, WalidMagdy HamdyMubarak AbedAlhakimFreihat, James Glass, and Bilal Randeree. 2016. SemEval-2016 Task 3: Community Question Answering. *Proceedings of SemEval* (2016), 525–545.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[3] Alberto Barrón-Cedeno, Simone Filice, Giovanni Da San Martino, Shafiq R Joty, Lluís Màrquez, Preslav Nakov, and Alessandro Moschitti. 2015. Thread-Level Information for Comment Classification in Community Question Answering.. In *ACL (2)*. Citeseer, 687–693.

[4] Marie-Catherine De Marneffe and Christopher D Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*. Association for Computational Linguistics, 1–8.

[5] Wenzheng Feng, Yu Wu, Wei Wu, Zhoujun Li, and Ming Zhou. [n. d.]. Beihang-MSRA at SemEval-2017 Task 3: A Ranking System with Neural Matching Features for Community Question Answering. ([n. d.]).

[6] Alejandro Figueroa. 2017. Automatically generating effective search queries directly from community question-answering questions for finding related questions. *Expert Systems with Applications* 77 (2017), 11–19.

[7] Simone Filice, Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2016. KeLP at SemEval-2016 Task 3: Learning semantic relations between questions and answers. *Proceedings of SemEval* 16 (2016), 1116–1123.

[8] Simone Filice, Giovanni Da San Martino, and Alessandro Moschitti. 2017. KeLP at SemEval-2017 task 3: Learning pairwise patterns in community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation. Vancouver, Canada, SemEval*, Vol. 17. 327–334.

[9] Amir Globerson, Gal Chechik, Fernando Pereira, and Naftali Tishby. 2007. Euclidean embedding of co-occurrence data. *Journal of Machine Learning Research* 8, Oct (2007), 2265–2295.

[10] Shafiq Joty, Alessandro Moschitti, Fahad A Al Obaidli, Salvatore Romeo, Kateryna Tymoshenko, and Antonio Uva. 2016. ConvKN at SemEval-2016 Task 3: Answer and question selection for question answering on Arabic and English fora. *Proceedings of SemEval* (2016), 896–903.

[11] Suyoun Kim, Takaaki Hori, and Shinji Watanabe. 2017. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 4835–4839.

[12] Yuta Koreeda, Takuya Hashito, Yoshiki Niwa, Misa Sato, Toshihiko Yanase, Kenzo Kurotsuchi, and Kohsuke Yanai. 2017. bunji at SemEval-2017 task 3: Combination of neural similarity features and comment plausibility features. In *Proceedings of the International Workshop on Semantic Evaluation. Vancouver, Canada, SemEval*, Vol. 17. 353–359.

[13] Rémi Lebret and Ronan Collobert. 2013. Word emdeddings through hellinger PCA. *arXiv preprint arXiv:1312.5542* (2013).

[14] Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*. 2177–2185.

[15] Yitan Li, Linli Xu, Fei Tian, Liang Jiang, Xiaowei Zhong, and Enhong Chen. 2015. Word Embedding Revisited: A New Representation Learning and Explicit Matrix Factorization Perspective.. In *IJCAI*. 3650–3656.

[16] Todor Mihaylov and Preslav Nakov. 2016. SemanticZ at SemEval-2016 Task 3: Ranking relevant answers in community question answering using semantic similarity based on fine-tuned word embeddings. *Proceedings of SemEval* (2016), 879–886.

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[18] Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 Task 3: Community Question Answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval '17)*. Association for Computational Linguistics, Vancouver, Canada.

[19] Preslav Nakov, Lluís Màrquez, Alessandro Moschitti, Walid Magdy, Hamdy Mubarak, Abed Alhakim Freihat, Jim Glass, and Bilal Randeree. 2016. SemEval-2016 Task 3: Community Question Answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval '16)*. Association for Computational Linguistics, San Diego, California.

[20] Xipeng Qiu and Xuanjing Huang. 2015. Convolutional Neural Tensor Network Architecture for Community-Based Question Answering.. In *IJCAI*. 1305–1311.

[21] Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer.

[22] Aliaksei Severyn and Alessandro Moschitti. 2013. Automatic feature engineering for answer selection and extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 458–467.

[23] Gen Takahashi, Takeshi Yamada, Shoji Makino, and Nobutaka Ono. 2016. Acoustic scene classification using deep neural network and frame-concatenated acoustic feature. *Detection and Classification of Acoustic Scenes and Events* (2016).

[24] Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108* (2015).

[25] Quan Hung Tran, Vu Tran, Tu Vu, Minh Le Nguyen, and Son Bao Pham. 2015. JAIST: Combining multiple features for answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval*, Vol. 15. 215–219.

[26] Bingning Wang, Kang Liu, and Jun Zhao. 2016. Inner Attention based Recurrent Neural Networks for Answer Selection.. In *ACL (1)*.

[27] Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 707–712.

[28] Mengqiu Wang and Christopher D Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 1164–1172.

[29] Guoshun Wu. [n. d.]. Yixuan Sheng adn Man Lan, and Yuanbin Wu. 2017a. ECNU at SemEval-2017 task 3: Using traditional and deep learning methods to address community question answering task. In *Proceedings of the Workshop on Semantic Evaluation. Vancouver, Canada, SemEval*, Vol. 17. 365–369.

[30] Guoshun Wu and Man Lan. 2016. ECNU at SemEval-2016 Task 3: Exploring traditional method and deep learning method for question retrieval and answer ranking in community question answering. *Proceedings of SemEval* (2016), 872–878.

[31] Guoshun Wu, Yixuan Sheng, Man Lan, and Yuanbin Wu. 2017. ECNU at SemEval-2017 Task 3: Using Traditional and Deep Learning Methods to Address Community Question Answering Task. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 365–369.

[32] Yu Wu, WenZheng Feng, Wei Wu, Ming Zhou, and Zhoujun Li. 2017. Beihang-MSRA at SemEval-2017 task 3: A ranking system with neural matching features for Community Question Answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation. Vancouver, Canada, SemEval*, Vol. 17. 281–287.

[33] Shuo Yang, Lei Zou, Zhongyuan Wang, Jun Yan, and Ji-Rong Wen. 2017. Efficiently Answering Technical Questionsïż¡ï€¡A Knowledge Graph Approach. *Thirty-First AAAI Conference on Artificial Intelligence* (2017).

[34] Scott Wen-tau Yih, Ming-Wei Chang, Chris Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. (2013).

[35] Naiyun Zhou, Andrey Fedorov, Fiona Fennessy, Ron Kikinis, and Yi Gao. 2017. Large scale digital prostate pathology image analysis combining feature extraction and deep neural network. *arXiv preprint arXiv:1705.02678* (2017).

DLocRL: A Deep Learning Pipeline for Fine-Grained Location Recognition and Linking in Tweets

Canwen Xu
Wuhan University
School of Computer Science
Hubei, China
xucanwen@whu.edu.cn

Jing Li
Inception Institute of Artificial Intelligence
United Arab Emirates
jing.li@inceptioniai.org

Xiangyang Luo*
State Key Lab of Mathematical Engineering and Advanced Computing
Henan, China
xiangyangluo@126.com

Jiaxin Pei
Wuhan University
School of Computer Science
Hubei, China
pedropei@whu.edu.cn

Chenliang Li
Wuhan University
School of Cyber Science and Engineering
Hubei, China
cllee@whu.edu.cn

Donghong Ji
Wuhan University
School of Cyber Science and Engineering
Hubei, China
dhji@whu.edu.cn

ABSTRACT

In recent years, with the prevalence of social media and smart devices, people causally reveal their locations such as shops, hotels, and restaurants in their tweets. Recognizing and linking such fine-grained location mentions to well-defined location profiles are beneficial for retrieval and recommendation systems. In this paper, we propose **DLocRL**, a new deep learning pipeline for fine-grained location recognition and linking in tweets, and verify its effectiveness on a real-world Twitter dataset.

CCS CONCEPTS

• **Information systems → Information extraction**.

KEYWORDS

POI recognition and linking; social media content analysis; named entity recognition; entity linking

ACM Reference Format:
Canwen Xu, Jing Li, Xiangyang Luo, Jiaxin Pei, Chenliang Li, and Donghong Ji. 2019. DLocRL: A Deep Learning Pipeline for Fine-Grained Location Recognition and Linking in Tweets. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313491

1 INTRODUCTION

Twitter is a place where users can share their daily life activities by posting tweets (*i.e.,* short messages, up to 140 characters each). In many tweets, locations are implicitly or causally revealed by users at fine-grained granularity [14, 16, 17], for example, a restaurant, a shopping mall, a park or a landmark building. Here, a fine-grained location is equivalent to a point-of-interest (POI), which is a focused geographical entity [16, 26]. In this paper, we target on *recognizing* mentions of POIs and *linking* these mentions to well-defined location profiles.

The two tasks are important for several reasons. First, recognizing mentions of POIs is beneficial for information retrieval [2]. An example is that query understanding can be enhanced by exploiting POI information in location-based information retrieval systems [7, 28]. In addition, recognized POIs can be integrated into knowledge bases and support many business intelligence applications such as POI recommendation and location-aware advertising [16, 20, 22]. Second, the linked location profile can serve as side information for the tweet, and vice versa. For example, Twitter sentiment analysis can be conducted in a more precise manner by incorporating the location profile content [10].

However, recognizing POI mentions and linking these mentions to well-defined location profiles are both challenging. Due to the informal writing of tweets, POIs are usually mentioned by incomplete name, nickname, acronym or misspellings. For example, the mention *vivo* may refer to *VivoCity* which is a comprehensive shopping mall in Singapore, or a smartphone brand. Even if we have successfully recognized a POI mention, it remains challenging to link the mention to a specified location profile (*i.e.,* a specific shopping mall with address and geo-coordinates in this case).

On the other hand, existing solutions [10, 14, 16] on these two tasks require a large set of features manually designed for each task and domain, which demands task and domain expertise. For example, Li *et al.* [16] carefully designed lexical, grammatical, geographical and BILOU schema features (totally 11 features) to extract fine-grained locations from tweets. Ji *et al.* [14] proposed a joint model to recognize and link fine-grained locations from tweets with 24 hand-crafted features. Recently, Han *et al.* [10] proposed a probabilistic model with seven types of hand-crafted features to link fine-grained location in user comments. It is now generally admitted that distributed representations could better capture lexical semantics [8]. We would envision for a system that is based on

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313491

distributed representations, and that can learn informative features for location recognition and linking by itself without human effort.

In this paper, we propose DLocRL, a new **D**eep pipeline for fine-grained **Loc**ation **R**ecognition and **L**inking in tweets. DLocRL is designed to adopt effective representation learning, semantic composition, and attention and gate mechanisms to exploit the multiple semantic context features for location recognition and linking. DLocRL consists of two core modules: *recognition* module and *linking* module.

The *recognition* module aims to extract a text segment referring to a fine-grained location (*i.e.,* POI) from a given tweet. In DLocRL, we formulate the recognition task as a sequence labeling problem (*i.e.,* assigning tags to tokens in the given sequence). We adopt bi-directional long short-term memory with conditional random fields output layer (BiLSTM-CRF) to train a POI recognizer.

The *linking* module aims to link each recognized POI to a corresponding location profile. Before linking POIs to location profiles, an extensive collection of location profiles (CLP) is constructed. Given an input pair ⟨POI, Profile⟩, the *linking* module is trained to judge whether the location profile corresponds to the POI. More specifically, DLocRL utilizes two pairs of parallel LSTMs to encode the left context and right context for a POI, respectively. The location profiles are represented by their constituents with TF-IDF weights and one-hot schema. The Manhattan distance is used to measure the geographical distances between users and profiles. Finally, the representation of ⟨POI, Profile⟩ comes from three sources: tweet-level contextual information, location profile representation and geographical distance. The representation is then fed into a fully-connected layer for final linking prediction. Moreover, to take advantage of the geographical coherence information among mentioned POIs in the same tweet, we develop Geographical Pair Linking (Geo-PL), a post-processing strategy to further enhance the linking accuracy.

In summary, we make the following contributions: (1) We propose **DLocRL**, a new deep learning pipeline for fine-grained Location Recognition and Linking in tweets. To the best of our knowledge, our work is the first attempt to address location recognition and linking in the paradigm of deep learning. (2) We develop the Geographical Pair Linking (Geo-PL) approach, a post-processing strategy to further improve linking performance. (3) We conduct extensive experiments on a real-world Twitter dataset. The experimental results show the effectiveness of DLocRL on fine-grained location recognition and linking. We also conduct ablation studies to validate the effectiveness of each design choice.

2 RELATED WORK

2.1 Location Mention Recognition and Disambiguation

Recognition. Facing noisy and short tweets, traditional NER methods suffer from their unreliable linguistic features [36]. Prior solutions exploit comprehensive linguistic features like Part-of-Speech tags, capitalization [27], Brown clustering [3] to improve recognition performance. For example, Li *et al.* [16] carefully designed lexical, grammatical, geographical and BILOU schema features (totally 11 features) to extract fine-grained locations from tweets. Ji *et al.* [14] proposed a joint model to recognize and link fine-grained locations from tweets with 24 hand-crafted features. Zhang *et al.* [34] designed four types of features and trained a classifier to select the best match among gazetteer candidates. Generally, location gazetteers are widely used in location mention recognition [16, 21, 34]. For noisy text, Malmasi *et al.* [21] proposed an approach based on Noun Phrase extraction and n-gram based matching instead of the traditional methods using NER or CRF. Different from these existing works, our approach DLocRL does not require hand-crafted features.

Disambiguation. For entity disambiguation, many studies attempt to exploit the coherence among mentioned entities. Pair-wise [15] and global collective methods [11, 12, 32] have been applied to explore the coherence. Zhang *et al.* [34] and Ji *et al.* [14] observed that the geographical coherence is effective in location disambiguation task. Also, instead of tweet-level coherence, Li *et al.* [18] utilized user-level coherence to facilitate entity disambiguation. Besides, Shen *et al.* [30] introduced user interest modeling to collective disambiguation.

To form an entirely end-to-end model for joint recognition and linking, Guo *et al.* [9] and Ji *et al.* [14] adopted structural SVM and perceptron, respectively. Furthermore, the beam search algorithm [35] is also used in [14] to search the best combination of the mention recognition and linking. In this work, we devise DLocRL, a new pipeline which can be upgraded and optimized much easier than previous joint models and still achieves better performance.

2.2 Neural Networks for NER and EL

Named Entity Recognition (NER). The use of neural models for NER was pioneered by Collobert *et al.* [5], where an architecture based on temporal convolutional neural networks (CNNs) over word sequence was proposed. Chiu *et al.* [4] utilized CNN to detect character-level features and LSTM to capture the word-level context in a sentence. Yang *et al.* [33] proposed a gated recurrent unit (GRU) network to learn useful morphological representation from the character sequence of a word. Recently, Peters *et al.* [24] proposed deep contextualized word representation, which can model syntax, semantics, and polysemy. Akbik *et al.* [1] proposed contextual string embeddings, to leverage the internal states of a trained character language model to produce a novel type of word embedding.

Entity Linking (EL). Neural networks were firstly used for entity linking by Sun *et al.* [31]. They proposed a model which takes consideration of the semantic representations of mention, context, and entity. Recently, Phan *et al.* [25] proposed a neural network for entity linking with LSTM and attention mechanism. They also proposed *Pair Linking* to enhance collective linking by measuring the cosine similarity of the text embeddings between two mentioned entities.

3 MODEL DESCRIPTION

Figure 1 shows the workflow of DLocRL, which consists of recognition module and location linking module.

3.1 Location Recognition

Figure 2 shows the architecture of the location recognition module. The representation of a word consists of its pre-trained word embedding, BILOU pre-label and character-level representation.

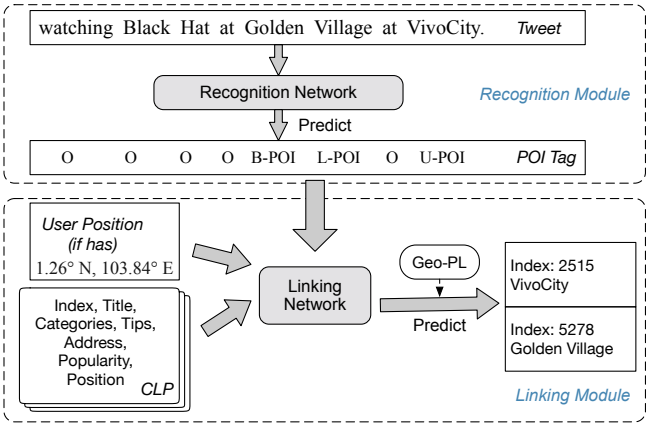


Figure 1: A running example to illustrate the workflow of DLocRL.

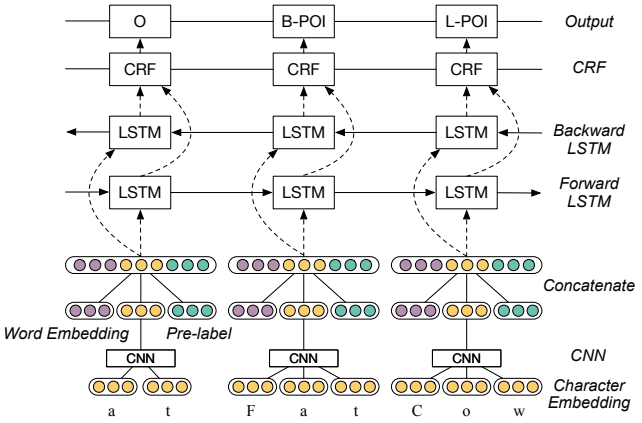


Figure 2: The architecture of location recognition module.

Finally, BiLSTM-CRF is utilized to infer tag sequence based on the representations.

Pre-trained Word Embedding. To deal with the problem of informal spellings and casual expressions, we should capture the information from tweets as accurate as possible. We use the GloVe[23] embeddings pre-trained on a large-scale Twitter corpus of two billion tweets. The vocabulary of pre-trained embeddings covers most common misspellings and aliases of most common words.

Pre-label. POI inventory, containing partial and familiar names from Foursquare, is proposed in [16] to pre-label candidate location mentions in tweets. It has been proved that pre-label is an essential resource to improve recognition performance. We use a CRF toolkit[1] to automatically assign the pre-labels with BILOU scheme[27].

Character-level Representation. Previous studies [4, 13] have shown that character-level information (*e.g.,* prefix and suffix of a word) is an effective resource for NER task. CNN and BiLSTM are commonly used in previous works to extract character-level representation. In our model, we use CNN because of its lower computational cost [4, 19].

BiLSTM-CRF. LSTM is a variant of recurrent neural network (RNN) and is designed to deal with vanishing gradients problem. BiLSTM uses two LSTMs to represent each token of the sequence based on both the past and the future context of the token. As shown in Figure 2, one LSTM processes the sequence from left to right, the other one from right to left. For a word, we concatenate its pre-trained word embedding, pre-label and character-level representation as its final representation, which is then fed into a BiLSTM layer. Then, the output sequence is fed into a CRF layer to infer the tag sequence. BiLSTM-CRF is a state-of-the-art approach to named entity recognition [6, 13].

3.2 Location Linking

Figure 3 shows the architecture of the location linking module. We use two pairs of parallel LSTMs (four in total) to encode the left-side and right-side contexts of a mention, respectively. Note that the input of the two parallel LSTMs for each side is re-weighted by two different *attentions* from the location profile. The output sequences of LSTMs are "fused" by a *fusion gate* and then fed into a max-pooling layer over all time steps as the final representation for each side context. Next, we concatenate the left-side context representation, right-side context representation, the representation of location profile together with the *Manhattan distance* between user coordinates (*i.e.,* user position attached to the tweet) and coordinates from location profile. Finally, a multilayer perceptron (MLP) with a sigmoid activated output layer is used to output a scalar ranged between zero and one as the matching score.

Location Profile Mapping Dictionary. Following [14], we use a location profile mapping dictionary to recall all possible candidates for a location mention. The *key* of the dictionary is a possible POI mention, and the *value* is the list of candidate profile indexes for the POI mention. If the mention is not in the mapping dictionary, we predict it as *un-linkable*. The dictionary is constructed with Foursquare check-in tweets.

Mention's Context. Following [25], we use LSTM networks to capture two-side contextual information of the POI mention. The difference is that we use all words in a tweet instead of specific window size. The left-side context starts from the leftmost word in a tweet and ends at the rightmost word inside the mention. Conversely, the right-side context starts from the tweet's rightmost word and ends at the leftmost word of the mention. Since the tweet does not have a long context, we use all words to understand the tweet as a whole. Note that the input of LSTM layers is re-weighted with a multi-attention mechanism, which is to be detailed shortly.

Behavioral and Semantic Information. Users often reveal their locations and describe what they are doing in tweets. For example, a user posts "Great lobster @ Red Robin!", which contains information about behavior (having lobster). Here, we name such information as ***behavioral information***. Obviously, since few restaurants serve lobster, it could be beneficial for POI disambiguation. According to our observation, tweets share the same behavioral information with the tips (*i.e.,* user comments) in the Foursquare location profile. On the other hand, some words in a tweet may be directly semantically relevant to the category or address of a location. For instance, in the tweet "Best wine at Elle's in the center", *wine* is closer to *bar* in word embedding space. Such information

[1] CRF++: https://github.com/taku910/crfpp

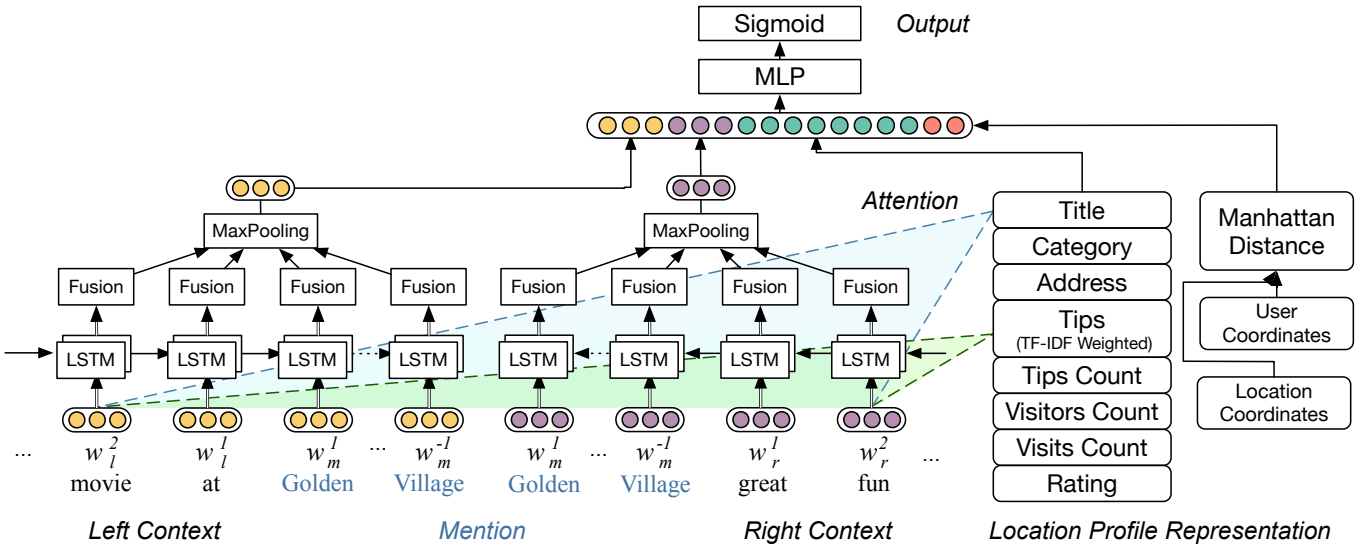


Figure 3: The architecture of location linking network. The input is a pair ⟨tweet, profile⟩, and the output is the matching score.

is beneficial for location disambiguation between *Elle's Bar* and *Elle's Salon*. Similarly, in the same example, *center* is a synonym for *Central Region*. This can be helpful for picking a location profile titled "Elle's Bar - Central Region" out of other branches of Elle's Bar. We name such information as ***semantic information***.

Multi-attention Mechanism. To comprehensively exploit *semantic information* and *behavioral information*, we develop a multi-attention mechanism to assign weights for the input sequence. As shown in Figure 3, we use the *title* representation and *tips* representation as two attention vectors. Specifically, given an input word embedding w_i in a tweet and an attention vector p (*i.e.,* representation of title or tips from location profile), the re-weighted input word embedding w_i' is defined by:

$$z_i = \tanh\left(V_p p + V_w w_i\right) \tag{1}$$

$$s_i = \text{softmax}(v_k z_i + v_b) \tag{2}$$

$$w_i' = w_i s_i \tag{3}$$

where V_*, v_k, and v_b are attention parameters which can be learned during training. The TF-IDF weighted tips representation (to be detailed shortly) contains the most characteristic behavioral information about a location. Thus, the attention from tips can highlight the parts which are most relevant with the tips. Similarly, the title contains rich semantic information (including *location name*, *branch name* and *business status*) which can highlight the parts relevant to the location.

Fusion Gate. To collect important information from the output of two parallel LSTMs, we borrow a simple but practical *fusion gate* from [29]. Given the output of two parallel LSTMs with different attentions (*i.e.,* x_i^1 and x_i^2), the output of fusion gate u_i is formally defined by:

$$f_i = \text{sigmoid}(W_1 x_i^1 + W_2 x_i^2 + b) \tag{4}$$

$$u_i = f_i \odot x_i^1 + (1 - f_i) \odot x_i^2 \tag{5}$$

where W_1, W_2, and b are learnable parameters. The fusion gate enables the model to learn how to weight the two input sequences, which solves the problem of the imbalanced importance of behavioral information and semantic information.

Location Profile. The properties (exclude "category") of a location profile fall into two classes: *word-based* and *numeric-based*. For *word-based* properties (*i.e.,* title, address, and tips), we represent each property by averaging all word embeddings inside it. In particular, the word embeddings of "tips" are weighted by TF-IDF schema. For *numeric-based* properties (*i.e.,* tips count, visitors count and visits count), we normalize them through the whole CLP. For *category* property, we encode it as a one-hot vector. Finally, we concatenate representations of all properties together as the location profile representation.

Manhattan Distance. To measure the distance between user coordinates and location coordinates in a city, we use the Manhattan distance (*i.e.,* taxicab distance) defined by:

$$d(p, q) = |p_x - q_x| + |p_y - q_y| \tag{6}$$

where p and q are two pairs of coordinates. Manhattan distance is better than Euclidean distance since the road network in a city or suburb is usually orthogonal. Note that not every tweet has an attached user position. For these situations, we fix the distance value to a pre-defined constant. Manhattan distance is also used in our post-processing, geographical pair linking, which will be discussed in Section 4.

Final Prediction. For all candidate location profiles in our mapping dictionary for a mention, we predict the matching score between the mention and each candidate. Formally, the final predicted location profile p_i for mention m_i is computed by:

$$p_i = \underset{p_j \in c(m_i)}{\arg\max}\, \phi(m_i, p_j) \tag{7}$$

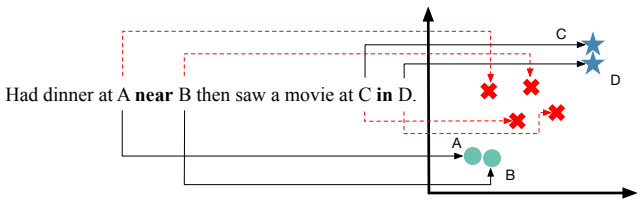


Figure 4: A wrong case of taking into account the coherence among all mentions. A, B, C and D are true POIs mentioned by a user. The incorrect predictions are marked with red dashed arrows and crosses.

where $c(m_i)$ is the candidate set for m_i and $\phi(m_i, p_j)$ is the matching score between m_i and p_j.

4 GEOGRAPHICAL PAIR LINKING

In this section, we introduce Geographical Pair Linking (Geo-PL), a post-processing strategy to enhance our linking performance.

For location linking, one of the most valuable information is the geographical coherence among mentioned locations in the same tweet. First, instead of directly calling the branch name of a chain restaurant, users are more likely to mention them with another location (usually a landmark). Second, people may describe a route by listing the locations along the way one by one, which also reveals the coherence among the mentioned locations.

For geographical coherence, prior studies often measure the distances among all mentions in a tweet. According to our observation, geographical coherence between two locations is strong enough for location disambiguation. Moreover, as illustrated in Figure 4, when a user mentions four locations with a conventional way like "Had dinner at A near B then saw a movie at C in D," the geographical coherence measurement among all mentions brings obvious error. It obtains the collective minimum but disregards the strong connection between two mentions.

To exploit pair-wise geographical coherence, we developed **Geographical Pair Linking (Geo-PL)** algorithm based on the *Pair Linking* algorithm [25]. The original algorithm measures the cosine similarity between text embeddings, while we use Manhattan distance discussed in Section 3.2 to measure the geographical coherence. Geo-PL iteratively resolves all pairs of mentions, starting from the most *confident* pair. The confidence score of a pair of links $m_i \mapsto p_i$ and $m_j \mapsto p_j$ is defined by:

$$conf(i,j) = (1-\beta)\frac{[\phi(m_i,p_i)+\phi(m_j,p_j)]}{2} + \beta\frac{1}{d(p_i,p_j)} \quad (8)$$

where β is a given coefficient representing the preference between the matching scores and the geographical coherence; d is the Manhattan distance defined by Equation 6. In the case of $p_i = p_j$, we temporarily set β to 0 when calculating. The procedure of Geo-PL (the same as Pair Linking) is detailed in Algorithm 1. Note that if there is only one POI mention in a tweet, we simply predict its location profile by Equation 7.

Algorithm 1: Geo-PL Algorithm

Input: N mentions($m_1, \ldots, m_N$). Mention m_i has candidate profiles $\{e|e_i \in C(m_i)\}$
Output: $\Gamma = (p_1, \ldots, p_N)$

$p_i \leftarrow null, \forall p_i \in \Gamma$;
foreach $pair(m_i, m_j) \wedge m_i \neq m_j$ **do**
  $Q_{m_i,m_j} \leftarrow top_pair(m_i, C(m_i), m_j, C(m_j))$;
  $Q.add(Q_{m_i,m_j})$;
end
while $\exists p_i \in \Gamma, p_i = null$ **do**
  $(m_i, e_i, m_j, e_j) \leftarrow most_confident_pair(Q)$;
  $p_i \leftarrow e_i$;
  $p_j \leftarrow e_j$;
  for $k := 1 \rightarrow N \wedge p_k = null$ **do**
    $Q_{m_k,m_i} \leftarrow top_pair(m_k, C(m_k), m_i, \{p_i\})$;
    $Q_{m_k,m_j} \leftarrow top_pair(m_k, C(m_k), m_j, \{p_j\})$;
  end
end

5 EXPERIMENT

5.1 Data Preparation

We use a Singaporean national Twitter dataset released by Ji *et al.* [14]. The dataset includes a CLP and a set of labeled tweets.

326,853 Foursquare *check-in tweets*, containing both formal auto-generated POI name and informal user mentions, are collected to build the CLP (including 22,414 valid location profiles) and the POI inventory (including 27,386 entries). Location profile mapping dictionary (see Section 3.2) is also constructed with check-in tweets. Informal mentions in check-in tweets and the indexes of crawled profiles are used as keys and values, respectively. The final location profile mapping dictionary has 24,750 keys and 63,091 *<key, value>* mappings.

The dataset consists of 3,611 labeled tweets and 1,542 POI locations, 543 of which can be linked to a profile in the CLP. 10% of tweets have an attached user position. These tweets are labeled by human annotators. All possible POI mentions are labeled with one of *NPOI* (*i.e.,* not a POI mention), *index* of linked location profile, or *NIL* (*i.e.,* cannot be linked to a profile).

We split the 3,611 labeled tweets randomly into three subsets: 2,500 tweets for training, 211 tweets for validation, and 900 tweets for testing. Same POI inventory and location profile mapping dictionary used in [14] are employed through all models in our experiments.

5.2 Parameter Setting and Evaluation Metrics

We set β, the coefficient of Geo-PL preference to 0.8 based on fine-tuning on the validation set. When concatenating the scalars to be the input of MLP, we duplicate them to the same number of dimensions as the word-based properties (*i.e.,* 200 dimensions in our experiments). This setting prevents the scalars from being "ignored" during training.

We conduct experiments on location recognition subtask, location linking subtask and the whole task, respectively. We adopt three metrics, Precision (Pr), Recall (Re) and F_1 which are widely

Table 1: Performance on Location Recognition and Location Linking.

Methods	Location Recognition			Location Linking			Recognition + Linking		
	Pr	*Re*	F_1	*Pr*	*Re*	F_1	*Pr*	*Re*	F_1
Li *et al.* [16]	**0.8962**	0.7661	0.8261	-	-	-	-	-	-
Shen *et al.* [30]	-	-	-	0.6723	0.6349	0.6531	-	-	-
Pipelined [16] [30]	-	-	-	-	-	-	0.6339	0.5635	0.5966
JoRL_L [14]	0.8926	0.7823	**0.8338**	-	-	-	0.8152	0.5952	0.6881
DLocRL	0.8575	**0.8091**	0.8326	**0.8235**	**0.7778**	**0.8000**	**0.8350**	**0.6825**	**0.7511**

used to evaluate NER and EL tasks. In particular, we use the ground-truth mentions instead of the predicted output of the recognition module, to get the metric scores for location linking.

5.3 Overall Comparison

We compare our model with state-of-the-art solutions. More specifically, we compare our model with [16] and [14] for recognition, and [30] for linking, respectively. Note that the performance of [14] on linking subtask is not provided since it is a joint model. For the whole task, we choose JoRL_L as the baseline since it is the state-of-the-art joint model in supervised learning manner. All models are fine-tuned on the validation set.

The result is shown in Table 1. We observe that DLocRL achieves the highest *recall* on location recognition. DLocRL beats Li *et al.* [16] and JoRL_L [14] with relative recall improvements of 5.61% and 3.43%, respectively. We contribute this to the fact that word/character embeddings enable our model to be more "tolerant" than prior works which depend on hand-crafted features. Although the precision score of DLocRL is not good, it does not have an impact on the whole task, because the location profile mapping dictionary can filter out most false positive (FP) prediction in the linking module.

On location linking subtask, our method outperforms the work [30] by 22.49% on precision, 22.51% on recall and 22.49% on F_1.

On the whole task, our model prominently outperforms the state-of-the-art joint solution (JoRL_L) on all three metrics (2.43% on precision, 14.67% on recall, 9.16% on F_1). Also, our model dramatically outperforms the prior state-of-the-art *pipeline* by 31.72% on precision, 21.12% on recall and 25.90% on F_1.

5.4 Effect of Multi-attention for Linking

We conduct experiments to verify the effectiveness of the multi-attention mechanism. Table 2 shows the linking performance with single attention from tips/title attentions and multi-attention. Note that the single attention approaches can only slightly improve the performance because they have "prejudice" which considers either behavioral information or semantic information. By "fusing" this two information, DLocRL eliminates the prejudice so it can effectively filter out noisy information. With the multi-attention mechanism, DLocRL improves the performance of linking module by 4.25% on precision, 4.26% on recall, and 4.26% on F_1, compared with the baseline.

5.5 Effect of Geo-PL for Linking

As discussed in Section 4, we introduce a novel post-processing strategy, named Geo-PL. Here, we conduct experiments to show the impact of Geo-PL for location linking. Table 3 shows the experimental results. Compared with no Geo-PL component, DLocRL (with Geo-PL) significantly improves precision by 13.46%, recall by 8.89% and F_1 by 11.11%.

Table 2: Effect of Multi-attention Mechanism for Linking.

Attention	Location Linking		
	Pr	*Re*	F_1
Baseline (*without attention*)	0.7899	0.7460	0.7673
Single Tips Attention	0.7983	0.7540	0.7755
Single Title Attention	0.7899	0.7460	0.7673
Tips + Title (*multi-attention*)	0.8235	0.7778	0.8000

Table 3: Effect of Geo-PL for Linking.

Strategy	Location Linking		
	Pr	*Re*	F_1
Without Geo-PL	0.7258	0.7143	0.7200
With Geo-PL	0.8235	0.7778	0.8000

6 CONCLUSIONS

In this paper, we introduce DLocRL, the first deep neural network based pipeline to recognize fine-grained location mentions in tweets and link the recognized locations to location profiles. Moreover, we develop a novel post-processing strategy which can further improve location linking performance. Through extensive experiments, we demonstrate the effectiveness of DLocRL against state-of-the-art solutions on a real-world Twitter dataset. The ablation experiments show the effectiveness of multi-attention mechanism and Geo-PL strategy on location linking.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their careful reading and their many insightful comments and suggestions. This research was supported by National Natural Science Foundation of China (No.U1636219, No.U1804263, No.61872278, No.61502344), Natural Scientific Research Program of Hubei Province (No.2017CFB-502, No.2017CFA007), the National Key R&D Program of China (No.2016QY01W0105, No.2016YFB0801303), and Plan for Scientific Innovation Talent of Henan Province (No.2018JR0018). Xiangyang Luo is the corresponding author.

REFERENCES

[1] Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual String Embeddings for Sequence Labeling. In *Proc. COLING*. 1638–1649.

[2] Krisztian Balog. 2018. Entity-Oriented Search. (2018).

[3] Peter F. Brown, Vincent J. Della Pietra, Peter V. de Souza, Jennifer C. Lai, and Robert L. Mercer. 1992. Class-Based n-gram Models of Natural Language. *Computational Linguistics* 18, 4 (1992), 467–479.

[4] Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. In *TACL*, Vol. 4. 357–370.

[5] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[6] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-Based Dependency Parsing with Stack Long Short-Term Memory. In *Proc ACL*. 334–343.

[7] Fredrik Espinoza, Per Persson, Anna Sandin, Hanna Nyström, Elenor Cacciatore, and Markus Bylund. 2001. Geonotes: Social and navigational aspects of location-based information systems. In *Proc. UBICOMP*. 2–17.

[8] Yoav Goldberg. 2017. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies* 10, 1 (2017), 1–309.

[9] Stephen Guo, Ming-Wei Chang, and Emre Kiciman. 2013. To Link or Not to Link? A Study on End-to-End Tweet Entity Linking. In *Proc. NAACL-HLT*. 1020–1030.

[10] Jialong Han, Aixin Sun, Gao Cong, Wayne Xin Zhao, Zongcheng Ji, and Minh C Phan. 2018. Linking Fine-Grained Locations in User Comments. *IEEE Transactions on Knowledge and Data Engineering* 30, 1 (2018), 59–72.

[11] Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proc. SIGIR*. 765–774.

[12] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust Disambiguation of Named Entities in Text. In *Proc. EMNLP*. 782–792.

[13] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).

[14] Zongcheng Ji, Aixin Sun, Gao Cong, and Jialong Han. 2016. Joint Recognition and Linking of Fine-Grained Locations from Tweets. In *Proc. WWW*. 1271–1281.

[15] Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of Wikipedia entities in web text. In *Proc. SIGKDD*. 457–466.

[16] Chenliang Li and Aixin Sun. 2014. Fine-grained location extraction from tweets with temporal awareness. In *Proc. SIGIR*. 43–52.

[17] Daifeng Li, Zhipeng Luo, Ying Ding, Jie Tang, Gordon Guo-Zheng Sun, Xiaowen Dai, John Du, Jingwei Zhang, and Shoubin Kong. 2017. User-level microblogging recommendation incorporating social influence. *Journal of the Association for Information Science and Technology* 68, 3 (2017), 553–568.

[18] Guoliang Li, Jun Hu, Jianhua Feng, and Kian-Lee Tan. 2014. Effective location identification from microblogs. In *Proc. ICDE*. 880–891.

[19] Jing Li, Aixin Sun, and Shafiq Joty. 2018. SegBot: A Generic Neural Text Segmentation Model with Pointer Network. In *Proc. IJCAI*. 4166–4172.

[20] John Lingad, Sarvnaz Karimi, and Jie Yin. 2013. Location extraction from disaster-related microblogs. In *Proc. WWW*. 1017–1020.

[21] Shervin Malmasi and Mark Dras. 2015. Location Mention Detection in Tweets and Microblogs. In *Proc. PACLING*. 123–134.

[22] Lih-Bin Oh and Heng Xu. 2003. Effects of multimedia on mobile consumer behavior: An empirical study of location-aware advertising. *Proc. ICiS* (2003), 56.

[23] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proc. EMNLP*. 1532–1543.

[24] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. NAACL-HLT*. 2227–2237.

[25] Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. 2017. NeuPL: Attention-based Semantic Matching and Pair-Linking for Entity Disambiguation. In *Proc. CIKM*. 1667–1676.

[26] Adam Rae, Vanessa Murdock, Adrian Popescu, and Hugues Bouchard. 2012. Mining the web for points of interest. In *Proc. SIGIR*. 711–720.

[27] Lev-Arie Ratinov and Dan Roth. 2009. Design Challenges and Misconceptions in Named Entity Recognition. In *Proc. CoNLL*. 147–155.

[28] Jochen Schiller and Agnès Voisard. 2004. *Location-based services*. Elsevier.

[29] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. DiSAN: Directional Self-Attention Network for RNN/CNN-Free Language Understanding. In *Proc. AAAI*.

[30] Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. 2013. Linking named entities in Tweets with knowledge base via user interest modeling. In *Proc. SIGKDD*. 68–76.

[31] Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015. Modeling Mention, Context and Entity with Neural Networks for Entity Disambiguation. In *Proc. IJCAI*. 1333–1339.

[32] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large Margin Methods for Structured and Interdependent Output Variables. *Journal of Machine Learning Research* 6 (2005), 1453–1484.

[33] Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. 2016. Multi-task cross-lingual sequence tagging from scratch. *arXiv preprint arXiv:1603.06270* (2016).

[34] Wei Zhang and Judith Gelernter. 2014. Geocoding location expressions in Twitter messages: A preference learning method. *J. Spatial Information Science* 9, 1 (2014), 37–70.

[35] Yue Zhang and Stephen Clark. 2008. Joint Word Segmentation and POS Tagging Using a Single Perceptron. In *Proc. ACL*. 888–896.

[36] Xin Zheng, Jialong Han, and Aixin Sun. 2018. A Survey of Location Prediction on Twitter. *IEEE Trans. Knowl. Data Eng.* 30, 9 (2018), 1652–1671.

Recurrent Convolutional Neural Network for Sequential Recommendation

Chengfeng Xu
Institute of AI, Soochow University
suzhou, China
cfxu@stu.suda.edu.cn

Pengpeng Zhao*
Institute of AI, Soochow University
suzhou, China
ppzhao@suda.edu.cn

Yanchi Liu
Rutgers University
New Jersey, USA
yanchi.liu@rutgers.edu

Jiajie Xu
Institute of AI, Soochow University
suzhou, China
xujj@suda.edu.cn

Victor S.Sheng
University of Central Arkansas
Conway, Arkansas, USA
ssheng@uca.edu

Zhiming Cui
Suzhou University of Science and Technology, China
zmcui@mail.usts.edu.cn

Xiaofang Zhou
The University of Queensland
Queensland, Australia
zxf@itee.uq.edu.au

Hui Xiong
Rutgers University
New Jersey, USA
hxiong@rutgers.edu

ABSTRACT

The sequential recommendation, which models sequential behavioral patterns among users for the recommendation, plays a critical role in recommender systems. However, the state-of-the-art Recurrent Neural Networks (RNN) solutions rarely consider the non-linear feature interactions and non-monotone short-term sequential patterns, which are essential for user behavior modeling in sparse sequence data. In this paper, we propose a novel **R**ecurrent **C**onvolutional **N**eural **N**etwork model (RCNN). It not only utilizes the recurrent architecture of RNN to capture complex long-term dependencies, but also leverages the convolutional operation of Convolutional Neural Network (CNN) model to extract short-term sequential patterns among recurrent hidden states. Specifically, we first generate a hidden state at each time step with the recurrent layer. Then the recent hidden states are regarded as an "image", and RCNN searches non-linear feature interactions and non-monotone local patterns via intra-step horizontal and inter-step vertical convolutional filters, respectively. Moreover, the output of convolutional filters and the hidden state are concatenated and fed into a fully-connected layer to generate the recommendation. Finally, we evaluate the proposed model using four real-world datasets from various application scenarios. The experimental results show that our model RCNN significantly outperforms the state-of-the-art approaches on sequential recommendation.

KEYWORDS

Sequential Recommendation; Recurrent Neural Network; Convolutional Neural Network.

*The corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313408

ACM Reference Format:
Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Jiajie Xu, Victor S.Sheng, Zhiming Cui, Xiaofang Zhou, and Hui Xiong. 2019. Recurrent Convolutional Neural Network for Sequential Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313408

1 INTRODUCTION

Recommender systems play a significant role in engaging and satisfying users with services provided in the age of information explosion. To build effective recommender systems, a key factor is to accurately characterize users' interests and tastes, which are intrinsically dynamic and evolving. To achieve this goal, the sequential recommendation has been proposed to recommend the successive item(s) that a user is likely to interact with based on his/her past activity sequences.

Early sequential methods like Markov chains [18] are proposed for modeling user's sequential pattern. However, these methods based on Markov chains model local sequential behaviors between every two adjacent items but have difficulties to model high-order relationships. More recently, deep neural networks have been intensively studied in related domains and made extraordinary impacts on sequential recommendation. For instance, Recurrent Neural Network (RNN) has become a popular sequential neural model. For better holding the long-term dependencies, two variants of RNN in particular, namely the Long-Short Term Memory (LSTM) [5] and the Gated Recurrent Unit (GRU) [1] have been widely used. Besides, other variants of RNN, including session-based RNN [4], hierarchical personalized RNN [10], and attention-based RNN [8] are found helpful in modeling sequences.

However, RNN holds an assumption that temporal dependency changes monotonically [9], which means that the current item or hidden state is more important than the previous one. This monotonic temporal dependency of RNN impairs the modeling of user's short-term interest. Literature work has shown that feature interactions can help capture sequential patterns especially with

sparse data [3, 11, 21]. Thus, it is crucial to consider the interactions between hidden state features in RNN when learning from sparse sequential data. In addition, CNN and RNN have been integrated for various application scenarios, such as image processing [2] and text classification [7]. However, none of them are designed for sequential recommendation.

In this paper, we propose a novel recurrent neural network model, namely **R**ecurrent **C**onvolutional **N**eural **N**etwork (RCNN), for sequential recommendation. It leverages the strengths of both the recurrent architecture of LSTM to capture complex long-term dependencies and the convolutional operation of CNN to extract local sequential patterns among hidden states. Specifically, we first generate a hidden state, *i.e*, a hidden sequential preference representation, at each time step by inputting a current item into the recurrent layer. Then, we treat recent hidden states of each time step as an "image" and search for local sequential features using horizontal and vertical convolutional filters. An intra-step horizontal filter is used to capture non-linear feature interactions, while an inter-step vertical filter is used for non-monotone local patterns. Moreover, the outputs of CNN and the hidden state vector of LSTM are concatenated together to describe user's overall interest, and then are fed into a fully-connected layer to generate a recommendation list. In summary, the main contributions of this paper are listed as follows:

- To the best of our knowledge, it is the first to propose to integrate both RNN and CNN to combine user's long- and short-term preference to generate a high-level hybrid representation for sequential recommendation.
- RCNN uses horizontal and vertical convolutional filters to learn high-order correlations among hidden state from locally to globally in a hierarchical way.
- We conduct experiments on three real datasets which demonstrate the superior performance of the proposed model with other state-of-the-art methods.

2 RELATED WORK

Different from general recommendation, sequential recommendation views the interactions as a sequence in time order and aims to predict the successive item(s) that a user is likely to interact with in the near future. Early sequential recommendation methods are based on Markov chains, which use sequential connections between users' actions to make prediction. A typical model FPMC [13] factorizes a transition matrix into item latent factors by integrating matrix factorization and Markov chains. HRM [18] further extends the idea by leveraging representations learning as latent factors. However, the above methods focus on modeling local sequential patterns between two adjacent actions while neglecting the global information of the whole sequences.

Recently, researchers have a surge of interest in applying deep learning to recommendation systems. Restricted Boltzmann Machine (RBM) [14] is one of the first neural networks applied to solve recommendation problems. Auto-encoders have also been a popular choice as the deep learning architecture of recommender systems [15, 24]. In addition, with the recent success of CNN on capturing local feature representation of images and texts [6, 7], it has shown a notable performance for top-n sequential recommendation. Caser [17] proposed to embed a sequence of recent items into "image" in the latent spaces and learn local features of the image as a sequential pattern using convolutional filters. However, CNN is capable of extracting local information while may fail to capture long-term dependencies.

On the other hand, as one of the most popular neural networks, Recurrent Neural Network (RNN) has been widely applied to model sequential data. Hidasi et al. applied RNN to model session data with a gated recurrent unit, which utilized session-parallel mini-batches and employed ranking loss functions for learning the model [4]. Tan et al. further proposed a data augmentation technique to improve the performance of RNN for session-based recommendation [16]. However, RNN holds a monotonicity assumption that the current item or hidden state is more important than the previous one, which impairs modeling of user's short-term interest. To deal with this problem, attentional mechanisms have been recently explored due to the ability of modeling user's attention. For example, NARM [8] was proposed as a hybrid encoder with an attention mechanism to emphasize user's main purpose in the current session. SHAN [22] was designed as a two-layer hierarchical attention network to take user-item and item-item interactions into account.

With the rapid development of deep neural networks, researchers further proposed to integrate CNN and RNN for various application scenarios. Lai et al. applied CNN to learn representation of texts and a bi-directional recurrent structure to capture contextual information for text classification [7]. Haque et al. designed a convolutional neural network as encoder and a multilayer LSTM network as decoder for image denoising [2]. However, none of these studies are designed for sequential recommendation and we propose to integrate RNN and CNN in an innovative way. The proposed RCNN utilizes the recurrent architecture of RNN to capture complex long-term dependencies, while employs the convolutional operation of CNN to extract short-term sequential patterns and item-specific features from the hidden states generated by RNN. With this integration method, RCNN is able to address sequential recommendation problems.

3 PROBLEM STATEMENT

In this section, we first introduce basic notations that will be used in this paper. Let $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ and $\mathcal{I} = \{i_1, i_2, ..., i_{|\mathcal{I}|}\}$ represent the sets of users and items respectively. Our task focuses on recommendation scenarios with implicit feedback [12, 13], which concerns whether a user $u \in \mathcal{U}$ is interacted with an item $i \in \mathcal{I}$ at time t. Each user is associated with a sequence of items from $\mathcal{I}$. By sorting interaction records in a chronological order, we can form the interaction sequence $\mathcal{I}^u = \{i_1^u, ..., i_t^u, ..., i_{|\mathcal{I}^u|}^u\}$ for user u, where $i_t^u \in \mathcal{I}$. The index t for i_t^u denotes the relative time index, instead of absolute time index in temporal recommendation like [13, 19].

With the above notations, we define the sequential recommendation task as follows. Given a user u's history sequence $\mathcal{I}^u$, we aim to infer the item(s) that user u would probably interact with in the future. Note that it is straightforward to convert the above task into *next-item* recommendation at time $t + 1$ or $next\text{-}future$ recommendation in the near future by replacing the next item with

the next item collection. For the sake of simplicity, we keep the *next-future* recommendation as the major task throughout the paper.

4 THE PROPOSED APPROACH

In this section, we introduce the proposed Recurrent Convolutional Neural Network model (RCNN), which incorporates a recurrent neural network to learn users' long-term interests and a convolutional neural network to learn users' sequential patterns and item-specific features among hidden states. As shown in Figure 1, RCNN consists of three components: a Recurrent Layer, Convolutional Layers, and a Fully-connected Layer. In the following subsections, we will introduce each part of our model and corresponding network learning strategy in detail.

4.1 Recurrent Layer

The first layer of RCNN is a long-short term memory network. LSTM [5], a variant of RNN, is able to alleviate the problem of "vanishing gradients" suffered from RNN by introducing a gate mechanism to hold long-term dependencies. Therefore, we use LSTM model in our approach for the concise and general purpose. Given the input $\mathbf{x}_t$ at the time t, the output $\mathbf{h}_t$ of the LSTM hidden layer is computed as follows.

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i),\\
\mathbf{f}_t &= \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f),\\
\tilde{\mathbf{c}}_t &= tanh(\mathbf{W}_c[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c),\\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t,\\
\mathbf{o}_t &= \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o),\\
\mathbf{h}_t &= \mathbf{o}_t \odot tanh(\mathbf{c}_t).
\end{aligned} \tag{1}$$

where $\mathbf{h}_0 = 0$, $\sigma(\cdot)$ is a logistic sigmoid function to do non-linear projection, and $\odot$ is the Hadamard product between two vectors. $\mathbf{i}_t, \mathbf{f}_t, \mathbf{o}_t$ and $\mathbf{c}_t$ are an input gate, a forget gate, an output gate and a cell state of the t-th object, respectively. $\mathbf{h}_t$ represents the hidden state vector to remember and store nodes of past states.

4.2 Convolutional Layers

The outputs of the recurrent layer are then fed simultaneously into a horizontal convolution layer and a vertical convolution layer. We present the previous k hidden states of user u at time t as a $k \times d$ matrix $\mathbf{E}^{(u,t)}$, where d is the hidden size and the rows of the matrix preserve the order of items. And then we regard the matrix $\mathbf{E}^{(u,t)}$ as an "image" to make the horizontal and the vertical convolution operations respectively. More details are explained below.

Horizontal Convolution on Hidden State. The convolution involves a filter vector sliding over a sequence, and picks up signals for sequential patterns at different time steps. The matrix $\mathbf{F} \in \mathbb{R}^{1\times w}$ is a filter for the horizontal convolution, where the height of a horizontal filter is 1 and the width equals to w. For each time step t in the sequence $\mathcal{I}^u$, we have a window matrix $\mathbf{E}^{(u,t)} \in \mathbb{R}^{1\times d}$ with a single hidden state (*i.e.*, $\mathbf{E}^{(u,t)} = h_t$). The result of the interaction for window matrix $\mathbf{E}^{(u,t)}$ is produced as follows:

$$v_t = \sum_{j=0}^{d-w+1} f(\mathbf{E}^{(u,t)}_{j:j+w-1} \cdot \mathbf{F} + b) \tag{2}$$

where $\cdot$ is the inner product, j is the position of $\mathbf{E}^{(u,t)}$'s row vector, $b \in \mathbb{R}$ is a bias term and f is the ReLU activation function. Moreover, we use n horizontal filters with the same size to generate multiple convolution value. The final horizontal convolution result $\mathbf{o}_t \in \mathbb{R}^n$ is

$$\mathbf{o}_t = [v_t^1\ v_t^2\ ...\ v_t^n] \tag{3}$$

where v_t^l is the convolution value generated with the l-th filter and $\mathbf{o}_t$ is a new feature representation generated from n filters for the window vector at time step t and is considered as the current item's feature interaction.

Vertical Convolution on Hidden State. We use tilde(~) for the symbols of the vertical convolution. We denote the vertical filter $\tilde{\mathbf{F}} \in \mathbb{R}^{\tilde{k}\times 1}$ for the operation of vertical convolution. For each time step t in the sequence $\mathcal{I}^u$, we have a window matrix $\tilde{\mathbf{E}}^{(u,t)} \in \mathbb{R}^{\tilde{k}\times d}$ with $\tilde{k}$ consecutive items, denoted as:

$$\tilde{\mathbf{E}}^{(u,t)} = [\mathbf{h}_{t-\tilde{k}+1}, ..., \mathbf{h}_{t-1}, \mathbf{h}_t] \tag{4}$$

where each commas represents a row vector concatenation and the full width of $\tilde{\mathbf{E}}$ equals to d. $\tilde{\mathbf{F}}$ interacts with each column of $\tilde{\mathbf{E}}^{(u,t)}$ by sliding d times from left to right for user u at time step t in a valid way, and finally yielding the vertical convolution result $\tilde{\mathbf{v}}_t \in \mathbb{R}^d$.

$$\tilde{\mathbf{v}}_t = [\tilde{v}_1\ \tilde{v}_2\ ...\ \tilde{v}_d] \tag{5}$$

$$\tilde{\mathbf{o}}_t = \tilde{\mathbf{v}}_t \odot \mathbf{h}_t. \tag{6}$$

where $\tilde{\mathbf{v}}_t$ is equal to the weights of recent items' hidden features and $\tilde{\mathbf{o}}_t$ is the final output of the vertical convolution, which denotes user u's local sequential pattern at time t. With one vertical filter we can learn to enhance the impact of the k previous items' hidden states on current interest for user u, which highlights user's short-term interest within recent $\tilde{k}$ time steps.

4.3 Fully-connected Layer

To capture the current overall preference of user u, we concatenate the outputs of the two convolutional layers and the hidden vector of LSTM together, and project them to a fully-connected layer with $\mathcal{I}$ nodes, written as

$$\mathbf{y}^{(u,t)} = \sigma(\mathbf{W}' \begin{bmatrix} \mathbf{o}_t \\ \mathbf{h}_t \\ \tilde{\mathbf{o}}_t \end{bmatrix} + \mathbf{b}') \tag{7}$$

where $\mathbf{W}' \in \mathbb{R}^{|\mathcal{I}|\times(2d+n)}$ and $\mathbf{b}' \in \mathbb{R}^{|\mathcal{I}|}$ are the weight matrix and the bias term for the output layer, respectively. $\mathbf{h}_t$ intends to capture user's long-term preferences, whereas $\mathbf{o}_t$ learns item-specific features and $\tilde{\mathbf{o}}_t$ captures short-term sequential patterns. The value $y_i^{(u,t)}$ is associated with the probability of how likely user u will interact with item i at time step t.

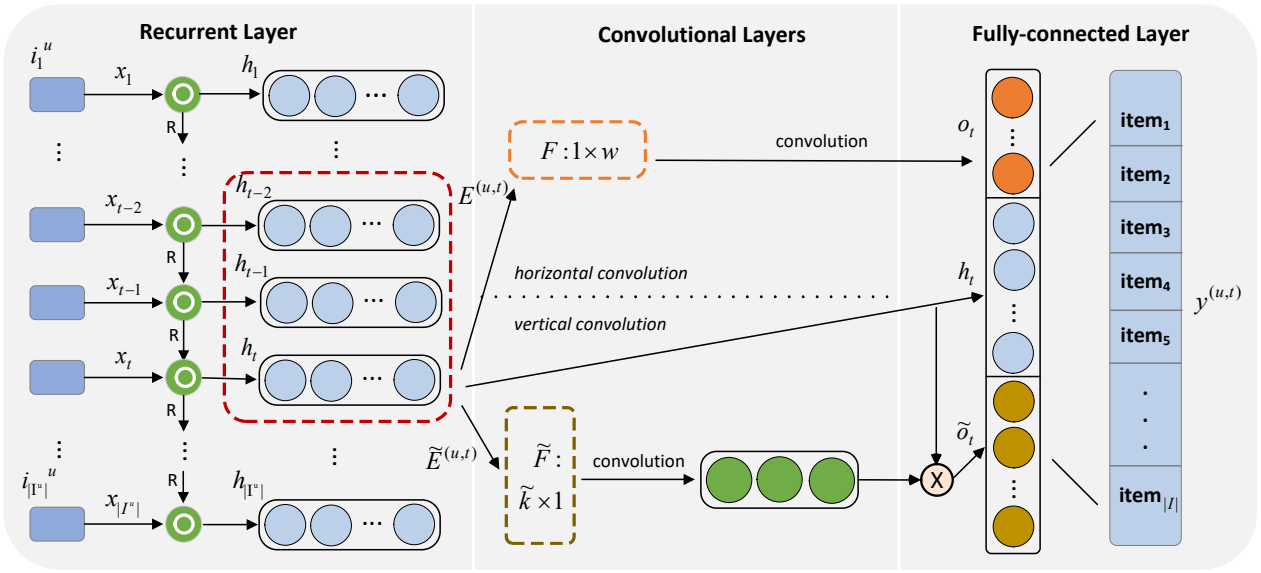


Figure 1: The overall architecture of RCNN. The red dash rectangular box represents the window matrix, while the orange and brown dash rectangular boxes are horizontal and vertical convolutional filters.

4.4 Network Learning

At last, we adapt cross-entropy as our loss function for model optimization. The objective function to be minimized in the model optimization is define as follows:

$$\mathcal{L} = -\sum_{u}^{|\mathcal{U}|}\sum_{t}^{|\mathcal{I}^u|} \hat{\mathbf{y}}^{(u,t)} log(\mathbf{y}^{(u,t)}) + (1-\hat{\mathbf{y}}^{(u,t)}) log(1-\mathbf{y}^{(u,t)}) + \lambda||\Theta||^2 \tag{8}$$

where $\mathbf{y}^{(u,t)}$ is the probability of all items interacted by user u at time $t+1$, and $\hat{\mathbf{y}}^{(u,t)}$ denotes a one-hot vector. If item i is interacted at time $t+1$, $\hat{\mathbf{y}}_i^{(u,t)} = 1$. Otherwise, $\hat{\mathbf{y}}_i^{(u,t)} = 0$. Θ represent all of the model parameters that are be learned and λ is the regularization weight .

5 EXPERIMENTS

In this section, we first introduce the experiment settings. And then we conduct experiments to answer the following research questions:

Q1: What is the performance of our model RCNN, comparing to other state-of-the-art methods?

Q2: How much do the horizontal and vertical convolutional operations help RCNN?

Q3: How do the parameters, such as the size of convolutional filters, affect the performance of RCNN?

5.1 Experimental Setup

5.1.1 Datasets. We study the effectiveness of our proposed approach on three public datasets with different kinds of items. Gowalla[1] and Foursquare[2] are two widely used check-ins datasets, which contains user-POI check-ins information in the location-based social networking sites (LBSN). Taobao is a user-purchase dataset obtained from IJCAI 2017 competition [3]. We eliminated users interacting with fewer than 15 items and items interacted by fewer than 10 users in the three datasets. After preprocessing, the basic statistics of all datasets are listed in Table 1.

Table 1: Statistics of the three datasets.

Dataset	#users	#items	#feedbacks	avg. seq. len	sparsity(%)
Gowalla	5,073	7,020	252,944	49.86	99.29
Foursquare	2,031	3,113	106,229	52.30	98.32
Taobao	2,481	1,419	72,553	29.24	97.94

5.1.2 Evaluation Metrics. For the two recommendation tasks, i.e., next-item and next-future recommendation, we divided the datasets into training/validation/test set with a 98/1/1 split and 70/10/20 split, respectively. In addition, we remove duplicate items from the test set.

To evaluate the performance of each approach, we adopt a variety of metrics widely used in previous works [4, 13], including *Hit Ratio*(HR@N), *Mean Average Precision* (MAP), *Precision*@N, and *Recall*@N. In our experiments, we choose $N = \{1, 5, 10\}$. To make results more stable, we repeated each experiment five times and used the average value of each metrics.

5.1.3 Baselines. We consider the following baselines for performance comparisons:

[1] http://snap.stanford.edu/data/loc-gowalla.html

[2] https://sites.google.com/site/yangdingqi

[3] https://tianchi.shuju.aliyun.com/datalab/dataSet

- **BPR-MF** [12] is a state-of-the-art method for non-sequential recommendation, which optimizes Matrix Factorization model using a pairwise ranking loss.
- **FPMC**[4][13] is a classic hybrid model combing MC and MF for next-basket recommendation. FPMC allows a basket of several items at each step. For our sequential recommendation, each basket has a single item.
- **Caser**[5] [17] is the first method of leveraging convolutional filters to learn users' sequential patterns for sequential recommendation. It incorporates the convolutional neural network with a latent factor model.
- **RUM**[6] [20] utilizes external memories to improve sequential recommendation, which contains two variants, item-level (RUM^I) and feature-level (RUM^F).
- **GRU** [1] is an extension of RNN [23], which is equipped with two gates to control the information flow for sequential recommendation.
- **LSTM** [5] is another variant of RNN, which contains a memory cell and three multiplicative gates to hold long-term dependencies.
- **NARM**[7] [8] is an RNN-based state-of-the-art model, which employs an item-level attention mechanism to capture user's main purpose from hidden states and combines it with the sequential behavior as user's final representation to generate recommendation.

5.2 Comparison of Performance

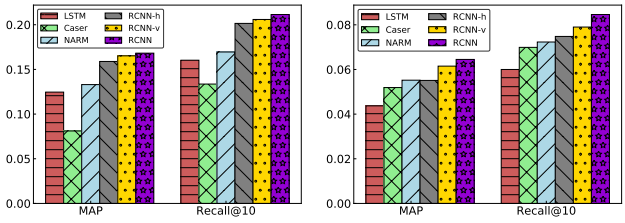
The results of different methods for both recommender tasks are illustrated in Table 2. It can be observed that:

First, among all baseline models, BPR-MF has the most unfavorable performance in nearly all the cases, since BPR-MF is an non-sequential recommendation method. The classic model FPMC performs better than BPR-MF on most cases by taking the sequential correlation between adjacent actions into account. The experimental results verify that sequential information can help improve the recommendation performance in real-world systems.

Second, both RNNs-based methods (*i.e.*, GRU, LSTM and NARM) and memory-based models (*i.e.*, Caser and RUM) improve the effectiveness by using deep neural networks. This shows the ability of neural networks on modeling users' general tastes and their sequential behaviors. On Foursquare and Gowalla datasets, both Caser and RUM achieve better performances than LSTM and GRU. But they get much worse performances on Taobao dataset. In fact, Taobao dataset have more sequential signals than Foursquare and Gowalla datasets. Therefore, both LSTM and GRU can achieve favorable performances, because of their advantages in modeling long-term dependencies. It calls us to combine a deep model with recurrent structure for improvements.

Third, in terms of memory-based methods, RUM performs better than Caser under the metric of Prec@N on Taobao dataset, yet performs worse under the metrics of Recall@N and MAP on most datasets. Overall, RUM and Caser perform well, which indicates that both KV-MN architecture and convolutional operations can capture users' preferences. Furthermore, the state-of-the-art attention-based RNN model NARM beats all the other baselines on all datasets owing to the effectiveness of the attentive mechanism. This shows the effectiveness of the attentive mechanism to capture user's evolving appetite for items.

Finally, we compare our proposed model RCNN with all the baselines. It is clear to see that RCNN consistently and significantly outperforms these baselines on all datasets. This shows that leveraging the power of convolutional neural network, RCNN can model users' sequential pattern better and capture many important features, resulting in a better recommendation. Moreover, we also observe that the performances of most models degrade as the datasets become sparser.



(a) Taobao (b) Gowalla

Figure 2: Performance with variants of RCNN in terms of Rec@10 and MAP on two datasets.

5.3 Influence of Components

To evaluate the contribution of convolutional filters to form user's final hybrid representation, we conduct experiments to analyze each component of convolution. Figure 2 shows MAP and Recall@10 on Taobao and Gowalla. (We obtained the similar results on another dataset.) RCNN-h and RCNN-v denote a downgrade version of RCNN that only uses horizontal and vertical convolution, respectively. From Figure 2, we can observe that Caser outperforms LSTM on Gowalla, while LSTM performs better than Caser on Taobao. This suggests that Caser's performance is greatly affected by the sequential signals of the data. NARM performs better than Caser and LSTM on both datasets due to the attention mechanism. However, NARM performs worse than RCNN-v, which indicates that attention mechanism of vertical convolution plays a better impact than NARM due to use of the filter and nonlinear function. RCNN-h performs better than NARM on most cases, suggesting the feature interactions of items are also important for sequential recommendation. Overall, our final model RCNN performs the best on the two datasets.

5.4 Influence of Hyper-parameters

In this subsection, we investigate the impact of the parameters w and $\tilde{k}$ in our model. Due to space limit, we only show MAP on Foursquare and Taobao datasets. We apply grid search over combinations of $w \in [8, 16, 32, 64, 128]$ and $\tilde{k} \in [2, 3, 4, 5, 6]$ for a better recommendation performance. The experiment results are shown

[4]https://github.com/khesui/FPMC
[5]https://github.com/graytowne/caser_pytorch
[6]https://github.com/ch-xu/RUM
[7]https://github.com/lijingsdu/sessionRec_NARM

Table 2: Performance comparison with baselines on all three datasets for next-item and next-future recommendation. The best performance is highlighted in boldface (higher is better).

Datasets	Methods	Next-item Recommendation				Next-future Recommendation						
		HR@1	HR@5	HR@10	MAP	Prec@1	Prec@5	Prec@10	Recall@1	Recall@5	Recall@10	MAP
Gowalla	BPR-MF	0.0185	0.0481	0.0645	0.0382	0.1426	0.0787	0.0539	0.0128	0.0330	0.0450	0.0315
	FPMC	0.0219	0.0582	0.0755	0.0421	0.1451	0.0788	0.0563	0.0146	0.0342	0.0450	0.0349
	Caser	0.0231	0.0674	0.0954	0.0486	0.1579	0.0906	0.0672	0.0164	0.0455	0.0699	0.0519
	RUM^I	0.0351	0.0684	0.0863	0.0537	0.1547	0.0846	0.0597	0.0139	0.0380	0.0512	0.0380
	RUM^F	0.0337	0.0670	0.0887	0.0543	0.1570	0.0799	0.0565	0.0149	0.0347	0.0480	0.0360
	GRU	0.0394	0.0997	0.1415	0.0738	0.1401	0.0858	0.0643	0.0133	0.0404	0.0594	0.0429
	LSTM	0.0428	0.1080	0.1380	0.0773	0.1491	0.0884	0.0652	0.0139	0.0405	0.0600	0.0437
	NARM	0.0559	0.1307	0.1715	0.0995	0.1892	0.1034	0.0765	0.0189	0.0506	0.0730	0.0552
	RCNN	**0.0574**	**0.1427**	**0.1950**	**0.1031**	**0.2107**	**0.1290**	**0.0961**	**0.0211**	**0.0579**	**0.0846**	**0.0645**
	Improv.	2.68%	9.18%	13.70%	3.62%	11.36%	24.76%	25.62%	11.64%	14.43%	15.89%	16.85%
Foursquare	BPR-MF	0.0389	0.0926	0.1216	0.0677	0.1201	0.0754	0.0796	0.0095	0.0209	0.0413	0.0268
	FPMC	0.0399	0.0822	0.0955	0.0621	0.1952	0.1013	0.0621	0.0161	0.0393	0.0476	0.0371
	Caser	0.0177	0.0448	0.0635	0.0356	0.1674	0.0951	0.0721	0.0179	0.0429	0.0602	0.0482
	RUM^I	0.0592	0.1172	0.1418	0.0902	0.2068	0.1122	0.0809	0.0163	0.0402	0.0572	0.0451
	RUM^F	0.0605	0.1236	0.1472	0.0957	0.2319	0.1120	0.0773	0.0185	0.0423	0.0558	0.0460
	GRU	0.0443	0.0980	0.1280	0.0738	0.1327	0.0775	0.0584	0.0101	0.0301	0.0448	0.0343
	LSTM	0.0409	0.1024	0.1305	0.0732	0.1574	0.0930	0.0683	0.0124	0.0344	0.0502	0.0399
	NARM	0.0611	0.1364	0.1713	0.1006	0.2462	0.1268	0.0914	0.0191	0.0484	0.0683	0.0547
	RCNN	**0.0694**	**0.1487**	**0.1984**	**0.1124**	**0.2748**	**0.1477**	**0.1040**	**0.0205**	**0.0537**	**0.0744**	**0.0602**
	Improv.	13.58%	9.02%	15.82%	11.73%	11.62%	16.39%	13.79%	7.33%	10.95%	8.93%	10.05%
Taobao	BPR-MF	0.0508	0.1749	0.2656	0.1176	0.1608	0.1309	0.1052	0.0170	0.0735	0.1208	0.0807
	FPMC	0.0601	0.1809	0.2447	0.1225	0.2126	0.1416	0.1020	0.0235	0.0764	0.1139	0.0808
	Caser	0.0290	0.0842	0.1189	0.0581	0.1374	0.0888	0.0669	0.0277	0.0885	0.1336	0.0811
	RUM^I	0.0705	0.2209	0.3249	0.1530	0.2112	0.1382	0.1083	0.0248	0.0786	0.1238	0.0842
	RUM^F	0.0826	0.2390	0.3370	0.1649	0.2140	0.1487	0.1125	0.0245	0.0849	0.1288	0.0866
	GRU	0.1511	0.3108	0.3906	0.2321	0.4306	0.2063	0.1357	0.0465	0.1148	0.1544	0.1195
	LSTM	0.1737	0.3527	0.4208	0.2593	0.4346	0.2123	0.1405	0.0491	0.1191	0.1603	0.1246
	NARM	0.1943	0.3724	0.4478	0.2834	0.4486	0.2195	0.1489	0.0492	0.1233	0.1695	0.1330
	RCNN	**0.2656**	**0.4752**	**0.5506**	**0.3672**	**0.5566**	**0.2804**	**0.1832**	**0.0622**	**0.1589**	**0.2114**	**0.1682**
	Improv.	36.70%	27.60%	22.96%	29.57%	24.07%	27.74%	23.04%	26.42%	28.87%	24.72%	26.47%

Table 3: The performance of RCNN with varying w and $\tilde{k}$ in terms of MAP on Foursquare and Taobao.

Dataset	MAP: w \ $\tilde{k}$	$\tilde{k}=2$	$\tilde{k}=3$	$\tilde{k}=4$	$\tilde{k}=5$	$\tilde{k}=6$
Foursquare	w =8	0.0577	0.0566	0.0587	0.0566	0.0564
	w =16	0.0567	0.0589	0.0577	0.0580	0.0570
	w =32	0.0584	0.0577	0.0594	0.0575	0.0569
	w =64	0.0598	0.0590	**0.0602**	0.0579	0.0574
	w =128	0.0596	0.0580	0.0578	0.0574	0.0574
Taobao	w =8	0.1671	0.1673	0.1668	0.1665	0.1671
	w =16	0.1672	0.1675	0.1687	0.1692	0.1672
	w =32	0.1675	0.1674	0.1696	0.1685	0.1686
	w =64	0.1671	0.1693	0.1693	0.1684	0.1682
	w =128	**0.1698**	0.1695	0.1678	0.1672	0.1695

in Table 3. We can see that either fixing w or fixing $\tilde{k}$, the performance of our model is improved first and then deteriorated on most cases. This suggests that a proper size of filters can capture users' short-term interest and high-order feature interactions better. Consequently, the best sizes are chosen as $w = 64, \tilde{k} = 4$ on Foursquare and $w = 128, \tilde{k} = 2$ on Taobao. $\tilde{k}$ on Foursquare is larger than the value on Taobao. It may be because the average sequence length on Foursquare is longer.

6 CONCLUSIONS

In this paper, we proposed a novel **R**ecurrent **C**onvolutional **N**eural **N**etwork model (RCNN), which leverages the strengths of both RNN and CNN for sequential recommendation. RCNN utilizes the recurrent architecture of RNN to capture complex long-term dependencies, while uses the convolutional operation of CNN to extract short-term sequential patterns. Our experimental results on three real-world datasets show that RCNN significantly outperformed the state-of-the-art approaches on sequential recommendation.

7 ACKNOWLEDGMENTS

This research was partially supported by the NSFC (61876117, 61876217, 61872258, 61728205), the Suzhou Science and Technology Development Program (SYG201803) and the Open Program of Neusoft Corporation (SKLSAOP1801).

REFERENCES

[1] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).
[2] Kazi Nazmul Haque, Mohammad Abu Yousuf, and Rajib Rana. 2018. Image denoising and restoration with CNN-LSTM Encoder Decoder with Direct Attention. *arXiv preprint arXiv:1801.05141* (2018).
[3] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *SIGIR*. ACM, 355–364.
[4] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).
[5] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *NIPS*. 1097–1105.
[7] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification.. In *AAAI*, Vol. 333. 2267–2273.
[8] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *CIKM*. ACM, 1419–1428.
[9] Qiang Liu, Shu Wu, and Liang Wang. 2017. Multi-behavioral sequential prediction with recurrent log-bilinear model. *IEEE Transactions on Knowledge and Data Engineering* 29, 6 (2017), 1254–1267.
[10] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent

neural networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 130–137.
[11] S. Rendle. 2010. Factorization machines. In *ICDM*. IEEE, 995–1000.
[12] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.
[13] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *WWW*. ACM, 811–820.
[14] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted Boltzmann machines for collaborative filtering. In *ICML*. 791–798.
[15] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. AutoRec: Autoencoders Meet Collaborative Filtering. In *WWW*. 111–112.
[16] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved Recurrent Neural Networks for Session-based Recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 17–22.
[17] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*. ACM, 565–573.
[18] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning Hierarchical Representation Model for NextBasket Recommendation. In *SIGIR*. 403–412.
[19] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent recommender networks. In *WSDM*. ACM, 495–503.
[20] Chen X, Xu H, Zhang Y, Tang J, Cao Y, Qin Z, and Zha H. 2018. Sequential recommendation with user memory networks. In *WSDM*. ACM, 108–116.
[21] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In *IJCAI*. 3119–3125.
[22] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential Recommender System based on Hierarchical Attention Networks. In *IJCAI*. 3926–3932.
[23] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model for next basket recommendation. In *SIGIR*. ACM, 729–732.
[24] Alice X. Zheng, Alice X. Zheng, Alice X. Zheng, and Martin Ester. 2016. Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. In *WSDM*. 153–162.

No More than What I Post: Preventing Linkage Attacks on Check-in Services

Fengli Xu
Beijing National Research Center for Information Science and Technology Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
liyong07@tsinghua.edu.cn

Zhen Tu
Beijing National Research Center for Information Science and Technology Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
liyong07@tsinghua.edu.cn

Hongjia Huang
Beijing National Research Center for Information Science and Technology Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
liyong07@tsinghua.edu.cn

Shuhao Chang
Beijing National Research Center for Information Science and Technology Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
liyong07@tsinghua.edu.cn

Funing Sun
Tecent Inc., Beijing 100084, China
liyong07@tsinghua.edu.cn

Diansheng Guo
Tecent Inc., Beijing 100084, China
liyong07@tsinghua.edu.cn

Yong Li
Beijing National Research Center for Information Science and Technology Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
liyong07@tsinghua.edu.cn

ABSTRACT

With the flourishing of location based social networks, posting check-ins has become a common practice to document one's daily life. Users usually do not consider check-in records as violations of their privacy. However, through analyzing two real-world check-in datasets, our study shows that check-in records are vulnerable to linkage attacks. To address this problem, we design a *partition-and-group* framework to integrate the information of check-ins and additional mobility data to attain a novel privacy criterion — $k^{\tau,l}$*-anonymity*. It ensures adversaries with arbitrary background knowledge cannot use check-ins to re-identify users in other anonymous datasets or learning unreported mobility records. The proposed framework achieves favorable performance against state-of-art baseline in terms of improving check-in utility by 24%∼57% while providing stronger privacy guarantee at the same time. We believe this study will open a new angle in attaining both privacy-preserving and useful check-in services.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313506

CCS CONCEPTS

• **Security and privacy** → **Privacy protections**; *Social network security and privacy*; • **Networks** → *Social media networks.*

KEYWORDS

Check-ins, privacy-preserving data publishing, linkage attacks, mobility data privacy

ACM Reference Format:
Fengli Xu, Zhen Tu, Hongjia Huang, Shuhao Chang, Funing Sun, Diansheng Guo, and Yong Li. 2019. No More than What I Post: Preventing Linkage Attacks on Check-in Services. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3308558.3313506

1 INTRODUCTION

Check-in service has now become a popular feature that is widely adopted by the mainstream social media platforms, such as Facebook, Twitter and Wechat. It facilitates users to document their daily activities with mobility trace and share them with public audience. Users usually do not associate the self-reported check-ins with privacy risks, since they only check-in to places they feel comfortable [4]. However, the uniqueness of human mobility often exposes their check-in records to linkage attacks, *i.e.*, revealing their identities and unreported mobility records in other anonymous mobility datasets, such as call detail records [5], transportation records [8], and credit card records [12]. Moreover, recent researches show

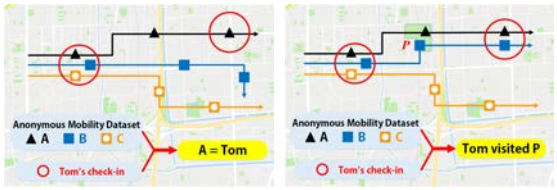
that most users are unaware or not able to fully anticipate the privacy risks embedded in posting check-ins [23]. Therefore, it is a paramount task for the check-in service providers to quantify the potential privacy exposures and put forward feasible solutions.

Previous efforts attempted to address the problem of linkage attacks on mobility data by ensuring user's anonymity in anonymous mobility datasets [12, 15]. That is, making sure adversary cannot achieve unique linkages based on user's check-ins through generalizing the records in anonymous mobility datasets. However, such approach often requires unacceptable data utility degeneration [35], and cannot prevent adversary from learning additional unreported mobility records [18]. It is also unrealistic for users and check-in service providers to assume all anonymous mobility datasets have been properly sanitized, since studies repeatedly demonstrated that insecure datasets had been irreversibly spread across the Internet [13, 22]. Therefore, these findings suggest it is impractical to prevent linkage attacks by sanitizing anonymous mobility datasets. In this paper, we investigate and address this problem through a novel angel — looking at the public mobility records, *i.e.*, check-ins.

In this paper, we put forward several contributions to attain both privacy-preserving and useful check-in services. First, we extend the frameworks of *k-anonymity* [26] and *l-diversity* [18] into check-in privacy preserving, and devise a novel privacy criterion $k^{\tau,l}$-anonymity. It ensures the posted check-ins cannot be exploited to distinguish user from at least other $k-1$ users in any anonymous mobility datasets, and for any time window of duration τ user's actual locations are indistinguishable from at least other $l-1$ locations. Second, we further propose a *partition-and-group* framework to optimize the check-in utility under $k^{\tau,l}$-anonymity privacy guarantee by carefully partitioning user population into small anonymity groups. Third, we conduct a thorough trace-driven evaluation on the proposed framework based on two real-world datasets. The evaluation results demonstrate that our framework significantly outperforms state-of-art baseline method in terms of achieving 24%∼57% check-in utility improvement while providing stronger privacy guarantee in the same time. In addition, 32%∼62% check-in utility boost of our framework is achieved by introducing additional mobility data, which showcases the benefits of integrating additional mobility data in privacy-preserving check-in service. Finally, our study reveals two intriguing trade-offs between the utility and privacy in check-in services: (*i*) in order to achieve modest privacy gains, users need to sacrifice significant check-in utility, *i.e.*, reducing spatio-temporal resolution of check-ins. (*ii*) users may increase the utility of their check-ins with same privacy level by letting check-in service providers to collect moderate amount of additional mobility data. Such findings may have direct implications on how to defend linkage attacks with the joint effort of check-in service providers and individual users.

2 RELATED WORKS

Linkage Attack: The linkage attacks were widely studied in multiple scenarios and had received increasing attention in recent years [17, 18, 26, 27]. The most prominent two branches are *re-identification attack* and *probabilistic attack* [7]. Specifically, the *re-identification attack* aims at recovering individuals' identities



(a) *Re-identification attack* (b) *Probabilistic attack*

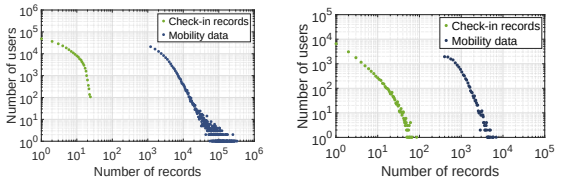
Figure 1: Illustration of linkage attacks on check-ins.

in anonymous datasets by achieving unique linkages with public datasets. For example, 87% of American population can be uniquely re-identified with the public accessible information of ZIP code, gender and date of birth [26]. Similar findings have been established in wide range of scenarios, including web browsing records [25], call detail records [35], app usage records [27, 32] and so on. One popular privacy model against such attack is *k*-anonymity, which requires to make the records of each individual indistinguishable from at least $k-1$ others [26]. On the other hand, *probabilistic attack* is a more generic linkage attack, which aims at improving some belief on individuals through correlating the datasets. Researchers demonstrated that by combining online social network data and sparse offline location data individual's locations can be predicted with high precision [19]. In addition, the salary class of individuals can be accurately inferred by correlating census data with public available information [18]. To defend such attacks, *l*-diversity and *t*-closeness have been proposed to ensure the diversity on the sensitive information within each anonymity group [17, 18, 29, 30].

We position our study in a novel scenario of defending against the linkage attacks on social media check-ins. We aim to provide strong privacy guarantee for check-in service against both *re-identification attack* and *probabilistic attack*, and design privacy solution compatible with unstructured spatio-temporal data.

Mobility Data Privacy: The literature in this area can be further broken down into two categories: aggregated mobility data privacy and individual mobility data privacy. As for the former, recent study found evidence that aggregated mobility data suffered from the risks of leaking individual trajectories [28, 33]. In addition, *differential privacy* has been applied on aggregated mobility data to provide provable privacy guarantees for individuals [1, 14]. As for individual mobility privacy, *geo-indistinguishability* model is devised to achieve practical privacy guarantee in individual mobility collecting [2, 6]. In addition, vast amount of literature were dedicated to ensure location anonymity in the context of geo-referenced queries in location based service [3, 10, 34]. On the other hand, *cloaking*, *generalization* and *suppression* techniques are leveraged to achieve *k*-anonymity in releasing anonymous individual trajectories [15, 16, 20, 21, 24]. However, recent studies showed that such approaches will be likely to result in significant data utility degeneration [11, 35].

We tackle the specific problem of designing privacy-preserving check-in service, which is closely related to prior effort in individual trajectories releasing. However, it differs from previous works that users have strong requirement for check-in utility and social desirability, which poses significant challenges on privacy solutions.



(a) WeChat dataset (b) Weibo dataset

Figure 2: The distribution of the number of user's check-ins and mobility records.

3 PROBLEM FORMULATION

We first introduce the privacy framework of k-anonymity and l-diversity, which is the basis of our model. Then, we elaborate on our novel privacy criterion — $k^{\tau,l}$-anonymity.

k-anonymity: The k-anonymity framework is originally devised to defend the re-identification attacks in relational database [26]. In check-in privacy scenario, the adversary attempts to recover user's identity in other anonymous mobility datasets by achieving a unique linkage between user's anonymous mobility data and public check-ins, which is illustrated in Figure 1(a). Therefore, it requires data sanitizing techniques that render each individual's attributes indistinguishable from at least other $k-1$ individuals', which forms an anonymity group that prevents any individuals from being uniquely re-identified.

l-diversity: It is put forward to ensure users' diversity on sensitive attributes within each anonymity group [18]. Specifically, it is designed to prevent *probabilistic attack*, where the adversary aims to learn additional mobility records the users do not report, which is illustrated in Figure 1(b). It requires each individual cannot be uniquely re-identified with the non-sensitive attributes (i.e., check-ins), while the sensitive attributes (i.e., unreported mobility records) should be of at least l different categories within each anonymity group.

$k^{\tau,l}$-anonymity: Inspired by the insights and limitations of previous models, we design a novel privacy criterion *$k^{\tau,l}$-anonymity* to address the privacy issues in check-in services. Specifically, *$k^{\tau,l}$-anonymity* requires:(i) any users on social media cannot be distinguished from at least other $k-1$ users in any other anonoymous mobility datasets based on their public check-ins; (ii) for any time window of duration τ users' unreported locations cannot be discriminated from at least $l-1$ other potential locations. Therefore, the knowledge adversary can acquire through linkage attacks, *i.e.*, users identity in other anonymous mobility datasets and unreported mobility records, is effectively bounded by user specific parameters k, τ, l. In other words, *$k^{\tau,l}$-anonymity* is able to provide strong privacy guarantee against both *re-identification attack* and *probabilistic attack*.

4 DATASETS

4.1 Data Collection

We utilize two real-world datasets collected from large scale user population in two mainstream social media platforms: WeChat and Weibo. The detailed information is discussed as follows.

WeChat Dataset[1]: WeChat platform is currently the most popular social media platform in China. This dataset consists of 530,050 check-ins collected from 100,000 users, which are randomly selected from the general user population spread across Beijing city. It covers two and a half month of usage, *i.e.*, from Jan. 1 to Mar. 15, 2018. We also collect an additional mobility dataset including over 193 millions mobility records from same user population during same time period. The mobility records are collected when users invoke location based services in WeChat, such as posting check-ins and using map services.

Weibo Dataset[2]: This dataset is collected by a previous research [31]. It contains 11,866,425 mobility records and 78,412 check-ins on Weibo platform from 17,425 users located in Shanghai during one week, *i.e.*, from Apr.19 to Apr.26, 2016. Different from WeChat dataset, the Weibo dataset is collected by internet service provider by performing deep packet inspection on cellular traffic.

To demonstrate the basic statistics of datasets, we show the probability distribution function (PDF) of number of mobility records and check-in records of each user in Figure 2. From the results, we can observe that they all follow a fat-tailed distribution, which echos with the findings in previous researches [9]. It indicates that our datasets are representative of typical check-in behaviors.

5 SOLUTION

5.1 Definitions

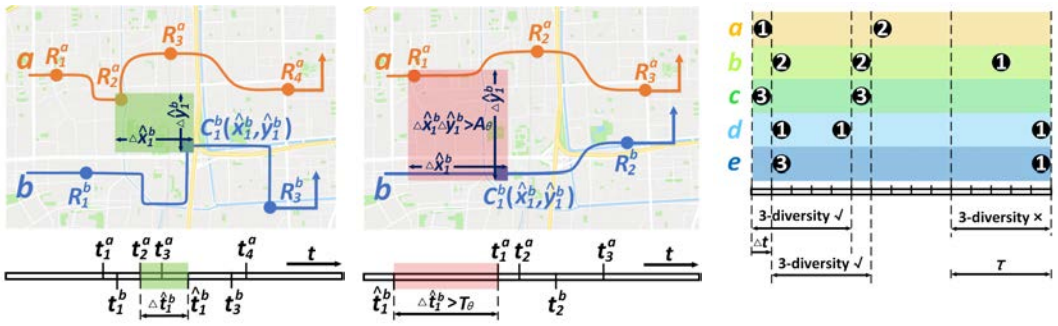
Formally, we define the additional mobility data of user i as $R^i = \left\{r_m^i\right\}$, where r_m^i is the m-th record of user i. It can be expressed as a tuple $r_m^i = (x_m^i,\ y_m^i,\ t_m^i)$, with x_m^i, $y_m^i$and t_m^i denoting the longitude, latitude and time stamp, respectively. On the other hand, we denote the *check-in records* as $C^i = \left\{c_m^i\right\}$, where c_m^i is the m-th check-ins of user i. Since the check-in records after sanitization may have various spatial and temporal resolution, c_m^i is defined as $(\hat{x}_m^i,\ \Delta\hat{x}_m^i,\ \hat{y}_m^i,\ \Delta\hat{y}_m^i,\ \hat{t}_m^i,\ \Delta\hat{t}_m^i)$, with $[\hat{x}_m^i,\ \hat{x}_m^i+\Delta\hat{x}_m^i] \times [\hat{y}_m^i,\ \hat{y}_m^i+\Delta\hat{y}_m^i]$ and $[\hat{t}_m^i,\ \hat{t}_m^i + \Delta\hat{t}_m^i]$ denoting the coverage in spatial and temporal dimensions, respectively.

5.2 Basic Operations

We limit our data sanitizing techniques to *generalization* and *suppression*, i.e., addressing the privacy problem by reducing check-in's spatiotemporal resolution or leaving out check-ins. Such operations avoid adding noises to check-in records that may displace users to places they never been to or injecting fabricated check-ins, which maintains the integrity of check-ins and avoid compromising their social figures. On the other hand, to effectively defend against *probabilistic attack*, we also define a *diversity check* operation to ensure the diversity on sensitive information within anonymity groups. In addition, we also define a cost function to measure the check-in utility loss in attaining privacy criterion. The basic operations are described as follows:

Generalization: It is to reduce spatial and temporal resolution of check-ins so that they overlap with other user's check-ins or unreported mobility records. In this way, the adversary can no longer uniquely link the check-ins with anonymous mobility data, which

[1]https://weixin.qq.com/
[2]https://weibo.com/



(a) *Generalization* (b) *Suppression* (c) *Diversity check*

Figure 3: The illustrations of three basic data sanitizing operations.

effectively prevents the *re-identification attacks*. We define the *generalization* operation as $G(c^\star, r^\star)$, where $c^\star$ and $r^\star$ are check-in and other user's mobility record, respectively. This operation will output *generalized* check-ins, which is demonstrated in Figure 3(a).

Suppression: When the spatial and temporal resolution of check-in records is too low, their utility is diminished. In real-world scenario, some "outlier" check-ins may require significant *generalization* to prevent *re-identification attacks*, which renders the check-ins useless. Specifically, *suppression* operation $S(c^\star)$ will return *true* for leaving out the check-ins $c^\star$ when spatial coverage exceed A_θ or temporal coverage exceed T_θ. That is, the system will recommend users not to post such check-ins. The *suppression* operation is demonstrated in Figure 3(b). Without loss of generality, A_θ and T_θ are set to $1000km^2$ and 120 hours, respectively.

Diversity check: We define *diversity check* operation as $D(\{R^\star\}, \tau, l)$, with $\{R^\star\}$ denoting the unreported mobility records of the inspected anonymity group. The illustration of *diversity check* is presented in Figure 3(c). Specifically, the operation search the total time duration with a sliding time window of duration τ and step length of minimal time resolution Δt. Then, it computes the number of distinct locations in each time window with each individual contribute at most one distinct location. If there is a time window with less than l distinct locations then the operation returns $false$ for failing the diversity check. Otherwise, it returns *true* for passing.

Cost function: It is defined as a linear combination of the spatial and temporal coverage of the investigated check-in, which can be computed as follows,

$$U(c_m^i) = \begin{cases} \lambda_a \cdot \sqrt{A} + \lambda_t \cdot T, & if A < A_\theta \text{ and } T < T_\theta, \\ \lambda_a \cdot \sqrt{A_\theta} + \lambda_t \cdot T_\theta, & otherwise, \end{cases}$$

where $A = |\Delta\hat{x}_m^i| \times |\Delta\hat{y}_m^i|$ and $T = |\Delta\hat{t}_m^i|$ denote the spatial and temporal coverage of generalized check-in. In this study, we set both λ_a and λ_t to 0.5, which indicates 1 km diameter of spatial coverage and 1 hour temporal coverage map to similar cost. In addition, since the check-ins are *suppressed* if their spatial coverage exceed A_θ or temporal coverage exceed T_θ, we set cost function at maximum value to represent complete lost in utility.

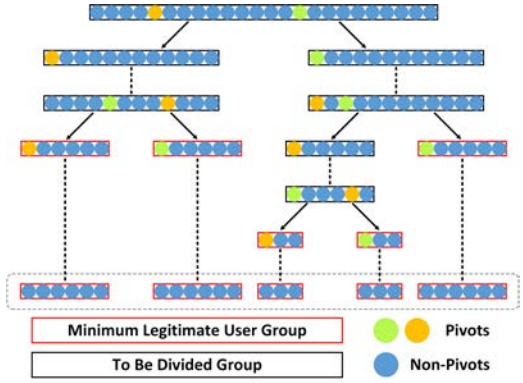


Figure 4: Illustration of *partition-and-group* framework.

5.3 Partition-and-Group Framework

One key problem in optimizing the privacy mechanism on large scale check-ins is how to partition the user population into optimal anonymity groups. The check-in utility will be significantly improved by carefully classifying the users into numerous small anonymity group that passes *diversity check* compared with putting all of them in one group. We use the word "legitimate" to refer to the anonymity groups that pass the *diversity check*. Achieving the optimal partition of user population requires to enumerate all the legitimate anonymity group, which is a NP-hard problem and cannot be readily solved in real-world scenario. We design a novel *partition-and-group* framework to efficiently optimize the check-in utility through a "divide-and-conquer" manner. The idea is to iteratively break down the user population into two small subsets until the *minimum legitimate anonymity groups* are met, which is illustrated in Figure 4. An important problem is determining whether a anonymity group is *minimum legitimate anonymity group*, i.e., the anonymity group cannot be divided into smaller subsets that all pass *diversity check*. We exploit a convenient property of *diversity check* to address this problem, which is formally described in the following proposition.

Proposition 5.1. *If an anonymity group does not pass the* diversity check*, then any subsets of this anonymity group will not pass the* diversity check.

Proof. Suppose the unreported mobility records of an anonymity group do not pass the *diversity check* of parameters (τ, l). Based

Algorithm 1: Partition-and-group algorithm.

Input: Check-in data $\mathbb{C}$, mobility data $\mathbb{R}$
Input: Anonymity k, diversity l, time window τ
Output: Sanitized check-in data $\hat{\mathbb{C}}$
foreach $i,\ j\ \in\ \mathbb{C},\ j \neq i$ **do**
  $\mathbb{C}^\star \leftarrow k^{\tau,\ l}$-merge$(\mathbb{C}[\{i,j\}], \mathbb{R}[\{i,j\}], 2, 0, 0)$;
  $W[a,b] \leftarrow sum([U(c^\star)|\ \forall\ c^\star \in \mathbb{C}^\star])$;
checkin_stack.insert$(\mathbb{C})$; *mobility_stack.insert*$(\mathbb{R})$;
stop $\leftarrow$ *false*;
while *stop* $\neq$ *false* **do**
  $\mathbb{C}^\star \leftarrow$ *checkin_stack.pop*();
  $\mathbb{R}^\star \leftarrow$ *mobility_stack.pop*();
  if *! divide-2-group*$(\mathbb{C}^\star, \mathbb{R}^\star, W, k, l, \tau)$ **then**
    checkin_group.insert$(\mathbb{C}^\star)$;
    mobility_group.insert$(\mathbb{R}^\star)$;
  else
    $\mathbb{C}_1, \mathbb{C}_2, \mathbb{R}_1, \mathbb{R}_2 \leftarrow$ divide-2-group$(\mathbb{C}^\star, \mathbb{R}^\star, W, k, l, \tau)$;
    checkin_stack.insert$(\{\mathbb{C}_1, \mathbb{C}_2\})$;
    mobility_stack.insert$(\{\mathbb{R}_1, \mathbb{R}_2\})$;
  if *checkin_stack* $== \emptyset$ **then**
    stop $\leftarrow$ *true*;
while *checkin_group* $\neq \emptyset$ **do**
  $\mathbb{C}^\star \leftarrow$ *checkin_stack.pop*();
  $\mathbb{R}^\star \leftarrow$ *mobility_stack.pop*();
  $\mathbb{C}^\star \leftarrow k^{\tau,\ l}$-merge$(\mathbb{C}^\star, \mathbb{R}^\star, k, l, \tau)$;
  $\hat{\mathbb{C}}$*.insert*$(\mathbb{C}^\star)$;
Return $\hat{\mathbb{C}}$;

on the definition of *diversity check*, there exist at least one time window $[t, t+\tau]$ that the number of distinct locations is less than l. Since the number of distinct locations increases with number of users monotonically, any subsets of inspected anonymity group will have less than l distinct locations in $[t, t+\tau]$. Therefore, any subsets of inspected anonymity group will not pass the *diversity check*. □

The above proposition guarantees that an anonymity group is *minimum legitimate anonymity group* if it cannot be further divided into two legitimate subsets, since any subsets of anonymity groups that cannot pass *diversity check* will not pass the *diversity check*. Build upon this proposition, we design the *partition-and-group* algorithm with the pseudocode presented in Algorithm 1. The elementary building block is $k^{\tau,\ l}$*-merge* function, which first performs *diversity check* on the given user group and then enumerates through all the users within the group to find optimal mobility records from each user for the check-in to *generalize* with. The *partition-and-group* algorithm first computes the cost matrix W, with $W[i,j]$ filled with the cost of achieving 2-anonymity on the check-ins of user i and j with $k^{\tau,\ l}$*-merge* algorithm. Then, it iteratively partition each anonymity group into two subsets with *divide-2-group* algorithm, and when an anonymity group cannot be divided further it is considered as a final anonymity group. The *divide-2-group* algorithm equally divides the user group into 2 anonymity groups based on their distance to two pivot users that have maximum overall distance to other users. If both groups fail the *diversity check*, the input anonymity group is deemed unable to be further divided. On the other hand, the failed group keeps borrowing one most distant user from the succeed group, until they both pass or fail the *diversity check*. Finally, we apply $k^{\tau,\ l}$*-merge* function on each final anonymity group to ensure all users are protected by $k^{\tau,\ l}$*-anonymity*.

6 EVALUATION

6.1 Performance Comparison

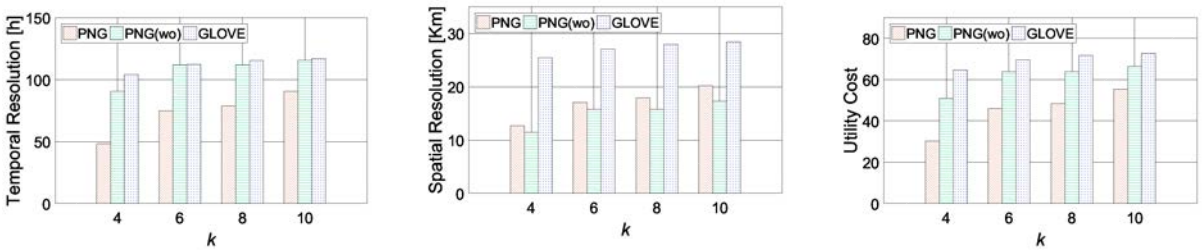
Our solution, denoted by *PNG*, aims to achieve $k^{\tau,\ l}$*-anonymity* to prevent both *re-identification attack* and *probabilistic attack*. In order to show its superiority, we consider two baselines, *i.e.*, *PNG(wo)* and *GLOVE*. *PNG(wo)* is a degraded version of *PNG*, in the condition that only *k-anonymity* is guaranteed to defend *re-identification attack*. On the other hand, *GLOVE* [15] is a state-of-art solution to achieve same privacy guarantee as *PNG(wo)*. To compare the performance of these three solutions, we utilize three metrics of average temporal resolution, average spatial resolution and average utility cost of the sanitized check-ins. Note that *GLOVE* cannot be scalable to large populations due to the high computation complexity. In order to ensure fair comparison, we measure the performance of these three solutions based on two subsets with 5,000 users that are randomly sampled from our two datasets for one-month duration.

We show the performance comparison of these three solutions with different values of k and l(=k/2) in Figure 5 and Figure 6. We can observe that our *PNG* solution outperforms the other two baselines in all privacy settings. With 4-anonymity and 2-diversity on Weibo and WeChat datasets, the average temporal resolutions of sanitized check-ins are 23h and 48h, while the spatial resolutions are 11km and 12km, respectively. Such spatial and temporal resolution is sufficient to accomodate user's need in documenting their daily life. However, the average spatial and temporal resolutions for *PNG(wo)* are much higher, and most of sanitized check-ins from *GLOVE* are too coarse-grained to use with the average temporal resolution reaches as much as 104h. Similar results are observed in average spatial resolution. Furthermore, when it comes to the average utility loss, *PNG* has 24% and 53% improvements in the comparison with *PNG(wo)* and *GLOVE* on WeChat dataset. In addition, *PNG* has 27% and 57% improvements in the comparison with *PNG(wo)* and *GLOVE* on Weibo dataset. In summary, all these results have demonstrated that our proposed *PNG* solution can significantly reduce check-in utility loss even when a stricter privacy criterion $k^{\tau,\ l}$*-anonymity* is met.

6.2 Impact of System Parameters

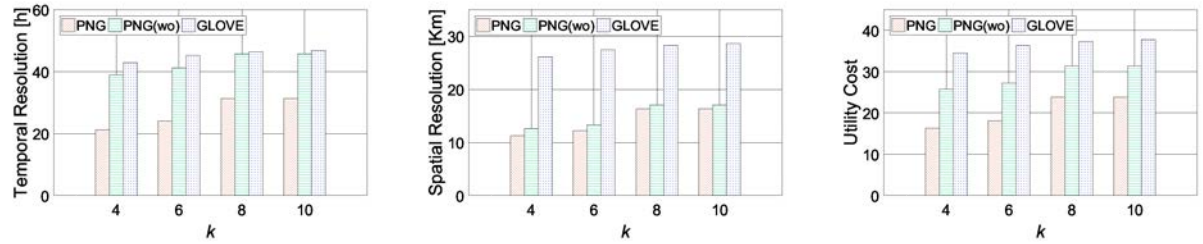
Now we analyze the impact of three key system parameters, *i.e.*, k, l and the amount of available additional mobility records, on the performance of our *PNG* solution.

First, based on both WeChat and Weibo datasets, we measure the performance of *PNG* with different settings of k and l, and show the results in Figure 7 and Figure 8. Take WeChat dataset for example, with a fixed 2-*diversity*, the average temporal resolution, spatial resolution and utility cost increase monotonously as k grows from 2 to 14. However, further increase of k does not result in a significant check-in utility degeneration, suggesting that achieving a stricter privacy guarantee will only cause limited margin check-in utility



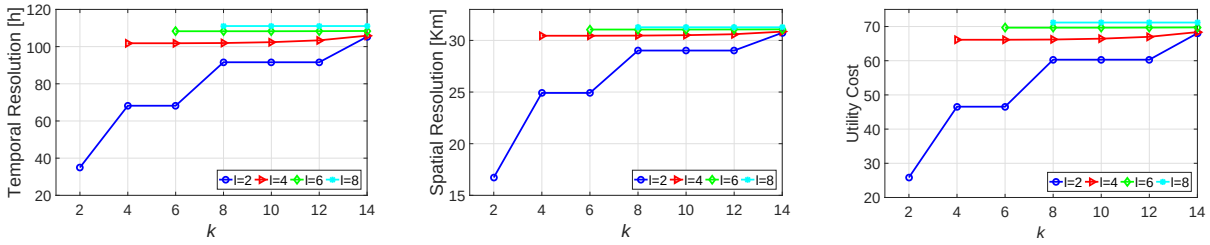
(a) Temporal resolution (b) Spatial resolution (c) Utility cost

Figure 5: The performance comparison between our solution and baseline on WeChat data.



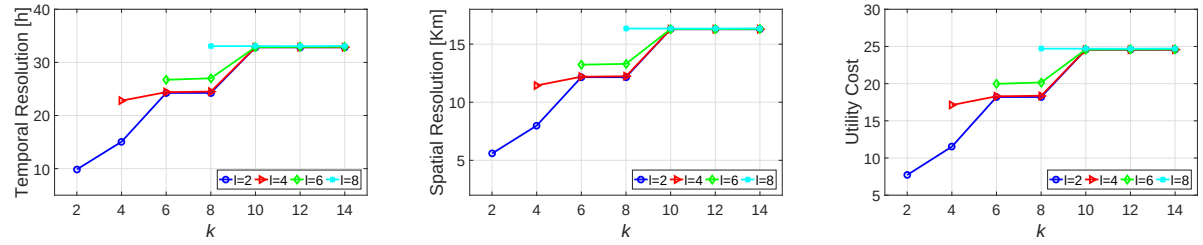
(a) Temporal resolution (b) Spatial resolution (c) Utility cost

Figure 6: The performance comparison between our solution and baseline on Weibo data.



(a) Temporal resolution (b) Spatial resolution (c) Utility cost

Figure 7: The performance of our algorithm under different k and l on WeChat dataset.

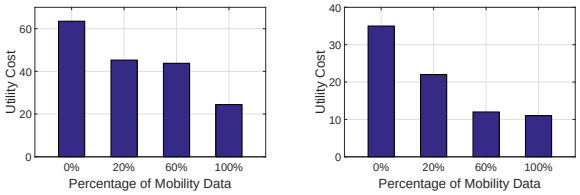


(a) Temporal resolution (b) Spatial resolution (c) Utility cost

Figure 8: The performance of our solution under different k and l on Weibo dataset.

loss. In other words, it indicates our solution can achieve favorable check-in utility when strong privacy protection is needed. As for *probabilistic attack*, a lager l indicates stronger privacy protection. For both WeChat and Weibo datasets, a larger l will also cause additional check-in utility loss. However, the additional utility cost for preventing *probabilistic attack* is much smaller when k is of higher value. It indicates the *PNG* framework provides efficient solution to defend both *re-identification attack* and *probabilistic attack*.

Second, we evaluate the impact of the amount of additional mobility data. Generally speaking, with more complete knowledge about user's mobility behavior, the system is able to better measure the privacy sensitive of each check-in record and derive better privacy solutions. The results of different percentages of additional mobility data are shown in Figure 9. In Figure 9(a), we can observe



(a) WeChat Dataset (b) Weibo Dataset

Figure 9: Impact of the amount of additional mobility data.

that only 20% additional mobility data in WeChat dataset grants the system a 28.6% performance boost in check-in utility. In addition, when more than 60% additional mobility data is provided the performance of system gradually reaches a relative high point, with 30.9% utility improvement compares with no additional mobility data. Similar results are observed on Weibo dataset, which is shown in Figure 9(b). To conclude, the above evaluation verifies our intuition that moderate amount of additional mobility data can lead to significant check-in utility improvement, which showcases the feasibility of our system in real-world scenario.

7 CONCLUSION

In this paper, we investigate the problem of understanding and defending the linkage attacks on check-in services. We design a novel *partition-and-group* framework that integrates the information of check-ins and additional mobility data to provide privacy-preserving and useful check-in service. Evaluation results show that the proposed framework significantly outperforms state-of-art baseline in terms of improving the check-in utility by 24%∼57% and providing stronger privacy guarantee in the same time. We believe our study opens a new angle on measuring and preserving user privacy on check-in services.

ACKNOWLEDGMENT

This work was supported in part by The National Key Research and Development Program of China under grant 2017YFE0112300, the National Nature Science Foundation of China under 61861136003, 61621091 and 61673237, and Beijing National Research Center for Information Science and Technology under 20031887521.

REFERENCES

[1] Gergely Acs and Claude Castelluccia. 2014. A case study: privacy preserving release of spatio-temporal density in paris. (2014), 1679–1688.
[2] Miguel E. Andrs, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geo-indistinguishability:differential privacy for location-based systems. 901–914.
[3] Bhuvan Bamba, Ling Liu, Peter Pesti, and Ting Wang. 2008. Supporting Anonymous Location Queries in Mobile Environments with PrivacyGrid. In *International Conference on World Wide Web*. 237–246.
[4] Igor Bilogrevic, KÃĪvin Huguenin, Stefan Mihaila, Reza Shokri, and Jean Pierre Hubaux. 2015. Predicting Users' Motivations behind Location Check-Ins and Utility Implications of Privacy Protection Mechanisms. *Network and Distributed System Security Symposium* (2015).
[5] Alket Cecaj, Marco Mamei, and Nicola Bicocchi. 2014. Re-identification of anonymized CDR datasets using social network data. In *IEEE International Conference on Pervasive Computing and Communications Workshops*. 237–242.
[6] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. 2013. A Predictive Differentially-Private Mechanism for Mobility Traces. 8555 (2013), 21–41.
[7] Bee-Chung Chen, Daniel Kifer, Kristen LeFevre, Ashwin Machanavajjhala, et al. 2009. Privacy-preserving data publishing. *Foundations and Trends® in Databases* 2, 1–2 (2009), 1–167.
[8] Rui Chen, Benjamin Fung, Bipin C Desai, and Nériah M Sossou. 2012. Differentially private transit data publication: a case study on the montreal transportation system. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 213–221.
[9] Z. Cheng. 2011. Exploring millions of footprints in location sharing services. *Icwsm* (2011).
[10] Chi-Yin Chow and Mohamed F Mokbel. 2011. Trajectory privacy in location-based services and data publication. *ACM Sigkdd Explorations Newsletter* 13, 1 (2011), 19–29.
[11] Yves-Alexandre De Montjoye, César A Hidalgo, Michel Verleysen, and Vincent D Blondel. 2013. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports* 3 (2013), 1376.
[12] Yves-Alexandre De Montjoye, Laura Radaelli, Vivek Kumar Singh, et al. 2015. Unique in the shopping mall: On the reidentifiability of credit card metadata. *Science* 347, 6221 (2015), 536–539.
[13] Cynthia Dwork. 2006. Differential Privacy. In *International Colloquium on Automata, Languages, and Programming*. 1–12.
[14] Chen Gao, Chao Huang, Yue Yu, Huandong Wang, Yong Li, and Depeng Jin. 2019. Privacy-preserving Cross-domain Location Recommendation. In *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*.
[15] Marco Gramaglia and Marco Fiore. 2015. Hiding mobile traffic fingerprints with GLOVE. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*. ACM, 26.
[16] Marco Gramaglia, Marco Fiore, Alberto Tarable, and Albert Banchs. 2017. Preserving Mobile Subscriber Privacy in Open Datasets of Spatiotemporal Trajectories. *IEEE INFOCOM* (2017).
[17] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. 2007. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*. IEEE, 106–115.
[18] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. 2007. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1, 1 (2007), 3.
[19] Cameron Marlow, Cameron Marlow, and Cameron Marlow. 2010. Find me if you can: improving geographical prediction with social and spatial proximity. In *International Conference on World Wide Web*. 61–70.
[20] Anna Monreale, Gennady L Andrienko, Natalia V Andrienko, Fosca Giannotti, Dino Pedreschi, Salvatore Rinzivillo, and Stefan Wrobel. 2010. Movement Data Anonymity through Generalization. *Trans. Data Privacy* 3, 2 (2010), 91–121.
[21] Mehmet Ercan Nergiz, Maurizio Atzori, and Yucel Saygin. 2008. Towards trajectory anonymization: a generalization-based approach. In *Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS*. ACM, 52–61.
[22] Paul Ohm. 2009. Broken promises of privacy: Responding to the surprising failure of anonymization. *Ucla L. Rev.* 57 (2009), 1701.
[23] Sameer Patil, Greg Norcie, Apu Kapadia, and Adam J. Lee. 2012. Reasons, rewards, regrets:privacy considerations in location sharing as an interactive practice. 1–15.
[24] Vincent Primault, Sonia Ben Mokhtar, Cédric Lauradoux, and Lionel Brunie. 2015. Time distortion anonymization for the publication of mobility data with high utility. In *Trustcom/BigDataSE/ISPA, 2015 IEEE*, Vol. 1. IEEE, 539–546.
[25] Jessica Su, Ansh Shukla, Sharad Goel, and Arvind Narayanan. 2017. De-anonymizing Web Browsing Data with Social Networks. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1261–1269.
[26] Latanya Sweeney. 2002. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 05 (2002), 557–570.
[27] Zhen Tu, Runtong Li, Yong Li, Gang Wang, Di Wu, Pan Hui, Li Su, and Depeng Jin. 2018. Your Apps Give You Away: Distinguishing Mobile Users by Their App Usage Fingerprints. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 3 (2018), 138.
[28] Zhen Tu, Fengli Xu, Yong Li, Pengyu Zhang, and Depeng Jin. 2018. A New Privacy Breach: User Trajectory Recovery From Aggregated Mobility Data. *IEEE/ACM Transactions on Networking* 26, 3 (2018), 1446–1459.
[29] Zhen Tu, Kai Zhao, Fengli Xu, Yong Li, Li Su, and Depeng Jin. 2017. Beyond k-anonymity: protect your trajectory from semantic attack. In *2017 14th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 1–9.
[30] Zhen Tu, Kai Zhao, Fengli Xu, Yong Li, Li Su, and Depeng Jin. 2018. Protecting Trajectory from Semantic Attack Considering k-Anonymity, l-diversity and t-closeness. *IEEE Transactions on Network and Service Management* (2018).
[31] Huandong Wang, Chen Gao, Yong Li, Gang Wang, Depeng Jin, and Jingbo Sun. 2018. De-anonymization of Mobility Trajectories: Dissecting the Gaps between Theory and Practice. In *Network and Distributed System Security Symposium*.
[32] Pascal Welke, Ionut Andone, Konrad Blaszkiewicz, and Alexander Markowetz. 2016. Differentiating smartphone users by app usage. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 519–523.
[33] Fengli Xu, Zhen Tu, Yong Li, Pengyu Zhang, Xiaoming Fu, and Depeng Jin. 2017. Trajectory Recovery From Ash: User Privacy Is NOT Preserved in Aggregated

Mobility Data. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1241–1250.

[34] Toby Xu and Ying Cai. 2007. Location anonymity in continuous location-based services. In *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*. ACM, 39.

[35] Hui Zang and Jean Bolot. 2011. Anonymization of location data does not work: A large-scale measurement study. In *Proceedings of the 17th annual international conference on Mobile computing and networking*. ACM, 145–156.

Open-world Learning and Application to Product Classification

Hu Xu
University of Illinois at Chicago, Chicago, IL, USA
hxu48@uic.edu

Bing Liu
University of Illinois at Chicago, Chicago, IL, USA
liub@uic.edu

Lei Shu
University of Illinois at Chicago, Chicago, IL, USA
lshu3@uic.edu

P. Yu
Tsinghua University, Beijing, China
psyu@tsinghua.edu.cn

ABSTRACT

Classic supervised learning makes the *closed-world assumption* that the classes seen in testing must have appeared in training. However, this assumption is often violated in real-world applications. For example, in a social media site, new topics emerge constantly and in e-commerce, new categories of products appear daily. A model that cannot detect new/unseen topics or products is hard to function well in such open environments. A desirable model working in such environments must be able to (1) reject examples from unseen classes (not appeared in training) and (2) incrementally learn the new/unseen classes to expand the existing model. This is called *open-world learning* (OWL). This paper proposes a new OWL method based on meta-learning. The key novelty is that the model maintains only a dynamic set of seen classes that allows new classes to be added or deleted with no need for model re-training. Each class is represented by a small set of training examples. In testing, the meta-classifier only uses the examples of the maintained seen classes (including the newly added classes) on-the-fly for classification and rejection. Experimental results with e-commerce product classification show that the proposed method is highly effective[1].

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by classification**; **Anomaly detection**; **Lifelong machine learning**; **Neural networks**; • **Information systems** → *Nearest-neighbor search.*

KEYWORDS

Open-world Learning; Product Classification

ACM Reference Format:
Hu Xu, Bing Liu, Lei Shu, and P. Yu. 2019. Open-world Learning and Application to Product Classification. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313644

[1]The data and code are available at https://www.cs.uic.edu/~hxu/.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313644

1 INTRODUCTION

An AI agent working in the real world must be able to recognize the classes of things that it has seen/learned before and detect new things that it has not seen and learn to accommodate the new things. This learning paradigm is called *open-world learning* (OWL) [2, 7, 9]. This is in contrast with the classic supervised learning paradigm which makes the *closed-world assumption* that the classes seen in testing must have appeared in training. With the ever-changing Web, the popularity of AI agents such as intelligent assistants and self-driving cars that need to face the real-world open environment with unknowns, OWL capability is crucial.

For example, with the growing number of products sold on Amazon from various sellers, it is necessary to have an open-world model that can automatically classify a product based on a set S of product categories. An emerging product not belonging to any existing category in S should be classified as "unseen" rather than one from S. Further, this unseen set may keep growing. When the number of products belonging to a new category is large enough, it should be added to S. An open-world model should easily accommodate this addition with a low cost of training since it is impractical to retrain the model from scratch every time a new class is added. As another example, the very first interface for many intelligent personal assistants (IPA) (such as Amazon Alexa, Google Assistant, and Microsoft Cortana) is to classify user utterances into existing known domain/intent classes (e.g., Alexa's skills) and also reject/detect utterances from unknown domain/intent classes (that are currently not supported). But, with the support to allow the 3rd-party to develop new skills (Apps), such IPAs must recognize new/unseen domain or intent classes and include them in the classification model. These real-life examples present a major challenge to the maintenance of the deployed model.

Most existing solutions to OWL are built on top of closed-world models [2, 3, 9, 30], e.g., by setting thresholds on the logits (before the softmax/sigmoid functions) to reject unseen classes which tend to mix with existing seen classes. One major weakness of these models is that they cannot easily add new/unseen classes to the existing model without re-training or incremental training (e.g., OSDN [3] and DOC [30]). There are incremental learning techniques (e.g., iCaRL [24] and DEN [21]) that can incrementally learn to classify new classes. However, they miss the capability of rejecting examples from unseen classes. This paper proposes to solve OWL with both capabilities in a very different way via meta-learning.

Problem Statement: At any point in time, the learning system is aware of a set of seen classes $S = \{c_1, \ldots, c_m\}$ and has an OWL model/classifier for S but is unaware of a set of unseen classes

$U = \{c_{m+1}, \dots\}$ (any class not in S can be in U) that the model may encounter. The goal of an OWL model is two-fold: (1) classifying examples from classes in S and reject examples from classes in U, and (2) when a new class c_{m+1} (without loss of generality) is removed from U (now $U = \{c_{m+2}, \dots\}$) and added to S (now $S = \{c_1, \dots, c_m, c_{m+1}\}$, still being able to perform (1) without re-training the model.

Two main challenges for solving this problem are: (1) how to enable the model to classify examples of seen classes into their respective classes and also detect/reject examples of unseen classes, and (2) how to incrementally include the new/unseen classes when they have enough data without re-training the model.

As discussed above, existing methods either focus on the challenge (1) or (2), but not both. To tackle both challenges in an unified approach, this paper proposes an entirely new OWL method based on meta-learning [1, 10–12, 32]. The method is called *Learning to Accept Classes* (L2AC). The key novelty of L2AC is that the model maintains a dynamic set S of seen classes that allow new classes to be added or deleted with no model re-training needed. Each class is represented by a small set of training examples. In testing, the meta-classifier only uses the examples of the maintained seen classes (including the newly added classes) on-the-fly for classification and rejection. That is, the learned meta-classifier classifies or rejects a test example by comparing it with its nearest examples from each seen class in S. Based on the comparison results, it determines whether the test example belongs to a seen class or not. If the test example is not classified as any seen class in S, it is rejected as unseen. Unlike existing OWL models, the parameters of the meta-classifier are not trained on the set of seen classes but on a large number of other classes which can share a large number of features with seen and unseen classes, and thus can work with any seen classification and unseen class rejection without re-training.

We can see that the proposed method works like a nearest neighbor classifier (e.g., kNN). However, the key difference is that we train a meta-classifier to perform both classification and rejection based on a learned metric and a learned voting mechanism. Also, kNN cannot do rejection on unseen classes.

The main contributions of this paper are as follows.

(1) It proposes a novel approach (called L2AC) to OWL based on meta-learning, which is very different from existing approaches.

(2) The key advantage of L2AC is that with the meta-classifier, OWL becomes simply maintaining the seen class set S because both seen class example classification and unseen class example rejection/detection are based on comparing the test example with the examples of each class in S. To be able to accept/classify any new class, we only need to put the class and its examples in S.

The proposed approach has been evaluated on product classification and the results show its competitive performance.

2 L2AC FRAMEWORK

As an overview, Fig. 1 depicts how L2AC classifies a test example into an existing seen class or rejects it as from an unseen class. The training process for the meta-classifier is not shown, which is detailed in Sec. 2.2. The L2AC framework has two major components: a ranker and a meta-classifier. The ranker is used to retrieve some examples from a seen class that are similar/near to the test example. The meta-classifier performs classification after it reads the retrieved examples from the seen classes. The two components work together as follows.

Assume we have a set of seen classes S. Given a test example x_t that may come from either a seen class or an unseen class, the ranker finds a list of top-k nearest examples to x_t from each seen class $c \in S$, denoted as $x_{a_{1:k}|c}$. The meta-classifier produces the probability $p(c = 1|x_t, x_{a_{1:k}|x_t,c})$ that the test x_t belongs to the seen class c based on c's top-k examples (most similar to x_t). If none of these probabilities from the seen classes in S exceeds a threshold (e.g., 0.5 for the sigmoid function), L2AC decides that x_t is from an unseen class (rejection); otherwise, it predicts x_t as from the seen class with the highest probability (for classification). We denote $p(c = 1|x_t, x_{a_{1:k}|x_t,c})$ as $p(c|x_t, x_{a_{1:k}})$ for brevity when necessary. Note that although we use a threshold, this is a general threshold that is not for any specific classes as in other OWL approaches but only for the meta-classifier. More practically, this threshold is pre-determined (not empirically tuned via experiments on hyper-parameter search) and the meta-classifier is trained based on this fixed threshold.

As we can see, the proposed framework works like a supervised lazy learning model, such as the k-nearest neighbor (kNN) classifier. Such a lazy learning mechanism allows the dynamic maintenance of a set of seen classes, where an unseen class can be easily added to the seen class set S. However, the key differences are that all the metric space, voting and rejection are learned by the meta-classifier.

Retrieving the top-k nearest examples $x_{a_{1:k}}$ for a given test example x_t needs a ranking model (the ranker). We will detail a sample implementation of the ranker in Sec. 3 and discuss the details of the meta-classifier in the next section.

2.1 Meta-Classifier

Meta-classifier serves as the core component of the L2AC framework. It is essentially a binary classifier on a given seen class. It takes the top-k nearest examples (to the test example x_t) of the seen class as the input and determines whether x_t belongs to that seen class or not. In this section, we first describe how to represent examples of a seen class. Then we describe how the meta-classifier processes these examples together with the test example into an overall probability score (via a voting mechanism) for deciding whether the test example should belong to any seen class (classification) or not (rejection). Along with that we also describe how a joint decision is made for open-world classification over a set of seen classes. Finally, we describe how to train the meta-classifier via another set of meta-training classes and their examples.

2.1.1 Example Representation and Memory. Representation learning lives at the heart of neural networks. Following the success of using pre-trained weights from large-scale image datasets (such as ImageNet [25]) as feature encoders, we assume there is an encoder that captures almost all features for text classification.

Given an example x representing a text document (a sequence of tokens), we obtain its continuous representation (a vector) via an encoder $h = g(x)$, where the encoder $g(\cdot)$ is typically a neural network (e.g., CNN or LSTM). We will detail a simple encoder implementation in Sec. 3.

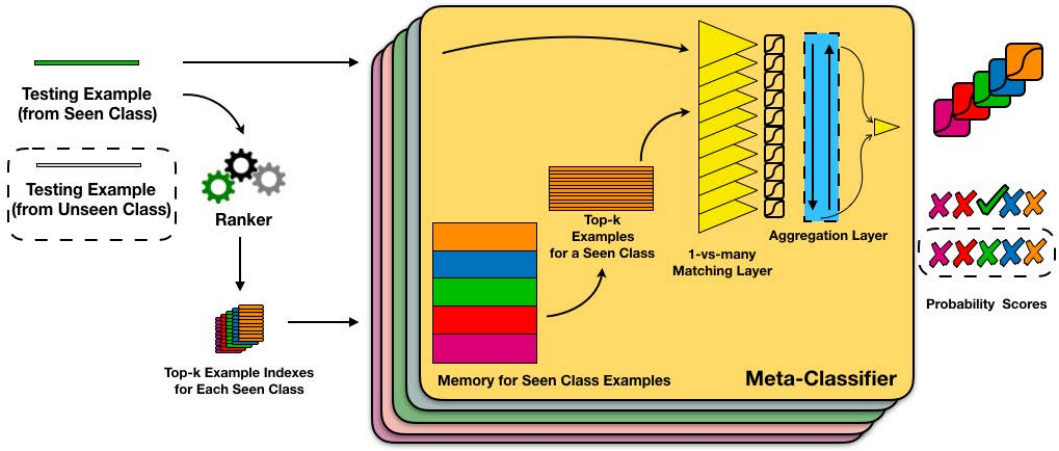


Figure 1: Overview of the L2AC framework (best viewed in colors). Assume the seen class set S has 5 classes and their examples are indicated by 5 different colors. L2AC has two components: a ranker and a meta-classifier. Given a (green) testing example from a seen class, the ranker first retrieves the top-k nearest examples (memory indexes) from each seen class. Then the meta-classifier takes both the test example and the top-k nearest examples for a seen class to produce a probability score for that class. The meta-classifier is applied 5 times (indicated by 5 rounded rectangles) over these 5 seen classes and yields 5 probability scores, where the 3rd (green) class attends the maximum score as the final class (green) prediction. However, if the test example (grey) is from an unseen class (as indicated by the dashed box), none of those probability scores from the seen classes will predict positive, which leads rejection.

Further, we save the continuous representations of the examples into the memory of the meta-classifier. So later, the top-k examples can be efficiently retrieved via the index (address) in the memory. The memory is essentially a matrix $E \in \mathbb{R}^{n \times |h|}$, where n is the number of all examples from seen classes and $|h|$ is the size of the hidden dimensions. Note that we will still use x instead of h to refer to an example for brevity. Given the test example x_t, the meta-classifier first looks up the actual continuous representations $x_{a_{1:k}}$ of the top-k examples for a seen class. Then the meta-classifier computes the similarity score between x_t and each x_{a_i} $(1 \le i \le k)$ individually via a 1-vs-many matching layer as described next.

2.1.2 1-vs-many Matching Layer. To compute the overall probability between a test example and a seen class, a 1-vs-many matching layer in the meta-classifier first computes the individual similarity score between the test example and each of the top-k retrieved examples of the seen class. The 1-vs-many matching layer essentially consists of k shared matching networks as indicated by big yellow triangles in Fig. 1. We denote each matching network as $f(\cdot,\cdot)$ and compute similarity scores $r_{1:k}$ for all top-k examples $r_{1:k} = f(x_t, x_{a_{1:k}})$.

The matching network first transforms the test example x_t and x_{a_i} from the continuous representation space to a single example in a similarity space. We leverage two similarity functions to obtain the similarity space. The first function is the absolute values of the element-wise subtraction: $f_{\text{abssub}}(x_t, x_{a_i}) = |x_t - x_{a_i}|$. The second one is the element-wise summation: $f_{\text{sum}}(x_t, x_{a_i}) = x_t + x_{a_i}$. Then the final similarity space is the concatenation of these two functions' results: $f_{\text{sim}}(x_t, x_{a_i}) = f_{\text{abssub}}(x_t, x_{a_i}) \oplus f_{\text{sum}}(x_t, x_{a_i})$, where $\oplus$ denotes the concatenation operation. We then pass the result to two fully-connected layers (one with Relu activation) and a sigmoid function:

$$r_i = f(x_t, x_{a_i}) = \sigma\Big(W_2 \cdot \text{Relu}\big(W_1 \cdot f_{\text{sim}}(x_t, x_{a_i}) + b_1\big) + b_2\Big). \quad (1)$$

Since there are k nearest examples, we have k similarity scores denoted as $r_{1:k}$. The hyper-parameters are detailed in Sec. 3.

2.1.3 Open-world Learning via Aggregation Layer. After getting the individual similarity scores, an aggregation layer in the meta-classifier merges the k similarity scores into a single probability indicating whether the test example x_t belongs to the seen class. By having the aggregation layer, the meta-classifier essentially has a *parametric voting mechanism* so that it can learn how to vote on multiple nearest examples (rather than a single example) from a seen class to decide the probability. As a result, the meta-classifier can have more reliable predictions, which is studied in Sec. 3.

We adopt a (many-to-one) BiLSTM [15, 27] as the aggregation layer. We set the output size of BiLSTM to 2 (1 per direction of LSTM). Then the output of BiLSTM is connected to a fully-connected layer followed by a sigmoid function that outputs the probability. The computation of the meta-classifier for a given test example x_t and $x_{a_{1:k}}$ for a seen class c can be summarized as:

$$p(c|x_t, x_{a_{1:k}}) = \sigma\big(W \cdot \text{BiLSTM}(r_{1:k}) + b\big). \quad (2)$$

Inspired by DOC [30], for each class $c \in S$, we evaluate Eq. 2 as:

$$\hat{y} = \begin{cases} \textit{reject}, \text{ if } \max_{c \in S} p(c|x_t, x_{a_{1:k}}) \le 0.5; \\ \\ \arg\max_{c \in S} p(c|x_t, x_{a_{1:k}}), \text{ otherwise.} \end{cases} \quad (3)$$

If none of existing seen classes S gives a probability above 0.5, we *reject* x_t as an example from some unseen class. Note that given a large number of classes, eq. 3 can be efficiently implemented in parallel. We leave this to future work. To make L2AC an easily

accessible approach, we use 0.5 as the threshold naturally and do not introduce an extra hyper-parameter that needs to be artificially tuned. Note also that as discussed earlier, the seen class set S and its examples can be dynamically maintained (e.g., one can add to or remove from S any class). So the meta-classifier simply performs open-world classification over the current seen class set S.

2.2 Training of Meta-Classifier

Since the meta-classifier is a general classifier that is supposed to work for any class, training the meta-classifier $p_\theta(c|x_t, x_{a_{1:k}|x_t}, c)$ requires examples from another set M of classes called *meta-training classes*. A large $|M|$ is desirable so that meta-training classes have good coverage of features for seen and unseen classes in testing, which is in similar spirit to few-shot learning [19]. We also enforce $(S \cup U) \cap M = \varnothing$ in Sec. 3, so that all seen and unseen classes are totally unknown to the meta-classifier.

Next, we formulate the meta-training examples from M, which consist of a set of pairs (with positive and negative labels). The first component of a pair is a training document x_q from a class in M, and the second component is a sequence of top-k nearest examples also from a class in M.

We assume every example (document) of a class in M can be a training document x_q. Assuming x_q is from class $c \in M$, a positive training pair is $(x_q, x_{a_{1:k}|x_q,c})$, where $x_{a_{1:k}|x_q,c}$ are top-k examples from class c that are most similar or nearest to x_q; a negative training pair is $(x_q, x_{a_{1:k}|x_q,c'})$, where $c' \in M$, $c \neq c'$ and $x_{a_{1:k}|x_q,c'}$ are top-k examples from class c' that are nearest to x_q. We call c' one *negative class* for x_q. Since there are many negative classes $c' \in M \backslash c$ for x_q, we keep top-n negative classes for each training example x_q. That is, each x_q has one positive training pair and n negative training pairs. To balance the classes in the training loss, we give a weight ratio $n : 1$ for a positive and a negative pair, respectively.

Training the meta-classifier also requires validation classes for model selection (during optimization) and hyper-parameters (k and n) tuning (as detailed in Experiments). Since the classes tested by the meta-classifier are unexpected, we further use a set of *validation classes* $M' \cap M = \varnothing$ (also $M' \cap (S \cup U) = \varnothing$), to ensure generalization on the seen/unseen classes.

3 EXPERIMENTS

We want to address the following Research Questions (RQs): **RQ1** - what is the performance of the meta-classifier with different settings of top-k examples and n negative classes? **RQ2** - How is the performance of L2AC compared with state-of-the-art text classifiers for open-world classification (which all need some forms of re-training).

3.1 Dataset

We leverage the huge amount of product descriptions from the Amazon Datasets [14] and form the OWL task as the following. Amazon.com maintains a tree-structured category system. We consider each path to a leaf node as a class. We removed products belonging to multiple classes to ensure the classes have no overlapping. This gives us 2598 classes, where 1018 classes have more than 400 products per class. We randomly choose 1000 classes from the 1018 classes with 400 randomly selected products per class as the *encoder training set*; 100 classes with 150 products per class are used as the (classification) *test set*, including both seen classes S and unseen classes U; another 1000 classes with 100 products per class are used as the *meta-training set* (including both M and M'). For the 100 classes of the test set, we further hold out 50 examples (products) from each class as test examples. The rest 100 examples are training data for baselines, or seen classes examples to be read by the meta-classifier (which only reads those examples but is not trained on those examples). To train the meta-classifier, we further split the meta-training set as 900 *meta-training classes* (M) and 100 *validation classes* (M').

For all datasets, we use NLTK[2] as the tokenizer, and regard all words that appear more than once as the vocabulary. This gives us 17,526 unique words. We take the maximum length of each document as 120 since the majority of product descriptions are under 100 words.

3.2 Ranker

We use cosine similarity to rank the examples in each seen (or meta-training) class for a given test (or meta-training) example x_t (or x_q)[3]. We apply cosine directly on the hidden representations of the encoder as $cosine(h_*, h_{a_i}) = \frac{h_* \cdot h_{a_i}}{|h_*|_2 |h_{a_i}|_2}$, where $*$ can be either t or q, $|\cdot|_2$ denotes the l-2 norm and $\cdot$ denotes the dot product of two examples.

Training the meta-classifier also requires a ranking of negative classes for a meta-training example x_q, as discussed in Sec. 2.2. We first compute a *class vector* for each meta-training class. This class vector is averaged over all encoded representations of examples of that class. Then we rank classes by computing cosine similarity between the class vectors and the meta-training example x_q. The top-n (defined in the previous section) classes are selected as negative classes for x_q. We explore different settings of n later.

3.3 Evaluation

Similar to [30], we choose 25, 50, and 75 classes from the (classification) test set of 100 classes as the seen classes for three (3) experiments. Note that each class in the test set has 150 examples, where 100 examples are for the training of baseline methods or used as seen class examples for L2AC and 50 examples are for testing both the baselines and L2AC. We evaluate the results on all 100 classes for those three (3) experiments. For example, when there are 25 seen classes, testing examples from the rest 75 unseen classes are taken as from one *rejection class* c_{rej}, as in [30].

Besides using macro F1 as used in [30], we also use weighted F1 score overall classes (including seen and the rejection class) as the evaluation metric. Weighted F1 is computed as

$$\sum_{c \in S \cup \{c_{\text{rej}}\}} \frac{N_c}{\sum_{c \in S \cup \{c_{\text{rej}}\}} N_c} \cdot \text{F1}_c, \tag{4}$$

where N_c is the number of examples for class c and F1_c is the F1 score of that class. We use this metric because macro F1 has a bias on the importance of rejection when the seen class set is small

[2] https://www.nltk.org/

[3] Given many examples to process, the ranker can be implemented in a fully parallel fashion to speed up the processing, which we leave to future work as it is beyond the scope of this work.

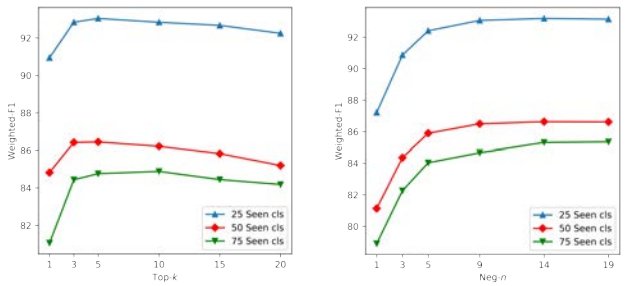


Figure 2: Weighted F1 scores for different k's ($n = 9$) and different n's ($k = 5$).

(macro F1 treats the rejection class as equally important as one seen class). For example, when the number of seen classes is small, the rejection class should have a higher weight as a classifier on a small seen set is more likely challenged by examples from unseen classes. Further, to stabilize the results, we train all models with 10 different initializations and average the results.

3.4 Hyper-parameters

For simplicity, we leverage a BiLSTM [15, 27] on top of a GloVe [23] embedding (840b.300d) layer as the encoder (other choices are also possible). Similar to feature encoders trained from ImageNet [25], we train classification over the encoder training set with 1000 classes and use 5% of the encoding training data as encoder validation data. We apply dropout rates of 0.5 to all layers of the encoder. The classification accuracy of the encoder on validation data is **81.76%**. The matching network (the shared network within the 1-vs-many matching layer) has two fully-connected layers, where the size of the hidden dimension is 512 with a dropout rate of 0.5. We set the batch size of meta-training as 256.

To answer RQ1 on two hyper-parameters k (number of nearest examples from each class) and n (number of negative classes), we use the 100 validation classes to determine these two hyper-parameters. We formulate the validation data similar to the testing experiment on 50 seen classes. For each validation class, we select 50 examples for validation. The rest 50 examples from each validation seen class are used to find top-k nearest examples. We perform grid search of averaged weighted F1 over 10 runs for $k \in \{1, 3, 5, 10, 15, 20\}$ and $n \in \{1, 3, 5, 9\}$, where $k = 5$ and $n = 9$ reach a reasonably well weighted F1 (87.60%). Further increasing n gives limited improvements (e.g., 87.69% for $n = 14$ and 87.68% for $n = 19$, when $k = 5$). But a large n significantly increases the number of training examples (e.g., $n = 14$ ended with more than 1 million meta-training examples) and thus training time. So we decide to select $k = 5$ and $n = 9$ for all ablation studies below. Note the validation classes are also used to compute (formulated in a way similar to the meta-training classes) the validation loss for selecting the best model during Adam [16] optimization.

3.5 Compared Methods

To the best of our knowledge, DOC [30] is the only state-of-the-art baseline for open-world learning (with rejection) for text classification. It has been shown in [30] that DOC significantly outperforms the methods CL-cbsSVM and cbsSVM in [9] and OpenMax in [3]. OpenMax is a state-of-the-art method for image classification with rejection capability.

To answer RQ2, we use DOC and its variants to show that the proposed method has comparable performance with the best open-world learning method with re-training. Note that DOC cannot incrementally add new classes. So we re-train DOC over different sets of seen classes from scratch every time new classes are added to that set. It is thus actually unfair to compare our method with DOC because DOC is trained on the actual training examples of all classes. However, our method still performs better in general. We used the original code of DOC and created six (6) variants of it.

DOC-CNN: CNN implementation as in the original DOC paper without Gaussian fitting (using 0.5 as the threshold for rejection). It operates directly on a sequence of tokens.
DOC-LSTM: a variant of DOC-CNN, where we replace CNN with BiLSTM to encode the input sequence for fair comparison. BiLSTM is trainable and the input is still a sequence of tokens.
DOC-Enc: this is adapted from DOC-CNN, where we remove the feature learning part of DOC-CNN and feed the hidden representation from our encoder directly to the fully-connected layers of DOC for a fair comparison with L2AC.
DOC-*-Gaus: applying Gaussian fitting proposed in [30] on the above three baselines, we have 3 more DOC baselines. Note that these 3 baselines have exactly the same models as above respectively. They only differ in the thresholds used for rejection. Gaussian fitting in [30] is used to set a good threshold for rejection. We use these baselines to show that the Gaussian fitted threshold improves the rejection performance of DOC significantly but may lower the performance of seen class classification. The original DOC is **DOC-CNN-Gaus** here.

The following baselines are variants of L2AC.
L2AC-n9-NoVote: this is a variant of the proposed L2AC that only takes one most similar example (from each class), i.e., $k = 1$, with one positive class paired with $n = 9$ negative classes in meta-training ($n = 9$ has the best performance as indicated in answering RQ1 above). We use this baseline to show that the performance of taking only one sample may not be good enough. This baseline clearly does not have/need the aggregation layer and only has a single matching network in the 1-vs-many layer.
L2AC-n9-Vote3: this baseline uses exactly the same model as L2AC-n9-NoVote. But during evaluation, we allow a non-parametric voting process (like kNN) for prediction. We report the results of voting over top-3 examples per seen class as it has the best result (ranging from 3 to 10). If the average of the top-3 similar examples in a seen class has example scores with more than 0.5, L2AC believes the testing example belongs to that class. We use this baseline to show that the aggregation layer is effective in learning to vote and L2AC can use more similar examples and get better performance.
L2AC-k5-n9-AbsSub/Sum: To show that using two similarity functions ($f_{\text{abssub}}(\cdot,\cdot)$ and $f_{\text{sum}}(\cdot,\cdot)$) gives better results, we further perform ablation study by using only one of those similarity functions at a time, which gives us two baselines.
L2AC-k5-n9/14/19: this baseline has the best $k = 5$ and $n = 9$ on the validation classes, as indicated in the previous subsection. Interestingly, further increasing k may reduce the performance as

Table 1: Weighted F1 (WF1) and macro F1 (MF1) scores on a test set with 100 classes with 3 settings: 25, 50, and 75 seen classes. The set of seen classes are incrementally expanded from 25 to 75 classes (or gradually shrunk from 75 to 25 classes). The results are the averages over 10 runs with standard deviations in parenthesis.

Methods	$\|S\| = 25$ (WF1)	$\|S\| = 25$ (MF1)	$\|S\| = 50$ (WF1)	$\|S\| = 50$ (MF1)	$\|S\| = 75$ (WF1)	$\|S\| = 75$ (MF1)
DOC-CNN	53.25(1.0)	55.04(0.39)	70.57(0.46)	76.91(0.27)	81.16(0.47)	86.96(0.2)
DOC-LSTM	57.87(1.26)	57.6(1.18)	69.49(1.58)	75.68(0.78)	77.74(0.48)	84.48(0.33)
DOC-Enc	82.92(0.37)	75.09(0.33)	82.53(0.25)	84.34(0.23)	83.84(0.36)	88.33(0.19)
DOC-CNN-Gaus	85.72(0.43)	76.79(0.41)	83.33(0.31)	83.75(0.26)	84.21(0.12)	87.86(0.21)
DOC-LSTM-Gaus	80.31(1.73)	70.49(1.55)	77.49(0.74)	79.45(0.59)	80.65(0.51)	85.46(0.25)
DOC-Enc-Gaus	88.54(0.22)	80.77(0.22)	84.75(0.21)	85.26(0.2)	83.85(0.37)	87.92(0.22)
L2AC-n9-NoVote	91.1(0.17)	82.51(0.39)	84.91(0.16)	83.71(0.29)	81.41(0.54)	85.03(0.62)
L2AC-n9-Vote3	91.54(0.55)	82.42(1.29)	84.57(0.61)	82.7(0.95)	80.18(1.03)	83.52(1.14)
L2AC-k5-n9-AbsSub	92.37(0.28)	84.8(0.54)	85.61(0.36)	84.54(0.42)	83.18(0.38)	86.38(0.36)
L2AC-k5-n9-Sum	83.95(0.52)	70.85(0.91)	76.09(0.36)	75.25(0.42)	74.12(0.51)	78.75(0.57)
L2AC-k**5**-n**9**	93.07(0.33)	86.48(0.54)	86.5(0.46)	85.99(0.33)	84.68(0.27)	88.05(0.18)
L2AC-k5-n14	**93.19**(0.19)	86.91(0.33)	**86.63**(0.28)	86.42(0.2)	85.32(0.35)	88.72(0.23)
L2AC-k5-n19	93.15(0.24)	86.9(0.45)	86.62(0.49)	86.48(0.43)	**85.36**(0.66)	88.79(0.52)

L2AC may focus on not-so-similar examples. We also report results on $n = 14$ or 19 to show that the results do not get much better.

3.6 Results Analysis

From Table 1, we can see that L2AC outperforms DOC, especially when the number of seen classes is small. First, from Fig. 2 we can see that $k = 5$ and $n = 9$ gets reasonably good results. Increasing k may harm the performance as taking in more examples from a class may let L2AC focus on not-so-similar examples, which is bad for classification. More negative classes give L2AC better performance in general but further increasing n beyond 9 has little impact.

Next, we can see that as we incrementally add more classes, L2AC gradually drops its performance (which is reasonable due to more classes) but it still yields better performance than DOC. Considering that L2AC needs no training with additional classes, while DOC needs full training from scratch, L2AC represents a major advance. Note that testing on 25 seen classes is more about testing a model's rejection capability while testing on 75 seen classes is more about the classification performance of seen class examples. From Table 1, we notice that L2AC can effectively leverage multiple nearest examples and negative classes. In contrast, the non-parametric voting of L2AC-n9-Vote3 over top-3 examples may not improve the performance but introduce higher variances. Our best $k = 5$ indicates that the meta-classifier can dynamically leverage multiple nearest examples instead of solely relying on a single example. As an ablation study on the choices of similarity functions, running L2AC on a single similarity function gives poorer results as indicated by either L2AC-k5-n9-AbsSub or L2AC-k5-n9-Sum.

DOC without encoder (DOC-CNN or DOC-LSTM) performs poorly when the number of seen classes is small. Without Gaussian fitting, DOC's (DOC-CNN, DOC-LSTM or DOC-Enc) performance increases as more classes are added as seen classes. This is reasonable as DOC is more challenged by fewer seen training classes and more unseen classes during testing. As such, Gaussian fitting (DOC-*-Gaus) alleviates the weakness of DOC on a small number of seen training classes.

4 RELATED WORK

Open-world learning has been studied in text mining and computer vision (where it is called open-set recognition) [2, 7, 9]. Most existing approaches focus on building a classifier that can predict examples from unseen classes into a (hidden) *rejection class*. These solutions are built on top of closed-world classification models [2, 3, 30]. Since a closed-world classifier cannot detect/reject examples from unseen classes (they will be classified into some seen classes), some thresholds are used so that these closed-world models can also be used to do rejection. However, as discussed earlier, when incrementally learning new classes, they also need some form of re-training, either full re-training from scratch [3, 30] or partial re-training in an incremental manner [2, 9].

Our work is also related to class incremental learning [21, 24, 26], where new classes can be added dynamically to the classifier. For example, iCaRL [24] maintains some exemplary data for each class and incrementally tunes the classifier to support more new classes. However, they also require training when each new class is added.

Our work is clearly related to meta-learning (or learning to learn) [32], which turns the machine learning tasks themselves as training data to train a meta-model and has been successfully applied to many machine learning tasks lately, such as [1, 8, 10–12]. Our proposed framework focuses on learning the similarity between an example and an arbitrary class and we are not aware of any open-world learning work based on meta-learning.

The proposed framework is also related to zero-shot learning [20, 22, 31] (in that we do not require training but need to read training examples), k-nearest neighbors (kNN) (with additional rejection capability, metric learning [34] and learning to vote), and Siamese networks [4, 17, 33] (regarding processing a pair of examples). However, all those techniques work in closed-worlds with no rejection capability.

Product classification has been studied in [5, 6, 13, 18, 28, 29], mostly in a multi-level (or hierarchical) setting. However, given the dynamic taxonomy in nature, product classification has not been studied as an open-world learning problem.

5 CONCLUSIONS

In this paper, we proposed a meta-learning framework called L2AC for open-world learning. L2AC has been applied to product classification. Compared to traditional closed-world classifiers, our meta-classifier can incrementally accept new classes by simply adding new class examples without re-training. Compared to other open-world learning methods, the rejection capability of L2AC is trained rather than realized using some empirically set thresholds. Our experiments showed superior performances to strong baselines.

ACKNOWLEDGMENTS

Bing Liu's work was partially supported by the National Science Foundation (NSF IIS 1838770) and by a research gift from Huawei.

REFERENCES

[1] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. 2016. Learning to learn by gradient descent by gradient descent. In *NIPS*. 3981–3989.

[2] Abhijit Bendale and Terrance Boult. 2015. Towards open world recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1893–1902.

[3] Abhijit Bendale and Terrance E Boult. 2016. Towards open set deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1563–1572.

[4] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a" siamese" time delay neural network. In *Advances in Neural Information Processing Systems*. 737–744.

[5] Ali Cevahir and Koji Murakami. 2016. Large-scale Multi-class and Hierarchical Product Categorization for an E-commerce Giant. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 525–535.

[6] Jianfu Chen and David Warren. 2013. Cost-sensitive learning for large-scale hierarchical classification. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 1351–1360.

[7] Zhiyuan Chen and Bing Liu. 2018. *Lifelong machine learning*. Morgan & Claypool Publishers.

[8] Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. 2018. Learning to Teach. *arXiv preprint arXiv:1805.03643* (2018).

[9] Geli Fei, Shuai Wang, and Bing Liu. 2016. Learning cumulatively to become more knowledgeable. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1565–1574.

[10] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. 2017. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734* (2017).

[11] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *International Conference on Machine Learning*. 1126–1135.

[12] Chelsea Finn, Kelvin Xu, and Sergey Levine. 2018. Probabilistic Model-Agnostic Meta-Learning. *arXiv preprint arXiv:1806.02817* (2018).

[13] Vivek Gupta, Harish Karnick, Ashendra Bansal, and Pradhuman Jhala. 2016. Product Classification in E-Commerce using Distributional Semantics. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 536–546.

[14] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 507–517.

[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, Vol. 2.

[18] Zornitsa Kozareva. 2015. Everyone likes shopping! multi-class product categorization for e-commerce. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1329–1333.

[19] Brenden Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua Tenenbaum. 2011. One shot learning of simple visual concepts. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, Vol. 33.

[20] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. 2009. Learning to detect unseen object classes by between-class attribute transfer. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 951–958.

[21] Jeongtae Lee, Jaehong Yun, Sungju Hwang, and Eunho Yang. 2017. Lifelong Learning with Dynamically Expandable Networks. *arXiv preprint arXiv:1708.01547* (2017).

[22] Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. 2009. Zero-shot learning with semantic output codes. In *NIPS*. 1410–1418.

[23] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[24] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. 2017. iCaRL: Incremental Classifier and Representation Learning. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 5533–5542.

[25] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, and others. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 115, 3 (2015), 211–252.

[26] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671* (2016).

[27] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[28] Dan Shen, Jean-David Ruvini, and Badrul Sarwar. 2012. Large-scale item categorization for e-commerce. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 595–604.

[29] Dan Shen, Jean David Ruvini, Manas Somaiya, and Neel Sundaresan. 2011. Item categorization in the e-commerce domain. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 1921–1924.

[30] Lei Shu, Hu Xu, and Bing Liu. 2017. DOC: Deep Open Classification of Text Documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, 2911–2916. https://www.aclweb.org/anthology/D17-1314

[31] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In *NIPS*. 935–943.

[32] Sebastian Thrun and Lorien Pratt. 2012. *Learning to learn*. Springer.

[33] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, and others. 2016. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*. 3630–3638.

[34] Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. 2003. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*. 521–528.

Place Deduplication with Embeddings

Carl Yang
University of Illinois, Urbana Champaign, 201 N Goodwin Ave, Urbana, IL 61801, USA
Facebook Inc., 770 Broadway, New York, NY 10003, USA
jiyang3@illinois.edu

Do Huy Hoang
Facebook Inc., 770 Broadway, New York, NY 10003, USA
hoangmit@fb.com

Tomas Mikolov
Facebook Inc., 770 Broadway, New York, NY 10003, USA
tmikolov@fb.com

Jiawei Han
University of Illinois, Urbana Champaign, 201 N Goodwin Ave, Urbana, IL 61801, USA
hanj@illinois.edu

ABSTRACT

Thanks to the advancing mobile location services, people nowadays can post about places to share visiting experience on-the-go. A large place graph not only helps users explore interesting destinations, but also provides opportunities for understanding and modeling the real world. To improve coverage and flexibility of the place graph, many platforms import places data from multiple sources, which unfortunately leads to the emergence of numerous duplicated places that severely hinder subsequent location-related services. In this work, we take the anonymous place graph from Facebook as an example to systematically study the problem of place deduplication: We carefully formulate the problem, study its connections to various related tasks that lead to several promising basic models, and arrive at a systematic two-step data-driven pipeline based on place embedding with multiple novel techniques that works significantly better than the state-of-the-art.

KEYWORDS

place deduplication; feature generation; metric learning

ACM Reference Format:
Carl Yang, Do Huy Hoang, Tomas Mikolov, Jiawei Han. 2019. Place Deduplication with Embeddings. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13-17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313456

1 INTRODUCTION

With the prevalence of mobile devices and online social platforms, maps and location related apps have become an integral part of people's everyday life. It has encouraged the development of various location-oriented infrastructures and services [13, 23–25, 36, 52, 53]. For most of them, one key task is the maintenance of a high-quality map database (*a.k.a. place graph*), which consists of various real-world places and their attributes. On one hand, such a database can support various map-based activities like *search*, *like*, *check-in* and *post*. Improving the efficiency and quality of such activities is beneficial to various stakeholders including business owners, advertisement agencies, and common users. On the other hand, the frequent location-based activity data generated by users provide an invaluable opportunity to improve the representation and understanding of the real world, and may further shed light on the capturing and modeling of various real-world activities.

However, since place pages are usually created from multiple sources whose qualities are hard to verify upon creation, the place graphs are often subject to large amounts of duplications, which hurts the quality of subsequent location-based services. For example, when a user touring in New York City wants to make a post at Times Square, if the system returns a list of places like "New York Time Square" "Manhattan, Time Square" "TimeSquare, NYC", the user will get confused about which one to post on. Worse still, if the user also wants to explore restaurants or shopping malls around Times Square, then she/he may need to manually combine multiple lists of stores to get a full picture of all available choices.

Such user experiences are mainly due to the problem of duplicated places. To ease users' exploration of places and sharing of experiences, in this work, we take the *massive* anonymous place graph from Facebook as an example to systematically study the problem of *place deduplication*, and comprehensively evaluate the performance of different methods through extensive experiments on *duplication candidate fetch* and *pair-wise duplication prediction*.

Specifically, we start with *large-scale* unsupervised feature generation to encode various ad-hoc place attributes through *embedding learning* and *network-based embedding smoothing*. Upon that, we develop an effective supervised metric learning framework to find the most useful features indicating place duplications. To fully leverage the noisy labeled data, we novelly pack the model with a series of techniques including *batch-wise hard sampling*, *source-oriented attentive training*, and *soft clustering-based label denoising*. Extensive experiments on real-world place data show that our pipeline can outperform state-of-the-art industry-level baselines by significant margins, while each of our novel model components effectively pushes the limit of the overall pipeline.

Note that, although we focus on the example of Facebook data, our place deduplication pipeline is *general* and ready to be applied to any online platform with place data or other deduplication tasks. Full implementation of our place embedding pipeline based on PyTorch will be released after the acceptance of this work.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313456

2 PROBLEM FORMULATION

Nowadays, many online platforms maintain a *place graph* to map the real world. The building blocks of such place graphs, however, instead of actual places, are usually *online place pages* that capture certain attributes of actual places, as illustrated in Figure 1. Since such online places can be created by users, imported from third-party companies or scraped from the Web, many of them are *duplicated*, as they in fact describe the same real-world places. In this work, to improve the quality of place graphs, we aim at the following novel but emergent problem.

Definition 1. *Place Deduplication. Given a large set of online place pages, find those describing the same real-world places.*

Input. Given a set of online places $\mathcal{P}$, we represent the attributes of each place $p_i \in \mathcal{P}$ as $\mathbf{a}_i \in \mathcal{A}$. Such attributes can be ad-hoc, for example, place names, addresses, phone numbers, visitor histories, images, relevant posts and so on, which are essentially combinations of numerical, categorical, textual and even visual features. Moreover, due to the reality, many attributes can be missing and inaccurate.



Figure 1: Example input with ad-hoc page features.

Output. For each place $p_i \in \mathcal{P}$, we aim to compute a unique fixed-sized low-dimensional vector (*a.k.a.* embedding) $\mathbf{u}_i \in \mathcal{U}$, to encode its ad-hoc attributes $\mathbf{a}_i \in \mathcal{A}$. After getting such embeddings, various related tasks can be efficiently addressed through off-the-shelf machine learning algorithms. For example, duplication candidate fetch (find places that may be potentially duplicated) can be done through fast k-NN search [14, 20], pair-wise duplication prediction (decide if two places are duplications) simply involves the computation of pair-wise Euclidean distances, and meta-page discovery (find common actual places among all online places) can be achieved using clustering techniques such as k-means [17].

Labels. To learn the proper place embedding $\mathcal{U}$ that facilitates place deduplication, besides place attributes $\mathcal{A}$, we also aim to leverage some labeled data $\mathcal{L}$, which are mainly from curation, crowdsourcing, and feedbacks. The labels are usually in the form of place pairs like $l = \{(a, b), y_l\}$, where a and b are two probe places. $y_l = 1$ denotes a positive pair, meaning a and b are labeled as duplications, and $y_l = 0$ denotes a negative pair. Labels from different sources can have varying qualities and biases.

Present work. Taking the ad-hoc raw place attributes $\mathcal{A}$, we firstly transform them into fixed-sized numerical vectors $\mathcal{X}$ with unsupervised learning techniques, which captures the key information in $\mathcal{A}$ and can be easily processed by subsequent machine learning algorithms. Guided by labeled place pairs in $\mathcal{L}$, we then explore the feature representation $\mathcal{X}$ through supervised learning techniques and learn the final place embeddings $\mathcal{F}$ as vectors in a metric space where labeled duplicated places are close and non-duplicated places lie far apart. While we focus on place deduplication, the concepts and methods developed in this work are not bounded to the particular problem, but can be generally useful for deduplicating various objects like texts and users with similar types of attributes.

3 RELATED WORK AND MOTIVATIONS

We find the place deduplication problem closely related to several key tasks that heavily rely on advanced machine learning techniques, especially in the domains of computer vision (CV) and natural language processing (NLP).

3.1 CV: Person Re-Identification

One of the key tasks in CV is person re-identification [4–6, 12, 38, 46], also referred to as face recognition [37, 41, 42]. Given a large number of human images, the task is to find images of the same person. In concept, this task is quite close to ours as defined in Section 2.

In CV, mainstream algorithms often tackle re-identification by projecting the images into a latent embedding space, while tuning the embedding space based on labeled data with a *triplet loss* to enforce the correct order for each probe image. Their conceptual optimization objective can be expressed as follows.

$$\mathcal{J}_{tr} = \sum_{t=(a,p,n)} \max(0, d(\mathbf{u}_a, \mathbf{u}_p) - d(\mathbf{u}_a, \mathbf{u}_n) + \alpha), \qquad (1)$$

where $\mathbf{u} = \mathbf{f}(\mathbf{x})$ and $d(\mathbf{u}_i, \mathbf{u}_j) = d_e(\mathbf{u}_i, \mathbf{u}_j) = ||\mathbf{u}_i - \mathbf{v}_j||_2^2$. $\mathbf{x}$ is the original image features (*e.g.*, 96×96 pixels), $\mathbf{f}(\cdot)$ denotes the embedding projection function (*e.g.*, a deep CNN), α is the margin hyper-parameter, and $||\cdot||_2^2$ denotes the Frobenius norm.

However, while images naturally come with rich visual features that can be well explored by the CNN models, the attributes of our places are much more ad-hoc and hard to capture. Another limitation of their models is the requirement of training data to be accurate and in the form of *triplets*, *i.e.*, $t = (a, p, n)$, where a is the probe image, p is the positive image and n is the negative image. In person re-identification, the labels are often generated from image collections taken of the same persons. Therefore, while the quality of input images may be different, the labels are often accurate, and the triplet requirement is easy to be satisfied.

In our scenario of place deduplication, labels are generated from different sources like curation and crowdsource based on samples of all possible place pairs, and are thus prone to varying qualities and biases. Moreover, instead of triplets, the training data naturally come as *pairs*, in the form of $l = \{(a, b), y_l\}$, as defined in Section 2. Finally, place duplication labels are extremely sparse– in our case, only about 10^{-5} percent of place pairs are labeled. It requires non-trivial adaptions of the CV models to work for place deduplication, which we will talk about in Section 5 and 6.

3.2 NLP: Entity Resolution

If we only consider place names, the problem is close to synonym detection. However, as most algorithms leverage sentence-level contexts to learn word similarities [32, 33, 35, 39, 43, 47], they are not directly applicable to place deduplication where such contexts

are often not available. Instead, we may consider direct matching of place names, which can be done using deep neural networks [3, 15, 19, 30, 40, 45, 49]. However, these models are known to be more suitable for longer sentences with more complex structures, rather than the short simple place names, and they require large amounts of training data, which is extremely hard to acquire for place deduplication [7, 9, 31]. Finally, by considering additional place attributes like location, ad-hoc models have been developed based on heuristic feature engineering [8, 10, 22], but their flexibility is limited to take more different place attributes, such as address and category. Thus, it is desirable for us to develop our own systematic and flexible place deduplication pipeline beyond the existing works on entity resolution, as we will discuss in Section 5 and 6.

Interestingly, another particular trend of entity resolution is to construct the word networks from search engine query logs and compute network clustering to detect synonym sets [2, 18, 34]. Since it is non-trivial to put everything we have into a single network, their methods are not directly applicable. Nonetheless, we find the ideas of network construction and clustering intriguing, and are able to leverage both of them in Section 5 and 6, respectively.

4 DATA AND EXPERIMENTAL SETUPS

In this section, we describe the massive dataset we use, upon which place deduplication is desired. It is a subset of the anonymous real-world place graph from Facebook including a total of 730M online place pages. In this work, we take the Facebook place graph as an example to show the effectiveness and efficiency of the methods we develop, while they are generally applicable to any online platforms with place data or other deduplication tasks.

4.1 Data Preparation

As we discussed in Section 2, places can have various ad-hoc attributes with varying qualities and incompleteness, which need to be leveraged through specifically designed methods, as we will show in Section 5.

We process and generate training data of about 4.6M pairs of places from 19 different sources, including curation, crowdsourcing, feedbacks and so on. Labels from curation are generally of higher qualities, while most of the others like feedbacks generated through surveys and crowdsourcing produced by cheap labors may be rather *noisy*. They cover a total of about 4M places. Among them, about 1.1M pages are labeled with duplications and 3M pages with non-duplications. The average number of duplications is 0.35, whereas the maximum number of duplications is 297. The average number of non-duplications is 1.98, and the maximum number of non-duplications is over 1K. Therefore, the training data are *sparse* and *biased*, and such sparsity and bias are different across the sources. In Section 6, we will discuss how to leverage such training data with varying noise, sparsity, and bias.

4.2 Evaluation Methods

The testing data we use is a golden set of 47K pages completely separate from the training data, which is deliberately curated for place quality evaluation for production. It contains different place pages from the training data.

To compare different embedding models, we firstly compute the vector representations $\mathcal{U}$ of all places in the testing set. For the set of places labeled with both duplications and non-duplications (*i.e.*, $a \in \Omega$), we compute the average pair-wise accuracy as follows:

$$\text{ACC} = \frac{1}{|\Omega|} \sum_{a \in \Omega} \frac{1}{|\Phi_a||\Psi_a|} \sum_{p \in \Phi_a, n \in \Psi_a} \mathbb{I}(||\mathbf{u}_a - \mathbf{u}_p||_2^2 < ||\mathbf{u}_a - \mathbf{u}_n||_2^2),$$

where Φ_a and Ψ_a are the sets of labeled duplications and non-duplications of place a, respectively. This metric measures the utility of the embedding for the task of pair-wise duplication prediction.

Besides ACC, we also compute the nearest neighbors of each place through fast k-NN [20], and compute the precision and recall at K based on the set of places labeled with duplications (*i.e.*, $a \in \Theta$), to measure the utility of the embedding for the task of point-wise duplication candidate fetch.

5 UNSUPERVISED FEATURE GENERATION

The key place attributes we aim to leverage in this work are name, address, coordinate and category, while other attributes can also be easily incorporated in the future. Among them, name and address are textual, coordinate is numerical, and category is categorical, which need to be handled differently. Moreover, many of them are noisy and incomplete, requiring the model to be robust and flexible.

5.1 Encoding Name

We start with place name, because it is often the primary and most indicative feature towards deduplication(*e.g.*, "Metropolitan Museum of Art" and "The MET"). Motivated by the success of word embedding in NLP, we propose to capture the place name semantics through embedding. The idea is to leverage the distributional information about words and infer word semantic similarities based on the context, so as to recognize common misspellings (*e.g.*, "capitol" as "capital", "corner" as "conner"), acronyms (*e.g.*, "street" as "st"), synonyms (*e.g.*, "plaza" and "square") and *etc.*

Method 1: *Skip-gram* + place name corpus. We firstly train a simple word embedding model on all place names we have, which include 730M short texts. For preprocessing, we prune all place names into the shortest name variants by removing all location prefixes and suffixes in the names (*e.g.*, remove "New York" from "Time Square New York"); we also normalize all place names by replacing special characters with spaces (*e.g.*, replace "$" or emojis with a single space) and change all letters to lower cases.

Since place names are often quite short (on average only 3.11 words after preprocessing), we directly apply the *Skip-gram* model [28] by sampling word-context pairs from place names. Place embedding is then computed as the average of word embedding.

This method, while efficiently providing numerical place name representations that can be processed by subsequent machine learning algorithms, has quite a few drawbacks as listed below.

- Limited distributional information: The training corpus, although is quite large, provides rather limited statistics around words and their contexts, since the corpus is chunked into short texts.
- Biased samples: Unlike sampling word-context pairs from fixed-sized sliding windows, when sampling from the place names, it is hard to avoid the bias towards either shorter or longer names.

- Ignorance of word internal structure: Standard word vectors ignore word internal structure that contains rich information, which might be especially useful for rare or misspelled words.

Method 2: *Fast-text* + Facebook post corpus. To ameliorate the above limitations, we adopt the advanced *Fast-text* method [27], with strategies including position-dependent weighting, phrase generation and subword enrichment [1, 21]. Moreover, to capture more casual languages used by social network users who frequently create the place pages, instead of training the *Fast-text* model on standard NLP corpus like Wikipedia[1], we use anonymized word embeddings derived from the public posts on Facebook in 10 years. The corpus has around 1.9T words in total, which is about 3 times bigger than the massive Common Crawl data[2] used to train the state-of-the-art public word embedding [27].

5.2 Incorporating Address

In our place graph, about two-thirds of the places have address information. It is especially useful in differentiating branches of the same stores (*e.g.*, the different Starbucks in a city). Compared with names, addresses are more often incomplete, but less noisy, because most of them are validated and filtered before recorded. Therefore, we observe much fewer misspellings and abbreviations. Moreover, addresses are often longer than names, which naturally provides richer distributional information and allows less biased samples. Finally, in addresses, we do not really want semantically similar words to be close– for example, different words like "street", "avenue" and "boulevard" in the address should indicate different places although their semantic meanings are similar. We use the same two approaches for place names to embed place addresses.

5.3 Leveraging Coordinate and Category

Place coordinates and categories provide additional information about duplications. Coordinates are just 2-dim numerical vectors, whereas categories can be converted to 0-1 dummy variables. In this work, we focus on 19 common ones such as Shopping and Restaurant, while the algorithm can generalize to any other appropriate subsets of categories.

A simple way to leverage place coordinate and category is to concatenate the 2-dim numerical coordinate and 19-dim 0-1 dummy vector of category to the place embedding. However, in practice, we find such concatenation not helpful and even lead to worse performance as can be seen later, probably because such variables are not compatible with the word embeddings.

Motivated by a recent work on place recommendation [50], we refine the place embedding through unsupervised embedding smoothing on a place network. The idea is to require the embeddings of places that have similar coordinates or same categories to be close.

Specifically, we construct a place network $\mathcal{N} = \{\mathcal{P}, \mathcal{E}\}$, where $\mathcal{P}$ is the set of all places. We then construct two types of edges $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$ based on coordinates and categories, respectively. Following the grid-based binning approach in [51], we group places into squared bins based on coordinates, and add a coordinate edge $e_i \in \mathcal{E}_1$ between places in the same bins; a category edge $e_j \in \mathcal{E}_2$ is added between places belonging to the same categories.

Following [50], we derive the loss that enforces smoothness among places that are close on the place network as

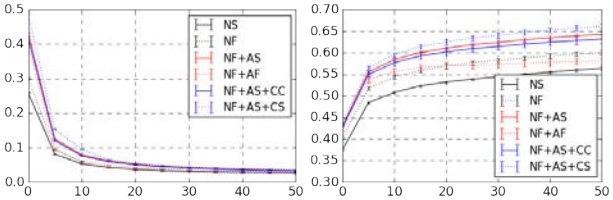
$$\begin{aligned}\mathcal{J}_{sm} &= -\sum_{(p_i,p_c)} \log p(p_c|p_i) \\ &= -\sum_{(p_i,p_c)} \log[\phi_c^T \mathbf{g}(\mathbf{a}_i) - \log \sum_{p'_c \in \mathcal{P}} \exp(\phi_{c'}^T \mathbf{g}(\mathbf{a}_i))] \\ &= -\mathbb{E}_{(p_i,p_c,\gamma)} \log \sigma(\gamma \phi_c^T \mathbf{g}(\mathbf{a}_i)). \end{aligned} \quad (2)$$

In the first line, we firstly formulate the standard *Skip-gram* objective adapted to predict the correct graph context of place based on coordinates and categories, which is then decomposed into the second line, where ϕ_c is the learnable context embedding of place p_c, $\mathbf{g}(\cdot)$ is a learnable smoothing function (*e.g.*, a single layer perceptron with the same input and output sizes) that maps the original place embedding $\mathbf{a}_i$ to the smoothed embedding space. This objective is hard to estimate due to the summation over all contexts in $\mathcal{P}$ in the second term, so we write it into an equivalent expectation over the distribution of $p(p_i, p_c, \gamma)$ in the third line, where $\sigma(x) = 1/(1 + e^{-x})$, and approximate it by applying the popular negative sampling approach [28].

5.4 Experimental Evaluations

We compare the following combinations to comprehensively show the overall effectiveness of our framework as well as how each of the model components helps in improving the embedding quality.

- **NS:** 50-dim place *Name* embedding produced by the *Skip-gram* model trained on the place name corpus.
- **NF:** 300-dim place *Name* embedding produced by the *Fast-text* model trained on the Facebook post corpus.
- **NF+AS:** *NF* concatenated with the 50-dim place *Address* embedding produced by the *Skip-gram* model trained on the place address corpus.
- **NF+AF:** *NF* concatenated with the 300-dim place *Address* embedding produced by the *Fast-text* model trained on the Facebook post corpus.
- **NF+AS+CC:** *NF+AS Concatenated* with the 21-dim place *Coordinate and category* vectors.
- **NF+AS+CS:** *NF+AS Smoothed* on the place network constructed *w.r.t.* place *Coordinate and category*.



(a) PRE@K (b) REC@K

Figure 2: Performance of feature generation methods.

Figure 2 and Table 1 show the performance of different feature generation methods. As can be clearly observed: 1) When we only consider the place name, *NF* performs significantly better than *NS* regarding all metrics, which supports our idea of training the advanced *Fast-text* model on the massive Facebook post corpus; 2) Incorporating place address is generally helpful, but the simple *AS*

[1] https://dumps.wikimedia.org/enwiki/latest/
[2] https://commoncrawl.org/2017/06

Method	avg. PRE	avg. REC	ACC
NS	.03794 ± .00010	.5334 ± .0034	.6212 ± .0325
NF	.04553 ± .00018	.5708 ± .0052	.6718 ± .0107
NF+AS	.05216 ± .00030	.6162 ± .0067	.6917 ± .0181
NF+AF	.05008 ± .00018	.5866 ± .0061	.6424 ± .0324
NF+AS+CC	.05026 ± .00034	.6036 ± .0083	.6716 ± .0297
NF+AS+CS	.05508 ± .00024	.6476 ± .0058	.6989 ± .0135

Table 1: Performance of feature generation methods.

model we advocate for produces more significant improvements than the heavy *AF* model, which tells the importance of choosing the proper model rather than the complex one; 3) To leverage place coordinate and category, our proposed network-based embedding smoothing method (*CS*) is much more effective than the direct vector concatenation (*CC*).

The *avg.* PRE and *avg.* REC are taken over the 100 measures of PRE@K and REC@K when K varies from 1 to 100, to directly compare and reveal the relative effectiveness of different methods. The absolute values of *avg.* PRE are pretty low because the numbers of labeled duplications are low in our testing data. Based on such results, we will use *NF+AS+CS* as our feature generation model and use its output $\mathcal{X}$ as the input of subsequent supervised metric learning models. Note that, other than places with the particular attributes as we focus on in this work, the methods developed here can be combined in different ways to capture textual, categorical and numerical attributes of various real-world objects.

6 SUPERVISED METRIC LEARNING

Besides place attributes, we also aim to leverage labeled place pairs through supervised metric learning. The idea is to learn a non-linear projection function $\mathcal{F}$, which transforms the place features $\mathcal{X}$ into a metric space, where duplicated places are close and non-duplicated places are far apart. Therefore, the model should be able to explore and stress the important features that indicate duplications.

6.1 Basic Models

We construct the basic metric learning models by applying simple MLP upon the place embedding $\mathcal{X}$. However, since direct leverage of standard triplet loss commonly used in person re-identification [37] and entity resolution [33] in Eq. 1 requires the sampling of the third place based on our pair-wise place labels, which can introduce a lot of non-relevant and trivial training data, we propose to replace the triplet loss with a pair-wise contrastive loss as

$$\mathcal{J}_{pr} = \sum_{l=\{(a,b),y_l\}} [y_l d(\mathbf{u}_a, \mathbf{u}_b) + (1 - y_l) \max(0, \alpha - d(\mathbf{u}_a, \mathbf{u}_b))],$$

where $d = d_e$ is the same as in Eq. 1. Besides the Euclidean distance function, we also tried various other distance functions including cosine similarity and learnable bilinear distance [33].

We comprehensively evaluated the performance of various basic models, and found that the *PE* (Pair-wise loss with Euclidean distance function) constantly performs the best, indicating the effectiveness of replacing the standard triplet loss in traditional CV and NLP tasks. Based on such observation, we will focus on further improving the *PE* model in the rest of this work.

6.2 Hard Sampling

Training the model with our massive noisy labeled place pairs can take quite significant time, most of which is wasted on trivial samples. To this end, we design the following novel criteria for dynamic batch-wise hard sampling towards our pair-wise loss.

$$C_{l=(a,b)} = \mathbb{I}\,[y_l d(\mathbf{u}_a, \mathbf{u}_b) - (1 - y_l) d(\mathbf{u}_a, \mathbf{u}_b) > \frac{\beta}{|B|} \sum_{l' \in B} y_{l'} d(\mathbf{u}_{a'}, \mathbf{u}_{b'}) - (1 - y_{l'}) d(\mathbf{u}_{a'}, \mathbf{u}_{b'})], \quad (3)$$

where B denotes the set of samples in a batch, β is the slack hyper-parameter controlling the amount of hard samples, and $d(\cdot)$ can be implemented with either $d_f(\cdot)$ or $d_b(\cdot)$. A sample pair l is selected to contribute to the loss for gradient computation when $C_l = 1$. The idea is to only focus on the hard samples where the model fails to put the labeled duplications close enough or non-duplications far away, compared with the average distances.

6.3 Attentive Training

Our labeled data are from different sources, which may have varying quality and bias. For example, data curated by a particular team at a certain time might be biased towards specific metrics. This often happens when people recognize the ignorance of particular situations and aim to improve them upon iterating to the next period of curation. Moreover, data labeled by the professional curation teams and rookie crowdsourcing workers can have quite different qualities. Therefore, it is intuitive to automatically assign different weights to the samples from different sources during training.

To deal with the varying quality and bias of training samples from multiple sources, we design a novel source-oriented attentive training technique based on the idea of self-attention [11, 44], which has shown to be effective in various deep learning scenarios.

For each place p_i, rather than learning a single embedding vector $\mathbf{f}(\mathbf{x}_i)$, we learn two embedding vectors $\mathbf{k}(\mathbf{x}_i)$ and $\mathbf{v}(\mathbf{x}_i)$, which we call the *key embedding* and *value embedding*, respectively. In practice, we can implement both $\mathbf{k}(\cdot)$ and $\mathbf{v}(\cdot)$ as FNNs and let them share a few layers. Besides, we also learn a global source embedding $\mathbf{q}(s_j)$ for each training source s_j, which can be implemented as an embedding look-up table. The size of $\mathbf{q}(\cdot)$ is twice as $\mathbf{k}(\cdot)$. Subsequently, we design the weight of the training sample $l = (a, b)$ from source s as

$$w_{ls} = \frac{\exp([\mathbf{k}(\mathbf{x}_a), \mathbf{k}(\mathbf{x}_b)]^T \mathbf{q}(s))}{\sum_{s'} \exp([\mathbf{k}(\mathbf{x}_a), \mathbf{k}(\mathbf{x}_b)]^T \mathbf{q}(s'))}, \quad (4)$$

where $[\cdot]$ denotes vector concatenation. With $\mathcal{W}$ defined, we rewrite our loss function in Eq. 3 into

$$\mathcal{J}'_{pr} = \sum_{s} \sum_{l=\{(a,b),y_l\} \in \mathcal{T}_s} w_{ls} [y_l d(\mathbf{v}(\mathbf{x}_a), \mathbf{v}(\mathbf{x}_b)) + (1 - y_l) \max(0, \alpha - d(\mathbf{v}(\mathbf{x}_a), \mathbf{v}(\mathbf{x}_b)))], \quad (5)$$

where $\mathcal{T}_s$ denotes the set of training samples from source s.

Training with Eq. 5 allows the model to learn a source-oriented self-attention weight for each training pair. To be specific, if similar samples keep appearing in a certain source, they are likely redundant and less useful, and the model will automatically reduce their contribution to the loss. However, if similar samples appear across different sources, the model will put more trust on their correctness by relatively increasing their attention weights. Finally, unseen samples will likely get higher attention.

6.4 Label Denoising

Place deduplication can be essentially regarded as a semi-supervised clustering problem, where pages of the same places should naturally form clusters in the embedding space. This is true mainly because the duplication relationship is *transitive*. While such clusters are ideal for duplication detection, they can also help reduce the noise within training data, *i.e.*, labels that conflict with the cluster structures are likely noisy. To leverage this insight, we design a novel soft clustering-based label denoising module based on the idea of self-training neural networks [16, 29, 48]. Particularly, we introduce a loss based on the KL-divergence

$$\mathcal{J}_{dn} = \rho \, \mathrm{KL}(C||\mathcal{D}) = \rho \sum_i \sum_k c_{ik} \log \frac{c_{ik}}{d_{ik}}, \tag{6}$$

where ρ is a hyper-parameter. $d_{ik} \in \mathcal{D}$ is the probability of assigning place p_i to the kth cluster, under the assumption of Student's t-distribution with degree of freedom set to 1 [26], *i.e.*,

$$d_{ik} = \frac{(1 + ||\mathbf{f}(\mathbf{x}_i) - \mathbf{u}_k||^2)^{-1}}{\sum_j (1 + ||\mathbf{f}(\mathbf{x}_i) - \mathbf{u}_j||^2)^{-1}}. \tag{7}$$

It is basically a kernel function that measures the similarity between the embedding of place p_i and the cluster center u_k. C is an auxiliary target distribution defined as

$$c_{ik} = \frac{d_{ik}^2 / g_k}{\sum_{k'} d_{ik'}^2 / g_{k'}}, \tag{8}$$

where $g_k = \sum_i d_{ik}$ is the total number of places softly assigned to the kth cluster. Raising $\mathcal{D}$ to the second power and then dividing by the number of places per cluster allows the target distribution C to improve cluster purity and leverage the confident assignments to help reduce the influence of noisy labels.

6.5 Experimental Evaluations

We compare the following methods to demonstrate the effectiveness of our proposed supervised metric learning techniques.

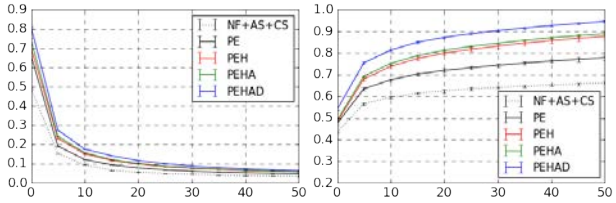
- **NF+AS+CS:** The best place features we got through unsupervised feature generation in Section 5, which is also the input of the following metric learning methods.
- **PE:** The best basic metric learning method adapted from CV and NLP with the *Pair-wise contrastive loss* and *Euclidean distance function*, as discussed in Section 6.1.
- **PEH:** *PE* improved with batch-wise hard sampling.
- **PEHA:** *PEH* improved with our novel source-oriented attentive training technique.
- **PEHAD:** *PEHA* improved with our novel soft clustering-based label denoising technique.

We also compare our models with several state-of-the-art classification algorithms that can be applied for pair-wise duplication prediction, *i.e.*, logistic regression (*LR*), support vector machine (*SVM*), random forest (*RF*) and gradient boosting decision tree (*GBDT*). These algorithms do not produce place embeddings and thus cannot be efficiently evaluated through fast k-NN in datasets with hundreds of millions of places. Therefore, we only compute their standard classification accuracy of duplication prediction on the testing data, as shown in Table 2.

Figure 3 and Table 3 show the performance of compared models. As we can see, our supervised metric learning framework can significantly improve the place embedding quality, because labeled data allow the models to pick out the more important features that differentiate duplication pairs from non-duplications. Moreover, each of our proposed model components further boosts the overall model performance, and their coherent combination (*PEHAD*) has the best performance regarding all metrics in the evaluation. Our models also significantly outperform all of the state-of-the-art non-embedding algorithms in Table 2– *PEHAD* achieves over 18% relative improvement on *GBDT* (the strongest baseline).

Method	LR	SVM	RF	GBDT	PEHAD
ACC	0.7514	0.7699	0.7786	0.7828	0.9279

Table 2: Performance of state-of-the-art algorithms.

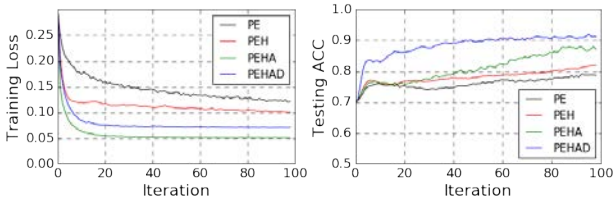


(a) PRE@K (b) REC@K

Figure 3: Performance of our metric learning methods.

Method	avg. PRE	avg. REC	ACC
NF+AS+CS	.05508 ± .00024	.6476 ± .0058	.6989 ± .0135
PE	.07533 ± .00034	.7534 ± .0083	.7926 ± .0297
PEH	.08970 ± .00039	.8434 ± .0102	.8580 ± .0238
PEHA	.09274 ± .00019	.8554 ± .0047	.8815 ± .0100
PEHAD	.10780 ± .00021	.9119 ± .0062	.9279 ± .0104

Table 3: Performance of our metric learning methods.



(a) PRE@K (b) REC@K

Figure 4: Training loss and testing accuracy curves.

To better understand how each of the components contributes to the overall model, we also closely observe the training loss and testing accuracy during the training of different models. As we can see in Figure 4: 1) Hard sampling is effective in speeding up the convergence of training loss, and allows the model to keep learning after the loss seems to converge; 2) Attentive training leads to lower losses due to the additional weights. It does not lead to significant performance gain on top of the basic *PE* model with hard sampling, but it does make the training process more stable and reduce the performance variance across different trains of the same model; 3) Label denoising empowers the model to rapidly remove the influence of noisy labels and achieve the peak performance after a small number of training iterations, which may further allow efficient model training with early stop.

Finally, all techniques here are also not bounded to the problem of place deduplication, but can rather be combined to leverage noisy pair-wise labels from multiple sources in various other domains.

7 CONCLUSIONS

In this paper, we collect data from the real-world place graph of Facebook as an example to systematically study the novel problem of place deduplication and comprehensively evaluate the effectiveness of our proposed place embedding pipeline. The concepts and methods developed, although do not provide an end-solution for place deduplication, successfully improve the quality of place embedding for intermediate tasks like duplication prediction and candidate fetch, which largely facilitates further place deduplication.

As for future work, it is interesting to further improve both the unsupervised feature generation model to incorporate more attributes, and the supervised metric learning model to leverage various training signals with efficiency. It is also interesting to apply the learned place embedding to facilitate other tasks related to place graph quality such as junk place prediction and common place clustering, as well as downstream applications including next destination recommendation and location-based ads ranking.

REFERENCES

[1] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL* (2017).

[2] Kaushik Chakrabarti, Surajit Chaudhuri, Tao Cheng, and Dong Xin. 2012. A framework for robust discovery of entity synonyms. In *KDD*. ACM, 1384–1392.

[3] Haolan Chen, Fred X Han, Di Niu, Dong Liu, Kunfeng Lai, Chenglin Wu, and Yu Xu. 2018. MIX: Multi-Channel Information Crossing for Text Matching. In *KDD*. ACM, 110–119.

[4] Shi-Zhe Chen, Chun-Chao Guo, and Jian-Huang Lai. 2016. Deep ranking for person re-identification via joint representation learning. *TIP* 25, 5 (2016), 2353–2367.

[5] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. 2017. Beyond triplet loss: a deep quadruplet network for person re-identification. In *CVPR*, Vol. 2. IEEE.

[6] De Cheng, Yihong Gong, Sanping Zhou, Jinjun Wang, and Nanning Zheng. 2016. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *CVPR*. IEEE, 1335–1344.

[7] Peter Christen, Dinusha Vatsalan, and Qing Wang. 2015. Efficient entity resolution with adaptive and interactive training data selection. In *ICDM*. IEEE, 727–732.

[8] Hsiu-Min Chuang and Chia-Hui Chang. 2015. Verification of poi and location pairs via weakly labeled web data. In *WWW*. ACM, 743–748.

[9] Guilherme Dal Bianco, Renata Galante, Carlos A Heuser, Marcos Gonçalves, and Sergio Canuto. 2016. A practical and effective sampling selection strategy for large scale deduplication. In *ICDE*. IEEE, 1518–1519.

[10] Nilesh Dalvi, Marian Olteanu, Manish Raghavan, and Philip Bohannon. 2014. Deduplicating a places database. In *WWW*. ACM, 409–418.

[11] Misha Denil, Loris Bazzani, Hugo Larochelle, and Nando de Freitas. 2012. Learning where to attend with deep architectures for image tracking. *Neural computation* 24, 8 (2012), 2151–2184.

[12] Shengyong Ding, Liang Lin, Guangrun Wang, and Hongyang Chao. 2015. Deep feature learning with relative distance comparison for person re-identification. *Pattern Recognition* 48, 10 (2015), 2993–3003.

[13] Gregory Ference, Mao Ye, and Wang-Chien Lee. 2013. Location recommendation for out-of-town users in location-based social networks. In *CIKM*. ACM, 721–726.

[14] Vincent Garcia, Eric Debreuve, and Michel Barlaud. 2008. Fast k nearest neighbor search using GPU. In *CVPRW*. IEEE, 1–6.

[15] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *CIKM*. ACM, 55–64.

[16] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. 2017. Improved Deep Embedded Clustering with Local Structure Preservation. In *IJCAI*.

[17] Jiawei Han, Jian Pei, and Micheline Kamber. 2011. *Data mining: concepts and techniques*. Elsevier.

[18] Yeye He, Kaushik Chakrabarti, Tao Cheng, and Tomasz Tylenda. 2016. Automatic discovery of attribute synonyms using query logs and table corpora. In *WWW*. ACM, 1429–1439.

[19] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *NIPS*. 2042–2050.

[20] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734* (2017).

[21] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *EACL*, Vol. 2. 427–431.

[22] Ivan Kozhevnikov and Vladimir Gorovoy. 2016. Comparison of Different Approaches for Hotels Deduplication. In *KESW*. Springer, 230–240.

[23] Xutao Li, Gao Cong, Xiao-Li Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. 2015. Rank-geofm: A ranking based geographical factorization method for point of interest recommendation. In *SIGIR*. ACM, 433–442.

[24] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *KDD*. ACM, 831–840.

[25] Yong Liu, Wei Wei, Aixin Sun, and Chunyan Miao. 2014. Exploiting geographical neighborhood characteristics for location recommendation. In *CIKM*. 739–748.

[26] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 9, Nov (2008), 2579–2605.

[27] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *LREC*.

[28] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. 3111–3119.

[29] Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *CIKM*. ACM, 86–93.

[30] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text Matching as Image Recognition.. In *AAAI*. 2793–2799.

[31] Kun Qian, Lucian Popa, and Prithviraj Sen. 2017. Active Learning for Large-Scale Entity Resolution. In *CIKM*. ACM, 1379–1388.

[32] Longhua Qian, Guodong Zhou, Fang Kong, and Qiaoming Zhu. 2009. Semi-supervised learning for semantic relation classification using stratified sampling strategy. In *EMNLP*. 1437–1445.

[33] Meng Qu, Xiang Ren, and Jiawei Han. 2017. Automatic Synonym Discovery with Knowledge Bases. In *KDD*. ACM, 997–1005.

[34] Xiang Ren and Tao Cheng. 2015. Synonym discovery for structured entities on heterogeneous graphs. In *WWW*. ACM, 443–453.

[35] Stephen Roller, Katrin Erk, and Gemma Boleda. 2014. Inclusive yet selective: Supervised distributional hypernymy detection. In *COLING*. 1025–1036.

[36] Mohamed Sarwat, Justin J Levandoski, Ahmed Eldawy, and Mohamed F Mokbel. 2014. LARS*: An efficient and scalable location-aware recommender system. *TKDE* 26, 6 (2014), 1384–1399.

[37] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *CVPR*. IEEE, 815–823.

[38] Chi Su, Shiliang Zhang, Junliang Xing, Wen Gao, and Qi Tian. 2016. Deep attributes driven multi-camera person re-identification. In *ECCV*. 475–491.

[39] Ang Sun and Ralph Grishman. 2010. Semi-supervised semantic pattern discovery with guidance from unsupervised pattern clusters. In *COLING*. 1194–1202.

[40] Qiang Sun and Yue Wu. 2018. A Multi-level Attention Model for Text Matching. In *ICANN*. Springer, 142–153.

[41] Yi Sun, Xiaogang Wang, and Xiaoou Tang. 2015. Deeply learned face representations are sparse, selective, and robust. In *CVPR*. IEEE, 2892–2900.

[42] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. 2014. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*. 1701–1708.

[43] Dmitry Ustalov, Alexander Panchenko, and Chris Biemann. 2017. Watset: automatic induction of synsets from a graph of synonyms. In *ACL*.

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*. 5998–6008.

[45] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations.. In *AAAI*, Vol. 16. 2835–2841.

[46] Faqiang Wang, Wangmeng Zuo, Liang Lin, David Zhang, and Lei Zhang. 2016. Joint learning of single-image and cross-image representations for person re-identification. In *CVPR*. IEEE, 1288–1296.

[47] Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. 2014. Learning to distinguish hypernyms and co-hyponyms. In *COLING*. 2249–2259.

[48] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *ICML*. IEEE, 478–487.

[49] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *SIGIR*. ACM, 55–64.

[50] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. 2017. Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation. In *KDD*. ACM, 1245–1254.

[51] Carl Yang, Chao Zhang, Xuewen Chen, Jieping Ye, and Jiawei Han. 2018. Did You Enjoy the Ride: Understanding Passenger Experience via Heterogeneous Network Embedding. In *ICDE*.

[52] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *SIGIR*. ACM, 325–334.

[53] Hongzhi Yin, Xiaofang Zhou, Yingxia Shao, Hao Wang, and Shazia Sadiq. 2015. Joint modeling of user check-in behaviors for point-of-interest recommendation. In *CIKM*. ACM, 1631–1640.

Cyberbullying Ends Here: Towards Robust Detection of Cyberbullying in Social Media

Mengfan Yao
University at Albany, SUNY
Department of Computer Science
myao@albany.edu

Charalampos Chelmis
University at Albany, SUNY
Department of Computer Science
cchelmis@albany.edu

Daphney–Stavroula Zois
University at Albany, SUNY
Electrical and Computer Engineering Department
dzois@albany.edu

ABSTRACT

The potentially detrimental effects of cyberbullying have led to the development of numerous automated, data–driven approaches, with emphasis on classification accuracy. Cyberbullying, as a form of abusive online behavior, although not well–defined, is a repetitive process, i.e., a sequence of aggressive messages sent from a bully to a victim over a period of time with the intent to harm the victim. Existing work has focused on harassment (i.e., using profanity to classify toxic comments independently) as an indicator of cyberbullying, disregarding the repetitive nature of this harassing process. However, raising a cyberbullying alert immediately after an aggressive comment is detected can lead to a high number of false positives. At the same time, two key practical challenges remain unaddressed: (i) detection timeliness, which is necessary to support victims as early as possible, and (ii) scalability to the staggering rates at which content is generated in online social networks.

In this work, we introduce *CONcISE*, a novel approach for timely and accurate Cyberbullying detectiON on Instagram media SEssions. We propose a sequential hypothesis testing formulation that seeks to drastically reduce the number of features used in classifying each comment while maintaining high classification accuracy. *CONcISE* raises an alert only after a certain number of detections have been made. Extensive experiments on a real–world Instagram dataset with $\sim 4M$ users and $\sim 10M$ comments demonstrate the effectiveness, scalability, and timeliness of our approach and its benefits over existing methods.

CCS CONCEPTS

• **Information systems** → *Collaborative and social computing systems and tools*; *Social networking sites*; *Data mining*; *World Wide Web*; *Social networks*; • **Computing methodologies** → *Machine learning*.

KEYWORDS

Classification, cyberharassment, optimization, sequential selection

ACM Reference Format:
Mengfan Yao, Charalampos Chelmis, and Daphney–Stavroula Zois. 2019. Cyberbullying Ends Here: Towards Robust Detection of Cyberbullying in Social Media. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313462

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313462

1 INTRODUCTION

Cyberbullying, a type of cyberharassment, can take many forms [34], typically however, refers to **repetitive hostile behavior** using digital media (e.g., hurtful comments, videos and images) in an effort to intentionally and repeatedly harass or harm individuals [1, 20]. Cyberbullying is permanent (i.e., content remains accessible online unless removed) and potentially widespread[1] (i.e., online social media provide a wide audience, and quick spread of online posts) [10].

The potentially devastating real–world consequences to victims (including learning difficulties, psychological suffering and isolation, escalated physical confrontations, suicide) [12, 13] have resulted in numerous cyberharassmet classification methods with a focus on detection accuracy [5, 16, 27–29, 33, 43, 44]. While high accuracy is undoubted, the state–of–the–art relies on a **fixed** set of features learned during training for **offline** detection (i.e., after all correspondence has become available), hindering the ability to respond in a **timely** manner (i.e., as soon as possible) to cyberbullying events. Moreover, the **scalability** of existing methods to the staggering rates at which content is generated (e.g., 95 million photos and videos are shared on Instagram per day[2]) has largely remained unaddressed.

Present Work. We propose a two–stage **online** approach designed to reduce the time to raise a cyberbullying alert by (i) sequentially examining comments as they become available over time, and (ii) minimizing the number of feature evaluations necessary for a decision to be made for each comment. We formalize the problem as a sequential hypothesis testing problem, and propose a novel algorithm that satisfies four **key properties**: *accuracy*, *repetitiveness*, *timeliness*, and *efficiency*. We focus on hateful comments, captions and hashtags on Instagram, the online social media platform with the highest percentage of users reporting experiencing cyberbullying [10]. Instagram has more than 800 million registered users as of Sep. 2017, and over 40 billion uploaded photos as of Oct. 2015. To ensure the **reproducibility** of our work, we make the source code of our approach available at https://github.com/IDIASLab/CONcISE.

Outline. The rest of this paper is organized as follows. We first review prior and related work in Section 2. We then formulate the problem of cyberbullying detection in online social networks, define

[1] stopbullying.gov Facts About Bullying: https://www.stopbullying.gov/media/facts/index.html#stats
[2] 33 Mind–Boggling Instagram Stats & Facts for 2018: https://www.wordstream.com/blog/ws/2017/04/20/instagram-statistics

our optimization function, and derive the optimal stopping rule and classification strategy in Section 3. We describe our evaluation methodology and results on a real–world Instagram dataset in Section 4. We conclude with a discussion of our results, limitations, and possible future directions in Section 5.

2 RELATED WORK

In this section, we briefly summarize prior and related work, which mainly constitutes 3 main bodies of research: (i) state–of–the–art cyberbullying detection, (ii) online streaming feature selection (OSFS) methods, and (iii) online learning algorithms for classification.

Cyberbullying Detection. An excellent review of existing cyberbullying detection methods is provided in [1]. More recent approaches include [5, 24, 27, 29, 44]. Meanwhile, only a few approaches have been proposed for cyberbullying detection on Instagram [7, 16, 27, 29, 30, 39, 45]. However, all existing works focus on classification accuracy, neglecting the equally important aspects of scalability and timeliness. With the exception of an incremental cyberbullying detection method to improve detection responsiveness [27], and our proposed approach, no prior or related work has studied cyberbullying as a repetitive process.

Online Classification Methods. Machine learning methods are usually trained offline and deployed without further updates once training is complete. Once deployed, the accuracy of offline–trained models however, deteriorates with time, whereas their retraining may be prohibitive if data is large and/or evolving. At the same time, online learning algorithms [4, 9, 19, 36, 41, 42] that examine data points one at a time, updating their model parameters as new samples arrive, are facing scalability constraints [14]. Nevertheless, most existing methods assume high data quality, a property that is often violated in streaming data, in which noise and missing values are the norm [14]. Although related, our framework fundamentally differs from online classification methods in that the belief on a classification outcome is updated at each time step, allowing detection in real–time.

Online Streaming Feature Selection Methods. Feature selection methods are often employed to identify a small subset of informative features that can globally describe the data well so as to reduce training time, memory requirements and ultimately improve classification accuracy [11, 17, 22, 23, 25, 31]. In contrast to offline feature selection methods, which require all features to be available upfront, online streaming feature selection (OSFS) methods [21, 38, 40, 46] fix the size of the training data, and strive to maintain a feature subset that is sequentially updated with the arrival of each new feature based on relevancy and redundancy [38, 46] and group membership [21, 40]. Unlike OSFS methods, online feature selection methods (e.g., [15, 18, 26, 37]) may update the subset of features with the arrival of new data points (as opposed to new features) while assuming that the full feature space remains unchanged.

To the best of our knowledge, all existing feature selection methods, both off– and on–line, and batch or streaming, output a global subset of discriminatory features to be used for classification (i.e. the same set of features is used after selection for classifying all data points). This is in stark contrast to our approach that conducts online streaming feature selection and classification simultaneously for each new data point, resulting in a potentially different number of features used to classify each data point. Figure 1 illustrates this difference between existing classification methods (Figure 1.a) and our framework (Figure 1.b).

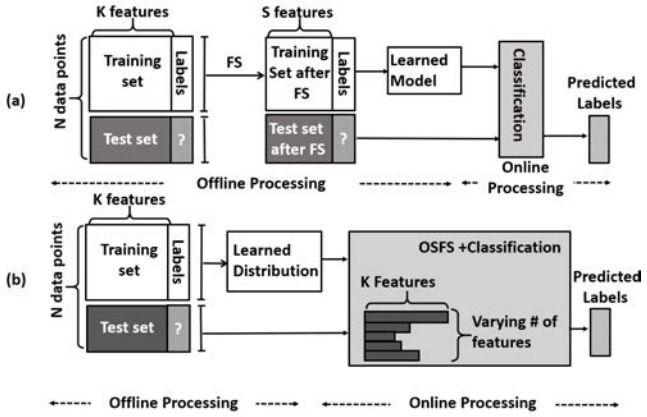


Figure 1: Given a $N \times K$ data matrix and column vector Y of labels, feature selection (FS) is applied to select $S << K$ features before training and classification in (a) existing classification methods. Compare this with (b) our framework, where a model is trained offline once on all features, and online streaming feature selection (OSFS) and classification are performed simultaneously, resulting in a variable number of features used for each data point.

3 PROPOSED METHOD

We use a general data representation, applicable to a wide variety of social media platforms as follows. Each media session $s \in \mathcal{M}$ belongs to a user $u \in \mathcal{U}$, has an associated media object (i.e., image or video) along with its corresponding caption and hashtags, and a set of comments $\{m_1, \ldots, m_{N_s}\}$ from users in $\mathcal{U}$, where N_s denotes the number of comments in s. Our main idea is to formulate cyberbullying detection on Instagram media sessions as a two–stage, online framework with the following desired properties:

- **High Accuracy:** Following [16, 27], we differentiate between **cyberbullying** (i.e., an act of aggression that is repeated over time) and **cyberaggression** (i.e., harassment manifested as one–off profanity). Each media session s is only considered as cyberbullying if more than a given number of aggressive comments have been posted, so that the number of false positives can be reduced.
- **Timeliness:** All comments in a media session are ordered in time. As a result, earlier detection using a **partial** set of the media session (i.e., a small number of comments) allows for timely alerts to be raised. In our approach, we encode the belief of a media session to be indicative of cyberbullying as the number of identified aggressive comments. We measure timeliness as the number of comments "saved" in comparison to a baseline. Figure 4 illustrates this metric.
- **Efficiency:** Given the soaring amount of daily Instagram sessions and comments, classification must be scalable.

To achieve the aforementioned desired properties, we propose a two–stage approach for cyberbullying detection on Instagram as

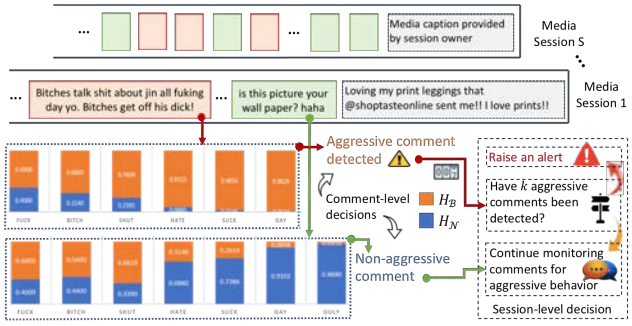


Figure 2: Overview of the proposed approach. Given a set of media sessions $\mathcal{M}$, and an alert threshold, our approach examines comments as they become available over time and raises an alert only after the number of comment–level detections surpasses the threshold. The posterior probability evolution of an aggressive (upper) and non–aggressive (lower) comment as more features are examined is provided for illustration purposes. Notice that the number of features used to make a decision in each case differs.

shown in Figure 2. In the first stage, comments in s are examined and classified sequentially as aggressive or non–aggressive. Whenever an aggressive comment is detected, a comment–level decision is made. In the second stage, before a new comment is examined, the number of comment–level detections is compared to a predefined threshold, and the media session is classified as cyberbullying when the threshold is exceeded.

Next, we introduce our proposed optimization framework that automatically computes the probability of a comment to be indicative of aggression with high accuracy, while accounting for the effort of the framework in improving its chances of reaching a highly accurate conclusion.

3.1 Problem Formulation

In our hypothesis testing formulation, only two hypotheses exist: (i) $H_{\mathcal{B}}$, which denotes the true hypothesis that m is an aggressive comment, and (ii) $H_{\mathcal{N}}$, which represents the case where m is a non–aggressive comment. Each comment m is described by a vector of features $f(m) = \{y_1, y_2, \ldots, y_K\}$, where K is the total number of features, and $y_k \in \mathcal{Y}$. For each feature y_n, the probability $p(y_n|H_{\mathcal{B}})$ (similarly $p(y_n|H_{\mathcal{N}})$) of the evaluation of the nth feature to observe value y_n when the true hypothesis is $H_{\mathcal{B}}$ (similarly when the true hypothesis is $H_{\mathcal{N}}$) is empirically computed. Similarly, the *a priori* probability $P(H_{\mathcal{B}}) = p$ of m being an aggressive comment is also estimated empirically. The probability of m being a non–aggressive comment can be computed as $P(H_{\mathcal{N}}) = 1 - p$.

To calculate the belief for m, the framework evaluates features $f(m)$ sequentially as illustrated in Figure 2. When examining each comment, at each step, the framework has to select between stopping and continuing the evaluation process based on the accumulated information thus far and the cost of reviewing additional features. The cost coefficient $c_n > 0$, where $n = 1, \ldots, K$ represents the value of time and effort spent evaluating the nth feature. We additionally consider misclassification costs $C_{ij} \geq 0, i = \mathcal{B}, \mathcal{N}, j = 1, \ldots, L$, where C_{ij} denotes the cost of selecting possibility j when the true hypothesis is H_i, and L denotes the number of decision choices (e.g., aggressive or non–aggressive). We factor misclassification costs into our approach to quantify the relative importance of detection errors.

We now formally describe our proposed sequential evaluation process to minimize the number of features used to accurately classify each comment m. Specifically, our proposed sequential evaluation process comprises a pair (R, D_R) of random variables. Random variable R takes values in the set $\{0, \ldots, K\}$, and indicates the feature that the framework stops at, and 0 indicates that no features were evaluated. Hence it is referred to as *stopping time* in decision theory [32]. Random variable D_R denotes the possibility to select among L possible choices. Assuming that the random variables y_n are *independent under each hypothesis* $H_i, i = \{\mathcal{B}, \mathcal{N}\}$, the conditional joint probability of $\{y_1, \ldots, y_n\}$ can be computed as $P(y_1, \ldots, y_n|H_i) = \prod_{k=1}^{n} p(y_k|H_i), i = \mathcal{B}, \mathcal{N}$. Both the decision to stop at stage n (i.e., the event $\{R = n\}$), and the selection of possibility j (i.e., $\{D_R = j\}$) depend only on the accumulated information $\{y_1, \ldots, y_R\}$ by the stopping time R. Equivalently, features that may be examined in the future are not used.

3.2 Optimization Setup

Our goal is to use the least number of features for detecting aggression at the comment–level without sacrificing accuracy. To minimize the number of features considered, the stopping time R and the classification rule D_R have to be selected. To this end, we first define the following cost function:

$$J(R, D_R) = \mathbb{E}\left\{ \sum_{n=1}^{R} c_n + \sum_{j=1}^{L} \sum_{i=\mathcal{B},\mathcal{N}} C_{ij} P(D_R = j, H_i) \right\}. \quad (1)$$

The first expression in the cost function regularizes the number of features, whereas the second expression, commonly referred to as Bayes Risk, penalizes the average cost of the classification rule. Our goal can be interpreted as finding the minimum average cost with respect to both random variables R and D_R, i.e., $\min_{R, D_R} J(R, D_R)$, to derive the optimal stopping and classification rules. Intuitively, the optimal rule is to stop at corresponding stopping time R, and use the optimum classification rule D_R. Once the optimal classification rule has been established, the resulting cost becomes only a function of R, and can thus be optimized with respect to R.

3.3 Classification Rule

The classification rule D_R depends only on the accumulated information $\{y_1, \ldots, y_R\}$ by stopping time R. Such accumulated information increases as more features are evaluated, thus, it becomes infeasible to optimize the cost function in Eq. (1). Instead, the *a posteriori probability* $\pi_n \triangleq P(H_{\mathcal{B}}|y_1, \ldots, y_R)$, which corresponds to a sufficient statistic of the accumulated information, can be used to reach the same solution in Eq. (1) without having to consider all $|\mathcal{Y}|^R$ possible combinations in the accumulated information. Specifically, the posterior probability, π_n, at stage n, where the nth feature is extracted and evaluated to generate outcome y_n, can be iteratively computed as:

$$\pi_n = \frac{p(y_n|H_{\mathcal{B}})\pi_{n-1}}{\pi_{n-1}p(y_n|H_{\mathcal{B}}) + (1 - \pi_{n-1})p(y_n|H_{\mathcal{N}})}, \quad (2)$$

where π_{n-1} denotes the posterior probability of a comment being an aggressive comment given the first $n-1$ features, and $\pi_0 = p$.

Using Eq. (2) and the fact that $x_R = \sum_{n=0}^{K} x_n \mathbb{1}_{\{R=n\}}$ for any sequence of random variables $\{x_n\}$, where $\mathbb{1}_A$ is the indicator function for event A (i.e., $\mathbb{1}_A = 1$ when A occurs, and 0 otherwise), the average cost in Eq. (1) can be written compactly as:

$$J(R, D_R) = \mathbb{E}\left\{ \sum_{n=1}^{R} c_n \right\} + \mathbb{E}\left\{ \sum_{j=1}^{L} (C_{Bj}\pi_R + C_{Nj}(1-\pi_R)) \mathbb{1}_{\{D_R=j\}} \right\}. \tag{3}$$

In order to obtain the optimal classification rule D_R for any stopping time R (i.e., find the optimum Bayes test given that the values of the first R features $y_1, \ldots, y_R$ are observed), an independent of D_R lower bound for the second part of Eq. (3) is needed. Since D_R contributes only to this portion of the average cost, the optimal classification rule D_R for a given stopping time R can then be derived. Specifically, for any classification rule D_R given stopping time R, $\sum_{j=1}^{L} (C_{Bj}\pi_R + C_{Nj}(1-\pi_R)) \mathbb{1}_{\{D_R=j\}} \geq g(\pi_R)$, where $g(\pi_R) \triangleq \min_{1 \leqslant j \leqslant L} \left[C_{\mathcal{B}j}\pi_R + C_{\mathcal{N}j}(1-\pi_R) \right]$. The optimal rule is thus defined as follows:

$$\mathcal{D}_R^{optimal} = \arg\min_{1 \leqslant j \leqslant L} \left[C_{\mathcal{B}j}\pi_R + C_{\mathcal{N}j}(1-\pi_R) \right]. \tag{4}$$

From Eq. (4), we deduce that $J(R, \mathcal{D}_R^{optimal}) \leq J(R, D_R)$ since the optimal classification rule results to the smallest average cost. Based on the last observation, Eq. (3) can be written as:

$$\tilde{J}_R \triangleq J(R, \mathcal{D}_R^{optimal}) = \min_{D_R} J(R, D_R) = \mathbb{E}\left\{ \sum_{n=1}^{R} c_n + g(\pi_R) \right\}, \tag{5}$$

which depends only on the stopping time R.

3.4 Stopping Rule

The solution for optimizing $\tilde{J}$ in Eq. (5) with respect to R can be determined by solving the optimization problem $\min_{R \geq 0} \tilde{J}_R = \min_{R \geq 0} \mathbb{E}\left\{ \sum_{n=1}^{R} c_n + g(\pi_R) \right\}$. This constitutes a classical problem in optimal stopping theory for Markov processes [32]. Since the stopping time R can take values in $\{0, 1, \ldots, K\}$, the optimum strategy will consist of a maximum of $K+1$ stages. In addition, Bellman's principle of optimality [2] states that the solution we seek must also be optimum, if instead of the first stage we start from any intermediate stage and continue toward the final stage. We derive our optimal stopping rule, $R^{optimal}$, as follows:

$$R^{optimal} = \min \left\{ 0 \leq n \leq K | S_n = \tilde{J}_n \right\}, \tag{6}$$

where for $n = K-1, \ldots, 0$, the n–th stage cost $S_n(\pi_n)$ is related to $S_{n+1}(\pi_{n+1})$ through the following recursion:

$$S_n(\pi_n) = \min \left[g(\pi_n), c_n + \int A_n(y_{n+1}) \times S_{n+1}\left(\frac{p(y_{n+1}|H_{\mathcal{B}})\pi_n}{A_n(y_{n+1})} \right) dy_{n+1} \right], \tag{7}$$

$A_n(y_{n+1}) \triangleq \pi_n p(y_{n+1}|H_{\mathcal{B}}) + (1-\pi_n) p(y_{n+1}|H_{\mathcal{N}})$ and $S_K(\pi_K) = g(\pi_K)$.

The optimal stopping rule derived by Eq. (7) has a very intuitive structure. *i.e.,* stop at the stage where the cost of stopping (the first expression in the minimization) is no greater than the expected cost of continuing given all information accumulated at the current stage (the second expression in the minimization).

3.5 Practical Considerations

In this section, we describe *CONcISE*, a novel algorithm for cyberbullying detection on Instagram media sessions based on Eq. (4) and Eq. (6). For each media session $s \in \mathcal{M}$, we maintain the number of aggressive comments identified at any given time along with the time of that classification. We use this information to decide when to raise an alert. Specifically, we set a threshold, and raise an alert the first time the number of comment–level detections surpasses that threshold. This design provides the flexibility of trading timeliness for false positives. To select the threshold t, we performed a grid search over values $\{2, 3, \ldots, 10\}$ and computed Area Under Receiver Operating Characteristic curves (AUC) scores of *CONcISE* as a function of t so as to optimize the precision–recall trade–off [3]. We found that best precision and recall is achieved when $t = 5$ (c.f. Section 4).

Within a given session s, the posterior probability π_0 of a given comment m is initially set to the prior probability p of a comment being an instance of aggression, and the two terms in Eq. (7) are compared. If the first term is less than the second, *CONcISE* classifies the comment based on the optimal rule of Eq. (4). Otherwise, the first feature is extracted from the comment and evaluated. *CONcISE* repeats these steps until either it decides to classify the given comment using $< K$ features, or all K features are examined.

4 EXPERIMENTAL EVALUATION

In this section, we provide a thorough experimental analysis to evaluate our proposed approach. All methods are implemented in Python, and all experiments were performed on a 64–bit machine with a dual–core Intel processor @2.7GHz and 16GB memory.

4.1 Baseline Methods

State–of–the–art Cyberbullying Detection Methods. We evaluate the effectiveness of *CONcISE* against the state–of–the–art for cyberbullying detection on Instagram, i.e., **DRFS [16]**, the best performing approach for cyberbullying detection on Twitter, **RF [6]** (i.e., **R**andom **F**orest classifier), adapted for Instagram with the best performing features used in [16], **DLR [27]** (i.e., **D**ynamic **L**ogistic **R**egression), the method for incremental cyberbullying detection on Vine [27], and **TM [7]**, a method that uses the entire history of all comments in a media session to derive a **T**emporal **M**odel for classification. Note that with the exception of [27], no research exists in the literature that addresses the challenge of timely cyberbullying detection by examining cyberbullying as a repetitive process.

Online Streaming Feature Selection Methods. Since our proposed approach, *CONcISE*, sequentially examines features one–by–one, as opposed to indiscriminately using a common set of features for all comments, we compare *CONcISE* with **SAOLA [40]** and **OFS–Density [46]**, i.e., the two best performing online streaming feature selection methods in terms of accuracy and scalability.

4.2 Dataset

We use the Instagram dataset collected by Hosseinmardi et al. [16]. The dataset has been originally collected using snowball sampling starting from a random seed node. For each user, all media the user shared, users who commented on the media, and the comments posted on the media have been collected. In total, the dataset contains 3,829,756 users and 9,828,760 comments. Of all media sessions which contain at least 40% profanities, 47.5% have been manually labeled as positive if "*there are negative words and/or comments with intent to harm someone or other, and the posts include two or more repeated negativity against a victim*" [16].

We augmented this dataset with $\sim 10K$ comment–level labels obtained from 10 experts. The comments span in total 22.1% of the media sessions with 40% or more profanity. We use this small subset of labeled comments for training, and the remaining unlabeled comments (i.e., 77.9% of labeled media sessions in the original dataset from [16]) for testing.

4.3 Offensive Language Dictionaries

For a fair comparison with **DRFS**, we use the exact same features (i.e., the counts of keywords 'ugly', 'shut', 'suck', 'gay', 'beautiful', 'sick', 'bitch', 'work', 'hate', and 'fuck') after removing punctuation marks and stopwords. To directly compare our approach with **DLR** [27], we consider the frequencies of 1,384 **profane** unigrams and bigrams [35]. We further consider a dictionary of 349 offensive unigrams and bigrams (**Noswearing**) [8] to examine the robustness of *CONcISE* on the dictionary used for detection. This results in three variants of *CONcISE*: **CONcISE–10**, **CONcISE–Profane**, and **CONcISE–Noswearing**, respectively, that use the corresponding dictionaries.

4.4 Comparison with Cyberbullying Detection State–of–the–art.

Table 1 summarizes the performance of *CONcISE* as compared to the state–of–the–art for cyberbullying detection, in terms of accuracy, recall and precision of the bullying class, area under the ROC curve (AUC), the average number (and standard deviation) of features used by our approach in detecting aggressive comments (similarly for the number of comments needed to detect a cyberbullying session). The best performing method is marked in bold.

Clearly, all three variants of *CONcISE* outperform the baselines, often by a considerable margin. At the same time, *CONcISE* significantly reduces both the average number of features used to classify individual comments as well as the number of comments examined before determining if a session is an instance of cyberbullying. Even though RDFS achieves the highest precision among all approaches, *CONcISE* consistently achieves the highest (or closely follows the highest) performance, with the additive advantage of using only a small percent of all features. Specifically, *CONcISE* uses up to 99.7% less features on average than the total number of features available (i.e., for Noswearing) to make a comment–level decision. Similarly, *CONcISE* uses at least 20.5% less comments on average than the best performing baseline (i.e., RF–comment with 33.07 comments on average) to make a decision at the media session level.

Table 1: Performance comparison of *CONcISE* with state–of–the–art cyberbullying detection methods.

Method	Accuracy	Recall	Precision	AUC	Avg. # of features (std.)	Avg. # of comments (std.)
RDFS	0.751	0.449	**0.858**	**0.877**	10 (0)	83
RF	0.791	0.703	0.749	0.862	13 (0)	83
DLR	0.497	0.651	0.616	0.521	4 (0)	35.90 (45.43)
DLR–10	0.733	0.463	0.760	0.684	10 (0)	35.37 (36.54)
DLR–Profane	0.522	0.644	0.638	0.541	1,384 (0)	47.70 (53.71)
DLR–Noswearing	0.762	0.598	0.743	0.733	349 (0)	45.26 (32.49)
TM	0.749	0.783	0.649	0.816	115(0)	83(0)
CONcISE–10	0.783	**0.794**	0.695	0.864	**2.97 (1.17)**	**26.68 (16.92)**
CONcISE–Profane	**0.806**	0.769	0.745	0.860	3.76 (3.19)	30.62 (21.82)
CONcISE–Noswearing	**0.806**	0.776	0.742	0.862	3.92 (2.16)	29.75 (21.09)

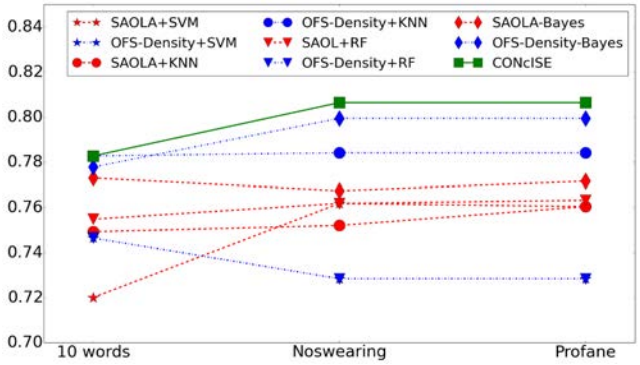


Figure 3: Accuracy comparison of *CONcISE*, SAOLA, and OFS–Density.

4.5 Comparison with State–of–the–art OSFS.

Next, we compare *CONcISE* to the baselines in terms of feature selection quality, over the feature space defined by the corresponding disctionary used, i.e., 10 unigrams, Noswearing and Profane dictionaries respectively. As SAOLA and OFS-Density do not perform classification in conjuction to feature selection, we use KNN, SVM, Random Forest and Standard Bayes for both baselines; KNN and SVM have previously been shown to achieve the highest classification performance in [40, 46]. We report the accuracy of *CONcISE*, as well as all combinations of baselines and classifiers. Figure 3 shows that *CONcISE* consistently outperforms OFS-Density and SAOLA over all 3 feature spaces.

4.6 Timeliness

To compare the best performing variance of *CONcISE*, namely, *CONcISE–Noswearing*, against baselines with respect to *timeliness*, we introduce a measure "# of saved comments" (as illustrated in Figure 4), defined as $i - j$ where i and j, respectively, denotes the number of comments that *CONcISE* and a given baseline have to inspect before raising an alert.

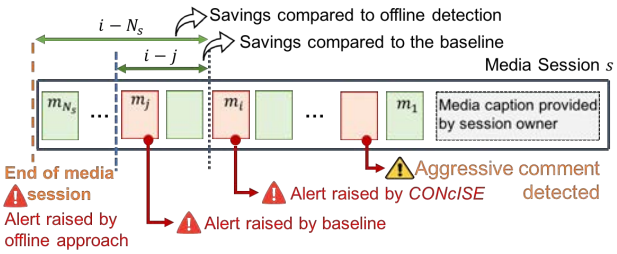


Figure 4: Illustration of timeliness, i.e., the number of comments "saved".

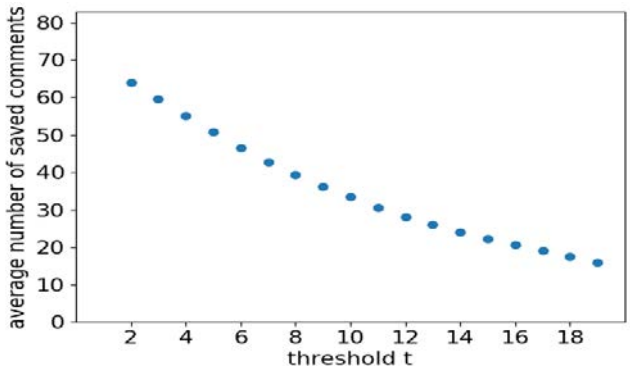


Figure 5: Average number of saved comments with respect to RDFS (a static method) as the threshold for comments detected as harassment content used to raise an alert increases.

By this definition, we respectively compare the timeliness of *CONcISE–Noswearing* to the best performing offline (i.e., RDFS) and online approaches (i.e., DLR–Noswearing), with respect to accuracy for varying threshold t. Figure 5 shows the absolute number of comments "saved" by CONcISE as compared to RDFS. As expected, the higher the threshold the more comments must be marked for harassment before an alert can be raised. Hence, the higher the threshold, the less the savings. Figure 6 shows the distribution of density of the number of saved comments as compared to DLR–Noswearing and provides strong evidence that *CONcISE–Noswearing* can detect cyberbullying media sessions faster than DLR–Noswearing (in addition to being more accurate) for all thresholds considered.

4.7 Scalability Analysis

We found baselines RDFS, RF, and TM to be the fastest among all methods with respect to runtime for session–level decisions, for an average of 0.0021, 0.0011, and 0.002 seconds respectively in making a session level decision. However, such "speed" is meaningless if considered in isolation to timeliness. Both RDFS, RF, and TM require all comments of a media session to be available for classification, making them offline for all practical purposes. The rest of the methods achieve similar runtime (s), ranging from 0.0077 to 0.0153 second per media session, with *CONcISE–10* being the fastest. Even though the difference may seem negligible for this dataset, considering a real–world scenario where a million media sessions are to be evaluated in real–time.

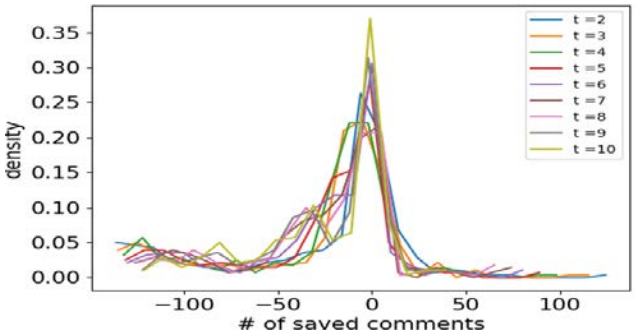


Figure 6: Distribution of average number of saved comments with respect to DLR–Noswearing for various thresholds.

5 CONCLUSION

Contributions. In this paper, we have proposed a novel approach for online cyberbullying detection on Instagram which can achieve *timely* and *accurate* detection while being *scalable* to the staggering rates at which content is generated. To achieve this goal, our two–stage approach begins by classifying comments in a media session as they become available, and an alert at the session level is raised when a threshold is exceeded. Our sequential hypothesis testing formulation drastically reduces the number of features used for classification, while maintaining high accuracy. Extensive evaluation experiments using a large–scale, real–world dataset from Instagram demonstrate the effectiveness of our proposed approach with respect to accuracy, timeliness, efficiency, and robustness, and show that it consistently outperforms the state–of–the–art, often by a considerable margin.

Limitations. Next, we would like to note the limitations of our present work that point to interesting future research directions. For one, our experimental evaluation is limited to a single dataset and therefore the performance of our approach should not be generalized to other platforms. Additionally, the inability to validate the labels in the dataset provided by [16] prevents us from obtaining a more granular understanding of the effect of false positives (similarly false negatives) on the accuracy of our approach. More importantly, comment–level labels, even though imperative for capturing the repetitive nature of cyberbullying as a process over a period of time, can be costly or time consuming to obtain.

Future Directions. There are several opportunities to build on this work. First, we plan to explore features such as user– and network–information and the sequence of conversations to further improve classification accuracy. We are also planning to evaluate the performance of our approach on additional datasets from diverse platforms including Ask.fm and Twitter, which are reported to be key social networking venues where users frequently become victims of cyberbullying. Given the broad definition of cyberbullying, we may also design detection strategies grounded in a more nuanced, multi–dimensional representation of repetitive harassment instead of striving for a global and/or simplified indicator of cyberbullying.

REFERENCES

[1] Mohammed Ali Al-garadi, Kasturi Dewi Varathan, and Sri Devi Ravana. 2016. Cybercrime detection in online communications: The experimental case of cyberbullying detection in the Twitter network. *Computers in Human Behavior* 63 (2016), 433–443.

[2] D. P. Bertsekas. 2005. *Dynamic Programming and Optimal Control*. Vol. 1. Athena Scientific.

[3] Michael Buckland and Fredric Gey. 1994. The relationship between recall and precision. *Journal of the American society for information science* 45, 1 (1994), 12–19.

[4] Jiuwen Cao, Tao Chen, and Jiayuan Fan. 2014. Fast online learning algorithm for landmark recognition based on BoW framework. In *Industrial Electronics and Applications (ICIEA), 2014 IEEE 9th Conference on*. IEEE, 1163–1168.

[5] Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, and Athena Vakali. 2017. Detecting Aggressors and Bullies on Twitter. In *Proceedings of the 26th International Conference on World Wide Web Companion*. 767–768.

[6] Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, and Athena Vakali. 2017. Mean birds: Detecting aggression and bullying on twitter. In *Proceedings of the 2017 ACM on Web Science Conference*. ACM, 13–22.

[7] Vivek Singh Devin Soni. [n. d.]. Time Reveals AllWounds: Modeling Temporal Dynamics of Cyberbullying Sessions. In *Proceedings of the Twelfth International AAAI Conference on Web and Social Media (ICWSM 2018)*.

[8] AllSlang family. [n. d.]. Internet Slang Swear Word List & Curse Filter. https://www.noswearing.com/dictionary.

[9] Sujatha Das Gollapalli, Cornelia Caragea, Prasenjit Mitra, and C Lee Giles. 2013. Researcher homepage classification using unlabeled data. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 471–482.

[10] Leam Hackett. 2017. The Annual Bullying Survey 2017. https://www.ditchthelabel.org/wp-content/uploads/2017/07/The-Annual-Bullying-Survey-2017-1.pdf. (accessed on Aug. 30 2018).

[11] M. A. Hall. 1999. *Correlation-based feature selection for machine learning*. Ph.D. Dissertation. The University of Waikato.

[12] Sameer Hinduja and Justin W Patchin. 2007. Offline consequences of online victimization: School violence and delinquency. *Journal of school violence* 6, 3 (2007), 89–112.

[13] Dianne L Hoff and Sidney N Mitchell. 2009. Cyberbullying: Causes, effects, and remedies. *Journal of Educational Administration* 47, 5 (2009), 652–665.

[14] Steven CH Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. 2018. Online Learning: A Comprehensive Survey. *arXiv preprint arXiv:1802.02871* (2018).

[15] Steven CH Hoi, Jialei Wang, Peilin Zhao, and Rong Jin. 2012. Online feature selection for mining big data. In *Proceedings of the 1st international workshop on big data, streams and heterogeneous source mining: Algorithms, systems, programming models and applications*. ACM, 93–100.

[16] Homa Hosseinmardi, Rahat Ibn Rafiq, Richard Han, Qin Lv, and Shivakant Mishra. 2016. Prediction of cyberbullying incidents in a media-based social network. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. 186–192.

[17] Guichun Hua, Min Zhang, Yiqun Liu, Shaoping Ma, and Liyun Ru. 2010. Hierarchical feature selection for ranking. In *Proceedings of the 19th international conference on world wide web*. ACM, 1113–1114.

[18] Hao Huang, Shinjae Yoo, and Shiva Prasad Kasiviswanathan. 2015. Unsupervised feature selection on data streams. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1031–1040.

[19] Jyrki Kivinen, Alexander J Smola, and Robert C Williamson. 2004. Online learning with kernels. *IEEE transactions on signal processing* 52, 8 (2004), 2165–2176.

[20] Robin M Kowalski and Susan P Limber. 2013. Psychological, physical, and academic correlates of cyberbullying and traditional bullying. *Journal of Adolescent Health* 53, 1 (2013), S13–S20.

[21] Haiguang Li, Xindong Wu, Zhao Li, and Wei Ding. 2013. Group feature selection with streaming features. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE, 1109–1114.

[22] Jiguang Liang, Xiaofei Zhou, Li Guo, and Shuo Bai. 2015. Feature selection for sentiment classification using matrix factorization. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 63–64.

[23] T. Marill and D. Green. 1963. On the effectiveness of receptors in recognition systems. *IEEE Transactions on Information Theory* 9, 1 (1963), 11–17.

[24] Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 145–153.

[25] Guansong Pang, Longbing Cao, Ling Chen, and Huan Liu. 2016. Unsupervised feature selection for outlier detection by modelling hierarchical value-feature couplings. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 410–419.

[26] Simon Perkins and James Theiler. 2003. Online feature selection using grafting. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 592–599.

[27] Rahat Ibn Rafiq, Homa Hosseinmardi, Richard Han, Qin Lv, and Shivakant Mishra. 2018. Scalable and timely detection of cyberbullying in online social networks. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*. ACM, 1738–1747.

[28] Rahat Ibn Rafiq, Homa Hosseinmardi, Richard Han, Qin Lv, Shivakant Mishra, and Sabrina Arredondo Mattson. 2015. Careful what you share in six seconds: Detecting cyberbullying instances in Vine. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*. ACM, 617–622.

[29] Elaheh Raisi and Bert Huang. 2017. Cyberbullying detection with weakly supervised machine learning. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*. ACM, 409–416.

[30] Elaheh Raisi and Bert Huang. 2018. Weakly Supervised Cyberbullying Detection Using Co-Trained Ensembles of Embedding Models. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 479–486.

[31] Weixiang Shao, Lifang He, Chun-Ta Lu, Xiaokai Wei, and S Yu Philip. 2016. Online unsupervised multi-view feature selection. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 1203–1208.

[32] Albert N Shiryaev. 2007. *Optimal Stopping Rules*. Vol. 8. Springer Science & Business Media.

[33] Mifta Sintaha, Shahed Bin Satter, Niamat Zawad, Chaity Swarnaker, and Ahanaf Hassan. 2016. *Cyberbullying detection using sentiment analysis in social media*. Ph.D. Dissertation. BRAC University.

[34] Peter K Smith, Jess Mahdavi, Manuel Carvalho, and Neil Tippett. 2006. An investigation into cyberbullying, its forms, awareness and impact, and the relationship between age and gender in cyberbullying. *Research Brief No. RBX03-06. London: DfES* (2006).

[35] Luis von Ahn. [n. d.]. Offensive/Profane Word List. https://www.cs.cmu.edu/~biglou/resources/bad-words.txt.

[36] Jialei Wang, Peilin Zhao, and Steven CH Hoi. 2016. Soft confidence-weighted learning. *ACM Transactions on Intelligent Systems and Technology (TIST)* 8, 1 (2016), 15.

[37] Jialei Wang, Peilin Zhao, Steven CH Hoi, and Rong Jin. 2014. Online feature selection and its applications. *IEEE Transactions on Knowledge and Data Engineering* 26, 3 (2014), 698–710.

[38] Xindong Wu, Kui Yu, Hao Wang, and Wei Ding. 2010. Online streaming feature selection. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. Citeseer, 1159–1166.

[39] Mengfan Yao, Charalampos Chelmis, and Daphney-Stavroula Zois. 2018. Cyberbullying Detection on Instagram with Optimal Online Feature Selection. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 401–408.

[40] Kui Yu, Xindong Wu, Wei Ding, and Jian Pei. 2016. Scalable and accurate online feature selection for big data. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 2 (2016), 16.

[41] Aonan Zhang, Jun Zhu, and Bo Zhang. 2013. Sparse online topic models. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1489–1500.

[42] Liang Zhang, Jie Yang, and Belle Tseng. 2012. Online modeling of proactive moderation system for auction fraud detection. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 669–678.

[43] Xiang Zhang, Jonathan Tong, Nishant Vishwamitra, Elizabeth Whittaker, Joseph P Mazer, Robin Kowalski, Hongxin Hu, Feng Luo, Jamie Macbeth, and Edward Dillon. 2016. Cyberbullying Detection with a Pronunciation Based Convolutional Neural Network. In *15th IEEE International Conference onMachine Learning and Applications (ICMLA)*. 740–745.

[44] Rui Zhao and Kezhi Mao. 2017. Cyberbullying Detection based on Semantic-Enhanced Marginalized Denoising Auto-Encoder. *IEEE Transactions on Affective Computing* 8, 3 (2017), 328–339.

[45] Haoti Zhong, Hao Li, Anna Cinzia Squicciarini, Sarah Michele Rajtmajer, Christopher Griffin, David J Miller, and Cornelia Caragea. 2016. Content-Driven Detection of Cyberbullying on the Instagram Social Network. In *IJCAI*. 3952–3958.

[46] Peng Zhou, Xuegang Hu, Peipei Li, and Xindong Wu. 2019. OFS-Density: A novel online streaming feature selection method. *Pattern Recognition* 86 (2019), 48–61.

Enhancing Fashion Recommendation with Visual Compatibility Relationship

Ruiping Yin
Beijing Institute of Technology
School of Computer Science and Technology
Beijing, China
University of Technology Sydney
Centre for Artificial Intelligence
Sydney, Australia
yrp@bit.edu.cn

Kan Li
Beijing Institute of Technology
School of Computer Science and Technology
Beijing, China
likan@bit.edu.cn

Jie Lu
University of Technology Sydney
Centre for Artificial Intelligence
Sydney, Australia
Jie.Lu@uts.edu.au

Guangquan Zhang
University of Technology Sydney
Centre for Artificial Intelligence
Sydney, Australia
Guangquan.Zhang@uts.edu.au

ABSTRACT

With the increasing of online shopping services, fashion recommendation plays an important role in daily online shopping scenes. A lot of recommender systems have been developed with visual information. However, few works take into account compatibility relationship when they are generating recommendations. The challenge is that fashion concept is often subtle and subjective for different customers. In this paper, we propose a fashion compatibility knowledge learning method that incorporates visual compatibility relationships as well as style information. We also propose a fashion recommendation method with domain adaptation strategy to alleviate the distribution gap between the items in target domain and the items of external compatible outfits. Our results indicate that the proposed method is capable of learning visual compatibility knowledge and outperforms all the baselines.

CCS CONCEPTS

• **Information systems** → **World Wide Web**; *Web interfaces*; Web log analysis.

KEYWORDS

Fashion Recommendation; Viusal Compatibility; Image Representation

ACM Reference Format:
Ruiping Yin, Kan Li, Jie Lu, and Guangquan Zhang. 2019. Enhancing Fashion Recommendation with Visual Compatibility Relationship. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313739

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313739

1 INTRODUCTION

A large portion of sales in the e-commerce are affected by fashion and lifestyle, which constitute apparel, footwear, bags, accessories, etc. Intelligent fashion recommendation received a lot of attention in computer vision and machine learning community [3, 9, 22]. They have huge potential profits for the fashion industry. A lot of companies have established their own recommender systems to give users advice to enhance their shopping experience, such as Amazon, Alibaba and eBay [14, 23].

Many approaches have been proposed to analyze user preferences on fashion criterion and generate personalized recommendation. Most of fashion recommendation approaches take into account characteristics of image, as visual information is one of the most important factor in describing fashion items [5, 9, 13]. Such approaches can substantially improve recommendation accuracy than others without visual information.

However, a few of them considers the problem of compatibility of fashion items. We know that when we choose a piece of clothing, it is not just a matter of considering the style of the dress. We also need to consider its effect with other clothes we wear. Some examples of compatibility and incompatibility outfits have been shown in Figure 1. Normally one would not pair a red T-shirt with green pants. Moreover, a black robe is incompatible with a pair of pink running shoes. This is partially because it is difficult to model compatibility relationship between fashion items.

When designing this recommendation system, we mainly consider the problem of learning visual compatibility relationship of items on pixel level. The visual compatibility relationship that needs to be learnt is whether fashion matching between one item and another item conforms to the human aesthetic by understanding the picture. In the traditional fashion recommendation, authors often only consider the styles and categories of clothes and ignore the sense of harmony between items as an outfit. In this paper, our approach considers visual compatibility relationship to recommend fashion items, which is closer to the actual needs of people.



(a) compatible outfits.



(b) incompatible outfits.

Figure 1: Examples of compatible and incompatible outfits.

In order to learn the knowledge of the matching model between fashion items, we face two challenges: 1) How to learn the common domain knowledge about fashion compatibility relationship between items. In other words, there are a few of outfits that we observed on the online shopping website for a single person and fashion concept is often subtle and subjective for different customers. 2) How to incorporate the learnt domain knowledge into our recommender system. For the first challenge, we propose a novel method to incorporate the compatibility relationship knowledge into the image representation. Our method allows learning an embedding from the images of the fashion items to a latent space, so that two items that is a good match are close in this latent space and items that don't match are far apart. An external dataset which contains a number of outfits being given by experts is also been used to train our model. For the second challenge, we adjust the popular Bayesian personalized ranking (BPR) [17] model to include the compatibility relationship knowledge that we learnt. Moreover, because we use the external dataset to learn the domain knowledge between the items in order to solve the problem of the distribution gap between the source domain and target domain, we propose a domain adaptation method to alleviate this difference.

In this paper, our contributions are as follows:

- We propose a fashion compatibility relationship learning method that incorporates visual compatibility relationships as well as style information into a visual embedding.
- We propose a fashion recommendation method with domain adaptation strategy to alleviate the distribution gap between the items in target domain and the items of external compatible outfits.
- We conduct a case study to illustrate how our method understands images. Furthermore, through an extensive set of experiments on several datasets, we demonstrate our method significantly outperforms several alternative methods.

The rest of this paper is organized as follows. In Section 2, we review some of the relevant methods. Section 3 introduces some notations and presents our approach in detail. The experiments and results analysis are demonstrated in Section 4. Lastly, conclusions and future work are discussed in Section 5.

2 RELATED WORK

In this section, we first review current approaches about visual fashion compatibility learning. Then we have a survey about fashion recommendation.

2.1 Visual Fashion Compatibility Learning

Visual compatibility measures whether fashion items complement one another across visual categories [1, 12, 20, 24]. For example, in Figure 1, a brown jacket is more compatible with a black casual pants. Oramas and Tuytelaars [15] introduced a hierarchical method to model visual compatibility by discovering mid-level visual elements. Li et al. [10] incorporated the appearances and metadata into their automatic composition system using an end-to-end deep neural network. Veit et al. [21] proposed a learning framework to recover a style space for fashion items from co-occurrence information and category labels which is able to learns compatibility between items. Han et al. [4] jointly train a Bi-LSTM model and a visual-sematic embedding for fashion compatibility learning. All of these methods are designed to generate outfits in which each item belongs to different categories. However, on one hand, category information is not accessible in many situations. On the other hand, these methods require a prior knowledge about which two categories can be put together. In contrast to these approaches, our method learns the compatibility relationships without a strict constraint of categories.

2.2 Fashion Recommendation

As mentioned above, there are a few approaches for recommending fashion items. One of the most famous method is visual Bayesian personalized ranking (VBPR) which learns user visual preference from implicit feedback [5]. The preference predictor in VBPR can be formulated as follows:

$$\hat{r}_{u,i} = \alpha + \beta_u + \beta_i + \gamma_u^T \gamma_i + \theta_u^T \theta_i \tag{1}$$

where α is the global offset, β_u and β_i are user/item bias, and γ_u and γ_i are latent feature vectors describing user u and item i, respectively, θ_u and θ_i are visual feature vectors of user u and item i. The inner product $\gamma_u^T \gamma_i$ and $\theta_u^T \theta_i$ indicates the scores that user u assigned to item i in terms of latent aspect and visual aspect.

This work improves the performance of recommender system significantly. The reason is that it incorporates visual information into the preference prediction model. However, in this approach, the authors did not consider the compatibility relationship between fashion items.

There are also other approaches which concentrate on fashion recommendation [2, 7, 8, 18]. Hu et al. [6] proposed a functional tensor factorization method to recommend outfits to users by learning from the interactions between user and fashion items. Packer et al. [16] learnt an interpretable image representation and modelled the dynamics of personalized users' visual preference to generate clothing recommendation. Liu et al. [11] separated style and category information from the image representation using a multilayer

Table 1: Major Notations Used in This Paper

Notation	Description
$\mathcal{U}, \mathcal{I}$	user set, item set
$\mathcal{V}$	image set of items
R	implicit feedback matrix
X	the set of images in clothing collocation dataset
C	clothing collocation pair list
$\mathcal{P}_u, \mathcal{V}_u, T_u$	positive item set of user u in training/validation/test sets
f_i	image representation vector of image i
γ_u, γ_i	latent features of user u, item i
θ_u, θ_i	visual features of user u, item i
W_{enc}, W_{dec}	weights of encoders and decoders, respectively

perceptron. However, all these methods only considered visual information and co-occurrence information from the online shopping website. It cannot sufficient learn compatibility relationship knowledge. In this paper, we utilize an external expert fashion collection dataset which contains a lot of good matches and propose a domain adaptation method to transfer the knowledge from the external dataset to our recommendation scene.

3 NOTATIONS AND PROBLEM FORMULATION

We first explain the symbols used in the next paper. We will consider $\mathcal{U}$ be the set of all users and $\mathcal{I}$ the set of all items. For each item, we have a corresponding images $\mathcal{V}_i$ which can represent this product. $R \subseteq len(\mathcal{U}) \times len(\mathcal{V})$ is the implicit feedback matrix whose rows correspond to customers and whose columns correspond to products. This means that $R_{ui} = 1$ stands for user u has bought item i, and 0 otherwise. We also have a hand-crafted clothing collocation dataset $X = \{x_1, x_2, ..., x_n\}$ labelled by experts which contains a set of clothing images and a clothing collocation pair list $C = \{(x_i, x_j)|x_i, x_j \in X\}$. Note that despite in the item set $\mathcal{V}$ and clothing collocation set X are both contains clothing images, the images in this two datasets are differences and no overlapped items. Furthermore, we can incorporate additional information like category data of products or demographic data about customers. However, we just focus on sales and visual information which is very important in fashion recommendation. Table 1 lists the major notations used throughout this paper.

The fashion recommendation task with visual compatibility relationship to be solved in this paper is to provide a personalized ranking list to each user with the help of visual information. First, given a set of fashion items $X = \{x_1, x_2, ...\}$ and collocations using these items C, learning visual compatibility knowledge is to learn an embedding $\mathcal{F}$ where the distance between item i and j, $d(\mathcal{F}(x_i), \mathcal{F}(x_i))$, is as small as possible if $(x_i, x_j) \in C$. After that, with user interaction records and item images, we try to learn the user's preference towards collaborative information and visual information to generate a ranking list for each user.

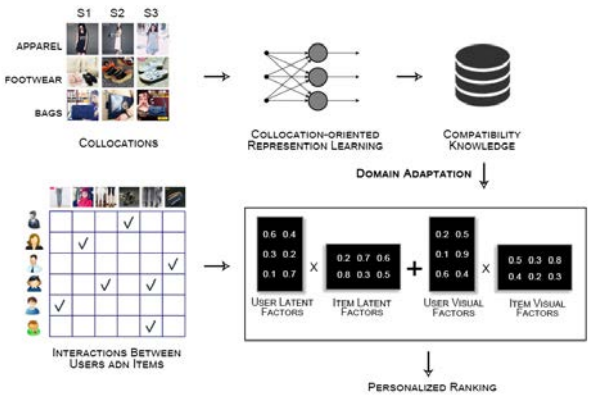


Figure 2: The proposed framework of our method.

4 ENHANCING FASHION RECOMMENDATION WITH VISUAL COMPATIBILITY RELATIONSHIP

We propose a fashion recommendation method which considers compatibility relationship between fashion items. In this method, we combine the collaborative information among users and items with compatibility knowledge. In order to allow the algorithm to understand the aesthetics of humans, we carefully construct an image representation model, through which we can determine what kind of information the resulting representation contains. This makes the computer learn the compatibility knowledge that people understand. After that, we incorporate the generated compatibility knowledge into our recommendation framework with a domain adaptation strategy. The framework of the entire recommender system is shown in Figure 2.

4.1 Learning visual compatibility knowledge from fashion items

In this section, our goal is to learn the visual compatibility relationships between fashion items. Conventional methods are mostly relying on category information to learn image representations. Instead of annotating images with labels or categories, which is costly, we leverage the weakly-labeled web data provided by the external dataset to learn compatibility knowledge.

Since there is no fixed category for the tasks we are going to perform, we cannot use the softmax-based cross entropy loss function for training. So, we chose triplet network to learn the image representation. The advantage of the triplet network is the distinction of details, that is, when the two inputs are similar, the triplet network can better model the details, which is equivalent to adding two measures of the difference of the input differences to learn a better representation of the input. The structure of the network is shown in the Figure 3.

In our task, we take the first item in an item pair in list C as an anchor, the second item as a positive sample, and select an item that is not in the list as the negative sample. More specifically, we can't randomly select negative samples on the entire candidate set, because this will cause $d(A, N)$ to be much larger than $d(A, P)$,

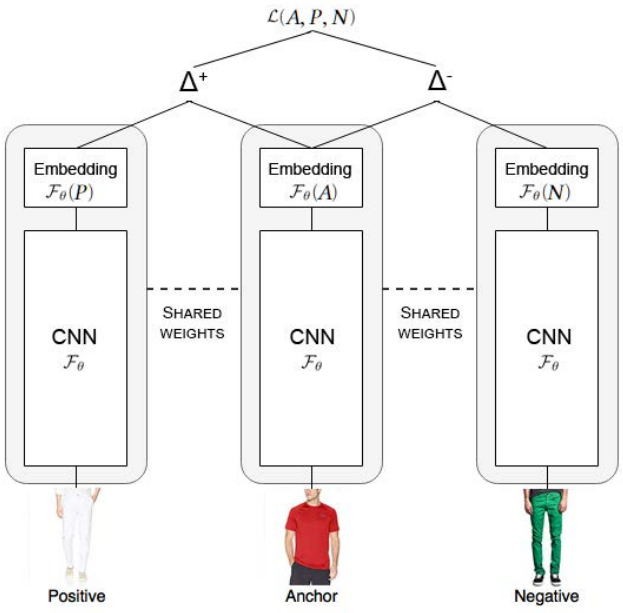


Figure 3: The illustration of our model for learning compatibility knowledge.

which will make the model unable to fully train and enter the prematurely. The state of the joint. So at each training, we need to choose a negative sample with $d(A, P)$ as close as possible to $d(A, N)$. This may make the model as difficult as possible to reduce the risk of overfitting.

Given a fashion image set, $X = \{x_1, x_2, ...\}$, x_i is a picture containing the t-th item. Also give a list $C = \{(x_i, x_j)|x_i \in X\}$ which denotes clothing collocation pairs. We need to learn a x_i to f_i mapping function, so that x_i and x_j are as close as possible, if the two items have good compatibility, and by contrast, the distance between f_i and f_j is as far as possible is i and j are not good compatibility. In other words, we try to learn a new representation of images. More formally, we minimize the following objective function:

$$\mathcal{L}(A, P, N) = max(||\mathcal{F}(A) - \mathcal{F}(P)||_2 - ||\mathcal{F}(A) - \mathcal{F}(N)||_2 + \alpha, 0) \quad (2)$$

where $\mathcal{F}$ is the mapping function. A is the anchor item, P is the item which have good compatibility with A, and N is not a good compatibility. α is the threshold parameter.

As shown in Figure 3, the network contains three sub-networks which shares weights with each other. In each sub-network, we encode the item image with Convolutional Neural Networks. There are many ConvNets architectures to choose from, and we use a variant of AlexNet for simplicity. The AlexNet variant, is the same as the original, except that we do not use pretrained weights and we replace local response normalization with batch normalization. We use the output of the fc6 layer as the encoding feature of the input image. The dimension of the image encoding is 4096.

Using more powerful architectures (e.g. [19]), may achieve better performance; however, we found that AlexNet is sufficient to show the effectiveness of our method. Because this network has fewer parameters, it can be trained more easily and reduces the risk of overfitting during training.

4.2 Fashion recommender system with visual compatibility knowledge

As mentioned above, the recommendation task can be regard as a ranking problem according to the user's preference. Our preference predictor is based on the basis of Matrix Factorization, which is the most promising model for rating prediction as well as modeling implicit feedback. The most related work to this problem is the VBPR model proposed in [5], which learns the visual user preference predictor using a pairwise ranking optimization framework.

We conduct our model based on pairwise learning. We defined the preference predictor as same as VBPR except for the reduce dimension method. To avoid missing information, we use an autoencoder to process dimension reducing. The formulation of encoder and decoder of autoencoder are as follows:

$$Enc(\mathcal{I}) = W_{enc} \cdot \mathcal{F}(\mathcal{I}) + b_{enc} \quad (3)$$

$$Dec(\mathcal{I}) = W_{dec} \cdot Enc(\mathcal{I}) + b_{dec} \quad (4)$$

Thus, the final preference predictor are as follows:

$$\hat{r}_{u,i} = \alpha + \beta_u + \beta_i + \gamma_u^T \gamma_i + \theta_u^T Enc(f_i) \quad (5)$$

For this implicit feedback ranking problem, we conduct the pairwise ranking optimization framework to train the model. The objective is as follows:

$$\max_\theta \sum_{(u,i,j) \in D_S} \ln \sigma\left(\hat{r}_{uij}\right) - \lambda_\Theta ||\Theta||^2 + ||f_i - \hat{f}_i||^2 + ||f_j - \hat{f}_j||^2 \quad (6)$$

where $\hat{r}_{uij} = \hat{r}_{u,i} - \hat{r}_{u,j}$.

Because the compatibility knowledge is learned from an external dataset also called source domain, images from the external dataset and the target dataset belong to different feature space. For example, in our experiments, the backgrounds of images in the source domain are very excursive. But in the target domain, the backgrounds are very clean and neat. Thus, we propose a domain adaption method which uses the domain adaptation technique to ensure that knowledge extracted from the source domain is consistent with the target domain and that knowledge transfer is positive.

When two items are bought by a customer at the same time, it regards as a co-occurrence pair. Most of the time, in the domain of fashion recommendation, we can assumption that the co-occurrence items should be a good clothing matching. Thus, we add the co-occurrence similarity into the objective above as follows to alleviate the distribution gap between source domain and target domain:

$$D_{i,j} = Frequent_{i,j} * ||Enc(f_i - f_j)|| \quad (7)$$

where $Frequent_{i,j}$ is the frequency of co-occurrence in the train set. When item i and item j are bought in a same bundle, we assume

that they are a good match, and the distance between them should be very closer.

In the training procedure, the training set D_S consists of triples in the form of (u, i, j, c), where u denotes the user and item i which they expressed positive feedback, and a non-observed item j. c is the co-occurrence frequency. It can be formalized by:

$$D_S = (u, i, j, c)|i \in I_u^+ \wedge j \in I_u^+ \tag{8}$$

The final formulation is as follows:

$$\sum_{(u,i,j)\in D_S} \ln \sigma\left(\hat{r}_{uij}\right) - \lambda_\Theta \|\Theta\|^2 + ||f_i - \hat{f}_i|| + ||f_j - \hat{f}_j|| + D_{i,j} \tag{9}$$

5 EXPERIMENTS AND ANALYSIS

We perform experiments on several datasets to evaluate the performance of the proposed method. All experiments were conducted on a workstation with a 6-core Intel CPU and two Titan-X (Pascal) graphics cards. Although there is a huge number of images and transaction records, it is still possible to train our model in half of day.

5.1 Datasets and evaluation metrics

The first dataset was provided by *Taobao.com* which is one of the most famous Chinese website for online shopping. It consists of clothing collocation suggestions from fashion experts, image data of Taobao items, and user behavior data. In this dataset, each line represents an item list which delimited by semicolon, every semicolon refers to a Collocation set. Every collocation set includes several goods, delimited by comma. We formatted this dataset into a pair-wise format, which means these two items is a good matching.

Another group of datasets contains user transaction records from two different sources. The first one were introduced in [5] and consist of reviews of clothing items crawled from *Amazon.com*. It was separated into 4 subcategories, named *Amazon Fashion*, *Amazon Women* and *Amazon Men*. The other one was crawled from *Tradesy.com*, which includes serveral kinds of feedback, like clicks, purchases, sales, etc.

The statistical information for the four datasets is provided in Table 2.

Table 2: Dataset statistics

Dataset	Users	Items	Interactions
Amazon Fashion	64,583	234,892	513,367
Amazon Women	97,678	347,591	827,678
Amazon Men	34,244	110,636	254,870
Tradesy.com	33,864	326,393	655,409

We measure recommendation performance of our method by calculating AUC and diversity. The AUC measures the quality of a ranking based on pairwise comparisons. Formally, we have

$$AUC = \frac{1}{|\mathcal{U}|} \sum_{u\in\mathcal{U}} \frac{1}{|\mathcal{D}_u|} \sum_{(i,j)\in\mathcal{D}_u} \xi\left(r_{u,i} > r_{u,j}\right) \tag{10}$$

where $\mathcal{D}_u = \{(i,j)|(u,i) \in \mathcal{T}_u \wedge (u,j) \notin (\mathcal{P}_u \cup \mathcal{V}_u \cup T_u)\}$. In other words, we are calculating the fraction of times that the 'observed' items i are preferred over 'non-observed' item j.

AUC is a typical metric to assess the recommendation performance in reproducing known user opinions that have been removed from the test dataset. The risk of such a metric is that, with recommendations based on similarity and overlap, customers will be exposed to a narrowing band of popular commodities. In other words, we also need other metrics to evaluate the recommendation performance.

Personalization, also named Inter-user diversity, considers the uniqueness of different customers' recommendation list. Given two users i and j, the different between their recommendation lists can be measured by the inter-list distance,

$$h_{ij}(L) = 1 - \frac{q_{ij}(L)}{L} \tag{11}$$

where $q_{ij}(L)$ is the number of common items in the top L places of the both lists: if the two lists are identical, $q_{ij}(L) = 0$ will equals 0 whereas completely different lists have $q_{ij}(L) = 1$.

Averaging $h_{ij}(L)$ over all pairs of users we obtain the mean distance $h(L)$, for which greater or lesser values mean, respectively, greater or lesser personalization of users' recommendation lists.

5.2 Experimental settings and baselines

We compared the proposed method in terms of accuracy and diversity against the following baselines:

- PopRank: Always recommends the top-k most popular items to users.
- BPR-MF (2009): This is a content-free algorithm based on matrix factorization which is designed for top-k recommendation tasks [17]. It optimizes pair-wise preferences between observed and unobserved items.
- VBPR (2016): It is a state-of-the-art image-based recommender system proposed by [5] for implicit feedback. The authors incorporate visual information provided by a pre-trained CNN.
- DVBPR (2017): This is the extension of VBPR by learning 'fashion aware' image representations directly [9].
- CO-BPR: Our method proposed in this paper.

We carefully choose the hyper-parameters and tuned them via grid search for each baseline method. For BPR-MF, VBPR, DVBPR and CO-BPR, we used a mini-batch size of 32 for all experiments. The number of latent factor selected from {6, 8, 10, 12, 14, 16, 18, 20}. We set it to be 12 in all experiments.

5.3 Results

We evaluated our proposed method by comparing it to state-of-the-art methods using some real-world datasets. We report recommendation performance in terms of the AUC and diversity in Table 3. Data with three sparsity ratios in target domain are chosen as training set. Comparing all the methods on these four datasets, we make the following observations:

Table 3: Recommendation Performance in Terms of AUC and Diversity with different sparsity

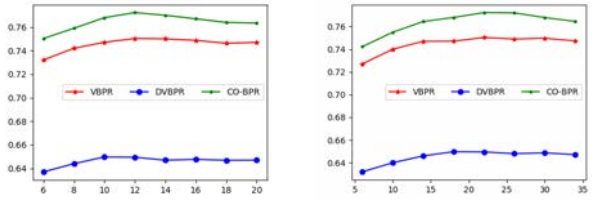
		AUC			Diversity		
		D=0.0010	D = 0.0005	D = 0.0001	D=0.0010	D = 0.0005	D = 0.0001
Amazon Fashion	*POPRANK*	0.5553	0.5627	0.6298	0.0000	0.0000	0.0000
	BPR-MF	0.5866	0.5951	0.6163	0.5621	0.5285	0.3093
	VBPR	0.6953	0.7116	0.7503	0.9715	0.9861	**0.9945**
	DVBPR	0.6134	0.6199	0.6497	**0.9826**	**0.9920**	0.9856
	CO-BPR	**0.7126**	**0.7242**	**0.7723**	0.9806	0.9891	0.9926
Amazon Women	*POPRANK*	0.5534	0.5897	0.6426	0.0000	0.0000	0.0000
	BPR-MF	0.5884	0.6176	0.6437	0.6335	0.5548	0.3786
	VBPR	0.6747	0.6861	0.7161	0.9777	0.9868	**0.9935**
	DVBPR	0.6209	0.6305	0.6714	0.9859	0.9890	0.9871
	CO-BPR	**0.6792**	**0.6901**	**0.7295**	**0.9952**	**0.9921**	0.9898
Amazon Men	*POPRANK*	0.5607	0.6118	0.6538	0.0000	0.0000	0.0000
	BPR-MF	0.5969	0.6269	0.6447	0.5655	0.4583	0.3501
	VBPR	0.6754	0.6857	0.7164	0.9716	0.9760	0.9870
	DVBPR	0.6270	0.6471	0.6726	0.9796	0.9788	0.9797
	CO-BPR	**0.6815**	**0.6982**	**0.7358**	**0.9836**	**0.9902**	**0.9961**
Tradesy.com	*POPRANK*	0.4105	0.3939	0.4756	0.0000	0.0000	0.0000
	BPR-MF	0.5830	0.5763	0.5317	0.8522	0.8813	0.8694
	VBPR	0.6553	0.6819	0.6927	**0.9923**	**0.9964**	**0.9931**
	DVBPR	0.6134	0.6199	0.6497	0.9826	0.9920	0.9856
	CO-BPR	**0.6718**	**0.6873**	**0.7106**	0.9874	0.9901	0.9920

1) Compared with POPRANK method on Amazon datasets, the value of AUC increase with the increase of sparsity ratio. It tells us that customer more likely to purchase popular items on Amazon. However, on Tradesy.com, customers prefer to choose unpopular items.

2) Compared with methods without visual information, we can see that visual information substantially improve recommendation accuracy and diversity.

3) Compared with VBPR and DVBPR, we can see that learning visual compatibility relationship from an external dataset is very effective. Our proposed method CO-BPR outperforms all the comparison methods on all the four datasets. This demonstrates the significant benefits of generating recommendations with visual compatibility relationship.

Furthermore, to investigate the dimensionality sensitivity, we illustrate the performance of VBPR, DVBPR and CO-BPR with varying dimensionality in Figure 4. It is clear that both latent dimensionality and visual dimensionality are note very sensitive.



(a) latent dimensionality (b) visual dimensionality

Figure 4: Performance of VBPR, DVBPR and CO-BPR with varying dimensionality measured by AUC.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we have introduced a novel method for the fashion recommendation task with learning compatibility knowledge in visual aspect. A triplet network is used to learn compatibility knowledge from an external dataset. Domain adaptation strategy is used to alleviate the distribution gap between source domain and target domain. The experimental results show that our method is superior to several baselines in the AUC and diversity indicators.

In the future work, we intend to combine the learned image representation with the human understandable semantics to improve the interpretability of the model. In addition, we will use the image segmentation method to learn collocation knowledge from unmarked street photos on the web, thus reducing the reliance on manual labeling datasets.

ACKNOWLEDGMENTS

This research was supported by National Key R&D Program of China (No.2016YFB0801100), Beijing Natural Science Foundation (No.4172054, No.L181010), and National Basic Research Program of China (No.2013CB329605). This work was also supported by the Australian Research Council (ARC) under Grant [DP170101632].

REFERENCES

[1] Anurag Bhardwaj, Vignesh Jagadeesh, Wei Di, Robinson Piramuthu, and Elizabeth Churchill. 2014. Enhancing visual fashion recommendations with users in the loop. *arXiv preprint arXiv:1405.4013* (2014).

[2] Peter Gaspar. 2017. User preferences analysis using visual stimuli. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 436–440.

[3] Tiezheng Ge, Liqin Zhao, Guorui Zhou, Keyu Chen, Shuying Liu, Huiming Yi, Zelin Hu, Bochao Liu, Peng Sun, Haoyu Liu, et al. 2017. Image matters: Jointly train advertising CTR model with image representation of ad and user behavior. *arXiv preprint arXiv:1711.06505* (2017).

[4] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S Davis. 2017. Learning fashion compatibility with bidirectional lstms. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 1078–1086.

[5] Ruining He and Julian McAuley. 2016. VBPR: Visual bayesian personalized ranking from implicit feedback.. In *AAAI*. 144–150.

[6] Yang Hu, Xi Yi, and Larry S Davis. 2015. Collaborative fashion recommendation: A functional tensor factorization approach. In *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 129–138.

[7] Vignesh Jagadeesh, Robinson Piramuthu, Anurag Bhardwaj, Wei Di, and Neel Sundaresan. 2014. Large scale visual recommendations from street fashion images. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1925–1934.

[8] Shatha Jaradat. 2017. Deep cross-domain fashion recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 407–410.

[9] Wang-Cheng Kang, Chen Fang, Zhaowen Wang, and Julian McAuley. 2017. Visually-aware fashion recommendation and design with generative image models. In *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 207–216.

[10] Yuncheng Li, Liangliang Cao, Jiang Zhu, and Jiebo Luo. 2017. Mining fashion outfit composition using an end-to-end deep learning approach on set data. *IEEE Transactions on Multimedia* 19, 8 (2017), 1946–1955.

[11] Qiang Liu, Shu Wu, and Liang Wang. 2017. DeepStyle: Learning user preferences for visual recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 841–844.

[12] Si Liu, Jiashi Feng, Zheng Song, Tianzhu Zhang, Hanqing Lu, Changsheng Xu, and Shuicheng Yan. 2012. Hi, magic closet, tell me what to wear!. In *Proceedings of the 20th ACM international conference on Multimedia*. ACM, 619–628.

[13] Corey Lynch, Kamelia Aryafar, and Josh Attenberg. 2016. Images don't lie: Transferring deep visual semantic features to large-Scale multimodal learning to rank. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 541–548.

[14] Mingsong Mao, Jie Lu, Jialin Han, and Guangquan Zhang. 2019. Multiobjective e-commerce recommendations based on hypergraph ranking. *Information Sciences* 471 (2019), 269–287.

[15] Jose Oramas and Tinne Tuytelaars. 2016. Modeling visual compatibility through hierarchical mid-level elements. *arXiv preprint arXiv:1604.00036* (2016).

[16] Charles Packer, Julian McAuley, and Arnau Ramisa. 2018. Visually-aware personalized recommendation using interpretable image representations. *arXiv preprint arXiv:1806.09820* (2018).

[17] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[18] Devashish Shankar, Sujay Narumanchi, HA Ananya, Pramod Kompalli, and Krishnendu Chaudhury. 2017. Deep learning based large scale visual recommendation and search for e-commerce. *arXiv preprint arXiv:1703.02344* (2017).

[19] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning.. In *AAAI*, Vol. 4. 12.

[20] Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-prod2vec: Product embeddings using side-information for recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 225–232.

[21] Andreas Veit, Balazs Kovacs, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie. 2015. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *Proceedings of the IEEE International Conference on Computer Vision*. 4642–4650.

[22] Wei Wang, Guangquan Zhang, and Jie Lu. 2017. Hierarchy visualization for group recommender systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 99 (2017), 1–12.

[23] Dianshuang Wu, Jie Lu, and Guangquan Zhang. 2015. A fuzzy tree matching-based personalized e-learning recommender system. *IEEE Transactions on Fuzzy Systems* 23, 6 (2015), 2412–2426.

[24] Qian Zhang, Dianshuang Wu, Jie Lu, Feng Liu, and Guangquan Zhang. 2017. A cross-domain recommender system with consistent information transfer. *Decision Support Systems* 104 (2017), 49–63.

Discovering Product Defects and Solutions from Online User Generated Contents

Xuan Zhang
Virginia Tech* & Sam's Club
*Blacksburg, Virginia
xuancs@vt.edu

Zhilei Qiao
University of Alabama at Birmingham
Birmingham, Alabama
qiaozl@uab.edu

Aman Ahuja
Virginia Tech
Arlington, Virginia
aahuja@vt.edu

Weiguo Fan
University of Iowa
Iowa City, Iowa
weiguo-fan@uiowa.edu

Edward A. Fox
Virginia Tech
Blacksburg, Virginia
fox@vt.edu

Chandan K. Reddy
Virginia Tech
Arlington, Virginia
reddy@cs.vt.edu

ABSTRACT

The recent increase in online user generated content (UGC) has led to the availability of a large number of posts about products and services. Often, these posts contain complaints that the consumers purchasing the products and services have. However, discovering and summarizing product defects and the related knowledge from large quantities of user posts is a difficult task. Traditional aspect opinion mining models, that aim to discover the product aspects and their corresponding opinions, are not sufficient to discover the product defect information from the user posts. In this paper, we propose the Product Defect Latent Dirichlet Allocation model (PDLDA), a probabilistic model that identifies domain-specific knowledge about product issues using interdependent three-dimensional topics: Component, Symptom, and Resolution. A Gibbs sampling based inference method for PDLDA is also introduced. To evaluate our model, we introduce three novel product review datasets. Both qualitative and quantitative evaluations show that the proposed model results in apparent improvement in the quality of discovered product defect information. Our model has the potential to benefit customers, manufacturers, and policy makers, by automatically discovering product defects from online data.

CCS CONCEPTS

• **Information systems** → **Document topic models**; *Content analysis and feature selection*; • **Mathematics of computing** → *Bayesian computation.*

KEYWORDS

Product Defect Discovery; Topic Model; Opinion Mining

ACM Reference Format:
Xuan Zhang, Zhilei Qiao, Aman Ahuja, Weiguo Fan, Edward A. Fox, and Chandan K. Reddy. 2019. Discovering Product Defects and Solutions from Online User Generated Contents. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313732

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313732

1 INTRODUCTION

The number of Internet users almost tripled (4.2 Billion in Dec. 2018) in the past decade, and 3.4 Billion among them are social media users [1]. As a result, a vast volume of online UGC, including product defect related posts, is produced every day. For example, the official database of National Highway Traffic Safety Administration (NHTSA) has more than 1 million complaints related to the automobile industry. Even more posts are submitted to vehicle quality forums such as `toyotanation.com` and `honda-tech.com`. Large numbers of complaints occur in the customer service logs of manufacturers as well. This feedback can be useful for new customers to make purchase decisions, for existing customers to find resolutions to the problems, and for the manufacturers to improve the quality of products and services. Thus, identifying and summarizing product defect and resolution information from online user generated content has wide utility.

I have a 2002 Ford Explorer. **The o/d light was flashing and the transmission was slipping**. *We took it to our mechanic and it was diagnosed that the transmission needed to replace*	**I have an application completely stuck off screen. When I click on it main screen comes back but not the application!** *What app is it? try starting in safe mode (depress shift key when starting). That startup takes a while*.

Figure 1: Examples of product defect posts (bold: symptom, *italic*: resolution).

Two product defect posts from NHTSA and Apple MacBook forum, containing defect and solution, are shown in Figure 1. There are two important types of sentences in the posts: sentences in bold which describe the symptoms of the problem, and sentences in italics which illustrate resolution efforts to diagnose and fix the issue. In addition, there are component words, which may occur in both symptom and resolution sentences, and which give focus to a part, aspect, or subsystem of the product. Component, symptom, and resolution characterize product defects, and often occur as key entities in (or attributes of) product defect posts.

Domain-specific product defect mining can be treated as a natural extension of aspect opinion mining. Pioneered by the success of LDA-based models [3] in discovering latent representations of text,

[1] https://www.internetworldstats.com/emarketing.htm;
https://www.brandwatch.com/blog/amazing-social-media-statistics-and-facts/

most of the aspect-opinion mining research uses two-dimensional topic models following an "aspect + opinion" pattern, to summarize users' comments on a specific aspect [4, 11, 14, 15, 25]. In a sample post *The screen is very clear and great* [21], *screen* is an aspect, while *clear* and *great* are positive opinions.

The following three considerations explicate the distinctions between aspect opinion mining and product defect mining.

- *Component* (or *aspect*) is important in both types of mining; defects are always related to some property/feature of a product.
- The *opinion* entity is not sufficient to describe the other essential types of defect information. A product post usually contains information on *symptom* and *resolution*, which state how the defect manifests, and the fixes users/mechanics tried, respectively. Instead of *{aspect, opinion}*, an entity pattern of the form *{component, symptom, resolution}* is needed for modeling defects.
- The occurrence pattern of the key entities in consumer posts is distinctive. Traditional aspect opinion mining classifies text into "aspect" and "opinion" entities at the word level, which often occur as **word pairs** (e.g., "screen" and "clear" in the last example) [11]. But product defect mining needs to segment entities (especially "symptom" and "resolution") at the sentence level. Instead of occurring as word pairs, entities appear as **word groups** of varied sizes in posts.

Considering the unique characteristics of the product defect mining problem, we develop PDLDA, a three-dimensional LDA-based probabilistic model, which has the following primary contributions:

- The proposed model identifies the key entities of product defects – *component*, *symptom*, and *resolution* – from user posts as interdependent three-dimensional topics. For each entity of a defect, we have a corresponding topic. These topics are connected by the dependencies among entities.
- In contrast to the traditional aspect-opinion models, the proposed PDLDA model does not require word pairs. Existing LDA models for aspect summarization [10, 20, 22] usually take word pairs (e.g., an aspect word and an opinion word) as input. In contrast, PDLDA accepts user posts which have varying numbers of component, symptom, and resolution words, and even those without symptom or resolution entities.
- We share new datasets with practical utility that also can be employed in future studies.

2 PROPOSED MODEL

2.1 Problem Definition

Our work aims to build a model that can effectively identify three key entities of defects: *component*, *symptom*, and *resolution*. Component words (e.g., "engine", "brake", "power train") point out the flawed units which may cause the defect. Symptom words (e.g., "acceleration", "hesitate", "fail") describe what a product defect looks like. In contrast, resolution words (e.g., "check", "replace", "reset") show how people attempt to diagnose and fix the issues. Each of these entities is comprised of words, which usually follow a multinomial distribution across post documents. Furthermore, both symptom words and resolution words rely on the defective components. Therefore, our **research problem** is: given a dataset with D posts, the PDLDA model should produce a component topic φ for each of the K defects, which is the core entity. The component topic should be followed by symptom topics ψ and resolution topics τ which depend on it. Together, these joint topics should provide comprehensive information about a defect.

Table 1: Notation used in this paper.

K	number of topics
D	number of posts (documents)
V_c, V_s, V_r	size of component, symptom, and resolution word vocabularies, respectively
N_c, N_s, N_r	number of component, symptom, and resolution words in a post, respectively
C, S, R	Component, Symptom, and Resolution words in the corpus
H	stands for parameters of all Dirichlet distributions, including α, β, δ, γ, ϵ in the graphical diagram
φ, ψ, τ	word distributions (multinomial) over component, symptom, and resolution topics
θ, η, π	component topic distribution (multinomial) over documents, symptom topic distribution over component topics, and resolution topic distribution over component topics
λ_s, λ_r	Bernoulli distribution which decides whether a regular or a background topic is assigned to a word
t_s, t_r	indicator variables that decide whether a word is generated from a regular ($t = 0$) or a background topic ($t = 1$), for symptom and resolution words
z_i, y_i, x_i	topic assigned to the i^{Th} component, symptom, or resolution word
$z^{\neg i}$,$y^{\neg i}$, $x^{\neg i}$	topic assignment vector for various types of words in the corpus excluding the i^{th} word
$n_c^T(d, k)$	number of component words in document d assigned to topic k; T represents topic distribution
$n_c^W(k, v_c)$	number of component words v_c assigned to component topic k; W represents word distribution
$n_s^T(k, l)$	number of symptom words assigned to topic l, whose corresponding component topic is k
$n_s^W(l, v_s)$	number of symptom words v_s assigned to symptom topic l
$n_r^T(k, m)$	number of resolution words assigned to topic m, whose corresponding component topic is k
$n_r^W(m, v_r)$	number of resolution words v_r assigned to resolution topic m
NTI_s, NTI_r	normalized $TF * IDF$ value of symptom or resolution word

2.2 Entity Extraction

When multiple types of entities are involved, we have to separate words of various types. In our case, we need to find the words belonging to component, symptom, and resolution entities, given a post. Some researchers [10, 20, 22, 24] classify words into categories before entering them into LDA models, while others assign a type to each word with the generative process [2, 16]. The first strategy is more efficient for product defect mining, since words in one sentence usually belong to the same entity (except that a component word may appear in symptom and resolution sentences). The entity extraction methods are introduced in Section 3.1.2 in detail.

2.3 PDLDA Model

As shown in Figure 2, PDLDA models key entities (component, symptom, and resolution) of defects with interdependent topics. There are three types of observed entities in the graphical model: component words (W_c), symptom words (W_s), and resolution words (W_r). The topics assigned to these words are z, y, and x, respectively. We use two conditional probability distributions η and π to model the dependence between symptom and resolution with the component. The notation used in our paper is described in Table 1. The post generation process is illustrated in Algorithm 1.

In the first five 1st-level loops, PDLDA determines the word distributions. For each component topic, a word distribution φ is

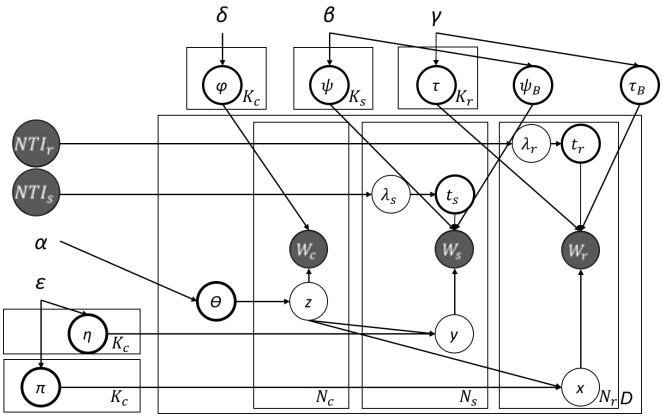


Figure 2: Plate notation of PDLDA model.

sampled from Dirichlet distribution δ. We also sample two corresponding symptom and resolution topic distributions, η and π, which depend on the component topic, from the Dirichlet distribution ϵ. They are used to create the dependence matrices. Word distributions ψ and τ for each symptom and resolution topic (both regular and background topics) are sampled, respectively.

The topic assignment is determined in the last 1st-level loop. Given a post d, we first sample a component topic z from Multinomial θ for each of the N_c component words in a document d. The component topic sampling is the same as with standard LDA. As a result, we get a set of topics, $Z_d = \{z_1, z_2, \cdots, z_{N_c}\}$. Next, to choose a y for each symptom word W_s, a component topic z' is randomly drawn from the component topic set Z_d, since we assume y depends on a certain z'. Then, we identify whether W_s is a *background* or *regular* symptom word by sampling an indicator t_s from a Bernoulli distribution λ_s. If the word is a regular symptom word ($t_s = 0$), we sample a symptom topic y from Multinomial $\eta_{z'}$ (a multinomial distribution assuming the current component topic is z') for each symptom word W_s, then sample the word from Multinomial ψ_y; otherwise ($t_s = 1$), the background symptom topic is assigned to the word, and we sample it from Multinomial ψ_B. The same sampling process is used during the generation of each resolution word W_r. Note PDLDA can be applied to a set of posts as long as they have a component entity; they do not have to have all three entities. PDLDA can sample topics for the existing entities even when a post has zero or one of: symptom or resolution entities.

A special feature of the PDLDA model is the presence of two switch variables t_s and t_r, which are designed to reduce overlap between different symptom (or resolution) topics. In particular, we give the most common words high probability of being assigned into a background symptom (or resolution) topic. Thus, a background topic will be formed by the words which are common to all the defects, such as "problem", "work", "stop". The TF-IDF value of a word is used to decide whether it is a background or a regular word. Specifically, we draw indicators t_s and t_r from Bernoulli distributions λ_s and λ_r, respectively. The parameter of these distribution is initialized by the Normalized TF-IDF value (NTI_s or NTI_r) of that word, in particular, $P(t_s = 0) = normalized\ TF * IDF$.

Algorithm 1: Post generation process of PDLDA

for *each component topic* **do**
  Sample word distribution $\varphi \sim Dirichlet(\delta)$
  Sample conditional topic distribution $\eta \sim Dirichlet(\epsilon)$ for *symp* entities
  Sample conditional topic distribution $\pi \sim Dirichlet(\epsilon)$ for *reso* entities
for *each regular symptom topic* **do**
  Sample word distribution $\psi \sim Dirichlet(\beta)$
for *background symptom topic* **do**
  Sample word distribution $\psi_B \sim Dirichlet(\beta)$
for *each regular resolution topic* **do**
  Sample word distribution $\tau \sim Dirichlet(\gamma)$
for *background resolution topic* **do**
  Sample word distribution $\tau_B \sim Dirichlet(\gamma)$
for *each post d* **do**
  Sample *comp* topic distribution $\theta_d \sim Multinomial(\alpha)$
  for *each comp word* W_c *of d* **do**
    Sample a *comp* topic $z \sim Multinomial(\theta_d)$
    Sample $W_c \sim Multinomial(\varphi_z)$
  for *each symp word* W_s *of d* **do**
    Draw a *comp* topic z' from the set of z
    Sample an indicator $t_s \sim Bernoulli(\lambda_s)$
    if $t_s = 0$ **then**
      Sample a *symp* topic $y \sim Multinomial(\eta_{z'})$
      Sample $W_s \sim Multinomial(\psi_y)$
    else if $t_s = 1$ **then**
      Sample $W_s \sim Multinomial(\psi_B)$
  for *each reso word* W_r *of d* **do**
    Draw a *comp* topic z' from the set of z
    Sample an indicator $t_r \sim Bernoulli(\lambda_r)$
    if $t_r = 0$ **then**
      Sample a *reso* topic $x \sim Multinomial(\pi_{z'})$
      Sample $W_r \sim Multinomial(\tau_x)$
    else if $t_r = 1$ **then**
      Sample $W_r \sim Multinomial(\tau_B)$

Using this sampling strategy, especially sampling a component topic for related symptom and resolution topics, we eliminate a key drawback of the existing aspect mining LDA models [10, 20, 22] which require word pairs. For these methods, the opinion topic sampling for an opinion word relies on the aspect topic assigned to the aspect word in that word pair[2].

2.4 Model Inference

We will now provide the details of the inference of the proposed model which uses collapsed Gibbs sampling. The Gibbs updating rules discussed below show how PDLDA assigns topics to the component, symptom, and resolution words in a post.

The Gibbs updating rule for a component word i is:

$$P(z_i = k|z^{\neg i}, C, H) \propto \frac{n_c^T(d,k) + \alpha_k - 1}{\sum_{k'=1}^{K}(n_c^T(d,k') + \alpha_{k'}) - 1} \cdot \frac{n_c^W(k, v_c) + \delta_{v_c} - 1}{\sum_{v'=1}^{V_c}(n_c^W(k,v') + \delta_{v'}) - 1} \quad (1)$$

[2]This is the main reason why these models only accept word pairs as input.

Table 2: Datasets used in our evaluation.

Datasets	All	Labeled	
	Posts	Posts	Clusters
FOCUS (98-04)	10,099	1,941	4
MACBOOK	2,098	413	5
Patient.info	43,928	43,928	10

Assuming the corresponding component topic of symptom word i is k, the Gibbs updating rule for symptom words is:

$$P(y_i = l|z, y^{\neg i}, S, H) \propto \frac{n_s^T(k,l) + \epsilon_l - 1}{\sum_{l'=1}^{K}(n_s^T(k,l') + \epsilon_{l'}) - 1} \cdot \frac{n_s^W(l, v_s) + \beta_{v_s} - 1}{\sum_{v'=1}^{V_s}(n_s^W(l,v') + \beta_{v'}) - 1} \quad (2)$$

Assuming the corresponding component topic of resolution word i is k, the Gibbs updating rule for resolution words is:

$$P(x_i = m|z, x^{\neg i}, R, H) \propto \frac{n_r^T(k,m) + \epsilon_m - 1}{\sum_{m'=1}^{K}(n_r^T(k,m') + \epsilon_{m'}) - 1} \cdot \frac{n_r^W(m, v_r) + \gamma_{v_r} - 1}{\sum_{v'=1}^{V_r}(n_r^W(m,v') + \gamma_{v'}) - 1} \quad (3)$$

As mentioned before, the model decides whether a symptom (or resolution) word should be assigned with a background or regular symptom (or resolution) topic based on its TF-IDF value. The TF-IDF value is normalized for each post and used to set the parameter of the Bernoulli distribution λ_s or λ_r, from which we sample the type (background or regular) of that word.

3 EVALUATION

3.1 Experimental Setup

3.1.1 Datasets. To evaluate the usability of our model on different kinds of data, we use a wide range of datasets for our evaluation, including the complaint database of NHTSA[3], the problem discussion threads in the official forum of Apple[4], and the disease discussion threads in the Patient.info forum[5]. In the NHTSA and Apple datasets, we aim to discover the product defects and resolutions. In the Patient.info dataset, the human diseases can be considered analogous to product defects, and we aim to find their treatments, which are analogous to defect resolutions. Products which received the most complaints are selected from those data sources, and a post dataset is constructed for each, shown in Table 2. Specifically, we select posts of FORD FOCUS (1998-2004), Apple MacBook, and Patient.info to create 3 datasets. The entire datasets are used for topic coherence and qualitative evaluation. In addition, subsets of these datasets are manually labeled for post clustering evaluation. The post clusters are manually tagged by four Virginia Tech undergraduate students (each post tagged by 2 students), with an inter-annotator agreement rate of 81.2%. The post and cluster count of subsets are shown in columns 3 and 4 in Table 2.

[3]https://www-odi.nhtsa.dot.gov/downloads/flatfiles.cfm
[4]https://discussions.apple.com
[5]https://patient.info/forums

3.1.2 Data Preprocessing. Since this research primarily focusses more on the design of the topic model, rather than segmentation of the entities, we used light weight methods to implement entity segmentation. The frequent item set mining approach is used to extract components in all the datasets, by following the method introduced in [8]. Regarding the extraction of symptom and resolution entities, different strategies are used on various datasets.

- For the Apple forum and Patient.info datasets, we rely on the thread structure to separate them. Each forum thread is taken as a document. Then, we take the sentences in the first post as symptom sentences and those in the "most recommended solution" post as resolution sentences.
- A lexicon-based method is applied to the NHTSA datasets. Three types of sentences exist in the NHTSA posts: ownership, symptom, and resolution. We go through the sentences of a small dataset which has 400 posts of different vehicle models, find the feature words (e.g., "buy", "purchase", and "own") of ownership sentences, and find the feature words (e.g., "fix", "remedy", and "resolve") of resolution sentences. In this way, two lexicons are created, which will help decide the sentence type according to the lexicon word occurrence.

Topic models are quite sensitive to noise such as stop-words. Thus, a filter based on POS tagging further removes less-informative candidate symptom and resolution words. Based on our observations, we assume the most informative words are nouns, verbs, and adjectives. We apply the Stanford POS tagger [18] to symptom and resolution sentences, keeping only nouns, verbs, and adjectives. Then, we can provide observations (three entities for each post) to the PDLDA model.

3.1.3 Implementation and Parameter Setting. PDLDA is implemented in Java. We use symmetric priors for the Dirichlet distribution parameters $\alpha, \beta, \delta, \gamma$, and ϵ, and set them with empirically derived values $\alpha = 0.01$, $\beta = \gamma = \delta = \epsilon = 0.001$.

3.2 Performance Evaluation

3.2.1 Evaluation Criteria. Both quantitative and qualitative evaluations are conducted for the evaluation of the proposed PDLDA model.

- *Entity Clustering Accuracy*: Precision, Recall, and F-Measure [11] metrics are selected to measure entity (e.g., symptom and resolution) clustering performance. We follow the process introduced in [23] to calculate them.
- *Topic Coherence*: The Pointwise Mutual Information (PMI) metric [12] is chosen for topic coherence measurement.
- *Qualitative Evaluation*: The defects of different product models are identified from posts as joint topics.

3.2.2 Baseline Methods.

- *Standard LDA* [3]: After entity extraction using the methods in the last section, we run standard LDA on the words of three entities separately to get three kinds of topics.
- *ILDA* [10]: We get the component (aspect) and the symptom (opinion) topics by taking symptom sentences as input, and get the component (aspect) and the resolution (opinion) topics from resolution sentences. A dependency parser [5] is used to extract aspect and opinion entities.

Table 3: Average Precision, Recall, and F-1 (%) of post entity clustering.

Dataset	FORD FOCUS						APPLE MacBook						Patient.info					
Entity	Symptom			Resolution			Symptom			Resolution			Symptom			Resolution		
Method	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
LDA	69.09	55.47	56.35	68.72	25.00	36.55	31.1	24.84	27.33	32.32	25.94	28.69	43.72	45.97	44.45	31.17	29.98	30.32
ILDA	69.58	37.74	47.52	64.86	32.31	39.02	30.31	28.27	29.65	30.79	23.13	25.11	40.89	30.80	34.36	31.45	19.90	23.17
ME-LDA	64.38	26.53	23.77	73.62	25.94	22.69	37.44	**33.84**	34.98	35.39	**31.21**	31.93	41.19	41.30	40.52	30.88	30.47	30.39
PDLDA	**88.82**	**57.08**	**59.24**	**91.36**	**33.34**	**43.51**	**57.7**	32.84	**35.29**	**49.62**	30.25	**33.69**	**53.82**	**43.82**	**44.52**	**36.96**	**30.56**	**31.15**

- *ME-LDA* [25]: We use the same input used in ILDA to produce topics by ME-LDA. Our own component lexicon and the opinion lexicon in [8] are used to label word types.

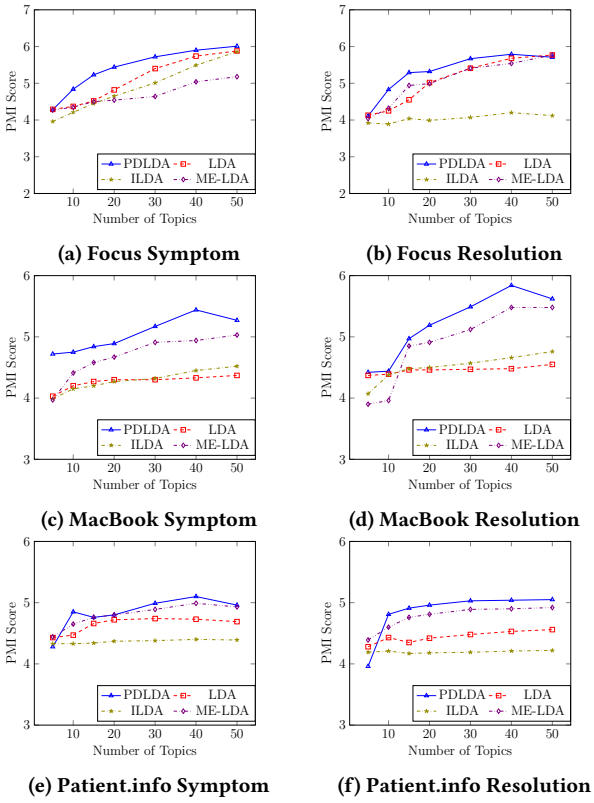
3.3 Experimental Results

3.3.1 Post Clustering. For this evaluation, the posts are clustered along two dimensions. First, they are clustered according to the symptom entity, then according to the resolution entity, using different models. When clustering symptom entities of posts, we take the symptom topic ID assigned to the majority of symptom words of a post as its symptom cluster. A similar clustering process occurs when the posts are clustered according to their resolution entities.

The clustering performance on the three datasets is shown in Table 3. We can see the precision and F-measure of PDLDA are better than the baselines, for both symptom and resolution clustering on all the datasets. ME-LDA has better recall on the MacBook dataset. Since PDLDA jointly models the three entities, it leverages the influence of component entities when clustering dependent entities. Therefore, it outperforms the standard LDA. In addition, PDLDA extracts more reasonable words on product defects than ILDA and ME-LDA; thus it captures more important features when clustering post entities. Also, the clustering performance on resolution entities is low for all of the models, compared to the symptom entities. This reveals that the words in resolution sentences are more difficult to separate. We observe that the words describing problem fixing measures are quite common, especially when we exclude the component words.

3.3.2 Topic Coherence. The topic coherence scores over 3 datasets are shown in Figure 3. Since the sampling of component topics in PDLDA is the same as the standard LDA, they perform quite similarly in terms of the topic coherence of components. Thus we only show the topic coherence of symptom and resolution topics. PDLDA outperforms the baseline methods in most cases, especially when the number of topics is over 15. Both the topic interdependency and entity identification mechanisms contribute to this. The PMI of PDLDA stops increasing and converges to the baselines when the topic number is very large. A possible reason is, the actual number of topics in the datasets is limited; thus, more overlap among topics occurs along with the topic number increase, which may prevent the PMI increase. ME-LDA outperforms LDA and ILDA, as it benefits from incorporating prior knowledge (i.e., tagged word types) when identifying word types. Simple models (LDA and ILDA) perform better when the number of topics is small (e.g., 5 topics).

3.3.3 Qualitative Evaluation. While capturing critical defects of a vehicle model, PDLDA uses joint topics, composed of the most probable component, symptom, and resolution words, to show the defect information. In our output, each defect is led by its



(a) Focus Symptom (b) Focus Resolution

(c) MacBook Symptom (d) MacBook Resolution

(e) Patient.info Symptom (f) Patient.info Resolution

Figure 3: PMI coherence of symptom and resolution topics.

component topic, followed by relevant symptom and resolution topics. The component topic distribution $p(\theta_{d,k})$ in document d is determined by Eq. (4). Given the component topic k, the most relevant symptom topic l can be obtained by sorting $\eta_{k,l}$, which is calculated by Eq. (5). Similarly, the most relevant resolution topic m given component topic k is obtained by sorting $\pi_{k,m}$, which is calculated by Eq. (6). The keywords of all types of topics can be figured out using the second part of the Gibbs updating rules.

$$\theta_{d,k} = \frac{n_c^T(d,k) + \alpha_k}{\sum_{k'=1}^{K}(n_c^T(d,k') + \alpha_{k'})} \tag{4}$$

$$\eta_{k,l} = \frac{n_s^T(k,l) + \epsilon_l}{\sum_{l'=1}^{K}(n_s^T(k,l') + \epsilon_{l'})} \tag{5}$$

$$\pi_{k,m} = \frac{n_r^T(k,m) + \epsilon_m}{\sum_{m'=1}^{K}(n_r^T(k,m') + \epsilon_{m'})} \tag{6}$$

Table 4: Example of topics extracted by LDA, ILDA, ME-LDA, and PDLDA (Nonrelevant words of each topic are underlined).

Component				Symptom				Resolution			
LDA	**ILDA**	**ME-LDA**	**PDLDA**	**LDA**	**ILDA**	**ME-LDA**	**PDLDA**	**LDA**	**ILDA**	**ME-LDA**	**PDLDA**
ignition	key	lock	ignition	turn	ford	turn	turn	dealer	have	replace	tow
key	focus	steer	key	lock	start	ignition	lock	contact	ford	ignition	lock
cylinder	car	remove	cylinder	start	turn	steer	key	repair	take	key	key
lock	vehicle	rotor	lock	steer	key	wheel	insert	maker	numerous	turn	locksmith
wheel	problem	open	wheel	key	put	lock	park	failure	recall	tow	call
				stick	get	start	stick	shop	resolve	start	drill
				unable	insert	key	remove	aware	receive	lock	common
				put	go	work	place	local	ignition	work	tumbler
				set	ignition	attempt	unlock	advise	get	call	advise
				unlock	common	stick	freeze	owner	defective	leave	aaa

Due to space constraints, Table 4 shows joint topics only for the "ignition" issue of FOCUS 98-04. The top 5 words are shown for component topics, while the top 10 words are shown for symptom/resolution topics. Looking at the joint topics produced by PDLDA, we see that the ignition component problem caused the key stuck in the lock so it could not turn, which was resolved by car towing and locksmith service. For each defect, users can locate the flawed units by reading the component topic, then learn what happens and how it is fixed, by reading the dependent topics. The nonrelevant words in the topics are underlined, through which we can see the topics generated by PDLDA have fewer nonrelevant and general words (e.g., "put" and "get" in column 6).

As demonstrated above, PDLDA can effectively and accurately identify defects and related solutions of products from online UGC, especially for products which people can fix by themselves, such as vehicles, electronics, etc. And it can also help patients find the symptoms and related treatments of diseases. Based on them, customers can make purchase decisions, manufacturers can improve products, governments can undertake administrative actions, and patients can find useful information for disease diagnosis and treatment.

4 RELATED WORK

Product defect mining is an extension of aspect-based opinion mining which has not been thoroughly investigated in the literature. Earlier works on product defect mining [1] classified product defects, as discussed in online forums, according to the level of severity. This method attempted to discover whether a defect exists in a review, or the severity of the defect, but did not account for the details of the defect (e.g. symptoms). Later, Tutubalina and Ivanov [19] extracted key phrases on product problems from user reviews according to the syntax dependencies among words. But, it relies on pre-defined syntax dependency rules, which is hard to generalize.

LDA based models have been successfully used for aspect-based opinion mining, and have incorporated various related factors, due to the unsupervised and flexible characteristics of LDA. Titov and McDonald [17] separated global and local opinion topics using a multiple-grain LDA model. Lin and He [9] pointed out the importance of sentiment in terms of opinion mining and added it as a key latent variable in their topic model. Zhao et al. [25] proposed a Maximum Entropy-LDA hybrid model to separate aspect words and aspect-specific opinion words. Ahuja et al. [2] incorporated Geotaggers into their aspect-opinion model by assuming people living in different places may discuss different topics and hold varied opinions. Park et al. [13] proposed an LDA model which connected Mobile Application (APP) descriptions with corresponding reviews in order to improve mobile APP searching. A holistic model which included most of the above key entities (e.g., aspect, opinion, sentiment, and granularity) was proposed by Wang et al. [21]. They further boosted their LDA model with domain knowledge learned by lifelong machine learning, which can be regarded as an extension of [7] and [6]. Rakesh et al. [16] suggested a spike-and-slab prior over the document-topic distribution, which generates fine-grained summaries of online reviews. However, the entities captured by these traditional aspect opinion mining models are not specific enough for product defect mining. In addition, these models cannot adapt to the characteristics of defect description (e.g., key entity locations).

5 CONCLUSION

This paper addresses the problem of product defect identification. To identify the key information of product defects, we develop a PDLDA model which extracts interdependent topics (component, symptom, and resolution) from UGC. This model differs from existing aspect summarization models as it jointly identifies the key entities of product defects as interdependent three-dimensional topics. In addition, PDLDA eliminates the dependence on word pairs that are required by the existing models. Collapsed Gibbs sampling is used for model inference. Both quantitative and qualitative evaluations are performed. PDLDA is shown to effectively identify product defects and to outperform existing methods. Since the (three) types of entities modeled by PDLDA frequently exist in the posts on different products, it can be easily generalized to health applications as well as other products, especially to products which people can fix by themselves, such as software, cell phones, cameras, etc. As future work, we plan to explore other entities related to product defects, such as user requirements and product improvements. The datasets and code will be released on Github[6].

Acknowledgments

This work was supported in part by the US National Science Foundation grants IIS-1619028, IIS-1707498, and IIS-1838730, and NVIDIA Corporation.

[6] https://github.com/aizest/PDLDA

REFERENCES

[1] Alan S Abrahams, Jian Jiao, G Alan Wang, and Weiguo Fan. 2012. Vehicle defect discovery from social media. *Decision Support Systems* 54, 1 (2012), 87–97.

[2] Aman Ahuja, Wei Wei, Wei Lu, Kathleen M Carley, and Chandan K Reddy. 2017. A Probabilistic Geographical Aspect-Opinion Model for Geo-tagged Microblogs. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 721–726.

[3] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet Allocation. *Journal of machine learning research* 3 (2003), 993–1022.

[4] Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 804–812.

[5] Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 740–750.

[6] Zhiyuan Chen and Bing Liu. 2014. Topic modeling using topics from many domains, lifelong learning and big data. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 703–711.

[7] Zhiyuan Chen, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Leveraging Multi-Domain Prior Knowledge in Topic Models.. In *IJCAI*, Vol. 13. 2071–77.

[8] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge Discovery and Data mining*. ACM, 168–177.

[9] Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*. ACM, 375–384.

[10] Samaneh Moghaddam and Martin Ester. 2011. ILDA: interdependent LDA model for learning latent aspects and their ratings from online product reviews. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 665–674.

[11] Samaneh Moghaddam and Martin Ester. 2012. On the design of LDA models for aspect-based opinion mining. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*. ACM, 803–812.

[12] David Newman, Youn Noh, Edmund Talley, Sarvnaz Karimi, and Timothy Baldwin. 2010. Evaluating topic models for digital libraries. In *Proceedings of the 10th annual joint conference on Digital libraries*. ACM, 215–224.

[13] Dae Hoon Park, Mengwen Liu, ChengXiang Zhai, and Haohong Wang. 2015. Leveraging User Reviews to Improve Accuracy for Mobile APP Retrieval. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 533–542.

[14] Dae Hoon Park, C Zhai, and Lifan Guo. 2015. SpecLDA: Modeling product reviews and specifications to generate augmented specifications. In *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM.

[15] Michael Paul and Roxana Girju. 2010. A Two-dimensional Topic-aspect Model for Discovering Multi-faceted Topics. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI'10)*. AAAI Press, 545–550.

[16] Vineeth Rakesh, Weicong Ding, Aman Ahuja, Nikhil Rao, Yifan Sun, and Chandan K. Reddy. 2018. A Sparse Topic Model for Extracting Aspect-Specific Summaries from Online Reviews. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1573–1582.

[17] Ivan Titov and Ryan McDonald. 2008. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 111–120.

[18] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 173–180.

[19] Elena Tutubalina and Vladimir Ivanov. 2014. Unsupervised approach to extracting problem phrases from user reviews of products. *COLING 2014*. (2014), 48.

[20] Hao Wang and Martin Ester. 2014. A sentiment-aligned topic model for product aspect rating prediction. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1192–1202.

[21] Shuai Wang, Zhiyuan Chen, and Bing Liu. 2016. Mining aspect-specific opinion using a holistic lifelong topic model. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 167–176.

[22] Wei Xue, Tao Li, and Naphtali Rishe. 2015. Aspect and ratings inference with aspect ratings: supervised generative models for mining hotel reviews. In *Web Information Systems Engineering–WISE 2015*. Springer, 17–31.

[23] Mohammed J Zaki, Wagner Meira Jr, and Wagner Meira. 2014. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press.

[24] Tian-Jie Zhan and Chun-Hung Li. 2011. Semantic dependent word pairs generative model for fine-grained product feature mining. In *Advances in Knowledge Discovery and Data Mining*. Springer, 460–475.

[25] Wayne Xin Zhao, Jing Jiang, Hongfei Yan, and Xiaoming Li. 2010. Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. 56–65.

Your Style Your Identity: Leveraging Writing and Photography Styles for Drug Trafficker Identification in Darknet Markets over Attributed Heterogeneous Information Network

Yiming Zhang, Yujie Fan
Department of CSEE
West Virginia University, WV, USA
{ymzhang,yf0004}@mix.wvu.edu

Wei Song, Shifu Hou
Department of CSEE
West Virginia University, WV, USA
{ws0016,shhou}@mix.wvu.edu

Yanfang Ye*, Xin Li
Department of CSEE
West Virginia University, WV, USA
{yanfang.ye,xin.li}@mail.wvu.edu

Liang Zhao
Department of IST
George Mason University, VA, USA
lzhao9@gmu.edu

Chuan Shi
School of Computer Science
BUPT, Beijing, China
shichuan@bupt.edu.cn

Jiabin Wang, Qi Xiong
Tecent Security Lab
Tecent, Guangdong, China
luciferwang@tencent.com

ABSTRACT

Due to its anonymity, there has been a dramatic growth of underground drug markets hosted in the darknet (e.g., Dream Market and Valhalla). To combat drug trafficking (a.k.a. illicit drug trading) in the cyberspace, there is an urgent need for automatic analysis of participants in darknet markets. However, one of the key challenges is that drug traffickers (i.e., vendors) may maintain multiple accounts across different markets or within the same market. To address this issue, in this paper, we propose and develop an intelligent system named *uStyle-uID* leveraging both writing and photography styles for drug trafficker identification at the first attempt. At the core of *uStyle-uID* is an attributed heterogeneous information network (AHIN) which elegantly integrates both writing and photography styles along with the text and photo contents, as well as other supporting attributes (i.e., trafficker and drug information) and various kinds of relations. Built on the constructed AHIN, to efficiently measure the relatedness over nodes (i.e., traffickers) in the constructed AHIN, we propose a new network embedding model *Vendor2Vec* to learn the low-dimensional representations for the nodes in AHIN, which leverages complementary attribute information attached in the nodes to guide the meta-path based random walk for path instances sampling. After that, we devise a learning model named *vIdentifier* to classify if a given pair of traffickers are the same individual. Comprehensive experiments on the data collections from four different darknet markets are conducted to validate the effectiveness of *uStyle-uID* which integrates our proposed method in drug trafficker identification by comparisons with alternative approaches.

CCS CONCEPTS

• **Security and privacy → Web application security**; • **Networks →** *Online social networks*; • **Computing methodologies → Machine learning algorithms**.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313537

KEYWORDS

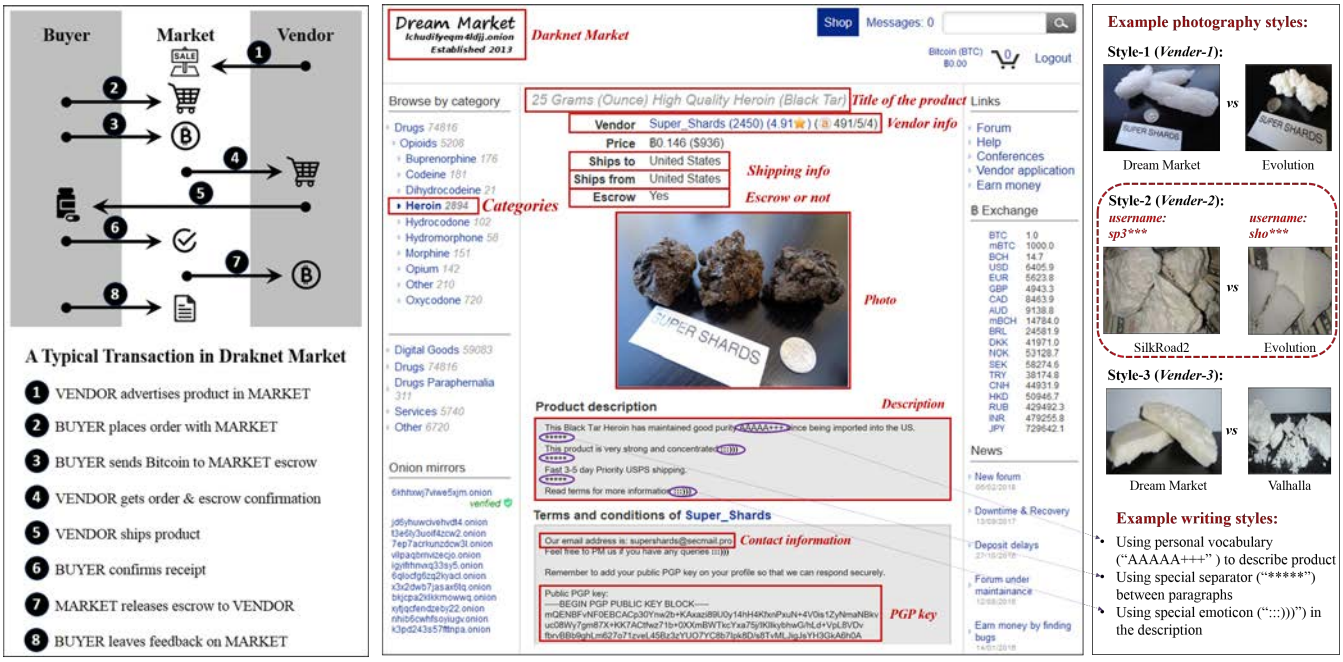
Darknet Market; Drug Trafficker Identification; Attributed Heterogeneous Information Network (AHIN); Network Embedding.

ACM Reference Format:
Yiming Zhang, Yujie Fan, Wei Song, Shifu Hou, Yanfang Ye*, Xin Li, Liang Zhao, Chuan Shi, and Jiabin Wang, Qi Xiong. 2019. *Your Style Your Identity*: Leveraging Writing and Photography Styles for Drug Trafficker Identification in Darknet Markets over Attributed Heterogeneous Information Network. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313537

1 INTRODUCTION

The market of illicit drugs (e.g., cannabis, cocaine, heroin) is considerably lucrative - i.e., the estimated yearly revenue for the global market reached about $426-$652 billion in 2017 [23]. Driven by such remarkable profits, the crime of drug trafficking (a.k.a. illicit drug trading) has never stopped but co-evolved along with the advance of modern technologies [4, 16, 17, 28, 32, 38, 39]. Darknet, as a hidden part of the Internet, employs advanced encryption techniques to protect the anonymity of its users. The markets hosted in the darknet are built on The Onion Router (TOR) service to hide the IP address, the escrow system, the encrypted communication tools like Pretty Good Privacy (PGP), and the virtually untraceable cryptocurrency (e.g., bitcoin) to facilitate anonymous transactions among participants. Figure 1.(a) illustrates a typical transaction in darknet markets. Due to its anonymity, there has been a dramatic growth of underground drug markets hosted in the darknet (e.g., Silk Road 3 [33], Dream Market [27], Valhalla [37], known as "eBay of drugs" or "Amazon of drugs"). Illegal trading of drugs in these markets has turned into a serious global concern because of its severe consequences on society (e.g., violent crimes) and public health at regional, national and international levels [8].

To combat drug trafficking in the cyberspace, there is an urgent need for analysis of participants in darknet markets, as it could provide valuable insight to the investigation of drug trafficking ecosystem and prediction of future incidents while building proactive defenses [6]. However, one of the key challenges is that drug traffickers may maintain multiple accounts across different markets or in the same market for the reasons [2, 4, 34] such as ripper (i.e.,



(a) A typical transaction in darknet market.

(b) Illicit drug advertised by a trafficker (i.e., vender) in darknet market.

(c) Illustration of writing & photography styles.

Figure 1: Illustration of drug trafficking in Darknet market.

an old account has lost the trust of other members), branding (i.e., a vendor creates an alias to positively review his/her own products or services), and anonymity. Linking different accounts to the same individuals is essential to track their status and better understand the online drug trafficking ecosystem [40]. Given the growing scale of darknet markets and the large number of user accounts, it is simply impossible to manually link suspicious accounts and track their latest status. Therefore, it is highly desirable to develop novel methodologies that can automatically link multiple accounts of the same individuals in darknet markets.

To automate the process, some of existing approaches [2, 5, 20] relied on stylometry analysis which aims at linking different accounts to the same user based on his/her writing styles (e.g., as illustrated in Figure 1.(b) and (c), the vendor likes using specific emoticon in the product description). Since drug traffickers in darkent markets have to prove the possession of illegal drugs by posting their own product photos, their distinct photography styles might be revealed by the posted photos. A recent research [40] proposed to link multiple accounts of the same vendors in different darknet markets based on their distinct photography styles (e.g., the way to display products and camera model as shown in Figure 1.(c)). Though each kind of analysis has shown success in fingerprinting underground market participants, using them respectively may suffer different challenges (e.g., stylometry analysis only is sensitive to the language of content [40], while photography style analysis only may face the challenge of intrinsic ambiguity arising from resale or photo plagiarizing). *Can we leverage both writing and photography styles to develop an integrated framework for drug trafficker identification in darknet markets?* To the best of our knowledge, this has yet studied in the open literature.

In this paper, we propose to leverage both writing and photography styles to develop an intelligent system (named *uStyle-uID*) to automatically link multiple accounts of the same individuals for drug trafficker identification in darknet markets. In *uStyle-uID*, given a pair of vendors (denoted by their usernames in the related markets), to determine whether they are the same individual, we not only analyze their posted contents (i.e., including their posted texts and photos), but also consider their writing styles and photography styles as well as other supporting attributes (i.e., vendor and drug information) and various kinds of relations. To depict vendors, drugs, texts, photos and their associated attributes as well as the rich relations among them, we present an attributed heterogeneous information network (AHIN) [26] for modeling. To tackle the challenge of high computation cost and memory constraint of measuring the relatedness over vendors in the constructed AHIN, we propose a new network embedding model named *Vendor2Vec* to learn low-dimensional attribute-aware embeddings for the nodes in AHIN. The proposed *Vendor2Vec* model leverages complementary attribute information of each node to guide the meta-path based random walk for path-instance sampling; then a skip-gram model [30] is utilized to learn effective node representations for AHIN. Finally, based on the learned latent representations of the nodes (i.e., vendors) in AHIN, we devise a learning model named *vIdentifier* to classify whether a given pair of vendors are the same individual.

2 PROPOSED METHOD

The overview of our developed system *uStyle-uID* for drug trafficker identification in darknet markets is shown in Figure 2. In this section, we will introduce the detailed approaches which are integrated in *uStyle-uID* for drug trafficker identification.

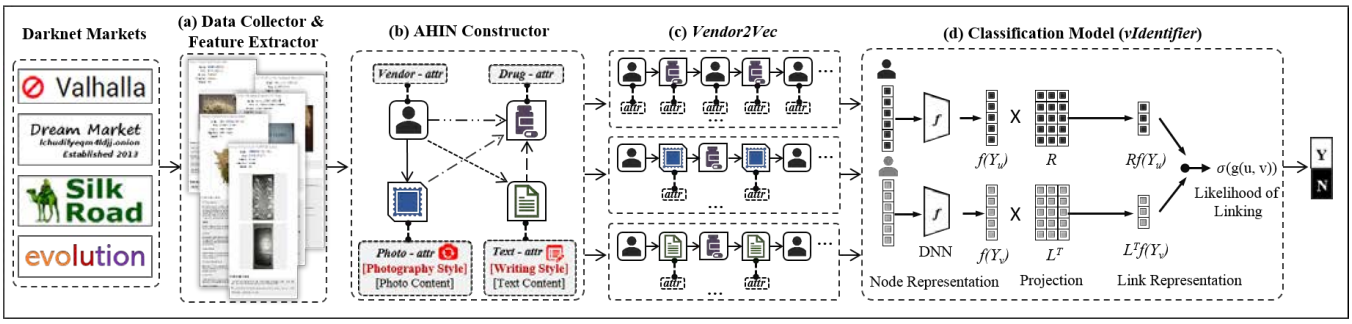


Figure 2: System architecture of *uStyle-uID*.

2.1 Feature Extraction

We propose to characterize vendors in darknet markets in a comprehensive view by extracting various features.

(1) Posted text and writing style extraction. To fingerprint a vendor based on his/her posted texts, we consider both his/her posted text content and writing style. For text content, we apply *doc2vec* [25] to convert each text of variant size into a fixed length feature vector (empirically, we set the dimension to 100). For writing style [2, 20], we propose to extract multi-scale stylometry features at three different levels as follows: 1) ***Lexical features*** can be further divided into character-based and word-based groups to capture stylistic traits. At this level, we extract *i*) number of characters, *ii*) number of digits/white spaces/special characters, *iii*) number of words, *iv*) average word length, and *v*) vocabulary richness [36]. 2) ***Syntactic features*** capture the writing style from the sentence structure. In this category, we adopt *i*) frequency of punctuation, *ii*) frequency of function word, *iii*) number of sentences beginning with a capital letter, and *iv*) frequency of parts-of-speech *n*-grams (we set $n = 3$ in our case). 3) ***Structural features*** represent the way an author organizes the layout of his/her posted text. We consider *i*) total number of paragraphs, *ii*) indentation of paragraph, *iii*) whether there's separator between paragraphs, and *iv*) number of words/sentences/characters per paragraph. For each posted text, we then concatenate its converted feature vector representing the posted text content and the feature vector describing its writing style as an attribute associated with this posted text.

(2) Posted photo and photography style extraction. To represent the content of a posted photo, we propose to utilize *image2vec* [15] to convert it into a fixed length feature vector (empirically, we set the dimension to 100). Since drug traffickers in darkent markets have to prove the possession of illegal drugs by posting their own product photos, their distinct photography style might be revealed by the posted photos. We propose to capture the photography style by extracting its *low-level* and *high-level* features. 1) ***Low-level features*** refer to the information that can be directly obtained from a photo's exchangeable image file format (EXIF) data, which include *i*) camera make and model, *ii*) camera angle, *iii*) exposure time, *iv*) focal length, and *v*) image size. 2) ***High-level features*** are extracted from the photo's original content. We first convert the photo into its HSV (hue, saturation, value) representation and then extract the following five types of high-level features: *i*) colorfulness, *ii*) exposure of light, *iii*) saturation, *iv*) hue count, and *v*) contrast. In our current implementation, colorfulness, exposure of light and saturation are calculated using the method in [7]; while hue count and contrast are measured by [24]. For each posted photo by a vendor, we then concatenate its converted feature vector representing the posted photo content and the feature vector describing its photography style as an attribute associated with this posted photo.

(3) Attributed features of vendors and drugs. Besides the above extracted features, vendors' basic information and drugs they sell also play an important role in resolving their identities. Therefore, we further extract three kinds of features to depict each vendor: *username*, *PGP key* and *contact information*. Note that, for username, we first apply standard string matching techniques to measure the similarity of two usernames, if their similarity is greater than a user-specific threshold, we regard these two usernames as the same (e.g., "MF***Jones" and "MF***J0nes"). For each drug, we further extract its *category, escrow information* and *shipping information* (e.g., from where and to where). Then, we apply one-hot encoding [41] to convert the extracted features to a binary feature vector to be an attribute associated with each vendor/drug.

(4) Relation-based Features. In order to characterize the rich relations among vendors, drugs, posted texts and photos, we further extract the following relation-based features: 1) ***R1***: the *vendor-sell-drug* relation indicates whether a vendor sells a drug; 2) ***R2***: the *vendor-write-text* relation denotes if a vendor writes a text; 3) ***R3***: the *vendor-post-photo* relation indicates whether a vendor posts a photo; 4) ***R4***: the *text-describe-drug* relation denotes whether a text describes a drug; 5) ***R5***: the *photo-characterize-drug* relation indicates if a photo characterizes a drug.

2.2 AHIN Construction

Though heterogeneous information network (HIN) [35] has shown the success of modeling different types of entities and relations, it has limited capability of modeling additional attributes attached to entities. Thus, to depict vendors, drugs, texts, photos and their associated attributes as well as the rich relationships among them, we propose to use attributed HIN (AHIN) for representation.

Definition 2.1. ***Attributed heterogeneous information network (AHIN)*** [26]. Let $\mathcal{T} = \{T_1, ..., T_m\}$ be a set of m entity types. For each entity type T_i, let $\mathcal{X}_i$ be the set of entities of type T_i and A_i be the set of attributes defined for entities of type T_i. An entity x_j of type T_i is associated with an attribute vector $\mathbf{f}_j = (f_{j1}, f_{j2}, ..., f_{j|A_i|})$. An AHIN is defined by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ with an *entity* type

mapping $\phi: \mathcal{V} \rightarrow \mathcal{T}$ and a *relation* type mapping $\psi: \mathcal{E} \rightarrow \mathcal{R}$, where $\mathcal{V} = \bigcup_{i=1}^{m} \mathcal{X}_i$ denotes the entity set and $\mathcal{E}$ is the relation set, $\mathcal{T}$ denotes the entity type set and $\mathcal{R}$ is the relation type set, $\mathcal{A} = \bigcup_{i=1}^{m} A_i$, and the number of entity types $|\mathcal{T}| > 1$ or the number of relation types $|\mathcal{R}| > 1$. The ***network schema*** [26] for an AHIN $\mathcal{G}$, denoted by $\mathcal{T}_\mathcal{G} = (\mathcal{T}, \mathcal{R})$, is a graph with nodes as entity types from $\mathcal{T}$ and edges as relation types from $\mathcal{R}$.

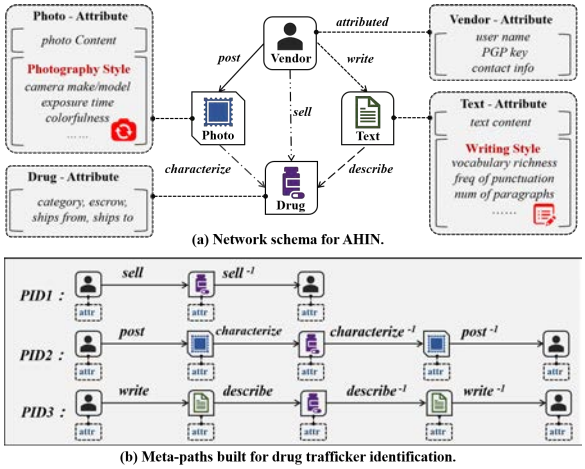


Figure 3: Network schema and meta-paths.

In our application, we have four entity types and five types of relations among them; meantime, each entity is also attached with an extracted feature vector representing its associated attributes. Based on the definitions above, the network schema for AHIN in our case is shown in Figure 3.(a) (to facilitate the illustration, attribute information is shown in its original form). Then, we adopt the concept of meta-path [35] to formulate higher-order relationships among entities in AHIN. In our application, we focus on three most meaningful meta-paths (i.e., ***PID1-PID3*** as shown in Figure 3.(b)) to jointly characterize the relatedness between two vendors from different views: (1) ***PID1*** means that two vendors can be connected through the path that they both sell the same kind of drug (e.g., heroin); (2) ***PID2*** denotes that two vendors can be linked if their posted photos describe the same kind of drug (e.g., as shown in Figure 1.(c), the vendor with username of "sp3***" in SilkRoad2 and the vendor with username of "sho***" in Evolution can be linked via this meta-path); (3) ***PID3*** indicates that two vendors can be connected if their written texts describe the same kind of drug.

2.3 *Vendor2Vec*

To reduce the computation and space cost in network mining, scalable representation learning method [10, 14] for AHIN is in need. We formalize the AHIN representation learning problem as below.

Definition 2.2. ***AHIN Representation Learning*** [10, 14]. Given an AHIN $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, the representation learning task is to learn a function $f: \mathcal{V} \rightarrow \mathbb{R}^D$ that maps each node $v \in \mathcal{V}$ to a vector in a D-dimensional space $\mathbb{R}^D$, $D \ll |\mathcal{V}|$ that are capable of preserving both structural and semantic relations among them.

To solve this problem, we propose a novel *attribute-aware* AHIN embedding model named *Vendor2Vec* which consists of attribute-aware meta-path random walk and skip-gram model. Given an AHIN $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ with schema $\mathcal{T}_\mathcal{G} = (\mathcal{T}, \mathcal{R})$, and a meta-path scheme $\mathcal{P}$ in the basic form: $T_1 \rightarrow \cdots T_t \rightarrow T_{t+1} \cdots \rightarrow T_L \rightarrow T_{L+1}$, we use attribute-aware meta-path to guide a random walker in AHIN, the transition probability at step i is calculated as:

$$p(v^{i+1}|v_t^i, \mathcal{P}) = \begin{cases} \frac{sim(\mathbf{f}_{v'}, \mathbf{f}_{v^{i+1}})}{\sum_{v^c \in N_{T_{t+1}}(v_t^i)} sim(\mathbf{f}_{v'}, \mathbf{f}_{v^c})} & (v_t^i, v^{i+1}) \in \mathcal{E}, \phi(v^{i+1}) = \phi(v') = T_{t+1} \\ \frac{1}{|N_{T_{t+1}}(v_t^i)|} & (v_t^i, v^{i+1}) \in \mathcal{E}, \phi(v^{i+1}) = T_{t+1}, v' = \emptyset \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where v' denotes the latest entity the walker visited is with the same type of v^{i+1}, $sim(\mathbf{f}_{v^i}, \mathbf{f}_{v^j})$ is the similarity between two entities' attribute vectors (e.g, it can be calculated by using cosine similarity measure), ϕ is the node type mapping function, $N_{T_{t+1}}(v_t^i)$ denote T_{t+1} type of neighborhood of node v_t^i, v^c denotes a node in $N_{T_{t+1}}(v_t^i)$. Since we have three different meta-paths, we simply combine the path instances sampled via each meta-path, and feed them into the skip-gram model [29] to learn the node embeddings.

2.4 Classification Model

The problem of determining if a given pair of vendors are the same individual can be considered as a link prediction in an AHIN. In this paper, motivated by [1], we propose to devise a classification model for AHIN (named *vIdentifier* as shown in Figure 2.(d)) to predict the likelihood of the link between two nodes.

We first apply a Deep Neural Network (DNN) into node embedding to learn $f_\theta: \mathbb{R}^D \rightarrow \mathbb{R}^d$ that maps a vendor's learned latent representation onto a low-dimensional manifold. The mapping function f_θ is defined as [1]:

$$\begin{aligned} f_\theta: Y_u &\rightarrow FC_{\{\mathbf{W}_1, \mathbf{b}_1\}} \rightarrow BatchNorm \rightarrow relu \\ &\rightarrow FC_{\{\mathbf{W}_2, \mathbf{b}_2\}} \rightarrow BatchNorm \rightarrow f_\theta(Y_u), \end{aligned} \tag{2}$$

where $FC_{\{\mathbf{W}, \mathbf{b}\}}$ is a fully-connected layer with weight matrix $\mathbf{W}$ and bias vector $\mathbf{b}$, BatchNorm is described in [22], $relu(x) = max(0, x)$ is an element-wise activation function, and $\theta = \{\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2, \mathbf{b}_2, ...\}$.

Given a pair of vendors, to define a general link function $g(u, v) \in \mathbb{R}$, we consider a low-rank affine projection in the manifold space [1]: $g(u, v) = f(Y_u)^T \times M \times f(Y_v)$, where the low-rank projection matrix $M = L \times R$ with $L \in \mathbb{R}^{d \times b}$ and $R \in \mathbb{R}^{b \times d}$, $b < d < D$. We can factor $g(u, v)$ into an inner product $\langle L^T f(Y_u), Rf(Y_v) \rangle$.

Then, we utilize the method proposed in [1] to conduct the optimization. The function is devised as following:

$$Pr(\mathcal{G}) \propto \prod_{u,v \in \mathcal{V}} \sigma(g(u, v))^{D_{uv} f(u,v)} (1 - \sigma(g(u, v)))^{1 - f(u,v)}, \tag{3}$$

where $\sigma(x) = 1/(1 + exp(-x))$ is the standard logistic,

$$f(u, v) = \begin{cases} 1 & \text{if } u \text{ and } v \text{ are the same individual} \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

D_{uv} is the frequency that vendors u and v co-occur within a specific window in the path instances sampled by our above proposed attribute-aware meta-path random walk.

3 EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we conduct three sets of experiments using data collections from darknet markets to fully evaluate the performance of our developed system *uStyle-uID* for drug trafficker identification.

3.1 Experimental Setup

To fully evaluate our proposed method, we have collected the data from four different darknet markets *Valhalla*, *Dream Market*, *SilkRoad2* and *Evolution*. For the former two darknet markets, we develop a set of crawling tools to scrape weekly snapshots from June 2017 to August 2017. For the rest of markets, we collect their public data dumps. After data collection, we merely retain the vendors who at least posted two different drugs, each of which should be at least with one transaction. We summarize the collected data in Table 1. Due to the anonymity of darknet markets, it is difficult to access the actual ground-truth. Similar to the approach used in [40], we propose an alternative solution: for a given vendor, we randomly split his/her posted texts and photos into two even parts to form a positive example; then we randomly match this given vendor to the other in the same darknet market to generate a negative example. In this pseudo setting, the attribute vector attached to each vendor node is set as null. In the following experiments, we conduct ten-fold cross validations and use accuracy (*ACC*) and *F1* as performance measures to evaluate different methods. The parameters for *Vendor2Vec* are empirically set as follows: node dimension $D = 100$, walks per node $r = 10$, walk length $l = 80$ and window size $w = 10$, while for the parameter of d in *vIdentifier* is set as 30 and the remaining ones are consistent with [1].

Table 1: Information of the dataset.

Dataset	Vendors	Drugs	Entities	Relations
Valhalla (Val)	522	13,150	27,535	54,698
Dream Market (DM)	2,547	67,270	139,493	281,080
SilkRoad2 (SR2)	681	15,231	31,354	66,648
Evolution (Evol)	1,650	36,798	79,451	164,792

3.2 Comparisons of Different Features

In this set of experiments, we first evaluate the effectiveness of different features for drug trafficker identification.

- **Text-based features**: (1) *text content only* (***f-1***), (2) *writing style only* (***f-2***), and (3) *text content and writing style* (***f-3***).
- **Photo-based features**: (1) *photo content* (***f-4***), (2) *photography style* (***f-5***), and (3) *photo content and photography style* (***f-6***).

The experimental results are illustrated in Table 2, from which we can see that different features show different performances. To put this into perspective, (1) the relatedness over vendors depicted by style-based correlations (***f-2*** and ***f-5***) perform better than content-based correlations (***f-1*** and ***f-4***); (2) feature engineering (***f-3*** and ***f-6***) helps the performance; (3) the photo-based features (***f-4***, ***f-5***, and ***f-6***) perform better than the text-based ones (***f-1***, ***f-2***, and ***f-3***); (4) our proposed method *uStyle-uID* which integrates different levels of semantics, leveraging both writing and photograph styles, obtains a significantly better performance.

Table 2: Comparisons of different features.

Metric	Method	Feature	Val	DM	SR2	Evol
ACC	Text-based	***f-1***	0.780	0.798	0.782	0.792
		f-2	0.792	0.807	0.795	0.805
		f-3	0.803	0.819	0.806	0.815
	Photo-based	***f-4***	0.795	0.808	0.796	0.805
		f-5	0.807	0.817	0.808	0.816
		f-6	0.818	0.828	0.819	0.827
	uStyle-uID	/	**0.876**	**0.903**	**0.881**	**0.889**
F1	Text-based	***f-1***	0.782	0.784	0.772	0.784
		f-2	0.796	0.788	0.784	0.790
		f-3	0.804	0.809	0.792	0.797
	Photo-based	***f-4***	0.785	0.794	0.792	0.793
		f-5	0.798	0.806	0.796	0.802
		f-6	0.810	0.817	0.806	0.809
	uStyle-uID	/	**0.865**	**0.894**	**0.868**	**0.879**

3.3 Network Embedding Model Comparisons

In this set of experiments, we evaluate our proposed method *Vendor2Vec* by comparisons with several state-of-the-art network embedding models including **DeepWalk** [31], **node2vec** [18] and **metapath2vec** [10]. For these embedding methods, we use the same parameters as *Vendor2Vec*. The results from Table 3 show that *Vendor2Vec* consistently and significantly outperforms all state-of-the-art embedding models. The success of *Vendor2Vec* lies in: (1) the proper consideration and accommodation of the heterogeneous property of AHIN; (2) the advantage of the attribute setting and the proposed attribute-aware meta-path guided random walk for sampling the high-quality path instances (i.e., without the attribute information such as writing and photography styles, the generated path instances are of low quality and less useful to our application).

Table 3: Comparisons of network embedding models.

Metric	Method	Val	DM	SR2	Evol
ACC	DeepWalk	0.703	0.714	0.699	0.707
	node2vec	0.726	0.731	0.722	0.729
	metapath2vec	0.741	0.754	0.744	0.748
	Vendor2Vec	**0.876**	**0.903**	**0.881**	**0.889**
F1	DeepWalk	0.682	0.688	0.669	0.679
	node2vec	0.703	0.710	0.692	0.709
	metapath2vec	0.718	0.735	0.727	0.731
	Vendor2Vec	**0.865**	**0.894**	**0.868**	**0.879**

3.4 Comparisons with Alternative Approaches

In this set of experiments, based on our collected datasets, we compare our developed system *uStyle-uID* with alternative approaches:

(1) feeding all the features (i.e., ***f-3***, ***f-6***, and feature vectors of vendors and drugs) into a generic DNN [19] to make the identification (denoted as *Hybrid-DNN*); (2) replacing the *vIdentifier* in *uStyle-uID* by a generic DNN (denoted as *AHIN-DNN*); (3) replacing the *vIdentifier* in *uStyle-uID* by SVM (denote as *AHIN-SVM*). For the generic DNN, we implement the model in Keras [40] and retain the default parameters. The experimental results are illustrated in Table 4. From the results we can observe that *AHIN-DNN* added the knowledge represented as AHIN performs better than *Hybrid-DNN*, which shows that using meta-path based approach over AHIN is able to build the higher-level semantic connection between vendors with a more expressive view. We also note that *uStyle-uID* significantly outperforms other baselines, which demonstrates that *uIdentifier* indeed helps the performance compared with the generic DNN and state-of-the-art shallow learning classification model.

Table 4: Comparisons of other alternative approaches.

Metric	Method	Val	DM	SR2	Evol
ACC	Hybrid-DNN	0.831	0.839	0.833	0.838
	AHIN-DNN	0.854	0.876	0.856	0.863
	AHIN-SVM	0.843	0.851	0.847	0.848
	uStyle-uID	**0.876**	**0.903**	**0.881**	**0.889**
F1	Hybrid-DNN	0.809	0.818	0.812	0.814
	AHIN-DNN	0.841	0.864	0.845	0.855
	AHIN-SVM	0.832	0.837	0.832	0.840
	uStyle-uID	**0.865**	**0.894**	**0.868**	**0.879**

4 CROSS-MARKET DRUG TRAFFICKER IDENTIFICATION AND CASE STUDIES

To better understand and gain deeper insights into the ecosystem of drug trafficking in darknet markets, we further apply our developed system *uStyle-uID* for cross-market drug trafficker identification. For the detected cross-market vendor pairs, we further sample 798 pairs and validate them using conclusive evidences. Among these 798 detected cross-market pairs, 726 pairs (90.09%) are with high confidence that they are the same individuals and 22 pairs are uncertain (2.76%). As shown in Figure 4, for one of our detected cross-market vendor pairs, though "The***shop" on Evolution and "***Store" on Valhalla have different usernames, PGP keys and contact information, they sell similar drugs (i.e., heroin and MDMA) with similar writing and photography styles. After further investigation , we find that "The***shop" and "***Store" are both the members of a group named "***DrugShop" in Finland according to the description in the terms and conditions of vendors. This indicates that they might work together as an organization and we can link them to the same group of drug traffickers. Such kind of information can provide investigative insight for law enforcement to trace their activities and thus to build proactive defenses.

5 RELATED WORK

To combat drug trafficking in darknet markets, there have been many research efforts on darknet market data analysis [3, 9, 20, 40]. Among these studies, there have been some methods proposed



(a) Vendor "The***shop" on Evolution (b) Vendor "***Store" on Valhalla

Figure 4: An example of detected cross-market vendor pair.

to tackle the challenges of authorship identification such as [20, 40]. However, these approaches mainly relied on either stylometry analysis or photography style analysis. Different from existing works, we propose to leverage both writing and photography styles together with their contents for drug trafficker identification.

In order to depict different entities, associated attributes and the rich relationships among them, it is important to model them properly. Though HIN has shown the success of modeling different types of entities and relations [11–13, 21, 35, 42, 43], it has limited capability of modeling additional attributes attached to entities. To address this challenge, we propose to use AHIN for representation. To better address representation learning for HIN, many efficient network embedding methods have been proposed such as metagraph2vec [11], metapath2vec [10], HIN2vec [14]. However, these models are unable to deal with the attribute information associated with each entity. To address this issue, we propose *Vendor2Vec* to learn the desirable node representations in AHIN.

6 CONCLUSION

To combat drug trafficking, in this paper, we design and develop an intelligent system named *uStyle-uID* to automate drug trafficker identification in darknet markets. In *uStyle-uID*, we propose to leverage both writing and photography styles at the first attempt. To depict vendors, drugs, texts, photos and their associated attributes as well as the rich relationships among them, we present a structural AHIN to model them which gives the vendors higher-level semantic representations. Then, a meta-path based approach is used to characterize the semantic relatedness over vendors. To efficiently measure the relatedness over vendors in AHIN, we propose a new network embedding model *Vendor2Vec* which leverages complementary attribute information attached in the nodes to guide the meta-path based random walk for path instances sampling. Then, we transform the identification task into a link prediction problem and further present a learning model named *vIdentifier* to solve the problem. The promising experimental results on the collected datasets from four darknet markets demonstrate that *uStyle-uID* outperforms alternative approaches.

7 ACKNOWLEDGMENTS

Y. Zhang, Y. Fan, W. Song, S. Hou, Y. Ye and X. Li's work is partially supported by the DoJ/NIJ under grant NIJ 2018-75-CX-0032; the NSF under grants CNS-1618629, CNS-1814825, CNS-1845138 and OAC-1839909; the WV Higher Education Policy Commission Grant (HEPC.dsr.18.5); and the WVU Research and Scholarship Advancement Grant (R-844).

REFERENCES

[1] Sami Abu-El-Haija, Bryan Perozzi, and Rami Al-Rfou. 2017. Learning edge representations via low-rank asymmetric projections. In *CIKM*. ACM, 1787–1796.

[2] Sadia Afroz, Aylin Caliskan Islam, Ariel Stolerman, Rachel Greenstadt, and Damon McCoy. 2014. Doppelgänger finder: Taking stylometry to the underground. In *Security and Privacy (SP), 2014 IEEE Symposium on*. IEEE, 212–226.

[3] Julian Broséus, Damien Rhumorbarbe, Caroline Mireault, Vincent Ouellette, Frank Crispino, and David Décary-Hétu. 2016. Studying illicit drug trafficking on Darknet markets: structure and organisation from a Canadian perspective. *Forensic science international* 264 (2016), 7–14.

[4] Julia Buxton and Tim Bingham. 2015. The rise and challenge of dark net drug markets. *Policy Brief* 7 (2015).

[5] Prima Chairunnanda, Nam Pham, and Urs Hengartner. 2011. Privacy: Gone with the Typing! Identifying Web Users by Their Typing Patterns.. In *SocialCom/PASSAT*. 974–980.

[6] Richard Colbaugh and Kristin Glass. 2011. Proactive defense for evolving cyber threats. In *ISI*. IEEE, 125–130.

[7] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z Wang. 2006. Studying aesthetics in photographic images using a computational approach. In *ECCV*. Springer, 288–301.

[8] Hakan Demirbuken and others. 2011. The Global Afghan Opium Trade: A Threat Assessment. *UNODC Report* (2011), 3–4.

[9] Martin Dittus, Joss Wright, and Mark Graham. 2018. Platform Criminalism: The'last-mile'geography of the darknet market supply chain. In *WWW*. International World Wide Web Conferences Steering Committee, 277–286.

[10] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD*. ACM, 135–144.

[11] Yujie Fan, Shifu Hou, Yiming Zhang, Yanfang Ye, and Melih Abdulhayoglu. 2018. Gotcha-sly malware!: Scorpion a metagraph2vec based malware detection system. In *KDD*. ACM, 253–262.

[12] Yujie Fan, Yiming Zhang, Yanfang Ye, and Xin Li. 2018. Automatic Opioid User Detection from Twitter: Transductive Ensemble Built on Different Meta-graph Based Similarities over Heterogeneous Information Network.. In *IJCAI*. 3357–3363.

[13] Yujie Fan, Yiming Zhang, Yanfang Ye, Wanhong Zheng, and others. 2017. Social media for opioid addiction epidemiology: Automatic detection of opioid addicts from twitter and case studies. In *CIKM*. ACM, 1259–1267.

[14] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. HIN2Vec: Explore Meta-paths in Heterogeneous Information Networks for Representation Learning. In *CIKM*. ACM, 1797–1806.

[15] Dario Garcia-Gasulla, Eduard Ayguadé, Jesús Labarta, Javier Béjar, Ulises Cortés, Toyotaro Suzumura, and R Chen. 2017. A visual embedding for the unsupervised extraction of abstract semantics. *Cognitive Systems Research* 42 (2017), 73–81.

[16] Michael Gilbert and Nabarun Dasgupta. 2017. Silicon to syringe: Cryptomarkets and disruptive innovation in opioid supply chains. *International Journal of Drug Policy* 46 (2017), 160–167.

[17] P. Griffiths and J. Mounteney. 2017. Disruptive potential of the internet to transform illicit drug markets and impact on future patterns of drug consumption. *Clinical Pharmacology & Therapeutics* 101, 2 (2017).

[18] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*. ACM, 855–864.

[19] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, and others. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine* 29, 6 (2012), 82–97.

[20] Thanh Nghia Ho and Wee Keong Ng. 2016. Application of stylometry to darkweb forum user identification. In *ICS*. Springer, 173–183.

[21] Shifu Hou, Yanfang Ye, Yangqiu Song, and Melih Abdulhayoglu. 2017. Hindroid: An intelligent android malware detection system based on structured heterogeneous information network. In *KDD*. ACM, 1507–1515.

[22] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[23] Dev Kar and Joseph Spanjers. 2017. Transnational crime and the developing world. (2017).

[24] Yan Ke, Xiaoou Tang, and Feng Jing. 2006. The design of high-level features for photo quality assessment. In *CVPR*, Vol. 1. IEEE, 419–426.

[25] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *ICML*. 1188–1196.

[26] Xiang Li, Yao Wu, Martin Ester, Ben Kao, Xin Wang, and Yudian Zheng. 2017. Semi-supervised clustering in attributed heterogeneous information networks. In *WWW*. 1621–1629.

[27] Dream Market. 2018. *Dream Market*. http://6khhxwj7viwe5xjm.onion.

[28] James Martin. 2014. Drugs on the dark net: How cryptomarkets are transforming the global trade in illicit drugs. (2014).

[29] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[30] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[31] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*. ACM, 701–710.

[32] Damien Rhumorbarbe, Ludovic Staehli, Julian Broseus, Quentin Rossy, and Pierre Esseiva. 2016. Buying drugs on a Darknet market: A better deal? Studying the online illicit drug market through the analysis of digital, physical and chemical data. *Forensic science international* 267 (2016), 173–182.

[33] SilkRoad. 2018. *SilkRoad*. http://silkroad7rn2puhj.onion.

[34] Martijn Spitters, Femke Klaver, Gijs Koot, and Mark van Staalduinen. 2015. Authorship analysis on dark marketplace forums. In *EISIC*.

[35] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *VLDB Endowment* 4, 11 (2011), 992–1003.

[36] Fiona J Tweedie and R Harald Baayen. 1998. How variable may a constant be? Measures of lexical richness in perspective. *Computers and the Humanities* 32, 5 (1998), 323–352.

[37] Valhalla. 2018. *Valhalla*. http://valhallaxmn3fydu.onion.

[38] Joe Van Buskirk, Amanda Roxburgh, Michael Farrell, and Lucy Burns. 2014. The closure of the Silk Road: What has this meant for online drug trading? *Addiction* 109, 4 (2014).

[39] Joe Van Buskirk, Amanda Roxburgh, Sundresan Naicker, and Lucinda Burns. 2015. A response to Dolliver's "Evaluating drug trafficking on the Tor network". *International Journal of Drug Policy* 26, 11 (2015).

[40] Xiangwen Wang, Peng Peng, Chun Wang, and Gang Wang. 2018. You are your photographs: Detecting multiple identities of vendors in the darknet marketplaces. In *ACCCS*. ACM, 431–442.

[41] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*. 649–657.

[42] Yiming Zhang, Yujie Fan, Shifu Hou, Jian Liu, Yanfang Ye, and Thirimachos Bourlai. 2018. iDetector: Automate Underground Forum Analysis Based on Heterogeneous Information Network. In *ASONAM*. IEEE, 1071–1078.

[43] Yiming Zhang, Yujie Fan, Yanfang Ye, Liang Zhao, Jiabin Wang, Qi Xiong, and Fudong Shao. 2018. KADetector: Automatic Identification of Key Actors in Online Hack Forums Based on Structured Heterogeneous Information Network. In *ICBK*. IEEE, 154–161.

Abstractive Meeting Summarization via Hierarchical Adaptive Segmental Network Learning

Zhou Zhao*†
Zhejiang University
zhaozhou@zju.edu.cn

Haojie Pan*
The Hong Kong University of Science and Technology
hpanad@cse.ust.hk

Changjie Fan
NetEase Fuxi AI Lab
fanchangjie@corp.netease.com

Yan Liu
University of Southern California
yanliu.cs@usc.edu

Linlin Li
Alibaba Group
linyan.lll@alibaba-inc.com

Min Yang
Shenzhen Institutes of Advanced Technology
min.yang@siat.ac.cn

Deng Cai
State Key Lab of CAD & CG, Zhejiang University
dengcai@gmail.com

ABSTRACT

Abstractive meeting summarization is a challenging problem in natural language understanding, which automatically generates the condensed summary covering the important points in the meeting conversation. However, the existing abstractive summarization works mainly focus on the structured text documents, which may be ineffectively applied to the meeting summarization task due to the lack of modeling the unstructured long-form conversational contents. In this paper, we consider the problem of abstractive meeting summarization from the viewpoint of hierarchical adaptive segmental encoder-decoder network learning. We propose the hierarchical neural encoder based on adaptive recurrent networks to learn the semantic representation of meeting conversation with adaptive conversation segmentation. We then develop the reinforced decoder network to generate the high-quality summaries for abstractive meeting summarization. We conduct the extensive experiments on the well-known AMI meeting conversation dataset to validate the effectiveness of our proposed method.

CCS CONCEPTS

• **Information systems** → **Retrieval tasks and goals**; **Information retrieval**; **Summarization**.

KEYWORDS

Abstractive Meeting Summarization, Adaptive Network Learning, Neural Networks

*These two authors contributed equally to the paper.
†Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313619

ACM Reference Format:
Zhou Zhao, Haojie Pan, Changjie Fan, Yan Liu, Linlin Li, Min Yang, and Deng Cai. 2019. Abstractive Meeting Summarization via Hierarchical Adaptive Segmental Network Learning. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313619

1 INTRODUCTION

Abstractive meeting summarization is a challenging task in natural language understanding, which could be of great value to the users by providing quick access to the important content of past meetings [24]. The essential problem of abstractive meeting summarization is to automatically generate the condensed summary covering the important points in the meeting conversation. In order to provide the high-quality summaries, many existing works learn the semantic representation to capture the overall content of the conversation for abstractive meeting summarization. Currently, the existing meeting summarization works [2, 10, 12, 16, 21, 24] generate the meeting summaries mainly based on the extracted semantic rules, which suffer from the low maintainability [23]. On the other hand, most of the existing abstractive summarization approaches [3, 14, 26] mainly focus on the structured text documents, which learn the semantic representation of text documents and then generate the summaries for abstractive document summarization. Although these works have achieved great performance in abstractive document summarization, they may still be ineffectively applied to meeting summarization due to the lack of modeling the unstructured long-form meeting conversation contents.

The meeting conversation contents are the sequential collection of unstructured utterances, which have long-term semantic dependencies [24]. Recently, the hierarchical neural encoder [8] has been proposed to learn the paragraph-level semantic representation for document modeling. We then employ the hierarchical neural encoder to learn the high-level semantic representation that models the long-term semantic dependencies in the meeting conversation contents. On the other hand, although the sequential utterances in

the meeting conversation are topically consistent, they are unstructured and have different semantic contents. To tackle this issue, we employ the binary neurons [4] and then develop the hierarchical recurrent encoder based on adaptive binary neural networks. The adaptive hierarchical encoder network then learns the high-level semantic representation of meeting conversation with adaptive conversation segmentation. Thus, leveraging hierarchical neural encoder with adaptive binary neural networks is important for modeling the unstructured long-form meeting conversation contents for abstractive meeting summarization.

In this paper, we present the problem of abstractive meeting summarization from the viewpoint of hierarchical adaptive segmental encoder-decoder network learning. We first propose the hierarchical neural encoder with adaptive recurrent networks to learn the high-level semantic representation of meeting conversation contents with adaptive conversation segmentation. We then develop the the reinforced decoder networks to generate the high-quality summaries for abstractive meeting summarization. We next devise the Hierarchical encoder-decoder network learning framework with Adaptive conversation Segmentation for abstractive meeting summarization, named as HAS. When a certain meeting conversation is issued, HAS can automatically generate the natural language summaries for it based on its conversation contents. The main contributions of this paper are as follows:

- Unlike the previous studies, we study the problem of abstractive meeting summarization from the viewpoint of hierarchical encoder-decoder network learning framework with adaptive conversation segmentation.
- We propose the hierarchical adaptive encoder to learn the high-level semantic representation of meeting conversation contents, and then devise the reinforced decoder network to generate the summaries for abstractive meeting summarization.
- We validate the effectiveness of our proposed method through extensive experiments on the well-known AMI meeting conversation dataset.

The rest of this paper is organized as follows. In Section 2, we introduce the problem of abstractive meeting summarization from the viewpoint of hierarchical encoder-decoder network learning framework with adaptive conversation segmentation. A variety of experimental results are presented in Section 3. We provide a brief review of the related work about meeting summarization and abstractive document summarization in Section 4. Finally, we provide some concluding remarks in Section 5.

2 ABSTRACTIVE MEETING SUMMARIZATION VIA HIERARCHICAL ADAPTIVE SEGMENTAL NETWORKS

In this section, we first present the problem of abstractive meeting summarization from the viewpoint of hierarchical adaptive segmental encoder-decoder network learning framework. We then introduce the hierarchical adaptive segmental encoder networks that learn the high-level semantic representation of meeting conversation contents. We next illustrate the reinforced decoder networks to generate the summaries for abstractive meeting summarization.

2.1 The Problem

Before presenting the learning framework, we first introduce some basic notions and terminologies. We denote the utterances in meeting conversation by $\mathbf{u} \in U$ and its meeting summary by $\mathbf{z} \in Z$, where U and Z are the sets of meeting conversations and meeting summaries. The sequential collection of utterances in meeting conversation are denoted by $\mathbf{u} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n)$, where $\mathbf{u}_i$ is the i-th utterance and n is the number of utterances in meeting conversation $\mathbf{u}$. The word-level representation of utterance $\mathbf{u}_i$ is denoted by $\mathbf{u}_i = (\mathbf{u}_{(i,1)}, \mathbf{u}_{(i,2)}, \ldots, \mathbf{u}_{(i,m_i)})$, where $\mathbf{u}_{(i,j)}$ is the embedding vector for the j-th word and m_i is the number of the words in utterance $\mathbf{u}_i$. The word-level representation of meeting summary $\mathbf{z}$ is given by $\mathbf{z} = (z_1, z_2, \ldots, z_{l_z})$, where z_j is the j-th word and l_z is the number of words in summary $\mathbf{z}$. Since the meeting conversation and its utterances are sequential data with variant length, it is natural to choose the variant recurrent neural network called long-short term memory network (LSTM) [7] to learn their feature representations by

$$\mathbf{i}_t = \delta(\mathbf{W}_i \mathbf{x}_t + \mathbf{G}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \quad (1)$$

$$\hat{\mathbf{c}}_t = tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{G}_t \mathbf{h}_{t-1} + \mathbf{b}_f) \quad (2)$$

$$\mathbf{f}_t = \delta(\mathbf{W}_f \mathbf{x}_t + \mathbf{G}_f \mathbf{h}_{t-1} + \mathbf{b}_f) \quad (3)$$

$$\mathbf{c}_t = \mathbf{i}_t \cdot \hat{\mathbf{c}}_t + \mathbf{f}_t \cdot \mathbf{c}_{t-1} \quad (4)$$

$$\mathbf{o}_t = \delta(\mathbf{W}_o \mathbf{x}_t + \mathbf{G}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \quad (5)$$

$$\mathbf{h}_t = \mathbf{o}_t \cdot tanh(\mathbf{c}_t) \quad (6)$$

where δ represents the sigmoid activation function; $\mathbf{W}$s, $\mathbf{G}$s are the weight matrices, and $\mathbf{b}$s are the bias vectors. The memory cell $\mathbf{c}_t$ maintains the history of the inputs observed up to the timestep. Update operations on the memory cell are modulated by three gates $\mathbf{i}_t$, $\mathbf{f}_t$ and $\mathbf{o}_t$, which are all computed as a combination of the current input $\mathbf{x}_t$ and of the previous hidden state $\mathbf{h}_{t-1}$, followed by a sigmoid activation. The input gate $\mathbf{i}_t$ controls how the current input should be added to the memory cell; the forget gate $\mathbf{f}_t$ is used to control what the cell will forget from the previous memory $\mathbf{c}_{t-1}$, and the output gate $\mathbf{o}_t$ controls whether the current memory cell should be passed as output.

Specifically, we learn the semantic representation of utterance $\mathbf{u}_i$ by word-level bi-directional LSTM networks, which consists of a forward LSTM and a backward LSTM. The backward LSTM has the same network structure with the forward one while its input sequence is reversed. We denote the hidden state of the forward LSTM for the t-th word in utterance $\mathbf{u}_i$ by $\mathbf{h}_{(i,t)}^{(w,f)}$, and the hidden state of this word in the backward LSTM by $\mathbf{h}_{(i,m_i-t)}^{(w,b)}$. Thus, we take the output of the last bi-directional LSTM cell, $\mathbf{h}_{(i,m_i)}^{(w)} = [\mathbf{h}_{(i,m_i)}^{(w,f)}, \mathbf{h}_{(i,1)}^{(w,b)}]$ as the semantic representation of utterance $\mathbf{u}_i$. The semantic representation of the sequential collection of the utterances in the meeting conversation is given by utterance-level LSTM networks, denoted by $\mathbf{h}^{(u)} = (\mathbf{h}_1^{(u)}, \mathbf{h}_2^{(u)}, \ldots, \mathbf{h}_n^{(u)})$, where $\mathbf{h}_i^{(u)}$ is the hidden state in the utterance-level LSTM network.

Using the notations above, the problem of abstractive meeting summarization is formulated as follows. Given the collection of utterances in the meeting conversation $\mathbf{u}$, our goal is to learn the

encoder-decoder network model $g(f(\mathbf{u}))$, where the encoder network $f(\mathbf{u})$ learns the semantic representation of the meeting conversation contents, and the decoder network $\hat{\mathbf{z}} = g(f(\mathbf{u}))$ generates natural language summary $\hat{\mathbf{z}}$ for abstractive meeting summarization. We present the details of the hierarchical adaptive segmental encoder-decoder network learning framework in Figure 1.

2.2 Adaptive Encoder Networks

In this section, we propose the hierarchical adaptive segmental encoder $f(\cdot)$ to learn the high-level semantic representation of meeting conversation contents with adaptive conversation segmentation.

The meeting conversation contents contain a sequential collection of unstructured utterances, which have long-term semantic dependencies [24]. Furthermore, although the utterance in the meeting conversation are topically consistent, they have different conversational contents. Inspired by binary neurons [4], we propose the hierarchical adaptive encoder networks that learn the semantic representation of meeting conversation contents with adaptive conversation segmentation. We learn the semantic representation of sequential utterance collection from utterance-level LSTM networks by $\mathbf{h}^{(u)}(\mathbf{h}_1^{(u)}, \mathbf{h}_2^{(u)}, \ldots, \mathbf{h}_n^{(u)})$. When a certain conversational topic change is estimated in the current utterance, the conversation segmentation can be done by resetting the LSTM parameters of the next utterance. To enable the conversation segmentation, we define an adaptive recurrent neural networks with binary gate function, which decides whether to transfer the LSTM parameters (i.e., hidden state $\mathbf{h}_t^{(u)}$ and memory cell $\mathbf{c}_t^{(u)}$) of current utterance to update the LSTM parameters of the next utterance (i.e., hidden state $\mathbf{h}_{t+1}^{(u)}$ and memory cell $\mathbf{c}_{t+1}^{(u)}$) by Equations (1), (2), (3), (4) and (5). Therefore, the adaptive recurrent neural networks enable the conversation segment with variable number of utterances in the encoder network. Formally, the t-th binary gate $\alpha_t(\cdot)$ is defined as a step function, which is computed as a non-linear combination of the semantic representation of the the $t+1$-th utterance from the word-level LSTM networks (i.e., $\mathbf{h}^{(w)}_{(t+1,m_{t+1})}$), and the semantic representation of the tth utterance from the utterance-level LSTM networks (i.e., $\mathbf{h}_t^{(u)}$), given by

$$\begin{aligned}&\alpha_t(\mathbf{h}^{(w)}_{(t+1,m_{t+1})}, \mathbf{h}_t^{(u)}) = \\ &1[\delta(\mathbf{w}_\alpha^T(\mathbf{W}_{\alpha w}\mathbf{h}^{(w)}_{(t+1,m_{t+1})} + \mathbf{W}_{\alpha u}\mathbf{h}_t^{(u)} \\ &+\mathbf{b}_\alpha)) > 0.5].\end{aligned}$$

The $1[\cdot]$ is a step function and $\delta(\cdot)$ is a sigmoid function. The $\mathbf{w}_g$ is a learnable row vector, $\mathbf{W}_{\alpha w}$, $\mathbf{W}_{\alpha u}$ and $\mathbf{b}_\alpha$ are the learnable weights and biases. For example, the inputs of binary gate $\alpha_2(\mathbf{h}^{(w)}_{3,3}, \mathbf{h}_2^u)$ are the semantic representation of utterance $\mathbf{h}_2^u$ from the utterance-level LSTM networks and the semantic representation of utterance $\mathbf{h}^{(w)}_{3,3}$ from the word-level LSTM networks in Figure 1. Given the sequential representation of utterances $(\mathbf{h}_1^{(u)}, \mathbf{h}_2^{(u)}, \ldots, \mathbf{h}_n^{(u)})$ with the values of binary gate $(\alpha_1, \alpha_2, \ldots, \alpha_{n-1})$, we develop the segment-level LSTM networks as follows. If the value of binary gate $\alpha_t(\cdot) = 1$, the hidden state of utterance-level LSTM networks at time $t+1$, $\mathbf{h}_{t+1}^{(u)}$, is passed as the input to the segment-level LSTM networks. For example, the hidden states $\mathbf{h}_3^{(u)}$, $\mathbf{h}_6^{(u)}$, ..., $\mathbf{h}_N^{(u)}$ in utterance-level LSTM networks are passed as the inputs to the segment-level LSTM networks in Figure 1. We denote semantic representation of the sequential segment collection from meeting conversation contents in the segment-level LSTM networks by $\mathbf{h}^{(s)} = (\mathbf{h}_1^{(s)}, \mathbf{h}_2^{(s)}, \ldots, \mathbf{h}_k^{(s)})$, where $\mathbf{h}_j^{(s)}$ is the hidden state in the segment-level LSTM network and k is the number of conversational segments. Therefore, hierarchical adaptive segmental encoder network is given by $f(\mathbf{u}) = (\mathbf{h}_1^{(s)}, \mathbf{h}_2^{(s)}, \ldots, \mathbf{h}_k^{(s)})$. We note that the hierarchical adaptive segmental encoder network learns the semantic representation of meeting conversation contents from conversational segments with variable number of utterances.

2.3 Reinforced Decoder Networks

In this section, we propose the reinforced decoder network $g(\cdot)$ based on segment-level LSTM networks to generate the natural language summary for abstractive meeting summarization.

Given the encoded semantic representation of meeting conversation contents $f(\mathbf{u}) = (\mathbf{h}_1^{(s)}, \mathbf{h}_2^{(s)}, \ldots, \mathbf{h}_k^{(s)})$ from the segment-level LSTM networks, the decoder predicts the next word in summary by sampling $z_t \sim p_\theta(z_t|\mathbf{z}_{1:t-1}, f(\mathbf{u})) = g_t(\mathbf{h}_t^{(g)}, \mathbf{v}_t, f(\mathbf{u}))$, where $g_t(\cdot)$ is the recurrent generator. The $\mathbf{h}_t^{(g)}$ is the decoder state and $\mathbf{v}_t$ is the context vector at the t-th decoder step. The context vector $\mathbf{v}_t$ is computed as the weighted sum of the encoded semantic representation from the segment-level LSTM networks by $\mathbf{v}_t = \sum_{i=1}^k \beta_{ti}\mathbf{h}_i^{(s)}$. The attention weight β_{ti} is given by $\beta_{ti} = \frac{\exp((\mathbf{h}_{t-1}^{(g)})^T\mathbf{h}_i^{(s)})}{\sum_{i=1}^k \exp((\mathbf{h}_{t-1}^{(g)})^T\mathbf{h}_i^{(s)})}$. One common approach for training the decoder network is based on the maximum likelihood estimation, given by

$$\mathcal{L}_{ML}(g(f(\mathbf{u}))) = \sum_{t=1}^{l} \log p_\theta(z_t|\mathbf{z}_{1:t-1}, f(\mathbf{u})). \tag{7}$$

We note that when computing the conditional probability $p_\theta(z_t|\mathbf{z}_{1:t-1}, f(\mathbf{u}))$, the previous word in the human-authored summaries rather than the generated word is chosen for training in maximum likelihood estimation. However, this makes the learnt decoder network suboptimal [1]. To tackle this problem, we employ the reinforcement learning framework for training the decoder network.

In the setting of reinforcement learning, we define the generation of next word as action and the decoding probability $p_\theta(\hat{z}_t|\hat{\mathbf{z}}_{1:t-1}, f(\mathbf{u}))$ as the policy. In the task of abstractive meeting summarization, the reward is usually received at the end of the generated summaries. We then choose the ROUGE [9] as the reward function $R_\mathbf{z}(\hat{\mathbf{z}})$, which is calculated by comparing the generated summary $\hat{\mathbf{z}}$ with the human-written summary $\mathbf{z}$. Specifically, we define the expected cumulative reward at each time step using value function by $Q(\hat{z}_t|\hat{\mathbf{z}}_{1:t-1}, f(\mathbf{u})) = E_{p_\theta(\hat{\mathbf{z}}_{t+1:T}|\hat{\mathbf{z}}_{1:t}, f(\mathbf{u}))}R_\mathbf{z}(\hat{\mathbf{z}})$. The value function $Q(z_t|\mathbf{z}_{1:t-1}, f(\mathbf{u}))$ is then estimated by aggregating the Monte-Carlo simulation at each time step, given by

$$\begin{aligned}&Q(\hat{z}_t|\hat{\mathbf{z}}_{1:t-1}, f(\mathbf{u})) \approx \\ &\begin{cases} \frac{1}{N}\sum_{n=1}^N R_\mathbf{z}([\hat{\mathbf{z}}_{1:t-1}, z_t, \hat{\mathbf{z}}^n_{t+1:l_z}]), & t < l \\ R_\mathbf{z}([\hat{\mathbf{z}}_{1:t-1}, \hat{z}_t]). & t = l_z \end{cases}\end{aligned} \tag{8}$$

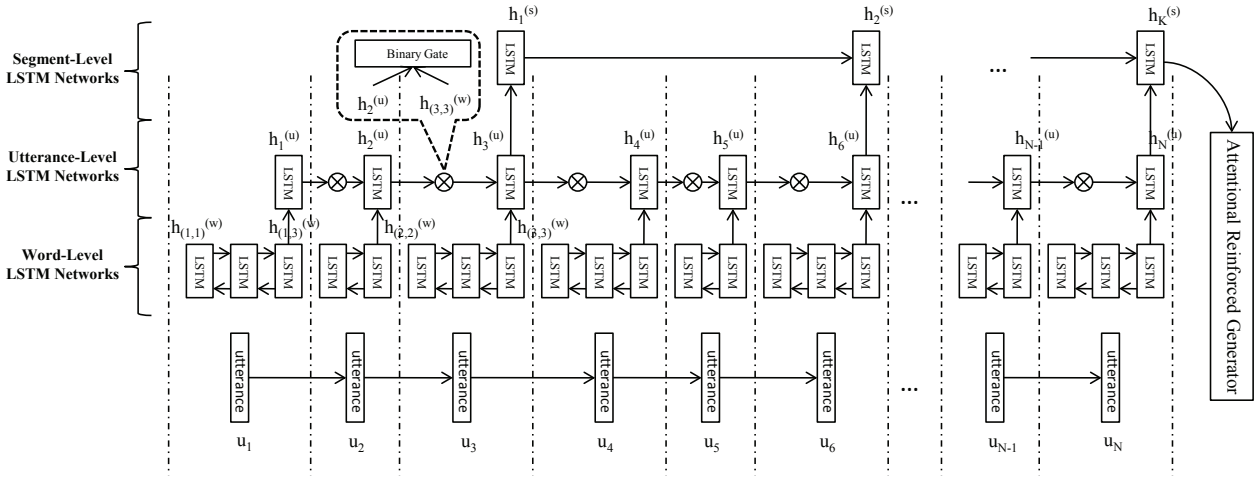


Figure 1: The Framework of Abstractive Meeting Summarization via Hierarchical Adaptive Segmental Encoder-Decoder Networks. The hierarchical encoder network learns the semantic representation of meeting conversation contents with adaptive conversation segmentation. The reinforced recurrent decoder network then generates the summaries for abstractive meeting summarization.

We denote that $\{\hat{\mathbf{z}}^1_{t+1,l_z}, \ldots, \hat{\mathbf{z}}^N_{t+1:l_z}\} \sim p_\theta(\hat{\mathbf{z}}_{t+1:l}|\hat{\mathbf{z}}_{1:t}, f(\mathbf{u}))$ as the set of simulated generated summaries, which are randomly sampled starting from the $t+1$-th time step on the current state and action. According to the policy gradient theorem, the gradients of the reinforced decoder network can be estimated by

$$\nabla_\theta \mathcal{L}_{RL}(g(f(\mathbf{u}))) = \sum_{t=1}^{l} \nabla_\theta \log p_\theta(\hat{z}_t|\hat{\mathbf{z}}_{1:t-1}, f(\mathbf{u})) Q(\hat{z}_t|\hat{\mathbf{z}}_{1:t-1}, f(\mathbf{u})). \quad (9)$$

3 EXPERIMENTS

In this section, we conduct several experiments on the AMI meeting corpus [6], to show the effectiveness of our approach for the problem of abstractive meeting summarization.

3.1 Experimental Setting

We first present the details of AMI meeting corpus dataset. The AMI meeting corpus is a collection of 142 meeting records along with their corresponding human-authored abstractive summaries. In each meeting, the group of people are engaged in a team and each speak assumes a certain role in a team. Following the experimental setting in [16, 21], we use 100 dialogs as training set, 20 dialogs as validation set and the remaining 22 dialogs as the testing set. So the training, validation and testing data do not have overlap. For evaluation, we employ ROUGE [9], which compares machine summaries with human gold-standard summaries using n-gram overlap and skip-gram overlap. We train the proposed method on machines with Linux OS, Intel(R) Core i7-5930K 3.5GHz and two GTX TITIAN X graphic cards. The initial learning rate of our method is set to 1, the decay rate is set to 0.5 and the drop rate is set to 0.5. The vocabulary size of our method is set to 10,517.

3.2 Experimental Comparisons

We evaluate the performance of our proposed method based on three widely-used evaluation for the problem of abstractive meeting summarization [10, 16], i.e., ROUGE-1, ROUGE-2 and ROUGE-SU. We compare our proposed method with other five state-of-the-art methods for the problem of abstractive meeting summarization as follows:

- **TextRank** method [11] is the extractive meeting summarization model which selects the important utterances from the meeting conversations for summarization.
- **Fusion** method [10] is the abstractive meeting summarization model which aggregates the most relevant sentences in the word graph and then selects the best paths as abstractive summaries.
- **TG** method [16] extends multi-sentence fusion algorithm to select the best templates for generating abstractive summarizes, which is based on the relationship between summaries and their source meeting transcripts.
- **AED-RNN** method [14] is under the encoder-decoder framework that generates the abstractive summaries for meeting conversation with attentional recurrent neural networks.
- **PGN** method [20] is the pointer-generator network, which is current state-of-art model in general abstractive text summarization task.
- **SEASS** method [26] is the selective encoding model based on sequence-to-sequence framework for abstractive summarization.
- **HAS-ML** method is the adaptive segmental network model that jointly segments the meeting conversation and generates the abstractive summaries, which is trained by maximum likelihood estimation.
- **HAS-RL** method is the adaptive segmental network model, which is trained by reinforcement learning.

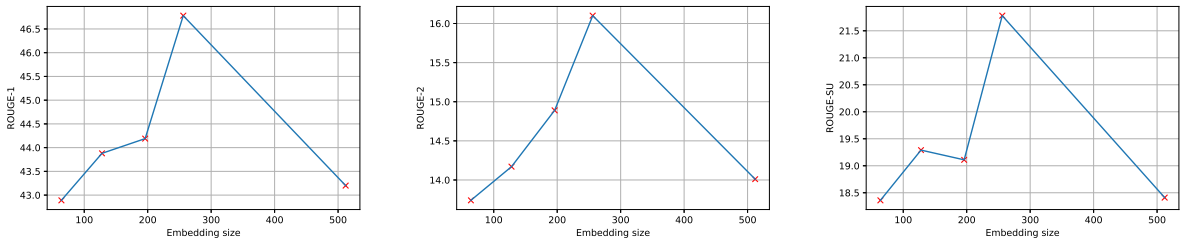


Figure 2: Effect of the dimension of word embedding on ROUGE-1, ROUGE-2 and ROUGE-SU

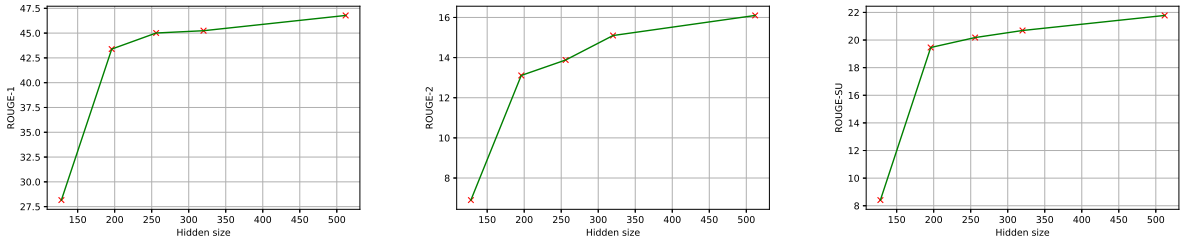


Figure 3: Effect of the dimension of LSTM hidden state on ROUGE-1, ROUGE-2 and ROUGE-SU

Table 1: Experimental results on ROUGE-1, ROUGE-2 and ROUGE-SU.

Method	ROUGE-1	ROUGE-2	ROUGE-SU
TextRank	25.12	3.37	6.61
Fusion	32.30	4.80	8.10
TG	31.50	6.70	11.40
AED-RNN	39.74	14.29	15.16
PGN	41.52	14.89	17.01
SEASS	43.93	14.29	19.10
HAS-ML	46.78	16.10	21.78
HAS-RL	**48.64**	**17.45**	**22.13**

Among them, methods TextRank, Fusion, TG, HAS-ML and HAS-RL are proposed for meeting summarization while methods AED-RNN and SEASS are designed for document summarization. The methods Fusion, TF generate the abstractive summaries for meeting conversation based on the constructed word graph. The methods ADE-RNN, PGN, and SEASS are under the encoder-decoder framework, which generates the abstractive summaries with attentional recurrent neural networks. Unlike the previous abstractive summarization methods, our HAS model jointly segments the meeting conversation and generates the abstractive summaries for meeting conversation. To exploit the effect of different training process, we denote our HAS model based on maximum likelihood by HAS-ML, and the one based on reinforcement learning by HAS-RL. We choose the ROUGE-SU evaluation criteria as the reward in the reinforcement learning for HAS-RL. The input words of our methods are initialized by pre-trained word embeddings and the weights of LSTMs are randomly by a Gaussian distribution with zero mean.

Table 1 shows the overall experimental results of the methods for abstractive meeting summarization on ROUGE-1, ROUGE-2 and ROUGE-SU, respectively. The hyperparameters and parameters are chosen to conduct the testing evaluation. We report the average value of all the methods on three evaluation criteria. The experimental results reveal a number of interesting points:

- The methods based on abstractive summarization, Fusion, TG, AED-RNN, SEASS, HAS-ML and HAS-RL outperform the extractive-based method TextRank, which suggests that the generation-based approach is critical for the problem.
- The methods based on recurrent neural networks, AED-RNN and SEASS achieve better performance than other baselines. This suggests that the deep representation of meeting conversation also improves the performance of abstractive meeting summarization.
- In all the cases, both of our HAS-ML and HAS-RL methods achieve the best performance. This fact shows that the adaptive segmental network learning framework that joint exploit the conversation segmentation and abstractive summarization generation can further improves the performance of abstractive meeting summarization.
- The HAS model trained by reinforcement learning, HAS-RL, outperforms the one trained by maximum likelihood, HAS-ML, which suggests that reinforcement learning is effective for generating the summaries in the problem of abstractive meeting summarization.

In our approach, there is two essential parameters, which are the dimension of hidden LSTM state and word embedding size. We first study the effect of word embedding on our method by varying the dimension of word embedding from 64 to 512 on ROUGE-1, ROUGE-2 and ROUGE-SU in Figures 2(a), 2(b) and 2(c). We can see that our method achieves the best performance when the size of word

Human-Authored Summary
the project manager acquainted the team with the tools and equipment around them and then had the team members introduce themselves by name and what role they had in the project. the project manager then introduced the upcoming project along with more tools and equipment to the team members. the team members then participated in an exercise in which they drew their favorite animals. after the drawing exercise, the project manager talked about the project finances and production costs. the team then discussed their experiences with remotes and various features to consider when producing a remote
Summary Generated by HAS-ML
the project manager opened the meeting and introduced the project, to design a remote control. the team members then participated in an exercise in which they each drew their favorite animal and discussed why they liked the animal. the project manager then led the team in calculating the production costs of the remote and what features they would like to include in the remote they are to create. the team discussed the interior workings of a remote and how to make the remote more technologically innovative but decided to use a spongy material in their design
Summary Generated by HAS-RL
the project manager opened the meeting and introduced the project, to design a remote control. the project manager then introduced the upcoming project to the team members and then the team members participated in an exercise in which they each drew their favorite animal and discussed what they liked about the animal. the project manager then led the team in calculating the production costs of the remote and what features they would like to include in the remote they are to create. the project manager briefed the team on some new requirements and led them in a discussion on their experiences with remotes and various features to consider in making the remote

Figure 4: A comparison between a human-authored summary and the summaries created by methods HAS-ML and HAS-RL.

embedding is set to 256. We then investigate the effect of hidden LSTM state on our method by varying the dimension of hidden LSTM state from 128 to 512 on ROUGE-1, ROUGE-2 and ROUGE-SU in Figures 3(a), 3(b) and 3(c), respectively. We can observe that the performance trend of our method becomes convergent after the dimension of hidden LSTM state greater than 256.

Finally, we demonstrate the effectiveness of the summaries generated by our HAS model with a human-authored one in Figure 4. We can observe that, compared with the human-authored summary, both the summaries generated by HAS-ML and HAS-RL methods also cover the main points in the meeting conversation well. Compared with the summary generated HAS-ML method, the summary generated by HAS-RL method are more fluent and natural.

4 RELATED WORK

In this section, we briefly review some related work on meeting summarization and abstract document summarization.

The existing work for the problem of meeting summarization can be mainly categorized as extractive-based approaches [5, 18, 25] and abstractive-based approaches [2, 10, 12, 16, 21, 24]. Xie et al. [25] treat extractive meeting summarization task as a binary classification problem. Bui et al. [5] investigate the directed graphical models for extractive meeting summarization. Riedhammer et al. [18] propose the unsupervised method for extractive meeting summarization. A recent study [13] revealed that people generally prefer abstractive summaries to the extracted ones for meeting summarization. Mehdad et al. [10] build the entailment graph for meeting summarization. Wang et al. [24] apply multiple-sequence alignment to generate templates for abstractive meeting summarization. Oya et al. [16] leverage the relationship between human-authored summaries and their source meeting transcriptions to select the templates for generating abstractive summaries for meetings. Murray et al. [12] formulate the abstractive meeting summarization as a Markov Decision Process and value iteration is used for natural language generation. Banerjee et al. [2] generate abstractive summaries by fusing important content from several utterances with dependency graph. Singla et al. [21] develop the abstractive summarization systems with automatic communities for spoken conversation summarization. Unlike previous studies, we formulate the problem of meeting summarization from the viewpoint of adaptive segmental encoding, which can be solved by recurrent neural networks with policy gradient learning.

Abstractive document summarization is the process of automatically generating natural language summaries from an input document while retaining the important points [3]. Banerjee et al. [3] tackle the abstractive summarization with integer linear programming based on word graph. Rush et al. [19] develop the neural attention-based summarization model based on neural machine translation. Tan et al. [22] propose the graph-based attention mechanism for abstractive document summarization. See et al. [20] present the hybrid pointer-generator architecture with coverage for text summarization. Paulus et al. [17] propose the neural summarization method that combines supervised word prediction and reinforcement learning. Nema et al. [15] introduce the query-based abstractive summarization task. Zhou et al. [26] develop the selective encoding for abstractive sentence summarization. However, the problem formulation of abstractive meeting summarization is different from the problem of document summarization, because of the unstructured long meeting conversation contents. Therefore, the existing abstractive document summarization methods may not be directly applied to our problem.

5 CONCLUSION

In this paper, we present the problem of abstractive meeting summarization from the viewpoint of hierarchical adaptive segmental encoder-decoder network learning framework. We first propose the adaptive segmental encoder networks that learn the semantic representation of meeting conversation contents with adaptive conversation segmentation. We then devise the reinforced decoder networks to generate the natural language summaries for abstractive meeting summarization. We evaluate the effectiveness of our method through extensive experiments on the well-known AMI meeting conversation dataset.

ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of China under Grant No.61602405 and No.61836002, Alibaba Innovative Research, China Knowledge Centre for Engineering Sciences and Technology , Joint Research Program of ZJU and Hikvision Research Institute, and was partially funded by Microsoft Research Asia.

REFERENCES

[1] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086* (2016).

[2] Siddhartha Banerjee, Prasenjit Mitra, and Kazunari Sugiyama. 2015. Abstractive meeting summarization using dependency graph fusion. In *WWW*. ACM, 5–6.

[3] Siddhartha Banerjee, Prasenjit Mitra, and Kazunari Sugiyama. 2015. Multi-Document Abstractive Summarization Using ILP Based Multi-Sentence Compression.. In *IJCAI*. 1208–1214.

[4] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432* (2013).

[5] Trung H Bui, Matthew Frampton, John Dowding, and Stanley Peters. 2009. Extracting decisions from multi-party dialogue using directed graphical models and semantic similarity. In *SIGDIAL*. ACL, 235–243.

[6] Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, et al. 2005. The AMI meeting corpus: A pre-announcement. In *International Workshop on Machine Learning for Multimodal Interaction*. Springer, 28–39.

[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[8] Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. *ACL* (2015).

[9] Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. ACL, 71–78.

[10] Yashar Mehdad, Giuseppe Carenini, Frank Wm Tompa, and Raymond T Ng. 2013. Abstractive Meeting Summarization with Entailment and Fusion.. In *ENLG*. 136–146.

[11] Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

[12] Gabriel Murray. 2015. Abstractive meeting summarization as a Markov decision process. In *Canadian Conference on Artificial Intelligence*. Springer, 212–219.

[13] Gabriel Murray, Giuseppe Carenini, and Raymond Ng. 2010. Generating and validating abstracts of meeting conversations: a user study. In *Proceedings of the 6th International Natural Language Generation Conference*. ACL, 105–113.

[14] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Ça glar Gulçehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. *CoNLL 2016* (2016), 280.

[15] Preksha Nema, Mitesh Khapra, Anirban Laha, and Balaraman Ravindran. 2017. Diversity driven Attention Model for Query-based Abstractive Summarization. *ACL* (2017).

[16] Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. 2014. A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *INLG*. 45–53.

[17] Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A Deep Reinforced Model for Abstractive Summarization. *arXiv preprint arXiv:1705.04304* (2017).

[18] Korbinian Riedhammer, Benoit Favre, and Dilek Hakkani-Tür. 2010. Long story short–global unsupervised models for keyphrase based meeting summarization. *Speech Communication* 52, 10 (2010), 801–815.

[19] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685* (2015).

[20] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. *ACL* (2017).

[21] Karan Singla, Evgeny Stepanov, Ali Orkan Bayer, Giuseppe Carenini, and Giuseppe Riccardi. 2017. Automatic Community Creation for Abstractive Spoken Conversations Summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*. 43–47.

[22] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive Document Summarization with a Graph-Based Attentional Neural Model. In *ACL*, Vol. 1. 1171–1181.

[23] Kees van Deemter, Emiel Krahmer, and Mariët Theune. 2005. Real vs. template-based natural language generation: a false opposition. *Computational Linguistics* 31, 1 (2005), 15–24.

[24] Lu Wang and Claire Cardie. 2013. Domain-Independent Abstract Generation for Focused Meeting Summarization.. In *ACL*. 1395–1405.

[25] Shasha Xie, Yang Liu, and Hui Lin. 2008. Evaluating the effectiveness of features and sampling in extractive meeting summarization. In *Spoken Language Technology Workshop, 2008. SLT 2008. IEEE*. IEEE, 157–160.

[26] Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2017. Selective Encoding for Abstractive Sentence Summarization. *ACL* (2017).

Adversarial Point-of-Interest Recommendation

Fan Zhou*
University of Electronic Science and Technology of China
fan.zhou@uestc.edu.cn

Ruiyang Yin
University of Electronic Science and Technology of China
yinruiyang94@gmail.com

Kunpeng Zhang
University of Maryland, College park
kzhang@rhsmith.umd.edu

Goce Trajcevski
Iowa State University, Ames
gocet25@iastate.edu

Ting Zhong
University of Electronic Science and Technology of China
zhongting@uestc.edu.cn

Jin Wu
University of Electronic Science and Technology of China
wj@uestc.edu.cn

ABSTRACT

Point-of-interest (POI) recommendation is essential to a variety of services for both users and business. An extensive number of models have been developed to improve the recommendation performance by exploiting various characteristics and relations among POIs (e.g., spatio-temporal, social, etc.). However, very few studies closely look into the underlying mechanism accounting for why users prefer certain POIs to others. In this work, we initiate the first attempt to learn the distribution of user latent preference by proposing an *Adversarial POI Recommendation (APOIR)* model, consisting of two major components: (1) the *recommender* (*R*) which suggests POIs based on the learned distribution by maximizing the probabilities that these POIs are predicted as unvisited and potentially interested; and (2) the *discriminator* (*D*) which distinguishes the recommended POIs from the true check-ins and provides gradients as the guidance to improve *R* in a rewarding framework. Two components are co-trained by playing a minimax game towards improving itself while pushing the other to the boundary. By further integrating geographical and social relations among POIs into the reward function as well as optimizing *R* in a reinforcement learning manner, APOIR obtains significant performance improvement in four standard metrics compared to the state of the art methods.

KEYWORDS

POI recommendation, adversarial learning, policy gradient

ACM Reference Format:
Fan Zhou, Ruiyang Yin, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Jin Wu. 2019. Adversarial Point-of-Interest Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313609

1 INTRODUCTION AND MOTIVATION

Location-based recommendation systems primarily aim at suggesting spatial entities (i.e., point-of-interest (POI)) to users, and have recently spurred a significant research interests in both academia and industry. Different variations of POI recommendations have been explored, including next POI recommendation [4, 7], time-aware POI recommendation [13] and out-of-town POI recommendation [21].

A body of existing works has focused on improving the POI recommendation performance by exploiting various implicit context features embedded in check-ins, such as their spatial information [13, 14, 17]; semantics [9]; social relations [5, 11, 36]; temporal characteristics [18, 33]; and sequential dependence [6, 19, 27, 35]. Complementary to these, approaches that model user preference through mining various features for POI recommendation have been proposed – e.g., Matrix Factorization (MF) [11, 13, 14, 17], Context Embedding (CE) [15, 27, 33] and Pairwise Ranking (PR) [4, 12, 22, 33]. A recent work [16] summarized that geographical information and social influence are the two most effective factors for modeling user preference, while MF based methods such as GeoMF and RankGeoFM exhibit superior performance on POI recommendation. Some of the existing methods make simple but proven-to-be effective assumptions for POI recommendation.

At the heart of our motivation is the observation that many of the existing studies lack formal underlying mechanisms to understand essential user check-in behavior, which may often lead to elusive results. Inspired by recent advances in deep generative models [8, 25] that are successfully and widely used in the areas of computer vision and information retrieval, we attempt to learn user latent preference in a generative way, rather than extracting different features and quantitatively analyzing their impact on POI recommendation as done in most of the existing works.

Towards that, we propose a novel POI recommendation approach, called *Adversarial POI Recommendation (APOIR)*, which learns the underlying check-in distribution in an adversarial manner by simultaneously training two synergistic components. Specifically, we model a user u and his check-in locations l in a generative way, with two neural network components: *recommender* (*R*) and *discriminator* (*D*) – being co-trained alternatively, to optimize the generative process of $u \rightarrow l$. *R* recommends POIs based on the currently learned user preference distribution, while *D*, acting as a catalyzer, judges whether the recommended POIs are true locations visited by that user and provides guidance to improve *R*. Two effective features (geographical and social influence) are also incorporated into the APOIR to further improve the performance. Note that our model can be generalized to include other features.

*Corresponding author: Fan Zhou (fan.zhou@uestc.edu.cn)

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313609

To train APOIR, we alternate the updates of the recommender and the discriminator as follows. Given the currently learned preference distribution, a set of POIs are sampled to update the R using the policy gradient method. Subsequently, D is updated using user u's true check-ins along with the ones sampled from the updated R. After an equilibrium is reached – which is, the true user preference distribution is close enough to the empirical learned distribution, a list of POIs with high probabilities based on the learned distribution are generated and eventually recommended. Our main contributions can be summarized as follows:

- We address the POI recommendation in a generative way, which is a novel approach in the area of spatial mining and POI recommendation.
- We propose a method – APOIR – to learn underlying user preference distribution, which significantly boosts the recommendation performance. In addition, APOIR successfully unifies reinforcement learning and matrix factorization methods into an adversarial learning framework for POI recommendation.
- We evaluate our method on three public location-based social network (LBSN) datasets and compare it to several state-of-the-art models. The results show that the APOIR approach performs well, e.g., achieving 10.17%, 11.7% and 10.0% improvement over the best baseline on Gowalla, Foursquare and Yelp datasets in terms of Precision@5.

2 RELATED LITERATURE

ML based POI Recommendation Matrix Factorization (MF) and its variants are prevalent techniques in traditional POI recommendation. Typically, a user-POI matrix is factorized to identify user latent references. For example, in [14] user and POI latent factors in the factorization model are augmented with activity area vectors of users and influence vectors of POIs, to deal with the challenge of matrix sparsity. User interest and check-in behavior [12], user preference ranking and metric embedding [4], as well as POI semantic categories [9] are also taken into account for POI recommendation in the MF-based framework.

An extensive experimental evaluation [16] has been recently conducted to compare 12 representative POI recommendation models and draw relevant findings, among which are: (a) geographical information [13, 14]; and (b) implicit feedback of user preference [17], are the most relevant factors to the recommendation performance. According to their results, RankGeoFM [13], IRenMF [17] and GeoMF [14] are the three state-of-the-art POI recommendation models. Our work differs from these methods in that we propose a generative model to learn underlying user preference distribution in an adversarial manner.

GAN and POI Recommendation Generative adversarial networks (GANs) have gained recent popularity due to their successes in natural image generation [8]. GANs have also been used for modeling sequential data, such as discrete token generation [29] and have been applied in information retrieval [25]. However, few efforts have been made towards recommending POIs in LBSNs with adversarial networks due to the following challenges: (1) data sparsity inherent in LBSN data, e.g., the density of the check-ins used in POI recommendation is usually around 0.1% [16]; (2) complexity. Various complicated, latent, and mixed-typed factors, including geographical, spatio-temporal, sequential, social influence, etc., need to be combined into a unified framework for addressing POI recommendation. Our proposed method APOIR provides a fundamentally different way for POI recommendation by learning the user preference distribution, and thus opens up a new perspective for addressing POI recommendation problems – e.g., adversarial time-aware POI recommendation and next POI recommendation.

3 PROBLEM FORMULATION AND PRELIMINARIES

Given a set of POIs $\mathcal{L}$ ($|\mathcal{L}| = M$) and a set of users $\mathcal{U}$ ($|\mathcal{U}| = N$), each with associations to multiple historical check-ins $\mathcal{L}^{u_i}$, POI recommendation aims at recommending each user $u_i \in \mathcal{U}$ with *top-K new* POIs in the set of $\widetilde{\mathcal{L}}^{u_i} = \mathcal{L} - \mathcal{L}^{u_i}$ that u_i is likely to be interested in but has never visited before.

Matrix Factorization (MF): [10] decomposes the user check-in matrix $\mathbf{C} \in \mathbb{R}^{N \times M}$ into a user matrix $\mathbf{U} \in \mathbb{R}^{N \times Q}$ and a POI matrix $\mathbf{L} \in \mathbb{R}^{M \times Q}$ with Q-dimensional latent factors by:

$$\arg\min_{\mathbf{U},\mathbf{L}} \|\mathbf{C} - \mathbf{U}\mathbf{L}^\intercal\|_F^2 + \kappa^u \|\mathbf{U}\|_F^2 + \kappa^l \|\mathbf{L}\|_F^2 \tag{1}$$

where κ^u and κ^l are regularization coefficients, N is the number of users and M is the number of check-ins. The probability of recommending a POI l_j to user u_i is thus derived based on the inner product between the latent factor of user $\mathbf{u}_i$ and that of POI $\mathbf{l}_j$ (denoted as $\mathbf{u}_i\mathbf{l}_j^\intercal$). The preference score vector of u_i over all POIs is denoted $\mathbf{u}_i\mathbf{L}^\intercal$.

Gated Recurrent Units (GRU): [3] is a variant of recurrent neural network (RNN) models consisting of gating mechanisms that control the influence of the hidden state of previous unit h_{t-1} on the state h_t at time step t, i.e., learning to ignore the previous units if necessary. Specifically,

$$\begin{aligned} g_t &= \sigma(W\mathbf{l}_j + Uh_{t-1}) \\ s_t &= \sigma(W\mathbf{l}_j + Uh_{t-1}) \\ \tilde{h}_t &= \tanh(W\mathbf{l}_j + U(s_t \odot h_{t-1})) \\ h_t &= (1 - g_t)h_{t-1} + g_t\tilde{h}_t \end{aligned} \tag{2}$$

where g_t and s_t are update and reset gates, respectively; $\tilde{h}_t$ is a candidate hidden state; W and U are parameter matrices of respective units; σ and tanh are sigmoid and hyperbolic tangent functions, respectively; $\odot$ is element-wise product.

Generative Adversarial Nets (GAN): [8] aims at obtaining the Nash equilibrium between a discriminator D and a generator G by optimizing the following minimax objective :

$$\mathcal{J}_{GAN} = \mathop{\mathbb{E}}_{x \sim p_x}[\log(D(x))] + \mathop{\mathbb{E}}_{\tilde{x} \sim p_g}[\log(1 - D(\tilde{x}))] \tag{3}$$

where p_x is the data distribution and p_g is the model distribution implicitly defined by $\tilde{x} = G(z)$, and $\mathcal{J}_{GAN}$ is maximized w.r.t $D(x)$ and minimized w.r.t. $D(G(\tilde{x}))$. The generator G takes a noise prior distribution $z \sim p(z)$ (e.g., uniform or Gaussian) as input and upon which a sample is generated using a deep neural network. The discriminator D_ϕ – usually another neural network – plays the role of classifier and distinguishes that a certain sample coming

from the true distribution p_x or the generator G. It has been demonstrated that this game achieves global equilibrium if and only if $p_g(x) = p_x(x)$, where p_g is the defined distribution and the optimal discriminator is $D^*(x) = p_x(x)/(p_x(x) + p_g(x))$ [8].

4 OUR PROPOSED APPROACH: APOIR

We now discuss in detail our proposed method, and feature modeling and training details.

Temporal & sequential preference modeling. Before presenting the APOIR method, we first leverage a variant of GRU, combined with MF, to capture both temporal and sequential preference of users. Given a sequence of POIs $l_1, l_2, \cdots, l_t$, each associated with a check-in time $\tau_1, \tau_2, \cdots, \tau_t$, we compute the time interval between adjacent POIs as $\Delta_{\tau_t} = \tau_i - \tau_{i-1}$, $i \in [1, t]$. Then, we can modify the candidate hidden state of Eq.(2) with a time gate T_t as:

$$\begin{aligned} T_t &= \sigma_t(W\mathbf{l}_j + \sigma([\Delta_{\tau_t}; \tau_t]W_t)) \\ \tilde{h}_t &= \tanh(W\mathbf{l}_j + U(s_t \odot T_t \odot h_{t-1})) \end{aligned} \tag{4}$$

where $[\Delta_{\tau_t}; \tau_t]$ is a concatenation of time interval between two successive check-ins and the current check-in time. Now time gate T_t captures the temporal preference of users, as well as POI representation $\mathbf{l}_j$, and is used to control the influence of previous hidden state h_{t-1} in Eq.(4). Finally, a user's temporal and sequential preferences are coded in the last hidden state h_t, which is then used to update user representation with an element-wise product as $\hat{\mathbf{u}}_i = \mathbf{u}_i \odot h_t$. The new user latent factor representation $\hat{\mathbf{u}}_i$ would be used in the following adversarial learning.

Note that above temporal GRU is similar to recent advances on recurrent unit modification [18, 19, 37] towards capture contextual information associated with input data in RNN, except that we only consider temporal features here because we would learn other important factors in an adversarial manner later.

4.1 Adversarial Learning

In this work, we use a generative process $u_i \rightarrow \mathcal{L}$ to model the relationship between a user u_i and the set of POIs $\mathcal{L}$. The underlying true distribution of the user preferences over POIs (expressed as a conditional probability $p_{true}(\mathcal{L}|u_i)$) will be learned by alternatively optimizing two competing components: *Recommender* and *Discriminator*.

4.1.1 Recommender. $R_\theta(l^R|u_i)$, parameterized by θ, is used to recommend a set of unvisited but potentially interested POIs l^R for u_i. The recommender R_θ here is a generator analogue in GAN where it mainly fits a true distribution of data $p_{true}(\mathcal{L}|u_i)$. Similar to this, the POIs recommended by R_θ are based on a process of sampling from the learned empirical distribution $R_\theta(\mathcal{L}|u_i)$. Such a user-POI preference distribution is approximated using the pairwise bayesian personalized ranking (BPR) [22], where user representation is obtained via above described temporal GRU. The objective is to learn a R_θ where the true distribution is close enough to the empirical one so that it is difficult for the discriminator to decide whether the POI is generated by R_θ or from the true distribution.

4.1.2 Discriminator. $D_\phi(u_i, l^R)$, parameterized by ϕ, is used to discriminate whether the recommended POI l^R matches the true preference $\mathcal{L}^{u_i}$ of user u_i. In other words, it constructs a binary vector where 1 denotes that the recommended POI is exactly the true check-ins (positive example) while 0 means a mismatch (non-visited example). The goal of $D_\phi(u_i, l^R)$ is to improve its ability to distinguish historical check-ins from the recommended ones upon R_θ, for a given user u_i.

4.1.3 Objective. According to the GAN paradigm [8], the above two models can be unified into a minimax game: $R_\theta(l^R|u_i)$ maximizes the probability of $D_\phi(u_i, l^R)$ not being able to discriminate recommended POIs l^R from the truth. Correspondingly, $D_\phi(u_i, l^R)$ judges and improves the recommendation performance of $R_\theta(\mathcal{L}|u_i)$. Eventually, $R_\theta(l^R|u_i)$ recommends POIs for user u_i with high quality once an equilibrium reaches, in which $D_\phi(u_i, l^R)$ cannot distinguish the recommended POIs from the truth. Formally, we have the following objective trained alternatively between the recommender R_θ and the discriminator D_ϕ:

$$\begin{aligned} \mathcal{J}^{R^*, D^*} = \min_\theta \max_\phi \sum_{u_i \in \mathcal{U}} \Big(& \mathbb{E}_{l^+ \sim \mathcal{L}^{u_i}}[\log D_\phi(u_i, l^+)] \\ & + \mathbb{E}_{l^R \sim R_\theta(l^R|u_i)}[\log(1 - D_\phi(u_i, l^R))] \Big) \end{aligned} \tag{5}$$

where l^+ is the set of visited (positive) check-ins, and discriminator $D_\phi(u_i, l^R)$ estimates the probability of POIs l^R being preferred/visited by user u_i. We use the sigmoid function $\sigma(D_\phi(u_i, l^R))$ as the discrimination score, similar to [25].

As shown in Eq.(5), both the discriminator and the recommender are iteratively optimized in a minimax game, and we now describe them in details.

4.1.4 Training Discriminator. The objective of the discriminator is to maximize the probability of correctly distinguishing the true check-in locations from the generated recommended POIs by the recommender, given positive samples from true preference distribution and non-visited samples from the recommender. That is, the training objective of D_ϕ is to find an optimal ϕ^* by maximizing:

$$\begin{aligned} \phi^* = \arg\max_\phi \sum_{u_i \in \mathcal{U}} \Big(& \mathbb{E}_{l^+ \sim \mathcal{L}^{u_i}}[\log D_\phi(u_i, l^+)] \\ & + \mathbb{E}_{l^R \sim R_{\theta^*}(l^R|u_i)}[\log(1 - D_\phi(u_i, l^R))] \Big) \end{aligned} \tag{6}$$

where $l^R \sim R_{\theta^*}(l^R|u_i)$ is the generated POIs by the current optimal R_θ , and $l^+ \sim \mathcal{L}^{u_i}$ are the positive samples. $D_\phi(u_i, l^R)$ can also be considered as the probability of R_θ assigning correct labels to recommended POIs l^R. Since function $D_\phi(\cdot)$ is differentiable w.r.t parameters ϕ, the above objective can be solved by stochastic gradient descent [25].

4.1.5 Training Recommender. Similarly to SeqGAN [29], the recommender R_θ generates (selects) a list of ranked POIs for user u_i. Specifically, given the current D_ϕ which is fixed after Eq.(6), we can minimize the following objective to find an optimal θ^*:

$$\begin{aligned} \theta^* &= \arg\min_\theta \sum_{u_i \in \mathcal{U}} \mathbb{E}_{l^R \sim R_\theta(l^R|u_i)} \log(1 - \sigma(D_\phi(u_i, l^R))) \\ &= \arg\max_\theta \sum_{u_i \in \mathcal{U}} \mathbb{E}_{l^R \sim R_\theta(l^R|u_i)} \log(1 + \exp(D_\phi(u_i, l^R))) \end{aligned} \tag{7}$$

which is the objective considered in previous works [25, 34]. However, it does not consider the contexts associated with items, or POIs

in our case. To explicitly explore the rewards from POI contexts, we modify above objective as:

$$\theta^* = \arg\max_\theta \sum_{u_i \in \mathcal{U}} \mathbb{E}_{l^R \sim R_\theta(l^R|u_i)} \log(\lambda + \exp(D_\phi(u_i, l^R)))$$
$$= \arg\max_\theta \sum_{u_i \in \mathcal{U}} \underbrace{\mathbb{E}_{l^R \sim R_\theta(l^R|u_i)} \log \mathcal{R}}_{\mathcal{J}^R(u_i)}$$

where we add a factor λ which is a constant for each u_i, and a defined reward $\mathcal{R} = \lambda + \exp(D_\phi(u_i, l^R))$ will be used as the reward (will explain it later).

Since the POIs l^R sampled from the recommender R_θ is discrete, $\mathcal{J}^R(u_i)$ cannot be directly optimized with gradient descent as in continuous GANs [8]. Following the discrete GANs [25, 29, 34] where the gradient descent is not available, we use the probability of being sampled for each POI to replace the discrete POI as:

$$\nabla_\theta \mathcal{J}^R(u_i) = \nabla_\theta \mathbb{E}_{l^R \sim R_\theta(l^R|u_i)} \log \mathcal{R} = \sum_{j=1}^{M} \nabla_\theta R_\theta(l^j|u_i) \log \mathcal{R}$$
$$= \sum_{j=1}^{M} R_\theta(l^j|u_i) \nabla_\theta \log \mathcal{R} = \mathbb{E}_{l^R \sim R_\theta(l^R|u_i)}[\nabla_\theta \log \mathcal{R}]$$
$$\backsimeq \frac{1}{K} \sum_{k=1}^{K} \nabla_\theta \log R_\theta(l^k|u_i) \log \mathcal{R} \quad (8)$$

where K is the number of POIs sampled by the recommender and l^k is the k^{th} sampled POI. Eq.(8) shows that we use policy gradient based REINFORCE algorithm [26] to derive the gradient. In the context of reinforcement learning [23], $\log(\lambda + \exp(D_\phi(u_i, l^k)))$ acts as the *reward* for the *policy* $R_\theta(l^k|u_i)$ when taking *action* of recommending POI l^k in the *environment* u_i.

4.2 Modeling Reward

We note that above reinforcement learning based likelihood sampling is also used in previous work [25, 29, 34] for sequential data generation and information retrieval. However, these work did not explicitly model the reward of Eq.(8). Instead, they can be considered as optimizations for various methods, e.g., IRGAN [25] is essentially an adversarial optimization method for MF in item recommendation.

In this section, we proceed with modeling the context of POIs and the implicit feedback into the reward function. In this work, we exploit two most important factors [16], i.e., geographical and social influence, for explicitly measuring reward of candidate POIs. Specifically, we consider following reward λ in Eq.(8):

$$\lambda = \alpha \mathcal{R}_{geo} + (1 - \alpha)\mathcal{R}_{soc} \quad (9)$$

where $\mathcal{R}_{geo}$ and $\mathcal{R}_{soc}$ are the reward from the geographical and social factors, respectively. Hyperparameter α is used for scaling the two factors. $\mathcal{R}_{geo}$ and $\mathcal{R}_{soc}$ are in fact two 1-D vectors ($\mathcal{R}_{geo}, \mathcal{R}_{soc} \in \mathbb{R}^{1\times M}$) constructed as follows:

– **Geographical reward** $\mathcal{R}_{geo}$ is initialized with 0 in each column. We set the j^{th} item $l^j \in \mathcal{R}_{geo}$ as 1 if POI l^j is within a distance d to *any* visited POIs for user u_i. That is, we are interested in including the nearby POIs $\mathcal{N}(l^j)$ for all check-ins of user u_i into the candidate list and magnify its importance since people are normally visiting the neighboring POIs [14, 16, 17, 31].

– **Social reward** $\mathcal{R}_{soc}$ is generated in a similar way by setting the j^{th} column $u^j \in \mathcal{R}_{soc}$ to 1 if corresponding POI has been visited by u_i's friend $u_j \in \mathcal{F}(u_i)$, where $\mathcal{F}(u_i)$ denotes the friends of u_i – motivated by the observation that people may visit the POIs where their friends have visited before [2, 11, 16, 31].

4.2.1 Overall Rewards. Note that the second term $\exp(D_\phi(u_i, l^k))$ in the reward of Eq.(8), the function of the discriminator, is also a reward item ($1 \times M$ vector) . Now, we have the following overall reward:

$$\mathcal{R} = \beta(\alpha \mathcal{R}_{geo} + (1 - \alpha)\mathcal{R}_{soc}) + (1 - \beta)\mathcal{R}_D \quad (10)$$

where hyperparameter β controls the effect from POI context and the discriminator $\mathcal{R}_D = \exp(D_\phi(u_i, l^k))$, both of which can be learned from the data. Then, Eq.(8) can be reformulated as:

$$\nabla_\theta \mathcal{J}^R(u_i) \backsimeq \frac{1}{K} \sum_{k=1}^{K} \nabla_\theta \log R_\theta(l^k|u_i) \log(\beta(\alpha \mathcal{R}_{geo}$$
$$+ (1 - \alpha)\mathcal{R}_{soc}) + (1 - \beta) \exp(D_\phi(u_i, l^k))) \quad (11)$$

Essentially, reward $\mathcal{R}$ acts as a regularizer to the recommended POIs from the recommender $R_\theta(l^k|u_i)$ – which outputs the user preference probability over POIs (also a $1\times M$ vector). As the training process goes, $\mathcal{R}$ may gradually push recommender R_θ to produce the POIs matching the preference of the user. It has been proved that if we know the true preference distribution of users, the above adversarial minimax training of APOIR can achieve Nash equilibrium [8] – the recommender exactly fits the true distribution of the user preference, i.e., $R_\theta(\mathcal{L}|u_i) = p_{true}(\mathcal{L}|u_i)$, the discriminator cannot distinguish the recommended POIs from the truth, i.e., the probability of l^R being preferred by u_i based on $D_\phi(u_i, l^R)$ is close to 0.5.

The overall logic of adversarial POI recommendation is summarized in Algorithm 1.

4.3 Discussion

The complexity of the APOIR training is linear in the number of GAN iterations, each of which has a time complexity $O(NK|\widetilde{\mathcal{L}}^{u_i}|)$ in terms of the number of candidate POIs $\widetilde{\mathcal{L}}^{u_i}$. We note that possible improvements can result from reducing the size of $|\widetilde{\mathcal{L}}^{u_i}|$ through filtering the POIs by considering spatial and/or categorical ranking influence [9].

Although both APOIR and IRGAN [25] leverage REINFORCE method for training the generator (recommender for APOIR), we highlight their fundamental differences: (1) APOIR explicitly models the POI reward which can help better understanding and interpreting the recommendation methods; In contrast, IRGAN is a GAN based optimization method for MF; and (2) APOIR additionally incorporates a temporal GRU for modeling the user dynamic preference.

We also note that the reward $\mathcal{R}$ in APOIR only explicitly considers two POI contexts (arguably, the most important two [16]), whereas temporal and sequential factors have been incorporated in the context GRU units. Other factors such as categorical information of POIs have been used in the literature [9]. Incorporating them into the reward function Eq.(9) for better understanding of the POI context, along with incorporating POI embedding [27, 33]

Algorithm 1: Adversarial POI Recommendation.

Input: Recommender R_θ, Discriminator D_ϕ, training data S.

Initialize R_θ and D_ϕ with random parameters θ and ϕ.
Pretrain R_θ on S with MF.
repeat
  Sample K POIs l^R for each user via R_θ;
  foreach $l^k \in l^R$ **do**
    Calculate reward $\mathcal{R}$ of l^k using Eq.(10);
  end
  Update R_θ parameters via policy gradient Eq.(11);
  Recommend POIs for each user using updated R'_θ;
  Sample a set of true locations: $l^+ \sim \mathcal{L}^{u_i}$;
  Update D_ϕ parameters via Eq.(6) with l^+ and l^R;
until *converge*

Output: Recommend *top-K* POIs based on the optimal R_θ for $u_i \in \mathcal{U}$.

for capturing the context of POIs, are also potential sources of improving the Interpretability. However, it may introduce more hyperparameters to tune. How to learn the weights of different contextual factors in the reward function remains an open problem. Such explainable considerations regarding the recommendation model are subject of our future work.

5 EXPERIMENTAL OBSERVATIONS

In this section, we compare the performance of APOIR to baselines using three real-world datasets.

The hyperparameters of APOIR are empirically tuned as: $\alpha = 0.4$, $\beta = 0.7$ for all experiments. The geographical distance parameter d is set to $50km$ following [2].

Table 1: Statistics of three datasets used in experiments.

Dataset	#Users	#POIs	#Check-ins	Sparsity
Gowalla	18,737	32,510	1,278,274	99.865%
Foursquare	24,941	28,593	1,196,248	99.900%
Yelp	30,887	18,995	860,888	99.860%

Datasets: We conducted our experiments on three publicly available LBSN datasets[1]: Gowalla, Foursquare and Yelp. For all datasets, we filter out those POIs with fewer than 10 visitors and those users (usually aka. the colder-start users) with fewer than 15 check-in POIs. Since Foursquare data does not have social information, we only report results from those not considering social information. Therefore, we remove the social factor $\mathcal{R}_{soc}$ in Eq.(9) for comparison on Foursquare. And Yelp data does not have check-ins time. Thus, we replace the temporal GRU in APOIR with a matrix factorization based user representation. The datasets after pre-processing are described in Table 1.

Following previous works [16, 27], we partition each dataset into training set and test set. For each user, we use the earlier 75% check-ins as the training data and the most recent 25% check-ins as the test data. All datasets are very sparse (the frequency of most POIs being visited is extremely low). Since a POI recommender system typically aims at recommending POIs that a user has not visited before, we further merge repetitive check-ins and use the earliest one. This can also avoid a testing interaction appearing in the training set.

Baselines: We compare APOIR with 10 approaches, covering from the most popular/representative POI recommendation techniques to models using different kinds of context information:

– **USG** [28]: is a collaborative filtering-based recommendation with user preference, social influence and geographical influence simultaneously incorporated.
– **MGMPFM** [1]: combines Poisson factor model and a multi-center Gaussian based geographical modeling method.
– **LFBCA** [24]: is a link-based method that constructs a graph to model users and their relations.
– **iGSLR** [30]: exploits personalized geographical preference and social influence with FCF (friend-based CF) and KDE (kernel density estimation).
– **LORE** [32]: considers sequential influence in addition to social and geographical influence by FCF, KDE and MF.
– **IRenMF** [17]: incorporates characteristics of neighboring POIs in both individual level and region level into weighted matrix factorization for POI recommendation.
– **GeoMF** [14]: integrates spatial influence in user geographical regions and its propagation.
– **RankGeoFM** [13]: is a ranking based geographical factorization method incorporating the spatial-temporal factors.
– **GeoTeaser** [33]: a temporal POI embedding model to capture the contextual check-in information and the temporal characteristics using word2vec framework.
– **PACE** [27]: builds a word2vec-based architecture to jointly learn the embeddings of users and POIs to predict both user preference over POIs and context associated with users and POIs.

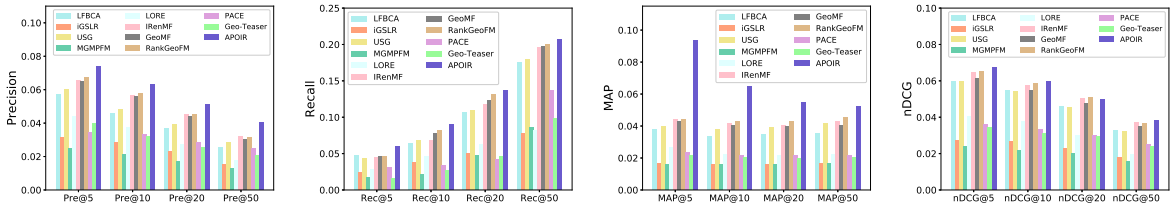
We exclude many MF-based approaches, since they have already been shown to be inferior to RankGeoFM [13]. Several recent approaches such as SG-CWARP [15] and ASMF [11] are also excluded because of their worse performance as compared to GeoTeaser [33] and PACE [27].

Metrics: We compare the model performance using four standard metrics in POI recommendation, i.e., Pre@K (precision), Rec@K (recall), nDCG@K (normalized discounted cumulative gain), and MAP@K (mean average precision) [16, 27]. They show different perspectives of the performance evaluation. Precision and recall measure the number of correct recommendations, while nDCG and MAP consider the rank of the recommendations by assigning higher score to hits at higher positions. We report the average score for all methods, and perform one-sample paired t-test to judge the statistical significance where necessary.

5.1 Results

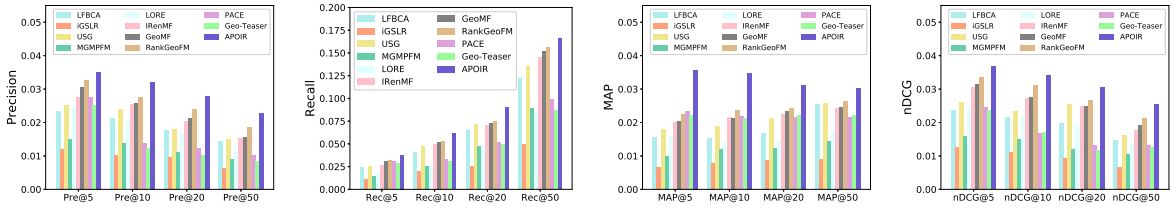
Overall Performance: Figures 1-3 illustrate the comparisons among different methods.From the results, we have following observations: **(1)** APOIR consistently performs the best and significantly improves the POI recommendation performance over the baselines on all metrics across datasets. Take the Yelp data for example (Figure 2), APOIR achives 10.0% on Pre@5, 16.0% on Rec@5, 23.3% on MAP@5 and 9.4% on nDCG@5 over RankGeoFM which performs

[1] http://spatialkeyword.sce.ntu.edu.sg/eval-vldb17/



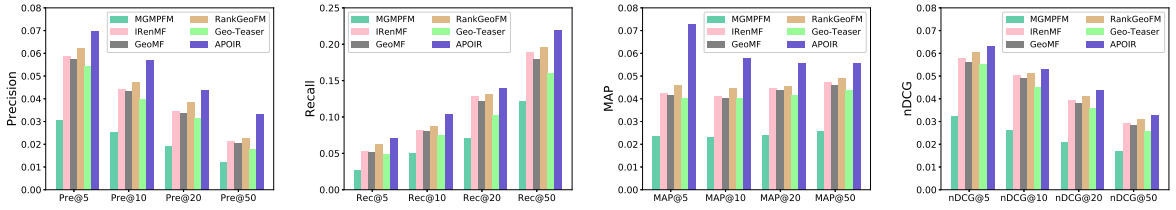
(a) Precision (b) Recall (c) MAP (d) nDCG

Figure 1: Comparisons Among Different Algorithms on Gowalla Data.



(a) Precision (b) Recall (c) MAP (d) nDCG

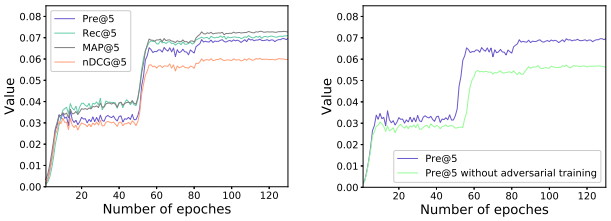
Figure 2: Comparisons Among Different Algorithms on Yelp Data.



(a) Precision (b) Recall (c) MAP (d) nDCG

Figure 3: Comparisons Among Different Algorithms on Foursquare Data.

the second best. **(2)** The strength of APOIR comes from its capability of capturing user preference with an adversarial learning process. Figure 4(a) plots the training process of APOIR. We observe in experiments that discriminator exhibits strong ability at the beginning of the training but deteriorates as the recommender grows to produce more competitive POIs – Figure 4(b) compares the performance with or without adversarial training. **(3)** MF-based methods, such as IRenMF, GeoMF and RankGeoFM, outperform other baselines which proves the effectiveness of MF in modeling latent features underpinning the complex interactions between users and POIs and the implicit feedback from the geographical information. **(4)** The most recent context embedding-based models such as GeoTeaser and PACE do not exhibit the performance as expected. We conjecture that the dataset used in their released implementation is relatively small and dense. Thus, it is difficult to obtain similar performance with larger and sparser datasets – having more POIs and more low-frequency visited POIs may result in worse embedding in word2vec[20].



(a) APOIR training (b) Effect of adversarial training

Figure 4: Training Process.

6 CONCLUSIONS

We proposed a novel approach – APOIR – for POI recommendation by learning the underlying user preference over POIs in an adversarial manner. APOIR leverages different contexts into the rewards in the reinforcement learning and adopts a generative framework for training two competing components: a recommender and a discriminator. By pushing each other to be close to its limit, the recommender may approach the true preference of users (upon which POIs are sampled and recommended), reaching an equilibrium where it becomes difficult for the discriminator to distinguish these generated POIs from the truly visited ones. Comprehensive experiments on three datasets have demonstrated the effectiveness

of APOIR, with a significant performance improvement on POI recommendation when compared to existing methods.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China (Grant No.61602097 and No.61472064), NSF grants III 1213038 and CNS 1646107, and ONR grant N00014-14-10215.

REFERENCES

[1] Chen Cheng, Haiqin Yang, Irwin King, and Michael R Lyu. 2012. Fused Matrix Factorization with Geographical and Social Influence in Location-Based Social Networks. *AAAI* (2012).
[2] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and Mobility: User Movement in Location-Based Social Networks. In *KDD*.
[3] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555* (2014).
[4] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized Ranking Metric Embedding for Next New POI Recommendation.. In *IJCAI*.
[5] Qiang Gao, Goce Trajcevski, Fan Zhou, Kunpeng Zhang, Ting Zhong, and Fengli Zhang. 2018. Trajectory-based Social Circle Inference. In *SIGSPATIAL*.
[6] Qiang Gao, Fan Zhou, Kunpeng Zhang, and Goce Trajcevski. 2017. Identifying Human Mobility via Trajectory Embeddings. In *IJCAI*.
[7] Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Zhong Ting, and Fengli Zhang. 2019. Predicting Human Mobility via Variational Attention. In *WWW*.
[8] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *NIPS*.
[9] Jing He, Xin Li, and Lejian Liao. 2017. Category-aware Next Point-of-Interest Recommendation via Listwise Bayesian Personalized Ranking. In *IJCAI*.
[10] Yehuda Koren, Robert M Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *IEEE Computer* (2009).
[11] Huayu Li, Yong Ge, Richang Hong, and Hengshu Zhu. 2016. Point-of-Interest Recommendations: Learning Potential Check-ins from Friends. In *KDD*.
[12] Huayu Li, Yong Ge, Defu Lian, and Hao Liu. 2017. Learning User's Intrinsic and Extrinsic Interests for Point-of-Interest Recommendation: A Unified Approach. In *IJCAI*.
[13] Xutao Li, Gao Cong, Xiao-Li Li, and Shonali Krishnaswamy. 2015. Rank-GeoFM: A Ranking based Geographical Factorization Method for Point of Interest Recommendation. In *SIGIR*.
[14] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *KDD*.
[15] Xin Liu, Yong Liu, and Xiaoli Li. 2016. Exploring the Context of Locations for Personalized Location Recommendations. *IJCAI* (2016).
[16] Yiding Liu, Tuan-Anh Pham, Gao Cong, and Quan Yuan. 2017. An Experimental Evaluation of Point-of-interest Recommendation in Location-based Social Networks. *PVLDB* (2017).
[17] Yong Liu, Wei Wei, Aixin Sun, and Chunyan Miao. 2014. Exploiting Geographical Neighborhood Characteristics for Location Recommendation. In *CIKM*.
[18] Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. 2017. A Deep Recurrent Collaborative Filtering Framework for Venue Recommendation. In *CIKM*.
[19] Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. 2018. A Contextual Attention Recurrent Architecture for Context-Aware Venue Recommendation. In *SIGIR*.
[20] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *NIPS*.
[21] Tuan-Anh Nguyen Pham, Xutao Li, and Gao Cong. 2017. A General Model for Out-of-town Region Recommendation. In *WWW*.
[22] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*.
[23] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. *NIPS* (1999).
[24] Hao Wang, Manolis Terrovitis, and Nikos Mamoulis. 2013. Location recommendation in location-based social networks using user check-in data. In *SIGSPATIAL*.
[25] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In *SIGIR*.
[26] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* (1992).
[27] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. 2017. Bridging Collaborative Filtering and Semi-Supervised Learning: A Neural Approach for POI Recommendation. In *KDD*.
[28] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *SIGIR*.
[29] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN - Sequence Generative Adversarial Nets with Policy Gradient. In *AAAI*.
[30] Jia-Dong Zhang and Chi-Yin Chow. 2013. iGSLR: personalized geo-social location recommendation: a kernel density estimation approach. In *SIGSPATIAL*.
[31] Jia-Dong Zhang and Chi-Yin Chow. 2015. GeoSoCa: Exploiting Geographical, Social and Categorical Correlations for Point-of-Interest Recommendations. In *SIGIR*.
[32] Jia-Dong Zhang, Chi-Yin Chow, and Yanhua Li. 2014. LORE: exploiting sequential influence for location recommendations. In *SIGSPATIAL*.
[33] Shenglin Zhao, Tong Zhao, Irwin King, and Michael R Lyu. 2017. Geo-Teaser: Geo-Temporal Sequential Embedding Rank for Point-of-interest Recommendation. In *WWW*.
[34] Wei Zhao, Benyou Wang, Jianbo Ye, Yongqiang Gao, Min Yang, Zhou Zhao, and Xiaojun Chen. 2017. Leveraging Long and Short-term Information in Content-aware Movie Recommendation. *arXiv.org* (2017).
[35] Fan Zhou, Qiang Gao, Kunpeng Zhang, Goce Trajcevski, Zhong Ting, and Fengli Zhang. 2018. Trajectory-User Linking via Variational AutoEncoder. In *IJCAI*.
[36] Fan Zhou, Bangying Wu, Yi Yang, Goce Trajcevski, Kunpeng Zhang, and Ting Zhong. 2018. vec2Link: Unifying Heterogeneous Data for Social Link Prediction. In *CIKM*.
[37] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to Do Next - Modeling User Behaviors by Time-LSTM. In *IJCAI*.

Context-aware Variational Trajectory Encoding and Human Mobility Inference

Fan Zhou
University of Electronic Science and Technology of China
fan.zhou@uestc.edu.cn

Xiaoli Yue
University of Electronic Science and Technology of China
yue.xiaol16@gmail.com

Goce Trajcevski
Iowa State University, Ames
gocet25@iastate.edu

Ting Zhong*
University of Electronic Science and Technology of China
zhongting@uestc.edu.cn

Kunpeng Zhang
University of Maryland, College park
kzhang@rhsmith.umd.edu

ABSTRACT

Unveiling human mobility patterns is an important task for many downstream applications like point-of-interest (POI) recommendation and personalized trip planning. Compelling results exist in various sequential modeling methods and representation techniques. However, discovering and exploiting the *context* of trajectories in terms of abstract topics associated with the motion can provide a more comprehensive understanding of the dynamics of patterns. We propose a new paradigm for moving pattern mining based on learning trajectory context, and a method – *Context-Aware Variational Trajectory Encoding and Human Mobility Inference (CATHI)* – for learning user trajectory representation via a framework consisting of: (1) a variational encoder and a recurrent encoder; (2) a variational attention layer; (3) two decoders. We simultaneously tackle two subtasks: (T1) recovering user routes (*trajectory reconstruction*); and (T2) predicting the trip that the user would travel (*trajectory prediction*). We show that the encoded contextual trajectory vectors efficiently characterize the hierarchical mobility semantics, from which one can decode the implicit meanings of trajectories. We evaluate our method on several public datasets and demonstrate that the proposed CATHI can efficiently improve the performance of both subtasks, compared to state-of-the-art approaches.

KEYWORDS

human mobility, variational inference, encoder-decoder

ACM Reference Format:
Fan Zhou, Xiaoli Yue, Goce Trajcevski, Ting Zhong, and Kunpeng Zhang. 2019. Context-aware Variational Trajectory Encoding and Human Mobility Inference. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313608

*Corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313608

1 INTRODUCTION

The past decade has witnessed a rapid growth of both academic and practical interest on mining human mobility patterns from location-based social networks (LBSN) such as Twitter and Foursquare. Availability of large volumes of LBSN data has spurred research in studying user behavior and movement patterns [13, 31], point-of-interest (POI) recommendation [16], trip planning [22], along with various privacy-preserving issues [28].

Majority of the existing studies focus on modeling the sequential patterns at *location-* or *POI*-level. Various techniques such as leveraging Markov Chain (MC), Matrix Factorization (MF), Recurrent Neural Networks (RNN), or the word2vec framework have been developed towards various application tasks. For example, in POI recommendation, the state-of-the-art methods are MF [20] and RNN [26] based models that learn user preference over POIs and include the contextual influence of neighboring POIs for generating implicit user-location feedback matrices [25]. More recently, the representation learning such as word2vec [27] has been introduced for embedding POI (and associated features) into latent space [12, 13, 40]. These approaches operate at a fine-grained POI level – which is also the case for trip recommendations, where a sequence of POIs are organized as an itinerary, with various other features (e.g., location, time, distance, user profile and social interactions, etc.) incorporated. In addition, it is expected to learn the POI transitions with MC model [6] or maximize the reward of the selected POIs, e.g., through Monte Carlo Tree Search (MCTS) [21].

Complementary to these, some trajectory mining tasks aim at uncovering implicit mobility patterns, clustering or classifying trajectories – which are typically reduced to location-level learning. For example, sequential and periodic pattern [9, 35] mining requires measuring location or trajectory (a sequence of locations) similarity with various time-series metrics [8], while trajectory clustering groups together similar ones. However, the *Trajectory Context Learning (TCL)* focuses on understanding and encoding the semantics of a trajectory at *trajectory level*. As a higher level of mobility semantic learning, discovering the context of trajectories may enable more comprehensive understanding of the human mobility patterns and provide additional benefits for downstream applications such as POI/trip recommendation.

In this paper, we take initial steps towards learning trajectory context by adapting the problem to an encoder-decoder framework,

inspired by recent progress of representation learning in natural languages processing (NLP). Specifically, we investigate the TCL problem by simultaneously addressing two (sub)tasks: (T1) recovering the user's route by (*Trajectory Reconstruction*) and (T2) predicting the trip that she would travel (*Trajectory Prediction*). To accomplish this, we propose the *Context-Aware Variational Trajectory Encoding and Human Mobility Inference (CATHI)* method for embedding the contexts of trajectories in an unsupervised manner within a seq2seq framework which consists of two encoders and two decoders, tuned with a novel variational attention mechanism. The two sub-tasks of TCL problem are then trained in parallel with the proposed model. In summary, our main contributions are:

- We address the TCL problem in LBSN applications as a novel learning paradigm for analyzing human mobility patterns. We propose an approach for encoding the semantics of trajectories and inferring the trajectory context, which is the first context-aware trajectory learning model and opens a new perspective for understanding user check-in behaviors.
- The proposed CATHI method leverages two variational latent factors which are respectively the traditional latent variables and the variational hidden state attention, for learning trajectory-level context. This novel variational architecture is not only capable of estimating the probability density and optimizing the lower bound of the data likelihood, but also captures the sequential and temporal characteristics of human mobility.
- We address the trajectory prediction and reconstruction simultaneously using the proposed CATHI method, which is evaluated on several public LBSN datasets and compared to competing models, demonstrating that it achieves state-of-the-art performance, which also reflects upon the efficiency of TCL.

2 OUR PROPOSED METHOD: CATHI

We now formally define the TCL problem and introduce the CATHI method consisting of two subtasks – (T1): trajectory reconstruction and (T2): trajectory prediction.

Given a set of POIs $\{l_1, l_2, ..., l_m\}$ and a set of trajectories $\mathbf{T} = \{T_1, T_2, ..., T_n\}$, where T_i is a sequence of POIs generated by a particular user, let $\mathbf{T}^r$ denote the training data consisting of a set of variable-length trajectories generated by certain users in a particular area (e.g., in a city or a scenic spot). We denote the t^{th} POI for trajectory T_i as l_i^t and its embedding vector (with word2vec [27]) as $\mathbf{v}_i^t \in \mathbb{R}^d$, where d is the POI embedding dimensionality.

2.1 Overview

The main objective of TCL is to produce two surrounding trajectories $\mathbf{T}_i^s = \{T_{i-1}, T_{i+1}\}$ for a given trajectory $T_i \in \mathbf{T}^r$. Generally, given a tuple $< T_{i-1}, T_i, T_{i+1} >$, where T_i is the i^{th} trajectory in $\mathbf{T}^r$, we are interested in coupling the reconstruction of T_{i-1} and prediction of T_{i+1}. That is, let $T_i = \{l_i^1, l_i^2, ..., l_i^N\}$ be a trajectory with N POIs, TCL tries to construct the previous trajectory $T_{i-1} = \{l_{i-1}^1, l_{i-1}^2, ..., l_{i-1}^Q\}$ and predict the next trajectory $T_{i+1} = \{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^M\}$, respectively.

To this end, our CATHI model consists of four components (shown in Figure 1): (1) *recurrent encoder*, (2) *variational encoder*, (3) *variational attention layer*, and (4) *two decoders*. More specifically, a recurrent encoder maps a sequence of POIs to a set of hidden states $\{\mathbf{h}_i^1, \mathbf{h}_i^2, ..., \mathbf{h}_i^N\}$. Meanwhile, a variational encoder learns the distribution of the latent variable z. Subsequently, a variational attention on the hidden states of recurrent encoder learns the distribution of another latent variable $\mathbf{c}$. Finally, two decoders, conditioned on two latent variables z and $\mathbf{c}$, are employed to generate the previous trajectory and the next one respectively. Both decoders have a two-layer architecture in which the first-layer decoder $\texttt{Dec}_1$ alone can be employed to reconstruct the surrounding trajectories, and the second-layer decoder $\texttt{Dec}_2$ is added to refine the results of $\texttt{Dec}_1$.

2.2 Trajectory Context Encoding

Throughout this work, the Trajectory Context Encoding (TCE) relies on variational autoencoder [19] and Long Short-Term Memory (LSTM) [18] – one spatio-temporal LSTM for the encoder and two variants of LSTM for the decoders. Details will be presented below.

2.2.1 Recurrent Trajectory Encoder. The recurrent encoder is a bi-directional LSTM. At each time step t, one can obtain an annotation for each POI by concatenating the forward and backward hidden states, i.e., $\mathbf{h}_i^t = [\overrightarrow{\mathbf{h}}_i^t; \overleftarrow{\mathbf{h}}_i^t]$, interpreted as the representation of the trajectory $l_i^1, l_i^2, ..., l_i^t$. The last hidden state $\mathbf{h}_i^N$ thus represents the full trajectory T_i.

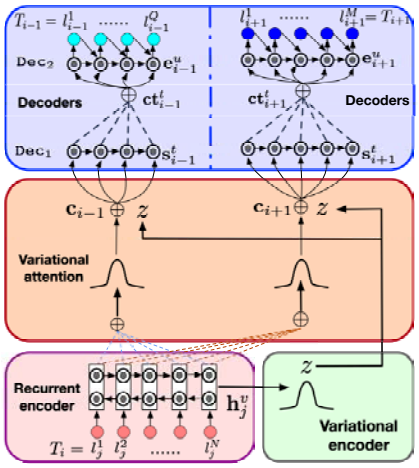


Figure 1: Trajectory context encoding with two encoders and two decoders with each having two layers.

Previous analysis shows that POIs' contextual information, such as geographical location and social information, has influence over user check-in behavior. Integrating them into the model might be helpful for better representation of users. However, vanilla LSTM does not consider the contextual information associated with POIs. To address this limitation, we present an extension of LSTM that is capable of encoding the spatio-temporal features in check-ins.

Given a sequence of check-ins $l_i^1, \cdots, l_i^t, \cdots, l_i^N$, each l_i^t associated with a check-in time τ_i^t and a geo-location g_i^t, we compute both time interval and distance between adjacent check-ins as $\Delta_{\tau_i^t} = \tau_i^t - \tau_i^{t-1}$ and $\Delta_{g_i^t} = g_i^t - g_i^{t-1}$, $t \in [1, N]$. Then we add a time gate $\mathbf{t}_i^t$ and a geographical gate $\mathbf{g}_i^t$ and modify the candidate

gate $\tilde{\mathbf{h}}_i^t$ as

$$\begin{aligned}
\mathbf{t}_i^t &= \sigma(W_t \mathbf{v}_i^t + \sigma([\Delta_{\tau_i^t}; \tau_i^t] W_t) + b_t) \\
\mathbf{g}_i^t &= \sigma(W_g \mathbf{v}_i^t + \sigma(\Delta_{g_i^t} W_g) + b_g) \\
[\mathbf{i}_i^t, \mathbf{f}_i^t, \mathbf{o}_i^t] &= \sigma(W \mathbf{v}_i^t + U \mathbf{h}_i^{t-1}) \\
\tilde{\mathbf{h}_i^t} &= \tanh(W' \mathbf{v}_i^t + U'(\mathbf{t}_i^t \odot \mathbf{g}_i^t \odot \mathbf{h}_i^{t-1})) \\
\mathbf{m}_i^t &= \mathbf{f}_i^t \mathbf{m}_{i-1}^t + \mathbf{i}_i^t \tilde{\mathbf{h}_i^t} \\
\mathbf{h}_i^t &= \tanh(\mathbf{m}_i^t) \odot \mathbf{o}_i^t
\end{aligned} \tag{1}$$

where $\mathbf{v}_i^t$ is the embedding vector of the POI l_i^t, $[\Delta_{\tau_t^t}; \tau_t^t]$ is a concatenation of time interval and the current check-in time; $\mathbf{i}_i^t, \mathbf{f}_i^t, \mathbf{o}_i^t$ are input, forget and output gates in vanilla LSTM; W and U are parameter matrices w.r.t different gates. Now the two new gates capture spatio-temporal preference of users, which is used to control the influence of previous hidden state $\mathbf{h}_i^{t-1}$ in Eq.(1).

Note that above contextual LSTM is employed in the recurrent encoder. The hidden state $\mathbf{h}_i^t$ contains the information from both the preceding POIs and the following POIs in T_i. Due to the tendency of RNNs to better represent recent inputs, the hidden state $\mathbf{h}_i^t$ will focus more on the POIs around current check-in l_i^t (cf. [1]). Instead of only attention to different words in NLP encoder such as [5], we follow the typical assumption in trajectory modeling that adjacent POIs have more (related) information than long-distance ones, both temporally and geographically. Such constraints are achieved by the proposed spatio-temporal gates. This sequence of hidden states, combined with the following variational variables, can later be used to compute context vectors in the decoder.

2.2.2 Variational Trajectory Encoder. We use a variational encoder [19] for encoding the trajectory as a latent variable z, which is then used to reconstruct the original trajectory under the generative model: given a trajectory T_i, its likelihood is

$$\begin{aligned}
\log p_\theta(T_i) &= \log p_\theta(T_i) \int_z q_\phi(z|T_i) \mathrm{d}z \geq \triangleq \mathcal{L}(T_i) \\
&= \mathbb{E}_{z \sim q_\phi(z|T_i)}[\log p_\theta(T_i|z)] - \mathbb{KL}\left[q_\phi(z|T_i)||p_\theta(z)\right]
\end{aligned} \tag{2}$$

where $q_\phi(z|T_i)$ is an approximation to the true posterior $p_\theta(z|T_i)$ (a.k.a *recognition model* or *encoder*) parameterized by ϕ. $\mathbb{KL}\left[q_\phi(\cdot)||p_\theta(\cdot)\right]$ is the KL divergence between the learned latent posterior distribution $q(z|T_i)$ and the prior $p(z)$ (for brevity, we will omit the parameters ϕ and θ in subsequent formulas). Since the objective (of the model) is to minimize the KL divergence between the proposal $q(z|T_i)$ and the true distribution $p(z|T_i)$ – we can alternatively maximize the evidence lower bound (ELBO) $\mathcal{L}(T_i)$ w.r.t. both θ and ϕ, which are jointly trained with separate neural networks such as multi-layer perceptrons.

2.2.3 Variational attention. Traditional attention mechanisms [1] are used for dynamically aligning the input and output sequences. Specifically, we compute a probabilistic distribution by

$$\gamma_{vt} = \frac{\exp(\mathbf{s}_{i\pm1}^v W^\intercal \mathbf{h}_i^t)}{\sum_{t'=1}^N \exp(\mathbf{s}_{i\pm1}^v W^\intercal \mathbf{h}_i^{t'})} \tag{3}$$

where $\mathbf{s}_{i\pm1}^v$ denotes the v^{th} hidden state of previous (next) trajectory, $\mathbf{h}_i^t$ is t^{th} hidden hidden state of recurrent encoder, and W is the parameter matrix needed to be learneded. Then, the attention vector $\mathbf{c}_{i\pm1}$ is calculated by summing the weighted input:

$$\mathbf{c}_{i\pm1}^v = \sum_{t=1}^N \gamma_{vt} \mathbf{h}_i^t \tag{4}$$

which is fed into the decoder at v^{th} step. We will denote the attention vector as $\mathbf{c}$ for simplicity.

However, if we directly use above attention vector $\mathbf{c}$, this deterministic attention is powerful enough to reconstruct the input and may eliminate the influence of the variational encoder – a.k.a. "bypassing" phenomena in combining VAE and seq2seq models [3]. Therefore, we introduce a variational attention on the hidden states of the recurrent encoder inspired by [2]. That is, the attention vector $\mathbf{c}$ is also treated as a latent factor as z, both of which are combined to reconstruct the input data by maximizing the ELBO:

$$\begin{aligned}
\mathcal{L}(T_i) &= \mathbb{E}[\log p(T_i|z, \mathbf{c})] - \mathbb{KL}\left[q(z, \mathbf{c}|T_i)||p(z, \mathbf{c})\right] \\
&= \mathbb{E}[\log p(T_i|z, \mathbf{c})] - \rho \mathbb{KL}\left[q(z|T_i)||p(z)\right] \\
&\quad - \psi \mathbb{KL}\left[q(\mathbf{c}|T_i)||p(\mathbf{c})\right]
\end{aligned} \tag{5}$$

which is derived based on the fact that $\mathbf{c}$ and z are conditionally independent given T_i. The prior $p(\mathbf{c})$ is initialized via a Gaussian $p(\mathbf{c}) = \mathcal{N}(\bar{\mathbf{h}}_i, \mathbf{I})$ with $\bar{\mathbf{h}}_i$ as the mean of the hidden states of the recurrent encoder. At the early stage of variational training, the terms $\mathbb{KL}\left[q(\cdot|T_i)||p(\cdot)\right]$ in Eq.(5) may discourage encoding interesting information into the latent variables z and $\mathbf{c}$, thereby easily resulting in model collapse – largely because of the strong learning capability of autoregressive models such as RNN as observed in [3]. One efficient solution is to control the weight of KL-divergence term by gradually increasing its co-efficiencies ρ and ψ (from 0 to 1), which is also called KL cost annealing [3].

2.2.4 Trajectory Decoders. As mentioned above, there are two decoders in CATHI: two LSTMs trained to reconstruct the previous trajectory T_{i-1} and predict the next trajectory T_{i+1}, respectively.
– **Decoder 1.** In the case of trajectory T_{i+1} decoding, the first decoder $\texttt{Dec}_1$ predict next POI l_{i+1}^t given: (1) all the previously predicted POIs $\{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^{t-1}\}$; (2) the variational attention vector $\mathbf{c}$; and (3) the variational latent variable z. For example, the decoder is trained to maximize the probability of $\{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^M\}$ given trajectory T_i which can be decomposed into the ordered conditional probabilities:

$$p(\{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^M\}) = \prod_{t=1}^M p(l_{i+1}^t | l_{i+1}^{<t}, \mathbf{c}, z) \tag{6}$$

where $l_*^{<t}$ denote previous $t-1$ POIs in the next trajectory. Each conditional probability $p(l_{i+1}^t | l_{i+1}^{<t}, \mathbf{c}, z)$ depends on the context vectors $\mathbf{c}$, z and all the previous POIs $\{l_{i+1}^1, l_{i+1}^2, ..., l_{i+1}^{t-1}\}$. With the first LSTM decoder, the conditional probability can be modeled as:

$$p(l_{i+1}^t | l_{i+1}^{<t}, \mathbf{c}, z) = \mathrm{LSTM}(\mathbf{v}_{i+1}^{t-1}, \mathbf{c}, z, \mathbf{s}_{i+1}^{t-1}) \tag{7}$$

where $\mathbf{s}_{i+1}^{t-1}$ is the hidden state of the decoder LSTM.

Therefore, given a tuple $< T_{i-1}, T_i, T_{i+1} >$, the objective of trajectory context encoding is to maximize the log-probabilities for the previous and next trajectories of T_i:

$$\mathcal{J} = \log \prod_t p(l_{i+1}^t | l_{i+1}^{<t}, \mathbf{c}_{i+1}, z) p(l_{i-1}^t | l_{i-1}^{<t}, \mathbf{c}_{i-1}, z) \tag{8}$$

where the two decoders (for T_{i+1} and T_{i-1}) focus on different POIs in T_i with distinct context vectors $\mathbf{c}_{i\pm1}$, iterating through all the tuples in the dataset during each epoch of the TCL training.

– Decoder 2. Intuitively, the first decoder(s) already can be employed to reconstruct the surrounding trajectories of T_i. However, the generated trajectories (both previous and next one) suffer the "myopic" posterior in the sense that the POIs generated previously at time step t are not informed about the whole future in the sequence – recall that at time step t, only the POIs $l_*^{<t}$ are used in Equation (8) without considering possible POIs $l_*^{>t}$ thereafter. We now introduce a refinement process to CATHI when inferring the around trajectories.

Specifically, we add a second decoder $\texttt{Dec}_2$ upon the first decoder $\texttt{Dec}_1$ to refine the final trajectory. In contrast to $\texttt{Dec}_1$, $\texttt{Dec}_2$ exploits the information regarding all the POIs of the generated trajectories by $\texttt{Dec}_1$ at each time step. In addition to the previous hidden state and previously generated POIs, $\texttt{Dec}_2$ leverages the context vector $\mathbf{ct}_{i\pm1}^t$, the weighted sum of the hidden states of $\texttt{Dec}_1$, to "polish" the generated trajectory, i.e., T_{i-1} and T_{i+1}. That is, $\texttt{Dec}_2$ updates the context vector $\mathbf{ct}_{i+1}^t$ at time step t as:

$$\mathbf{ct}_{i+1}^t = \sum_{u=1}^{M} \beta_{ut} \mathbf{s}_{i+1}^u$$

This is actually another attention vector on the first decoder weighted by a probabilistic distribution

$$\beta_{ut} \propto \exp\{\mathbf{e}_{i+1}^u W^\intercal \mathbf{s}_{i+1}^t\} \tag{9}$$

where $\mathbf{e}_{i+1}^u$ is the hidden state of $\texttt{Dec}_2$ and $\mathbf{s}_{i+1}^t$ is the hidden state of $\texttt{Dec}_1$.

2.2.5 Inference. After describing how to learn the context of trajectories, we now turn our attention from training to trajectory inference for addressing the two subtasks – trajectory reconstruction and trajectory prediction. In many seq2seq based tasks such as neural machine translation [1] and dialogue generation [3], beam search is the *de facto* decoding method at test time, due to the shrunk search space and reduced computational complexity. Beam search maintains a set of B highest-scoring beams which are prefixes of trajectories. At each time step, every possible POI is added to every existing beam, yielding a total of $B \times D$ beams where D is the number of all possible POIs. Then we discard all but the B most likely beams according to the log-likelihood of the trained model. This procedure is repeated until a segment ends with the end-of-trajectory token or reaches the preset maximum length of trajectories. The segment that ends with end-of-trajectory token will then be moved to the set of complete hypotheses. Finally, we choose the one with highest probability from the hypotheses.

3 EVALUATION

We now present the results of our evaluation regarding the performance of TCL solutions with the two context-aware trajectory inference tasks: trajectory reconstruction and prediction.

Datasets. We conducted our experiments on three datasets shown in Table 1. The trajectories extracted from Flickr photos and videos are typically used for trip recommendation [6, 21, 22]. **Toronto, Osaka, Glasgow and Edinburgh** data were built by Lim et al. [22]; **Melbourne** data were built by [6], and **Cali.Adv, Hollywood, Epcot, DisneyLand and MagicKing** data were built by [21]. **Foursquare** dataset [34] is often used for POI recommendation and contains 573,703 check-ins in Tokyo collected for approximately 10 months (from 12 April 2012 to 16 February 2013). We randomly selected 400 users and their historical trajectories for evaluation. **Geolife** dataset [38] was collected in Microsoft Geolife project in a period of over five years (from April 2007 to August 2012). Since the trajectories in original Geolife dataset contain only GPS points (longitude and latitude), we cluster all points based on their locations to obtain 4,796 POIs – that is, we save the two digits after the decimal point of longitude and latitude. For all the datasets, we randomly chose 90% trajectories for training and the rest for testing.

Table 1: Descriptives of datasets.

Dataset	#POI	#Trajectory	#User
Flickr@Edinburgh	29	5,028	1,454
Flickr@Glasgow	29	2,227	601
Flickr@Melbourne	87	5,106	1,000
Flickr@Osaka	29	1,115	450
Flickr@Toronto	30	6,057	1,395
Flickr@Cali.Adv	25	6,907	2,593
Flickr@Hollywood	13	3,858	1,972
Flickr@Epcot	17	5,816	2,725
Flickr@Disneyland	31	11,758	3,704
Flickr@MagicKing	27	8,126	3,342
Foursquare@Tokyo	30	1,1254	400
Geolife	4,796	24,943	179

We note that the two tasks evaluated in our experiments are predicting and reconstructing a sequence, which are more complicated than location prediction and/or recommendation which requires predicting/recommending a point. Thus, for the two tasks, the datasets are large enough to evaluate the methods, which is quite different from the datasets used in venue recommendation/prediction [9, 26].

Baselines. Since TCL is a novel problem proposed in this work, we compare our method with several state-of-the-art approaches from the area of sequential pattern modeling and trajectory recommendation, with two metrics for measuring accuracy and visiting-order of POIs in each reconstructed and predicted trajectory. The baselines consist of:

- **Random**: A naive approach chooses at random to construct a trajectory with desired length.
- **POIPopu** [11]: It selects the most popular and unvisited POI at each time.
- **POIRank** [6]: It plans a trajectory by first ranking POIs with rankSVM and then connects them according to ranking scores.
- **Markov and MRank** [6]: Markov considers the POI-POI transition probabilities and plans a trajectory by maximizing the transition likelihood. Markov-Rank learns both POI ranking and Markov transition.
- **MPath and MPRank** [6]: MarkovPath (MPath) and MarkovPath-Rank (MPRank) eliminate sub-tours in Markov and Markov-Rank by finding the best path using an integer programming.
- **ST-RNN** [23]: is a RNN-based method incorporating spatial and temporal context for predicting the next location.

Table 2: Trajectory Reconstruction Performance Among Different Algorithms on Flickr Data.

	Edinburgh	Glasgow	Melbourne	Osaka	Toronto	Cali.Adv.	Hollywood	Epcot	Disneyland	MagicKing
					F_1					
Random	0.474 ± 0.071	0.502 ± 0.048	0.442 ± 0.098	0.497 ± 0.051	0.492 ± 0.058	0.458 ± 0.076	0.504 ± 0.043	0.488 ± 0.057	0.438 ± 0.086	0.440 ± 0.080
POIPopu	0.581 ± 0.214	0.589 ± 0.202	0.518 ± 0.174	0.569 ± 0.165	0.534 ± 0.147	0.512 ± 0.159	0.593 ± 0.189	0.618 ± 0.241	0.531 ± 0.187	0.565 ± 0.221
POIRank	0.583 ± 0.213	0.594 ± 0.208	0.531 ± 0.187	0.551 ± 0.170	0.591 ± 0.204	0.517 ± 0.167	0.596 ± 0.198	0.649 ± 0.232	0.537 ± 0.194	0.564 ± 0.220
Markov	0.550 ± 0.188	0.577 ± 0.194	0.455 ± 0.142	0.536 ± 0.173	0.566 ± 0.194	0.509 ± 0.173	0.500 ± 0.174	0.469 ± 0.165	0.500 ± 0.155	0.460 ± 0.150
MPath	0.559 ± 0.170	0.585 ± 0.193	0.486 ± 0.137	0.540 ± 0.197	0.568 ± 0.186	0.486 ± 0.145	0.536 ± 0.157	0.476 ± 0.159	0.496 ± 0.155	0.449 ± 0.130
MRank	0.601 ± 0.225	0.602 ± 0.232	0.490 ± 0.187	0.555 ± 0.201	0.590 ± 0.248	0.517 ± 0.183	0.580 ± 0.193	0.554 ± 0.208	0.512 ± 0.187	0.521 ± 0.194
MPRank	0.578 ± 0.215	0.611 ± 0.231	0.496 ± 0.168	0.574 ± 0.217	0.599 ± 0.236	0.528 ± 0.162	0.573 ± 0.173	0.583 ± 0.200	0.539 ± 0.201	0.543 ± 0.185
ST-RNN	0.665 ± 0.147	0.678 ± 0.171	0.679 ± 0.142	0.703 ± 0.159	0.708 ± 0.123	0.648 ± 0.142	0.645 ± 0.152	0.637 ± 0.168	0.611 ± 0.157	0.627 ± 0.160
DeepMove	0.679 ± 0.153	0.687 ± 0.161	0.688 ± 0.132	0.700 ± 0.152	0.726 ± 0.119	0.645 ± 0.128	0.672 ± 0.102	0.643 ± 0.170	0.618 ± 0.155	0.662 ± 0.158
CARA	0.689 ± 0.155	0.709 ± 0.112	0.669 ± 0.143	0.745 ± 0.144	0.722 ± 0.122	0.688 ± 0.130	0.645 ± 0.126	0.673 ± 0.148	0.631 ± 0.157	0.673 ± 0.150
CATHI	**0.778 ± 0.128**	**0.807 ± 0.121**	**0.715 ± 0.125**	**0.913 ± 0.079**	**0.843 ± 0.115**	**0.755 ± 0.127**	**0.796 ± 0.123**	**0.810 ± 0.136**	**0.742 ± 0.147**	**0.731 ± 0.155**
					pairs-F_1					
Random	0.102 ± 0.120	0.052 ± 0.144	0.015 ± 0.073	0.074 ± 0.160	0.088 ± 0.123	0.053 ± 0.119	0.095 ± 0.158	0.074 ± 0.143	0.040 ± 0.098	0.054 ± 0.112
POIPopu	0.207 ± 0.366	0.196 ± 0.382	0.117 ± 0.280	0.152 ± 0.316	0.100 ± 0.266	0.113 ± 0.239	0.217 ± 0.333	0.308 ± 0.409	0.150 ± 0.274	0.237 ± 0.354
POIRank	0.212 ± 0.368	0.208 ± 0.389	0.138 ± 0.305	0.130 ± 0.315	0.209 ± 0.380	0.123 ± 0.254	0.225 ± 0.346	0.345 ± 0.415	0.165± 0.292	0.235 ± 0.354
Markov	0.169 ± 0.326	0.180 ± 0.367	0.037 ± 0.169	0.116 ± 0.313	0.172 ± 0.355	0.119 ± 0.280	0.122 ± 0.280	0.153 ± 0.460	0.146 ± 0.493	0.197 ± 0.314
MPath	0.130 ± 0.292	0.177 ± 0.370	0.049 ± 0.202	0.153 ± 0.228	0.155 ± 0.344	0.088 ± 0.283	0.144 ± 0.256	0.167 ± 0.440	0.149 ± 0.485	0.123 ± 0.375
MRank	0.244 ± 0.395	0.265 ± 0.412	0.114 ± 0.273	0.164 ± 0.345	0.263 ± 0.420	0.181 ± 0.284	0.221 ± 0.311	0.213 ± 0.319	0.187 ± 0.306	0.188 ± 0.305
MPRank	0.204 ± 0.369	0.265 ± 0.415	0.080 ± 0.256	0.220 ± 0.379	0.260 ± 0.411	0.192 ± 0.311	0.176 ± 0.265	0.228 ± 0.353	0.196 ± 0.317	0.215 ± 0.332
ST-RNN	0.436 ± 0.172	0.334 ± 0.075	0.355 ± 0.109	0.445 ± 0.188	0.418 ± 0.161	0.314 ± 0.132	0.347 ± 0.116	0.309 ± 0.159	0.351 ± 0.175	0.362 ± 0.196
DeepMove	0.448 ± 0.228	0.445 ± 0.198	0.440 ± 0.194	0.542 ± 0.202	0.582 ± 0.189	0.454 ± 0.223	0.439 ± 0.215	0.402 ± 0.234	0.412 ± 0.222	0.398 ± 0.266
CARA	0.440 ± 0.230	0.403 ± 0.199	0.402 ± 0.188	0.528 ± 0.204	0.574 ± 0.182	0.446 ± 0.210	0.427 ± 0.210	0.404 ± 0.212	0.408 ± 0.206	0.412 ± 0.288
CATHI	**0.556 ± 0.210**	**0.609 ± 0.187**	**0.474 ± 0.174**	**0.782 ± 0.201**	**0.617 ± 0.179**	**0.517 ± 0.205**	**0.565 ± 0.207**	**0.590 ± 0.204**	**0.493 ± 0.211**	**0.503 ± 0.240**

- **DeepMove** [9]: a most recent POI prediction method learning user periodical patterns with attention mechanism and the sequential patterns within the recurrent neural networks.
- **CARA** [26]: a most recent POI recommendation method jointly learns the user dynamic preference and contextual information associated with check-ins under the GRU [7] architecture.

Note that the last three methods (ST-RNN, DeepMove and CARA) are not targeting sequence learning. We extend them to infer a trajectory by predicting (or reconstruct) a sequence of POIs iteratively – e.g., after predicting POI l_*^{i+1}, we incorporate l_*^{i+1} into a LSTM for predicting l_*^{i+2}, and so on.

Metrics We use F_1 and pairs-F_1 score to evaluate the two subtasks in TCL following previous works [6, 21].

3.1 Experimental Observations

Overall performance: Table 2 and 3 respectively show the experimental results of trajectory reconstruction and trajectory prediction on 10 Flickr datasets, while Table 4 shows the results on Geolife and Foursquare (we exclude others due to the limited space). Note that the values before and after '±' are respectively the mean and the standard deviation.

Firstly, CATHI consistently yields the best performance on two subtasks for both metrics, while exhibiting significant improvement over the baselines. The effectiveness of CATHI lies in its trajectory-level embedding which simultaneously encodes the context information and captures the sequential patterns of trajectories (at both POI-level and trajectory-level). On the contrary, ST-RNN, DeepMove and CARA are deep learning based methods learning mobility at the POI-level and usually exhibit the secondary performance. In addition, one can also observe that CATHI achieves the lower value of standard deviation than other methods (except Random which samples around the mean and thus has minimum value of standard deviation), which means that the results of CATHI are more stable.

On the other hand, CATHI employs contextual recurrent networks as the encoders and decoders and proves their superior performance compared to MC-based models, especially in terms of trajectory ordering (measured by pairs-F_1). Note that the performance of MC-based models may even drop to the same level (if not worse) of random and greedy methods. This is because MC (or its variants) learns the transition patterns at the POI-level and works well when predicting the next one or previous one POI. However, both two subtasks in our work require to form a POI sequence.

Moreover, we observe that the performance of trajectory reconstruction is slightly better than the trajectory prediction, which is also true for other baselines. This phenomenon motivates a decomposition of the subtasks into two steps: (1) reconstruct the trajectories of individuals, and (2) incorporate the results to improve the prediction accuracy.

Validity of components: Recall that our CATHI model consists of several functional parts. In order to investigate the effectiveness of each component, we compare a series of variants of CATHI, including: (1) CATHI-1, which uses a basic seq2seq model, with contextual LSTM for trajectory context learning; (2) CATHI-2, which incorporates a deterministic attention mechanism into CATHI-1; (3) CATHI-3, which further incorporates variational encoder-decoder into CATHI-2; (4) CATHI-4, which considers attention vector as another variational variable; and (5) the CATHI, which can be considered as adding the second decoder on CATHI-4. Obviously, these methods gradually builds up our CATHI method. Thus, we can understand the rationale behind by scrutinizing the performance of each module.

Figure 2 plots the performance of the five methods on five Flickr datasets. From the trajectory prediction task, we can see that each component of CATHI contributes to the trajectory context learning. Among various modules, deterministic attention is the most influential one. We also observe that our variational attention works well across all comparisons which proves that variational encoder-decoder can be used to enhance seq2seq model, but should consider the attention vectors as variational variables. Finally, CATHI outperforming CATHI-4 demonstrates that the second decoder indeed improve the TCL performance by refining the results of using one decoder, especially for better learning visiting order – it largely improves the pairs-F_1 results.

4 RELATED WORK

POI-level Models. Modeling the patterns of user check-ins is a primary step in most application tasks such as POI recommendation

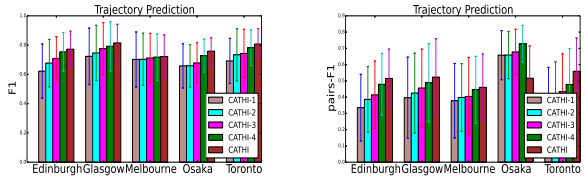
Table 3: Trajectory Prediction Performance Among Different Algorithms on Flickr Data.

	Edinburgh	Glasgow	Melbourne	Osaka	Toronto	Cali.Adv.	Hollywood	Epcot	Disneyland	MagicKing
					F_1					
Random	0.474 ± 0.071	0.503 ± 0.048	0.442 ± 0.098	0.500 ± 0.049	0.492 ± 0.058	0.424 ± 0.091	0.498 ± 0.043	0.438 ± 0.078	0.408 ± 0.099	0.406 ± 0.095
POIPopu	0.515 ± 0.154	0.542 ± 0.166	0.480 ± 0.151	0.587 ± 0.227	0.531 ± 0.167	0.494 ± 0.160	0.543 ± 0.192	0.543 ± 0.186	0.455 ± 0.166	0.465 ± 0.172
POIRank	0.518 ± 0.173	0.565 ± 0.205	0.511 ± 0.186	0.547 ± 0.173	0.568 ± 0.198	0.483 ± 0.156	0.516 ± 0.192	0.510 ± 0.157	0.453 ± 0.162	0.457 ± 0.153
Markov	0.520 ± 0.176	0.551 ± 0.169	0.457 ± 0.142	0.630 ± 0.228	0.557 ± 0.194	0.489 ± 0.172	0.524 ± 0.193	0.541 ± 0.191	0.456 ± 0.131	0.470 ± 0.159
MPath	0.550 ± 0.169	0.563 ± 0.182	0.488 ± 0.130	0.623 ± 0.189	0.575 ± 0.200	0.506 ± 0.169	0.541 ± 0.184	0.540 ± 0.182	0.469 ± 0.128	0.483 ± 0.171
MRank	0.541 ± 0.227	0.637 ± 0.244	0.465 ± 0.207	0.627 ± 0.248	0.612 ± 0.241	0.500 ± 0.195	0.531 ± 0.196	0.563 ± 0.193	0.464 ± 0.171	0.443 ± 0.147
MPRank	0.545 ± 0.248	0.642 ± 0.261	0.464 ± 0.222	0.610 ± 0.252	0.620 ± 0.261	0.492 ± 0.186	0.545 ± 0.170	0.564 ± 0.192	0.465 ± 0.108	0.467 ± 0.134
ST-RNN	0.613 ± 0.115	0.711 ± 0.086	0.669 ± 0.091	0.645 ± 0.140	0.689 ± 0.102	0.624 ± 0.113	0.643 ± 0.081	0.638 ± 0.108	0.642 ± 0.126	0.669 ± 0.121
DeepMove	0.602 ± 0.148	0.732 ± 0.162	0.701 ± 0.156	0.672 ± 0.136	0.712 ± 0.134	0.706 ± 0.149	0.682 ± 0.147	0.652 ± 0.183	0.656 ± 0.166	0.682 ± 0.155
CARA	0.616 ± 0.142	0.726 ± 0.158	0.704 ± 0.166	0.670 ± 0.140	0.704 ± 0.129	0.702 ± 0.146	0.680 ± 0.142	0.655 ± 0.115	0.650 ± 0.161	0.677 ± 0.167
CATHI	**0.772 ± 0.123**	**0.815 ± 0.127**	**0.722 ± 0.148**	**0.758 ± 0.091**	**0.807 ± 0.106**	**0.737 ± 0.129**	**0.777 ± 0.139**	**0.781 ± 0.141**	**0.737 ± 0.151**	**0.713 ± 0.152**
					pairs-F_1					
Random	0.041 ± 0.114	0.076 ± 0.116	0.012 ± 0.052	0.124 ± 0.136	0.046 ± 0.129	0.046 ± 0.103	0.095 ± 0.144	0.068 ± 0.126	0.042 ± 0.094	0.048 ± 0.100
POIPopu	0.097 ± 0.250	0.116 ± 0.301	0.064 ± 0.215	0.225 ± 0.412	0.109 ± 0.295	0.100 ± 0.232	0.169 ± 0.313	0.156 ± 0.299	0.080 ± 0.215	0.090 ± 0.232
POIRank	0.114 ± 0.283	0.174 ± 0.375	0.120 ± 0.296	0.130 ± 0.315	0.175 ± 0.362	0.091 ± 0.228	0.136 ± 0.298	0.108 ± 0.245	0.076 ± 0.199	0.072 ± 0.196
Markov	0.121 ± 0.287	0.129 ± 0.317	0.039 ± 0.174	0.283 ± 0.234	0.160 ± 0.348	0.108 ± 0.276	0.149 ± 0.302	0.176 ± 0.319	0.088 ± 0.341	0.086 ± 0.243
MPath	0.101 ± 0.285	0.144 ± 0.343	0.041 ± 0.188	0.185 ± 0.367	0.177 ± 0.374	0.105 ± 0.271	0.145 ± 0.297	0.145 ± 0.304	0.091 ± 0.334	0.087 ± 0.252
MRank	0.205 ± 0.348	0.319 ± 0.435	0.113 ± 0.280	0.313 ± 0.443	0.288 ± 0.411	0.123 ± 0.297	0.168 ± 0.322	0.221 ± 0.360	0.116 ± 0.243	0.142 ± 0.375
MPRank	0.229 ± 0.386	0.341 ± 0.460	0.124 ± 0.308	0.297 ± 0.439	0.325 ± 0.449	0.110 ± 0.275	0.166 ± 0.286	0.195 ± 0.327	0.104 ± 0.315	0.156 ± 0.324
ST-RNN	0.301 ± 0.147	0.392 ± 0.113	0.373 ± 0.122	0.305 ± 0.159	0.418 ± 0.172	0.347 ± 0.153	0.391 ± 0.109	0.303 ± 0.120	0.327 ± 0.176	0.328 ± 0.145
DeepMove	0.334 ± 0.152	0.398 ± 0.142	0.392 ± 0.156	0.362 ± 0.162	0.424 ± 0.188	0.352 ± 0.195	0.428 ± 0.198	0.402 ± 0.202	0.420 ± 0.199	0.368 ± 0.198
CARA	0.368± 0.198	0.340 ± 0.209	0.391 ± 0.183	0.394 ± 0.186	0.422 ± 0.119	0.360 ± 0.202	0.422 ± 0.201	0.416 ± 0.220	0.428 ± 0.212	0.385 ± 0.221
CATHI	**0.515 ± 0.180**	**0.523 ± 0.236**	**0.460 ± 0.205**	**0.516 ± 0.199**	**0.559 ± 0.205**	**0.431 ± 0.195**	**0.515 ± 0.201**	**0.533 ± 0.206**	**0.429 ± 0.209**	**0.422 ± 0.212**

Table 4: F_1 and pairs-F_1 Comparisons Among Different Algorithms on Geolife and Foursquare data.

		Trajectory Reconstruction		Trajectory Prediction	
	Methods	F1	Pairs-F1	F1	Pairs-F1
Geolife	Random	0.148±0.077	0.048±0.086	0.141±0.067	0.044±0.083
	POIPopu	0.190±0.138	0.003±0.026	0.163±0.122	0.114±0.092
	POIRank	0.188±0.133	0.002±0.025	0.157±0.124	0.128±0.122
	ST-RNN	0.502±0.111	0.158±0.088	0.509±0.107	0.223±0.138
	DeepMove	0.490±0.212	0.203±0.223	0.502±0.221	0.228±0.232
	CARA	0.502±0.111	0.158±0.088	0.509±0.107	0.223±0.138
	CATHI	**0.536±0.109**	**0.318±0.110**	**0.520±0.102**	**0.358±0.104**
Foursquare	Random	0.508±0.036	0.026±0.107	0.489±0.040	0.028±0.105
	POIPopu	0.555±0.158	0.117±0.307	0.532±0.170	0.109±0.305
	POIRank	0.583±0.189	0.172±0.365	0.550±0.189	0.142±0.342
	Markov	0.533±0.153	0.074±0.247	0.551±0.159	0.111±0.307
	MPath	0.527±0.143	0.065±0.236	0.599±0.124	0.147±0.209
	ST-RNN	0.609±0.153	0.333±0.210	0.642±0.116	0.342±0.163
	DeepMove	0.702±0.167	0.412±0.212	0.678±0.121	0.359±0.192
	CARA	0.719±0.169	0.425±0.228	0.684±0.119	0.365±0.166
	CATHI	**0.809±0.145**	**0.616±0.190**	**0.749±0.087**	**0.473±0.150**

and trip planning. The sequential patterns of check-ins are usually modeled by various machine learning techniques, such as Markov Chains (MC) [6, 17], matrix/tensor factorization [20], pairwise ranking model [16], Recurrent Neural Networks (RNN) [9, 13, 13, 23, 26] and distributed representation methods [10, 24, 32, 33, 37] (e.g. word2vec [27]). Trip planning leverages the spatio-temporal check-ins for recommending a sequence of POIs (POI-level modeling paradigm) and a use of MC to model the POI→POI transition by learning from historical behavior and trajectories have been discussed in [6, 21]. This line of researches focuses on modeling the sequential patterns of POIs (combined with various features of POIs and users) for solving different application tasks. However, they fail to incorporate the *context of trajectories* which is the objective of our work.



(a) F_1. (b) pairs-F_1.

Figure 2: Comparison of modules in CATHI.

Trajectory Pattern Learning. Uncovering semantic patterns characterizing human trajectory has been studied extensively, and the results can be categorized into: (1) *statistical patterns learning*: measuring and quantifying models such as continuous-time random-walk [4] and Lévy flight [15], accounting for characteristics of individual human trajectories [29]. (2) *similarity mining*: measuring the similarity or distance between two trajectories, such as Dynamic Time Warping or Edit distance [8] – which may exploit the uniqueness and regularity of human mobility. (3) *periodical pattern mining*: finding (sub-)sequences and periodical motion patterns, enabling travel recommendation [6, 21], life pattern understanding [23, 30, 36], recovering trajectories associated with users [13, 39] and next location prediction [9, 14, 23]. The numerous works in semantic trajectory mining have not formally defined (and addressed) the *trajectory context learning (TCL)*, which is part of our study. Our proposed CATHI is also different from conventional approaches in that it captures higher level of semantics and provides more comprehensive understanding of human mobility by learning long-short term dependency at the trajectory level.

5 CONCLUSIONS

We introduced a novel problem – Trajectory Context Learning (TCL) – aiming at mining of high-level human motion patterns. Towards its solution, we relied on Trajectory Context Embedding and proposed CATHI – a method for jointly learning hierarchical and sequential patterns of trajectories, beneficial for many downstream tasks needing inference of human trajectories. As demonstrated, TCL provides valuable insights and promising guidelines for further investigations of intricacies of motion patterns. As part of our future work we plan to investigate how to leverage it for modeling the topics of sequential patterns, and its use in a wider range of applications, such as POI recommendation and road planning.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China (Grant No.61602097 and No.61472064), NSF grants III 1213038 and CNS 1646107, and ONR grant N00014-14-10215.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

[2] Hareesh Bahuleyan, Lili Mou, Olga Vechtomova, and Pascal Poupart. 2018. Variational Attention for Sequence-to-Sequence Models. In *COLING*.

[3] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *CoNLL*.

[4] D Brockmann, L Hufnagel, T Geisel, and Roger M Whitaker. 2006. The scaling laws of human travel. *Nature* (2006).

[5] Ji Li Zhipeng Cai, Mingyuan Yan, and Yingshu Li. 2016. Sequential Short-Text Classification with Recurrent and Convolutional Neural Networks. In *NAACL-HLT*.

[6] Dawei Chen, Cheng Soon Ong, and Lexing Xie. 2016. Learning Points and Routes to Recommend Trajectories. In *CIKM*.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[8] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn J. Keogh. 2008. Querying and mining of time series data: experimental comparison of representations and distance measures. In *PVLDB*.

[9] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. DeepMove: Predicting Human Mobility with Attentional Recurrent Networks. In *WWW*.

[10] Shanshan Feng, Gao Cong, Bo An, and Yeow Meng Chee. 2017. POI2Vec: Geographical Latent Representation for Predicting Future Visitors. *AAAI* (2017).

[11] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized Ranking Metric Embedding for Next New POI Recommendation. In *IJCAI*.

[12] Qiang Gao, Goce Trajcevski, Fan Zhou, Kunpeng Zhang, Ting Zhong, and Fengli Zhang. 2018. Trajectory-based Social Circle Inference. In *SIGSPATIAL*.

[13] Qiang Gao, Fan Zhou, Kunpeng Zhang, and Goce Trajcevski. 2017. Identifying Human Mobility via Trajectory Embeddings. In *IJCAI*.

[14] Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Zhong Ting, and Fengli Zhang. 2019. Predicting Human Mobility via Variational Attention. In *WWW*.

[15] Marta C González, César A Hidalgo, Albert-László Barabási, A. L. BarabÃąsi, and Wang. 2008. Understanding individual human mobility patterns. *Nature* (2008).

[16] Jing He, Xin Li, and Lejian Liao. 2017. Category-aware Next Point-of-Interest Recommendation via Listwise Bayesian Personalized Ranking.. In *IJCAI*.

[17] Jing He, Xin Li, Lejian Liao, Dandan Song, and William K Cheung. 2016. Inferring a Personalized Next Point-of-Interest Recommendation Model with Latent Behavior Patterns. In *AAAI*.

[18] Sepp Hochreiter and JÃijrgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[19] Diederik P Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes.. In *ICLR*.

[20] Xutao Li, Gao Cong, Xiao Li Li, Tuan Anh Nguyen Pham, and Shonali Krishnaswamy. 2015. Rank-GeoFM: A Ranking based Geographical Factorization Method for Point of Interest Recommendation. In *SIGIR*.

[21] Kwan Hui Lim, Jeffrey Chan, Shanika Karunasekera, and Christopher Leckie. 2017. Personalized Itinerary Recommendation with Queuing Time Awareness. *SIGIR* (2017).

[22] Kwan Hui Lim, Jeffrey Chan, Christopher Leckie, and Shanika Karunasekera. 2015. Personalized tour recommendation based on user interests and points of interest visit durations. In *IJCAI*.

[23] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: a recurrent model with spatial and temporal contexts. In *AAAI*.

[24] Xin Liu, Yong Liu, and Xiaoli Li. 2016. Exploring the Context of Locations for Personalized Location Recommendations. *IJCAI* (2016).

[25] Yiding Liu, Tuan-Anh Pham, Gao Cong, and Quan Yuan. 2017. An Experimental Evaluation of Point-of-interest Recommendation in Location-based Social Networks. *PVLDB* (2017).

[26] Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. 2018. A Contextual Attention Recurrent Architecture for Context-Aware Venue Recommendation. In *SIGIR*.

[27] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[28] Roman Schlegel, Chi-Yin Chow, Qiong Huang, and Duncan S. Wong. 2017. Privacy-Preserving Location Sharing Services for Social Networks. *IEEE Trans. Services Computing* 10, 5 (2017), 811–825.

[29] Chaoming Song, Tal Koren, Pu Wang, and Albert-László Barabási. 2010. Modelling the scaling properties of human mobility. *Nature Physics* (2010).

[30] Hongjian Wang and Zhenhui Li. 2017. Region representation learning via mobility flow. In *CIKM*.

[31] Fei Wu and Zhenhui Li. 2016. Where did you go: Personalized annotation of mobility records. In *CIKM*.

[32] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. 2017. Bridging Collaborative Filtering and Semi-Supervised Learning: A Neural Approach for POI Recommendation. In *SIGKDD*.

[33] Cheng Yang, Maosong Sun, Wayne Xin Zhao, Zhiyuan Liu, and Edward Y Chang. 2017. A Neural Network Approach to Jointly Modeling Social Networks and Mobile Trajectories. *TOIS* (2017).

[34] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu. 2015. Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2015), 129–142.

[35] Chao Zhang, Jiawei Han, Lidan Shou, Jiajun Lu, and Thomas La Porta. 2014. Splitter: Mining Fine-Grained Sequential Patterns in Semantic Trajectories. *PVLDB* (2014).

[36] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014. Sequential click prediction for sponsored search with recurrent neural networks. In *AAAI*.

[37] Shenglin Zhao, Tong Zhao, Irwin King, and Michael R Lyu. 2017. Geo-Teaser: Geo-Temporal Sequential Embedding Rank for Point-of-interest Recommendation. In *WWW*.

[38] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei Ying Ma. 2009. Mining interesting locations and travel sequences from GPS trajectories. In *WWW*.

[39] Fan Zhou, Qiang Gao, Kunpeng Zhang, Goce Trajcevski, Zhong Ting, and Fengli Zhang. 2018. Trajectory-User Linking via Variational AutoEncoder. In *IJCAI*.

[40] Fan Zhou, Bangying Wu, Yi Yang, Goce Trajcevski, Kunpeng Zhang, and Ting Zhong. 2018. vec2Link: Unifying Heterogeneous Data for Social Link Prediction. In *CIKM*.

Variational Session-based Recommendation Using Normalizing Flows

Fan Zhou
University of Electronic Science and Technology of China, Chengdu, China
fan.zhou@uestc.edu.cn

Zijing Wen
University of Electronic Science and Technology of China, Chengdu, China
wenzijing93@gmail.com

Kunpeng Zhang
University of Maryland, College park
kpzhang@umd.edu

Goce Trajcevski
Iowa State University, Ames
gocet25@iastate.edu

Ting Zhong*
University of Electronic Science and Technology of China, Chengdu, China
zhongting@uestc.edu.cn

ABSTRACT

We present a novel generative Session-Based Recommendation (SBR) framework, called VAriational SEssion-based Recommendation (VASER) – a non-linear probabilistic methodology allowing Bayesian inference for flexible parameter estimation of sequential recommendations. Instead of directly applying extended Variational AutoEncoders (VAE) to SBR, the proposed method introduces normalizing flows to estimate the probabilistic posterior, which is more effective than the agnostic presumed prior approximation used in existing deep generative recommendation approaches. VASER explores soft attention mechanism to upweight the important clicks in a session. We empirically demonstrate that the proposed model significantly outperforms several state-of-the-art baselines, including the recently-proposed RNN/VAE-based approaches on real-world datasets.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → *Neural networks.*

KEYWORDS

Session-based recommendation, variational autoencoders, normalizing flows

ACM Reference Format:
Fan Zhou, Zijing Wen, Kunpeng Zhang, Goce Trajcevski, and Ting Zhong. 2019. Variational Session-based Recommendation Using Normalizing Flows. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313615

1 INTRODUCTION

Session-based recommendation (SBR) [18, 36] aims at predicting user's next action based on a series of recent actions. It is a kind of sequence learning/recommendation task where longer-term user historical activities are usually unavailable and the recommendations need to be made in accordance with an assumed short-term interests of the (anonymous) user.It has been widely used in various applications, such as news recommendation, e-commerce, video and classified advertisement recommendation [10, 36].

Recent advances in deep learning have spurred the use of recurrent neural networks (RNNs) based methods to model SBR [17, 18, 28, 33], achieving significant improvement on recommendation accuracy over traditional sequence-based models such as factorizing personalized Markov chains (FPMC) [16, 38] and feature-based matrix factorization (MF) [5, 51]. Specifically, GRU4Rec [18] – a first application of augmented gated recurrent units (GRUs) [8] – was developed to address SBR by encoding user's preference and learning it for next-click prediction. Subsequently, a few improvements to GRU4Rec have been proposed – e.g., incorporating attention mechanism [28]; employing hierarchical recurrent networks [28]; augmenting data with additional features associated with items [19]; prioritizing short attention/memory [33]; and introducing more sophisticated ranking algorithms [17].

Complementary to these works, recent efforts on incorporating stochastic latent variables trained by deep generative models (e.g., variational autoencoders (VAE) [25, 40]) have enabled significant progress in many natural language processing tasks (e.g., dialogue generation and machine translation [2, 3, 11, 14, 21]). Similarly, Various generative models including VAEs have demonstrated potential for learning effective non-linear representations of user-item interactions [7, 23, 27, 30, 32] in collaborative filtering settings. For the most part, they either model the generation process of auxiliary information (e.g., content and ratings) [7, 27, 30] or build a probabilistic latent-variable framework that shares statistical strength among users and items [7, 23, 32].

Despite the improvements over conventional item recommendation, the aforementioned Bayesian models cannot be directly generalized to SBR due to the following reasons:
(1) **Data availability**: the lack of users' profile information and long-term interaction data makes these models not work well in SBR settings.
(2) **Bypassing issue**: autoregressive models (e.g., LSTM [20] and GRU [8]) combined with the soft attention mechanisms [1] have capabilities of reconstructing an encoded session on their own. This

*Corresponding author: zhongting@uestc.edu.cn

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313615

may weaken the effects of the incorporated latent factors [3], which can potentially reduce the performance of the VAE-based models. (3) **Biased inference**: VAE based models usually assume a predefined prior for latent factors [24], e.g., multivariate Gaussian which might result in the inferred approximate posterior greatly deviating from the true distribution.

We extend VAEs to model implicit feedbacks of user-item interactions in a session, and present the *VAriational SEssion-based Recommendation (VASER)*. While retaining the *Bayesian inference* of VAEs and enabling exploration of *non-linear* probabilistic latent-variable models, the VASER model: (1) effectively addresses the problem of unimodal and simple parametric problems of existing SBR methods; and (2) largely ameliorates the bias inference problem of existing VAE based recommendation methods. Specifically, we make the following contributions:

- VASER augments the RNNs based SBR models with stochastic latent variables, enabling stable and effective approximate inference of a high-level "objective" of an entire session from the observed clicks.
- We exploit the flows to approximate the real posterior of stochastic latent factors, which can largely alleviate the inference bias in existing VAE based recommendation models and improve the next click prediction accuracy.
- We demonstrate that VASER achieves improvements in SBR performance over the baselines on several real-world datasets.

2 PRELIMINARIES

We now introduce the basic notation used in the rest of the paper and formally define the problem. Recalling that SBR aims at predicting which item an anonymous user would like to click next given his current sequence of interaction data, we also provide an overview of the RNN based approaches.

2.1 Problem Definition

Formally, we have a set of sessions $\mathbf{S}$, and each session $\mathbf{s}_i \in \mathbf{S}$, consists of a sequence of user actions (e.g., click, purchase, etc.). $\mathbf{s}_i = [x_{i,1}, \cdots, x_{i,N}]$ (interchangeably denoted by $x_{i,(1:N)}$), where $x_{i,j} \in \mathbb{R}$ ($1 \leq j \leq N$, N is the length of the session.) is an interaction with item j in the session, assumed to be mapped to the domain $\mathbb{R}$. When no ambiguity arises, we will omit the index of the session – thus, given the prefix $\mathbf{s}' = [x_1, \cdots, x_{N-1}]$ of a session $\mathbf{s}$, the SBR model predicts the label(s) of the next action x_N by learning a classification distribution $\mathbf{y} = [\hat{y}_1, \cdots, \hat{y}_M]$ over M items, where $\hat{y}_j$ refers to a (predicted) probability or a ranking score for the N^{th} interaction with item j. Note that in practice, usually more than one recommendation is made, which is often referred to *top-k* session-based recommendation [28, 52].

2.2 SBR with RNNs

Existing RNN based models, with or without attention, train the sessions in a seq2seq manner. The main differences among them are how to decode the latent factors (or more precisely the last hidden state of the RNN) and how to embed the items. In "vanilla" GRU based models [17–19], decoding reconstructs the session and embedding is a separate layer of training. In attentive RNN-based models [30, 33], however, the encoder acts as an embedding layer

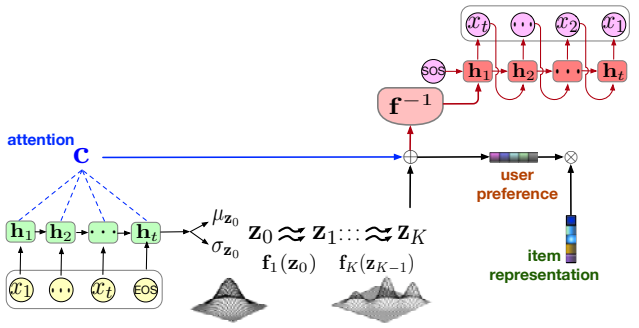


Figure 1: Overview of VASER. Recommendation is made based on the posterior $q(\mathbf{z}_K)$ of the last hidden state and the attention vector. Items are represented by embedding vectors.

– i.e., they train item embedding along with calculating loss of training sessions. Therefore, this type of *supervised* training may indeed "memorize" the sequential information of a given session, which may be "conducted" in the testing phase as the items in testing sessions would look-up the embedding matrix. As observed in the experiments in [30], this dynamic embedding method may significantly improve the performance.

An important observation is that all these works train the model in an *explicit autoregressive* fashion, i.e., they split the sessions (both training and testing) into a set of sub-sessions. Thus, a session $\mathbf{s} = [x_1, \cdots, x_N]$ would be divided into $N-1$ sub-sessions: $\mathbf{s}^1 = [x_1, \cdots, x_{N-1}]$, $\mathbf{s}^2 = [x_1, \cdots, x_{N-2}]$, $\cdots$, $\mathbf{s}^{N-1} = [x_1, x_2]$ and the original session $\mathbf{s}$ – all of which would be fed into the models for training or testing. Although not explicitly specified, this kind of autoregressive training improves the overall performance of the models, since a longer session actually contains (and thus "memorizes") the sub-sessions. We note that this autoregressive training trick has also been explored in recent CNN based SBR models [45, 52].

3 MAIN METHODOLOGIES

We propose the VASER model, as illustrated in Figure 1, consisting of two main components, namely GRU module and attention module. The GRU module captures sequential preferences, and the hidden state can exploit the non-linear preferences. The attention module is used to enhance the GRU network by dynamically selecting and linearly combining different parts of the input sequence. VASER employs a deterministic attention mechanism. VASER incorporates the normalizing flows for flexible posterior approximation. In the sequel, we present the general framework of VASER with theoretical background and training procedure.

3.1 Session Generative Model

We consider a click session generative process as follows. For each session $\mathbf{s} = [x_1, \cdots, x_N]$, the model samples $d-$dimensional latent representation from an appropriate *prior* distribution $p(\mathbf{z})$. The latent factor $\mathbf{z}$ is then transformed via a non-linear function $f_\theta(\mathbf{z})$ – a suitable likelihood function parameterized by θ – to produce a probability distribution $\pi(\mathbf{z})$ (e.g., a *multinomial* distribution) over

M candidate items, from which a session $\mathbf{s}$ is assumed to have been drawn ($\mathbf{z} \sim p(\mathbf{z}); \pi(\mathbf{z}) \propto \exp\{f_\theta(\mathbf{z})\}$):

$$\mathbf{s} \sim f_\theta(\mathbf{z}) = p_\theta(\mathbf{s}|\mathbf{z}) = \prod_{t=2}^{N} p_\theta(x_t|x_{1:t-1}, \mathbf{z}) \tag{1}$$

where $x_{1:t-1}$ indicates the prefix click sequence preceding current click x_t, and $f_\theta(\mathbf{z})$ is a deep neural networks such as a multilayer perceptron (MLP). Thus, the session generation involves making a sequence of discrete decisions, each of which samples an item from a multinomial distribution with a softmax function, to produce a probability vector $\pi(\mathbf{z})$ over the entire item set. The multinomial distribution has been demonstrated to model click data well (cf. [26, 32], although these work were originally designed for CF based recommendation).

This generative process is similar to the sentence generation in [21] and trajectory generation in [53], except that we do not take side-information (e.g., item category, click time, etc.) into account. However, it is straightforward to add additional latent factors to capture various item features, if available, for disentangling the representation.

3.2 Variational Session Inference

In general, the marginal log-likelihood of a session s $\log p_\theta(\mathbf{s}) = \log \int_{\mathbf{z}} p_\theta(\mathbf{s}|\mathbf{z})p(\mathbf{z})\mathrm{d}\mathbf{z}$ is intractable to compute or differentiate directly for flexible generative models, especially for high-dimensional latent variables. Instead, one usually resorts to variational inference by defining a simple parametric distribution over the latent variables (e.g, a factorized Gaussian) $q_\phi(\mathbf{z}|\mathbf{s})$, and maximizing the evidence lower bound (ELBO) on the marginal log-likelihood of each observation:

$$\log p_\theta(\mathbf{s}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})} \log \left[\frac{p_\theta(\mathbf{s}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{s})}\right] + \mathbb{KL}\left[q_\phi(\mathbf{z}|\mathbf{s})||p_\theta(\mathbf{z}|\mathbf{s})\right] \tag{2}$$

$$\geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}\left[\log p_\theta(\mathbf{s}, \mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{s})\right] \triangleq \mathcal{L}(\mathbf{s}; \theta, \phi) \tag{3}$$

There are numerous ways to optimize the ELBO, among which VAEs [25] use a parametric inference network and reparameterization of $q_\phi(\mathbf{z}|\mathbf{s})$ to alternatively maximize following reformulation:

$$\begin{aligned}\mathcal{L}_{\text{VAE}}(\mathbf{s}; \theta, \phi) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}[\log p_\theta(\mathbf{s}) + \log p_\theta(\mathbf{z}|\mathbf{s}) - \log q_\phi(\mathbf{z}|\mathbf{s})] \\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}[\log p_\theta(\mathbf{s})] - \mathbb{KL}\left[q_\phi(\mathbf{z}|\mathbf{s})||p_\theta(\mathbf{z}|\mathbf{s})\right] \end{aligned} \tag{4}$$

Since the first term is a constant, then the objective of maximizing ELOB $\mathcal{L}_{\text{VAE}}(\mathbf{s}; \theta, \phi))$ of $\log p_\theta(\mathbf{s})$ becomes to minimize the Kullback-Leibler (KL) divergence between $q_\phi(\mathbf{z}|\mathbf{s})$ and the true distribution $p_\theta(\mathbf{z}|\mathbf{s})$ (which is always ≥ 0). For brevity, we will sometimes omit the parameters ϕ and θ in subsequent formulae.

3.3 Inference with Normalizing Flows

It is desirable to reduce the (non-negligible) inference gaps, and various improved posterior approximations have been effective in improving variational inference. Although none of the existing methods is able to completely close the gap between approximate posterior and true posterior [6], employing richer posterior/prior distributions can effectively reduce it. The approximation gap, caused by the encoding cost $\mathbb{KL}\left[q_\phi(\mathbf{z}|\mathbf{s})||p_\theta(\mathbf{z}|\mathbf{s})\right]$, is largely due to the improper assumption of the probabilistic distribution [9, 24].

We leverage the flow method [39] to construct more accurate posterior approximation of the session distributions, rather than simple Gaussian assumption in existing works [4, 49]. Normalizing Flows (NF) [39] is a powerful framework for building flexible posterior distributions through an iterative procedure. The main idea is to transform a simple distribution into a complex one through a series of invertible mappings which, in theory, can approximate any complex distribution. Given a variable $\mathbf{z}_0$ with known probability distribution $p_0(\mathbf{z}_0)$ (e.g., Gaussian here) and a chain of invertible transformations $\mathbf{f} = [\mathbf{f}_1, \cdots, \mathbf{f}_K]$, then $\mathbf{z}_k$ can be calculated by composing the transformations from $\mathbf{f}$ as:

$$\mathbf{z}_K = \mathbf{f}_K(\mathbf{z}_{K-1}) = \mathbf{f}_K(\mathbf{f}_{K-1}(\mathbf{z}_{K-2})) = \mathbf{f}_K(\mathbf{f}_{K-1}(\cdots \mathbf{f}_1(\mathbf{z}_0))) \tag{5}$$

Given that each $\mathbf{f}_k \in \mathbf{f}$ is invertible (i.e., $\mathbf{z}_{k-1} = \mathbf{f}_k^{-1}(\mathbf{z}_k)$), and according to the definition of probability $\int p_k(\mathbf{z}_k)d\mathbf{z}_k = \int p_{k-1}(\mathbf{z}_{k-1})d\mathbf{z}_{k-1} = 1$, for a collection of variables $\mathbf{z} = [\mathbf{z}_0, \ldots, \mathbf{z}_K]$, we can obtain the distributions $p_K(\mathbf{z}_K)$ more flexibly:

$$p_K(\mathbf{z}_K) = p_0(\mathbf{z}_0)\left|\det \frac{d\mathbf{z}_K}{d\mathbf{z}_0}\right|^{-1} \tag{6}$$

where $\det \frac{d\mathbf{f}}{d\mathbf{z}}$ is the Jacobian determinant of $\mathbf{f}$.

The path traversed by the random variables $\mathbf{z}_k = \mathbf{f}_k(\mathbf{z}_{k-1})$ with initial distribution $p_0(\mathbf{z}_0)$ is called the *flow*, and the whole path formed by the successive distributions $p_K(\mathbf{z}_K)$ refers to a *normalizing flow*. To ensure Eq.(6) is tractable, it should satisfy that (a) the transformation $\mathbf{f}_k$ must be easy to invert, and (2) the determinant of its Jacobian is easy to compute [39]. The two constraints allow the transformation to be made deeper by composing multiple instances of it, and the result will still be a valid normalizing flow. Now the log-likelihood of approximate posterior $q_K(\mathbf{z}_K|\mathbf{s})$ can be computed iteratively by using the log on both sides of Eq.(6)

$$\log q_K(\mathbf{z}_K|\mathbf{s}) = \log q_0(\mathbf{z}_0|\mathbf{s}) - \sum_{k=1}^{K} \log \det \left|\frac{d\mathbf{z}_k}{d\mathbf{z}_{k-1}}\right| \tag{7}$$

where the base distribution $\mathbf{z}_0 \sim q_\phi(\mathbf{z}_0|\mathbf{s})$ is a Gaussian in our implementation.

One of the flow transformations is the *planar flow* introduced in [39], given by:

$$\mathbf{f}(\mathbf{z}) = \mathbf{z} + \mathbf{u}\sigma(\mathbf{w}^\intercal \mathbf{z} + b) \tag{8}$$

where $\mathbf{u}, \mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ are parameters, and σ is a suitable smooth non-linear activation function (e.g., tanh). According to the *Matrix determinant lemma*, the Jacobian of this transformation is:

$$\left|\det \frac{\partial \mathbf{f}}{\partial \mathbf{z}}\right| = \left|1 + \mathbf{u}^\intercal \sigma'(\mathbf{w}^\intercal \mathbf{z} + b)\mathbf{w}\right|,$$

where σ' is the derivative activation and can be computed in $O(d)$ time – d is the dimension of $\mathbf{z}$.

In this paper, we use the planar flow as the invertible transformation for its simplicity and efficiency. We parameterize the approximation posterior $q_\phi(\mathbf{z}|\mathbf{s})$ with a flow, i.e., $q_\phi(\mathbf{z}|\mathbf{s}) := q(\mathbf{z}_K)$, the ELBO of Eq.(2) can be modified as

$$\begin{aligned}\mathcal{L}(\mathbf{s}; \theta, \phi) =& \mathbb{E}_{q(\mathbf{z}_0)}\left[\log p_\theta(\mathbf{s}|\mathbf{z}_K)\right] - \mathbb{E}_{q(\mathbf{z}_0)}\left[\log q(\mathbf{z}_0)\right] + \beta \mathbb{E}_{q(\mathbf{z}_0)}\left[\log p_\theta(\mathbf{z}_K)\right] \\ &+ \beta \mathbb{E}_{q(\mathbf{z}_0)}\left[\sum_{k=1}^{K} \log \det \left|\frac{d\mathbf{z}_k}{d\mathbf{z}_{k-1}}\right|^{-1}\right] \end{aligned} \tag{9}$$

where the first term is trained to reconstruct the sessions; the second term is a constant; and the last two terms are the flows. The coefficient β is a regularizer of the flows, which is very similar to the annealing factor for regularizing KL-divergence [3].

4 EXPERIMENTAL OBSERVATIONS

We now describe the experimental settings and report the empirical evaluation results.

Table 1: Statistics of the datasets.

Datasets	YOOCHOOSE 1/64	YOOCHOOSE 1/4
#clicks	557,248	832,6407
#train sessions	355,385	621,6184
#test sessions	52,956	56,616
#items	17,626	30,903
avg. session length	6.27	5.83

4.1 Datasets

For fair comparison, we evaluate different methods on a real-world transaction datasets YOOCHOOSE[1], which has been widely used for evaluating SBR approaches. Following previous works [28, 33, 44], we preprocess the primary data as follows: (1) We filter out sessions of length 1 and items that appear less than 5 times for the two datasets; (2) We respectively use the sessions of subsequent day for testing, and then filter out clicks from the test set where the clicked items did not appear in the training set; and (3) We sort the training sequences by time and train all models on more recent fractions (i.e., 1/64 and 1/4) of training sessions. Table 1 shows the statistics of the datasets.

4.2 Baselines

To demonstrate the effectiveness of our model, we conduct extensive comparisons to the following state-of-the-art methods:

- **Item-KNN** [42]: It is an item-to-item model that recommends items that are similar to previously visited items based on cosine similarity.
- **GRU4Rec**[2] [18]: It is an RNN-based deep learning model for session-based recommendation. It employs GRU units to capture sequential patterns and utilizes session-parallel minibatching trick and ranking-based loss functions during the training.
- **NARM**[3] [28]: It is an RNN-based model employing (deterministic) attention mechanism to capture main purpose from the hidden states and combines it with the sequential behavior as the final representation to generate recommendations.
- **STAMP** [33]: It is a priority model which captures users' general interests from the long-term memory of a session context, and current interests from the short-term memory of recent clicks.
- **ReLaVaR** [4]: It is a Bayesian version of GRU4Rec which treats the network recurrent units as stochastic latent variables with some prior distributions and infers the corresponding posteriors for prediction and recommendation generation. This is an item-level variational inference based SBR method which uses independent Gaussian as the prior for items.
- **VRM** [49]: It is a recent proposed method directly applying VAE on session-based recommendation. Unlike ReLaVaR, an item-level variational method, VRM models the stochastic inference on the session-level.

[1]http://2015.recsyschallenge.com/challenge.html
[2]https://github.com/hidasib/GRU4Rec
[3]https://github.com/lijingsdu/sessionRec_NARM

4.3 Metrics

Following previous works [16, 17, 28, 33], the primary evaluation metric is Recall@20 – i.e., the proportion of cases having the desired item falling into the top-20 predicted items in all test cases. Note that the Recall score is equal to the Hit-Precision score used in [33]. The second metric is MRR@20 (Mean Reciprocal Rank) – i.e., the average of reciprocal ranks of the desired items. The reciprocal rank is set to zero if the rank is lower than top-20. MRR takes into account the rank of the item, which is important when the order of recommendations matters. Note that the higher the Recall@20 and MRR@20, the better the performance.

4.4 Settings

For all methods, the embedding size of items is set to 50. The number of hidden units in GRU layer is set to 100. All models are trained with Adam and the mini-batch size is fixed at 512. Following [28, 33], we truncated BPTT using a fixed window of 30 time-steps for the two YOOCHOOSE datasets. Also following [28, 33], 10% of the training data are used as the validation set. For the VASER model, parameters d, K and β are respectively 100, 16 and 0.2, if not specified.

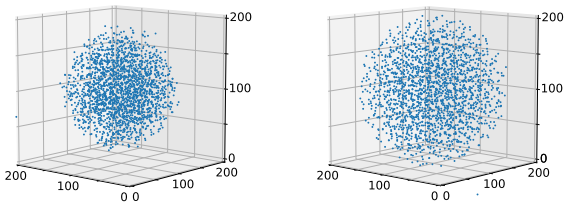
4.5 Overall Performance

Table 2: Performance comparison among all session-based recommendation methods over two datasets.

	YOOCHOOSE1/64		YOOCHOOSE1/4	
	Recall@20(%)	MRR@20(%)	Recall@20(%)	MRR@20(%)
item-KNN	53.12	22.13	52.43	21.75
GRU4REC	62.40	25.36	59.58	22.62
NARM	70.13	29.38	69.75	29.30
STAMP	70.21	29.22	70.45	29.47
ReLaVaR	64.32	25.26	60.53	22.76
CRM	69.32	28.75	68.22	28.35
VASER	71.85	30.05	70.74	29.75

4.5.1 Comparison against SBR baselines (Q1). : Table 2 shows the results of comparison to the existing state-of-the-art SBR methods, from which we can clearly observe that the proposed model perform the best on two metrics throughout two datasets.

Overall, the RNN based methods, including ours, consistently outperform the traditional baselines, which demonstrates that autoregressive models are good at learning sequential user click behaviors.



(a) NARM. (b) VASER.

Figure 2: Visualization of the encoding space obtained via NARM and VASER on YOOCHOOSE 1/64. For better viewing, we randomly select 2,048 test sessions and plot the encoding space using t-SNE.

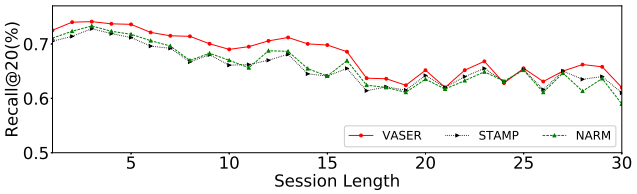


Figure 3: Impact of session length (YOOCHOOSE 1/64).

Nevertheless, RNN models alone cannot deal with complicate user-click sessions which usually have unintended clicks and/or contain one or more browse themes. This problem can be largely overcome by incorporating the attention mechanism in recent methods like NARM and STAMP. The most recent work GRU4Rec++ does not exhibit expected results on the two datasets, regardless that it can improve their original method (GRU4Rec) with the sampling trick. This result also proves one of our motivations that autoregressive models are constrained with their capability of modeling sparse and high-dimensional data.

By modeling session generation in a probabilistic generative latent variable framework, our model outperform the best baseline (either NARM or STAMP) by a significant margin. Note that in our reimplementation, the two baselines (NARM or STAMP) exhibit higher scores than their original reporting on two YOOCHOOSE datasets.

The benefit of VASER can be visualized in Figure 2, where the encoding space of NARM and VASER is plotted with t-SNE. Recall that both methods predict the last item x_N based on the learned representation of prefix session $\mathbf{s}' = [x_1, \cdots, x_{N-1}]$. The main difference is that the encoding space of NARM is the concatenation of deterministic attention $\mathbf{c}$ and the last hidden state $\mathbf{h}_{N-1}$ of GRU, while the encoding space of VASER is the combination of the posterior of hidden state $q(\mathbf{z}_K)$ and the posterior of attention $q(\mathbf{c}_k)$. Apparently, VASER explores more space for encoding the sessions and exhibits more scattered distribution. The benefits of such encoding can be understood intuitively, i.e., the more inseparable the sessions, the more difficult for the models to discriminate the spatial adjacent ones, which, consequently, are more prone to making wrong predictions.

We also investigated the impact of session length on the recommendation performance. Intuitively, the longer the sessions, the worse the prediction performance on average. The results on YOOCHOOSE 1/64 are shown in Figure 3 (results on YOOCHOOSE 1/4 are consistent, but omitted due to the lack of space), whereby we compared to the two best baselines. Our model slightly improve the recommendation performance over the baselines. However, we argue that due to the vanishing gradient problem of autoregressive model, it is hard for RNNs-based methods to further improve the performance on modeling extremely long-term dependencies.

4.5.2 Effect of Components in VASER (Q2). By comparing to the methods directly applying VAE on SBR, both item-level and session-level, we can clearly see the importance of the flow based posterior approximation used in VASER.

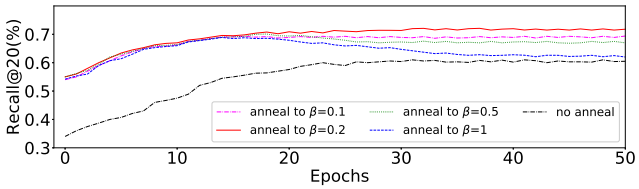
Another important observation is that directly applying VAE on modeling items or sessions is not competitive. ReLaVaR, operating stochastic inference on item-level, is less effective than VRM, which models sessions in a variational seq2seq manner. Note that we omit comparison with CVRM, a variant of VRM, which takes the category information into account, due to the *extremely* sparse category labels on the YOOCHOOSE dataset. In fact, the category information plays a less important role in improving the recommendation performance according to the results in [49]. Although allowing Bayesian inference, the two models may incur larger inference gaps and underfitting problem due to the amortized inference alone used for posterior approximation [9]. This is in accordance with the findings in modeling language with vanilla VAEs [3, 21], i.e., the autoregressive models are powerful enough to decode the entire sequence, resulting in uselessness of stochastic latent factors. More importantly, these methods approximate an improper assumed distribution $q_\phi(\mathbf{z}|\mathbf{s})$, e.g., the choice of diagonal-covariance Gaussian in [3, 21], and thus are subjected to heavy bias inference problem, as explained in Sec. 3.2. In contrast, our VASER model can largely alleviate this problem benefiting from the normalizing flows with flexible posterior approximation.

4.6 Impact of Parameters (Q3)

There are two important factors affecting the performance of the VASER model, i.e., the coefficient β regularizing the flows and the determinant of Jacobian matrix, and K, the number of invertible transformations.

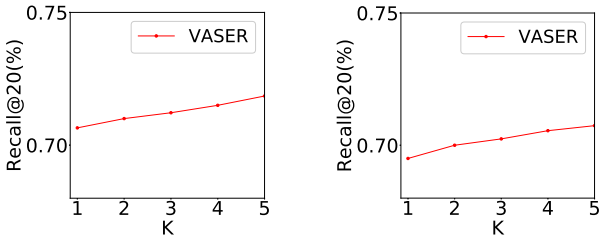
Figure 4 shows the impact of β on VASER, where β is gradually annealed to the value of 0.1, 0.2, 0.5 and 1. We observe in our experiments that the flow terms are usually ordered larger than the reconstruction term. Without annealing or annealing to a larger value, the performance of VASER model are not appealing, and even experience overfitting problem. On the contrary, if the value of β is too small (e.g., below 0.2, the flows does not take effect as the decoder RNN will make the model converge, when the model rely less on the latent factors. As a consequence, there is a significant performance decline. As we explained earlier, this is caused by the overpower performance of RNN decoder. In addition to cost annealing, another possible way of alleviating this problem is to replace RNN with the dilated CNN suggested by [21].

Figure 5 investigates the impact of K on two datasets. The planar flows used in VASER modifies the initial density by applying a series



(a) YOOCHOOSE 1/64.

Figure 4: The impact of β.



(a) YOOCHOOSE 1/64. (b) YOOCHOOSE 1/4.

Figure 5: The impact of K.

of contractions and expansions in the original space. Although, in theory, more transformations could approximate more complicated distribution, a smaller value is enough for the model. Since the RHS of Eq.(8) can be interpreted as a *single-neuron* MLP, it may result in the information going through a single bottleneck. As the volume of the space grows exponentially with the number of dimensions d, it requires many coupling layers to transform a simple base distribution into a complex one [24]. This is demonstrated by the results on YOOCHOOSE 1/64 dataset in Figure 5(a), where the model require more transformations to obtain higher performance.

5 RELATED WORK

5.1 Sequential Recommendation

SBR is essentially a sequence learning problem including typical scenarios such as click/purchase recommendation in e-commerce, music/video recommendation, news items etc. Since only short-term interaction data are available and there is lack of user profile, CF based latent factor models fail to work in these scenarios. Non-parametric methods, such as k-Nearest Neighbor (KNN) and context tree can be used to estimate the user/item similarity for recommending the most similar items to the ones that have been visited/clicked by a user [12, 15, 22, 35, 42]. Naturally, other sophisticated sequence learning approaches can also be adapted to solve the session-based recommendation problem, which incurs MC based models [48] and hybrid CF models like FPMF [16, 38], etc.

RNN models have been successfully applied in many sequence learning tasks, such as machine translation [1], human mobility learning [13], and session-based recommendation [17, 18, 28, 31, 33, 37]. GRU4Rec [18] is a representative RNN based method for SBR, which embeds the clicks into the final hidden state of GRU to represent the current preference. This method has achieved significant improvement against previous sequence learning approaches like FPMC and item similarity based KNN. Several works have been proposed to improve GRU4Rec with various models. For example, NARM and EDRec [28, 34] employ soft attention mechanism [1] to capture the user's main purpose in the current session, which is combined with the last hidden state of GRU to compute the recommendation scores for each candidate item. Hidasi et al. [17] improved their GRU4Rec model by introducing tailored ranking loss functions.

5.2 Deep Generative Recommendation

Although there exist many deep recommendation models as mentioned above, relatively few works in the literature focus on applying generative models in the recommendation systems. Previous autoencoder based models [29, 41, 43, 46, 47, 50] show promising performance but are restricted to learning representation of items, and thus are difficult for Bayesian inference due to lack of Bayesian nature or high computational cost. Several recent works extend the ideas of applying VAEs to CF-based recommendation but mainly focus on combining various auxiliary features [7, 23]. The most related work are ReLaVaR [4] and VRM/CVRM [49], both of which apply VAE on the SBR tasks. ReLaVaR is an item-level stochastic inference method while VRM/CVRM are modeling session in a stochastic seq2seq manner. As sequential VAE models, they can be considered as directly applying VAEs in the SBR scenario.

Compared to existing sequential recommendation approaches, we model the problem within a probabilistic recommendation setting which allows our model for Bayesian inference. In addition, we derive novel flow-based model tailored for SBR task with the flexible posterior approximation, rather than presumed Gaussian distribution in previous work.

6 CONCLUSIONS AND FUTURE WORK

We presented VASER, a flow based generative framework for learning sequential click patterns. A distinct feature of the proposed model implementing VASER is that it enables learning non-linear interactions between user-clicks while allowing Bayesian inference. As demonstrated by the experiments, VASER achieves significant improvements for the session-based recommendation problem in comparison to existing methods. One of the most important implications of the results from this work is that instead of using amortized inference as in existing collaborative/sequential variational recommendation methods, flow based techniques could effectively improve the density approximation and deserves more attention in the recommendation community. In our future work, we are planning to focus on augmenting VASER to consider auxiliary information – e.g., coupling sequential information with other related contexts (category, price and click time), and on tackling the overall efficiency (e.g., by incorporating CNNs).

ACKNOWLEDGEMENTS

This work was supported by National Natural Science Foundation of China (Grants No.61602097 and No.61472064), NSF Grants III 1213038 and CNS 1646107, and an ONR grant N00014-14-10215.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.

[2] Hareesh Bahuleyan, Lili Mou, Olga Vechtomova, and Pascal Poupart. 2018. Variational Attention for Sequence-to-Sequence Models. In *COLING*.

[3] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. In *CoNLL*.

[4] Sotirios P Chatzis, Panayiotis Christodoulou, and Andreas S Andreou. 2017. Recurrent Latent Variable Networks for Session-Based Recommendation. In *DLRS@RecSys*.

[5] Tianqi Chen, Weinan Zhang, Qiuxia Lu, Kailong Chen, Zhao Zheng, and Yong Yu. 2012. SVDFeature: a toolkit for feature-based collaborative filtering. *JMLR* (2012).

[6] Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2017. Variational Lossy Autoencoder. In *ICLR*.

[7] Yifan Chen and Maarten de Rijke. 2018. A Collective Variational Autoencoder for Top-N Recommendation with Side Information. In *DLRS@RecSys*.

[8] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555* (2014).

[9] Chris Cremer, Xuechen Li, and David K Duvenaud. 2018. Inference Suboptimality in Variational Autoencoders. In *ICML*.

[10] Gabriel de Souza Pereira Moreira, Felipe Ferreira, and Adilson Marques da Cunha. 2018. News Session-Based Recommendations using Deep Neural Networks. In *DLRS@RecSys*.

[11] Yuntian Deng, Yoon Kim, Justin Chiu, Demi Guo, and Alexander M Rush. 2018. Latent Alignment and Variational Attention. In *NIPS*.

[12] Mukund Deshpande and George Karypis. 2004. Item-based top-N recommendation algorithms. *ACM TOIS* (2004).

[13] Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Xucheng Luo, and Fengli Zhang. 2017. Identifying Human Mobility via Trajectory Embeddings. *IJCAI* (2017).

[14] Anirudh Goyal, Alessandro Sordoni, Marc-Alexandre Côté, Nan Rosemary Ke, and Yoshua Bengio. 2017. Z-Forcing: Training Stochastic Recurrent Networks. In *NIPS*.

[15] Huifeng Guo, Ruiming Tang, Yunming Ye, Feng Liu, and Yuzhou Zhang. 2018. An Adjustable Heat Conduction based KNN Approach for Session-based Recommendation. *arXiv* (2018).

[16] Ruining He and Julian McAuley. 2016. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *ICDM*.

[17] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In *CIKM*.

[18] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR*.

[19] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. In *RecSys*.

[20] Sepp Hochreiter and JÃijrgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* (1997).

[21] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward Controlled Generation of Text. In *ICML*.

[22] Dietmar Jannach and Malte Ludewig. 2017. When Recurrent Neural Networks meet the Neighborhood for Session-Based Recommendation. In *RecSys*.

[23] G Karamanolakis, K R Cherian, AR Narayan Proceedings of the 3rd, and 2018. 2018. Item Recommendation with Variational Autoencoders and Heterogeneous Priors. In *DLRS@RecSys*.

[24] Diederik P Kingma, Tim Salimans, Rafal Józefowicz, Xi Chen, Ilya Sutskever, and Max Welling. 2016. Improving Variational Autoencoders with Inverse Autoregressive Flow. In *NIPS*.

[25] Diederik P Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *ICLR*.

[26] Rahul G Krishnan, Dawen Liang, and Matthew D Hoffman. 2018. On the challenges of learning with inference networks on sparse, high-dimensional data. In *AISTATS*.

[27] Wonsung Lee, Kyungwoo Song, and Il-Chul Moon. 2017. Augmented Variational Autoencoders for Collaborative Filtering with Auxiliary Information. In *CIKM*.

[28] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *CIKM*.

[29] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep Collaborative Filtering via Marginalized Denoising Auto-encoder. In *CIKM*.

[30] Xiaopeng Li and James She. 2017. Collaborative Variational Autoencoder for Recommender Systems. *KDD* (2017).

[31] Zhi Li, Hongke Zhao, Qi Liu, Zhenya Huang, Tao Mei, and Enhong Chen. 2018. Learning from History and Present: Next-item Recommendation via Discriminatively Exploiting User Behaviors. *KDD* (2018).

[32] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. *WWW* (2018).

[33] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation. In *KDD*.

[34] Pablo Loyola, Chen Liu, and Yu Hirate. 2017. Modeling User Session and Intent with an Attention-based Encoder-Decoder Architecture. In *RecSys*.

[35] Fei Mi and Boi Faltings. 2018. Context Tree for Adaptive Session-based Recommendation. *arXiv* (2018).

[36] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-Aware Recommender Systems. *Comput. Surveys* (2018).

[37] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks. In *RecSys*.

[38] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW*.

[39] Danilo Jimenez Rezende and Shakir Mohamed. 2015. Variational Inference with Normalizing Flows. *ICML* (2015).

[40] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *ICML*.

[41] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey E Hinton. 2007. Restricted Boltzmann machines for collaborative filtering. In *ICML*.

[42] Badrul Munir Sarwar, George Karypis, Joseph A Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*.

[43] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. AutoRec: Autoencoders Meet Collaborative Filtering. In *WWW*.

[44] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved Recurrent Neural Networks for Session-based Recommendations. In *DLRS@RecSys*.

[45] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *WSDM*.

[46] Hao Wang, Xingjian Shi, and Dit-Yan Yeung. 2016. Collaborative Recurrent Autoencoder: Recommend while Learning to Fill in the Blanks. In *NIPS*.

[47] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In *KDD*.

[48] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning Hierarchical Representation Model for NextBasket Recommendation. In *SIGIR*.

[49] Zhitao Wang, Chengyao Chen, Ke Zhang, Yu Lei, and Wenjie Li. 2018. Variational Recurrent Model for Session-based Recommendation. In *CIKM*.

[50] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. In *WSDM*.

[51] Fajie Yuan, Guibing Guo, Joemon M Jose, Long Chen, Haitao Yu, and Weinan Zhang. 2016. LambdaFM: Learning Optimal Ranking with Factorization Machines Using Lambda Surrogates. In *CIKM*.

[52] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2018. A Simple but Hard-to-Beat Baseline for Session-based Recommendations. *arXiv* (2018).

[53] Fan Zhou, Qiang Gao, Goce Trajcevski, Kunpeng Zhang, Ting Zhong, and Fengli Zhang. 2018. Trajectory-User Linking via Variational AutoEncoder. In *IJCAI*.

Improving Top-K Recommendation via Joint Collaborative Autoencoders

Ziwei Zhu, Jianling Wang, and James Caverlee
Texas A&M University
{zhuziwei,jlwang,caverlee}@tamu.edu

ABSTRACT

In this paper, we propose a Joint Collaborative Autoencoder framework that learns both user-user and item-item correlations simultaneously, leading to a more robust model and improved top-K recommendation performance. More specifically, we show how to model these user-item correlations and demonstrate the importance of careful normalization to alleviate the influence of feedback heterogeneity. Further, we adopt a pairwise hinge-based objective function to maximize the top-K precision and recall directly for top-K recommenders. Finally, a mini-batch optimization algorithm is proposed to train the proposed model. Extensive experiments on three public datasets show the effectiveness of the proposed framework over state-of-the-art non-neural and neural alternatives.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

recommender systems; Autoencoder; hinge-based loss function

ACM Reference Format:
Ziwei Zhu, Jianling Wang, and James Caverlee. 2019. Improving Top-K Recommendation via Joint Collaborative Autoencoders. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313678

1 INTRODUCTION

Since feedback like user clicks, purchases, and views is much more widespread than explicit ratings, implicit top-K recommenders offer great opportunities for far-reaching impact [6, 7, 17]. One of the most influential approaches is collaborative filtering (CF) [19], which learns the preference coherence among users (user-user CF) or the rating correlations among items (item-item CF). Recently, we have seen a series of works that aim to extend CF to new neural network frameworks that offer the promise of flexibility, non-linearity, and structural complexity, leading to more expressiveness and state-of-the-art recommendation performance [4, 5, 18, 20].

While promising, one limitation of these new neural neighborhood-based models is that they typically attempt to learn low-dimension embeddings of only users or only items, without careful consideration of the important interactions between users and items. As a result, the quality of recommendation may be restricted. In contrast, effective modeling of user-item interactions could lead to improved recommendation, as has been demonstrated in earlier non-neural approaches [16, 22, 25, 26]. We experimentally confirm this intuition in Section 3.3, where we find that while user-based and item-based Autoencoders do agree on many recommendations, there is a significant amount of disagreement between the two.

Hence, we propose in this paper a new Joint Collaborative Autoencoder (JCA) model that *jointly learns* user-user and item-item correlations simultaneously so that recommenders can take advantage of as much hidden information as possible for better recommendation quality. We show how to model these user-item interactions in JCA and demonstrate the importance of careful normalization to alleviate the influence of feedback heterogeneity (without which, naive methods may recommend items a user dislikes). Further, JCA adopts a pairwise hinge-based objective function for implicit top-K recommenders, which is especially designed for maximizing precision and recall for top-K recommendation. Through extensive experiments over three public datasets, we demonstrate how the proposed JCA model improves upon both non-neural and neural methods alike.

In sum, the contributions of this paper are as follows: (i) we propose a neural neighborhood-based CF approach called Joint Collaborative Autoencoder (JCA) that captures both user-user and item-item correlations simultaneously. (ii) We adopt a pairwise hinge-based objective function to optimize top-K precision and recall directly. (iii) We present a mini-batch optimization algorithm so that JCA can be trained on large datasets, without the (impractical) need to load the entire rating matrix as in a naive batch learning approach. (iv) Finally, through extensive experiments, we show that the proposed framework outperforms state-of-the-art neural and non-neural baselines.

2 RELATED WORK

Due to the flourishing development of neural networks, instead of focusing on the traditional non-neural algorithms [6, 7, 7, 12, 15, 17], more recent efforts have been dedicated to building neural based methods for implicit top-K recommendation. For example, Wu et al. [24] proposed a Collaborative Denoising Autoencoder (CDAE) model, which implements a neural version of the influential non-neural method SVD++ [9]. He el at. [5] presented a two-pathway architecture named NCF, ensembling a neural-based implementation of matrix factorization named Generalized Matrix Factorization (GMF) and a Multi-Layer Perceptron (MLP). Furthermore, there are many neural models for contextual recommenders [2, 3, 11, 14, 21] because neural models can be naturally extended to integrate auxiliary information. For example, Niu et al. [14] proposed a pairwise neural model integrating two GMF structures to implement BPR for

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313678

image recommendation, and considered visual, topic and geography information as contextual features. Similar to these existing neural-based recommenders, the model proposed in this paper is designed to take advantage of the flexibility, non-linearity, and structural complexity of neural architectures. Besides, we also demonstrate how to jointly learn from both user-user and item-item correlations for improved recommendation quality.

Previous works [16, 22, 25, 26] have identified the potential of combining user-based and item-based collaborative filtering models to improve recommendation. Although their motivations are similar to our proposed model, there are three main differences: (i) we focus on implicit top-K recommenders; (ii) we adopt an Autoencoder-based framework as the model foundation which is a neural-based algorithm; and (iii) we fuse the user-based and item-based models by jointly training them in a unified learning process, while previous work typically combines the two models via heuristics.

3 PRELIMINARIES

In this section, we define the implicit top-K recommendation problem, introduce the basics of an Autoencoder-based recommender, and empirically show that traditional user-based and item-based Autoencoder-based recommenders are complementary, which motivates the design of our proposed model.

3.1 Implicit top-K Recommendation

Formally, we denote the set of users as $\mathbb{U} = \{1, 2, \ldots, N\}$, the set of items as $\mathbb{I} = \{1, 2, \ldots, M\}$, and the set of historical feedback from the users to the items as $\mathbb{O} = \{u, i, r_{u,i}\}$ where u indexes one user, i indexes one item, and $r_{u,i}$ represents the rating from the user u to the item i.[1] For implicit top-K recommenders, there are usually only positive feedback signals, which means we only know which items user u likes but have no idea which items user u dislikes, i.e. all $r_{u,i}$ are 1. Based on this historical feedback, our goal is to predict the preference of each user u to the items they have not interacted with and recommend the items with the highest predicted scores to user u. Furthermore, we denote $\mathbb{O}_u^+$ as the set of items user u has given positive feedback to, and $\mathbb{O}_u^-$ denotes the set of items user u has not given feedback to. In practice, we will randomly sample a subset of the unrated items to train the model, hence in this paper we use $\mathbb{O}_u^-$ to denote the randomly selected negative samples.

3.2 Autoencoders for Recommendation

Toward tackling the implicit top-K recommendation problem, Autoencoder based models [20, 24] have shown promising performance in comparison with non-neural and other neural models. We input to an Autoencoder the rating vectors of users (or items) with 1 denoting a positive interaction, and 0 denoting a missing (or unknown) interaction. Then the Autoencoder outputs the corresponding completed rating vectors. Such an Autoencoder-based recommender called AutoRec has been proposed in [20], whose model can be written as:

$$\widehat{\mathbf{R}} = f(g(\mathbf{RV} + \mathbf{b}_1)\mathbf{W} + \mathbf{b}_2),$$

where the binary implicit feedback matrix $\mathbf{R}$ is input into AutoRec, and $\widehat{\mathbf{R}}$ is the predicted preference matrix. If we set both $f(\cdot)$ and $g(\cdot)$ as identity functions, and we also ignore the bias nodes, then AutoRec becomes: $\widehat{\mathbf{R}} = \mathbf{RVW}$, which is equivalent to the user-based neighborhood model with two factorized matrices as the similarity measurement (similar to the FISM model proposed in [7]). As a result, when we feed user rating vectors $\mathbf{R}^U \in \mathbb{R}^{N \times M}$ as the input (that is, user rating vectors are the rows), AutoRec learns the similarity among items. We refer to this model as user-based AutoRec (or U-AutoRec). In a similar fashion, we can input item rating vectors $\mathbf{R}^I \in \mathbb{R}^{M \times N}$ and learn similarity among users. We refer to this model as item-based AutoRec (or I-AutoRec).

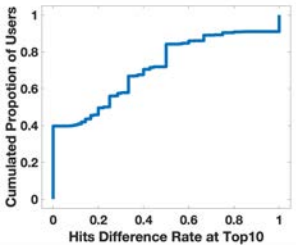


Figure 1: The CDF of the corresponding Hits Difference Rate between user-based AutoRec and item-based AutoRec.

3.3 User-based AutoRec and Item-based AutoRec Are Complementary

Sarwar showed in [19] that traditional (non-neural) user and item-based CF models are fundamentally different. Here, we investigate the difference between a neural neighborhood-based CF methods (U-AutoRec and I-AutoRec). More specifically, we explore whether different information from the same dataset captured by different models can produce complementary recommendations.

In order to answer this question, we run U-AutoRec and I-AutoRec on the same MovieLens 1M dataset (the details of this dataset and the model parameter settings are introduced in Section 5.1 and 5.2). For each user in this dataset, these two models will recommend two lists of ranked movies; we focus here on the top-10 ranked movies in the two lists. We consider the *hits* of a model as the set of movies that are in the recommended 10 movies and also in the test set, i.e. the ground truth of the user's preference. We denote the hits of U-AutoRec as $\mathbb{H}_U$ and of I-AutoRec as $\mathbb{H}_I$. Then we define the *hits difference rate* of these two model for one user as:

$$\text{HitsDifferenceRate} = \frac{size(\mathbb{H}_U \cup \mathbb{H}_I) - size(\mathbb{H}_U \cap \mathbb{H}_I)}{size(\mathbb{H}_U \cup \mathbb{H}_I)}.$$

where $size(\cdot)$ calculates the number of elements in a set, $\cap$ and $\cup$ are set intersection and union operators. The hits difference rate demonstrates the difference in correct recommendations between two models: a large difference rate indicates the two models provide complementary recommendations. The CDF of the hits difference rate for the users in MovieLens is presented in Figure 1. We can observe that more than 60% of users have a hits difference rate larger than 20%, with some users (around 30%) having a hits difference rate larger than 50%. Therefore, based on the observations from Figure 1

[1] In this paper, matrices are denoted by boldface uppercase letters like $\mathbf{A}$, and vectors are denoted by boldface lowercase letters like $\mathbf{a}$. $\mathbf{A}_i$ represents the row vector of the i-th row of matrix $\mathbf{A}$, and $(\mathbf{A}^\top)_j$ represents the row vector of the j-th column of matrix $\mathbf{A}$. We denote a user-item-rating matrix with N users and M items as $\mathbf{R} \in \mathbb{R}^{N \times M}$. $\mathbf{R}_u \in \mathbb{R}^{1 \times M}$ is the rating vector of user u, $(\mathbf{R}^\top)_i \in \mathbb{R}^{1 \times N}$ is the rating vector of item i. Besides, we use the syntax similar to Python to denote the matrix slicing operation, for example $\mathbf{A}[:, [1, 2]]$ denotes the first two columns of matrix $\mathbf{A}$.

	ML1M			Yelp			Games		
	P@10	R@10	F@10	P@10	R@10	F@10	P@10	R@10	F@10
U-AutoRec	**.2343**	.1698	.1969	.0230	.0608	.0333	.0152	.1022	.0265
I-AutoRec	.1782	.1454	.1602	.0202	.0556	.0296	.0138	.0979	.0241
UI-AutoRec	.2209	**.1801**	**.1984**	**.0240**	**.0646**	**.0350**	**.0171**	**.1192**	**.0298**

Table 1: Comparison between U-AutoRec, I-AutoRec and the average model of them – UI-AutoRec, where the best results are marked in bold. P represents precision, R represents recall, and F represent F1 score.

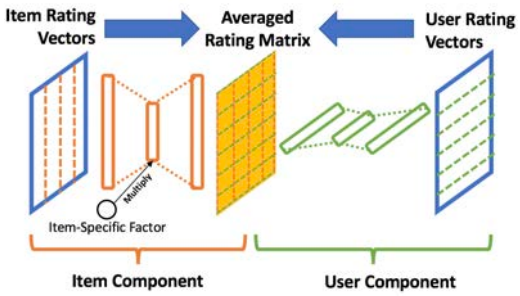


Figure 2: Overview of JCA: The Joint Collaborative Autoencoder takes both the Item Component with item rating vectors (left) and the User Component with user rating vectors (right) to recover the complete rating matrix (middle).

we can conclude that U-AutoRec and I-AutoRec generate different recommendations which are complementary to each other.

We further verify the conclusion that U-AutoRec and I-AutoRec are complementary by comparing the recommendation quality of U-AutoRec, I-AutoRec, and a simple averaging model of them (named UI-AutoRec, which averages the outputs of separately trained U-AutoRec and I-AutoRec to be the final result). In experiments over three datasets (details introduced in Section 5.1 and 5.2), we observe in Table 1 that the simple average of U-AutoRec and I-AutoRec can improve the top-10 recommendation quality, compared to either of the solo models, further indicating that U-AutoRec and I-AutoRec are complementary to each other. Therefore, combining the two models together to simultaneously explore both sides of information is promising for improved top-K recommendation.

4 PROPOSED MODEL

In this section, we present the design of a new **J**oint **C**ollaborative **A**utoencoder (**JCA**) model with a pairwise hinge-based objective function. In addition, we present a mini-batch optimization algorithm to optimize the JCA model.

4.1 Joint Collaborative Autoencoder

The intuition of JCA is to directly model both users and items simultaneously, rather than treating users and items in isolation. Further, naive combination of users and items without consideration of the heterogeneity of feedback can lead to overestimation of predicted scores in the output layer even for items a user dislikes. Hence, we describe in the following the design of the JCA structure with a special item normalization factor (to handle feedback heterogeneity), leading to the model shown in Figure 2.

Joint User-Item Correlations Modeling. Given N users and M items, the proposed JCA takes the whole rating matrix $\mathbf{R} \in \mathbb{R}^{N \times M}$ as a dual input. As illustrated in Figure 2, the rating matrix is input to the model on both the lefthand and righthand sides. The key difference is in how the matrix is sliced: the lefthand side takes as input one item rating vector $(\mathbf{R}^\top)_i \in \mathbb{R}^{1 \times N}$ as one sample (i.e. one column of $\mathbf{R}$), while the righthand side takes as input one user rating vector $\mathbf{R}_u \in \mathbb{R}^{1 \times M}$ as one sample (i.e. one row of $\mathbf{R}$). In essence, there are two components to the model, the first taking $\mathbf{R}_u$ as one input sample, which we name the *User Component*, and the other one with $(\mathbf{R}^\top)_i$ as one input sample, which we name the *Item Component*. The two components are two Autoencoder structures, which predict and recover the rating matrix independently. The first-layer weights of the user and item components are denoted as $\mathbf{V}^U \in \mathbb{R}^{M \times H^U}$ and $\mathbf{V}^I \in \mathbb{R}^{N \times H^I}$ respectively, where H^U and H^I represent the dimensions of the hidden layers of the two components. In the same way, we denote the second-layer weights as $\mathbf{W}^U \in \mathbb{R}^{H^U \times M}$ and $\mathbf{W}^I \in \mathbb{R}^{H^I \times N}$. The biases for the hidden layers and output layers are denoted as $\mathbf{b}_1^U$ and $\mathbf{b}_2^U$ for the user component, $\mathbf{b}_1^I$ and $\mathbf{b}_2^I$ for the item component. We set the sigmoid function as the activation function for hidden layers and output layers in both components.

Unlike the traditional user-based or item-based AutoRec, which inputs one or several incomplete rating vectors of a user or item and outputs the completed rating vectors, this Joint Collaborative Autoencoder takes the whole rating matrix as input, and outputs the whole completed rating matrix. The user component and item component output two completed rating matrices independently, and we combine the two outputs to be the final output as:

$$\widehat{\mathbf{R}} = \frac{1}{2}[h_{item}(\mathbf{R}) + h_{user}(\mathbf{R})], \tag{1}$$

where $\widehat{\mathbf{R}}$ is the predicted rating matrix, $h_{item}(\cdot)$ is the item component output, and $h_{user}(\cdot)$ is the user component output.

Learnable Item Normalization Factor. One challenge with traditional Autoencoder-based recommenders is that there is no normalization for the hidden layer. As a result, the heterogeneity of feedback can create problems. Take the user-based AutoRec as an example, inputting a user rating vector with more positive feedback into the AutoRec model results in a hidden layer with a larger norm, which leads to higher predicted scores in the output layer even for the items the user dislikes. A similar situation arises for the item-based AutoRec, where an item with more positive feedback has overall higher predicted scores even for the users who dislike it. Inspired by FISM [7] which adopts a user specific normalization constant, we add a learnable item normalization factor to alleviate the influence of this item feedback heterogeneity.

Specifically, we add a learnable *Item Normalization Factor* $\mathbf{f} \in \mathbb{R}^{M \times 1}$ multiplied to the hidden layer of the item component. This parameter is to normalize the norm of the latent factor for each item to solve the problem that items with more feedback will have higher predicted ratings. The user component does not need a normalization factor because the recommender ranks the predicted preference scores and recommends the top-K items for each user independently. Hence the norm of the rating vector of each user has no influence to the recommendation result.

The JCA Model. Combining the joint user-item correlations model with this learnable item normalization factor leads to the full Joint

Collaborative Autoencoder model:

$$\widehat{\mathbf{R}} = \frac{1}{2}[\sigma(\sigma(\mathbf{R}\mathbf{V}^U + \mathbf{b}_1^U)\mathbf{W}^U + \mathbf{b}_2^U) + \sigma(\sigma((\mathbf{R}^\top\mathbf{V}^I + \mathbf{b}_1^I) \circ \mathbf{f})\mathbf{W}^I + \mathbf{b}_2^I)^\top], \tag{2}$$

where $\sigma(x) = 1/(1+e^{-x})$ is the sigmoid function, $\circ$ is the element-wise product (because $\mathbf{f}$ is a column vector, $\circ$ multiplies each entry of $\mathbf{f}$ to each corresponding row of $(\mathbf{R}^\top\mathbf{V}^I + \mathbf{b}_1^I)$).

4.2 Pairwise Hinge-based Objective Function for Implicit Top-K Recommenders

We could learn the described model parameters by optimizing a traditional objective function for recommenders. Nevertheless, many of these conventional objective functions attempt to optimize MSE or AUC [17, 20, 24], which are not aligned with top-K recommendation metrics. As a result, we introduce a pairwise hinge-based objective function to directly optimize implicit top-K recommendation performance. In addition, we will show how the hinge-based objective function links to top-K precision and recall.

Inspired by [23] and [13], which proposed new objective functions to optimize position sensitive ranking loss functions such as ADG and NDCG, we adopt a pairwise hinge-based objective function to optimize top-K ranking result, shown in Equation (3):

$$\underset{\mathbf{\Theta}}{\text{minimize}} \quad \mathcal{L} = \sum_{u\in\mathbb{U}} \sum_{\substack{i\in\mathbb{O}_u^+ \\ j\in\mathbb{O}_u^-}} max(0, \widehat{r}_{u,j} - \widehat{r}_{u,i} + d) + \frac{\lambda}{2}\|\mathbf{\Theta}\|_F^2, \tag{3}$$

where $\widehat{r}_{u,i}$ and $\widehat{r}_{u,j}$ are the predicted ratings for user u to item i with positive feedback and j from negative sampling; λ is the trade-off parameter of the L2-norm regularization term; $\mathbf{\Theta}$ represents all the parameters; d is a hyper-parameter determining how large the margin between positive feedback and random negative samples can be; and $\|\cdot\|_F$ is the Frobenius norm.

Because in practice we can not use all unobserved entries as the negative samples, we set a constant margin d for the random negative samples and positive feedback to counteract the influence from random negative sampling strategy.

Analogies to Precision@k and Recall@k. *Precision@k* (*P@k*) and *Recall@k* (*R@k*) are two most commonly used metrics to evaluate implicit top-K recommenders. They are defined as:

$$P@k = |\mathbb{O}_u^k \cap \mathbb{O}_u^+|/k, \quad |\mathbb{O}_u^k \cap \mathbb{O}_u^+|/|\mathbb{O}_u^+|, \tag{4}$$

where $\mathbb{O}_u^k$ is the set of items with the top k predicted preference scores for user u, and $|\cdot|$ calculates the number of elements in a set.

If we set $k = |\mathbb{O}_u^+|$, then *Precision@k* and *Recall@k* will be the same, and we can rewrite the formula as:

$$P@k = R@k = 1 - (\sum_{i\in\mathbb{O}_u^+} \delta((\sum_{j\in\mathbb{O}_u^-} (\widehat{r}_{u,j} - \widehat{r}_{u,i})) > 0))/|\mathbb{O}_u^+|, \tag{5}$$

where $\delta(x)$ is the Heaviside function, which returns 1 if x is true, otherwise 0.

The analogy between (5) and (3) is clear. To maximize the top-K precision and recall shown in Equation (5) is equivalent to minimizing $\sum_{i\in\mathbb{O}_u^+} \delta((\sum_{j\in\mathbb{O}_u^-} (\widehat{r}_{u,j} - \widehat{r}_{u,i})) > 0)$, which is equivalent to minimizing $\sum_{i\in\mathbb{O}_u^+, j\in\mathbb{O}_u^-} max(0, \widehat{r}_{u,j} - \widehat{r}_{u,i} + d)$ in Equation (3). Here, we use the differentiable Hinge loss function to replace the non-differentiable Heaviside function.

4.3 Mini-batch Optimization Algorithm

We can use any popular optimization algorithm to train the proposed JCA. However, one challenge is that in each training iteration, the whole user-item-rating matrix is used for training, which is not practical especially for large datasets. As a result, we propose a mini-batch optimization algorithm for handling large datasets.

For the mini-batch optimization algorithm, in each training iteration, we need to feed a mini-batch consisting of n user rating vectors denoted as $\mathbf{B}^U = \mathbf{R}[\mathbf{p}, :] \in \mathbb{R}^{n\times M}$ into the user component, where $\mathbf{p}$ is a list of n randomly sampled row indexes; and another mini-batch consisting of m item rating vectors denoted as $\mathbf{B}^I = \mathbf{R}[:, \mathbf{q}]^\top \in \mathbb{R}^{m\times N}$ into the item component, where $\mathbf{q}$ is a list of m randomly sampled column indexes. The user component recovers and outputs $\widehat{\mathbf{B}}^U$, and item component outputs $\widehat{\mathbf{B}}^I$. Due to the distinct shapes of $\widehat{\mathbf{B}}^U$ and $\widehat{\mathbf{B}}^I$, we can not simply average the them as Equation (2). Nevertheless, there are $n \times m$ common entries covered by both of $\widehat{\mathbf{B}}^U$ and $\widehat{\mathbf{B}}^I$, and we denote this commonly covered part as a matrix $\mathbf{C} \in \mathbb{R}^{n\times m}$. In this way, the whole rating matrix will be decomposed into several small matrices. We can consider one small matrix as one mini-batch. Then during each training iteration, one mini-batch $\mathbf{C}$ will be used for one time of updating, where only the weights connecting the training samples in $\mathbf{C}$ will be updated. Hence, the model formulation in (2) can be rewritten in the mini-batch format:

$$\begin{aligned}
\widehat{\mathbf{C}} &= \frac{1}{2}(\widehat{\mathbf{B}}^U[:, \mathbf{q}] + \widehat{\mathbf{B}}^I[\mathbf{p}, :]),\\
\widehat{\mathbf{B}}^U &= \sigma(\sigma(\mathbf{B}^U\mathbf{V}^U + \mathbf{b}_1^U)\mathbf{W}^U + \mathbf{b}_2^U),\\
\widehat{\mathbf{B}}^I &= \sigma(\sigma(((\mathbf{B}^I)^\top\mathbf{V}^I + \mathbf{b}_1^I) \circ \mathbf{f})\mathbf{W}^I + \mathbf{b}_2^I)^\top,
\end{aligned}$$

where $\widehat{\mathbf{B}}^U$, $\widehat{\mathbf{B}}^I$, and $\widehat{\mathbf{C}}$ are the predicted user mini-batch matrix, item mini-batch matrix, and the combined predicted mini-batch matrix, respectively; $\widehat{\mathbf{B}}^U[:, \mathbf{q}]$ means the columns indexed by $\mathbf{q}$ in $\widehat{\mathbf{B}}^U$, and $\widehat{\mathbf{B}}^I[\mathbf{p}, :]$ represents the rows indexed by $\mathbf{p}$ in $\widehat{\mathbf{B}}^I$.

The objective function (3) can be rewritten as:

$$\underset{\mathbf{\Theta}}{\text{minimize}} \quad \mathcal{L} = \sum_{u\in{}^{\mathbf{C}}\mathbb{U}} \sum_{\substack{i\in{}^{\mathbf{C}}\mathbb{O}_u^+ \\ j\in{}^{\mathbf{C}}\mathbb{O}_u^-}} max(0, \widehat{r}_{u,j} - \widehat{r}_{u,i} + d) + \frac{\lambda}{2}\|\mathbf{\Theta}\|_F^2,$$

where ${}^{\mathbf{C}}\mathbb{U}$ is the users set in mini-batch matrix $\mathbf{C}$, ${}^{\mathbf{C}}\mathbb{O}_u^+$ is the items set that user u gives positive feedback to in $\mathbf{C}$, and ${}^{\mathbf{C}}\mathbb{O}_u^-$ is the negative samples of user u in $\mathbf{C}$. This approach has the benefit of only needing to load the sampled user and item rating vectors into memory rather than storing the whole rating matrix as the batch learning algorithm does.

5 EXPERIMENTS

In this section, we empirically evaluate the proposed model in comparison with state-of-the-art non-neural and neural approaches over three datasets. We aim to answer two key questions: (i) How does the proposed JCA framework perform compared with other approaches for implicit top-K recommendation? and (ii) Are the proposed Joint Collaborative Autoencoder structure JCA and the pairwise hinge-based objective function effective?

	Users	Items	Ratings	Sparsity(%)
MovieLens 1M (ML1M)	6,027	3,062	574,026	3.11
Yelp	12,705	9,245	318,314	0.271
Video Games (Games)	19,056	9,073	184,609	0.107

Table 2: Characteristics of the three datasets.

5.1 Datasets

We use three public datasets from different domains for our experiments: MovieLens 1M[2] (ML1M), Yelp[3], and Video Games[4] (customer reviews from the Video Games category on Amazon). For ML1M and Yelp, we treat ratings higher or equal to 4 as positive feedback, and consider all other ratings as missing entries. For Game, we consider all purchase behaviors as positive feedback. These strategies are widely used for evaluating implicit recommenders [7, 17, 27]. Then we iteratively remove users and items with fewer than 5 positive feedback. Finally, we split the datasets into 20% for testing and 80% for training. In the training set, we further hold 10% as a validation set. The datasets are summarized in Table 2.

5.2 Experimental Setup

Metrics. We adopt *Precision@k* and *Recall@k* averaged across all users as the evaluation metrics for personalized ranking (recall Equation (4)). We also report the *F1@k*, which is defined as:

$$F1@k = (2 \cdot P@k \cdot R@k)/(P@k + R@k).$$

Baselines. To evaluate the proposed model, we consider eight state-of-the-art methods including neural and traditional models:

MF [10]. The conventional matrix factorization model with MSE objective function and Alternating Least Squares (ALS) optimizer.

BPR [17]. BPR learns an MF model by the pairwise ranking objective function. We use Gradient Descent to optimize the objective function.

CDAE [24]. This is an Autoencoder-based model, which implements the neural version of SVD++. We adopt the sigmoid function as the activation functions in both the hidden layer and output layer, and use MSE as the objective function.

U-AutoRec [20]. This is a user-based AutoRec model, which takes user rating vectors as inputs and learns item-item correlations. It uses the MSE loss function.

I-AutoRec [20]. Similar to U-AutoRec, but takes item rating vectors as inputs and learns user-user correlations.

UI-AutoRec. This model averages the outputs of U-AutoRec and I-AutoRec.

NCF [5]. This model combines a neural version of MF model (named GMF) and a multi-layer perceptron (MLP) model. The model structure configuration is similar according to the original paper where MLP has four hidden layers with 8, 16, 32, and 64 hidden neurons, respectively.

NPR [14]. This is a neural version of BPR with two GMFs to implement the pairwise structure and the BPR objective function.

Reproducibility. All code, data, and experimental settings can be found at https://github.com/Zziwei/Joint-Collaborative-Autoencoder. We implement the proposed model based on Tensorflow [1] and use Adam [8] optimizer. We set the learning rate 0.003 for all methods.

[2]http://files.grouplens.org/datasets/movielens/ml-1m.zip

[3]https://www.yelp.com/dataset/challenge

[4]http://snap.stanford.edu/data/amazon/productGraph/categoryFiles/ratings_Video_Games.csv

	ML1M			Yelp			Games		
	@1	@5	@10	@1	@5	@10	@1	@5	@10
BPR	.0427	.1306	.1704	.0068	.0185	.0221	.0138	.0189	.0175
MF	.0477	.1427	.1837	.0148	.0307	.0333	.0255	.0294	.0265
NCF	.0515	.1454	.1870	.0147	.0316	.0343	.0246	.0319	.0281
NPR	.0486	.1427	.1806	.0110	.0267	.0295	.0158	.0221	.0198
I-AutoRec	.0472	.1249	.1602	.0139	.0278	.0296	.0198	.0266	.0241
U-AutoRec	.0526	.1541	.1968	.0148	.0311	.0333	.0217	.0290	.0265
UI-AutoRec	.0540	.1562	.1984	.0149	.0319	.0350	**.0270**	**.0340**	.0298
CDAE	**.0552**	**.1578**	**.1999**	**.0154**	**.0321**	**.0353**	.0252	.0331	**.0301**
JCA	**.0602**	**.1634**	**.2080**	**.0164**	**.0346**	**.0376**	**.0297**	**.0364**	**.0322**
%improve	8.9%	3.5%	4.1%	6.5%	7.8%	6.5%	10.0%	7.1%	7.0%

Table 3: F1@k comparison between the proposed JCA and eight state-of-the-art baselines, where the results of proposed method and the best baselines are marked in bold.

For the mini-batch size, we set 256 for single side based methods (AutoRec and CDAE), and 1,500 for other methods (both bs^U and bs^I are set to be 1500 for JCA). For the regularization trade-off parameter, we grid search the best ones over [0.5, 0.1, 0.05, 0.01, 0.005, 0.001] for each model by the validation set. For ML1M dataset: we set $\lambda = 0.001$ for JCA, 0.05 for JCA-MSE, and 0.01 for JCA-BPR; for Yelp dataset: $\lambda = 0.005$ for JCA, 0.05 for JCA-MSE, and 0.001 for JCA-BPR; for Games dataset $\lambda = 0.001$ for JCA, 0.001 for JCA-MSE, and 0.001 for JCA-BPR. Moreover, we set the margin parameter $d = 0.15$ for JCA. We grid search hidden layer dimension over [40, 80, 120, 160, 200, 240] and set 160 hidden neurons for JCA (we set the hidden dimensions the same for both user and item component, fine tuning the dimensions separately may provide better performance). We re-sample negative samples in each epoch and set the negative sampling rate 1 for all models. Other model-specific hyper-parameters of baselines are fine-tuned over validation sets.

5.3 Proposed Model vs. State-of-the-art Models

We begin by investigating the recommendation quality of JCA compared with eight baselines over three datasets. The precision and recall results are presented in Figure 3 and Figure 4. For both metrics, we see that JCA produces the best performance. Besides, we also calculate the F1-score to further evaluate the recommendation quality as shown in Table 3, where the F1 scores for the proposed method and the best baselines are marked in bold, and the improvement rates are calculated for all settings. In terms of average *F1@k* result (across three different values of k), JCA can outperform the best baseline by around 5.5% on the MovieLens dataset, around 6.9% on the Yelp dataset, and around 8% on the Games dataset. We further do a t-test on the small improvement cases (ML1M @5 and @10), where we find the p-value is smaller than 1.5% for both, showing the statistical significance of the performance improvement.

In addition, we observe that: (i) the performance improvement is larger for sparser datasets, which may be because in high sparsity scenarios, user-user correlations or item-item correlations alone are not enough, while joint learning of user-item correlations can extract more useful information; (ii) the performance improvement is larger for smaller k (k defines how many items to be recommended to users), which indicates that JCA is more competitive when limited number of items should be recommended to users.

5.4 Proposed Model and Objective Function

To demonstrate the effectiveness of the proposed JCA structure and the pairwise hinge-based objective function, we implement two variations of JCA using alternative objective functions (MSE and

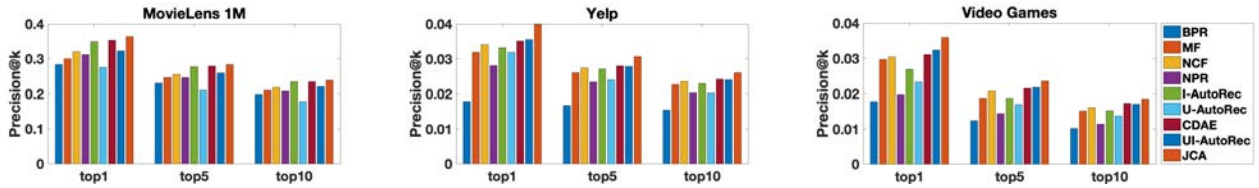


Figure 3: Precision@k results of the proposed method vs. baselines for the three datasets.

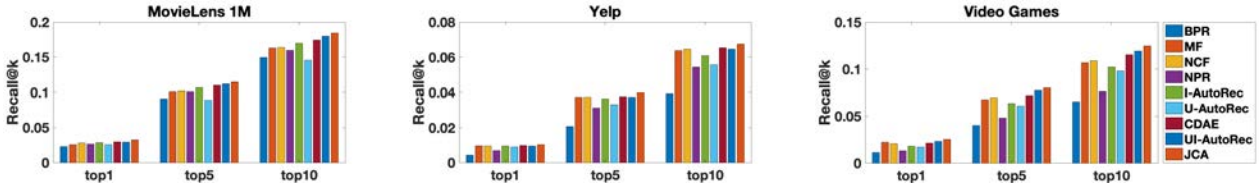


Figure 4: Recall@k results of the proposed method vs. baselines for the three datasets.

	ML1M			Yelp			Games		
	P@10	R@10	F@10	P@10	R@10	F@10	P@10	R@10	F@10
JCA-MSE	**.2346**	**.1768**	**.2017**	**.0256**	**.0665**	**.0370**	**.0178**	**.1205**	**.0311**
NCF	.2182	.1637	.1870	.0235	.0636	.0343	.0161	.1089	.0281
UI-AutoRec	.2209	**.1801**	.1984	.0240	.0646	.0350	.0171	**.1192**	.0298
CDAE	**.2344**	.1743	**.1999**	**.0242**	**.0654**	**.0353**	**.0173**	.1154	**.0301**
JCA-BPR	**.2106**	**.1597**	**.1817**	**.0234**	**.0639**	**.0343**	**.0173**	**.1160**	**.0301**
NPR	**.2078**	**.1594**	**.1806**	**.0203**	**.0543**	**.0295**	**.0114**	**.0762**	**.0198**
BPR	.1975	.1498	.1704	.0154	.0391	.0221	.0101	.0655	.0175

Table 4: Comparison between JCA models and baselines with same objective functions, where the results of the variations of JCA and the best baselines are marked in bold.

	ML1M			Yelp			Games		
	P@10	R@10	F@10	P@10	R@10	F@10	P@10	R@10	F@10
JCA	**.2384**	**.1845**	**.2080**	**.0261**	**.0674**	**.0376**	**.0185**	**.1247**	**.0322**
JCA-MSE	.2346	.1768	.2017	.0256	.0665	.0370	.0178	.1205	.0311
JCA-BPR	.2106	.1597	.1817	.0234	.0639	.0343	.0173	.1160	.0301

Table 5: Comparison between JCA models with different objective functions, where the best results are marked in bold.

BPR), and compare the performance with other models using the same objective functions. We've argued in Section 4.2 that these objective functions are not aligned with top-K recommendation, so we seek to untangle their impact. We denote the model combine JCA structure with MSE objective function as JCA-MSE, and the one with BPR loss function as JCA-BPR.

Here we report the comparison between the JCA variations (JCA-MSE and JCA-BPR) and the best three models using the MSE objective function – CDAE, UI-AutoRec and NCF – and the models using the BPR objective function – NPR and BPR. We follow the same parameter tuning process as introduced in 5.2 for the two JCA variations. We report the Precision, Recall and F1 at top 10, but note that the results for top 1 and 5 have similar patterns.

First, let's focus on the comparison between JCA variations and other models with the same objective functions to evaluate the effectiveness of the JCA structure. Table 4 shows that JCA-MSE produces better performance than CDAE, UI-AutoRec and NCF for all three datasets in terms of F1@10 and Precision@10. From the perspective of recall, there are only one exceptional case: on ML1M dataset UI-AutoRec is 1.8% better than JCA-MSE for Recall@10. For the BPR objective function, JCA-BPR outperforms NPR and BPR significantly. Based on these comparisons, we can conclude that the proposed JCA model is effective for implicit top-K recommenders, regardless of the choice of objective function.

	ML1M			Yelp			Games		
	P@10	R@10	F@10	P@10	R@10	F@10	P@10	R@10	F@10
JCA	**.2384**	**.1845**	**.2080**	**.0261**	**.0674**	**.0376**	**.0185**	**.1247**	**.0322**
JCA-NF	.2303	.1777	.2006	.0257	.0669	.0371	.0179	.1214	.0313

Table 6: Comparison between JCA models with/without the learnable item normalization factor, where the best results are marked in bold.

Next, we turn to investigate whether the pairwise hinge-based objective function is effective or not. Comparing the results of the three variations of JCA in Table 5, we can observe the recommendation performance trend: JCA > JCA-MSE > JCA-BPR for all three metrics and all three datasets. This provides empirical evidence that the proposed hinge objective function is more effective and suitable for implicit top-K recommenders.

Further, to validate the effectiveness of the learnable item normalization factor in the proposed JCA structure, we also implement a variation of JCA without the item normalization factor named JCA-NF in comparison with JCA. The result is presented in Table 6, which shows that JCA model with the item normalization factor produces better recommendation performance for all three datasets, and the performance improvement rate is around 5%.

6 CONCLUSION AND FUTURE WORK

In this paper, we show that user-based AutoRec and item-based AutoRec models explore different information from the data and produce complementary recommendations. Based on this observation, we propose the Joint Collaborative Autoencoder framework that learns both user-user and item-item correlations simultaneously, leading to improved implicit top-K recommendation quality. Besides, the proposed JCA adopts a pairwise hinge-based objective function to optimize the top-K precision and recall. We also present a mini-batch optimization algorithm so that JCA can be trained on large datasets without the (impractical) need to load the entire rating matrix as in naive batch learning approach. By extensive experiments on three public datasets, we show that the proposed framework outperforms state-of-the-art baselines. In our continuing work, we are exploring how to incorporate auxiliary information, such as textual and visual information, into the framework to further improve the recommendation quality.

Acknowledgments. This work is supported in part by NSF IIS-1841138.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

[2] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.

[3] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. 2015. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 278–288.

[4] Kostadin Georgiev and Preslav Nakov. 2013. A non-iid framework for collaborative filtering with restricted boltzmann machines. In *International Conference on Machine Learning*. 1148–1156.

[5] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[6] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee, 263–272.

[7] Santosh Kabbur, Xia Ning, and George Karypis. 2013. Fism: factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 659–667.

[8] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[9] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.

[10] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).

[11] Chenyi Lei, Dong Liu, Weiping Li, Zheng-Jun Zha, and Houqiang Li. 2016. Comparative deep learning of hybrid representations for image recommendations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2545–2553.

[12] Gai Li and Weihua Ou. 2016. Pairwise probabilistic matrix factorization for implicit feedback collaborative filtering. *Neurocomputing* 204 (2016), 17–25.

[13] Daryl Lim, Julian McAuley, and Gert Lanckriet. 2015. Top-n recommendation with missing implicit feedback. In *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 309–312.

[14] Wei Niu, James Caverlee, and Haokai Lu. 2018. Neural Personalized Ranking for Image Recommendation. In *Proceedings of 11th ACM International Conference on Web Search and Data Mining*.

[15] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 502–511.

[16] Aghny Arisya Putra, Rahmad Mahendra, Indra Budi, and Qorib Munajat. 2017. Two-steps graph-based collaborative filtering using user and item similarities: Case study of E-commerce recommender systems. In *Data and Software Engineering (ICoDSE), 2017 International Conference on*. IEEE, 1–6.

[17] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[18] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*. ACM, 791–798.

[19] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[20] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 111–112.

[21] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[22] Jun Wang, Arjen P De Vries, and Marcel JT Reinders. 2006. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 501–508.

[23] Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, Vol. 11. 2764–2770.

[24] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 153–162.

[25] Pratibha Yadav and Shweta Tyagi. 2017. Hybrid fuzzy collaborative filtering: an integration of item-based and user-based clustering techniques. *International Journal of Computational Science and Engineering* 15, 3-4 (2017), 295–310.

[26] Akihiro Yamashita, Hidenori Kawamura, and Keiji Suzuki. 2011. Adaptive fusion method for user-based and item-based collaborative filtering. *Advances in Complex Systems* 14, 02 (2011), 133–149.

[27] Shuang-Hong Yang, Bo Long, Alexander J Smola, Hongyuan Zha, and Zhaohui Zheng. 2011. Collaborative competitive filtering: learning recommender using context of user choice. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 295–304.

Demonstrations Track Chairs' Welcome

It is our great pleasure to welcome you to the *WWW 2019 Demonstrations Track*. We received 58 proposals from all around the world covering a broad range of topics. We evaluated them regarding relevance, quality, and novelty, selecting 27 demo papers that will be presented along with the poster session during two afternoon sessions.

Review and acceptance statistics are as follows:

Submission Type	Reviewed	Accepted	Acceptance Rate
Demos	58	27	46.55%

Number of Reviewers	Average Assignments Per Reviewer
59	3.95

Number of Submissions	Minimum number of reviewers per paper	Majority number of reviewers per paper
58	3	4

We wish that you will find the demo program interesting, providing you with a valuable opportunity to learn and share ideas with other researchers and practitioners from institutions around the world.

Amin Mantrach
Demo Track Chair
Criteo AI Lab, Palo Alto

Fabrizio Silvestri
Demo Track Chair
Facebook London



Demo Track Organization

Demo Track Chairs:
Amin Mantrach *(Criteo AI Lab, USA)*
Fabrizio Silvestri *(Facebook London, UK)*

Program Committee:
Luca Maria Aiello (Nokia Bell Labs, UK)
Hussein Al-Olimat (Wright State University, USA)
Muhammad Intizar Ali (National University of Ireland Galway, Ireland)
Ghislain Auguste Atemezing (Mondeca, France)
Xiao Bai (Yahoo Research, USA)
Payam Barnaghi (University of Surrey, UK)
Paolo Bellavista (University of Bologna, Italy)
Maria Bermudez-Edo (University of Granada, UK)
Karima Boudaoud (I3S-CNRS / University of Nice Sophia Antipolis, France)
Andrei Ciortea (University of St. Gallen, Switzerland)
Luca Costabello (Accenture Labs, Ireland)
Anwitaman Datta (Nanyang Technological University, Singapore)
Soumya Kanti Datta (EURECOM, France)
Yasmin Fathy (University of Surrey, UK)
Raúl García-Castro (Polytechnic University of Madrid, Spain)
Manas Gaur (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Paul Grace (University of Southampton, UK)
Christophe Gravier (Jean Monnet University, France)
Bin Guo (Telecom SudParis, France)
Amelie Gyrard (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Manfred Hauswirth (Technical University of Berlin and Fraunhofer FOKUS, Germany)
Erisa Karafili (Imperial College London, UK)
Mehdi Kaytoue (Infologic, France)
David Laniado (Eurecat - Technology centre of Catalonia, Spain)
Gyu Myoung Lee (Liverpool John Moores University, UK)
Chunbin Lin (Amazon, USA)
Fangming Liu (Huazhong University of Science and Technology, China)
Maria Maleshkova (University of Bonn, Germany)
Albert Meroño-Peñuela (Vrije Universiteit Amsterdam, Netherlands)
Franco Maria Nardini (Institute of Information Science and Technologies, Italy)
David Nichols (University of Waikato, New Zealand)
Salvatore Orlando (Ca' Foscari University of Venice, Italy)
Josiane Xavier Parreira (Siemens, Austria)
Pankesh Patel (Fraunhofer CESE, USA)

Filipa Peleja (New University of Lisbon, Portugal)
Raffaele Perego (ISTI-CNR, Italy)
Adriano Pereira (Federal University of Minas Gerais, Brazil)
Danh Le Phuoc (Technical University of Berlin, Germany)
Gauthier Picard (Laboratoire Hubert Curien, Institut Henri Fayol, and MINES Saint-Etienne, France)
Thomas Ricatte (Criteo AI Labs, France)
Michele Ruta (Polytechnic University of Bari, Italy)
Saeedeh Shekarpour (University of Dayton, USA)
Bart Thomee (Google, USA)
Nicola Tonellotto (Institute of Information Science and Technologies, Italy)
Flavian Vasile (Criteo AI Labs, France)
Maria Esther Vidal (Simón Bolívar University, Venezuela)
Ziqi Wang (Facebook, UK)
Bebo White (SLAC National Accelerator Laboratory, USA)
Sanjaya Wijeratne (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Zhengming Xing (Criteo AI Labs, USA)
Pranjul Yadav (Criteo AI Labs, USA)
Achille Zappa (National University of Ireland Galway, Ireland)
Antoine Zimmermann (École des Mines de Saint-Étienne, France)

Flavien Balbo (École des Mines de Saint-Étienne, France)
Cristian Consonni (University of Trento, France)
Walid Iguider (University of Cagliari, Italy)
Mirko Marras (University of Cagliari, Italy)
Swati Padhee (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Joy Prakash Sain (Ohio Center of Excellence in Knowledge-enabled Computing, USA)
Julien Tissier (Jean Monnet University, France)



RecBoard: A Web-based Platform for Recommendation System Research and Development

Mohit Chawla*
Cornell Tech, Cornell University
New York, New York
mc2683@cornell.edu

Kriti Singh*
Cornell Tech, Cornell University
New York, New York
ks2259@cornell.edu

Longqi Yang
Cornell Tech, Cornell University
New York, New York
ylongqi@cs.cornell.edu

Deborah Estrin
Cornell Tech, Cornell University
New York, New York
destrin@cornell.edu

ABSTRACT

This paper introduces **RecBoard**, a unified web-based platform that facilitates researchers and practitioners to train, test, deploy, and monitor recommendation systems. RecBoard streamlines the end-to-end process of building recommendation systems by providing a collaborative user interface that automates repetitive tasks related to dataset management, model training, visualization, deployments, and monitoring. Our demo prototype demonstrates how RecBoard can empower common tasks in research and development. RecBoard will be open-sourced and publicly available upon publication.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Recommender Systems; Machine Learning; Web Platforms

ACM Reference Format:
Mohit Chawla, Kriti Singh, Longqi Yang, and Deborah Estrin. 2019. RecBoard: A Web-based Platform for Recommendation System Research and Development. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314133

1 INTRODUCTION

To innovate on and deploy recommender systems, researchers often need to perform a series of repetitive tasks that involve dataset loading, dataset management, model monitoring, visualization, and deployment, as shown in Fig. 1. These repetitive tasks lead to significant overhead for model development and maintenance.

However, existing platforms are limited in addressing this overhead. On one hand, many solutions are focused on limited stages of the development pipeline (Fig. 1). For example, OpenRec [12] supports systematic model training, TensorFlow serving [9] and Clipper [1] enable model deployment in production, and AutoML [3] automates the process of searching for model configurations. On the other hand, existing systems that attempt to consolidate different development stages have a steep learning curve and are restricted to limited use cases — BigML [5] does not offer any recommendation model; Amazon's SageMaker [4] offers a train-test-deploy pipeline only in a hosted notebook environment, and MLBase [6] requires using a task language to declare machine learning tasks and does not provide deployment supports.

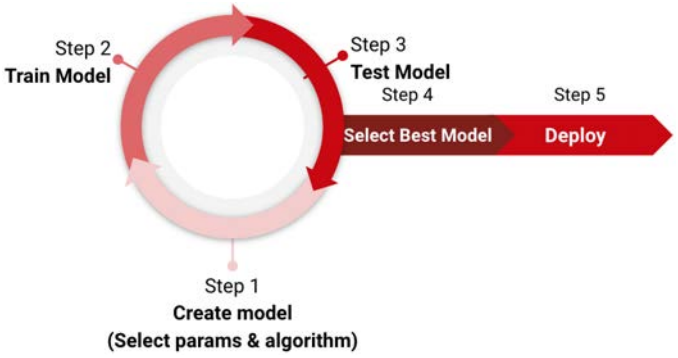


Figure 1: End-to-end process of developing recommender systems. It comprises of various stages: selecting parameters and algorithm, training the model, testing and selecting the best model, and deploying the best model into production.

In this paper, we develop **RecBoard**, an open-sourced and web-based platform that eases the end-to-end process of creating, testing, deploying, and monitoring recommendation models. It unifies five tasks in productionizing a recommendation system: (1) selecting dataset, model and model parameters, (2) training and testing the model, (3) visualizing the training progress (e.g., loss and precision) in real time, (4) collaborating with peers, (5) deploying the model, and (6) monitoring the deployed model. RecBoard is intended for researchers who aspire to create or productionize recommendation systems while collaborating with their peers.

*Both authors contributed equally.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314133

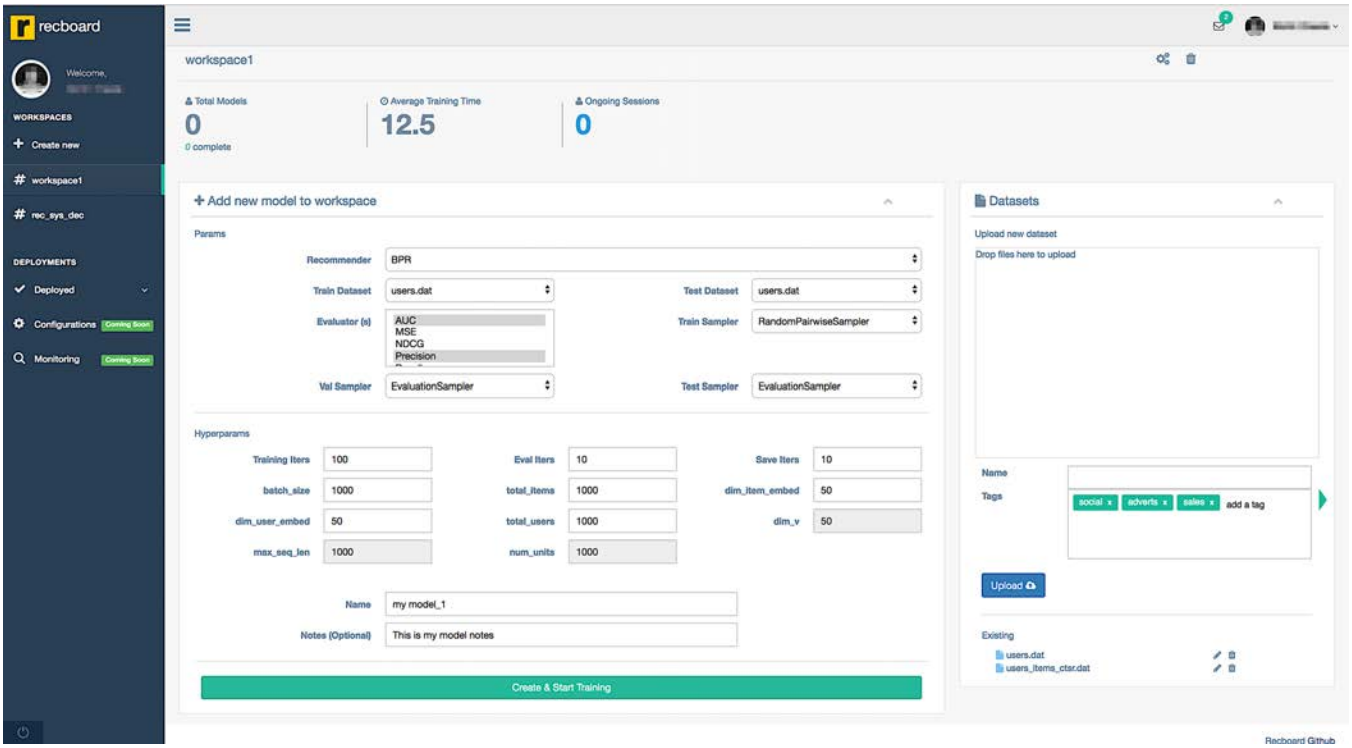


Figure 2: Sample user's workspace. A workspace encapsulates all related datasets and models.

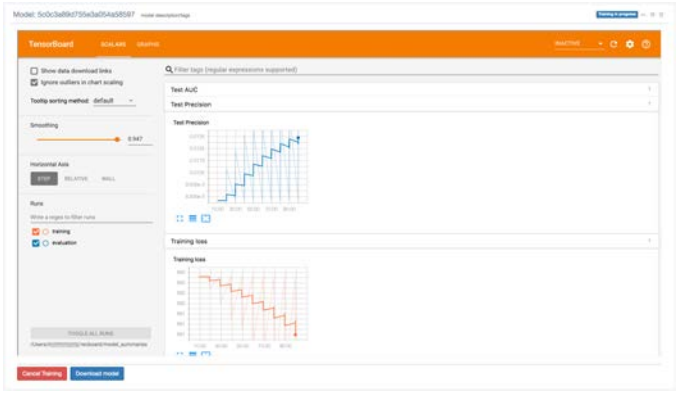


Figure 3: Real-time model variables visualization during training and testing using TensorBoard.

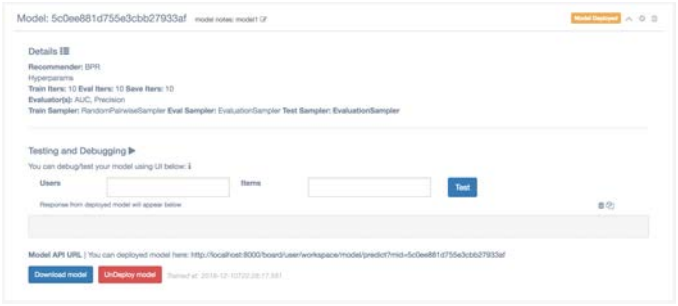


Figure 4: UI (with testing) for a deployed model

2 SYSTEM FEATURES

RecBoard eliminates operational inefficiencies across all stages of productionizing recommendation models. In this section, we present its major functionalities and features.

2.1 Training

RecBoard facilitates users to train models using the user interface as shown in Fig. 2 and enables them to monitor training progress (e.g., loss, precision, and recall) in real-time, as shown in Fig. 3. Since our core training library, OpenRec [12] is based on TensorFlow, we use TensorBoard [11] as the default visualizer. RecBoard allows users to plug in their custom visualization solutions.

2.2 Deployment

RecBoard allows users to deploy a model as a REST API using TensorFlow Serving [9]. This leads to fast, automated and easy to manage deployments. The *Deployment Manager* can be configured to use other deployment solutions, such as Clipper [1].

2.3 Monitoring

RecBoard enables users to monitor metrics and keep track of a model, which is deployed as a REST API. For debugging purposes, an input form is provided, and it enables the user to quickly test if the API of the deployed model is working, as shown in Fig. 4.

2.4 Collaboration

RecBoard enables users to collaborate via features such as dataset sharing, model visualization sharing (Fig. 5).

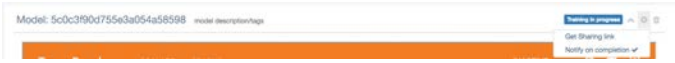


Figure 5: Sharing a model's visualization with peers

2.5 Management

RecBoard provides better model management by introducing workspaces, and model and dataset sharing and tagging.

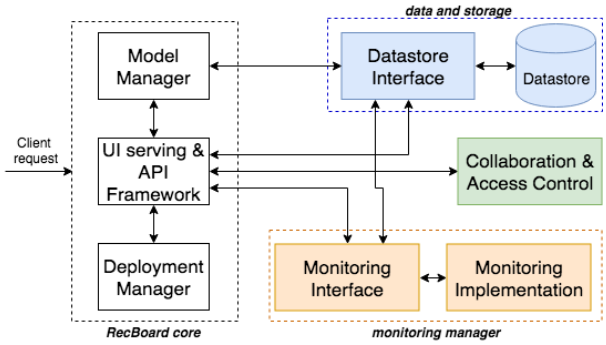


Figure 6: The architecture of RecBoard. All components are designed to automate repetitive tasks in different stages of developing recommender systems.

3 ARCHITECTURE OVERVIEW

The architecture of RecBoard is shown in Fig. 6, which highlights the major components of the system. RecBoard views the process of developing recommender systems as a multi-stage pipeline (Fig. 1) and provides user interfaces to perform tasks relevant to each stage of the pipeline. The tasks include managing datasets, training and testing model(s), organizing models into workspace(s), deploying model(s), and monitoring deployed models. *RecBoard core* is the central component that handles model management (via *Model Manager*), serves the user interface and handles model deployment.

3.1 Modular Design

To make the pipeline efficient, we define reusable components by identifying repetitive tasks and clustering those tasks, as shown in Fig. 7. For example, *Model Manager* automates repetitive tasks such as loading datasets, initializing model, and starting training. Furthermore, RecBoard is modularised in a way that its components can be replaced by custom modules by advanced users. For example, a different database can be connected easily and custom monitoring solutions can be added. Since each model training task is independent of others and is executed outside of the HTTP request-response cycle, we leverage distributed task queues to parallelize model training related operations. A task queue handles invoking

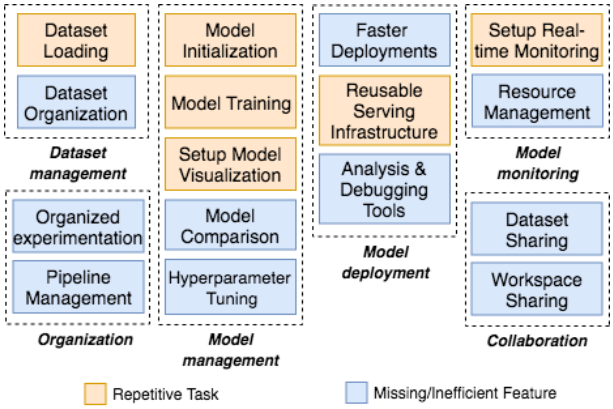


Figure 7: There are a number of repetitive tasks in end-to-end process and some missing features in existing platforms. These can be clustered into reusable components.

code that trains the model, processes results and stores them in a persistent database for later use. This architecture supports an on-premise solution so that users with constraints around data sharing can use RecBoard.

3.2 Components

RecBoard core consists of Model Manager, Deployment Manager and UI Serving & API framework. For a request from client, **UI Serving & API framework** interacts with other components of system and serves the user interface. **Model Manager** provides mechanisms to manage all model related tasks such as training, visualization, testing, management (create, delete, and edit), and download. **Deployment Manager** provides infrastructure for deploying (and un-deploying) a trained model and managing cloud configurations. **Monitoring Manager** provides infrastructure support for monitoring metrics (e.g., requests and up-time) of a deployed model via a monitoring interface. Mechanisms for monitoring are provided by a default implementation. The **Data and storage** component is part of persistence layer and provides an interface which can be implemented by any database wrapper. We used MongoDB in the initial prototype. The **Collaboration and access control** component provides infrastructure for collaboration via features such as: sharing training visualization and sharing datasets.

4 SAMPLE USER WORKFLOWS/EXPERIENCE

This section describes examples of user experience for different tasks performed via RecBoard.

4.1 Developing Recommendation Systems for Online Supermarket

A researcher R_1 at an online supermarket company wants to create a recommender system for clothes and deploy it as a REST API so that other teams in the company can use it to provide clothes' recommendations to users. She opens RecBoard and creates a workspace named "*Clothes Experiments*" to organize all recommender models related to clothes. She splits a dataset into training, validation and testing, uploads them to RecBoard, and adds tags (clothes,

train/test/eval) to them. She decides to create and deploy a Probabilistic Matrix Factorization (PMF) [8] based model. For experimentation, she chooses 3 sets of hyper-parameters to test upon and deploy the best of these. Using the *Add Model to Workspace* Section (Fig. 2), she selects a recommender and other hyper-parameters to add 3 models to the workspace. Now she can monitor key training indicators (e.g., loss, precision, and recall) of these 3 models in real time. After some time, all models are trained and the Model 2's results are the best. She decides to deploy it. To do so, she adds configurations for deployment corresponding to a cloud provider. Next, she clicks the *Deploy* button to deploy the model. After the model is deployed, she sees model testing UI along with API URL (Fig. 4) and decides to test if API is working. To test, she wants to know the output of the model for users with ids 2, 90 for items with ids 3, 81. So, she enters these ids and clicks the *TestAPI* button. She sees responses from the server and verifies that the API is working fine. So, she shares the API URL with other teams.

4.2 Collaborating on Recommendation Systems Development

R_1's team member is a recommender systems developer, R_2. R_2 proposes to experiment with recommenders other than PMF (deployed already). So, R_1 wants to collaborate with R_2. To do so, R_1 shares the "*Clothes Experiments*" workspace (created earlier) using the *Share Workspace* button and entering R_2's email. Now, R_2 has access to the workspace and she creates other models based on Bayesian Personalized Ranking (BPR). She trains new models and finds that, say, one of her models performs better than the model deployed earlier. So, she decides to deploy this model and update other teams with new API URL. Once other teams migrate to this new API, she undeploys previous model using *Undeploy* button (Fig. 4).

5 EVALUATION

In this section, we evaluate RecBoard in terms of efficiency and present examples to demonstrate how users can customize modules and add new recommendation algorithms as per their use case.

5.1 Efficiency: RecBoard as a platform for efficient and organized experimentation

Previously, the end-to-end process was inefficient because it involved writing multiple lines of glue code for repetitive tasks (shown in orange in Fig. 7) and manually setting up pipelines for commonly needed features (shown in blue in Fig. 7). Now, the user can perform the same tasks by a few clicks in the UI, which leads to increased efficiency.

5.2 Extensibility and Maintainability: Adapting RecBoard by customizing existing implementation

It is important that the existing implementation is extensible because users might need additional features and custom implementations of modules. RecBoard provides ways to adapt the platform to custom use cases. For example, additional recommendation algorithms can be added by adding a file similar to the ones in the *recommenders* folder, different datastore (Fig. 6) can be added by extending implementation of the *RecboardDB* class.

As shown above, the existing implementation of RecBoard is extensible and maintainable as it is modular, based on DRY [2] and SOLID [7] principles, and is testable.

6 DEMO OVERVIEW

We plan to demonstrate RecBoard using a user-item interaction dataset with 5551 users and 16980 items. This dataset, *citeulike-a*, was collected by Wang et al. [10] from CiteULike and Google Scholar. CiteULike allows users to create their own collections of articles. There are abstracts, titles, and tags for each article. Auxiliary information such as authors, groups, posting time, and keywords is not used. The details can be found at www.citeulike.ort/faq/data.adp.

The demonstration starts by logging in as a registered user and creating a workspace using *Create New* button in left panel and entering workspace name (e.g., MyWorkspace). This creates a workspace, as shown in Fig. 2. This is followed by uploading the dataset into workspace created above using the *Upload* button. Once the dataset is uploaded, a model is trained by selecting hyper-parameters and other details and clicking *Create and Start Training* button. This would show training process with real-time graphs as shown in Fig. 3. Once training is completed, *Deploy* button is clicked, which deploys the model on local server and presents an API URL along with UI component to test the deployed model's REST API (Fig. 4). We show that deployed model is working by inputing comma-separated sample user-ids and item-ids followed by clicking *Test* button. This shows sample response from the deployed model.

7 CONCLUSION AND FUTURE WORK

RecBoard contributes towards reducing operational inefficiencies in the process of creating recommendation systems. This is made feasible by (1) identifying repetitive tasks in the pipeline, (2) defining reusable components for these tasks, and (3) creating a user interface to automate these tasks using the components defined. This can improve the speed at which recommender models can be productionized, irrespective of production environment or cloud provider. In the future, we plan to improve RecBoard by adding features for dataset visualization, cross-validation and comparison of deployed models. We also plan to explore the idea of a similar platform in other domains, such as Computer Vision, and Natural Language Processing.

8 ADDITIONAL MATERIAL

Screencast video link: http://mohitchawla.in/recboard/index.html

ACKNOWLEDGMENTS

This research was supported by the National Science Foundation under grant IIS-1700832 and by Yahoo Research (via the Connected Experiences Laboratory at Cornell Tech). The work was further supported by the small data lab at Cornell Tech, which received funding from NSF, NIH, RWJF, UnitedHealth Group, Google, and Adobe. We thank the anonymous reviewers for their insightful comments and suggestions.

REFERENCES

[1] Daniel Crankshaw, Xin Wang, Guilio Zhou, Michael J Franklin, Joseph E Gonzalez, and Ion Stoica. 2017. Clipper: A Low-Latency Online Prediction Serving System.. In *NSDI*. 613–627.

[2] Andrew Hunt and David Thomas. 1999. *The Pragmatic Programmer: From Journeyman to Master*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[3] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren (Eds.). 2018. *Automatic Machine Learning: Methods, Systems, Challenges*. Springer. In press, available at http://automl.org/book.

[4] Amazon Web Services Inc. 2018. Amazon SageMaker. http://aws.amazon.com/sagemaker/. Online; accessed: 2018-12-08.

[5] BigML Inc. 2018. BigML: Machine Learning made beautifully simple for everyone. https://bigml.com. Online; accessed: 2018-12-08.

[6] Tim Kraska, Ameet Talwalkar, John C Duchi, Rean Griffith, Michael J Franklin, and Michael I Jordan. 2013. MLbase: A Distributed Machine-learning System.. In *Cidr*, Vol. 1. 2–1.

[7] Robert C Martin. 2000. Design principles and design patterns. *Object Mentor* 1, 34 (2000), 597.

[8] Andriy Mnih and Ruslan R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.

[9] Christopher Olston, Noah Fiedel, Kiril Gorovoy, Jeremiah Harmsen, Li Lao, Fangwei Li, Vinu Rajashekhar, Sukriti Ramesh, and Jordan Soyke. 2017. TensorFlow-Serving: Flexible, high-performance ML serving. *arXiv preprint arXiv:1712.06139* (2017).

[10] Hao Wang, Binyi Chen, and Wu-Jun Li. 2013. Collaborative Topic Regression with Social Regularization for Tag Recommendation.. In *IJCAI*. 2719–2725.

[11] Kanit Wongsuphasawat, Daniel Smilkov, James Wexler, Jimbo Wilson, Dandelion Mané, Doug Fritz, Dilip Krishnan, Fernanda B Viégas, and Martin Wattenberg. 2018. Visualizing dataflow graphs of deep learning models in TensorFlow. *IEEE transactions on visualization and computer graphics* 24, 1 (2018), 1–12.

[12] Longqi Yang, Eugene Bagdasaryan, Joshua Gruenstein, Cheng-Kang Hsieh, and Deborah Estrin. 2018. OpenRec: A Modular Framework for Extensible and Adaptable Recommendation Algorithms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 664–672.

CrowdPT: Summarizing Crowd Opinions as Professional Analyst

Chung-Chi Chen
Department of Computer Science and Information Engineering, National Taiwan University, Taiwan
cjchen@nlg.csie.ntu.edu.tw

Hen-Hsen Huang
Department of Computer Science, National Chengchi University, Taiwan
MOST Joint Research Center for AI Technology and All Vista Healthcare, Taiwan
hhhuang@nccu.edu.tw

Chia-Wen Tsai
Department of Computer Science and Information Engineering, National Taiwan University, Taiwan
cwtsai@nlg.csie.ntu.edu.tw

Hsin-Hsi Chen
Department of Computer Science and Information Engineering, National Taiwan University, Taiwan
MOST Joint Research Center for AI Technology and All Vista Healthcare, Taiwan
hhchen@ntu.edu.tw

ABSTRACT

This paper demonstrates a novel analytics service, CrowdPT, for capturing the key information, price target (PT), of individual investors on social media. PT, which is mentioned as a conclusion in most of analysts' reports, indicates not only the market sentiment (bullish/bearish) of investors, but also the analysis results. In order to provide the latest opinions of individual investors, we monitor Twitter in real time and update the information in price chart daily. For all component stocks in Dow Jones Industrial Average, textual information from numerous tweets is summarized into a single number, PT, in CrowdPT. Case studies confirm the effectiveness of our analytics service in the financial domain, and show that capturing the PT of individual investors is promising for stock price prediction. The Web API of CrowdPT is also provided for academic purpose.

CCS CONCEPTS

• **Information systems** → **Information extraction**.

KEYWORDS

price target, financial social media, crowd opinion

ACM Reference Format:
Chung-Chi Chen, Hen-Hsen Huang, Chia-Wen Tsai, and Hsin-Hsi Chen. 2019. CrowdPT: Summarizing Crowd Opinions as Professional Analyst. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314122

1 INTRODUCTION

The application of artificial intelligence technologies to financial domain has attracted much attention in recent years. A variety of related tasks and researches are addressed in the shared tasks and workshops collocated with predominant conferences, including SemEval-2017 Task 5, FiQA-2018, FNP-2018, ECONLP-2018, and FinNum-2019.

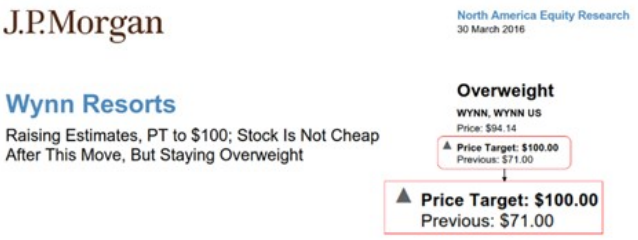


Figure 1: Example of professional analyst's report.

According to the important information embedded in the numerals in financial narratives, it has caught many researchers' eyes [1, 2, 4]. Especially, forecasting the price target (PT), the price level that the price of a financial instrument will reach, is one of the major goals of analyzing financial instrument. As shown in Figure 1, most of professional analysts show their PTs in the top of the first page of analyst reports for the customers to capture their analysis results at a glance. In the left side of Figure 1, the analyst uses a piece of short text to summarize the report, and mentions the PT. In the right side of Figure 1, the analyst shows the analysis results in a table with the information such as the overweight comment and the PT. However, social media data is unstructured, and the PT information in tweets could only be extracted with natural language processing technique. Since PT is an extremely crucial indicator in the procedure of financial analysis, capturing the PT based on the crowd opinions is an important topic to investigate.

Sometimes the institutional investors may not release the report for a certain company. At this time, the price target forecasted by individual investors can be used to complement the missing information. For example, Bloomberg does not record the price targets of the analyst for $AXP in August, October, and November in 2017. With the PT sorted out by our system, the information can be filled up. Even though PT is quite meaningful for presenting the opinions and analysis results of investors, to the best of our

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314122

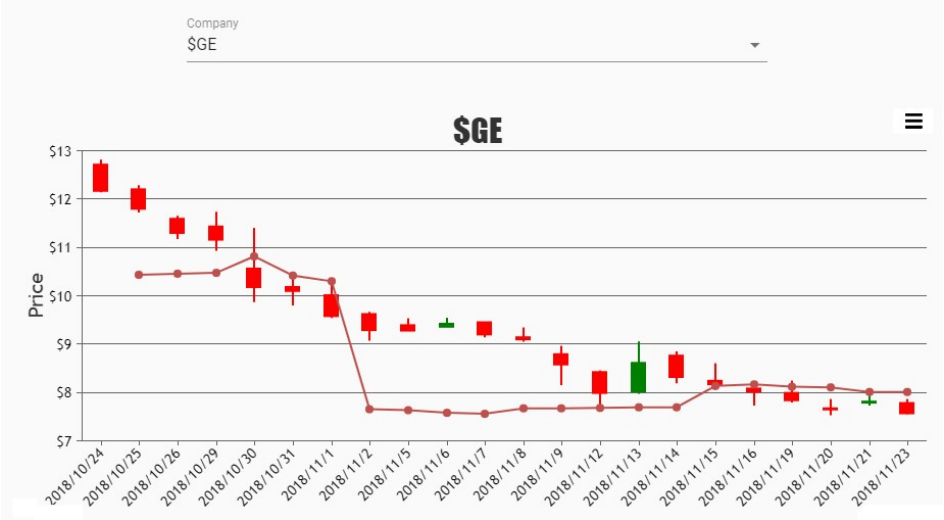


Figure 2: Screenshot of CrowdPT.

Table 1: Examples of subcategories.

Subcategory	Target Numeral	Tweet
price target	14.35	$CIEN, CIEN seems to have broken out of a major horizontal resistance. Targets $**14.35**.
support or resistance	46	$CTRP, $**46** Breakout Should be Confirmed with Wm%R Stochastic Up
buying	137.89	$SPY Long 1/2 position **137.89**
selling	36.50	$KOG - hopefully a better outcome than getting kneecapped by $BEXP selling itself dirt cheap at **36.50**
other	110.2	$FB (**110.20**) is starting to show some relative strength and signs of potential B/O on the daily.

knowledge, none of the previous systems provide the function for extracting the PTs of individual investors. Once the PT is extracted, the user gets not only the market sentiment, but also the degree of the market sentiment, calculating by the difference between the PT and the newest close price. Moreover, the expected value of a certain financial instrument is estimated. Many in-depth empirical experiments can be extended with the PT extracted from social media text.

This work presents CrowdPT, a service providing the PTs of individual investors based on social media text. For each stock, our system provides the PT extracted from social media text. Case studies show that PT from social media data is useful for investors when making decisions, and provides more insights for the behavior of investors.

Our contributions are three-fold: (1) We propose CrowdPT, a novel Fintech application based on natural language processing, and demonstrate its potential values. (2) In addition to the demonstration website, we also provide our model as a cloud service for downstream applications and extended research. (3) CrowdPT provides the real-time market information for all component stocks in Dow Jones Industrial Average. The case studies show the PTs of individual investors based on financial social media data help investors make investment decisions.

2 SYSTEM OVERVIEW

Figure 2 shows a screenshot of CrowdPT. The red bar and green bar, called candlestick chart, are used to show the daily open, high, low, and close price of a stock. The red line shows the average of the PTs from financial tweets, which can be easily compared with the market price shown in the candlestick chart. For those days on which no numerals are classified as PT, approximate PTs are shown with dotted lines. In Figure 2, we use $GE, General Electric Company, as an example. On Oct 25, the price of $GE was about 12, and the PT that we sorted out from the financial tweets was about 10.5. It shows that individual investors forecast the price will fall down to 10.5 in the future. On Oct 30, $GE's stock price came to the PT on Oct 25. Moreover, individual investors modified their PT to about 7 on Nov 2. Then, the price matched this PT on Nov 12, and descended to crowd's PT on Nov 15. These cases show the usefulness of crowd price target and indicate that CrowdPT provides the leading indicator for stock market prediction.

CrowdPT summarizes the textual information from financial tweets into single numeral (PT), and provides the real-time information for all component stocks in Dow Jones Industrial Average. The market sentiment, bullish or bearish, can be captured at a glance with the price chart. When the PT is higher than the close price, it means that individual investors take bullish market sentiment

toward the stock. On the other hand, investors have bearish market sentiment to the stock when the PT is lower than the close price. Beyond market sentiment, CrowdPT provides more fine-grained information for individual investors' opinions. With the PTs sorted out, not only the degree of the market sentiment but also the expected price level would be captured.

Additionally, we provide the Web API for disambiguating the meaning of the numerals in financial tweet. We follow the taxonomy tailor-made for the numerals in financial social media data [1]. Categories and subcategories are shown in Figure 3. Table 1 shows the example tweets for each subcategory.

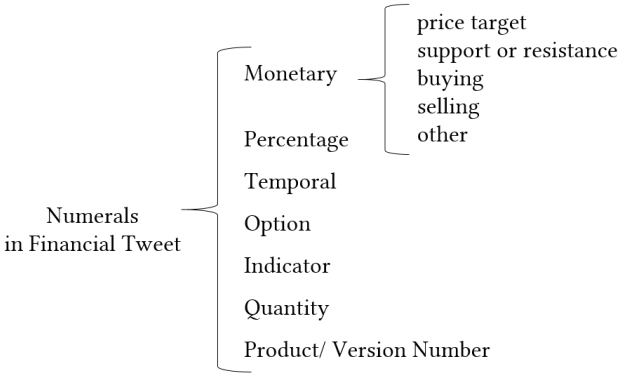


Figure 3: Numeral taxonomy in financial tweets.

3 TECHNICAL IMPLEMENTATION

In CrowdPT, we adopt the word-based CNN model for financial numeral understanding [1]. Here, we merge the subcategories that do not imply any subjective opinions of investors, including "money", "quote", "change", and the rare subcategory, "stop loss" in Monetary category into one class, called "other".

3.1 Dataset

We adopt FinNum dataset for experiments [1]. Three experts in the financial domain are involved to annotate the numerals in the SemEval-2017 Task 5 dataset, in which financial tweets from Twitter and StockTwits are collected and labeled with market sentiment. Total 1,341 numerals are annotated. The average agreement of the annotation on price target achieves kappa value of 65.59%, which is considered as substantial agreement [3].

3.2 System Flow

Figure 4 illustrates the flowchart of price target extraction. Firstly, we use tweepy [1] , a Python library, to monitor the latest tweets published in Twitter. Only the tweets mentioning the cashtags of the constituent stocks of Dow Jones Index will be crawled. Secondly, the price target filter is performed at the end of the day (GMT -8) to sort out the price target for each cashtag. Then, a classifier based on convolutional neural network (CNN) is adopted for extracting the price target. Finally, we collect the close price from Yahoo Finance

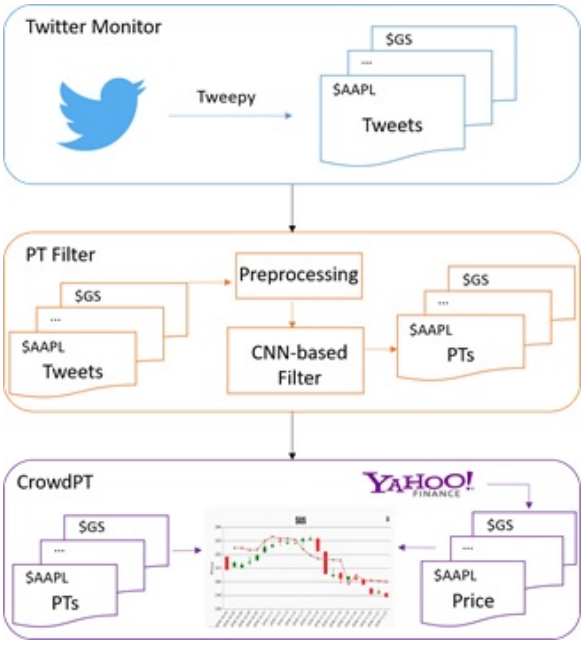


Figure 4: System flowchart of price target extraction.

Table 2: Experimental results of subcategory classification.

	target	sup/res	buying	selling	other
P	42.50	41.34	36.80	35.36	50.66
R	50.90	30.68	27.29	54.05	80.53
F1	45.32	34.27	28.55	40.17	61.51

with the fix_yahoo_finance [2] and render the price chart. Our system also provides the analysis results in json format via Web APIs.

3.3 Results of Numeral Disambiguation

In the 7-way category classification task, our word-based CNN model achieves an accuracy of 67.61%, which significantly outperforms other models with RNN structures under 95% confidence level. Please refer to our previous work [1] for more details of model setting and comparison. In the subcategory classification task, the precision, recall and F1-score for each subcategory are shown in Table 2.

4 DETAILS OF WEB API

Our Web API returns the classification results of the numerals in input financial tweets. A financial tweet should contain at least one cashtag, and our target tweets are those financial tweets that have at least one numeral. In this paper, we demonstrate the usefulness of the extracted PT, and more empirical studies could be extended based on other (sub)categories.

The participants of the financial market can be separated into two groups, institutional investors and individual investors. Yet, only the

[1] http://www.tweepy.org/

[2] https://github.com/ranaroussi/fix-yahoo-finance

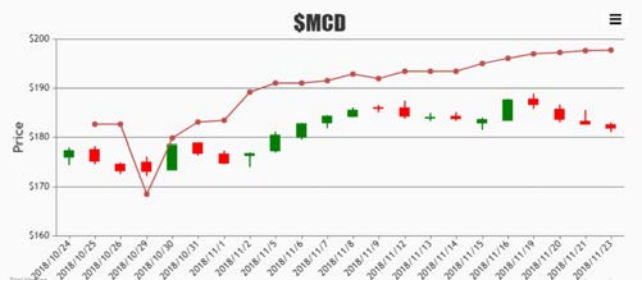


Figure 5: Correlation of stock price and crowd's PT.



Figure 6: Evidence for crowd's PT as leading indicator.

analysis results of institutional investors are recorded in structural format by market information providers like Bloomberg. None of previous systems attempts to summarize the textual information on the financial social media data into price target. In CrowdPT, the latest PT from individual investors can be obtained and be compared with the latest close price in the provided price chart.

5 INVESTIGATION OF CROWD PRICE TARGET

We perform backtesting of the PT of both institutional and individual investors in 2017, and discuss our findings as follows. The performances of both institutional investors and individual investors can be compared within CrowdPT. For example, the price of $DWDP reached the September price target of institutional investors, but did not reach their November price target until now. In contrast, the price did not reach the September price target of individual investors, but reached their November price target. That shows the information of both institutional investors and individual investors may be comparable. With CrowdPT, investors can obtain more insights than before and develop new trading strategies.

Figure 5 shows the correlation between stock price and crowd's PT. The price did not match the PT during November, but the close prices follow the trend of PT. In other words, the crowds upgrade the PT first and the close price of $MCD rises toward the PT. Figure 6 shows that CrowdPT not only provides the bullish information but also the bearish information. It also shows that the crowd's PT can be the leading indicator for stock market movement prediction. The cases in Figure 2, 5, 6 show the evidence that CrowdPT provide

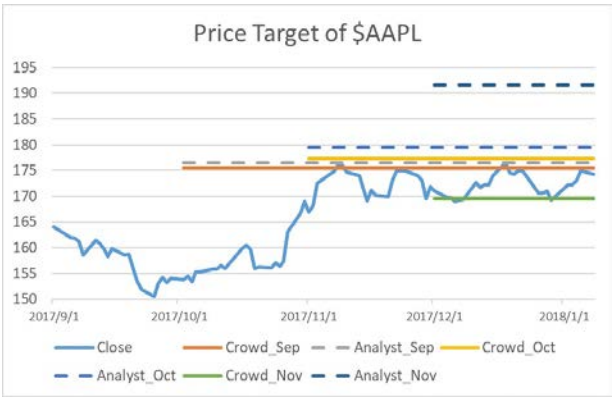


Figure 7: Price target of $AAPL.

the longer-term market movement forecast, which is more sensible for investment, and different from previous work [5] focus on speculation (predict the short-term price movement).

More empirical studies can be conducted with the analysis results of CrowdPT. For instance, the difference of the tendency of setting price target between institutional investors and individual investors can be investigated. Figure 7 shows the comparison of the price target of $AAPL in September, October, and November. We find that the price targets of the analysts are always higher than that of the crowd. That shows the analysts may be more optimistic than individual investors, or individual investors may tend to focus on the short-term prediction. According to this finding, investors could consider to follow the opinions of the crowd for short-term trading and follow the opinions of analysts for long-term investment.

6 CONCLUSION

We present CrowdPT [3], a service aiming at providing price-target oriented analysis results based on financial social media text. Novel applications based on our model are demonstrated. Investors can benefit from the information provided by our system. In particular, trading strategies combine the price targets of institutional investors and individual investors have great potential for further studies. With our Web API [4], users can get the fine-grained classification results based on the tailor-made taxonomy for numerals in financial social media data.

ACKNOWLEDGMENTS

This research was partially supported by Ministry of Science and Technology, Taiwan, under grants MOST 107-2218-E-009-050-, MOST-106-2923-E-002-012-MY3, MOST-107-2634-F-002-011-, MOST-108-2634-F-002-008-, and MOST-107-2218-E-009-050-, and by Academia Sinica, Taiwan, under grant AS-TP-107-M05.

REFERENCES

[1] Chung-Chi Chen, Hen-Hsen Huang, Yow-Ting Shiue, and Hsin-Hsi Chen. 2018. Numeral Understanding in Financial Tweets for Fine-grained Crowd-based Forecasting. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. IEEE, 136–143.

[3] http://nlg6.csie.ntu.edu.tw/CrowdPT
[4] http://nlg6.csie.ntu.edu.tw/FinNum/?demo=CrowdPT&tweet= <input tweet>

[2] Matthew Lamm, Arun Tejasvi Chaganty, Christopher D Manning, Dan Jurafsky, and Percy Liang. 2018. QSRL: A Semantic Role-Labeling Schema for Quantitative Facts. In *Proceedings of the 1st Financial Narrative Processing Workshop*.
[3] J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics* (1977), 159–174.
[4] Soichiro Murakami, Akihiko Watanabe, Akira Miyazawa, Keiichi Goshima, Toshihiko Yanase, Hiroya Takamura, and Yusuke Miyao. 2017. Learning to generate market comments from stock prices. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1374–1384.
[5] Yumo Xu and Shay B Cohen. 2018. Stock movement prediction from tweets and historical prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1970–1979.

TIM: A Tool for Gaining Insights into Psychotherapy

Ronan Cummins
Ieso Digital Health
Cambridge, UK
r.cummins@iesohealth.com

Michael P. Ewbank
Ieso Digital Health
Cambridge, UK
m.ewbank@iesohealth.com

Alan Martin
Ieso Digital Health
Cambridge, UK
a.martin@iesohealth.com

Valentin Tablan
Ieso Digital Health
Cambridge, UK
v.tablan@iesohealth.com

Ana Catarino
Ieso Digital Health
Cambridge, UK
a.catarino@iesohealth.com

Andrew D. Blackwell
Ieso Digital Health
Cambridge, UK
a.blackwell@iesohealth.com

ABSTRACT

We introduce and demonstrate the usefulness of a tool that automatically annotates therapist utterances in real-time according to the therapeutic role that they perform in an evidence-based psychological dialogue. This is implemented within the context of an on-line service that supports the delivery of one-to-one therapy. When combined with patient outcome measures, this tool allows us to discover the active ingredients in psychotherapy. In particular, we show that particular measures of therapy content are more strongly correlated with patient improvement than others, suggesting that they are a critical part of psychotherapy. As this tool gives us interpretable measures of therapy content, it can enable services to quality control the therapy delivered. Furthermore, we show how specific insights can be presented to the therapist so they can reflect on and improve their practice.

CCS CONCEPTS

• **Information systems** → **Decision support systems**; • **Computing methodologies** → *Discourse, dialogue and pragmatics*; • **Applied computing** → *Health care information systems.*

KEYWORDS

Psychotherapy; Natural Language Processing; Deep Learning; Telemedicine; eHealth

ACM Reference Format:
Ronan Cummins, Michael P. Ewbank, Alan Martin, Valentin Tablan, Ana Catarino, and Andrew D. Blackwell. 2019. TIM: A Tool for Gaining Insights into Psychotherapy. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3308558.3314128

1 INTRODUCTION

Mental health is a substantial healthcare concern for the world's population. It is estimated that one in four people are affected by a mental illness during their lifetime and that mental illness costs the global economy $1 trillion per year in lost productivity alone. Psychotherapy is an effective intervention for a range of mental health problems. In particular, cognitive behavioural therapy (CBT) is a popular and effective evidence-based talking therapy that teaches patients the skills and techniques needed to help overcome and manage their symptoms. While CBT has been shown to be as effective as medication on a range of disorders, only half of those diagnoseable with a specific condition actually recover. In fact, the recovery rates for CBT have largely been stagnant, or even decreasing [8], in the last 30 years. Furthermore, given that the vast majority of therapy happens behind closed doors, it is difficult to objectively measure the quality of therapy being delivered or to identify the aspects of therapy that are most effective.

While there has been a technological revolution in the area of physical health in the last 50 years, where practitioners have been equipped with more accurate instruments, diagnostic tests, and medical devices (e.g. x-rays, MRI, blood tests, DNA sequencing), there has been very little change in how psychotherapy has been delivered. However, it is increasingly being recognised that online applications have a crucial role to play in increasing access to services, alleviating the burden on existing mental health services, and in driving innovation in the mental health domain.

One recent development in the mental healthcare sector in the UK is the use of online systems to support the delivery of psychotherapy. One particular case is internet-enabled CBT (IECBT) which allows a therapist to deliver CBT to a patient over their mobile device via real-time text-based conversation. This is an important development as it means that every interaction between a therapist and patient is recorded. In this work, we present a tool[1] that uses deep learning to automatically categorise therapist utterances according to the role that they play in therapy. In order to train and evaluate our model, human annotations were gathered from 290 hours of therapy. We then applied our tool to a large-scale dataset containing session transcripts from over 14,000 cases of IECBT (approximately 90,000 hours of therapy) to calculate a quantifiable measure of treatment delivered. Finally, using a logistic regression analysis, we investigated the relationship between the quantity (or âĂŸdosageâĂŹ) of each aspect of therapy delivered and clinical outcomes.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314128

[1] https://youtu.be/TSfuxYctyik

2 BACKGROUND AND ONLINE CBT

While there has been an increase in online forums and web-based applications for mental healthcare, in most cases these deliver self-help materials for patients with mild symptoms and are not a replacement for one-to-one therapy. In IECBT, a patient communicates one-to-one with a qualified CBT therapist using a real-time text-based message system. IECBT has been shown to be clinically effective for the treatment of depression [5] and is currently commissioned in the UK by the National Health Service (NHS) under the Improving Access to Psychological Therapies (IAPT) [2] programme. As CBT is a short-term goal-orientated intervention, the typical number of sessions is between five and 20. As mandated by IAPT, symptoms are measured before each therapy session using self-reported instruments i.e. Patient Health Questionnaire (PHQ-9) and Generalised Anxiety Disorder Assessment (GAD-7). An example segment of therapy delivered using this modality is shown in Figure 1.

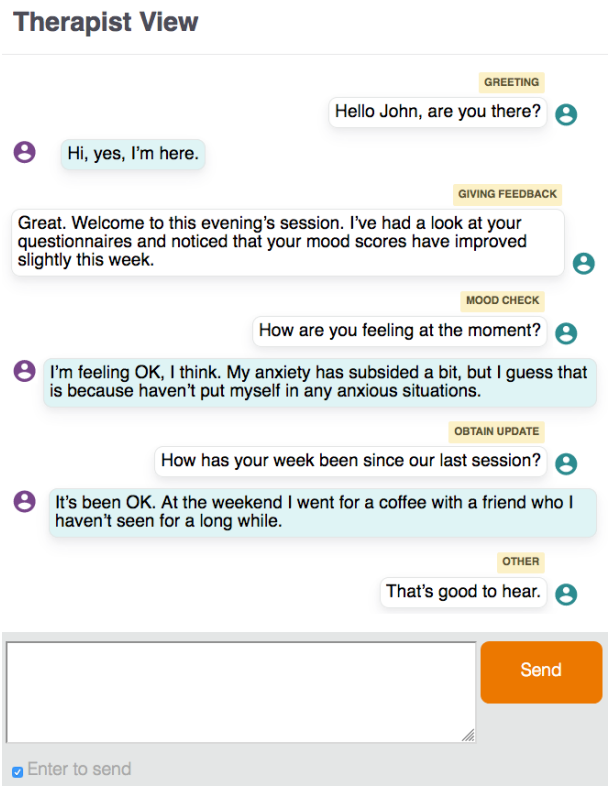


Figure 1: A screenshot of an example therapy session with labels being applied in real-time to each therapist utterance by our deep learning model (TIM).

Some related work [6, 11] has developed a dialog-act modelling approach for motivational interviewing conducted on spoken transcripts; the aim of which is to measure the fidelity of the session to the therapy model. This differs from our work in terms of the psychotherapy domain, the modality, and the fact that they do not aim to discover which aspects of therapy are most effective.

Table 1: Categories of therapist utterances (* means that the category is stylistic and was determined by a regular expression)

Therapist Utterance Categories	
Mood Check	Planning for the Future
Obtain Update	Elicit Feedback
Bridge	Summarise Session
Risk Check	Greeting
Set Agenda	Arrange next Session
Review Homework	Goodbye
Set Goals	Other
Formulation	*Socratic Questioning
Give Feedback	*Therapeutic Thanks
Change Methods	*Therapeutic Empathy
Perceptions of Change	*Therapeutic Praise
Set Homework	*Collaboration

3 HUMAN ANNOTATION

In conjunction with an experience clinically trained CBT practitioner, a research psychologist developed a set of 19 content-based categories and five stylistic categories that capture the role that therapist utterances play in CBT. The research psychologist annotated 290 hours of therapy (11,221 utterances) applying one or more of the 19 content-based categories to each utterance. The five stylistic categories were identified using regular expressions. A list of all the therapy categories used in the annotation effort are shown in Table 1. Using the annotated data, we trained a deep learning model using 8,859 utterances from 230 therapy sessions with the remaining data being kept for validation (30 transcripts) and testing purposes (30 transcripts). The deep learning model is described in the next section.

4 DEEP LEARNING ARCHITECTURE

We developed a deep learning model (see Figure 2) to automatically classify each utterance into one or more of the 24 categories. Firstly, we used word2vec [10] on a preprocessed version of the entire data set of over 90,000 transcripts (approx. 200M words) to learn word embeddings that are suited to the domain of psychotherapy. We preprocessed our data by tokenizing according to whitespace and punctuation and then by lower-casing all tokens (we kept punctuation symbols as separate tokens). This resulted in a vocabulary of 89,260 words, each represented as a continuous dense vector of length 200.

We modelled each utterance in a transcript as a sequence of word embeddings and fed these into a bidirectional long short-term memory [7] (BiLSTM). We used max-pooling over the hidden states of the BiLSTM to encode each utterance as a fixed-length vector. In order to model each utterance in the context of the entire transcript, we fed each of the fixed-length utterance representations in a transcript into another BiLSTM and used the hidden state at each time-step to feed into our final output layer. For both our BiLSTM stages, we used a hidden dimension of 400 and used dropout of 0.5 [4]. A fully connected layer maps each fixed length utterance-in-context representation into a vector of length 23 (23 classes with

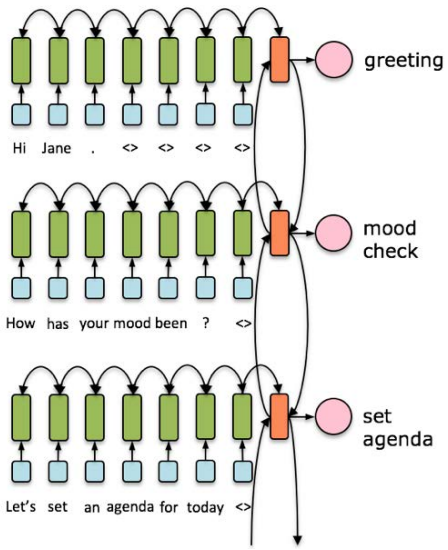


Figure 2: A diagram of the deep learning architecture where word representations (blue) are transformed via an LSTM into word-in-context representations (green) which are max-pooled into utterance representations (orange) and are fed into another LSTM to create utterance-in-context representations (pink).

the 'Other' class modelled as all zeros) with a sigmoid activation function used on this output. Conceptually, each category is modelled as a binary classifier. The training labels were obtained from manual annotation for 19 of the categories, and from high-precision regular expression for the remaining five categories. We call our model TIM (Therapy Insights Model).

5 EXPERIMENTS

We compared our model to a number of other baseline systems for sentence classification. As a simple baseline, we used a multilayer perceptron (MLP) that uses simple unigram lexical features as input. This MLP had one hidden layer the size of which was tuned on the validation set. As a more advanced baseline we used StarSpace [12] which is reported to outperform FastText [9] on a number of classification tasks. On the validation set, we found that a bigram model worked best. We also used a BiLSTM with max-pooling as another baseline [3]. For this baseline, we tuned the hidden dimension and dropout on the validation set. These three baselines treat each utterance in isolation and have no wider contextual information available during classification.

5.1 Classification Results

Table 2 shows the results (F1-score) of the various automated approaches to the utterance annotation task for the 19 manually annotated categories. The MLP that uses the simple lexical features is the worst performing model. The best non-contextual model is the BiLSTM. Overall, the best performing model is TIM which uses contextual information from the wider transcript when annotating an utterance.

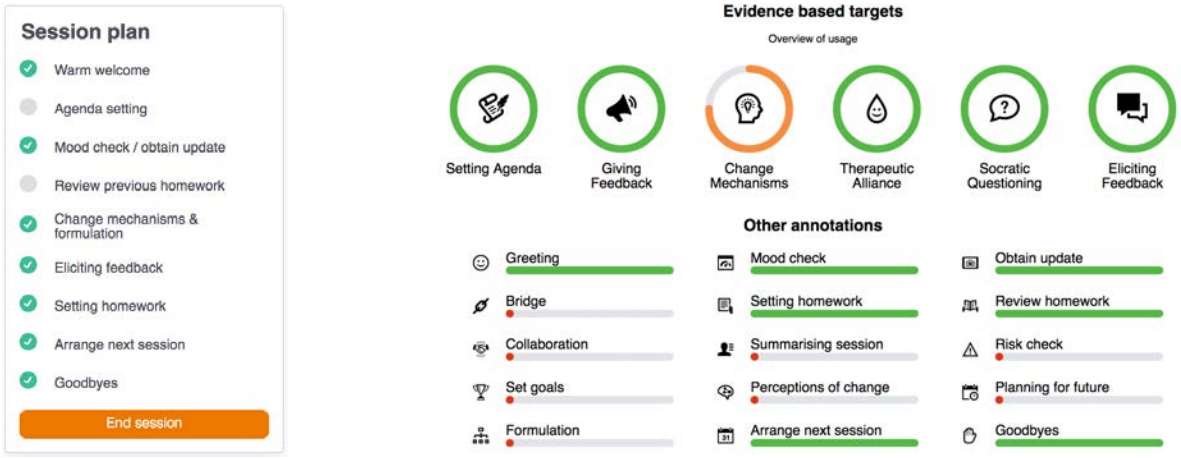
Table 2: F1 scores for the manually annotated categories on test data for the three baselines and TIM.

Category	#	MLP	SS	BiLSTM	TIM
Arrange next Session	45	0.68	0.84	0.86	0.87
Bridge	12	0.50	0.52	0.60	0.56
Change Methods	428	0.62	0.69	0.65	0.68
Elicit Feedback	42	0.30	0.34	0.55	0.61
Formulation	44	0.35	0.39	0.54	0.65
Give Feedback	74	0.26	0.39	0.37	0.37
Goodbye	41	0.84	0.91	0.87	0.92
Greeting	28	0.81	0.89	0.95	1.00
Mood Check	24	0.55	0.56	0.66	0.68
Obtain Update	35	0.37	0.43	0.52	0.61
Other	293	0.52	0.60	0.58	0.63
Perceptions of Change	20	0.12	0.37	0.25	0.25
Planning for the Future	24	0.20	0.21	0.35	0.49
Review Homework	37	0.21	0.43	0.39	0.40
Risk Check	15	0.52	0.81	0.70	0.72
Set Agenda	53	0.54	0.63	0.64	0.77
Set Goals	12	0.25	0.29	0.59	0.64
Set Homework	52	0.29	0.44	0.47	0.60
Summarise Session	0	n/a	n/a	n/a	n/a
macro-avg-F1		**0.44**	**0.54**	**0.58**	**0.64**
micro-avg-F1		**0.39**	**0.53**	**0.56**	**0.66**

6 OUTCOME RELATED INSIGHTS

In order to understand the therapy features that are associated with reduced symptoms, we performed a logistic regression on statistically reliable improvement. Patients are defined as having reliably improved if they have dropped by 6 points on the PHQ-9 scale or 4 points on the GAD-7 scale (with no reliable increase in the other) [2]. Patients scoring above 9 on the PHQ-9 or above 7 on the GAD-7 were classed as meeting the clinical threshold at assessment and were included in this regression analysis (approx. 14,000 cases). The TIM tool was used to automatically annotate all sessions (excluding the final session) for those patients (approx. 90,000 hours of therapy). Then the number of words of each therapy feature in Table 1, averaged over all sessions attended by a patient (excluding the final session), was used as a measure of the 'dosage' of that feature administered to that patient (case), while controlling for the number of sessions and other patient variables [1].

The results revealed a significant relationship between a number of features and statistically reliable improvement. 'Therapeutic Praise', 'Planning for the Future', 'Perceptions of Change', 'Set Agenda', 'Elicit Feedback', 'Review Homework' and 'Give Feedback' were all positively associated with reliable improvement. 'Change Methods' - a CBT-specific feature of therapy was also positively related with improvement. By contrast, increases in'non-therapy' related content ('Other', 'Greeting' and 'Goodbye') were negatively related with improvement, as were 'Therapeutic Empathy', 'Risk Check' and 'Bridge'. The output from this analysis can be used in a number of important clinical ways, including quality control, clinical decision support, and clinical supervision.



(a) clinical decision support **(b) therapist feedback**

Figure 3: A screen-shot of in-session clinical decision support (a) and of therapist feedback (b) after a particular session (green indicates a suitable amount of that particular content was delivered).

6.1 Automated Quality Control

The output of the logistic regression model enables automated quality assurance. Using the logistic regression model as a predictive model, we can estimate the probability of recovery for any therapy session based on the quantity of the specific categories. Space limits the inclusion of a screen-shot of the output of a daily ranking of sessions.

6.2 Clinical Decision Support

The real-time nature of the automatic annotation tool allows us to monitor the delivery of crucial elements in live therapy. For example, therapists can be reminded to set an agenda or to set homework when they have not done so. Figure 3a shows an example of this in terms of a session plan which is updated as items are addressed by the therapist.

6.3 Automated Clinical Supervision

Clinical supervision allows therapists to discuss their practice with an experienced supervisor as part of their professional development. Given the data-driven nature of our insights, we can supply our therapists with summaries and insights for every session that they deliver. Figure 3b is an example of a session summary presented after a session which indicates which aspects of therapy are sufficient and which may be lacking. This can help therapists reflect on the specific session or on the therapy they deliver more generally.

7 CONCLUSION

This paper demonstrates a tool (TIM) that automatically annotates CBT transcripts. Along with paired outcome data, it engenders a sea-change in how psychotherapy can evolve in the digital age. We have demonstrated how it can be used to inform clinical best practice and furthermore, we outline three use cases for how this technology can be used in practice.

REFERENCES

[1] Ana Catarino, Sarah Bateup, Valentin Tablan, Katherine Innes, Stephen Freer, Andy Richards, Richard Stott, Steven D Hollon, Samuel Robin Chamberlain, Ann Hayes, et al. 2018. Demographic and clinical predictors of response to internet-enabled cognitive–behavioural therapy for depression and anxiety. *BJPsych open* 4, 5 (2018), 411–418.

[2] David M Clark. 2011. Implementing NICE guidelines for the psychological treatment of depression and anxiety disorders: the IAPT experience. *International review of psychiatry* 23, 4 (2011), 318–327.

[3] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364* (2017).

[4] Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*. 1019–1027.

[5] Tad Hirsch, Christina Soma, Kritzia Merced, Patty Kuo, Aaron Dembe, Derek D. Caperton, David C. Atkins, and Zac E. Imel. 2018. "It's Hard to Argue with a Computer": Investigating Psychotherapists' Attitudes Towards Automated Evaluation. (2018), 559–571.

[6] Tad Hirsch, Christina Soma, Kritzia Merced, Patty Kuo, Aaron Dembe, Derek D Caperton, David C Atkins, and Zac E Imel. 2018. It's hard to argue with a computer: Investigating Psychotherapists' Attitudes towards Automated Evaluation. In *Proceedings of the 2018 Designing Interactive Systems Conference*. ACM, 559–571.

[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[8] Tom J Johnsen and Oddgeir Friborg. 2015. The effects of cognitive behavioral therapy as an anti-depressive treatment is falling: A meta-analysis. *Psychological Bulletin* 141, 4 (2015), 747.

[9] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* (2016).

[10] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[11] Michael Tanana, Kevin A Hallgren, Zac E Imel, David C Atkins, and Vivek Srikumar. 2016. A comparison of natural language processing methods for automated coding of motivational interviewing. *Journal of substance abuse treatment* 65 (2016), 43–50.

[12] Ledell Yu Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. 2018. Starspace: Embed all the things!. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

QAnswer: A Question Answering prototype bridging the gap between a considerable part of the LOD cloud and end-users

Dennis Diefenbach
Université de Lyon, CNRS UMR 5516
Laboratoire Hubert Curien
Saint-Etienne, France
dennis.diefenbach@univ-st-etienne.fr

Pedro Henrique Migliatti
Universidade Federal de São Carlos, MaLL
São Carlos, Brasil
619744@comp.ufscar.br

Omar Qawasmeh
Université de Lyon, CNRS UMR 5516
Laboratoire Hubert Curien
Saint-Etienne, France
omar.alqawasmeh@univ-st-etienne.fr

Vincent Lully
Sorbonne Université
Paris, France
vincent.lully@sorbonne-universite.fr

Kamal Singh
Université de Lyon, CNRS UMR 5516
Laboratoire Hubert Curien
Saint-Etienne, France
kamal.singh@univ-st-etienne.fr

Pierre Maret
Université de Lyon, CNRS UMR 5516
Laboratoire Hubert Curien
Saint-Etienne, France
pierre.maret@univ-st-etienne.fr

ABSTRACT

We present QAnswer, a Question Answering system which queries at the same time 3 core datasets of the Semantic Web, that are relevant for end-users. These datasets are Wikidata with Lexemes, LinkedGeodata and Musicbrainz. Additionally, it is possible to query these datasets in English, German, French, Italian, Spanish, Pourtuguese, Arabic and Chinese. Moreover, QAnswer includes a fallback option to the search engine Qwant when the answer to a question cannot be found in the datasets mentioned above. These features make QAnswer as the first prototype of a Question Answering System over a considerable part of the LOD cloud.

CCS CONCEPTS

• **Information systems → Web search engines**; **Information retrieval**; *Resource Description Framework (RDF).*

KEYWORDS

Question Answering, Semantic Web, Multilingual, Multi-Knowledge-Base

ACM Reference Format:
Dennis Diefenbach, Pedro Henrique Migliatti, Omar Qawasmeh, Vincent Lully, Kamal Singh, and Pierre Maret. 2019. QAnswer: A Question Answering prototype bridging the gap between a considerable part of the LOD cloud and end-users. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3308558.3314124

1 INTRODUCTION

In the last decade, some new datasets adhering to Semantic Web standards were published on the Web. This growth can be seen by looking at the Linked Open Data Cloud[1] (LOD cloud), which collects datasets that have been published using Semantic Web Technologies. Though in 2007, the LOD cloud contained 12 datasets, it now contains 1,231[2] datasets. The LOD cloud does not only contain a lot of datasets, but at the same time some of these datasets are very large. A dump of the LOD cloud called LOD-a-lot[3] contains around 5 TB of structured data in uncompressed N-triples format. While Semantic Web standards are designed to make data to be machine comprehensible, they do not allow, at the same time, easy accessibility of the data to non experts and in particular to end-users. Question Answering (QA) is seen as the technology that can allow to bridge this gap between Semantic Web data and end-users. While the industry presents QA solutions on their proprietary Knowledge Base (such as Google and Baidu), no system reached a point which allows querying a substantial part of the LOD cloud.
Table 1 shows a list of QA systems which we are aware of, that are available online and query part of the LOD cloud. We indicate the datasets which they are able to query and the languages which are supported. For a general list and an overview of QA systems over KB, we refer to [4]. QAnswer represents a breakthrough since it allows to query many more datasets, at the same time, in real-time and in many more languages. The main algorithm behind QAnswer is described in [3]. In the following text, we show how the algorithm in [3] was implemented to create a first prototype of a QA System over the Semantic Web and which are the improvements over previous works.

2 DESCRIPTION

The algorithm behind QAnswer is described in [3]. It was shown that it has the following distinctive features:

- **Multilingual**, it supports multiple languages. In the previous work, it was shown that the algorithm works for English, German, French, Italian, Spanish, Portuguese. Moreover, the algorithm can easily be adapted to new languages.
- **Robust**, users ask questions using keywords, natural language questions and even malformed questions, i.e., syntactically wrong questions. The algorithm is robust enough to deal with all these scenarios, but not to spelling mistakes.

[1] https://lod-cloud.net

[2] stand December 2018

[3] http://lod-a-lot.lod.labs.vu.nl

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314124

QA system	Lang	KBs	Url
Qakis[2]	en	DBpedia	http://qakis.org/
Quint[1]	en	Freebase	https://gate.d5.mpi-inf.mpg.de/quint/quint
AskNow[5]	en	DBpedia	http://asknowdemo.sda.tech
Frankenstein[8]	en	DBpedia	http://frankenstein.qanary-qa.com
gAnswer [10]	en	DBpedia	http://ganswer.gstore-pku.com/
Ask Platypus[9]	en, fr, es	Wikidata	https://askplatyp.us
QAnswer	en, fr, de, es, pt, zh, ar	Wikidata, LinkedGeoData, MusicBrainz, DBpedia, DBLP	http://qanswer.eu/qa

Table 1: List of Question Answering systems that query part of the LOD cloud and that are available online.

- **Real-time**, the algorithm can answer questions in real-time, i.e., on existing benchmarks like QALD and SimpleQuestions an average run-time of 2 seconds per query is realistic.
- **Low hardware footprint**, the algorithm uses specific indexes that guarantee low disk and memory footprint. Our demo will show that QAnswer can query 700Gb of n-triples and can be run on a (standard) laptop having 4 cores and 16 Gb of RAM.
- **Portable**, making question answering over a new dataset can be difficult. Some approaches need a lot of training data, other are not designed to be portable at all. Our system is designed such that any new dataset can be used as a base for a new QA system.
- **Multi-Knowledge-base**, the algorithm allows to query multiple Knowledge-bases at the same time.
- **Precision and Recall**, the algorithm was tested on multiple benchmarks and can compete with most of the existing approaches [3].

We are using the algorithm described in [3] to query what we believe are the three more significant datasets in the LOD cloud for end-users: Wikidata, LinkedGeoData and MusicBrainz. While, DBpedia and Freebase can also be queried, we consider these datasets as outdated since both are not maintained anymore.

We now describe the improvements over previous works brought by QAnswer.

- Non-european languages: While it was shown that the algorithm in [3] can be used to answer questions in multiple languages, it was only tested over European languages. Recently, we could also successfully apply it to two non-european languages, namely Chinese and Arabic. Example requests that can now be addressed are:



Figure 1: Screenshot of QAnswer for فيلم ذيب.

  - زوجه باراك اوباما, i.e., asking for the wife of Obama in Arabic.
  - "巴黎的博物馆", i.e., searching in Chinese for the museums in Paris.

  A screenshot of an example can be seen in Figure 1.

- LinkedGeoData: One of the largest open databases for geographical information is OpenStreetMap (OSM). It collects geographical information thanks to more than 2 million registered users. It contains information about streets, buildings, points of interest (like shops, restaurants, museums, fountains), cities, regions and many more. The data is natively stored in a PostgreSQL database with PostGIS extension. LinkedGeoData is the RDF extract of OSM data. The last public extraction was performed in 2015. We extracted a new export, covering entire Europe, and made the dataset queriable by QAnswer. Note that there is no online demo querying geo-spatial data and we are aware of only one work tackling this problem, which is presented in [7]. Some example requests that can be answered using this dataset are:
  - "Steinwenden", i.e. searching for a city
  - "Worms Renzstraße", i.e. searching for a street in a city
  - "give me fountains in Saarbrücken", i.e. searching for a point of interest in a city

  A screenshot of an example can be seen in Figure 2.

- Lexemes: In the previous work, the algorithm presented in [3] was used to query Wikidata. When Wikidata was new, it contained only Q-items, i.e., resources describing a thing or an idea, but not the word describing it. Since 2018, Wikidata was extended to also store new information such as words, phrases and sentences, in many languages. This information is stored in new types of entities, called Lexemes (L), Forms

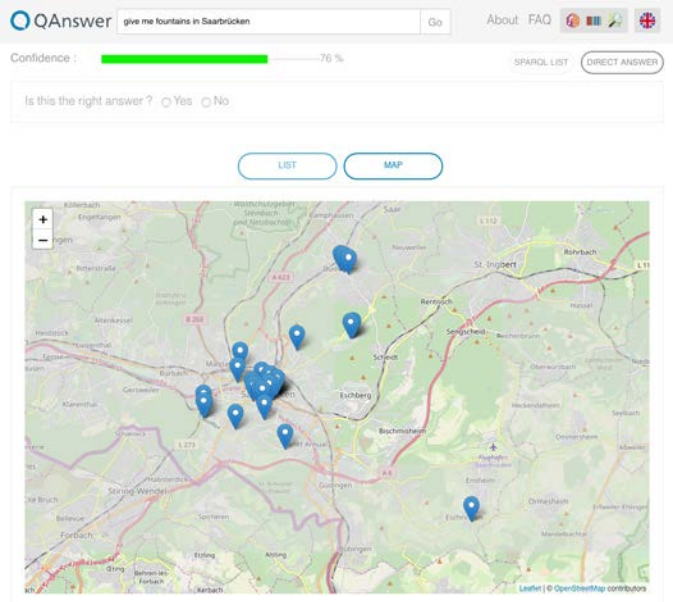


Figure 2: Screen shot of QAnswer for "give me fountains in Saarbrücken"

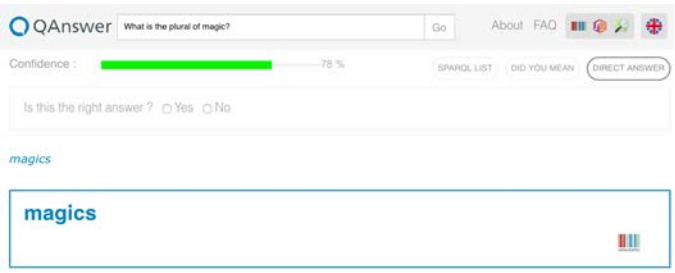


Figure 3: Screen shot of QAnswer for "What is the plural of magic?"

(F) and Senses (S). The Lexeme extension can be seen as a structured representation of the Wiktionary[4]. There exists an earlier attempt to semantify wiktionary by DBpedia project[5]. This project is unfortunately not maintained anymore and the extraction framework does not work anymore due to changes in the structure of Wiktionary. While the Lexemes in Wikidata still represent a small portion of the information contained in Wiktionary, we believe that the active Wikidata community will be able to fill this gap. Example questions that can be answered using this data are:

- "what is the pronunciation of magic?", i.e. searching for the pronunciation of a word
- "What is the plural of magic?", i.e., searching for forms of a word

A screenshot of an example can be seen in Figure 3.

[4]https://www.wiktionary.org
[5]https://wiki.dbpedia.org/wiktionary-rdf-extraction

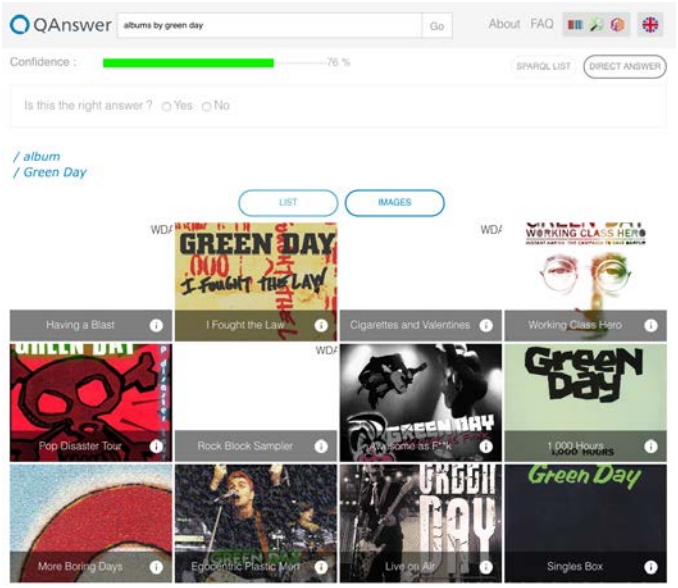


Figure 4: Screen shot of QAnswer for "albums by Green Day"

- Musicbrainz: Musicbrainz is one of the largest music databases that is open and available online. The data is natively stored in a relational database (i.e. a PostgreSQL database). In previous work, it was already possible to query MusicBrainz, but it relied on the dump provided by the following RML mappings: https://github.com/LinkedBrainz/MusicBrainz-R2RML. We improved the mappings in different ways. The most radical change was related to how some information is organized. For example in the original LinkedBrainz dump every album appears multiple times (every time for each publication year and country where it was released). The same problem appears for songs. In particular the dump didn't allow to recognize that the different publications of the albums and songs were in fact referring to the same album and song. This means that when asking for albums or songs, many duplicates appeared, a behaviour that is not expected by users and also not explicable for them. We therefore restructured the export to avoid this. Other changes represent a better coverage to external links like to Wikipedia, Wikidata and Social Media. Moreover, the dump is enriched with the links to the covers of albums from http://coverartarchive.org. Finally, we enriched the artists, albums and songs with links to youtube so that users can effectively hear the pieces they are querying. Example requests where Musicbrainz is used to answer some queries are:
  - "albums by green day", i.e., searching for albums of a band
  - "record label from turin", i.e. searching for record labels in a city,
  - "songs blink-182", i.e., searching for songs of a band

  A screenshot of an example can be seen in Figure 4.

- Qwant: While the Semantic Web grew very fast in the last years, it contains only a fraction of the information contained

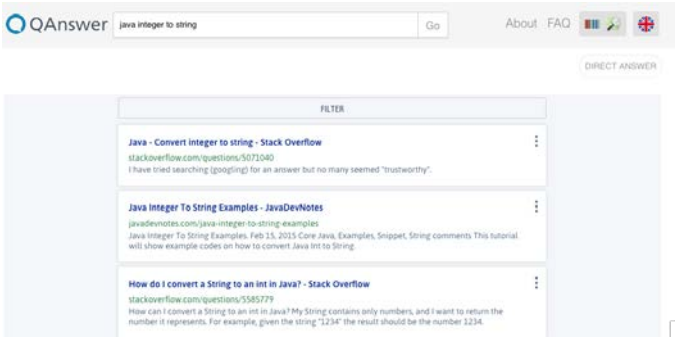


Figure 5: Screen shot of QAnswer for "java integer to string"

in the web. Whereas, most of the information is stored in a non-structured format, i.e., in the form of HTML pages. This information is typically accessed with traditional information retrieval techniques [6].

We therefore integrated, a fall back option, which is a traditional search engine, namely Qwant[6]. This means that when we cannot find the information requested by the users in one of the underlying datasets, we search for an answer using Qwant. Note that this represents a new problem in the area of QA over Knowledge Bases. Current benchmarks only contain questions whose answer can be found in the underlying knowledge base. There are a few exceptions in the QALD benchmark where less than 1% of the questions are not answerable. However, in a real scenario many of the questions asked by an end-user are not contained in the underlying Knowledge-Base. This means that recognizing when a question should not be answered (because it is not answerable considering the underlying knowledge-base) is an important and not well studied problem. Technically speaking this comes down to the fact that in current benchmarks only the macro F-measure is evaluated and the micro F-measure is generally ignored.

We chose Qwant as a fall-back option since it does not rely on a Knowledge Graph to provide direct answers. Moreover, the main distinctive feature as compared to other existing search engines is that Qwant does not track users and does not personalize search results and therefore users are not trapped in a filter bubble. Example requests where Qwant is used as a fall-back are:

- "How many legs has a horse?"
- "With how many degrees should I cook a pizza?"
- "java integer to string"

A screenshot of an example can be seen in Figure 5.

3 DEMO

A demo of the current version can be found under:

http://qanswer.eu/qa

Moreover, by clicking on the above examples you will be redirected to the online demo.

[6]https://www.qwant.com

4 CONCLUSION

We presented QAnswer a QA system which queries 3 key Semantic Web datasets at the same time, namely, Wikidata with lexemes, LinkedGeoData and Musicbrainz. These datasets contain a huge amount of information like books, films, persons, music, streets, points of interest and many more. The data can be queried using natural language in 8 different languages, namely English, German, French, Italian, Spanish, Pourtuguese, Arabic and Chinese. In particular two non-European languages, Arabic and Chinese, are included. All together this represents a first prototype of a QA system querying a considerable part of the Semantic Web.

This represents also a step towards new challenges in this area like: correctly selecting which Knowledge Base to query, new scalability scenarios, making use of links between the datasets to deal with redundant information, studying the impact of the dataset quality on the QA performance, and studying the problem of not answering a question.

In future, we would like to make the algorithm and infrastructure publicly available via web APIs so that new RDF datasets can be indexed and accessed using natural language. We believe that this work will further boost the publication of RDF data and therefore the expansion of the Semantic Web.

REFERENCES

[1] Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2017. QUINT: Interpretable Question Answering over Knowledge Bases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 61–66.

[2] Elena Cabrio, Julien Cojan, Alessio Palmero Aprosio, Bernardo Magnini, Alberto Lavelli, and Fabien Gandon. 2012. QAKiS: an open domain QA system based on relational patterns. In *Proceedings of the 2012th International Conference on Posters & Demonstrations Track-Volume 914*.

[3] Dennis Diefenbach, Andreas Both, Kamal Deep Singh, and Pierre Maret. 2018. Towards a Question Answering System over the Semantic Web. *Semantic Web Journal* (2018).

[4] Dennis Diefenbach, Vanessa López, Kamal Deep Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowl. Inf. Syst.* 55, 3 (2018), 529–569. https://doi.org/10.1007/s10115-017-1100-y

[5] Mohnish Dubey, Sourish Dasgupta, Ankit Sharma, Konrad Höffner, and Jens Lehmann. 2016. AskNow: A Framework for Natural Language Query Formalization in SPARQL. In *ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Proceedings*. 300–316. https://doi.org/10.1007/978-3-319-34129-3_19

[6] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*.

[7] Dharmen Punjani, Ellie Nagaishi, Andreas Both, Manolis Koubarakis, Ioannis Angelidis, Konstantina Bereta, Themis Beris, Dimitris Bilidas, Kelly A. Martin, Nikolaos Karalis, Christian Lange, D Pantazi, Costas Papaloukas, and George Stamoulis. 2018. Template-Based Question Answering over Linked Geospatial Data. In *GIR'18*.

[8] Kuldeep Singh, Arun Sethupat Radhakrishna, Andreas Both, Saeedeh Shekarpour, Ioanna Lytra, Ricardo Usbeck, Akhilesh Vyas, Akmal Khikmatullaev, Dharmen Punjani, Christoph Lange, et al. 2018. Why Reinvent the Wheel: Let's Build Question Answering Systems Together. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1247–1256.

[9] Thomas Pellissier Tanon, Marcos Dias de Assunção, Eddy Caron, and Fabian M. Suchanek. 2018. Demoing Platypus - A Multilingual Question Answering Platform for Wikidata. In *The Semantic Web: ESWC 2018 Satellite Events - ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers*. 111–116. https://doi.org/10.1007/978-3-319-98192-5_21

[10] Lei Zou, Ruizhe Huang, Haixun Wang, Jeffrey Xu Yu, Wenqiang He, and Dongyan Zhao. 2014. Natural language question answering over RDF: a graph data driven approach. In *SIGMOD 2014, Snowbird, UT, USA*. 313–324. https://doi.org/10.1145/2588555.2610525

Voyageur: An Experiential Travel Search Engine

Sara Evensen
Megagon Labs
sara@megagon.ai

Aaron Feng
Megagon Labs
aaron@megagon.ai

Alon Halevy
Megagon Labs
alon@megagon.ai

Jinfeng Li
Megagon Labs
jinfeng@megagon.ai

Vivian Li
Megagon Labs
vivian@megagon.ai

Yuliang Li
Megagon Labs
yuliang@megagon.ai

Huining Liu
Megagon Labs
yuliang@megagon.ai

George Mihaila
Megagon Labs
george@megagon.ai

John Morales
Megagon Labs
john@megagon.ai

Natalie Nuno
Megagon Labs
natalie.nuno@megagon.ai

Ekaterina Pavlovic
Megagon Labs
kate@megagon.ai

Wang-Chiew Tan
Megagon Labs
wangchiew@megagon.ai

Xiaolan Wang
Megagon Labs
xiaolan@megagon.ai

ABSTRACT

We describe Voyageur, which is an application of experiential search to the domain of travel. Unlike traditional search engines for online services, experiential search focuses on the experiential aspects of the service under consideration. In particular, Voyageur needs to handle queries for subjective aspects of the service (e.g., quiet hotel, friendly staff) and combine these with objective attributes, such as price and location. Voyageur also highlights interesting facts and tips about the services the user is considering to provide them with further insights into their choices.

CCS CONCEPTS

• **Information systems** → **Search interfaces**; **Structured text search**; *Information retrieval*; • **Computing methodologies** → **Information extraction**; *Natural language processing*.

KEYWORDS

experiential search; subjective database; linguistic domain; information extraction;

ACM Reference Format:
Sara Evensen, Aaron Feng, Alon Halevy, Jinfeng Li, Vivian Li, Yuliang Li, Huining Liu, George Mihaila, John Morales, Natalie Nuno, Ekaterina Pavlovic, Wang-Chiew Tan, and Xiaolan Wang. 2019. Voyageur: An Experiential Travel Search Engine. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314125

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314125

1 INTRODUCTION

The rise of e-commerce enables us to plan many of our future experiences with online search engines. For example, sites for searching hotels, flights, restaurants, attractions or jobs bring a wealth of information to our fingertips, thereby giving us the ability to plan our vacations, restaurant gatherings, or future career moves. Unfortunately, current search engines completely ignore any experiential aspect of the plans they are helping you create. Instead, they are primarily database-backed interfaces that focus on searching based on objective attributes of services, such as price range, location, or cuisine.

The need for experiential search. An experiential search engine is based on the observation that at a fundamental level, users seek to satisfy an experiential need. For example, a restaurant search is meant to fulfill a social purpose, be it romantic, work-related or a reunion with rowdy friends from college. A vacation trip plan is meant to accomplish a family or couple need such as a relaxing location with easy access to fun activities and highlights of local cuisine. To satisfy these needs effectively, the user should be able to search on experiential attributes, such as whether a hotel is romantic or has clean rooms, or whether a restaurant has a good view of the sunset or has a quiet ambience.

Table 1 shows a snippet from a preliminary study that investigates which attributes users care about. We asked human workers on Amazon Mechanical Turk [1] to provide the most important criteria in making their decisions in 7 common verticals. We then conservatively judge whether each criterion is an experiential one or not. As the table shows, the majority of attributes of interest are experiential.

To the best of our knowledge, online services for these domains today lack support for directly searching over experiential attributes. The gap between a users' experiential needs and search

Table 1: Experiential attributes in different domains.

Domain	%Exp. Attr	Some examples
Hotel	69.0%	cleanliness, food, comfortable
Restaurant	64.3%	food, ambiance, variety, service
Vacation	82.6%	weather, safety, culture, nightlife
College	77.4%	dorm quality, faculty, diversity
Home	68.8%	space, good schools, quiet, safe
Career	65.8%	work-life balance, colleagues, culture
Car	56.0%	comfortable, safety, reliability

capabilities raises an important challenge: *can we build search systems that place the experience the user is planning at the center of the search process?*

Challenges. Supporting search on experiential aspects of services is challenging for several reasons. First, the universe of experiential attributes is vast, their precise meaning is often vague, and they are expressed in text using many linguistic variations. The experience of "quiet hotel rooms" can be described simply as "quiet room" or "we enjoyed the peaceful nights" in hotel reviews. Second, by definition, experiential attributes of a service are *subjective* and personal, and database systems do not gracefully handle such data. Third, the experiential aspect of a service may depend on how they relate to other services. For example, a significant component of a hotel experience is whether it is close to the main destinations the user plans to visit. Finally, unlike objective attributes that can be faithfully provided by the service owner, users expect that the data for experiential attributes come from other customers. Currently, such data is expressed in text in online reviews and in social media. Booking sites have made significant efforts to aggregate and surface comments from reviews. Still, while these comments are visible when a user inspects a particular hotel or restaurant, users still cannot search by these attributes.

This paper describes Voyageur, the first search engine that explicitly models experiential attributes of services and supports searching them. We chose to build Voyageur in the domain of travel because it is complex and highly experiential, but its ideas also apply to other verticals.

The first idea underlying Voyageur is that the experiential aspects of the service under consideration need to be part of the database model and visible to the user. For example, when we model a hotel, we'll also consider the time it takes to get there from the airport and nearby activities that can be done before check-in in case of an early arrival. Furthermore, Voyageur will fuse information about multiple services. So the proximity of the hotel from the attractions of interest to the user is part of how the system models a hotel. Of course, while many of the common experiential aspects can be anticipated in advance, it is impractical that we can cover them all. Hence, the second main idea in Voyageur is that its schema should be easily extensible, it should be able to handle imprecise queries, and be able to fall back on unstructured data when its database model is insufficient.

Even with the above two principles, Voyageur still faces the challenge of selecting which information to show to the user about a particular entity (e.g., hotel or attraction). Ideally, Voyageur should display to the user aspects of the entity that are most relevant to the decision she is trying to make. Voyageur includes algorithms for discovering items from reviews that best summarize an entity, highlight the most unique things about them, and useful actionable tips.

2 OVERVIEW OF VOYAGEUR

We illustrate the main ideas of Voyageur. Specifically, we show how Voyageur supports experiential queries and how these queries assist a user in selecting a hotel.

User scenario. Elle Rios is a marketing executive living in Tokyo. She is planning a vacation to *San Francisco* in early October. Her goal is to have a relaxing experience during the vacation. Her entire travel experience will be influenced by a variety of services, including flights, hotels, local attractions, and restaurants. Elle visits the Voyageur website and first enters the destination with the travel period. Voyageur then displays a series of screens with recommendations for each of these services.

In searching for hotels, Elle's experiential goal to have a relaxing stay is achieved by a balance of her *objective* constraints (her budget for hotels is *$250-350 per night*) and *subjective* criteria; she is an introvert and she knows that *quiet* hotels with *friendly staff* will help her relax[1]. In addition, Elle also cares about whether the hotel is conveniently located for reaching the famous and historic attractions she wants to visit and the high-quality vegetarian restaurants she is interested in.

Figure 1 shows a screenshot of Voyageur, where Elle can plan a trip that satisfies her requirements. (The screens for attraction and restaurant search are similar). The screenshot shows how Voyageur emphasizes the experiential aspects of trip planning. First, Voyageur allows users to express their subjective criteria (in addition to their objective criteria) and generates recommendations accordingly through the *experiential search* function (Box A). Second, Voyageur tailors the display of *interesting facts and tips* and *summary of reviews* based on the search criteria entered (Boxes B and C). Third, Voyageur supports a series of additional features, such as map view and travel wallet, to further improve the user's experience in the hotel search (Box D). The map view provides a holistic view of the trip by putting together all recommended entities of different types. The travel wallet, as we explain later, takes into consideration the user's travel history and preferences if the user chooses to share them with Voyageur.

Experiential search. Elle expresses her objective and experiential/subjective criteria as query predicates to Voyageur's interface. While the objective requirements like "$250 to 300 per night" can be directly modeled and queried in a typical hotel database, answering predicates like *"quiet"* and *"friendly staff"* is challenging as these are subjective terms and cannot be immediately modeled in a traditional database system. Voyageur addresses this challenge with a subjective database engine that explicitly models the subjective attributes and answers subjective query predicates. Voyageur extracts subjective attributes such as *room quietness* and *staff quality* from hotel customer reviews, builds a summary of the variations of these terms, and then matches those attributes with the input query predicates.

[1]Elle can also directly search for hotels with relaxing atmosphere in Voyageur.

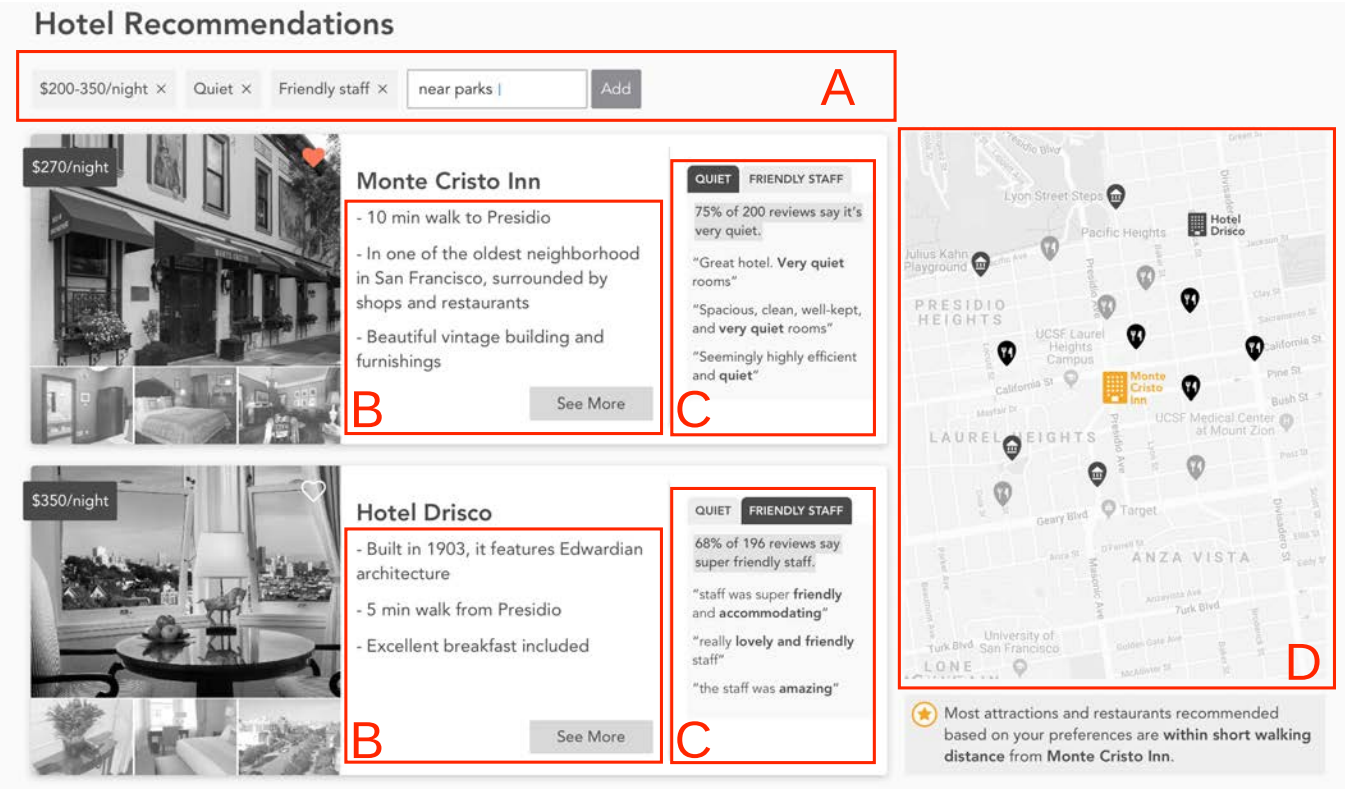


Figure 1: A screenshot of the hotel recommendation screen of Voyageur. The user can interactively customize the search (Box A), to view interesting facts/tips (Box B) and a *review summary* (Box C) of each recommended hotel and to check out a *map view* (Box D) of the recommendations.

In Figure 1, Voyageur generates a ranked list of hotels by matching the query predicates specified by Elle with the subjective attributes extracted from the underlying hotel reviews. The review summaries (Box C) show that the selected hotels are clearly good matches. Specifically, Voyageur recommends Monte Cristo since 75% of 200 reviewers agree that it is very quiet and it has friendly staff (not shown). Hotel Drisco, next on the list, is recommended because 68% of 196 reviewers agree that it has friendly staff and the it is also very quiet (not shown).

Interesting facts and tips. Along with the search results, Voyageur shows snippets of travel tips and/or interesting facts of each result (Box B) it thinks is relevant for Elle. An interesting fact typically highlights an unusual or unique experience about the service. For example, being very close to Presidio Park (one of the largest parks in San Francisco) is unique to Monte Cristo Inn and Hotel Drisco and is thus an interesting fact to show for each hotel. The fact that Monte Cristo Inn is a "beautiful vintage building and furnishings" is unique only to Monte Cristo Inn. Such interesting facts can be important for decision making. It also enables Elle to better anticipate the type of experiences that will be encountered at the hotel [8]. On the other hand, tips are snippets of information that propose a potential action the user may take to either avoid a negative experience or create a positive one [2]. For example, a useful tip for a hotel may be that there is free parking two blocks away.

While the interesting facts and tips are useful, they are not always available for every service and can be incomplete. Existing work [2, 6] proposed mining useful travel tips from customer reviews with promising results. In Voyageur, we formulate the problem of finding tips and interesting facts as a query-sentence matching problem to find tips and interesting facts relevant to the users' query. Our algorithms prefer to select sentence snippets from reviews to match the user's query. The challenge we face is that users' query predicates and the tips/facts in reviews are described in different vocabularies and linguistic forms. Moreover, labeled data for the matching task is generally not available. Thus, novel techniques are needed to construct good matching functions between queries and sentence snippets.

Review summarization. The summary of reviews (Box C) provides Elle with an explanation of why the specific hotel is recommended and the summary saves her from reading the repetitive and lengthy reviews. Voyageur summarizes the reviews of each recommended hotel in two different formats: (1) statistical statements and (2) sample review snippets. For example, in Figure 1, Voyageur summarizes the room quietness attribute of Monte Cristo Inn with the statistical statement "75% of 200 reviews say it is very quiet" and 3 randomly selected sample review snippets that match the *quiet* requirement.

Additional features. The following features further improve Elle's ability to create a positive travel experience:

- **Map view.** In each recommendation screen, a map view (Box D of Figure 1) marks the locations of the recommended

hotels. Whenever a hotel is selected, the map view is centered at a chosen hotel and shows the recommended local attractions and restaurants so the user can better plan how to travel between these places.

- **Travel wallet.** Users have the option of creating a *travel wallet*, which is similar to the Wallet feature on many smartphones. It contains information about the user that she shares only when she chooses to. In the case of a travel wallet, this information records her travel preferences. The travel wallet can be created explicitly by answering questions or can be collected automatically from previous travels. The travel wallet is used by Voyageur to further personalize the search results.
- **Trip summary.** Finally, after making several choices (flight, hotel, attractions, etc.) through all the recommendation screens, the user can view a summary of the trip underlying the key experiential components. In Elle's case, the summary includes a timeline with important dates, transportation methods to/from the airport, and tips/facts about the chosen hotel and each planned tourist attractions.

3 IMPLEMENTATION OF VOYAGEUR

We briefly touch upon the technology underlying Voyageur.

3.1 A subjective database engine

As mentioned, the main challenge of building a successful experiential search engine is the modeling and querying of the *subjective attributes*, where there is typically no ground truth to the values of such attributes. Examples of such attributes include the cleanliness of hotel rooms, quality of the food served, and cultural value of tourist attractions. They are not explicitly modeled in today's search engines and therefore not directly queryable.

Voyageur is developed on top of OpineDB [3], a *subjective database engine*. OpineDB goes beyond traditional database engines by supporting the modeling, extraction, aggregating, and effective query processing of the subjective data. Next, we illustrate the key design elements of OpineDB by showcasing its application to hotel search in Voyageur.

Data model, extraction, and aggregation. The main challenge in modeling subjective attributes is the wide range of linguistic variations with which the attributes are described in text. Consider an attribute `room_quietness` of hotels. The review text can be of various forms such as (1) "*the neighborhood seems very quiet at night*", (2) "*on busy street with traffic noise*", or (3) "*quiet and peaceful location*". In addition, OpineDB needs to *aggregate* these phrases into a meaningful signal for answering queries, which may themselves include new linguistic variations.

OpineDB models subjective attributes with a new data type called the *linguistic domain* and provides an aggregated view of the linguistic domain through a *marker summary*. The linguistic domain of an attribute contains all phrases for describing the attribute from the reviews. E.g., "quiet at night", "traffic noise", and "peaceful location". A subset of phrases is then chosen as the *domain markers* (or *markers* for short) for each linguistic domain. The phrases are aggregated based on the markers to constitute the marker summary.

For quietness, the markers might be [very_quiet, average, noisy, very_noisy]. To construct the marker summary of a hotel's quietness, OpineDB needs to assign each quietness phrase to its closest marker and compute the frequencies of the markers. For example, the summary [very_quiet:20, average: 70, noisy:30, very_noisy:10] for a hotel would represent that the hotel is closer to being `average` in quietness than to the other markers.

The linguistic domain is obtained by extracting phrases from reviews. Various techniques are available for this task in opinion mining and sentiment analysis [4, 7]. The marker summaries are currently histograms computed from the extraction relations. However, we can also leverage more complex aggregate functions.

Query processing. The query predicates from Box A is formulated as an SQL-like query for OpineDB to process.

```
select  *  from Hotels h
where   price_pn ≤ 350 and price_pn ≥ 200 and
        "quiet" and "friendly staff"
```

Here, price_pn is an objective attribute of the Hotels relation while "*quiet*" and "*friendly staff*" are subjective predicates. OpineDB needs to interpret these predicates using the linguistic domains in order to find the best subjective attributes of the Hotels relation that can be used to answer them. In general, this is not a trivial matching problem since the query terms may not be directly modeled in the schema. For example, the user may ask for "romantic hotels", but the attribute for romance might not be in the schema. For such cases, OpineDB leverages a combination of NLP and IR techniques to find a best-effort reformulation of the query term into a combination of schema attributes. For example, for "romantic hotels", OpineDB will match it to a combination of "exceptional service" and "luxurious bathrooms" which are modeled by the schema.

After computing the interpretation, OpineDB uses the marker summaries to compute a *membership score* for each pair of hotel and query predicate. Finally, OpineDB combines multiple predicates using a variant of fuzzy logic.

3.2 Mining interesting facts and tips

We formulate the problem of finding useful travel tips and interesting facts as a query-sentence matching problem. We adopt an approach similar to an existing work for mining travel tips from reviews [2]. The approach consists of a *filtering phase* and a *ranking phase*.

Filtering phase. This phase constructs a set of candidate tip/fact sentences by applying filters and classifiers to all the review sentences. According to [2], effective filters for tips include phrase patterns (e.g, sentences containing "make sure to") and part-of-speech tags (e.g, sentences starting with verbs). For constructing the candidate set of interesting facts, we select sentences that contain a least one *informative token*, which are words or short phrases frequently mentioned in reviews of the target entity but not frequently mentioned in reviews of similar entities. We also found that an interesting fact is more likely to appear in sentences with an extreme sentiment (very positive or negative). So we also apply sentiment analysis to select such sentences. For both tips and interesting facts, we further refine the candidate sets by removing duplicates, i.e., sentences of similar meaning or the unimportant

ones. We do so by applying TextRank [5], a classic algorithm for sentence summarization.

Ranking phase. Instead of simply selecting candidates for interesting facts/tips, sets, we implemented a novel ranking function for finding candidates that best match the user's query predicates. The ranking function considers not only the *significance* of a candidate as computed in the filtering phase but also the *relevance* of the candidate with the query. Measuring the relevance is not trivial since the tips/interesting facts can use vocabularies different from the ones used in the query. In the previous example, a fact that matches the query "near park" is "10 min walk to Presidio" which has no exact-matched word. The similarity function leverages a combination of NLP and IR techniques, analogous to query interpretation in OpineDB.

3.3 Datasets and tools

The Voyageur demo will serve hotels, attractions, and restaurants search in the San Francisco area in a browser-based web app. We collected the underlying data from two datasets: the Google Local Reviews[2] and the TripAdvisor datasets[3]. Our dataset consists of 18,500 reviews of 227 hotels, 6,256 reviews of 545 attractions and 67,382 reviews of 4,007 restaurants. We implemented the front end of Voyageur using the JavaScript library React.

4 CONCLUSION

The motivation for Voyageur is based on the discrepancy between the needs of users searching for services and the current state of search engines. The ideas of Voyageur are applicable to many other verticals beyond travel. At the core of the technical challenges that Voyageur and systems like it need to address is the ability to discover and aggregate evidence from textual reviews in response to user queries. This is a technical challenge that draws upon techniques from NLP, IR and Database technologies.

REFERENCES

[1] Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. 2011. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data? *Perspectives on psychological science* 6, 1, 3–5.

[2] Ido Guy, Avihai Mejer, Alexander Nus, and Fiana Raiber. 2017. Extracting and Ranking Travel Tips from User-Generated Reviews. In *WWW*. 987–996.

[3] Yuliang Li, Aaron Xixuan Feng, Jinfeng Li, Saran Mumick, Alon Halevy, Vivian Li, and Wang-Chiew Tan. 2019. Subjective Databases. *CoRR* abs/1902.09661.

[4] Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool.

[5] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In *EMNLP*. 404–411.

[6] Sapna Negi, Maarten de Rijke, and Paul Buitelaar. 2018. Open Domain Suggestion Mining: Problem Definition and Datasets. *CoRR* abs/1806.02179.

[7] Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, AL-Smadi Mohammad, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. 2016. SemEval-2016 task 5: Aspect based sentiment analysis. In *SemEval-2016*. 19–30.

[8] Jonathan Skinner and Dimitrios Theodossopoulos. 2011. *Great expectations: Imagination and anticipation in tourism*. Vol. 34. Berghahn Books.

[2] http://cseweb.ucsd.edu/~jmcauley/datasets.html#google_local
[3] http://nemis.isti.cnr.it/~marcheggiani/datasets/

Tracy: Tracing Facts over Knowledge Graphs and Text

Mohamed H. Gad-Elrab
Max-Planck Institute for Informatics
Saarland Informatics Campus, Germany
gadelrab@mpi-inf.mpg.de

Daria Stepanova
Bosch Center for Artificial Intelligence
Renningen, Germany
daria.stepanova@de.bosch.com

Jacopo Urbani
Vrije Universiteit Amsterdam
Amsterdam, The Netherlands
jacopo@cs.vu.nl

Gerhard Weikum
Max-Planck Institute for Informatics
Saarland Informatics Campus, Germany
weikum@mpi-inf.mpg.de

ABSTRACT

In order to accurately populate and curate Knowledge Graphs (KGs), it is important to distinguish $\langle s\ p\ o\rangle$ facts that can be traced back to sources from facts that cannot be verified. Manually validating each fact is time-consuming. Prior work on automating this task relied on numerical confidence scores which might not be easily interpreted. To overcome this limitation, we present Tracy, a novel tool that generates human-comprehensible explanations for candidate facts. Our tool relies on background knowledge in the form of rules to rewrite the fact in question into other easier-to-spot facts. These rewritings are then used to reason over the candidate fact creating semantic traces that can aid KG curators. The goal of our demonstration is to illustrate the main features of our system and to show how the semantic traces can be computed over both text and knowledge graphs with a simple and intuitive user interface.

KEYWORDS

Knowledge Graph; Fact-checking; Explainable Evidence; Reasoning

ACM Reference Format:
Mohamed H. Gad-Elrab, Daria Stepanova, Jacopo Urbani, and Gerhard Weikum. 2019. Tracy: Tracing Facts over Knowledge Graphs and Text. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314126

1 INTRODUCTION

Motivation and Problem. Knowledge Graphs (KGs) are repositories of factual knowledge in the form of $\langle s\ p\ o\rangle$ facts where s, o are entities and p specifies a semantic relation between them, e.g., $\langle$*London capitalOf UK*$\rangle$. Currently, a large number of KGs are publicly available on the Web (e.g., Wikidata [16], YAGO [15]) but some of them might contain doubtful if not incorrect facts as they are partly built by automatic information extraction, crowd-sourcing, or other noisy methods [12, 18]. Due to the increasing usage of KGs for tasks like query-answering, dialog systems, etc., it is important to validate each candidate fact to ensure the correctness of the KG.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314126

This task, which is often referred to as *fact-checking* or *truth discovery* [8], can be performed manually by KG curators but this is time-consuming. Therefore, the automation of this process is gaining more attention. Existing methods for automatic fact-checking (e.g., [5, 8, 9, 11, 13]) usually proceed in two steps as follows: First, they search for explicit mentions of the fact in the Web sources like news or other textual corpora. For example, for a fact such as $\langle$*Sadiq_Khan citizenOf UK*$\rangle$, they search for *"khan is a citizen of UK"* or *"Khan's nationality is British"*. Then, the extracted evidence is used to infer whether the candidate fact can indeed be verified.

Existing approaches have two main limitations: First, they quantify the confidence using numerical scores, which is not adequate in case the final decision is made by KG curators. Indeed, such scores are hard to understand or justify without explanations. Only few approaches (*e.g.*, [1, 5]) attempt to explain the results. For example, Defacto [5] shows the sources used in computing the scores as an explanation and [1] reports a comparison between computed scores to explain its final decision for humans. Second, searching for explicit mentions is often not sufficient since textual sources are *incomplete* and *biased* in what is stated explicitly. For instance, the citizenship of London's mayor Sadiq Khan would rarely be mentioned. Moreover, some predicates (*e.g.*, *influencedBy*) are ambiguous and their interpretation is domain-specific.

Proposed Approach. We introduce Tracy, a tool designed to support KG curators in deciding the correctness of the candidate facts. The main novelty of our tool consists of finding *semantically related evidence* in textual sources and the underlying KG, and providing *human-comprehensible explanations* for the facts. In Tracy, the semantically related evidence is extracted according to intentional background knowledge given as rules of the form $H \leftarrow B_1, B_2, \ldots, B_n$ which can be either specified by humans or automatically extracted using rule mining methods [14] By utilizing rules, Tracy enables users to combine clues from different resources (structured and unstructured) thus overcoming the problem that arises when the fact is never explicitly mentioned. For example, consider the rule

$$citizenOf(X, Y) \leftarrow mayorOf(X, Z), locatedIn(Z, Y)$$

which intuitively states that mayors of cities are normally citizens of countries where these cities are located. This rule can be used by our tool to verify the fact $\langle$*Sadiq_Khan citizenOf UK*$\rangle$ by searching in news articles whether he is a major of some city and then looking up the city's corresponding country in the KG. The combination

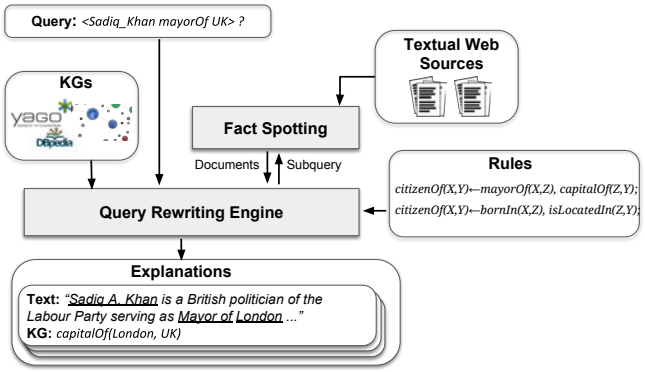


Figure 1: Tracy system overview.

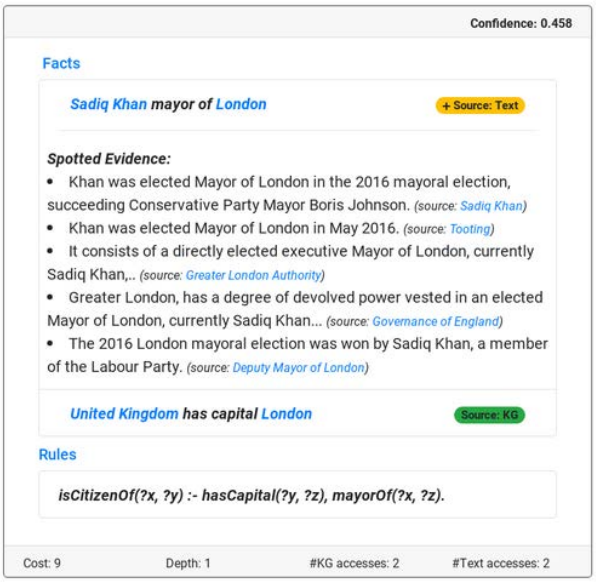


Figure 2: Example explanation returned by Tracy.

of these two semantically-related facts allows us to construct a human-readable explanation for the correctness of the fact.

As illustrated by the above example, Tracy uses the rules to decompose the spotting of an input fact into more frequently stated and thus easier-to-spot related facts. The search for evidence for the rule's body might trigger the execution of other rules, and this effectively creates semantic traces that *explain*, in a human-readable format, why a fact is likely to be true. These traces are presented to a KG curator in a comprehensible format in order to make a final decision about the truth value of the fact in question.

Outline. In the remaining, we briefly describe the framework underlying Tracy and the realization of the main components. Afterwards, we illustrate the features offered by the demonstrated graphical user interface, allowing users to experience different usage scenarios. Our demo is made available under the following link https://www.mpi-inf.mpg.de/impact/exfakt#Tracy.

2 SYSTEM OVERVIEW

Tracy follows ExFaKT framework introduced in our earlier work [3]. Tracy implements two main components: (i) *Query Rewriting Engine (QRE)*, which is responsible for rewriting facts and generating explanations, and (ii) *Fact Spotting (FS)*, which is the component that finds evidences in Web sources. Fig. 1 shows a graphical representation of this architecture.

Our framework receives as input a candidate fact (*i.e.*, the query), a set of rules, and two knowledge sources: (i) a KG, which we view as a *reliable* source; and (ii) an unstructured collection of text corpora, which we consider as an *unreliable* source, since its extractions by *FS* might be noisy.

Rules are Horn clauses of the form of $H \leftarrow B_1, \ldots, B_n$ where H is the rule's head and $B_1, \ldots, B_n$ is the body of the rule. In general, rules can be automatically extracted from KGs using systems such as [2, 4, 6, 17]. However, extracted rules are restricted to the predicates in the KG. Thus, Tracy also accepts manually specified rules with user-defined predicates, which later can be spotted in the text.

The workflow of our system is as follows: First, a user submits an input fact Q (*i.e.*, the query) to *QRE*. Then, this component possibly rewrites it into easier facts according to the logic specified by the rules contacting *FS* every time it cannot find evidences in the KG. If a sufficient number of evidences is found, then QRE uses them to return one or more explanations for Q.

2.1 Query Rewriting Engine

The process starts by considering rules, whose head predicates are the same as in the query. Then, it visits the rules' bodies and effectively rewrites the query into additional subqueries. The subqueries might trigger new rules and this results in a recursive process until all subqueries are answered. Such rule-based evaluation strategy is well-known in logic programming and commonly used for query answering. Indeed, our system implements an adaptation of a set-based version of standard SLD resolution [7].

Our adaption consists of three modifications which are crucial for our scenario of interest: First, we impose a maximum depth onto the recursive process to ensure termination. Second, the system attempts to answer the query consulting *FS* whenever it cannot be answered relying on the KG. Third, we care about the quality of the sources; and hence, we still rewrite the fact if the evidence source is not trustful seeking more reliable evidences. These two last modifications are particularly important: In fact, it is well-known that KGs are highly incomplete and a conventional query answering engine would simply fail when evidence is not found. In contrast, the ability of our system to also include extractions from additional sources overcomes this problem and enables the construction of explanations even when the content of the KG is not sufficient.

Once this process is finished, the answers (either from the KG or Web sources) are used to compute the explanations. This is done by revisiting the rules bottom-up. Fig. 2 shows an extract of an example explanation provided by Tracy. In this case, the explanation illustrates that the fact that *Sadiq Khan is a citizen of UK* is likely to be true according to the rule

$$isCitizenOf(X, Y) \leftarrow mayorOf(X, Z),\ hasCapital(Y, Z)$$

and additional evidence found both in the KG and textual corpora.

Note that a given query might have multiple complex explanations or a single *trivial* one, i.e., the fact itself. Moreover, some explanations may be subsumed by others. Ideally, we aim at computing non-trivial explanations that are (i) **concise** with a small number of atoms; (ii) **close to the query**, i.e., obtained by using few rules; (iii) **reliable**, i.e., contain as many facts from the KG as possible, since KGs are usually more reliable than text.

In order to recognize these explanations, we compute a confidence score for each explanation E, defined as:

$$confidence(E) = \frac{1}{|E|} \sum_{a \in E} \frac{trust(sources[a])}{depth[a]}$$

where a is the atom of the explanation E. This confidence score is directly correlated with the quality of the sources containing a ($trust(sources[a])$) and inversely correlated with the depth of the rewriting performed to reach the atom a (*i.e.*, the number of used rewriting rules) and the number of distinct atoms in the explanation.

2.2 Fact Spotting

Tracy follows a modular design allowing integrating any fact-spotting method. As default, we implemented a dictionary based fact-spotting procedure similar to [9, 13]. It is a simple and highly scalable method that does not require training. The main idea consists of first converting the SPO query into a textual representation (*i.e.*, verbalization) using a paraphrasing dictionary. Then, queries with the paraphrases are issued to a spotting engine to retrieve documents that mention them.

We used two types of dictionaries, depending on whether the used KG is YAGO or Wikidata (these are the only two which are currently supported). With YAGO, we use the paraphrases learned by PATTY [10] after some manual filtering. With Wikidata, we exploit the name aliases included in the KG.

We also implemented two types of spotting engines, one that uses a *local* text corpus and another one that consults a *remote* one:

- **Local corpus:** We used Elasticsearch [1] to index all Wikipedia articles. As a prepossessing step, we filtered the textual parts in the articles by removing semi-structured parts such as tables and info-boxes. Then, we index the article sentences separately along with the title. For spotting an SPO query, we issue a boolean query with the paraphrases of each part of the SPO separately. Then, we collect the *top-5* matching sentences as evidence for the query.
- **Remote corpus:** We also provide the option for searching the Web using Bing Web search API [2]. To query the API, we compose a string query containing all possible paraphrases of SPO and use the *top-5* search results as evidence.

2.3 Experimental Results

To illustrate the potential of our system, we report the results of an experiment using the YAGO KG. We simulated the case where we need to verify 300 unseen candidate facts, uniformly distributed over six different relations *influences*, *isPoliticianOf*, *wroteMusicFor*, *mayorOf*, *actedWith*, and *countryWonPrize*. A rule set was compiled for each predicate by selecting the top-ranked rules mined by AMIE [4] and adding other manually created rules involving new predicates that do not exist in YAGO.

Table 1: Direct fact-spotting vs Tracy explanations

	Textual Source	Recall	Prec@5	F1@5
Direct-spotting	Wikipedia	0.25	0.85	0.40
Tracy		0.50	0.87	0.64
Direct-spotting	Web Search	0.41	0.85	0.55
Tracy		0.90	0.97	0.93

Then, we used our system to extract explanations for the candidates over YAGO using either the Wikipedia articles (local fact-spotting) or Web search (remote fact-spotting). Later, we asked MTurk workers to judge the correctness of each candidate fact based on the provided explanations. Based on these annotations, we computed *recall*, *precision@5 (Prec@5)*, and F_1*@5* score.

Table 1 shows a comparison between the results of using Tracy against the baseline method which consists of using only *direct fact-spotting* (thus without any rewriting). We can observe that Tracy doubles the recall while slightly increasing the precision@5. This leads to significant enhancement in the F_1 score. These results are encouraging, as they show that our system is 1) indeed capable of finding additional evidence that might not be directly mentioned and that 2) the explanations are actually valuable for human annotators for judging the correctness of candidate facts. More detailed experimental results can be found in [3].

3 DEMONSTRATION

We built a Web interface to allow an easy interaction with the system. Fig. 3 shows a screenshot illustrating its main components.

System options. The left-hand side of Fig. 3 shows the *system options menu*, which is the part where the user can select from a predefined list of KGs and textual sources to work with. Moreover, the interface allows the user to tune the QRE procedure by specifying a *trust* value for each textual source and the limit on the number of used rules and output explanations.

Input. The top part in Fig. 3 contains the *input form* with three fields for the SPO *query* and a text area for the *rules*. In the *query* fields, the user can select the subject and the object of the fact in question from a set of KG entities and specify the predicate, which might be out of KG. In the *rules* text area, the user is expected to input the rules using the standard logic programming syntax. For example, a rule expressing that "*a person is a citizen of a country if he was born in one of its cities*" is written as:

$$isCitizenOf(?x, ?y) \text{ :- } bornIn(?x, ?z),\ isA(?z, \text{`}City\text{'}),\ in(?z, ?y)$$

where $?x, ?y, ?z$ are variables, '$City$' is a constant, *isCitizenOf* is a head predicate and the rest are body predicates. These rules can also involve out-of-KG predicates, which Tracy spots in the text.

Output. Below the rules text area, Tracy returns the list of explanations for the input fact. Each *explanation* card contains:

- A set of facts that support the correctness of the query;

[1] https://www.elastic.co/

[2] http://azure.microsoft.com/en-us/services/cognitive-services/bing-web-search-api

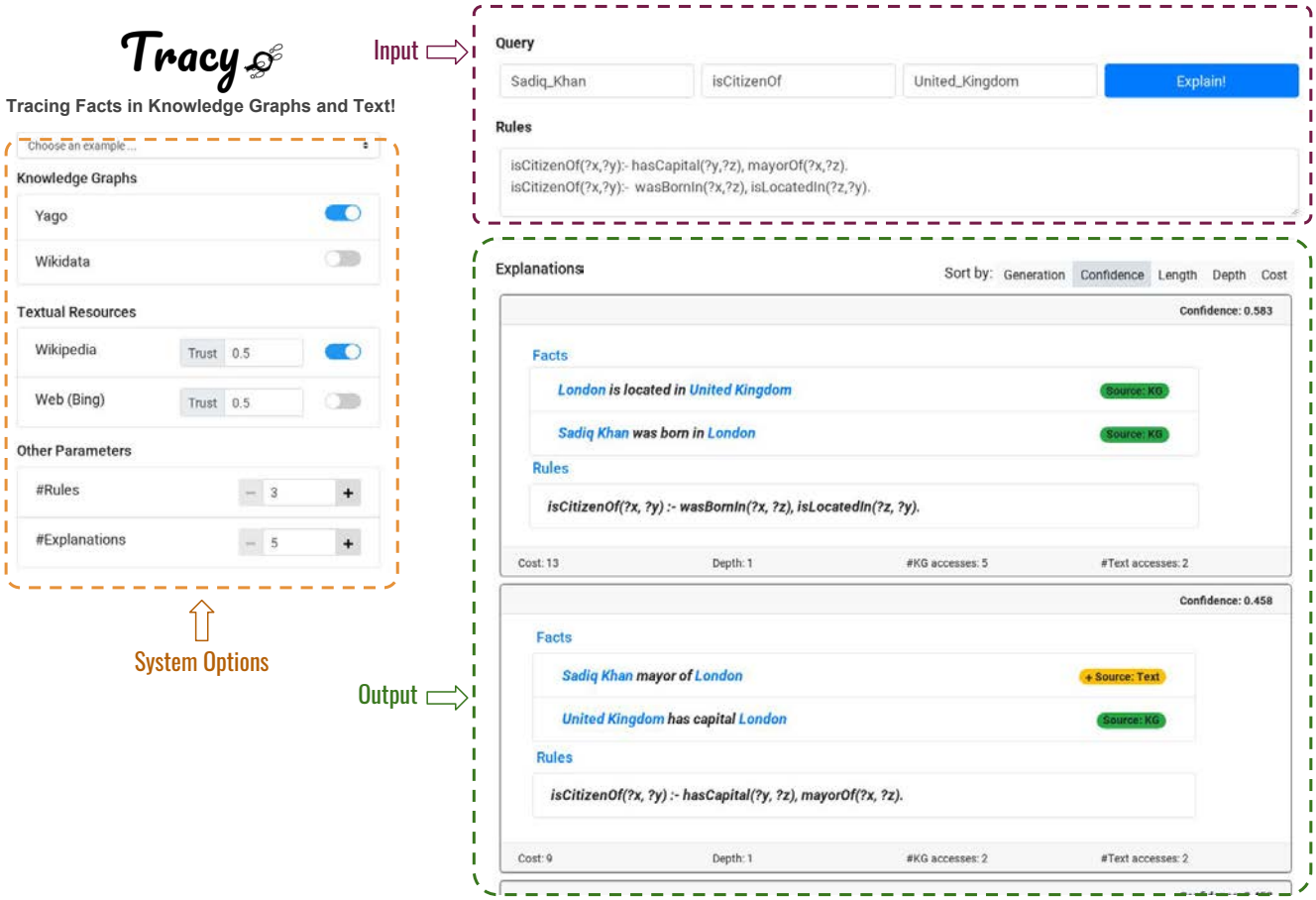


Figure 3: Tracy Web interface.

- In case the explanation includes facts with textual evidence, the user can view top-5 matching sentences with the links to the Web-pages where they have been found;
- The set of rules used by Tracy to find the evidence;
- Explanation confidence score computed as described in Section 2.1 and some execution insights (*e.g.*, resources accesses).

Explanations can be sorted based on their generation order, quality, length, rewriting depth, or cost.

During the demonstration session, We will showcase how Tracy can be used to extract non-trivial explanations for several hard to spot queries in different domains (*e.g.*, art, sport, and politicians). Users will have the chance to observe the results of changing the ruleset and other parameters to experience the performance of Tracy in different scenarios.

Users will also have the possibility to test Tracy with some arbitrary queries and/or other rule sets. We will encourage them to first try to collect related evidence manually before using Tracy with the same query. This illustrates how Tracy facilitates the discovery of related evidence required for judging the correctness of the query.

4 CONCLUSION

Our tool represents a first step towards producing more human understandable evidence for unverified facts. Tracy can help curators judging the correctness of new facts faster and more accurately. As future work, it is interesting to enhance our tool to support more complex rules, *e.g.*, with negations. Such rules are useful to represent additional background knowledge. Moreover, developing a more robust spotting engine that is capable of recognizing negative mentions of facts would allow more complex reasoning. Finally, we plan to further enhance the interface to make our system easier to use for KG curators.

ACKNOWLEDGMENTS

This work was partially supported by the ERC Synergy Grant 610150 (imPACT).

REFERENCES

[1] Xin Luna Dong and Divesh Srivastava. 2013. Compact Explanation of Data Fusion Decisions. In *Proceedings of the 22Nd International Conference on World Wide Web (WWW '13)*. ACM, 379–390.

[2] Mohamed H Gad-Elrab, Daria Stepanova, Jacopo Urbani, and Gerhard Weikum. 2016. Exception-enriched rule learning from knowledge graphs. In *Proceedings of ISWC*. 234–251.

[3] Mohamed H. Gad-Elrab, Daria Stepanova, Jacopo Urbani, and Gerhard Weikum. 2019. ExFaKT: A Framework for Explaining Facts over Knowledge Graphs and Text. In *Proceedings of WSDM*. 87–95.

[4] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2013. AMIE: Association Rule Mining Under Incomplete Evidence in Ontological Knowledge Bases. In *Proceedings of WWW*. 413–422.

[5] Daniel Gerber, Diego Esteves, Jens Lehmann, Lorenz Bühmann, Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, and René Speck. 2015. DeFacto-Temporal and Multilingual Deep Fact Validation. *Web Semant.* 35, P2 (Dec. 2015), 85–101.

[6] Vinh Thinh Ho, Daria Stepanova, Mohamed H. Gad-Elrab, Evgeny Kharlamov, and Gerhard Weikum. 2018. Rule Learning from Knowledge Graphs Guided by Embedding Models. In *The Semantic Web - ISWC 2018, Proceedings, Part I*. 72–90.

[7] Robert Kowalski and Donald Kuehner. 1971. Linear resolution with selection function. *Artificial Intelligence* 2, 3 (1971), 227 – 260.

[8] Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. 2015. A Survey on Truth Discovery. *SIGKDD Explorations* 17 (2015), 1–16.

[9] Ndapandula Nakashole and Tom M. Mitchell. 2014. Language-Aware Truth Assessment of Fact Candidates. In *Proceedings of ACL*. 1009–1019.

[10] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. PATTY: A Taxonomy of Relational Patterns with Semantic Types. In *Proceedings of EMNLP*. 1135–1145.

[11] Jeff Pasternack and Dan Roth. 2013. Latent credibility analysis. In *Proceedings of WWW*. 1009–1020.

[12] Heiko Paulheim. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web* 8, 3 (2017), 489–508.

[13] Kashyap Popat, Subhabrata Mukherjee, Jannik Strötgen, and Gerhard Weikum. 2017. Where the Truth Lies: Explaining the Credibility of Emerging Claims on the Web and Social Media. In *Proceedings of WWW*. 1003–1012.

[14] Daria Stepanova, Mohamed H. Gad-Elrab, and Vinh Thinh Ho. 2018. Rule Induction and Reasoning over Knowledge Graphs. In *Reasoning Web* (2018) *(Lecture Notes in Computer Science)*, Vol. 11078. Springer, 142–172.

[15] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge. In *Proceedings of WWW*. 697–706.

[16] Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of ACM* 57, 10 (2014), 78–85.

[17] Zhichun Wang and Juan-Zi Li. 2015. RDF2Rules: Learning Rules from RDF Knowledge Bases by Mining Frequent Predicate Cycles. *CoRR* abs/1512.07734 (2015).

[18] Zhuoyu Wei, Jun Zhao, Kang Liu, Zhenyu Qi, Zhengya Sun, and Guanhua Tian. 2015. Large-scale Knowledge Base Completion: Inferring via Grounding Network Sampling over Selected Instances. In *Proceedings of CIKM '15*. 1331–1340.

LearnerExp: Exploring and Explaining the Time Management of Online Learning Activity

Huan He
Key Laboratory of Intelligent Networks and Network Security, Ministry of Education
Xi'an Jiaotong University
Xi'an, Shaanxi, China
hehuan@mail.xjtu.edu.cn

Qinghua Zheng
School of Electronic and Information Engineering
Xi'an Jiaotong University
Xi'an, Shaanxi, China
qhzheng@mail.xjtu.edu.cn

Bo Dong
School of Continuing Education
National Engineering Lab for Big Data Analytics
Xi'an Jiaotong University
Xi'an, Shaanxi, China
dong.bo@mail.xjtu.edu.cn

ABSTRACT

How do learners schedule their online learning? This issue is concerned by both course instructors and researchers, especially in the context of self-paced online learning environment. Many indicators and methods have been proposed to understand and improve the time management of learning activities, however, there are few tools of visualizing, comparing and exploring the time management to gain intuitive understanding. In this demo, we introduce the **LearnExp**, an interactive visual analytic system designed to explore the temporal patterns of learning activities and explain the relationships between academic performance and these patterns. This system will help instructors to comparatively explore the distribution of learner activities from multiple aspects, and to visually explain the time management of different learner groups with the prediction of learning performance.

CCS CONCEPTS

• **Information systems** → **Data analytics**; • **Human-centered computing** → *Visualization systems and tools*; • **Applied computing** → E-learning.

KEYWORDS

Online learning; time management; visualization

ACM Reference Format:
Huan He, Qinghua Zheng, and Bo Dong. 2019. LearnerExp: Exploring and Explaining the Time Management of Online Learning Activity. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314140

1 INTRODUCTION

Accessing world class educational resources without geographical and social boundaries has become the daily behavior of online learners. With the advantages of learning anywhere and anytime, learners can schedule their learning at their own pace. However, this self-scheduled learning process also challenges their time management [1, 14]. In the less supervised online learning environment, the inappropriate scheduling of study time (e.g., absence, procrastination, etc.) may lead to reduced learning performance and even exam failures [16], which in turn would result in reduced completion rate and course rating [1]. Therefore, understanding the time management pattern of online learning activities has been an important issue for both learners and instructors.

With the support of learning management system (LMS), the detailed learning activities can be tracked via logs (e.g., viewing videos, reading materials, posting threads, etc.). Existing studies have proposed many indicators and models based on these logs to measure the time management features, such as the interval time between two tasks [1], the time that students spent on a certain task and LMS [4, 9], the remaining time between the deadlines [7], the aggregated daily activity count [14], etc. By using these indictors, recent studies examined the learners' behavioral patterns [12], self-regulated learning strategic [10, 11, 15], the influences on learning performance [16], visualization tools for supporting self-regulated learning [7], etc. These indicators and methods provide basis for measuring and improving the time management. In addition, many visualization tools have been developed to further investigate the patterns of student engagement, including SST [6], moocRP [13], DropoutSeer [3], PeakVisor [2] and iForum [5], etc.

Theses systems demonstrate the student engagement from different perspectives. However, the instructor has limited tools to customize student groups and explore each student's complete study history freely. To address this issue, we introduce the LearnerExp system for the instructor to explore the time management intuitively. First, we propose the activity vector to represent daily learning activities based on well-studied indicators and existing models. Moreover, we embed the activity vector in the calendar-like activity matrix to facilitate users understand the learning process intuitively. Second, we use grade point prediction model based on convolutional neural network to explain the relationships between learning performance and time management. Third and last, we develop a visual analytic system for course instructors to explore the temporal activity patterns of different learner groups. Main contributions of this work are summarized as follows.

- A learner activity model for describing and analyzing the time management in learning activities, which not only captures the temporal aspect of learning activities, but also supports visual exploration.
- A web-based interactive visual analytic system for exploring and explaining the time management of online learning activity based on the proposed learner activity model.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314140

2 LEARNER ACTIVITY MODEL

In this section, we introduce the learner activity representation and the prediction model for analyzing the time management.

2.1 The Learner Activity Representation

In online courses, the time-stamped learner-generated events are recorded by the LMS and saved in log files on a daily basis to describe the learning activity of each day, which consist of event timestamp, learner identifier, course information, operation and other data recorded by JavaScript script embedded in the LMS. With these log files, we count the day activity of each learner based on well-studied indicators. As shown in Figure 1(a), the day activity is measured by a set of indicators, including daily interaction counts of learner-course, learner-learner and learner-system, video viewing time, session time [4, 8, 14] etc.

Based on the day activity, the history of whole activities in a semester can be represented as the activity vector (Figure 1(b)). In terms of notation, we let $\boldsymbol{x}_i$ be the activity vector for learner i, where $i = 1, ..., N$. The dimensionality of activity vector $\boldsymbol{x}_i$ is $D \times K$, where K is the number of indicators and D is the number of days in the semester. The activity vector $\boldsymbol{x}_i$ is a complete representation of a learner's detailed activities on a daily basis, which captures the temporal aspect of learning activities. It can be further used in predicting learning performance to analyze the patterns of time management. However, this representation is not suitable for users to gain an intuitive understanding.

In order to visualize the activity vector, we first compress the $D \times K$-dimensional activity vector $\boldsymbol{x}_i$ into $D \times 1$-dimensional vector, and then embed this compressed vector in the calendar-based $W \times 7$-dimensional activity matrix by date, where W represents the number of weeks in a semester (usually $W < 28$). As shown in Figure 1(c), the compressed day activity is represented as a color-encoded cell (the darker the higher, indicates that the activity on the corresponding date is more frequent). There are many ways to compress the day activity vector, such as linear weighted aggregation, PCA-based selection, etc. In this demo, we use a weighted sum of all indicators to compress day activity according to the suggestions from instructors.

This calendar-like representation of activity matrix is consistent with the calendar that users often use. By adding some auxiliary graphical elements (e.g., segment lines, week numbers, etc.), the user can clearly understand learners' study histories throughout the semester. As shown in Figure 1(d), the learning activity history of five learners are fully recorded by the activity matrix, and different patterns of time management can be intuitively presented.

2.2 The Grade Point Prediction Model

Since the grade point is often used in management to evaluate the learning performance, we convert exam scores into grade points from 0 to 4, which is consistent with our school practice (0 indicates unqualified, 1 to 4 respectively indicates from passing to excellent). To capture the temporal aspects of learner activity representation, we use a grade point prediction model based on convolutional neural network to analyze the relationship between time management patterns and grade point.

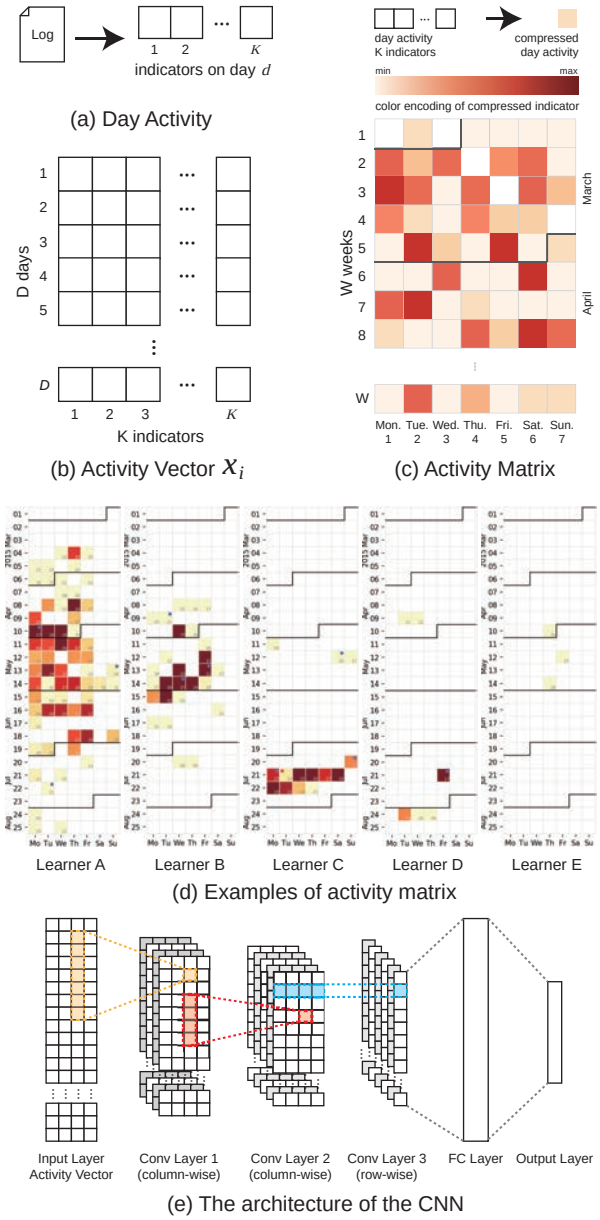


Figure 1: The learner activity representation and the prediction model based on convolutional neural network

As shown in Figure 1(e), the model consists of three convolutional layers using the rectified linear unit (ReLU) as activation function to capture the temporal features of the activity vector. Specifically, the first two column-wise convolutional layers take convolutions that capture the short-term (e.g., 1 cell = 7 days) and long-term (e.g., 1 cell = 4 weeks) patterns of learning activity, respectively. The third row-wise convolutional layer takes convolutions that capture the pattern of activity indicators at each timepoint. Then, the fully-connected layer extracts latent representation from the convolutional layer via a sigmoid function. Finally, the last layer outputs the probabilities of each grade point by a softmax function.

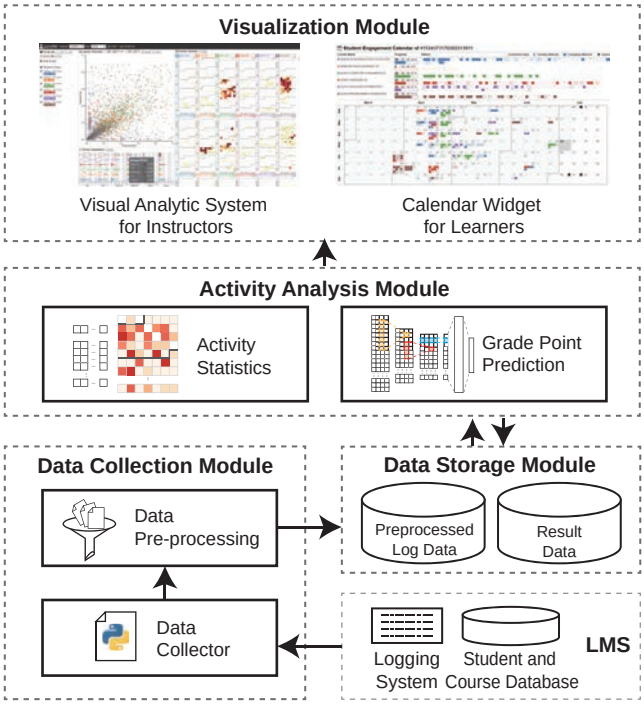


Figure 2: The architecture of the LearnerExp system

We trained this model on the dataset from the online undergraduate program of our university, which consists of 10,529 students and their learning logs in the first semester (14,942,964 lines). The average accuracy of current trained model is 72.2% and we will improve it in future research. The prediction results will be used in the exploration of the relationship between time management patterns and grade points in the LearnerExp system.

3 THE LEARNEREXP SYSTEM

3.1 The Architecture

The architecture of LearnerExp system is shown in Figure 2. It consists of four major components: *data collection module*, *data storage module*, *activity analysis module* and *visualization module*. Since we separate the different functions into modules, each module can be executed independently or integrated into the subsystem of the LMS. For example, the calendar widget in visualization module is integrated in learner's dashboard of the LMS.

Data collection and data storage module. In data collection module, we use Python scripts to collect course data and raw learner logs from the LMS. After filtering irrelevant information and filling missing fields, the preprocessed learner log data is saved in the data storage module. The personal identification information in log data is removed and will be not saved for privacy concerns. In addition, to facilitate access from the visualization module, the intermediate results of statistics and predictions are also saved in the data storage module.

Activity analysis module. This module includes two submodules: the activity statistics module measures the activity vector of each learner and generates the activity matrix defined in Section 2.1. Then, the grade point prediction module predicts the probability of each learner's grade points based on the pre-trained model mentioned in Section 2.2. The learners with a higher probability of grade 0 (i.e., failing the exam) will be grouped by the probability value for further analysis in the visualization module.

Visualization module. To make full use of the analysis results, we developed a web-based visual analytic system that enables course instructors to explore the engagement distribution and the temporal patterns of learners' activities from various aspects (Figure 3). In addition, we developed a web-based calendar widget to help learners understand their own learning activity history (Figure 2). This widget is integrated in the dashboard of the LMS, learners can check their activity history immediately after login.

3.2 Visual Design

In this section, we illustrate the visual design of the LearnerExp system and the interactions among the views. As shown in Figure 3, there are four coordinated views to show the analysis results from different perspectives: the Group List (Figure 3(a) and (b)), the Learner Overview (Figure 3(c)), the Group Comparison view (Figure 3(d)) and the Activity Calendar view (Figure 3(e)).

The Learner Overview and The Group List. To provide course instructors with intuitive impression on the overall activity distribution of all learners, we use scatter plot to present the statistics on the entire semester (Figure 3(c)). The values of each indicator over the entire semester are aggregated to represent learner's overall performance. For example, as shown in Figure 3(c), the *attendance rate* represents the ratio of the total number of daily videos watched to the total number of course videos. In this chart, each gray dot represents a learner, while each colored dot represents a learner who belongs to a specific learner group. Both the x-axis and the y-axis can be set to a specified indicator to see the distribution of same population under different indicators.

There are two types of learner groups in our system: the prediction groups generated by the model described in Section 2.2, and the temporary groups customized by selecting learners on the Leaner Overview. We developed an interactive selector to make it easier for users to intuitively select specific learners of interest (Figure 3(f)). By drawing a closed shape on the scatter plot with mouse, users can create a group of the selected dots within the shape. The newly created group will be automatically named and displayed in the "Temporary Groups" of the Group List (Figure 3(b)). To facilitate identification of learner groups in different views, the color-encoding used in the Group List is also used in other views.

The Group Comparison View. This view presents a set of box charts to simultaneously compare the distribution differences in learner groups across multiple indicators. As shown in Figure 3(d), from left to right, each box chart displays an indicator, in which each series represents a learner group from top to bottom. The color of each series is consistent with the color displayed in the Group List. When the mouse is over the series, the detailed statistics of the group on this indicator will be displayed.

The Activity Calendar View. In this view, the activity matrix defined in Section 2.1 is displayed to explore the details of learner's learning activity history. This view displays the learner groups in columns from left to right, in which the activity matrix of each

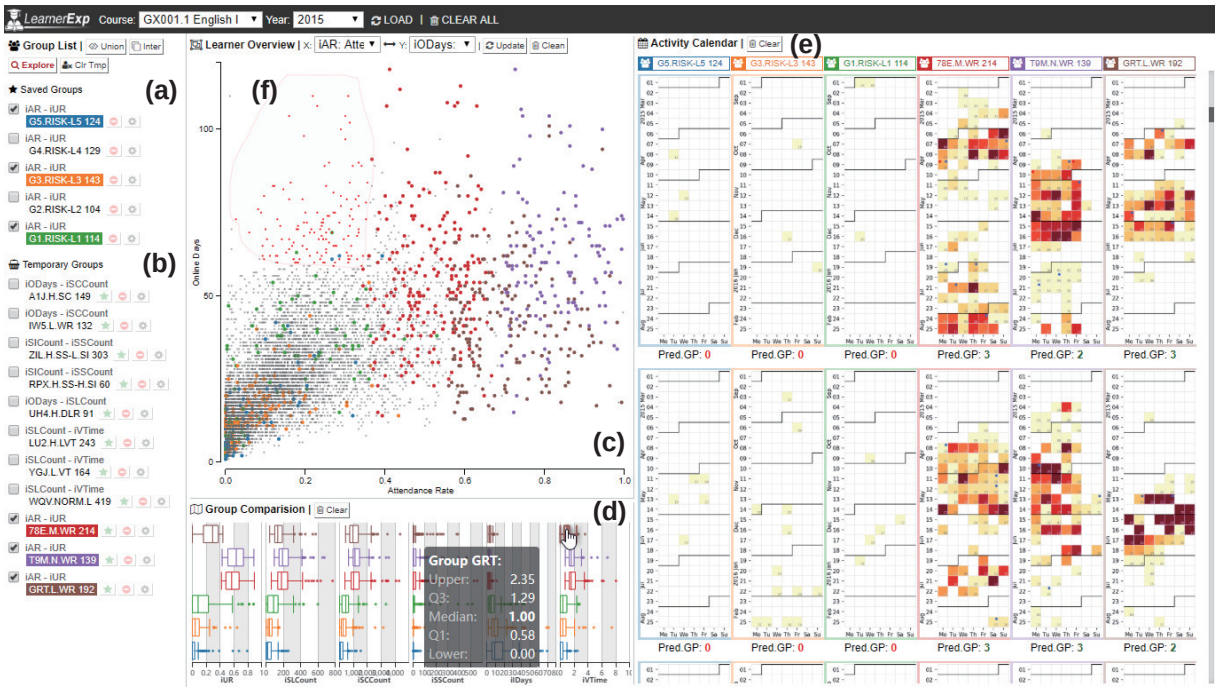


Figure 3: Screenshot of the LearnerExp system. (a) The saved group list. (b) The temporary group List. (c) The Learner Overview. (d) The Group Comparation View. (e) The Activity Calendar view. (f) A group of learners selected by the interactive selector.

learner is shown from top to bottom. When the mouse is over each activity matrix, the corresponding dot in the Learner Overview will be enlarged, and the corresponding location in each box chart will be marked to show the learner's relative performance in the whole population and in the group.

4 DEMONSTRATION

During the demonstration the attendee will evaluate LearnerExp [1] with a real dataset from our university, which is the same dataset as we mentioned in Section 2.2 but without personal privacy information. Additionally, we provide an exemplified scenario in which the course instructor wants to find the learners who may fail the posttest exam [2]. The demonstration scenario is the following.

(1) The user starts by loading data. After selecting the course in the "Course" drop-down list and clicking the "Load" button, the corresponding dataset will be loaded into the web page. All the learners of this course will be shown in the Learner Overview, and the prediction groups with different probability of failing the exam will be listed in the Group List. Figure 3(a) shows five groups, which are listed by the probability of failure of the exam from high to low.

(2) Next, the user can select groups and click the "Explore" button. Then, the selected groups will be colored, and the details will be displayed in each view. As shown in Figure 3, the learner dots belong to each group are colored and enlarged to show their relative position in the Leaner Overview (Figure 3(c)), and their study histories are listed in the Activity Calendar View (Figure 3(e)). For example, learners in the red, purple, and brown groups show regular time management characteristics and their predicted score points are higher than the blue, orange, and green groups.

(3) In addition to the prediction groups, the user can customize groups of interested learners for comparative exploration with the interactive selector (Figure 3(f)). After creating new groups, the user can select these newly created groups from the "Temporary Groups" and click the "Explore" button to continue exploring.

(4) Finally, after repeatedly exploring different groups, the user may be interested in a few customized groups and plan to analyze them in the future. By clicking the star-icon button next to the group label in the Group List, the corresponding temporary group will be saved and moved to the "Saved Groups".

5 CONCLUSION AND FUTURE WORK

In this demo, we present an interactive visual analytic system, LearnerExp, for visually exploring the patterns of online learning engagement and time management based on our proposed learner activity model. In future research, we will further expand the features of the LearnerExp and investigate the impact of different time management patterns on learning performance.

ACKNOWLEDGMENTS

This research was partially supported by "The Fundamental Theory and Applications of Big Data with Knowledge Engineering" under the National Key Research and Development Program of China with Grant No. 2018YFB1004500, the MOE Innovation Research Team No. IRT17R86, the National Science Foundation of China under Grant Nos. 61721002 and 61532015, and Project of China Knowledge Center for Engineering Science and Technology.

[1] Screencast for LearnerExp: https://youtu.be/j_mtLp-8v0U
[2] Live demo with source code: https://hehuan2112.github.io/LearnerExp

REFERENCES

[1] J. Broadbent and W. L. Poon. 2015. Self-regulated learning strategies & academic achievement in online higher education learning environments: A systematic review. *The Internet and Higher Education* 27 (2015), 1–13. https://doi.org/10.1016/j.iheduc.2015.04.007

[2] Qing Chen, Yuanzhe Chen, Dongyu Liu, Conglei Shi, Yingcai Wu, and Huamin Qu. 2016. PeakVizor: Visual Analytics of Peaks in Video Clickstreams from Massive Open Online Courses. *IEEE Transactions on Visualization and Computer Graphics* 22, 10 (Oct. 2016), 2315–2330. https://doi.org/10.1109/TVCG.2015.2505305

[3] Y. Chen, Q. Chen, Mingqian Zhao, S. Boyer, K. Veeramachaneni, and H. Qu. 2016. DropoutSeer: Visualizing learning patterns in Massive Open Online Courses for dropout reasoning and prediction. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*. 111–120. https://doi.org/10.1109/VAST.2016.7883517

[4] Gianni Fenu, Mirko Marras, and Massimiliano Meles. 2017. A Learning Analytics Tool for Usability Assessment in Moodle Environments. *Journal of e-Learning and Knowledge Society* 13, 3 (Sept. 2017). https://doi.org/10.20368/1971-8829/1388

[5] Siwei Fu, Jian Zhao, Weiwei Cui, and Huamin Qu. 2017. Visual Analysis of MOOC Forums with iForum. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (Jan. 2017), 201–210. https://doi.org/10.1109/TVCG.2016.2598444

[6] Diego Alonso Gómez-Aguilar, Ángle Hernández-García, Francisco J. García-Peñalvo, and Roberto Therón. 2015. Tap into visual analysis of customization of grouping of activities in eLearning. *Computers in Human Behavior* 47 (June 2015), 60–67. https://doi.org/10.1016/j.chb.2014.11.001

[7] Kalle Ilves, Juho Leinonen, and Arto Hellas. 2018. Supporting Self-Regulated Learning with Visualizations in Online Learning Environments. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18)*. ACM, New York, NY, USA, 257–262. https://doi.org/10.1145/3159450.3159509

[8] Srećko Joksimović, Dragan Gašević, Thomas M. Loughin, Vitomir Kovanović, and Marek Hatala. 2015. Learning at distance: Effects of interaction traces on academic achievement. *Computers & Education* 87 (Sept. 2015), 204–217. https://doi.org/10.1016/j.compedu.2015.07.002

[9] Ayaan M. Kazerouni, Stephen H. Edwards, and Clifford A. Shaffer. 2017. Quantifying Incremental Development Practices and Their Relationship to Procrastination. In *Proceedings of the 2017 ACM Conference on International Computing Education Research*. ACM, 191–199. https://doi.org/10.1145/3105726.3106180

[10] René F. Kizilcec, Mar Pérez-Sanagustín, and Jorge J. Maldonado. 2016. Recommending Self-Regulated Learning Strategies Does Not Improve Performance in a MOOC. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale (L@S '16)*. ACM, New York, NY, USA, 101–104. https://doi.org/10.1145/2876034.2893378

[11] René F. Kizilcec, Mar Pérez-Sanagustín, and Jorge J. Maldonado. 2017. Self-regulated learning strategies predict learner behavior and goal attainment in Massive Open Online Courses. *Computers & Education* 104 (Jan. 2017), 18–33. https://doi.org/10.1016/j.compedu.2016.10.001

[12] Mohammad Javad Mahzoon, Mary Lou Maher, Omar Eltayeby, Wenwen Dou, and Kazjon Grace. 2018. A Sequence Data Model for Analyzing Temporal Patterns of Student Data. *Journal of Learning Analytics* 5, 1 (April 2018), 55–74. https://doi.org/10.18608/jla.2018.51.5

[13] Zachary A. Pardos and Kevin Kao. 2015. moocRP: An Open-source Analytics Platform. In *Proceedings of the Second (2015) ACM Conference on Learning @ Scale*. ACM Press, 103–110. https://doi.org/10.1145/2724660.2724683

[14] Jihyun Park, Renzhe Yu, and Fernando Rodriguez. 2018. Understanding Student Procrastination via Mixture Models. In *Proceedings of the 11th International Conference on Educational Data Mining*. 11.

[15] Masanori Yamada, Yoshiko Goda, Takeshi Matsuda, Yutaka Saito, Hiroshi Kato, and Hiroyuki Miyagawa. 2016. How does self-regulated learning relate to active procrastination and other learning behaviors? *Journal of Computing in Higher Education* 28, 3 (Dec. 2016), 326–343. https://doi.org/10.1007/s12528-016-9118-9

[16] Tzu-Chi Yang, Meng Chang Chen, and Sherry Y. Chen. 2018. The influences of self-regulated learning support and prior knowledge on improving learning performance. *Computers & Education* 126 (Nov. 2018), 37–52. https://doi.org/10.1016/j.compedu.2018.06.025

Gradual Machine Learning for Entity Resolution

Boyi Hou, Qun Chen, Jiquan Shen, Xin Liu, Ping Zhong, Yanyan Wang, Zhaoqiang Chen, Zhanhuai Li
School of Computer Science, Northwestern Polytechnical University
Xi'an, Shaanxi
{ntoskrnl,shenjiquan,liuxin,pingzhong,wangyanyan,chenzhaoqiang}@mail.nwpu.edu.cn,{chenbenben,lizhh}@nwpu.edu.cn

ABSTRACT

Usually considered as a classification problem, entity resolution can be very challenging on real data due to the prevalence of dirty values. The state-of-the-art solutions for ER were built on a variety of learning models (most notably deep neural networks), which require lots of accurately labeled training data. Unfortunately, high-quality labeled data usually require expensive manual work, and are therefore not readily available in many real scenarios. In this demo, we propose a novel learning paradigm for ER, called *gradual machine learning*, which aims to enable effective machine labeling without the requirement for manual labeling effort. It begins with some easy instances in a task, which can be automatically labeled by the machine with high accuracy, and then gradually labels more challenging instances based on iterative factor graph inference. In gradual machine learning, the hard instances in a task are gradually labeled in small stages based on the estimated evidential certainty provided by the labeled easier instances. Our extensive experiments on real data have shown that the proposed approach performs considerably better than its unsupervised alternatives, and its performance is also highly competitive compared to the state-of-the-art supervised techniques. Using ER as a test case, we demonstrate that gradual machine learning is a promising paradigm potentially applicable to other challenging classification tasks requiring extensive labeling effort.

Video: https://youtu.be/99bA9aamsgk

CCS CONCEPTS

• **Computing methodologies** → **Learning paradigms**.

KEYWORDS

gradual machine learning; entity resolution; unsupervised learning

ACM Reference Format:
Boyi Hou, Qun Chen, Jiquan Shen, Xin Liu, Ping Zhong, Yanyan Wang, Zhaoqiang Chen, Zhanhuai Li. 2019. Gradual Machine Learning for Entity Resolution. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314121

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314121

1 INTRODUCTION

The task of entity resolution (ER) aims at finding the records that refer to the same real-world entity [5]. Consider the example shown in Figure 1. ER needs to match the paper records between two tables T_1 and T_2. The pair of $< r_{1i}, r_{2j} >$, in which r_{1i} and r_{2j} denote a record in T_1 and T_2 respectively, is called a *matching* pair if and only if r_{1i} and r_{2j} refer to the same paper. In the example, r_{11} and r_{21} are *matching* while r_{11} and r_{22} are *unmatching*. The state-of-the-art solutions for ER were built on a variety of learning models (e.g., deep neural network [9]), which require lots of accurately labeled training data. Unfortunately, high-quality labeled data usually require expensive manual work, are therefore not easily available.

T_1

ID	Title	Author	Venue	Year
r_{11}	Belief Reasoning in MLS Deductive Databases	H Jamil	SIGMOD Conference	1999
r_{12}	Efficient Index Structures for String Databases	T Kahveci, A Singh	VLDB	2001
	.....			

T_2

ID	Title	Author	Venue	Year
r_{21}	Belief Reasoning in MLS Deductive Databases	HM Jamil	SIGMOD Conference	1999
r_{22}	Reasoning on Regular Path Queries	D Calvanese	SIGMOD RECORD	2003
	.....			

Figure 1: An ER Example

It can be observed that the dependence of the popular learning models (e.g., DNN) on high-quality labeled data is not limited to the task of ER. The dependence is actually crucial for their success in various domains (e.g., image and speech recognition [13]). However, in the real scenarios, where high-quality labeled data is scarce, their efficacy can be severely compromised. To address the limitation resulting from such dependence, we propose a novel learning paradigm for ER, called *gradual machine learning*, in which *gradual* means proceeding in small stages. Inspired by the gradual nature of human learning, which is adept at solving the problems with increasing hardness, gradual machine learning begins with some easy instances in a task, which can be automatically labeled by the machine with high accuracy, and then gradually reasons about the labels of the more challenging instances based on the observations provided by the labeled instances.

We note that there already exist many learning paradigms for a variety of classification tasks, including transfer learning [10], lifelong learning [4], curriculum learning [2] and self-training learning [8] to name a few. Transfer learning focused on using the labeled training data in a domain to help learning in another target domain. Lifelong learning studied how to leverage the knowledge mined from past tasks for the current task. Curriculum learning

investigated how to organize a curriculum (the presenting order of training examples) for better performance. Self-training learning aimed to improve the performance of a supervised learning algorithm by incorporating unlabeled data into the training data set. More recently, Snorkel [11] aimed to enable automatic and massive machine labeling by specifying a wide variety of labeling functions, whose results are supposed to be fed to DNN models.

However, the following two properties of gradual machine learning make it fundamentally different from the existing learning paradigms:

- Distribution misalignment between easy and hard instances in a task. The scenario of gradual machine learning does not satisfy the i.i.d (independent and identically distributed) assumption underlying most machine learning models: the labeled easy instances are not representative of the unlabeled hard instances. The distribution misalignment between the labeled and unlabeled instances renders most existing learning models unfit for gradual machine learning.
- Gradual learning by small stages in a task. Gradual machine learning proceeds in small stages. At each stage, it chooses to only label the instance with the highest degree of evidential certainty in a task based on the observations provided by the labeled instances. The process of iterative labeling can be performed in an unsupervised manner without requiring any human intervention.

The contributions of this demo can be summarized as follows: (1) a general paradigm of gradual machine learning **(Section 2)**; (2) a solution for ER based on the proposed paradigm **(Section 3)**; (3) an empirical study validating the efficacy of the proposed solution **(Section 4)**.

2 LEARNING PARADIGM

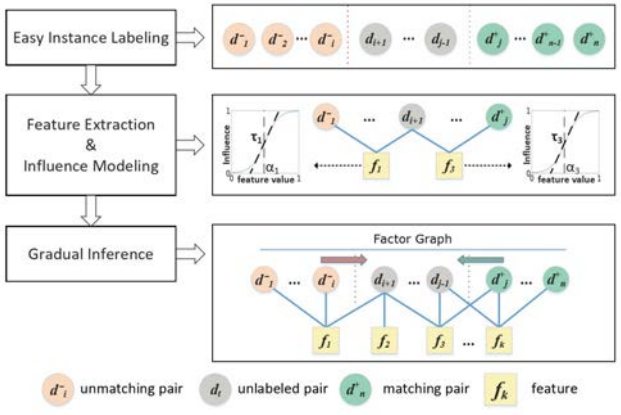


Figure 2: Paradigm Overview.

The paradigm of gradual machine learning, as shown in Figure 2, consists of the following three essential steps:

- **Easy Instance Labeling.** Given a classification task, it is usually very challenging to accurately label all the instances in the task without good-coverage training examples. However, the work can become much easier if we only need to automatically label some easy instances in the task. In the case of ER, while the pairs with the medium similarities are usually challenging for machine labeling, highly similar (resp. dissimilar) pairs have fairly high probabilities to be equivalent (resp. inequivalent). They can therefore be chosen as easy instances. In real scenarios, easy instance labeling can be performed based on the simple user-specified rules or the existing unsupervised learning techniques. Gradual machine learning begins with the observations provided by the labels of easy instances. Therefore, the high accuracy of automatic machine labeling on easy instances is critical for its ultimate performance on a given task.
- **Feature Extraction and Influence Modeling.** Features serve as the medium to convey the knowledge obtained from the labeled easy instances to the unlabeled harder ones. This step extracts the common features shared by the labeled and unlabeled instances. To facilitate effective knowledge conveyance, it is desirable that a wide variety of features are extracted to capture as much information as possible. For each extracted feature, this step also needs to model its influence over the labels of its relevant instances.
- **Gradual Inference.** This step gradually labels the instances with increasing hardness in a task. Since the scenario of gradual learning does not satisfy the i.i.d assumption, we propose to fulfill gradual learning from the perspective of evidential certainty. As shown in Figure 2, we construct a factor graph, which consisting of the labeled and unlabeled instances and their common features. Gradual learning is conducted over the factor graph by iterative factor graph inference. At each iteration, it chooses the unlabeled instance with the highest degree of evidential certainty for labeling. The iteration is repeatedly invoked until all the instances in a task are labeled. Note that in gradual inference, a newly labeled instance at the current iteration would serve as an evidence observation in the following iterations.

3 SOLUTION FOR ER

3.1 Easy Instance Labeling

Given an ER task consisting of record pairs, the solution identifies the easy instances by the simple rules specified on record similarity. The set of easy instances labeled as *matching* is generated by setting a high lowerbound on record similarity. Similarly, the set of easy instances labeled as *unmatching* is generated by setting a low upperbound on record similarity. The effectiveness of the rule-based approach can be explained by the monotonicity assumption of precision [1]. With the metric of pair similarity, the underlying intuition of monotonicity assumption is that the more similar two records are, the more likely they refer to the same real-world entity. According to the monotonicity assumption, we can *statistically* state that a pair with a high (resp. low) similarity has a correspondingly high probability of being an equivalent (resp. inequivalent) pair. These record pairs can be deemed to be easy in that they can be automatically labeled by the machine with high accuracy. In comparison, the instance pairs having the medium similarities are more challenging because labeling them either way by the machine would introduce considerable errors.

3.2 Feature Extraction and Influence Modeling

Given an ER workload, we extract three types of features from its pairs, which include:

(1) Attribute value similarity. Different attributes may require different similarity metrics.
(2) Similarity based on the maximal number of common consecutive tokens in string attributes. Consecutive tokens can usually provide additional information besides that implied by the attribute value similarity features.
(3) The tokens occurring in both records or in one and only one record. Representing a token by o_i, we denote the feature of o_i occurring in both records by *Same(o_i)*, and the feature of o_i occurring in one and only one record by *Diff(o_i)*.Unlike the previous two types of features, which treat an attribute value as a whole, this type of feature considers the influence of each individual token on pair equivalence probability.

These three types of features can provide good coverage of the information contained in record pairs. We observe that all the three types of features can be supposed to satisfy the monotonicity assumption of precision. Therefore, for each feature f, we model its influence over pair labels by a monotonous sigmoid function with two parameters, α_f and τ_f as shown in Figure 2, which denote the x-value of the function's midpoint and the steepness of the curve respectively. Formally, given a feature f and a pair d, the influence of f w.r.t d is represented by

$$P_f(d) = \frac{1}{1 + e^{-\tau_f(x_f(d)-\alpha_f)}}, \tag{1}$$

in which $x_f(d)$ represents f's value w.r.t d. Since the third type of features has the constant value of 1, we first align them with record similarity and then model their influence by sigmoid functions. It is worthy to point out that monotonicity of precision is a universal assumption underlying the effectiveness of the existing machine metrics for classification tasks. Our proposed solution for feature influence modeling can, therefore, be potentially generalized for other classification tasks.

3.3 Gradual Inference

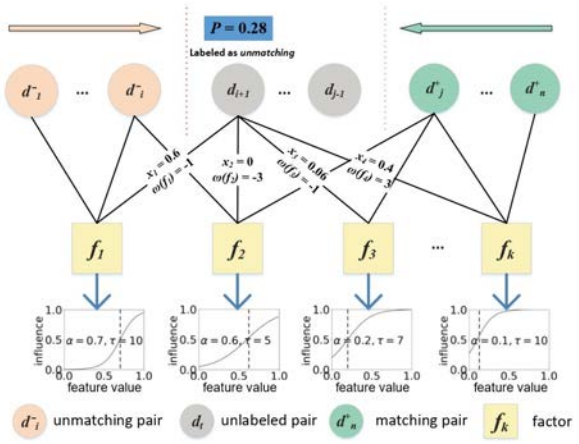


Figure 3: An Example of Factor Graph.

To enable gradual machine learning, we construct a factor graph, G, which consists of the labeled easy instances, the unlabeled hard instances and their common features. Gradual machine learning is attained by iterative factor graph inference on G. In G, the labeled easy instances are represented by the *evidence variables*, the unlabeled hard instances by the *inference variables*, and the features by the *factors*. The value of each variable represents its corresponding pair's equivalence probability. An evidence variable has the constant value of 0 or 1, which indicate the status of *unmatching* and *matching* respectively. It participates in gradual inference, but its value remains unchanged during the inference process. The values of the inference variables should instead be inferred based on G. An example of factor graph inference has been shown in Figure 3.

Note that the influence of a feature over a pair is specified by the sigmoid function as shown in Eq. 1. Therefore, in the factor graph, we represent the factor weigh of f w.r.t d by

$$\omega_f(d) = \theta_f(d) \cdot \log(\frac{P_f(d)}{1 - P_f(d)}) = \theta_f(d) \cdot \tau_f(x_f(d) - \alpha_f), \tag{2}$$

in which $\log(\cdot)$ codes the estimated influence of f on d by sigmoid regression, and $\theta_f(d)$ represents the confidence on influence estimation. We consider $\theta_f(d)$ as the confidence on the regression result provided by its corresponding sigmoid function, and estimate it based on the theory of regression error bound [3].

A factor graph infers the equivalence probability of a pair d, $P(d)$, by:

$$P(d) = \frac{\prod\limits_{f \in F_d} e^{\omega_f(d)}}{1 + \prod\limits_{f \in F_d} e^{\omega_f(d)}}, \tag{3}$$

in which F_d denotes the feature set of the pair d. The process of gradual inference essentially learns the parameter values (α and τ) of all the features such that the inferred results maximally match the evidence observations on the labeled instances. Formally, the objective function can be represented by

$$(\hat{\alpha}, \hat{\tau}) = \arg\min_{\alpha,\tau} -\log \sum_{V_I} P_{\alpha,\tau}(\Lambda, V_I), \tag{4}$$

in which Λ denotes the observed labels of evidence variables, V_I denotes the inference variables in G, and $P_{\alpha,\tau}(\Lambda, V_I)$ denotes the joint probability of the variables in G. Since the variables in G are conditionally independent, $P_{\alpha,\tau}(\Lambda, V_I)$ can therefore be represented by:

$$P_{\alpha,\tau}(\Lambda, V_I) = \prod_{d \in \Lambda \cup V_I} P_{\alpha,\tau}(d). \tag{5}$$

Accordingly, the objective function can be simplified into

$$(\hat{\alpha}, \hat{\tau}) = \arg\min_{\alpha,\tau} -\log \sum_{d \in \Lambda} P_{\alpha,\tau}(d). \tag{6}$$

Given a factor graph, G, at each stage, gradual inference first reasons about the parameter values of the features and the equivalence probabilities of the unlabeled pairs by maximum likelihood, and then labels the unlabeled pair with the highest degree of evidential certainty. We define evidential certainty as the inverse of entropy [12], which is formally defined by

$$H(d) = -(P(d) \cdot \log_2 P(d) + (1 - P(d)) \cdot \log_2(1 - P(d))), \tag{7}$$

in which $H(d)$ denotes the entropy of d. An inference variable once labeled would become an evidence variable and serve as an evidence observation in the following iterations. The iteration is repeatedly invoked until all the inference variables are labeled. In our programs, we deploy SciPy [7] to implement the process of factor graph inference.

However, repeated inference by maximum likelihood estimation over a large-sized factor graph of the whole variables is usually very time-consuming [14]. It is also unnecessary because at each iteration, only the inference variables receiving considerable evidential support from evidence variables need to be considered for labeling. Therefore, we have proposed a scalable solution for gradual inference. It first selects the top-m unlabeled variables with the most evidential support in G as the candidates for labeling. To reduce the invocation frequency of factor graph inference, it then approximates probability estimation on the m candidates by a more efficient algorithm. Finally, it infers via maximum likelihood the probabilities of only the top-k most promising variables among the m candidates. For each variable in the final set of k candidates, its probability is not inferred over the entire graph of G, but over a potentially much smaller subgraph. More technical details of scalable gradual inference can be found in our technical report [6], but omitted here due to space limit.

4 EMPIRICAL EVALUATION & DEMO PLAN

In this section, we empirically evaluate the performance of our proposed approach (denoted by GML) on real data. We have empirically compared the proposed solution with four alternatives, including unsupervised rule-based (UR), unsupervised clustering (UC), support vector machine (SVM) and deep neural network (DNN) [9]. Our evaluation was conducted on 3 real datasets, DBLP-Scholar[1](denoted by DS), Abt-Buy[2] (denoted by AB) and Songs[3] (denoted by SG).

The detailed evaluation results are presented in Table 1, in which r and p stand for recall and precision respectively, and the results on F-1 have been highlighted. For the supervised approaches of SVM and DNN, we report their performance provided with different sizes of training data, which are measured by the fraction of training data among the whole workload. For DNN, the training data consists of the data used for model training and the data used for validation. Therefore, we report the fractions of both parts in the table. It can be observed that GML performs considerably better than the unsupervised alternatives, UR and UC. In most cases, their performance differences on F-1 are larger than 5%. The performance of GML in terms of F-1 is also highly competitive compared to both supervised approaches of SVM and DNN. GML beats SVM (with the maximum training data size at 30%) on all the three test datasets; GML also beats DNN (with the maximum training data size at 30%) on both AB and SG.

Demo Plan. We have implemented a demo system for gradual machine learning, whose screenshots are shown in Fig. 4. It consists of four components: Easy Instance Labeling (EIL), Feature Extraction(FE), Gradual Inference (GI) and finally Result Report (RR). EIL labels easy instances. FE extracts features, models the influence of features and constructs factor graph. GI demonstrates the process of scalable gradual inference, which includes top-m candidate selection based on evidential support measurement, efficient entropy estimation and approximate factor graph inference. In GI, we have also designed an animation to visualize the iterative labeling process of GML. Finally, RR reports the experimental results. The attendees will be invited to operate the demo system step by step on a laptop using various datasets. They will be able to look into the details of the visualized process of GML.

[1]available at https://dbs.uni-leipzig.de/file/DBLP-Scholar.zip
[2]available at https://dbs.uni-leipzig.de/file/Abt-Buy.zip
[3]available at http://pages.cs.wisc.edu/~anhai/data/falcon_data/songs

Table 1: Comparative Evaluation of GML

	GML			UR			UC		
	r	p	F-1	r	p	F-1	r	p	F-1
DS	0.885	0.944	**0.914**	0.923	0.840	**0.880**	0.793	0.939	**0.860**
AB	0.461	0.790	**0.582**	0.645	0.428	**0.514**	0.806	0.268	**0.402**
SG	0.979	0.962	**0.970**	0.993	0.825	**0.901**	0.995	0.808	**0.892**

	SVM								
	10%			20%			30%		
	r	p	F-1	r	p	F-1	r	p	F-1
DS	0.890	0.918	**0.903**	0.892	0.918	**0.904**	0.896	0.921	**0.908**
AB	0.667	0.387	**0.490**	0.674	0.404	**0.505**	0.535	0.525	**0.530**
SG	0.995	0.855	**0.920**	0.992	0.925	**0.957**	0.991	0.945	**0.968**

	DNN								
	10%(5%:5%)			20%(15%:5%)			30%(25%:5%)		
	r	p	F-1	r	p	F-1	r	p	F-1
DS	0.949	0.869	**0.907**	0.945	0.956	**0.950**	0.982	0.929	**0.955**
AB	0.043	0.254	**0.074**	0.441	0.601	**0.509**	0.444	0.707	**0.546**
SG	0.777	0.830	**0.802**	0.952	0.900	**0.925**	0.938	0.970	**0.954**



Figure 4: Demo System Screenshots

5 ACKNOWLEDGEMENT

This work is supported by the Ministry of Science and Technology of China, National Key Research and Development Program (2016YFB1000703), National Natural Science Foundation of China (61732014, 61672432, 61472321 and 61502390).

REFERENCES

[1] Arvind Arasu, Michaela Götz, and Raghav Kaushik. 2010. On Active Learning of Record Matching Packages. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data (SIGMOD '10)*. ACM, New York, NY, USA, 783–794. https://doi.org/10.1145/1807167.1807252

[2] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum Learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09)*. ACM, New York, NY, USA, 41–48. https://doi.org/10.1145/1553374.1553380

[3] Songxi Chen. 1994. Empirical Likelihood Confidence Intervals for Linear Regression Coefficients. *Journal of Multivariate Analysis* 49, 1 (1994), 24–40. https://doi.org/10.1006/jmva.1994.1011

[4] Zhiyuan Chen, Bing Liu, Ronald Brachman, Peter Stone, and Francesca Rossi. 2018. *Lifelong Machine Learning: Second Edition*. Morgan & Claypool. https://ieeexplore.ieee.org/document/8438617

[5] Peter Christen. 2012. *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer Publishing Company, Incorporated.

[6] Boyi Hou, Qun Chen, Xin Liu, Ping Zhong, Yanyan Wang, Zhaoqiang Chen, and Zhanhuai Li. 2019. Gradual Machine Learning for Entity Resolution (Technical Report). https://arxiv.org/abs/1810.12125 [Online].

[7] Eric Jones, Travis Oliphant, Pearu Peterson, et al. 2001–. SciPy: Open source scientific tools for Python. http://www.scipy.org/ [Online].

[8] Rada Mihalcea. 2004. Co-training and Self-training for Word Sense Disambiguation. In *Proceedings of the Eighth Conference on Computational Natural Language Learning, CoNLL 2004, Held in cooperation with HLT-NAACL 2004, Boston, Massachusetts, USA, May 6-7, 2004*. 33–40. http://aclweb.org/anthology/W/W04/W04-2405.pdf

[9] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep Learning for Entity Matching: A Design Space Exploration. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD 2018, Houston, TX, USA, June 10-15, 2018*. 19–34. https://doi.org/10.1145/3183713.3196926

[10] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359. https://doi.org/10.1109/TKDE.2009.191

[11] Alexander J. Ratner, Stephen H. Bach, Henry R. Ehrenberg, and Chris Ré. 2017. Snorkel: Fast Training Set Generation for Information Extraction. In *Proceedings of the 2017 ACM International Conference on Management of Data (SIGMOD '17)*. ACM, New York, NY, USA, 1683–1686. https://doi.org/10.1145/3035918.3056442

[12] Claude E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal* 27, 3 (1948), 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

[13] Dong Yu and Li Deng. 2014. *Automatic Speech Recognition: A Deep Learning Approach*. Springer Publishing Company, Incorporated.

[14] Xiaofeng Zhou, Yang Chen, and Daisy Zhe Wang. 2016. ArchimedesOne: Query Processing over Probabilistic Knowledge Bases. *Proc. VLDB Endow.* 9, 13 (2016), 1461–1464. https://doi.org/10.14778/3007263.3007284

Dixit: Interactive Visual Storytelling via Term Manipulation

Chao-Chun Hsu*
Academia Sinica
Taipei, Taiwan
joe32140@iis.sinica.edu.tw

Yu-Hua Chen*
Academia Sinica
Taipei, Taiwan
cloud60138@iis.sinica.edu.tw

Zi-Yuan Chen*
Academia Sinica
Taipei, Taiwan
zychen@iis.sinica.edu.tw

Hsin-Yu Lin
Academia Sinica
Taipei, Taiwan
swat230@iis.sinica.edu.tw

Ting-Hao (Kenneth) Huang
Pennsylvania State University
State College, PA, USA
txh710@psu.edu

Lun-Wei Ku
Academia Sinica
Most Joint Research Center for AI Technology and All Vista Healthcare
Taipei, Taiwan
lwku@iis.sinica.edu.tw

ABSTRACT

In this paper, we introduce **Dixit**, an interactive visual storytelling system that the user interacts with iteratively to compose a short story for a photo sequence. The user initiates the process by uploading a sequence of photos. Dixit first extracts text *terms* from each photo which describe the objects (e.g., boy, bike) or actions (e.g., sleep) in the photo, and then allows the user to add new terms or remove existing terms. Dixit then generates a short story based on these terms. Behind the scenes, Dixit uses an LSTM-based model trained on image caption data and FrameNet to distill terms from each image, and utilizes a transformer decoder to compose a context-coherent story. Users change images or terms iteratively with Dixit to create the most ideal story. Dixit also allows users to manually edit and rate stories. The proposed procedure opens up possibilities for *interpretable* and *controllable* visual storytelling, allowing users to understand the story formation rationale and to intervene in the generation process.

CCS CONCEPTS

• **Computing methodologies** → **Natural language generation**; Scene understanding; • **Human-centered computing** → **Interactive systems and tools**.

KEYWORDS

Interactive System, Visual Storytelling, Web Application

ACM Reference Format:
Chao-Chun Hsu*, Yu-Hua Chen*, Zi-Yuan Chen*, Hsin-Yu Lin, Ting-Hao (Kenneth) Huang, and Lun-Wei Ku. 2019. Dixit: Interactive Visual Storytelling via Term Manipulation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314131

∗ denotes equal contribution

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314131

1 INTRODUCTION

In visual storytelling, as introduced by Huang et al. [5], we take a sequence of photos as input and attempt to generate a short story narrating the sequence. This is an interdisciplinary task comprising image semantic reasoning and structural text generation. In contrast to image captioning or video captioning, which have less annotation variation across annotators, authors of visual stories can set down totally different stories for a given image sequence based on their preference and imagination. Instead of generating one story and leaving it *to* the user, we believe that working on a story draft together and iterating *with* the user results in much better story quality. In this paper, we introduce ***Dixit***[1], an *interactive* visual storytelling system. The purpose of Dixit is to enable users to control the story generation process and create their preferred stories. This system is inspired by our observation that people often start their short visual story with the key objects or actions in the pictures, and then build connections between these terms to form the final story. However, recent studies using deep neural networks (DNNs) are generally designed in an end-to-end fashion without intermediate human-recognizable representations [5, 11]. This closed process prevents users from adjusting intermediate variables and thus makes it impossible to create diverse stories from a given image sequence. In addition, as story generation is a subjective process depending on one's imagination, there is no ground truth given identical inputs, and automatic metrics fail to evaluate the quality of the story with an absolute score. In sum, we have the following four main issues:

First, **lack of interpretability in intermediate DNN Layers.** Because of the end-to-end DNN model's powerful ability as a universal approximator, in recent years it has achieved significant improvements in many fields, including computer vision and natural language processing. However, though some studies have attempted to get inside the DNN black box, it is difficult to clearly explain what information propagates to the next layer in a DNN model. For example, in visual storytelling, how exactly an end-to-end framework selects entities from images remains a mystery and the model's behavior is difficult to interpret. Second, **inability to modify generated stories.** With an end-to-end model, another problem is that the same input always yields the same result. That is,

[1]Dixit is available at https://dixitdemo.github.io/index.html . The web page will be responsive and can be used via mobile devices upon demo.

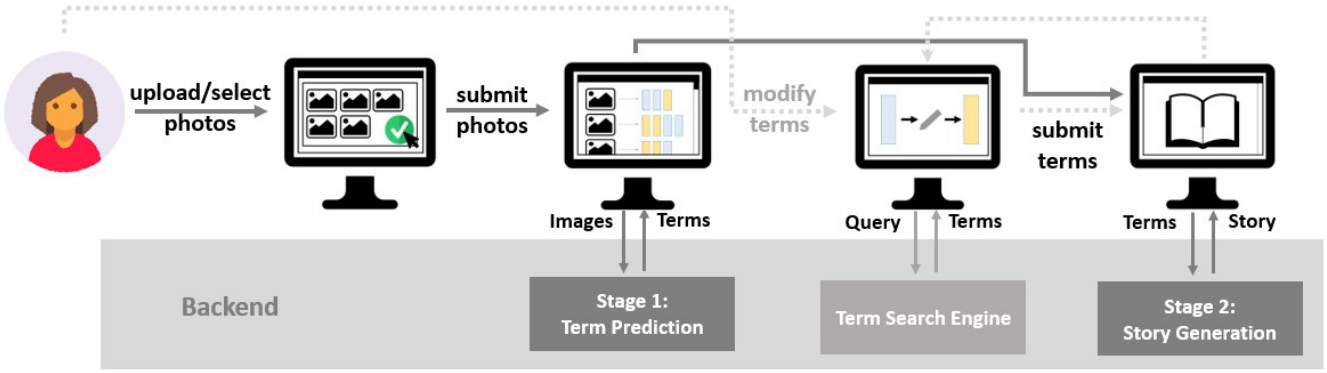


Figure 1: System architecture. The user initiates the process by uploading a sequence of photos. Dixit first extracts text *terms* from each photo which describe the objects or actions in the photo, and then allows the user to add new terms or remove existing terms. Dixit then generates a short story based on these terms.

we can obtain only a single story given the same image sequence – but this goes against the creative nature of storytelling. Although some generative models such as the variational autoencoder and the generative adversarial network produce more results by sampling from a prior distribution, they cannot generate storylines in the desired direction. Third, **difficulty in automatic evaluation.** Automatic metrics such as BLEU and METEOR have been shown to be insufficient to evaluate the quality of a visual story generation system [11]. Due to its subjective characteristics, human evaluation, which takes money and time, has become an essential part of the storytelling task. Finally, **data scarcity.** The quantity of image-story pair data is relatively insufficient because visual storytelling is an complex task; thus data annotation is expensive and time consuming.

To address these four issues, we introduce a two-stage interactive visual story generation system with which (i) users can understand what model sees in the images as predicted terms represent the objects and actions in the images, and also observe how these terms influence story generation , (ii) users can modify terms in the first stage to control the story generation, (iii) we can obtain human evaluation by asking users to score the quality of the generated story and the whole user-system interaction record in a turn is saved for future usage, and (iv) the modular nature of the two-stage framework, which utilizes image caption data and story corpora separately, allows us to train the visual storytelling model without image-story pair data.

2 RELATED WORK

Whereas end-to-end text generation from visual contents has been widely explored for tasks such as image caption, video caption, and visual storytelling, recent work has also been done on two-stage frameworks. Alexander et al. propose two-stage style transfer for image captioning. They first extract objects and verbs from an image and generate a stylish caption using an RNN trained on story corpora [7]. However, each such caption is only a single story-like sentence and is independent of other captions; put together, the captions do not constitute a context-coherent story. Xu et al. propose a skeleton-based narrative story generation method which in the first stage extracts the skeleton of the input sentence after which they generate the next sentence according to the given skeleton, one-by-one [12]. Although this framework shares with our system the insight that key phrases are important for generating a coherent story, its skeleton is extracted from pure text and does not serve as the grounding for the story. Moreover, as the framework generates the skeleton after generating the sentence, it lacks the ability to consider all important terms together to create the best fit.

3 SYSTEM

Workflow. Users of our system either upload images from their devices or choose from an image pool. After five images are uploaded or chosen, the first stage uses a term prediction model that captures terms – i.e., the objects and actions in images – to help users understand what the model sees in the current photos. As explained in Section 4, the actions are represented by verb frames. In the second stage, the predicted terms are then concatenated and fed into a story generator to construct a story with appropriate and coherent context. Note that the terms can be modified arbitrarily to control the story generation. If users want a story that depicts objects other than those detected from the images, they can remove the terms and add desired terms; to this end the system also includes a term search function. After the story is generated, the users are asked to rate the story; the complete interaction record (including image selection and term modification) is saved in the database. We expect the routine of modifying terms and generating a story to be repeated many times, as the user controls the story generation process and gradually gets better at creating good stories. For instance, in Table 1, the generated story in the second row is not perfect because some predicted terms are, although reasonable, not preferred. However, after removing these terms and adding better ones, users create a better story. This auto-labeling is an efficient way to provide samples for the training of future story models.

User Interface. A screenshot of the user interface is shown in Figure 2. The user's work flow starts at the left-top block, where the

					
Predicted Terms	"man", "Placing", "bike"	"motorcycle", "rider", "Emptying", "bike"	"trees", "sign", "Preventing_or_letting",	"man", "Placing", "bench"	"boy", "seat", "Placing"
Story	the man was sitting on his bike.	he was riding his bike with a rider on his motorcycle.	he stopped at a stop sign from a trees.	the man sat on the bench.	the boy sat in his seat.
Modified Terms	"boy", "Placing", "bike"	"rider", "bike"	"Seeking", "trees", "forest"	"bike", "dock"	"Sleep", "seat"
New Story	the boy sat down at his bike.	he was riding his bike like a rider.	he went into the forest looking for trees.	he threw his bike at the dock.	he went back to his seat to sleep.

Table 1: Via term adjustment, story generation is focused on specific objects or actions

user is asked to upload their own images or choose from the image pool. After the users submit images to the system, the predicted terms are shown in the left-bottom block. We meanwhile provide a term searching area (right bottom) using which the user searches for and modifies predicted terms. Term search is used for nouns and verb frames. For nouns, the search function retrieves nouns containing the query sub-string. For instance, for the query *wood*, the search engine returns *wood*, *woods*, *firewood*, *woodworker*, *woodworking*, and *plywood*. While searching for verb frames, instead of directly querying by the frame name, we search for lexical units of the verb frame. Results for the query *watch* would include *seeking*, of which *watch* is a lexical unit of *seeking*. After clicking the magic wand icon in the left-bottom area, the selected terms are sent to our story generation module, after which the generated story is retrieved and shown in the left middle area. The user is then asked to rate the story via the star icons. Since the concept of terms can be unfamiliar to users, we provide a term description area in the right-top corner. When user hovers the pointer over a term, the description of that term is displayed in the term description area. For noun terms, the descriptions are related images illustrating that noun, whereas for verb terms, the descriptions are textual explanations of the verbs. In the middle, on the right, a storage area is provided for the user to temporarily store unused terms.

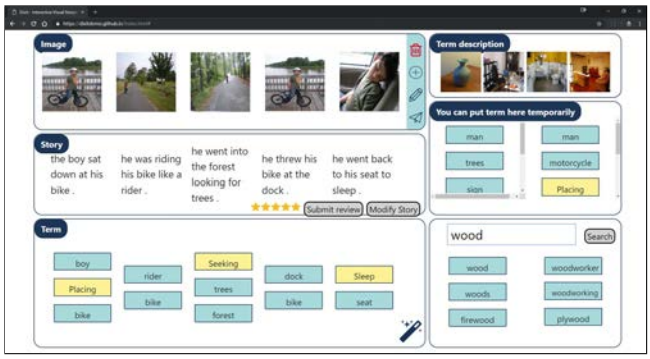


Figure 2: System user interface

4 DATA PREPARATION

Due to the modular nature of the proposed system, we train the model in two stages4 separately, without parallel data.

To train the term prediction model, we extract terms from captions in the COCO dataset [6] as the gold labels for the first stage to identity terms in images. The selection of terms is inspired by Semstyle [7]. Each sentence is represented as a combination of several noun (objects) and verb (actions) terms, which preserves the most crucial information needed to generate stories. Given a sentence, we extract verb terms by using Open-SESAME [9] to predict verb frames and leave nouns as noun terms. Frames are defined in FrameNet [1], a lexical database of semantic frame. The basic idea of FrameNet is straightforward: each frame is a description of a type of event, relation, or entity, and the participants in it.

To train the story generator, we use ROCStories Corpora [8], which contains stories with five sentences. For each story, we use a technique similar to back-translation to build our term-to-story corpora. That is, we transform stories to their corresponding terms, and then use these term-story pairs to train the story generator.

5 IMAGE-TO-TERM PREDICTION

As shown in Figure 3, the image-to-term prediction process can be considered a simplified image captioning problem which instead predicts the terms of an image. Given an image, we first convert it to the RGB format and resize it to 3x224x224 (channel * width * height) and then utilize a pre-trained Resnet-152 [3] model to extract image features. Resnet is a very deep neural network model with residual links between layers, originally pre-trained for image classification. The literature has suggested that it captures image features from low-level components like arcs to high-level concepts. With the image features produced in the previous step, we feed image features into an LSTM [4] to generate terms. Note that as

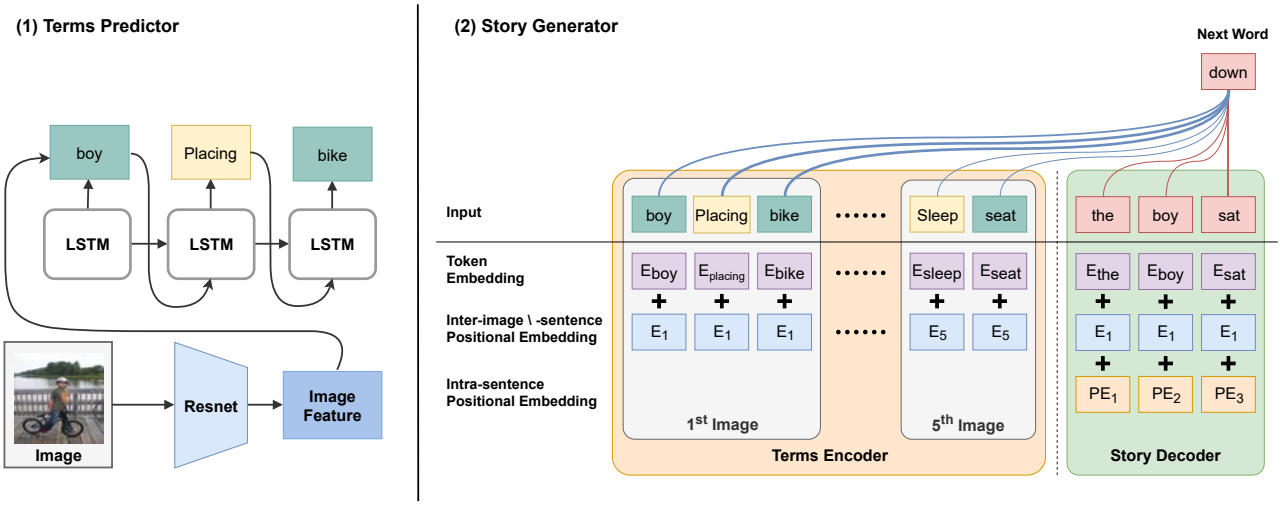


Figure 3: Architecture of (1) term prediction model and (2) story generation model. Predicted terms in green boxes denote noun terms, and terms in yellow boxes denote verb frames. In the training process, which uses the a single vocabulary, we treat both noun terms and verb frames equally. Term box colors are used here to emphasize their different origins.

described in Section 4, the LSTM model is trained with images and terms extracted from their corresponding captions in the COCO dataset; this pipeline is similar to that depicted in [7].

6 TERM-TO-STORY GENERATION

Transformer [10] is used as the basic model for story generation. In the Transformer paper, Vaswani et al. propose positional embedding as sequential information to replace the recurrent neural network. Here, we extend Transformer to be sentence-aware with the addition of an inter-sentence positional embedding to indicate the order of sentences within a story. This is shown in Figure 3. We do not consider the order of the input terms because terms represent only what appears in the images and need no sequential signal. At decoding time, an input embedding is the summation of the token embedding, the inter-image/-sentence positional embedding (InterPE), and the intra-sentence positional embedding (IntraPE). In particular,

- The InterPE is randomly initialized and updated during the training procedure. Similar to the segment embedding in BERT [2], we differentiate sentences/images by adding E_s to token embeddings of the s-th sentence/image; s denotes the order of a sentence/image in a story. $s \in \{1, 2, 3, 4, 5\}$.
- For IntraPE, we follow the Transformer implementation:

$$\text{IntraPE}_{(pos,\,2i)} = sin(pos/10000^{2i/d_{model}})$$
$$\text{IntraPE}_{(pos,\,2i+1)} = cos(pos/10000^{2i/d_{model}})$$

  where pos is the position and i is the dimension. d_{model} denotes the dimension of the input and output token. The IntraPE parameters are fixed, and the sinusoidal representation allows the model to extrapolate the sentence length that is longer than the training instances. The IntraPE PE_s is added to the s-th token embedding in a sentence. $s \in \{1, 2, ..., n\}$ and n is the max sentence length.

Based on the terms predicted for each image in the first step, we concatenate terms predicted for the five images as the input to the story generator. While decoding the story at every time step, the sentence-aware Transformer generates the context-coherent story conditioned on all input terms and the prior words of the story.

7 CONCLUSION AND FUTURE WORK

In this paper, we propose a two-stage interactive visual storytelling system that extracts terms (nouns and verb frames) and generates stories based on these terms. Users interactively adjust the terms to produce the desired story, and then record and rate the story quality for later model improvement. By splitting the task into two steps and their middle semantic term layer, the training data are not limited to the pair of images and their story, but include more accessible images with captions, and the text stories themselves. In addition, the process of visual story generation becomes interpretable. We believe this is a worthy direction for the advance of visual storytelling.

ACKNOWLEDGEMENT

This research is partially supported by Ministry of Science and Technology, Taiwan, under Grant no. MOST108-2634-F-001-004-, MOST108-2634-F-002-008- and MOST107-2218-E-002-009-.

REFERENCES

[1] Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 86–90.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.
[4] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[5] Ting-Hao Kenneth Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, et al. 2016. Visual storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1233–1239.
[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.
[7] Alexander Mathews, Lexing Xie, and Xuming He. 2018. Semstyle: Learning to generate stylised image captions using unaligned text. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 8591–8600.
[8] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A Corpus and Cloze Evaluation for Deeper Understanding of Commonsense Stories. In *Proceedings of NAACL-HLT*. 839–849.
[9] Swabha Swayamdipta, Sam Thomson, Chris Dyer, and Noah A Smith. 2017. Frame-semantic parsing with softmax-margin segmental rnns and a syntactic scaffold. *arXiv preprint arXiv:1706.09528* (2017).
[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.
[11] Xin Wang, Wenhu Chen, Yuan-Fang Wang, and William Yang Wang. 2018. No Metrics Are Perfect: Adversarial Reward Learning for Visual Storytelling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 899–909.
[12] Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A Skeleton-Based Model for Promoting Coherence Among Sentences in Narrative Story Generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 4306–4315.

Exam Keeper: Detecting Questions with Easy-to-Find Answers

Ting-Lun Hsu*
Institute of Information Science, Academia Sinica
Taipei, Taiwan
hsutingl@iis.sinica.edu.tw

Shih-Chieh Dai[1]
Institute of Information Science, Academia Sinica
Taipei, Taiwan
sjdai@iis.sinica.edu.tw

Lun-Wei Ku
Institute of Information Science, Academia Sinica
Taipei, Taiwan
lwku@iis.sinica.edu.tw

ABSTRACT

We present Exam Keeper, a tool to measure the availability of answers to exam questions for ESL students. Exam Keeper targets two major sources of answers: the web, and apps. ESL teachers can use it to estimate which questions are easily answered by information on the web or by using automatic question answering systems, which should help teachers avoid such questions on their exams or homework to prevent students from misusing technology. The demo video is available at https://youtu.be/rgq0UXOkb8o [1]

CCS CONCEPTS

• **Information systems → Decision support systems**; **Data analytics**.

KEYWORDS

Question Answering, Information Retrieval, Web Application

ACM Reference Format:
Ting-Lun Hsu, Shih-Chieh Dai[1], and Lun-Wei Ku. 2019. Exam Keeper: Detecting Questions with Easy-to-Find Answers. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314130

1 INTRODUCTION

The goal of the teacher is not always the same as that of the student, especially for short term goals. Although teachers always hope that their students really understand the knowledge they are imparting, students sometimes just want to complete their assignments with as little effort as possible. This, in the day when students can find answers with a quick web search, or use problem-solving, educational apps to find the answers. In addition, with the recent wide success of artificial intelligence, many worry that machines will soon be able to replace them in their jobs. This highlights the importance of practicing on problems that machines cannot easily solve.

However, from our observation, the advantage in this competition between teacher and student is with the latter. Many models and tools have been created to assist students to find answers, but as far as we know, there is no tool for teachers to monitor this process. Hence, we present Exam Keeper, which utilizes both retrieval-based and learning-based question answering techniques to provide a reference for teachers to know which questions have answers that are easily found on the web or using apps. Teachers can use this information to design their tests or homework problem sets accordingly to reduce the incidence of students copying answers, and hence to achieve the goal of effective practice and evaluation.

2 RELATED WORK

In the literature, there are several mobile applications which students can use to get answers, including *yuansouti*[2] and *afanti*[3]. The motivation behind such tools is to help students find answers when they are stuck on their homework assignments. However, many default to using these tools as soon as they see the questions, without even trying to solve the questions by themselves. Thus, these "time-saving" apps have become a nightmare for parents and teachers.

Researchers have also developed exam robots for high-school level exams such as AI-MATHS[4] for math exams in China and Todai[5] for the University of Tokyo entrance exam. Though these systems show the current level of AI systems for exams, they are neither open to the public nor used for educational purposes.

Previous systems for answering English exams have used a variety of different models to answer different types of questions. Examples include a state-of-the-art language model (LM)[6] trained on large datasets such as the One Billion Word Corpus [1] to answer fill-in-the-blank questions on English Exams, the reading strategies model [6] for reading comprehension, and the OpenAI transformer language model [4] for fit-the-best-sentence questions. With the invention of BERT [2], these questions can be answered more easily than before. Moreover, even for document-level questions such as SQuAD[5], BERT also achieves state-of-the-art performance. As a result, our system uses fine-tuned BERT models to predict the answer for each type of question.

3 SYSTEM

Fig. 1 illustrates the Exam Keeper framework. The first step is providing the exam questions and the answer keys. Exam Keeper provides two ways for users to input their data for examination: uploading existing files or using Exam Creator. Using the former method, teachers upload two files: one with the exam questions and the other with the answer keys. The two files should be formatted according to the instructions on the website. Teachers can also design and check exam questions one-by-one in the question design

*Equal contribution.
[1]The web page will be responsive and can be used via PCs upon demo.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314130

[2]http://www.yuansouti.com
[3]http://www.afanti100.com
[4]http://www.doudoushuxue.com/about/intro
[5]https://www.nii.ac.jp/userdata/results/pr_data/NII_Today/60_en/all.pdf
[6]https://github.com/tensorflow/models/tree/master/research/lm_1b

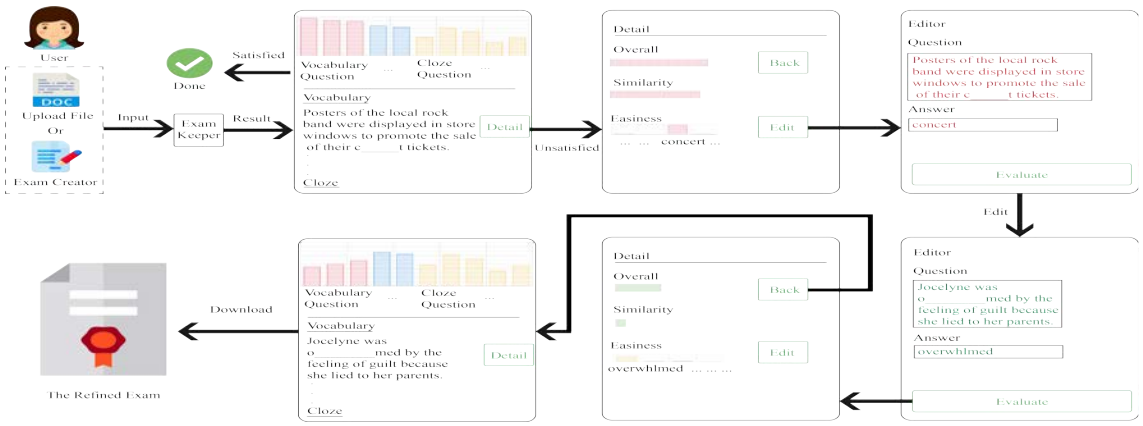


Figure 1: System flow

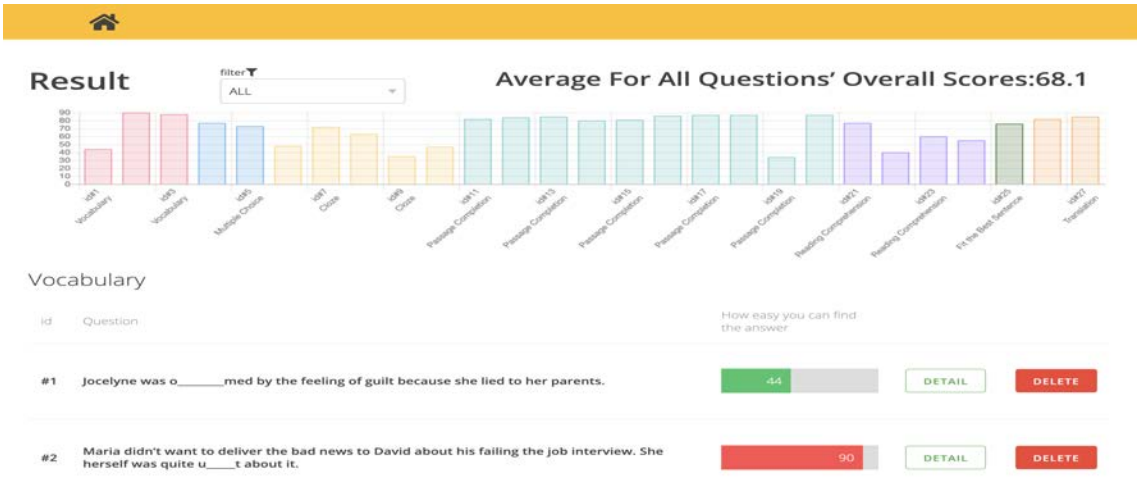


Figure 2: Result page

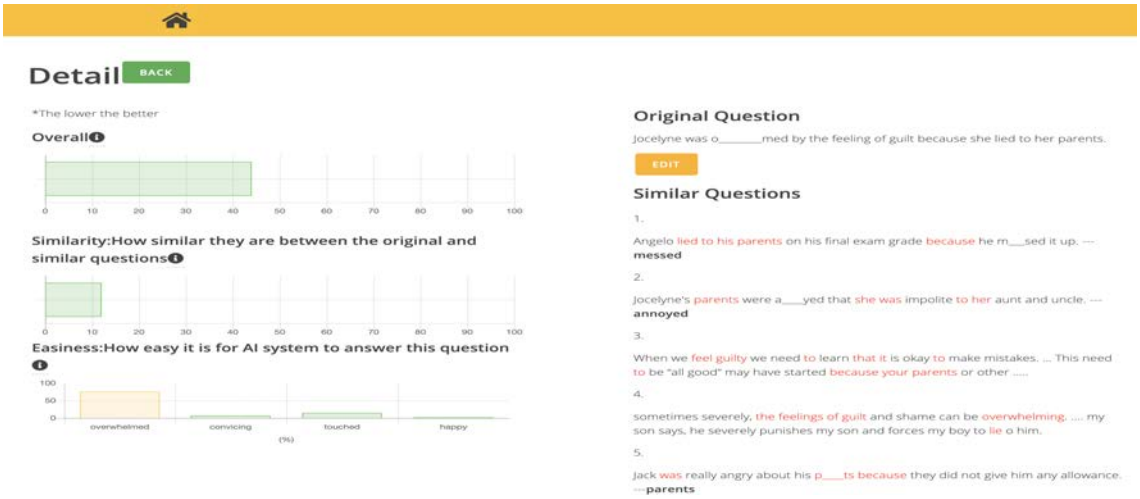


Figure 3: Question details interface

procedure by Exam Creator provided by Exam Keeper. Using Exam Creator, teachers can *test* the question and edit it accordingly until they are satisfied.

The current version of Exam Keeper supports the following question types: vocabulary, multiple-choice, cloze, passage completion, fit the best sentence, reading comprehension, and translation. Table 1 shows an example question for each type.

Exam Keeper represents the examination results at different granularities over several pages. As an overview, the top of the result page shown in Fig. 2 shows the overall accessibility of the answers in the exam and the corresponding score for each question, followed by the submitted questions with their individual scores. We mark each score with a color for straightforward visualization: red for easily-found answers (the teacher should reconsider this question), yellow for medium, and green for hard (should be good to use). The lower the score, the better the question.

The detailed calculation of the question score can be viewed clicking the *detail* button, which shows three scores: Overall, Similarity, and Easiness, as well as similar questions found on the Web and in the exam questions database. The Overall score is the average of the Similarity and Easiness. Similarity is based on how similar the original question is to the retrieved questions; it is measured in relation to answers found on the web. Easiness is measured in relation to answers found using the available QA systems or apps. Fig. 3 illustrates an example details page of a Vocabulary question, which also provides an editing function for question refinement. After going through all the questions, the teacher can download the exam sheet from Exam Creator.

4 MODELS

Exam Keeper includes several models: a retrieval-based model to retrieve similar question-answer pairs and one learning-based model for each type of question. To train these models and retrieve similar questions, we use the CLOTH [8] dataset for cloze questions, the RACE [3] dataset for reading comprehension, and the SWAG [9] dataset for fit-the-best-sentence questions. For the other types of questions, we collect raw data from free and public websites in China, Japan and Taiwan that gather exams created by English teachers.

4.1 Retrieval Based Model

We use Apache Solr[7], an extension of Apache Lucene[8], to retrieve similar questions for each input question. When a query is searched, Solr will return the documents ranked by modified Vector Space Model(VSM) score[9] with the query. VSM score of document d for query q is the Cosine Similarity of the weighted query vectors $V(q)$ and $V(d)$ whose weights are tf-idf values. Solr uses modified VSM score formula to rank documents that is not between 0 and 1. However, because we hope that the value of a score is between 0 and 1, we change it back to the original one.

The collected datasets are fed into Solr with the question type represented using the *question_type* field in Solr. For each question query, we search using Google Search and add the top ten snippets

Table 1: Example question types

Vocabulary
Maria did not want to deliver the bad news to David about his failing the job interview. She herself was quite u_____t about it.
Multiple Choice
Mangoes are a _____ fruit here in Taiwan; most of them reach their peak of sweetness in July. (A) mature (B) usual (C) seasonal (D) particular
Cloze
It is believed that taking breaks from a problem can help __1__ a moment of insight or stimulate new ideas. Unconventional solutions can also be explored. That is why some of the most successful companies in the world, such as 3M and Google, encourage their employees to __2__ all sorts of relaxing activities, such as playing pinball and wandering about the campus. 1. (A) spark (B) carve (C) drill (D) grind 2. (A) refer to (B) answer for (C) take part in (D) put up with
Passage Completion
One of these stories __1__ the Fortune cookie's origin back to 13th- and 14th-century China, which was then occupied by the Mongols. According to the legend, notes of __2__ plans for a revolution to overthrow the Mongols were hidden in mooncakes that would ordinarily have been stuffed with sweet bean paste. The revolution turned out to be __3__ and eventually led to the formation of the Ming Dynasty. (A) account (B) appeared (C) competing (D) contained (E) replaced (F) secret (G) successful (H) tastes (I) traces (J) treats
Fit the Best Sentence
On stage, a woman takes a seat at the piano. She _____ (A) sits on a bench as her sister plays with the doll. (B) smiles with someone as the music plays. (C) is in the crowd, watching the dancers. (D) nervously sets her fingers on the keys.
Reading Comprehension
For more than two hundred years, the White House has stood as a symbol of the United States Presidency, the U.S. government, and the American people. (...Skipped...) 1. What is this passage mainly about? (A) The design of the White House. (B) The location of the White House. (C) The importance of the White House. (D) The history of the White House. 2. Who initiated the construction of the White House? (A) John Adams. (B) James Hoban. (C) George Washington. (D) Thomas Jefferson.
Translation
Please translate the following sentences to English (This could be any language other than English) 1. 軍主導の暫定政権が来月予定していた総選挙の延期を示唆したことに 2. 相較於他們父母的世代，現今年輕人享受較多的自由和繁榮。 3. 成功的人都有着清晰的愿景。

[7] http://lucene.apache.org/solr/
[8] http://lucene.apache.org
[9] http://lucene.apache.org/core/7_4_0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html

into Solr according to the question type. Then we query the question in Solr and propose the top five results as similar questions. The top question's VSM score is used as the question's Similarity score in the details page.

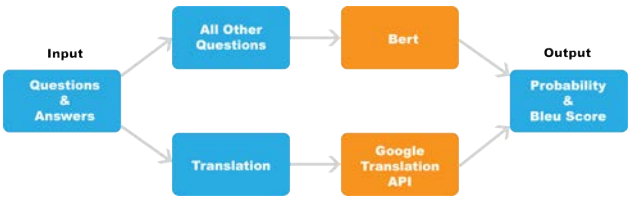


Figure 4: Model training flow

4.2 Learning Based Model

When Exam Keeper takes questions as input, based on the user-specified sections in the input file, we assign each question to one of the four categories, i.e., fill-in-the-blank, fit-the-best-sentence, reading comprehension, and translation. Models are then applied to questions by category. Fig. 4 illustrates the flow of the training process.

4.2.1 Fill-in-the-blank. This category includes four types of questions: vocabulary, multiple-choice, cloze, and passage completion. Though they appear different, they are transformed into the same format when they are input into the model: one blank masks at least one word off in a context and the model predicts the word(s). We utilize the current state-of-the-art to provide the most appropriate reference information by adopting the fine-tuned BERT [2][10] model, which is designed to solve cloze questions. After putting the context into the model, for each blank we retrieve the values for each word from BERT's dictionary. For vocabulary questions, we first search for the most likely words for the blank, and keep those words with the same first and last character given by the question. Then the top four candidates with their corresponding output values are selected. That with the maximum value is the proposed answer; the other three are also provided in the details page for teachers to use as they are similar semantically but are more difficult (for the learning models to answer). For the other questions, the softmax probability serves as the score for each answer candidate.

4.2.2 Fit-the-best-sentence. We use the BERT model designed for this type of question.

4.2.3 Reading Comprehension. During BERT's training process, in order to nail down the relationship between the contexts of two sentences, a binarized next-sentence prediction task is pretrained. This is the task we utilize to examine the reading comprehension question. In this task, the two context sentences A and B are stacked together and put into the BERT model, which predicts whether B is the next context sentence of A. In this sense, when performing downstream tasks that require information on two context sentences, we can put them into BERT as A and B and use the output of BERT's final layer for fine-tuning on each specific task.

[10]https://github.com/laiguokun/bert-cloth

Here, for reading comprehension questions, we set the article as A and each answer candidate as B, feed them to the pretrained BERT-Large model, and take the final hidden vector $C_i \in R^H$ corresponding to the first input token for each answer candidate i. The same procedure is followed for the article and question to yield C_q. In the fine-tuning stage, we use element-wise subtraction and multiplication [7] and concatenate the output of $C_q \ominus C_i$ and $C_q \otimes C_i$ to build matching representations $M_i \in R^{2H}$ for the article, question, and answer candidate. $\ominus$ and $\otimes$ are the element-wise vector subtraction and multiplication operations.

$$M_i = \begin{bmatrix} C_q \ominus C_i \\ C_q \otimes C_i \end{bmatrix} \tag{1}$$

Finally, we take the dot product of M_i and vector $V \in R^{2H}$ to get the score of i^{th} choice, where V is the only parameter needed to be trained for fine-tuning BERT on this task. Then we obtain the probability of each choice by applying softmax function over the scores of K choices.

$$Pi = \frac{e^{V \cdot M_i}}{\sum_{j=1}^{K} e^{V \cdot M_j}}. \tag{2}$$

4.2.4 Translation. We connect our application to the Google Translation API[11] and use the BLEU score between the generated translation sentence and the answer sentence to obtain the score. This score then is used as a measure of how easy it is for students to find the correct translation using Google Translate.

5 CONCLUSION

In this paper, we propose Exam Keeper, a system which calculates how easy it is for students to find exam answers. We hope this system benefits teachers in an era when technology distracts students from learning. Our future plan is to extend the application area of Exam Keeper to more subjects and to deploy it to different learning groups. In the end, we believe this is a worthwhile direction to explore for NLP techniques to improve educational environments.

ACKNOWLEDGMENTS

This research is partially supported by Ministry of Science and Technology, Taiwan, under Grant no. 105-2221-E-001-007-MY3 and MOST108-2634-F-002-008-.

REFERENCES

[1] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. 2013. One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling. *CoRR* abs/1312.3005 (2013). arXiv:1312.3005 http://arxiv.org/abs/1312.3005

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[3] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard H. Hovy. 2017. RACE: Large-scale ReAding Comprehension Dataset From Examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. 785–794. https://aclanthology.info/papers/D17-1082/d17-1082

[4] Alec Radford. 2018. Improving Language Understanding by Generative Pre-Training.

[5] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016,*

[11]https://cloud.google.com/translate/

Austin, Texas, USA, November 1-4, 2016. 2383–2392. http://aclweb.org/anthology/D/D16/D16-1264.pdf

[6] Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. 2018. Improving Machine Reading Comprehension with General Reading Strategies. *CoRR* abs/1810.13441 (2018). arXiv:1810.13441 http://arxiv.org/abs/1810.13441

[7] Shuohang Wang, Mo Yu, Jing Jiang, and Shiyu Chang. 2018. A Co-Matching Model for Multi-choice Reading Comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*. 746–751. https://aclanthology.info/papers/P18-2118/p18-2118

[8] Qizhe Xie, Guokun Lai, Zihang Dai, and Eduard H. Hovy. 2018. Large-scale Cloze Test Dataset Created by Teachers. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. 2344–2356. https://aclanthology.info/papers/D18-1257/d18-1257

[9] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A Large-Scale Adversarial Dataset for Grounded Commonsense Inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. 93–104. https://aclanthology.info/papers/D18-1009/d18-1009

ExQuisiTe: Explaining Quantities in Text

Yusra Ibrahim
Max Planck Institute for Informatics
Saarbrücken, Germany
yibrahim@mpi-inf.mpg.de

Gerhard Weikum
Max Planck Institute for Informatics
Saarbrücken, Germany
weikum@mpi-inf.mpg.de

ABSTRACT

Web pages and other documents often contain tables to provide numerical details in a structured manner. Typically, the text explains and highlights important quantities, often using approximate numbers and aggregates such as totals, averages or ratios. For a human reader, it is crucial to navigate between text and tables to understand the key information in its context, drill down into tables when details are needed, and obtain explanations on specific figures from the accompanying text.

In this demo, we present ExQuisiTe: a system to align quantity mentions in the text with related quantity mentions in tables, and enable extractive summarization that considers table contents. ExQuisiTe links quantity mentions in the text with quantities in tables, to support user-friendly exploration. ExQuisiTe handles exact single-cell references as well as rounded or truncated numbers and aggregations such as row or column totals.

CCS CONCEPTS

• **Applied computing** → **Document management and text processing**; • **Information systems** → *World Wide Web.*

KEYWORDS

Information Extraction, Quantities in Text, Web Tables

ACM Reference Format:
Yusra Ibrahim and Gerhard Weikum. 2019. ExQuisiTe: Explaining Quantities in Text. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3308558.3314134

1 INTRODUCTION

Motivation. The Web contains a wealth of pages with embedded tables, and reports with spreadsheets are abundant in enterprises. Such documents, with financial or statistical data, are challenging to read, as they are often packed with numbers and tables. For example, in a financial report, a reader can stumble upon a statement like "..overall revenues were up 21 percent year-over-year... ", giving rise to questions such as: "What was the revenue of the previous year?" or "Which particular product or sector contributed to this increase?". In such cases, the table(s) accompanying the text can provide answers. However, long documents contain several tables, and table cells are referenced at many spots throughout the report. Moreover, many textual references round or truncate numbers, or refer to aggregates such as row or column totals, which are not explicitly given in the table(s). Therefore, it is tedious work to navigate between text and tables to answer the reader's questions.

Generally, what a reader would desire is an easy and seamless way of drilling down from text passages to the relevant table cells for additional detail, and zooming out from tables to the relevant sentences that explain the numbers.

Contribution. To address the above desiderata, we propose *ExQuisiTe*, a system that identifies relations between quantities in text and tables. ExQuisiTe automatically detects these relations and generates an easy-to-read document where numbers in text are linked to their source tables and respective cells. It identifies simple mentions of single-cell table quantities as well as mentions of aggregate quantities. For example, in Figure 1 the mention "1,683" in the text refers to a simple quantity in the table; and in Figure 2

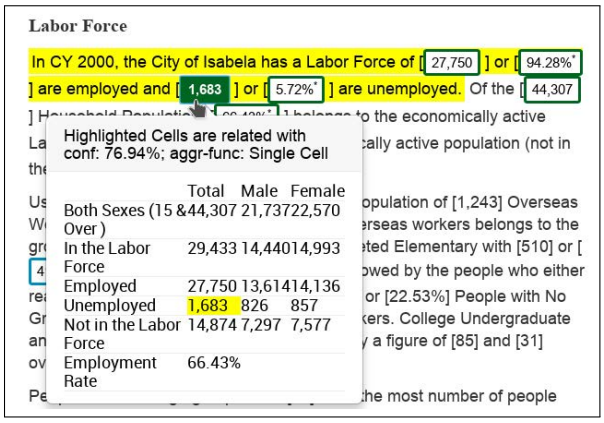


Figure (1) Simple quantity reference.

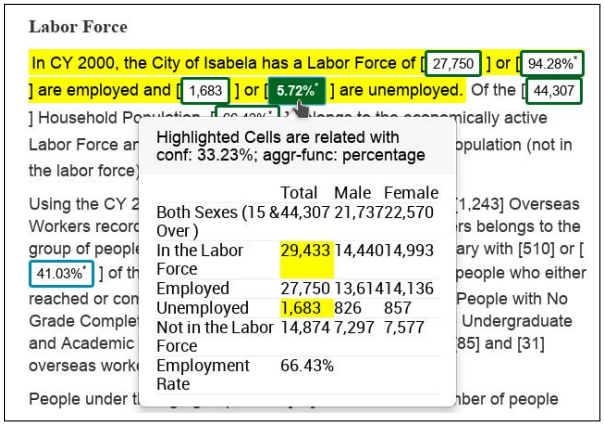


Figure (2) Aggregate quantity reference.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314134

the mention "5.72%" refers to an aggregate quantity (percentage) in the same table.

Furthermore, ExQuisiTe can guide *Extractive Text Summarization (ETS)* systems by emphasizing sentences with aggregate quantities. Current summarization systems [2, 9] do not include table data, and ExQuisiTe opens the opportunity for them to harness table data. Once ExQuisiTe identifies references of simple and aggregate table quantities in the text, it can suggest sentences with aggregations to be included in the summary generated by the ETS algorithm.

For example, in Figure 2 the highlighted sentence covers more cells in the table than the other sentences. It contains more aggregate mentions, and hence it provides a better summary, with judicious consideration of the numbers in the tables. ExQuisiTe is base on the BriQ algorithm [6], and consists of four configurable stages: (i) Document Extraction, (ii) Local Resolution, (iii) Global Resolution, and (iv) Markup and Summary Generation. The first stage extracts text segments and their possible related tables using string similarity measures. The second stage identifies potential alignments between quantity mentions in text and tables based on local features. Then, the third stage collectively aligns quantities in the text to their relevant quantities in tables. Finally, the system generates markup for the document with the inferred alignments and selects important sentences for summarization.

The code of ExQuisiTe as well as all the annotated data used for training is available on the project web page [1]. Our main contributions are:

- an end-to-end system for quantity alignment,
- a system that generates salient suggestions for a downstream ETS method,
- an open-source efficient pipeline that can be flexibly configured on a Spark cluster for online document processing.

2 COMPUTATIONAL MODEL

Our algorithm handles the following inputs:

- a piece of text, like a (part of a) web page, with a set of m text mentions of quantities $X = \{x_i : i = 1, \ldots, m\}$,
- a table q with r rows and c columns and a set of n mentions of quantities $T = \{t_j : j = 1, \ldots, n\}$.

Text mentions include terms containing numbers or numerals such as "123 patients", "37K EUR", "1.5%" or "twenty pounds".
Table mentions include two types of quantities. The first are *simple mentions*, such as '1,683' in Figure 1. Given a table with r rows and c columns we have at most $r \cdot c$ single-cell quantity mentions. The second type is *aggregate mentions*, computed as an aggregation of one or more table cells, such as '5.72%' in Figure 2.

In this demo we consider the following *aggregate functions*: average, sum, difference, percentage, and change ratio.

For aligning quantity mentions between text and tables, we aim to compute as output a subset of mention pairs $\langle x_i, t_j \rangle$ where $x_i \in X$ is a text mention and $t_j \in T$ is a table mention, including aggregate quantities. These pairs should denote the same quantity with high confidence.

[1] https://www.mpi-inf.mpg.de/briq/

3 SYSTEM COMPONENTS

3.1 Document Extraction

Long web pages can cover a variety of thematic aspects, along with several tables. Therefore, we decompose the input web page into coherent segments which we refer to as *documents*. We define a coherent document to be a paragraph together with all "related" tables from the same web page. Each document can be processed independently from the other documents. Hence, we can leverage a distributed computing framework, Spark, for online page processing.

This module first decomposes the input web page into paragraphs, then recognizes related tables for each paragraph using pairwise similarities between all paragraphs and all tables in the web page.

3.1.1 Quantity Extraction. For each document, quantity mentions are extracted from the text and the tables, using regular expressions. Our method pays particular attention to the challenges of aggregated quantities (e.g., column totals). Therefore, we generate table candidates as combinations of cells with an associated aggregate function. For example, we generate aggregate mention for a column total even if the table does not explicitly show the total. Aggregate quantities are automatically generated by considering (i) all rows and columns for totals and averages and (ii) all pairs of cells in the same row or column for difference, percentage, and change ratio. We prune the aggregate quantity candidate, to ensure computational tractability and to control spurious matches.

3.2 Local Resolution

This module first computes features for each text mention and each table mention by analyzing the surrounding context. Then, it computes similarity-based features for each pair of text mention and table mention including aggregate quantities. After that, it uses a binary classifier that accepts or rejects candidate mention-pairs. This binary classifier assigns a confidence score to each mention-pair, and we use this score in the following steps. At the end of this module, we filter the candidate mention-pairs according to their confidence score and other measures which we will explain later.

3.2.1 Mention-Pair Classification. We use manually annotated web pages with ground-truth alignment to train a *Random Forest (RF)* classifier. The classifier operates *locally* in the sense that it predicts the alignment confidence for each mention-cell pair in isolation, It serves two purposes: First, it enables the subsequent filtering step, which significantly reduces the number of candidate pairs for achieving an acceptable running time in the global resolution step. Second, it provides a prior for that global resolution step.

3.2.2 Classifier Feature. For the mention-pair classifier, we designed a variety of features that capture information a human reader would use in order to determine if text mention x and table cell t denote the same quantity. This includes *surface form similarities*, *context features*, and *quantity features*. For more details refer to Ibrahim et al. [6].

3.2.3 Adaptive Filtering. This stage reduces the number of mention-pair candidates from 1000s of candidates to 100s for tractability of global inference algorithms. We design the *adaptive filtering*

Sales were up 5% on both a reported and organic basis, compared with the second quarter of 2012. Segment profit was up 11% and segment margins increased 60 bps to 13.3%.

Table 1: Transportation Systems

($ Millions)	2Q 2012	2Q 2013	% Change
Sales	900	947	5%
Segment Profit	114	126	11%
Segment Margin	12.7%	13.3%	60 bps

Table 2: Automation & Control

($ Millions)	2Q 2012	2Q 2013	% Change
Sales	3,962	4,065	3%
Segment Profit	525	585	11%
Segment Margin	13.3%	14.4%	110 bps

Figure (3) Example with Coupled Quantities

algorithm to work in two stages. In the first stage, we develop a *text mention tagger* to predict the aggregation function for each text mention or tag the mention as a single-cell match. Then, we prune mention-pairs based on this tagger's outcome. In the second stage, we prune mention-pairs based on *value difference* and *unit mismatch*. Finally, we sort mention-pairs according to classifier scores and select top-k mention-pairs for each quantity mention based on *mention type* and *score distribution*. For more details refer to Ibrahim et al. [6].

3.3 Global Resolution

This module takes as input the candidate mention-pairs from the classifier and outputs the alignment of quantity mentions. We harness dependencies among mentions to resolve ambiguities. Consider the example in Figure 3. The text mentions "11%" and "13.3%" have exact matches in both of the shown tables, and local-resolution algorithms cannot infer the proper alignment. However, when considering these two mentions jointly with "60 bps" and "5%", it becomes clear that all of these refer to the first table.

We devised an unsupervised algorithm for this kind of global resolution. The algorithm encodes dependencies among mentions into a graph and uses *Random Walks with Restarts(RWRs)* to infer the best joint alignment.

3.3.1 Graph Construction. We construct an undirected weighted graph $G = (V, E)$ for each document:

- The node set V consists of all quantity mentions in the document's text and tables.
- The edge set E consists of three kinds of edges connecting related nodes: text-text edges, table-table edges, and text-table edges as explained below.

(i) Text-text edges: connects each pair of text quantity mentions that are within a certain proximity or have similar surface forms. Edge weights are computed based on a linear combination of proximity and string similarity. *(ii) Table-table edges:* connects each pair of table quantity mentions in the same row or the same column of the same table, and edge weights are set uniformly. *(iii) Text-table edges:* connects each pair of text and table mention that is kept by the adaptive filtering stage, and edge weights are set to the confidence scores returned by the classifier. After this initial graph construction, all edge weights are normalized to obtain a stochastic graph, via dividing each node's outgoing weights by the total weight of these edges.

3.3.2 Graph Algorithm. In our setting, we employ random walks with restart: starting from a text mention, the graph is stochastically traversed, with a certain probability of jumping back to the initial node. This technique is also known as topic-specific or personalized PageRank [4]. Our implementation iterates RWRs for each text mention until the estimated visiting probabilities of the candidate table mentions change by less than a specified convergence bound. This way we obtain a ranked list of table mentions for each text mention x.

Alignment decisions: The RWR from text mention x computes the stationary probability $\pi(t|x)$ for each table mention t. Pair $\langle x, t^* \rangle$ forms an alignment if and only if (i) t^* is the table mention with the highest overall score, and (ii) its overall score exceeds the defined confidence threshold. We then exploit the alignment decisions to update the graph, such that after identifying an alignment $\langle x, t^* \rangle$, x we modify the graph by removing all edges (x, t) for any $t \neq t^*$.(If no alignment is found for x, then all text-table edges adjacent to x are removed.)

3.4 Markup and Summarization

This module integrates the output of our system with the content of the web page and displays the results to the user in the form of an HTML page. This module is also responsible for analyzing the aligned quantities and highlighting the important sentences for the summarization engine. It estimates the importance of a sentence based on its coverage of table cells.

We define the *coverage of a quantity mention* in the text to be the number of individual single-cells it refers to. For mentions referring to an aggregate table quantity such as the sum of a column, we include all the individual single-cells in this column. Then, we compute the *coverage of a sentence* as the sum of the coverage of its mentions.

For each table, we extract all the sentences that reference it. Then, we compute a score for each sentence based on its coverage. After that, we highlight the sentence with the maximum score in the generated HTML. For example in Figure 4 even though the second sentence has more simple quantity references to the table than the last sentence, the latter provides a better summary of the table. The last sentence discusses the overall CO2 emission in the world and has the highest coverage of table cells, while the second sentence only discusses the emission of India and the EU.

4 EXPERIMENTAL RESULTS

We trained and evaluated our system on a manually annotated corpus of 495 web pages. The F1 score of our system is 79% for the simple quantity mentions, and 40% for the aggregate mentions. We carried out a run time analysis on a Spark cluster with 10 executors, each with 6 cores and 30GB of memory, and with 50GB of driver memory. The throughput of our system is 2.5K documents per minute. For full evaluation results refer to Ibrahim et al. [6].

5 DEMO OVERVIEW

In this demonstration, we will show how ExQuisiTe aligns quantity mentions in web pages. ExQuisiTe is web-based therefore it only requires a modern web browser. It has two main views: the first view is for configuring the system and selecting the type of input document, and the second view is for displaying the alignment results.

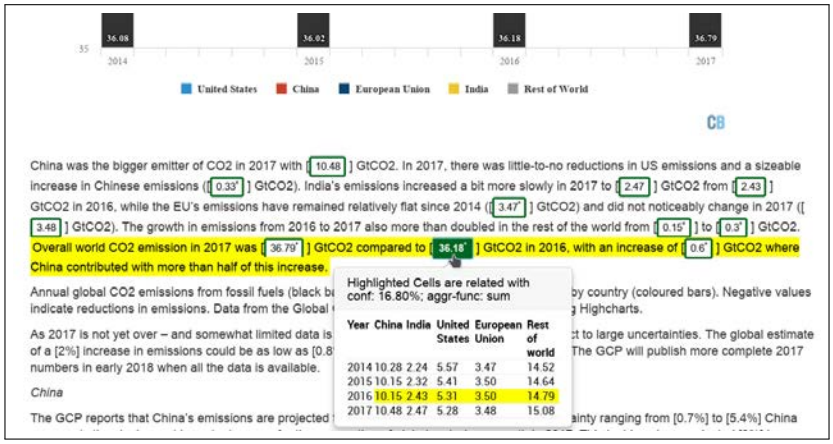


Figure (4) The figure shows the results of the system as described in Section 5

Configuration and Input. ExQuisiTe gives the user control over the settings of the different components. The user can select the type of input she wants to process. Currently, we support HTML input given as a valid page URL or a file containing the HTML content.

For the global resolution the user can choose either (i) Random Walk With Restart (RWR) or (ii) No Global Resolution. RWR is the default option. The second option deactivates the global resolution module and uses only the outcome of the classifier. It uses the confidence value given by the classifier to select the highest-confidence table mention for each text mention.

The user can adjust the threshold for the final confidence score: mapping the text mention to a table mention or to a NIL. In the case of local resolution only, this threshold is applied to the classifier's confidence score. In the case of RWR global resolution, the threshold is applied to the RWR outcome. Finally, there is an option to turn adaptive filtering on or off.

Results Display and Interactive Exploration. The system processes the document according to the user's configuration. Then, it displays the results embedded in the original HTML document as shown in Figure 4.

The text quantity mentions appear between square bracket. Each text mention is hyperlinked to its aligned table mention using a colored button with the mention's surface form as the hypertext. The color of the button is determined by the table, such that each table is assigned a color and all its related mentions in the text are assigned the same color.

The system marks the aggregate quantity mentions with a '*' superscript. When a text mention is clicked, the system displays a pop-up with the table. This pop-up includes—in addition to the table—the confidence of the alignment, the type of alignment, and the related cells in the table. The type of alignment can take one of the following values: single-cell, average, sum, difference, percentage, and change ratio.

The system marks the salient sentences for downstream summarization with yellow background. For each table, the system marks the sentence with the highest coverage. In Figure 4, the last sentence has the highest coverage of table cells.

6 RELATED WORK

Although information extraction research has targeted Web tables, no prior work has examined the relation between mentions in the text and tables within a document. Quantity annotation has been addressed in [5, 12], but these methods rely on external knowledge bases, linking table cells to entries in the knowledge base. Further methods focused on named entities, by annotating table cells with entities and classes from the knowledge base [1, 3, 8, 10, 11]. Table data fusion for search and schema inference was studied in [7, 13–15].

Our work differs from all these prior works in two main aspects: (i) we do not rely on any external knowledge base, and (ii) we handle approximate and aggregated quantities mentions which do not have exact matches.

REFERENCES

[1] C. S. Bhagavatula, T. Noraset, and D. Downey. 2015. TabEL: Entity Linking in Web Tables. *ISWC* (2015).
[2] M. Gambhir and V. Gupta. 2017. Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review* 47, 1 (2017), 1–66.
[3] A L. Gentile, P. Ristoski, S. Eckel, D. Ritze, and H. Paulheim. 2017. Entity Matching on Web Tables: a Table Embeddings approach for Blocking. *EDBT* (2017).
[4] T. H. Haveliwala. 2003. Topic-Sensitive PageRank: A Context-Sensitive Ranking Algorithm for Web Search. *TKDE* (2003).
[5] Y. Ibrahim, M. Riedewald, and G. Weikum. 2016. Making Sense of Entities and Quantities in Web Tables. *CIKM* (2016).
[6] Y. Ibrahim, M. Riedewald, G. Weikum, and D. Zeinalipour-Yazti. 2019. Bridging Quantities in Tables and Text. In *ICDE 2019*. IEEE.
[7] O. Lehmberg and C. Bizer. 2017. Stitching Web Tables for Improving Matching Quality. *PVLDB* (2017).
[8] G. Limaye, S. Sarawagi, and S. Chakrabarti. 2010. Annotating and Searching Web Tables Using Entities, Types and Relationships. *VLDB* (2010).
[9] A. Nenkova and K. McKeown. 2012. A survey of text summarization techniques. *Mining text data* (2012).
[10] D. Ritze and C. Bizer. 2017. Matching Web Tables To DBpedia - A Feature Utility Study. *EDBT* (2017).
[11] D. Ritze, O. Lehmberg, and C. Bizer. 2015. Matching HTML Tables to DBpedia. *WIMS* (2015).
[12] S. Sarawagi and S. Chakrabarti. 2014. Open-domain quantity queries on web tables: annotation, response, and consensus models. *SIGKDD* (2014).
[13] P. Venetis, A. Y. Halevy, J. Madhavan, M. Pasca, W. Shen, F. Wu, G. Miao, and C. Wu. 2011. Recovering Semantics of Tables on the Web. *PVLDB* (2011).
[14] M. Yakout et al. 2012. Infogather: entity augmentation and attribute discovery by holistic matching with web tables. *SIGMOD 2012* (2012).
[15] M. Zhang and K. Chakrabarti. 2013. Infogather+: Semantic matching and annotation of numeric and time-varying attributes in web tables. *SIGMOD* (2013).

Can You Give Me a Reason?: Argument-inducing Online Forum by Argument Mining

Makiko Ida*
Tokyo University of Agriculture and Technology
makiko.ida@katfuji.lab.tuat.ac.jp

Gaku Morio†
Tokyo University of Agriculture and Technology
morio@katfuji.lab.tuat.ac.jp

Kosui Iwasa
Tokyo University of Agriculture and Technology
iwasa@katfuji.lab.tuat.ac.jp

Tomoyuki Tatsumi
Tokyo University of Agriculture and Technology
tatsumi-t@go.tuat.ac.jp

Takaki Yasui
Tokyo University of Agriculture and Technology
yasui@katfuji.lab.tuat.ac.jp

Katsuhide Fujita
Tokyo University of Agriculture and Technology
katfuji@cc.tuat.ac.jp

ABSTRACT

This demonstration paper presents an argument-inducing online forum that stimulates participants with lack of premises for their claim in online discussions. The proposed forum provides its participants the following two subsystems: (1) **Argument estimator** for online discussions automatically generates a visualization of the argument structures in posts based on argument mining. The forum indicates structures such as claim-premise relations in real time by exploiting a state-of-the-art deep learning model. (2) **Argument-inducing agent for online discussion (AIAD)** automatically generates a reply post based on the argument estimator requesting further reasons to improve the argumentation of participants.

Our experimental discussion demonstrates that the argument estimator can detect the argument structures from online discussions, and AIAD can induce premises from the participants. To the best of our knowledge, our argument-inducing online forum is the first approach to either visualize or request a real-time argument for online discussions. Our forum can be used to collect and induce claim–reasons pairs rather than only opinions to understand various lines of reasoning in online arguments such as civic discussions, online debates, and education objectives. The argument estimator code is available at
https://github.com/EdoFrank/EMNLP2018-ArgMining-Morio
and the demonstration video is available at
https://youtu.be/T9fNJfneQV8.

CCS CONCEPTS

• **Information systems → Social networks**; • **Networks → Online social networks**; • **Computing methodologies → Discourse, dialogue and pragmatics**.

KEYWORDS

Argumentation mining; Online forum; Neural network

ACM Reference Format:
Makiko Ida, Gaku Morio, Kosui Iwasa, Tomoyuki Tatsumi, Takaki Yasui, and Katsuhide Fujita. 2019. Can You Give Me a Reason?: Argument-inducing Online Forum by Argument Mining. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314127

1 INTRODUCTION

Online forums are some of the best places to gather divergent perspectives on a controversial topic because many participants can discuss a topic without being limited by time or location. However, most participants in online forums are less likely to show premises (i.e., reasons) for their claims [3, 5]. For example, Figure 1 shows a histogram of the number of corresponding premises of a claim in an online civic discussion [3]. The figure illustrates that claims with zero or a single premise account for a large fraction, indicating that online discussions generally lack premises.

To address this issue, we propose a novel argument-inducing online forum (Figure 2) that can be used to not only collect but also induce reasoning in online discussions. The basic architecture of the proposed forum is almost the same as that of a classic forum comprising threads, where the forum has a log-in system, and participants can freely post their comments in threads. However, the proposed forum has the following two novel key features:
(1) **Argument estimator** is a visualizer of an argument structure based on argument mining. Argument mining is a newly emerging discipline that focuses on extracting claims and premises and inferring their structures from a discourse. In this work, we extended the solid argument mining scheme presented by Stab et al. [9]. We consider each post has independent arguments (an argument consists of one claim and possibly empty premises), and inferred the argument structure via a state-of-the-art deep learning model [4]. Then, we visualized the inferred arguments in the online forum in real time, indicating a color-coded claim or premise sentences and *reasoning* links that are indicated by arrows to ensure that the participants can recognize their argument.
(2) **The argument-inducing agent for online discussion (AIAD)** is a template-based agent that uses the inferred argument structure

*Both authors contributed equally to this research.
†Both authors contributed equally to this research.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314127

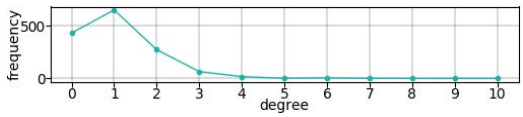


Figure 1: Histogram of a claim's degree for an online discussion. The horizontal axis indicates the claim's degree, where 0 denotes that claims without corresponding premises are counted.

that is intended to induce reasoning from participants. The agent is similar to debater agents, whereas the AIAD responds to a participant when a post contains a claim but with two or less premises. On a dialectical level, AIAD asks: "*Can you give me an additional argument for your claim?*" and encourages the user to answer a new argument accordingly.

Our experimental result demonstrates that the argument visualization and AIAD effectively support online discussions.

2 RELATED WORKS

Argumentation mining is a multi-disciplinary research field used to investigate reasoning processes such as logic, natural language processing, and machine learning. Additionally, argument mining is expected to be applicable for performing difficult tasks that require an understanding of human reasoning processes, not only for an automatic essay scoring [10] and debater system (e.g., IBM Project Debater) but also for supporting online discussions. Sato et al. [7] proposed an end-to-end debater that can generate argumentation for some values by specifying a controversial proposition. Although the debaters are successful applications of argument mining, few applications are available for online forums.

Gurkan et al. [2] proposed a deliberatorium, which is a platform for large-scale online discussions, to consider argumentation in a discussion. The platform introduces a logic-centered argumentation map, and it differs from other platforms such as Wikipedia and chat systems. Their experimental results indicated that the system can successfully provide summaries but cannot commit to mediating a discussion.

3 SYSTEM OVERVIEW

The proposed system has the following important subsystems: i) an argument estimator that functions as a visualizer of argument structures and ii) an AIAD that functions as an argument-inducing agent.

After inputting the username and password, the participants can post their opinions to a thread (Figure 2). The full thread is available on the left side of the figure. Clicking on one thread can lead to another. Moreover, creating a new thread with a controversial topic by clicking the "CREATE NEW TOPIC " button at the top right or top left in Figure 2 is permissible.

When a thread appears on the web page, the visualized argument structure discriminated by the argument estimator is shown. By focusing on the cursor of a post, we can understand the argument structure presented in the post. Each sentence or clause is color-coded, where the blue sentence indicates a premise and red indiactes a claim. The color-coded arrows in posts point out the inner-post relationship that represents a support or attack reasoning: the arrow direction is a conclusion, while the tail is a premise. Because we introduce the scheme of an argument tree, an argument comprises one claim and a (possibly empty) set of premises, and a maximum of one relation source for each sentence is allowed; the relationships are always trees. In addition, we can understand the inter-post interaction (IPI) that represents a support or attack relation from the sentences of the focused post. For example, a kangaroo's post involves two premises, two claims, and one post-to-post relationship between the post and its parent post, as shown in Figure 2.

Figure 3 shows an overview of the system's flow when the participants post comments on a forum. Our forum comprises the following four parts: user interface on a web browser, server-side database, argument estimator, and AIAD. We developed an original user interface and APIs that operate through the following six steps: (1) The participant posts a comment to our forum. (2) The document in the post is parsed into sentences or propositions and recorded into the database. (3) The argument estimator module detects the change of record in the database, discriminating either claim or premise and relations between the claim and premises in the thread. (4) The inferred argument structure is recorded by the database; thus, the argument is visualized on the user interface. (5) The AIAD, in turn, detects the change in the argument structure in the database. (6) The AIAD generates a comment and responds to a poorly reasoned post.

3.1 Argument Estimator

Argument estimator predicts argument structures through machine learning. Given that our forum supports Japanese and English discussions, we introduced a different technique and dataset for each language to train the estimator.

In Japanese discussions, we use the same dataset of Morio et al. [3]. Because the dataset automatically separates a document into sentences, we can consider that argument component candidates are already given. In this work, we employ a state-of-the-art model: parallel constrained pointer architectures (PCPAs) [4], which analyzes the sentence-level argument structure in threads. PCPA is a neural architecture that uses pointer networks [6], where PCPA can effectively discriminate between sentence types (i.e., claim, premise, or non-argumentative) and relations (inner-post relation (IPR) as well as IPI) in the thread.

PCPA achieves a significant reduction in its time complexity because it constrains the search space when discriminating between IPR and IPI. Formally, given n_p and n_s, which are the averages of the number of posts per thread and sentences per post, respectively, the standard approach needs $O({n_p}^2 * {n_s}^2)$ during computation. However, PCPA uses only $O(n_p * {n_s}^2)$. Therefore, our argument estimator provides a rapid result during testing, thus supporting a real-time inference. Generally, the inference process consumes less than 1 s.

In English discussions, we use an original token-level dataset obtained from *ChangeMyView*[1] in Reddit.[2] The dataset is annotated in token level (but not sentence level). Therefore, we should point

[1]https://www.reddit.com/r/changemyview/

[2]The dataset is not yet publicly available; however, we will make it public soon.

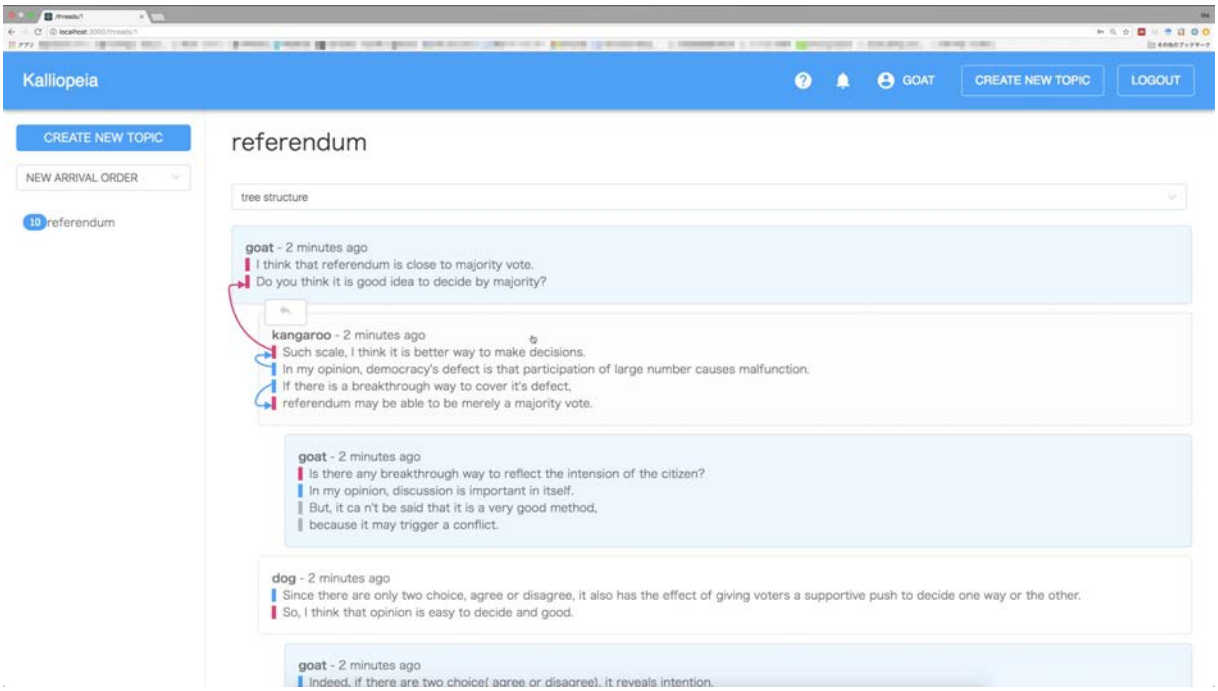


Figure 2: Thread view of our online forum. Arrows appear only when the cursor is placed above a post. The discussion was conducted in Japanese. Therefore, all posts are translated from Japanese into English; however, this forum can also support English discussions.

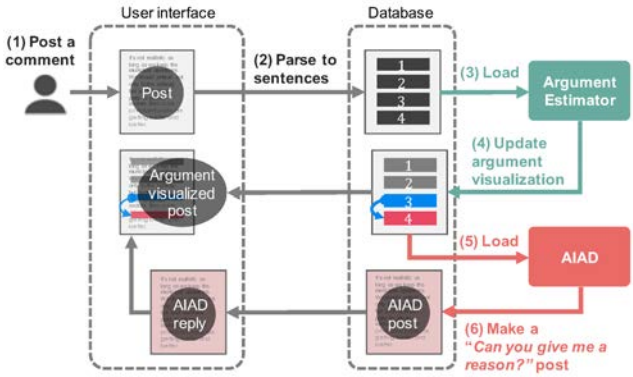


Figure 3: System Overview

out the proposition boundaries before using PCPA because PCPA assumes that propositions are given.[3] Hence, the task is considered to be a sequence tagging problem (this procedure corresponds to (2) in Figure 2). We exploit the bi-directional LSTM (BiLSTM) and conditional random fields (CRFs), which is commonly used for the argument mining task [1, 8]. After proposition detection, we employed PCPA in the same manner as in the Japanese discussion.

[3]This approach is not anomalous. Potash et al. [6] also provided the assumption that proposition boundaries are already given.

3.2 AIAD

According to the results obtained from the argument estimator, the AIAD module, in turn, generates a response to a participant to drill down the claim by inducing its premises behind a claim, forming a *Can you give me a reason?* sentence. The possible target sentence to which AIAD responds is determined using a simple rule-based technique. The target sentence should be in the newest post for each participant, and the following restrictions are considered based on our preliminary experiments.

- The target sentence should be a claim and have fewer than two premises.
- The target post is not a reply to the AIAD.
- The target post has more than one reply after the AIAD's comments.

We generate a comment by using a template-based technique. The comment comprise the target sentence (i.e., claim sentence), greetings, and *Can you give me an additional argument for your claim?* or *Do you have any premise or evidence for this claim?*, as presented:

> Thanks for your comment!
> >> "*the target claim sentence*"
> Do you have any premise or evidence for this claim?

When	Yes	Sometimes	No
creating an own claim	4	3	9
reading other posts	7	7	2
looking back a discussion	9	3	4

Table 1: Survey related to "When did you view the argument visualization in a discussion?"

Group	A		B	
Discussion topic	1-AIAD	2-Baseline	2-AIAD	1-Baseline
#Sentences per post	3.46	2.9	2.6	1.79
#Premises per claim	2.6	1.2	2.33	0.69
Premise (%) per post	61	54	48	27

Table 2: Statistics of the participant replies to AIAD

4 EXPERIMENTS

4.1 Experimental Setting

A total of 16 Japanese undergraduate students participate in the online discussion. The participants are randomly divided into two groups (i.e., "Group-A "and "Group-B "), and the participants in each group discussed the following two topics: The first is "Limitation of activity time of a club in middle and high schools in Japan," and the second is "Ban on agreement about the starting date of companies' recruitment activities in Japan." Further, we introduce the random-inducing agent as a baseline. This agent randomly chooses the target sentence from the last post of each participant. The AIAD joins the former topic discussion of Group-A and the latter topic discussion of Group-B. The random inducing agent joins the latter topic discussion of Group-A and the former topic discussion of Group-B. In addition, the participants do not know that the comments by the agents are automatically generated.

4.2 Experimental Result

Finally, 585 posts are submitted in the discussions. Based on the survey after the discussions, 14 of the 16 participants answered that our forum is accessible for discussions. Table 1 summarizes the effectiveness of the argument visualization. The table shows that the argument estimator is extensively used when the discussion flow is reviewed.

Figure 4 shows the representative of the AIAD's response post, which can capture the Squirrel's claim and induce the participant's premises. Table 2 summarizes the statistics of participant's replies to AIAD. It presents that the participant's replying posts to AIAD tend to contain a large number of sentences, including the premises.

5 DEMONSTRATION SCENARIO

In the demonstration, we will present how our online forum works for English users. Further, the online forum will be opened during the conference dates. We will show the argument visualization using the argument estimator and then immediately present the argument structures. In addition, the argument-inducing agent (AIAD) will ask *Can you give me an additional argument for your claim?* when the reasoning is insufficient. Generally, the actual discussions occur during the demonstration.

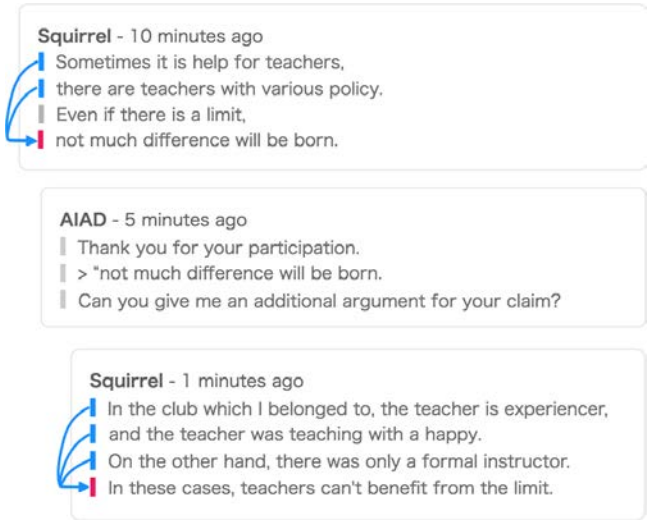


Figure 4: Post of AIAD, which successfully induced squirrel's premises. This discussion was conducted in Japanese language and then translated to English, as presented in the figure.

6 CONCLUSION

This paper presented a novel online discussion forum that can encourage participants and induce additional arguments by using argument visualization and an argument-inducing agent based on argument mining. The argument visualization subsystem visualizes the argument structures that contain claims, premises, and reasoning relations by using a state-of-the-art deep learning technique. The agent that employs a template-based technique induces the premises of their claims by replying to a participant when the premises in their comments are insufficient. The experimental results demonstrated that the AIAD is effective for inducing premises in online discussions. The future work involves proposing a more effective visualization method, thereby improving the template algorithms and a large-scale English discussion.

ACKNOWLEDGMENT

This work was supported by JST AIP-PRISM (JPMJCR18ZL) and JST CREST (JPMJCR15E1), Japan. We would like to thank Mr. Ryo Egawa for your cooperation.

REFERENCES

[1] Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. 2017. Neural End-to-End Learning for Computational Argumentation Mining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, 11–22.

[2] Ali GÃijrkan, Luca Iandoli, Mark Klein, and Giuseppe Zollo. 2010. Mediating debate through on-line large-scale argumentation: Evidence from the field. *Information Sciences* 180, 19 (2010), 3686 – 3702. https://doi.org/10.1016/j.ins.2010.06.011

[3] Gaku Morio and Katsuhide Fujita. 2018. Annotating Online Civic Discussion Threads for Argument Mining. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. 546–553.

[4] Gaku Morio and Katsuhide Fujita. 2018. End-to-End Argument Mining for Discussion Threads Based on Parallel Constrained Pointer Architecture. In *Proceedings of the 5th Workshop on Argument Mining, in conjunction with EMNLP*

2018. Association for Computational Linguistics, Brussels, Belgium, 11–21. http://aclweb.org/anthology/W18-5202

[5] Joonsuk Park and Claire Cardie. 2018. A Corpus of eRulemaking User Comments for Measuring Evaluability of Arguments. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*.

[6] Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. Here's My Point: Joint Pointer Architecture for Argument Mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1375–1384.

[7] Misa Sato, Kohsuke Yanai, Toshinori Miyoshi, Toshihiko Yanase, Makoto Iwayama, Qinghua Sun, and Yoshiki Niwa. 2015. End-to-end Argument Generation System in Debating. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*. Association for Computational Linguistics and The Asian Federation of Natural Language Processing, Beijing, China, 109–114. http://www.aclweb.org/anthology/P15-4019

[8] Claudia Schulz, Steffen Eger, Johannes Daxenberger, Tobias Kahse, and Iryna Gurevych. 2018. Multi-Task Learning for Argumentation Mining in Low-Resource Settings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, 35–41. http://aclweb.org/anthology/N18-2006

[9] Christian Stab and Iryna Gurevych. 2017. Parsing Argumentation Structures in Persuasive Essays. *Computational Linguistics* 43, 3 (2017), 619–659. https://doi.org/10.1162/COLI_a_00295

[10] Kaveh Taghipour and Hwee Tou Ng. 2016. A Neural Approach to Automated Essay Scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1882–1891. https://doi.org/10.18653/v1/D16-1193

UNVEIL: Capture and Visualise WiFi Data Leakages

Shubham Jain
Imperial College London
sjain@imperial.ac.uk

Eden Bensaid
Massachussets Institute of Technology
edenbd@mit.edu

Yves-Alexandre de Montjoye
Imperial College London
demontjoye@imperial.ac.uk

ABSTRACT

In the past few years, numerous privacy vulnerabilities have been discovered in the WiFi standards and their implementations for mobile devices. These vulnerabilities allow an attacker to collect large amounts of data on the device user, which could be used to infer sensitive information such as religion, gender, and sexual orientation. Solutions for these vulnerabilities are often hard to design and typically require many years to be widely adopted, leaving many devices at risk.

In this paper, we present UNVEIL – an interactive and extendable platform to demonstrate the consequences of these attacks. The platform performs passive and active attacks on smartphones to collect and analyze data leaked through WiFi and communicate the analysis results to users through simple and interactive visualizations.

The platform currently performs two attacks. First, it captures probe requests sent by nearby devices and combines them with public WiFi location databases to generate a map of locations previously visited by the device users. Second, it creates rogue access points with SSIDs of popular public WiFis (e.g. _Heathrow WiFi, Railways WiFi) and records the resulting internet traffic. This data is then analyzed and presented in a format that highlights the privacy leakage. The platform has been designed to be easily extendable to include more attacks and to be easily deployable in public spaces. We hope that UNVEIL will help raise public awareness of privacy risks of WiFi networks.

CCS CONCEPTS

• **Security and privacy** → **Social aspects of security and privacy**; • **Human-centered computing** → *Information visualization.*

KEYWORDS

WiFi Security, Smartphones, Visualisation, Public Demonstration

ACM Reference Format:
Shubham Jain, Eden Bensaid, and Yves-Alexandre de Montjoye. 2019. UNVEIL: Capture and Visualise WiFi Data Leakages. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314143

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314143

1 INTRODUCTION

Mobile devices like smartphones and tablets have become extremely popular over the past 10 years. Market research company Newzoo [11] reported that more than 3.3 billion smartphones and 230 million tablets were in use at the end of last year. WiFi is a core component of mobile devices, enabling them to connect to the internet. However, several vulnerabilities and design flaws have been discovered in WiFi protocols and their implementations over the years. These flaws can be exploited to track individuals [16] and to obtain sensitive information, such as religion, gender, and sexual orientation about them [1, 12].

Multiple experiments have been designed to illustrate the data leakage in WiFi networks. However, all previous approaches demonstrate specific vulnerabilities or require a user to install an application. Projects such as Wombat [8], Probr [15], and the Digital Marauder's Map [5] have demonstrated how WiFi can be used to track people's movements in real time. The Haystack Project [14] introduced an Android app to analyze mobile traffic and help individuals identify privacy leaks on their device. MITMProxy [3] is an open source and interactive HTTPS proxy that can be used to reverse-engineer installed applications and detect if any personal information is being leaked by the apps [2].

Our platform, UNVEIL, aims to raise awareness of the privacy risks of WiFi vulnerabilities through an interactive demonstration. The data is collected from nearby mobile devices (unless the owner opts out), analyzed, and the results can be viewed on installed screens. The platform modularity ensures that it can be deployed easily in public spaces like museums or train stations, making it accessible for general public.

UNVEIL currently supports two attacks:

- Generation of a map of locations visited by nearby mobile devices, using probe requests broadcasted by them;
- Profiling of users connected to our rogue access points, using the internet traffic generated by the device.

Neither attack requires users to download an app or perform any action. Moreover, the platform can be easily extended to include new attacks.

2 DATA

In this section, we describe in detail the types of data collected by UNVEIL.

2.1 Probe Requests

Probe requests are sent by WiFi-enabled devices to discover the surrounding available WiFi access points (AP). Discovering the network by scanning all possible channels and listening to beacons is not considered to be very efficient (*passive scanning*). To enhance

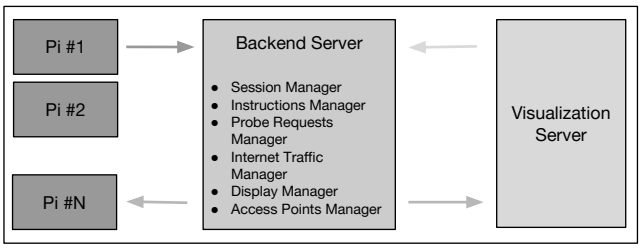


Figure 1: System architecture. UNVEIL has 3 components: Raspberry Pis, Backend Server, and Visualization Server.

this discovery process, devices often use what is called *active scanning*. In active scanning, a device still goes through each channel, but instead of passively listening to the signals on that frequency, they broadcast *probe requests* asking which known networks are available on that channel and receive responses from the APs. These probe requests contain the (possibly randomized) MAC address of the device along with the name of the AP (SSID) being probed. The device keeps a list of SSIDs of previously connected APs. We query publicly available WiFi Location databases to associate SSIDs with geographic coordinates [17, 18].

Some devices use randomized MAC addresses to prevent leaking location history through probe requests [9]. However, recent studies have shown that MAC address randomization policies are neither universally implemented nor necessarily addressing privacy concerns [7]. For instance, timing attacks such as the one described in [10] can be used to group together the probe requests (using different randomized MAC addresses) coming from the same device.

2.2 Internet Traffic

Typically, mobile devices are set by default to automatically connect to known WiFi SSIDs, without any interaction required from the user. UNVEIL creates rogue WiFi access points using popular public network SSIDs that many devices are likely to connect to. Internet traffic generated by any device connected to UNVEIL's access point is sniffed and recorded. The DNS and HTTP requests are analyzed to profile the user. The DNS requests made by the device contain the websites visited by the user and the applications they use. These might contain information allowing an attacker to infer sensitive user attributes such as gender, age group, sexual orientation, religion, mental health, race, and nationality. The HTTP requests, furthermore include the user agent, which often contains details regarding the device manufacturer, model, and version of the OS (see Fig. 3).

3 SYSTEM ARCHITECTURE

The UNVEIL platform is designed to be modular and easily extendable. It is structured into three main components: Raspberry Pis, a Backend Server, and a Visualization Server. Figure 1 shows a high-level representation of the architecture. Raspberry Pis are responsible for collecting the data. The Backend Server manages the Pis and analyzes the data. The Visualization Server runs the frontend component that retrieves the analysis results from the Backend Server and showcases them. The Visualization Server also provides the controller screen for managing the overall system. The communication between all the components takes place via the REST APIs.

3.1 Raspberry Pi

Raspberry Pis are responsible for collecting the WiFi data and for sending it to the Backend Server for analysis. The inbuilt WiFi antenna in the Pi is used to provide internet connectivity for the Pi itself, while an external USB antenna is used to collect the probe requests and make the rogue access point available.

Each Pi runs a Pi-Controller process that starts during the boot. This process runs two subprocesses in parallel. One subprocess checks and updates the state of the Pi every 10 seconds. State is represented by the mode, SSID and channel the external USB antenna is operating in. The other subprocess periodically requests the Backend Server for the instructions to be executed on the Pi and sends back acknowledgement on successful execution. The Pi-Controller maintains an extendable instruction set of operations that can be successfully executed on the Pi.

The Pi can operate external USB antenna in two modes: listening to probe requests or creating an access point depending on the instruction fetched from the Backend Server:

- *ProbeReq* - This mode is used while listening for probe requests. The operating channel changes periodically (every 5 seconds) to capture probe requests across all the commonly used channels (1, 6 and 11). SSID for the state is set to null in this mode.
- *AccessPoint* - In this mode, Pi creates an access point with the SSID and channel received from the Backend Server. It sniffs and collects all the traffic passing through, along with the MAC address of the device.

The Pi instruction set currently supports 5 operations:

- *Start ProbeReq*. Start the probe request collection.
- *Stop ProbeReq*. Stop the probe request collection.
- *Start AccessPoint*. Get an SSID and channel number from the Backend Server and create an access point.
- *Stop AccessPoint*. Stop the access point.
- *Stop All*. Stop the data collection process, if any.

Whenever any *Stop* instruction is executed, the collected data is sent to the Backend Server for analysis and is deleted from the Pi.

We use the Raspberry Pi 3B+[1] with external antenna USB WiFi adapter Alfa Network AWUS036NHA[2] and Kali Linux as the OS. The Pi-Controller provides various modules to control WiFi antennas. It is implemented in Python 3.6 and is designed to work on any device running Kali Linux with two wifi antennas. We use hostapd[3] and dnsmasq[4] to create access points and Wireshark to collect the data.

[1] https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/
[2] https://wikidevi.com/wiki/ALFA_Network_AWUS036NHA
[3] https://w1.fi/hostapd/
[4] http://www.thekelleys.org.uk/dnsmasq/doc.html



Figure 2: Visualization of the probe requests analysis. The top left corner shows the global map with markers denoting the location of all the observed SSIDs. The text in the center left contains geographic addresses of the SSIDs. The table on the bottom left displays the count of devices per manufacturer, with the most frequent ones at the top. The figure on the right contains the map of UK with the estimated count of the number of number of devices in each detected location. For privacy reasons, the numbers shown in this figure are fake. (Maps provided by ©OpenStreetMap [13])

3.2 Backend Server

The Backend Server is responsible for managing the demonstration, components, data, and analyses in the experiment (Fig. 1). Session Manager is responsible for starting, managing, and ending the demonstration session as well as archiving and cleaning the collected data once the session ends. Instruction Manager maintains a queue of instructions for each pi. The first instruction in the queue is sent when requested by the Pi. The queue is then locked for next 30 seconds or until the acknowledgment of successful execution is not received from the Pi. While the queue is locked, no instruction can be added during that interval. On receipt of successful execution, the instruction is removed from the queue.

Probe Requests Manager and Internet Traffic Manager are responsible for analysis of the captured probe requests and the internet traffic data. Probe Requests Manager associates each SSID with its geographic location (if available) and estimates a count of devices which are near the Pi and probing that SSID. Internet Traffic Manager analyzes the data generated by the devices connected to our rogue access points. All the analysis results for each session for each mode are stored in the database for visualization.

Display Manager is responsible for handling the communications with the Visualization Server. It manages the content to be displayed on the web browser based on the presenter's selections made through the control screen. The Backend Server is implemented in Python 3.6 using Django [4] and MongoDB [6].

3.3 Visualization Server

The Visualization Server is responsible for serving the web pages that visualize the results of data analyses and allow to control the demonstration. The control screen provides controls to start and stop the demonstration, display the live status of the Pis, and select which data to show on the visualization screens. The selected data is retrieved from the Backend Server for visualization.

Figure 2 illustrates the result of the probe requests analysis. The analysis shows the location of the captured SSIDs with the estimated count of the devices associated with those SSIDs.

Figure 3 shows the profile created for a connected user. Each user data is composed of 3 sections: Device Details, DNS Queries, and Internet Traffic. Device Details section displays the MAC address, manufacturer, model number, and OS of the device. DNS queries are used to identify which websites the user visited and which apps are running (including those in background). We filter out the common DNS queries (e.g. Whatsapp, Facebook, Google) to display user-specific ones. The Internet Traffic section is further separated into HTTP and HTTPS requests. HTTP requests reveal the complete URL of the request, which is shown in the visualization.

The control screen can be accessed on a browser, preferably on a laptop or a tablet. It provides buttons to start and stop the demonstration, select data to be visualized, and zoom in and out of the map illustrating the probe requests analysis.



Figure 3: Visualization of the connected users. The top section illustrates the retrieved mobile device details. The center section shows relevant DNS queries generated from the user. The bottom section shows the details of the internet traffic generated by the user. For privacy reasons, the data shown in this figure is fake.

The Visualization Server has been developed using ReactJS[5], Leaflet.js[6], and OpenStreetMap [13].

[5]https://reactjs.org/
[6]https://leafletjs.com/

3.4 Extending the Platform

The platform can be easily extended to demonstrate other data leakages in WiFi networks. The Pi instruction set can be appended with new instructions. These can be designed using the existing modules in Pi-Controller. Backend Server can be extended by adding new analysis managers which can use the available infrastructure of the database, session and instruction management, helping streamline the development and deployment process. A new web page can be added to the Visualization Server to show the analysis results and buttons can be added to the control screen for interacting with the display.

4 DEMONSTRATION

We demonstrate the platform by collecting data in real time and visualizing it. Mobile device users can interact with the geographic data using the buttons on the control screen, as well as try to infer characteristics of the profiled users from the collected data.

A session runs with only one Raspberry Pi collecting probe requests while other Pis operate as rogue access points posing as common UK public WiFis. We provide an option for the user to opt out by entering the MAC Address of their device on our website. Instructions to opt out are clearly publicized with posters around the demonstration space.

A live demonstration requires deploying Raspberry Pis around the space and at least two screens to view the results. Screens should be large enough (preferably 4K) for the audience to view the data. Control screen will be available on the tablet for the audience to zoom into the probe requests map, and see if the data from their smartphones is being captured. The users can also see if their phone has connected to our access points and what data has been collected. All the data (raw and results of analyses) are deleted at the end of each demonstration session.

The demonstration can be deployed in any public or closed space and only requires (wireless) internet connection, electricity supply for the Raspberry Pis, and two screens for visualization. Through this demonstration, we hope to show how sensitive individual-level data can be collected and processed with only £75 worth of hardware.

The platform will be open sourced before the demonstration, along with instructions to deploy and extend it. It will be made available at https://cpg.doc.ic.ac.uk/blog/unveil.

5 CONCLUSION

We have designed and developed a platform to demonstrate privacy attacks on mobile device WiFi. The platform can be easily deployed in public spaces to help raise awareness of the privacy risks in WiFi networks.

6 ACKNOWLEDGEMENTS

The authors would like to thank Imperial College London's Data Science Institute and the Computational Privacy Group, especially Axel Oehmichen, Luc Rocher, Ali Farzanehfar, and Andrea Gadotti for their support. We acknowledge support from the French Development Agency (AFD). The opinions expressed in this article are the authors' own and do not reflect the view of the AFD.

REFERENCES

[1] John S Atkinson, John E Mitchell, Miguel Rio, and George Matich. 2018. Your WiFi is leaking: What do your mobile apps gossip about you? *Future Generation Computer Systems* 80 (2018), 546–557.

[2] Timothy A Chadza, Francisco J Aparicio-Navarro, Konstantinos G Kyriakopoulos, and Jonathon Chambers. 2017. A look into the information your smartphone leaks. (2017).

[3] Aldo Cortesi, Maximilian Hils, Thomas Kriechbaumer, and contributors. 2010–. mitmproxy: A free and open source interactive HTTPS proxy. https://mitmproxy.org/ [Version 4.0].

[4] Django Software Foundation. 2018. Django (Version 2.1). Retrieved 2019-01-01 from https://docs.djangoproject.com/

[5] Xinwen Fu, Nan Zhang, Aniket Pingley, Wei Yu, Jie Wang, and Wei Zhao. 2009. The digital Marauder's map: a new threat to location privacy. In *Distributed Computing Systems, 2009. ICDCS'09. 29th IEEE International Conference on*. IEEE, 589–596.

[6] MongoDB Inc. 2018. MongoDB. Retrieved 2019-02-19 from www.mongodb.com/

[7] Jeremy Martin, Travis Mayberry, Collin Donahue, Lucas Foppe, Lamont Brown, Chadwick Riggins, Erik C Rye, and Dane Brown. 2017. A study of MAC address randomization in mobile devices and when it fails. *Proceedings on Privacy Enhancing Technologies* 2017, 4 (2017), 365–383.

[8] Célestin Matte and Mathieu Cunche. 2017. Wombat: An experimental Wi-Fi tracking system. In *8e édition de l'Atelier sur la Protection de la Vie Privée (APVP)*.

[9] Célestin Matte and Mathieu Cunche. 2018. *Spread of MAC address randomization studied using locally administered MAC addresses use historic*. Ph.D. Dissertation. Inria Grenoble Rhône-Alpes.

[10] Célestin Matte, Mathieu Cunche, Franck Rousseau, and Mathy Vanhoef. 2016. Defeating MAC address randomization through timing attacks. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. ACM, 15–20.

[11] Newzoo. 2019. Global Mobile Market Report. Retrieved 2019-01-01 from https://resources.newzoo.com/hubfs/Factsheets/Newzoo_The_Global_Mobile_Market_Report_Fact_Sheet.pdf

[12] Piers O'Hanlon, Ravishankar Borgaonkar, and Lucca Hirschi. 2017. Mobile subscriber WiFi privacy. In *Security and Privacy Workshops (SPW), 2017 IEEE*. IEEE, 169–178.

[13] OpenStreetMap contributors. 2017. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org.

[14] Abbas Razaghpanah, Narseo Vallina-Rodriguez, Srikanth Sundaresan, Christian Kreibich, Phillipa Gill, Mark Allman, and Vern Paxson. 2015. Haystack: In situ mobile traffic analysis in user space. *ArXiv e-prints* (2015).

[15] Joel Scheuner, Genc Mazlami, Dominik Schöni, Sebastian Stephan, Alessandro De Carli, Thomas Bocek, and Burkhard Stiller. 2016. Probr-a generic and passive WiFi tracking system. In *Local Computer Networks (LCN), 2016 IEEE 41st Conference on*. IEEE, 495–502.

[16] Edwin Vattapparamban, Bekir Sait Çiftler, İsmail Güvenç, Kemal Akkaya, and Abdullah Kadri. 2016. Indoor occupancy tracking in smart buildings using passive sniffing of probe requests. In *Communications Workshops (ICC), 2016 IEEE International Conference on*. IEEE, 38–44.

[17] WiFiSpc. 2018. WiFi Space. Retrieved 2019-01-01 from https://wifispc.com

[18] WiGLE. 2019. Wireless Geographic Logging Engine. Retrieved 2019-01-01 from https://wigle.net

SmartDBO: Smart Docker Benchmarking Orchestrator for Web-application

Devki Nandan Jha
Newcastle University
Newcastle Upon Tyne, UK
d.n.jha2@newcastle.ac.uk

Michael Nee
Newcastle University
Newcastle Upon Tyne, UK
info@michael-nee.co.uk

Zhenyu Wen
Newcastle University
Newcastle Upon Tyne, UK
zhenyu.wen@newcastle.ac.uk

Albert Zomaya
Sydney University
Sydney, Australia
albert.zomaya@sydney.edu.au

Rajiv Ranjan
Newcastle University
Newcastle Upon Tyne, UK
raj.ranjan@newcastle.ac.uk

ABSTRACT

Containerized web-applications have gained popularity recently due to the advantages provided by the containers including light-weight, packaged, fast start up and shut down and easy scalability. As there are more than 267 cloud providers, finding a flexible deployment option for containerized web-applications is very difficult as each cloud offers numerous deployment infrastructure. Benchmarking is one of the eminent options to evaluate the provisioned resources before product-level deployment. However, benchmarking the massive infrastructure resources provisioned by various cloud providers is a time consuming, tedious and costly process and is not practical to accomplish manually.

In this demonstration, we present **Smart D**ocker **B**enchmarking **O**rchestrator (**SmartDBO**), a general orchestration framework that automatically benchmarks (deploys and executes) users' containerized web-applications across different cloud providers while meeting the constraints of budget and deployment configurations. SmartDBO aims to answer two questions: (i) how to automate the benchmarking of containerized web-application across multi-cloud environments?, (ii) how to maximize the diversity in a benchmarking solution which covers maximum numbers of cloud providers and types of provisioned infrastructures without exceeding users' budgets? We evaluate all the features of SmartDBO using SimplCommerce and TPC-W executing across Amazon AWS and Microsoft Azure.

CCS CONCEPTS

• **Information systems → Web applications**; • **Computer systems organization → Cloud computing**; • **Software and its engineering → Cloud computing**; *Organizing principles for web applications.*

KEYWORDS

Web-application; Docker Container; Benchmark; Orchestrator; Cloud

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314137

ACM Reference Format:
Devki Nandan Jha, Michael Nee, Zhenyu Wen, Albert Zomaya, and Rajiv Ranjan. 2019. SmartDBO: Smart Docker Benchmarking Orchestrator for Web-application. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314137

1 INTRODUCTION

Web-application have become an integral part of our life providing various functions including social networking, banking, e-commerce, etc. With the growing demand, these applications are becoming complex and follow multi-tier architecture, which makes it hard to develop, debug, deploy and upgrade [4]. However, microservices architecture brings a new approach to overcome the difficulty of developing this large software by breaking it into a suite of small, independent and versioned modular services [2, 9]. Evolution of container technology with different features including self-contained, light weight, fast start up and shut down, easy scalability, etc. provides great support for developing microserviced web-applications [8]. It allows each modular service of the complex web-application to be deployed as an independent containerized web-application. The containerized web-applications can interact with each other through Restful APIs and be easily deployed, upgraded and scaled.

Additionally, cloud has become the main infrastructure for the deployment of containerized web-applications. There are about 267 cloud providers which provide various computing resources for the containerized application using 3 main *Deployment Infrastructures (DI)*: Bare Metal (BM), Virtual Machine (VM) and Container Service (CS) [5]. Each type of DI has the variety of hosts either predefined by cloud providers or user customized. For example, Amazon EC2 provides different types of pre-customized hosts (VMs) for their customers along with the self customized hosts. In the rest of the paper, we use host to represent an instance of the DI. The containerized web-application can be deployed and executed on any type of host.

Various cloud providers provision heterogeneous hosts that may significantly affect the performance of web-applications, which has a strong impact on user satisfaction [3]. Thus, we need to benchmark the hosts before deploying the real applications [6]. Most of the research in benchmarking is focused on developing a better benchmark application in terms of reality of emulation, efficiency and scalability [1, 10]. The challenge of deploying a benchmark

for containerized applications across multiple clouds has not been well explored. If a user manually performs the deployment, it will be tedious, error-prone and requires a lot of time and expertise. Due to the fact that different cloud environments have different accessing schemes and different ways to interact, an orchestrator is desired that can automatically run the benchmark applications across multiple clouds, and collect the required system parameters while providing the flexibility for users to customize their experiment settings.

Previous benchmark frameworks from both academia [1, 7, 10] and industry[1, 2] mainly focus on VM based applications, which is not applicable for containerized web-applications. Also, the variety of cloud providers offer a massive configuration choices of host. For instance, Amazon EC2 provides 43 types of hosts for their customers[3] excluding self customized hosts. Thus, this poses another challenge for the proposed orchestrator, i.e., how to provide an optimized recommendation to help users in selecting the hosts from the provided clouds?

The aforementioned challenges discussed above can be resolved by answering the following research questions:

- *How to automatically orchestrate the benchmark applications across multi-cloud environments based on the user's specification?*
- *How to find an optimal set of hosts for benchmarking in a fixed budget that covers the diversity both within a cloud provider and among the cloud providers?*

To answer these questions, we developed SmartDBO that makes the following contributions:

- We developed a noble orchestrator, SmartDBO that automates the definition and execution of benchmarks for containerized web-applications. In particular, the orchestrator allows the user to choose the benchmark applications and hosts of DI across different cloud providers.
- SmartDBO also has a native feature of optimization that can maximize the utility of a user's budget by maximizing the number of cloud providers and the hosts of the DI.

To demonstrate the effectiveness of SmartDBO, we chose AWS and Azure as the cloud providers. TPC-W[4], a traditional web-application benchmark is used for testing and the workload for testing the benchmark application is generated by the Apache JMeter[5]. Moreover, our SmartDBO captures the live performance of the examined experiments and stores the results in the Database for further analysis and reporting.

2 SYSTEM OVERVIEW

This section discusses the architecture and the implementation details of SmartDBO.

[1]https://www.cloudorado.com/cloud_server_comparison.jsp
[2]https://cloudharmony.com/status
[3]https://aws.amazon.com/ec2/instance-types/
[4]https://cs.nyu.edu/ totok/professional/software/tpcw/tpcw.html
[5]https://jmeter.apache.org/

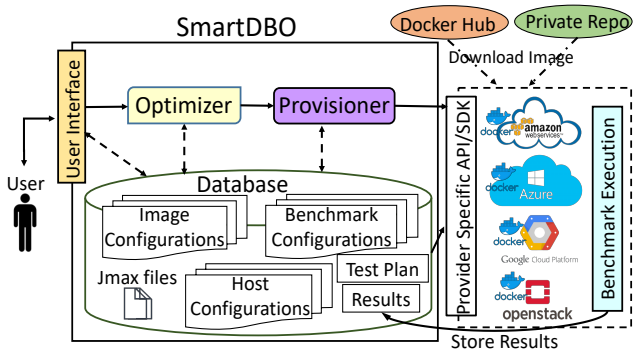


Figure 1: System Architecture of SmartDBO

2.1 SmartDBO Architecture

Figure 1 illustrates the architecture of SmartDBO and the dependencies of each component. SmartDBO is implemented as a web-application that provides a *User Interface* for users to interact, explore and manage their benchmarking experiments. The *User Interface* allows the user to choose an existing benchmark application or customize a new application. Moreover, users can select the available hosts from different cloud providers, define the benchmarking time for each selected host, and specify the total budget for running the experiments via the *User Interface*. Next, this configuration information is stored in a relational Database.

The *Optimizer* is designed to create an optimized host list based on the information provided by the user. It retrieves the necessary information (host configurations, time of benchmark and budget) from the Database and applies a heuristic algorithm to generate an optimized host list for running the benchmarking experiments. More details about the *Optimizer* are given in § 2.2. The generated host list is automatically stored in the Database. If the user chooses to execute the benchmark experiments, the *Provisioner* will be triggered to provision the resources, deploy the benchmark applications and execute the applications based on the user entered information and optimized host list. The benchmark is executed for the specified interval of time and the completion is notified to the *Provisioner*. The results are stored in the Database in real-time for further evaluation and analysis. Finally, the user is notified after completion of the experiment while releasing the cloud resources.

2.2 SmartDBO Implementation

The SmartDBO is mainly implemented by using *Microsoft ASP.NET Core 2.1 C#*[6] programming language. The *User Interface* is built using *CSS*, *JavaScript* and *HTML5* to ensure a good user experience. We aim to provide a unified interface for users to deploy their benchmark applications over multiple clouds, therefore hiding the difficulties of interacting with heterogeneous cloud APIs. Moreover, the SmartDBO also provides an optimized solution for users based on their budgets. These two main features are implemented in the modules of *Optimizer* and *Provisioner* and will be detailed as follows.

[6]https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-2.1

Optimizer. SmartDBO benchmarks containerized web-application in a multi-cloud environment. Let N represent the cloud providers $C_i | i \in \{1, N\}$ where each provider C_i has T type of hosts $v_{i,t} | t \in \{1, T\}$. In our evaluation, we assume a one-to-one mapping between host and container. Consider $C(v_{i,t})$ to be the unit cost of using $v_{i,t}$, $\tau_{i,t}$ is the time units for which $v_{i,t}$ is chosen to run and $\mathcal{B}$ is the user budget for the benchmark, finding an optimal set of hosts for the benchmarking is modeled as a Binary Integer Linear Programming problem (BILP). The defined objective function is shown below,

$$\textbf{maximize:} \sum_{\mathbf{i,t}} \mathbf{x_{i,t}} + \lambda\{\sum_{\mathbf{i}}\{\sum_{\mathbf{t}} \mathbf{x_{i,t}} - \mathbf{T}\}\}$$

where $x_{i,t} | x_{i,t} \in \{0, 1\}$ is a binary variable which represents whether $v_{i,t}$ is selected or not. The first factor of the optimization problem is to comprehend maximum selection of hosts and the second considers a penalizing factor to boost the spanning of the maximum number of cloud providers. λ is a tunable parameter which is incorporated to maintain a balance. There are certain constraints associated with the objective function as given below.

$$\sum_{i,t} C(v_{i,t}) \times \tau_{i,t} \leq \mathcal{B} \tag{1}$$

$$\forall i \forall t \; x_{i,t} \in \{0, 1\}, x_{i,t} = \begin{cases} 1, & \text{when VM, } V_{i,t} \text{ is selected} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$\forall i \forall t \; \tau_{i,t} \geq 0 \tag{3}$$

$$\forall i \; \sum_{t} x_{i,t} \geq 1, \; \forall t \; \sum_{i} x_{i,t} \geq 1 \tag{4}$$

Constraint (1) states that the total cost of benchmarking different containers running inside the host must be less than the defined budget. Also, the cost is calculated only if a host is selected as specified by the constraint (2). Constraint (3) represents variable stability constraints while constraint (4) enforces the selection of at least one cloud provider and at least one host configuration.

We developed and implemented a heuristic algorithm for *Optimizer* to solve the problem formalized above. The algorithm selects an optimized list of hosts while satisfying all the defined constraints. The source code and the implementation details of *Optimizer* is available as an open-source project on github[7]. The advantage of *Optimizer* for selecting hosts in both multi-cloud and single cloud is illustrated in Figure 2. The results show that the solution generated from the *Optimizer* covers more types of host in both single cloud and multi-cloud scenarios, compared to the case of random host selection with a given budget.

Provisioner. Once the *Optimizer* generates a benchmark plan, the user can decide whether he/she wants to submit the plan for execution via the user friendly web interface. If the user agrees to perform the experiment, the functions implemented in *Provisioner* will be triggered. First, the *Provisioner* will check the connection and the requirement of the resources on different clouds. Next, it uses a background process application, Hangfire[8], to create and launch the hosts on the selected cloud providers.

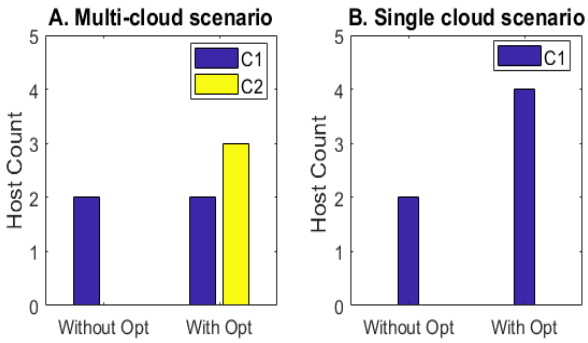


Figure 2: Showing advantage of Optimizer

3 DEMONSTRATION

In this section, we will walk through the execution workflow of SmartDBO by showing the benchmarking of a sample e-commerce web-application, SimplCommerce[9]. The SimplCommerce application is configured with 50 test products to replicate a small real-life e-commerce store.

3.1 Experiment setup

SmartDBO. The proposed orchestrator is implemented in C# language which can be executed in a variety of environments e.g. Linux, Windows, MacOS. In this demonstration, we run our orchestrator on a Lenovo PC with Intel(R) Core(TM) i5-6200U CPU @2.3GHz - 2.4GHz with a memory 16 GB and storage of 512 GB SSD. The system is installed with 64 bit Windows 10 operating system. We execute SmartDBO using Visual Studio 2017[10] IDE. It is open-sourced and the current version of code along with samples and installation details are available on github[7].

Required open-source tools. In this demonstration, we utilized a popular benchmark application called TPC-W[11] to emulate the activities of a sample e-commerce web application. The load on the web-application is created by Apache JMeter[12] according to the test plan defined by the user. To emulate real traffic, JMeter is not configured on the same cloud where the benchmark applications are running. PostgreSQL Database[13] is associated with SmartDBO, and stores different host, benchmark and image configurations for setting up the experiments in real-time.

Cloud providers. We deployed our containerized benchmark application on AWS and Azure with different host configurations. In this demonstration, we use VM as the default host configuration. SmartDBO can provision any types of VMs which are available on the subscribed cloud providers.

[7] https://github.com/smartdockerbenchmarkingorchestrator/Smart-Docker-Benchmarking-Orchestrator

[8] https://www.hangfire.io/

[9] https://www.simplcommerce.com/

[10] https://visualstudio.microsoft.com/vs/

[11] https://cs.nyu.edu/ totok/professional/software/tpcw/tpcw.html

[12] https://jmeter.apache.org/

[13] https://www.postgresql.org/

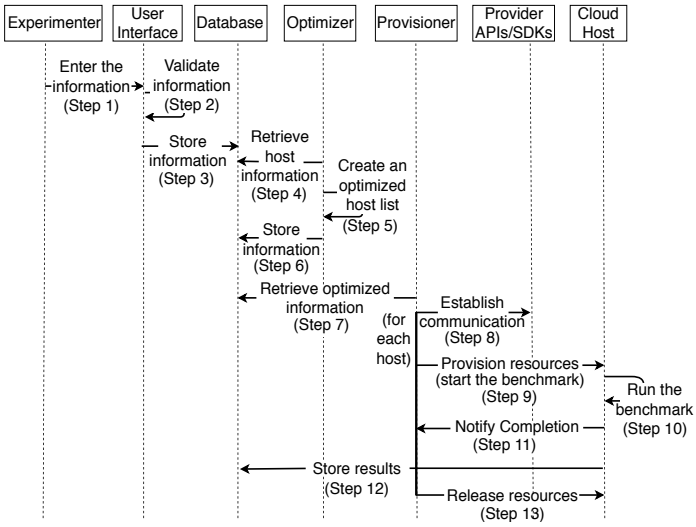


Figure 3: SmartDBO Execution Workflow

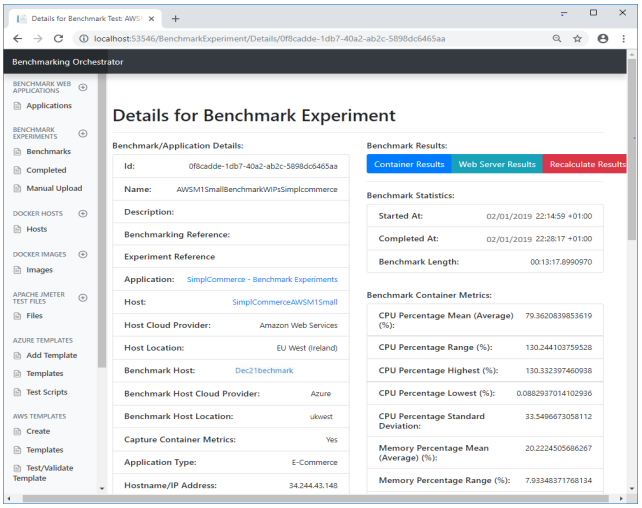


Figure 4: SmartDBO Execution Interface

3.2 The execution workflow

In this subsection, we use an example to demonstrate the process of orchestrating a benchmark application as shown in Figure 3.

Define the benchmark application. At the beginning, SmartDBO is designed for benchmarking different types of web-application. The application is defined in terms of a web-application image and a JMeter workload image using the interactive *User Interface* as depicted in Step 1 of Figure 3. The images are referred from Docker hub or other private repository or created using the provided template. For the demonstration, we create an application using SimplCommerce and a JMeter docker image.

Define the host configuration. When we decide the benchmark application, our SmartDBO allows us to define the host configuration to run the benchmark application. The *User Interface* allows us to choose a name, description, host type (application/benchmark) and cloud-specific (e.g. credentials, template) information as shown in Step 1. It also provides the HTTP and TLS authentication to guarantee secured communication.

Define the test plan. SmartDBO allows us to define the workload using a test plan where we can provide the JMX file for continuous workload generation. New test plans can be easily added by filling the template provided by the framework (Step 1). The plan includes all the parameters for JMeter to generate a continuous load for benchmarking the web-application.

Define the benchmark experiment. Once we have the benchmark application and hosts configured, we can set the whole experiment for performing the benchmarks. Then, we define the benchmark experiment which maps the load generator to the benchmarking host. To this end, we pair a load generator with the host running the benchmark web-application. Along with this, we can specify the execution time and the budget for executing the experiment. The entered data is validated (Step 2) after which it is stored in the Database (Step 3).

Perform optimization. The *Optimizer* retrieves the input host information from the Database (Step 4) to generate an optimized host list for the execution (Step 5). At the same time the optimized host list is stored in the Database (Step 6).

Execute the benchmark. *Provisioner* retrieves the optimized host list from the Database (Step 7), we can then start to execute the experiment. First the communication with the cloud host is established using the provider specific APIs/SDKs (Step 8) and then resources are provisioned for benchmarking (Step 9). The *Provisioner* is notified of the completion (Step 11) and the result is stored in the Database (Step 12) for visualization and further analysis. The experiment result provides different performance metrics including benchmark statistics (e.g. started at, benchmark length), benchmark container metrics (e.g. CPU percentage mean, CPU percentage range, memory percentage mean, network input total, network output total), benchmark web-server metrics (e.g. throughput, average response time, number of requests, number of errors, average latency) and Apdex rating (Apdex count, Apdex rating, Apdex satisfied count,). Finally, the resources are released (Step 13).

Figure 4 shows the overview of a successful execution of TPC-W benchmark using SimplCommerce web-application executing on AWS and Azure, with the obtained result.

Report the result. The benchmark execution results can be easily visualized using the *User Interface*. There are multiple options to report the results in different formats including on-screen, csv file, excel sheet, pdf for further analytics.

The complete demonstration explained here is available as a screencast video on YouTube [14].

ACKNOWLEDGEMENTS

This research was partly supported by Engineering and Physical Sciences Research Council, UK under grant EPSRC - EP/R033293/1.

[14] https://www.youtube.com/watch?v=c44FSCR3C-w&feature=youtu.be

REFERENCES

[1] Emmanuel Cecchet, Veena Udayabhanu, Timothy Wood, and Prashant Shenoy. 2011. Benchlab: an open testbed for realistic benchmarking of web applications. In *Proceedings of the 2nd USENIX conference on Web application development*. USENIX Association, 4–4.

[2] Tomas Cerny, Michael J. Donahoo, and Michal Trnka. 2018. Contextual Understanding of Microservice Architecture: Current and Future Directions. *SIGAPP Appl. Comput. Rev.* 17, 4 (Jan. 2018), 29–45. https://doi.org/10.1145/3183628.3183631

[3] Justin Cheng, Caroline Lo, and Jure Leskovec. 2017. Predicting intent using activity logs: How goal specificity and temporal range affect user behavior. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 593–601.

[4] Mariela Curiel and Ana Pont. 2018. Workload Generators for Web-Based Systems: Characteristics, Current Status, and Challenges. *IEEE Communications Surveys & Tutorials* 20, 2 (2018), 1526–1546.

[5] Maria Fazio, Antonio Celesti, Rajiv Ranjan, Chang Liu, Lydia Chen, and Massimo Villari. 2016. Open issues in scheduling microservices in the cloud. *IEEE Cloud Computing* 3, 5 (2016), 81–88.

[6] Devki Nandan Jha, Saurabh Garg, Prem Prakash Jayaraman, Rajkumar Buyya, Zheng Li, and Rajiv Ranjan. 2018. A Holistic Evaluation of Docker Containers for Interfering Microservices. In *2018 IEEE International Conference on Services Computing (SCC)*. IEEE, 33–40.

[7] Diego Lugones, Jordi Arjona Aroca, Yue Jin, Alessandra Sala, and Volker Hilt. 2017. AidOps: A Data-driven Provisioning of High-availability Services in Cloud. In *Proceedings of the 2017 Symposium on Cloud Computing (SoCC '17)*. ACM, New York, NY, USA, 466–478. https://doi.org/10.1145/3127479.3129250

[8] Dirk Merkel. 2014. Docker: lightweight linux containers for consistent development and deployment. *Linux Journal* 2014, 239 (2014), 2.

[9] Zhongshan Ren, Wei Wang, Guoquan Wu, Chushu Gao, Wei Chen, Jun Wei, and Tao Huang. 2018. Migrating Web Applications from Monolithic Structure to Microservices Architecture. In *Proceedings of the Tenth Asia-Pacific Symposium on Internetware (Internetware '18)*. ACM, New York, NY, USA, Article 7, 10 pages. https://doi.org/10.1145/3275219.3275230

[10] Joel Scheuner, Jürgen Cito, Philipp Leitner, and Harald Gall. 2015. Cloud workbench: Benchmarking iaas providers based on infrastructure-as-code. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 239–242.

R2SIGTP: a Novel Real-Time Recommendation System with Integration of Geography and Temporal Preference for Next Point-of-Interest

Xu Jiao
Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Tianjin University of Technology
Tianjin, China
jiaoxu1999@sina.com

Yingyuan Xiao*
Key Laboratory of Computer Vision and System, Ministry of Education, Tianjin University of Technology
Tianjin, China
yyxiao@tjut.edu.cn

Wenguang Zheng
Tianjin Key Laboratory of Intelligence Computing and Novel Software Technology, Tianjin University of Technology
Tianjin, China
wenguangz@tjut.edu.cn

Hongya Wang
Donghua University
Shanghai, China
hywang@dhu.edu.cn

Youzhi Jin
Tianjin University of Technology
Tianjin, China
jinyouzhi.lyg@gmail.com

ABSTRACT

With the rapid development of location-based social networks (LBSNs), point of interest (POI) recommendation has become an important way to meet users' personalized demands. The aim of POI recommendation is to provide personalized recommendation of POIs for mobile users. However, traditional POI recommendation systems cannot satisfy users' personalized demands. The reason is that the traditional POI recommendation system cannot recommend the next POI to a user based on the user's context information. Also, the traditional POI recommendation system provides no real-time guarantee on performance. In this demo, we propose a novel real-time next POI recommendation system named R2SIGTP which provides more personalized real-time recommendation compared with existing ones. Our system has the following advantages: 1) it has real-time performance; 2) it uses a unified approach to integrate geographic and preference information; 3) it considers the feedback of each single user to provide more personalized recommendation. We have implemented our system. R2SIGTP is easy to use and can be used by the mobile terminal's browser to recommend the next POI to the user in real-time based on the automatically identified user location and current time. The experimental results on real-world LBSNs show that R2SIGTP's performance is satisfactory.

CCS CONCEPTS

• **Human-centered computing** → **Social recommendation**; *User models*; Scenario-based design.

KEYWORDS

POI recommendation, tensor, sentiment analysis

ACM Reference Format:
Xu Jiao, Yingyuan Xiao, Wenguang Zheng, Hongya Wang, and Youzhi Jin. 2019. R2SIGTP: a Novel Real-Time Recommendation System with Integration of Geography and Temporal Preference for Next Point-of-Interest. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3308558.3314120

1 INTRODUCTION

POI recommendation is one of the most important tasks in LBSNs, which helps users discover new interesting locations in the LBSNs. POI recommendations using users' check-in history are often influenced by multiple factors such as geography, time, sequence, etc. Geographical influences are the most essential feature that distinguishes POI recommendations from traditional recommendations. Since a user's check-in behavior presents a spatial clustering phenomenon, geographical influence can be modeled by power law distribution[5], Gauss distribution, Poisson distribution, and kernel density estimation[6]. Temporal influence in a POI recommendation system performs in two aspects: periodicity and non-uniformness. And the sequential influence[7] is the result of the interaction of temporal periodicity, adjacency of POIs in geographical space, property of POIs and human habits. At present, most existing POI recommendation systems use the methods based on content, link analysis[1], collaborative filtering[8] or matrix factorization[2] in the recommendation process.

Although POI recommendation system has achieved great success, there are still many challenges for it. 1) Most existing POI recommendation systems use offline modeling and online recommendation system architecture. As the system runs, a large amount of incremental data is generated in real-time. To take advantage of these incremental data, the system must recompute the offline model of the system, which is very time consuming and can seriously affect the operation of the online recommendation. Obviously, this strategy is not suitable for frequent updates and affects the

*Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314120

accuracy of the recommendation system. 2) The user-POI check-in matrix is extremely sparse compared to the user-item matrix of the traditional recommendation system. Because the total number of POIs is quite large, and the number of POIs visited by a single user is very small. 3) Users' reviews are direct feedbacks to POI recommendation results. For a specific POI, most existing POI recommendation systems provide a general solution for all the users by analyzing all the corresponding reviews. Thus, it cannot provide personalized feedback for each single user. 4) Most existing POI recommendation systems integrate multi-aspect informations of check-ins. However, they do not construct a unified recommendation process to integrate these informations, that is, the implicit correlations between the various informations are ignored. 5) Most existing POI recommendation systems use Euclidean distance. This leads to deviations in actual applications and makes POI recommendation systems inaccurate.

Aiming to solve the aforementioned problems, a real-time personalized next POI recommendation system R2SIGTP was designed. R2SIGTP utilizes multi-aspect informations to model user temporal preferences. According to the current location of a user and time , R2SIGTP recommends the next POI which the user is most likely to access. R2SIGTP has the following advantages:

- It leverages a scalable parallel incremental approach to model users' check-in history, making the system real-time scalable.
- It classifies the POIs into users' preferences and uses the preferences to model users' check-in history such that the sparsity of check-in data can be overcome.
- It provides personalized feedback for each single user by using convolution neural network(CNN) to analyze the users' reviews.
- According to users' choices, it integrates geographic and preference information into the unified recommendation process.
- It effectively reduces the deviations by using road network distance for POI recommendation.

2 SYSTEM ARCHITECTURE

In this section, we introduce the architecture and background technologies of R2SIGTP. The architecture of R2SIGTP is shown in Figure 1. The detailed description is as follows.

2.1 User Temporal Preference Modeling

User temporal preference modeling is an important part of the system. The accuracy of users' preferences predicted in different time slots determines the accuracy of our system.Our system uses tensor to model user temporal preference for the following reasons:1) tensor is typically used to recover the missing/sparse data through tensor decomposition; 2) another usage of tensor is to isolate and analyze the patterns hidden in a dataset. Next, we will show the details of tensor construction and decomposition.

Tensor Construction: We define a 3 order-tensor $\chi \in \mathbb{R}^{I \times J \times K}$ to model user temporal preference, where I, J and K denote the size of user, time and preference dimension, respectively. Each entry of the tensor $\chi(i, j, k)$ is a sum of check-in frequencies of all the POIs which belong to preference k and are visited by user i in the time slot j. Note that we divide everyday into 6 time slots (i.e.

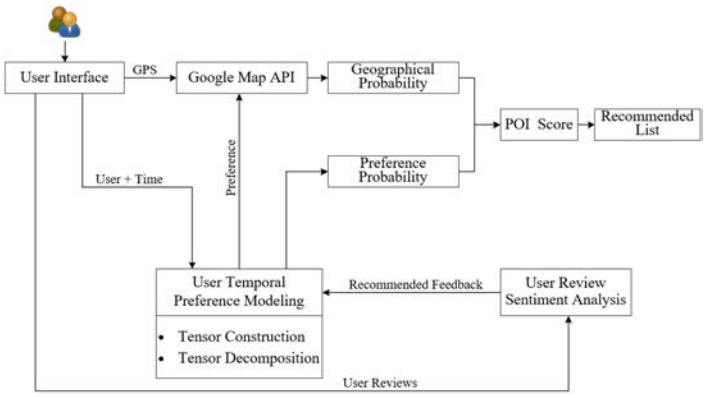


Figure 1: Architecture of R2SIGTP

0:00-7:00, 7:00-9:00, 9:00-12:00, 12:00-14:00, 14:00-18:00, 18:00-0:00). Accordingly, a week is divided into 42 time slots. Each of time slots is represented by a Time ID. In particular, we select preferences instead of the venues used in most tensor models as a dimension, which can greatly overcome the sparsity of check-in data. Because the amount of POIs in the real world is enormous, but only a few hundreds of preferences can describe our real life in detail.

Tensor Decomposition: After the tensor is modeled, our task is to infer missing entries in the tensor. We can extract the latent features of each user, time slot and preference by decomposing the tensor χ into three matrixes A, B, C and a core tensor G. As is shown in Equation (1).

$$\chi \approx G \times_1 A \times_2 B \times_3 C = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_p \circ b_q \circ c_r \qquad (1)$$

Here, $A \in \mathbb{R}^{I \times P}$, $B \in \mathbb{R}^{J \times Q}$, and $C \in \mathbb{R}^{K \times R}$ are the factor matrices. $G \in \mathbb{R}^{P \times Q \times R}$ is the core tensor. There are many methods for tensor decomposition. For our system, as a large number of users continue to access POIs, incremental data will continue to emerge. So our tensor is not static. This requires a fast and scalable tensor decomposition method for parallel processing of incremental data. Our system uses the Sambaten method [3] for tensor decomposition, which allows our system to update the user temporal preference model frequently without interrupting recommendations. Therefore, our system has real-time performance.

2.2 User Review Sentiment Analysis

Sentiment analysis of users' reviews requires knowledge in the natural language processing (NLP) field. Deep learning technology in the field of NLP has shown strong effectiveness for the reason that it satisfies various requirements such as multilayer classification representation and feature representation learning.

Our system uses CNN to analyze the users' reviews in TensorFlow. First, we do data preprocessing. The dataset our system used contains 9566 POI review sentences, half positive and half negative. Next, we construct the model(CNN). The first layer embeds words into low-dimensional vectors. The next layer performs convolutions over the embedded word vectors using multiple filter sizes. We

max-pool the result of the convolutional layer into a long feature vector, add dropout regularization, and classify the result using a softmax layer. For details, please refer to Yoon Kim's article[4].

2.3 POI Score Calculation

Generally speaking, there are two main factors considered by users when choosing a POI, preference factor and geographical factor. Specifically, the preference factor denotes the probability of the activities to be chosen by a user in a certain time slot, and the geographical factor represents the influence of distances between the prefer POIs and the user's current location. Therefore, we define POI score as follows:

$$S_{POI} = \alpha * P_p + \beta * P_G \qquad s.t. \quad \alpha + \beta = 1 \tag{2}$$

where, S_{POI} denotes the POI score, P_p is the preference probability, P_G represents geographical probability. α, β are the weights which reflect the priority of preference and geographic factors. Priority can be chosen by the user himself. After obtaining the score of each POI, we will arrange them in a decreasing order and generate a recommendation list for a user. Next, we will show the details of preference probability and geographical probability.

Preference Probability: Given a particular user i and a specific time slot t, we obtain a preference vector $v_{i,t}$ through the approximate tensor $\hat{\chi}$ constructed in subsection 2.1. Normalize $v_{i,t}$ to get $v'_{i,t}$, and each entry $v'_{i,t,j} \in v'_{i,t}$ represents the probability of each preference j which is chosen by the user i in the time slot t.

Geographical Probability: In order to improve the accuracy of the recommendation results, our system uses road network distance instead of Euclidean distance. In reality, Euclidean distance cannot represent the actual distance between two locations. Consider two locations at the two sides of a river, the Euclidean distance is small but the road network distance is relatively large. Arrange entries of $v'_{i,t}$ in a descending order, we select the first x preferences with positive value in $v'_{i,t}$. Then, for each preference, we employ Google Map API to search the corresponding POIs around the user's current location and calculate the road network distances between these POIs and the user's current location. After normalized distance of these POIs, we can generate the geographical probability.

2.4 Recommended Feedback

Users' reviews are the most direct feedback to the POI recommendation results. Most previous systems only provide a binary decision (positive or negative) for a particular POI by analyzing the reviews of all the users. However, such approach is unfair for some users because the analysis results only reflect the preference of the majority and ignore some users with opposite opinions. Therefore, our system focuses on each single user and provides a feedback to the temporal preference model by analyzing his personal reviews, leading to a more personalized recommendation results.

Without loss of generality, a user's review of a POI k would affect the probability of the preference which the POI k belongs to. Thus, our system iteratively adjusts $v'_{i,t,j}$ according to the review of user i to a POI k which belongs to preference j by using Equation (3).

$$v'_{i,t,j} = v'_{i,t,j} \times (1 + s_k * \gamma) \tag{3}$$

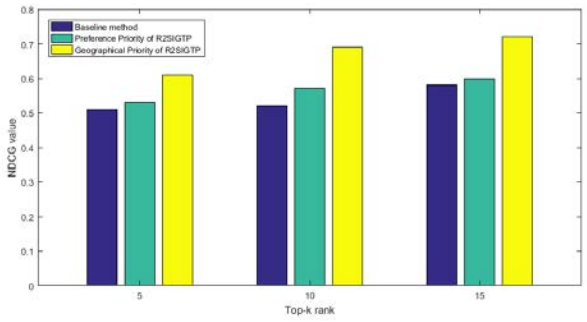


Figure 2: Performance of R2SIGTP and baseline method

where, s_k represents the result of a sentiment analysis of the user i to the POI k and γ is the tuning parameter. When the adjustment is finished, the vector $v'_{i,t}$ is normalized again.

3 EVALUATION

3.1 Dataset and Parameters

We use a collection of Foursquare check-ins of New York lasting for about 11 months (from 8 January 2012 to 27 December 2012). In order to make the recommendation results more accurate, we filter out noise and invalid check-ins. Our dataset contains 238972 check-ins, 37628 venues, and 300 preferences. Based on a lot of experimental tests and data analysis, the parameters of our system are determined as follows:

- $\alpha = 0.7, \beta = 0.3$, if preference is preferred by a user.
- $\alpha = 0.3, \beta = 0.7$, if geography is preferred by a user.
- γ is fixed to 0.1.

3.2 Baseline Method

Our system could provide more personalized POIs recommendation for each single user. The main reason is that our system integrates geographic and preference information into a unified recommendation process. To evaluate the effectiveness of R2SIGTP, we choose the baseline method as a comparison. The baseline approach also takes into account preference and geographical factors, but it does not integrate these two factors into a unified recommendation process. More specifically, the baseline method first ranks the recommendation POIs according to the preference ranking of the user's in current time, and then sorts POIs that belong to the same preference in ascending order of their distance from the current location.

3.3 Recommendation Results Evaluation

In order to accurately evaluate the final recommendation result, R2SIGTP and the baseline method use the same dataset. We selected 200 users to evaluate the performance of the two methods. We generate a Top-k (k=15) recommendation list for every recommendation of each user. We choose Normalized Discounted Cumulative Gain(NDCG) as an evaluation metric to evaluate the performance of the two methods. NDCG is widely used to measure ranking algorithms. Figure 2 shows the NDCG value of R2SIGTP and the baseline method. Since priority can be selected by users in

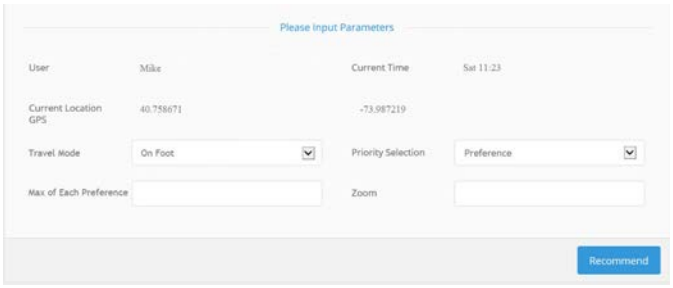


Figure 3: Start page of R2SIGTR

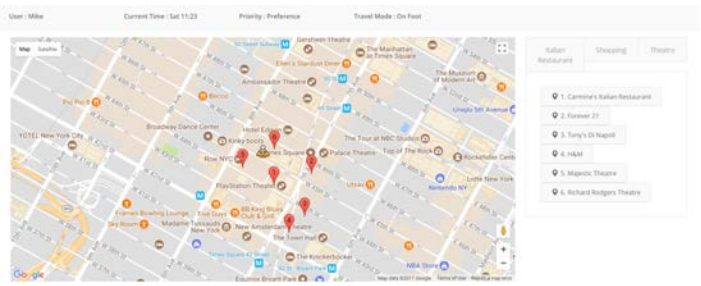


Figure 4: Interface of R2SIGTP

R2SIGTP, we compare the baseline method with R2SIGTP which selected preference priority and geographical priority, respectively. From the results, we can observe that R2SIGTP with preference priority is slightly better than the baseline method, and R2SIGTP with geographical priority is significantly better than the baseline method. This is because R2SIGTP with preference priority is more similar to the baseline method. When R2SIGTP meets the requirements of users to select geographical priority, the baseline method is powerless for users' requirements. Figure 2 demonstrates the final recommendation results of R2SIGTP.

4 DEMONSTRATION

R2SIGTP is composed of a browser frontend and a server backend. The web server is deployed with Apache Tomcat and PHP. Users can use R2SIGTP through a browser on any Internet-enabled terminal.

We show the start page of R2SIGTP in Figure 3. User column needs to fill in a user's account. R2SIGTP can automatically identify the current time and the user's current location and display it in the corresponding position. Priority(preference or geography) and travel mode(on foot or vehicle) can be selected by users. Max of each preference column is used to set the maximum recommended number of POIs for each preference. Zoom column is used to set the scale of the map.

After a user has completed filling and setting, the interface will change to another layout as shown in Figure 4. The user information, the current time, the priority and the travel mode are displayed at the top of the interface. Our system will set different search range for users according to different travel mode. For example, search range of on foot mode is small, and search range of vehicle mode is large. The map which can be zoomed in/out by a user is displayed in the middle of the interface. The map shows the current location of the user and the POIs in the recommendation list. Clicking on each POI on the map will present the corresponding introduction. The right side of the interface shows the top N1 (N1=3 here) preferences and the corresponding recommendation list in current time which shows the top N2 (N2=6 here) POIs.

5 CONCLUSION

In this demo paper, we present a novel real-time recommendation system called R2SIGTP to make next POI recommendation. Compared to the existing systems, our system could provide more personalized POIs recommendation for each single user. The main reason is that our system integrates geographic and preference information into a unified recommendation process. Furthermore, our system also considers the feedback of each single user to adjust the corresponding recommendation. The experimental results show the effectiveness of our system.

ACKNOWLEDGMENTS

This work is supported by the National Nature Science Foundation of China (61170174 and 61702368), major research project of National Nature Science Foundation of China (91646117), Natural Science Foundation of Tianjin (17JCYBJC15200) and Tianjin Science and Technology Correspondent Project (16JCTPJC53600).

REFERENCES

[1] Hakan Bagci and Pinar Karagoz. 2016. Context-Aware Friend Recommendation for Location Based Social Networks using Random Walk. In *Proceedings of the 25th International Conference Companion on World Wide Web (WWW '16)*. April, 2016, Montreal, Canada, 531–536. https://doi.org/10.1145/2872518.2890466

[2] C. Cheng, H. Yang, I. King, and M. R. Lyu. 2012. Fused matrix factorization with geographical and social influence in location-based social networks. In *Proceeding of the 26th AAAI Conference on Artificial Intelligence (AAAI '12)*. July 22-26, 2012, Toronto, Ontario, Canada, 17–23.

[3] Ekta Gujral, Ravdeep Pasricha, and Evangelos E Papalexakis. 2018. SamBaTen: Sampling-based Batch Incremental Tensor Decomposition. In *Proceedings of the 2018 SIAM International Conference on Data Mining (SDM '18)*. November, 2018, Singapore, 387–395. https://doi.org/10.1137/1.9781611975321.44

[4] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP '14)*. October, 2014, Doha, Qatar, 1746–1751. https://doi.org/10.3115/v1/d14-1181

[5] Mao Ye, Peifeng Yin, Wang Chien Lee, and Dik Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information (SIGIR '11)*. July, 2011, Beijing, China, 325–334. https://doi.org/10.1145/2009916.2009962

[6] Jia Dong Zhang and Chi Yin Chow. 2013. iGSLR: personalized geo-social location recommendation:a kernel density estimation approach. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '13)*. November, 2013, Orlando, Florida, USA, 334–343. https://doi.org/10.1145/2525314.2525339

[7] Shenglin Zhao, Tong Zhao, Haiqin Yang, Michael R Lyu, and Irwin King. 2016. STELLAR: Spatial-Temporal Latent Ranking for Successive Point-of-Interest Recommendation. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI '16)*. February 12-âĂŞ17, 2016, Phoenix, Arizona, USA, 315–321. https://doi.org/10.1007/978-981-13-1349-3_5

[8] Yu Zheng, Lizhu Zhang, Zhengxin Ma, Xing Xie, and Wei Ying Ma. 2011. Recommending friends and locations based on individual location history. *ACM Trans. Web.* 5, 1, Article 5 (February 2011), 44 pages. https://doi.org/10.1145/1921591.1921596

SGX-PySpark: Secure Distributed Data Analytics

Do Le Quoc
TU Dresden, Scontain UG

Franz Gregor
TU Dresden, Scontain UG

Jatinder Singh
University of Cambridge

Christof Fetzer
TU Dresden, Scontain UG

ABSTRACT

Data analytics is central to modern online services, particularly those data-driven. Often this entails the processing of large-scale datasets which may contain private, personal and sensitive information relating to individuals and organisations. Particular challenges arise where cloud is used to store and process the sensitive data. In such settings, security and privacy concerns become paramount, as the cloud provider is trusted to guarantee the security of the services they offer, including data confidentiality. Therefore, the issue this work tackles is *"How to securely perform data analytics in a public cloud?"*

To assist this question, we design and implement SGX-PySpark– a secure distributed data analytics system which relies on a trusted execution environment (TEE) such as Intel SGX to provide strong security guarantees. To build SGX-PySpark, we integrate PySpark - a widely used framework for data analytics in industry to support a wide range of queries, with SCONE - a shielded execution framework using Intel SGX.

CCS CONCEPTS

• **Information systems → Data analytics**; • **Security and privacy → Distributed systems security**.

KEYWORDS

Confidential computing; data analytics; security; distributed system

ACM Reference Format:
Do Le Quoc, Franz Gregor, Jatinder Singh, and Christof Fetzer. 2019. SGX-PySpark: Secure Distributed Data Analytics. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314129

1 INTRODUCTION

Cloud-based services are used to collect, process and analyze large amounts of user's personal data, some of it highly sensitive, such at that relating to personal finances, political views, health, and so forth. Indeed, we have seen increasing attention of regulators on issues regarding the way in which personal data is handled and processed - the EU's General Data Protection Regulation a case in point. Thus, confidentiality and integrity of the data processing in clouds are becoming more important, not least because of increased demands for accountability regarding service providers, and the potential serious legal consequences (and fines) for the data mishandling, mismanagement and leakage, and more generally, for failing to implement the appropriate security measures [9]. Service providers must ensure that data is always protected, i.e., at rest, during transmission, and computation. Many organisations make use of public cloud services for the processing to reduce time and computation cost. This setting is vulnerable to many security threats, e.g., data breaches [10]. Concerns are compounded when we consider attacks from inside the cloud provider, where attackers might have root privileges and/or physical access to machines deployed at the service providers' premises. Therefore, to protect the sensitive data and the analytics computation over the data, service providers cannot rely solely on the operating system access control nor their security policy-based mechanisms.

An promising approach to helping resolve these security challenges is to make use of Trusted Execution Environments (TEEs), such as Intel Software Guard Extensions (SGX). Intel SGX protects the confidentiality and integrity of application code and data even against privileged attackers with root access and physical access. In general, Intel SGX provides an isolated secure memory area called *enclaves*, where the code and data can be executed safely. These security guarantees are solely provided by the CPU, thus even if system software is compromised, the attacker can never access the enclave's content. This approach supports data analytics at processor speeds while ensuring the security guarantee for both computation and sensitive data.

While promising at first glance, to build a practical secure data analytics system using TEEs, e.g., Intel SGX, we need to deal with several challenges. ($\mathcal{A}$) In the current version, Intel SGX supports only a limited memory space (~ 94MB) for applications running inside enclaves. Meanwhile, most big data analytics systems (e.g., Hadoop and Apache Spark [1]) are extremely memory-intensive, since these systems are almost always based on Java Virtual Machine (JVM). ($\mathcal{B}$) Intel SGX still suffers from side-channel attacks [12]. These side-channel attacks happen both at memory level [12, 16] and network level [16]. ($\mathcal{C}$) Deployment and bootstrapping of a data analytics framework to run inside enclaves is not trivial, in fact, challenging. Securely transferring configuration secrets such as certificates, encryption keys and passwords to start the framework inside enclaves is complicated because these secrets need to be protected on the network as well as securely moved into the enclaves. ($\mathcal{D}$) Typically, Intel SGX requires users to heavily modify the source code of their application to run inside enclaves. Thus, transparently supporting a unmodified distributed data analytics framework to run inside enclaves is not a trivial task.

In the context of building secure data analytics systems using Intel SGX, VC3 [13] is one of the first works that applied SGX technology for Hadoop MapReduce framework. VC3 handles challenges

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314129

($\mathcal{A}$) and ($\mathcal{D}$) by using C/C++ to implement the framework and support unmodified Hadoop. However, the challenge ($\mathcal{B}$) is outside the scope of this work. Recently, Opaque [16] overcomes this issue by introducing a oblivious mechanism to hide access patterns at network level. Opaque provides secure data analytics on Apache Spark framework using Intel SGX. It deals with ($\mathcal{A}$) by reimplementing SQL operators for Spark SQL Catalyst engine [6] using C++. These operators run inside enclaves and communicate with Scala code of Spark using a JNI interface. Opaque supports common operators including map, reduce, filter, sort, aggregation, and join, but not all operators of Apache Spark. This means that it does not handle the challenge ($\mathcal{D}$) completely. In addition, it does not support *remote attestation* to verify the integrity of the code running inside SGX enclaves. It also does not handle the challenge (C) since it does not provide a secrets transferring mechanism for execution inside enclaves. Finally, Opaque requires to run Spark master/driver at client side or in a trusted domain. This might affect significantly the performance of the system.

In this work, we overcome these limitations by building a secure data analytics system called SGX-PySpark. We handle the challenge ($\mathcal{A}$) by using PySpark [3], a system built on top of Apache Spark to support data analytics using Python processes. Instead of running a whole JVM inside an enclave to secure Apache Spark or reimplementing operators in C/C++ as Opaque, we run only Python processes inside enclaves since these processes perform analytics over encrypted data. Thus, our system supports out-of-the-box operators of PySpark (the challenge ($\mathcal{D}$)), i.e., users do not need to modify their source code. To run Python processes inside Intel SGX enclaves, our system makes use of SCONE [4, 7] a shielded execution framework which enables unmodified applications to run inside Intel SGX enclaves.

In addition, SGX-PySpark, with the help of SCONE, supports a remote attestation mechanism to ensure the code and data running inside enclaves are correct and not modified by an attacker. SGX-PySpark also copes with challenge (C) by providing a mechanism to securely transfer secrets (keys and certificates) to Python processes running inside enclaves. To handle challenge ($\mathcal{B}$), SGX-PySpark protects its execution against side channel attacks at memory level using a mechanism integrated with SCONE, called Varys [12]. Finally, the design of SGX-PySpark allows users to run the Spark driver/master in the same infrastructure as workers.

2 BACKGROUND

2.1 Intel SGX

Intel SGX is an ISA extension which is a set of special CPU instructions for Trusted Execution Environments (TEE). These instructions enable applications to create *enclaves* – protected areas in the applications address space to provide strong confidentiality and integrity guarantees against adversaries with privileged root access. Intel SGX enables trusted computing by isolating the environment of each enclave from untrusted applications outside the enclave. In addition, by offering the remote attestation mechanism, Intel SGX allows a remote party to attest the application executing inside an enclave [8].

The enclave memory is acquired from Enclave Page Cache (EPC)—a dedicated memory region protected by an on-chip Memory Encryption Engine (MEE). The MEE transparently encrypts cache lines on cache-line evictions and decrypts and verifies cache lines with on cache-line loads. The EPC cannot be directly accessed by non-enclave applications including operating systems. To support multiple enclaves on a system, the EPC is partitioned into 4KB pages which can be assigned to various enclaves. Currently, the size of EPC is limited to 128MB in which only ~ 94MB can be used for user applications and the rest is used to store SGX metadata. Fortunately, SGX supports a secure paging mechanism to an unprotected memory region even though the paging mechanism may introduce significant overheads.

The EPC is managed as the rest of the physical memory by an operating system (or a hypervisor in virtualized environments). The operating system makes use of SGX instructions to allocate and free EPC pages for enclaves. In addition, the operating system is supposed to expose the enclave services (creation and management) to applications. Since the operating system cannot be trusted, the SGX hardware verifies the correctness of EPC pages allocations and denies any operations that would violate the security guarantees. For example, the SGX hardware will not allow the operating system to allocate the same EPC page for different enclaves.

2.2 SCONE

Our system builds on SCONE [7] – a shielded execution framework to enable unmodified applications to run inside SGX enclaves. In the SCONE platform, the source code of an application is recompiled against a modified standard C library (SCONE libc) to facilitate the execution of system calls. The address space of the application stays within an enclave, and the application only can access the untrusted memory via the system call interface.

SCONE uses the compiler-based approach to prepare and build native applications for executing inside SGX enclaves. SCONE applies its mechanism into GNU Compiler Collection (GCC) tool-chain to change the compiling process such that it can build position independent, statically linked code, and eventually linked with the starter program. Therefore, SCONE natively supports C/C++ applications. For Python applications e.g., PySpark executors, we need to compile the CPython/PyPy interpreter with SCONE to run these Python processes inside SGX enclaves. Similarly, to run a Java application inside an enclave, we compile JVM with SCONE.

2.3 PySpark

PySpark is built on top of Apache Spark [1] to provide the Python API for users. Thus, before explaining PySpark, it is useful to understand what is Apache Spark. Apache Spark [1] is an open-source large-scale data analytics framework. Today, it has become the most popular and widely used big data framework in both academia and industry. Comparing to earlier frameworks such as Hadoop MapReduce, Spark is much faster in processing large-scale datasets since it enables the in-memory computing concept where intermediate data is cached in memory to reduce latency [11].

For the in-memory computation, Spark introduces the core abstraction – Resilient Distributed Datasets (RDDs) [15] for distributed data-parallel computing. An RDD is an immutable and fault-tolerant

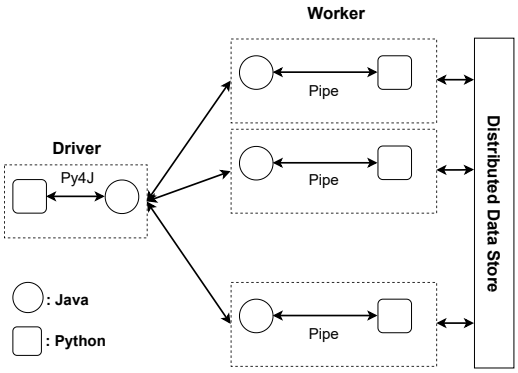


Figure 1: PySpark's architecture.

collection of elements (objects) that is distributed or partitioned across a set of nodes in a cluster [1].

RDDs support two types of operations: `transformations` and `actions`. Transformations return a new RDD, such as *map()* and *filter()*, and actions are operations performed on RDDs that return the output to the driver program or store it in a persistent storage system (e.g *reduce()*, *count()*, *collect()*, *saveAsTextFile()*, etc). Transformations are performed lazily, i.e., they are computed whenever an action operation is invoked [11].

In a deployment, Spark contains a main program called *driver* which coordinates the processes (tasks) executed in other nodes (i.e., workers) in a cluster. When a Spark application is submitted to the driver, it will split the job into tasks and perform scheduling to dispatch tasks to workers. The workers then spawn processes (called *executors*) to handle received tasks.

Since Python supports powerful libraries such as scipy, numpy, scikit-learn, and pandas with a simple and concise syntax, it is a favorite language of data scientists. For that reason, PySpark has been proposed as a Spark programming model to Python. Figure 1 shows the high-level architecture of PySpark.

PySpark extends the Spark runtime to enable executing Python programs on top of Spark. Typically, a PySpark job (a Python process) is submitted, and a JVM is started to communicate with the Python process using Py4J [2]. The Python process creates a *SparkContext* object in the JVM and the SparkContext orchestrates the computation as the regular Spark framework (to reuse almost the same Spark infrastructure). However, the difference is that in PySpark, the executors are Python processes. Each Python process (per CPU core) takes care of the execution of the assigned tasks. Each worker (JVM-based program) submits received tasks to the pool of Python processes and communicates with them using Pipes. The Python processes perform the computation and store back resulting data to an RDD (as pickle objects) in the JVM.

3 SGX-PYSPARK

Our idea to design SGX-PySpark— a secure distributed data analytics system using Intel SGX, is quite simple. We execute only sensitive parts, i.e., the computation parts that process the input sensitive data, inside enclaves. The computation parts outside of enclaves can only access encrypted data. In general, we first encrypt the input data and upload the encrypted data into a distributed storage in an untrusted infrastructure (e.g., a public cloud). Thereafter, SGX-PySpark decrypts and processes the encrypted data inside enclaves in a distributed manner.

Figure 2 illustrates SGX-PySpark's architecture. SGX-PySpark consists of two main components: *(i)* Configuration and attestation service (CAS) component and *(ii)* PySpark with integration with the SCONE library to run inside enclaves. SGX-PySpark maintains the native layout of PySpark (see §2.3).

To secure data analytics computations on top of PySpark, SGX-PySpark runs the driver and the Python processes inside Intel SGX enclaves using the SCONE library. We need to ensure the confidentiality and integrity of the driver, since it is responsible for splitting and scheduling tasks in the system. We protect the Python processes since they are the sensitive computation parts since they directly decrypt and process the input data. Note that in SGX-PySpark, we encrypt not only the input sensitive data but also the computation over it (Python code of analytics jobs).

3.1 Trusted Enclave-based Driver

In PySpark, when we submit a job, the Python driver program makes use of Py4J to start a JVM (Spark driver) and create a JavaSparkContext. This JavaSparkContext orchestrates the job as a regular Spark framework. The JVM-based process takes the responsibility to convert a submitted job into tasks. In detail, it first converts the logical DAG of operations in the submitted job into a physical execution plan, i.e., it divides the DAG into a number of stages. Thereafter, it divides these stages into smaller tasks. Next, the Spark scheduler distributes these tasks to executors (Python processes) deployed on worker nodes for execution (see §2.3).

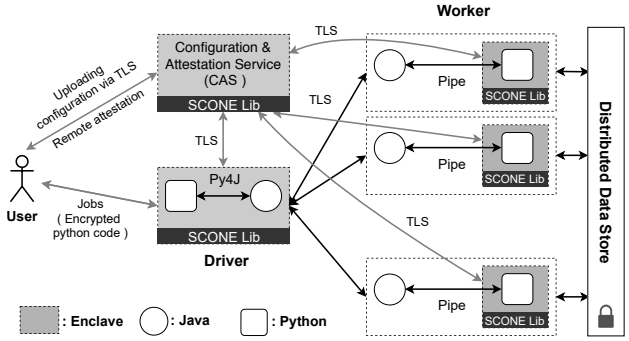


Figure 2: SGX-PySpark's architecture.

To provide secure data analytics, we need to protect the driver as it is the command center of the whole system. To achieve this, in SGX-PySpark, we recompile JVM running in the driver node with the SCONE library to execute the whole driver inside an SGX enclave.

3.2 Trusted Enclave-based Executors

To execute the analytics job (Python code), PySpark launches Python processes (worker processes) and communicates with them using pipes to transfer the user-specified Python code and processed data. Since these Python processes interact directly with the input

data, we protect them against malicious activities, i.e., ensure the integrity and confidentiality by running them inside enclaves with the help of the SCONE platform. Note that, in SGX-PySpark both input data and computation (Python code) are encrypted before upload to the system. They are decrypted inside enclaves using keys transparently obtained from CAS (see §3.3).

3.3 Configuration and Remote Attestation Service

In SGX-PySpark, we need to make sure that the shared secrets such as certificates, passwords to start PySpark, or keys for encrypting/decrypting the input data and computations can never be revealed to untrusted components. Furthermore, we need to securely transfer these secrets to the driver and executors (Python processes) running inside enclaves (see challenge (C) in §1). To achieve these goals, we extend SCONE with a configuration and attestation service (CAS) that transfers security secrets only to the components that have authenticated themselves successfully against it. CAS enhances the Intel attestation service [8] to bootstrap and establish trust across the machines running SGX-PySpark and maintain a secure configuration for the system. In detail, CAS remotely attests the driver and worker processes of PySpark running inside enclaves, before providing encryption/decryption keys and other configuration parameters. CAS is itself launched inside an SGX enclave. A user of SGX-PySpark, first encrypts the input data and his data analytics job and then uploads the secrets (cryptographic keys) and system configurations to CAS. A basic implementation of CAS is presented in [14]. In addition, to guarantee the correctness of computation in SGX-PySpark, we design and implement an auditing service in CAS to keep result logs during runtime. This auditing service protects our data analytics system against rollback attacks.

3.4 Network and File System Protection

File system shield. To protect integrity and confidentiality of the Python codes and the sensitive input data stored on disk, we design in SGX-PySpark a file system shield using SCONE library. In the case SGX-PySpark would write computation results to a file, the shield encrypts the contents before writing. The shield ensures the integrity of these files by keeping their metadata inside CAS (see §3.3). The keys to encrypt the computation content are different from the secrets used by the SGX implementation. They are instead a part of configuration that is uploaded to CAS by users at the startup time of the data analytics system.

Network shield. In SGX-PySpark, to protect the confidentiality of secrets transfered from CAS to computation components running inside enclaves, we need to protect the network communication between CAS and the components to make sure that an attacker cannot observe the network traffic to steal the secrets. To achieve this security requirement, we enhance SGX-PySpark by designing the network shield that wraps the communication between our system components in TLS connections and ensures that all data passes to the connection and is TLS-encrypted. The certificates for TLS connections are saved in a configuration file protected by our file system shield.

4 DEMONSTRATIONS

In this section, we demonstrate how a user can securely perform data analytics using SGX-PySpark[1]. For demonstration purposes, we consider a simple and classical workload of "wordcount". Assume that the user wants to perform data analytics (e.g. wordcount) over a sensitive input data.

4.1 Protecting Data

A straightforward way to securely process the input data is that the user encrypts the data before uploading it to an untrusted domain such as a public cloud. In this demo, we show that this mechanism is not enough to protect secrets inside the input data, since the user needs to decrypt the encrypted data and then process it in memory; an attacker, especially attackers with privileges, can just dump the memory content to steal the secrets. In SGX-PySpark, the input data is encrypted using the file system shield, and then decrypted and processed inside SGX enclaves which cannot be accessed even by strong attackers with root access.

4.2 Protecting Computation

When the user submit a job (Python code) to the system, an attacker might learn what kind of computation the user wants to perform. We show in this demo that by using SGX-PySpark, the user can further encrypt his computation using the same file system shield before uploading to the untrusted domain (e.g., a cloud) for execution. The secret key for decryption is transferred into executors running inside SGX enclaves using the same mechanism as in §4.1.

4.3 Benchmarks

Beside the wordcount workload, we also make use of a standard data analytics benchmark (i.e.,TPC-H [5]) to demonstrate that SGX-PySpark supports a wide range of queries as native PySpark. Figure 3 presents the latency comparison between SGX-PySpark with native PySpark in processing TPC-H queries. The performance overhead incurred when running Python processes inside enclaves is not significant compared to the native PySpark. The reason for this is that the main overhead of SGX-PySpark is introduced by communication between Python processes and JVMs in workers.

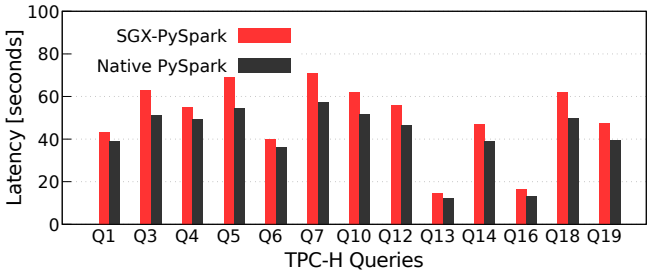


Figure 3: TPC-H benchmark.

Acknowledgments. This work is supported by the European Unions Horizon 2020 research and innovation programme under

[1]The demo repository is available here: https://github.com/doflink/sgx-pyspark-demo

grant agreements No. 777154 (ATOMSPHERE) and No. 780681 (LEGaTO).

REFERENCES

[1] Apache Spark. https://spark.apache.org. Accessed: Jan, 2019.

[2] Py4J. http://py4j.sourceforge.net. Accessed: Jan, 2019.

[3] PySpark. http://spark.apache.org/docs/2.2.0/api/python/pyspark.html. Accessed: Jan, 2019.

[4] Scontain Technology. https://sconedocs.github.io/. Accessed: Jan, 2019.

[5] TPC-H Benchmark. http://www.tpc.org/tpch/. Accessed: Jan, 2019.

[6] Michael Armbrust, Reynold S. Xin, Cheng Lian, Yin Huai, Davies Liu, Joseph K. Bradley, Xiangrui Meng, Tomer Kaftan, Michael J. Franklin, Ali Ghodsi, and Matei Zaharia. 2015. Spark SQL: Relational Data Processing in Spark. In *Proceedings of the International Conference on Management of Data (SIGMOD)*.

[7] Sergei Arnautov, Bohdan Trach, Franz Gregor, Thomas Knauth, Andre Martin, Christian Priebe, Joshua Lind, Divya Muthukumaran, Dan O'Keeffe, Mark L. Stillwell, David Goltzsche, Dave Eyers, Rüdiger Kapitza, Peter Pietzuch, and Christof Fetzer. 2016. SCONE: Secure Linux Containers with Intel SGX. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.

[8] Victor Costan and Srinivas Devadas. 2016. Intel SGX Explained. Cryptology ePrint Archive, Report 2016/086.

[9] General Data Protection Regulation. 2016. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46. *Official Journal of the European Union (OJ)* (2016).

[10] Tim Greene. Biggest data breaches of 2015. https://www.networkworld.com/article/3011103/security/biggest-data-breaches-of-2015.html. Accessed: Jan, 2019.

[11] Holden Karau, Andy Konwinski, Patrick Wendell, and Matei Zaharia. 2015. *Learning Spark: Lightning-Fast Big Data Analysis.* " O'Reilly Media, Inc.".

[12] Oleksii Oleksenko, Bohdan Trach, Robert Krahn, Mark Silberstein, and Christof Fetzer. 2018. Varys: Protecting SGX Enclaves from Practical Side-Channel Attacks. In *2018 USENIX Annual Technical Conference (USENIX ATC)*.

[13] Felix Schuster, Manuel Costa, Cédric Fournet, Christos Gkantsidis, Marcus Peinado, Gloria Mainar-Ruiz, and Mark Russinovich. 2015. VC3: Trustworthy Data Analytics in the Cloud Using SGX. In *Proceedings of the Symposium on Security and Privacy (SP)*.

[14] Bohdan Trach, Alfred Krohmer, Franz Gregor, Sergei Arnautov, Pramod Bhatotia, and Christof Fetzer. 2018. ShieldBox: Secure Middleboxes using Shielded Execution. In *Proceedings of the Symposium on SDN Research (SOSR)*.

[15] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2012. Resilient Distributed Datasets: A Fault Tolerant Abstraction for In-Memory Cluster Computing. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation (NSDI)*.

[16] Wenting Zheng, Ankur Dave, Jethro G. Beekman, Raluca Ada Popa, Joseph E. Gonzalez, and Ion Stoica. 2017. Opaque: An Oblivious and Encrypted Distributed Analytics Platform. In *Proceedings of the 14th USENIX Conference on Networked Systems Design and Implementation (NSDI)*.

TweetSenti: Target-dependent Tweet Sentiment Analysis

Quanzhi Li
Machine Intelligence Technology
Alibaba Group, Bellevue, WA,
USA, quanzhi.li@alibaba-inc.com

Qiong Zhang
Machine Intelligence Technology
Alibaba Group, Bellevue, WA,
USA, qz.zhang@alibaba-inc.com

Luo Si
Machine Intelligence Technology
Alibaba Group, Bellevue, WA,
USA, luo.si@alibaba-inc.com

ABSTRACT

TweetSenti is a system for analyzing the sentiment of an entity in tweets. A sentence or tweet may contain multiple entities, and they do not always have the same sentiment polarity. Therefore, it is necessary to detect the sentiment for a specific target entity. This type of target-dependent (entity level) sentiment analysis has become attractive and has been used in many applications, but it is still a challenging task. TweetSenti employs a new approach for detecting the entity level sentiment. Our model splits a sentence into a left context and a right context according to the target entity, and it also exploits two different types of word embeddings to represent a word, the general word embedding and the sentiment specific word embedding. A hybrid neural network is used to capture both the sequence and structure information of the two sides of the target entity. The sequence information is learned by attention-based bi-directional LSTM models. The structure information is captured by multi-context CNN models. Based on this algorithm, we built a web-based application that users can interact with and analyze an entity's sentiment in Twitter at real-time.

CCS CONCEPTS

• Computing methodologies~Natural language processing • Computing methodologies~Neural networks

KEYWORDS

Target-dependent sentiment, entity level sentiment classification, Twitter, social media

ACM Reference format:

Quanzhi Li, Qiong Zhang and Luo Si. 2019. TweetSenti: Target-dependent Tweet Sentiment Analysis. In *Proceedings of WWW '19: The Web Conference (WWW '19), May 13, 2019, San Francisco, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314141

1 Introduction

With the permeation of social media in our lives, people are expressing their opinions about basically everything on social media platforms, such as Twitter, Facebook and Reddit. Sentiment analysis for entities, such as people, organizations, and products, has applications in various areas. Companies want to find out what consumers think of their products, brands, services or competitors. In public actions, it can be used to analyze online reactions to social and cultural phenomena. In politics, we can use it to keep track of society's opinions on politicians, the government, policy changes, or to predict election results.

There have been many studies on message or sentence level sentiment classification. Traditional sentiment classification approaches use sentiment lexicons to generate various features. Pang et al. [11] treat sentiment classification as a special case of text categorization. Many studies follow Pang's approach by designing features and applying different learning algorithms on them [4, 6, 10]. Feature engineering plays an important role in sentence sentiment classification [9]. Deep learning has also been used in sentence level sentiment analysis, mainly by exploiting word embedding [1, 8].

An entity in a sentence (post or tweet) does not necessarily have the same polarity type as the sentence, and different entities in the same sentence may have different polarities. For example, in the sentence of "*Farmers like Trump more than Hillary*", the two entities, *Trump* and *Hillary*, will have different sentiment polarities. There are some studies on entity level sentiment prediction [5, 2, 3, 14, 15, 18, 17] in the last several years. Jiang et al. [5] use both entity dependent and independent features to assign sentiment polarity to entities. By using POS features and the CRF algorithm, Mitchell et al. [7] identify polarities for people and organizations in tweets. Dong et al. [2] apply adaptive recursive neural network on the entity sentiment classification. The above two approaches use syntax parsers to parse tweets to generate features. Vo and Zhang [15] use target dependent and independent features, and also features based on lexicons.

Compared to the basic LSTM algorithm, TD-LSTM from [13] improves the sentiment classifier performance by treating an entity as a target. TC-LSTM from [14] extended TD-LSTM by adding target representations, which are word vectors, into the input of the LSTM cell unit. Several studies also explored applying attention mechanism with LSTM, and it has shown performance improvement [3, 16, 17, 18]. Wang et al [16] propose an attention-based LSTM model, ATAE-LSTM, and their experiment shows that attention mechanism is effective. The approach from [3] tries to enhance the attention-based LSTM by exploiting syntactic information to construct a syntax-based attention model.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13, 2019, San Francisco, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314141

In our TweetSenti system, we employed a new approach to address this problem. The main difference between our approach and previous studies are as follow: First, previous studies exploit only LSTM (with or without word attention), which captures mainly the sequence information of a sentence. Our approach uses a hybrid approach, which captures both the sequence and structure information of a sentence by integrating the attention-based bi-directional LSTM with the multi-context CNN model. Second, The context of an entity will affect its polarity, and usually an entity has a left context and also a right one. In our approach, we split a sentence into a left context and a right context according to the target entity, and we capture both contexts and also the whole sentence which captures the interaction between the left and the right context, through the multi-context CNN network structure. Third, previous studies use only the general word embedding (WE), such as embedding from word2vec [8] or C&W [1], to represent a word. Our approach uses not only WE but also the sentiment specific word embedding (SSWE) to represent a word.

In the demonstration, we will provide a highly interactive session, in which the audience can interact with the web application of our system. Several scenarios will be demonstrated.

2 TweetSenti System

In the following two subsections, we first describe the high-level workflow of TweetSenti, and then introduce how we build the entity sentiment classification module that supports the system.

2.1 TweetSenti Workflow

Figure 1 shows the high-level workflow of this application. We briefly introduce some of its components below.

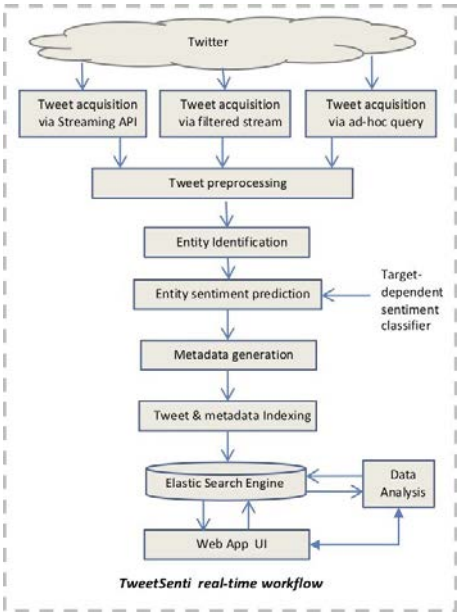


Figure 1: The high-level system real-time workflow

Tweet Acquisition. This system has three channels to acquire tweets from Twitter, all at real time. The main approach is through the Twitter streaming API, and this is done by our backend streaming tool at real-time. The second approach is via the filtered stream, which acquires tweets by monitoring a list of keywords, such as names of companies and political figures. These keywords can be modified at any time, and the system will make changes accordingly. Finally, users can submit queries via the system interface, and TweetSenti will retrieve related tweets on the fly from Twitter.

Tweet Preprocessing. In order for the sentiment classifier to predict the sentiment polarity for an entity, some preprocessing steps are necessary, such as normalizing numbers, and removing URLs and retweet mentions.

Entity Identification. To recognize entities from a tweet, we use TweetNLP package [20]. Some necessary preprocessing steps for entity identification is done by this package itself.

Entity Sentiment Prediction. For each entity extracted by the entity identification module, we use our entity sentiment classifier to predict its sentiment polarity. The polarity value and other metadata will be analyzed and presented to users by TweetSenti.

Metadata Generation. Some metadata are extracted or created in this step, and they will be displayed on UI or used for other analysis. For example, the tweet creation time is extracted and used for displaying the creation time, and generating the sentiment trending chart as well.

Data Indexing. A retrieved tweet and the generated metadata from it by the above components are ingested into an Elastic Search engine cluster (ES) [21] and indexed. We chose ES because it provides all the functions our application needs, and it has good scalability and reliability.

Data Analysis. This module is to analyze the data indexed and stored in the ES engine, and it also generates related information for TweetSenti to present them on the UI. For example, given a set of tweets talking about Trump, this model calculates how the sentiment for *Trump* have changed over a given period of time, and it also identifies a list of terms associated with the positive tweets and negative tweets. These information will be presented to users.

2.2 Entity Sentiment Classification Module

Figure 2 shows the construction process of the entity sentiment classification model. The following three subsections describe the three main components in the classification model construction process: the word embeddings, the framework of the classifier, and how the classifier is the trained.

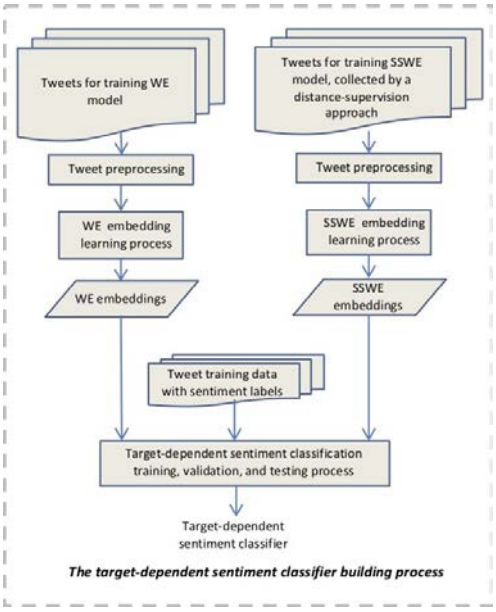


Figure 2: The classification module construction process

2.2.1 Word Embedding.:As mentioned before, we concatenate the WE embedding and the SSWE embedding together to represent a word.

WE model. The general embedding of a word captures both the syntactic structure and semantics of the word. Traditional bag-of-words and bag-of-n-grams hardly capture the semantics of words. Word embedding has been used in many NLP tasks. word2vec [8] is used in our algorithm to generate the WE embedding. About 200 million tweets were collected from Twitter to build the embedding model. A total of 3 billion words were processed, and the embeddings were generated for 3.5 million unique terms. The embedding size, window size and word count threshold were empirically set as 300, 8 and 5, respectively. Appropriate preprocessing steps are conducted on each tweet before it is fed into the learning process, such as removing URLs, normalizing numbers, etc.

SSWE model. SSWE extends the C&W model [1] by incorporating the sentiment information into the neural network to learn the embedding; it captures the sentiment information of sentences as well as the syntactic contexts of words [12]. Given an original (or corrupted) n-gram and the sentiment polarity of a tweet as input, it predicts a two-dimensional vector *(f0, f1)*, for each input n-gram, where *(f0, f1)* are the language model score and sentiment score of the input n-gram, respectively. The training objectives are twofold: the original n-gram should get a higher language model score than the corrupted n-gram, and the polarity score of the original n-gram should be more aligned to the polarity label of the tweet than the corrupted one. The SSWE model was built from tweets using a distance-supervision approach, which uses emotion symbols to determine if a tweet is positive or negative. If the tweet contains one of the following positive symbols, it is treated as a positive tweet: *:) :-) =) :D ;)*. And it is negative tweet if it contains one of these symbols: *:-(:(=(;(*. Eventually, a total of ten million tweets were collected from Twitter, where 5 million of them contain positive emotions and the other 5 million contain negative ones. Mentions and URLs were removed from tweets. The embedding size, window size and word count threshold were empirically set as 50, 3 and 2, respectively [12].

2.2.2 The Classification Model To exploit both the sequence information and structure information of a sentence, we use a hybrid neural network, which combines the attention-based bi-directional LSTM and multi-context CNN models. The high-level network structure is shown in Figure 3. In Figure 3, R_F and R_B are the two attention-based LSTM vectors, learned from the forward and backward information, respectively. Given a target entity in a sentence, we split the sentence into two contexts: its left context and its right context. The target term is also included in each context . This is based on the assumption that the sentiment toward the target is decided by both contexts, and treating them separately will give us more information undiscovered before. C_{left} is the vector learned from the left context by a CNN model, and C_{right} is the vector learned from the right context also through a CNN model. C_{all} is generated from the whole sentence, which reflects the interaction between the left and right contexts of the entity. We concatenate these five vectors together, forming a vector V: $V = [R_F, R_B, C_{left}, C_{right}, C_{all}]$. Then, this vector is fed to the fully connected hidden layers, and a softmax layer is used to predict the sentiment polarity.

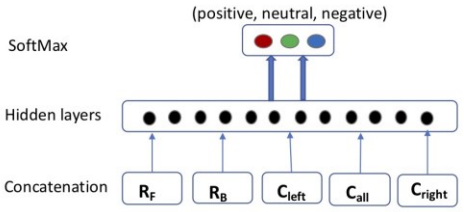


Figure 3: The high-level model structure

2.2.3 Model Training. Because our system works on Twitter, we use tweets to train and build our model. We combine two data sets together. The first data set is from Dong et al. [2014]. This data set was manually annotated with three sentiment polarity labels (positive, neutral and negative) toward target entities. It has 6248 training tweets and 692 testing tweets, with a balanced number of positive, negative, and neutral tweets (25%, 25%, and 50%, respectively). Dong et al. [2014] report that the data set was annotated with 82.5% inter-subject agreement among annotators. This data set has been used in many previous studies. The second data set is from SemEval 2017. It has a total of 50,333 tweets, with 19,902 positive tweets, 22,591 neutral, and 7,840 negative ones [19]. Combining two data sets gave us more training data.

Like previous studies, we use three-fold validation to tune our model and hyper-parameters. Stochastic gradient descent, AdaDelta update, shuffled mini-batch, back-propagation and dropout are used. The WE and SSWE embedding were fine-tuned during the training process of our hybrid neural network. Based on the validation result, we set the feature map (filter) width of

the three CNN layers as 2 and 3, the number of feature map as 300, and batch size as 30.

To see how our model works compared to other methods, we conducted an experiment and compared it to two state-of-the-art entity level sentiment prediction algorithms, AB-LSTM2 from Yang et al. [18], which is an attention-based LSTM approach, and LSTM+SynATT+TarRep from He et al. [3], which is also a variant of the attention-based LSTM model. The result shows that our model slightly outperformed these two methods, with about 3% performance increase, using F1 measure that has been used in previous work [2, 5, 14, 17].

3 System Demonstration

We will provide a highly interactive demonstration, in which the audience can interact with the web application of our system. The following three scenarios will be demonstrated:

#1. A user can input a tweet, or just a sentence, and the system will extract all the entities from the text, analyze their sentiment, and display their sentiment scores. It will also show the sentiment difference between a specific entity and the tweet, i.e. entity level vs. tweet level sentiment.

#2. Our system is connected to Twitter's real-time streaming data. We will demonstrate the real-time sentiment change for some popular people and entities in some hot events that will be happening during the conference period, to make the demonstration more interesting.

#3. Users can also provide an entity (e.g. person, organization), and the system will retrieve the related tweets from the data sources described above. These tweets will be analyzed and the sentiment analysis is conducted for the provided entity at real-time. It will also show the sentiment trending change.

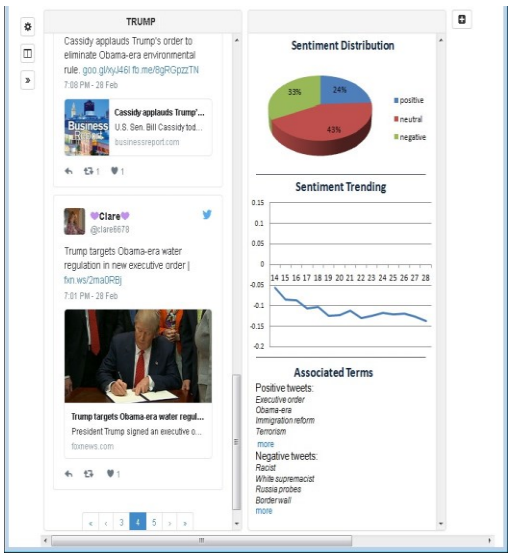


Figure 4: A screenshot showing the sentiment analysis results for "Trump". The panel on the left displays the related tweets, and the one on the right presents the sentiment analysis.

An example of scenario #3: Figure 4 shows how it works for scenario #3 mentioned above. In this example, "*Trump*" is the entity name input by a user. Our system gets the tweets related to Trump from the data sources described before, analyzes their sentiments toward *Trump*, and presents both the tweets and the sentiment analysis result in this window. This screenshot has two panels. The one on the left displays the tweets containing term Trump, and the one on the right presents three types of sentiment analysis results. The top one has a pie chart showing the sentiment distribution of all the tweets collected for *Trump* during the specified time period. Sentiment trending is displayed in the middle of this panel. It shows how the sentiment for *Trump* changed over this period of time. Our sentiment prediction algorithm can output a probability value for each polarity, which is a real number. The value for each day is the average sentiment value of all the related tweets from that day for *Trump*. At the bottom of the panel, it shows a list of terms associated with positive tweets and negative tweets. They are ordered by the strength of their association with the polarity type. By clicking the + icon at the top right corner of this window, one can add more panels to show sentiment analysis for multiple entities in the same window. If we change the time range to minutes or hours, and the "Auto Update" is set as On, we can see the sentiment information displayed in the right panel changing dynamically, as new tweets are being processed at real-time. To get the demo scenario #1 mode, users can click the setting icon at the top left corner and make corresponding changes.

REFERENCES

[1] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural language processing (almost) from scratch. J. Mach. Learn. Res., 12:2493–2537.

[2] L. Dong, F. Wei, C. Tan, D. Tang, M. Zhou, K. Xu, 2014. adaptive recursive neural network for target-dependent twitter sentiment classification. ACL.

[3] R. He, W. S. Leey, H. T. Ngy, and D. Dahlmeier, 2018. Effective Attention Modeling for Aspect-Level Sentiment Classification, COLING

[4] X. Hu, J. Tang, H. Gao, and H. Liu. 2013. Unsupervised sentiment analysis with emotional signals. In WWW, pages 607–618.

[5] L. Jiang, M. Yu, M. Zhou, X. Liu, T. Zhao, 2011. target-dependent twitter sentiment classification, ACL

[6] B. Liu. 2012. Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies, 5(1):1–167

[7] M. Mitchell, J. Aguilar, T. Wilson, and B. V. Durme. 2013. Open domain targeted sentiment. EMNLP.

[8] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. NIPS.

[9] S. M. Mohammad, S. Kiritchenko, and X. Zhu. 2013. Nrc-canada: Building the state-of-the- art in sentiment analysis of tweets. SemEval 2013.

[10] Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. LREC.

[11] B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. EMNLP, pages 79–86

[12] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin. 2014. Learning sentiment specific word embedding for twitter sentiment classification. ACL, pages 1555–1565.

[13] D. Tang, B. Qin, X. Feng, and T. Liu. 2015. Target-Dependent Sentiment Classification with Long Short Term Memory. ArXiv preprint arXiv: 1512.01100 .

[14] D. Tang, B. Qin, X. Feng, T. Liu, 2016. Effective LSTMs for Target-Dependent Sentiment Classification, COLING

[15] D. Vo and Y. Zhang, 2015. Target-dependent twitter sentiment classification with rich automatic features, IJCAI

[16] S. Wang and C. D Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. ACL, pages 90–94

[17] Y. Wang and M. Huang and L. Zhao and X. Zhu, 2016. Attention-based LSTM for Aspect-level Sentiment Classification, EMNLP

[18] M. Yang, W. Tu, J. Wang, F. Xu, X. Chen, 2017. Attention-Based LSTM for Target-Dependent Sentiment Classification, AAAI
[19] Sara Rosenthal, Noura Farra, Preslav Nakov, SemEval-2017 Task 4: Sentiment Analysis in Twitter. SemEval 2017.
[20] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, et al.. Improved part-of-speech tagging for online conversational text with word clusters. NAACL2013.
[21] ElasticSearch, https://www.elastic.co/, 2016

Querying Data Lakes using Spark and Presto

Mohamed Nadjib Mami*
Fraunhofer IAIS & University of Bonn
mami@cs.uni-bonn.de

Damien Graux
Fraunhofer IAIS
damien.graux@iais.fraunhofer.de

Simon Scerri
Fraunhofer IAIS & University of Bonn
scerri@cs.uni-bonn.de

Hajira Jabeen
University of Bonn
jabeen@cs.uni-bonn.de

Sören Auer
TIB and L3S Research Center
auer@l3s.de

ABSTRACT

Squerall is a tool that allows the querying of heterogeneous, large-scale data sources by leveraging state-of-the-art Big Data processing engines: Spark and Presto. Queries are posed on-demand against a Data Lake, i.e., directly on the original data sources without requiring prior data transformation. We showcase Squerall's ability to query five different data sources, including inter alia the popular Cassandra and MongoDB. In particular, we demonstrate how it can jointly query heterogeneous data sources, and how interested developers can easily extend it to support additional data sources. Graphical user interfaces (GUIs) are offered to support users in (1) building intra-source queries, and (2) creating required input files.

CCS CONCEPTS

• **Information systems** → **Database query processing**; **Parallel and distributed DBMSs**; **Mediators and data integration**; • **Applied computing** → **Information integration and interoperability**; • **Computing methodologies** → *Knowledge representation and reasoning*.

KEYWORDS

Heterogeneous Databases, Data Lake, SQL, NoSQL, SPARQL, Query

ACM Reference Format:
Mohamed Nadjib Mami, Damien Graux, Simon Scerri, Hajira Jabeen, and Sören Auer. 2019. Querying Data Lakes using Spark and Presto. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314132

1 INTRODUCTION

During the last four decades, a variety of data storage and management techniques have been developed in both research and industry. Today, we benefit from a multitude of storage solutions, varying in their data model (e.g. tabular, document, graph) or their ability to scale storage and querying. There are dozens of continuously evolving storage and data management solutions; for the NoSQL family,

*This research was partially supported by the European Union's H2020 research and innovation programme BETTER under the Grant Agreement number 776280.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314132

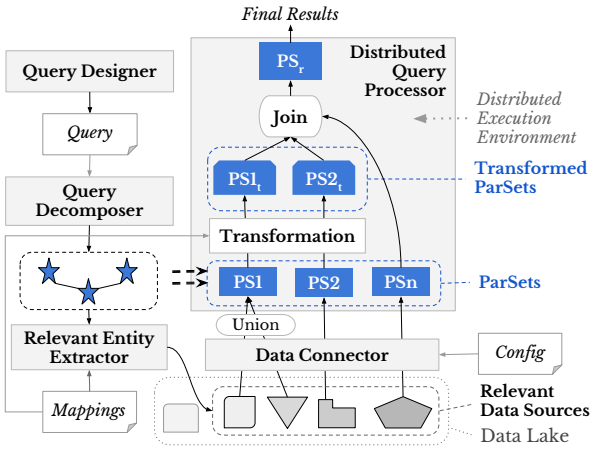


Figure 1: Squerall High-level Architecture.

Cassandra, MongoDB, Couchbase, HBase, Neo4j, DynamoDB, are just a few examples. As a result, users can choose a system that suits their individual application needs.

However, those systems do not inter-operate, every stored datum is locked in the respective system it is stored in. For example, an e-commerce company might store *product* information in a Cassandra database, *offers* in MongoDB to benefit from its capability to store hierarchical multi-level values, and information about *Producers* obtained from an external source in a relational format. Without transforming and moving the data into a unified (scalable) data management solution, the data can hardly be explored and business insights be extracted using ad hoc uniform querying.

We have taken on the mission of bridging this gap and developed Squerall[1], a software that allows to query heterogeneous data directly in its original form and source. We have used standardized and time-proven Semantic Web techniques to enable the uniform querying of heterogeneous data. We support the mapping of terms in the original data to terms in higher-level ontologies, and the querying of the resulting uniform view using SPARQL [13].

Similar efforts to integrate and query large data sources exist in the literature. For instance, [4] defines a mapping language to express access links to NoSQL databases. [12] allows to run CRUD operations over NoSQL databases. [1] proposes a unifying *programming* model to directly access databases using *get*, *put* and *delete* primitives. [8] proposes a SQL-like language containing invocations

[1]Associated website: <https://eis-bonn.github.io/Squerall/>

to the native query interface of relational and NoSQL databases. [7] is a hybrid platform with consideration for both heterogeneous and dynamic data sources (streams). However, Squerall offers the highest number of supported data sources while providing the richest query capability, including joining, aggregation and ordering.

We demonstrate Squerall's ability and efficiency in querying five different data sources (namely: CSV, Parquet, Cassandra, MongoDB and MySQL), and how it can be easily extended to support additional data sources, through several application scenarios.

2 PRINCIPLES AND ARCHITECTURE

In this section we introduce the basic principles and terminology needed to understand the architecture, subsequently described.

- **Mappings:** Squerall implements the so-called Ontology-Based Data Access (OBDA) [11] paradigm. In OBDA, data schemata are mapped to higher-level ontologies, forming a middleware against which SPARQL queries are posed. For example, *Offers(published, closed)* is a MongoDB collection mapped as follows: *(published → bsbm:validFrom, closed → bsbm:validTo, Offers → bsbm:Offer)* with *bsbm:validFrom*, *bsbm:validTo* and *bsbm:Offer* are two properties and a class of the BSBM ontology, respectively. Mappings are a core component of OBDA systems. Therefore, in order for every data source entity (Cassandra tables, MongoDB documents, etc.) to be queried using Squerall, having a mapping is a prerequisite.
- **Distributed Query Execution:** In Squerall, queries are executed in a separate distributed *environment*, which is, in particular, resilient to faults (node failure does not halt the entire query execution), and elastic and horizontally-scalable (more nodes can be added to accommodate more expensive computations). This is required for the following reasons:

- As we are dealing with dispersed data sources, intra-source query execution does not happen locally, but has to be brought into a more resilient, elastic and distributed environment, e.g. HDFS [3], or Spark's RDDs [14]. An exception is when certain sub-queries can be pushed down to the original source, which reduces what is loaded into the query execution environment.
- As we are dealing with large data volumes, query execution cannot be delegated to a single machine, as the intermediate results can overflow both its storage and computational capacity.
- In many modern NoSQL databases, in order to guarantee query performance, certain traditional query operators, e.g., join [10] are dropped. Such missing operations have to be executed in a higher-level capability environment.

- **Enabling joinability:** As data from different sources, e.g., *Product* and *Producer*, is generated by different applications, they may not be able to be readily cross-joined. Thus, modifications on the possible join values ought to be incorporated, e.g., to enable *Product* to join with *Producer*, modify `Product.producer_id` attribute values.

Squerall is comprised of five components (see Figure 1):

(1) **Query Decomposer:** Validates and analyzes SPARQL queries provided by a user. Particularly, the Query Decomposer extracts the Basic Graph Pattern (BGP, i.e., the conjunctive set of triple patterns given in the `where` clause) of the query and decomposes it into star-shaped sub-graph patterns having the same subject, *stars* for short. This component also detects links between stars, e.g., the triple `?product bsbm:producer ?producer` represents a link between the two stars represented by the variables `?product` and `?producer` (e.g., see colored boxes in Figure 1).

(2) **Relevant Entity Extractor:** Each star is analyzed separately; this component searches in the mappings for entities that are mapped to every predicate of the star. For example, given the graph pattern `{ ?x bsbm:validFrom ?y . ?x bsbm:validTo ?z ...}`, the entity *Offers(published, closed)* —a MongoDB collection, and the mappings *(published → bsbm:validFrom, closed → bsbm:validTo)*, this component decides that the entity *Offer* is *relevant* to the star of `?x`. Star type (when present in the query) is taken into account, e.g., `?x a bsbm:Offer`.

(3) **Data Connector:** Once relevant data entities are detected, they are connected to the distributed execution environment. Every detected entity is loaded into what we call a *ParSet* (short of Parallel dataSet), which are data structures that can be distributed and operated on in parallel. The Data Connector expects external users to input the necessary connection metadata, e.g., *user, password, host, port, cluster name*, etc.

(4) **Distributed Query Processor:** Following the principles introduced earlier, queries are executed in parallel. Query execution occurs on and across the ParSets. Links between stars retrieved by the Query Decomposer are transformed into joins between the relevant detected data entities. Figure 2 depicts the process: stars and links between stars (left colored boxes) are mapped to ParSets and joins between ParSets (step I). All stars are incrementally joined forming *Results ParSet* (step II). When disjoinability points are known, join values are altered to enable joinability. We incorporate *transformations* which users need to declare to determine what are the possible changes to make,

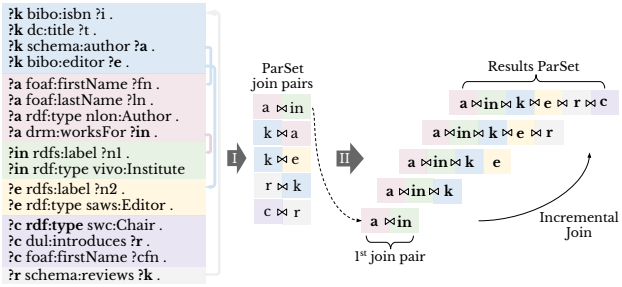


Figure 2: Example of query decomposition, and ParSets join.

Table 1: Query execution times (seconds) using Presto and Spark and the difference percentage between them (%).

	SPARQL SELECT Queries from BSBM (adapted)								
	Q1	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**	**Q8**	**Q10**
– Scale 1: scale factor of 0.5 millions products –									
Presto	55.34	28.89	15.84	53.63	49.24	43.38	18.63	14.38	89.08
Spark	98.78	189.57	59.96	277.30	222.76	191.26	159.51	91.38	300.38
Diff. %	**178.48**	**656.17**	**378.57**	**517.06**	**452.40**	**440.88**	**856.31**	**635.33**	**337.21**
– Scale 2: scale factor of 1.5 millions products –									
Presto	139.15	48.59	16.30	133.45	115.35	116.08	42.16	14.37	405.84
Spark	102.86	584.67	70.76	673.12	637.18	611.65	447.27	75.19	888.98
Diff. %	73.92	**1203.36**	**434.05**	**504.41**	**552.40**	**526.93**	**1060.83**	**523.31**	**219.05**
– Scale 3: scale factor of 5 millions products –									
Presto	276.91	131.87	30.58	340.61	350.04	334.29	98.11	18.96	784.01
Spark	132.37	1813.69	93.19	2131.10	1846.59	1833.47	1390.99	79.33	2703.43
Diff. %	47.80	**1375.40**	**304.71**	**625.67**	**527.54**	**548.46**	**1417.80**	**418.47**	**344.82**

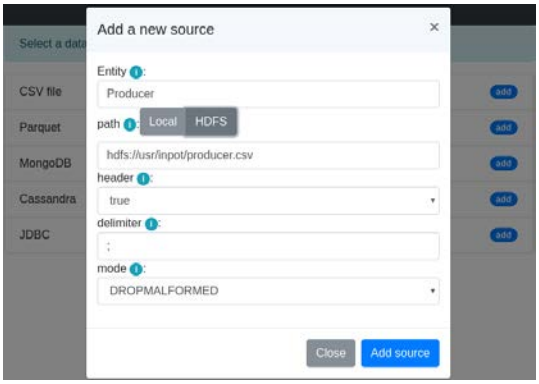


Figure 3: Data Connection GUI.

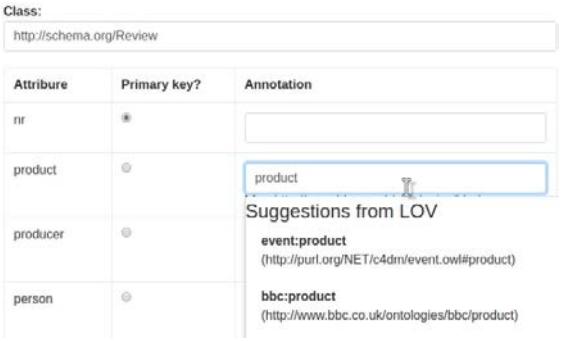


Figure 4: Data Mapping GUI.

e.g., increase the values of the `producer_id` attribute in the *Product* Cassandra table by a constant value.

(5) **Query Designer:** Querying the Data Lake using SPARQL as a uniform query language assumes a minimum level of knowledge about SPARQL. Therefore, we see the necessity of supporting users in their query creation. Users are not expected to provide queries only in plain-text, but can interact with the Query Designer to create correct queries in an intuitive way.

3 ACHIEVED PERFORMANCE

We evaluate the performance of `Squerall` in querying five different data sources: Cassandra, MySQL, MongoDB, Parquet, and CSV. As evaluation data we choose the BSBM [2] benchmark because it allows to (1) generate increasing scales of data (2) generate data in a friendly format, relational, and (3) provide a set of SPARQL queries. We pick five relational tables (*Product, Producer, Offer, Review, and Person*) and load them into the five data sources. To measure accuracy, we load the relational data into a relational database and run equivalent SQL queries. For sizes beyond the relational database capability (query time exceeds 3600s threshold) we self-compare Spark-based and Presto-based `Squerall`. For performance and scalability we evaluate the execution time of BSBM queries against three increasing data sizes: 0.5m, 1,5m and 5m scale factors (number of products). We experiment with all BSBM `SELECT` queries with some adaptation, i.e., queries are modified to involve only the five tables we populated. All the queries have join, from one between 2 data sources to 4 between all the data sources.

The evaluation was carried out in a cluster of three nodes, each having a 16-core DELL PowerEdge processor, 256GB RAM, and 3TB SATA disk. The evaluation results show 100% accuracy across all the three scales. For performance and scalability, `Squerall` exhibits reasonable performance across all three scales (see Table 1). Presto-based `Squerall` exhibits better performance to the Spark-based alternative in the majority of the cases. To measure this difference, we calculate the difference, in percentage, between Presto and Spark execution times (third row under each scale result in Table 1), e.g., for Q2 in the first scale Presto-based `Squerall` is 656% faster.

4 DEMONSTRABLE SCENARIOS

`Squerall` is a realization of the high-level architecture presented in section 2. We here describe its core technologies as well as concrete demonstrable scenarios using it.

1. Core technologies: *RML* [6] along FNO [5] are used to express mappings and to declare query-independent transformations. *Apache Spark* and *Presto* are used to implement both the Data Connector and Distributed Query Processor. Spark is a general-purpose processing engine, and Presto is a distributed SQL query engine for running interactive SQL queries. In Spark, ParSets are represented by *DataFrames*, and the Distributed Execution Environment is the memory pool where DataFrames are stored and transformed. DataFrames are tabular data structures, which are created and queried using *Spark SQL API*. Presto, in the other hand, is used like a database: users directly issue SQL queries without programmatically dealing with its underlying data structure. Thus, ParSets are represented by Presto's internal data structures, which are transparent to the user. Both Spark and Presto provide wrappers, called connectors[2] to connect to a data source. They load the data (fully, or partially if part of the query is pushed down to the data source) to their internal in-memory data structures. Dozens connectors are available relieving developers from creating their own wrappers.

2. User Interfaces: We provide 3 GUIs to help users produce `Squerall`'s inputs: *Config* and *Mappings* files and *SPARQL query*.

(a) **Data Connection** (Figure 3): This interface shows users the query engine-specific options needed to connect to a particular data source, e.g., *user, password, port, host, cluster name,* etc.

(b) **Mapping Creation** (Figure 4): using the connection information from the Data Connection GUI, this interface extracts the data schema and prompts users to map it. It has a built-in search functionality that sends requests to the LOV catalog[3] to search for adequate terms from existing ontologies.

(c) **Query Designing SPARQL** (Figure 5): This interface offers users a number of widgets to interact with. It is able to automatically generate a syntactically correct SPARQL query based on users' input. For example, users are prompted to only enter variable values or to pick from a pre-existing menu list, other

[2] <https://spark-packages.org> and <https://prestodb.io/docs/current/connector.html>
[3] Linked Open Vocabularies is a Web service to publish, search and visualize ontologies <https://lov.linkeddata.es/>.

constant query constructs are pre-entered.

3. Scenarios: Users can explore Squerall through a six-scenario story. They start by using Squerall's 3 GUIs to build the inputs Squerall needs for query execution. Afterwards, they run queries and observe the execution steps and results. Finally, they uncover Squerall's code and learn how it can be extended to support more data sources and query engines. Those scenarios use similar data as the ones used in the performance evaluation section 3.

(1) **Providing connection metadata.** Guided by the dedicated GUI, Figure 3, users provide necessary information to enable Spark and Presto to know and connect to a data source. For example, to load a CSV file, users specify the *delimiter* and select the *strategy* to adopt when the parser encounters a mal-formed line, e.g., dropping the line; or to connect to a MongoDB cluster, users specify the replica-set name.
(2) **Creating mappings.** Users leverage the dedicated interface, Figure 4, to create mappings for the previously connected data. They visualize the data schema and map it to ontology terms (properties and classes). They can try to find terms in existing ontologies using the LOV catalog-powered search; in absence of suitable results, they can input their own terms.
(3) **Building SPARQL queries.** Users explore and use the capabilities of the *Query Designing* GUI, Figure 5. They use the different widgets and incrementally augment their SPARQL query. They start building the query by forming stars (triple patterns of the same subject) and linking them together, then include other query operations like filtering. Users can experiment with the full spectrum of SPARQL fragment supported, which includes *filters, aggregate functions, solution modifiers, and regex*.
(4) **Data Transformation Declaration.** Users are invited to experiment with the two offered options of declaring transformations: (1) *as part of SPARQL query:* they add a new clause at the very end of a SPARQL query; (2) *as part of the mappings:* they amend RML mappings to, instead of mapping an entity attribute, e.g., column `producer_id`, directly to an ontology term, they first modify its values using declared FNO functions. The former is used when transformations vary from a query to query, the latter is used when transformations are fixed.
(5) **Executing SPARQL queries.** Users experiment with querying the data using Squerall. They can either evaluate BSBM queries or their own queries. They explore the query execution steps by visualizing the logs, from query parsing to stars and joins detection, to mapping lookup, and then to query execution.
(6) **Extending Squerall with more data sources and new query engines.** We show how developers can conveniently support more data sources in both Spark and Presto implementations. We also show them how they can programmatically incorporate a new query engine implementation, alongside Spark and Presto, thanks to Squerall's modular code design.

A screencast walking through the various interfaces and the query execution is available[4].



Figure 5: Query Designing GUI.

5 CONCLUSION

Squerall addresses the *Variety* challenge of Big Data, which remains poorly addressed, by making use of Semantic Web standards and best practices. Squerall can conveniently be (pragmatically) extended to embrace new data sources, by making use of the query engines' own wrappers. This approach solves one of the most tedious tasks acknowledged across the literature, i.e., handcrafting wrappers. For example, in addition to the five sources evaluated here, other sources like Couchbase or Elasticsearch can also be easily supported. As a result, Squerall is both innovative and unique in its capability to support a high number of data source types. Additionally, Squerall has been integrated[5] into SANSA [9], a framework for scalable processing and analysis of large-scale RDF data, widening its scope to also access non-RDF data sources. As future work, we plan to expand the supported SPARQL fragment to include OPTIONAL and UNION. Squerall source code is publicly available under an Apache-2.0 license on GitHub[6], accompanied by detailed documentation on installation and usage. This is further facilitated with a Dockerfile to quickly run the BSBM use-case described here (including the input files and adapted queries).

REFERENCES

[1] Paolo Atzeni, Francesca Bugiotti, and Luca Rossi. 2012. Uniform access to non-relational database systems: The SOS platform. In *International Conference on Advanced Information Systems Engineering*. Springer, 160–174.
[2] Christian Bizer and Andreas Schultz. 2009. The berlin SPARQL benchmark. *International Journal on Semantic Web and Information Systems* 5, 2 (2009), 1–24.
[3] Dhruba Borthakur et al. 2008. HDFS architecture guide. *Hadoop Apache Project* 53 (2008), 1–13.
[4] Olivier Curé, Robin Hecht, Chan Le Duc, and Myriam Lamolle. 2011. Data integration over nosql stores using access path based mappings. In *International Conference on Database and Expert Systems Applications*. Springer, 481–495.
[5] Ben De Meester, Wouter Maroy, Anastasia Dimou, Ruben Verborgh, and Erik Mannens. 2017. Declarative data transformations for Linked Data generation: the case of DBpedia. In *European Semantic Web Conference*. Springer, 33–48.
[6] Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. 2014. RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data.. In *LDOW*.
[7] Martin Giese, Ahmet Soylu, Guillermo Vega-Gorgojo, Arild Waaler, Peter Haase, Ernesto Jiménez-Ruiz, Davide Lanti, Martín Rezk, Guohui Xiao, Özgür Özçep, et al. 2015. Optique: Zooming in on big data. *Computer* 48, 3 (2015), 60–67.
[8] Boyan Kolev, Patrick Valduriez, Carlyna Bondiombouy, Ricardo Jimenez-Peris, Raquel Pau, and José Pereira. 2016. CloudMdsQL: querying heterogeneous cloud data stores with a common language. *Distributed and parallel databases* 34, 4 (2016), 463–503.

[4] <https://github.com/EIS-Bonn/Squerall/blob/master/evaluation/screencasts>

[5] <https://github.com/SANSA-Stack/SANSA-DataLake>
[6] <https://github.com/EIS-Bonn/Squerall>

[9] Jens Lehmann, Gezim Sejdiu, Lorenz Bühmann, Patrick Westphal, Claus Stadler, Ivan Ermilov, Simon Bin, Nilesh Chakraborty, Muhammad Saleem, Axel-Cyrille Ngonga Ngomo, and Hajira Jabeen. 2017. Distributed Semantic Analytics using the SANSA Stack. In *ISWC Resources Track*.

[10] Franck Michel, Catherine Faron-Zucker, and Johan Montagnat. 2016. A mapping-based method to query MongoDB documents with SPARQL. In *International Conference on Database and Expert Systems Applications*. Springer, 52–67.

[11] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. 2008. Linking data to ontologies. In *Journal on Data Semantics X*. Springer, 133–173.

[12] Rami Sellami and Bruno Defude. 2018. Complex queries optimization and evaluation over relational and NoSQL data stores in cloud environments. *IEEE Transactions on Big Data* 4, 2 (2018), 217–230.

[13] W3C SPARQL Working Group et al. 2013. SPARQL 1.1 Overview. http://www.w3.org/TR/sparql11-overview/.

[14] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J Franklin, Scott Shenker, and Ion Stoica. 2012. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *USENIX conference on Networked Systems Design and Implementation*. USENIX Association, 2–2.

Automated Fact Checking in the News Room

Sebastião Miranda
Priberam Labs
Lisbon, Portugal
ssm@priberam.com

David Nogueira
Priberam Labs
Lisbon, Portugal
dan@priberam.com

Afonso Mendes
Priberam Informática, S.A.
Lisbon, Portugal
amm@priberam.com

Andreas Vlachos
Dep. of Computer Science and Technology, University of Cambridge
Cambridge, United Kingdom
andreas.vlachos@cst.cam.ac.uk

Andrew Secker
BBC News Labs
London, United Kingdom
andrew.secker@bbc.co.uk

Rebecca Garrett
BBC Monitoring
London, United Kingdom
becky.garrett@bbc.co.uk

Jeff Mitchel
Psychology Dep., University of Bristol
Bristol, United Kingdom
jeff.mitchell@bristol.ac.uk

Zita Marinho
Priberam Labs
Institute of Systems and Robotics, IST
Lisbon, Portugal
zam@priberam.com

ABSTRACT

Fact checking is an essential task in journalism; its importance has been highlighted due to recently increased concerns and efforts in combating misinformation. In this paper, we present an automated fact checking platform which given a claim, it retrieves relevant textual evidence from a document collection, predicts whether each piece of evidence supports or refutes the claim, and returns a final verdict. We describe the architecture of the system and the user interface, focusing on the choices made to improve its user friendliness and transparency. We conduct a user study of the fact-checking platform in a journalistic setting: we integrated it with a collection of news articles and provide an evaluation of the platform using feedback from journalists in their workflow. We found that the predictions of our platform were correct 58% of the time, and 59% of the returned evidence was relevant.

CCS CONCEPTS

• **Computing Methodologies** → **AI**; *NLP*; Information Extraction; • **Human Centered Computing** → **HCI**; *HCI design and evaluation methodologies*; User studies; Usability testing; • **Information Systems** → **Information Retrieval**; *Evaluation of retrieval results*; Presentation of retrieval results; *Retrieval tasks and goals*; Question answering.

KEYWORDS

Fact Checking; Computational Journalism; Media Tools

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314135

ACM Reference Format:
Sebastião Miranda, David Nogueira, Afonso Mendes, Andreas Vlachos, Andrew Secker, Rebecca Garrett, Jeff Mitchel, and Zita Marinho. 2019. Automated Fact Checking in the News Room. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314135

1 INTRODUCTION

Grounding information on reliable sources is a daunting experience, given the increasing amount of information circulating the web and other media platforms. Nevertheless, checking the veracity of claims is crucial for preserving trust in news sources. Manual verification of claims is a tedious task, that consumes a lot of time and effort from journalists and professional fact-checkers, and it typically requires searching for specific entities and content over large amounts of unstructured text [2, 4].

The rising interest in fact checking has led to the development of a number of approaches and tools automating the task or parts of it, with the motivation of facilitating the work of journalists, and interested readers more broadly [3, 9]. The most popular effort, ClaimBuster [5], proposed a fact checking platform which detects factual claims that are worth checking and then uses APIs to query search engines and databases (Google and Wolfram Alpha respectively). It also compares claims against previously fact checked ones in its database. The latter approach is also used by the system developed by Full Fact, Live, which is used to fact check repeated or paraphrased claims.[1] Neither of these approaches is able to fact check previously unchecked claims, while the queries through existing commercial APIs are not tailored to fact checking, thus retrieving information that is not necessarily relevant. Other approaches rely instead on the detection of rumours based on the spread of readership over social media [7]. However, a rumorous claim is not necessarily false, and vice versa [16]. There is also work that checks claims against tables, such as those released by

[1] https://fullfact.org/blog/2017/jun/automated-fact-checking-full-fact/

Claim:
Tesla builds car factory in Shanghai.

Evidence:
Electric carmaker Tesla has signed an agreement with Chinese authorities to build a factory in Shanghai. "We hope it will be completed very soon," Tesla chief Elon Musk said.

Figure 1: Example of claim and evidence. Extracted named entities in bold: organizations (purple), locations (blue) and people (green).

the World Bank[2], using trained classifiers to select the appropriate tuple [10]. However such approaches are typically restricted to fact checking numerical claims against tabular sources. Finally, some approaches aim at providing a pipeline of tools for information retrieval [14], but do not go as far as to provide an actual fact check mechanism.

In this work, we propose an automated fact checking platform that checks claims by identifying sentences providing evidence in a large document collection. These sentences are classified as supporting, refuting or only related to the claim, and then combined into a final verdict using a state-of-the-art neural network-based approach [15]. The model is trained on the recently released FEVER dataset [11], a large scale fact checking dataset derived from Wikipedia comprising 185K claims. Unlike previous work, our model is able to check novel claims without relying on a database of fact checks. Also, the evidence retrieved and classified as supporting or refuting provides a justification of the verdict, which is likely to be relevant not only to assess the overall correctness of the platform, but also as part of the fact checking research conducted by journalists. Our retrieval model considers more information about a specific claim than generic search engines by leveraging information from words and named entities present in the dataset to obtain the best matching evidence for each claim (see Figure 1).

We thoroughly evaluate the platform through user testing by journalists from the British Broadcasting Corporation (BBC) in the context of their workflow. Of the total 488 evidence passages retrieved by the system, the journalists reported that 58% were relevant and 59% were accurately classified as SUPPORTS/REFUTES/OTHER.

Our platform can be applied to document collections beyond the ones used in this paper and our findings should help inform future research in automated fact checking and computational journalism more broadly.

2 FACT CHECKING SYSTEM

Our fact-checking system comprises three main components: a document retrieval step, a sentence ranking, and a classification model, as shown in Figure 2.

Initially we retrieve documents from a collection of news articles, using a customized search engine based on inverse indexing and retrieval using the Okapi-BM25 algorithm [8]. Data is incrementally indexed from word and entity-level inverted indexes. The document retrieval component searches for documents whose features best match the claim. In the end of this step, we end up with approximately $10K$ documents related to the claim.

[2]https://data.worldbank.org/

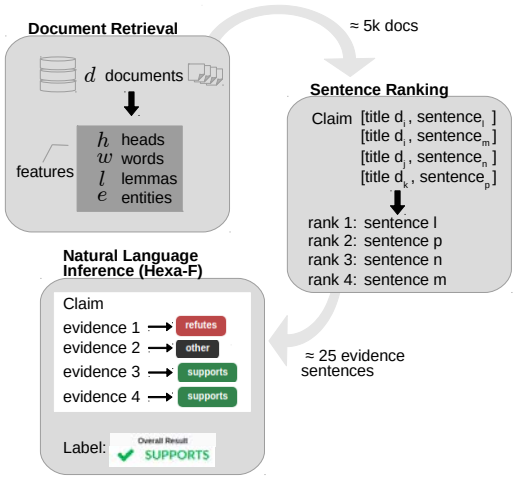


Figure 2: Fact-checking model comprised of three components: a) Document retrieval (top left), b) Sentence ranking (middle right), c) NLI prediction model (bottom).

Following this, we rank the sentences in these documents according to predefined token and feature matching rules. We compute the cosine similarity of the claim with each best candidate sentence, using word embeddings trained on the One Billion Word Benchmark corpus [1], and select those above a threshold tuned during development. This step aims at providing only sentences with high relevance to the claim, reducing the number of potential evidence that may be only marginally related with the claim.

In the final component, we classify each of the extracted candidate evidence in terms of whether they support, refute or are just related to the claim (other). We employ the natural language inference (NLI) model from the Hexa-F system [15] (one of the best performing systems in the FEVER shared task [12]) to classify the relation between the selected evidence sentences and the claim, one of SUPPORTS/REFUTES/OTHER, and a similar label which expresses whether the combined set of evidence sentences SUPPORTS, REFUTES or is simply related to the claim.

2.1 Engineering Considerations

In the first document retrieval component, we use inverted indexes to extract relevant documents, in Figure 3. We select all documents that match features in the claim, such as lemmas, words and extracted named entities [6]. This document retrieval step takes about 50 ms to retrieve around 5k documents out of a dataset with about $445k$ documents.

In the second component, we rank all the sentences from the 5k documents based on how well each sentence feature $\phi(s_i)_j$ matches the claim $\phi(c)_j$, using a positional ranking approach. We use a ranking score based on ordered distances between N matching features in the sentence i: $S_1(s_i, c) = \sum_{j=1}^{N} \exp\left(-d_{i,j}\right)$, where $d_{i,j} = \text{pos}(\phi(s_i)_j) - \text{pos}(\phi(s_i)_{j-1})$ represents the word distance between the position (pos) of each j-th feature. This score is maximized if all words in the sentence match the claim exactly; it decreases exponentially with the word distance between matches. This part takes on average 336 ms. Next, we filter the sentences based on

the following rules: we keep only those sentences with length < 500 words; those that contain all the entities mentioned in the claim, as well as novel words that were not previously seen in previously selected sentences (less than 90% overlap with all words previously encountered). The novelty filter increases the diversity in the evidence passed to the entailment step.

In the end of the second component, we re-rank the sentences by averaging the feature matching score of each sentence S_1 and the cosine similarity between the claim and the sentence embeddings $S_2 = \cos(s_i, c)$. We obtain these embeddings via a weighted average of all words in the sentence, weighted by term-frequency/inverse document frequency. To improve the performance of the sentence re-ranking component, we added the title of the document to each sentence and considered the weighted sum of all words in the title and sentence combined. This part takes about 652 ms, although it could be easily parallelizable. On average, 76 sentences are retrived per claim after these steps.

We further removed all sentences with an averaged similarity score below a given threshold ($(S_1 + S_2)/2 < 0.6$), to ensure high quality evidence. In the end, about 25 sentences on average are selected. After re-ranking, all selected sentences are used as input in the NLI model (third component) which takes about 738 ms to predict labels for each of the sentences and the overall label for the claim.

3 USER INTERFACE

The interface allows end-users to input a claim (black bar at the top of Figure 3), and receive a set of evidence sentences as output. The evidence is displayed in three columns: the top five ranking sentences that are in favour of the claim on the left, the top five that are against the claim in the middle, and the top five other sentences related to the claim on the right. In the bottom, a final overall label is presented to the user as either other, supporting or refuting the claim (label at the bottom). The interface allows users to directly evaluate it, providing feedback for evidence sentence w.r.t. the correctness of the label ("correct label?") and its relevance ("relevant?"), as well as the correctness of the overall prediction. We provide a video with an example of a user interacting with the platform via the interface in https://vimeo.com/309336679.

4 USER EVALUATION DESIGN

11 BBC journalists provided feedback on the overall system and on the classification model. The journalists were asked to interact with the system by providing factual claims and evaluating the output of the model. For each claim, up to 15 evidence sentences were presented to them, 5 per category, each classified as SUPPORTS, REFUTES or OTHER. For each sentence, the journalists provided feedback on two aspects: relevance and correctness.

To assess correctness, for each evidence sentence returned by the system, the journalist inputs the label which, according to his/her research on the subject, would be the correct one via the buttons in the "correct label" box (see Figure 3). This measures primarily the accuracy of the entailment component of the system, assuming that the sentences returned are all related. For the final classification of the model (see Figure 3 (bottom)) they also assessed the overall prediction, to whether the claim was globally supported or refuted considering all the retrieved evidence.

Precision	Class			
	SUPPORTS	REFUTES	OTHER	ALL
Relevant	71%	69%	49%	59 %
Evidence Correctness	48%	27%	70%	58 %
Global Correctness	56%	26%	31%	42 %

Table 1: User evaluation on the full dataset.

To assess relevance, the journalists were also asked to provide feedback as to whether they found each sentence returned relevant (see the buttons in the "relevant ?" box in Figure 3). This part of the feedback aims to evaluate the quality of the retrieved evidence: whether it helps the journalists fact-check the input claim, regardless of the classification label attributed by the system. It also serves as a proxy to measure the precision of the retrieval component, as all sentences shown to the journalists should be relevant (ideally).

The journalists also had access to (i) a Question Answering (QA) [13] tool that could serve as an additional source of information, (ii) the full document's text with annotated entities [6]. The QA implemented in the platform was used by the journalists, but was not evaluated in this paper. We provide an example of the additional information, available for each extracted evidence sentence in Figure 4.

5 RESULTS

Table 1 summarizes the results of the system is the user evaluation conducted. We show precision for each class (row:Relevant) measuring the proportion of the retrieved evidence sentences that were deemed relevant by the journalists, and precision for the predictions of the platform for the relation of each evidence sentence to the claim (row:Evidence) and the global prediction for the claim taking all the evidence into account (Overall result presented in Figure 3) (row:Global). The platform was evaluated on 67 claims in total, with a total of 488 evidence sentences retrieved: 30% supporting, 14% refuting and 56% related to the claim, as classified by our platform.

In 71% of the claims checked by the journalists, it was reported that the evidence shown in the SUPPORTS column was relevant, and so was for 69% of the evidence in the REFUTES columns. We consider these results to be very encouraging, given the difficulty of the task. Retrieving evidence that contradicts a given claim, is not usually as simple as retrieving related evidence by feature matching. Introducing different information, e.g. entities, dates, actions, etc., that could refute the claim requires more complex language understanding methods. We further observe that the precision in predicting evidence as SUPPORTS is 48% in the full dataset and increases to 67% in the subset of evidence deemed relevant (not shown in the table). The same trend is observed for the REFUTES label from 27% to 39%. This result suggests that the retrieval component still requires some improvement, especially for retrieving evidence refuting the claim. Strategies beyond feature matching are needed to improve the retrieval of relevant but opposing arguments. The precision of the classifier predicting the global label of the claim given the evidence also requires further improvement.

Additionally, we received textual feedback from the journalists about the overall quality of the platform. They mentioned that it is helpful for fact checking, despite not being always accurate, both in the retrieval of relevant evidence and in their evaluation. An interesting remark mentioned possible improvements for handling

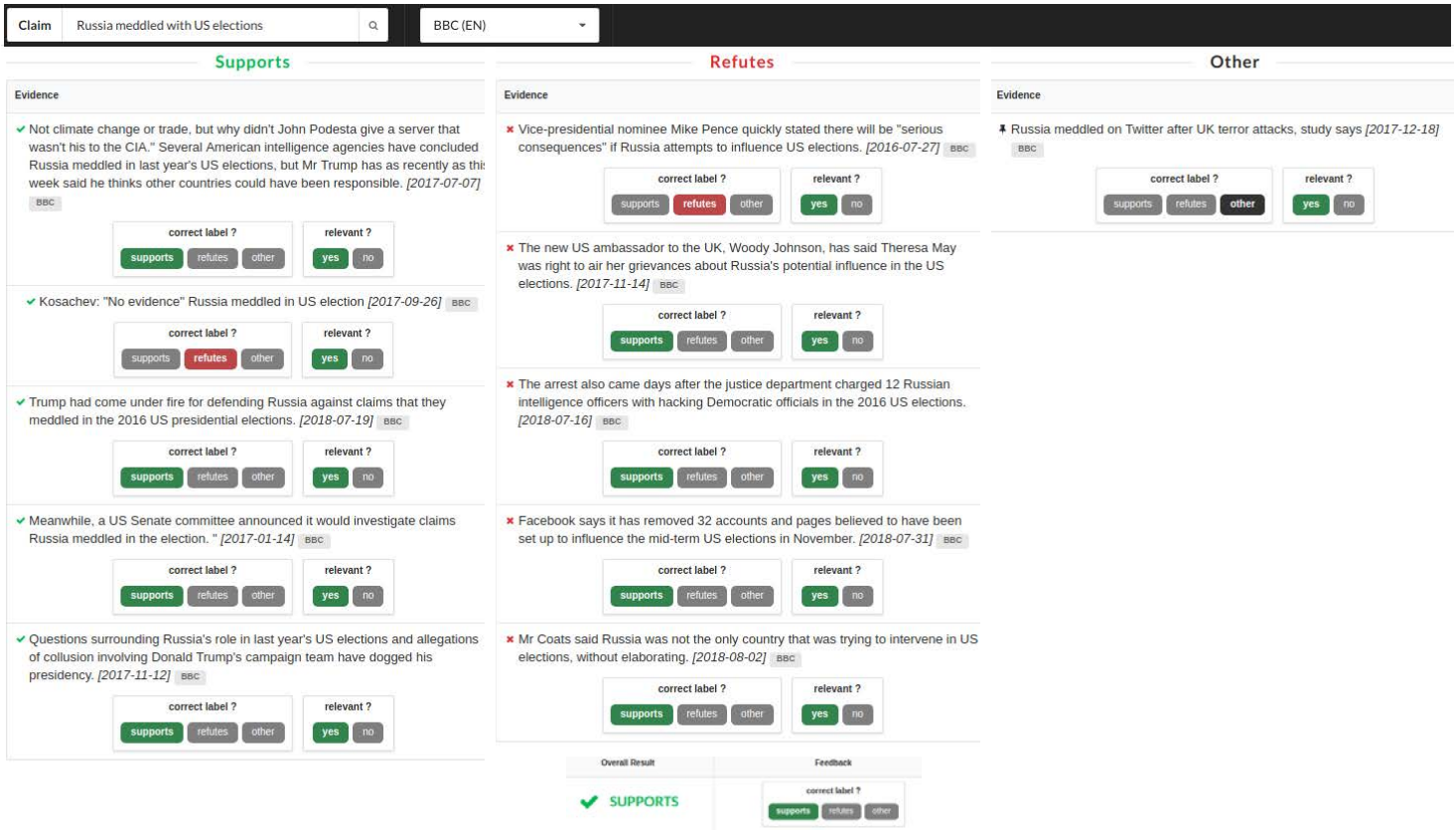


Figure 3: Example of the fact checking interface. Claim: "Russia meddled with US elections" (top). Five maximum evidence sentences for each column: SUPPORTS/REFUTES/OTHER **(middle). Example of final system decision and feedback buttons (bottom).**

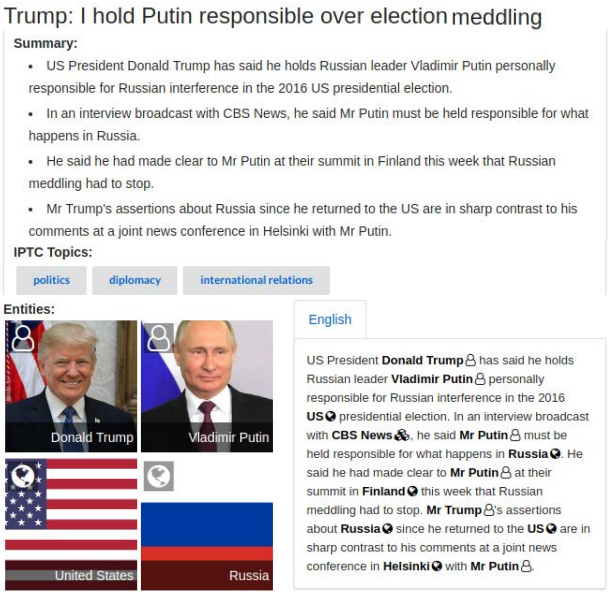


Figure 4: Example of additional information for each evidence. Original document (right) with entities in bold.

time-frames and dates. For instance, claims using the present tense should refer to current events, while those mentioned in the past tense together with dates should refer to that specific time-period only. Also they suggested that presenting the evolution of results over time would be very helpful to substantiate the claim. Handling time constraints in the retrieval process is a very interesting and challenging research direction. On the whole, the journalists reported that the system has a lot of potential to help their work.

This user testing was extremely useful both for BBC journalists to experiment with state-of-the-art technology and for us to receive feedback to improve our platform in the future.

6 CONCLUSIONS AND FUTURE WORK

This paper introduces a novel fact checking platform aimed to assist journalists in their investigative work-flow. Our platform can be used for search of supporting and refuting evidence regarding factual claims. We evaluated using on a journalistic corpus with testing by eleven journalists, which found it to yield relevant results in 59% of the retrieved evidence. The performed user study provided very fruitful feedback to direct future work in automated fact checking. Suggested improvements such as handling temporal remarks, pose an interesting issue that we found very relevant to advance research in the field of information retrieval for fact checking.

ACKNOWLEDGMENTS

The authors would like to thank James Thorne for insightful discussion and the BBC journalists for preforming the user testing and for their valuable feedback. This work is supported by the EU H2020 SUMMA project (grant agreement № 688139), and by Lisbon Regional Operational Programme (Lisboa 2020), under the Portugal 2020 Partnership Agreement, through the European Regional Development Fund (ERDF), within project INSIGHT (№ 033869).

REFERENCES

[1] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. *One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling*. Technical Report. Google.

[2] Sarah Cohen, Chengkai Li, Jun Yang, and Cong Yu. 2011. Computational Journalism: a call to arms to database researchers. *CIDR 2011, California, USA*. January (2011), 148–151.

[3] Lucas Graves. 2018. *Understanding the Promise and Limits of Automated Fact-Checking*. Technical Report. Reuters Institute, University of Oxford.

[4] Naeemul Hassan, Bill Adair, James T Hamilton, Chengkai Li, Mark Tremayne, Jun Yang, and Cong Yu. 2015. The Quest to Automate Fact-Checking. In *Proceedings of the 2015 Computation+Journalism Symposium*.

[5] Naeemul Hassan, Gensheng Zhang, Fatma Arslan, Josue Caraballo, Damian Jimenez, Siddhant Gawsane, Shohedul Hasan, Minumol Joseph, Aaditya Kulkarni, Anil Kumar Nayak, et al. 2017. ClaimBuster: the first-ever end-to-end fact-checking system. *Proceedings of the VLDB Endowment* 10, 12 (2017), 1945–1948.

[6] Afonso Mendes, David Nogueira, Samuel Broscheit, Filipe Aleixo, Pedro Balage, Rui Martins, Sebastião Miranda, and Mariana Almeida. 2017. SUMMA at TAC Knowledge Base Population Task 2017. In *Proceedings of the Text Analysis Conference KBP 2017*. NIST, USA.

[7] Vahed Qazvinian, Emily Rosengren, Dragomir R Radev, and Qiaozhu Mei. 2011. Rumor has it : Identifying Misinformation in Microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 1589–1599.

[8] Stephen E. Robertson, Steve Walker, and M Beaulieu. 2000. Experimentation as a way of life: Okapi at TREC. *Information processing & management* 36, 1 (2000), 95–108.

[9] James Thorne, , and Andreas Vlachos. 2018. Automated Fact Checking: Task Formulations, Methods and Future Directions. In *Proceedings of the 27th International Conference on Computational Linguistics*. ACL, 3346–3359.

[10] James Thorne and Andreas Vlachos. 2017. An extensible framework for verification of numerical claims. In *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 37–40.

[11] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a Large-scale Dataset for Fact Extraction and VERification. In *NAACL-HLT*.

[12] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Workshop on Fact Extraction and VERification (FEVER). In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*. ACL.

[13] Dirk Weissenborn, Pasquale Minervini, Isabelle Augenstein, Johannes Welbl, Tim Rocktäschel, Matko Bosnjak, Jeff Mitchell, Thomas Demeester, Tim Dettmers, Pontus Stenetorp, et al. 2018. Jack the Reader. In *Proceedings of the 56th of ACL system demonstrations*. 25–30.

[14] Gregor Wiedemann and Andreas Niekler. 2014. Document Retrieval for Large Scale Content Analysis using Contextualized Dictionaries. In *Terminology and Knowledge Engineering 2014*. Berlin, Germany, 10 p.

[15] Takuma Yoneda, Jeff Mitchell, Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Four Factor Framework For Fact Finding (HexaF). In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*. ACL, 97–102.

[16] Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. Detection and Resolution of Rumours in Social Media: A Survey. *ACM Comput. Surv.* 51, 2, Article 32 (Feb. 2018), 36 pages.

InfraNodus: Generating Insight Using Text Network Analysis

Dmitry Paranyushkin
Nodus Labs, Paris, France, dmitry@noduslabs.com

ABSTRACT

In this paper we present a web-based open source tool and a method for generating insight from any text or discourse using text network analysis. The tool (InfraNodus) can be used by researchers and writers to organize and to better understand their notes, to measure the level of bias in discourse, and to identify the parts of the discourse where there is a potential for insight and new ideas. The method is based on text network analysis algorithm, which represents any text as a network and identifies the most influential words in a discourse based on the terms' co-occurrence. Graph community detection algorithm is then applied in order to identify the different topical clusters, which represent the main topics in the text as well as the relations between them. The community structure is used in conjunction with other measures to identify the level of bias or cognitive diversity of the discourse. Finally, the structural gaps in the graph can indicate the parts of the discourse where the connections are lacking, therefore highlighting the areas where there's a potential for new ideas. The tool can be used as stand-alone software by end users as well as implemented via an API into other tools. Another interesting application is in the field of recommendation systems: structural gaps could indicate potentially interesting non-trivial connections to any connected datasets.

CCS CONCEPTS

• Human-centered computing~Visualization systems and tools • Applied computing~Document analysis • Information systems~Information extraction • Networks~Topology analysis and generation

KEYWORDS

text mining; mental maps; text network analysis; discourse bias; bias; network analysis; graph theory; TNA; information interfaces; network topology; cognitive science; topic modelling; ideation; comprehension; creativity; research; insight;

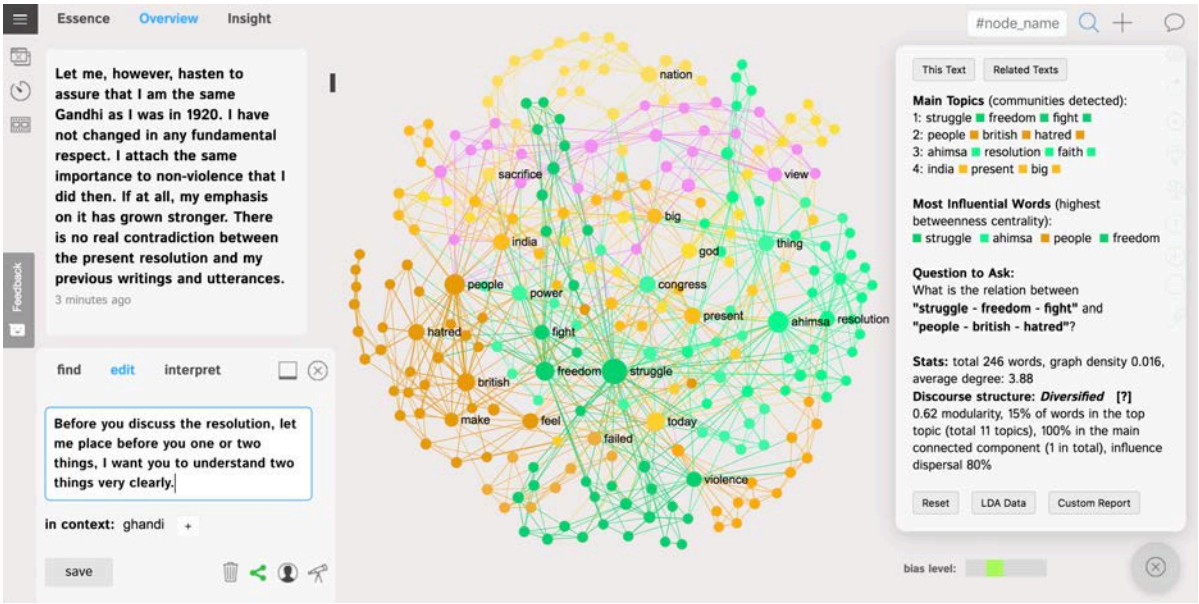


Figure 1: InfraNodus interface visualizing the text, the main topics inside, discourse structure, and bias index.

1 Introduction and Background

Mental models, mental maps and mind maps are often used to represent complex ideas [1], [2] and have a positive impact on teaching and learning [3], [4], [5]. Mental maps can be extracted from texts to produce cognitive maps, which can then be used for sentiment analysis [6], [7]. In the recent years this field has been enhanced with the network metaphor: the nodes are the concepts and the connections are the relations between them [8], [9]. The relations between the concepts can be plotted on the basis of their co-occurrences, which, in turn can facilitate the process of semantic priming — the process by which words tend to be recognized faster when used with the words that have close semantic proximity to them [10], [11], [12], [13]. Text network analysis methods based on this approach have been shown to be successful for better comprehension of texts and topic modeling [14], [15], [6].

In this paper we introduce an open-source tool for text network analysis InfraNodus[1], which can be used to enhance visual representation of ideas, their consequent analysis, and to generate insight. InfraNodus implements a method for text network analysis based on plotting a text as a network graph. The nodes on this graph represent the words and the edges are their co-occurrences [16], [17], [18]. Once a text is represented as a graph in this way, a wide range of tools from network analysis are then applied to detect communities of closely related concepts or topical clusters (topic modeling), identify the most influential nodes (top keywords), perform both quantitative and qualitative analysis of text, evaluate the discourse structure and bias, identify structural gaps, and perform comparative analysis of several texts [16], [19], [20]. Moreover, visual representation of a text as a network used in the process of ideation helps towards a more coherent expression of ideas [21], so the tool may find useful application in creative writing, research, in educational context, and in therapy.

State-of-the-art and its limitations. Similar tools have been presented in scientific literature: iVisClustering [22], TopicNets [23] and VisiRR [24] among others. However, those tools are not available to general public, their source code is closed, they are mainly focused on studying a large corpus of texts, and their application is limited to topic modelling and keyword-based text retrieval. InfraNodus is available both online and as a standalone open-source version. Its interface is specifically designed for real time text processing, providing text network visualization and analysis live as the new data is added. Finally, InfraNodus offers discourse structure analysis providing the measurement of the bias level in discourse and identifying the structural gaps in discourse, which is not available in other tools and methods, to our knowledge.

Relation to other text mining approaches. The approach implemented in InfraNodus can enhance the existing methods of finding structure within text that employ graphical models and topic modeling: latent semantic analysis or LSA [25], pLSA [26], Pachinco allocation [27], latent dirichlet allocation or LDA [28], relational topic models [29], word2vec algorithm [30], [31] and its extension lda2vec [32]. These methods are based on retrieving the topics from text by identifying the clusters of co-occurrent words within them, based on the bag-of-words and skip-gram models. This data can then be used to classify similar documents, improve text indexing and retrieval methods, and to identify evolution of certain topics overs a period of time within a specific text corpus [33].

The method used in InfraNodus is different from the probabilistic methods outlined above (LDA, pLSA, word2vec, lda2vec etc) even though its underlying assumption is similar: identifying the words that occur closer to each other in text can be used for topic modelling. InfraNodus uses graph theory instead of probability distribution to identify the related words and assign them into topical clusters. This application of graph theory helps gain a better understanding of the textual discourse structure and to provide advanced visualization tools, which can be useful both for quantitative and qualitative analysis.

The main difference is that methods like LDA or word2vec will take co-occurrences into account, but they do not provide insight into the structure of a narrative. These methods may ascribe words to the same topic (following bag-of-words model), but there will be no information provided about the relation of these topics to each other: how connected they are, how equally represented they are in a text, etc. The method implemented in InfraNodus provides this insight both qualitatively and quantitatively. Another issue with probabilistic methods is that they often require prior training on a document corpus, and they may be quite difficult to understand, to implement, and to interpret. The complex math behind LDA or word2vec may be difficult to access for the people who cannot afford to go into the technical details. Text network analysis approach used in InfraNodus, on the other side, does not require any training, it can work in real-time during the process of writing on a simple document, it is very intuitive and the math behind it is relatively simple. It is also much easier to understand intuitively as it provides direct visual representation.

At the same time, InfraNodus also offers LDA topic modelling and keyword extraction functionality, so that users can enhance the data obtained using LDA with additional information about the narrative structure and the topical cluster data obtained using network analysis (and vice versa). The method provided in InfraNodus does not compete with other topic modelling methods and does not attempt to outperform them. It provides the ability to get a better understanding of the text network topology, which can yield many other insights about the discourse structure that are not available in LDA or word2vec: how connected a discourse is, whether it's biased towards a certain set of concepts, how dominant a specific topic is towards the whole discourse, whether there are structural gaps that may indicate a potential for new ideas.

2 The Method and Its Implementation

InfraNodus is an open-source software written in JavaScript (Node.Js) using Sigma.Js, Cytoscape and Graphology libraries in the front-end and java-based Neo4J graph database for the backend[2]. It is platform-independent and can run both on desktop (hosted) and mobile (cloud) devices. Its source code is open and is available under the GPL open-source license on www.github.com/noduslabs/infranodus. Users can either install InfraNodus on their own machine or use the cloud-based version on www.infranodus.com. The resulting graphs and results can be embedded on other websites and exported into popular graph formats for further processing and analysis (e.g. Gephi). Some of the functionality is also available via an API, so InfraNodus can be used in conjunction with other text mining and analysis software.

In this section we will demonstrate step-by-step implementation of the text network analysis method as it is applied in InfraNodus. We will also demonstrate the qualitative and quantitative insights that can be obtained by the researchers using this method. For this demonstration we will use a well-known speech by Mahatma Gandhi "Quit India" [34].The interactive network visualization with all the analytics presented in this paper can be accessed via www.infranodus.com/texts/ghandi.

[1] Online: www.infranodus.com, source: github.com/noduslabs/infranodus, video tutorials: www.infranodus.com/#tutorials & https://youtu.be/bMmTRBZpjWw

[2] www.nodejs.org is a JavaScript run-time environment, www.sigmajs.org is a network visualization library created by Alexis Jacomy, graphology.github.io is a library for graph analysis created by Guillaume Plique, Cytoscape.org is a platform for complex network analysis, Neo4J.org is a graph database

2.1 Step 1: Text Normalization

First, all the words in the text are converted into their lemmas to reduce redundancy — keeping the morphological root of each word, bringing the different variations of the same word to the same common denominator (e.g. "books" become "book", "contacted" becomes "contact"). The syntax information (",", ".") is also removed. The paragraph structure is retained.

2.2 Step 2: Stop words Removal

The words that function as liaisons and that do not carry any additional meaning are removed from the text (e.g. "is", "are", "a", "the", etc.) The users can adjust the list of stop words used at that point using the Settings pane in the software or apply the tf–idf method to automatically generate the stopwords relevant to a document corpus.

The result of applying the steps 1 and 2 above is a word sequence, e.g. a sentence that reads:
Before you discuss the resolution, let me place before you one or two things, I want you to understand two things very clearly.

becomes
discuss resolution place thing want understand thing clear

2.3 Step 3: Text-to-Network Conversion

The text is then converted into a directed network graph. The normalized words (lemmas) are the nodes in the network graph and their co-occurrences are the edges.

Two consecutive scans are performed. The first scan creates the connections (graph edges) between the lemmas (nodes) that appear next to each other (bigrams) with the edge weight value of 3. The second scan uses a window of 4 lemmas (4-grams) to add a second layer of connections. The connections between the lemmas that are are only 1 word apart get the edge weight value of 2, while the lemmas that are separated by 2 words get the edge weight value of 1:

[discuss → (weight: 3) → resolution], [discuss → (weight: 2) → place], [discuss → (weight: 1) → thing],
[resolution → (weight: 3) → place], [resolution → (weight: 2) → thing], [resolution → (weight: 1) → want], etc.

These connections are encoded as edges on a directed text graph and are given the proper weight during analysis and visualization. Optionally, the links between the paragraphs can be made, where the last word of the previous paragraph links to the first word of the next one with the edge weight value of 1.

All the connections are unique and are saved even if they appeared earlier in the text. Later, for the purposes of graph visualization and community detection, the weights of the parallel edges are summed up.

2.4 Step 4: Extracting Most Influential Keywords Using Betweenness Centrality

We then apply a ranking algorithm to identify the nodes (the words) with the highest betweenness centrality – these are the nodes that appear most often on the shortest paths between any two randomly chosen nodes in the network [35], [36]. Those words are central for meaning circulation and can be seen as the meaning junctions within the discourse — they are shown bigger on the graph.

In the example Gandhi's text, we obtain these most influential words that have the highest betweenness centrality[3]:
struggle - ahimsa - people - freedom (see Figure 1)

2.5 Step 5: Topic Modelling Using Community Detection and Force-Atlas Layout

We then apply community detection algorithm [37], [38] based on modularity. This is an iterative algorithm that detects the groups of nodes that are more densely connected together than with the rest of the network. As a result, we obtain the groups of nodes (words) which tend to appear together in the text: topical clusters. We then apply Force-Atlas algorithm [39], which aligns densely connected clusters together while pushing the most connected nodes apart, so that the network structure is more visible on the graph. We then get a visual network representation of the text with a clearly defined community structure (using both color and network topology) and the specific topical clusters (see Figure 1).

The topics identified[4] are displayed on the separate Analytics pane:
1 (green): struggle - freedom - fight
2 (brown): people - british - hatred
3 (light green): ahimsa - resolution - faith
4 (orange): india – present - big

2.6 Step 6: Summarization, Visual Text Search and Nonlinear Reading

The graph visualization is interactive. After the main topics are identified, InfraNodus highlights the parts of the text that contain the highest concentration of topics identified (using the Analytics or Essence panes).

The user can also select any nodes on the graph to find the part of the text that mentions the topics selected (see Figure 2). The user can also compare two or more different text graphs using the text "diff" feature.

Therefore, the graph can be used to read through the text in a non-linear way, following the concepts that are relevant — using the graph as a heuristic device to make sense of the text or to look at it from a new perspective.

[3] LDA method (also available in InfraNodus / Analytics pane) yields similar results for the top 4 keywords in this text: "struggle – people – british - ahimsa".
[4] According to LDA, the top 4 topics are "ahimsa – resolution - present", "british – people - hatred", "struggle – freedom - today", "power – india - congress".

2.7 Step 7: Discourse Structure and the Measure of Discourse Bias

InfraNodus also identifies the structure of discourse based on the structure of its text network graph. The algorithm takes into account (1) the graph's modularity (M — a value higher than 0.4 indicates a highly pronounced community structure [38]); (2) the percentage of words in the topical cluster with the most nodes, or in the main connected (giant) component (C), if there are several connected components in the graph; (3) the dispersal of influence among the topical clusters (communities) measured in terms of Shannon entropy (E) for a string that contains a 1-letter code of the community assigned to each of the top 4 most influential keywords (e.g. if each of the top 4 nodes belongs to a different community A, B, C and D, the string will be 'ABCD' and the entropy in bits per symbol will be 2).

Based on these criteria a score is given to the text graph, which can be used to measure the level of bias in discourse:

- **Dispersed**: high modularity (M > 0.65) and (C < 50% and E >= 1.5) - the community structure is highly pronounced, the influential words are distributed among the communities. Therefore, the discourse has several topics, which are weakly related.
- **Diversified**: high modularity (0.6 >= M > 0.4) and (C < 50% and E >= 1.5) - pronounced community structure, the most influential words are distributed among the communities. Therefore, the discourse has several topics, each of them has a relatively high number of nodes in the graph, the topics are connected.
- **Focused**: medium modularity (0.4 > M >= 0.2 and E > 0.5) or (M > 0.4 and C >= 0.5 and 0.25 < E < 0.75) - communities are present but not easily detectable, influential words may be concentrated around one topic, the discourse is focused on a certain perspective, the other perspectives are weakly presented.
- **Biased**: low modularity (M < 0.2), no detectable communities, influential words are concentrated around one subject - the discourse is biased

For example, the structure of the discourse in our example is "Diversified" meaning that it has several distinct topics and that those topics are still connected on the global level, the bias level is low (see Figure 1).

2.8 Step 8: Insight Generation using Structural Gaps

Graph visualization is also used to identify the structural gaps in the graph. This can be done both in a qualitative way by the researchers or automatically by the software's algorithm (detecting the distinct communities that are not well connected). It has been demonstrated [40], [41] that structural gaps are the parts of a graph that indicate the potential for new ideas. Text network visualization can be used to identify those gaps and to propose new connections between different ideas, topics and topical clusters [5].

In the example we use those are
"british" - "people" - "hatred" and *"struggle" - "freedom" - "fight"*

A quick analysis of the text snippets that include those topics and of the graph shows that there is a structural gap between the nationalist sentiment that Ghandi is talking about (in relation to the "hatred" for "British") and at the same time about the fact that, in his opinion, revolutions that are based on violence do not live up to democratic ideals. The algorithm shows that there is an important link between these two ideas (see Figure 3).

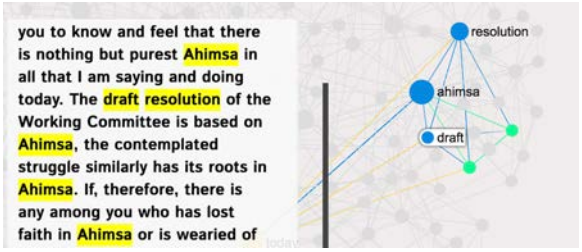


Figure 2: Visual graph text search feature.

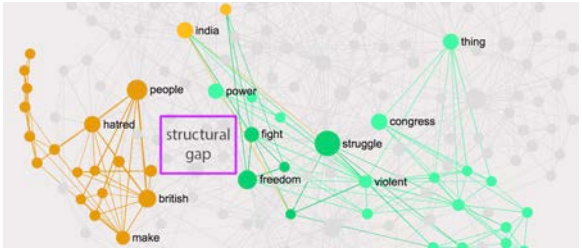


Figure 3: Structural gaps point out potential insights.

3 Conclusion and Practical Applications

In the previous section we demonstrated the implementation of text network visualization and analysis algorithm in InfraNodus software. We have shown how it can used to obtain a visual representation of text, to perform topic modeling, to extract relevant parts using the interactive interface and the visual search, to compare different texts, to analyze the structure of discourse, and to identify the structural gaps to generate new ideas in order to further develop the discourse.

The tool is currently used by researchers, marketing professionals, students, lawyers, artists and activists worldwide (20000 users a year according to Google Analytics for the online version as of December 2018) and it became first available in its beta version in 2014. The range of its practical applications is quite diverse: text categorization, search engine optimization, measure of bias, sentiment analysis, computer-assisted research and creative writing.

The structural gap identification feature can be potentially useful for enhancing recommender systems with non-trivial suggestions based on bridging the different clusters of disconnected datasets together, providing interesting insight for the users.

ACKNOWLEDGMENTS

This research was supported solely by the contributions made by the users of the online version of InfraNodus.

[5] We could also apply the same methodology to the topical clusters identified using LDA topic modelling (see footnote 4)

REFERENCES

[1] M. Minsky, "A Framework for Representing Knowledge," *Artificial Intelligence, Memo no. 306,* 1975.

[2] J. Sowa, "Semantic Networks," in *Encyclopedia of Cognitive Science*, 1992.

[3] Y. Liu, G. Zhao, G. Ma and Y. Bo, "The Effect of Mind Mapping on Teaching and Learning: A Meta-Analysis," *Standard Journal of Education and Essay,* vol. 2(1), p. 017– 031, 2014.

[4] M. Davies, "Concept mapping, mind mapping and argument mapping: what are the differences and do they matter?," *Higher Education,* vol. 62, no. 3, p. 279–301, 2011.

[5] J. Redford, K. Thiede, J. Wiley and T. Griffin, "Concept mapping improves metacomprehension accuracy among 7th graders," *Learning and Instruction,* vol. 22, no. 4, pp. 262-270, 2012.

[6] K. Carley, "Extracting Team Mental Models through Textual Analysis," *Journal of Organizational Behavior,* 1997.

[7] D. Jonassen and Y. Cho, "Externalizing Mental Models with Mindtools," in *Understanding Models for Learning and Instruction*, Boston, Springer, 2008.

[8] S. S. Sonawane and P. A. Kulkarni, "Graph based Representation and Analysis of Text Document: A Survey of Techniques.," *International Journal of Computer Applications,* vol. 96, no. 19, 2014.

[9] E. Castillo, O. Cervantes and D. Vilariño, "Text Analysis Using Different Graph-Based Representations," *Computación y Sistemas,* vol. 21, no. 4, 2018.

[10] R. Popping, "Knowledge Graphs and Network Text Analysis," *Social Science Information,* 2003.

[11] T. Heyman, B. Van Rensbergen, G. Storms, K. A. Hutchison and S. De Deyne, "The influence of working memory load on semantic priming," *Journal of Experimental Psychology: Learning, Memory, and Cognition,* vol. 41, no. 3, pp. 911-920, 2015.

[12] S. Evert and G. Lapesa, "Evaluating Neighbor Rank and Distance Measures as Predictors of Semantic Priming," *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics,* p. 66–74, 2013.

[13] J. Bullinaria and J. Levy, "Extracting semantic representations from word co-occurrence statistics: A computational study," *Behavior Research Methods,* no. 39, 2007.

[14] F. D. Malliaros and K. Skianis, "Graph-Based Term Weighting for Text Categorization," *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining - ASONAM '15,* p. 1473–1479, 2015.

[15] B. Paley, "TextArc text visualization software," 2002.

[16] S. Corman, T. Kuhn, D. Mcphee and K. Dooley, "Studying Complex Discursive Systems: Centering Resonance Analysis of Communication," *Human Communication Research,* vol. 28, no. 2, p. 157–206, 2002.

[17] D. Paranyushkin, "Identifying the pathways for meaning circulation using text network analysis.," *Nodus Labs,* 2011.

[18] F. Rousseau and M. Vazirgiannis, "Graph-of-word and TW-IDF: new approach to ad hoc IR," *Proceedings of the 22nd ACM international conference on information & knowledge management - CIKM '13,* p. 59–68, 2013.

[19] D. Paranyushkin, "Addresses to the Federal Assembly of the Russian Federation by Russian presidents, 2008–2012: comparative analysis," *Russian Journal of Communication,* vol. 5, no. 3, pp. 265-274, 2013.

[20] F. D. Malliaros and K. Skianis, "Graph-Based Term Weighting for Text Categorization," in *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015 - ASONAM '15*, 2015.

[21] D. Paranyushkin, "Direct Visual Feedback on the Process of Ideation using Text Network Graphs Encourages a more Coherent Expression of Ideas," *Nodus Labs,* 2018.

[22] H. Lee, J. Kihm, J. Choo, J. Stasko and H. Park, "iVisClustering: An Interactive Visual Document Clustering via Topic Modeling," *Computer Graphics Forum,* vol. 31, no. 3, p. 1155–1164, 2012.

[23] B. Gretarsson, J. O'Donovan, S. Bostandjiev, T. Höllerer, A. Asuncion, D. Newman and P. Smyth, "TopicNets: Visual Analysis of Large Text Corpora with Topic Modeling," *ACM Transactions on Intelligent Systems and Technology,* vol. 3, no. 2, p. 1–26, 2012.

[24] J. Choo, H. Kim, E. Clarkson, Z. Liu, C. Lee, F. Li and H. Park, "VisIRR: A Visual Analytics System for Information Retrieval and Recommendation in Large-Scale Document Data," *ACM Transactions on Knowledge Discovery from Data,* vol. 20, 2018.

[25] T. K. Landauer, P. W. Foltz and D. Laham, "An introduction to latent semantic analysis," *Discourse Processes,* no. 25, pp. 259-284, 1998.

[26] G. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 1999.

[27] W. Li and A. McCallum, "Pachinko Allocation: DAG-Structured Mixture Models of Topic Correlations," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

[28] D. Blei, A. Ng and M. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research,* vol. 1, no. 03, pp. 993-1022, 2003.

[29] J. Chang and D. Blei, "Hierarchical Relational Models for Document Networks," *Annals of Applied Statistics,* vol. 4, no. 1, p. 124–150, 2010.

[30] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *Advances in Neural Information Processing Systems,* no. 26, p. 3111–3119, 2013.

[31] Y. Goldberg and O. Levy, "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method," 2014.

[32] C. E. Moody, "Mixing Dirichlet Topic Models and Word Embeddings to Make lda2vec," 2016.

[33] D. Blei, "Probabilistic Models of Text and Images," p. 2004.

[34] M. Gandhi, "Quit India," 1942. [Online]. Available: https://en.wikipedia.org/wiki/Quit_India_speech. [Accessed 15 09 2018].

[35] L. Freeman, "A Set of Measures of Centrality Based on Betweenness," *Sociometry,* vol. 40, no. 1, pp. 35-41, 1977.

[36] A. Brandes, "Faster Algorithm for Betweenness Centrality," *Journal of Mathematical Sociology,* vol. 25, no. 2, pp. 163-177, 2001.

[37] S. Fortunato, "Community Detection in Graphs," *Complex Networks and Systems Lagrange Laboratory,* 2010.

[38] V. Blondel, J.-L. Guillaume, R. Lambiotte and E. Lefebvre, "Fast Unfolding of Communities in Large Networks," *Journal of Statistical Mechanics: Theory and Experiment,* vol. 2008, 2008.

[39] M. Jacomy, T. Venturini, S. Heymann and M. Bastian, "ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software," *PLoS ONE,* vol. 9, no. 6, 2014.

[40] R. Burt, Structural holes: The social structure of competition, 1992.

[41] L. Noy, Y. Hart, N. Andrew, O. Ramote, A. Mayo and U. Alon, "A quantitative study of creative leaps," in *International Conference on Computational Creativity*, 2012.

Bridging Screen Readers and Voice Assistants for Enhanced Eyes-Free Web Search

Alexandra Vtyurina
University of Waterloo
Waterloo, Ontario, Canada
sasha.vtyurina@uwaterloo.ca

Adam Fourney
Microsoft Research
Redmond, WA, USA
adamfo@microsoft.com

Meredith Ringel Morris
Microsoft Research
Redmond, WA, USA
merrie@microsoft.com

Leah Findlater
University of Washington
Seattle, WA, USA
leahkf@uw.edu

Ryen W. White
Microsoft Research
Redmond, WA, USA
ryenw@microsoft.com

ABSTRACT

People with visual impairments often rely on screen readers when interacting with computer systems. Increasingly, these individuals also make extensive use of voice-based virtual assistants (VAs). We conducted a survey of 53 people who are legally blind to identify the strengths and weaknesses of both technologies, as well as the unmet opportunities at their intersection. We learned that virtual assistants are convenient and accessible, but lack the ability to deeply engage with content (e.g., read beyond the first few sentences of Wikipedia), and the ability to get a quick overview of the landscape (list alternative search results & suggestions). In contrast, screen readers allow for deep engagement with content (when content is accessible), and provide fine-grained navigation & control, but at the cost of increased complexity, and reduced walk-up-and-use convenience. In this demonstration, we showcase VERSE, a system that combines the positive aspects of VAs and screen readers, and allows other devices (e.g., smart watches) to serve as optional input accelerators. Together, these features allow people with visual impairments to deeply engage with web content through voice interaction.

CCS CONCEPTS

• **Information systems → Search interfaces**; • **Human-centered computing → Accessibility systems and tools**; **Natural language interfaces**; **Sound-based input / output**.

KEYWORDS

Virtual assistants, voice search, screen readers, accessibility

ACM Reference Format:
Alexandra Vtyurina, Adam Fourney, Meredith Ringel Morris, Leah Findlater, Ryen W. White. 2019. Bridging Screen Readers and Voice Assistants for Enhanced Eyes-Free Web Search. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314136

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314136

1 INTRODUCTION

People with visual impairments are often expert users of audio-based interfaces, with screen readers being a prime example. Screen readers work by transforming the visual content in a graphical user interface into audio by vocalizing on-screen text. To this end, they are an important accessibility tool for people who are blind – so much so that every major operating system includes screen reader functionality (e.g., VoiceOver[1], TalkBack[2], Narrator[3]), and there is a strong market for third-party offerings (e.g., JAWS[4], NVDA[5]). Despite their importance, screen readers have many limitations. For example, they are complex to master, and depend on the cooperation of content creators to provide accessible markup (e.g., *alt text* for images). This includes the myriad of web page owners who host documents on the Internet.

Voice activated virtual assistants (VAs), such as Apple's Siri, Amazon's Alexa, and Microsoft's Cortana, offer another audio-based interaction paradigm, and are mostly used for everyday tasks such as controlling a music player, checking the weather, and setting up reminders [6]. In addition to these household tasks, however, voice assistants are also used for general purpose web search and information access [4]. Here, in contrast to screen readers, VAs are marketed to a general audience, and are limited to shallow investigations of web content. Being proficient users of audio-based interfaces, people who are blind often use VAs, and would benefit from broader VA capabilities [1, 5]. Additionally, extended VA functionality could be useful for a wider audience during activities such as driving and cooking.

In this work, we explore augmenting a VA interaction model with basic functionality of screen readers to better support free-form, voice-based web search. Through an online survey with 53 blind screen reader and VA users, we investigated what challenges people experience when searching the web with a screen reader and when getting information from a voice assistant. Based on these findings, we developed VERSE (Voice Exploration, Retrieval, and SEarch) – a prototype that employs a VA model, yet provides rich functionality for web exploration. In the remainder of this document we: (1) present a quick overview of the survey findings that motivate the

[1]https://www.apple.com/accessibility/mac/vision/
[2]https://support.google.com/accessibility/android/answer/6283677
[3]https://www.microsoft.com/en-us/accessibility/windows
[4]https://www.freedomscientific.com/Products/Blindness/JAWS
[5]https://www.nvaccess.org/

design of VERSE, (2) present an overview of the VERSE system, and (3) conclude by outlining the hardware and logistics requirements for demonstrating VERSE at TheWebConf'19.

2 ONLINE SURVEY

To better understand the problem space of non-visual web search with screen readers and VAs, we designed an online survey consisting of 44 questions spanning five categories: general demographics, use of screen readers for accessing information in a web browser, use of virtual assistants for retrieving online information, comparisons of screen readers to virtual assistants for information seeking tasks, and possible future integration scenarios (e.g., voice-controlled screen readers).

We recruited adults living in the U.S. who are legally blind and who use both screen readers and voice assistants. Through a collaboration with an online survey organization that specializes in recruiting people with various disabilities, we ensured that all participants could successfully access the survey. The survey took an average of 49 minutes to complete, and participants were compensated $50 for their time. Two researchers iteratively analyzed the open-ended responses using techniques for open coding and affinity diagramming [2] to identify themes.

2.1 Participants

A total of 53 respondents completed the survey (28 female, 25 male). Participants were diverse in age, education level, and employment status. All participants reported being legally blind, and most had experienced visual disability for a prolonged period of time (μ = 31.6 years, σ = 17 years). As such, all but three respondents reported having more than three years of experience with screen reader technology. Likewise, most participants were experienced users of voice assistant technology. 35 respondents (66%) reported having more than three years of experience with such systems, 17 participants (32%) – between one and three years, and one participant (2%) reported having used VA technology for less than a year.

2.2 Findings

We found that respondents made frequent and extensive use of both virtual assistants and screen reader-equipped web browsers to search for information online, but both methods had shortcomings. Moreover, we found transitioning between VAs and browsers introduced additional challenges and opportunities for future integration. Each of these trade-offs is codified by a theme below.

2.2.1 ***Theme 1:*** *Brevity vs. Detail.* The amount of information provided by voice assistants can differ substantially from that returned by a search engine. VAs provide a single answer (suitable for simple question answering but not for exploratory search tasks [8]), that may be short and provide limited insight. This aspect was pointed out by 27 respondents. For example P24 noted that: *"a virtual assistant will only give you one or two choices, and if one of the choices isn't the answer you are seeking, it's hard to find any other information"*. These concerns were echoed concisely by P37: *"you just get one answer and sometimes it's not even the one you were looking for"*, and P30: *"a lot of times, a virtual assistant typically uses one or two sources in order to find the information requested, rather than the entire web"*. In contrast, 20 respondents reported that using a search engine via a screen reader-equipped browser affords access to multiple sources, and access to more details if needed (e.g., P46: *"you can study detailed information more thoroughly"*). But those details come at a price – when using a screen reader a user has to cut through the clutter on web pages before getting to the main content, as mentioned by 8 survey respondents (e.g., P18: *"you don't get the information directly but instead have to sometimes hunt through lots of clutter on a web page to find what you are looking for"*).

2.2.2 ***Theme 2:*** *Granularity of Control vs Ease of Use.* Our survey participants widely recognized that VAs were a convenient tool for performing simple tasks (22 people), but greater control was needed for in-depth exploration (e.g., P38: *"They are good for specific, very tailored tasks."*). This trade-off in control, between VAs and screen reader-equipped browsers, was apparent at all stages of performing a search: query formulation (P30: *"[with VAs] you have to be more exact and precise as to the type of information you are seeking."*), results navigation (P22: "*[with screen readers] I can navigate through [results] as I wish*"), and information management (P51: "*If I use a screen reader for web searching I can bookmark the page and return to it later. I cannot do it with a virtual assistant.*")

Additionally, 15 respondents reported that screen readers are advantageous in that they provide a greater number of navigation modes, each operating at different granularities. For example, P18 reports: *"[with screen readers] you can navigate by heading, landmark or words"*. Similar sentiments are reported by P24: *"It's easier to scan the headings with a screen reader when searching the web"*, and P31: *"one is able to navigate through available results much faster than is possible with virtual assistants."*

Finally, screen readers afford greater control by allowing users to customize multiple settings (speech rate, pitch) to fit people's preferences – a functionality not yet available in voice assistants (P29: *"sometimes you can get what you need quicker by going down a web page, rather then waiting for the assistant to finish speaking"*). The desire for customization of VAs was mentioned by only one participant in our survey, but has been identified as a limitation of VAs in prior work [1].

The increased dexterity of screen readers comes at a price of having to memorize many keyboard commands or touch gestures, whereas VAs require minimal to no training (P38: *"[with VAs] you don't have to remember to use multiple screen reader keyboard commands"*). This specific tradeoff was mentioned by 3 participants.

2.2.3 ***Theme 3:*** *Transitioning between systems.* Another prominent theme detailed the frequent need for users to transition from a VA to a screen reader-equipped web browser. Out of 53 survey respondents, 39 recalled recent situations where they began a session with a VA but then had to switch to using a web browser. Reasons for switching mentioned in participants' incident descriptions included: failure of automatic speech recognition (4 people), a VA response that lacked sufficient details (11 people), or the lack of any relevant response (14 people).

Transitions between system are not well-supported at present, and respondents suggested numerous ways in which this could be improved. For example P24 notes: *"A virtual assistant could give you basic information and then provide a link to view more in depth results using a screen reader."* Likewise P21 suggested that, upon performing in-depth search, the VA "*(could) ask you if you wanted more details.*

If you replied yes, it would open a web page such as Google [in a browser] and perform a search". Such a strategy would save people from having to re-input their query and begin a completely new search session.

2.2.4 ***Theme 4:*** *Incidental vs. Intentional Accessibility.* Finally, one of the valuable features of voice assistants is their audio-first design. Thus, while targeting a general audience, VAs are immediately, and incidentally, accessible to people with visual impairments. This was mentioned by 7 participants. For example, P38 reports: *"You don't have to worry about dealing with inaccessible websites"*, while P42 notes that such an approach *"levels the playing field, as it were [since] everyone searches the same way."*

3 VERSE

Inspired by our survey findings, we created VERSE (Voice Exploration, Retrieval and SEarch), a prototype situated at the intersection of voice-based virtual assistants and screen readers. People interact with VERSE primarily through speech, in a manner similar to existing voice-based agents such as Amazon Alexa or Google Assistant. For example, when asked a direct question VERSE will often respond directly with a concise answer (Figure 1a). However, VERSE differs from existing agents in that it enables an additional set of voice commands that allow users to access different online sources and search engine features (such as related searches), as well as to engage more deeply with content for select sources (for example, allowing navigation over a document's headings).

As with screen readers, VERSE addresses the need to provide shortcuts and accelerators for common actions. To this end, VERSE optionally allows users to perform gestures on a companion device such as a phone or smart watch (see Table 2). For most actions, these companion devices are not strictly necessary. However, to simplify rapid prototyping, we limited microphone activation to gestures, rather than also allowing activation via keyword spotting (e.g., "Hey Google"). Specifically, microphone activation is implemented as a double-tap gesture performed on a companion device (e.g., smartphone or smartwatch). Although hands-free interaction can be a key functionality for VA users [3], a physical activation is a welcomed ancillary, and at times, a preferred option [1]. There are no technological blockers for implementing voice-only activation in the future versions of VERSE.

The following scenario, and the video accompanying this paper [7], illustrate VERSE's capabilities, and indicate how VERSE could be demonstrated at TheWebConf'19.

3.1 Example Demonstration Scenario

Alice recently overheard a conversation about the Challenger Deep and is interested to learn what it is. She is sitting on a couch, her computer is in another room, and a VERSE-enabled speaker is on the coffee table. Alice activates VERSE and asks "What is the Challenger Deep?". The VERSE speaker responds with a quick answer – similar to Alice's other smart speakers – but also notes that VERSE found a number of other web pages, Wikipedia articles, and related searches (Table 1a). Alice decides to explore the Wikipedia articles ("Go to Wikipedia"), and begins navigating the list of related Wikipedia entries ("next") before backtracking to the first article, this time rotating the crown on her smart watch as a shortcut to quickly issue the *previous* command (Table 1b).

Alice decides that the first Wikipedia article sounded good after all, and asks for more details ("Tell me more"). VERSE loads the Wikipedia article and begins reading from the introduction section (Table 1c), but Alice interrupts and asks for a list of section titles ("Read section titles"). Upon hearing that there is a section about the Challenger Deep's history, Alice asks for it by section name ("Read History section").

Finally, Alice wonders if there may be other useful resources beyond Wikipedia, and decides to return to the search results ("Go to search results"). As before, Alice rotates the crown on her smart watch to quickly scroll through the results. Alice identifies an interesting webpage from the list VERSE reads out to her, and decides to explore it more deeply on her phone ("Send this to my phone") and the chosen web page opens on her iPhone (Table 1d) where Alice can navigate it using the phone's screen reader.

3.2 VERSE Design Elements

The design of VERSE was informed by a select set of themes that emerged during the analysis of survey responses. Below we discuss how VERSE addresses each of the themes.

3.2.1 VERSE: Brevity vs. Detail. The design of VERSE addresses *Theme 1* by providing users with an instant concise answer (in a manner similar to VAs) as well as an opportunity to explore multiple sources returned by a web search engine (breadth), and then to engage with individual content items and web sources (depth).

With respect to breadth, VERSE leverages the Bing.com search API[6] to collect results across numerous search verticals: web pages, facts, entities, dictionary definitions, Wikipedia articles, news stories, images, videos, and related search queries. If facts, entities, or definitions are present, VERSE reads them out similar to existing VAs, then follows by summarizing the results available in other verticals (see Table 1a).

With respect to depth, VERSE allows voice- and gesture-based navigation of Wikipedia articles. We chose Wikipedia as it has rich data, is often included among the top results, and has a consistent page structure that facilitates screen reader-like navigation. When a desired Wikipedia article is selected, the user can say "tell me more," or perform an alternative gesture (Table 2) to get a quick summary about the article (e.g., the number of sections and words), then hear narration of the first section. At any time, the user can ask for an overview of the article's sections (*"read section titles,"*), and can ask for a given section by name (*"read <section name>"*). This interaction is illustrated in Table 1c.

3.2.2 VERSE: Granularity vs. Ease of Use. To address *Theme 2* from the survey findings, VERSE allows users a quick and easy way to navigate between search results using either voice commands or touch gestures. By saying *"next"* or *"previous,"* the user is able to move on to the next element in the selected search vertical (Table 1b). A similar effect is achieved by swiping right and left on a companion device (Table 2). These gestures mirror those used by screen readers on popular smart phones.

[6] https://azure.microsoft.com/en-us/services/cognitive-services/bing-web-search-api/

Table 1: Example usage scenario

(a) Initial search

Alice: *(activation)* What is the Challenger Deep?
VERSE: The Challenger Deep is the deepest known point in the Earth's seabed hydrosphere. With the depth 10898 to 10916m by direct measurements from submersibles and slightly more by sonar bathymetry. I also found 2 entities, 9 web pages, 8 related search queries, 10 videos, 10 Wikipedia articles.

(b) Results navigation

Alice: *(activation)* Go to Wikipedia.
VERSE: I found 10 Wikipedia articles. The first one is Challenger Deep, Wikipedia. The Challenger Deep is the...
Alice: *(activation)* next article
VERSE: Article 2. Deepsea Challenger, Wikipedia. Deep Sea Challenger, DCV1, is 7.3 metre, 24 feet...
Alice: (*rotate smartwatch crown to return to previous element*)
VERSE: Article 1. Challenger Deep, Wikipedia. The Challenger Deep is the deepest ...

(c) Wikipedia exploration

Alice: *(activation)* Tell me more.
VERSE: I parsed the Wikipedia article called Challenger Deep. It has 16 sections and 3127 words. Here's the first section. Introduction. The Challenger Deep is the deepest...
Alice: *(activation)* Read section titles.
Verse: Introduction. Section 1: History of depth mapping from the surface. Section 2: Descents. Section 2.1...
Alice: *(activation)* Read History section.
VERSE: Okay, here's the section number 1, titled History of depth mapping from the surface. Over many years, the search for the point of maximum depth...

(d) Transition between modalities

Alice: *(activation)* Go to web results.
VERSE: I found 9 web pages. The first one is Challenger Deep, Wikipedia.
Alice: *(rotates smartwatch crown to advance to next result)*
VERSE: Page 2. Deepest ocean Challenger Deep, Extreme Science. Challenger Deep got its name from...
Alice:*(activation)* Send this to my phone.
VERSE: Opening this link in your phone app. *(Web page opens within VERSE app)*
Alice: *(uses VoiceOver gestures to navigate on the page)*

To switch between different search verticals, a user can say *"go to <search source>"* (e.g., *"Go to Wikipedia."*). VERSE will respond with the number of elements found in the new vertical and start reading the first element (Table 1b). Alternatively, the user can swipe up or down to move along the available search sources.

Finally, when exploring Wikipedia articles, VERSE also supports screen reader-inspired navigation modes (by headings, sentences, paragraphs, and words). The navigation mode then impacts the granularity of navigation commands & gestures, such as *"next"* and *"previous"*. Without loss of generality, one can switch modes by saying *"navigate by headings"*, or can swipe up or down on a companion device to iterate between modes – again, these gestures are familiar to people who use screen readers on mobile devices.

3.2.3 VERSE: Transitioning between modalities. VERSE addresses *Theme 3* by giving users an opportunity to seamlessly transition between voice-based interaction and a more traditional screen reader-equipped web browser. If the user requests an in-depth exploration of a web resource that is not Wikipedia, VERSE will open its url within the VERSE phone application. The user can then explore the web page using the device's built-in screen reader (in our case, VoiceOver). From this point onward, all gestures are routed to the default screen-reader until a "scrub" gesture is performed[7], or a new voice query, is issued. Gesture parity between VERSE and the screen reader ensures a smooth transition. This interaction is illustrated in Table 1d.

3.2.4 VERSE: Incidental vs Intentional Accessibility. Finally, as already noted, VERSE submits user queries, and retrieves results, via Bing.com search API. This allowed us to design a truly audio-first experience consistent with existing VAs, rather than attempting to convert visual web content to auditory format. Likewise, our treatment of Wikipedia allows VERSE to focus on the article's main content rather than on other visual elements. This behaviour is consistent with the brief one or two sentence summaries narrated by existing virtual assistants, but allows convenient and efficient access to the entire article content.

4 EQUIPMENT AND LOGISTICS

We are proposing to run a demonstration that showcases the VERSE prototype system. The demo consists of a few smart speakers, smart watches, and laptops which we will supply. Our demonstration requires that internet connectivity – preferably wireless – be available. The demonstration should not require much physical space (e.g., one table), and can be scaled down as needed. Given the potential for a noisy environment, we will provide headsets with microphones, and will also modify the prototype to accept typed input as a contingency. We will also prepare a video showcasing the system [7]. Exhibit visitors will be free to conduct their own search and browsing sessions – we will not prescribe any particular flow or golden path through the system. Our hope is that the demo will inspire those researching web browser standards to consider new ways to support voice-based navigation of the web.

5 CONCLUSION

We have investigated the challenges that people who are blind experience when searching for information online using screen readers and voice assistants (e.g. Siri, Alexa). To identify the gaps and opportunities for improvement, we ran an online survey with 53 screen reader and voice assistant users. Based on the findings

[7] A standard VoiceOver gesture for "go back".

Table 2: Mapping of voice commands and corresponding gestures in VERSE.

Voice commands	Phone gestures	Watch gesture	Action
(Activation gesture)	Double tap with two fingers	Double tap with one finger	VERSE opens mic
"Cancel"	One tap with two fingers	One tap with one finger	Stop voice output
"Go to <source>"	Up/down swipe	Up/down swipe	Previous/next search source
"Next"/"Previous"	Right/left swipe	Right/left swipe or rotate crown	Next/previous element
"Tell me more"	Double tap with one finger	n/a	Continue reading the most recently mentioned answer / result

from the survey, we created VERSE – a system prototype for non-visual web search and browsing. Design of VERSE combines the advantages of both screen readers and voice assistants, and allows voice-based, as well as gesture-based, interaction.

REFERENCES

[1] Ali Abdolrahmani, Ravi Kuber, and Stacy Branham. 2018. "Siri Talks at You": An Empirical Investigation of Voice Activated Personal Assistant (VAPA) Usage by Individuals Who are Blind. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, 249–258.

[2] Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser. 2017. *Research methods in human-computer interaction*. Morgan Kaufmann.

[3] Ewa Luger and Abigail Sellen. 2016. Like having a really bad PA: the gulf between user expectation and experience of conversational agents. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 5286–5297.

[4] Rishabh Mehrotra, A Hassan Awadallah, Ahmed E Kholy, and Imed Zitouni. 2017. Hey Cortana! Exploring the use cases of a Desktop based Digital Assistant. In *SIGIR 1st International Workshop on Conversational Approaches to Information Retrieval (CAIR'17)*.

[5] Alisha Pradhan, Kanika Mehta, and Leah Findlater. 2018. Accessibility Came by Accident: Use of Voice-Controlled Intelligent Personal Assistants by People with Disabilities. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 459.

[6] Janice Y. Tsai, Tawfiq Ammari, Abraham Wallin, and Jofish Kaye. 2018. Alexa, play some music: Categorization of Alexa Commands. In *Voice-based Conversational UX Studies and Design Wokrshop at CHI*. ACM.

[7] Alexandra Vtyurina, Adam Fourney, Meredith Ringel Morris, Leah Findlater, and Ryen W. White. 2019. Demo video of VERSE. https://aka.ms/verse_demo_www2019

[8] Ryen W White and Resa A Roth. 2009. Exploratory search: Beyond the query-response paradigm. *Synthesis lectures on information concepts, retrieval, and services* 1, 1 (2009), 1–98.

Graph-based Interactive Data Federation System for Heterogeneous Data Retrieval and Analytics

Xuan-Son Vu
Department of Computing Science, Umeå University, Sweden
sonvx@cs.umu.se

Addi Ait-Mlouk
Department of Computing Science, Umeå University, Sweden
addia@cs.umu.se

Erik Elmroth
Department of Computing Science, Umeå University, Sweden
elmroth@cs.umu.se

Lili Jiang
Department of Computing Science, Umeå University, Sweden
lili.jiang@cs.umu.se

ABSTRACT

Given the increasing number of heterogeneous data stored in relational databases, file systems or cloud environment, it needs to be easily accessed and semantically connected for further data analytic. The potential of data federation is largely untapped, this paper presents an interactive data federation system (https://vimeo.com/319473546) by applying large-scale techniques including heterogeneous data federation, natural language processing, association rules and semantic web to perform data retrieval and analytics on social network data. The system first creates a Virtual Database (VDB) to virtually integrate data from multiple data sources. Next, a RDF generator is built to unify data, together with SPARQL queries, to support semantic data search over the processed text data by natural language processing (NLP). Association rule analysis is used to discover the patterns and recognize the most important co-occurrences of variables from multiple data sources. The system demonstrates how it facilitates interactive data analytic towards different application scenarios (e.g., sentiment analysis, privacy-concern analysis, community detection).

KEYWORDS

heterogeneous data federation, RDF, interactive data analysis

ACM Reference Format:
Xuan-Son Vu, Addi Ait-Mlouk, Erik Elmroth, and Lili Jiang. 2019. Graph-based Interactive Data Federation System for Heterogeneous Data Retrieval and Analytics. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314138

1 INTRODUCTION

Motivation. The ultra-connected world has been generating massive volumes of heterogeneous data stored in different data sources. And these data sources need to be normalized and interconnected to create a federated database that can be used to analyze, extract useful knowledge, and present it as a valid element for decision making. Semantic web techniques (e.g., RDF, SPARQL) have been widely used for data federation and linkage. However, the primary issue of the semantic web is insufficient integrated solution. To tackle this issue, we applied data federation, semantic web, and data mining technologies to develop this system, which can complement with each other. The system allows users to select data sources, interact with visualized graphs, and run customized queries across federated data to meet specific needs for data analytics.

Related Work. Related with data federation, there exist some popular enterprise data virtualization tools, such as IBM InfoSphere Federation Server (https://ibm.co/2qWQbom) and Oracle Data Service Integrator (https://goo.gl/6MKXkF). As well as some open source frameworks such as Teiid (teiid.jboss.org) which we used in this paper. Similar efforts in data federation have been seen from academia such as BioMart (www.ensembl.org/biomart) and Maelstrom (www.maelstrom-research.org). In the context of RDF and Linked Open Data, various works have been proposed in the literatures [2, 3]. In addition, several tools offering RDF and linked data visualization have been developed e.g., Sgvizler [6], LODWheel [7], IsaViz [1]. RDF-Gravity[2], etc. Also, many researchers used federated SPARQL queries to analyze and visualize linked data[4]. However, considering advanced data analytics across federated data is ignored. In this demo paper, we proposed RDF-supported data visualization framework over federated databases enriched with data mining (e.g., association rules) and NLP techniques (e.g., sentiment analysis). It can efficiently federate and analyze large heterogeneous data sources for general or specific analysis needs (e.g., community detection and the like).

Dataset. We processed one of the largest social science research databases, myPersonality corpus [3], which was collected from over 6 million volunteers on Facebook (FB). The data was anonymized and sampled to share with registered scholars around the world. In this paper, we used four of its data sources including *demographic dataset, personality dataset, political views*, and *FB status updates dataset*. One more *community detection dataset* was inferred based on these four. More information about these data sources is available in the menu of Data Source of the demo system.

Contributions. We build up a federation system, mapping the multiple heterogeneous, distributed, and autonomous data sources

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314138

[1] www.w3.org/2001/11/IsaViz
[2] https://www.salzburgresearch.at/publikation/rdf-gravity-3/
[3] https://www.psychometrics.cam.ac.uk/productsservices/mypersonality

into a unified federated database interface, where user can choose data sources in their area of interest; we provide data analytics including data exploration and search, which empowers users with ability to explore the data via data mining algorithms (i.e., association rules), and search by queries to lead advanced analytics. we implement interactive visualization, which allows users to plot the result in different formats like tables, graphs, scatter plots, and download the results. The system is scalable by adding other data sources, applying other data mining algorithms, and aiming at other data analytic scenarios.

2 SYSTEM DESCRIPTION

The system architecture is shown in Figure 1 with three goals including data federation, data linkage, and data analytics. On the server side, data from multiple data sources were preprocessed and connected through a VDB in the local deployed Teiid server. Different techniques are firstly applied to process raw data including generating RDF from raw data and indexing text-based data (*Process 1*). Association rule analysis is applied to support data exploration, such as exploring hidden patterns and co-occurrences of variables from multiple data sources in visualized ways (*Process 2*). After data exploration, users can go to data search and issue queries across RDF endpoints for general/specific data analytics, which are powered by semantic web techniques (*Process 3*). Based on the three processes mentioned above, on the end-user side, users are enabled to view metadata of data sources, explore the individual and federated data, and further construct simple/advanced queries of their (research) interests, to get data analytic results. Figure 2 presents the user interface.

Ahead of the above procedure, we apply NLP techniques [10] for raw data preprocessing (e.g., data normalization, standardization, non-utf8 characters removal etc.) and infer additional variables such as sentiment analysis and privacy concern based on the given variables. More information about these variables can be found in the menu of *Data Source* in the system.

2.1 Data Federation and Data Linkage

Data Federation. Our data federation component was built based on an open source framework Teiid, which is a data virtualization system that allows applications to use data from multiple, heterogeneous data stores. We created virtual database (VDB) for data federation, where data is accessed and virtually integrated in real-time across distributed data sources without copying or otherwise moving data from its system of record.

Data Linkage. After data federation, we applied semantic web techniques (e.g., RDF, SPARQL) for data linkage, which is a method for publishing structured data using vocabularies like schema.org that can be connected together and interpreted by machines. We created our own RDF generator, which standardizes the raw data to a unified RDF format and stored in RDF database as shown in Figure 1, so they can be read automatically by computers and enable data from different sources to be connected and queried. Afterwards, we apply the inverted indexing schema from ElasticSearch (www.elastic.co) (ES) for data indexing, which is an open source, distributable, and highly scalable search engine. The indexed results are stored to ES database. These two steps are critical to facilitate efficient data analytics in the following.

2.2 Data Analytics

Data Exploration. After getting to understand the data sources, users are guided to apply data mining techniques to explore patterns and correlations over data variables. We take association rules technique as an example to show how data mining techniques discover the relationship between variables in federated data. Association rules technique was initiated by Agrawal [1] to analyze transactional databases. It usually defined as an implication of the form $A \rightarrow B$ such as $A, B \subset I$ and $A \cap B = \emptyset$. Every rule is composed of two different sets of items A and B, where A is called antecedent or left-hand-side (LHS) and B called consequent or right-hand-side (RHS). In order to select interesting rules from the set of all possible rules, constraints on various measures of significance and interest are used. The best-known constraints are minimum thresholds on support and confidence. The support is defined as the proportion of transactions in the database which contain the items A, and the confidence defines how frequently items in B appear in a transaction that contains A. In this system, we apply Apriori algorithm [1] to extract association rules among variables over the federated data. For example, given the FB users with specific variables like age (e.g., 31-40), gender (e.g, female), and relation status (e.g., married), the system may present association rule graph with quantified scores of the fourth variable "personality", which indicates the prone of specific personality (e.g, high-score neuroticism, low-score agreeableness etc.) of these types of users.

Data Search. After exploring the federated data sources, users come to the page of *Data Search* and issue queries over the indexed data on ElasticSearch. As shown in Figure 3, for demonstration purpose, we created two dashboards of *Case-Study 1&2* and *Case-Study 3* for users. Each dashboard provides a search box as well as filter configurations. We have the pure long-text based variable (i.e., FB status update) to support text-based query, and additionally users will add filters to setup constraints for other variables to customize their own query. The third dashboard supports general advanced search (e.g., Boolean search).

3 DEMONSTRATION OF THE SYSTEM

For demonstration, let's say Alice, a psychologist researcher, who wants to research the correlation of personality and stresses in people's life based on social network behaviours. Alice is enabled to apply given data mining algorithms (i.e., association rules) to explore the hidden patterns across selected variables of interests. Next, Alice will further extrapolate the returned results in *Data Search* to have more detailed information of those people. These two steps will be described in case-study 1 and case-study 2 accordingly.

3.1 Case-study 1: association rules based sentiment analysis

In this case study, Alice will select her variables of interests including sentiment_score_subjectivity (sentiS), cNEU (neuroticism), cCON (conscientiousness), cAGR (agreeableness) [8] in the graph

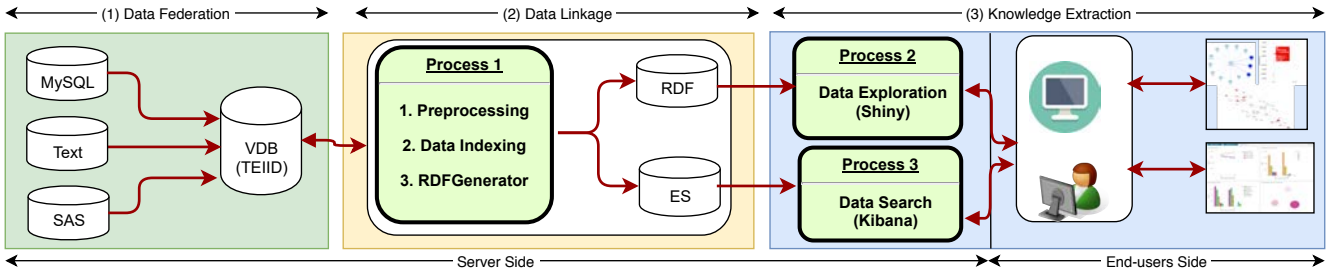


Figure 1: Overview of system architecture: (1) Data Federation, (2) Data Linkage, and (3) Data Analytics.

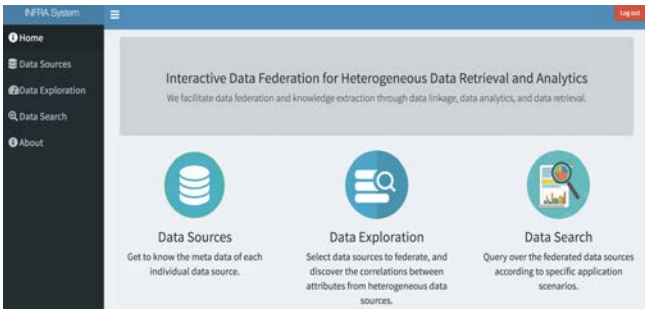


Figure 2: User interface of the system

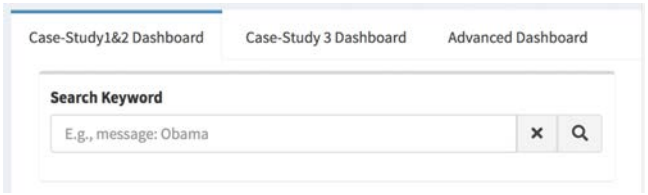


Figure 3: *Data Search* interface of the system

visualization panel (1) to be inferred through association rules. We applied the Apriori algorithm [1] to extract frequent variables, which satisfies the minimum support requirement specified by the user, and then generate association rules based on the user-specified confidence threshold. As shown in Figure 4, in *Data Exploration*, the four variables were chosen to run the inference. Parameter configuration panel (2) was displayed on right side including selecting data source, setting support and confidence thresholds for association rules. Above the configuration panel, a component of "User Guide–Help" gives user guidance of exploring this page. Afterwards, Alice moves to the bottom panel (3) and clicks "association rule graph" tab to see the results like in Figure 5. In this figure, the rectangles represent variables and the circles represent association rules. Larger size of circle imply more data records matching the rule, while the darker circle represent more importance of the rule.

Regarding the research question, Alice found a hidden pattern between three variables regarding neurotic people (cNEU), which is a personality trait that reflects one's ability to deal with emotion states, such as stress and anxiety. The pattern suggests that people who are not agreeableness (N.AGR) and do not have strong purposes in the way they say on FB status (sentiS=0) then most likely they are neurotic people. This hidden pattern however might not be sufficiently significant to draw such a conclusion to define neurotic people. Therefore, Alice will verify this discovered pattern in *Data Search* to confirm this pattern across data sources. The case-study 2 will explain in more detail how Alice confirms the explored pattern.

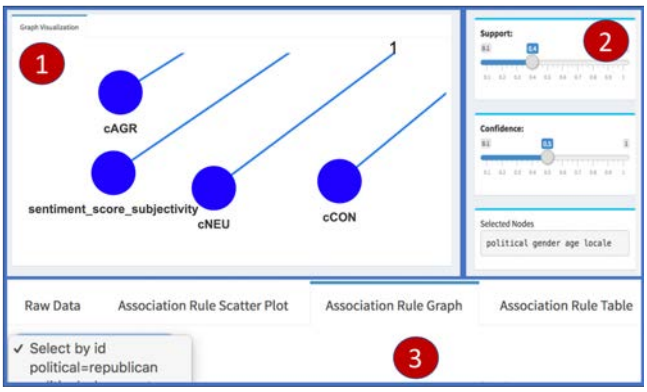


Figure 4: Selected variables on *Data Exploration*

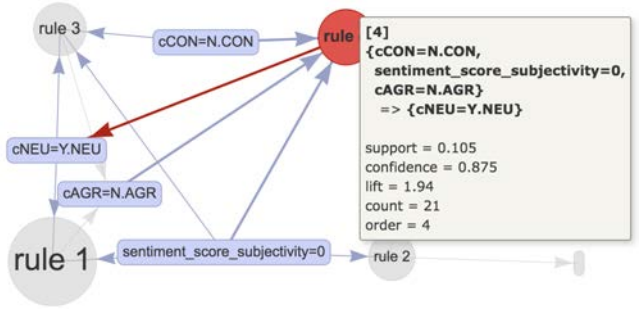


Figure 5: Case study of association rule based sentiment analysis on selected variables.

3.2 Case-study 2: personality and sentiment analysis

In this case study, we show how *Data Search* can be used to confirm the hidden pattern that Alice found out using the association rules. As shown in Figure 6, in *Data Search*, after issuing the query "sentiment_score_subjectivity:0 AND cagr:N.AGR AND ccon:N.CON" (Q1), Alice got a line chart showing the fraction between different personality traits and age group. It clearly shows

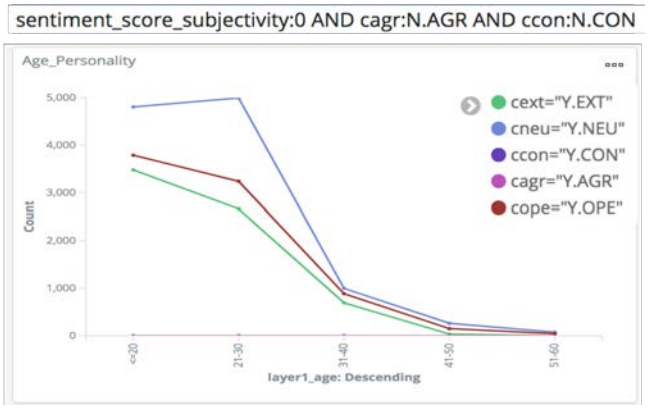


Figure 6: Case study of using *Data Search* to reconfirm the results from *Data Exploration*

that the neurotic group of people is the major group among five personality traits. Furthermore, Alice can find more information related to this group of neurotic people. For instance, a graph called "Age_Political_Views" shows that, mainly *neurotic people* have political views of "doesn't care" and "democratic". This extrapolation is a powerful feature that *Data Search* can provide for data analysts. Due to the limited space, we cannot show all analytic graphs that Alice can see based on her query. Thus, we specially create a dashboard called *Case-Study 1&2 Dashboard* for this demonstration. Anyhow, Alice can also issue the query Q1 to "Advanced Dashboard" and find more related information to the neurotic group of people to confirm the hidden pattern.

Remarks: In order to have a finally verified answer on the research question, Alice must also run a number of statistical analysis to reconfirm the statistical signification of the pattern. Nevertheless, Alice's job now already becomes much easier. In future, we plan to add statistical analysis to the system as well.

3.3 Case-study 3: privacy-concern analysis

This case study shows how privacy-concern analysis is done through community detection and data analysis based on different variables (i.e., age, gender, personality, FB status). It is clearly that privacy-concerns on social network is an important factor to protect people's privacy on personal data [9]. Since there was not any user defined privacy-concerns information in the given dataset, we apply an approach from [10] to infer people's privacy based on their FB status and personality. As shown in Figure 7, in *Data Search*, the *Case-Study 3 Dashboard* displays four different charts to characterize privacy-concerns of people across data sources and their demographic information. The chart (1) shows the personality difference based on different age groups, chart (2) shows the number of communities based on different privacy-concerns, chart (3) shows the correlation between privacy-concerns and age, and chart (4) shows the association between gender, privacy, and sentiment embodied in their FB status. Especially, users can simply search based on keywords. For instance, if a user issues a query "Obama" and sees people's sentiment polarity (e.g., positive, neutral, negative) when they talked about Obama. It is found that 56.9% of users who mentioned the word "Obama" are females, in which 19.95% of them have negative sentiment when they mentioned about Obama. After digging deeper into the raw data, we realized that FB users complained about Obama mainly because of high unemployment rate in USA during his presidency, such as one post saying "because of obama there are no more jobs left in america".

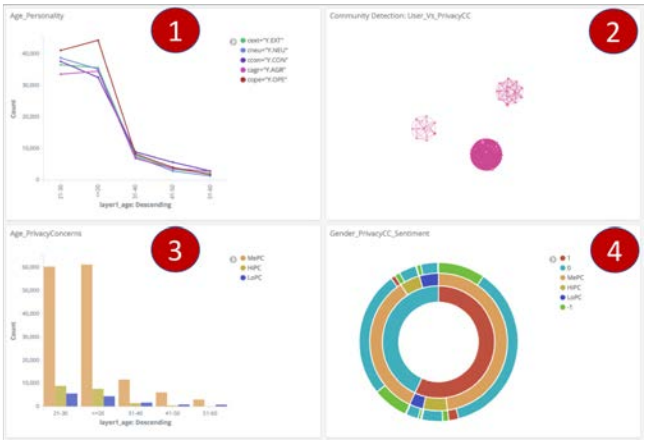


Figure 7: Case study of privacy-concern analysis

3.4 Implementation and demo environment

The system was implemented in Java and R. Data federation mechanism was built on Teiid. We developed the interactive and user-friendly interfaces using R shiny (shiny.rstudio.com/) and ArulesViz [5]. We built our own RDFGenerator to generate RDF files from VDB (Virtual Data Bases), and stored them in Apache Jenna Fuseki (jena.apache.org). To support *Data Search*, we employed ElasticSearch and Kibana for high performance data retrieval. Thanks to the plugin based architecture, Kibana can be easily extended to suit particular needs. For instance, according to the different data analytic requirements from different end-users, we support extended customized dashboard for each user need.

Brief information of the web-based prototype:

- System video URL: https://vimeo.com/319473546
- Multiple Operating System(s): Linux, macOS, Windows
- State-of-the-art browsers e.g., Chrome (recommended)
- Memory: from 1GB

4 CONCLUSION

In this paper, we have proposed a graph-based data federation system for heterogeneous data analytics. It is an open-source SPARQL query builder and result-set visualizer for heterogeneous data sources (i.e., myPersonality corpus) which allows users to easily construct and explore data over heterogeneous data sources. The system is scalable by easily adding new heterogeneous data sources, and customizing data representation and analytics by users, especially researchers based on their own interests. The future work will be adding more text-based data sources, extending more data mining algorithms options for data exploration to accompany data search.

ACKNOWLEDGEMENT

This research is funded by Umeå University in Sweden on federated database research. The authors also thank Michal Kosinski, David Stillwell and the myPersonality project for data sharing.

REFERENCES

[1] Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB '94)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 487–499.

[2] Miika Alonen, Tomi Kauppinen, Osma Suominen, and Eero Hyvönen. 2013. Exploring the Linked University Data with Visualization Tools. In *The Semantic Web: ESWC 2013 Satellite Events*, Philipp Cimiano, Miriam Fernández, Vanessa Lopez, Stefan Schlobach, and Johanna Völker (Eds.). Springer Berlin Heidelberg, 204–208.

[3] Josep Maria Brunetti, Sören Auer, and Roberto García. 2012. The Linked Data Visualization Model. In *Proceedings of the 2012th International Conference on Posters & Demonstrations Track - Volume 914 (ISWC-PD'12)*. Germany, 5–8.

[4] Marija Djokic-Petrovic, Vladimir Cvjetkovic, Jeremy Yang, Marko Zivanovic, and David J. Wild. 2017. PIBAS FedSPARQL: a web-based platform for integration and exploration of bioinformatics datasets. *Journal of Biomedical Semantics* 8, 1 (20 Sep 2017), 42.

[5] Michael Hahsler and Radoslaw Karpienko. 2017. Visualizing association rules in hierarchical groups. *Journal of Business Economics* 87, 3 (01 Apr 2017), 317–335.

[6] Martin G. Skjæveland. 2015. Sgvizler: A JavaScript Wrapper for Easy Visualization of SPARQL Result Sets. In *The Semantic Web: ESWC 2012 Satellite Events*, Elena Simperl, Barry Norton, Dunja Mladenic, Emanuele Della Valle, Irini Fundulaki, Alexandre Passant, and Raphaël Troncy (Eds.). Springer Berlin, 361–365.

[7] Magnus Stuhr, Dumitru Roman, and David Norheim. 2010. LODWheel - JavaScript-based Visualization of RDF Data -. In *Proceedings of the Second International Conference on Consuming Linked Data - Volume 782 (COLD'11)*. Germany, 73–84.

[8] Xuan-Son Vu, Lucie Flekova, Lili Jiang, and Iryna Gurevych. 2018. Lexical-semantic resources: yet powerful resources for automatic personality classification. In *Proceedings of the 9th Global WordNet Conference*. 173–182.

[9] Xuan-Son Vu and Lili Jiang. 2018. Self-adaptive Privacy Concern Detection for User-generated Content. In *Proceedings of the 19th International Conference on Computational Linguistics and Intelligent Text Processing, Long papers, p., March 2018.*

[10] Xuan-Son Vu, Lili Jiang, Anders Brändström, and Erik Elmroth. 2017. Personality-based Knowledge Extraction for Privacy-preserving Data Analysis. In *Proceedings of the Knowledge Capture Conference (K-CAP 2017)*. ACM, USA, Article 45, 4 pages.

XFake: Explainable Fake News Detector with Visualizations

Fan Yang*
Department of Computer Science and Engineering, Texas A&M University
nacoyang@tamu.edu

Shiva K. Pentyala*
Department of Computer Science and Engineering, Texas A&M University
pk123@tamu.edu

Sina Mohseni*
Department of Computer Science and Engineering, Texas A&M University
sina.mohseni@tamu.edu

Mengnan Du*
Department of Computer Science and Engineering, Texas A&M University
dumengnan@tamu.edu

Hao Yuan*
School of Electrical Engineering and Computer Science, Washington State University
dumengnan@tamu.edu

Rhema Linder
Department of Computer Science and Engineering, Texas A&M University
rhema@tamu.edu

Eric D. Ragan
Department of Computer & Information Science & Engineering, University of Florida
eragan@ufl.edu

Shuiwang Ji
Department of Computer Science and Engineering, Texas A&M University
sji@tamu.edu

Xia (Ben) Hu
Department of Computer Science and Engineering, Texas A&M University
xiahu@tamu.edu

ABSTRACT

In this demo paper, we present the XFake system, an explainable fake news detector that assists end-users to identify news credibility. To effectively detect and interpret the fakeness of news items, we jointly consider both attributes (e.g., speaker) and statements. Specifically, *MIMIC*, *ATTN* and *PERT* frameworks are designed, where MIMIC is built for attribute analysis, ATTN is for statement semantic analysis and PERT is for statement linguistic analysis. Beyond the explanations extracted from the designed frameworks, relevant supporting examples as well as visualization are further provided to facilitate the interpretation. Our implemented system is demonstrated on a real-world dataset crawled from *PolitiFact*[1], where thousands of verified political news have been collected.

CCS CONCEPTS

• **Human-centered computing** → **Natural language interfaces**; *Graphical user interfaces*; *Web-based interaction.*

KEYWORDS

Fake news detection, explainable models, system visualization.

ACM Reference Format:
Fan Yang*, Shiva K. Pentyala*, Sina Mohseni*, Mengnan Du*, Hao Yuan*, Rhema Linder, Eric D. Ragan, Shuiwang Ji, and Xia (Ben) Hu. 2019. XFake: Explainable Fake News Detector with Visualizations. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314119

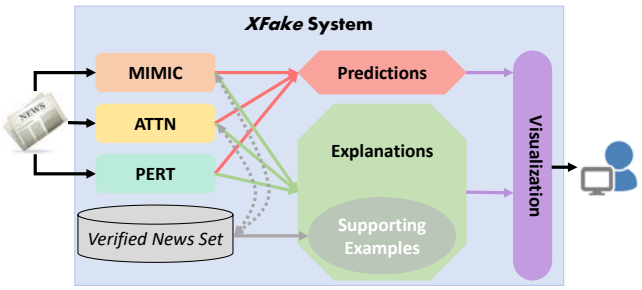


Figure 1: The architecture of XFake system.

1 INTRODUCTION

With the prevalence of web applications, it has become much easier for the public to get access to various news items through different channels, such as news websites and social media. However, this kind of convenience also brings about the wide spread of fake news at the same time [10]. Fake news is detrimental to both individuals and society, and it is usually generated as hoaxes to mislead public through false or biased information. In 2016, the term "Fake News" was even shown as the word of the year by Macquarie Dictionary.

Though increasingly relevant, effective fake news detection is still considered to be challenging due to the following two aspects. First, fakeness of news could come from diversified perspectives [7], which is beyond the boundaries of traditional textual analysis. For example, a fake news item may result from its contexts or speaker, and it may not directly relate to its contents. The source of fakeness is highly dependent on each specific instance. Second, fakeness detection results need further explanation, which is important and necessary for users' final decisions [2]. The explanation could provide evidences on model predictions and further help users better understand why certain news items are classified as fake.

[1] https://www.politifact.com/

* Those authors contributed equally in developing the system.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314119

Considering these two challenges, we design and implement an explainable system, named **XFake**, on fake news detection task. Particularly, three different frameworks (i.e., *MIMIC*, *ATTN* and *PERT*) are designed to analyze news items from different perspectives. With MIMIC, we aim to judge each news item from its attribute information, which may include news context, speaker and etc. For ATTN, our goal is to investigate each news statement through its semantic meaning. By employing PERT, we aim to study the linguistic features of news statements for detection. All three designed frameworks are self-explainable, where relevant explanations could be extracted for interpreting detection results. Besides, we also provide supporting examples for each news instance by corresponding matching algorithm, and visualize detection results for user-friendly interaction. Both supporting examples and visualizations further assist users to understand why the system make a certain prediction on a given news. Overall, XFake system is capable of not only giving prediction scores on news fakeness, but also providing relevant explanations as prediction evidences. With the aid of XFake, users would be convenient to identify the news credibility.

To best of our knowledge, XFake is the first explainable fake news detector over multiple perspectives. Corresponding system architecture and demonstration details will be respectively introduced in the following section 2 and section 3.

2 XFAKE ARCHITECTURE

The architecture of XFake is illustrated in Figure 1. The system input is the targeting news with attributes, and the output contains both prediction and explanation results. The whole system is trained on a verified news set. All components will be introduced as below.

MIMIC Framework. MIMIC is designed to analyze the news attributes. Enlightened by [9], we employ a deep neural network as the teacher, and use a shallow model as the student to emulate the teacher's performance for better explainability. Basically, the overall idea is to mimic the performance of neural networks with tree ensemble models, so that we can keep the good performance from neural networks and good explainability from tree ensemble models simultaneously. The structure of MIMIC is illustrated in Figure 2. Specifically, we use GloVe [6] as our word embedding scheme in MIMIC. After obtaining the soft labels from the teacher model, we further use them to train a student model consisting of 80 decision trees. By calculating the node importance of each attribute in the ensemble trees, we could get the significance of each attribute in the input news instance. Further, through the constructed trees and importance scores, users could know what happened inside, which provides an overall transparency towards the model itself.

ATTN Framework. ATTN is designed to analyze news statement simply from semantic perspective. To build an explainable model for semantic analysis with good performance, we employ several techniques, including pre-trained word embedding, convolutional neural network [4], and self-attention mechanism [8]. Self-attention is used because it can capture global relationships between different words efficiently. In addition, the weight matrix generated in attention mechanism is input dependent, which helps provide instance-level explanation. By applying different kernel sizes for the convolutional network, we can get explanations based on one-gram, two-gram or three-gram analysis. The overall illustration

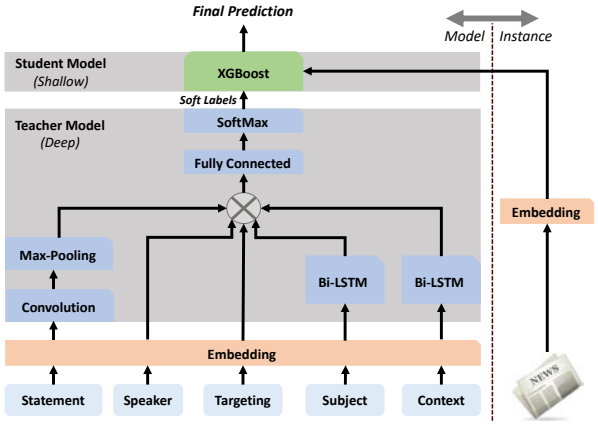


Figure 2: The structure of MIMIC framework.

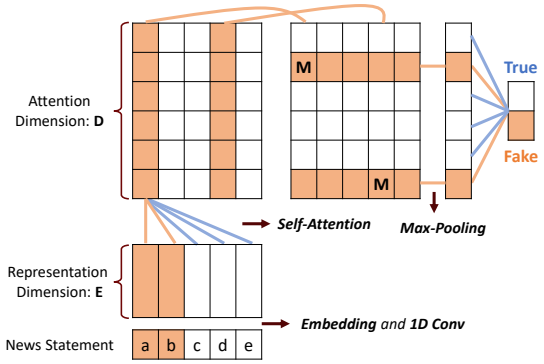


Figure 3: The illustration of ATTN framework.

of ATTN is given by Figure 3. In ATTN, we employ word2vec [5] as the pre-trained embeddings and each vector representation has a dimension of 300 (i.e. $E = 300$). Each spatial location learns a 512-dimensional vector representation for each word (i.e. $D = 512$).

PERT Framework. PERT is designed for news statement analysis as well, but it is uniquely from a linguistic perspective. For effective analysis, we employ eight linguistic features, including *Adjective ratio*, *Noun ratio*, *Verb ratio*, *Propn* (such as Google, etc.) *ratio*, *Sentiment score*, *Normalized text length*, *Whether contains the mark "?"*, *Whether contains the mark "!"* [3]. For each news item in the training set, we extract its linguistic features and train an XGBoost [1] classifier using these features. The trained XGBoost is then used to make predictions for new items. Further, we use perturbation-based method to provide explanations. The idea is that feature importance can be measured by observing how much the score (e.g., accuracy) decreases when the feature is not available. To this end, we can remove a feature from the dataset, and then re-train the classifier for score checking. Since re-training may be computationally expensive, we replace the feature value with random noise, drawn from the same distribution as the original one, instead of removing. The computed prediction difference is utilized as the significance score for the corresponding feature.

Prediction & Explanation. XFake outputs the prediction of news as a unified score, indicating its probability to be fake. The prediction result is the ensemble output from MIMIC, ATTN and PERT,

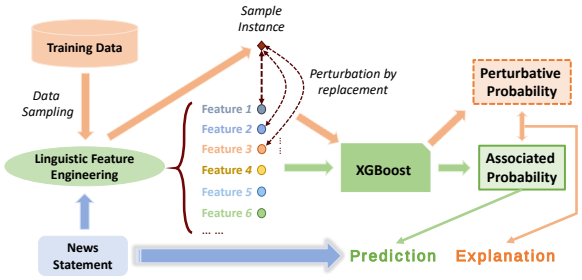


Figure 4: The mechanism of PERT framework.

where different results are combined together in a weighted sum manner. The weights for combination are tuned on a validation set, according to the performance of different frameworks. The higher the prediction score, the higher the probability for fake news. As for the explanation from XFake, it is mainly extracted from three different perspectives enabled by MIMIC, ATTN and PERT. MIMIC could provide explanation by key components of news, ATTN is able to explain predictions by word/phrase attribution, and PERT generates explanation by linguistic features. Beyond this, XFake also explains predictions from data perspective with supporting examples. These examples are generated from both MIMIC and ATTN by retrieving training samples given corresponding explanations (i.e. key attributes or important words/phrases). Showing these training samples which are highly similar to the input news would be helpful for users to understand the working patterns of XFake. Besides, we also assign different similarity scores for different supporting examples based on the matching extent between the input and support. Supporting examples with higher scores would be considered more informative in delivering explanation.

Visualization. To further facilitate the explanability of XFake, we visualize the outputs of both prediction and explanation by *D3 JavaScript*. Visualization mainly lies in three aspects. First, for numerical values, such as prediction score and attribute significance, we visualize them by histograms, which straightforwardly indicate the results and influences. Second, to enhance the explanability for word/phrase attribution, we visualize the outputs by highlighting important words/phrases with heatmaps, where the darkness positively relates to the importance of word/phrase. Third, for better model explanability, the ensemble trees are visualized with interactive diagrams which are capable of showing both overall structure and specific activated paths (depending on the input). Through those visualization schemes, users could have a better sense towards XFake about why certain news are classified as fake or true.

3 SYSTEM DEMONSTRATION

XFake is implemented by *Python* and deployed in *FLASK* with an *HTML* front end. To demonstrate the system, we will first introduce the data source and then go through a specific use case of XFake.

3.1 Data Source

The news data, which is used to train, tune and evaluate XFake, comes from a political fact-checking website, named PolitiFact. It is a Pulitzer prize-winning website containing tons of political news with diversified categories. The reasons why we employ this data source are in three folds. First, PolitiFact provides professional justification and fine-grained labels for all news items, where the core principles in independence, transparency and fairness guarantee its high credibility among the public. Second, the news collected by PolitiFact have various attribute information, which directly meets our data requirement for analysis. Third, raw data in PolitiFact can be effectively crawled through an available API [2], and it is convenient to obtain the customized dataset for system implementation.

When processing the data, we only keep the attributes which are highly related to news fakeness. Specifically, the maintained attributes for XFake include *Subject*, *Context*, *Speaker*, *Targeting* and *Statement*, although some news may not have all five attributes. Besides, to effectively measure the news fakeness and train the system, we transform the original multi-class data to binary data where each news item is labelled as either *True* or *False*[3]. Particularly, labels with *Mostly True*, *Half True*, *No Flip*, *Half Flip* are switched to label *True*, and labels with *Mostly False*, *Pants On Fire*, *Full Flop* are switched to label *False*. Furthermore, as for data statistics, there are *5104* news crawled in total, which are evenly distributed. The training, validation, test set of XFake respectively contain *4083*, *510*, *511* items. The validation set is used to test the performance of different frameworks and further tune the hyper-parameters, while the test set is employed to conduct the overall evaluation.

3.2 Use Cases

With the testing on validation set, we obtain *67.1%*, *67.3%*, *53.2%* accuracy respectively for MIMIC, ATTN and PERT framework. By normalizing weights according to the performance, we fix the coefficients *0.36*, *0.36*, *0.28* in weighted sum correspondingly for all use cases. Considering the fake news identification scenario, we show a specific case demonstration of XFake as follows.

Illustrated by Figure 5, users start by inputting news into the text boxes. In XFake, we provide a button "Random News" to help users explore the system, which is used to retrieve random items from our test set. Similarly, buttons "Fake Examples" and "True Examples" are also provided to help quickly access some representative fake and true news. After clicking "Submit", users would obtain all the outputs including both prediction and explanation in a few seconds. As for the prediction of the example in Figure 5, we get the score 0.76, which means that the given news has the probability 76% to be fake. Regarding to the explanation, we can obtain it from both attribute and statement analysis. Aided by MIMIC, for this example, we know that "Statement" plays the most important role compared with others. Through ATTN, we can easily check those highlighted words, such as "invited" and "Russia", with different darkness, which would also show the contribution scores when mouse is hovering around. PERT gives users a clear view about which linguistic features contribute to fake and which to true. In the example, we observe that features "Propn Ratio", "Adjective Ratio" and "Noun Ratio" mainly contribute this news to be fake.

XFake further provides supporting examples and visualized trees for users to better understand the system. As shown in Figure 6, we give two supporting news for instance, where one is retrieved based on the important attributes (Context & Statement) from MIMIC

[2] http://static.politifact.com.s3.amazonaws.com/api/v2apidoc.html
[3] A news item labelled as False is regarded as the fake news.

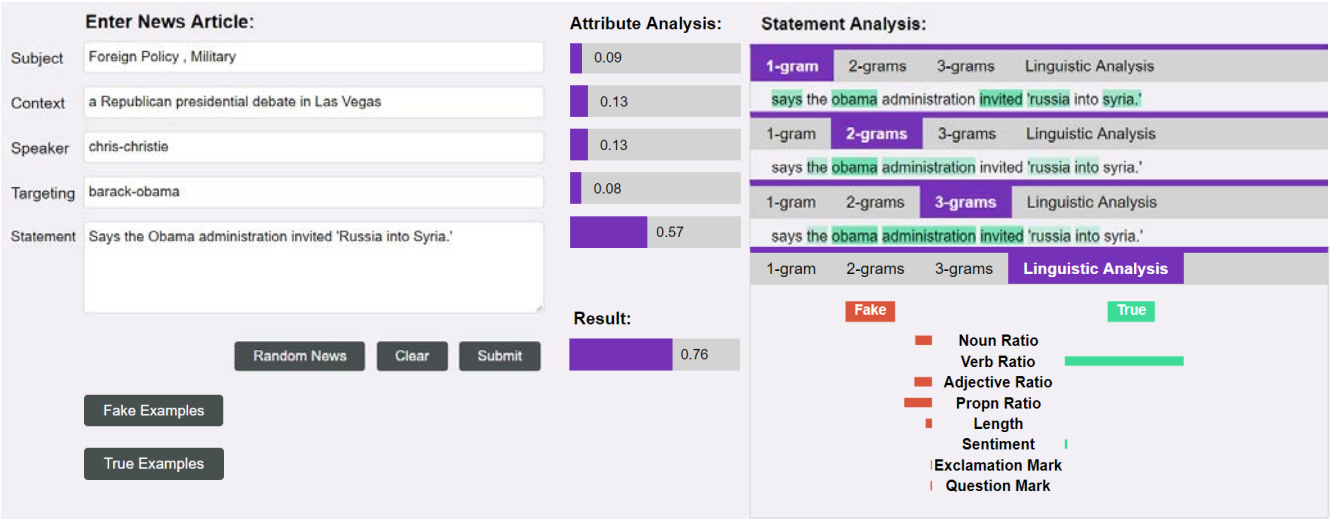


Figure 5: Prediction and explanation from XFake.

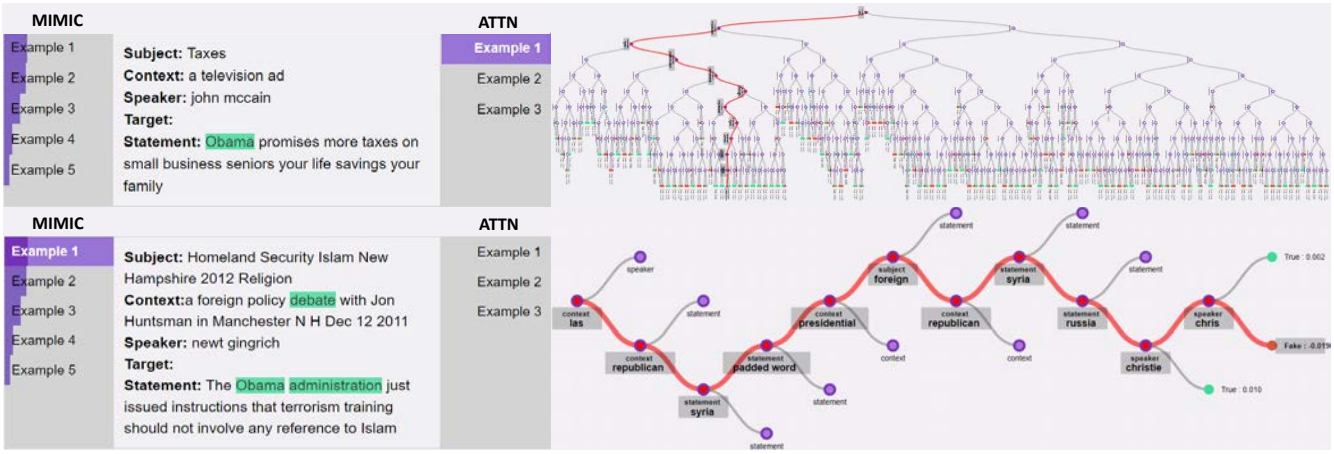


Figure 6: Supporting examples and ensemble trees from XFake.

and the other is obtained by matching significant word ("Obama") from ATTN. For the support extraction with MIMIC, we also attach a similarity score, indicating how much attribute information it overlaps with the input one. Besides, 80 decision trees are visualized with interactive diagrams and highlight the activated path of each tree regarding to the input. In Figure 6, we only show one decision tree for example. We can see that each decision tree can be expanded or compressed flexibly, which allows users to track the decision process closely. Given a certain news, each decision tree has only one activated path, corresponding to one specific decision attached at the end of the path with relevant contribution score. Those visualized trees largely enhance the model explainability of XFake.

To demonstrate the effectiveness of XFake in real cases, we conduct relevant human evaluations by Amazon Mechanical Turk (AMT), with *147* valid testing users in total covering diversified gender, age and education level. The involved user tasks include *Fact Check* and *Prediction Guess*, where the first one is to test the usefulness of the generated explanation and the second one is to indicate the users' understanding towards the system. The evaluation metrics are accuracy and time for user prediction. The human study shows a clear trade-off between the speed and accuracy regarding to generated explanations. On one hand, explanation does help users better understand and predict system behavior. On the other hand, explanation would take users more time to review and interpret detection results for benefits. Due to space constraints, we limit the details of the human evaluation study in this paper.

4 CONCLUSIONS AND FUTURE WORK

We design and implement an explainable fake news detector, named XFake, to help end-users identify the news credibility. XFake is capable of jointly analyzing news items from multiple perspectives, and can effectively provide explanation through attribute importance, word/phrase significance, linguistic features as well as relevant supporting examples. Corresponding visualizations further enhance the explanability of XFake and facilitate the decision making for users. In future, applying XFake to multi-source (i.e., not limited to

the PolitiFact website) or multi-modal (i.e., not limited to textual form) news data could be a promising extension for the system.

ACKNOWLEDGMENTS

The authors thank the anonymous reviewers for their helpful comments and the funding agencies for their generous supports. The work is in part supported by DARPA grant N66001-17-2-4031. The views and conclusions in this paper are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD*. ACM, 785–794.

[2] David Gunning. 2017. Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web* (2017).

[3] Wafa Herzallah, Hossam Faris, and Omar Adwan. 2018. Feature engineering for detecting spammers on Twitter: Modelling and analysis. *Journal of Information Science* 44, 2 (2018), 230–247.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[6] Jeffrey Pennington and etc. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on EMNLP*. 1532–1543.

[7] Victoria L Rubin, Yimin Chen, and Niall J Conroy. 2015. Deception detection for news: three types of fakes. In *Proceedings of the 78th ASIS&T Annual Meeting*. 83.

[8] Ashish Vaswani, Noam Shazeer, and etc. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[9] William Yang Wang. 2017. " liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648* (2017).

[10] Xinyi Zhou and Reza Zafarani. 2018. Fake News: A Survey of Research, Detection Methods, and Opportunities. *arXiv preprint arXiv:1812.00315* (2018).

GraviTIE: Exploratory Analysis of Large-Scale Heterogeneous Image Collections

Sean T. Yang
University of Washington
Seattle, Washington
tyyang38@uw.edu

Luke Rodriguez
University of Washington
Seattle, Washington
rodriglr@uw.edu

Jevin D. West
University of Washington
Seattle, Washington
jevinw@uw.edu

Bill Howe
University of Washington
Seattle, Washington
billhowe@cs.washington.edu

ABSTRACT

We present GraviTIE (Global Representation and Visualization of Text and Image Embeddings, pronounced "gravity"), an interactive visualization system for large-scale image datasets. GraviTIE operates on datasets consisting of images equipped with unstructured and semi-structured text, relying on multi-modal unsupervised learning methods to produce an interactive *similarity map*. Users interact with the similarity map through pan and zoom operations, as well as keyword-oriented queries. GraviTIE makes no assumptions about the form, scale, or content of the data, allowing it to be used for exploratory analysis, assessment of unsupervised learning methods, data curation and quality control, data profiling, and other purposes where flexibility and scalability are paramount. We demonstrate GraviTIE on three real datasets: 500k images from the Russian misinformation dataset from Twitter, 2 million art images, and 5 million scientific figures. A screencast video is available at https://vimeo.com/310511187.

KEYWORDS

GraviTIE; MultiDEC; Visualizing Large-Scale Image Collections

ACM Reference Format:
Sean T. Yang, Luke Rodriguez, Jevin D. West, and Bill Howe. 2019. GraviTIE: Exploratory Analysis of Large-Scale Heterogeneous Image Collections . In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314142



Figure 1: Main interface of GraviTIE. GraviTIE (Global Representation and Visualization of Text and Image Embedding) is an interactive web visualization application and discovery engine for large image-text datasets.

1 INTRODUCTION

Countless images exist on the web. These include photographs, memes, art images, scientific figures, and more. Analysis of these image datasets increasingly requires the application of large-scale machine learning methods to answer even relatively straightforward questions. For example, the dataset recently released by Twitter[1] contains rich information about Russian influence, but even conceptually simple questions such as "Do images posted from Russian accounts tend to differ from those posted by other accounts?" require a combination of unsupervised learning (to determine image similarity), NLP methods (to contextualize the image), and structured queries (to compare images by source and year).

These analysis tasks have several defining characteristics. First, tasks increasingly require some form of unsupervised learning. Ground truth labels or other obvious sources of supervision are rarely available. In these situations, interactive, qualitative exploration of the results is the only option. Second, the data is *multimodal*: each record consists of some combination of an image, free text, and a set of structured attributes. Each of these elements may be of arbitrary size, or altogether missing. Third, the datasets are large, making direct-browsing and main-memory approaches infeasible. For example, the Twitter dataset contains 9 million tweets with over 543k images, the Artstor dataset contains more than 2 million high-quality images of artwork along with text descriptions, and the Viziometrics [8] dataset consists of over 9 million figures from the scientific literature, along with their captions.

[1] https://about.twitter.com/en_us/values/elections-integrity.html#data

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314142

In this demonstration, we present GraviTIE (**G**lobal **R**epresentation **a**nd **Vi**sualization of **T**ext and **I**mage **E**mbeddings, pronounced "gravity"), an interactive visualization system for large-scale image-text datasets. Fig. 1 shows the interface of GraviTIE.

For pre-processing, GraviTIE takes as input a large set of image-text pairs. It learns an embedding for the pair with clustering objective by running MultiDEC[14] on the text and image, such that the two models tend to produce the same distribution. The properties of MultiDEC are tightly coupled to the design of the interface. In particular, this model 1) accommodates multi-modal heterogeneous data while 2) maintaining reasonable performance at scale. The learned embedding is then dimension-reduced to two dimensions for display purposes.

The GraviTIE interface features a large similarity map to display thousands of nodes, where each of the node represents an image positioned by a multi-modal vector embedding. Similar images are clustered together on this similarity map which helps the users efficiently navigate a large number of images. Users can refine the current view using a combination of free-text queries (with appropriate operators “must include”, “might contain” or “exclude”) and structured query filters. Only a sample of the images is displayed to ensure performance at scale and to afford hover interactions with individual images; displaying all images would lead to extremely dense regions with significant occlusion. Users can pan, zoom, filter, and search iteratively, gradually refining the set of displayed images and inspecting the images in each cluster via hover interactions at any time. Each view is associated with a unique url (i.e. RESTful), making the system stateless and programmable, allowing refinements to be saved and shared with collaborators. The data displayed in each view can be directly downloaded to provide custom curated subsets for specific tasks. For example, an art historian can construct a dataset of impressionist paintings by selecting particular clusters and filtering for a particular time period. The RESTful API design also affords additional downstream applications to be developed using the search and download functions.

GraviTIE is designed to be adaptable to a number of applications across multiple modalities and communities. In its current form, art historians are using GraviTIE to explore influence patterns in art, computer scientists are using it to qualitatively assess unsupervised learning methods, and social scientists are using it to understand the behavior of Russian actors on Twitter.

2 RELATED WORK

Analysis and visualization of large scale images have been receiving great attention from researchers. Hochman et al. [7] explore over 2 million photos on Instagram by sorting the images in terms of their properties (e.g., hue or create time) and showing the complete image sets of different cities to reveal the local culture and social patterns. The T-SNE Map [2], one of Google Arts & Culture experiments by Diagne et al., is a 3D interactive platform created by computing the visual similarity of art works and grouping the art works with t-SNE algorithm. PixPlot[3] extends the idea by applying UMAP [10] algorithm which can scale large data set and cluster million of data points and allows users to implement custom datasets. These applications provide a simple platform to explore large scale image sets, but do not support query and do not exploit text or structured metadata to assist search and curation. Wang et al. [13] introduced iMap, a treemap-based visualization for navigating image search and clustering results, but the number of images displayed is limited. iGraph [5] constructs a similarity graph by computing the affinity between images, the relationship between texts, and the connection between images and texts, which results in a system that requires enormous amount of computation.

[2]https://artsexperiments.withgoogle.com/tsnemap/
[3]http://dhlab.yale.edu/projects/pixplot/

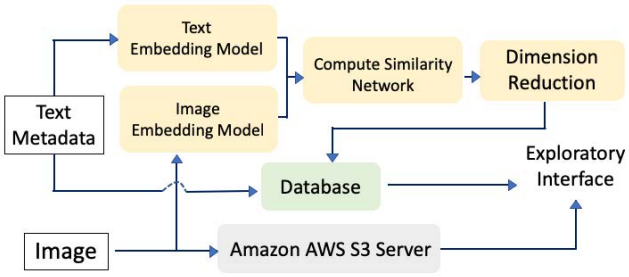


Figure 2: GraviTIE system overview. Yellow: learning the similarity map. Green: learned embeddings and associated metadata are stored in a database. Gray: images are stored in a cloud-based object store.

3 GRAVITIE SYSTEM OVERVIEW

Figure 2 summarizes the architecture of GraviTIE. The pipeline can roughly be divided into three parts: Constructing similarity map, textual and numeric data storage, and image storage.

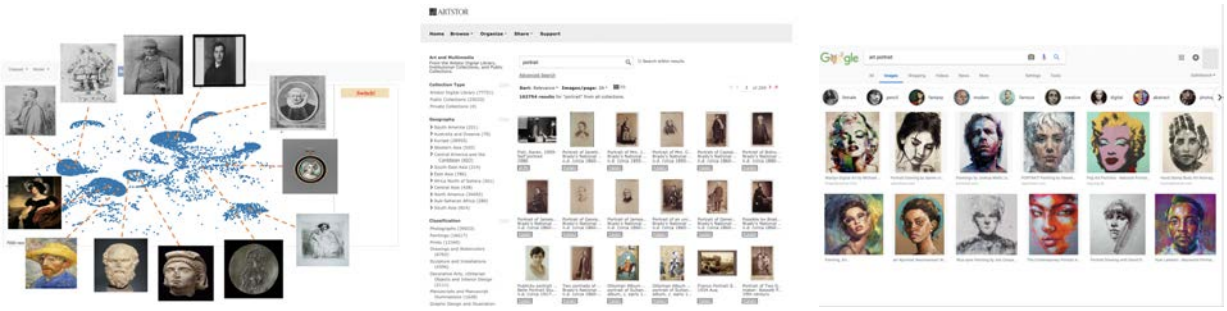
Learning the Similarity Map: The front end interface of GraviTIE takes a 2-d vector of coordinates for each image-text pair to construct the similarity map visualization. In this demonstration, all datasets are pre-processed by the same pipeline in which we first separately embed image and text, acquire a combined representation vector for each text and image pair, and finally apply a dimensional reduction algorithm to obtain the desired 2-d numeric vectors. We use MultiDEC [14] to learn representations of these image-caption pairs. MultiDEC is a method for clustering image-caption pairs by iteratively computing an auxiliary target distribution and matching both image distribution and text distribution to the target. Because MultiDEC produces distinct and semantically meaningful representations for both image and text, it is a natural candidate for this task. Users can also visualize the visual features and textual features alone.

Textual and Numeric Data Storage: The metadata, textual description, and the 2-d numeric vector for each image-text pair is saved in a MySQL database. GraviTIE also utilizes MySQL full text search (boolean mode) to support keyword search.

Image Storage: GraviTIE uses Amazon AWS S3 to store millions of image for display purposes.

4 GRAVITIE DEMONSTRATION DATASETS

We will demonstrate the GraviTIE system on three large-scale datasets to show the versatility and the breadth of GraviTIE.



(a) Search for "portrait" on GraviTIE. (b) Search for "portrait" on Artstor. (c) Search for "portrait" on Google Images.

Figure 3: Search results for "portrait" using (a) GraviTIE, (b) Artstor, and (c) Google Images. GraviTIE summarizes the space of relevant images; different clusters represent different types of portraits. Artstor and Google Images each show only a small set of top-ranked images, and each engine uses a different opaque ranking function, making the results unpredictable.

Russian Tweets. Twitter [4] released over 9 millions tweets from 3,841 accounts that are believed to be connected to Russian Internet Research Agency. This dataset has received significant attention from researchers to understand foreign influence on US politics. We process and display all 543,220 tweets with images on GraviTIE to investigate relationships from a visual perspective.

In this particular demonstration, we first embed images with ResNet-50 [6] model, pre-trained on 1 million ImageNet dataset[4], and acquire a text representation with a Glove [12] model that is pre-trained on 2 billion tweets. We then apply *multiDEC*[15] to learn a joint representation for the combined image-text pair. Finally, UMAP [10] is performed to produce 2-d vectors for visualization.

Art Imagery. GraviTIE includes 2,144,301 art images from the ArtStor collection. Artstor is a non-profit organization that creates digital libraries for scholars arts, humanities, social sciences and architecture. They approached us looking for new ways to explore their large image sets that went beyond simple metadata queries. For these images, the metadata is sparse, so the results of the queries are unreliable. A goal of GraviTIE is to utilize the full coverage of arbitrary data, with no assumptions on completeness. We also want to visually convey the results of unsupervised image clustering. A group of art historians are currently using GraviTIE to find patterns of influence and providing us with feedback.

We extracted visual features for the art images using pre-trained ResNet-50. We concatenate values of title, art classification, creator, repository, and date to serve as the description of each image. We then trained a FastText [2] model based on the description and acquired the textual/meta-data representation for each art work. Similar to the pipeline for the twitter data, we applied *multiDEC* followed by *UMAP* to obtain the final 2-d representation.

Scientific Figures. In addition to the art images and twitter data sets, we use a large database of scientific figures that we have been extracted from PubMed, arXiv and other repositories of the scientific literature. The goal is to understand how the use of different types of figures vary over time and across fields. The example dataset includes over 5 million figures from 800k scientific papers from the arXiv.org. ArXiv provides the raw pdfs; we extract images and captions from these pdfs using pdffigures2.0 [3] and then follow the similarity map pipeline described previously.

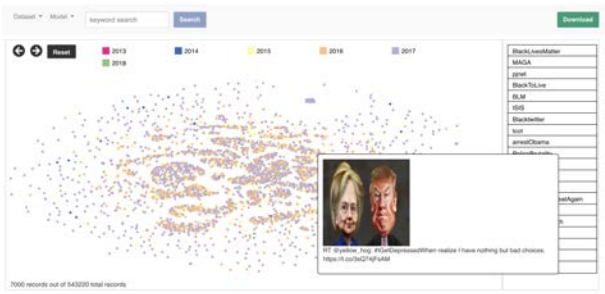
[4]https://about.twitter.com/en_us/values/elections-integrity.html#data

5 KEY FEATURES

GraviTIE supports interactive queries and the exploration images *collections*. The central feature is a similarity map supporting mixed queries, panning, zooming and visual inspection of individual images. Each image-text pair is represented as a single glyph colored by a user-configurable attribute and situated in the 2-D plane based on the similarity with other image-text pairs. Users can hover over each glyph to view the corresponding image and text content associated with that image (Fig. 4a). The query system (Fig. 4c) affords dataset refinement and curation via keyword search or structured predicates. Users can customize the current view through a highlighting feature (Fig. 4d). These customizations can easily be saved and shared with other users since each view is associated with a unique url. Below we highlight other features of the system:

Similarity Map. : The similarity map is what distinguishes the design of GraviTIE from other systems for interacting with large image collections. It makes no assumptions about the images (unlike, say Building Rome in A Day [1]), and can make use of multi-modal information if available. The similarity map can also be used as a replacement for conventional list-based search interfaces: by providing a visual summary of all results rather than the top few items in a ranked list, we allow the user to process information and identify patterns faster than scanning a list of textual information [11].

Consider the simple experiment in Figure 3. We search for the keyword *portrait* on three platforms: GraviTIE, Artstor, and Google Images. GraviTIE reveals that there are several dense clusters of similar images, and upon inspection, these clusters are easily identified as corresponding to print portraits, painted portraits, photograph portraits, and other semantic types. However, the Artstor search results display only the top k images ranked by query relevance, which in this case consist entirely of print portraits. The ranking function used by Google Images makes another seemingly arbitrary choice, displaying entirely painted portraits. Not only do both of these sets of results seem incorrect in general, but even for those situations where they are what the user wants, GraviTIE can emulate similar results within a few hover-and-zoom interactions.



(a) Similarity Map: Each point represents an image-text pair. The user can hover on a point to display the visual and textual content.



(b) Direct inspection: When the number of records is less than 300, thumbnail images are displayed instead of glyphs.



(c) Advanced search: Keyword semantics are configurable.



(d) Highlighting: Items with a selected keyword are highlighted.

Figure 4: Some features of GraviTIE: (a) Similarity map, (b) Direct inspection, (c) Advanced search, (d) Highlighting

Scalability, Query, and Reproducibility. : The similarity map also directly affords scalability. By simply applying a sampling mechanism, we can retain the overall structure of the (embedded) image space while controlling system performance and not occluding individual images. By combining offline pre-processing and sampling, we maintain high performance interactions for datasets of arbitrary scale. Cluster-aware sampling approaches and their impact on interactivity and user task performance remain future work.

GraviTIE supports exploring a set of related keywords at a time. Given a set of user-provided search terms, the user can force each term to be included or excluded. For instance, the results of the query result of Fig.4c would definitely include the term *president*, might contain *trump*, *hilary*, *election*, and *obama*, and would exclude *senator*. In addition, selecting a keyword highlights the elements in the similarity map associated with that keyword, as in Fig. 4d. To initialize this interaction, users can toggle between default interesting keywords (eg. Fig. 4b shows the top 20 frequent hashtags) or type their own (eg. Fig. 4d). If an item includes multiple highlighted keywords, it is indicated with a distinguished color.

Every view in GraviTIE is associated with a unique url to afford collaboration, provenance, and sharing. The browser history can be used to retrace steps, and particular views can be saved simply by copying and pasting links. In addition, a user can download the corresponding dataset associated with any view for further analysis. This feature has been valuable in supporting research by providing an easy way to construct high-quality training datasets from important subsets of images. For example, GraviTIE can be used to quickly find a large set of phylogenetic tree images in the scientific literature for information extraction research[9].

6 CONCLUSION

We have built a fully functional prototype of GraviTIE with several large-scale, image sets to explore. The tool is publicly available and easily accessible in both desktop and mobile environments. For the conference, we plan to have several examples of how the tool can be used to identify and zoom into clusters of similar images, the text associated with these images and nuances in the output of these different unsupervised algorithms. Conference attendees will be able to initiate our pre-baked queries but also initiate their own queries. As an example, we will default to the term "portrait" in the art image set. This will illustrate the distinct types of portraits that exits in the 2.5 million ArtStor images, as mentioned above. Early feedback from collaborators in art history suggest that they can use GraviTIE to quickly understand the coverage of the database in certain time periods and styles (e.g., how many portrait archetypes exist in 20th century paintings). Our goal for the demonstration is to show how GraviTIE works and to gather feedback from users on what kind of functionality is needed to help them explore image sets in ways not available.

7 ACKNOWLEDGEMENT

We would like to thank Artstor who provided their valuable art collection to visualize on GraviTIE.

REFERENCES

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M Seitz, and Richard Szeliski. 2009. Building rome in a day. In *ICCV*. IEEE, 72–79.

[2] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606* (2016).

[3] Christopher Clark and Santosh Divvala. 2016. PDFFigures 2.0: Mining figures from research papers. In *JCDL*. IEEE, 143–152.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. Ieee, 248–255.

[5] Yi Gu, Chaoli Wang, Jun Ma, Robert J Nemiroff, David L Kao, and Denis Parra. 2017. Visualization and recommendation of large image collections toward effective sensemaking. *Information Visualization* 16, 1 (2017), 21–47.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

[7] Nadav Hochman and Lev Manovich. 2013. Zooming into an Instagram City: Reading the local through social media. *First Monday* 18, 7 (2013).

[8] Poshen Lee, Jevin West, and Bill Howe. 2016. VizioMetrix: A Platform for Analyzing the Visual Information in Big Scholarly Data. In *BigScholar Workshop (co-located at WWW)*.

[9] Poshen Lee, T. Sean Yang, Jevin West, and Bill Howe. 2017. PhyloParser: A Hybrid Algorithm for Extracting Phylogenies from Dendrograms. (2017).

[10] Leland McInnes and John Healy. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

[11] Douglas L Nelson, Valerie S Reed, and John R Walling. 1976. Pictorial superiority effect. *Journal of Experimental Psychology: Human Learning and Memory* 2, 5 (1976), 523.

[12] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. 1532–1543.

[13] Chaoli Wang, John P Reese, Huan Zhang, Jun Tao, Yi Gu, Jun Ma, and Robert J Nemiroff. 2015. Similarity-based visualization of large image collections. *Information Visualization* 14, 3 (2015), 183–203.

[14] Sean Yang, Kuan-Hao Huang, and Bill Howe. 2019. MultiDEC: Multi-Modal Clustering of Image-Caption Pairs. *arXiv preprint arXiv:1901.01860* (2019).

[15] Sean Yang, Kuan-Hao Huang, and Bill Howe. 2019. MultiDEC: Multi-Modal Clustering of Image-Caption Pairs. (2019). arXiv:arXiv:1901.01860

TaxVis: a Visual System for Detecting Tax Evasion Group

Hongchao Yu
School of Electronic and Information Engineering
Key Lab of Intelligent Networks and Network Security
Xi'an Jiaotong University
Xi'an, Shaanxi, China
yuhongchao@stu.xjtu.edu.cn

Huan He
School of Electronic and Information Engineering
Key Lab of Intelligent Networks and Network Security
Xi'an Jiaotong University
Xi'an, Shaanxi, China
hehuan@mail.xjtu.edu.cn

Qinghua Zheng
School of Electronic and Information Engineering
Key Lab of Intelligent Networks and Network Security
Xi'an Jiaotong University
Xi'an, Shaanxi, China
qhzheng@mail.xjtu.edu.cn

Bo Dong
School of Continuing Education
National Engineering Lab for Big Data Analytics
Xi'an Jiaotong University
Xi'an, Shaanxi, China
dong.bo@mail.xjtu.edu.cn

ABSTRACT

The demo presents TaxVis, a visual detection system for tax auditor. The system supports tax evasion group detection based on a two-phase detection approach. Different from the pattern matching based methods, this two-phase method can analyze the suspicious groups automatically without artificial extraction of tax evasion patterns. In the first phase, we use a network embedding method node2vec to learn representations that embed corporations from a Corporation Associated Network (CANet), and use LightGBM to calculate a suspicious score for each corporation. In the second phase, the system use three detection rules to analyze the transaction anomaly around the suspicious corporations. According to these transaction anomalies, we can discover potential suspicious tax evasion groups. We demonstrate TaxVis on tax data of Shaanxi province in China to verify the usefulness of the system.

CCS CONCEPTS

• **Information systems** → **Expert systems**; • **Applied computing** → *System forensics.*

KEYWORDS

TaxVis; Network embedding; Tax evasion group.

ACM Reference Format:
Hongchao Yu, Huan He, Qinghua Zheng, and Bo Dong. 2019. TaxVis: a Visual System for Detecting Tax Evasion Group. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019,San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314144

1 INTRODUCTION

Tax evasion detection is a challenging work for every country. It is reported by the Chinese government that the loss rate of tax revenue in China is above 22 percent [7]. In addition to suspicious individual detection, there is a new tendency that corporations plot together to evade tax [2, 7]. The existing solutions for detecting tax evasion groups are mainly based on graph pattern matching. In these methods, the pattern of tax evasion group must be extracted from massive cases by the tax auditors manually. As a result, there is a limitation that these patterns cannot be constantly updated to keep up with the rapid changes of tax evasion since it takes great effort for tax auditors to identify new patterns.

Given the limitations of the pattern matching based methods, we designed a two-phase approach for detecting tax evasion groups. The first phase is suspicious individual detection which use node2vec and LightGBM to detect suspicious corporations in the CANet. In [2, 6, 7], we can find that the relationships between corporations play an import role in many tax evasion patterns. As show in Figure 1, in order to utilize these relationships, we choose the network embedding method node2vec to automatically embed the corporations from the transaction and investment relationship graph. And then we use LightGBM to calculate a suspicious score for each corporation. The second phase is suspicious group analysis which can inspect whether the corporation belongs to a suspicious tax evasion group. In this phase, we use three rules to detect transaction anomalies around each corporation, and then according to these anomalies we can get some suspicious groups.

We demonstrate TaxVis based on the tax data of Shaanxi province. After selecting the detection year, our system will automatically compute the suspicious score of each corporation. Next, clicking one corporation, the system will display the details of its transactions, which include the transaction volume, the commodity category, and the transaction corporation etc. And then based on these transaction details, we use three rules to analyze the abnormal transaction of the selected corporation in a transaction flow chart. Finally, we can find whether the corporation belongs to a suspicious tax evasion group. A demonstration video is available at the following address: https://youtu.be/46M0talBclo. Our contributions are as following:

- We propose a two-phase approach to calculate the suspicious score based on the representations of corporations in CANet and detects suspicious tax evasion groups automatically.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17,2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314144

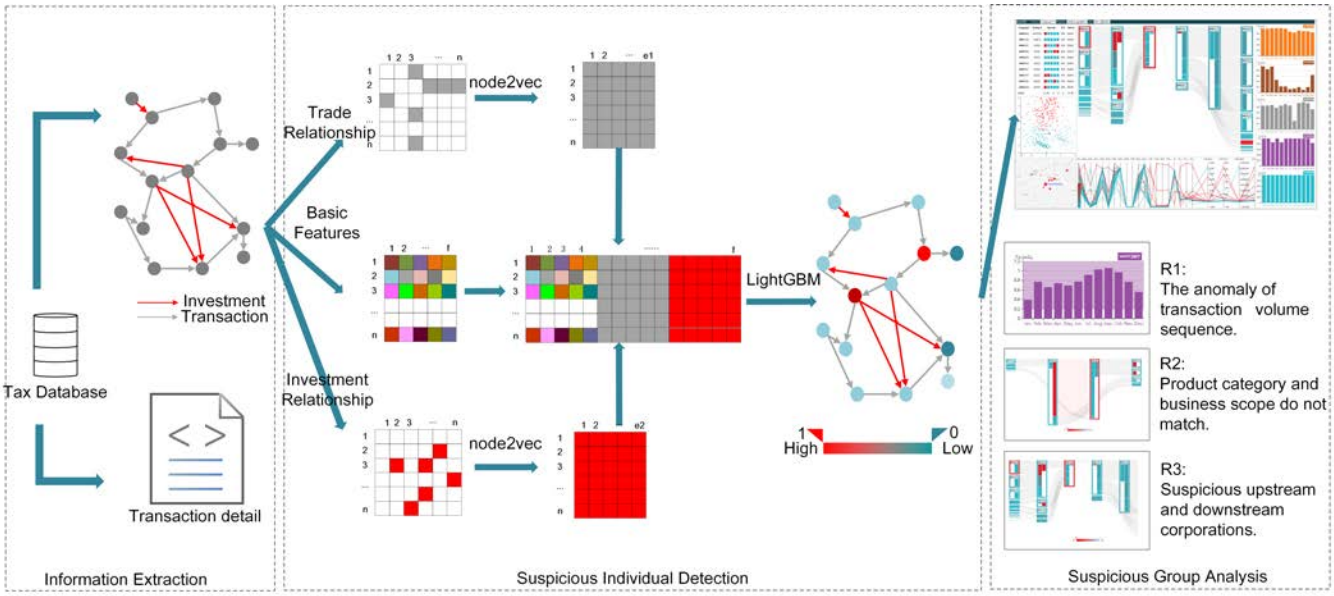


Figure 1: The structure of the TaxVis. First, we extract information from tax database. Then we detect suspicious individuals by node2vec and LightGBM. Finally, we use a visualization system to analyze the suspicious tax evasion groups.

- We design a transaction flow chart to visualize the transaction anomalies around corporations.
- We present a visual analytic system based on our proposed two-phase approach and the transaction flow chart for tax audition officer to discover suspicious tax evasion groups.

2 SYSTEM DESIGN

The structure of this system is shown in Figure 1, which consists of three components.

The first component is the information extraction, which collects all the basic features of corporations and the relationships between them, including transaction and investment relationships. The second component is the suspicious individual detection. It detects suspicious individuals by combining the node representation vectors and basic feature vectors. The last component is the suspicious group analysis. We use three detection rules to detect whether the corporation belongs to a suspicious group. The details of each component are as follows.

2.1 Information Extraction

In this component, we extract the corporationâĂŹs information from tax database. Each corporation's information consists of three parts. The first part is the corporationâĂŹs basic features, including the corporation's registration time, the registration area and the age of the legal representative, etc. The second part is the transaction and investment relationships between corporations. The third part is the transaction detail data that can be used to detect suspicious groups. The transaction detail includes the transaction volume per month, the transaction time and the commodity category, etc. Based on the above data, we build a corporation associated network (CANet) $G = \{V, E_1, E_2, A\}$, where $V = \{V_1, \ldots, V_n\}$ is a set of corporations, $E_1 = \{e_{ij}^1\}$ denotes the transaction relationships between corporations, $E_2 = \{e_{ij}^2\}$ denotes the investment relationships between corporations, and $A = \{a_i\}$ denotes the attributes of corporations.

2.2 Suspicious Individual Detection

In this component, we detect suspicious individuals in the CANet with node2vec and LightGBM. The technical details are as follows.

2.2.1 Node2vec. Node2vec is a network embedding method based on random walk that can learn continuous feature representations for nodes in a network [4]. The method is built on the assumption that the representations of adjacent nodes are approximated. Choosing node2vec is based on the following three reasons.

Transaction and investment relationships are important information in detecting tax evasion. In the previous researches about tax evasion groups, transaction and investment relationships between corporations play an important role in some suspicious tax evasion patterns [7]. Therefore, itâĂŹs necessary to combine these relationship features into the corporation features, and node2vec is suitable to learn relationship representations of corporations in the transaction and investment network.

Low time complexity. The time complexity of node2vec is O(|V|d), where V is the set of the vertices in the graph and d is the representation dimension [3]. Since in practice we have an extremely large CANet that contains millions of corporation nodes, it is necessary to choose node2vec to perform real time detection.

Both Local and global network structure can be extracted. According to the previous studies, some suspicious patterns involve not only local structure, but also global structure in the transaction and investment relationships [2, 7]. Compared with the previous methods, node2vec provides a trade-off between breadth-first (BFS) and depth-first (DFS) graph searches. Through breadth-first search,

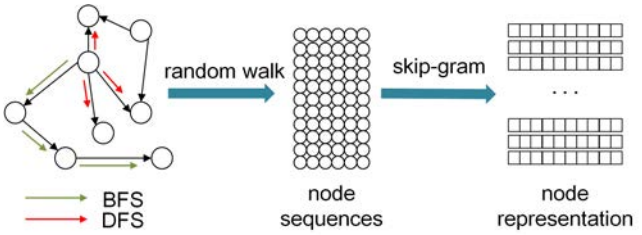


Figure 2: The procedure of the node2vec.

we can get local network structure information, and through depth-first search, we can get global network structure information. In this algorithm, we can get any combination of the two kinds of information by adjusting the parameters.

As is shown in Figure 2, node2vec consists of three steps: computing transition probability matrix, obtaining node sequences in the graph by random walk, and representing the node vectors by the skip-gram algorithm. In this system, we use node2vec to represent the transaction network and the investment network respectively, then combine the representation vectors and the basic features of the nodes, and finally combine these as inputs to LightGBM algorithm to get the suspicious score of each corporation.

2.2.2 LightGBM. LightGBM is a gradient boosting decision tree algorithm, which is effective when the feature dimension is high and data size is large [5]. In the experiment, we tested four classification algorithms (i.e., LightGBM, RandomForest, SVM and AdaBoost) for better performance. For each algorithm, we performed two experiments separately, one uses all the features of the corporation, and the other uses the basic features of the corporation. The result is show in Figure 3, in which the dotted line represents the basic feature experiments, and the solid line represents all feature experiments. As we can see, LightGBM outperforms other algorithms when we use all features. Besides, compare with using basic feature, using all features significantly improved the performances of suspicious individual detection algorithms.

2.3 Suspicious Group Analysis

In this component, we develop a visual analytic system to anticipate the suspicious groups with the following detection rules. **Rule 1: IF the sequence of transaction volume has anomaly, THEN the transaction is fraudulent**. The anomaly of the transaction volume is an important signal for fraud behaviors [1]. Tax evasion case studies found that tax evasion corporations that have been established soon will run away if they open a lot of invoices in a short time. As shown in Figure 4(c), in our system we use the $3-\theta$ principle to detect transaction anomaly. When the trading volume of a month exceeds a range calculated based on the trading volume of past 12 months, the edge in the flow chart will be marked red and the warning message will be shown when mouse moves over the edge.

Rule 2: IF the product category does not meet the business scope of the two parties, THEN the transaction is fraudulent. This rule refers to the fact that some corporations manage to increase their own input costs by purchasing invoices of other category goods. Therefore, in our approach, when a corporation buys

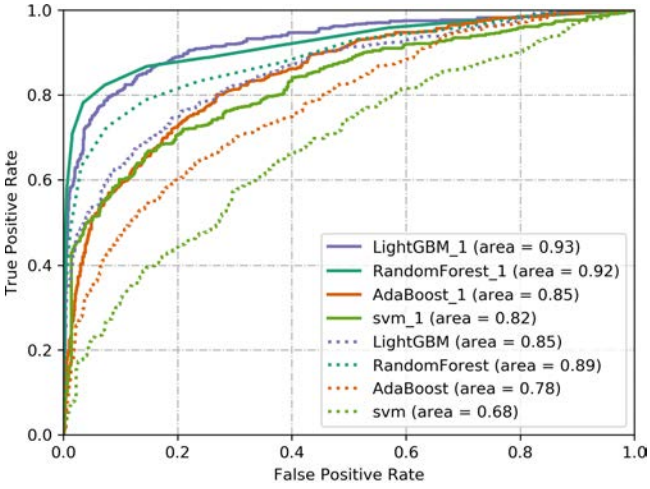


Figure 3: The ROC curves of the four algorithms.

and sells goods and the corporation's business scope do not match, we assume the transaction is fraudulent.

Rule 3: IF there exist suspicious upstream and suspicious downstream corporations, THEN the corporation is suspicious. Due to the special characteristics of the value-added tax (VAT): the amount of tax paid should be the output minus the input. In order to reduce taxes, some corporations may increase the cost of purchasing goods or reduce sales revenue. Therefore, all corporations in the VAT trading chain restrict each other and many tax evasion taxes are carried out in groups [7]. So if there exists suspicious upstream and suspicious downstream corporations, we assume that the corporation is suspicious though its suspicious score is low.

In order to analyze these anomalies, we design a transaction flow chart. As shown in Figure 4(b), the corporations are represented as rectangle blocks, and the transaction edge between corporation nodes are represented as curves. Each rectangle block consists of two components, the border and the internal influence area. The color of the border corresponds to the suspicious score (the darker red indicates higher suspicious score). The internal influence area is divided into two parts. The left area represents the influence of the upstream corporations and the right area represents the influence of the downstream corporations. Each area is divided into several areas according to the relative volume of each corporation. The color of each area represents the suspicious score of the corresponding corporation.

In the transaction flow chart, the gray edge indicates normal transaction, while the red indicates abnormal transaction (Figure 4(b-2)), which represents those transactions match the rule 1 and rule 2 aforementioned. When mouse is over the red edge, users can see the specific abnormal information. In addition, we use a histogram to show the monthly volume of transactions (Figure 4(c)). In addition to the transaction flow chart, this system also designs a parallel coordinate showing the features of all corporations, a classification result chart showing the performance of the detection algorithm and a map chart showing the geographic location of the selected corporation.

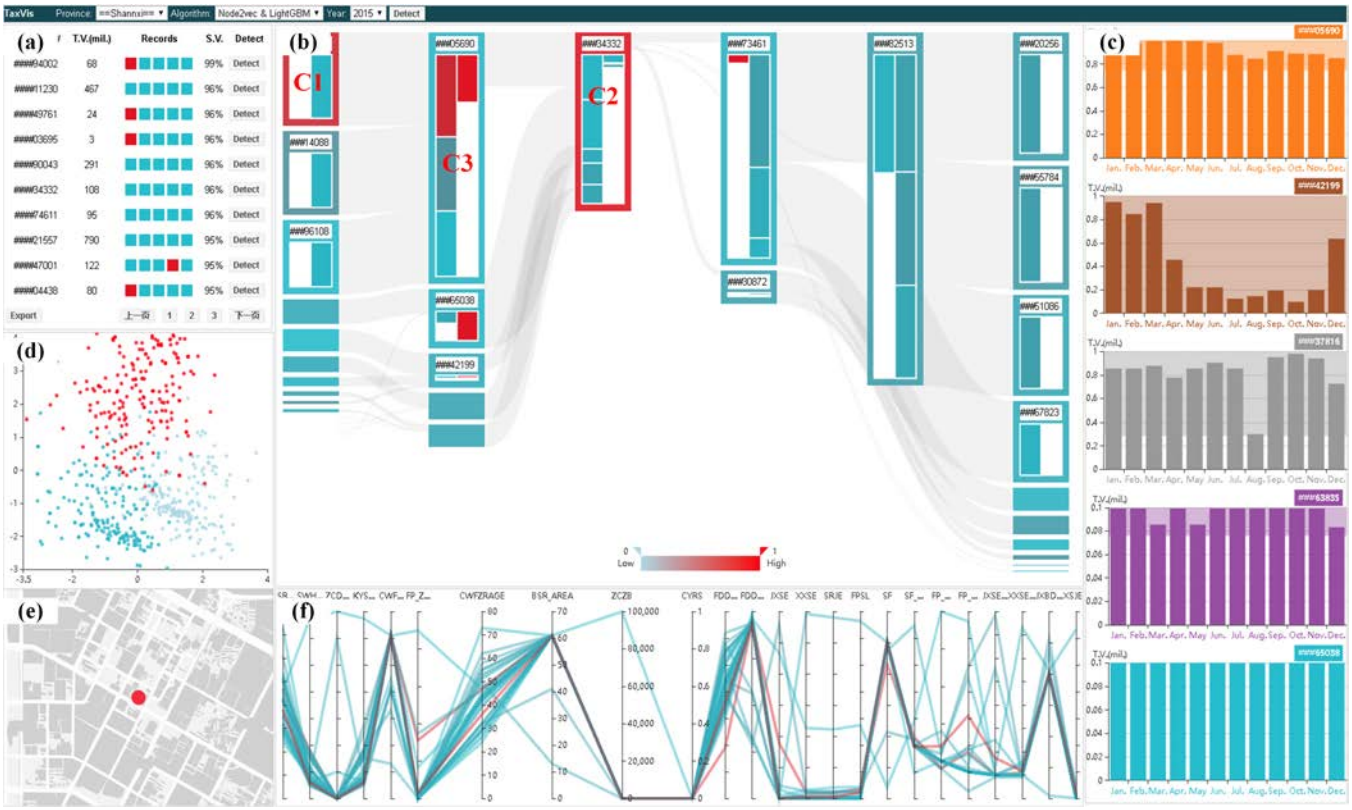


Figure 4: Screenshot of the TaxVis platform, including (a) the suspicious score table showing the suspicious individual detection result in descending order of the suspicious score; (b) the transaction flow chart depicting the trading corporations of the selected corporation and the transaction details between them; (c) the trading volume chart showing the trading volume between the corporations pre month; (d) the classification result chart showing the performance of the detection algorithm; (e) the map chart showing the geographic location of the selected corporation; (f) the parallel coordinate showing the features of all corporations.

3 DEMONSTRATION OUTLINE

We will demonstrate our system based on real tax dataset from Shaanxi province tax data in 2014-2015. There are 10,120 corporations in the dataset.

Detection of tax evasion individual. After selecting the year, our system will automatically show the detection results. The results are listed in descending order of suspicious score. In the result table (Figure 4(a)) each row corresponds to a corporation. The detail consists of the corporation id, the turnover of the past year, the tax evasion record of the past 5 years, and the suspicious score. Selecting one corporation, clicking the detect button and the rest of the page will display the corresponding information for the selected corporation.

Analysis of the tax evasion tax group. According to the previous three rules, we can check the abnormal transaction edges around the suspicious corporation to analyze the suspicious group. For example, as shown in Figure 4(b), the suspicious score of corporation C3 is very low, the suspicious scores of its upstream corporation C1 and downstream corporation C2 are very high. As a result, we can infer that the corporations C1, C2 and C3 may belong to a suspicious tax evasion group.

4 CONCLUSION AND FUTURE WORK

In this demo, we propose a two-phase approach to calculate the suspicious score based on the representations of corporations in CANet and detects suspicious tax evasion groups automatically. In future work, on the one hand, we will use some techniques to discover anomalous transactions in the second phase. On the other hand, we will apply some methods to automatically extract the tax evasion patterns to enhance the pattern matching based methods.

ACKNOWLEDGMENTS

This research was partially supported by "The Fundamental Theory and Applications of Big Data with Knowledge Engineering" under the National Key Research and Development Program of China with Grant No. 2018YFB1004500, the MOE Innovation Research Team No. IRT17R86, the National Science Foundation of China under Grant Nos. 61721002, 61532015, and Project of China Knowledge Centre for Engineering Science and Technology.

REFERENCES

[1] Bart Baesens, VÃĺronique Van Vlasselaer, and Wouter Verbeke. [n. d.]. *Fraud Analytics Using Descriptive, Predictive, and Social Network Techniques: A Guide to*

Data Science for Fraud Detection. John Wiley & Sons, Inc. https://doi.org/10.1002/9781119146841

[2] Walter Didimo, Luca Giamminonni, Giuseppe Liotta, Fabrizio Montecchiani, and Daniele Pagliuca. [n. d.]. A visual analytics system to support tax evasion discovery. 110 ([n. d.]), 71–83. https://doi.org/10.1016/j.dss.2018.03.008

[3] Palash Goyal and Emilio Ferrara. [n. d.]. Graph embedding techniques, applications, and performance: A survey. 151 ([n. d.]), 78–94. https://doi.org/10.1016/j.knosys.2018.03.022

[4] Aditya Grover and Jure Leskovec. [n. d.]. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16* (2016). ACM Press, 855–864. https://doi.org/10.1145/2939672.2939754

[5] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. [n. d.]. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 3146–3154. http://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf

[6] Jianfei Ruan, Zheng Yan, Bo Dong, Qinghua Zheng, and Buyue Qian. [n. d.]. Identifying suspicious groups of affiliated-transaction-based tax evasion in big data. 477 ([n. d.]), 508–532. https://doi.org/10.1016/j.ins.2018.11.008

[7] Feng Tian, Tian Lan, Kuo-Ming Chao, Nick Godwin, Qinghua Zheng, Nazaraf Shah, and Fan Zhang. [n. d.]. Mining Suspicious Tax Evasion Groups in Big Data. 28, 10 ([n. d.]), 2651–2664. https://doi.org/10.1109/TKDE.2016.2571686

Tablepedia: Automating PDF Table Reading in an Experimental Evidence Exploration and Analytic System

Wenhao Yu
University of Notre Dame
Notre Dame, IN
wenhaoyu97@gmail.com

Zongze Li
Sichuan University
Chengdu, Sichuan, China
waterobrien@gmail.com

Qingkai Zeng
University of Notre Dame
Notre Dame, IN
qzeng@nd.edu

Meng Jiang
University of Notre Dame
Notre Dame, IN
mjiang2@nd.edu

ABSTRACT

Web research, data science, and artificial intelligence have been rapidly changing our life and society. Researchers and practitioners in the fields take a large amount of time to read literature and compare existing approaches. It would significantly improve their efficiency if there was a system that extracted and managed experimental evidences (say, a specific method achieves a score of a specific metric on a specific dataset) from tables of paper PDFs for search, exploration, and analytic. We build such a demonstration system, called *Tablepedia*, that use rule-based and learning-based methods to automate the "reading" of PDF tables. It has three modules: template recognition, unification, and SQL operations. We implement three functions to facilitate research and practice: (1) finding related methods and datasets, (2) finding top-performing baseline methods, and (3) finding conflicting reported numbers. A pointer to a screencast on Vimeo: https://vimeo.com/310162310

KEYWORDS

PDF table, Experimental evidence, Data science

ACM Reference Format:
Wenhao Yu, Zongze Li, Qingkai Zeng, and Meng Jiang. 2019. Tablepedia: Automating PDF Table Reading in an Experimental Evidence Exploration and Analytic System. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314118

1 INTRODUCTION

The motivation of building the proposed system was originally from our literature study on *multilabel classification* which is to predict the labels of objects where multiple labels may be assigned to each object. It cost us as long as 23 days to collect, read, and digest hundreds of related works. We found two papers of this topic that were accepted to ACM SIGKDD 2017 Research Track: PPDSparse [11] and AnneXML [9]. Each of them proposed a new multilabel classification model and compared with baseline methods. They both reproduced and tested existing methods (such as SLEEC, FastXML, PfastreXML, and PDSparse) on publicly available data sets (such as AmazonCat-13K, Delicious-200K, and WikiLSHTC-325K) using standard evaluation metrics (such as Precision@1, P@3, and P@5). Table 1 summarizes and compares the numbers given by the two papers, [11] on the left and [9] on the right. We find out that almost half of the pairs have bigger than 3% difference on the scores, which has been able to be claimed as significant improvement on well-accepted benchmarks. This may be due to the random initialization, parameter settings, or computational environments. We have no idea about the true reason, but we argue that it is worthwhile of investigating the *experimental evidences* in data science literature.

Table 1: Tablepedia found inconsistent numbers by two KDD papers [11] (left) and [9] (right) for multilabel classification. Precision differences of bigger than 3% are underlined. It is worthwhile of attention to the inconsistency.

Dataset	(%)	SLEEC	FastXML	PfastreXML	PDSparse
AmazonCat	P@1	90.56/89.19	94.02/93.10	86.06/89.94	87.43/89.31
-13K	P@3	76.96/75.17	79.93/78.18	86.06/77.24	87.43/74.03
	P@5	62.63/61.09	64.90/63.38	63.65/63.53	56.70/60.11
Delicious	P@1	47.78/47.03	48.85/43.20	26.66/37.62	37.69/34.37
-200K	P@3	42.05/41.67	42.84/38.68	23.56/35.62	30.16/29.48
	P@5	39.29/38.88	39.83/36.21	23.21/34.03	27.01/27.04
WikiLSHTC	P@1	58.34/55.57	50.01/49.75	57.17/58.10	60.70/61.26
-325K	P@3	36.70/33.06	32.83/33.10	37.03/37.61	39.62/39.48
	P@5	26.45/24.07	24.13/24.45	27.19/27.69	29.20/28.79

Again, it took us long long time to make the comparison table (and several other tables that compared methods in other conference or journal papers). Therefore, we aim at building a system to extract and manage such experimental results in the literature. We hope that researchers and practitioners in the fields of data science and artificial intelligence will use it as Wikipedia to satisfy their needs of exploring and analyzing the experimental evidences.

The key challenges lie in automating the "reading" of tables in the experimental sections of paper PDFs. *First*, there was no well-defined structure of experimental evidence. The tables are embedded in the PDF format. It takes careful engineering efforts on cropping, parsing, and cleaning the tables. *Second*, the tables have different kinds of templates, so there was no standard of interpreting the cells. *Third*, the roles of row and column names (such as SLEEC and P@1), say, datasets or methods or metrics, are unknown. The gap between PDF table and queryable database is huge.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314118

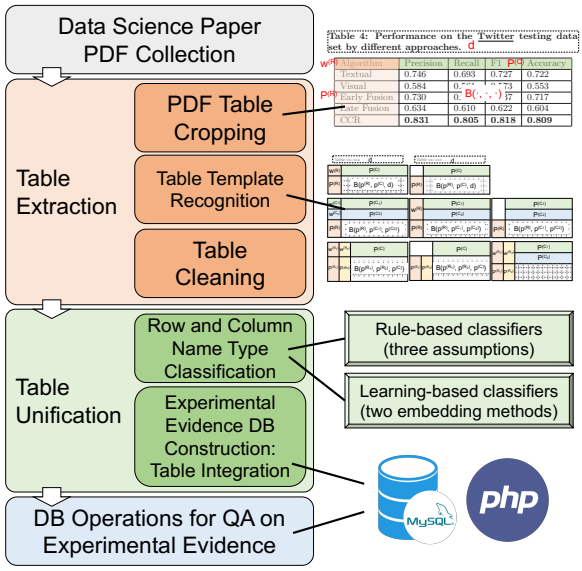


Figure 1: Tablepedia workflow: from PDF collection, to table extraction, to experimental evidence database construction, to database operations and visualization.

Proposed approach. This paper presents a novel system, called Tablepedia, which transforms data science paper PDFs into a structured database of experimental evidences, and support multiple exploratory and analytic functions over the constructed database for knowledge discovery. It has three modules. The first module *table extraction* crops the tables from PDFs and recognize their templates. The second module *table unification* classifies the column names and row names into the three types of labels (method, dataset, and metric) and then unifies each cell into a **(method, dataset, metric, score, source)-tuple**. The score is the cell's value and the source is the PDF file name, page, and number of the table. This module constructs a five-column database of the tuples for every table that contains experimental results. The third module *database operation for QA* uses SQL operations (i.e., *select* and *join*) for question-answering on the experimental result database.

Contributions. The contributions and features of the Tablepedia system are summarized as follows.

- A novel system that extracts experimental evidences from massive literature in PDF format. This builds up the first experimental knowledge base for data science and artificial intelligence research.
- An effort-light framework that leverages both rule-based and learning-based methods to unify the tables of experimental results into (method, dataset, metric, score, source)-tuples.
- Capabilities for exploration and analysis over the structured knowledge base to facilitate research and practice.

The Tablepedia demo system will be made available online for interactive use after its demonstration in the conference. A pointer to a screencast on Vimeo: https://vimeo.com/310052262

2 THE TABLEPEDIA SYSTEM

In this section, we first introduce the workflow of our Tablepedia system and the details of the three modules of the system.

Overview. Figure 1 shows the overflow. Tablepedia collects a set of data science paper PDFs. It has three modules to process the PDF data. It first crops the tables from PDFs, recognizes the table templates, and cleans the table data. Second, it classifies the row and column names of each table into three categories (method, dataset, metric). The experimental evidence database is constructed through the integration of table cells. Lastly, it designs database operations for knowledge exploration in the structured database.

Expected output and impact. Figure 2 shows a snapshot of the experimental evidence database. It has several examples of data records. They are experimental facts that can be found in tables of conference and journal papers on building recommender systems that were published in the same year: TOIS'11 [8], TIST'11 [6], and WSDM'11 [7]. The tables share popular method names such as "User Mean", "NMF", and "PMF". Which method performs the best? Are the reported numbers of their performances consistent in these tables? When the tables were well structured into such a database, the above questions could be easily answered. The number of publications in the field of data science has been tremendously increasing because of the great use of data mining and machine learning in real applications. Practitioners are curious about what method will generate good performance on a specific task and dataset. Researchers are wondering whether the baseline methods are the state-of-the-art and whether the reported numbers on the baselines are correct when they review papers.

2.1 Table Extraction

We use Tabula to extract tabular content from PDF [2]. Tabula was created by Manuel Aristaran *et al.* with the first release made available early 2013 as an open source project. The developers stated that they were inspired by academic papers [12] about analysis and extraction of tabular content. Tabula is available as a Java library [1]. Unfortunately, it does not work for scanned documents, so we filter those files out.

A table $T = \{\mathcal{R}, C, d, \mathcal{B}\}$ has four components: (1) a model of horizontal *Rows* (identifiable by name) $\mathcal{R}$, (2) a model of horizontal *Columns* C, (3) *Caption* and the set of words in the caption d, and (4) cells (data elements) in the table's *Body* $\mathcal{B}$. We observe that the tables can be categorized into eight major templates (with very few exceptions). Figure 3 visualizes the components of each template.

For cleaning the raw tables, we count the number of digit-format cells in the table: we filter out the tables that have fewer than 6 digit cells, and thus, we get rid of more than 99% of the non-experimental result tables. We use [10] to remove the text that was not the table's caption but located around the table.

2.2 Table Unification

We define the set of concept items that can be found as row names or column names:

$$\mathcal{P} = \cup_{T=[\mathcal{R},C,d,\mathcal{B}]} P^{(R_{(\cdot)})} \cup P^{(C_{(\cdot)})}, \quad (1)$$

where T is a table, $P^{(R_{(\cdot)})}$ is the set of row names (no matter single row or double rows), and $P^{(C_{(\cdot)})}$ is the set of column names. We denote by $\mathcal{L}$ by the set of three labels for the concept items:

$$\mathcal{L} = \{\text{“}method\text{”}, \text{“}dataset\text{”}, \text{“}metric\text{”}\}. \quad (2)$$

	A	B	C	D	E
1	**Method**	**Dataset**	**Metric**	**Score**	**Source**
10	UserMean	Epinions	MAE	0.9319	TOIS11-paper7-table3
11	UserMean	Epinions	MAE	0.9285	TIST11-paper3-table3
12	UserMean	Epinions	MAE	0.9285	WSDM11-paper12-table5
109	ItemMean	Epinions	RMSE	1.1973	TOIS11-paper7-table4
110	ItemMean	Epinions	RMSE	1.2584	TIST11-paper3-table3
111	ItemMean	Epinions	RMSE	1.2584	WSDM11-paper12-table5
112	Trust	Epinions	RMSE	1.2132	TIST11-paper3-table3
113	NMF	Epinions	RMSE	1.1832	TOIS11-paper7-table4
114	NMF	Epinions	RMSE	1.1832	TIST11-paper3-table3
115	NMF	Epinions	RMSE	1.1832	WSDM11-paper12-table5
116	SVD	Epinions	RMSE	1.1812	TOIS11-paper7-table4
117	TCF	Epinions	RMSE	1.1761	TIST11-paper3-table3
118	PMF	Epinions	RMSE	1.1760	TOIS11-paper7-table4
119	PMF	Epinions	RMSE	1.1760	TIST11-paper3-table3
120	PMF	Epinions	RMSE	1.1760	WSDM11-paper12-table5
121	SoRec	Epinions	RMSE	1.1492	TOIS11-paper7-table4
122	RSTE	Epinions	RMSE	1.1256	TIST11-paper3-table3
123	RSTE	Epinions	RMSE	1.1256	WSDM11-paper12-table5
124	SR1VSS	Epinions	RMSE	1.1016	WSDM11-paper12-table5
125	SR1PCC	Epinions	RMSE	1.1013	WSDM11-paper12-table5
126	SR2VSS	Epinions	RMSE	1.0958	WSDM11-paper12-table5
127	SR2PCC	Epinions	RMSE	1.0954	WSDM11-paper12-table5
169	SoRec	MovieLens	RMSE	...	...

Figure 2: Tablepedia generates this experimental evidence database from data science paper PDFs. For a dataset and an evaluation metric, one can use the database to check what the state-of-the-art (highlighted in yellow) is and whether the reported numbers in existing researches are consistent (green box) or conflicting (red box).

Then we define table unification as a two-step problem.

Problem (Table unification). ***Given*** *a set of tables* $\{T\}$ *and each table has been well defined based on its template, (1)* ***classify*** *the concepts into three categories, or say,* ***find*** *a classification function* $f: \mathcal{P} \rightarrow \mathcal{L}$*; (2)* ***unify*** *the cells into (method, dataset, metric, score, source)-tuples, or say,* ***find*** *a function of three variables* $g: P^{(\text{“}method\text{”})} \times P^{(\text{“}dataset\text{”})} \times P^{(\text{“}metric\text{”})} \rightarrow \mathbb{R}$*, where the target value is the score (a real number) as in the Table's body function B.*

Tablepedia develops an ensemble learning approach that iteratively predicts the labels of concept items using two classifiers of different methodologies. The first classifier is an **assumption/rule-based** method. The first assumption is:

Assumption 1 (Row/column header indication). *If the upper-leftmost cell of the table has a specific word (e.g., "Methods", "Dataset"), the names on the corresponding columns/rows are more likely to have the label as the word indicates.*

For example, if the upper top cell of a table has word "Methods", then the row names such as "User Mean", "Item Mean", and "NMF" are likely to be labelled as "method". Then we use these "seed" concept items to label columns, rows, and captions of all the tables. If the columns/rows/captions are partially labelled, we will be able to use the following assumptions to predict the labels of the remaining concepts on the columns/rows/captions:

Assumption 2 (Row/column type consistency). *The concept items on the same column or row are likely to have the same type of label. For example, if we know (1) "NMF" is a "method" and (2) "SVD" locates in the same column/row as "NMF" does, then "SVD" is likely to be a "method".*

Assumption 3 (Cell context completeness). *A table often covers all the three types of labels on its columns, rows, and caption, in order to provide complete contexts [4] to explain the values in the cells. For example, if the caption has a metric name (i.e., "MAE") and the row names are methods, then the column names are likely to be datasets.*

The second classifier is a **learning-based** method. It learns low-dimensional representations (or called *embeddings*) of the concept items for label prediction. It learns (a) *semantic concept embeddings* [5] from the unstructured paper texts and (b) *structural concept embeddings* [3] from the co-occurrences of concepts in the table structures (i.e., columns and rows), and then feeds the concatenated feature vectors into standard multi-class classification models (e.g., random forests, support vector machines, neural networks) to predict the labels. This solution will be likely to assign the same label to different forms of the same concept.

The ensemble learning approach uses the boosting strategy to iteratively learn the classifiers. Relying too much on one classifier may result in lots of errors when the prediction confidence is weak. In each iteration, we only expand the set of classified concepts for next round of training by a certain small number. So after several iterations, we can have precise labels of the concept items.

The next step is to unify the table cells into the tuples and put into the five-column experimental result database (ERD). The ERD's quality relies on the accuracy of concept typing. Tablepedia achieves an F1 score of 0.8477, which is much higher than using rule-based or learning-based method only (0.6542).

2.3 Database Operations for QA

When the ERD was constructed, we would be able to use SQL queries and operations to answer interesting questions. There could be many questions and corresponding SQL queries.

Question 1. How many methods were used/proposed on the Epinions dataset? And how many metrics were used?

Question 2. What are the top three methods on the Epinions dataset if the evaluation metric is RMSE?

Question 3. Are there conflicting reported numbers in the database? What are they?

SQL consists of many types of expressions, predicates and statements such as *select*, *join*, and *distinct*, based upon *relational algebra* and *tuple relational calculus*. Suppose the experimental result data table is constructed and named as "ERD". Here are the SQL queries that find answers to the above questions.

SQL Queries 1.

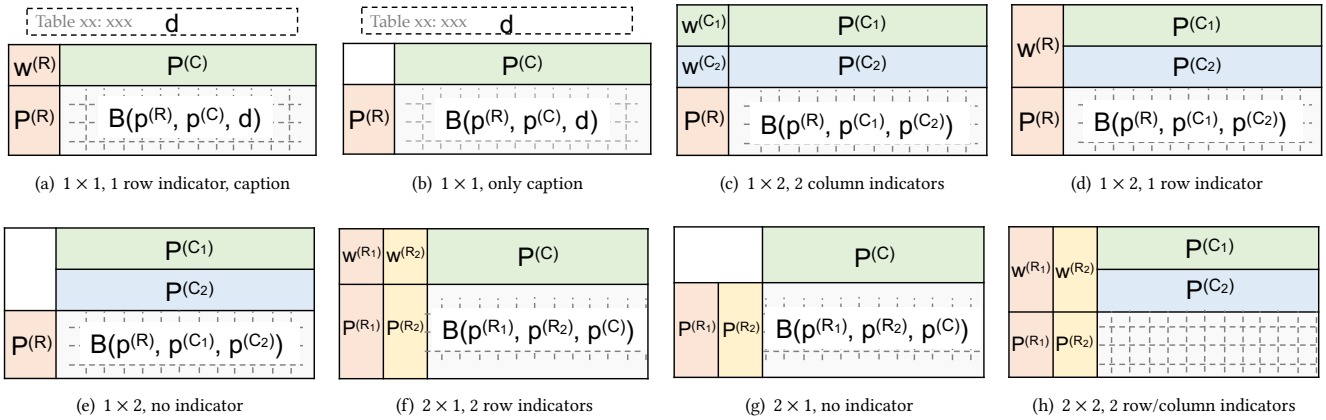
```
select count(distinct Method) from ERD where Dataset="Epinions";
select count(distinct Metric) from ERD where Dataset="Epinions";
```

SQL Query 2.

```
select * from ERD where Dataset = "Epinions" and Metric = "RMSE" order by Score desc limit 3;
```

SQL Query 3.

```
select distinct d1.Method, d1.Dataset, d1.Metric, d1.Score, d1.Source from ERD as d1, ERD as d2 where d1.Method = d2.Method and d1.Dataset = d2.Dataset and d1.Metric = d2.Metric and d1.Score <> d2.Score order by d1.Method, d1.Dataset, d1.Metric;
```



(a) 1×1, 1 row indicator, caption (b) 1×1, only caption (c) 1×2, 2 column indicators (d) 1×2, 1 row indicator

(e) 1×2, no indicator (f) 2×1, 2 row indicators (g) 2×1, no indicator (h) 2×2, 2 row/column indicators

Figure 3: Eight major table templates: We will use the first seven templates which cover more than 95% of the tables in our dataset. The cells in the table's body are triplets based on rows/columns/caption. (Best view in color)

The term count is used for question "how many"; order by is used for ranking/finding "top three"; and the third query uses *self-join* to compare values in a column ("Score") with other values in the same column in the same table ("ERD").

We developed user-friendly functions in this module to answer the questions. For example, users can fill in the underlined values in the questions, the SQL queries will be updated, and then correct answers will be returned.

3 DATA STATISTICS

We downloaded from web portals such as ACM Digital Libraries a PDF file collection of four data science conference proceedings (WWW, SIGKDD, ICDM, and WSDM) and three ACM transactions (TOIS, TIST, and TKDD) last decade (2008–2017). After careful PDF converting, cropping, and cleaning, we have **456 tables**.

Tablepedia categorized **4,476 concepts** in the 456 tables into three classes, {dataset, method, metric}. The resulting database has as many as **29,081 data records** (or called experimental result facts). The database includes **1,541** unique datasets, **1,685** unique methods, and **450** unique metric names. Each dataset, method, and metric has **18.9**, **17.3**, and **64.6** related data records in average, respectively. The associations between the concepts are rich.

4 DEMO SCENARIOS

Tablepedia uses the database to answer the following questions. This is just to show the power of exploring quantitative knowledge in the experimental result database and the usefulness of our approach. Because the database was constructed with only 456 tables, we are **NOT** claiming that the answers to these questions are the truths all over the tons of literature.

***Question 1:* Find related methods, metrics, and datasets.**

Q-1(a) How many methods were used for the Epinions dataset?
select count(distinct Method) from ERD where Dataset="Epinions";
A-1(a) 36. If one uses more SQL queries to look for the detail, one will see the method names such as "UserMean", "ItemMean", "Trust", "NMF", "SVD", "TCF", "PMF", "SoRec", and "RSTE".

Q1(b) How many metrics were used to evaluate on Epinions?
select count(distinct Metric) from ERD where Dataset="Epinions";
A-1(b) 7. More queries will find the concrete metric names such as "F1 score", "Precision", "Recall", "MAE", and "RMSE".

Q1(c) How many datasets used with Epinions in the same table?
select count(distinct Dataset) from ERD where Source=(select (distinct Source) from ERD where Dataset= "Epinions");
A-1(c) 17. The data names are "Amazon", "Ciao", "Douban", and so on. They are popular datasets for evaluating recommender systems.

***Question 2:* Find top-performing methods on a dataset.**

Q2(a) What are the top 3 methods on Epinions in terms of RMSE?
select Method, Score from ERD where Dataset = "Epinions" and Metric = "RMSE" order by Score desc limit 3; // desc is for the fact that a smaller RMSE means a better performance.
A-2(a) "SR2pcc" (1.0954), "SR2vss" (1.0958), "SR1pcc" (1.1013).

Q2(b) What are the top 3 methods on Amazon in terms of F1?
select Method, Score from ERD where Dataset = "Amazon" and Metric = "F1" order by Score limit 3; // Compared to Q2(a), desc was deleted because a bigger F1 means a better performance.
A-2(b) "LEMON" (0.953), "LEMON-auto" (0.91), "LC" (0.815).

***Question 3:* Find conflicting reported numbers.**

Surprisingly, we found a large set of conflicting records in the database. A number of them are worthy of investigation: *First*, as presented in Table 1, the two KDD 2017 papers on multilabel classification, [11] and [9], gave different numbers for the same set of methods, the same datasets, and the same metrics, respectively. Though variance could happen when reproducing the results, we found many of the precision differences are bigger than 3%, which is often a sufficient margin to claim a new achievement! *Second*, if the dataset is Epinions, *plus* the metric is MAE, *plus* the ratio of training data is 80%, then we have three pairs of conflicting numbers reported by [8] and [6]: (1) UserMean: 0.9319 vs 0.9285, (2) ItemMean: 0.9115 vs 0.9913, (3) Trust: 0.9044 vs 0.9215. The same pairs can be observed for RMSE as well. *Finally*, we also find a number of conflicting pairs that were not correctly aligned because of the missing contexts in the extraction such as the ratio of training data and the number of dimensions. In this demo, while showing the importance of integrating PDF tables, we are aware of tons of challenging and interesting future works.

REFERENCES

[1] M. Aristarán. 2013. Tabula-Java. https://github.com/tabulapdf/tabula-java.

[2] Max Göbel, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi. 2012. A methodology for evaluating algorithms for table understanding in PDF documents. In *Proceedings of the 2012 ACM symposium on Document engineering*. ACM, 45–48.

[3] Huan Gui, Jialu Liu, Fangbo Tao, Meng Jiang, Brandon Norick, Lance Kaplan, and Jiawei Han. 2017. Embedding learning with events in heterogeneous information networks. *IEEE TKDE* 29, 11 (2017), 2428–2441.

[4] Meng Jiang, Christos Faloutsos, and Jiawei Han. 2016. Catchtartan: Representing and summarizing dynamic multicontextual behaviors. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 945–954.

[5] Meng Jiang, Jingbo Shang, Taylor Cassidy, Xiang Ren, Lance M Kaplan, Timothy P Hanratty, and Jiawei Han. 2017. Metapad: Meta pattern discovery from massive text corpora. In *KDD*. ACM, 877–886.

[6] Hao Ma, Irwin King, and Michael R Lyu. 2011. Learning to recommend with explicit and implicit social relations. *ACM TIST* 2, 3 (2011), 29.

[7] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *WSDM*. 287–296.

[8] Hao Ma, Tom Chao Zhou, Michael R Lyu, and Irwin King. 2011. Improving recommender systems by incorporating social contextual information. *ACM TOIS* 29, 2 (2011), 9.

[9] Yukihiro Tagami. 2017. AnnexML: Approximate nearest neighbor search for extreme multi-label classification. In *KDD*. 455–464.

[10] Yingchen Yang and Wo-Shun Luk. 2002. A framework for web table mining. In *Proceedings of the 4th international workshop on Web information and data management*. 36–42.

[11] Ian EH Yen, Xiangru Huang, Wei Dai, Pradeep Ravikumar, Inderjit Dhillon, and Eric Xing. 2017. Ppdsparse: A parallel primal-dual sparse method for extreme classification. In *KDD*. 545–553.

[12] Burcu Yildiz, Katharina Kaiser, and Silvia Miksch. 2005. pdf2table: A method to extract table information from pdf files. In *IICAI*. 1773–1785.

CityFlow: A Multi-Agent Reinforcement Learning Environment for Large Scale City Traffic Scenario

Huichu Zhang
Shanghai Jiao Tong University
Shanghai, China
zhc@apex.sjtu.edu.cn

Siyuan Feng
Shanghai Jiao Tong University
Shanghai, China
hzfengsy@sjtu.edu.cn

Chang Liu
Shanghai Jiao Tong University
Shanghai, China
only-changer@sjtu.edu.cn

Yaoyao Ding
Shanghai Jiao Tong University
yyding@sjtu.edu.cn

Yichen Zhu
Shanghai Jiao Tong University
zyc_IEEE@sjtu.edu.cn

Zihan Zhou
Shanghai Jiao Tong University
footoredo@sjtu.edu.cn

Weinan Zhang*
Shanghai Jiao Tong University
wnzhang@sjtu.edu.cn

Yong Yu
Shanghai Jiao Tong University
yyu@apex.sjtu.edu.cn

Haiming Jin
Shanghai Jiao Tong University
jinhaiming@sjtu.edu.cn

Zhenhui Li†
Pennsylvania State University
State College, Pennsylvania, USA
jessieli@ist.psu.edu

ABSTRACT

Traffic signal control is an emerging application scenario for reinforcement learning. Besides being as an important problem that affects people's daily life in commuting, traffic signal control poses its unique challenges for reinforcement learning in terms of adapting to dynamic traffic environment and coordinating thousands of agents including vehicles and pedestrians. A key factor in the success of modern reinforcement learning relies on a good simulator to generate a large number of data samples for learning. The most commonly used open-source traffic simulator SUMO is, however, not scalable to large road network and large traffic flow, which hinders the study of reinforcement learning on traffic scenarios. This motivates us to create a new traffic simulator CityFlow with fundamentally optimized data structures and efficient algorithms. CityFlow can support flexible definitions for road network and traffic flow based on synthetic and real-world data. It also provides user-friendly interface for reinforcement learning. Most importantly, CityFlow is more than twenty times faster than SUMO and is capable of supporting city-wide traffic simulation with an interactive render for monitoring. Besides traffic signal control, CityFlow could serve as the base for other transportation studies and can create new possibilities to test machine learning methods in the intelligent transportation domain.

*Corresponding author
†Corresponding author

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '19, May 13–17, 2019, San Francisco, CA, USA
© 2019 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3314139

CCS CONCEPTS

• **Computing methodologies** → **Multi-agent systems**; **Simulation environments**; • **Applied computing** → **Transportation**.

KEYWORDS

Reinforcement Learning Platform; Microscopic Traffic Simulation; Mobility

ACM Reference Format:
Huichu Zhang, Siyuan Feng, Chang Liu, Yaoyao Ding, Yichen Zhu, Zihan Zhou, Weinan Zhang, Yong Yu, Haiming Jin, and Zhenhui Li. 2019. CityFlow: A Multi-Agent Reinforcement Learning Environment for Large Scale City Traffic Scenario. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308558.3314139

1 INTRODUCTION

Traffic signal control problem, one of the biggest urban problems, is drawing increasing attention in recent years [5, 9, 10]. Recent advances are enabled by large-scale real-time traffic data collected from various sources such as vehicle tracking device, location-based mobile services, and road surveillance cameras through advanced sensing technology and web infrastructure. Traffic signal control is interesting but complex because of the dynamics of traffic flow and the difficulties to coordinate thousands of traffic signals. Reinforcement learning becomes one of the promising approaches to optimize traffic signal plans, as shown in several recent studies [5, 9, 10]. At the same time, traffic signal control is also one of the major real-world application scenarios for reinforcement learning [6].

To successfully deploy reinforcement learning technique for traffic signal control, the traffic simulator becomes the most important factor. Because the learning method relies on a large set of data

samples. These data samples can hardly be collected from the real world directly. Aside from the consequence of bad decisions, a city simply cannot generate enough data samples for learning. If we treat each minute as a data sample, a city can only generate 1,440 (24 hours by 60 minutes) data samples in a day. Such a small size of sample is not enough to train a deep reinforcement learning model to be powerful enough to make good decisions. Thus, it becomes crucial to have a simulator that is fast enough to generate a large set of data samples.

The most popular public traffic simulator SUMO [7] (Simulation of Urban Mobility) has been frequently used in many recent studies. SUMO, however, is not scalable to the size of the road network and the size of traffic flow. For example, it can only perform around three simulation steps per second on a 30×30 grid with tens of thousands of vehicles, the situation is even worse if we use the python interface to get information about the system to support reinforcement learning. A city, however, is often at the size of a thousand intersections (e.g. there are 30×40 intersections of major roads in Hangzhou, China) and hundreds of thousands vehicles, which is beyond the current simulation capacity of SUMO.

To enable the reinforcement learning for intelligent transportation, we create a traffic simulator CityFlow[1], which can be scaled to support the city-wide traffic simulation. One of the major improvements over SUMO is that CityFlow enables multithreading computing. To the best of our knowledge, this is the first open-source simulator that can support city-wide traffic simulator. CityFlow is flexible to define road network, vehicle models, and traffic signal plans. It is more than twenty times faster than SUMO. We have also provided friendly interface for reinforcement learning testbed. We plan to demonstrate these functions at the demo session.

Finally, our scalable traffic simulator CityFlow will open many new possibilities besides traffic signal control scenario. First, it could support various large-scale transportation research studies, such as vehicle routing through mobile app, traffic jam prevention. Second, similar to OpenAI Gym[2] which provides a set of benchmark environments for reinforcement learning, CityFlow could serve as a benchmark reinforcement learning environment for transportation studies. Besides traffic signal control, reinforcement learning has been used in transportation studies such as taxi dispatching [12] and mixed autonomy systems [11]. But all the existing studies either use SUMO or over-simplified traffic simulator. Third, we plan to better calibrate the simulation parameters by learning from real-world observations. This will make the simulator not only generate data samples fast but also generate "real" data samples.

2 BRIEF DESCRIPTION

2.1 System Design

CityFlow is a microscopic traffic simulator which simulates the behavior of each vehicle at each time step, providing highest level of detail in the evolution of traffic. However, microscopic traffic simulators are subject to slow simulation speed [13]. Unlike SUMO, CityFlow uses multithreading to accelerate the simulation. Data structure and simulation algorithm are also optimized to further speedup of the process.

[1]https://github.com/cityflow-project/CityFlow/
[2]https://gym.openai.com/

2.1.1 Road Network. Road network is the basic data structure in CityFlow. **Road** represents a directional road from one **intersection** to another **intersection** with road-specific properties. A **road** may contain multiple **lanes**. Each **lane** holds a Linked List of vehicles. Linked List supports fast insertion and searching of leading vehicles. **Segments** are small fragments of a **lane**. We design segments in order to efficiently find all vehicles within a certain range of the lane. This structure is crucial for fast lane change operation. **Intersection** is where roads intersects. An **intersection** contains several **roadlinks**. Each roadlink connects two roads of the intersection and can be controlled by traffic signals. A **roadlink** contains several **lanelinks**. Each lanelink represents a specific path from one lane of incoming road to one lane of outgoing road. **Cross** represents the cross point between two lanelinks. This structure is crucial for fast intersection logic.

2.1.2 Car Following Model. The car-following model is the core component CityFlow. It computes the desired speed of each vehicle at next step using information like traffic signal, leading vehicles, etc. and ensures that no collisions occur in the system. Currently, the car following model used in CityFlow is a modification of the model proposed by Stephen Krauß [4]. The key idea is that: the vehicle will drive as fast as possible subject to perfect safety regularization (e.g. being able to stop even if leading vehicle stops using maximum deceleration). Unlike SUMO [7], we use ballistic position update rule instead of Euler position update. Ballistic update yields more realistic dynamics for car-following models based on continuous dynamics especially for larger time-steps (e.g. 1 second) [8].

Basically, vehicles are subject to several speed constraints, maximum speed which meets all these constraints will be chosen. Currently, following constraints are considered:

- vehicle and driver's maximum acceleration
- road speed limit
- collision free following speed
- headway time following speed
- intersection related speed

Due to page limit, we only present the detail of collision free speed computation. It takes v_F current speed of following vehicle, v_L current speed of leading vehicle, d_F maximum deceleration of following vehicle, d_L maximum deceleration of leading vehicle, gap current gap between two vehicles, $interval$ the length of each time step as parameters and compute the no-collision-speed s by solving a quadratic equation using equation 1.

$$
\begin{aligned}
c &= \frac{v_F \cdot interval}{2} - \frac{v_L^2}{2 \cdot d_L} - gap \\
a &= \frac{1}{2 \cdot d_F} \\
b &= \frac{interval}{2} \\
s &= \frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}
\end{aligned} \tag{1}
$$

Intersection related speed is handled by intersection logic and is illustrated in the next section.

2.1.3 Intersection Logic. The behavior of vehicles in intersection is complex and it requires careful design to efficiently mimic real

world behavior [1, 3]. Basically, vehicles in intersection should obey following two rules:

- fully stop at red signal, stop if possible at yellow signal
- yield to vehicles with higher priority (e.g. turning vehicles should yield straight-moving vehicles)

To avoid collisions at intersection, it is non-trivial to check if there are vehicles on the opposite lane. The simplest method is to use brute force search to find all vehicles within a certain range and check if they will collide within a certain time period. But this method is very time consuming. Instead, we precompute all the cross points between lanelinks in intersection. When a vehicle approaches the intersection, it will notify all cross points in the intersection about its arrival. The cross points is responsible for deciding which vehicle could pass and which vehicle should yield. The time complexity of our algorithm is $O(N_{crosspoints})$. Due to page limit, we omit the detail of our algorithm.

2.1.4 Lane Change Model. Lane change model addresses two questions for a vehicle: when and how to change lane. Vehicles may change lanes when there are more free space on adjacent lanes or a lane change is required to follow its route. Notice that it is slow to traverse all vehicles in adjacent lanes. Instead, by maintaining the vehicle information in **segments** which are small fragments of each lane, we only need to search for related vehicles in adjacent segments in constant time (up to three segments for each lane), which largely reduce time complexity.

When a vehicle decides to change lane, it needs to find a way to notify other vehicles. Here we use a similar mechanism in SUMO. When a vehicle changes lane, the simulation engine will put a copy of it to its destination lane, called shadow vehicle. A shadow vehicle has the same function as a normal vehicle, and it can become the leader of other vehicles in the car following model. The vehicle and its shadow moves consistently, which is guaranteed by the simulation engine in the way that their speed constraints will be applied to each other. After the lane change finishes, the simulation engine will just remove the original vehicle and let its shadow vehicle replace it.

2.2 Python Interface

In order to support multi-agent reinforcement learning, we provide a python interface via *pybind11* [2]. User can perform simulation step by step and get various kinds of information about current state, e.g. number of vehicles on lane, speed of vehicles. Besides, we provide interface to control the elements in simulator at each time step. Currently, users can control traffic signals and add vehicles on-the-fly. We plan to support more types of controlling functions such as vehicle behavior control and road property control in the future. Below is a sample usage of python interface.

Listing 1: usage of python interface

```
import engine
eng = engine.Engine(config_file)
phase = [...] # the traffic signal phase of
    each time step
for step in range(3600):
    eng.set_tl_phase("intersection_1_1",
        phase[step])
    eng.next_step()
    eng.get_current_time()
    eng.get_lane_vehicle_count()
    eng.get_lane_waiting_vehicle_count()
    eng.get_lane_vehicles()
    eng.get_vehicle_speed()
    # do something
```

2.3 Frontend

We provide a web-based Graphic User Interface. User can check the replay output by the simulator. In order to support viewing large-scale simulation, we use *WebGL*-based library *PixiJS*[3] for fast rendering of vehicles and traffic signals. Figure 1 shows some screenshots of the GUI under several scenarios.

3 PERFORMANCE

3.1 Efficiency

We compare the performance between SUMO and CityFlow under different scenarios. The experiment runs on Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz. As Figure 2 shows, CityFlow outperforms SUMO in all scenarios from small traffic to large traffic with single thread. The speedup is even more significant with more threads. We achieve about 25 times speedup on large scale 30×30 road networks with tens of thousands of vehicles using 8 threads, which is 72 steps of simulation per second. Besides, CityFlow shows better efficiency when retrieving information of the simulation via python interface. This is mainly because SUMO uses socket for interaction while CityFlow uses *pybind11* for seamless C++ and python integration.

3.2 Effectiveness

We evaluate the effectiveness of CityFlow by comparing to SUMO because SUMO is already a widely-used traffic simulator and its effectiveness is acceptable by domain experts. We compare the average duration of vehicles (time for a vehicle to enter and leave the road network) under different traffic volume settings. As Table 1 shows, the difference is within reasonable range.

Table 1: Duration of vehicles under different traffic volume

Vehicles/Hour	100	200	300	400	500
SUMO	40.76	41.57	42.75	44.08	45.93
CityFlow	40.79	41.58	42.62	43.84	45.45
Difference	0.07%	0.04%	0.30%	0.54%	1.06%

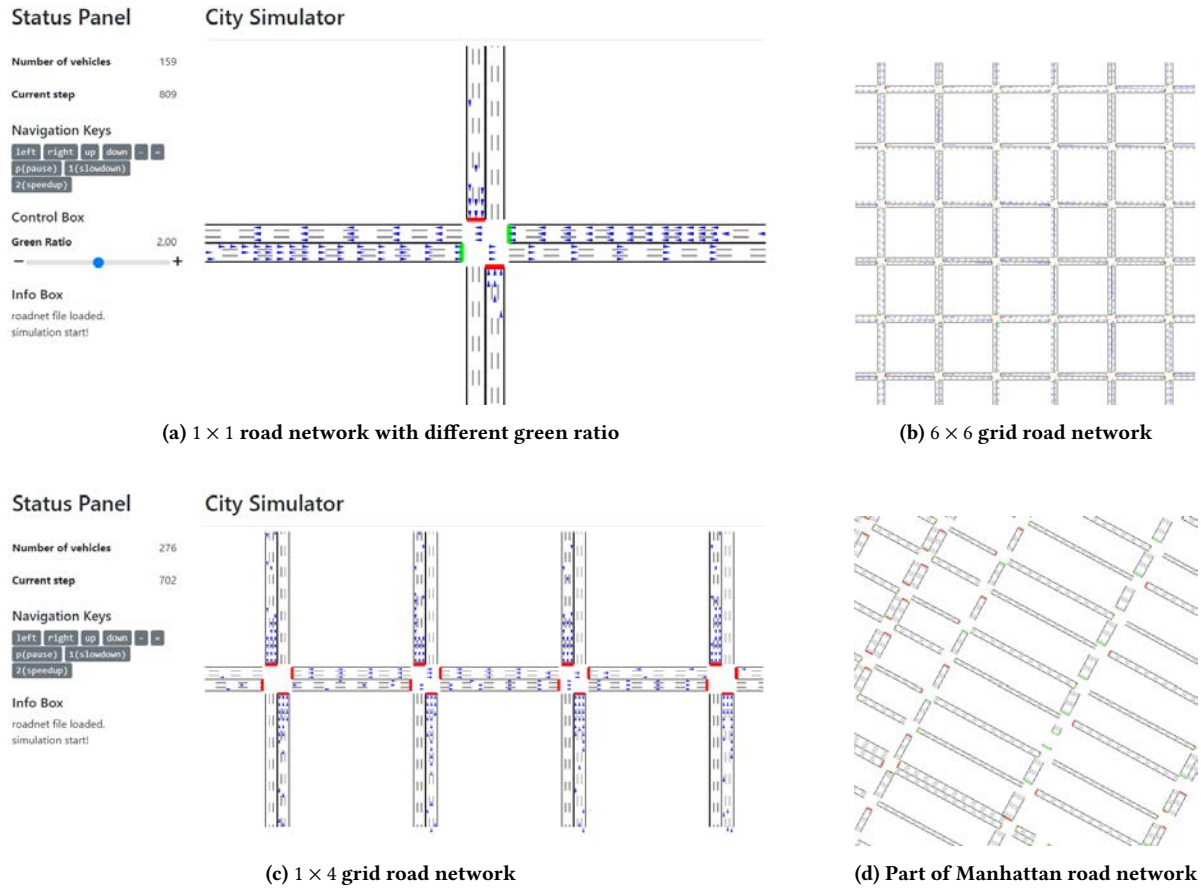
4 DEMO DETAIL

We plan to demonstrate CityFlow in different traffic scenarios and show its capability to serve as reinforcement learning testbed.

The demo consists of following parts:

- Simulating traffic in various scenarios, from synthetic grid scenarios to real world scenarios, and from small road networks with dozens of vehicles to large scale networks with tens of thousands of vehicles.

[3] https://github.com/pixijs/pixi.js



(a) 1×1 **road network with different green ratio**

(b) 6×6 **grid road network**

(c) 1×4 **grid road network**

(d) Part of Manhattan road network

Figure 1: Screenshot of CityFlow in different scenario

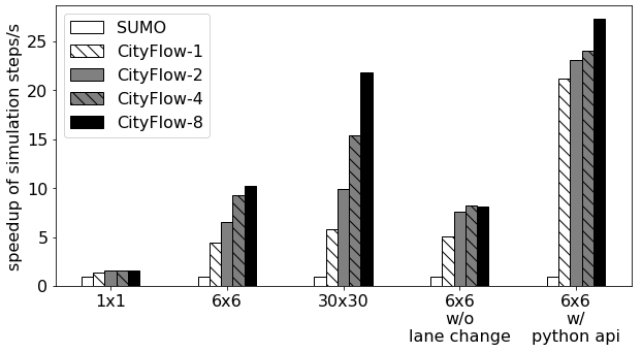


Figure 2: Speedup of CityFlow compared to SUMO

- Show the effectiveness the car-following model, intersection logic and lane change behavior of the simulator.
- Show a complete reinforcement learning training episode of optimizing traffic signal plan. Participants can observe gradual improvement of traffic condition during the training.
- Demo participants can control cycle length, green ratio of traffic signal and change the volume of traffic and see instant feedback of how the traffic condition would change.

We have published a video on Youtube[4], which demonstrate the expected effect. The project is under active development and we are likely to add other features (e.g. more map options, vehicle controls) and demonstrate more functions at the conference.

No special hardware is required since we are demonstrating a software project (learning platform). We will bring our laptop. It would be great if a monitor is provided.

5 SUMMARY

We propose CityFlow, an efficient, multi-agent reinforcement learning environment for large scale city traffic scenario. Researchers can use it as a testbed for traffic signal control problem and conduct research on urban mobility. We will demonstrate the usage and some results of RL-controlled traffic signal plan. Also, we are actively developing the project and plan to support more RL scenarios like dynamic vehicle routing, policy of reversible lane or limited lane as well as open source the project in the near future.

[4]https://youtu.be/qeE4hRmWONM

REFERENCES

[1] Martin Fellendorf and Peter Vortisch. 2010. Microscopic traffic flow simulator VISSIM. In *Fundamentals of traffic simulation*. Springer, 63–93.

[2] Wenzel Jakob, Jason Rhinelander, and Dean Moldovan. 2017. pybind11 – Seamless operability between C++11 and Python. https://github.com/pybind/pybind11.

[3] Daniel Krajzewicz and Jakob Erdmann. 2013. Road intersection model in SUMO. In *1st SUMO User Conference-SUMO*, Vol. 21. 212–220.

[4] Stefan Krauß. 1998. *Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics*. Ph.D. Dissertation. Universitat zu Koln.

[5] Li Li, Yisheng Lv, and Fei-Yue Wang. 2016. Traffic signal timing via deep reinforcement learning. *IEEE/CAA Journal of Automatica Sinica* 3, 3 (2016), 247–254.

[6] Yuxi Li. 2017. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274* (2017).

[7] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. 2018. Microscopic Traffic Simulation using SUMO, In The 21st IEEE International Conference on Intelligent Transportation Systems. *IEEE Intelligent Transportation Systems Conference (ITSC)*. https://elib.dlr.de/124092/

[8] Martin Treiber and Venkatesan Kanagaraj. 2015. Comparing numerical integration schemes for time-continuous car-following models. *Physica A: Statistical Mechanics and its Applications* 419 (2015), 183–195.

[9] Elise Van der Pol and Frans A Oliehoek. 2016. Coordinated deep reinforcement learners for traffic light control. *Proceedings of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016)* (2016).

[10] Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. 2018. IntelliLight: A Reinforcement Learning Approach for Intelligent Traffic Light Control. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. 2496–2505.

[11] Cathy Wu, Aboudy Kreidieh, Eugene Vinitsky, and Alexandre M Bayen. 2017. Emergent Behaviors in Mixed-Autonomy Traffic. In *Conference on Robot Learning*. 398–407.

[12] Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping Ye. 2018. Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms: A Learning and Planning Approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 905–913.

[13] Derek Yin and Tony Qiu. 2011. Comparison of macroscopic and microscopic simulation models in modern roundabout analysis. *Transportation Research Record: Journal of the Transportation Research Board* 2265 (2011), 244–252.

Author Index

Aakash 1601
Abdelzaher, Tarek 296, 2192
Aberer, Karl 1544, 1747
Abu-El-Rub, Noor 2529
Acharyya, Rupam 2270
Adelani, David Ifeoluwa 2056
Agarwal, Aman 4
Aghakhani, Hojjat 3108
Agichtein, Eugene 2799
Agrawal, Rakshit 15, 2472, 3441
Ait-Mlouk, Addi 3595
Akoglu, Leman 928
Alambo, Amanuel 514
Alay, Ozgu 1532
Albo, Sefi 1532, 2799
Alimpertis, Emmanouil 2536
Almeida, Jussara 818
Alorainy, Wafa 3006
Alrashed, Tarfah 26
Al-Rfou, Rami 37
Altenburger, Kristen M. 2543
Althoff, Tim 571
Alzaidy, Rabah 2551
An, Bo 616
An, Sungtae 2558
An, Weiting 60
Anand, Avishek 2736
Anderson, Ashton 1954
Ansah, Jeffery 2565
Antonakakis, Manos 840
Arora, Akhil 1886
Artzi, Yoav 3237
Atreja, Shubham 2943
Aucinas, Andrius 526
Auer, S??ren 3574
Awadallah, Ahmed Hassan 1245
Ayoade, Gbadebo 2992
Azcorra, Arturo 1454
Badjatiya, Pinkesh 49
Bagchi, Amitabha 2258
Bai, Jinze 60
Bai, Tian 72, 840
Bailey, Peter 26, 2572
Balashankar, Ananth 1702
Balasubramaniam, Thirunavukarasu 3093
Balsamo, Duilio 2572
Bao, Forrest Sheng 2630
Barría Pineda, Jordan 2068
Bashardoust, Ashkan 480
Basu Roy, Senjuti 1612
Batra, Nipun 2872
Belletti, Francois 1782
Bendersky, Michael 4, 795, 2785
Bendersky, Mike 627
Benevenuto, Fabrício 818
Bensaid, Eden 3550
Benson, Austin 94, 1409
Benzi, Kirell 1290
Bernaschi, Massimo 105
Bernstein, Abraham 2366
Berti-Equille, Laure 2580
Bertino, Elisa 594
Besbes, Omar 1713, 1782
Bhat, Suma 2771
Bhatia, Gurpreet Singh 3086
Bhattacharya, Arnab 1886
Bhowmik, Rajarshi 116
Bi, Jingping 273
Bian, Yuchen 2137
Biega, Asia J. 2623
Bigham, Jeffrey 3187
Bigham, Jeffrey P. 549
Bing, Lidong 1006, 2608, 3503
Bonifati, Angela 127
Borisov, Nikita 840
Bourgaux, Camille 1465
Bourgeois, Dylan 1544
Boyd, Danah 480, 1365, 1521
Bright, Travis 2601, 2587
Brodowsky, Bernhard 627
Brost, Brian 2594
Brunstrom, Anna 1532, 2068
Buchanan, Erik 3116
Budak, Ceren 139, 3137
Bukhari, Syed Ahmad Chan 1623
Burgess, Matthew 1365
Burnap, Pete 3006
Bursztein, Elie 2601
Butts, Carter 2536
Cadamuro, Gabriel 2608
Cai, Deng 3455
Cai, Qingpeng 1421
Callan, Jamie 340
Callejo, Patricia 1454
Cao, Jiannong 1669, 2722
Cao, Yanan 438
Cao, Yixin 151
Cao, Yue 1210
Caragea, Cornelia 1555, 1829, 2551
Caragea, Doina 1555, 2701
Carmel, David 162
Carrington, Patrick 549
Cassidy, Cameron 549, 1747
Castro, Ignacio 729
Catarino, Ana 3483, 3503
Celestini, Alessandro 105
Cen, Lei 2922
Chae, Dong-Kyu 2616
Chakraborty, Abhijnan 2623
Chakraborty, Sandip 1657
Chan, T-H. Hubert 173
Chandar, Praveen 2971
Chandra, Swarup 503
Chang, Chun-Ming 2814
Chang, Jonathan 184
Chang, Shuhao 3405
Chang, Yi 594, 1443
Chattopadhyay, Tithi 196
Chavan, Prithviraj 1601, 3493
Chawla, Nitesh 717
Chelmis, Charalampos 3427
Chen, Cen 2630
Chen, Changhao 1175
Chen, Chih-Ming 2637
Chen, Chun 228
Chen, Chung-Chi 3498
Chen, Enhong 2312
Chen, Francine 3101

Chen, Gong 207
Chen, Guihai 2091
Chen, Hsin-Hsi 3498
Chen, Huajun 2366
Chen, Huiyuan 218
Chen, Hung-Fang 2814
Chen, Jenny 2169
Chen, Jiaoyan 2366
Chen, Jiawei 228
Chen, Ken 2644
Chen, Li 240
Chen, Liheng 2644
Chen, M.K.Sophie 2652
Chen, Minmin 1782
Chen, Qun 3526
Chen, Weiteng 2483
Chen, Xiangning 491
Chen, Xiaojun 2865
Chen, Yan-Ying 3101
Chen, Yatong 263
Chen, Yingzhi 1119
Chen, Yongjun 2743, 3215
Chen, Yu-Hua 3531
Chen, Yu-Jen 2814
Chen, Yujun 2659
Chen, Zhaoqiang 3526
Chen, Zhenpeng 251
Chen, Zhenzhong 3020
Chen, Zhumin 1095
Chen, Zi-Yuan 3531, 2666
Cheng, Denny 2666
Cheng, Hong 972
Cheng, Xueqi 583, 1198, 3335
Cheng, Yun 1017
Cheng, Ziqiang 2666
Chiang, Chun-Wei 3187
Chin, Alex 263
Choi, Jaeho 2616, 2687
Chu, Cuong Xuan 2673
Chu, Xianqi 2985
Chu, Xiaokai 273
Chua, Tat-Seng 151, 2401
Clarke, Einat 2601
Coey, Dominic 285
Cohn, Trevor 1028
Cong, Gao 1017
Conrads, Felix 1623
Copeland, John 207
Counts, Scott 1061
Cox, Ingemar 1487, 2505
Cranshaw, Justin 1061
Crescenzi, Valter 605
Cudre-Mauroux, Philippe 1398, 2147, 2158
Cuevas, Ángel 1454
Cuevas, Rubén 1454
Cui, Hang 296
Cui, Peng 2022, 3123
Cui, Zeyu 307
Cui, Zhiming 3398
Cummins, Ronan 3503
Cunha, Tiago 318
Cunningham, Tom 285
Da Hora, Diego 3194
Dai, Quanyu 329
Dai, Shih-Chieh 3536
Dai, Zhuyun 340
Danescu-Niculescu-Mizil, Cristian 184
Dang, Yingnong 2659
Das, Gautam 1612
Datta, Amit 351
De, Pradipta 1657
de Alfaro, Luca 15
Dean, Jeff 1
DeBlasio, Joe 1279
De Cristofaro, Emiliano 1487, 2601, 2169, 2158
de Melo, Gerard 116
de Montjoye, Yves-Alexandre 3550
de Rijke, Maarten 1095, 2103, 2879, 3279
Desir, Antoine 1713
Devabhaktuni, Vijay 829
Dhamnani, Sunny 1601
Diamantaris, Michalis 3063
Diao, Yufeng 363
Diefenbach, Dennis 3215, 3507
Ding, Yaoyao 3620
Do, Huy Hoang 3420
Doerfler, Periwinkle 372, 3521, 3610, 2659
Dong, Yuxiao 1509, 2680
Du, Bowen 1175
Du, Lun 3314
Du, Mengnan 383, 3600, 2687
Du, Nan 3041
Dua, Robin 627
Duan, Xiaoni 2694
Dubuc, Gilles 3194
Dumais, Susan 26
Dunne, Drew 2169
Durmus, Esin 2701
E. Robertson, Ronald 693
Efthymiadis, Kyriakos 1579
Eide, Darrin 2893
Eirinaki, Magdalini 2778, 2957
Eksombatchai, Pong 2236
Elad, Guy 1354
Eliassi-Rad, Tina 2344
Elifff, David M. 2601, 3595
Engels, Gregor 670
Ensafi, Roya 2851, 394, 405
Esfandiari, Hossein 405
Espín-Noboa, Lisette 2708
Estrin, Deborah 2169, 3493
Evensen, Sara 3511, 3503
Fan, Changjie 3455, 2715, 3041
Fan, Weiguo 3441
Fan, Wenqi 417
Fan, Xiaochao 363
Fan, Xinxin 273, 1965, 3314, 3448
Fan, Zhen 340
Fang, Lanting 427
Fang, Zheng 438
Farooqi, Shehroze 448
Feamster, Nick 196
Feng, Aaron 3511, 2715
Feng, Fuli 491
Feng, Jie 459, 1771
Feng, Kaiyu 427
Feng, Siyuan 3620
Feng, Yan 228
Feng, Yunhe 2722
Feng, Zhe 2729

Feng, Zhitao 2290
Fetahu, Besnik 906, 1567, 2736, 3564
Fida, Mah-Rukh 1532
Findlater, Leah 1724, 3590
Fish, Benjamin 480, 2056
Flores, Huber 2517
Flores, René D. 2907, 3258
Fourney, Adam 874, 3590
Fox, Edward 3441
Freire, Juliana 1476
Friedler, Sorelle 480
Fu, Zuohui 2757
Fujita, Katsuhide 3545
Fung, Pascale 605, 1222
Fyodorov, Yaroslav 162
Gómez, Vicenç 2764
Gad-Elrab, Mohamed H. 3516
Gajda, Przemek 627
Gandhi, Anshul 1759
Ganguly, Niloy 1657
Gao, Chen 491, 2659
Gao, Han 2091, 2743, 3041
Gao, Jinhua 3335
Gao, Jun 60, 2750, 3049, 2091
Gao, Yang 503, 1142, 2992
Garcia-Gathright, Jean 1376, 2971, 3579
Gaur, Manas 514
Ge, Yingqiang 2757
Geng, Shijie 2757, 2764
Geyik, Sahin 3116
Ghasemisharif, Mohammad 526, 538
Ghosh, Anurag 3086
Ghoting, Amol 1679
Giles, C. Lee 2551
Gionis, Aristides 1387
Gleason, Cole 549, 739, 1909
Glenski, Maria 560
Gligoric, Kristina 681, 2301
Goga, Oana 1920
Golbandi, Nadav 795
Gong, Boqing 1806, 2771
Gordon, Mitchell 571
Gottlob, Georg 605, 2258
Goyal, Vineet 1713
Grabowicz, Przemyslaw 2056
Grasso, Giovanni 605
Graux, Damien 3574
Gregor, Franz 3564
Grusky, Max 3237
Guan, Saiping 583
Guarino, Stefano 105
Guennemann, Stephan 1310
Guille, Adrien 2587
Gulati, Avni 2778
Gummadi, Krishna P. 1920, 2623
Gunasinghe, Hasini 594
Gunathillake, Ashanie 3165, 2929
Guo, Huifeng 1119
Guo, Jinsong 605, 2715
Guo, Minyi 2000, 3307
Guo, Qingyu 616
Guo, Teng 2290
Guo, Weidong 2792, 1988
Gupta, Jai 2785
Gupta, Kshitiz 1245
Gupta, Manish 49, 2915
Gurevych, Iryna 3179, 3108
Guy, Ido 1354
H. Chi, Ed 1782
H. Son, Sang 2011
Haghdan, Maysam 829, 2056
Halevy, Alon 3511
Han, Fred.X 2792, 2192, 3420
Han, Jizhong 2844
Han, Shuguang 627
Han, Weihao 2985
Hancock, Braden 638
Harel, Shahar 2799
Hartman, Timo 2056
Hashemian, Golli 1256
Hasthanasombat, Apinan 648
Havens, Timothy 2957
He, Dongxiao 2886
He, Huan 659, 3521, 3610, 2091, 1421
He, Suining 2011, 2806
He, Xiangnan 151, 491
He, Xiaofeng 1965, 3215
He, Yancheng 1106, 2792
He, Yeye 2413
He, Yuan 417
Hecht, Brent 1931
Heidari, Hoda 2623
Heindorf, Stefan 670
Helic, Denis 1634
Henderson, Jette 659, 2837, 2957
Ho, Chien-Ju 1794
Ho, Daniel E. 2543
Ho, Joyce C 659, 3130
Honavar, Vasant 2907, 3610
Horta Ribeiro, Manoel 681
Hosey, Christine 2971
Hou, Boyi 3526, 2290
Hou, Shifu 3448
Houben, Geert-Jan 1521
Howe, Bill 3605
Hsiao, Hsu-Chun 2814
Hsu, Chao-Chun 3531, 2601
Hsu, Ting-Lun 3536
Hu, Aiqun 427
Hu, Desheng 693
Hu, Guangneng 2822
Hu, Shaohan 2192
Hu, Songlin 2844
Hu, Tianran 705
Hu, Xia 383, 3600
Hu, Xinghai 3020
Hu, Zhiyuan 2929
Hu, Zikun 151
Hu, Ziniu 251, 2830, 717
Huang, Danny Yuxing 196
Huang, Heng 2045
Huang, Hen-Hsen 3498
Huang, Heyan 1142
Huang, Hongjia 3405
Huang, Jiaming 616
Huang, Jianhui 273
Huang, Jiantao 2886, 2630
Huang, Junzhou 972
Huang, Longtao 2844
Huang, Minlie 1771, 2033, 2401, 2448

Huang, Qinghua....3349
Huang, Ting-Hao....1164, 3531
Huang, Wenbing....972
Huang, Xuanjing....2425, 3013, 3342, 2886
Hui, Pan....2517
Hui, Pengrui....616
Huo, Chengfu....1028, 2355
Ibosiola, Damilola....729
Ibrahim, Akil Arif....1601
Ibrahim, Rania....739
Ibrahim, Yusra....3541
Ida, Makiko....2851, 3545
Immorlica, Nicole....751, 762
Ioannidis, Sotiris....3063
Iqbal, Umar....2483
Islam, M. Raihanul....3079
Iwasa, Kosui....3545
Iyengar, Garud....1713
Jain, Mohit....2943
Jain, Sagar....1782, 3550
Jang, Myung-Hwan....807
Jehan, Tristan....2594, 2814
Jenkins, Porter....3300
Jha, Devki Nandan....3555
Ji, Donghong....3391
Ji, Houye....2022
Ji, Shenggong....773
Ji, Shiyu....2858
Ji, Shouling....2964
Ji, Shuiwang....383, 2743, 3600
Ji, Yimu....3349
Jia, Shengbin....2865
Jia, Yiling....2872
Jia, Yuting....784
Jiang, Jyun-Yu....795
Jiang, Lili....3123, 3328, 3595, 3615
Jiang, Shan....693
Jiang, Shaojie....2879
Jiang, Wenjun....2192
Jiang, Yu....372, 983, 2886, 3560
Jin, Depeng....459, 491, 2886
Jin, Haiming....3620
Jin, Woojeong....1210
Jin, Xiaolong....583
Jing, Xiao-Yuan....3349
Jo, Yong-Yeon....807
Jourjon, Guillaume....3165
Junuthula, Ruthwik....318, 829, 2056
Kaafar, Mohamed Ali....2851
Kakade, Sham....2608
Kamar, Ece....874, 1175, 2893
Kang, Jin-Soo....2616, 2565, 2659
Kaplan, Lance....296
Kapravelos, Alexandros....3244
Karunanayake, Naveen....3165, 3223
Kautz, Henry....2270
Kavuluru, Ramakanth....514
Kawamae, Noriaki....470, 1691, 2900
Kenney, Doug....1256
Kern, Roman....1634
Khademi, Aria....2907
Khan, Latifur....503
Kharraz, Amin....840
Khatouni, Ali Safari....1532
Khattar, Dhruv....2915
Kim, Kang-Min....853
Kim, Kwanwoo....2225
Kim, Myunghwan....1679
Kim, Sang-Wook....807, 2616
Kim, Seunghyeon....863
Kim, Tiffany Hyun-Jin....2814
Kim, Yeachan....853
Kim, Yongdae....2225
Kim, Yongsung....874
Kitani, Kris M....549
Kleinberg, Jon....94
Knoblock, Craig....1944
Kobayashi, Tetsunori....3187
Kocayusufoglu, Furkan....886
Koeune, François....1579
Kokkodis, Marios....896
Kong, Deguang....2922
Kong, Fanshuang....2929, 1841
Koone, Mary E....1612
Koopman, Bevan....1646
Korlakai Vinayak, Ramya....2608
Kosta, Sokol....1590
Kotila, Mikko....1454
Kourtellis, Nicolas....1432, 2736
Koutsoupias, Elias....917, 906
Krawczyk, Hugo....594, 3108
Ku, Lun-Wei....3531, 3536
Ku, Yu-Ming....2814
Kuhlman, Caitlin....2936
Kumar, Shanu....2943
Kumar, Suhansanu....3208
Kumaraguru, Ponnurangam....3086
Kundu, Ashish....594
Kursuncu, Ugur....514
Kutlu, Mucahid....3158
Kuzi, Saar....162
Kwashie, Selasie....2565
Lagerspetz, Eemil....2517
Lai, Kunfeng....1106, 2792
Lalmas, Mounia....1256, 2680
Lam, Wai....1006
Lampos, Vasileios....1487, 2505, 3349, 3244
Latifur, Khan....2992
Lauw, Hady....1864
Lavi, Ron....2950
Lazos, Philip....917
Le, Huyen....2957
Le, Tuan....928, 3158
Lee, Chia-Jung....26
Lee, Dongwon....2137
Lee, Hongrae....638
Lee, Ji-Min....853
Lee, Jongwuk....863
Lee, Jungho....853
Lee, Kyumin....1841
Lee, Sanghack....2907
Lee, Sangkeun....853
Lei, Chenyi....2964
Le Mens, Gaël....2764
Lemmerich, Florian....2708
Le Quoc, Do....3564
Leskovec, Jure....571, 2236, 2999
Lessig, Lawrence....2
Lever, Charles....840
Li, Ang....2971
Li, Changchun....939
Li, Cheng....4, 795
Li, Chenliang....3391

Li, Dingcheng 950
Li, Feifei..... 961, 1988, 2687, 2830, 2659
Li, Jia 972
Li, Jian..... 1509
Li, Jinfeng 3511
Li, Jing 218, 3391
Li, Jinyang 2192
Li, Jiuyong 2565
Li, Jixue 2565, 3434
Li, Liangda..... 2978, 3215
Li, Minne..... 983
Li, Ning..... 1164
Li, Pengfei..... 995
Li, Piji..... 1006
Li, Ping 950, 2715, 3328
Li, Qian..... 438
Li, Qiang..... 329
Li, Qing..... 417
Li, Qiuchi 3293
Li, Quanzhi..... 3569
Li, Ruirui 2978
Li, Tianrui..... 773
Li, Vivian..... 3511, 2000, 3307, 2630
Li, Ximing..... 939, 3448
Li, Xiucheng..... 1017
Li, Xue 2985
Li, Yaliang..... 3041
Li, Yandong..... 1806
Li, Yanen 2080
Li, Yi-Fan.... 503, 2992, 459, 491, 3363, 3405, 1028, 2771, 3511
Li, Yusen..... 1153
Li, Zekun..... 307
Li, Zhanhuai..... 3526
Li, Zhao 60, 616, 2964
Li, Zhenhui..... 2181, 3300, 3620
Li, Zongze..... 3615
Liang, Shangsong..... 1039, 2323, 2333, 3377
Liang, Zhibin..... 173
Liao, Lizi..... 2401
Lillo, Fabrizio 1051, 1256
Lin, Allen Yilun 1061
Lin, Christopher..... 26
Lin, Felix Xiaozhu..... 2125
Lin, Hongfei 363, 2290
Lin, Hongtao 1073
Lin, Hsin Yu..... 3531
Lin, Junyu 2844
Lin, Qingwei..... 2659
Lin, Wenqing..... 1084, 3123
Lin, Xinyi 2652
Lin, Xuemin..... 1130
Lin, Yujie 1095
Lin, Yu-Ru..... 2068
Linder, Rhema..... 3600
Ling, Huang 83
Lipani, Aldo..... 2323, 2333
Litvak, Nelly..... 3151
Liu, Bang..... 1106
Liu, Bin..... 1119
Liu, Bing..... 3413, 2999
Liu, Boge 1130, 3620
Liu, Dongxin 2192
Liu, Haifeng..... 3215, 3006
Liu, Hongyi 3020
Liu, Huining..... 3511, 2125
Liu, Junxin 3013, 3342
Liu, Lin 2565
Liu, Luyang 1142
Liu, Ninghao 383
Liu, Shang..... 3020
Liu, Shaosu 3034
Liu, Shengzhong..... 2192
Liu, Shuchang 2757
Liu, Shuchun..... 3384
Liu, Tianqiao..... 3027
Liu, Xi 2068
Liu, Xiaoguang 1153
Liu, Xiaozhong 2045, 2425
Liu, Xin..... 3370, 3526
Liu, Xinyu..... 1153
Liu, Xinyue..... 1841
Liu, Xuanzhe..... 251, 1234, 2125
Liu, Xuefeng..... 1669, 3321
Liu, Yanbing 438
Liu, Yanchi..... 3398
Liu, Yiding 2181
Liu, Yifang 3034
Liu, Yiqun 1187, 1210, 1977, 2103
Liu, Yuanqiang..... 2125
Liu, Yunhuai..... 2011
Liu, Yunxin 2125
Liu, Zitao..... 3027
Liu, Zongtao..... 1164, 526
Lofi, Christoph..... 3286
Loizon, Noha..... 2851
Lombardi, Flavio 105
Lu, Chris Xiaoxuan..... 1175
Lu, Hongyu 1187
Lu, Hua..... 995
Lu, Jianan..... 351, 3434
Lu, Xuan..... 251, 2158, 3384, 2722
Lu, Zhicong 3314, 3507
Luo, Dongsheng 2137
Luo, Jiebo 705
Luo, Xiangyang 3391
Luo, Yong 427
Luo, Zhipeng..... 2985
Lutu, Andra 1532
Lyu, Shanshan 1198
M. Altenburger, Kristen..... 263
Ma, Fenglong 3041
Ma, Hao..... 1509, 3049, 1095
Ma, Shaoping 1187, 1210, 1977, 2103, 2844
Ma, Weizhi..... 1187, 1210, 1977
Ma, Xiaojuan..... 1222, 1853
Ma, Yao..... 417
Ma, Yiming..... 3116, 1234
Ma, Zane 840
Maciejewski, Ross..... 2344
Mackenzie, Joel..... 1245
Makhijani, Rahul..... 3056
Mami, Mohamed Nadjib..... 3574
Mancuso, Vincenzo 1532
Maneriker, Pranav 1897
Manner, Jukka..... 2517
Mao, Jiaxin..... 2103
Mao, Jieming..... 751, 762
Mao, Kaili..... 3321
Mao, Yishun 2448
Mao, Yongyi 2929
Maragh, Raven..... 2957, 3063, 3507

Marina, Mahesh 1724
Marincenko, Maija 372
Marinho, Zita 3579
Markatos, Evangelos 1175, 1432
Markopoulou, Athina 2536
Markov, Ilya 3279
Martens, Wim 127, 3503
Mascolo, Cecilia 648
Mason, Joshua 840
Masood, Rahat 2851
Mastrostefano, Enrico 105
Matic, Srdjan 3072
Mazurek, Michelle 1920
McAuley, Julian 1343
McCoy, Damon 372
Mehmood, Qaiser 1623
Mehrotra, Rishabh ... 1256, 2594, 2680
Mehta, Sneha 3079
Mei, Qiaozhu 251
Mejer, Avihai 162
Mellia, Marco 1532
Melo, Philipe 818, 3293
Mendes, Afonso 3579
Meng, Helen 972
Meng, Wei 207
Meng, Zaiqiao 3377
Mensah, Samuel 2929
Merchant, Arpit 3086
Messias, Johnnatan 818
Metzler, Donald 2785
Migliatti, Pedro 3507, 3511
Mikolov, Tomas 3420
Miller, Andrew 840, 1268
Miranda, Sebastiao 3579
Mirian, Ariana 1279
Mirrokni, Vahab 405
Mirza, Paramita 1818
Mislove, Alan 1920, 3579
Mittal, Prateek 1702
Miz, Volodymyr 1290
Mohseni, Sina 3600
Molli, Pascal 1268
Moniruzzaman, A B M 3093
Monz, Christof 2879
Moon, Soo-Mook 2215, 3511, 1567
Morio, Gaku 3545
Morris, Meredith Ringel 549
Moscicki, Angelika 372
Moshfeghi, Yashar 1300
Mota, Nuno 2623
Mueen, Abdullah 2529
Muhlstein, Larry 1343
Mukherjee, Subhabrata 1310
Murai, Shogo 1321, 1332
Murley, Paul 840
Musen, Mark 2708
Naaman, Mor 2169, 3237
Nagatani, Koki 3101
Najork, Marc 4, 627, 795, 886
Nakano, Teppei 3187
Nayak, Richi 3093, 3555
Ngonga Ngomo, Axel-Cyrille 1623
Ni, Jianmo 1343, 3244
Nilizadeh, Shirin 1106, 1702, 2792, 3108
Niu, Jianwei 1724, 3321, 3579
Novgorodov, Slava 1354
Novikov, Sergey 627
Noy, Natasha 1365, 1376
Nuno, Natalie 3511
Nurmi, Petteri 2517
Ogunlana, Foluso 917
Ohkuma, Tomoko 3101
Olson, Lindsey 2601
Ordozgoiti, Bruno 1387
Ostapuk, Natalia 1398
Ou, Wenwu 3123
Ouyang, Jihong 939
Ouyang, Wentao 1198
Overgoor, Jan 1409
Ozcaglar, Cagri 3116
Pärssinen, Matti 1454
Pal, Aditya 2236
Pan, Feiyang 1421, 995
Pan, Hanxiao 2290, 3455
Panisson, Andr?? 2572
Papadopoulos, Panagiotis 1432
Paranyushkin, Dmitry 3584
Park, Dae Hoon 1443
Park, Sunju 807
Parthasarathy, Srinivasan 1897
Pastor, Antonio 1454
Pathak, Jyotishman 514
Paudel, Bibek 2366
Pavlinovic, Zvonimir 1702
Pavlovic, Ekaterina 3511
Peddamail, Jayavardhan Reddy 2203
Pei, Changhua 3123, 3391
Pei, Shichao 1465, 3130, 3137
Peng, Qu 2830
Peng, Richard 961, 3384
Pentyala, Shiva 3600
Pereira, Olivier 1579
Perera, Dilruk 3144
Perozzi, Bryan 37, 394
Pham, Kien 1476
Piao, Ailing 2192
Pierson, Emma 2999
Polakis, Jason 3063
Pollick, Frank 1300
Popescul, Alexandrin 627
Potthast, Martin 670, 1735
Primault, Vincent 1487
Prokhorenkova, Liudmila 1498, 3151
Psounis, Konstantinos 2536
Psyllidis, Achilleas 1521
Pujara, Jay 1944, 3507
Qi, Jianzhong 2436
Qian, Zhiyun 2483
Qiao, Fang 1245
Qiao, Zhilei 3441, 2312
Qin, Lu 1130, 2785, 2247
Qin, Zhiwei 983
Qiu, Jiezhong 1509, 2644
Qiu, Minghui 2630
Qiu, Sihang 1521
Qu, Bingqing 2147
Qu, Meng 1073, 2494
Qu, Xiaoru 60
Qu, Yanru 2644
Rücklé, Andreas 3179
Radinsky, Kira 1354, 2799, 3600
Raghavan, Hema 1679

Rahman, Hafeezul....340
Rahman, Md Mustafizur....3158
Raiber, Fiana....162
Rainshmidt, Elad....162
Rajan, Vaibhav....3172
Rajasegaran, Jathushan....1532, 3165
Rajtmajer, Sarah....3272
Rakkappan, Lakshmanan....3172
Ramakrishnan, Naren....3079
Rangwala, Huzefa....3079
Ranieri, Juri....372
Ranjan, Rajiv....3555
Ranu, Sayan....1886, 2258
Rappaz, Jérémie....1544
Ray Chowdhury, Jishnu....1555, 2673
Reddy, Chandan K.....2472, 3441
Redi, Miriam....1567
Redmiles, Elissa M.....1920
Ren, Pengjie....1095, 2879
Ren, Weijun....1073, 1210, 2355
Ren, Zhaochun....1095, 2323
Resende, Gustavo....818
Restelli, Marcello....1290, 1376, 3258
Ringel Morris, Meredith....3590
Rochet, Florentin....1579
Rodriguez, Luke....3605
Romero, Daniel....318, 3137, 972
Rosenburg, Chuck....2236, 3194
Ruggieri, Salvatore....1051
Rundensteiner, Elke....2936
Sain, Joy Prakash....514
Saini, Shiv Kumar....1601
Saito, Susumu....3187
Salam, Md Abdus....1612
Saldanha, Emily....560
Saleem, Muhammad....1623
Saleh, Hamed....538
Salutari, Flavia....3194
Santos, Aecio....1476, 1634
Sapiezynski, Piotr....1920
Sato, Masahiro....3101
Sattath, Or....2950
Savage, Saiph....1279, 3187
Scells, Harrisen....1646, 3201, 3574
Scheidegger, Carlos....480, 3116
Schockaert, Steven....670, 2460
Schrijvers, Okke....2729, 3579
Seddighin, Masoud....538
Sen, Shilad....1931
Seneviratne, Suranga....3165
Sengupta, Satadal....1657
Serafino, Paolo....917
Shafiq, Zubair....448, 2483, 2957
Shah, Daksh....3086
Shah, Harshay....3208
Shah, Julie....2460
Shan, Huasong....3215
Shan, Liren....961
Shao, Huajie....2192
Shao, Jinjin....2858
Shao, Yunqiu....1187, 2103
Shapiro, Jacob....2608, 3223
Sharma, Prakhar....2601, 3116
Shen, Huawei....1198, 3335
Shen, Jiaxing....1669, 251
Shen, Xiao....329
Shen, Zhihong....2893
Sheng, Victor....3398
Sheng, Ying....886, 514
Shi, Chuan....2022
Shi, Qihao....228
Shi, Shuyang....2999
Shi, Yiyu....3328
Shijia....2865
Shim, Hyunjung....863
Shin, Changhyun....2215
Shin, Kang G.....2806
Shin, Kijung....1679
Shin, Seungwon....2225
Shin, Youjin....3230
Shine, Alana....1691
Shokouhi, Milad....26, 1245, 1702, 2448, 3413
Si, Luo....2270, 3569
Singal, Raghav....1713
Singh, Abhishek Pratap....2258
Singh, Anjali....2943
Singh, Jatinder....3564
Singh, Kamal....3507
Singh, Kapil....594
Singh, Kriti....3493
Singh, Rajkarn....1724
Singh Goud, Jaipal....2915
Sips, Robert-Jan....3286
Skaf-Molli, Hala....1268
Skinner, Will....1735, 3034
Slivkins, Aleksandrs....751
Smadja, Uzi....3237
Smeros, Panayiotis....1747
Smirnova, Alisa....526, 2158, 2169, 2729
Son, Sooel....2225, 3293
Song, Guojie....3314
Song, Junshuai....60
Song, Kaisong....2425
Song, Yangqiu....3370
Sosio, Nicola....1376
Soulie-Fogelman, Fran??oise....2886, 818
Sozio, Mauro....173, 3272, 2971
Staicu, Cristian-Alexandru....1735, 1175
Starov, Oleksii....3244
Steck, Harald....3251
Stepanova, Daria....3516
Stewart, Ian....3258
Stewart, Walter F.....2558
Stones, Rebecca....1153
Stringhini, Gianluca....729, 3072
Strohmaier, Markus....1634, 2708
Su, Dong....594, 2192
Subramanian, Lakshminarayanan....1702
Suchanek, Fabian....1465
Sun, Aixin....1017, 2033
Sun, Changlong....2425, 3123, 2985, 2203
Sun, Jimeng....2558
Sun, Mingming....2715
Sundaram, Hari....3208
Suresh, Amoghavarsha....1759
Swarnkar, Krishnkant....3179
Symul, Laura....2999
Szárnyas, Gábor....1623
Sørensen, Jannick....1590

Tablan, Valentin 3503
Tajima, Keishi 2694
Takanobu, Ryuichi 1771
Tan, Chenhao 318, 2666
Tan, Wang-Chiew 3511
Tang, Haihong 1771, 2494
Tang, Jiaxi 1782, 1509
Tang, Jiliang 417, 3027
Tang, Maolin 3093
Tang, Pingzhong 1421
Tang, Ruiming 1119, 1794
Tang, Xianfeng 1806, 2181
Tang, Zhongyi 1164
Tao, Hemeng 2992
Tarable, Alberto 1724
Taraborelli, Dario 1567
Tarkoma, Sasu 2517
Tata, Sandeep 886
Tatsumi, Tomoyuki 3545, 3377
Thakur, Madhukar 2601
Thirumuruganathan, Saravanan 1612
Thirunarayan, Krishnaprasad 514
Thom, Jennifer 2971, 2999
Thomas, Kurt 372, 1279, 2601
Thuraisingham, Bhavani 503, 2992
Tian, Deyu 1234, 1818
Tikhonov, Alexey 1498, 3151
Timm, Thomas 127
title and chair/organizer names here, Update 3490
Tong, Hanghang 2344
Tonge, Ashwini 1829
Trajcevski, Goce 2750, 3462, 3469, 3476
Tran, Thanh 1841
Trigoni, Niki 1175
Trimponias, George 1853
Trovò, Francesco 1376
Truong, Quoc-Tuan 1864
Tsai, Chia-Wen 3498
Tsai, Ming-Feng 2637
Tsangouri, Christina 2169
Tschantz, Michael Carl 351, 3405
Turk, Ata 1875
Turkoglu, Duru 1875
Turnbull, Bradley 3265
Tyson, Gareth 729, 2851, 3072
Tzamos, Christos 762
Ugander, Johan 263, 1409, 3056
Uhlig, Steve 729
Umar, Prasanna 3272
Urbani, Jacopo 3516
Vachery, Jithin 1886
Vallina, Pelayo 1454
van den Akker, Bram 3279
van den Bercken, Laurens 3286
Vandergheynst, Pierre 1290
Vanvalkenburg, Maryann 2936
Varma, Vasudeva 49, 2915
Vasconcelos, Marisa 818
Vedula, Nikhita 1897
Velcin, Julien 2587
Veldt, Nate 1909
Venkatadri, Giridhari 1920
Venkatasubramanian, Suresh 480, 3108
Vincent, Nicholas 1931
Viswanath, Pramod 2771, 3579
Vo, Nguyen 886
Voelker, Geoffrey M. 1279
Volkova, Svitlana 560
Vtyurina, Alexandra 3590
Vu, Binh 1944
Vu, Xuan-Son 3595
Vucetic, Slobodan 72
Walk, Simon 1634, 2708
Waller, Isaac 1954
Wang, Bai 2022
Wang, Benyou 3293
Wang, Can 228
Wang, Chengyu 1965
Wang, Chenxi 983
Wang, Chenyang 1210, 1977
Wang, Chi 1509
Wang, Chuan-Ju 2637
Wang, Chun-Wen 2814
Wang, Dan 329, 2390, 1153
Wang, Haoyu 1988
Wang, Hongjian 3300
Wang, Hongning 2080, 2872
Wang, Hongwei 2000, 3307
Wang, Huandong 459
Wang, Huazheng 2080
Wang, Jianling 3483, 983
Wang, Junshan 3314
Wang, Kuansan 1509, 2893
Wang, Kun 2865, 3321
Wang, Liang 307, 2366
Wang, Ningxia 240
Wang, Ruchuan 3349
Wang, Shuai 2011
Wang, Tianshi 2192, 2978
Wang, Xiang 151, 2022
Wang, Xiaolan 3511
Wang, Xiaoyu 1806
Wang, Xinbing 784
Wang, Xuanhui 4
Wang, Xueying 3328, 2830
Wang, Yanyan 3526
Wang, Yequan 2033
Wang, Yichen 2236
Wang, Yongqing 1198, 3335
Wang, Yongzhen 2045, 2169
Wang, Yuanzhuo 583
Wang, Zhaoyuan 773
Wang, Zhenghui 2644
Wang, Zhibo 2722, 3027
Wang, Zhongjie 2483
Wang, Zihao 1006
Wang, Zijian 2056
Wardle, Claire 3
Weber, Ingmar 3258, 2652
Wei, Haojie 1106, 3300, 2472
Wei, Xiaochi 1142
Wei, Xu 83, 2181
Weikum, Gerhard 1818, 2673, 3516, 3541
Weinberg, S. Matthew 196
Welton, Randy 514
Wen, Hongkai 1175
Wen, Xidao 2068, 3555
Wen, Zijing 3476
Wendt, James 886, 2091
West, Jevin 3605

West, Robert 681
White, Ryen W. 3590
Whitehouse, Kamin 2872, 1702, 3006
Wilson, Christo 693
Wirth, Anthony 1909
Wong, Kam-Fai 3049, 3230
Wu, Chuhan 3013, 3342, 363
Wu, Fangzhao 3013, 3342
Wu, Fei 1164, 3300, 3349
Wu, Guobin 983, 3462
Wu, Qingyun 2080
Wu, Qitian 2091
Wu, Shu 307
Wu, Xian 717, 2978
Wu, Xintao 3356
Wu, Yongkai 3356
Wu, Yongshang 2999
Wu, Zhiwei Steven 751
Xavier Ferreira, Matheus 196
Xia, Feng 2290, 3363
Xia, Yinglong 705
Xiang, Chongyuan 3034
Xiang, Dongwei 1234, 2865
Xiao, Cao 2558
Xiao, Huiru 3370, 3215
Xiao, Yingyuan 2837, 3560
Xie, Haiyong 1806
Xie, Xiaohui 2103
Xie, Xing 2000, 3013, 3307, 3342
Xie, Yufei 3384
Xin, Liu 83
Xiong, Hui 2312, 3398, 2115, 1782
Xu, Canwen 3391
Xu, Chengfeng 3398, 3363, 3405
Xu, Hu 3413
Xu, Jiajie 3398
Xu, Kan 363
Xu, Kevin S. 829
Xu, Mengwei 2125
Xu, Shizhen 2494
Xu, Shuyuan 2757
Xu, Tong 1106, 2312, 2659, 2792
Xu, Zhangwei 83, 1073, 2652, 2659
Yan, Yaowei 2137
Yan, Yuliang 2045, 2837
Yang, Carl 3420
Yang, Dingqi 1222, 2147, 2158
Yang, Fan 383, 3600, 1398, 2147, 2158
Yang, Keping 240, 363, 2290, 2886, 995
Yang, Longqi 2169, 3493, 3455, 1853, 2822
Yang, Sean 3605
Yang, Songfan 3027, 2858, 2659
Yang, Xinru 3123, 1164, 2666
Yang, Yaodong 983
Yang, Yi-Hsuan 2637
Yang, Yihui 2115
Yang, Yinfei 2630
Yang, Yonghua 240, 459
Yao, Di 273
Yao, Huaxiu 1806, 2181
Yao, Mengfan 3427
Yao, Shuochao 2192
Yao, Tangren 3384
Yao, Ziyu 2203
Yasui, Takaki 1818, 3545
Ye, Jieping 983
Ye, Yanfang 2022, 3448
Ye, Zixin 2103
Yeo, Jihwan 2215
Yi, Yuhao 961
Yikun, Ban 83
Yilmaz, Emine 2323, 2333
Yin, Dawei 417, 717
Yin, Ming 1794
Yin, Ruiping 3434
Yin, Ruiyang 3462
Yitao, Duan 83
Yoon, Changhoon 2225
Yoshida, Yuichi 1321, 1332
You, Jiaxuan 2236
Yu, Cong 638
Yu, Huacheng 2301
Yu, Jinkai 1119, 3130
Yu, Philip S 2022
Yu, Philip S. 3413
Yu, Wenhao 3615
Yu, Wenhui 2247
Yu, Yanwei 1130, 1806, 2644, 3363, 3600, 3620
Yuan, Quan 240
Yue, Xiaoli 3469
Yuen, Chak Fai 2258
Zaccardi, Maria Chiara 1376
Zagheni, Emilio 3258
Zaman, Anis 2270
Zelle, Dustin 37
Zeng, Qingkai 3615
Zeng, Yulong 2281
Zhang, Chao 459
Zhang, Chenwei 3041
Zhang, Chuxu 717, 2011
Zhang, Dongjie 438
Zhang, Dongmei 2659
Zhang, Dongyu 363, 2290
Zhang, Fengli 2750
Zhang, Fuzheng 2000
Zhang, Guangquan 3434
Zhang, Haiqiao 3328
Zhang, Hengrui 2091
Zhang, Hongyang 2301, 2659
Zhang, Huichu 3620
Zhang, Jianjin 2985
Zhang, Jin 2355
Zhang, Jingyuan 950
Zhang, Kunpeng 2750, 3462, 3469, 3476
Zhang, Le 2312, 329
Zhang, Liangjie 2985
Zhang, Long 616, 3356
Zhang, Min 1187, 1210, 1977, 2103
Zhang, Mingyang 459, 795
Zhang, Peiran 2115, 2323, 2333, 2425, 2985, 3101
Zhang, Qinqin 784
Zhang, Qiong 3569
Zhang, Richong 2929
Zhang, Rui 2436
Zhang, Shaodian 2644
Zhang, Si 2344, 2355, 2366, 2687
Zhang, Weinan 784, 2644, 3620
Zhang, Wen 2366

Zhang, Wenjie 1130, 2137
Zhang, Xiangliang 3130
Zhang, Xiao-Yu 307, 2378, 2390, 3441, 3448
Zhang, Yongfeng 1142, 2390, 2757, 2822, 3123
Zhang, Yunpeng 3215
Zhang, Yuzhou 1119
Zhang, Zhaohua 1153, 2401
Zhang, Zhenyu 438
Zhang, Zhongzhi 961, 2413
Zhao, Eric 417
Zhao, Jiashu 717
Zhao, Kai 491
Zhao, Kaiqi 427
Zhao, Lujun 2425
Zhao, Mengchen 616
Zhao, Miao 2000, 3307
Zhao, Mingjun 1106
Zhao, Pengpeng 3398
Zhao, Qi 886
Zhao, Yiran 2192
Zhao, Yunxiang 2436
Zhao, Zhou 1771, 3455
Zheng, Gang 995, 3521, 3610
Zheng, Shuyu 1234
Zheng, Wenguang 773, 3526, 3560
Zhong, Ting 2750, 3462, 3469, 3476
Zhou, Aoying 1965
Zhou, Chang 60, 2750, 3462, 3469, 3476, 2448
Zhou, Jingren 1130, 2630, 3349
Zhou, Sheng 228, 2722
Zhou, Xiaofang 3398
Zhou, Yilun 2460
Zhou, Yunhong 2978
Zhou, Zihan 3620
Zhu, Changlei 2312, 2448
Zhu, Hai 2366
Zhu, Hengshu 2312
Zhu, Hongbo 2115, 2472
Zhu, Shitong 2483
Zhu, Xiaoyan 2033, 2401, 2448, 3620
Zhu, Zhaocheng 2494
Zhu, Zhihua 273
Zhu, Ziwei 3483
Zhuang, Fuzhen 1421
Zhuang, Tao 1771
Zhuang, Yueting 2666
Zimmermann, Roger 3144
Zohar, Aviv 2950
Zois, Daphney-Stavroula 3427
Zomaya, Albert 2505, 3555
Zuccon, Guido 1646
Zuniga, Agustin 2517
Zuo, Song 2281

OUR PARTNERS AND SPONSORS

Diamond



Gold







Silver













Bronze





























Others









The Web Conference 2019 was organized by Web4Good, a California 501(c)(3) non-profit organization.